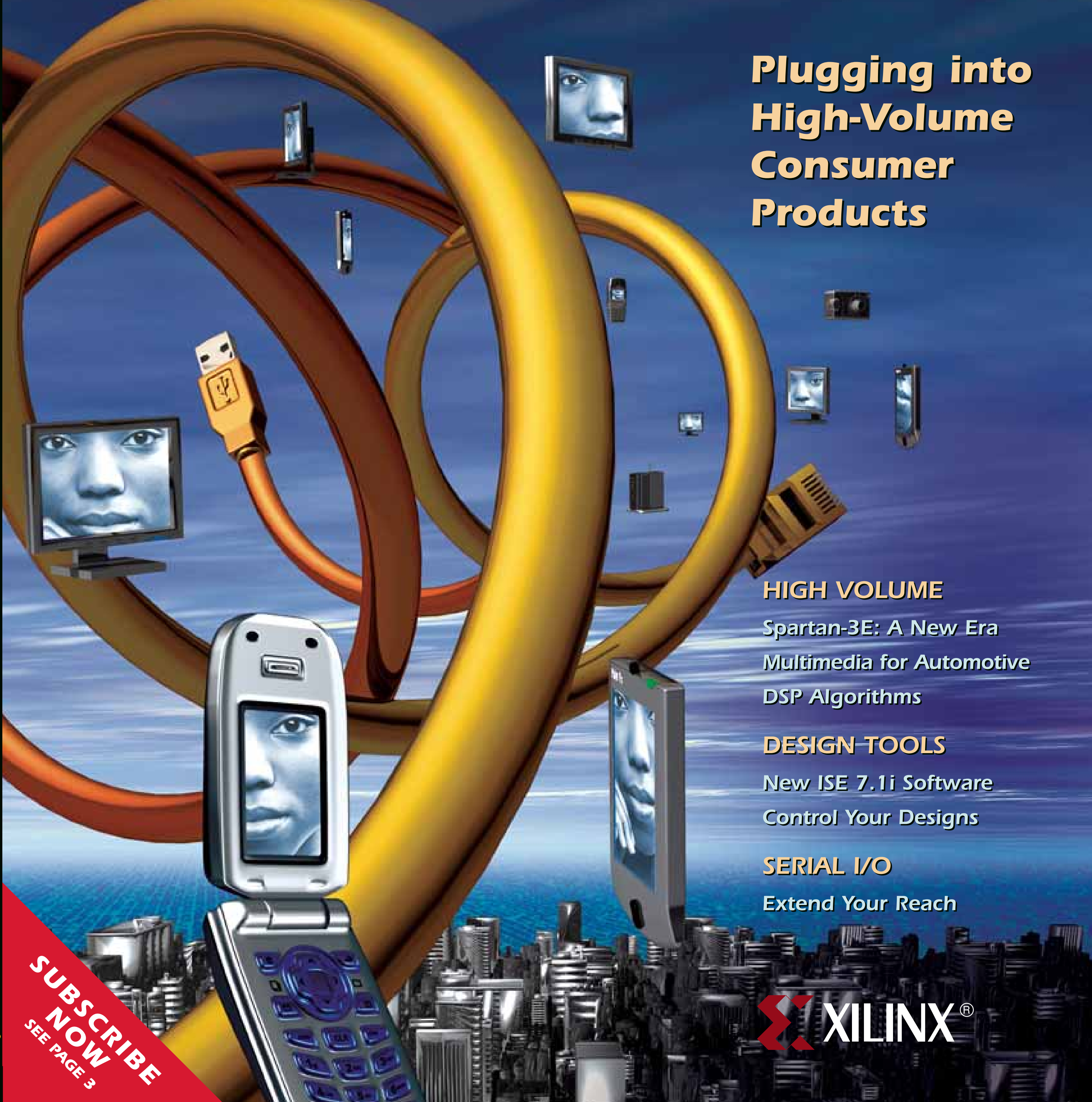
ISSUE 53, SECOND QUARTER 2005 XCELL JOURNAL XILINX, INC.

Xcell journal

Issue 53
Second Quarter 2005

THE AUTHORITATIVE JOURNAL FOR PROGRAMMABLE LOGIC USERS



Plugging into High-Volume Consumer Products

HIGH VOLUME

Spartan-3E: A New Era

Multimedia for Automotive

DSP Algorithms

DESIGN TOOLS

New ISE 7.1i Software

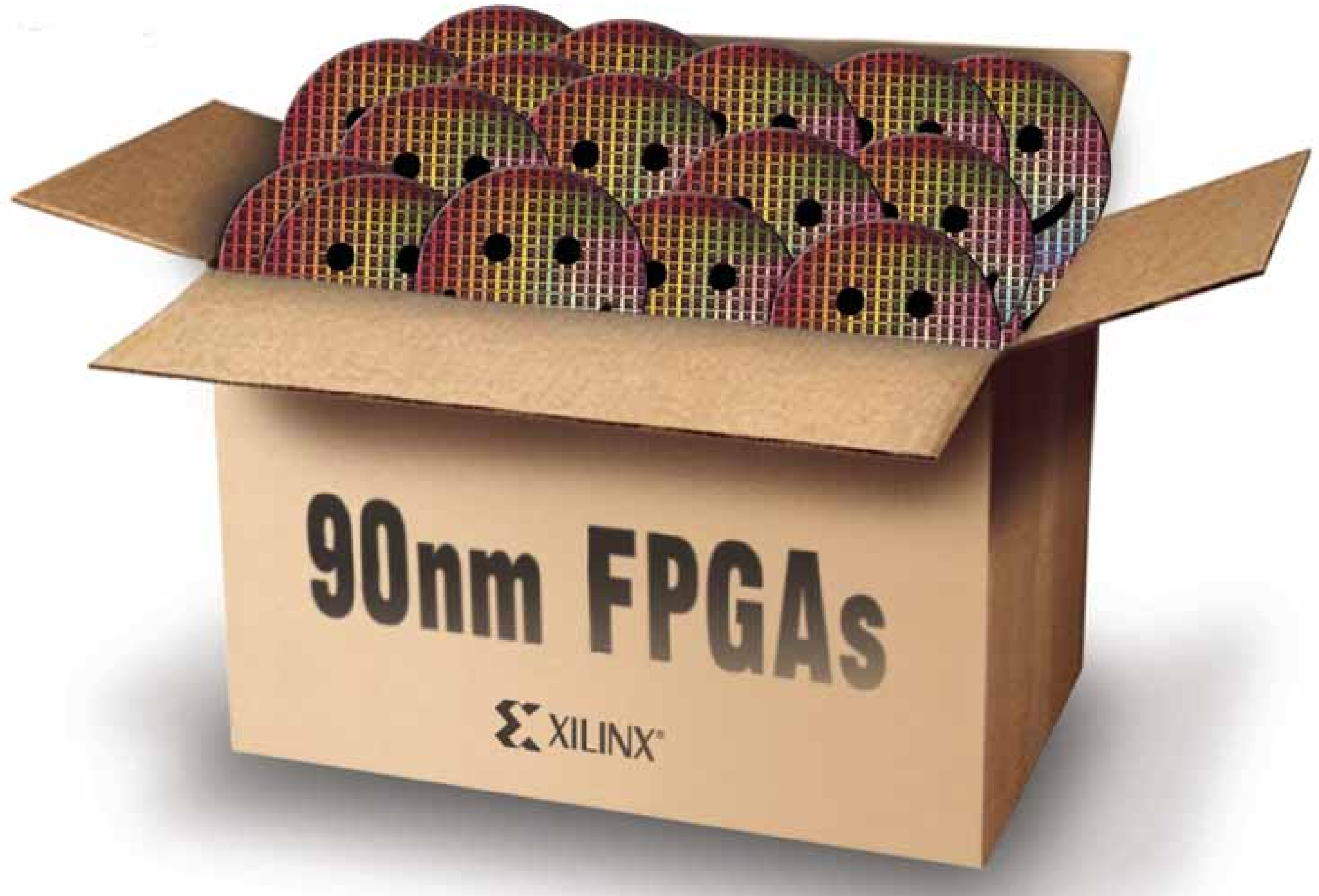
Control Your Designs

SERIAL I/O

Extend Your Reach

SUBSCRIBE NOW
SEE PAGE 3

XILINX®



Now Shipping In Volume!

Xilinx is the only FPGA supplier in the world to have achieved high-volume 90nm production, resulting in the lowest-cost FPGAs in the industry. Our leadership in 90nm products gives you all the performance and features you need, at the lowest price points ever.

Both our Spartan™ and Virtex™ product lines—the world's most widely adopted FPGAs—are shipping on our optimized 90nm process now. Contact your Xilinx rep today and let's ramp up for success together.

XILINX®

The Programmable Logic Company℠

www.xilinx.com/spartan3

Pb-free devices available now

·Proven 90nm Process
·Over 1 Million Units Shipped!

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

Xcell PUBLICATIONS

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

What's Both High and Low, and Used Everywhere...?

This issue of the *Xcell Journal* focuses on high-volume, low-cost consumer applications and the design tools used to implement them.

Business Viewpoints

Our Business Viewpoints column offers an independent business perspective by Rich Wawrzyniak, Senior Analyst, ASICs Services for Semico Research Corporation. His article, "Changing the Systems Landscape with Low-Cost FPGAs," discusses how advanced processing technology is driving down the costs of FPGAs and changing the ASSP/ASIC crossover point.

High-Volume Solutions

Approximately one year ago Xilinx® engineers were asked to develop the lowest cost, highest density FPGA family on the market. This section features articles using the new Spartan™-3E devices in several high-volume applications. Their low cost makes these devices successful in applications previously reserved for ASIC and gate-array technologies.

Industry Expert

In "Extend Your Reach," well-known signal integrity expert Howard Johnson discusses the Virtex™-4 RocketIO™ transceiver, which incorporates three forms of equalization to stretch the reach and performance of serial links.

Design Tools

Software support for the Spartan-3E family is available with the newly announced ISE™ 7.1i tools. This section has articles from Xilinx and some of its partners describing the use of their design tools to implement a complete solution.

Board Room

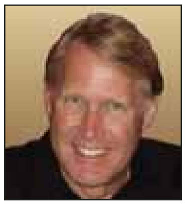
The Board Room section describes some of the Spartan-3 hardware development boards and other design solutions to help you determine which platform is best for your application and design task.

Power Play

The new Power Play section features technical descriptions of our partners' power solutions. In Steve Sharp's lead article, "Conquering the Three Challenges of Power Consumption," he writes, "To conquer the key challenges of power consumption, it takes a combination of good product design, proper device technology, and tools that let you take control of system power management."

And... It's Time to Re-Subscribe

Periodically, we must clean our mailing database. As of January 1, 2005, you must re-subscribe to continue receiving the *Xcell Journal* FREE. If you subscribed after January 1, 2005, you do not have to re-subscribe. If you subscribed before this date, please visit our website at *www.xilinx.com/xcell/subscribe/* and take a minute to renew your FREE subscription and ensure uninterrupted delivery.

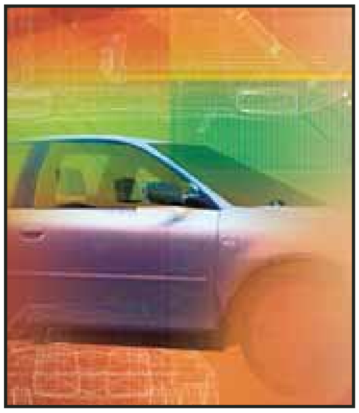
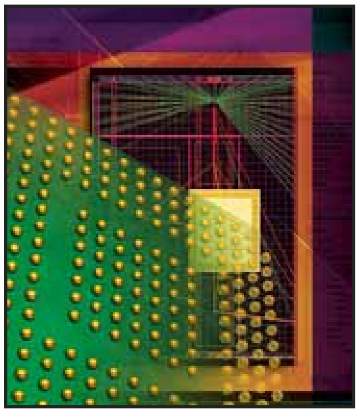
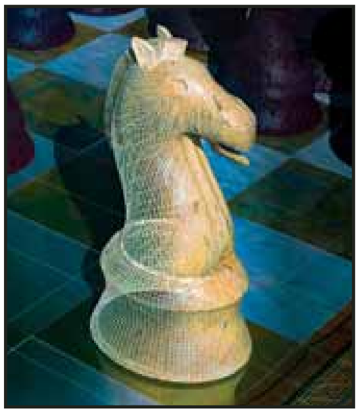


Forrest Couch

Forrest Couch
Managing Editor

SECOND QUARTER 2005, ISSUE 53

Xcell journal

Extend Your Reach

RocketIO transceivers included in the Virtex-4 FPGA family incorporate highly flexible equalization circuits that significantly extend the range and performance of high-speed serial links.

by Howard Johnson, Ph.D.
President
Signal Consulting, Inc.
howie03@sigcon.com

Mike Degerstrom
Senior Staff Signal Integrity Design Engineer
Xilinx, Inc.
mike.degerstrom@xilinx.com

Every multigigabit backplane, trace, and cable distorts the signals passing through it. This degradation may be slight or devastating, depending on the conductor geometry, materials, length, and type of connectors used.

Because they spend their lives working with sine waves, communications engineers like to characterize this distortion in the frequency domain. Figure 1 shows the channel gain, also called the frequency response, of a perfectly terminated typical 50 ohm stripline (or 100 ohm differential stripline). This stripline acts like a low-pass filter, attenuating high-frequency sine waves more than lower frequency waves.

Figure 2 illustrates the degradation inherent to a digital signal passing through 20 inches (.5 meters) of FR-4 stripline. The dielectric and skin-effect losses in the trace reduce the amplitude of the incident pulse and disperse its rising and falling edges. We like to call the received pulse, much smaller than normal, a “runt pulse.” In a binary communication system, any runt pulse that fails to cross the receiver threshold by a sufficient margin causes a bit error.

For the purposes of this discussion, three things degrade the amplitude of the runt pulse in a high-speed serial link: losses in the traces or cables, reflections due to connectors and other signal transitions, and the limited bandwidth of the driver and receiver.

A classic test of dispersion appears in Figure 3. This particular waveform – adjusted so that the long flat portions of the test signal represent the worst-case, longest runs of ones or zeros available in your data code – displays the runt-pulse amplitude. In the absence of reflections, crosstalk, or other noise, this single waveform (as measured at the receiver) represents a worst-case test of channel dispersion. Longer traces introduce progressively more dispersion, eventually causing receiver failure at (in this example) a length of 1.5 meters.

One measure of signal quality at the receiver is voltage margin. This number equals the minimum distance (in volts) between the signal amplitude and the receiver threshold at the instant sampling occurs. In a system with zero reflections, crosstalk, or other noise you could theoretically operate with a very small voltage margin and still expect the system to operate perfectly.

In a practical system, however, you must maintain a healthy noise margin sufficient to soak up the maximum amplitude of all reflections, crosstalk, and other noise in the system, while still keeping the received signal sufficiently above the threshold to account for the limited bandwidth and noise inherent to the receiver.

Following the example in Figure 4, a runt-pulse amplitude equal to 85% of the nominal low-frequency signal amplitude exceeds the receiver threshold by only 35%, instead of the nominal 50%. A smaller runt pulse with amplitude 75% of the normal size would reduce the voltage margin by half – a huge hit to your noise budget, but still workable. For generic binary communication using no equalization, we would like to see the runt pulse arrive with amplitude never smaller than 70% of the low-frequency pulse amplitude.

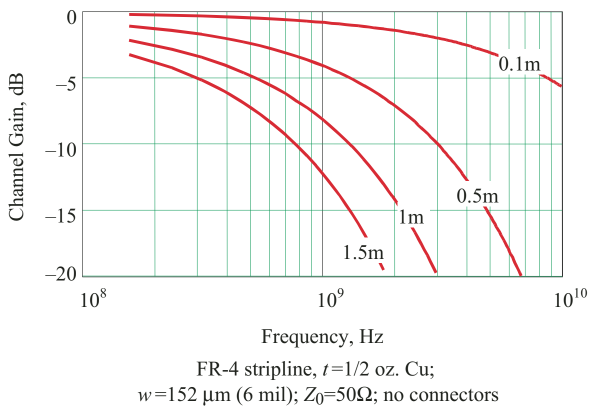


Figure 1 – The effective channel gain associated with a long PCB trace depends on the trace width, dielectric materials, length, and type of connectors used.

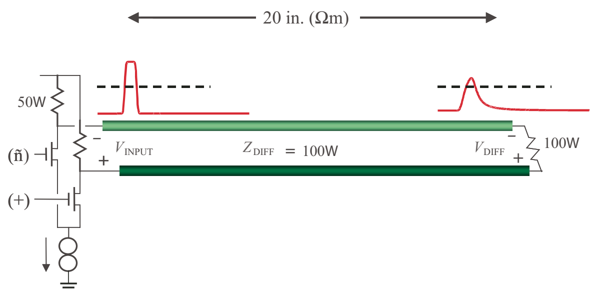


Figure 2 – Long traces reduce the amplitude of the input pulse and disperse its rising and falling edges.

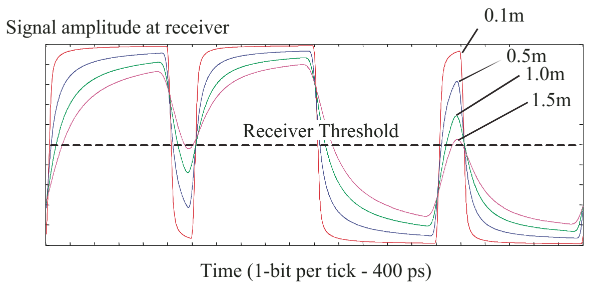


Figure 3 – This test waveform displays the worst-case runt-pulse amplitude.

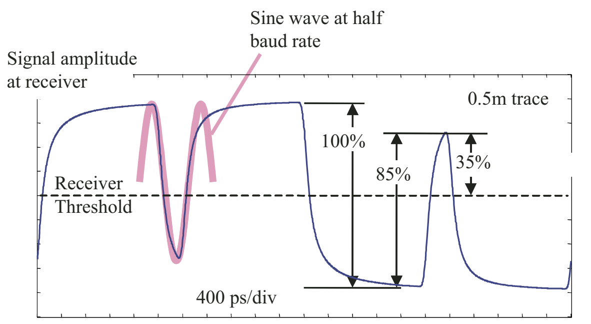


Figure 4 – A runt-pulse amplitude equal to 85% of the nominal low-frequency signal amplitude reduces the voltage margin above the threshold to only 35%, instead of the nominal 50%.

Runt-Pulse Degradation

On the left side of Figure 4 is a sine wave with a period of two baud. To the extent that the runt-pulse pattern (101) looks somewhat like this sine wave, you should be able to infer the runt-pulse amplitude from a frequency-domain plot of channel attenuation. Let's try it.

In Figure 4, the data waveform has a baud rate of 2.5 Gbps. One half this frequency (the equivalent sine wave frequency) equals 1.25 GHz. According to Figure 5, the half-meter curve gives you 4.5 dB of attenuation at 1.25 GHz. The same curve also shows 1.5 dB of attenuation at 1/10th this frequency, corresponding roughly to the lowest frequency of interest in an 8B10B coded data transmission system. The difference between these two numbers (-3 dB) approximates the ratio of runt-pulse amplitude to low-frequency signal amplitude at the receiver. With only -3dB degradation, the system satisfies our 70% frequency-domain criterion for solid link performance – precisely explaining why time-domain waveforms look so good at a half-meter.

Looking closely at Figure 4, the actual runt-pulse amplitude in the time domain is 85%, not quite as bad as the -3dB predicted by our quick frequency-domain approximation. This discrepancy arises partly from the harmonic construction of a square wave, where the fundamental amplitude exceeds the amplitude of the square wave signal from which it is extracted, and partly from the natural fuzziness inherent to any quick rule-of-thumb translation between the time and frequency domains. The simple frequency-domain criteria conservatively estimates these factors.

If your data code permits longer runs of zeros or ones than 8B10B coding, then you must use a correspondingly lower frequency as your "lowest frequency of interest." In the time domain, you will see the

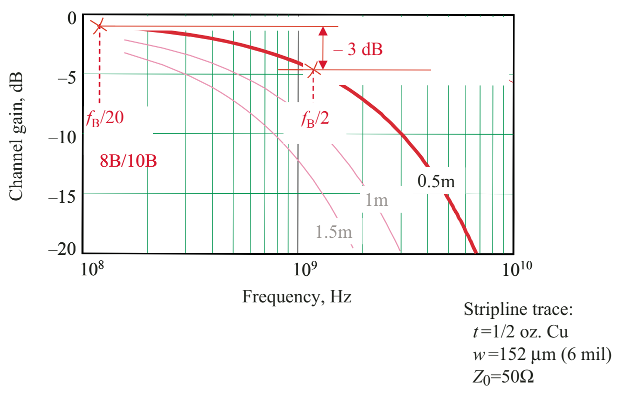


Figure 5 – The difference between high-frequency and low-frequency channel gain in this 2.5 Gbps system equals 3 dB.

received signal creep closer to the floor (or ceiling) of its maximum range before the runt pulse occurs, making it even more difficult for the worst-case runt pulse to cross the threshold.

As a rule of thumb, we look at the difference between the channel attenuation at the highest frequency of operation (the 101010 pattern) and the lowest frequency of operation (determined by your data coding run length) to quickly estimate the degree of runt-pulse amplitude degradation at the receiver. This simple frequency-domain method only crudely estimates link performance. It cannot substitute for rigorous time-domain simulation, but it can greatly improve your understanding of link behavior.

A channel with less than 1 dB of runt-pulse degradation works great with just about any ordinary CMOS logic family, assuming that you solve the clock skew problem either with low-skew clock distribution or by using a clock recovery unit at the receiver. A channel with as much as 3 dB degradation requires nothing more sophisticated than a good differential architecture with tightly placed well-controlled receiver thresholds. A channel with 6 dB of degradation requires equalization.

Transmit Pre-Emphasis

The Xilinx® Virtex™-4 RocketIO™ transceiver incorporates three forms of equalization that extend your reach on deeply degraded channels. The first is transmit pre-emphasis.

Figure 6 illustrates a simple binary waveform x[n] and the related first-difference waveform x[n]-x[n-1]. If you are familiar with calculus, you can think of the first-difference waveform as a kind of derivative operation. On every edge, the difference waveform creates a big kick. The transmit pre-emphasis circuit adds together a certain proportion of the main signal and the first difference waveform to superimpose the big kick at the beginning of every transition. As viewed by the receiver, each kick boosts the amplitude of the runt pulses without enlarging low-frequency portions of your signal, which are already too big.

The first-difference idea helps you see how pre-emphasis works, but that is not how it is built. The actual circuit sums not two but three delayed terms, called the pre-cursor, cursor, and post-cursor. This architecture gives you the capacity to realize both first and second differences by adjusting the coefficients associated with these three terms. Programmable 5-bit multiplying DACs control the three coefficients. The first and third amplitudes are always inverted with respect to the main center term, a trick that is accomplished by using the NOT-Q outputs of the first and third flip-flops. As an example, Figure 7 plots the frequency response corresponding to the particular coefficient set [-0.056, 0.716, -0.228].

Over the critical range from DC to 1.25 GHz, the pre-emphasis response rises smoothly – just the opposite of the plummeting curves drawn in Figure 5. The response peaks at 1.25 GHz. If you clock this pre-emphasis circuit at a higher data rate, the peak shifts correspondingly higher, always appearing just where you want it at a frequency equal to half the data rate.

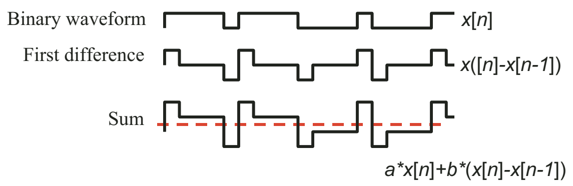


Figure 6 – The transmit pre-emphasis circuit creates a big kick at the beginning of every transition.

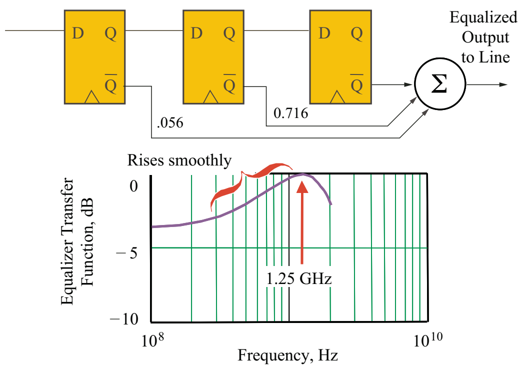


Figure 7 – Over the critical range from DC to 1.25 GHz, the pre-emphasis response rises smoothly.

Figure 8 overlays the pre-emphasis response with the channel response at 1 meter, showing a composite result (the equalized channel) that appears much flatter than either curve alone. In very simplistic terms, a flatter composite channel response should make a better-looking signal in the time domain.

The time-domain benefits of pre-emphasis appear in Figure 9. At shorter distances the signal appears over-equalized. The overshoot at each transition works fine in a binary system, assuming that the receiver has ample headroom to avoid saturation with the maximum-sized signal. At 1 meter, the signal looks quite nice, with very little runt-pulse degradation visible and (if you look closely) very little jitter. The 1.5 meter waveform now just meets the 70% criteria for runt-pulse success.

Compared to a simple differential architecture, the pre-emphasis circuit has at least doubled the length of channel over which you may safely operate.

Linear Receive Equalizer

In addition to the pre-emphasis circuit, the RocketIO transceiver also incorporates a sophisticated 6-zero, 9-pole receive-based linear equalizer. This circuit precedes the data slicer. It comprises three cascaded stages of active analog equalization that may be individually enabled, turning on zero, one, two, or all three stages in succession.

Figure 10 presents the set of four possible frequency-response curves attainable with this receiver-equalization architecture. Each section of the equalizer is tuned to approximate the channel response of a typical PCB channel with an attenuation of about 3 dB at 2.5 GHz. With all stages on, you get a little more than 9 dB of boost at 2.5 GHz. Because the response keeps rising all the way to 5 GHz, this equalizer is useful for data rates up to and beyond 10 Gbps.

When setting up the equalizer, first select the number of sections of the RX linear equalizer that best match your overall channel response. Then fine-tune the overall pulse response using the 5-bit programmable coefficients in the transmit pre-emphasis circuit to obtain the lowest ISI, the lowest jitter, or a combination of both. After building the circuit, a clock phase adjustment internal to the receiver helps you map out bit error rate (BER) bathtub curves, so you can corroborate the correctness of your equalizer settings.

The flexibility provided by these two forms of equalization lets you interoperate with an amazing array of serial-link standards, meeting exact transmitted signal specifications and at the same time adding receiver-based equalization to keep your system working at the peak of performance.

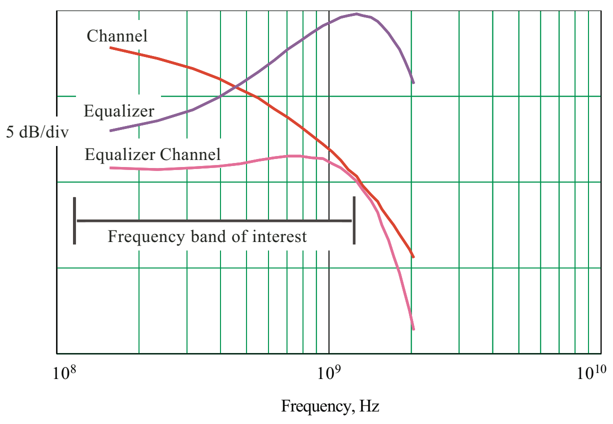


Figure 8 – Composing the pre-emphasis circuit with the channel produces an overall response much flatter than either curve alone.

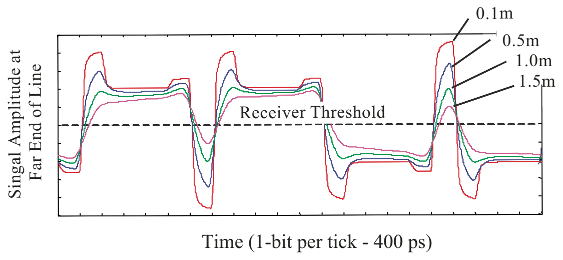


Figure 9 – A pre-emphasis circuit at least doubles the length of channel over which you may safely operate.

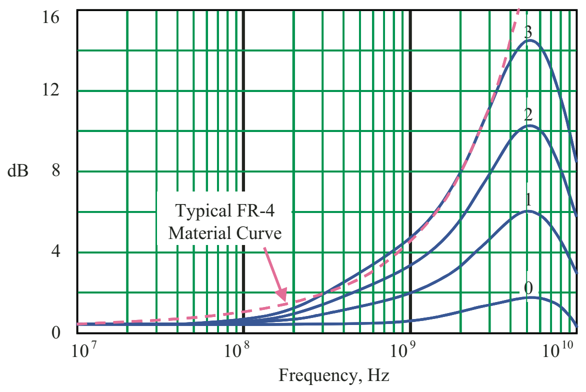


Figure 10 – The linear equalizer in the receiver may be set to one of four distinct response curves preprogrammed to match the response of various lengths of FR-4 PCB trace.

Decision-Feedback Equalizer

As a last defense against the slings and arrows of uncertain channel performance, the RocketIO transceiver includes a manually adjustable six-tap decision-feedback equalizer (DFE). This device is integrated into the slicer circuit at the receiver. The DFE is particularly useful with poor-quality legacy channels not initially designed to handle high serial data rates. It has the remarkable property of accentuating the incoming signal without exacerbating crosstalk.

Those of you familiar with signal processing will recognize that a DFE inserts poles into the equalization network, while a TX pre-emphasis circuit creates zeros. (A very accessible book about digital equalization, including DFE circuits, is John A.C. Bingham's "The Theory and Practice of Modem Design.")

Working together, the DFE, TX-pre-emphasis, and RX linear equalizer provide an incredibly rich array of possible adjustments.

Conclusion

For any channel with as much as 6 dB of runt-pulse degradation, a simple pre-emphasis adjustment easily doubles the length at which your link operates.

If you anticipate more than 6 dB of runt-pulse degradation, we strongly suggest that you simulate your system in detail before making the final equalizer adjustments. Contact your local Xilinx customer support office or visit the Xilinx website to obtain the necessary RocketIO models and associated design kits for modeling your channel. The modeling effort is well worth it, as equalization can substantially extend the reach of your circuits.

Howard Johnson, PhD, is the author of High-Speed Digital Design *and* High-Speed Signal Propagation. *He frequently conducts technical workshops for digital engineers at Oxford University and other sites worldwide. For more information, visit* www.sigcon.com, *or e-mail* howie03@sigcon.com.

Figures 1, 3, 4, and 9 are adapted with permission from Johnson and Graham, *High-Speed Signal Propagation: Advanced Black Magic*, Prentice-Hall, 2003.

Changing the Systems Landscape with Low-Cost FPGAs

The advent of low-cost FPGAs signals the dawn of a new era.

by Richard Wawrzyniak
Sr. Market Analyst: ASIC and SoC
Semico Research Corp.
richw@semico.com

We are living in very exciting times today, with new, compelling consumer applications springing up seemingly out of nowhere. These exciting times can be applied specifically to FPGAs, as we continue to see growth and widespread acceptance of the concept of programmable logic by the marketplace at large.

The wind in the sails of the FPGA industry is our old friend Moore's Law, which states that semiconductor devices will see a doubling of available transistors every 18 to 24 months. The increase in transistor budgets available to designers and the movement to ever-tighter process geometries translates directly into devices with rising performance levels and smaller die areas.

A good example of this is the recent introduction of FPGAs produced using 90 nm process geometries. Smaller geometries have allowed FPGAs with more logic elements and embedded memory, while at the same time decreasing the die area needed for the solution. A part with a given performance level using 130 nm process geometries can now be produced with a higher performance level in a smaller die area at 90 nm. Die area and yield have a great impact on device costs, so FPGA manufacturers can offer these new 90 nm families at decreasing price points.

Why is this important, and how does it change the systems landscape? Lower price points for a given performance level allow system designers and system architects to have an end product with increased performance and a lower BOM. For systems manufacturers, this leads to increased profitability, a lower overall system cost, or both.

At face value, this situation seems rather straightforward: lower device costs equal lower system costs – everyone is a winner. However, the real answer to this question goes much deeper than lower costs. To gain a better perspective on this issue, we must explore some of the background of the FPGA market and the ASIC industry as a whole.

A Difficult Design Environment

In today's market environment, there are a series of trends adversely influencing the ASIC design community. One of the trends most often discussed is the rise in NRE, which are costs associated with the design of ASICs. Figure 1 shows how these costs have increased by process geometry.

NRE costs have risen in large part because of the rising device complexity. As the possible gate counts have increased at each process node, so too has the task of assembling these gates into a design that meets the needs of the market.

With its low-to-none NRE costs, programmable logic is a great aid to designers who need to prove out their designs in the shortest amount of time to meet a moving market window but do not have the engineering or financial resources necessary to commission a new system-on-chip design.

Figure 2 shows the relationship between design cycle times, gate count, and product life cycles. Designers are under pressure to hit a market window that is shrinking, while at the same time their designs are becoming more complex. These trends are at odds with product life cycles, which are also shrinking.

The net effect of these trends is to make the design environment more challenging for ASIC designers. Even before a design is complete, the market at which it is aimed could have changed – necessitating changes in the design itself. This quickly becomes a no-win situation for all concerned.

Furthermore, as the cost to create a complex ASIC solution increases, so too must the size of the market towards which the solution is targeted. If the design costs for a complex ASIC are in the $30 million range, the size of its target market must be many times larger than the design costs just to recover the initial investment in the silicon. Admittedly, there are not many applications today that can command such high unit volumes to guarantee the recovery of this initial investment.

An additional detrimental effect is the decline in ASIC design starts because of rising design costs, decreasing product life cycles, and shrinking market windows. Several of these trends and ASIC design start numbers for various ASIC product types (including FPGAs and PLDs by end application) are detailed in an upcoming report from Semico titled, "ASIC Design Starts: An Industry in Transition."

Low-Cost FPGAs Change the Equation

Now that we've reviewed current trends in the ASIC market, we can now look at how low-cost FPGAs can change the landscape to one more favorable for designers.

Designers can now apply the traditional strengths of programmable logic to their system solutions – namely the low cost of entry, off-the-shelf availability, short design cycles, reprogrammability itself, and a known and working architecture. Plus, they can use FPGAs as test vehicles to examine and prove out different types of semiconductor intellectual property (SIP) with an ease that is impossible in standard cell or SoC markets. ASIC designers can test out several different types of SIP using FPGAs in the same time it took just to try out one type of SIP using the traditional standard cell or SoC route. This in itself is a great aid to ASIC designers in arriving at the best possible silicon solution.

However, the road for FPGAs does not end here; this is merely the start of the story in terms of aiding ASIC designers. Previously, when FPGA silicon was more costly, ASIC designers could successfully prototype with FPGAs and enter into limited production with their FPGA solution. But at some point, usually between 30,000 and 100,000 units (depending on the price sensitivity of the end application), the design would change over to a full-blown ASIC solution. At this point, many of the gains seen through the use of programmable logic were given up; designers were back on a more traditional ASIC path, with all the previously mentioned problems and pitfalls.

As shown in Figure 3, low-cost FPGAs have pushed out the point at which a standard cell or SoC solution must be employed to arrive at the absolute lowest cost. This chart also takes into consideration the total cost to create the ASIC solution (NRE charges and mask set costs, for example), as compared to the same cost to create the silicon solution using programmable logic. The benefits are numerous, particularly the ability to use a silicon solution that fits into market windows and is at the right price point.

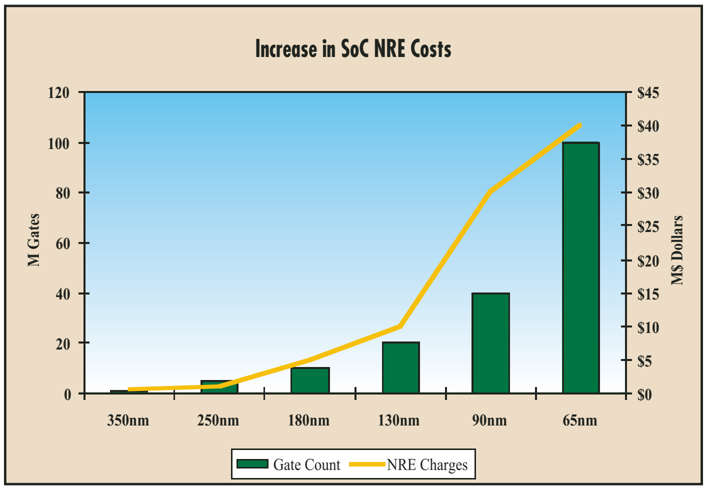


Figure 1 - NRE costs by process geometry Source: Semico Research Corp.

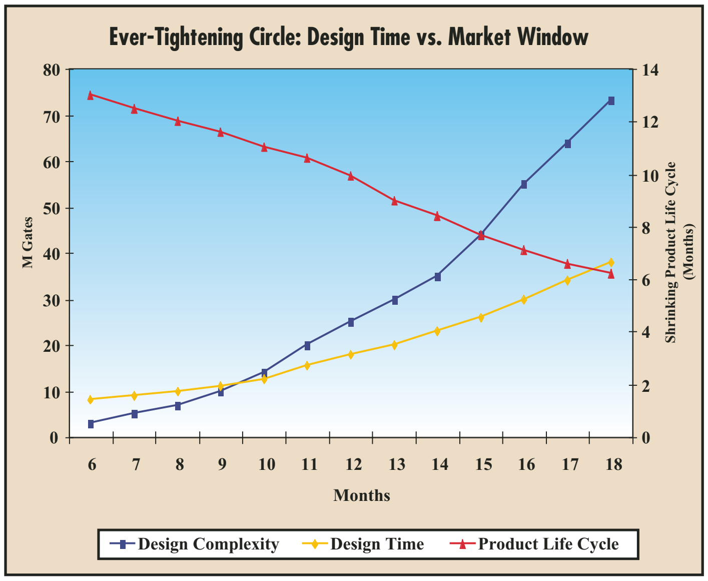


Figure 2 - Comparison of design and market trends Source: Semico Research Corp.

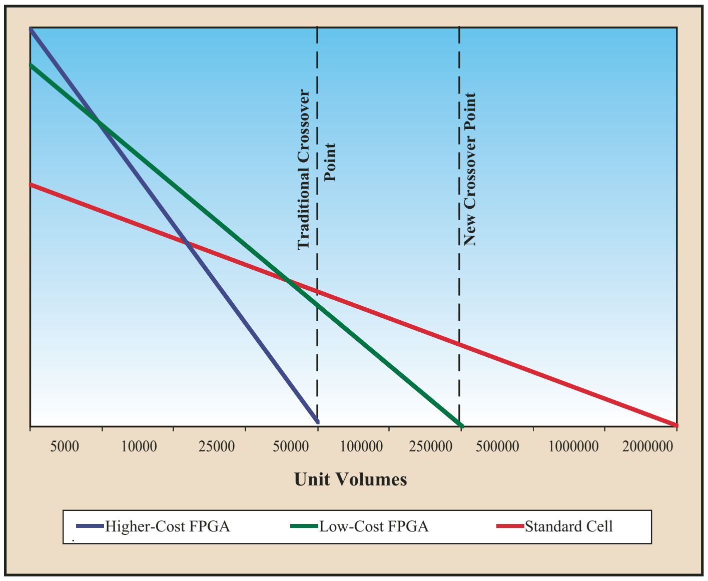


Figure 3 - Deeper volume penetration by FPGAs Source: Semico Research Corp.

Conclusion

As FPGA vendors continue to use smaller process geometries when they become available, the ASIC industry will see a further extension of FPGAs into applications where they can deliver a solid solution at the necessary price points.

Designs and end-system solutions that might not have made it into production because of a prohibitively long design cycle now have a much better chance of succeeding, whereas before they might have been cancelled because they couldn't make that ever-important market window.

The introduction of low-cost, full-featured FPGAs has changed the systems landscape for the better. Today, ASIC designers or system architects can use a programmable logic solution with the knowledge that a FPGA can be used farther into the production cycle than was previously possible.

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.

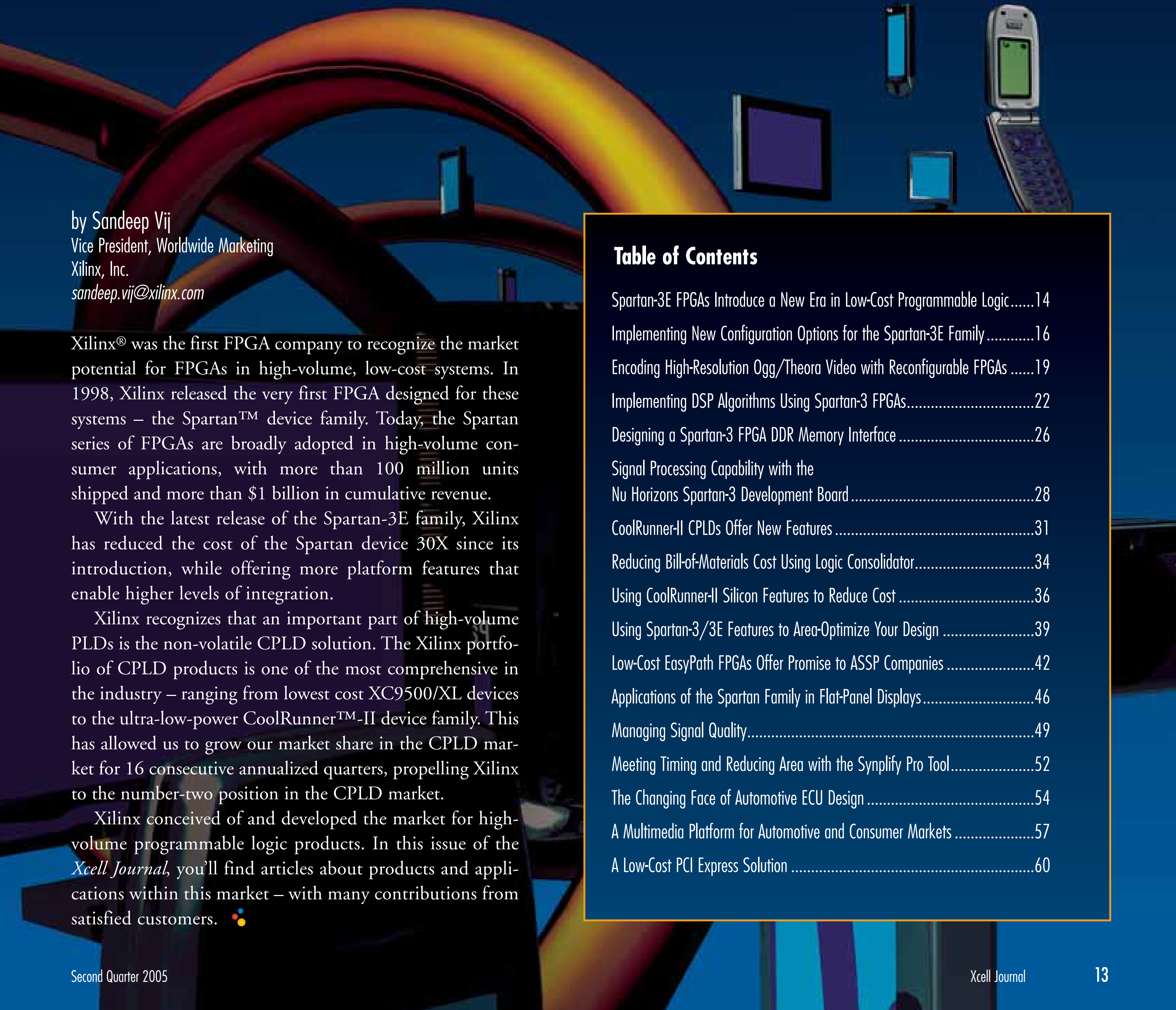
You'll also be able to speed up system analysis with the 16900's hosted power mode—which enables you and your team to remotely access and operate the 16900 over the network from your fastest PCs.

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

Leading the High-Volume Programmable Revolution

Xilinx pioneered – and leads – the adoption of PLDs in low-cost systems.



by Sandeep Vij
Vice President, Worldwide Marketing
Xilinx, Inc.
sandeep.vij@xilinx.com

Xilinx® was the first FPGA company to recognize the market potential for FPGAs in high-volume, low-cost systems. In 1998, Xilinx released the very first FPGA designed for these systems – the Spartan™ device family. Today, the Spartan series of FPGAs are broadly adopted in high-volume consumer applications, with more than 100 million units shipped and more than $1 billion in cumulative revenue.

With the latest release of the Spartan-3E family, Xilinx has reduced the cost of the Spartan device 30X since its introduction, while offering more platform features that enable higher levels of integration.

Xilinx recognizes that an important part of high-volume PLDs is the non-volatile CPLD solution. The Xilinx portfolio of CPLD products is one of the most comprehensive in the industry – ranging from lowest cost XC9500/XL devices to the ultra-low-power CoolRunner™-II device family. This has allowed us to grow our market share in the CPLD market for 16 consecutive annualized quarters, propelling Xilinx to the number-two position in the CPLD market.

Xilinx conceived of and developed the market for high-volume programmable logic products. In this issue of the *Xcell Journal*, you'll find articles about products and applications within this market – with many contributions from satisfied customers.

Table of Contents

Spartan-3E FPGAs Introduce a New Era in Low-Cost Programmable Logic

The Spartan-3E XC3S100E FPGA is the first 100,000-gate device available for under $2.

by Richard Terrill
Senior Manager, High-Volume Products Marketing
Xilinx, Inc.
richard.terrill@xilinx.com

First introduced in 1998, the Spartan™ family was the world's first FPGA series tailored for low-cost applications. With the introduction of the Spartan-3E family, Xilinx® now has seven families of Spartan FPGAs in production – with more than 100 million units shipped to date. The Spartan series features the world's most accepted low-cost FPGA architecture and is familiar to thousands of engineers.

Moore's law allows Xilinx to offer ever-lower prices in the Spartan series of FPGAs. The Spartan-3E family is our third Spartan family in 90 nm, and gives us the lowest possible manufacturing costs. These low costs make programmable logic successful in high-volume and low-cost production applications, an area previously reserved for ASIC and gate-array technologies.

The Spartan-3 family (introduced in 2003) is optimized for I/O-centric designs, and is ideally suited for systems that have large I/O requirements. The Spartan-3E family is optimized for gate-centric designs, and is well-suited for designs that require a relative higher gate-to-I/O ratio. The older Spartan-II/IIE and Spartan-XL families remain suitable candidates for legacy designs or for systems with higher core voltages.

Spartan-3 FPGAs have found remarkable success in the production of systems that typically would have used an ASIC or gate array. For example, many flat-panel display systems employ Spartan-3 devices to manage the display driver and control functions. The ability to modify the design after layout and adapt the system to changing market conditions makes FPGAs highly desirable.

Spartan-3E devices extend the reach of FPGAs into production volumes by further reducing costs, while preserving the low NRE and high flexibility of programmable logic. Many mainstream applications using Spartan-3E FPGAs will have an ASIC crossover point of 250,000 units – meaning that the total cost favors Spartan-3E devices over ASICs for the first quarter-million units of production.

The Spartan-3E Family

The Spartan-3E family is our newest low-cost FPGA family, and further reduces the price points for low-cost FPGAs to unprecedented levels. Through 90 nm process technology, 300 mm wafers, and application-driven architecture choices, Xilinx has extended FPGAs into volumes and applications previously reserved for mask-programmed ASICs. Spartan-3E devices offer one of the lowest costs-per-logic (CPL) of any FPGA.

Spartan-3E FPGAs have been architected for digital consumer applications, and all high-volume/low-cost applications will benefit from its advanced features and capabilities.

The five-member family ranges from the 100,000-gate XC3S100E through the 1.6 million-gate XC3S1600E, adding features such as 64/66 PCI, mini LVDS, and faster embedded multipliers for low-cost DSP, all to better serve low-cost applications. Table 1 highlights the key features of the Spartan-3E family.

	Spartan-3E FPGA Family				
	XC3S100E	XC3S250E	XC3S500E	XC3S1200E	XC3S1600E
System Gates	100K	250K	500K	1,200K	1,600K
Logic Cells	2,160	5,508	10,476	19,512	33,192
Block RAM Bits	72K	216K	360K	504K	648K
Distributed RAM Bits	15K	38K	73K	136K	231K
DCMs	2	4	4	8	8
Multipliers	4	12	20	28	36
I/O Standards	18	18	18	18	18
Max Single-Ended I/O	108	172	232	304	376
Max Differential I/O Pairs	40	68	92	124	156
	Package and I/O Offerings				
	XC3S100E	XC3S250E	XC3S500E	XC3S1200E	XC3S1600E
VQ100 14 x 14 mm	66	66			
CP132 8 x 8 mm		92	92		
TQ144 20 x 20 mm	108	108			
PQ208 28 x 28 mm		158	158		
FT256 17 x17 mm		172	190	190	
FG320 19 x 19 mm			232	250	250
FG400 21 x 21 mm				304	304
FG484 23 x 23 mm					376

Table 1 – Spartan-3E family key features matrix

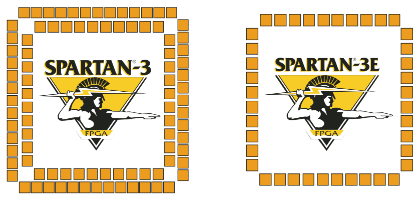


Figure 1 – Spartan-3 staggered I/O compared to Spartan-3E inline I/O

The Gate-Centric FPGA

Spartan-3E devices feature an optimized inline I/O ring for the absolute lowest cost. Figure 1 shows the inline I/O ring of Spartan-3E FPGAs versus the staggered I/O pad approach used by Spartan-3 devices. Inline I/Os are more efficient for smaller densities and allow us to add more logic for a given I/O count, as shown in Figure 2.

To further reduce die size, we refined the I/O layout, removed some less-common I/O standards, and resized the output buffers. Even a small area reduction in each I/O adds up to a relatively large savings because it is repeated for each I/O pad. Many of the decisions in the Spartan-3E architecture were driven by direct feedback from our high-volume and low-cost customers.

Spartan-3E FPGAs have a lower CPL compared to Spartan-3 devices and are the lowest cost FPGAs for gate-centric designs. Correspondingly, Spartan-3 devices are well suited for I/O-intensive FPGA designs. Together, Spartan-3 and Spartan-3E FPGAs will continue to meet customer needs in low-cost system design.

New Features in Spartan-3E Devices

Besides the optimized inline I/O ring to lower costs, Spartan-3E FPGAs also have a host of new features that include:

- Support for low-cost commodity Flash memory (SPI/BPI) configuration memory
- PCI 64/66 and PCI-X support
- DDR 333 memory interface
- Mini-LVDS, RSDS
- DCM clock frequency input range expanded down to 5 MHz (ideal for video)
- 325 MHz multipliers aligned with block RAM for low-cost DSP

One of the most significant new features available in Spartan-3E devices is support for low-cost serial commodity Flash configuration memory. With Spartan-3E FPGAs you can use generic, low-cost serial EPROMs or byte-wide Flash devices available from multiple vendors to configure the device. You can use the configuration memory for other system functions, and even reprogram it under the control of the Spartan-3E FPGA, adding tremendous flexibility for system designers.

Conclusion

With the arrival of the Spartan-3E FPGA family, there are now even more reasons to consider programmable logic in the production of low-cost systems. With prices starting under $2 and a substantially lower cost configuration solution, Spartan-3E devices will serve as the production solution for increasing numbers of low-cost, high-volume, and consumer applications.

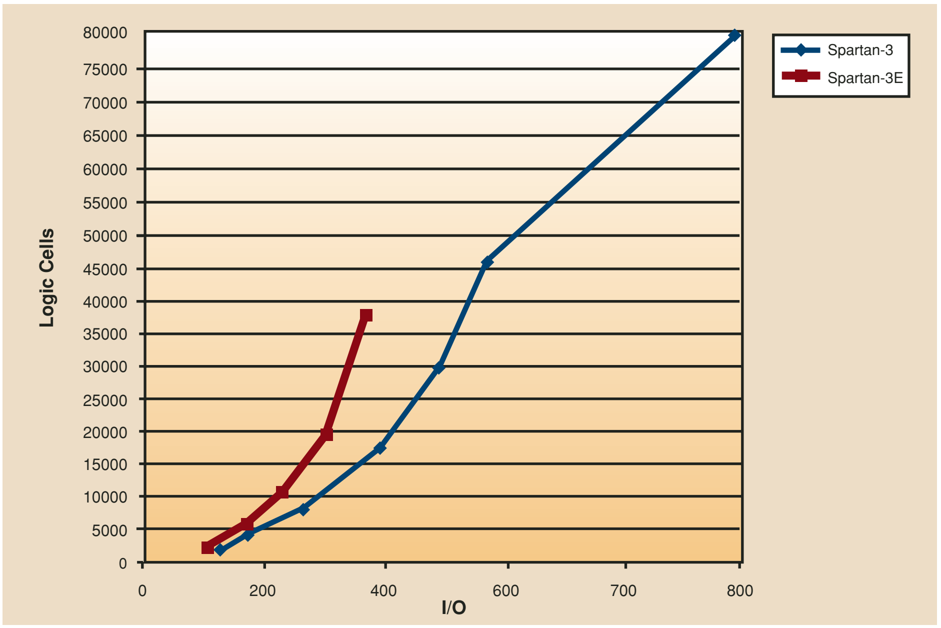


Figure 2 – I/O versus logic curves for Spartan-3 and Spartan-3E devices

Implementing New Configuration Options for the Spartan-3E Family

The new Spartan-3E family supports serial (SPI) and parallel flash memory for configuration.

by Steve Knapp
Sr. Applications Engineering Manager
Xilinx, Inc.
steve.knapp@xilinx.com

Ward Williams
Sr. Strategic Marketing Manager
Xilinx, Inc.
ward.williams@xilinx.com

Kirk Owyang
Sr. Solutions Marketing Manager
Xilinx, Inc.
kirk.owyang@xilinx.com

The new Xilinx® Spartan™-3E FPGA family reduces your total system cost in many ways, including new low-cost configuration memory options. You can now choose the configuration memory solution that best suits your specific application requirements. For configuration memory, you can choose between industry-standard, commodity serial peripheral interface (SPI) or parallel NOR flash PROMs, competitively priced Xilinx Platform Flash, or other low-cost memories with a microcontroller.

Nearly all of the Spartan-3E configuration pins become available as user I/Os after configuration. Consequently, you can leverage any remaining space in the configuration memory for application data, such as code for an embedded MicroBlaze™ processor, serial numbers, or Ethernet MAC IDs. SPI and parallel flash PROMs additionally offer random-accessible, byte-addressable, read/write memory. If the application requires additional space, upgrade to the next larger PROM. Most SPI and parallel flash PROMs are available in a common footprint across multiple densities.

Xilinx Platform Flash still provides an excellent solution for stand-alone two-chip programmable logic solutions (FPGA + dedicated configuration PROM), with competitive cost-per-megabit pricing. Platform Flash also excels with features such as JTAG in-system programmability and patented compression technology.

SPI flash PROMs are popular in high-volume consumer electronics applications, where they store system parameters or code for an embedded processor. SPI flash PROMs also offer a low-cost pin-saving configuration solution for Spartan-3E FPGAs. SPI flash memory is multi-sourced and multiple densities are available within the same package footprint.

Parallel NOR flash is the preferred solution for FPGA applications with an embedded processor such as the MicroBlaze soft-core processor. At power-on, the FPGA configures from one end of the parallel flash. After configuration, the MicroBlaze processor either executes directly from the opposite end of memory or shadows the code to external SDRAM.

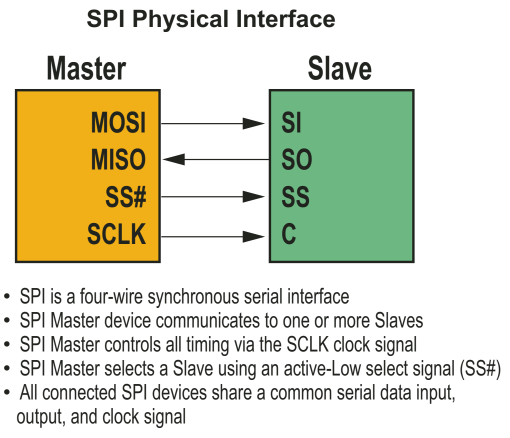

SPI Physical Interface
Master
Slave
MOSI
MISO
SS#
SCLK
SI
SO
SS
C
• SPI is a four-wire synchronous serial interface
• SPI Master device communicates to one or more Slaves
• SPI Master controls all timing via the SCLK clock signal
• SPI Master selects a Slave using an active-Low select signal (SS#)
• All connected SPI devices share a common serial data input, output, and clock signal

Figure 1 – SPI physical interface

SPI Serial Flash Vendor	Tested SPI Flash Family
Atmel	AT45DBxxx
NexFlash	NX25Pxx
Programmable Microelectronics Corporation (PMC)	Pm25LVxxx
Silicon Storage Technology (SST)	SST25LFxxxA SST25VFxxxA SST25VFxx *
ST Microelectronics	M25Pxx

* SST25VFxx supports only the READ (0X03) SPI read command; all others support the FAST_Read (0X0B) command.

Table 1 – SPI flash memory families supported by Spartan-3E FPGAs

Spartan-3E Device	Configuration Bits	Minimum SPI Device Capacity	Unused Memory After Configuration
XC3S100E	581K	1 Mb	443K
XC3S250E	1,352K	2 Mb	696K
XC3S500E	2,267K	4 Mb	1,829K
XC3S1200E	3,832K	4 Mb	264K
XC3S1600E	5,958K	8 Mb	2,234K

Table 2 – Required SPI device sizes and estimated cost

What is SPI?

The serial peripheral interface (SPI) interface is a four-wire synchronous interface (Figure 1). SPI was originally pioneered as a serial communications interface between CPUs, MCUs, peripherals, and other devices supporting this protocol. Now SPI is popular in the embedded processing and consumer electronics markets. Many modern microcontrollers use this interface, as well as a wide variety of third-party peripherals.

SPI Flash Suppliers

Although SPI is a standard four-wire interface, the various available SPI flash PROMs use different command protocols. Spartan-3E FPGAs support up to four different command protocols. Currently, SPI flash PROMS from five vendors (Atmel, NexFlash, Programmable Microelectronics Corporation [PMC], Silicon Storage Technology [SST], and ST Microelectronics) are tested and supported, as shown in Table 1. Devices from other SPI suppliers are being tested and will be supported in the near future.

Memory Requirements

A single Spartan-3E FPGA requires roughly 600K to 6M bits for configuration, a small amount of memory relative to the large SPI capacities available today. Table 2 lists the amount of configuration memory required by each Spartan-3E device type, the minimum required SPI device size, and the extra memory that is available for other purposes. Note that it is not necessary to have a dedicated configuration memory device for each FPGA.

Multiple FPGAs can share a single SPI flash PROM in a configuration daisy chain.

Interfacing to SPI

Figure 2 shows the typical connections between a Spartan-3E FPGA and SPI flash memory, as well as the FPGA I/O pins used to control the configuration procedure. When the FPGA is in SPI flash configuration mode, three dual-pur-

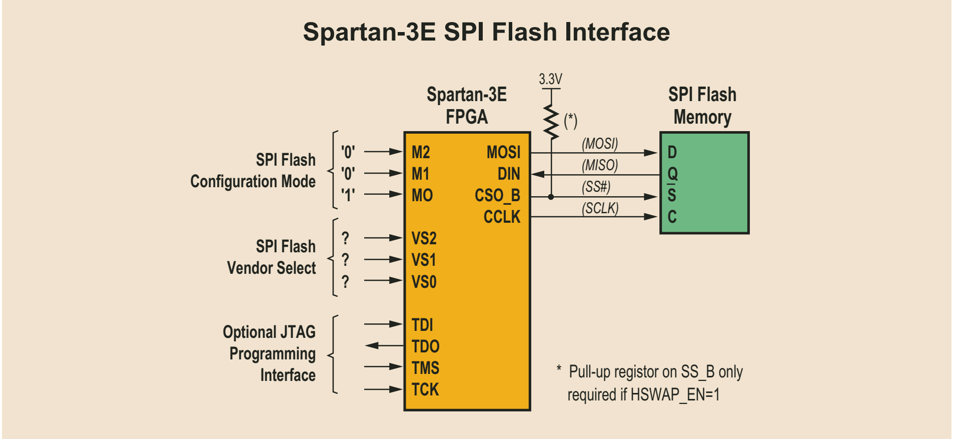
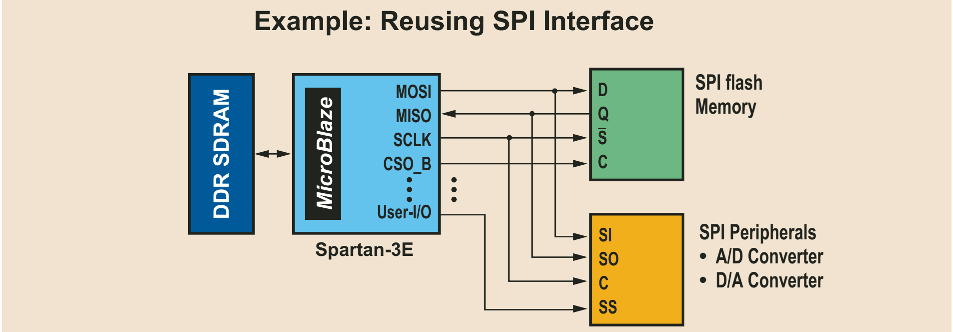

Spartan-3E SPI Flash Interface
Spartan-3E FPGA
3.3V
(*)
SPI Flash Memory
SPI Flash Configuration Mode
'0'
'0'
'1'
M2
M1
M0
MOSI
DIN
CSO_B
CCLK
(MOSI)
(MISO)
(SS#)
(SCLK)
D
Q
S
C
SPI Flash Vendor Select
?
?
?
VS2
VS1
VS0
Optional JTAG Programming Interface
TDI
TDO
TMS
TCK
* Pull-up registor on SS_B only required if HSWAP_EN=1

Figure 2 – Interface between Spartan-3E FPGA and SPI flash memory



- FPGA configures from SPI flash memory
- After configuration, the FPGA copies MicroBlaze code from SPI flash to external DDR SDRAM
- SPI flash available for non-volatile data or parameter storage
- Additional SPI peripherals require a single select pin per device

Figure 3 – Connecting SPI flash memory to multiple devices

pose pins, called VS[2:0], define the type of attached SPI flash and operate as follows:

- These pins are only activated in SPI configuration mode (M[2:0]="001")
- The VS pins are sampled when INIT_B goes high
- The VS pins are reusable as user I/O after FPGA configuration completes (DONE goes high)

Then, the FPGA issues the command sequence appropriate for the selected SPI flash.

After Configuration

After configuration, all of the pins connected to the SPI flash PROM are available as user I/O pins. If not using the SPI flash PROM after configuration, drive the FPGA's CSO_B pin high to disable the PROM, freeing the FPGA's MOSI, DIN, and CCLK pins as user I/O.

If large enough, the SPI flash PROM can also contain non-volatile application data, such as MicroBlaze processor code or data such as serial numbers and Ethernet MAC IDs. Figure 3 shows an example of using SPI flash memory for multiple purposes.

Third-Party Peripherals

In addition to flash memory, many peripherals utilize the same SPI. These include:

- Memories (EEPROM, EPROM)
- Analog-to-digital converters (ADCs)
- Digital-to-analog converters (DACs)
- Thermal management
- Display drivers
- Microprocessors, microcontrollers (MCUs), and digital signal processors (DSPs)
- Many application-specific standard products (ASSPs)

After configuration, the FPGA user application can remain as the SPI bus master and communicate with the attached SPI flash PROM and any attached SPI peripherals. All of the attached SPI peripherals share common serial-input, serial-output, and clock signals. Each SPI peripheral has a separate select input. The SPI flash PROM used for configuration is selected through the FPGA's CSO_B pin. Each additional SPI peripheral is selected by a separate user I/O pin.

Parallel NOR Flash

Spartan-3E FPGAs also provide a parallel configuration interface, primarily designed to configure the FPGA from industry-standard parallel NOR flash. The FPGA's asynchronous memory interface is flexible and can connect to a variety of memory devices such as EEPROM, OTP EPROM, masked ROM, NVRAM, or even asynchronous SRAM.

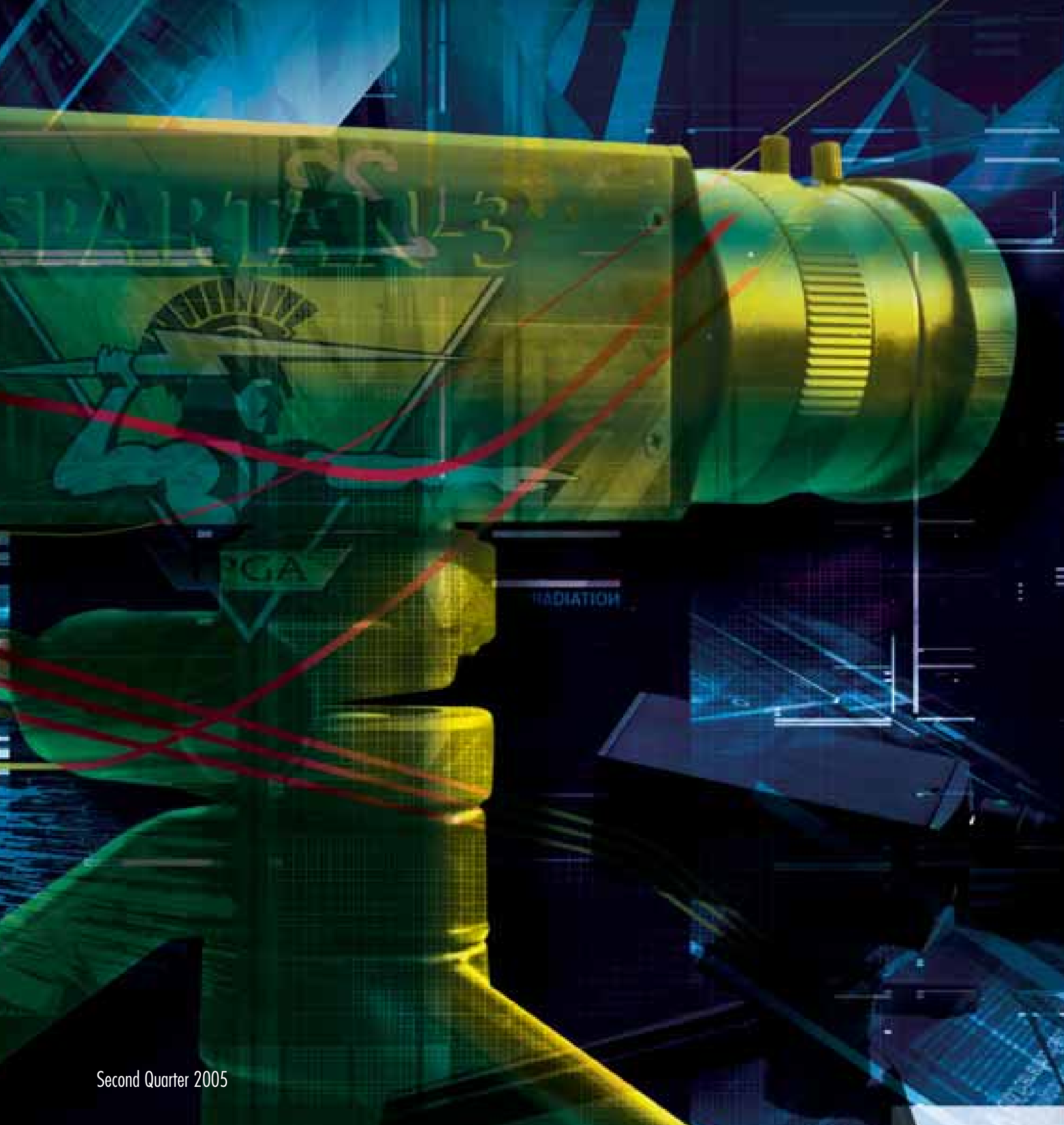
The memory requirements are fairly simple. During configuration, the FPGA provides up to 24 address lines and receives byte-wide data. The FPGA has four control lines driving the memory's chip-select, output-enable, write-enable, and an optional byte-enable for high-density flash PROMs with a x8/x16 mode control (BYTE#). The memory must be 3.3V, 2.5V, or 1.8V and must have a read access time of 200 ns or faster.

Conclusion

In addition to low device costs, the new low-cost Spartan-3E FPGA family lowers overall system cost in many ways, including new support for inexpensive SPI and parallel flash memory for configuration. Currently, a number of flash memory families are supported and more memory vendors will announce their support for Spartan-3E devices in the coming months. Please check the Spartan-3E pages on the Xilinx website for a current list of supported SPI flash, *www.xilinx.com/spartan3e/*.

Encoding High-Resolution Ogg/Theora Video with Reconfigurable FPGAs

Once the traditional application area of custom ASICs, modern FPGAs can now handle high-performance video encoding.



by Andrey Filippov
President
Elphel, Inc.
andrey@elphel.com

Much of the Spring 2003 issue of the *Xcell Journal* in which my article about Spartan™-IIE-based Elphel Model 313 cameras appeared ("How to Use Free Software in FPGA Embedded Designs") was dedicated to the Xilinx® Spartan-3 FPGA. I immediately started to think about using these devices in our new generation of Elphel network cameras, but it wasn't until last year that I was finally able to start working with them.

