



magazine Embedded

EMBEDDED SOLUTIONS FOR PROGRAMMABLE LOGIC DESIGNS



**Unlock the
Power of Xilinx
Programmability**

INSIDE

**Accelerating PowerPC
Software Applications**

**Nucleus Integration with
Xilinx FPGA System Design**

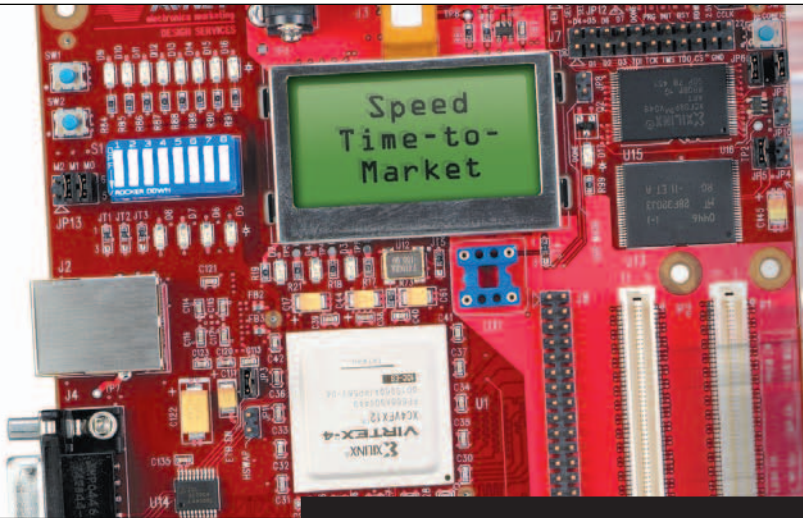
**Take Electronic Motor Drives
to the Next Level**

**Build and Optimize a MicroBlaze
Soft-Processor System Your Way**

**Building Reliable and Upgradable
Software-Defined Radios**

 **XILINX®**

Support Across The Board.™



Get Started Now with Xilinx® Virtex-4™ FPGAs

Virtex-4™ FPGA Features:

- Multi-platform FPGA family
- Support for three application domains
- 90 nm process technology
- Reduced power consumption
- Reduced cost per function

Virtex-4™ FX12 Evaluation Kit Features:

- Virtex-4™ FX12 FPGA
- 4 MB Flash and 32 MB DDR SDRAM
- 128x64 OSRAM® graphical display
- National Semiconductor® DP83847 10/100 Ethernet PHY
- VHDL source code for sample design

Xilinx® is revolutionizing the fundamentals of FPGA economics with the Virtex-4™ family. To help you get a jumpstart on your next design, Avnet Electronics Marketing has created the Virtex-4 FX Evaluation Kit. This evaluation kit delivers a low-cost, feature-rich platform to develop and test designs targeted to the Virtex-4 FX subfamily.

Special Offer Available Exclusively to Qualified ESC Boston Attendees

Visit the Xilinx booth [#501] to learn how you can qualify for special ESC pricing on a Virtex-4 FX12 Evaluation Kit from Avnet. Use this kit to get hands-on experience with the powerful embedded processing solutions presented at the Xilinx booth.

For more information about the kit, go to em.avnet.com/virtex4fxesc

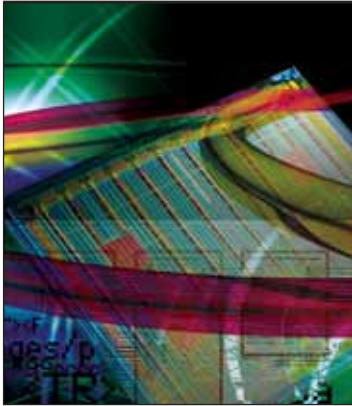
To learn how to qualify for special pricing, visit Booth #501 at ESC Boston.



Enabling success from the center of technology™

1 800 332 8638
www.em.avnet.com





C O N T E N T S

Welcome4

ARTICLES

Faster and More Flexible Embedded Systems.....5
 Accelerating PowerPC Software Applications.....8
 Hardware/Software Co-Verification14
 Moving Embedded Systems onto FPGAs.....17
 Tracing MicroBlaze Processors with a Logic Analyzer20
 Nucleus Integration with Xilinx FPGA System Design.....24
 Building Reliable and Upgradable Software-Defined Radios30
 Take Electronic Motor Drives to the Next Level34

CUSTOMER SUCCESS

Xilinx and Birger Engineering38
 Xilinx and LG Electronics40
 Xilinx and Photonic Bridges.....42

APPLICATION NOTES

Embedded Application Notes46

TechOnLine

Build and Optimize a MicroBlaze Soft-Processor System Your Way.....50

PRODUCT BRIEFS

PowerPC and MicroBlaze DK for Virtex-4 FX12 Edition54
 MicroBlaze DK Spartan-3 SP305 Edition56

BOARDS

More Integration, Easier Development58

EDUCATION

Embedded Systems Development.....62
 Advanced Features and Techniques of Embedded Systems Development.....63

EDITOR IN CHIEF
Carlis Collins
editor@xilinx.com
408-879-4519

MANAGING EDITOR
Forrest Couch
forrest.couch@xilinx.com
408-879-5270

ASSISTANT MANAGING EDITOR
Charmaine Cooper Hussain

XCELL ONLINE EDITOR
Tom Pyles
tom.pyles@xilinx.com
720-652-3883

ADVERTISING SALES
Dan Teie
1-800-493-5551

ART DIRECTOR
Scott Blair

www.xilinx.com/xcell/embedded



Xilinx, Inc.
2100 Logic Drive
San Jose, CA 95124-3400
Phone: 408-559-7778
FAX: 408-879-4780

© 2005 Xilinx, Inc. All rights reserved. XILINX, the Xilinx Logo, and other designated brands included here-in are trademarks of Xilinx, Inc. PowerPC is a trademark of IBM, Inc. All other trademarks are the property of their respective owners.

The articles, information, and other materials included in this issue are provided solely for the convenience of our readers. Xilinx makes no warranties, express, implied, statutory, or otherwise, and accepts no liability with respect to any such articles, information, or other materials or their use, and any use thereof is solely at the risk of the user. Any person or entity using such information in any way releases and waives any claim it might have against Xilinx for any loss, damage, or expense caused thereby.

Unlock the Power of Xilinx Programmability

Strategic shifts in market dynamics are one of the most important aspects to consider when designing new products. I've experienced a number of situations where our teams were ahead of the curve and able to exploit these shifts to market advantage. One example that comes to mind is the adoption of surface-mount packages. We built surface-mount devices and justified this to management for a number of quarters while waiting for customers to appear. When the shift occurred, it was as though every customer made the change on the same day.

Embedded processing FPGAs are another fundamental strategic shift in market dynamics that will have profound effects for both FPGA and embedded processor vendors, and the basis of major growth for both Xilinx® and our customers.

Three key forces are driving the opportunity for embedded processing in FPGAs. First, customers are increasingly requiring CPU functionality in our offerings. Second, pure-play embedded suppliers are finding that they can no longer serve all of the needs customers have for their products due to rising development costs. Third, processor vendors have realized that performance improvements can no longer come from simply increasing clock rate, and must deploy parallelism, an inherent FPGA capability.

One of the greatest appeals of our FPGA embedded solution is the ability to easily stitch together the CPU, buses, and common-commodity peripherals, creating a "just-what-I-need" processor subsystem that "just works." The customer can then concentrate on the secret sauce that differentiates their products.

To win the hearts and minds of customers and capitalize on the embedded strategic market dynamic, the development tools and complete embedded solution must be closely tied to the typical environment our customers are used to. This includes OSs, compilers, debuggers, development boards, software development kits (SDKs), and board support packages (BSPs), which all must be in a familiar framework.

With that said, in our second issue of *Embedded Magazine* we have included a collection of articles from Xilinx and our partners that provide insight to our progress in the strategic shift of the market dynamics I've discussed. Partners such as Impulse C, Accelerated Technologies, QNX, and Altium highlight the latest advances of both their well-known and new solutions for our embedded platforms. We've also included a discussion of the latest advances with our 32-bit MicroBlaze™ soft processor and its new integrated option for an IEEE-compatible floating-point unit.

The latest application advances with the Virtex™-4 FX FPGA and immersed PowerPC™ are contained in the APU accelerator piece, as well as summaries of two remarkable Xilinx application notes. Multiple articles describe various aspects of the integrated design environment with Xilinx Platform Studio and the Embedded Development Kit (EDK).

Finally, we have aligned the release of *Embedded Magazine* with the Embedded Systems Conference. We have another exciting lineup at the conference, featuring free hands-on workshops for you to get familiar with our Platform Studio tool suite, PowerPC, and MicroBlaze solutions as well as our industry-leading silicon of Spartan™-3 and Virtex-4 families. Be sure to visit us at the Hynes Convention Center, booth 501, in Boston from September 12-14 to meet our embedded experts.

I believe that we are in the midst of the next strategic shift with embedded processing FPGAs. I hope you find our second edition of *Embedded Magazine* informative and inspiring and invite you to unlock the power of Xilinx programmability. The advantages are enormous.



Mark Aaldering

Vice President
Embedded Processing
& IP Divisions

Faster and More Flexible Embedded Systems

Programmable Platform FPGA devices and intelligent tools combine to create higher performance processing solutions.

by Jay Gould
Product Marketing Manager,
Xilinx Embedded Solutions Marketing
Xilinx, Inc.
jay.gould@xilinx.com

Copyright © 2005 Xilinx, Inc. All rights reserved.

Are there any real-time computing requirements that don't mandate that your new systems be faster and more flexible than your previous designs? Ever-changing industry standards dictate that tomorrow's embedded system designs accommodate a new level of customization, while high performance demands challenge traditional processing designs.

You don't want to be restricted by an overly customized design, and you can't just keep toggling the system clock at ever-increasing speeds to improve performance. There has to be a better way to create faster and more flexible embedded processing systems.

Platform FPGAs are programmable SOCs that support a multitude of sophisticated designs, and include on-board memory, DSP capability, embedded processing, and hardware accelerated co-processing. The re-programmability and field upgradeability of these new devices mean that you can fix bugs, enhance features, optimize performance, and add emerging industry standard support throughout product life cycles and even after deployment in the field.

With these powerful capabilities immersed in a programmable SoC device, all you need are the appropriate tools to unleash and harness this embedded performance.

Intelligent Tools

In recent industry surveys, design engineers made it clear that they often value intelligent tools more than the actual devices and operating systems they use to build their own end products. If this trend is accurate, choosing the appropriate tool suite before beginning your next embedded system design will be crucial to your product schedule and overall success.

Today's development environments need to provide "platform-aware" tools that understand all system options and support multiple types of processor cores, as well as the creation and customization of co-processing and IP. Design wizards and automatic module



generation will reduce errors and streamline the development process, while integrating hardware and software debuggers together will enable you to find and fix bugs faster. If you choose wisely, intelligent tools will accelerate development and optimize performance.

Standard Flows and Innovation

Xilinx® created the Xilinx Platform Studio (XPS) tool suite with the development of Platform FPGAs in mind, supporting existing traditional flows for both hardware (HDL/netlists for FPGAs) and embedded

hard- and MicroBlaze™ soft-processor core designs, the tools are smart enough to remove MicroBlaze options if you have chosen PowerPC, and vice versa.

Importing, creating, and customizing IP is streamlined through a separate design wizard, and supports an IP repository to facilitate IP reuse elsewhere on this design or in the future on a different design.

XPS additionally innovates and accelerates the development process with a variety of automatic generators that replace tedious and error-prone manual design steps. Being aware of the Platform FPGA

ware debug capabilities through a single probe. Other traditional methods require multiple probes and switching hardware connections between the different steps.

In fact, XPS uniquely integrates the hardware and software debuggers together so that they can cross-trigger each other. This new visibility into the system allows embedded design teams to find and fix bugs faster, regardless of whether the flaws originate in hardware or software.

Acceleration Through Co-Processing

Let's say that you now have a flexible processor-based platform that satisfies most of your system requirements. How fast can you clock the core to meet your performance requirements?

You probably have realized that clocking your processor faster won't take care of all of your performance challenges. Besides the physical limitations of discrete processors and heat dissipation, accelerated clocking can't ensure that your core can service and complete all the real-time event responses and applications with which you have burdened it. More and more "multi-processor" solutions are emerging to partition and offload lower priority tasks from a main control processor so that the main unit can ensure real-time responses.

Programmable platforms introduce some additional ways to approach this problem, with off-the-shelf devices that you can customize yourself for your own unique applications. Supporting both hard and soft processor cores, one solution offered by Platform FPGAs is to focus high-priority tasks on an immersed hard processor while offloading lower priority tasks to a soft-processor core instantiation. You have the option to add one or more MicroBlaze soft processors to a Platform FPGA device already running an embedded PowerPC engine. Example devices supporting this are the Virtex™-II Pro FPGA or new Virtex-4 FX family devices with built-in PowerPCs. The PowerPC cores in these devices can be complemented with MicroBlaze IP cores inserted as macros and built out of FPGA hardware resources in the silicon.

Another alternate and promising approach is to implement the concept of

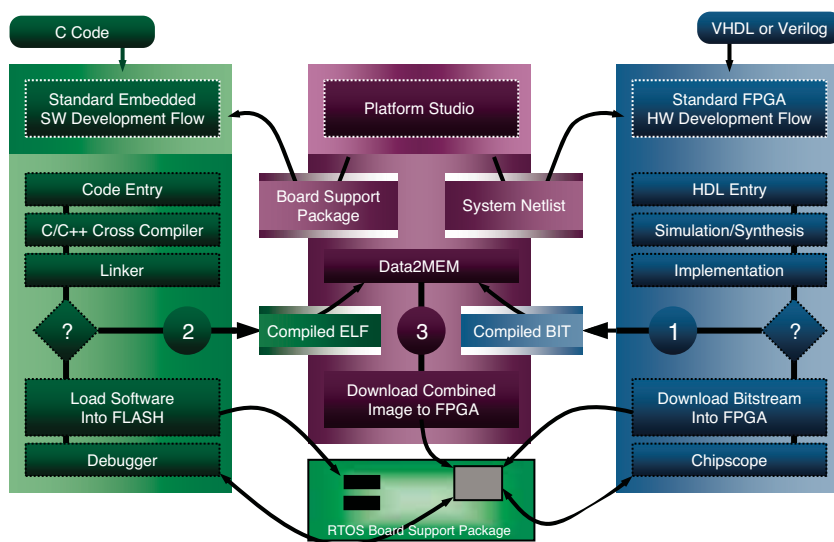


Figure 1 – XPS design flow

software design (C/ELF code for processing core engines) (Figure 1). In addition to providing a unified development tool suite for supporting the complete spectrum of programmable processing solutions, XPS won the IEC's (International Engineering Consortium) DesignVision Innovation Award for introducing new capabilities that accelerate embedded development. With platform-aware tools like XPS, you can quickly create a real-time hardware/software system through an abstract flow of design wizards.

The design wizards guide you through the process of creating a basic system and can reduce errors by masking-off design options not supported by your initial selections and assumptions. For example, although XPS supports both PowerPC™

silicon properties and options, XPS can automatically generate software drivers for selected peripherals, generate sample test code for board options, and even create BSPs (board support packages) for some of the more widely used RTOS/eOS (real-time operating systems/embedded operating systems) such as Wind River Systems's VxWorks or embedded Linux.

XPS also provides a unique utility (Data2Mem) that merges C code into the FPGA bitstream, enabling software development and debug to proceed in real time without time-consuming re-runs of FPGA place and route tools.

Xilinx even provides new efficiencies with a unified JTAG connection methodology that combines FPGA download, FPGA debug, C code download, and soft-

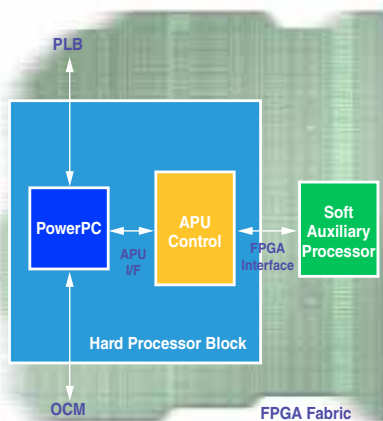


Figure 2 – Virtex-4 FX APU

“co-processing” and use the intelligent tools to build a direct connect from the embedded PowerPC cores to high-performance FPGA fabric, where hardware accelerator functions can operate as extensions to the PowerPC. As shown in Figure 2, you can improve the overall system performance by offloading computationally demanding applications from the main CPU.

By its very nature, FPGA hardware fabric is parallel in structure and can be used to accelerate system functions orders of magnitude faster than clocking methods can provide. In this example, the PowerPC core is complemented by an APU (Auxiliary Processor Unit), which inter-

faces to a parallel soft processor that can handle applications such as data processing, floating-point mathematics, and video processing. This direct connection provides a high-bandwidth, low-latency solution with parallel advantages over other multi-core processor and arbitrated busing solutions.

Performance Analysis

Do you need to find out where your performance is lost in your design? Embedded software debugging and analysis is always a bit of a challenge because code execution is often “invisible” to you. On paper, your design looks like it meets specifications, but when running in real-time hardware with asynchronous interrupts and real-world situations, you find that often you don’t meet your own performance requirements. Now is the time when intelligent tools can provide you with a unique view inside the operating device rather than leave you guessing outside of a black box.

Version 7.1 of Xilinx Platform Studio introduces a series of performance analysis tools and views that provide great insight as to how your software is actually executing and where performance is leaking away from you (Figure 3). By knowing which software functions take up the most execution time and which functions call other functions – as well as the number of times called – you can

get an illuminating view of exactly how your embedded design is running. Functions that take a long time to execute, or functions that are called a large number of times by other routines, may be excellent candidates to accelerate by moving them to parallel hardware as co-processing extensions.

Figure 3 also shows that if the tools track and display your software execution clearly, you can quickly and easily identify areas that could be more efficient. This can save a lot of what-if experiment scenarios that are time-consuming and often result in relatively small performance improvements. In-lining some C code or an entire function may provide tiny localized speed-ups, but moving time-consuming routines into high-performance FPGA hardware can often result in an order-of-magnitude improvement. With intelligent views of the code execution by specific function names, you can see exactly which software routines to adjust, providing a much higher return on improving system performance.

Conclusion

Intelligent platform-aware tools can help you identify the inefficiencies in your embedded software code and allow you to optimize performance. Knowing which specific software functions you need to streamline allows you to evolve your hardware/software partitioning and accelerate more modules in programmable FPGA fabric.

The high-performance nature of parallel FPGA hardware resources and the advent of easy-to-use, programmable co-processing technologies like the Virtex-4 FX APU enable you to create faster and more flexible embedded processing systems.

Xilinx offers clear advantages for embedded processing over traditional discrete or competitive FPGA solutions. Our tools, combined with our programmable embedded Platform FPGAs, offer a significant performance improvement for real-time developers.

To learn more about the Platform Studio tool suite, please visit www.xilinx.com/edk. A good starting point to learn about all of our embedded processing solutions is www.xilinx.com/processor.

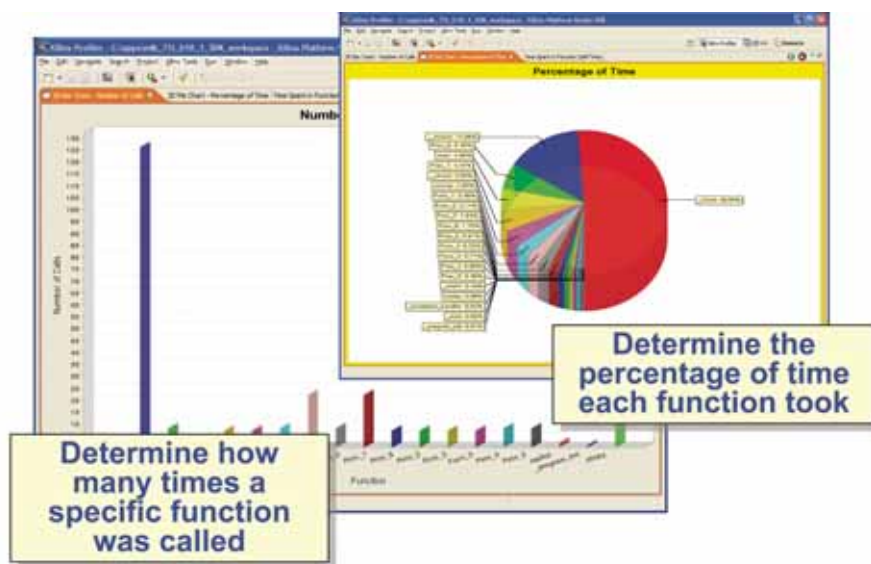


Figure 3 – XPS performance analysis views

Accelerating PowerPC Software Applications

Using custom APU peripherals, C-to-hardware tools enable fast creation of Virtex-4 hardware accelerators.

David Pellerin
Chief Technology Officer
Impulse Accelerated Technologies, Inc.
david.pellerin@impulsec.com

Greg Edverson
Senior Software Engineer
Pico Computing, Inc.
greg@picocomputing.com

Kunal Shenoy
Design Engineer
Xilinx, Inc.
kunal.shenoy@xilinx.com

Dan Isaacs
Director, Embedded PowerPC Marketing, APD
Xilinx, Inc.
dan.isaacs@xilinx.com

Copyright © 2005 Xilinx, Inc. All rights reserved.

The Xilinx® Virtex™-4 FX family of FPGA devices provides embedded systems developers with new alternatives for creating high-performance, hardware-accelerated applications. With an integrated industry-standard PowerPC™ processor and innovative Auxiliary Processor Unit (APU) interface, the Virtex-4 FX device allows system designers to efficiently connect custom hardware accelerators to the integrated processor, yielding unprecedented performance.

In the past, software programmers who wanted to take advantage of FPGAs for algorithm acceleration have experienced significant technical barriers because of the complexity of writing low-level hardware descriptions to represent higher level software functions.

In this article, we'll show how the power of the Virtex-4 FX FPGA can be made readily available to embedded systems designers and software programmers through the use of software-to-hardware tools. The emergence of such tools bring the performance benefits of FPGAs to anyone who can program in C. Accelerated FPGA-based designs are now easier and more practical for a wide range of application domains, including image processing, DSP, and data encryption.

From Software to FPGA Hardware

By virtue of their massively parallel structures, FPGAs have the potential to dramatically accelerate embedded software applications. But because these devices require different (hardware-oriented) skills than traditional processors, the creation and programming of a system based on an FPGA has remained challenging for all but the most hardware-savvy software programmers. This is changing, however. With the introduction of simplified FPGA-based computing platforms and streamlined tools for platform building, most of the barriers to FPGA adoption have been removed. In addition, the introduction of software-to-hardware tools for FPGAs has dramatically improved the practicality of these devices as software-programmable computing platforms.

The tools that make this shift possible – enabling FPGA-based platforms to be considered viable alternatives to traditional processors for embedded systems – serve two basic needs. At the front end, software-to-hardware compiler tools accept high-level descriptions written in a language familiar to embedded software programmers. The de facto standard for embedded systems design is standard C, with C++ and Java beginning to make inroads as well.

At the back end, existing synthesis and place and route technologies are combined with system-level platform building tools, allowing designers to develop and target complete systems on programmable logic to specific development boards. Both of these needs are being met today, by tools currently available.

In the area of software-to-hardware compilation, compiler tools such as Impulse

CoDeveloper (Figure 1) can simplify the generation of FPGA hardware from higher level C-language descriptions of software algorithms. These tools provide the necessary bridge between the domains of software programming and lower level hardware design.

Serving the need for platform building tools is Xilinx Platform Studio, which supports a wide variety of Xilinx FPGA-based boards and systems. Platform Studio makes it practical for an embedded systems designer – who may have little or no expe-

rience with low-level FPGA design – to assemble a complete system within a single FPGA, including one or more processors and related peripherals. When these tools are combined with a platform-aware software-to-hardware compiler, the complete system can include custom accelerators originally written in C.

Accelerating Embedded Applications

The Virtex-4 FX family of devices provides an ideal platform for hardware acceleration of embedded applications. The Virtex-4 FX12 FPGA, for example, includes more than 12,000 logic cells; an integrated PowerPC 405 core, which can operate at speeds as fast as 450 MHz; and dual 10/100/1000 Ethernet MACs, configured by the processor through the device control register (DCR) interface or through the FPGA fabric.

Looking inside the embedded processor block (see Figure 2), the PowerPC 405 CPU is directly coupled to the unique and innovative APU controller, which provides direct access to hardware accelerators implemented in the FPGA logic. The APU controller supports three classes of instructions: PowerPC floating-point instructions, APU load and store instructions, and user-defined instructions (UDI). UDIs are program-

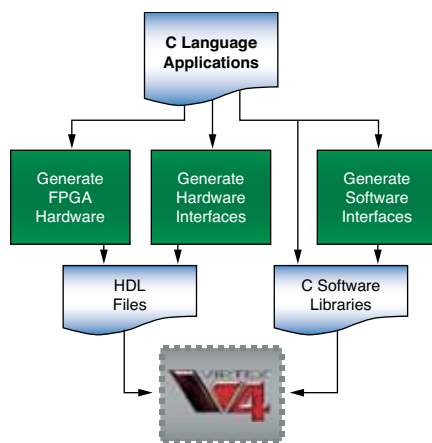


Figure 1 – Impulse CoDeveloper tools simplify the conversion of C subroutines to lower level FPGA logic and provide the necessary software-to-hardware communications on the Virtex-4 platform.

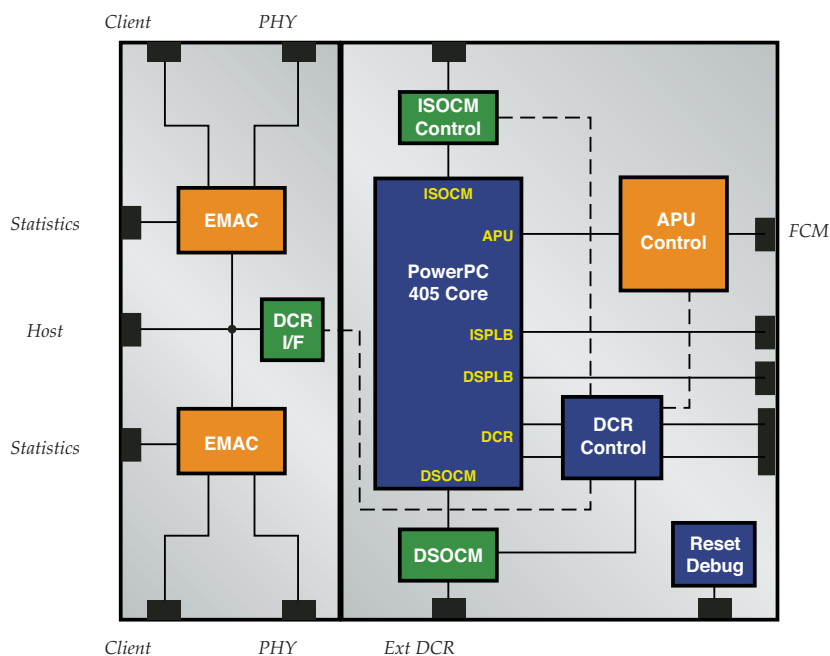


Figure 2 – The Virtex-4 FX12 embedded processing block includes the embedded PowerPC 405 processor and a high-performance APU interface, along with dual 10/100/1000 EMACs.

grammed into the APU controller either dynamically through the PowerPC 405 via the DCR bus or statically during FPGA configuration via the bitstream. The APU supported instructions are executed by hardware acceleration co-processing engines implemented in the FPGA logic.

When packaged in a highly integrated

compact device such as the Pico Computing E-12 card (see sidebar, “A Wide Range of Development Platforms”), the FX12 device becomes a complete embedded development platform that requires little or no hardware design expertise. For embedded application developers requiring a wider range of hardware

peripherals (such as direct access to video and audio signals), the Xilinx ML403 board, also based on the FX12 device, provides an excellent embedded systems development platform.

On-chip, the Virtex-4 FX APU controller provides a flexible high-bandwidth interface between the FPGA fabric and the

Taking Advantage of Parallelism in FPGAs

A key aspect of any software-to-hardware design flow is the use of parallelism to increase performance. When accelerating C applications using FPGAs, parallelism can be exploited at two distinct levels: at the application system level and at the level of statements (or blocks of statements) within a specific subroutine or loop.

Although there are ongoing attempts to create compiler technologies that can exploit both levels of parallelism with a high degree of automation, the best approach today is to focus automation efforts (represented by the software-to-hardware compiler) on the lower level aspects of the problem, while at the same time providing software programmers an appropriate and easy-to-use programming model that allows higher level, coarse-grained parallelism to be expressed. In this way programmers can make hardware/software partitioning decisions and experiment with alternative algorithmic approaches, leaving the task of low-level optimization to automated compiler tools. This approach is particularly useful for platforms such as the Virtex-4 device that include embedded processors.

A number of programming models can be applied to FPGA-based programmable platforms, but the most successful of these models share a common attribute: they support modularity and parallelism through a dataflow-like method of design partitioning and abstraction. Communicating sequential processes, or CSP, is one such programming model. CSP has proven to be highly effective in expressing application-level parallelism for FPGA targets. This programming model is directly supported in the Impulse C tools provided by Impulse Accelerated Technologies, Inc.

