XCELL

Issue 21
Second Quarter 1996

THE QUARTERLY JOURNAL FOR XILINX PROGRAMMABLE LOGIC USERS

XILINX®
The Programmable Logic Company℠

Inside This Issue:

GENERAL

Fawcett: PLDs, Pins, PCBs (part 2) 2
Guest Editorial 3
Customer Success Story 6-7
1996 Data Book 7
University Workshops 8
Training Update 9
Fiscal Year Financial Results 9
New Product Literature 10
Upcoming Events 10
Development Systems Chart 11
Component Availability Chart 12-13
Alliance Program Charts 14-16
Programming Support Charts 17-19

PRODUCTS

XC9500 Product Life Cycle 20-21

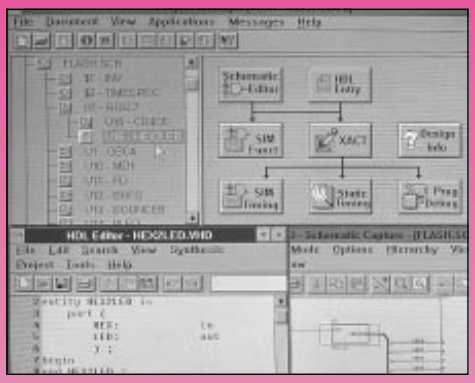
DEVELOPMENT SYSTEMS

Foundation Series 22-23
XACT-CPLD Introduced 24
XACT*step* 6.0.1 Software Update 25

HINTS & ISSUES

Using OrCAD Capture and Simulate 26-28
Foundation on a Network 29
Viewlogic's Workview Office 30-31
HDL Synthesis and Clock Enables 32-34
Ten-Digit Synchronous BCD Counter ... 35
Structured Floorplanning 36-39
Minimum Delays 40-41
Visit FPGA Newsgroup 41
Advanced Carry-Logic Techniques ... 42-44
PCI-Based Reconfigurable Computers .. 45
Questions & Answers 46-47
Technical Support Resources 47
Fax Response Form 48

PRODUCT INFORMATION



VHDL Made Easy! Introducing Foundation Series Software

The new Foundation Series packages are complete, fully integrated sets of development tools for CPLD and FPGA device design that include the HDL Wizard, a set of tools that help users quickly learn and implement HDL-based designs...

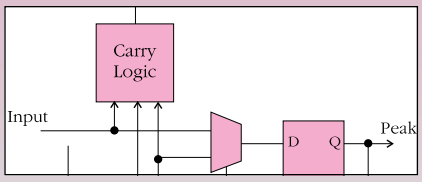
See Page 22

XACT*step*™ 6.0.1 Release

This update to the XACT*step* development system adds XC4000E FPGA and XC9500 CPLD support...

See Page 25

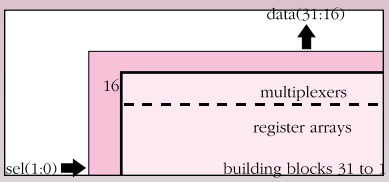
DESIGN TIPS & HINTS



Advanced Carry Logic Techniques

The XC4000 FPGA's carry logic can be used for lots more than adders and counters...

See Page 42



Structured Floorplanning

Implement high-ordered designs efficiently and easily with the XC8100 family...

See Page 36

Part 2 of 2

PLDs, Pins and PCBs *(part 2)*

By BRADLY FAWCETT ◆ Editor

Part 1 of this article (*XCELL* 20, page 2) discussed the inevitability of design changes during all stages of an electronic system's lifecycle. Changes can occur as a result of the debugging and testing of the initial design, due to specification changes during the design, or even to add features to a mature product to extend that product's life.

An important benefit of user-programmable logic is tolerance of change; with PLDs, design changes can be implemented quickly and easily. However, printed circuit board (PCB) designs are not as easy to change, typically requiring new drawings (masks) and the manufacturing of new prototypes, with all the associated expenses and delays. Thus, to garner all the benefits of the flexibility of programmable logic, programmable logic device architectures should isolate the PCB design from logic changes that occur within the PLD device. Device architectures should do this in two ways — by supporting pin-locking and with footprint compatibility.

Pin-locking — that is, for signals entering and leaving a PLD, maintaining the pin locations during design changes internal to the PLD — was discussed at length in part 1. Support for pin-locking is a key feature of the latest generations of Xilinx CPLDs and FPGAs.

Equally important, **footprint compatibility** maximizes PLD design flexibility, and has been incorporated in all Xilinx components since the XC2000 family — the world's first FPGA! Footprint compatibility refers to the availability of PLDs of various gate densities with the same package and with an identical pinout. With a range of footprint-compatible devices available, users may migrate a given PLD design to a higher- or lower-density device without changing the printed circuit board.

Footprints in the Silicon

There are several scenarios where a common device footprint provides a significant advantage. The most prevalent of these is when a design is being modified to add features without changing the pinout requirements, and, as a result, the design grows to exceed the gate density of the PLD device that was initially selected. By moving the design to a footprint-compatible device with higher capacity, a re-layout of the printed circuit board is avoided, saving both time and money.

> *"footprint compatibility maximizes PLD design flexibility, and has been incorporated in all Xilinx components since the XC2000 family"*

On the other hand, a design can be initially prototyped in a larger device than needed, to allow room for expansion and experimentation. Once the design is fixed, it can be migrated to a smaller, less-expensive device in the same package as a cost reduction. Again, footprint compatibility between the devices avoids changes to the printed circuit board. *(There is, however,*

Continued on page 5

XCELL

Please direct all inquiries, comments and submissions to:

Editor: Bradly Fawcett

Xilinx, Inc.
2100 Logic Drive
San Jose, CA 95124
Phone: 408-879-5097
FAX: 408-879-4676
E-Mail: brad.fawcett@xilinx.com

XILINX®

The First Ten Weeks

By WILLEM ROELANDTS ◆ Chief Executive Officer

It is a pleasure to address you after being in my new position for ten weeks. I feel even better today about my decision to join Xilinx than I felt when I started, now that I have a better understanding of the company and its products. During the past ten weeks I have spent most of my time meeting with the Xilinx people, our foundry partners, our sales and distribution organizations, our shareholders and our customers. In general, I am very pleased with the feedback and inputs that I have received.

I have found that Xilinx people are very capable, technically competent (after all, Xilinx did invent FPGA technology), and very motivated.

The relationship between Xilinx and our foundry IC partners is both excellent and enduring. Due to strong teamwork, it has withstood the inevitable ups and downs of the IC business.

The quality of our independent sales organization is very strong. We have had long-standing relationships with them and they are very familiar with our products. Some have invested in their own Field Application Engineers (FAEs), in order to serve our customers better.

Our customers are pleased with our products. They like the time-to-market advantages of using FPGAs and CPLDs in design and production, and our standard parts simplify their production and inventory management. Our users have also pointed out some issues and opportunities, including on-time delivery, the importance of software tools and the continuing need for bigger and faster chips.

During this time I also have reflected on our strategic directions. I can assure you that the major components of this highly successful strategy will not change. We will continue to:

- **Focus on FPGA technology** by aggressively increasing size and performance. The XC4000EX family will reach 125,000 gates in 1997. Our XC5000 family is an excellent solution for the low-end FPGA market. The XC8100 family of one-time-programmable FPGAs provides other unique features.
- **Add technologies** for specific customer requirements. The very successful XC9500 family uses a CPLD architecture, but adds in-system programmability (ISP), a feature highly appreciated by our users.
- **Develop reconfigurable logic**, which is the ability to dynamically change the logic configuration of the FPGA during the operation of the device. This technology promises to change the way logic is designed. It is an area in which Xilinx has done a lot of work and I intend to aggressively continue the effort.
- **Use outside foundries for our wafer processing**. It provides the best flexibility and enables us to provide our customers a stable flow of high quality products.

Continued on the next page

"I can assure you that the major components of this highly successful strategy will not change."

XILINX®

Continued from the previous page

- **Work with independent sales organizations**, but provide them with technical resources for training and support.
- **Use the six sigma defect rate standard** as our norm for product quality. In order to guarantee the quality of our products, we will continue to do most product testing in-house.
- **Provide technical support that is second-to-none** in the industry. The ability to contact experts quickly is critical to maintaining the productivity of our users.

"There is no doubt in my mind that software has to be a core competency of Xilinx; it is as important as our ability to design FPGA chips."

There are some changes or refocusing of strategic components that I feel will make Xilinx even more successful. They include:

The importance of software

There is no doubt in my mind that software has to be a core competency of Xilinx; it is as important as our ability to design FPGA chips. That's why we acquired NeoCAD. I believe we have the right people and now we are going to execute. Our objective is to be the best in the industry, and I intend to give this my personal attention. You have already seen some of the results with the introduction of the XACT*step*™ 6.0/5.2 release, the Japanese version of this product, and the new Foundation™ release — a fully-integrated, "shrink-wrap" package for the PC platform. Foundation software packages offer the best functionality in their price class (starting at less that $500), and are complete software sets delivered from Xilinx! This is only the beginning; more new capabilities and performance improvements will be introduced this year.

The opportunity of logistics

The IC industry goes through cycles of feast and famine, and last year we grew more than expected, causing us to sometimes miss our delivery schedules (although our record was better than most). I believe that with better planning of the supply chain, better management of the distribution process, shorter manufacturing cycles and more aggressive use of information technology, we should be able to do a better job and meet 100% of our delivery commitments.

Design paradigm change

With the densities of our largest FPGAs exceeding 100,000 gates, our users will no longer be able to design logic functions one gate at a time. In response, we are moving to a new paradigm. Xilinx is creating libraries of specific functions in software that will be tested and guaranteed — we call these LogiCore™modules. We envision a design process where the designer selects the needed functions, adds the application's specific logic, and lets the software put it all together. This methodology will dramatically reduce the time required to design complex logic functions, improving time-to-market and the efficiency of our users. Of course, the complete implementation of this vision will take some time. The first LogiCore product, the PCI module, has proven this concept and has been tremendously successful. We are going to aggressively pursue this strategy.

So there you have it — an overview of my thinking after 10 weeks on the job. I will continue to keep you informed of the progress we are making on the execution of our strategy. ◆

Continued from page 2

one caveat to consider when migrating a design from a larger to a smaller PLD device. For some smaller devices, the package may have more physical pins than there are input/output pads on the device. Thus, some package pins may be left unconnected. A larger device in the same family may have more I/O pads on the die and, therefore, have connections to all the pins of the given package. Thus, if migration to a smaller part is anticipated, the initial design in the larger device should avoid using those pin locations that are not connected in the smaller device.)

In other words, **footprint compatibility lessens any risks associated with the initial device selection**, which often must be based on a rough estimate of the design's requirements. If the selected device turns out to be too small, the design is migrated to a larger device. If the selected device is too big, the design can be moved to a smaller device. In either case, potentially expensive and time-consuming changes to the PCB are not necessary.

Footprint-compatible devices also provide the user with more inventory flexibility. Devices that are on-hand can be used for prototyping or initial production, and the design can then be migrated to a footprint-compatible device for quantity production. If a sudden demand "upside" should develop, users have the option to move to a larger device in the same family or a similar-sized device from another footprint-compatible family.

Compatible From the Start

Recognizing these benefits, Xilinx always has maintained footprint compatibility within component product families and subfamilies whenever multiple devices share common packages. For example, the XC3030 and XC3042 devices share a common footprint in the PC84, TQ100, and VQ100 packages. That same footprint is maintained in the equivalent density members of the XC3000A, XC3100, XC3100A, and XC3000L sub-families. *(The only exceptions are the XC3000 series and its derivatives in the PC 84 package, where some of the larger devices need two additional GND and V_{CC} connections, and in the PQ208 package, where the XC3090 and XC3195 devices do not have compatible footprints.)*

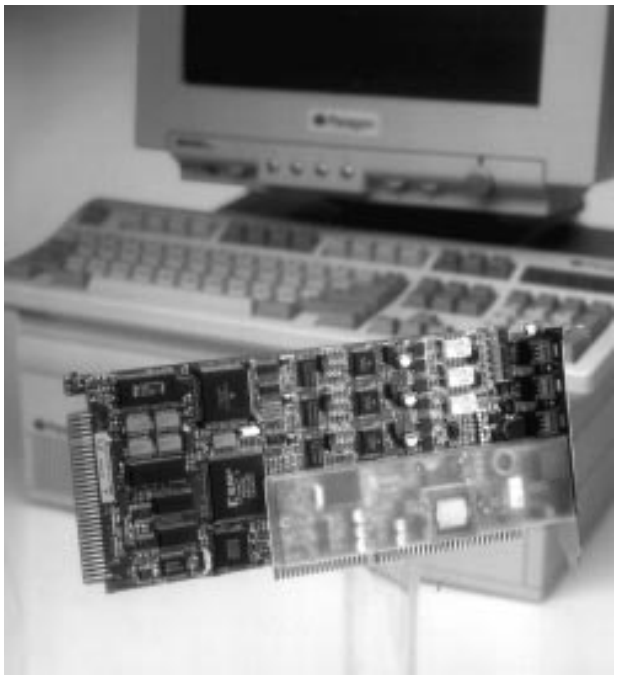
Cross-family compatibility began with the XC4000 series of FPGAs and includes the XC5000 series and the XC8100 series — all sharing common footprints in common packages. This provides designers with many options. For example, as reported in *XCELL* 19, VTEL Corp., a manufacturer of video teleconferencing systems and one of the first adopters of the XC5000 family, prototyped their designs in XC4000 series FPGAs while awaiting the availability of XC5000 components and development tools. The resulting designs were easily migrated to lower-cost, footprint-compatible XC5000 devices for production systems. In a similar scenario, a designer could exploit the re-usable nature of the SRAM-based XC4000 or XC5000 FPGAs for debug and prototyping purposes, and then switch to the one-time-programmable XC8100 family for production.

Designers should avoid getting locked into programmable logic solutions that offer little flexibility in pin assignments and device selection. Xilinx CPLDs and FPGAs offer the best pin-locking capabilities in the industry, and the broadest spectrum of footprint-compatible devices. These features allow users to avoid modifications to printed circuit board designs, thereby accelerating time-to-market and accommodating the inevitable design changes that occur throughout a product's total life cycle. ◆

"Designers should avoid getting locked into programmable logic solutions that offer little flexibility in pin assignments and device selection."

FPGAs Go "Down Under" in an

Engineers at communications equipment specialist Tennyson Technologies (Notting Hill, Victoria, Australia) are experienced users of Xilinx XC3000 and XC4000 series FPGAs. Thus, when a new project created a need for high integration levels, design flexibility, and a fast time-to-market, all at a reasonable cost, it was no surprise that they turned to the latest Xilinx FPGA technology — the XC5000 series. In Tennyson's new MicroAccess PCTA terminal adapter card, both bus interface and communication control functions are integrated into a single XC5206-6 FPGA device.



The MicroAccess system includes a plug-in card for PC systems and the accompanying software. It allows any PC or PC LAN to automatically make connections to off-site systems; the connection can be made to last only as long as information is being exchanged, much like a telephone call. With support for voice and data transfers, the MicroAccess system permits connectivity through ISDN, regular telephone line or X.25 services.

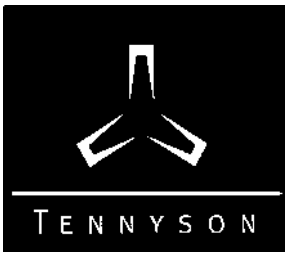
The logic functions implemented in the FPGA device include the ISA-bus interface, FIFO control, communications control, V110 rate adaptation, data compression/decompression, and other glue logic. The bus interface supports plug-and-play capability and accounts for about one-half of the logic in the FPGA. The bulk of the communications control logic consists of the counters used to assemble and synchronize the frames of data. About 75% of the available CLBs are used in this design, as well as most of the I/O pins available on the PQ208 package.

While any of several FPGA families could have provided the required density and functionality, Tennyson's engineers were attracted to the XC5000 architecture's VersaRing™ feature, in which extra routing channels around the perimeter of the array increase the flexibility of I/O connections. In order to meet the time-to-market goals, the designers realized that the printed circuit board (PCB) would need to be designed in parallel with the system's logic. Thus, the pinout for the XC5206 FPGA was fixed prior to the design of its internal logic. Through each design iteration, the VersaRing concept held true; changes to the FPGA design did not force any changes to the original PCB layout.

The flexibility provided by the SRAM-based FPGA was key to the successful design of the system. For example, the board was originally intended to support a 16-bit ISA bus interface only, but the specification was later changed to require support for both 8-bit XT and 16-bit AT systems. Since the entire bus interface is implemented in the FPGA, this requirement was accommodated, without requiring changes to the PCB layout as a result.

Taking advantage of the in-system-programmable FPGA technology, the MicroAccess board has been designed in anticipation of future field upgrades. New

ISDN Terminal Adapter



software (stored in Flash memory on the board) and new FPGA configuration programs can be downloaded to units in the field via an ISDN interface. In fact, one set of FPGA configuration programs has been dubbed the "Emergency Xilinx" diagnostic mode; this configuration runs just the ISDN channel, allowing for the downloading of new software and configuration bitstreams to the system.

Tennyson Technology's development environment is PC-based, and includes Viewlogic System's schematic editor and simulator, as well as the Xilinx XACT® development system. XACT-Performance™ time specifications and some floorplanning of the FPGA placement were used to meet the required 33 MHz system clock rate, a fairly aggressive goal when using the -6 speed grade. Both the Viewsim simulator and the X-Delay timing calculator were used to verify the functionality and timing of the FPGA design.

With the aid of Xilinx FPGAs, Tennyson Technology's MicroAccess system is allowing some of Australia's leading organizations to combine telephone and data traffic into single services, providing for better communications with branch offices and other remote locations while reducing overall communication costs. ◆

1996 Data Book Available Soon

The *1996 Xilinx Programmable Logic Data Book* will be available this summer. It will contain data sheets and product specifications describing Xilinx component and development system products, including the latest information on the new XC4000E, XC4000EX , XC6200, XC8100 and XC9500 device families. Other included device families described are the XC7200A and XC7300 CPLD families, the XC3000A, XC3000L, XC3100A, XC3100L, XC4000L, XC4000XL and XC5200 FPGA families, and the XC1700 family of Serial PROMs.

The product specifications for the XC2000, XC3000, XC3100, XC4000, XC4000A, XC4000D, and XC4000H FPGA families are not included. These products are still available; however, we recommend using the products in the data book for new designs because they offer better performance at lower cost than the older technologies. Product specifications for the older products are available at WebLINX, the Xilinx site on the World Wide Web (www.xilinx.com), or through your local Xilinx sales representative.

The military/high-reliability and HardWire™ product lines are overviewed in the 1996 edition, but will retain their own, separate detailed product specification literature.

While the book provides detailed, easy-to-access information about Xilinx products, as a book, it can only offer a "snapshot" of Xilinx products in early 1996. Inquire with your sales representative, WebLINX or this journal for the latest information on new devices, speed grades, package types and development systems. ◆

Xilinx Hosts University Workshops

If you are a university professor who would like to incorporate programmable logic technology into your engineering curriculum, get started by attending a Xilinx University Workshop. Xilinx technology has been used in many engineering courses, including beginning and advanced digital design, processor architecture, digital signal processing, VLSI design, data communications and various project-oriented laboratory courses.

This summer, Xilinx will be hosting four workshops in the United States and two in Asia. These workshops provide a thorough introduction to programmable logic technology, and discuss how to integrate the technology into first-year through fifth-year university courses. The workshops are taught by a team of Xilinx training professionals and local professors with experience at using programmable logic in their own coursework. Technologies covered at the workshops include field programmable gate arrays (FPGAs), complex programmable logic devices (CPLDs), and dynamically reconfigurable logic. Although most workshops will cover the same basic material, some will have special themes, as described below. Hands-on labs using PC-based development tools are included in each workshop.

These three-day workshops are available free of charge to professors and instructors. However, due to the popularity of the workshops, attendance is limited to two people from each university or college. Seating and hotel rooms at each workshop are limited, so early registration is advised.

To register, contact Jason Feinsmith, Xilinx University Program Manager, at 408-879-4961 or e-mail xup@xilinx.com or visit www.xilinx.com/programs/univ.htm for information. ◆

WORKSHOP SCHEDULE

UNITED STATES

Xilinx/Washington State University Workshop
Richland, WA ◆ June 24 - 26
In addition to the basic course material, Dr. Donald Hung will discuss his first-hand experiences in developing courses that use programmable logic.

Xilinx/Cornell University Reconfigurable Computing Workshop
Ithaca, NY ◆ July 10 -12
This will be our first hands-on workshop devoted to the topic of reconfigurable computing with FPGAs. Targeted at educators who are familiar with reprogrammable logic and are very interested in the concept of dynamically reconfigurable computing, this workshop will condense the basic material and focus on this new and promising area of study. Several researchers will be present to discuss their work.

Xilinx/Massachusetts Institute of Technology Workshop
Boston, MA ◆ July 15 - 17
For those particularly interested in computer architectures, this workshop will include a look at MIT's newly-created computer structures curriculum based on Xilinx devices and the "electric legos" concept.

Xilinx/Oakland University Workshop
Detroit, MI ◆ July 24 - 26
Dr. Subra Ganesan will discuss his work in using FPGAs in digital signal processing applications at this workshop.

ASIAN SCHEDULE

Workshops are planned for the week of August 19 in China, and the week of August 26 in Taiwan. Further details were not available at the time of this printing.