One of the factors that slowed my company's adoption of this new technology was the fact that at first I could not find appropriate software that could handle the devices selected, as it is essential that our end users can modify our products without expensive software development tools. When I visited the Xilinx website in Summer 2004 and found that the current version of the free downloadable WebPACK™ software could handle the XC3S1000 – the largest device available in a small FT256 package – I knew it was the right time to switch to the Spartan-3 device.

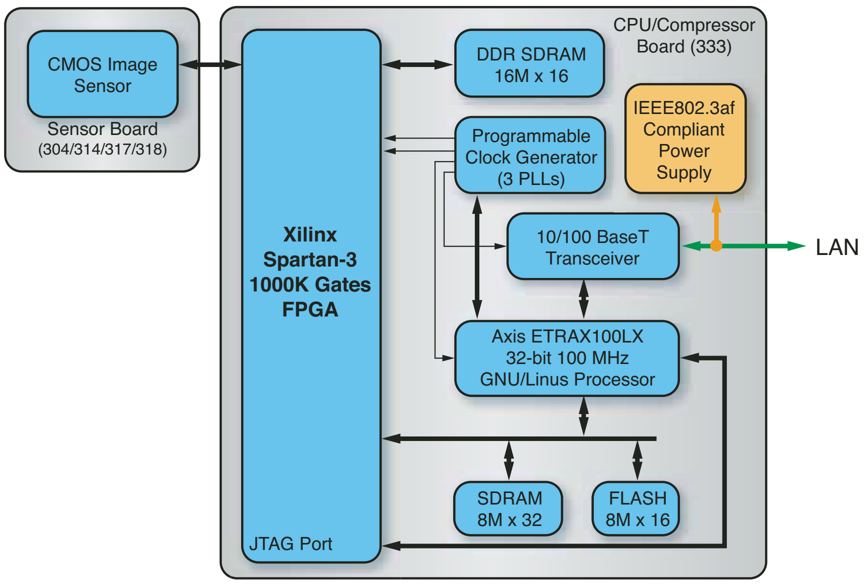

CMOS Image Sensor
Sensor Board (304/314/317/318)
Xilinx Spartan-3 1000K Gates FPGA
JTAG Port
CPU/Compressor Board (333)
DDR SDRAM 16M x 16
Programmable Clock Generator (3 PLLs)
IEEE802.3af Compliant Power Supply
10/100 BaseT Transceiver
LAN
Axis ETRAX100LX 32-bit 100 MHz GNU/Linus Processor
SDRAM 8M x 32
FLASH 8M x 16

Figure 1 – Camera system block diagram

The Camera Hardware

The new Model 333 camera (Figure 1) uses the same Linux-optimized CPU (ETRAX100LX by Axis Communications) as the earlier Model 313, but with increased system memory – 32 MB of SDRAM and 16 MB of Flash. The second major upgrade is the use of 32 MB of DDR SDRAM as a dedicated frame buffer that works in tandem with the FPGA, supplementing its processing power with high capacity and I/O bandwidth.

The Spartan-3 DDR I/O functionality made it possible to increase the memory bandwidth without increasing board size – the complete system still fits on a 1.5 x 3.5-inch four-layer board (see Figure 2). The actual board area is even smaller, as the new one is designed to fit the sealed RJ45 connectors for outdoor applications.

For the camera circuit design, the goals include combining high computational performance with small size (that also simplifies preserving high-speed signal integrity on the PCB) and providing the flexibility for the reconfigurable FPGA on the system level. For the latter, I decided to split the camera circuitry into two boards: one main board and a second containing just a sensor with minimal related components. On the main board the FPGA I/O pins go directly to the inter-board connector, so it is possible to change the pin functions (including polarity) to match the particular sensor boards. A similar solution allowed the earlier Model 313 camera to support different types of sensors (most became available after the board design). It even works in our 11-megapixel Model 323 cameras without any PCB modifications.

Selecting the Video Encoding Technique

After the prototype camera was ready, it took just a couple of weeks to modify the code developed for the Spartan-IIE-based camera and to implement motion JPEG compression. Half of that time was spent trying to figure out how to configure the new FPGA with the generated bitstream. In the camera, JTAG pins of the device are connected directly to the processor I/O pins, so I could not use the software that comes with Xilinx configuration hardware. The JTAG instruction register is six bits wide, not five as it was in the Spartan-IIE devices with which I was familiar. After some trial and error, I figured that out and found that the same code could run at 125 MHz (instead of 90 MHz in the previous model) and used just 36% (not 98% as before) of available slices – plenty of room for more challenging tasks.

Of course, I had some challenging tasks in mind, as motion JPEG is not a really good option for high-resolution/high-frame-rate cameras because the amount of data to be transferred or stored is quite huge. It is a waste of network bandwidth or hard disk space when recording such video streams, as fixed-view cameras in most cases have very little difference between consecutive frames. Something like MPEG-2 could make a difference; that was the standard I was planning to implement in the camera.

But as soon as I got some books on MPEG-2 and started combing through online resources, I found another fundamental difference between MPEG and JPEG – not just that it can use the similarity between consecutive frames. Contrary to JPEG, MPEG-2 requires you to pay licensing fees for using the encoders based on this standard. The fee is small compared to the cost of the hardware, but it still could be a hassle and does not provide freedom for implementation.

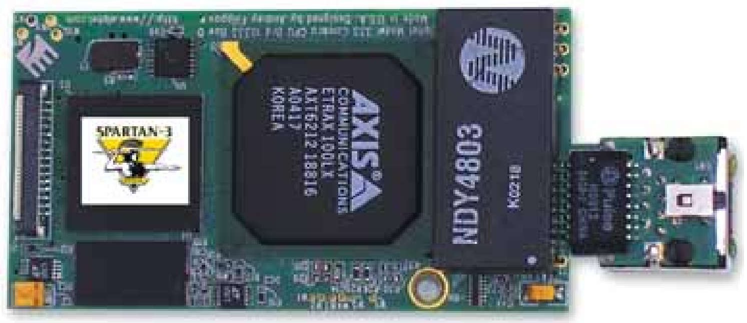


Figure 2 – Camera system board

It did not take long to find a perfect alternative – Theora, based on the VP3 codec developed by On2 Technologies (*www.on2.com*) and released as open-source software for royalty-free use and modifications (see *www.theora.org/svn.html*).

Theora is an advanced video codec that competes with MPEG-4 and other similar low-bit-rate video compression schemes. It is now supported by the Xiph.org Foundation along with Ogg, the transport layer used with Theora to deliver the video content. The bitstream format is stable enough and supported by multiple players running on different operating systems. Like JPEG and MPEG, it uses a two-dimensional 8 x 8 DCT.

FPGA Implementation

The code for the Elphel Model 333 camera FPGA is written in Verilog HDL (Figure 3). It is designed around the 8-channel SDRAM controller that uses the Spartan-3 DDR capabilities. The structure of the memory accesses and specially organized data mapping both serve the same goal: optimizing memory bandwidth that otherwise would be a system bottleneck.

The rest of the code that currently uses two-thirds of the general FPGA resources (slices) and 20 of 24 block RAM modules includes video compression modules, a sensor, and system interfaces.

A detailed description of the camera code is available, together with the source code, at Sourceforge (*https://sourceforge.net/projects/elphel*).

Conclusion

High-performance reconfigurable FPGAs made it possible to build a fast high-resolution low-bit-rate network camera capable of running 30 fps at a resolution of 1280 x 1024 pixels (12 fps at a resolution of 2048 x 1536). Many of the new features of the Spartan-3 devices proved to be very useful in this design: embedded multipliers for DSP functions, advanced digital clock management, DDR I/O functions, an increased number of global clock networks for the DDR SDRAM controller, and large block RAM modules for the various tables and buffers in the camera.

The free video encoder (Theora) and completely open implementation of the camera (all software and Verilog code is provided under the GNU General Public License) makes the second most important function of Elphel products possible. You can use these cameras not only as finished products but also as universal development platforms – demonstrating the power and flexibility of the Spartan-3 family. It is possible to add your own code, rerun the tools (both for the FPGA code and the C-language camera software), and immediately try the new camera with advanced image processing implemented.

For more information, visit *www.elphel.com*, *https://sourceforge.net/projects/elphel/*, and *www.theora.org*.

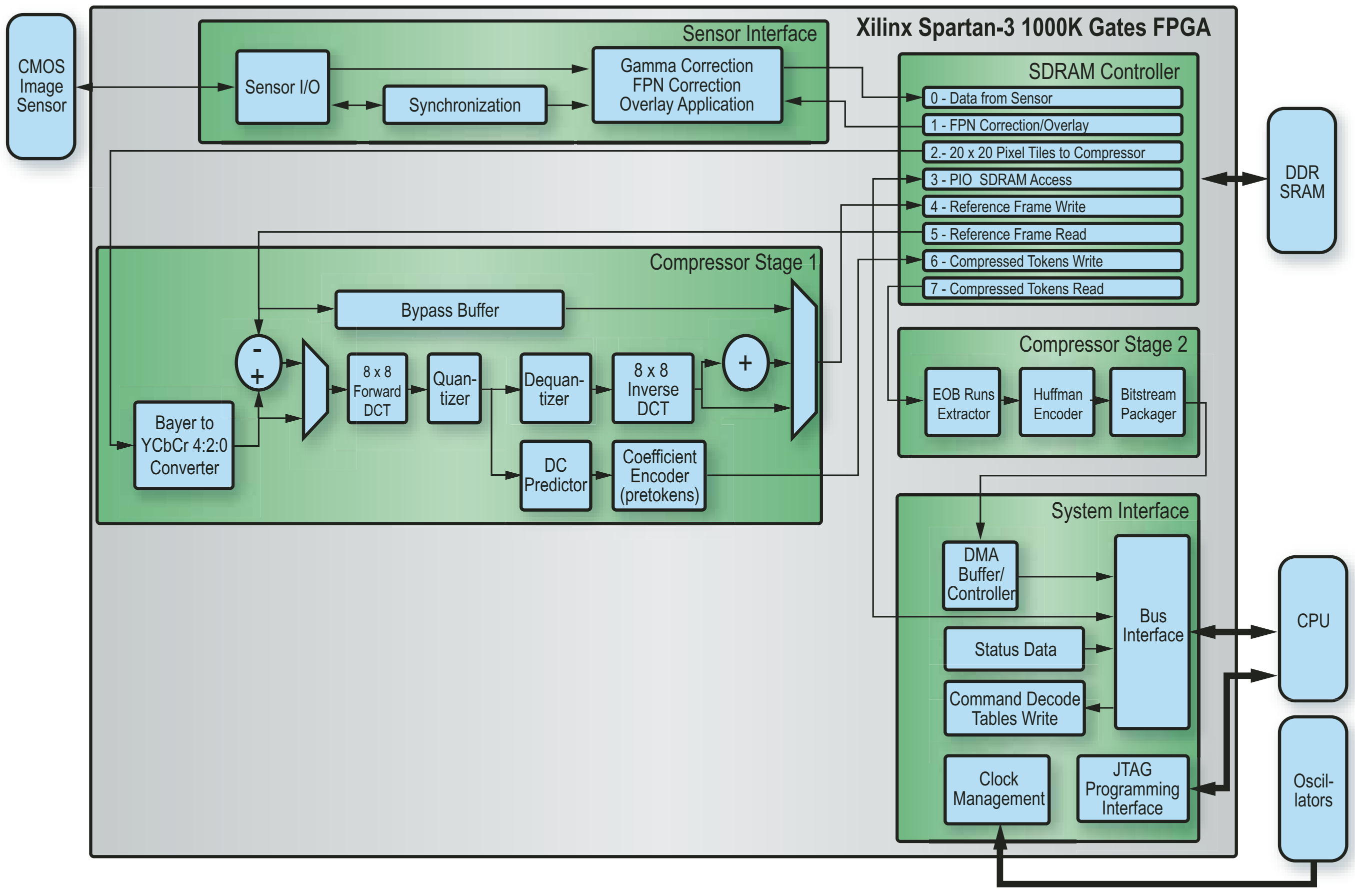
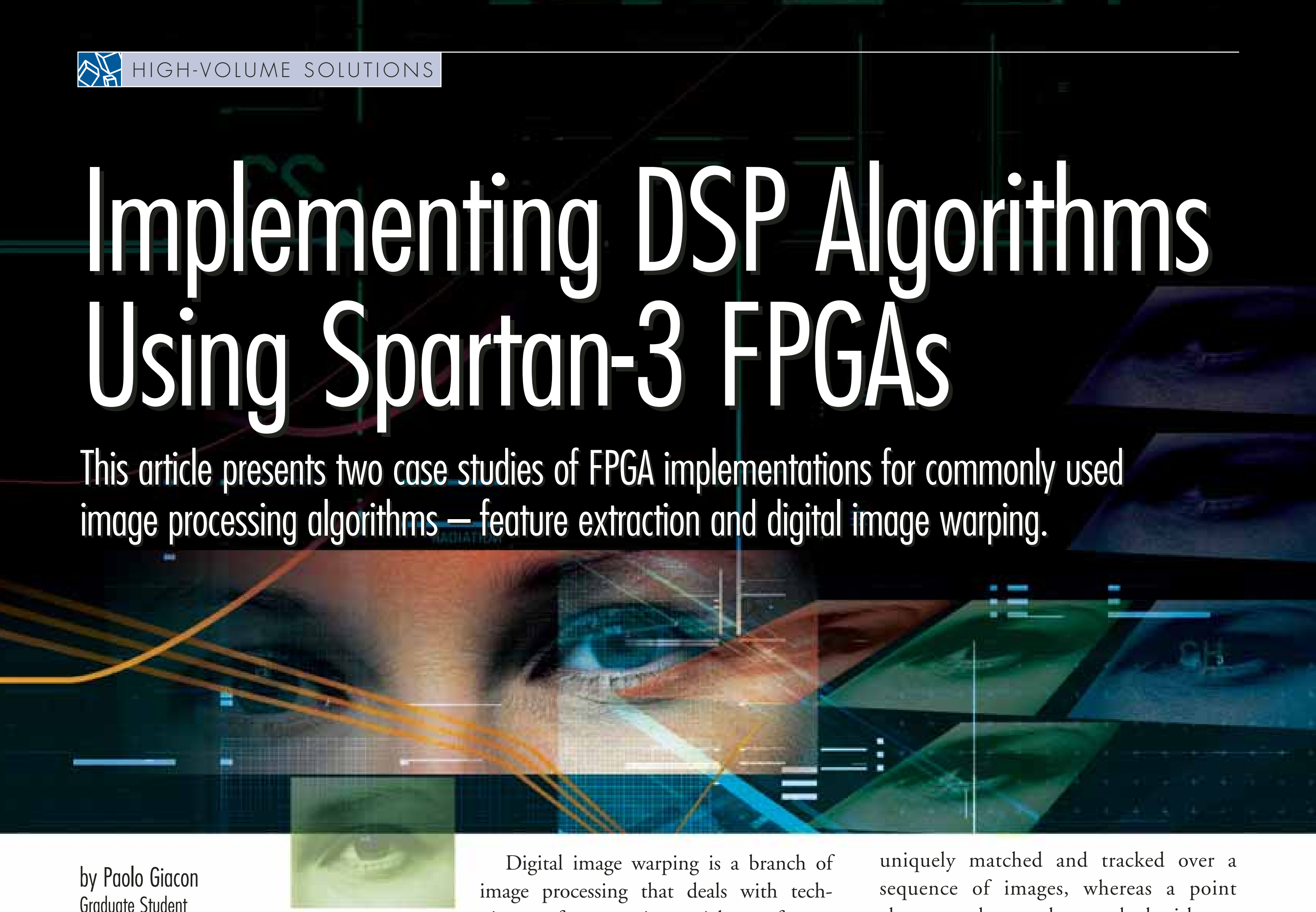


Figure 3 – Block diagram of the FPGA code



Implementing DSP Algorithms Using Spartan-3 FPGAs

This article presents two case studies of FPGA implementations for commonly used image processing algorithms – feature extraction and digital image warping.

by Paolo Giacon
Graduate Student
Università di Verona, Italy
paolo.giacon@students.univr.it

Saul Saggin
Undergraduate Student
Università di Verona, Italy
saul.saggin@students.univr.it

Giovanni Tommasi
Undergraduate Student
Università di Verona, Italy
giovanni.tommasi@students.univr.it

Matteo Busti
Graduate Student
Università di Verona, Italy
matteo.busti@students.univr.it

Computer vision is a branch of artificial intelligence that focuses on equipping computers with the functions typical of human vision. In this discipline, feature tracking is one of the most important pre-processing tasks for several applications, including structure from motion, image registration, and camera motion retrieval. The feature extraction phase is critical because of its computationally intensive nature.

Digital image warping is a branch of image processing that deals with techniques of geometric spatial transformations. Warping images is an important stage in many applications of image analysis, as well as some common applications of computer vision, such as view synthesis, image mosaicing, and video stabilization in a real-time system.

In this article, we'll present an FPGA implementation of these algorithms.

Feature Extraction Theory

In many computer vision tasks we are interested in finding significant feature points – or more exactly, the corners. These points are important because if we measure the displacement between features in a sequence of images seen by the camera, we can recover information both on the structure of the environment and on the motion of the viewer.

Figure 1 shows a set of feature points extracted from an image captured by a camera. Corner points usually show a significant change of the gradient values along the two directions (x and y). These points are of interest because they can be uniquely matched and tracked over a sequence of images, whereas a point along an edge can be matched with any number of other points on the edge in a second image.

The Feature Extraction Algorithm

The algorithm employed to select good features is inspired by Tomasi and Kanade's method, with the Benedetti and Perona approximation, considering the eigenvalues α and β of the image gradient covariance matrix. The gradient covariance matrix is given by:

$$H = \begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_x^2 \end{pmatrix}$$

where I_x and I_y denote the image gradients in the x and y directions.

Hence we can classify the structure around each pixel observing the eigenvalues of H:

No structure	: $\alpha \approx \beta \approx 0$
Edge	: $\alpha \approx 0, \beta >> 0$
Corner	: $\alpha >> 0, \beta >> 0$

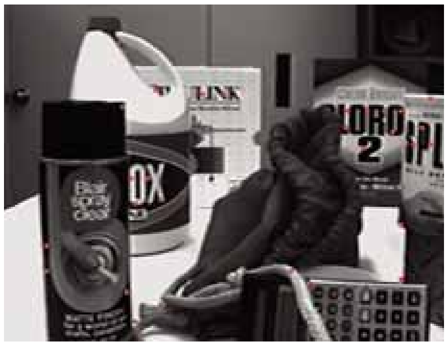


Figure 1 – Feature points extracted from an image captured by a camera

Using the Benedetti and Perona approximation, we can choose the corners without computing the eigenvalues.

We have realized an algorithm that, compared to the original method, doesn't require any floating-point operations. Although this algorithm can be implemented either in hardware or software, by implementing it in FPGA technology we can achieve real-time performance.

Input:

- 8-bit gray-level image of known size (up to 512 x 512 pixels)
- The expected number of feature points (wf)

Output:

- List of selected features (FL). The type of the output is a 3 x N matrix whose:
  - First row contains the degrees of confidence for each feature in the list
  - Second row contains the x-coordinates of the feature points
  - Third row contains the y-coordinates of the feature points

Semantic of the Algorithm

In order to determine if a pixel (i, j) is a feature point (corner), we followed Tomasi and Kanade's method.

First, we calculate the gradient of the image. Hence the 2 x 2 symmetric matrix G = [a b; b c] is computed, whose entries derive from the gradient values in a patch around the pixel (i, j).

If the minimum eigenvalue of G is greater than a threshold, then the pixel (i, j) is a corner point. The minimum eigenvalue is computed using an approximation to avoid the square root operation that is expensive for hardware implementations.

The corner detection algorithm could be summarized as follows:

The image gradient is computed by mean of convolution of the input image with a predefined mask. The size and the values of this mask depend on the image resolution. A typical size of the mask is 7 x 7.

- For each pixel (i, j) loop:

$$a_{i,j} = \sum_{k}^{N} (I_x^k)^2$$

$$b_{i,j} = \sum_{k}^{N} I_x^k I_y^k$$

$$c_{i,j} = \sum_{k}^{N} (I_y^k)^2$$

where N is the number of pixels in the patch and I_x^k and I_y^k are the components of the gradient at pixel k inside the patch.

- $P_{i,j} = (a - t)(c - t) - b^2$

where t is a fixed integer parameter.

- If $(P_{i,j} > 0)$ and $(a_{i,j} > t)$, then we retain pixels (i,j)
- Discard any pixel that is not a local maximum of $P_{i,j}$
- End loop
- Sort, in decreasing order, the feature list *FL* based on the degree of confidence values and take only the first *wf* items.

Implementation

With its high-speed embedded multipliers, the Xilinx® Spartan™-3 architecture meets the cost/performance characteristics required by many computer vision systems that could take advantage of this algorithm.

The implementation is divided into four fundamental tasks:

1. Data acquisition. Take in two gradient values along the x and y axis and compute for each pixel three coefficients used by the characteristic polynomial. To store and read the gradient values, we use a buffer (implemented using a Spartan-3 block RAM).
2. Calculation of the characteristic polynomial value. This value is important to sort the features related to the specific pixel. We implemented the multiplications used for the characteristic polynomial calculus employing the embedded multipliers on Spartan-3 devices.
3. Feature sorting. We store computed feature values in block RAM and sort them step by step by using successive comparisons.
4. Enforce minimum distance. This is done to keep a minimum distance between features; otherwise we get clusters of features heaped around the most important ones. This is implemented using block RAMs, building a non-detect area around each most important feature where other features will not be selected.

Spartan-3 Theoretical Performance

The algorithm is developed for gray-level images at different resolutions, up to 512 x 512 at 100 frames per second.

The resources estimated by Xilinx System Generator are:

- 1,576 slices
- 15 block RAMs
- 224 LUTs
- 11 embedded multipliers

The embedded multipliers and extensive memory resources of the Spartan-3 fabric allow for an efficient logic implementation.

Applications of Feature Extraction

Feature extraction is used in the front end for any system employed to solve practical control problems, such as autonomous navigation and systems that could rely on vision to make decisions and provide control. Typical applications include active video surveillance, robotic arms motion, measurement of points and distances, and autonomous guided vehicles.

Image Warping Theory

Digital image warping deals with techniques of geometric spatial transformations.

The pixels in an image are spatially represented by a couple of Cartesian coordinates (x, y). To apply a geometric spatial transformation to the image, it is convenient to switch to homogeneous coordinates, which allow us to express the transformation by a single matrix operation. Usually this is done by adding a third coordinate with value 1 (x, y, 1).

In general, such transformation is represented by a non-singular 3 x 3 matrix H and applied through a matrix-vector multiplication to the pixel homogeneous coordinates:

$$\begin{bmatrix} H_{1,1} & H_{1,2} & H_{1,3} \\ H_{2,1} & H_{2,2} & H_{2,3} \\ H_{3,1} & H_{3,2} & H_{3,3} \end{bmatrix} \bullet \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} H_{1,1}x + H_{1,2}y + H_{1,3} \\ H_{2,1}x + H_{2,2}y + H_{2,3} \\ H_{3,1}x + H_{3,2}y + H_{3,3} \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} x'/w' \\ y'/w' \\ 1 \end{bmatrix} \Rightarrow (x'/w', y'/w') \quad (1)$$

The matrix H, called homography or collineation, is defined up to a scale factor (it has 8 degrees of freedom). The transformation is linear in projective (or homogeneous) coordinates, but non-linear in Cartesian coordinates.

The formula implies that to obtain Cartesian coordinates of the resulting pixel we have to perform a division, an operation quite onerous in terms of time and area consumption on an FPGA. For this reason, we considered a class of spatial transformations called “affine transformations” that is a particular specialization of homography. This allows us to avoid the division and obtain good observational results:

$$\begin{bmatrix} A_{1,1} & A_{1,2} & A_{1,3} \\ A_{2,1} & A_{2,2} & A_{2,3} \\ 0 & 0 & 1 \end{bmatrix} \bullet \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} A_{1,1}x + A_{1,2}y + A_{1,3} \\ A_{2,1}x + A_{2,2}y + A_{2,3} \\ 1 \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \Rightarrow (x', y') \quad (2)$$

Affine transformations include several planar transformation classes as rotation, translation, scaling, and all possible combinations of these. We can summarize the affine transformation as every planar transformation where the parallelism is preserved. Six parameters are required to define an affine transformation.

Image Warping Algorithms

There are two common ways to warp an image:

- Forward mapping
- Backward mapping

Using forward mapping, the source image is scanned line by line and the pixels are copied to the resulting image, in the position given by the result of the linear system shown in equation (2). This technique is subject to several problems, the most important being the presence of holes in the final image in the case of significant modification of the image (such as rotation or a scaling by a factor greater than 1) (Figure 2).

The backward mapping approach gives better results. Using the inverse transformation A^{-1}, we scan the final image pixel by pixel and transform the coordinates. The result is a pair of non-integer coordinates in the source image. Using a bilinear interpolation of the four pixel values identified in the source image, we can find a value for the final image pixel (see Figure 3).

This technique avoids the problem of holes in the final image, so we adopted it as our solution for the hardware implementation.

Implementation

Software implementations of this algorithm are well-known and widely used in applications where a personal computer or workstation is required. A hardware implementation requires further work to achieve efficiency constraints on an FPGA.

Essentially, the process can be divided in two parts: transformation and interpolation. We implemented the first as a matrix-vector multiplication (2), with four multipliers and four adders. The second is an approximation of the real result of the interpolation: we weighted the four pixel values approximating the results of the transformation with two bits after the binary point. Instead of performing the calculations given by the formula, we used a LUT to obtain the pixel final value, since we divided possible results of the interpolation into a set of discrete values.

Spartan-3 Theoretical Performance

We designed the algorithm using System Generator for DSP, targeting a Spartan-3 device. We generated the HDL code and synthesized it with ISE™ design software, obtaining a resource utilization of:

- 744 slices (1,107 LUTs)
- 164 SRL16
- 4 embedded multipliers

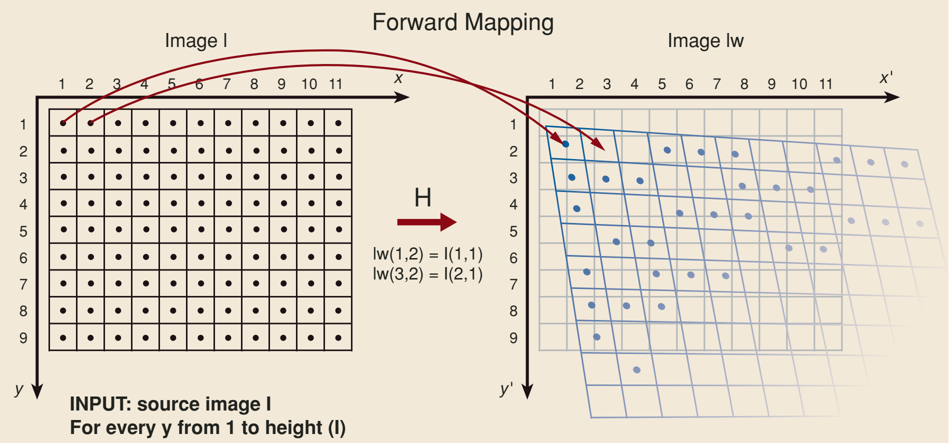
The design can process up to 46 fps (frames per second) with 512 x 512 images. Theoretical results show a boundary of 360+ fps in a Spartan-3-based system.

Applications of Image Warping

Image warping is typically used in many common computer vision applications, such as view synthesis, video stabilization, and image mosaicing.

Image mosaicing deals with the composition of sequence (or collection) of images after aligning all of them respective to a common reference frame. These geometrical transformations can be seen as simple relations between coordinate systems.

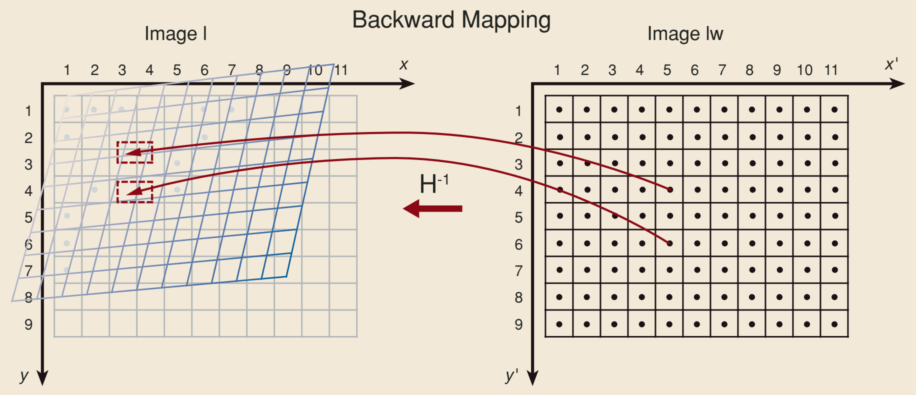
By applying the appropriate transformations through a warping operation and merging the overlapping regions of a warped image, we can construct a single panoramic image covering the entire visible area of the scene. Image mosaicing provides a powerful way to create detailed three-dimensional models and scenes for virtual reality scenarios based on real imagery. It is employed in flight simulators, interactive multi-player games, and medical image systems to construct true scenic panoramas or limited virtual environments.



```
INPUT: source image I
For every y from 1 to height (I)
   For every x from 1 to width(I)
      Calculate x', u = round(x')
      Calculate y', v = round(y')
      If 1<= u <= wodth(Iw) and 1<= v <= height(Iw)
          Copy I(x,y) to Iw(u,v)

OUTPUT: warped image Iw
```

Figure 2 – Forward mapping with a scaling factor greater than one



Iw(4,5) = interpolation of { I(3,2), I(4,2), I(3,3), I(4,3)}
Iw(4,7) = interpolation of { I(3,4), I(4,4), I(3,5), I(4,5)}

```
INPUT: source image I
For every v from 1 to height (Iw)
   For every u from 1 to width(Iw)
      Calculate x
      Calculate y
      Calculate Iw(u,v)as bilinear interpolation the four pixel values:

         $I(\lfloor x \rfloor, \lfloor y \rfloor), I(\lfloor x \rfloor, \lceil y \rceil), I(\lceil x \rceil, \lceil y \rceil), I(\lceil x \rceil, \lfloor y \rfloor)$ such as
         $Iw(u,v) = (1-h)\ (1-k) \bullet p00 + h \bullet (1-k) \bullet p10 + (1-h) \bullet k \bullet p01 + h \bullet k \bullet p11$
         where: $\begin{cases} h = x - \lfloor x \rfloor, k = y - \lfloor y \rfloor \\ p00 = I(\lfloor x \rfloor, \lfloor y \rfloor), p10 = I(\lceil x \rceil, \lfloor y \rfloor), p01 = I(\lfloor x \rfloor, \lceil y \rceil), p11 = I(\lceil x \rceil, \lceil y \rceil) \end{cases}$

OUTPUT: warped image Iw
```

Figure 3 – Backward mapping with a scaling factor greater than one

Conclusion

The challenge is to design efficient, effective, and reliable vision modules with the highest possible reliability.

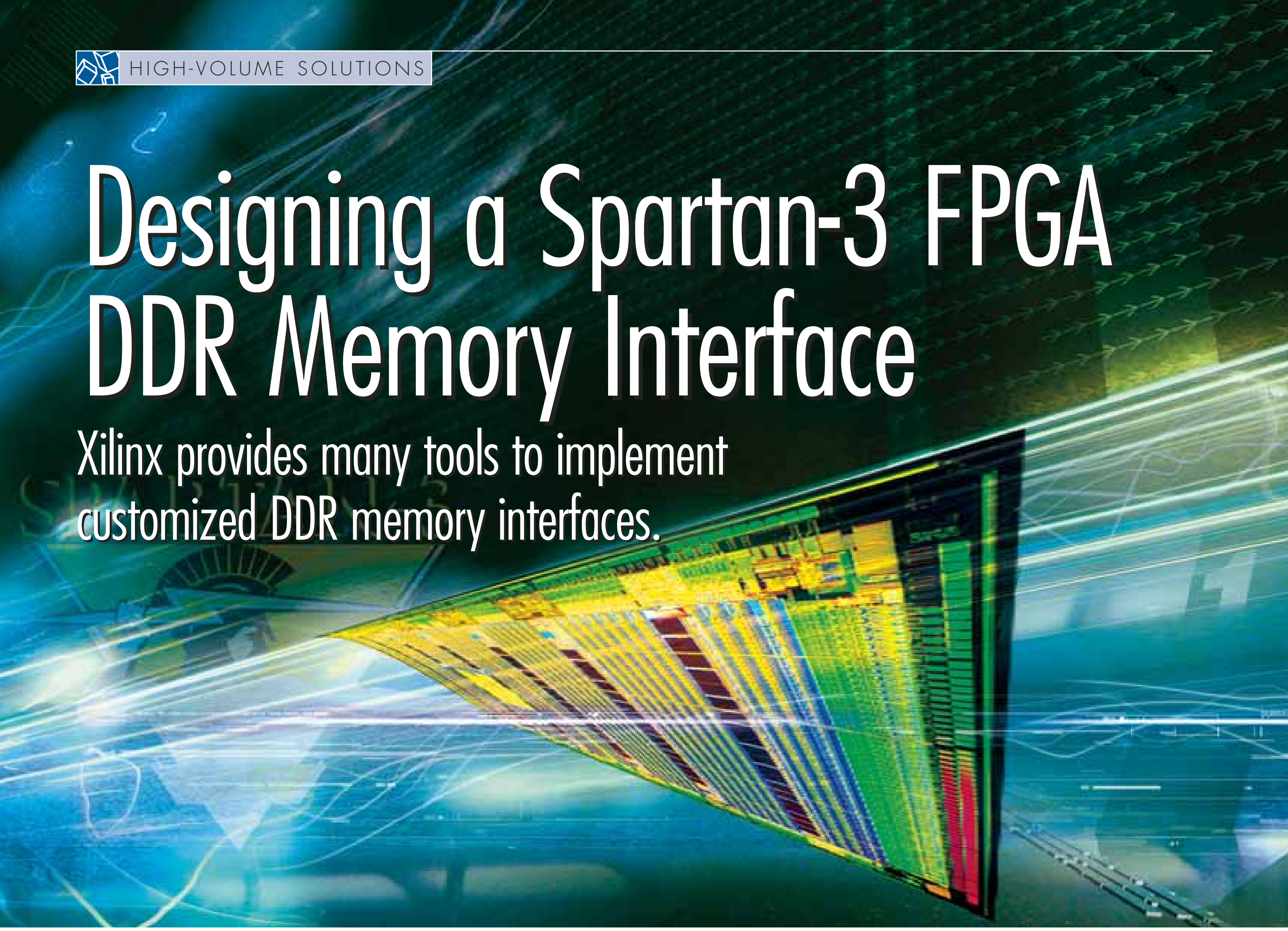
Ultimodule, a Xilinx XPERTS partner, and the VIPS Laboratory at the Università di Verona have defined a foundation platform for computer vision using Ultimodule's system-on-module family. The platform provides a stereovision system for real-time extraction of three-dimensional data and a real-time image-processing engine implementing most of the algorithms required when an application relies on vision to make decisions and provide control.

The platform supports applications that require high performance and robust vision analysis, both in qualitative and computational terms (real-time), including active video surveillance, robotic arm motion and control, autonomous vehicle navigation, test and measurement, and hazard detection. The platform provides modules with all required system control logic, memory, and processing hardware, together with the application software. Interconnecting modules allow fast development of a complex architecture.

The platform leverages Xilinx Spartan-3 devices, which are an optimal choice for image processing IP cores because of their flexibility, high performance, and DSP-oriented targeting. The Spartan-3 family provides a valid, programmable alternative to ASICs. This characteristic, coupled with its low cost structure, adds considerable value when time to market is crucial.

For more information about feature extraction, you can e-mail the authors at *paolo.giacon@students.univr.it* or *saul.saggin@students.univr.it.* For more information about image warping, you can e-mail *matteo.busti@students.univr.it* or *giovanni.tommasi@students.univr.it.*

We are grateful for the support from our advisor, Professor Murino, in the Vision, Image Processing, and Sound (VIPS) Laboratory in the Dipartimento di Informatica at the Università di Verona, and contributions from Marco Monguzzi, Roberto Marzotto, and Alessandro Negrente.



Designing a Spartan-3 FPGA DDR Memory Interface

Xilinx provides many tools to implement customized DDR memory interfaces.

by Rufino Olay
Marketing Manager, Spartan Solutions
Xilinx, Inc.
rufino.olay@xilinx.com

Karthikeyan Palanisamy
Staff Engineer, Memory Applications Group
Xilinx, Inc.
karthi.palanisamy@xilinx.com

Memory speed is a crucial component of system performance. Currently, the most common form of memory used is synchronous dynamic random access memory (SDRAM).

The late 1990s saw major jumps in SDRAM memory speeds and technology because systems required faster performance and larger data storage capabilities. By 2002, double-data-rate (DDR) SDRAM became the standard to meet this ever-growing demand, with DDR266 (initially), DDR333, and recently DDR400 speeds.

DDR SDRAM is an evolutionary extension of "single-data-rate" SDRAM and provides the benefits of higher speed, reduced power, and higher density components. Data is clocked into or out of the device on both the rising and falling edges of the clock. Control signals, however, still change only on the rising clock edge.

DDR memory is used in a wide range of systems and platforms and is the computing memory of choice. You can use Xilinx® Spartan™-3 devices to implement a custom DDR memory controller on your board.

Interfacing Spartan-3 Devices with DDR SDRAMs

Spartan-3 platform FPGAs offer an ideal connectivity solution for low-cost systems, providing the system-level building blocks necessary to successfully interface to the latest generation of DDR memories.

Included in all Spartan-3 FPGA input/output blocks (IOB) are three pairs of storage elements. The storage-element pair on either the output path or the three-state path can be used together with a special multiplexer to produce DDR transmission. This is accomplished by taking data synchronized to the clock signal's rising edge and converting it to bits synchronized on both the rising and falling edge. The combination of two registers and a multiplexer is referred to as double-data-rate D-type flip-flop (FDDR).

Memory Controllers Made Fast and Easy

Xilinx has created many tools to get designers quickly through the process of building and testing memory controllers for Spartan devices. These tools include reference designs and application notes, the Memory Interface Generator (MIG), and more recently, a hardware test platform.

Xilinx application note XAPP454, "DDR2 SDRAM Memory Interface for Spartan-3 FPGAs," describes the use of a Spartan-3 FPGA as a memory controller,

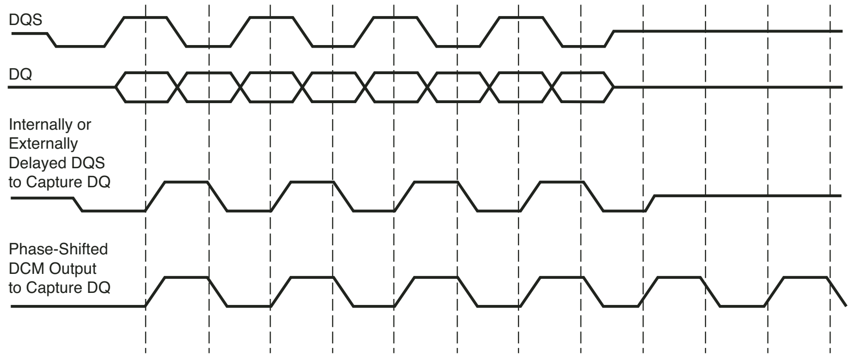

DQS
DQ
Internally or Externally Delayed DQS to Capture DQ
Phase-Shifted DCM Output to Capture DQ

Figure 1 – Read operation timing diagram

with particular focus on interfacing to a Micron MT46v32M16TG-6T DDR SDRAM. This and other application notes illustrate the theory of operations, key challenges, and implementations of a Spartan-3 FPGA-based memory controller.

DDR memories use non-free-running strobes and edge-aligned read data (Figure 1). For 333 Mbps data speeds, the memory strobe must be used for higher margins. Using local clocking resources, a delayed strobe can be centered in the data window for data capture.

To maximize resources within the FPGA, you can explore design techniques such as using the LUTs as RAMs for data capture – while at the same time minimizing the use of global clock buffers (BUFGs) and digital clock managers (DCMs) – as explained in the Xilinx application notes. Results are given with respect to the maximum data width per FPGA side for either right and left or top and bottom implementations. Implementation challenges such as these are mitigated with the new Memory Interface Generator.

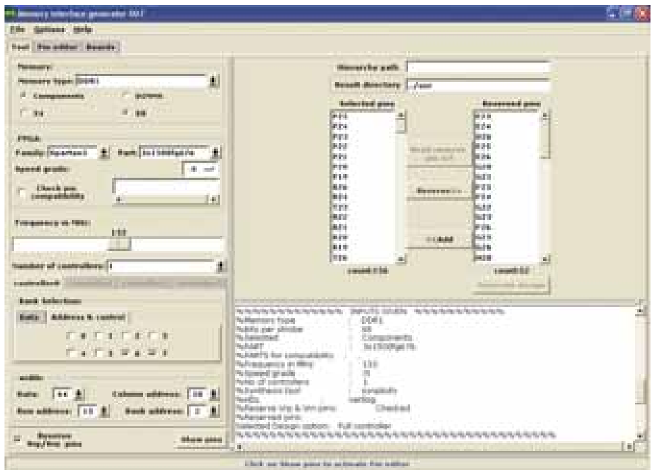


Figure 2 – Using the MIG 007 to automatically create a DDR memory controller

Feature	Utilization	Percent Used
Number of Slices	2,277 out of 13,312	17%
Number of DCMs	1 out of 4	25%
Number of External IOBs	147 out of 487	30%

Table 1 – Device utilization for a DDR 64-bit interface in an XC3S1500 FPGA

Xilinx created the Memory Interface Generator (MIG 007) to take the guesswork out of designing your own controller. To create the interface, the tool requires you to input data including FPGA device, frequency, data width, and banks to use. The interactive GUI (Figure 2) generates the RTL, EDIF, SDC, UCF, and related document files.

As an example, we created a DDR 64-bit interface for a Spartan XC3S1500-5FG676 using MIG. The results in Table 1 show that the implementation would use 17% of the slices, leaving more than 80% of the device free for data-processing functions.

Testing Out Your Designs

The last sequence in a design is the verification and debug in actual hardware. After using MIG 007 to create your customized memory controller, you can implement your design on the Spartan-3 Memory Development Kit, HW-S3-SL361, as shown in Figure 3. The $995 kit is based on a Spartan-3 1.5M-gate FPGA (the XC3S1500) and includes additional features such as:

- 64 MB of DDR SDRAM Micron MT5VDDT1672HG-335, with an additional 128 MB DDR SDRAM DIMM for future expansion
- Two-line LCD
- 166 MHz oscillator
- Rotary switches
- Universal power supply 85V-240V, 50-60 MHz

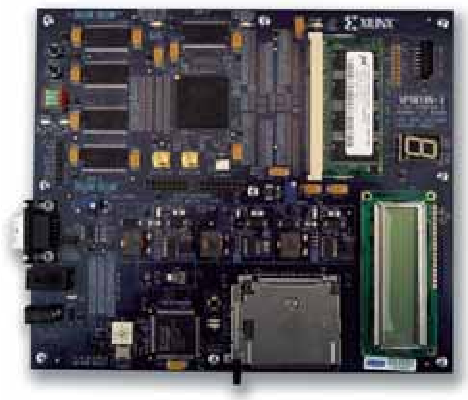
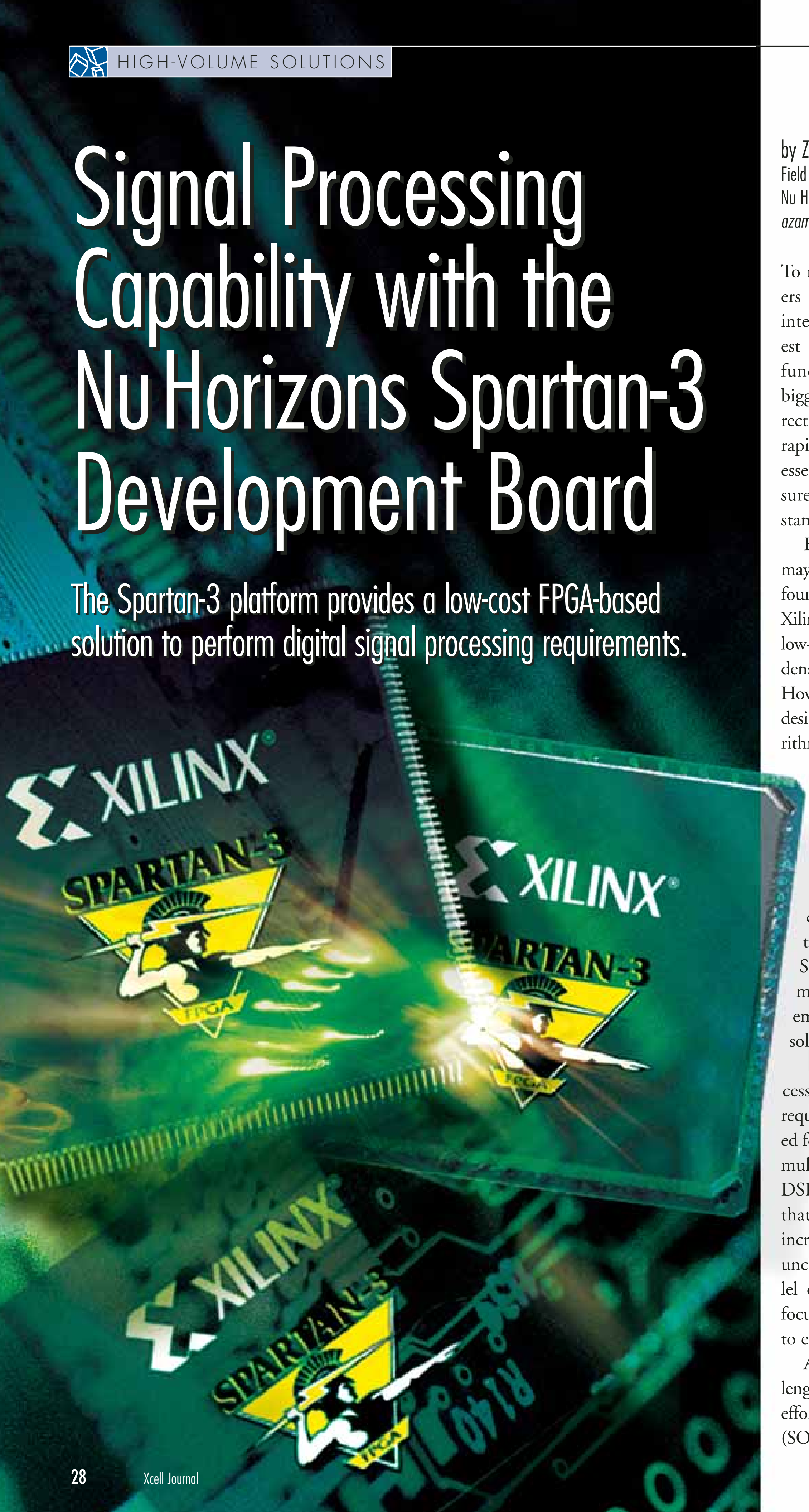


Figure 3 – Spartan-3 memory development board (HW-S3-SL361)

Conclusion

With the popularity of DDR memory increasing in system designs, it is only natural that designers use Spartan-3 FPGAs as memory controllers. Implementing the controller need not be difficult.

For more information about the application notes, GUI, and development board, please visit *www.xilinx.com/products/design_resources/mem_corner/index.htm.*



Signal Processing Capability with the Nu Horizons Spartan-3 Development Board

The Spartan-3 platform provides a low-cost FPGA-based solution to perform digital signal processing requirements.

by Zulfiqar Ali Zamindar
Field Application Engineer
Nu Horizons Electronics Corp.
azamindar@nuhorizons.com

To meet their system design goals, designers today must prototype a new idea and integrate its product features into the lowest cost silicon, complete with versatile functionality. Specifically, two of the biggest challenges are to get the design correct in the first place and to fix a problem rapidly. Immediate design solutions are essential for meeting time-to-market pressures, keeping up with changing industry standards, and prototyping quickly.

Even after the design is completed, a need may exist for field upgradeability if a bug is found or a new functionality is available. The Xilinx® Spartan™ FPGA has been a great low-cost programmable platform for low-density control logic and system interfaces. However, when it comes to signal processing, designers traditionally purchase a fixed algorithm standard product for high-speed multiply and accumulate (MAC) functions. This requirement demands additional design resources, verification time, system components, and more board space.

With the explosion of the wireless communication market and proliferation of low-cost 90 nm processes, Spartan-3 devices – with plenty of logic, memory, and as many as 104 18 x 18 embedded hard multipliers – are the ideal solution for many signal processing needs.

The challenge for today's digital processing systems is their large memory requirements and the very fast MACs needed for rapid mathematical operations. Every multimedia system contains an external DSP processor and memory component that reduces system performance and increases component costs. Once you have uncovered a solution that integrates a parallel or semi-parallel system, you can then focus on improving the overall DSP design to eliminate performance bottlenecks.

As a result of digital processing challenges, many companies have focused their efforts on developing the system-on-chip (SOC) concept by adding feature sets to bring additional functionality to a single piece of silicon. Customized ASICs have become very costly solutions in today's competitive landscape. Traditional DSP processors are capable of carrying out high-speed MAC operations, but have bandwidth limitations. FPGA technology has made tremendous progress in recent years by increasing a large number of intellectual properties to reduce the cost of silicon development in various markets. This is accomplished by optimizing architectures, using leading process technologies, and adding IP cores.

Some of the typical applications for digital signal processing are digital cameras, phones, 3G wireless, video conferencing systems, and high-definition digital televisions. Having a signal processing capability inside an FPGA is the perfect design innovation – the stepping stone to system-on-chip in an FPGA without the high cost of complex customized chip development.

System Generator for DSP

Xilinx expanded its features in the Spartan-3 FPGA by adding embedded multipliers in the architecture. This technological innovation is similar to embedded block memory, clock management, and multiple standards for high-speed I/O circuits, all standard characteristics of the Xilinx Spartan-3 and Virtex™-II Pro families.

Time to market remains critical for companies developing both system hardware and software. With Xilinx System Generator (SysGen), you can simultaneously create behavioral-level hardware blocks and simulate the entire system with just a few tool clicks. The design environment allows you to create block-based systems like digital QAM modulators for software-defined radio, finite impulse response (FIR) filters, image processing functions, mathematical operators, A/D and delta-sigma D/A conversion, and all-in-one silicon for widely used applications.

Using System Generator within the Simulink design environment from The MathWorks, you have unrestricted access to many blocks. You can select both Xilinx and third-party blocks, drag and drop to the Simulink work space, connect, and simulate a system within minutes. The Xilinx System Generator block is used to select various implementation options such as FPGA device, package, speed, system clock, synthesis options, and HDL. The need to write HDL is eliminated, as the tool creates the proper language for you; however, it can write HDL if you prefer.

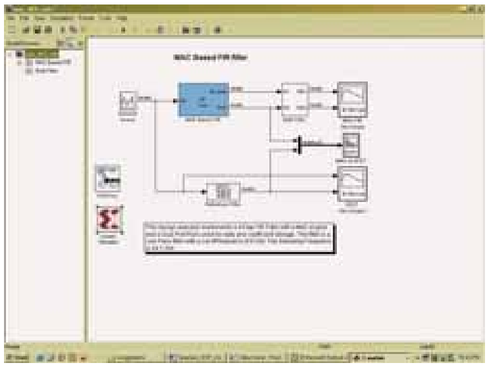


Figure 1 – MAC FIR filter block diagram in System Generator

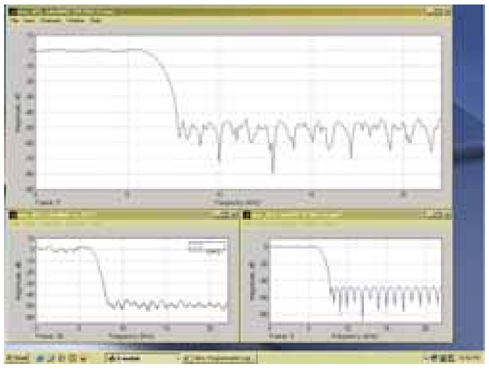
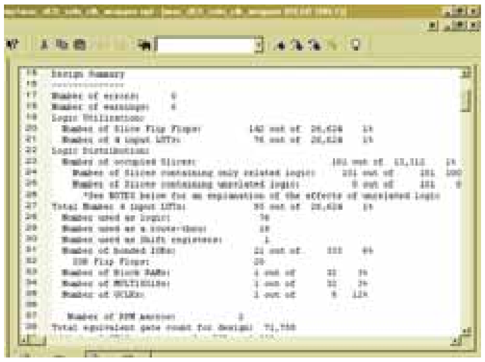


Figure 2 – Simulation view of FIR filter



```
Design Summary
--------------
Number of errors:      0
Number of warnings:    4
Logic Utilization:
  Number of Slice Flip Flops:        142 out of  26,624    1%
  Number of 4 input LUTs:             76 out of  26,624    1%
Logic Distribution:
  Number of occupied Slices:                          101 out of  13,312    1%
    Number of Slices containing only related logic:     101 out of     101  100%
    Number of Slices containing unrelated logic:          0 out of     101    0%
      *See NOTES below for an explanation of the effects of unrelated logic
Total Number 4 input LUTs:            95 out of  26,624    1%
  Number used as logic:               76
  Number used as a route-thru:        18
  Number used as Shift registers:      1
  Number of bonded IOBs:              22 out of     333    6%
    IOB Flip Flops:                   20
  Number of Block RAMs:                1 out of      32    3%
  Number of MULT18X18s:                1 out of      32    3%
  Number of GCLKs:                     1 out of       8   12%

   Number of RPM macros:           1
Total equivalent gate count for design:  71,755
```

Figure 3 – Implementation summary

Simulink also enables you to integrate blocks from many different libraries. Commonly used DSP block sets include math functions, signal management, a variety of filters, transforms, encoders, decoders, and linear feedback shift registers. For example, you can create a fast Fourier transform (FFT) or FIR core with the easy-to-use GUI in System Generator for DSP, customize the core as per your application, and run the Xilinx ISE™ tool in the background to build your signal processor system. This flow can synthesize, place, route, and generate hardware configuration files.

The Simulink environment allows you to verify the functionality of each block or subsystem created with scopes and graphs to view images or observe data.

Digital filters are among the most significant components in digital signal processing applications. The function of a filter is to eliminate undesirable parts of the signal (random noise) or to extract signals in a particular frequency range. Basic FIR filters are used extensively in video broadcasting and wireless communications. A mathematical expression of a basic FIR filter is:

$$Y(n) = \text{SUM } h(k) * x(n-k); \quad k=0 \text{ to } k=N-1$$

It consists of an input sample, output sample, and coefficients. Imagine "x" is a continuous stream of input signal and "y" is a resulting filtered stream of output signal. The "n" and "k" in the equation correspond to a particular instant in time, so to compute "y (n)" at time "n," a group of input samples at "n" different points in time are required, or numerically x (n), x (n-1), x (n-2) ... x (n-k). A group of "n" input samples are multiplied by "n" different coefficients and summed together to form a result "y (n)."

This design example implements a 43-tap FIR filter with a MAC engine and a dual-port RAM used for data and coefficient storage. The filter is a low-pass filter with a cut-off frequency of 6 KHz. The sampling frequency is 44.1 KHz.

Figure 1 represents the model of the filter. The model has a coefficient width of 12, a coefficient binary point value of 12, a data width value of 10, a data binary point value of 8, and a sampling frequency of 44.1 KHz. Scope shots of the filtered output are shown in Figure 2. An implementation summary displaying the use of RAM and multiplier resources is shown in Figure 3. This design achieved ~ 125 MHz performance in a -4 speed grade of the Spartan-3 device.

The NuHorizons Spartan-3 Board

Xilinx, its distributors, and third-party companies offer several boards for prototyping or emulating a DSP-based system. A low-cost prototyping platform from Nu Horizons Electronics Corp. is the Spartan-3 development board (HW-AFX-SP3-2000-DB) (Figure 4). The board comprises these elements (Figure 5):

- Xilinx XC3S2000-4FG676 Spartan-3 device
- XCF08 Flash PROM for configuration
- 4 x 24-character LCD display
- Graphical LCD interface
- 64 Mb SDRAM (2-1 Mb x 16 x 4)
- 32 Mb Flash 2 Mb x 16
- 50 MHz clock oscillator
- PLL clock multiplier
- CAN 2.0B transceiver
- RS232 interface
- PS2 interface
- Audio CODEC
- Two-channel A/D and D/A converter
- 10/100 Ethernet MAC
- 10/100 Ethernet PHY
- Flash memory interface
- SDRAM memory interface
- LED/push buttons
- JTAG configuration header for programming
- 16-bit LVDS I/F with clock and control
- Test point headers for debugging

This fully loaded Spartan-3 development board is priced at $449, which includes all of the features necessary to prototype a DSP-based system design. The board comes with a user's manual, power supply, documents, and design files. The option of getting the board bundled with ISE Foundation™ software, System Generator, and ChipScope™ Pro software is also available at an additional cost.

Besides DSP designs, the Spartan-3 platform is a great tool to implement many other Xilinx reference designs. Several designs written by Nu Horizons' field application engineers cover topics such as memory controllers, embedded processors, hardware-in-the-loop with digital signal processing, and system monitor design using ADC and DAC on the board. ADC and DAC are very powerful attributes of our low-cost board, and two of the many competitive board features. The Spartan-3 platform can be expanded with the add-on ADC board from Linear Technology.

Figure 4 – Spartan-3 board

Conclusion

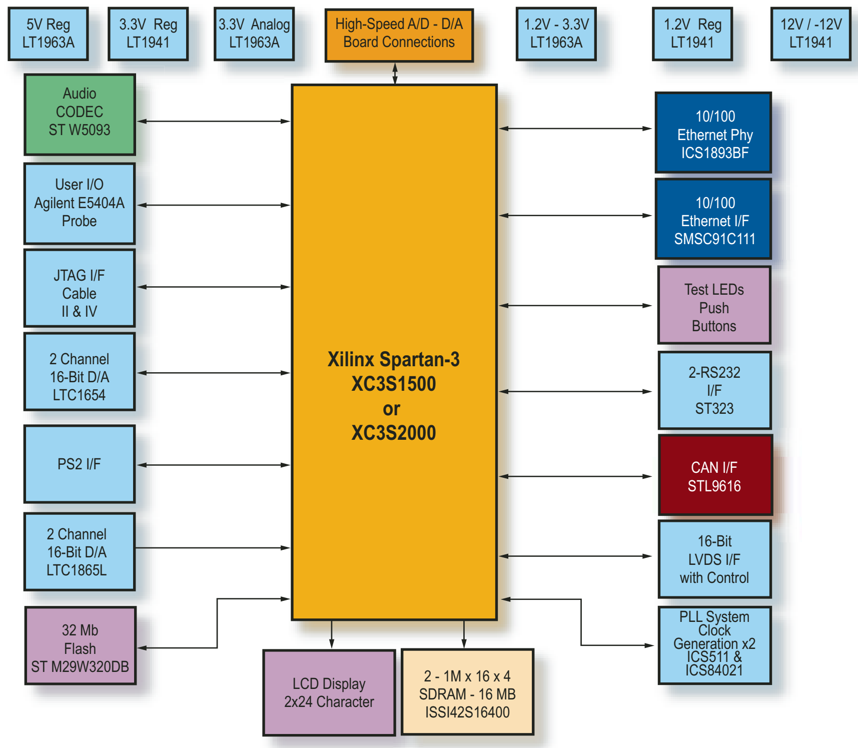
With fast multipliers and lower cost FPGAs, engineers now have the ideal solution to their signal and image processing requirements. With tools like System Generator, anyone can implement a powerful parallel or semi-parallel customized DSP system-on-chip design within days.

The Spartan-3 board from Nu Horizons is a perfect solution for prototyping signal processing using Spartan-3 FPGAs. The board has all of the interfaces necessary to create designs for wireless and digital imaging applications if you want to:

- Integrate your logic and signal processing capability on a single chip
- Prototype a configurable DSP system-on-chip
- Reduce the cost of conventional serially implemented external signal processors
- Improve system performance

Nu Horizons is also in the process of releasing a Virtex-4 platform board for customers requiring higher density logic, more memory, and hard MACs running up to 500 MSPS for high-performance interfaces to video and imaging applications.

For all of the designs and related documentation for the Spartan-3 board, visit the Nu Horizons website at *www.nuhorizons.com/sp3/*.


5V Reg LT1963A
3.3V Reg LT1941
3.3V Analog LT1963A
High-Speed A/D - D/A Board Connections
1.2V - 3.3V LT1963A
1.2V Reg LT1941
12V / -12V LT1941
Audio CODEC ST W5093
User I/O Agilent E5404A Probe
JTAG I/F Cable II & IV
2 Channel 16-Bit D/A LTC1654
PS2 I/F
2 Channel 16-Bit D/A LTC1865L
32 Mb Flash ST M29W320DB
Xilinx Spartan-3 XC3S1500 or XC3S2000
LCD Display 2x24 Character
2 - 1M x 16 x 4 SDRAM - 16 MB ISSI42S16400
10/100 Ethernet Phy ICS1893BF
10/100 Ethernet I/F SMSC91C111
Test LEDs Push Buttons
2-RS232 I/F ST323
CAN I/F STL9616
16-Bit LVDS I/F with Control
PLL System Clock Generation x2 ICS511 & ICS84021
Figure 5 – Spartan-3 XC3S1500 /2000 board block diagram

CoolRunner-II CPLDs Offer New Features

New updates to CoolRunner-II 32- and 64-macrocell CPLDs are now available.

by Steve Prokosch
High-Volume Product Solutions Marketing Manager
Xilinx, Inc.
steve.prokosch@xilinx.com

Since joining the CPLD market in 1992, Xilinx® has gone from being the new kid on the block to the second largest CPLD supplier, and is closing the gap on the number-one position in both volume and sales dollars. Growth has come from listening to customer input and creating innovative ideas to satisfy customer needs.

Continued improvement is an ongoing goal. Toward this end, Xilinx has extended the features of its 32- and 64-macrocell CPLD devices by augmenting a feature, currently only available on higher density parts, that will help ease design difficulties and reduce total system costs. In conjunction with added features, Xilinx is also offering new packages to decrease the cost-per-I/O for small footprint packages. These new packages will help Xilinx continue to penetrate low-cost, battery-operated devices.

Xilinx is achieving CPLD market share growth through both traditional applications such as computing, data processing, networking, and telecommunications, and non-traditional applications such as consumer products (set top boxes and large screen TVs – both plasma and LCD) and key handheld markets (PDAs, handset, and other battery operated products). With new market success and continued traditional market use, CPLDs will continue to show growth in system logic solutions.

New Banking for CoolRunner-II CPLDs

Xilinx has re-introduced its CoolRunner-II™ 32- and 64-macrocell devices with an added feature known as I/O banking. I/O banking is useful in systems that use different supply voltage levels on the same printed circuit board. Usually, a mismatch of voltage levels occurs between a system processor and the devices with which it communicates. This communication may be as simple as a serial to parallel conversion, or as integral as a processor interface to a display.

Another example is the connection of a processor to an external memory card such as CF (compact flash) or SD (secure digital). Because of competition and constant price pressures, high-profile devices such as processors must continue to use advanced process technologies. These process technologies continue to lower voltage swings because of the physical properties of wafer geometries.

Yet standards organizations in markets like memory cards, wireless communications, and bus architectures go through specification changes more slowly. To bridge the gap, voltage and I/O translation is a necessary component in today's architectures. Care must be taken to match input and output voltage-switching thresholds for that particular voltage standard.

Designers face the challenge of pasting together many logic level standards on a variety of part types. What device can do the job and yet leave room for design upgrades or changes? Discrete devices can usually address this mismatch, but may add cost, power consumption, printed circuit board layers, and area to a design. They may have limited functionality – and you may have to buy additional parts to completely solve the problem. The more parts you use, the higher the chance of picking a part that may be discontinued, and the more purchase orders and stocking requirements you generate. Plus, if you need to assemble more parts, the board assembly cost will probably increase. But if a simple AND gate is all you need, a discrete logic device is probably the right choice.

Let's look at the case where you need more than just a simple gate. CPLDs have been continually cost reduced and are now on par with some discrete logic devices. If you are using multiple discrete logic devices, the benefits of a CPLD become even more appealing.

This trend opens up a whole new level of integration at a low price point. For the first time in history, CPLDs can compete with discrete solutions, add functionality, and cost less. For instance, a typical voltage translation device usually costs from $0.50 to $3.50, depending on volume, package type, the number of I/Os to be used, and the process technology of the device. CPLD devices are below $1.00 for the smallest density and lowest cost package. But you get a whole lot more within that single CPLD than any discrete function logic device.

With the addition of I/O banks on CoolRunner-II 32- and 64-macrocell CPLDs, voltage and standard I/O translation can occur in very small, cost-competitive CPLDs. When using a CoolRunner-II CPLD, other benefits also include the following:

- Security
- Input hysteresis
- Optional output control
- Low power operation
- Multiple package options with varying I/O
- IEEE1149.1 Boundary Scan test
- Adjustable from 1.5V to 3.3V

CoolRunner-II Family of CPLDs

Features	XC2C32A	XC2C64A	XC2C128	XC2C256	XC2C384	XC2C512
Macrocells	32	64	128	256	384	512
F_{Toggle} (MHz)	500	500	450	450	450	450
F_{SYSTEM} (MHz)	323	263	263	256	217	217
Max I/O	33	64	100	184	240	270
I/O Banks	2	2	2	2	4	4
LVCMOS, LVTTL (1.5, 1.8, 2.5, 3.3)	Yes	Yes	Yes	Yes	Yes	Yes
HSTL, SSTL	No	No	Yes	Yes	Yes	Yes
DualEDGE	Yes	Yes	Yes	Yes	Yes	Yes
DataGATE, CoolCLOCK	No	No	Yes	Yes	Yes	Yes
Standby Power (μW)	28.8	30.6	34.2	37.8	41.4	45.0
Advanced Security	Yes	Yes	Yes	Yes	Yes	Yes
Packages (Size, Type)	**Maximum User I/O**					
VQ44 (10 x 10 mm, leaded)	33	33				
PC44 (16.5 x 16.5 mm, leaded)	33	33				
CP56 (6 x 6 mm, chip-scale)	33	45				
QFG32 (5 x 5 mm, Pb-free)	21					
QFG48 (7 x 7 mm, Pb-free)		37				
VQ100 (14 x 14 mm, leaded)		64	80	80		
CP132 (8 x 8 mm, chip-scale)			100	106		
TQ144 (20 x 20 mm, leaded)			100	118	118	
PQ208 (28 x 28 mm, leaded)				173	173	173
FT256 (17 x 17 mm, BGA)				184	212	212
FG324 (23 x 23 mm, BGA)					240	270

Table 1 – CoolRunner-II family overview

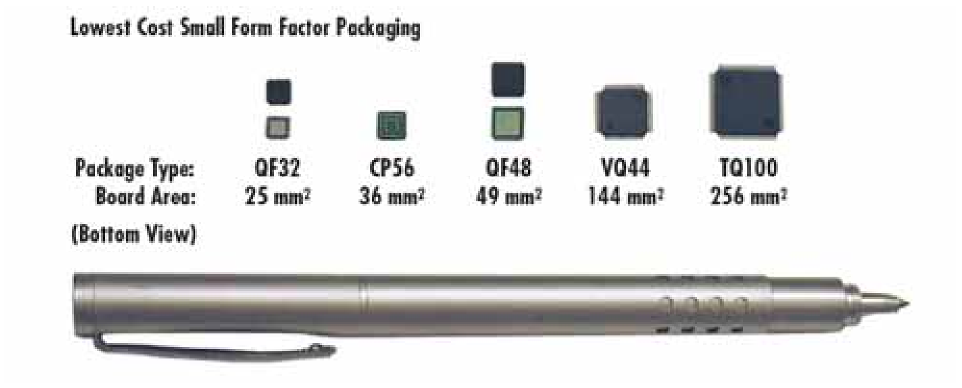

Lowest Cost Small Form Factor Packaging
Package Type: QF32 CP56 QF48 VQ44 TQ100
Board Area: 25 mm² 36 mm² 49 mm² 144 mm² 256 mm²
(Bottom View)

Figure 1 – MLF package comparison

With these features, CPLDs offer more than voltage translation – and can do a lot more. Because discrete level shifters perform a specific task, you may be stuck with a specific I/O choice. With Xilinx CPLDs, you get choices on I/O, and you can use them for other functions. Thus, they are a better choice when you need level translation versatility and some logic, extra I/Os, or other system integration functions. See Table 1 for features and package types.

Additional Low-Cost Packages

With the migration of CPLDs into non-traditional markets, package form factor and cost have become growing concerns. The Xilinx CPLD group has listened to customer needs and found a better small form-factor package. This package is a low-cost alternative to more expensive chip-scale BGA packages with similar I/O counts. The MLF (micro lead frame) package, shown in Figure 1, is small and packs a lot of I/O.

Xilinx has introduced two new MLF packages for the CoolRunner-II family of CPLDs: a 32-pin QFG (quad flat no lead, Pb-free) package with 21 I/O for the 32-macrocell device, and a 48-pin QFG package with 37 I/O for the 64-macrocell device. These additions bridge the gap between more expensive chip-scale BGA and low-cost thin quad flat pack packages.

MLF packages, also known as QFN (quad flat no lead), offer small sizes and high I/O counts, but unlike their BGA chip-scale cousins, are easy to assemble and easy to probe. For assembly, they are like regular 0.5 mm pin spacing thin quad flat pack (TQFP) packages. This makes pin alignment and solder reflow very easy compared to BGA packages.