At the heart of the Impulse C programming model are processes and streams (Figure 4). Processes are independently synchronized, concurrently operating portions of an application that are written in a standard language (in this case C language). Processes perform the work of the application by accepting data, performing computations, and generating relevant outputs.

Unlike traditional C subroutines, processes are considered persistent; they are normally called once (whether in hardware or software) and continue as long as there is streaming data to be processed. The data processed by such an application flows from process to process by means of streams, or in some cases by means of messages or shared memories, which are also supported in the programming model. Streams represent one-way channels of communication between concurrent processes and are self-synchronizing with respect to the processes by virtue of buffering. The primary method of synchronization between processes is therefore the data being passed on the streams.

The key to allocating processing power within such a system – and using such a programming model – is to implement one or more processes in the FPGA to handle the heavy computation, and implement other processes on embedded or external microprocessors to handle file I/O, memory management, system setup, and other non-performance-critical tasks. Using tools such as those included with Impulse C, an application comprising multiple parallel C processes can be modeled entirely in software, verified using a standard desktop C debugging environment, and then, after the application is functionally complete, incrementally moved into the FPGA for further optimization and acceleration.

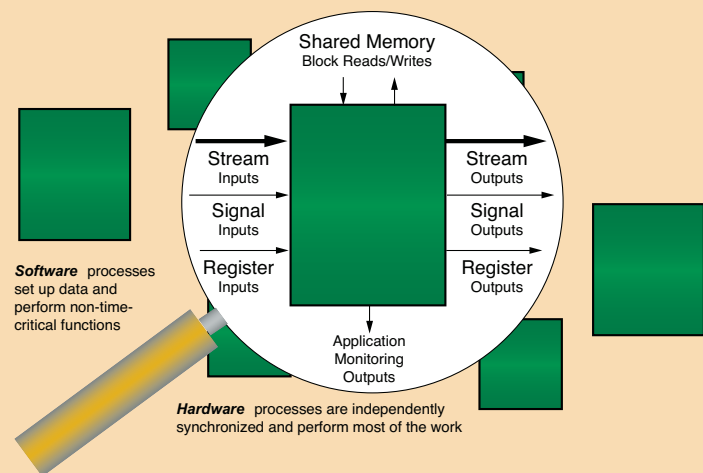


Figure 4 – The Impulse C programming model emphasizes the use of processes, streams, and shared memories for hardware/software partitioning.

pipeline of the on-chip PowerPC. Fabric co-processor modules (FCMs) implemented in the FPGA fabric are connected to the embedded PowerPC processor through the APU interface, allowing the creation of custom hardware accelerators. These hardware accelerators operate as extensions to the PowerPC, thereby offloading the CPU from demanding computational tasks.

Software engineers can access the FCM from within assembler or C code. Assembler mnemonics are available for user-defined instructions and pre-defined load/store instructions, enabling programmers to invoke hardware-accelerated functions into the regular program flow. Programmers can also define custom instructions designed specifically for the hardware functionality of the FCM. When combined with C-to-hardware compiler tools, the APU controller allows software programmers to create hardware-accelerated software applications with little or no FPGA design expertise.

C-to-Hardware Tools Increase Design Productivity

To make productive use of any computing platform, software programmers need appropriate compiler and debugging tools. Impulse C, from Impulse Accelerated Technologies, gives software programmers access to FPGAs by allowing hardware accelerators to be compiled directly from software descriptions.

These accelerators, which are typically represented by one or more software subroutines, are automatically compiled into efficient, high-performance hardware that can be mapped directly into FPGA gates. In the case of the Virtex-4 FPGA, Impulse C is also capable of automatically generating software/hardware interfaces using the APU. This is particularly useful for applications that combine both traditional and FPGA-based processing.

Because it is based on standard C, Impulse C allows FPGA algorithms to be developed and debugged using popular C and C++ development environments, including Microsoft Visual Studio and

GCC-based tools. The CoDeveloper software-to-hardware compiler translates specific C-language subroutines to low-level FPGA-hardware (see Figure 3) while optimizing the generated logic and identifying opportunities for parallelism. The compiler is also capable of unrolling loops and generating loop pipelines to exploit the extreme levels of parallelism possible in an FPGA. Instrumentation and monitoring functions generate debugging visualizations for highly parallel multi-process applications, helping system designers identify dataflow bottlenecks and other areas for acceleration.

For applications involving the embedded PowerPC and MicroBlaze™ processors, the Impulse C compiler automates the creation of hardware/software interfaces and generates outputs compatible with Xilinx Platform Studio. This makes it possible to create high-performance, mixed hardware/software applications for FPGA-based platforms without the need to write low-level VHDL or Verilog.

For large applications comprising multiple hardware and software elements, Impulse C includes interface libraries (see sidebar, “Taking Advantage of Parallelism in FPGAs”) and related compiler features, allowing parallelism to be expressed at the

design process is highly iterative, reflecting the fact that decisions made up-front (such as C coding styles and system-level partitioning decisions) may have a dramatic impact on the results obtained after C compilation, synthesis, place and route, and final bitmap generation. At each point in the process, the tools provide feedback, allowing you to evaluate and estimate performance before moving to subsequent (and perhaps more time-consuming) phases of the platform generation process.

Let’s summarize the steps required for a typical PowerPC-based application using Impulse and Xilinx tools:

1. The application is initially written in standard C, using common C development tools. These tools include readily available tools such as Visual Studio, Eclipse, or GCC and GDB, and may also involve more comprehensive cross-development tools. During this phase, a baseline for validation (a software test bench, also written in C) is established, allowing you to quickly test later design iterations.
2. A C profiler such as gprof may be invoked, or other, less sophisticated methods are used to identify computational hotspots. Often these hotspots can be isolated to a few C subroutines or inner code loops requiring acceleration. Application monitoring (made possible by instrumenting the C code during software testing) can help characterize these hotspots and analyze data movement.
3. Using software-to-hardware interface functions provided in the Impulse C library, data streams or shared memories create abstract connections between the main algorithm running on the PowerPC and hardware-accelerated subroutines running in the FPGA. The modified software algorithm, which now includes one or more independently synchronized processes, is simulated again in a standard C environment to ensure its correct behavior.

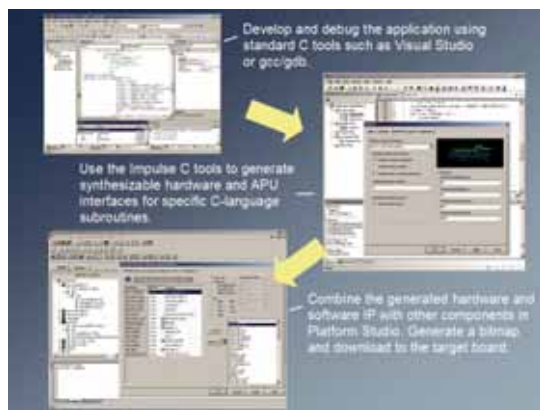


Figure 3 – The C-language-to-FPGA-accelerator design flow

level of multiple and independently synchronized processes. These processes can be mapped either to software running on an embedded PowerPC or MicroBlaze processor, or to FPGA hardware.

For all such applications, integration of front-end compiler tools and back-end platform building tools is important. The

4. C-language subroutines (or processes, as they may now be referred to) representing hardware accelerators are analyzed and optimized by the Impulse C compiler, resulting in hardware description files compatible with FPGA synthesis tools. Optimization reports generated in this phase help you understand the impact of various coding styles, and make appropriate revisions in the original C code for improved performance. During this compilation process, additional compiler outputs are generated that represent hardware-to-software interfaces, including (in the case of the Virtex-4 FPGA) the necessary APU interface logic. Software run-time libraries are also generated at this point, corresponding to the abstract stream and shared memory interfaces specified on the processor side of the application.

5. The generated hardware and software files are exported from the Impulse tools (as a PCORE peripheral) and imported directly into the Xilinx Platform Studio environment.

The stream and shared memory interfaces defined in the C application are mapped to APU, PLB, or other interfaces where appropriate, along with other components (such as standard processor peripherals or non-standard IP blocks) to create the complete system. From within the Platform Studio interface, the entire application (both hardware and software) is built, resulting in a downloadable bitmap.

Evaluating FPGA Acceleration

Using the Pico E-12 card and the Xilinx ML403 development kit, in conjunction with Impulse C and Platform Studio, we set out to compare the relative performance of the embedded PowerPC 405 processor both with and without APU hardware acceleration – and using only C programming techniques. To investigate a range of potential application domains, we selected the following three representative algorithms:

1. An image filter. This algorithm allowed us to evaluate two pipelined hardware

routines for processing a stream of image data. The algorithm chosen for this experiment is a relatively simple 3 x 3 edge-detection function operating on a 512 x 512 image buffer. This algorithm allowed us to quickly evaluate the performance of data streaming through the Virtex-4 APU interface, as well as the potential speedups of using multiple, pipelined hardware processes.

2. A triple-DES encryption engine. This algorithm allowed us to evaluate the impact of various C-level optimization strategies, as well as the practicality of adapting and optimizing legacy C code for a streaming programming model. One million character blocks (of eight characters each) were processed to obtain performance numbers for this test.

3. A fractal image generator. This algorithm is computationally intensive and can be characterized in many ways to explore the size/performance space. For this experiment, we created a single hardware process in the FPGA as an APU peripheral. This hardware process communicates with a single controlling software process running on the embedded PowerPC. The design of this algorithm, which generates a 1024 x 768 pixel image with a selectable level of image accuracy, is scalable such that additional hardware accelerator processes can be easily added, up to the limit of the target FPGA.

For each of these algorithms, various combinations of compiler loop unrolling, pipelining, and maximum stage delays

were selected in the Impulse C compiler. In this way the applications could be optimized (in most cases without modifying the original C code) to obtain a desirable balance of size, cycle delays, and maximum clock speed in the generated hardware. In most cases, we determined that using a relatively low clock rate (50 MHz) in the FPGA fabric – in combination with increased cycle-by-cycle throughput (through the use of automated pipelining) – produced the best overall results given the nominal overhead of software-to-hardware data communication.

Using these algorithms as a baseline, numerous tests were performed in which the same C code was compiled both to the FPGA (as an APU accelerator) and to the embedded PowerPC processor as a software-only application. The results of these tests are summarized in Table 1.

As the chart shows, the hardware-accelerated algorithms show an impressive increase in performance, even at reduced FPGA clock rates, compared to the PowerPC software-only version.

Conclusion

In this article, we have demonstrated how it is possible, using an FPGA-based platform and C-to-hardware tools, to create highly accelerated systems without low-level hardware design skills. The Virtex-4 FX device, when implemented in a card such as the Pico E-12 or on a prototyping board such as the Xilinx ML403, promises to revolutionize the way that FPGA devices are applied for high-performance embedded computing.

Software-to-hardware tools such as Impulse C, when combined with the platform building capabilities of Platform Studio, make programming for such devices practical and efficient. 🌈

Application	PowerPC Only (300 MHz)	PowerPC/APU (300/50 MHz)	Acceleration
Image Filter (512 x 512 Image)	0.1414 sec	0.0124 sec	11.4 X
Encryption (8M Characters)	2.257 sec	0.0667 sec	33.84 X
Fractal Image (10K Max Iterations)	660 sec	31 sec	21.29 X

Table 1 – Virtex-4 APU acceleration results

A Wide Range of Development Platforms

Providing embedded application developers – software programmers – with an easy-to-use hardware platform is a critical first step in making FPGAs viable as embedded development platforms. A growing number of vendors are offering FPGA-based prototyping and high-performance computing platforms ranging from low-cost, single-FPGA systems to larger FPGA grids intended for hardware-accelerated computing.



Figure 5 – The Pico Computing EP-12 card packages the FX12 or LX-25 device with a CompactFlash interface in an extremely compact form-factor.

There are two versions of the Pico E-12. The Logic Optimized (LO) version is based on the Virtex-4 LX-25 device, while the Embedded Processor (EP) version is based on the Virtex-4 FX12 device, with its integrated PowerPC processor.

In either case, the FPGA on the E-12 card is configured from the 64 MB of on-board flash memory using an on-board loader. The unique design of this loader allows new FPGA images to be swapped into the FPGA on demand. A large number of FPGA images can be stored in flash memory, and any image can be loaded at any time through on-board software or external software communicating with the E-12 card through its external interfaces. The contents of the 128 MB of external RAM remain intact through the swapping sequence, allowing subsequent FPGA images to operate on existing RAM data.

The Xilinx ML403 (shown in Figure 6), the first of several Virtex-4 FX embedded processing development boards, combines the Virtex-4 FX12 with a wide variety of software-configurable interfaces, including network interfaces; serial, parallel, and USB ports; LVDS and D/A and A/D interfaces; and a VGA driver. As such, the ML403 is ideal for embedded systems designers requiring direct FPGA access to external hardware devices.

Figure 7 shows a comparison between the APU with Impulse-generated hardware accelerators and a processor/software-only implementation. Both systems are utilizing the ML403 in this example. You can see that the APU-accelerated version is significantly faster than the processor-only version.

The Pico E-12 card mentioned in the main article (which is available from Pico Computing, www.picocomputing.com) is a CompactFlash form-factor package that draws well under 2W. It features 10/100/1000 Ethernet, 64 MB of Flash, 128 MB of RAM, and a wide host of peripheral adapters such as A/D, D/A, asynchronous serial, synchronous serial, CAN bus, relay control, and JTAG. The card is supported by platform development and programming tools appropriate for software developers. The Pico E-12 platform advances desktop and portable computing by providing massively parallel hardware computing resources in a low-power, self-contained package (Figure 5).



Figure 6 – Xilinx ML403 development system

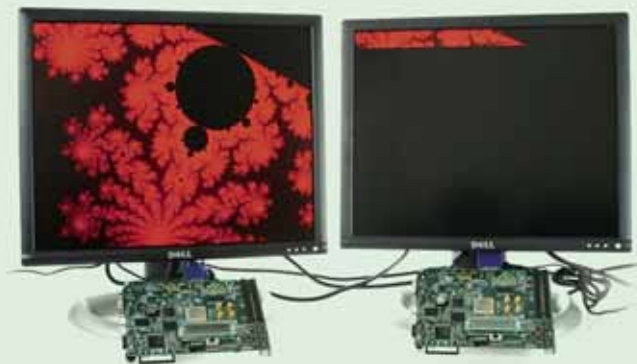
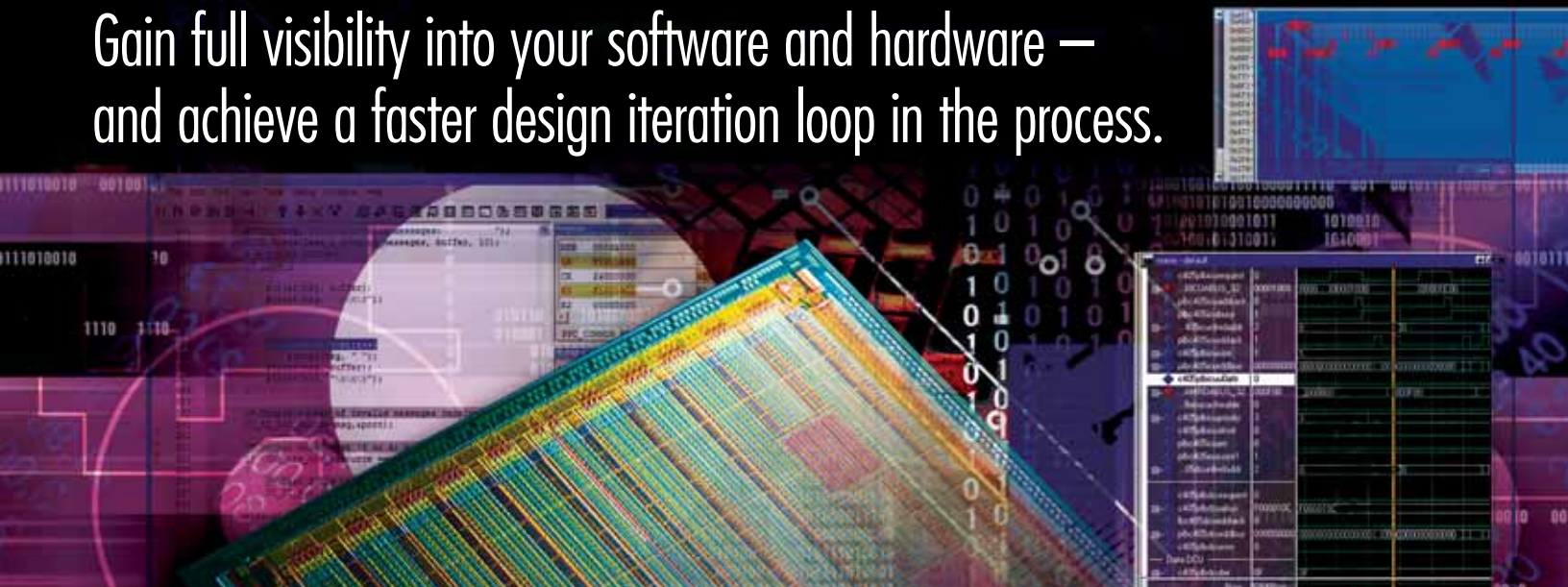


Figure 7 – Xilinx ML403 development system showing APU hardware accelerated implementation versus software only executing on the PowerPC

Hardware/Software Co-Verification

Gain full visibility into your software and hardware — and achieve a faster design iteration loop in the process.



by Ross Nelson
Seamless FPGA Product Manager
Mentor Graphics Corporation
ross_nelson@mentor.com

Copyright © 2005 Xilinx, Inc. All rights reserved.

You've probably been there: clever detective work leads you to a small change in the HDL for your embedded processor-based design. Now you just have to run synthesis, place and route, and darn ... you suddenly realize it will be another day before you can see the result.

Large devices allow you to stuff a whole system into the FPGA, but debugging these complex systems with limited visibility — and a one-day turnaround — can consume weeks of your precious time.

Hardware/software co-verification has been successfully applied to complex ASIC designs for years. Now available to FPGA designers, Seamless FPGA from Mentor Graphics brings together the debug productivity of both a logic simulator and a software debugger. Seamless FPGA co-verification enables you to remove synthesis and place and route from the design iteration loop, while yielding performance gains 1,000 times faster than logic simulation.

Shortening the Design Iteration Loop

Because development boards are readily available, many FPGA designers incorporate them into the highly iterative design loop. Unfortunately, the development board brings major overhead to every design iteration. This overhead comes in the form of logic synthesis, followed by place and route. Although necessary to produce a final design, you can remove these time-consuming steps from the highly iterative design debug loop by targeting simulation as the verification platform.

With simulation as the verification engine, the only overhead between editing the HDL and verification becomes a relatively quick compile of your HDL. The time you can save on your next embedded FPGA is easy to calculate: How many times did you run place and route on your last FPGA design? And how long did place and route consume your PC for each run?

It's true that simulation runs slower than the real-time speed of a development board. Seamless FPGA provides some innovative ways to dramatically increase the rate at which your embedded software simulates. The increase in a typical system is several orders of magnitude.

Improving Hardware and Software Visibility

To debug your FPGA design, you need full and clear visibility. You need to know what is happening in the hardware and what the software is doing. You need to be able to change a register, or force a signal to a different state. Sometimes you need to be able to stop time and take a closer look. The more visibility you have, the more quickly you can see the problem or prove you have resolved the bug.

Hardware Visibility

Probing inside or even on the pins of your FPGA is a challenge. The ChipScope™ Pro analyzer from Xilinx® helps with this, but in a logic simulator (in addition to viewing every signal) you can also change their values. Working from your source HDL, you can step through the code, view variables, or stop time. For detailed, immediate, and hassle-free visibility, it is hard to beat logic simulation.

Software Visibility

Software visibility in logic simulation is another item with which to contend. Running the fully functional processor model allows you to execute software, but

knowing what is in R3 of the processor is almost impossible if you are given only waveforms.

Co-verification provides an enhanced processor model connected to a software debugger. In the Mentor Graphics XRAY debugger, you can view and change everything from registers to memory, stack, and variables. XRAY also provides a source code view with symbolic debug. You can step through code at the source or assembly level and use breakpoints to halt execution or run powerful macros.

If you are using the Accelerated Technology Nucleus real-time operating system (RTOS), you can view the status of tasks, mailboxes, queues, pipes, signals, events, semaphores, and the memory pool.

Much Faster Than Logic Simulation Alone

Running substantial amounts of software on a standard processor model in logic simulation is not practical; the run times are just too long. However, running this software actually turns out to be one of the most effective verification strategies available. The payoff for running diagnostics, device drivers, board support package (BSP) code, booting the RTOS, and running low-level application code is huge. It is not surprising that verifying hardware – by putting it through its paces the way the software will actually use it – is effective. Similarly, the software is tested against the actual design (including any external board-level components that are included in the simulation) before the board is actually built.

The challenge has always been to run enough software to really boot the system and do something interesting. Co-verification is able to speed up the run time by taking advantage of one simple observation: most of the simulation time is spent re-validating the same processor-to-memory path. Although you need to test your memory subsystem and try several dozen corner cases, you don't need to repeat those same tests over again every time you fetch an instruction from memory. Similarly, you need to verify that the processor can push a value on the stack and pop it off again with the correct result, but repeating this test every time a software function is called would be overkill.

Accesses to hardware peripherals always generate bus cycles in the logic simulation, but instruction fetches and stack operations can typically be offloaded for faster execution. By allowing you to specify which bus cycles are run in the logic simulator and which are not, Seamless FPGA allows you to make the performance tradeoff. And you can change this specification at any time during your simulation session. You can run through reset with full cycle-accurate behavior, and then switch off instruction fetches and stack accesses to boot the RTOS.

Accessing memory through the logic simulator requires several hardware clock cycles. Each clock cycle requires significant work in the logic simulator as it drags along the heavy weight of all the other logic in your FPGA. Using a “back door” to directly access the memory contents, instead of running the bus cycle in the logic simulator, allows accesses to occur many orders of magnitude faster.

The speedup is very significant. For example, the following data is from a typical design configuration with a PowerPC™ running Nucleus on the Xilinx Virtex™-II Pro FPGA. Booting the Nucleus RTOS in

logic simulation alone requires 12 hours and 13 minutes. The same task with these techniques employed accomplishes the task in only six seconds – 7,330 times faster.

Using this technique, Seamless FPGA maintains one coherent view of memory contents through a back door into Xilinx block RAM memory models or any other memory device. So if your DMA controller drops something into memory that the processor later executes, it will still all work together correctly. And if the processor generates a large data packet and instructs hardware to transmit it using DMA, there are no data inconsistencies.

Identifying Processor Bus Bottlenecks

The performance of your FPGA platform can be seriously impacted by the memory structure of the design. What should be located in cache versus block RAM or external memory? Where are the bottlenecks? Do other bus masters starve the processor? Questions like these are important, but getting the answers can be difficult without real data from your hardware/software application.



Figure 1 – Seamless FPGA's system profiler helps you tune performance by providing detailed data on bus transactions and utilization, software function execution time, bus arbitration delay, memory hot spots, and software code profiling.

... software doesn't execute alone – it interfaces with hardware, and the hardware/software interface often stretches across disciplines and design teams.

Seamless FPGA gathers performance data from the simulation and displays it graphically in the system profiler (Figure 1), enabling you to identify:

- Which functions are consuming most of the CPU time
- Unexpected lulls or bursts of activity
- Cache efficiency and memory hot spots
- Code execution and duration at the function level
- Bus utilization and bus master contention

Ease of Use and Integration

Seamless FPGA is easy to use and set up. Using the knowledge you have already entered in Xilinx Platform Studio (XPS), Seamless FPGA automatically configures itself to co-verify your design. You may already know how to use ModelSim, and Seamless FPGA leaves the full functionality and user interface unchanged. The XRAY software debugger uses many of the

same menu icons for operations like step, step over, and run.

To set up Seamless FPGA, simply choose File > Import from Xilinx Platform Studio and specify your XPS project file. The import process does all of the setup steps and in about one minute proceeds to invoke ModelSim and the XRAY debugger.

If you have two or more Xilinx processors in your design, you will have additional software debugger windows, one for each processor.

Once ModelSim and XRAY have been invoked (Figure 2), you are ready to verify your design. In ModelSim, enter any stimulus commands needed – typically this is reset and clock, plus any design-specific stimulus – and then click “run.” In XRAY, click “go” or “step” to start stepping through your embedded code. By default, all bus cycles are routed to the hardware simulation.

To increase software execution speed, three icon selections are provided. These icons are labeled “optimizations” because they increase the rate of software execution by directing Seamless FPGA to access

memory contents through a back door without requiring the logic simulator to run every bus cycle. The first button directs all instruction fetch cycles to use the back door. A second button allows you to specify any number of address ranges, which use the back door. When accesses use the back door, you can either choose to keep advancing the logic simulation in lock step with the software or remove that requirement.

The optimization settings can be changed at any time on the fly during a simulation session. This allows you to quickly run to a certain point in your software, and then enable all bus cycles for detailed cycle-accurate verification.

Conclusion

With large FPGA designs employing embedded processors, it's not possible to complete a design in a few weeks. These designs are very sophisticated; unfortunately, so are the bugs that you must track down and resolve to produce an effective system on schedule.

Software content in your FPGA can bring lower system costs, higher configurability, and increased functionality. But software doesn't execute alone – it interfaces with hardware, and the hardware/software interface often stretches across disciplines and design teams.

Seamless FPGA bridges the hardware/software gap with a productive software and hardware debug environment that provides the visibility to find bugs and performance bottlenecks efficiently. And once you have fixed them, you can quickly turn the fix and verify it, without having to wait for your PC to rumble through place and route for hours on end.

Try Seamless FPGA on your design today. For your free 30-day evaluation copy, visit www.seamlessfpga.com. The included example design and Quick Start Guide will get you up and running in no time. For more information, e-mail seamless_fpga@mentor.com. 

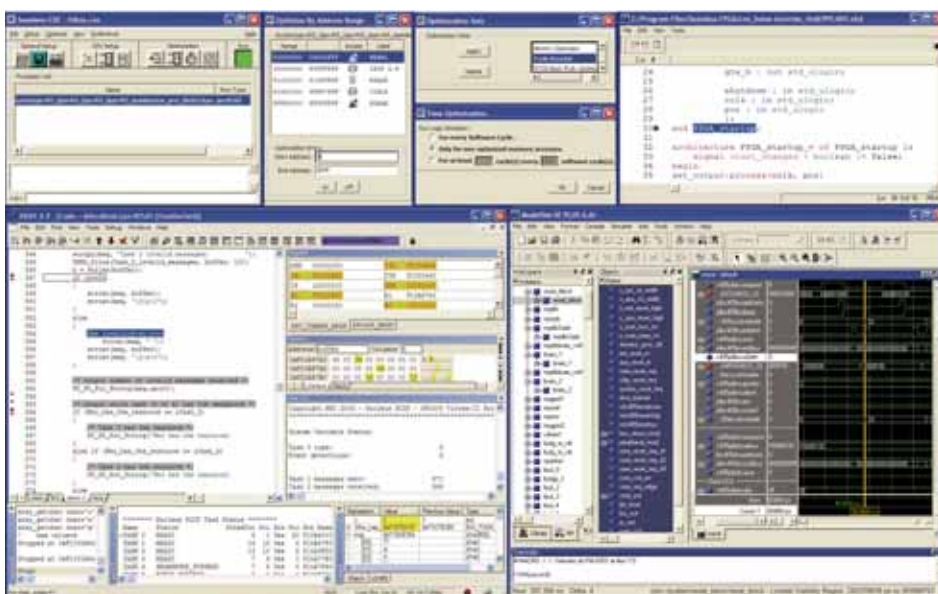


Figure 2 – Seamless FPGA running the Nucleus RTOS on a Xilinx Virtex-II Pro PowerPC processor. XRAY (lower left) provides symbolic software debug and RTOS task status. ModelSim (lower right) enables full hardware debug and control. Seamless FPGA control panels (upper left) allow dynamic selection of bus cycles routed to ModelSim. The ModelSim HDL source window is also displayed (upper right).

Moving Embedded Systems onto FPGAs

Altiium Designer allows hardware developers to move system complexity from the board level into the “soft” programmable logic realm.

by Nancy Eastman
Regional Director, USA, Altiium Limited
Altiium Limited
nancy.eastman@altium.com

Copyright © 2005 Xilinx, Inc. All rights reserved.

The development of electronic products is a juggling act that balances the drive to embed more and more intelligence into a design with the time needed to create, implement, and test the application. The history of electronics charts a continuous movement toward designing at higher levels of abstraction to efficiently contend with increasing levels of complexity.

Microprocessors and digital design paradigms allowed portions of design problems – such as adding more intelligent features to a design or executing complex signal processing functions – to be moved from hard-wired components into a highly fluid and easily updateable realm: software. This enabled some complexity to be dealt with in a “soft” environment that was flexible throughout the design process.