For up-to-date information, visit www.xilinx.com/programs/univ.htm on the World Wide Web. ◆

Fiscal 1996: *Another Record Year*

As in every year since Xilinx was founded, the company again achieved record revenues in fiscal year 1996 (April 1995-March 1996). Fiscal 1996 revenues totaled $560.8 million, an increase of 58% over fiscal 1995, reflecting the continuing strength of our product line and expansion of the programmable logic market.

Key accomplishments of fiscal 1996 include the following:

- The flagship XC4000 series products contributed revenues of over $250 million, more than doubling the revenue of the prior year.
- International revenues grew by more than 80% to $198 million.
- Xilinx Ireland, our first wholly-owned manufacturing site outside of the U.S., became fully operational.
- The merger with NeoCAD Corp. was completed, providing access to powerful new software solutions.
- Access to leading-edge process technology was enhanced with a 25% equity investment in an 8-inch wafer fabrication facility in a joint venture with United Microelectronics Corp.
- Our commitment to product development was exemplified by research and development spending of approximately $65 million, 63% more than the second-largest programmable logic supplier.

As noted by Willem Roelandts, Chief Executive Officer, "Xilinx is well-positioned as we enter fiscal 1997. New products that we introduced in the second half of fiscal 1996, including the high-performance XC4000E, the high-density XC4000EX, and the Flash-based in-system-programmable XC9500 CPLD family, should become key revenue drivers next year. Moreover, all these silicon products will be supported by a new, more-powerful, and easier-to-use software solution. Looking ahead, we remain optimistic about the overall growth of programmable logic and our position within this market."

Paced by growth in the European market, revenues for the fourth fiscal quarter (ending in March) reached a record $149.7 million, up 37% from the same quarter one year ago, and 4% from the immediately previous quarter.

Founded in 1984, Xilinx is the world's largest supplier of programmable logic devices. Xilinx stock is traded on the NASDAQ exchange under stock symbol XLNX. ◆

Advanced Training Available

The advanced training course is again available at Xilinx headquarters in San Jose, California. The session focuses on features of the Xilinx devices and development system that allow experienced users to create more efficient designs.

Advanced training will be useful for anyone "pushing the envelope" in terms of the speed and density of Xilinx FPGA products, especially the XC4000 family devices. The primary focus is on floorplanning and the use of the new graphical Floorplanner™.

Classes are scheduled regularly at Xilinx headquarters starting in May. The class is open to all in-warranty Xilinx customers at no charge. Previous experience with the Xilinx products is a prerequisite for the advanced training course.

For more information or to register, visit WebLINX (www.xilinx.com) on the World Wide Web, call Xilinx Training at 408-879-5090, or e-mail to customer.training@xilinx.com. ◆

New Product Literature

Learn about the newest Xilinx products and services through our extensive library of product literature. The most recent pieces are listed below. To order or to obtain a complete list of all available literature, please contact your local Xilinx sales representative. ◆

TITLE	DESCRIPTION	NUMBER
FPGAs		
XC4000 to XC4000E Conversion Guide	Technical Data	#0010295-01
Interfacing between 5-V and 3.3-V	Technical Data	#0010296-01
Efficient Shift Registers	Technical Data	#0010298-01
CPLDs		
XC9500 Quick Reference Guide	Features & Benefits	#0010408-01
Software		
XACT*step* 5.2 Sell Sheet	Features & Benefits	#0010289-01
Xilinx Foundation Series Sell Sheet	Features & Benefits	#0010292-01
Other		
1996 Training Brochure		#0010134-05
Literature Packet: DSP	Technical Data	Packet #2
Literature Packet: RADD (Reconfigurable Architectures)	Technical Data	Packet #15

UPCOMING EVENTS

Look for Xilinx technical papers and/or product exhibits at these upcoming industry forums. For further information about any of these conferences, please contact Kathleen Pizzo (Tel: 408-879-5377 FAX: 408-879-4676). ◆

Design Automation Conference (DAC)
June 3-7
Las Vegas, Nevada

Intertronic
June 4-7
Paris, France

DSP Roadshow
June 5-6
Manchester, UK

DSP Scandinavia
June 18-19
Copenhagen, Denmark

International Conference on Application Specific Systems, Architectures, and Processors
Aug. 19-21
Chicago, Illinois

Electronic Design Automation and Test Conference (EDA&T)
Sept. 5 -6 , Beijing, China
Sept. 9-10, Seoul, Korea
Sept. 12-13, Hsinchu, Taiwan

DSP Roadshow
Sept. 25-26
London, UK

International Workshop on Field Programmable Logic and Applications
Sept. 23-25
Darmstadt, Germany

XILINX RELEASED SOFTWARE STATUS - MAY 1996

	Key	Product Category	Product Description	Product Function	Xilinx Part Reference Number	Current Version by Platform PC1 *6.2*	SN2 *4.1.x*	HP7 *9.01*	Last Updt Comp	Previous Version Release	Comments Features
XILINX INDIVIDUAL PRODUCTS		CORE EPLD	XC7K, XC9500 Support	Core Implementation	DS-550-xxx	6.00	5.20	5.20	11/95	5.10	PC update by request only
		XABEL-CPLD	XC7K, XC9500 Support	Entry/Simulation/Core	DS-571-PC1	6.10			na	6.01	No key required with version 6.0.1
	N	XACT-CPLD	XC7K, XC9500 Support	Core + I/F	DS-560-xxx	6.0	6.0	6.0	na	na	First ship 4/23/96
		Mentor	8.4=A.4	I/F and Libraries	DS-344-xxx		5.20	5.20	11/95	5.11	No AP1 update
		OrCAD		I/F and Libraries	DS-35-PC1	5.20			11/95	5.10	Support for SDT+, VST+ v1.2
		Synopsys		I/F and Libraries	DS-401-xxx		5.20	5.20	11/95	3.30	Includes XC5200 X-BLOX Support
		Viewlogic	PROcapture	I/F and Libraries	DS-390-PC1	6.00			11/95	5.11	Includes PRO Series 6.1
		Viewlogic	PROsim	I/F and Libraries	DS-290-PC1	6.00			11/95	5.11	Includes PRO Series 6.1
		Viewlogic		I/F and Libraries	DS-391-xxx	6.00	5.20	5.20	11/95	5.11	
		XABEL		Entry, Simulation, Lib, Optimizer	DS-371-xxx	5.20	5.20	5.20	11/95	5.10	Now available on HP7
		XBLOX		Module Generator & Optimizer	DS-380-xxx	5.20	5.20	5.20	11/95	5.10	
	E	Verilog	2K, 3K, 4K, 7K Libraries	Models & XNF Translator	ES-VERILOG-xxx		1.00	1.00	na	na	Sun and HP
XILINX PACKAGES			SILICON SUPPORT 2K	3K	4K	5K	7K	8K	9K							
		Cadence	X	X	X	X	X			DS-CDN-STD-xxx		5.20	5.20	11/95	5.11	
		XC8000						X		DS-8000-STD-xxx	1.10	1.10	1.10	na	na	Incl sch and sim libraries
		XC8000						X		DS-8000-EXT-xxx	1.10	1.10	1.10	08/95	1.00	Incl sch, sim, and synth libraries
		Mentor	X	X	X	X	X			DS-MN8-STD-xxx		5.20	5.20	11/95	5.11	No AP1 update
		Mentor	X	X	X	X	X			DS-MN8-ADV-xxx		7.00	7.00	na	na	
	U	OrCAD	X	X	X	X	X			DS-OR-BAS-PC1	6.00			11/95	5.11	Update by request only
		OrCAD	X	X	X	X	X			DS-OR-STD-PC1	6.00			11/95	5.11	
		Synopsys		X/E	X	X	X			DS-SY-STD-xxx		5.20	5.20	11/95	5.12	Includes DS-401 v5.2
		Synopsys		X/E	X	X	X			DS-SY-ADV-xxx		7.00	7.00	na	na	Includes DS-401 v5.2
	U	Viewlogic	X	X	X	X	X			DS-VL-BAS-PC1	6.00			11/95	5.11	Update by request only
		Viewlogic	X	X	X	X	X			DS-VL-STD-xxx	6.00	6.00	6.00	11/95	5.11	
		Viewlogic	X	X	X	X	X			DS-VL-ADV-xxx	7.00	7.00	7.00	na	na	
		Viewlogic/S	X	X	X	X	X			DS-VLS-BAS-PC1	6.00			11/95	5.11	Includes PROSeries 6.1
		Viewlogic/S	X	X	X	X	X			DS-VLS-STD-PC1	6.00			11/95	5.11	Includes PROSeries 6.1
		Viewlogic/S	X	X	X	X	X			DS-VLS-EXT-PC1	6.00			11/95	5.11	Incl PROSeries 6.1/PROsynth 5.02X
		Viewllogic/S	X	X	X	X	X			DS-VLS-ADV-PC1	7.00			na	na	
	U	3rd Party Alliance	X	X	X	X	X			DS-3PA-BAS-xxx	6.00			na	na	
		3rd Party Alliance	X	X	X	X	X			DS-3PA-STD-xxx	6.00	5.20	5.20	na	5.10	Includes 502/550/380
		3rd Party Alliance	X	X	X	X	X			DS-3PA-ADV-xxx	7.00	7.00	7.00	na	na	Includes 502/550/380 & Foundry
		Foundation Series	X	X	X	X	X	SCH		DS-FND-BAS-PC1	6.00			na	na	New Aldec software v2.2: 1st ship 3/15/96
		Foundation Series	X	X	X	X	X	SCH		DS-FND-BSV-PC1	6.00			na	na	New Aldec software v2.2: 1st ship 3/15/96
		Foundation Series	X	X	X	X	X	SCH		DS-FND-STD-PC1	6.00			na	na	New Aldec software v2.2: 1st ship 3/15/96
		Foundation Series	X	X	X	X	X	SCH		DS-FND-STV-PC1	6.00			na	na	New Aldec software v2.2: 1st ship 3/15/96
		LogiCore - PCI			4KE					LC-DI-PCIM-C	1.00	1.00	1.00	na	na	Req signed lic agreement; 1st ship 3/15/96
	N	Evaluation	X	X	X	X	X	X	X	DS-EVAL-XXX-C	1.00	1.00	1.00	na	na	PC, Sun, HP kits now available
PRE-RELEASE				SCH.	SIM.	SYN.	LIB.									
	PR	XC4000E		X	X					PR-MN8-STD-xxx-4E		1.00	1.00	na	na	By request only; Req In-wnty prod
	PR	XC4000E		X	X					PR-OR-STD-xxx-4E	1.00			na	na	By request only; Req In-wnty prod
	PR	XC4000E			VHDL	VHDL Verilog				PR-SY-STD-xxx-4E		1.00	1.00	na	na	By request only; Req In-wnty prod
	PR	XC4000E		X	X					PR-VL-STD-xxx-4E	1.00	1.00	1.00	na	na	By request only; Req In-wnty prod
	PR	XC4000E		X	X					PR-3PA-STD-xxx-4E	1.00	1.00	1.00	na	na	By request only; Req In-wnty prod

KEY: *N=New Product, E= Engineering Software for in-warranty users by Request Only, U= Update by request only, PR = Pre-release requiring in-warranty status or Product Marketing apporval*

COMPONENT AVAILABILITY CHART

PINS	TYPE	CODE	XC3020A	XC3030A	XC3042A	XC3064A	XC3090A	XC3020L	XC3030L	XC3042L	XC3064L	XC3090L	XC3142L	XC3190L	XC3120A	XC3130A	XC3142A	XC3164A	XC3190A	XC3195A	XC4003E	XC4005E	XC4006E	XC4008E	XC4010E	XC4013E	XC4020E	XC4025E	XC4028EX	XC4036EX	XC4044EX	XC4052XL	XC4062XL	XC4085XL	XC40125XL
44	PLASTIC LCC	PC44		◆												◆																			
	PLASTIC QFP	PQ44																																	
	PLASTIC VQFP	VQ44																																	
	CERAMIC LCC	WC44																																	
48	PLASTIC DIP	PD48																																	
64	PLASTIC VQFP	VQ64		◆					◆							◆																				
68	PLASTIC LCC	PC68	◆	◆											◆	◆																				
	CERAMIC LCC	WC68																																	
	CERAMIC PGA	PG68																																	
84	PLASTIC LCC	PC84	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆										
	CERAMIC LCC	WC84																																	
	CERAMIC PGA	PG84	◆	◆	◆										◆	◆	◆																		
100	CERAMIC QFP	CQ100																																	
	PLASTIC PQFP	PQ100	◆	◆	◆										◆	◆	◆				◆	◆													
	PLASTICTQFP	TQ100														◆	◆																		
	PLASTIC VQFP	VQ100		◆	◆				◆	◆			❖			◆	◆				◆														
	TOP BRZ. CQFP	CB100	◆		◆										◆	◆																			
120	CERAMIC PGA	PG120																			◆														
132	PLASTIC PGA	PP132			◆	◆											◆	◆																	
	CERAMIC PGA	PG132			◆	◆											◆	◆																	
144	PLASTICTQFP	TQ144			◆	◆	❖			◆	◆	❖	◆	◆			◆	◆				◆	◆												
	CERAMIC PGA	PG144																																	
156	CERAMIC PGA	PG156																				◆	◆												
160	PLASTIC PQFP	PQ160				◆	◆											◆	◆	◆		◆	◆	◆	◆	◆								
164	CERAMIC QFP	CQ164																																	
	TOP BRZ. CQFP	CB164					◆												◆	◆															
175	PLASTIC PGA	PP175					◆												◆	◆															
	CERAMIC PGA	PG175					◆												◆	◆															
176	PLASTICTQFP	TQ176					◆					◆		◆					◆																
184	CERAMIC PGA	PG184																																	
191	CERAMIC PGA	PG191																						❖	❖										
196	TOP BRZ. CQFP	CB196																																	
208	PLASTIC PQFP	PQ208					◆												◆	◆		◆	◆	◆	◆	◆								
	METAL MQFP	MQ208																																	
	HI-PERF QFP	HQ208																								◆	◆		◆						
223	CERAMIC PGA	PG223																		◆						◆	◆	◆							
225	PLASTIC BGA	BG225																							◆	◆									
	WINDOWED BGA	WB225																																	
228	TOP BRZ. CQFP	CB228																																	
240	PLASTIC PQFP	PQ240																								◆									
	METAL MQFP	MQ240																																	
	HI-PERF QFP	HQ240																								◆	◆	❖	◆						
299	CERAMIC PGA	PG299																										◆	◆						
304	HI-PERF QFP	HQ304																										◆	◆	◆					
352	PLASTIC BGA	BG352																											◆						
411	CERAMIC PGA	PG411																												◆	◆	❖			
432	PLASTIC BGA	BG432																												◆	◆	◆			
499	CERAMIC PGA	PG499																															◆	◆	
596	PLASTIC BGA	BG596																															◆	◆	◆	◆

◆ = Product currently shipping or planned ❖ = New since last issue of XCELL

MAY 1996

PINS	TYPE	CODE	XC4005L	XC4010L	XC4013L	XC5202	XC5204	XC5206	XC5210	XC5215	XC8100	XC8101	XC8103	XC8106	XC8109	XC7236A	XC7272A	XC7318	XC7336	XC7336Q	XC7354	XC7372	XC73108	XC73144	XC9536	XC95108	XC95216
44	PLASTIC LCC	PC44									◆		◆			◆		◆	◆	◆	◆				◆		
	PLASTIC QFP	PQ44																◆	◆	◆							
	PLASTIC VQFP	VQ44											◆										◆		◆		
	CERAMIC LCC	WC44														◆			◆	◆	◆						
48	PLASTIC DIP	PD48																									
64	PLASTIC VQFP	VQ64																									
68	PLASTIC LCC	PC68															◆				◆	◆					
	CERAMIC LCC	WC68															◆				◆	◆					
	CERAMIC PGA	PG68																									
84	PLASTIC LCC	PC84	◆	◆		◆	◆	◆	◆			◆	◆	◆	◆		◆					◆	◆			◆	
	CERAMIC LCC	WC84															◆					◆	◆				
	CERAMIC PGA	PG84																									
100	CERAMIC QFP	CQ100																									
	PLASTIC PQFP	PQ100				◆	◆	◆				◆	◆	◆								◆	◆			◆	
	PLASTIC TQFP	TQ100																								◆	
	PLASTIC VQFP	VQ100				❖	❖	❖																			
	TOP BRZ. CQFP	CB100																									
120	CERAMIC PGA	PG120																									
132	PLASTIC PGA	PP132																									
	CERAMIC PGA	PG132																									
144	PLASTIC TQFP	TQ144				◆	◆	◆	◆																		
	CERAMIC PGA	PG144																					◆				
156	CERAMIC PGA	PG156				◆	◆	❖																			
160	PLASTIC PQFP	PQ160					◆	◆	◆	❖					◆								◆	◆		◆	◆
164	CERAMIC QFP	CQ164																									
	TOP BRZ. CQFP	CB164																									
175	PLASTIC PGA	PP175																									
	CERAMIC PGA	PG175																									
176	PLASTIC TQFP	TQ176		◆				❖	❖																		
184	CERAMIC PGA	PG184																									
191	CERAMIC PGA	PG191							◆																		
196	TOP BRZ. CQFP	CB196																									
208	PLASTIC PQFP	PQ208	◆	◆	◆			◆	◆																		
	METAL MQFP	MQ208																									
	HI-PERF QFP	HQ208								◆																	◆
223	CERAMIC PGA	PG223																									
225	PLASTIC BGA	BG225			◆				◆	◆					◆								◆	◆			
	WINDOWED BGA	WB225																									
228	TOP BRZ. CQFP	CB228																									
240	PLASTIC PQFP	PQ240			◆				◆																		
	METAL MQFP	MQ240																									
	HI-PERF QFP	HQ240								◆																	
299	CERAMIC PGA	PG299								◆																	
304	HI-PERF QFP	HQ304								◆																	
352	PLASTIC BGA	BG352								❖																	
411	CERAMIC PGA	PG411																									
432	PLASTIC BGA	BG432																									
499	CERAMIC PGA	PG499																									
596	PLASTIC BGA	BG596																									

XILINX ALLIANCE - EDA COMPANIES & PRODUCTS - MAY 1996 - 1 OF 2

	Company Name	Product Name	Version	Function	DesignKit	2K/3K/4K	XC 5200	CPLD 7K9K	XC 8100	Uni Lib	PLATFORMS PC	Sun	RS6000	HP7
DIAMOND	Aldec	Active-CAD	2.0	Schematic Entry & Simulation & HDL Editor	Included	✓	✓	✓	✓	✓	✓			
	Cadence	Verilog	2.3.2	Simulation	Xilinx Front End	✓	✓	7k	✓	✓		✓	✓	✓
		Concept	2.1	Schematic Entry	Xilinx Front End	✓		7k	✓	✓		✓	✓	✓
		FPGA Designer	9504	Topdown FPGA Synthesis	Call Xilinx	✓	✓	7k	Q3	✓		✓	✓	✓
		Synergy	2.3	FPGA Synthesis	Call Xilinx	✓	✓	7k	✓	✓		✓	✓	✓
		Composer	4.3.4/4.4	Schematic Entry	Xilinx Front End	✓		7k	TBD	✓		✓	✓	✓
	Mentor Graphics	Autologic	A.x_F	Synthesis	Xilinx Synthesis Lib.	✓	✓	7k	✓	✓		✓	✓	✓
		Design Architect	A.x_F	Schematic Entry	Call Xilinx	✓	✓	7k	✓	✓		✓	✓	✓
		QuickSim II	A.x_F	Simulation	Call Xilinx	✓	✓	7k	✓	✓		✓	✓	✓
	OrCAD	Simulate (Win)	6.10	Simulation	Call Xilinx				Q3		✓			
		VST 386+ (DOS)	1.2	Simulation	Call Xilinx	✓		7k		✓	✓			
		Capture (Win)	6.11	Schematic Entry	Call Xilinx	✓	✓	7k	✶✓	✓	✓			
		SDT 386+ (DOS)	1.2	Schematic Entry	Call Xilinx	✓		7k		✓	✓			
		PLD 386+ (DOS)	2.0	Synthesis	Call OrCAD	✓		7k			✓			
	Synario Design Automation	ABEL	6.2	Synthesis, Simulation	XEPLD Fitter			7k,9k		✓	✓			
		Synario	2.2	Schematic Entry, Synthesis & Simulation	SYN-LCA	✓	✓	7k,9k	TBD	✓	✓			
	Synopsys	FPGA Compiler	3.4a	Synthesis	Call Xilinx	3K, 4K	✓	7k	✓	✓		✓	✓	✓
		VSS	3.4a	Simulation	Call Xilinx	✓	✓	7k	✓	✓		✓	✓	✓
		Design Compiler	3.4a	Synthesis	Call Xilinx	✓	✓	7k	✓	✓		✓	✓	✓
	Viewlogic	WorkView Office	5.5	Schem/Sim/Synth	Call Xilinx	✓	XACT6	✓	✓	✓	✓	✓	✓	✓
		ProSynthesis	5.02	Synthesis	call Xilinx	✓	XACT6	7k	✓	✓	✓	✓	✓	✓
		ProSim	6.1	Simulation, Timing Analysis	call Xilinx	✓	XACT6	7k	✓	✓	✓	✓	✓	✓
		ProCapture	6.1	Schematic Entry	call Xilinx	✓	XACT6	7k	✓	✓	✓	✓	✓	✓
RUBY	Capilano Computing	Design Works	3.1	Schematic Entry/Sim	XD-1	✓				✓	✓			
	Compass Design Automation	ASIC Navigator		Schematic Entry	Xilinx Design Kit	✓	✓	7k	Q3			✓		✓
		X-Syn		Synthesis		✓			Q3			✓		✓
		QSim		Simulation		✓	✓	7k	Q3			✓		✓
	Escalade	DesignBook	2.0	Design Entry		✓				✓	✓	✓		
	Exemplar Logic	Galileo	3.2	Synthesis/Timing Analysis Simulation	Included	3k,4k	✓	7k	✓		✓	✓		
	IK Technology Co.	Ishizue Professionals	1.05.02	Schematic Entry/Simulation	Xilinx Design Kit	✓					✓	✓		✓
	IKOS Systems	Voyager	2.11	Simulation	Xilinx Tool Kit	✓						✓		✓
	INCASES Engineering GmbH	Theda	4.0	Schematic Entry	Xilinx Kit	✓								✓
	ISDATA	LOG/iC Classic	4.2	Synthesis	LCA-PP	✓		7k		✓	✓	✓		✓
		LOG/iC2	4.2	Synthesis Simulation	Xilinx Mapper	✓	✓	7k		✓	✓	✓		✓
	Logic Modeling Corp. (Synopsis Division)	Smart Model		Simulation Models	In Smart Model Lib.	✓	✓	7k,9k			✓	✓	✓	✓
		LM1200		Hardware Modeler	Xilinx Logic Module	✓		7k,9k			✓	✓	✓	
	Model Technology	V-System/VHDL	4.4	Simulation		3k,4k			✓	✓	✓	✓	✓	
	Protel Technology	Advanced Schematic	3.1	Schematic Entry/Client Server	Xilinx Interface	✓		7k			✓			
	Quad Design Technology	Motive	4.3	Timing Analysis	XNF2MTV	✓			✓		✓	✓	✓	✓
	SimuCad	Silos III	95.100	Simulation	Included	✓				✓	✓			
	Sophia Sys & Tech	Vanguard	5.31	Schematic Entry	Xilinx I/F Kit	✓		✓			✓	✓		✓
	Summit Design Corp.	Visual HDL	3.0	Graphical Design Entry/Simulation/Debug	EDIF Interface	✓	✓	✓	✓	✓	✓	✓	✓	✓
	Synplicity, Inc.	Synplify-Lite	2.5f	Synthesis	Xilinx Mapper	3K, 4K	✓			✓	✓	✓		
		Synplify	2.5f	Synthesis	included	3K, 4K	✓			✓	✓	✓		
	TopDown Design Solutions	V-BAK	1.1	XNFto VHDL translator	XNF interface	✓	✓			✓	✓	✓	✓	
	Veda Design Automation Inc	Vulcan	4.5	Simulation	XILINX Tool Kit	✓						✓		✓