For debug, MLF packages have external pins that can be probed just as easily as a thin quad flat pack package. These features make it a great addition to the CoolRunner-II family of CPLDs for consumer products.

These MLF packages also offer better electrical characteristics. With their small size, bulk capacitance and inductance are lower than TQFP packages.

Conclusion

Many product choices are available when considering voltage translation, but CPLDs are the best choice for many reasons. You get additional logic capability, more package choices, lower power, free JTAG testability, I/O options, advanced security, and cost savings. Current users will benefit from the same pin-out, as well as the new I/O banking feature.

Novice users can find out how to use these new CPLDs at *www.xilinx.com/cpld/*.

Don't forget that the design software is always free and easy to use. Download your copy of ISE™ WebPACK™ software at *www.xilinx.com/products/design_resources/design_tool/index.htm* to start your design today. With these design tools and world-class support, Xilinx CPLDs make a great choice for logic solutions and replacement of discrete logic devices.

TRAQUAIR
www.traquair.com

The Compact DSP & FPGA Solution

micro-line C6713Compact Embedded DSP/FPGA board

Virtex-II FPGA
250k, 500k or 1MGates

TMS320C6713 DSP
Up to 1800MIPS/1350 MFLOPS

IEEE 1394 FireWire
400MBit/sec, IIDC DCAM, SBP2

Optional Ethernet
10/100BaseT, TCP/IP, UDP, ICMP, IGMP, Telnet, HTTP, SMTP, POP3, FTP

Optional Analog I/O
12/14/16-bit A/D/A via FPGA I/O Pins or DSP EMIF

The micro-line C6713Compact is a high performance single board DSP/FPGA solution, offering exceptional capabilities and flexibility.

Measuring only 67 x 120mm, it combines a Xilinx Virtex-II FPGA with Texas Instruments' most powerful floating point DSP processor, the TMS320C6713, as well as with up to 64MBytes of SDRAM, 8MBytes of FLASH ROM, FireWire, and optional Ethernet communications and analog I/O.

It is suitable for stand-alone operation or as a mezzanine daughter card, and has extensive digital I/O capabilities for easy integration with the on-board FPGA, DSP and FireWire resources.

Download the datasheet and learn more at:
www.traquair.com/ads/xcell/c6713compact.html

Traquair Data Systems, Inc. Tel: 607-266-6000 Email: sales@traquair.com Web: www.traquair.com

Reducing Bill-of-Materials Cost Using Logic Consolidator

Logic Consolidator helps analyze the total cost of discrete logic on the board and the potential savings to be gained by using a Xilinx CPLD.

by Monita Chan
Product Marketing Manager, High Volume
Xilinx Hong Kong
monita.chan@xilinx.com

Ajay Panicker
Field Applications Engineer
CG-CoreEl, India
ajay@cg-coreel.com

System enhancements and cost reduction are becoming major trends for upgrading existing system designs. To be competitive, companies need to enhance their existing feature sets and lower their total cost, providing more for less.

Traditional systems have always used discrete logic devices for implementing the simple glue and control logic that every design requires. The main argument in favor of discrete logic was that it was cheap and easily available. Yet discrete logic was not the most optimal choice from a board area, power consumption, or programmability standpoint.

The introduction of programmable logic devices provided programmability, flexibility, and higher levels of integration. However, it also came at a cost premium over discrete devices.

Improvements in manufacturing processes and packaging technologies in the last 10 years have significantly lowered the cost of programmable logic devices. Today, CPLD unit costs have been driven down to a level such that they can replace discrete logic devices, yet maintain advantages such as higher performance, design flexibility, re-programmability, and reliability (Figure 1).

Logic Consolidator

To help customers quickly and accurately estimate their costs and area savings by using a single CPLD instead of multiple discrete devices, Xilinx® offers a tool called Logic Consolidator. This is a Microsoft Excel-based tool that estimates the number of discrete logic devices that can be integrated into a single CPLD. It compares Xilinx XC9500XL and CoolRunner™-II 10,000-unit pricing to the 10,000-unit average resale price of 7,400 discrete logic products in the lowest cost plastic package. Also, based on particular quotes, you can specify your own unit pricing for accurate comparison.

Logic Consolidator makes life easy for designers as well as managers. For purchasing managers, the easy-to-use Excel tool enables them to accurately list the number of near-obsolete discrete logic devices that can be replaced by the most optimal Xilinx CPLD – and the cost savings justifying such an upgrade. For designers, it saves precious design time that is usually spent estimating the right PLD to replace a range of discrete logic devices.

In this article, we'll present a real-life case study where Logic Consolidator helped a customer analyze the real cost benefits provided by Xilinx CPLDs over discrete logic devices, saving thousands of dollars in the process.

Customer Case Study

The Logic Consolidator tool was instrumental in convincing an initially skeptical customer to integrate all of the discrete logic on their board to a single Xilinx CPLD.

Wipro Technologies, one of the biggest hardware design services companies in India

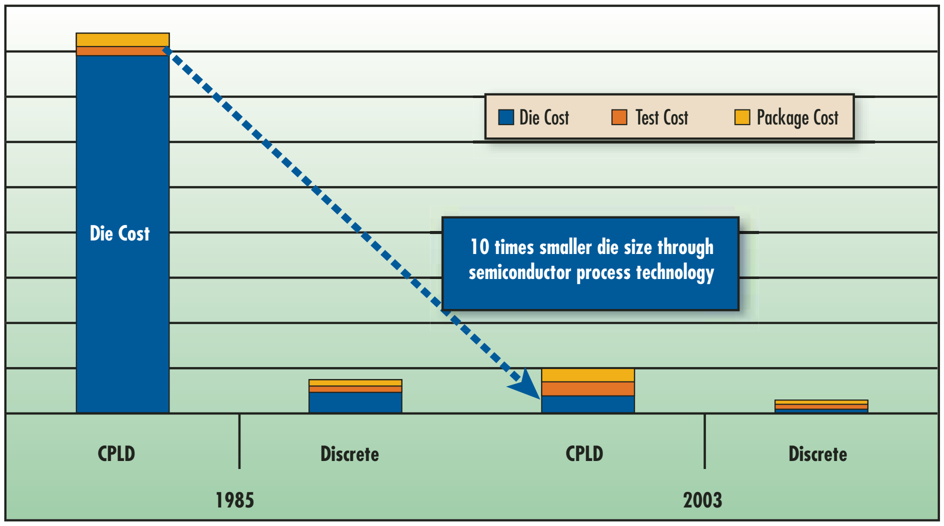


Figure 1 – The costs of a CPLD have come down dramatically because of lower die sizes.

(and a Xilinx XPERTS partner), was working on a project aimed at upgrading an existing telecom product, mainly to cut manufacturing costs and include some enhanced features. During the initial stage, CG-CoreEl's Xilinx sales team introduced Logic Consolidator to the customer and helped them identify the appropriate CPLD for replacing multiple discrete devices on board.

The tool is very intuitive. It saves a lot of time in finding the right CPLD to replace all of the discrete logic. Figure 2 shows a screenshot of the Logic Consolidator tool as used by the customer.

In this case, Wipro was able to identify 73 discrete logic devices that could be replaced by a single Xilinx CPLD.

They could choose either a XC9500XL device or a CoolRunner-II device, depending on the supply voltage available. Both options offered an immediate bill-of-materials cost savings – $0.74 with the XC95288XL device and $1.81 with the XC2C256 device, with about 40% more room available in the CPLD to add more logic for any feature enhancement.

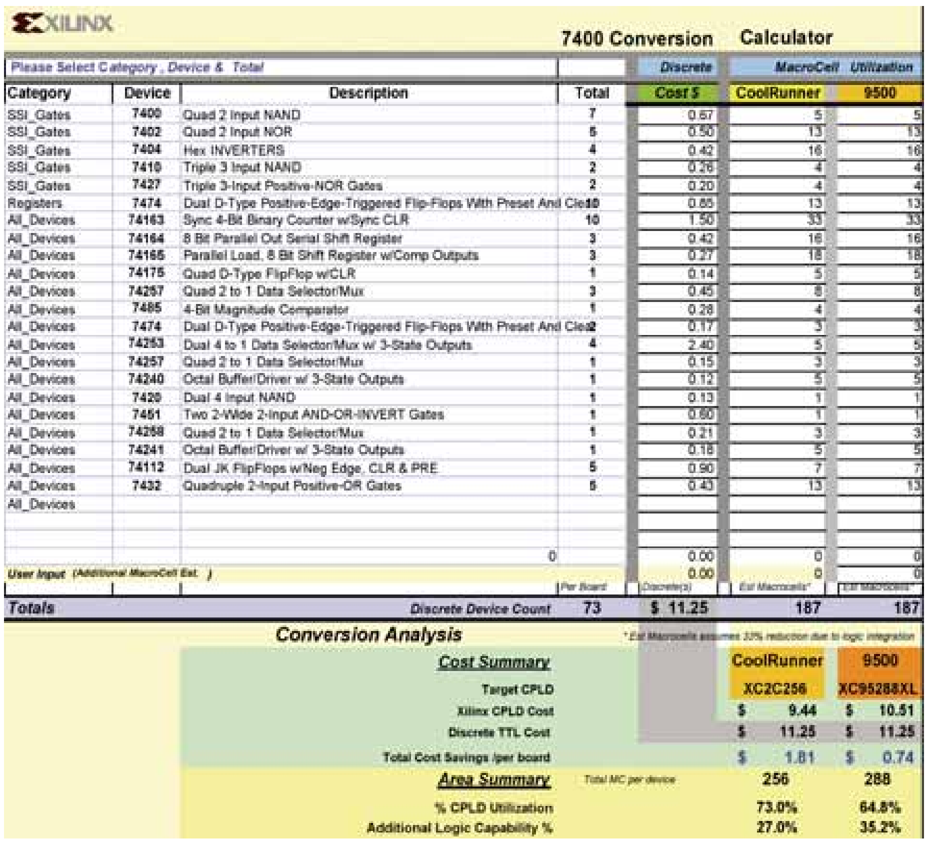
When Wipro engineers actually implemented the design, because of optimizations performed by the Xilinx ISE™ software tool, the design fitted in the next smallest Xilinx device, the XC95144XL-10TQ144C, with 144 macrocells With a listed price of $6.45 for the XC95144XL device, it provided further cost savings for the customer. On the PCB front, the potential savings are approximately more than 70% in board real estate, 60% or more in board routing, and a 50% reduction in board design time.

Add to this the benefits of programmability, reduced board complexity, and increased reliability through reduced power consumption and lower EMI, and it was apparent to Wipro that the Xilinx CPLD was the obvious choice.

After using Logic Consolidator for this card, Wipro was so impressed with its efficiency and usefulness that they decided to use it for all of the other cards in their telecom product upgrade project, and are planning to use it in future projects as well.

Conclusion

Logic Consolidator is a powerful tool that allows customers to analyze the costs of different alternatives for themselves. Because it is built using Microsoft Excel, it is easy to learn and use. To get a copy of Logic Consolidator, contact your Xilinx sales representative. And for more information about or to download Logic Consolidator, visit *www.xilinx.com/products/cpldsolutions/logic_tool.htm*. A new version now includes more logic device choices.



XILINX

7400 Conversion Calculator

Please Select Category, Device & Total

Category	Device	Description	Total	Discrete Cost $	MacroCell Utilization CoolRunner	9500
SSI_Gates	7400	Quad 2 Input NAND	7	0.67	5	5
SSI_Gates	7402	Quad 2 Input NOR	5	0.50	13	13
SSI_Gates	7404	Hex INVERTERS	4	0.42	16	16
SSI_Gates	7410	Triple 3 Input NAND	2	0.26	4	4
SSI_Gates	7427	Triple 3-Input Positive-NOR Gates	2	0.20	4	4
Registers	7474	Dual D-Type Positive-Edge-Triggered Flip-Flops With Preset And Cle	10	0.85	13	13
All_Devices	74163	Sync 4-Bit Binary Counter w/Sync CLR	10	1.50	33	33
All_Devices	74164	8 Bit Parallel Out Serial Shift Register	3	0.42	16	16
All_Devices	74165	Parallel Load, 8 Bit Shift Register w/Comp Outputs	3	0.27	18	18
All_Devices	74175	Quad D-Type FlipFlop w/CLR	1	0.14	5	5
All_Devices	74257	Quad 2 to 1 Data Selector/Mux	3	0.45	8	8
All_Devices	7485	4-Bit Magnitude Comparator	1	0.28	4	4
All_Devices	7474	Dual D-Type Positive-Edge-Triggered Flip-Flops With Preset And Cle	2	0.17	3	3
All_Devices	74253	Dual 4 to 1 Data Selector/Mux w/ 3-State Outputs	4	2.40	5	5
All_Devices	74257	Quad 2 to 1 Data Selector/Mux	1	0.15	3	3
All_Devices	74240	Octal Buffer/Driver w/ 3-State Outputs	1	0.12	5	5
All_Devices	7420	Dual 4 Input NAND	1	0.13	1	1
All_Devices	7451	Two 2-Wide 2-Input AND-OR-INVERT Gates	1	0.60	1	1
All_Devices	74258	Quad 2 to 1 Data Selector/Mux	1	0.21	3	3
All_Devices	74241	Octal Buffer/Driver w/ 3-State Outputs	1	0.18	5	5
All_Devices	74112	Dual JK FlipFlops w/Neg Edge, CLR & PRE	5	0.90	7	7
All_Devices	7432	Quadruple 2-Input Positive-OR Gates	5	0.43	13	13
All_Devices						
			0	0.00	0	0
User Input (Additional MacroCell Est.)				0.00	0	0
			Per Board	Discrete(s)	Est Macrocells*	Est Macrocells*
Totals		Discrete Device Count	73	$ 11.25	187	187

Conversion Analysis

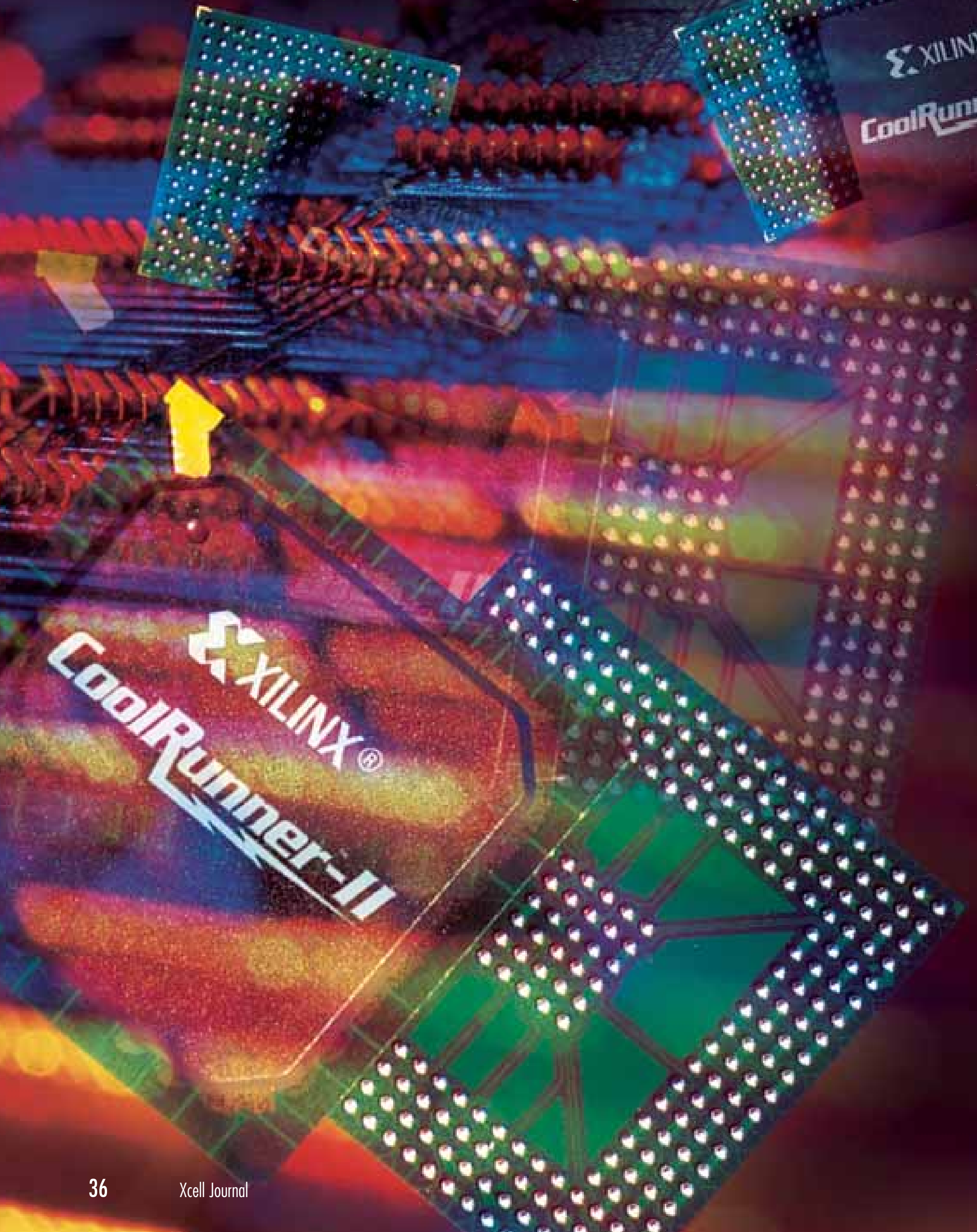
*Est Macrocells assumes 33% reduction due to logic integration

	CoolRunner	9500
Cost Summary		
Target CPLD	XC2C256	XC95288XL
Xilinx CPLD Cost	$ 9.44	$ 10.51
Discrete TTL Cost	$ 11.25	$ 11.25
Total Cost Savings /per board	$ 1.81	$ 0.74
Area Summary Total MC per device	256	288
% CPLD Utilization	73.0%	64.8%
Additional Logic Capability %	27.0%	35.2%

Figure 2 – Logic Consolidator helped Wipro Technologies understand the cost benefits of using a CPLD in place of multiple discrete logic devices.

Using CoolRunner-II Silicon Features to Reduce Cost

CoolRunner-II CPLDs offer features that lower total PCB costs through component count reduction, smaller board space, and lower manufacturing costs.



by Steve Prokosch
Product Solutions Marketing Manger
Xilinx, Inc.
sprokosch@xilinx.com

Xilinx® introduced the CoolRunner™-II family of CPLDs in 2001, continuing to lower power consumption through process technology improvements. This is what you would expect from a technology-driven company. But differences in this family exist that have never been employed in a re-programmable logic device. These features deliver even lower power, higher speed, and most importantly, integrated functions that reduce system cost.

The CoolRunner-II design team looked at methods to add functionality while preserving the low-cost aspect of the current CPLD market. To this end, clock doublers, clock dividers, input hysteresis, and I/O banking were designed in from a cost-per-feature standpoint. This led to extremely creative ways to design features in the smallest amount of silicon. If the feature did not reduce system cost enough to merit the die size increase, that feature was not designed into the device. With this balance of features and cost, CoolRunner-II CPLDs offer a unique advantage when compared to other devices.

Cost-Reducing Features

Above and beyond the normal process technology shrinks that every silicon component goes through, CoolRunner-II CPLDs have very special features that help most designers lower the total components in their design. These component reductions can be very simple or very complex, depending on how creative you are as a designer. They range from clocking features to integration of simple logic functions to voltage and I/O-level differences. The principal idea behind these features is to integrate functions and simplify the designer's task. The first and most basic is clock generation.

For most designs, a single clock source may not generate all of the necessary frequencies needed in the design. Also, if the PCB is larger than a hand-held device, clock skew may come into play. The design team considered some of these clocking design problems and created solutions internal to the CPLD.

Clock Doubler and Divider

Two features included with CoolRunner-II devices are DualEDGE and (in 128-macrocell and higher devices) a clock divider. DualEDGE flip-flops offer a performance boost for sequential operations, while the clock divider can reduce power consumption.

But that's just the tip of the iceberg. Let's start with a simple clock-doubling scheme where we can generate a frequency double that of the incoming clock. This yields two clock sources from a single clock input. So if you happen to need a faster clock to improve a serial data flow, DualEDGE flip-flops are a perfect fit. It also works well for improved pulse width modulation or higher resolution of timers/counters. Thus, you need not use another clock source for portions of your design that need to run faster.

Another key point is the fact that you can preserve a clock input pin. So with DualEDGE flip-flops, you get a free clock without using an input pin on the CPLD. Figure 1 shows a simple diagram of what DualEDGE flip-flops look like.

The clock divider feature (Figure 1) is another way to eliminate extra clock sources while again preserving valuable I/O pins. The neat thing about the CoolRunner-II clock divider circuit is that it actually improves duty cycle. For instance, if you have a 60/40 duty cycle on an incoming clock source, the internal divider circuitry actually performs duty-cycle correction. So internal to the CPLD, this will have a 50/50 duty cycle. This may help circumvent clock skew problems and again eliminate multiple oscillator inputs.

The clock divider is not simply a divide by two; it has eight different divide-by settings (see Figure 2). So if you use the clock divider in conjunction with DualEDGE flip-flops, you can generate multiple combinations of clock frequencies. And for those troublesome odd clock frequency problems, just use an integer divide-by that yields an odd number (for example, 6, 10, 14) to get even more clock choices.

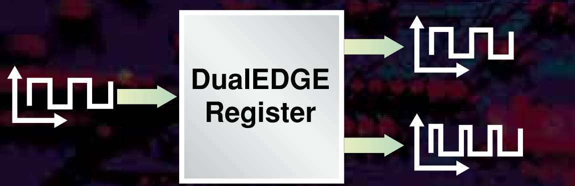


Figure 1 – Xilinx CoolRunner-II clock doubler

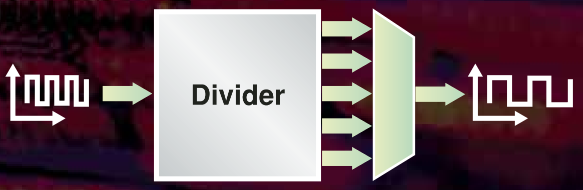


Figure 2 – Xilinx CoolRunner-II clock divider

Input Pin Features

Because of the low power nature of CoolRunner-II CPLDs, the design team believed that there may be portable applications that go into noisy environments. This led to the idea of integrating hysteresis on input pins. This single feature, available on all CoolRunner-II CPLDs, adds little to silicon area but brings many advantages. With input hysteresis, the problem of noisy environments was eliminated – and as a side benefit reduced power. This power reduction is subtle, but nonetheless an advantage that other CPLDs do not include.

So how does it reduce cost? Because the hysteresis is programmable on a pin-by-pin basis, it eliminates the need for external Schmitt trigger devices. Although these components may be cheap, the extra insertion cost could prove expensive if added after the initial board fabrication. Better to be safe than sorry when it comes to reliability.

One other cost that is commonly overlooked is PCB routing and the associated costs with extra layers of circuit board material. Fewer layers is always less expensive.

I/O Banking – Voltage and I/O Standard Translation

With today's vast selection of components, the chances of picking parts that all have the same voltage input levels are slim. Consider the time-to-market demand on today's design engineers. If you cannot reuse some portion of the previous design, chances of making the current product release cycle are reduced. For example, a design that uses legacy ASSP devices may not line up with the same voltage or I/O structure of the new processor you just specified.

With the simple addition of a CPLD, the voltage problem goes away – and you get extra logic for all of those new features that marketing dreamed up for the next generation of products. But that's not all, because you still need to consider if the I/O standards match. CoolRunner-II CPLDs do that for you as well.

What does this have to do with cost savings? If we look at the cost of discrete devices versus low-cost Xilinx CPLDs, you can achieve the same function for less money and get a whole lot more. Xilinx has a free downloadable tool called Logic Consolidator to show you just how much you can save (recently updated with even more parts to compare). You can download it at *www.xilinx.com/products/cpldsolutions/logic_tool.htm*. There are even associated documents to explain how we did the comparison.

That's still not all, because there is more to cost than the silicon. In fact, if you look at the cost of discrete devices and the associated stocking, shipping, assembly, and routing costs, in most cases a single integrated chip costs less. Also note the CoolRunner-II CPLD's added reliability, versatility, and low power.

Other Cost-Reducing Ideas

Still other features on CoolRunner-II CPLDs may be overlooked. OTF (on-the-fly) reprogramming is one of those features that can make your CPLD design do double duty for no extra cost. Here's one good example of using OTF: on board power-up, use a CPLD to configure the Xilinx FPGA devices on board. Once that is complete, reprogram the CPLD to perform some system function that requires power management, integrating discrete logic functions, or implementing fixes to standard products. For more information regarding OTF, see *http://www.xilinx.com/bvdocs/appnotes/xapp388.pdf*.

If you're designing a small-form-factor device, consider using a new package option for Xilinx CoolRunner-IIA CPLDs. The package is commonly referred to as micro lead frame or quad flat no-lead and is extremely small, yet priced similarly to standard thin-quad flat package devices. This package is an attractive way to use minimal board space and still get the great low-power operation of CoolRunner-II CPLDs.

Conclusion

Perhaps you will be able to apply some of these cost-saving ideas to your next design. If you need more information on any of these topics, there is a wealth of information from the Xilinx CPLD website at *www.xilinx.com/cpld/index.htm*.

If you need something more substantial on these features, an excellent application note exists at *http://www.xilinx.com/bvdocs/appnotes/xapp378.pdf*.

With these innovative features and your imagination, design problems will seem more manageable and hopefully make your design more successful.

Nucleus

Embedded FPGA made easy.

Increase your embedded software performance by using Nucleus PLUS on your processor-based Xilinx designs. By integrating with Xilinx's MLD technology, Nucleus PLUS is configured to your system design automatically, making software integration with your FPGA design easy. The compact design of Nucleus PLUS can run from on-chip memory to help minimize power dissipation and deliver increased performance. This, combined with a wealth of middleware, including the Nucleus NET TCP/IP solution supporting Xilinx Ethernet IP, makes it ideal for products targeted at the networking, telecommunications, datacoms and consumer markets.

Nucleus PLUS is readily available for:

- **MicroBlaze**
- **PowerPC 405**
- **Spartan-3 and Virtex-4**

Discover how Nucleus software can make your embedded development for Xilinx FPGAs easy. Go to our dedicated Web page for more articles and free downloads at **www.acceleratedtechnology.com/xilinx.**

Nucleus EDGE, our new embedded development environment based on Eclipse, offers a complete tools solution for developing designs with Xilinx PowerPC cores. Nucleus EDGE provides you with a Builder, Editor and Project Manager, utilizing optimizing compilers, comprehensive embedded debuggers and RTOS profiling tools.

For more information on the Nucleus complete development solution from UML through to embedded RTOS, go to **www.acceleratedtechnology.com.**

Nucleus. Embedded made easy.

Accelerated Technology®
A Mentor Graphics Division

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

Using Spartan-3/3E Features to Area-Optimize Your Design

Spartan-3/3E devices include features that were typically the domain of only high-end FPGAs – distributed memory, embedded multipliers, and others that can yield significant optimization.

by Suhel Dhanani
Senior Marketing Manager, Spartan Products
Xilinx, Inc.
suhel.dhanani@xilinx.com

Ken Chapman
Staff Engineer, General Products Division
Xilinx, Inc.
ken.chapman@xilinx.com

The Xilinx® Spartan™-3 series is the first low-cost FPGA that does not compromise on features – it offers fast embedded multipliers, look-up tables that can be optimally configured to form shift registers, distributed memory, and large amounts of embedded block RAMs.

All of these features let you design for the lowest possible area, lowering the size and thereby the cost of the FPGA in your design.

With more than 100 million units shipped, Spartan devices have quickly become the world's most accepted low-cost FPGA architecture, familiar to thousands of engineers. With every generation of the Spartan architecture, Xilinx has rolled out even more logic and I/O at a lower price. With the release of the Spartan-3 series we did the same, but also added embedded features that let you integrate even more functionality in your favorite Spartan device. Table 1 shows the advantages of these features.

Features That Optimize Area

The Spartan-3/3E architecture is very memory-intensive. Not only does this architecture have embedded 18 Kb block RAMs, but half the look-up tables (LUT) can be configured as memory (Figure 1).

Spartan-3/3E Features To Reduce Area Utilization	
Embedded Multipliers	Area-efficient implementation of various DSP functions
Up to Eight Digital Clock Managers (DCM)	Clock management functions such as clock multiplication/ division, phase alignment can be integrated within the FPGA
Distributed RAM	Small FIFOs, LIFO buffers, scratch-pad memory, register banks
18 Kb Block RAM	Large FIFOs, buffers, cache memory, program storage for processors
LUT Configured as 16-bit Shift Register (SRL16E)	FIFOs, delay lines, state machines, logic-register, tradeoff in regular intensive designs
Differential Signaling Support – LVDS, RSDS	Lower number of pins, reduced power consumption, reduced EMI, high noise immunity

Table 1 – Features included in the Spartan-3/3E family of devices

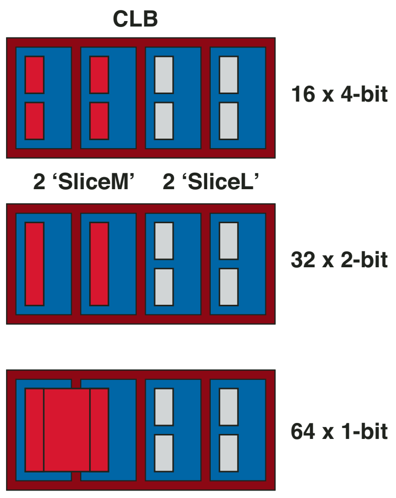


Figure 1 -The LUTs in two slices of the CLB are configurable as 16 x 4, 32 x 2, or a 64 x 1 memory.

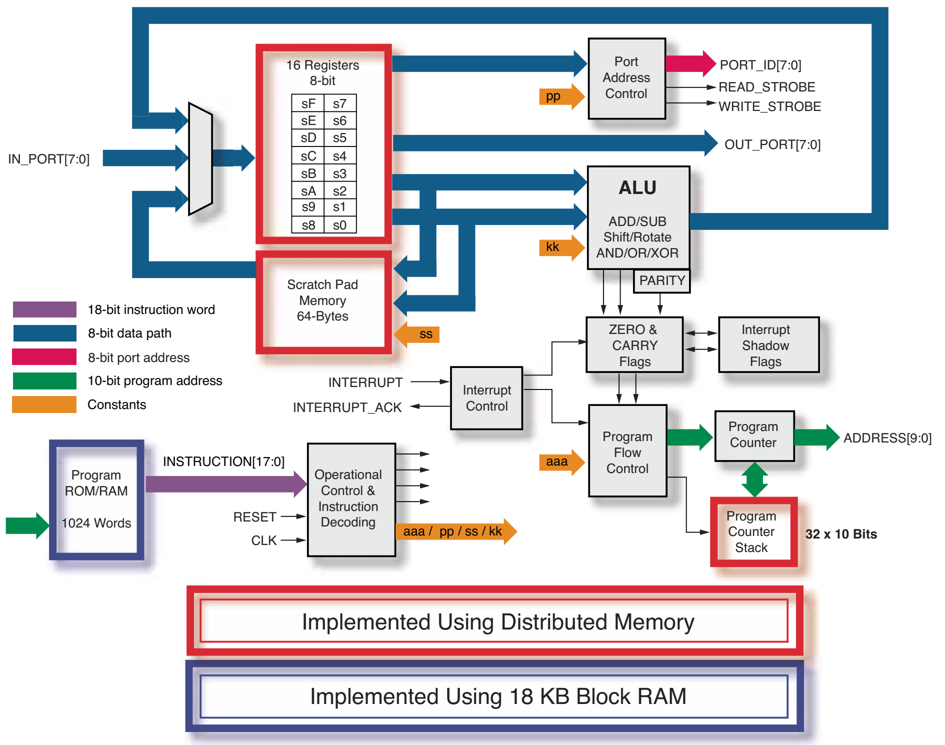


Figure 2 – The 8-bit PicoBlaze microcontroller takes only 192 logic cells by making effective use of Spartan-3/3E memory resources.

The advantage of having distributed memory is that you can very efficiently implement any design that requires a large number of smaller size memory structures without running out of registers or block RAMs, offering the potential for unrivaled data bandwidth. A good example of this is the 8-bit PicoBlaze™ microcontroller, which takes only 192 logic cells (LC) within the Spartan-3/3E architecture. As shown in Figure 2, this is possible only because the register stack, scratchpad memory, program counter stack, and program ROM are constructed using the FPGA memory resources.

A 4-input LUT in a slice-M can also be configured in the SRL16E mode. This provides a 16-bit shift register with clock enable in addition to the dedicated flip-flop, as shown in Figure 3.

When writing to this shift register, the new data is always placed in location "0" and all other data moves along by one location. However, data can be read from any location using the address inputs A[3:0].

The SRL16E provides a very cost-efficient way to implement a delay line. For example, Figure 4 shows how one LUT can be used to generate a 5-cycle delay for the data. Simply apply the data into the SRL16E data input, hard-code the address pins at "0100," and read the data out from that address. Synthesis tools can implement this technique automatically for anything up to a 16-cycle delay in each LUT.

The FIFO is a common component of many system designs. The normal way to construct a FIFO is by using a memory (Figure 5). Because of the need to write at the same time as reading, dual-port memory is required. Two address counters are also needed to point at the write address and the read address. Additional comparator logic is then required to determine the state of the FIFO: full, empty, or half-full.

In Figure 6, a FIFO can be implemented using one SRL16E for each bit of the data path width (Din). The SRL16E gives a dual-port operation but in half the space of the real dual-port memory. We also only need one up/down counter instead of two up counters, which again is half the space. As a final bonus, the counter status tells you precisely how many words are stored in the FIFO, with the most significant bit naturally providing a most useful "half-full flag."

The embedded multipliers available in the Spartan-3 fabric are useful in a variety of DSP applications. Each multiplier can replace as many as 450 logic cells that would normally be required to implement a multiply function.

Many implementations of finite impulse response (FIR) filters – base stations, digital

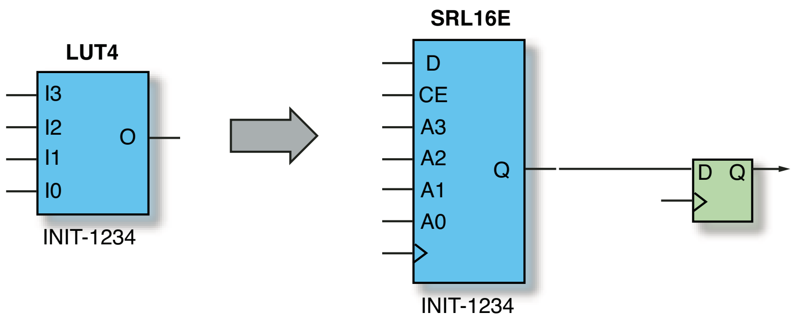


Figure 3 – Half of the LUTs in the Spartan-3 FPGA are configurable as a 16-bit shift register.

video systems, wireless LANs, xDSL, and cable modems – use multiply-accumulate functions. You can implement high-performance FIR filters that use multiple MACs with minimal area penalty within the Spartan-3 fabric. Such MAC-intensive functions would ordinarily take significant logic resources in competing FPGAs that lack embedded multipliers.

Conclusion

Spartan-3/3E features were designed to save costs by allowing the design to be implemented using the smallest (and thereby the lowest cost) silicon. Spartan devices utilize the world's most advanced manufacturing process and incorporate features that allow you to integrate even more functionality in the smallest FPGA.

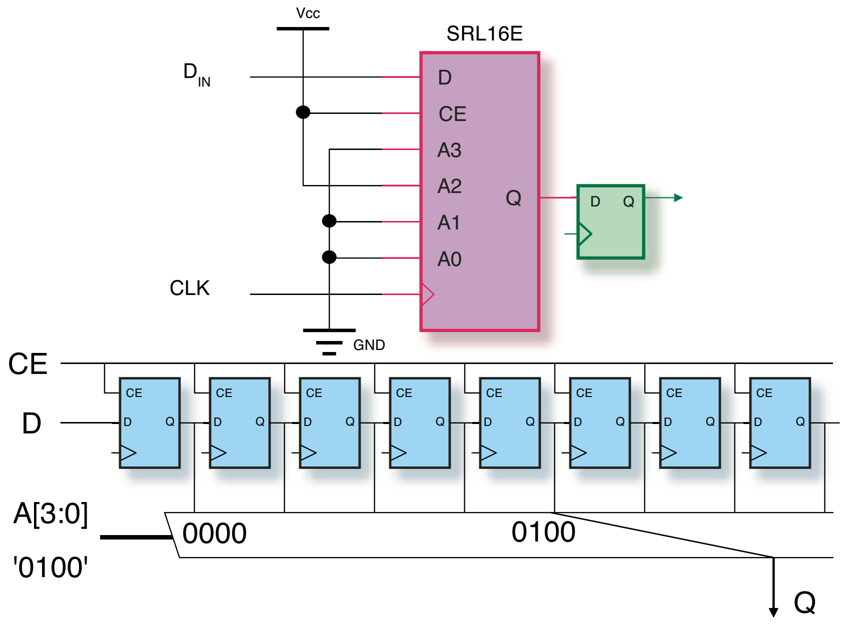


Figure 4 – LUT configured to generate a 5-cycle delay

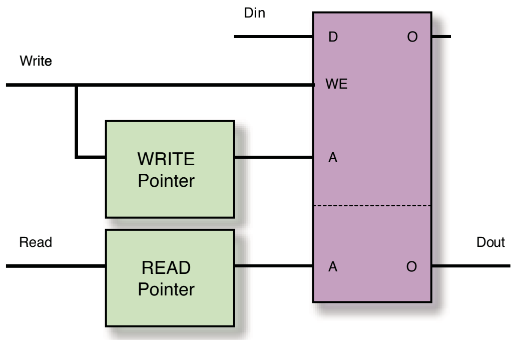


Figure 5 – Traditional FIFO requiring dual-port RAM and two counters

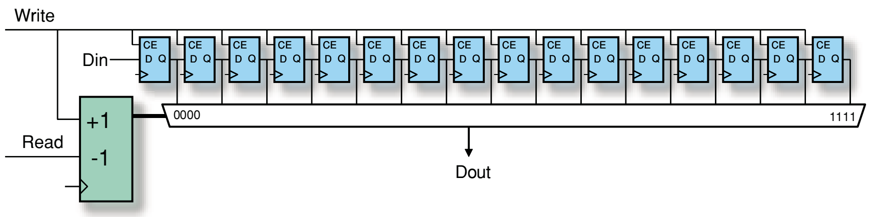


Figure 6 – SRL16E mode with a single counter can implement a FIFO. The data path would normally be several LUTs wide.

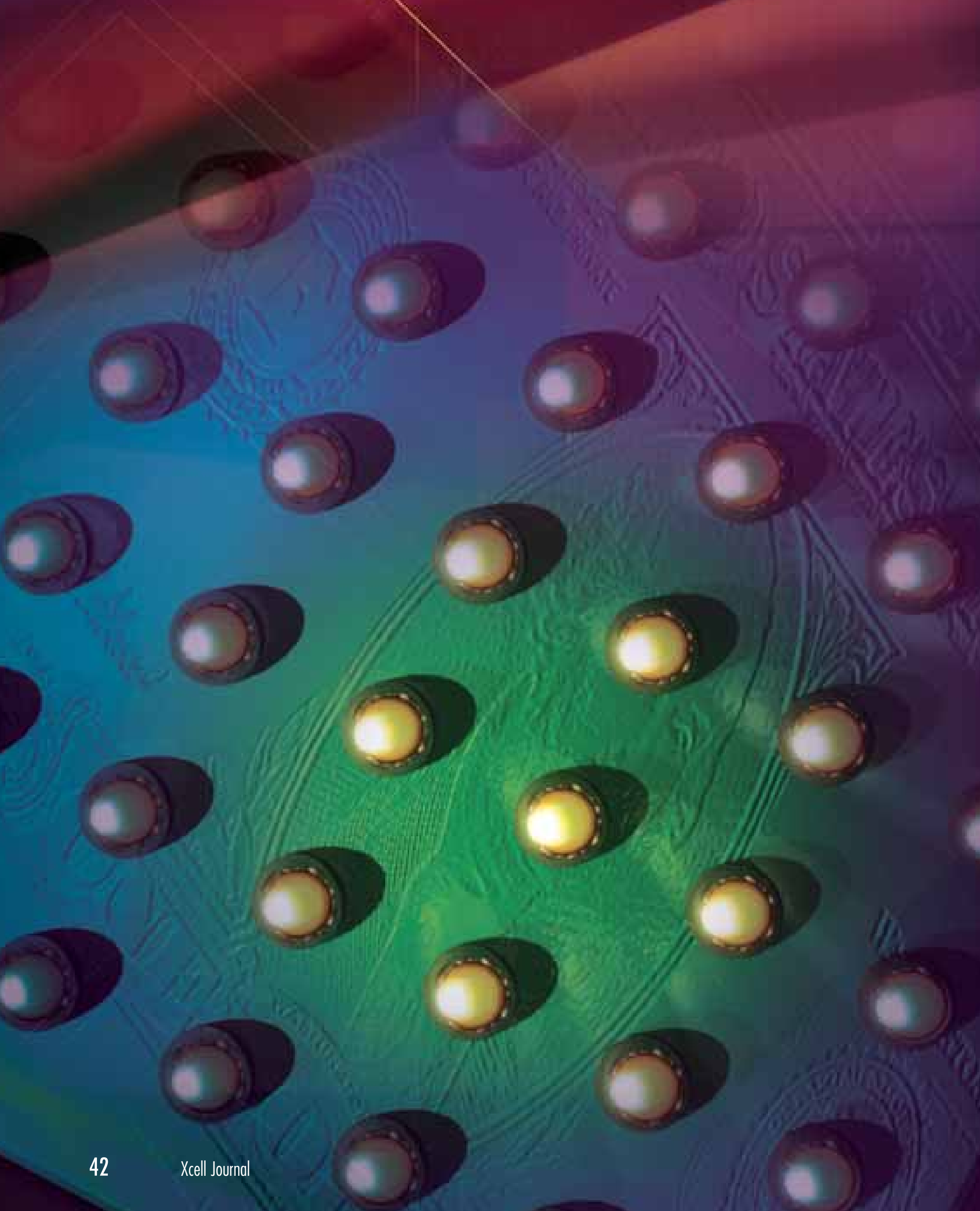
Synthesis Tools Support SRL16E Mode

The following code describes the full functionality of an SRL16E – and yet it will be automatically reduced to an SRL16E by the synthesis tools. The entire code is thus implemented using one look-up table:

```
storage: process (clk)
 begin
if clk'event and clk='1' then
 —enable only applies to the input path
 if input_enable = '1' then
  store0_data <= data_in;
  store1_data <= store0_data;
  store2_data <= store1_data;
  store3_data <= store2_data;
  store4_data <= store3_data;
  store5_data <= store4_data;
  store6_data <= store5_data;
  store7_data <= store6_data;
  store8_data <= store7_data;
  store9_data <= store8_data;
  store10_data <= store9_data;
  store11_data <= store10_data;
  store12_data <= store11_data;
  store13_data <= store12_data;
  store14_data <= store13_data;
  store15_data <= store14_data;
 end if;
 —Select only applies to the output path
 case output_select is
  when "0000" => data_out <= store0_data;
  when "0001" => data_out <= store1_data;
  when "0010" => data_out <= store2_data;
  when "0011" => data_out <= store3_data;
  when "0100" => data_out <= store4_data;
  when "0101" => data_out <= store5_data;
  when "0110" => data_out <= store6_data;
  when "0111" => data_out <= store7_data;
  when "1000" => data_out <= store8_data;
  when "1001" => data_out <= store9_data;
  when "1010" => data_out <= store10_data;
  when "1011" => data_out <= store11_data;
  when "1100" => data_out <= store12_data;
  when "1101" => data_out <= store13_data;
  when "1110" => data_out <= store14_data;
  when "1111" => data_out <= store15_data;
  when others => data_out <= 'X';
 end case;
end if;
 end process;
```

Low-Cost EasyPath FPGAs Offer Promise to ASSP Companies

EasyPath FPGAs offer quick time to market and flexibility at prices below those offered by structured ASICs.



by Gokul Krishnan
Senior Marketing Manager
Xilinx, Inc.
gokul.krishnan@xilinx.com

As the semiconductor industry continues its march towards smaller process geometries, it has to deal with exponentially increasing mask set costs. Customers who want to spin a 90 nm ASIC must pay upwards of a million dollars in NRE. In addition, they must incur significant engineering and design expenses, increasing the overall cost of product development to prohibitive levels.

ASSP companies have come under increasing pressure to weigh more carefully the risk/reward tradeoffs for introducing new products. ASSP vendors are being forced to develop only those "blockbuster" products that will allow a reasonable return on their large up-front investments.

Recent innovations such as Xilinx® Spartan™-3 FPGAs and EasyPath™ technology, when used in combination with a wide IP portfolio, allow ASSP vendors to address many more opportunities in a cost-effective way while assuming minimal risks. In this article, we will examine how Spartan-3 FPGAs – together with their EasyPath counterparts – can alleviate some of the challenges faced by consumer electronics ASSP vendors.

Customer-Specific FPGAs

Customer-specific FPGAs like those in the EasyPath program are identical to standard FPGA offerings, but use patented testing techniques to significantly improve yield. Once your design is frozen, Xilinx develops custom test patterns that specifically test only the resources used by that particular design. You reap the benefits of these consequently higher yields in the form of lower costs.

With EasyPath FPGAs, you can realize a 30-80% reduction in unit prices (compared to equivalent standard FPGAs) when you move to high volume. Because customer-specific FPGAs are the same piece of silicon as standard FPGAs – except cheaper – they offer a one-for-one match of all the features offered in a standard FPGA. As a result, almost no involvement is required from customer engineering teams; the conversion process from design completion to production is completely seamless.

Furthermore, the lead times from the time a design is frozen to volume production in customer-specific FPGAs can be as short as 8-10 weeks – an advantage of almost 12 weeks when compared to ASIC methodologies. Table 1 shows a comparison of EasyPath FPGAs versus structured ASIC methodologies, illustrating why EasyPath FPGAs are a better overall solution.

Selection Criteria	Structured ASICs	*EasyPath* FPGAs
Time to Prototype Samples	4-8 Weeks	0 Weeks
Total Time to Volume Production	12-15 Weeks	8 Weeks
Vendor NRE/Mast Costs	$100K-200K	$75K
Design Cost for Conversion	$250K-$300K	$0
Additional Cost of Tools for Conversion	$100K-$200K	$0
Unit Costs	Low	Low
Risk	High	Low
Flexibility to Make Changes In-System	Inflexible	Flexible
Design Conversion from Prototype to Production	Additional Engineering	Conversion Free

***Xilinx Market Analysis**

Table 1 – EasyPath FPGAs offer significant advantages in time to market and total cost of ownership when compared to structured ASICs.

Addressing ASSP Market Needs

Historically, ASSP vendors have used FPGAs as prototyping vehicles, moving to custom ASICs in high volume to take advantage of their low unit costs. This was a reasonable strategy at 0.18μ and larger process nodes because lower NRE costs allowed companies to take the financial/market risk and go to market with multiple products.

As the demands of system integration, performance, and power have pushed ASSP vendors down to 0.13μ and 90 nm, NRE costs and design complexity have increased significantly. A study in International Business Strategies' Fourth Quarter 2003 report estimates the product development cost of a 0.13μ ASIC to be greater than $10 million, including the cost of design, verification, and prototyping. This implies that an ASSP vendor has to sell at least 250,000 units of a $40 ASP device (see Figure 1) just to recoup the development cost. Clearly, if other sales and marketing costs are included and a reasonable ROI is expected, the lifetime volume potential of a device must be significantly higher.

Customer-specific FPGAs, on the other hand, significantly reduce the market/financial risk element. You can now have multiple variations of a particular design (or indeed, multiple designs) catering to different market segments – and go to market with all of them at the same time. This is because the NRE charges are minuscule compared to ASICs. You no longer need to spend valuable engineering resources to convert an FPGA design into an ASIC RTL, nor spend multiple weeks in verification and simulation to benefit from low-cost, high-volume solutions.

Time to Market

In today's world of complex design cycles, short product life cycles, and quickly changing market needs, time to market is of paramount importance. Unfortunately, the complexity of deep sub-micron designs increases the time required to develop and debug a design, while increasing the risk of re-spins.

Because ASIC re-spins can set back a schedule at least three months, ASSP vendors take a tremendous risk of being late to market. Some studies have shown (as in the IBS 2003 report) that just a three-month slip in the schedule can cause a product revenue

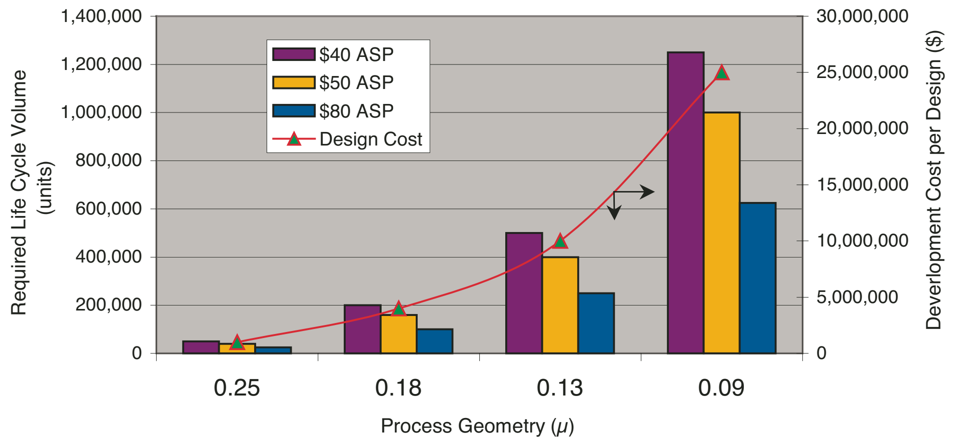


Figure 1 – The increasing cost of developing ASSPs as process nodes shrink means that companies are forced to go after high-volume "blockbuster" products to get a reasonable return on investment.

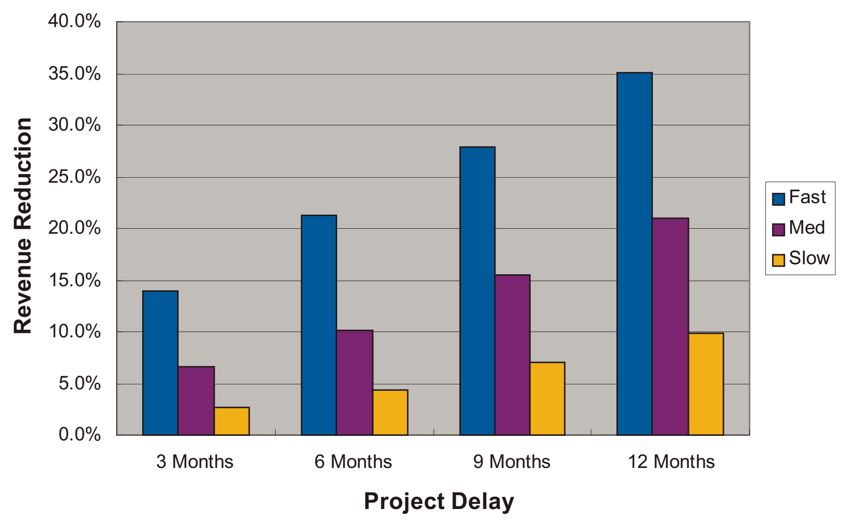


Figure 2 – The impact of time to market on product revenue in fast-, medium-, and slow-moving market segments (IBS 2003).

reduction of about 15% (see Figure 2). In the extreme case of seasonal products (widely prevalent in the consumer electronics industry), schedule slippage can mean that an ASIC product will miss customer demand altogether.

Customer-specific FPGAs allow ASSP vendors to postpone spending decisions until market uncertainties are removed. Once a design is completed, you no longer need to plan three to four months in advance and hope that the silicon you get at the end of the day works to your specifications. With one of the highest levels of reliability and shortest lead times in the industry, customer-specific FPGAs require ASSP vendors to incur little to no risk in the conversion process. Furthermore, with just eight to ten weeks to production, ASSP vendors can go to market with confidence and hit that prime market window.

Technical Challenges

A common challenge that many ASSP vendors have to deal with is the changing landscape of industry standards. This is particularly true in wired telecom and wireless applications, where the lack of a coordinated effort to specify various telecommunications protocols and interface standards can result in incompatibilities between different vendors' products.

Ratifying standards is a time-consuming process and vendors often have to freeze their designs much earlier to get first-mover advantages. ASSP vendors must either gamble that their final specifications will get adopted in a standard; implement a superset of all possible future combinations; or delay the design phase, running the risk of being late to market. These options are far from optimal.

One of the major advantages of FPGAs over ASICs is the flexibility to make design changes in case of a specification change or design error. Traditionally, designers have had to forgo this advantage as they move from FPGAs to an inflexible custom solution like standard cell or structured ASIC. However, recent developments by Xilinx in customer-specific FPGAs now preserve some of the flexibility of the FPGA while also maintaining a low-cost approach.

Xilinx Spartan-3 and Virtex™-4 EasyPath FPGAs enable you to prototype two different designs in a standard FPGA and then move to an EasyPath device that supports both those designs simultaneously in production. For example, you can use one bitstream to perform system diagnostics on the entire system and the other to load the second application-specific bitstream. Alternatively, you can choose to implement designs for two different products in a single device and take advantage of a common part number for inventory management purposes.

In addition, with Spartan-3 and Virtex-4 EasyPath FPGAs, you can change LUTs and I/Os even after these devices have been deployed in the field (see Figure 3). As a result, you can now fix minor bugs or tweak certain parameters to further optimize your designs.

For instance, a line card in a router might need to have the drive strength (and slew rate) adjusted a notch or two depending on what load it encounters. You can implement a range of drive strengths that are available

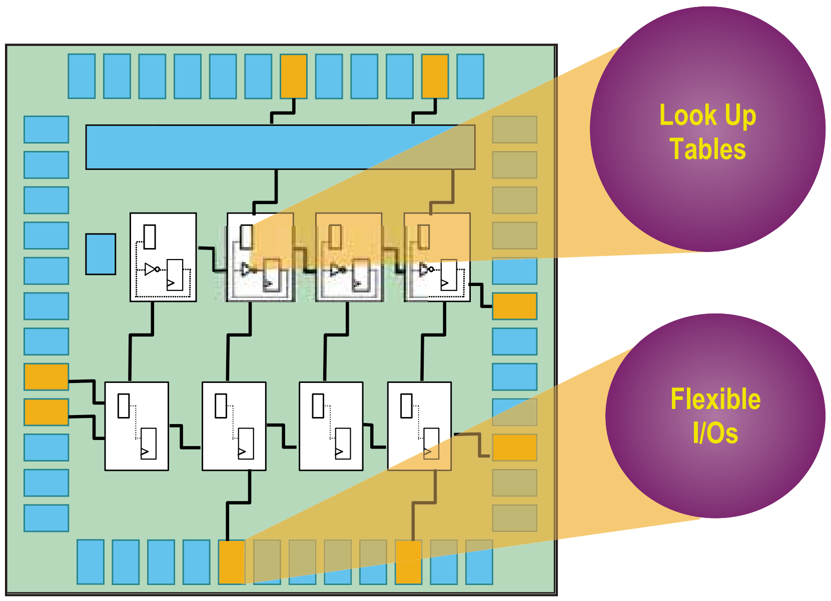


Figure 3 – EasyPath FPGAs allow ASSP customers to fix minor design bugs by changing LUTS and I/Os even after they are in high-volume production.

for certain I/Os. This new level of customer-specific flexibility offers a unique blend of FPGA-like features at ASIC-like prices.

Another important issue is the availability of qualified, reliable IP. ASSP vendors can take advantage of the vast portfolio of IP offered by FPGA vendors during the prototyping phase. When the design is ready for production, you can use the same IP without any additional charges or qualification effort. ASIC solutions, on the other hand, require ASSP vendors to either maintain their own portfolio of state-of-the-art IP or incur large expenditures in acquiring/qualifying standard IP blocks.

Case Study: Consumer Electronics

Companies developing consumer electronic products increasingly find themselves having to incorporate more and more functionality while maintaining lower costs. In no other area are these trends more evident than in home networking, which involves the aggregation and distribution of content within a home through wireless modems, cable, xDSL, and satellite set-top boxes, among others.

In these applications, the ability of chipset vendors to integrate various standards and encrypt/decrypt data streams is critical. Xilinx Spartan-3 FPGAs (and their EasyPath analogs) support multiple I/O standards such as Ethernet, IEEE-1394, USB2.0, and PCI, allowing you to develop designs that are either feature-rich or segmented by end usage without any additional effort. The Spartan-3 family also offers an extensive library of encryption cores such as AES, DES, and TDES for applications where security is important.

In addition, Spartan-3 EasyPath FPGAs support various SSTL and HSTL I/Os for interfacing to different high-speed memories such as SRAM and DRAM, as well as reduced swing differential signaling (RSDS), with applications in LCD TVs and other display products where power consumption is critical.

Finally, for applications that require computational capability like image/video processing, the Spartan-3 EasyPath FPGA's MicroBlaze™ soft-processor cores bring the MIPS cost down below $0.02 per DMIP, while at the same time keeping the core area to about 1% in an XC3S5000. This provides you with low-cost processing power while leaving enough room for other peripherals. For example, at a unit cost of $12.95 (the 50K unit price) for an EasyPath XC3S1500 device, you can prototype with confidence using Spartan-3 standard FPGAs and convert to EasyPath FPGAs for high volume.

Conclusion

Power, performance, and cost have ranked high among the reasons why ASSP vendors have historically migrated to ASIC solutions over FPGAs. But recent advances in process technologies have significantly mitigated these concerns to a point where solutions like EasyPath FPGAs have become viable high-volume ASSP alternatives.

The use of triple-oxide technology, for example, has allowed Xilinx to selectively vary gate-oxide thicknesses to optimize specific regions for low power or high performance. As a result – and contrary to industry trends – static power consumption in the Xilinx 90 nm Virtex-4 family is about 50% lower than in 0.13µ FPGA devices.

On the performance front, the use of specialized embedded hard macros for processor cores and DSP slices allows Xilinx to provide operating frequencies on par with ASICs. Even on the cost front, the move to smaller process nodes and the development of clever techniques to improve yields has allowed FPGA vendors to shift the crossover point between FPGAs and ASICs to higher and higher volumes.

ASSP vendors are being squeezed from both ends – by market uncertainties and risk on one side and rising product development costs on the other. EasyPath FPGAs offer a way out. They eliminate the risk of a new product introduction by significantly reducing the total cost of development and the time to production. ASSP vendors can now address many more markets and segments that might otherwise have been uneconomical.

Four Screamers

...four screaming fast C6416 DSPs, that is

Radio frequency signal processing solution

- **Four, 720 MHz TMS320C6416 DSPs** each with:
  - 64MB SDRAM
  - Dedicated 200 MB/s links between DSPs and FPGAs
- **Two, 4M Gate User-Programmable FPGAs** each with:
  - 18MB private DDR SBSRAM
  - 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 12Gb/s
  - StarFabric PICMG2.17 Compliant Port

Quadia

Download Complete User Manuals NOW!

Innovative Integration
... real time solutions!

www.innovative-dsp.com • 805.520.3300 phone

Applications of the Spartan Family in Flat-Panel Displays

A full feature set and low prices make Spartan-3 devices attractive for use within the flat-panel display market.

by Danny Mok
Product Marketing Manager (High Volume)
Xilinx Hong Kong
danny.mok@xilinx.com

With its combination of low cost and full feature set, the Xilinx® Spartan™-3 family is uniquely positioned to enable various digital consumer systems. One market segment where Spartan-3 devices are being widely accepted is the flat-panel display (FPD) market.

Flat-panel displays are the fastest growing segment of a new breed of consumer televisions. There are three types of flat-panel displays on the market: LCD, plasma, and projection (DLP). All three types are poised to dramatically increase unit shipments – see Figure 1 from iSuppli Research.

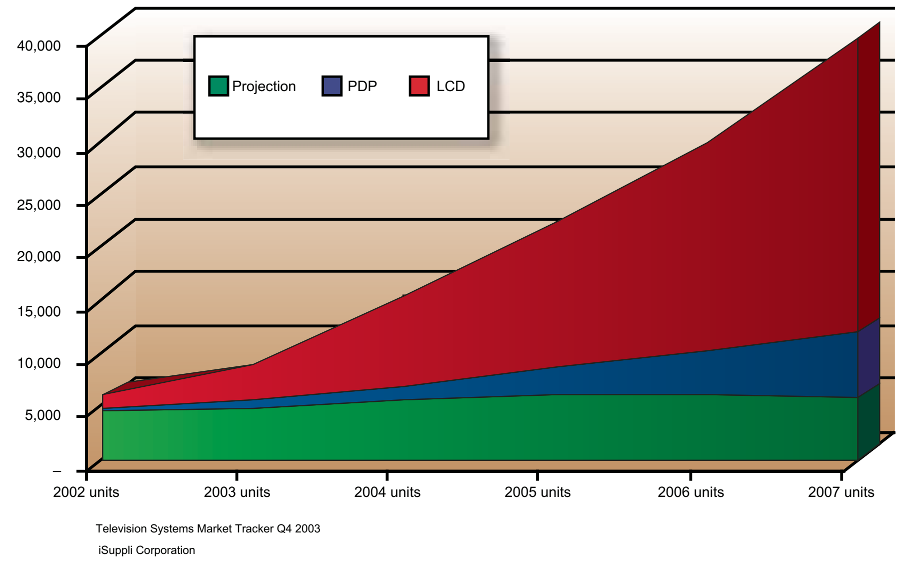


Figure 1 – Most industry experts agree on the explosive growth projections for flat-panel displays.

The Spartan-3 Feature Set

Spartan-3 FPGAs offer various features that are very useful for FPD system designers. These include embedded multipliers that enable very efficient DSP implementation; shift-register functionality features that enable high performance and reduce resource utilization for implementation of pipelined or multi-channel functions; large memory resources; and built-in support for popular differential I/O standards used in the display market. Table 1 lists the various Spartan-3 features and their use in DSP implementation.

A Spartan-3 device can be used in FPD designs in various ways. Let's look at some typical FPGA usage within the FPD system.

Front-End Pre-Processing

The digital RGB signal will typically require some kind of pre-processing before it is fed into the main image processing engine. This pre-processing can be anything from discrete cosine transfer, decryption, or interlacing/de-interlacing. Normally this is how an FPGA is generally used – in conjunction with the customer's image-processing ASIC or ASSP. One common use is to assist the ASSP in image scaling and de-interlacing.

Core Processing

Although core image processing is generally done using standard ASSPs or custom ASICs, FPGAs are sometimes used for custom processing. Xilinx, along with its AllianceCORE™ and XPERTS partners, provides a range of intellectual property cores that can ease the design of these functions. Visit the Xilinx IP Center (*www.xilinx.com/ipcenter/*) for more details.

Spartan-3 Silicon Features	Value in FPD Applications
Embedded Multipliers	Efficient implementation of MAC FIR and other DSP Functions
Shift Register Logic	Efficient implementation of multi-channel functions
Block RAM and Distributed RAM	Video Line Buffers, Cache Tag Memory, Scratch-Pad Memory for DSP co-efficients, Packet Buffers, FIFOs
Built-In Support for RSDS	RSDS support without termination resistors, other special design considerations

Table 1 – Spartan-3 features allow for an area-efficient design in FPD systems.

Examples of such functions include gamma correction, image scaling, edge detection, sharpness, contrast, and frame buffer. Given the range of features that Spartan-3 devices provide, you can implement many of these functions with minimal silicon area. Some of the intellectual property cores utilized by our customers include:

- Color Space Converter
- JPEG Codec
- Discrete Cosine Transfer
- MPEG Video Encoder/Decoder

Figure 2 shows the various functions in a typical FPD that you can implement in a Spartan FPGA.

Interfacing

Many times the FPGA is used for interfacing to the screen or to the external memory. Spartan-3 support for low-swing differential I/O standards (such as reduced swing differential signaling [RSDS]), as well as other single-ended standards such as SSTL/HSTL, allows it to be used in such applications.

The FPGA can also be used to implement the timing control unit (TCON) to control the horizontal and vertical pixel displays format. TCON is more or less a vertical and horizontal display counter. The large amount of flip-flops in the Spartan-3 architecture and a built-in carry chain allows for an efficient implementation of this function.

Also, with abundant differential I/O channels, Spartan-3 FPGAs have an efficient architecture for designs that are more I/O-intensive. Figure 3 shows how even a small Spartan-3E device has enough differential I/O resources to drive a large LCD/plasma screen.

In many designs customers require an interface to high-speed DDR external memory. Xilinx provides reference designs and application notes that allow you to build an efficient memory controller within the Spartan-3 fabric.

Conclusion

With some of the industry's lowest price points and full-featured capabilities, a Spartan-3/3E FPGAs is ideal for various low-

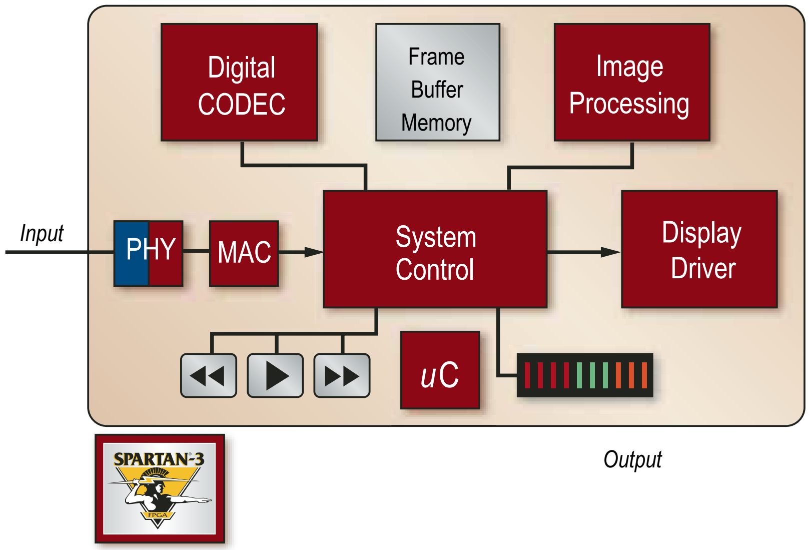


Figure 2 – A typical block diagram of functions found within a display application

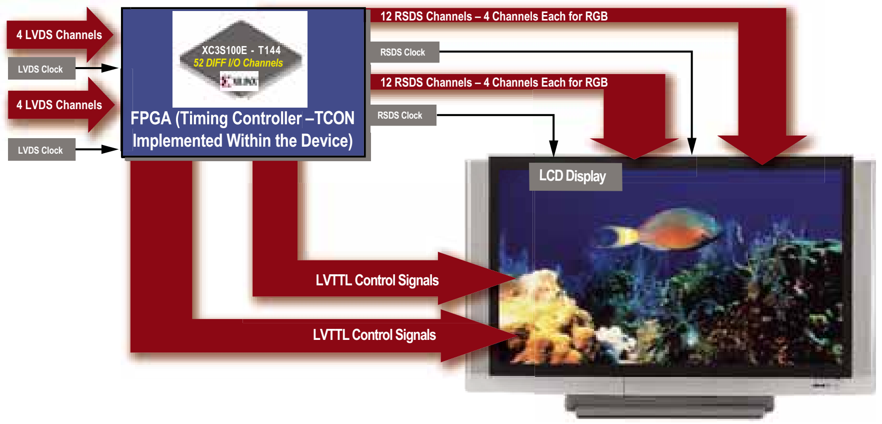


Figure 3 – The large number of differential I/Os allows even small and inexpensive Spartan-3 devices to drive large displays.

cost digital consumer systems. In flat-panel display systems, the Spartan-3/3E architecture is even more useful because it allows high-performance DSP to be implemented in a fraction of the area consumed by competing FPGAs. It also has built-in support for low-swing display I/O standards such as RSDS.

Spartan-3 FPGAs have been used to implement these functions successfully within flat-panel display systems:

- SD/HD color space conversion
- 4:4:4 to 4:2:2 downsampling
- Digital RGB-to-USB card reader function
- Timing control and panel RSDS drivers
- Image compression/decompression

The programmable nature of an FPGA solution reduces the inherent development risk in a new system design. With its host of other features such as multiple I/O banks, on-chip digital clock managers, and extensive block and distributed memory, Spartan-3 devices can also efficiently implement many control/glue logic functions, effectively reducing the size, complexity, and cost of your system.

Let Xilinx help you get your message out to thousands of programmable logic users worldwide.

That's right ... by advertising your product or service in the Xilinx *Xcell Journal,* you'll reach more than 70,000 engineers, designers, and engineering managers worldwide.