Today, the availability of high-capacity, high-performance programmable devices (such as FPGAs) at relatively low costs is shifting the balance again, allowing previously fixed design elements such as the processor and its peripheral components and logic blocks to move into a soft domain (Figure 1). This holds the promise of greater design flexibility and the freedom to change crucial design decisions – the partitioning of functions between software and hardware implementation or even the choice of processor – throughout the development cycle.

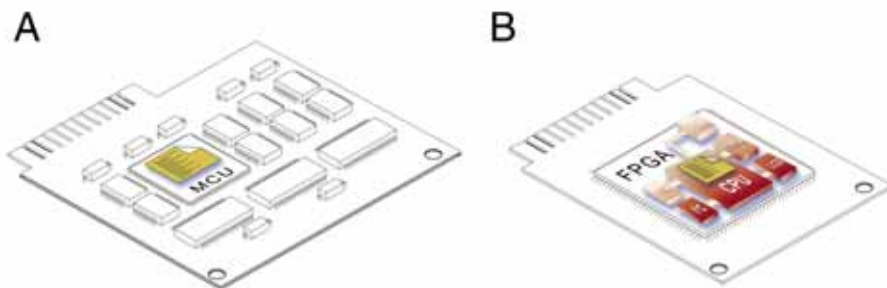


Figure 1 – As shown in illustration “A,” microprocessors allowed some of the design problem to be moved into easily changeable software, but much of the circuitry remains hard-wired. In illustration “B,” moving the bulk of the system onto an FPGA platform means that both the software and execution platform are easily changeable during development.

Barriers to Mainstream Adoption

To date, the development of FPGA-based processor applications has been a niche exercise, at least compared to the number of embedded systems developed using discrete off-the-shelf processors. Certainly FPGAs have been widely used to contain much of the glue logic that surrounds the processor in an embedded system. But the processor and its major peripheral components have remained mainly hard-wired and outside the programmable space.

This is partly a pricing issue. Historically, FPGAs that were large and capable enough to provide a platform for processor applications have been far more expensive than comparable off-the-shelf microcontroller units (MCUs). Therefore, designers needed a very good reason to justify the extra expense of taking an FPGA approach, limiting the range of applications targeted. More recently, however, devices like the Xilinx® Spartan™-3 family have pushed the pricing envelope and, when combined with a suitable FPGA-based processor core, are changing the cost/benefit equation.

Price is not the only barrier to mainstream penetration of FPGAs as an embedded systems platform. Another and perhaps far more intractable problem is changing the way we think about programmable logic devices in general. Rather than seeing them simply as an efficient way to integrate logic blocks, we need to look at the big picture, reassessing the whole design process in the context of the reconfigurability that FPGAs offer.

The Big Picture

A clue to this big-picture view of the FPGA phenomenon as it relates to embedded design lies in the history of the microprocessor itself. Originally developed for use in calculators and then personal computers, the microprocessor revolutionized mainstream electronics design when the devices could be bought at a fraction of the cost of the products in which they were used. The technology progressed to the point where a relatively user-friendly development paradigm could be widely adopted – in this case high-level programming languages such as C.

The flexibility and power of software allowed designs to be created in a new way, where large parts of a system’s functionality could be created and modified on the fly without redesigning hardware. The ability to use C to program embedded applications meant that this power and flexibility was available to a wide engineering audience, effectively making embedded processor-based design the mainstay of the electronics industry.

FPGAs have the potential to create a similar revolution in design by dramatically increasing the amount of system that can be “soft.” As previously mentioned, large-scale programmable devices are now available at prices that allow them to compete with discrete processor systems. What is needed now to drive the adoption of FPGAs for embedded applications is a user-friendly, accessible development method that facilitates the easy integration of processors, peripheral hardware, and

software within a programmable platform. This method should allow the integration of FPGA design with the board design process and facilitate the rapid design changes possible within this new “soft” design paradigm.

The Need for a User-Friendly Development Paradigm

FPGA design techniques are traditionally based around the FPGA as a component within a larger system. But when the FPGA is the system platform, sourcing the necessary system components in the HDL realm and instantiating them at the register transfer level is a complex process – a process that is daunting for the majority of engineers, who are not FPGA specialists.

These same engineers, however, will have no trouble developing a very complex system at the board level. At the board level, the complexity of the system is embodied in the off-the-shelf components used to create the design. Engineers can simply use these components as is without needing to understand the underlying complexity.

The key to unlocking the potential of FPGAs as a mainstream embedded systems platform is providing a seamless transition between current board-level design practices and FPGA-based system design.

The Future of the Design Desktop

One recent development in this direction is Altium Designer, an electronic product development system from design solutions provider Altium Limited. Altium Designer provides a graphical capture environment for FPGAs that includes libraries of high-level FPGA components. These components include a range of processor cores and peripherals, which are provided pre-synthesized for a wide range of target FPGA devices. The components are ready to use, making system hardware creation a drag-and-drop exercise (Figure 2).

The system includes its own royalty-free 32-bit processor – the TSK3000 – that can be used across a wide range of FPGA devices and families. Other supported execution platforms include the Xilinx MicroBlaze™ core and the hard PowerPC™ processor embedded in Virtex™-II Pro devices.

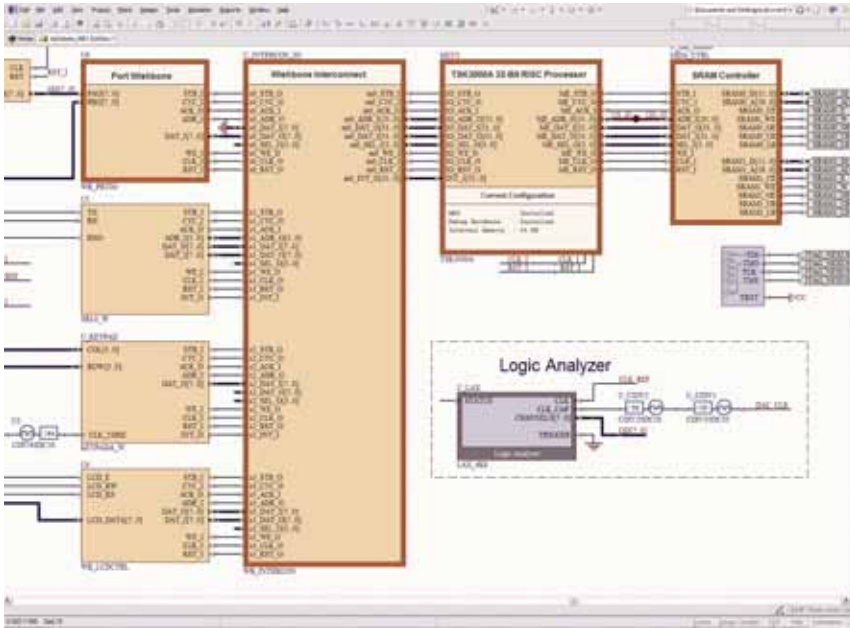


Figure 2 – Typical block-level definition of system hardware created in Altium Designer from ready-to-use, pre-synthesized components, including the processor and configurable Wishbone bus interconnects.

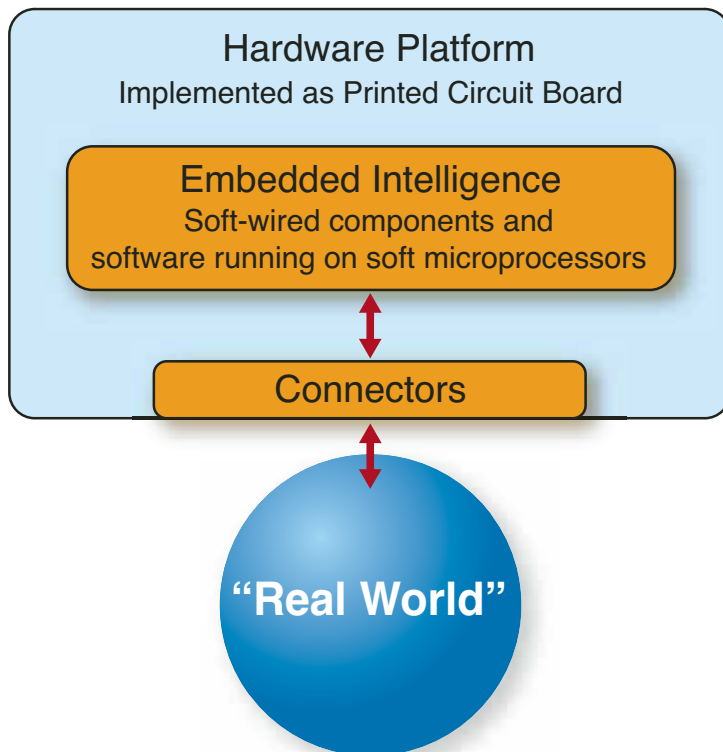


Figure 3 – With FPGAs, the embedded intelligence of a product can encompass both software and soft-wired system components contained within the FPGA. The PCB then becomes simply a platform for connecting the device intelligence to the outside world.

Altium Designer makes extensive use of the royalty-free Wishbone processor interconnect bus, and supplies several configurable bus connection components for easy interconnection with processor peripherals. Wishbone-based wrapper cores for MicroBlaze and PowerPC processors make it possible to retarget designs between processors without having to re-engineer the system. A common compiler engine and integrated tool chains support this process at the software level.

This approach allows embedded developers to choose the most appropriate execution platform for their applications. You can commence a design using the vendor-neutral TSK3000 and move it to a PowerPC if you need a higher degree of performance, or migrate to a MicroBlaze solution optimized for the particular Xilinx device you are targeting.

Conclusion

Altium Designer allows hardware developers to move system complexity from the board level into the “soft” programmable logic realm using their existing skill set. This drastically increases the number of engineers who can free themselves from hard-wiring system components and design in an environment where both the software and hardware that make up a product’s intelligence are easily changeable on the fly.

The move toward “softening” the design process that began with the availability of cheap microprocessors is being taken to a new level by current advances in FPGA technology. Today the bulk of the intelligence in an electronic device resides primarily in the embedded software. With FPGAs, the embedded intelligence can span both software and soft-wired components contained within a programmable platform (Figure 3). Opening up of this potential to mainstream embedded developers will fuel an explosion in FPGA use and set the foundation for tomorrow’s electronic products.

For more information about Altium Designer and its capabilities, or to learn more about the emerging soft design paradigm, please visit www.altium.com.

Tracing MicroBlaze Processors with a Logic Analyzer

Agilent's MicroBlaze trace core and inverse assembler simplify the task.

by Joel Woodward
Product Manager
Agilent Technologies
joel_woodward@agilent.com

Copyright © 2005 Xilinx, Inc. All rights reserved.

Logic analyzers provide deep trace and sophisticated triggering for capturing software trace. These measurements can be critical for tracking events leading up to a crash, evaluating the interaction of software with hardware, or analyzing software performance.

However, microprocessor technology shifts (such as increased use of cache memory) have made the information that can be captured on the pins of stand-alone microprocessors less relevant. These shifts make it increasingly difficult to debug embedded systems. Fortunately, a new logic analysis innovation for Xilinx® FPGAs allows you to quickly take MicroBlaze™ soft-core processor measurements.

Software Execution with Logic Analyzers

Beginning in the 1980s, design teams used logic analyzers extensively to debug embedded systems, designing in PC board connectors with a specified layout for tracing the processor. For 32-bit processors, this requirement typically included 102 to 136 signals comprising data, address, and status signals.

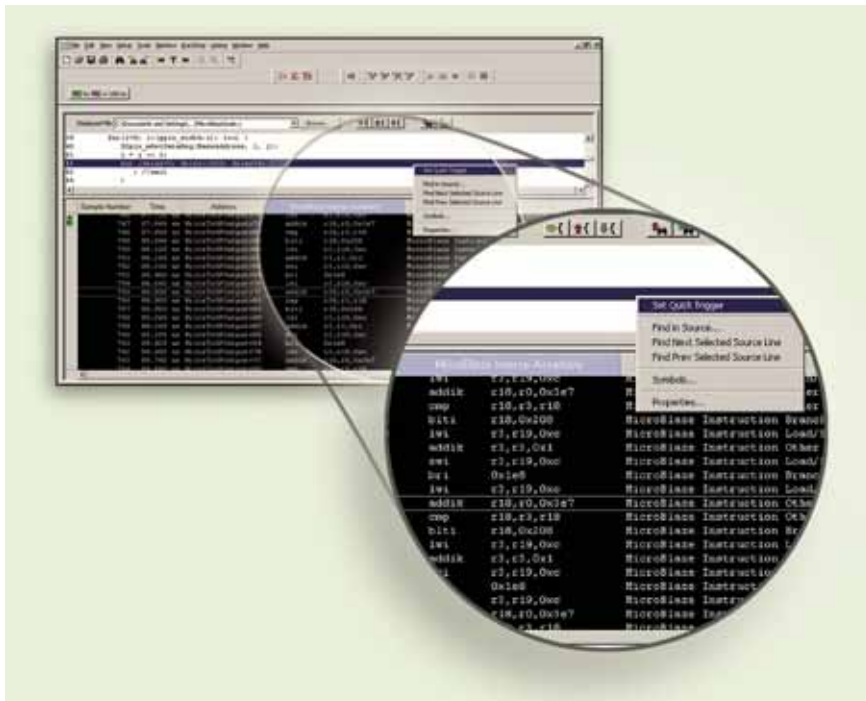


Figure 1 – Agilent's MicroBlaze inverse assembler works with all of Agilent's 1680, 1690, and 16900 series logic. Agilent logic analyzers come standard with a source correlation window so that team members can set up a measurement at the assembly or source level.

Logic analyzers still require a certain arrangement of processor address, status, and data signals to be laid out on your target system. Each processor has an associated logic analysis configuration file and inverse assembler that works with a specified PC board layout. Inverse assembly software converts the acquired ones and zeros to instruction mnemonics.

In the late 1990s, 32-bit embedded processors began incorporating technologies that made it more difficult to make logic analysis trace measurements. Shrinking real estate available for debug made it impractical to include connectors for tracing. Even if the design team had designed in connectors for trace measurement, no signals of relevance would be transmitted to the pins monitored by the logic analyzer if the processor was executing out of cache memory. Turning off cache caused the system to run at a slower rate and could mask problems that would only occur at real system speeds. In addition, pipelining and out-of-order execution made it more difficult for logic analysis vendors to unravel information on the bus.

Recently, Agilent and Xilinx have collaborated to develop a logic analysis trace solution for MicroBlaze processors that overcomes the traditional difficulties of tracing software execution using a logic analyzer.

MicroBlaze Inverse Assembly

For PC board layout, design teams enable the inverse assembler by routing at least the MicroBlaze program counter signals (PC_Ex) and the valid cycle signal (Valid_Instr) to pins. Routing these signals to a specified layout allows for fast connection to a logic analyzer through Mictor, Samtec, or soft-touch probing. You can also connect the logic analyzer to these signals with a berg strip or header, using individual flying leads.

Because of their reprogrammable nature, FPGAs with MicroBlaze processors can be traced late in the development cycle. As long as a sufficient number of pins have been reserved for debug, you can route the required MicroBlaze signals to a specified pinout without PC board changes.

The Agilent inverse assembler for MicroBlaze processors reconstructs program flow by capturing the address of each

executed instruction and looking up the associated opcode in the OMF (object module format) file. It then decodes the opcode into a MicroBlaze mnemonic, as shown in Figure 1.

As pins available for debug are often scarce, the inverse assembler includes a capability that reduces the number of required pins. Although 32 PC_Ex signals exist, the number of external signals needed to capture a logic analysis trace is typically much less. This reduction is accomplished using two different techniques.

First, you do not need to trace the upper address bits. For any given design, a certain number of upper address bits are static. You can achieve more pin reduction – with one additional pin decreased for each static upper address bit in the program counter – by specifying this information in the logic analysis user interface.

Second, you also do not need to trace the lower two address bits. All instructions start on 4-byte boundaries. Using these techniques, tracing software execution of a 1 MB program requires only about 18 pins. Simultaneous capture of data requires additional pins.

Tracing with Cache Enabled and Using Source Correlation

For stand-alone processors with cache enabled, logic analysis tracing becomes impossible. Fortunately, the situation changes for processors embedded in FPGAs. A logic analyzer can trace MicroBlaze processors even when cache is enabled. Captured signals are routed from the execution stage of the MicroBlaze pipeline.

Agilent logic analyzers come standard with a source correlation window. By reading a symbol file (in .elf format), the logic analyzer can associate captured addresses with the high-level software associated with that address. Opcode images are found in the text sections of the .elf object files. As you step through assembly instructions, the equivalent line in the source code for this instruction is also highlighted. You can alternatively step through high-level source code while the logic analyzer simultaneously displays the associated instruction mnemonics in the lower window. Simply

right-click in the source code to set up the logic analysis trigger (trace specification) for the next acquisition, as shown in Figure 1.

MicroBlaze Trace Core (MTC)

An optional MicroBlaze trace core, or MTC, reduces the amount of time and number of pins required to trace MicroBlaze processors with a logic analyzer (see Figure 2). The MTC core, co-developed by Agilent and Xilinx, works exclusively with the Xilinx Platform Studio included with the Embedded Development Kit. Design teams can graphically add an MTC core to their design. Core parameters include pin compression using time-division multiplexing (TDM), pin location, and I/O standards.

The MTC core provides four key values:

1. The MTC core connects required MicroBlaze signals to pins (pre-synthesis).
2. The core incorporates TDM to reduce the number of pins required by a factor of two. Two MicroBlaze signals are TDM'd onto a single pin, with data valid on the rising edge of the clock for signal one and data valid on the falling edge of the clock for signal two. A demux clocking mode in the logic analyzer decompresses the information and splits it into two separate logic analysis channels.
3. The MTC core includes technology that reduces initial setup time from hours to seconds and eliminates manual errors that can happen during PC board layout. Tracing MicroBlaze processors can be accomplished late in product development, as the MTC core eliminates the need to layout a PC

board with a specific MicroBlaze signal pattern for Mictor, Samtec, or soft-touch probes. Through JTAG, the logic analyzer sends an auto setup message to the MTC core. The core outputs a training pattern on a specific MTC pin. The logic analyzer looks for this training pattern across its channels and discovers which channel is connected to the MTC pin. The logic analyzer knows how

each MTC core input is routed through the core to pins from its communication with the MTC core. Using this correlation, the instrument now has sufficient knowledge to determine how to set up the physical connection between specific microprocessor signals and the input channel on the logic analyzer. This process is sequentially repeated for each MTC output pin.

4. Lastly, the MTC core – constructed entirely of flops and LUTs – uses a multi-stage pipeline (typically four) to minimize impact on device timing when the core is inserted (Figure 3). MTC cores are very small. An MTC core in a Xilinx XC2V3000 device consumes roughly 1% of the LUTs and flops.

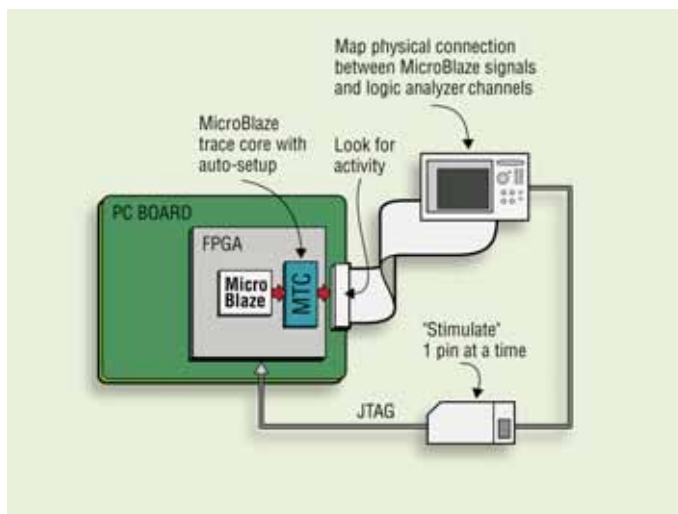


Figure 2 – Agilent's MicroBlaze trace core (MTC) reduces setup time for an initial trace measurement. You can literally connect a logic analyzer to a connector with MTC core outputs routed to it. Within seconds, the logic analyzer is ready to take a measurement. The MTC core, with its pin compression technology, reduces the number of required pins for tracing MicroBlaze processors by 50%.

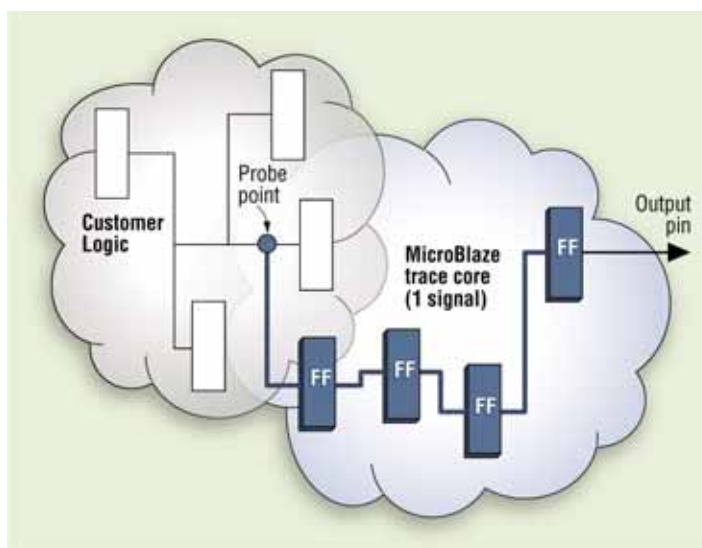


Figure 3 – The thick blue lines show the flops and routes added by the MTC core. Because there is a flop in the fabric – in addition to one at the I/O buffer – the router can use timing solely within the MTC core to move across the chip, thereby minimizing the impact of timing changes with the addition of the MTC core.

Conclusion

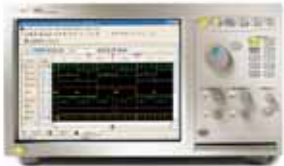
FPGAs enable fast, accurate processor execution tracing not available for stand-alone processors. Agilent's inverse assembler for the MicroBlaze soft-processor core provides you with an effective tool for tracing software execution. Agilent logic analyzers, equipped with precise time resolution, correlate MicroBlaze execution history with other software or hardware events acquired simultaneously. This allows you to quickly isolate problems associated with hardware and software interaction.

Agilent's royalty-free MTC core, distributed as part of the Xilinx Embedded Development Kit (beginning with version 8.1i), minimizes the time to set up measurement and eliminates the need for a specified PC board layout. For more information, visit www.agilent.com/find/microblaze.



X-ray vision for your designs

Agilent 16900 Series logic analysis system with FPGA dynamic probe



- Increased visibility with FPGA dynamic probe
- Intuitive Windows® XP Pro user interface
- Accurate and reliable probing with soft touch connectorless probes
- 16900 Series logic analysis system prices starting at \$21,000



Agilent Direct

Get a quick quote and/or FREE CD-ROM with video demos showing how you can reduce your development time.

U.S. 1-800-829-4444, Ad# 7909

Canada 1-877-894-4414, Ad# 7910

www.agilent.com/find/new16900

www.agilent.com/find/new16903quickquote

Now you can see inside your FPGA designs in a way that will save days of development time.

The FPGA dynamic probe, when combined with an Agilent 16900 Series logic analysis system, allows you to access different groups of signals to debug inside your FPGA—without requiring design changes. You'll increase visibility into internal FPGA activity by gaining access up to 64 internal signals with each debug pin.

You'll also be able to speed up system analysis with the 16900's hosted power mode—which enables you and your team to remotely access and operate the 16900 over the network from your fastest PCs.

The intuitive user interface makes the 16900 easy to get up and running. The touch-screen or mouse makes it simple to use, with prices to fit your budget. Optional soft touch connectorless probing solutions provide unprecedented reliability, convenience and the smallest probing footprint available. Contact Agilent Direct today to learn more.



Agilent Technologies

dreams made real

Nucleus Integration with Xilinx FPGA System Design

Accelerated Technology's Nucleus, integrated with EDK, provides the most extensive solution for embedded system architects.

by Aaron Spear

Lead Architect, Tools Solution
Accelerated Technology, A Mentor Graphics Division
aaron_spear@mentor.com

Phillip Walker

Technical Marketing Engineer
Accelerated Technology, A Mentor Graphics Division
phillip_walker@mentor.com

Copyright © 2005 Xilinx, Inc. All rights reserved.

The FPGA-based embedded system design flow provides many benefits to system developers, as well as new challenges. Chief among the benefits are accelerated design flows that allow you to move quickly from the design and testing cycles to marketing and selling. With this accelerated flow, it is more important than ever that the hardware and software designs are in sync throughout the entire engineering design cycle.

Accelerated Technology, A Mentor Graphics Division, has developed a version of its Nucleus embedded software suite that integrates with the Xilinx® Embedded Development Kit (EDK). This provides a tight integration of software systems in the FPGA embedded systems design flow. EDK is based on a data-driven code base that makes it extensible and open. By leveraging this functionality, Nucleus software is able to achieve a level of integration into the FPGA-based embedded system design flow that was previously not possible.

The Nucleus embedded software suite for Xilinx FPGA system design includes a complete tool offering and target software platform, including high-level modeling with xtUML and advanced target software debugging with the Eclipse-based Nucleus EDGE environment.

Auto Configuration with MLD

The underlying technology of the Xilinx approach is microprocessor library definition (MLD). This technology allows for automatic kernel configurations and the generation of board support packages (BSPs). This unique and straightforward approach allows the Nucleus embedded software suite to configure to FPGA system designs created in EDK. This eliminates the need to re-report the software system to the new memory map and peripherals for every hardware design cycle.

The Data-Driven Approach

EDK uses two main data repositories to store information on hardware- and software-related settings. All hardware-related settings are stored in the MHS (microprocessor hardware specification) file, while software-related settings are stored in the MSS (microprocessor software specification) file. These files provide a database that other tools in the Xilinx system can access for any given project.

The data-generation component, as defined in a TCL file, takes hardware details from the MHS data file and custom information from the MLD file and decides which files to generate and what parameters to customize. The Nucleus-

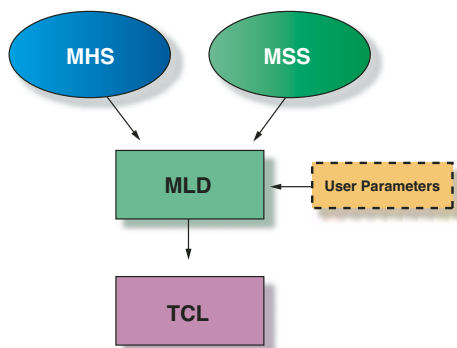


Figure 1 – Dataflow in Xilinx EDK

specific functions of the TCL file encapsulate the entire algorithm, generating consistent information used by the Nucleus kernel to support a wide range of hardware IP configurations (see Figure 1).

The elements of the Nucleus PLUS real-time kernel modified by the data generation file include:

- The number and type of peripherals used
- Memory map information
- Locations of memory-mapped device registers
- Timer configurations
- Interrupt controller configurations

Core Generation

Xilinx currently offers two processor choices for implementation in their FPGAs: the PowerPC™ 405 hard-core processor and the MicroBlaze™ soft-core processor. The Nucleus embedded software suite currently works with both options. Once you have selected your processor and configured your system inside EDK, enabling Nucleus is as simple as a drop-down menu selection.

Configuring Nucleus and BSP Generation

After you have generated your basic system design and core selection in EDK, you are ready to implement the Nucleus embedded software suite. As Figure 2 illustrates, we leveraged Xilinx MLD technology to add this functionality to EDK.

Once you have configured Nucleus to fit your system requirements, generating the corresponding BSP is straightforward. Simply choose the “Generate Libraries and BSPs” from the Tools menu option in EDK. This will build the correct libraries and any associated applications that the Nucleus embedded suite requires in order to run on your newly designed system.

System Debugging with Nucleus EDGE

You now have your hardware defined and the Nucleus PLUS kernel configured to run on your new platform. But what about software development? Where do we go from here?

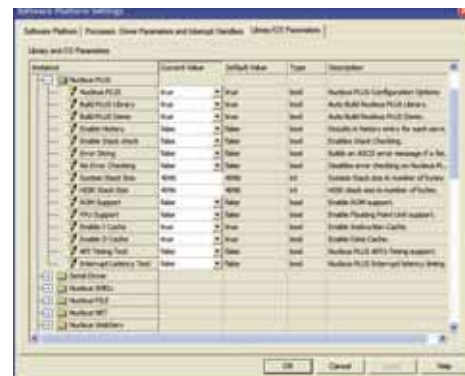


Figure 2 – Components of the Nucleus system are now configurable from within the EDK through the Library/OS Parameters dialog.

Introduction to Nucleus EDGE

Accelerated Technology offers the Eclipse-based Nucleus EDGE development environment, a complete development environment that supports JTAG debugging of both PowerPC and MicroBlaze processors. Nucleus EDGE extends the Eclipse platform for multi-core, multi-process, multi-thread debugging. It can be installed as a stand-alone or into Platform Studio SDK to provide advanced debugging and project management capabilities.

