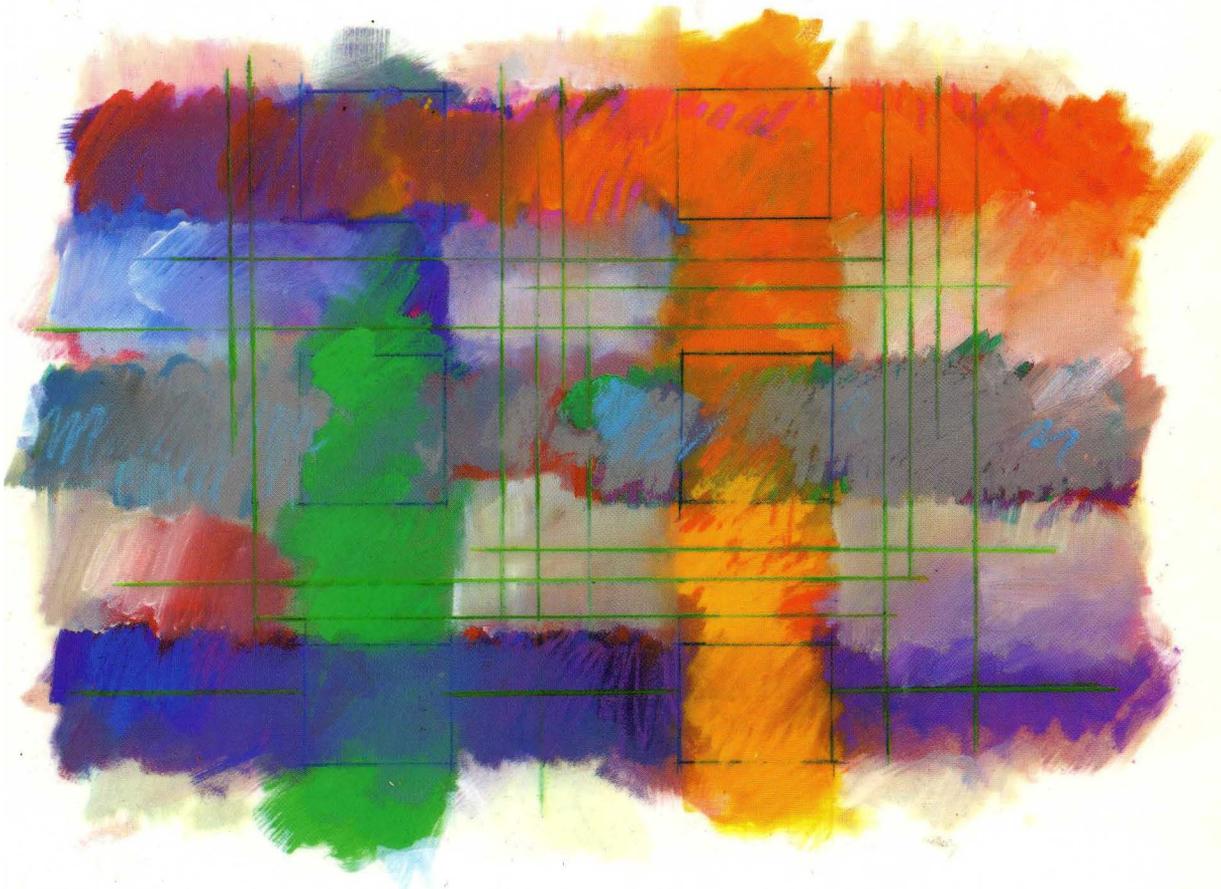




# The Programmable Gate Array Data Book





## **SECTION TITLES**

---

**1 Programmable Gate Arrays**

**2 Product Specifications**

**3 Quality, Testing, Packaging**

**4 Technical Support**

**5 Development Systems**

**6 Applications**

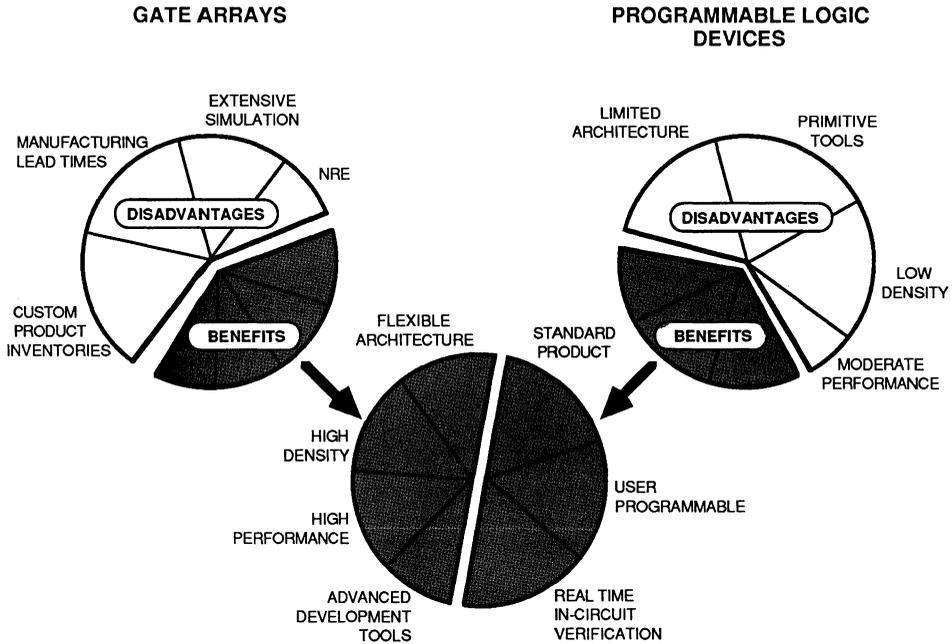
**7 Article Reprints**

**8 Index**

---



The Programmable Gate Array Company



1162 01

THE PROGRAMMABLE GATE ARRAY  
(LOGIC CELL™ ARRAY)

**1 Programmable Gate Arrays**

About the Company	1-1
Introduction to Programmable Gate Arrays	1-3
A Cost of Ownership Comparison	1-9

**2 Product Specifications**

XC3000 Logic Cell Array Family	2-1
XC2064, XC2018	2-55
XC2064, XC2018 Military	2-97
XC1736 Serial Configuration PROM	2-113

**3 Quality, Testing, and Packaging**

Quality Assurance and Reliability	3-1
Testing Methodology	3-11
Packaging	3-17

**4 Technical Support**

Technical Seminars and Users' Group Meetings	4-1
Video Tapes	4-2
Newsletter	4-3
Technical Bulletin Board	4-4
Field Application Engineers	4-6
User's Guide	4-7
XACT Manuals	4-8

**5 Development Systems**

Hardware Requirements	5-1
Development System Overview	5-4
Design Flow	5-9
LCA Macro Library Listings	5-25
Product Briefs	5-32
Ordering Information	5-47

---

**6 Applications**

Introduction	6-1
Estimating Size and Performance	6-3
Counting Gates	6-7
Incorporating PLD Equations into LCAs	6-9
The XC2000 User's Guide to the XC3000 Family	6-13
Additional Electrical Parameters	6-14
LCA Performance	6-16
Metastable Recovery	6-19
Majority Logic, Parity	6-20
Multiple Address Decoding	6-21
Adders and Comparators	6-22
Building Latches Out of Logic	6-25
Synchronous Counters, Fast and Compact	6-26
40 MHz Presettable Counter	6-27
Frequency Counter	6-29
Serial Code Conversion	6-30
Serial Pattern Detectors	6-32
8-Bit Format Converter	6-33
Megabit FIFO in Two Chips	6-35
State Machine Design	6-37
Self-Diagnosing Hardware	6-38
PS/2 Micro Channel Interface	6-41
High Speed Bar Code Reader	6-43
DRAM Controller	6-45
Logic Analyzer/In-Circuit Emulator	6-51

**7 Article Reprints**

Building Tomorrow's Disk Controller Today	7-1
The Acid Test	7-5
Programmable Logic Betters the Odds	7-8
Using LCAs in a Satellite Earth Station	7-12
Faster Turnaround for a T1 Interface	7-17
Two, Two, Two Chips in One	7-19
LCA Stars in Video	7-22

**8 Index**

Index	8-1
Sales Office Listing	8-3

---

## **1 Programmable Gate Arrays**

**2 Product Specifications**

**3 Quality, Testing, Packaging**

**4 Technical Support**

**5 Development Systems**

**6 Applications**

**7 Article Reprints**

**8 Index**

---



# Programmable Gate Arrays

About the Company .....	1-1
Introduction to Programmable Gate Arrays .....	1-3
A Cost of Ownership Comparison .....	1-9

## Contributors

Peter Alfke	Dave Lautzenheiser
Keath Armstrong	Wes Patterson
Jim Chumbley	Richard Ravel
Chuck Erickson	Steve Schreifels
Lee Farrell	Robert Stransky
Brad Fawcett	Thomas Waugh
Dave Galli	Perry Wu
Jim Hsieh	Pardner Wynn
Steve Knapp	

© Copyright 1988 by Xilinx, Inc. All Rights Reserved.

Patents Pending

Xilinx, Logic Cell, LCA, XACT, XACTOR, Programmable Gate Array, and Logic Processor are trademarks of Xilinx, Inc. The Programmable Gate Array Company is a Service Mark of Xilinx, Inc.

IBM is a registered trademark and PC/AT, PC/XT, PS/2, and Micro Channel are trademarks of International Business Machines Corporation. ABEL is a trademark and Data I/O is a registered trademark of Data I/O Corporation. FutureNet is a registered trademark and DASH is a trademark of FutureNet Corporation, a Data I/O Company. SimuCad and Silos are registered trademarks and P-Silos and P/C-Silos are trademarks of SimuCad Corporation. Microsoft is a registered trademark and MS-DOS is a trademark of Microsoft Corporation. Logitech is a registered trademark of LOGITECH Inc. Lotus is a registered trademark of Lotus Development Corporation. AboveBoard and AboveBoard/PS are trademarks of Intel Corporation. RAMpage!, SixPakPlus and SixPakPremium are registered trademarks of AST Research, Inc. Mouse Systems is a trademark of Mouse Systems Corporation. Centronics is a registered trademark of Centronics Data Computer Corporation. PAL and PALASM are registered trademarks of Advanced Micro Devices, Inc., Inc. DAISY, Logician, and DNIX are registered trademarks of Daisy Systems. UNIX is a trademark of AT&T Technologies, Inc. CUPL is a trademark of Personal CAD Systems. Apollo, AEGIS and DOMAIN

are registered trademarks of APOLLO Computer. Mentor and IDEA are registered trademarks and QuickSim, NETED, EXPAND are trademarks of Mentor Graphics, Inc. ValidGED and ValidSim are trademarks of Valid Logic Systems, Inc. Sun is a registered trademark of Sun Microsystems, Inc. SCHEMA II is a trademark of Omation Corporation. OrCAD is a registered trademark of OrCAD Systems Corporation. Viewlogic is a registered trademark of Viewlogic Systems, Inc. CASE Technology is a trademark of CASE Technology, a division of Teredyne's Electronic Design Automation Group. Xilinx, Inc. does not assume any liability arising out of the application or use of any product described herein; nor does it convey any license under its patent, copyright or maskwork rights or any rights of others. Xilinx, Inc. reserves the right to make changes, at any time, in order to improve reliability, function or design and to supply the best product possible. Xilinx, Inc. cannot assume responsibility for the use of any circuitry described other than circuitry entirely embodied in their product. No other circuit patent licenses are implied. Xilinx, Inc. cannot assume responsibility for any circuits shown or represent that they are free from patent infringement or of any other third party right. Xilinx, Inc. assumes no obligation to correct any errors contained herein or to advise any user of this text of any correction if such be made.

Printed in U.S.A.



## About the Company...

---

Xilinx is the first company to develop a user-programmable gate array. The invention of a high-density, general-purpose user-programmable logic device was the result of a number of breakthroughs, and several aspects of the device's array architecture have been patented. Since the introduction of its first product in 1985, Xilinx has continued to lead in the development of new programmable gate arrays with higher speeds, higher densities, and lower costs.

Due to their density and the convenience of user programmability, Xilinx's Programmable Gate Arrays represent an important new alternative in the market for Application Specific Integrated Circuits (ASICs). The company continues to concentrate its resources exclusively on expanding its growing family of Programmable Gate Arrays and associated development systems, and on providing technical support to a rapidly growing customer base.







# INTRODUCTION TO PROGRAMMABLE GATE ARRAYS

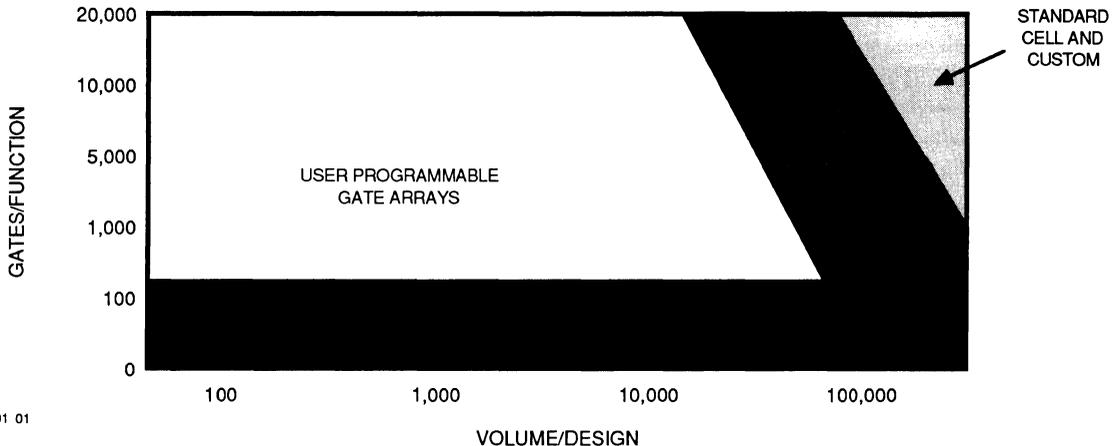
Steady advances in the level of integration in electronic circuits have improved many equipment features, reducing costs, power consumption, and system size, while increasing performance and reliability. Increasing levels of integration are most evident in microprocessor and memory ICs. With each process generation, the technology gap between these VLSI circuits and other standard logic ICs has widened. To achieve comparable densities for their proprietary logic functions, designers of digital equipment have been forced

to consider factory-programmed custom and semicustom Application Specific Integrated Circuits (ASICs).

Recent breakthroughs in logic architectures have resulted in the first high density ASICs that can be configured by the user. These user programmable gate arrays combine the logic integration benefits of custom VLSI with the design, production, and time to market advantages of standard products. The flexibility of user programming significantly reduces the risks of design changes and production rate changes.

## ASIC ALTERNATIVES

*Application Specific ICs are the best solution for most logic functions. The best ASIC solution depends on density requirements and production volumes.*



1101 01

### USER PROGRAMMABLE GATE ARRAYS

Unlike conventional gate arrays, programmable gate arrays require no fixed costs, and no custom factory fabrication. Since each device is identical, manufacturing costs follow the same learning curve as other high-volume standard product ICs.

### STANDARD CELL AND CUSTOM ICs

Standard cell and custom ICs require unique masks for all layers used in manufacturing. This imposes extra costs and delays for development, but results in the lowest production costs for high volume applications. Standard cell ICs offer the advantages of high level building blocks and analog functions.

### PROGRAMMABLE LOGIC DEVICES (PLDs)

PLDs are often used in place of five to ten SSI/MSI devices, and are the most efficient ASIC solution for densities up to a few hundred gates. Programmable Logic Devices (PLDs) include a number of competing alternatives, all based on variations of AND-OR plane architectures. The primary limitations of the PLD architecture are the number of flip-flops, the number of input/output signals, and the rigidity of the AND-OR plane logic and its interconnections. The use of one function often precludes the use of many other similar functions.

### GATE ARRAYS

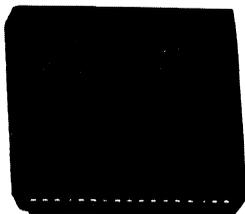
Gate arrays implement user logic by interconnecting transistors or simple gates into more complex functions during the last stages of the manufacturing process. Gate arrays offer densities up to 100,000 gates or more, with utilization of 80-90% for smaller devices, and 40-60% for the largest.

Unlike standard IC products, gate array costs include fixed costs as well as the production cost per unit. Gate arrays become cost effective when production volumes are high enough to provide a broad base for amortization of fixed costs.



# XC2000

## PROGRAMMABLE GATE ARRAY FAMILY



The XC2000 series of programmable gate arrays was introduced in 1985. Price reductions since that time have reflected steadily increasing production volumes. The family includes two compatible arrays: the XC2064 with 1200 gates, and the XC2018 with 1800 gates.

### FEATURES

- Fully user-programmable:
  - I/O Functions
  - Logic and storage functions
  - Interconnections
- Three performance options: 33, 50, and 70 MHz toggle rates
- Three package types: Dual In-line Package  
Plastic Leadless Chip Carrier  
Pin Grid Array
- TTL or CMOS input thresholds



### THE XC2000 FAMILY OF PROGRAMMABLE GATE ARRAYS

	XC2064	XC2018
Number of gates	1200	1800
Configurable Logic Blocks	64	100
Combinatorial Logic Functions	128	200
Latches and flip-flops	122	174
Input/Outputs	58	74

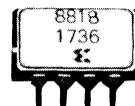
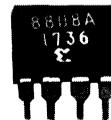
# XC1736

## CMOS SERIAL CONFIGURATION PROM

The 1736 Serial Configuration PROM is a companion device that provides permanent storage of LCA configuration programs. It can be used whenever a dedicated device is preferable to sharing of a larger EPROM, or to loading from a microprocessor.

### FEATURES

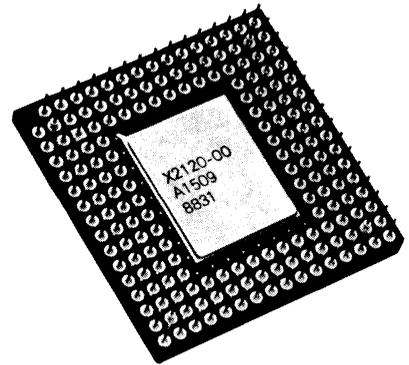
- One-Time Programmable (OTP) 36,288 bit serial memory designed to store configuration programs for Programmable Gate Arrays
- Simple interface to a Logic Cell Array (LCA) requires only two I/O pins
- Daisy chain support for multiple devices
- Cascadable for large arrays or many LCAs
- Storage of multiple configurations for a single LCA
- Low power CMOS EPROM process
- Space-efficient, low-cost 8-pin DIP packages



# XC3000

## PROGRAMMABLE GATE ARRAY FAMILY

The XC3000 series is a second generation family of CMOS programmable gate arrays that includes five compatible members with logic densities from 2000 to 9000 gates.



### FEATURES

- Fully user-programmable:
  - I/O Functions
  - Logic and storage functions
  - Interconnections
- Five member product family
  - 2000-9000 gates
  - Compatibility for ease of design migration
- Two performance options:
  - 50 and 70 MHz toggle rates
- Second generation architecture
  - 5-input logic functions
  - 2 flip-flops per CLB/IOB
  - Enhanced routing resources
  - Three-state drivers for wide ANDs
- Programmable voltage slew rates on outputs
- Three package types:
  - Plastic Leadless Chip Carrier
  - Pin Grid Array
  - Quad Flat Package

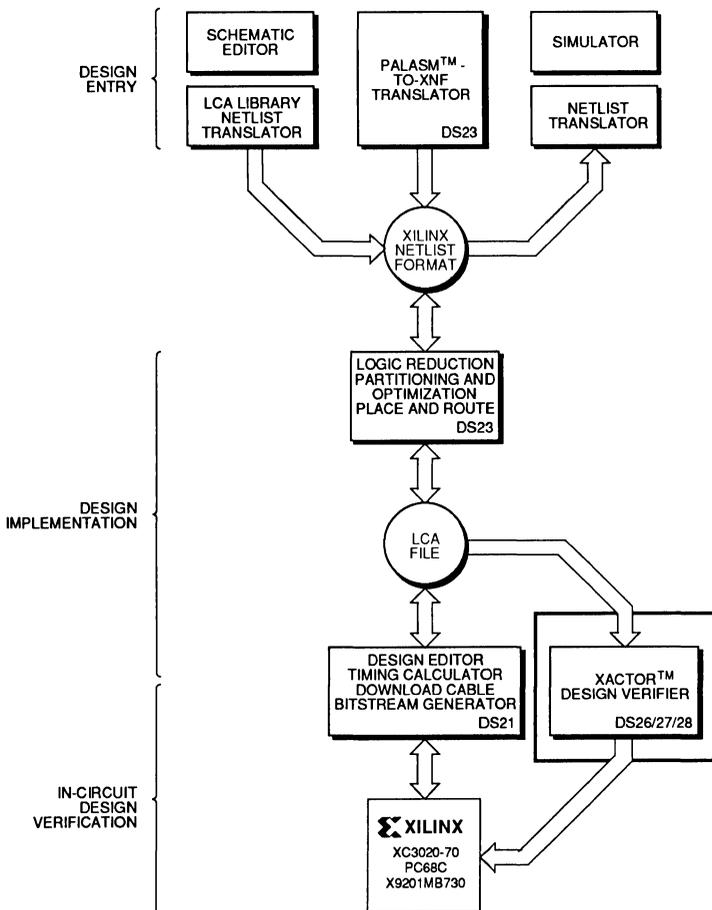
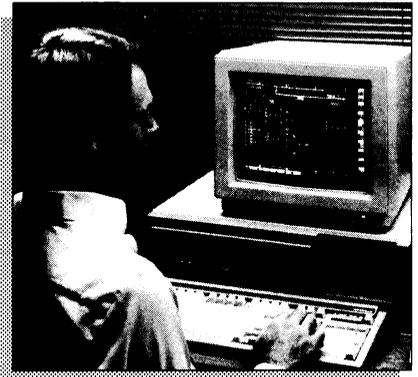
### The XC3000 Family of Programmable Gate Arrays

	XC3020	XC3030	XC3042	XC3064	XC3090
Number of gates	2000	3000	4200	6400	9000
Configurable Logic Blocks	64	100	144	224	320
Combinatorial Logic Functions	128	200	288	448	740
Latches and flip-flops	256	360	480	688	928
Input/Outputs	64	80	96	120	144



# DEVELOPMENT SYSTEMS

Designing with Xilinx Programmable Gate Arrays is similar to designing with other gate arrays. Designers can use familiar CAE tools for design entry and simulation. The open Xilinx development system includes a standard netlist format, the Xilinx Netlist File (XNF), that provides a bridge between schematic editors or simulators, and the Xilinx XACT software for design implementation and real time design verification. The Xilinx software is supported on the PC/AT and compatibles as well as on popular engineering workstations.



1101 06

## Design Entry Software

consists of libraries and netlist interfaces for standard CAE software such as FutureNet, Schema II, OrCAD, Viewlogic, Daisy, Mentor, Valid, CASE, and PALASM. Programmable gate array libraries permit design entry with standard TTL functions, with Boolean equations, and with user-defined macros.

## Simulation Software

includes models and netlist interfaces to standard simulator software, such as SILOS and CADAT, that is used for logic and timing simulations.

## Design Implementation Software

is used to convert schematic netlists and Boolean equations into efficient designs for programmable gate arrays. The software includes programs that perform partitioning, optimization, placement and routing, and interactive design editing.

## In-circuit Design Verification Tools

permit real-time verification and debugging of a programmable gate array design as soon as it is placed and routed. Designers benefit from faster and more comprehensive design verification, and from reduced requirements to generate simulation vectors to exercise a design.

# TECHNICAL SUPPORT

## SOFTWARE UPDATES

Xilinx is continuing to improve the XACT development system software, and new versions are released two-three times per year. Updates are provided free of charge during the first year after purchase, provided the user returns the registration card. After the first year, users are encouraged to purchase a Software Maintenance Agreement so that they will continue to receive software updates.

## XILINX USER GROUP

Xilinx users are invited to attend training and information exchange sessions that are held two-three times per year in various locations worldwide. These User Group meetings are intended for experienced users of Xilinx Programmable Gate Arrays, and they emphasize the efficient use of the XACT development system.

## FIELD APPLICATIONS ENGINEERS

Xilinx provides local technical support to customers through a network of Field Applications Engineers (FAEs). For the name and phone number of the nearest FAE, customers may call one of the Xilinx sales offices listed in the back of this book.

## APPLICATIONS HOT LINE

Xilinx maintains an applications hot line to provide technical support to LCA users. This service is available from 8:00 am to 6:00 pm Pacific Time. Call 1-800-255-7778 and ask for Applications Engineering.

## BULLETIN BOARD

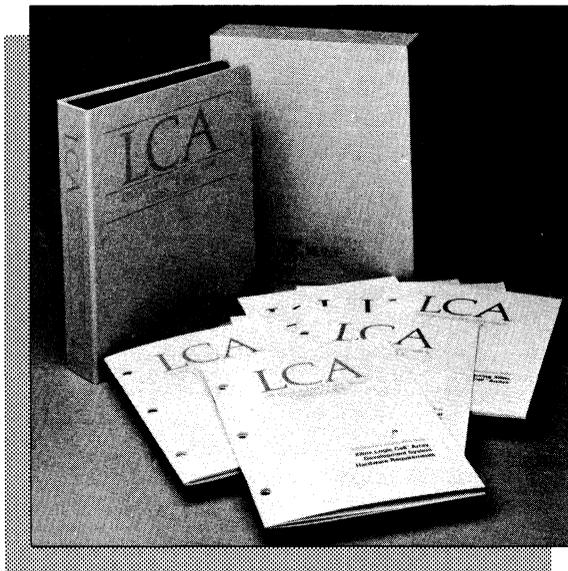
To provide customers with up-to-date information and an immediate response to questions, Xilinx provides 24-hour access to an electronic bulletin board. The Xilinx Technical bulletin board provides the following services to all registered XACT customers.

- Read files from the bulletin board
- Check current software version numbers
- Download files
- Upload files
- Leave messages for other Bulletin Board Users.

## TECHNICAL LITERATURE

In addition to this databook, technical literature for the Xilinx programmable gate array includes four volumes that are delivered with every XACT development system.

- User's Guide  
The User's Guide is a collection of "how-to" applications notes on such subjects as getting started with an LCA design, Boolean equation design entry, use of the simulator, placement and routing optimization, and LCA configuration.
- Reference Manuals (2 vols)  
The XACT Reference Manuals include a detailed description of each Xilinx software program.
- Macro library  
The Xilinx development system includes over 100 macros, including counters, registers, and multiplexers. The macro library manual includes schematics and documentation for each macro.





# A Cost of Ownership Comparison

## CONTENTS

- Executive Summary**
- ROM vs. EPROM Analogy**
- Who Recognizes the Costs?**
- Total Cost = Fixed Cost + (Variable Cost) (Units)**
- Fixed Development Costs for Gate Arrays**
  - Simulation
  - Time to Design For Testability
  - NRE Charges
  - Design Iterations
  - Test Program Development
  - Second Source
  - Summary of Fixed Development Costs
- Variable Costs**
  - Unit Cost (Cents/Gate)
  - Inventory
- Yield to Production**
- Cost of Ownership Analysis**
- Breakeven Analysis**
- Time to Market**
- Product Life Cycles**
- References**

## EXECUTIVE SUMMARY

### Introduction

Custom or mask-programmed gate arrays have many hidden costs beyond the obvious unit cost and NRE (non-recurring engineering) charges. Most of these additional costs are due to the fact that a gate array is a custom integrated circuit, one manufactured exclusively for a particular customer. Compared to a standard product, there are many hidden expenses, both during the design phase and after purchase, beyond the direct device cost.

User programmable gate arrays, on the other hand, are high volume standard products—manufactured and fully tested devices that are used by all customers. There is no customization of the silicon.

### Programmable Gate Arrays

- Standard product
- Off-the-Shelf delivery
- Fast time to market
- Programmed by the user
- No NRE
- No inventory risk
- Fully factory tested
- Simulation useful
- In-circuit design verification
- Design changes anytime
- Second source exists

### Gate Arrays

- Custom product
- Months to manufacture
- Manufacturing delays
- Programmed in the factory
- NRE Costs
- Design specific
- User develops test
- Simulation critical
- Not possible
- NRE charge repeated
- Additional cost and time

### Methodology

This analysis compares the total costs of custom gate arrays with those of user programmable gate arrays. It looks at the various categories of costs, both fixed and variable, for devices from 2000 to 9000 gates, 90% of the gate array market according to most studies.

Because the gate array has fixed or up-front development costs (NRE, extra simulation time, generating test vectors, etc.) that the programmable gate array does not, its total cost of ownership is higher until a sufficient quantity is purchased. This analysis allows the user to calculate total cost of ownership at different quantities and derive breakeven quantities—the volume below which it is more cost effective to use the programmable gate array (Breakeven Analysis). The overall objective is to determine the production volumes at which each product is most cost effective.

### Executive Summary Conclusion

The choice between user and mask programmed gate arrays must take into account more than the NRE and cents/gate unit cost. The use of a custom product entails many other costs and risks. Because of these fixed costs, it is less expensive at lower volumes to use a standard product: a programmable gate array. Since many of the

## A Cost of Ownership Comparison

hidden costs of using a custom gate array do not accrue to any one department, only the project manager can recognize the total cost.

Similar considerations have led to the widespread acceptance of EPROM memories as compared to ROMs, despite a higher EPROM cost per unit. The same factors can be applied in the choice of a gate array.

Figure 1 shows a representative breakeven graph for a 2000 gate device. The data are for 1990. The vertical axis shows the total project cost—fixed costs plus unit costs

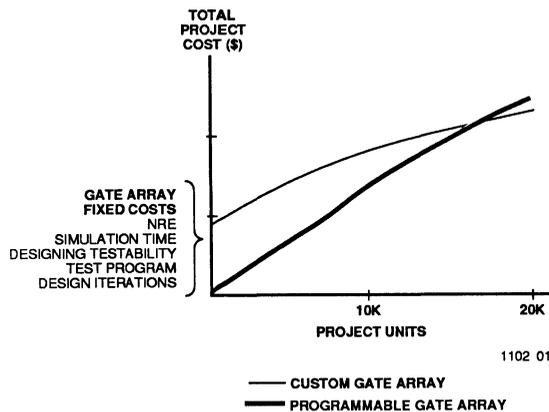


Figure 1. Typical Breakeven Analysis 2000 Gates—1990

multiplied by the number of units. At lower volumes, the custom gate array is more expensive because of fixed costs that are incurred even if no units are purchased. The programmable gate array project cost starts at zero, but rises faster because of a higher cost per-unit. In this case the breakeven volume is between 10k and 20k units. This paper will discuss the various components of this analysis and show the user how to make a similar calculation for a specific situation.

Several significant factors are omitted from this graph. First, the additional fixed costs (NRE, simulation) of bringing on a custom gate array second source are not included. Second, and much more important, the cost of the longer time to market when designing with the mask gate array is not included. This factor is reviewed in Time to Market. Both of these factors would raise the custom gate array curve and increase the breakeven quantity. In other words, the programmable gate array would be more cost effective at an even higher production volume.

## ROM VS. EPROM ANALOGY

There is a relevant historical precedent for the use of a flexible standard product instead of a custom product with a lower direct cost per unit. While EPROMs have a cost per bit that is 2 to 3 times that of ROMs, they have consistently captured almost half the programmable memory market, measured in bits shipped. See Figure 2. Many of the reasons for the use of EPROMs are the same as those for the use of programmable gate arrays: faster time to market,

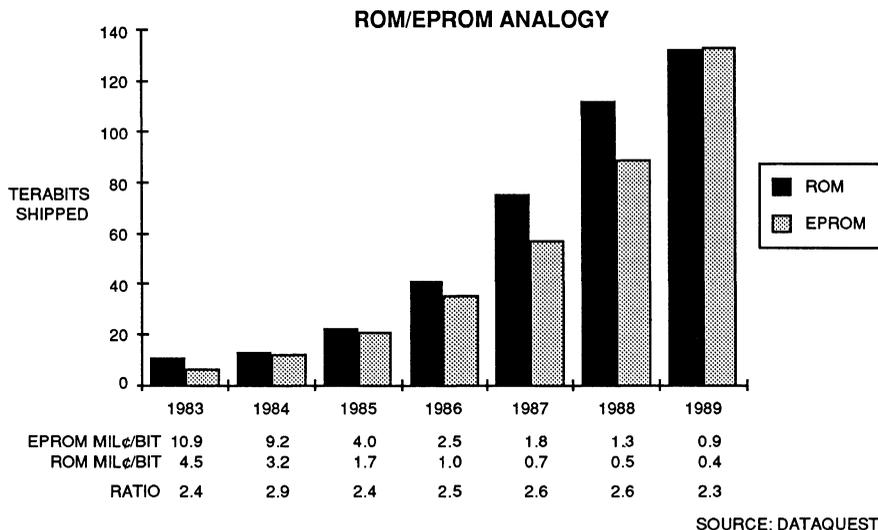


Figure 2. ROM/EPROM Analogy

lower inventory risk, easy design changes, faster delivery, and second sources. The higher price per bit is offset by the elimination of inventory and production risks.

Gate arrays have even more disadvantages versus programmable gate arrays than do ROMs versus EPROMs. The upfront design time, risk, and expense of ROMs is minimal, while that of gate arrays is substantial. ROM test tape generation is automatic, while that for gate arrays requires extensive engineering effort. Therefore, programmable gate arrays may be even more widely used versus gate arrays than are EPROMs versus ROMs.

### WHO RECOGNIZES THE COSTS?

Many of the elements of the total cost of ownership for a gate array do not accrue to a single department, and often are not fully recognized. For example, the additional engineering time needed to design for testability may not be seen by purchasing. The inventory costs of a custom product may not be recognized by the design department. However, these are real costs, and they influence the profitability of the product and company. The person making the choice between custom gate arrays and programmable gate arrays should consider the total costs of ownership for each alternative.

$$\text{TOTAL COST} = \text{FIXED COST} + (\text{VARIABLE COST})(\text{UNITS})$$

The total costs of using a product can be separated into two components. The first is the fixed costs: upfront development costs that are independent of volume. Some examples of these for gate arrays are the masking charge, simulation charge, and test program development. Due to amortization of these costs, the user's cost per unit can be very high until a sufficient volume of units is purchased. The second component of total cost is the variable cost, the incremental cost per unit. Besides the obvious unit cost, another element of variable cost is inventory cost.

This analysis will examine costs by these two categories. Fixed costs are summarized first, then variable costs. They are added to produce total cost.

### FIXED DEVELOPMENT COSTS

#### Simulation

With a custom product, it is critical that the device work the first time. Otherwise, the user must pay to have the device prototyped a second time and will incur the manufacturing delay a second time. Custom gate arrays do not support a conventional, iterative, modular design process—the

design is all-or-nothing. Simulation is a useful tool with programmable gate arrays, but it is a critical one with gate arrays, and the designer can expect to spend more time simulating a custom gate array design. The programmable gate array designer can count on in-circuit verification and on-line changes if necessary.

Gate array simulation cost includes both computer time charges and the time of the engineer doing the simulation. While the gate array vendor may or may not charge explicitly for computer time, an estimate would be \$5,000 and 2.5 manweeks of simulation effort for a 2000 gate array, and \$10,000 and 7 manweeks for a 9000 gate array. This compares to 0.5 and 1 week for the programmable gate array, with no simulation charge.

Typically one fully burdened manweek, including computer support, costs about \$2000.

	2000 Gates	9000 Gates
<b>Gate Array</b>		
Simulation Charge	\$5K	\$10K
Man Weeks	2.5 MW	7 MW
<b>Programmable Gate Array</b>		
Simulation Charge	None	None
Man Weeks	0.5 MW	1 MW

### Time to Design for Testability

One key to getting a successful gate array the first time is to focus on testing issues. The user must guarantee that the device can be fully tested in a reasonable amount of time. Since the gate array vendor's only guarantee is that the device will pass the test program, the user must be certain that if the IC meets the user-generated test specifications, it will work in the circuit.

Spending extra time in the design phase provides insurance that the device can be tested. A 1987 Dataquest ASIC Market Report observes that "an engineer can sit down at a \$20,000 CAE/CAD station and design a \$1,000,000 test problem." Designing in testability may also be the only way to provide for testing of complex sequential circuitry, or elements like long counters. Therefore the gate array designer must spend additional time in the design phase. An estimate is 1 additional week for a 2000 gate array, and 2 additional weeks for a 9000 gate array.

The programmable gate array is a standard product with no incremental test costs. It is fully tested by Xilinx before shipment. No application specific testing is needed.

## A Cost of Ownership Comparison

### Gate Array Incremental Cost

2000 Gates	9000 Gates
1 Man Week	2 Man Weeks

### NRE Charges

NRE (Non-Recurring Engineering) charges cover the on-line vendor interface, design verification, mask charges, prototype samples and a nominal simulation (pre- and post-layout) time. The charges may vary with estimated production volumes. At volumes below 50,000 units, \$15,000 to \$20,000 is a competitive quote for lower density gate arrays. At the 9000 gate level, NRE charges may be in excess of \$30,000.

There are no NRE charges for programmable gate arrays. The entire design process is done by the customer. Programmable gate array software tools run on common workstations and personal computers, and are much less expensive than comparable tools for custom gate arrays.

#### Typical Gate Array NRE for 10,000 to 50,000 units

2000 Gates	9000 Gates
\$15K-\$20K	\$30K-\$40K

### Design Iterations

The phrase "We need to add this feature" is all too common to the designer of electronic equipment. Designers often find themselves faced with the need to modify a design during prototyping or initial customer evaluation. Changes may be required to add features or reduce costs. As systems become more complex, "bugs" can be more prevalent.

Design iterations are almost never due to the failure of the gate array vendor. Rather, it is a risk associated with the choice of an inflexible technology in a very dynamic industry.

Industry data suggest that about half of all gate array designs are modified before they are released to production. When a modification is required, NRE costs are incurred for the second pass. Since resimulation is likely to involve less effort than the initial simulation, 25% (50% probability times one half the effort) of the simulation cost is added.

### Gate Array Incremental Cost

50% Probability of (original NRE time and cost + one half of original simulation time and cost)

### Test Program Development

As noted in Time to Design for Testability, testability is critical to production success for gate arrays. Gate array vendors rarely make production errors, but faulty devices may not be detected because the test vectors are not comprehensive.

The estimate for test vector development is 2 weeks for a 2000 gate array, and 4 weeks for a 9000 gate array. Since the programmable gate array is a standard product, it is fully tested at the factory. No application-specific testing is needed.

A risk that the program manager should consider involves the level of experience or knowledge that the design team has with test development. If the first pass design is unsuccessful, how much time and effort will be required to debug the problem? Both additional cost and time to market are at risk.

### Gate Array Incremental Cost

2000 Gates	9000 Gates
2 Man Weeks	4 Man Weeks

### Second Source

If a second source is required, the gate array designer must identify a compatible vendor and resubmit the design. This involves another NRE charge and time for translating logic and resimulation. The model used here is the NRE charge plus one half the simulation cost.

Programmable gate arrays are standard products that already have a second source.

### Gate Array Incremental Cost

	2000 Gates	9000 Gates
NRE Simulation Charge	\$15K-\$20K	\$30K-\$40K
Man Weeks	1 MW	3 MW

## Summary of Gate Array Fixed Development Costs

The summary in Table 1 shows typical fixed costs for both a 2000 gate and a 9000 gate array. Since assumptions may vary, a blank column is provided as a worksheet.

### VARIABLE COSTS

#### Production Unit Cost (Cents/Gate)

Gate array prices are often quoted in terms of cents per gate. For 1.5 micron, 2000 gate arrays, at the volumes considered in this analysis (10,000 to 30,000 units), a figure of 0.15–0.20 cents/gate (without package) is typical. At similar volumes, the cost per gate (without package) for a programmable gate array is 2–3 times the cost of a custom gate array. For reasons explained below, this gap is expected to narrow over the next few years. All of the cents/gate numbers are for die only. Since CMOS gate arrays and programmable gate arrays use the same packages, the package adders are equivalent.

An important consideration in calculating the total cost of ownership is the year during which most of the production

volume will be purchased. Since programmable gate arrays are newer products, their cost is declining at a steeper rate than gate arrays. They are in the introduction phase of their life cycle, while gate arrays are in a more mature phase of the cycle. Price comparisons should be based on projections over the production life of the product.

A standard product has more silicon content and less factory overhead than a custom product. Since all customers buy the same product, there is more of the semiconductor learning curve with cumulative volume. Given the profitability levels of array manufacturers, gate array prices may decline only slightly over time and could even rise.

#### 1990 Programmable Gate Array Unit Costs— Without Package

	2000 Gates	4000 Gates	9000 Gates
<b>Programmable</b> (Cents/Gate)	20KU Qty 0.30–0.40	10KU Qty 0.35–0.45	10KU Qty 0.50–0.60

	Typical 2000 Gates	Typical 9000 Gates	Customer Application
1. Simulation			
NRE	\$5,000	\$10,000	_____
Man Weeks	2 MW	6 MW	_____
2. Design for Testability			
NRE Charges	1 MW	2 MW	_____
3. NRE Charges	\$15K–\$20K	\$30K–\$40K	_____
4. Design Iterations @ 50% probability			
NRE	\$11,250	\$22,500	_____
Man Weeks	0.5 MW	1.5 MW	_____
5. Test Program Development			
NRE	2 MW	4 MW	_____
6. Second Source (NRE + 50% SIM)			
NRE	\$20,000	\$40,000	_____
Man Weeks	1 MW	3 MW	_____
<b>Total Without Second Source</b>			
NRE	\$33,750	\$67,500	_____
Man Weeks	5.5 MW	13.5 MW	_____
<b>Total With Second Source</b>			
NRE	\$53,750	\$107,500	_____
Man Weeks	6.5 MW	16.5 MW	_____
<b>Total Fixed Costs @ \$2K/MW</b>			@ \$__ /MW
Without Second Source	\$44,750	\$94,500	_____
With Second Source	\$66,750	\$140,500	_____

Table 1. Typical Fixed Costs

Process Technology

There are also technology reasons for the steeper decline in cost of the programmable gate array. Figure 3 shows that the processes used for logic IC's, including gate arrays, typically lag behind those used for memory IC's. Since the programmable gate array is a standard IC built on a memory process, it can take advantage of each new process to shrink the die and reduce costs.

With a conventional gate array, the process that is available at the time of design is usually used throughout the production lifetime of the product. Except for very high

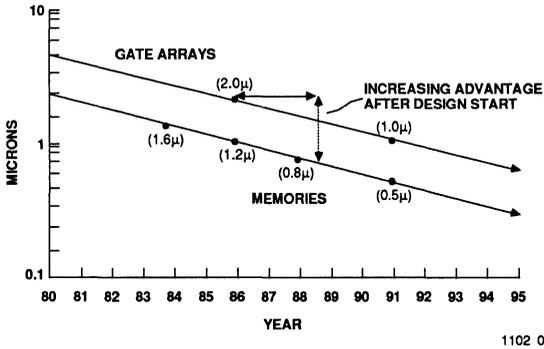


Figure 3. ROM/EPROM Analogy

volume applications, few gate arrays are retooled to take advantage of process advances. The time from design start to end of production lifetime is usually several years. Over this period, the programmable gate array will move to successively more advanced processes, resulting in steadily decreasing costs. By the end of the production lifetime, the programmable gate array will be several processes ahead and the cost difference will be reduced significantly.

Pad-Limited Die Sizes

As gate arrays and programmable gate arrays grow in I/O pin count, a phenomenon known as "pad-limiting" is more likely to occur. The spacing between I/O pads is determined by mechanical limitations of the equipment used for lead bonding. In I/O intensive applications the number of pads around the outer edge of the die determines the die size, instead of the number of gates. See Figure 4. In I/O intensive applications, a "cost per I/O" may be a more useful measure than "cost per gate."

For a given I/O count, in the pad-limited case the programmable gate array and the gate array would be the same die size. As a result, the higher volume, standard product, programmable gate array could actually be less expensive on a per-unit basis than the custom product gate array. There would be no breakeven quantity—the programmable gate array would have a lower cost of ownership at all volumes.

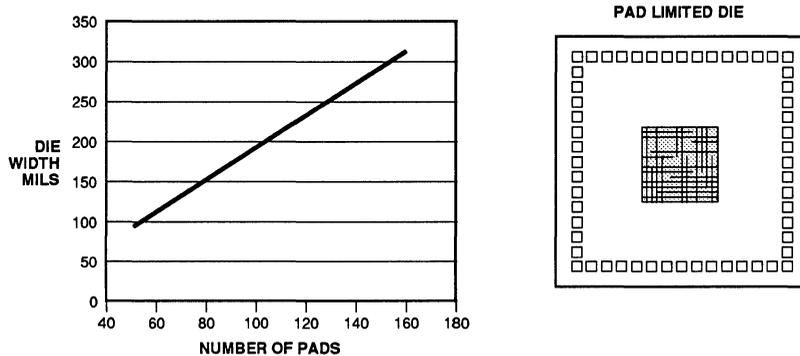


Figure 4. Minimum Die Size vs. I/O

1102 03

### Effect of Die Cost on Total Cost

Figure 5 illustrates a third point about the capability of programmable gate arrays to narrow the cost difference with custom gate arrays. The chart shows the contribution to total device cost of wafer, die, assembly and test. Wafer cost represents about 20% to 40% of the total device cost, and die cost about 30% to 50%. A 50% difference in die cost—between a gate array and a programmable gate array—shown in the chart translates to only a 20% difference (80 vs. 100) in total cost by the time the device has been tested. This comparison is based on production of the programmable gate array in a more advanced process than the custom gate array, as discussed in Unit Cost (Cents/Gate).

### Inventory Reserves

Inventories include extra devices ordered and stocked to cover contingencies. For a custom product this is the only way parts can be delivered in less than the normal production time (2–4 months). Contingencies are often thought of in terms of negative events like a defective lot or manufacturing shortfalls.

However, contingencies also include positive events like stocking for large, upside orders or where demand is difficult to estimate. This can be especially true during a product's introduction, when design changes and demand spikes occur simultaneously.

With a custom product it is also necessary to build inventory as the product nears the end of its life cycle. Demand is low and difficult to forecast, and it may not be possible to reorder a small quantity. Spares and replacements must be stocked. A JIT inventory system is less practical.

Since minimum manufacturing quantities for semiconductors are determined by wafer lots, a custom product will have excess WIP (work in process) or finished goods inventory if the desired order quantity is less than the mini-

mum economical wafer lot quantity. Inventory is created and costs are incurred. Moreover, there is the problem of inventory ownership if the parts are never ordered by the customer.

Although the safety stock reserve is a function of the cost of the product itself, a figure of 10% is reasonable for gate arrays that have unit costs under \$25.00. In comparison, since changes to programmable gate arrays can be made in software in minutes, and since only one part type is widely stocked, the comparative safety stock reserve is 0%.

### Gate Array Incremental Inventory Cost

10% Additional Unit Cost

### YIELD TO PRODUCTION

Due to rapidly changing markets, many designs never go into production. Sometimes a company will develop competing projects, with only one moving to production. Many times the market will change, or competition will emerge, and projects will be cancelled or redirected. Of course each design team expects that its project will succeed, but in the aggregate this is not true. If a company chooses gate arrays as the primary logic technology, and starts many designs, this factor will occur.

The March, 1986 Technology Research Letter states that only 1/3 of gate array designs go into production. The 1987 Dataquest ASIC and Standard Logic Semiconductor Volume 1 reports 50% go into production. With 50%, the true cost of the gate array should recognize additional costs for simulation, designing for testability, and NRE. For 2000 gates, using the numbers in Summary of Fixed Development Costs, this would mean an additional (\$5,000 + 3 MW + \$17,500). For 9000 gates the number is (\$10,000 + 8 MW + \$35,000).

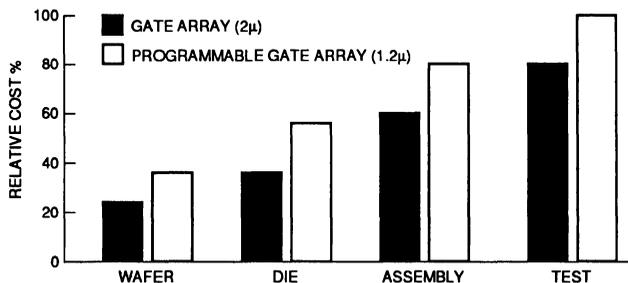


Figure 5. Relative Manufacturing Cost by Stage of Completion

**A Cost of Ownership Comparison**

**Gate Array Incremental Cost**

Simulation Cost + Time to Design for Testability + NRE Cost

mary of Fixed Development Costs). Therefore, at lower unit volumes the programmable gate array is less expensive, until the gate array can amortize the upfront fixed costs.

**COST OF OWNERSHIP ANALYSIS**

While gate arrays have a lower unit cost, they have incremental fixed costs that must be incurred before the first unit is received. Example costs are shown in Table 1 (Summary of Fixed Development Costs).

Therefore, at lower unit volumes the programmable gate array is less expensive, until the gate array can amortize the upfront fixed costs. Table 2 is a form that can be used for calculating the total cost of ownership at various volumes. Table 2 points to the "breakeven quantity"—the quantity where the unit cost of the two devices is the same—of the next section.

Project Quantity	1,000	5,000	10,000	20,000
<b>Gate Array—No second source</b>				
1. Fixed costs from Table 1	_____	_____	_____	_____
2. Unit cost	_____	_____	_____	_____
3. Inventory reserves: (Line 2)(1.1)	_____	_____	_____	_____
4. Total variable cost = (Line 3)(Qty)	_____	_____	_____	_____
5. Total cost = Line 1 + Line 4	_____	_____	_____	_____
6. Unit cost = Line 5/Qty	_____	_____	_____	_____
<b>Gate Array—second source</b>				
7. Fixed costs from Table 1	_____	_____	_____	_____
8. Second source costs	_____	_____	_____	_____
9. Total fixed costs	_____	_____	_____	_____
10. Total variable cost—Line 4 above	_____	_____	_____	_____
11. Total cost = Line 9 + Line 10	_____	_____	_____	_____
12. Unit cost = Line 11/Qty	_____	_____	_____	_____
<b>Programmable Gate Array</b>				
13. Unit cost	_____	_____	_____	_____

**Table 2. Total Cost vs. Volume Purchased**

## BREAKEVEN ANALYSIS

Figure 6 is a graphical representation of the breakeven calculation for the case of 2000 gates, 1990 pricing, and no second source. Up to the breakeven unit volume, the programmable gate array solution has a lower total project cost.

Similar graphs can be built for different assumptions by filling in Table 1. For the gate array, the breakeven graph is merely line 5 or line 11 plotted versus quantity. For the programmable gate array, it is line 13 times the quantity plotted versus the quantity.

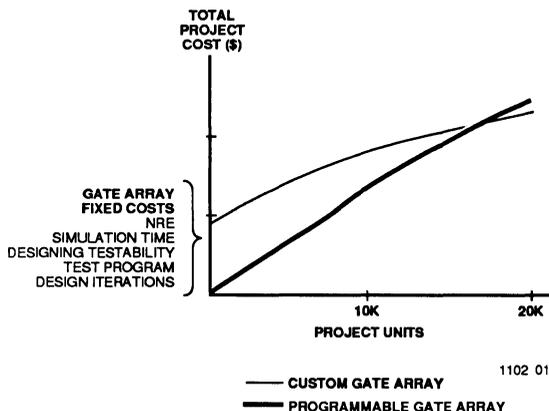


Figure 6. Typical Breakeven Analysis 2000 Gates—1990

## TIME TO MARKET

There are numerous examples of products that failed due to late market entry. A study by McKinsey & Co. stated that a product that is six months late to market will miss out on 1/3 of the potential profit over the product's lifetime. If there is any problem in simulation, or any iteration of the gate array design, then a gate array would easily add six additional months to a product schedule.

At the 2,000 gate level, assume the gate array is used in a

\$2,000 product that has 15% profit margins. For 10,000 units sold:

$$\text{Lost Profit} = \$2,000 \times 10,000 \times 15\% \times 1/3 = \$1.0 \text{ million or } \$100 \text{ per device}$$

At the 9000 gate level, assume the gate array is used in a \$10,000 product that has 20% profit margins. For 2000 units sold:

$$\text{Lost Profit} = \$10,000 \times 2,000 \times 20\% \times 1/3 = \$1.33 \text{ million or } \$667 \text{ per device}$$

Note that these catastrophic costs are not included in any of the previous sections. They are a quantitative estimate of the risk of using a custom product.

## PRODUCT LIFE CYCLES

In the electronics industry the lifetimes of products are shrinking. In the personal computer industry it is not uncommon to find product upgrades within 6-12 months. This means that the volumes associated with any one gate array design can be much smaller than anticipated, even if the end product still exists. It also means that it is critical to achieve a rapid design time.

The numbers used for Figure 1 show that 2000 gate programmable gate arrays are more economical at volumes up to 10,000 to 20,000 units. These volumes will represent an increasing number of products.

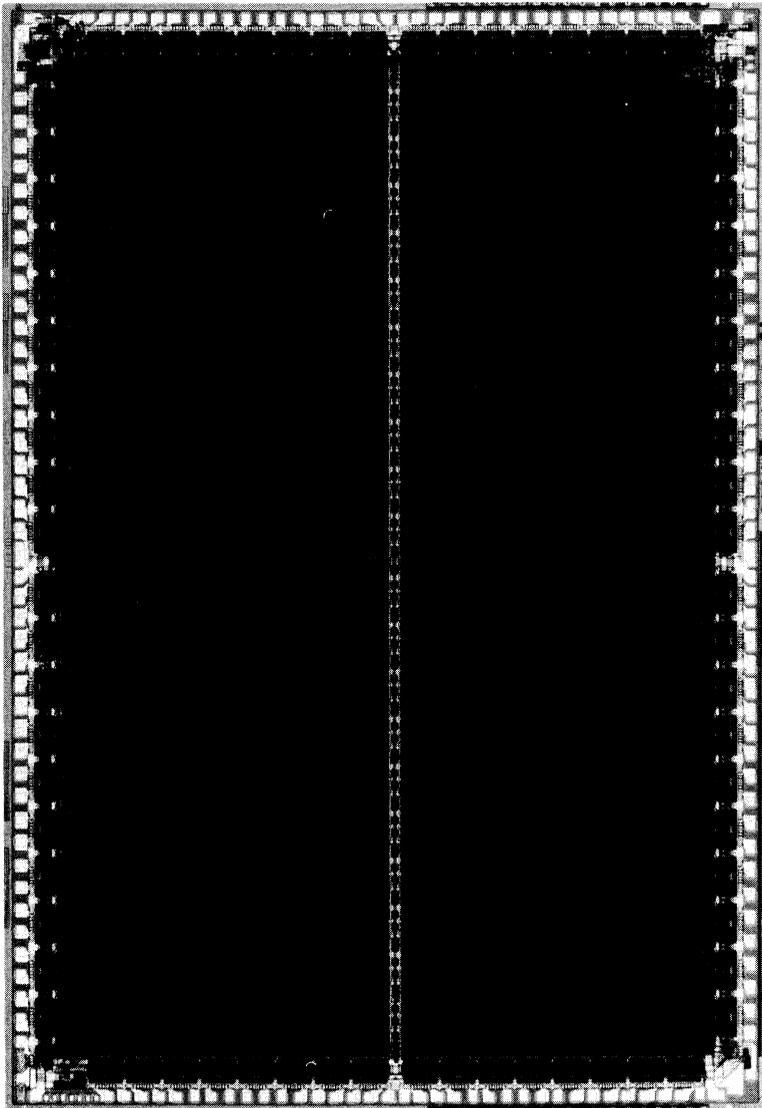
## REFERENCES

1. *Technology Research Group Letter*, March 1986, page 7.
2. Dataquest Inc., *Gate Arrays—Product Analysis*, page 3, ASIC and Standard Logic Semiconductors, 1987.
3. Reinertsen, Donald G., "Whodunit? The Search for the New-Product Killers," *Electronic Business*, July 1983, pages 62-66.
4. Integrated Circuit Engineering Corp., *ASIC Outlook*, 1987.



The Programmable Gate Array Company

---



**XC3090 Die**

---

**1 Programmable Gate Arrays****2 *Product Specifications*****3 Quality, Testing, Packaging****4 Technical Support****5 Development Systems****6 Applications****7 Article Reprints****8 Index**

---

## XC3000 Logic Cell Array Family

Features, Description, Architecture .....	2-1
Interconnect .....	2-7
Internal Buses .....	2-11
Programming .....	2-14
Special Configuration Functions .....	2-21
Performance .....	2-23
Power .....	2-24
Development Systems .....	2-29
Pin Descriptions .....	2-30
Parametrics .....	2-37
Ordering Information .....	2-40
Package Dimensions .....	2-50

## XC2064, XC2018 Logic Cell Array

Features, Description, Architecture .....	2-55
Interconnect .....	2-59
Power .....	2-64
Programming .....	2-66
Special Configuration Functions .....	2-71
Performance .....	2-72
Pin Descriptions .....	2-78
Parametrics .....	2-82
Ordering Information, Package Dimensions .....	2-91

## XC2064 R/B, XC2018 R/B Military LCAs

Features, Description .....	2-97
Static Burn-In Circuits .....	2-100
XC2064 R/B Test Specifications .....	2-101
XC2018 R/B Test Specifications .....	2-105
Switching Characteristics .....	2-109

## XC1736 Serial Configuration PROM

Features, Description, Block Diagram .....	2-113
Pin Description .....	2-114
Programming Flow Chart .....	2-117
Parametrics .....	2-118
Ordering Information, Package Dimensions .....	2-123

---



# XC3000 Logic Cell™ Array Family

## Product Specification

### FEATURES

- High Performance—50 and 70 MHz Toggle Rates
- Second Generation User-Programmable Gate Array
  - I/O functions
  - Digital logic functions
  - Interconnections
- Flexible array architecture
  - Compatible arrays, 2000 to 9000 gate logic complexity
  - Extensive register and I/O capabilities
  - High fan-out signal distribution
  - Internal three-state bus capabilities
  - TTL or CMOS input thresholds
  - On-chip oscillator amplifier
- Standard product availability
  - Low power, CMOS, static memory technology
  - Performance equivalent to TTL SSI/MSI
  - 100% factory pre-tested
  - Selectable configuration modes
- Complete XACT™ development system
  - Schematic Capture
  - Automatic Place/Route
  - Logic and Timing Simulation
  - Design Editor
  - Library and User Macros
  - Timing Calculator
  - XACTOR In-Circuit Verifier
  - Standard PROM File Interface

### DESCRIPTION

The CMOS XC3000 Logic Cell™ Array (LCA) family provides a group of high-performance, high-density, digital, integrated circuits. Their regular, extendable, flexible, user-programmable array architecture is composed of a configuration program store plus three types of configurable elements: a perimeter of I/O Blocks, a core array of Logic Blocks and resources for interconnection. The general structure of a Logic Cell Array is shown in Figure 1 on the next page. The XACT development system provides schematic capture and auto place-and-route for design entry. Logic and timing simulation, and in-circuit emulation are available as design verification alternatives. The design editor is used for interactive design optimization, and to compile the data pattern which represents the configuration program.

The Logic Cell Array's user logic functions and interconnections are determined by the configuration program data stored in internal static memory cells. The program can be loaded in any of several modes to accommodate various system requirements. The program data resides externally in an EEPROM, EPROM or ROM on the application circuit board, or on a floppy disk or hard disk. On-chip initialization logic provides for optional automatic loading of program data at power-up. Xilinx's companion XC1736 Serial Configuration PROM provides a very simple serial configuration program storage in a one-time-programmable eight-pin DIP.

Basic Array	Logic Capacity (usable gates)	Configurable Logic Blocks	User I/Os	Program Data (bits)
XC3020	2000	64	64	14779
XC3030	3000	100	80	22176
XC3042	4200	144	96	30784
XC3064	6400	224	120	46064
XC3090	9000	320	144	64160

The XC3000 Logic Cell Arrays are an enhanced family of Programmable Gate Arrays, which provide a variety of logic capacities, package styles, temperature ranges and speed grades.

### ARCHITECTURE

The perimeter of configurable I/O Blocks (IOBs) provides a programmable interface between the internal logic array and the device package pins. The array of Configurable Logic Blocks (CLBs) performs user-specified logic functions. The interconnect resources are programmed to form networks, carrying logic signals among blocks, analogous to printed circuit board traces connecting MSI/SSI packages.

The blocks' logic functions are implemented by programmed look-up tables. Functional options are implemented by program-controlled multiplexers. Interconnecting networks between blocks are implemented with metal segments joined by program-controlled pass tran-

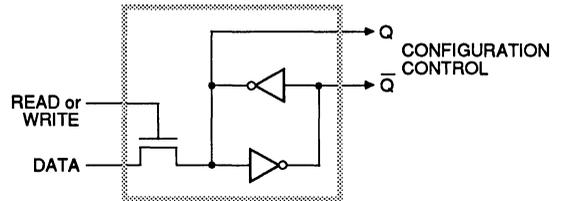
sisters. These functions of the Logic Cell Array are established by a configuration program which is loaded into an internal, distributed array of configuration memory cells. The configuration program is loaded into the Logic Cell Array at power-up and may be reloaded on command. The Logic Cell Array includes logic and control signals to implement automatic or passive configuration. Program data may be either bit serial or byte parallel. The XACT development system generates the configuration program bit-stream used to configure the Logic Cell Array. The memory loading process is independent of the user logic functions.

**CONFIGURATION MEMORY**

The static memory cell used for the configuration memory in the Logic Cell Array has been designed specifically for high reliability and noise immunity. Integrity of the LCA configuration memory based on this design is assured even under adverse conditions. Compared with other programming alternatives, static memory provides the best combination of high density, high performance, high reliability and comprehensive testability. As shown in Figure 2, the basic memory cell consists of two CMOS inverters plus a pass transistor used for writing and read-

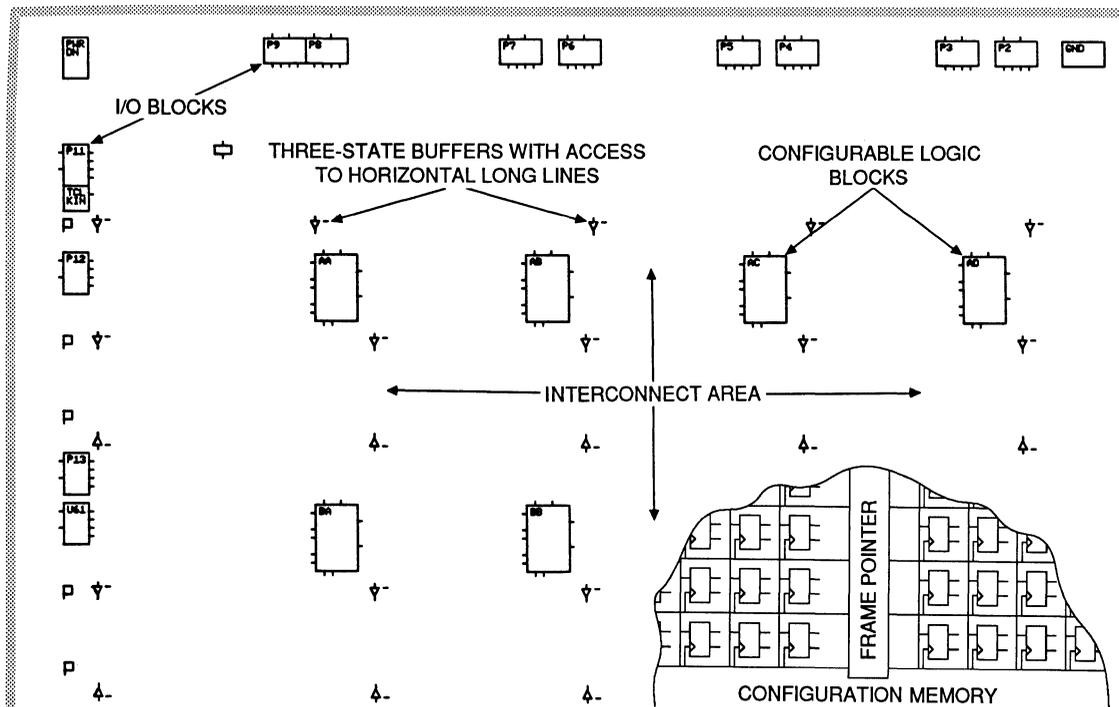
ing cell data. The cell is only written during configuration and only read during readback. During normal operation the cell provides continuous control and the pass transistor is "off" and does not affect cell stability. This is quite different from the operation of conventional memory devices, in which the cells are frequently read and re-written.

The memory cell outputs Q and  $\bar{Q}$  use full Ground and Vcc levels and provide continuous, direct control. The additional capacitive load together with the absence of address decoding and sense amplifiers provide high stability to the



1105 12

**Figure 2.** A static configuration memory cell is loaded with one bit of configuration program and controls one program selection in the Logic Cell Array.



**Figure 1.** The structure of the Logic Cell Array consists of a perimeter of programmable I/O blocks, a core of configurable logic blocks and their interconnect resources. These are all controlled by the distributed array of configuration program memory cells.

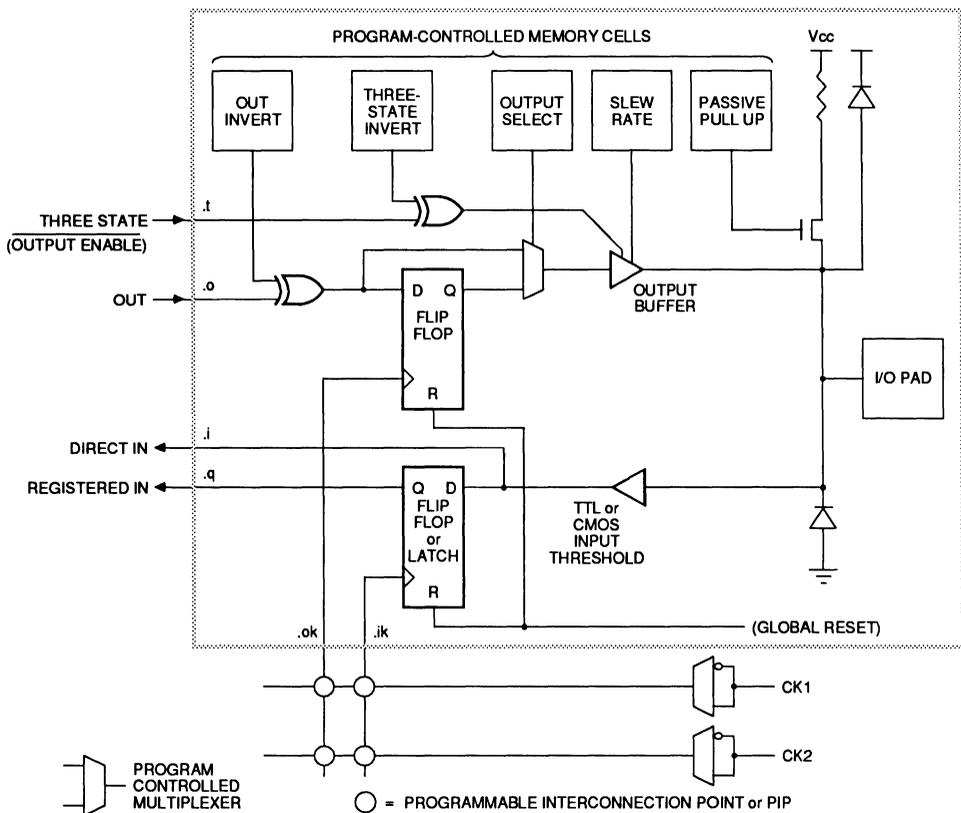
cell. Due to the structure of the configuration memory cells, they are not affected by extreme power supply excursions or very high levels of alpha particle radiation. In reliability testing no soft errors have been observed, even in the presence of very high doses of alpha radiation.

The method of loading the configuration data is selectable. Two methods use serial data, while three use byte wide data. The internal configuration logic utilizes framing information, embedded in the program data by the XACT development system, to direct memory cell loading. The serial data framing and length count preamble provide programming compatibility for mixes of various Xilinx programmable gate arrays in a synchronous, serial, daisy-chain fashion.

### I/O BLOCK

Each user-configurable I/O Block (IOB), shown in Figure 3, provides an interface between the external package pin of the device and the internal user logic. Each I/O Block includes both registered and direct input paths. Each IOB provides a programmable three-state output buffer which may be driven by a registered or direct output signal. Configuration options allow each IOB an inversion, a controlled slew rate and a high impedance pull-up. Each input circuit also provides input clamping diodes to provide electro-static protection, and circuits to inhibit latch-up produced by input currents.

The input buffer portion of each I/O Block provides thresh-



1105 01

**Figure 3.** The Input/Output Block includes input and output storage elements and I/O options selected by configuration memory cells. A choice of two clocks is available on each die edge. All user inputs are programmed for TTL or CMOS thresholds.

old detection to translate external signals applied to the package pin to internal logic levels. The global input-buffer threshold of the I/O Blocks can be programmed to be compatible with either TTL or CMOS levels. The buffered input signal drives the data input of a storage element which may be configured as a positive edge-triggered "D" flip-flop or a low level-transparent latch. The sense of the clock can be inverted (negative edge/high transparent) as long as all IOBs on the same clock net use the same clock sense. Clock/load signals (I/O Block pins *.ik* and *.ok*) can be selected from either of two die edge metal lines. I/O storage elements are reset during configuration or by the active low chip  $\overline{\text{RESET}}$  input. Both direct input [from I/O Block pin *.j*] and registered input [from I/O Block pin *.q*] signals are available for interconnect.

For reliable operation inputs should have transition times of less than 100 ns and should not be left floating. Floating CMOS input-pin circuits might be at threshold and produce oscillations. This can produce additional power dissipation and system noise. A typical hysteresis of about 300 mV reduces sensitivity to input noise. Each user I/O Block includes a programmable high impedance pull-up resistor which may be selected by the program to provide a constant HIGH for otherwise undriven package pins. Although the Logic Cell Array provides circuitry to provide input protection for electrostatic discharge, normal CMOS handling precautions should be observed.

Flip-flop loop delays for the I/O Block and logic block flip-flops are about 3 nanoseconds. This short delay provides good performance under asynchronous clock and data conditions. Short loop delays minimize the probability of a metastable condition which can result from assertion of the clock during data transitions. Because of the short loop delay characteristic in the Logic Cell Array, the I/O Block flip-flops can be used to synchronize external signals applied to the device. Once synchronized in the I/O Block, the signals can be used internally without further consideration of their clock relative timing, except as it applies to the internal logic and routing path delays.

Output buffers of the I/O Blocks provide CMOS-compatible 4 mA source-or-sink drive for high fan-out CMOS or TTL compatible signal levels. The network driving I/O Block pin *.o* becomes the registered or direct data source for the output buffer. The three-state control signal [I/O Block pin *.f*] can control output activity. An open-drain type output may be obtained by using the same signal for driving the output and three-state signal nets so that the buffer output is enabled only for a LOW.

Configuration program bits for each I/O Block control features such as optional output register, logical signal inversion, and three-state and slew rate control of the output.

The program-controlled memory cells of Figure 3 control the following options:

- **Logical Inversion of the output** is controlled by one configuration program bit per I/O Block.
- **Logical three-state control** of each I/O Block output buffer is determined by the states of configuration program bits which turn the buffer on, or off, or select the output buffer three-state control interconnection [I/O Block pin *.f*]. When this I/O Block output control signal is HIGH, a logic "1", the buffer is **disabled** and the package pin is high impedance. When this I/O block output control signal is LOW, a logic "0", the buffer is **enabled** and the package pin is active. Inversion of the buffer three-state control logic sense (output enable) is controlled by an additional configuration program bit.
- **Direct or registered output** is selectable for each I/O block. The register uses a positive-edge, clocked flip-flop. The clock source may be supplied [I/O Block pin *.ok*] by either of two metal lines available along each die edge. Each of these lines is driven by an invertible buffer.
- **Increased output transition speed** can be selected to improve critical timing. Slower transitions reduce capacitive load peak currents of non-critical outputs and minimize system noise.
- A high impedance **pull-up resistor** may be used to prevent unused inputs from floating.

### Summary of I/O Options

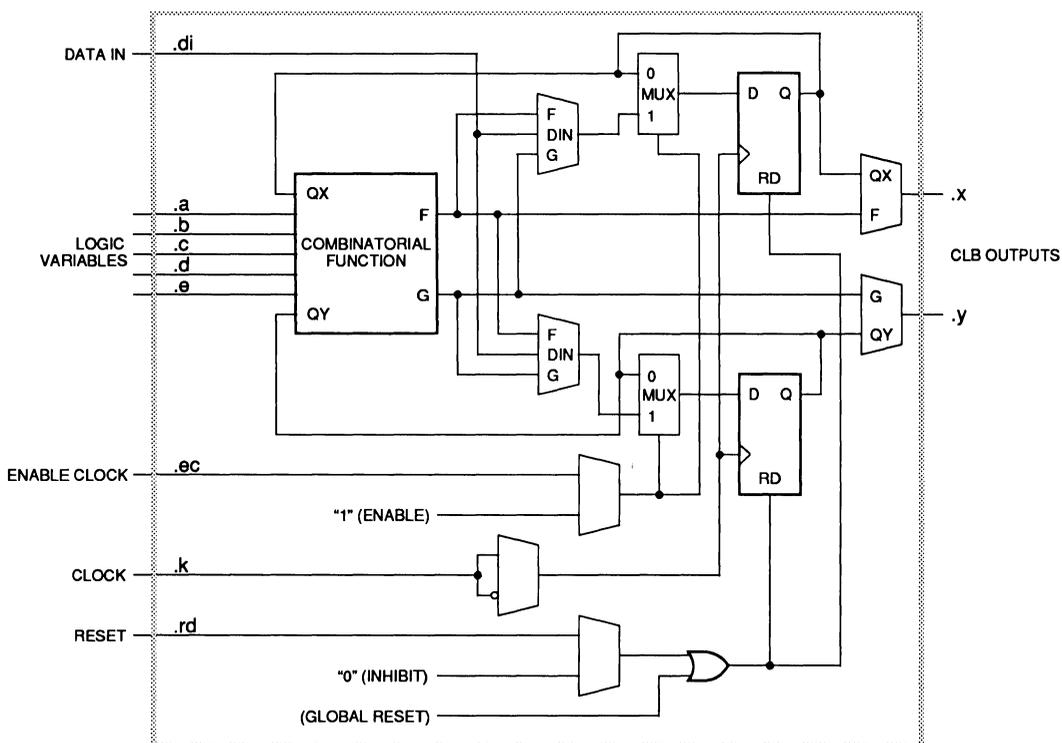
- Inputs
  - Direct
  - Flip-flop/latch
  - CMOS/TTL threshold (chip inputs)
  - Pull-up resistor/open circuit
- Outputs
  - Direct/registered
  - Inverted/not
  - Three-state/on/off
  - Full speed/slew limited
  - Three-state/output enable (inverse)

## CONFIGURABLE LOGIC BLOCK

The array of Configurable Logic Blocks (CLBs) provides the functional elements from which the user's logic is constructed. The logic blocks are arranged in a matrix within the perimeter of I/O Blocks. The XC3020 has 64 such blocks arranged in 8 rows and 8 columns. The XACT development system is used to compile the configuration data which are to be loaded into the internal configuration memory to define the operation and interconnection of each block. User definition of configurable logic blocks and their interconnecting networks may be done by automatic translation from a schematic capture logic diagram or optionally by installing library or user macros.

Each configurable logic block has a combinatorial logic section, two flip-flops, and an internal control section. See Figure 4. There are: five logic inputs [.a, .b, .c, .d and .e]; a common clock input [.k]; an asynchronous direct reset input [.rd]; and an enable clock [.ec]. All may be driven from the interconnect resources adjacent to the blocks. Each CLB also has two outputs [.x and .y] which may drive interconnect networks.

Data input for either flip-flop within a CLB is supplied from the function F or G outputs of the combinatorial logic, or the block input, data-in [.di]. Both flip-flops in each CLB share the asynchronous reset [.rd] which, when enabled and HIGH, is dominant over clocked inputs. All flip-flops are



1105 02

**Figure 4.** Each Configurable Logic Block includes a combinatorial logic section, two flip-flops and a program memory controlled multiplexer selection of function.

It has: five logic variable inputs .a, .b, .c, .d and .e.  
 a direct data in .di  
 an enable clock .ec  
 a clock (invertible) .k  
 an asynchronous reset .rd  
 two outputs .x and .y

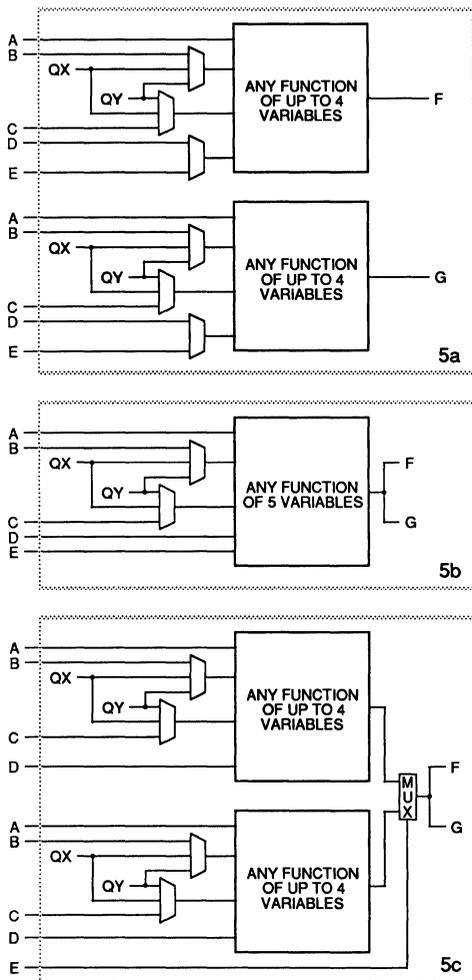


Figure 5

- 5a. Combinatorial Logic Option 1 generates two functions of four variables each. One variable, A, must be common to both functions. The second and third variable can be any choice of B, C, Qx and Qy. The fourth variable can be any choice of D or E.
- 5b. Combinatorial Logic Option 2 generates any function of five variables: A, D, E and and two choices out of B, C, Qx, Qy.
- 5c. Combinatorial Logic Option 3 allows variable E to select between two functions of four variables: Both have common inputs A and D and any choice out of B, C, Qx and Qy for the remaining two variables. Option 3 can then implement some functions of six or seven variables.

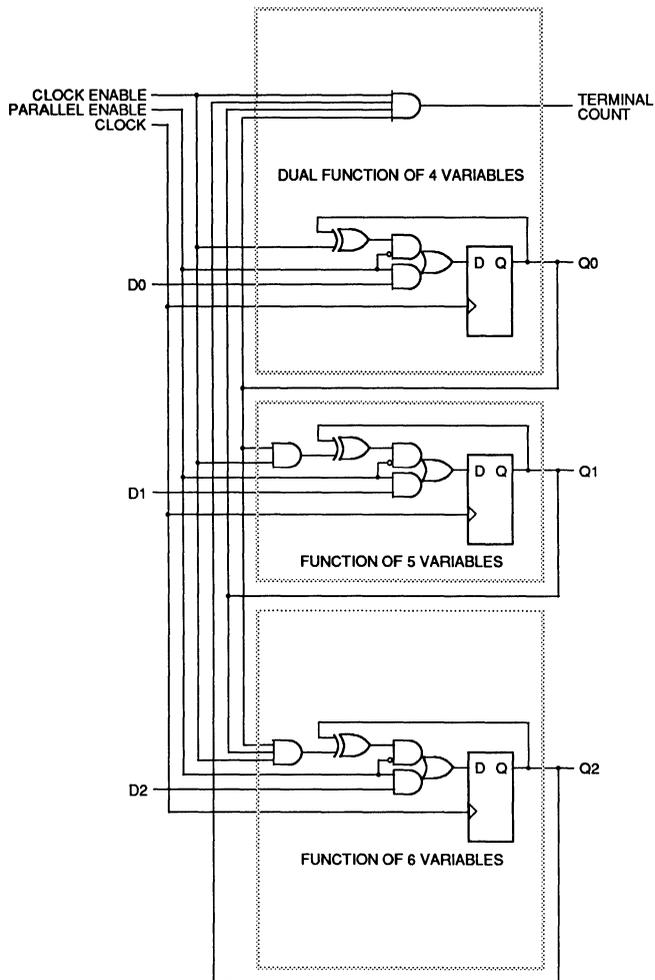


Figure 6. The C8BCP macro (modulo 8 binary counter with parallel enable and clock enable) uses one combinatorial logic block of each option.

1105 03

reset by the active low chip input,  $\overline{\text{RESET}}$ , or during the configuration process. The flip-flops share the enable clock [.ec] which, when LOW, recirculates the flip-flops' present states and inhibits response to the data-in or combinatorial function inputs on a CLB. The user may enable these control inputs and select their sources. The user may also select the clock net input [.k], as well as its active sense within each logic block. This programmable inversion eliminates the need to route both phases of a clock signal throughout the device. Flexible routing allows use of common or individual CLB clocking.

The combinatorial logic portion of the logic block uses a 32 by 1 look-up table to implement Boolean functions. Variables selected from the five logic inputs and two internal block flip-flops are used as table address inputs. The combinatorial propagation delay through the network is independent of the logic function generated and is spike free for single input variable changes. This technique can generate two independent logic functions of up to four variables each as shown in Figure 5a, or a single function of five variables as shown in Figure 5b, or some functions of seven variables as shown in Figure 5c. Figure 6 shows a modulo 8 binary counter with parallel enable. It uses one CLB of each type. The partial functions of six or seven variables are implemented using the input variable [.e] to dynamically select between two functions of four different variables. For the two functions of four variables each, the independent results (F and G) may be used as data inputs to either flip-flop or either logic block output. For the single function of five variables and merged functions of six or seven variables, the F and G outputs are identical. Symmetry of the F and G functions and the flip-flops allows the interchange of CLB outputs to optimize routing efficiencies of the networks interconnecting the logic blocks and I/O Blocks.

### PROGRAMMABLE INTERCONNECT

Programmable Interconnection resources in the Logic Cell Array provide routing paths to connect inputs and outputs of the I/O and logic blocks into logical networks. Interconnections between blocks are composed from a two-layer grid of metal segments. Specially designed pass transistors, each controlled by a configuration bit, form programmable interconnect points (PIPs) and switching matrices used to implement the necessary connections between selected metal segments and block pins. Figure 7 is an example of a routed net. The XACT development system provides automatic routing of these interconnections. Interactive routing (Editnet) is also available for design optimization. The inputs of the logic or I/O Blocks are multiplexers which can be programmed to select an input network from the adjacent interconnect segments. **As the switch connections to block inputs are unidirectional**

(as are block outputs) they are usable only for block input connection and not routing. Figure 8 illustrates routing access to logic block input variables, control inputs and block outputs. Three types of metal resources are provided to accommodate various network interconnect requirements:

- General Purpose Interconnect
- Direct Connection
- Long Lines (multiplexed busses and wide AND gates)

### General Purpose Interconnect

General purpose interconnect, as shown in Figure 9, consists of a grid of five horizontal and five vertical metal segments located between the rows and columns of logic and I/O Blocks. Each segment is the "height" or "width" of a logic block. Switching matrices join the ends of these segments and allow programmed interconnections between the metal grid segments of adjoining rows and columns. The switches of an unprogrammed device are all non-conducting. The connections through the switch matrix may be established by the automatic routing or by using "Editnet" to select the desired pairs of matrix pins to be connected or disconnected. The legitimate switching matrix combinations for each pin are indicated in Figure 10 and may be highlighted by the use of the show matrix command in XACT.

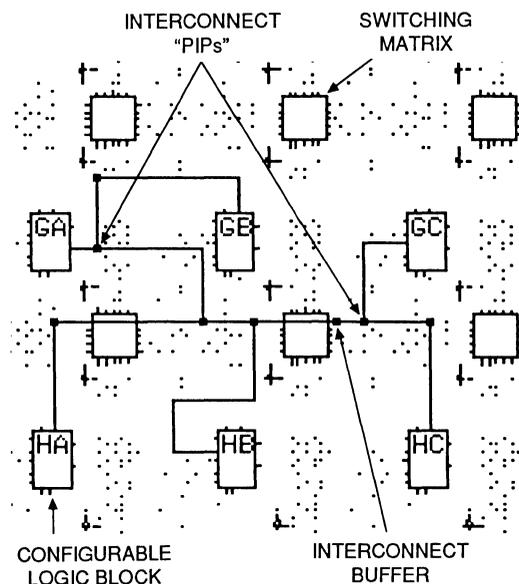
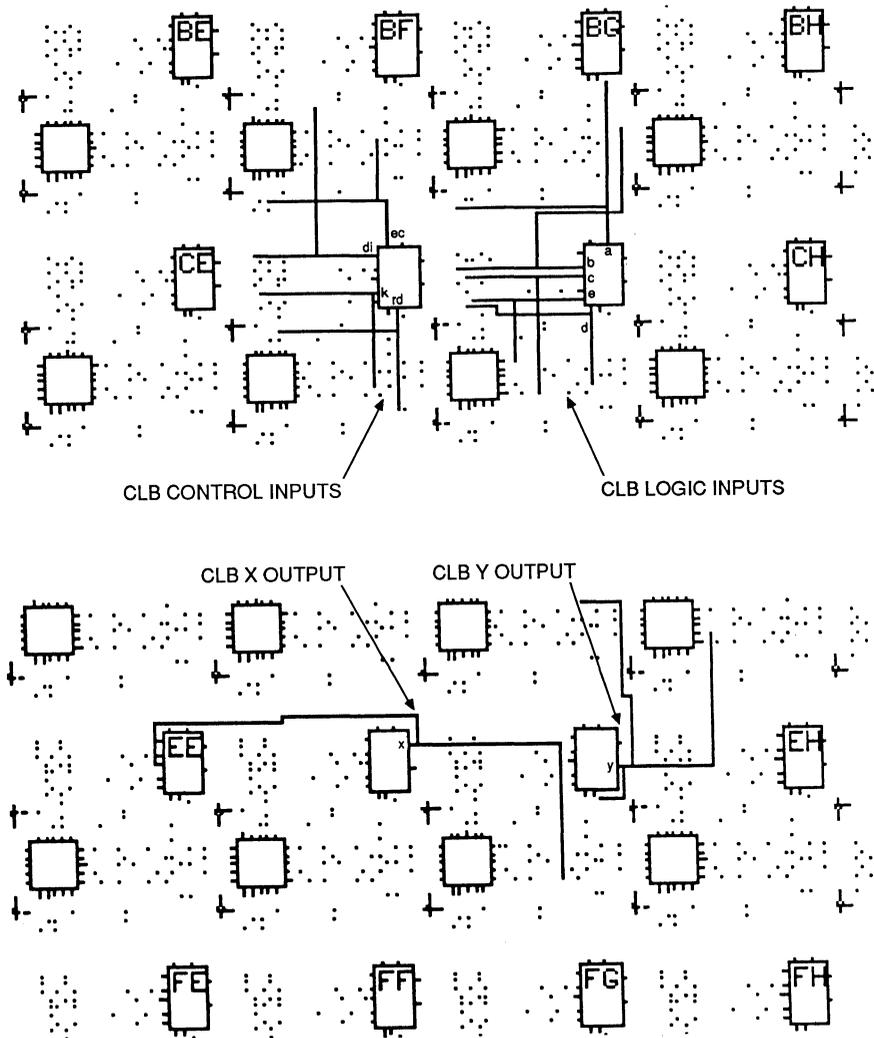


Figure 7. An XACT view of routing resources used to form a typical interconnection network from CLB GA.



**Figure 8.** The Xilinx XACT Development System view of the locations of interconnect access, CLB control inputs, logic inputs and outputs. The dot pattern represents the available programmable interconnection points (PIPs).

Some of the interconnect "PIPs" are directional. This is indicated on the XACT design editor status line:

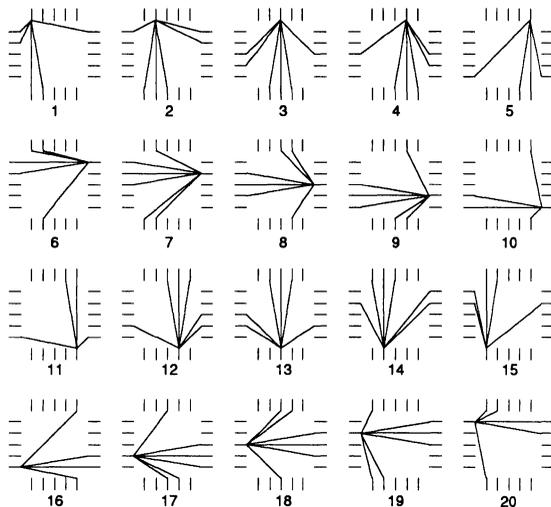
- ND is a nondirectional interconnection.
- D:H->V is a PIP which drives from a horizontal to a vertical line.
- D:V->H is a PIP which drives from a vertical to a horizontal line.
- D:C->T is a "T" PIP which drives from a cross of a T to the tail.
- D:CW is a corner PIP which drives in the clockwise direction.
- P0 indicates the PIP is non-conducting , P1 is "on."

Special buffers within the general interconnect areas provide periodic signal isolation and restoration for improved performance of lengthy nets. The interconnect buffers are available to propagate signals in either direction on a given general interconnect segment. These bi-directional (bidi) buffers are found adjacent to the switching matrices, above and to the right and may be highlighted by the use of the "Show Matrix" command in XACT. The other PIPs adjacent to the matrices are access to or from long lines. The development system automatically defines the buffer direction based on the location of the interconnection network source. The delay calculator of the XACT development system automatically calculates and displays the block, interconnect and buffer delays for any paths selected. Generation of the simulation netlist with a worst-case delay model is provided by an XACT option.

### Direct Interconnect

Direct interconnect, shown in Figure 11, provides the most efficient implementation of networks between adjacent logic or I/O Blocks. Signals routed from block to block using the direct interconnect exhibit minimum interconnect propagation and use no general interconnect resources. For each Configurable Logic Block, the .x output may be connected directly to the .b input of the CLB immediately

to its right and to the .c input of the CLB to its left. The .y output can use direct interconnect to drive the .d input of



1105 13

Figure 10. Switch matrix interconnection options for each "pin"

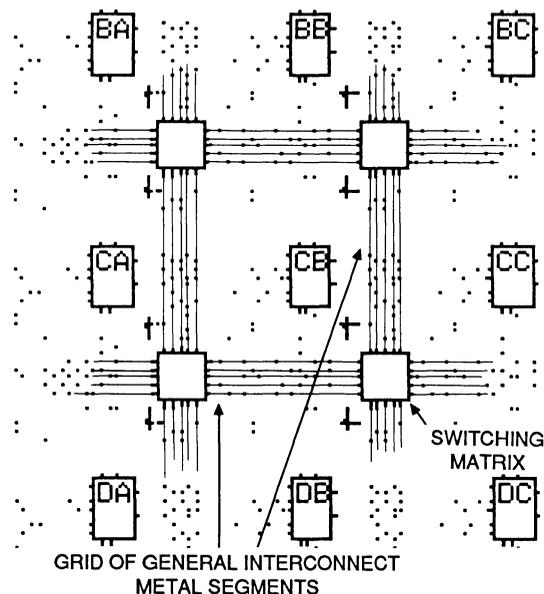


Figure 9. Logic Cell Array general-purpose interconnect is composed of a grid of metal segments which may be interconnected through switch matrices to form networks for CLB and I/O block inputs and outputs.

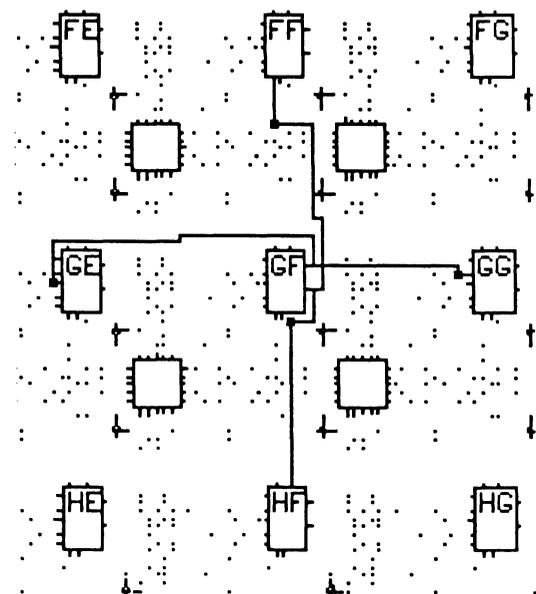


Figure 11. The .x and .y outputs of each CLB have single contact, direct access to inputs of adjacent CLBs.

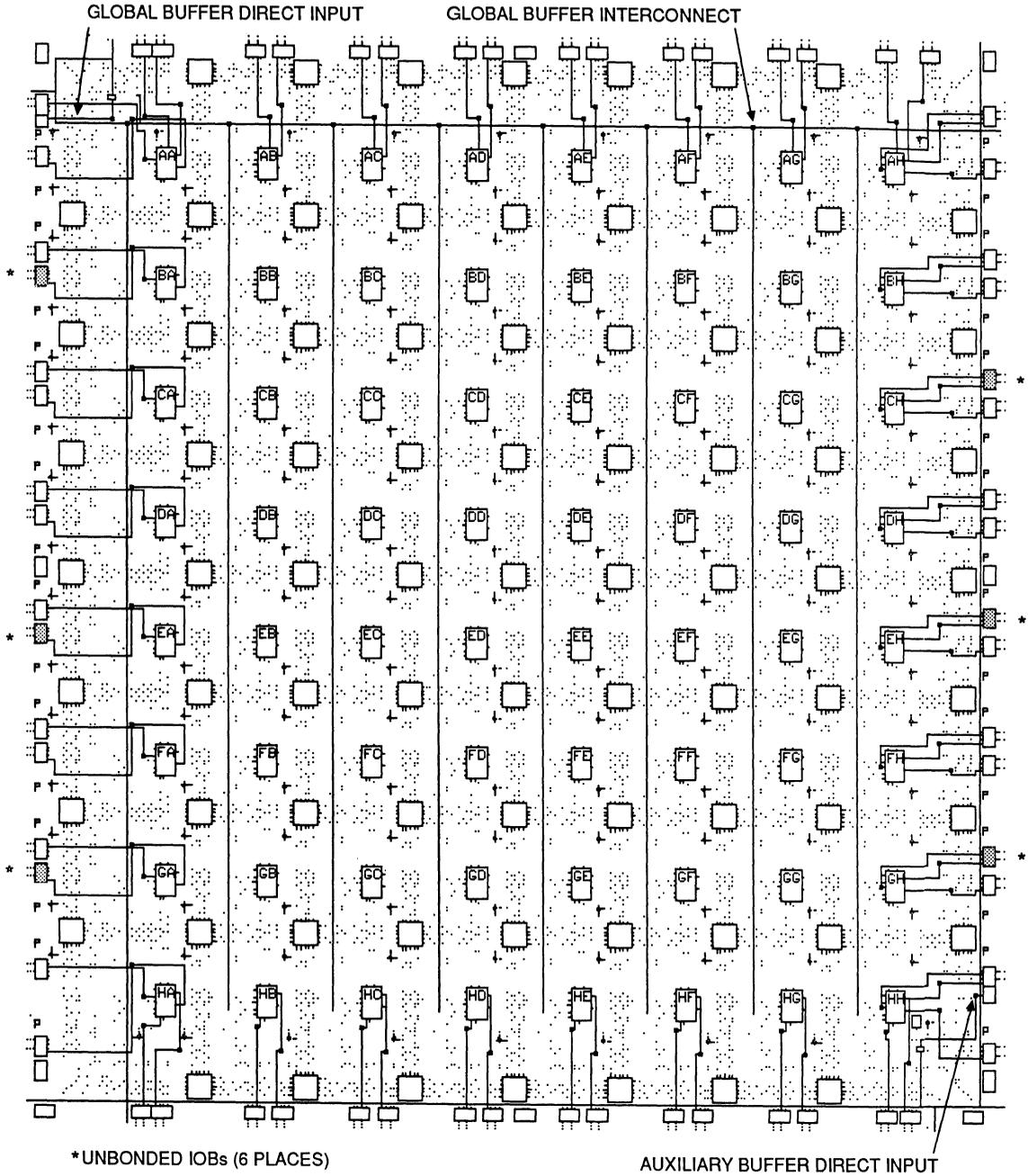


Figure 12. X3020 die dege I/O blocks are provided with direct access to adjacent CLBs.

the block immediately above and the .a input of the block below. Direct interconnect should be used to maximize the speed of high performance portions of logic. Where logic blocks are adjacent to I/O Blocks, direct connect is provided alternately to the I/O Block inputs [.i] and outputs [.o] on all four edges of the die. The right edge provides additional direct connects from CLB outputs to adjacent IOBs. Direct interconnections of I/O Blocks with CLBs are shown in Figure 12.

### Long Lines

The long lines bypass the switch matrices and are intended primarily for signals which must travel a long distance, or must have minimum skew among multiple destinations. Long Lines, shown in Figure 13, run vertically and horizontally the height or width of the interconnect area. Each interconnection column has three vertical long lines, and each interconnection row has two horizontal long lines. An additional two long lines are located adjacent to the outer sets of switching matrices. In devices larger than the XC3020, two vertical long lines in each column are connectable half-length lines. On the XC3020 only the outer long lines are.

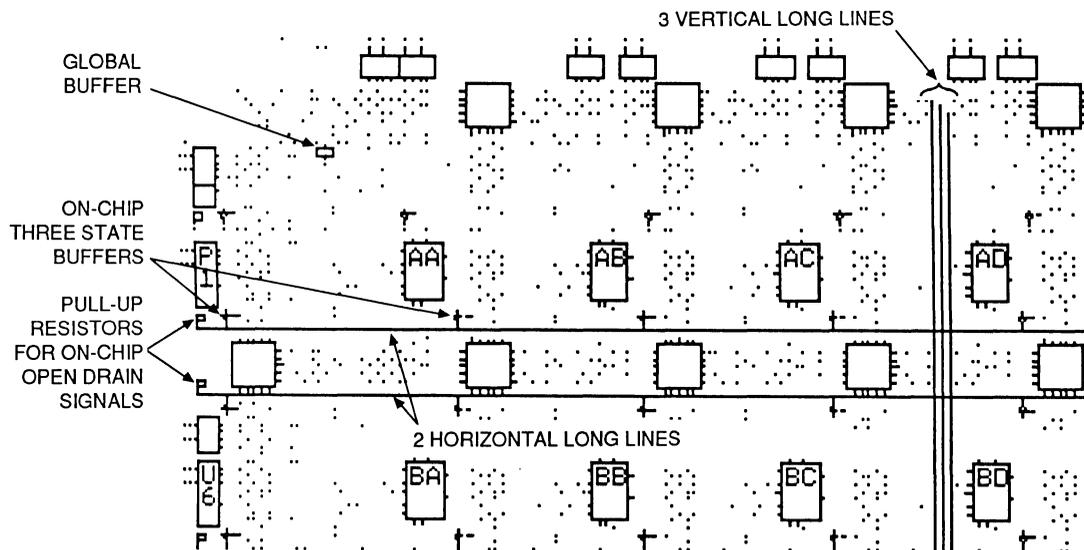
Long lines can be driven by a logic block or I/O block output on a column by column basis. This capability provides a common low skew control or clock line within each column of logic blocks. Interconnections of these long lines are shown in Figure 14. Isolation buffers are provided at each input to a long line and are enabled automatically by the development system when a connection is made.

A buffer in the upper left corner of the Logic Cell Array chip drives a global net which is available to all .k inputs of logic blocks. Using the global buffer for a clock signal provides a skew-free, high fan-out, synchronized clock for use at any or all of the I/O and logic blocks. Configuration bits for the .k input to each logic block can select this global line or another routing resource as the clock source for its flip-flops. This net may also be programmed to drive the die edge clock lines for I/O Block use. An enhanced speed, CMOS threshold, direct access to this buffer is available at the second pad from the top of the left die edge.

A buffer in the lower right corner of the array drives a horizontal long line that can drive programmed connections to a vertical long line in each interconnection column. This alternate buffer also has low skew and high fan-out. The network formed by this alternate buffer's long lines can be selected to drive the .k inputs of the logic blocks. CMOS threshold, high speed access to this buffer is available from the third pad from the bottom of the right die edge.

### Internal Busses

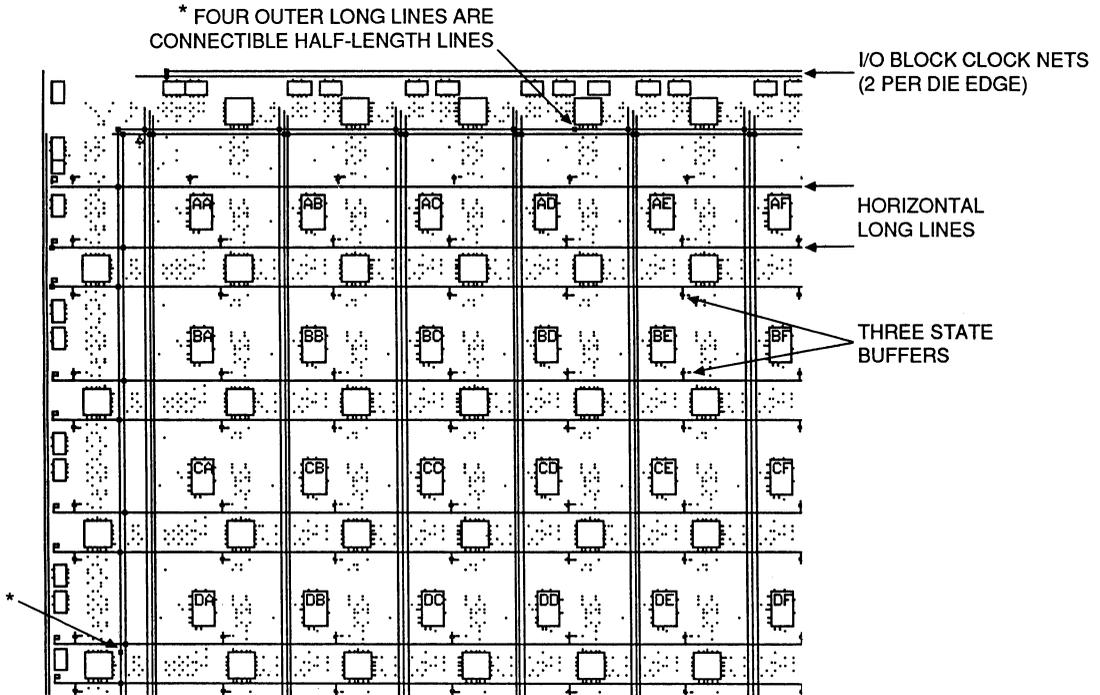
A pair of three-state buffers is located adjacent to each configurable logic block. These buffers allow logic to drive the horizontal long lines. Logical operation of the three-state buffer controls allows them to implement wide multiplexing functions. Any three-state buffer input can be selected as drive for the horizontal long line bus by applying a low logic level on its three-state control line. See Figure 15a. The user is required to avoid contention which can result from multiple drivers with opposing logic



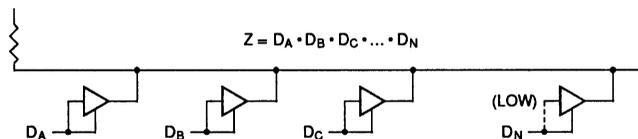
**Figure 13.** Horizontal and vertical long lines provide high fan-out, low-skew signal distribution in each row and column. The global buffer in the upper left die corner drives a common line throughout the LCA.

levels. Control of the three-state input by the same signal that drives the buffer input, creates an 'open drain' wired-AND function. A logical HIGH on both buffer inputs creates a high impedance which represents no contention. A logical LOW enables the buffer to drive the long line low. See Figure 15b. Pull-up resistors are available at each end

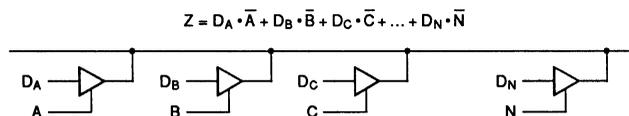
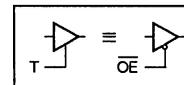
of the long line to provide a HIGH output when all connected buffers are non-conducting. This forms fast, wide gating functions. When data drives the inputs, and separate signals drive the three-state control lines, these buffers form multiplexers (three-state buses). In this case care must be used to prevent contention through multiple active



**Figure 14.** Programmable interconnection of long lines is provided at the edges of the routing area. Three-state buffers allow the use of horizontal long lines to form on-chip wired-AND and multiplexed buses. The left two vertical long lines per column (except 3020) and the outer perimeter long lines may be programmed as connectible half-length.