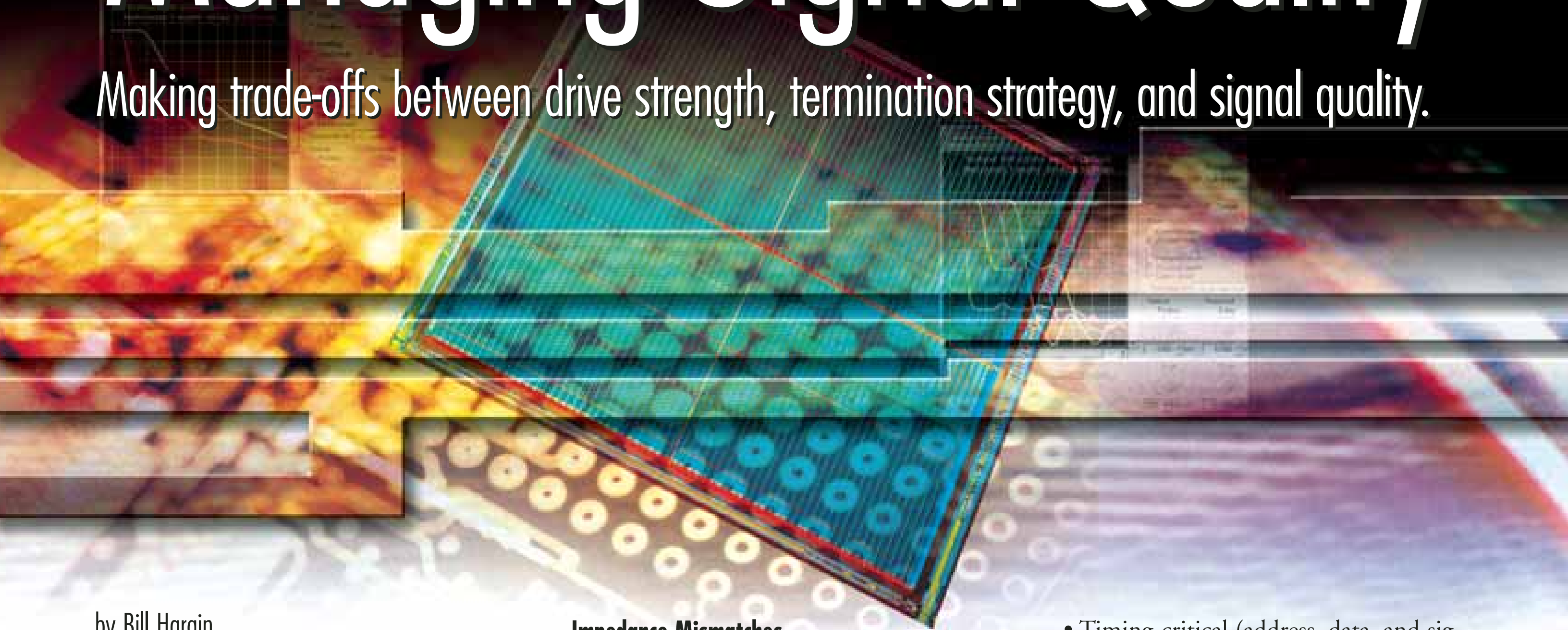
The Xilinx *Xcell Journal* is an award-winning publication, dedicated specifically to helping programmable logic users – and it works.

We offer affordable advertising rates and a variety of advertisement sizes to meet any budget!

Call today :
(800) 493-5551
or e-mail us at
xcelladsales@aol.com

Join the other leaders in our industry and advertise in the Xcell Journal!

XILINX®



Managing Signal Quality

Making trade-offs between drive strength, termination strategy, and signal quality.

by Bill Hargin
Product Manager, HyperLynx
Mentor Graphics
bill_hargin@mentor.com

Driven by advances in lithography, IC switching speeds continue their progressively faster march. At the same time, escalating clock speeds result in much less forgiving timing margins.

The techniques for managing high-speed effects can be broken into three broad categories, sometimes referred to as "the three Ts":

- Technology – selecting driver technology fast enough to meet your functional needs (but as slow as possible)
- Topology – selecting topologies that meet timing requirements while minimizing the impact of signal reflections
- Termination – managing signal reflections using passive components

Sounds easy, right? The problem is that there are thousands of such choices to be made when designing a PCB, and you must balance these against timing requirements and electromagnetic compliance (EMC).

Impedance Mismatches

Reflections can occur at any impedance discontinuity, including variations in board stackup, trace-width variations, ball-grid-array breakouts, stubs, vias, loads, connector transitions, or large power-plane discontinuities.

The significance of a reflection is impacted by several factors, including the impedance difference, the length over which the impedance difference occurs relative to the overall length of the transmission path, and your technology's tolerance for noise.

Some reflections – if significant enough – cannot be resolved using the three Ts. For this reason, careful pre-layout impedance planning with a tool like HyperLynx's Stackup Planning (shown in Figure 1) is a critical part of a proactive impedance-control process.

The First "T": Triage by Technology

Several strategies exist for dealing with non-ideal routing. The first is to know which nets can accommodate poor routing and which cannot. A "technology triage" strategy works well, dividing nets into those that are:

- Signal integrity-critical (clocks, strobes, and signals that require clean edges)
- Timing-critical (address, data, and signals that can have non-ideal edges but must align with timing requirements)
- Signals with driver edge rates faster than 5 ns

A quick look at the effect of fast driver edge rates is instructive. Figure 2 shows the effect of increasing driver edges on the same 5 inch trace. The 10 ns and 5.0 ns drivers produce clean receiver waveforms. The faster 2.5 ns and 1.0 ns drivers, however, produce reflections and ringing on the yellow and red receiver waveforms.

The Second "T": Topology, Signal Integrity, and Timing

Signal-integrity problems tend to disappear when nets are short relative to how fast they are driven, as reflections settle much more quickly. From the fastest 1.0 ns waveform in Figure 2, the reflections eventually smooth out at a half-inch trace length. Although academically instructive, a health-conscious engineer certainly would not want to specify more than a few carefully planned high-speed nets with a half-inch maximum length requirement.

Sometimes, departures from "good practice" routing can actually be a key to resolving signal-integrity problems. Consider the case of a clock with multiple receivers, each of which is skew-sensitive (the clock must arrive at each receiver at close to the same time). In this case, a daisy-chain route may not be ideal because it delivers the signal to each receiver serially, inherently creating skew.

Here, a superior scheme may be a "star" pattern, in which each receiver (or small subsets of receivers) has its own routing branch. Each receiver can be placed at approximately the same delay length from the driver, and each receiver is considerably more isolated from other receivers than on a daisy chain.

Underscoring the interrelationship and trade-offs between the three Ts, however, star routing introduces several new problems. Multiple branches present the driver IC with a low impedance, requiring it to dynamically sink and source significant current. In practice, you may need to use a stronger driver technology for this topology example, such as a Xilinx® Spartan™-3 LVCMOS33_F_24mA driver instead of a LVCMOS33_F_8mA, as shown in Figure 3.

The Third "T": Termination

As a general rule, any line with an edge rate faster than 5 ns on nets running longer than an inch should be considered a termination candidate. Although reducing costs (see sidebar, "Conserve Board Space with Spartan-3 FPGAs") is important, the associated signal quality benefits are critical – impacting whether the product works at all. Let's review some termination strategies for various trace topologies and design requirements.

Termination Types

The classic methods of terminating digital transmission lines are well known. You can terminate the source, the far end, or both; you can employ "distributed" terminations at several locations; or you can use two

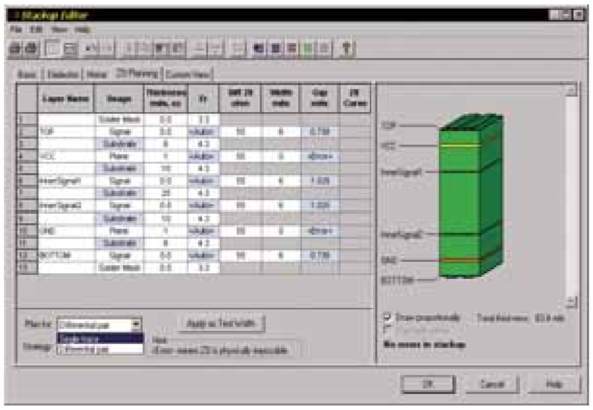


Figure 1 – Careful pre-layout impedance planning using the HyperLynx Stackup Planning tool helps eliminate signal reflections.

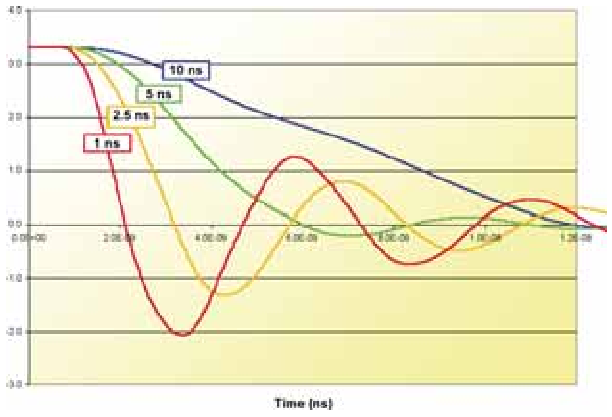


Figure 2 – The effect of driver edge rates on a 5 inch trace. The 10 ns and 5 ns edges look fine, with reflections and ringing effects appearing on the receiver waveforms for the faster 2.5 ns and 1 ns drivers. For sub-nanosecond driver switching speeds, receiver waveforms get progressively worse.

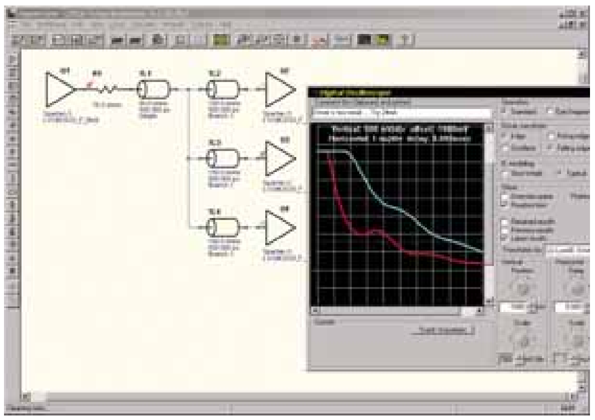


Figure 3 – The HyperLynx oscilloscope and transmission-line schematic show a driver that is too weak for the parallel, low impedance star topology. It would be better in this case to use a stronger 24 mA buffer.

parallel DC terminating resistors pulled to opposing power supplies to achieve a specific DC bias for Thevenin termination.

Here are some general termination guidelines:

- Source termination is useful in point-to-point/one-directional connections
- Far-end termination is useful in multipoint connections
- Distributed termination can be helpful if you have a plug-in system with variable configuration

Each of these techniques has advantages and disadvantages. Parallel DC termination is the simplest, both from the standpoint of component count (only one, built into Spartan-3 FPGAs) and the choice of value (equal to the line impedance). However, it burns the most power and may unacceptably load the driver IC. AC termination requires an additional component (more expensive, extra board space) and more engineering work (finding the optimal capacitor value), but reduces power consumption.

Series termination creates a voltage plateau that persists until a reflection is received back from the end of the line, so series terminators do not work properly from a timing standpoint unless the receiver ICs are clustered near the end of the net, as shown in Figure 4.

There are several ways to terminate at a junction or star connection. One way is to have a series termination at every driver. This has the advantage of reducing settling time at the receiver while consuming a minimum amount of power. Several conditions must be met for a single-series termination strategy to be effective. Each branch must be close to the same length; otherwise, reflections coming back from each branch are not in sync and end up "bleeding" from branch to branch.

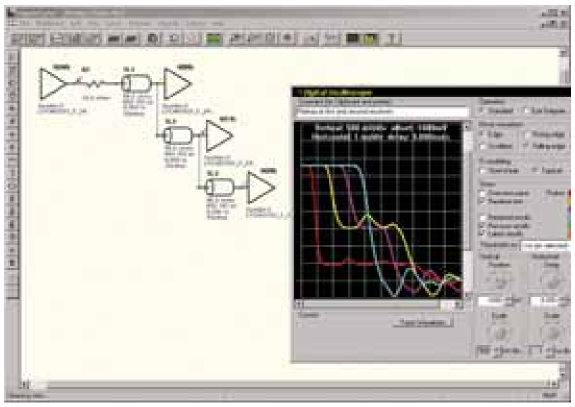


Figure 4 – Series terminators do not work properly from a timing standpoint unless the receiver ICs are clustered near the end of the net, because series termination works by creating a voltage "plateau" that persists until a reflection is received back from the end of the line.

Each branch must also be the same impedance (or close), or it will be impossible to choose an effective single resistor value. If branches are longer than three-quarters of an inch, it makes sense to make their parallel impedances equal to the inbound line impedance from the driver. You can also terminate at the junction itself by changing the trace impedance or by using parallel DC termination – both dampening reflections quickly, and attenuating the signal.

The most appropriate choice depends on network topology and signal direction. For nets that have complex routing patterns, it may be difficult to find a termination scheme that works even in theory. This is where a "what-if" simulation tool like HyperLynx can be an indispensable ally in comparing alternatives.

For a discussion regarding termination component placement, see the "Using Spartan-3 FPGAs to Optimize Termination Component Placement" sidebar.

Conclusion

Spartan-3 devices – with built-in terminators for both single-ended and differential signaling and support for LVTTL, LVCMOS, SSTL, HSTL, GTL, LVDS, and RSDS – are a key enabler for hardware engineers seeking high-speed technology at a reasonable price. But the increased capabilities of modern devices force today's engineers to shoulder the burden of proactively resolving signal integrity, timing, and EMC problems.

With the money you'll save by manufacturing with Spartan-3 FPGAs, consider adding signal integrity analysis software to your toolbox. Several features are important in good analysis software, including the ability to recommend termination strategies and run "what-if" simulations early in the design cycle.

Critical for post-layout analysis are both "interactive" and whole-board "batch" simulation, where violations are flagged and recommendations are made for an entire PCB.

Armed with the latest Xilinx technology, the "three Ts," and the appropriate simulation software, you will not only be ready for today's technology, but ready for what's coming down the road as well. For more information, visit *www.mentor.com/hyperlynx/*.

Using Spartan-3 FPGAs to Optimize Termination Component Placement

Ideally, terminators on a PCB are located at precisely the position they're designed to terminate: exactly at the last receiver IC for parallel termination or exactly at the driver for source termination. Unfortunately, in the real world of dense PCBs, ideal placement is often impossible. Therefore, it's common to find terminators that are located a "stub" length away from their ideal positions. But how long can the stub be before the terminator fails?

Series termination will start to fail when the stub is longer than about 10% of the driver switching time. We can simulate this with a series resistor placed within this guideline and compare the result to a series terminator placed an inch away.

For parallel termination, if the terminator is located upstream from the last receiver on the net, the stub length can be as long as about 15% of the driver switching time.

Fortunately, with Spartan-3 devices, these terminators are located on the FPGA itself, removing the need for placement gyrations and allowing you to focus on using the "three Ts" discussed in this article to address signal quality, timing, and EMC.

Conserve Board Space with Spartan-3 FPGAs

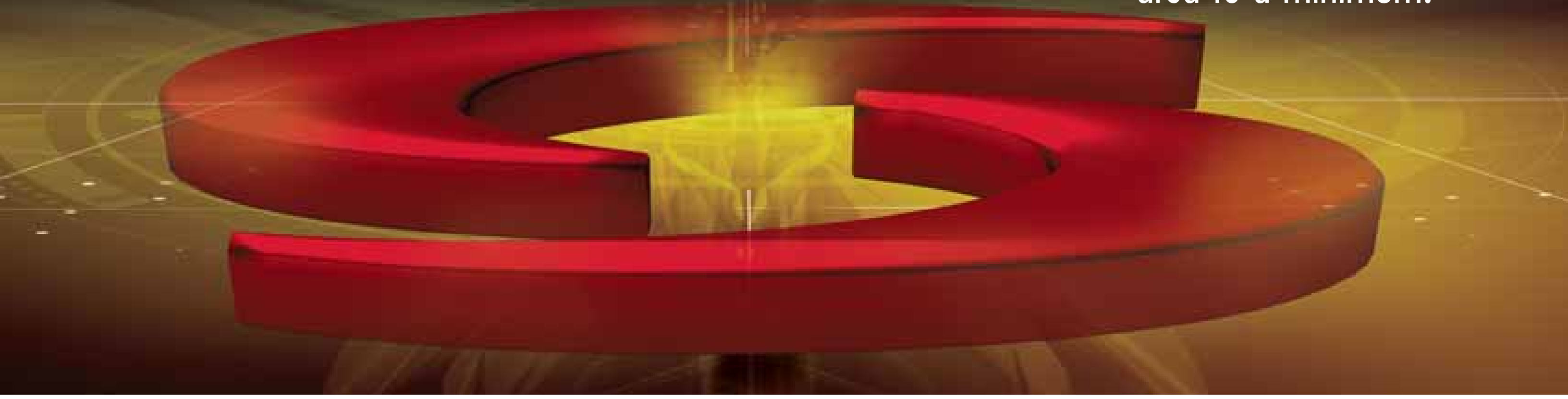
Spartan-3 FPGAs have built-in termination resistors, allowing hardware designers to save significant board space that would have previously been allocated to surface-mount resistors.

Traditional external terminations require two traces for every chip-to-chip connection: a trace for each resistor lead. On-chip termination using Spartan-3 devices cuts this in half, requiring less routing area and possibly reduced layer count.

To get an idea of the benefit, consider a 4 x 6 in, 6-layer PCB with three Spartan-3 FPGAs on it – including a combined routable area of 77 square inches. With previous FPGA technologies, external terminating resistors would consume about 10,000 square mils of board space (40 x 250 mils) each. A board of 4,000 nets (1,333 with termination) would result in a 13.33 square-inch board space savings and a 17% reduction in manufacturing costs.

Meeting Timing and Reducing Area with the Synplify Pro Tool

How to meet timing while still keeping the area to a minimum.



by Steve Elzinga
Senior Product Applications Engineer
Xilinx, Inc.
steve.elzinga@xilinx.com

Steve Pereira
Senior Technical Marketing Manager
Synplicity, Inc.
stevep@synplicity.com

Being the first to deliver a product in a given market greatly increases its chance of success. Effective use of chip resources allows for less expensive parts and a less expensive overall solution.

Synplify Pro software has many timing closure features that increase your chances of using slower speed-grade parts without compromising on quality. The Synplify Pro tool offers a perfect fit for the high-volume, low-cost Spartan™ series of Xilinx® FPGAs.

Identifying Critical Areas

Usually, only a few sections or modules in a design fail timing. It is often a good idea to recognize these structures while coding your design. During coding, the designer usually knows what device the design is targeted for but not the speed grade. Bumping up to the next speed grade can add unbudgeted costs to the project. In this article, we'll outline how to design with performance and area in mind, along with a few tricks-of-the-trade for reducing area.

You should know roughly how many levels of logic the requirements will tolerate before failing timing. Keep this in mind while coding counters, state machines, decoding, and data path logic. When the logic is nearing the first draft, synthesize and place and route; this will give you a glimpse at the problems ahead. If you meet timing, then read further to the area-saving section.

If you don't meet timing, here is a checklist you can go through:

- First and foremost, use the Synplify or Synplify Pro products. Some designers are often unaware of the impact of a quality synthesis tool. If you don't have Synplify or Synplify Pro and are not meeting timing, or you want to reduce area, you can download a full-featured evaluation copy from the Synplicity website for free.
- Is Synplify software reporting positive slack in the log file (.srr)? If the Synplify product's estimates are incorrect (usually because of excessive routing delays caused by congestion), the logic optimizations will be affected. The Synplify solution has to work on the same critical areas as reported by place and route. If there are excessive routing delays, apply the -route constraint to either the clock or the path (see the online documentation for more information).
- Do not use the global frequency box on the front panel of the Synplify tool unless you need to constrain combinatorial paths. Specify all the clocks in the constraints editor (SCOPE). Ensure that unrelated clocks are put into different clock groups.
- If you have clocks that are related and in the same clock group, make sure the periods have a common multiple. If the clocks are slightly different, this can cause very small clock-to-clock setup delays, making timing impossible. This can again have an effect on logic optimizations. For example, if you have two clocks, Clk1 20 Mhz and Clk2 23 Mhz, change to Clk1 20 Mhz and Clk2 25 Mhz.
- Are the I/O constraints correct? Excessive input or output delay can wreak havoc with synthesis. If the critical path includes an I/O and one level of

We hope these guidelines and hints will help you quickly achieve the performance required for your FPGAs while minimizing the logic resources…

logic, there is not much that synthesis tools can do about it. Turn on I/O register packing with the syn_useioff switch.

- Ensure that pipelining is enabled. Try a run with re-timing on. We have found that this gives a ~ 5% performance improvement for Spartan-3 devices. It is also a very good idea to surgically apply the retiming attribute to registers on the critical path with the global switch off. This can increase performance while keeping the area increase to a minimum.
- Add all timing exceptions: false paths and multi-cycle paths. Just adding these constraints can make a huge difference in performance.
- Does the critical path start or end inside a black box or Coregen .edn/.ngc/.ngo? Is it possible to re-code in generic code? If not, add the .edn file to the Synplify project. The Synplify tool will optimize the logic around these cores. If you have .ngc files, use ngc2edf.exe to convert the core to .edn and add to the Synplify project.
- Provide I/O types. Remember to specify the speed and type for I/Os. Synplify software needs this timing information to optimize the driving/driven logic.
- Are there gated clocks in the design? Turn on the gated-clock converter in the Synplify Pro tool. This engine will push the clock from the register input pin onto the clock line while maintaining the same functionality. This will dramatically increase the performance of this path.
- Turn off resource sharing.

Are you now meeting timing? If not, you may want to try Amplify FPGA physical synthesis, which generates detailed placement and performs physical optimizations for additional performance and timing predictability.

How to Reduce Area

Here are few ways to reduce area for Spartan-3 FPGAs. These techniques will have an impact on the timing.

- Use as many of the dedicated resources as possible. Spartan-3 devices have plenty of resources that you can tap into to reduce LUT usage. Here are some suggestions:
- Try to keep black boxes/Coregen to a minimum. Wherever possible replace these components with generic code. Synplify software is capable of optimizing around black boxes, but not the boxes themselves. Significant effort has gone into dedicated resource management for each Xilinx architecture. With generic code, the Synplify product can remove redundant logic, merge identical logic, and pack into the dedicated resources.
- Synplify software may pack logic into registers for performance reasons. If this is undesirable, either loosen the timing constraint or force the logic into RAMs/ROMs/multipliers. Please see the Synplify online documentation for more information.
- There are times when the Synplify tool will not pack logic into the dedicated resources. Common occurrences are:
  - RAMs; either the address bus or the data output must be registers
  - Reset or preset is synchronous/asynchronous, causing failure to map
  - Enable is synchronous/asynchronous, causing failure to map (our online help has examples)
  - Timing constraints are too tight; logic is mapped to registers for timing reasons
  - ROMs are less than half-populated; use the syn_romstyle attribute to force
- Clock tree management. Clock management is extremely design-dependent, so it is difficult to offer generic advice. The idea is to pack each clock quadrant of the chip with as much logic as possible.
- The Synplify tool will build feedback logic for registers without resets. This extra MUX can add to the LUT count. Reset every register in the design if possible.
- If certain paths/modules are clocked by the critical clock but are not critical themselves, either supply a multicycle path or give them another clock line off the DCM. The perfect solution is to have as little as possible high-frequency/critical logic on the chip.
- Experiment with different state machine implementations. Try a run with the Synplify Pro product's FSM Explorer. This is a timing-driven state machine engine that will increase runtime but can often find a better FSM solution for your particular design.
- Turn on resource sharing. Resource sharing tells the tool to share logic such as adders and multipliers whenever possible. This may have an impact on timing, but usually results in fewer resources used.

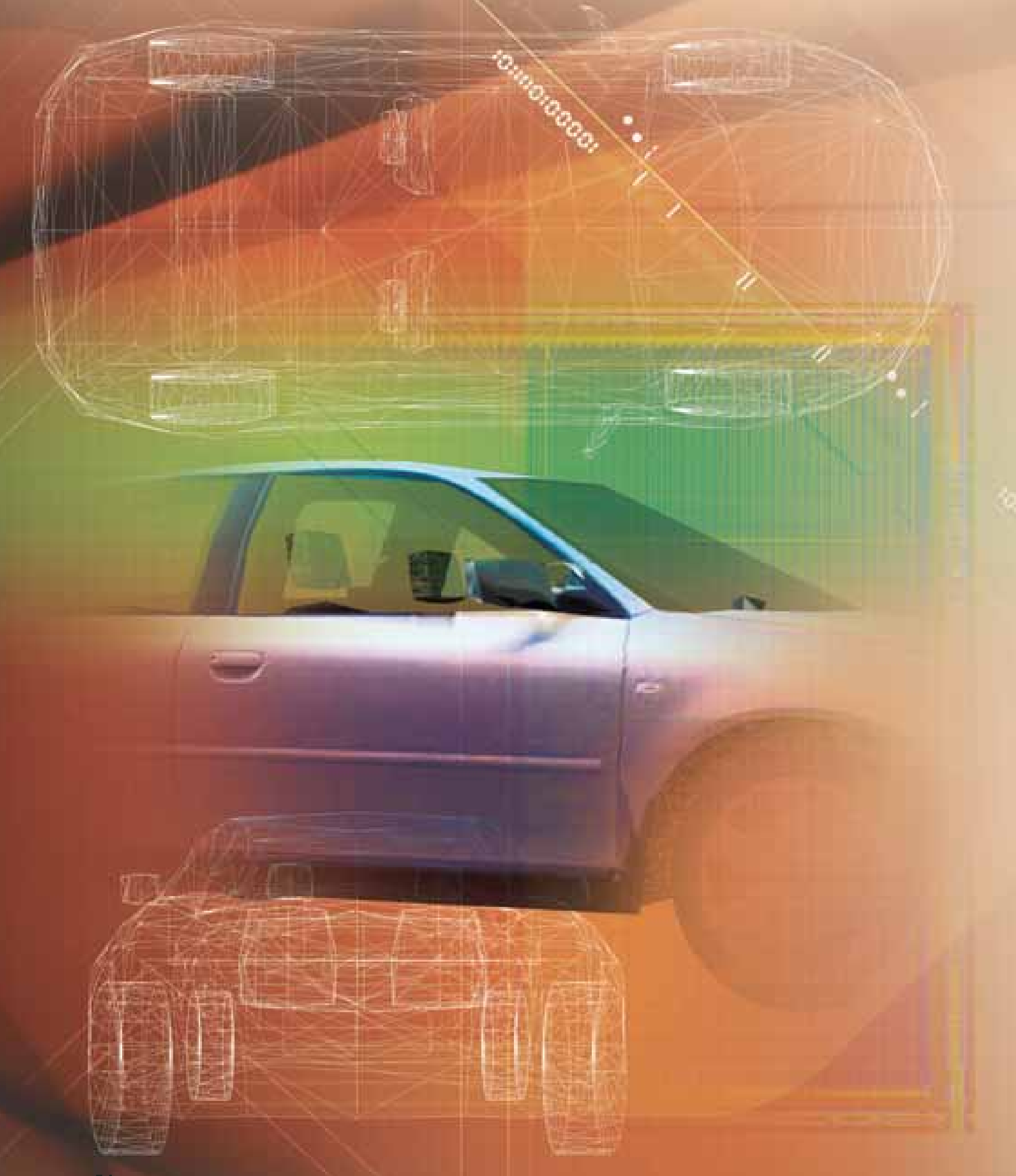
Conclusion

Spartan-3 devices are increasingly being used in higher volume applications. But along with higher volumes comes the requirement that part cost is as low as possible. We hope these guidelines and hints will help you quickly achieve the performance required for your FPGAs while minimizing the logic resources – and therefore cost – of your next design project.

For more information, please contact your local Synplicity sales office, which you can find on our website, *www.synplicity.com.*

The Changing Face of Automotive ECU Design

Why are automotive electronic control unit designers moving away from microcontrollers and microprocessors and looking at programmable logic devices (PLDs) instead?



by Karen Parnell
Automotive Product Marketing Manager
Xilinx, Inc.
karen.parnell@xilinx.com

Microcontrollers and microprocessors have long been the natural choice to implement control functions in instrument clusters, door modules, and other control-based electronic control units (ECUs). Design experience and legacy code are two of the many reasons to use such devices time and time again.

The tendency is to build the software topology in logically partitioned function blocks that meet the ECU specification. But only when these function blocks are compiled and targeted at a single device will we see the real-time issues associated with sequential code. Even though these functions were built discretely, they will run sequentially based on a pre-defined prioritization structure.

You may not even see these timing issues immediately. Any interrupt collision issues will not be caught in simulation because the tendency is to only test what you know should work – and it is almost impossible to test for indeterminate states or real-time fault conditions. Any function that runs off of an interrupt can only really be tested after the system is built up and

The CPLD can interface to the microcontroller through a simple serial bus and provide low-cost inputs and outputs to the devices that need to be controlled.

run in real time. At this point we can catch more of the errors, but only when the product hits the road will every conceivable interrupt state be tried and tested. We have to hope that these are not major failures and that they can be fixed with a software patch at the next scheduled vehicle service.

Function Interdependency

With programmable logic, you can build up a system in the same way as with a microprocessor or microcontroller – by implementing discrete functional blocks. But unlike micro code, these functions are totally independent: instantiated in hardware (as opposed to software), they will run in real time and are not susceptible to erroneous errors or interrupt states.

Each function will operate on its own and is guaranteed to run as described every time, under every condition. Another benefit is that you can simulate every state before building any hardware, so the risks of operations or tasks not functioning as required – or happening out of sequence – are dramatically reduced.

For example, when building an instrument cluster, you may need to display a "tell tale" based on the driver pushing a switch; respond to a CAN packet dictating the position of the tachometer; and handle an error message from the braking system that must be displayed urgently to alert the driver – all at the same time.

In software-based systems, these tasks run sequentially. You must give priority to the error message, but which task runs when and in what order depends on when the interrupt hits the micro.

In a PLD-based system, all of these functions are handled in parallel, displayed as soon as the message hits the device. The parallel processing available in FPGAs and CPLDs offers a great advantage in applications where instant, uninterrupted signals must be processed reliably.

Board-Level Functions

As designs become more complex, more peripheral functions are controlled by 8-bit microcontrollers – but they may not have sufficient I/O. There are a number of solutions to this issue, including stepping up to a more costly 16-bit microcontroller. Another is to partition the design over a low-cost 8-bit microcontroller and low-cost CPLD.

Here's a good example: a system has multiple ADC inputs used as data inputs to help control and position multiple stepper motors (such as in an instrument cluster) plus other I/O for LED control and switch inputs. The low-cost microcontroller is ideal for the sequential signal processing but has limited I/O to interface to off-chip devices.

The CPLD can interface to the microcontroller through a simple serial bus and provide low-cost inputs and outputs to the devices that need to be controlled. Although this approach may increase the board component count, the combined cost of an 8-bit microcontroller and CPLD is lower than that of a 16-bit microprocessor.

This approach also allows the board to be adapted to suit many customer configurations through the CPLD, providing the ability to interface to many different devices under control. The Xilinx® CoolRunner™-II family of CPLDs are not only easy to design with, but offer extremely low power consumption, thus eliminating the need to compromise power or performance. The low-cost XC9500XL devices are ideal if you require 5V-compatible I/O.

Figure 1 shows a typical example of where a CPLD can be used alongside a microcontroller, providing extra I/O to connect to multiple indicator LEDs and also provide I/O to control multiple motors. The CPLD in this example also provides simple PWM functions and ADC interfacing to aid with motor control, which can enable you to use a lower cost, basic microcontroller without PWMs.

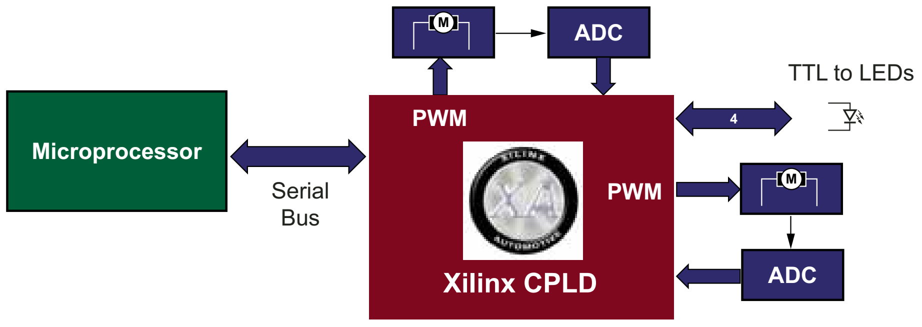


Figure 1 – I/O expansion example

Interfacing and Bridging

With the increasing number of bus protocols supported by ASSP and microcontroller vendors, interface translation is a growing problem that requires an inexpensive and easy solution. Offering the lowest cost method for translating one bus protocol into another, CPLD devices can support interface bridging applications, including voltage-level shifting (3.3V in to 1.8V out), bus translation applications (translating a proprietary language to an industry-standard language), serial-to-parallel and parallel-to-serial bus conversions, and DDR memory interfacing.

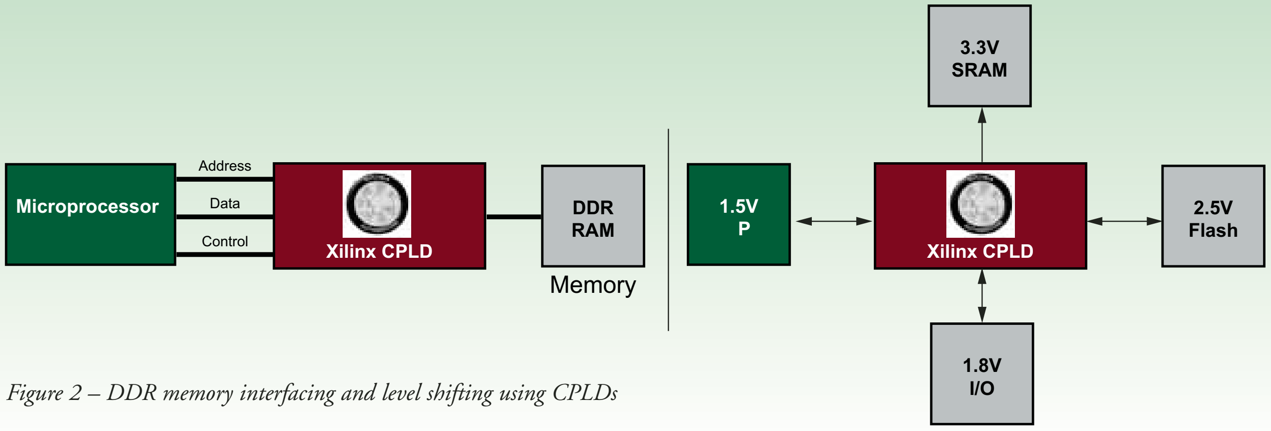


Figure 2 – DDR memory interfacing and level shifting using CPLDs

Figure 2 shows how CPLDs can be used to interface to high-speed memory and also shows an example of a CPLD level-shifting between different voltage levels.

CoolRunner-II CPLDs are ideal for DDR memory interfacing, as they include DualEDGE-triggered registers, a global clock divider, and voltage-referenced I/O standards including SSTL_2. These features provide the capability to interface between a microprocessor and high-speed memory devices such as DDR SDRAM. To make implementing this function easier, Xilinx offers a reference design and application note (XAPP384, "Interfacing to DDR SDRAM with CoolRunner-II CPLDs") for downloading at the Xilinx website (*www.xilinx.com/literature/*). This reference design is particularly useful if your chosen microprocessor does not support HSTL or SSTL memory interfaces.

Conclusion

Automotive design has always been a challenging environment when faced with small-form-factor ECU specifications, wide temperature ranges, high-quality and reliability requirements, and low cost points. Today's designs also need to be flexible, upgradeable, and easy to test fully.

Programmable logic is a viable alternative to microprocessors and microcontrollers because they offer true design interdependency and parallel processing. CPLDs can be used to great effect where microcontrollers have run out of I/O or processing power, or simply for memory interfacing or voltage-level shifting.

Xilinx Automotive Devices

Designing electronics systems for automobiles has always been challenging. With automotive electronics comprising as much as 90% of the functional innovation in new vehicles, it has become even more important to choose the right devices to meet temperature, quality, and technological requirements.

The Xilinx Automotive (XA) range of FPGAs, CPLDs, and configuration devices not only offer the design flexibility and time-to-market advantage associated with programmable logic devices, but also meet the extended temperature requirements and quality standards required in today's automotive advanced electronics systems.

The XA device range is offered in both the extended temperature Q-grade (-40º C to +125º C) and industrial I-grade (-40º C to +85º C) and is qualified to the industry-recognized AEC-Q100 standard (for more information, visit *www.aecouncil.com*). The XA range includes the lowest power CPLD devices available (CoolRunner-II CPLDs), the lowest cost FPGA (Spartan™-3 devices), and Platform Flash configuration memory devices (Table 1).

For more information, visit *www.xilinx.com/automotive/*.

FPGA/CPLD	Packages	Density Range	I/O	Standards Supported	Multipliers	Block RAM (Kbits)
Spartan-3	VQG100, TQG144, PQG208,FTG256, FGG456, FGG676	50K-1500K system gates	487	23	32	576
Spartan-IIE	TQ144, PQ208, FT256, FG456	50K-300K system gates	329	19	–	64
CoolRunner-II	CPG56, CPG132, VQG44, VQG100, TQG144	32-256 macrocells	118	5	–	–
XC9500XL	CS144, VQ44, VQ64, TQ100, TQ144	36-144 macrocells	117	–	–	–
Flash PROM	VOG20	1-4 Mb	–	–	–	–

Table 1 – The Xilinx Automotive family of CPLDs, FPGAs, and configuration memory devices

A Multimedia Platform for Automotive and Consumer Markets

Xilinx FPGAs and Xylon IP cores shorten design cycles and lower production costs for multimedia applications.

by Davor Kovacec
CEO
Xylon d.o.o.
dkova@xylon.hr

Market leadership in the fast-moving electronics market continuously demands more innovative and cost-effective products. With an ever-shrinking time-to-market window, design tools and prefabricates are important elements of success.

But the right development platform can make all the difference between success and failure, harmonizing the contradictory requirements of being standard and highly configurable at the same time.

Combining Xilinx® FPGAs with the Xylon logicBRICKS IP cores library, Xylon's feature-rich Multimedia FPGA Platform is ideal for addressing the time-to-market and flexibility needs of the high-volume customer base. You can quickly turn system designs running on this generic FPGA development platform into specialized products. Such a design approach retains a large portion of design reuse through different hardware (IP cores) and software modules. You can reuse these same modules in many system designs for different applications.

Multimedia FPGA Platform

The basic functional blocks of the Multimedia FPGA Platform are output to displays, with inputs provided from video, human machine interface, and communication interfaces. These functional blocks support a variety of different displays, video input types, input devices, or communication interfaces.

The Multimedia FPGA Platform is an FPGA design based on hardware modules in the form of IP cores. You can add additional features using third-party IP cores or by designing a customer-specific circuit.

The main components of the Multimedia FPGA Platform are:

- **logiCVC** – a compact video controller for display driving
- **logiBITBLK** – for 2D graphics acceleration
- **logiCAN/logiUART** – for communication
- **UltiWIN** – for frame grabbing (video input)
- **UltiMEM** – a multi-port SDRAM/DDRAM memory controller

The Spartan™-3 hardware resources, dedicated multipliers, digital clock managers (DCMs), and block RAMs ensure that IP cores operate at higher speeds and consume less area of the FPGA. The video input scaling circuits take advantage of the dedicated 18 x 18 multipliers, while the high-capacity block RAMs used in all Xylon IP cores contribute to more efficient sharing of memory bandwidth and better overall system performance.

In addition, the DCMs provide finer clock generation and eliminate the need for costly clock-generation ASSPs, while digital termination and DDR support enable better and lower cost support for SDRAM or DDRAM memories.

Figure 1 shows the Multimedia FPGA Platform block schematic configured for an automotive backseat entertainment application. This example is significant because it has the capability to show a live video stream on any display screen. We accomplished this by instantiating more than one

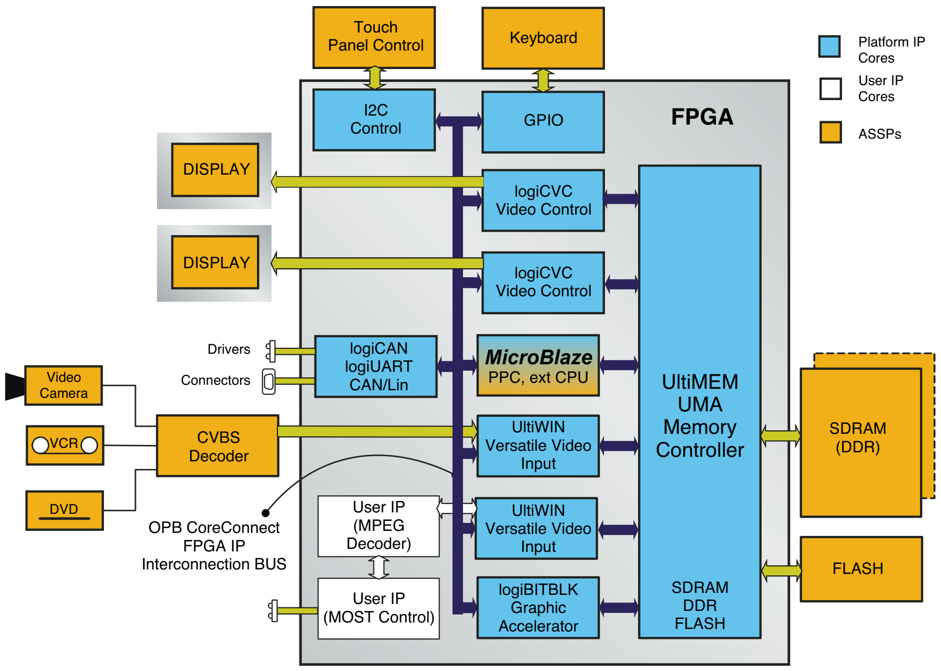


Figure 1 – Multimedia FPGA Platform: automotive back seat entertainment

logiCVC for multiple display driving and instantiating more than one UltiWIN for simultaneous video input streaming.

A DVD, VCR, or camera CVBS output is used as the first video source, while a MOST network is used as the second. The CVBS analog signal is converted to an ITU656 digital signal by a composite video signal decoder, an ASSP component. The MOST MAC and MPEG decoder are customer-added IP cores. The I2C and GPIO IP cores are used for the keyboard and touch controller interface. The UltiMEM IP core is configured for unified memory architecture and six access ports. The 32-bit DDRAM assures 800 MBps bandwidth for display refresh, video streaming, graphics acceleration, and CPU program execution. The logiCAN and logiUART-local interconnect network (LIN) IP cores are used for network connections with in-car body electronics.

Memory Bandwidth Requirements

The selection of memory bandwidth is important for system performance. The six IP cores share common memory, as shown in Figure 1. Two logiCVC driving displays of 400 x 234 resolution and 16 bpp color depth require 16 MBps each. Two UltiWIN processing CVBS video sources require 54 MBps each. The Xilinx MicroBlaze™ soft-processor core running at 100 MHz requires 400 MBps, while the logiBITBLK, processing 16 bpp images and running at 50 MHz, requires 100 MBps. This gives a total bandwidth requirement of:

2 x 16 MBps + 2 x 54 MBps + 400 MBps + 100 MBps = 640 MBps

The overall memory bandwidth for a 32-bit DDRAM running at 100 MHz is 800 MBps. This leaves 160 MBps of spare memory bandwidth that can be used for efficient memory arbitration and access to memory without stalls.

System Gate Count

Table 1 shows the size of each IP core and the total number of slices required for the backseat entertainment FPGA design. Every single IP is compact with a low gate count, resulting in a lower cost solution.

The complete backseat entertainment system is a complex multimedia application that fits into a Spartan-3 XC3S1000 device, leaving approximately 2,000 slices for extra FPGA circuits. A separate display controller application driving a single display, including 2D graphics acceleration

IP Core	Description	FPGA Slices
MicroBlaze	Xilinx 32-bit RISC CPU	530
logiCVC	Xylon Compact Video Controller	2 x 615[1]
UltiMEM	Ultimodule Multiport SDRAM/DDRAM Control	410[2]
logiBITBLK	Xylon 2D Graphic Accelerator	765
GPIO	Xilinx General I/O	40
INT Control	Xilinx OPB-INT Control	40
UltiWIN	Ultimodule Versatile Video Input	2 x 960[1,2]
logiCAN	Xylon CAN 2.0B Control	430
I2C	Xilinx OPB-I2C Control	200
logiUART	Xylon FIFO UART Control (LIN Bus)	225
Total		**5,876**

Table 1 – IP core size

Note: 1. Back seat entertainment applications require two video inputs and video outputs, such as two logiCVC and two UltiWIN.

2. Xylon is an Ultimodule Inc. technology partner. Xylon has an open license to tailor UltiMEM and UltiWIN IP source code to customer needs.

and a UMA memory architecture, fits into a smaller Spartan-3 XC3S200 device.

We reduced costs further by using the MicroBlaze core, which eliminates the need for a separate external CPU. The IP core integration and system configuration for the MicroBlaze core is provided by the Xilinx Embedded Development Kit (EDK).

End Applications

The Multimedia FPGA Platform is primarily aimed at the automotive market, including applications such as navigation, infotainment, backseat entertainment, and driving assistance.

Example infotainment applications include such automotive displays as VCR videos, game consoles, rear-parking cameras, and navigation systems. Figure 2 shows a backseat entertainment prototype system displaying a contemporary game console on one display and a DVD movie on another display.

Driving assistance is an another example of an automotive application using multimedia platform IPs. The system comprises synchronized stereo video camera inputs, a DSP algorithm for image processing, and CAN IP for communication with dashboard or other human interfaces.

Xylon is in the process of designing an automotive reference board, which provides the ability to prototype or evaluate the Multimedia FPGA Platform by connecting displays, input devices, vehicle communication buses, and video signal sources.

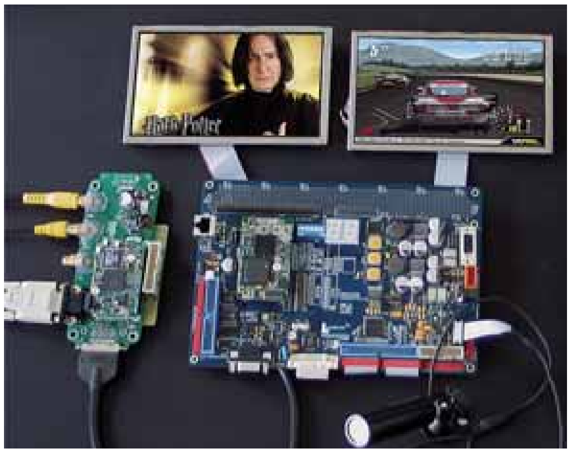


Figure 2 – Backseat entertainment prototype

Other multimedia applications are equally suitable, such as consumer, medical, and measurement instrumentation or factory automation.

Today, automotive OEM and first-tier manufacturers have used Xilinx FPGAs and Xylon IPs to develop infotainment systems.

Conclusion

The value of the Multimedia FPGA Platform from Xylon is in the high number of different displays tested; the broad feature set; and the ability to configure for performance, price, low development, and production cost, all with virtually no obsolescence.

Xilinx FPGAs, and in particular the Spartan-3 family with its low cost-per-gate and rich architecture, provided an excellent base on which to build. Combining Spartan-3 devices with Xylon logicBRICKS IP cores provided the ideal combination of feature set, form factor, performance, and time-to-market capabilities for the development of the Multimedia FPGA Platform.

For more information about the Multimedia FPGA Platform and logicBRICKS Xylon IP library, please contact us at *sales@logicbricks.com*, or visit *www.logicbricks.com*.

Multimedia FPGA Platform Features

- Displays supported: wide-resolution range 128 x 64 to 1280 x 1024 pixels; wide range of color depths 1- to 24-bit bpp, TFT/STN and other flat panel displays; 8- to 32-bit memory interface, overlays, multiple display support
- Acceleration: ROP2 operation, block fill, transparency, color expansion, pattern operations
- Memory interface: flexible multiport memory controller, unified memory architectures for cost-sensitive applications, priority, round-robin and mixed-port arbitration policy, configurable data width for access port and configurable memory bus
- Video input: YUV-RGB and de-interlace, picture position, picture cropping, picture scaling using multi-pass bi-linear interpolation, multiple video input source support
- Communication: CAN controller and UART improved for LIN bus

A Low-Cost PCI Express Solution

Spartan FPGAs are ideal for next-generation PCI applications and systems.

by Abhijit Athavale
Product Marketing Manager
Xilinx, Inc.
abhijit.athavale@xilinx.com

PCI has been the most widely used bus standard in the PC, server, and embedded markets for the past decade. Because PCI is limited by its shared, central arbitration-based architecture and system-synchronous clocking scheme, current and next-generation processors are outstripping its ability to keep up.

PCI's emerging replacement is PCI Express, a new connectivity standard that preserves the flexibility and familiarity of PCI while dramatically increasing bandwidth and performance. The controlling body for the PCI specification, the PCI SIG, has ratified PCI Express as the next-generation PCI. PCI Express-based products are now becoming available; shipments are expected to achieve high volume as early as 2006. Figure 1 shows the adoption forecast for PCI Express.

PCI Express uses serial I/O technology to create point-to-point connections and is reverse-compatible to PCI, preserving many original PCI advantages. It scales from a single lane (1x) to a 32 lane (32x) architecture, offering a bandwidth of 2.5 Gbps per lane. PCI 32/33 has a bandwidth of 1 Gbps, while PCI 64/66 has a bandwidth of 4 Gbps.

The 1x PCI Express implementation matches up very well with PCI 32/33, the most commonly used PCI interface across all markets. A two-lane implementation (5 Gbps) is an incremental improvement over PCI 64/66. At the high end, a 32-lane PCI Express implementation supports a total of 80 Gbps, providing more than enough bandwidth to support the vast majority of next-generation applications.

Implementation Details

PCI Express is a three-layer specification: physical (PHY), logical, and transport, all defining separate functionalities. Also included in the specification are advanced features for hardware error recovery and system power management. (For more information about PCI Express, visit *www.pcisig.com*.)

Since 2000, Xilinx® has offered a line of PCI 32- and 64-bit solutions for Spartan™ series FPGAs. The most logical successor is a PCI Express solution using an external PHY chip paired with a Spartan-3 or Spartan-3E device. The PCI Express specification defines an interface to hook a PHY chip up to a separate device that houses the logical and transport layers (called a PIPE interface – a white paper about this is available from Intel).

In the two-chip solution, the transport layer resides in a dedicated PHY chip, and the logic and transport layers reside in a Spartan FPGA. A broad range of PHY devices are available from manufacturers such as Genesys Logic, Philips Semiconductor, and Texas Instruments. PHY pricing will be less than $10 for high volumes (250,000 units per year). (See the sidebar, "PHY Vendors," for contact information.) Xilinx has collaborated with Phillips Semiconductor and delivered this solution to our customers.

To implement the interface, Xilinx and several of our IP partners (including Eureka, GDA, and Northwest Logic) provide PIPE IP cores for Spartan-3 and Spartan-3E devices. A single-lane PCI Express controller requires approximately 500,000 gates (50% of a Spartan XC3S1000) for the logical and transport layer core, leaving the rest of the FPGA available for the user application (see

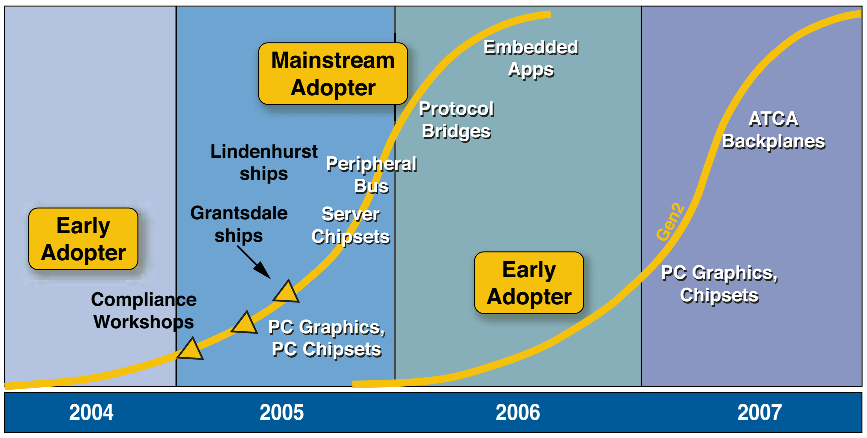


Figure 1 – PCI Express adoption forecast

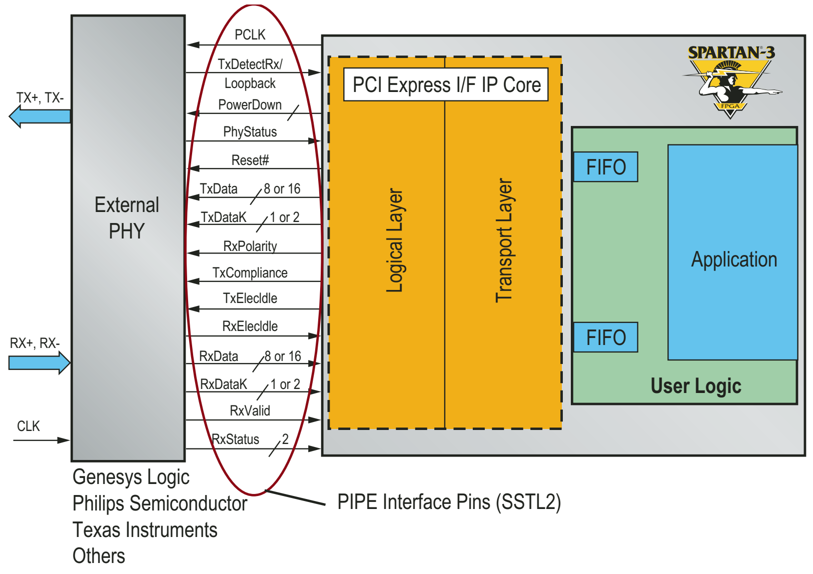


Figure 2 – PIPE interface between a Spartan FPGA and an external PHY

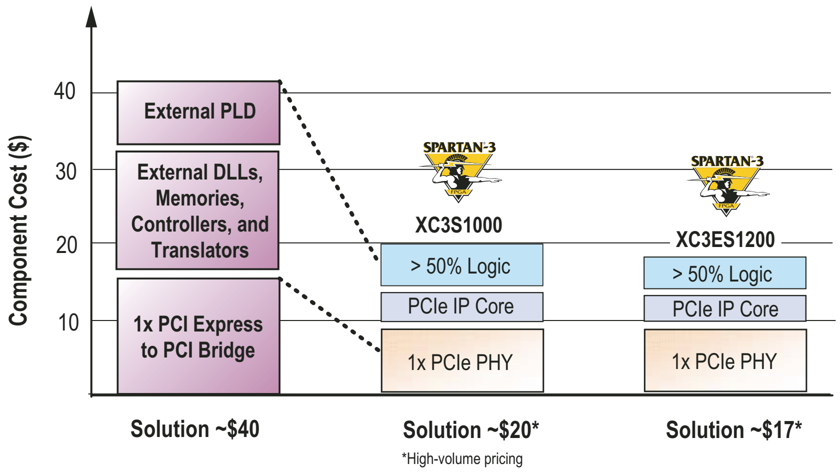


Figure 3 – Single-lane PCI Express implementation options

the "PCI Express Core IP" sidebar for details on Northwest Logic's product and *www.xilinx.com/pciexpress/* for details on PCI Express IP from our other IP partners.) Figure 2 shows the implementation of a PIPE interface using a Spartan FPGA and external PHY.

Figure 3 illustrates a range of options to implement a single-lane PCI Express interface. The cost of a standard-product option is fairly high (>$40), making it tenuous for high-volume/low-cost applications. The Spartan options drop that cost substantially, and add the flexibility of programmable logic to integrate and implement other system capabilities. In 250K quantities (reasonable for typical consumer applications), the Spartan-3E version will cost approximately $17.

Conclusion

In addition to reducing total costs, the Spartan FPGA + PHY option gives you substantial flexibility to build "PCI Express-to-anything" bridges and integrate other circuit elements. As most systems have a range of bandwidth requirements, preserving flexibility is important so that you can add lanes without dramatically changing the layout.

Spartan-3 and Spartan-3E FPGAs are available in a wide range of densities, and preserve migration up and down in overall bandwidth. And because FPGAs are fully reprogrammable post-deployment, they eliminate the risks associated with first-generation ASSPs and ASICs.

If you are currently using PCI for your interconnect standard and are architecting your next-generation designs, you should consider the PCI Express option from Xilinx. We encourage you to find out how Spartan-3 and Spartan-3E FPGAs will help you meet your current and future design requirements. More information about Spartan-3 and Spartan-3E FPGAs, PCI Express IP, and compatible PHY devices is available at *www.xilinx.com/pciexpress/*.

PCI Express IP

PCI Express IP cores are available from multiple vendors including Xilinx and our partners. One such core from Northwest Logic is featured below.

Northwest Logic's PCI Express Core is specifically designed for low-cost Spartan-3 FPGAs. A Spartan-3-based PCI Express design uses the Spartan-3 device with a low-cost physical interface for a PCI Express (PIPE)-compatible PHY chip. The PHY chip implements the low-level PCI Express physical layer, while the device takes care of the upper-level data link and transaction layers.

Another version of the PCI Express Core uses the internal MGTs in Virtex-II Pro and Virtex-4 FX FPGAs to provide a fully integrated PCI Express solution.

Northwest Logic's PCI Express Core is one of the smallest PCI Express cores available, enabling you to target the smallest and consequently lowest cost FPGA. The core is provided with a comprehensive verification suite and expert support to ensure rapidly developed and validated designs.

Also available is a PCI Express Development Board for quickly prototyping a complete PCI Express System. A demo GUI, drivers, and PCI Express FPGA reference design are also included.

For more information (including pricing and core size for a particular FPGA family), visit the Northwest Logic website at *www.nwlogic.com*.

PHY Vendors

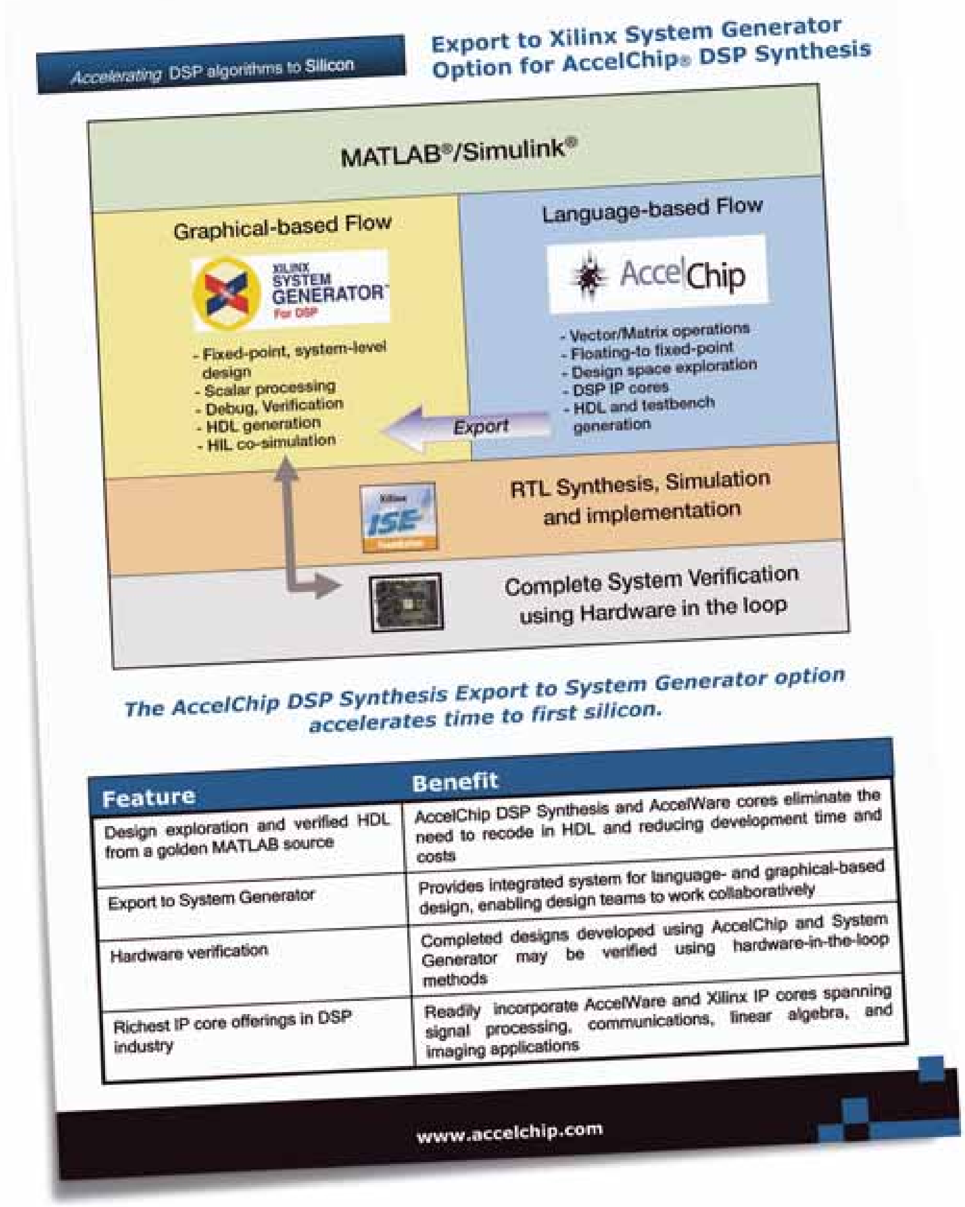
Genesys Logic
www.genesysamerica.com

Philips Semiconductor
www.semiconductors.philips.com

Texas Instruments
www.ti.com/pciexpress/

STOP OVER-THE-WALL ENGINEERING.

(FINALLY, ALGORITHM DEVELOPERS, DSP SYSTEM ARCHITECTS AND HARDWARE DESIGNERS ARE ALL ON THE SAME PAGE)



Accelerating DSP algorithms to Silicon

Export to Xilinx System Generator Option for AccelChip® DSP Synthesis

MATLAB®/Simulink®

Graphical-based Flow

XILINX SYSTEM GENERATOR™ For DSP

- Fixed-point, system-level design
- Scalar processing
- Debug, Verification
- HDL generation
- HIL co-simulation

Language-based Flow

AccelChip

- Vector/Matrix operations
- Floating-to fixed-point
- Design space exploration
- DSP IP cores
- HDL and testbench generation

Export

Xilinx ISE

RTL Synthesis, Simulation and implementation

Complete System Verification using Hardware in the loop

The AccelChip DSP Synthesis Export to System Generator option accelerates time to first silicon.

Feature	Benefit
Design exploration and verified HDL from a golden MATLAB source	AccelChip DSP Synthesis and AccelWare cores eliminate the need to recode in HDL and reducing development time and costs
Export to System Generator	Provides integrated system for language- and graphical-based design, enabling design teams to work collaboratively
Hardware verification	Completed designs developed using AccelChip and System Generator may be verified using hardware-in-the-loop methods
Richest IP core offerings in DSP industry	Readily incorporate AccelWare and Xilinx IP cores spanning signal processing, communications, linear algebra, and imaging applications

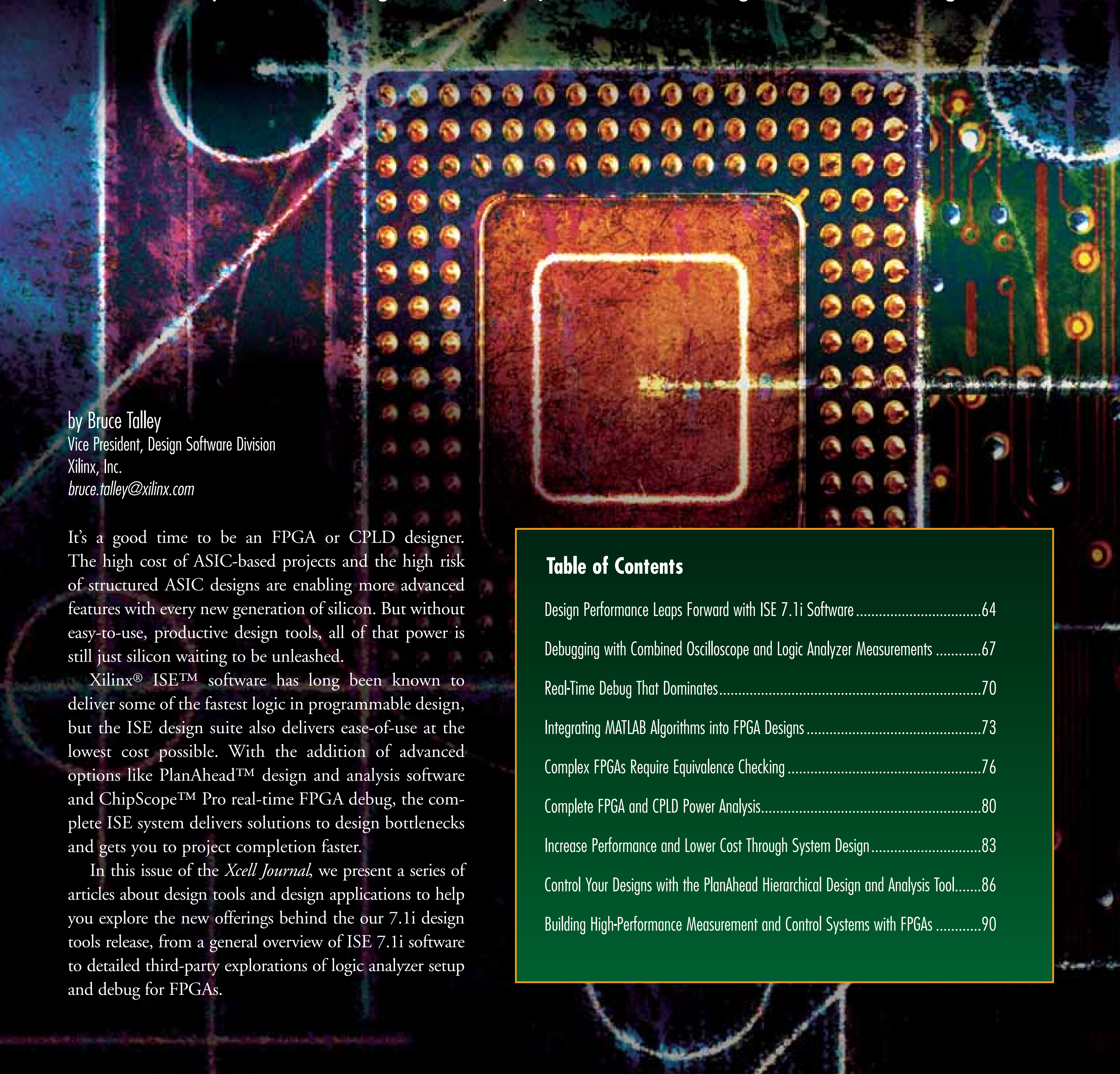
www.accelchip.com

Company partnerships don't usually generate a lot of interest, but a partnership that yields team-building technology does. For example, the industry's first DSP design flow from a mixed MATLAB/Simulink environment to a verified FPGA. It's based on a tight integration between AccelChip DSP Synthesis and Xilinx System Generator for DSP that accelerates development of signal processing systems. Work more collaboratively and get designs to market faster by eliminating the need to recapture in HDL. Readily incorporate AccelWare and Xilinx IP cores and verify the entire system using hardware-in-the-loop simulation, further speeding development. How's that for teamwork? **Stop your Over-the-Wall engineering. Download free whitepapers at accelchip.com/papers or call (408) 943-0700.**

XILINX® AccelChip

Design Leadership from Xilinx

The shortest path from design start to project finish is through Xilinx 7.1i design tools.



by Bruce Talley
Vice President, Design Software Division
Xilinx, Inc.
bruce.talley@xilinx.com

It's a good time to be an FPGA or CPLD designer. The high cost of ASIC-based projects and the high risk of structured ASIC designs are enabling more advanced features with every new generation of silicon. But without easy-to-use, productive design tools, all of that power is still just silicon waiting to be unleashed.