Nucleus EDGE is state-of-the-art debugging technology that includes features such as:

- Full multi-core/multi-process/multi-thread debugging, with support for synchronous operation on multiple cores simultaneously
- Advanced C-like scripting language (codelets)
- Data-driven target/core/peripheral descriptions (XML)
- Support for freeze-mode/run-mode debugging (OS- and hardware-dependent)
- Pluggable kernel awareness (data-driven)
- Pluggable connection devices
- Pluggable core support
- Pluggable real-time trace support
- Pluggable profiling engine

Creating and Building Applications

Nucleus EDGE provides a powerful build and project management environment. The Nucleus EDGE builder is a front end for any tool that transforms one or more files from one format into another format; examples include a compiler that transforms a C file into an object module or a linker that transforms N object modules into an executable. You can plug tools into the Nucleus EDGE builder by writing a simple XML description for that tool. We currently have built-in support for 32 different tool sets, including Xilinx GNU for both PowerPC and MicroBlaze processors.

BSPs

When targeting traditional processors with Nucleus EDGE, you are responsible for creating and maintaining BSPs that the debugger and project manager use. One advantage of Nucleus software integration with EDK is that BSPs are generated automatically, making maintenance painless. When you finish your hardware design and generate the BSP, the Nucleus EDGE BSP is also generated. This BSP is then used to determine appropriate tool defaults when creating applications, or knowing the layout of memory and peripherals when debugging.

Getting Started with Project Management

Nucleus EDGE provides a powerful user interface to change compiler settings for a given project, or optionally override them for a particular file. You are free to type in the command-line arguments if you know them, or you can peruse the options using a tree, which contains information about the command and allowable settings for it. It is nice not to have to wade through obscure compiler documentation to find the setting you need and its syntax (Figure 3).

Editing and Building

Nucleus EDGE provides a full-featured context-sensitive editor for C/C++ as well as assembly. The editor provides the following features:

- Configurable syntax highlighting (you can change the colors)

- Outliner that aids in navigation for your active source file
- Right-click navigation for declaration/definition of function calls
- During debugging, hovering over variables displays their current value; additionally, you can define your own script functions to render tool tips for your application data types
- Code completion for both functions and macros

Any errors in your source during building are displayed in the build console. You can click on errors; the editor synchronizes to the location for you. The editor decorates all warning or error source locations with special icons (Figure 4). Figure 4 also shows the outliner at the right side of the file.

Debugging PowerPC and MicroBlaze Processors

For Nucleus PLUS kernel applications, Nucleus EDGE can support run-mode debugging – the debug of individual tasks while the rest of the system continues to run. To accomplish this, it can use a serial port, Ethernet connection, or even the Xilinx JTAG UART.

For MicroBlaze processors, Nucleus EDGE currently supports debugging through XMD (Xilinx Microprocessor Debugger). For PowerPC, connection options include XMD, or, for PowerPC designs in which you have instantiated a dedicated JTAG scan chain, third-party JTAG devices such as Abatron's BDI2000 and MacCraigor Systems On Chip Demon family of connections.

Platform Debugging (Hardware/Software Co-Debugging)

One useful feature gained from connecting through XMD is that you can leverage ChipScope™ Pro hardware debugging features simultaneously while you debug your software using Nucleus EDGE. In the ChipScope Pro GUI, you get a logic analyzer view of signals inside your core. You can then configure the ChipScope Pro analyzer to halt the processor when the state of a certain peripheral changes, for example.

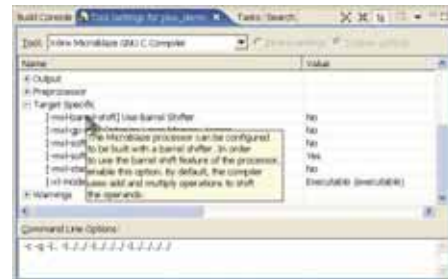


Figure 3 – Nucleus EDGE build settings for MicroBlaze GNU



Figure 4 - Building with errors



Figure 5 - MicroBlaze registers

When this occurs, Nucleus EDGE synchronizes, and you see the exact state of your software when the event occurred.

Debugging

Nucleus EDGE contains many special features for embedded debugging. The register view, for example, shows groups of native processor registers as well as memory-mapped peripherals. Those bits in the register that are set are highlighted in an optional graphical control. Bit-mapped registers show the bits set, and allow you to control them individually.

In Figure 5, you can see that “Exception Enable” is bit 6 in the MSR (the bold box around the bit), and that the bit is not currently set (the blue background). Gone are the days of getting out your calculator to do binary conversions and counting bits to

figure out if a bit is set. Figure 5 also shows how values that changed from the last step are color-coded (red).

Breakpoints

One other compelling feature that Nucleus EDGE offers above and beyond the capabilities of Platform Studio SDK is built-in integration with hardware breakpoints. As you may know, you can locate as many as eight program counter hardware breakpoints (used for stepping), as well as four read watch points and four write watch points in a given MicroBlaze design. Nucleus EDGE offers a completely integrated and graphical method to set both types of breakpoints. Also, the Nucleus EDGE debug engine is able to use the hardware breakpoints seamlessly to enable stepping in ROM.

Multi-Core Debugging

The number of MicroBlaze cores that can be placed in a design is only limited by the size of the FPGA. However, the MicroBlaze debug module can support debugging of as many as eight MicroBlaze cores simultaneously. The Nucleus EDGE user interface and debug engine have the ability to create “synchronization groups” of different cores. When one of these cores stops, all of the cores in the group are stopped.

Although Xilinx does not currently ship an IP block that supports configuration of synchronous control of multiple cores, it is a relatively trivial matter to implement it yourself – after all, you have an FPGA. Simply tie together a memory-mapped register with some MUX logic on the MB_HALT pin (that indicates that a core has gone into debugging state) as well as the DBG_STOP pin (that can force a core into debugging state). This way, when one core in a group either hits a breakpoint or has an exception, all of the cores stop. Then, in Nucleus EDGE, you can provide a codelet script that sets this register appropriately.

Codelets/Scripting

Nucleus EDGE contains support for a scripting language that we call “codelets.” The syntax is standard ISO/ANSI C, with a few extensions. Simply put, codelets are scripts that run in the debugger but have

full visibility and control over the target.

You can access target registers, memory, and variables, as well as call target functions from within a codelet. You can read and write host files as well as sockets. You can open “channel viewers” in the debugger GUI and execute them through any different expression evaluation. You can call them from the command line, or when hitting a breakpoint, or by typing an expression in the watch window. Codelets are meant to be an enabling technology. They allow you to get inside your hardware in a way that is not otherwise possible. Some things that customers have done with codelets include:

- Board initialization during debug
- Complex conditional breakpoints
- Custom hardware validation/regression testing
- Virtual console I/O
- “Poor man’s” kernel awareness
- SmartWatch – the ability to define a codelet that is used to “render” a given data type to a string, giving you nice tool tips for your data structures when debugging

Channels

Nucleus EDGE contains an abstraction for communications that we call channels. Any byte stream can be a channel. Files, sockets, and serial ports can all be channels. Codelets can also be used to create channels. On top of that, the GUI provides the ability to write “channel viewer plug-ins,” a way to render the data that comes from these channels. Using this infrastructure offers all kinds of interesting capabilities. Nucleus EDGE currently ships with the following built-in channel viewers:

- Generic text console I/O (standard I/O with the app)
- VT-100 compatible console I/O (supports escape sequences)
- Strip chart recorder that allows you to plot any value over time in real time
- Windows Media Player streaming plug-in (plays MP3s, MPEG video) (on Windows hosts only)

- Binary data viewer (like the memory view, in effect a “protocol analyzer”)

These viewers are just the beginning. Channels also allow us to abstract the mechanism used to connect to a profiling agent or run-mode debugging, for instance. When coupled with the Xilinx JTAG UART, this yields a powerful infrastructure for getting inside your application.

Kernel Awareness

Nucleus EDGE kernel awareness gives you the ability to see a snapshot of the state of your system, as well as providing the ability to set thread-dependent breakpoints. We currently provide out-of-the-box kernel awareness for the Nucleus PLUS kernel. However, Nucleus EDGE also gives you the ability to configure your own kernel awareness. This can be done for a third-party RTOS, an in-house kernel, or no RTOS at all.

Nucleus EDGE provides a data-driven mechanism to describe how it should iterate objects of a given type and display their attributes. They do not even have to be software objects – they could be anything that is memory-mapped (Figure 6).



Figure 6 - Kernel awareness

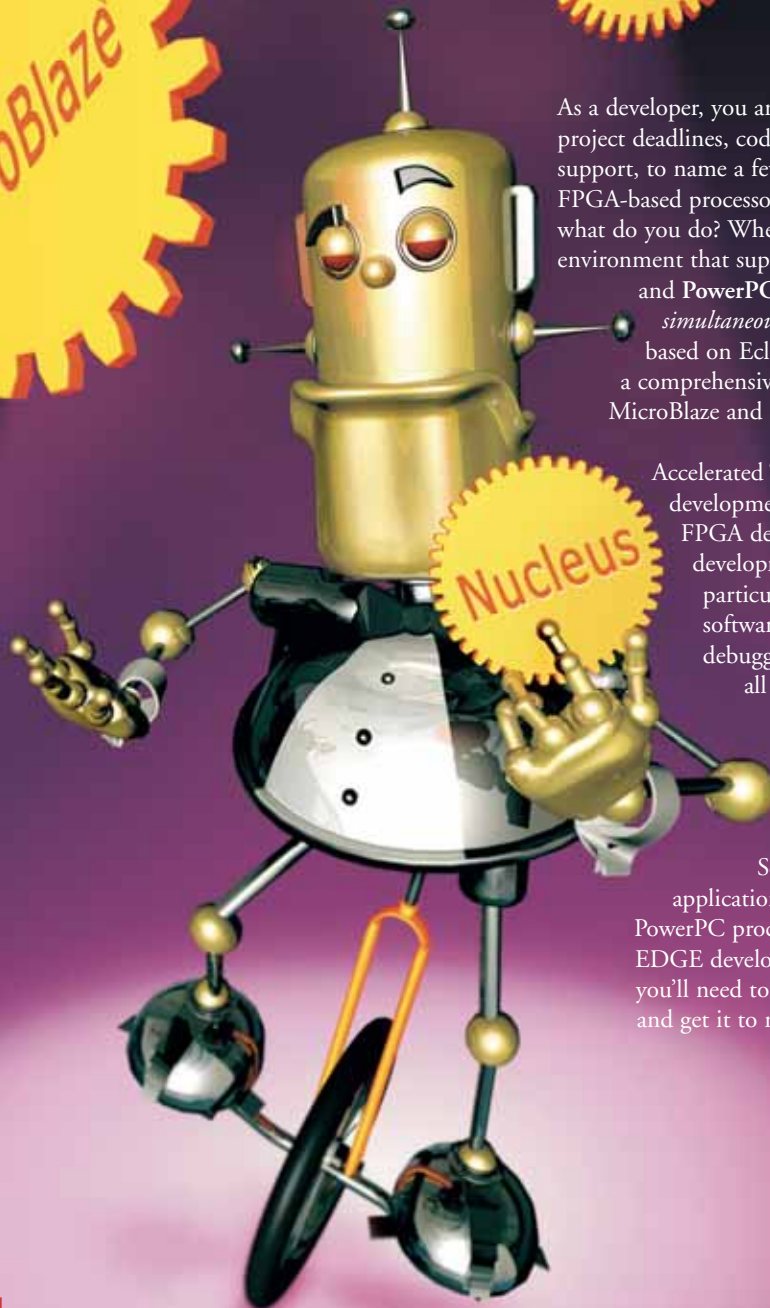
Conclusion

Configurable cores are the future of embedded development. With the combination of auto configuration of Nucleus target software and advanced debugging with Nucleus EDGE, Accelerated Technology has bridged long-standing gaps in integrated system design. By supporting both PowerPC- and MicroBlaze-based FPGA systems, Accelerated Technology distinguishes itself from the competition and provides unparalleled software tools and support for FPGA system designers.

For more information, evaluations, and updates to these exciting technologies, visit www.acceleratedtechnology.com/xilinx. 🌟

NucleusEDGE

Comprehensive Tool Suite for FPGA Developers



As a developer, you are concerned with many issues—project deadlines, code quality and integrated tool support, to name a few. And now that you've selected an FPGA-based processor to power your next application, what do you do? Where can you find a rich development environment that supports both the Xilinx MicroBlaze™ and PowerPC™ processors, *individually and simultaneously*? The Nucleus® EDGE software, based on Eclipse, answers this need by providing a comprehensive, fully developed tool suite for both MicroBlaze and immersed PowerPC developers alike.

Accelerated Technology recognized a lack of development tools designed specifically for FPGA designers and created a rich development environment for their particular needs. The Nucleus EDGE software consists of an IDE, compiler, debugger and system profiler — all seamlessly integrated so that FPGA developers can create their product from conception to deployment in one complete environment.

So whether you're developing your application with a MicroBlaze or immersed PowerPC processor-based FPGA, the Nucleus EDGE development environment is the only tool you'll need to completely develop your product and get it to market quickly.

**Accelerated
Technology®**
A Mentor Graphics Division

Accelerated Technology, A Mentor Graphics Division
info@AcceleratedTechnology.com • www.AcceleratedTechnology.com

For a FREE evaluation of Nucleus for Xilinx EDK visit:
AcceleratedTechnology.com/xilinx

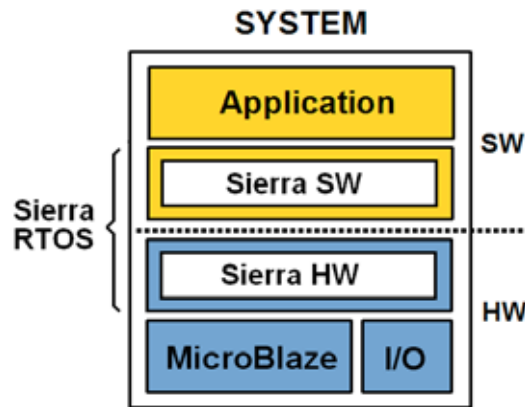
Nucleus. FPGA made easy.

Sierra RTOS from RealFast

- The Fastest Real-Time Kernel in the World!

The Sierra RTOS is a **unique** implementation of a real-time kernel in hardware.

Due to hardware implementation, the Sierra can fully utilize hardware benefits: increased performance, parallelism and 100% predictability.



The Sierra hardware together with a small software driver makes a **complete stand-alone RTOS** kernel and can be used in any embedded system. The Sierra is easy integrated with Xilinx Embedded Development Kit to the **MicroBlaze** or **PowerPC**.

>> Key Benefits

- High quality and performance at low cost
- Plug-and-play to Xilinx MicroBlaze and PowerPC.
- Full support of Xilinx EDK.
- Zero kernel overhead, 1-3 instructions per service call and no scheduling overhead.
- Only 2 kb software driver code.
- Scalable micro-kernel structure and plug-in architecture supports: UDP/IP, TCP/IP, file system, multi-processor support etc.
- **Perfect for FPGA design!**

>> Suitable Target Systems

- Highly suitable for FPGA designs since access time between CPU and peripherals can be minimized.
- Small/medium sized embedded systems where performance and predictability is important.
- Old systems where higher performance is needed at a low cost
- Systems that uses the RTOS extensively.
- MicroBlaze/PowerPC + Sierra RTOS – a complete system can be built with a Xilinx 200K-gate Spartan-3 FPGA

>> RealFast provides:

- Sierra RTOS support and optimized customizations.
- Worldwide courses in HW/SW System Design with Xilinx EDK for single- and multi-processor systems
- Sierra demo version available now at: www.realfast.se/xilinx

RealFast

More information:
www.realfast.se/xilinx
 Susanna Nordstrom
susanna.nordstrom@realfast.se

Building Reliable and Upgradable Software-Defined Radios

The QNX Neutrino RTOS provides the dynamic upgradability, fault tolerance, and hard real-time capabilities demanded by mission-critical SDR devices.

by Paul N. Leroux
Technology Analyst
QNX Software Systems
paul@qnx.com

Copyright © 2005 Xilinx, Inc. All rights reserved.

With their multi-gigabit transceivers, multiplier arrays for parallel signal processing, and reconfigurable logic, Xilinx® Virtex™-II Pro and Virtex-4 FX FPGAs serve as ideal platforms for building flexible and cost-effective software-defined radios (SDRs). For instance, by supporting partial reconfiguration – the ability to configure or reconfigure a portion of an FPGA on the fly – these FPGAs allow a single set of processing resources to support multiple waveforms concurrently. System designers can, as a result, eliminate redundant per-channel hardware and reduce power consumption in their SDR devices.

Until recently, the CPU in most SDR designs existed as a discrete component that communicated with the FPGA fabric over a high-speed interconnect. The Virtex-II Pro and Virtex-4 FX devices take a different approach, immersing high-performance PowerPC™ cores directly into the FPGA. This approach not only provides high-speed access ports between the CPU and the FPGA fabric, but also lowers the component count and frees up board space.

... SDR devices can operate nonstop, even when being upgraded with new waveforms, applications, drivers, or other software components.

Just as important, the embedded PowerPC cores allow SDR designers and software engineers to leverage an array of standards-based, off-the-shelf RTOSs and tool chains, such as the QNX Neutrino RTOS and QNX Momentics development suite.

The term “software” in SDR can be misleading at first. It suggests a device in which software running on the CPU handles functions such as signal generation and detection, modulation and demodulation, encryption and decryption, and signal frequency selection. In reality, most such functions run on signal processors in the FPGA fabric.

Nonetheless, SDR devices must still rely on CPU-based software to download and manage new waveforms for the signal processors, handle voice capture and play-

software engineers achieve these qualities, while significantly reducing development time and effort.

Making the (Up)grade

With Xilinx FPGAs, you can achieve a key objective of SDR: a single, reconfigurable hardware platform that can dynamically adapt to a variety of radio environments. Yet that goal is defeated, or seriously diminished, if the system software running on the FPGA’s PowerPC core doesn’t provide an equivalent measure of upgradability and reconfigurability.

Unfortunately, such dynamic requirements pose a serious challenge to conventional RTOSs. The problem springs from traditional embedded design, where software typically remained stable, with little or no change, over a product’s entire life

critical to building a dynamic and reliable SDR device:

1. The OS kernel contains only a small core of fundamental services, such as timers, messages, and scheduling. All higher level services and programs – drivers, file systems, protocol stacks, and user applications – run outside the kernel as separate, memory-protected components.
2. Most software components communicate through message passing, a well-defined communication mechanism that allows programs to exchange data while remaining safely isolated from each other.

Properly implemented, this message passing also serves as a virtual “software bus” that allows almost any software component, be it a device driver, protocol stack, or application, to be removed, added, or upgraded on the fly. SDR devices can thus support new functionality without system resets or service interruptions.

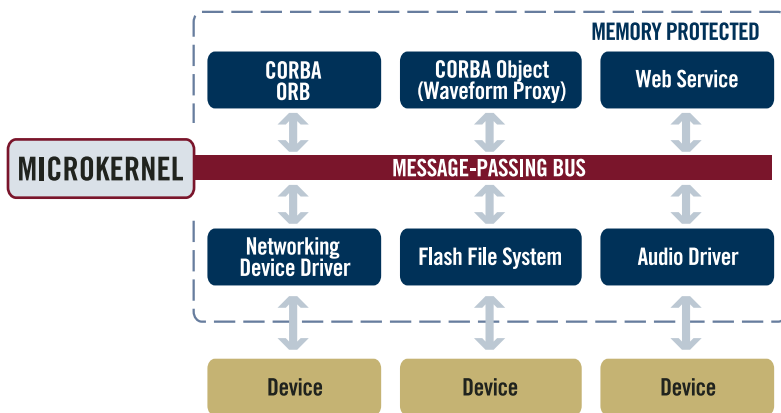


Figure 1 – QNX Neutrino’s microkernel architecture allows software components – including waveforms, device drivers, protocol stacks, and other services – to be upgraded on the fly.

back, manage information displays, and coordinate numerous other command and control tasks. Given the complexity and dynamic nature of the typical SDR device, this software must offer a robust mix of reliability, upgradability, and real-time performance. In this article, we’ll explore how the QNX Neutrino microkernel RTOS helps system designers and

cycle. Having been created for such products, most RTOSs still reflect the old reality. Consequently, deployed systems cannot easily be maintained, upgraded, or extended unless removed from service.

To address this problem, QNX Neutrino uses a true microkernel architecture (Figure 1). Microkernel RTOSs have two defining characteristics – both

Web Services

With QNX Neutrino’s virtual software bus, SDR devices can operate nonstop, even when being upgraded with new waveforms, applications, drivers, or other software components. Nevertheless, one question remains: When a server makes a new waveform or service available, how do remote SDR devices actually discover that component and learn how to use it?

To discover and activate new waveforms and filters, SDR devices can use Common Object Request Broker Architecture (CORBA) middleware. In fact, the U.S. Department of Defense (DoD) mandates the use of CORBA for this purpose in its Joint Tactical Radio System (JTRS) program. An example of CORBA middleware is Objective Interface’s ORBexpress, a highly efficient, real-time implementation that can run on QNX Neutrino.

Web Services Architecture

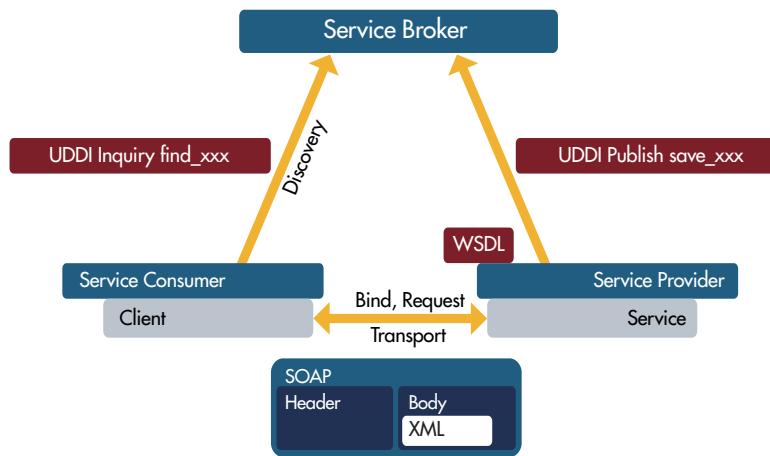


Figure 2 – Using Web services, an SDR device can readily access higher level services over a wide area network.

Designed to let heterogeneous systems interoperate with one another, CORBA provides distributed processing in a single enterprise and over a closed network, and is well-suited to waveform provisioning. However, its inability to readily traverse Internet firewalls makes it unsuitable for accessing higher level services over a wide area network (WAN), such as the Global Information Grid (GIG) envisioned by the DoD. In net-centric operational warfare (NCOW) systems, for instance, multi-function field radios must access new tactical information services through the GIG as those services become available. For such applications, QNX's implementation of Web services offers an ideal solution (Figure 2).

Web services offer full access to Web capabilities (such as the uniform resource identifier [URI] for accessing remote resources) and can work with various security protocols, including HTTP-S and IPSec with strong encryption. Web services are also firewall-friendly – through their well-defined transport binding to HTTP, they can use existing Internet security protocols and be accessed from anywhere on the Internet. And because Web services are based on open standards such as XML, SOAP, WSDL, HTTP, and UDDI, they provide a vendor-neutral, language-independent, and platform-independent method of accessing remote services and data.

Put simply, Web services can span multiple enterprises over a WAN and eliminate the complexities of firewall traversal, allowing a JTRS radio to access services on a GIG that has high levels of VPN security and routing.

To help you take advantage of Web services, QNX has developed a Web services technology development kit (TDK). Using this TDK, embedded developers can implement WSDL, SOAP, and XML-based Web service applications in native C or C++ without having to deal with the complexity of XML and protocol transformations. Moreover, the TDK conforms to the Web Services Interoperability Organization (WSI) Basic Profile 1.0. Applications developed with the TDK can therefore interoperate with other standards-based implementations, such as Microsoft .NET. Just as important, QNX designed the TDK to meet the tight resource constraints of SDR devices and other embedded systems.

Mission: Critical

For most SDR devices, reliability is an absolute must. There is little or no tolerance for any software fault that can defeat the device's main purpose: signaling. If for some reason a software fault does occur, the device should recover from it gracefully, without a system reset.

This requirement poses a challenge to conventional OSs, which bind most system

services (drivers, file systems, protocol stacks) to the OS kernel. With this approach, a single coding error in any system service can corrupt memory used by other services or by the kernel itself, causing system-wide failure.

In comparison, a microkernel OS like QNX Neutrino runs all such services outside of the kernel as separate memory-protected components. This architecture offers two key reliability benefits. First, it makes it much easier to isolate and correct coding errors before the errors can make their way into a deployed system. For instance, if any service or application under development attempts to access memory outside of its process container, the OS will identify the process responsible, indicate the location of the fault, and create a process dump file viewable with source-level debugging tools. The dump file can include all of the information the debugger needs to identify the source line that caused the problem, including a history of function calls, contents of data items, and other diagnostic information. Compare this to a conventional OS, where such errors can crash the entire system without leaving a trace of the cause.

Second, microkernel architecture enables dramatically shorter mean time to repair (MTTR). Consider what happens when, for example, a driver faults in a deployed system: the OS can terminate the driver, reclaim the resources the driver was using, and then restart it, often within a few milliseconds. From start to finish, the entire procedure can be orders of magnitude faster than the conventional solution, which is to reboot the entire system.

The process is made even simpler by QNX's critical processing monitoring technology, which employs "heartbeating" to detect problems before they escalate and allows you to specify the exact recovery actions the OS should follow (Figure 3).

The Real Benefits of Real Time

As a true RTOS, QNX Neutrino offers the hard real-time capabilities needed to coordinate multiple concurrent SDR applications and services. It can, for instance, handle command and control programs, map-based tactical displays (in a JTRS

Allowing a single set of resources to support multiple waveforms is a key advantage of Virtex-II Pro and Virtex-4 FX FPGAs.

radio), multiple modulation and encryption schemes, and various other applications, while ensuring that time-critical tasks (such as transmission of voice packets between the host processor and a signal processor) always occur in a timely and predictable manner.

To enable this predictable behavior, QNX Neutrino offers numerous features, including a priority-based preemptive scheduler, a preemptible kernel, nested interrupts, mechanisms to avoid priority inversion, and a flexible choice of scheduling algorithms. Together, these features help you ensure that high-priority threads meet their

necessary processing capacity. Moreover, QNX Neutrino's real-time capabilities offer greater design flexibility: you have the option to move applications from a DSP to the host CPU or vice versa. With a general-purpose OS, any tasks with real-time constraints have to be delegated to a DSP.

Bred in the Bone

SDR holds the key to dealing with multiple evolving transmission standards. But for a time, SDR was itself in need of common standards that could ensure software reusability and interoperability across various

shelf (COTS) technology, the SCA defines an operating environment based on two existing industry standards: CORBA middleware and a Portable Operating System Interface (POSIX) operating system.

QNX Neutrino represents an ideal operating system for SCA implementations. To begin with, it complies with POSIX 1003.1-2001 and features an extensive set of POSIX options, including real-time extensions and threads. This not only satisfies SCA requirements, but makes it very easy to port Linux, Unix, and other POSIX-based open-source programs to QNX Neutrino's highly efficient real-time architecture. Just as important, the QNX Neutrino microkernel was designed from the beginning to support POSIX – POSIX is “bred in the bone.” Such an approach obviates the need for a complex POSIX adaptation layer used by other OSs, resulting in greater performance and lower memory requirements.

In addition, this robust POSIX support greatly simplifies the task of porting any real-time CORBA implementation, such as ORBexpress, to QNX Neutrino. This is key, as real-time CORBA provides the framework that allows SCA-compliant devices to dynamically learn new waveforms.

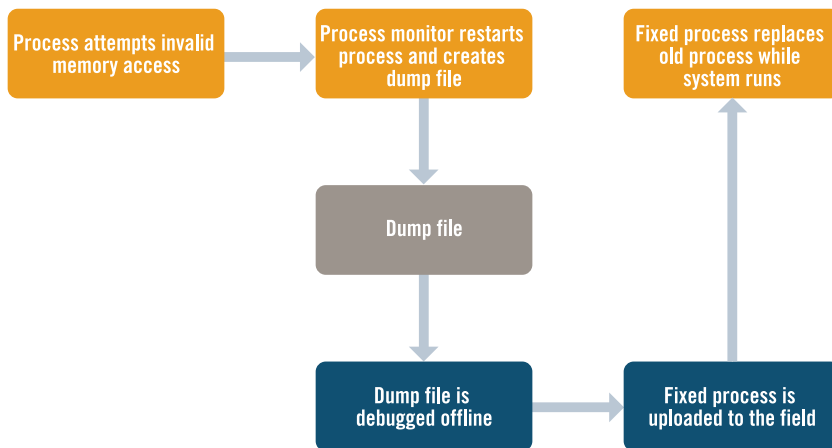


Figure 3 – With its critical process monitor, QNX Neutrino can restart problem components automatically, without downtime or operator intervention. The monitor can also generate a process dump file for postmortem debugging, allowing SDR developers to engineer a fix that can then be uploaded to the field.

deadlines consistently, no matter how many other threads are competing for CPU time.