**Figure 15a.** Three-state buffers implement a Multiplexer where the selection is accomplished by the buffer three-state signal.



**Figure 15b.** Three-state buffers implement a Wired-AND function. When all the buffer three-state lines are HIGH, (high impedance), the pull-up resistor(s) provide the HIGH output. The buffer inputs are driven by the control signals or a LOW.

buffers of conflicting levels on a common line. Figure 16 shows three state buffers, long lines and pull-up resistors.

### Crystal Oscillator

Figure 16 also shows the location of an internal high speed inverting amplifier which may be used to implement an on-chip crystal oscillator. It is associated with the auxiliary buffer in the lower right corner of the die. When the oscillator is configured by "MAKEBITS" and connected as a signal source, two special user I/O Blocks are also configured to connect the oscillator amplifier with external crystal oscillator components as shown in Figure 17. A divide by two option is available to assure symmetry. The oscillator circuit becomes active before configuration is

complete in order to allow the oscillator to stabilize. Actual internal connection is delayed until completion of configuration. In Figure 17 the feedback resistor, R1, between output and input biases the amplifier at threshold. The value should be as large as practical to minimize loading of the crystal. The inversion of the amplifier, together with the R-C networks and an AT cut series resonant crystal, produce the 360 degree phase shift of the Pierce oscillator. A series resistor, R2, may be included to add to the amplifier output impedance when needed for phase shift control, crystal resistance matching, or to limit the amplifier input swing to control clipping at large amplitudes. Excess feedback voltage may be corrected by the ratio of C2/C1. The amplifier is designed to be used from 1 MHz to one-half the specified CLB toggle frequency. Use at frequen-

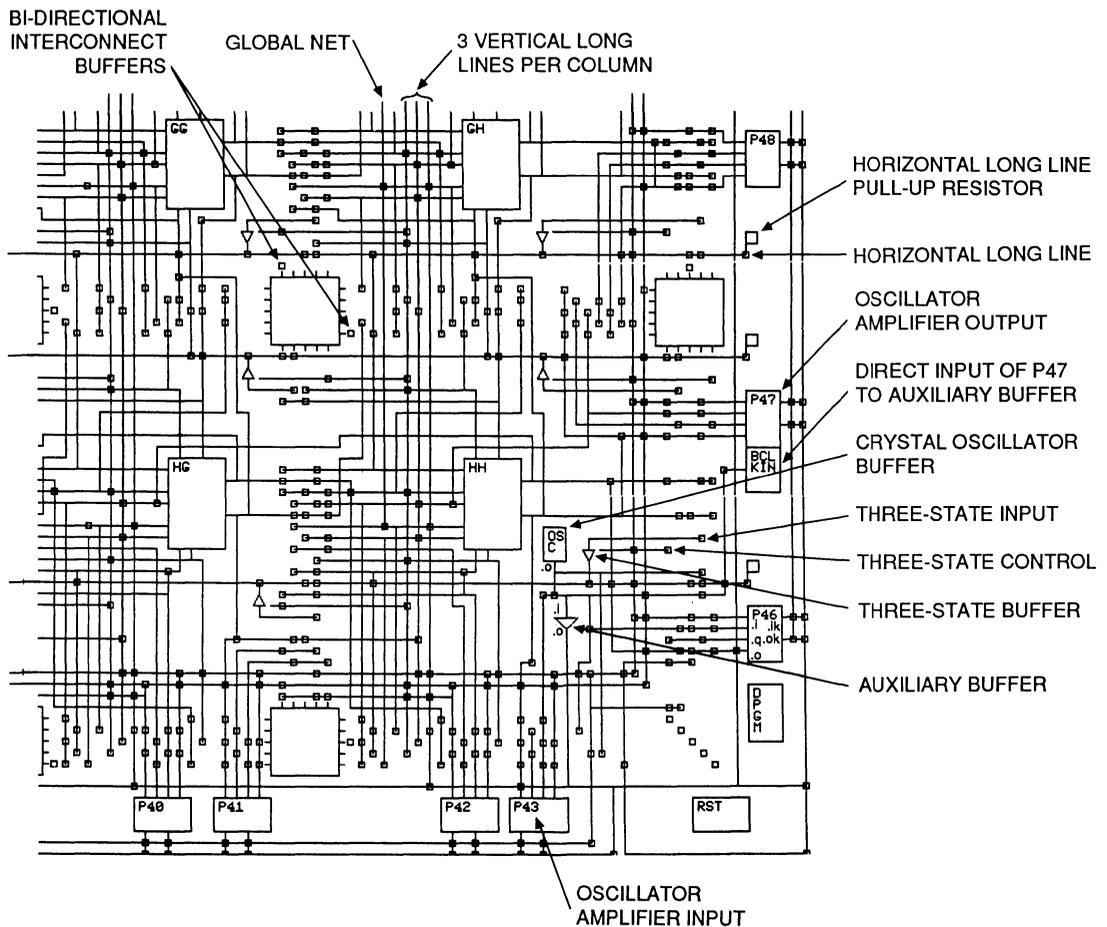


Figure 16. An XACT Development System extra large view of possible interconnections in the lower right corner of the XC3020.

cies below 1 MHz may require individual characterization with respect to a series resistance. Crystal oscillators above 20 MHz generally require a crystal which operates in a third overtone mode, where the fundamental frequency must be suppressed by the R-C networks. When the oscillator inverter is not used, these I/O Blocks and their package pins are available for general user I/O.

**PROGRAMMING**

**Initialization Phase**

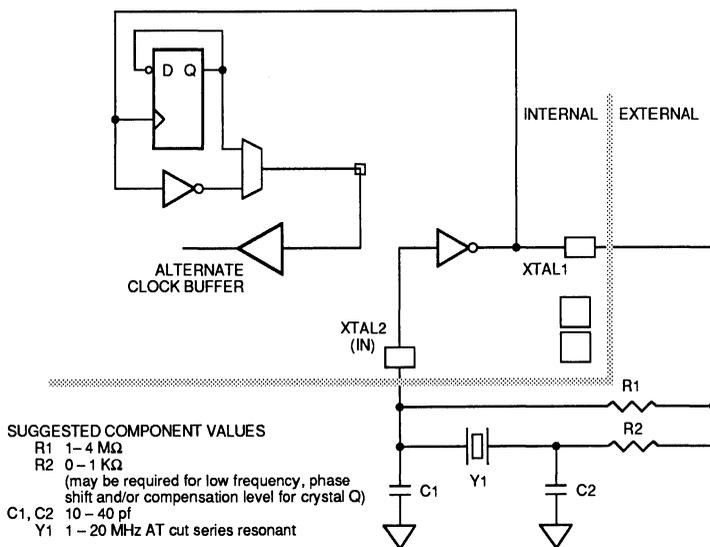
An internal power-on-reset circuit is triggered when power is applied. When Vcc reaches the voltage at which portions of the LCA begin to operate (2.5 to 3 Volts), the programmable I/O output buffers are disabled and a high impedance pull-up resistor is provided for the user I/O pins. A time-out delay is initiated to allow the power supply voltage to stabilize. During this time the power-down mode is inhibited. The Initialization state time-out (about 11 to 33 ms) is determined by a 14-bit counter driven by a self-generated, internal timer. This nominal 1 MHz timer is subject to variations with process, temperature and power supply over the range of 0.5 to 1.5 MHz. As shown in Table 1, five configuration mode choices are available as

determined by the input levels of three mode pins; M0, M1 and M2.

In Master configuration modes the LCA becomes the source of Configuration Clock (CCLK). The beginning of configuration of devices using Peripheral or Slave modes must be delayed long enough for their initialization to be completed. An LCA with mode lines selecting a Master configuration mode extends its initialization state using four times the delay (43 to 130 ms) to assure that all daisy-chained slave devices which it may be driving will be ready even if the master is very fast, and the slave(s) very slow. Figure 18 shows the state sequences. At the end of initiali-

**Table 1**

M0	M1	M2	Clock	Mode	Data
0	0	0	active	Master	Bit Serial
0	0	1	active	Master	Byte Wide Addr. = 0000 up
0	1	0	—	reserved	—
0	1	1	active	Master	Byte Wide Addr. = FFFF down
1	0	0	—	reserved	—
1	0	1	passive	Peripheral	Byte Wide
1	1	0	—	reserved	—
1	1	1	passive	Slave	Bit Serial



**SUGGESTED COMPONENT VALUES**  
 R1 1 – 4 MΩ  
 R2 0 – 1 KΩ  
 (may be required for low frequency, phase shift and/or compensation level for crystal Q)  
 C1, C2 10 – 40 pF  
 Y1 1 – 20 MHz AT cut series resonant

	68 PIN	84 PIN	132 PIN	175 PIN
	PLCC	PLCC	PGA	PGA
XTAL 1 (OUT)	47	57	J11	P13
XTAL 2 (IN)	43	53	L11	P15

1105 14

**Figure 17.** When activated in the "MAKEBITS" program and by selecting an output network for its buffer, the crystal oscillator inverter uses two unconfigured package pins and external components to implement an oscillator. An optional divide-by-two mode is available to assure symmetry.

zation the LCA enters the Clear state where it clears the configuration memory. The active low, open-drain initialization signal  $\overline{\text{INIT}}$  indicates when the Initialization and Clear states are complete. The LCA tests for the absence of an external active low  $\overline{\text{RESET}}$  before it makes a final sample of the mode lines and enters the Configuration state. An external wired-AND of one or more  $\overline{\text{INIT}}$  pins can be used to control configuration by the assertion of the active low  $\overline{\text{RESET}}$  of a master mode device or to signal a processor that the LCAs are not yet initialized.

If a configuration has begun, a re-assertion of  $\overline{\text{RESET}}$  for a minimum of three internal timer cycles will be recognized and the LCA will initiate an abort, returning to the Clear state to clear the partially loaded configuration memory words. The LCA will then re-sample  $\overline{\text{RESET}}$  and the mode lines before re-entering the Configuration state. A re-program is initiated when a configured LCA senses a HIGH to LOW transition on the  $\overline{\text{DONE/PROG}}$  package pin. The LCA returns to the Clear state where the configuration memory is cleared and mode lines re-sampled, as for an aborted configuration. The complete configuration program is cleared and loaded during each configuration program cycle.

Length count control allows a system of multiple Logic Cell Arrays, of assorted sizes, to begin operation in a synchronized fashion. The configuration program generated by the MakePROM program of the XACT development system begins with a preamble of 11111110010 followed by a 24-bit 'length count' representing the total number of configuration clocks needed to complete loading of the configuration program(s). The data framing is shown in Figure 19. All LCAs connected in series read and shift

preamble and length count in on positive and out on negative configuration clock edges. An LCA which has received the preamble and length count then presents a HIGH Data Out until it has intercepted the appropriate number of data frames. When the configuration program memory of an LCA is full and the length count does not compare, the LCA shifts any additional data through, as it did for preamble and length count.

When the LCA configuration memory is full and the length count compares, the LCA will execute a synchronous start-up sequence and become operational. See Figure 20. Three CCLK cycles after the completion of loading configuration data the user I/O pins are enabled as configured. As selected in MAKEBITS, the internal user-logic reset is released either one clock cycle before or after the I/O pins become active. A similar timing selection is programmable for the  $\overline{\text{DONE/PROG}}$  output signal.  $\overline{\text{DONE/PROG}}$  may also be programmed to be an open drain or include a pull-up resistor to accommodate wired ANDing. The High During Configuration (HDC) and Low During Configuration (LDC) are two user I/O pins which are driven active when an LCA is in its Initialization, Clear or Configure states. They and  $\overline{\text{DONE/PROG}}$  provide signals for control of external logic signals such as reset, bus enable or PROM enable during configuration. For parallel Master configuration modes these signals provide PROM enable control and allow the data pins to be shared with user logic signals.

User I/O inputs can be programmed to be either TTL or CMOS compatible thresholds. At power-up, all inputs have TTL thresholds and can change to CMOS thresholds

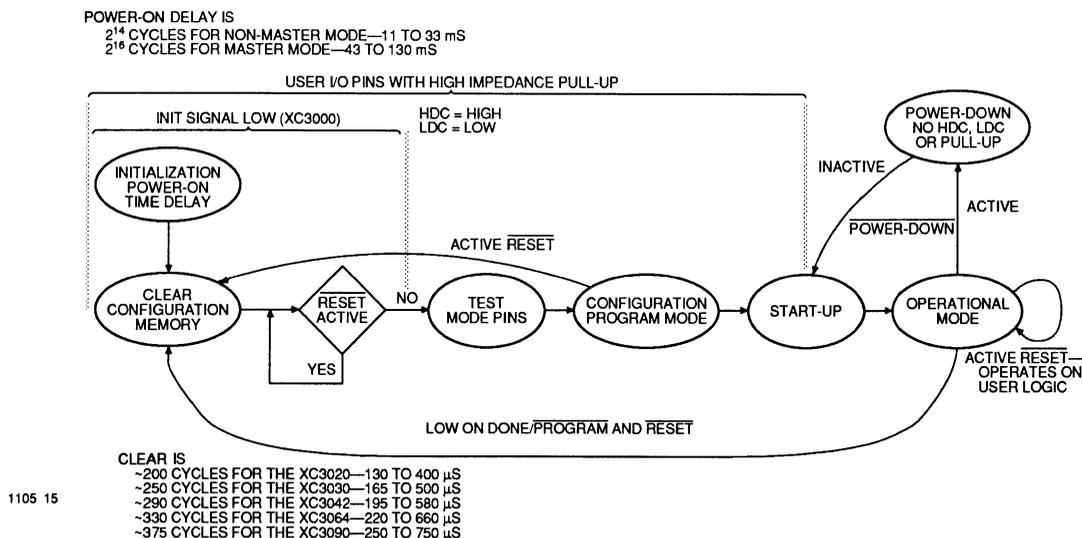


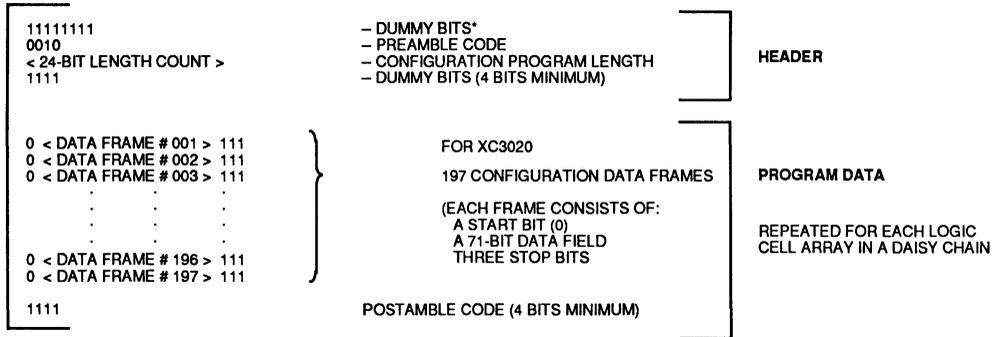
Figure 18. A state diagram of the configuration process for power-up and reprogram.

at the completion of configuration if the user has selected CMOS thresholds. The threshold of PWRDWN and the direct clock inputs are fixed at a CMOS level.

If the crystal oscillator is used it will begin operation before configuration is complete to allow time for stabilization before it is connected to the internal circuitry.

**Configuration Data**

Configuration data to define the function and interconnection within a Logic Cell Array are loaded from an external storage at power-up and on a re-program signal. Several methods of automatic and controlled loading of the required data are available. Logic levels applied to mode se-



\*THE LCA DEVICES REQUIRE 4 DUMMY BITS MIN., XACT 2.10 GENERATES 8 DUMMY BITS

1105 05

Device	XC3020	XC3030	XC3042	XC3064	XC3090
Gates	2000	3000	4200	6400	9000
CLBs	64	100	144	224	320
Row X Col	(8 X 8)	(10 X 10)	(12 X 12)	(16 X 14)	(20 X 16)
IOBs	64	80	96	120	144
Flip-flops	256	360	480	688	928
Bits per frame (w/ 1 start 3 stop)	75	92	108	140	172
Frames	197	241	285	329	373
Program Data = Bits * Frames + 4 (excludes header)	14779	22176	30784	46064	64160
PROM size (bits) = Program Data + 40 bit Headers	14819	22216	30824	46104	64200

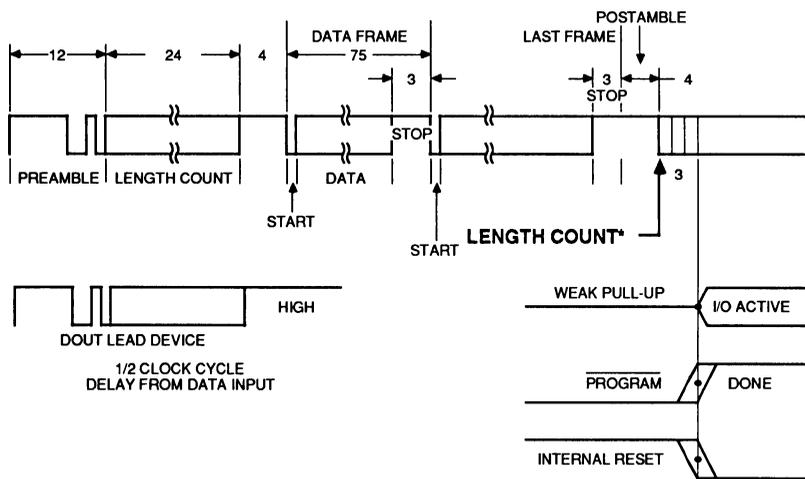
**Figure 19.** The internal Configuration Data Structure for an LCA shows the preamble, length count and data frames which are generated by the XACT Development System.

The Length Count produced by the "MAKEBIT" program = [(40-bit preamble + sum of program data + 1 per daisy chain device) rounded up to multiple of 8] - (2 ≤ K ≤ 4) where K is a function of DONE and RESET timing selected. An additional 8 is added if roundup increment is less than K. K additional clocks are needed to complete start-up after length count is reached.

lection pins at the start of configuration time determine the method to be used. See Table 1. The data may be either bit-serial or byte-parallel, depending on the configuration mode. Various Xilinx Programmable Gate Arrays have different sizes and numbers of data frames. To maintain compatibility between various device types, the Xilinx 2000 and 3000 product families use compatible configuration formats. For the XC3020, configuration requires 14779 bits for each device, arranged in 197 data frames. An additional 40 bits are used in the header. See Figure 20. The specific data format for each device is produced by the MAKEBITS command of the development system and one or more of these files can then be combined and appended to a length count preamble and be transformed into a PROM format file by the 'MAKE PROM' command of the XACT development system. A compatibility exception precludes the use of a 2000 series device as the master for 3000 series devices if their DONE or RESET are programmed to occur after their outputs become active. The "tie" option of the MAKEBITS program defines output levels of unused blocks of a design and connects these to unused routing resources. This prevents indeterminant levels which might produce parasitic

supply currents. If unused blocks are not sufficient to complete the 'tie,' the FLAGNET command of EDITLCA can be used to indicate nets which must not be used to drive the remaining unused routing, as that might affect timing of user nets. NORESTORE will retain the results of TIE for timing analysis with QUERYNET before RESTORE returns the design to the untied condition. TIE can be omitted for quick breadboard iterations where a few additional mA of Icc are acceptable.

The configuration bit-stream begins with HIGH preamble bits, a four-bit preamble code and a 24-bit length count. When configuration is initiated, a counter in the LCA is set to 0 and begins to count the total number of configuration clock cycles applied to the device. As each configuration data frame is supplied to the LCA, it is internally assembled into a data word. As each data word is completely assembled, it is loaded in parallel into one word of the internal configuration memory array. The configuration loading process is complete when the current length count equals the loaded length count and the required configuration program data frames have been written. Internal user flip-flops are held reset during configuration.



\* THE CONFIGURATION DATA CONSISTS OF A COMPOSITE 40-BIT PREAMBLE/LENGTH-COUNT, FOLLOWED BY ONE OR MORE CONCATENATED LCA PROGRAMS, SEPARATED BY 4-BIT POSTAMBLES. AN ADDITIONAL FINAL POSTAMBLE BIT IS ADDED FOR EACH SLAVE DEVICE AND THE RESULT ROUNDED UP TO A BYTE BOUNDARY. THE LENGTH COUNT IS TWO LESS THAN THE NUMBER OF RESULTING BITS.

TIMING OF THE ASSERTION OF DONE AND TERMINATION OF THE INTERNAL RESET MAY EACH BE PROGRAMMED TO OCCUR ONE CYCLE BEFORE OR AFTER THE I/O OUTPUTS BECOME ACTIVE.

1105 06

Figure 20. Configuration and start-up of one or more LCAs.

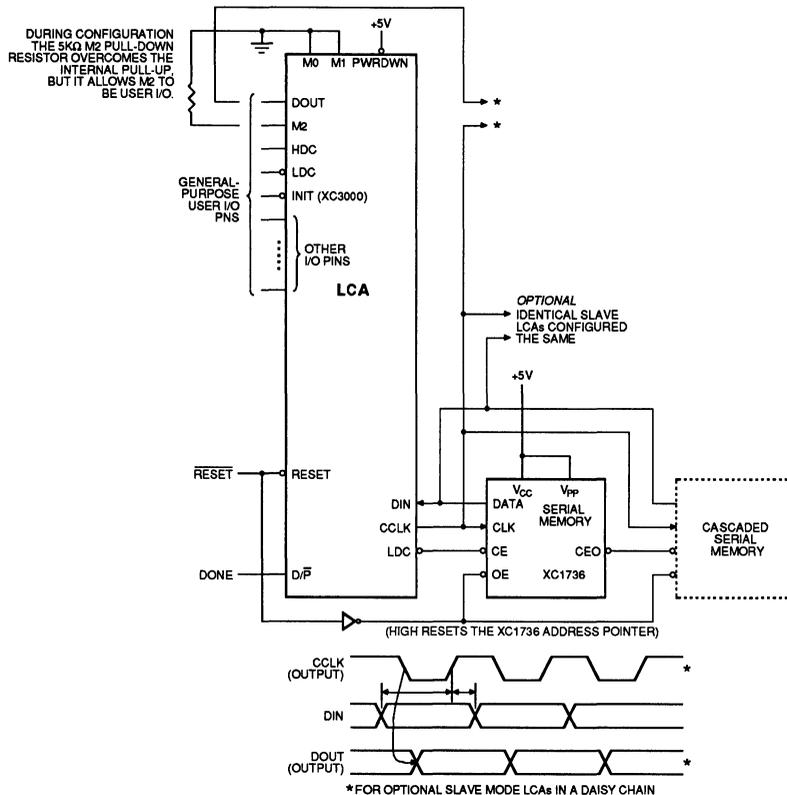
1105 06

Two user programmable pins are defined in the unconfigured Logic Cell array. High During Configuration (HDC) and Low During Configuration (LDC) as well as DONE/PROG may be used as external control signals during configuration. In Master mode configurations it is convenient to use LDC as an active-low EPROM Chip Enable. After the last configuration data-bit is loaded and the length count compares, the user I/O pins become active. Options in the MAKEBITS program allow timing choices of one clock earlier or later for the timing of the end of the internal logic reset and the assertion of the DONE signal. The open-drain DONE/PROG output can be AND-tied with multiple Logic Cell Arrays and used as an active high READY, an active low PROM enable or a RESET to

other portions of the system. The state diagram of Figure 18 illustrates the configuration process.

**Master Mode**

In Master mode, the Logic Cell Array automatically loads configuration data from an external memory device. There are three Master modes which use the internal timing source to supply the configuration clock (CCLK) to time the incoming data. Serial Master mode uses serial configuration data supplied to data-in (DIN) from a synchronous serial source such as the Xilinx Serial Configuration PROM shown in Figure 21. Parallel Master Low and Master High modes automatically use parallel data sup-



**Figure 21. Master Serial Mode.** The one-time-programmable XC1736 Serial Configuration PROM supports automatic loading of configuration programs up to 36K bits. Multiple devices can be cascaded to support additional LCAs. An early DONE inhibits the XC1736 data output a  $\overline{\text{CCLK}}$  cycle before the LCA I/O become active.



**Peripheral Mode**

Peripheral mode provides a simplified interface through which the device may be loaded byte-wide, as a processor peripheral. Figure 23 shows the peripheral mode connections. Processor write cycles are decoded from the common assertion of the active low Write Strobe ( $\overline{WRT}$ ), and two active low and one active high Chip Selects ( $\overline{CS0}$ ,  $\overline{CS1}$ ,  $CS2$ ). If all these signals are not available, the unused inputs should be driven to their respective active levels. The Logic Cell Array will accept one byte of data on the D0–D7 inputs for each selected processor Write cycle. Each byte of data is loaded into a buffer register. The LCA generates a configuration clock from the internal timing generator and serializes the parallel input data for internal framing or for succeeding slaves on Data Out (DOUT). A output HIGH on RDY/BUSY pin indicates the completion of loading for each byte when the input register is ready for a new byte. As with Master

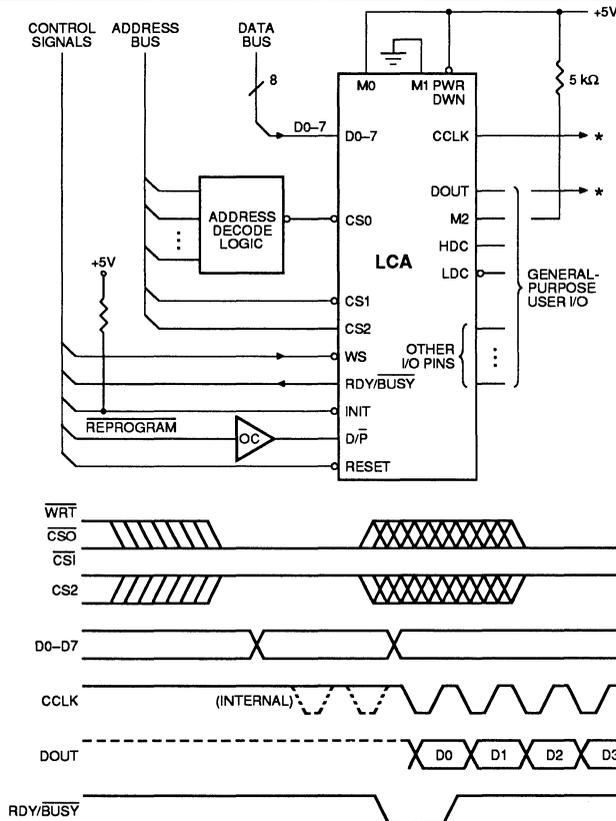
modes, Peripheral mode may also be used as a lead device for a daisy-chain of slave devices.

**Slave Mode**

Slave mode provides a simple interface for loading the Logic Cell Array configuration as shown in Figure 24. Serial data are supplied in conjunction with a synchronizing input clock. Most Slave mode applications are in daisy-chain configurations in which the data input are supplied by the previous Logic Cell Array's data out, while the clock is supplied by a lead device in Master or Peripheral mode. Data may also be supplied by a processor or other special circuits.

**Daisy-Chain**

The Xilinx XACT development system is used to create a composite configuration bit stream for selected LCAs



\* FOR OPTIONAL SLAVE MODE LCAs IN A DAISY CHAIN

**Figure 23. Peripheral Mode.** Configuration data are loaded using a byte-wide data bus from a microprocessor.

including: a preamble, a length count for the total bit-stream, multiple concatenated data programs and a postamble plus an additional fill bit per device in the serial chain. After loading and passing-on the preamble and length count to a possible daisy-chain, a lead device will load its configuration data frames while providing a HIGH DOUT to possible down-stream devices as shown in Figure 25. Loading continues while the lead device has received its configuration program and the current length count has not reached the full value. The additional data are passed through the lead device and appear on the Data Out (DOUT) pin in serial form. The lead device also generates the Configuration Clock (CCLK) to synchronize the serial output data and data in of down-stream LCAs. Data are read in on DIN of slave devices by the positive edge of CCLK and shifted out the DOUT on the negative edge of CCLK. A parallel Master mode device uses its internal timing generator to produce an internal CCLK of 8 times its EPROM address rate, while a Peripheral mode device produces a burst of 8 CCLKs for each chip select

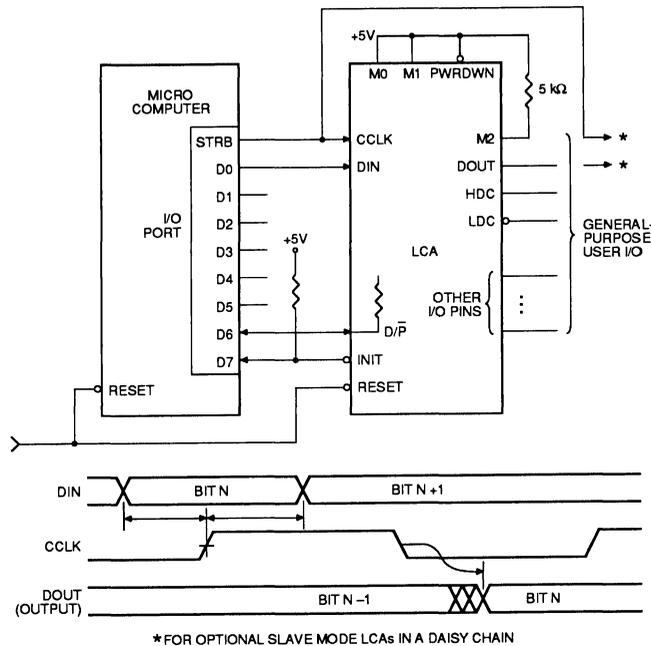
and write-strobe cycle. The internal timing generator continues to operate for general timing and synchronization of inputs in all modes.

### Special Configuration Functions

The configuration data include control over several special functions in addition to the normal user logic functions and interconnect:

- Input thresholds
- Readback enable
- DONE pull-up resistor
- DONE timing
- RESET timing
- Oscillator frequency divided by two

Each of these functions is controlled by configuration data bits which are selected as part of the normal XACT development system bit-stream generation process.



1105 19

**Figure 24.** Slave Mode. Bit-serial configuration data are read at rising edge of the CCLK. Data on DOUT are provided on the falling edge of CCLK.

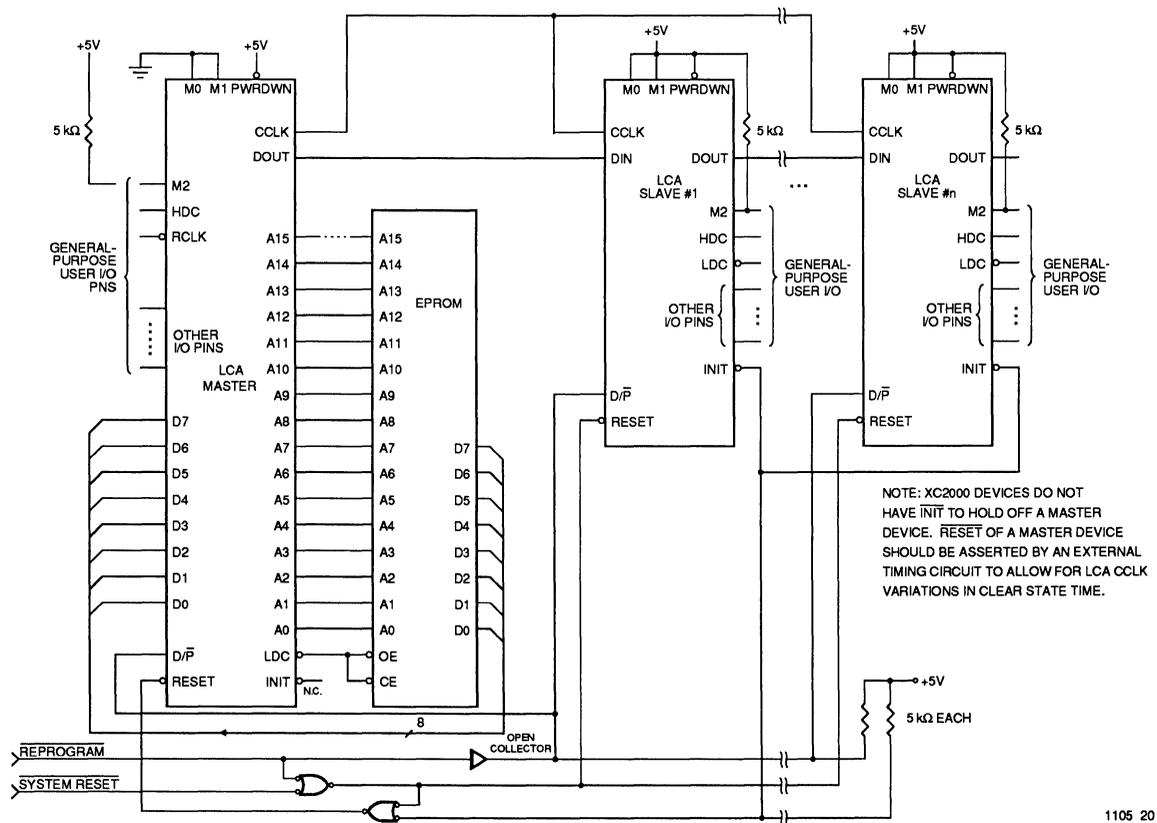
**Input Thresholds**

Prior to the completion of configuration all LCA input thresholds are TTL compatible. Upon completion of configuration the input thresholds become either TTL or CMOS compatible as programmed. The use of the TTL threshold option requires some additional supply current for threshold shifting. The exception is the threshold of the PWRDWN input and direct clocks which always have a CMOS input. Prior to the completion of configuration the user I/O pins each have a high impedance pull-up. The configuration program can be used to enable the I/O Block pull-up resistors in the Operational mode to act either as an input load or to avoid a floating input on an otherwise unused pin.

**Readback**

The contents of a Logic Cell Array may be read back if it has been programmed with a bit-stream in which the Readback option has been enabled. Readback may be used for verification of configuration and as a method of determining the state of internal logic nodes during debugging with the XACTOR In-Circuit debugger. There are three options in generating the configuration bit-stream:

- "Never" will inhibit the Readback capability.
- "One-time," will inhibit Readback after one Readback has been executed to verify the configuration.
- "On-command" will allow unrestricted use of Readback.



**Figure 25. Master Mode configuration with daisy chained slave mode devices.**  
 All are configured from the common EPROM source. The Slave mode device **INIT** signals delay the Master device configuration until they are initialized. A well defined termination of **SYSTEM RESET** is needed when controlling multiple LCAs.

Any XC3000 slave driven by an XC2000 master mode device must use "early DONE and early internal reset".  
 (The XC2000 master will not supply the extra clock required by a "late" programmed XC3000.)

Readback is accomplished without the use of any of the user I/O pins; only M0, M1 and CCLK are used. The initiation of readback is produced by a LOW to HIGH transition of the M0/RTRIG (Read Trigger) pin. Once the readback command has been given, the input CCLK is driven by external logic to read back each data bit in a format similar to loading. After two dummy bits, the first data frame is shifted out, in inverted sense, on the M1/RDATA (Read Data) pin. All data frames must be read back to complete the process and return the mode select and CCLK pins to their normal functions.

The readback data includes the current state of each internal logic block storage element, and the state of the [./ and .r] connection pins on each I/O Block. These data are imbedded into unused configuration bit positions during readback. This state information is used by the Logic Cell Array development system In-Circuit Verifier to provide visibility into the internal operation of the logic while the system is operating. To readback a uniform time-sample of all storage elements it may be necessary to inhibit the system clock.

### Re-program

The Logic Cell Array configuration memory can re-written while the device is operating in the user's system. To initiate a re-programming cycle, the dual function package pin  $\text{DONE}/\overline{\text{PROG}}$  must be given a HIGH to LOW transition. To reduce sensitivity to noise, the input signal is filtered for 2 cycles of the LCA's internal timing generator. When re-program begins the user programmable I/O output buffers are disabled and high impedance pull-ups are provided for the package pins. The device returns to the Clear state and clears the configuration memory before it indicates 'initialized'. Reprogram control is often implemented using an external open collector driver which pulls  $\text{DONE}/\overline{\text{PROG}}$  LOW. Once it recognizes a stable request, the Logic Cell Array will hold a LOW until the new configuration has been completed. Even if the re-program request is externally held LOW beyond the configuration period, the Logic Cell

Array will begin operation upon completion of configuration.

### DONE Pull-up

$\text{DONE}/\overline{\text{PROG}}$  is an open drain I/O pin that indicates the LCA is in the operational state. An optional internal pull-up resistor can be enabled by the user of the XACT development system when 'Make Bits' is executed. The  $\text{DONE}/\overline{\text{PROG}}$  pins of multiple LCAs in a daisy-chain may be connected together to indicate all are DONE or to direct them all to re-program.

### DONE Timing

The timing of the DONE status signal can be controlled by a selection in the MAKEBITS program to occur a CCLK cycle before, or after, the timing of outputs being activated. See Figure 20. This facilitates control of external functions such as a PROM enable or holding a system in a wait state.

### RESET Timing

As with DONE timing, the timing of the release of the internal RESET can be controlled by a selection in the MAKEBITS program to occur a CCLK cycle before, or after, the timing of outputs being enabled. See Figure 20. This reset maintains all user programmable flip-flops and latches in a 'zero' state during configuration.

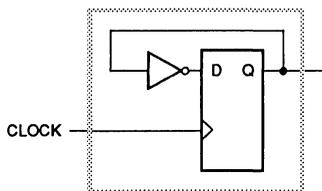
### Crystal Oscillator Division

A selection in the MAKEBITS program allows the user to incorporate a dedicated divide-by-two flip-flop in the crystal oscillator function. This provides higher assurance of a symmetrical timing signal. Although the frequency stability of crystal oscillators is high, the symmetry of the waveform may be affected by bias or feedback drive.

## PERFORMANCE

### Device Performance

The high performance of the Logic Cell Array is due in part to the manufacturing process, which is similar to that used for high speed CMOS static memories. Performance can be measured in terms of minimum propagation times for logic elements. The parameter which traditionally describes the overall performance of a gate array is the toggle frequency of a flip-flop. The configuration for determining the toggle performance of the Logic Cell Array is shown in Figure 26. The flip-flop output Q is fed back through the combinatorial logic as  $\bar{Q}$  to form the toggle flip-flop.



1105 07

**Figure 26.** "Toggle" Flip-Flop used to characterize device performance.

Actual Logic Cell Array performance is determined by the timing of critical paths, including both the fixed timing for the logic and storage elements in that path, and the timing associated with the routing of the network. Examples of internal worst case timing are included in the performance data to allow the user to make the best use of the capabilities of the device. The XACT development system timing calculator or XACT generated simulation models should be used to calculate worst case paths by using actual impedance and loading information. Figure 27 shows a variety of elements which are involved in determining system performance. Actual measurement of internal timing is not practical and often only the sum of component timing is relevant as in the case of input to output. The relationship between input and output timing is arbitrary and only the total determines performance. Timing components of internal functions may be determined by measurement of differences at the pins of the package. A synchronous logic function which involves a clock to block-output, and a block-input to clock set-up is capable of higher speed operation than a logic configuration of two synchronous blocks with an extra combinatorial block level between them. System clock rates to 60% of the toggle frequency are practical for logic in which an extra combinatorial level is located between synchronized blocks. This allows implementation of functions of up to 25 variables. The use of the wired-AND is also available for wide, high speed functions.

### **Logic Block Performance**

Logic block performance is expressed as the propagation time from the interconnect point at the input of the combinatorial logic to the output of the block in the interconnect area. Combinatorial performance is independent of the specific logic function because of the table look-up based implementation. Timing is different when the combinatorial logic is used in conjunction with the storage element. For the combinatorial logic function driving the data input of the storage element, the critical timing is data set-up relative to the clock edge provided to the flip-flop element. The delay from the clock source to the output of the logic block is critical in the timing of signals produced by storage elements. Loading of a Logic Block output is limited only by the resulting propagation delay of the larger interconnect network. Speed performance of the logic block is a function of supply voltage and temperature. See Figures 28 and 29.

### **Interconnect Performance**

Interconnect performance depends on the routing resource used to implement the signal path. As discussed earlier, direct interconnect from block to block provides a

fast path for a signal. The single metal segment used for Long lines exhibits low resistance from end to end, but relatively high capacitance. Signals driven through a programmable switch will have the additional impedance of the switch added to their normal drive impedance.

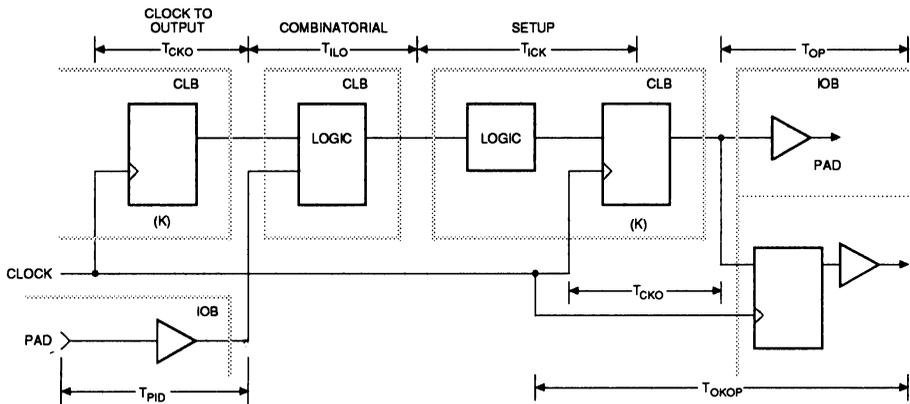
General purpose interconnect performance timing depends on the number of switches and segments used, the presence of the bi-directional re-powering buffers and the overall loading on the signal path at all points along the path. In calculating the worst case timing for a general interconnect path the timing calculator portion of the XACT development system accounts for all of these elements. As an approximation, interconnect timing is proportional to the summation of totals of local metal segments beyond each programmable switch. In effect, the time is a sum of R-C time each approximated by an R times the total C it drives. The R of the switch and the C of the interconnect is a function of the particular device performance grade. For a string of three local interconnects, the approximate time at the first segment, after the first switch resistance would be three units; an additional two units after the next switch plus an additional unit after the last switch in the chain. The interconnect R-C chain terminates at each re-powering buffer. The capacitance of the actual block inputs is not significant; the capacitance is in the interconnect metal and switches. Figure 30 illustrates this.

## **POWER**

### **Power Distribution**

Power for the LCA is distributed through a grid to achieve high noise immunity and isolation between logic and I/O. Inside the LCA, a dedicated Vcc and ground ring surrounding the logic array provides power to the I/O drivers. See Figure 31. An independent matrix of Vcc and ground lines supplies the interior logic of the device. This power distribution grid provides a stable supply and ground for all internal logic, providing the external package power pins are all connected and appropriately decoupled. Typically a 0.1  $\mu$ F capacitor connected near the Vcc and ground pins of the package will provide adequate decoupling.

Output buffers capable of driving the specified 4 mA loads under worst-case conditions may be capable of driving 25 to 30 times that current in a best case. Noise can be reduced by minimizing external load capacitance and reducing simultaneous output transitions in the same direction. It may also be beneficial to locate heavily loaded output buffers near the ground pads. The I/O Block output buffers have a slew limited mode which should be used where output rise and fall times are not speed critical.



1105 21

		Speed Grade	-50		-70			Units
	Description	Symbol	Min	Max	Min	Max		
Logic input to Output	Combinatorial	$T_{ILO}$		14		9		ns
K Clock	To output	$T_{CKO}$		12		8		ns
	Logic-input setup	$T_{ICK}$	12		8			ns
	Logic-input hold	$T_{CKI}$	0		0			ns
Input/Output	Pad to input (direct)	$T_{PID}$		10		7		ns
	Output to pad (enabled)	$T_{OP}$		14		10		ns
	I/O clock to pad	$T_{OKPO}$		18		13		ns
FF toggle frequency		$F_{CLK}$		50		70		MHz

**Figure 27. Examples of Primary Block Speed Factors.**

Actual timing is a function of various block factors combined with routing factors.

Overall performance can be evaluated with the XACT timing calculator or by an optional simulation.

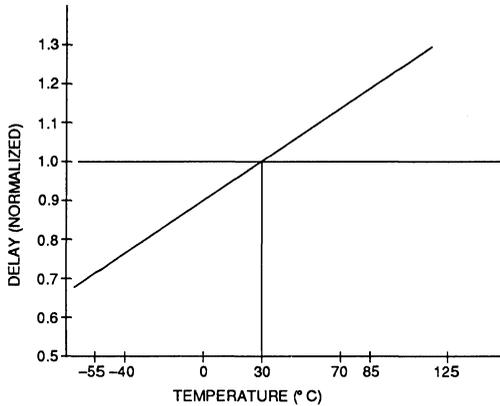
Slew-limited outputs maintain their DC drive capability, but generate less external reflections and internal noise. More than 32 fast outputs should not be switching in the same direction exactly simultaneously. A few ns of deliberate skew can alleviate this problem of "ground-bounce".

**Power Dissipation**

The Logic Cell Array exhibits the low power consumption characteristic of CMOS ICs. For any design the user can

use Figure 32 to calculate the total power requirement based on the sum of the capacitive and DC loads both external and internal. The configuration option of TTL chip input threshold requires power for the threshold reference. The power required by the static memory cells which hold the configuration data is very low and may be maintained in a power-down mode.

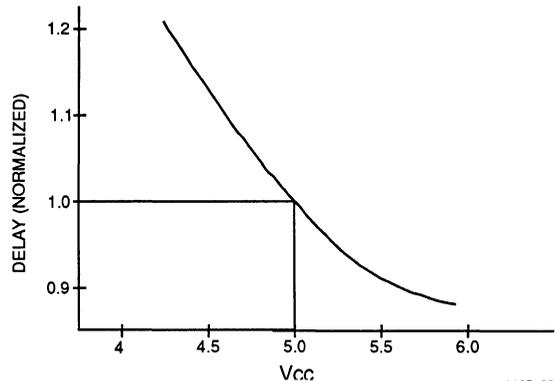
Typically most of power dissipation is produced by external capacitive loads on the output buffers. This load and frequency dependent power is  $25 \mu\text{W}/\text{pF}/\text{MHz}$  per output. Another component of I/O power is the DC loading on each output pin by devices driven by the Logic Cell Array.



NOTE: NORMALIZED FOR FOUR TEMPERATURES

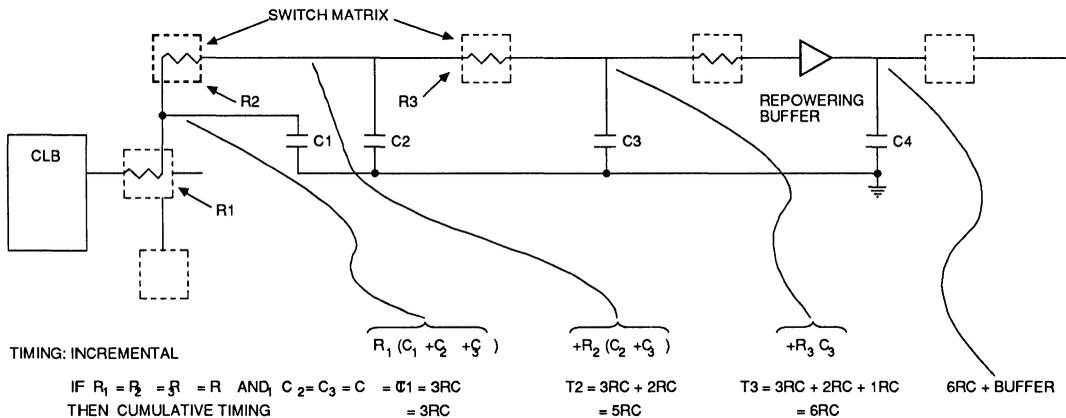
1105 08

**Figure 28.** Change in speed performance as a function of temperature, normalized for 30°C.



1105 22

**Figure 29.** The speed performance of a CMOS device increases with Vcc within the operating range.



**Figure 30.** Interconnection timing example. Use of the XACT timing calculator or XACT-generated simulation model provide actual worst-case performance information.

1105 23

Internal power dissipation is a function of the number and size of the nodes, and the frequency at which they change. In an LCA the fraction of nodes changing on a given clock is typically low (10–20%). For example, in a large binary counter, the average clock cycle produces changes equal to one CLB output at the clock frequency. Typical global clock buffer power is between 1.7 mW/MHz for the XC3020 and 3.6 mW/MHz for the XC3090. The internal capacitive load is more a function of interconnect than fan-out. With a “typical” load of three general interconnect segments, each Configurable Logic Block output requires about 0.4 mW per MHz of its output frequency.

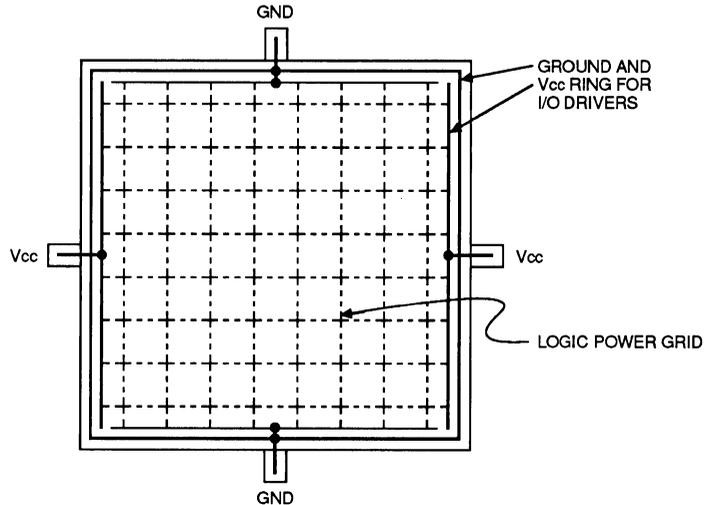
$$\text{Total Power} = V_{cc} \cdot I_{cco} + \text{external (DC + capacitive)} \\ + \text{internal (CLB + IOB + Long Line + pull-up)}$$

Because the control storage of the Logic Cell Array is CMOS static memory, its cells require a very low standby current for data retention. In some systems, this low data retention current characteristic can be used as a method of preserving configurations in the event of a primary power loss. The Logic Cell Array has built in power-down logic which, when activated, will disable normal operation of the device and retain only the configuration data. All internal operation is suspended and output buffers are placed in their high impedance state with no pull-ups.

Power-down data retention is possible with a simple battery-backup circuit because the power requirement is extremely low. For retention at 2.4 volts the required current is typically on the order of 50 nanoamps.

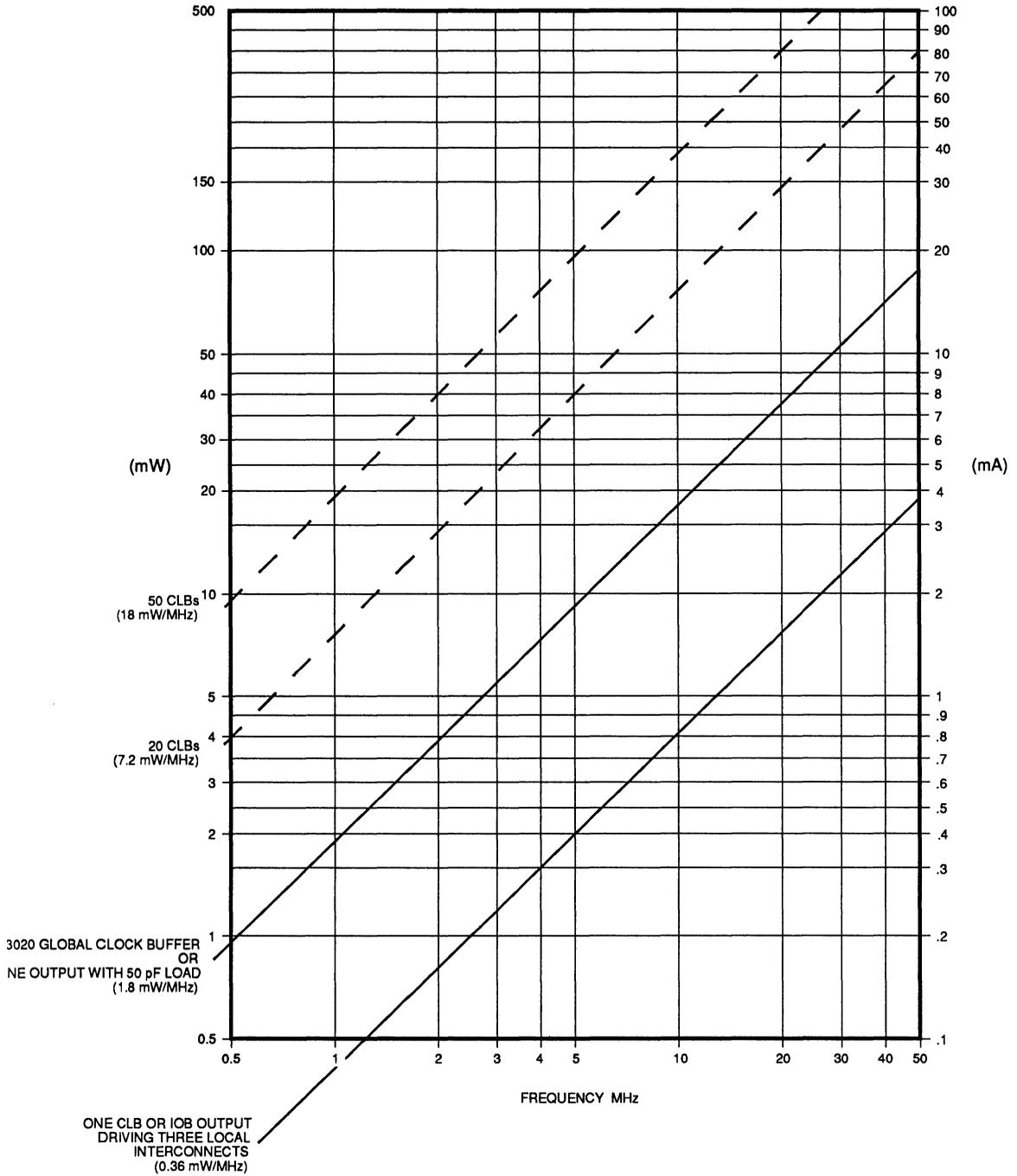
To force the Logic Cell Array into the Power-Down state, the user must pull the  $\overline{\text{PWRDWN}}$  pin low and continue to supply a retention voltage to the  $V_{cc}$  pins of the package. When normal power is restored,  $V_{cc}$  is elevated to its normal operating voltage and  $\overline{\text{PWRDWN}}$  is returned to a HIGH. The Logic Cell Array resumes operation with the same internal sequence that occurs at the conclusion of configuration. Internal I/O and logic block storage elements will be reset, the outputs will become enabled and the  $\overline{\text{DONE/PROG}}$  pin will be released. No configuration programming is involved.

When the power supply is removed from a CMOS device it is possible to supply some power from an input signal. The conventional electro-static input protection is implemented with diodes to the supply and ground. A positive voltage applied to an input (or output) will cause the positive protection diode to conduct and drive the power pin. This condition can produce invalid power conditions and should be avoided. A large series resistor might be used to limit the current or a bi-polar buffer may be used to isolate the input signal.



1105 24

Figure 31. LCA Power Distribution.



1105 09

**Figure 32. LCA Power Consumption by Element.** Total chip power is the sum of  $V_{cc} \cdot I_{cco}$  plus effective internal and external values of frequency dependent capacitive charging currents and duty factor dependent resistive loads.

## DEVELOPMENT SYSTEMS

To accomplish hardware development support for the Logic Cell Array, Xilinx provides a development system with several options to support added capabilities. The XACT system provides the following:

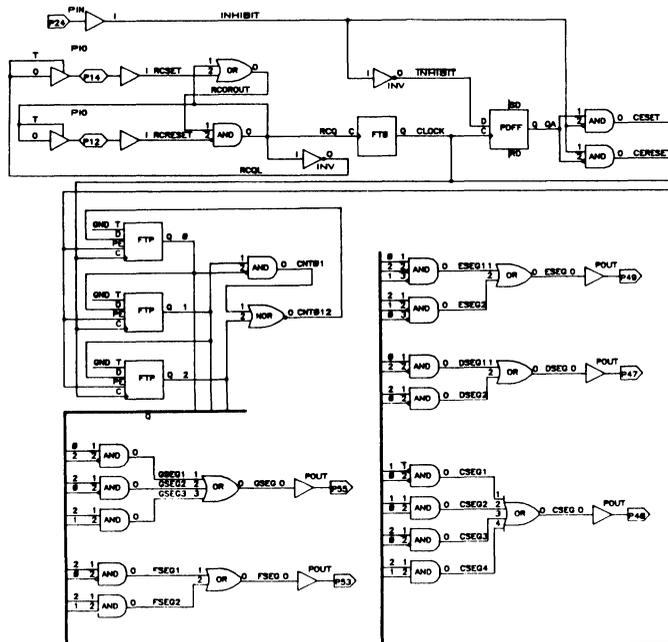
- Schematic entry
- Automatic place and route
- Interactive design editing for optimization
- Interactive timing calculations
- Macro library support, both for standard Xilinx supplied functions and user defined functions
- Design entry checking for consistency and completeness
- Automatic design documentation generation
- PROM programmer format output capabilities
- Simulation interface support including automatic netlist (circuit description) and timing extraction
- Logic and timing simulation
- In-circuit design verification for multiple devices

The host system on which the XACT system operates is an IBM PC/AT or compatible system with DOS 3.0 or higher. The system requires 640K bytes of internal RAM, 1.5 to 5.5 Mbyte of Extended Memory, color graphics and a mouse. A complete system requires one parallel I/O port and two serial ports for the mouse and in-circuit emulation.

## Designing with the XACT Development System

Designing with the Logic Cell Array is similar to using conventional MSI elements or gate array cells. A range of supported packages, including FutureNet and Schema, provide schematic capture with elements from a macro library. The XACT development system then translates the schematic description into partitioned Logic Blocks and I/O Blocks, based on shared input variables or efficient use of flip-flop and combinatorial logic. Design entry can also be implemented directly with the XACT development system using an interactive graphic design editor. The design information includes both the functional specifications for each block and a definition of the interconnection networks. Automatic placement and routing is available for either method of design entry. After routing the interconnections, various checking stages and processing of that data are performed to insure that the design is correct. Design changes may be implemented in minutes. The design file is used to generate the programming data which can be downloaded directly into an LCA in the user's target system and operated. The program information may be used to program PROM, EPROM or ROM devices, or stored in some other media as needed by the final system.

Design verification may be accomplished by using the XILINX XACTOR In-Circuit Design Verifier directly in the target system and/or the P-SILOS logic simulator.



Partial Sample—Schematic Capture

## PIN DESCRIPTIONS

### 1. Permanently Dedicated Pins.

#### Vcc

Two to eight (depending on package type) connections to the nominal +5 V supply voltage. All must be connected.

---

#### GND

Two to eight (depending on package type) connections to ground. All must be connected.

---

#### PWRDWN

A LOW on this CMOS compatible input stops all internal activity to minimize Vcc power, and puts all output buffers in a high impedance state, but configuration is retained. When the PWRDWN pin returns HIGH, the device returns to operation with the same sequence of buffer enable and DONE/PROGRAM as at the completion of configuration. All internal storage elements are reset. If not used, PWRDWN must be tied to Vcc.

---

#### RESET

This is an active low input which has three functions.

Prior to the start of configuration, a LOW input will delay the start of the configuration process. An internal circuit senses the application of power and begins a minimal time-out cycle. When the time-out and RESET are complete, the levels of the "M" lines are sampled and configuration begins.

If RESET is asserted during a configuration, the LCA is re-initialized and will restart the configuration at the termination of RESET.

If RESET is asserted after configuration is complete it will provide an asynchronous reset of all IOB and CLB storage elements of the LCA.

---

#### CCLK

During configuration, Configuration Clock is an output of an LCA in Master mode or Peripheral mode. LCAs in Slave mode use it as a clock input. During a Readback operation it is a clock input for the configuration data being shifted out.

#### DONE

The DONE output is configurable as open drain with or without an internal pull-up resistor. At the completion of configuration, the circuitry of the LCA becomes active in a synchronous order, and DONE may be programmed to occur one cycle before or after that.

#### PROG

Once configuration is done, a HIGH to LOW transition of this pin will cause an initialization of the LCA and start a reconfiguration.

---

#### M0

As Mode 0, this input and M1, M2 are sampled before the start of configuration to establish the configuration mode to be used.

#### RTRIG

As a Read Trigger, a LOW-to-HIGH input transition, after configuration is complete, will initiate a Readback of configuration and storage element data by CCLK. This operation may be limited to a single request, or be inhibited altogether, by selecting the appropriate readback option when generating the bit stream.

---

#### M1

As Mode 1, this input and M0, M2 are sampled before the start of configuration to establish the configuration mode to be used. If Readback is to be used, a 5 KΩ resistor should be used to define mode level inputs.

#### RDATA

As an active low Read Data, after configuration is complete, this pin is the output of the readback data.

## 2. User I/O Pins that can have special functions.

### M2

As Mode 2 this input has a passive pullup during configuration. Together with M0 and M1 it is sampled before the start of configuration to establish the configuration mode to be used. After configuration this pin becomes a user programmable I/O pin.

### HDC

High During Configuration is held at a HIGH level by the LCA until after configuration. It is available as a control output indicating that configuration is not yet completed. After configuration this pin is a user I/O pin.

### LDC

Low During Configuration is held at a LOW level by the LCA until after configuration. It is available as a control output indicating that configuration is not yet completed. It is particularly useful in Master mode as a LOW enable for an EPROM. After configuration this pin is a user I/O pin. If used as a LOW EPROM enable, it must be programmed as a HIGH after configuration.

### INIT

This is an active low open drain output which is held LOW during the power stabilization and internal clearing of the configuration memory. It can be used to indicate status to a configuring microprocessor or, as a wired AND of several slave mode devices, a hold-off signal for a master mode device. After configuration this pin becomes a user programmable I/O pin.

### BCLKIN

This is a direct CMOS level input to the alternate clock buffer (Auxiliary Buffer) in the lower right corner.

### XTL1

This user I/O pin can be used to operate as the output of an amplifier driving an external crystal and bias circuitry.

### XTL2

This user I/O pin can be used as the input of an amplifier connected to an external crystal and bias circuitry. The I/O Block is left unconfigured. The oscillator configuration is activated by routing a net from the oscillator buffer symbol output and by the MAKEBITS program.

### CS0, CS1, CS2, WRT

These four inputs represent a set of signals, three active low and one active high, which are used in the Peripheral mode to control configuration data entry. The assertion of all 4 generates a write to the internal data buffer. The removal of any assertion, clocks in the D0–D7 data present.

### RCLK

During Master parallel mode configuration  $\overline{RCLK}$  represents a “read” of an external dynamic memory device (normally not used).

### RDY/BUSY

During Peripheral parallel mode configuration this pin indicates when the chip is ready for another byte of data to be written to it. After configuration is complete, this pin becomes a user programmed I/O pin.

### D0–D7

This set of 8 pins represent the parallel configuration byte for the parallel Master and Peripheral modes. After configuration is complete they are user programmed I/O pin.

### A0–A15

This set of 16 pins present an address output for a configuration EPROM during Master parallel mode. After configuration is complete they are user programmed I/O pin.

### DIN

This user I/O pin is used as serial Data input during Slave or Master Serial configuration. This pin is Data 0 input in Master or Peripheral configuration mode.

### DOUT

This user I/O pin is used during configuration to output serial configuration data for daisy-chained slaves' Data In.

### TCLKIN

This is a direct CMOS level input to the global clock buffer.

## 3. Unrestricted User I/O Pins.

### I/O

A pin which may be programmed by the user to be Input and/or Output pin following configuration. Some of these pins present a high impedance pull-up (see next page) or perform other functions before configuration is complete (see above).

Table 2a. XC3000 Family Configuration Pin Assignments

CONFIGURATION MODE: <M2:M1:M0>					68	84	84	132	175	USER
SLAVE <1:1:1>	MASTER-SER <0:0:0>	PERIPHERAL <1:0:1>	MASTER-HIGH <1:1:0>	MASTER-LOW <1:0:0>	PLCC	PLCC	PGA	PGA	PGA	OPERATION
PWR DWN (I)	PWR DWN (I)	PWR DWN (I)	PWR DWN (I)	PWR DWN (I)	10	12	B2	A1	B2	PWR DWN (I)
VCC	VCC	VCC	VCC	VCC	18	22	F3	C8	D9	VCC
M1 (HIGH) (I)	M1 (LOW) (I)	M1 (LOW) (I)	M1 (HIGH) (I)	M1 (LOW) (I)	25	31	J2	B13	B14	RDATA
M0 (HIGH) (I)	M0 (LOW) (I)	M0 (HIGH) (I)	M0 (LOW) (I)	M0 (LOW) (I)	26	32	L1	A14	B15	RTRIG (I)
M2 (HIGH) (I)	M2 (LOW) (I)	M2 (HIGH) (I)	M2 (HIGH) (I)	M2 (HIGH) (I)	27	33	K2	C13	C15	IO
HDC (HIGH)	HDC (LOW)	HDC (HIGH)	HDC (HIGH)	HDC (HIGH)	28	34	K3	B14	E14	IO
LDC (LOW)	LDC (LOW)	LDC (LOW)	LDC (LOW)	LDC (LOW)	30	36	L3	D14	D16	IO
INIT	INIT	INIT	INIT	INIT	34	42	K6	G14	H15	IO
GND	GND	GND	GND	GND	35	43	J6	H12	J14	GND
					43	53	L11	M13	P15	XTL2 OR IO
RESET (I)	RESET (I)	RESET (I)	RESET (I)	RESET (I)	44	54	K10	P14	R15	RESET (I)
DONE	DONE	DONE	DONE	DONE	45	55	J10	N13	R14	PROGRAM (I)
		DATA 7 (I)	DATA 7 (I)	DATA 7 (I)	46	56	K11	M12	N13	IO
					47	57	J11	P13	T14	XTL1 OR IO
		DATA 6 (I)	DATA 6 (I)	DATA 6 (I)	48	58	H10	N11	P12	IO
		DATA 5 (I)	DATA 5 (I)	DATA 5 (I)	49	60	F10	M9	T11	IO
		CS0 (I)			50	61	G10	N9	R10	IO
		DATA 4 (I)	DATA 4 (I)	DATA 4 (I)	51	62	G11	N8	R9	IO
VCC	VCC	VCC	VCC	VCC	52	64	F9	M8	N9	VCC
		DATA 3 (I)	DATA 3 (I)	DATA 3 (I)	53	65	F11	N7	P8	IO
		CS1 (I)			54	66	E11	P6	R8	IO
		DATA 2 (I)	DATA 2 (I)	DATA 2 (I)	55	67	E10	M6	R7	IO
		DATA 1 (I)	DATA 1 (I)	DATA 1 (I)	56	70	D10	M5	R5	IO
		RDY/BUSY	RCLK	RCLK	57	71	C11	N4	P5	IO
DIN (I)	DIN (I)	DATA 0 (I)	DATA 0 (I)	DATA 0 (I)	58	72	B11	N2	R3	IO
DOUT	DOUT	DOUT	DOUT	DOUT	59	73	C10	M3	N4	IO
CCLK (I)	CCLK	CCLK	CCLK	CCLK	60	74	A11	P1	R2	CCLK (I)
		WS (I)	A0	A0	61	75	B10	M2	P2	IO
		CS2 (I)	A1	A1	62	76	B9	N1	M3	IO
			A2	A2	63	77	A10	L2	P1	IO
			A3	A3	64	78	A9	L1	N1	IO
			A15	A15	65	81	B6	K1	M1	IO
			A4	A4	66	82	B7	J2	L2	IO
			A14	A14	67	83	A7	H1	K2	IO
			A5	A5	68	84	C7	H2	K1	IO
GND	GND	GND	GND	GND	1	1	C6	H3	J3	GND
			A13	A13	2	2	A6	G2	H2	IO
			A6	A6	3	3	A5	G1	H1	IO
			A12	A12	4	4	B5	F2	F2	IO
			A7	A7	5	5	C5	E1	E1	IO
			A11	A11	6	8	A3	D1	D1	IO
			A8	A8	7	9	A2	D2	C1	IO
			A10	A10	8	10	B3	B1	E3	IO
			A9	A9	9	11	A1	C2	C2	IO
					X	X	X			XC3020
					X	X				XC3030
					X	X	X			XC3042
							X			XC3064
							X	X		XC3090

 REPRESENTS A 50KΩ TO 100KΩ PULL-UP  
 \* INIT IS AN OPEN DRAIN OUTPUT DURING CONFIGURATION  
 (I) REPRESENTS AN INPUT

AVAILABLE PACKAGES

Note: Pin assignments of "PGA Footprint" PLCC sockets and PGA packages are not electrically identical. Generic I/O pins are not shown.

**Table 2b. XC3000 Family 68-Pin PLCC Pinouts**

PLCC Pin Numbers	XC-3020		PLCC Pin Numbers	XC-3020
10	PWRDN		44	RESET
11	TCLKIN-I/O		45	DONE-PG
12	I/O		46	D7-I/O
13	I/O	◀	47	XTL1(OUT)-BCLKIN-I/O
14	I/O		48	D6-I/O
15	I/O		49	D5-I/O
16	I/O		50	CS0-I/O
17	I/O		51	D4-I/O
18	VCC		52	VCC
19	I/O	◀	53	D3-I/O
20	I/O		54	CS1-I/O
21	I/O		55	D2-I/O
22	I/O	◀	56	D1-I/O
23	I/O		57	RDY/BUSY-RCLK-I/O
24	I/O	◀	58	D0-DIN-I/O
25	M1-RDATA		59	DOUT-I/O
26	M0-RTRIG		60	CCLK
27	M2-I/O		61	A0-WS-I/O
28	HDC-I/O		62	A1-CS2-I/O
29	I/O		63	A2-I/O
30	LD0-I/O		64	A3-I/O
31	I/O		65	A15-I/O
32	I/O		66	A4-I/O
33	I/O		67	A14-I/O
34	INIT-I/O		68	A5-I/O
35	GND		1	GND
36	I/O		2	A13-I/O
37	I/O		3	A6-I/O
38	I/O		4	A12-I/O
39	I/O		5	A7-I/O
40	I/O		6	A11-I/O
41	I/O		7	A8-I/O
42	I/O		8	A10-I/O
43	XTL2(IN)-I/O		9	A9-I/O

◀ 6 Unbonded IOBs 3020

Unprogrammed IOBs have a default pull-up. This prevents an undefined pad level for unbonded or unused IOBs. Programmed outputs are default slew-limited.

Table 2c. XC3000 Family 84-Pin PLCC and PGA Pinouts

PLCC Pin Number	PGA Pin Number	XC-3020* XC-3030
12	B2	PWRDN
13	C2	TCLKIN-I/O
14	B1	I/O *
15	C1	I/O
16	D2	I/O
17	D1	I/O
18	E3	I/O
19	E2	I/O
20	E1	I/O
21	F2	I/O
22	F3	VCC
23	G3	I/O
24	G1	I/O
25	G2	I/O
26	F1	I/O
27	H1	I/O
28	H2	I/O
29	J1	I/O
30	K1	I/O
31	J2	M1-RDATA
32	L1	M0-RTRIG
33	K2	M2-I/O
34	K3	HDC-I/O
35	L2	I/O
36	L3	LDC-I/O
37	K4	I/O
38	L4	I/O *
39	J5	I/O
40	K5	I/O
41	L5	I/O *
42	K6	INIT-I/O
43	J6	GND
44	J7	I/O
45	L7	I/O
46	K7	I/O
47	L6	I/O
48	L8	I/O
49	K8	I/O
50	L9	I/O *
51	L10	I/O *
52	K9	I/O
53	L11	XTL2(IN)-I/O

PLCC Pin Number	PGA Pin Number	XC-3020* XC-3030
54	K10	RESET
55	J10	DONE-PG
56	K11	D7-I/O
57	J11	XTL1(OUT)-BCLKIN-I/O
58	H10	D6-I/O
59	H11	I/O
60	F10	D5-I/O
61	G10	CS0-I/O
62	G11	D4-I/O
63	G9	I/O
64	F9	VCC
65	F11	D3-I/O
66	E11	CS1-I/O
67	E10	D2-I/O
68	E9	I/O
69	D11	I/O *
70	D10	D1-I/O
71	C11	RDY/BUSY-RCLK-I/O
72	B11	D0-DIN-I/O
73	C10	DOOUT-I/O
74	A11	CCLK
75	B10	A0-WS-I/O
76	B9	A1-CS2-I/O
77	A10	A2-I/O
78	A9	A3-I/O
79	B8	I/O *
80	A8	I/O *
81	B6	A15-I/O
82	B7	A4-I/O
83	A7	A14-I/O
84	C7	A5-I/O
1	C6	GND
2	A6	A13-I/O
3	A5	A6-I/O
4	B5	A12-I/O
5	C5	A7-I/O
6	A4	I/O *
7	B4	I/O *
8	A3	A11-I/O
9	A2	A8-I/O
10	B3	A10-I/O
11	A1	A9-I/O

Unprogrammed IOBs have a default pull-up. This prevents an undefined pad level for unbonded or unused IOBs. Programmed outputs are default slew-limited.

\* Indicates unconnected package pins for the XC3020.

**Table 2d. XC3000 Family 132 Pin PGA Pinouts**

PGA Pin Number	XC-3042 XC-3064						
C4	GND	B13	M1-RD	P14	RESET	M3	DOUT-I/O
A1	PWRDN	C11	GND	M11	VCC	P1	CCLK
C3	I/O	A14	M0-RT	N13	DONE-PG	M4	VCC
B2	I/O	D12	VCC	M12	D7-I/O	L3	GND
B3	I/O	C13	M2-I/O	P13	XTAL1-I/O	M2	A0-WS-I/O
A2	I/O*	B14	HDC-I/O	N12	I/O	N1	A1-CS2-I/O
B4	I/O	C14	I/O	P12	I/O	M1	I/O
C5	I/O	E12	I/O	N11	D6-I/O	K3	I/O
A3	I/O*	D13	I/O	M10	I/O	L2	A2-I/O
A4	I/O	D14	LDC-I/O	P11	I/O*	L1	A3-I/O
B5	I/O	E13	I/O*	N10	I/O	K2	I/O
C6	I/O	F12	I/O	P10	I/O	J3	I/O
A5	I/O	E14	I/O	M9	D5-I/O	K1	A15-I/O
B6	I/O	F13	I/O	N9	CS0-I/O	J2	A4-I/O
A6	I/O	F14	I/O	P9	I/O*	J1	I/O*
B7	I/O	G13	I/O	P8	I/O*	H1	A14-I/O
C7	GND	G14	INIT-I/O	N8	D4-I/O	H2	A5-I/O
C8	VCC	G12	VCC	P7	I/O	H3	GND
A7	I/O	H12	GND	M8	VCC	G3	VCC
B8	I/O	H14	I/O	M7	GND	G2	A13-I/O
A8	I/O	H13	I/O	N7	D3-I/O	G1	A6-I/O
A9	I/O	J14	I/O	P6	CS1-I/O	F1	I/O*
B9	I/O	J13	I/O	N6	I/O*	F2	A12-I/O
C9	I/O	K14	I/O	P5	I/O*	E1	A7-I/O
A10	I/O	J12	I/O	M6	D2-I/O	F3	I/O
B10	I/O	K13	I/O	N5	I/O	E2	I/O
A11	I/O*	L14	I/O*	P4	I/O	D1	A11-I/O
C10	I/O	L13	I/O	P3	I/O	D2	A8-I/O
B11	I/O	K12	I/O	M5	D1-I/O	E3	I/O
A12	I/O*	M14	I/O	N4	RCLK-BUSY/RDY-I/O	C1	I/O
B12	I/O	N14	I/O	P2	I/O	B1	A10-I/O
A13	I/O*	M13	XTAL2-I/O	N3	I/O	C2	A9-I/O
C12	I/O	L12	GND	N2	D0-DIN-I/O	D3	VCC

Unprogrammed IOBs have a default pull-up. This prevents an undefined pad level for unbonded or unused IOBs. Programmed outputs are default slew-limited.

\* Indicates unconnected package pins for the XC3042.

Table 2e. XC3000 Family 175-Pin PGA Pinouts

PGA Pin Number	XC-3090	PGA Pin Number	XC-3090	PGA Pin Number	XC-3090	PGA Pin Number	XC-3090
B2	PWRDN	D13	I/O	R14	DONE-PG	R3	D0-DIN-I/O
D4	TCLKIN-I/O	B14	M1-RDATA	N13	D7-I/O	N4	DOUT-I/O
B3	I/O	C14	GND	T14	XTAL1(OUT)-BCLKIN-I/O	R2	CCLK
C4	I/O	B15	M0-RTRIG	P13	I/O	P3	VCC
B4	I/O	D14	VCC	R13	I/O	N3	GND
A4	I/O	C15	M2-I/O	T13	I/O	P2	A0-WS-I/O
D5	I/O	E14	HDC-I/O	N12	I/O	M3	A1-CS2-I/O
C5	I/O	B16	I/O	P12	D6-I/O	R1	I/O
B5	I/O	D15	I/O	R12	I/O	N2	I/O
A5	I/O	C16	I/O	T12	I/O	P1	A2-I/O
C6	I/O	D16	LDC-I/O	P11	I/O	N1	A3-I/O
D6	I/O	F14	I/O	N11	I/O	L3	I/O
B6	I/O	E15	I/O	R11	I/O	M2	I/O
A6	I/O	E16	I/O	T11	D5-I/O	M1	A15-I/O
B7	I/O	F15	I/O	R10	CS0-I/O	L2	A4-I/O
C7	I/O	F16	I/O	P10	I/O	L1	I/O
D7	I/O	G14	I/O	N10	I/O	K3	I/O
A7	I/O	G15	I/O	T10	I/O	K2	A14-I/O
B8	I/O	G16	I/O	T9	I/O	K1	A5-I/O
B8	I/O	H16	I/O	R9	D4-I/O	J1	I/O
C8	I/O	H15	INIT-I/O	P9	I/O	J2	I/O
D8	GND	H14	VCC	N9	VCC	J3	GND
D9	VCC	J14	GND	N8	GND	H3	VCC
C9	I/O	J15	I/O	P8	D3-I/O	H2	A13-I/O
B9	I/O	J16	I/O	R8	CS1-I/O	H1	A6-I/O
A9	I/O	K16	I/O	T8	I/O	G1	I/O
A10	I/O	K15	I/O	T7	I/O	G2	I/O
D10	I/O	K14	I/O	N7	I/O	G3	I/O
C10	I/O	L16	I/O	P7	I/O	F1	I/O
B10	I/O	L15	I/O	R7	D2-I/O	F2	A12-I/O
A11	I/O	M16	I/O	T6	I/O	E1	A7-I/O
B11	I/O	M15	I/O	R6	I/O	E2	I/O
D11	I/O	L14	I/O	N6	I/O	F3	I/O
C11	I/O	N16	I/O	P6	I/O	D1	A11-I/O
A12	I/O	P16	I/O	T5	I/O	C1	A8-I/O
B12	I/O	N15	I/O	R5	D1-I/O	D2	I/O
C12	I/O	R16	I/O	P5	RDY/BUSY-RCLK-I/O	B1	I/O
D12	I/O	M14	I/O	N5	I/O	E3	A10-I/O
A13	I/O	P15	XTAL2(IN)-I/O	T4	I/O	C2	A9-I/O
B13	I/O	N14	GND	R4	I/O	D3	VCC
C13	I/O	R15	RESET	P4	I/O	C3	GND
A14	I/O	P14	VCC				

Unprogrammed IOBs have a default pull-up. This prevents an undefined pad level for unbonded or unused IOBs. Programmed outputs are default slew-limited.

Pins A2, A3, A15, A16, T1, T2, T3, T15 and T16 are not connected.  
Pin A1 does not exist.

## PARAMETRICS

Absolute Maximum Ratings			Units
V <sub>CC</sub>	Supply voltage relative to GND	-0.5 to 7.0	V
V <sub>IN</sub>	Input voltage with respect to GND	-0.5 to V <sub>CC</sub> + 0.5	V
V <sub>TS</sub>	Voltage applied to three-state output	-0.5 to V <sub>CC</sub> + 0.5	V
T <sub>STG</sub>	Storage temperature (ambient)	-65 to + 150	°C
T <sub>SOL</sub>	Maximum soldering temperature (10 sec @ 1/16 in.)	+ 260	°C
T <sub>J</sub>	Junction temperature plastic	+ 125	°C
	Junction temperature ceramic	+ 150	°C

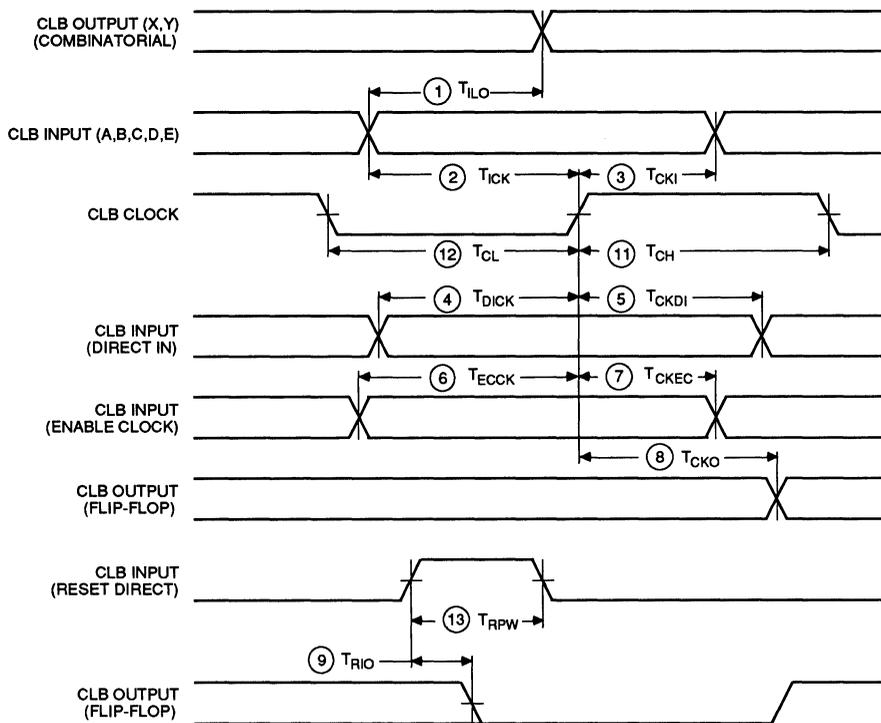
\*Note: Stresses beyond those listed under Absolute Maximum Ratings may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those listed under Recommended Operating Conditions is not implied. Exposure to Absolute Maximum Ratings conditions for extended periods of time may affect device reliability.

Recommended Operating Conditions				Min	Max	Units
V <sub>CC</sub>	Supply voltage relative to GND	Commercial	0°C to 70°C	4.75	5.25	V
	Supply voltage relative to GND	Industrial	-40°C to 85°C	4.5	5.5	V
	Supply voltage relative to GND	Military	-55°C to 125°C	4.5	5.5	V
V <sub>IHT</sub>	High-level input voltage — TTL configuration			2.0	V <sub>CC</sub>	V
V <sub>ILT</sub>	Low-level input voltage — TTL configuration			0	0.8	V
V <sub>IHC</sub>	High-level input voltage — CMOS configuration			70%	100%	V <sub>CC</sub>
V <sub>ILC</sub>	Low-level input voltage — CMOS configuration			0	20%	V <sub>CC</sub>
T <sub>IN</sub>	Input signal transition time				250	ns

Electrical Characteristics Over Operating Conditions			Min	Max	Units
V <sub>OH</sub>	High-level output voltage (@ I <sub>OH</sub> = -4.0 ma V <sub>CC</sub> min)	Commercial	3.86		V
V <sub>OL</sub>	Low-level output voltage (@ I <sub>OL</sub> = 4.0 ma V <sub>CC</sub> min)			0.32	V
V <sub>OH</sub>	High-level output voltage (@ I <sub>OH</sub> = -4.0 ma V <sub>CC</sub> )	Industrial	3.76		V
V <sub>OL</sub>	Low-level output voltage (@ I <sub>OL</sub> = 4.0 ma V <sub>CC</sub> )			0.37	V
V <sub>OH</sub>	High-level output voltage (@ I <sub>OH</sub> = -4.0 ma V <sub>CC</sub> )	Military	3.7		V
V <sub>OL</sub>	Low-level output voltage (@ I <sub>OH</sub> = 4.0 ma V <sub>CC</sub> )			0.4	V
I <sub>CCPD</sub>	Power-down supply current (V <sub>CC</sub> = 5.0 V @ 70°C)	XC3020		500	μA
		XC3030		800	μA
		XC3042		1150	μA
		XC3064		1650	μA
		XC3090		2500	μA
I <sub>CCO</sub>	Quiescent LCA supply current in addition to I <sub>CCPD</sub> <sup>1</sup>				
	Chip thresholds programmed as CMOS levels			500	μA
	Chip thresholds programmed as TTL levels			10	mA
I <sub>IL</sub>	Leakage Current Commercial/Industrial Temperature		-10	+10	μA
	Leakage Current Military -55°C to 125°C		-20	+20	μA
C <sub>IN</sub>	Input capacitance, all packages except PGA 175 (sample tested) All Pins except XTL1 and XTL2 XTL1 and XTL2			10 15	pF pF
	Input capacitance, PGA 175 (sample tested) All Pins except XTL1 and XTL2 XTL1 and XTL2			15 20	pF pF
I <sub>RIN</sub>	Pad pull-up (when selected) @ V <sub>IN</sub> = 0V (sample tested)		0.02	0.17	mA
I <sub>RL</sub>	Horizontal long line pull-up (when selected) @ logic LOW		0.4	3.4	mA

Note: 1. With no output current loads, no active input or long line pull-up resistors, all package pins at V<sub>CC</sub> or GND, and the LCA configured with a MAKEBITS "tie" option. See LCA power chart for additional activity dependent operating component.

## CLB SWITCHING CHARACTERISTIC GUIDELINES



1105 26

## CLB SWITCHING CHARACTERISTIC GUIDELINES (Continued)

Testing of the switching characteristic guidelines is modeled after testing specified by MIL-M-38510/605. Devices are 100% functionally tested. Benchmark timing patterns are used to provide correlation to the switching characteristic guideline values.

		Speed Grade		-50		-70				Units
	Description	Symbol		Min	Max	Min	Max	Min	Max	
CLB Logic input	Combinatorial	1	TiLO		14		9			ns
Reset direct	CLB output	9	TRIO		15		10			ns
	Reset Direct width*	13	TRPW	10		7				ns
	Master Reset pin to CLB out		TMRQ		35		25			ns
CLB K Clock input	To CLB output	8	TCKO		12		8			ns
	Additional for Q returning through F or G to CLB out		TQLO		11		7			ns
	Logic-input setup	2	TICK	12		8				ns
	Logic-input hold	3	TCKI	1		1				ns
	Data In setup	4	TDICK	8		5				ns
	Data In hold (1)	5	TCKDI	6		4				ns
	Enable Clock setup	6	TECCK	10		7				ns
	Enable Clock hold	7	TCKEC	1		1				ns
	Clock (high)*	11	TCH	9		7				ns
Clock (low)*	12	TCL	9		7				ns	
Flip-flop Toggle rate	Q through F/G to flip-flop in*		FCLK	50		70				MHz

\* These timing limits are based on calculations.

The speed of block inputs is a function of interconnect.

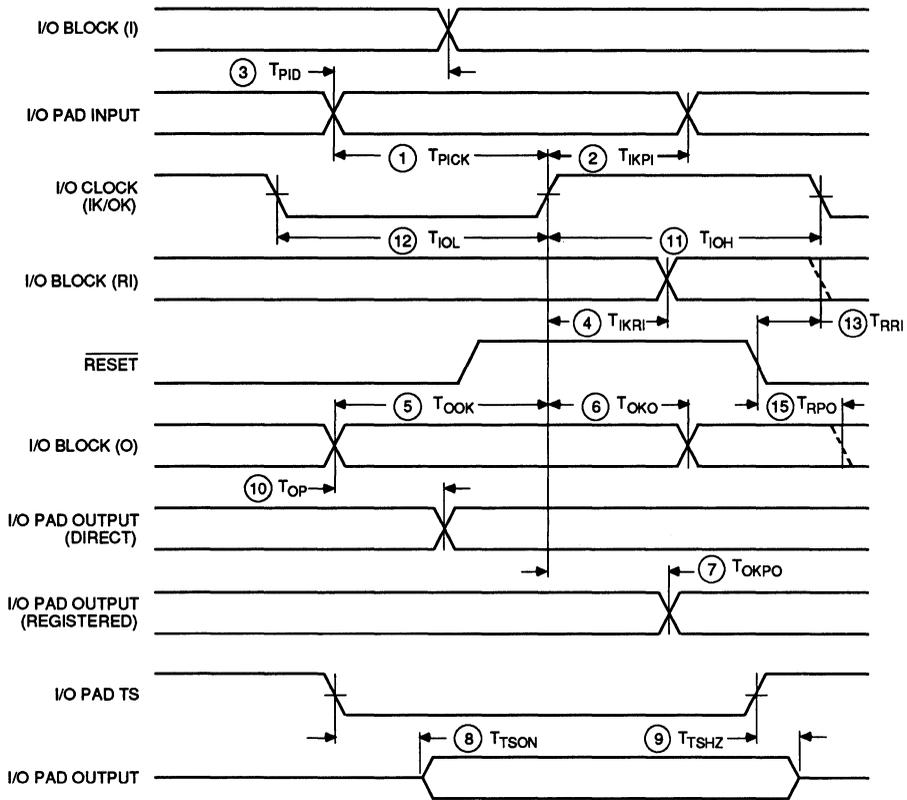
Note 1. The CLB K to Q output delay (TCKO, #8) of any CLB, plus the shortest possible interconnect delay, is always longer than the Data In hold time requirement (TCKDI, #5) of any CLB on the same die.

## BUFFER (Internal) SWITCHING CHARACTERISTIC GUIDELINES

		-50		-70				Units		
	Description	Symbol		Min	Max	Min	Max		Min	Max
Clock Buffer**	GCLK, ACLK				9		6			ns
TBUF**	Data to Output (Long line buffer)				8		5			ns
	Three-state to Output				34		22			ns
	Single pull-up resistor				17		11			ns
Pair of pull-up resistors										ns
Bi-directional	BIDI				6		4			ns

\*\* Timing is based on the XC3020, for other devices see XACT timing calculator.

**IOB SWITCHING CHARACTERISTIC GUIDELINES**



1105 27

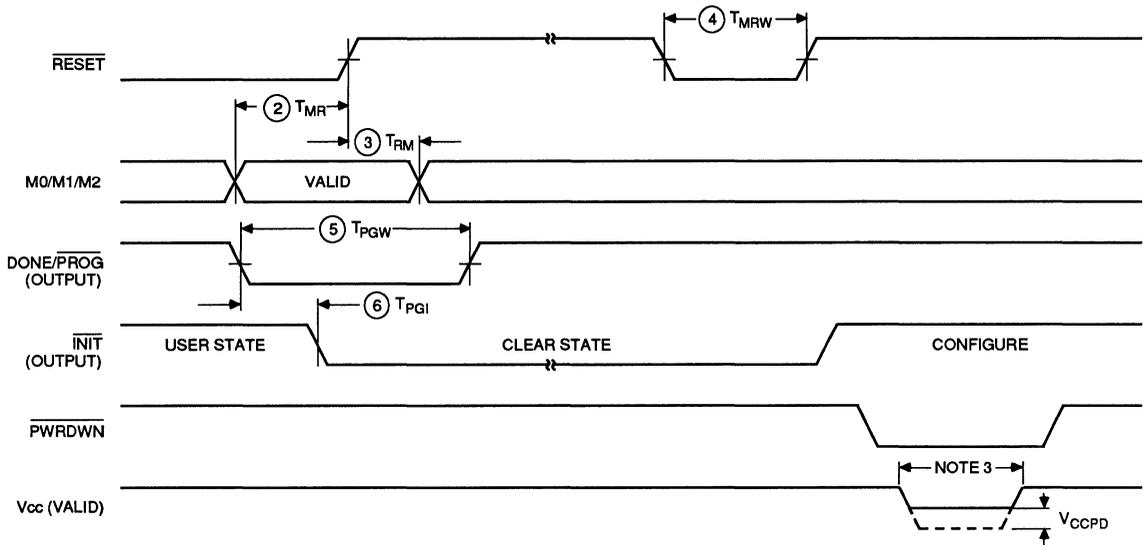
## IOB SWITCHING CHARACTERISTIC GUIDELINES (Continued)

Testing of the switching characteristic guidelines is modeled after testing specified by MIL-M-38510/605. Devices are 100% functionally tested. Benchmark timing patterns are used to provide correlation to the switching characteristic guideline values. Actual worst-case timing is provided by the XACT Timing calculator or Simulation modeling.

				-50		-70				Units
	Description	Symbol		Min	Max	Min	Max	Min	Max	
Pad (package pin)	To inputs TCLKIN, BCLKIN	3	TPIDC		5		3			ns
	To inputs DIRECT IN		TPID		10		7			ns
I/O Clock	To I/O RI input (FF)	4	TIKRI		10		7			ns
	I/O pad-input setup	1	TPICK	30		20				ns
	I/O pad-input hold	2	TIKPI	0		0				ns
	To I/O pad (fast)	7	TOKPO		18		13			ns
	I/O pad output setup	5	TOKO	15		10				ns
	I/O pad output hold	6	TOKO	0		0				ns
	Clock (high)	11	TIOH	9		7				ns
	Clock (low)	12	TIOH	9		7				ns
Output	To pad (enabled fast)	10	TOPF		14		10			ns
	To pad (enabled slow)	10	TOPS		33		25			ns
Three-state	To pad begin hi-Z (fast)	9	TTSHZ		12		8			ns
	To pad valid (fast)	8	TTSON		20		14			ns
Master Reset (Package Pin)	To input RI	13	TRRI		45		30			ns
	To output (FF)	15	TRPO		55		37			ns

- Notes:
- Timing is measured at pin threshold, with 50 pF external capacitive loads (incl. test fixture).  
 Typical fast mode output rise/fall times are 2 ns and will increase approximately 2%/pF of additional load.  
 Typical slew rate limited output rise/fall times are approximately 4 times longer.  
 A maximum total external capacitive load for simultaneous fast mode switching in the same direction is 500 pF per power/ground pin pair. For slew-rate limited outputs this total is 4 times larger.
  - Voltage levels of unused (bonded and unbonded) pads must be valid logic levels. Each can be configured with the internal pull-up resistor or alternatively configured as a driven output or driven from an external source.
  - For more information on input/output timing see the applications section of the Xilinx Data Book.

## GENERAL LCA SWITCHING CHARACTERISTICS

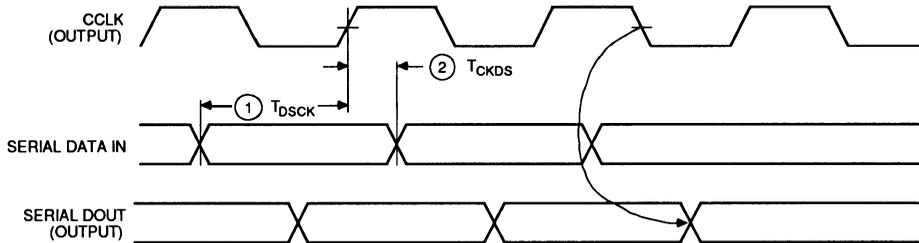


1105 28

				-50		-70				Units
Description		Symbol		Min	Max	Min	Max	Min	Max	
RESET (2)	M2, M1, M0 setup	2	T <sub>MR</sub>	1	0	1	0	0	0	ns
	M2, M1, M0 hold	3	T <sub>RM</sub>	1		1				ns
	Width (low) Abort	4	T <sub>MRW</sub>	6		6				ns
DONE/PROG	Progam width (low)	5	T <sub>PGW</sub>	6		6				ns
	Start INIT	6	T <sub>PGI</sub>		7		7			ns
PWRDWN (3)	Power Down V <sub>cc</sub>		V <sub>CCPD</sub>	2.0						V

- Notes:
1. V<sub>cc</sub> must rise from 2.0 Volts to V<sub>cc</sub> minimum in less than 10 ms for master modes.
  2. RESET timing relative to valid mode lines (M0, M1, M2) is relevant when RESET is used to delay configuration.
  3. PWRDWN transitions must occur during operational V<sub>cc</sub> levels.

## MASTER SERIAL MODE SWITCHING CHARACTERISTICS GUIDELINES



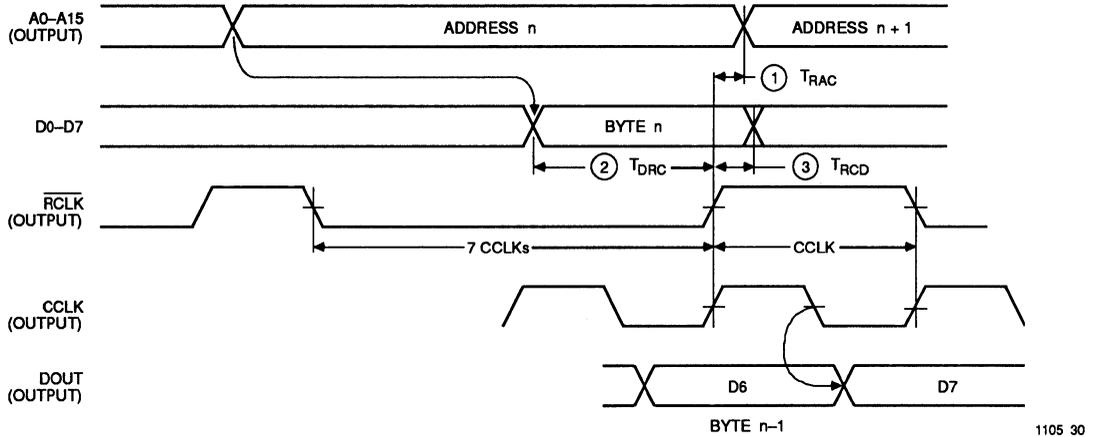
1105 29

		Speed Grade		-50		-70				Units
	Description	Symbol	Min	Max	Min	Max	Min	Max		
CCLK <sup>3</sup>	Data In setup	1	T <sub>DSCK</sub>	60		60				ns
	Data In hold	2	T <sub>CKDS</sub>	0		0				ns

- Notes:
1. At power-up, V<sub>cc</sub> must rise from 2.0 V to V<sub>cc</sub> min. in less than 10 ms, otherwise delay configuration using **RESET** until V<sub>cc</sub> is valid.
  2. Configuration can be controlled by holding **RESET** low with or until after the **INIT** of all daisy-chain slave mode devices is HIGH.
  3. Master serial mode timing is based on slave mode testing.

## MASTER PARALLEL MODE PROGRAMMING SWITCHING CHARACTERISTIC GUIDELINES

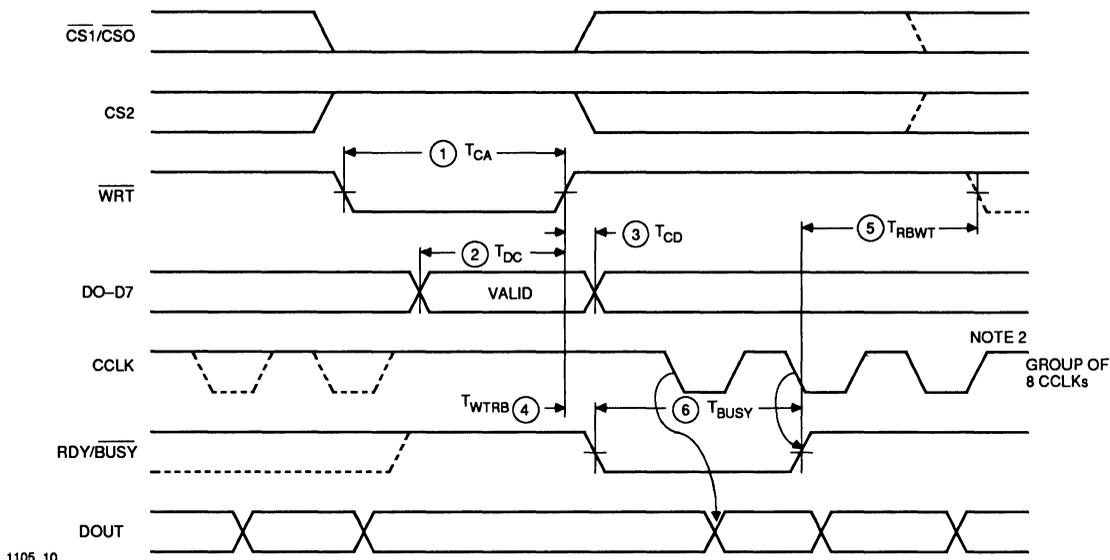
Testing of the switching characteristic guidelines is modeled after testing specified by MIL-M-38510/605. Devices are 100% functionally tested. Benchmark timing patterns are used to provide correlation to the switching characteristic guideline values. Actual worst-case timing is provided by the XACT Timing calculator or Simulation modeling.



			-50		-70				Units
	Description	Symbol	Min	Max	Min	Max	Min	Max	
RCLK	To address valid	1 TRAC	0	200	0	200			ns
	To data setup	2 TDRC	60		60				ns
	To data hold	3 TRCD	0		0				ns
	RCLK high	TRCH	600		600				ns
	RCLK low	TRCL	4.0		4.0				µs

- Notes: 1. At power-up, Vcc must rise from 2.0 V to Vcc min. in less than 10 ms, otherwise delay configuration using  $\overline{\text{RESET}}$  until Vcc is valid.  
 2. Configuration can be controlled by holding  $\overline{\text{RESET}}$  low with or until after the  $\overline{\text{INIT}}$  of all daisy-chain slave mode devices is HIGH.

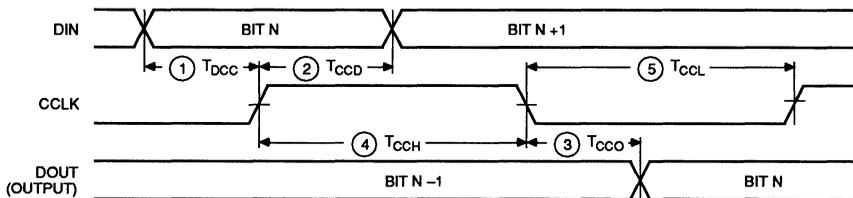
## PERIPHERAL MODE PROGRAMMING SWITCHING CHARACTERISTICS



			-50		-70				Units	
	Description	Symbol		Min	Max	Min	Max	Min		Max
Write	$\overline{\text{WRT}}$ Low	1	$T_{CA}$	0.5		0.5				$\mu\text{s}$
	DIN setup DIN hold	2 3	$T_{DC}$ $T_{CD}$	60 0		60 0				ns ns
	Ready/ $\overline{\text{Busy}}$	4	$T_{WTRB}$		60		60			ns
RDY	WRT Active	5	$T_{RBWT}$	0		0				
	Busy	6	$T_{BUSY}$	4	9	4	9			CCLK

- Notes:
1. Configuration must be delayed until the  $\overline{\text{INIT}}$  of all LCAs is HIGH.
  2. Time from end of  $\overline{\text{WRT}}$  to CCLK cycle for the new byte of data depends on completion of previous byte processing and the phase of the internal timing generator for CCLK.
  3. CCLK and DOUT timing is tested in slave mode.