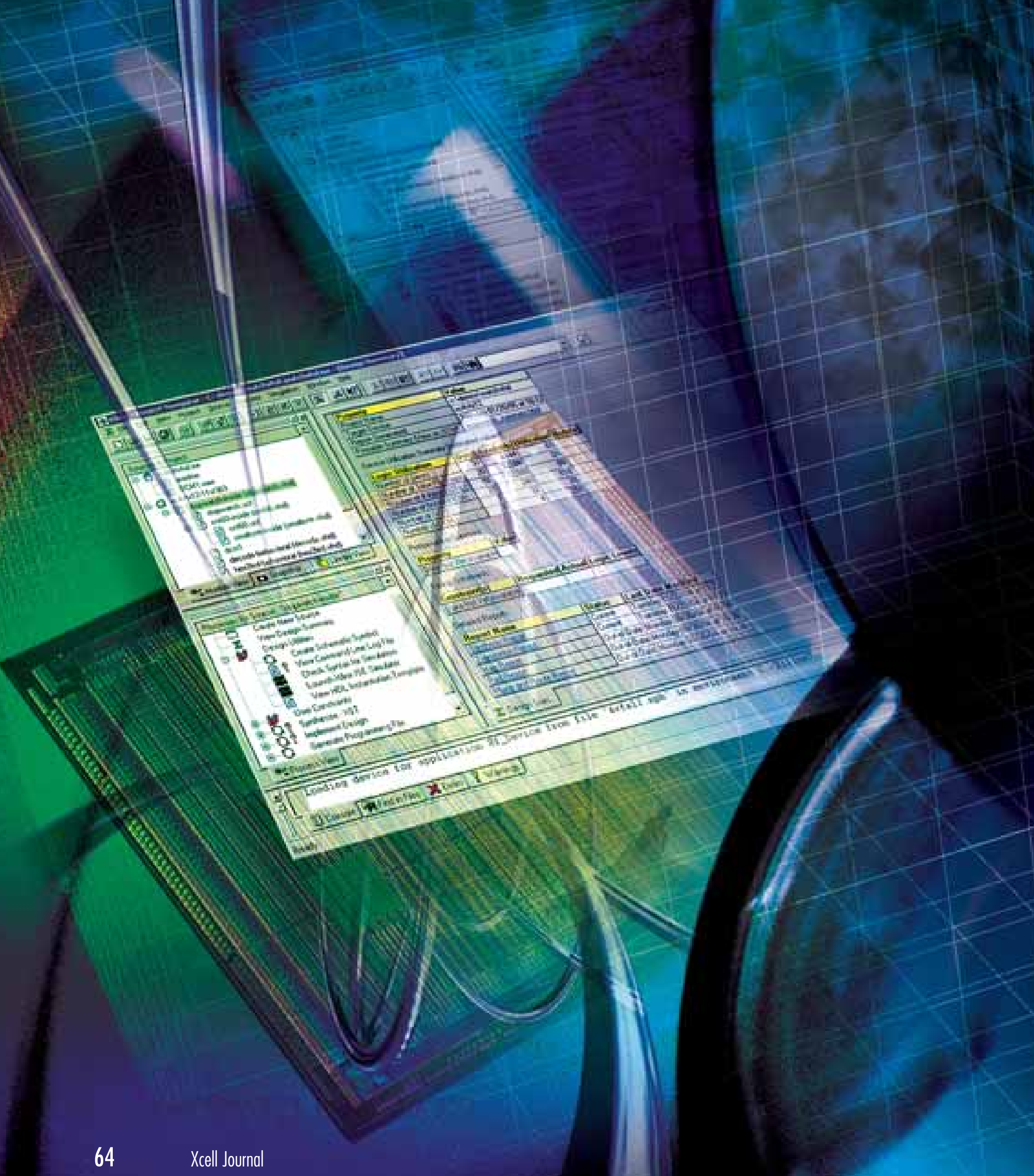
Xilinx® ISE™ software has long been known to deliver some of the fastest logic in programmable design, but the ISE design suite also delivers ease-of-use at the lowest cost possible. With the addition of advanced options like PlanAhead™ design and analysis software and ChipScope™ Pro real-time FPGA debug, the complete ISE system delivers solutions to design bottlenecks and gets you to project completion faster.

In this issue of the *Xcell Journal*, we present a series of articles about design tools and design applications to help you explore the new offerings behind the our 7.1i design tools release, from a general overview of ISE 7.1i software to detailed third-party explorations of logic analyzer setup and debug for FPGAs.

Table of Contents

Design Performance Leaps Forward with ISE 7.1i Software

Programmable design has never been easier or more cost-effective.



by Lee Hansen
Sr. Product Marketing Manager
Xilinx, Inc.
lee.hansen@xilinx.com

Mike Weir
North American Direct Marketing Manager
Xilinx, Inc.
mike.weir@xilinx.com

Some of the fastest tools in programmable logic have gotten even faster. Xilinx® ISE™ 7.1i software includes new technology, new device support, and new speed files that get you through the design flow faster, with the highest device performance possible.

ISE 7.1i software includes the new Virtex™-4 -12 speed grade, making the industry's leading-edge Virtex-4 FPGA-based designs even faster. ISE 7.1i software, in combination with Virtex-4 devices, can deliver up to 70% faster design performance than the nearest competing solution.

More high-performance technology is packed directly into ISE software than any other design offering, including core capabilities like "timing-driven mapping," global optimization, design re-timing, and FPGA physical synthesis. Together, this ProActive Timing Closure technology leads to higher overall design performance, with timing closure reached faster and less time spent in the overall design flow.

In addition to the advanced spectrum of ISE implementation tools, this collection of new technology slashes design times, getting you through even advanced, high-density design flows faster and cutting project time and costs.

PlanAhead Extends Performance Advantages

If you need even more design performance, the PlanAhead™ hierarchical floorplanner from Xilinx can help. PlanAhead software is a separately purchased tool option to ISE software that is ideal for high-density Virtex-4 or Spartan™-3E designs.

PlanAhead implements a block-based hierarchical design approach that can analyze, detect, and correct potential implementation problems earlier in the design cycle, leading to greater consistency, faster place and route cycles, fewer design iterations, tighter utilization control, and quicker incremental design changes. Customer-based benchmarks confirm that PlanAhead software can as much as double your out-of-the-box ISE design performance.

You can also lock placement results for individual blocks that already meet timing so that subsequent place and route iterations only affect the blocks that need to change, thus delivering faster reimplementation compile times. PlanAhead wraps area groups, incremental design, and modular design into a single ASIC-strength floorplanner. See the article "Control Your Designs with the PlanAhead Hierarchical and Design Analysis Tool" in this issue of *Xcell* for more detailed information.

Ease-of-Use Slashes Design Time

ISE 7.1i software includes a host of new tools and capabilities that you can't find anywhere else, including Technology Viewer, Message Filtering, Design Summary, ISE Simulator, and ModelSim Xilinx Edition-III. In addition to the advanced spectrum of ISE implementation tools, this collection of new technology slashes design times, getting you through even advanced, high-density design flows faster and cutting project time and costs.

New, More Powerful HDL Simulation Choices

Two new advanced HDL simulation offerings are now available with ISE 7.1i software: ISE Simulator and ModelSim Xilinx Edition-III. These state-of-the-art HDL simulation tools give you more performance and design capacity than previous versions of ISE software, and include new integration into the design flow.

ISE Simulator is a new, full-featured HDL simulator integrated within ISE software. It offers you the ability to simulate directly from the ISE Project Navigator process window; test benches and stimulus can be generated automatically, and graphing is direct in the ISE Project Navigator display window, as shown in Figure 1. ISE Simulator supports VHDL and Verilog and functional and timing simulation. ISE Simulator is also licensed through the existing Xilinx software registration ID process.

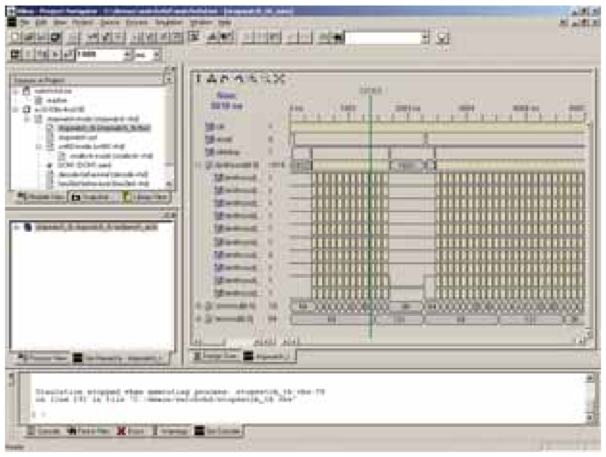


Figure 1 – ISE Simulator graphing window

Two different versions of ISE Simulator are available. ISE Simulator Lite comes included with ISE BaseX and ISE Foundation™ software at no charge. The full ISE Simulator is a design option that you can purchase as an upgrade to your existing ISE Foundation seat or purchase as a part of a new ISE Foundation seat.

Xilinx and Mentor Graphics recently announced an extended OEM agreement that has led to the release of the new ModelSim Xilinx Edition-III HDL simulator. The free MXE-III starter now offers designers a 50% faster HDL simulation environment and 20 times more design capacity than the previous version.

For larger designs, you can order the MXE-III full version, which provides five times the design capacity and 30% faster performance than the MXE-III starter. MXE-III enables you to simulate devices with up to 2 million system gates (depending on coding). Built on the industry-leading ModelSim 6.0a HDL simulation environment, MXE-III has enhanced GUIs that give designers easy access to capabilities and faster debugging. With the MXE-III full version, you now have the option to license your software with a USB dongle as well as the traditional methods of parallel and Ethernet ID.

Technology Viewer

ISE 7.1i software now includes the new Technology Viewer, which lets you view your post-synthesis HDL-based design at the block level (in a schematic-like display) to get an early picture of how your design will be represented during the implementation flow. Technology Viewer is built on the same graphic interface as RTL Viewer, so there are no new tools or menus to learn. Full hierarchy is represented, so you can easily push down into the design as far as necessary, highlighting and identifying critical elements in your design by pin, net, or instance name.

ISE software gives you more ways to view the progress of HDL-based design implementation as you move through the programmable design flow. RTL Viewer displays pre-synthesis design elaboration, while Technology Viewer shows the post-synthesis block-level design. FPGA Editor lets you see and edit post-place and route FPGA results.

Display Summary and Message Filtering

ISE Project Navigator includes a new Design Summary view that takes the most commonly sought-after design information and places it in one easy-to-use automated display, eliminating the need to search through multiple tool reports and outputs to find exactly what you need. Upon starting a new project, Design Summary shows you the key core information about your project, such as project location and target device. As you proceed through the design flow, Design Summary adds and updates more information, like logic utilization, performance summary, and design constraint results (Figure 2). Design Summary also contains a list of links that take you to more detailed implementation reports so that you can easily jump to more information. And Design Summary is right up front in ISE software, easily accessible as a selectable tab in the Project Navigator display window.

Message Filtering is a new ISE option that works with design reporting tools to capture the information that is exactly right for your design project. Message Filtering lets you select the report information that you deem non-critical to your design and suppress it from future reports. Message Filtering lets you get more streamlined and pertinent reports and quickly see the data necessary to your project, making debug and verification quicker and easier.

Lower Project Costs

ISE 7.1i software also offers new cost-savings options including new Spartan-3E device support; Virtex-4 device support in ISE WebPACK™ software and ISE BaseX; expanded Linux operating system support; and new electronic software delivery.

Expanded Device Support

Spartan-3E devices are the newest low-cost FPGA offerings from Xilinx, further reducing price points for FPGAs; they are now supported in ISE 7.1i software. Spartan-3E devices extend the reach of FPGAs into production volumes by reducing costs, while preserving the low NRE and high flexibility of programmable logic. Mainstream applications using Spartan-3E FPGAs and ISE 7.1i design tools will see lower ASIC crossover points and lower total project costs.

Virtex-4 FPGAs are now supported in more configurations of ISE 7.1i software as well. ISE WebPACK software adds Virtex-4 LX15 and LX25 support, while ISE BaseX now includes Virtex-4 LX15, LX25, SX25, and FX12 devices. You can choose the Xilinx device and ISE configuration that best fits your budget and design demands.

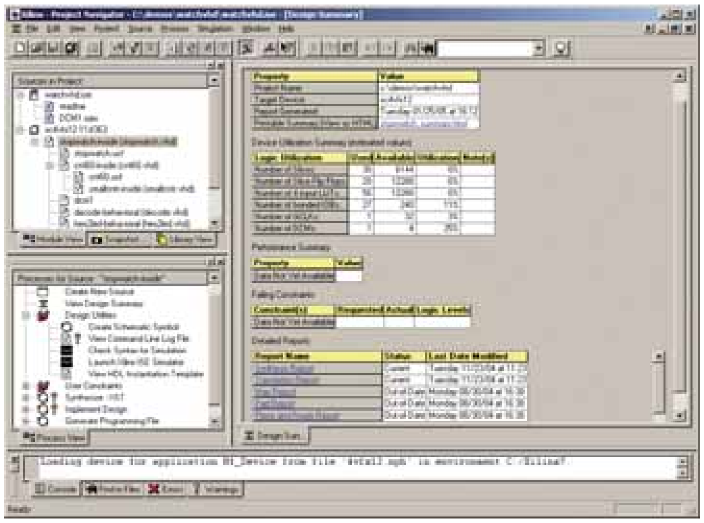


Figure 2 – Design summary in ISE Project Navigator

ISE 7.1i software also expands Linux operating system support. ISE Foundation software is available in both 32-bit and 64-bit Red Hat Enterprise 3 versions; both CDs are included in product update shipments. ISE WebPACK software also now runs on Linux Red Hat Enterprise 3, letting you use free, downloadable ISE software on the platform that best fits your needs.

Electronic Software Delivery

Xilinx is also now offering ISE Foundation software and ISE BaseX via electronic download to all in-maintenance customers. ESD gives you the option of getting your update ISE copy by the traditional box shipment or downloading from our secure delivery system. ISE 7.1i software became available via download just six days after its product announcement, so if you need your product shipment, you no longer have to wait for the box to arrive.

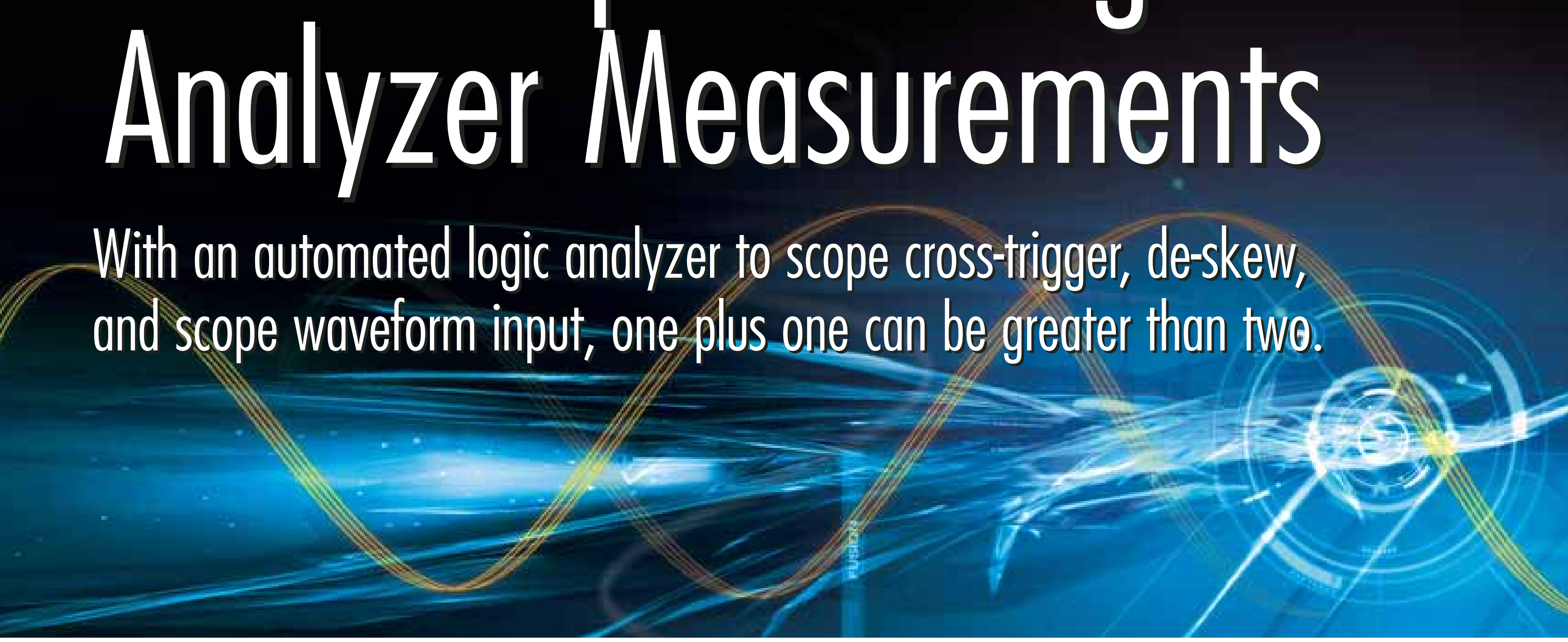
ESD also offers a cleaner delivery and registration system. Because your registration data is obtained and confirmed upon login, no product ID step is required. Product downloads and registration keys are available together online in a speedy and secure environment. Your ISE update can now be on your system and running faster than ever before. Xilinx is one of the only PLD vendors offering all of their logic design tools through electronic download.

Conclusion

ISE software sets the programmable design bar even higher with the release of 7.1i design tools. With faster performance, new design tools, advanced HDL simulation, expanded device and operating system support, and electronic download, we believe no competitor offers you more, whether you're making the next handheld CPLD-based PDA or high-speed serial, high-density FPGA-based blade server.

From logic development with ISE Foundation software, ISE BaseX, or ISE WebPACK software, embedded design with Platform Studio, system design with System Generator for DSP, real-time verification with ChipScope™ Pro analyzer, or floorplanning with PlanAhead design tools, Xilinx and ISE software have your project covered.

All ISE configurations, ChipScope Pro analyzer, PlanAhead software, MXE-III, and ISE Simulator are available for purchase from the Xilinx online store, Xilinx distributors, or by calling 1-800-888-FPGA (3742).



Debugging with Combined Oscilloscope and Logic Analyzer Measurements

With an automated logic analyzer to scope cross-trigger, de-skew, and scope waveform input, one plus one can be greater than two.

by Brad Frieden
Logic Applications Specialist
Agilent Technologies
brad.frieden@agilent.com

With higher speeds in today's digital designs, it's not uncommon to encounter challenges during board turn on that relate to timing or signal integrity. Most of the work is distinguishing whether it's a logic problem or a signal integrity problem. Spending too much time sorting out such things can reduce time to market and create even more time pressures when what you'd really like to do is design.

Many designers find a real-time oscilloscope and logic analyzer invaluable when it comes time to turn on, validate, and debug the hardware in their system. But they are often used independently – the oscilloscope for signal integrity or timing issues and the logic analyzer for problems with the logic.

But it turns out that these tools are most powerful when used together. You can significantly reduce debug challenges by taking advantage of tools that automate the integration of scopes and logic analyzers. In this article, we'll describe these tools and how you can more quickly understand digital design problems by capturing the analog characteristics of key digital signals simultaneously with a functional view.

An Example Situation

Consider a packet communication system where something is going wrong at boot up. This system (shown in Figure 1) should be passing packet data across a serial channel to a monitor, but the data is not being received. We can take a variety of approaches to figure out what exactly is causing the problem.

One approach is to view the condition of state machines in the design using a logic analyzer, with the possibility of seeing some improper states. Another approach is to probe onto the serial channel with an oscilloscope in an attempt to find some improper signals, particularly with respect to signal integrity. The best approach is to use both the oscilloscope and logic analyzer to better understand the exact target condition under which the error is occurring.

When Working Independently

When approaching the problem with a logic analyzer, a number of signals are of particular interest. We will certainly want to see what is happening with the transmit state machine driving packet data generation, along with the data and related transaction IDs . We would probably also like to see what is happening over on the receive side of the design, where packets should be collected. Signals of interest there include the state machine, data

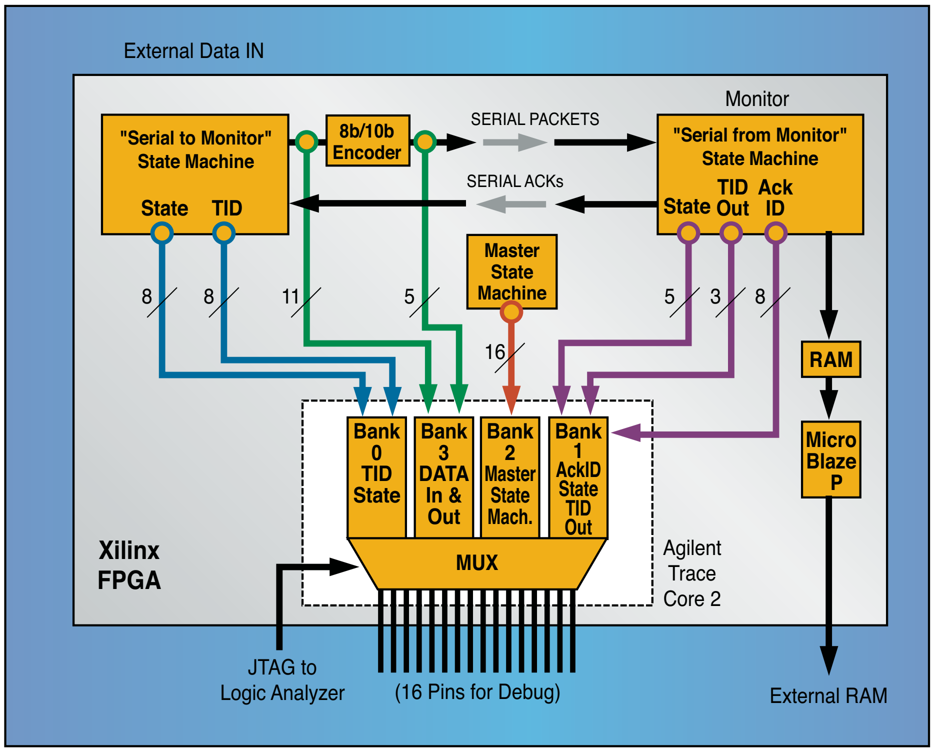


Figure 1 – Probe points of interest brought out through the Agilent trace core

received, and acknowledge IDs, which get sent back to the transmitter to let it know that data was received and to keep the data coming.

This design is implemented with a Xilinx® Virtex™-II FPGA, so we have the option to use a measurement core inside the FPGA to move our probe points around to various locations, collecting the data on the Agilent logic analyzer by using the logic analyzer-based FPGA Dynamic Probe application. We use a measurement core with four signal banks and 16 FPGA pins for visibility (Figure 1).

Because the data is not being received, a reasonable starting point is to probe these points of interest, set up the logic analyzer to trigger on anything, and just see what's happening. The answer is not much. In fact, simply looking at the activity indicators before even taking a trace, it becomes obvious that the transmit state machine is locked up. Acquiring data through bank 0 shows that the state machine is stuck at 01H, the "idle" state, and transaction IDs (TIDs) that should be counting are stuck at 0DH.

If we change our probe points to signals of interest over to the receive side through measurement bank 1, things are hung up there too, at 02H, the receive "idle" state.

Smarter Logic Analyzer Triggering

Now that we know the condition of the transmit state machine once it hung up, we can redefine the trigger point to be not that value, which should allow us to view state machine activity leading up to this hung condition. The acquisition reveals what looks like a normal state machine cycle just before the hang up. Similarly, by switching probe points with the FPGA Dynamic Probe over to the receive side and defining our trigger as not "idle," there is also a normal state machine progression. So we might try next to come at this problem with the oscilloscope.

The Oscilloscope Insight

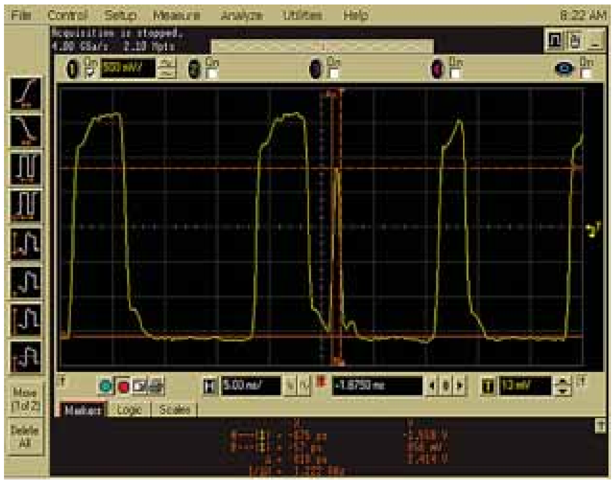
A point of particular interest for an oscilloscope measurement is to view activity on the serial channel and look for anything unusual. We saw the serial channel between the transmitter and receiver earlier in Figure 1. Because this link is differential and running at 400 Mbps, we used a 1.5 GHz differential active probe and accessories to let us plug in to the 0.1 inch connectors on the board.

As we attach to this link and get a view of the eye diagram on the data, it looks fine. No glitches, closing eye, or noise. Trying again from boot up, we see a flurry of activity for a second or so where there appear to be unexpected glitches or narrow pulses in infinite persistence mode. Fortunately we can trigger on a narrow pulse, and set up the scope to do that for a more detailed view (Figure 2). The resultant capture upon boot up is a single-shot view of a narrow pulse. This is highly interesting and reveals a definite problem in the system, but we're still left scratching our head as to what's going on.

How About a Little Cooperation?

What if we cross-trigger the logic analyzer when we catch this narrow pulse on the oscilloscope? We could see what condition the logic is at when this situation presents itself on the serial channel.

It turns out this is a relatively easy thing to do because of application software in the logic analyzer, enabled by the E5850A time-correlation fixture created to automate the process. By time-correlating the logic analyzer and oscilloscope and pulling the oscilloscope capture into the logic analyzer, we should be able to get a very helpful view to track down the problem.



Acquisition is stopped.
4.00 GSa/s 2.10 Mpts

Figure 2 – Scope trigger on a serial channel glitch and single-shot capture right out of reset

Time-Correlating the Instruments

Time-correlation is accomplished by double-probing a common reference signal with both the logic analyzer and oscilloscope. Common BNC cables are placed between one instrument's trigger-out connector and the other instrument's trigger input. The instruments are also connected together via the LAN, which allows the logic analyzer to control the oscilloscope and pull out its data.

A calibration routine that runs on the logic analyzer makes repetitive measurements in both the "logic analyzer triggers scope" direction and the "scope triggers logic analyzer" direction, to determine the delay in the cross-trigger for each direction. Once these delays are known, they can be applied to the oscilloscope data after importing to the logic analyzer, time-aligning the logic analyzer and oscilloscope data.

A Better Look

Now we're going to trigger the oscilloscope on the narrow pulse occurring on the serial channel, cross-trigger the logic analyzer, and bring both captured scope and logic analyzer data together on one screen. The goal is to see exactly what state machine activity is occurring inside the FPGA when the narrow pulse happens on the serial channel.

This cross-trigger is set up through the time-correlation application. All we have to set up is the scope trigger on the narrow pulse; the rest is taken care of.

Rebooting the system, we observe the capture on the logic analyzer, which fortunately is very revealing. The trigger point for the oscilloscope where the narrow glitch occurs is visible, and we can also see time-correlated to this event the transmit states of the state machine. That actually looks okay.

As we switch over to the "receive" state machine with the FPGA Dynamic Probe, the state machine also looks okay, at least initially (Figure 3). But with a change of time/division for a "macro" view, we see that the state machine stalls 12 packets after the serial channel glitch. The transaction IDs and acknowledge IDs increment for a time, but then stop. That's not acceptable, and it points to the possibility of invalid data being input to the state machine, somehow related to this narrow pulse on the serial channel.

Time-correlated measurements have helped show that these are interrelated events. The next step would be to isolate which one causes the other – correcting the logic causing a serial channel narrow pulse, or fixing the narrow pulse causing incorrect state machine logic.

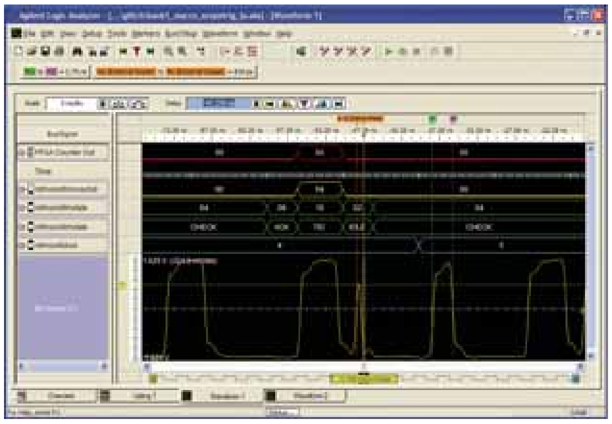


Figure 3 – Scope trigger on glitch, cross-trigger to logic analyzer, and capture of internal receive state machine, initially looking okay

Cross-Triggering in the Other Direction

We've determined that the first time we have a narrow pulse, it appears that it coincides with problems in our state machines. But does a narrow pulse always exist in the serial channel when the state machines lock up? One approach to determine the answer is to have the logic analyzer cross-trigger the oscilloscope and have the logic analyzer trigger on the lock up.

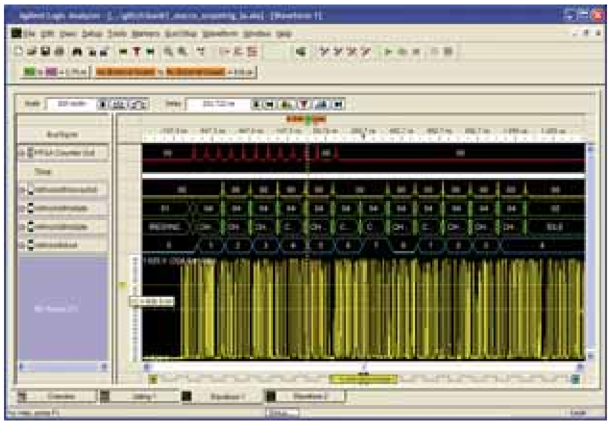


Figure 4 – Zoom-out reveals relationship of glitch to receive state machine stalling after 12 packets

Fortunately, to switch the direction of the cross-trigger simply requires checking a box in the user interface; the application already has determined the cross-trigger delay from the calibration performed earlier.

Because we know that the receive state machine locks up on a value of 02H (idle) immediately following 10H (TID), we can set up the logic analyzer to trigger when it sees TID, followed by idle for greater than one clock cycle.

Doing this, importing the related scope capture from the serial channel, and making multiple runs, we see that indeed there is a direct relation between the first narrow pulse and the state machines locking. We can search through the serial stream capture on the oscilloscope and find the location of narrow pulses with the E2681A EZJIT jitter analysis software package, validating that there are always multiple narrow pulses in the vicinity of the lock up. We've made an important step to narrow down the possible causes of the problem.

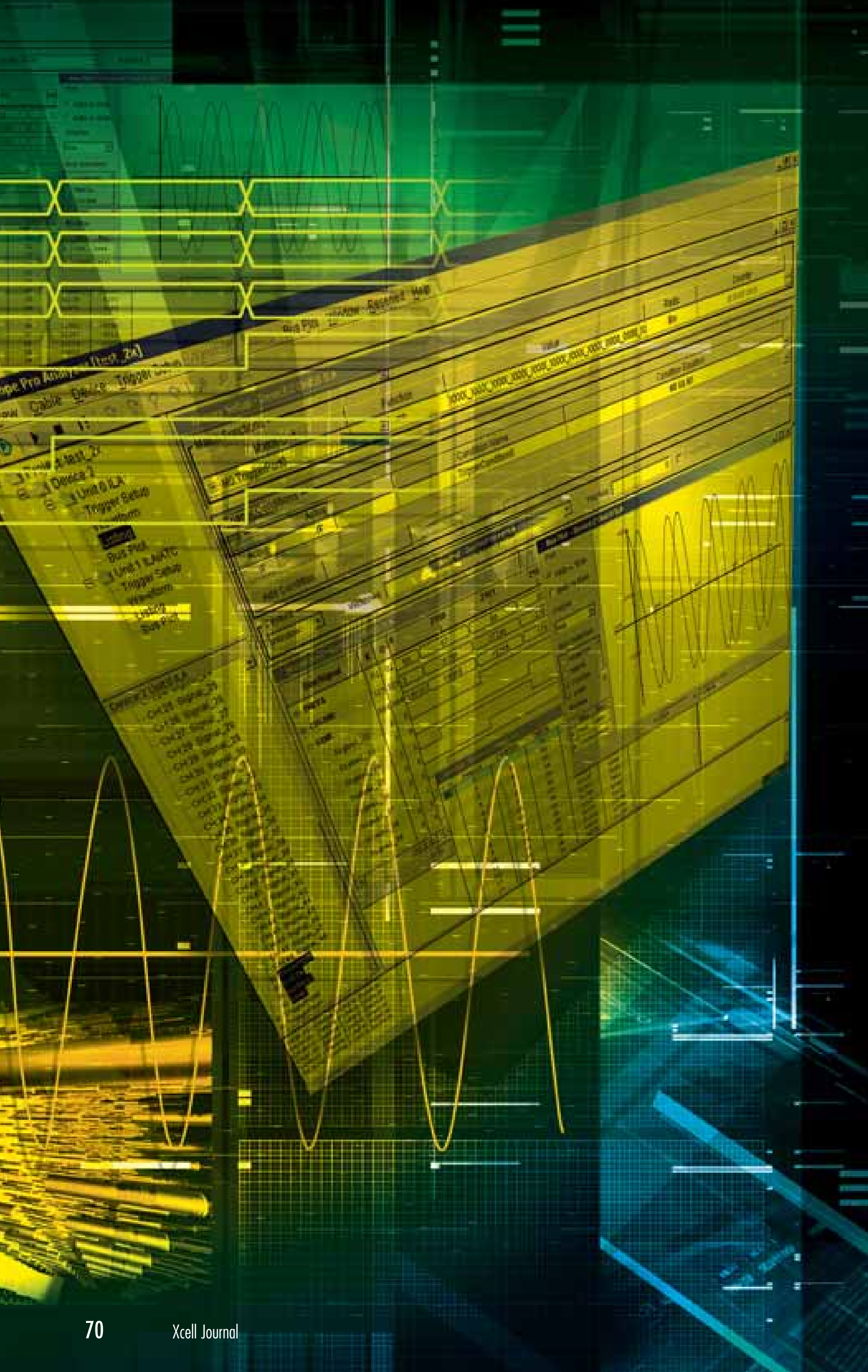
Conclusion

With complex interactions between asynchronous digital circuitry and the fast data rates that are susceptible to signal-integrity issues, it makes sense to pair up your logic analyzer and real-time oscilloscope. With the ability to automate the time-correlation of each instrument's capture to the other and import scope waveforms into the logic analyzer, it becomes much easier to track down elusive cause-and-effect problems, whether the source is logic or signal integrity.

For information about logic analyzers and oscilloscopes from Agilent, visit *www.agilent.com/find/logic/* and *www.agilent.com/find/scopes/*.

Real-Time Debug That Dominates

Debug your design with the ChipScope Pro system.



by Lee Hansen
Sr. Product Marketing Manager
Xilinx, Inc.
lee.hansen@xilinx.com

Brent Przybus
Sr. Product Marketing Manager
Xilinx, Inc.
brent.przybus@xilinx.com

Verification continues to be one of the most time-consuming and time-critical phases of the design flow. The fixed nature of an ASIC or structured ASIC makes identifying and finding problems on the board prohibitively expensive and massively time-consuming.

HDL verification and timing analysis tools offer a good first line of defense in predicting design behavior, but we believe no design tool offering beats the advantage that the Xilinx® ChipScope™ Pro system brings to FPGA real-time verification.

Faster FPGA Verification

The ChipScope Pro system, available as a separately purchased option to ISE™ software, allows logic and embedded software designers to debug their FPGAs in real time. You can find design problems quickly while the chip is running on the board, interacting with the rest of the system. Then, leveraging the FPGA's re-programmability, you can implement design changes quickly and send them back to the device on board in a matter of minutes through the FPGA programming cable.

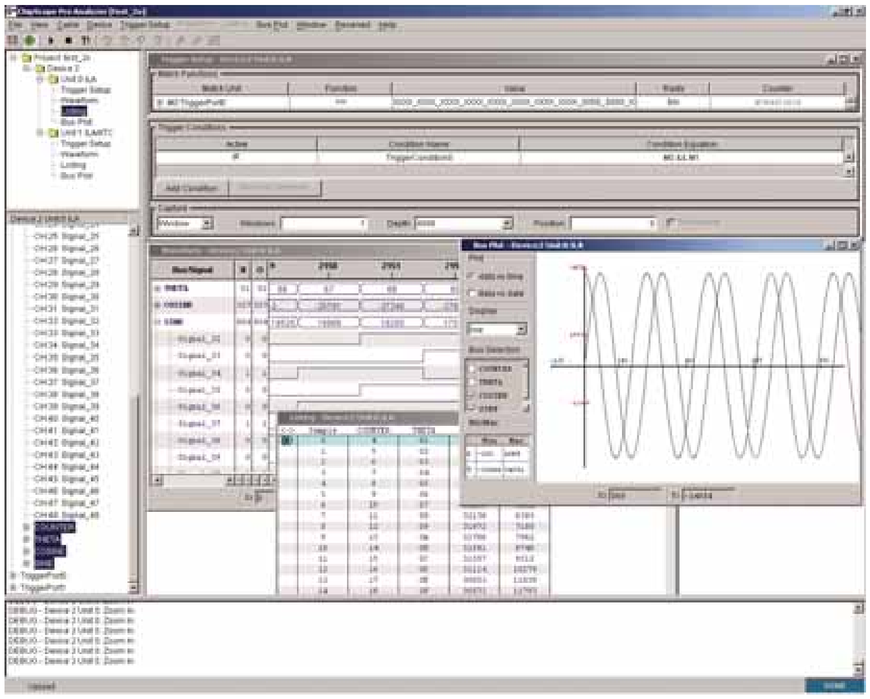


Figure 1 – ChipScope Pro logic analzyer

Debug Through Software Cores

The ChipScope Pro package of tools includes software debug cores, Core Inserter, CORE Generator™ software, and the integrated logic analyzer. The ChipScope Pro system is a separate option that plugs in directly to ISE design tools and comes in a 60-day evaluation version.

Four software debug cores are included with ChipScope Pro tools:

- ILA – for accessing and capturing logic signals up to 315 MHz – 50% faster than the previous ChipScope Pro release and among the fastest debug cores available
- IBA – for embedded processor bus signal capture, protocol detection, debugging and verifying control, address, and data buses
- VIO – for setting virtual inputs like external switches, mimicking output devices like LEDs, or for simulating external logic
- ATC2 – the advanced Agilent Technologies Trace Core 2, for linking the ChipScope Pro system to your Agilent logic analyzer and FPGA Dynamic Probe

You can quickly and easily configure and insert these low-profile cores into your FPGA logic during the design capture phase using CORE Generator software, or insert cores directly into the design netlist using Core Inserter. These cores are then synthesized and instantiated into your FPGA design, allowing you to view any internal signal within the FPGA.

For engineers designing embedded processor systems using the Virtex™-II Pro or Virtex-4 FX FPGA families, the ChipScope Pro system enables debug of embedded processor buses, including the IBM CoreConnect processor local bus or on-chip peripheral bus supporting the IBM PowerPC™ 405 processor. You can also view and debug embedded processor signals for the MicroBlaze™ soft-processor core, supporting all leading Xilinx FPGA device families.

Signals are captured at or near operating system speed and then brought out through the programming interface, freeing up pins for your design, not gobbling them up for debug. The ChipScope Pro real-time debug tool is one of the only tools that allow you to change probe points without having to re-synthesize and re-route your design. Using the ISE FPGA Editor, you can change signal probe points and then quickly reprogram your FPGA and debug a whole new set of signals in a matter of minutes.

Integrated Logic Analysis

You can analyze captured signals through the ChipScope Pro software logic analyzer (Figure 1). The ChipScope Pro logic analyzer is an advanced display and debug tool that makes logic and bus analysis easy. The ChipScope Pro logic analyzer supports multiple window views and bus plotting in either data-versus-time or data-versus-data formats.

Capture mode lets you compare data captured after multiple trigger events; signal filtering lets you ignore data that is not critical to your analysis, saving you memory and time. Using the listing viewer, you can import bus token files and view instructions in the order they occur.

To facilitate processor system debug environments that use software debuggers in addition to ChipScope Pro tools, you can share the JTAG connection to the FPGA with the ChipScope Pro analyzer.

In addition to providing data capture capabilities, the ChipScope Pro system also includes the Virtual I/O console, the interface to one of the industry's first real-time virtual input/output cores. Through the Virtual I/O console, you can control virtual inputs and pulse trains and view output activity. The ChipScope Pro system now runs on Windows, Solaris, and Linux Red Hat Enterprise 3.

New Capabilities

ChipScope Pro 7.1i software offers new capabilities that take real-time debug to the next level. Leading the way is the new remote debug capability. ChipScope Pro tools can now run in server/client mode over a TCP/IP connection. You can sit in your office while debugging a board next door in the lab or on the other side of the world. You can share a single board/system in the lab with other debug engineers on your team, or allow help desk personnel to debug a problem remotely at a customer site.

The ChipScope Pro system supports the newest low-cost FPGA offering from Xilinx, the Spartan™-3E family.

ATC2 software debug core has also been enhanced for automatic setup, allowing the logic analyzer to automatically find which ATC2/FPGA pins are connected to which logic analyzer pod signals, making setup faster and easier.

ChipScope Pro cores have also been performance-enhanced. System debug can now occur at clock speeds greater than 315 MHz, one of the fastest verification cores available.

Core Inserter can now be used across multiple netlists from one or more sources, allowing system integrators to debug entire designs rather than one section at a time. ChipScope Pro software also supports the newest Xilinx USB platform programming cable.

Linking to Agilent Logic Analyzers

The ChipScope Pro system also links internal FPGA debug to Agilent Technologies' bench-top logic analyzers using the included ChipScope Pro ATC2 core. This core synchronizes the ChipScope Pro system to Agilent's FPGA Dynamic Probe software, an optionally purchased plug-in to your Agilent 1680, 1690, or 16900 logic analyzer.

This unique partnership between Xilinx and Agilent delivers deeper trace memory, faster clock speeds, and more trigger options, all using even fewer pins on the FPGA. The advanced technology contained within the ATC2 core and FPGA Dynamic Probe isn't available in other FPGA or ASIC real-time verification solutions.

The FPGA Dynamic Probe measures new groups of internal FPGA signals in seconds without forcing you to recompile your design and with no impact on design timing. It achieves wider internal visibility over a fixed number of pins with 64 internal probe points for every pin, helping to conserve FPGA resources for your design. It eliminates error-prone and time-consuming tasks with features like automated signal/bus labeling from FPGA design to logic analyzer, and mapping FPGA pins from board layout to logic analysis channels. Figure 2 shows an example setup using an Agilent logic analyzer, FPGA Dynamic Probe, and ChipScope Pro software in a target system.

ATC2 can support 2, 4, 8, 16, 32, or 64 banks, with each bank supporting from 4 to 128 signal widths. Using optional 2X TDM in state debug mode, signal counts can be doubled, offering as many as 8,192 signals possible for debug in one system. And you can change input banks on the fly using the JTAG programming interface. ATC2 also operates in either state mode, for functional debug in one time domain with the widest signal capture; or timing mode, for making measurements across multiple time domains.

The ATC2 software debug core has also been enhanced for automatic setup, allowing the logic analyzer to automatically find which ATC2/FPGA pins are connected to which logic analyzer pod signals, making setup faster and easier. ATC2 can automatically find the ideal sampling point of each ATC2/FPGA pin in both phase and voltage offset. ATC2 also offers an increased number of multiplexer bank inputs from 32 to 64; and you can now start up your ATC2 debug with a known bank selected.

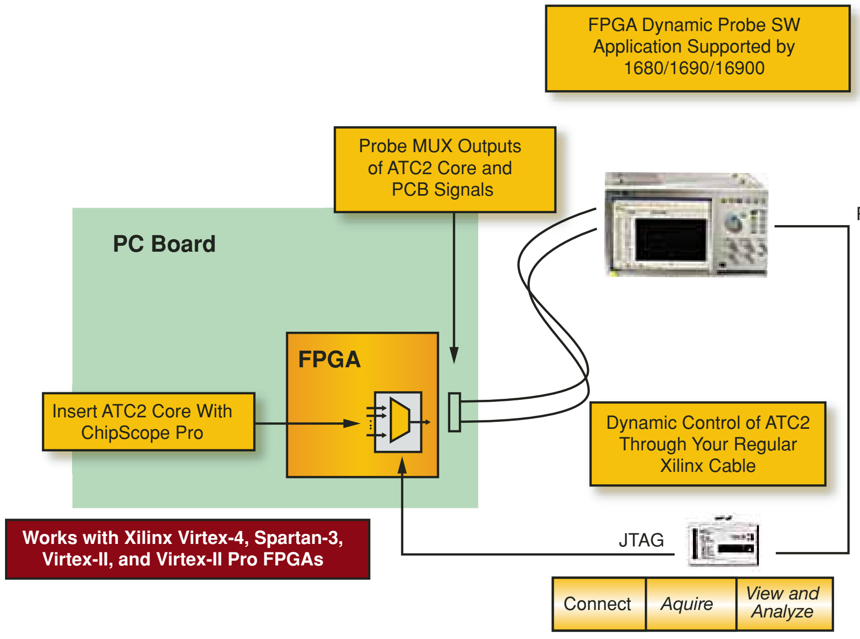

FPGA Dynamic Probe SW Application Supported by 1680/1690/16900
Probe MUX Outputs of ATC2 Core and PCB Signals
PC Board
Xilinx Parallel Cable
FPGA
Insert ATC2 Core With ChipScope Pro
Dynamic Control of ATC2 Through Your Regular Xilinx Cable
Works with Xilinx Virtex-4, Spartan-3, Virtex-II, and Virtex-II Pro FPGAs
JTAG
Connect
Aquire
View and Analyze

Figure 2 – ATC2 core, FPGA Dynamic Probe, and ChipScope Pro system

Conclusion

Nothing in FPGA or ASIC verification approaches what ChipScope Pro tools offer: real-time verification at or near system operating speeds (with a minimum impact on design space and I/O pins) and the ability to capture any signal inside the FPGA while it is running on the board, interacting with the system. Add to that the advantage of FPGA reprogrammability, and you can identify problems and change your design in a matter of minutes or hours, not days or weeks or months using other offerings. To order your ChipScope Pro 7.1i copy today, contact your Xilinx sales representative.

Integrating MATLAB Algorithms into FPGA Designs

The integration of AccelChip DSP Synthesis and Xilinx System Generator for DSP provides a seamless path from MATLAB/Simulink to verified FPGA-based DSP systems.

by Eric Cigan
Product Marketing Manager
AccelChip Inc.
eric.cigan@accelchip.com

Narinder Lall
Senior DSP Marketing Manager
Xilinx, Inc.
narinder.lall@xilinx.com

There are two kinds of people in electronic design: those who think in terms of words and those who think in terms of pictures. This dichotomy is most apparent in the world of DSP. Some designers prefer to develop algorithms in language form, while others choose to draw out block diagrams showing data flows. Until now, different sets of tools were required for each method – but why should you have to choose?

Xilinx® System Generator for DSP is well established as a productive tool for creating DSP designs using graphical methods. With a visual programming environment that leverages The MathWorks Simulink tool and its predefined Xilinx block set of DSP functions, System Generator meets the needs of both system architects (to integrate design components) and hardware designers (to optimize implementations). (For more details, see "Implementing DSP Algorithms in FPGAs" in the Winter 2004 issue of the *Xcell Journal*.)

Many DSP algorithm developers have found that the MATLAB language best meets their preferred development style. With more than 1,000 built-in functions, as well as toolbox extensions for signal processing, communications, and wavelet processing, MATLAB offers a rich environment for algorithm development and debugging.

In addition to the IP functions provided in MATLAB, the MATLAB language is uniquely adept with vector- and array-based waveform data at the core of algorithms in applications such as wireless communications, radar/infrared tracking, and image processing.

The AccelChip DSP Synthesis tool was developed specifically for algorithm developers and DSP architects who have embraced a language-based flow. With AccelChip, you begin with MATLAB M-files to perform stimulus creation, algorithm evaluation, and post-processing. You can load the M-files into the AccelChip tool; they become the "golden source" for a design flow that ultimately produces optimized implementations in Xilinx FPGAs.

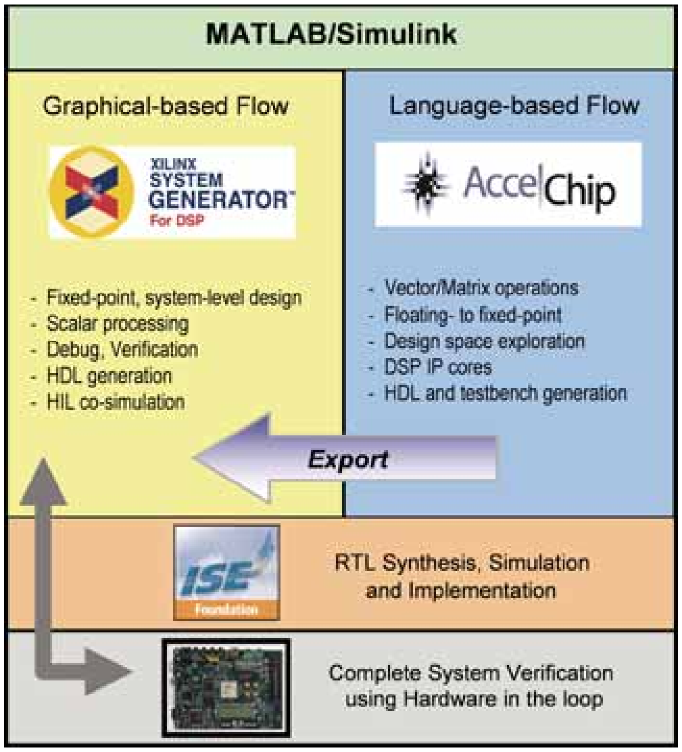

MATLAB/Simulink
Graphical-based Flow
XILINX SYSTEM GENERATOR For DSP
- Fixed-point, system-level design
- Scalar processing
- Debug, Verification
- HDL generation
- HIL co-simulation
Language-based Flow
AccelChip
- Vector/Matrix operations
- Floating- to fixed-point
- Design space exploration
- DSP IP cores
- HDL and testbench generation
Export
ISE Foundation
RTL Synthesis, Simulation and Implementation
Complete System Verification using Hardware in the loop

Figure 1 – System Generator/AccelChip interface

Unifying Words and Pictures

In the past, design teams looked on System Generator and AccelChip DSP Synthesis as mutually exclusive design tool options, but wished for access to the best aspects of each tool. In response, AccelChip and the DSP division at Xilinx collaborated in an effort to combine AccelChip's tools with System Generator. The result allows you to mix language-based algorithm design in MATLAB and graphical block-oriented design in Simulink in a novel unified electronic-system-level (ESL) design environment (Figure 1).

AccelChip's DSP Synthesis tool augments System Generator by providing a seamless integration path for algorithm developers, enabling the rapid creation of IP blocks, directly from M-files, that enhance the Xilinx block set in System Generator. In addition, AccelChip has optional AccelWare toolkits that complement System Generator with additional IP cores optimized for Xilinx cores. AccelWare toolkits include mathematical building blocks, signal processing, communications, and advanced math to implement linear algebra functions.

Kalman Filter Design Example

To illustrate this approach, let's take an advanced algorithm written in MATLAB, use AccelChip to synthesize the design, and then integrate it into a System Generator model. Our example is a Kalman filter – a recursive, adaptive filter well-suited to combining multiple noisy signals into a clearer signal (for details on the topic, see Arthur Gelb's book, "Applied Optimal Estimation").

Kalman filters embed a mathematical model of an object – such as a commercial aircraft being tracked by ground-based radar – and use the model to predict future behavior. Kalman filters then use measured signals (such as the signature of the aircraft returned to the radar receiver) to periodically correct the prediction.

```
function [S] = simple_kalman(A)
    DIM = size(A,2);
    persistent p P_cap
    if isempty(P_cap)
        P_cap = [8 0 0; 0 8 0; 0 0 8];
        p     = ones(DIM,1)/2;
    end;
    I = eye(DIM);
    R = [128 0 0;0 128 0; 0 0 128];

    % estimate step:
    %p_est     = p;
    P_cap_est = P_cap+I;

    % correction step:
    K     = P_cap_est * inv(P_cap_est+R);
    p     = p + K * ( A' - p );
    P_cap = (I - K)*P_cap_est;
    S     = p';
```

Figure 2 – Kalman filter example M-file

Figure 2 shows the MATLAB M-file describing the Kalman filter. The algorithm defines matrices R and I that describe the statistics of the measured signal and the predicted behavior. The last nine lines of the algorithm are the code that predicts and corrects the estimate.

This algorithm illustrates the flexibility and conciseness of the MATLAB language. Common operators such as addition and subtraction operate on variables like the two-dimensional arrays A or P_cap without having to write loops, as you would in languages like C. Multiplication of two-dimensional arrays is automatically performed as matrix multiplication without any special annotation. MATLAB operators such as matrix transposition allow the MATLAB code to be compact and easily readable. And complex operations like matrix inversion are completed using MATLAB's extensive linear algebra capabilities. Although such an algorithm could be constructed as a block diagram, doing so would obscure the algorithm structure so readily apparent in MATLAB.

With AccelChip, a first step in synthesizing a complete algorithm is to generate any major cores that are referenced – in this case, the matrix inverse indicated by the function call inv(P_cap_est+R). But you can implement a matrix inverse in many ways; the choice of which method to use depends on the size, structure, and values of the matrix.

Using the matrix inverse IP core from the AccelWare toolkit, you can choose from micro-architectures designed for different applications. These micro-architectures can be optimized for speed, area, power, or noise. In this case, the most suitable approach is to use the AccelWare QR matrix inverse core.

Synthesizing RTL with AccelChip

With the MATLAB M-file loaded into AccelChip, the next step is to simulate the floating-point design to establish a baseline. You would then use AccelChip to convert the design to fixed-point math, verifying it in MATLAB as shown in Figure 3. AccelChip offers an array of tools to help you trim bits from the design and verify fixed-point design effects like saturation and rounding. AccelChip aids in this process by propagating bit growth throughout the design and letting you use directives to set constraints on bit width. This algorithmic design exploration allows you to attain the ideal quantization that minimizes bit widths while managing overflows or underflows, allowing early trade-offs of silicon area versus performance metrics.

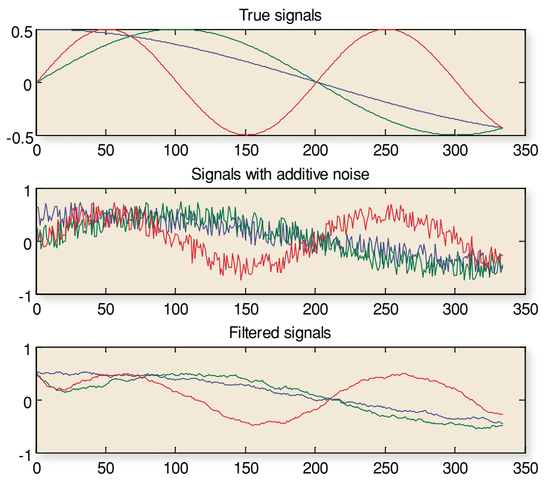


Figure 3 – The Kalman filter constructs estimates of the three signals based on input signals corrupted by noise.

Once you have attained suitable quantization, the next step is to generate RTL for your target Xilinx device. At this point, you can use the AccelChip GUI to set constraints on the design using the following design directives to achieve further optimizations:

- Rolling/unrolling of FOR loops
- Expansion of vector and matrix additions and multiplications
- RAM/ROM memory mapping of vectors and two-dimensional arrays
- Pipeline insertion
- Shift-register mapping

Using these directives constitutes hardware-based design exploration, allowing the design team to further improve quality of results. In synthesizing the RTL, AccelChip evaluates the entire design and schedules the entire algorithm, performing necessary boundary optimization in the process.

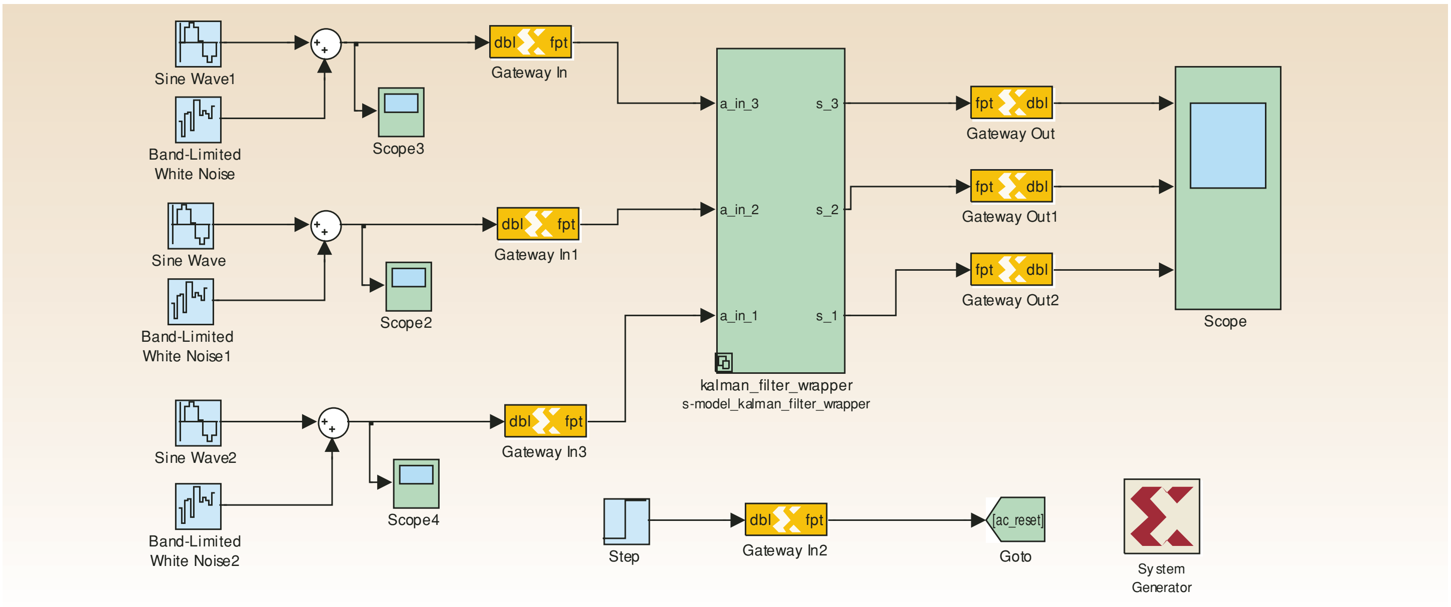


Figure 4 – AccelChip exports a cycle-accurate System Generator block that supports both simulation and RTL code generation.

Throughout this flow, AccelChip maintains a uniform verification environment through a self-checking test bench; the input/output vectors that were generated when verifying the fixed-point MATLAB design are used to verify the generated RTL. The RTL verification step also gives AccelChip the information necessary to compute the throughput and latency of the Kalman filter. This is essential information to assess whether the design meets specifications and is critical for achieving cycle-accurate simulation.

Exporting from AccelChip to System Generator

Although RTL verification is a key step in the design flow, designers want to see algorithms running in hardware. System Generator's hardware-in-the-loop co-simulation interfaces make this a push-button flow, allowing you to bring the full power of MATLAB and Simulink analysis functions to hardware verification.

Now that you have run RTL verification in AccelChip, you are ready to export the AccelChip design to System Generator by going to the "Export" pull-down menu in the AccelChip GUI and selecting "System Generator." AccelChip then generates a cycle-accurate System Generator block that supports both simulation and RTL code generation.

At this point, you transition the design flow to System Generator, where a new block for the Kalman filter is available in the Simulink library browser. You need only select the Kalman filter block and drag it into the destination model to incorporate the AccelChip-generated Kalman filter into a System Generator design.

Figure 4 shows the resulting diagram for the Kalman filter. In the center of the System Generator diagram is the Kalman filter, and around it are the gateway blocks needed for a System Generator design.

Once the AccelChip-generated block is included in the System Generator design, you can perform a complete, system-level simulation of cycle-accurate, bit-true models to verify that the system meets specifications. You can use AccelChip-generated blocks for System Generator in conjunction with the Xilinx block set. Once this system-level verification step is completed, the next step in the System Generator flow is to move on to design implementation. The "Generate" step in System Generator compiles the design into hardware.

Verification Options

All design files generated by AccelChip, including exported System Generator files, are verified back to the original "golden" source MATLAB M-file. AccelChip's verification approach is based on the generation of a test bench from the MATLAB source – this test bench is applied at the RTL level within AccelChip and can be applied in System Generator to verify the correctness of the design.

Once verified in the System Generator environment, you can verify the AccelChip-generated block using System Generator's supported methods – including HDL co-simulation and hardware-in-the-loop – to accelerate hardware-level simulation 10 to 100 times.

Conclusion

The integration of AccelChip with Xilinx System Generator is the first solution to unite MATLAB-based algorithmic synthesis favored by algorithm developers with the graphical design flow used by system architects and hardware designers. It uses the rich MATLAB language and its companion toolboxes to create System Generator IP blocks of complex DSP algorithms.

By using these tools together, design teams can employ the most productive means of modeling hardware for implementation, fully involving algorithm developers in the FPGA design process and completing higher quality designs more quickly.

For more information on AccelChip and its interface to Xilinx System Generator, visit *www.accelchip.com*. For more information on System Generator, visit *www.xilinx.com/products/design_resources/dsp_central/grouping/index.htm*.

Complex FPGAs Require Equivalence Checking

Synopsys' Formality equivalence checker proves identical functionality.

by Lonn Fiance
Director, Strategic Alliances
Synopsys
lonn@synopsys.com

Robert Vallelunga
Product Manager
Synopsys
robertv@synopsys.com

With the latest process technologies, complex high-performance systems – once the exclusive domain of ASIC and custom chips – can now be created with FPGAs. Although this benefits both consumers and companies, it also creates significant new design and verification challenges that must be addressed to meet time-to-market requirements.

As a designer, you use system-level analysis and simulation to evaluate different architectural alternatives. The result of this architectural exploration is the RTL (register transfer language) specification, which forms the definition of your device's functionality.

Depending on the product requirements, you can realize this functionality in a number of forms: using Xilinx® devices as FPGAs designed into the final product or prototypes for verification of complex ASICs.

The RTL description defines the functionality, so maintaining equivalent behavior for all implementations under all circumstances is critical. As design sizes have increased, the ability to prove equivalence by exhaustive simulation has disappeared.

ASIC designers recognized this issue several years ago and turned to equivalence checking (EC) to maintain functionality. When you design one of today's high-complexity FPGA devices, whether used in the final product or as an ASIC prototype, you are clearly facing these same challenges. As a result, EC is quickly becoming a mandatory verification methodology in FPGA design flows. This methodology becomes even more important if you have selected the EasyPath™ low-cost FPGA solution for your production needs.

In this article, we'll provide an overview of EC technology and the benefits it brings to FPGA design. Using the Synopsys Formality equivalence checker in your FPGA design flow enables you to quickly verify the implementation of your device, giving you the freedom to focus your efforts on other design tasks.

Introduction to Equivalence Checking

EC is a formal, static verification technology that uses mathematical techniques to determine if two versions of the same design, at different stages of development, are functionally equivalent. The power of this technique is its ability to compare between:

- Two RTL versions
- An RTL description and a gate-level netlist
- Two gate-level netlists

EC flows consist primarily of four stages:

1. Read: the EC tool reads RTL descriptions or netlists for the reference and implementation designs and segments the logic in each design into smaller components called logic cones (Figure 1). Logic cones are simply small groups of logic bordered by registers, ports, and black boxes. The end points for each logic cone are known as compare points.

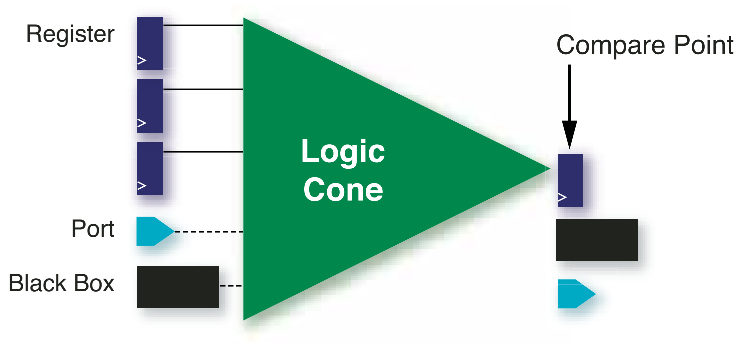


Figure 1 – Logic cone with compare point

2. Match: using a variety of techniques, the equivalence checker maps corresponding compare points between the implementation and reference designs (Figure 2). The fastest of these techniques is name-based matching, but differences between the RTL and gate-level representations can lead to highly dissimilar names, making advanced matching algorithms necessary. Registers may have been duplicated, merged, or otherwise optimized away such that the logic cones between design versions may be quite different, making the matching process far from simple.

3. Verification: using mathematical techniques, each set of matched compare points is proven to be functionally equivalent or non-equivalent.

4. Debug: when non-equivalent logic cones have been identified, graphical debug techniques are available to isolate the logic causing functional deviations.

Static verification has two primary benefits: it is orders of magnitude faster than dynamic verification and provides 100% verification coverage. EC can prove that different versions of a design are equivalent (or not) in a matter of minutes, rather than the hours or days required for dynamic simulation. Equivalence checking also eliminates

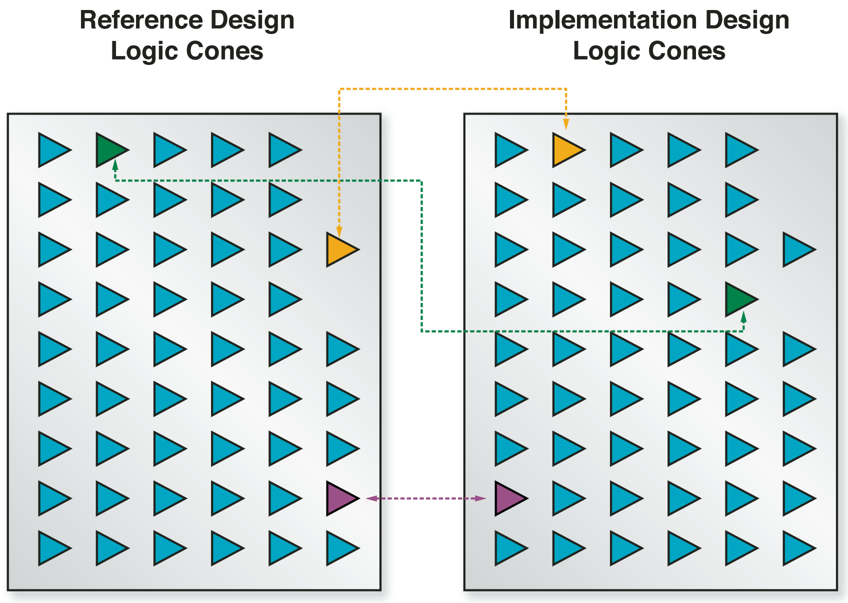

Reference Design Logic Cones
Implementation Design Logic Cones

Figure 2 – Matched (or mapped) compare points between design versions

Today's very large programmable devices may contain hundreds of thousands of logic cells, making equivalency checking a mandatory component of the modern competitive design flow.

the need to create extensive sets of test patterns in an attempt to demonstrate equivalence by automatically identifying compare points and then applying sophisticated verification techniques. Dynamic simulation remains an important part of your verification strategy, but its primary use is to ensure proper RTL behavior.

Equivalence Checking in FPGA Flows

Because the value of EC is well understood by ASIC designers, its initial application to FPGA prototyping is an obvious choice. In the prototyping context, you want to prove that your FPGA prototype implements the RTL functionality that will be used to create an expensive ASIC.

The FPGA verification usually happens in two phases: first by proving the equivalence of the reference RTL to the post-synthesis netlist, and second by proving the equivalence between the post-synthesis netlist to the post place and route (PAR) netlist. You can perform a similar check between the reference RTL and the ASIC implementation netlist(s) to ensure that your final ASIC matches the functionality proven in the FPGA prototype, as shown in Figure 3.

The benefits of EC also apply when you intend to use the FPGA as the final implementation of your design. The ability to quickly and exhaustively prove the correspondence between your reference RTL, the post-synthesis netlist, and the post-PAR netlist significantly reduces simulation time or multiple programming iterations, neither of which can conclusively prove the equivalence between various versions of your design.

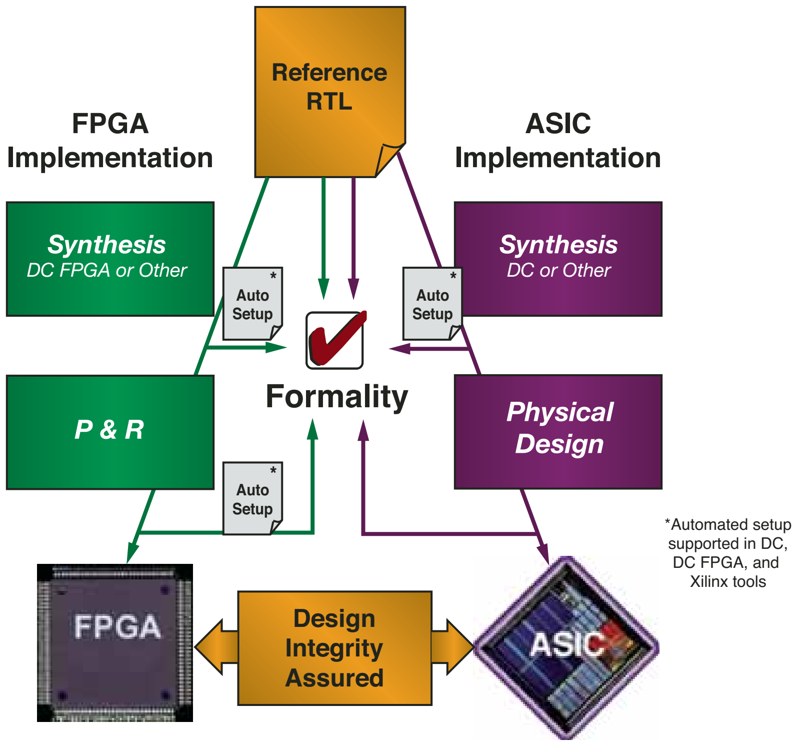


Figure 3 – Verifying an ASIC implementation against an FPGA prototyped implementation

Block-level design elements like digital clock managers (DCMs) and block RAMs, inherent features of the Xilinx architecture, pose interesting challenges in FPGA designs. When they are instantiated – as is common in FPGA prototyping flows – they are treated as "black boxes" for synthesis and EC purposes. The contents of a black box are not verified, but the signals at the periphery of the "black-boxed" element are proven to be equivalent.

RAMs can also be inferred from the RTL. These inferred RAMs can be directly verified by Formality as long as the inferred RAM does not become excessively large. Large memories can be fully verified using memory-specific verification tools.