Allowing a single set of resources to support multiple waveforms is a key advantage of Virtex-II Pro and Virtex-4 FX FPGAs. Likewise, a mature RTOS like QNX Neutrino allows you to maximize processor resources while minimizing overall cost. There is no need to adopt an expensive high-end CPU, along with its attendant thermal dissipation issues, to achieve the

SDR platforms. To address the problem, the DoD's JTRS Joint Program Office (JPO) developed the Software Communications Architecture (SCA) and has mandated its use for all JTRS development projects. The Software Defined Radio Forum has since adopted the SCA, while the Object Management Group (OMG) plans to promote it as a commercial standard.

To ensure waveform portability and compatibility with commercial off-the-

Conclusion

Working closely with Xilinx, QNX Software Systems has developed board support packages (BSPs) for the Virtex-II Pro ML300 evaluation platform, the ML310 Virtex-II Pro development platform, and the ML403 Virtex-4 FX evaluation platform. These BSPs are compatible with the version 6.3 of the QNX Neutrino RTOS and eliminate software/hardware integration issues, allowing you to immediately begin application development.

For more information about these BSPs, or how QNX Neutrino can help enable your SDR project, visit www.qnx.com, or call QNX at (800) 676-0566 (in North America) or +1 (613) 591-0931. 

Take Electronic Motor Drives to the Next Level

When you combine embedded processors on FPGAs with Xilinx motor drive components, there's no limit to what you can achieve.

by Geir Kjosavik
Embedded Processing Marketing Manager
Xilinx, Inc.
geir.kjosavik@xilinx.com

Copyright © 2005 Xilinx, Inc. All rights reserved.

Electric motors are everywhere. I counted 334 of them in my home, and the number keeps growing. If I were to buy a new PC, for example, I'd have to count many more. Two or three motors drive different cooling fans. The DVD-ROM drive and DVD burner each contain four motors: tray control, spindle drive, pickup sledge drive, and laser focus servo. The hard drive has two motors, and even though floppy drives are no longer included, common peripherals such as force feedback joysticks, printers, and scanners more than make up for their loss.

Outside the home, trains, cranes, and electric cars contain motors that can output anything from 100 to several thousand horsepower. Stationary applications contain big motors as well. Industrial compressors, pumps, and fans are just a few examples.

Different applications also require different types of motors. A small stepper motor can position an inkjet printer head with incredible precision, but it takes a monstrous three-phase AC induction motor to drive a city water supply pump.

For almost every motor, an electronic motor drive circuit controls its speed and torque. The electronics that control the print head motor and the water pump motor are as different as the motors themselves. A \$.50 8-bit microcontroller can drive a stepper motor, while the current drive comprises a few surface-mounted metal-oxide semiconductor field-effect transistors (MOSFETs).

On a water pump motor, the fat copper rails connecting it to the power stage hint at large electric currents. The power stage is based around gigantic insulated gate bipolar transistor (IGBT) transistors that alternate the direction of the current through the motor's three-phase windings.

Serious Computing Power Needed

The computing horsepower required to efficiently control pump motors is almost as impressive as the motor itself – at least to an electrical engineer. At the heart of the motor

A clever motor drive design requires that software, power stage, and PWMs work together in perfect unison.

drive sits a high-performance CPU, and a busy one at that. Among its many tasks is generating the three 120-degree out-of-phase sine waves on which AC induction motors thrive. It varies the duty cycle of the drive currents at high frequency, and the inductive load “sees” the waveform as a smooth sine wave. This is pulse width modulation (PWM).

The sine waves, amplified many times by the power stage, make the motor spin. The CPU controls the speed and torque of the motor by varying the frequency and amplitude of the sine wave, respectively. However, as in an automobile, holding the accelerator in a fixed position doesn't guarantee travel at a constant speed. This is why some clever engineers invented the cruise control, and also why equally clever engineers came up with electronic motor controls.

Throw Some Math in There

In simple terms, the CPU controls the speed and torque of the motor this way:

1. Read speed and torque values
2. Compare with desired settings
3. Calculate increments or decrements
4. Adjust speed and torque using values from #3
5. Repeat series from #1

The most commonly used method to calculate the adjustments in step #3 is known as PID – proportional integration and derivation. PID comprises a series of computations that take the differences between desired and current operation as inputs and spit out suitable adjustments proportional to those differences – small adjustments for small deviations, bigger changes if the motor is really off-kilter. You can't just add or subtract the entire difference, because it takes several control loop iterations for the motor to react to a change in input. Such an approach would lead to

instability, with wildly oscillating values.

The motor drive measures the speed of the motor by decoding signals from an optical or magnetic encoder on the motor's drive shaft. A state machine known as quadrature encoder interface (QEI) decodes these signals. Measuring phase currents at certain points in the PWM cycle gives you the torque.

You can then feed other parameters to the control algorithm, such as voltage and current waveform profiles. If you take the latter and throw complex math at it, such as an algorithm bearing the “Star Trek”-like name of flux vector control, you can predict the motor's behavior very accurately without an opto-mechanical or magnetic interface. This is useful when controlling smaller motors, where gadgets like shaft encoders take up too much space and cost more than the incremental CPU power to eliminate them.

Most industrial motor drives are also connected to some kind of network, so when you add the computational powers required to run a TCP/IP or DeviceNET stack on top of everything, performance requirements can reach as much as several hundred MIPS – if you solve your embedded computing needs with brute force. Like hunting sparrows with a rocket launcher, it works, but there are more elegant approaches to meeting performance requirements.

Look Ma, No Brushes

The AC induction motor is generally considered the workhorse of all industrial motor types. Another popular motor for high-performance applications is the brushless DC motor (BLDC), also known as the permanent magnet AC motor. The apparent confusion arises from the fact that it is built on the principles of a DC motor but operates from an alternating current. The BLDC motor doesn't need a sine wave, but the motor drive must know its exact rotational position at any time, thus calling for a dif-

ferent position encoding/decoding mechanism than AC induction motors. When it comes to controlling currents and voltages, PWMs are still the name of the game.

Doing It My Way

Because you cannot build a high-performance motor drive without PWMs, many embedded processors and microcontrollers come with PWMs built-in. These PWMs are the result of long discussions between the chip vendor's marketing and engineering departments. It's a battle between customer requirements on one side and die size estimates, design time, and testability on the other. The resulting design is a compromise between what the “average” customer wants and what can be done within the confines of cost and time-to-market requirements.

A Square PWM in a Square Hole

PWMs come with different features and settings: varying resolution and speed, edge-aligned versus center-aligned, dead time generation, and symmetrical outputs. Most on-chip modules offer software configurations of these parameters, but more often than not, off-the-shelf modules will not meet your exact requirements. Even if they did, that technology would also be available to your competitors. Using a standard module often requires limiting adaptations to the power stage and control software.

A clever motor drive design requires that software, power stage, and PWMs work together in perfect unison. In other words, you need full freedom in designing all three, which effectively rules out on-chip PWMs.

FPGAs to the Rescue

Custom PWMs = custom logic = FPGAs. With an FPGA, you can also do custom QEI interfaces, an ADC interface, safety circuitry, and timer arrays, all designed exactly the way you want them.

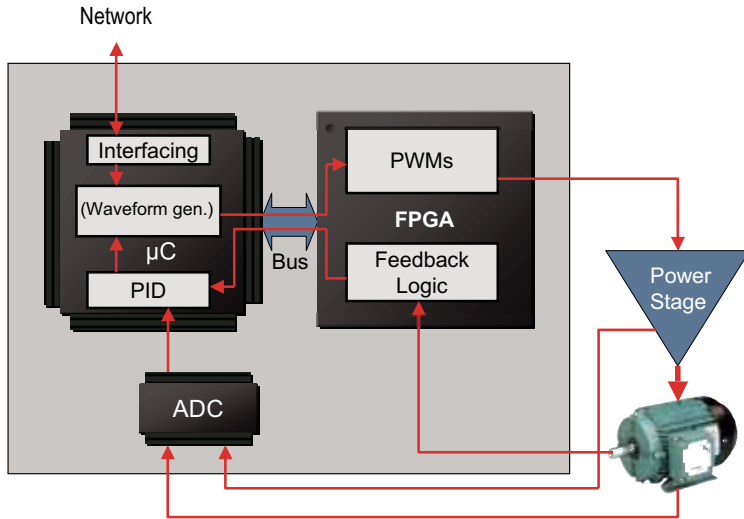


Figure 1 – Block diagram traditional motor drive design

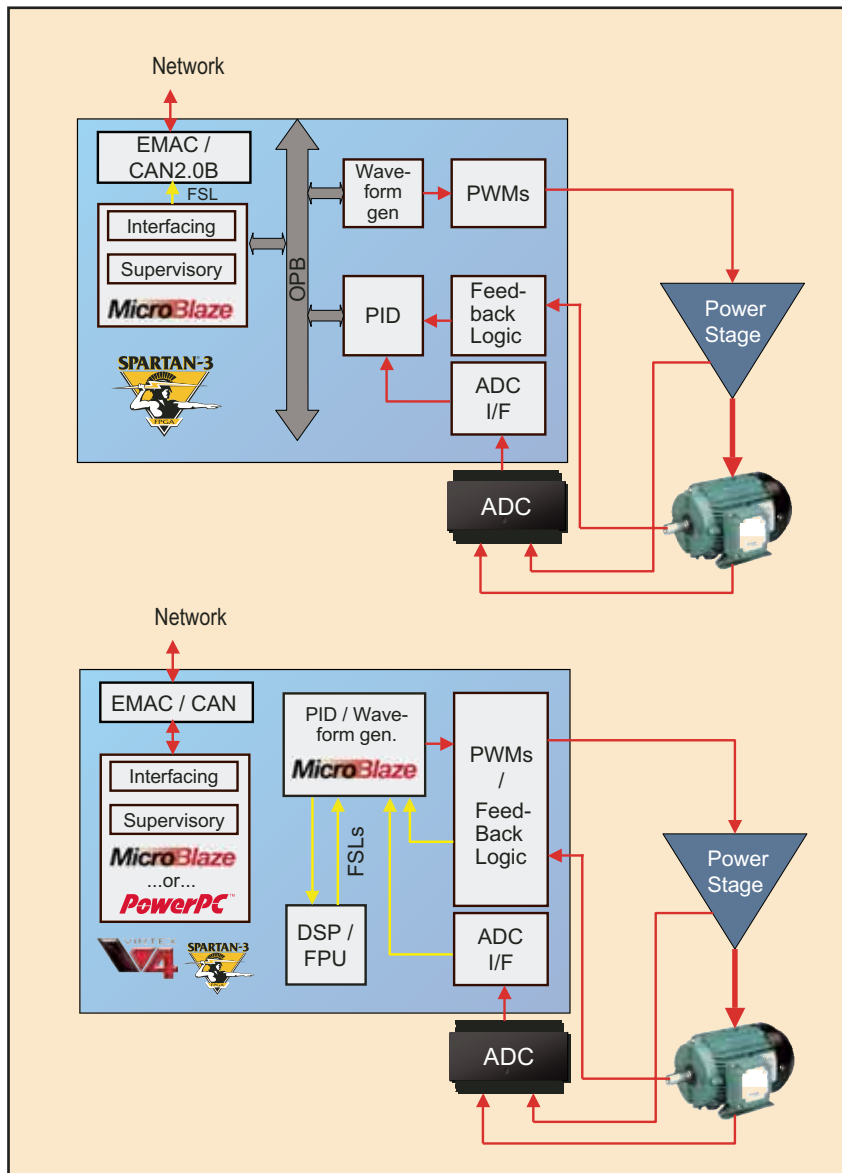


Figure 2 – Two-block diagram of embedded FPGA design

The Next Step: Embedded Integration

The traditional high-performance AC induction motor drive comprises an FPGA and an embedded processor, as shown in Figure 1. This configuration works well, except that the control loop traverses the bus between the two components twice. Because this bus is often shared with other system functions, performance is indeterministic and easily becomes an unnecessary bottleneck in loop response time.

Xilinx® embedded technology allows you to bring the embedded processor onto the FPGA. The benefits of this approach go beyond fewer components, smaller size, and fewer suppliers. It also improves both performance and design time.

Co-Processor Interface

It does not matter whether you choose the 32-bit MicroBlaze™ soft-core processor or hard-coded PowerPC™ 405 processor. Both offer some unique benefits through circuitry dedicated to interfacing with on-chip peripherals in the FPGA fabric. Figure 2 shows some design examples in which all control loop data travels to and from the CPU over a dedicated, deterministic link that is not shared with any other resources.

Eliminating bottlenecks in a design is like peeling away the layers of an onion. You design around one hurdle, and the next one reveals itself immediately. Now that an important hardware pipe has been effectively unclogged, you should not be surprised to see a potential for improvements in your software. Figure 3 shows the software analyzer that comes packaged with the Xilinx Embedded Development Kit (EDK). This tool could prove exceptionally helpful in identifying resource-consuming functions. Motor control software, like anything else, is likely to have its share.

Intelligent Versus Brute Force

I have already mentioned the brute-force approach to designing embedded systems. FPGA processors with a co-processing interface effectively put an end to throwing raw MIPS at any performance

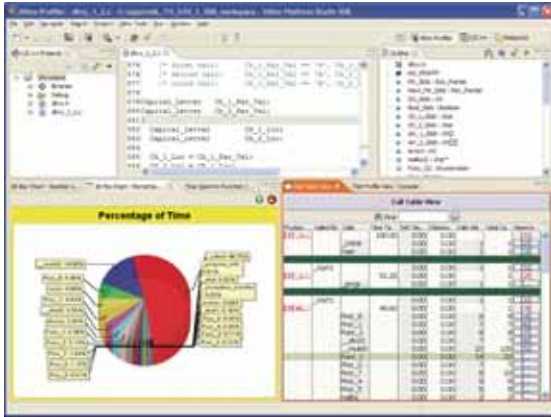


Figure 3 – Software profiling screen shot

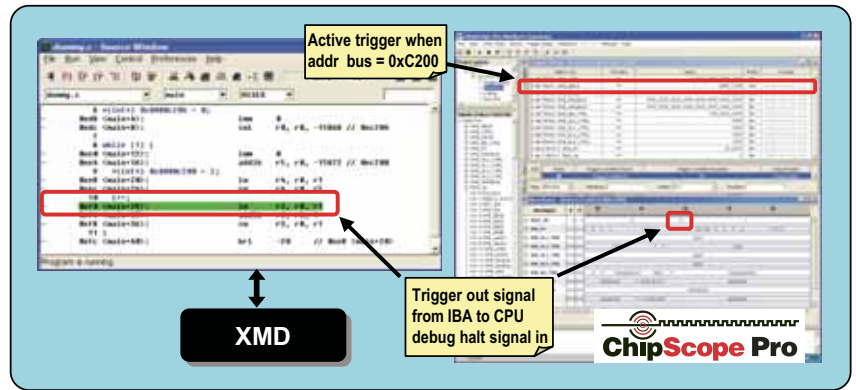


Figure 4 – ChipScope screenshot

challenge. A carefully designed hardware/software mix can reduce CPU performance requirements by several orders of magnitude.

Typical software bottlenecks in a drive are floating-point math functions, DSP functions, and of course, the PID function. As the designer, you should define the optimum mix between hardware and software modules. The FPGA gives you full freedom to balance this any way you want.

A New World of Debugging

The traditional design split of an embedded processor plugged onto an FPGA can be a nightmare to debug. With two different sets of debugging tools probing two different chips, visibility of interaction between the two is extremely limited.

The Xilinx ChipScope™ Pro analyzer lets you debug software and hardware inter-

action like never before. You can set the debugger to trigger when the base counter inside the PWM reaches 0x3FF and observe what code lines were executed within a 50 μs window around that point – without halting execution. Or set the debugger to trigger on each Phase X zero crossing in the sine table and observe the resulting PWM waveform over the next 5 ms.

With full visibility of any chain of events within the system, you can track down even the most elusive bugs in minimum time.

Motor Control Components Ready to Go

As much as we at Xilinx recognize the fact that you are the drive design experts, we do provide a few bits and pieces to give you some ideas of what's possible with embedded processing on FPGAs.

Our reference design, “Spinning Wheels,”

contains complete implementations of one BLCD and one AC induction motor drive. The IP of the solution is openly available as VHDL source code:

- BLDC drive unit
- Hall effect BLCD decoder
- AC induction drive unit
- QEI

The solution comes packaged with reference designs that include MicroBlaze integration with C code examples, tutorials, application notes, and simulation test benches if you don't have a power stage and a motor handy at all times.

Take a look at the blocks and play with them to get a feeling for what's possible. Once you have pieced them together, feel free to sprinkle the design with an FPU and a CAN2.0B interface, or whatever it takes to make it the perfect drive.

Conclusion

Motor control design is tough work. The quest for smarter, faster, and more power-efficient designs imposes an enormous strain on software and hardware engineers. By adopting FPGAs with on-chip embedded processors, you are free to implement the creative design ideas needed to pull ahead of your competitors. With full flexibility in PWM design and control over the hardware/software split in the control functions, the possibilities are endless. ●●

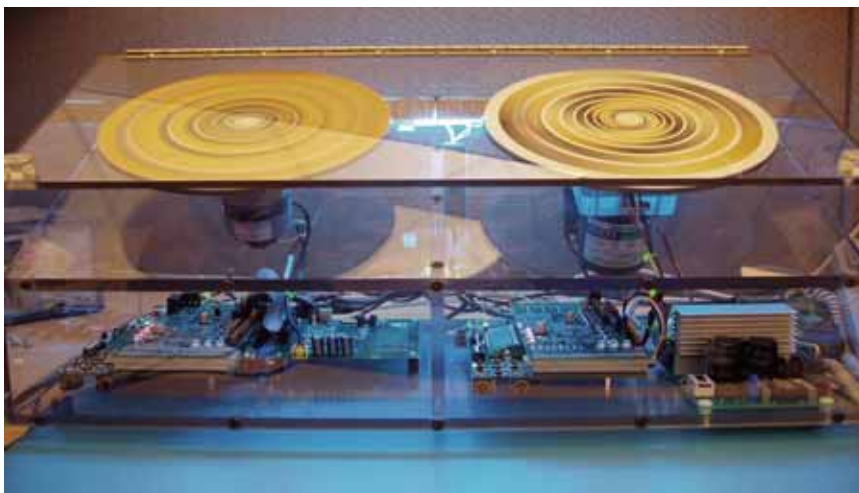


Figure 5 – Spinning Wheels reference design

PARTNERING FOR SUCCESS

XILINX AND BIRGER ENGINEERING

The Xilinx embedded processing solution enables Birger Engineering to develop the Vision Processor Unit (VPU) for a Personal Satellite Assistant (PSA).

THE CHALLENGE

Design a low-power and cost-effective Vision Processor Unit (VPU) for a Personal Satellite Assistant (PSA) that processes all image data at a rate of at least 15 frames per second from eight monochrome cameras and one color camera.

THE SOLUTION

The Xilinx® Virtex™ -II Pro embedded PowerPC™ processor, wide range of processor IP, and Platform Studio tool suite in the Embedded Development Kit (EDK) achieve increased system integration and cost-efficiency.

THE RESULTS

- Processor performance requirement met with the PowerPC
- Strict power budget satisfied
- Functionality of an entire stack of graphic cards implemented in a single FPGA

The Personal Satellite Assistant (PSA) is a volleyball-sized robot designed to operate on a wide variety of spacecraft. The goal is to provide astronauts with a robot assistant to help them with their daily tasks, monitor the environment on the space vehicle, and to venture into situations that might be too dangerous for humans. Central to the operation of the PSA is the Vision Processor Unit (VPU). The Xilinx embedded processing solution – which includes a PowerPC processor in a Virtex-II Pro FPGA – enabled Birger Engineering to develop a low-power, cost-effective VPU for a PSA.

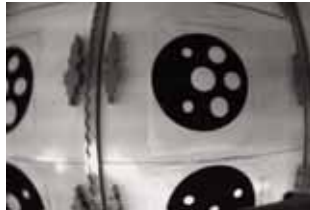
Although it may sound like the sort of curious technology we've come to expect from our favorite sci-fi programs, the idea of creating a self-propelled, self-navigating, free-floating robotic sphere presents some interesting possibilities for the unique environmental conditions encountered in space. On the practical side, having a hands-free "eye-and-nose-in-the-sky" keeps the folks back on earth and up in space constantly apprised of critical environmental conditions. Using its four pairs of monochrome cameras, the floating robot provides constant visual monitoring wherever it goes, and its various gas-sniffing sensors constantly check for gas leaks and other atmospheric anomalies that could prove life threatening if left undetected and unattended.



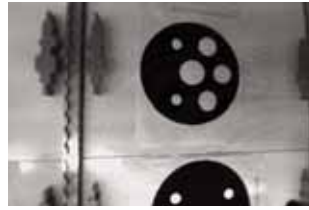
The PSA also provides a sort of free-roaming video conferencing facility. Effectively a "PDA with wings," the images captured with its single color camera are compressed by a JPEG module in the VPU and transmitted over a wireless Ethernet link to enable real-time visual teleconferencing.

“The real challenge was satisfying the cost and power budgets within the constraints of the design schedule. That’s where using the Xilinx Virtex-II Pro FPGA and the Platform Studio tool suite in EDK really paid off.”

Erik Widding
President
Birger Engineering



Camera exhibiting fish-eye distortion



Rectified image with distortion removed

APPLICATION AREAS

Develop a low-power, cost-effective VPU for a PSA in a single Virtex-II Pro platform for programmable systems.

PRODUCTS USED

- PowerPC in a Virtex-II Pro FPGA
- System ACE™ Compact Flash
- Xilinx Platform Studio tool suite EDK
- ISE Foundation™ software
- ChipScope Pro analyzer

The PSA explores new technological possibilities and challenges that may create value in as-yet-undiscovered ways. Using a combination of sophisticated image processing and analysis techniques – and some surprisingly simple, barcode-like location identification concepts – the PSA manages to keep track of where it is at all times while avoiding collisions with both static and moving objects.

“We were pretty confident that we understood the problem well enough from an algorithmic perspective,” said Erik Widding, president of Birger Engineering.

The Birger Engineering design team built the VPU on a PC/104-Plus form-factor PCB using a single Virtex-II Pro 20 FPGA. Sequenced video data from the cameras passes from a frame grabber into the common SDRAM bank. Then a “de-warping” module implemented in one of the on-chip PowerPC processors retrieves the video data from memory in 640 x 480 pixel VGA-sized frames to remove the “fish-eye” distortion inherent to cameras used on the PSA. This has to be done to enable the PSA to accurately analyze the image data.

The “de-warped” data is then output back into memory in pairs of quarter VGA frames (320 x 240 pixels) in preparation for the next module – a stereo disparity module – which searches for textural matches in the images. The disparities of these matching blocks are representative of the parallax effect and can be used to triangulate the actual distance to the object in view. The output of the stereo disparity module takes the form of a depth map.

While all of this image processing is occurring, the same data that was passed to the de-warping

module is also fed from the SDRAM bank to a module called a “blob-finder” that is used to read a series of coded circular fiducials to identify the PSA’s exact location.

The complexity of the various algorithms notwithstanding, the real challenge in designing the VPU was providing all of the various modules timely and manageable access to the single memory bank over the processor local bus (PLB). The design team found the Platform Studio tool suite in EDK invaluable in unraveling the complexities of getting these modules (each of which is a PLB master) to operate independently, and then to interact as intended while passing the equivalent of a few hundred VGA frames back and forth across the PLB – every second.

By simply commenting out a few lines of code in the microprocessor hardware specification (MHS) file, which is a description of the system generated by XPS, the team was able to arbitrarily change the structure of the system to isolate and validate various module combinations, until finally arriving at a fully implemented design that achieved nearly 70 percent utilization of the PLB.

By using the Platform Studio tool suite in EDK and the PowerPC in the Virtex-II Pro FPGA, Birger Engineering developed a low-power, cost-effective VPU for a PSA.

About Birger Engineering, Inc.

Birger Engineering, Inc. (Boston, MA) is a full-service product development firm specializing in the development of algorithms, electronics, and systems for imaging. Additional information about Birger Engineering is available at www.birger.com.

Corporate Headquarters

Xilinx, Inc.
2100 Logic Drive
San Jose, CA 95124
Tel: (408) 559-7778
Fax: (408) 559-7114
Web: www.xilinx.com

European Headquarters

Xilinx, Ltd.
Citywest Business Campus
Saggart,
Co. Dublin
Ireland
Tel: +353-1-464-0311
Fax: +353-1-464-0324
Web: www.xilinx.com

Japan

Xilinx, K.K.
Shinjuku Square Tower 18F
6-22-1 Nishi-Shinjuku
Shinjuku-ku, Tokyo
163-1118, Japan
Tel: 81-3-5321-7711
Fax: 81-3-5321-7765
Web: www.xilinx.co.jp

Asia Pacific

Xilinx
Unit 1201, Tower 6, Gateway
9 Canton Road
Tsimshatsui, Kowloon,
Hong Kong
Tel: 852-2-424-5200
Fax: 852-2-494-7159
E-mail: ask-asiapac@xilinx.com

 **XILINX®**
The Programmable Logic Company™

FORTUNE 2004
100 BEST COMPANIES TO WORK FOR

© 2005 Xilinx Inc. All rights reserved. The Xilinx name is a registered trademark; Virtex-II Pro is a trademark; and The Programmable Logic Company is a service mark of Xilinx, Inc. PowerPC is a trademark of International Business Machines Corporation in the United States, or other countries, or both. All other trademarks are the property of their owners.

PARTNERING FOR SUCCESS

XILINX AND LG ELECTRONICS

A Xilinx embedded processing solution helps LG Electronics develop a lower cost WCDMA transceiver card

THE CHALLENGE

Design a more cost-effective transceiver card for WCDMA (wideband code-division multiple access) base stations that delivers enhanced processing functions while reducing system cost.

THE SOLUTION

Leverage the Xilinx® Virtex™ -II Pro embedded PowerPC™ processor, wide range of processor IP, and Embedded Development Kit to realize increased system integration and cost-efficiency.

THE RESULTS

- 85 MHz PowerPC performance
- Easier debugging of the diagnostic module
- Minimal impact on FPGA utilization
- Reduced component cost and PCB area

The transceiver card (transceiver/receiver module) in a WCDMA base station is an extremely complex board that can add significant expense to the overall

system solution. The Xilinx embedded processing solution – which includes a PowerPC processor in a Virtex-II Pro FPGA – enabled LG Electronics to realize significant functionality improvements while dramatically reducing cost.

As of June 2004, 30 WCDMA 3G networks and 12 CDMA2000 1xEV-DO 3G networks were operating in Europe, Japan, South Korea, the United States, and South America. By the end of 2004, Europe deployed 12 more WCDMA 3G networks, and at least five more CDMA2000 1xEV-DO 3G networks began operating in other global regions. The deployment of 3G networks is obviously ramping up and will represent the majority of wireless carrier capital spending by 2005. The slowdown of the equipment industry in the past few years has also made the 3G equipment market extremely competitive, with cost-effectiveness becoming the number-one selection criteria of infrastructure.



Production board for WCDMA BTS transceiver module

A WCDMA base station consists of amplifiers, transceiver/receiver module, base-band processing module, networking interface module, and main processing module. A critical element of this infrastructure is the transceiver card, a complex board that performs digital up/down conversion, digital filtering, antenna diversity, and bridging between the baseband-processing module and amplifiers. Because of this complex functionality, LG Electronics used an external processor to control test data flow of the diagnostic function.

“The Xilinx embedded processing solution ... helped LG Electronics implement functions into a programmable system that we had never thought of.”

WanRae Kim
Senior Engineer
Node B Systems
LG Electronics



www.xilinx.com

XILINX®

The Programmable Logic CompanySM



Test-data flow of diagnostic function in previous design



Test-data flow of diagnostic function in improved design

LG Electronics replaced an expensive, external processor with the PowerPC within the Virtex-II Pro FPGA to realize significant performance improvements and lower costs.

APPLICATION AREAS

- Design an integrated transceiver module with test-data flow controller, digital up/down conversion, and digital filter in a Virtex-II Pro FPGA.

PRODUCTS USED

- PowerPC in a Virtex-II Pro FPGA
- EDK
- Xilinx processor IPs

LG Electronics revisited its existing architecture to better meet these cost/performance demands and found that Xilinx offered the ideal solution. After reviewing the Xilinx Embedded Development Kit and the wide variety of Xilinx processor IP options, LG Electronics chose the PowerPC processing power within the existing Virtex-II Pro FPGA to implement the functions that the external processor had performed in their previous designs.

By implementing this functionality through the Xilinx embedded design methodology, LG Electronics was able to run the PowerPC at 85 MHz with user-defined RAM controller, UART, and GPIO. Plus, the code size was only 90 KB, which allowed the FPGA to continue to run at 70% utilization. This increased system integration not only reduced the cost of materials, but also simplified the debugging of the diagnostic module through GDB. LG Electronics had found its cost-effective solution for high-performance WCDMA transceiver card design.