### SLAVE MODE PROGRAMMING SWITCHING CHARACTERISTICS

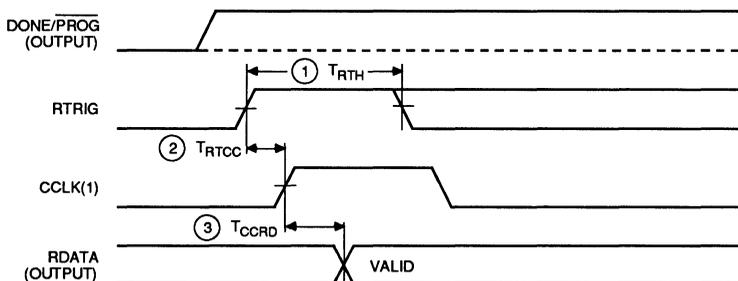


1105 31

			-50		-70				Units
	Description	Symbol	Min	Max	Min	Max	Min	Max	
CCLK	To DOUT	3	T <sub>CCO</sub>		100		100		ns
	DIN setup	1	T <sub>DCC</sub>	60		60			ns
	DIN hold	2	T <sub>CCD</sub>	0		0			ns
	High time	4	T <sub>CCH</sub>	0.5		0.5			ns
	Low time	5	T <sub>CCL</sub>	0.5	5.0	0.5	5.0		ns
	Frequency		F <sub>CC</sub>		1		1		MHz

Note: Configuration must be delayed until the INIT of all LCAs is HIGH.

### PROGRAM READBACK SWITCHING CHARACTERISTICS



1105 32

			-50		-70				Units
	Description	Symbol	Min	Max	Min	Max	Min	Max	
RTRIG	RTRIG high	1	TRTH	250		250			ns
CCLK	RTRIG setup	2	TRTCC	10		10			ns
	RDATA delay	3	T <sub>CCRD</sub>		100		100		ns

Notes: 1. CCLK and DOUT timing are the same as for slave mode.  
 2. RETRIG (M0 positive transition) shall not be done until after one clock following active I/O pins.  
 3. Readback should not be initiated until configuration is complete.

		68 PIN		84 PIN		132 PIN	175 PIN	CERAMIC QUAD FLAT PACK
		PLASTIC PLCC	CERAMIC PGA	PLASTIC PLCC	CERAMIC PGA	CERAMIC PGA	CERAMIC PGA	
		-PC 68	-PG 68	-PC 84	-PG 84	-PG 132	-PG 175	
XC3020	-50	C I		C I	C I M B			UNDER DEVELOPMENT
	-70	C I		C I	C I			
XC3030	-50			C I	C I M B			
	-70			C	C			
XC3042	-50			C I	C I	C I M B		
	-70			C	C	C		
XC3064	-50					C I M B		
	-70					C		
XC3090	-50						C I M B	
	-70						C	

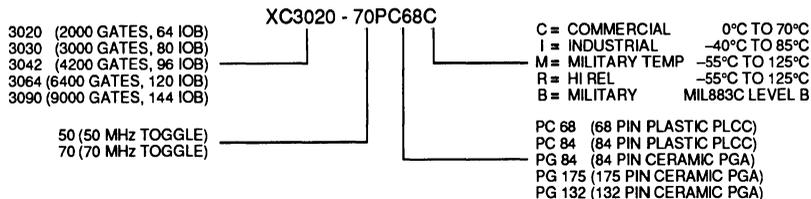
C = COMMERCIAL                      0°C TO 70°C  
 I = INDUSTRIAL                      -40°C TO 85°C  
 M = MILITARY TEMP                -55°C TO 125°C  
 R = HI REL                            -55°C TO 125°C  
 B = MILITARY                        MIL-STD-883, CLASS B

1105 11

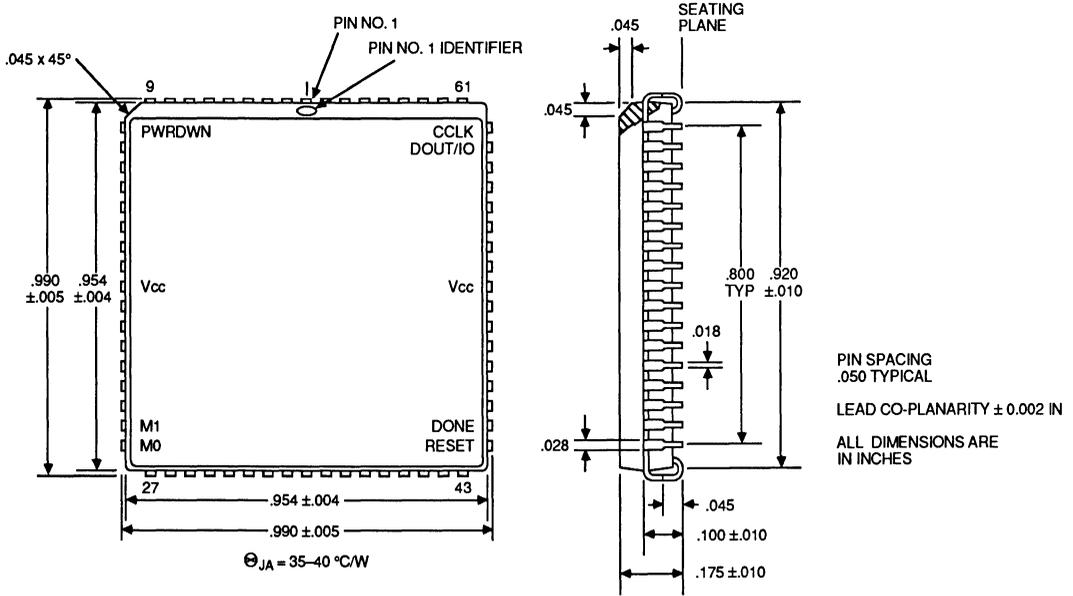
**Table 3.** LCA Package and Temperature Options

## ORDERING INFORMATION

Further information is available from XILINX franchised distributors or from the nearest XILINX sales representative. Part numbers are composed as follows:

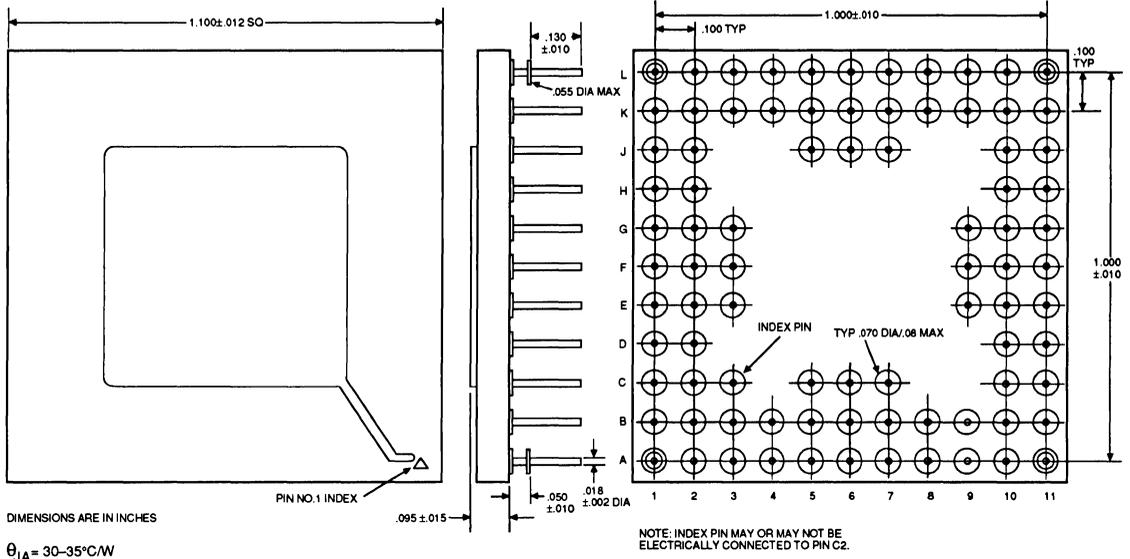


1105 33



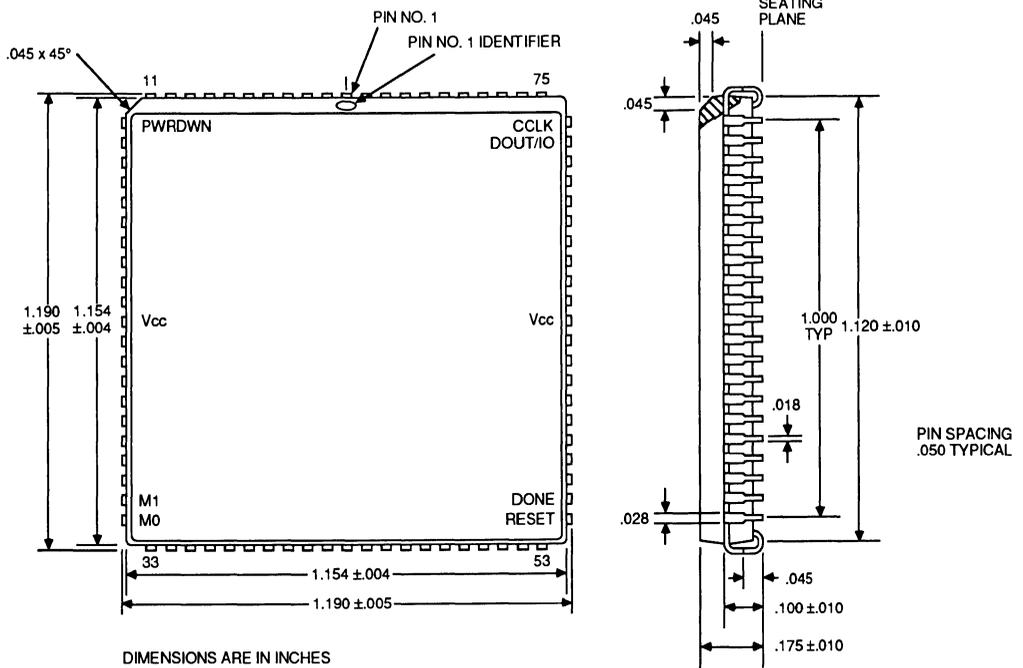
1105 34

68-Pin PLCC Package



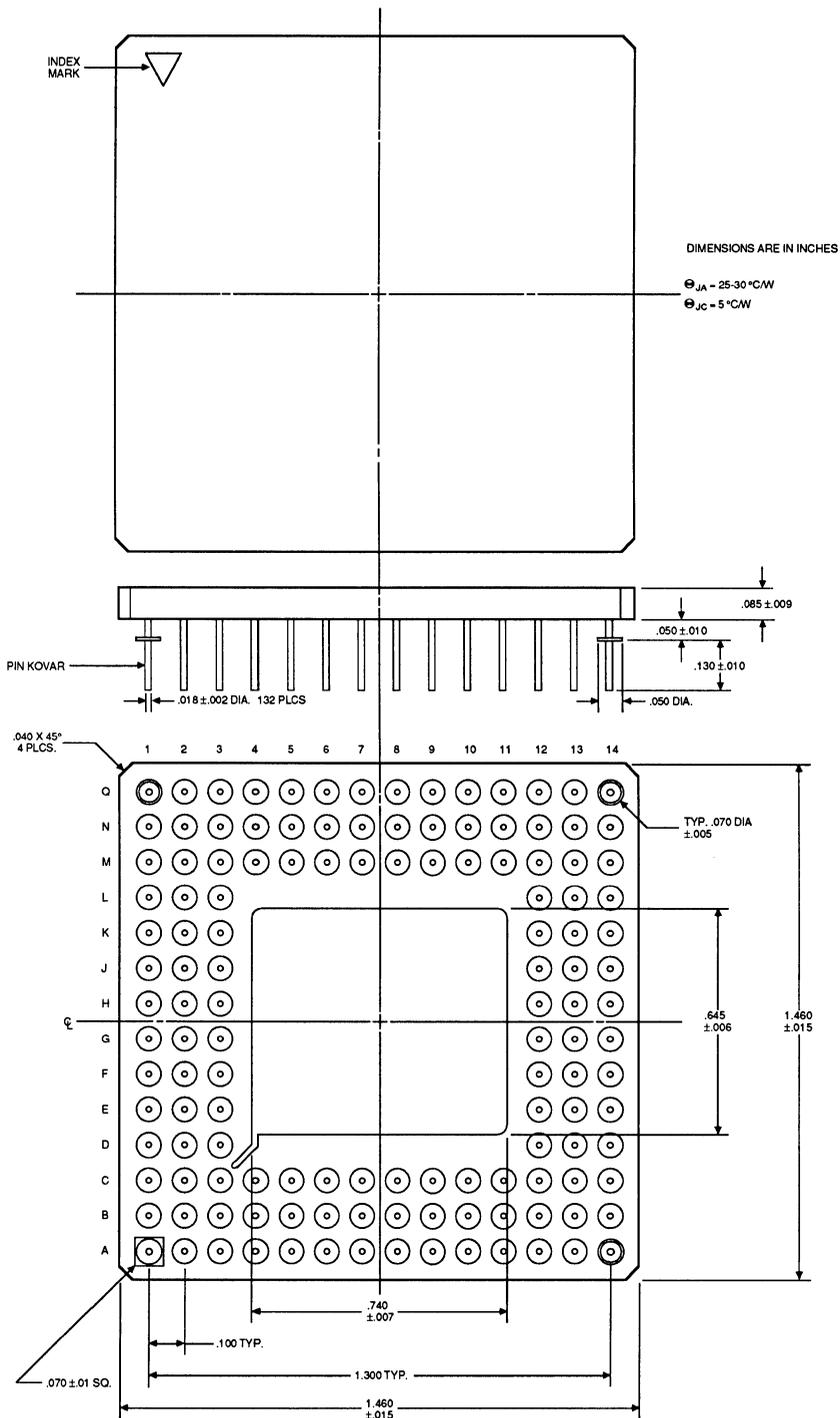
1105 35

84-Pin PGA Package

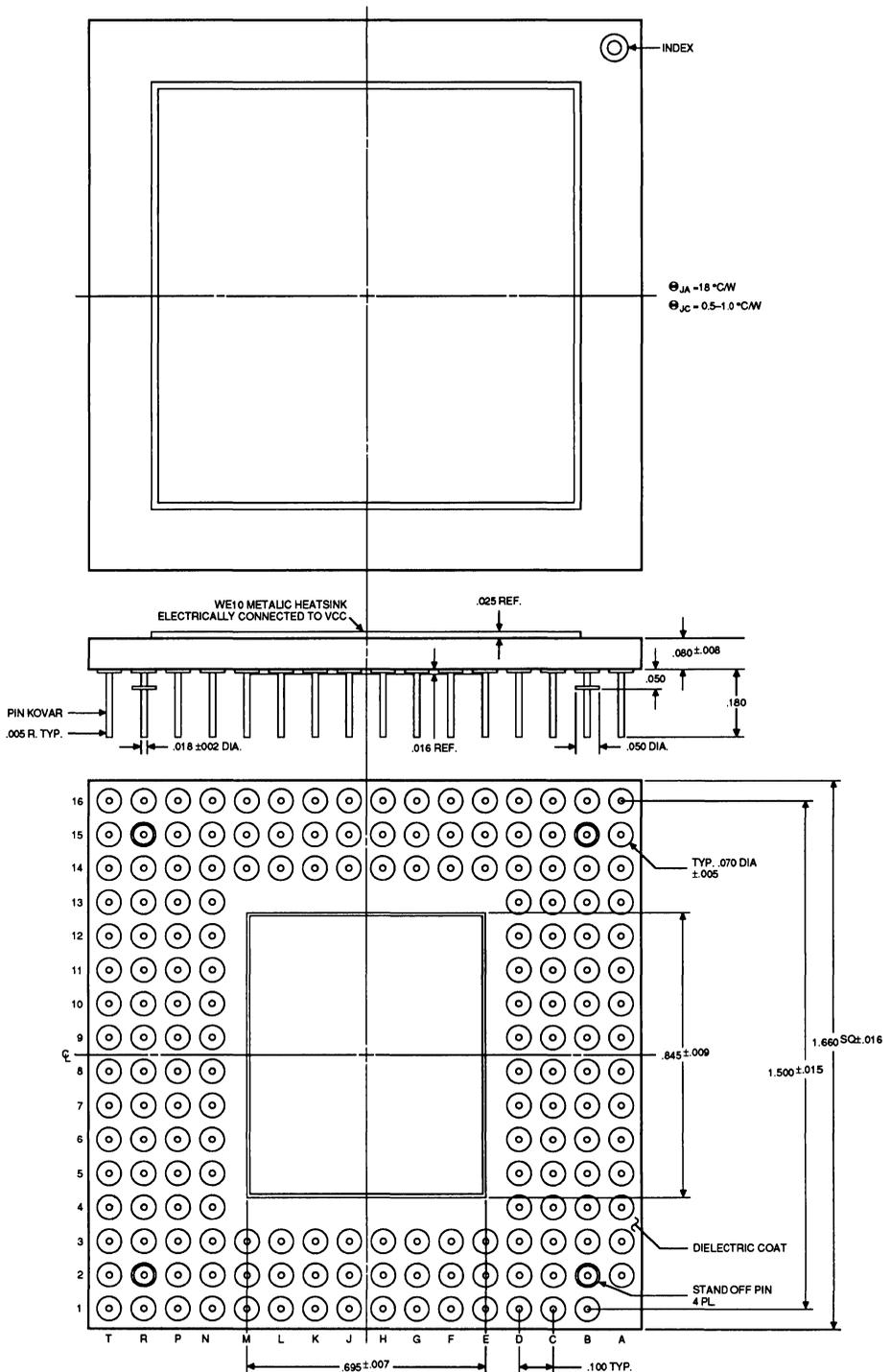


1105 36

84-Pin PLCC Package



132-Pin PGA Package



1105 37

175-Pin PGA Package





# XC2064 XC2018 Logic Cell™ Array

## Product Specification

### FEATURES

- Fully user-programmable:
  - I/O functions
  - Digital logic functions
  - Interconnections
- General-purpose array architecture
- Complete user control of design cycle
- Compatible arrays with logic cell complexity equivalent to 1200 and 1800 usable gates
- Standard product availability
- 100% factory-tested
- Selectable configuration modes
- Low-power, CMOS, static memory technology
- Performance equivalent to TTL SSI/MSI
- TTL or CMOS input thresholds
- Complete development system support
  - XACT Design Editor
  - Schematic Entry
  - XACTOR In Circuit Emulator
  - Macro Library
  - Timing Calculator
  - Logic and Timing Simulator
  - Auto Place / Route

### DESCRIPTION

The Logic Cell™ Array (LCA) is a high density CMOS integrated circuit. Its user-programmable array architecture is made up of three types of configurable elements: Input/Output Blocks, Logic Blocks and Interconnect. The designer can define individual I/O blocks for interface to external circuitry, define logic blocks to implement logic functions and define interconnection networks to compose larger scale logic functions. The XACT™ Development System provides interactive graphic design capture and automatic routing. Both logic simulation and in-circuit emulation are available for design verification.

The Logic Cell Array is available in a variety of logic capacities, package styles, temperature ranges and speed grades.

Part Number	Logic Capacity (usable) gates	Configurable Logic Blocks	User I/Os	Configuration Program (bits)
XC2064	1200	64	58	12038
XC2018	1800	100	74	17878

The Logic Cell Array's logic functions and interconnections are determined by data stored in internal static memory cells. On-chip logic provides for automatic loading of configuration data at power-up. The program data can reside in an EEPROM, EPROM or ROM on the circuit board or on a floppy disk or hard disk. The program can be loaded in a number of modes to accommodate various system requirements.

### ARCHITECTURE

The general structure of a Logic Cell Array is shown in Figure 1. The elements of the array include three categories of user programmable elements: I/O Blocks, Configurable Logic Blocks and Programmable Interconnections. The I/O Blocks provide an interface between the logic array and the device package pins. The Configurable Logic Blocks perform user-specified logic functions, and the interconnect resources are programmed to form networks that carry logic signals among blocks.

Configuration of the Logic Cell Array is established through a distributed array of memory cells. The XACT development system generates the program used to configure the Logic Cell Array. The Logic Cell Array includes logic to implement automatic configuration.

### Configuration Memory

The configuration of the Xilinx Logic Cell Array is established by programming memory cells which determine the logic functions and interconnections. The memory loading process is independent of the user logic functions.

The static memory cell used for the configuration memory in the Logic Cell Array has been designed specifically for high reliability and noise immunity. Based on this design, which is covered by a pending patent application, integrity of the LCA configuration memory is assured even under adverse conditions. Compared with other programming alternatives, static memory provides the best combination of high density, high performance, high reliability and comprehensive testability. As shown in Figure 2, the basic memory cell consists of two CMOS inverters plus a pass transistor used for writing data to the cell. The cell is only written during configuration and only read during read-back. During normal operation the pass transistor is "off" and does not affect the stability of the cell. This is quite different from the normal operation of conventional memory devices, in which the cells are continuously read and rewritten.

The outputs Q and  $\bar{Q}$  control pass-transistor gates directly. The absence of sense amplifiers and the output capacitive load provide additional stability to the cell. Due to the

structure of the configuration memory cells, they are not affected by extreme power supply excursions or very high levels of alpha particle radiation. In reliability testing no soft errors have been observed, even in the presence of very high doses of alpha radiation.

### Input/Output Block

Each user-configurable I/O block (IOB) provides an interface between the external package pin of the device and the internal logic. Each I/O block includes a programmable input path and a programmable output buffer. It also provides input clamping diodes to provide protection from electro-static damage, and circuits to protect the LCA from latch-up due to input currents. Figure 3 shows the general structure of the I/O block.

The input buffer portion of each I/O block provides threshold detection to translate external signals applied to the package pin to internal logic levels. The input buffer threshold of the I/O blocks can be programmed to be

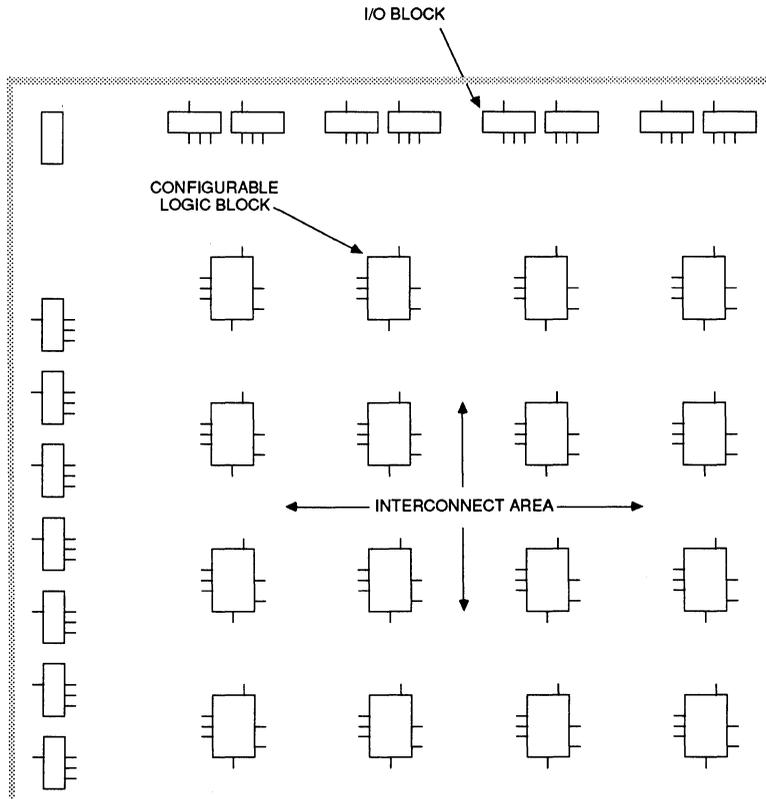


Figure 1. Logic Cell Array Structure

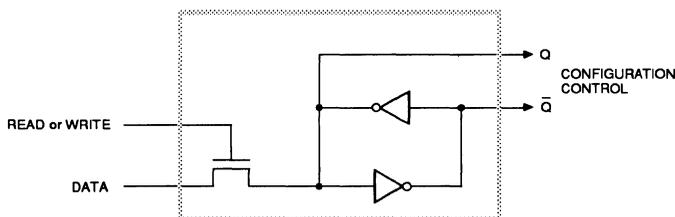
compatible with either TTL (1.4 V) or CMOS (2.2 V) levels. The buffered input signal drives both the data input of an edge triggered D flip-flop and one input of a two-input multiplexer. The output of the flip-flop provides the other input to the multiplexer. The user can select either the direct input path or the registered input, based on the content of the memory cell controlling the multiplexer. The I/O Blocks along each edge of the die share common clocks. The flip-flops are reset during configuration as well as by the active-low chip  $\overline{\text{RESET}}$  input.

Output buffers in the I/O blocks provide 4 mA drive for high fan-out CMOS or TTL compatible signal levels. The output data (driving I/O block pin O) is the data source for the I/O

block output buffer. Each I/O block output buffer is controlled by the contents of two configuration memory cells which turn the buffer ON or OFF or select logical three-state buffer control. The user may also select the output buffer three-state control (I/O block pin TS). When this I/O block output control signal is HIGH (a logic "1") the buffer is disabled and the package pin is high-impedance.

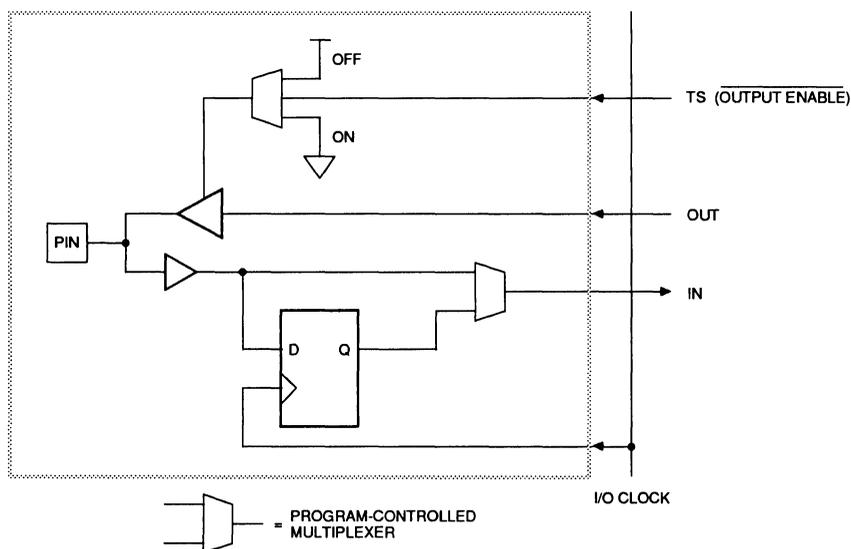
### Configurable Logic Block

An array of Configurable Logic Blocks (CLBs) provides the functional elements from which the user's logic is constructed. The Logic Blocks are arranged in a matrix in the center of the device. The XC2064 has 64 such blocks



1104 02

Figure 2. Configuration Memory Cell



1104 03

Figure 3. I/O Block

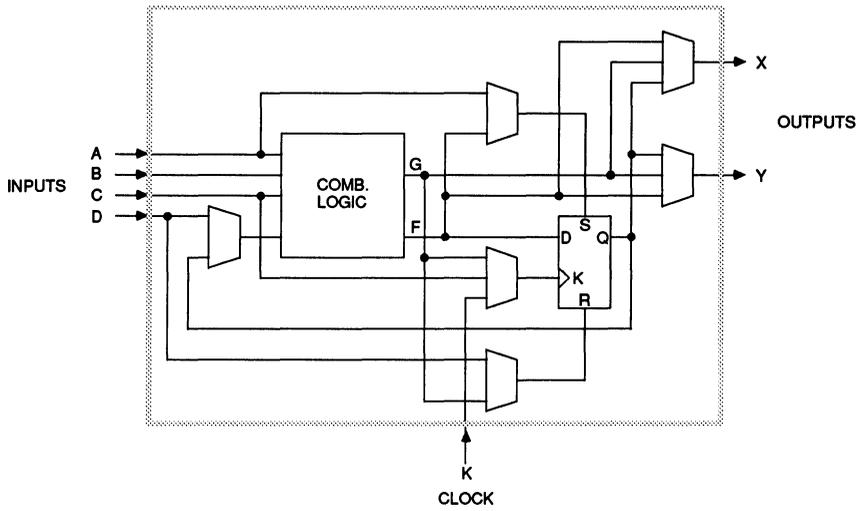


Figure 4. Configurable Logic Block

1104 04

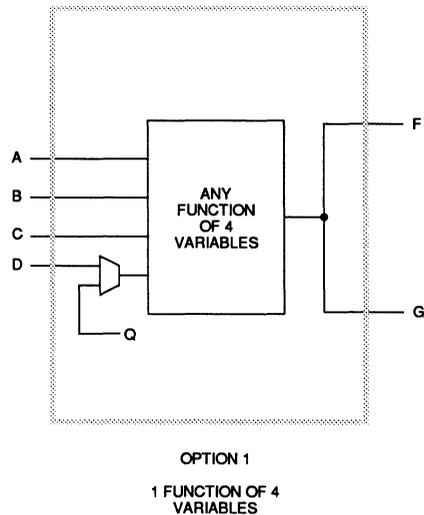
arranged in an 8-row by 8-column matrix. The XC2018 has 100 logic blocks arranged in a 10 by 10 matrix.

Each logic block has a combinatorial logic section, a storage element, and an internal routing and control section. Each CLB has four general-purpose inputs: A, B, C and D; and a special clock input (K), which may be driven from the interconnect adjacent to the block. Each CLB also has two outputs, X and Y, which may drive interconnect networks. Figure 4 shows the resources of a Configurable Logic Block.

The logic block combinatorial logic uses a table look-up memory to implement Boolean functions. This technique can generate any logic function of up to four variables with a high speed sixteen-bit memory. The propagation delay through the combinatorial network is independent of the function generated. Each block can perform any function of four variables or any two functions of three variables each. The variables may be selected from among the four inputs and the block's storage element output "Q". Figure 5 shows various options which may be specified for the combinatorial logic.

If the single four-variable configuration is selected (Option 1), the F and G outputs are identical. If the two-function alternative is selected (Option 2), logic functions F and G may be independent functions of three variables each. The three variables can be selected from among the four logic block inputs and its storage element output "Q". A

third form of the combinatorial logic (Option 3) is a special case of the two-function form in which the B input dynamically selects between the two function tables providing a single merged logic function output. This dynamic selec-



tion allows some five-variable functions to be generated from the four block inputs and storage element Q. Combinatorial functions are restricted in that one may not use both its storage element output Q and the input variable of the logic block pin "D" in the same function.

If used, the storage element in each Configurable Logic Block (Figure 6) can be programmed to be either an edge-sensitive "D" type flip-flop or a level-sensitive "D" latch. The clock or enable for each storage element can be selected from:

- The special-purpose clock input K
- The general-purpose input C
- The combinatorial function G

The user may also select the clock active sense within each logic block. This programmable inversion eliminates the need to route both phases of a clock signal throughout the device.

The storage element data input is supplied from the function F output of the combinatorial logic. Asynchronous SET and RESET controls are provided for each storage element. The user may enable these controls independently and select their source. They are active

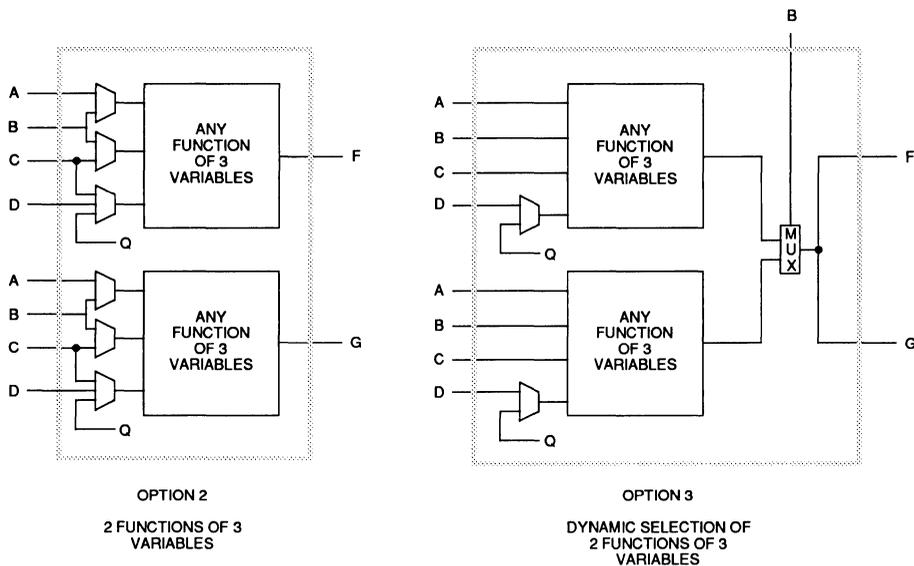
high inputs and the asynchronous reset is dominant. The storage elements are reset by the active-low chip RESET pin as well as by the initialization phase preceding configuration. If the storage element is not used, it is disabled.

The two block outputs, X and Y, can be driven by either the combinatorial functions, F or G, or the storage element output Q (Figure 4). Selection of the outputs is completely interchangeable and may be made to optimize routing efficiencies of the networks interconnecting the logic blocks and I/O blocks.

### Programmable Interconnect

Programmable interconnection resources in the Logic Cell Array provide routing paths to connect inputs and outputs of the I/O and logic blocks into desired networks. All interconnections are composed of metal segments, with programmable switching points provided to implement the necessary routing. Three types of resources accommodate different types of networks:

- General purpose interconnect
- Long lines
- Direct connection



1104 05

**Figure 5. CLB Combinatorial Logic Options**

Note: Variables D and Q can not be used in the same function.

**General-Purpose Interconnect**

General-purpose interconnect, as shown in Figure 7a, is composed of four horizontal metal segments between the rows and five vertical metal segments between the columns of logic and I/O blocks. Each segment is only the "height" or "width" of a logic block. Where these segments would cross at the intersections of rows and columns, switching matrices are provided to allow interconnections of metal segments from the adjoining rows and columns. Switches in the switch matrices and on block outputs are specially designed transistors, each controlled by a configuration bit.

Logic block output switches provide contacts to adjacent general interconnect segments and therefore to the switching matrix at each end of those segments. A switch matrix can connect an interconnect segment to other segments to form a network. Figure 7a shows the general interconnect used to route a signal from one logic block to three other logic blocks. As shown, combinations of closed switches in a switch matrix allow multiple branches for each network. The inputs of the logic or I/O blocks are multiplexers that can be program-med with configuration bits to select an input network from the adjacent interconnect segments. Since the switch connections to block inputs are unidirectional (as are block outputs) they are usable *only* for input connection. The development system software provides automatic routing of these interconnections. Interactive routing is also available for design optimization. This is accomplished by selecting a network

and then toggling the states of the interconnect points by selecting them with the "mouse". In this mode, the connections through the switch matrix may be established by selecting pairs of matrix pins. The switching matrix combinations are indicated in Figure 7b.

Special buffers within the interconnect area provide periodic signal isolation and restoration for higher general interconnect fan-out and better performance. The repowering buffers are bidirectional, since signals must be able to propagate in either direction on a general interconnect segment. Direction controls are automatically established by the Logic Cell Array development system software. Repowering buffers are provided only for the general-purpose interconnect since the direct and long line resources do not exhibit the same R-C delay accumulation. The Logic Cell Array is divided into nine sections with buffers automatically provided for general interconnect at the boundaries of these sections. These boundaries can be viewed with the development system. For routing within a section, no buffers are used. The delay calculator of the XACT development system automatically calculates and displays the block, interconnect and buffer delays for any selected paths.

1104 06

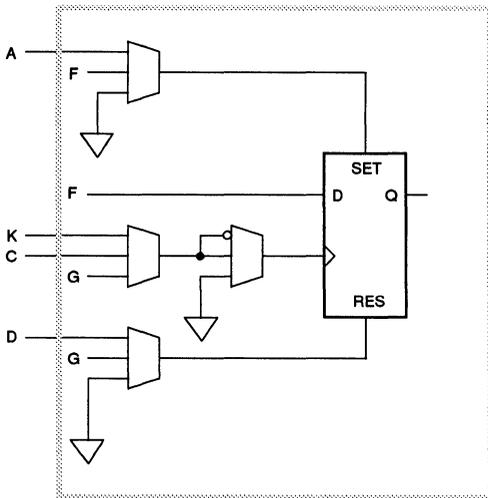
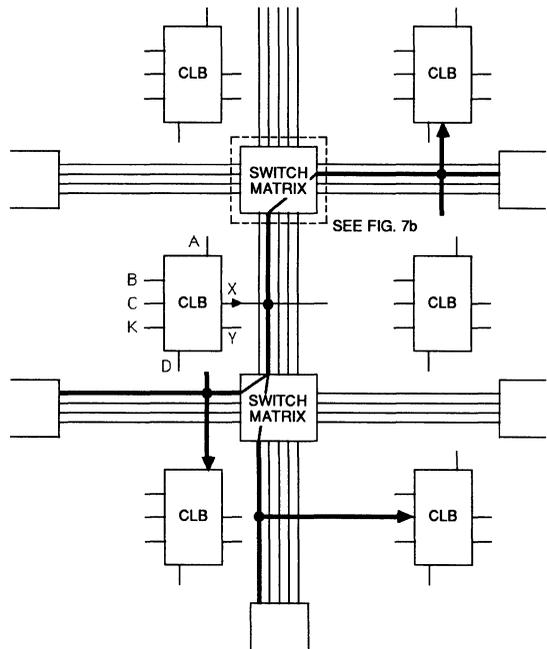


Figure 6. CLB Storage Element



1104 07

Figure 7a. General-Purpose Interconnect

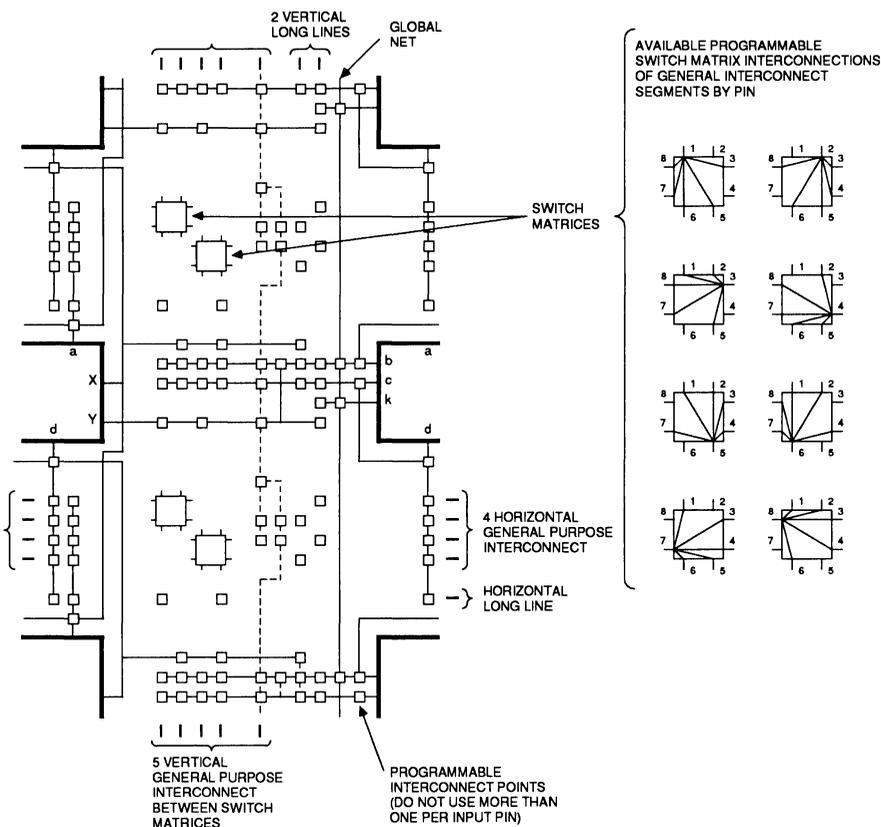
## Long Lines

Long-lines, shown in Figure 8a, run both vertically and horizontally the height or width of the interconnect area. Each vertical interconnection column has two long lines; each horizontal row has one, with an additional long line adjacent to each set of I/O blocks. The long lines bypass the switch matrices and are intended primarily for signals that must travel a long distance or must have minimum skew among multiple destinations.

A global buffer in the Logic Cell Array is available to drive a single signal to all B and K inputs of logic blocks. Using

the global buffer for a clock provides a very low skew, high fan-out synchronized clock for use at any or all of the logic blocks. At each block, a configuration bit for the K input to the block can select this global line as the storage element clock signal. Alternatively, other clock sources can be used.

A second buffer below the bottom row of the array drives a horizontal long line which, in turn, can drive a vertical long line in each interconnection column. This alternate buffer also has low skew and high fan-out capability. The network formed by this alternate buffer's long lines can be selected to drive the B, C or K inputs of the logic blocks.



1104 08

Figure 7b. Routing and Switch Matrix Connections

Alternatively, these long lines can be driven by a logic or I/O block on a column by column basis. This capability provides a common, low-skew clock or control line within each column of logic blocks. Interconnections of these long lines are shown in Figure 8b.

### Direct Interconnect

Direct interconnect, shown in Figure 9, provides the most efficient implementation of networks between adjacent logic or I/O blocks. Signals routed from block to block by means of direct interconnect exhibit minimum interconnect propagation and use minimum interconnect resources. For each Configurable Logic Block, the X output may be connected directly to the C or D inputs of the CLB above and to the A or B inputs of the CLB below it. The Y output can use direct interconnect to drive the B input of the block immediately to its right. Where logic blocks are adjacent to I/O blocks, direct connect is provided to the I/O block input (I) on the left edge of the die, the output (O) on the right edge, or both on I/O blocks at the top and

bottom of the die. Direct interconnections of I/O blocks with CLBs are shown in Figure 8b.

### Crystal Oscillator

An internal high speed inverting amplifier is available to implement an on-chip crystal oscillator. It is associated with the auxiliary clock buffer in the lower right corner of the die. When configured to drive the auxiliary clock buffer, two special adjacent user I/O blocks are also configured to connect the oscillator amplifier with external crystal oscillator components, as shown in Figure 10. This circuit becomes active before configuration is complete in order to allow the oscillator to stabilize. Actual internal connection is delayed until completion of configuration. The feedback resistor R1 between output and input, biases the amplifier at threshold. It should be as large a value as practical to minimize loading of the crystal. The inversion of the amplifier, together with the R-C networks and crystal, produce the 360-degree phase shift of the Pierce oscillator. A series resistor R2 may be included to add to

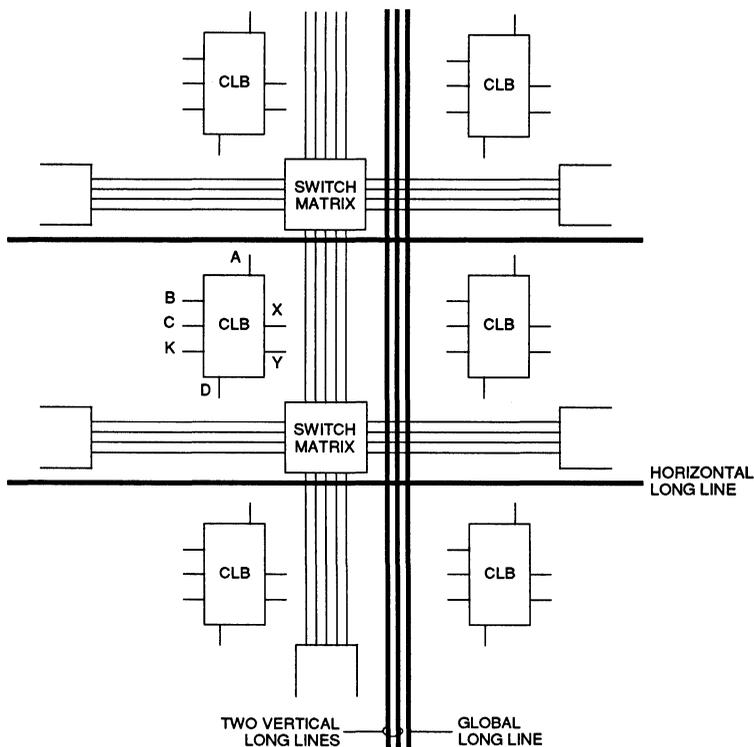


Figure 8a. Long Line Interconnect

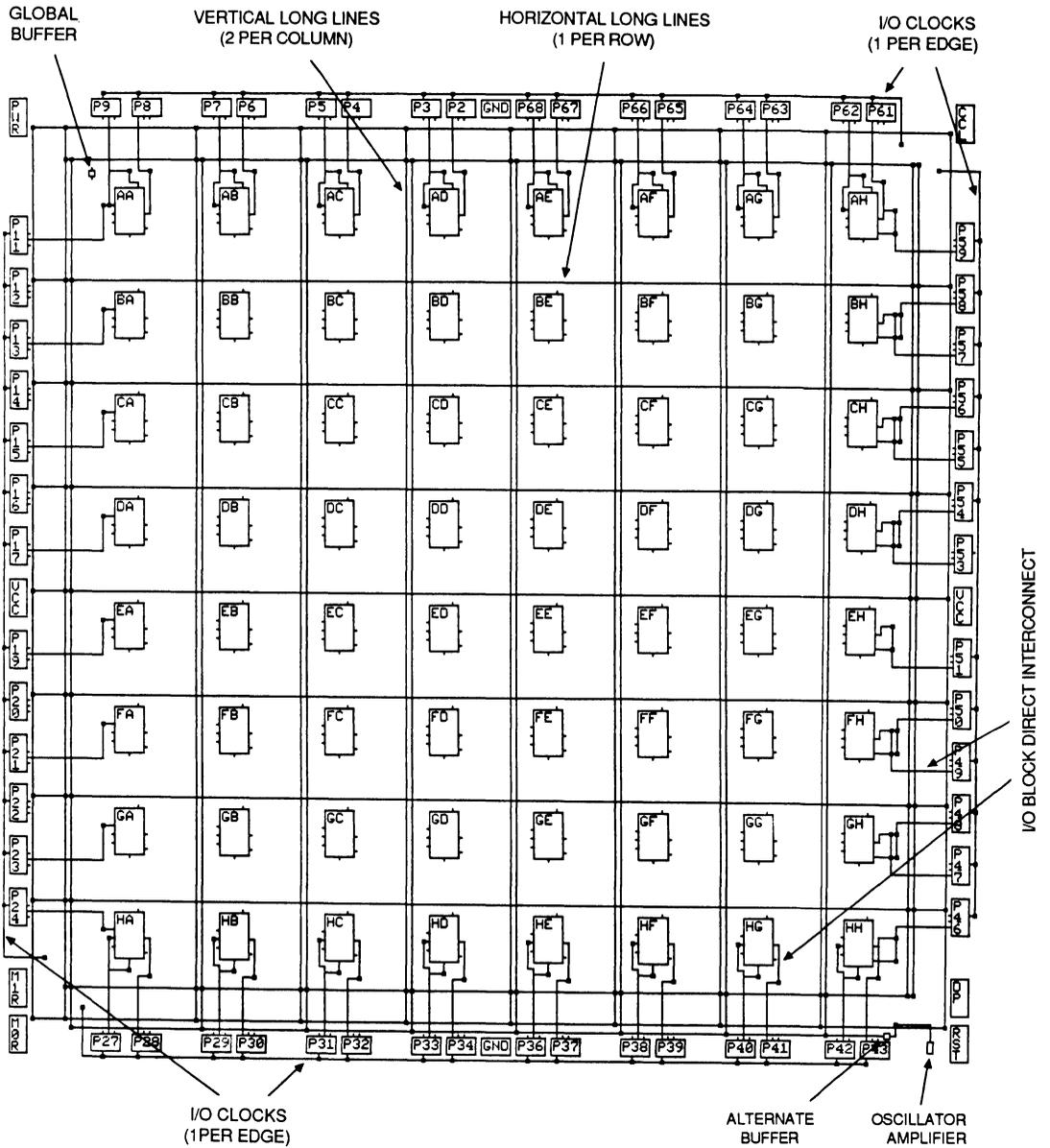


Figure 8b. XC2064 Long Lines, I/O Clocks, I/O Direct Interconnect

the amplifier output impedance when needed for phase-shift control or crystal resistance matching or to limit the amplifier input swing to control clipping at large amplitudes. Excess feedback voltage may be adjusted by the ratio of  $C2/C1$ . The amplifier is designed to be used over the range from 1 MHz up to one-half the specified CLB toggle frequency. Use at frequencies below 1 MHz may require individual characterization with respect to a series resistance. Operation at frequencies above 20 MHz generally requires a crystal to operate in a third overtone mode, in which the fundamental frequency must be suppressed by the R-C networks. When the amplifier does not drive the auxiliary buffer, these I/O blocks and their package pins are available for general user I/O.

## POWER

### Power Distribution

Power for the LCA is distributed through a grid to achieve high noise immunity and isolation between logic and I/O.

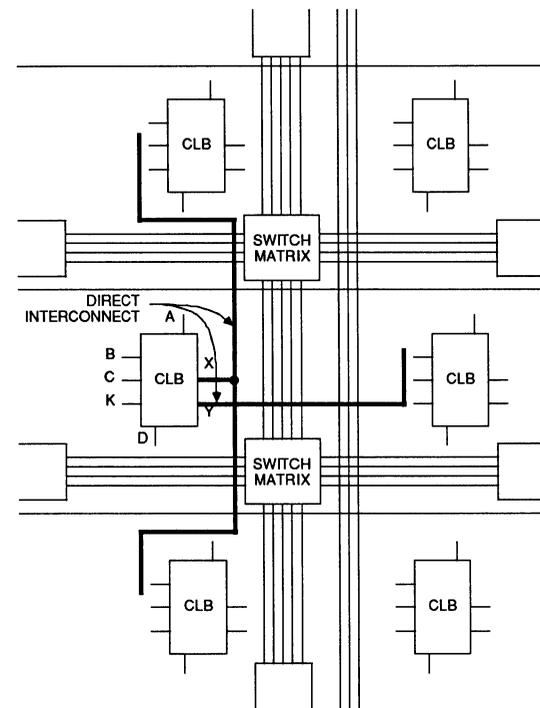


Figure 9. Direct Interconnect

1104 10

For packages having more than 48 pins, two  $V_{cc}$  pins and two ground pins are provided (see Figure 11). Inside the LCA, a dedicated  $V_{cc}$  and ground ring surrounding the logic array provides power to the I/O drivers. An independent matrix of  $V_{cc}$  and ground lines supplies the interior logic of the device. This power distribution grid provides a stable supply and ground for all internal logic, providing the external package power pins are appropriately decoupled. Typically a  $0.1 \mu\text{F}$  capacitor connected between the  $V_{cc}$  and ground pins near the package will provide adequate decoupling.

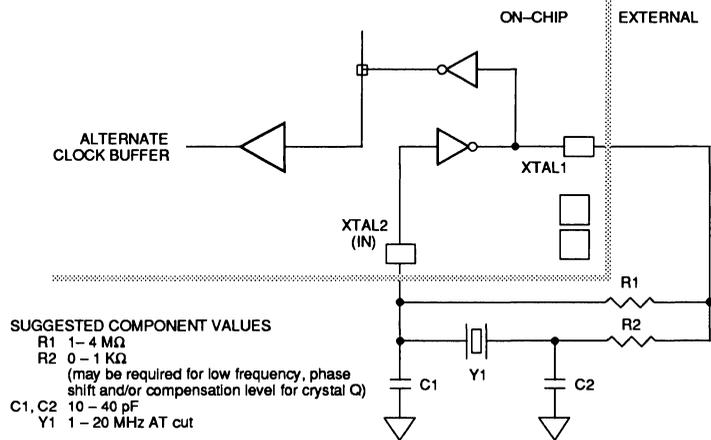
Output buffers capable of driving the specified 4 mA loads under worst-case conditions may be capable of driving 25 to 30 times that current in a best case. Noise can be reduced by minimizing external load capacitance and reducing simultaneous output transitions in the same direction. It may also be beneficial to locate heavily loaded output buffers near the ground pads. Multiple  $V_{cc}$  and ground pin connections are required for package types which provide them.

### Power Dissipation

The Logic Cell Array exhibits the low power consumption characteristic of CMOS ICs. Only quiescent power is required for the LCA configured for CMOS input levels. The TTL input level configuration option requires additional power for level shifting. The power required by the static memory cells which hold the configuration data is very low and may be maintained in a power-down mode.

Typically most of power dissipation is produced by capacitive loads on the output buffers, since the power per output is  $25 \mu\text{W} / \text{pF} / \text{MHz}$ . Another component of I/O power is the DC loading on each output pin. For any given system, the user can calculate the I/O power requirement based on the sum of capacitive and resistive loading of the devices driven by the Logic Cell Array.

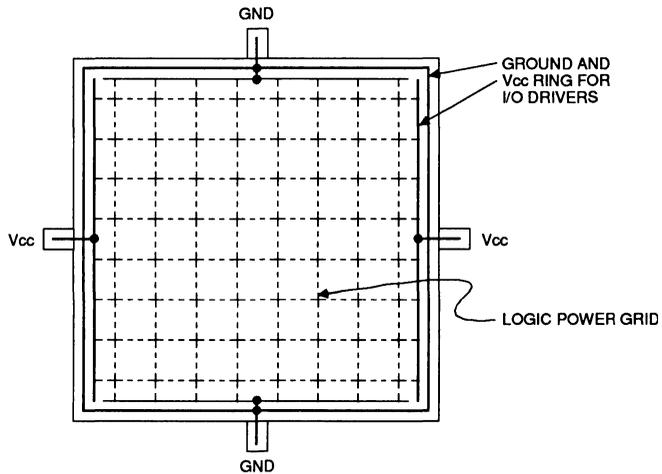
Internal power supply dissipation is a function of clock frequency and the number of nodes changing on each clock. In an LCA the fraction of nodes changing on a given clock is typically low (10–20%). For example, in a 16-bit binary counter, the average clock produces a change in slightly less than 2 of the 16 bits. In a 4-input AND gate there will be 2 transitions in 16 states. Typical global clock buffer power is about  $3 \text{ mW} / \text{MHz}$  for the XC2064 and  $4 \text{ mW} / \text{MHz}$  for the XC2018. With a “typical” load of three general interconnect segments, each Configurable Logic Block output requires about  $0.4 \text{ mW} / \text{MHz}$  of its output frequency. Graphs of power versus operating frequency are shown in Table 1.



	XTAL1	XTAL2
48 DIP	33	30
68 PLCC	46	43
68 PGA	J10	L10
84 PLCC	56	53
84 PGA	K11	L11

1104 11

Figure 10. Crystal Oscillator



1104 12

Figure 11. LCA Power Distribution

**PROGRAMMING**

Configuration data to define the function and interconnection within a Logic Cell Array are loaded automatically at power-up or upon command. Several methods of automatically loading the required data are designed into the Logic Cell Array and are determined by logic levels applied to mode selection pins at configuration time. The form of the data may be either serial or parallel, depending on the configuration mode. The programming data are independent of the configuration mode selected. The state diagram of Figure 12 illustrates the configuration process.

Input thresholds for user I/O pins can be selected to be either TTL-compatible or CMOS-compatible. At power-up, all inputs are TTL-compatible and remain in that state until the LCA begins operation. If the user has selected CMOS compatibility, the input thresholds are changed to CMOS levels during configuration.

Figure 13 shows the specific data arrangement for the XC2064 device. Future products will use the same data format to maintain compatibility between different devices of the Xilinx product line, but they will have different sizes and numbers of data frames. For the XC2064, configuration requires 12,038 bits for each device. For the XC2018, the configuration of each device requires 17,878 bits. The XC2064 uses 160 configuration data frames and the XC2018 uses 197.

The configuration bit stream begins with preamble bits, a preamble code and a length count. The length count is loaded into the control logic of the Logic Cell Array and is used to determine the completion of the configuration process. When configuration is initiated, a 24-bit length counter is set to 0 and begins to count the total number of configuration clock cycles applied to the device. When the current length count equals the loaded length count, the configuration process is complete. Two clocks before completion, the internal logic becomes active and is reset. On the next clock, the inputs and outputs become active as configured and consideration should be given to avoid configuration signal contention. *(Attention must be paid to avoid contention on pins which are used as inputs during configuration and become outputs in operation.)* On the last configuration clock, the completion of configuration is signalled by the release of the DONE / PROG pin of the device as the device begins operation. This open-drain output can be AND-tied with multiple Logic Cell Arrays and used as an active-high READY or active-low , RESET, to other portions of the system. High during configuration (HDC) and low during configuration (LDC), are released one CCLK cycle before DONE is asserted. In master mode configurations, it is convenient to use LDC as an active-low EPROM chip enable.

As each data bit is supplied to the LCA, it is internally assembled into a data word. As each data word is completely assembled, it is loaded in parallel into one word of the internal configuration memory array. The last word must be loaded before the current length count compare is true. If the configuration data are in error, e.g., PROM address lines swapped, the LCA will not be ready at the length count and the counter will cycle through an additional complete count prior to configuration being "done".

Figure 14 shows the selection of the configuration mode based on the state of the mode pins M0 and M1. These package pins are sampled prior to the start of the configuration process to determine the mode to be used. Once configuration is DONE and subsequent operation has begun, the mode pins may be used to perform data readback, as discussed later. An additional mode pin, M2, must be defined at the start of configuration. This package pin is a user-configurable I/O after configuration is complete.

**Initialization Phase**

When power is applied, an internal power-on-reset circuit is triggered. When Vcc reaches the voltage at which the LCA begins to operate (2.5 to 3 Volts), the chip is initialized, outputs are made high-impedance and a time-out is initiated to allow time for power to stabilize. This time-out (15 to 35 ms) is determined by a counter driven by a self-generated, internal sampling clock that drives the configuration clock (CCLK) in master configuration mode. This internal sampling clock will vary with process, temperature and power supply over the range of 0.5 to 1.5 MHz. LCAs with mode lines set for master mode will time-out of their initialization using a longer counter (60 to 140 ms) to assure that all devices, which it may be driving in a daisy chain, will be ready. Configuration using peripheral or

1104 13

MODE PIN			MODE SELECTED
M0	M1	M2	
0	0	0	MASTER SERIAL
0	0	1	MASTER LOW MODE
0	1	1	MASTER HIGH MODE
1	0	1	PERIPHERAL MODE
1	1	1	SLAVE MODE

MASTER LOW ADDRESSES BEGIN AT 0000 AND INCREMENT  
 MASTER HIGH ADDRESSES BEGIN AT FFFF AND DECREMENT

**Figure 14. Configuration Mode Selection**

slave modes must be delayed long enough for this initialization to be completed.

The initialization phase may be extended by asserting the active-low external  $\overline{\text{RESET}}$ . If a configuration has begun, an assertion of  $\overline{\text{RESET}}$  will initiate an abort, including an orderly clearing of partially loaded configuration memory bits. After about 3 clock cycles for synchronization, initialization will require about 160 additional cycles of the inter-

nal sampling clock (197 for the XC2018) to clear the internal memory before another configuration may begin. The same is true of a configured part in which the reconfigurable control bit is set. When a HIGH-to-LOW transition on the  $\text{DONE} / \overline{\text{PROG}}$  package pin is detected, thereby initiating a reprogram, the configuration memory is cleared. This insures an orderly configuration in which no internal signal conflicts are generated during the loading process.

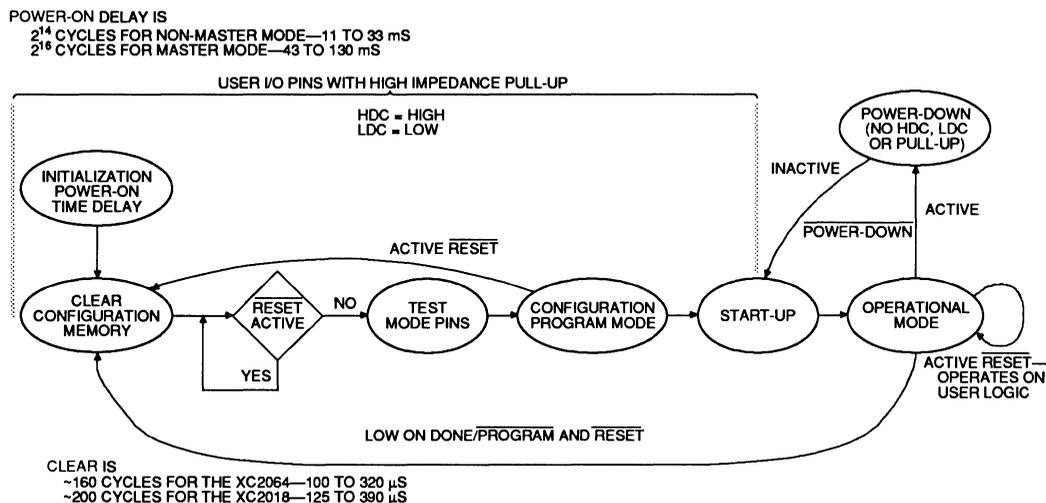


Figure 12. A State Diagram of the Configuration Process for Power-up and Re-program

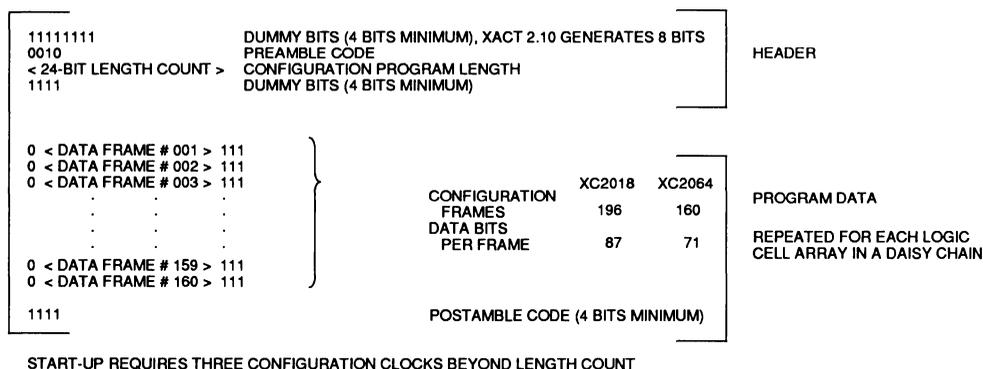


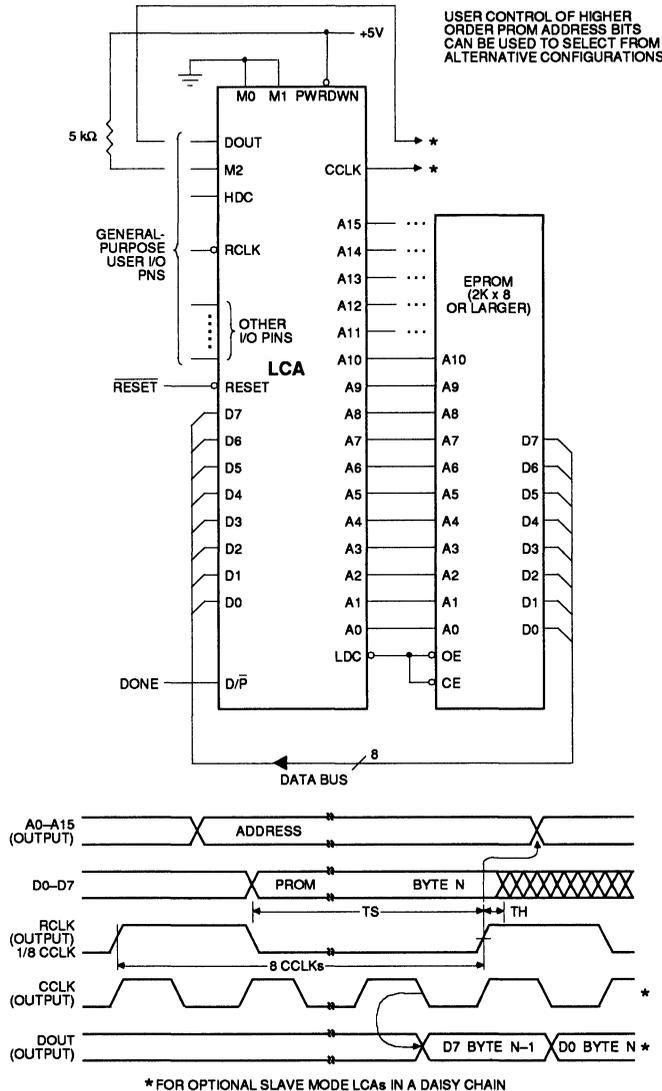
Figure 13. XC2064 Internal Configuration Data Arrangement

**Master Mode**

In master mode, the Logic Cell Array automatically loads the configuration program from an external memory device. Figure 15a shows an example of the master mode connections required. The Logic Cell Array provides sixteen address outputs and the control signals **RCLK** (read clock), **HDC** (high during configuration) and **LDC** (low during configuration) to execute read cycles from the external memory. Parallel eight-bit data words are read and internally serialized. As each data word is read, the

least significant bit of each byte, normally D0, is the next bit in the serial stream.

Addresses supplied by the Logic Cell Array can be selected by the mode lines to begin at address 0 and incremented to read the memory (master low mode), or they can begin at address FFFF Hex and be decremented (master high mode). This capability is provided to allow the Logic Cell Array to share external memory with another device, such as a microprocessor. For example, if the processor begins its execution from low memory, the Logic



**Figure 15a. Master Parallel Mode.** Configuration data are loaded automatically from an external byte wide PROM. An XC2000 LDC signal can provide a PROM inhibit as the user I/Os become active.

Cell Array can load itself from high memory and enable the processor to begin execution once configuration is completed. The DONE /  $\overline{\text{PROG}}$  output pin can be used to hold the processor in a Reset state until the Logic Cell Array has completed the configuration process.

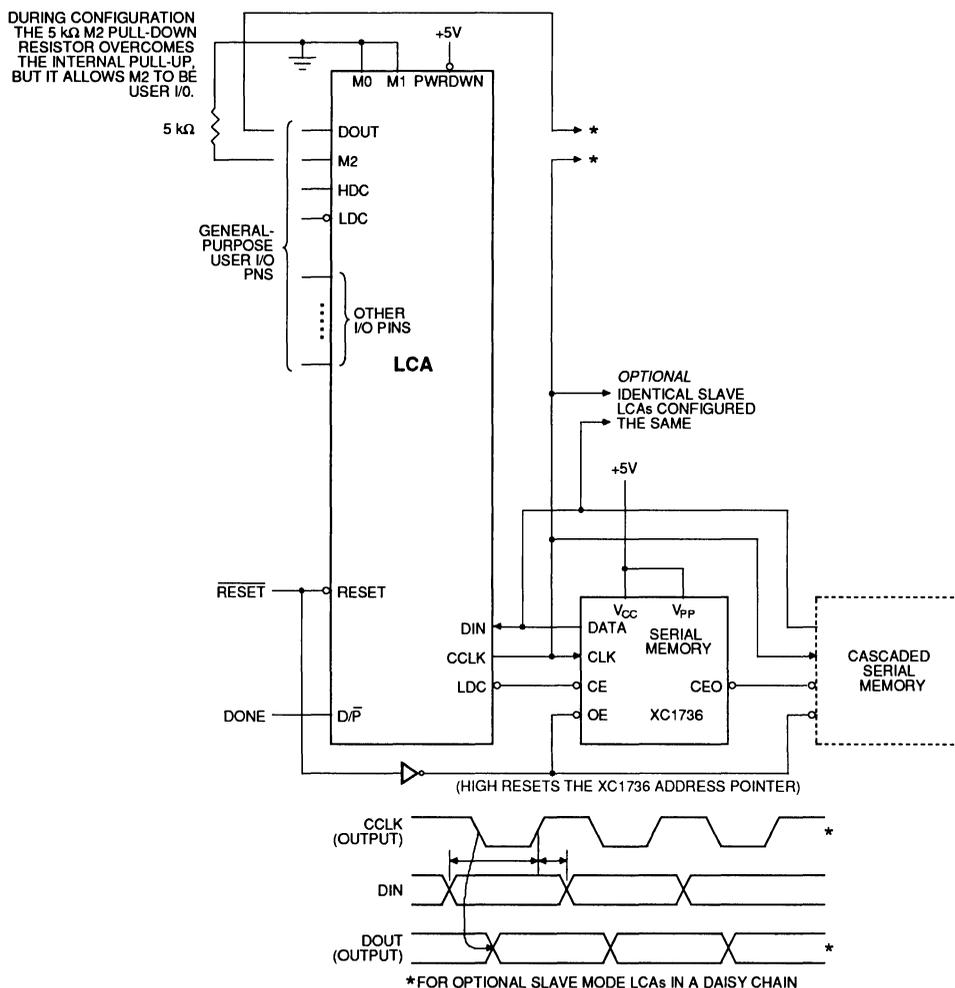
The master serial mode uses serial configuration data, synchronized by the rising edge of RCLK, as in Figure 15b.

### Peripheral Mode

Peripheral mode provides a simplified interface through which the device may be loaded as a processor peripheral.

Figure 16 shows the peripheral mode connections. Processor write cycles are decoded from the common assertion of the active-low write strobe ( $\overline{\text{WRT}}$ ), and two active-low and one active-high chip selects ( $\overline{\text{CS0}}$   $\overline{\text{CS1}}$   $\overline{\text{CS2}}$ ). If all these signals are not available, the unused inputs should be driven to their respective active levels. The Logic Cell Array will accept one bit of the configuration program on the data input (DIN) pin for each processor write cycle. Data is supplied in the serial sequence described earlier.

Since only a single bit from the processor data bus is loaded per cycle, the loading process involves the processor reading a byte or word of data, writing a bit of the data to the Logic Cell Array, shifting the word and writing a bit until all bits of the word are written, then continuing in the



1104 17

**Figure 15b. Master Serial Mode.** The one time programmable XC1736 Serial Configuration PROM supports automatic loading of configuration programs up to 36K bits. Multiple XC1736s can be cascaded to support additional LCAs. An XC2000 LDC signal can provide an XC1736 inhibit as the user I/Os become active.

same fashion with the next word, etc. After the configuration program has been loaded, an additional three clocks (a total of three more than the length count) must be supplied in order to complete the configuration process. When more than one device is being used in the system, each device can be assigned a different bit in the processor data bus, and multiple devices can be loaded on each processor write cycle. This "broadside" loading method provides a very easy and time-efficient method of loading several devices.

**Slave Mode**

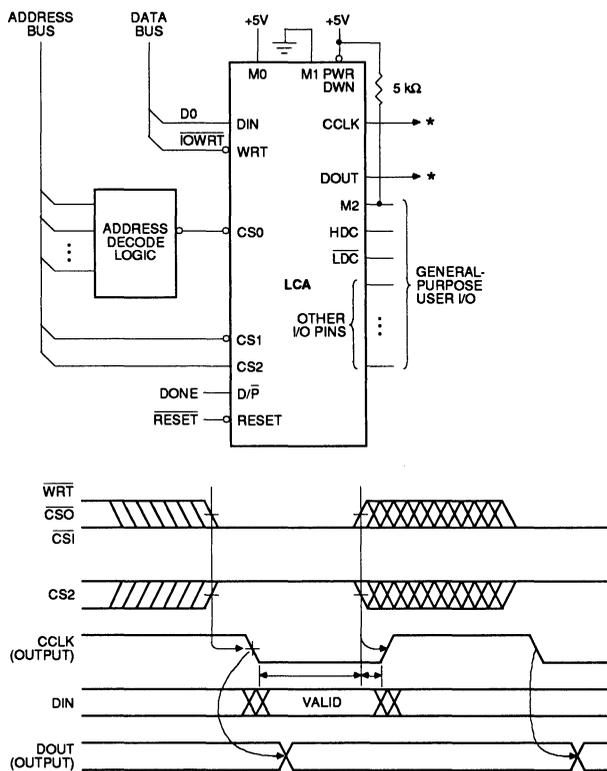
Slave mode, Figure 17, provides the simplest interface for loading the Logic Cell Array configuration. Data is supplied in conjunction with a synchronizing clock. For each LOW-to-HIGH input transition of configuration clock (CCLK), the data present on the data input (DIN) pin is loaded into the internal shift register. Data may be supplied by a processor or by other special circuits. Slave mode is used for downstream devices in a daisy-chain configuration. The data for each slave LCA are supplied by the preceding LCA in the chain, and the clock is

supplied by the lead device, which is configured in master or peripheral mode. After the configuration program has been loaded, an additional three clocks (a total of three more than the length count) must be supplied in order to complete the configuration process.

**Daisy Chain**

The daisy-chain programming mode is supported by Logic Cell Arrays in all programming modes. In master mode and peripheral mode, the LCA can act as a source of data and control for slave devices. For example, Figure 18 shows a single device in master mode, with 2 devices in slave mode. The master mode device reads the external memory and begins the configuration loading process for all of the devices.

The data begin with a preamble and a length count which are supplied to all devices at the beginning of the configuration. The length count represents the total number of cycles required to load all of the devices in the daisy chain. After loading the length count, the lead device will load its configuration data while providing a HIGH DOUT to down-



\*FOR OPTIONAL SLAVE MODE LCAs IN A DAISY CHAIN

**Figure 16. XC2000 Peripheral Mode.** Configuration data are loaded using serialized data from a microprocessor.

stream devices. When the lead device has been loaded and the current length count has not reached the full value, memory access continues. Data bytes are read and serialized by the lead device. The data are passed through the lead device and appear on the data out (DOUT) pin in serial form. The lead device also generates the configuration clock (CCLK) to synchronize the serial output data. A master mode device generates an internal CCLK of 8 times the EPROM address rate, while a peripheral mode device produces CCLK from the chip select and write strobe timing.

## Operation

When all of the devices have been loaded and the length count is complete, a synchronous start-up of operation is performed. On the clock cycle following the end of loading, the internal logic begins functioning in the reset state. On the next CCLK, the configured output buffers become active to allow signals to stabilize. The next CCLK cycle produces the DONE condition. The length count control of operation allows a system of multiple Logic Cell Arrays to begin operation in a synchronized fashion. If the crystal oscillator is used, it will begin operation before configuration is complete to allow time for stabilization before it is

connected to the internal circuitry.

## SPECIAL CONFIGURATION FUNCTIONS

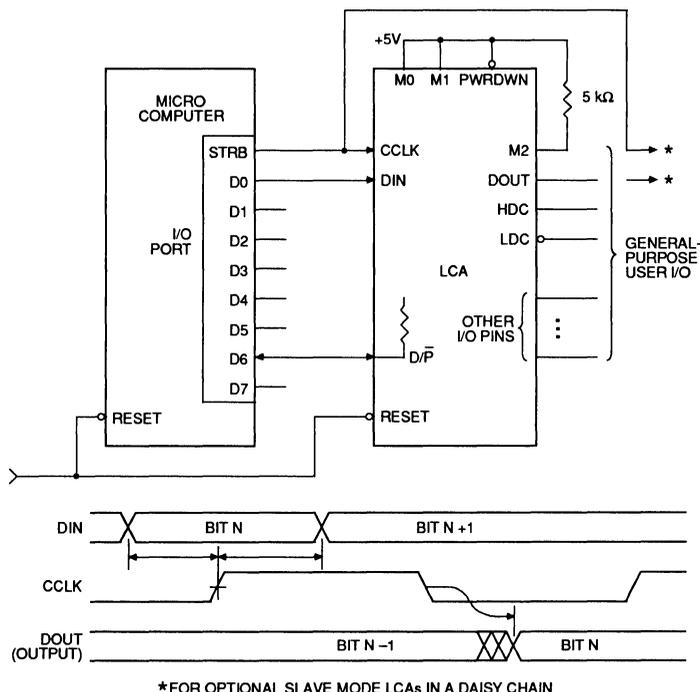
In addition to the normal user logic functions and interconnect, the configuration data include control for several special functions:

- Input thresholds
- Readback enable
- Reprogram
- DONE pull-up resistor

Each of these functions is controlled by a portion of the configuration program generated by the XACT Development System.

### Input Thresholds

During configuration, all input thresholds are TTL level. During configuration input thresholds are established as specified, either TTL or CMOS. The  $\overline{\text{PWRDWN}}$  input threshold is an exception; it is always a CMOS level input. The TTL threshold option requires additional power for threshold shifting.



1104 19

**Figure 17. Slave Mode.** Bit-serial configuration data are read at rising edge of the CCLK. Data on DOUT are provided on the falling edge of CCLK. Identically configured non-master mode LCAs can be configured in parallel by connecting DINs and CCLKs.

## Readback

After a Logic Cell Array has been programmed, the configuration program may be read back from the device. Readback may be used for verification of configuration, and as a method of determining the state of internal logic nodes during debugging. Three readback options are provided: on command, only once, and never.

An initiation of readback is produced by a LOW-to-HIGH transition of the M0 / RTRIG (read trigger) pin. Once the readback command has been given, CCLK is cycled to read back each data bit in a format similar to loading. After two dummy bits, the first data frame is shifted out, in inverted sense, on the M1 /  $\overline{\text{RDATA}}$  (read data) pin. All data frames must be read back to complete the process and return the mode select and CCLK pins to their normal functions. Read back data includes the state of all internal storage elements. This information is used by the Logic Cell Array development system In-Circuit Debugger to provide visibility into the internal operation of the logic while the system is operating. To read back a uniform time sample of all storage elements, it may be necessary to inhibit the system clock.

## Re-program

The Logic Cell Array configuration memory may be rewritten while the device is operating in the user's system. The LCA returns to the Clear state where the configuration memory is cleared, I/O pins disabled, and mode lines re-sampled. Re-program control is often implemented using an external open collector driver which pulls DONE/PROG LOW. Once it recognizes a stable request, the Logic Cell Array will hold DONE/PROG LOW until the new configuration has been completed. Even if the DONE/PROG pin is externally held LOW beyond the configuration period, the Logic Cell Array will begin operation upon completion of configuration. To reduce sensitivity to noise, these re-program signals are filtered for 2–3 cycles of the LCA's internal timing generator (2 to 6  $\mu\text{s}$ ). Note that the Clear time out for a Master mode re-program or abort does not have the 4 times delay of the Initialization state. If a daisy chain is used, an external RESET is required, long enough to guarantee clearing all non-master mode devices. For XC2000 series LCAs this is accomplished with an external time delay.

In some applications the system power supply might have momentary failures which can leave the LCA's control logic in an invalid state. There are two methods to recover from this state. The first is to cycle the Vcc supply to less than 0.1 Volt and reapply valid Vcc. The second is to provide the LCA with simultaneous LOW levels of at least 6 $\mu\text{s}$  on RESET and DONE/PROG pins after the RESET pin has been HIGH following a return to valid Vcc. This

guarantees that the LCA will return to the Clear state. Either of these methods may be needed in the event of an incomplete voltage interruption. They are not needed for a normal application of power from an off condition.

## DONE Pull-up

The DONE /  $\overline{\text{PROG}}$  pin is an open drain I/O that indicates programming status. As an input, it initiates a reprogram operation. An optional internal pull-up resistor may be enabled.

## Battery Backup

Because the control store of the Logic Cell Array is a CMOS static memory, its cells require only a very low standby current for data retention. In some systems, this low data retention current characteristic facilitates preserving configurations in the event of a primary power loss. The Logic Cell Array has built in power-down logic which, when activated, will disable normal operation of the device and retain only the configuration data. All internal operation is suspended and all output buffers are placed in their high impedance state.

Power-down data retention is possible with a simple battery-backup circuit because the power requirement is extremely low. For retention at 2.0 volts, the required current is typically on the order of 50 nanoamps. Screening to this parameter is available. To force the Logic Cell Array into the power-down state, the user must pull the  $\overline{\text{PWRDWN}}$  pin LOW and continue to supply a retention voltage to the Vcc pins of the package. When normal power is restored, Vcc is elevated to its normal operating voltage and  $\overline{\text{PWRDWN}}$  is returned to a HIGH. The Logic Cell Array resumes operation with the same internal sequence that occurs at the conclusion of configuration. Internal I/O and logic block storage elements will be reset, the outputs will become enabled and then the DONE/PROG pin will be released. No configuration programming is involved.

## PERFORMANCE

The high performance of the Logic Cell Array results from its patented architectural features and from the use of an advanced high-speed CMOS manufacturing process. Performance may be measured in terms of minimum propagation times for logic elements.

Flip-flop loop delays for the I/O block and logic block flip-flops are about 3 nanoseconds. This short delay provides very good performance under asynchronous clock and data conditions. Short loop delays minimize the probability

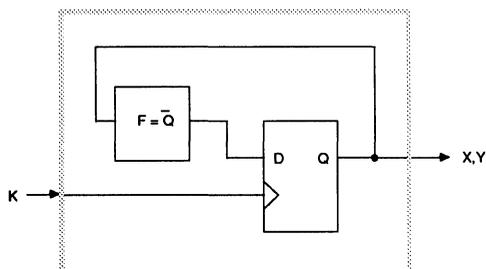


of a metastable condition which can result from assertion of the clock during data transitions. Because of the short loop delay characteristic in the Logic Cell Array, the I/O block flip-flops can be used very effectively to synchronize external signals applied to the device. Once synchronized in the I/O block, the signals can be used internally without further consideration of their clock relative timing, except as it applies to the internal logic and routing path delays.

### Device Performance

The single parameter which most accurately describes the overall performance of the Logic Cell Array is the maximum toggle rate for a logic block storage element configured as a toggle flip-flop. The configuration for determining the toggle performance of the Logic Cell Array is shown in Figure 19. The clock for the storage element is provided by the global clock buffer and the flip-flop output Q is fed back through the combinatorial logic to form the data input for the next clock edge. Using this arrangement, flip-flops in the Logic Cell Array can be toggled at clock rates from 33–70 MHz, depending on the speed grade used.

Actual Logic Cell Array performance is determined by the critical path speed, including both the speed of the logic and storage elements in that path, and the speed of the particular network routing. Figure 20 shows a typical system logic configuration of two flip-flops with an extra combinatorial level between them. Depending on speed grade, system clock rates to 35 MHz are practical for this logic. To allow the user to make the best use of the capabilities of the device, the delay calculator in the XACT Development System determines worst-case path delays using actual impedance and loading information.



1104 21

**Figure 19. Logic Block Configuration for Toggle Rate Measurement**

### Logic Block Performance

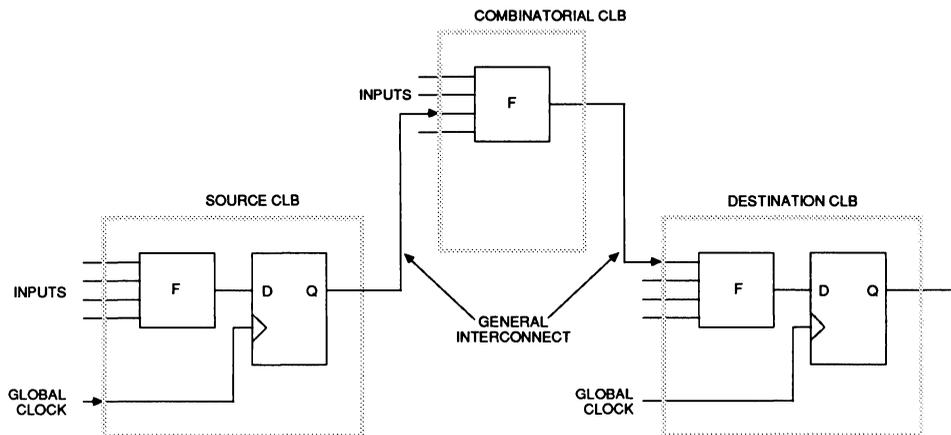
Logic block propagation times are measured from the interconnect point at the input of the combinatorial logic to the output of the block in the interconnect area. Combinatorial performance is independent of logic function because of the table look-up based implementation. Timing is different when the combinatorial logic is used in conjunction with the storage element. For the combinatorial logic function driving the data input of the storage element, the critical timing is data set-up relative to the clock edge provided to the storage element. The delay from the clock source to the output of the logic block is critical in the timing of signals produced by storage elements. The loading on a logic block output is limited only by the additional propagation delay of the interconnect network. Performance of the logic block is a function of supply voltage and temperature, as shown in Figures 22 and 23.

### Interconnect Performance

Interconnect performance depends on the routing resource used to implement the signal path. As discussed earlier, direct interconnect from block to block provides a minimum delay path for a signal.

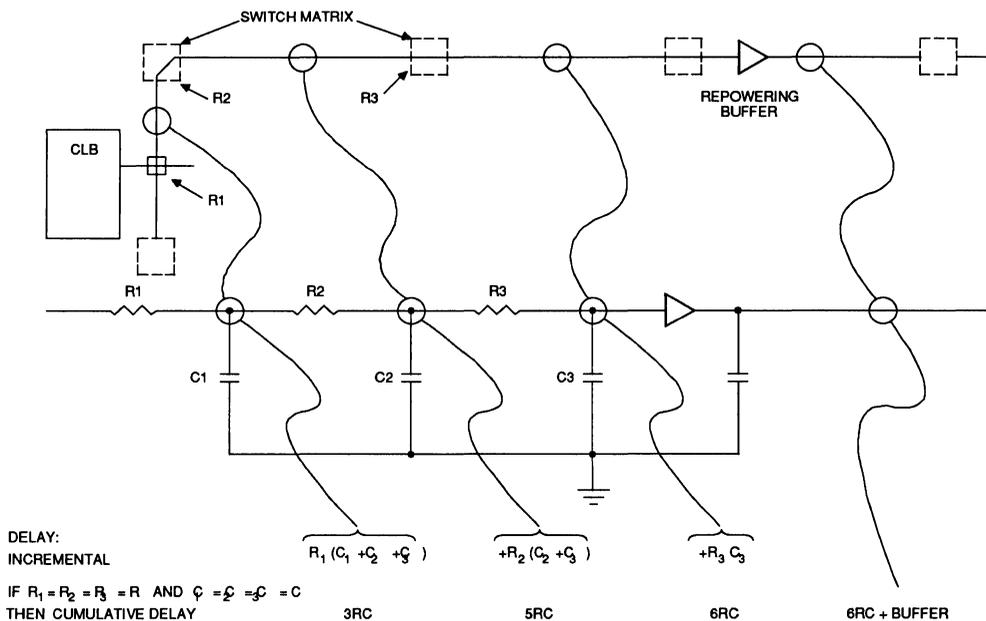
The single metal segment used for long lines exhibits low resistance from end to end, but relatively high capacitance. Signals driven through a programmable switch will have the additional impedance of the switch added to their normal drive impedance.

General-purpose interconnect performance depends on the number of switches and segments used, the presence of the bidirectional repowering buffers and the overall loading on the signal path at all points along the path. In calculating the worst-case delay for a general interconnect path, the delay calculator portion of the XACT development system accounts for all of these elements. As an approximation, interconnect delay is proportional to the summation of totals of local metal segments beyond each programmable switch. In effect, the delay is a sum of R-C delays each approximated by an R times the total C it drives. The R of the switch and the C of the interconnect are functions of the particular device performance grade. For a string of three local interconnects, the approximate delay at the first segment, after the first switch resistance, would be three units; an additional two delay units after the next switch plus an additional delay after the last switch in the chain. The interconnect R-C chain terminates at each repowering buffer. Nearly all of the capacitance is in the interconnect metal and switches; the capacitance of the block inputs is not significant. Figure 21 shows an estimation of this delay.



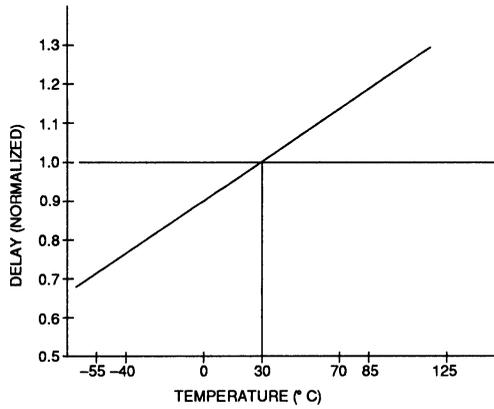
1104 22

Figure 20. Typical Logic Path



1104 23

Figure 21. Interconnection Delay Factors Example

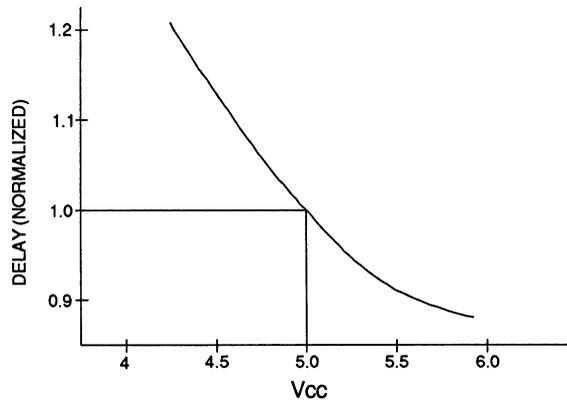


NOTE: NORMALIZED FOR 30°C

1104 25

Figure 22. Delay vs. Temperature

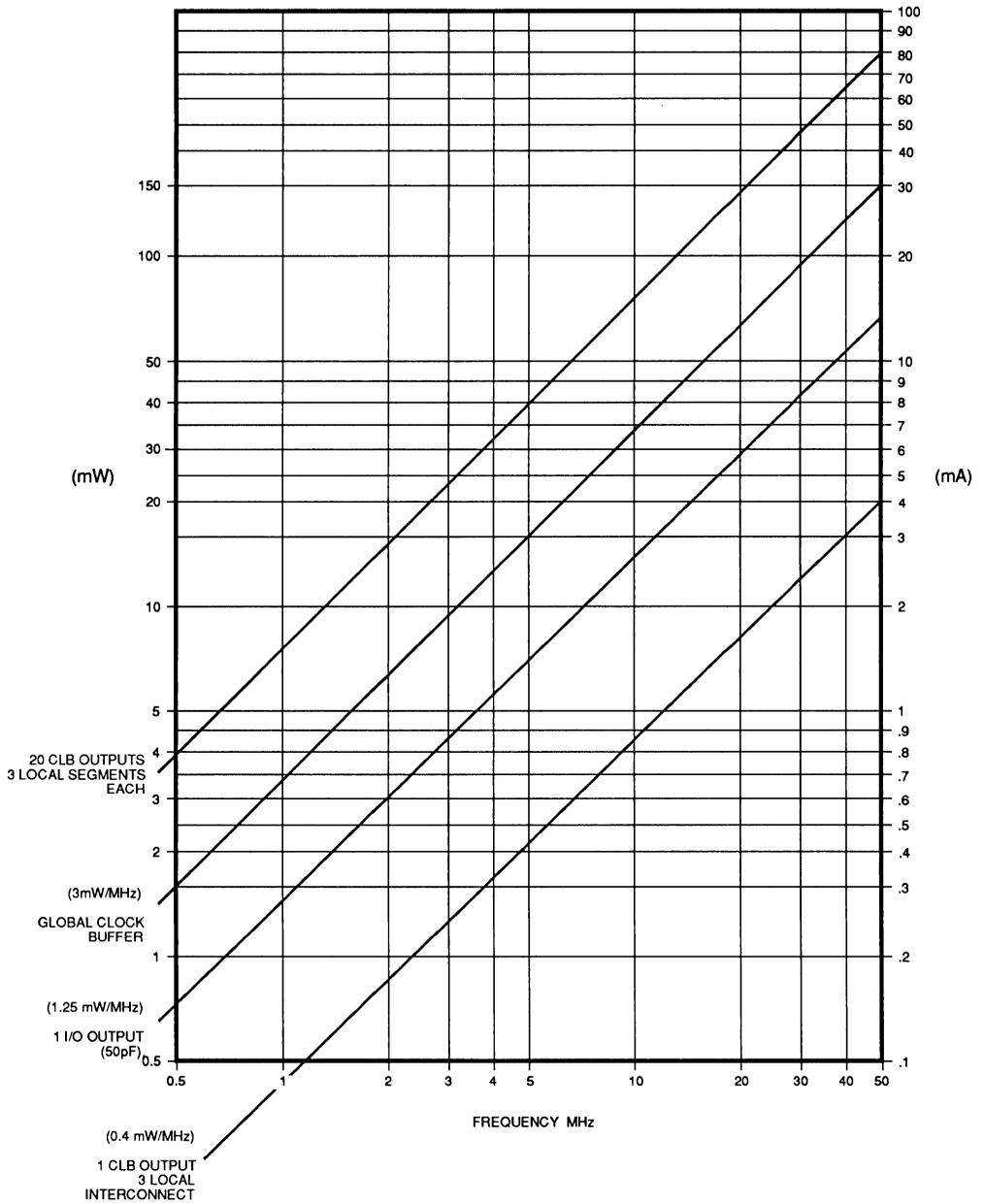
---



1104 26

Figure 23. Delay vs. Power Supply Voltage

---



1104 27

Table 1. Typical LCA Power Consumption By Element

## DEVELOPMENT SYSTEMS

To accomplish hardware development support for the Logic Cell Array, Xilinx provides a development system with several options to support added capabilities. The XACT system provides the following:

- Schematic entry
- Automatic place and route
- Interactive design editing for optimization
- Interactive timing calculations
- Macro library support, both for standard Xilinx supplied functions and user defined functions
- Design entry checking for consistency and completeness
- Automatic design documentation generation
- PROM programmer format output capabilities
- Simulation interface support including automatic netlist (circuit description) and timing extraction
- Logic and timing simulation
- In-circuit design verification for multiple devices

The host system on which the XACT system operates is an IBM PC/AT or compatible system with DOS 3.0 or higher. The system requires 640K bytes of internal RAM, 1.5 to 5.5 Mbyte of Extended Memory, color graphics and a mouse. A complete system requires one parallel I/O port and two serial ports for the mouse and in-circuit emulation.

### Designing with the XACT Development System

Designing with the Logic Cell Array is similar to using conventional MSI elements or gate array cells. A range of supported packages, including FutureNet and Schema, provide schematic capture with elements from a macro library. The XACT development system then translates the schematic description into partitioned Logic Blocks and I/O Blocks, based on shared input variables or efficient use of flip-flop and combinatorial logic. Design entry can also be implemented directly with the XACT development system using an interactive graphic design editor. The design information includes both the functional specifications for each block and a definition of the interconnection networks. Automatic placement and routing is available for either method of design entry. After routing the interconnections, various checking stages and processing of that data are performed to insure that the design is correct. Design changes may be implemented in minutes. The design file is used to generate the programming data which can be down loaded directly into an LCA in the user's target system and operated. The program information may be used to program PROM, EPROM or ROM devices, or stored in some other media as needed by the final system.

Design verification may be accomplished by using the XILINX XACTOR In-Circuit Design Verifier directly in the target system and/or the P-SILOS logic simulator.

## PIN DESCRIPTIONS

### 1. Permanently Dedicated Pins.

#### Vcc

One or two (depending on package type) connections to the nominal +5 V supply voltage. All must be connected.

---

#### GND

One or two (depending on package type) connections to ground. All must be connected.

---

#### PWRDWN

A LOW on this CMOS compatible input stops all internal activity to minimize Vcc power and puts all output buffers in a high impedance state, but Configuration is retained. When the PWRDWN pin returns HIGH the device returns to operation with the same sequence of buffer enable and DONE/PROGRAM as at the completion of configuration. All internal storage elements are reset. If not used, PWRDWN must be tied to Vcc.

---

#### RESET

This is an active low input which has three functions.

Prior to the start of configuration, a LOW input will delay the start of the configuration process. An internal circuit senses the application of power and begins a minimal time-out cycle. When the time-out and RESET are complete, the levels of the "M" lines are sampled and configuration begins.

If RESET is asserted during a configuration, the LCA is re-initialized and will restart the configuration at the termination of RESET.

If RESET is asserted after configuration is complete, it will provide an asynchronous reset of all IOB and CLB storage elements of the LCA.

RESET can also be used to recover from partial power failure. See section on Re-program under "Special Configuration Functions."

---

#### CCLK

During configuration, Configuration Clock is an output of an LCA in Master mode or Peripheral mode. LCAs in Slave mode use it as a clock input. During a Readback operation it is a clock input for the configuration data being shifted out.

---

## DONE

The DONE output is configurable as open drain with or without an internal pull-up resistor. At the completion of configuration, the circuitry of the LCA becomes active in a synchronous order, and DONE may be programmed to occur one cycle before or after that.

## PROG

Once configuration is done, a HIGH to LOW transition of this pin will cause an initialization of the LCA and start a reconfiguration.

---

## M0

As Mode 0, this input and M1, M2 are sampled before the start of configuration to establish the configuration mode to be used.

## RTRIG

As a Read Trigger, a LOW-to-HIGH input transition, after configuration is complete, will initiate a Readback of configuration and storage element data by CCLK. This operation may be limited to a single request, or be inhibited altogether, by selecting the appropriate readback option when generating the bit stream.

---

## M1

As Mode 1, this input and M0, M2 are sampled before the start of configuration to establish the configuration mode to be used. If Read back is to be used, a 5 k $\Omega$  resistor should be used to define mode level inputs.

---

## RDATA

As an active low Read Data, after configuration is complete, this pin is the output of the readback data.

---

## 2. User I/O Pins that can have special functions.

## M2

As Mode 2, this input has a passive pullup during configuration. Together with M0 and M1 it is sampled before the start of configuration to establish the configuration mode to be used. After configuration this pin becomes a user programmable I/O pin.

---

## HDC

High During Configuration is held at a HIGH level by the LCA until after configuration. It is available as a control output indicating that configuration is not yet completed. After configuration this pin is a user I/O pin.

---

## LDC

Low During Configuration is held at a LOW level by the LCA until after configuration. It is available as a control output indicating that configuration is not yet completed. It is particularly useful in Master mode as a LOW enable for an EPROM. After configuration this pin is a user I/O pin.

If used as a LOW EPROM enable, it must be programmed as a HIGH after configuration.

---

## XTL1

This user I/O pin can be used to operate as the output of an amplifier driving an external crystal and bias circuitry.

---

## XTL2

This user I/O pin can be used as the input of an amplifier connected to an external crystal and bias circuitry. The I/O Block is left unconfigured. The oscillator configuration is activated by routing a net from the oscillator buffer symbol output and by the MAKEBITS program.

---

## CS0, CS1, CS2, WRT

These four inputs represent a set of signals, three active low and one active high, which are used in the peripheral mode to control configuration data entry. The assertion of all four generates a LOW CCLK. In master mode, these pins become part of the parallel configuration byte (D4,D3,D2,D1). After configuration is complete, they are user-programmed I/O.

---

## RCLK

During Master parallel mode configuration  $\overline{RCLK}$  represents a "read" of an external dynamic memory device (normally not used).

---

## D0-D7

This set of 8 pins represent the parallel configuration byte for the parallel Master and Peripheral modes. After configuration is complete they are user programmed I/O pin.

---

## A0-A15

This set of 16 pins present an address output for a configuration EPROM during Master parallel mode. After configuration is complete they are user programmed I/O pin.

---

## DIN

This user I/O pin is used as serial Data input during Slave or Master Serial configuration. This pin is Data 0 input in Master or Peripheral configuration mode.

---

## DOUT

This user I/O pin is used during configuration to output serial configuration data for daisy-chained slaves' Data In.

---

## 3. Unrestricted User I/O Pins.

## I/O

A pin which may be programmed by the user to be Input and/or Output pin following configuration. Some of these pins present a high impedance pull-up (see next page) or perform other functions before configuration is complete (see above).

CONFIGURATION MODE: <M2.M1.M0>					48	68	68	USER						
MASTER-SER <0:0:0>	SLAVE <1:1:1>	PERIPHERAL <1:0:1>	MASTER-HIGH <1:1:0>	MASTER-LOW <1:0:0>	DIP	PLCC	PGA	OPERATION						
GND						1	B6	GND						
<<HIGH>>					A13 (O)	2	A6	IO						
					A6 (O)	1	3		B5					
					A12 (O)		4		A5					
					A7 (O)		2		5	B4				
					A11 (O)		3		6	A4				
					A8 (O)		4		7	B3				
					A10 (O)		5		8	A3				
					A9 (O)		6		9	B2				
					PWRDWN (I)					7	10	B2	PWR DWN	
<<HIGH>>					8	11	B1	IO						
					9	12	C2							
					9	13	C1							
					14	14	D2							
					10	15	D1							
					16	16	E2							
					11	17	E1							
VCC					12	18	F2	VCC						
<<HIGH>>					13	19	F1	VCC						
					14	20	G2							
					14	21	G1							
					15	22	H2							
					15	23	H1							
					16	24	J2							
					M1 (LOW)	M1 (HIGH)	M1 (LOW)		M1 (HIGH)	M1 (LOW)	17	25	J1	RDATA (O)
					M0 (LOW)	M0 (HIGH)	M0 (HIGH)		M0 (LOW)	M0 (LOW)	18	26	K1	RTRIG (I)
M2 (LOW)	M2 (HIGH)				19	27	K2	IO						
HDC (HIGH)					20	28	L2							
<<HIGH>>					20	29	K3							
LDC (LOW)					21	30	L3							
<<HIGH>>					22	31	K4	IO						
					22	32	L4							
					23	33	K5							
					23	34	L5							
GND					24	35	K6	GND						
<<HIGH>>					25	36	L6	IO						
					25	37	K7							
					26	38	L7							
					26	39	K8							
					27	40	L8							
					D7 (I)		28		41	K9				
					D6 (I)		29		42	L9				
					30	43	L10		XTL2 OR IO					
RESET (I)		31	44	K10	RESET									
DONE (O)		32	45	K11	PROG (I)									
<<HIGH>>					33	46	J10	XTL1 OR IO						
					33	47	J11							
<<HIGH>>					34	48	H10	IO						
					34	49	H11							
					CS0 (I)	D4 (I)	35		50	G10				
					CS1 (I)	D3 (I)	36		51	G11				
VCC					52	F10	VCC							
<<HIGH>>					53	F11	IO							
					CS2 (I)	D2 (I)		37	54	E10				
					55	55		E11						
RCLK		WRT (I)	D1 (I)	38	56	D10	IO							
DIN (I)		RCLK	D0 (I)	39	57	D11								
DOUT (O)					40	58	C10							
CCLK (O)	CCLK (I)	CCLK (O)			41	59	C11	CCLK (I)						
<<HIGH>>					42	60	B11	IO						
					A0 (O)		43		61	B10				
					A1 (O)		44		62	A10				
					A2 (O)		45		63	B9				
					A3 (O)		46		64	A9				
					A15 (O)		46		65	B8				
					A4 (O)		47		66	A8				
					A14 (O)		47		67	B7				
A5 (O)		48	68	A7										

<<HIGH>> IS HIGH IMPEDANCE WITH A 20-50 KΩ INTERNAL PULL-UP DURING CONFIGURATION

Table 2a. XC2064 Pin Assignments

A PLCC in a "PGA-Footprint" socket has a different signal pinout than a PGA device.