XILINX ALLIANCE - EDA COMPANIES & PRODUCTS - MAY 1996 - 2 OF 2

	Company Name	Product Name	Version	Function	DesignKit	2K/3K/4K	XC 5200	CPLD 7K9K	XC 8100	Uni Lib	PLATFORMS PC	Sun	RS6000	HP7
RUBY	Veribest	Veribest VHDL	14.0	Schematic Entry	Xilinx FPGA Design Kit	3k,4k			TBD	✓	✓	✓		✓
		Veribest Verilog	14.0	Simulation	Xilinx FPGA Design Kit	3k,4k			✓	✓	✓	✓		✓
		VeriBest Simulator	14.0	Simulation	Xilinx FPGA Design Kit	3k,4k	✓		TBD	✓	✓	✓		✓
		DMM	14.x	Design Management	Xilinx FPGA Design Kit	3k,4k	✓		TBD	✓	✓	✓		✓
		VeriBest Synthesis	14.0	Synthesis	Xilinx FPGA Design Kit	3k,4k	✓		TBD	✓	✓	✓		✓
		Synovation	12.2	Synthesis	Xilinx FPGA Design Kit	3k,4k				✓	✓	✓		✓
		PLDSyn	12.0	Design Entry Synthesis		✓		7k			✓	✓		✓
		VerBest Design Capture	14.x	Design Capture	Xilinx FPGA Design Kit	3k,4k	✓		TBD	✓	✓	✓		✓
EMERALD	Accolade Design Automation	PeakVHDL		Synthesis, Simulation	Included	✓				✓	✓			
	ACEO Technology, Inc.	Asyn	3.3	Synthesis	Included	✓	✓			✓	✓	✓	✓	✓
		Softwire	3.3	Multi-FPGA Partitioning	Included	✓	✓			✓	✓	✓	✓	✓
		Gatran	3.3	ASIC to FPGA Netlist Mapping	Included	✓	✓			✓	✓	✓	✓	✓
	Acugen Software, Inc.	Sharpeye	2.60	Testability Analysis	AALCA interface	✓	✓	7k			✓	✓		✓
		ATGEN	2.60	Automatic Test Generation	AALCA interface	✓	✓	7k			✓	✓		✓
	ALPS LSI Technologies, Inc.	Edway Design Systems		Synthesis/Simlulation		✓		✓			✓			
	Aptix Corporation	System Explorer	2.1	System Emulation	Axess 2.Yes	✓								
	Aptix Corporation	ASIC Explorer	2.3	ASIC Emulation	Axess 2.3	4K	✓			✓				
	Aster Ingenierie S.A.	XILLAS	4.1	LASAR model generation	Worst Case Simulation	✓		7k			✓	✓		✓
	Chronology Corporation	TimingDesigner	3.0	Timing Specification and Analysis	Included	✓	✓	7k,9k	✓	✓	✓	✓		✓
		QuickBench	1.0	Visual Test Bench Generator	Included	✓	✓	7k,9k	✓	✓	✓	✓		✓
	CINA -Computer Integrated Network Analysis	SmartViewer	1.0c	Schematic	XNFInterface	✓		7k			✓			
	Epsilon Design Systems	Logic Compressor		Synthesis optimization		✓	✓		✓		✓	✓	✓	✓
	Flynn Systems	Probe	3.0	Testability Analysis	Xilinx Kit	✓	✓	7k			✓			
		FS-ATG	3.0	Test Vector Generation	Xilinx Kit	✓	✓	7k			✓			
		CKTSIM	3.0	Logic Analysis	Xilinx Kit	✓	✓	7k			✓			
		FS-SIM	3.0	Simulation	Xilinx Kit	✓	✓	7k			✓			
	Fujitsu LSI	PROVERD		Top-Down Design System	Included	3k,4k					✓			
	Harmonix Corporation	PARTHENON	2.3	Synthesis		4k		7k			✓	✓		
	MINC	PLDesigner-XL	3.3	Synthesis	Xilinx Design Module	✓					✓	✓		✓
	Teradyne	Lasar	6	Simulation	Xilinx I/F Kit	✓						✓		✓
	The Rockland Group	One-Hot State Machine Lib.		Graphic Design for One-Hot State Machines	Xilinx Kit	✓	✓			✓	✓	✓	✓	✓
	Tokyo Electron Limited	ViewCAD	1.2	FLDL to XNF translator	XNFGEN	✓								
	Trans EDA Limited	TransPRO	1.2	Synthesis	Xilinx Library	✓						✓		✓
	Zuken	Tsutsuji		Synthesis/Simulation	XNFInterface	3k,4k						✓	✓	✓
	Zycad	Paradigm RP		Rapid Prototyping		✓						✓		✓
		Paradigm XP		Gate-level Sim		✓						✓		✓

Alliance Program Categories:

Diamond: These partners have strong strategic relationships with Xilinx and have a direct impact on our releases. Typically, Xilinx is directly involved in the development and testing of the interface to XACT*step* software for these products.

Ruby: These partners have a high degree of compatibility and have repeatedly shown themselves to be significant contributors to our users' development solutions.

Emerald: Proven Xilinx compatibility

XILINX ALLIANCE - EDA CONTACTS - MAY 1996

Company Name	Contact Name	Phone Number	E-mail Address
Accolade Design Automation	Dave Pellerin	(800) 470-2686	pellerin@seanet.com
ACEO Technology, Inc.	Ray Wei	(510) 656-2189	ray@aceo.com
Acugen Software, Inc.	Nancy Hudson	(603) 881-8821	
Aldec	David Rinehart	(702) 456-1222 x12	dave@aldec.com
ALPS LSI Technologies, Inc.	David Blagden	441489571562	
Alta Group	Baruch Deutsch	(408) 523-4137	baruch@altagroup.com
Aptix Corporation	Henry Verheyen	(408) 428-6209	henry@aptix.com
Aster Ingenierie S.A.	Christopher Lotz	+33-99537171	
Cadence	Itzhak Shapira Jr.	(408) 428-5739	itzhak@cadence.com
Capilano Computing	Chris Dewhurst	(604) 522-6200	
Chronology Corporation	MikeMcClure	(206) 869-4227 x116	sales@chronology.com
CINA - Computer Integrated Network Analysis	Brad Ashmore	(415) 940-1723	bashmore@cina.com
Compass Design Automation	Marcia Murray	(408) 474-5002	
Epsilon Design Systems, Inc.	Cuong Do	(408) 934-1536	CuongEDS@aol.com
Escalade	RodDudzinski	(408) 654-1651	
Exemplar Logic	Mary Murphy	(510) 337 3728	murphy@exemplar.com
Flynn Systems	Matt Van Wagner	(603) 598-4444	matt@flynn.com
Fujitsu LSI	Masato Tsuru	+81-4-4812-8043	
Harmonix Corporation	Shigeaki Hakusui	(617) 935-8335	
IK Technology Co.	Tsutomu Someya	+81-3-3839-0606	someya@ikt.co.jp
IKOS Systems	Brad Roberts	(408) 366-8509	brad@ikos.com
INCASES Engineering GmbH	Christian Kerscher	+49-89-839910	ckerscher@muc.incases.com
ISDATA	Ralph Remme	+49-72-1751087	ralph.remme@isdata.de
ITS	Frank Meunier	(508) 897-0028	
Logic Modeling Corp. (Synopsis Division)	Marnie McCollow	(503) 531-2412	marniem@synopsys.com
Logical Devices	David Mot	(303) 279-6868 x209	
Mentor Graphics	SamPicken	(503) 685-1298	samp@mentorg.COM
MINC	Kevin Bush	(719) 590-1155	
Minelec	Marketing Department	+32-02-4603175	
Model Technology	Greg Seltzer	(503) 526-5465	greg_seltzer@model.com
OrCAD	Mike Jingozian	(503) 671-9500	mikej@orcad.com
Protel Technology	Luise Markham	(408) 243-8143	
Quad Design Technology, Inc.	Britta Sullivan	(805) 988-8250	
SimuCad	Richard Jones	(510) 487-9700	richard@simucad.com
Sophia Sys & Tech	Tom Tilbon	(408) 232-4764	
Summit Design Corporation	Ed Sinclair	(503) 643-9281	
Synario Design Automation	Jacquelin Taylor	(206) 867-6257	taylorja@data-io.com
Synopsys	Lynn Fiance	(415) 694-4289	lynnf@synopsys.com
Synplicity, Inc.	Alisa Yaffa	(415) 961-4962	alisa@synplicity.com
Teradyne	Mike Jew	(617) 422-3753	jew@teradyne.com
The Rockland Group	Rocky Awalt	(916) 622-7935	rocky@trg.com
Tokyo Electron Limited	Shige Ohtani	+81-3-334-08198	shige@xilinx.tel.com.jp
TopDown Design Solutions	Art Pisani	(603) 888-8811	
Trans EDA Limited	James Douglas	+44-703-255118	
VEDA DESIGN AUTOMATION INC	Kathie O'Toole	(408) 496-4515	
Veribest	Ravi Ravikumar	(415) 691-6445	rravikum@veribest.com
Viewlogic	Preet Virk	(508) 480-0881	pvirk@viewlogic.com
Visual Software Solutions, Inc.	Andy Bloom	(305) 423-8448	
Zuken	Makato Ikeda	+81-4-594-27787	
Zycad	Mike Hannig	(201) 989-2900	

Inquiries about the Xilinx Alliance Program can be e-mailed to alliance@xilinx.com.

PROGRAMMER SUPPORT FOR XILINX XC1700 SERIAL PROMS — MAY 1996

MANUFACTURER	MODEL	1736A 1765	17128	1718D 1736D 1765D	1718L 1765L	17128D 17256D
ADVANTECH	PC-UPROG		V2.1	5/96	5/96	5/96
	LABTOOL-48		V1.0	5/96	5/96	5/96
ADVIN	PILOT-U24	10.83	10.83	5/96	5/96	5/96
	PILOT-U28	10.83	10.83	5/96	5/96	5/96
	PILOT-U32	10.83	10.83	5/96	5/96	5/96
	PILOT-U40	10.83	10.83	5/96	5/96	5/96
	PILOT-U84	10.83	10.83	5/96	5/96	5/96
	PILOT-142	10.83	10.83	5/96	5/96	5/96
	PILOT-143	10.83	10.83	5/96	5/96	5/96
	PILOT-144	10.83	10.83	5/96	5/96	5/96
	PILOT-145	10.83	10.83	5/96	5/96	5/96
B&C MICROSYSTEMS	Proteus-UP40	V3.4e	V3.4e	V3.7Q	V3.7Q	V3.7Q
BP MICROSYSTEMS	CP-1128	C	V2.17			
	EP-1140	C	V2.17			
	BP-1200	V3.12	V3.12	V3.15	V3.15	V3.15
	BP-2100	V3.12	V3.12	V3.15	V3.15	V3.15
BYTEK	135H-FT/U	V42	V51			
	MTK-1000	V42	V51			
	MTK-2000	V42	V51			
	MTK-4000	V42	V51			
DATA I/O	UniSite	V4.0	V4.1	4/96	4/96	4/96
	2900	V2.1	V2.2	4/96	4/96	4/96
	3900	V1.5	V1.6	4/96	4/96	4/96
	AutoSite	V1.5	V1.6	4/96	4/96	4/96
	ChipLab	V1.1	V1.0	4/96	4/96	4/96
	2700	V3.0	V3.0	V4/96	4/96	4/96
DEUS EX MACHINA	XPGM		V1.40	V1.40	V1.40	V1.40
EE TOOLS	ALLMAX/ALLMAX+	V1.3	V1.5	V2.44	V2.44	V2.441
	MEGAMAX	V2.34	V2.34	V1.1E	V1.1E	V1.1E
ELAN DIGITAL SYSTEMS	3000-145	C	C			
	5000-145	C	C			
	6000 APS	K2.01	K2.02	3Q96	3Q96	3Q96
HI-LO SYSTEMS RESEARCH	All-03A	V3.30	V3.30	V3.58	V3.58	V3.58
	All-07	V3.30	V3.30	V3.58	V3.58	V3.58
ICE TECHNOLOGY LTD	Micromaster 1000/1000E	V1.1	V3.00	5/96	5/96	5/96
	Speedmaster 1000/1000E	V1.1	V3.00	5/96	5/96	5/96
	Micromaster LV	V3.003	V3.00	5/96	5/96	5/96
	LV40 Portable					
	Speedmaster LV		V3.00	5/96	5/96	5/96
LEAP ELECTRONICS	LEAPER-10					
	LEAP-SU1					
	LEAP-U1					

MANUFACTURER	MODEL	1736A 1765	17128	1718D 1736D 1765D	1718L 1765L	17128D 17256D	17128L 17256L
LINK COMPUTER GRAPHICS	CLK-3100	V5.08	V5.08	5/96			
LOGICAL DEVICES	ALLPRO-40	V2.2					
	ALLPRO-88	V2.2	V2.3	5/96		5/96	
	ALLPRO-88XR	V1.1	V2.3	5/96		5/96	
	CHIPMASTER 3000	V2.0	V2.1	5/96	5/96		
	CHIPMASTER 6000	V1.15					
	XPRO-1	V1.01	V1.01	5/96	5/96	5/96	
MQP ELECTRONICS	MODEL 200	C	6.45				
	SYSTEM 2600						
	PINMASTER 48						
MICROPROSS	ROM 5000 B	C	V1.70	5/96	5/96	5/96	
	ROM 3000 U	C	V3.60	5/96	5/96	5/96	
	ROM 9000				6/96	6/96	
NEEDHAM'S ELECTRONICS	EMP20	V1.5	V1.5	V3.10	V3.10	V3.10	
RED SQUARE	IQ-180	C	V9.2	6/96	6/96	6/96	
	IQ-280	C	V9.2	6/96	6/96	6/96	
	Uniwriter 40	C	V9.2	6/96	6/96	6/96	
	Chipmaster 5000	C	V9.2	6/96	6/96	6/96	
SMS	Expert	B/93	A/94	B/96	B/96	B/96	B/96
	Optima	B/93	A/94	B/96	B/96	B/96	B/96
	Multisyte			B/96		B/96	
	Sprint Plus 48	B/93	A/94	B/96	B/96		
STAG	Eclipse		4.4	5/96	5/96	5/96	
	Quasar	10.76C	10.76C	5/96	5/96	5/96	
SUNRISE	T-10 UDP	V3.31	V3.31	6/96	6/96	6/96	
	T-10 ULC	V3.31	V3.31	6/96	6/96	6/96	
SUNSHINE	POWER-100	V8.18	V8.18	6/96	6/96	6/96	
	EXPRO-60/80	V8.18	V8.18	6/96	6/96	6/96	
SYSTEM GENERAL	TURPRO-1	V2.21F	V2.21F	5/96	5/96	5/96	
	TURPRO-1 FLX	V2.21F	V2.21F	5/96	5/96	5/96	
	APRO	C	V2.14	5/96	5/96		
	MULTI-APRO	V1.13E	V1.13E	5/96	5/96	5/96	
TRIBAL MICROSYSTEMS	TUP-300	C	V3.31	V3.58	V3.58	V3.58	
	TUP-400	C	V3.31	V3.58	V3.58	V3.58	
	FLEX-700	C	V3.31	V3.58	V3.58	V3.58	
XELTEK	SuperPRO	1.5B	1.7C	5/96		5/96	
	SuperPRO II	1.5B	1.7C	5/96		5/96	
XILINX	HW-112	4.02	4.06	5.00	5.00	5.00	
	HW-130		1.11	2.01	2.01	2.01	1.00

PROGRAMMER SUPPORT FOR XILINX XC7200 EPLDS — MAY 1996

MANUFACTURER	MODEL	7236	7236A	7272	7272A
ADVANTECH	PC-UPROG LabTool-48	Apr-96 V1.31A	Apr-96 V1.31A	Apr-96 V1.31A	Apr-96 V1.31A
ADVIN SYSTEMS	Pilot-U40 Pilot-U84	May-96 May-96	May-96 May-96	 May-96	 May-96
B&C MICROSYSTEMS INC.	Proteus	May-96	May-96	May-96	May-96
BP MICROSYSTEMS	BP-1200 BP-2100	3.15 3.15	3.15 3.15	3.15 3.15	3.15 3.15
DATA I/O	UniSite 2900 3900 AutoSite	May-96 May-96 May-96 May-96	May-96 May-96 May-96 May-96	May-96 May-96 May-96	May-96 May-96 May-96
DEUS EX MACHINA ENGINEERING	XPGM	V1.40	V1.40	V1.40	V1.40
E E TOOLS	ALLMAX/ALLMAX+ MEGAMAX	V2.4U V1.1E	V2.4U V1.1E	V2.4U V1.1E	V2.4U V1.1E
ELAN DIGITAL SYSTEMS	6000 APS	3Q96	3Q96	3Q96	3Q96
HI-LO SYSTEMS RESEARCH	All-03A All-07	V3.09 V3.09	V3.09 V3.09	V3.09 V3.09	V3.09 V3.09
ICE TECHNOLOGY LTD	Micromaster 1000/E Speedmaster 1000/E Micromaster LV Speedmaster LV	May-96 May-96 May-96 May-96	May-96 May-96 May-96 May-96	May-96 May-96 May-96 May-96	May-96 May-96 May-96 May-96
LEAP ELECTRONICS	LEAPER-10 LEAP-SU1 LEAP-U1				
LOGICAL DEVICES	ALLPRO-88 ALLPRO-88XR XPRO-1	Jul-96 Jul-96 Jul-96	Jul-96 Jul-96 Jul-96	Jul-96 Jul-96 Jul-96	Jul-96 Jul-96 Jul-96
MICROPROSS	ROM9000				
MQP ELECTRONICS	SYSTEM 2000 PINMASTER 48				
NEEDHAM'S ELECTRONICS	EMP20	V3.10	V3.10	V3.10	V3.10
SMS	Expert Optima Multisyte	B/96 B/96 B/96	B/96 B/96 B/96	B/96 B/96 B/96	B/96 B/96 B/96
STAG	Eclipse	May-96	May-96	May-96	May-96
SUNRISE ELECTRONICS	T-10 UDP T-10 ULC	Aug-96 Aug-96	Aug-96 Aug-96	Aug-96 Aug-96	Aug-96 Aug-96
SUNSHINE ELECTRONICS	POWER-100 EXPRO-60/80	Aug-96 Aug-96	Aug-96 Aug-96		
SYSTEM GENERAL	TURPRO-1/FX MULTI-APRO	Jul-96 Jul-96	Jul-96 Jul-96	Jul-96 Jul-96	Jul-96 Jul-96
TRIBAL MICROSYSTEMS	TUP-300 TUP-400 FLEX-700	V3.09 V3.09 V3.09	V3.09 V3.09 V3.09	V3.09 V3.09 V3.09	V3.09 V3.09 V3.09
XELTEK	SUPERPRO SUPERPRO II	Aug-96 Aug-96	Aug-96 Aug-96	Aug-96 Aug-96	Aug-96 Aug-96
XILINX	HW-130	1.14	1.14	1.04	1.04