About LG Electronics

LG Electronics (LGE) is a major global player in electronics and telecommunications, operating 72 subsidiaries around the world with over 55,000 employees worldwide. LGE focuses on cell phone, wire and wireless telecom, digital TV, CD-RW, DVD, CD-ROM, DVD-ROM drives, PCs, monitors, mobile handsets, CRTs, and PDPs. LGE is continually strengthening its core competencies further its reputation as the "Digital Leader" in electronic products and equipment for the digital era.

Corporate Headquarters

Xilinx, Inc.
2100 Logic Drive
San Jose, CA 95124
Tel: (408) 559-7778
Fax: (408) 559-7114
Web: www.xilinx.com

European Headquarters

Xilinx, Ltd.
Citywest Business Campus
Saggart,
Co. Dublin
Ireland
Tel: +353-1-464-0311
Fax: +353-1-464-0324
Web: www.xilinx.com

Japan

Xilinx, K.K.
Shinjuku Square Tower 18F
6-22-1 Nishi-Shinjuku
Shinjuku-ku, Tokyo
163-1118, Japan
Tel: 81-3-5321-7711
Fax: 81-3-5321-7765
Web: www.xilinx.co.jp

Asia Pacific

Xilinx
Unit 1201, Tower 6, Gateway
9 Canton Road
Tsimshatsui, Kowloon,
Hong Kong
Tel: 852-2-424-5200
Fax: 852-2-494-7159
E-mail: ask-asiapac@xilinx.com

 **XILINX**[®]
The Programmable Logic CompanySM

FORTUNE 2004
100 BEST COMPANIES TO WORK FOR

© 2005 Xilinx Inc. All rights reserved. The Xilinx name is a registered trademark; Virtex-II Pro is a trademark; and The Programmable Logic Company is a service mark of Xilinx, Inc. PowerPC is a trademark of International Business Machines Corporation in the United States, or other countries, or both. All other trademarks are the property of their owners.

PARTNERING FOR SUCCESS

XILINX AND PHOTONIC BRIDGES

A Xilinx embedded processing solution helps Photonic Bridges develop a higher performance, lower cost RPR solution

THE CHALLENGE

Design a high-performance, reliable, lower cost RPR solution for the MetroWave product family.

THE SOLUTION

Leverage the Xilinx® Virtex™ -II Pro embedded PowerPC™ processor to offload the RPR algorithm from a stand-alone processor.

THE RESULTS

- Photonic Bridges uses the Virtex-II platform for programmable systems to deliver MetroWave multi-service transport platform (MSTPs) on time
- The product is fully compliant with the newly approved RPR standard
- The high level of system integration greatly simplifies PCB design

Building state-of-the-art optical network equipment with emerging resilient packet ring (RPR) features requires high-performance processing power with fast input and output response time.

The Xilinx Virtex-II Pro FPGA with embedded PowerPC processors and multi-gigabit transceivers (MGTs) enabled Photonic Bridges Inc to meet the challenging specifications and deliver RPR technology on time and on budget.

Known as one of “The 10 Hottest Technologies for 2004*,” RPR is a demanding new feature in state-of-the-art optical networking equipment. Telecom equipment vendors want to ship RPR-ready equipment as soon as possible.

To build new RPR features into their MetroWave MSTP products, Photonic Bridges needed to address challenges in RPR specifications, such as fairness algorithms, spatial reuse, and sub-50 ms fail-over restoration time. MetroWave MSTPs are leading-edge products that enable telephone companies to use their highly reliable voice-oriented networks to economically transport and manage a variety of high-growth data services.



SDH RPR processing card

Designing RPR through traditional methods requires designers to separate SW implementation of complex algorithms in stand-alone processors and place the fast-response portion in hardware logic. However, this methodology is risky and often costly. The nature of RPR requires considerably faster communication between hardware and software, which increases the communication challenges among software, firmware, and hardware designers. At the same time, high-performance processors for algorithm computing add significant cost to the system.

“*The Xilinx platform for programmable systems enabled Photonic Bridges to design the demanding technical features of RPR into our MetroWave MSTP products in record time. The Virtex-II Pro FPGA was able to lower our total system cost as well.*”

Lucas Hsu
VP of Research and Development
Photonic Bridges



www.xilinx.com



The Programmable Logic CompanySM



APPLICATION AREAS

Design complex RPR algorithm and high-speed SDH backplane interface in a single Virtex-II Pro platform for programmable systems.

PRODUCTS USED

- PowerPC in a Virtex-II Pro FPGA
- MGT transceivers in a Virtex-II Pro FPGA
- EDK

To lower design risk and cost, Photonic Bridges turned to Xilinx Virtex-II Pro FPGAs with embedded PowerPC processing and MGT technologies. In their design, Photonic Bridges off-loaded the RPR algorithm from the stand-alone processor into the Xilinx Virtex-II Pro FPGA's PowerPC. This not only satisfied their algorithm processing needs, but it also significantly shortened the communication time between hardware and software. Plus, system engineers didn't have to worry about potential performance degrade, which happens in the traditional design method. As a result, the expensive high-performance external processor was replaced by the Xilinx Virtex-II Pro FPGA.

Additionally, in the same FPGA, Photonic Bridges utilizes four MGTs for SDH backplane interfaces. By increasing the level of system integration, Photonic Bridges reduced system complexity and increased system stability.

About Photonic Bridges

Photonic Bridges, founded in California's Silicon Valley in 2000, designs and produces advanced optical telecommunication equipment focused on the Metro market segment, the key growth area in the telecommunications/data market. By combining western-style technology, innovation, and management philosophy with Chinese service orientation and low-cost manufacturing, Photonic Bridges is developing as a world-class telecommunications equipment provider. Photonic Bridges' headquarters, along with R&D, manufacturing, and logistic support, are located in Shanghai. Sales support operations are in Beijing, Shanghai, Shenyang, and Guangzhou, with additional offices to support its growing business planned. For more information about Photonic Bridges, please visit www.photonicbridges.com.

*Telecommunications Magazine at SUPERCOMM 2004

Corporate Headquarters

Xilinx, Inc.
2100 Logic Drive
San Jose, CA 95124
Tel: (408) 559-7778
Fax: (408) 559-7114
Web: www.xilinx.com

European Headquarters

Xilinx, Ltd.
Citywest Business Campus
Saggart,
Co. Dublin
Ireland
Tel: +353-1-464-0311
Fax: +353-1-464-0324
Web: www.xilinx.com

Japan

Xilinx, K.K.
Shinjuku Square Tower 18F
6-22-1 Nishi-Shinjuku
Shinjuku-ku, Tokyo
163-1118, Japan
Tel: 81-3-5321-7711
Fax: 81-3-5321-7765
Web: www.xilinx.co.jp

Asia Pacific

Xilinx
Unit 1201, Tower 6, Gateway
9 Canton Road
Tsimshatsui, Kowloon,
Hong Kong
Tel: 852-2-424-5200
Fax: 852-2-494-7159
E-mail: ask-asiapac@xilinx.com



FORTUNE 2004
100 BEST COMPANIES TO WORK FOR

© 2005 Xilinx Inc. All rights reserved. The Xilinx name is a registered trademark; Virtex-II Pro is a trademark; and The Programmable Logic Company is a service mark of Xilinx, Inc. PowerPC is a trademark of Internal Business Machines Corporation in the United States, or other countries, or both. All other trademarks are the property of their owners.

High Performance Internet

Leave your competition behind with **Treck Inc.**



Treck's suite of TCP/IP protocols is designed specifically for simple integration into embedded systems. Xilinx FPGA processors provide an optimal platform for the wide range of Treck's fully RFC compliant Internet protocols. The powerful combination of Treck software and Xilinx technology delivers TCP performance more than six times faster than any competing FPGA solution. Treck's support model grants direct access to protocol engineers for all products including IPv4, IPv6, IPSEC, IKE, DHCP, and Web Server.

Treck Inc.

<http://www.treck.com/xilinx.htm>
(513)528-5732 Call for Free Demo

threadx, making things better at work

41st floor

Jim's sales presentation is printed in vivid color on his *Ink-Jet Printer*

43rd floor

ABC hosts a worldwide conference with their *VoIP Router*

39th floor

Tim and Pam see their new baby for the first time on their doctor's *Sonogram*

33rd floor

Jane accesses her e-mail through an *802.11g Wireless Access Point*

express logic's ThreadX – the RTOS deployed everywhere. at work. at home. at play.

Making things better. This is what Express Logic is all about. Our ThreadX® RTOS is used in over 100 million electronic products that help make your life better everyday—at work, at home, and at play. In fact, you have probably already used a

THREADX

ThreadX product today! How do we make things better? It all starts with our complete focus on true embedded systems. Our ThreadX RTOS is a blend of advanced technology, powerful services, and most importantly, unparalleled ease of use. With real-time

performance, understandable source code, extensive processor support, minimum footprint, a broad choice of development tools and 100% royalty-free licensing, it's no wonder that ThreadX has been chosen over and over. Let Express Logic help you make your next product better with ThreadX.

For more, visit expresslogic.com or call 1-888-THREADX

ThreadX is a registered trademark of Express Logic, Inc., © 2004 by Express Logic, Inc.



Embedded Application Notes

Popular application notes available from Xilinx and our partners.

Copyright © 2005 Xilinx, Inc. All rights reserved.

In this section, we'll break out excerpts from Xilinx® application notes and provide information on how to access the complete articles. The first two application notes are "High Performance TCP/IP on Xilinx FPGA Devices Using the Treck Embedded TCP/IP Stack," by Satish Narayanaswamy of Xilinx (XAPP546) and "UltraController-II: Minimal Footprint Embedded Processing Engine" by Punit Kalra, also from Xilinx (XAPP575).

XAPP546 describes how to use the Treck TCP/IP stack with Xilinx EDK tools and the Gigabit System Reference Design (GSRD) system.

XAPP575 presents the features and benefits of the PowerPC™-based UltraController-II, along with a tutorial of applications included with the design.

XAPP546: High Performance TCP/IP on Xilinx FPGA Devices Using the Treck Embedded TCP/IP Stack

TCP/IP is a popular communications protocol software stack that allows reliable data communications between two hosts. Most people already use TCP/IP to check e-mail, browse the Web, or transfer files. TCP/IP is being used more and more in the embedded world as well.

Treck, Inc. is a leading provider of embedded TCP/IP stacks that allow Xilinx FPGAs to communicate in a wide range of networking environments. Treck's dual IPv4/IPv6 TCP/IP stack provides IPv4 functionality today and allows a Xilinx FPGA to support IPv6 networks of the future. Treck also provides optional protocols, such as an embedded web server, FTP, IPSEC, and DHCP, to enhance the functionality of Xilinx FPGAs.

This application note describes how to get started using the Treck TCP/IP stack



using Xilinx EDK tools. An evaluation version of the Treck TCP/IP stack is included. An example TCP application uses the Treck TCP/IP stack to send TCP data over Gigabit Ethernet on the Virtex™-II Pro ML300 development board to a remote PC-based server.

Introduction

The Treck TCP/IP stack offers a high-performance TCP/IP software solution that can be used with the PowerPC™ 405 processor inside the Virtex-II Pro series of Xilinx FPGAs.

Some features of the Treck TCP/IP stack include:

- Zero-copy send and receive, which help deliver maximum throughput for bridging applications
- Jumbo frames support (in the case of Gigabit Ethernet devices)
- TCP checksum offload support for devices that support TCP checksum offload in hardware

- Fully RFC-compliant TCP/IP stack for maximum interoperability
- Standard sockets interface API

The Treck TCP/IP stack can be used with or without any operating system software. This application note discusses the use of the Treck TCP/IP stack on a stand-alone system (without an operating system).

This application note also provides the Treck library as a binary file for evaluation purposes. The Treck library allows full functionality of the stack for a limited period of time before it times out and requires a restart to continue evaluation.

Contact Treck at www.treck.com for information about purchasing the sources for the Treck TCP/IP stack. An example TCP client and server application is also available as part of this application note. Xilinx EDK tools are used to compile and link the client application with the Treck library to create a complete TCP/IP application for the ML300 board.

The client application uses Treck TCP/IP on the ML300 board to transmit TCP data to a remote PC. The server application running on the PC prints the TCP throughput every second on the console. The sources, as well as Windows and Linux binaries, are included for the server application.

The reference design files can be downloaded from the Xilinx website.

The Treck embedded TCP/IP stack is well suited for TCP/IP applications running on Xilinx FPGAs. Its support of zero-copy applications and checksum offload in hardware is utilized in a high-performance architecture like GSRD. Treck also offers different protocols and applications like IPSEC, IPV6, HTTP, and Telnet. The combination of the Treck TCP/IP stack and the flexible Xilinx FPGA hardware

platform offers an ideal solution for TCP/IP termination at high data rates.

For more of XAPP546, you can access the unabridged application note for “High Performance TCP/IP on Xilinx FPGA Devices Using the Treck Embedded TCP/IP Stack” at www.xilinx.com/bvdocs/appnotes/xapp546.pdf.

XAPP575: UltraController-II: Minimal Footprint Embedded Processing Engine

UltraController-II is a minimal footprint Virtex-II Pro embedded processing engine based on the embedded PowerPC 405 (PPC405) processor core. Computing performance is maximized and FPGA resource usage minimized by running code strictly from the integrated PPC405 caches. General-purpose input/output (GPIO) is available directly from the PPC405 core. Interrupt handling is provided for a user-defined external interrupt line, a programmable interval timer (PIT), and a fixed interval timer (FIT). You can easily incorporate the UltraController-II black-box processing engine into larger ISE™ software designs to gain additional degrees of freedom by balancing the high performance of FPGA fabric with the algorithmic flexibility of software.

This application note presents the features and benefits of the UltraController-II, along with a brief overview of the tutorial applications included with the design. The accompanying tutorials and reference designs include VHDL, Verilog, and example C-code applications with step-by-step procedures. XAPP575 also provides performance characteristics and describes how to field an UltraController-II system using Xilinx configuration solutions.

Introduction

The UltraController-II reference design is a black-box processing engine that includes 32 bits of user-defined GPIO as well as interrupt handling capability. UltraController-II applications are developed within the 16 KB instruction- and data-side cache memory of the PPC405 core. If you are developing embedded designs that require PLB and OPB bus resources, the Xilinx Platform Studio (XPS) toolset creates expandable processor designs that leverage the full set of IP and software drivers offered by Xilinx (see www.xilinx.com/ledk).

Features:

- Scalable CPU clock (up to 400 MHz in a Virtex-II Pro speed grade -7 device)
- Integrated cache-based program store
- 16 KB I-side
- 16 KB D-side
- No block RAMs used
- 32-bit output
- 32-bit input
- External user interrupt line (EXT)
- Programmable interval timer (PIT)
- Fixed interval timer (FIT)
- Watchdog timer (WDT)

Benefits:

- Processing power is determined by program execution speed; UltraController-II can be clocked at the maximum PPC405 input clock frequency, which far exceeds any soft-core processor implementation
- Program instruction and data access speeds are maximized by using the integrated caches
- A minimal footprint processing engine frees up FPGA logic and block RAM resources
- 32-bit output and input ports can interface with logic internal or external to the FPGA
- An external interrupt line allows UltraController-II applications to perform high-speed, base-level computation and handle time-critical events as they occur
- Timer resources (PIT and FIT) are available for commonly implemented embedded solutions for:
 - Time-of-day computation
 - Data-logging for system-service routines
 - Periodic servicing of time-sensitive external devices
- A watchdog timer (WDT) is available to monitor system sanity and recover from upsets by issuing a system reset

UltraController-II Internals

In today’s complex and competitive design environment, products must be designed and verified more rapidly than ever before. This can be accomplished by partitioning designs into functional sub-blocks. The UltraController-II solution inherently separates a design into hardware and software functional sub-blocks, or components, due to its cache-based nature.

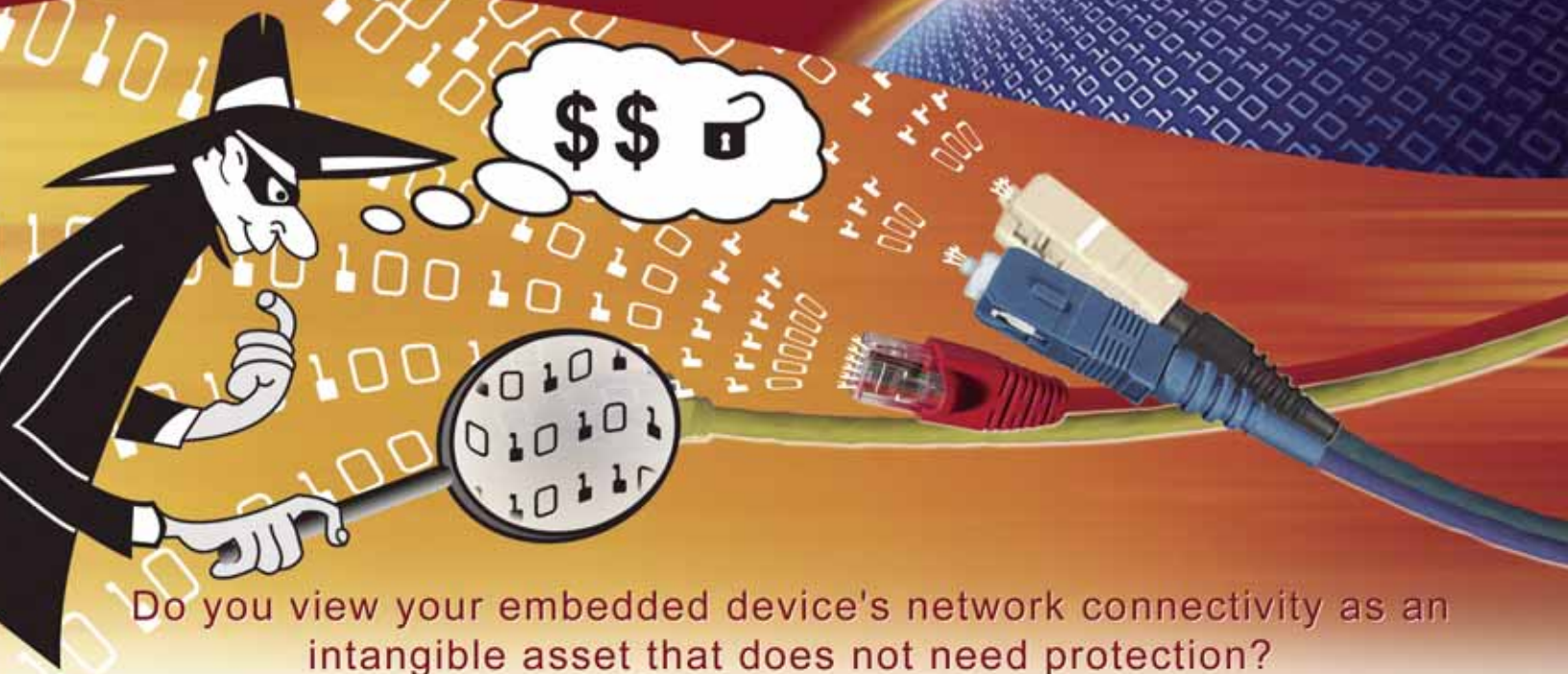
The original block RAM-based UltraController provides you the ability to initialize both the hardware and software components of the system through the bitstream. Compared with traditional embedded systems of similar size, a single bitstream file with both hardware and software components eliminates the need for external non-volatile storage that only contains software.

UltraController-II designs also use a single initialization file. In addition, UltraController-II allows you to independently modify the hardware or software components by taking advantage of Xilinx configuration solutions. You can create multiple software design iterations without modifying the bitstream, thereby limiting the scope of any introduced design changes. Once an UltraController-II black box is integrated into a larger ISE design and verified, the hardware can be locked down to become a “golden” bitstream. This golden bitstream gives you independence from development tool revisions and the ability to reestablish a known hardware state at any time in the future.

UltraController-II offers additional built-in functionality and a reduced resource footprint when compared with UltraController. Program storage for both the instructions and data now resides within the PPC405 caches, thereby eliminating the need for any block RAM. GPIO is available directly from the PPC405 block and provides access to 32 bits of input and output. Exception handling permits you to process a user interrupt line, the PIT, and the FIT interrupts. Reset and boot logic are also covered.

For more of XAPP575, you can access the unabridged application note for “UltraController-II: Minimal Footprint Embedded Processing Engine” at www.xilinx.com/bvdocs/appnotes/xapp546.pdf.

What's your 'Net worth?



Do you view your embedded device's network connectivity as an intangible asset that does not need protection?

Sorry to burst your bubble, but in a world where hackers can take control of connected devices more conveniently than authorized users can, network connectivity could shift from being an asset to being a liability faster than a dot-com stock coming back to ground.

As security is being recognized for what it is -- a multi-dimensional, multi-layered problem requiring application-specific, standardized & interoperable solutions -- it is important to choose a security partner who not only meets your security software requirements for today, but also helps future-proof your security roadmap.

TeamF1 offers a complete OEM-focused security and availability software platform that is pre-integrated with Xilinx processing solutions leveraging high performance cores in the programmable fabric to offload the CPU and boost system performance.

	X.509	Kerberos	RADIUS	Firewall	
SSL / TLS	SSH	IKE / IKEv2	NAT	NAT-T	
TeamF1 Security Framework					
	IPsec	WPA / 802.11i	802.1x	Secure FS	
DoS Resistance	Operating System			L2 redundancy	
Xilinx Virtex™-II Pro / Virtex™-4 FX					

A Broad Portfolio of Embedded Security Products

Take stock of the situation and invest in leading edge security technologies that you can bank on - the long term gains are worth it!

(510) 505-99F1

sales@TeamF1.com

www.TeamF1.com





Mind eCos and Linux on Virtex 4 FX: *the SoC HW/SW platform of the future*

Virtex 4 FX: the SoC platform of the future

- extremely fast development [1]
- a low cost, off-the-shelf chip delivers multiple hard core and soft core CPU's (PowerPC 405, MicroBlaze), hard core Gbit/s Ethernet interfaces, high-speed I/O, DSP blocks, on-chip memory and large amounts of programmable logic
- co-design and co-debugging of advanced electronic designs and high performance SW solutions comes in reach, also for smaller development teams

Linux and eCos: the operating systems of the future

- Open Source operating systems deliver performance, flexibility and low cost
- eCos: hard real-time, small footprint, easy step-up from home-brew or proprietary RTOS solutions
- Linux: high-end, scalable, portable across many platforms

Mind: the SoC development partner of the future

- dedicated to custom engineering, tool chains, support and training for FPGA and Open Source SW solutions
- specialized engineers know the FPGA, the operating system and the interface between the two
- experienced in full tool chain set-up for FPGA and SW development, including an entire tool chain on a single, low cost Linux system, with integrated make tools

[1] In a recent customer project, Mind developed HW and SW for a dual CPU (2 x PowerPC 405) design on Virtex-II Pro (V2P30) with shared memory, running Linux and eCos, in a two week timeframe. This is orders of magnitude faster than equivalent multi-core ASIC design projects on which Mind assisted earlier.



Build and Optimize a MicroBlaze Soft-Processor System Your Way

The Xilinx MicroBlaze solution makes custom processing on FPGAs easy.

by Helen Yu
Product Marketing Manager,
Xilinx Embedded Processing Solutions
Xilinx, Inc.
helen.yu@xilinx.com

Copyright © 2005 Xilinx, Inc. All rights reserved.

Finding a processor to meet performance, feature, and cost targets is very challenging. The Xilinx® MicroBlaze™ soft-processor provides a scalable solution that is fully customizable, area-efficient, and can be optimized for your most cost-sensitive designs.

MicroBlaze v4.00, the newest version of the 32-bit soft-processor core from Xilinx, raises the bar for flexibility and ease-of-use with new user-configurable hardware options, improved debug capabilities, and complete backward compatibility with earlier releases. In this article, you will learn how to increase the performance of a software-based filter design using the MicroBlaze processor and the award-winning Xilinx Platform Studio embedded tool suite.

Build What You Need

The Xilinx MicroBlaze processor is a RISC-based, 32-bit reconfigurable soft-processor that can be customized with different peripherals and memory configurations. Unlike a hard processor, which is implemented in the FPGA at the transistor level, a soft core is an IP block. The MicroBlaze soft processor core is optimized for our FPGA architecture. One of the many advantages of soft-core designs is that they are very configurable. In other words, you do not have to incorporate every available feature. Instead, you can disable any feature that you are not using to save valuable logic resources on your device. As a result, the MicroBlaze solution allows you to tune the processor system architecture to your application.

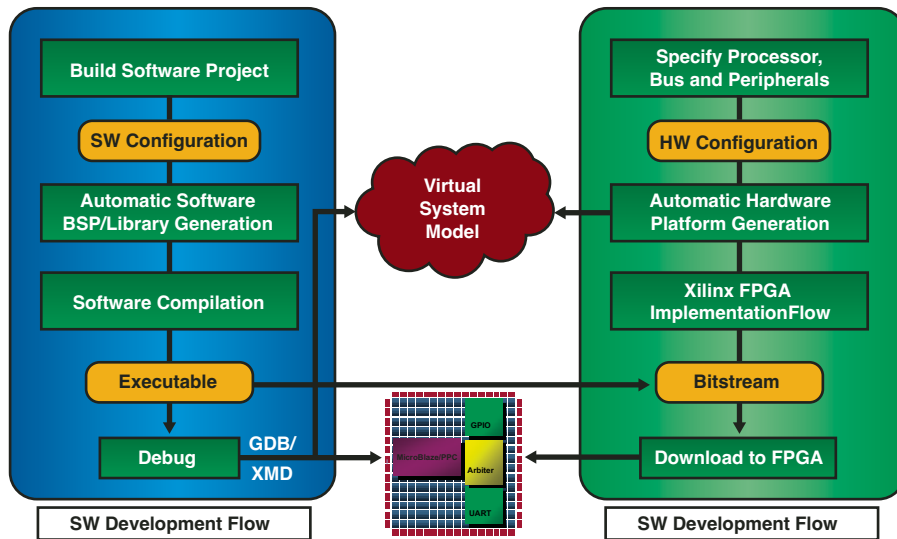


Figure 1 – The Xilinx Platform Studio (XPS) tool flow

form software model and libraries for that system. A new feature in the XPS called the Virtual Platform Generator provides the capability to generate a virtual MicroBlaze platform model. Now you no longer have to prototype on a hardware development board at your desk or even go through the flow of developing that hardware. With Virtual Platform Generator, you can generate the software model, drivers, and libraries of your system and then compile, profile, and execute software on that model.

The Next Step – Profile Your Software Application

By now you have specified your hardware processor system and generated the virtual model and software libraries. The next step is to start a C application project using either XPS or the Eclipse-based XPS Software Development Kit (SDK) tools. Once the software project is built and compiled, you can run the executable and then step the application through the debugger tool.

Profiling in SDK is interactive and provides graphical profile views. In our MicroBlaze design example, the core arithmetic loop function implements a FIR filter function. This small loop of code makes calls to floating-point library functions, which consume a large part of the CPU time in the FIR function call (as shown in Figure 2). How can you avoid calling floating-point libraries? Well, you could convert to a fixed-point operation, but that would be quite time-consuming. Another option is to use a processor with an FPU. The FPU would be able to do these computations natively and thus avoid library calls (as shown in Figure 3).

Conveniently, the new MicroBlaze v4.00 includes an integrated FPU option (depicted in Figure 4). The FPU handles single-precision floating-point arithmetic and is compatible with IEEE-754. Its integration ensures high performance, low latency, and a compact design. Because the MicroBlaze FPU is yet another configurable feature of the MicroBlaze core, it takes no extra space in the FPGA if it is not needed.

Let’s look at how we might improve the performance of our FIR filter by enabling this MicroBlaze FPU.

Getting Started with XPS

The Xilinx Platform Studio (XPS) embedded tool suite provides an integrated environment for creating the software and hardware specification flows for a MicroBlaze system, as shown in Figure 1. In this article, we’ll discuss a MicroBlaze system that uses a floating-point finite impulse response (FIR) algorithm and its performance improvement after adding a floating-point unit (FPU) to the design.

The XPS base system builder (BSB) wizard helps you quickly build a working system targeted at an existing or custom development board. Based on your board selection, the BSB offers you a number of options for creating a basic system on that board, including processor type, debug interface, cache configuration, memory type and size, and peripheral selection. For each option, functional default values are pre-selected in the wizard.