CONFIGURATION MODE: <M2:M1:M0>					68	84	84	USER OPERATION		
MASTER-SER <0:0:0>	SLAVE <1:1:1>	PERIPHERAL <1:0:1>	MASTER-HIGH <1:0:1>	MASTER-LOW <1:0:0>	PLCC	PLCC	PGA			
GND					1	1	C6	GND		
<<HIGH>>					A13 (O)	2	2	A6	IO	
					3	3	A5			
					4	4	B5			
					A6 (O)	3	5	C5		
					A12 (O)	4	6	A4		
					A7 (O)	5	7	B4		
					A11 (O)	6	8	A3		
					A8 (O)	7	9	A2		
					A10 (O)	8	10	B3		
A9 (O)	9	11	A1							
PWRDWN (I)					10	12	B2	PWR DWN		
<<HIGH>>					11	13	C2	IO		
					12	14	B1			
					13	15	C1			
					14	16	D2			
					15	17	D1			
					18	18	E3			
					16	19	E2			
					20	20	E1			
VCC					17	21	F2	VCC		
<<HIGH>>					18	22	F3			
					19	23	G3			
					24	24	G1			
					20	25	G2			
					26	26	F1			
					21	27	H1			
					22	28	H2			
					23	29	J1			
					24	30	K1			
					M1 (LOW)	M1 (HIGH)	M1 (LOW)	M1 (HIGH)	M1 (LOW)	25
M0 (LOW)	M0 (HIGH)	M0 (HIGH)	M0 (LOW)	M0 (LOW)	26	32	L1	RTRIG (I)		
M2 (LOW) M2 (HIGH)					27	33	K2	IO		
<<HIGH>>					28	34	K3			
HDC (HIGH)					29	35	L2			
<<HIGH>>					30	36	L3			
LDC (LOW)					31	37	K4			
<<HIGH>>					32	38	L4			
					39	39	J5			
					33	40	K5			
					34	41	L5			
					42	42	K6			
					35	43	J6			
GND					44	44	J7		GND	
<<HIGH>>					36	45	L7	IO		
					37	46	K7			
					38	47	L8			
					48	48	L8			
					39	49	K8			
					40	50	L9			
					41	51	L10			
					42	52	K9			
D7 (I)					43	53	L11	XTL2 OR IO		
D6 (I)					44	54	K10	RESET		
RESET (I)					45	55	J10	PROG (I)		
DONE (O)					46	56	K11	XTL1 OR IO		
<<HIGH>>					47	57	J11	IO		
<<HIGH>>					48	58	H10			
					D5 (I)	49	59		H11	
					59	60	F10			
					61	61	G10			
					CS0 (I)	D4 (I)	50		62	G11
					CS1 (I)	D3 (I)	51	63	G9	
VCC					52	64	F9	VCC		
<<HIGH>>					53	65	F11	IO		
					CS2 (I)	D2 (I)	54		66	E11
					55	67	E10			
					56	68	E9			
					69	69	D11			
WR (I)	D1 (I)	56	70	D10						
RCLR	DIN (I)	RCLR	D0 (I)	57	71	C11				
BOUT (O)					58	72	B11			
CCLK (O)	CCLK (I)	CCLK (O)	CCLK (O)	59	73	C10	CCLK (I)			
<<HIGH>>					60	74	A11	IO		
					A0 (O)	61	75		B10	
					A1 (O)	62	76		B9	
					A2 (O)	63	77		A10	
					A3 (O)	64	78		A9	
					A15 (O)	65	79		B8	
					A4 (O)	66	80		A8	
					A14 (O)	67	81		B6	
					82	82	B7			
					83	83	A7			
A5 (O)	68	84	C7							

<<HIGH>> IS HIGH IMPEDANCE WITH A 20-50 KΩ INTERNAL PULL-UP DURING CONFIGURATION

Table 2b. XC2018 Pin Assignments

## PARAMETRICS

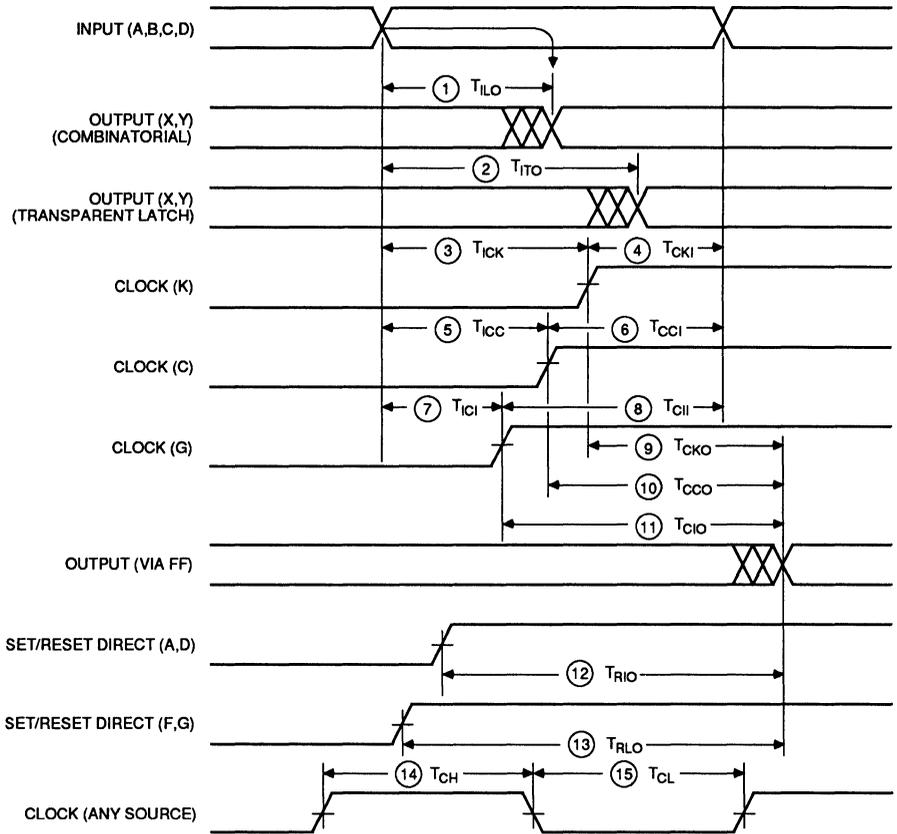
Absolute Maximum Ratings			Units
V <sub>CC</sub>	Supply voltage relative to GND	-0.5 to 7.0	V
V <sub>IN</sub>	Input voltage with respect to GND	-0.5 to V <sub>CC</sub> + 0.5	V
V <sub>TS</sub>	Voltage applied to three-state output	-0.5 to V <sub>CC</sub> + 0.5	V
T <sub>STG</sub>	Storage temperature (ambient)	-65 to +150	°C
T <sub>SOL</sub>	Maximum soldering temperature (10 sec @ 1/16 in.)	+260	°C

\*Note: Stresses beyond those listed under Absolute Maximum Ratings may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those listed under Recommended Operating Conditions is not implied. Exposure to Absolute Maximum Ratings conditions for extended periods of time may affect device reliability.

Recommended Operating Conditions				Min	Max	Units
V <sub>CC</sub>	Supply voltage relative to GND	Commercial	0°C to 70°C	4.75	5.25	V
	Supply voltage relative to GND	Industrial	-40°C to 85°C	4.5	5.5	V
	Supply voltage relative to GND	Military	-55°C to 125°C	4.5	5.5	V
V <sub>IHT</sub>	High-level input voltage — TTL configuration			2.0	V <sub>CC</sub>	V
V <sub>ILT</sub>	Low-level input voltage — TTL configuration			0	0.8	V
V <sub>IHC</sub>	High-level input voltage — CMOS configuration			.7 V <sub>CC</sub>	V <sub>CC</sub>	V
V <sub>ILC</sub>	Low-level input voltage — CMOS configuration			0	.2 V <sub>CC</sub>	V

Electrical Characteristics Over Operating Conditions			Min	Max	Units
V <sub>OH</sub>	High-level output voltage (@ I <sub>OH</sub> = -4.0 ma V <sub>CC</sub> min)	Commercial	3.86		V
V <sub>OL</sub>	Low-level output voltage (@ I <sub>OL</sub> = 4.0 ma V <sub>CC</sub> min)			0.32	V
V <sub>OH</sub>	High-level output voltage (@ I <sub>OH</sub> = -4.0 ma V <sub>CC</sub> )	Industrial	3.76		V
V <sub>OL</sub>	Low-level output voltage (@ I <sub>OL</sub> = 4.0 ma V <sub>CC</sub> )			0.37	V
V <sub>OH</sub>	High-level output voltage (@ I <sub>OH</sub> = -4.0 ma V <sub>CC</sub> )	Military	3.7		V
V <sub>OL</sub>	Low-level output voltage (@ I <sub>OL</sub> = 4.0 ma V <sub>CC</sub> )			0.4	V
I <sub>CCO</sub>	Quiescent operating power supply current				
	CMOS thresholds (@ V <sub>CC</sub> Max)			5	mA
	TTL thresholds (@ V <sub>CC</sub> Max)			10	mA
I <sub>CCPD</sub>	Power-down supply current (@ V <sub>CC</sub> = 5.0 V)			0.5	mA
I <sub>IL</sub>	Leakage Current Commercial/Industrial Temperature		-10	+10	μA
	Leakage Current Military -55°C to 125°C		-100	+100	μA
C <sub>IN</sub>	Input capacitance (sample tested)			10	pF

CLB SWITCHING CHARACTERISTICS

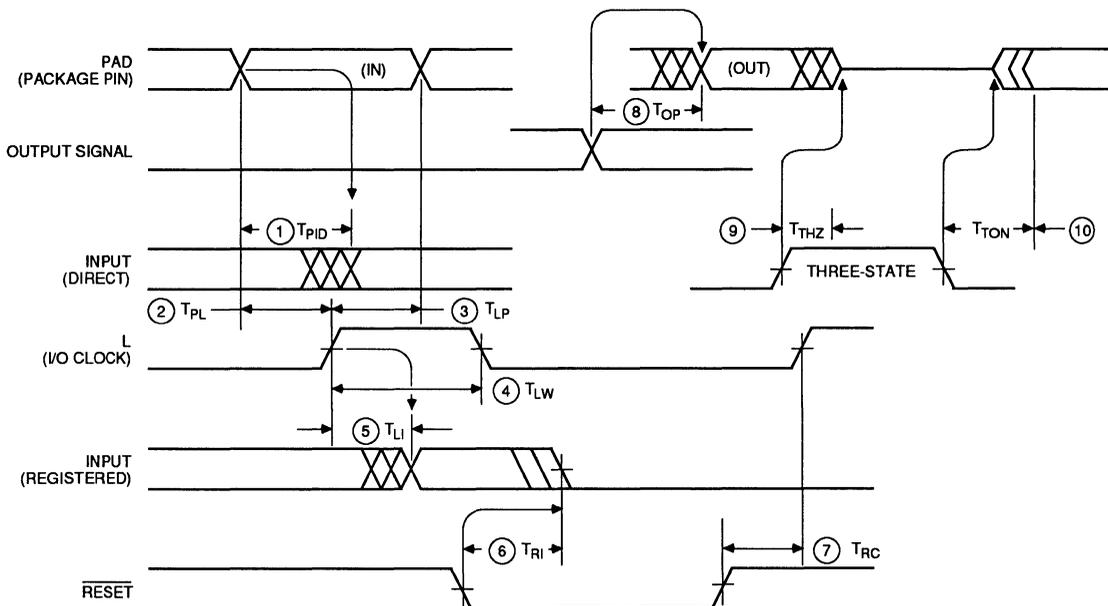


## CLB SWITCHING CHARACTERISTICS (Continued)

		Speed Grade		-33		-50		-70		Units
	Description	Symbol		Min	Max	Min	Max	Min	Max	
Logic input to Output	Combinatorial	1	T <sub>ILO</sub>		20		15		10	MHz
	Transparent latch	2	T <sub>Iro</sub>		25		20		14	
	Additional for Q through F or G to out		T <sub>QLO</sub>		13		8		6	
K Clock	To output	9	T <sub>CkO</sub>		20		15		10	
	Logic-input setup	3	T <sub>ICK</sub>	12		8		7		
	Logic-input hold	4	T <sub>CKI</sub>	1		1		1		
C Clock	To output	10	T <sub>CCO</sub>		25		19		13	
	Logic-input setup	5	T <sub>ICC</sub>	12		9		6		
	Logic-input hold	6	T <sub>CCI</sub>	6		1		1		
Logic input to G Clock	To output	11	T <sub>CIO</sub>		37		27		20	
	Logic-input setup	7	T <sub>CI</sub>	6		4		3		
	Logic-input hold	8	T <sub>CI</sub>	9		5		4		
Set/Reset direct	Input A or D to out	12	T <sub>RIO</sub>		25		22		16	
	Through F or G to out	13	T <sub>RLO</sub>		37		28		21	
	Master Reset pin to out		T <sub>MRO</sub>		35		25		20	
	Separation of set/reset		T <sub>RS</sub>	17		9		7		
	Set/Reset pulse-width		T <sub>RPW</sub>	12		9		7		
Flip-flop Toggle rate	Q through F to flip-flop		F <sub>CLK</sub>	33		50		70		
Clock	Clock high	14	T <sub>CH</sub>	12		8		7		
	Clock low	15	T <sub>CL</sub>	12		8		7		

- Notes:
1. All switching characteristics apply to all valid combinations of process, temperature and supply with a nominal chip power dissipation of 250 mW.
  2. Units are ns unless otherwise specified.

## IOB SWITCHING CHARACTERISTICS

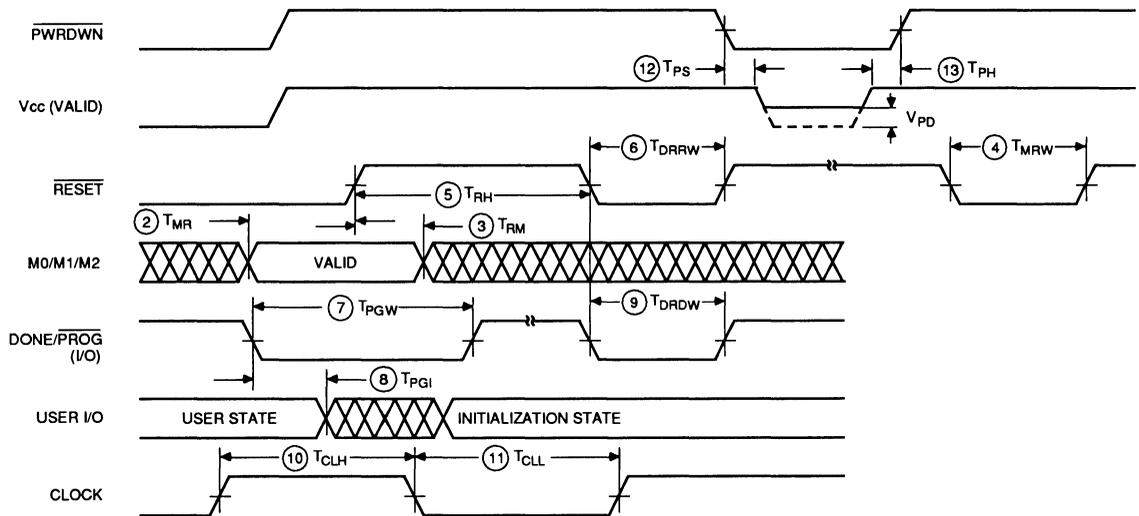


1104 31

	Description	Speed Grade		-33		-50		-70		Units ns
		Symbol	Min	Max	Min	Max	Min	Max		
Pad (package pin)	To input (direct)	1	T <sub>PI</sub> D		12		8		6	
I/O Clock	To input (storage)	5	T <sub>L</sub> I		20		15		11	
	To pad-input setup	2	T <sub>P</sub> L	12		8		6		
	To pad-input hold	3	T <sub>L</sub> P	0		0		0		
	Pulse width Frequency	4	T <sub>L</sub> W	12	33	9	50	7	70	
Output	To pad (output enabled)	8	T <sub>O</sub> P		15		12		9	
Three-state	To pad begin hi-Z	9	T <sub>T</sub> HZ		25		20		15	
	To pad end hi-Z	10	T <sub>T</sub> ON		25		20		15	
RESET	To input (storage)	6	T <sub>R</sub> I		40		30		25	
	To input clock	7	T <sub>R</sub> C		35		25		20	

Note: Timing is measured at 0.5 V<sub>CC</sub> levels with 50pF output load.

## GENERAL LCA SWITCHING CHARACTERISTICS

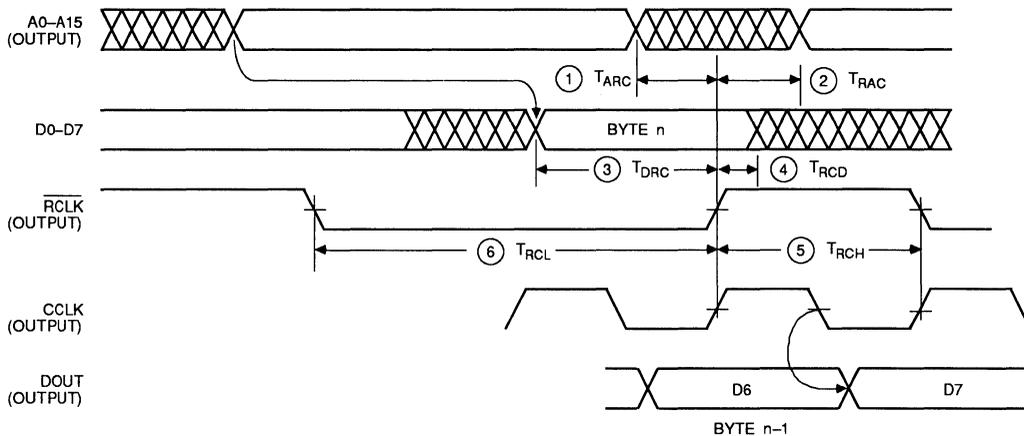


1104 32

		Speed Grade		-33		-50		-70		Units
	Description	Symbol	Min	Max	Min	Max	Min	Max	ns	
$\overline{\text{RESET}}$ (2)	M2, M1, M0 setup	2	TMR	60		60		60		
	M2, M1, M0 hold	3	TRM	60		60		60		
	Width—FF Reset	4	TMRW	150		150		150		
	High before $\overline{\text{RESET}}$ (5)	5	TRH	6		6		6		$\mu\text{s}$
	Device Reset (5)	6	TDRRW	6		6		6		$\mu\text{s}$
$\overline{\text{DONE/PROG}}$	Progam width (low)	7	TPGW	6		6		6		$\mu\text{s}$
	Initialization	8	TPGI		7		7		7	$\mu\text{s}$
	Device Reset (5)	9	TDRDW	6		6		6		$\mu\text{s}$
CLOCK	Clock (high)	10	TCLH	12		8		7		
	Clock (low)	11	TCLL	12		8		7		
$\overline{\text{PWR DWN}}$ (4)	Setup to Vcc	12	T <sub>PS</sub>	0		0		0		
	Hold from Vcc	13	T <sub>PH</sub>	0		0		0		
	Power Down		V <sub>PD</sub>	2.0		2.0		2.0		V

- Notes:
1. VCC must rise from 2.0 Volts to VCC minimum in less than 10 ms for master mode.
  2.  $\overline{\text{RESET}}$  timing relative to power-on and valid mode lines (M0, M1, M2) is relevant only when  $\overline{\text{RESET}}$  is used to delay configuration.
  3. Minimum CLOCK widths for the auxillary buffer are 1.25 times the TCLH, TCLL.
  4.  $\overline{\text{PWR DWN}}$  must be inactive until after initialization state is complete.
  5. After  $\overline{\text{RESET}}$  is high,  $\overline{\text{RESET}} = \text{D/P} = \text{LOW}$  for 6 $\mu\text{s}$  will abort to CLEAR.

**MASTER MODE PROGRAMMING SWITCHING CHARACTERISTICS**

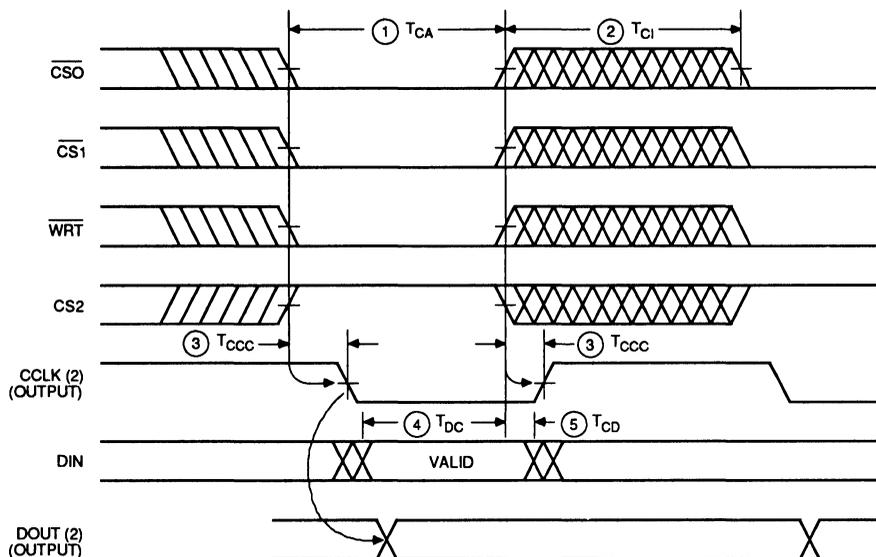


1104 33

		Speed Grade		-33		-50		-70		Units
	Description	Symbol		Min	Max	Min	Max	Min	Max	ns
RCLK	From address invalid	1	TARC		0		0		0	μs
	To address valid	2	TRAC		200		200		200	
	To data setup	3	TDRC	60		60		60		
	To data hold	4	TRCD	0		0		0		
	$\overline{\text{RCLK}}$ high	5	TRCH	600		600		600		
	$\overline{\text{RCLK}}$ low	6	TRCL	4.0		4.0		4.0		

Notes: 1. CCLK and DOUT timing are the same as for slave mode.  
 2. At power-up, Vcc must rise from 2.0 V to Vcc min. in less than 10 ms.

## PERIPHERAL MODE PROGRAMMING SWITCHING CHARACTERISTICS

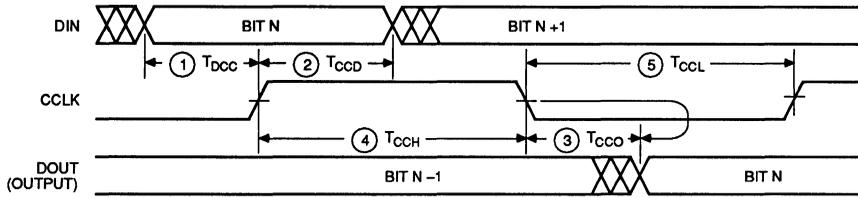


1104 34

		Speed Grade		-33		-50		-70		Units
	Description	Symbol		Min	Max	Min	Max	Min	Max	ns
Controls (1) ( $\overline{CS0}$ , $\overline{CS1}$ , $\overline{CS2}$ , $\overline{WRT}$ )	Active (last active input to first inactive)	1	$T_{CA}$	0.25	5.0	0.25	5.0	0.25	5.0	$\mu s$
	Inactive (first inactive input to last active)	2	$T_{CI}$	0.25		0.25		0.25		$\mu s$
CCLK(2) (OUTPUT) DIN setup DIN hold	CCLK(2)	3	$T_{CCC}$		75		75		75	
	DIN setup	4	$T_{DC}$	50		50		50		
	DIN hold	5	$T_{CD}$	0		0		0		

- Notes:
1. Peripheral mode timing determined from last control signal of the logical AND of ( $\overline{CS0}$ ,  $\overline{CS1}$ ,  $\overline{CS2}$ ,  $\overline{WRT}$ ) to transition to active or inactive state.
  2. CCLK and DOUT timing are the same as for slave mode.
  3. Configuration must be delayed at least 40 ms after Vcc min.

### SLAVE MODE PROGRAMMING SWITCHING CHARACTERISTICS

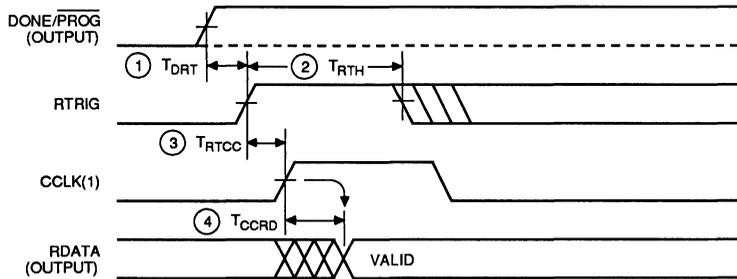


1104 35

		Speed Grade		-33		-50		-70		Units
	Description	Symbol		Min	Max	Min	Max	Min	Max	ns
CCLK	To DOUT	3	TcCO		65		65		65	
	DIN setup	1	TdCC	10		10		10		
	DIN hold	2	TcCD	40		40		40		
	High time	4	TcCH	0.25		0.25		0.25		μs
	Low time	5	TcCL	0.25	5.0	0.25	5.0	0.25	5.0	μs
	Frequency		FCC		2		2		2	MHz

Note: Configuration must be delayed at least 40 ms after Vcc min.

### PROGRAM READBACK SWITCHING CHARACTERISTICS



1104 36

		Speed Grade		-33		-50		-70		Units
	Description	Symbol		Min	Max	Min	Max	Min	Max	ns
RTRIG	PROG setup	1	TDRT	300		300		300		
	RTRIG high	2	TRTH	250		250		250		
CCLK	RTRIG setup	3	TRTCC	100		100		100		
	RDATA delay	4	TCCRd		100		100		100	

Notes: 1. CCLK and DOUT timing are the same as for slave mode.  
 2. DONE/PROG output/input must be HIGH (device programmed) prior to a positive transition of RTRIG (M0).

		48 PIN		68 PIN		84 PIN	
		PLASTIC DIP	CERAMIC DIP	PLASTIC PLCC	CERAMIC PGA	PLASTIC PLCC	CERAMIC PGA
		-PD 48	-CD 48	-PC 68	-PG 68	-PC 84	-PG 84
XC-2064	-33	C	I M	C I	C I M B		
	-50	C	I	C I	C I M B		
	-70			C			
XC-2018	-33			C I		C I	C I M B
	-50			C I		C I	C I M B
	-70			C		C	C

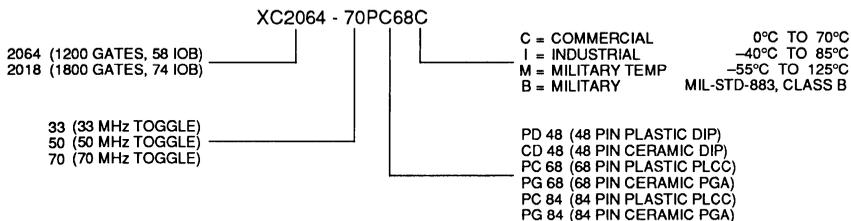
C = COMMERCIAL                      0°C TO 70°C  
 I = INDUSTRIAL                      -40°C TO 85°C  
 M = MILITARY                        -55°C TO 125°C  
 B = MILITARY                        MIL-STD-883, CLASS B

1104 37

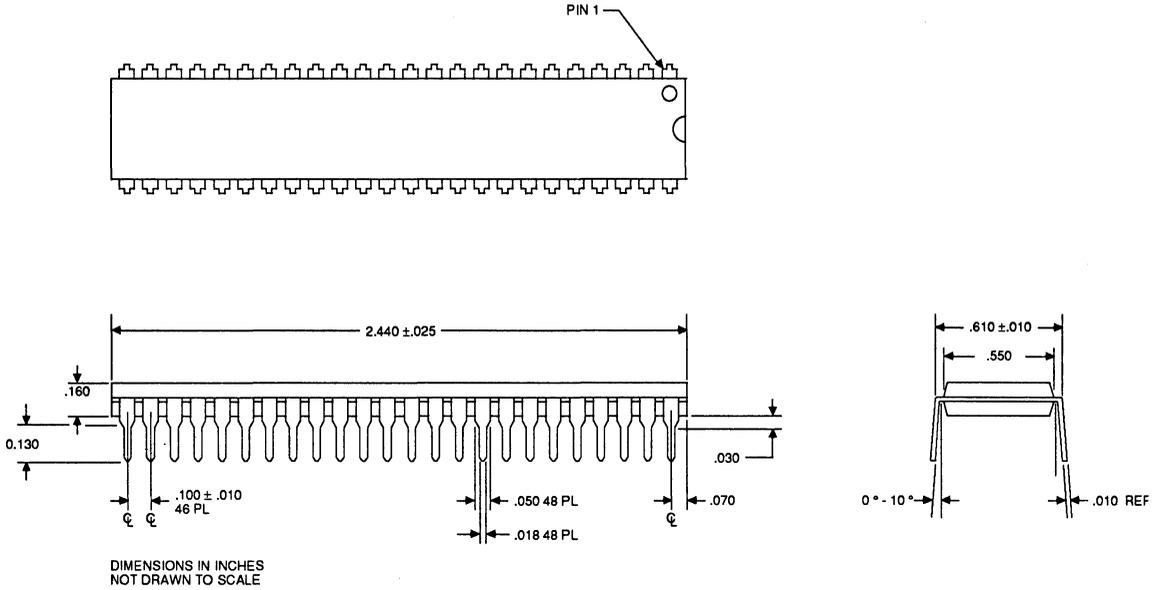
**Table 5. LCA Package and Temperature Options**

### Ordering Information

Further information is available from Xilinx franchised distributors or from the nearest Xilinx sales representative. Part numbers are composed as follows:

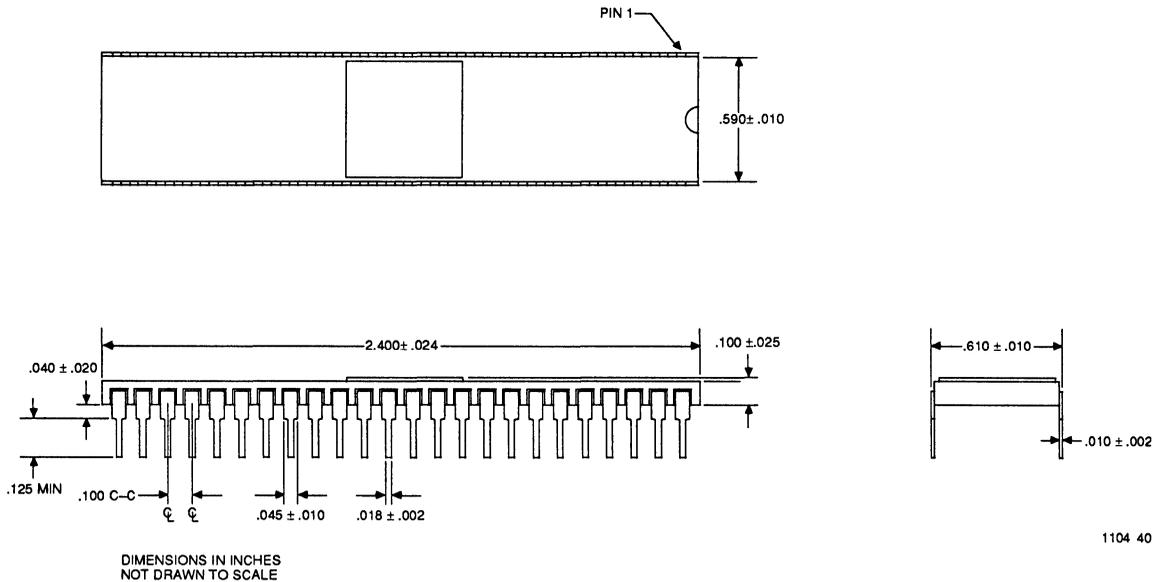


1104 38



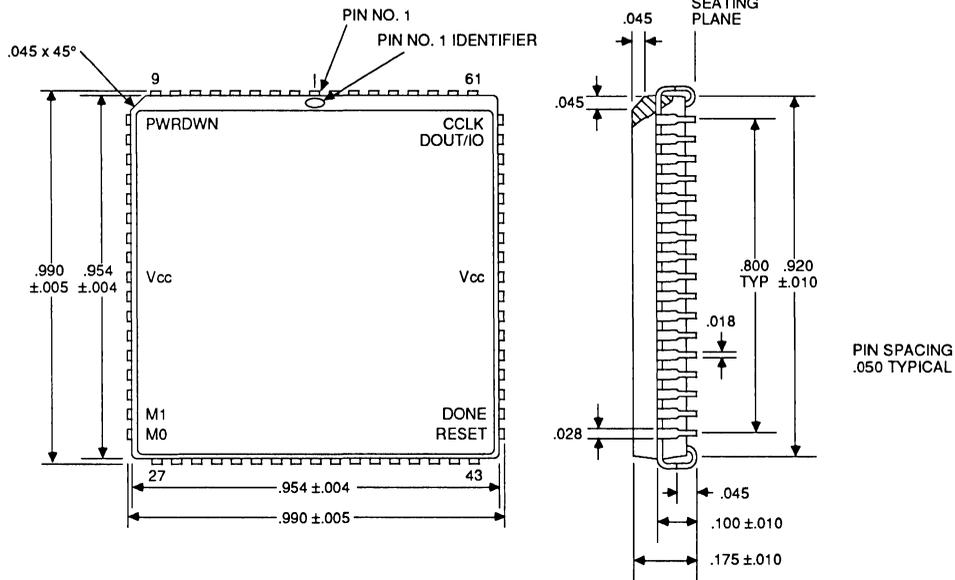
1104 39

**48-Pin Plastic DIP Package**



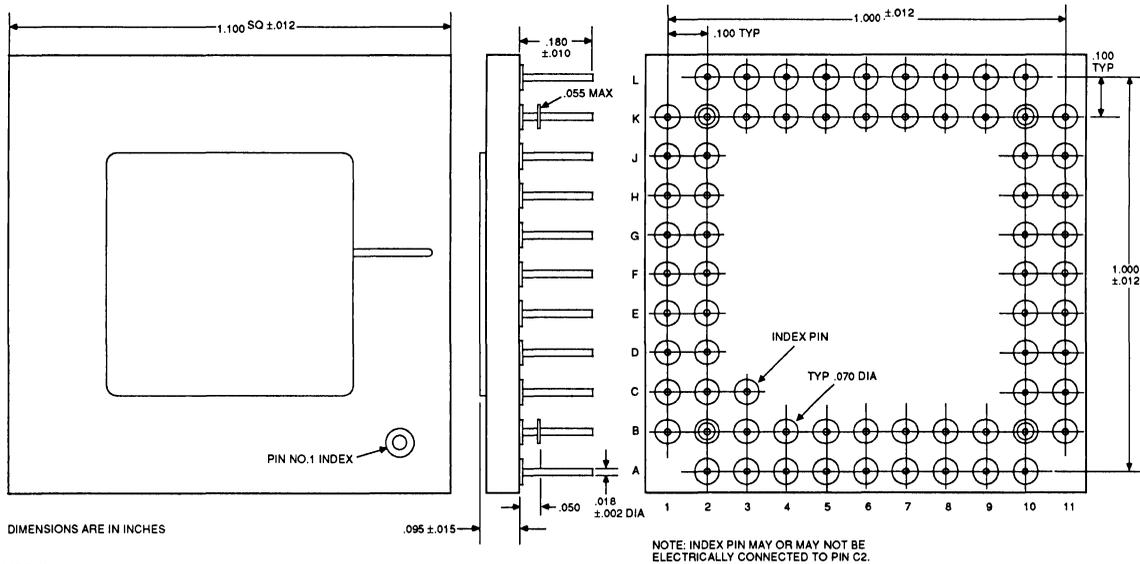
1104 40

**48-Pin Ceramic DIP Package**



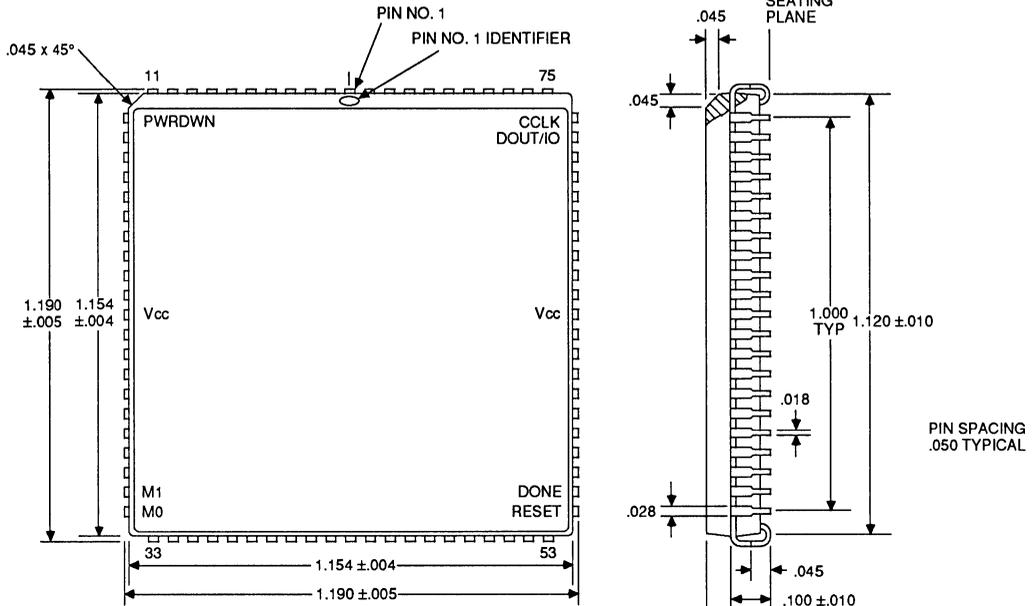
1104 41

**68-Pin PLCC Package**



1104 42

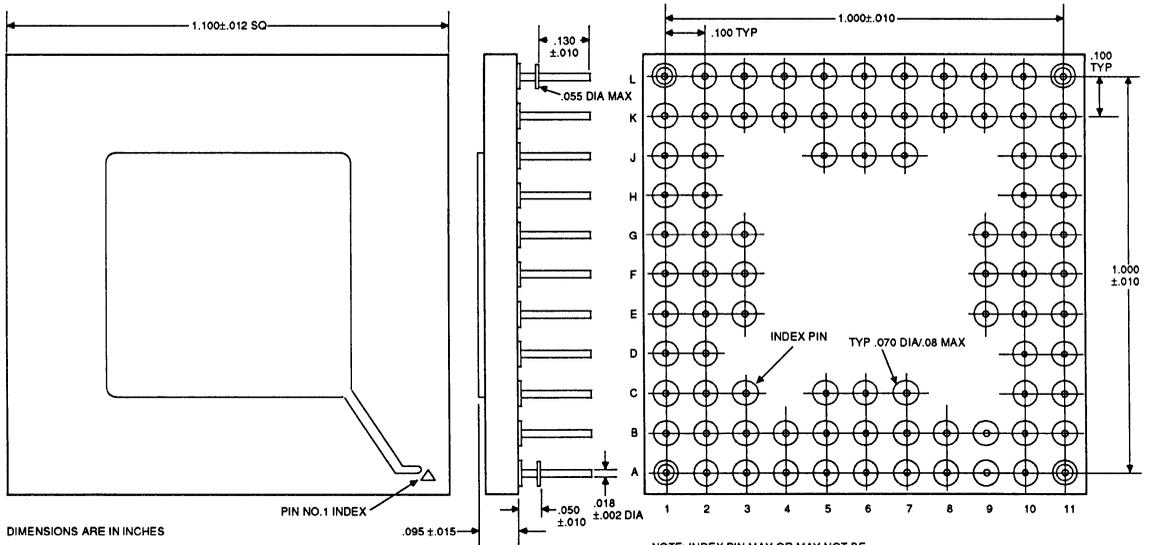
**68-Pin PGA Package**



DIMENSIONS ARE IN INCHES  
 $\Theta_{JA} = 30-35 \text{ }^\circ\text{C/W}$

1104 43

84-Pin PLCC Package



DIMENSIONS ARE IN INCHES

NOTE: INDEX PIN MAY OR MAY NOT BE ELECTRICALLY CONNECTED TO PIN C2.

1104 44

84-Pin PGA Package

### SOCKET INFORMATION

The following sockets, with matching hole patterns, are available for PLCC devices.

Description	Vendor	Part Number
<b>68 PIN</b>		
PCB solder tail, tin plate	AMP	821574-1
Surface mount, tin plate	AMP	821542-1
PCB solder tail, tin plate	Burndy*	QILE68P-410T
PCB solder tail, tin plate	Honda Tsuhin Kogyo*	TC-A68D or TC-A68DB
Surface mount, tin plate	Honda Tsuhin Kogyo	TC-A68S
PCB solder tail, tin plate	Midland-Ross*	709-2000-068-4-1-1
PCB solder tail, tin plate	Methode*	213-068-001
Surface mount, tin plate	Methode	213-068-002
<b>84 PIN</b>		
PCB solder tail, tin plate	AMP	821573-1
Surface mount, tin plate	AMP	821546-1
PCB solder tail, tin plate	Honda Tsuhin Kogyo*	TC-A84D or TC-A84DB
Surface mount, tin plate	Honda Tsuhin Kogyo	TC-A84S
PCB solder tail, tin plate	Midland-Ross*	709-2000-084-4-1-1
PCB solder tail, tin plate	Methode*	213-084-001
Surface mount, tin plate	Methode	213-084-002

\* Sockets will plug into pin-grid array (PGA) wire-wrap sockets for breadboard use.





# XC2064B XC2018B Military Logic Cell™ Arrays

Product Specification. See Note 1.

## FEATURES

- MIL-STD-883 Class B Processing. Complies with paragraph 1.2.1.b.
- User-programmable gate arrays
- Low power CMOS static memory technology
- Standard product. Completely tested at factory
- Design changes made in minutes
- Complete user control for design cycle. Secure design process
- Complete PC or workstation based development system
  - Schematic entry
  - Auto Place/ Route (DS23)
  - XACT Design Editor (DS21)
  - Logic & Timing Simulator (DS22)
  - XACTOR In-circuit Verifier (DS24)

## DESCRIPTION

The Logic Cell™ Array (LCA) is a high density CMOS programmable gate array. Its patented array architecture consists of three type of configurable elements: Input/Output Blocks, Configurable Logic Blocks and Interconnect. The designer can define individual I/O blocks for interface to external circuitry, define logic blocks to implement logic functions and define interconnection networks to compose larger scale logic functions.

The Logic Cell Array's logic functions and interconnections are determined by the configuration program stored

Part Number	Logic Capacity (usable gates)	Configurable Logic Blocks	User I/O's	Configuration Program (bits)
XC2064	1200	64	58	12038
XC2018	1800	100	74	17878

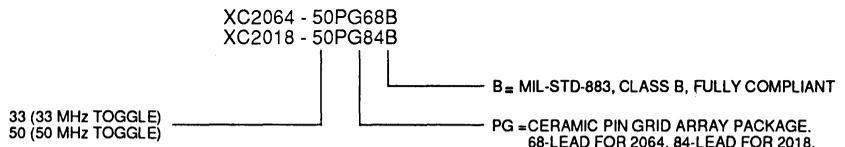
in internal static memory cells. On-chip logic provides for automatic loading of configuration data at power-up or on command. The program data can reside in an EEPROM, EPROM or ROM on the circuit board or on a floppy disk or hard disk.

Several methods of automatically loading the required data are designed into the Logic Cell Array and are determined by logic levels applied to mode selection pins at configuration time. The form of the data may be either serial or parallel, depending on the configuration mode. The programming data are independent of the configuration mode selected.

The XACT development system allows the user to define the logic functions of the device. Schematic capture is available for design entry, while logic and timing simulation, and in-circuit debugging are available for design verification. XACT is used to compile the data pattern which represents the configuration program. This data can then be converted to a PROM programmer format file to create the configuration program storage.

See the XC2064/XC2018 Commercial datasheet for a full description.

## ORDERING INFORMATION



1107 01

TSC0026 Rev.05

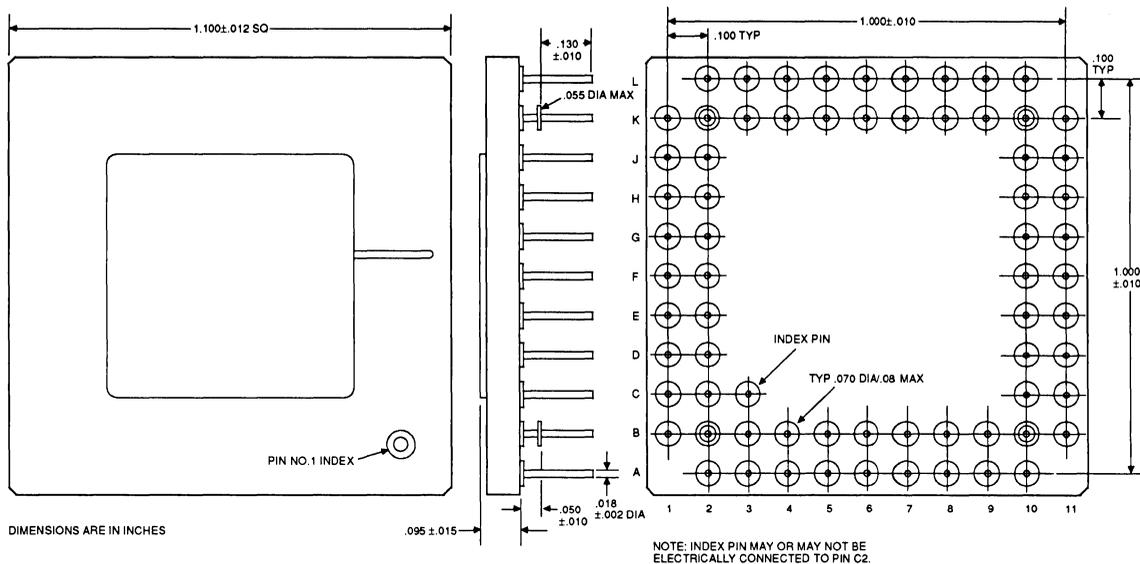
PIN ASSIGNMENTS

CONFIGURATION MODE: <M2:M1:M0>					68	84	USER OPERATION
MASTER-SER <0:0:0>	SLAVE <1:1:1>	PERIPHERAL <1:0:1>	MASTER-HIGH <1:1:0>	MASTER-LOW <1:0:0>	PGA	PGA	
GND					B6	C6	GND
<<HIGH>>					A13 (O)	A6	A6
					A6 (O)	B5	C5
					A12 (O)	A5	A4
					A7 (O)	B4	B4
					A11 (O)	A4	A3
					A8 (O)	B3	A2
					A10 (O)	A3	B3
PWRDWN (I)					A2	A1	
<<HIGH>>					B2	B2	PWR DWN
					B1	C2	
					C2	B1	
					C1	C1	
					D2	D2	
					D1	D1	
					E3	E3	
VCC					E2	E2	
<<HIGH>>					E1	F2	VCC
					F2	F3	
					F1	G3	
					G1	G1	
					G2	G2	
					F1	F1	
					G1	H1	
H2	H2						
H1	J1						
M1 (LOW)	M1 (HIGH)	M1 (LOW)	M1 (HIGH)	M1 (LOW)	J1	J2	RDATA (O)
M0 (LOW)	M0 (HIGH)	M0 (HIGH)	M0 (LOW)	M0 (LOW)	K1	L1	RTRIG (I)
M2 (LOW)	M2 (HIGH)				K2	K2	
HDC (HIGH)					L2	K3	
<<HIGH>>					K3	L2	
LDC (LOW)					L3	L3	
<<HIGH>>					K4	K4	
					L4	L4	
					J5	J5	
					K5	K5	
					L5	L5	
GND					K6	J6	GND
<<HIGH>>					J7		
					L6	L7	
					K7	K7	
					L7	L6	
					L8	L8	
					K8	K8	
					L8	L9	
RESET (I)					K9	L10	XTL2 OR I/O
DONE (O)					L9	K9	RESET
<<HIGH>>					J10	K11	PROG (I)
					J11	J11	XTL1 OR I/O
<<HIGH>>					H10	H10	
					H11	H11	
					F10	F10	
					G10	G10	
					G11	G11	
VCC					G11	G9	
<<HIGH>>					F10	F9	VCC
					F11	F11	
					E10	E11	
					E11	E9	
CS0 (I)					D10	D10	
CS1 (I)					D11	C11	
CS2 (I)					C10	B11	
WRT (I)					C11	C10	
RCLK	DIN (I)			RCLK	B11	A11	CCLK (I)
DOUT (O)					A0 (O)	B10	
CCLK (O)					A1 (O)	B9	
<<HIGH>>					A2 (O)	B9	A10
					A3 (O)	A9	A9
					A15 (O)	B8	B8
					A4 (O)	A8	A8
					A14 (O)	B7	B6
						B7	B7
						A7	A7
	A5 (O)	A7	C7				

<<HIGH>> IS HIGH IMPEDANCE WITH A 20-50 KΩ INTERNAL PULL-UP DURING CONFIGURATION

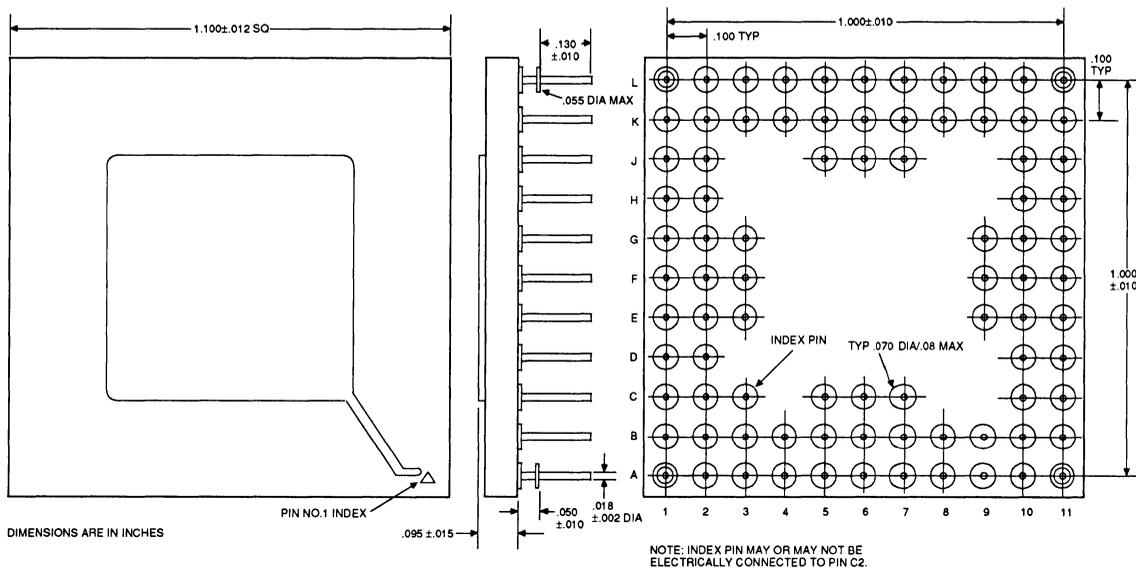
### CASE OUTLINE DRAWINGS

Conform to MIL-M-38510 Appendix C, Case P-BC.



1107 03

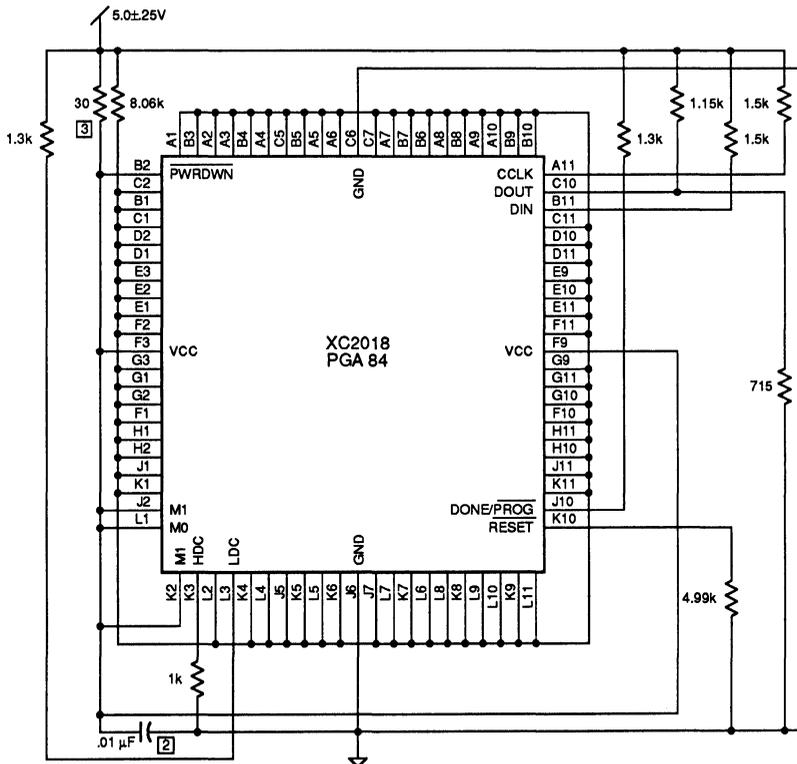
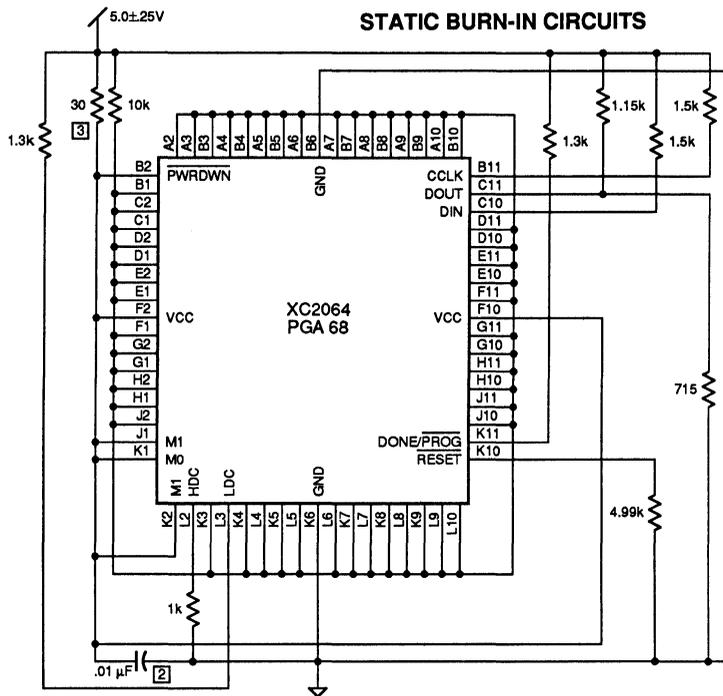
### XC2064: 68-Pin PGA Package



1107 04

### XC2018: 84-Pin PGA Package

STATIC BURN-IN CIRCUITS



NOTES:

1. UNLESS OTHERWISE SPECIFIED, ALL RESISTORS ARE METAL FILM AND ARE RATED FOR 1/8 WATT AT 150°C WITH A BUILD TOLERANCE OF 1% AND A 5% TOLERANCE OVER LIFE.
2. CAPACITOR HAS 10% TOLERANCE, 50 V RATING WITH AN X7R TEMPERATURE CHARACTERISTIC.
3. 30 Ω RESISTOR IS METAL OXIDE AND IS RATED FOR 1 W AT 150°C WITH A TOLERANCE OF 5%.

Absolute Maximum Ratings		Limits	Units
V <sub>CC</sub>	Supply voltage relative to GND	-0.5 to 7.0	V
V <sub>IN</sub>	Input voltage with respect to GND	-0.5 to V <sub>CC</sub> + 0.5	V
V <sub>TS</sub>	Voltage applied to three-state output	-0.5 to V <sub>CC</sub> + 0.5	V
T <sub>STG</sub>	Storage temperature (ambient)	-65 to +150	°C
T <sub>SOL</sub>	Maximum soldering temperature (10 sec @ 1/16 in.)	+260	°C
T <sub>J</sub>	Maximum junction temperature	+150	°C

Note: Stresses beyond those listed under Absolute Maximum Ratings may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those listed under Recommended Operating Conditions is not implied. Exposure to Absolute Maximum Ratings conditions for extended periods of time may affect device reliability.

Test	Symbol	Conditions -55°C ≤ T <sub>c</sub> ≤ +125°C V <sub>CC</sub> = 5.0 V ±10%	Group A Subgroups	Limits		Units
				Min	Max	
High Level Output Voltage	V <sub>OH</sub>	V <sub>CC</sub> = 4.5 V, I <sub>OH</sub> = -4.0 mA	1,2,3	3.7		V
Low Level Output Voltage	V <sub>OL</sub>	V <sub>CC</sub> = 5.5 V, I <sub>OL</sub> = 4.0 mA	1,2,3		0.4	V
Quiescent Operating	I <sub>CCO</sub>	V <sub>CC</sub> = 5.5 V, CMOS Inputs, V <sub>IN</sub> = V <sub>CC</sub>	1,2,3		10	mA
Power Supply Current		TTL Inputs, V <sub>IN</sub> = V <sub>CC</sub> = 5.5 V	1,2,3		15	mA
Power-Down Supply Current	I <sub>CCPD</sub>	V <sub>IN</sub> = V <sub>CC</sub> = 5.0 V, PWR DWN = 0 V	1,2,3		0.5	mA
Leakage Current	I <sub>IL</sub>	V <sub>CC</sub> = 5.5 V, V <sub>IN</sub> = V <sub>CC</sub> = 0 V	1,2,3	-10	10	μA
Input High Level TTL	V <sub>IHT</sub>	Guaranteed Input High	1,2,3	2.0		V
Input Low Level TTL	V <sub>ILT</sub>	Guaranteed Input Low	1,2,3		0.8	V
Input High Level CMOS	V <sub>IHC</sub>	Guaranteed Input High	1,2,3	.7 V <sub>CC</sub>		V
Input Low Level CMOS	V <sub>ILC</sub>	Guaranteed Input Low	1,2,3		.2 V <sub>CC</sub>	V

**Switching Characteristics, General LCA**

DONE/ <u>PROG</u> Program Width (Low) Initialization	T <sub>PGW</sub>	4	See Fig. 3	9,10,11	6	7	μs
	T <sub>PGI</sub>	5		9,10,11			μs
<u>PWR DWN</u> <sup>2</sup> Power Down Supply	V <sub>PD</sub>			1,2,3	3.5		V

**Table 1. Electrical Performance Characteristics**

Test	Sym	Conditions -55°C ≤ Tc ≤ +125°C Vcc = 5.0 V ±10%	Group A Subgroups	2064-33 Limits		2064-50 Limits		Units
				Min	Max	Min	Max	

**Switching Characteristics, Peripheral Mode Programming**

Controls ( $\overline{CS}$ , $\overline{WRT}$ ) <sup>3,4</sup> Last Active Input to First Inactive First Inactive Input to Last Active	T <sub>CA</sub>	1	See Fig. 4	9,10,11	0.5	1.0	0.5	1.0	μs
	T <sub>CI</sub>	2		9,10,11	0.5		0.5		μs
CCLK <sup>5</sup>	T <sub>CC</sub>	3		9,10,11		75		75	ns
DIN Setup	T <sub>DC</sub>	4		9,10,11	50		50		ns
DIN Hold	T <sub>CD</sub>	5		9,10,11	5		5		ns

**Switching Characteristics, Program Readback<sup>6</sup>**

RTRIG Setup	T <sub>RTH</sub>	1	See Fig. 7	9,10,11	250		250		ns
CCLK, RTRIG Setup RDATA Delay	T <sub>RTCC</sub>	2		9,10,11	100		100		ns
	T <sub>CCRD</sub>	3		9,10,11		100		100	ns

**Benchmark Patterns<sup>7</sup>**

T <sub>PID</sub> + interconnect + 8 (T <sub>ILO</sub> ) + TOP. Measured on 8 cols.	T <sub>B1</sub>			9,10,11		187		140	ns
T <sub>PID</sub> + interconnect + 8 (T <sub>ITO</sub> ) + TOP. Measured on 8 cols.	T <sub>B2</sub>			9,10,11		227		180	ns
T <sub>PID</sub> + interconnect + 8 (T <sub>OLO</sub> ) + TOP. Measured on 8 cols.	T <sub>B3</sub>			9,10,11		331		244	ns
T <sub>PID</sub> + interconnect + 8 (T <sub>QLO</sub> ) + TOP. Measured on 8 cols. Tested on all CLBs with Tick + interconnect.	T <sub>B4</sub>			9,10,11		85		62	ns
T <sub>PID</sub> + interconnect + 8 (T <sub>IC</sub> ) + TOP. Measured on 8 cols. Tested on all CLBs with Ticc + interconnect.	T <sub>B5</sub>			9,10,11		66		49	ns
T <sub>PID</sub> + interconnect + 8 (T <sub>ICI</sub> ) + TOP. Measured on 8 cols. Tested on all CLBs with Tici + interconnect.	T <sub>B6</sub>			9,10,11		90		67	ns
T <sub>PID</sub> + interconnect + 8 (T <sub>RIO</sub> ) + TOP. Measured on 8 rows.	T <sub>B7</sub>			9,10,11		251		212	ns
T <sub>PID</sub> + interconnect + T <sub>P</sub> L + T <sub>L</sub> I + TOP. Tested on all IOB's.	T <sub>B8</sub>			9,10,11		243		180	ns

Table 1. Electrical Performance Characteristics (Continued)

Test	Sym	Conditions -55°C ≤ T <sub>c</sub> ≤ +125°C V <sub>cc</sub> = 5.0 V ±10%		Group A Subgroups	2064-33 Limits		2064-50 Limits		Units
					Min	Max	Min	Max	
<b>Application Guidelines, Switching, CLB<sup>7</sup></b>									
Logic Input to Output, Combinatorial	T <sub>ILO</sub>	1	See Fig. 1	N/A		20		15	ns
Transparent Latch Additional for Q Through F or G to Out	T <sub>ILO</sub>	2		N/A		25		20	ns
	T <sub>QLO</sub>			N/A		13		8	ns
K Clock, To Output Logic-Input Setup Logic-Input Hold	T <sub>CKO</sub>	9		N/A		20		15	ns
	T <sub>ICK</sub>	3		N/A	12		8		ns
	T <sub>CKI</sub>	4		N/A	1		1		ns
C Clock, To Output Logic- Input Setup Logic-Input Hold	T <sub>CCO</sub>	10		N/A		25		19	ns
	T <sub>ICC</sub>	5		N/A	12		9		ns
	T <sub>CCI</sub>	6		N/A	6		1		ns
Logic Input to G Clock, To Output Logic-Input Setup Logic-Input Hold	T <sub>CIO</sub>	11		N/A		37		27	ns
	T <sub>ICI</sub>	7		N/A	6		4		ns
	T <sub>CI</sub>	8		N/A	9		5		ns
Set/Reset Direct, Input A or D to Out Through F or G to Out Master Reset Pin to Out Separation of Set/Reset Set/Reset Pulse-Width	T <sub>RIO</sub>	12		N/A		25		22	ns
	T <sub>RLO</sub>	13		N/A		37		28	ns
	T <sub>MRO</sub>			N/A		55		45	ns
	T <sub>RS</sub>			N/A	17		9		ns
	T <sub>RPW</sub>			N/A	12		9		ns
Flip-Flop Toggle Rate, Q Through F to Flip-Flop	F <sub>CLK</sub>			N/A	33		50		MHz
Clock High <sup>a</sup>	T <sub>CH</sub>	14		N/A	12		8		ns
Clock Low <sup>a</sup>	T <sub>CL</sub>	15		N/A	12		8		ns

Table 1. Electrical Performance Characteristics (Continued)

Test	Sym	Conditions -55°C ≤ T <sub>c</sub> ≤ +125°C V <sub>cc</sub> = 5.0 V ±10%		Group A Subgroups	2064-33 Limits		2064-50 Limits		Units
					Min	Max	Min	Max	

**Application Guidelines, Switching, IOB<sup>7</sup>**

Pad (Package Pin) to Input (Direct)	TPID	1	See Fig. 2	N/A		12		8	ns
I/O Clock									
To Input (Storage)	TLI	5		N/A		20		15	ns
To Pad-Input Setup	TPL	2		N/A	12		8		ns
To Pad Input Hold	TLP	3		N/A	0		0		ns
Pulse Width	TLW	4		N/A	12		9		ns
Frequency				N/A		33		50	MHz
Output, To Pad (Output Enable)	TOP	8		N/A		15		12	ns
Three-State, To Pad Begin hi-Z	TTHZ	9		N/A		25		20	ns
To Pad End hi-Z	TTON	10		N/A		25		20	ns
RESET, To Input (Storage)	TRI	6		N/A		40		30	ns
To Input Clock	TRC	7		N/A		35		25	ns

**Application Guidelines, Switching, Slave Mode Programming<sup>7</sup>**

CCLK, To DOUT	Tcco	3	See Fig. 6	N/A		100		100	ns
DIN Setup	TdCC	1		N/A	10		10		ns
DIN Hold	TccD	2		N/A	40		40		ns
High Time	TcCH	4		N/A	0.5		0.5		μs
Low Time	TcCL	5		N/A	0.5	1.0	0.5	1.0	μs
Frequency	Fcc			N/A		1		1	MHz

**Application Guidelines, Switching, Master Mode Programming<sup>7,9</sup>**

RCLK, From Address Invalid	TARC	1	See Fig. 5	N/A		0		0	ns
To Address Valid	TRAC	2		N/A		200		200	ns
To Data Setup	TDRC	3		N/A	60		60		ns
To Data Hold	TRCD	4		N/A	0		0		ns
RCLK High	TRCH	5		N/A	600		600		ns
RCLK Low	TRCL	6		N/A	4.0		4.0		μs

**Application Guidelines, Switching, General LCA<sup>7</sup>**

RESET <sup>10</sup>									
M2, M1, M0 Setup	TMR	1	See Fig. 3	N/A	1		1		μs
M2, M1, M0 Hold	TRM	2		N/A	1		1		μs
Width (Low)	TMRW	3		N/A	150		150		ns

Table 1. Electrical Performance Characteristics (Continued)

### XC2018B Test Specification

Absolute Maximum Ratings		Limits	Units
V <sub>CC</sub>	Supply voltage relative to GND	-0.5 to 7.0	V
V <sub>IN</sub>	Input voltage with respect to GND	-0.5 to V <sub>CC</sub> + 0.5	V
V <sub>TS</sub>	Voltage applied to three-state output	-0.5 to V <sub>CC</sub> + 0.5	V
T <sub>STG</sub>	Storage temperature (ambient)	-65 to +150	°C
T <sub>SOL</sub>	Maximum soldering temperature (10 sec @ 1/16 in.)	+260	°C
T <sub>J</sub>	Maximum junction temperature	+150	°C

Note: Stresses beyond those listed under Absolute Maximum Ratings may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those listed under Recommended Operating Conditions is not implied. Exposure to Absolute Maximum Ratings conditions for extended periods of time may affect device reliability.

Test	Symbol	Conditions -55°C ≤ T <sub>c</sub> ≤ +125°C V <sub>CC</sub> = 5.0 V ±10%	Group A Subgroups	Limits		Units
				Min	Max	
High Level Output Voltage	V <sub>OH</sub>	V <sub>CC</sub> = 4.5 V, I <sub>OH</sub> = -4.0 mA	1,2,3	3.7		V
Low Level Output Voltage	V <sub>OL</sub>	V <sub>CC</sub> = 5.5 V, I <sub>OL</sub> = 4.0 mA	1,2,3		0.4	V
Quiescent Operating	I <sub>CCO</sub>	V <sub>CC</sub> = 5.5 V, CMOS Inputs, V <sub>IN</sub> = V <sub>CC</sub>	1,2,3		10	mA
Power Supply Current	I <sub>CCPD</sub>	TTL Inputs, V <sub>IN</sub> = V <sub>CC</sub> = 5.5 V	1,2,3		15	mA
Power-Down Supply Current		V <sub>IN</sub> = V <sub>CC</sub> = 5.0 V, PWR DWN = 0 V	1,2,3		0.5	mA
Leakage Current	I <sub>IL</sub>	V <sub>CC</sub> = 5.5 V, V <sub>IN</sub> = V <sub>CC</sub> = 0 V	1,2,3	-10	10	μA
Input High Level TTL	V <sub>IHT</sub>	Guaranteed Input High	1,2,3	2.0		V
Input Low Level TTL	V <sub>ILT</sub>	Guaranteed Input Low	1,2,3		0.8	V
Input High Level CMOS	V <sub>IHC</sub>	Guaranteed Input High	1,2,3	.7 V <sub>CC</sub>		V
Input Low Level CMOS	V <sub>ILC</sub>	Guaranteed Input Low	1,2,3		.2 V <sub>CC</sub>	V

#### Switching Characteristics, General LCA

DONE/ $\overline{\text{PROG}}$ Program Width (Low) Initialization	T <sub>PGW</sub>	4	See Fig. 3	9,10,11 9,10,11	6	7	μs
	T <sub>PGI</sub>	5					
$\overline{\text{PWR DWN}}$ <sup>2</sup> Power Down Supply	V <sub>PD</sub>			1,2,3	3.5		V

Table 1. Electrical Performance Characteristics

Test	Sym	Conditions		Group A Subgroups	2018-33 Limits		2018-50 Limits		Units
		-55°C ≤ Tc ≤ +125°C	Vcc = 5.0 V ±10%		Min	Max	Min	Max	

**Switching Characteristics, Peripheral Mode Programming**

Controls ( $\overline{CS}$ , $\overline{WRT}$ ) <sup>3,4</sup> Last Active Input to First Inactive First Inactive Input to Last Active	T <sub>CA</sub>	1	See Fig. 4	9,10,11	0.5	1.0	0.5	1.0	μs
	T <sub>CI</sub>	2		9,10,11	0.5		0.5		μs
CCLK <sup>5</sup>	T <sub>CCC</sub>	3		9,10,11		75		75	ns
DIN Setup	T <sub>DC</sub>	4		9,10,11	50		50		ns
DIN Hold	T <sub>CD</sub>	5		9,10,11	5		5		ns

**Switching Characteristics, Program Readback<sup>6</sup>**

RTRIG Setup	T <sub>RTH</sub>	1	See Fig. 7	9,10,11	250		250		ns
CCLK, RTRIG Setup RDATA Delay	T <sub>RTCC</sub>	2		9,10,11	100		100		ns
	T <sub>CCRD</sub>	3		9,10,11		100		100	ns

**Benchmark Patterns<sup>7</sup>**

T <sub>PID</sub> + interconnect + 10 (T <sub>ILO</sub> ) + TOP. Measured on 10 cols.	T <sub>B1</sub>			9,10,11		238		178	ns
T <sub>PID</sub> + interconnect + 10 (T <sub>IRO</sub> ) + TOP. Measured on 10 cols.	T <sub>B2</sub>			9,10,11		288		228	ns
T <sub>PID</sub> + interconnect + 10 (T <sub>QLO</sub> ) + TOP. Measured on 10 cols.	T <sub>B3</sub>			9,10,11		410		302	ns
T <sub>PID</sub> + interconnect + 10 (T <sub>QLO</sub> ) + TOP. Measured on 10 cols. Tested on all CLBs with Tick + interconnect.	T <sub>B4</sub>			9,10,11		85		62	ns
T <sub>PID</sub> + interconnect + 10 (T <sub>QLO</sub> ) + TOP. Measured on 10 cols. Tested on all CLBs with Ticc + interconnect.	T <sub>B5</sub>			9,10,11		66		49	ns
T <sub>PID</sub> + interconnect + 10 (T <sub>QLO</sub> ) + TOP. Measured on 10 cols. Tested on all CLBs with Tici + interconnect.	T <sub>B6</sub>			9,10,11		90		67	ns
T <sub>PID</sub> + interconnect + 10 (T <sub>RIO</sub> ) + TOP. Measured on 10 rows.	T <sub>B7</sub>			9,10,11		318		269	ns
T <sub>PID</sub> + interconnect + T <sub>PL</sub> + T <sub>LI</sub> + TOP. Tested on all IOB's.	T <sub>B8</sub>			9,10,11		274		204	ns

Table 1. Electrical Performance Characteristics (Continued)

Test	Sym	Conditions -55°C ≤ T <sub>c</sub> ≤ +125°C V <sub>cc</sub> = 5.0 V ±10%		Group A Subgroups	2018-33 Limits		2018-50 Limits		Units
					Min	Max	Min	Max	
<b>Application Guidelines, Switching, CLB<sup>7</sup></b>									
Logic Input to Output, Combinatorial	T <sub>ILO</sub>	1	See Fig. 1	N/A		20		15	ns
Transparent Latch Additional for Q Through F or G to Out	T <sub>ILO</sub>	2		N/A		25		20	ns
	T <sub>QLO</sub>			N/A		13		8	ns
K Clock, To Output Logic-Input Setup Logic-Input Hold	T <sub>CKO</sub>	9		N/A		20		15	ns
	T <sub>ICK</sub>	3		N/A	12		8		ns
	T <sub>CKI</sub>	4		N/A	1		1		ns
C Clock, To Output Logic- Input Setup Logic-Input Hold	T <sub>CCO</sub>	10		N/A		25		19	ns
	T <sub>ICC</sub>	5		N/A	12		9		ns
	T <sub>CCI</sub>	6		N/A	6		1		ns
Logic Input to G Clock, To Output Logic-Input Setup Logic-Input Hold	T <sub>CIO</sub>	11		N/A		37		27	ns
	T <sub>ICI</sub>	7		N/A	6		4		ns
	T <sub>CII</sub>	8		N/A	9		5		ns
Set/Reset Direct, Input A or D to Out Through F or G to Out Master Reset Pin to Out Separation of Set/Reset Set/Reset Pulse-Width	T <sub>RIO</sub>	12		N/A		25		22	ns
	T <sub>RLO</sub>	13		N/A		37		28	ns
	T <sub>MRS</sub>			N/A		55		45	ns
	T <sub>RS</sub>			N/A	17		9		ns
	T <sub>RPW</sub>			N/A	12		9		ns
Flip-Flop Toggle Rate, Q Through F to Flip-Flop	F <sub>CLK</sub>			N/A	33		50		MHz
Clock High <sup>a</sup>	T <sub>CH</sub>	14		N/A	12		8		ns
Clock Low <sup>a</sup>	T <sub>CL</sub>	15		N/A	12		8		ns

Table 1. Electrical Performance Characteristics (Continued)

Test	Sym	Conditions -55°C ≤ Tc ≤ +125°C Vcc = 5.0 V ±10%		Group A Subgroups	2018-33 Limits		2018-50 Limits		Units
					Min	Max	Min	Max	

**Application Guidelines, Switching, IOB<sup>7</sup>**

Pad (Package Pin) to Input (Direct)	TPID	1	See Fig. 2	N/A		12		8	ns
I/O Clock									
To Input (Storage)	TLI	5		N/A		20		15	ns
To Pad-Input Setup	TPL	2		N/A	12		8		ns
To Pad Input Hold	TLP	3		N/A	0		0		ns
Pulse Width	TLW	4		N/A	12		9		ns
Frequency				N/A		33		50	MHz
Output, To Pad (Output Enable)	TOP	8		N/A		15		12	ns
Three-State, To Pad Begin hi-Z	TTHZ	9		N/A		25		20	ns
To Pad End hi-Z	TTON	10		N/A		25		20	ns
$\overline{\text{RESET}}$ , To Input (Storage)	TRI	6		N/A		40		30	ns
To Input Clock	TRC	7		N/A		35		25	ns

**Application Guidelines, Switching, Slave Mode Programming<sup>7</sup>**

CCLK, To DOUT	TCCO	3	See Fig. 6	N/A		100		100	ns
DIN Setup	TdCC	1		N/A	10		10		ns
DIN Hold	TCCD	2		N/A	40		40		ns
High Time	TcCH	4		N/A	0.5		0.5		μs
Low Time	TcCL	5		N/A	0.5	1.0	0.5	1.0	μs
Frequency	Fcc			N/A		1		1	MHz

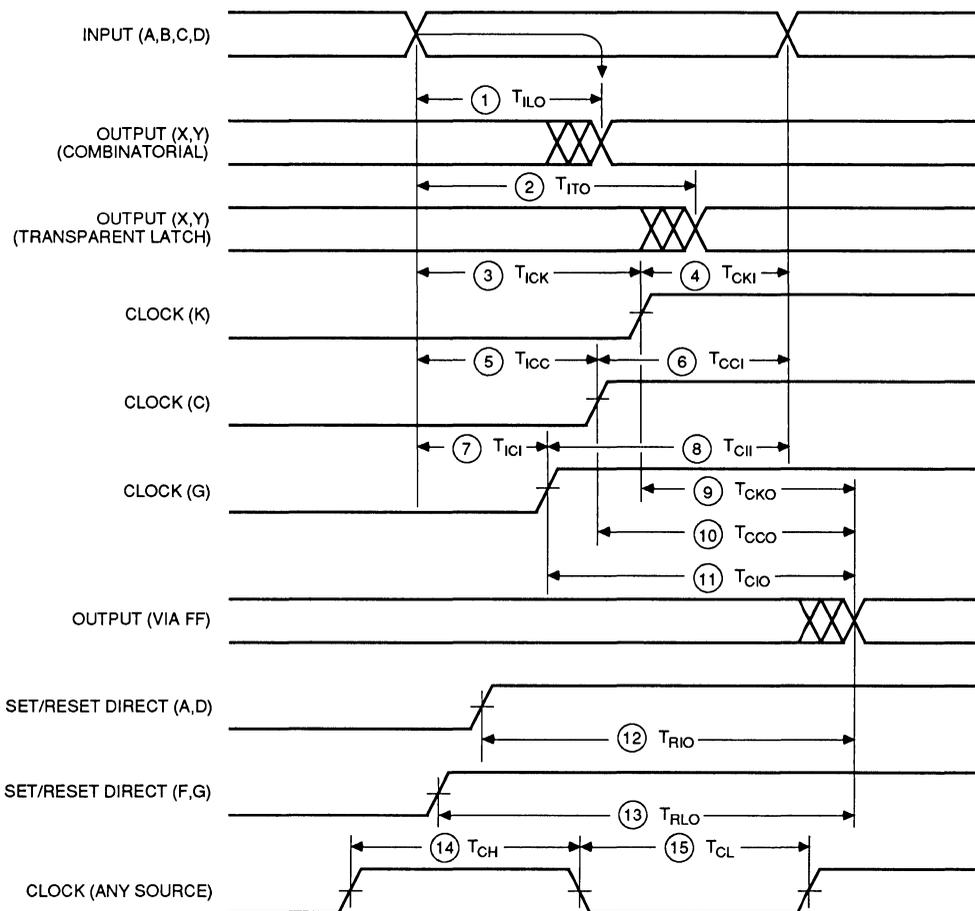
**Application Guidelines, Switching, Master Mode Programming<sup>7,9</sup>**

$\overline{\text{RCLK}}$ , From Address Invalid	TARC	1	See Fig. 5	N/A		0		0	ns
To Address Valid	TRAC	2		N/A		200		200	ns
To Data Setup	TDRC	3		N/A	60		60		ns
To Data Hold	TRCD	4		N/A	0		0		ns
$\overline{\text{RCLK}}$ High	TRCH	5		N/A	600		600		ns
$\overline{\text{RCLK}}$ Low	TRCL	6		N/A	4.0		4.0		μs

**Application Guidelines, Switching, General LCA<sup>7</sup>**

$\overline{\text{RESET}}$ <sup>10</sup> M2, M1, M0 Setup	TMR	1	See Fig. 3	N/A	1		1		μs
M2, M1, M0 Hold	TRM	2		N/A	1		1		μs
Width (Low)	TMRW	3		N/A	150		150		ns

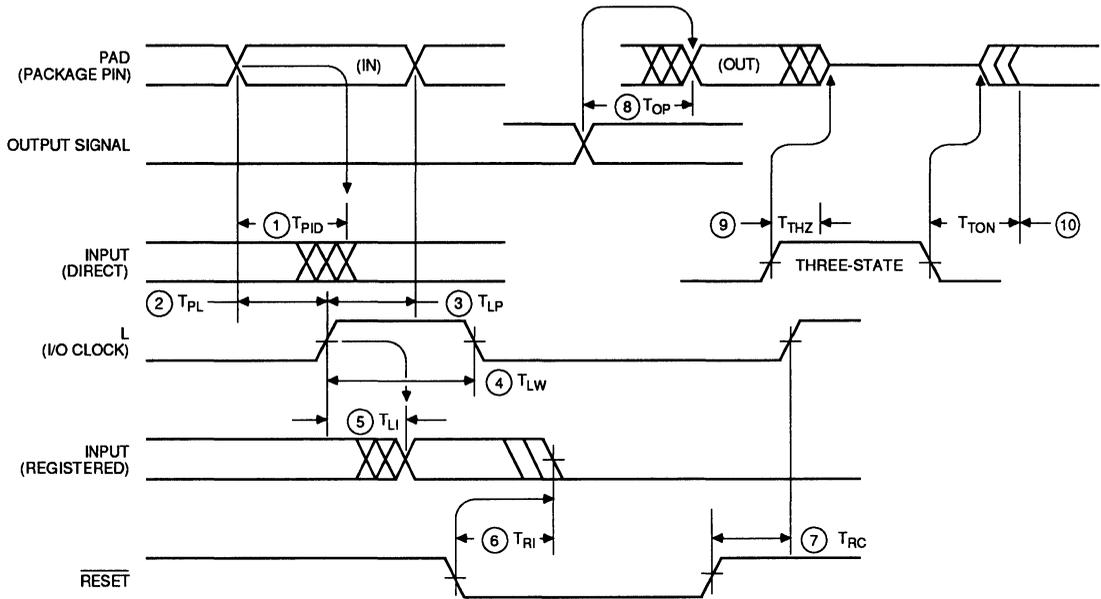
Table 1. Electrical Performance Characteristics (Continued)



Timing is measured at 0.5  $V_{CC}$  levels with 50 pF minimum output load.  
 Input signal conditioning: Rise and fall times  $\leq$  6 ns, Amplitude = 0 and 3V

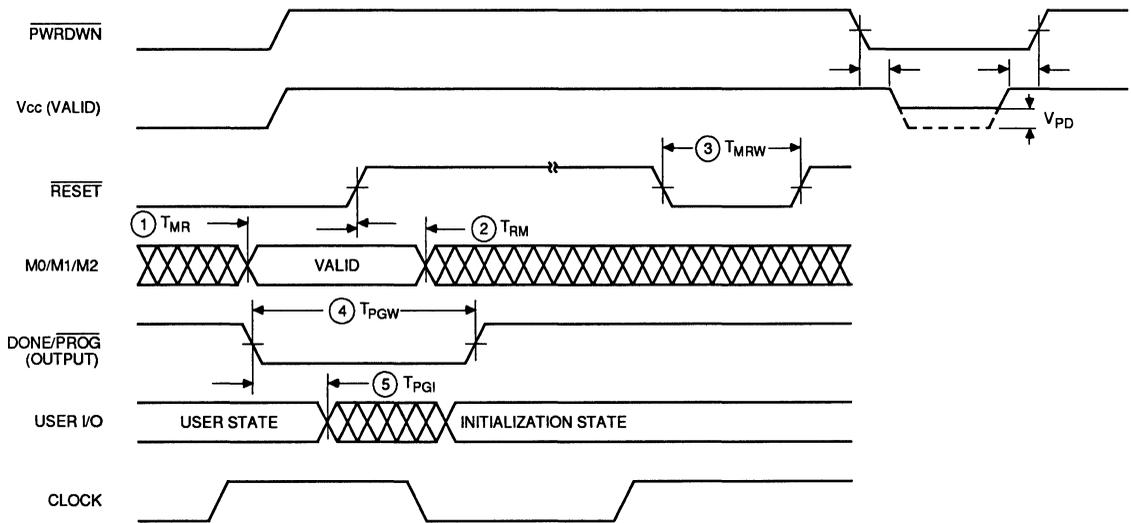
1107 06

**Figure 1. Switching Characteristics Waveforms, CLB**



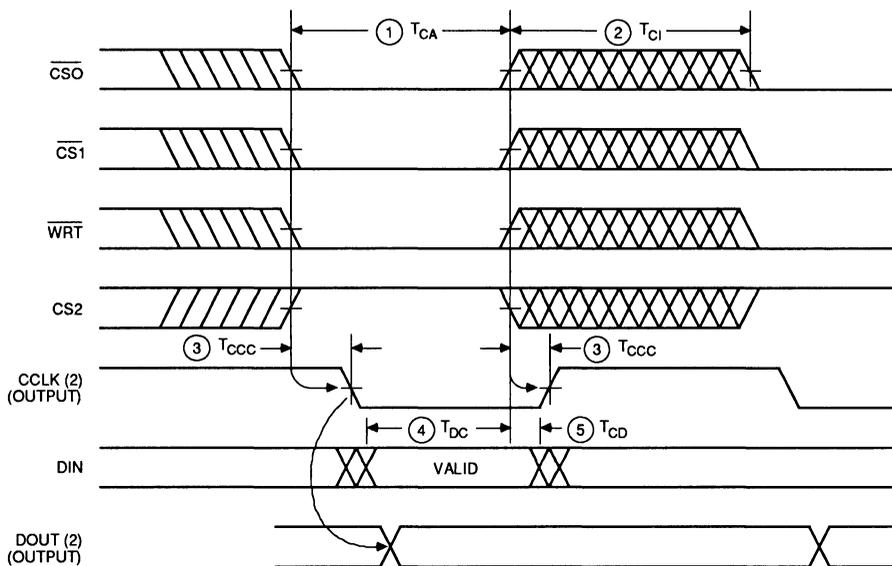
1107 07

Figure 2. Switching Characteristics, IOB



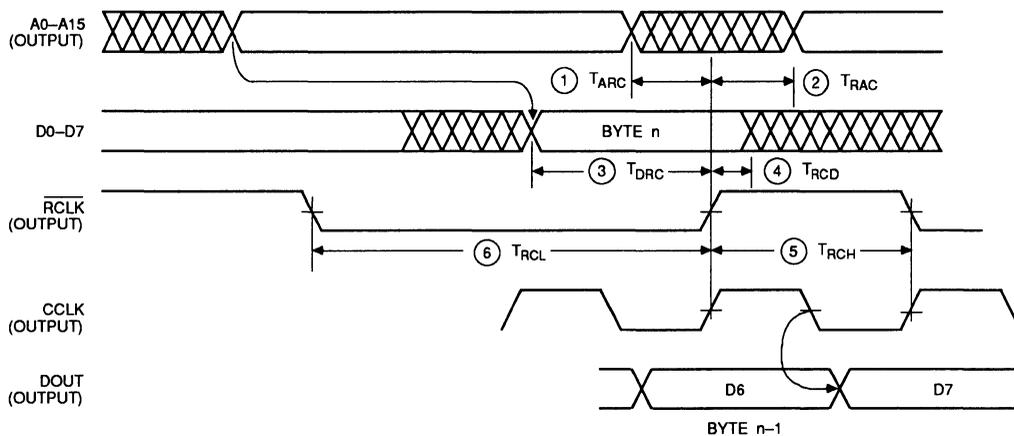
1107 08

Figure 3. General LCA Switching Characteristics



1107 09

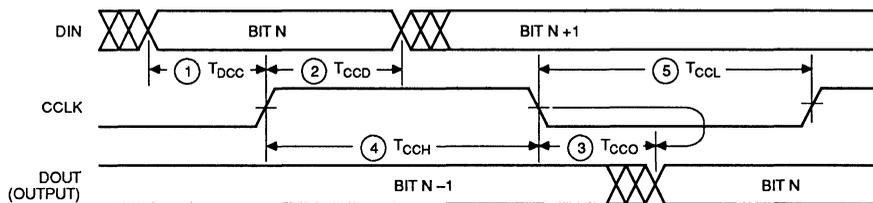
Figure 4. Peripheral Mode Programming Characteristics



CCLK and DOUT timing are the same as for slave mode.  
 At power-up,  $V_{CC}$  must rise from 2.0 V to  $V_{CC}$  min. in less than 10 ms.

1107 10

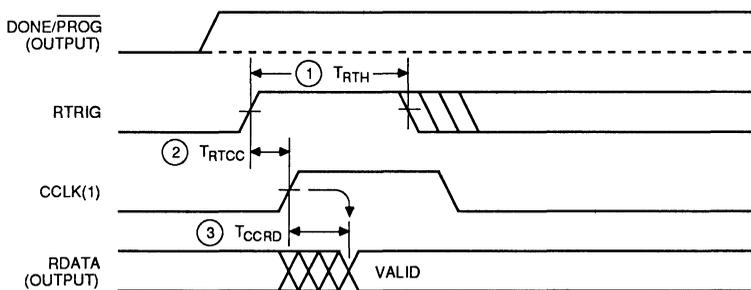
Figure 5. Master Mode Programming Switching Characteristics



Configuration must be delayed at least 40 ms after  $V_{CC}$  min.

1107 11

Figure 6. Slave Mode Programming Switching Characteristics



1107 12

Figure 7. Program Readback Characteristics

- Notes:
1. Xilinx maintains this specification as a controlled document. To comply with the intent of MIL-STD-883, and to assure that you are using the most recently released device performance parameters, please request a copy of the current revision of this Table I Test Specification from Xilinx.
  2.  $\overline{PWR\ DWN}$  must be active before  $V_{CC}$  goes below specified range, and inactive after  $V_{CC}$  reaches specified range.
  3. Peripheral mode timing determined from last control signal of the logical AND of ( $\overline{CS0}$ ,  $\overline{CS1}$ ,  $CS2$ ,  $\overline{WRT}$ ) to transition to active or inactive state.
  4. Configuration must be delayed at least 40 ms after  $V_{CC}$  min.
  5. CCLK and DOUT timing are the same as for slave mode.
  6.  $D/\overline{P}$  must be high before RTRIG goes high.
  7. Testing of the Applications Guidelines is modeled after testing specified by MIL-M-38510/605. Devices are first 100% functionally tested. Benchmark patterns are then used to measure the Application Guidelines. Characterization data are taken at initial device qualification, prior to introduction of significant changes, and at least twice yearly to monitor correlation between benchmark patterns, device performance, XACT software timings, and the data sheet.
  8. Minimum CLOCK widths for the auxiliary buffer are 1.25 times the  $T_{CLH}$ ,  $T_{CLL}$ .
  9.  $V_{CC}$  must rise from 2.0 V to  $V_{CC}$  minimum in less than 10 ms for master mode.
  10.  $\overline{RESET}$  timing relative to power-on and valid mode lines (M0, M1, M2) is relevant only when  $\overline{RESET}$  is used to delay configuration.

## Product Specification

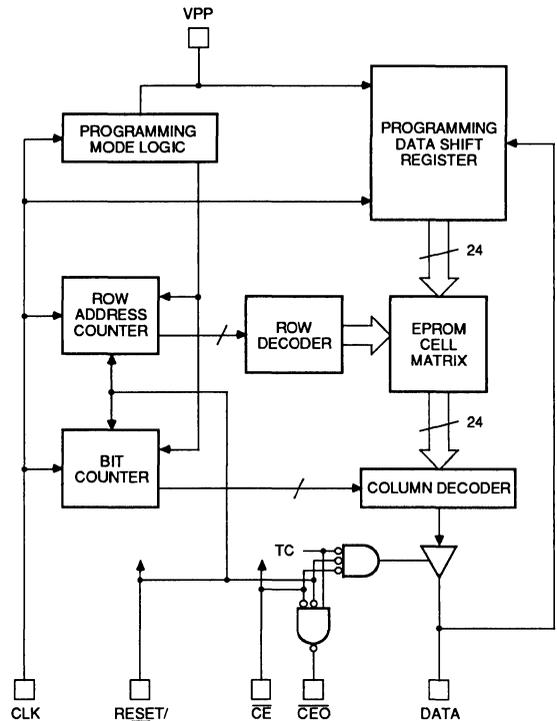
### FEATURES

- One-Time Programmable (OTP) 36,288 x 1 bit serial memory designed to store configuration programs for Programmable Gate Arrays
- Simple interface to a XILINX Logic Cell™ Array (LCA) requires only two I/O pins
- Daisy chain configuration support for three XC2064's or two XC2018's
- Cascadable to provide more memory for additional configurations or future higher-density arrays
- Storage for multiple configurations for a single Logic Cell Array
- Low power CMOS EPROM process
- Space-efficient, 8-pin plastic/ceramic DIP package
- PC-based programmer for development. Production programming support from leading programmer manufacturers

### DESCRIPTION

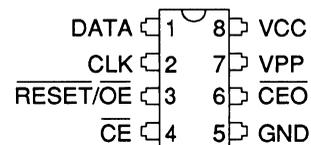
The XC1736 Serial Configuration PROM (SCP) provides an easy-to-use, cost-effective configuration memory for the Xilinx family of programmable gate arrays. Packaged in an economical 8-pin plastic DIP package, the XC1736 uses a simple serial access procedure to configure one or more Logic Cell Arrays (LCAs). The 36,288 x 1 organization of the configuration PROM supplies enough memory to configure three XC2064's or two XC2018's. Multiple Serial Configuration PROMs can be cascaded to provide a larger memory for more configurations or future higher density arrays. Multiple configurations for a single LCA can also be loaded from the XC1736.

The XC1736 can be programmed with the Xilinx XC-DS81 Configuration PROM Programmer or with programmers from other manufacturers. The Logic Cell Array design file is first compiled into a standard HEX format with the PC-based XC-DS21 Development System. It can then be transferred to the XC-DS81 programmer connected to the PC's serial port.



1106 05

Figure 1. XC1736 Block Diagram



1106 04

XC1736 Pin Assignments

Table 1. XC1736 Pin Assignments

Pin Name	I/O	Description
1 DATA	O	Three-state DATA output for reading. Input/Output pin for programming.
2 CLK	I	Clock input. Used to increment the internal address and bit counters for reading and programming.
3 RESET/ OE	I	Output Enable input. A LOW level on both the $\overline{CE}$ and $\overline{OE}$ inputs enables the data output driver. A HIGH level on RESET/ $\overline{OE}$ resets both the address and bit counters.
4 $\overline{CE}$	I	Chip Enable input. A LOW level on both $\overline{CE}$ and $\overline{OE}$ enables the data output driver. A HIGH level on $\overline{CE}$ disables both the address and bit counters and forces the device into a low power mode. Used for device selection.
5 GND		Ground pin.
6 $\overline{CEO}$	O	Chip Enable Out output. This signal is asserted LOW on the clock cycle following the last bit read from the memory. It will stay LOW as long as $\overline{CE}$ and $\overline{OE}$ are both LOW. It will follow $\overline{CE}$ , but if $\overline{OE}$ goes HIGH, $\overline{CEO}$ will stay HIGH until the entire PROM is read again.
7 VPP		Programming Voltage Supply. Used to enter programming mode (+6V) and to program the memory (+21V) Must be connected directly to Vcc for normal read operation. No overshoot above +22V permitted.
8 Vcc		+5 volt power supply input.

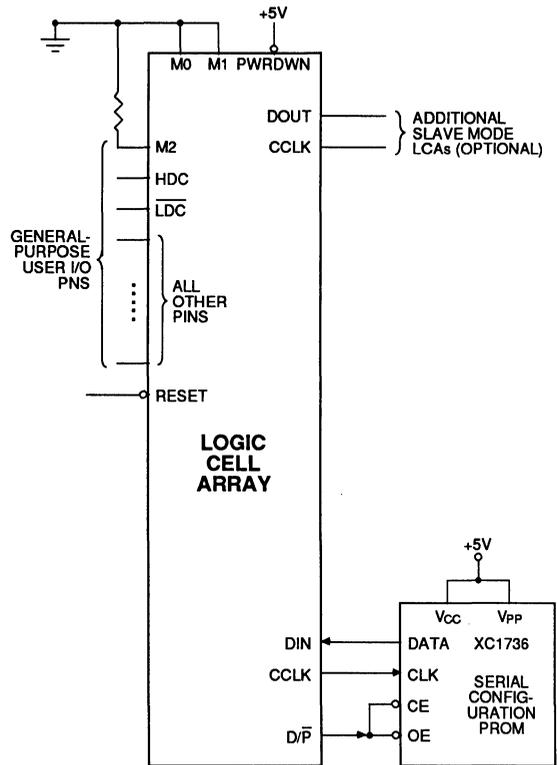


Figure 2. Master Serial Mode Configuration

1106 06

## LCA MASTER SERIAL MODE SUMMARY

The I/O and logic functions of the Xilinx Programmable Gate Array, and their associated interconnections, are established by a configuration program. The program is loaded either automatically upon power up, or on command, depending on the state of the three LCA mode pins. In Master Mode, the Logic Cell Array automatically loads the configuration program from an external memory. The Serial Configuration PROM has been designed for compatibility with the Master Serial Mode.

Upon power-up or upon reconfiguration, an LCA will enter Master Serial Mode whenever all three of the LCA's mode select pins are LOW ( $M0=0$ ,  $M1=0$ ,  $M2=0$ ). Data are read from the Serial Configuration PROM sequentially on a single data line. Synchronization is provided by the rising edge of the temporary signal CCLK, which is generated during configuration.

Master Serial Mode provides a simple configuration interface. Only a serial data line and two control lines are required to configure an LCA. Data from the Serial Configuration PROM is read sequentially, accessed via the internal address and bit counters which are incremented on every valid rising edge of CCLK.

### Programming The LCA With Counters Reset Upon Completion

Figure 2 shows the connections between an LCA and its SCP. The DATA line from the SCP is connected to the DIN input of the LCA. CCLK is connected to the CLK input of the SCP. At power-up or upon reconfiguration, the  $D/\bar{P}$  signal goes low (pulled low by the LCA at reset, or by external circuitry for reconfiguration), enabling the SCP and its DATA output. During the configuration process,

CCLK will clock data out of the SCP on every rising clock edge. At the completion of configuration, the  $DONE/\overline{PROG}$  signal will go high and reset the internal address counters of the SCP.

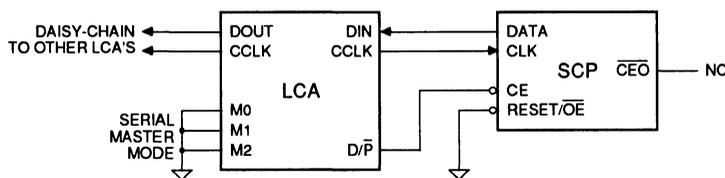
If the user-programmable, dual function DIN and CCLK pins are used only for the configuration process, they should be programmed on the LCA so that no nodes are floating or in contention. For example, both DIN and CCLK can be programmed as output highs during normal operation. An alternate method is to program both DIN and CCLK as inputs, with external pullup resistors attached.

If DIN and CCLK are to be used for another function after configuration, the user must avoid contention. The Low During Configuration ( $\overline{LDC}$ ) pin can be used to control the SCP's  $\overline{CE}$  and  $\overline{OE}$  inputs to disable the SCP's DATA pin 1 clock cycle before  $D/\bar{P}$  is active.

If the LCA is to be reprogrammed after initial power-up, note that the LCA requires several microseconds to respond after the  $D/\bar{P}$  pin is pulled low. In this case, the  $\overline{LDC}$  pin can be used instead of the  $D/\bar{P}$  pin to control the SCP.

### Programming The LCA With Counters Unchanged Upon Completion

When multiple LCA configurations for a single LCA are stored in a Serial Configuration PROM, the  $\overline{OE}$  pin should be tied low as shown in Figure 3. Upon power-up, the internal address counters will be reset and configuration will begin with the first program stored in memory. Since the  $\overline{OE}$  in is held low, the address counters are left unchanged after configuration is complete. Therefore, to reprogram the LCA with another program, the  $DONE/\overline{PROG}$  line is pulled low and configuration begins at the last value of the address counters.

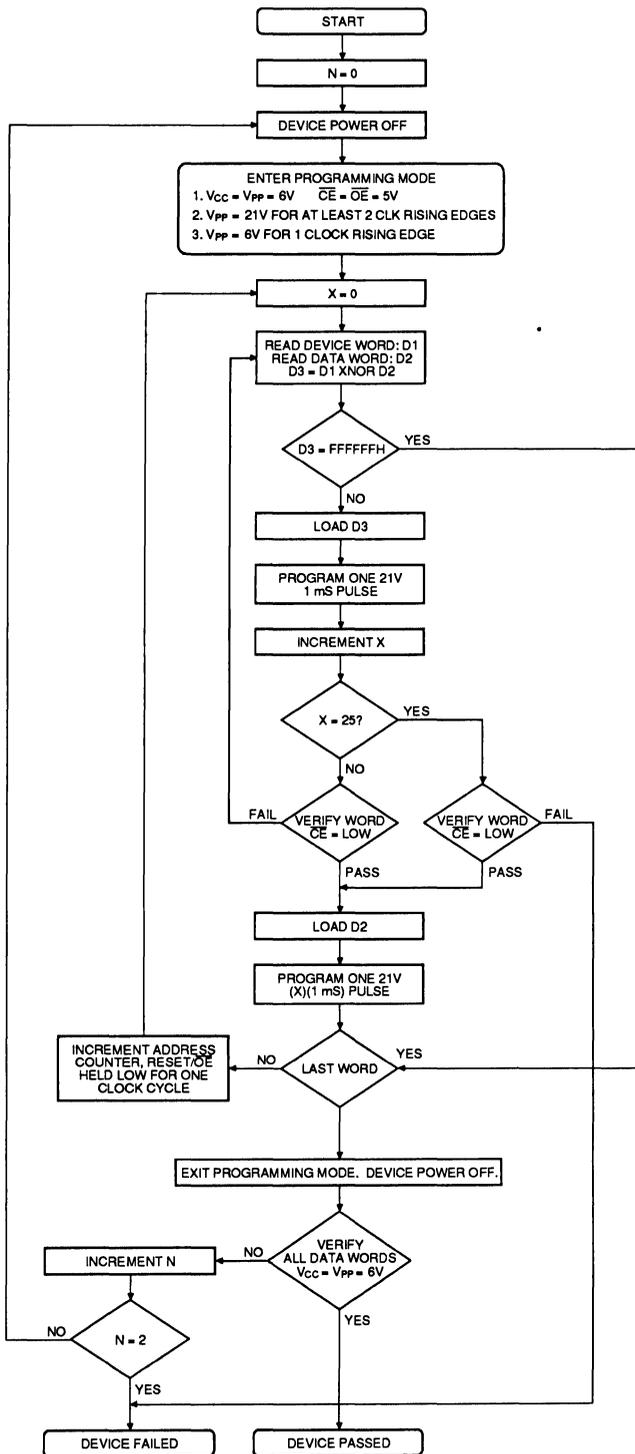


1106 07

**Figure 3. Address Counters Not Reset**

- Notes:
1. If M2 is tied directly to ground, it should be programmed as an input during operation.
  2. If the LCA is reset during configuration, it will abort back to initialization state.  $D/\bar{P}$  will not go high, so an external signal is required to reset the 1736 counters.





1106 09

Figure 5. Programming Mode

## PARAMETRICS

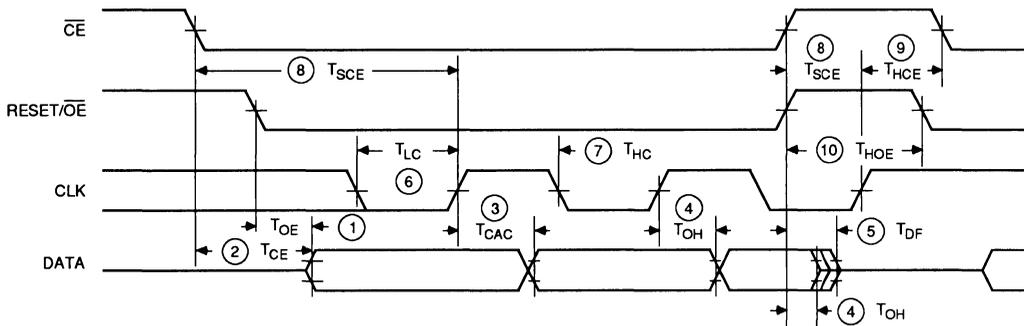
Absolute Maximum Ratings			Units
V <sub>CC</sub>	Supply voltage relative to GND	-0.5 to 7.0	V
V <sub>IN</sub>	Input voltage with respect to GND	-0.5 to V <sub>CC</sub> + 0.5	V
V <sub>TS</sub>	Voltage applied to three-state output	-0.5 to V <sub>CC</sub> + 0.5	V
T <sub>STG</sub>	Storage temperature (ambient)	-65 to + 125	°C
T <sub>SOL</sub>	Maximum soldering temperature (10 sec @ 1/16 in.)	+ 260	°C

\*Note: Stresses beyond those listed under Absolute Maximum Ratings may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those listed under Recommended Operating Conditions is not implied. Exposure to Absolute Maximum Ratings conditions for extended periods of time may affect device reliability.