Using Formality in a Xilinx Design Flow

Formality, the proven equivalency checking tool from Synopsys, allows you to verify designs 10-100X faster than simulation-based techniques – and identify and correct logic errors 5-10X faster – because of its graphical schematic debug capabilities. This performance advantage enables you to run Formality at every stage of the implementation flow and catch errors when they are first introduced, greatly reducing the cost of correcting them.

Functional verification using Formality is the clear winner in terms of performance and error isolation. Today's very large programmable devices may contain hundreds of thousands of logic cells, making equivalency checking a mandatory component of the modern competitive design flow.

Synopsys and Xilinx have created the Formality EC flow for Xilinx FPGA designs (shown in Figure 4). Design discrepancies can result from the numerous transformations that take place during synthesis and in ISE™ software (NGDbuild, MAP, and PAR). These transformations may change the design to

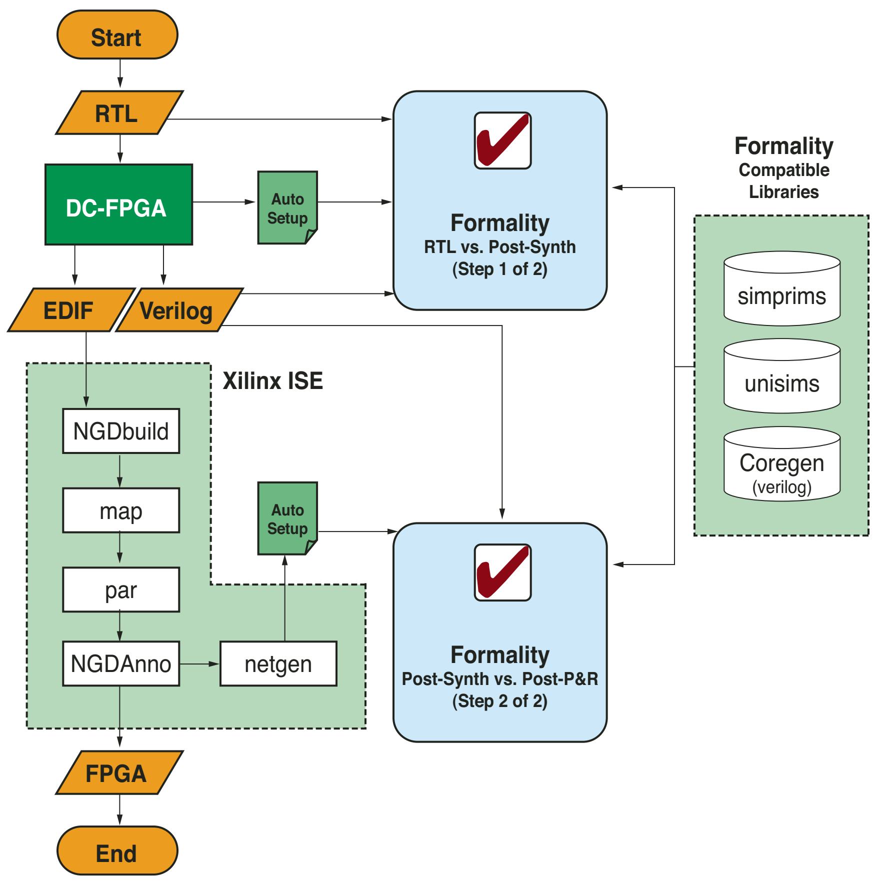


Figure 4 – Formality/DC FPGA/Xilinx verification flow

improve timing, reduce area or power, better match the architectural features of the Xilinx device, or meet design rules. Transformations such as combinatorial reductions, sequential optimizations (retiming), FSM re-encoding, register merging or duplication, and place and route optimizations increase the risk of unintended functional changes.

EC tools consider two versions of a design independently, without knowing how they were created. Therefore, Formality has no awareness of the transformations that occurred during design implementation. These transformations include changes to the logic, in net and instance names, in hierarchy, and to the number or meaning of state elements. These changes may impact the names of compare points, preventing the use of efficient name-based matching techniques and increasing the use of advanced but slower matching techniques.

Additionally, any changes to a bounding element of a logic cone results in a change to the logic cone itself. For example, if you optimize away a register during synthesis, you remove the end point of a logic cone, forcing that logic cone to be expanded until another end point is found. The effect is that the matching of compare points may no longer be possible; even if a match is made, the logic between corresponding cones is no longer equivalent and verification will fail.

Tight Linkage Between Tools Improves Usability

In the past, EC tools required extensive tool setup to understand these logic optimization transformations. Fortunately, Synopsys Design Compiler FPGA and Xilinx PAR tools write out an automated setup file for Formality. This file contains information telling Formality which optimizations were performed in particular areas of the design, minimizing the manual setup information that you might otherwise need to provide. Formality uses this setup information to understand the differences between the two versions and, after validating the information, uses it to more quickly complete the equivalence check. Linking multiple tools through the setup file helps you achieve the fastest possible time to results and eliminate errors that might be introduced in a manual setup process.

Complex FPGA devices are increasingly being designed using modular, or hierarchical, flows. Formality provides an ideal way to verify each individual module and ensure that they have been correctly stitched together at the upper levels of your design.

Formality is a mandatory step in the EasyPath methodology. EasyPath devices are not reprogrammable, so ensuring that the device is 100% equivalent to the reference RTL is essential. Formality's static analysis techniques prove functional equivalence, minimizing the risk of implementing incorrect functionality to help you reach volume production sooner.

Conclusion

Today's FPGAs have achieved the same level of functional complexity as some ASICs. The ability to realize very complex functionality in FPGAs has created tremendous challenges in the FPGA verification process. Formality helps you:

- Achieve 100% coverage
- Reduce runtimes 10-100X versus traditional dynamic verification
- Quickly isolate discrepancies between the RTL and implementation
- Reduce or eliminate the time spent re-simulating after a design change
- Verify that changes did not unintentionally impact other functionality

Synopsys and Xilinx have worked together to create a proven FPGA static verification solution centered on Formality, Design Compiler FPGA, and Xilinx ISE implementation tools. Formality provides a fast, thorough functional verification methodology for proving equivalence between multiple representations of your design. For more information on Formality or DC FPGA, visit *www.synopsys.com* or contact your Synopsys sales representative.

Complete FPGA and CPLD Power Analysis

The Power Central website contains everything you need to accurately predict FPGA or CPLD power consumption.

by Lee Hansen
Sr. Product Marketing Manager
Xilinx, Inc.
lee.hansen@xilinx.com

Tony Thomas
Technical Marketing Engineer
Xilinx, Inc.
tony.thomas@xilinx.com

Device power consumption has become one of the leading design issues facing FPGA and CPLD engineers today. Device consumption can affect everything from cost and longevity of the device in your project to system performance and battery life in hand-held applications. Xilinx® design tools have expanded over the last few releases to offer you more ways to generate accurate estimates of your device power consumption.

The Power Essentials

There are two main components to FPGA chip power consumption:

- Dynamic power is largely determined by the switching power of the core and the switching speed of the I/O. Dynamic power is largely affected by capacitive load, supply voltage, and switching frequency.
- Quiescent power is dominated largely by transistor leakage current and by DC current from a few specialized FPGA circuits.

The drive to lower FPGA costs drives transistor size down, which tends to raise quiescent power consumption. Design demands also continue to force design performance faster, leading to higher switching rates and higher dynamic power consumption.

Xilinx Virtex™-4 FPGAs can deliver as much as 70% more performance than the nearest competing FPGA offering. Normally, this trend might raise dynamic power consumption. Designs are also getting denser year to year; again, in normal circumstances this tends to raise static and dynamic power. However, the Virtex-4 family offers dramatic static and dynamic power reduction through architecture, design, and process, offering as much as

1/10th the static power of other 90 nm FPGAs. You can also obtain a 20X power reduction using the available Virtex-4 embedded functions.

How has Xilinx been able to reduce power in its Virtex-4 FPGA? Triple-oxide is the Virtex-4 process innovation that has reduced leakage current, whereas two-oxide thickness was used for Virtex-II Pro devices and all past families. However, Virtex-4 FPGAs add a third middle-thickness-oxide transistor, thereby reducing static power.

The Virtex-4 architecture has also enabled dynamic power-per-CLB to be reduced by 50% at comparable frequencies. Even at a 50% higher operating frequency, Virtex-4 devices reduce dynamic power by 20%.

An overall increase in power demand also emphasizes average junction temperature and thermal power consumption. Exceeding the allowable junction temperature leads to reduced system performance, forces reduced device utilization, and reduces device reliability. The more information you have about junction temperature and how your design will affect it, the more informed your decision will be on package types and thermal design considerations, as well as where you can make design changes to help reduce potential thermal problems.

Power consumption is design-dependent and affected by output loading, system performance (switching frequency), design density (number of interconnects), design activity (the percentage of interconnects switching), logic block and interconnect structure, and power supply voltage levels. You can perform power calculations at three distinct phases of the design cycle:

- Concept phase – calculating a rough estimate of power based on estimates of logic capacity and frequency of operation. Supported using Xilinx Web Power Tools.

- Detailed design phase – calculating power more accurately based on detailed information about how the design is implemented in the FPGA. Supported through XPower and included with all copies of ISE™ software.

- System integration phase – measuring power using benchtop instrumentation (board-level measurements).

Xilinx Power Tools utilize "activity rate" to reach accurate power estimates. Clocks and other input signals have an absolute frequency. Synchronous logic signals use a percentage "activity rate" relative to the associated clock. An activity rate of 100% indicates that a net is expected to change state on every clock cycle. Activity rates can be set globally (12% of the nets switch each clock cycle), on groups of signals, or individual signals. "Activity rate" allows you to adjust clock frequency in XPower and see the effect on power consumption to your design results. The frequency of logic elements displayed by XPower is a function of their output signal(s).

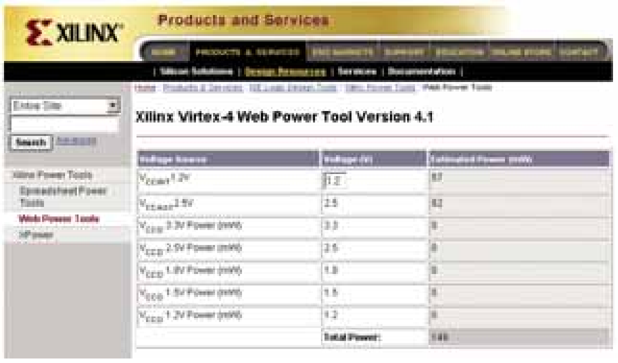
Power Tools

The Xilinx Power Central website for FPGA and CPLD power analysis (*www.xilinx.com/power/*) contains links to Xilinx web-based power tools, information on XPower, ISE-integrated power analysis software, white papers, design examples, and links to power-centric partner products and software.

Web-Based Power Tools

Web-based power estimation is the quickest and easiest way to get an idea of device power consumption early in the design flow. Xilinx offers a complete set of web-based power tools on Power Central. A new version is released every quarter, so information is current, and no installation or downloading is required – just an Internet connection and a web browser. You can specify design parameters and save and load design settings, eliminating the need to re-enter design parameters with iterative use. Just an estimate of design behavior and a target device will get you started.

Version 4.1 of Xilinx web-based Power Tools was released in March 2005. This new release is designed to help customers quickly and easily estimate the power consumption of their target Xilinx device, and in particular includes important new Virtex-4 LX, SX, and FX family enhancements (Figure 1).



Xilinx Virtex-4 Web Power Tool Version 4.1

Voltage Source	Voltage (V)	Estimated Power (mW)
V_{CCINT} 1.2V	1.2	87
V_{CCAUX} 2.5V	2.5	62
V_{CCO} 3.3V Power (mW)	3.3	0
V_{CCO} 2.5V Power (mW)	2.5	0
V_{CCO} 1.8V Power (mW)	1.8	0
V_{CCO} 1.5V Power (mW)	1.5	0
V_{CCO} 1.2V Power (mW)	1.2	0
	Total Power:	149

Figure 1 – Web Power Tools

- Power results are now affected by changes in ambient temperature – one of the only power tools able to consider ambient temperature

- Iccintq and Iccauxq have been updated with new, more accurate figures

- The Virtex-4 XtremeDSP™ block power has been updated with new measurements at low, medium, and high toggle rates

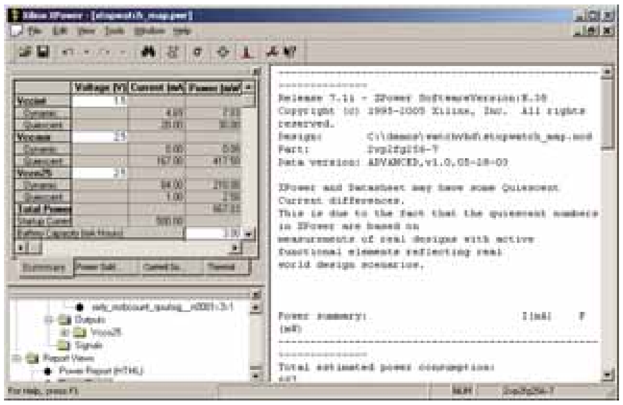
- Latest thermal data updates for all supported devices

In addition, new updates that affect both Virtex-4 FPGAs and other Xilinx device families include:

- VccInt can now be varied for Virtex-4, Virtex-II, Virtex-II Pro, and Spartan-3 devices

- More accurate power estimation for all open-drain outputs

- New and improved online help updated with all features available for Virtex-4 FPGAs and other families; all web spreadsheets now have clickable header links that take you directly to online help



```
Release 7.1i - XPower SoftwareVersion:E.38
Copyright (c) 1995-2005 Xilinx, Inc.  All rights
reserved.
Design:       C:\demos\watchvhdl\stopwatch_map.ncd
Part:         2vp2fg256-7
Data version: ADVANCED,v1.0,05-18-03

XPower and Datasheet may have some Quiescent
Current differences.
This is due to the fact that the quiescent numbers
in XPower are based on
measurements of real designs with active
functional elements reflecting real
world design scenarios.


Power summary:                          I(mA)    P
(mW)
-------------------------------------------------
Total estimated power consumption:
```

Figure 2 – XPower

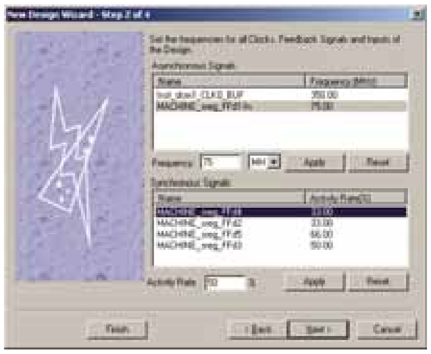

New Design Wizard - Step 2 of 4
Set the frequencies for all Clocks, Feedback Signals and Inputs of the Design
Asynchronous Signals
Frequency
Apply
Reset
Synchronous Signals
Activity Rate
Finish
< Back
Next >
Cancel

Figure 3 – XPower new design wizard

XPower – Integrated, Design-Specific Power Analysis

XPower, a free part of all Xilinx ISE design tool configurations, allows you to get a much more detailed estimate of your design-based power requirements. XPower estimates device power based on a mapped or placed and routed design. XPower calculates an estimate of power with an average design suite error of less than 10% for mature in-production FPGA and CPLDs. It considers device data along with your design files and reports estimated device power consumption at a high level of accuracy, customized to your specific design information.

XPower is integrated directly into ISE software and gives hierarchical and detailed net power displays, detailed summary reports, and a power wizard that makes it easy for new users to run XPower. XPower can accept simulated design activity data and runs in both GUI and batch mode (Figure 2).

XPower considers each net and logic element in the design. The ISE design files provide exact resource use; XPower cross-references routing information with characterized capacitance data. Physical resources are then characterized for capacitance. Design characterization is continuous and ongoing for newer devices to provide the most accurate results. XPower uses net toggle rates as well as output loading. XPower then computes power and junction temperature, and can display individual net power data as well.

Available to CPLD and FPGA users, the XPower design wizard allows for easy data entry to XPower and is an ideal tool for beginners. The wizard helps you enter all the data required by XPower to get a good power estimate, including design files, simulation data, operating environment, output loading, and activity rates. The wizard makes XPower results more accurate the first time, and eliminates costly and time-consuming learning curves (Figure 3).

Power Central also contains reference information to help you understand and reduce power across your entire PLD project. Application notes and user guides with specific device and design examples go through everything from power distribution considerations to minimizing power in hand-held CPLD applications. There is also key information from power supply partners like Intersil and Texas Instruments on third-party power-based products and offerings.

Conclusion

The Xilinx Power Central website delivers everything you need to analyze your FPGA or CPLD-based design, with tools like XPower in ISE software, Web Power Tools supporting the leading Xilinx FPGA and CPLD product families, and complete partner and reference data.

During this demo, you will learn the following:

- Simplifying the embedded processor verification flow
- Easily importing designs from Xilinx Platform Studio EDK to Seamless FPGA
- Verification with Seamless FPGA: accelerating the simulation process
- Performance profiling and analysis

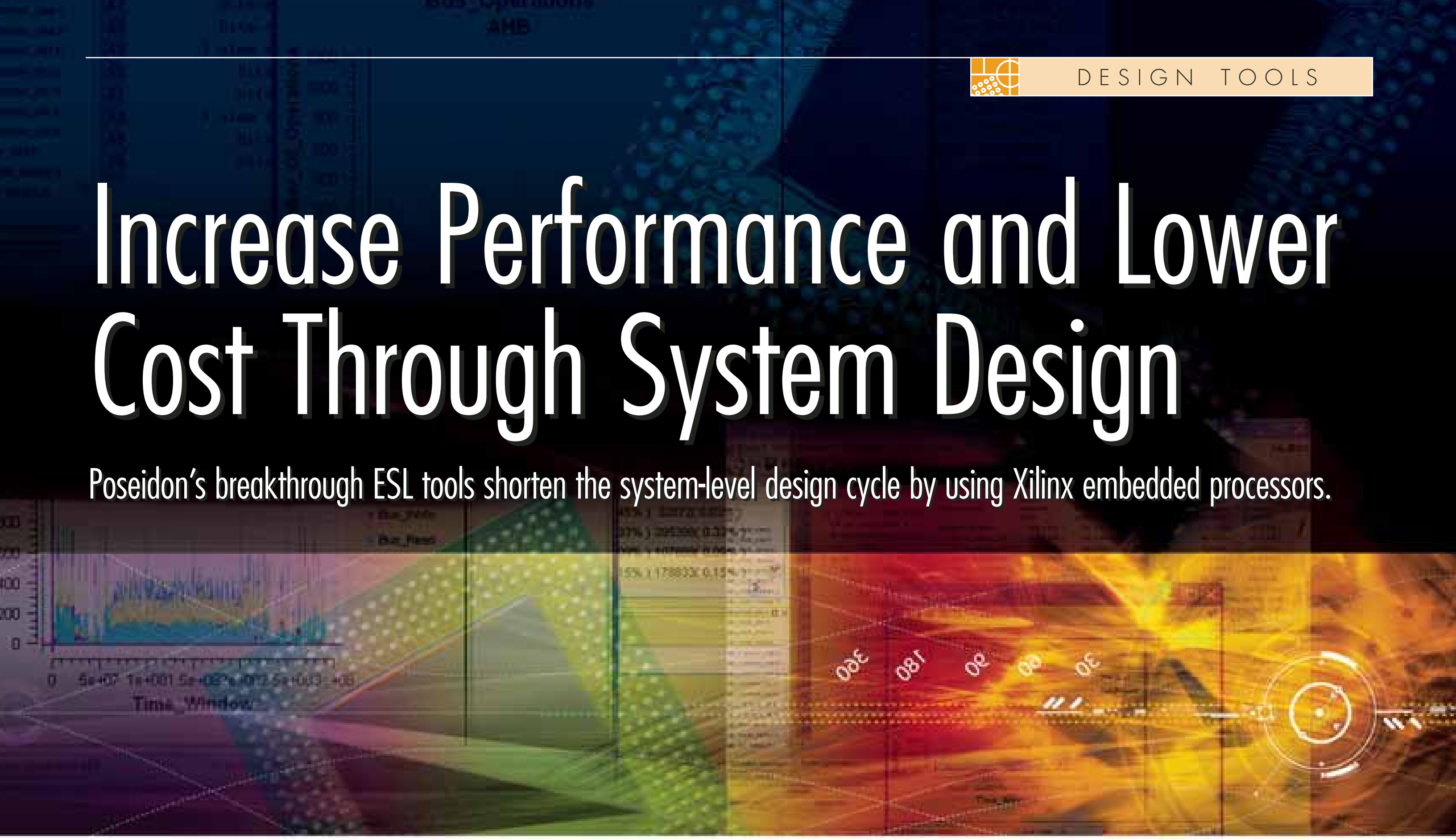
Watch the demo today:
http://www.mentor.com/products/fv/hwsw_coverification/seamless_fpga/demo_request.cfm

For additional details about Seamless FPGA, visit us at www.seamlessfpga.com

Mentor Graphics®

Increase Performance and Lower Cost Through System Design

Poseidon's breakthrough ESL tools shorten the system-level design cycle by using Xilinx embedded processors.



By Farzad Zarrinfar
VP of Worldwide Sales & Marketing
Poseidon Systems
Farzad.Zarrinfar@poseidon-systems.com

Bill Salefski
VP of Engineering
Poseidon Systems
bills@poseidon-systems.com

Stephen Simon
Director of Sales & Business Development
Poseidon Systems
ssimon@poseidon-systems.com

The introduction of embedded processors for FPGAs (like the immersed PowerPC™ and Xilinx® MicroBlaze™ soft-processor core) has greatly expanded the benefits of platform FPGAs. These advantages include lower FPGA power consumption, lower device cost, and more scalable performance, particularly when balancing your application's execution between what will run in the fabric of the FPGA versus the embedded processor(s).

Designing efficient processor-based system architectures and optimizing overall system performance for a specific application is challenging. For example, effectively partitioning the hardware and software early in the design is difficult, yet critical to the development of the architecture. Also, architectures must be verified early in the design cycle to maximize the benefits of processor-based designs. You cannot wait until RTL development to discover that your architecture does not support the system requirements. This "redesign loop" can easily delay the overall design cycle.

Newer and better alternatives exist to partition, analyze design bottlenecks in, implement, and verify system architectures for Xilinx FPGAs with embedded hard or soft processors. Electronic system-level (ESL) tools and predefined cores can help you make critical architectural decisions, simplify your design tasks, and reduce your overall design cycle. In this article, we'll show how Poseidon's new ESL tools can help you realize these benefits.

Triton Tool Suite

Poseidon's Triton Tool suite is a system design and acceleration environment that enables you to quickly develop, analyze, and optimize system architectures. With these ESL tools, the abstraction level of the design is above RTL – you can quickly address system issues without having to solve the detailed issues surrounding RTL implementation. Thus, you can quickly perform architectural "what-if" analysis and accelerate time to market. The tool suite was created specifically for processor-based systems that require efficient robust architectures with the need to optimize performance, power, and cost.

Triton comprises two main tools:

- Triton Tuner – a system and software analysis tool
- Triton Builder – a hardware/software partitioning tool

Triton Tuner is a simulation and analysis environment based on SystemC. Simulation is performed at the transaction level from models of both the processor and surrounding buses and peripherals. Using Tuner, you co-simulate the hardware with the application software.

During simulation, the tool collects both hardware and software performance data. The environment then provides tools to visualize and analyze the data, reducing the effort required to identify inefficiencies in the design. Figure 1 shows a bus activity graph – a visualization tool to help identify bus congestion and bottlenecks in the system design.

Triton Tuner also links hardware and software events, providing a key tool in determining causal relationships between hardware and software systems. This feature greatly simplifies the task of evaluating and optimizing system performance. Tuner also provides tools to optimize the memory structure. This can greatly reduce the need for costly caches and other high-speed memories.

Triton Builder is an automated partitioning tool, which simplifies the task of accelerating the performance of processor-based algorithms. With Triton Builder, you can easily offload compute-intensive loops and functions from software to hardware. The tool automatically creates a DMA (direct memory access)-based hardware accelerator to perform the offloaded task. All hardware and source code modifications are also created to greatly simplify the repartitioning process.

With the builder tool, you can control a number of key parameters in which to optimize the solution to your system requirements. Together, these tools provide a system design and optimization environment that unobtrusively plugs into a Xilinx EDK design flow, enabling you to quickly make dramatic improvements in the performance and power consumption of your application.

Triton Builder generates the RTL for the complete accelerator, the driver required to invoke the accelerator, and inserts the driver into the proper place in the original application. The RTL can be generated in either Verilog or VHDL. This integral generation process ensures that necessary hardware and software components are exactly matched for trouble-free design. The tool also generates an RTL test bench and SystemC model for verifying the new hardware.

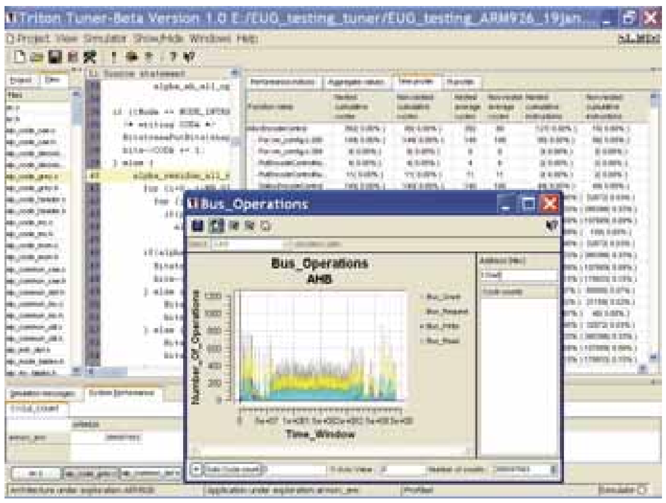


Figure 1 – Bus activity graph

Design Architecture of FPGA-Based Accelerator

The builder tools create a design architecture ideally matched to FPGA design. Using Triton Builder, you can quickly create a peripheral using the C code that runs on the processor. The tool moves the computationally intensive portions of the code into a hardware accelerator. This acceleration hardware is connected directly onto the processor bus and implemented on the FPGA fabric. The block diagram of a typical accelerated architecture is shown in Figure 2.

If while executing the application code the processor hits a section of code that has been moved to hardware, control is passed to the accelerator through the inserted driver, which then performs the accelerated function. The accelerator runs independently from the processor, freeing it to perform other tasks. When the accelerated task is completed, the results are passed back to the application program. You can implement multiple independent accelerators within the same system.

To create a complete accelerator peripheral, you need more than just a C synthesis tool. Poseidon Builder includes not just a C-to-RTL synthesizer that creates the compute core, but it also generates the communication and control hardware. The Poseidon tool creates a complete accelerator peripheral, the block diagram of which is shown in Figure 3. In addition to the compute core, the accelerator includes a multi-channel DMA controller, bus interface, local memories, and other logic to create a plug-and-play peripheral.

Interfacing to Xilinx Tools

Triton Tools are extremely flexible, allowing you to use either Triton Tuner or Triton Builder independently or together as an integrated suite. These tools were developed to enhance the productivity in developing processor-based designs and vastly increase their capability.

The Triton tools link seamlessly to EDK tools. A typical system design flow is usually an iterative process where you would analyze the system performance, determine inefficiencies, modify the system, and check the resultant performance. When the resultant architecture performs to the desired level, the system is then transferred back into the Xilinx

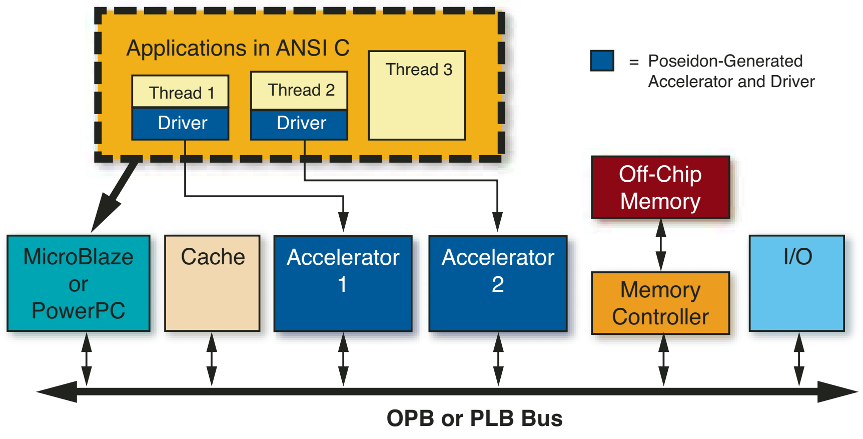


Figure 2 - Accelerated system architecture

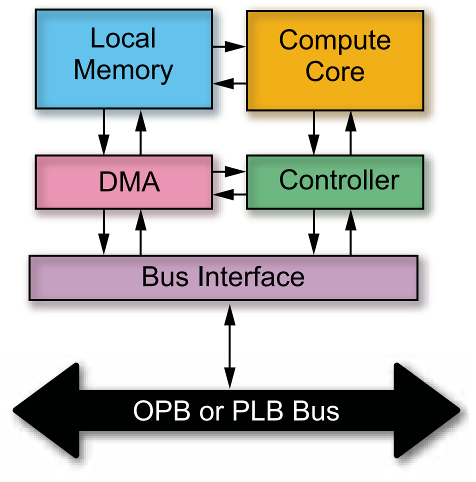


Figure 3 - Accelerator block diagram

tool chain. The Triton tools accelerate the process of identifying problem areas and establish an integrated flow that allows you to move between the tools to develop the optimal architecture.

A typical flow (shown in Figure 4) comprises these steps:

- The designer develops the target architecture using selected Xilinx tools
- The architecture description is read from the microprocessor hardware specification file (.mhs) from EDK, and the ANSI C application source code is read into the Triton tools
- Triton Tuner profiles the ANSI C code, reveals bottlenecks in the code or architecture, and eliminates inefficiencies
- Triton Builder partitions compute-intensive algorithms in ANSI C into hardware and generates a hardware accelerator
- Triton Tuner verifies that the new system performs to the desired level
- RTL, test bench, modified C code, driver, and architecture are exported back into the Xilinx environment

Conclusion

Poseidon's Triton Tool suite enables you to rapidly and predictability analyze, optimize, and accelerate processor-based architectures. With Triton Tuner's SystemC simulation environment, you can develop robust efficient processor architectures. With Triton Builder, you can add sophisticated hardware acceleration to your processor-based systems.

The Triton tool suite enables design architects to achieve higher system throughput, reduced power consumption, and cost, as well as shorter design cycles. For more information, visit our website at *www.poseidon-systems.com* or contact the sales department at (925) 292-1670. To be qualified for a free tool evaluation of the Poseidon Builder and Tuner and a free white paper, e-mail *farzad.zarrinfar@poseidon-systems.com*.

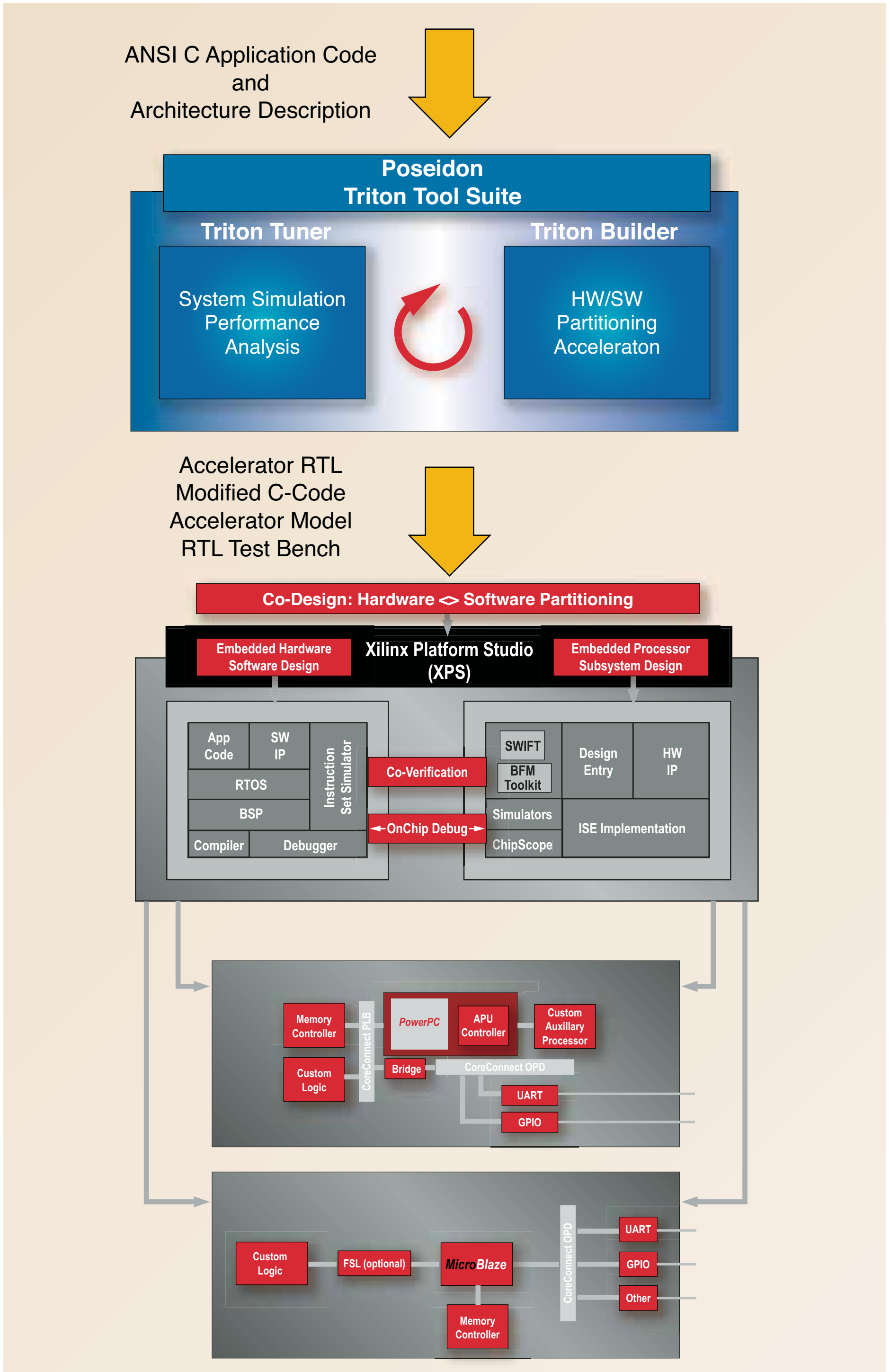


Figure 4 - Integrated Poseidon and Xilinx tool flow

Control Your Designs with the PlanAhead Hierarchical Design and Analysis Tool

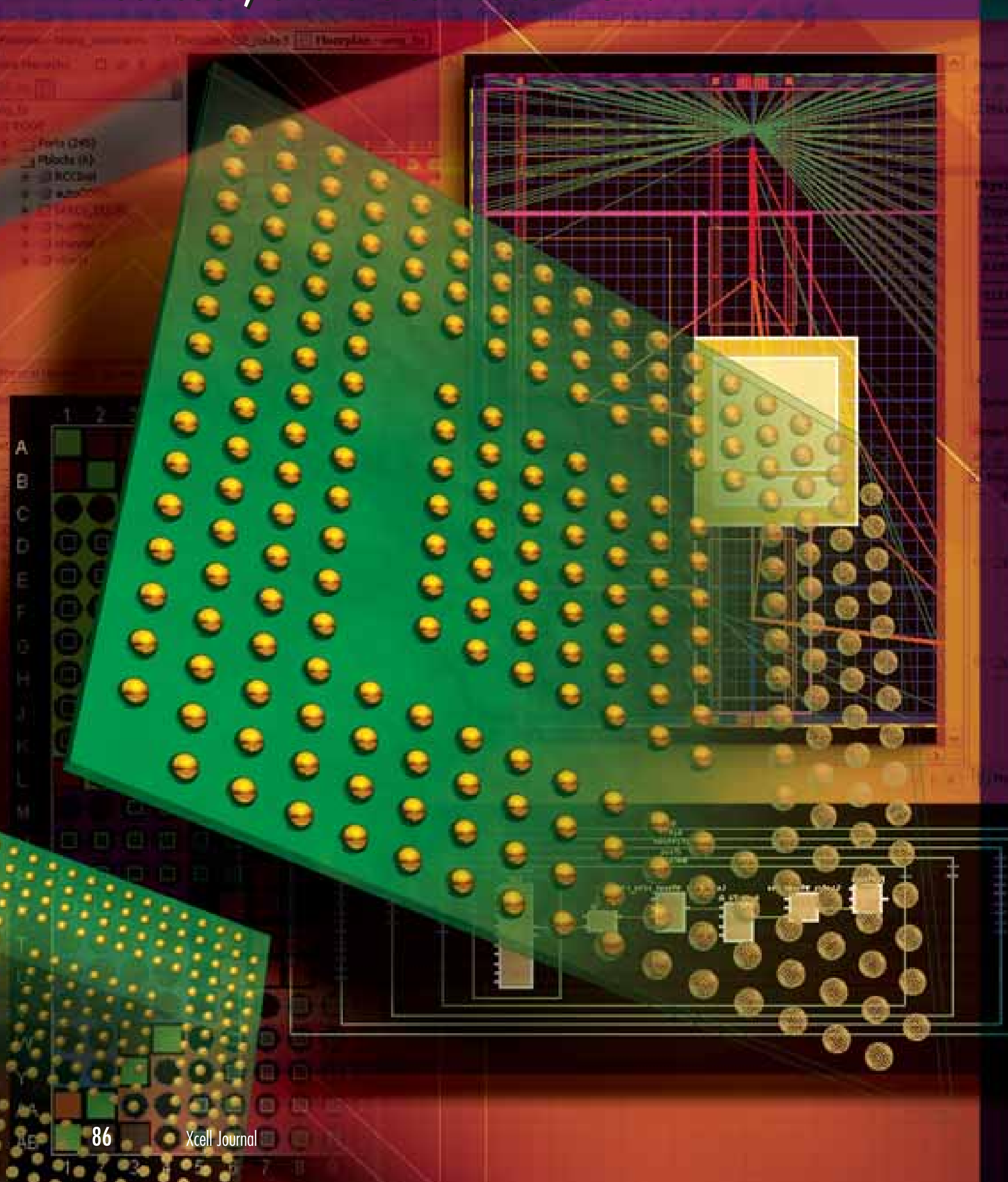
You can improve performance and reduce design closure times through quick analysis, feedback, and a block-based flow.

by Salil Raje
Software Development Director
Xilinx, Inc.
salil.raje@xilinx.com

David Knol
Senior Staff Engineer
Xilinx, Inc.
david.knol@xilinx.com

Brian Jackson
Product Marketing Manager
Xilinx, Inc.
brian.jackson@xilinx.com

Mark Goosman
Product Marketing Manager
Xilinx, Inc.
mark.goosman@xilinx.com



If you are pushing the limits of performance and density in an FPGA, then you are also likely facing problems that threaten your design deadlines. Long place and route runtimes and repeatability of results are two common sources of anxiety.

Also, traditional FPGA tools provide a flat, push-button implementation approach – a flow that provides few clues about how to improve your design goals and even fewer opportunities to make the necessary changes.

If you can relate to these challenges, Xilinx® offers a new tool that will put you in control of your design. The PlanAhead™ hierarchical design and analysis tool allows you to employ ASIC design methodologies for complex FPGAs. PlanAhead software easily integrates into your existing flow between synthesis and place and route, as shown in Figure 1. You can analyze, detect, and correct many potential problems before place and route. And with its block-based incremental capabilities, you can divide and conquer your design one block at a time.

Analysis

PlanAhead design tools take a synthesized EDIF netlist as input and one or more UCF files for constraints. The powerful and easy-to-use graphical environment displays the Xilinx device that you have targeted (see Figure 2). At this point you have several tools available to explore your design space.

Timing Analysis

TimeAhead is a flexible timing analyzer built into PlanAhead software. It allows you to estimate route delays before running place and route. Using the PlanAhead block-based approach, the accuracy of timing estimates will improve as blocks in the design are implemented through place and route.

TimeAhead also allows you to analyze and identify performance bottlenecks in small sections of the design. You can use this feedback to improve results through re-synthesis or better floorplanning.

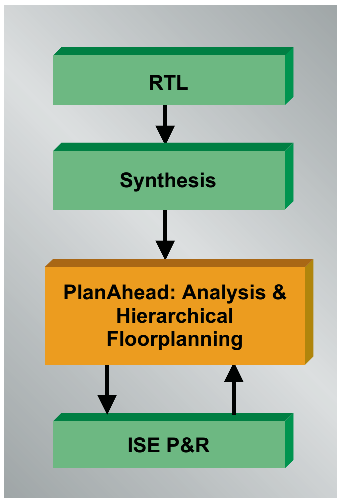


Figure 1 – PlanAhead software fits into your existing design flow

Device Selection

It is easy to explore different devices (within compatible families) for a netlist because PlanAhead tools can open multiple device views (floorplans) on the same netlist. In combination with TimeAhead, this enables you to decide on the speed grade that matches your target performance and the optimal size of your Xilinx part. It can also help you choose your pin package.

Multiple Floorplans

PlanAhead software has the ability to open multiple device views on the same netlist and create a different floorplan in each of the device views. You can analyze each floorplan for utilization and performance characteristics and ultimately run through place and route. This gives you a powerful what-if mechanism for design space exploration.

Statistics

You can generate a detailed statistical report for any block in your design that includes information on clocks, resource utilization, RPMs, carry chains, clock regions, and the number of internal versus external nets. This could help you narrow down problems in your design to specific blocks or modules that you could then re-code or re-synthesize with different options.

Schematic Viewer

A comprehensive schematic viewer can help you navigate timing paths, trace through cones of logic, or determine the floorplanned block connectivity. Some simple net and pin tracing capabilities within the schematic viewer can save you hours wading through VHDL/Verilog files.

Connectivity Display

PlanAhead software bundles all nets connecting any two floorplanned blocks into a single line, with thickness and color reflecting the number of such nets. This bundled connectivity shows the flow of data through the design and is extremely useful in arriving at the "right" floorplan.

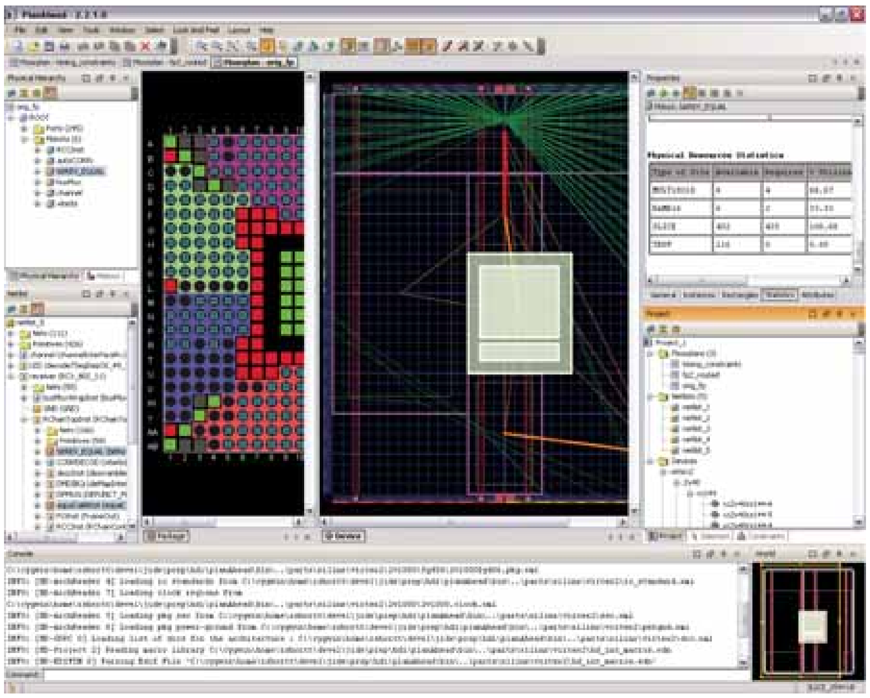


Figure 2 – The PlanAhead analysis environment

Constraints Editing

You can view, edit, add, and delete any of your constraints and see the effect of timing constraint changes by re-running TimeAhead. This eliminates the need to re-run place and route – a lengthy process – just to verify constraint changes. If you decide that certain paths are not important, you can false path them, decide that certain clocks need to be tightened, or add a MAX_DELAY constraint on a long meandering net.

Design Rule Checks (DRCs)

PlanAhead software sports a comprehensive set of design-rule checks to flag potential problems before launching a time-consuming place and route run:

- SSO limit violations
- I/O bank rule violations
- Clock region rules
- Carry chain height
- DSP48 internal register optimization

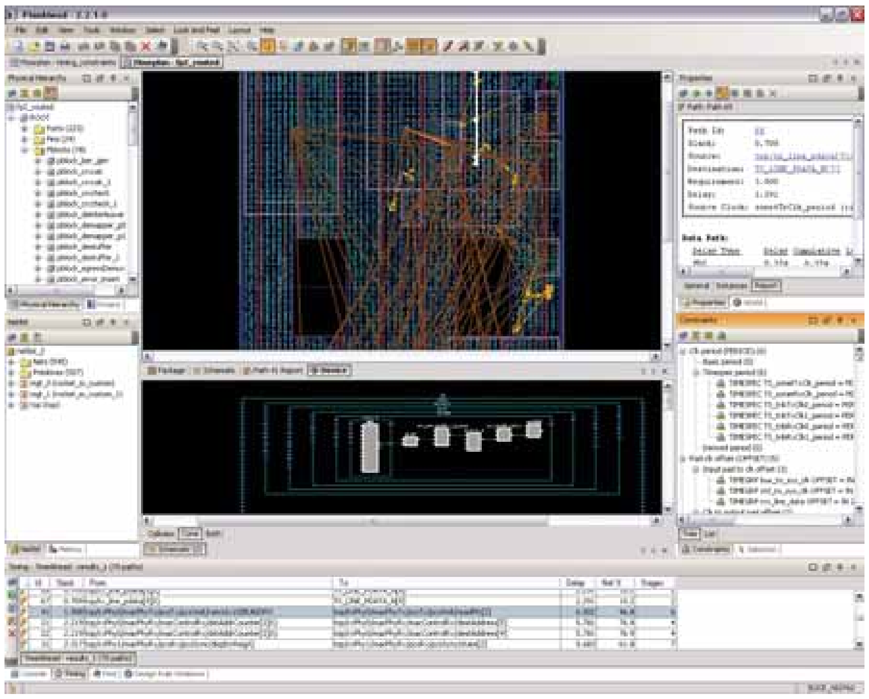


Figure 3 – Analyzing placement with TimeAhead and the schematic viewer

Placement

You can import the placement of your block or entire design and make changes to the placement locations. PlanAhead design tools give you very easy-to-use cross-probing functionality between the netlist tree display, schematic, placement, timing paths, and user floorplanned blocks.

Figure 3 shows the post-place and route environment. The PlanAhead analysis features work together to make it easy for you to comprehend the effect of physical implementation on performance.

Metric Maps

With metric maps (introduced in the latest PlanAhead release), you can detect problem areas such as a cluster of negative slack instances within a certain hierarchical module on the placement surface. Congested areas or utilization issues are also detectable. Better, faster, and smarter visualization puts you firmly in control of the FPGA and leads to faster turnaround times in fixing problems.

Block-Based Design

If you've ever struggled to get a giant FPGA out the door on time, you might have wondered how to break up your design into smaller, more manageable blocks. Imagine a flow that lets you focus your optimization efforts on a critical block, implement it, and verify its performance before tackling the rest of the design. You may have also wished for a flow that enables you to divide up a design between team members.

PlanAhead software was built from scratch to be your hierarchical design tool. At its core lies the PBlock (or Physical Block): a physical design entity that contains one or more logical (RTL or structural EDIF) hierarchical instances.

Simply create a PBlock, assign a region for it, export its netlist and relevant constraints for place and route, import its placement results, and verify its timing. PBlock technology enables several key methodologies.

Flexible Block Partitioning

PBlocks can comprise any number of hierarchical instances or logic elements (LUTs, FFs). PBlocks need not adhere to your original logical hierarchy and can be dynamically redefined whenever necessary throughout your flow. Let's explore the advantages of this technology.

Performance-Based Floorplanning

PBlocks can group logic together to improve performance on your design. Corralling critical paths into smaller regions on the device can potentially improve performance through reduced route delays. You can use any of PlanAhead's analysis features to drive PBlock creation and floorplan generation. Once defined, a floorplan can absorb minor to moderate changes to the netlist to help keep your place and route results predictable.

Block Analysis

If design analysis reveals that your critical paths are contained by a few hierarchical instances, you might consider the following technique. Create a PBlock with these instances and export the PBlock to place and route. PlanAhead software will also export timing constraints relevant to those instances.

Your place and route results serve as a timing sanity check – if the trial PBlock does not meet timing, you are guaranteed that the logic will not meet timing in the global context. It is easy to pinpoint and address the potential showstoppers in your design.

Bottom-Up Flow

Once a PBlock has run through place and route and its results look promising, you can use the placement results for the

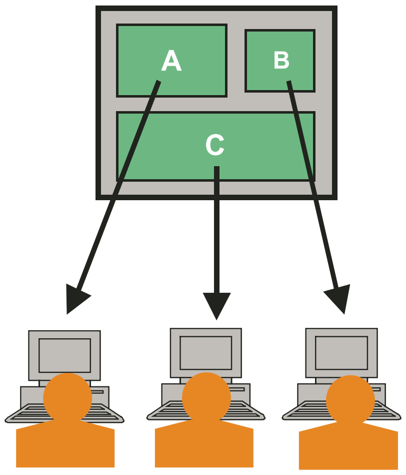


Figure 4 – Block-based team design

PBlock in a powerful team-based design flow. For example, the design manager could divide up the logic into PBlocks and assign one or more to each team member (Figure 4). Team members focus on achieving satisfactory results for each of their assigned PBlocks and return those results to the manager.

The manager then stitches the design back together by importing each PBlock result into PlanAhead software as it arrives. Because PlanAhead design tools do not lock down routing, the design manager will need to launch a final route run to complete the design.

IP Re-Use

PlanAhead software has the ability to export any IP module, which can then be run through place and route. Once you are satisfied with performance, you can save the module's placement in your team's IP library. Another team member can import the IP and its placement into an entirely new design.

PlanAhead design tools also allow IPs to be moved around on the device – the relative placement of all logic elements within the IP is maintained and the net route delays should see minimal variation. The time spent to meet performance on your IP modules need not be repeated for each design using this IP.

Incremental Design

Xilinx ISE™ software's incremental guided implementation flow in place and route requires you to floorplan your design. This guiding flow will re-use your previous implementation results in conjunction with the floorplan to help preserve results in the new implementation. You can use PlanAhead software to easily generate the necessary floorplan for guided incremental flow.

With larger FPGA design sizes, more users are complaining about unpredictability of results and long place and route runtimes. PlanAhead block-based flows put you in control of your larger designs and their increasing complexities.

Table 1 offers several examples of customers who have had marked results after adopting the PlanAhead methodology.

Conclusion

Even though FPGAs continue to grow in size and sophistication, performance requirements and time-to-market pressures remain status quo for FPGA designers. Existing design flows and methodologies are struggling to keep pace, but PlanAhead software provides a revolutionary boost to your current flow.

Intuitive and powerful, its design analysis and block-based hierarchical design capabilities put you in control. To learn more about PlanAhead design tools, visit *www.xilinx.com/planahead/*.

	Device	Before	After
Customer A	XC2V1000	Inconsistent Results: 25 MHz to 60 MHz	Consistently Exceeding 63 MHz
Customer B	XC2V8000	9 Hours Place and Route Runtime: 147 MHz	3 Hours Place and Route Runtime: 172 MHz
Customer C	XC2VP70	54 MHz	102 MHz
Customer D	XC2V6000	Design Time 3 Weeks: 160 MHz	Design Time 3 Days: 178 MHz
Customer E	XC2V4000	120 MHz	170 MHz
Customer F	XC2V4-LX60	93 MHz	109 MHz

Table 1 – Customer designs demonstrate a boost in performance and productivity with PlanAhead software.

Xilinx Events and Tradeshows

Xilinx participates in numerous trade shows and events throughout the year. This is a perfect opportunity to meet our silicon and software experts, ask questions, see demonstrations of new products, and hear other customer success stories.

For more information and the current schedule, visit *www.xilinx.com/events/*.

Worldwide Events Schedule

North America

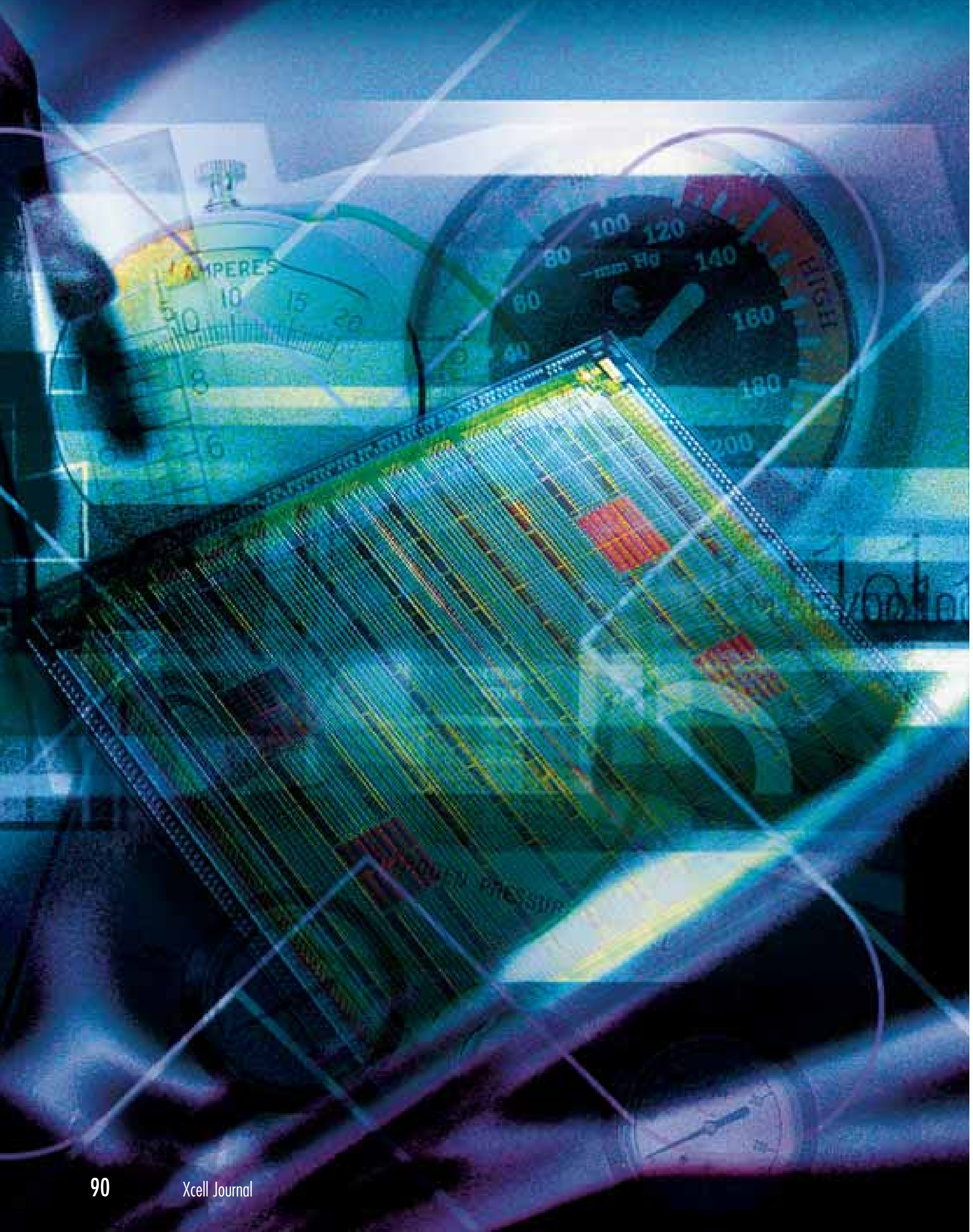
April 21	AdvancedTCA Seminar San Jose, CA
April 21- May 19	Low-Cost Joint Seminar with Synplicity various locations
May 22-25	Wind River 2005 Worldwide User Conference Orlando, FL
June 7-9	SUPERCOMM 2005 Chicago, IL
July 11-15	NSREC 2005 Seattle, WA

Europe, Middle East, Africa (EMEA)

May 11	Mentor Graphics EDA Tech Forum Dresden, Germany
May 11-12	The MathWorks Model-Based Design Conference Warwickshire, UK
May 25	TI Developer Conference Milan, Italy
May 30	TI Developer Conference Tel Aviv, Israel
June 2	JTAG Technologies DFT Boundary-scan Seminar Barcelona, Spain
June 8-9	The MathWorks Model-Based Design Conference Munich, Germany
June 13-14	The MathWorks Model-Based Design Conference Paris, France
June 21	Mentor Graphics EDA TechForum Reading, UK
June 23	Mentor Graphics EDA TechForum Paris, France

Building High-Performance Measurement and Control Systems with FPGAs

National Instruments has customized off-the-shelf FPGA hardware using graphical development tools.



by Mike Trimborn
LabVIEW Embedded Product Manager
National Instruments
mike.trimborn@ni.com

For more than 27 years, National Instruments (NI) has leveraged industry-standard computers, the Internet, and cutting-edge technology to help engineers and scientists build test, measurement, and control solutions. We are a pioneer and leader in virtual instrumentation – a concept that has changed the way engineers and scientists approach measurement and control.

Virtual instrumentation increases productivity and lowers costs through easy-to-integrate software, such as the NI LabVIEW graphical development environment; and modular hardware, such as PCI and PXI modules for data acquisition, instrument control, and machine vision.

NI and Xilinx® have collaborated to take virtual instrumentation one step further by bringing the flexibility and performance of FPGAs to measurement and control I/O.

NI LabVIEW for Virtual Instrumentation

Virtual instrumentation merges software tools with modular hardware to bring you flexible, programmable, and configurable instrumentation. These systems provide you with user-defined I/O functionality, speed, and resolution not available from traditional "box" instrumentation.

When implemented with high-end modular instrumentation and LabVIEW graphical programming, virtual instrumentation costs less and performs better than other commercial off-the-shelf (COTS) solutions.

The NI LabVIEW FPGA module extends the capabilities of LabVIEW software to target FPGAs on NI-Reconfigurable I/O (RIO) hardware. LabVIEW FPGA and RIO hardware devices bring custom timing, triggering, synchronization, and advanced counter I/O to instrumentation end users (Figure 1).

Within the constraints of a COTS hardware platform, LabVIEW provides a high-level application programming interface (API) for software-to-hardware and hardware-to-hardware interconnections so that

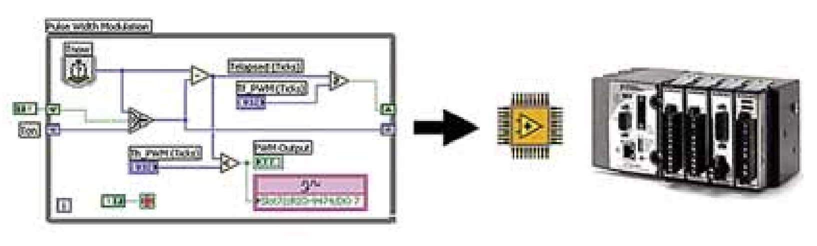


Figure 1 – LabVIEW block diagrams download directly to Xilinx FPGAs on NI CompactRIO.

you don't need to generate your own customized API. With NI LabVIEW and RIO hardware, you can rapidly define custom control logic and I/O lines in a LabVIEW program – called a virtual instrument (VI) – without prior knowledge of low-level development tools. Simply draw your hardware design in LabVIEW and the tool chain synthesizes the gates directly into the Xilinx FPGA (Figure 2).

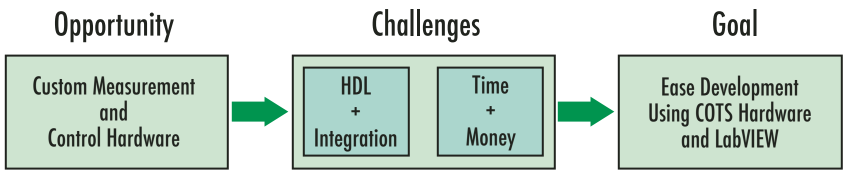

Opportunity
Challenges
Goal
Custom Measurement and Control Hardware
HDL + Integration
Time + Money
Ease Development Using COTS Hardware and LabVIEW

Figure 2 – Using NI LabVIEW saves time and money by eliminating HDL coding complexity and providing an easy-to-use GUI interface.

The intuitive graphical nature of LabVIEW also makes it a unique tool for prototyping Xilinx FPGA designs. You can iteratively change your design blocks in LabVIEW and see these changes quickly converted into actual FPGA behavior. Additionally, you can integrate previously developed VHDL or Verilog code within the LabVIEW block diagram using an HDL input node. Using the HDL input node, you can develop low-level algorithms in HDL and represent that algorithm as a single function block within the LabVIEW environment. This function block can be reused throughout the application or saved for other designs.

LabVIEW also provides a thin driver-level API to interface RIO hardware to the rest of a measurement and control system, simplifying driver development. Therefore, you can easily integrate existing instrumentation, motion, vision, or other COTS hardware into one complete system.

Xilinx FPGAs for Virtual Instrumentation

Several measurement and control instrumentation vendors already use FPGAs because of their performance, fast development time (when compared to other processing technologies), and vendor reconfigurability. However, until recently, this functionality has been out of reach for many instrumentation users. Most of the growth experienced by Xilinx FPGAs in the measurement and control industry has been among instrumentation vendors rather than their customers.

NI has taken the next logical step – bringing FPGA technology to instrumentation users – by using Xilinx Virtex™-II FPGAs on NI RIO hardware to merge LabVIEW's rapid development time with the performance and flexibility of Virtex-II devices (Figure 3).

An obstacle LabVIEW overcomes is that HDL software knowledge, while common among hardware designers, is not as popular among measurement and control users who need flexible architectures for custom I/O, timing, and control logic. Chip-to-chip interconnects require knowledge of low-level timing and communication protocols between components. Furthermore, software and hardware systems often require the design of low-level drivers and APIs to interface a custom FPGA board to the rest of a measurement and control system.

This requirement forces you away from the GUI approach used for most other tasks. Such a cumbersome custom development can require weeks or months and cost in excess of $100,000. With the combined usage of LabVIEW and Virtex-II FPGAs, you can successfully avoid these delays and extra costs.

Another consideration is that typical NI data-acquisition products address a broad range of built-in functionality for timing, triggering, and advanced I/O handling (synchronization and event counting). However, in some applications the timing model or built-in functionality breaks down. In these cases, providing hardware with a user-programmable Virtex-II FPGA becomes advantageous because you can define the device to serve the specific needs of your system and application.

With NI RIO hardware, you also save time and money. Rather than building multiple boards for different applications, you can reduce costs by using only one device and reprogram it for different application needs.

Additionally, because code execution is implemented within the Virtex-II device, the hardware responds faster than software, which is very beneficial in applications requiring high-speed control and tight triggering and synchronization. When imple-

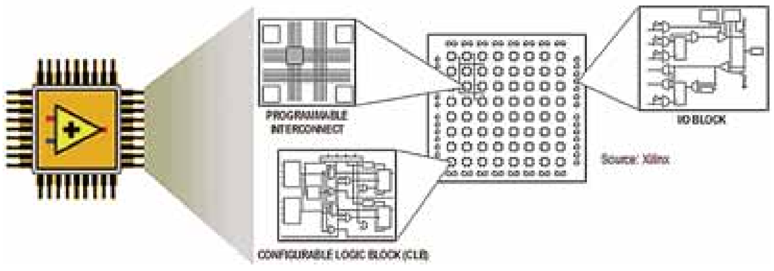


Figure 3 – FPGA hardware architecture on NI-reconfigurable I/O hardware

menting control loops within Virtex-II devices, digital loops can run up to 20 MHz and analog loops up to 125 kHz. These rates are not achievable in implementations using only system-level software.

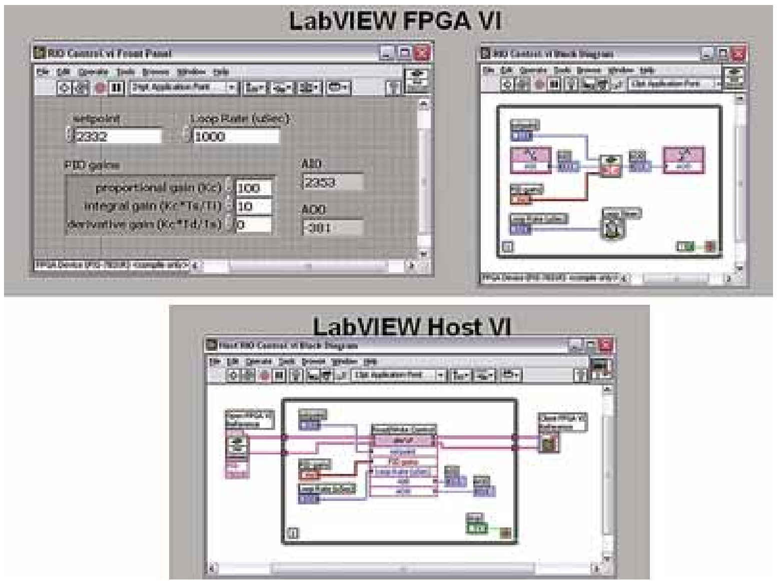
Implementation of LabVIEW FPGA

One of our key objectives during the implementation of LabVIEW FPGA was to preserve the same semantics, level of abstraction, and ease of use between the hardware implementation using LabVIEW FPGA and the desktop version of LabVIEW running on Windows. This allows us to leverage the development skills of our existing LabVIEW developers and migrate code between targets. Furthermore, because LabVIEW has always been a parallel language and was originally designed with programming hardware in mind, its parallelism is an ideal match for programming parallel FPGA hardware.

To implement a LabVIEW block diagram onto the FPGA, we generate a set of state machines (in addition to the functionality described in the blocks of the diagram) that behave as an "enable chain" to control the flow of data. We also make data from the LabVIEW FPGA VI available to external programmers very intuitively by representing this data as controls and indicators on the front panel. You can treat the VI running on the FPGA just as if it were a VI on any other processing engine. We then provide a small set of functions to access the LabVIEW FPGA data from a LabVIEW host VI (Figure 4).

To speed up development and take advantage of Xilinx compiler tools, we decided to generate an intermediate netlist representation from the LabVIEW block diagram in a hardware description language, in our case VHDL. This allows us to represent some control logic at higher levels of abstraction, which speeds up the development of our code generators and makes debugging much easier. And because LabVIEW is a very generic programming environment for the host, we developed the module generators themselves in LabVIEW, enabling us to increase our productivity even further.

Once we had met the objectives of creating identical semantics between LabVIEW implemented on a FPGA and LabVIEW implemented on other processing engines, we turned our attention to enhancing the language constructs to take further advantage of the fact that we are running native on hardware. We optimized our timing with single-cycle timed loops, as the strict use of the enable chain had forced us to have a cycle delay for each function. Using the single-cycle timed loop, we can execute complete loops within one FPGA clock cycle. Customers can also integrate legacy VHDL or Verilog code.


LabVIEW FPGA VI
RIO Control.vi Front Panel
RIO Control.vi Block Diagram
LabVIEW Host VI
Host RIO Control.vi Block Diagram

setpoint 2332
Loop Rate (uSec) 1000
PID gains
proportional gain (Kc) 100
integral gain (Kc*Ts/Ti) 10
derivative gain (Kc*Td/Ts) 0
AI0 2353
AO0 -381

Figure 4 – LabVIEW FPGA block diagram and front panel with corresponding LabVIEW host block diagram

Automotive Application

NI alliance partner GÖPEL Electronic, a leading German PCB and electronic device testing company, created a portable measuring unit for onboard vehicle analysis and diagnostics. They used LabVIEW FPGA and CompactRIO, the latest member of the RIO hardware family, to create a portable, rugged, and extremely versatile handheld data recorder for use in test drives in laboratories, environmental chambers, wind tunnels, and on proving grounds. The system, called CARLOS (in-CAR LOgging System), is also ideal for endurance and long-term tests, as well as vehicle calibration and diagnostics.