Once you have defined the system architecture, you can then generate the virtual plat-

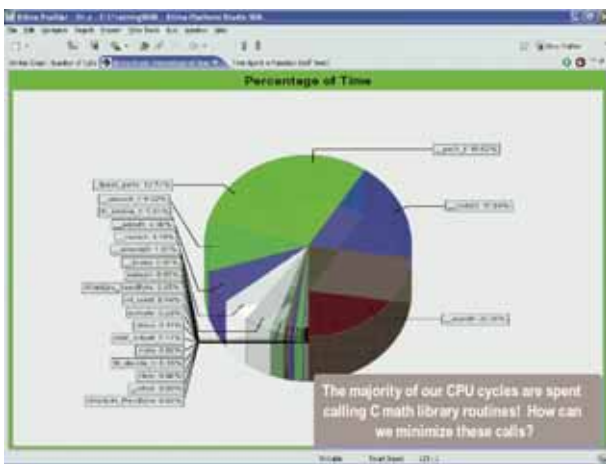


Figure 2 – Floating-point library calls occupy CPU time

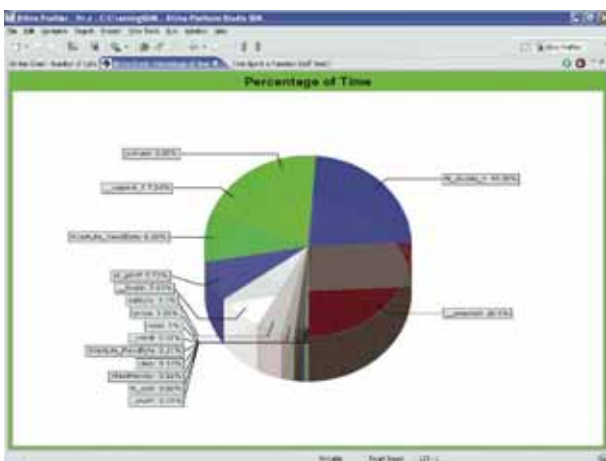


Figure 3 – Adding FPU improves performance

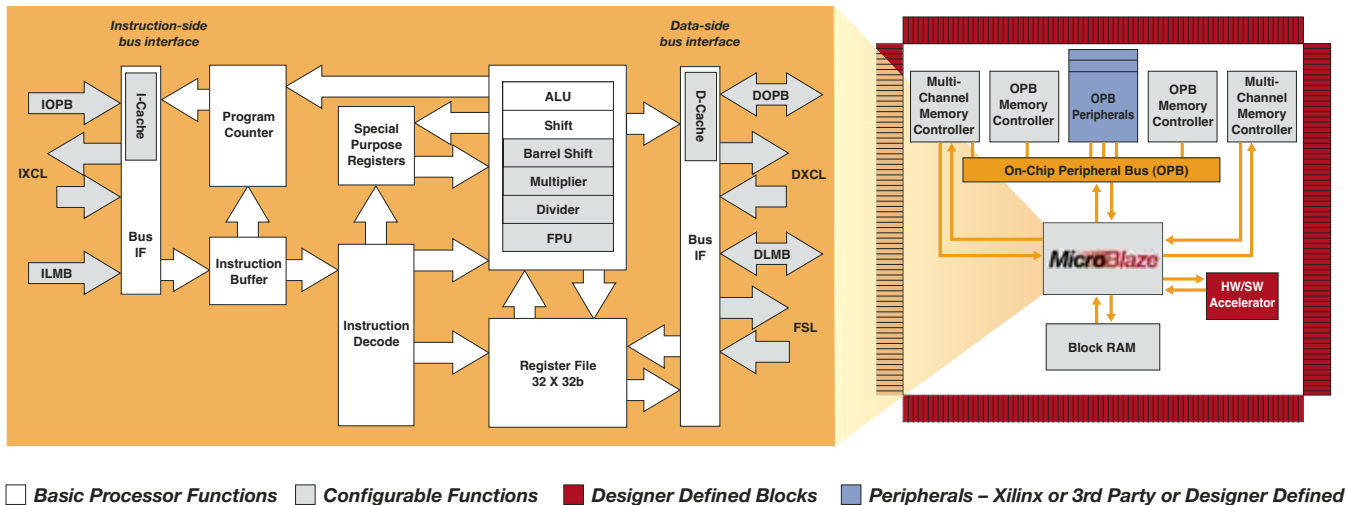


Figure 4 – MicroBlaze v4.00 architecture

FPU to the Rescue

To compare performance results, you will need to profile a floating-point FIR filter application on two different configurations of the MicroBlaze processor. The flat profile views in Figures 5 and 6 display the time spent on the main function call with and without an FPU, respectively. Adding the FPU to the design reduced the number of CPU cycles required by this application from 62 milliseconds to about 4.7 milliseconds – a 13x improvement. In addition, the FIR function call is now more than 50 times faster than in the previous run without an FPU. By simply enabling the MicroBlaze FPU option, you have substantially improved the performance of the MicroBlaze floating-point FIR design.

To further enhance system performance, consider implementing the entire FIR function in hardware as an assist engine connected to a fast simplex link (FSL) interface. The FSL is a simple yet powerful point-to-point interface that connects user-developed co-processors to the MicroBlaze processor pipeline. Like the FPU option,

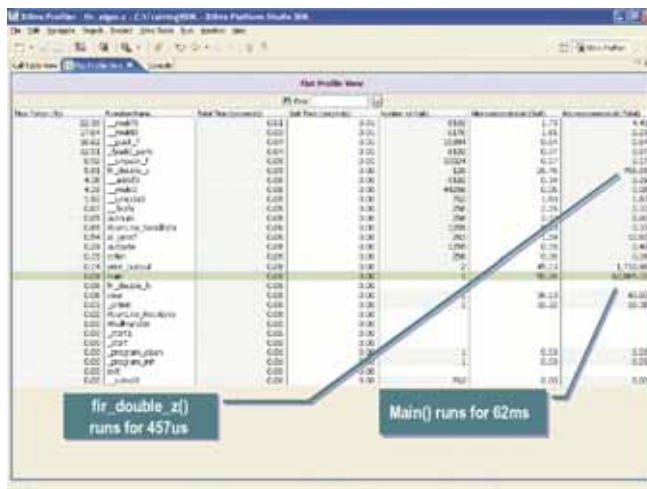


Figure 5 – Flat profile view for FIR filter design without FPU

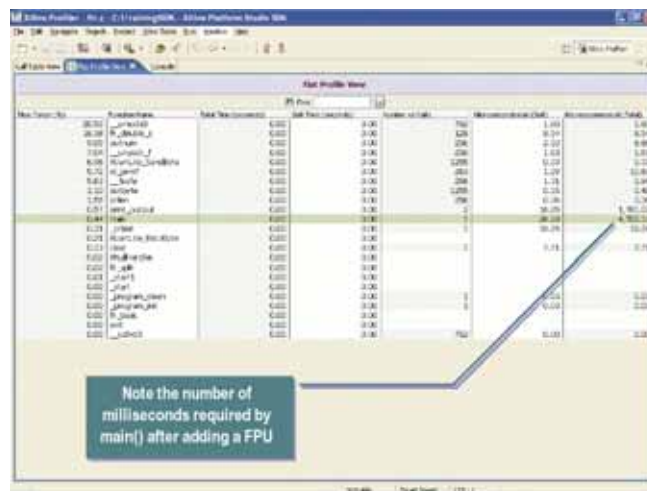


Figure 6 – Flat profile view for FIR filter design with FPU

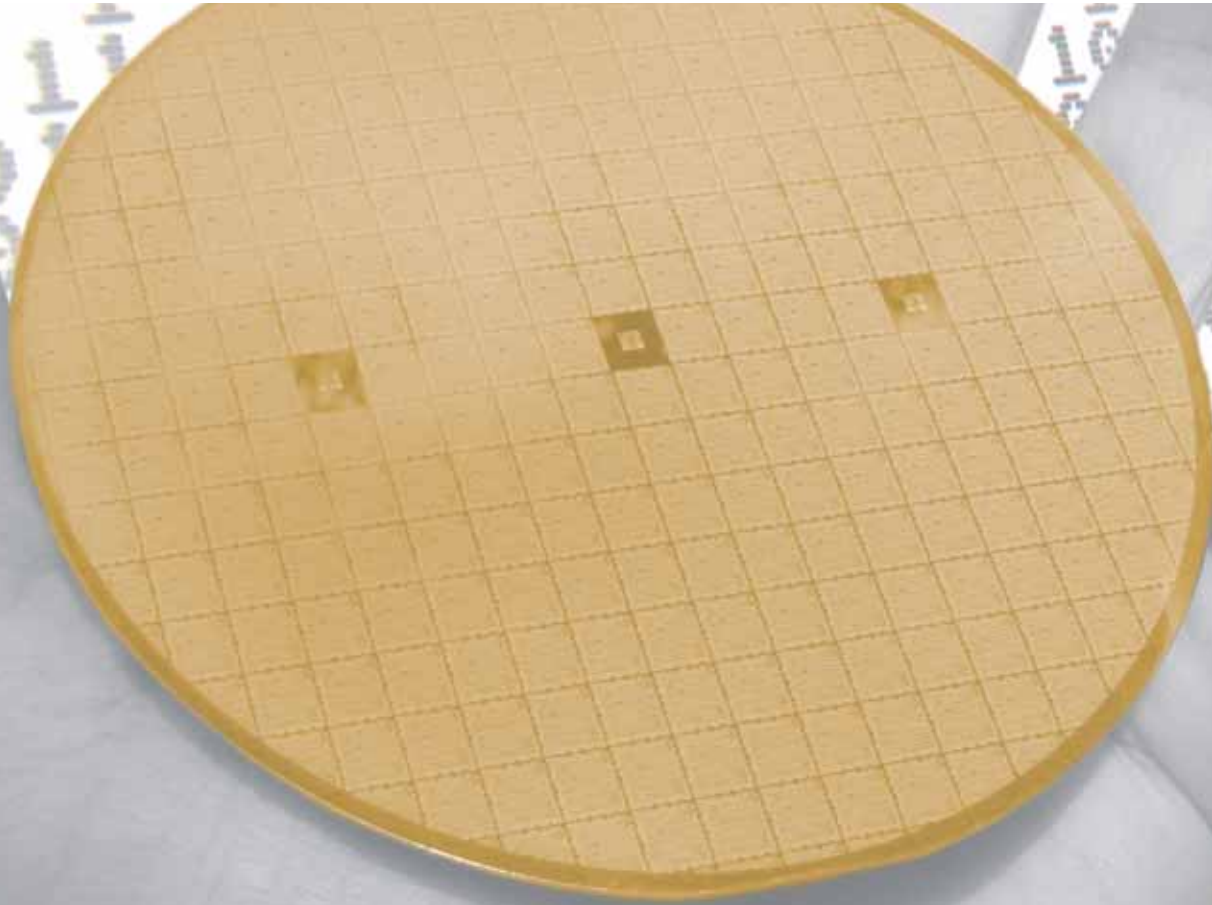
the FSL interface is a configurable feature within the MicroBlaze processor. This high level of configurability allows you to tailor your processor system to the exact needs of the target embedded application, which provides great flexibility but does not add to the cost if these features are not used.

Conclusion

This sample MicroBlaze FIR filter application has demonstrated how an algorithm that is a candidate for hardware acceleration can – with minimal work – be implemented on a mixed hardware/software platform for the purpose of performance evaluation. Utilizing the Xilinx Platform Studio embedded tool suite, you can generate cycle-accurate software models and profile the performance of software running on a fully custom virtual system. You can quickly tune the performance of your processor architecture and system architecture to achieve an optimal balance of performance versus hardware resources.

For more information on the MicroBlaze soft-processor core, visit www.xilinx.com/microblaze.

Grasp the full power of Xilinx processing...



Celoxica tools help you get the most extreme performance for your software algorithm. Augment your Xilinx processor with custom hardware co-processors using Celoxica's C-synthesis technology.

The DK Design Suite programs Xilinx logic directly from software descriptions, and provides software API interfaces to connect hardware logic to embedded processors. Celoxica offers comprehensive support for Xilinx devices and embedded processors including Virtex-4 FX, Virtex II Pro, Spartan 3 and MicroBlaze.

The combination of Xilinx devices and embedded processors with Celoxica technology produces performance gains of 50x-300x to deliver the algorithm acceleration you need.

For more information and design examples visit www.celoxica.com





PowerPC and MicroBlaze Development Kit Virtex-4 FX12 Edition Accelerate Your Embedded Development

Creating a new, real-time embedded system can be quite a challenge today, especially if you are designing your own custom hardware, software, and firmware. A completely integrated development environment of hardware, design tools, IP, and working reference designs can rapidly accelerate your embedded development.

Xilinx® provides powerful Platform FPGA devices with a variety of flexible embedded system capabilities to handle any real-time application. By supplying a spectrum of processing solutions including high-performance hard PowerPC™ cores, co-processor acceleration options, and flexible soft MicroBlaze™ processor cores, Xilinx can ensure that you have all the programmable elements to build exactly the system you require.

With pre-verified reference designs and a powerful evaluation board integrating a broad set of system options, you can kick-start the design process and focus on adding value to your designs. Intelligent “platform-aware” tools ease the design process and complete the Xilinx Embedded Development Kit, helping you get to market faster and satisfy your product requirements.



Build Faster and More Flexible Embedded Processing Systems

Programmable platforms and innovative tools allow you to craft an embedded design with the perfect combination of feature set, performance, area, and cost. Choose the most effective processor core for your application, customize your IP, optimize the performance, and validate your software on a development board before you even have your own custom hardware back from the shop.

Programmable Platform FPGAs are Versatile SOCs. Programmable, reprogrammable and field-upgradable platforms mean that your product gets to market quicker and has a longer life. System features immersed in the chip allow you to build a wide variety of applications with off-the-shelf devices. Embedded capabilities include hard and soft processor options, processing IP, on-board memory, and even DSP capability.

Complete Spectrum of Processing Options. The Xilinx Virtex™-4 FX12 and the ML403 evaluation platform support both the high-performance PowerPC hard and flexible MicroBlaze soft-processing cores. Choose the appropriate processing core for your application and feel confident knowing that the IP and tools are a unified environment.

Optimize Your Design. Broad features of the ML403 development board empower you to optimize performance via a high-bandwidth Auxiliary Processor Unit (APU) controller to execute custom instructions, integrated Ethernet MACs, multiple memory, audio, video, and other interfaces.

Intelligent Tools and Reference Designs Accelerate Development. Easy-to-use design wizards and automatic generators will reduce user errors and streamline the development process, even generating software drivers, sample test code, and BSPs (board support packages). Integrated hardware/software debuggers enable you to find and fix bugs faster and more efficiently.



The PowerPC and MicroBlaze Development Kit Virtex-4 FX12 Edition includes the following:

- Virtex-4 ML403 development board
- Platform Studio embedded tool suite
- ISE FPGA design software
- Pre-verified reference designs
- All documentation, JTAG probe, power supply, cables and flash device

ML403 Evaluation Platform (Development Board):

- **Xilinx devices** – XC4VFX12-FF668-10C, XC95144XL, XCCACE, XCF32P
- **Memory** – 64 MB DDR SDRAM, 1 MB ZBT SRAM, 512 MB compact flash, 8MB linear flash, and 4kb IIC EEPROM
- **Clocks** – 100 MHz oscillator and two clock sockets
- **Display** – 16 x 2 character LCD
- **Connectors and Interfaces** – Four SMA connectors, two PS/2 connectors, two audio, RS-232 serial port, three USB ports, PC4 JTAG, DB15 VGA, RJ-45 Ethernet, and GPIO

Development Tools and IP

(for both PowerPC and MicroBlaze core design):

- **Embedded Development Kit with Platform Studio embedded tool suite**
 - Graphical tools for developing/debugging embedded platforms
 - GNU compiler, debugger, and utilities
 - Platform Studio SDK – Software Development Kit, Eclipse-based IDE for powerful embedded software development and debug
 - MicroBlaze soft-processor license
 - IP library of processor peripheral cores and other evaluation cores
 - XMD – Xilinx microprocessor debug engine for run time target control
- **ISE FPGA design tool suite**
 - Timing-driven FPGA hardware implementation tools
 - Design entry, synthesis, and verification capabilities
 - Data2MEM – application for loading on-chip memory



ML403 Development Board

Embedded Design Services – XDS

Xilinx Design Services's (XDS) experience in system, logic, and embedded software design complements your own resources to optimize your budget, schedule, and performance requirements. Go to www.xilinx.com/xds for more information on how XDS can help you solve your schedule and design challenges.

Embedded Systems Training and Development Courses

Learn how to effectively develop, debug, and simulate an embedded system using the newest advancements in Platform Studio technology. A variety of courses are available. Go to www.support.xilinx.com/support/education-home.htm to learn more.

Take the Next Step

For more information on all Xilinx Embedded Processing Solutions, visit www.xilinx.com/processor.

Ordering Info:

Part number: DO-ML403-EDK-ISE



Corporate Headquarters

Xilinx, Inc.
2100 Logic Drive
San Jose, CA 95124
Tel: (408) 559-7778
Fax: (408) 559-7114
Web: www.xilinx.com

European Headquarters

Xilinx
Citywest Business Campus
Saggart,
Co. Dublin
Ireland
Tel: +353-1-464-0311
Fax: +353-1-464-0324
Web: www.xilinx.com

Japan

Xilinx, K.K.
Shinjuku Square Tower 18F
6-22-1 Nishi-Shinjuku
Shinjuku-ku, Tokyo
163-1118, Japan
Tel: 81-3-5321-7711
Fax: 81-3-5321-7765
Web: www.xilinx.co.jp

Asia Pacific

Xilinx Asia Pacific Pte. Ltd.
No. 3 Changi Business Park Vista, #04-01
Singapore 486051
Tel: (65) 6544-8999
Fax: (65) 6789-8886
RCB no: 20-0312557-M
Web: www.xilinx.com

Distributed By:

FORTUNE 2005
100 BEST COMPANIES TO WORK FOR



© 2005 Xilinx, Inc. All rights reserved. XILINX, the Xilinx logo, and other designated brands included herein are trademarks of Xilinx, Inc. All other trademarks are the property of their respective owners.



MicroBlaze Development Kit Spartan-3 SP305 Edition

The Low-Cost Embedded Development Solution

Creating a new embedded system can be quite challenging today, especially if you are looking for a low-cost solution with the flexibility to design your own custom hardware, software, and firmware.

The MicroBlaze™ Development Kit, Spartan™-3 SP305 Edition, powered by the 3S1500 device and supported by industry-standard peripherals, connectors, and interfaces, provides a low-cost, easy-to-use development platform for Spartan-3-based embedded designs. The Spartan-3 SP305 development board gives you instant access to cutting-edge 90 nm technology and the complete platform capabilities of the Spartan-3 FPGA family, including the configurable MicroBlaze 32-bit soft-processor core. Develop embedded applications via the industry-leading Xilinx® ISE™ and Embedded Development Kit (EDK) with Platform Studio embedded tool suite and proven IP cores. The MicroBlaze Development Kit, Spartan-3 SP305 Edition brings embedded designs to reality quicker, at low cost, and on schedule.



Build Lower Cost and Configurable Embedded Processing Systems

Programmable platforms and innovative tools allow you to customize your embedded design with the ideal combination of feature set, performance, area, and cost.

MicroBlaze Soft-Processor Solution

The Spartan-3 SP305 development board supports the MicroBlaze 32-bit soft-processor core. With the MicroBlaze soft processor, you have complete flexibility to select any combination of peripherals, memory, and interface features that you need to give you the best system performance at the lower cost on a single FPGA.

Optimize Your Embedded Design

Broad features of the Spartan-3 SP305 development board empower you to optimize performance via the MicroBlaze fast simplex link (FSL) to execute custom functions, integrated Ethernet MACs, multiple memory, audio, video, and other interfaces. FSL allows you to connect hardware co-processors to accelerate time-critical algorithms.

Intelligent Tools and Reference Designs Accelerate Development

EDK includes the Platform Studio tool suite, the MicroBlaze soft-processor core license, as well as all the documentation and soft peripheral IP that you need to start designing Spartan-3-based embedded processor systems today.

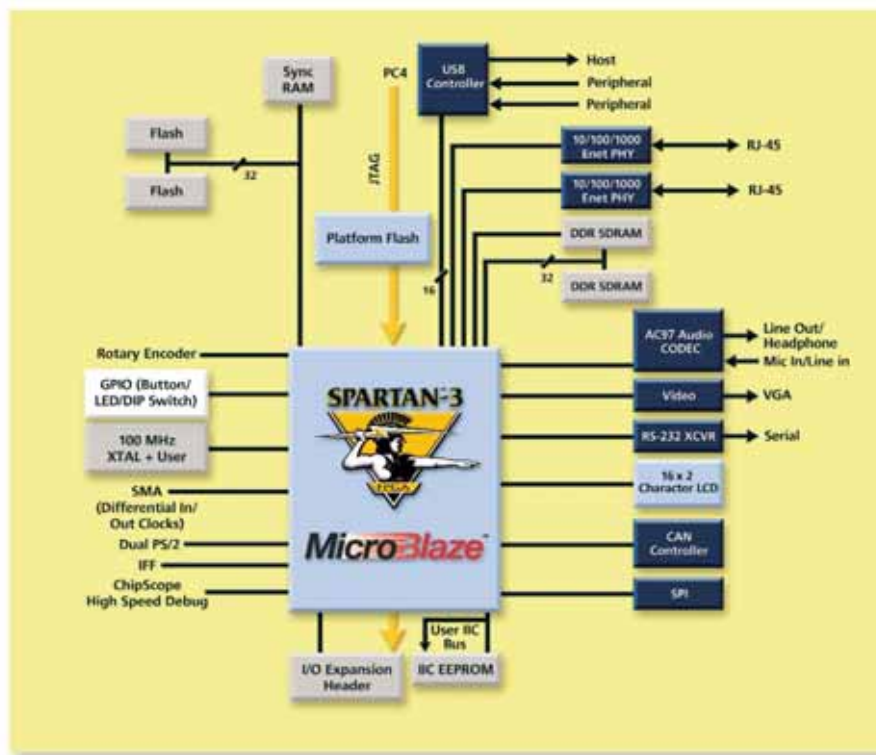
The MicroBlaze Development Kit Spartan-3 SP305 Edition includes the following:

- EDK with Platform Studio embedded tool suite
- ISE FPGA design software
- Pre-verified reference designs
- All documentation, JTAG probe, power supply, cables, and flash device
- Spartan-3 SP305 development board
 - Xilinx devices –XC3S1500-FG676-10C, XCF32P
 - Clocks –100 MHz oscillator, two clock sockets
 - Memory – 64 MB DDR SDRAM, 9 Mb ZBT SRAM, 8 MB linear flash and 4 Kb IIC EEPROM
 - Display – 16 x 2 character LCD

- Connectors and Interfaces
 - Four SMA connectors (differential clocks)
 - Two PS/2 connectors (keyboard/mouse)
 - Two audio (in/out, microphone, headphone)
 - Two UART RS-232 serial ports
 - SPI port, CAN port
 - Three USB ports (two peripherals/one host)
 - PC4 JTAG
 - DB15 VGA display
 - RJ-45 Ethernet port (connected to SMSC MAC/PHY device)
 - RJ-45 Ethernet port (connected to Intel PHY device)
 - Rotary encoder with push-button switch
 - Expansion headers (32 single-ended, 16 differential, and 16 general-purpose I/O)
 - General-purpose I/O (buttons/LEDs)

Take the Next Step

Order your MicroBlaze Development Kit, Spartan-3 SP305 Edition today and get a head start on your MicroBlaze in Spartan-3 embedded design. For more information, visit www.xilinx.com/sp305.



MicroBlaze™

Platform Studio™



Corporate Headquarters

Xilinx, Inc.
2100 Logic Drive
San Jose, CA 95124
Tel: (408) 559-7778
Fax: (408) 559-7114
Web: www.xilinx.com

European Headquarters

Xilinx
Citywest Business Campus
Saggart,
Co. Dublin
Ireland
Tel: +353-1-464-0311
Fax: +353-1-464-0324
Web: www.xilinx.com

Japan

Xilinx, K.K.
Shinjuku Square Tower 18F
6-22-1 Nishi-Shinjuku
Shinjuku-ku, Tokyo
163-1118, Japan
Tel: 81-3-5321-7711
Fax: 81-3-5321-7765
Web: www.xilinx.co.jp

Asia Pacific

Xilinx Asia Pacific Pte. Ltd.
No. 3 Changi Business Park Vista, #04-01
Singapore 486051
Tel: (65) 6544-8999
Fax: (65) 6789-8886
RCB no: 20-0312557-M
Web: www.xilinx.com

Distributed By:

FORTUNE 2005
100 BEST COMPANIES TO WORK FOR

XILINX®
The Programmable Logic Company™

© 2005 Xilinx, Inc. All rights reserved. XILINX, the Xilinx logo, and other designated brands included herein are trademarks of Xilinx, Inc. All other trademarks are the property of their respective owners.

More Integration, Easier Development

Well designed and feature-rich, Virtex-4 platforms enable you to expedite hardware and software development.

by Saeid Mousavi
Sr. Strategic Marketing Manager
Xilinx, Inc.
saeid.mousavi@xilinx.com

Copyright © 2005 Xilinx, Inc. All rights reserved.

Designs for embedded systems require a series of hardware and software components, including compilers, debuggers, operating systems, IP cores, companion chipsets, and prototyping platforms. The flexibility, reprogrammability, functionality, and performance of Xilinx® Virtex™-4 FPGAs, along with supporting hardware/software components, provide an excellent solution for today's challenging and complex embedded system designs.

As illustrated in Figure 1, embedded system designs with Xilinx FPGAs and supported processors (such as the PowerPC™ 405 hard-core and MicroBlaze™ soft-core processor) comprise both standard FPGA hardware and processor software developments in parallel. This parallel integration and flow is enabled by the Xilinx Embedded Development Kit (EDK) and ISE™ Foundation™ software.

The final design requires a hardware platform to provide a vehicle for evaluation and validation. Xilinx offers a series of Spartan™, Virtex-II Pro-, and Virtex-4-based development, evaluation, and prototyping platforms that you can use for such purposes.

These embedded system development and prototyping platforms are carefully designed to provide the right set of FPGAs, interfaces, connectors, and support packages. For Virtex-4-based embedded system designs utilizing embedded hard-coded PowerPC or MicroBlaze soft processors, we recommend the ML401, ML402, ML403, ML405, and ML410 platforms. These platforms are supported by a wide range of reference designs, IP cores (evaluation), and Xilinx and third-party tools.

ML403

Populated with a Virtex-4 XC4VFX12 device, the ML403 is a low-cost, feature-rich, and easy-to-use embedded system development platform (Figure 2). The XC4VFX12 device – with one embedded PowerPC 405 processor and more than

12,000 FPGA slices, along with access to a wide set of connectors and interfaces – makes the ML403 a good candidate for embedded system developments. The FPGA gates and PowerPC processor can be accessed through on-board USB (host and peripheral), 10/100/1000 Ethernet, audio in/out, CompactFlash card interface, and 64 general-purpose I/O (GPIO) pins.

FPGA configuration is supported by Parallel Cable IV cable (JTAG), System ACE™ controller (JTAG), platform flash memory, and linear flash plus CPLD. System ACE technology supports multiple bitstreams and provides an easy, scalable, and reusable configuration.

The ML403 board contains headers (0.1") for easy expansion or adaptation of the board for other applications through customized modules. The expansion connectors contain connections to single-ended (32) and differential FPGA I/Os (16 pairs), ground, 2.5V/3.3V/5V power, JTAG chain, and the IIC bus.

The availability of 64 MB of on-board DDR SDRAM (32-bit interface at 266 MHz) and external memory through a

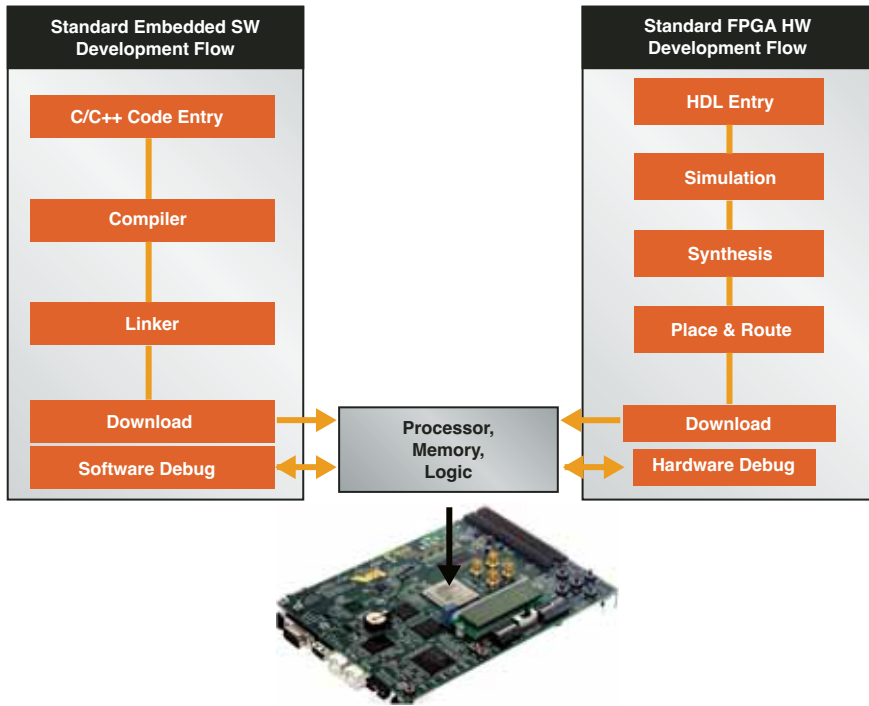


Figure 1 – FPGA hardware and processor software design flow



Figure 2 – Xilinx HW-V4-ML403 platform



Figure 3 – Xilinx HW-V4-ML410 platform

Cardbus reader provides ample storage for operating system developments. The ML403 is currently supported by Linux and Vxworks operating systems.