## DC CHARACTERISTICS

Symbol	Description	Min	Max	Units
V <sub>CC</sub>	Supply voltage relative to GND    Comm/Ind    -40°C to 85°C	4.5	5.5	V
V <sub>PP</sub>	Supply voltage relative to GND    Military    -55°C to 125°C	4.5	5.5	V
V <sub>IH</sub>	High-level input voltage	2.0	V <sub>CC</sub>	V
V <sub>IL</sub>	Low-level input voltage	0	0.8	V
V <sub>OH</sub>	High-level output voltage (I <sub>OH</sub> = - 4mA)	3.86	0.32	V
V <sub>OL</sub>	Low-level output voltage (I <sub>OL</sub> = 4mA)			
V <sub>OH</sub>	High-level output voltage (I <sub>OH</sub> = - 4mA)	3.76	0.37	V
V <sub>OL</sub>	Low-level output voltage (I <sub>OL</sub> = 4mA)			
V <sub>OH</sub>	High-level output voltage (I <sub>OH</sub> = - 4mA)	3.7	0.4	V
V <sub>OL</sub>	Low-level output voltage (I <sub>OL</sub> = 4mA)			
V <sub>PP1</sub>	V <sub>PP</sub> programming voltage (No overshoot permitted)	20	22	V
V <sub>PP2</sub>	V <sub>PP</sub> during programming mode	5.75	6.25	V
ICCA	Supply current, active mode (I <sub>CC</sub> + I <sub>PP</sub> )		10	mA
ICCS	Supply current, standby mode (I <sub>CC</sub> + I <sub>PP</sub> )		0.5	mA
I <sub>L</sub>	Input or output leakage current	-10	10	μA
I <sub>PPP</sub>	Supply current, programming mode		50	mA

## AC CHARACTERISTICS



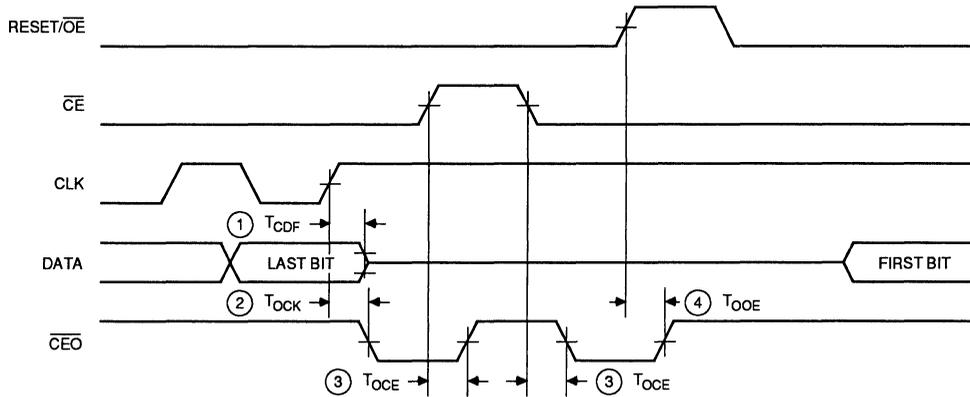
1106 10

Symbol	Description	Limits		Units
		Min	Max	
1	$T_{OE}$ $\overline{OE}$ to Data Delay		100	ns
2	$T_{CE}$ $\overline{CE}$ to Data Delay		250	ns
3	$T_{CAC}$ CLK to Data Delay		400	ns
4	$T_{OH}$ Data Hold From $\overline{CE}$ , $\overline{OE}$ , or CLK	0		ns
5	$T_{DF}$ $\overline{CE}$ or $\overline{OE}$ to Data Float Delay		50	ns
6	$T_{LC}$ CLK Low Time	200		ns
7	$T_{HC}$ CLK High Time	200		ns
8	$T_{SCE}$ $\overline{CE}$ Setup Time to CLK (Guarantees Counters Will or Will Not Change)	100		ns
9	$T_{HCE}$ $\overline{CE}$ Hold Time to CLK (Guarantees Counters Will or Will Not Change)	0		ns
10	$T_{HOE}$ $\overline{OE}$ High Time (Guarantees Counters Are Reset)	100		ns

Notes: 1. A.C. test load = 50 pF.

2. Float delays are measured with minimum tester A.C. load and maximum D.C. load.

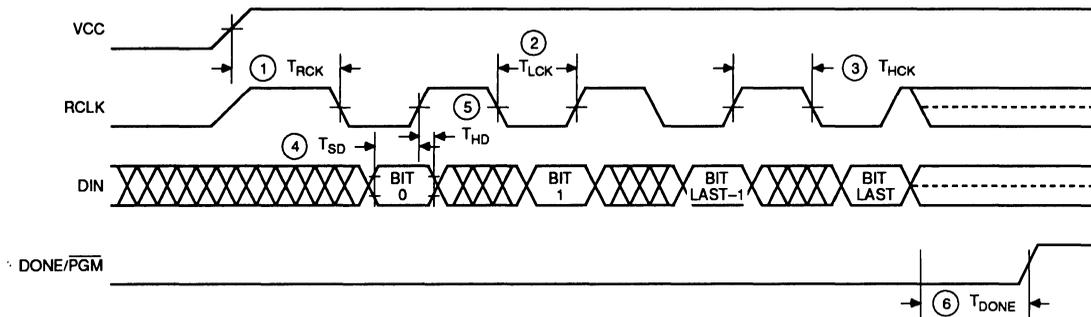
AC CHARACTERISTICS



1106 11

Symbol	Description	Limits		Units
		Min	Max	
1	$T_{CDF}$		40	ns
2	$T_{OCK}$		100	ns
3	$T_{OCE}$		100	ns
4	$T_{OOE}$		100	ns

## LCA SERIAL MASTER MODE CHARACTERISTICS

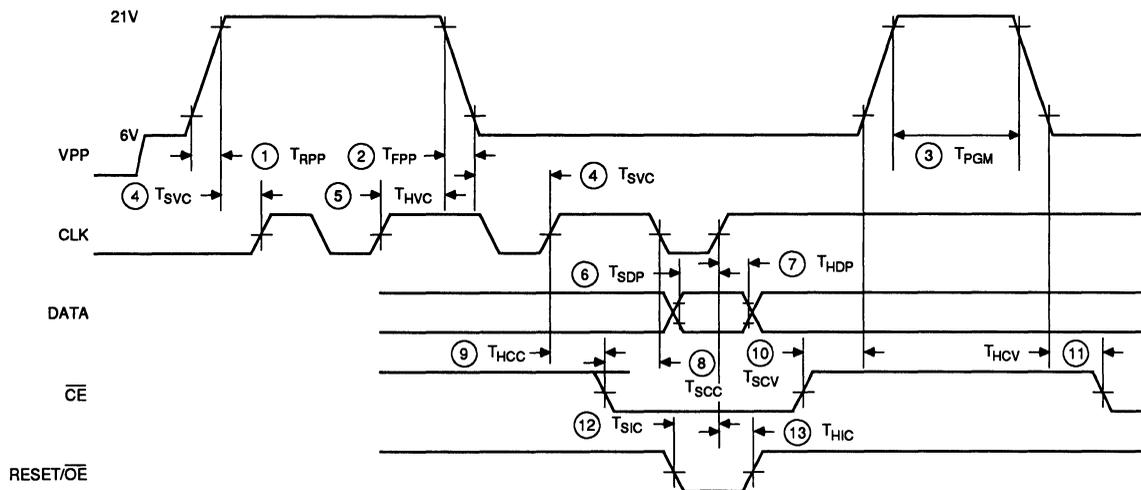


1106 12

Symbol	Description	Limits		Units
		Min	Max	
1	$T_{RCK}$	Vcc $\approx$ 3V to RCLK Delay <sup>1</sup>		ms
2	$T_{LCK}$	10	160	ns
3	$T_{HCK}$	250		ns
4	$T_{SD}$	250		ns
5	$T_{HD}$	100		ns
6	$T_{DONE}$	0		ns
			3	$\mu$ s

<sup>1</sup>This time may be extended by holding the LCA's  $\overline{RESET}$  input low.

PROGRAMMING CHARACTERISTICS



1106 13

Symbol	Description	Limits		Units	
		Min	Max		
1	TRPP	10% to 90% Rise Time of VPP	50	70	µs
2	TFPP	90% to 10% Fall Time of VPP	50	70	µs
3	TPGM	VPP Programming Pulse Width	0.95	1.05	ms
4	TSVC	Setup of VPP to CLK to Enter Programming Mode	100		ns
5	THVC	Hold of VPP to CLK to Enter Programming Mode	300		ns
6	TSDP	Data Setup to CLK for Programming	50		ns
7	THDP	Data Hold to CLK for Programming	0		ns
8	TSCC	CE Setup to CLK for Programming/Verifying	100		ns
9	THCC	CE Hold From CLK for Programming/Verifying	200		ns
10	TSCV	CE Setup to VPP for Programming	100		ns
11	THCV	CE Hold From VPP for Programming	50		ns
12	TSIC	OE Setup to CLK to Increment Address Counter	100		ns
13	THIC	OE Hold From CLK to Increment Address Counter	0		ns

\*During programming CE should only be changed while CLK is high and has been high for 200ns.

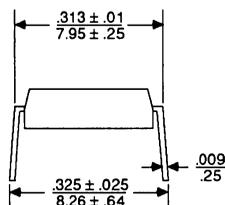
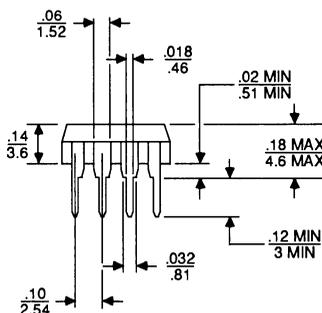
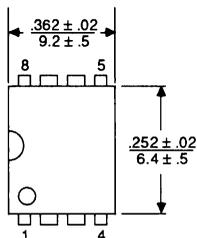
## ORDERING INFORMATION

XC1736 - PD8C

1106 01

PD8 (8 PIN PLASTIC DIP)  
CD8 (8 PIN CERAMIC DIP)

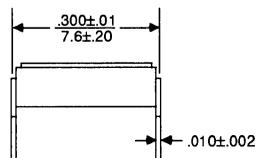
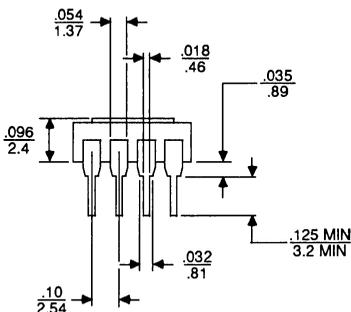
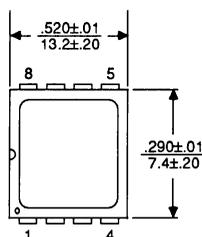
C (COMM/IND -40° TO 85°C)  
M (MILITARY TEMP -55° TO 125°)



$\frac{\text{INCHES}}{\text{MM}}$  NOT DRAWN TO SCALE

1106 02

**8-Pin Plastic DIP Package**



$\frac{\text{INCHES}}{\text{MM}}$  NOT DRAWN TO SCALE

1106 14

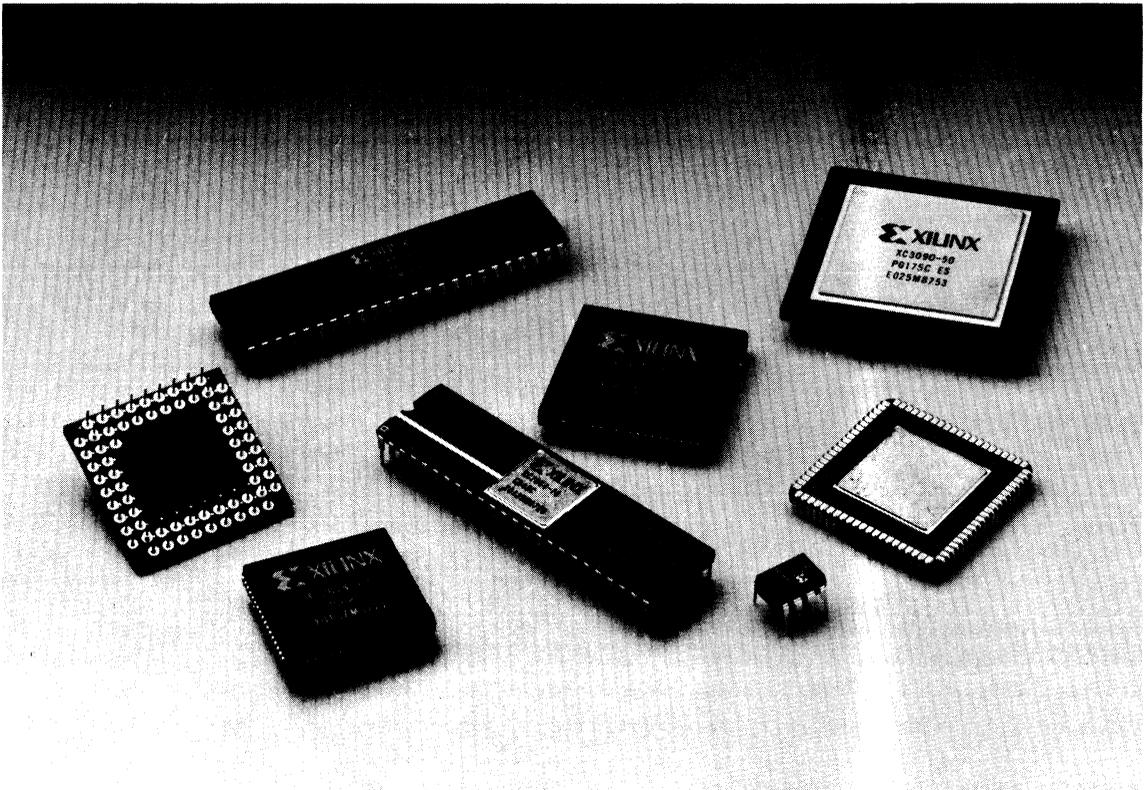
**8-Pin Ceramic Sidebraced DIP Package**

P/N 0010043



The Programmable Gate Array Company

---



**1 Programmable Gate Arrays**

**2 Product Specifications**

**3 *Quality, Testing, Packaging***

**4 Technical Support**

**5 Development Systems**

**6 Applications**

**7 Article Reprints**

**8 Index**

---



# Quality Testing, and Packaging

---

Quality Assurance and Reliability .....	3-1
Quality Assurance Program .....	3-1
Reliability Introduction .....	3-1
Die Qualification .....	3-2
Package Integrity and Assembly Quality .....	3-2
Description of Tests .....	3-4
Data Integrity .....	3-7
Alpha Particle Sensitivity .....	3-8
Electro-Static Discharge .....	3-8
Latch-up .....	3-9
Radiation Hardness .....	3-9
High-Temperature Performance .....	3-9
Test Methodology .....	3-11
Memory Cell Testing .....	3-12
Interconnect Testing .....	3-12
IOB and CLB Testing .....	3-14
Testing the Speed of the LCA .....	3-14
Packaging .....	3-17
User I/O Availability .....	3-17
Speed/Temperature Selections .....	3-17
Ceramic/Plastic Quad Flat Pack .....	3-18

---



# Quality Assurance and Reliability

---

## QUALITY ASSURANCE PROGRAM

All aspects of the Quality Assurance Program at Xilinx have been designed in compliance with the requirements of Appendix A of MIL-M-38510. This program emphasizes heavily the aspects of operator training and certification, the use of "accept only on zero defects" lot sampling plans, and extensive audits of both internal departments and outside suppliers.

Xilinx utilizes the world-class wafer fabrication facilities of Seiko-Epson's plant in Fujimi, Suwa, Japan and the high-volume assembly resources of ANAM in Seoul, the Republic of Korea. Periodic quality assurance audits of these facilities to the full requirements of MIL-STD-883 are routinely performed.

Xilinx calculates its outgoing component quality level, expressed in PPM (defective parts per million devices shipped), using the industry-standard methods now adopted by JEDEC and published in JEDEC Standard 16. These figures of merit are revised and published quarterly by Xilinx Quality Assurance and are available from local manufacturer's representatives or from Xilinx. Beginning in Q3, 1988, these summary data will be available for downloading from the Xilinx Electronic Bulletin Board at (408) 559-9323 [1200/2400 baud; 8 data bits; no parity; 1 stop bit] supporting all of the following communications protocols: ASCII, Kermit, XModem, -CRC, and Telink.

## RELIABILITY INTRODUCTION

From its inception, Xilinx has been committed to delivering the highest quality, most reliable programmable gate arrays available. A strong Quality Assurance and Reliability program begins at the initial design stages and is carried through to final shipment. The final proof of our success is in the performance of the Logic Cell™ Array (LCA) in our

customers' systems applications. An extensive, on-going reliability-testing program is used to predict the field performance of our devices.

These tests provide an accelerated means of emulating long-term system operation in severe field environments. From the performance of the devices during these tests, predictions of actual field performance under a variety of conditions can be calculated.

This report describes the nature and purpose of the various reliability tests performed on finished devices. Updated summaries are available upon request from the Quality Assurance and Reliability Department at Xilinx.

## OUTLINE OF TESTING

Qualification testing of devices is performed to demonstrate the reliability of the die used in the device, and the materials and methods used in the assembly of the device. Testing methods are derived from and patterned after the methods specified in MIL-STD-883.

Referral to the test methods of MIL-STD-883 is not intended to imply that nonhermetic products comply with the requirements of MIL-STD-883. These test methods are recognized industry-wide as stringent tests of reliability and are commonly used for nonmilitary-grade semiconductor devices, as well as for fully compliant military-grade products.

Hermetic packages are qualified using the test methods specified in MIL-STD-883. The Group D package qualification tests are performed on one lot of each package type from each assembly facility every twelve months.

A summary of the reliability demonstration tests used at Xilinx is contained in Table 1.

**DIE QUALIFICATION**

Name of Test	Test Conditions	Lot Tolerance Percent Defective Minimum Sample Size/ Maximum Acceptable Failures
1. High Temperature Life	1000 hr min. equivalent at temperature = 125°C Actual test temperature = 145°C Max. rated operating voltage. Life test circuit equivalent to MIL-STD-883	LTPD = 5, s = 105, c = 2
2. Biased Moisture Life	1000 hr min. exposure T = 85°C, RH = 85% Max. rated operating voltage. Biased moisture life circuit equivalent to MIL-STD-883	LTPD = 5, s = 105, c = 2

**NON-HERMETIC PACKAGE INTEGRITY and ASSEMBLY QUALIFICATION**

Name of Test	Test Conditions	Lot Tolerance Percent Defective Minimum Sample Size/ Maximum Acceptable Failures
3. Unbiased Pressure Pot	96 hr min. exposure T = 121°C, P = 2 atm H <sub>2</sub> O sat.	LTPD = 5, s = 105, c = 2
4. Thermal Shock	MIL-STD-883, Method 1011, Cond. C -65°C to +150°C 100 cycles	LTPD = 7, s = 75, c = 2
5. Temperature Cycling	MIL-STD-883, Method 1010, Cond. C -65°C to +150°C 200 cycles	LTPD = 5, s = 105, c = 2
6. Salt Atmosphere	MIL-STD-883, Method 1009, Cond. A 24 hrs	s = 25, c = 0
7. Resistance to Solvents	MIL-STD-883, Method 2015	s = 8, c = 0
8. Solderability	MIL-STD-883, Method 2003	s = 15, c = 0

**Table 1A. Reliability Testing Sequence for Non-Hermetic Logic Cell Arrays**

---

**HERMETIC PACKAGE INTEGRITY and ASSEMBLY QUALIFICATION**

---

Name of Test	Test Conditions	Lot Tolerance Percent Defective Minimum Sample Size/ Maximum Acceptable Failures
1. Subgroup D1: Physical Dimensions	MIL-STD-883, Method 2016	LTPD = 15, s = 34, c = 2
2. Subgroup D2 a. Lead Integrity b. Seal (fine and gross leak)	MIL-STD-883, Method 2028 MIL-STD-883, Method 1014 (not required for PGA's)	LTPD = 15, s = 34, c = 2
3. Subgroup D3 a. Thermal Shock-15 cycles b. Temp. cycling-100 cycles c. Moisture Resistance d. Seal (fine & gross leak) e. Visual Examination f. End-point electricals	MIL-STD-883, Method 1011, Cond. B MIL-STD-883, Method 1010, Cond. C MIL-STD-883, Method 1004 MIL-STD-883, Method 1014 MIL-STD-883, Method 1004 <i>and</i> Method 1010. Group A, subgroup 1	LTPD = 15, s = 34, c = 2
4. Subgroup D4 a. Mechanical Shock b. Vibration, Variable Freq. c. Constant Acceleration min, Y <sub>1</sub> only (Cond. D for large PGAs) d. Seal (fine & gross leak) e. Visual Examination f. End-point electricals	MIL-STD-883, Method 2002, Cond. B MIL-STD-883, Method 2007, Cond. A MIL-STD-883, Method 2001, Cond. E MIL-STD-883, Method 1014 MIL-STD-883, Method 1010 Group A, subgroup 1	LTPD = 15, s = 34, c = 2
5. Subgroup D5 a. Salt Atmosphere b. Seal (fine & gross leak) c. Visual Examination	MIL-STD-883, Method 1009, Cond. A MIL-STD-883, Method 1014 MIL-STD-883, Method 1009	LTPD = 15, s = 34, c = 2
6. Subgroup D6: Internal Water Vapor Content	MIL-STD-883, Method 1018, 5000 ppm water at 100°C	S = 3; c = 0 or S = 5; c = 1
7. Subgroup D7: Lead Finish Adhesion	MIL-STD-883, Method 2025	LTPD = 15, s = 34 leads, (3 device min.) c = 0
8. Subgroup D8: Lid Torque	N/A to Xilinx packages	N/A to Xilinx packages

---

**Table 1B. Reliability Testing Sequence for Hermetic Logic Cell Arrays**

## DESCRIPTION OF TESTS

### Die Qualification

- 1. High Temperature Life** This test is performed to evaluate the long-term reliability and life characteristics of the die. It is defined by the Military Standard from which it is derived as a "Die-Related Test" and is contained in the Group C Quality Conformance Tests. Because of the acceleration factor induced by higher temperatures, data representing a large number of equivalent hours at a normal temperature of 70°C can be accumulated in a reasonable period of time. Xilinx performs its High Temperature Life test at a higher temperature, 145°C, than the more common industry practice of 125°C. For comparison, the Reliability Testing Data Summary in Table 2 gives the equivalent testing hours at 125°C.
- 2. Biased Moisture Life** This test is performed to evaluate the reliability of the die under conditions of long-term exposure to severe, high-moisture environments which could cause corrosion. Although it clearly stresses the package as well, this test is typically grouped under the die-related tests. The device is operated at maximum-rated voltage, 5.5 VDC, and is exposed to a temperature of 85°C and a relative humidity of 85% throughout the test.

### Package Integrity and Assembly Qualification

- 3. Unbiased Pressure Pot** This test is performed at a temperature of 121°C and a pressure of 2 atm. of saturated steam to evaluate the ability of the plastic encapsulating material to resist water vapor. Moisture penetrating the package could induce corrosion of the bonding wires and nonglassivated metal areas of the die [bonding pads only for Xilinx LCAs]. Under extreme conditions, moisture could cause drive-in and corrosion under the glassivation. Although it is difficult to correlate this test to actual field conditions, it provides a well-established method for relative comparison of plastic packaging materials and assembly and molding techniques.
- 4. Thermal Shock** This test is performed to evaluate the resistance of the package to cracking and resistance of the bonding wires and lead frame to separation or damage. It involves nearly instantaneous change in temperature from -65°C. to +150°C.
- 5. Temperature Cycling** This test is performed to evaluate the long-term resistance of the package to

damage from alternate exposure to extremes of temperature or to intermittent operation at very low temperatures. The range of temperatures is -65°C to +150°C. The transition time is longer than that in the Thermal Shock test but the test is conducted for many more cycles.

- 6. Salt Atmosphere** This test was originally designed by the US Navy to evaluate resistance of military-grade ship board electronics to corrosion from sea water. It is used more generally for nonhermetic industrial and commercial products as a test of corrosion resistance of the package marking and finish.
- 7. Resistance to Solvents** This test is performed to evaluate the integrity of the package marking during exposure to a variety of solvents. This is an especially important test, as an increasing number of board-level assemblies are subjected to severe conditions of automated cleaning before system assembly operations occur. This test is performed according to the methods specified by MIL-STD-883.
- 8. Solderability** This test is performed to evaluate the solderability of the leads under conditions of low soldering temperature following exposure to the aging effects of water vapor.
- 9. Vibration, Variable-Frequency** This test is performed to evaluate the resistance of the completed assembly to vibrations during storage, shipping, and operation.

## TESTING FACILITIES

Xilinx has the complete capability to perform High Temperature Life Tests, Thermal Shock, Biased Moisture Life Tests, and Unbiased Pressure Pot Tests in its own Reliability Testing Laboratory. Other tests are being performed by outside testing laboratories with DESC laboratory suitability for each of the test methods they perform.

## SUMMARY

The Table 2 testing data shows the actual performance of the Logic Cell Arrays during the initial qualification tests to which they have been subjected. These test results demonstrate the reliability and expected long life inherent in the nonhermetic product line. This series of tests is ongoing as a part of the Quality Conformance Program on nonhermetic devices.

**XILINX XC2018 and XC2064 Reliability Testing Summary**

Device Type: XC2018 and XC2064  
 Die Attach Method: Silver Epoxy  
 Molding Compound: Nitto MP 150 SG

Process/Technology: 1.5 and 2.0 Micron Double Layer Metal CMOS  
 Package Type: 68 & 84 Lead PLCC  
 Date: 2Q, 1988

Test Description	Combined Sample	Failures	Equivalent Mean Hrs/Device at T = 125°C	Equivalent Device Hrs at T = 125°C	Equivalent Failure Rate in FIT at T <sub>j</sub> = 70°C
1. High Temperature Life Test 145°C	2,308	20	1,288	2,857,745	104
2. Biased Moisture Life Test T = 85°C; RH = 85%	508	5	996	505,780	
3. Unbiased Pressure Pot Test +121°C, 2 atm sat. steam	478	5	246	117,350	
4. Thermal Shock Test -65°C/+150°C 100 cy. (min)	154	0	1,100	169,400	
5. Temperature Cycling Test -65°C/+150°C 100 cy. (min)	210	0	1,000	210,000	
6. Salt Atmosphere Test MIL-STD-883, Method 1009, Cond. A	50	0	24	1,200	
7. Resistance to Solvents Test MIL-STD-883, Method 2105	16	0			
8. Solderability Test MIL-STD-883, Method 2003	30	0			

**Table 2. Reliability Testing Summary, 1.5 and 2.0 Micron**

**XILINX XC2018, XC2064, and XC3020 Reliability Testing Summary, Initial Lots**

---

Device Type: XC2018, XC2064, and XC3020  
Die Attach Method: Silver Epoxy  
Molding Compound: Nitto MP 150 SG

Process/Technology: 1.2 Micron Double Layer Metal CMOS  
Package Type: 68 & 84 Lead PLCC  
Date: 2Q, 1988

1. High Temperature Life Test 145°C	Combined Sample	Failures	Equivalent Mean Hrs/Device at T = 125°C	Equivalent Device Hrs at T = 125°C	Equivalent Failure Rate in FIT at T <sub>j</sub> = 70°C
	1,964	42	2,491	4,892,851	131

**Table 2a. Reliability Testing Summary, 1.2 micron**

## DATA INTEGRITY

### Memory Cell Design

An important aspect of the Logic Cell Array's reliability is the robustness of the static memory cells used to store the configuration program.

The basic cell is a single-ended five-transistor memory element (Figure 1). By eliminating a sixth transistor, which would have been used as a pass transistor for the complementary bit line, a higher circuit density is achieved. During normal operation, the outputs of these cells are fixed, since they determine the user configuration. Write and readback times, which have no relation to the device performance during normal operation, will be slower without the extra transistor. In return, the user receives more functionality per unit area.

This explains the basic cell, but how is the Logic Cell Array user assured of high data integrity in a noisy environment? We must consider three different situations: normal operation, a write operation and a read operation. In the normal operation, the data in the basic memory element is not changed. Since the two circularly linked inverters that hold the data are physically adjacent, supply transients result in only small relative differences in voltages. Each inverter is truly a complementary pair of transistors. Therefore, whether the output is high or low, a low impedance path exists to the supply rail, resulting in extremely high noise immunity. Power supply or ground transients of several volts have no effect on stored data.

The transistor driving the bit line has been carefully designed so that whenever the data to be written is opposite the data stored, it can easily override the output of the feedback inverter. The reliability of the write operation is

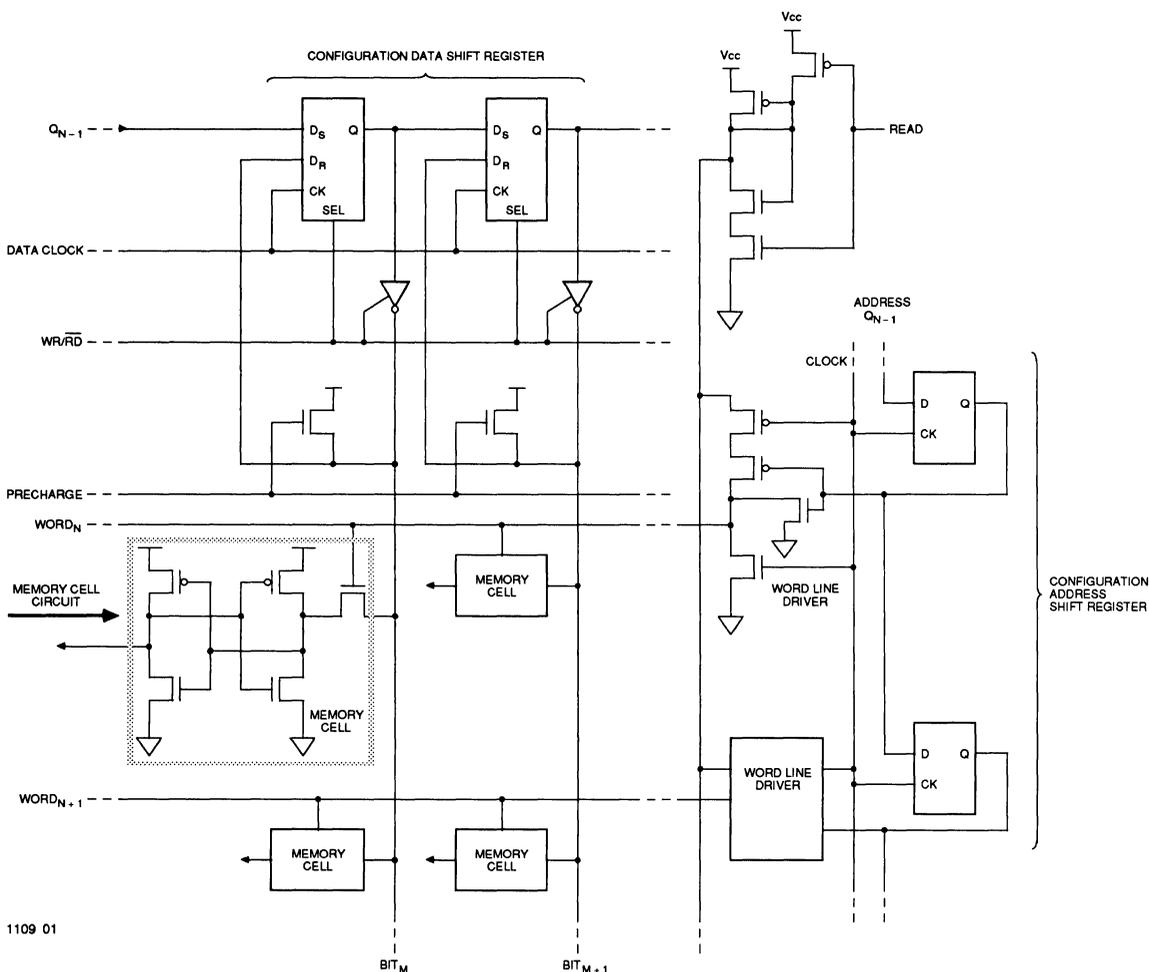


Figure 1. Configuration Memory Cell

guaranteed within the tolerances of the manufacturing process.

In the read mode, the bit line, which has a significant amount of parasitic capacitance, is precharged to a logic one. The pass transistor is then enabled by driving the word line high. If the stored value is a zero, the line is then discharged to ground. Reliable reading of the memory cell is achieved by reducing the word line high level during reading to a level that insures that the cell will not be disturbed.

**Alpha Particle (Soft Error) Sensitivity:**

The CMOS static memory cell was designed to be insensitive to alpha particle emissions. To verify that this design goal was achieved, the following tests were performed.

A one microcurie alpha particle source (Americum 241) was placed in direct contact with the top surface of an XC2064 die. This allows the die to capture at least 40% of the emissions from the radiation source. The following sequence of tests was performed:

1. A complex pattern containing roughly 50% logic ones was loaded into the XC2064. The operating conditions were 25°C and 5.0 volts.
2. A pause of variable duration was allowed.
3. The entire contents of the XC2064 were read back and compared with the original data.

Validation tests to ensure that the test setup would detect errors were performed before and after the alpha particle tests. The results are as follows:

Test	Time Duration	Readback Time	Total Time Exposed	Number of Errors
1	10 sec	70 sec	80 sec	0
2	120 sec	70 sec	190 sec	0
3	300 sec	70 sec	370 sec	0
4	1500 sec	70 sec	1570 sec	0
Total			2210 sec	

**Analysis**

A one microcurie source emits  $3.7 \times 10^4$  alpha particles per second. Assuming that 40% of these are captured by the XC2064 during this experiment; this corresponds to  $5.3 \times 10^7$  alpha particles per hour.

The alpha particle emission rate of the molding compound used by Xilinx is specified to emit fewer than 0.003 alpha particles per square centimeter per hour (alpha particles/cm<sup>2</sup>/hr). The surface area of the XC-2064 die is less than 0.5 cm<sup>2</sup>, so less than 0.0015 alpha particles per hour will be captured by the XC2064 in normal operation. The error rate acceleration in this test is therefore equal to:

$$\frac{5.3 \times 10^7 \text{ particles/hour}}{0.0015 \text{ particles/hour}} = 3.6 \times 10^{10}$$

The 0.61 hours of test time without error then corresponds to  $2.2 \times 10^{10}$  hours or 2.5 million years of error-free operation.

Most ceramic packages are specified to emit less than 0.01 alpha particles/cm<sup>2</sup>/hr which is about three times more than the plastic compound. For an XC2064 in a ceramic package, this still results in error-free operation for almost a million years.

The highest rate of alpha particle emission comes from the sealing glass used in cerdip packages and some ceramic packages (frit lids). For instance, KCIM glass emits about 24 alpha particles/cm<sup>2</sup>/hr. Low alpha glasses are specified at 0.8 alpha particles/cm<sup>2</sup>/hr.

Because these glasses are used only for the package seal, they present a relatively small emitting cross section to the die (less than 0.1 cm<sup>2</sup> square). A low alpha glass would therefore cause fewer than 0.8 alpha particle hits per hour. The acceleration factor is then  $6.6 \times 10^8$ , which translates to about 46,000 years without an error.

The memory cell of the Xilinx Logic Cell Array has been designed so that soft errors caused by alpha particles can safely be ignored.

**ELECTROSTATIC DISCHARGE**

Electrostatic discharge (ESD) protection for each pad is provided by a circuit that uses forward and reverse biased distributed resistor-diodes (Figure 2). In addition, inherent capacitance integrates any current spikes. This gives sufficient time for the diode and breakdown protections to provide a low impedance path to the power-supply rail. Geometries and doping levels are optimized to provide sufficient ESD protection for both positive and negative discharge pulses.

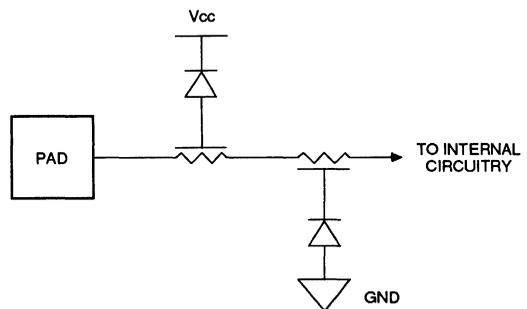


Figure 2. Input Protection Circuitry





Xilinx is committed to providing the highest level of quality and reliability for the Logic Cell™ (LCA) Array. Quality is best assured by taking the necessary steps to achieve zero defects. Comprehensive testing confirms that every Logic Cell Array is free from defects and conforms to the data sheet specifications. The memory cell design assures integrity of the configuration program.

## TESTING

As quality consciousness has grown among semiconductor users, awareness of the importance of testability has also increased. Testing for standard components, including memories and microprocessors, is accomplished with carefully developed programs which exhaustively test the function and performance of each part. For reasons explained below, most application specific ICs cannot be comprehensively tested. Without complete testing, defective devices might escape detection and be installed into a system. In the best case, the failure will be detected during system testing at a higher cost. In the worst case, the failure will be detected only after shipment of the system to a customer.

Testing advantages of the Logic Cell Array can be illustrated through comparison with two other application specific ICs: Erasable Programmable Logic Devices (EPLDs) and gate arrays.

**EPLDs:** In order to test all memory cells and logic paths of programmable logic devices controlled by EPROM memory cells, the part must be programmed with many different patterns. This in turn requires expensive quartz lid packages and many lengthy program/test/erase cycles. To save time and reduce costs, this process is typically abbreviated.

**Gate Arrays:** Since each part is programmed with metal masks, the part can only be tested with a program tailored to the specific design. This in turn requires that the designer provide sufficient controllability and observability for comprehensive testability. The design schedule must also include time for the development of test vectors and a test program specification. If the gate array user requires a comprehensive test program, then he must perform exhaustive and extensive fault simulation and test grading. This requires substantial amounts of expensive computer time. Additionally, it typically requires a series of

time-consuming and expensive iterations in order to reach even 80% fault coverage. The cost of greater coverage is often prohibitive. In production, many gate array vendors either limit the number of vectors allowed or charge for using additional vectors.

The replacement of all storage elements with testable storage elements, known as scan cells, improves testability. Although this technique can reduce the production testing costs, it can add about 30% more circuitry, decrease performance by up to 20%, and increase design time.

**Logic Cell Arrays:** The testability of the Logic Cell Array is similar to other standard products, including microprocessors and memories. This is the result of the design and the test strategies:

### *Design strategy:*

- Incorporates testability features because each functional node can be configured and routed to outside pads
- Permits repeated exercise of the part without removing it from the tester because of the short time to load a new configuration program
- Produces a standard product which guarantees that every valid configuration will work.

### *Test strategy:*

- Performs reads and writes of all bits in the configuration memory, as in memory testing
- Uses an efficient parallel testing scheme in which multiple configurable logic blocks are fully tested simultaneously
- Is exhaustive since the circuits in every block are identical

The Logic Cell Array user can better appreciate the Logic Cell Array test procedure by examining each of the testing requirements:

- All of the configuration memory bits must be exercised and then verified. This is performed using readback mode.
- All possible process-related faults, such as short circuits, must be detected. The Logic Cell Array is config-

ured such that every metal line can be driven and observed directly from the input/output pads.

- All testing configurations must provide good controllability and observability. This is possible since all configurable logic blocks can be connected to input/output pads. This makes them easy to control by testing different combinations of inputs and easy to observe by comparing the actual outputs with expected values.

These points bring out an important issue: the Logic Cell Array was carefully designed to achieve 100% fault coverage. With the Xilinx testing strategy, the number of design configurations needed to fully test the Logic Cell Array is minimized and the test fault coverage of the test patterns is maximized. In addition, the user's design time is reduced because the designer does not have to be concerned about testability requirements during the design cycle. The Logic Cell Array concept not only removes the burden of the test program and test vector generation from the user, but also removes the question of fault coverage and eliminates the need for fault grading. The Logic Cell Array is a standard part that guarantees any valid design will work. These issues are critically important in quality-sensitive applications. The designer who uses the Logic Cell Array can build significant added value into his design by providing higher quality levels.

### TESTING OF THE LOGIC CELL ARRAY

The logic cell array is tested as a standard product. Every device is tested for: 1) 100% functionality; 2) D.C. parametrics; and 3) speed. This allows the end-user to design and use the logic cell array without worrying about testing for a particular application.

The strategy for testing the logic cell array is to test the functionality of every element inside the LCA. These elements consist of memory cells, metal interconnects, transistor switches, bidirectional buffers, inverters, decoders, and multiplexers. If each element is functional, then the user's design will also be functional if the proper design procedures are used.

The static memory cells and the symmetry of the logic cell array make it 100% testable. The logic cell array can be programmed and reprogrammed with as many patterns as required to fully test it. This is done with as many as 50 configuration/test patterns. Each configuration/test pattern consist of: 1) A set of test vectors that configure the LCA with a hardware design that utilizes specific elements; and 2) A set of test vectors that exercise those specific elements. The symmetry of the LCA allows the test engineer to develop the test for one CLB or IOB and then apply it to all others. All configuration/test patterns are exercised at both Vcc minimum and maximum.

### Memory Cell Testing

The static memory cells have been designed specifically for high reliability and noise immunity. The basic memory cell consists of two CMOS inverters and a pass transistor used for both writing and reading the memory cell data (See Figure 1). The cell is only written during configuration. Writing is accomplished by raising the gate of the pass transistor to Vcc and forcing the two CMOS inverters to conform to the data on the word line. During normal operation the memory cell provides continuous control of the logic, and the pass transistor is "off" and does not affect memory cell stability. The output capacitive load and the CMOS levels of the inverters provide high stability. The memory cells are not affected by extreme power supply excursions.

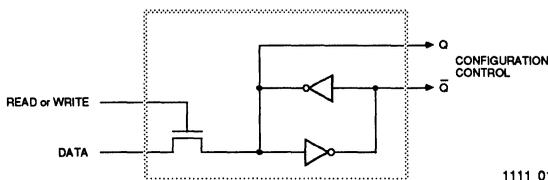


Figure 1. Configuration Memory Cell

The memory cells are directly tested in the logic cell array with three test patterns that are equivalent to those used on a random access memory device. The first test pattern writes 95% of all the RAM cells to a logical zero and then reads each RAM cell back to verify its contents. The second test pattern writes 95% of all the RAM cells to a logical one and also verifies the contents. The third pattern is used to verify that all I/O and configurable logic blocks can have their logic value read back correctly. All RAM cells are thus written and verified for both logic levels.

### Interconnect Testing

The programmable interconnect is implemented using transistor switches to route signals through a fixed two layer grid of metal conductors. The transistor switches "on" or "off" depending on the logic value of the static memory cell that controls the switch. The interconnect is tested with configuration/test patterns that: 1) Test for continuity of each metal segment; 2) Test for shorts between metal segments; and 3) Check the ability for each switch to connect two metal lines. This can be accomplished with a pattern similar to Figure 2. Each interconnect line will be set to a logic 'one' while the others are set to logic zero. This checks for shorts between adjacent interconnects while at the same time checking for continuity of the line.



### **I/O Block Testing**

Each I/O Block includes registered and direct input paths and a programmable three-state output buffer. The testing of these functions is accomplished by several configuration/test patterns that implement and test each option that is available to the user. One method used to test the I/O blocks is to configure them as a shift register that has a Tri-State control (See Figure 3). This allows a test pattern to check the ability of each I/O block to latch and output data that is derived from either the previous I/O block or from the tester. Several of these patterns are used to exercise different input and output combinations allowed for each I/O block. Configuration/test patterns are also used to precondition the device to test D.C. parameters such as VIH, VIL, VOH, VOL, TTL standby current, CMOS standby current and input/output leakage. The VOH/VOL Test is done while all outputs are either all Low or all High.

### **Configurable Logic Block Testing**

Each configurable logic block has a combinatorial logic section, a flip-flop section, and an internal control section. The combinatorial logic section of the logic block uses an array of RAM cells (16x1 in or 32X1 in) as a look up table to implement the Boolean functions. This section is tested as an array of memory cells. Configuration/test patterns are used to verify that each RAM cell can be logically decoded as the output of the array. The flip-flop section of the logic block is tested with configuration/test patterns that configure the LCA as shift registers. Each shift register pattern will have different data in the look-up tables and will have a different pin used as the input to each shift register. Other configuration/test patterns are used to implement and test the internal control section.

### **TESTING THE SPEED OF THE LOGIC CELL ARRAY**

The speed of the device is checked with configuration/test patterns that have been correlated to data sheet A.C. values.

Most of these patterns are shift registers with interconnect, IOBs and CLBs in the data path (See Figure 4). They are designed with the idea that all elements in the path must be fast enough for the proper data to get to the next input of the shift register before the next clock occurs. If any element doesn't meet the specified A.C. value, then the shift register will clock in the wrong data and fail the test. The complexity of the logic between two shift register cells determines the maximum frequency required for the clock pulse input of the shift register. This can be used to reduce the performance requirement of the tester in use. The patterns used consist of a TCKO + TILO + INTERCONNECT + TICK for each shift register. This increases the shift register clock pulse separation time to 30 to 40 ns. The configuration of each pattern is varied so that all of the interconnect, IOBs, and CLBs are tested at speed.

### **HARDWARE TESTING CONSIDERATIONS FOR THE LCA**

Currently the logic cell array is being tested on Sentry testers. The 68 and 84 pin versions can be tested on a 60 pin tester with 256K of extended local memory. The 3000 series products are being tested on a 120 pin tester with 512K of extended local memory.

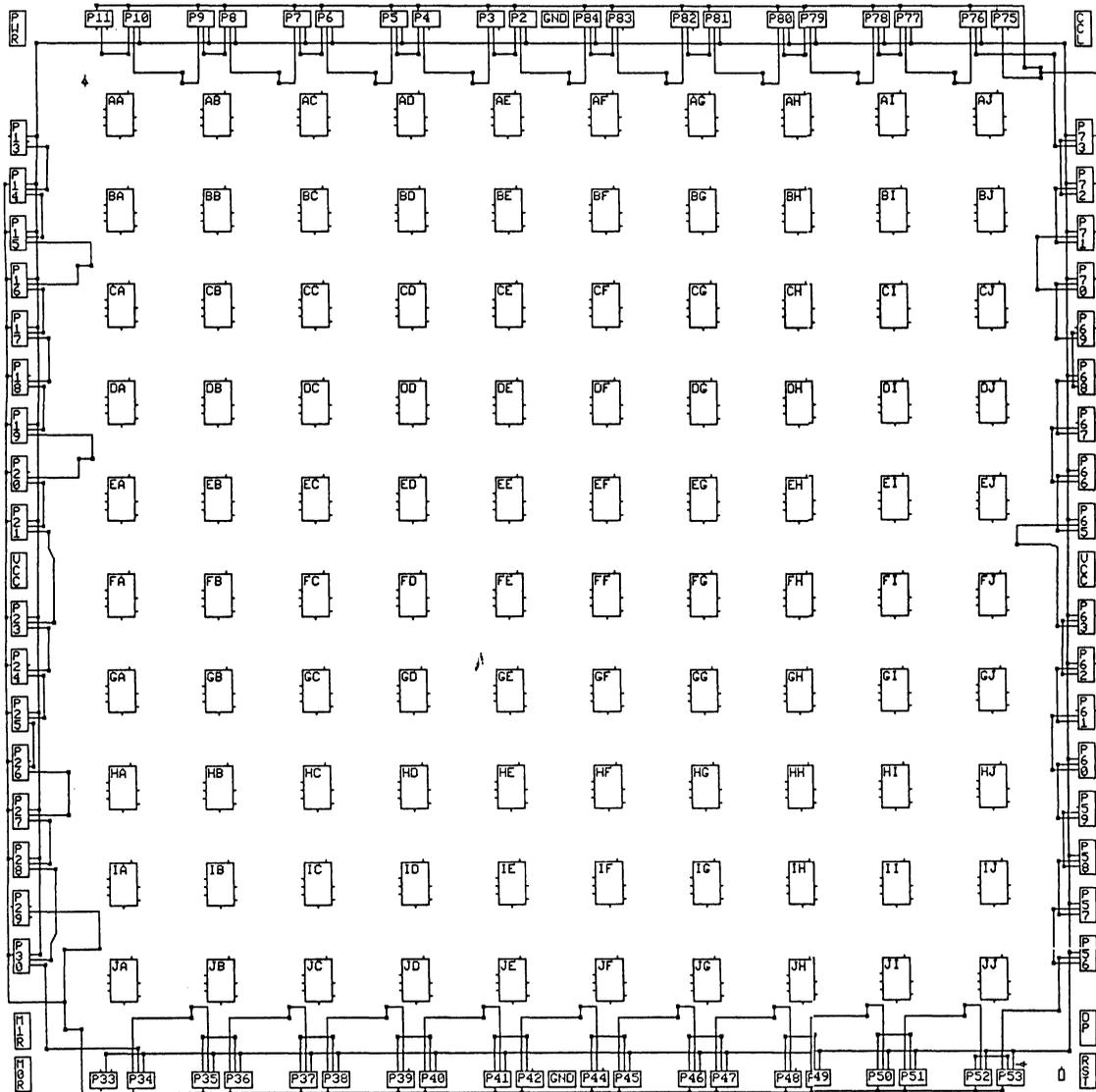


Figure 3. IOB Test Pattern

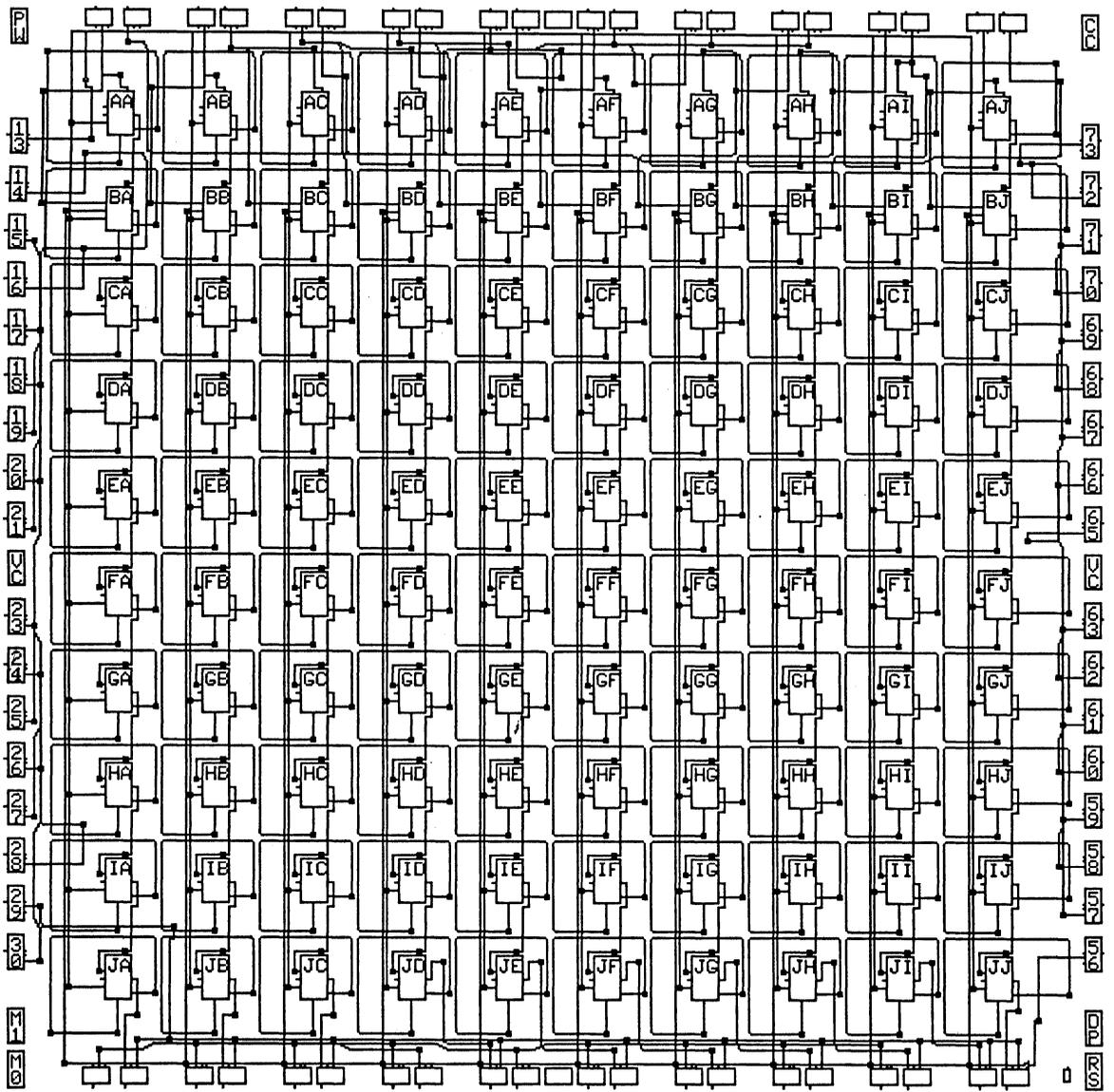


Figure 4. Speed Test Pattern



# Packaging

## PACKAGE AND USER I/O AVAILABILITY

Number of User I/O Available

	48 PIN	68 PIN	84 PIN	132 PIN	164 PIN	175 PIN
XC2064	40	58				
XC2018		58	74			
XC3020		58	64			
XC3030			74			
XC3042			74	96		
XC3064				110		
XC3090					142	144

1112 02

## PACKAGE/SPEED/TEMPERATURE SELECTIONS

		48 PIN		68 PIN		84 PIN		132 PIN	175 PIN	CERAMIC QUAD FLAT PACK
		PLASTIC DIP	CERAMIC DIP	PLASTIC PLCC	CERAMIC PGA	PLASTIC PLCC	CERAMIC PGA	CERAMIC PGA	CERAMIC PGA	
		-PD 48	-CD 48	-PC 68	-PG 68	-PC 84	-PG 84	-PG 132	-PG 175	-CQXXX
XC2064	-50	C	I	CI	CIMB					UNDER DEVELOPMENT
	-70			C	C					
XC2018	-50			CI		CI	CIMRB			
	-70			C		C	C			
XC3020	-50			CI		CI	CIMB			
	-70			CI		CI	CI			
XC3030	-50					CI	CIMB			
	-70					C	C			
XC3042	-50					CI	CIM	CIMB		
	-70					C	C	C		
XC3064	-50							CIMB		
	-70							C		
XC3090	-50								CIMB	
	-70								C	

C = COMMERCIAL                      0°C TO 70°C  
 I = INDUSTRIAL                      -40°C TO 85°C  
 M = MILITARY TEMP                 -55°C TO 125°C  
 R = HI REL                             -55°C TO 125°C  
 B = MILITARY                         MIL-STD-883, CLASS B

1112 012

**CERAMIC QUAD FLAT PACK (CQFP)**

The Ceramic Quad Flat Pack (also called Quad Cerpack) is a cavity down, pressed ceramic package. The leads are gull-wing, on four sides, with 25 Mil pitch. It is for surface mount Commercial, Industrial, and Military (including MIL-STD-883 Class B) applications. JEDEC has developed a standard that Xilinx will follow.

The initial version will be a 164 lead package for the XC3090. First production is scheduled for IQ 1989. Other versions for the rest of the 3000 family will follow with production early in 1989.

**PLASTIC QUAD FLAT PACK (PQFP)**

The Plastic Quad Flat Pack is an EIAJ standard package. The leads are gull-wing on four sides. It is for surface mount Commercial applications.

The initial versions will be 31 Mil (0.8 mm) pitch, 80 leads and 20 Mil (0.65 mm) pitch, 100 leads. First production is scheduled for 4Q 1988.

---

**FOR MORE INFORMATION ON SMT**

The following organizations provide SMT consulting and training, component part lists, and related services:

**D Brown Associates**  
(Surface Mounting Directory  
and SMT: How to Get Started)  
Box 43  
Warrington, PA 18976  
(215) 343-0123

**Electronics Manufacturing  
Productivity Facility (EMPF)**  
1417 North Norma Street  
Ridgecrest, CA 93555-2510  
(619) 446-7706

**International Quality  
Technologies, Inc.**  
4300 Stevens Creek Blvd  
Suite 203  
San Jose, CA 95129  
(408) 246-6071

**National Training Center**  
Northhampton Area College  
3835 Greenpond Rd  
Bethlehem, PA 18017  
(215) 861-5486

**Surface Mount Council**  
C/O IPC  
7380 N Lincoln Ave  
Lincolnwood, IL 60646  
(312) 677-2850

**1 Programmable Gate Arrays**

**2 Product Specifications**

**3 Quality, Testing, Packaging**

**4 *Technical Support***

**5 Development Systems**

**6 Applications**

**7 Article Reprints**

**8 Index**

---



## Technical Support

---

Technical Seminars and Users' Group Meetings .....	4-1
Video Tapes .....	4-2
Newsletter .....	4-3
Technical Bulletin Board .....	4-4
Field Application Engineers .....	4-6
User's Guide .....	4-7
XACT Manuals .....	4-8

---

Beyond the technical data in this book, Xilinx provides a wealth of additional technical information to LCA users. The following pages give an overview of the existing

material, beginning with Technical Seminars and ending with detailed Technical Manuals.

---



## Technical Seminars and Users' Group Meetings

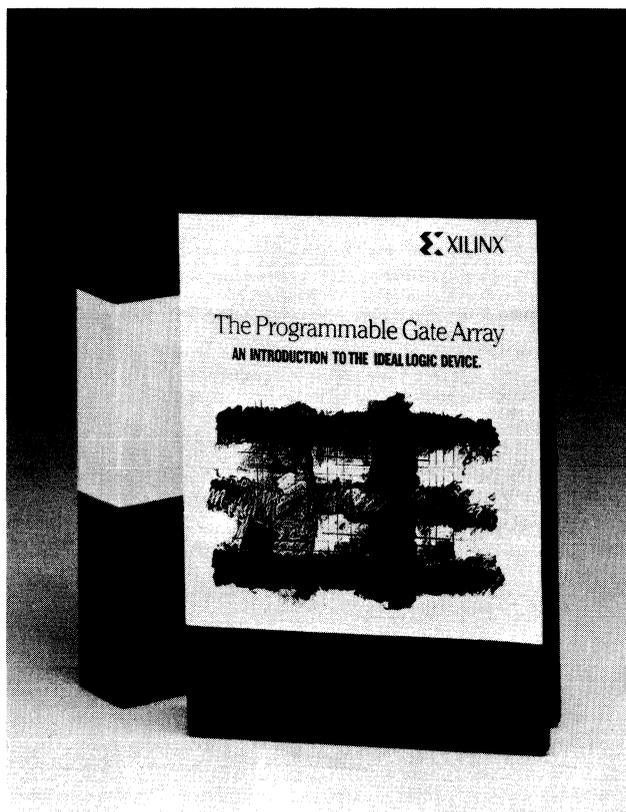
XILINX Seminars and Users' Group Meetings											
Tokyo	Vancouver	Minneapolis	Ottawa	Montreal	Manchester	Oslo	Helsinki				
Osaka	Seattle	Milwaukee	Toronto	Burlington	Nottingham	København	Stockholm				
Seoul	Portland	Chicago	Rochester	Boston	Reading	Dortmund	Malmö				
Taipei	Salt Lake City	Boulder	Detroit	Danbury	London	Frankfurt	Hamburg				
Hong Kong	Sunnyvale	Colorado Springs	Ann Arbor	Long Island	Amsterdam	Heidelberg	Berlin				
	Los Angeles	Phoenix	Cleveland	Holmdel	Bruxelles	Karlsruhe	Nürnberg				
	San Diego	Tucson	Indianapolis	Philadelphia	Paris	Stuttgart	München				
			Huntsville	Baltimore	Lannion	Zürich	Salzburg				
			Dallas	Atlanta	Rennes	Milano	Wien				
				Orlando	Grenoble	Torino	Tel Aviv				
				Tampa	Toulouse	Padova	Haifa				
					Madrid	Roma					

Xilinx sponsors technical seminars at locations throughout North America, Europe, and Asia.

Product-oriented seminars are directed toward new and potential users of Programmable Gate Arrays. These seminars include a basic description of the Logic Cell Array architecture and its benefits of this technology. Experienced users will also find these seminars useful for learning about newly-released products from Xilinx.

Users' Group meetings are intended for experienced users of Xilinx Programmable Gate Arrays, and emphasize the use of the various development system tools to generate LCA-based designs.

Contact your local Xilinx sales office, sales representative, or distributor for information about seminars in your area.

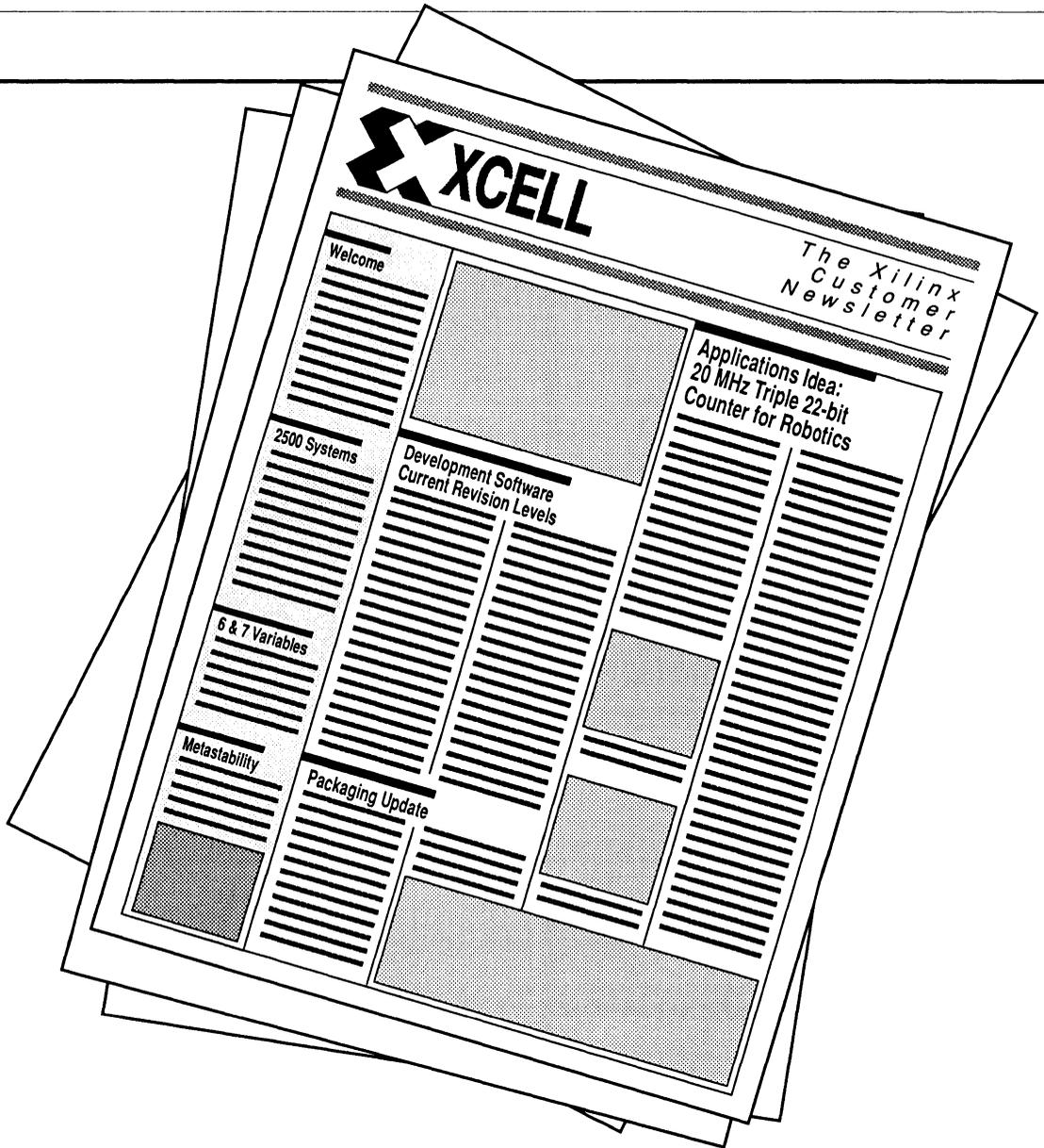


---

A one-hour videotape, entitled "Programmable Gate Arrays: The Ideal Logic Device," is available from Xilinx. The presentation is divided into three main sections. The first portion of the videotape is an overview of the Logic Cell Array architecture and the development system, including some example applications. The second section contains a description of the Xilinx product families, a more detailed description of the XC3000 series architecture, a description of the LCA configuration modes, and a brief discussion of programmable gate array performance in terms of

speed, density, and cost. Development systems and the design methodology are discussed in the last third of the presentation, including on-screen demonstrations of some of the software tools. Additional videotapes covering specific details are in preparation.

VHS copies are available in NTSC, PAL, and SECAM formats; contact your local Xilinx sales office, sales representative, or distributor.



---

In September '88 Xilinx started a quarterly technical newsletter to supply up-to-date information to registered Xilinx customers. This newsletter gives updates on hardware and software availability and revision levels. It also carries information on PC clone compatibility, software bugs and

work-arounds. Applications ideas and user tips and a list of relevant magazine articles make this a valuable source of information for the systems designer using Xilinx LCAs.



## M)SG-SECTION

Enter this section to send and receive messages.

### TYPE:

**E<CR>** [E)NTER]

To send a message.

**T<CR>** [T)O-YOU]

To read messages.

**L<CR>** [L)IST]

To list headers of all readable messages in the area. Some messages are private and can be read only by the addressee.

**K<CR>** [K)ILL]

To delete a message you've just read.

**R<CR>** [R)EPLY]

To reply to a message you've just read.



## F)ILE-SECTION

The file section is divided into several areas. Enter this section to read, upload and download files.

### TYPE:

**A<CR>** [A)REA]

To list the existing file areas.

After choosing a file area, you can now list all the files in this area.

### TYPE:

**F<CR>** [F)ILES]

To display the available files, the size of each file in bytes and a brief description of the contents of each file.

**D<CR>** [D)OWNLOAD]

To download one or more files from the bulletin board.

**L<CR>** [L)IST]

To locate a file in any accessible area

**T<CR>** [T)YPE]

To display a text file on the screen.

\* Files can be uploaded from any file area by typing:

**U<CR>** [U)PLOAD]



## B)ULLETIN

Enter this section to read the "latest and greatest" information on the Xilinx Bulletin Board. A list of bulletins is automatically displayed.

### TYPE:

**<BULLETIN#><CR>**

To display a specific bulletin on the screen.



## G)OODBYE

At this time you can leave a message for the system operator. See the M)SG-Section for instructions on how to send messages to other bulletin board users.

To provide customers with up-to-date information and an immediate response to questions, Xilinx provides a 24-hour electronic bulletin board. The Xilinx Technical Bulletin Board (XTBB) is available to all registered XACT

customers. Users with full privilege can read files on the bulletin board, download those of interest to their own systems or upload files to the XTBB. They can also leave messages for other XTBB users.

New bulletin board users must answer a questionnaire when they first access the XTBB. After answering the questionnaire callers can browse through the bulletin and general information file areas. Before exiting, they should leave a message for the system operator requesting full access. A caller with a valid XACT protection key will be given full user privileges within 24 hours.

The software and hardware requirements for accessing the Xilinx Technical Bulletin Board are:

Baud Rate	1200 or 2400
Character Format	8 data bits, no parity, 1 stop bit
Phone Number	(408) 559-9327
Transfer Protocols	ASCII, Kermit, Xmodem, - CRC, Telink

Information contained on the XTBB is divided into three general categories: 1. Bulletins, 2. Files and 3. Messages.

1. Bulletins contain tidbits of up-to-date information; they can be displayed on screen but cannot be downloaded.
2. Files can contain just about anything (text, user programs, etc.). XTBB users can download files to their own systems or upload files to the bulletin board.
3. Messages are used to communicate with other XTBB users; they can be general—available to everyone—or private.

The XTBB is based on a bulletin board system called FIDO. FIDO is a menu-driven system—you choose commands from menus to decide what happens next. To choose a menu command, simply type the first letter of the command and press return <CR>. Listed below are some helpful hints for using the XTBB.

- To perform a sequence of commands, type the first letter of each command, followed by a space, and press return. For example, typing **F A 1 F <CR>** [F]ile A]rea 1 F]iles] from the main menu will list all of the files contained in file area 1.
- Often the user is asked a question and prompted to choose between two options (e.g. [yes NO]). The option displayed in all capital letters is the default choice. To select this option, simply press return. Otherwies, type your choice and hit return.
- The XTBB has an extensive help section. To get help, type **?<CR>**. If you have questions about a specific command, type the first letter of the command followed by a question mark and a carriage return (e.g. **F?<CR>**). A short explanation of the command will be displayed.
- For more information, read the XTBBHLP.TXT file located in the GENINFO file area (file area 1).



## Field Applications Engineers

---

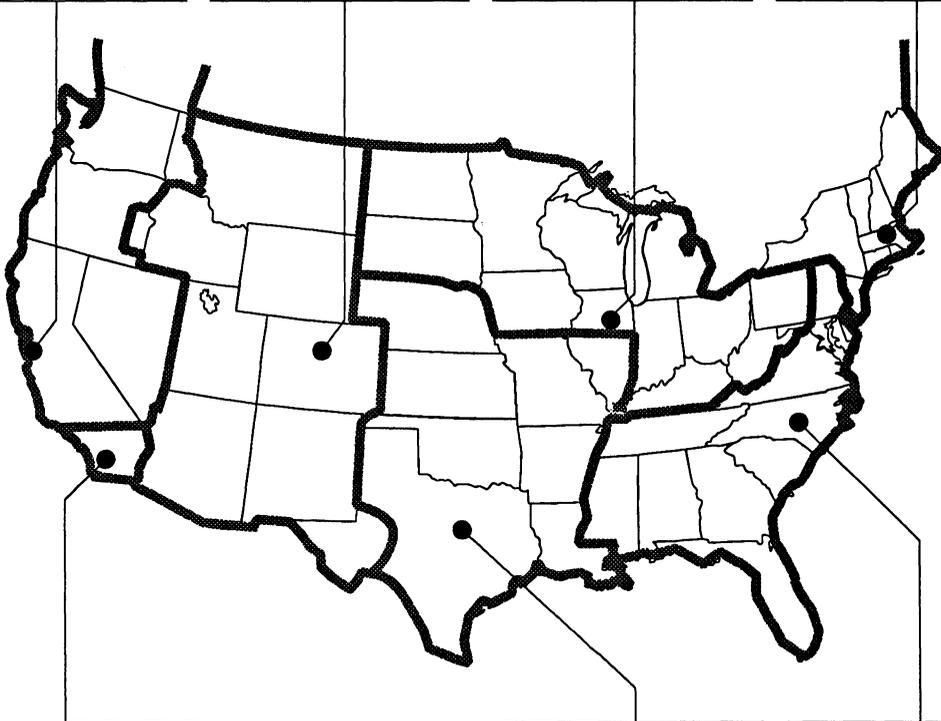
1270 Oakmead Pkwy.  
Suite 201  
Sunnyvale, CA 94086  
Tel: 408-245-1361

3100 Arapahoe Rd.  
Suite 404  
Boulder, CO 80303  
Tel: 303-443-4780

919 N. Plum Grove Rd.  
Suite A  
Schaumburg, IL 60173  
Tel: 312-490-1972

20 Mall Rd.  
Suite 469  
Burlington, MA 01803  
Tel: 617-229-7799

---



2081 Business Center Dr.  
Suite 180  
Irvine, CA 92715  
Tel: 714-955-0831

14506 Sandyside Dr.  
Austin, TX 78728  
Tel: 512-251-7148

10704 Spiralwood Ct.  
Raleigh, NC 27612  
Tel: 919-846-3922

---

### North America

There are 7 Xilinx Field Applications Engineers in the locations shown above. Additional technical support is provided by Headquarters Applications. Outside California dial 1-800-255-7778

The world-wide network of Xilinx Representatives and Distributors also gives technical support.

### Europe

The Xilinx European sales office in England has a resident Field Applications Engineer (tel 44-73081-6725).

### Japan

Xilinx Japan is located in Tokyo and expects to have a Field Applications Engineer in late 1988.

---



---

The Xilinx User's Guide is a binder with several self-contained notes giving practical and tutorial information on the following subjects:

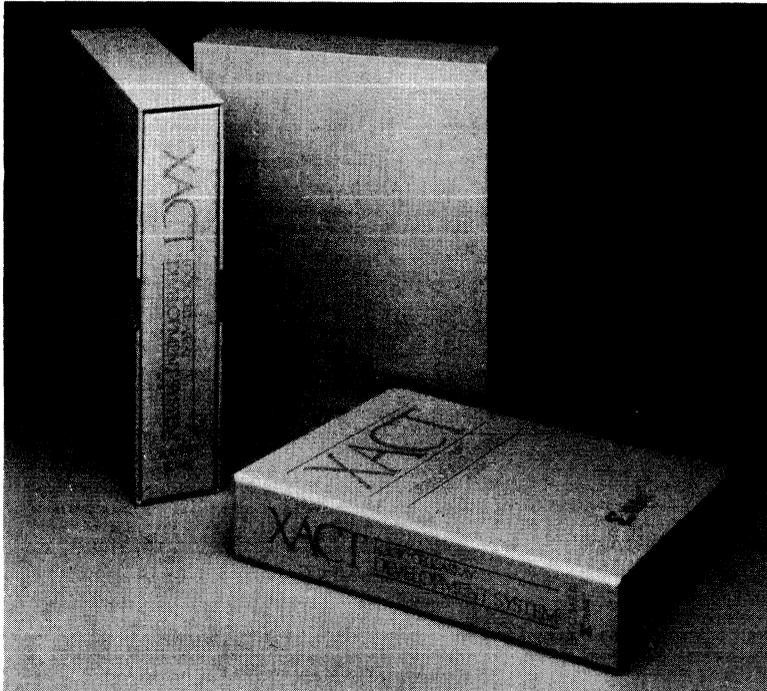
- Getting Started
- Design Entry

- Design Implementation
- Design Verification

The Xilinx User's Guide is included with every system.

This series of notes is being expanded.

---



---

The first two binders of this 3-volume set are the LCA Development System Manuals, providing exhaustive reference information on:

- Executive Program
- LCA Editor
- Macros
- Simulator (SILOS)
- PROM Formatter
- Bit -Stream Generator

- Demo Board
- Place and Route
- XNF to LCA  
and information on schematic capture

The third volume provides detailed information on each of the 2000 and 3000 series XACT macros, including schematics, block count, and examples of typical placements.

**1 Programmable Gate Arrays**

**2 Product Specifications**

**3 Quality, Testing, Packaging**

**4 Technical Support**

**5 *Development Systems***

**6 Applications**

**7 Article Reprints**

**8 Index**

---

PC Hardware Requirements .....	5-1
Development Overview .....	5-4
Design Flow .....	5-9
High Level Design Entry .....	5-10
Schematic Capture .....	5-10
PLD Conversion .....	5-13
Design Implementation .....	5-16
Automatic Place and Route .....	5-17
XACT Design Editor .....	5-19
Design Verification .....	5-21
Macro Library .....	5-25
Product Briefs .....	5-32
XACT Design Editor .....	5-32
PC-SILOS Simulator .....	5-33
Automated Design Implementation .....	5-34
XACTOR In-Circuit Design Verifier .....	5-36
Schematic Entry Interfaces .....	5-37
FutureNet DASH .....	5-38
Schema II .....	5-39
Daisy .....	5-40
Mentor .....	5-42
OrCAD .....	5-43
7400 TTL Library .....	5-44
Complete Development Systems .....	5-45
Configuration PROM Programmer .....	5-46
Ordering Information .....	5-47
Options and Systems Requirements .....	5-48

---



# Development System Hardware Requirements

Xilinx provides an integrated Development System for design and implementation of Logic Cell Arrays. The LCA development system operates on an IBM® PC/AT™ or IBM PS/2 model 60 or 80 and provides a range of support features. This provides the user with an effective, convenient, low risk method of logic design entry, simulation, LCA generation and verification for single chip logic designs of up to 9000 gates. Several popular workstation suppliers have developed compatible software which allows logic diagram design entry, simulation and LCA partitioning programs to be run on their systems.

The system configuration needed to run the Xilinx software consists of a 80286 or 80386 based IBM PC or "compatible". The software runs under the MS-DOS operating system version 3.0 and later, requires a 20 Megabyte hard disk, a mouse and a color monitor. A math co-processor can enhance performance of Automatic Placement and Routing by 10 or 20%.

The recommended configuration consists of:

- A "386" PC/AT or PS/2 model 80
- 40 M byte hard disk drive plus a 1.2 M Byte high density floppy disk drive
- Two RS-232-C serial ports
- One parallel port
- Enhanced Graphics Adapter and Display (EGA and EGD)
- Mouse
- MS-DOS version 3.0 or higher
- IBM compatible BIOS and keyboard

Highest system performance can be obtained by using a 80386 based system or a PS/2 model 80. To assure integrity, all Xilinx software is tested on IBM systems and several of the more "compatible" systems before release. Xilinx LCA development software includes some of the first DOS based programs to make extensive use of the "protected" mode of the processor. This has exposed protected mode IBM-incompatibilities of some "clones", usu-

ally in BIOS or Keyboard Controller. Xilinx software includes system exercises called PMTEST and PMINFO to help test IBM compatibility and measure relative performance.

## MEMORY

In protected mode the processor uses extended address space, which is found above the 1 Meg address. Earlier versions of the Xilinx XACT Design Editor used expanded memory (EMS or LIM standard) which is not suitable for the higher performance required by the larger array chips. Some of the Xilinx programs run in conventional memory space while the larger programs have a loader program of about 100 K bytes, which resides in conventional base memory and installs the Xilinx program in extended address space. See Figure 1. If the extended memory space is limited and becomes filled, the balance of the program will be "backfilled" above the loader in conventional memory. The availability of sufficient extended memory for an entire Xilinx protected mode program makes it possible to invoke nonprotected mode programs such as text editors or utilities without terminating a Xilinx program. For different LCA designs, the XACT Design Editor has different minimal requirements of available memory, as displayed by the BIOS at power-up.

LCA Gates	Total Memory Required for XACT 2.1
-----------	------------------------------------

2000 or less	2048K Bytes
3000	2816K Bytes
4200	3584K Bytes
6400	4864K Bytes
90006	6144K Bytes

Note that this amount of memory must be available to XACT, i.e., it does not include the memory used for other resident programs.

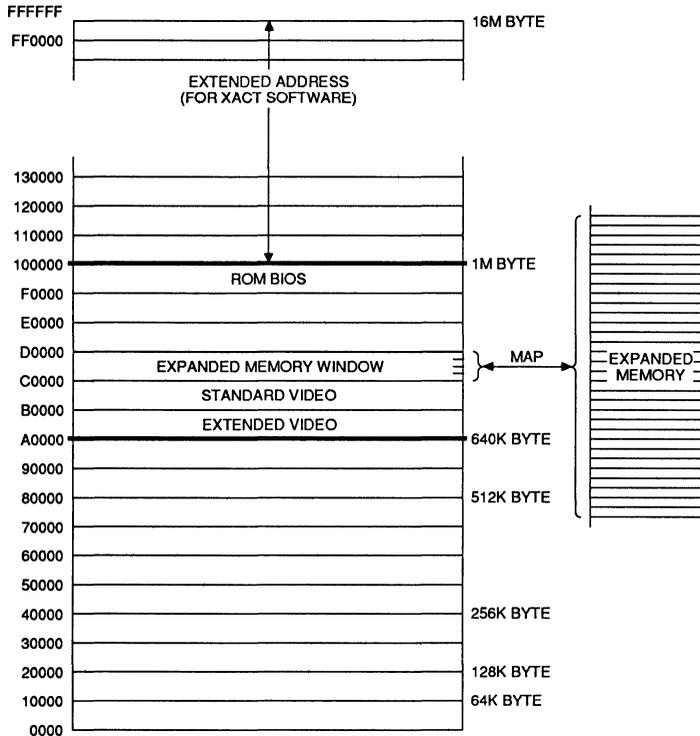
Note also that the Compaq 386 has only 640K bytes of its first Megabyte available to any user.

Disk requirements for Xilinx software are a function of the program options which the user chooses, the number and

**Development System Hardware Requirements**

type of active LCA designs and the number of other user software packages. A high density 1.2 M byte 5.25" floppy drive is needed for installation of the Xilinx software on the hard disk. The Xilinx software is also available on PS/2 3.5" diskettes. Some of the larger Xilinx programs are:

PROGRAM	APROX. SIZE
XACT	3.5 M byte
PIN2XNF	1.5 M byte
APR	1.0 M byte



1164 01

**Figure 1. IBM Memory Map**

The Compaq 386 reserves the 384K bytes of memory between 640K and 1M for internal use, making them unavailable to XACT.

## I/O Ports

The Xilinx LCA Development System requires several I/O ports. A parallel port is needed for the software execution protection key. The key must be in place to allow Xilinx software to execute but is virtually transparent, and the port can be used simultaneously for parallel printer or the Xilinx download cable. Several printer types are supported for text or graphic hard copy. Serial COM ports are used for a mouse, the XACTOR In-Circuit Design Verifier and the Configuration PROM Programmer.

## Display

The original IBM PC provided for two display options. Monochrome Display for text, and four color, 320x200 pixel, 8x8 dot character, Color Graphics Display for text and graphics. The interface to the displays was done by the Monochrome Display Adapter (MDA) and Color Display Adapter (CGA). IBM has added the higher resolution 16 color, 640x350 pixel, 8x14 dot character, Enhanced Graphic Display (EGD) and the corresponding Enhanced Graphic Adaptor (EGA). The Xilinx LCA Development System operates with a CGA or EGA display, but, future software enhancements might not support CGA and the higher resolution EGA is therefore recommended. The use of the EGA display is necessary for FutureNet schematic capture. The PS/2 compatible VGA is preferred for the higher density 3000 family LCAs. The Hercules monochrome display is not compatible.

## Mouse

The Xilinx Development System programs are compatible with several varieties of mice. These include Mouse Systems PC Mouse (no device driver required), Microsoft (serial or parallel), LogiTech C7 and the FutureNet mouse.

The Xilinx software will directly support any mouse which emulates the PC Mouse or has a device driver which provides Microsoft compatibility and defines it's COM port.

## PC Setup

When the system is powered up it uses commands from the DOS CONFIG.SYS file to install selected device driver programs (such as Mouse driver) in memory and define buffer and file sizes. Examples of these statements are:

```
device=c:\lib\msmouse.sys /1
files=10
buffers=20
```

After CONFIG.SYS functions are implemented the system executes the commands found in the AUTOEXEC.BAT file. This file contains DOS commands such as:

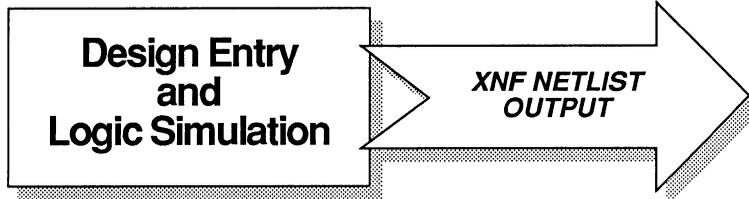
```
path=c:\; ... c:\xact;c:\schema; ...
set xact=c:\xact
set grmode=ega
set swmode=9
set minbytes=65000
```

The first line shows the portion of the path established by the XACT and SCHEMA installation procedures. These are the default directories created and used in the Xilinx installation procedures. The SET SWMODE= sets a parameter defining one of several alternative ways of switching the processor from protected to real mode. Several alternatives are made available in order to accommodate various "clone" idiosyncrasies. Possible values are 9 (default), 10, 7, 4, and for 80386 based systems, 3.

See the Xilinx installation instructions and PC manuals for additional information.

## Logic Cell Array Design Flow

### STEP 1



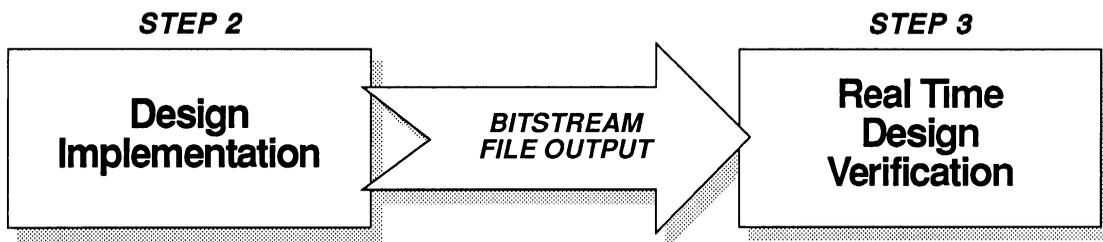
- ❑ Open development system supports design entry and simulation on popular CAE systems
- ❑ Interfaces available for PC and workstation-based environments:
  - ❑ FutureNet, Schema, CASE, VIEWlogic, PCAD, OrCAD, etc.
  - ❑ Daisy, Mentor, Valid
  - ❑ SILOS, CADAT simulators
- ❑ Standard macro library includes over 100 elements
- ❑ PALASM™ compatible Boolean equation input can be merged with schematic input
- ❑ Logic simulation can be used to verify design function



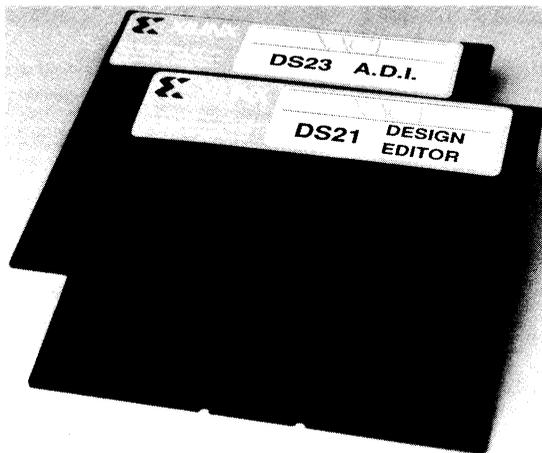
Schematic entry on your PC/workstation

---

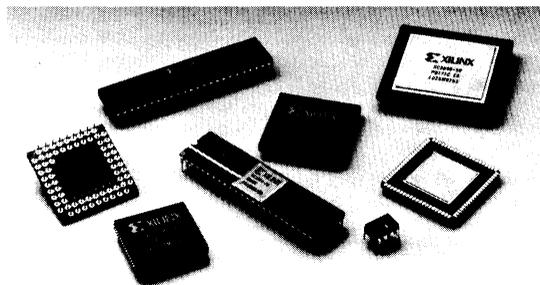
## Logic Cell Array Design Flow



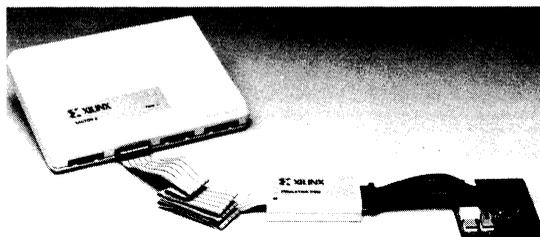
- ❑ Complete system translates design into programmable gate arrays
- ❑ Partitions gate level design logic into Logic Cell Array architecture (CLB/IOB)
- ❑ Automatic logic reduction and partitioning removes unused logic (e.g. unused counter outputs)
- ❑ Logic synthesis software optimizes design for Logic Cell Array architecture
- ❑ Programs run on IBM™ PC/AT™ or compatible personal computer
- ❑ LCA user-programmability permits real time, in-circuit debugging
- ❑ Download cable allows LCA to be programmed in-circuit from designer's PC
- ❑ XC-DS28 XACTOR In-Circuit Design Verifier reads and displays the internal LCA storage element states



Automated design implementation software



Immediate production

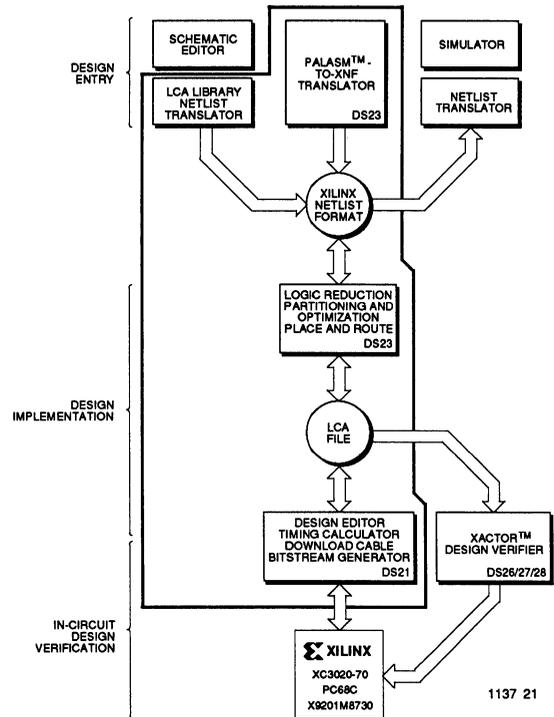
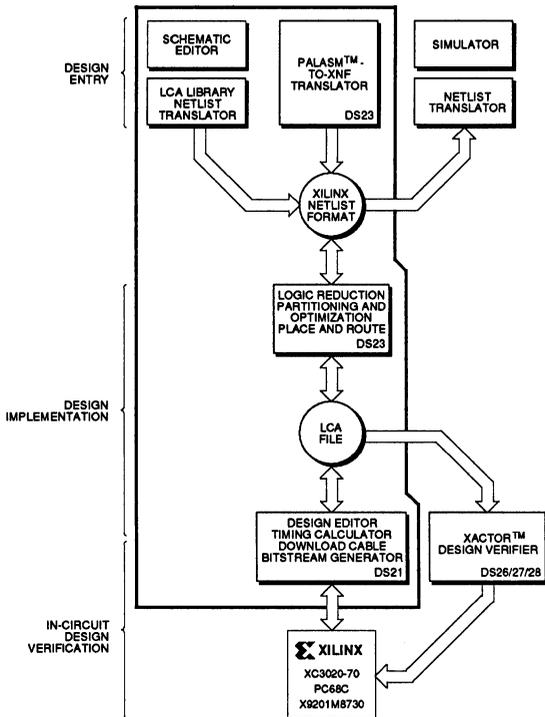


Design changes are made in minutes

## XACT Development System Order Guide

- **USER DOES NOT HAVE A SCHEMATIC EDITOR.**  
Xilinx provides complete system using FutureNet DASH-LCA.

- **USER ALREADY HAS A SCHEMATIC EDITOR:**
  - FutureNet DASH
  - Daisy ACE or DED II
  - Mentor IDEA
  - Schema II
  - OrCAD SDT



1137 21

**PURCHASE:**

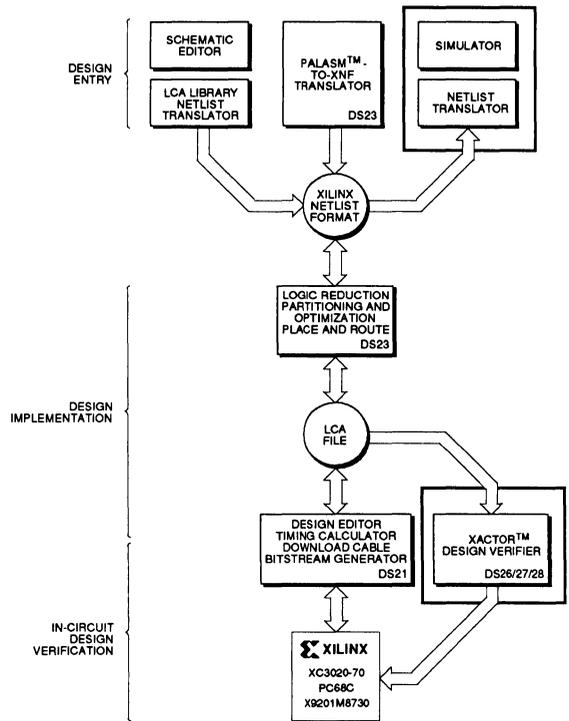
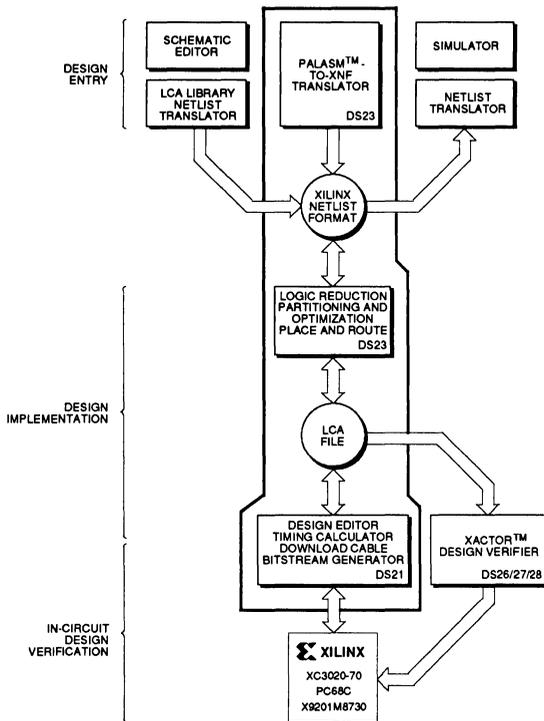
DS53 (= FutureNet DASH-LCA editor + DS23 + DS21)  
with optional  
DS40 TTL Library for DASH-LCA

**PURCHASE:**

DS31 FutureNet DASH library and translator  
or  
DS32 Schema II library and translator  
or  
DS33 Daisy library and translator  
or  
DS34 Mentor library and translator  
or  
DS35 OrCAD SDT library and translator  
and  
DS23 A.D.I.  
and  
DS21 XACT Design Editor

## XACT Development System Order Guide

- **USER ALREADY HAS A SCHEMATIC EDITOR:**
  - Valid
  - CASE
  - PCAD
  - VIEWlogic
  - Any editor with Xilinx XNF output
- **ADD SIMULATION AND/OR IN-CIRCUIT DEBUGGING CABABILITY.**



1137 22

**PURCHASE:**

Library and translator from vendor  
and  
DS23 A.D.I.  
and  
DS21 XACT Design Editor

**PURCHASE:**

DS22 PC-SILOS™ Simulator  
or  
Any simulator with Xilinx XNF interface  
and/or  
DS26/27/28 XACTOR™



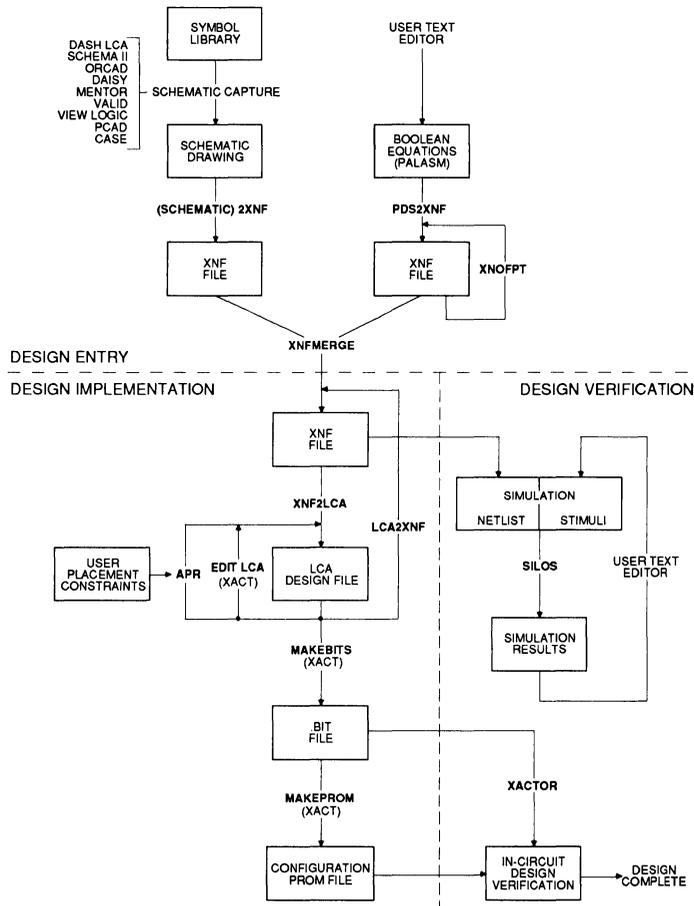


# Xilinx Programmable Gate Arrays Design Flow

This section provides an overview of the Xilinx Programmable Gate Array design process and the Xilinx Development System software. For more extensive coverage of specific features and development system commands, refer to the appropriate sections of the various Xilinx technical manuals. It is assumed that the reader is familiar with the Logic Cell Array (LCA) architecture, and DOS operating system commands.

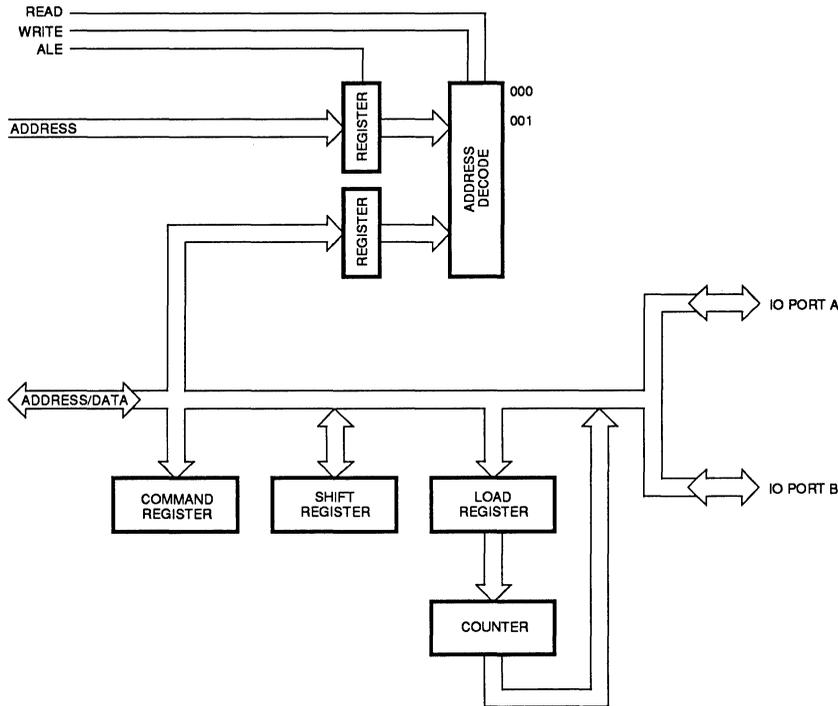
The Xilinx Programmable Gate Array Development System Software runs on an IBM PC/AT or PS/2 model 60 or

80, or on true IBM compatibles. It uses protected mode extended addressing, requires (depending on LCA device type) 2 to 6 M Bytes of RAM, a mouse, MSDOS 3.0 or higher, two serial ports, a parallel port, 20 M Byte (preferably 40 M) hard disk and an EGA or VGA display. The software consists of several packages which provide an integrated design entry implementation and verification capability. Schematic Capture, Logic Synthesis, Automated Design Implementation (partition, place and route), Simulation and In-Circuit Design Verification are available to augment the XACT Design Editor. See Figure 1.



1115 01

Figure 1. Xilinx Development System Design Entry, Design Implementation, and Verification Flowchart



1115 02 NEW

Figure 2. Sample of a Microprocessor Interface Implemented in an XC3020

A simple microprocessor peripheral design example illustrates some of the basics of the Programmable Gate Array design process. See Figure 2. It uses a sixteen bit address bus multiplexed with eight bits of data. The address is stored and decoded to control read/write functions of the data bus. A writable command register provides count control of a loadable eight bit counter, data and shift control of a bidirectional shift register and direction control of two I/O ports. Relevant files for the design flow of this example are included in the Xilinx software for the user to review.

### HIGH LEVEL DESIGN ENTRY

Xilinx has established the Xilinx Netlist File (XNF) as an intermediate design description to standardize the XACT interface for design entry and verification software. Xilinx and others continue to add software to enhance the design entry and verification process. Each of these programs use the intermediate XNF interface language to convert to an XACT *name.lca* file. The conversion translates the network description automatically into partitioned Configurable Logic Block (CLB) and I/O block (IOB) functions. Connectivity is defined in terms of block-pin net-names but without routing determined. The resulting block placement is arbitrary. Automatic Place and Route (APR) or Editica in XACT are needed to implement an orderly block placement and interconnect net routing.

FutureNet, Schema II, Daisy and other schematic capture design entries are available. A PLD conversion utility provides an alternate method of design entry and optimization. An XNF file can be generated by a PC/PS2 based schematic capture system or produced on another engineering work station and ported to a system with XACT, for conversion to an LCA file.

### Schematic Capture

Schematic capture provides one method of design entry. Xilinx offers schematic entry, symbol libraries and XNF translators for FutureNet, Schema II and Daisy programs. Other schematic capture and XNF translator packages are available from CADAT, Valid, Viewlogic and others. Schematic capture provides a simple, quick, automatic and familiar design methodology. An additional "74XXX" library is available in DASH-LCA (FutureNet) to supplement the basic Xilinx macros.

The Xilinx DASH-LCA version of FutureNet schematic capture is invoked with the command *fn 3* (*fn 2* for the X2000 family). The first command issued in FutureNet is *lib \xact\3000 (or x2000)* to define the symbol library to be used. FutureNet functions are implemented with a combination of keyboard/menu commands and mouse button operations. A full description of commands is included in the Xilinx FutureNet documentation. Similar detailed

documentation is provided for users who choose schematic capture alternatives such as Schema etc. Some of the FutureNet command groups and example keyboard commands are shown in Table 1.

For a full description of commands and capabilities read the FutureNet documentation and the Xilinx Logic Cell Array Development System documentation.

After the schematic capture program is invoked and a symbol library is selected for the intended LCA device family to be used, Xilinx library symbols are loaded, positioned and interconnected in the drawing. See Figure 3. A key step is to name the nets (with attribute 5) as well as to assign desired IOB package pins and assign desired CLB and package pin locations for "primitive" latch/flip-flop symbols (with attribute 80). Similar location attributes can be assigned to primitive symbols in SCHEMA and other LCA schematic capture packages. Unnamed nets will be assigned default names. For simulation and timing analysis, these are usually less meaningful than user mnemonics. Constraint flag symbols can be attached to selected nets to generate schematic constraints which are passed through the XNF to LCA conversion to the LCA automatic placement and routing process (APR). These constraints can provide guidance to the placement software. Artificial logic can be added to reserve LCA IOB and CLB locations

and provide placement constraints, such as assigning unused IOBs to prevent the APR program from using configuration pins for I/O. Some of the flag types identify critical nets, longline routing and external nets (nodes not to be absorbed into a block). Some users find it more convenient to assign placement and routing constraints in a user generated constraint text file *name.cst* used by the Xilinx Automatic Placement and Routing (APR) program.

The schematic capture allows hierarchical drawings, but pad symbols must be used only in the top level drawing. IOBs are composed from appropriate pad symbol, input buffer, input flip-flop, output buffer etc. Direct CMOS input pins of an XC3000 device are indicated in schematic capture by connecting an input pad symbol directly to a global buffer (GCLK or ACLK) without an pad input buffer. Pin numbers for these inputs are predetermined and no package pad number is needed. The crystal oscillator amplifier (GXTL) connections are also predetermined package pins and, in addition, should not have the related IOBs configured.

The signal names VCC and GND, used for disabling an unused input, will result in gate minimization in the xnf2lca process. Extraneous inverters will be eliminated by use of input inversion on destinations unless the xnf2lca -k option is used.

Command Group	Command Examples
<b>executive</b>	LOAD name, SAVE name and quit
<b>area editing</b>	[m (move area), [c (copy area) and [erase (erase an area)
<b>line editing</b>	/l (draw line), /es (erase segment), /el (erase line), /en (erase net)
<b>symbol</b>	.l symbol (load library symbol), * symbol (load tag and drag symbol), .m (move), .c (copy), .e (erase)

Some of the mouse button functions are:

**mode = menu**

<b>left</b>	- draw, tag or drag
<b>center</b>	- execute command line
<b>right</b>	- menu screen

**mode = move, copy, erase**

<b>left</b>	- execute mode status field
<b>center</b>	- tag or drag
<b>right</b>	- return to base mode

**Table 1. Sample DASH-LCA (FutureNet) Commands**



The commands *dcm name.dwg* and *pinc name.dwg* must be issued for the top level drawing and each lower level drawing within it. (Not necessary for Xilinx library drawings.) This produces a *name.pin* file which is the FutureNet netlist-like design description. See Figure 4. The command *pin2xnf name* will then translate the .pin file into a Xilinx Netlist File (XNF file) of the design specified by the schematic capture. This can be merged with other portions of a design and then partitioned into LCA elements in a *name.lca* file for further processing by automatic place and route or by XACT-editlca.