WHITE=changed since last issue

PROGRAMMER SUPPORT FOR XILINX XC7300 EPLDS — MAY 1996

MANUFACTURER	MODEL	7318	7336	7336Q	7354	7372	73108	73144
ADVANTECH	PC-UPROG							
	LABTOOL-48	V1.31A	V1.31A	V1.31A	V1.31A	V1.31A	V1.31A	
ADVIN SYSTEMS	PILOT-U40	May-96	May-96	May-96	May-96			
	PILOT-U84	May-96	May-96	May-96	May-96	May-96	May-96	
B&C MICROSYSTEMS, INC.	PROTEUS	May-96	May-96	May-96	May-96	May-96	May-96	May-96
BP MICROSYSTEMS	BP-1200	V3.15	V3.15	V3.15	V3.15	V3.15	V3.15	
	BP-2100	V3.15	V3.15	V3.15	V3.15	V3.15	V3.15	
DATA I/O	2900	May-96	May-96	May-96	May-96			
	3900/AUTOSITE	May-96	May-96	May-96	May-96	May-96	May-96	
	UNISITE	May-96	May-96	May-96	May-96	May-96	May-96	
DEUS EX MACHINA ENGINEERING		XPGM	V1.40	V1.40	V1.40	V1.40	V1.40	V1.40
EE TOOLS	ALLMAX/ALLMAX+	V2.4U	V2.4U	V2.4U	V2.4U			
	MEGAMAX	V1.1E	V1.1E	V1.1E	V1.1E	V1.1E		
ELAN	6000 APS	3Q96	3Q96	3Q96	3Q96	3Q96		
HI-LO SYSTEMS RESEARCH	ALL-03A	V3.09	V3.09	V3.09	V3.09	V3.09	V3.09	
	ALL-07	V3.09	V3.09	V3.09	V3.09	V3.09	V3.09	
ICE TECHNOLOGY LTD	Micromaster 1000/E	May-96	May-96		May-96	May-96	May-96	
	Speedmaster 1000/E	May-96	May-96		May-96	May-96	May-96	
	Micromaster Lv	May-96	May-96	May-96	May-96	May-96	May-96	
	Speedmaster Lv	May-96	May-96	May-96	May-96	May-96	May-96	
LOGICAL DEVICES	ALLPRO-88	Jun-96	Jun-96	Jun-96	Jun-96	Jun-96		
	ALLPRO-88XR	Jun-96	Jun-96	Jun-96	Jun-96	Jun-96		
	XPRO-1	Jun-96	Jun-96	Jun-96	Jun-96	Jun-96		
MICROPROSS	ROM9000							
MQP ELECTRONICS	SYSTEM 2000							
	PINMASTER 48							
NEEDHAM'S ELECTRONICS	EMP20	V3.10	V3.10	V3.10	V3.10	V3.10	V3.10	
SMS	EXPERT	B/96	B/96	B/96	B/96	B/96	B/96	
	OPTIMA	B/96	B/96	B/96	B/96	B/96	B/96	
STAG	ECLIPSE	May-96	May-96	May-96	May-96	May-96	May-96	
SUNRISE	T-10 UDP	Jul-96	Jul-96	Jul-96	Jul-96	Jul-96	Jul-96	
	T-10 ULC	Jul-96	Jul-96	Jul-96	Jul-96	Jul-96	Jul-96	
SUNSHINE ELECTRONICS	POWER-100	Jul-96	Jul-96	Jul-96	Jul-96	Jul-96	Jul-96	
EXPRO-60/80								
SYSTEMGENERAL	TURPRO-1/FX	Jun-96	Jun-96	Jun-96	Jun-96	Jun-96	Jun-96	
	MULTI-APRO	Jun-96	Jun-96	Jun-96	Jun-96	Jun-96	Jun-96	
TRIBAL MICROSYSTEMS	FLEX-700	V3.09	V3.09	V3.09	V3.09	V3.09	V3.09	
	TUP-300	V3.09	V3.09	V3.09	V3.09	V3.09		
	TUP-400	V3.09	V3.09	V3.09	V3.09	V3.09		
XELTEK	SUPERPRO	Jul-96	Jul-96	Jul-96	Jul-96	Jul-96		
	SUPERPRO II	Jul-96	Jul-96	Jul-96	Jul-96	Jul-96		
XILINX	HW-130	1.15	1.15	1.06	1.16	1.07	1.07	1.02

WHITE=changed since last issue

XC9500 CPLDs:
Managing the "Product Life Cycle"

The XC9500 CPLD family incorporates a unique combination of product features specifically developed to meet all the needs for in-system programming (ISP) throughout the entire "product life cycle." This product life cycle starts with board-level prototyping and system debug, advances to programming and board-level testing during manufacturing, and finally completes with field upgrades.

The XC9500 product features reduce the total cost of ownership by eliminating many of the traditional problems of product development using PLDs. These features include 5 V in-system programmability, superior pin-locking capability, support for 10,000 program/erase cycles, and full 1149.1 JTAG support for in-system debug and version control.

Design and Prototyping

ISP-capable CPLDs provide a definite benefit over devices requiring an external programmer. ISP devices eliminate the handling errors and damage associated with removing the chip from the socket on the circuit board.

Through the multiple design iterations of the debug and prototyping process, the ISP CPLD can be repeatedly reprogrammed with different patterns while soldered on a printed circuit board (PCB). The ability of the architecture and tools to support pin-locking — that is, the ability to maintain a fixed pinout while making logic changes internal to the device — is crucial to avoid expensive and time-consuming board rework. The XC9500 CPLD, originally architected for in-system programming, offers users the best in pin-locking capability

Many existing ISP devices are fabricated using traditional EEPROM technology. However, the advanced XC9500 FastFLASH™ technology provides several advantages over EEPROM technology. Foremost among these is programming endurance. EEPROM technology typically allows 100-1000 program/erase cycles. FastFLASH technology has an endurance of 10,000 program/erase cycles. This high programming endurance minimizes or eliminates costly board rework resulting from reprogramming failures.

System Integration

When the entire system is assembled for test and debug, all important logic states should be easily accessible, and internal logic implementations within each device should be capable of being checked. Each XC9500 device supports the IEEE 1149.1 boundary-scan specification, including INTEST and USERCODE instructions used to easily access and debug user logic and track pattern revisions, respectively.

Manufacturing

Concurrent programming of all XC9500 CPLDs in the system with the standard JTAG interface simplifies the manufacturing flow in the production stage. Programming can be done after board assembly. This reduces production inventories, eliminates the need for tracking multiple

devices with multiple codes, and abolishes the stand-alone programming step. The resulting reduction in programming time also results in significant cost savings, especially at high unit volumes. Furthermore, built-in version control and in-system board customization contribute to manufacturing flexibility, saving time and money.

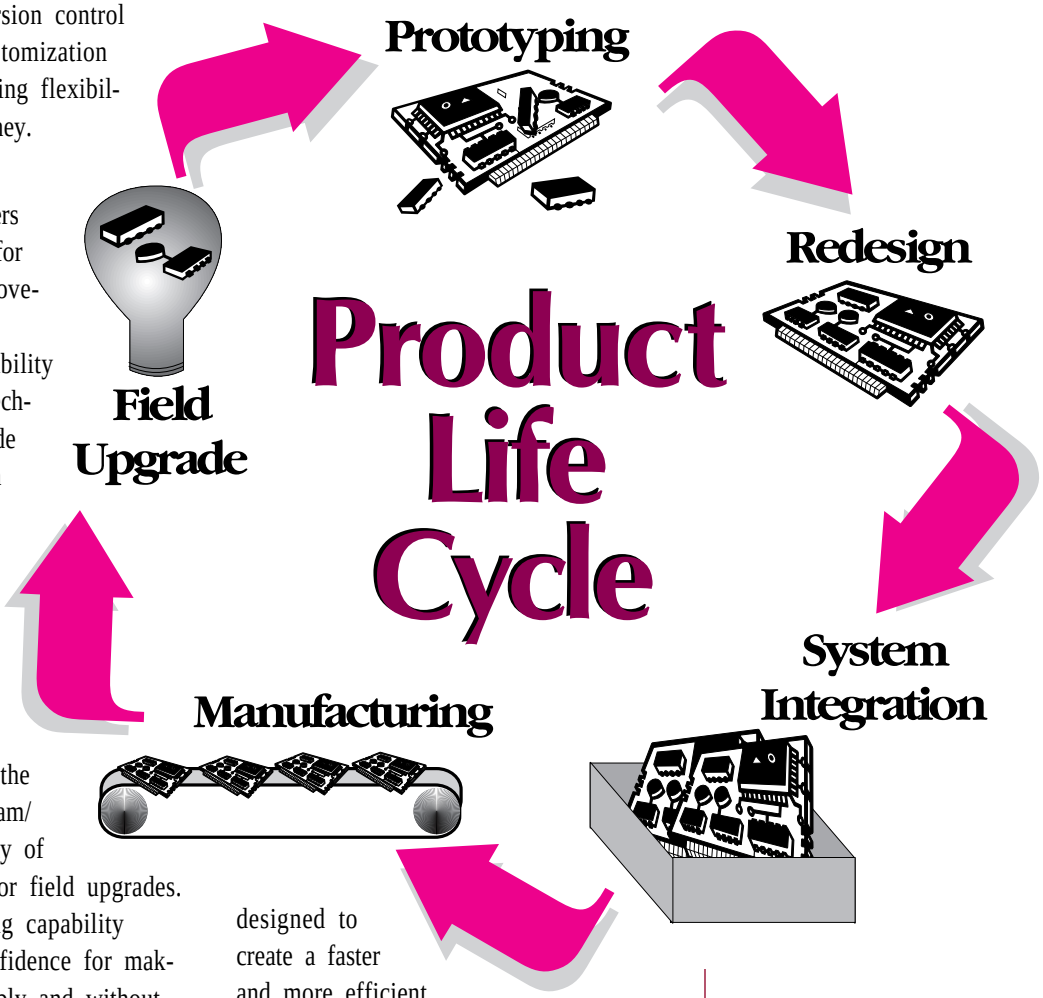
Field Upgrades

In the past, CPLD users were hesitant to design for field upgrades and improvements because of device reprogramming and reliability fears. With FastFLASH technology, true field upgrade and repair capability can now be designed into the product.

Since the XC9500 family is easily programmed through the JTAG port, any design can be easily prepared for "upgrading in the field." The 10,000 program/erase cycles ensure plenty of "reprogram headroom" for field upgrades. The enhanced pin-locking capability provides the highest confidence for making design changes reliably and without requiring expensive board re-work.

Two more XC9500 JTAG operations enhance field upgrade capability by aiding with system repair strategy. SAMPLE/PRELOAD operations allow output sampling and input stimulus preloading while the device is fully operational. The HIGHZ command disables bussed lines, facilitating the isolation and diagnosis of interconnect failures.

Xilinx products have always been designed to create a faster and more efficient product development process. By using the XC9500 in-system-programmable CPLDs, users can now add to those benefits the flexibility, pin-locking capability, performance, reliability and testability necessary for supporting the ever-shorter product life cycle. ◆



VHDL Made Easy!

Introducing the Xilinx *Foundation Series Software Solutions*

The software packages in the Xilinx Foundation Series™ are complete, fully-integrated sets of development tools that support a broad range of CPLD and FPGA design requirements. They include combinations of Windows-based design tools, integrating industry-standard hardware description languages (HDLs), synthesis, schematic entry and simulation tools with the Xilinx XACT*step*™ implementation tools. Emphasis has been placed on providing an easy-to-learn, HDL-based design environment.

Foundation Series packages are the lowest-cost and most complete design software packages in the programmable logic industry, featuring:

- Packages starting at $495
- Technology-independent migration paths
- Industry-standard HDL support: VHDL and ABEL-HDL
- Easy-to-achieve quality results with the Xilinx HDL Wizard
- Easy-to-use, push-button software
- Interactive tutorials for all tools including VHDL synthesis
- Tight integration with schematic, simulation, HDLs and XACT*step*
- Complete project and flow management

Breaking the Barriers to VHDL Synthesis

Xilinx provides the easiest way to obtain and learn VHDL with low-cost packages that include both a VHDL synthesis tool and a multimedia tutorial for learning VHDL at your own desk and at your own pace.

The multimedia tutorial educates users on:

- The VHDL Language
- How to write VHDL code for synthesis
- How to use VHDL successfully using the Xilinx Foundation Series tools
- How to write VHDL code to obtain the best results when targeting any Xilinx technology

The HDL Wizard, a set of tools that help users quickly learn and implement VHDL and ABEL-HDL designs, includes:

- Templates in the Language Assistant for cutting and pasting efficiently written code for common-functions (including user-created code)
- Error navigation and color coding in the HDL Editor for instant recognition of VHDL key words for easy debugging
- Tight integration of VHDL synthesis with the Schematic Editor to ease into VHDL design
- Automatic symbol generation and port declarations to accelerate design time and minimize the learning curve

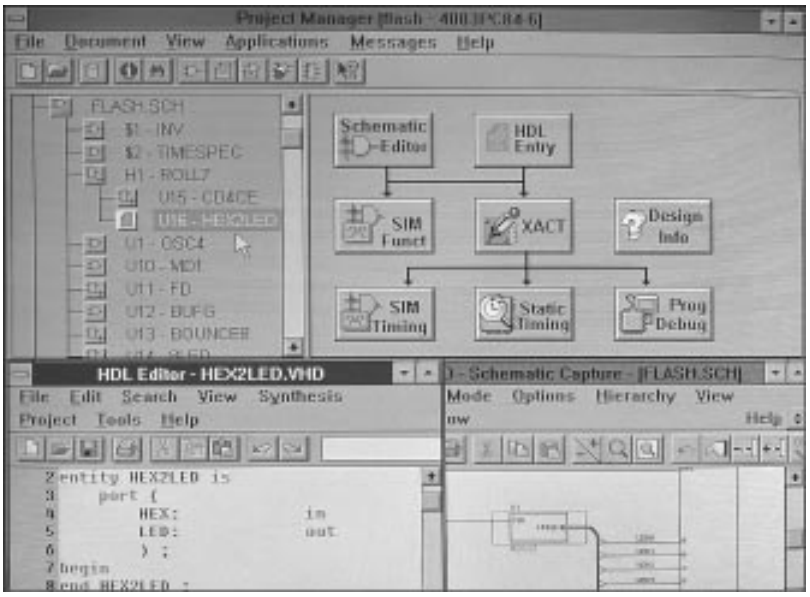
XILINX®

These tools not only make it easier to learn, write, and debug VHDL code, but also aid in producing high-quality code. The templates are written in coding styles specifically developed to produce efficient VHDL code for Xilinx device architectures.

The VHDL synthesis tool is not a subset VHDL tool; it is IEEE 1076 with 1164 std_logic compliant and will accept Synopsys-compatible code without modifications. Benchmark tests have demonstrated the competitiveness of the tool's results and execution run times. Typically, VHDL modules between 2,000 and 3,000 gates can be synthesized in two to three minutes.

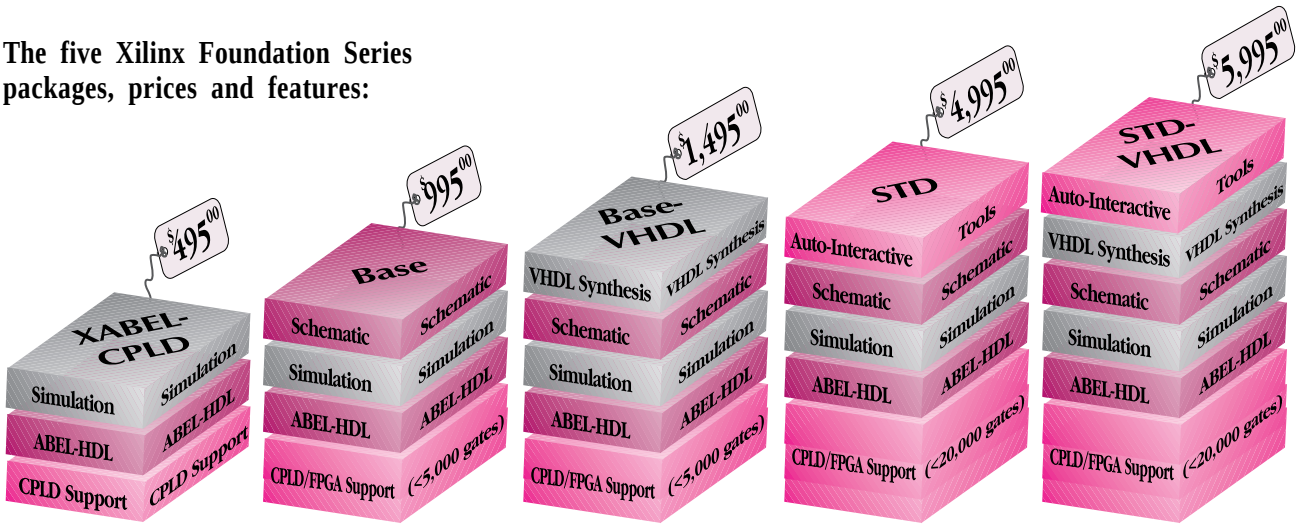
There are five Xilinx Foundation Series packages to match different needs, design entry preferences and budgets.

The breadth of technology supported in these packages makes them flexible enough to meet current and future programmable logic design needs. The tight integration of the toolset makes migration of designs from one Xilinx technology to another an easy, single-step process.



All of the Xilinx Foundation Series packages are available today for PC platforms running Windows v3.1. Versions for Windows 95 and Windows NT platforms will be introduced later this year. Contact your local Xilinx representative for more information or to request a demonstration. ◆

The five Xilinx Foundation Series packages, prices and features:



XACT-CPLD Supports Xilinx CPLD Design

Xilinx has expanded software support of our CPLD product lines with the introduction of the DS-560™ XACT-CPLD package. XACT-CPLD provides a complete low-cost, user-friendly environment for schematic, behavioral and VHDL design on the PC, Sun 4 and HP-700 platforms. XACT-CPLD supports both the XC7000 family, the industry's fastest and lowest-cost CPLDs, and the XC9500 family, the best in-system programmable CPLDs.

Features of the XACT-CPLD package include:

- **Automatic Device Selection**: Automatically implements the design in the smallest CPLD possible
- **Automatic Optimization, Partitioning and Mapping**: Designs are automatically optimized, partitioned and mapped into the device for optimal efficiency and design performance
- **XACT-Performance™**: Timing-driven optimization collapses logic to meet user-specified critical timing requirements.
- **Static Timing Analyzer**: Provides a complete pin-to-pin timing report of the design, including detailed internal path analysis
- **In-System Programming Support:** EZTag download software supports programming of Xilinx CPLDs anywhere in a JTAG chain. The download cable is included with the software.
- **XC9500 CAE Interfaces:** Includes Viewlogic and OrCAD interfaces and libraries for the PC platform and Viewlogic, Synopsys VHDL and Verilog interfaces and libraries for the Sun and HP platforms.

Schematic and VHDL Design Entry

Xilinx provides an open design environment allowing designers to choose from a variety of schematic entry, VHDL synthesizer and simulation tools such as those from OrCAD, Viewlogic, Mentor Graphics, Exemplar and Synopsys. When combined with the appropriate library and interface software, XACT-CPLD version 6 provides a complete environment for the processing of Xilinx CPLD designs.

ABEL-HDL Entry

XABEL owners can embed macros in their schematics or enter complete chip designs for Xilinx CPLDs, then use XACT-CPLD v6 to complete the design implementation. The new Xilinx XABEL-CPLD package also includes this core software. XABEL-CPLD is a complete ABEL-HDL based design tool designed specifically for text-only CPLD design.

Embedded Third-Party Compilers

The Xilinx fitter technology is licensed to many third-party development tool vendors, providing the flexibility and versatility of industry-standard design software environments with the speed, density and routability of Xilinx CPLDs. The fitter is fully integrated into the Data I/O Synario, Logical Devices CUPL, and Isdata LOGiC environments. For price and availability of the XACT-CPLD v6 based fitter, contact the development tool manufacturer. ◆

Product Availability and Pricing

The XACT-CPLD v6 core software is available *now* for immediate delivery.

Part Number	Version	Platform	Pricing
DS-560-PC1-C	v6	PC	$295
DS-560-SN2-C	v6	Sun	$495
DS-560-HP7-C	v6	HP	$495

New XACTstep™ Release Adds XC4000E and XC9500 Family Support

The new versions of XACT*step* (versions 6.0.1 for PCs and 5.2.1 for workstations) will begin shipping in June. This new release will add production software support for the new XC4000E FPGA and XC9500 CPLD programmable logic families, enabling designers to take full advantage of these latest component offerings.