Other important features of the ML403 board include 10/100/1000 tri-mode Ethernet, USB host and peripheral ports, stereo AC97 audio CODEC, RS-232 serial port, VGA output, PS/2 mouse and keyboard, 9 Mb of ZBT synchronous SRAM, and 8 MB of flash memory. Additional product information is available at www.xilinx.com/ml403.

ML401, ML402, and ML405

The ML401 and ML402 platforms share the same features as the ML403, but are populated with Virtex-4 XC4VLX25 and XC4VSX35 devices, respectively. The ML405 is populated with a Virtex-4 XC4VFX20 device and provides additional support for RocketIO™ multi-gigabit transceivers (MGTs) through two SATA, one SFP, and four SMA connectors.

Xilinx board database (XBD) files for all boards in the ML4 family are available as part of the EDK library.

ML410

Populated with a Xilinx Virtex-4 XC4VFX60 device and supported by a wide set of industry-standard connectors, interfaces, and companion chipsets, the ML410 platform (with ATX form factor) is an ideal platform for embedded system developments (Figure 3).

With two embedded PowerPC 405 processors, the ML410 is an excellent platform for parallel and distributed processing applications and development. In addition to the processors, the ML410 provides access to RocketIO MGTs through PCI Express slots (two); Serial ATA (two); and Z-dok-based personality modules. These PM101 and PM102 personality modules (the latter shown in Figure 4), are mainly designed to provide access to eight channels of RocketIO MGTs through SFP, X-PAK, and MSA300 connectors. Personality modules also provide access to the LVDS and single-ended I/Os of the FX60 device.

The ALI M1535D+ south bridge companion chipset provides access to many features supported by standard PCs. These basic PC features are accessible over the PCI bus. The ALI chipset supports one parallel port, two USB ports, two IDE connectors, GPIO, SMBus interface, AC97 audio CODEC, and PS/2 keyboard and mouse.

To host PCI-based modules, the ML410 provides access to two PCI Express downstream slots. The ML410 also provides access to two 33 MHz/32-bit PCI buses, a primary 3.3V PCI bus and secondary 5.0V PCI bus (a total of four slots). The FPGA is directly connected to the PCI Express and primary 3.3V PCI bus, while the 5.0V PCI



Figure 4 – PM102 personality module

bus is connected to the primary PCI bus through a PCI-to-PCI bridge.

For high-speed communication, the ML410 supports two 10/100/1000 Base-T PHY with RJ45 connectors for two independent systems (1x connected through RGMII/MII and 1x connected through SGMII).

To support high-speed storage applications, the ML410 platform provides two Serial ATA host connectors targeting 1.5 Gbps operation.

In addition to 64 MB of DDR memory, the ML410 provides support for as much as 1 GB of DDR 2 memory through a 64-bit

DIMM socket (supporting buffered and unbuffered DIMM). Together with the System ACE CF controller providing hard-disk access to additional memory on CompactFlash cards, the ML410 offers ample storage area for operating system and software development. The ML410 is currently supported by Linux, Vxworks, and QNX operating systems.

FPGA configuration is supported through Parallel Cable IV cable (JTAG), System ACE controller (JTAG), platform flash memory, and linear flash plus CPLD.

Additional product information is available at www.xilinx.com/ml410.

Conclusion

As summarized in Table 1, embedded system developments with Virtex-4 devices are supported by a wide range of Xilinx-designed platforms. These platforms enable you to significantly expedite hardware and software development.

In conjunction with these platforms, the availability of different reference designs, OPB- and PLB-based IP cores, EDK, operating systems, compilers, debuggers, and technical support make embedded system design an easy and pleasant experience. 🌈

	HW-V4-ML401	HW-V4-ML402	HW-V4-ML403	HW-V4-ML405	HW-V4-ML410
Board Type	Eval/Proto/Demo	Eval/Proto/Demo	Eval/Proto/Demo	Eval/Proto/Demo	Eval/Proto/Demo
Best Usage	FPGA /MicroBlaze Development	DSP Evaluation & Development	PPC Evaluation & Development	FPGA/RocketIO/PPC Eval/Development	Embedded System Dev.
Xilinx Device	LX25-FF668	SX35-FF668	FX12-FF668	FX20-FF672	FX60-FF1152
Price (Resale)	\$495	\$595	\$495	\$696	\$2,995
Memory	64 MB DDR/8 MB Flash/1 MB ZBT	64 MB DDR/8 MB Flash/1 MB ZBT	64 MB DDR/8 MB Flash/1 MB ZBT	128 MB DDR/8 MB Flash/1 MB ZBT	DDR2 256 MB Required DIMM / DDR1 64 MB
USB	2 Peripherals/1 Host	2 Peripherals/1 Host	2 Peripherals/1 Host	2 Peripherals/1 Host	YES (2 Peripheral Ports)
10/100/1000 Ethernet	1 Port	1 Port	1 Port	1 Port: 10/100/1000 & SGMII	2 Ports: 10/100/1000 & SGMII
HSSDC2	NO	NO	NO	NO	NO
Serial ATA	NO	NO	NO	YES (2 Ports Host only)	YES (2 Ports Host only)
SFP	NO	NO	NO	1 Port	Through Personality Module
PS/2 Mouse	YES	YES	YES	YES	YES
PS/2 Keyboard	YES	YES	YES	YES	YES
System ACE	YES	YES	YES	YES	YES
PC4 Interface	YES	YES	YES	YES	YES
JTAG	YES	YES	YES	YES	YES
Serial Port	YES	YES	YES	YES	YES (2)
Parallel Port	NO	NO	NO	NO	YES
Clock	-LVDS pair via SMAs (1) - Single-ended via socketed oscillators (2)	-LVDS pair via SMAs (1) - Single-ended via socketed oscillators (2)	-LVDS pair via SMAs (1) - Single-ended via socketed oscillators (2)	-LVDS pair via SMAs (1) - Single-ended via socketed oscillators (2) -For MGT: 125 MHz LVDS, 150 MHz LVDS, programmable LVDS clock generator	On board: - LVDS pair via SMAs (2) - Single-ended via SMA (1) - Single-ended via socketed oscillators (2) Personality module: LVDS pairs (1)
PCI Slot	NO	NO	NO	NO	3.3V(2), 5.0V(2), PCI Express(2)
GPIO	YES	YES	YES	YES	YES
Microphone	YES	YES	YES	YES	YES
Headphone	YES	YES	YES	YES	YES
Audio	YES (line in/line out)	YES (line in/line out)	YES (line in/line out)	YES (line in/line out)	YES (line in/line out)
Video (VGA)	YES	YES	YES	YES	YES
Display	LCD	LCD	LCD	LCD	LCD
Bundles With	Board + power supply + CompactFlash card + documentation	Board + power supply + CompactFlash card + documentation	Board + power supply + CompactFlash card + documentation	Board + power supply + CompactFlash card + documentation	Board + power supply + CompactFlash card + CD (documentation)

Table 1 – Xilinx embedded system development platforms

C to FPGA



Develop and debug
FPGA algorithms
using standard
C programming
methods and tools.

Finally, a practical C-to-hardware tool designed with software programmers in mind. The Impulse C software-to-hardware compiler translates untimed, hardware-independent C-language algorithms to low-level FPGA hardware, while optimizing the generated logic for increased parallelism.

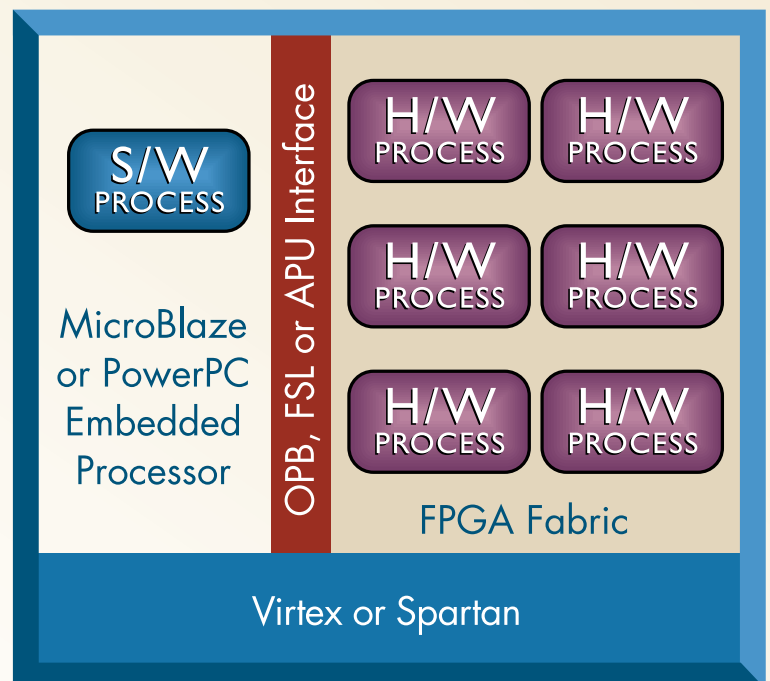
Impulse C takes FPGA embedded processing to new levels of performance and design efficiency. Enhanced instruction scheduling and loop pipelining can produce **up to 300x acceleration** of computationally intensive algorithms. The Impulse C tools add debugging visualizations for highly-parallel, multi-process applications to help you identify processing bottlenecks and partition your applications into hardware and software processing resources. For behavioral simulation, the Impulse C libraries are compatible with Microsoft® Visual Studio™, GCC, Eclipse and other popular C development environments. The generated hardware descriptions are fully compatible with Xilinx® ISE™ and Platform Studio™, and with other popular HDL simulation and synthesis tools.

The Impulse C compiler offers enhanced support for the Xilinx MicroBlaze™ and PowerPC™, by providing abstract software/hardware communication methods including streams, signals and shared memories. These C-compatible communication methods allow you to make use of available FPGA resources for hardware coprocessing without writing low-level hardware descriptions. Extensive examples and tutorials make your first projects quick and easy.

Prices start at \$2,695.

30 day evaluations are
free to qualified engineers.

www.impulseC.com



impulse
accelerated technologies



Embedded Systems Development

EMBD21000-7-ILT (v1.0)

Course Specification

Course Description

Embedded Systems Development introduces experienced FPGA designers to developing embedded systems using hard (embedded IBM PowerPC™) or soft (Xilinx® MicroBlaze™) processor cores, and soft peripheral cores, within the Embedded Development Kit (EDK) design environment. The course includes hands-on labs to provide personal experience with the development, debugging, and simulation of the embedded system.

Level – Intermediate

Course Duration – 2 days

Price – \$1000 USD or 10 Training Credits

Course Part Number – EMBD21000-7.1-ILT

Who Should Attend? – FPGA design engineers, system architects, and system engineers who are interested in Xilinx embedded systems development flow

Prerequisites

- FPGA design experience
- Completion of the *Fundamentals of FPGA Design* course or equivalent knowledge of Xilinx ISE™ implementation tools
- Basic understanding of C programming
- Basic microprocessor experience, understanding of PowerPC and MicroBlaze systems

Software Tools

- ISE 7.1 SP1
- ModelSim PE 6.0
- EDK 7.1

After completing this comprehensive training, you will have the necessary skills to:

- Effectively develop, debug, and simulate an embedded system
- Identify tools used in the EDK
- Understand the hardware and software flows defined in the EDK
- Identify IP included in the EDK and where to get additional information
- Understand the hardware and software simulation environments
- Integrate custom IP into the EDK

Course Outline

Day 1

- EDK Overview
- **Lab 1:** Simple Hardware Design
- Hardware Design
- Hardware Design Using EDK
- **Lab 2:** Adding IP to a Hardware Design
- Adding Your Own IP to the OPB Bus
- **Lab 3:** Adding Custom IP to an Embedded System

Day 2

- Software Development
- Address Management
- **Lab 4:** Writing Basic Software Applications
- Debugging
- **Lab 5:** Advanced Software Writing
- **Lab 6:** Using the Software Development Kit (SDK)

- System Simulation
- **Lab 7:** Performing System Simulation

Lab Descriptions

- **Lab 1:** Simple Hardware Design – Create an XPS project by using the Base System Builder to develop a basic hardware system for a target board.
- **Lab 2:** Adding IP to a Hardware Design – Learn to add IP, such as bridges, OPB peripherals, OPB buses, and others to the basic hardware design.
- **Lab 3:** Adding Custom IP to an Embedded System – Explore adding a custom IP to your design by using the Creating/Importing Peripheral Wizard.
- **Lab 4:** Writing Basic Software Applications – Write a basic C application that utilizes the UART and GPIO.
- **Lab 5:** Advanced Software Writing – Use the OPB Timer and create an interrupt service routine.
- **Lab 6:** Using the SDK – Writing applications using the Eclipse-based SDK.
- **Lab 7:** Performing System Simulation – Generate simulation scripts by using XPS and perform behavioral simulation.

Register Today

Xilinx delivers public and private courses in locations throughout the world. Please contact Xilinx Education Services for more information, to view schedules, or to register online.

Visit www.xilinx.com/education, and click on the region where you want to attend a course.

North America, send your inquiries to registrar@xilinx.com, or contact the registrar at 877-XLX-CLAS (877-959-2527). To register online, search by Keyword "Embedded" in the Training Catalog at <https://xilinx.onsaba.net/xilinx>.

Europe, send your inquiries to eurotraining@xilinx.com, call +44-870-7350-548, or send a fax to +44-870-7350-620.

Asia Pacific, contact our training providers at www.xilinx.com/support/training/asia-learning-catalog.htm, send your inquiries to education_ap@xilinx.com, or call +852-2424-5200.

Japan, see the Japanese training schedule at www.xilinx.co.jp/support/training/japan-learning-catalog.htm, send your inquiries to education_kk@xilinx.com, or call +81-3-5321-7750.

You must have your tuition payment information available when you enroll. We accept credit cards (Visa, MasterCard, or American Express) as well as purchase orders and training credits.



Course Description

Advanced Features and Techniques of Embedded Systems Development provides embedded systems developers the necessary skills to develop complex embedded systems and improve their designs by using the tools available in Xilinx® EDK. This course also helps developers understand and utilize advanced components of embedded systems designs for architecting a complex system.

This course builds on the skills learned in the *Embedded Systems Development* course. Labs provide hands-on experience with the development, verification, debugging, and simulation of the embedded system. All labs use a virtual hardware board to which designs are downloaded and verified.

Level – Advanced

Course Duration – 2 days

Price – \$1200 USD or 12 Training Credits

Course Part Number – EMBD33000-6.3-ILT

Who Should Attend? – FPGA design engineers, system architects, and system engineers who are interested in Xilinx embedded systems development flow

Prerequisites

- Experience in C programming
- *Embedded Systems Development* course or experience with embedded systems design
- Some HDL modeling experience
- Basic microprocessor experience and understanding of PowerPC™ and MicroBlaze™ systems

Software Tools

- ISE™ 6.3
- Mentor Graphics ModelSim
- EDK 6.3

After completing this comprehensive training, you will have the necessary skills to:

- Understand and utilize advanced features and techniques of Xilinx Platform Studio for embedded systems design
- Build a complete embedded system
- Architect a system that incorporates performance improvement and booting large applications from external memory
- Employ various debugging techniques

Course Outline

Day 1

- Embedded Systems Development Review
- **Lab 1:** Building a Complete Embedded System
- External Memory Controllers and File Systems
- **Lab 2:** External Memory Controllers and File Systems
- Interrupts
- Debugging Using ChipScope™ Pro
- **Lab 3:** Debugging Using ChipScope Pro
- OCM Bus

Day 2

- Performance Tuning
- **Lab 4:** Performance Tuning
- Board Support Packages
- BFM Simulation
- **Lab 5:** BFM Simulation
- Boot Loader
- **Lab 6:** Boot Loading from Flash Memory

Lab Descriptions

- **Lab 1:** Comprehensive System Development – Develop hardware that will incorporate IP cores to interface to push buttons, switches, LEDs, an LCD display, and serial communication. Develop an application that will interact with switches, push buttons, an LCD display, and serial communication. Generate a bitstream and download it onto a virtual hardware board.
- **Lab 2:** External Memory Controllers and File Systems – Design a system that will include an OPB (on-chip peripheral bus) EMC/SDRAM IP core. Develop an application that will perform file-related tasks on external memory.
- **Lab 3:** Debugging Using ChipScope Pro – Perform simultaneous hardware and software debugging on stack-related errors with the ChipScope Pro tool, GDB, and XMD.
- **Lab 4:** Performance Tuning – Profile a simple piece of code running on a processor and go through iterative steps of refinement to improve the performance by using cacheing and porting a repetitive function to hardware.
- **Lab 5:** BFM Simulation – Create an EDK system that includes IBM CoreConnect bus architecture bus functional models (BFM) and bus functional language constructs for an OPB IP. Simulate the OPB bus-based design to verify IP functionality.
- **Lab 6:** Boot Loading – Develop an application that performs desired tasks. Due to application size and resource limitations, store it in flash, load it through a boot loader program, and execute from external memory.

Register Today

Xilinx delivers public and private courses in locations throughout the world. Please contact Xilinx Education Services for more information, to view schedules, or to register online.

Visit www.xilinx.com/education and click on the region where you want to attend a course.

North America, send your inquiries to registrar@xilinx.com, or contact the registrar at 877-XLX-CLAS (877-959-2527). To register online, search by Keyword "Embedded" in the Training Catalog at <https://xilinx.onsaba.net/xilinx>.

Europe, send your inquiries to eurotraining@xilinx.com, call +44-870-7350-548, or send a fax to +44-870-7350-620.

Asia Pacific, contact our training providers at www.xilinx.com/support/training/asia-learning-catalog.htm, send your inquiries to education_ap@xilinx.com, or call +852-2424-5200.

Japan, see the Japanese training schedule at www.xilinx.co.jp/support/training/japan-learning-catalog.htm, send your inquiries to education_kk@xilinx.com, or call +81-3-5321-7750.

You must have your tuition payment information available when you enroll. We accept credit cards (Visa, MasterCard, or American Express) as well as purchase orders and training credits.

Develop Your Processing Skills

with Xilinx Education Services.



www.xilinx.com/education

Take advantage of two specially designed embedded processing courses by Xilinx, featuring in-depth training, hands-on labs and much more. Use your Xilinx Training Credits or simply pay as you go.

Course I — Embedded Systems Development (2 Days)

For experienced FPGA designers, an introduction to developing embedded systems using hard (embedded IBM PowerPC™) or soft (MicroBlaze™) processor cores within the Embedded Development Kit (EDK) design environment.

Course II — Advanced Features and Techniques of Embedded Systems Development (2 Days)

Embedded systems developers will develop the necessary skills to architect complex embedded systems and improve every design by using the tools available in the EDK. They will also learn the advanced features of Xilinx Platform Studio™ to optimize system performance.

Visit www.xilinx.com/education for more information and registration. Or call us at 1-877-XLX-CLASS.





Enabling a Connected World

MontaVista® Linux® – the most widely deployed Linux platform

From the network to the device, leading global manufacturers rely on MontaVista Linux to help connect the world. These manufacturers depend on MontaVista Linux to build advanced communications products required for today's highly competitive mobile communications networks, from the infrastructure out to the handset.

MontaVista Software is the recognized leader for the development of commercial grade Linux distributions that deliver leading edge technology to today's fastest growing and most demanding markets. With deep expertise in real-time systems technology and many years of experience, MontaVista Software leads the effort to make Linux the preferred platform for the new generation of intelligent devices. Companies that have chosen MontaVista have reaped the rewards of success based on the inherent benefits of MontaVista Linux including:

Faster Time to Market | Lower Development Costs | Open Source Dynamics | Advanced Technologies | Choice, Value and Flexibility

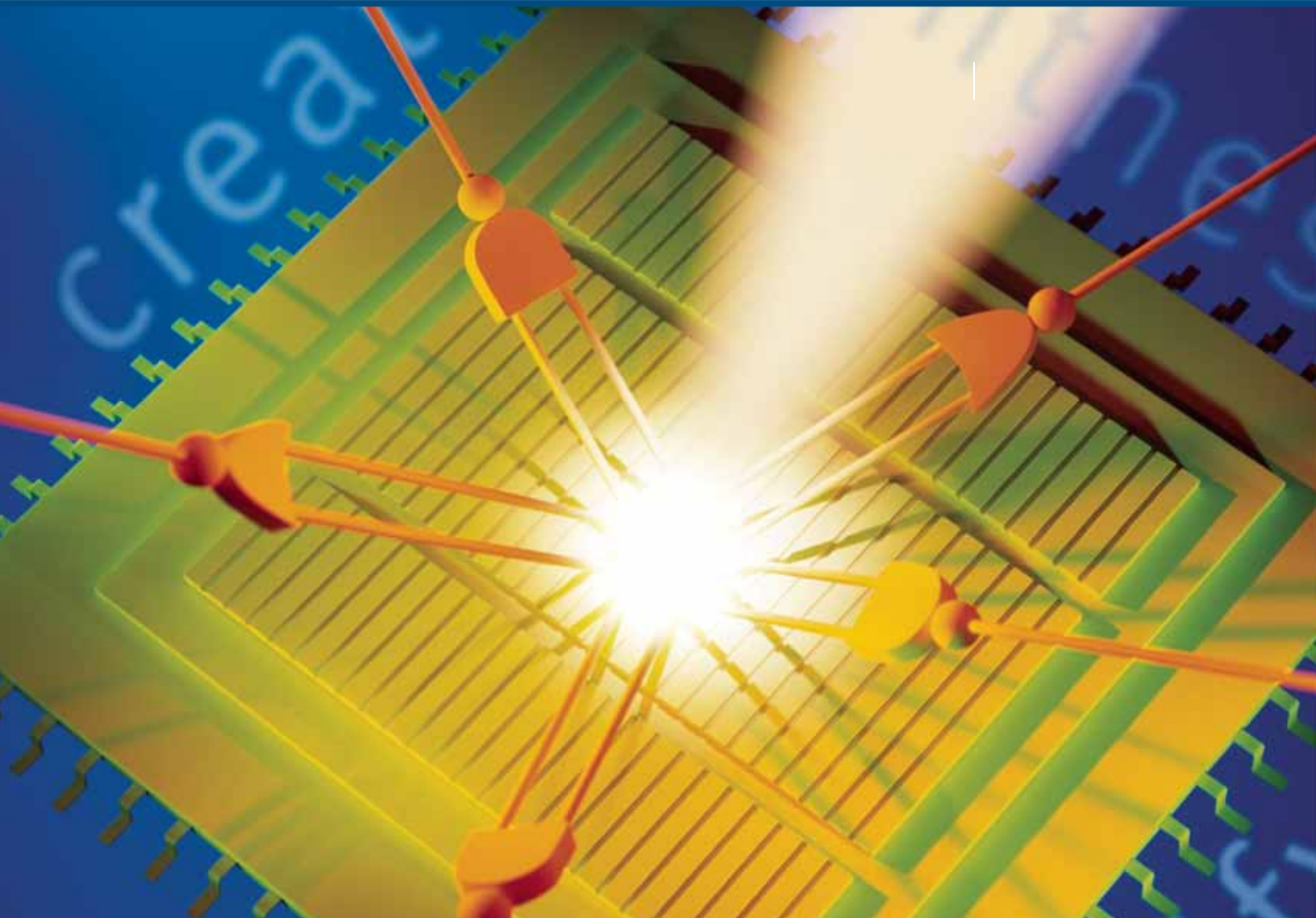
For more information, please visit our website at www.mvista.com

Copyright © 2005 MontaVista Software Inc. Linux is a registered trademark of Linus Torvalds. MontaVista is a registered trademark of MontaVista Software Inc. All other names mentioned are trademarks, registered trademarks or service marks of their respective owners.



Why are we the leader in FPGA design and verification?

Because Mentor helps you get your PowerPC®/MicroBlaze™ design right the first time.



DESIGN FOR MANUFACTURING + INTEGRATED SYSTEM DESIGN + ELECTRONIC SYSTEM LEVEL DESIGN + FUNCTIONAL VERIFICATION

FPGA DESIGN | Getting it right the first time. There's nothing better. Especially when you have huge devices with logic, memory, processors, and complex IP - all wrapped within another tight schedule. Mentor Graphics is the only EDA company that offers embedded software plus hardware design solutions, from design inception and synthesis to simulation and co-verification, for your embedded platform FPGA. For more information on the Mentor PowerPC/MicroBlaze solution, or other Mentor FPGA design solutions, visit mentor.com or call 800.547.3000 today.

**Mentor
Graphics®**
THE EDA TECHNOLOGY LEADER

Board Level Solutions DSP plus FPGA!

Four Screamers

...four screaming fast
C6416 DSPs, that is



Radio frequency signal processing solution

- ▶ Four, 720 MHz TMS320C6416 DSPs each with:
 - 8MB SDRAM
 - Dedicated 200 MB/s links between DSPs and FPGAs
- ▶ Two, 4M Gate User-Programmable FPGAs each with:
 - 18MB private DDR SDRAM
 - 128 MB private DDR SDRAM
- ▶ 64-bit/66MHz CompactPCI
 - 512MB Global DDR SDRAM
 - Two PMC Sites with JN4 to FPGA
 - External Data Port, up to 120Gb/s
 - StarFabric PICMG2.17 Compliant Port



Quadra

**Innovative
Integration**
... real time solutions!

www.innovative-dsp.com • 805.520.3300 phone

Hungry like the wolf...
for an ultra fast digital waveform
capture and playback solution?

Get your Lobo Data Sheets &
On-line Pricing Now!
www.innovative-dsp.com/lobo

Lobo's Features

- ▶ 225MHz/1350MFLOPS TMS320C6713 DSP
- ▶ 3 or 6 MGATE Virtex-II FPGA
- ▶ 32MB SDRAM, Up to 2GB DDR SDRAM for FPGA
- ▶ External I/O Interface for Data Card to FPGA DIO
- ▶ Separate Receive/Transmit Channels



**Innovative
Integration**
... real time solutions!

www.innovative-dsp.com • 805.520.3300 phone

lobo

powerful
connections



Network with the SBC6713e, our highest-performance stand-alone
DSP board with on-board 10/100 Ethernet

Features

- ▶ 300MHz, 32-bit, floating-point TMS320C6713 DSP
Powerful C/C++ libraries, Windows debugger
- ▶ 10/100 Ethernet via DM642 coprocessor for real-time network I/O
- ▶ Two OMNIBUS I/O expansion sites
Wide selection of analog input/output
Up to 24-bit resolution, up to 64 MHz sample rate
- ▶ Capable of 100% stand-alone operation
3U Size 100mm x 160mm
4 Mb Flash ROM
- ▶ Fantastic, on-board peripherals
RS232, 32-bit digital I/O, watchdog
600K gate Spartan-III for user-logic (optional)

Applications

- ▶ Embedded Servo Control
- ▶ Remote Data Acquisition
- ▶ Industrial Test & Measurement
- ▶ OEM Instrumentation

Download
Data Sheets
NOW!

**Innovative
Integration**
... real time solutions!

805.520.3300 phone
www.innovative-dsp.com

Onward to Glory

Complete Software
Radio Solution on a
single 6U card

Features

- ▶ 6 Million gate Virtex II FPGA
- ▶ Scalable Software Defined Radio
- ▶ Embedded Processing via MatLab
- ▶ TMS320C6416 DSP
- ▶ 105 MHz, 14 bit 2 Ch. Analog I/O
- ▶ 32 MB RAM
- ▶ 32/64-bit cPCI bus
- ▶ PMC expansion site
- ▶ STAR Fabric interface

Get your data sheets now!

www.innovative-dsp.com/quixote

sales@innovative-dsp.com
805.520.3300 phone • 805.579.1730 fax



Quixote

**Innovative
Integration**
... real time solutions!

Free Instant
On-Line Pricing!

805.520.3300 phone
www.innovative-dsp.com

**Innovative
Integration**
... real time solutions!

Fast and Flexible

Embedded Processing



Xilinx and Avnet Electronics Marketing are pleased to bring you hands-on, FREE in-booth workshops at ESC Boston, so you can gain valuable embedded design experience first hand. Only Xilinx Platform FPGAs offer programmable flexibility with the optimal choice of integrated hard and/or soft processors. It's a fast and flexible solution for your processing needs.

Hands-On Workshops

- Explore the latest Virtex-4 FX PowerPC system, including the power of the Auxiliary Processor Unit (APU) Controller, and the award-winning capabilities of Xilinx Platform Studio (9/13 – Noon, 2:30pm & 5pm, 9/14 – 11:15am & 1:45pm)
- Configure the latest MicroBlaze soft processor system with an integrated Floating Point Unit (FPU) in minutes (9/13 – 1:15pm & 3:45pm, 9/14 – 10am, 12:30pm & 3pm)

Live Demonstrations

- Evaluate the new development kits for PowerPC and MicroBlaze
- Realize the possibilities of single and dual-core PowerPC systems with the breakthrough Virtex-4 FX FPGA
- Test drive the latest configurable MicroBlaze 32-bit processor with low-cost Spartan-3 FPGA

Join Xilinx and Avnet at Embedded Systems Conference Boston – Booth #501

IN-DEPTH PRODUCT DEMONSTRATIONS | FREE HANDS-ON WORKSHOPS IN THE BOOTH | RECEPTION 5PM 9/13 TUESDAY