### PLD Conversion (Logic Synthesis/Equations)

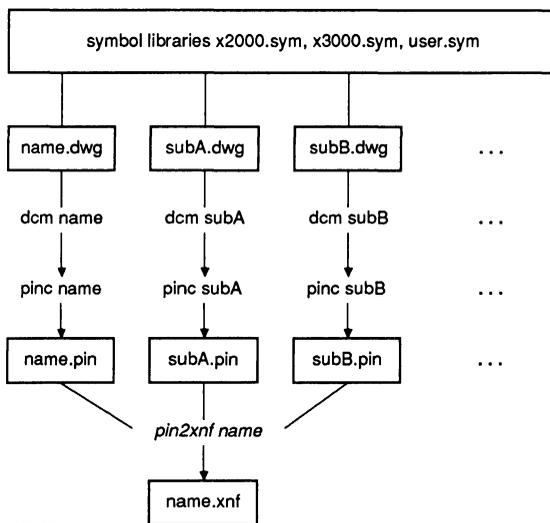
In addition to schematic capture design entry, Xilinx provides four programs that allow for the description of designs or portions of designs using Boolean equations. These four programs are PDS2XNF, XNFOPT, XNFMERGE, and XNF2LCA. PDS2XNF converts a PALASM design file into an XNF file. The PDS2XNF program supports over fifty programmable logic devices (PLDs), but it is not restricted to these. The designer can create logic equations in PALASM design files specifically for the LCA without any regard for a particular PLD type.

There are three basic design flows using these tools. The first method is to incorporate a PLD design into a schematic. In this case, a schematic is drawn that contains PLD symbols. Each of these PLD symbols contain a special parameter that references an XNF file that is translated from a PALASM design file using the PDS2XNF program. The output of the PDS2XNF program does not efficiently translate into an LCA file, so it is necessary to optimize the

design for the LCA architecture using the XNFOPT program. When the schematic is complete, it is converted into an XNF file using a schematic netlist-to-XNF translator like SCH2XNF for Schema II and PIN2XNF for FutureNet. This schematic XNF file is then merged with other XNF files into one XNF file using the XNFMERGE program. See Figure 5.

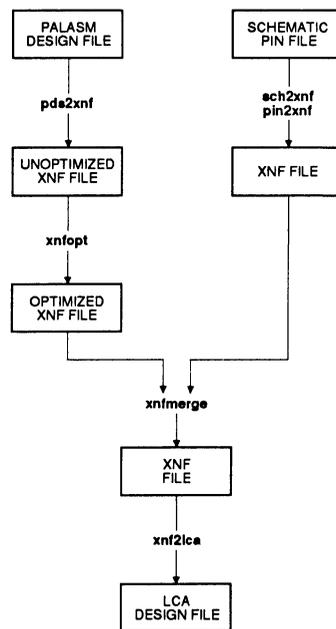
The second method is to translate an existing PALASM design file directly into a stand-alone LCA. The Logic Cell Array has more logic available than a PLD, but one can simply translate a PLD design directly into an LCA. First the design is translated into an XNF file using the PDS2XNF program, and then optimized using the XNFOPT program. The optimized design is translated into an LCA file using the XNF2LCA program.

The third method is to completely create an LCA design using PALASM equations. There is no need to specify a particular PLD part type in this case. You can use the USER PAL type in the input PALASM design file to tell PDS2XNF to translate the design without any regard for a particular PLD type. Once the PALASM design file is complete, it is translated to an XNF file using the PDS2XNF program. This XNF file must then be optimized for the LCA architecture using the XNFOPT program. Once optimized, the design is translated into an LCA file using XNF2LCA.



1115 10

Figure 4. Design Flow from Schematic Capture to Xilinx Netlist File with pin2xnf



1115 11

Figure 5. Design Flow for PLDs Included in a Schematic Diagram with xnfmerge

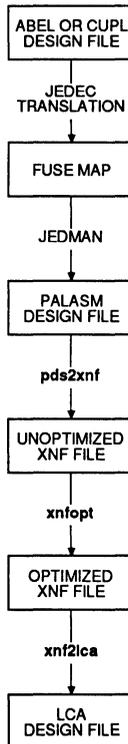
State machine descriptions and PLD equations written in other languages such as ABEL can be accommodated by compiling to a JEDEC file and disassembling the JEDEC file into a PALASM file with the JEDMAN program if a compatible PLD type exists. See Figure 6.

Figure 7 shows the PALASM design file of equations for a decoder.

For a full list of commands and capabilities refer to the Xilinx Logic Cell Array Development System documentation. After the PALASM design file is created, PDS2XNF is used to translate the design into an XNF file. The conversion is produced by a command, with sample options, of the form:

```
pds2xnf -p -i input.(xnf) output.(xnf)
```

Uppercase/lowercase is treated the same for options and depends upon the operating system for file names. The file extensions in ( ) are the assumed defaults. Leading path extensions can be included to indicate other than current directory. See Figure 8.



1115 12

Figure 6. Design Flow for a Non-PALASM PLD to an LCA Design

```

TITLE          7SEG.PDS
AUTHOR        BART REYNOLDS AND THOMAS WAUGH
COMPANY       XILINK
DATE          APRIL 8, 1988

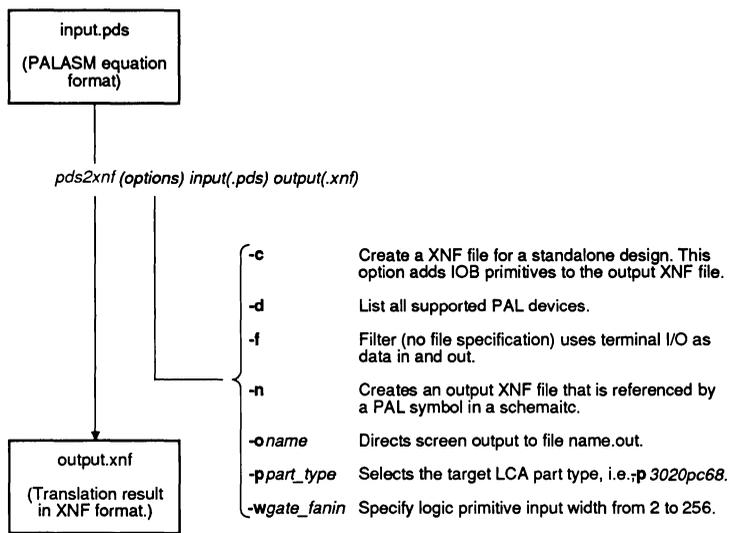
CHIP          7SEG   PAL10H8

;Input Pins   1  2  3  4  5  6  7  8  9 10
              NC NC D0 D1 D2 D3 NC NC NC NC
;Output Pins  11 12 13 14 15 16 17 18 19 20
              NC NC G  F  E  D  C  B  A  NC
;
;Input combinations
;
STRING ZERO   \D3 * /D2 * /D1 * /D0'
STRING ONE    \D3 * /D2 * /D1 * D0'
STRING TWO    \D3 * /D2 * D1 * /D0'
STRING THREE  \D3 * /D2 * D1 * D0'
STRING FOUR   \D3 * D2 * /D1 * /D0'
STRING FIVE   \D3 * D2 * /D1 * D0'
STRING SIX    \D3 * D2 * D1 * /D0'
STRING SEVEN  \D3 * D2 * D1 * D0'
STRING EIGHT  \ D3 * /D2 * /D1 * /D0'
STRING NINE   \ D3 * /D2 * /D1 * D0'
STRING TEN    \ D3 * /D2 * D1 * /D0'
STRING ELEVEN \ D3 * /D2 * D1 * D0'
STRING TWELVE \ D3 * D2 * /D1 * /D0'
STRING THIRTEEN \ D3 * D2 * /D1 * D0'
STRING FOURTEEN \ D3 * D2 * D1 * /D0'
STRING FIFTEEN \ D3 * D2 * D1 * D0'
  
```

```

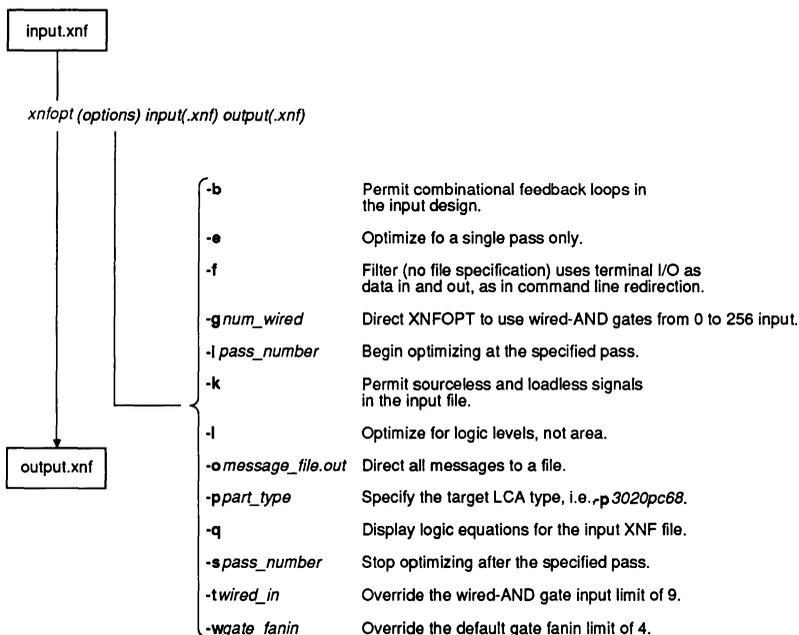
EQUATIONS
A = ZERO + TWO + THREE + FIVE + SIX + SEVEN
  + EIGHT + NINE + TEN + TWELVE + FOURTEEN
  + FIFTEEN
B = ZERO + ONE + TWO + THREE + FOUR + SEVEN
  + EIGHT + NINE + TEN + THIRTEEN
C = ZERO + ONE + THREE + FOUR + FIVE + SIX
  + SEVEN + EIGHT + NINE + TEN + ELEVEN
  + THIRTEEN
D = ZERO + TWO + THREE + FIVE + SIX + EIGHT
  + ELEVEN + TWELVE + THIRTEEN + FOURTEEN
E = ZERO + TWO + SIX + EIGHT + TEN + ELEVEN
  + TWELVE + THIRTEEN + FOURTEEN + FIFTEEN
F = ZERO + FOUR + FIVE + SIX + EIGHT + NINE
  + TEN + ELEVEN + TWELVE + FOURTEEN
  + FIFTEEN
G = TWO + THREE + FOUR + FIVE + SIX + EIGHT
  + NINE + TEN + ELEVEN + THIRTEEN
  + FOURTEEN + FIFTEEN
  
```

Figure 7. Equation Definition of a PLD Seven Segment Decoder Design



1115 13

**Figure 8. PDS2XNF Translates a PALASM Based Design Into a Xilinx Netlist File**



1115 14

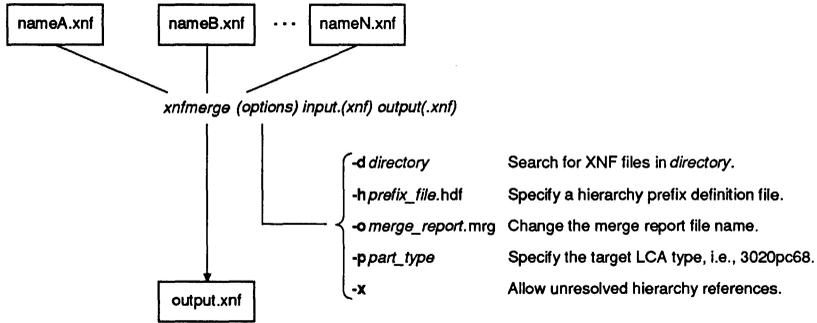
**Figure 9. XNFOPT Optimizes a Xilinx Netlist File for Programmable Gate Array Architecture**

DESIGN IMPLEMENTATION

Xilinx Netlist File and Optimization

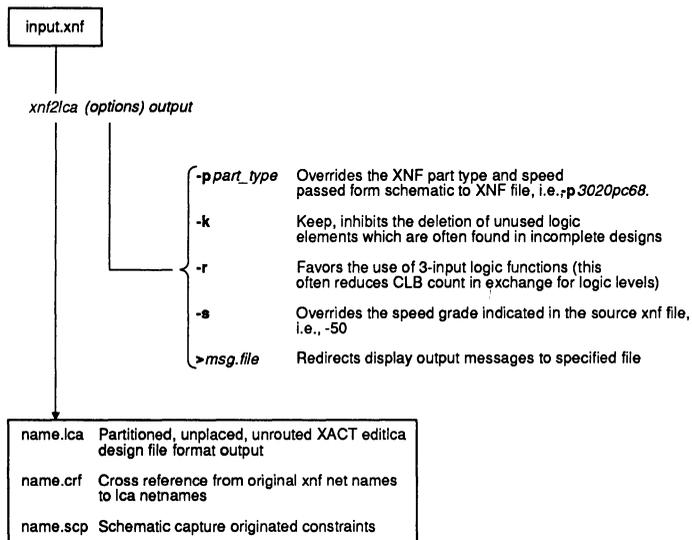
The XNFOPT program is used to optimize an XNF file for the Xilinx LCA architecture. It will not optimize logic that is

already expressed in CLB primitives. Although the XNFOPT program can be used to optimize any XNF file, it is most often used to optimize logic equations generated by the PDS2XNF program. In its default operation, XNFOPT runs continually, searching for better and better results, until it is stopped by entering a CTRL-BREAK on



1115 15

Figure 10. XNFMERGE Combines Several Partial Designs Into a Single Xilinx Netlist File



1115 16

Figure 11. XNF2LCA Translates a Xilinx Netlist File Into a Partitioned PGA Design File (.lca) and Associate Report and Constraint Files

the PC. The conversion is produced by a command, with sample options, of the form:

```
xnfopt -l -p3020pc68 -b input.(xnf) output.(xnf)
```

See Figure 9.

For more information on the use of these commands see the XACT Development System Reference Manual, Volume II, Chapter 8-Logic Synthesis, as well as the User's Guide section on Designing Logic Cell Arrays with Boolean Equations.

When incorporating PLD designs into schematics, the XNFMERGE program is needed to merge the different XNF files created from PALASM design files into the schematic's XNF file. In general, the XNFMERGE program is used to combine hierarchical lower-level logic element XNF files, into a top level XNF file. The conversion is produced by a command, with sample options, of the form:

```
xnfmmerge -d design -p3020pc68 inputA inputB output
```

See Figure 10.

For more information on the use of these commands see the XACT Developments System Reference Manual, Volume II, Chapter 5-XNF/LCA Interface, as well as the User's Guide section on Designing Logic Cell Arrays with Boolean Equations.

Once a single, flattened XNF file is produced for the design using the XNFMERGE program, the design is ready to be converted into an LCA file with the XNF2LCA program. Refer to the Logic Cell Array Editor section of the XACT LCA Development System binders for a complete list and descriptions of *xnf2lca* and *lca2xnf* commands and options. The conversion is produced by a command, with sample options, of the form:

```
xnf2lca -p3020pg68 -k -50 -r name
```

See Figure 11.

Figure 12 shows example sections of a design in the Xilinx Netlist File format. (XNF)

### Automatic Design Implementation (APR)

Automatic Placement and Routing is a program which is run on an LCA design file (*name.lca*) in order to automatically arrange the CLBs and IOBs, optimize their block-pin / net assignments and determine interconnection patterns to route the design and write the resulting design file to disk. The original design file can be the result of XACT-editlca or from an XNF2LCA conversion from schematic capture or logic synthesis. The program is analogous to

annealing (cooling) metal. The model begins with an elevated temperature and mobile blocks. As time proceeds the model "cools" and the blocks become more resistant to rearrangement, until the temperature reaches its final value. The program uses the input design file (*name.lca*), a file of constraints produced from parameters assigned in schematic capture (*name.scp*) and a text file of constraints produced by the user (*name.csf*). The primary results are a new design file (*output.lca*) and a report file of routing results, routing order, flag tables, delay

```
LCANET, 1
PROG, PIN2XNF, 1.80, 13:59:09 MAR 7, 1988
PART, 3020PC68-40
SYM, 25-NAND-38, NAND
PIN, 0, 0, PBEEN
PIN, 4, I, A2
PIN, 3, I, A1,, INV
PIN, 2, I, A0,, INV
PIN, 1, I, 25-1-075023
END
SYM, 25-O5-, NAND
PIN, 0, 0, 25-O5-
.
.
.
.
.
.
.
SYM, 6-CLB-1, CLB
PIN, X, 0, Sbus<0>
PIN, K, I, CWR
PIN, EC, I, 5-SHFTEN
PIN, E, I, S1
PIN, D, I, Sbus<1>
PIN, C, I, S0
PIN, B, I, DATA<0>
PIN, A, I, IOR0
CFG, BASE FGM
CFG, EQUATE G=~C*A+C*B
CFG, EQUATE F=~C*QX+C*D
CFG, CONFIG X:QX Y: DX:M DY: CLK:K RSTDIR: ENCLK:
END
SYM, 5-SHFTEN, INV
PIN, 0, 0, 5-SHFTEN
PIN, I, I, SREN
END
SYM, DPE, DFF
PIN, Q, 0, DPE
PIN, D, I, 4-SYNCPLS
PIN, C, I, CNTCLK
END
.
.
.
.
.
.
.
EXT, IO3B, B,, BLKNM=IO3B
EXT, IO4B, B,, BLKNM=IO4B
EXT, IO5B, B,, BLKNM=IO5B
EXT, IO6B, B,, BLKNM=IO6B
EXT, IO7B, B,, BLKNM=IO7B
EOF
```

Figure 12. Sections of a Xilinx Netlist File (XNF)

table and placement results. Refer to the Logic Cell Array Editor section of the XACT LCA Development System binders for a complete list and descriptions of APR commands and options.

APR is initiated by a command, with sample options, of the form:

```
apr -c name.cst -p -b50 -k10 input output
```

See Figure 13.

APRLOOP is a command which allows iterations of the APR program and saves the results of each. It is convenient to run it over night and evaluate several resulting

placements in terms of areas of congestion, querynet of selected nets for performance, number of unrouted pins, etc. The best aprloop result can be edited with XACT to move blocks, seed longlines and route critical nets. The result can then be run for additional time with a low starting temperature to minimize randomization of placement or with additional lock block or lock net constraints.

APRLOOP is initiated by a command, with sample options, of the form:

```
aprloop count (apr options) input output
```

where count is the number of iterations. The resulting files will have two digit suffixes appended to "output" name.

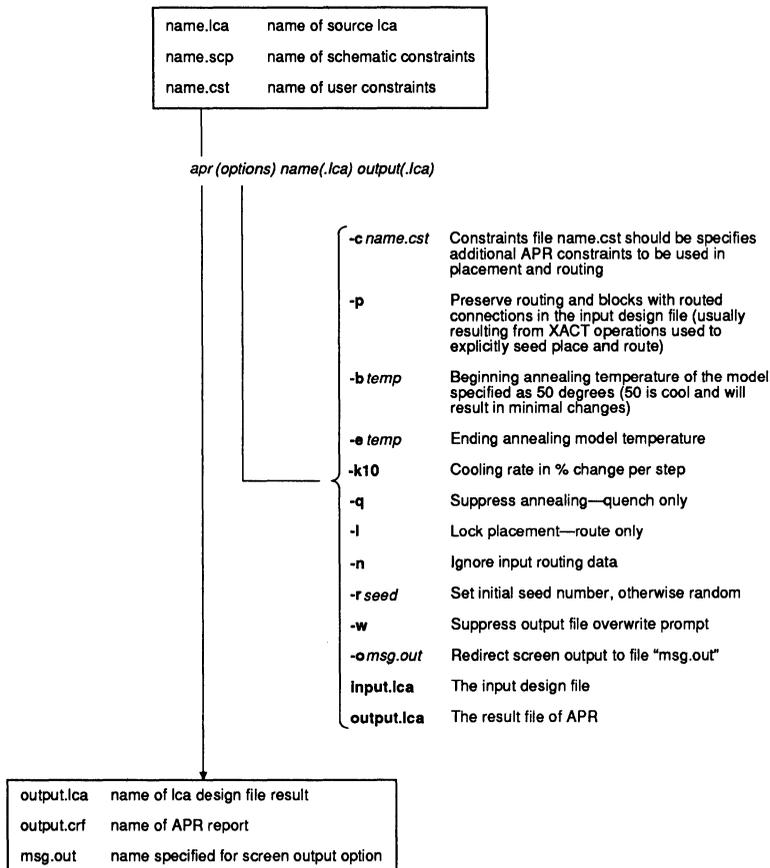


Figure 13. Automatic Placement and Routing of a Design

## XACT

### Software Organization

The software is installed on the user's hard disk in the `\xact`, `\xact\designs` and `\xact\macros` directories, which are created and added to the user path by the `a:install` operation from the installation disks or as otherwise described in individual program installation procedures. This fundamental software is invoked by the command `xact` which produces the graphic XACT executive screen. See Figure 14. Menu or keyboard commands select design name, device, speed and package options for a design to be created or select existing designs for modification. Unless otherwise specified, design files (name.LCA) and other files which are created, are placed in the current directory as specified by the XACT.PRO profile or by the `directory` command. If there is no XACT.PRO file in the directory which was the current directory when XACT was invoked, the profile of the \XACT directory will be used. See Figure 15. The default starts as `\xact\designs` directory. Details of the XACT executive and its programs can be found in the XACT LCA Development System binders.

### XACT Design Editor (DS-21)

The XACT Design Editor provides the designer with the ability to graphically enter a design and to generate the Logic Cell Array (LCA) configuration program data. Additional XACT programs can be invoked for configuration program generation and specification of system attributes can be made from the XACT executive. See Edit LCA, MAKEBITS and MAKEPROM of Figure 2. From within XACT, designs are created and/or modified with the `editlca` program selected by the mouse or from abbreviated keyboard entry, `e`. This graphics oriented program allows symbolic design entry by explicit assignment of interconnection nets to specified Configurable Logic Block (CLB) and I/O Block (IOB) pins. See Table 2 and Figure 16. The XACT LCA Development System documentation includes a tutorial which illustrates some of the basic functions. The editor can implement automatic or interactive routing of these nets. Within `editlca`, a block editor (`editblock`) allows the user to specify the functions of the individual blocks in terms of logic equations and block configuration options. See Figure 17. The designer can also implement standard functional logic blocks from Xilinx macros supplied in the



Figure 14. XACT Executive Screen with Programs Menu Selected

Directory C:\XACT\DESIGNS  
 Part 3020PC68  
 Speed -70  
 Cursor Cross  
 Mouse COM1 B1 Select B2 Done B3 Switch  
 Keydef F1 edit  
 Keydef Alt F1 quit y  
 Printer HPLaser  
 Design name.lca

lcact\macros directory and documented in the XACT LCA Development System "Macros" binder. These macros provide examples of implementing logic design functions in CLBs and IOBs. Within *editlca* of XACT, the designer can also verify logic path or interconnection timing, generate text or graphic printer disk files, generate custom user macros. Refer to the Logic Cell Array Editor section of the XACT LCA Development System binders for a complete list and descriptions of *editlca* and *editblock* commands.

Figure 15. An Example of an XACT.PRO File Showing Selected Options

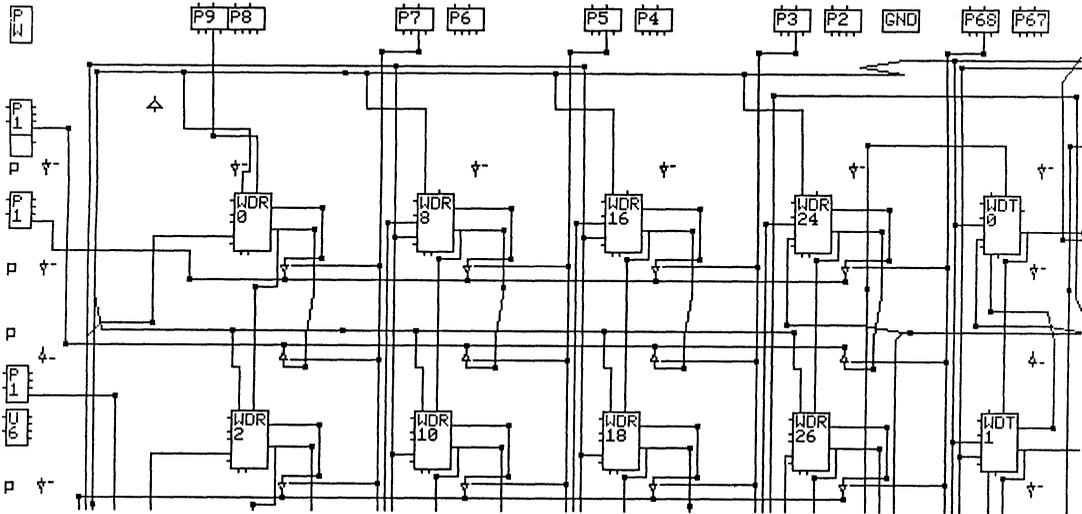


Figure 16. An XACT EditLCA Graphic Screen

X	QX	<b>Note:</b> M=F if E=0 M=G if E=1  <b>Blk: BB</b> <b>Sbus&lt;0&gt;</b>	
Y	M		
DX	K		
DV	EC	<b>A: 10R0</b> <b>B: DATA&lt;0&gt;</b> <b>C: S0</b> <b>D: Sbus&lt;1&gt;</b> <b>E: S1</b> <b>DI:</b> <b>ENC EC: 5-SHFTEN</b> <b>RD:</b> <b>CLK K: CWR</b> <b>QX X: Sbus&lt;0&gt;</b> <b>Y:</b>	
CLK	EC		
RSTDIR		<b>QXDC F</b> H X H H X L X H H	
ENCLK		<b>CBA G</b> H X H L X H	
<b>F = ~C*QX+C*D</b> <b>G = ~C*A+C*B</b>			

Figure 17. An XACT EditLCA Edit Block Screen

## Design Configuration

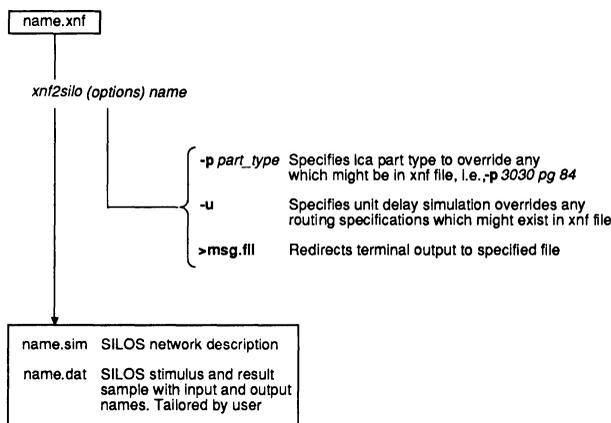
The MAKEBITS program is also invoked from the XACT executive. This program generates the binary database (*name.bit* disk file) for LCA configuration. The *makebits* program includes the *download* program which uses the binary configuration data and a parallel I/O port to drive the Xilinx Download Cable for direct in-circuit configuration of an LCA. This provides a convenient breadboard design verification capability. The conventional design flow uses the *makeprom* program to convert one or more binary *name.bit* files into a PROM format file. This file is used to program the configuration source PROM which programs the system LCAs on each power-up or reprogram command. Refer to the Logic Cell Array Editor section of the XACT LCA Development System binders for a complete list and descriptions of *makebit* and *makeprom* programs.

The LCA has several modes of configuration. Some use parallel data, others serial data; some provide their own timing, others are slaved to external clocks. Refer to the

device data Technical Data or the Configuration Application Note for detailed descriptions. The selection of a configuration mode and it's interface at the start of design will allow integration of that configuration interface and user system logic. The interaction of external printed circuit board layout and the internal routing of the LCA should also be considered as the design progresses.

## DESIGN VERIFICATION

Several methods of design verification are available at different points in the design process. With the basic DS-21 XACT Design Editor the designer can evaluate interconnect timing using the *querynet* command. Cumulative path timing between flip-flops and/or I/O can be evaluated with (*report*) *delay*. Compatibility of interconnect can be checked with the *drc* command. The Download cable and it's control program provide a convenient programming of a breadboard (or prototype) unit, but the user must allow sufficient margin for worst case production



1115 18

**Figure 18. The Optional Simulator Software Includes a Translator to Produce a Simulation Model from a Xilinx Netlist File**

Program	Command	File
editlca	file/save	name.lca (design text file of editlca commands)
		name.log (log of commands from previous edit)
	cutmacro print report	xact.pro (profile of default settings)
		macro.mac (preestablished editor commands)
		name.pic (to print: copy /b name.pic prn:)
makebits	makebits rawbit makemask	name.xxx (disk file of querynet, queryblock, delay or DRC)
		name.bit (binary database)
		name.rbt (ASCII text of configuration data)
		name.msk (binary data of config. care/don't)
makeprom	save	makebit.pro (profile of default options)
		name.xxx (PROM programmer HEX file)
		makeprom.pro (profile of default size, format, etc)

1115 08

**Table 2. List of XACT Programs with Some of Their Commands and Related Files**



input and output pins in a *.monitor* statement. The *.clk* statements specify timing of input transitions. For signal groups with common, periodic transition timing a *.pat* command is effective. Refer to the Simulator section of the

XACT LCA Development System binders for a complete list and descriptions of *xnf2sil0* and *SILOS* simulation commands and options.

```

$
$ Simulation input file
$ June 16, 1988
$
!INPUT top8.sim

$ Initial pulse to reset latches
GLOBALRESET- .CLK 0 S0 1 S1

$ Define the ABUS
.HEX ABUS0=AD3,AD2,AD1,AD0
.HEX ABUS1=AD7,AD6,AD5,AD4
.HEX ABUS2=A11.PAD,A10.PAD,A9.PAD,A8.PAD
.HEX ABUS3=A15.PAD,A14.PAD,A13.PAD,A12.PAD

. . . . .

$ Define the HEX values used in .PAT to be Resistive Strength
.IPARSE .HEX=R
.PAT ABUS3,ABUS2,ABUS1,ABUS0
$
$ Counter Test
0          F    F    F    0
200        0    0    0    1    $ CNT=1
400        F    F    F    2    $ CTEN Select
600        0    0    8    0    $ Preload Counter with 8 0
$
$ Shift Register Test
800        F    F    F    0    $ CREN Select
1000       0    0    0    D    $ S0,S1,S1OR0,CNT=1
1200       F    F    F    1    $ SREN Select

. . . . .

$ Define the HEX values used in .PAT to be Supply Strength
.IPARSE .HEX=S

.PAT IOB1, IOB2
$
$ Counter Test
0          0    1
75         1    1          $ CREN Select
175        0    1
275        0    0
375        0    1          $ CNT =1
475        1    1          $ CTEN Select

. . . . .

$. TABLE ABUS3,ABUS2,ABUS1,ABUS0;AB3,AB2,AB1,AB0;Y;
$+S3-CMDEN,S5-SHFTEN,CTEN,PAEN,PBEN;IOB1,IOB2,RD.PAD;
$+DATA<7>,DATA<6>,DATA<5>,DATA<4>,DATA<3>,DATA<2>,DATA<1>,DATA<0>;
$+CWR,S5-SHFTEN,S0,S1,S1OR0;
$+SBUS<7>,SBUS<6>,SBUS<5>,SBUS<4>,SBUS<3>,SBUS<2>,SBUS<1>,SBUS<0>

```

Figure 20. Simulation Stimulus (name.dat)





# LCA Macro Library Listings

		XC3000				XC2000			
		CLBs	XACT	DASH/LCA	SCHEMA	CLBs	XACT	DASH/LCA	SCHEMA
<b>PADS</b>									
PIN	Input Pad .....	0	✓	✓	-	0	✓	✓	-
PINQ	Input Pad with Registered Input .....	0	✓	✓	-	0	✓	✓	-
PINR	Input Pad with Registered and Direct Inputs .....	0	✓	-	-	-	-	-	-
PIO	Input/Output Pad .....	0	✓	✓	-	0	✓	✓	-
PIOQ	Input/Output Pad with Registered Input .....	0	✓	✓	-	0	✓	✓	-
POUT	Output Pad .....	0	✓	✓	-	0	✓	✓	-
POUTZ	Output Pad with 3-State Control .....	0	✓	✓	-	0	✓	✓	-
POUTQ	Output Pad with Registered Output .....	0	✓	-	-	-	-	-	-
PREG	Pad with Data Register .....	0	✓	✓	-	0	✓	✓	-
PREG2	Pad with 2-Bit Shift Register .....	0	✓	-	-	-	-	-	-
<b>IOB SCHEMATIC ELEMENTS</b>									
TBUF	Internal Three-State Buffer .....	0	✓	✓	✓	-	-	-	-
ACLK	Auxiliary Buffer .....	0	✓	✓	✓	0	✓	✓	✓
GCLK	Global Buffer .....	0	✓	✓	✓	0	✓	✓	✓
IBUF	Input Buffer .....	0	-	✓	✓	0	-	✓	✓
INFF	Input Flip-Flop .....	0	-	✓	✓	0	-	✓	✓
INLAT	Input Latch .....	0	-	✓	-	-	-	-	-
OBUF	Output Buffer .....	0	-	✓	✓	0	-	✓	✓
OBUFZ	Output Buffer with Output Enable .....	0	-	✓	✓	0	-	✓	✓
OUTFF	Output Flip-Flop .....	0	-	✓	-	-	-	-	-
OUTFFZ	Output Flip-Flop with OBUFZ .....	0	-	✓	-	-	-	-	-
BPAD	Bidirectional Package Pin Symbol .....	0	-	✓	✓	0	-	✓	✓
IPAD	Input Package Pin Symbol .....	0	-	✓	✓	0	-	✓	✓
OPAD	Output Package Pin Symbol .....	0	-	✓	✓	0	-	✓	✓
UPAD	Unbonded Die Pad Symbol .....	0	-	✓	✓	0	-	✓	✓
BUF	Internal non-inverting Buffer .....	0	-	✓	✓	0	-	✓	✓
INV	Inverter .....	1	-	✓	✓	1	-	✓	✓
PULLUP	Input pull-up Resistor .....	0	-	✓	-	-	-	-	-
OINV	Inverting Output Buffer .....	0	-	✓	-	-	-	✓	-
IOFF	I/O Pad as Flip-flop .....	-	-	-	-	0	-	✓	-

		XC3000				XC2000				
		CLBs	XACT	DASH/LCA	SCHEMA	DS-40	CLBs	XACT	DASH/LCA	SCHEMA
<b>GENERAL</b>										
GADD	1-Bit Full Adder .....	1	✓	✓	✓	-	1	✓	✓	✓
GCOMP	2-Bit Comparator .....	1	✓	✓	✓	-	1	✓	✓	✓
GLTGT	2-Bit Less Than/Greater Than Comparator .....	1	✓	✓	-	-	-	-	-	-
GEQGT	2-Bit Equal/Greater Than Comparator .....	1	-	-	✓	-	1	✓	✓	✓
GMUX	2-to-1 Mux .....	1	✓	✓	✓	-	1	✓	✓	✓
OSC	Crystal Osc .....	0	-	✓	✓	-	0	-	-	✓
GXTL	Crystal Osc (XACT: 3020 GXTL20, 2018 GXTL2) .....	0	✓	✓	-	-	0	✓	✓	-
GOSC	Low Frequency Resistor-Capacitor Oscillator .....	1	✓	-	✓	-	1	✓	-	✓
GMAJ	Majority Gate .....	1	✓	-	✓	-	1	✓	✓	✓
GXOR	Exclusive-OR .....	-	-	-	-	-	1	✓	✓	-
GXOR2	Dual Exclusive-OR .....	-	-	-	-	-	1	✓	-	-
GPARG	Parity Test (Even = Low) .....	-	-	-	-	-	1	✓	✓	-
HX83	4-Bit Binary Adder With Fast Carry .....	6	-	-	✓	-	-	-	-	-
HX85	4-Bit Magnitude Comparator .....	7	-	-	✓	-	-	-	-	-
HX280	9-Bit Parity Checker / Generator .....	3	-	-	✓	-	-	-	-	-
HX283	4-Bit Binary Full Adder .....	6	-	-	✓	-	-	-	-	-
HX518	8-Bit Identity Comparator .....	5	-	-	✓	-	-	-	-	-
HX521	8-Bit Identity Comparator .....	5	-	-	✓	-	-	-	-	-
HX125	Three-State Bus Buffer .....	0	-	-	✓	-	-	-	-	-
HX240	Octal Inverting Buffer, Three-State Outputs .....	4	-	-	✓	-	-	-	-	-
HX241	Octal Non-inverting Buffer, Three-State Outputs .....	1	-	-	✓	-	-	-	-	-
HX244	Octal Non-inverting Buffer, Three-State Outputs .....	0	-	-	✓	-	-	-	-	-
HX245	Octal Bidirectional Transceiver .....	1	-	-	✓	-	-	-	-	-
HX540	Octal Inverting, Three-State Outputs .....	0	-	-	✓	-	-	-	-	-
HX541	Octal Non-inverting, Three-State Outputs .....	0	-	-	✓	-	-	-	-	-
<b>LATCHES</b>										
LD	Data Latch .....	1	✓	✓	✓	-	1	✓	✓	✓
LDRD	Data Latch with Reset Direct .....	1	✓	✓	✓	-	1	✓	✓	✓
LDSD	Data Latch with Set Direct .....	1	✓	✓	✓	-	1	✓	✓	✓
LRS	Set-Reset Data Latch with Reset Dominant .....	1	✓	✓	-	-	-	-	-	-
LDM	Data Latch with 2-Input Data Mux .....	-	-	-	-	-	1	✓	✓	✓
LDMRD	Data Latch with 2-Input Data Mux with Reset Direct .....	-	-	-	-	-	1	✓	✓	✓
LDMSD	Data Latch with 2-Input Data Mux with Set Direct .....	-	-	-	-	-	1	✓	✓	✓
LDSRD	Data Latch with Set Direct, Reset Direct .....	-	-	-	-	-	1	✓	✓	✓
HX77	2-Bit Latch .....	1	-	-	✓	-	-	-	-	-
HX259	8-Bit Addressable Latch .....	8	-	-	✓	-	-	-	-	-
HX373	Octal Latch with Three-State Outputs .....	4	-	-	✓	-	-	-	-	-

**FLIP-FLOPS**

		XC3000					XC2000			
		CLBs	XACT	DASH/LCA	SCHEMA	DS-40	CLBs	XACT	DASH/LCA	SCHEMA
DFF	D Flip-Flop	1	-	-	✓	-	1	-	-	✓
FD	D Flip-Flop	1	✓	-	✓	-	1	-	-	✓
FDRD	D Flip-Flop with Reset Direct	1	✓	✓	✓	-	1	✓	✓	✓
FDS	D Flip-Flop with Set Direct	1	-	-	✓	-	1	✓	✓	✓
FDSRD	D Flip-Flop with Set Direct, Reset Direct	1	-	-	✓	-	1	✓	✓	✓
FDC	D Flip-Flop with Clock Enable	1	✓	✓	✓	-	1	✓	✓	✓
FDCRD	D Flip-Flop with Clock Enable, Reset Direct	1	✓	✓	✓	-	1	-	✓	✓
FDCR	D Flip-Flop with Clock Enable, Reset	1	✓	✓	✓	-	1	✓	✓	✓
FDCS	D Flip-Flop with Clock Enable, Set	1	✓	✓	✓	-	1	✓	✓	✓
FDR	D Flip-Flop with Reset	1	✓	✓	✓	-	1	✓	✓	✓
FDS	D Flip-Flop with Set	1	✓	✓	✓	-	1	✓	✓	✓
FRS	Set-Reset Flip-Flop with Reset Dominant	1	✓	✓	✓	-	1	✓	✓	✓
FSR	Set-Reset Flip-Flop with Set Dominant	1	✓	✓	✓	-	1	✓	✓	✓
FDM	D Flip-Flop with 2-Input Data Mux	1	✓	✓	✓	-	1	✓	✓	✓
FDMRD	D Flip-Flop with 2-Input Data Mux with Reset Direct	1	✓	✓	✓	-	1	✓	✓	✓
FDMSD	D Flip-Flop with 2-Input Data Mux with Set Direct	1	-	-	✓	-	1	✓	✓	✓
FDMR	D Flip-Flop with 2-Input Data Mux with Reset	1	✓	✓	✓	-	1	✓	✓	✓
FDMS	D Flip-Flop with 2-Input Data Mux with Set	1	✓	✓	✓	-	1	✓	✓	✓
FJK	J-K Flip-Flop	1	✓	✓	✓	-	1	✓	✓	✓
FJKRD	J-K Flip-Flop with Reset Direct	1	✓	✓	✓	-	1	✓	✓	✓
FJKSD	J-K Flip-Flop with Set Direct	1	-	-	✓	-	1	✓	✓	✓
FJKSRD	J-K Flip-Flop with Set Direct, Reset Direct	1	-	-	✓	-	1	✓	✓	✓
FJKS	J-K Flip-Flop with Set	1	✓	✓	✓	-	1	✓	✓	✓
FT0	Self Toggle Flip-Flop	1	✓	✓	✓	-	1	✓	✓	✓
FT0RD	Self Toggle Flip-Flop with Reset Direct	1	✓	✓	-	-	-	-	-	-
FT0R	Self Toggle Flip-Flop with Reset	1	✓	✓	✓	-	1	✓	✓	✓
FT	Toggle Flip-Flop	1	✓	✓	✓	-	1	✓	✓	✓
FTRD	Toggle Flip-Flop with Reset Direct	1	✓	✓	✓	-	1	-	✓	✓
FTP	Toggle Flip-Flop with Parallel Enable	1	✓	✓	✓	-	1	✓	✓	✓
FTPRD	Toggle Flip-Flop with Parallel Enable, Reset Direct	1	✓	✓	✓	-	1	✓	✓	✓
FTR	Toggle Flip-Flop with Reset	1	✓	✓	✓	-	1	✓	✓	✓
FTS	Toggle Flip-Flop with Set	1	✓	✓	✓	-	1	✓	✓	✓
FT2	2-Input Toggle Flip-Flop	1	-	-	✓	-	1	✓	✓	✓
FT2R	2-Input Toggle Flip-Flop with Reset	1	-	-	✓	-	1	✓	✓	✓
NDFP	Negative Edge Flip-Flop Primitive	1	-	✓	-	-	1	-	✓	-
PDFP	Positive Edge Flip-Flop Primitive	1	-	✓	-	-	1	-	✓	-

		XC3000				XC2000				
		CLBs	XACT	DASH/LCA	SCHEMA	DS-40	CLBs	XACT	DASH/LCA	SCHEMA
<b>DECODERS/ENCODERS</b>										
D2-4	1-of-4 Decoder .....	2	✓	✓	✓	-	2	✓	✓	✓
D2-4E	1-of-4 Decoder with Enable .....	2	✓	✓	✓	-	2	✓	✓	✓
74-139	1-of-4 Single Decoder with Enable, Low Output .....	2	✓	✓	✓	-	2	✓	✓	✓
D3-8	1-of-8 Decoder .....	4	✓	✓	✓	-	4	✓	✓	✓
D3-8E	1-of-8 Decoder with Enable .....	4	✓	✓	✓	-	5	✓	✓	✓
74-138	1-of-8 Decoder with Enables, Low Output .....	5	✓	✓	✓	-	6	✓	✓	✓
74-42	1-of-10 Decoder with Low Output .....	5	✓	✓	✓	-	7	✓	✓	✓
HX42	4-to-10 Line Decoder .....	5	-	-	✓	-	-	-	-	-
HX48	BCD to Seven Segment Decoder .....	5	-	-	✓	-	-	-	-	-
HX138	1-of-8 Decoder/Demultiplexer .....	5	-	-	✓	-	-	-	-	-
HX139	1-of-4 Decoder .....	2	-	-	✓	-	-	-	-	-
HX147	10-to-4 Line Priority Encoder .....	5	-	-	✓	-	-	-	-	-
HX148	3-to-8 Line Priority Encoder .....	9	-	-	✓	-	-	-	-	-
HX154	1-of-16 Decoder/Demultiplexer .....	9	-	-	✓	-	-	-	-	-
HX278	4-Bit Cascadable Priority Encoder .....	6	-	-	✓	-	-	-	-	-
<b>MULTIPLEXERS</b>										
M3-1	3-to-1 Mux .....	1	✓	✓	✓	-	2	✓	✓	✓
M3-1E	3-to-1 Mux with Enable .....	2	✓	✓	✓	-	2	✓	✓	✓
M4-1	4-to-1 Mux .....	2	✓	✓	✓	-	3	✓	✓	✓
M4-1E	4-to-1 Mux with Enable .....	2	✓	✓	✓	-	3	✓	✓	✓
74-352	4-to-1 Mux with Enable, Low Output .....	2	✓	✓	✓	-	3	✓	✓	✓
M4-2	4-to-2 Mux .....	1	✓	✓	-	-	-	-	-	-
M8-1	8-to-1 Mux .....	4	✓	✓	✓	-	7	✓	✓	✓
M8-1E	8-to-1 Mux with Enable .....	4	✓	✓	✓	-	7	✓	✓	✓
74-151	8-to-1 Mux with Enable, Complementary Outputs .....	4	✓	✓	✓	-	7	✓	✓	✓
74-152	8-to-1 Mux with Low Output .....	4	✓	✓	✓	-	7	✓	✓	✓
MZ8-1	8-to-1 Mux Using Three-State Buffers .....	0	✓	-	-	-	-	-	-	-
HX151	8 Input Multiplexer .....	5	-	-	✓	-	-	-	-	-
HX152	8 Input Multiplexer .....	4	-	-	✓	-	-	-	-	-
HX153	Dual 4 Input Multiplexer .....	6	-	-	✓	-	-	-	-	-
HX157	Quad 2 Input Multiplexer .....	4	-	-	✓	-	-	-	-	-
HX158	Quad 2 Input Multiplexer .....	3	-	-	✓	-	-	-	-	-
HX257	Quad 2-to-1 Multiplexer with Enable .....	2	-	-	✓	-	-	-	-	-
HX258	Quad 2-to-1 Inverting Multiplexer .....	2	-	-	✓	-	-	-	-	-
HX352	4-to-1 Data Selector / Multiplexer .....	5	-	-	✓	-	-	-	-	-

**REGISTERS**
**Data Registers**

RD4	4-Bit Data Register .....
RD4RD	4-Bit Data Register .....
RD8	8-Bit Data Register .....
RD8RD	8-Bit Data Register with Reset Direct .....
RD8CR	8-Bit Data Register with Clock Enable, Reset .....
HX174	Hex D Register with Master Reset .....
HX273	Octal D Flip-flop .....
HX298	Quad 2 Input Flip-flop .....
HX374	Octal D Flip-flops with Three-State Outputs .....
HX377	Octal D Flip-flops with Clock Enable .....
HX577	Octal D Flip-flops with Reset and Three-State Outputs .....

**Serial to Parallel**

RS4	4-Bit Shift Register .....
RS4RD	4-Bit Shift Register with Reset Direct .....
RS4C	4-Bit Shift Register with Clock Enable .....
RS4CRD	4-Bit Shift Register with Clock Enable, Reset Direct .....
RS4CR	4-Bit Shift Register with Clock Enable, Reset .....
74-195	4-Bit Serial to Parallel Shift Register with ParEna, MRLow .....
74-194	4-Bit Bi-Directional SR with ClkEna, ParEna, MRLow .....
RS8	8-Bit Shift Register .....
RS8RD	8-Bit Shift Register with Reset Direct .....
RS8R	8-Bit Shift Register with Reset .....
RS8C	8-Bit Shift Register with Clock Enable .....
RS8CRD	8-Bit Shift Register with Clock Enable, Reset Direct .....
RS8CR	8-Bit Shift Register with Clock Enable, Reset .....
RS8PR	8-Bit Shift Register with Parallel Enable, Reset .....
74-164	8-Bit Serial to Parallel Shift Register with Master Reset Low .....
HX164	8-Bit Serial In-Parallel Out Shift Register .....
HX166	Parallel Load 8-Bit Shift Register .....
HX179	4-Bit Parallel Access Shift Register .....
HX194	4-Bit Bidirectional Universal Shift Register .....
HX195	4-Bit Parallel Access Shift Register .....
HX198	8-Bit Bidirectional Shift Register .....
HX199	8-Bit Shift Register with Clock Inhibit .....
HX595	8-Bit Shift Register with Three-State Register Output .....

		XC3000				XC2000			
		CLBs	XACT	DASH-LCA	SCHEMA	CLBs	XACT	DASH-LCA	SCHEMA
RD4	4-Bit Data Register .....	2	✓	✓	-	4	✓	✓	✓
RD4RD	4-Bit Data Register .....	2	✓	✓	-	-	-	-	-
RD8	8-Bit Data Register .....	4	✓	✓	-	8	✓	✓	✓
RD8RD	8-Bit Data Register with Reset Direct .....	4	✓	✓	-	-	-	-	-
RD8CR	8-Bit Data Register with Clock Enable, Reset .....	4	✓	✓	✓	8	✓	✓	✓
HX174	Hex D Register with Master Reset .....	4	-	-	✓	-	-	-	-
HX273	Octal D Flip-flop .....	4	-	-	✓	-	-	-	-
HX298	Quad 2 Input Flip-flop .....	4	-	-	✓	-	-	-	-
HX374	Octal D Flip-flops with Three-State Outputs .....	4	-	-	✓	-	-	-	-
HX377	Octal D Flip-flops with Clock Enable .....	4	-	-	✓	-	-	-	-
HX577	Octal D Flip-flops with Reset and Three-State Outputs .....	4	-	-	✓	-	-	-	-
RS4	4-Bit Shift Register .....	2	✓	✓	✓	4	✓	✓	✓
RS4RD	4-Bit Shift Register with Reset Direct .....	2	✓	✓	-	-	-	-	-
RS4C	4-Bit Shift Register with Clock Enable .....	2	✓	✓	-	-	-	-	-
RS4CRD	4-Bit Shift Register with Clock Enable, Reset Direct .....	2	✓	✓	-	-	-	-	-
RS4CR	4-Bit Shift Register with Clock Enable, Reset .....	2	✓	✓	-	-	-	-	-
74-195	4-Bit Serial to Parallel Shift Register with ParEna, MRLow .....	3	✓	✓	✓	5	✓	✓	✓
74-194	4-Bit Bi-Directional SR with ClkEna, ParEna, MRLow .....	5	✓	✓	✓	12	✓	✓	✓
RS8	8-Bit Shift Register .....	4	✓	✓	✓	8	✓	✓	✓
RS8RD	8-Bit Shift Register with Reset Direct .....	4	✓	✓	-	-	-	-	-
RS8R	8-Bit Shift Register with Reset .....	4	✓	✓	✓	8	✓	✓	✓
RS8C	8-Bit Shift Register with Clock Enable .....	4	✓	✓	-	-	-	-	-
RS8CRD	8-Bit Shift Register with Clock Enable, Reset Direct .....	4	✓	✓	-	-	-	-	-
RS8CR	8-Bit Shift Register with Clock Enable, Reset .....	4	✓	✓	✓	8	✓	✓	✓
RS8PR	8-Bit Shift Register with Parallel Enable, Reset .....	4	✓	✓	✓	8	✓	✓	✓
74-164	8-Bit Serial to Parallel Shift Register with Master Reset Low .....	5	✓	✓	✓	8	✓	✓	✓
HX164	8-Bit Serial In-Parallel Out Shift Register .....	5	-	-	✓	-	-	-	-
HX166	Parallel Load 8-Bit Shift Register .....	6	-	-	✓	-	-	-	-
HX179	4-Bit Parallel Access Shift Register .....	5	-	-	✓	-	-	-	-
HX194	4-Bit Bidirectional Universal Shift Register .....	7	-	-	✓	-	-	-	-
HX195	4-Bit Parallel Access Shift Register .....	3	-	-	✓	-	-	-	-
HX198	8-Bit Bidirectional Shift Register .....	14	-	-	✓	-	-	-	-
HX199	8-Bit Shift Register with Clock Inhibit .....	7	-	-	✓	-	-	-	-
HX595	8-Bit Shift Register with Three-State Register Output .....	9	-	-	✓	-	-	-	-

		XC3000				XC2000				
		CLBs	XACT	DASH/LCA	SCHEMA	DS-40	CLBs	XACT	DASH/LCA	SCHEMA
<b>COUNTERS</b>										
<b>Modulo 2</b>										
C2BCP	1-Bit Binary Counter with Clock Enable, Parallel Enable .....	1	✓	✓	-	-	-	-	-	-
C2BCPRD	1-Bit Binary Counter with Clock Enable, ParEna, Reset Direct .....	1	✓	✓	-	-	-	-	-	-
C2BCR	1-Bit Binary Counter with Clock Enable, Reset .....	1	✓	✓	✓	-	1	✓	✓	✓
C2BCRD	1-Bit Binary Counter with Clock Enable, Reset Direct .....	1	✓	✓	✓	-	1	✓	✓	✓
C2BP	1-Bit Binary Counter with Parallel Enable .....	1	✓	✓	-	-	1	✓	✓	✓
C2BR	1-Bit Binary Counter with Reset .....	1	✓	✓	-	-	1	✓	✓	✓
C2BRD	1-Bit Binary Counter with Reset Direct .....	1	✓	✓	✓	-	1	✓	✓	✓
C4BCP	2-Bit Binary Counter with Clock Enable, Parallel Enable .....	2	✓	✓	✓	-	3	✓	✓	✓
C4BCPRD	2-Bit Binary Counter with Clock Enable, ParEna, Reset Direct .....	2	✓	✓	✓	-	-	-	-	-
C4BCR	2-Bit Binary Counter with Clock Enable, Reset .....	2	✓	✓	✓	-	2	✓	✓	✓
C4BCRD	2-Bit Binary Counter with Clock Enable, Reset Direct .....	2	✓	✓	✓	-	2	✓	✓	✓
C4JX	2-Bit Expandable Johnson Counter .....	1	✓	✓	-	-	-	-	-	-
C4JXRD	2-Bit Expandable Johnson Counter with Reset Direct .....	1	✓	✓	-	-	-	-	-	-
C4JXC	2-Bit Expandable Johnson Counter with Clock Enable .....	1	✓	✓	-	-	-	-	-	-
C4JXCRD	2-Bit Expandable Johnson Counter with ClkEna, Reset Direct .....	1	✓	✓	✓	-	-	-	-	-
C4JXCR	2-Bit Expandable Johnson Counter with ClkEna, Reset .....	1	✓	✓	✓	-	-	-	-	-
C4JCR	2-Bit Johnson Counter with Clock Enable, Reset .....	-	-	-	-	-	2	✓	✓	✓
<b>Modulo 6</b>										
C6JCR	3-Bit Johnson Counter with Clock Enable, Reset .....	2	✓	✓	✓	-	3	✓	✓	✓
<b>Modulo 8</b>										
C8BCP	3-Bit Binary Counter with Clock Enable, Parallel Enable .....	3	✓	✓	✓	-	5	✓	✓	✓
C8BCPRD	3-Bit Binary Counter with Clock Enable, ParEna, Reset Direct .....	3	✓	✓	-	-	-	-	-	-
C8BCR	3-Bit Binary Counter with Clock Enable, Reset .....	3	✓	✓	✓	-	4	✓	✓	✓
C8BCRD	3-Bit Binary Counter with Clock Enable, Reset Direct .....	2	✓	✓	✓	-	4	✓	✓	✓
C8JCR	4-Bit Johnson Counter with Clock Enable, Reset .....	2	✓	✓	✓	-	4	✓	✓	✓
<b>Modulo 10</b>										
C10BCRD	4-Bit BCD Counter with Clock Enable, ResetDir .....	3	✓	✓	✓	-	4	✓	✓	✓
C10BCPRD	4-Bit BCD Counter with Parallel Enable, ResetDir .....	4	✓	✓	✓	-	7	✓	✓	✓
74-160	4-Bit BCD Counter with Clock Enable, ParEnaL, MRLow .....	6	✓	✓	✓	-	8	✓	✓	✓
74-162	4-Bit BCD Counter with Clock Enable, ParEnaL, Reset Low .....	7	✓	✓	-	-	-	-	-	-
C10BPRD	4-Bit BCD Counter with Parallel Enable, Reset Direct .....	4	✓	✓	✓	-	6	✓	✓	✓
C10JCR	5-Bit Johnson Counter with Clock Enable, Reset .....	3	✓	✓	✓	-	5	✓	✓	✓
HX160	Presettable Decade Counter .....	8	-	-	-	✓	-	-	-	-
HX162	Presettable Decade Counter with Sync Clear .....	10	-	-	-	✓	-	-	-	-
HX168	4-Bit BCD Synchronous Up/Down Counter .....	11	-	-	-	✓	-	-	-	-
HX390	4-Bit Decade Counters with Clear .....	3	-	-	-	✓	-	-	-	-
<b>Modulo 12</b>										
C12JCR	6-Bit Johnson Counter with Clock Enable, Reset .....	3	✓	✓	✓	-	6	✓	✓	✓

**COUNTERS (Continued)**  
**Modulo 16**

C16BARD	4-Bit Binary Ripple Counter with Reset Direct .....
C16BCRD	4-Bit Binary Counter with Clock Enable, Reset Direct .....
C16BCP	4-Bit Binary Counter with Clock Enable, Parallel Enable .....
C16BCPRD	4-Bit Binary Counter with Clock Enable, ParEna, Reset Direct
74-161	4-Bit Binary Counter with Clock Enable, ParEnaL, MRLow .....
C16BCPR	4-Bit Binary Counter with Clock Enable, ParEna, Reset .....
74-163	4-Bit Binary Counter with Clock Enable, ParEnaL, Reset Low ..
C16BPRD	4-Bit Binary Counter with Parallel Enable, Reset Direct .....
C16BUDRD	4-Bit Binary Up-Down Counter with Parallel Enable, ResetDir ..
C16JCR	8-Bit Johnson Counter with Clock Enable, Reset .....
HX161	Presettable Binary Counter .....
HX163	Synchronous Binary Counter with Sync Clear .....
HX169	4-Bit Binary Synchronous Up/Down Counter .....
HX393	4-Bit Binary Counters with Clear .....
HX590	8 Bit Binary Counter with Register and Three-State Output .....

**Modulo 256**

C256BCRD	8-Bit Binary Counter with Clock Enable, Reset Direct .....
C256BCR	8-Bit Binary Counter with Clock Enable, Reset .....
C256BCP	8-Bit Binary Counter with Clock Enable, Parallel Enable .....
C256BCPRD	8-Bit Binary Counter with Clock Enable, ParEna, Reset Direct
C256FCRD	8-Bit Modulo 256 Feedback Shift Register w/ ClkEna, ResetDir

	XC3000				XC2000				
	CLBs	XACT	DASH-LCA	SCHEMA	DS-40	CLBs	XACT	DASH-LCA	SCHEMA
	2	✓	✓	✓	-	4	✓	✓	✓
	3	✓	✓	✓	-	4	✓	✓	✓
	5	✓	✓	-	-	-	-	-	-
	5	✓	✓	✓	-	6	✓	✓	✓
	6	✓	✓	✓	-	8	✓	✓	✓
	6	✓	✓	✓	-	10	✓	✓	-
	7	✓	✓	-	-	-	-	-	-
	4	✓	✓	✓	-	5	✓	✓	✓
	5	✓	✓	✓	-	8	✓	✓	✓
	4	✓	✓	✓	-	8	✓	✓	✓
	6	-	-	-	✓	-	-	-	-
	8	-	-	-	✓	-	-	-	-
	7	-	-	-	✓	-	-	-	-
	5	-	-	-	✓	-	-	-	-
	13	-	-	-	✓	-	-	-	-
	7	✓	✓	-	-	-	-	-	-
	7	✓	✓	-	-	-	-	-	-
	8	✓	✓	-	-	-	-	-	-
	8	✓	✓	-	-	-	-	-	-
	6	✓	✓	✓	-	9	✓	✓	✓

## Product Brief

---

### FEATURES

- Runs on an IBM® PC/AT™ or compatible computer
- Complete basic system for design using Logic Cell™ Arrays
- Interactive graphical design editor
- Simplified definition, placement and interconnection capability for logic design and implementation
- Macro library of 113 standard logic family equivalents
- Utility for user-defined macros
- Boolean equation or Karnaugh map alternatives to specify logic functions
- Point to point timing calculations for critical path analysis
- Automatic design consistency checking for connectivity and design violations
- Documentation support with hardcopy output of logical and physical configuration information
- Download cable to transfer configuration programs from PC to LCA in target system
- Compatible hardware and software options to enhance design productivity

### GENERAL

The XACT™ Design Editor provides users with a complete design and development system for specification and implementation of designs using Xilinx Logic Cell Arrays. Functional definition of Configurable Logic Blocks (CLBs), Input/Output Blocks (IOBs) and interconnection is performed with a menu driven interactive graphics editor. An automatic router greatly reduces the effort to interconnect logic.

Designs are captured with a graphics based design editor using either a mouse for menu driven entry, or a keyboard for command driven entry. Functions are specified by CLB and IOB definitions plus their interconnections. The macro library and user defined macros enable the user to easily implement complex functions.

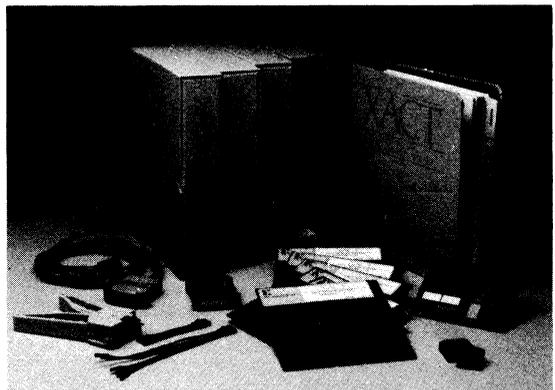
The check for logic connectivity and design rule violation is easily performed. All unused internal nodes are automatically configured to minimize power dissipation.

Interactive point-to-point timing delay calculation is provided for timing analysis and critical path determination. This ability enables the user to quickly identify and correct timing problems while the design is in progress.

The XACT Design Editor includes hardcopy generation to document a design and automatically track design changes. Logic Cell Array configuration programs can be automatically translated into standard EPROM programming bit pattern formats.

A download cable included with the design editor is useful for transferring configuration programs serially from the PC workstation to a Logic Cell Array installed in a system. During product development and debug this capability can be used to save the time required to write a modified configuration program into an EPROM.

Xilinx provides ongoing support for XACT users. For the first year, software updates are included. After that, the user may purchase the XC-SC21 Annual Support Agreement to continue to receive the latest software releases. XACT users also receive a technical bulletin which includes information about Logic Cell Arrays, software updates and hints for designers. In addition, Xilinx operates an electronic bulletin board to provide software enhancements and interactive factory support.





# XC-DS22 PC-SILOS™ SIMULATOR

---

## Product Brief

---

### FEATURES

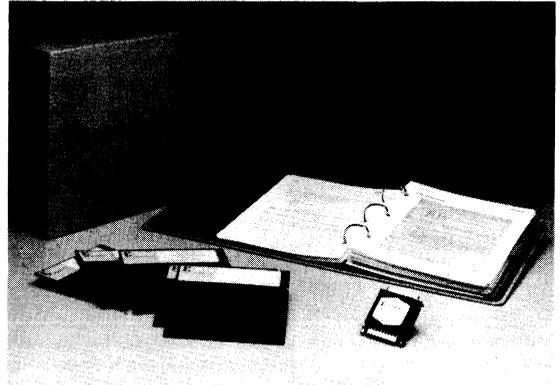
- General purpose event driven logic and timing simulator
- Input automatically generated from XNF file
- Control and observation of any physical circuit node
- Multiple file input for vectors and commands
- Interactive or batch mode operation
- Output available in printed or tabular formats
- Runs on an IBM® PC/AT™ or compatible personal computer

### GENERAL

PC-SILOS is a powerful PC based simulator that provides event driven logic and timing simulation of Logic Cell™ Array designs. Simulation is particularly useful for testing designs or design segments as well as for verifying critical timing over worst case power supply, temperature and process conditions.

Simulation is useful in several stages of the design cycle. After design entry, simulation may be used to debug logic in an unplaced and unrouted design. This saves design time because logic errors can be detected and corrected prior to final placement and routing. After a circuit has been placed, routed and then fully debugged using in-circuit emulation, worst case timing may be verified. This enables the user to select the correct Logic Cell Array speed grade for a particular application.

Network inputs for Logic Cell Array designs are automatically created by the XNF2SILO utility. The network includes logic and routing delay parameters and setup and hold times based upon the selected speed grade operating under worst case conditions. Simulation stimuli are created with a set of clock statements or with an input pattern for either pad inputs or internal nodes. Simulation results are available in tabular, plotted and graphic formats. This flexibility makes the debugging easy for both the circuit function and timing.





# XC-DS23 Automated Design Implementation

## Product Brief

### FEATURES

- Complete system for translating programmable gate array designs into Logic Cell™ Array implementations
- Accepts any combination of schematic netlists and PALASM2™ text files
- PLD conversion software allows design entry with PALASM2™ format equations
- Logic synthesis software optimizes designs for Logic Cell Array architecture
- Automatic logic reduction and partitioning removes unused, disabled logic
- Automatic placement and routing of logic minimizes design cycle time
- Runs on IBM® PC/AT™ or compatible personal computer

### GENERAL

The Automated Design Implementation (ADI) software converts design descriptions for programmable gate arrays into Logic Cell Array (LCA) implementations. Design descriptions may combine both schematics and PLD equations. Designers do not need extensive knowledge of the LCA architecture.

Logic synthesis software (XNFOPT) is provided to automatically optimize design logic for all (or specified portions) of the LCA design. Additional optimization software is included in the partitioning software (XNF2LCA) to automatically eliminate unused and redundant logic. An automatic place and route program maps the design logic into CLB's and IOB's.

### DESIGN IMPLEMENTATION PROCESS

Designers often describe portions of their design—such as counters and glue logic—with schematics, and other portions of the design—such as decoders—with Boolean equations. XC-DS23 ADI software allows designers to merge multiple modes of design entry into a single design.

The following figure shows a “top-level” design schematic with some glue logic, a TTL macro, and a “PLD” symbol. The “PLD” design is entered in a PALASM2™ format text file with the same name as the PLD symbol in the schematic (PAL1 in this example.) The multiple mode design is combined into a programmable gate array in three steps. First, the schematic is converted by the netlist translator to

a Xilinx Netlist Format (XNF) file. Second, the PALASM2 text file is translated (PDS2XNF) and then optimized (XNFOPT), resulting in a second XNF file.

Third, XNFMERGE flattens the design hierarchy and combines the multiple XNF files created from the different design entry methods. XNFMERGE is a tool that allows large designs to be entered in modules and then linked. This can be useful when several designers are working on the same design as a team.

XC-DS23 PLD Conversion Software (PDS2XNF and XNFOPT) also permits an entire design to be specified with PALASM2 format files. The equations can be generated without regard for the specified PLD type, thereby eliminating the architectural limitations of PLDs. For example, additional inputs could be added to a 10H8. After translating with PDS2XNF, the XNFOPT program uses logic synthesis algorithms to optimize the design for the multi-level logic of the LCA architecture. Designers may elect to optimize for either smallest area or highest performance.

XNF2LCA partitions the design logic into LCA resources: CLB's and IOB's. It also optimizes the design to remove unused logic like an unused terminal count or an AND gate with one input tied to ground.

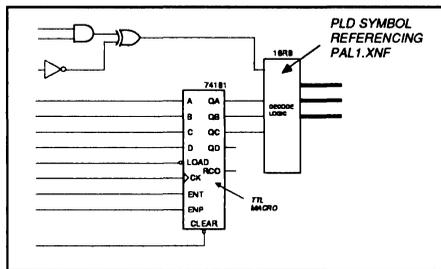
The automatic placement and routing program (APR) is very flexible. Through placement directives the designer can control the placement process to achieve best placement for a particular design. Routing resources can be specified to eliminate clock skews and minimize routing delays for critical paths. The result is faster product development.

After processing a design with the ADI software, the XC-DS21 XACT Design Editor may be used to interactively edit the placement and routing of critical areas of the design. XACT is then used to compile the placed and routed design into a configuration bitstream for programming LCAs.

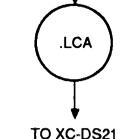
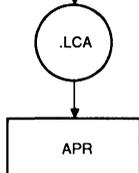
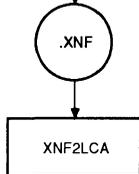
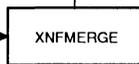
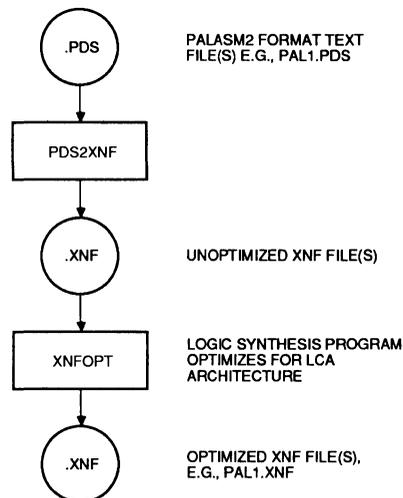
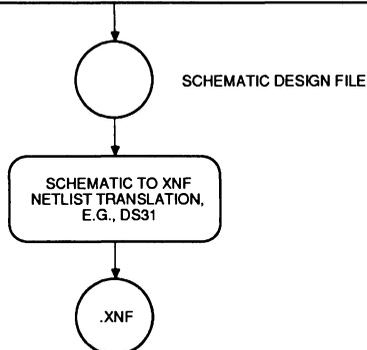
Xilinx provides ongoing support for users of the Automated Design Implementation software. For the first year, software updates are included. After that, the user may purchase the XC-SC23 Annual Support Agreement to continue to receive the latest software releases. Xilinx also maintains a Technical Hotline and a user electronic bulletin board to provide timely product support.

**SCHEMATIC DESIGN**

**PALASM2™ FORMAT/  
BOOLEAN EQUATION DESIGN**



1137 10



○ = FILE

□ = PROGRAM INCLUDED IN XC-DS23

1137 15



# XCDS26, XCDS27, XCDS28 XACTOR™ In-Circuit Design Verifier

## Product Brief

### FEATURES

- Real time in-circuit verification in user's target system
- Concurrent emulation of up to four devices
- Readback and display of Logic Cell™ Array internal storage element states
- Device status display with automatic update of asynchronous events
- Control and I/O pin isolation from target system
- Support for daisy chain programming of up to seven devices in a daisy chain
- Support for multiple device and package types
- Runs on an IBM® PC/AT™ or compatible personal computer

### GENERAL

The XACTOR™ real-time in-circuit design verifier provides interactive target system emulation of up to four Logic Cell Arrays from the host PC system. In-circuit debugging provides a powerful productivity enhancement to simulation, providing capabilities to verify functionality in the target system at full speed with all other circuits and system software.

The design verifier is composed of a microcomputer-based controller, and from one to four universal emulation pods, each with an emulation header. The controller is connected to the host PC through a serial port and provides local storage of configuration programs, control of individual device configurations, and control of the isolation of the pod device(s) from the target system. The user can set the state and isolation for each of the control signals to provide debugging of target hardware. Four general I/O pins are available to provide test points which may also be isolated from the target system.

Target Logic Cell Arrays can be programmed individually or in a daisy chain. Daisy chains of up to seven devices may be supported from any of the four pods. Individual

device isolation and configuration is controlled with mouse or keyboard commands and may be supplemented with user-defined setup files for easy system debugging.

Readback of device configuration may be performed on command for verification of the configuration process and interrogation of the internal states. The state of all internal storage elements is displayed after readback has been performed. Status displays showing the state of all isolation switches and control signal states are provided. The status display includes automatic reporting of asynchronous status changes in the target system.

### UNIVERSAL IN-CIRCUIT EMULATOR PODS

Additional pods may be connected to the XACTOR controller, up to a maximum of four pods per controller. Pod headers are interchangeable for different device and package types. Each pod provides a direct in-socket connection without disruption of the target system. Test points are provided to allow connection of a logic analyzer or other test equipment to aid in the system debugging.



## Product Brief

### FEATURES

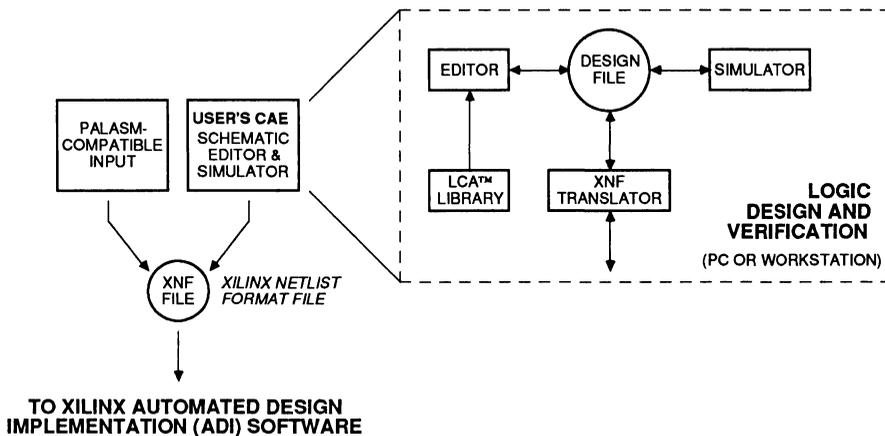
- Open development system allows Programmable Gate Array design entry with existing schematic editors
- Macro library of over 100 elements derived from XACT™ macro library
- Netlist translator to Xilinx Netlist Format (XNF) output
- User control for flagging critical paths in XC-DS23 Automatic Design Implementation (ADI) software
- Libraries and translators available from Xilinx:
  - XC-DS31 FutureNet DASH
  - XC-DS32 Schema II
  - XC-DS33 Daisy ACE or DED II
  - XC-DS34 Mentor IDEA
  - XC-DS35 OrCAD SDT
  - XC-DS3\_ Others under development
- Schematic editors, libraries and translators available from CAE vendor:
  - Valid, CASE, PCAD, VIEWlogic, Others

### GENERAL

Schematics are the most widely used method for describing and documenting designs. Design entry with schematic capture shortens product development time. Using familiar tools minimizes the time to market for programmable gate array designs.

Xilinx works with CAE vendors so that programmable gate array designs can be entered on a wide variety of schematic editors. Schematic interface packages consist of a library of primitives and macros, and a translator from the vendor's netlist format to the Xilinx Netlist Format (XNF). In some cases the interface package is sold by Xilinx (XC-DS-3X), and in others by the CAE vendor. Contact Xilinx for an up-to-date listing of existing interface packages.

### Xilinx Programmable Gate Array Design Entry Options





# XC-DS31 FutureNet DASH™ Schematic Designer Library

---

## Product Brief

---

### FEATURES

- Library and translator for users with the FutureNet DASH™ Schematic Designer
- Macro library of over 100 standard logic family equivalents derived from the XACT™ Macro Library
- Library of logic symbols including all two-input, three-input and four-input AND, OR and XOR gates plus storage, input/output and clock elements
- User control for flagging critical paths for the XC-DS23 Automatic Placement and Routing Program
- Converts schematic drawings to a Xilinx Netlist Format (XNF) output file
- Output compatibility with XC-DS23 Automated Design Implementation Program
- Runs on an IBM® PC/AT™ or compatible personal computer

### GENERAL

Schematic entry and automatic partitioning of Logic Cell Array designs shortens logic reduction and product development times. Complex designs can be specified schematically and quickly implemented for in-circuit design verification.

Xilinx's FutureNet DASH Schematic Designer Library provides the symbol library and conversion utility to permit designers to enter Logic Cell Array designs with the FutureNet DASH Schematic Designer. The Xilinx library provides the logic, I/O, and macro symbols to be used in

the schematic. A Xilinx conversion utility converts the schematic into an XNF output file.

Once partitioned, the design may be placed and routed with the PC-based XC-DS23 Automated Design Implementation Program. The Xilinx symbol library includes symbols to flag critical data and clock signals which the Automatic Placement and Routing Program uses to prioritize those signals for minimum delay.

Xilinx provides ongoing support for users of the FutureNet DASH Schematic Designer Library. For the first year, software updates are included. After that, the user may purchase the XC-SC31 Annual Support Agreement to continue to receive the latest software releases.



## Product Brief

### FEATURES

- Library and translator for users with the Schema II Schematic Editor
- Macro library of over 100 standard logic family equivalents derived from the XACT™ Macro Library
- Library of logic symbols including all two-input, three-input and four-input AND, OR and XOR gates plus storage, input/output and clock elements
- User control for flagging critical paths for the XC-DS23 Automatic Placement and Routing Program
- Converts schematic drawings to a Xilinx Netlist Format (XNF) output file
- Output compatibility with XC-DS23 Automated Design Implementation Program
- Runs on an IBM® PC/AT™ or compatible personal computer

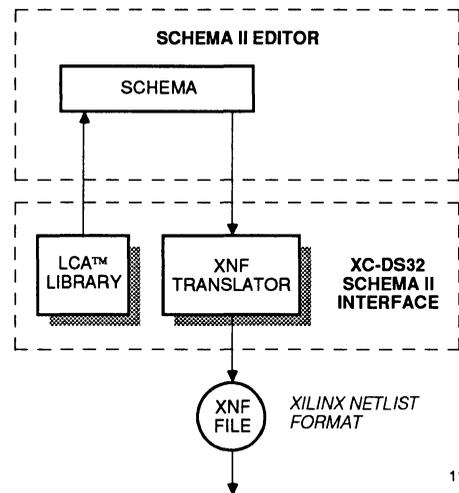
### GENERAL

Schematic entry and automatic partitioning of Logic Cell Array designs shortens logic reduction and product development times. Complex designs can be specified schematically and quickly implemented for in-circuit design verification.

Xilinx's Schema II Schematic Entry Interface provides the symbol library and conversion utility to permit designers to enter Logic Cell Array designs with the Schema II Schematic Editor. The Xilinx library provides the logic, I/O, and macro symbols to be used in the schematic. A Xilinx conversion utility converts the schematic into an XNF output file.

Once partitioned, the design may be placed and routed with the PC-based XC-DS32 Automated Design Implementation Program. The Xilinx symbol library includes symbols to flag critical data and clock signals which the Automatic Placement and Routing Program uses to prioritize those signals for minimum delay.

Xilinx provides ongoing support for users of the Schema II Schematic Entry Interface. For the first year, software updates are included. After that, the user may purchase the XC-SC32 Annual Support Agreement to continue to receive the latest software releases.



**To Xilinx XC-DS23 Automated Design Implementation Software**



# XC-DS33 Daisy Schematic Entry Interface with Unit-Delay Simulation

---

## Product Brief

---

### FEATURES

- Schematic entry via the Daisy DED II and ACE Schematic editors
- Unit-delay simulation with Daisy simulator
- Macro library of over 100 standard logic family equivalents derived from the XACT™ Macro Library
- Library of logic symbols including all two-input, three-input and four-input AND, OR and XOR gates plus storage, input/output and clock elements
- User control for flagging critical paths for the XC-DS23 Automatic Placement and Routing Program
- Converts schematic drawings to a Xilinx Netlist Format (XNF) output file
- Output compatibility with XC-DS23 Automated Design Implementation Program

### GENERAL

Schematic entry and automatic partitioning of Logic Cell Array designs shortens logic reduction and product development times. Complex designs can be specified schematically and quickly implemented for in-circuit design verification.

Xilinx's Daisy schematic interface provides the symbol library and conversion utility to permit designers to enter Logic Cell Array designs with either DED II or ACE on Daisy workstations. The Xilinx library provides the logic, I/O, and macro symbols to be used in the schematic. A Xilinx utility converts the Daisy format to an XNF output file.

Once partitioned, the design may be placed and routed with the PC-based XC-DS23 Automated Design Implementation Program. The Xilinx symbol library includes symbols to flag critical data and clock signals which the Automatic Placement and Routing Program uses to prioritize those signals for minimum delay.

Xilinx provides ongoing support for users of the Daisy Schematic Interface. For the first year, software updates are included. After that, the user may purchase the XC-SC33 Annual Support Agreement to continue to receive the latest software releases.

### DAISY SYSTEM REQUIREMENTS

Any Daisy workstation listed on the chart on the next page can be used to produce a design netlist in the Xilinx Netlist Format using a utility provided with the library.

An IBM PC/AT equipped with the XC-DS23 Automated Design Implementation Program and the XC-DS21 XACT Design Editor is then used to physically implement the design in an LCA.

PC/AT-based Daisy workstations (such as the Daisy ENTRY system) can have the XC-DS21 and XC-DS23 software installed in the PC's DOS partition. Designers using non-PC/AT workstations can transfer the schematic netlist to an MSDOS floppy and complete the LCA physical design on a PC/AT computer.

*Personal LOGICIAN® and LOGICIAN® are registered trademarks of Daisy Systems Corporation.*

## DAISY HARDWARE CONFIGURATIONS

SOFTWARE	DAISY LOGICIAN V,D	DAISY PERSONAL LOGICIAN* with CMG**	DAISY PERSONAL LOGICIAN* with CMG** and BMG***	DAISY ENTRY* SYSTEM with EGA
Schematic Capture—DEDII	X	X	X	X
Schematic Capture—ACE			X	X
DANCE/ADANCE	X	X	X	X
DRINK	X	X	X	X
LCA Netlist Extract—DRW2XNF	X	X	X	X
DOS File Format Conversion—NLINE	X	X	X	X
DOS File Format Conversion—DOSCOPY		X	X	X
Daisy Local Area Network—Daisy Ethernet	X	X	X	X
Xilinx PC/AT-Based Development Tools (Logic Reduction, Auto Place and Route, XACT, Download Cable)			X	X

\* IBM™ PC-AT™ based system.

\*\* CMG = Standard Daisy Character Map Graphics Card.

\*\*\* BMG = Personal Logician Bit Mapped Graphics Card.



# XC-DS34 Mentor Interface for Schematic Entry and Simulation

## Preliminary Product Brief

### FEATURES

- Mentor IDEA® Station can be used for schematic entry and unit-delay simulation of programmable gate array designs
- Macro library includes over 100 standard logic elements (counters, multiplexers, registers, etc.)
- Primitive library includes flip-flops, latches, AND, OR, XOR, NAND, NOR gates
- Xilinx Netlist Format (XNF) output is compatible with XC-DS23 Automated Design Implementation software and XC-DS21 XACT Design Editor

### GENERAL

The Xilinx XC-DS34 Mentor Interface allows designers to use their Mentor IDEA Station to enter and simulate programmable gate array schematics. The Logic Cell™ Array library includes over 100 macros for commonly used counters, registers, multiplexers, etc.

The resulting Mentor design files are then processed using Xilinx's XC-DS23 and XC-DS21 software, which

permits rapid design development and real-time in-circuit design verification.

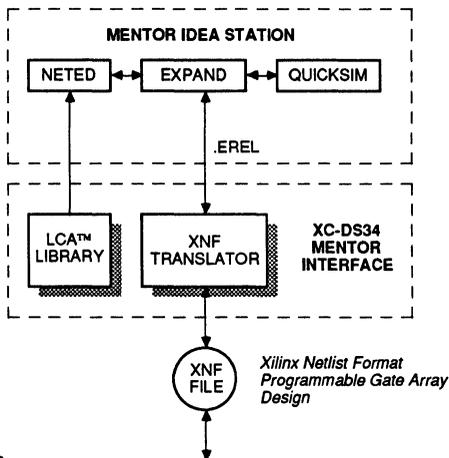
Xilinx provides ongoing support for users of the Mentor Interface. For the first year, software updates are included. After that, the user may purchase the XC-SC34 Annual Support Agreement to continue to receive the latest software releases. Xilinx also maintains a Technical Hotline and an electronic bulletin board to provide technical product support to LCA users.

### SYSTEM REQUIREMENTS

Mentor IDEA V6.0 or greater is required for schematic entry and simulation. The XC-DS34 Mentor Interface software requires the APOLLO SR9.6 or DOMAIN/IX operating system.

File transfer from the Mentor system to other PC-based Xilinx software may be accomplished using any existing PC/Apollo link, such as the DOMAIN/PCI software available from Mentor and Apollo.

### DEVELOPMENT SYSTEM PLATFORMS



1137 19

To Xilinx XC-DS23 Automated Design Implementation Software

	IBM PC	APOLLO
<b>XC-DS34 Mentor Interface</b>		
LCA Library for Mentor IDEA		✓
Mentor/XNF netlist translation		✓
Functional simulation (unit delay)		✓
Timing simulation (worst case)		Note 1
<b>XC-DS23 Automated Design Implementation</b>		
PLD equation-to-XNF translator	✓	
Logic synthesis optimization	✓	
Partition XNF design into LCA	✓	
Automatic placement and routing	✓	
<b>XC-DS21 XACT Design Editor</b>		
XACT placement/routing editor	✓	
Design rules check program	✓	
LCA bitstream compiler	✓	
Download cable (debugging)	✓	

Notes:

(1) Under development; will be provided as a separate software product when released

## Product Brief

### FEATURES

- Library and translator for users with the OrCAD SDT Schematic Editor
- Macro library of over 100 standard logic family equivalents derived from the XACT™ Macro Library
- Primitive library of logic symbols includes all two-input, three-input and four-input AND, OR and XOR gates plus storage, input/output and clock elements
- User control for flagging critical paths for the XC-DS23 Automatic Placement and Routing Program
- Converts schematic drawings to a Xilinx Netlist Format (XNF) output file
- Output compatibility with XC-DS23 Automated Design Implementation Program
- Runs on an IBM® PC/AT™ or compatible personal computer

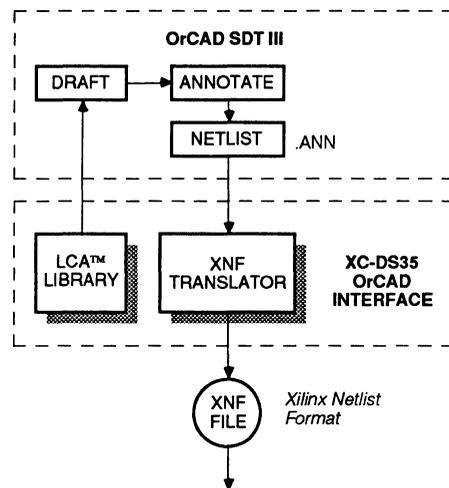
### GENERAL

Schematic entry and automatic partitioning of Logic Cell Array designs shortens logic reduction and product development times. Complex designs can be specified schematically and quickly implemented for in-circuit design verification.

Xilinx's OrCAD Schematic Entry Interface provides the symbol library and conversion utility to permit designers to enter Logic Cell Array designs with the SDT Schematic Editor. The Xilinx library provides the logic, I/O, and macro symbols to be used in the schematic. A Xilinx conversion utility converts the schematic into an XNF output file.

Once partitioned, the design may be placed and routed with the PC-based XC-DS23 Automated Design Implementation Program. The Xilinx symbol library includes symbols to flag critical data and clock signals which the Automatic Placement and Routing Program uses to prioritize those signals for minimum delay.

Xilinx provides ongoing support for users of the OrCAD Schematic Entry Interface. For the first year, software updates are included. After that, the user may purchase the XC-SC35 Annual Support Agreement to continue to receive the latest software releases.



**To Xilinx XC-DS23 Automated Design Implementation Software**



# XC-DS40 7400 TTL Library for FutureNet DASH and DASH-LCA

## Product Brief

### FEATURES

- Over 50 7400 TTL library elements for use with FutureNet DASH™ Schematic Designer
- Permits XC3000 family Logic Cell™ Array design using familiar 7400 MSI functions
- Supports XC-DS53, XC-DS54 PGA Development Systems with FutureNet DASH-LCA

### GENERAL

The XC-DS40 7400 TTL Library is a set of over 50 FutureNet DASH logic symbols for schematic entry of Logic Cell Array (LCA™) designs. The library can be used with the XC-DS53 and XCDS-54 Development Systems, or with the FutureNet DASH Schematic Designer. Use of the XC-DS40 Library allows the designer to enter complex programmable gate array designs in terms of familiar TTL functions.

The XC-DS40 TTL Library supplements the library of more than 100 logic elements that are included with XC-DS53 and XC-DS54.

Xilinx provides ongoing support for XACT users. For the first year, free software updates (such as support for new devices) are included. After that, users may purchase the XC-SC40 to continue receiving software updates. In addition, Xilinx operates a technical support telephone hotline and an electronic bulletin board to provide software enhancements and factory support.

### LIST OF TTL MACROS

7442	4-to-10 Line Decoder	74151	8-Input Multiplexer
7448	BCD-to-Seven-Segment Decoder	74152	8-Input Multiplexer
7477	4-Bit Latch	74153	Dual 4-Input Multiplexer
7483	4-Bit Binary Adder With Fast Carry	74154	1-of-16 Decoder/Demultiplexer
7485	4-Bit Magnitude Comparator	74157	Quad 2-Input Multiplexer
74125	Quad Three-State Bus Buffer	74158	Quad 2-Input Multiplexer
74138	1-of-8 Decoder/Demultiplexer	74160	Presettable Decade Counter
74139	Dual 1-of-4 Decoder	74161	Presettable Binary Counter
74147	10-to-4 Line Priority Encoder	74162	Presettable Decade Counter w/ Sync Clear
74148	8-to-3 Line Priority Encoder	74163	Synchronous Binary Counter w/ Sync Clear
		74164	Serial-In, Parallel-Out Shift Register
		74166	Parallel Load 8-Bit Shift Register
		74168	4-Bit BCD Synchronous Up/Down Counter
		74169	4-Bit Binary Synchronous Up/Down Counter
		74174	Hex D Flip-Flop with Master Reset
		74179	4-Bit Parallel Access Shift Register
		74194	4-Bit Bidirectional Universal Shift Register
		74195	4-Bit Parallel Access Shift Register
		74198	8-Bit Bidirectional Shift Register
		74199	8-Bit Shift Register with Clock Inhibit
		74240	Octal Inv Buffer, Three-State Outputs
		74241	Octal Non-Inverting Three-State Buffers
		74244	Octal Non-Inv Buffer, Three-State Outputs
		74245	Octal Bidirectional Transceiver
		74257	Quad 2-Input Multiplexer
		74258	Quad 2-Input Multiplexer
		74259	8-Bit Addressable Latch
		74273	Octal D Flip-Flop
		74278	4-Bit Cascadable Priority Register
		74280	9-Bit Parity Checker/Generator
		74283	4-Bit Binary Full Adder
		74298	Quad 2-Input Multiplexer with Storage
		74352	Dual 4-to-1 Data Selector/Multiplexer
		74373	Octal Latch with Three-State Outputs
		74374	Octal D Flip-Flops with Three-State Outputs
		74377	Octal D Flip-Flop with Clock Enable
		74390	Dual 4-Bit Decade Counters with Clear
		74393	Dual 4-Bit Binary Counters with Clear
		74518	8-Bit Identity Comparator
		74521	8-Bit Identity Comparator
		74540	Octal Inv Buffer, Three-State Outputs
		74541	Octal Non-Inv Buffer, Three-State Outputs
		74577	Octal D Flip-Flops with Reset and Three-State Outputs
		74590	8-Bit Binary Counter with Output Register
		74595	8-Bit Shift Register with Output Register



# XC-DS53, XC-DS54 PGA Development Systems with FutureNet DASH-LCA

## Product Brief

### FEATURES

- Complete development systems for programmable gate array design entry and implementation using schematics and/or Boolean equations
- FutureNet DASH™-LCA schematic editor provides easy-to-use hierarchical LCA design capability
- PALASM™-format text input permits combining PLD equations and schematics in the same design
- Extensive macro library of over 100 logic elements (PAL™ devices, TTL counters, registers, etc.)
- Logic optimization software uses logic synthesis algorithms to optimize programmable gate array designs for density or performance
- Automatic logic reduction and partitioning software removes unused, disabled logic
- Automatic placement and routing
- XACT Design Editor provides interactive physical editing, timing calculator, design consistency checking, and configuration bitstream compiling
- Download cable transfers configuration bitstreams directly from PC to LCA during design debugging
- Runs on IBM® PC/AT™ or compatible computer

### GENERAL

The XC-DS53 PGA Development System provides designers with a complete system for entering, implementing, compiling, and debugging designs for Xilinx programmable gate arrays. The XC-DS54 contains the design entry and design implementation software for designers who already have the XC-DS21 XACT Design Editor.

The FutureNet DASH-LCA schematic editor combined with a Xilinx PALASM-to-LCA translator program provides powerful design entry and documentation capability. Designs can be entered as schematics, Boolean equations, or both.

DASH-LCA supports unlimited levels of hierarchy and an extensive library of standard logic macros for LCA design. In addition, designers can flag critical timing paths to ensure critical signals are routed with minimum delay. The optional XC-DS40 TTL library adds over 50 additional 7400 MSI TTL functions for LCA design using FutureNet DASH-LCA or DASH.

Xilinx PALASM-to-LCA software permits text files containing Boolean equations for *any type* of PLD to be translated

into an LCA, or automatically linked to PAL symbols in a schematic.

The XC-DS53 and XC-DS54 also include the XC-DS23 Automated Design Implementation software, which permits designers to efficiently implement programmable gate array designs without detailed knowledge of LCAs. The logic optimization software uses logic synthesis algorithms to optimize for smallest area or highest performance. Automatic logic reduction and partitioning software eliminates unused logic, then partitions the design into Logic Cell Array physical resources (logic and I/O blocks). Finally, the Automatic Placement and Routing program is invoked to arrange logic blocks within the LCA to minimize signal delays on user-specified critical timing paths.

Designers can interactively place and route portions of the physical design, if necessary, then compile a configuration bitstream using the graphics-based XC-DS21 XACT Design Editor (included in the XC-DS53). XACT's timing calculator permits point-to-point timing calculations for critical path analysis, or a simulator (such as the optional XC-DS22 SILOS Simulator) can be used to simulate design function and timing.

During in-circuit design debug, designers can save time by using a Download Cable provided with XACT to transfer the configuration program from the PC directly into an LCA under development in a system.

Xilinx provides ongoing support for XACT users. For the first year, free software updates (such as support for new devices) are included. After that, users may purchase the XC-SC53 or XC-SC54 Annual Support Agreement to continue receiving software updates. In addition, Xilinx operates a technical support telephone hotline and an electronic bulletin board to provide software enhancements and factory support.

XC-DS53	XC-DS54	Includes:
√	√	FutureNet DASH-LCA Schematic Editor and LCA Library
√	√	Complete XC-DS23 Automated Design Implementation software (PALASM input, Logic optimizer, Automatic placement and routing)
√		Complete XC-DS21 XACT Design Editor (Interactive placement and routing, Configuration bitstream compiler, Download Cable, Demo Board)



# XC-DS81 Configuration PROM Programmer

## Product Brief

### FEATURES

- Programs XC1736 Serial Configuration PROM
- Connects to serial port of IBM PC/AT or compatibles
- Operates from PC via software provided with programming unit
- Accepts HEX format data files created by the XC-DS21 XACT Programmable Gate Array Design Editor

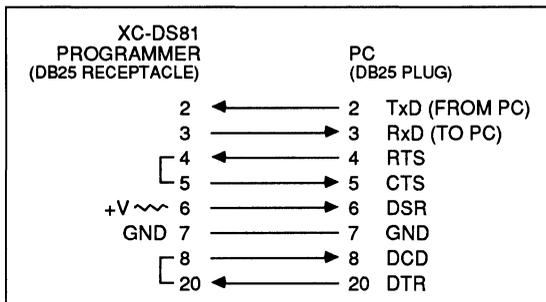
### GENERAL

Designers using the Xilinx XC1736 Serial Configuration PROM—an 8-pin mini-DIP PROM used to configure programmable gate arrays—can program the XC1736 with the XC-DS81 Configuration PROM Programmer.

The programming unit connects to a serial port of an IBM PC/AT or compatible and is controlled using PC-based software included with the XC-DS81.

Designers compile their Logic Cell Array designs into a standard HEX format file using the PC-based XACT development system. The programming software provided with the XC-DS81 is then used to download the HEX file into the programming unit and program a XC1736.

Since the XC-DS81 software is separate from the XACT software, the XC-DS81 software and programming unit can be installed on a PC other than the one used for design development—such as a PC located in a manufacturing area.



360 01

XC-DS81 Interface to PC





# Development Systems Ordering Information

Further information is available from your local distributor sales office or the nearest Xilinx sales representative.

Part Number	Description
<b>XACT Development Systems</b>	
XC-DS21	XACT™ Design Editor
XC-DS22	PC-SILOS™ Simulator
XC-DS23	Automated Design Implementation
XC-DS24*	XACTOR In-Circuit Design Verifier:
XC-DS26	XACTOR Universal Emulation Pod*
XC-DS27 -PD48	XACTOR Emulation Header for 48-pin DIP*
PC68	XACTOR Emulation Header for 68-pin PLCC*
PG68	XACTOR Emulation Header for 68-pin PGA*
PC84	XACTOR Emulation Header for 84-pin PLCC*
PG84	XACTOR Emulation Header for 84-pin PGA
PG175	XACTOR Emulation Header for 175-pin PGA
XC-DS28	XACTOR Controller Hardware and Software*
XC-DS31	FutureNet DASH™ Schematic Designer Library
XC-DS32	Schema II Schematic Entry Interface
XC-DS33	Daisy Schematic Entry Interface
XC-DS34	Mentor Schematic Entry Interface
XC-DS35	OrCAD Schematic Entry Interface
XC-DS40	7400 TTL Library for FutureNet DASH and DASH-LCA
XC-DS53	PGA Development System with FutureNet DASH-LCA
XC-DS54	PGA Logic Design and A.D.I. with FutureNet DASH-LCA
XC-DS81	Configuration PROM Programmer
<b>Evaluation Kit</b>	
XC-EK01	Logic Cell Array Evaluation Kit
<b>Service Contracts</b>	
XC-SC21	XACT™ Design Editor System Annual Support Agreement
XC-SC23	Automated Design Implementation Annual Support Agreement
XC-SC31	FutureNet DASH Schematic Designer Library Annual Support Agreement
XC-SC40	7400 TTL Library for FutureNet Annual Support Agreement
XC-SC32	Schema II Schematic Entry Interface Annual Support Agreement
XC-SC33	Daisy Schematic Entry Interface Annual Support Agreement
XC-SC34	Mentor Schematic Entry Interface Annual Support Agreement
XC-SC35	OrCAD Schematic Entry Interface Annual Support Agreement
XC-SC53	PGA Development System Annual Support Agreement
XC-SC54	PGA Logic Design and A.D.I. Annual Support Agreement

\* When ordering the XACTOR In-Circuit Design Verifier:  
*Order one DS28 XACTOR Controller, one to four DS26 Emulation Pods, and one DS27 Emulation Header (of desired package type) per Emulation Pod.*

# DEVELOPMENT SYSTEM OPTIONS

	DS-53	DS-54	DS-23	DS-21	DS-22	DS-28	DS-81	OPTIONAL SCHEMATIC LIBRARIES
FUTURENET DASH-LCA EDITOR AND LIBRARY	<input type="checkbox"/>	<input type="checkbox"/>						
PALASM INPUT PLUS LOGIC SYNTHESIS	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>					
AUTO PLACE & ROUTE	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>					
PHYSICAL DELAY EXTRACTION	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>					
XACT DESIGN EDITOR AND BITSTREAM COMPILER	<input type="checkbox"/>			<input type="checkbox"/>				
	OPTIONAL							
FUTURENET DASH-LCA AND DASH SCHEMATIC LIBRARY								<input type="checkbox"/> DS-31
TTL LIBRARY FOR FUTURENET DASH AND DASH-LCA								<input type="checkbox"/> DS-40
SCHEMA II SCHEMATIC ENTRY LIBRARY								<input type="checkbox"/> DS-32
DAISY SCHEMATIC ENTRY LIBRARY								<input type="checkbox"/> DS-33
MENTOR SCHEMATIC ENTRY LIBRARY								<input type="checkbox"/> DS-34
ORCAD SCHEMATIC ENTRY LIBRARY								<input type="checkbox"/> DS-35
PC-SILOS SIMULATOR					<input type="checkbox"/>			
XACTOR IN-CIRCUIT DESIGN VERIFIER						<input type="checkbox"/>		
XC1736 CONFIGURATION PROM PROGRAMMER							<input type="checkbox"/>	

1137 12

## SYSTEM REQUIREMENTS

### Recommended System Configuration

IBM PC/AT or fully compatible (Note 1) computer with:

- MSDOS 3.0 or higher
- 640K Bytes base system RAM
- 1.5M Bytes of additional extended memory (Note 2)
- 1 5.25" Diskette Drive
- 20 MB Hard Disk
- IBM compatible Enhanced Graphics Adapter (EGA) and Display (Note 3)
- 1 Serial Interface Port (Note 4)
- 1 Parallel Interface Port
- Mouse System™, Microsoft®, Logitech, or compatible mouse

### Configuration Notes:

1. 100% IBM compatibility is required. A partial list of known compatible machines includes IBM, Compaq,

AST Premium, and most 80386-based machines. Contact Xilinx for an up-to-date list of compatible machines.

2. Current (XACT 2.0) requirements are a total of 2.0M Bytes for 2000 gate devices. Additional extended memory is required for larger devices to get the following totals:

XC3030	2.75 M Bytes
XC3042	3.5 M Bytes
XC3064	4.75 M Bytes
XC3090	6.0 M Bytes

3. The lower-resolution IBM-compatible Color Graphics Adapter (CGA) and display is also supported, but not recommended.

4. Systems with both a serial-port mouse device and an XC-DS28 XACTOR option require a second serial interface port for XACTOR.

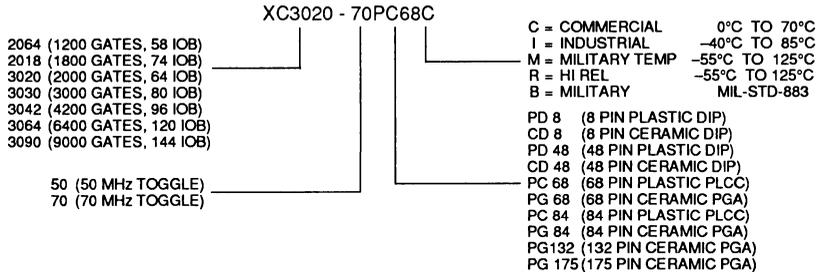


# Component Ordering Information

Further information is available from your local distributor sales office or the nearest Xilinx sales representative.

## PROGRAMMABLE GATE ARRAYS

Part numbers are composed as follows:



1137 13

		48 PIN		68 PIN		84 PIN		132 PIN	175 PIN	CERAMIC QUAD FLAT PACK
		PLASTIC DIP	CERAMIC DIP	PLASTIC PLCC	CERAMIC PGA	PLASTIC PLCC	CERAMIC PGA	CERAMIC PGA	CERAMIC PGA	
		-PD 48	-CD 48	-PC 68	-PG 68	-PC 84	-PG 84	-PG 132	-PG 175	
XC2064	-50	C	I	C I	C I M B					UNDER DEVELOPMENT
	-70			C	C					
XC2018	-50			C I		C I	C I M R B			
	-70			C		C	C			
XC3020	-50			C I		C I	C I M B			
	-70			C I		C I	C I			
XC3030	-50					C I	C I M B			
	-70					C	C			
XC3042	-50					C I	C I M	C I M B		
	-70					C	C	C		
XC3064	-50							C I M B		
	-70							C		
XC3090	-50								C I M B	
	-70								C	

1137 14

### Package and Temperature Options

## SUPPORT DEVICES

Part Number	Description
XC1736PD8C/CD8M	36K X 1 Serial Configuration PROM



The Programmable Gate Array Company



- 1 Programmable Gate Arrays**
- 2 Product Specifications**
- 3 Quality, Testing, Packaging**
- 4 Technical Support**
- 5 Development Systems**

## **6 *Applications***

- 7 Article Reprints**
  - 8 Index**
-

Introduction .....	6-1
Estimating Size and Performance .....	6-3
Counting Gates .....	6-7
Incorporating PLD Equations into LCAs .....	6-9
The XC2000 User's Guide to the XC3000 Family .....	6-13
Additional Electrical Parameters .....	6-14
LCA Performance .....	6-16
Metastable Recovery .....	6-19
Majority Logic, Parity .....	6-20
Multiple Address Decoding .....	6-21
Adders and Comparators .....	6-22
Building Latches Out of Logic .....	6-25
Synchronous Counters, Fast and Compact .....	6-26
40 MHz Presettable Counter .....	6-27
Frequency Counter .....	6-29
Serial Code Conversion .....	6-30
Serial Pattern Detectors .....	6-32
8-Bit Format Converter .....	6-33
Megabit FIFO in Two Chips .....	6-35
State Machine Design .....	6-37
Self-Diagnosing Hardware .....	6-38
PS/2 Micro Channel Interface .....	6-41
High Speed Bar Code Reader .....	6-43
DRAM Controller .....	6-45
Logic Analyzer/In-Circuit Emulator .....	6-51

---

## INTRODUCTION

The following pages show examples of systems and sub-systems solutions using Xilinx Programmable Gate Arrays. Some of these designs have been implemented, a few are in production, but most are conceptual designs. These are intended to demonstrate the devices' capabilities, to highlight special advantage, to emphasize the best design methods, and in general to stimulate the designer's imagination.