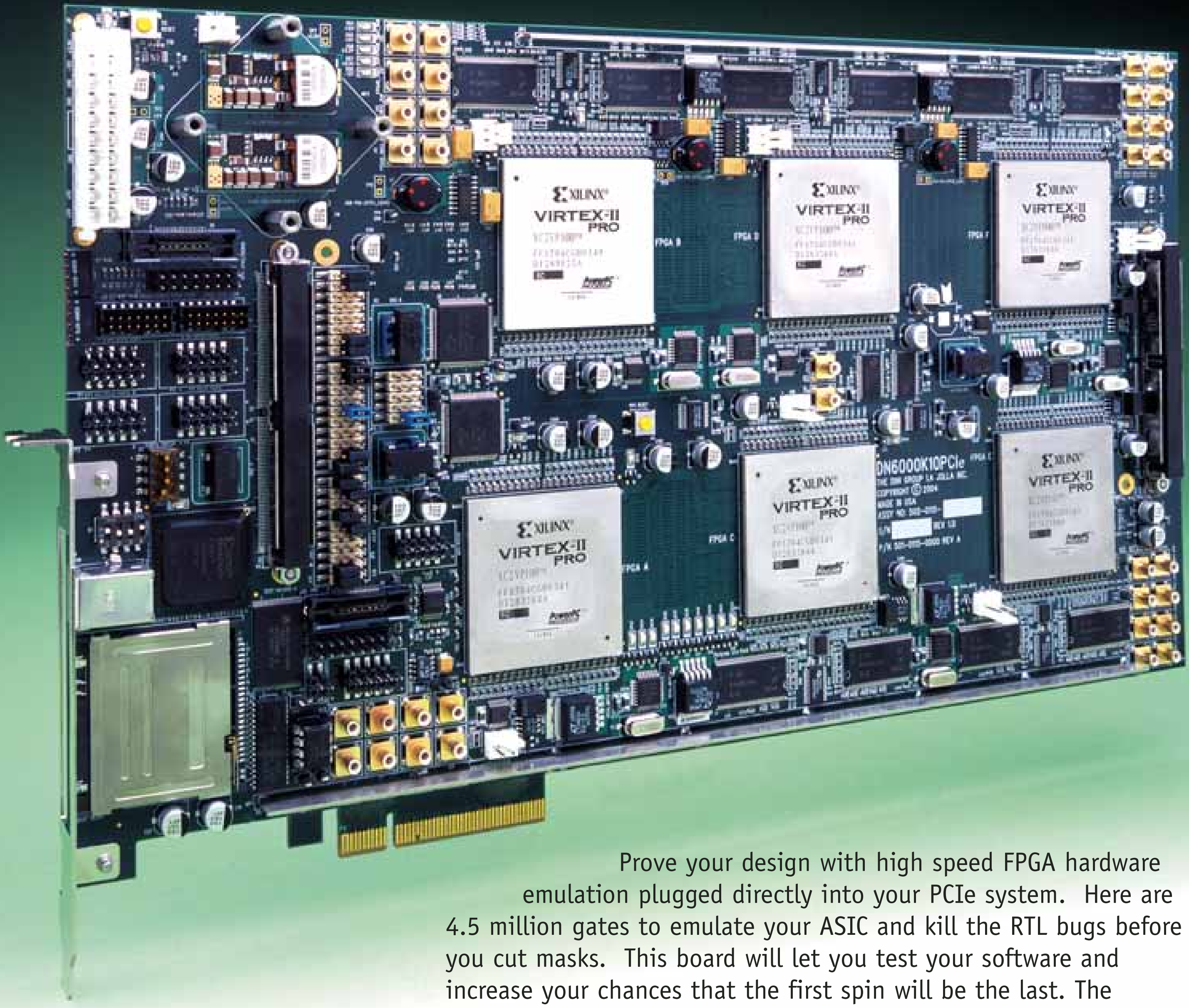
With CARLOS, you can easily and single-handedly navigate all device functions with a jog wheel, buttons, and soft keys. Several project functions embedded in CARLOS help you manage various measuring units simultaneously. Because the system is based on LabVIEW and Virtex-II FPGA technology, you can reconfigure its functionalities far beyond its factory settings to meet a wide variety of testing needs.

Conclusion

Xilinx and NI have collaborated to help measurement and control engineers build high-performance, custom I/O devices leveraging flexible off-the-shelf hardware powered by Xilinx FPGAs at a considerable cost and time savings. While LabVIEW can be used to define FPGAs quickly, the use of Xilinx Virtex-II FPGAs provides reconfigurable customization for many measurement and control platforms from NI.

Within hours, you can turn an off-the-shelf I/O board into a control device that exactly meets your needs without ever knowing VHDL or other low-level hardware design tools. By simply changing the block diagram of the LabVIEW FPGA code, you can recreate hardware with a completely different personality and functionality. This rapidly implemented, reconfigurable I/O approach is unprecedented in the measurement and control industry and continues to expand the capabilities of virtual instrumentation. For more information about LabVIEW FPGA, please visit *www.ni.com/fpga/*.

Prototype Your PCIe ASIC *HERE*



Prove your design with high speed FPGA hardware emulation plugged directly into your PCIe system. Here are 4.5 million gates to emulate your ASIC and kill the RTL bugs before you cut masks. This board will let you test your software and increase your chances that the first spin will be the last. The DN6000K10PCIe is packed with the features you need:

- 1,4 and 8-lane versions
- Six VirtexII-Pro FPGAs (-2vp100s, the big ones)
- 10 DDR (64Mx16) and 4 SSRAMs (2Mx36) external to the FPGAs
- Expansion capability to customize your application
- Synplicity Certify® models for quick and easy partitioning

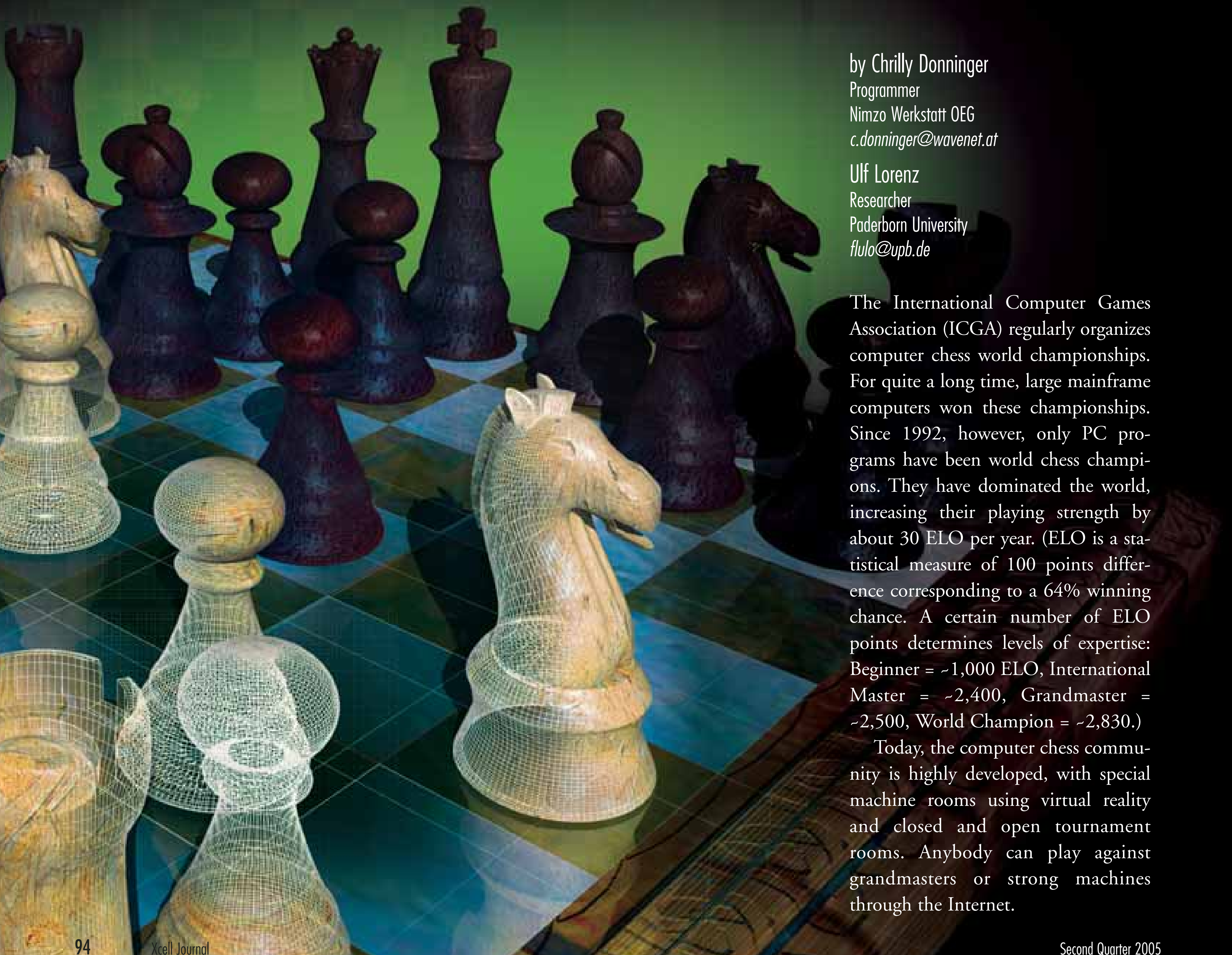
Like all our products, this new PCI Express bus board will help you get your ASIC to market on time and in budget. Call The Dini Group today-- PCIe is already here.

The DiNI Group

1010 Pearl Street, Suite 6 • La Jolla, CA 92037 • (858) 454-3419 • Email: sales@dinigroup.com

The Hydra Project

Hydra, currently the strongest chess program in the world, is a cutting-edge application that combines cluster computing with the fine-grain parallel FPGA world.



by Chrilly Donninger
Programmer
Nimzo Werkstatt OEG
c.donninger@wavenet.at

Ulf Lorenz
Researcher
Paderborn University
flulo@upb.de

The International Computer Games Association (ICGA) regularly organizes computer chess world championships. For quite a long time, large mainframe computers won these championships. Since 1992, however, only PC programs have been world chess champions. They have dominated the world, increasing their playing strength by about 30 ELO per year. (ELO is a statistical measure of 100 points difference corresponding to a 64% winning chance. A certain number of ELO points determines levels of expertise: Beginner = ~1,000 ELO, International Master = ~2,400, Grandmaster = ~2,500, World Champion = ~2,830.)

Today, the computer chess community is highly developed, with special machine rooms using virtual reality and closed and open tournament rooms. Anybody can play against grandmasters or strong machines through the Internet.

Hydra

The Hydra Project is internationally driven, financed by PAL Computer Systems in Abu Dhabi, United Arab Emirates. The core team comprises programmer Chrilly Donninger in Austria; researcher Ulf Lorenz from the University of Paderborn in Germany; chess grandmaster Christopher Lutz in Germany; and project manager Muhammad Nasir Ali in Abu Dhabi. FPGAs from Xilinx® are provided on PCI cards from AlphaData in the United Kingdom. The compute cluster is built by Megware in Germany, supported by the Paderborn Center for Parallel Computing. Taiwan is involved as well.

The goal of the Hydra Project is literally one of world dominance in the computer chess world: a final, widely accepted victory over human players. Indeed, we are convinced that in 2005, a computer will be the strongest chess entity, a world first.

Four programs stand out as serious contenders for the crown:

- Shredder, by Stefan Meyer-Kahlen, the dominating program over the last decade
- Fritz, by Frans Morsch, the most well-known program
- Junior, by Amir Ban and Shay Bushinsky, the current Computer Chess World Champion
- Our program, Hydra, in our opinion the strongest program at the moment

These four programs scored more than 95% of the points against their opponents in the 2003 World Championship.

Computational speed, as well as sophisticated chess knowledge, are the two most important features of chess programs. FPGAs play an important role in Hydra by harnessing the demands on speed and program sophistication. Additionally, FPGAs provide these benefits:

- Implementing more knowledge requires additional space, but nearly no additional time.
- FPGA code can be debugged and changed like software without long ASIC development cycles. This is an important feature of FPGAs, because the evolution of a chess program never ends, and the dynamic progress in computer chess enforces short development cycles. Therefore, flexibility is at least as important as speed.
- We can use a lot of fine-grain parallelism.

Technical Description

The key feature that enables computer chess programs to play as strong as – or stronger – than the best human players is their search algorithm. The programs perform a forecast: given a certain position, what can I do, what can my opponent do next, and what can I do thereafter?

Modern programs use some variant of the Alphabeta algorithm to examine the resulting game tree. This algorithm is optimal in the sense that in most cases it will examine only O($b^{t/2}$) many leaves, instead of b^t many leaves, assuming a game tree depth of t and a uniform branching of b. With the help of upper and lower bounds, the algorithm uses information that it collects during the search process to keep the remaining search tree small. This makes it a sequential procedure that is difficult to parallelize, and naïve approaches waste resources.

Although the Alphabeta algorithm is efficient, we cannot compute true values for all positions in games like chess. The game tree is simply far too large. Therefore, we use tree search as an approximation procedure. First, we select a partial tree rooted near the top of the complete game tree for examination; usually we select it with the help of a maximum depth parameter. We then assign heuristic values (such as one side has a queen more, so that side will probably win) to the artificial leaves of the pre-selected partial tree. We propagate these values up to the root of the tree as if they were true ones (Figure 1).

The key observation over the last 40 years of computer chess data is that the game tree acts as an error filter. The larger the tree that we can examine and the more sophisticated its shape, the better its error filter property. Therefore, what we need is speed.

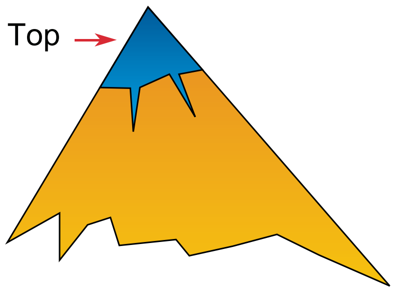


Figure 1 – Game tree search (in the blue part) leads to an approximation procedure.

The Hardware Architecture

Hydra uses the ChessBase/Fritz GUI running on a Windows XP PC. It connects to the Internet using ssh to our Linux cluster, which itself comprises eight dual PC server nodes able to handle two PCI buses simultaneously. Each PCI bus contains one FPGA accelerator card. One message passing interface (MPI) process is mapped onto each of

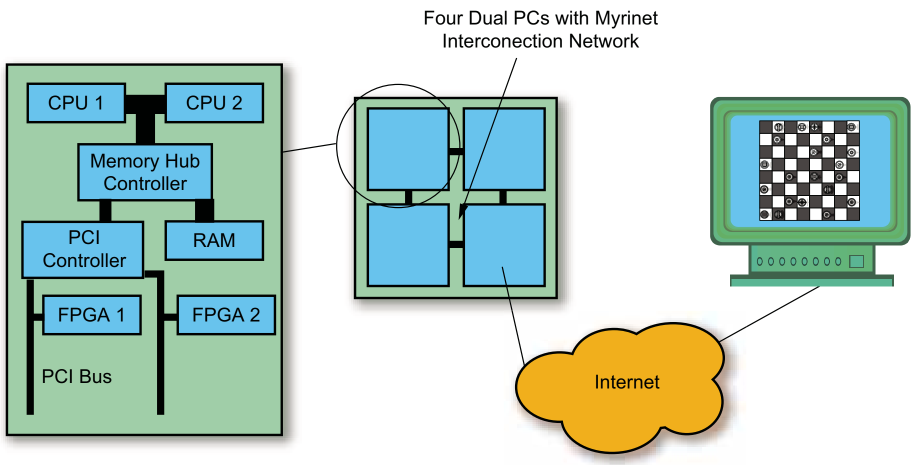


Figure 2 – A cluster of dual PCs supplied with two FPGA cards; each is connected to a GUI via the Internet.

the processors; one of the FPGAs is associated with it as well. A Myrinet network interconnects the server nodes (Figure 2).

The Software Architecture

The software is partitioned into two: the distributed search algorithm running on the Pentium nodes of the cluster and the soft co-processor on the Xilinx FPGAs.

The basic idea behind our parallelization is to decompose the search tree in order to search parts of it in parallel and to balance the load dynamically with the help of the work-stealing concept.

First, a special processor, P_0, gets the search problem and starts performing the forecast algorithm as if it would act sequentially. At the same time, the other processors send requests for work to other randomly chosen processors. When P_i (a processor that is already supplied with work) catches such a request, it checks whether or not there are unexplored parts of its search tree ready for evaluation. These unexplored parts are all rooted at the right siblings of the nodes of P_i's search stack. P_i sends back either a message that it cannot perform work, or it sends a work packet (a chess position with bounds) to the requesting processor P_j. Thus, P_i becomes a master itself, and P_j starts a sequential search on its own. The processors can be master and worker at the same time.

The relationship dynamically changes during computation. When P_j has finished its work (possibly with the help of other processors), it sends an answer message to P_i. The master/worker relationship between P_i and P_j is released, and P_j becomes idle. It again starts sending requests for work into the network. When processor P_i finds out that it has sent a wrong αβ-window to one of its workers (P_j), it makes a window message follow to P_j. P_j stops its search, corrects the window, and starts its old search from the beginning. If the message contained a "cutoff," which indicates superfluous work, P_j just stops its work. We achieve speed-ups of 12 on the 16 cluster entities, which means that Hydra now examines 36 million nodes per second.

At a certain level of branching, the remaining problems are small enough that we can solve them with the help of a Configware coprocessor, benefiting from the fine-grain parallelism inside the application. We have a complete chess program on-chip, consisting of modules for the search, the evaluation, generating moves, and executing or taking back moves. At present, we use 67 block RAMs, 9,879 slices, 5,308 TBUFs, 534 flip-flops, and 18,403 LUTs. An upper bound for the number of cycles per search node is nine cycles. We estimate that a pure software solution would require at least 6,000 Pentium cycles. The longest path consists of 51 logic levels, and the design runs at 30 MHz on a Virtex™-I 1000. We have just ported the design to a Virtex XC2VP70-5 so that we can now run the program with 50 MHz.

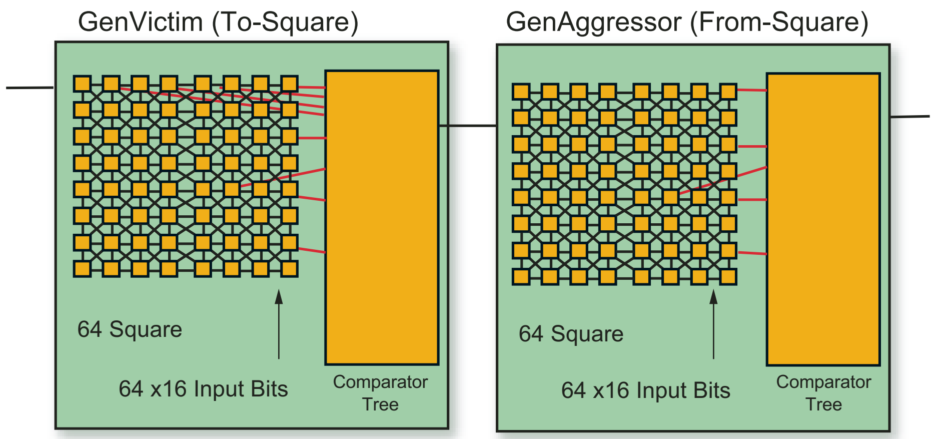
In software, a move generator is usually implemented as a quad-loop: one loop over all piece types; an inner loop over pieces of that type; a second inner loop for all directions where that piece can move; and the most internal loop for the squares to which the piece can move under consideration of the current direction. This is quite a sequential procedure, especially when we consider that piece-taking moves should be promoted to the front of the move list.

In a fine-grain parallel design, however, we have a fast, small move generator, which works very differently. In principle, the move generator consists of two 8 x 8 chess boards, as shown in Figure 3. The GenAggressor and GenVictim modules instantiate 64 square instances each. Both determine to which neighbor square incoming signals must be forwarded.

The square instances will send piece signals (if there is a piece on that square), respectively, forwarding the signals of far-reaching pieces to neighbor square instances. Additionally, each square can output the signal "victim found." Then we know that this square is a "victim" (a to-square of a legal move). The collection of all "victim found" signals is input to an arbiter (a comparator tree) that selects the most attractive not-yet-examined victim.

The GenAggressor module takes the arbiter's output as input and sends the signal of a super-piece (a combination of all possible pieces). For example, if a rook-move signal hits a rook of our own, we will find an "aggressor" (a from-square of a legal move). Thus, many legal moves are generated in parallel.

These moves must be sorted and we have to mask those already examined. The moves are sorted with the help of another comparator tree and the winner is determined within six levels of the tree. Sorting criteria is based on the value of attacked pieces and whether or not a move is a killer move.



1. Occupied squares send a signal in GenVictim
2. Free squares forward these signals
3. All squares receiving a signal are potential to-squares
4. Comparator tree selects most attractive to-square (taking moves).

1. Winner square generates signal of a super piece
2. Free squares forward the signals
3. Squares occupied by own pieces are potential from-squares
4. Comparator tree selects most attractive from-square

Figure 3 – The gen modules form the move generator.

Figure 4 shows the Finite State Machine for the recursive Alphabeta algorithm. On the left side of the figure you can see the states of the normal search, including the nullmove heuristic. The right side shows the states of the quiescence search. The main difficulty is controlling the timing, as some of the sub-logic (the evaluation and the move generator) may need two or three cycles instead of one. We enter the search at FS_INIT. If there is anything to do, and if nullmove is not applicable, we come to the start of the full search.

After possibly increasing the search depth (not shown in Figure 4), we enter the states FS_VICTIM and thereafter FS_AGGR, which give us the next legal move as described previously. Reaching the state FS_DOWN corresponds to a recursive call of the Alphabeta algorithm with a 1-point window (α,α + 1). If the search remaining depth is greater than zero, we look for a move in the state FS_START. Otherwise we enter the quiescence search, which starts with the evaluation inspection. In the quiescence search we only consider capture- and check-evasion moves.

The search stack (not shown) is realized by six blocks of dual-port RAM, organized as 16-bit wide RAMs. Thus, we can write two 16-bit words into the RAM, or one 32-bit word at one point of time. A depth variable controlled by the search FSM controls the data flow. Various tables capture different local variables of the recursive search.

Conclusion

We are quite optimistic that Hydra already plays better chess than anybody else. Nevertheless, we must now show this in a series of matches.

At the same time, we want to maintain the distance between Hydra and other computer players and even increase it. Therefore, in future versions of Hydra, we plan to switch to newer generations of Xilinx FPGAs, increase the number of processors further, and fine-tune the evaluation function.

For more information, visit *www.hydrachess.com* and *www.chessbase.com.*

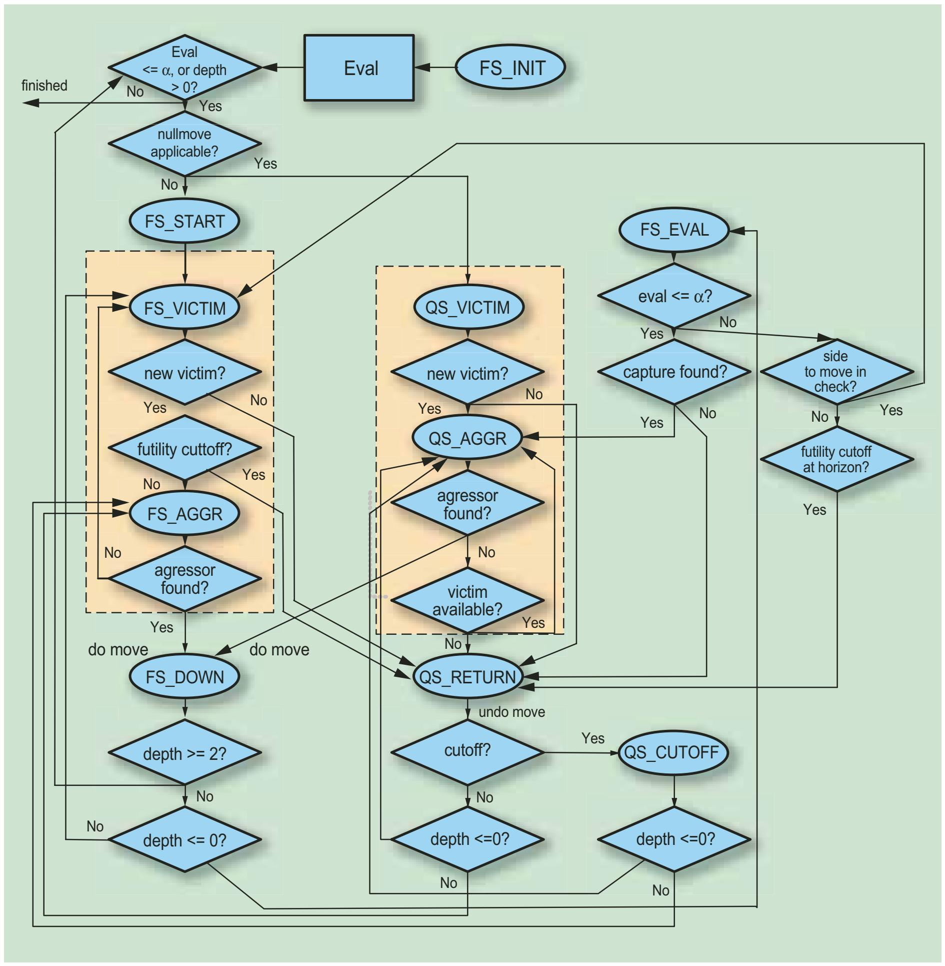


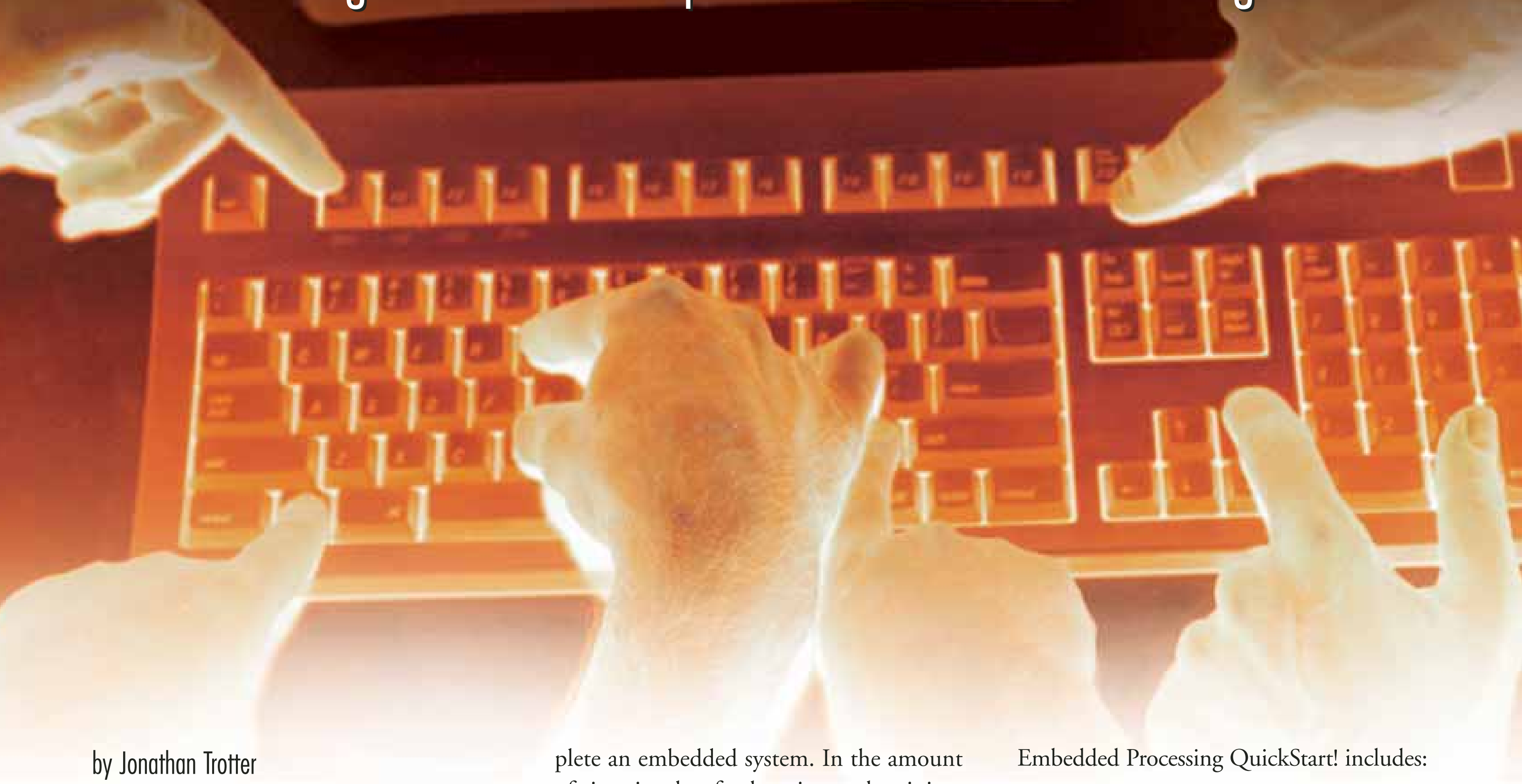
Figure 4 – Simplified flow chart for the 56-state FSM that operates the Alphabeta algorithm

History of Modern Computer Chess

- **1940-1970:** Programmers attempt to mimic human chess style, but resulting programs are weak
- **1970s:** Chess 4.5 is the first "strong" program, emphasizing tree search. It is the first program to win a tournament against humans (the Minnesota Open 1977)
- **1983:** Belle becomes National Master with 2,100 ELO
- **1988:** Hitec wins for the first time against a Grandmaster
- **1988:** IBM's Deep Thought plays on Grandmaster level
- **1992:** The ChessMachine, a conventional PC program by Ed Schröder, becomes World Champion.
- **1997:** IBM's Deep Blue beats Kasparov in a six-game match
- **2003:** Hydra's predecessor wins a human Grandmaster Tournament with 9 out of 11 points, reaching 2,768 ELO
- **February 2004:** Hydra achieves #1 rank in International Paderborn Computer Chess Championship
- **April 2004:** Hydra reaches 2,920 ELO on ChessBase chess server
- **August 2004:** Hydra scores 5.5:2.5 against Shredder and 3.5:0.5 against a 2,650 ELO Grandmaster
- **October 2004:** Hydra is crowned Machine World Team Champion against the human team, performing 2,950 ELO again

Accelerate Time to Market with Embedded Processing QuickStart!

Embedded Processing QuickStart! will give you the essential training to start and complete an embedded FPGA design.



by Jonathan Trotter
Titanium Business Development Manager
Xilinx, Inc.
jonathan.trotter@xilinx.com

Co-designing across processor and logic domains can present a significant challenge for both embedded software engineers and logic designers. To take advantage of the embedded capabilities within the hardware, designers should learn how to properly utilize the Xilinx® Embedded Development Kit (EDK) and how to apply the best methodology to implement its functions.

With time, any design team can overcome these hurdles and successfully complete an embedded system. In the amount of time it takes for learning and training, the competition could release their product first. A productive and effective design team will enable you to ship to your customers before your competitors do.

The Embedded Processing QuickStart! solution delivers individualized service that includes a QuickStart! application engineer at your site for a week. This Xilinx expert will train your team on creating embedded systems and teach them how to optimize supporting FPGA features. The assistance provided will help your team exceed expectations and achieve desired design results.

Embedded Processing QuickStart! includes:

- Configuration of the Xilinx design environment
- An instructor-led embedded systems development course
- Design architecture/implementation consultation and guidance
- Guidance with system partitioning
- Initial design techniques to enable faster and more effective debug and verification
- A comprehensive training plan

Benefits

If you had to choose between five individual devices or five individual functions on one device, you would probably choose the latter. The more dynamic design elements on one device, the more flexibility and options you have. One or all embedded processing solutions – such as the Ultra Controller, PicoBlaze™ soft-core processor, MicroBlaze™ soft-core processor, and the PowerPC™ 405 – are located or can be implemented on one powerful device.

These system components were once separate, discrete, and alone on the board. In the future, there will be a wave of programmable embedded platforms. Why not reunite the lonely CPU with some logic, block RAM, DSP, and connectivity? Instead of all of that functionality linked to a whole board, it can now be self-contained in a single device. The Xilinx embedded solution does not just help you for a single application; our solution will increase the system performance of all of your applications.

A CPU is better equipped to handle some functions than traditional FPGAs. A good example of this is arithmetic functions – CPUs are designed for this purpose. So why not let the processor do what it does best and leave the FPGA to concentrate on the tasks at which it excels? With the Xilinx embedded solution, you can do just that. By adding a processing solution to your design, you can create a more advanced system and provide more features to your customers.

Challenges

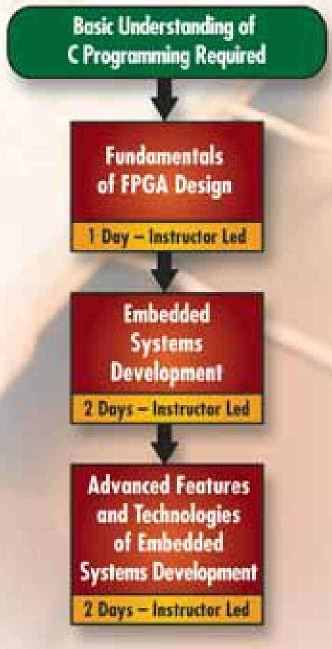
Customers are often reluctant to design with new technology because the time used for training is time not used for designing. The cost of using our flexible, scalable, powerful processing solution is time. And while you're learning the features of our platform FPGA, your competitors are coming up to speed as well. If time to market is critical for you, the risk of your competitors successfully completing their projects before yours can be catastrophic.

The difference between a good design and a great design is time. The difference between a novice design team and a productive, proficient designing machine is simply time. Time yields experience and familiarity. If your team has experience and is comfortable designing with a new technology, they will finish faster. When they encounter obstacles, they will know what steps are needed to proceed.

Embedded Processing QuickStart! Features

With Embedded Processing QuickStart!, Xilinx provides one week of on-site assistance by a trained Xilinx applications engineer. Regardless of your team's competence or skill level, the engineer will set up and customize EDK software for you.

Included during that week of on-site support is a two-day training course in embedded system development. The course covers the benefits of implementing an 8-bit PicoBlaze or a 32-bit MicroBlaze soft-processor core within the Spartan™-3 architecture, and highlights the advantages of the embedded IBM PowerPC core within Virtex™-II Pro and Virtex-4 FPGAs. With the Embedded Development Kit (EDK), you can create any of the combinations that fit your design specifications by applying the techniques taught in the training course.


Basic Understanding of C Programming Required
Fundamentals of FPGA Design
1 Day – Instructor Led
Embedded Systems Development
2 Days – Instructor Led
Advanced Features and Technologies of Embedded Systems Development
2 Days – Instructor Led

After the two-day course, the engineer remains on site to assist the design team in transferring what they learned in the course to a practical design environment. Most engineering teams will have quite a few lingering questions after such an in-depth course. With the trainer on-site for several days afterwards, these questions can be answered.

Another major feature of Embedded Processing QuickStart! is assistance with system partitioning. Some applications are better suited for a processor and others are better suited for FPGA fabric. To obtain optimum performance, your design team will be able to determine which functions to implement to logic and which functions should be executed by the CPU. This is critical for any embedded FPGA design. With proper hardware/software partitioning, your design can achieve superior performance.

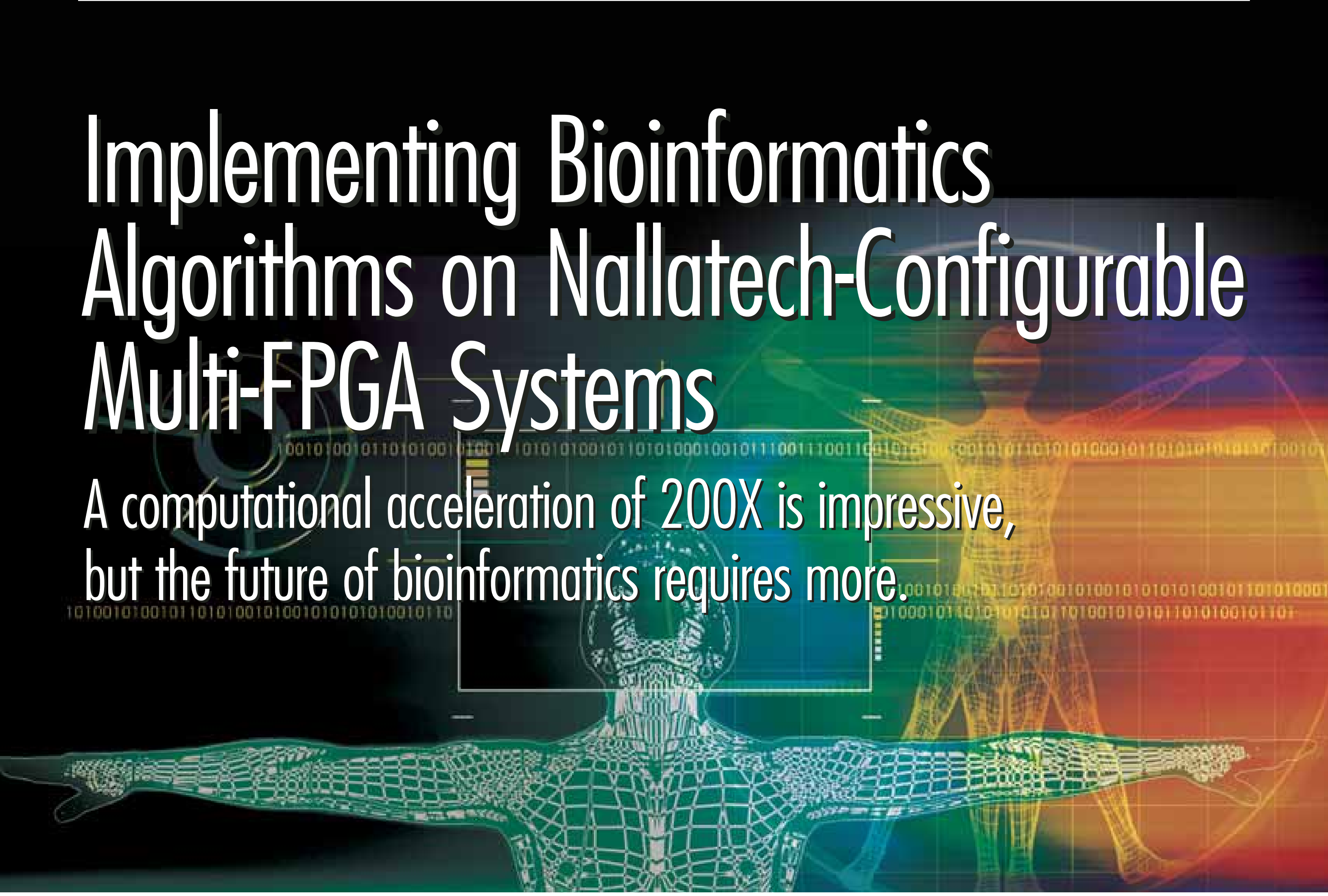
The on-site engineer management leaves behind a training plan tailored specifically to the needs of your design team, which can include system partitioning, CPU programming, C programming, debugging, and verification. This plan helps prevent schedule slips later in the project by ensuring that your team is skilled in the required disciplines. It also helps maintain a more effective and highly motivated team that is properly equipped and confident to handle any obstacle.

Conclusion

Software engineers must design across logic domains and hardware engineers must learn how to design in the software domain. The effect of these challenges lengthens the development schedule and can affect time to market.

The Embedded Processing QuickStart! solution offers an unprecedented level of on-site design support and training for the critical initial design phase of your project. This service will not only show your team where to begin; it will also empower them to complete the project on time and on budget.

For more information about Embedded Processing QuickStart!, contact your Xilinx representative, or go to *www.xilinx.com/epq/*.



Implementing Bioinformatics Algorithms on Nallatech-Configurable Multi-FPGA Systems

A computational acceleration of 200X is impressive, but the future of bioinformatics requires more.

by Keith Regester
Systems Applications Engineer
Nallatech
k.regester@nallatech.com

Jong-Ho Byun
Research Assistant
University of North Carolina at Charlotte
jbyun1@uncc.edu

Arindam Mukherjee
Assistant Professor
University of North Carolina at Charlotte
amukherj@uncc.edu

Arun Ravindran
Assistant Professor
University of North Carolina at Charlotte
aravindr@uncc.edu

In the past decade there has been an explosive growth of biological data, including genome projects, proteomics, protein structure determination, cellular regulatory mechanisms, and the rapid expansion in digitization of patient biological data. This has led to the emergence of a new area of research: bioinformatics, where powerful computational techniques are used to store, analyze, simulate, and predict biological information.

Although raw computational power as predicted by "Moore's Law" has led to the number of transistors that can be integrated doubling every 18 months, the genomic data at GenBank (the NIH genetic sequence database, an annotated collection of all publicly available DNA sequences) is doubling every six months. Proteomic and cellular imaging data is expected to grow even faster. Post-genomic-era bioinformatics will require high-performance computing power of the order of several hundreds of teraflops or more.

In recent years, FPGAs have emerged as high-performance computing accelerators capable of implementing fine-grained, massively parallelized versions of computationally intensive algorithms. The reprogrammability of FPGAs enables algorithm-specific computing architectures to be implemented using the same hardware resource across a range of algorithms. In this article, we'll describe the design flow, test results, and system optimization for implementing a Smith-Waterman algorithm using a scaleable Nallatech FPGA computing architecture.

Sequence Matching Overview

The objective of bioinformatic sequence matching is to identify similarities between subsequences of strings as far as possible. The dissimilarity of two sequences of nucleotides or amino acids can be defined as the minimum change required in one to get the other one, among all possible alignments between them – this is the edit distance of the pair of sequences.

Note that it depends on the choice of scoring scheme. For instance, a possible scoring scheme can assign 0 values to matches of nucleotides at certain positions in the sequences, 1 to gaps (insertions or deletions) at those positions, and 2 to mismatches (substitutions) at those positions.

The Smith-Waterman algorithm computes the local alignments or similarities between substrings of two sequences A and B of lengths m and n, respectively, using a dynamic programming approach. Dynamic programming is a strategy of building a solution gradually using simple recurrences.

The key observation for the alignment problem is that the similarity between sequences A[1..n] and B[1..m] can be computed by taking the maximum of the three following values:

- The similarity of A[1..n -1] and B[1..m -1] plus the score of substituting (sub) A[n] for B[m]
- The similarity of A[1..n] and B[1..m-1] plus the score of deleting (del) A[n]
- The similarity of A[1..n] and B[1..m -1] plus the score of inserting (ins) B[m]

To solve the problem with this recurrence, the algorithm builds an (n +1) x (m +1) matrix M, where each M[i, j] represents the similarity between sequences A[1..i] and B[1..j]. The first row and the first column represent alignments of one sequence with spaces. M[0, 0] represents the alignment of two empty sequences, and is set to zero. All other entries are computed with the following formula:

$$M[i, j] = \min \{M[i-1, j-1] + \mathrm{sub}(A[i], B[j]); M[i-1, j] + \mathrm{del}(A[i]); M[i, j-1] + \mathrm{ins}(B[j])\}$$

Because there are (m+1) x (n+1) positions to compute and each takes a constant amount of work, this algorithm has a time complexity of O (n2). Clearly, it also has quadratic space complexity because it needs to keep the entire matrix in memory.

FPGA Architectures and Networking Tools

The FPGA network hardware comprises Nallatech FPGA computing cards containing between one and seven FPGAs per card. Several such cards can be plugged into the PCI slots of a desktop server and networked using Nallatech's DIMEtalk networking software, greatly increasing the available computing capability. Let's briefly describe the different FPGA network components.

High-Capacity Motherboard – BenNUEY

The BenNUEY motherboard features a Xilinx® Virtex™-II FPGA and module sites for as many as six additional FPGAs. The PCI/control and low-level drivers abstract the PCI interfacing, resulting in a simplified design process for designs/applications.

Virtex-II Expansion Module – BenBlue-II

The BenBlue-II DIME-II module provides a substantial logic resource ideal for implementing applications that have a large number of processing elements. Through support for as many as two on-board Xilinx XC2V8000 FPGAs, the BenBlue-II can provide more than 200,000 logic cells on a single module. A Virtex-II Pro version of this module is also available, providing two XC2VP100 devices.

Multi-FPGA Management – DIMEtalk

To manage the large silicon resource pool provided by the hardware, the DIMEtalk tool accelerates the design flow for creating a reconfigurable data network by providing a communications channel between FPGAs and the host user environment.

FUSE Tcl/Tk Control and C++ APIs

FUSE is a reconfigurable operating system that allows flexible and scalable control of the FPGA network directly from applications using the C++ development API, which is complemented by a Tcl/Tk toolset for scripting base control.

Merging Algorithms and Networks

The design flow consists of two interdependent tasks:

- Capture of the Smith-Waterman algorithm in VHDL
- Creation of the FPGA network

The user VHDL design maps the Smith-Waterman algorithm to a systolic array structure. As shown in Figure 1, the array structure comprises computational units known as processing elements (PEs) that have local interconnections between them. Each element of the query sequence is hard-coded into a PE to achieve an area-efficient realization of approximately four slices.

Note that the length of the maximum length of the query sequence corresponds to the maximum number of PEs that can fit inside a given Virtex-II/Pro device. A C-based routine instantiates the VHDL code of a PE multiple times to create the array of interconnected PEs. Additional VHDL code is then inserted to interface with the FPGA network.

DIMEtalk networks are defined early in the application development phase using the supplied network design software tool, DTdesign. This allows the structure of the network to be outlined across the FPGAs in a system.

Having defined the network, Smith-Waterman blocks of design (VHDL or VHDL-wrapped source) are imported using the custom component insertion wizard and interconnected to the network in the DTdesign workspace.

FPGA I/O ports for the whole design can be mapped to the device pins using the high-

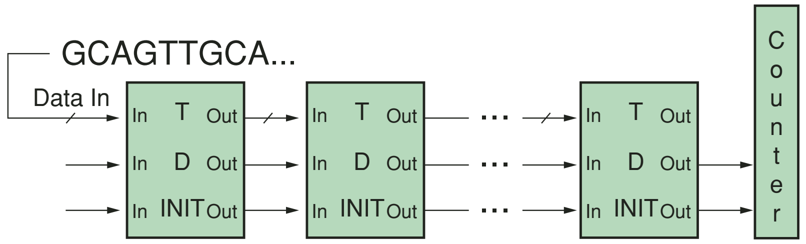


Figure 1 – Systolic array of Smith-Waterman processing elements

level drag-and-drop device editor. DTdesign then automatically generates VHDL code, user constraint files, and a VHDL test bench and configuration script files.

DIMEtalk components include:

- Routers that direct data around the network and interconnect all other component types
- Bridges that move data between different physical FPGA devices
- Nodes serving as the user interface to the network that can be connected to user FPGA designs, through node interfaces such as block memories (BRAMs), queues (FIFOs), and memory maps
- Edges serving as the user interface to the host computer

Note that the sizes of memory elements, such as BRAMs and FIFOs, depend on the application requirements and are easily interchangeable in the DIMEtalk GUI environment. Figure 2 shows a block diagram of components within the FPGA.

The entire network, including the Smith-Waterman application, is then synthesized using the Xilinx ISE™ 6.2i flow, and the place and route timing netlist is generated. Functional and timing simulations verify the operation of the design. Thereafter, the bitstreams for the entire multiple FPGA network are generated within the DIMEtalk environment and placed in corresponding work directories. Figure 3 shows a block diagram of a multi-FPGA network using five XC2V6000 devices as designed in DIMEtalk.

The bit files are then downloaded to the designated FPGAs using FUSE C++ API calls. In runtime, dedicated Tcl/Tk software API functions provide runtime host connectivity. This enables direct communication from the host system to all nodes within the network.

The database sequences downloaded from the GenBank database are formatted using a C-routine for input to the FPGA network. The formatting consists of coding the individual nucleotide bases using 4 bits ((A – ux00, C – ux01, G – ux10, T – ux11), where the bit denoted by "u" is undefined and the one denoted by "x" is set to 1 for the starting nucleotide base of a sequence (set to 0 otherwise). The sequences are then packed into groups of eight to fully use the 32-bit DIMEtalk networks. The length of an individual sequence could be well over 200,000 nucleotide bases long.

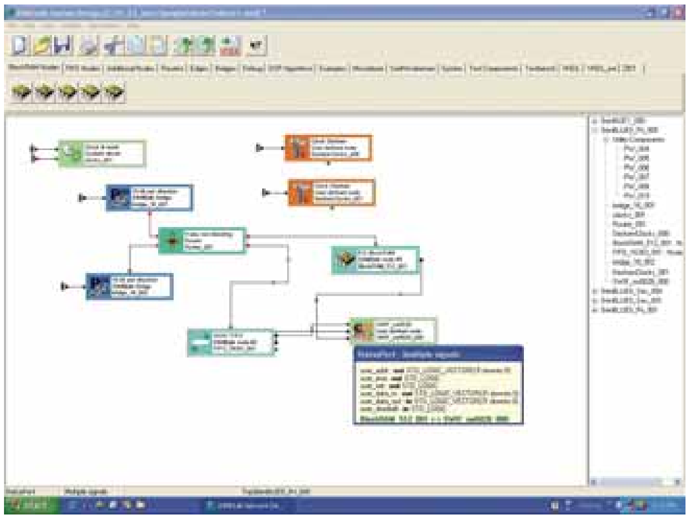


Figure 2 – Screen capture of network components and Smith-Waterman algorithm using DIMEtalk

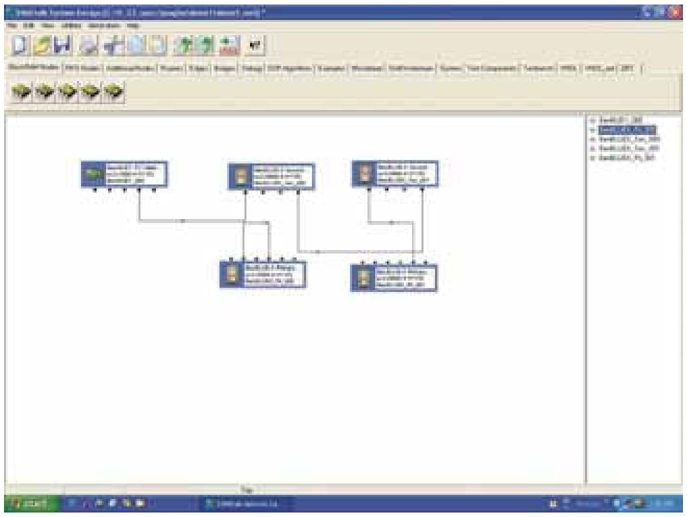


Figure 3 – Screen capture of a multi-FPGA network designed using DIMEtalk

Test Results

The initial test utilizes three of the five FPGAs from the network. The resulting multi-FPGA network comprises a BenNUEY motherboard with a Virtex-II XC2V3000-4 FPGA and an attached BenBLUE-II daughterboard with two Virtex-II XC2V6000-4 FPGAs.

The query sequences are hard-coded into the FPGAs, while the individual database sequence files are loaded in the main memory. The database sequences are first written across the DIMEtalk network into the 16K read-FIFOs of the individual FPGAs. Note that a single XC2V6000 device is capable of as many as 1.26 trillion cell updates per second. At the end of the processing, the final edit distance is written into a write-FIFO.

Table 1 gives the runtimes for the different GenBank databases in the case of a single XC2V6000 FPGA, and two XC2V6000 FPGAs in the network. The databases used were the GBUNA (3 MB, 1,276 sequences in one database file), GBPRI20 (187 MB, 2,036 sequences with each sequence in a separate database file) and GBROD10 (245 MB, 1,200 sequences with each sequence in a separate database file), all from the GenBank database.

The results are compared with those obtained using a SunFire 280R (two UltraSPARC III processors clocking at 1.05 GHz, with 8 MB L2 cache and 8 GB memory, running Solaris 9). Each XC2V6000 FPGA holds PEs corresponding to a 5,000-nucleotide-bases-long query sequence, besides a 10% overhead for the interface logic. We achieved an overall FPGA utilization of more than 80%.

Systems Optimization

The time required to compare a query sequence against a database sequence for the individual FPGA configuration is calculated as:

$$t_{db} = t_{clk} \times \frac{T}{N} \times (N \times |D| + |Q|)$$

Computation Platform	Time (GBUNA)	Time (GBPRI20)	Time (GBROD10)
Single XC2V6000	120 seconds	294 seconds	216 seconds
Network of two XC2V6000	90 seconds	220 seconds	175 seconds
SunFire 280R	265 seconds	11 hours, 24 minutes	7 hours, 45 minutes

Table 1 – Performance of the FPGA network for GenBank databases

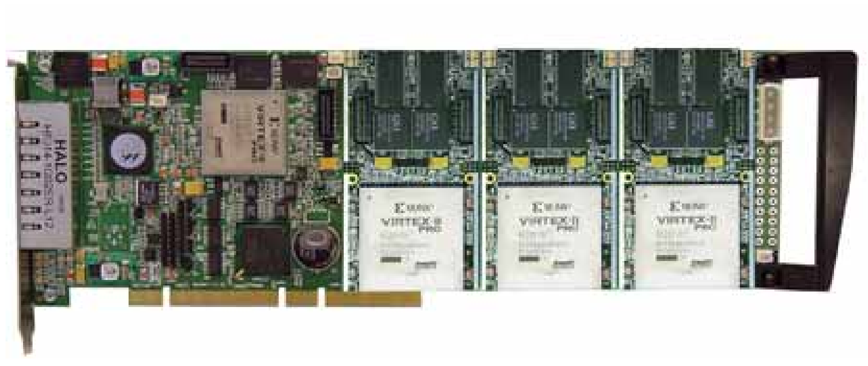


Figure 4 – The BenNUEY-PCI-4E motherboard configured with three BenDATA-II modules

Where |*D*| is the number of elements in a given database sequence D;

|*Q*| is the length of the query sequence Q (equal to the number of processing elements); *T* is the number of nucleotide sequences in the database; *N* is the number of FPGAs in the network; and t_{clk} is the clock period in the FPGA network.

A truly parallel FPGA network would have a processing time given by:

$$t_{db} = t_{clk} \times \frac{T}{N} \times (|D| + |Q|)$$

while a serial FPGA network would have a processing time given by:

$$t_{db} = t_{clk} \times T \times (|D| + |Q|)$$

In tasks where the processing is I/O-limited (|D| > |Q|), the FPGA network acts as a serial system, whereas in tasks where the processing is computation-limited (|Q| > |D|), the network behaves as a parallel system. Note that the use of large on-board memories relieves the I/O bottleneck.

For example, the Nallatech BenDATA-II Module and BenNUEY-PCI-4E motherboard adds 1.5 GB of on-board memory. In addition, the BenNUEY-PCI-4E motherboard has 4 GB Ethernet ports and eight Xilinx RocketIO™ ports, allowing direct connection to storage area networks (SANs), networked memory devices, and interlacing of MGTs between boards.

These Nallatech boards, which became available in early 2005, would be ideally suited for real-world bioinformatics applications (Figure 4).

Automatically evolving computational resource partitioners are currently being developed at UNC Charlotte for optimal resource allocation for a given computational problem and an FPGA network configuration. Figure 5 shows a system-level view of mapping an enormous sequence-matching application to a multi-device FPGA computing platform.

Conclusion

Looking forward, using Nallatech's scalable FPGA networks will help realize high-performance computing capability for bioinformatics applications at a low cost. A practical system will be evaluated on:

- Scaleability to execute diverse complex algorithms
- Ease of implementation and integration
- Price/performance demands

Our prototype implementation of the Smith-Waterman sequence alignment algorithm provides a computational speed improvement of more than two orders of magnitude (200X) while running on a three-FPGA network.

To find out more about Nallatech's scalable FPGA computing architectures and networking tools, visit *www.nallatech.com.* To discuss how Nallatech can support your FPGA acceleration and systems development needs, please contact Nallatech at *contact@nallatech.com* or call 1 (877) 44-NALLA.

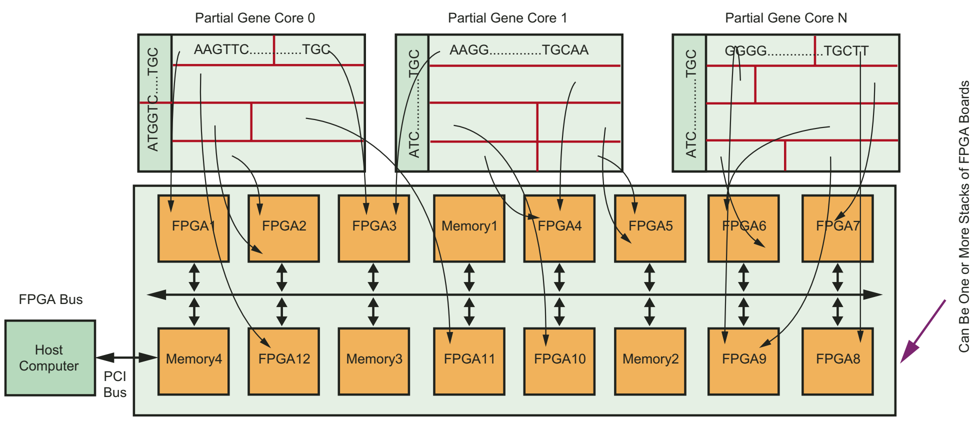
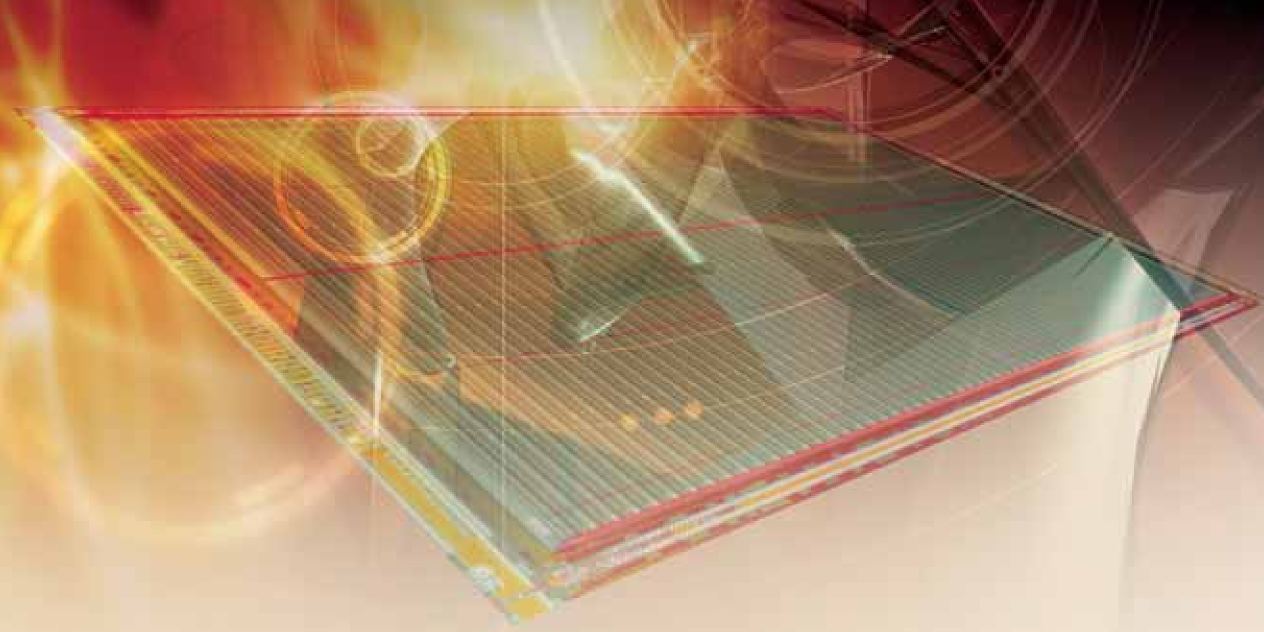


Figure 5 – Resource allocation in a multi-FPGA network

Conquering the Three Challenges of Power Consumption

Why is power such an issue?



by Steve Sharp
Sr. Manager, Corporate Solutions Marketing
Xilinx, Inc.
steve.sharp@xilinx.com

As chip technology progresses to 90 nm and below, power becomes a burning issue in system design. At this node, leakage plays a more major role in total power; smaller interconnect geometries with new dielectric materials affect dynamic power as well.

According to Jordan Selburn of market research firm iSupply, "Leakage current – essentially insignificant at the 0.35 micron node and earlier – has become a major issue as transistors become increasingly leakier. Studies have shown that at the 90 nm node, leakage power can equal dynamic power consumption and even exceed it at the 65 nm node."

Another factor facing system designers is the tighter power budgets around which they must design. This is not limited to any single type of system, but does affect most designers. Large systems with many boards or modules, as well as portable and consumer products, all face power budgeting issues.

In large systems, power budgeting is typically done for the total system, as well as distributed power regulation on a per-board or per-module basis. With multiple power supplies now on every board, it is not a simple task to increase the power budget for one board without affecting the entire system's power distribution plan.

In line-powered consumer products, the goal is usually to use the smallest and least-expensive power supply possible to keep costs under control. Exceeding the capabilities of a particular model power supply by only a few percent can necessitate the use of a larger, more expensive supply, and this might be unacceptable in light of total system cost. Designers would rather design in more features to differentiate the product than to use a larger power supply.

In portable consumer products, the overwhelming goal is to extend battery life for as long as possible. For these products, longer battery life – both in active and standby modes – is a significant competitive advantage.

With all of these challenges, it's no wonder that power issues are sounding the alarm bells for system designers today. iSuppli's Selburn continues, "On the customer side, chip designers can consider architectural approaches such as parallel processing at reduced clock speeds to reduce dynamic power, or gated clocks that essentially turn off entire sections of the chip when they are not needed. Despite these techniques, power consumption remains a serious issue for a large portion of the core silicon market, an issue that is becoming worse, not better, with time."

System Design Challenges

There are three key areas of power usage and control challenging system designers today: static power, dynamic power, and in-rush power. Each presents different issues and requires different methods to calculate and manage power.

Static power is the power consumed by a device when it is in its quiescent condition with no input signals being exercised. It is also referred to as steady-state or standby power. In today's 90 nm technology devices, leakage currents in the transistors are the biggest contributors to static power. This is usually the key parameter of concern to designers of portable equipment because of its effect on battery life, especially for devices that spend large amounts of time in a standby condition waiting for input from the outside world.

Dynamic power is the power consumed during normal operation. It is also referred to as operating power. Dynamic power is dependant on operating signal frequency, interconnect capacitance, and operating voltage. Because the voltage dependency is a square function, the reduction in voltage when moving to 90 nm devices has substantially reduced operating power in many devices. However, for large, high-performance systems with high operating frequencies, dynamic power is still a significant component of total system power.

In-rush power is the power required at device power-up. It is also referred to as power-up or start-up power, or power-on surge power (or current). Some devices require many times more power to begin operation than they do during normal operation, thereby placing demands on system power supplies. In a consumer system with very tightly controlled power supply size and cost, ensuring that in-rush power is not more than normal operating power is a key design goal.

Higher power levels can affect both manufacturers and end-customers alike, in four key areas:

- Performance. Higher power levels in a chip can limit device and end-system performance by forcing a lower system clock rate to stay within the system power budget.
- Reliability. As power goes up, so does the threat of brown-out and latch-up from high power-on surge. In addition, higher failures-in-time (FIT) rates will be expected due to higher device operating temperatures.
- Cost. As mentioned previously, higher power equals higher cost in the system because of larger, more expensive power supplies and thermal management components such as fans and head sinks.
- End-customer operating expenses. Higher power also impacts end users in the form of higher power bills (which can be significant for large systems) and shorter battery life for portable products.

Higher power levels in a chip can limit device and end-system performance by forcing a lower system clock rate to stay within the system power budget.

How Xilinx Helps Manage System Power

Virtex-4 FPGAs

With a significant reduction in power consumption over that of the competition, the new Virtex™-4 platform FPGAs offer significant benefits for system design, including reduced thermal concerns, easier power-supply design, lower cost power supply, and higher system reliability. Virtex-4 FPGAs dramatically reduce power consumption when compared to other FPGAs in all three key power areas:

- As much as 73 percent lower static power with the industry's first triple-oxide technology
- As much as 86 percent lower dynamic power enabled by embedded IP blocks
- Negligible in-rush current with unique power-saving configuration circuitry

This is enabled with industry-leading technologies such as 90 nm triple-oxide technology, high-performance embedded IP, and power-saving configuration circuitry.

Xilinx also provides comprehensive tools for power system design: Virtex-4 datasheet and user guide; a web-based power estimator; and XPower, included in ISE™ software.

Virtex-4 devices handle the three types of power usage and control in the following ways:

- Static power. As process geometries shrink to 90 nm and lower, the industry expects higher leakage and higher static power when channel length decreases. Working with fab partner United Microelectronics Corp., Xilinx solved this problem by using triple-oxide technology in the Virtex-4 90 nm process, which reduces leakage current significantly. Two-oxide thicknesses are widely used in the industry today, with a thin oxide in the core and thicker oxide in the I/O area. Virtex-4 devices add a third medium-thick oxide transistor used for certain functions in the FPGA. The result is 50% lower static power than that of Virtex-II Pro FPGAs. Other FPGA vendors have gone the other way when migrating to a 90 nm process, with static power increasing more than 2X compared to 130 nm devices.
- Dynamic power. New and existing Virtex-4 embedded functions lower dynamic power by 5 to 20x compared

to Virtex-II Pro FPGAs. This results in as much as 86% lower dynamic power than that of other 90 nm FPGAs.

Note these specific examples:

- PowerPC™ – as much as 86% power reduction
- Block RAM – as much as 82% power reduction
- DSP – as much as 23% reduction with XtremeDSP™ slice
- Ethernet MAC – as much as 83% power reduction
- Logic – although Virtex-4 devices consume similar dynamic power-per-logic cell when compared to other FPGAs, the embedded IP blocks often allow fewer general-purpose logic cells to be used. For example, when building a source-synchronous I/O (SSIO) interface, the new ChipSync™ block reduces the number of logic cells used.

• In-rush power. Other high-performance 90 nm FPGAs have exhibited levels of in-rush power more than four times that of Virtex-4 FPGAs. In Virtex-4 devices, by spending considerable time designing very power-efficient configuration logic, Xilinx has been able to keep in-rush power within 15-20% of the static power requirements and below typical operating power. This removes the need to use a larger power supply just to address in-rush current.

CoolRunner-II CPLDs

When Xilinx designed the CoolRunner™-II family of low-power CPLDs, our goal was to deliver one of the industry's lowest power levels for a programmable logic device. These devices have standby current requirements of less than 20 µA, making them ideal for battery-powered portable devices. Other CPLDs claiming to be low power have standby power 100 to 1000x higher, affecting battery life so significantly that they are unsuitable for portable applications.

The static RealDigital technology used in the logic of CoolRunner-II devices does away with power-hungry sense amplifiers and delivers low dynamic power as good as any other device available today.

In addition to these advantages in the basic circuit design and process technology, CoolRunner-II devices also offer power-management features unique to the CPLD industry, including a DataGate feature to reduce effective logic usage in the device and clock management and input hysteresis features to reduce internal operating frequencies and dynamic power.

Spartan-3 FPGAs

Our customers have told us that in today's cost-conscious consumer products, being forced to put in a bigger supply just to supply a high power-on or in-rush current is not a viable option for their system designs.

Attention to detail when designing the Spartan-3 configuration logic has yielded devices where the maximum quiescent power alone is guaranteed to be sufficient to power up the device. Spartan-3 devices have no in-rush current or power specification. When using these low-cost devices, you can focus on the product features and design without worrying about increased system cost because of high in-rush power requirements.

Power Management Tools

Web Power Tools are pre-implementation tools that estimate a design's power consumption based on the expected utilization of device resources, operating frequencies, and toggle rates.

Once you have implemented your design in the Xilinx software tools, you can use XPower to accurately estimate the power consumption. Actual power consumption must be determined in-circuit under the appropriate operating conditions.

Web Power Tools

The intuitive interface guides you through the steps of the data-entry process and ensures the most accurate estimates possible. The equations and values used by Web Power Tools are based on device characterizations for the family. Web Power Tools are available for the Virtex-4, Virtex-II Pro, Virtex-II, Virtex/Virtex-E, and Spartan-3 FPGA families, as well as CoolRunner-II CPLDs.

XPower

XPower is the first power-analysis software available for programmable logic design, allowing the analysis of total device power, power-per-net, routed, partially routed, or un-routed designs.

Power Management Hardware

Tools for managing power are not just of the software variety. Power-management chips from National Semiconductor, Intersil, Texas Instruments, and Linear Technology are available to make the job of supplying the multiple supply voltages needed by today's FPGAs easier, and they can be valuable companions to Virtex-4 or Spartan-3 devices. Their individual capabilities are highlighted in this issue of the *Xcell Journal* in the following pages.

Conclusion

To conquer the key challenges of power consumption, it takes a combination of good product design, proper device technology, and tools that let you take control of system power management.

Xilinx is an industry leader in power management and now offers many advantages within its programmable solutions:

• Virtex-4 FPGAs consume 1 to 5W less power than competing 90 nm FPGAs.

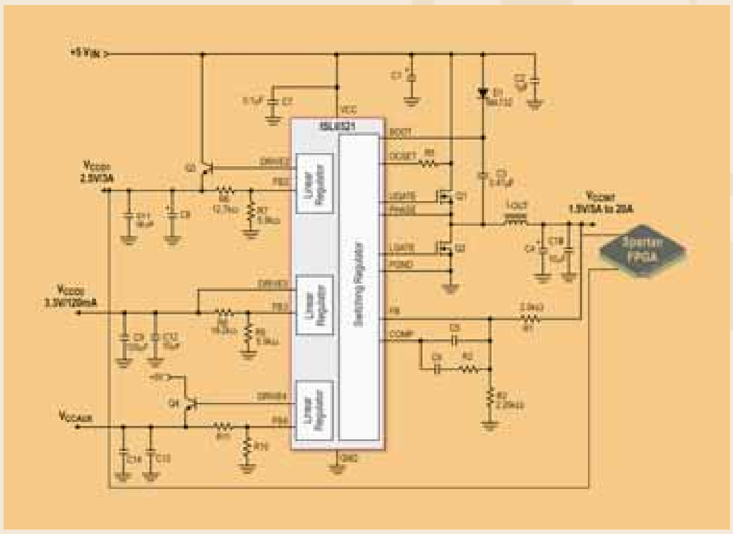
• Spartan-3 FPGAs are the one of the only low-cost FPGAs in the industry to eliminate power-on surge. In these devices, the maximum quiescent power alone is sufficient to guarantee device power-up.

• CoolRunner-II CPLDs are the world's lowest power CPLDs, ideal for even the most power-critical portable applications.