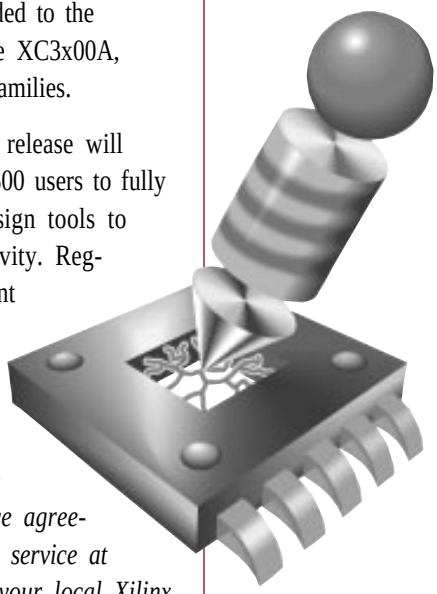
XACT*step* v5.2/v6, introduced in November 1995, combined a number of new graphical design tools to shorten software learning curves, simplify design flows, increase design performance, and speed design debug:

- **Design Manager and Flow Engine** — These easy-to-learn and use, Windows-based graphical user interface tools simplify the design process for Xilinx programmable logic devices. The Design Manager provides a hierarchical project management environment, automatically managing design files and providing version control. The Flow Engine walks users through each step in the design implementation process. Designers can use fully automatic design flows, or customize the flow by setting breakpoints at different stages of the process to allow detailed design analysis or optimization. In XACT*step* v5.2.1/v6.0.1, Design Manager support for both the XC4000E and XC9500 families is added.
- **High Performance Floorplanner™** — This comprehensive, Windows- and UNIX-based graphical floorplanning tool makes it easy for users to maximize a design's performance and density. With this tool, users can interactively place structured design elements and graphically plan their data flow, then pass their design knowledge to the automatic placement and routing tools. It also allows optimization of the XC4000 architecture's unique high-speed distributed RAM and three-state internal bus features. In XACT*step* v5.2.1/v6.0.1, Floorplanner support for the XC4000E family is added to the existing support for the XC3x00A, XC4000 and XC5200 families.
- **Hardware Debugger** — This Windows-based tool frees designers from the time-consuming task of writing exhaustive simulation vectors by allowing them to view internal signals while the FPGA device is running in-circuit. An unlimited number of internal nodes can be displayed in a waveform window. In XACT*step* v5.2.1/v6.0.1, support for the XC4000E family is added to the existing support for the XC3x00A, XC4000 and XC5200 families.

Highlights of XACT*step* v5.2.1/v6.0.1 Update

- Design Manager support for both the XC4000E and XC9500 families.
- Floorplanner support for the XC4000E family.
- Hardware Debugger support for the XC4000E family.

The new v5.2.1/v6.0.1 release will allow XC4000E and XC9500 users to fully leverage these proven design tools to maximize design productivity. Registered Xilinx development system owners with an active software maintenance agreement will receive this update automatically. *To check on the status of your maintenance agreement, call Xilinx customer service at 408-559-7778 or contact your local Xilinx sales office.* ◆



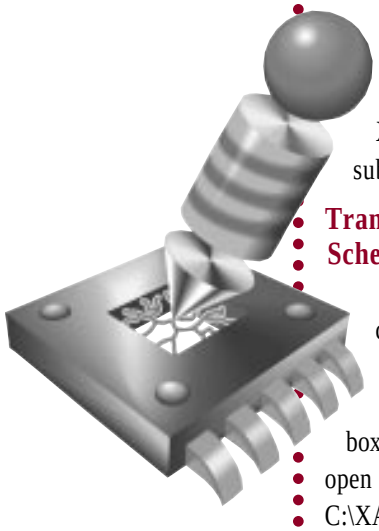
Using OrCAD Capture and Simulate

The OrCAD interface software provided by Xilinx supports the SDT 386+ schematic editor and VST 386+ simulator. The new Windows-based OrCAD tools, Capture and Simulate, are not directly supported. This article outlines the recommended flow for interfacing Capture and Simulate with the XACT*step* tools.

More information on the design flow is also available from OrCAD (*OrCAD Application Note #12, Using OrCAD Capture and Simulate with Xilinx XACT XDM or XACTstep*, and the Simulate for Windows on-line help topic: Xilinx).

USING CAPTURE

For Xilinx projects, keep each DSN file in a separate directory. Each project directory will contain the Capture DSN file, the INF netlist files, the XNF files used as input to the XACT*step* tools, and the XPROJECT\ subdirectory.



Translating Xilinx Schematic Libraries

OrCAD Capture can automatically convert the schematic libraries shipped by Xilinx. Simply choose **File➡Open➡Library**. In the dialog box, list files of type **SDT Library** and open the desired library (for example, C:\XACT\XC4000\XC4000.LIB). Capture will open another dialog box, asking where you would like to save the translated library. Save the library in the same directory, and do not change the library name. Changing the library name will result in an invalid INF netlist. In the above example, the library should be saved as C:\XACT\XC4000\XC4000.OLB.

The Xilinx Unified Libraries can also be downloaded from the Xilinx Technical BBS at 408-559-9327 (filename XOLB.ZIP). These libraries already have the default Xilinx part properties added (see the *Adding Xilinx Attributes* section).

Adding Xilinx Attributes

You can use part properties in Capture schematics to add Xilinx attributes to your design. Unlike SDT, Capture stores default part properties directly in the Xilinx library or your design. You can introduce the default Xilinx part properties (described in Chapter 11 of the *XACTstep OrCAD Interface/Tutorial Guide*) into the Xilinx libraries by using the **Tools➡Import/Export Properties** commands (described in Chapter 16 of the *OrCAD Capture for Windows User's Guide*), or download the libraries from the BBS.

To add a Xilinx attribute to a part:

1. Double-click on the part.
2. In the **Edit Part** dialog box, click **User Properties**.
3. In the **User Properties** dialog box, select the property you wish to edit (for example, OPTIONS_1 in an XC4000 design).
4. Enter the attribute in the value field (for example, loc=p11) and press <ENTER>.
5. When you are done editing properties, click OK to return to the **Edit Part** dialog box, then click OK to return to the schematic.

with XACT*step*™ version 6

USING CAPTURE (CONTINUED)

Unlike SDT, Capture allows properties to be placed on hierarchical blocks (formerly known as sheet symbols). Because the Xilinx flow uses the old INF netlist format, properties on hierarchical blocks are not supported.

For Xilinx attributes to be properly written into the INF netlist, select **Options➡Design Properties** when the **Design Manager** window is active. Select the **SDT Compatibility** tab, and enter the property names that will map into the INF netlist in any order. You do not have to include property names that are not used in the design (for example, if you only use the LOC,OPTIONS property in your XC3000 design, you do not have to enter the BASE, INIT, and other property names in the SDT Compatibility tab). When converting an SDT design, Capture automatically updates the SDT Compatibility table to reflect the property names listed in the SDT.CFG file.

Migrating Between Xilinx Families

Capture uses a design cache to store the library components that are used in each design. To migrate an existing design from one Xilinx architecture to another, the components in its cache must be replaced. In the Design Manager window, double-click on the **Design Cache** folder to open it (if there is no **Design Cache**, select **View➡Logical** to switch to logical view). Click on the first part in the cache, then select **Design➡Replace Cache**. In the dialog box, edit the **Part Library** field or use the Browse button to select the new Xilinx library. Repeat this procedure with each part in the cache.

Generating an XNF File

Make sure that the **Design Manager** window is the active window. From the menus, select **Tools➡Update Part References**. In the dialog box, click OK. Then select **File➡Save** to save the updated design.

Select **Tools➡Create Netlist**. In the dialog box, select the **VST** tab. You can change the name of the top level INF file, but do not change the destination directory.

Open an MS-DOS session and CD into the project directory. Execute the following command:

```
> sdt2xnf <design>.inf -p
 <parttype>
```

where <design> is the name of the top-level design and <parttype> is the target Xilinx device (for example, 3120APC84-2)

Exit the DOS session, then open the XACT*step* Design Manager. Select **File➡New Project**. In the **New Project** dialog box, click on the **Browse** button to select an input design. List files of type XNF, and select <design>.XNF as the input file. Click OK. Select the target family and click **Translate**. In the **Translate** dialog box, click OK.

Known Issues

Unlike SDT, Capture writes a configured library into an INF netlist only if parts from that library are used in the schematic. If the top level schematic contains no parts from a “standard” Xilinx library (XC2000, XC3000, XC4000, XC5200 or XC7000 library), an invalid INF file will be created. For example, if the top level

Continued on the next page

schematic contains only hierarchical blocks, XBLOX parts, and/or user library parts, SDT2XNF will issue the following error:

```
DS35-SDT-ERROR-033:
No standard Xilinx XC2000/
XC3000/XC4000 SDT libraries
were used in the INF file
design.inf'.
```

To prevent this error, place a part from the appropriate standard Xilinx library on the top level schematic. If the part is left unconnected, it will be trimmed by the Xilinx tools.

USING SIMULATE

For functional simulation, create an XFF file using the **Design➡Translate** command in the XACT*step* Design Manager or an XNFBA.XNF file by checking the Flow Engine **Setup➡Options➡Produce Timing Simulation Data** box. For timing simulation, the XNFBA.XNF file must be used.

Create a new simulation project. In the **Edit Simulate Project** dialog box, do not add any netlists to the project. Click **OK**.

Select **Tools➡Convert XNF to VHDL**. In the dialog box, edit the **XNF Input File** field or use the **Browse** button to select the input file. Specify the name of the output VHDL file and top level entity, and select whether you are performing a functional or timing simulation. Click OK.

Known Issues

The following is a list of problems that have been encountered while performing Xilinx simulations with Simulate v6.0. These issues have been addressed in Simulate v6.10.

- XFF files containing CLBMAP symbols cannot be simulated. To perform functional simulation on these designs, create an XNFBA.XNF file and select **Functional Simulation** in the **Convert XNF to VHDL** dialog box.
- There are no models for NAND gates. These models have been added, and a new XVHDL.AUX file is available on the OrCAD BBS (503-671-9401) or on the OrCAD web site (www.orcad.com/tbbs/SIMULATE/LIBRARY/ORCAD). Look for the file called xvhdl.zip.
- The VHDL model for the T pin of the BUFT component has incorrect polarity in some of the libraries. This affects simulation of the following components: BUFT, BUFT4, BUFT8, BUFT16, BUFE, BUFE4, BUFE8, BUFE16.
- The VHDL model of the FDPE component initializes the flip-flop output to 0 instead of 1.
- Some XBLOX™ designs containing falling-edge-triggered flip-flops cannot be simulated using timing information. Only functional simulation is possible on these designs. ◆

Running Xilinx Foundation Series Software on a Network

To save local hard disk space and allow users to access software from several PCs, the new Xilinx Foundation Series software can be installed onto a network server. A minimal set of files must be installed on each client PC, and each client PC requires a hardware key.

The installation procedure described in this article applies to the Xilinx Foundation Series CD only. It does not apply to the XACT*step* v5.2/6.0 CD or the Esperan Master Class CD.

Note: Throughout the procedure, C:\ refers to a local hard drive, D:\ refers to a CD-ROM drive, and N:\ refers to the network drive being used as the server.

Installing on the Server

The server installation can be performed from any PC that is on the network.

1. Insert the Xilinx Foundation Series CD into the local CD-ROM drive.
2. From Windows, select **FileRun**.
3. In the Command Line field, type D:\INSTALL.EXE -A.
4. The installation program will prompt you to specify the drive and directory for the server installation. Enter N:\ACTIVE and click NEXT>>.

Installing on the Client

1. From Windows, select **FileRun**.
2. In the Command Line field, type N:\ACTIVE\SETUP\SETUP.EXE.
3. Select the installation type. Choices are:

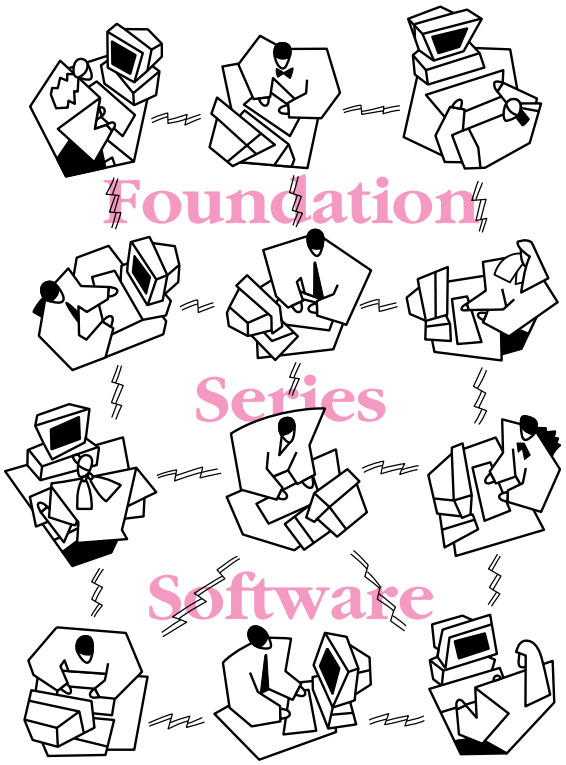
   Minimum - Installs required Windows files. Creates empty PROJECT\ and LIBDIR\ directories for user projects and user-created libraries.

   Typical - Installs required Windows files, executables, and sample projects. Xilinx libraries and documentation remain on the server.

   Copy all/Custom - Installs any combination of files. Use this option for non-network installs. The Foundation software will not search for any files on the server.

   Click NEXT>>.

4. The install prompts you to specify the installation directory. Enter C:\ACTIVE and click NEXT>>.
5. Fill out the User Name and Company fields and click NEXT>>.
6. If you selected Copy all/Custom, the install brings up a dialog box that allows you to select the files that you want to install. When you are finished selecting files, click NEXT>>.



When the installation is complete, restarting Windows is recommended. Exit all other applications, check the Restart Windows checkbox, and click FINISH. ◆

Using Viewlogic's Workview Office

Workview Office is the latest design package from Viewlogic Systems, and runs under the Windows 95 operating system. Because it was released prior to Workview Office, XACT*step* v6 was not designed to be fully-compatible with this new Viewlogic software. However, with a small amount of effort — as described below — the two development systems can be successfully used in concert to implement Xilinx FPGA and CPLD designs.

Here are the procedures to follow to use Workview Office with XACT*step* v6:

Installation Procedures

1. The Xilinx DS391 package (the libraries and netlist translaters for use with Viewlogic tools) must be installed into the Workview Office directory. The default for the install program is to locate it in the C:\PROSER directory.

2. Workview.msg isn't installed by the Xilinx tools and is no longer included with Viewlogic tools. The following error message is displayed when this file is missing:

```
Couldn't find the message file workview.msg
in the local or WDIR directories.
```

To resolve this problem, copy the workview.msg file from the PROSER\STANDARD directory on the XACT*step* 5.2/6.0 CD-ROM to the WVOFFICE\STANDARD directory on your system. This will eliminate this error.

Figure 1

Set-Up Procedures

1. The libraries used for schematic entry and simulation must be set up properly. In the Viewlogic Project Manager, under the **Library Search Order** dialog, is a button to add **FPGA Libs**. Using this button will produce a set of libraries like this:

```
dir [p] C:\temp
dir [r] c:\wvoffice;C:\XACT\unified\xc4000
(xc4000)
dir [r] c:\wvoffice;C:\XACT\unified\xblox
(xblox)
dir [r] c:\wvoffice;C:\XACT\unified\builtin
(builtin)
dir [r] c:\wvoffice;C:\XACT\unified\xbuiltin
(xbuiltin)
```

The path names to the libraries must be corrected. In order to avoid having to edit each line for every new project, use the following work-around:

A. In the autoexec.bat, add this line:

```
set xactlibs=C:\wvoffice (or wherever the unified
 directory is located)
```

B. Edit the file libs.lst in the WVOffice\standard directory. Change the line,

```
#define Xilinx XACT
```

to

```
#define Xilinx XACTLIBS
```

Then restart Windows 95. This will point the FPGA Libs to the unified directory. The libraries then should appear as shown in Figure 1. Any new families can be add by editing the libs.lst file.

The primary alias needs to be added. In the Library search order, add this library:

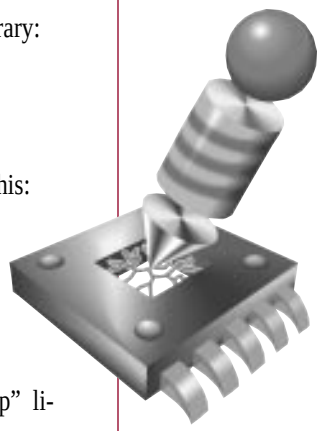
```
dir [w] . (primary)
```

See **Figure 2** for an example of how this should appear.

After this step is done, the libraries in the viewdraw.ini file should look like this:

```
dir [p] C:\temp
dir [w] . (primary)
dir [rm] C:\wvoffice\unified\xc4000 (xc4000)
dir [rm] c:\wvoffice\unified\xblox (xblox)
dir [rm] c:\wvoffice\unified\builtin (builtin)
dir [rm] c:\wvoffice\unified\xbuiltin (xbuiltin)
```

Be sure to add components from the ". (primary)" library and not the "C:\temp" library to make sure that they have the primary alias.



2. Invalid keywords in viewdraw.ini file can produce errors, such as:

```
C:\temp\viewdraw.ini
Line 58: WINDOW_BACKGROUND 15 0 0
Invalid keyword 'WINDOW_BACKGROUND'
```

These errors can cause the tools, such as XSIMMAKE, to fail. Replace the viewdraw.ini file with the one from the XACT*step* CD-ROM in PROSER\STANDARD, and update the libraries by resaving the project file.

3. XSIMMAKE expects vsm.exe to be a DOS program. However, in Workview Office, the Windows GUI version of vsm has been named vsm.exe and the DOS version is vsm_ngui.exe. XSIMMAKE will report that vsm failed. Change the names on these two executables so that the DOS version of vsm is now named vsm.exe and the Windows version is called vsm_gui.exe. In the Viewdraw Tools menu, select the customize item and then pick the **Common Menu Button**. Edit the **Create Digital Netlist** so the command now points to vsm_gui.exe.

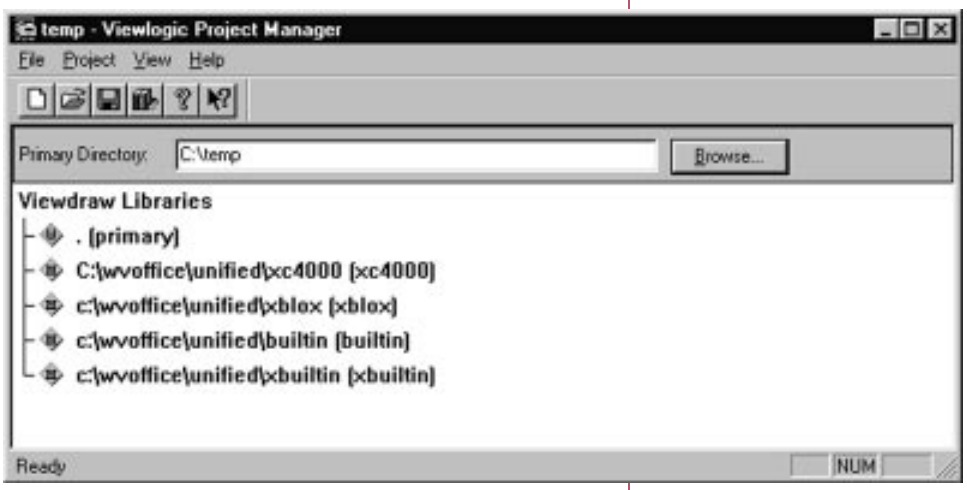


Figure 2

4. If the license file is set to expire in 30 days or less, the warning issued in vsm will cause XSIMMAKE to fail. To solve this, contact Viewlogic Systems to have your license file updated. ◆

HDL Synthesis *and* Built-In Clock Enables

The internal flip-flops in Xilinx FPGA architectures have built-in, dedicated clock enable (CE) inputs. Appropriate use of these clock enables avoids the need for gating clocks, facilitating good synchronous design techniques. Using these dedicated clock enable resources also avoids having to use the combinatorial logic resources in the logic blocks to implement the same functionality, potentially eliminating an extra level of logic from the design. This, in turn, can minimize delays along critical paths and save valuable logic resources.

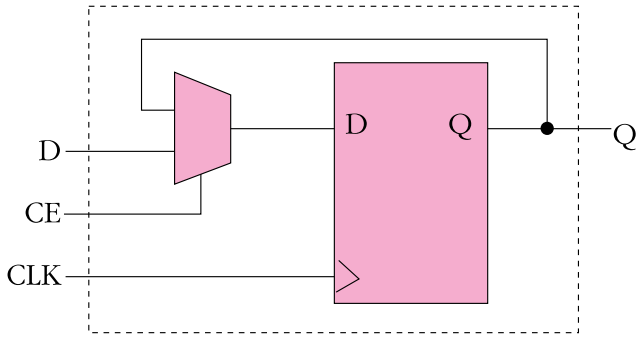


Figure 1

The built-in clock enable function is implemented using a multiplexer in front of the flip-flop's data input, as shown in **Figure 1**. When the clock enable signal is not asserted, the Q output of the flip-flop is fed to the D input, holding the flip-flop in its current state regardless of activity on the data and clock inputs. Implementing clock enables in this manner avoids the race conditions that could result if the clock enable line was used to directly gate the clock input. Implementing the same clock enable functionality using a look-up table (LUT) within a configurable logic block (CLB) would consume three inputs to the LUT; in most cases, this would add an additional LUT to the data path to the flip-flop as well as using additional routing resources.