A Xilinx programmable gate array can implement virtually any digital design. Xilinx offers a software package that covers the gamut from schematic capture through logic optimization to automatic place and route and to the generation of a programming bit stream. The designer can use these tools and achieve a working LCA design while paying very little attention to the architectural details of the Xilinx programmable gate array.

Such an approach, however, will not always achieve the highest possible performance and the lowest possible cost. For specific, well-structured designs it may pay to work out a good match with the LCA architecture. This chapter gives several examples of such solutions. The XC2000 and XC3000 family programmable gate arrays have inherent features different from those of LSTTL MSI circuits, or PAL devices, or conventional gate arrays. These four technologies all have different structures which lead to different strengths and weaknesses when they are being used to implement any specific type of logic.

**TTL-MSI** was originally defined to fit into a 16-pin package and to provide a maximum of flexibility, so that each standard part could be used in a myriad of applications. Some functions are therefore overdesigned (counters and shift registers have parallel inputs and outputs, when few applications need both) and some are crippled by the 16-pin limitation (notably the up-down counters).

**PAL devices** suffer from the rigidity of their AND-OR architecture and from the fixed assignment of flip-flops to output pins. While the number of inputs is generous, ideal for wide decoding, the limited number of product terms that can be ORed together makes many designs inefficient and slow. The number of flip-flops available in PALs is very limited.

**Gate Arrays** offer maximum flexibility and a high level of integration, but burden the user with high risk, high cost, and a long delay from finished design to working prototype. Generating test vectors and worrying about testability is another price the gate array user has to pay.

**Programmable Gate Arrays** offer a very large number of flip-flops (128 in the CLBs and another 128 in the IOB of the 3020, a total of more than 800 in the 3090). Different from the situation with Gate Arrays, these LCA flip-flops cannot, or do not have to be, traded off against logic. Logic coexists with the flip-flops in the form of function generators. The function generators are surprisingly versatile pieces of logic, unlimited in their flexibility, limited only by their fan-in of four or five signals.

When the logic has five inputs or less and is interspersed with flip-flops driven by a common clock, the Xilinx Programmable Gate Arrays are extremely efficient. Certain high fan-in functions like ALUs tend to be less efficient, and bus-oriented designs have to be routed carefully to take advantage of the long lines and 3-state drivers of the 3000-series.

Fortunately, the user normally has some freedom in structuring the system design. Whenever possible, this freedom should be used to improve either the performance or the efficiency of the implementation.

## GENERAL TOPICS

Most designers want to estimate density and performance before they begin an LCA design, and some want to know the definition of equivalent gates. Finally, there is interest in converting existing PAL designs to LCAs. All these subjects are covered in the beginning of this chapter.

While the data sheets provide worst-case guaranteed parameters, many designers need additional information about input and output characteristics, power consumption, crystal oscillator design, and the exact interpretation of certain ac parameters. CLB flip-flops show excellent recovery from metastable problems, an important concern with asynchronous interfaces.

### COMBINATORIAL FUNCTIONS

The 5-input function generator of the XC3000 family CLBs offers unlimited flexibility to implement any one of the more than 4 billion ( $2^{32}$ ) possible functions of up to five variables in one CLB, all with the same combinatorial delay. The 4-input function generator in the XC2000 family can implement any one of the 64K ( $2^{16}$ ) possible functions of four variables. The logic designer should take advantage of this flexibility while avoiding the possible speed penalty imposed by the limitation to only five or four inputs. This may lead to logic partitioning that is different from traditional design or from MSI or PAL implementation.

Majority logic is just one example in which the CLB excels: A 5 input majority function would use 29 gates when implemented with 2-input NANDs and inverters, but it fits into the combinatorial portion of one 3000 series CLB.

Address decoding is the classical strength of PAL devices. It is done efficiently in LCAs if the complete function includes the combination of several addresses or groups of addresses.

ALUs consume many LCA resources, but adders or subtractors can be implemented quite efficiently, even using carry-look-ahead for functions that exceed a width of eight bits.

### SEQUENTIAL FUNCTIONS

LCAs offer an abundance of flip-flops, from 119 in the XC2064 to 928 in the XC3090. Each CLB flip-flop (64 in the XC2064, 128 in the XC3020, 640 in the XC3090) has a "free" combinatorial function generator available as its input. This simplifies the design of shift registers and counters.

The "Corner Bender" serial-parallel or parallel-serial converter design, is a two-dimensional shift register array that fits very efficiently into an XC2064 or half of an XC3020, with 100% utilization of the CLB flip-flops.

Using the fast flip-flops and distributed logic in the LCA to their best advantage, a synchronous presettable counter of arbitrary length has been demonstrated to run at 40 MHz. This is much faster than any available popular microprocessor peripheral counter/timer.

State Machine design is another example in which the creative use of CLB resources can result in a straightforward and easily understood solution.

As explained in the beginning of this chapter, the CLB flip-flops are "metastable-resistant," they resolve metastable situations typically within a few nanoseconds. Designers are nevertheless encouraged to avoid asynchronous designs whenever possible. The combination of very fast CLB flip-flops with relatively slow and layout-dependent interconnects can lead to internal decoding spikes and glitches that cannot be observed with an oscilloscope, but which can play havoc with internal asynchronous logic. The high-speed, low-skew global clock lines and the individual Clock Enable inputs on each CLB favor synchronous design approaches that are inherently safer and more predictable.

### SYSTEMS DESCRIPTIONS

LCAs are universal programming building blocks, that are used in a wide variety of systems.

An 8-digit frequency counter implemented in a 2064 is a simple illustration. A PS/2 Micro Channel Controller and a DRAM Controller/Error Corrector demonstrate the versatility of the LCA in speed-critical applications.

Article reprints from the trade press indicate the broad range of LCA applications.

The purpose of this applications chapter is not to provide cookbook solutions, but rather to stimulate the imagination, convey ideas and demonstrate that LCAs offer a better solution for a large variety of digital designs.

## INTRODUCTION

Programmable gate arrays are available in a range of densities and speed grades. Before committing resources to design implementation, the user should estimate which Programmable Gate Array best fits the specific application. Such size and performance estimates cannot be expected to provide exact details, but they provide useful guidelines for device selection and cost estimates. A complete design will always be the final test for both density and performance.

Design fit estimates can be done in two steps. The first is a quick I/O and storage element count, with no regard for performance. The second step counts logic blocks based on details of the intended circuit, and includes gross performance estimates, still without regard for routing delays. Performance estimates should always be considered "best-case," recognizing that actual system performance can only be verified on a completed design.

### STEP 1: I/O and Storage Element Fit

A quick initial estimate of the fit of a system into a Programmable Gate Array can be made by counting the required input and output pins and internal storage elements. Table 1 lists Xilinx's XC2000 and XC3000 series Logic Cell Array (LCA) devices and their respective I/O and storage element counts. To estimate a fit, first count the required inputs and outputs and compare the total with the I/O pin count of the desired device. If the desired functions require more I/O than listed for a device, one must either select a larger device or package, or reduce the I/O requirements.

Device	Maximum I/O	Logic Block Storage	I/O Block Storage
XC2064	58	58	58
XC2018	74	100	100
XC3020	64	128	128
XC3030	80	200	160
XC3042	96	288	192
XC3064	120	448	240
XC3090	144	640	288

Table 1. I/O and Storage Element Summary

If the desired programmable gate array has enough I/O pins, the next step is to count the required storage elements. Table 1 shows both logic block storage elements and I/O block storage elements. Logic block storage elements should be considered first, since they are the most flexible. If the number of storage elements required is less than the number of logic storage elements, the desired functions can probably be performed in the chosen Logic Cell Array.

In some cases, the I/O block storage elements can also be used to meet storage element requirements. In particular, if the number of additional storage elements required beyond the available logic storage elements is less than the number of unused I/O pins, then the desired functions may still fit into the chosen device.

The following two examples illustrate the Step One quick estimation procedure:

#### Example 1. An 8-bit microprocessor peripheral.

Function	I/O requirements
8 bit data bus	8
5 bus-control signals	5
16 bits of output	16
4 bits of output control	4
2 internal control registers	—
Interrupt control logic	—
<b>TOTAL</b>	<b>33</b>

Even the smallest Xilinx Logic Cell Array, the XC2064, passes the I/O test. It has 58 user I/O in its 68 pin PLCC package.

Function	Storage Elements
Control registers (assume 8 bits)	16
Buffered input shift register	16
Miscellaneous control logic	10
<b>TOTAL</b>	<b>42</b>

All of the storage elements can be put into logic storage in the XC2064. The XC2064 should fit this application, provided the desired performance can be achieved.

**Example 2. A memory controller for a 32 bit high performance processor.**

Function	I/O Requirements
32 bit processor data bus	32
32 bit processor memory bus	32
32 bit memory bus	16 (muxed)
32 bit control register	-
32 bit DMA control	-
Address multiplexing control	-
RAS/CAS/Refresh generation	3
Memory error check and correct	-
Processor and memory timing	10
<b>TOTAL</b>	<b>93</b>

Based on this I/O count, the XC3042 with 96 pins would be marginal. An XC3064 with up to 120 I/O pins may be required.

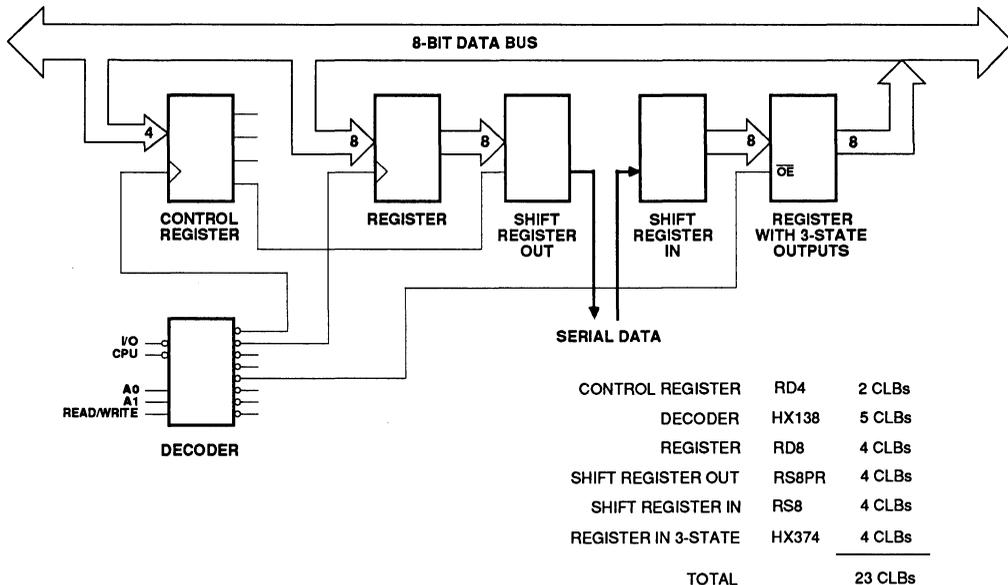
Function	Storage Elements
32 bit DMA (two-32 bit Counters)	64
Refresh generation (minimum)	10
32 bit control register	32
32 bit processor memory Address	32
Error check and correct	44
Miscellaneous control	20
<b>TOTAL</b>	<b>202</b>

With two storage elements per logic block, the XC3042 can provide up to 288 storage elements. Based on this estimate, the desired functions should fit into the device. Some caution is indicated for two reasons. First, the I/O count is very near the limit of the device. This could cause some routing congestion in the I/O area, making a higher pincount device a better choice. Second, high performance requires making the best use of a device's features. The 32-bit bus may impose critical performance requirements. Only the XC3064 and XC3090 allow a 32 bit internal bus, based on the number of available Long Lines. Choosing the XC3064 could address the I/O requirements as well as the performance needs.

**STEP 2: Logic Block Requirements**

After establishing design fit by counting I/O and storage elements, it may be necessary to make a more detailed analysis of the blocks required. The macro library summary table in the development system section of this data book may be used to determine specific Configurable Logic Block (CLB) counts for each function to be implemented.

The macro list shows the various gates and functions available with each design library. Each entry in the list includes the required number of logic blocks to implement that function. Where there are differences between XC2000 and XC3000 family block counts, these are noted. To develop a rough block count, the designer simply tabulates all of the blocks required by each of the functional elements in the design. Figure 1 shows a portion of a schematic and the block count from the macro list.

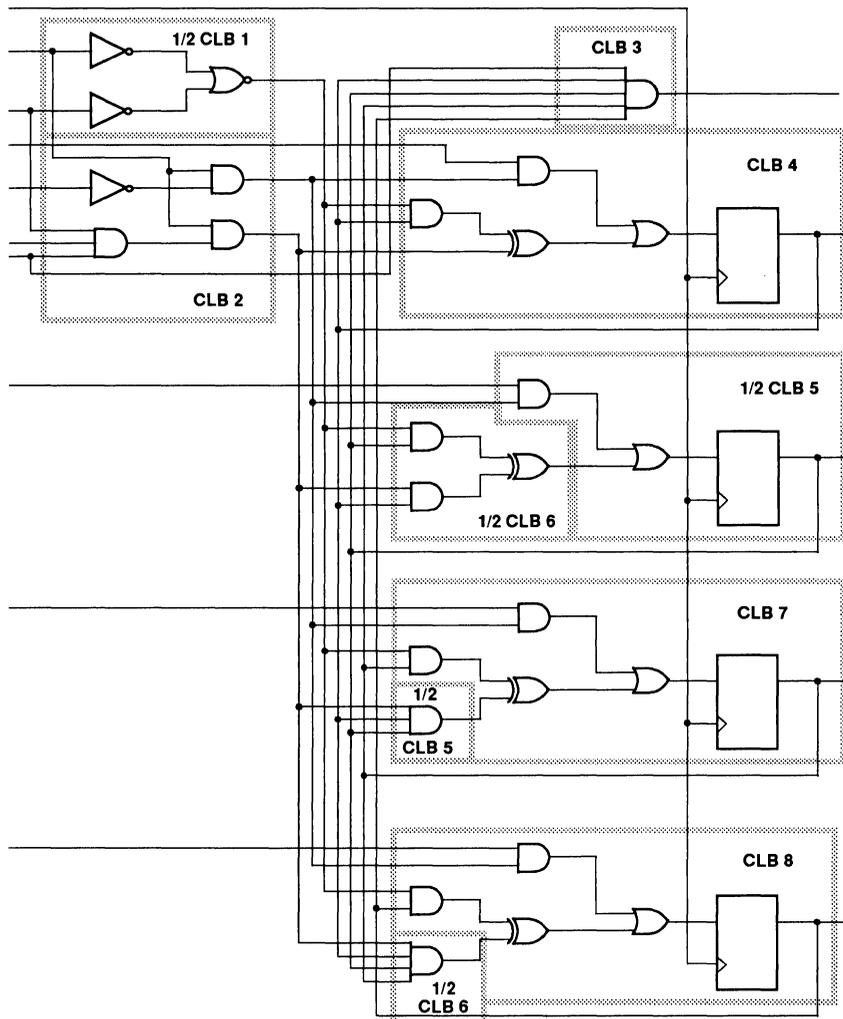


1133 01

In many schematics there are collections of random gates which need to be considered, along with the higher level functions such as counters, decoders and multiplexers. The following technique can be used to estimate the logic blocks required for random logic. Begin at an output point and move back along the path collecting gates until the number of inputs is 4 for XC2000 family devices, or 5 for XC3000 family devices. These gates can be marked in some way to show that they occupy a single logic block. Blocks identified by this method are added to the block count from the macro list analysis. Figure 2 shows an example of this gate collecting technique.

Estimating the block count for integrating PLD devices is more difficult. Each PLD output should be counted as at least one block. PLD devices using five or fewer of the inputs, will require only one block per output for the XC3000 family (four inputs for the XC2000 family). For complex equations using more than five (or four) inputs, a conservative estimate is to use three blocks per output pin.

Decisions about the appropriate device can be reviewed as more information is collected. Block count estimates which are near the limit of a device, either in block count or in I/O and storage element count, may suggest use of the next higher density device.



1133 02

**Figure 2. Gate Collecting Technique.** 7 1/2 CLBs are required to implement a 74-163 4-bit binary counter.

### Estimating Performance

After selecting the right programmable gate array based on logic resources, an estimation of performance is often the next step. If the system clock rate is less than 20% of the flip-flop toggle rate of the selected device, then the performance goals can usually be met easily. In cases of higher system clock rates or very complex functions, a more detailed analysis may be required.

The macro library for each device family includes the number of logic block levels used for each listed function, and the LCA datasheet specifies the block delay for each level.

Some routing delay must be added to the block delay. Such routing delays can add 25–50% to the block delays.

As an example, a circuit might have three levels of blocks in the path from one clock edge to another. For a device with 10 nsec block delays, this gives 30 nsec delay from the first clock to the setup required for the next clock.

Allowing 30% for routing (10 nsec) and 8 nsec for setup gives a total delay of 48 nsec. This should allow operation at a system clock rate of up to 20 MHz.

### Summary

The final determination whether a logic device meets the goals for integration and performance can come only after the design has been completed. For Programmable Gate Arrays, estimating logic capacity and performance should precede device selection. If the design fits, the programmable gate array development system, and the simplicity of in-system design verification allow cost effective and rapid design implementation.

Of course, specifications sometimes change during the execution of a design. Logic changes may result in different requirements for I/O and logic blocks. In such cases the Xilinx products product line simplifies the migration to a compatible array that meets the new requirements.

BY DAVE LAUTZENHEISER

Methods for evaluating the density of logic device families have evolved as logic technologies have evolved. With conventional SSI/MSI and similar families, it was easy to count the number of packages or devices required to implement a system function. With the advent of technologies which allow the integration of system functions into VLSI components, such simple device counting is no longer possible.

For these more highly integrated logic technologies the most common method of evaluating device densities is "gate count". For devices where the user logic is directly implemented in countable gates, this technique works very well. Most gate array density measurements are expressed as the number of equivalent two input NAND gates. In conventional gate array devices, the gates can be directly counted as usable gates, with some overhead for potential routing requirements or inefficiencies in logic library translation. By applying a utilization factor of 80% to 90%, users can easily size the logic device required for their system function.

In the various families of programmable devices available, programmable logic devices (PLDs) and programmable

gate arrays (PGAs), individual gates are not countable. For these types of devices, the architecture is not based on a unit of a gate, but is rather based on a larger functional element. The mapping of a particular system logic solution into these functional elements ultimately determines the density of the device. Simple density measures such as gate count may be misleading.

## GATE ARRAY DENSITY COMPARISONS

Since conventional and user programmable gate arrays are typically alternatives for the same logic applications, a common metric is needed for expressing logic density. Although logic is seldom designed at the gate level, the two-input gate remains the most widely used measure of logic capacity.

Mask-programmed gate arrays have a very simple architecture. Uncommitted transistors can be interconnected to form gates and higher level structures, and these equivalent gates can be counted directly. In order to provide the flexibility of user programmability, programmable gate arrays must modify the basic gate array

Element	Number of Elements	XC2064	XC3020	XC3090
Flip-flop	6	6 x 64 = 384	6 x 128 = 768	6 x 640 = 3840
Input Latch	4	4 x 58 = 232	4 x 64 = 256	4 x 144 = 576
Output FF	6	6 x 0 = 0	6 x 64 = 384	6 x 144 = 864
Three-State Output	3	3 x 58 = 174	3 x 64 = 192	3 x 144 = 432
Funct. Gen. (4 input)	15	15 x 64 = 960	15 x 128 = 1920	15 x 640 = 9600
Clock Drivers	4	4 x 2 = 8	4 x 2 = 8	4 x 2 = 8
<b>Total Gates</b>		1,758	3,528	15,320
<b>Nominal Gate Count</b>		1,200	2,000	9,000
<b>Usable Gates</b>				
50% Utilization		879	1,764	7,660
65% Utilization		1,143	2,293	9,958

Table 1. Total Gates

architecture. Programmable interconnection networks require more silicon resources than mask-programmed metal segments, and exhibit lower performance. To compensate for this, the “logic granularity” must be increased from individual gates to higher level programmable elements. These elements provide resources for both combinatorial logic and data storage, and can be programmed and interconnected to perform the required logic functions.

In this environment, it is no longer meaningful to count individual gates. In order to facilitate meaningful density comparisons, it is necessary to evaluate the logic capacity of all the resources available in a programmable gate array, and then to estimate a reasonable utilization factor. For example, the flip-flops in the programmable gate array provide a number of gates equal to the product of the total number of flip-flops available and the number of gates required to implement the same function in a conventional gate array. Table 1 shows the accumulation of total gates in three Xilinx programmable gate arrays.

As shown, the estimates of usable gates as a fraction of the total available gates is lower in programmable gate arrays than in mask-programmed arrays. The 9000 gate XC3090 has over 15,000 total gates. Fifty percent utilization of the *total* gates is about 7660 gates, equivalent to 85% utilization of a conventional 9000-gate array.

One critical element in this analysis is the number of equivalent gates for the function generator portion of the Configurable Logic Block of the LCA. Because of the flexibility of the function generator to directly implement *any* function of its inputs, assigning an equivalent gate count can only be based on some average number of gates for functions which it might implement. In normal applications, the number of gates will range from simple single gate functions to some maximum. For four inputs, the maximum is approximately 20 to 22 gates. For the 3000 family, the logic block can generate functions of 5, 6 or 7 inputs, with some limitations on the 6 and 7 input functions. For the purpose of this evaluation, the 3000 family logic blocks are evaluated as two four-input function generators.

Some functions appropriate for a four-input function generator and their 2-input gate equivalents are listed here.

4 input Parity Generator	12 gates
1 bit Full Adder	10
2 bit Comparator	17

Due to their architecture, the logic capacity of mask-programmed gate arrays is largely independent of application. Gate utilization for programmable gate arrays is more application dependent. In particular, the balance between storage elements and combinatorial logic fits most applications well. Applications that are heavily skewed toward either combinatorial logic or flip-flops will, however, exhibit lower gate utilization.

### PLD DENSITY COMPARISON

In recent years, another set of rules for estimating the “gate density” for programmable logic devices (PLDs) has emerged. In general, PLDs have a very rigid architecture which is rich in functionality, but quite limited in adaptability to a wide variety of logic solutions. Each functional element or structure has the capability of implementing any function with a fixed upper limit on the number of AND and OR terms. When compared with gate array measurements, PLD gate equivalent measurement rules can be misleading since they ignore or minimize the utilization limitations inherent in the PLD architecture. Techniques of this type can only give an approximate relative density, with specific functionality for a design determining the ultimate ability to implement the design in a particular technology.

### BENCHMARK COMPARISON

As Whetstones are used to measure computer performance, benchmarks provide another method to measure relative densities of differing technologies. No standard benchmarks have been developed for density measurement, but one which has been proposed is a frequency counter. This application is documented in the Frequency Counter application brief on page 6-29 in this chapter. This application requires about 1,000 gates in a mask-programmed gate array, and fits into one-half of an XC3020 2,000 gate Logic Cell Array. This indicates a good match between very different technologies and their gate count methods.

Regardless of the measurement technique used, programmable gate arrays provide significant density in implementing system logic functions. Care should be used when comparing different technologies with any technique other than implementing the design with each technology. In actual implementations, the Logic Cell Array families from Xilinx compare very favorably with alternative technologies.

The ideal design environment allows entering a design in many different ways, independent of the target technology, then mapping that design into the technology that best suits the application. The design entry system should support the integration of several design entry methods. For example, a designer should be able to enter a combination of schematics, Boolean equations, and state machines, using whichever method best expresses the logic in any part of the design. Once the design is entered, the designer should then be able to optimize it for the particular technology in which it is to be implemented. Xilinx's Programmable Gate Array design tools support the integration of both schematics and Boolean equations into a design.

Some designs or parts of designs are more easily expressed in Boolean equations than in schematics. A simple yet effective illustration of this is a seven segment decoder design, which expressed schematically requires many AND and OR gates, but expressed with equations requires only a text file. The XC-DS23 Automatic Design Implementation (ADI) package includes software that allows designers to integrate designs entered using a schematic editor with Boolean equations expressed in the PALASM2 format. The ability to create a design using schematics, equations, or a combination of the two makes the design entry process more flexible and more powerful.

There are four programs in the XC-DS23 that help the designer incorporate Boolean equations into a design. A translator program called PDS2XNF translates PALASM2 design files into Xilinx Netlist Format (XNF) files. XNF is a standard interface between the Xilinx Development System and various design entry packages, e.g., schematic editors. An optimization program called XNFOPT optimizes the logic for Xilinx's Logic Cell Array (LCA) architecture for highest efficiency in terms of density or speed. A program called XNFMERGE, merges various hierarchically related design files such as schematics and PLD designs into one flattened file. After merging, a program called XNF2LCA partitions a design's logic and assigns the logic to the elements of a specific LCA: configurable logic blocks (CLBs) and input/output blocks (IOBs). These four programs comprise the tools necessary to create designs that integrate schematics and equations.

## INCORPORATING BOOLEAN EQUATIONS INTO SCHEMATICS

There are a number of ways in which PLD equations can be used in designing LCAs. Perhaps the best way is to incorporate PLDs into a schematic (see Figure 1).

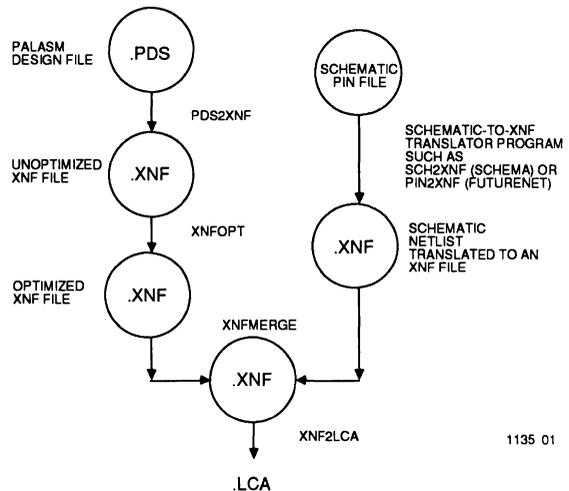


Figure 1. Design Flow of PLDs in a Schematic

Xilinx provides PLD symbols for both Schema II and DASH-LCA, so that designers can call up PLD symbols into their schematics. The user attaches a special attribute called a "FILE=" parameter to each PLD symbol. This parameter (FILE= *file\_name*) references an optimized XNF file that contains the translated Boolean equations for that particular PLD (see Figure 2).

The PLD design file, entered in the PALASM2 format (see Figure 3), is translated into the Xilinx Netlist Format using the PDS2XNF program and then optimized for the LCA

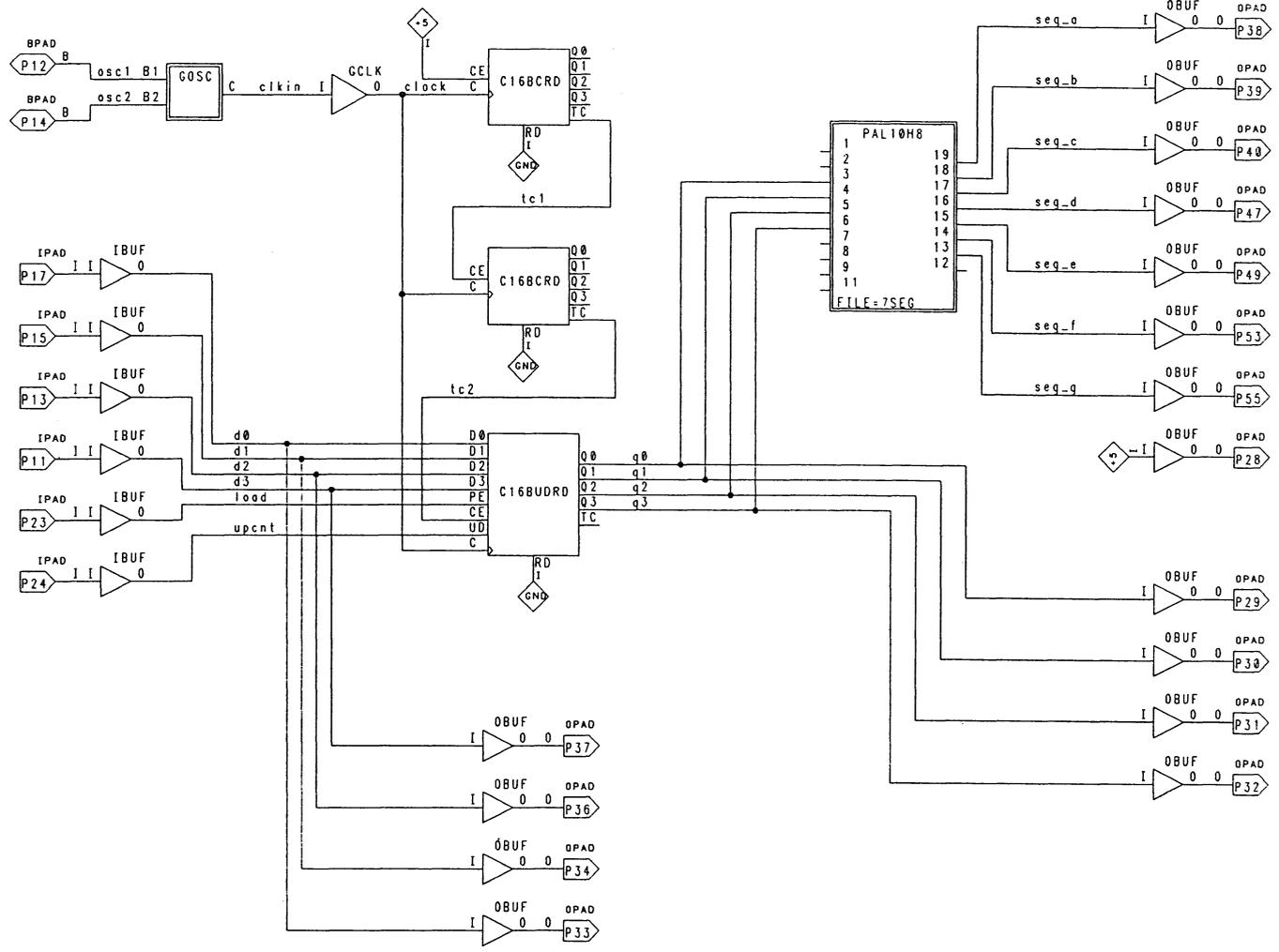


Figure 2. FutureNet Schematic of a Counter with a 7-Segment Decoder

architecture using the XNFOPT program. PLD equations like the ones shown below are in a sum-of-products form that does not always map efficiently into the CLBs of an LCA. The logic represented by Boolean equations such as these must be optimized using the XNFOPT program before it can be translated into an LCA.

```
TITLE      7SEG.PDS
AUTHOR    BART REYNOLDS AND THOMAS WAUGH
COMPANY   XILINX
DATE      APRIL 8, 1988
```

```
CHIP      7SEG   PAL10H8
```

```
;Input Pins   1  2  3  4  5  6  7  8  9 10
               NC NC D0 D1 D2 D3 NC NC NC NC
;Output Pins  11 12 13 14 15 16 17 18 19 20
               NC NC  G  F  E  D  C  B  A  NC
```

```
;
;Input combinations
;
STRING ZERO   \D3 * /D2 * /D1 * /D0'
STRING ONE    \D3 * /D2 * /D1 * D0'
STRING TWO    \D3 * /D2 * D1 * /D0'
STRING THREE  \D3 * /D2 * D1 * D0'
STRING FOUR   \D3 * D2 * /D1 * /D0'
STRING FIVE   \D3 * D2 * /D1 * D0'
STRING SIX    \D3 * D2 * D1 * /D0'
STRING SEVEN  \D3 * D2 * D1 * D0'
STRING EIGHT  \ D3 * /D2 * /D1 * /D0'
STRING NINE   \ D3 * /D2 * /D1 * D0'
STRING TEN    \ D3 * /D2 * D1 * /D0'
STRING ELEVEN \ D3 * /D2 * D1 * D0'
STRING TWELVE \ D3 * D2 * /D1 * /D0'
STRING THIRTEEN \ D3 * D2 * /D1 * D0'
STRING FOURTEEN \ D3 * D2 * D1 * /D0'
STRING FIFTEEN \ D3 * D2 * D1 * D0'
```

**EQUATIONS**

```
A = ZERO + TWO + THREE + FIVE + SIX + SEVEN
    + EIGHT + NINE + TEN + TWELVE + FOURTEEN
    + FIFTEEN
B = ZERO + ONE + TWO + THREE + FOUR + SEVEN
    + EIGHT + NINE + TEN + THIRTEEN
C = ZERO + ONE + THREE + FOUR + FIVE + SIX
    + SEVEN + EIGHT + NINE + TEN + ELEVEN
    + THIRTEEN
D = ZERO + TWO + THREE + FIVE + SIX + EIGHT
    + ELEVEN + TWELVE + THIRTEEN + FOURTEEN
E = ZERO + TWO + SIX + EIGHT + TEN + ELEVEN
    + TWELVE + THIRTEEN + FOURTEEN + FIFTEEN
F = ZERO + FOUR + FIVE + SIX + EIGHT + NINE
    + TEN + ELEVEN + TWELVE + FOURTEEN
    + FIFTEEN
G = TWO + THREE + FOUR + FIVE + SIX + EIGHT
    + NINE + TEN + ELEVEN + THIRTEEN
    + FOURTEEN + FIFTEEN
```

Figure 3. PALASM2 Design File Referenced in the Schematic

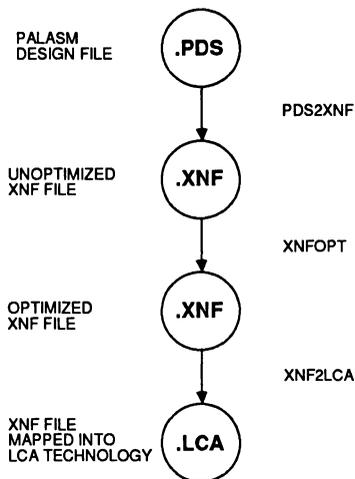
The schematic portion of the design is also translated into an XNF file. The schematic XNF file is considered the top level of the design hierarchy, since it contains references to other XNF files like optimized XNF files from PLD designs. The XNFMERGE program merges lower level XNF files into the top level XNF file from the schematic to form one flattened file that can be partitioned into the LCA architecture, then placed and routed just like any other design.

**CONVERTING PLD DESIGNS INTO LOGIC CELL ARRAY DESIGNS**

A second way of using Boolean equations to design LCAs is to translate an existing PLD design directly into the LCA architecture. In this case, the equations are not merged into a top level schematic. Instead, they are translated directly into an XNF file, optimized for the LCA architecture, and then translated into an LCA file that can be placed and routed and eventually made into a bitstream. There is much more logic available in an LCA than in a PLD, but PLD designs may be translated into an LCA in this manner.

**DESIGNING LOGIC CELL ARRAYS WITH BOOLEAN EQUATIONS**

A third method of using equations to design LCAs is to express the LCA completely in terms of Boolean equations



1135 02

Figure 4. Flow Chart of Direct Translation of PLD Equations

without regard for a particular PLD type (also shown in Figure 4). The PDS2XNF program will translate PLD equations into an XNF file, even if the PLD design file does not correspond to an existing PLD device. The PDS2XNF program recognizes some features that are not permissible in the standard PALASM2 format, including unlimited signal names in the design pinlist, internal signals not named in the pinlist, arbitrarily complex equations, and 32 character signal names. LCA design with equations does not allow the use of all the features of the LCA (e.g., input flip-flops on XC3000 IOBs), but it is a useful and convenient way of designing LCAs completely and quickly using Boolean equations.

**DESIGN OPTIMIZATION**

The cornerstone of the PLD conversion software is a program called XNFOPT. PLD equations are usually expressed in a sum-of-products form that is not optimally suited for the CLB architecture, with blocks of 4 or 5 inputs that perform any function of those inputs. The XNFOPT program is used to optimize logic so that the logic fits efficiently into the CLBs and IOBs of the LCA. XNF files translated from PLD equation design files should always be optimized using the XNFOPT program; in some situations XNFOPT may be used to optimize a design from a schematic as well, although there are some caveats.

There are some designs for which optimization using the XNFOPT program is inappropriate. XNFOPT is meant for synchronous designs, not asynchronous ones. XNFOPT significantly alters logic, so it may not preserve a design's asynchronous timing dependencies. Furthermore, although optimized logic is functionally equivalent to the original design, the structure may be significantly changed making debugging more difficult. It is generally a good idea to first simulate a design functionally to insure its correctness before optimizing it. Logic in an XNF file that is expressed as CLB primitives cannot be optimized, as it is already partitioned into the LCA architecture.

The XNFOPT program can be directed to optimize a design for speed or for density. Optimizing for density reduces the number of CLBs used by the design, while optimizing for speed reduces the number of levels of logic used by the design. In its default mode of operation, XNFOPT optimizes for density, using the -L option directs XNFOPT to optimize for speed.

Xilinx supports the PALASM2 equation entry language format directly, and supports other equation entry languages like ABEL and CUPL indirectly through the use of a JEDEC dis-assembler called JEDMAN, which converts

JEDEC files into the PALASM2 format. JEDMAN is not available from Xilinx. This PALASM2 file can then be translated into an XNF file and optimized like a PLD equation file originally entered in PALASM2. The major disadvantage of this method is that the signal names of the designs are lost when passed through the JEDEC assembler and dis-assembler. This can cause problems during design debugging.

In many cases, designs are most easily expressed using a combination of logic equations and schematics. Software tools in Xilinx's ADI package give designers the capability to integrate equation and schematic entry methods. The benefits of these logic synthesis tools are simpler and more powerful design entry, more efficient use of LCA logic, and clearer design documentation.

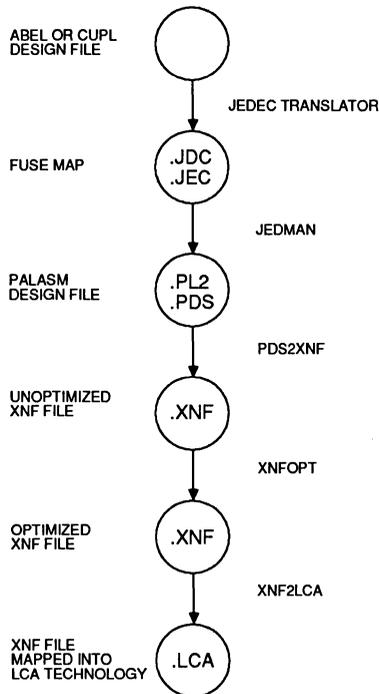


Figure 5. Converting PLD Designs from ABEL or CUPL

In late 1987 Xilinx introduced the XC3000 family which, within a year grew to five members: the XC3020, XC3030, XC3042, XC3064 and XC3090.

Designers already familiar with the original XC2000 family may be interested in the following synopsis of the differences between the 2000 and 3000 families.

### COMBINATORIAL LOGIC:

- Combinatorial logic is wider (5 inputs vs. of 4 in XC2000 family).

### CLB STORAGE ELEMENTS:

- There are two storage elements in a CLB.
- Storage element is a flip-flop only, the latch option is eliminated.
- Data input of either flip-flop can come from the combinatorial logic output, (F or G,) or from the one direct data input, DIN of each CLB.
- Clock input can be driven from local lines in addition to global and alternate clock lines.
- Enable Clock signal eliminates need for clock gating.
- Asynchronous set is removed but asynchronous reset is retained, consistent with TTL-MSI design practices.
- No flip-flop inputs are shared with combinatorial logic.
- Clock, Enable Clock, and Reset are common to both flip-flops.

### IOBs

- Both direct input (I) and registered input (Q) are available simultaneously.
- Input storage element can be flip-flop or latch.
- Output signal and 3-state control signal can be inverted.
- Output signal can be either direct or registered.
- Output buffer has a programmable passive pull-up to prevent floating input.
- Output buffer has option to slow down the output transition to reduce transient noise.

### ROUTING RESOURCES:

- 1 additional horizontal local line per row.

- 1 additional horizontal long line per row.

- 1 additional vertical clock line per column.

- All CLB inputs and outputs have access to both horizontal and vertical routing channels.
- Capability of magic box is enhanced.
- Both horizontal long lines have 3-state bus drivers.
- Bi-directional buffers are evenly distributed and used only when needed to boost a signal.

### OTHER DIFFERENCES

- Crystal oscillator has a "+ 2" option (50% duty cycle)
- Oscillator output can be connected to the general interconnect.
- Added fast, dedicated CMOS input buffer for the global and alternate clocks.
- Capability of the edge magic box is enhanced.
- Each edge has two IOCLKS, clock source multiplexers are located in each corner.
- IOCLK can be synchronized to the global and alternate clock. IOB on adjacent chip edges can be driven from the same clock source.
- All IOCLK clock drivers can be inverted.
- Two vertical long-lines in each column and one long-line on each horizontal and vertical edge can each be split into two half long-lines. (On the XC3020 this can only be done with one long-line on each edge.)

### DIFFERENCE IN CONFIGURATION

- Parallel peripheral mode is added.
- Modified timing for slave mode avoids hold time.
- No RCLK output in master serial mode, use CCLK.
- Modified pinout for master and peripheral mode, all data pins are on the right hand edge.
- INIT state brought out as output.
- Done and Reset, programmable one clock early or late.
- PWRDN does not affect "House Cleaning".



# Additional Electrical Parameters

## Application Brief

The XILINX LCA data sheets specify worst case device parameters, 100% tested in production and guaranteed over the full range of supply voltage and temperature.

Some users may be interested in additional data that is not 100% tested and, therefore, not guaranteed. Here are results from recent bench measurements:

### PULL-UP RESISTOR VALUES

I/OB Pull-ups	40 to 150 k $\Omega$
DONE Pull-up	2 to 8 k $\Omega$
Long Line Pull-up(each)	3 to 10 k $\Omega$

These values were measured with the node pulled LOW. At a logic HIGH the resistor value is 5 to 10 times higher.

### INPUTS

#### Hysteresis

All inputs, except PWRDN, and XTL2 when configured as the crystal oscillator input, have limited hysteresis, typically in excess of 200 mV for TTL input thresholds, in excess of 100 mV for CMOS thresholds.

#### Required Input Rise and Fall Times

For unambiguous operation, the input rise time should not exceed 200 ns, the input fall time should not exceed 80 ns.

These values were established through a worst-case test with internal ring oscillators driving all I/O pins except two, thus generating a maximum of on-chip noise. One of the remaining I/O pin was then tested as an input for single-edge response, the other one was the output monitoring the response. This specification may, therefore, be overly pessimistic, but, on the other hand, it assumes negligible PC board ground noise and good Vcc decoupling.

### OUTPUTS

All LCA outputs are true CMOS with n-channel transistors pulling down, p-channel transistors pulling up. Unloaded, these outputs pull rail-to-rail.

#### DC Parameters

##### Output Impedance

Sinking, near ground:	25 $\Omega$
Sourcing, near Vcc:	50 $\Omega$

##### Output Short Circuit Current

Sinking current by the LCA	96 mA
Sourcing current by the LCA	60 mA

The data sheets guarantee the outputs only for 4 mA at 320 mV in order to avoid problems when many outputs are sinking current simultaneously.

#### AC Parameters

	Fast <sup>1</sup>	Slow <sup>1</sup>
Unloaded Output Slew Rate	2.8 V/ns	0.5 V/ns
Unloaded Transition Time	1.45 ns	7.9 ns
Additional rise time for 812 pF	100 ns	100 ns
normalized	0.12 ns/pF	0.12 nspF
Additional fall time for 812 pF	50 ns	64 ns
normalized	0.06 ns/pF	0.08 nspF

<sup>1</sup> "Fast" and "Slow" refer to the output programming option.

There is good agreement between output impedance and loaded output rise and fall time, since the rise and fall time is slightly longer than two time constants.

## Power Dissipation:

LCA power dissipation is largely dynamic, due to the charging and discharging of internal capacitances. The dynamic power, expressed in mW per MHz of actual node or line activity is given below.

Clock line frequency is easy to specify, but the designer will usually have great difficulty estimating the average frequency on other nodes.

Two extreme cases are:

1. Binary counter, where half the total power is dissipated in the first flip-flop.
2. A shift register with alternating zeros and ones, where the whole circuit is exercised at the clocking speed.

	Dynamic Power (mW/MHz)
Output with 50 pF load*	1.9
Global Clock (XC3020)	1.7
Global Clock (XC3090)	3.6
CLB with Local Interconnect	0.36
Horizontal Long Line (XC3020)	0.09
Horizontal Long Line (XC3090)	0.15
Vertical Long Line (XC3020)	0.08
Vertical Long Line (XC3090)	0.19
Input without Pull-up	0.075

\*Add 2.5 mW/MHz for every 100 pF of additional load

Example: XC3020 with	Dynamic Power (mW/MHz)
3 outputs at 5 MHz	28
20 outputs at 0.1 MHz	4
Global Clock at 20 MHz	34
10 CLBs at 5 MHz	18
40 CLBs at 0.2 MHz	3
16 Vertical Long Lines at 1 MHz	1
20 Inputs at 4 MHz	6
<b>Total</b>	<b>94 mW</b>

## CCLK Frequency Variation

Configuration Clock (CCLK) is the internally generated free-running clock that is responsible for shifting configuration data into and out of the device.

CCLK frequency is fairly stable over  $V_{cc}$ , varying only 0.6% for a 10% change in  $V_{cc}$ , but is very temperature dependent, increasing 40% when the temperature drops from 25°C to -30°C, decreasing 40% when the temperature increases from 25°C to +130°C.

$V_{cc}$	T	Freq
4.5V	25°C	687 kHz
5.0V	25°C	691 kHz
5.5V	25°C	695 kHz
4.5V	-30°C	966 kHz
4.5V	+130°C	457 kHz

## CRYSTAL OSCILLATOR

The on-chip oscillator circuit consists of a high-speed, high gain inverting amplifier between two device pins, requiring an external biasing resistor R1 of 4 M $\Omega$ .

A series-resonant crystal Y1 and additional phase-shifting components R2, C1, C2 complete the circuit.

### Fundamental Frequency Operation up to 24 MHz:

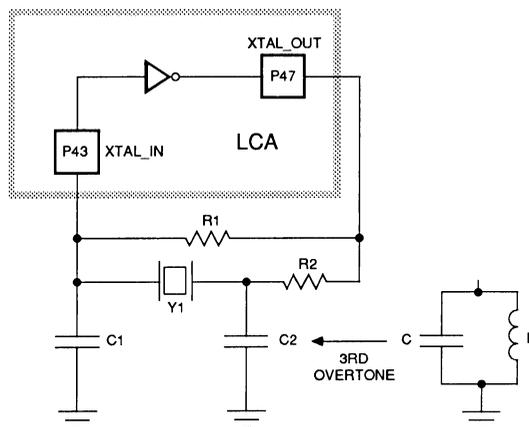
$C1 = C2 = 34$  pF

$R2 = 1$  K $\Omega$  up to 12 MHz, 800  $\Omega$  to 520  $\Omega$  for 15 to 24 MHz

### Third Overtone Operation from 20 MHz to 72 MHz:

Replace C2 with a parallel resonant LC tank circuit tuned to  $\approx 2/3$  of the desired frequency, i.e., twice the crystal fundamental frequency.

Frequency (MHz)	L ( $\mu$ H)	C (pF)	LC Tank		
			Freq (MHz)	R2 ( $\Omega$ )	C1 (pF)
32.00	1	60	20.6	430	23
35.00	1	44	24.0	310	23
49.00	1	31	28.6	190	23
72.00	1	18	37.5	150	12



Crystal Oscillator

1158 02

## Application Brief

### ESTIMATING CLB PERFORMANCE

Since the delays in LCA-based designs are lay-out dependent, the data sheet cannot give all the answers needed to predict the worst case guaranteed performance.

The timing calculator in XACT is a better tool, and a simulation using SILOS, after the design has been routed, will be the final arbiter for worst-case performance.

Still, most designer want to evaluate the possible performance, well before they have finished the design.

Here are some guidelines for XC3000 family devices:

1. A simple synchronous design-like a shift register, where a flip-flop feeds a flip-flop in the next vertical or horizontal CLB through the one level of combinatorial logic in front of the target flip-flop:

	-50	-70
clock-to-output	12 ns	8 ns
routing	6 ns	4 ns
logic set-up	12 ns	8ns
clock period	30 ns	20 ns
clock frequency	33 MHz	50 MHz

2. A similar design with flip-flops several rows or columns apart would double the routing delay:

	-50	-70
clock-to-output	12 ns	8 ns
routing	12 ns	8 ns
logic set-up	12 ns	8 ns
clock period	36 ns	24 ns
clock frequency	28 MHz	42 MHz

3. An additional level of combinatorial logic plus routing adds 14/9 ns for the logic, plus 6/4 ns for typical routing.

	-50	-70
clock-to-output	12 ns	8 ns
routing	12 ns	8 ns
logic delay	14 ns	9 ns
routing	6 ns	4 ns
logic set-up	12 ns	8 ns
clock period	56 ns	37 ns
clock frequency	18 MHz	27 MHz

Therefore, as a rule of thumb, the system clock rate should not exceed one third to one half of the specified toggle rate. Simple designs, like shift registers and simple counters, can run faster, like two thirds of the specified toggle rate.

These numbers assume synchronous clocking from the global clock lines. Remember, these are all worst-case numbers, guaranteed over temperature and supply voltage. Nobody should design with typical numbers.

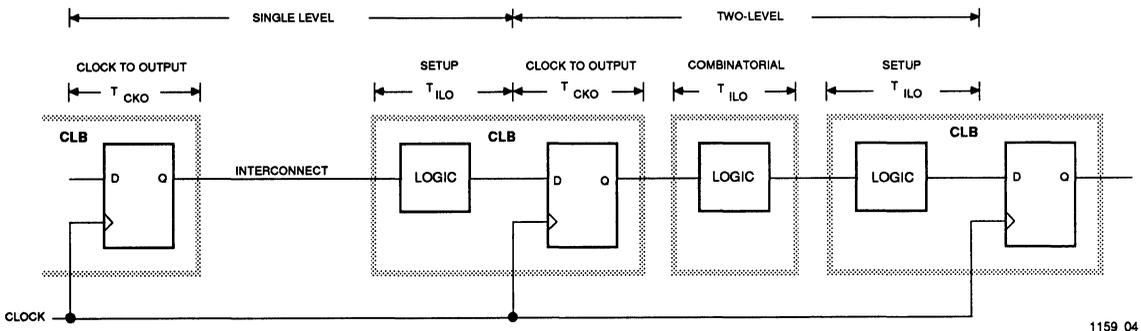


Figure 1. Critical Timing Parameters for Clocked CLB Driving Clocked CLB Directly (Single Level) and Driving it Through Additional Combinational Logic (Two-Level)

## DESIGNING FOR HIGHEST DATA TRANSFER RATE BETWEEN 3000 FAMILY LCAs

Worst case analysis of a synchronous data transfer between 3000 family devices postulates that the sum of clock-to-output propagation delay of the sending device, plus the input-to-clock set-up time of the receiving device, must be less than the clock period.

The inherent freedom in clock and signal routing makes it impossible to give exact values for an unprogrammed LCA without specifying certain restrictions:

On the transmitting LCA, the clock-pin to output-pin propagation delay is minimized if TCLKIN or BCLKIN are chosen as clock inputs. They are CMOS-level only, and offer the shortest on-chip clock delay.

The clock-pin to output delay is then

$$3 + 6 + 13 = 22 \text{ ns for the -70 part}$$

$$5 + 9 + 18 = 31 \text{ ns for the -50 part}$$

On the receiving LCA, the input-pin to clock-pin set-up time is the specified I/O pad input set-up time (parameter TPICK in the IOB switching characteristic table of the XC3000 family data sheet) minus the actual delay for clock buffering and routing.

Assuming the same clock buffer choice on the receiver as on the transmitter, the longest input-pin to clock-pin set-up time is:

$$20 - 3 - 6 = 11 \text{ ns for the -70 device}$$

$$30 - 5 - 9 = 16 \text{ ns for the -50 device}$$

Under these assumptions, the worst case (shortest) value for the clock period is:

$$22 + 11 = 33 \text{ ns, i.e., max 30 MHz for the -70 device}$$

$$31 + 16 = 47 \text{ ns, i.e., max 21 MHz for the -50 device}$$

If this is not fast enough, there are design methods that can improve the performance. Let us assume a -70 device. The easiest and safest method is to increase the clock delay on the receiving LCA, thus reducing the apparent input set-up time. Changing to a direct input (instead of TCLKIN) adds 4 ns to the clock delay, thus subtracts at least 3 ns from the input set-up time, reducing it to 30 ns (worst case), i.e., max 33.3 MHz.

More aggressive methods of increasing clock delay inside or outside the receiving LCA must be used with care, since they might reduce the "best case" set-up time (fast process, low temperature, high Vcc) to a value of less than zero, i.e., make it a hold time requirement, which, in conjunction with a best case very fast transmitting device, can lead to problems.

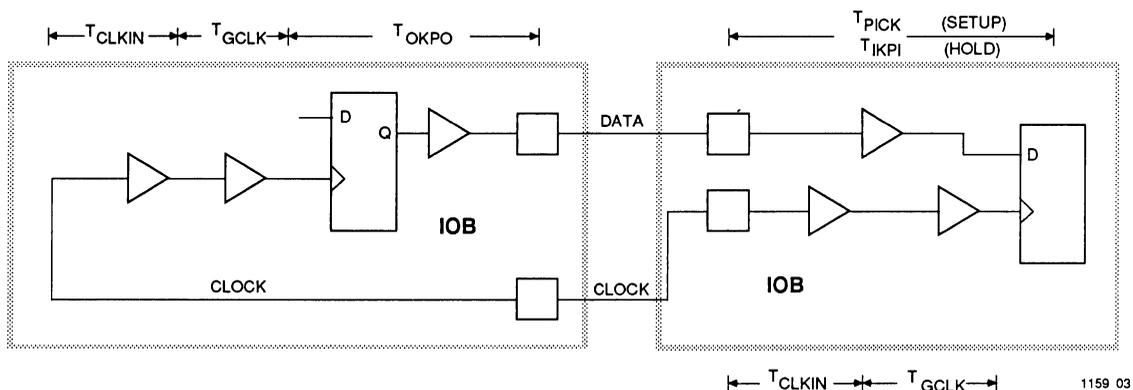


Figure 2. Critical Timing Parameters for Data Transfer Between LCAs

**INPUT SET-UP TIME ON A 3000 SERIES LCA IS BETTER THAN THE SPECIFICATION.**

The Xilinx 3000 series data sheet specifies a worst case input set-up time of 20 ns for the -70 speed grade (parameter #1 on page 41), but this is the set-up time with respect to the internal clock, not the clock input pad.

The clock delay is max 9 ns, the sum of pad to direct CLKIN (3 ns, assuming that the CMOS-compatible clock input is used) and the clock buffer (6 ns).

Xilinx does not guarantee any shortest values for all these parameters, an unrealistic worst-worst case analysis might, therefore, assume two extreme values:

20 ns set-up time for a slow data input with an infinitely fast clock path

9 ns hold time for an infinitely fast data input combined with a slow clock path

This is a meaningless mathematical exercise. In reality, all these delays track very well over temperature, supply and processing variations, never deviating more than 30% from each other's normalized value. If for any specific part and operating condition the set-up time is 50% of its max value, then the clock delay will be between 35% and 65% of its max value. (35% is 30% less than 50%, 65% is 30% more than 50%.)

A conservative analysis also shows that the fastest delay parameters are never shorter than 10% of the specified guaranteed value. This fastest value occurs at the lowest temperature and the highest supply voltage.

As a result, the longest input set-up time with respect to the CMOS-compatible clock input is 14 ns (20ns minus 70% of 9 ns), and the shortest set-up time is 1.1 ns (10% of: 20 ns minus 9 ns)

There will never be a hold time requirement if the user selects the CMOS compatible clock input option.

## CLB FLIP-FLOPS RECOVER SURPRISINGLY FAST FROM METASTABLE PROBLEMS

A specter is haunting digital design, the specter of metastability. From a poorly understood phenomenon in the seventies, it has developed into a scary subject for every designer of asynchronous interfaces. Now Xilinx offers data and a demonstration kit to help users analyze and predict the metastable behavior of LCAs.

Whenever a clocked flip-flop synchronizes a truly asynchronous input, there is a small but finite probability that the flip-flop output will exhibit an unpredictable delay. This happens when the input transition not only violates the setup and hold-time specification, but actually occurs within the tiny timing window where the flip-flop "decides" to accept the new input. Under these circumstances the flip-flop enters a symmetrically balanced state, called metastable, (meta = between) that is only conditionally stable. The slightest deviation from perfect balance will eventually cause the outputs to revert to one of the two stable states, but the delay in doing so depends not only on the gain bandwidth product of the circuit, but also on the original balance and the noise level of the circuit; it can, therefore, only be described in statistical terms.

The problem for the system designer is not the illegal logic level in the balanced state (it's easy enough to translate that to either a 0 or a 1), but the unpredictable timing of the final change to a valid logic state.

The basic phenomenon is as unavoidable as death and taxes, but the probability of erroneous operation can be determined, and the impact of various countermeasures can be evaluated quantitatively, if two fundamental flip-flop parameters are known, i.e., the metastability capture window, and the metastability recovery rate.

Xilinx has evaluated the XC3020 CLB flip-flop with the help of a mostly self-contained circuit on the Demonstration Board that is available to any Xilinx customer.

The result of this experimental evaluation shows the Xilinx CLB flip-flop superior in metastable performance to many popular MSI or PLD devices.

When an asynchronous event frequency of approximately 1 MHz is being synchronized by a 10 MHz clock, the CLB flip-flop will suffer an additional delay of

4.2 ns statistically once per hour

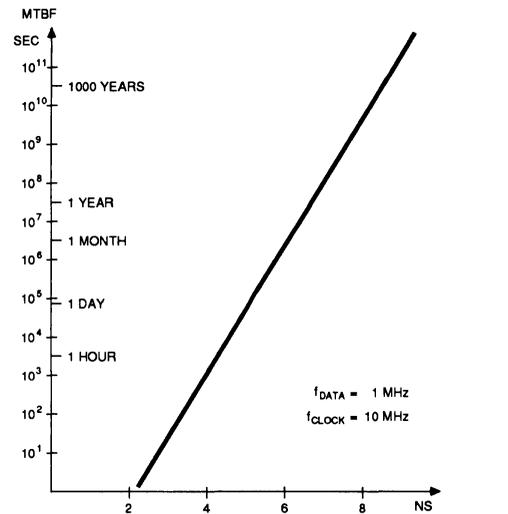
6.6 ns statistically once per year

8.4 ns statistically once per 1000 years

The frequency of occurrence of these metastable delays is proportional to the product of the asynchronous event frequency and the clock frequency.

If, for example, a 100 kHz event is synchronized by a 2 MHz clock, the above mentioned delays (besides being far more tolerable) will occur 50 times less often.

The mean time between metastable events lasting longer than a specified duration is an exponential function of that duration. Two points measured on that line, allow extrapolation to any desired MTBF (mean time between failure).



Metastable MTBF as a Function of Additional Acceptable Delay

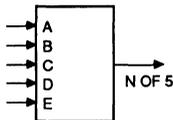
1160 01

### MAJORITY LOGIC, N-OF-X DECODING

Majority logic has interesting mathematical features, but has not become popular because its traditional logic implementation is quite complicated and expensive. Since LCAs can generate any function of five variables at the same cost and the same delay, they can easily decode majority logic.

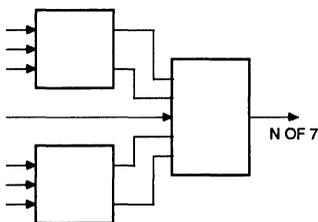
We can define the output F, G to be Low for 0, 1, or 2 inputs High, to be High for 3, 4, or 5 inputs High.

Majority logic is a special case of "N-of-X Decoding." A 3000-series CLB can directly encode any "N of 5" inputs active. This concept can be cascaded so that three LCBs encode any "N of 7" inputs active.



5-INPUT MAJORITY FUNCTION:

$$F = ABC + ABD + ABE + ACD + ACE + ADE + BCD + BDE + BCE + CDE$$

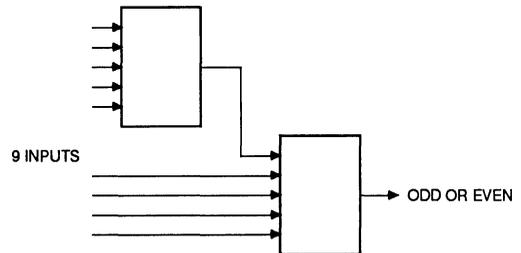


1139 01 **Figure 1. N of X, e.g., Majority Logic**

The first-level blocks can only have 3 inputs, since the two outputs can only encode 4 different states: none, one, two, or three active.

### PARITY

Two CLBs can generate the parity for nine inputs, or can check a nine-bit input for odd or even parity with a through-delay of two cascaded CLBs. Three CLBs can check 13 inputs; four CLBs can check 17 inputs; five CLBs can check 21 inputs; six CLBs can check 25 inputs; all with the same delay of two cascaded CLBs.

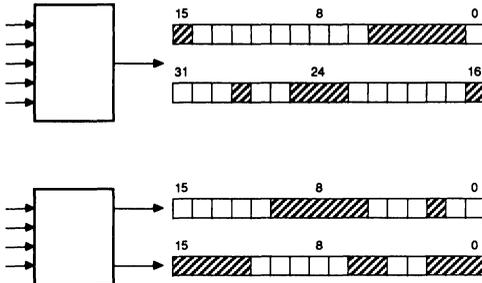


1139 02

**Figure 2. Parity**

## Application Brief

A 3000-series CLB can decode a 5-bit address in any conceivable way, or it can decode a 4-bit address in two different ways, each without any restrictions.



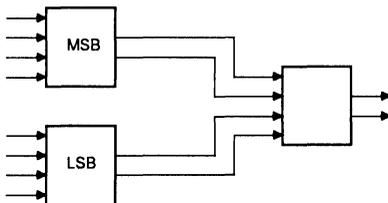
1140 01

**Figure 1. Multiple Address Decoding**

Three 3000-series CLBs can decode three distinct addresses in an 8-bit address field: One CLB decodes the lower 4 bits and encodes the result on its two outputs (00 = no match). The second CLB decodes and encodes the upper 4 bits in a similar way.

The third CLB encodes the four signals into two outputs (00 = no match). This works for any three distinct addresses, even when some share the same upper or lower nibble.

This scheme can be expanded to a 16-bit wide address, using seven CLBs.



1140 02

**Figure 2.**

### Address Block Detection

The idea mentioned above is not restricted to detecting three specific addresses, it can also detect three groups of addresses, as long as none of them straddles the boundaries defined by the individual CLBs. If they do, this circuit cannot detect *three* address blocks, but can still detect any *one* address block in an 8-bit address.

Suppose we want to decode the block of 8-bit addresses starting at 24 and including 68 (hex).

With one CLB we encode the least significant address nibble into a 2 bit output called LS:

- |                    |                 |
|--------------------|-----------------|
| inputs 0 through 3 | generate LS = 1 |
| inputs 4 through 8 | generate LS = 2 |
| inputs 9 through F | generate LS = 3 |

With another CLB we encode the most significant address nibble into a 2 bit output called MS:

- |                             |                  |
|-----------------------------|------------------|
| inputs 0, 1, 7, 8 through F | generate MS = 0  |
| input 2                     | generates MS = 1 |
| input 3, 4, 5               | generate MS = 2  |
| input 6                     | generates MS = 3 |

The third CLB then encodes these signals

MS	LS	Output
0	x	0
1	2,3	1
2	x	1
3	1,2	1
3	3	0

The solution can be generalized:

Three CLBs can decode any one block of an 8-bit address.

The LCA-structure accommodates 1-bit and 2-bit adders very efficiently. A 1-bit adder with 3 inputs (A, B, Cin) generating 2 outputs (S, Cout) fits exactly in one 2000 series CLB, where the flip-flop might be used for storing the carry in a bit serial adder. A 3000-series CLB can even include an additional control input, either ADD/SUBTRACT or ADD ENABLE.

A two-bit adder requires three 3000 series CLBs. The five inputs A0, B0, A1, B1, and Cin are common to all three CLBs, the outputs are S0, S1, and Cout. The propagation delay is only one CLB combinatorial delay, as little as 10 ns. Two such adders can be cascaded to form a four bit adder in 6 CLBs with a through-delay of two CLBs, i.e., 25 ns (allowing for some interconnect delay).

Four two-bit adders can be cascaded to form a byte-wide adder, using 12 CLBs with a through-delay of 4 CLBs, but there is also a slightly faster design using a carry lookahead technique: The third and fourth di-bit adders are changed, they no longer generate Carry out, but now each generates two outputs as a function of the four A and B inputs (ignoring Cin). These two outputs are called Carry Generate (when the addition exceeds a binary 3) and Carry Propagate (when the addition is exactly 3). These outputs from two di-bit adders are combined with Cin and generate the Carry inputs to the fourth di-bit adder and its carry out. The whole 8 bit adder uses 14 CLBs and has a through-delay of four CLB delays from input to S6 and S7, only three CLB delays from input to Cout.

For eight bits, this look-ahead carry scheme is of marginal use, it reduces only the carry delay, and only by one CLB delay. For this small speed improvement it uses two additional CLBs (14 instead of 12).

A 16-bit adder benefits from carry-lookahead. Simply cascading di-bit adders uses 24 CLBs at a max propagation delay of 8 CLBs from Cin to Cout or to S14, 15. A look-ahead carry scheme uses 30 CLB at a max prop delay of 5 CLBs from Cin to Cout (6 delays to S14, 15).

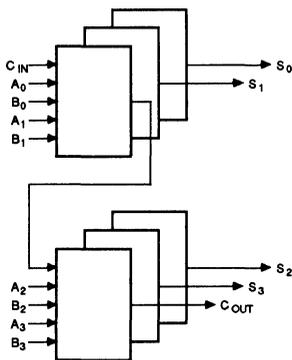


Figure 1. 4-Bit Adder

1141 01

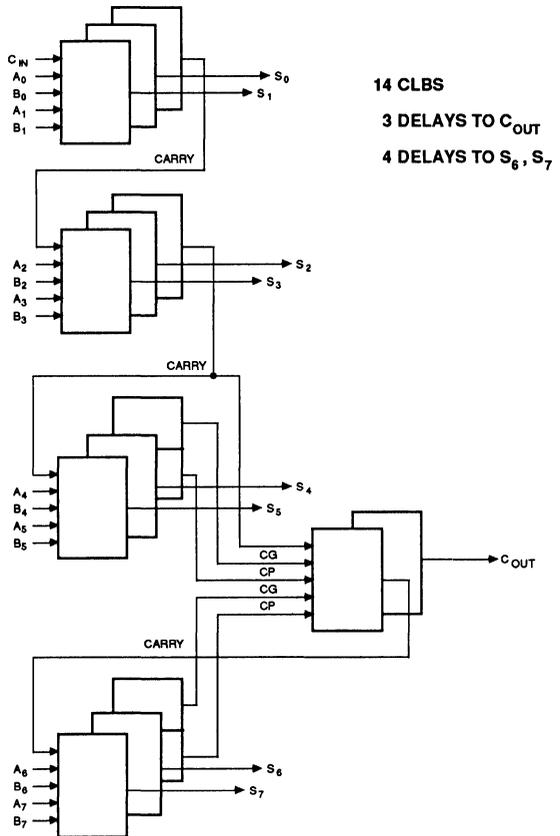


Figure 2. 8-Bit Adder with Carry Lookahead

1141 01

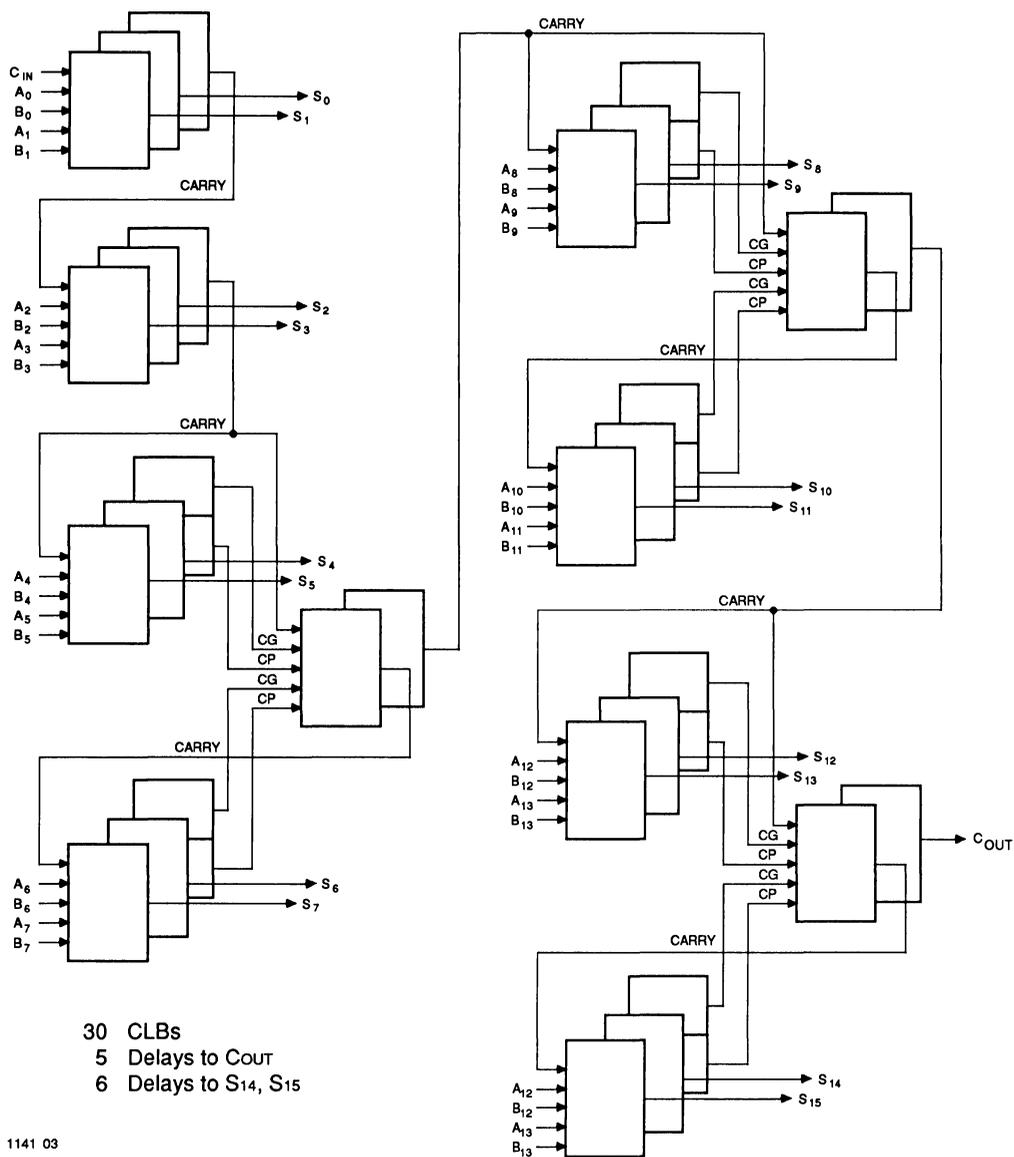


Figure 3. 16-Bit Adder with Carry Lookahead

**Bit-Serial Adder, Subtractor, Comparator**

The CLB architecture is ideally suited for bit-serial arithmetic, where the function generator performs the serial arithmetic (LSB first), and the associated flip-flop stores the carry or borrow.

A bit-serial **identity** comparator detects only whether the two operands are equal or not, without determining which one (if any) is larger. The bit stream can come in LSB or MSB first, the flip-flop gets set for any difference between A and B, and stays set until the end of the word, then gets reset before the beginning of the next word. This "difference detector" can also be implemented as a latch and folded into the combinatorial logic.

A bit-serial **magnitude** comparator distinguishes between  $A = B$ ,  $A > B$  and  $A < B$ . It can operate LSB first or MSB first, if the logic is adjusted:

LSB first: Start with both flip-flops reset

- if  $A > B$  set  $Q_x$ , reset  $Q_y$
- if  $A < B$  set  $Q_y$ , reset  $Q_x$

MSB first: Start with both flip-flops reset

- if  $A > B$  and  $Q_y = 0$ : set  $Q_x$
- if  $A < B$  and  $Q_x = 0$ : set  $Q_y$

Result in both cases:

$Q_x$	$Q_y$	
0	0	$A = B$
0	1	$A < B$
1	0	$A > B$
1	1	Impossible

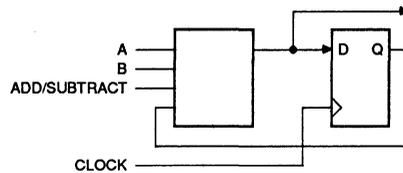


Figure 4. Serial Adder/Subtractor

1141 04

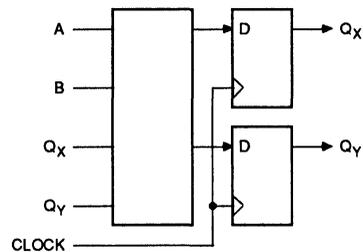


Figure 5. Serial Magnitude Comparator

1141 05

## Application Brief

Since the 3000-series, unlike the 2000-series, cannot configure its CLB flip-flops into latches, there must be other ways to design latches. Obviously, the I/O block can be configured with latches on either the input, the output, or both. Beyond that, every CLB can form a latch.

The five-input logic structure allows an amazing diversity of latch designs: four examples are documented in the 3000-series Macro Library on page 15 and 16. Here are additional ideas:

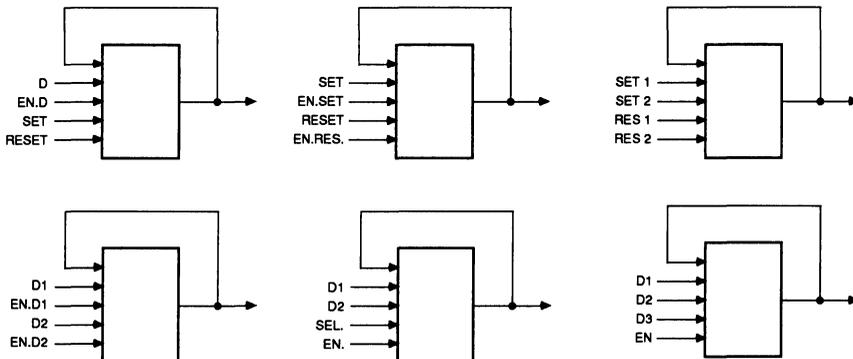
With F fed back to close the feedback path, there are four control inputs left. We might call them Set, Reset, Data and Enable, defining them such that S and R are independent of E, but D is activated by E. We can still define any of these four inputs as active High or active Low. That gives us 16 different latch designs, all with the same basic characteristics and the same timing.

We can also eliminate D and have two Enables, affecting

S and R (again 16 different flavors) or we use multiple S and multiple R, either ORed, or ANDed, or XORed. We can also have two D inputs, each with its own Enable; or we can have two D inputs, a Select input and an Enable input; or we can have an Enable and three D inputs defined in any arbitrary way. Majority gating could be one way: if none or one is active, reset the latch; if two or three are active, set the latch. Or, if none is active, reset; if one or two are active, hold; if three are active: set. Or we can assign positive or negative weights to the D inputs.

Since there are 65,536 different functions of four variables, there are many different ways to define a latch, not counting pin rotations and active High/active Low variations.

All these latches have the same timing characteristics: propagation delay from input to output = 14/9 ns for the 50/70 MHz part. Set-up time to the end of Enable, or min. Enable width = 19/14 ns assuming 5 ns interconnect delay.



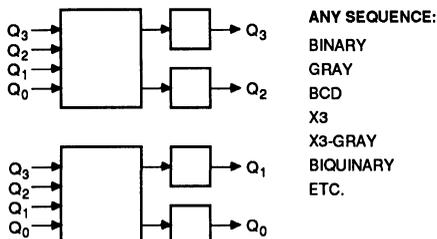
1142 01

Figure 1. Latched Logic

## Application Brief

### FULLY SYNCHRONOUS 4-BIT COUNTER USES ONLY TWO CLB'S TO COUNT ANY CODE

This four-bit counter operates synchronously and has a Count Enable (Clock Enable) input. Count length, count direction, and even the code sequence can be selected through configuration. There are 15!, i.e. more than  $10^{12}$  different possible sequences. All four outputs are available. This counter cannot be preset to an arbitrary value, but it can be cleared by an asynchronous input.



1143 01

Figure 1. Synchronous 4-Bit Counter In 2 CLBs

The advantage of a Gray code is its glitch-less decoding, since only one bit changes on any code transition. A Gray counter can also be read "on-the-fly" without the well-known problems of reading a binary counter e.g., on its transition between 7 and 8, where any code might be read.

Decimal	Binary	Gray	X3 Binary	X3 Gray
0	0000	0000	0011	0010
1	0001	0001	0100	0110
2	0010	0011	0101	0111
3	0011	0010	0110	0101
4	0100	0110	0111	0100
5	0101	0111	1000	1100
6	0110	0101	1001	1101
7	0111	0100	1010	1111
8	1000	1100	1011	1110
9	1001	1101	1100	1010
10	1010	1111		
11	1011	1110		
12	1100	1010		
13	1101	1011		
14	1110	1001		
15	1111	1000		

### FULLY SYNCHRONOUS 5-BIT COUNTER USES ONLY THREE CLBs

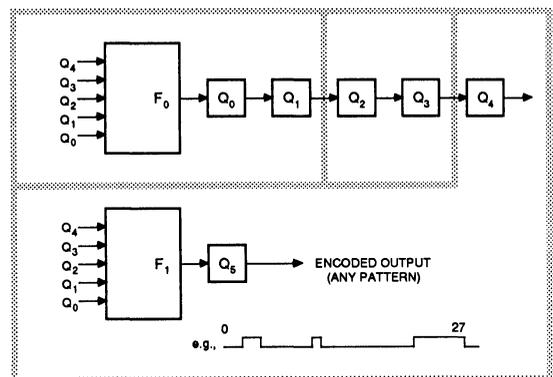
Three 3000-series CLBs can implement a modified shift-register counter with the following features:

- Fully synchronous operation
- Count Enable Asynchronous clear
- Count-Modulus defined during configuration: 2...32
- Only one meaningful output, Q5, but with complete freedom to define its waveform

Q0 through Q4 form a linear shift register counter. The 5 input combinatorial function F0 determines the modulus (there are no illegal or hang-up states). The 5-input combinatorial function F1 decodes the counter in any conceivable way, Q5 synchronizes and de-glitches F1.

#### Examples:

- + 28 counter with output High at times T2, 3, T10, T22 through T27
- + 19 counter with output Low at times T9, T12, T15, T18.



1144 01

Figure 2. Synchronous 5-Bit Counter In 3 CLBs

This application note describes a new counter architecture used to implement a very high speed presetable, up to 40-bit long binary counter in an XC3020 Programmable Gate Array. The design can easily be modified to implement two 20-bit counters, or the equivalent BCD counters.

Traditional counter designs always represent a compromise between two conflicting goals: highest clock speed/event resolution on one hand, sophisticated features (like preset to any arbitrary value, or decode any state) on the other hand.

Asynchronous ripple counters offer highest speed, but cannot be decoded in one clock period, thus cannot be made programmable.

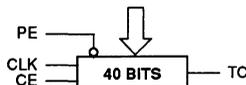
Synchronous counters allow decoding and presetting in one clock period, but pay for this with complex carry logic. Carry propagation is always the limiting factor in the traditional design of presetable synchronous counters, since the complete carry chain must reach a steady state before the next incoming clock edge. Brute force parallel decoding of all previous states becomes unmanageable beyond 8 stages, but cascaded decoding introduces additional delays. Either approach reduces the inherent resolution of the counter.

Decoding Terminal Count (TC) in order to preset the counter again, poses a similar problem. The design described in this paper separates the two functions of the carry chain into:

- One which decodes the terminal count of the whole counter and generates a Parallel Enable signal
- One which propagates the carry signal from the less significant to the more significant bit positions, and causes the appropriate flip-flop to toggle.

### Cascaded TC Decoding:

- The TC decoder must receive inputs from all counter bits, but only the LSB timing is critical, the more significant bits have been stable before. TC can, therefore, be decoded in a slow gating chain that starts at the most significant end of the counter.

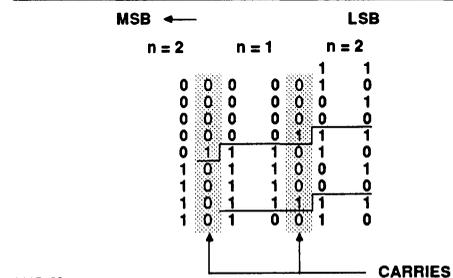


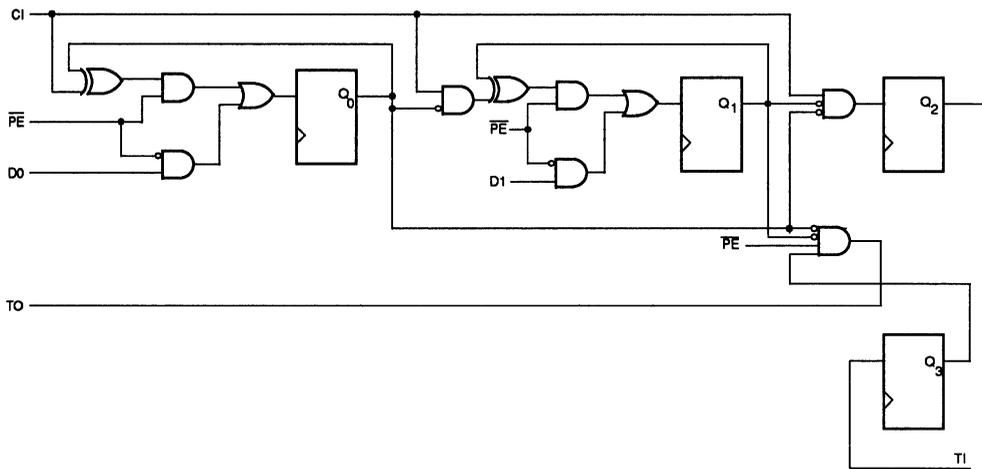
### Carry Propagation:

- Since a presetable counter only decodes one state, TC, the decision to toggle any of the more significant bits can be delayed and thus pipelined without any problem.

The counter is divided into a number of small sections, each two bits (a dibit) long, implemented as a synchronous presetable down-counter, with carry-in (=count enable), parallel enable and two data inputs. Terminal count (0.0) is decoded with an additional input coming from the next higher section. The least significant section decodes the state prior to TC, its output activates the parallel enable for all counters. The carry function between sections is pipelined. The carry flip-flop is set when carry-in is active and the dibit is in state 00 for a down counter, or state 11 for an up-counter. The carry flip-flop stays set for only one clock period, its output drives the carry-in function of the next higher section. As a result of this pipelining, the counter can be made arbitrarily long, without any speed penalty. Note that each dibit, except the first, makes its transition  $n$  clock pulses later than required by the binary code sequence ( $n$  is the relative position of the dibit,  $n=0$  for the input dibit). This code violation has no impact on TC decoding. This counter can be four times faster than presently available standard microprocessor peripherals like the 8254 and 9513. Typical applications are in instrumentation and communications, e.g., as the frequency determining counter in a phase-locked-loop frequency synthesizer.

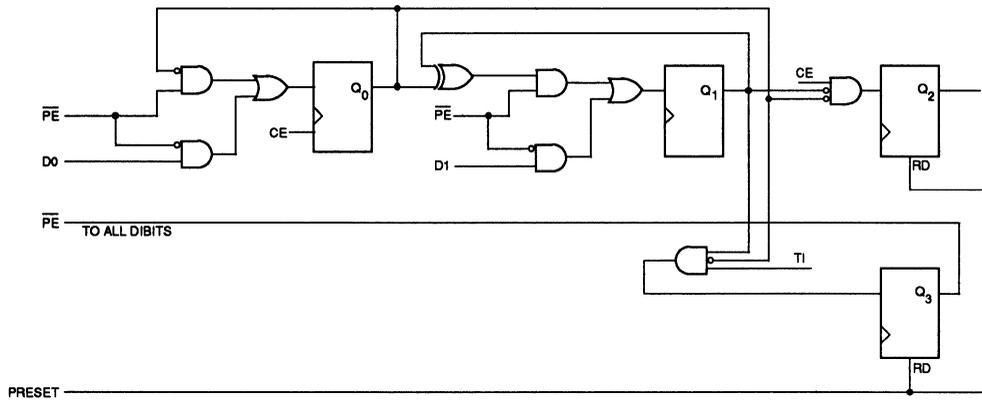
Different from conventional synchronous counters, the speed of this design is independent of its length. All speed-critical paths are single-level, their interconnect delay can be kept below 9ns, which means that a -70 device can count at a 40 MHz rate (worst case).





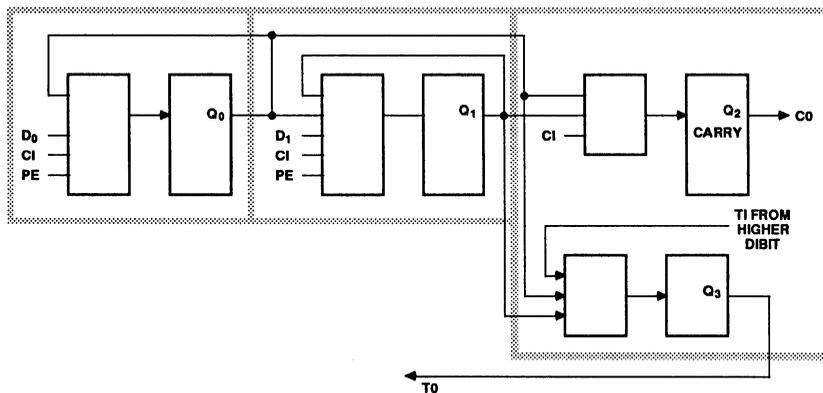
Any Dabit Except the Least Significant

1145 04



Least Significant Dabit

1145 05



Synchronous Presettable Counter—2 Bits in 3 CLBs

1145 03

## Application Brief

An 8-digit frequency counter is a good example to demonstrate the capabilities of the XC3000-family of Programmable Gate Arrays.

An on-chip 1 MHz Xtal oscillator generates a one second time-base. The unknown frequency is counted for one second in a counter with a capacity of eight BCD digits. At the end of the one second period the counter content is transferred into 32 output latches, and the counter is reset to start a new count period. The 32 outputs can drive a BCD encoded display, or they can be encoded on-chip into eight 7-segment outputs either parallel or time multiplexed.

The oscillator (minus Xtal) is provided on the 3000-series chip. The six-decade time-base divider is implemented as six blocks of 4-bit decimal counters, each using two Configurable Logic Blocks (CLBs) forming a synchronous counter. The ripple-clock delay between these six blocks is only a small fraction of the oscillator period and can thus be eliminated through re-synchronization in the control logic.

The eight frequency-counting decades are also implemented as 8 blocks of synchronous BCD counters each using two CLBs. Input resolution is 50 MHz worst case. The 32 latches are best implemented as edge-triggered flip-flops and are contained in the I/O Blocks. The control and synchronization logic consists of a 2-bit digital differentiator driven from the most significant bit of the time-base

counter clocked by the MHz oscillator. The whole control logic thus fits into one CLB.

The complete frequency counter with 32 latched 3-state outputs thus uses slightly less than one half of the resources of an XC3020. (It uses 29 of the 64 available CLBs plus 32 of the 64 available IOBs and 37 of the I/O pins). The other half of the XC3020 might be used to encode the counter output in the popular 7-segment format. This would use an additional 32 CLBs and increase the IOB count to 56, the I/O pin count to 61. Multiplexing the eight digits would be more efficient. It would use only 16 CLBs for storing and shifting the BCD information, four CLBs to encode it into the 7-segment format and four CLBs to generate the 1-of-8 output. Total CLB usage is thus  $29 + 24 = 53$ , IOB usage is down to 15 and total I/O pin usage is down to 19.

This frequency counter can also be implemented in the 64 CLBs of an XC2064, using: 24 CLBs for the time base counter, 32 CLBs for the frequency counter, and 4 CLBs for synchronization and control. 24 of the 32 latches can be implemented in the combinatorial logic of the three LSBs of each frequency counter decade. The MSB of each decade cannot implement the latch. Those 8 latches, therefore, occupy the four remaining CLBs. None of the 58 input latches is used in this design.

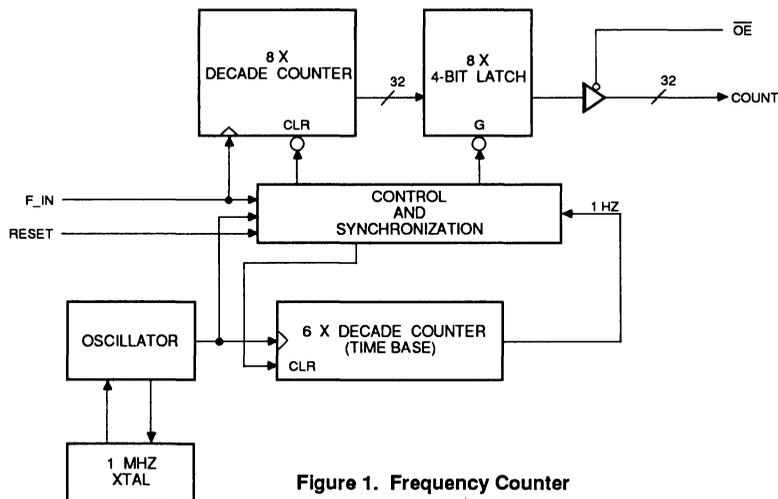


Figure 1. Frequency Counter

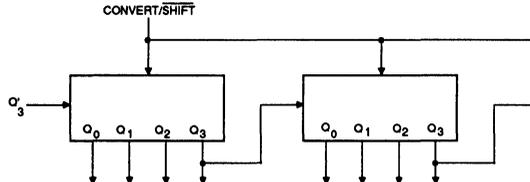


Figure 1. Binary to BCD (MSB First)

1146 03

The LCA architecture with its powerful function generators evenly interspersed between flip-flops lends itself very well to serial code conversion, where data is shifted into a register in one format, and shifted out of the same register in a converted format.

A binary to BCD converter requires 3 CLBs for every 4 bits of BCD output (i.e., for every digit). Data is shifted in serially, most significant bit first. Each shift thus doubles the content of the register.

In order to stay a valid BCD number, a 4-bit number of 5 or greater must not just be shifted, but must be converted into the proper BCD representation of its doubled value: A "one" is shifted into the next higher decade and the 5 is converted into a 0, a 6 into a 2, a 7 into a 4, an 8 into a 6, a 9 into an 8. When the binary LSB has been shifted in, BCD data is available in parallel form, or it can be shifted out serially with the conversion logic disabled.

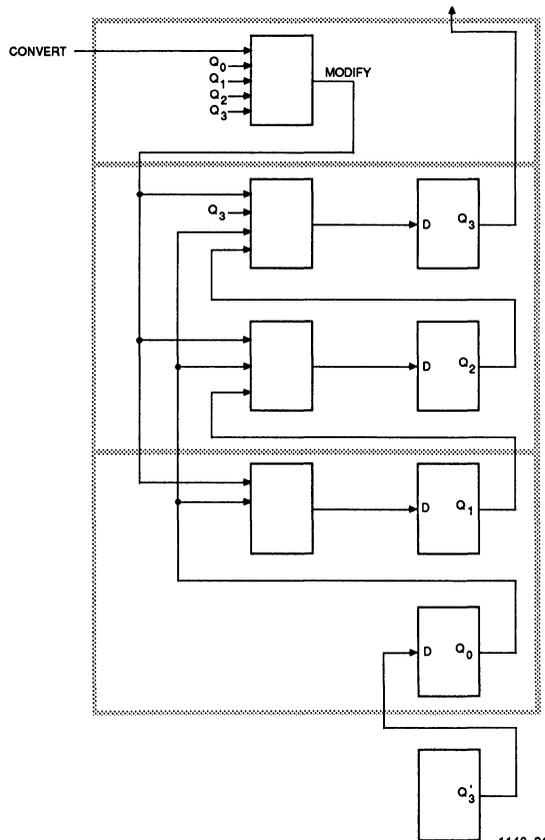
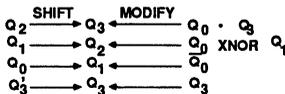


Figure 2. Binary to BCD converter 3 CLB's per 4 Bits:  
MSB First

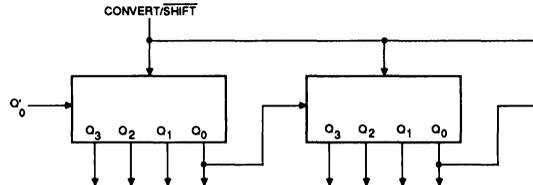
1146 01a

MODIFY: 5 → 0, 6 → 2, 7 → 4, 8 → 6, 9 → 8



1146 01b

## Application Brief



1146 04

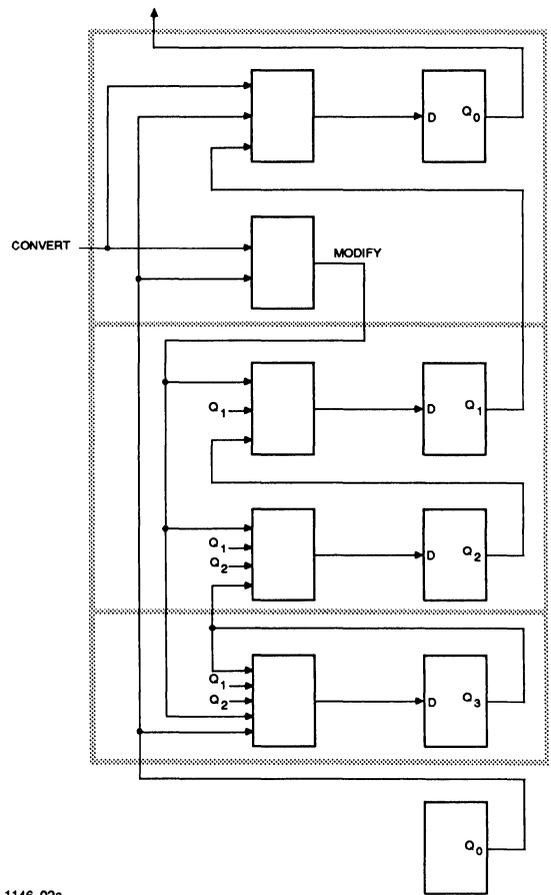
Figure 1. BCD to Binary (LSB First)

The LCA architecture with its powerful function generators evenly interspersed between flip-flops lends itself very well to serial code conversion, where data is shifted into a register in one format, and shifted out of the same register in a converted format.

A BCD-to-binary converter requires 3 CLBs per digit. BCD data is shifted in, least significant bit first. Once the complete BCD word has been shifted in, the conversion process begins, shifting out binary data, LSB first.

Each shift divides the content by two. When the LSB of a BCD digit is a "one", shifting it one position down would give it a weight of 8 in the lower decade instead of the weight of 5 appropriate for a 10 divided by 2. A value of 3 is therefore subtracted from the content of the decade whenever a "one" is being shifted into it.

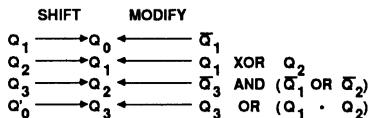
This design can be made smaller and faster by starting the conversion before the most significant BCD digit is being shifted in. Since these converters can be laid out with very short interconnect delays, they can operate at up to 60% of the specified toggle frequency, i.e. 42 MHz for the -70 parts.



1146 02a

Figure 2. BCD to Binary converter 3 CLB's per 4 Bits:  
LSB First

MODIFY: 0 → 5, 2 → 6, 4 → 7, 6 → 8, 8 → 9



1146 02b

## FIXED PATTERN DETECTOR

This circuit compares a serial bit-stream against a predetermined (configured) pattern. Two bits are compared in each XC3000-series CLB. The outputs of the comparator are ANDed in with 3-state buffers on a long line.

Data is shifted through DIN into the Y-flip-flop, then shifted through the upper half of the combinatorial array into the X-flip-flop of the same CLB. From there it is routed to the DIN input of the next CLB.

The lower half of the combinatorial array compares the content of the two flip-flops against data supplied on the A and D inputs. A match is indicated on the G output and routed to a 3-state buffer driving a long line.

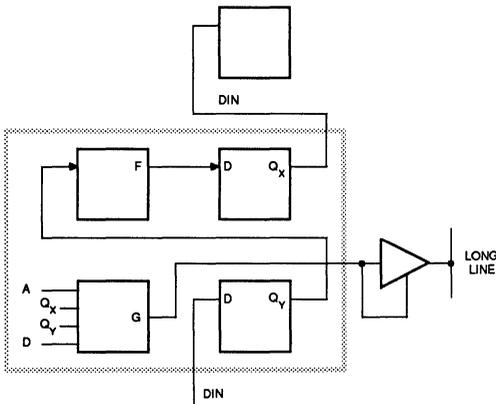
## DYNAMIC PATTERN DETECTOR OR CORRELATOR

This circuit compares a serial bit stream against a previ-

ously shifted-in pattern, using only one XC3000-series-CLB per pattern bit. The output of the comparators are ANDed with 3-state buffers on a long line. The desired pattern is first shifted through the DIN input into the Y-flip-flop, and then routed to the DIN input of the next CLB.

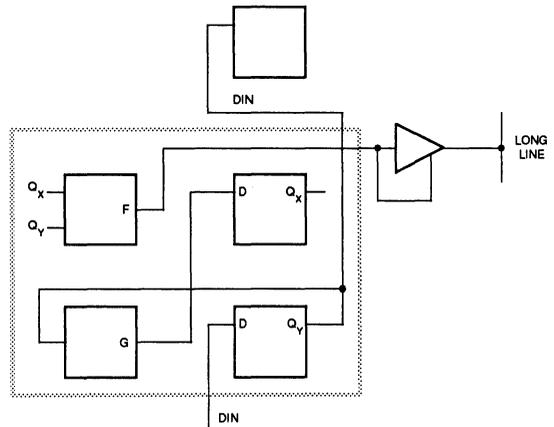
When the complete pattern has been shifted in, it is transferred with one clock pulse to the X-flip-flops, using the lower half of the function generator. Data to be detected is then shifted in through the DIN input into the Y-flip-flop, and from there to the DIN input of the next CLB. The upper half of the function generator compares the content of Qx and Qy, and indicates a match on the CLB output. For identity comparison, these outputs are ANDed through 3-state buffers driving a long line.

This circuit can also be used as a correlator, in which case the outputs must be summed in a Wallace-type adder.



1147 01

Figure 1. Fixed Pattern Detector



1147 02

Figure 2. Serial Comparator Finds Pattern Match or Correlates Patterns

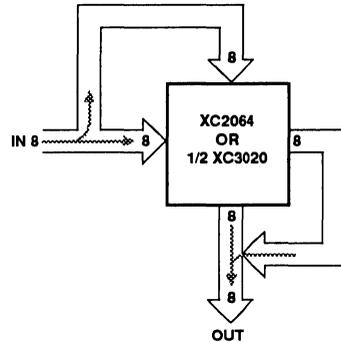
Pulse Code Modulation (PCM) has become the dominating encoding method in digital telephony. Analog signals are sampled at 8 kHz and represented by their 8-bit digital equivalent, using a logarithmic encoding scheme,  $\mu$ -law in the US and Japan, A-Law in the rest of the world using the CCITT standard.

These eight bits are usually transmitted serially (the T1 standard time-multiplexes 24 channels on a single wire at 1.544 MHz. The CCITT standard time-multiplexes 32 channels at 2.048 MHz.

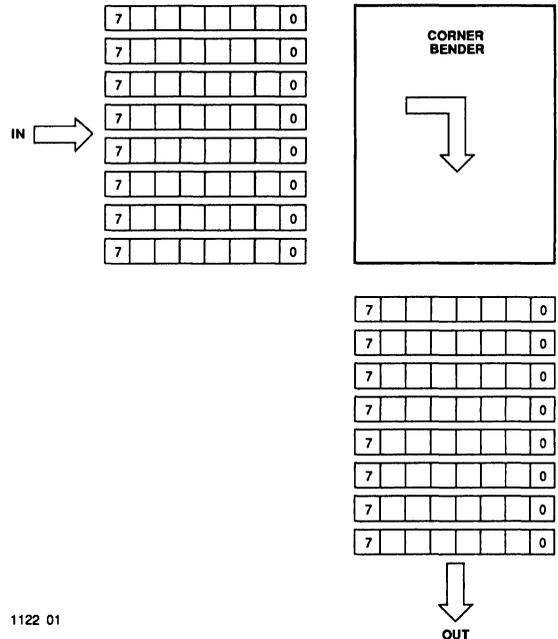
In the central office or PBX, however, the eight bits representing one particular sample must be routed together. The telephone system thus uses a large number of serial-to-parallel and parallel-to-serial converters, all operating on 8-bit words, all running synchronously. Eight S-P converters with 8 data inputs and 8 data outputs can easily be combined in one package. Eight serial data streams are shifted in simultaneously. After eight clock pulses the eight serial words can be shifted out in parallel, one word per clock pulse, and new serial bits can be shifted in simultaneously. It is interesting to note that the same circuit can also accept parallel words and shift them out in eight serial streams. The difference between S-P and P-S is not in the circuit, but in the mind of the beholder.

Such a “Corner Bender” is available as a standard part, the Plessey MJ 1410 8-Bit Format Converter. Its drawbacks are high power consumption (max 500 mW) and slow speed (2.4 MHz guaranteed worst case), a result of its nMOS heritage.

This design can be simplified and made to fit into an XC2064 or half an XC3020.



1122 03



1122 01

## “Corner Bender” or 8-Bit Format Converter

The LCA implementation of a 2-dimensional shift register is straight forward:

A common clock drives all flip-flops, organized in an 8x8 array. In mode A each flip-flop receives data from its “left” neighbor, in mode B each flip-flop receives data from its neighbor above.

For the first eight clock pulses the array is in mode A, receiving 8 bit streams and right-shifting them into the array. For the next eight clock pulses, the array is in mode B, down-shifting the previously received 64 bits.

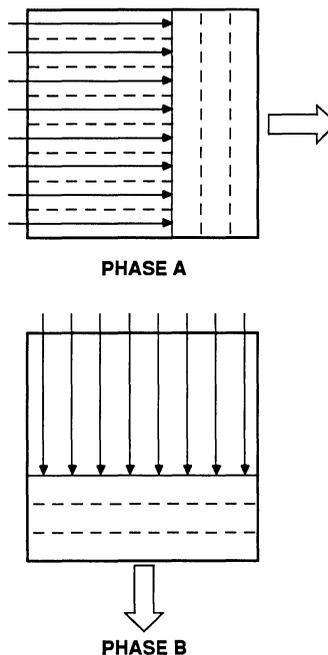
New serial data can be shifted in from one side while old parallel data is being shifted out at the opposite side. There is no need for any of the additional flip-flops required by the older designs.

After eight clock pulses the mode control is again changed to A and old data is shifted out on the right side while new data is shifted in from the left.

This design uses only 64 flip-flops, and a mode control signal derived from a divide-by-8 counter.

The physical routing of the input signals can be done on-chip, but the eight bottom output pins must externally be wire-ored with the eight right-hand outputs.

The design fits exactly into one XC2064 or into half of an XC3020 and can run at up to 35 MHz.