• Xilinx offers a comprehensive suite of power management tools, from Web Power Tools to the XPower analysis tool integrated into the ISE environment.

You can find comprehensive information on power consumption and solving key power challenges with Xilinx devices, tools, and solutions at *www.xilinx.com/power/*.

The Intersil ISL6521 Simplifies Power Solutions for Xilinx FPGAs



Features

- Provides one regulated voltage-switching regulator capable of 20A and three linear regulators capable of 120 mA or up to 3A with an external transistor
- Externally resistor-adjustable outputs
- Simple single-loop control design / voltage-mode PWM control
- Fast PWM converter transient response / high-bandwidth error amplifier / full 0% to 100% duty ratio
- Excellent output voltage regulation / all outputs: ±2% over temperature
- Overcurrent fault monitors / switching regulator does not require extra current sensing element, using instead the MOSFET's Rds(on)
- Small converter size / 300 kHz constant frequency operation / small external component count
- Commercial and industrial temperature range support
- Pb-free available (RoHS compliant)

Applications

- FPGA and PowerPC™-based boards
- General purpose, low-voltage power supplies

High integration with full features enables space and cost savings.

The Intersil ISL6521 quad regulator IC solution includes one switching regulator for loads as high as 20A and three linear regulators that each drive 120 mA or 3A with external transistors. High integration brings these four regulators together with full protection, power-on reset, and softstart features in one space- and cost-saving IC.

Applications for this new IC range broadly, providing power solutions for FPGAs, ASICs, POL, embedded systems, and I/O across medical, industrial, computing, and telecom.

The ISL6521 combines multiple switchers and/or linears in a single 16-lead SOIC package, delivering the integration and flexibility FPGA-based designers need. This single IC solution increases available board space while reducing costs and the number of required external components.

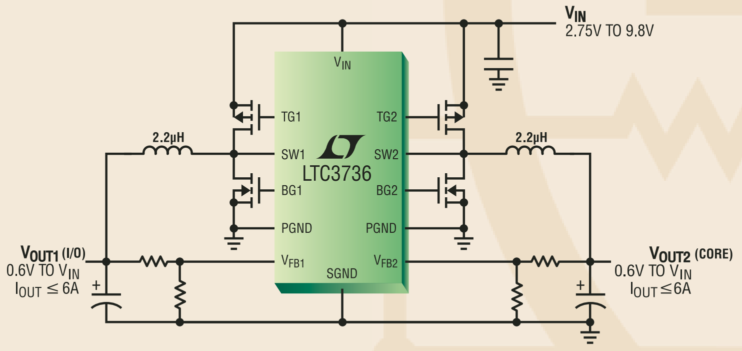
The ISL6521 PWM controller is intended to regulate the low-voltage supply that requires the greatest amount of current (usually the core voltage for the FPGA, ASIC, or processor) with a synchronous rectified buck converter. The linears are intended to regulate other system voltages, such as I/O and memory circuits. Both the switching regulator and linear voltage reference provide ±2% of static regulation over line, load, and temperature ranges. All outputs are user-adjustable by means of an external resistor divider. All linear controllers can supply up to 120 mA with no external pass devices.

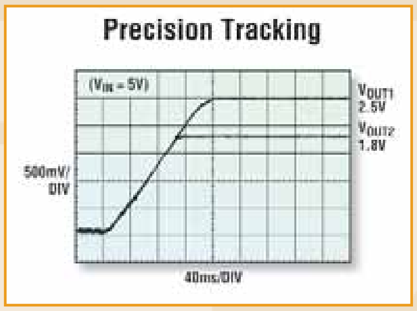
Employing bipolar NPNs for the pass transistors, the linear regulators can achieve output currents of 3A or higher with proper device selection. The ISL6521 monitors all output voltages. The PWM controller's adjustable overcurrent function monitors the output current by using the voltage drop across the upper MOSFET's rDS(ON). The linear regulator outputs are monitored through the FB pins for undervoltage events.

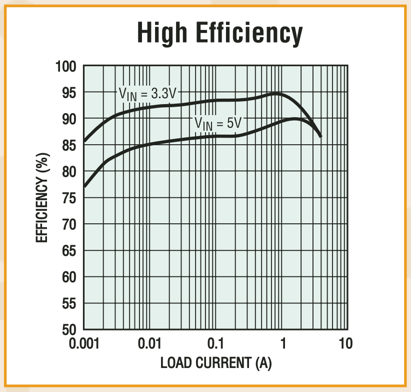
For a copy of "Power Management Application Guide for Xilinx® FPGAs," visit *www.intersil.com/Xilinx/index.asp.*

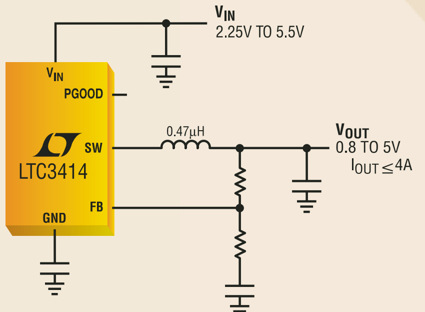
intersil
HIGH PERFORMANCE ANALOG

Spartan-3 FPGA Power Management Solutions









Linear Technology DC/DC converters power Spartan-3 devices.

Linear Technology offers a broad range of regulators to simplify the design and selection of DC/DC converters to power the Xilinx® Spartan™-3 family of FPGAs. Our converters range from low dropout linear regulators to sophisticated dual-output switching controllers that offer high efficiency and on-chip power supply tracking. Linear's free SwitcherCAD™ software allows quick and simple simulation of power supply designs. To download SwitcherCAD software, please visit *www.nuhorizons.com/linear*.

Single Output Regulators for Spartan-3 FPGAs — Core Voltage: 1.2V

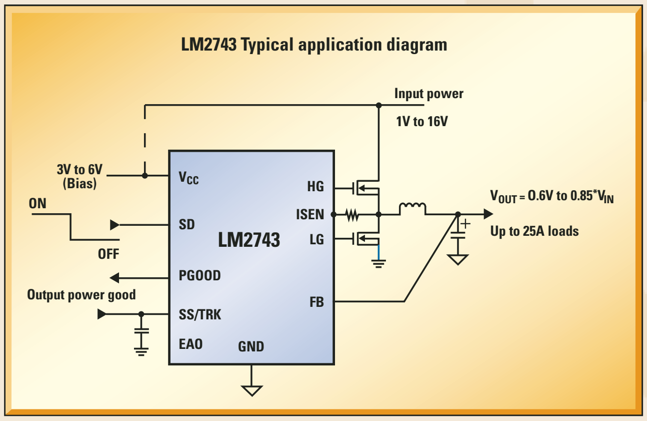
Input Supply	200 mA	500 mA	≤1A-1.5A	2A-2.5A	3A-5A	25A
1.8V	LTC3025 Linear	LT3021 Linear	LTC3026 Linear	LT3150 Linear	LTC3713 Controller	LTC3713 Controller
2.5V to 5V	LTC3405A Switcher	LTC3406 Switcher	LTC3412 Switcher	LTC3414 Switcher LTC3801 Controller	LTC3801 Controller LTC1773 Controller	LTC3713 Controller LTC3832 Controller LTC1778 Controller
12V	LT1616 Switcher	LT1976 Switcher LT1767 Switcher	LT1976 Switcher LT3431 Switcher LTC1778 Controller	LTC1771 Controller LTC1778 Controller	LTC1778 Controller	LTC1778 Controller
24V	LT1934 Switcher	LT1976 Switcher LT1767 Switcher	LT1976 Switcher LT3431 Switcher LTC1778 Controller	LT3431 Switcher LTC1778 Controller	LTC1778 Controller	LTC1778 Controller

NU HORIZONS ELECTRONICS CORP.
www.nuhorizons.com/linear

LINEAR TECHNOLOGY
www.linear.com

Versatile FPGA Power Solutions from National Semiconductor

LM2743/44 Low-Voltage Synchronous Buck Controllers



- Highly efficient 2A to 25A solution
- Input voltage from 1V to 16V
- Adjustable output voltage as low as 0.6V
- Power-good flag and output enable
- 1.5% reference accuracy over temperature
- Current limit without sense resistor
- Programmable softstart
- Switching frequency from 50 kHz to 1 MHz
- Available in small TSSOP-14 packaging

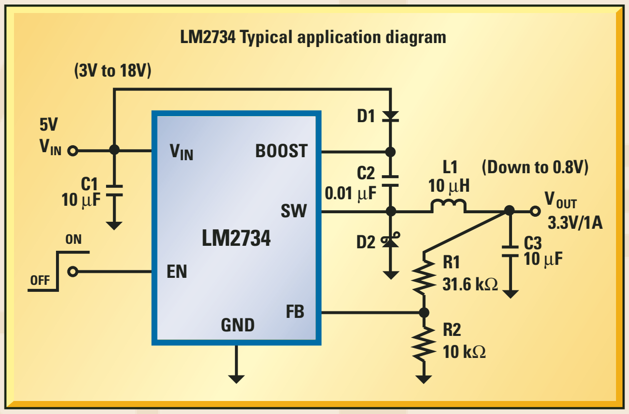
The LM2743 and LM2744 are high-speed synchronous buck regulator controllers that drive external MOSFETs to supply as much as 25A of current. They can provide simple down conversion to output voltages as low as 0.6V. Although the control sections of the ICs are rated for 3 to 6V, the driver sections are designed to accept input supply rails as high as 16V. The use of adaptive non-overlapping MOSFET gate drivers helps avoid potential shoot-through problems while maintaining high efficiency.

A wide range of switching frequencies from 50 kHz to 1 MHz gives Xilinx® FPGA and system power supply designers the flexibility to make better trade-offs between component size, cost, and efficiency. A versatile softstart and tracking pin allows ratiometric, coincidental, or offset tracking, which are critical for FPGA systems. An evaluation board is available.

For free samples, evaluation boards, or more information, visit *power.national.com* or call 1-800-272-9959.

National Semiconductor
The Sight & Sound of Information

LM2734/36 1A SOT-23 Buck Regulators



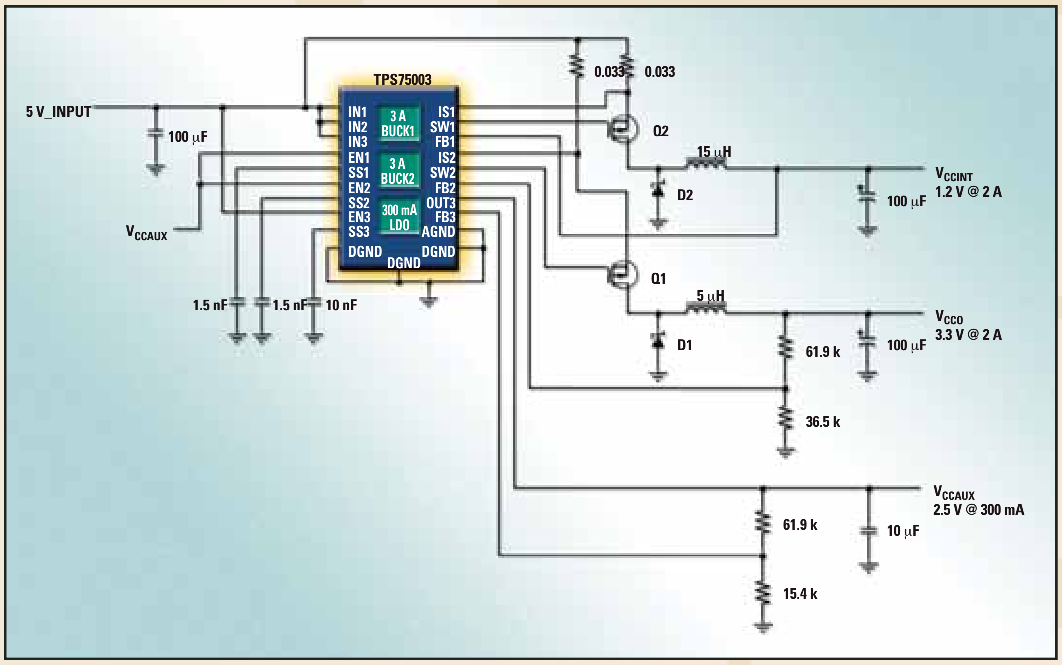
- Complete, easy-to-use switcher solution has the smallest footprint and highest power density in the industry
- Choice of switching frequencies allows designers to trade-off efficiency against solution size and EMI
- Current mode control improves phase margin, line regulation, and rejection of transients
- Internal softstart circuitry, cycle-by-cycle, thermal shutdown, and over-voltage protection

The LM2734 and LM2736 are monolithic, high-frequency (550 kHz and 1.6 MHz) PWM step-down DC/DC converters in tiny six-pin thin SOT-23 packaging. They provide local DC/DC conversion for Xilinx FPGAs with currents up to 1A.

Both regulators need no external compensation and are supported by WEBENCH®, National's online design tool. The ability to drive up to 1A loads with an internal 300 mΩ NMOS switch using state-of-the-art 0.5 µm BiCMOS technology results in the best power density available. The world-class control circuitry supports exceptionally high frequency conversion over the entire 3V to 20V input operating range, down to the minimum output voltage of 0.8V. Even though the operating frequencies are very high, efficiencies as high as 90% are easy to achieve. Additionally, a current-mode control loop provides fast transient response and accurate regulation in the smallest possible PCB area. An evaluation board is available.

Feature	LM2734	LM2736
Input Range	3.0V to 20V	3.0V to 18V
Output Load	1A	750 mA
Output Range	0.8V to 18V	1.25V to 16V
Internal References	0.8V, 2%	1.25V, 2%
Operating Frequency	550 kHz/1.6 MHz/3 MHz	

TI Power Solutions



Highly integrated triple supply powers Spartan-3 core, I/O, and V_{CCAUX} rails.

Features

- Two 95% efficient, 3A buck controllers and one 300 mA LDO
- Adjustable output voltages:
  - From 1.2V for bucks
  - From 1.0V for LDO
- Input voltage range of 2.2V to 6.5V
- Independent softstart for all three power supplies
- LDO stable with small ceramic output capacitor
- Independent enable for each supply for flexible sequencing
- 4.5 mm x 3.5 mm x 0.9 mm 20-pin QFN package
- 1 ku price: $1.90

Applications

- DSL modems
- Set-top boxes
- Plasma TV display panels
- DVD players

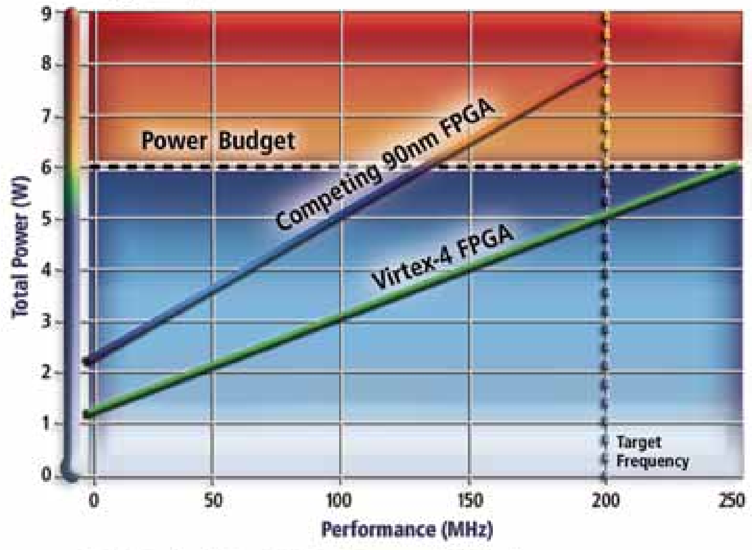
The TPS75003 power management IC for Xilinx® Spartan™-II, Spartan-IIE, and Spartan-3 FPGAs integrates multiple functions to significantly reduce the number of external components required – and simply your designs. Combining increased design flexibility with cost-effective voltage conversion, the device includes programmable softstart for in-rush current control and independent enables for sequencing the three channels. The TPS75003 meets all Xilinx startup profile requirements, including monotonic ramp and minimum ramp times.

For more information about the complete line of TI power management solutions for Xilinx FPGAs, including a library of reference designs, schematics, and BOMs, visit *www.ti.com/xilinxfpga*. For questions, samples, or an evaluation module, e-mail *fpgasupport@list.ti.com*.

TEXAS INSTRUMENTS

With Competing FPGAs, Power Has Become a Burning Issue.

Design Example:



Design Example: LX60 vs. 2S60. Target Frequency = 200 MHz. Worst-case process.
20K LUTs, 20K Flip-Flops, 1Mbit On-Chip RAM, 64 DSP Blocks, 128 2.5V I/Os
Based on Xilinx tool v4.0 and competitor tool v2.1
For higher density devices, achieve up to 5W lower power

VIRTEX™ V4

Get 1-5W lower power per FPGA, only with the Virtex-4 family

Check the specs for yourself at realistic operating temperatures ($T_j = 85^{\circ}C$). Different logic architecture or dielectric just won't do it. No competing FPGA comes close to Virtex-4 for total power savings:

- **73% lower static power**
- **Up to 86% lower dynamic power**
- **94% lower inrush current** (Take it to the lab and see!)

Unique Triple-Oxide Technology & Embedded IP

At the 90nm technology node, power is the next big challenge for system level designers. An inferior device can suffer leakage, dramatic surges in static power, and thermal runaway. That's why we designed our Virtex-4 FPGAs with Triple-Oxide Technology™, embedded IP, and power-saving configuration circuitry. Now you can meet your performance goals, while staying within the power budget.

Visit ***www.xilinx.com/virtex4/lowpower*** today, and get the right solution on board before your power issues start heating up.

XILINX®
The Programmable Logic Company℠

www.xilinx.com/virtex4/lowpower

View The TechOnLine Seminar Today

BREAKTHROUGH PERFORMANCE AT THE LOWEST COST

Spartan-3E Evaluation Kit from Avnet

Features

- Xilinx® XC3S100E TQ144 FPGA
- Cypress CY7C68013 USB 2.0 controller
- ST 4 MB SPI serial flash
- TI TPS75003 triple voltage regulator
- Windows USB download utility

Now shipping, Avnet design kits for Spartan FPGAs get you started fast.

Avnet is now shipping the newest member of its Spartan™ series design kits, the Spartan-3E Evaluation Kit, featuring the XC3S100E FPGA. This design kit, offered at $69, is an affordable and robust tool for becoming familiar with the new features of the Spartan-3E family. It features a Cypress USB controller that allows you to program the FPGA and/or the on-board flash device through the USB port on a PC.

In addition to the new Spartan-3E Evaluation Kit, Avnet offers a complete portfolio of other Spartan series design kits. These kits deliver a variety of low-cost, feature-rich platforms to develop and test designs targeted at Spartan-3 and Spartan-3L devices. The kits are available as either "evaluation" or "development" kits, depending on the resale price and mix of on-board peripherals. Please refer to the table for a complete list of part numbers and prices – all kits listed are available and shipping today.

In addition to the Spartan series design kits described here, Avnet offers a comprehensive portfolio of other design kits that support additional Xilinx FPGA families, such as Virtex™-4 and Virtex-II Pro/Virtex-II devices. It also offers a variety of kits targeted at specific technology domains, such as connectivity, embedded processing, and DSP.

Spartan-3E Design Kit		
Featured Device	Avnet Part Number	Price
XC3S100E	ADS-XLX-SP3E-EVL100	$69
Spartan-3 Design Kits		
Featured Devices	Avnet Part Number	Price
XC3S400	ADS-XLX-SP3-EVL400	$399
XC3S1500	ADS-XLX-SP3-EVL1500	$499
XC3S1500	ADS-XLX-SP3-DEV1500	$749
XC3S2000	ADS-XLX-SP3-DEV2000	$849
Spartan-3L Design Kit		
Featured Device	Avnet Part Number	Price
XC3S1000L	ADS-XLX-SP3-EVL1000L	$599

For a complete listing of the Xilinx design kits offered by Avnet, please visit *www.em.avnet.com/xlxboardlisting.*

TS101 TED Evaluation Board Using Spartan-3 FPGAs

Features

- Xilinx® XC3S400-4PQ208
- Xilinx XCF02SVO20C
- Seven-segment LEDs (4)
- LED
- Push switch
- DIP switch
- Four channels, 10-bit analog-to-digital converter
- Four channels, 10-bit digital-to-analog converter
- 32 Mb flash ROM (2M x 16 bits)
- 64 Mb SDRAM (4M x 16 bits)
- Interfaces:
  – Parallel port
  – Serial port
  – JTAG
  – Expansion connector (20 pin [12 signals])

The TS101 is an ideal platform for evaluating Xilinx Spartan-3 FPGAs.

The Circuit Design Learning Tool “TD-BD-TS101” is a toolkit for efficiently learning how to design circuits using HDL. The toolkit is based on a Xilinx Spartan™-3 (XC3S400-4PQ208) mounted on board as the main unit.

The main unit carries an RS-232C connector for communication with an external device. By using the toolkit in combination with your PC, you can easily check the board’s I/O operations to learn circuit designing.

This product comprises the main unit, AC adapter, documents, and CD-ROMs. On board, four channels of AD/DA converter, 32 bits of flash ROM, and 64 Mb of SDRAM extend the use of the evaluation board for several kids of circuit design.

You can program your designed circuit (test circuit) into on-board PROM through the parallel cable from your PC. After the data is written into PROM, re-supply the power or push the configuration button. Transfer test circuit data from the PROM to the FPGA (for configuring the FPGA) and execute the test circuit. Also, you can directly configure the bit file to the FPGA.

Part Number	TD-BD-TS101
Deliverables	TD-BD-TS101 Board
	Sample RTL Source Code
	User’s Manual
	Information for FPGA Mapping (UCF File)
	AC Adapter
	Configuration Cable

For more information, visit the
HiTech Global Distribution, LLC website at
www.hitechglobal.com/ted/ts101.htm.

Nu Horizons Spartan-3 2000 Development Platform

The ideal platform to evaluate Spartan-3 1500 and 2000 FPGAs.

The Nu Horizons Electronics Corp. Xilinx Spartan-3 1500/2000 development platforms are complete solutions for evaluating Xilinx Spartan-3 XC3S1500 and XC3S2000 FPGAs. Each board is a complete development environment, allowing you to evaluate multiple system designs on a single platform.

For industrial and automotive applications, the system board includes the ST Microelectronics L9616 CAN 2.0B physical layer device, which when coupled with the Bosch-certified CAN core (available from Nu Horizons) will get you well on your way to developing applications with the MicroBlaze soft-processor core. The external Ethernet controller allows you to create and evaluate CAN-to-Ethernet gateways.

With the addition of plug-in evaluation modules from Linear Technology, you can utilize analog-to-digital converters with 10/12/14-bit resolution and sampling rates from 10 to 135 Msps. Plug-in evaluation modules from Intersil allow you to utilize digital-to-analog converters with 8/10/12/14-bit resolution and throughput rates of 130 to 260 MHz – for use in applications such as medical imaging, 3G-4G base stations, and spectral analysis.

The Spartan-3 family from Xilinx has all of the features necessary for today's most demanding high-performance applications. Nu Horizons' Spartan-3 development platforms provide the exact hardware to develop and test to these requirements and meet your time-to-market expectations.

Features

- Based on the Xilinx® Spartan™-3 FPGA
  - Development boards available:
    - XC3S1500-4 FG676
    - XC3S2000-4 FG676
- ISSI 64 Mb SDRAM (IS42S16400)
  - Two 1M x 16 x 4 SDRAM – 16 MB
  - Footprint for 1M x 32 SSRAM provided
- ST 32 Mb flash (M29W320DB)
- ST high-speed CAN 2.0B PHY interface (L9616)
- 2 x 24-character LCD interface
- Sharp LCD panel connector I/F
- SMSC 10/100 Ethernet MAC and PHY (LAN91C111)
- ICS 10/100 Ethernet PHY (ICS1893BF)
- Linear Technology two-channel A/D and D/A converters
  - LTC1654 D/A 14-bit
  - LTC1865L A/D 16-bit
- ST audio CODEC (STW5093)
- PS2 port interface
- Intersil 2 – RS232 serial ports (ICL3237)
- 16-bit LVDS I/F with control signals
- Parallel Cable III and IV JTAG configuration support
- ICS PLL system clock generator (ICS511, ICS8422)

Additional plug-in evaluation modules available:

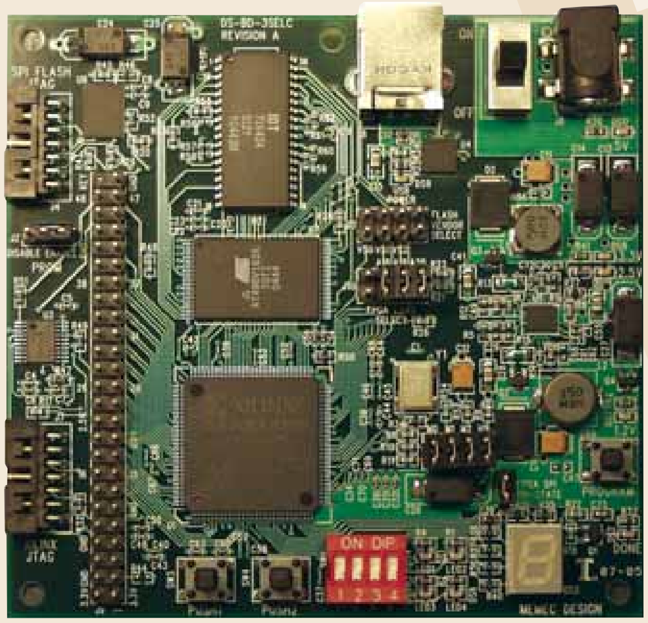
- Linear Technology high-speed A/D converters
- 10/12/14-bit 10 to 135 Msps ADCs
- Intersil high-speed D/A interface
- 8/10/12/14-bit 130 to 260 Msps DACs

Applications

- MicroBlaze™ soft-processor development
- DSP system development
- Industrial systems development
- Data communications/telecommunications
- Universal prototyping platform

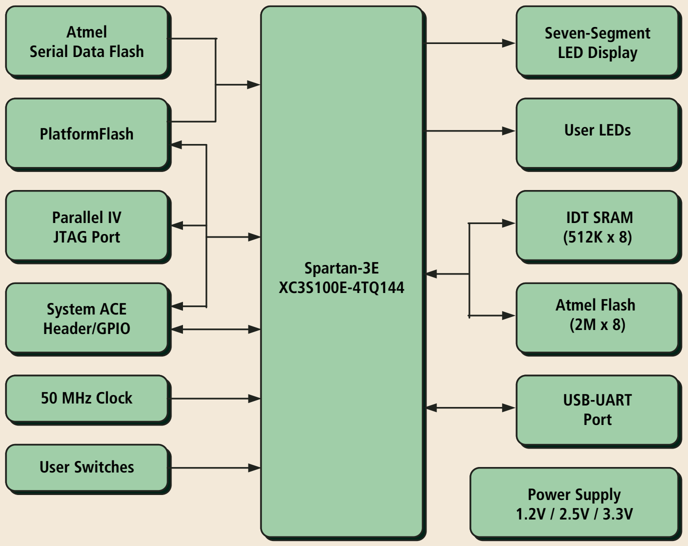
For all related documentation, visit *www.nuhorizons.com/sp3/*. And to evaluate the Spartan-3 2000 FPGA online today through our zero-cost TechOnLine VirtuaLab, visit *www.nuhorizons.com/VirtuaLab/*.

Memec Spartan-3E Board Solutions



Features

- XC3S100E-4TQ144 FPGA
- IDT 512K x 8 SRAM
- Atmel 2M x 8 flash memory
- Atmel 16 Mb serial data flash memory
- System Ace™ interface/user I/O header
- XCF01S Platform Flash ISP PROM
- Silicon Labs USB-UART interface
- Texas Instruments TPS75003 regulator
- Low cost



The Memec Spartan-3E LC Development Kit offers flexible configuration prototyping in a low-cost evaluation platform.

The Memec Spartan™-3E LC Development Kit for the Xilinx® Spartan-3E FPGA family provides a versatile prototype platform for evaluating the numerous configuration options available in Spartan-3E devices. In addition, the demo board provides the basic interface and support functions required by a low-cost platform for general-purpose prototype use.

The kit bundles a Spartan-3E-based demo board with a power supply, user guide, and reference designs. Xilinx ISE™ WebPACK™ software or BaseX software, along with a low-cost JTAG download cable, are available as kit options.

For more information or to order your Memec Spartan-3E LC Development Kit, visit *www.memec.com/xilinxkits*, or call (888) 488-4133 (in the U.S.) or (858) 314-8910 (outside the U.S.).

Xilinx FPGA Starter Kits

XILINX®

The Spartan-3 FPGA Starter Kit – $99

Complete solution includes:

- Xilinx® Spartan™-3 200,000-gate platform FPGA XC3S200-4FT256C
- Xilinx 2 Mb Platform Flash configuration PROM – XCF02S
- 1 MB of fast asynchronous SRAM (512K x 16 or 256K x 32)
- 3-bit, eight-color VGA display port
- Nine-pin RS-232 serial port
- PS/2-style mouse/keyboard port
- Four-character, seven-segment LED display
- Eight slide switches
- Eight individual LED outputs
- Four momentary contact push-button switches
- 50 MHz crystal clock oscillator
- Three 40-pin expansion connection ports
- Universal power supply 100-240V AC, 50/60 Hz
- JTAG cable
- Spartan-3 resource CD
- ISE™ Foundation™ evaluation CD
- EDK evaluation CD

Part numbers

- DO-SPAR3-DK
- DO-SPAR3-DK-J (Japanese version)

The Spartan-3 FPGA Starter Kit gives you instant access to the complete platform capabilities of the Xilinx Spartan-3 family. The kit brings high-volume designs to reality quicker, at a lower cost, and on schedule.

The Spartan-3E FPGA Starter Kit – $149 (available in Q3CY05)

Complete solution includes:

- Xilinx Spartan-3E 500,000-gate platform FPGA XC3S500E-4FG320
- 32 Mb parallel flash
- 8 Mb SPI flash
- 32 MB of DDR SDRAM
- Board interfaces
- Ethernet 10/100 PHY
- USB 2.0 PHY + controller
- 3-bit, eight-color VGA display port
- Nine-pin RS-232 serial port
- PS/2-style mouse/keyboard port
- Three 40-pin expansion connection ports

Additional features

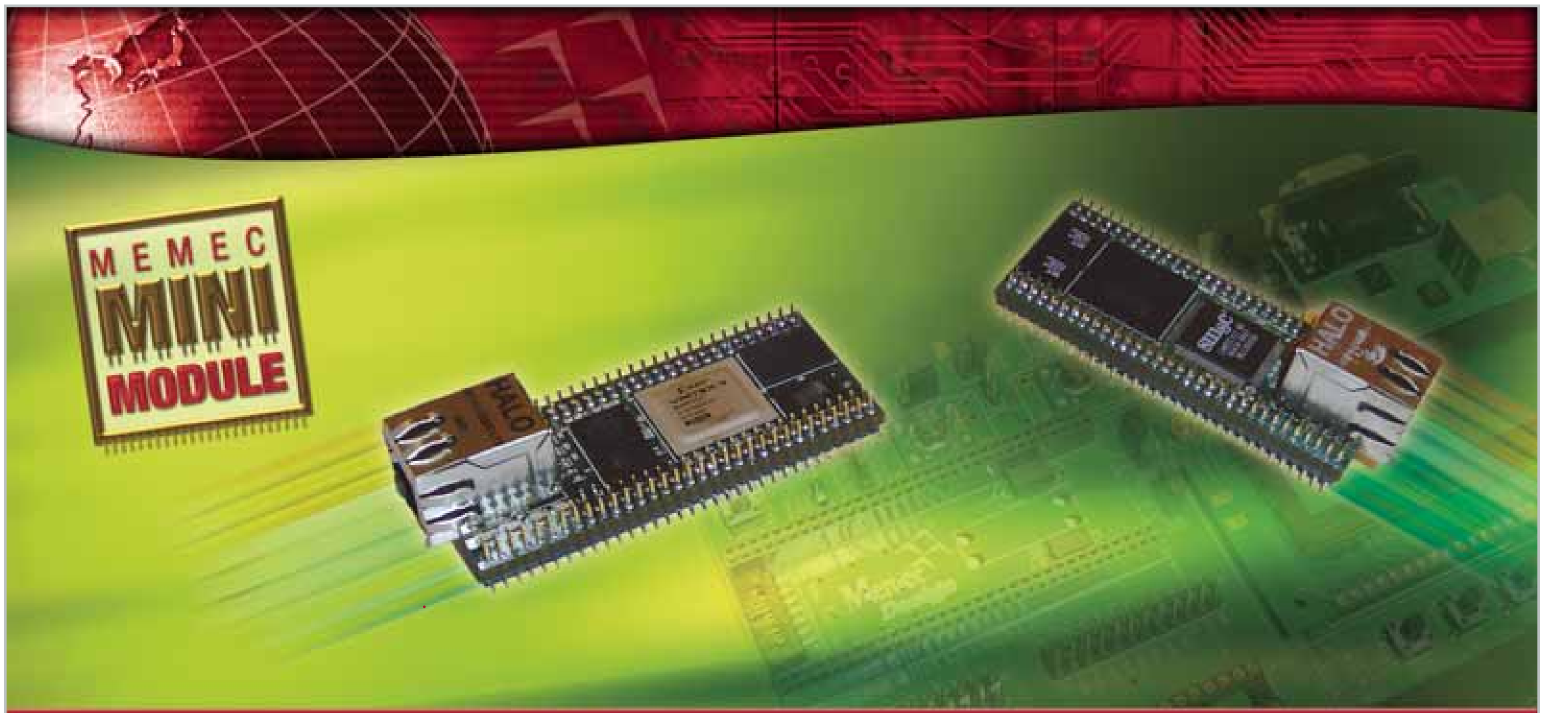
- Two-line LCD
- Four slide switches
- Eight individual LED outputs
- Two momentary contact push-button switches
- 90 MHz crystal clock oscillator
- Universal power supply 100-240V AC, 50/60 Hz
- JTAG cable
- Spartan-3/3E resource CD
- ISE Foundation evaluation CD
- EDK evaluation CD

Part numbers

- DO-SPAR3E-DK
- DO-SPAR3E-DK-J (Japanese version)

The Spartan-3E FPGA Starter Kit includes a full-featured development board based on the Spartan-3E platform FPGA family.

For more information about the Spartan-3 FPGA Starter Kit, visit *www.xilinx.com/s3boards/*. For more information about the Spartan-3E FPGA Starter Kit, visit *www.xilinx.com/s3eboards/*.



Memec Mini-Modules: The Ultimate Programmable System

What if... ***It became easier to integrate your favorite Xilinx FPGA onto your circuit board?***

What if... ***The job has already been done for you?***

Memec Mini-Modules are complete systems on a module, which pack all the necessary functions needed for an embedded processor system onto a tiny footprint the size of your thumb.

Spartan™-3 Mini-Module Features:

- Small Footprint (30 mm x 65.5 mm)
- Complete System-On-A-Module
- Supports MicroBlaze Processor
- 400K Gate Spartan-3 FPGA
- 10/100 Ethernet MAC and PHY
- SRAM and Flash Memory
- 76 User I/O

Virtex™-4 Mini Module Features:

- Small Footprint (30 mm x 65.5 mm)
- Complete System-On-A-Module
- Supports PowerPC Processor
- Based on the Virtex-4 FX12 FPGA
- 10/100/1000 Ethernet Port
- DDR and Flash Memory
- 76 User I/O

Visit www.memec.com/xilinx-minimodules

XILINX® The Programmable Logic Company℠

Xilinx Virtex-4™ FPGAs

www.xilinx.com/devices/

Product Selection Matrix

		Virtex-4 LX (Logic)								Virtex-4 SX (Signal Processing)			Virtex-4 FX (Embedded Processing & Serial Connectivity)					
		XC4VLX15	XC4VLX25	XC4VLX40	XC4VLX60	XC4VLX80	XC4VLX100	XC4VLX160	XC4VLX200	XC4VSX25	XC4VSX35	XC4VSX55	XC4VFX12	XC4VFX20	XC4VFX40	XC4VFX60	XC4VFX100	XC4VFX140
CLB Resources	CLB Array (Row x Column)	64 x 24	96 x 28	128 x 36	128 x 52	160 x 56	192 x 64	192 x 88	192 x 116	64 x 40	96 x 40	128 x 48	64 x 24	64 x 36	96 x 52	128 x 52	160 x 68	192 x 84
	Slices	6,144	10,752	18,432	26,624	35,840	49,152	67,584	89,088	10,240	15,360	24,576	5,472	8,544	18,624	25,280	42,176	63,168
	Logic Cells	13,824	24,192	41,472	59,904	80,640	110,592	152,064	200,448	23,040	34,560	55,296	12,312	19,224	41,904	56,880	94,896	142,128
	CLB Flip Flops	12,288	21,504	36,864	53,248	71,680	98,304	135,168	178,176	20,480	30,720	49,152	10,944	17,088	37,248	50,560	84,352	126,336
Memory Resources	Max. Distributed RAM Bits	98,304	172,032	294,912	425,984	573,440	786,432	1,081,344	1,425,408	163,840	245,760	393,216	87,552	136,704	297,984	404,480	674,816	1,010,688
	Block RAM/FIFO w/ECC (18 kbits each)	48	72	96	160	200	240	288	336	128	192	320	36	68	144	232	376	552
	Total Block RAM (kbits)	864	1,296	1,728	2,880	3,600	4,320	5,184	6,048	2,304	3,456	5,760	648	1,224	2,592	4,176	6,768	9,936
Clock Resources	Digital Clock Managers (DCM)	4	8	8	8	12	12	12	12	4	8	8	4	4	8	12	12	20
	Phase-matched Clock Dividers (PMCD)	0	4	4	4	8	8	8	8	0	4	4	0	0	4	8	8	8
I/O Resources	Max Select I/O™	320	448	640	640	768	960	960	960	320	448	640	320	320	448	576	768	896
	Total I/O Banks	9	11	13	13	15	17	17	17	9	11	13	9	9	11	13	15	17
	Digitally Controlled Impedence	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	Max Differential I/O Pairs	160	224	320	320	384	480	480	480	160	224	320	160	160	224	288	384	448
	I/O Standards	LDT-25, LVDS-25, LVDSEXT-25, BLVDS-25, ULVDS-25, LVPECL-25, LVCMOS25, LVCMOS18, LVCMOS15, PCI33, LVTTL, LVCMOS33, PCI-X, PCI66, GTL, GTL+, HSTL I (1.5V,1.8V), HSTL II (1.5V,1.8V), HSTL III (1.5V,1.8V), HSTL IV (1.5V,1.8V), SSTL2I, SSTL2II, SSTL18 I, SSTL18 II																
DSP Resources	XtremeDSP™ Slices	32	48	64	64	80	96	96	96	128	192	512	32	32	48	128	160	192
Embedded Hard IP Resources	PowerPC™ Processor Blocks	—	—	—	—	—	—	—	—	—	—	—	1	1	2	2	2	2
	10/100/1000 Ethernet MAC Blocks	—	—	—	—	—	—	—	—	—	—	—	2	2	4	4	4	4
	RocketIO™ Serial Transceivers	—	—	—	—	—	—	—	—	—	—	—	0	8	12	16	20	24
Speed Grades	Commercial (slowest to fastest)	-10, -11, -12	-10, -11, -12	-10, -11, -12	-10, -11, -12	-10, -11, -12	-10, -11, -12	-10, -11, -12	-10, -11	-10, -11, -12	-10, -11, -12	-10, -11, -12	-10, -11, -12	-10, -11, -12	-10, -11, -12	-10, -11, -12	-10, -11, -12	-10, -11
	Industrial (slowest to fastest)	-10, -11	-10, -11	-10, -11	-10, -11	-10, -11	-10, -11	-10, -11	-10	-10, -11	-10, -11	-10, -11	-10, -11	-10, -11	-10, -11	-10, -11	-10, -11	-10
	Configuration Memory Bits	4,765,568	7,819,904	12,259,712	17,717,632	23,291,008	30,711,680	40,347,008	51,367,808	9,147,648	13,700,288	22,745,216	4,765,568	7,242,624	13,550,720	21,002,880	33,065,408	47,856,896
	EasyPath™ Cost Reduction Solutions[4]	—	XCE4VLX25	XCE4VLX40	XCE4VLX60	XCE4VLX80	XCE4VLX100	XCE4VLX160	XCE4VLX200	XCE4VSX25	XCE4VSX35	XCE4VSX55	—	XCE4VFX20	XCE4VFX40	XCE4VFX60	XCE4VFX100	XCE4VFX140

Package[1]	Area	MGT[2]	Pins	4VLX15	4VLX25	4VLX40	4VLX60	4VLX80	4VLX100	4VLX160	4VLX200	4VSX25	4VSX35	4VSX55	4VFX12	4VFX20	4VFX40	4VFX60	4VFX100	4VFX140
SF363	17 x 17 mm	—	240	240	240										240					
FF668	27 x 27 mm	—	448	320	448	448	448					320	448		320					
FF1148	35 x 35 mm	—	768			640	640	768	768	768				640						
FF1513	40 x 40 mm	—	960						960	960	960									
FF672	27 x 27 mm	12	352													320 (8)[3]	352 (12)[3]	352 (12)[3]		
FF1152	35 x 35 mm	20	576														448 (12)[3]	576 (16)[3]	576 (20)[3]	
FF1517	40 x 40 mm	24	768																768 (20)[3]	768 (24)[3]
FF1760	42.5 x 42.5 mm	24	896																	896 (24)[3]

Notes: 1. SFA Packages (SF): flip-chip fine-pitch BGA (0.80 mm ball spacing).
FFA Packages (FF): flip-chip fine-pitch BGA (1.00 mm ball spacing).
All Virtex-4 LX and Virtex-4 SX devices available in the same package are footprint-compatible.
2. MGT: RocketIO Multi-Gigabit Transceivers.
3. Number of available RocketIO Muti-Gigabit Transceivers.
4. EasyPath solutions provide conversion-free path for volume production.

Pb-free solutions are available. For more information about Pb-free solutions, visit www.xilinx.com/pbfree.

Important: Verify all data in this document with the device data sheets found at http://www.xilinx.com/partinfo/databook.htm

Product Selection Matrix

		CLB Resources				Memory Resources			DSP	CLK Resources		I/O Features				Speed		PROM	
	System Gates (see note 1)	CLB Array (Row x Col)	Number of Slices	Equivalent Logic Cells	CLB Flip-Flops	Max. Distributed RAM Bits	# Block RAM	Block RAM (bits)	Dedicated Multipliers	DCM Frequency (min/max)	# DCMs	Digitally Controlled Impedance	Number of Differential I/O Pairs	Maximum I/O	I/O Standards	Commercial Speed Grades (slowest to fastest)	Industrial Speed Grades (slowest to fastest)	Configuration Memory (Bits)	EasyPath
Spartan-3E Family – 1.2 Volt																			
XC3S100E	100K	16 x 22	960	2,160	1920	15K	4	72K	4	5/326	2	NO	40	108	Single-ended LVTTL, LVCMOS3.3/2.5/1.8/ 1.5/1.2, PCI 3.3V – 32/64-bit 33/66MHz, PCI-X 100MHz, SSTL I 1.8/2.5, HSTL I 1.8, HSTL III 1.8 Differential LVDS2.5, Bus LVDS2.5, mini-LVDS, RSDS, LVPECL	-4 -5	-4	0.6M	
XC3S250E	250K	26 x 34	2448	5,508	4896	38K	12	216K	12	5/326	4	NO	68	172		-4 -5	-4	1.4M	
XC3S500E	500K	34 x 46	4656	10,476	9312	73K	20	360K	20	5/326	4	NO	92	232		-4 -5	-4	2.3M	
XC3S1200E	1200K	46 x 60	8672	19,512	17344	136K	28	504K	28	5/326	8	NO	124	304		-4 -5	-4	3.8M	
XC3S1600E	1600K	58 x 76	14752	33,192	29504	231K	36	648K	36	5/326	8	NO	156	376		-4 -5	-4	5.9M	
Spartan-3 and Spartan-3L Families – 1.2 Volt (see note 2)																			
XC3S50	50K	16 x 12	768	1,728	1,536	12K	4	72K	4	24/280	2	YES	56	124	Single-ended LVTTL, LVCMOS3.3/2.5/1.8/ 1.5/1.2, PCI 3.3V – 32/64-bit 33MHz, SSTL2 Class I & II, SSTL18 Class I, HSTL Class I, III, HSTL1.8 Class I, II & III, GTL, GTL+ Differential LVDS2.5, Bus LVDS2.5, Ultra LVDS2.5, LVDS_ext2.5, RSDS, LDT2.5, LVPECL	-4 -5	-4	.4M	
XC3S200	200K	24 x 20	1,920	4,320	3,840	30K	12	216K	12	24/280	4	YES	76	173		-4 -5	-4	1.0M	
XC3S400	400K	32 x 28	3,584	8,064	7,168	56K	16	288K	16	24/280	4	YES	116	264		-4 -5	-4	1.7M	
XC3S1000 XC3S1000L	1000K	48 x 40	7,680	17,280	15,360	120K	24	432K	24	24/280	4	YES	175	391		-4 -5	-4	3.2M	
XC3S1500 XC3S1500L	1500K	64 x 52	13,312	29,952	26,624	208K	32	576K	32	24/280	4	YES	221	487		-4 -5	-4	5.2M	✓
XC3S2000	2000K	80 x 64	20,480	46,080	40,960	320K	40	720K	40	24/280	4	YES	270	565		-4 -5	-4	7.7M	✓
XC3S4000 XC3S4000L	4000K	96 x 72	27,648	62,208	55,296	432K	96	1,728K	96	24/280	4	YES	312	712		-4 -5	-4	11.3M	✓
XC3S5000	5000K	104 x 80	33,280	74,880	66,560	520K	104	1,872K	104	24/280	4	YES	344	784		-4 -5	-4	13.3M	✓

Note: 1. System Gates include 20-30% of CLBs used as RAMs.
2. Spartan-3L devices offer reduced quiescent power consumption. Package offerings may vary slightly from those offered in the Spartan-3 family. See Package Selection Matrix for details.

Package Options and UserI/O[1]

		Spartan-3E (1.2V)						Spartan-3 (1.2V)								
			XC3S100E	XC3S250E	XC3S500E	XC3S1200E	XC3S1600E		XC3S50	XC3S200	XC3S400	XC3S1000	XC3S1500	XC3S2000	XC3S4000	XC3S5000
Pins	Area[2]	I/O's	108	172	232	304	376	I/O's	124	173	264	391	487	565	712	784
PQFP Packages (PQ) – wire-bond plastic QFP (0.5 mm lead spacing)																
208	30.6 x 30.6 mm			158	158				124	141	141					
VQFP Packages (VQ) – very thin QFP (0.5 mm lead spacing)																
100	16.0 x 16.0 mm		66	66					63	63						
TQFP Packages (TQ) – thin QFP (0.5 mm lead spacing)																
144	22.0 x 22.0 mm		108	108					97	97	97					
Chip Scale Packages (CP) – wire-bond chip-scale BGA (0.5 mm ball spacing)																
132	8 x 8 mm			92	92				89							
FGA Packages (FT) – wire-bond fine-pitch thin BGA (1.0 mm ball spacing)																
256	17 x 17 mm			172	190	190				173	173	173				
FGA Packages (FG) – wire-bond fine-pitch BGA (1.0 mm ball spacing)																
320	19 x 19 mm				232	250	250				221	221	221			
400	21 x 21 mm					304	304									
456	23 x 23 mm										264	333	333	333		
484	23 x 23 mm						376									
676	27 x 27 mm											391	487	489	489	
900	31 x 31 mm													565	633	633
1156	35 x 35 mm														712	784

Notes: 1. Numbers in table indicate maximum number of user I/Os.
2. Area dimensions for lead-frame products are inclusive of the leads.

Pb-free solutions are available. For more information about Pb-free solutions visit *www.xilinx.com/pbfree*.

Important: Verify all data in this document with the device data sheets found at http://www.xilinx.com/partinfo/databook.htm

For the latest information and product specifications on all Xilinx products, please visit the following links:

FPGA and CPLD Devices *www.xilinx.com/devices/*	Packaging *www.xilinx.com/packaging/*	Development Reference Boards *www.xilinx.com/board_search/*	Global Services *www.xilinx.com/support/gsd/*
Configuration and Storage Systems *www.xilinx.com/configsolns/*	Software *www.xilinx.com/ise/*	IP Reference *www.xilinx.com/ipcenter/*	

XILINX® The Programmable Logic Company℠

Xilinx Spartan™-3 FPGAs
www.xilinx.com/devices/

XILINX® The Programmable Logic Company℠

Xilinx CPLD

www.xilinx.com/devices/

Product Selection Matrix – CoolRunner™ Series

					I/O Features		Speed				Clocking	
	System Gates	Macrocells	Product Terms per Macrocell	Input Voltage Compatible	Output Voltage Compatible	Maximum I/O	I/O Banking	Min. Pin-to-pin Logic Delay (ns)	Commercial Speed Grades (fastest to slowest)	Industrial Speed Grades (fastest to slowest)	IQ Speed Grade	Global Clocks	Product Term Clocks per Function Block
CoolRunner-II Family – 1.8 Volt													
XC2C32A	750	32	40	1.5/1.8/2.5/3.3	1.5/1.8/2.5/3.3	33	2	3.8	-4 -6	-6	-6	3	17
XC2C64A	1,500	64	40	1.5/1.8/2.5/3.3	1.5/1.8/2.5/3.3	64	2	4.6	-5 -7	-7	-7	3	17
XC2C128	3,000	128	40	1.5/1.8/2.5/3.3	1.5/1.8/2.5/3.3	100	2	5.7	-6 -7	-7	-7	3	17
XC2C256	6,000	256	40	1.5/1.8/2.5/3.3	1.5/1.8/2.5/3.3	184	2	5.7	-6 -7	-7	-7	3	17
XC2C384	9,000	384	40	1.5/1.8/2.5/3.3	1.5/1.8/2.5/3.3	240	4	7.1	-7 -10	-10	-10	3	17
XC2C512	12,000	512	40	1.5/1.8/2.5/3.3	1.5/1.8/2.5/3.3	270	4	7.1	-7 -10	-10	-10	3	17
CoolRunner XPLA3 Family – 3.3 Volt													
XCR3032XL	750	32	48	3.3/5	3.3	36		5	-5 -7 -10	-7 -10	-10	4	16
XCR3064XL	1,500	64	48	3.3/5	3.3	68		6	-6 -7 -10	-7 -10	-10	4	16
XCR3128XL	3,000	128	48	3.3/5	3.3	108		6	-6 -7 -10	-7 -10	-10	4	16
XCR3256XL	6,000	256	48	3.3/5	3.3	164		7.5	-7 -10 -12	-10 -12	-12	4	16
XCR3384XL	9,000	384	48	3.3/5	3.3	220		7.5	-7 -10 -12	-10 -12	-12	4	16
XCR3512XL	12,000	512	48	3.3/5	3.3	260		7.5	-7 -10 -12	-10 -12	-12	4	16

Package Options and User I/O

		CoolRunner-II						CoolRunner XPLA3					
Pins	Area[1]	XC2C32A	XC2C64A	XC2C128	XC2C256	XC2C384	XC2C512	XCR3032XL	XCR3064XL	XCR3128XL	XCR3256XL	XCR3384XL	XCR3512XL
QFN Packages (QFG) – quad flat no-lead (0.5 mm lead spacing)													
32	5 x 5 mm	21											
48	7 x 7 mm		37										
PLCC Packages (PC) – wire-bond plastic chip carrier (1.27 mm lead spacing)													
44	17.5 x 17.5 mm	33	33					36	36				
PQFP Packages (PQ) – wire-bond plastic QFP (0.5 mm lead spacing)													
208	30.6 x 30.6 mm				173	173	173				164	172	180
VQFP Packages (VQ) – very thin QFP (0.5 mm lead spacing)													
44	12.0 x 12.0 mm	33	33					36	36				
100	16.0 x 16.0 mm		64	80	80				68	84			
TQFP Packages (TQ) – thin QFP (0.5 mm lead spacing)													
144	22.0 x 22.0 mm			100	118	118				108	120	118*	
Chip Scale Packages (CP) – wire-bond chip-scale BGA (0.5 mm ball spacing)													
56	6 x 6 mm	33	45						48				
132	8 x 8 mm			100	106								
Chip Scale Packages (CS) – wire-bond chip-scale BGA (0.8 mm ball spacing)													
48	7 x 7 mm							36	40				
144	12 x 12 mm									108			
280	16 x 16 mm										164		
FGA Packages (FT) – wire-bond fine-pitch thin BGA (1.0 mm ball spacing)													
256	17 x 17 mm				184	212	212				164	212	212
FBGA Packages (FG) – wire-bond fine-line BGA (1.0 mm ball spacing)													
324	23 x 23 mm					240	270					220	260

* JTAG pins and port enable are not pin compatible in this package for this member of the family.

Note 1: Area dimensions for lead-frame products are inclusive of the leads.

Product Selection Matrix – 9500 Series

						I/O Features		Speed				Clocking	
	System Gates	Macrocells	Product Terms per Macrocell	Input Voltage Compatible	Output Voltage Compatible	Maximum I/O	I/O Banking	Min. Pin-to-pin Logic Delay (ns)	Commercial Speed Grades (fastest to slowest)	Industrial Speed Grades (fastest to slowest)	IQ Speed Grade	Global Clocks	Product Term Clocks per Function Block
XC9500XV Family – 2.5 Volt													
XC9536XV	800	36	90	2.5/3.3	1.8/2.5/3.3	36	1	5	-5 -7	-7	NA	3	18
XC9572XV	1,600	72	90	2.5/3.3	1.8/2.5/3.3	72	1	5	-5 -7	-7	NA	3	18
XC95144XV	3,200	144	90	2.5/3.3	1.8/2.5/3.3	117	2	5	-5 -7	-7	NA	3	18
XC95288XV	6,400	288	90	2.5/3.3	1.8/2.5/3.3	192	4	6	-6 -7 -10	-7 -10	NA	3	18
XC9500XL Family – 3.3 Volt													
XC9536XL	800	36	90	2.5/3.3/5	2.5/3.3	36		5	-5 -7 -10	-7 -10	-10	3	18
XC9572XL	1,600	72	90	2.5/3.3/5	2.5/3.3	72		5	-5 -7 -10	-7 -10	-10	3	18
XC95144XL	3,200	144	90	2.5/3.3/5	2.5/3.3	117		5	-5 -7 -10	-7 -10	NA	3	18
XC95288XL	6,400	288	90	2.5/3.3/5	2.5/3.3	192		6	-6 -7 -10	-7 -10	NA	3	18
XC9500 Family – 5 Volt													
XC9536	800	36	90	5	5	36		10	-5 -6 -10 -15	-7 -10 -15	-15	3	18
XC9572	1,600	72	90	5	5	72		10	-7 -10 -15	-10 -15	-15	3	18
XC95108	2,400	108	90	5	5	108		10	-7 -10 -15 -20	-7 -10 -15 -20	NA	3	18
XC95144	3,200	144	90	5	5	133		10	-7 -10 -15	-10 -15	NA	3	18
XC95216	4,800	216	90	5	5	166		10	-10 -15 -20	-10 -15 -20	NA	3	18
XC95288	6,400	288	90	5	5	192		10	-10 -15 -20	-15 -20	NA	3	18

Pb-free solutions are available. For more information about Pb-free solutions visit *www.xilinx.com/pbfree*

Package Options and User I/O

		XC9500XV				XC9500XL				XC9500					
Pins	Area[1]	XC9536XV	XC9572XV	XC95144XV	XC95288XV	XC9536XL	XC9572XL	XC95144XL	XC95288XL	XC9536	XC9572	XC95108	XC95144	XC95216	XC95288
PLCC Packages (PC) – wire-bond plastic chip carrier (1.27 mm lead spacing)															
44	17.5 x 17.5 mm	34	34			34	34			34	34				
84	30.2 x 30.2 mm										69	69			
PQFP Packages (PQ) – wire-bond plastic QFP (0.5 mm lead spacing)															
100	23.3 x 17.2 mm										72	81	81		
160	31.2 x 31.2 mm											108	133	133	
208	30.6 x 30.6 mm				168				168					166	168
VQFP Packages (VQ) – very thin TQFP (0.5 mm lead spacing)															
44	12.0 x 12.0 mm	34	34			34	34			34					
64	12.0 x 12.0 mm					36	52								
TQFP Packages (TQ) – thin QFP (0.5 mm lead spacing)															
100	16.0 x 16.0 mm		72	81			72	81			72	81	81		
144	22.0 x 22.0 mm			117	117			117	117						
Chip Scale Packages (CS) – wire-bond chip-scale BGA (0.8 mm ball spacing)															
48	7 x 7 mm	36	38			36	38			34					
144	12 x 12 mm			117				117							
280	16 x 16 mm				192				192						
BGA Packages (BG) – wire-bond standard BGA (1.27 mm ball spacing)															
256	27 x 27 mm								192						
352	35.0 x 35.0 mm													166	192
FBGA Packages (FG) – wire-bond Fine-line BGA (1.0 mm ball spacing)															
256	17 x 17 mm				192				192						

Note 1: Area dimensions for lead-frame products are inclusive of the leads.

XILINX®
The Programmable Logic Company℠

Xilinx CPLD

www.xilinx.com/devices/

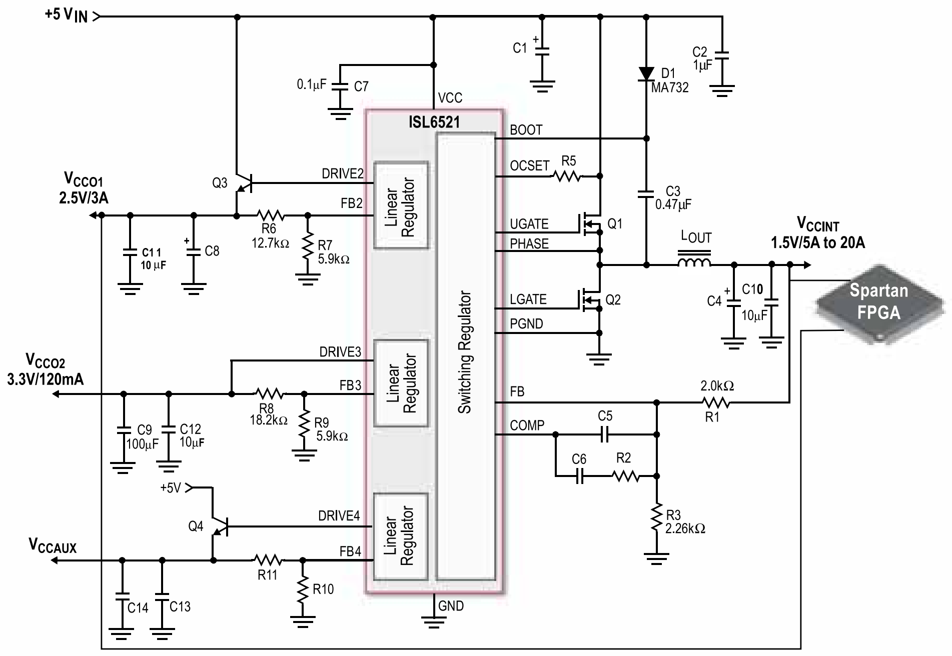
Intersil Switching Regulators

Intersil High Performance Analog

Power Solutions for FPGA Design Has Never Been So Easy

Intersil offers a wide range of Power Supply solutions for the latest generations of Xilinx FPGA-based systems.

Reduce your Spartan-3 board size and increase efficiency with Intersil's multiple output DC/DC Switching Regulators or Switching Regulators with Integrated MOSFETs.



Intersil's Xilinx Spartan-3 Power Solutions

	Peak I_{CCINT} Current (A) Requirement*		Recommended Intersil Power Solutions	
Device	@ 325MHz	@150 MHz	V_{IN}=+5V	V_{IN}=+3.3V
XC3S50	0.6	0.3	ISL6410, ISL6410A, or EL7536	
XC3S200	1.5	0.7	ISL6528, EL75XX Family or ISL6521	EL75XX Family or ISL6526
XC3S400	2.3	1.1		
XC3S1000	4.2	2.0		
XC3S1500	6.6	3.2		
XC3S2000	9.6	4.6	ISL6521 or ISL6539	ISL6526
XC3S4000	15.3	7.2		
XC3S5000	17.6	8.4		

Selecting the Right Switching Regulator for Spartan-3 FPGA-Based System

How many power supplies do you need?

Three or Four...

For up to 20A - use Intersil ISL6521

Two...

For smallest size - use Intersil ISL6528

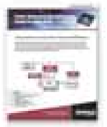
For highest efficiency - use Intersil ISL6539

One...

For ease of use and Integration up to 0.5A - use Intersil ISL6410 and ISL6410A Switching Regulators with Integrated FETs

For ease of use and Integration from 1A to 8A - use Intersil EL75XX Switching Regulators with Integrated FETs

For most flexibility up 20A - use Intersil ISL6526



For detailed application note on using Intersil Power Management Solutions in Xilinx® FPGAs go to **www.intersil.com/xfpgas**

Document includes Intersil's recommeded power supply solutions, IC schematics, functional block diagrams, BOMs, selection tables, Xilinx® FPGA power tables, DDR memory power solutions, sequencers and supervisory circuits, and layout considerations.

Download ISim:PE, Intersil's personal edition offline design simulation tool at **www.intersil.com/iSim**

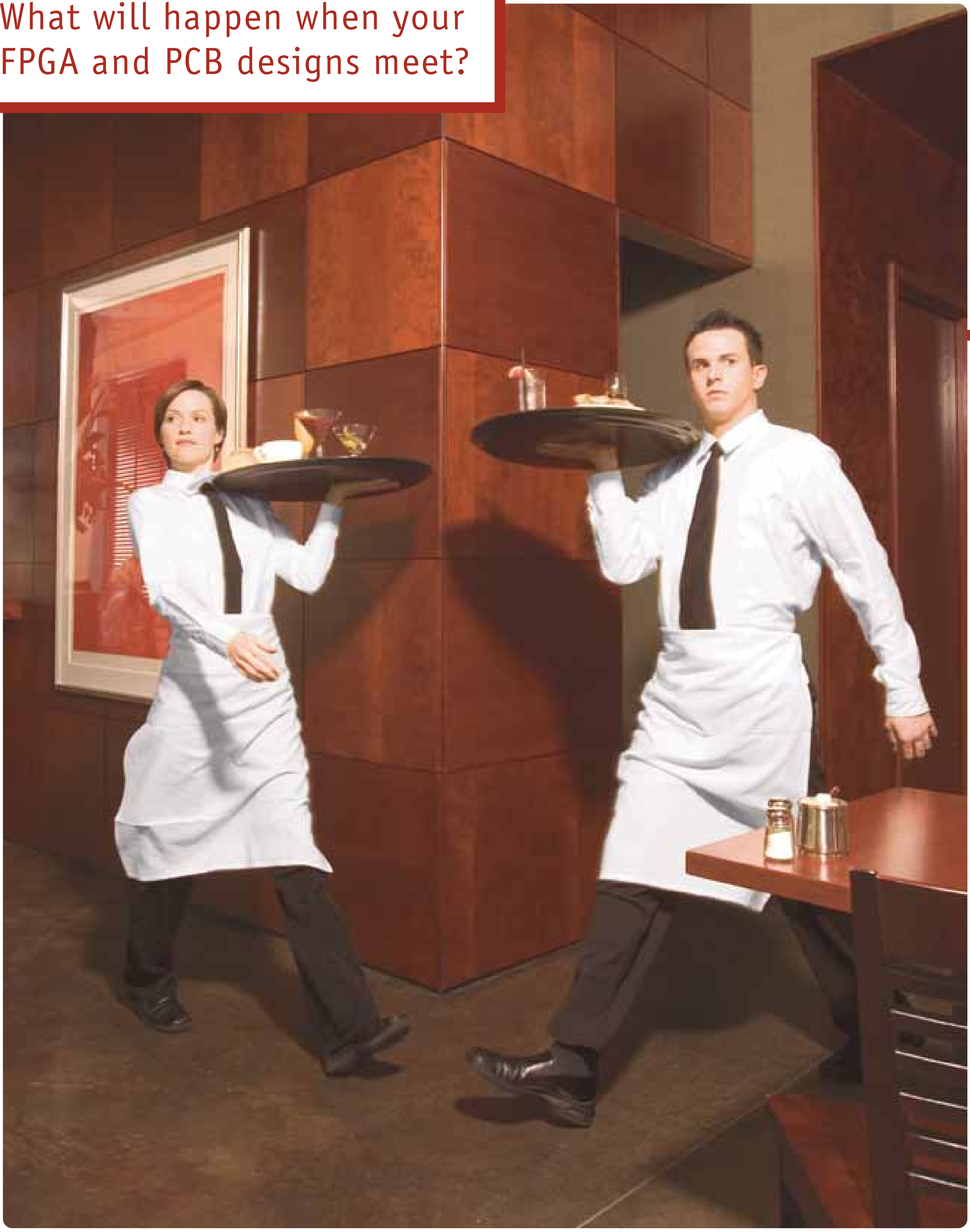
Find the power management device that matches your design requirements at **www.intersil.com/pmps**

Intersil – Switching Regulators for precise power delivery.

intersil® HIGH PERFORMANCE ANALOG

What will happen when your FPGA and PCB designs meet?

Integrated Systems

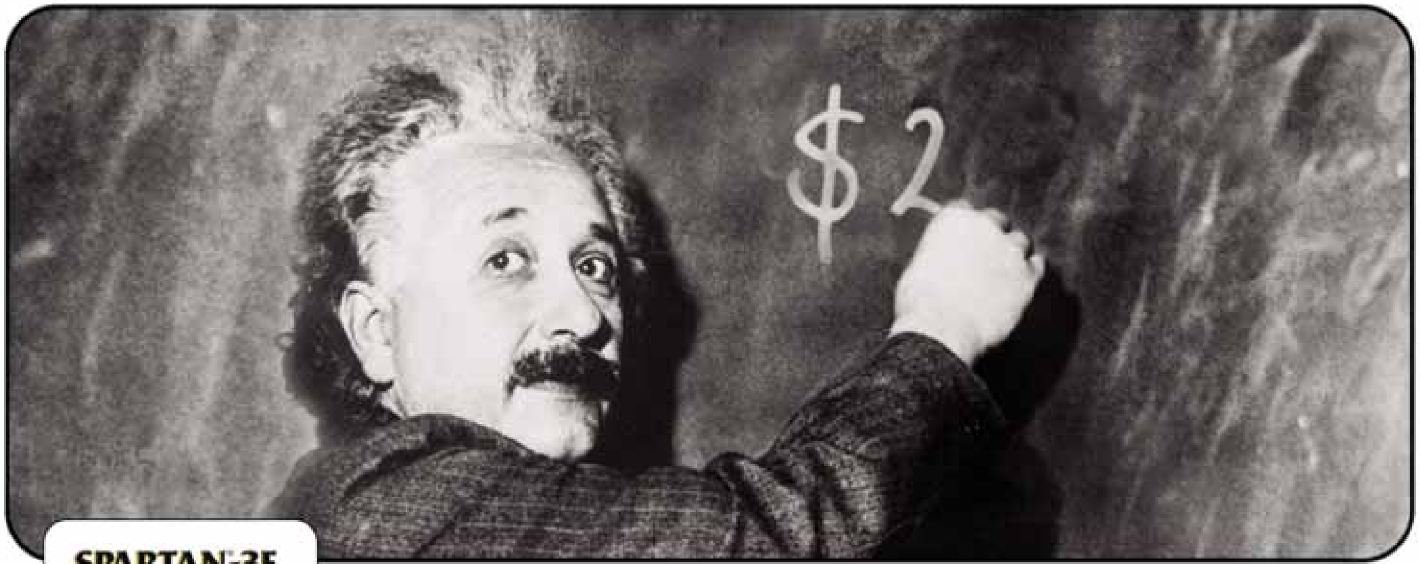


Integrated Systems | The potential for disaster is enormous if your designers work independently of one another. Mentor Graphics eliminates that potential with the only integrated systems design solution in EDA. We have superior design solutions in both PCB and FPGA, so it stands to reason that we'd create the only truly integrated system flow. Our unique class of tools empowers designers to concurrently design FPGA's and PCB's. Increase your system performance and design team productivity. Get the systems integration white paper at **www.mentor.com/techpapers** or call **800.547.3000.**

Mentor Graphics®

Introducing the new Spartan-3E family — the world's lowest-cost FPGAs

$$E = \frac{\$2}{100\text{K gates}}$$



SPARTAN-3E FPGA

Amazing product…Incredible price…An industry first!

Wow! A 90nm platform FPGA loaded with features for just $2.00!* The new Spartan-3E devices, optimized for gate-centric designs, are perfect for consumer digital apps and so much more. Reducing the previous unit cost benchmark by over 30%, Spartan-3E FPGAs make it easier than ever to replace your ASIC with a more flexible, faster-to-market solution.

Easiest to use Software

Get the most for your money

With Spartan-3E FPGAs, two dollars buys a lot — 100K gates (the lowest cost per logic cell in the industry), PLUS embedded multipliers for high-performance/low-cost DSP, tons of RAM, digital clock managers, and all the I/O support you need. With a density range up to 1.6 million gates, and our production-proven 90nm process, now is the time to make your money work smarter!

MAKE IT YOUR ASIC

XILINX®

The Programmable Logic Company℠

For more information visit
www.xilinx.com/spartan3e

Earth Friendly — Pb-free devices available now

FORTUNE 2005 100 BEST COMPANIES TO WORK FOR

* Pricing for 500K units, second half of 2006

PN 0010864