However, to take advantage of the built-in clock enable function, users of HDLs and logic synthesis tools need to be very careful when coding flip-flops. Different coding techniques can yield significantly different circuit implementations. The choice of user options for the synthesis compiler also can affect the synthesis results.

Using an HDL, there are a many ways to describe a flip-flop with a clock

CODING EXAMPLE 1

```
always @ (posedge clk or posedge rst)
begin
     if (rst)
           q <= 1'b0;
     else if (clken)
           q <= d;
end
```

CODING EXAMPLE 2

```
always @ (posedge clk or posedge rst)
begin
     if (rst)
           q <= 1'b0;
     else if (clken)
           q <= d;
     else
           q <= q;
end
```

"...to take advantage of the built-in clock enable function, users of HDLs and logic sysnthesis tools need to be very careful when coding flip-flops."

enable. The examples show four unique ways of describing a flip-flop with an asynchronous reset and a synchronous clock enable. *(The information presented below also applies if the asynchronous reset is removed from the description.)* While Verilog is used in the examples, the discussion applies to VHDL coding as well.

These different coding styles can yield differing results, as shown in **Table 1**. Furthermore, user-selected options in the synthesis compiler also can affect the results. For example, in the case of the Synopsys FPGA Compiler synthesis tool, two compilation variables affect flip-flop implementations; these variables control feedback paths in sequential circuits. The variables and their default values are:

```
hdlin_keep_feedback "FALSE"
hdlin_keep_inv_feedback "TRUE"
```

The hdlin_keep_feedback variable does not seem to affect the implementation of flip-flops for Xilinx FPGAs, but the hdlin_keep_inv_feedback variable does have a significant effect. (Incidentally, if you type "help hdlin_keep_inv_feedback" at the dc_shell prompt, it informs you that the default value for this variable is FALSE. This is not accurate; the default value is TRUE. This was changed with release 3.2b, and has been TRUE ever since.)

Table 1 summarizes the results of synthesizing code fragment examples 1-4, in terms of whether the built-in clock enable or a multiplexer circuit external to the flip-flop is used to implement the clock enable function. The results came from using v3.3b of the Synopsys compiler. *Other logic synthesis compilers may yield different results. If this information cannot be obtained from the documentation supplied with your synthesis tools, than you may want to use the HDL code examples given here to test the operation of your synthesis tool.*

Continued on the next page

CODING EXAMPLE 3

```
assign d_in = clken ? d : q;
always @ (posedge clk or posedge rst)
 begin
    if (rst)
          q <= 1'b0;
     else
          q <= d_in;
 end
```

CODING EXAMPLE 4

```
assign d_in = ((clken & d) | (~clken & q));
always @ (posedge clk or posedge rst)
begin
     if (rst)
          q <= 1'b0;
    else
          q <= d_in;
```


Table 1: Synthesis Results Using Synopys FPGA Compiler

hdlin_keep_inv_feedback variable	register size	Example (1)	(2)	(3)	(4)
False	single-bit	CE	CE	CE	MUX
False	multi-bit	CE	CE	CE	MUX
True	single-bit	CE	CE	CE	MUX
True	multi-bit	CE	MUX	MUX	MUX

CE = dedicated clock enable synthesized MUX = external multiplexer synthesized

Example (4) uses an explicit multiplexer equation that is external to the flip-flop code. This will **always** cause the compiler to generate a flip-flop with an external multiplexer, and will not use the dedicated clock enable flip-flop.

When FALSE, the hdlin_keep_inv_feedback variable will **always** cause the compiler to generate a flip-flop with a dedicated clock enable for **examples (1-3)**.

When TRUE (the default), the hdlin_keep_inv_feedback variable causes the compiler to generate circuits that are dependent on both the design and the coding style, as follows:

- If the register is a single-bit entity, the compiler will generate a flip-flop with a dedicated clock enable when using the coding styles of Examples (1-3). As stated above, an external multiplexer will be always be generated for example (4).
- If the register is part of a multi-bit bus (or vector), the synthesis results depend on how the code is written. Example (1) will generate a set of flip-flops with dedicated clock enable (CE) pins. Examples (2) and (3) will generate simple flip-flops with a multiplexer driving the D input. Basically, if you include the "else ➠ q gets q" clause in the code, the compiler interprets this (correctly) as a multiplexer, and will generate the multiplexer externally using feedback from the flip-flop. If you don't, the compiler will use flip-flops with dedicated clock enables.

Which is better? That depends upon the application. Using flip-flops with the dedicated clock enables is most efficient for highest speed and minimal area, and is usually recommended. As noted above, implementing an external multiplexer requires three inputs to a look-up table to generate the multiplexer. In an FPGA's fan-in limited architecture, this wastes resources and often causes an additional LUT delay.

However, there are occasions when an external multiplexer is better than a dedicated clock enable (CE) for placement reasons. Flip-flops within a single configurable logic block (CLB) share a common dedicated CE signal. If only the dedicated CE mechanism is used, there is no physical way to place two flip-flops with different clock enable signals in the same CLB. If the design has many unique, dedicated clock enables, placement problems may result, because once a flip-flop with a dedicated CE is placed in a CLB, no other flip-flop with a dedicated CE can be placed in that same CLB, possibly resulting in "wasted," unusable flip-flops. Flip-flops that do not use the dedicated CE lines have no such restrictions. This phenomenon is more prevalent in the XC5200 family, with four flip-flops per CLB, than in the XC4000 series, with only two flip-flops per CLB.

(A similar placement problem can occur with flip-flops having different clock or asynchronous control signals, since these inputs are also common to all the flip-flops in a CLB. However, most designs do not have enough unique clocks or resets to make this a problem.)

In summary, using coding styles that take advantage of the FPGA's built-in clock enable function usually results in smaller, faster designs. However, placement considerations may dictate the use of clock enable logic implemented in the CLBs look-up tables. In either case, it is important for the designer to understand which circuit implementation will be produced by the synthesis compiler, which is often a function of the user's coding style. ◆

"...it is important for the designer to understand which circuit implementation will be produced by the synthesis compiler, which is often a function of the user's coding style."

Ten-Digit Fully Synchronous BCD Counter Runs at 87 MHz

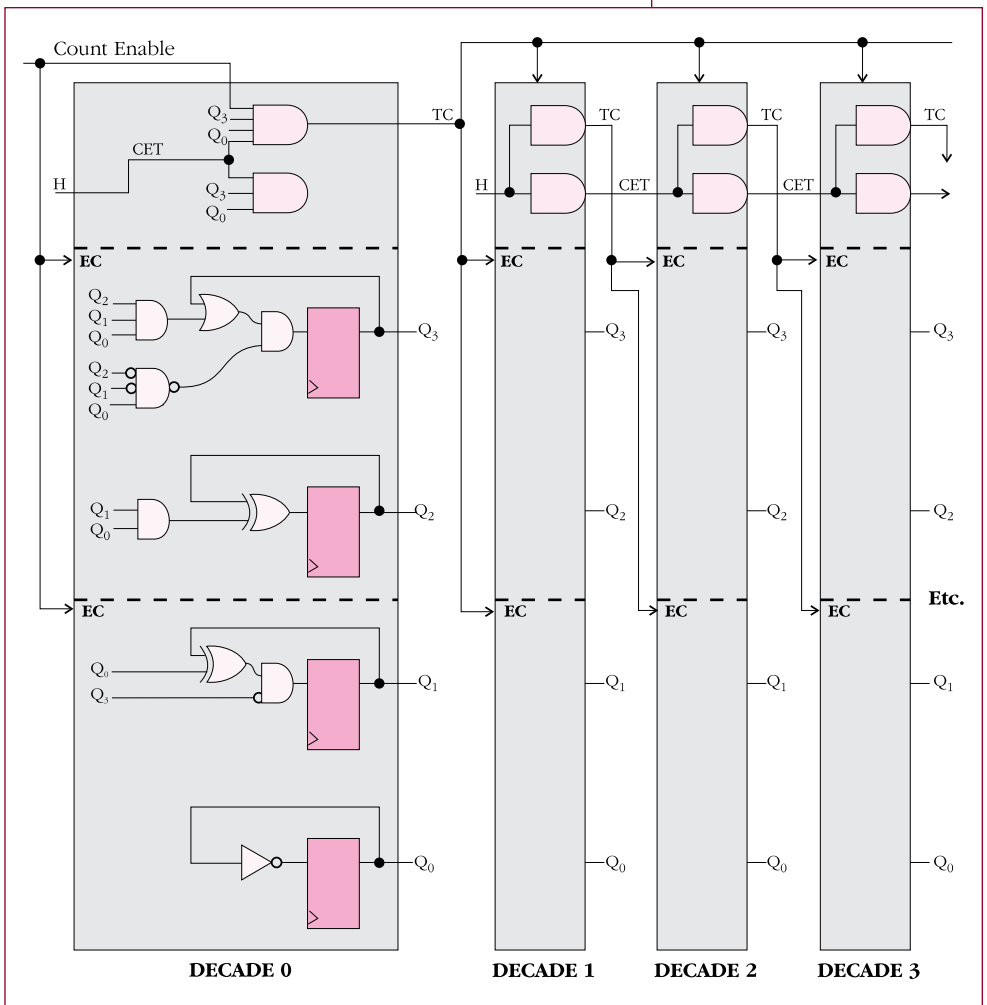
A binary-coded-decimal (BCD) counter design that operates at an impressive 87 MHz under worst-case conditions has been implemented in an XC3100A-09 FPGA device.

This synchronous BCD counter uses the count-enable-parallel/count-enable-trickle (CEP/CET) method of fast carry expansion *(as first introduced in 1969 in the Fairchild 9310, and made popular by the 74160-series of TTL-MSI counters.)* The decoded terminal count of the least-significant decade, ANDed with the incoming count enable control signal, drives the count-enable-parallel (CEP) inputs of all higher-order decades in parallel, effectively preventing them from incrementing while CEP is Low. This gives the conventional ripple-carry CET-chain nine full clock periods to settle. The count enable control input can start and stop this counter on any clock.

However, this design cannot be modified to be loadable, and even a modification to down-count results in slower speed. *(For more complex counters, the XC4000E family is a better choice.)*

The 10-digit counter occupies 29 CLBs. When floorplanned on three horizontal rows of CLBs, one BCD digit per column, using a horizontal longline for distributing CEP, this fully synchronous counter has been simulated to operate under worst-case conditions at

- A maximum of 87 MHz in an XC3130A-09 device
- A maximum of 41.5 MHz in any XC3100A-5 device
- A maximum of 40.0 MHz in any XC3000A-6 device



The design was also bench-tested in an XC3142-09 device, and ran at a clock rate of up to 146 MHz at room temperature and nominal V_{CC}. ◆

Look for the design files on the Xilinx technical Bulletin Board (applications area, filename BCDCNT.ZIP).

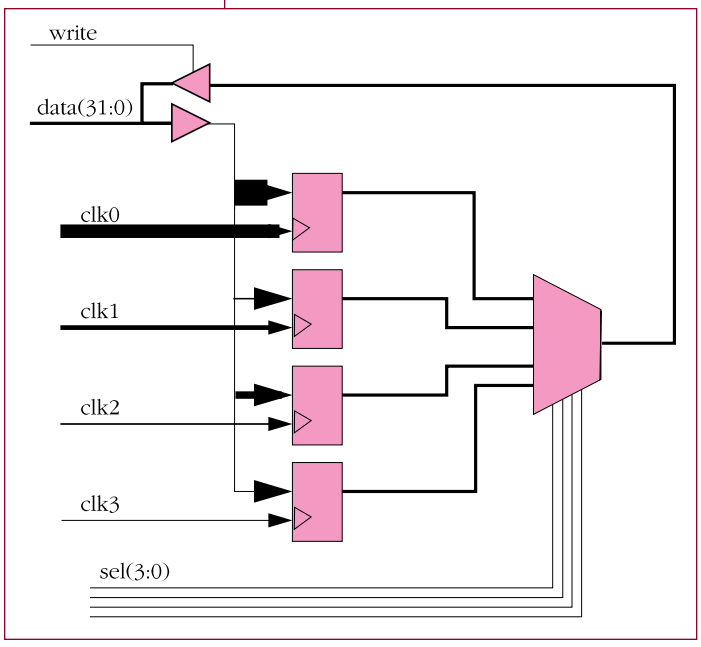
Structured Floorplanning for the XC8100:

The XC8100 architecture, design flow and development tools allow a designer to use structured floorplanning techniques to implement highly-ordered designs easily and efficiently. A highly-ordered design refers to a design that uses relatively simple functions repeated multiple times to create a larger design. Typically, this involves multiple iterations of the same operation on the individual elements of a bus or several buses. Examples include crossbar switches, FIR filters and FIFOs.

The flexible nature of the XC8100 configurable logic cell (CLC) facilitates the implementation of such highly-ordered logic. Furthermore, the XC8000 XACT*step*™ development tools preserve the original netlist, with TrueMap™ one-to-one mapping of the gates in the netlist to CLCs in the FPGA and the preservation of the netlist's symbolic names; these features are key to easily developing a floorplan of a design's repetitious structures.

Figure 1 shows an example of a design with a highly-ordered structure. This design consists of a 32-bit bi-directional bus, four clocks, four select lines, four 32-bit registers and a 32x4-bit mulitplexer.

Figure 1
High Level Design Example



The basic building block for this design can be implemented in two different ways. The first uses two sum-of-product cells and an OR cell to generate the 4-to-1 multiplexer, and uses a total of 11 CLCs per bit slice **(Figure 2).** The second uses four three-state buffers to generate the 4-to-1 multiplexer, and uses a total of 12 CLCs per bit slice **(Figure 3**). To determine which approach is better, the user must examine how the design will fit into the XC8100 FPGA architecture as a whole.

This example design has a 32-bit architecture, requiring 32 instances of this building block. Thus, the first approach would consume 11x32=352 CLCs; the second approach would use 12x32=384 CLCs. Either approach results in a design that fits into an XC8101 devices (384 CLCs total). However, by examining the cell array dimensions and some potential floorplans, the optimal approach can be determined.

The XC8101 device has an array of 16 rows and 24 columns, with a maximum of 72 I/Os using the 100-pin PQFP package. For optimal performance, the multiplexing logic should be located as close to the external pins as possible, and the source registers as close to the multiplexers as possible. The best way to achieve these goals is to split the 32-bit DATA bus so that half of the bus is on the top half of the chip and the other half is in the bottom half of the chip. The data bus lines will then be located on the top and bottom edges, and the clock and selector lines can drive from the left and right sides of the chip into the multiplexers and register arrays. A simple high-level floorplan of this implementation is illustrated in **Figure 4**.

In order to be as close as possible to the external pins, the multiplexing logic should occupy the bottom two and top

A Data Path Example

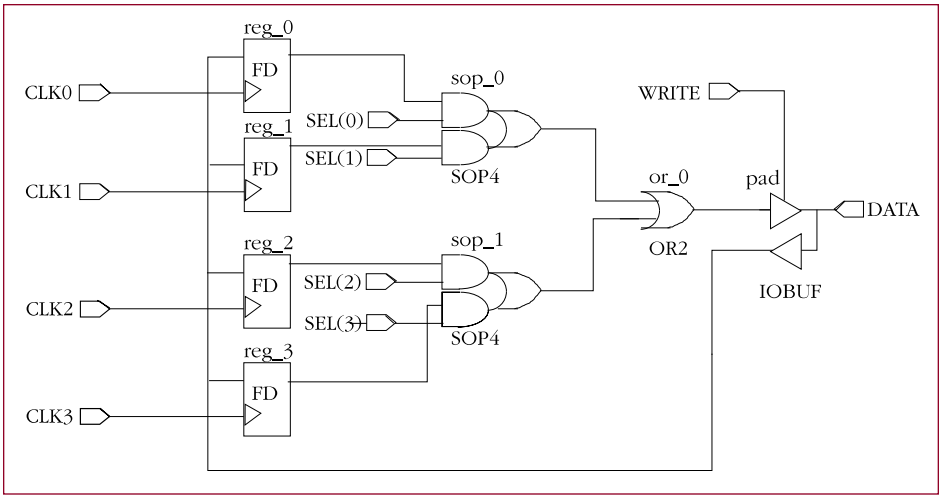


Figure 2
Data Path Building Block, Approach 1
11 CLCs

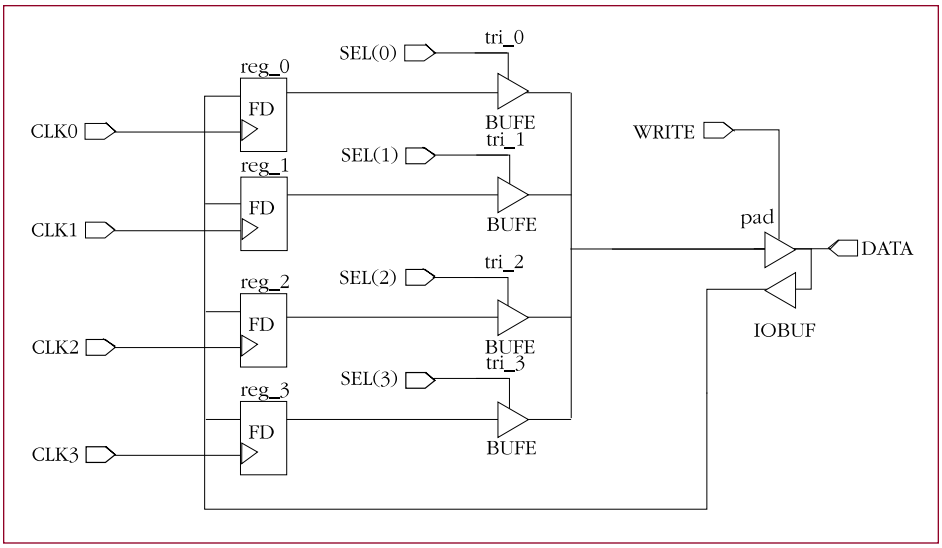


Figure 3
Data Path Building Block, Approach 2
12 CLCs

two rows of the array — a total of 48 CLCs for each half of the multiplexer logic. Using the first approach requires 16x3=48 CLCs for the multiplexers. The second approach requires 16x4=64 CLCs, overflowing the allotted area. While it would still be possible to place and route a design using the second approach, it would not be as regular, and, therefore, not as easily optimized; thus, the first approach will yield the better solution.

At this point, the XACT*step* Series 8000 software could place and route the design with few, if any, additional placement constraints. However, by giving the software absolute constraints using expert

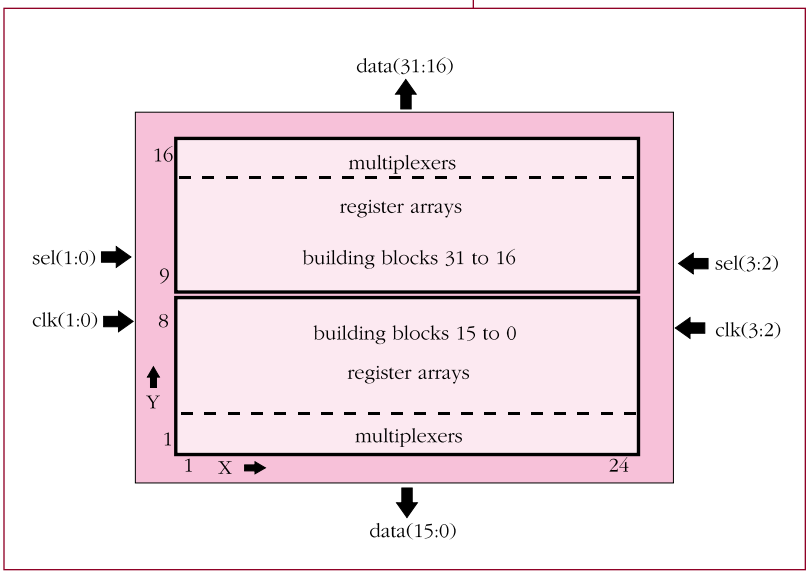


Figure 4 *Simple High-Level Floorplan*

Structured Floorplanning

Continued from the previous page

knowledge of the design's structure, a truly optimized design can be achieved. **Figures 5 and 6** show two possible floorplans for the multiplexer logic; where bb_0 refers to the DATA bus bit 0 building block, bb_1 refers to the DATA bus bit 1 building block, and sop_0, sop_1 and or_0 refer to the combinatorial logic shown in **Figure 2**. Both of these layouts work reasonably well; they are small and fit evenly into the allotted 2x24 CLC area. The second layout has a slight advantage in that the nets from the sop_0 and sop_1 instances to the or_0 instance will be of equal length; thus, this floorplan will be adopted.

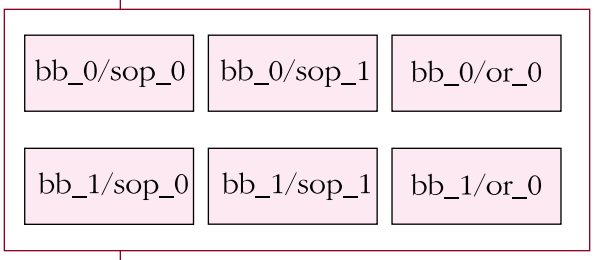


Figure 5 *Multiplexer Floorplan Layout 1, 2 bus bits*

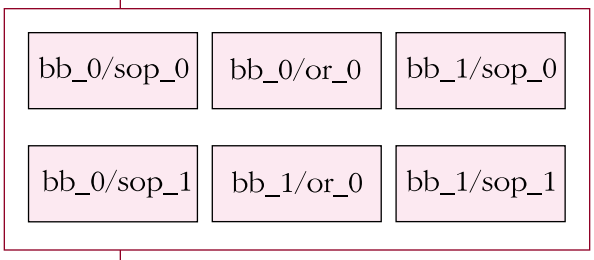


Figure 6 *Multiplexer Floorplan Layout 2, 2 bus bits*

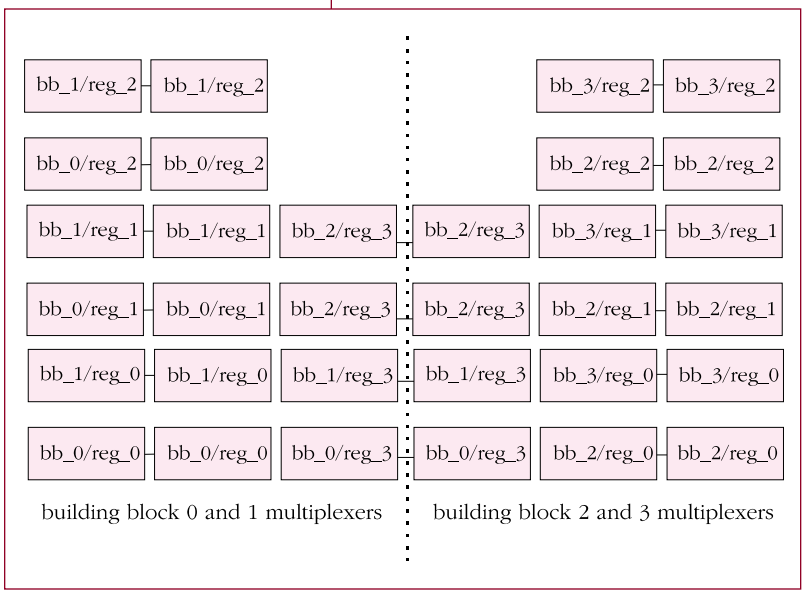


Figure 7 *Register Array Floorplan Layout (bottom half)*

Next, the register arrays can be placed in an manner that aligns them with the multiplexers. In the XC8100 architecture, a single D-type flip-flop (FD) requires two adjacent CLCs in the same row, since it uses the cascade connection between the CLCs. There are four registers, requiring eight CLCs, associated with each 4-to-1 multiplexer. Thus, a total of eight registers (16 CLCs) are associated with the pair of multiplexers located in the 2x3 cell area shown in **Figure 6.**

Considering just the bottom-half of the array, the multiplexer logic occupies the bottom two rows, leaving six rows for implementing their associated register arrays. Since the multiplexers are arranged in a configuration spanning three columns (2x3 CLCs), and each register must be constructed in a manner spanning two columns (1x2 CLCs), it is convenient to floorplan the register array in partitions of six columns, encompassing two of the 2x3 multiplexer structures (four bit-slices). Using this approach, the four building block register arrays associated with the four multiplexers will require 32 CLCs, and need to be placed within a 6x6 CLC area. A typically layout for this is implementation is shown in **Figure 7.** This layout keeps the registers that share a common clock in the same set of rows, with the exception of the registers that share clk3, which are split between two sets of rows.

Since the symbolic names for the signals and instances are in a predetermined numeric order, and the development tools preserve these names, the constraint file needed to implement this basic floorplan for the whole design can be generated by a short C program *(see page 39)*; this program could just as easily have been written in awk, Perl, BASIC or the constraints could be directly written using a text editor.

The final step in the process is the assignment of the clock signals to high drive resources. This design uses four

clocks, each of which can be allocated to a BUFGP input pin. If the BUFGP input pins were unavailable for some reason, using a regular IBUF and the internal global buffer BUFGS would work as well.

The BUFROW high-drive resources also can be used in this design to speed up the selector lines. If the selector inputs were used to directly drive the inputs of the 32 SOPs on both the top and bottom rows of the chip, the delay could be quite large. A better approach uses two sets of selector inputs, one for the top and one for the bottom. The selector inputs would now drive 16 SOPs; each can be driven a single BUFROW buffer to further reduce delay and skew. With four BUFROW resources available per double row, this is a perfect match; two selector lines can drive from the left side of the chip and two from the right side of the chip for both the top and bottom selectors, as shown in **Figure 8.**

In a similar manner, many highly-structured designs can be easily optimized by planning ahead and giving some thought to the back-end process. These techniques work equally well for schematic- or HDL-based design methodologies. Simple floorplans can help to select the most suitable architectural approach. Naming the instances in an orderly and predetermined fashion (for example, using FOR loops and/or instantiation in HDL or instance name attributes in schematics) makes it possible for constraints to be created using simple programs. Careful floorplanning also allows a designer to take full advantage of the XC8100 FPGA's flexible high-drive buffers. Together, these techniques can produce an elegant, compact and high-performance solution. ◆

C CODE FOR BASIC FLOORPLAN

```
main()
{
int i,x;
/* constrain the lower building block
 multiplexers in blocks of 2*/
for (i=0;i<16;i++){
x=i/2*3;
printf("constrain bb_%d/sop_0 x%dy%d\n",i,x,2);
printf("constrain bb_%d/sop_1 x%dy%d\n",i,x,1);
printf("constrain bb_%d/or_0 x%dy%d\n",i,x+1,2);
i++;
printf("constrain bb_%d/sop_0 x%dy%d\n",i,x+2,2);
printf("constrain bb_%d/sop_1 x%dy%d\n",i,x+2,1);
printf("constrain bb_%d/or_0 x%dy%d\n",i,x+1,1);
}
/* constrain the lower building block register
 arrays in blocks of 4*/
for(i=0;i<16;i++){
x=i/4*6;
printf("constrain bb_%d/reg_0 x%dy%d\n",i,x,3);
printf("constrain bb_%d/reg_1 x%dy%d\n",i,x,5);
printf("constrain bb_%d/reg_2 x%dy%d\n",i,x,7);
printf("constrain bb_%d/reg_3 x%dy%d\n",i,x+2,3);
i++;
printf("constrain bb_%d/reg_0 x%dy%d\n",i,x,4);
printf("constrain bb_%d/reg_1 x%dy%d\n",i,x,6);
printf("constrain bb_%d/reg_2 x%dy%d\n",i,x,8);
printf("constrain bb_%d/reg_3 x%dy%d\n",i,x+2,4);
i++;
printf("constrain bb_%d/reg_0 x%dy%d\n",i,x+4,3);
printf("constrain bb_%d/reg_1 x%dy%d\n",i,x+4,5);
printf("constrain bb_%d/reg_2 x%dy%d\n",i,x+4,7);
printf("constrain bb_%d/reg_3 x%dy%d\n",i,x+2,5);
i++;
printf("constrain bb_%d/reg_0 x%dy%d\n",i,x+4,4);
printf("constrain bb_%d/reg_1 x%dy%d\n",i,x+4,6);
printf("constrain bb_%d/reg_2 x%dy%d\n",i,x+4,8);
printf("constrain bb_%d/reg_3 x%dy%d\n",i,x+2,6);
}
/** The top half is left as an exersise for the
 reader **/
}
```

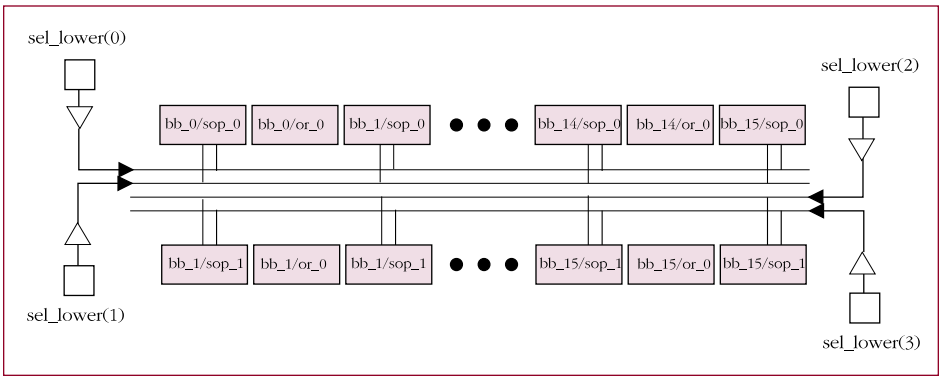


Figure 8
Lower Building Block Selectors with BUFROWs

A Look at "Minimum" Delays

Why They're So Elusive to Specify and How to Estimate Them

All of the timing parameters reported by the XACT*step* timing calculator (for example, when using the static timing analyzer) are worst-case delays that take into account process, temperature, and voltage variations.

However, in order to complete a true "worst-case" analysis of hold times at the system level (e.g., between interconnected devices on a board), CPLD and FPGA users often ask for minimum or "best-case" timing.

"In most cases, users do not have to concern themselves with "best-case" delays. Internal to the CPLD or FPGA device, we guarantee that minimum delays will never cause hold-time problems."

In defining product specifications, we try to balance user needs with our requirement to publish honest specifications that are feasible to test and can be guaranteed for years to come. Thus, like most IC manufacturers, Xilinx does not provide minimum or "best-case" timing parameters. Unfortunately, minimum delays are not easily tested. Today's CMOS devices are very fast, and, even if fast enough testers were readily available and the test times were affordable, the minimum numbers would change every time fabrication processes are changed, particularly when moving to finer-grained geometries. Thus, best-case timing parameters would be impossible to guarantee over the typical product life of an IC component.

This is further complicated by an industry practice known as "down-binning", which involves shipping a fast device against an order for a slower part. For example, "-2" speed grade devices might be marked as slower "-3" parts in order to fill an order for -3 devices. More typically, a device will get tested against the speed grade needed to fulfill a given order, even though it might qualify as a faster device had it been tested against the faster specification.

In most cases, users do not have to concern themselves with "best-case" delays. Internal to the CPLD or FPGA device, we guarantee that minimum delays will never cause hold-time problems. For chip-to-chip interconnections, good synchronous design practices alleviate potential hold-time violations, particularly if hold time requirements for incoming signals are not positive.

If two devices are directly interconnected and share a common clock without any skew, then any positive hold-time requirement on an input can only be satisfied by a guaranteed minimum clock-to-out delay on the output that drives that input. Thus, positive hold-time requirements on data inputs are very undesirable. Xilinx IC designers have gone to great lengths to guarantee zero hold time requirements for input registers in our CPLD and FPGA products. For example, the XC4000 and XC5200 series FPGAs feature an optional delay element in the input path that increases the data set-up time so that the pin-to-pin hold-time requirement on the input is never positive.

However, what if you are driving a device with a positive hold time requirement from an FPGA output? What minimum clock-to-out delay can be "guaranteed" for the FPGA? Without on-chip phase-locked-loops, there can never be a zero ns clock-to-output delay. The laws of physics are on your side.

In CMOS technology, all delays decrease when the temperature is lowered and when the supply voltage is increased. Therefore, to ensure operation under

worst-case conditions, our devices are tested at a high temperature (85° C junction temperature) and a low supply voltage (4.75 V for 5 V commercial parts).

Estimating Best Case Delays

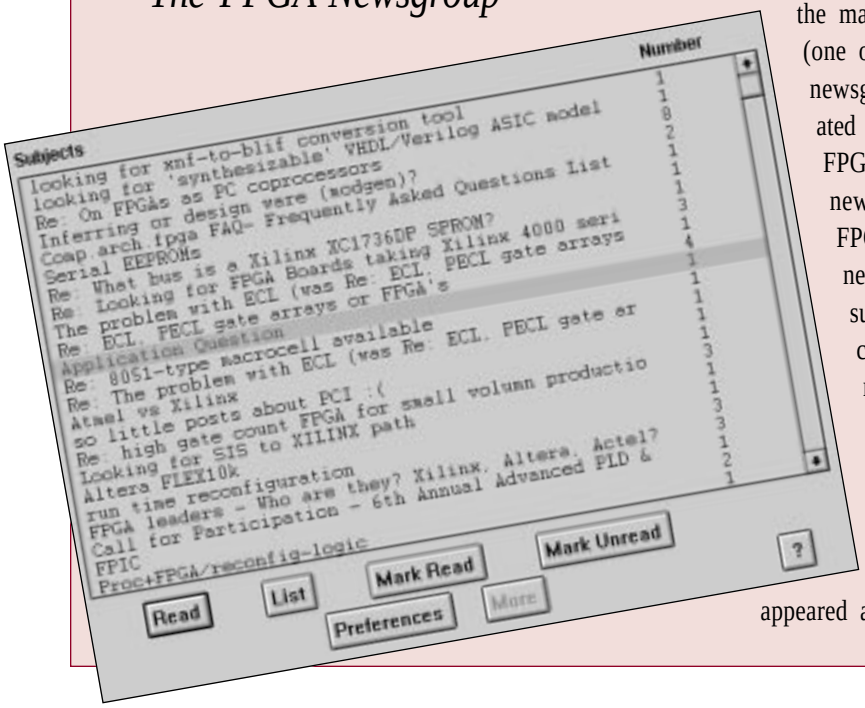
How short can the "best case" delay be when compared to the guaranteed and tested "worst-case" parameters? As an estimate, let's first subtract 10% for tester guardband (devices are always tested to slightly tighter parameters than specified, in order to avoid disagreements over tester calibration. Ten percent is probably very conservative, but 5% would be aggressive.) Then let's subtract 10% for the difference between the 4.75 V test voltage and the 5.25 V best-case supply voltage. Next, we'll subtract 30% for the difference between the 85° C test and the 0° C best-case junction temperature. Finally, we must subtract 40% for the difference between our slowest processing and fastest processing.

Multiplying 0.9 x 0.9 x 0.7 x 0.6 yields 0.34. That means, **you can expect to get a "best-case" delay of about a third of the specified worst-case value** for commercial grade products.

To be very conservative, for any given parameter we suggest that you assume a best-case value of 25% of the worst-case number that we specify for the same parameter *at the fastest available speed grade*. Thus, for the top-of-the-line, fastest part, the ratio between worst- and best-case delay is conservatively estimated as 4:1; for slower parts it is a larger ratio.

However, rather than relying on this estimate, the best advice is to design synchronously, whenever possible, and use devices with non-positive hold time requirements on data inputs. ◆

Visit comp.arch.fpga — The FPGA Newsgroup



Xilinx users with Internet access should review the material in the comp.arch.fpga newsgroup (one of more than 10,000 unmoderated newsgroups on the Internet!). Originally created as a forum for sharing ideas on using FPGAs for new computer architectures, this newsgroup has expanded to discuss all FPGA-related issues. It is a well-mannered newsgroup that covers a wide variety of subjects. Xilinx sometimes "takes it on the chin," as do our competitors, but the newsgroup can be helpful in clarifying confusing issues and tapping into other engineers' experience.

The discussion of minimum timing delays in the article on the page at left is a summation of material that first appeared as a "thread" in this newsgroup. ◆

Advanced Carry-Logic Techniques

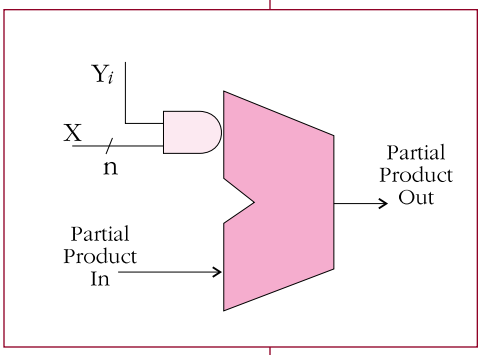
All XC4000 series FPGAs provide dedicated carry generation logic within configurable logic blocks (CLBs), and dedicated routing to propagate carry signals between CLBs. Most basic carry-logic applications, like adders and counters, are supported by library macros. However, the carry logic can be used in other ways for more specialized circuits.

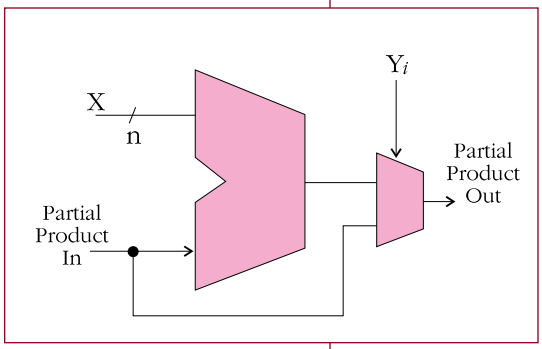
MULTIPLICATION

Perhaps the simplest adaptation of the carry logic is to provide a multiplier function. In a multiplier, one of the input words, X, is ANDed separately with each bit of the other input word, Y, and the resulting words are summed with appropriate binary weighting. Consequently, there is a need for gated adders, such as shown in **Figure 1(a).**

However, it is inefficient to implement this function as drawn. The carry logic connects directly to the input pins of the CLB, and there is no provision to dynamically gate the carry logic inputs. Instead of using an additional CLB for the gating, the function can be modified as shown in **Figure 1(b).** This circuit is functionally equivalent to the one of Figure 1(a), provided that all inputs are gated with the same signal, and the carry signal is not used directly.

Figure 1(b) can be implemented easily, as shown in **Figure 2**. Each bit of a conventional adder is modified such that the unconditional sum is created as an internal node in the function generator. This sum is multiplexed with the partial sum to the adder, which is already available within the function generator. The gating signal, Y_i, controls the multiplexer, and is brought in on a spare pin.





Figures 1(a), top left, & **1(b)**, bottom left, *Two Versions of a Gated Adder*

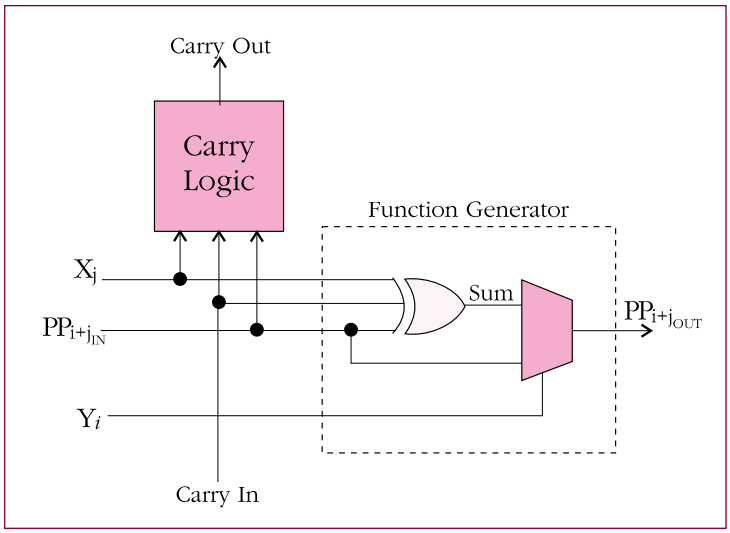


Figure 2
Implementation of a Gated Adder

2'S COMPLEMENT AND ABSOLUTE VALUE

The strategy for generating an absolute value is similar to that used in multiplication. The 2's complement of the input is generated unconditionally as an internal signal in the function generator. The sign of the input value is then used to select between the input directly or the 2's complement. Thus, the output is always positive.

The 2's complement function is best implemented by decrementing the input and inverting the result. This alternative to the traditional invert-and-add-one technique gives exactly the same result, but avoids having to modify data at the input to the carry logic.

Figure 3 shows how one bit of a standard decrementer is modified to provide the absolute value function. The inversion in front of the XOR gate is a part of the decrement, while the inversion at the output completes the generation of the 2's complement.

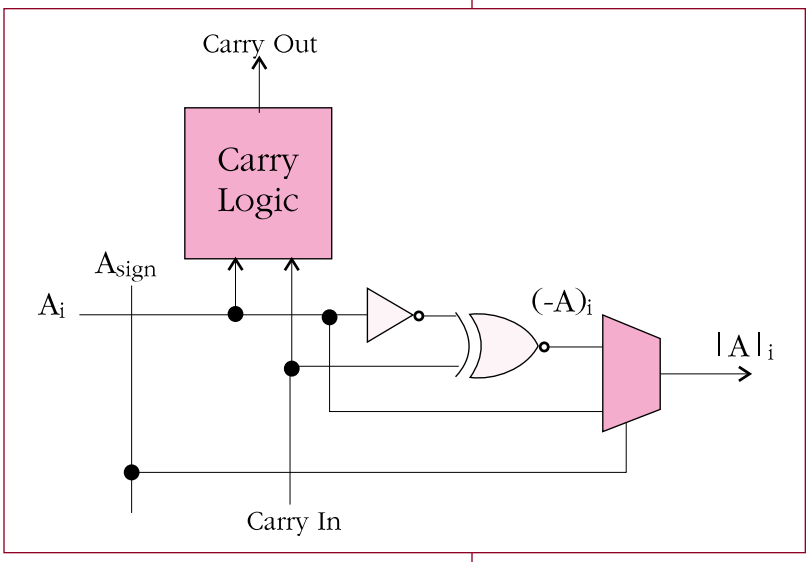


Figure 3
Absolute Value Generator

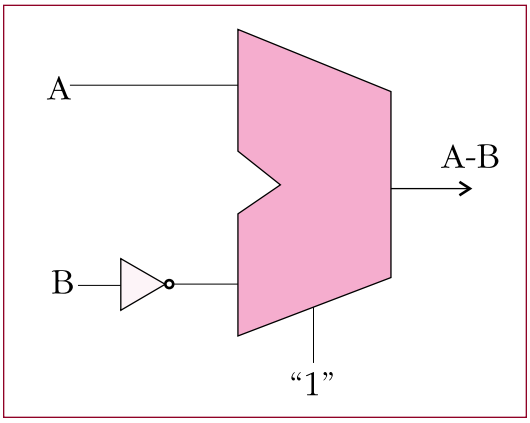
ABSOLUTE DIFFERENCE

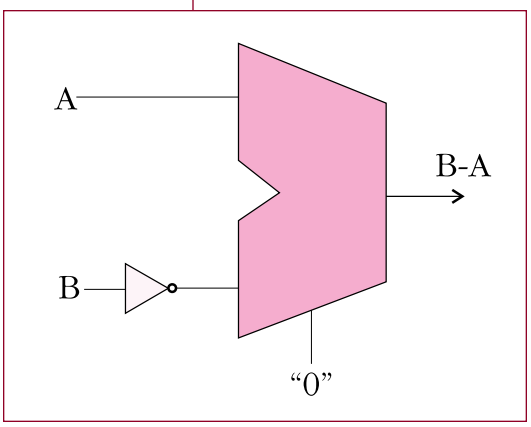
Where time allows, the absolute value of a difference, |A–B|, can be calculated using a single carry chain. This is essentially a two-stage operation, and can be completed in two successive clock periods or in the two phases of a single clock period.

The technique depends upon two alternative methods of subtracting. Traditionally, B is inverted at the input of an adder, and the carry is asserted to give the result A–B, as in **Figure 4(a)**. However, if both B and the output of the adder are inverted, and the incoming carry is not asserted, the result is B–A, as in **Figure 4(b)**.

The standard subtractor used with XC4000 carry logic is of the traditional variety. A configuration bit causes the B input to the carry logic to be inverted, and this cannot be changed dynamically. However, this inversion is common to both methods of subtraction described above. The carry input to the adder can easily be made a dynamic input, and there is space in the function generator to add an XOR gate that inverts the output

Continued on the next page





Figures 4(a), left, & **4(b)**
Two Ways to Subtract

Continued from the previous page

when needed. Thus, it is possible to generate either A–B or B–A on demand.

Given this capability, the absolute difference is obtained by simply choosing the function that yields a positive result. However, directly feeding back the sign of the output to select the function will result in instability, and a flip-flop must be added to eliminate this possibility (**Figure 5**).

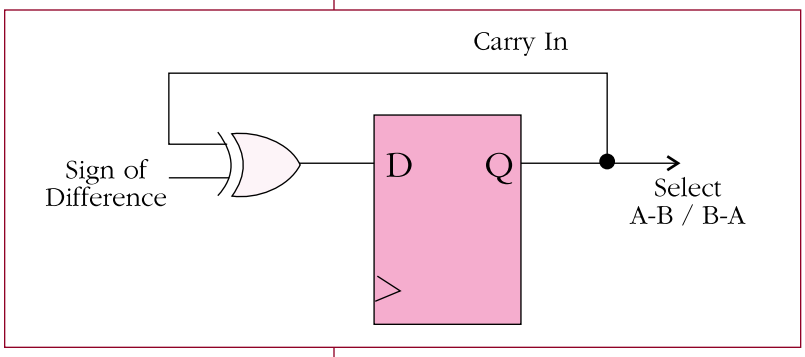


Figure 5
Function Select Flip-Flop

In the first of two operations on the same inputs, either subtraction can be performed. If the first result is negative, the flip-flop is toggled to select the other function to achieve a positive result in the second operation. If the first result is positive, the flip-flop is not toggled, and the first operation is repeated. In either case, the result of the second operation is positive.

PEAK DETECTOR

In a peak detector, the current peak value is stored in a register, and is subtracted from every new input value. If the difference is positive, the new input is larger, and it replaces the value in the register as a new peak. Otherwise, the register is unchanged.

Only the sign of the difference is of interest in determining whether the register is updated or not. The other difference bits need not be generated, and the corresponding function generators are free to be used in controlling the register.

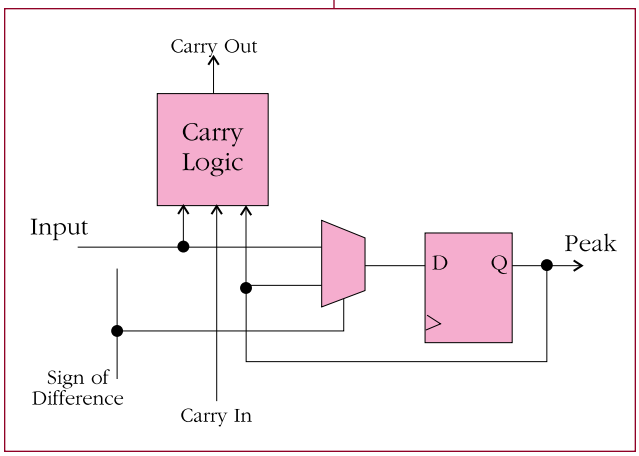
Figure 6(a) shows a typical bit. The sign of the difference is routed to all bits to select the value loaded into the register. This operation includes the sign bit of the register. Consequently, the subtraction must be sign-extended by one bit so that the sign of the difference is also available.

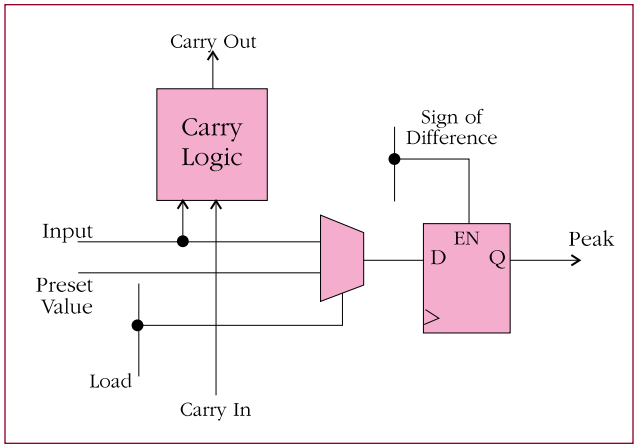
The peak detector can be reset by forcing the sign output to a one. This causes the current input value to be loaded as a new peak, regardless of the value of the previous peak.

The circuit described above is equivalent to using the sign of the difference to enable the peak register, and the CLB Enable Clock pin could be used equally well. However, the two techniques impose different routing constraints, and either may be more effective than the other in different situations.

If enable clock is used, the function generators can be used for other purposes. For example, to initialize the peak detector with a predetermined value that is only changed by a peak that exceeds it, as in **Figure 6(b)**. ◆

Figures 6(a), left, & **6(b)**
Two Peak Detectors





PCI-Based Reconfigurable Computers

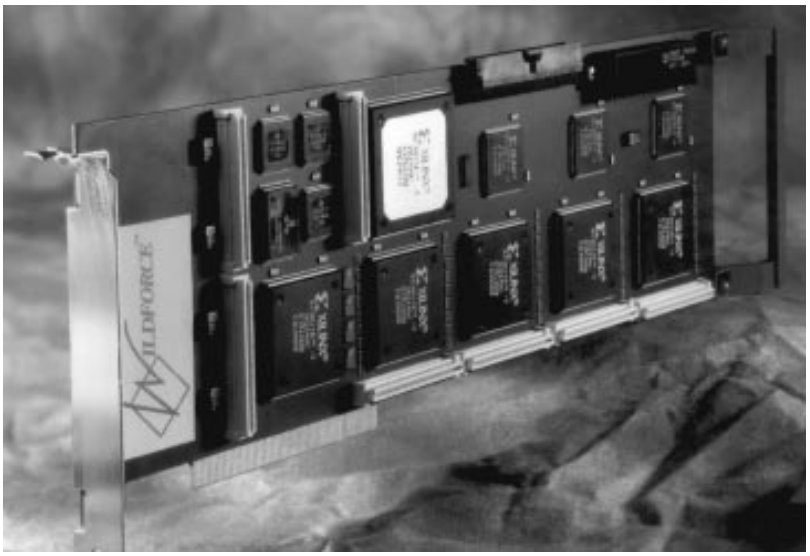
The Reconfigurable Computing Developer's Program presents the "Company of the Quarter" award to Annapolis Micro Systems, Inc. (Annapolis, Maryland), developer of the first commercially available, PCI-based reconfigurable computing board.

Annapolis Micro Systems, founded in 1982, has a strong background in hardware design, ASICs, system drivers and operating systems. The 33-person company provides custom electronic product design services, including expert Xilinx design services, to commercial and government employees. It has completed more than 400 Xilinx-based designs.

Annapolis has started moving away from contract work (although they still have a large ASIC design business) to focus on their reconfigurable WILDFIRE™ systems and design tools (based on the SPLASH technology developed by the Supercomputing Research Center and licensed by the National Security Agency). Using XC4000E series FPGAs, the WILDFIRE family turns a PC into a supercomputer by unleashing the power of reconfigurable computing.

WILDFIRE systems have been used to test complex algorithms, emulate ASIC designs, model computer architectures, and perform rapid prototyping for image processing, DSP, communications, text search, compression/decompression, sequence analysis, and pattern matching applications. By downloading algorithms directly into FPGAs, processing speeds far exceed those possible with standard, Von Neumann architectures. In a recent test, a particular DSP application on one WILDFIRE board outperformed a Cray YMP supercomputer by a factor of 15.

The WILDFIRE family now includes both VME and PCI systems. The original WILDFIRE system is based on a VME board with 16 parallel processing elements (PE). Each PE is composed of an XC4010E, XC4013E, or XC4020E FPGA and 512 Kbytes of high-speed memory.



Another XC4000E device implements the crossbar connections between the PEs.

As many as 16 WILDFIRE boards can fit in a single WILDFIRE VME chassis for greater capacities. Other configurations include:

- WILDCHILD — identical to WILDFIRE, but with eight PEs.
- WILDFORCE — a standard-size PCI card with four PEs, a user-programmable crossbar and provisions for add-on capabilities.
- WILD-ONE — a half-size PCI card with one PE, with provisions for add-on capabilities.

The WILDFORCE and WILD-ONE boards can be populated with XC4013E, XC4020E, or XC4025E FPGAs.

All of these systems share a common architecture, system controller, debugger and run-time libraries. They are "programmed" using industry-standard C and VHDL tools. The reconfigurable processors can support SIMD (single-instruction-multiple-data), MIMD (multiple-instruction-multiple-data) and systolic computing operations. ◆

MEMBER PCI LOCAL BUS SPECIAL INTEREST GROUP

To learn more about WILDFIRE, please contact Annapolis Micro Systems, Inc. at 410-841-2514 or Annapmicro@aol.com.

For more information about the Xilinx Reconfigurable Computing Developer's Program, visit WebLINX at www.xilinx.com/programs/reconfig.htm or call John Watson at 408-879-6584.

CPLDs

Q When using XABEL-CPLD™, how do I specify fast slew rates for Xilinx CPLDs?

By default, the slew rate is SLOW for all pins. The FAST attribute is used to selectively control the slew rate on a pin-by-pin basis for any output signal. In XABEL-CPLD, use the following syntax:

```
XEPLD PROPERTY 'FAST ON
 signal_list';
```

If you omit the signal name list, the FAST property applies to all pins. If you include a signal name list, the listed signals are given the specified setting and all other signals are given the opposite setting.

Q When using schematic capture, how do I specify fast slew rates for Xilinx CPLDs?

To assign individual pins in a schematic to use the faster slew rate, attach a FAST attribute to the output pad.

Q When using the Xilinx Synopsys Interface (XSI), how do I specify fast slew rates for Xilinx CPLDs?

The Synopsys HIGH attribute translates to a SLOW Xilinx slew rate, and the NONE attribute translates to a FAST Xilinx slew rate. As with the other design entry methods, the XSI default for outputs is SLOW. To set all outputs for FAST slew rate, use the following syntax:

```
set_pad_type -slewrate NONE
 all_outputs ()
```

Use this command after specifying the set_port_is_pad command and before implementing the insert_pads command. You can set any individual output for fast signal transition by using the following syntax:

```
set_pad_type -slewrate NONE
 port_name
```

Mentor Graphics

Q Running Convert Design to retarget designs *(as described in XCELL 20, page 41)* loses NET, LOC, and other properties associated with schematic ports and PADs in my design. What's happening?

This is due to a change in Design Architect that occured between Mentor A.1 and A.4. A fix is available by E-mailing xdocs@xilinx.com with "send 822" in the subject line. The problem also has been fixed in XACT 5.2.1.

Q While running Quicksim II under Solaris, I get the following errors on all of my RAMs and ROMs:

```
/MEMORY/sp_ram32': Problems
 loading binary '$LCA/gen_lib/
 sp_ram32/sp_ram32.ss5_b'
Could not load object file
 "/tools/ds344/gen_lib/sp_ram32/
 sp_ram32.ss5_b"
No such file or directory
```

What does this mean and how is it fixed?

Unlike standard QuickPart models, Xilinx RAMs and ROMs are described using "behavioral language models" (BLMs), that have as their core a binary executable file. This file is named differently for each platform: .sss_b for SunOS, .ss5_b for Solaris, etc. Thus, under Solaris, Quicksim expects to find a .ss5_b file where none exists, since Solaris-native versions of the RAM and ROM models are not part of released XACT 5.x software. However, an unofficial RAM and ROM patch is available on the Xilinx BBS:

```
Category: Software Help
Subcategory: Mentor
Filename: SOLARMEN.ZIP
```

This is a compressed TAR file that needs to be extracted in the $LCA directory.

Note: Currently, Xilinx does not officially support the Solaris operating system. This is an unofficial patch and, as such, has not been fully tested.

Q While running Fncsim8 under Solaris, Gen_sch8 or XBLXGS generates this error message:

```
crt1:bad [02] open: /tools/
 mentor/lib/mgc_ld.so
Abort - core dumped
```

What does this mean and how can I fix it?

Gen_sch8 and XBLXGS are incompatible with Solaris, so their use should be avoided under this operating system. This usually involves avoiding XBLOX or XABEL components on schematics. If the design contains RAM or ROM components, obtain the library patch described in the answer to the previous question.

Q I encountered the following error during EDIF2XNF:

```
Error: 3 port name I1 not
found on external library
primitive for cell OR2
```

What does this mean and how can I fix it?

This error usually occurs from mixing Xilinx libraries (*i.e.*, obsolete and Unified) or having an incorrect library setting in EDIF2XNF. For more information, E-mail xdocs@xilinx.com with "send 396" in the subject line.

Q Will XACT*step* v5.2 work with Mentor Graphics' B.1 release?

Xilinx strongly recommends using the Mentor A.x release with XACT*step* v5.2, as Xilinx has not tested Mentor B.1 and will not officially support this product until the Merged Release. However, testing by Mentor Graphics found that the implementation flow and most of the simulation flow should work properly. Gen_sch8 and XBLXGS were found not to work properly due to problems associated with dynamic linking to Mentor's Design Data Port (DDP). If you must use Mentor B.1, avoid using XBLOX or XABEL components in your designs, if possible, to eliminate the need to run Gen_sch8 or XBLXGS.

Synopsys

Q How can I initialize my XC4000E RAM module?

The contents of an XC4000E RAM at power up may be specified (initialized) by the user. When RAM modules are instantiated directly in the HDL source code, initialization values for the RAM may be entered using the following command:

```
set_attribute
  "instance_name" xnf_init
  "init_value" type string
```

Replace "instance_name" with the actual instance name of the RAM module that has been instantiated. For 16-location RAMs, specify a 4-digit hexadecimal value for "init_value". For 32-location RAMs, specify an 8-digit hexadecimal value. *NOTE: Although this mechanism permits RAM-initialization information to be carried into the FPGA implementation tools for incorporation into the configuration bitstream, it does not affect behavioral simulation. For behavioral simulation, a RAM's contents remain unknown until they are defined by valid write accesses. However, backannotated functional or timing simulation will reflect this RAM initialization information.*

Q What should my .synopsys_dc.setup and .synopsys_vss.setup files contain to synthesize and simulate XC4000E designs?

.synopsys_dc.setup: The .synopsys_dc.setup file for XC4000E synthesis should be identical to that for XC4000 designs, with the exception of the target- and link-library settings and the reference to the XBLOX DesignWare Library. An example .synopsys_dc.setup file is shown below. The target- and link-library settings were created using the commmand: synlibs 4005e-3

```
search_path = { . \
<XC4000E_DS401_install path>/
  synopsys/libraries/syn \
<Synopsys_install_path>/
  libraries/syn}
define_design_lib xblox_4000e
  -path \
<DS401_install_path>/
  synopsys/libraries/dw/lib/
  fpga/xc4000e
compile_fix_multiple_port_nets
  = true
xlnx_hier_blknm = 1
xnfout_library_version =
  "2.0.0"
bus_naming_style = "%s<%d>"
bus_dimension_separator_style
  = "><"
bus_inference_style =
  "%s<%d>"
link_library = {xprim_4005e-
  3.db xprim_4000e-3.db
  xgen_4000e.db \
xfpga_4000e-3.db xio_4000e-
  3.db}
target_library =
  {xprim_4005e-3.db
  xprim_4000e-3.db
  xgen_4000e.db \
xfpga_4000e-3.db xio_4000e-
  3.db}
symbol_library =
  {xc4000e.sdb}
synthetic_library =
  {xblox_4000e.sldb
  standard.sldb}
```

The reference to the XC4000E XBLOX DesignWare library is differentiated from the XC4000 XBLOX library with a new name: xblox_4000e.

.synopsys_vss.setup: The .synopsys_vss.setup file for XC4000E simulation should also be identical to that for XC4000 designs with the exception of the simulation-library reference. An example .synopsys_vss.setup file is shown below.

```
timebase=ns
time_res_factor=0.1
no_hazard_mesg=true
WORK > DEFAULT
DEFAULT : ./WORK
xc4000e:<XC4000E_DS401_install_path>/
  synopsys/libraries/sim/lib/
  xc4000e
```


HOTLINE SUPPORT

United States

Customer Support Hotline:
800-255-7778
Hrs: 8:00 a.m.-5:00 p.m. Pacific

Customer Support Fax Number:
408-879-4442
Avail: 24 hrs/day-7 days/week

E-mail Address:
hotline@xilinx.com

Customer Service*:
408-559-7778
ask for customer service
**Call for software updates, authorization codes, documentation updates, etc.*

Europe

UK, London Office
telephone: (44) 1932 349402
fax: (44) 1932 333530
BBS: (44) 1932 333540
e-mail: ukhelp@xilinx.com

France, Paris Office
telephone: (33) 1 3463 0100
fax: (33) 1 3463 0109
e-mail: frhelp@xilinx.com

Germany, Munich Office
telephone: (49) 89 99 15 49 30
fax: (49) 89 904 4748
e-mail: dlhelp@xilinx.com

X-TALK
NETWORK OF ELECTRONIC SERVICES

Xilinx World Wide Web Site
www.xilinx.com

Electronic Tech. Bulletin Board:
408-559-9327
Avail: 24 hrs/day-7 days/week

XDOCS e-mail document server
for instructions, send an e-mail to xdocs@xilinx.com with "help" as only item in the subject header.

XFACTS fax document server -
Call 1-408-879-4400.

Specific Question E-mail:
Digital Signal Processing ..... dsp@xilinx.com
PCI-bus ..... pci@xilinx.com
Plug and Play ISA ..... PnP@xilinx.com
PCMCIA card ..... pcmcia@xilinx.com
Async. Transfer Mode ..... atm@xilinx.com
Reconfig. Computing .. reconfig@xilinx.com

Corporate Headquarters
Xilinx, Inc.
2100 Logic Drive
San Jose, CA 95124
Tel: 408-559-7778
Fax: 408-559-7114

Europe
Xilinx, Ltd.
Suite 1B, Cobb House
Oyster Lane
Byfleet
Surrey KT147DU
United Kingdom
Tel: 44-1-932-349401
Fax: 44-1-932-349499

Japan
Xilinx, KK
Daini-Nagaoka Bldg. 2F
2-8-5, Hatchobori,
Chuo-ku, Tokyo 104
Japan
Tel: 81-3-3297-9191
Fax: 81-3-3297-9189

Asia Pacific
Xilinx Asia Pacific
Unit 4312, Tower II
Metroplaza
Hing Fong Road
Kwai Fong, N.T.
Hong Kong
Tel: 852-2424-5200
Fax: 852-2494-7159

FAX in Your Comments and Suggestions

To: Brad Fawcett, *XCELL* Editor **Xilinx Inc. FAX**: 408-879-4676

From: _____ Date: _____

❑ **Please add my name to the XCELL mailing list.**

NAME _____

COMPANY _____

ADDRESS _____

CITY/STATE/ZIP _____

PHONE _____

❑ **I'm interested in having my company's design featured in a future edition of XCELL as a Customer Feature.**

Comments and Suggestions: _____

Please use this form to FAX in your comments and suggestions, or to request an addition to the XCELL mailing list. Please feel free to make copies of this form for your colleagues.

XILINX®
2100 Logic Drive
San Jose, CA 95124-3450

U.S. Postage
PAID
First Class
Permit No. 2196
San Jose, CA