1122 02

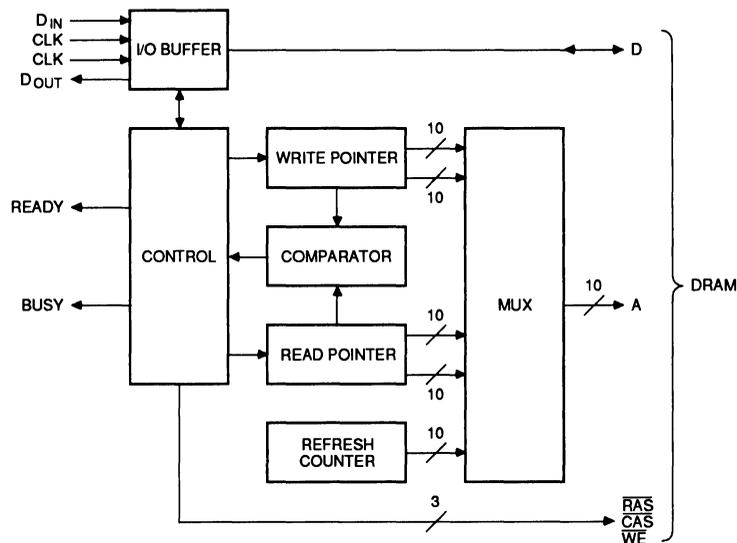
A bit serial FIFO buffer is a general-purpose tool to relieve system bottlenecks, e.g., in LANs, in communications, and in the interface between computers and peripherals. Small FIFOs are usually designed as asynchronous shift registers, but a larger FIFO with more than 256 locations is better implemented as a controller plus a two-port RAM, or as a controller plus a single port RAM, either SRAM or DRAM.

SRAMs are fast and easy to use, but at least four times more expensive than DRAMs of equivalent size. Dynamic RAMs offer low cost data storage, but require complex timing and address multiplexing, which makes them unattractive in small designs. For FIFOs with more than 256K bit capacity, a DRAM offers the lowest cost solution, if the controller can be implemented in a compact and cost-effective way. A Xilinx XC3020 Logic Cell Array can easily perform all the control and addressing functions with many gates left over for additional features.

This FIFO DRAM controller consists of:

- An input/output buffer with synchronizing logic
- A 20-bit write pointer (counter)
- A 20-bit read pointer (counter)
- A 20-bit full/empty comparator
- A 10-bit refresh counter
- A 5-to-1, 10-bit address multiplexer
- Control and arbitration logic

The write pointer defines the memory location where the incoming data is being written, the read pointer defines the memory location where the next data can be read. The identity comparator signals when the FIFO is getting full or empty.



1130 01

Figure 1. Megabit FIFO Controller in an XC3020

When write and read pointer become identical as a result of a write operation, then the FIFO is full, and further write operation must be prevented until data has been read out. When the two pointers become identical as a result of a read operation, then the FIFO is empty and further read operation must be prevented until new data has been written in. With a single-port RAM, read and write operations must be inherently sequential, and there is no danger of confusing the full and empty state, a problem that has plagued some two-port designs.

A straightforward design would use synchronous binary counters for the two pointers, but it is far more efficient to use linear shift register counters. Such counters require far less logic and are faster since they avoid the carry propagation problems of binary counters. LSR counters have two peculiarities: they count in a pseudo-random sequence and they usually skip one state, i.e., a 20-bit LSR counter repeats after  $2^{20}-1$  clock pulses. In a FIFO Controller, both these features are irrelevant, the address sequence is arbitrary, provided both counter sequence identically. The loss of one memory location is more than compensated by the two bits (one incoming, one outgoing) stored in the controller.

This design fits two shift register counter bits in one 3000-series CLB, the identity comparator uses the combinatorial portion of the same CLB.

The RAS/CAS multiplexing of the 20-bit address is performed without any logic by tapping every other bit of the shift register counter and using the 10 outputs before the incrementing shift as Row address, after the incrementing shift as Column address. (The Column address of any position is thus identical with the Row address of the following position, but since the binary sequence of a shift

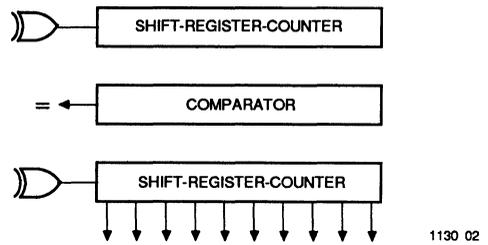


Figure 2. Shift-Register-Counter and Free Row-Column MUX

register counter is pseudo-random anyhow, this is no problem. It's an elegant and efficient trick).

Both 20-bit pointers, plus their 20-bit identity comparator, plus the Row/Column multiplexer thus fit into only 20 CLBs; refresh timer and refresh address counter and multiplexer use another 15 CLBs and the data buffer plus control and arbitration logic might take another 15 CLBs, for a total of 50 CLBs, an easy fit in an XC3020.

This design can easily be modified for 256K DRAMs. Other variations are: multiple parallel bits, e.g., byte-parallel operation, or byte parallel storage with bit-serial I/O. The latter case requires special attention when the FIFO is emptied after a non-integer number of bytes had been entered, requiring direct communication between the input Serial-to-Parallel converter and the output P/S converter.

This applications brief shows that the XC3020 can be programmed to control one or a few DRAMs as a large FIFO of up to a Megabyte, with data rates up to 16 Mbps serially or 2 Megabytes per second byte-parallel.

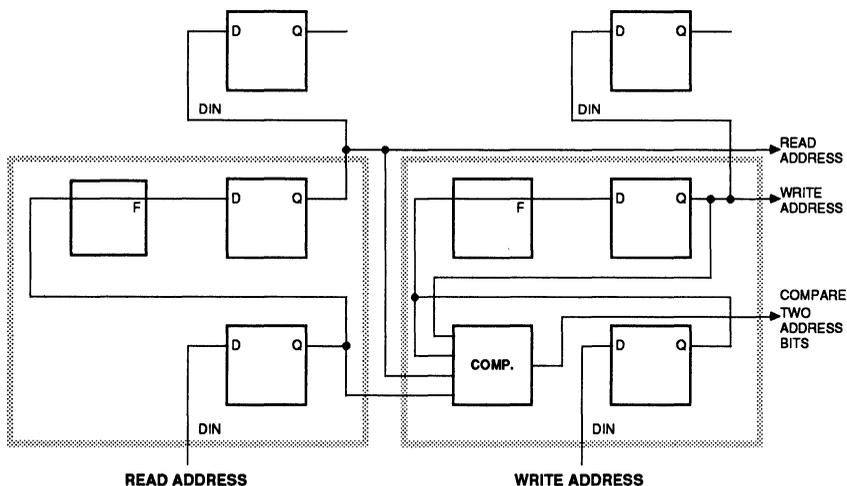


Figure 3. 2-Bit Slice of Two Counters and Comparator in Two CLBs

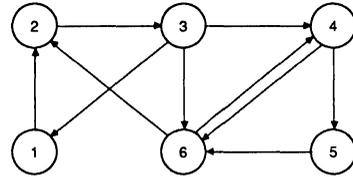
LCAs offer an abundance of flip-flops, but have limited fan-in on their gating structure. This suggests a different, inherently simpler, approach to the design of State Machines.

Dedicating one flip-flop to each state eliminates the need to decode the present state and encode the target state. The design thus becomes a one-on-one implementation of a synchronous Moore-type State Machine diagram:

Each *state* is represented by a flip-flop in a CLB. Its input function generator is used to OR the different conditions that might set this particular state. Out of the max 5 inputs in the 3000 series, one will usually be dedicated to the flip-flop itself, a second one to the reset condition as described later. This leaves 3 inputs. If a state has more inputs, they must be ORed in an additional cell or-in extreme cases- by the wired OR of a long line.

Each *conditional path* between two states is implemented by the combinatorial function in one CLB. Since the CLB can implement any function of 5 variables, there is no limit to the interdependence of these variables.

This approach uses more than the minimum number of flip-flops (a 16-state machine might be built with just four flip-flops, this approach uses 16). It is, therefore, mandatory to include a recovery mechanism, in case more than one state might get set accidentally. This recovery is implemented with a common clear input, distributed via a long line to the input structure of all state flip-flops. As this line is driven low by any decoded transition output, it



6	STATES
10	CONDITIONAL PATHS
16	CLBS

1123 01

### One CLB per State Plus One CLB per Conditional Path

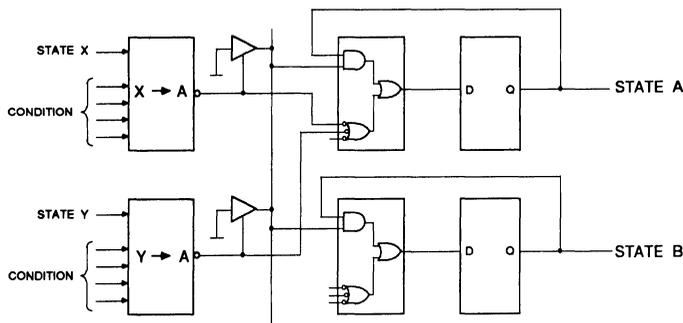
breaks the feedback path at each input structure, so that no state flip-flop will stay set, except the one that is just being set.

The advantage of this design is its simplicity: each state uses one CLB, with one synchronization flip-flop left free. Each arrow uses the combinatorial portion of one CLB.

Total number of blocks required = M + N  
 M = # of states, N = # of transition lines

Limitations to the basic design: any condition must be 4 signals or less, the number of inputs to any state must be 3 lines or less (add one LCB to increase either of these limitations by 4 units).

Maximum speed is lay-out dependent, but will usually be higher than 30 MHz for a -70 LCA.



1123 02

Synchronous State Machine



---

# Programmable Gate Arrays and Self-Diagnosing Hardware

---

Application Brief BY RICHARD B. RAVEL

---

## SELF-DIAGNOSING HARDWARE

Most designers would agree that it is desirable to incorporate self-diagnostics into their circuit boards, and they would do it more often if the cost of additional components and board space were acceptable. It is obviously best to consider diagnostics at the beginning of a project so that the board is designed with testability in mind. This not only makes a board more manufacturable, but it makes it easier to find failures in the end-user environment.

Programmable Gate Arrays are used in many different applications, and have the unique capability of having their specific functions defined by the systems in which they reside. Xilinx Logic Cell Arrays (LCAs) can also be re-programmed, in-system, as many times as necessary. This ability to dynamically re-configure the logic of the LCA makes board-level self-diagnostics a practical goal. An LCA can perform diagnostic functions at power-up or in test modes, and perform normal functions when the board is determined to be operational. (See the application note on "Configuring Xilinx Logic Cell Arrays" in the Programmable Gate Array User's Guide.) This approach to diagnostics based on reprogrammable gate arrays adds no additional cost to the circuit board.

This concept is really not new. Board-level self-diagnostics became popular with the advent of microprocessors. A special diagnostic program written for the microprocessor and stored in its normal EPROM could be invoked at power-up, by the press of a button, or by a special command. This approach adds little cost to the system because it requires only a small amount of EPROM storage. For example, when a PC is initially powered-up, all of the system RAM is tested. Further initialization of the PC will not take place unless this memory is 100% functional. If there is a memory failure, it can be isolated to the specific IC. The LCA allows an extension of this idea. The microprocessor will still have some special programs for diagnostics, but now the diagnostics can extend well beyond the immediate reach of the microprocessor, without adding circuitry just for this purpose.

## DIAGNOSTIC HARDWARE

The circuitry which is most easily diagnosable is that which is immediately accessible by the microprocessor. In many cases, however, some of this circuitry cannot be tested directly due to the specific design. Testing this logic, as well as other unrelated circuitry, requires additional logic on the board specifically for the purpose of diagnostics. This is typically not done because of board space and cost considerations.

The logic which typically surrounds a microprocessor is the I/O control, memory control, bus control, and interrupt control logic. The peripheral control logic is the most difficult to diagnose using the microprocessor. I/O control functions are usually implemented with dedicated peripheral controller chips, since they are cost effective and readily available. If they do not give the designer enough flexibility to perform all of the required functions, logic is added to the board. Some peripheral controller chips allow the designer to include diagnostic readback firmware. This firmware, however, would usually not include access to the supplemental circuitry which might be required. Even when this readback capability is available, its scope is limited.

Peripheral control logic is usually diagnosable by writing special programs for the microprocessor, and adding special circuitry that the microprocessor can access specifically for this purpose. Adding special logic for this purpose is certainly not desirable and, with the use of Logic Cell Arrays, not necessary.

The Logic Cell Arrays can perform many different peripheral control functions, and can also be the primary interface between a microprocessor and its peripherals. In these microprocessor designs, as previously mentioned, there would usually be circuitry on the board to which the microprocessor may or may not have immediate access. The LCA, as the bus and I/O controller for example, would easily be able to access this logic as an extension of the microprocessor.

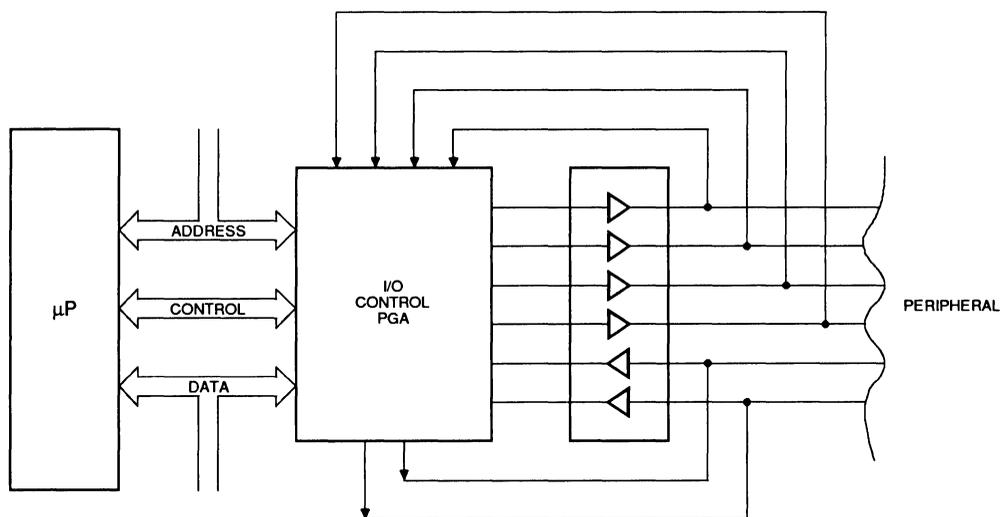
## SELF-TEST TECHNIQUES

LCAs can be used to implement hardware diagnostics. When the board is initially powered-up, the Logic Cell Array can be programmed with a special diagnostic configuration. The LCA can then be used in conjunction with the microprocessor to test the peripheral circuitry. This LCA configuration can include the ability to communicate status information about the peripherals and other circuitry to the microprocessor that might otherwise require additional logic.

The remainder of this paper will focus on several examples of how the Logic Cell Array can be used to perform board-level self-diagnostics.

## LOOPBACK

Many designs have special drivers and receivers on a board to which there is no direct access. A special loopback test connector is usually needed to test these drivers and receivers. This connector must be installed before and removed after testing. This usually means that this test is only performed as a last resort, when all other tests have failed to find a fault. The Logic Cell Array's user-configurable interconnections allow the drivers and receivers to be tested without any additional test connectors or manual intervention. To fully test this circuitry, it is only necessary to connect traces on the printed circuit board from the drivers and receivers to I/O pins on the Programmable Gate Array. This allows the LCA, with a diagnostic configuration loaded, to drive the receivers and to read the data from the drivers without the use of a loopback test connector. Refer to Figure 1.



1131 01

Figure 1. Diagram of LCA Used in LOOPBACK Testing

**TESTING I/O AND MEMORY ERROR DETECTION CIRCUITRY**

A microprocessor can use the LCA to drive the peripherals and additional supporting logic in non-standard ways. This is often valuable in diagnosing circuitry. For example, during normal operation of a serial communications channel, it is not possible to force an error in the transmission of the data. This can easily be done, however, when an LCA is used as an I/O controller. It can be programmed with a special diagnostic configuration that can force parity errors, overrun errors, and CRC/checksum errors in the data stream which should be caught by the error detection circuitry. This, along with special diagnostic firmware for a microprocessor, allows full testing of serial (or parallel) communications channels.

This same concept can be applied to testing standard memory when the LCA is performing memory control functions or has write access to the memory. With a diagnostic configuration in the LCA, it can force data into the memory with incorrect parity or check bits. The normal circuitry and firmware for reading the memory should then detect the errors, and the operation of the error correcting logic can be verified.

As soon as the microprocessor has determined that the

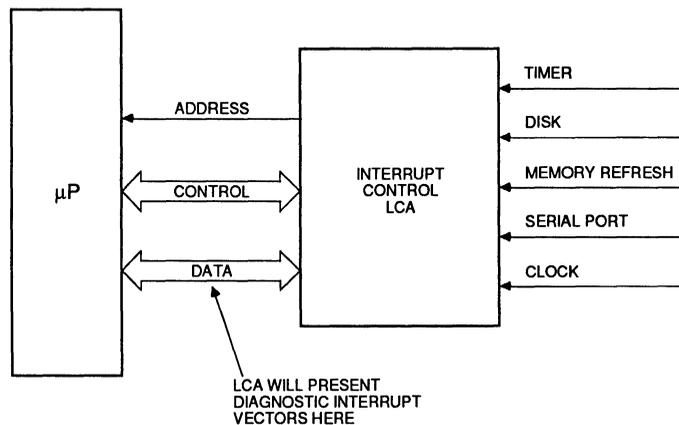
board is fully functional, it can re-program the LCAs for their normal functions and the board can begin normal operation.

**INTERRUPT VECTORS**

Interrupt circuitry is also difficult to test, as there is not always a straight-forward method of generating all of the interrupt requests. With a Logic Cell Array as the interrupt controller in the system, it can be configured with a special test configuration that can sequence through all of the different interrupts. It can even generate multiple interrupts in sequences that test interrupt prioritization logic. Refer to Figure 2.

**TIMEOUT INTERRUPTS**

Some systems include timeout interrupts such as watch-dog timers that do not occur in normal operation. This type of timeout interrupt is difficult to generate during normal operation. Again, using a Programmable Gate Array as an interrupt controller allows the designer to easily force this type of interrupt and verify that the detection circuitry and error recovery routines are functioning correctly.



**Figure 2. LCA Used as an Interrupt Controller**

1131 02

IBM's new general-purpose microcomputer, the Personal System 2, is available in several models, from the low-end Model 25 to the high-end Model 80. These third-generation PCs have several new and innovative features, including 3 1/2 inch floppy disk drives, high-resolution VGA graphics, and a 20 MHz 80386 processor as the main engine for the Model 80. Among the most interesting features is the Micro Channel interface, the bus specification for the interface between the system and adapter cards. The Micro Channel's streamlined characteristics and flexibility provide PS/2 designers and users with many advantages over previous PC architectures.

One key aspect of this architecture is the ability to configure the system without the need for DIP switches on the bus adapter cards. Defined with System Configuration

Utilities, an add-on card's addressing and other optional configuration data are established and stored in CMOS battery-backed memory on the main board.

Upon power-up, this information is loaded into Programmable Option Select (POS) registers residing on the adapter cards.

Figure 1 indicates one way in which a Programmable Gate Array can be used for the POS register section of a Micro Channel adapter card. The Micro Channel interface includes logic to decode the address, status, and control signals associated with the bus to identify the appropriate POS register to be accessed. These signals determine if the card is being addressed, and whether the current operation is a read or write.

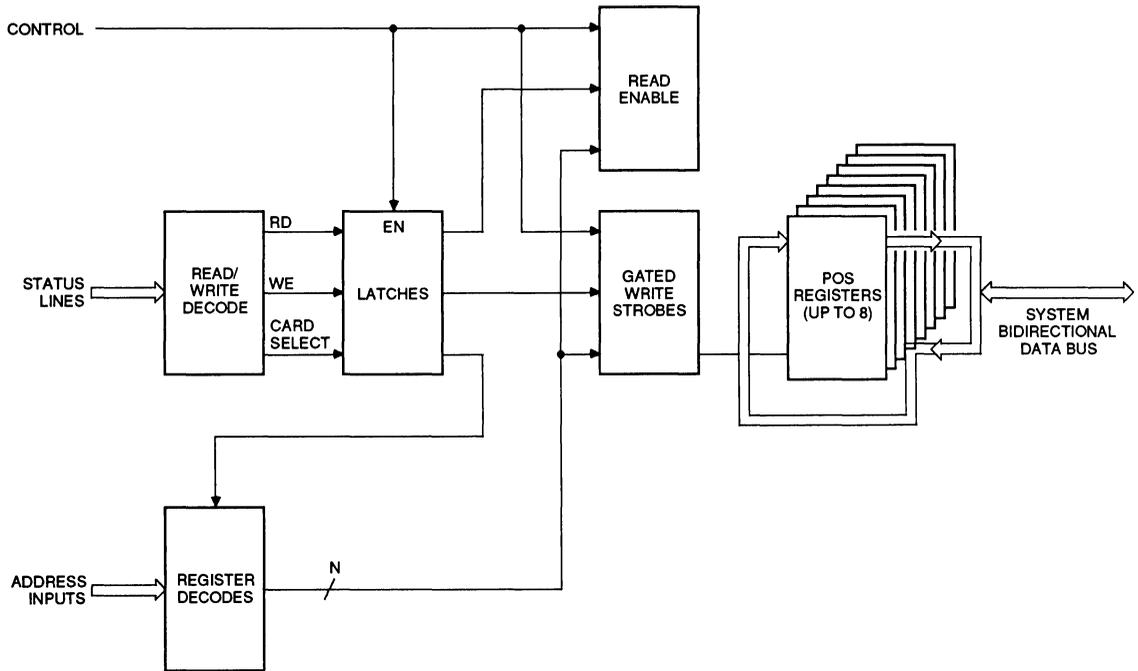


Figure 1. Micro Channel Interface Block Diagram

The Micro Channel specification reserves two POS registers for the upper and lower bytes of the Adapter Identification (ID). Six other byte-wide POS registers can hold additional configuration information; some of the bits within these are specifically dedicated to channel status information. Some applications will require the use of only portions of these six registers.

A second key aspect of the Micro Channel architecture is its ability to arbitrate the bus access of multiple adapters. The Micro Channel specification clearly defines the logic required for this arbitration. Each adapter in the system is assigned a priority level. These levels vary from the highest priority "2" to the lowest priority "F". This "-2, -1, 0, 1, 2...A, B, ...F" scheme defines unique priority levels. The higher levels are primarily used for memory refresh or error recovery. The lower levels are reserved for the System Board processor and spares. The middle levels are used for DNA Ports 0-7, typically used for high speed transfers. The priority level assigned to any adapter is stored in one of its POS register nibbles. The arbitration logic must be very fast in order to grant control of the bus

to the adapter with the highest priority.

As can be seen by the logic in Figure 2, this priority level (ARB ID 0:3) is driven onto the bus via an open collector driver. The logic then turns around and accepts the driven bus as input. The cycle may repeat a few times before the adapter with the highest priority level actually gains control of the bus. For proper operation each half of the cycle must complete in 50 ns, a performance that can be achieved in the 70 MHz Programmable Gate Array devices.

Implementation of the POS registers, arbitration, logic and control sections typically requires only 1/3 to 2/3 of a single XC2018 or XC3020; the remainder of the PGA is available for implementing the unique functionality of the specific adapter card. Some Xilinx users have developed the standard interface and stored it as a recallable macro function in the Xilinx development system. Applications including hard disk controllers, communication controllers, and specialized memory controllers have been developed for the PS/2 using Xilinx Programmable Gate Arrays.

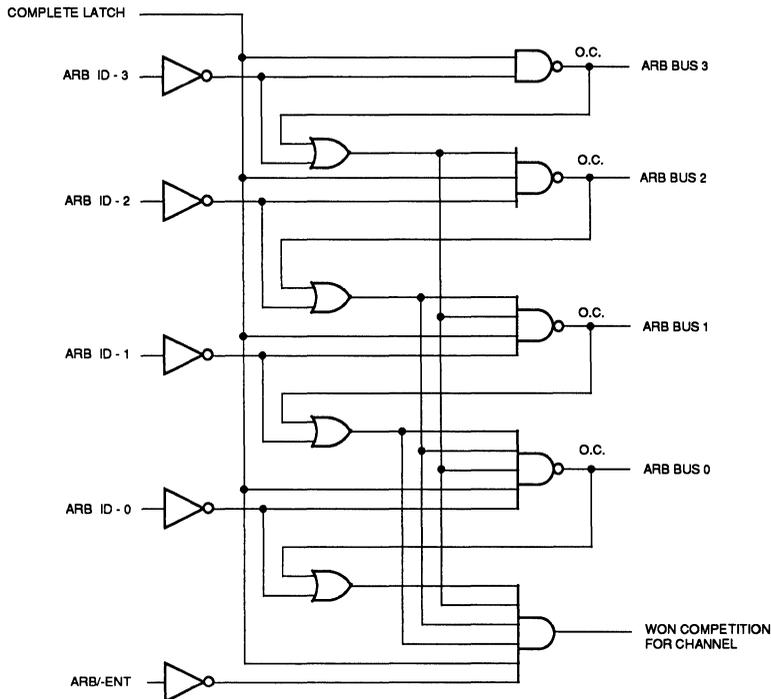


Figure 2. Local Arbitration Logic

Bar code readers have become familiar to the average consumer due to their widespread use in point-of-sale systems, such as those used in most grocery stores. Additionally, many industrial applications are now using bar codes to identify materials in various phases of manufacturing, inventory and distribution. Bar codes can be automatically read and processed to provide for computerized control and optimization of virtually all phases of a manufacturing process. In many industrial applications, the performance requirements of the bar code system are significantly higher than those of the point-of-sale system.

As an example, consider an industrial application where material on a conveyor is being scanned to read bar code data. If the conveyor is moving at 5 meters/second, and a bar code label with 30 bars is 25 mm long, that label passes a point scanner in 5 milliseconds, or 166 microseconds per bar. If the minimum bar width is 12% of the average time period, it must be detected and its width determined in 20 microseconds. For a maximum bar width five times the minimum, the time period is 100 microseconds. If a multiple scan system is being used, the active period for a bar is reduced proportional to the scan rate.

For this type of scanning system, some pre-processing of the scan data is required to insure that a processor can decode the label into code numbers and process them accordingly. In addition, systems may be required to operate with a variety of different bar code scanners or sensors for different objects or labels, at different scan rates. Traditionally, different logic and micro computer coding dedicated to each application was designed.

Programmable Gate Arrays provide a method of implementing the scanner specific interface logic in a cost effective, compact manner. At the same time, system design flexibility is provided while meeting the performance constraints of the system. Designers of bar code processing systems can design a single interface card that utilizes the programmability of the PGA to define the logic for a specific scanner in software as part of the system initialization.

The block diagram in Figure 1 shows the functions performed in the PGA. For each type of sensor, the specific controls and data signals have different functions and timing relationships. In addition, the amount of data can vary depending on the specific application. In different scanner interfaces, the counter control logic, edge detect circuitry and the operation of the black area and white area counters can be modified to fit the needs of the particular scanner. The bus control logic and the data loading into the interface memory remain virtually unchanged, providing a consistent interface to the processor.

Implementation of the interface logic can be accomplished with either an XC2000 or XC3000 family device. Because there are processor and memory data and address bus structures, the XC3000 family would provide a simpler solution. Even with a master clock rate of 10 or 15 MHz, the 16-bit counters for the black and white areas are easily implemented as up counters with reset capability. Edge detection is performed with either simple gating, or synchronization to the master clock. Memory FIFO control also involves counters, but these operate at a much slower rate. A new word is written to memory only after a bar in the target has been passed.

At the completion of a scan an End Of Label is signalled by forcing a data entry in the RAM of FF hex. An interrupt to the processor can be generated based on the End Of Label or End Of Scan condition. The microprocessor then normalizes the absolute timing information for black and white bars stored in the FIFO in order to extract the encoded data. This allows for variations in scan speed, scan angle, etc.

This type of interface represents only one of several methods that can be used to pass the information to the host processor. For other types of systems, different techniques may be used. Regardless of the technique, the flexibility of the PGA provides significant advantages over a fixed logic solution.

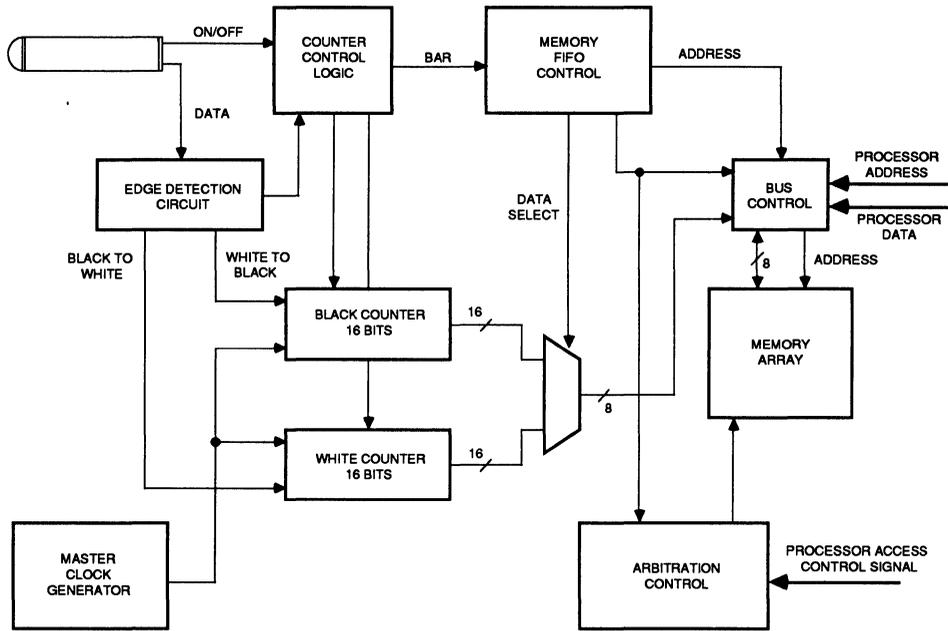


Figure 1. Bar Code Reader Interface

1126 01

## AN INTRODUCTION TO MEMORY CONTROL AND ERROR CORRECTION

The need to design memory controllers for systems that have a large amount of memory is a common design challenge that engineers must deal with today. Almost all large memory systems use dynamic random access memory (DRAM) because of its density and low cost. While designing large memory systems with static random access memory (SRAM), would make the design task easier, the drive to produce more cost effective products forces the engineer to design with DRAMs, despite their inherent drawbacks. The memory cell of a DRAM is a capacitor that holds a charge corresponding to the value of the data bit. Since all capacitors leak charge, a DRAM cell will gradually lose its charge, and its stored value, unless it is recharged. This recharging, known as refreshing, must typically be performed once every 2 to 4 ms depending on the DRAM. Refreshing is one of the DRAM controller's two primary functions. The other function is to arbitrate between requests for memory read and write accesses from the system's central processing unit and requirements for memory refreshes.

In addition to its need for periodic refreshing, the DRAM exhibits another problem that SRAM and other memory devices do not—greater susceptibility to soft errors. A *soft error* is the loss of a data bit in a memory cell in which the memory cell is not physically damaged. Rewriting the data in the cell corrects the error. This type of error is different from a *hard error* which is caused by a memory cell that has failed permanently. Soft errors in DRAMs are usually caused by alpha particles (helium nuclei), which are normally present in the atmosphere, but which are more often emitted by radioactive impurities in the IC packages of the DRAMs themselves. If an alpha particle hits a memory cell, it can corrupt the cell's charge, causing a data bit error. Most people believe that the likelihood of such an error is so low that it can be safely ignored. While this may have been true for the smaller memory systems of the past, it may no longer be so. The size of some memory systems today can make the likelihood of soft errors unacceptably high. The probability of a soft error can be reduced by device and packaging improvements and by reduction in signal noise. Another method of dealing with soft errors is

to incorporate error detection and correction into the memory system. This solution decreases system performance and adds the cost of redundant memory, but prevents parity errors from causing system failures.

## OPTIONS FOR DRAM CONTROLLER DESIGN

There are a number of options available to the engineer designing a memory system that requires DRAM control. (The following options apply to the design of error detection and correction circuits as well.) The simplest option is a standard off-the-shelf LSI memory controller. The manufacturers of these devices provide an integrated solution to DRAM control by combining CPU interface logic with the necessary memory access/memory refresh arbitration on a single chip. However, each memory system has unique timing and protocol requirements, and it is extremely difficult for these standard parts to accommodate the requirements of every system. This realization has driven many DRAM controller manufacturers to incorporate some degree of programmability into their parts to make them more flexible. Unfortunately, this has made the parts more complex, hungrier for power, and more expensive. Even so, they simply cannot meet every system's requirements without employing external "glue logic."

The need to match the DRAM controller to the specific requirements of the system has forced many engineers to consider two options for creating their own controllers: SSI/MSI packages or custom gate arrays. The use of SSI/MSI is low risk, but wastes space and power; while the use of the custom gate array provides a highly integrated solution, but at considerable risk and expense. Non-recurring engineering costs (NRE), testing and simulation costs, inventory risk, and a long design cycle make the custom gate array option unattractive for most designs. Recent architectural advances in high density user programmable logic have created a third option. Xilinx's 3000 family of programmable gate arrays brings unprecedented density to programmable logic, with devices containing as many 9000 usable gates. The architecture of the 3000 family devices make them particularly well-suited to memory controller applications.

**WHY IMPLEMENT A DRAM CONTROLLER WITH A PROGRAMMABLE GATE ARRAY?**

There are several reasons why one would want to design a DRAM controller with a Xilinx Logic Cell™ Array. First, the true programmability of the LCA gives the designer the freedom to design the DRAM controller to the exact specifications of the memory system. There is no need for the external "glue logic" often necessary with standard solutions, because any necessary design tweaking is implemented internally. The LCA solution has the advantage of the SSI/MSI or custom gate array solution in that it can be configured to meet unique system requirements. There is no loss in integration as with the SSI/MSI solution, and the cost and risks of the custom gate array solution can be avoided. Second, the density of the 3000 family of LCAs makes it possible to implement DRAM control and error detection/correction in a single LCA. This is traditionally a two chip solution using standard parts: a DRAM controller and a separate error correction and detection unit. It can of course be implemented in a single custom gate array, but again with the earlier caveats. Finally, the CMOS LCA consumes less power than traditional standard "programmable" controllers which are typically implemented in NMOS or bipolar processes.

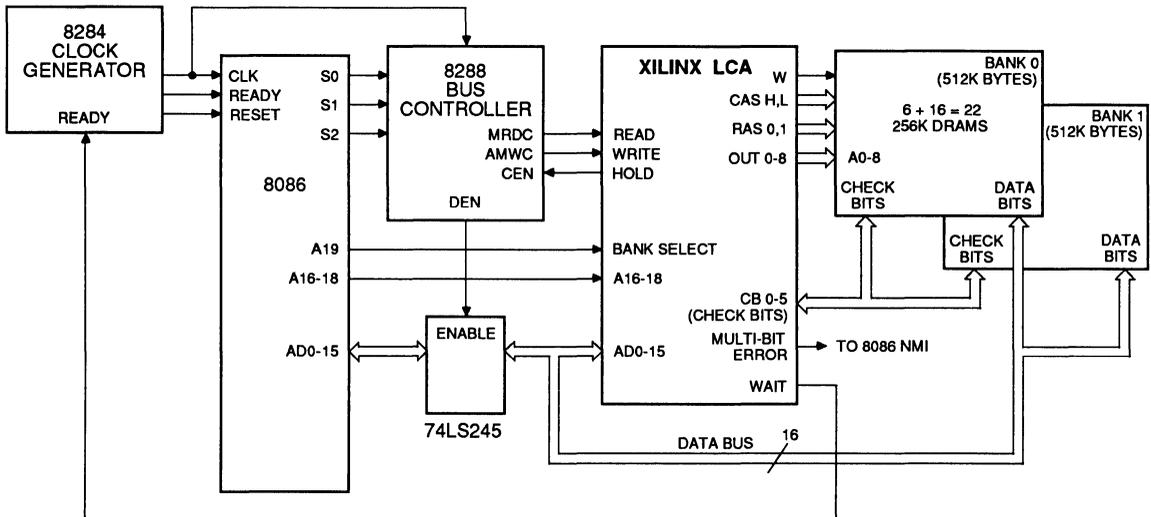
**DESIGN EXAMPLE**

The following design example shows the implementation of a DRAM controller and an error checker/corrector

(ECC) with an LCA. The example is an 8 MHz 8086-based system that directly addresses 1 MB of memory comprised of forty-four 256 KB DRAM chips: thirty-two for data and twelve for the correction bits. A single LCA can serve as both the DRAM controller and the ECC, which performs single bit error correction and double bit error detection. There are several features of the 3000 family architecture that make this design possible. These include five input configurable logic blocks (CLBs) with two storage elements, internal buses, and flexible input/output blocks (IOB).

**DESIGN OVERVIEW**

The DRAM Controller/ECC uses a 16 MHz clock synchronized with the processor's clock, and sits between the 8086 microprocessor with its 8288 bus controller and the system memory (Figure 1). The 8288 decodes the processor status lines (S2, S1, S0) and tells the DRAM Controller whether it is to perform a read or write access to the memory. (It is also possible to incorporate the bus controller logic into the larger LCAs). The DRAM Controller then performs the appropriate access issuing Row Address Strobe (RAS), Column Address Strobe (CAS), and Write, if necessary. The Error Checker and Corrector generates check bits on each write, and checks for and corrects errors on each read. The controller also signals the 8086 if the memory access requires a wait state or if a non-correctable error is detected.



1127 09

**Figure 1. System Overview**  
System overview of DRAM Controller with error correction and detection.

## SYSTEM TIMING

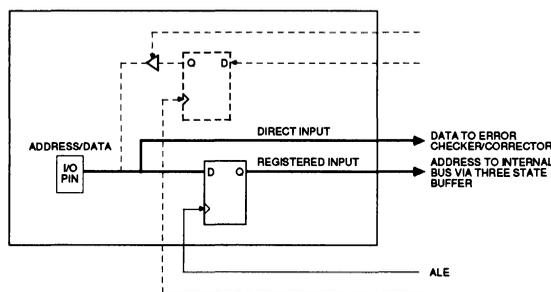
Figures 3a–3c show the timing involved in some of the different memory cycles. The Word Write (Figure 3a) requires no wait states as shown. The check bits from the ECC are written to memory along with the data. The Read cycle (Figures 3b & 3c) requires a minimum of one wait state. The insertion of a wait state is unavoidable because of the time it takes the 120 ns DRAMs to output the data. If the ECC detects no errors in the data, the WAIT signal is released and the read operation is completed. If an error is detected, the insertion of two more wait states is required to allow time to correct the error. The insertion of the two additional wait states effects system performance, but this is the trade-off for having error correction, which avoids the fatal system errors that occur with parity-checking-only solutions.

## DESIGN FOCUS

The 3000 family LCA architecture has a number of features that are essential to the DRAM controller design. The first such feature is the dual data input paths in the IOBs, one registered and one direct. This structure permits the

address and data on a multiplexed bus to be latched from the same I/O pin. Figure 4 is a bit sliced view of an IOB used to latch the multiplexed Address/Data bus. In this design, the address is latched into the IOB input flip-flop with the 8086's ALE. The data from the 8086 can enter the same input pin and go directly to the ECC circuit via the IOB direct input—there is no need for external latches.

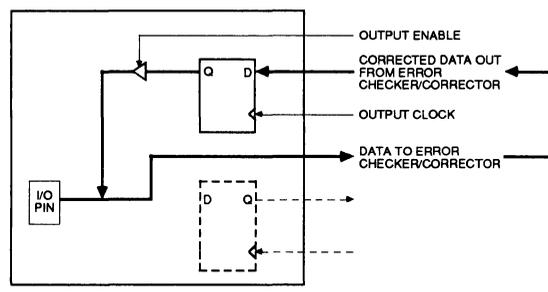
Another feature of the IOBs is the output flip-flops with three-state buffer enables. This feature permits bit error correction using only one I/O pin. Figure 5 shows a bit sliced view of how the ECC is accomplished. A memory read cycle provides the best example for showing the capabilities of the IOB structure. During a read, the IOB output is three-stated, permitting the DRAM data on the Data Bus to enter the ECC via the IOB direct input. If the ECC detects a data bit error, it corrects the error and latches the corrected data word into the output flip-flops of the IOBs. The Data Bus is then three-stated by turning off the DRAM outputs. The corrected word, latched in the outputs of the flip-flops, is then released onto the Data Bus by enabling the three-state buffer. This allows the corrected data to be read by the 8086 at the same time it is being written back to the DRAM.



1127 06

**Figure 4. Address and Data Latching**

Latching Data off a Multiplexed Address and Data Bus. The Input/Output block configuration shown above illustrates how the direct and registered inputs in the IOBs can be used to latch a multiplexed address/data bus into the LCA. The address is latched into the IOB flip-flop; the data flows directly into the ECC logic.



1127 07

**Figure 5. Data In and out through ECC**

Data Flow through the ECC The data from the bus goes into the LCA, where it is corrected in the ECC. The corrected data is then put back onto the bus via the IOB output flip-flop.

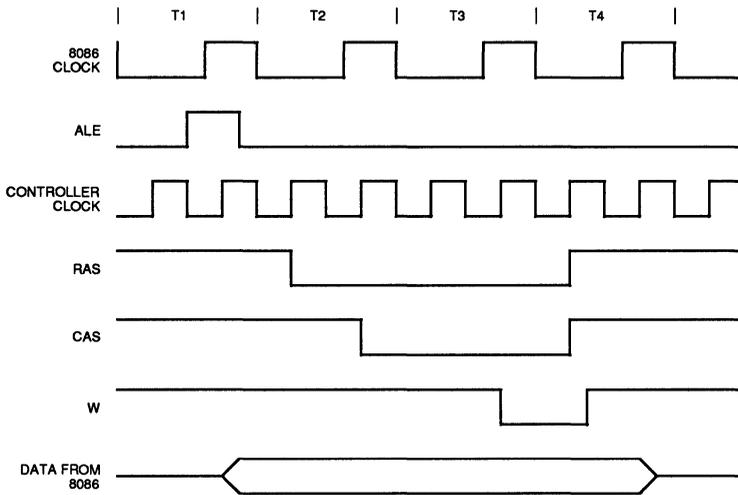


Figure 3a. Word Write Timing

1127 03

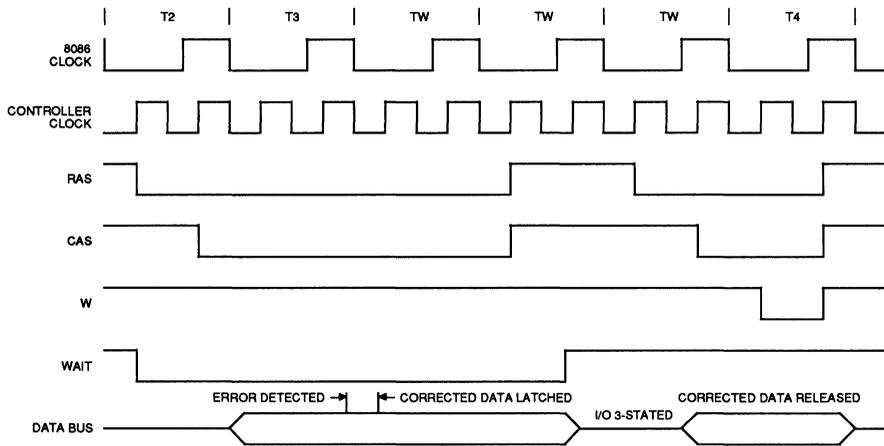


Figure 3b. Word Read Timing with Errors Detected

1127 04

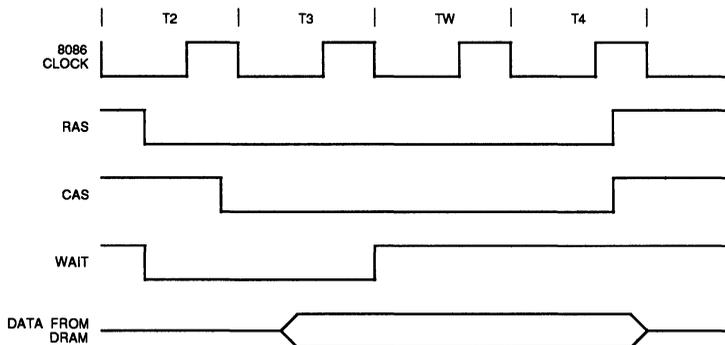


Figure 3c Word Read Timing with No Errors Detected

1127 05

Figure 2 is a block level diagram of the DRAM Controller and ECC that reside in the LCA. A functional description of each block follows:

The *refresh timer* is driven by the 16 MHz clock to provide a signal that tells the DRAM controller that the memory needs refreshing. Each of the 256 rows of memory in this system must be refreshed every 4 ms. The controller attempts to refresh eight rows every 125  $\mu$ sec, so that all 256 rows are refreshed in 4 ms. The refreshing technique employed in this design is a unique combination of burst and hidden refreshing to show the flexibility of the LCA-based solution. There is no need to force a system to conform to the constraints of an off-the-shelf part. The Hidden Refresh is performed when the 8086 is doing a read from or write to somewhere other than memory, like an I/O port. This involves giving the DRAM a refresh address from the *refresh address counter* via the *address selector* and a RAS pulse low from the *timing generator*. The Burst Refresh is performed only if it has not received its eight required refreshes during the 125  $\mu$ sec refresh period. When a Burst Refresh is required, the controller will isolate the memory from the 8086, insert wait states, and provide the number of refreshes it needs in order to complete the eight refreshes required during the refresh period.

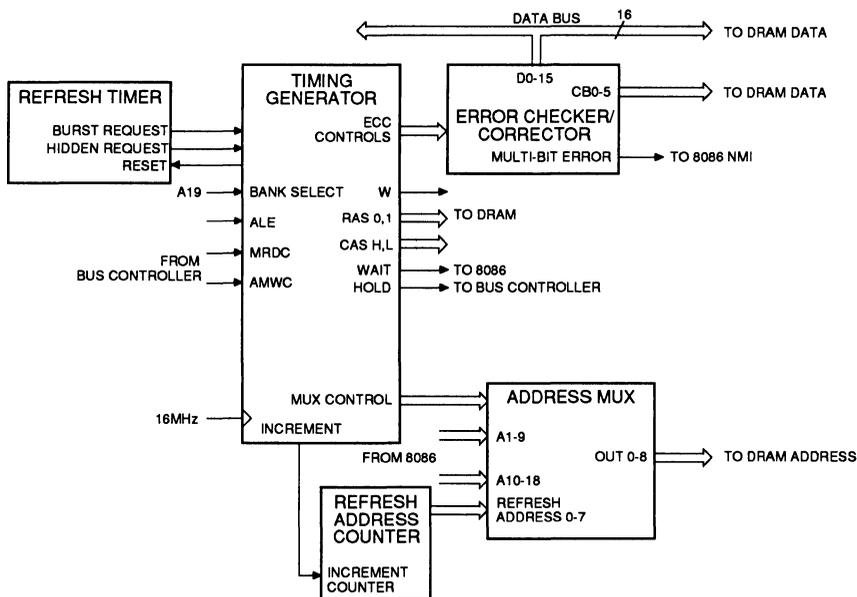
The *timing generator*, a state machine triggered by Address Latch Enable (ALE) at the beginning of the processor cycle, produces all the timing required to perform the memory accesses and refreshes. The signals generated

by this block include the row address and column address strobes (RAS and CAS), the WRITE signal, the WAIT state signal for the processor, the HOLD signal that isolates the processor from the memory, the clock for the *refresh address counter*, and the select control for the *address selector*.

The *refresh address counter* is an eight bit counter that provides the eight bit addresses necessary to refresh the DRAMs.

The *address selector* selects which address is sent to the DRAM. During a read or write cycle the *timing generator* select control signal tells the *address selector* to select the DRAM row address, strobe it with the RAS, and then select the column address and strobe it with the CAS. During a refresh, the *address selector* selects the address from the *refresh address selector* and strobes it into the DRAM with RAS.

During a write cycle, the *error checker/corrector (ECC)* generates six check bits using a modified Hamming code for each sixteen bit data word and writes them to memory along with the data. Use of a modified Hamming code permits single bit data correction and double bit error detection. During a read cycle, the ECC compares the check bits read back from memory with new check bits generated from the data read back. If the comparison yields a correctable error, the ECC will correct it. If the error is not correctable, it will flag the NMI on the processor.



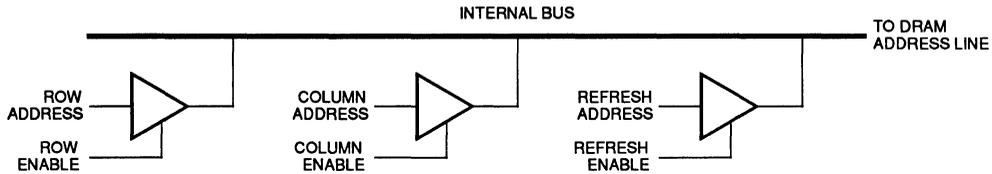
1127 02

**Figure 2. LCA Block Diagram**  
Block diagram of the DRAM controller functions implemented in the LCA.

Perhaps the most important feature of the LCA architecture for implementing a DRAM Controller is its internal three-state bus capability. The three-state buffer enables onto the horizontal longlines allow the designer to implement an internal bus in the LCA. This feature permits the implementation of the Address Selector without using any CLBs. Figure 6 shows a bit slice view. The row, column, and refresh addresses all have access onto the internal bus, and to the outputs that lead to the DRAMs. By controlling the three-state enables, only one address is allowed onto the bus at a time. This feature is essential to this design, and has many other applications including performing wired-AND functions and address decoding.

**CONCLUSION**

Although the bottom-up design of a DRAM controller is a complex task, it is necessary in cases in which off-the-shelf controllers do not meet the requirements of the system. SSI/MSI and custom gate array solutions involve trade-offs and compromises. Designing a DRAM controller and ECC with an LCA is a straightforward application and a good fit for the 3000 family architecture. The Programmable Gate Array offers the flexibility necessary to match the many different memory systems, the integration desirable for board level designs today, and the cost effectiveness required to make a competitive product.



1127 08

**Figure 6. Address Multiplexing Using Three-State Enables onto Internal Buses**

Logic analyzers and in-circuit emulators are similar types of electronic test equipment. Each involves the monitoring of certain digital signals within the system being tested. For general-purpose logic analyzers, these signals can be any nodes on the board being tested that the user selects. Another category of logic analyzers, sometimes called bus analyzers, are designed to plug into a specific microcomputer bus (such as a PC-bus or Multibus) and monitor and control the activity on that bus. In-circuit emulation of a specific component (usually a microprocessor) involves monitoring and controlling the signals input and output by the device being emulated.

A record of the activity on the nodes being tested by the analyzer is stored in a memory buffer called "trace memory"; this activity is called "acquisition mode." In some cases, the analyzer might also be controlling some of the target system's functions, such as single stepping a clock or interrupting a processor. After tracing is completed, the trace memory is then read and displayed by control logic in the analyzer; this is the "analysis mode." Trigger and breakpoint logic is integral to this operation; the user specifies when the tracing of signals in the target system should begin (triggering) and end (the breakpoint). Hence, a description of the combination of signals that make up the triggers and breakpoints must be stored in the analyzer before beginning operation, and the state of the target system must be continuously monitored to determine when a trig-

ger or breakpoint occurs. Triggers and breakpoints might be defined as a single event in the target system, or some ordered sequence of multiple events. Usually, trigger and breakpoint conditions are simply stored in registers and then continuously compared to the target system signals that are being monitored. Typically, logic analyzers and in-circuit emulators use many SSI/MSI latches and comparators to perform these functions.

The large number of logic functions and registers available in the Xilinx Programmable Gate Arrays make them ideal for implementing these trigger and breakpoint functions within a single device. Furthermore, the interface logic for controlling the trace memory can also be integrated into the PGA. This application can take advantage of the re-configurable nature of the PGA architecture. (Effectively, the Programmable Gate Array can perform different functions within the same system at different times by loading different configuration programs. This can lead to fewer packages on the printed circuit board and increased reliability.) One configuration of the PGA could be used for acquisition mode operation, capturing data, searching for the triggers and breakpoints, and filling the trace memory. When the breakpoint is reached, a different configuration could be loaded into the same PGA for the analysis mode; the PGA now controls the reading of the captured data from the trace memory. Additional functions, such as control of the user interface, also could be incorporated into the PGA.

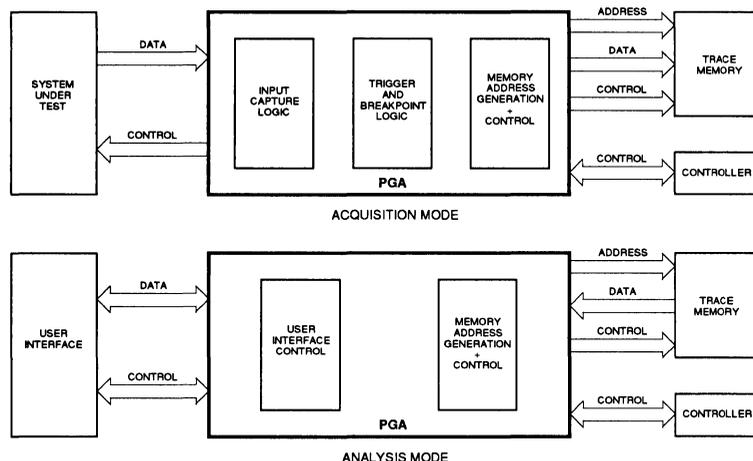


Figure 1. Logic Analyzer/In-Circuit Emulator



The Programmable Gate Array Company



**1 Programmable Gate Arrays**

**2 Product Specifications**

**3 Quality, Testing, Packaging**

**4 Technical Support**

**5 Development Systems**

**6 Applications**

**7 *Article Reprints***

**8 Index**

---



## Article Reprints

---

Building Tomorrow's Disk Controller Today .....	7-1
The Acid Test .....	7-5
Programmable Logic Better the Odds .....	7-8
Using LCAs in a Satellite Earth Station .....	7-12
Faster Turnaround for a T1 Interface .....	7-17
Two, Two, Two Chips in One .....	7-19
LCA Stars in Video .....	7-22

---

Jim Reynolds, President, Dave Randall, Chief Engineer, Andromeda Systems, Canoga Park, CA

### Reprogrammable logic with a flexible architecture enables a controller to keep up with today's high-capacity, high-speed disk drives

Computer manufacturers historically have relied on advances in CPU and semiconductor memory technology for increasing system throughput. At the same time, they accepted as inevitable the hardware-bound I/O bottleneck. This position is becoming untenable with recent advances in magnetic disk technologies, which have led to a proliferation of high-capacity, high-speed drives.

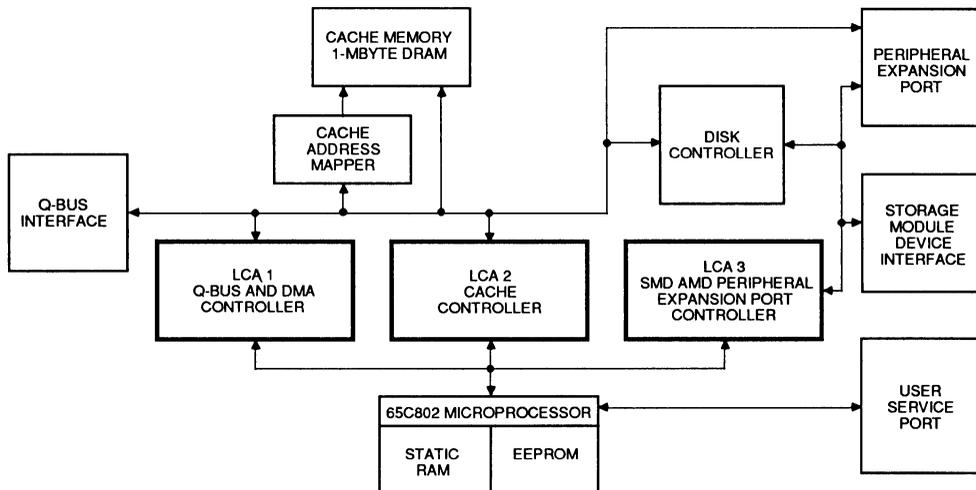
Full performance from these drives needs sophisticated controllers like Andromeda Systems' new Storage Module Device Controller (SMDC). With a 1-Mbyte data cache and dynamic read-ahead algorithms, the SMDC dramatically reduces average disk access time and significantly improves overall system performance (see box, "The Storage Module Device Controller"). The design and performance benefitted greatly from using Xilinx's Logic Cell Arrays (LCAs).

Very early in the design, it was clear that its high-performance caching scheme needed more SSI/MSI logic than

could be surface mounted onto a 35-in.<sup>2</sup> dual-width board. The only answer appeared to be VLSI custom or semicustom devices like gate arrays. But gate array definition requires absolute design accuracy, and so a prototype must be constructed long before custom-tooled ICs can be specified and manufactured. Paradoxically, the prototype itself required highly integrated logic.

To break that frustrating circle, it was necessary to convert directly from schematic capture to a silicon breadboard of multiple *electrically programmable logic devices* (EPLDs). Because many logic functions would be added to the prototype after the initial test, EPROM-based PALs were considered, like the EP1200 from Altera, which licenses the technology from Monolithic Memories.

The EP1200 could provide the minimum functionality on the silicon breadboard, but not the level of device integration for the production circuit board. To implement the various state machines and other logic of the design, each target gate array would need three EP1200s. The resulting schematic capture and simulation would then be used to fabricate the gate arrays for the final product.



1148 01

**Figure 1.** On Andromeda Systems' new Storage Module Device Controller, Xilinx Logic Cell Arrays handle the Q-bus interface and direct memory access (DMA) control, RAM/data-cache control, and SMD and peripheral expansion port control.

Fortunately, this circuitous design path was bypassed by using Xilinx's LCA (see box, "Xilinx's programmable gate array"). There are two basic differences between LCAs and other EPLDs. First, the LCA has the flexible architecture of a gate array. Second, LCAs employ static memory to hold the logic configuration data.

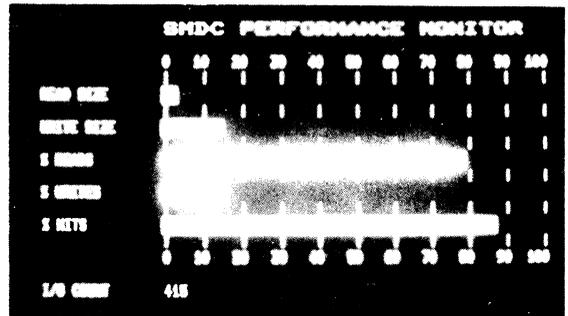
The LCAs brought several significant advantages to the controller design. Since the Xilinx 2064 LCA has 64 configurable logic blocks and the EP1200 only 20, a single LCA could replace the three target gate arrays, eliminating the fabrication delays and costs of custom tooling.

Furthermore, the position of the LCAs on the board could be determined before their internal logic configuration was designed. Other than dedication input and output pins, only a general idea of the function of each LCA was needed. The board layout and the internal LCA logic design could proceed in parallel, greatly reducing development time. Most design changes could be implemented merely by reprogramming the LCAs. Thus, use of the LCAs allowed the design to go directly from schematic capture to a production board, skipping the wire-wrapped prototype.

The first LCA on the SMDC is the Q-bus interface and direct memory access (DMA) controller (see Fig. 1). All but 5 of the 64 internal logic blocks were used. The LCA holds the DMA addressing logic, the bus registers, and the interrupt logic.

RAM/data-cache control is the job of the second LCA. It controls the cache and has the interface between the disk controller IC and the DMA logic. It signals cache-write enables, multiplexes memory addresses, and enable DMA reads and writes.

The third LCA controls the SMD port and peripheral expansion port. The expansion port is just a group of programmable I/O connections. Since the LCA is programmable, the control logic for the expansion port can be reconfigured for any desired I/O interface. Thus, this port provides for future expansions (like adding a tape drive, optical disk, or extra cache memory) at a fraction of the cost of a separate controller. Unused logic in this LCA will permit on-board functions to be added in future microcode revisions to the controller.



**Figure 2.** The user service port can create color bar graphs that dynamically show various attributes of the data cache, such as read times, forward block reads, and I/O completion rates.

Aside from the LSI circuitry, the only other logic on the SMDC board are TTL bus transceivers, SMD interface drivers, and a few PALs.

The RAM of the data cache is in ZIPs. Most of the interface logic was surface mounted to the board. Despite the board's small size, these VLSI devices permit several advanced features.

The SMDC's user service port connects directly to terminals or modems. No special test programs for specific system environments are needed to communicate with the controller. Users can define drives, assign logical units, format drives, and do other more esoteric functions.

This port can monitor the operation of the controller while the drive is in operation. The user can display color bar graphs that dynamically show various attributes of the data cache, such as read times, forward block reads, and I/O completion rates. Caching parameters can be adjusted, letting the user tune the system for optimum performance.

Firmware can alter the configuration data for the LCAs, modifying the circuit schematic and not the board. Since the firmware is in EEPROMs, the service port can accept microcode upgrades in the field via modem. PROM set replacement and on-shelf obsolescence are avoided.

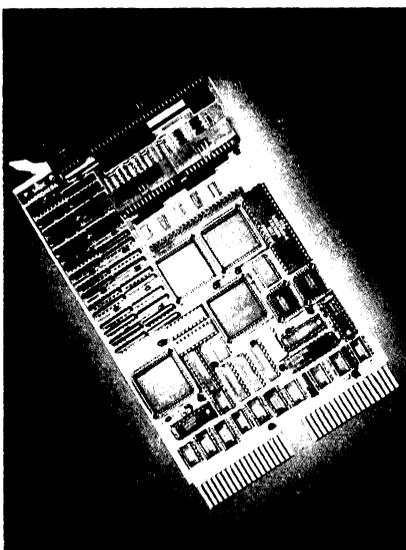
## THE STORAGE MODULE DEVICE CONTROLLER

Designed for LSI-11 and Micro/VAX II systems, Andromeda Systems' Storage Module Device Controller (SMDC) for Winchester drives supports two SMD or SMDE drives at data rates up to 25 Mbits/s. Another Andromeda controller, the ESDC, works with the Enhanced Small Device Interface, the ESDI, for Winchester or floppy-disk drives. Both controllers use the standard DU device driver and work with such operation systems as RT-11, TSX+, RSX, RSX-11M, MicroRSX, RSTS, MicroRSTS, Ultrix, DSM, Unix, and MicroVMS.

The SMDC achieves more performance and flexibility than did previous generations of disk controllers. It includes data caching, high data-transfer rates, a peripheral expansion port, field-loadable microcode, and a user service port. State-of-the-art VLSI components and packaging techniques fit the entire controller within the 35 sq in. of a dual-width Q-bus board (see figure).

Using Digital Equipment's Mass Storage Control Protocol (MSCP), the SMDC can partition two drives into as many as 16 logical units with up to 32 Gbytes each. On-board intelligence comes from a 65C802 microprocessor, and all the processor's code resides in just two EEPROMs. The majority of the remaining logic is implemented with Xilinx programmable Logic Cell Arrays (LCAs). Data integrity is ensured by 48-bit error detection and correction logic. An expansion port can be connected to accessory modules, allowing control of devices like tape drives, optical disks, or extra cache memory.

The performance of the SMDC is greatly enhanced with a 1-Mbyte data cache and unique caching algorithms.



Andromeda divides the cache into 1,024 granules. The information kept for each 1-Kbyte granule depends on select criteria, which include:

- The time data is first accessed
- The number of times data is read
- The time of the most recent read
- The size of the read.

This information is then entered into an equation that approximates how probable it is that the granule will be requested again soon. Those granules with low probabilities are designated to be overwritten by the next disk-read operation. During cache accesses, a memory mapper translates logical memory addresses into the physical addresses of the appropriate granule in much the same way that the Micro-Vax II memory management unit would.

## PREDICTIVE CACHING

In a novel departure from most caching schemes, the SMDC caching mechanism not only looks at the past, but tries to gaze into the future as well. As the system requests the data that has been pre-fetched into the cache, the controller retrieves not only the requested data, but also preemptively reads extra sequential blocks when specific probability conditions are met. As a result, the on-board cache's typical hit rate is over 80%. In other words, the data being sought by the application will be ready and waiting in the cache over 80% of the time.

Approximately 90% of the disk access time is due more to average seek times and rotational latency than to the actual data transfer rate. However, when a cache hit occurs, the access time depends only on the speed of the DMA channel responsible for sending the data to the Q-bus.

That DMA channel operates as fast as Q-bus specifications allow—to be specific, at a rate of up to 4 Mbytes/s. Consequently, with the SMDC cache, seek time and rotational latency are reduced to zero over 80% of the time. This reduces the average time for a four-block read from 27 ms to less than 6 ms.

In the majority of computer systems, mass-storage access time is undoubtedly the largest component of throughput. In this situation, use of the SMDC enormously improves total system performance.

\$\$\$\$\$

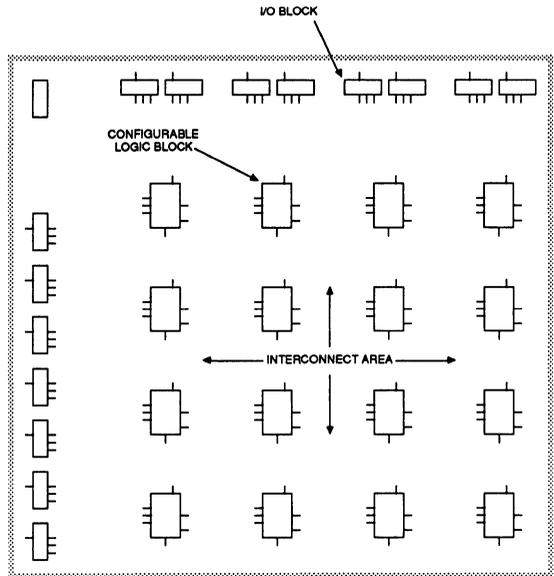
Andromeda Systems' Storage Module Device Controller is available now for \$2,195. (The company's ESDI controller is available for \$1,995.) For more information, call Don Talmadge at 818-709-7600, or circle 336 for the SMDC and 337 for the ESDC.

### XILINX'S PROGRAMMABLE GATE ARRAY

The Xilinx programmable gate array, known as a Logic Cell Array (LCA), is a high-density CMOS IC that combines user programmability with the flexibility of a gate array architecture and the economy and testability of standard products. Elements of the array include three categories of configurable elements: I/O blocks, configurable logic blocks, and programmable interconnections (see figure).

I/O blocks provide an interface between the external package pin and the internal logic. Each block includes a programmable input path and output buffer. The array of configurable logic blocks contains the functional elements from which the user's logic is constructed. Each array includes a combinatorial section, storage elements, and internal routing and control logic. Programmable interconnection resources connect the inputs and outputs of the I/O blocks and configurable logic blocks into the desired networks.

An LCA is configured by programming static memory cells that determine the logic functions and interconnections. On-chip logic provides for automatic loading of the configuration program at power-up or upon command. A personal computer-based development software package generates the configuration program. Other tools include a simulator, in-circuit, and schematic capture package.



1148 02

Reprinted with permission of *Electronic Products*.

By Steven K. Knapp, Field Applications Engineer, Xilinx Inc.

---

**Using an EPLD vendor's application, the same circuit was built with a Logic Cell Array. The comparison provides real insight to the differences between these two technologies.**

As with a processor, a PLD's architecture determines its functional logic density and its performance.

Among processors, general-purpose microprocessors and application-oriented devices like digital signal processors share many similar internal logic structures and functions. DSPs have a more rigid architecture, designed for particular applications, while general-purpose microprocessors address a much wider set of applications. The same situation exists in the world of PLDs.

Until recently, the sum-of-products (SOP, also known as AND-OR) structure was the only one available for PLDs. Its simple, fixed architecture fits a variety of low-density, high-speed applications. These include fast, wide logic functions found in decoders, multiplexers and counters.

Some recently introduced high-density PLDs are based on extensions of the conventional AND-OR architecture. Examples include the MMI 64R32 MegaPAL and the Altera EP1800 Erasable Programmable Logic Device (EPLD).

Like a DSP processor, the sum-of-products architecture of these devices efficiently meets certain needs. However, this architecture is not suitable for higher-density, register-intensive random logic structures commonly found in logic designs.

### LOGIC CELL ARRAY

The Programmable Gate Array differs vastly from the conventional sum-of-products architecture. Its flexible, register- and I/O-rich, array-style architecture addresses a wider set of common logic design problems.

Its Logic Cell Array architecture consists of three basic programmable elements: input/output blocks, configurable logic blocks and programmable interconnects.

Each I/O block can be individually configured as a direct or registered input, a direct or three-state output or as a bidirectional I/O.

The configurable logic blocks contain combinatorial logic plus storage elements. The combinatorial logic within each block implements any possible single function of four variables, or any two functions of up to three variables. The storage element is configurable as an edge-triggered flip-flop, or as a level-sensitive latch, both with asynchronous SET and RESET inputs. The programmable interconnect allows each of the storage elements to be clocked either synchronously or asynchronously.

Three levels of programmable interconnect resources provide the interconnection between blocks:

- Direct interconnect allows fast connections between adjacent blocks.
- General-purpose signals travel through an array of switching matrices that yield an efficient means of connecting scattered random logic.
- Long metal lines that traverse the chip distribute clock or other signals with high fan-out or requirements for minimum skew.

The best way to illustrate the difference between the Logic Cell Array and the more conventional AND-OR architectures is through a design example.

An X-Y position controller design was originally developed by Altera as an application example for its EP1800. The design is fairly simple, and illustrates the capability of programmable logic to address the problems faced by logic designers. The example can be used to demonstrate the capabilities of both the Altera EPLD, and the Xilinx Logic Cell Array.

X-Y position controllers are employed in a variety of design applications to control motors for printers, plotters, robotics, and numerical controllers. The controller compares the desired location loaded from an external processor with the present motor position stored within the PLD. Based on the results of the comparison, the controller drives two four-phase stepper motors (an X- and a Y-position motor) to its desired position.

In this design, X- and Y-position data is loaded into the device from an external microprocessor. The 1,800-gate EPLD has no input storage elements, so the data must be held valid by some external device until both motors reach

their final position. Since the Logic Cell Array has input flip-flops which hold the X- and the Y-position data, it offers a simpler interface to the processor.

The desired position data is compared against the present position data. The result of the comparison drives the 7-bit up/down counters which, in turn, drive the state-machine motor control. The stepper motors used in the application have 7.5 degree fullstep increments with optional 3.75 degree half-step increments available. A 7-bit binary up-down counter is required to keep track of the 96 motor steps required to rotate each motor a full 360 degrees.

In the EPLD designs, seven macrocells are used to implement the binary up/down counter for each half of the position controller. Because of the low-frequency clock (500 kHz), this same counter requires only seven Logic Cell Array logic blocks. If this were a high-speed counter, a few additional blocks would perform some level of carry-look-ahead.

If the desired position is greater than the present position, the state-machine-based motor control drives the stepper motor clockwise. If the position is less, the controller drives the motor counter-clockwise.

Four EPLD macrocells are required to implement the state-machine for each half of the position controller, while seven Logic Cell Array logic blocks are required to implement the same function.

To signal the processor that the entire operation is complete, an open drain interrupt signal is generated when both the X- and the Y-position motor have reached their

final location. This also signals the processor that further motor action may be taken. A master RESET signal resets the present position to zero.

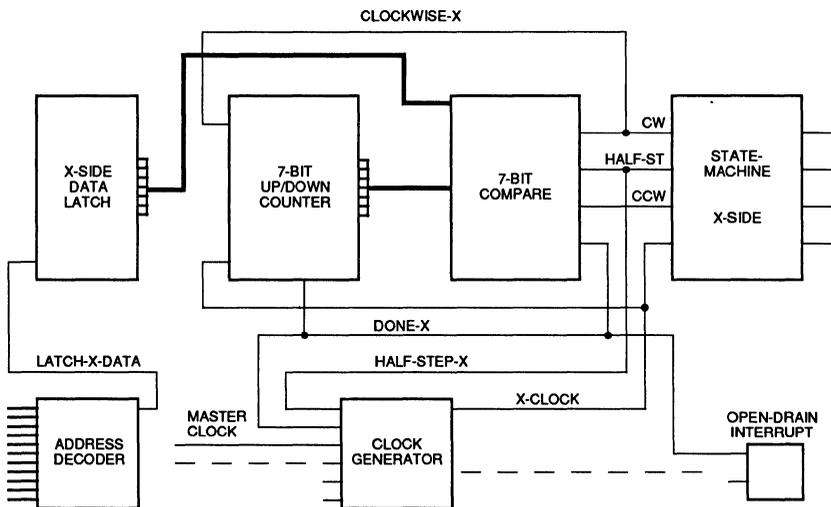
**RESOURCE REQUIREMENTS**

Based on the requirements of the design, the X-Y position controller example requires 47 of the 48 macrocells in the 1,800-gate EPLD and all of its 64 I/O pins. A number of the I/O pins are used simply because a register is hard-connected to the pin. The two 7-bit counter, for example, use 14 I/O pins because they cannot be buried. The X-Y controller design uses 98 percent of the macrocells and 100 percent of its I/O pins.

The same design (plus the input data registers) fits in 49 configurable logic blocks and 27 input/output blocks in a Logic Cell Array (the MASTER RESET input uses the Logic Cell Array's dedicated master reset pin). Fewer I/O pins are required, since the 7-bit up/down counters and the state-machines were buried in the Logic Cell Array design.

The X-Y controller design would occupy 77 percent of the 1,200-gate XC2064's logic blocks and 47 percent of its I/O pins. Or if the 1,800-gate Logic Cell Array were used, the X-Y controller would occupy just 49 percent of the logic blocks and 36 percent of its I/O. The remaining logic and I/O blocks can be used to build chip select logic to make a clean interface to the microprocessor.

The architectural differences between the two technologies allow a design which requires an entire 1,800-gate EPLD to fit in just a portion of a 1,200-gate Logic Cell Array. The EPLD can perform any one of the required tasks quite



1148 03

**The schematic above illustrates the X-axis half of the position controller used as the design example. The other half is identical to this circuit, and drives the Y axis instead of the X.**

well. The sum-of-products architecture allows EPLDs to implement fast counter, decoders, and multiplexers in a single pass. Combining these elements into a larger, more complex system with additional passes through the array however, hurts both performance and density.

The flexible Logic Cell Array architecture allows entire complex systems to be implemented efficiently and quickly. Given the same 1,800 gates of logic, and 1,800-gate Logic Cell Array could implement an X-Y-Z position controller in the same density used to implement an X-Y controller in an EPLD.

## CONCLUSION

The benefits of programmable logic devices within any design are generally undeniable. Both devices shown in the design example (the EP1800 EPLD, and the XC2064 Logic Cell Array) replace many SSI/MSI components.

However, not all PLDs are created equal. A designer should contemplate his system needs and goals when deciding which PLD to use. Like making the correct choice of a system processor, the correct PLD choice will have an impact on the final system performance.

Some guidelines and cautionary notes are necessary for designers new to the PLD approach. These are:

- Be wary of PLD gates counts. As mentioned before, not all PLDs are created equal. Equivalent gate counting is used by many manufacturers to compare their devices against a competitor's. Two devices with approximately

equal gate counts will differ on functional density within a system because of differences in architecture. A better way to determine density is to count the types and amounts of various resources on the chip, and compare those with the needs of your application.

- Understand which PLD architecture best suits your application. The sum-of-products (AND-OR) architecture is well suited to applications that require ANDing of a large number of inputs, like those found in address decoders and some counters. The interconnect structure of an AND-OR PLD makes single-pass logic functions quick and efficient. However, the complex random logic requirements of higher-density logic design stretches the limits of conventional AND-OR devices because of the feedback requirements.

- Determine your I/O and register requirements. In most AND-OR PLDs, outputs are hard-connectors to the registers within the device. Therefore, each register either uses or wastes an I/O pin and vice versa. In the Logic Cell Array, I/O and registers are separated by programmable interconnects. Therefore, entire register resources like counters and shift registers can be buried without using or wasting I/O pins.

Major enhancements in high-density PLDs will stem from architectural innovations rather than process-related developments. Regardless of which production process is used, all PLD manufacturers are searching for the optimal mix of high performance, high density, and production cost to best meet designer's logic needs.

Reprinted with permission from *Electronic Engineering Times*.

	MegaPAL	MegaPAL	EPLD	EPLD	LCA	LCA
Claimed Gate Density	1,500	5,000	1,200	1,800	1,200	1,800
Architecture	AND-OR	AND-OR	AND-OR	AND-OR	Array	Array
Developed By	MMI	MMI	Altera	Altera	Xilinx	Xilinx
Inputs (max.)	32	64	36	64	58	74
Outputs (max.)	16	32	24	48	58	74
Storage Elements	16	32	28	48	64	100
Elements	0	0	12	0	58	74
Speed For 16-Input AND (ns, On-through-Off, fastest)	40	50	50	50	45	45
Register-To-Register Clock Frequency (Max. MHz)	16	16	20	23.2	58.8	58.8
Quiescent Power (W)	1.4	3.25	0.015	0.001	0.025	0.025
Packages	40 DIP 44 PLCC	84 PLCC 88 PGA	40 DIP 44 CJCC 44 PLCC	68 CJCC 68 PLCC 68 PGA	48 DIP 68 PLCC 68 PGA	68 PLCC 84 PLCC
Process	Bipolar	Bipolar	CMOS	CMOS	CMOS	CMOS
Technology	Fuse	Fuse	EPROM	EPROM	SRAM	SRAM

1148 04

Shown above is data on several types of high-density EPLDs. Though the MegaPAL devices offer the most gates, they are bipolar, fuse-based design, and so are not reprogrammable.

by Cliff Dutton, GTECH Corp., Providence, RI

In countries throughout the world, the vitality of the on-line lottery industry is enhanced by seasonal and special promotional games. But new games require new bet-slips, and bet-slip readers must be able to accommodate frequent changes in format. To accomplish this, programmable gate arrays are replacing older, less flexible architectures.

In the development of GTECH's Solid State Reader, many existing technologies were evaluated, but they imposed unacceptable limitations on bet-slip processing, restricting bet-slip formats to rows and columns. Moreover, the process of reading the coupons was dependent on complex moving parts, and the reading elements were exposed to the external environment.

To maximize flexibility and minimize board space, Xilinx's (San Jose, CA) Logic Cell Array (LCA) was chosen for the Solid State Reader. The LCA, touted by the company as a "programmable gate array," represents a novel programmable logic device that is notable for its reprogrammable architecture. This architecture provides flexibility throughout the product's life span, which allows on-line bet-slips to be produced with marks in any arrangement. Each bet-slip reader at every terminal can be configured on-line to read any bet-slip from an active suite of eight different bet-slips.

Figure 1 shows three lottery bet-slips. Some of the pertinent features of the European Lotto game slip (a) include strobe marks along the top edge, the OCRB-3 characters (bottom center), and the name and address field (bottom right). In the sample bet-slip from a lottery in the U.S. (b), there are no OCR characters or name and address information. However, there is an area from which handwritten information must be extracted. Apart from the different features, the aspect ratios of bet-slips are not standard. Modern bet-slip processing systems must be able to read all of the different formats in many aspect ratios. A format that forgoes the usual row and column arrangement (c) is also depicted.

As there are no standard architectures or interfaces for image sensors, GTECH evaluated many image sensor approaches. However, direct comparison of sensor performance could not be made in the application environment. For example, comparisons of sensor sensitivity at the pixel level were impossible due to the differences in sensor-interface electronics. If degraded sensitivities were evident, they could derive from either the sensor or

the sensor interface. Similar difficulties hindered direct comparison of achieved resolution. To accurately evaluate these parameters, each sensor had to be designed into prototype readers. This involved driver and frame acquisition clock signal generation.

Because lotteries have no standard bet-slip size, as many "standards" as possible need to be accommodated. Thus, it was necessary to maintain flexibility in the format of the target image.

## PROTOTYPING A SYSTEM

The implementation of a prototype system had one goal: to prove the feasibility of recognizing handmade marks in an imaging system. Because the volume of readers is potentially high, component costs were a serious issue.

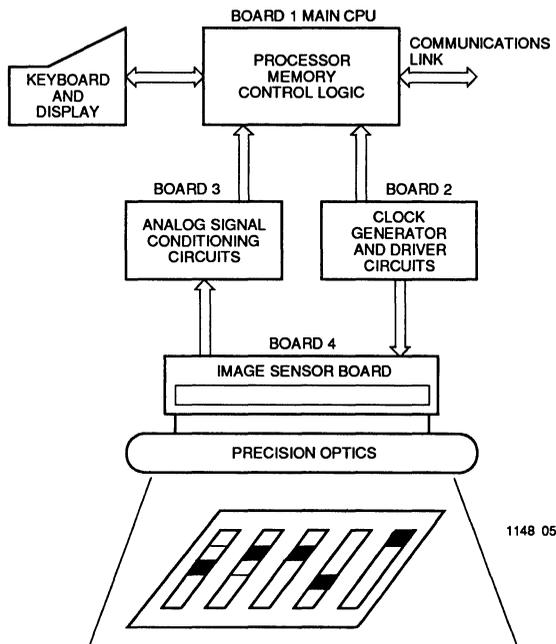
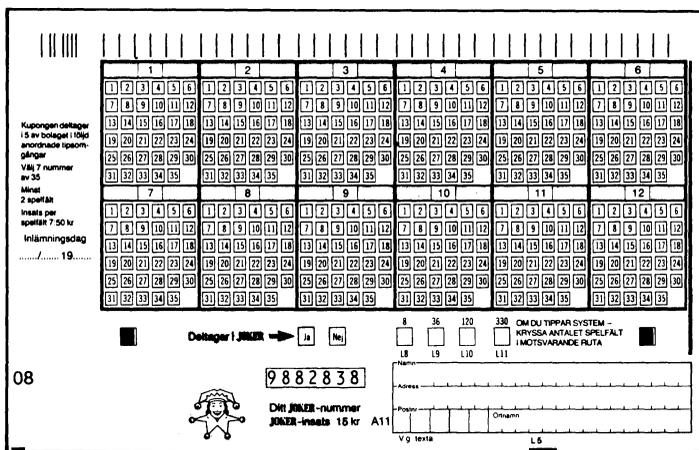


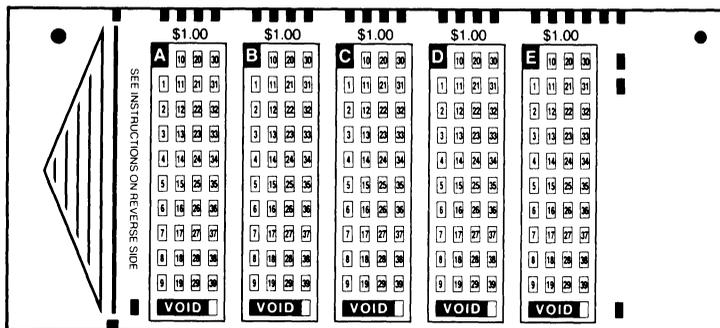
Figure 2. The goal of developing a prototype bet-slip processor (shown above) was to prove that handmade marks could be recognized in an imaging system. Four boards were initially developed for this modular design: CPU/memory, clock-driver, analog amplifier, and sensor mounting.

First, a working model was developed. To balance development costs, a set of printed circuit boards based on TTL logic devices was manufactured. Partitioned functionally, the board set supported modular design changes. Four pc boards were initially developed: a CPU/memory board, a clock-driver board, an analog amplifier board, and a sensor mounting board (Figure 2).

In the initial design, flexibility did not exist. Even though modularity protected the design from becoming obsolete, significant design alterations were required to accommodate different sensors. Because sensor clock signals are multiphase, new clock generators would be needed for new sensors. Also, bugs were difficult to find, and circuit board modifications were required to eradicate such bugs.



(a)



(b)



(c)

**Figure 1.** Betting slips for lotteries come in varied shapes and sizes. (a) Shown here are lotto slips from Europe and (b and c) the United States. Such variety in slip design must be accommodated in the development of bet-slip readers.

Finally, the target image aspect ratio was fixed because the clock generation circuits were implemented in hardware.

Aspect ratios of target images are important because only necessary information on the image needs to be processed. If the target image is 2:1 and the imaging format is 1:1, for example, then half the image is useless. A better solution would mirror the aspect ratio of the target image in the image format.

To overcome the limitations of hardwired logic and reduce board space, several technologies were evaluated. These included programmable logic arrays (PLAs), field programmable logic devices (FPLDs), semicustom and fullcustom devices, and Xilinx's Logic Cell Array (LCA).

Size constraints indicated the necessity for semicustom or full-custom integration, but traditional LSI technologies violated the flexibility constraint. Although full-custom was attractive, design costs were prohibitive and did not permit iterative development. Standard PLDs did not allow for the variety of register-like functions that the clock generation logic required.

Programmable logic arrays were attractive for some logic functions and would have been the least costly. However, PLAs did not allow the multiple register implementation necessary for clock generation. Thus, the counting algorithms would have remained external to any integration of the combinatorial logic. Also, although the PLA architecture would have saved board space, it would not have preserved the functional modularity achieved in the first implementation. Thus, it would have been impossible to evolve a PLA-based system in response to changes in sensor technology. Finally, any required changes would have to be performed by field replacement. With over 35,000 lottery terminals installed on five continents, this was unacceptable.

Field programmable logic devices, an update of the PLA-style architecture allowing limited reprogrammability, appeared to provide some of the flexibility needed. If the problem were merely a straight combinatorial one, FPLDs could have been used. However, the difficulty in supporting both registers and counting algorithms ruled out their use.

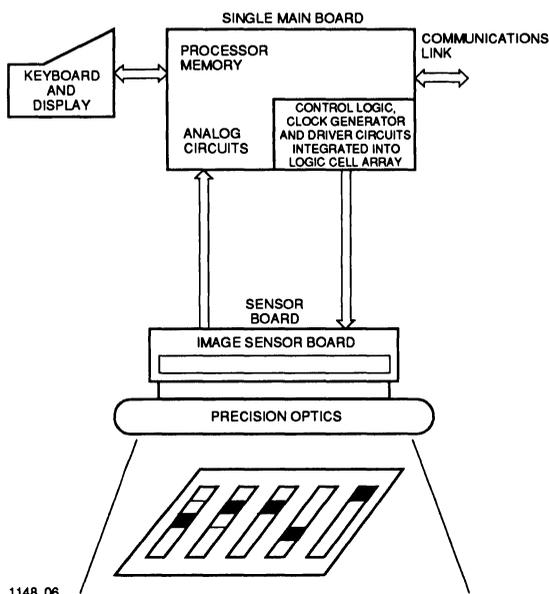
Semicustom and full-custom technologies would have solved all the functional problems, but they lack flexibility. Because the development of the reader was ongoing, the commitment to custom implementations was out of the question. In addition, nonrecurring engineering (NRE) costs were prohibitive and the devices could not be adapted to changing sensor technologies or changing bet-slip reading requirements.

Xilinx's LCAs permit a two-board set to be designed without sacrificing functional modularity. In addition, counting algorithms can be implemented in the LCAs. Finally, LCAs allow for a multiple-iteration development cycle.

### PUTTING A BUG TO REST

Initially, the TTL-based system was implemented in four pc boards. However, it contained a bug. For every horizontal line, an extra pixel pulse was being supplied. Although this was confusing to the eye, it was compensated for in firmware. Because the redesign of the clock driver board was a significant task, the bug was allowed to live through many iterations of the development cycle. When the design of the clock generation circuit was translated into the LCA, it was a trivial matter to delete a single horizontal clock pulse and put the bug to rest in an afternoon.

Using the LCA also provided the ability to vary the clock generation circuitry to evaluate different sensors. Because there is no standard architecture for solid-state digital imaging devices, clock requirements vary for different sensors. In a standard imaging application, it might be possible to source the appropriate support chips for each sensor from the manufacturer. But because development of the reader involved nonstandard video speeds in a noninterlaced mode, it was impossible to use standard support chips. If it had been necessary to develop a clock driver pc board for every sensor evaluated, it would have been impossible to evaluate more than one sensor in the development time. Because LCAs were used, varying multiphase clocks could be generated for different sensors under evaluation. Thus, the turnaround time for a design change in the clock generation circuits was reduced from one to six weeks to one day.



1148 06

**Figure 3.** GTECH's Solid State Reader uses Logic Cell Arrays (LCAs) to maximize flexibility and minimize board space. Frame-grabber, memory addressing, and sensor clock driver functions are consolidated in the LCA. By reducing the number of chips, the required number of boards shrinks from four to two.

The Solid State Reader does not rely on standard video output. Thus, the 4:3 standard aspect ratio for broadcast television is not a requirement. All image processing is internal to the system. Real-time display of the image is never required. Therefore, only those areas of the sensor that may contain relevant information need to be required. Information-bearing areas of a bet-slip vary with the bet-slip design, so it is helpful to redefine the area of the sensor that is acquired for processing.

Because the clock driver circuitry, the memory addressing logic, and the frame-grabber logic are all implemented in the reconfigurable LCA, it is possible to acquire only certain areas of the image. As each sensor has different horizontal and vertical clock pulses, this flexibility cannot be achieved in hardwired logic.

**Figure 3** illustrates the current architecture of the Solid State Reader. Because of the functions consolidated in the LCA, the system was reduced from four pc boards to two. This could have been done using other technologies, but they would not have preserved the functional modularity of the system. The LCA-based design provides both size reduction and functional modularity.

Reprinted with permission from *ESD: The Electronic System Design Magazine*.

Dave Farrow, M/A-Com Telecommunications, Germantown, MD

Conventional programmable logic devices (PLDs) include several interesting variations of latch-based AND-OR plane architectures in various technologies, all of which are useful for low-gate-density applications. Typically, a PLD can replace five to ten SSI/MSI parts.

A newer digital logic technology with an array architecture and flexible interconnection offers the programming flexibility of PLDs plus the gate density of low-end gate arrays. Architecturally, these devices have some similarities to gate arrays: they contain an internal matrix of logic blocks and a ring of configurable I/O interface blocks. Unlike conventional gate arrays, each part is a standard off-the-shelf unit that can be programmed by the user. The configuration program is automatically loaded into an on-chip static memory at power-up from either an on-board EPROM or an external source such as a floppy disk.

### THE EARTHSTATION SYSTEM

M/A-Com recently employed one of these "programmable gate arrays" in the design of a satellite earthstation, intended to network commercial facsimile operations. The network handles traffic at 56 kb/s, multiplexed into 26 channels and convolutionally encoded, yielding an overall

3 Mb/s transmission rate. The earthstation product, called an OPT (for On-Premises Terminal) is a "small-aperture" satellite earthstation, permitting efficient employment in a large number of remote locations, as illustrated in Figure 1.

Two main components comprise the OPT: an indoor unit and an outdoor unit. The outdoor unit includes the antenna and associated radio-frequency equipment.

At the outset of the design process, the indoor unit was intended to be contained in a small chassis that could support three standard-size boards. The boards originally planned for the system included one board each for controlling data traffic, transmit functions, receive functions, and demodulation. However, the chassis provided space for only three boards.

Project goals included the use of an existing proprietary custom chip design from a previous application. M/A-Com also investigated whether the design could be fit on only two boards, by using a gate array. Board design itself was driven by three primary factors: resource availability, cost, and schedule. Since reducing the number of required boards would reduce design time and keep product costs lower, M/A-Com decided to go with the gate array.

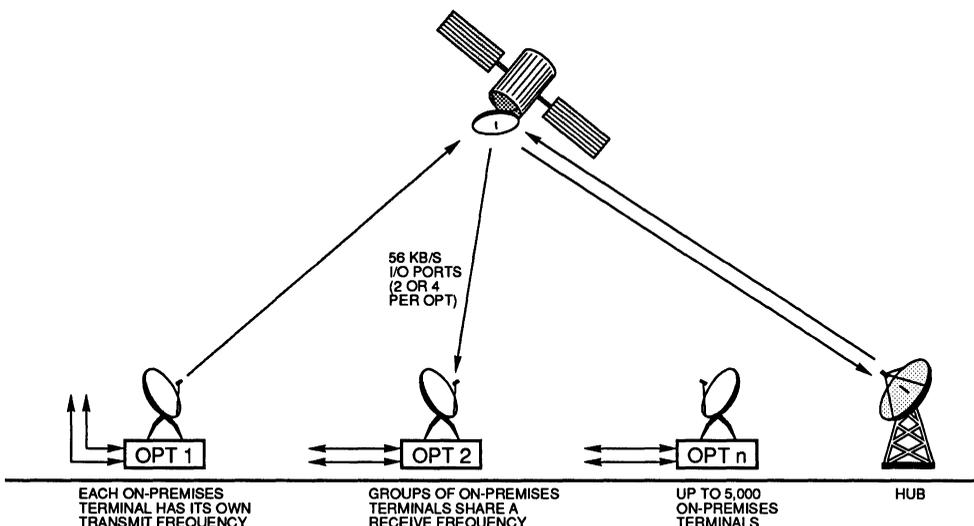


Figure 1. Satellite System

QTY.	DESCRIPTION	ITEM
3	8-BIT SHIFT REGISTER	74HCT164
6	4-BIT COUNTERS	74HCT163
4	DUAL D FLIP-FLOP	74HCT74
2	QUAD 2:1 MULTIPLEXER	74HCT157
1	QUAD XOR	74HCT86
1	HEX INVERTER	74HCT04
1	QUAD NOR	74HCT02
2	QUAD OR	74HCT32
3	QUAD AND	74HCT08
1	OCTAL LATCH	74HCT374
1	OCTAL BUFFER	74HCT244
25 ICs		

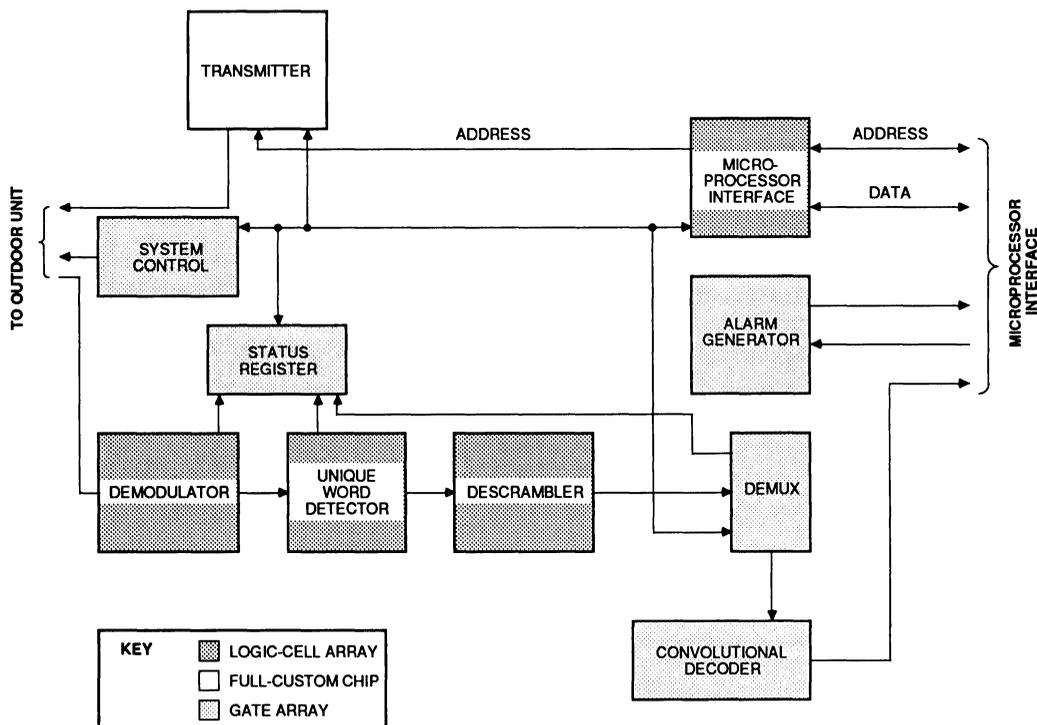
**Table 1. Standard Off-the-Shelf Equivalents to the Logic Contained in the LCA.**

The completed design employs a full custom IC, a gate array, and programmable logic, and subsists on only two boards. On one board, an Intel processor acts as a traffic-

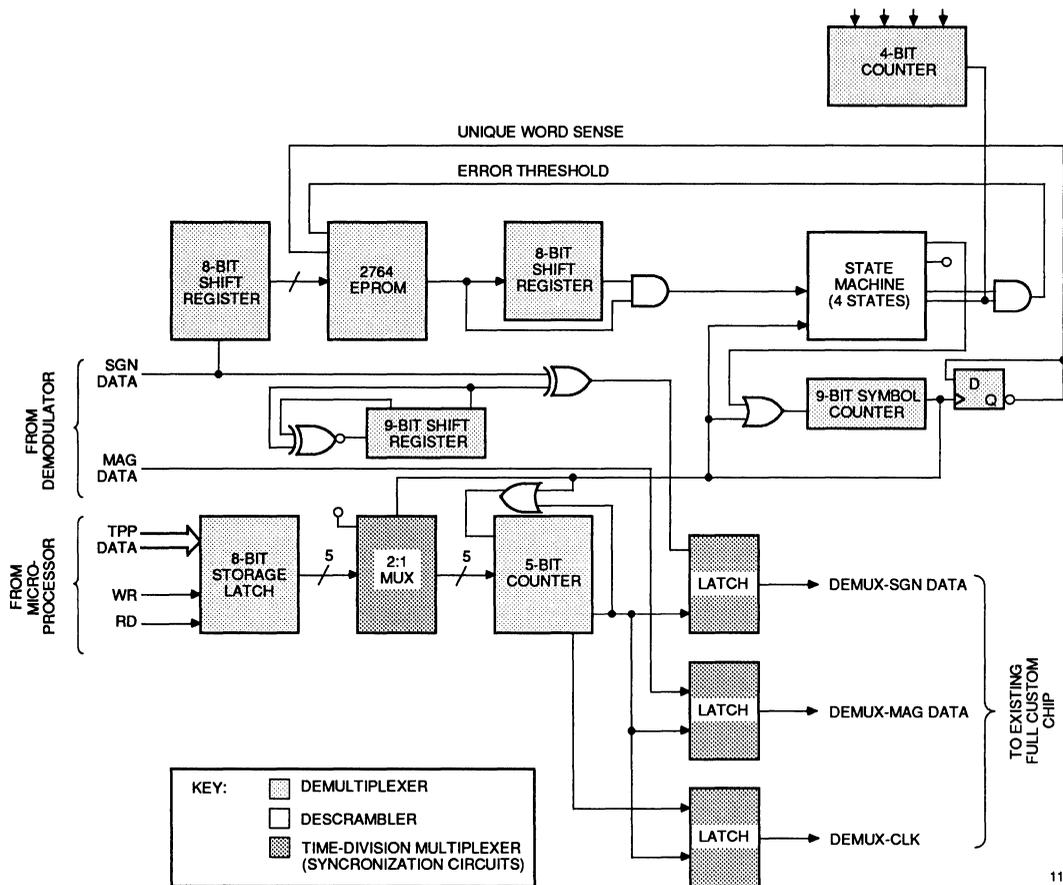
port controller and handles base-band X.25 data. Due to the use of semicustom and programmable technology, the remaining three functions were all merged onto the other board, which we call a "satellite channel interface" (see Figure 2).

We used a gate array for the transmit function, which otherwise would have required about 70 chips. For the receive function, we originally planned to use an existing full-custom ASIC (previously designed by M/A-Com) for forward error correction, and an additional 25 SSI/MSI parts for the receive logic. However, due to chassis constraints, the high density of components would have necessitated a multi-layer board for the initial design. Furthermore, based on previous experience, the likelihood of changes in the design specification was too high to risk a custom or semicustom solution for the initial design. Therefore, we originally planned to produce the high-density boards in quantity and to reduce the cost of the system at a later date, by first transferring the receive logic into a gate array and then replacing the expensive high-density four-layer board with a two-layer board.

While the design criteria were being prescribed and board-level functionality was being determined, we also investigated the newer programmable gate-array technology. The programmable part, the Xilinx Logic Cell Array (LCA),



**Figure 2. Block Diagram of Satellite Channel Interface.**



1148 10

Figure 3. A Schematic of the Digital Systems Incorporated into the LCA.

is architecturally similar to a gate array and is supported by a PC/AT-based workstation.

We determined that the internal organization of the LCA fitted the design requirements of the receive function. Specifically, the LCA provides many more flip-flops than other programmable logic devices, so that one chip contained enough functionality for our needs. Further, the LCA provided the required density savings, and its reprogrammability obviated the risks associated with late engineering changes. When engineering management was presented with the design alternatives, we decided to prototype a reduced portion of the receive circuit and thus evaluate the reconfigurable chip.

To implement the design, M/A-Com acquired the Xilinx XACT PC-based LCA development system. The system includes a macro library, with some of the required logic already defined. After several days of experimenting with the design tools, it took us one day to enter and only two hours to debug the design. We use Xilinx's XACTOR

in-circuit emulator for debugging.

Our original schematic was based on conventional LS and HCT parts; it included JK flip-flops and large counters (implemented by cascading common 4-bit counters), rather than gate-level elements. Since that method of design was inefficient for the LCA, we redesigned the receive circuit at the gate level and then implemented it in software via the cell array editor.

Using an LCA reduced the amount of hardware overhead normally associated with LKS and HCT technology. It was not necessary to waste control inputs, to cascade counters, or to determine what to do with unused bits of multiplexers. In our design, 25 SSI/MSI gate-equivalents did not even use up all the resources available in one LCA. Table 1 indicates the parts that we actually employed in the present design. Putting these functions in the LCA resulted in an 88% utilization of the internal cells, and a 60% utilization of the I/O cells. Thus it still remains feasible to add further functionality to the system, with no PCB

changes. We plan to do so in the future. Figure 3 is a schematic of the circuit placed in the LCA. Since the design is not I/O limited, there was no necessity to multiplex any of the input or output lines; but additional logic could have been added, should I/O multiplexing been needed. Note also that the descrambling circuit can easily be reconfigured, or made more complex. Changing the descrambler can be achieved merely by reprogramming the LCA.

One criticism leveled against the LCA is that it requires 12K bits of storage space to program the part during power-up. However, in our design, a 27C64 EPROM (used for a look-up table) was already on the board. A portion of this EPROM was available to store the LCA configuration program at no additional cost. Since the 12K bits of storage space are used to program all the RAM cell locations in the LCA, adding further functionality to the LCA would not require more storage space.

## ARCHITECTURE

From the OPT, transmission is executed in the SCPC (single channel per carrier) mode. All scrambling, encoding, and error-code generation are performed by M/A-Com's proprietary transmit gate array. The gate array contains registers, allowing it to be programmed to transmit in different schemes and protocols, including SCPC mode.

The OPT receives a TDM (time division multiplexed) bitstream composed of 56 kb/s data channels in a modulated 3-MHz carrier. The bitstream contains a UW (unique word), and data and parity bits for each channel in each frame. The received carrier is demodulated by analog circuitry on the SCI, which passes the digital bitstream to the LCA.

To isolate the UW and lock onto the data, the LCA contains several counters and a state machine, configured in TDM synchronizer. The state machine controls the synchronization algorithm, which manipulates the frames.

The TDM synchronizer moves between four states (see Figure 4). The first state entails acquiring "sync" by recognizing the unique word in the unsynchronized data stream. Once the unique word is acquired without errors, the second state occurs. The circuit verifies "sync" by detecting the unique word again one frame later in the bitstream. Upon second detection, the circuit is considered in sync, and the synchronizer shifts to the third state—the sync state—where data are allowed to proceed as long as the system detects at least one unique word in every 11 frames.

The fourth state is entered every time a unique word is missed; the system stays in the fourth state until the unique word is found or is missed 11 consecutive times. If the unique word is found, the system returns to state three; if it is not found after 11 attempts, then the first state (the search mode) is initiated again. This method of operation ensures that the demultiplexer will remain locked even in the presence of random bit errors in the data stream.

After the unique word is detected, the receiver locks onto the data. The LCA chip then descrambles the data stream. The data is originally scrambled by the transmitter to place a fairly equal number of ones and zeros into the transmitted carrier. If this is not done, the transmitted carrier may not contain an even distribution of spectral components, which makes it difficult for a demodulator to acquire the carrier. The descrambling process is merely the reverse of the 9-bit scrambling procedure.

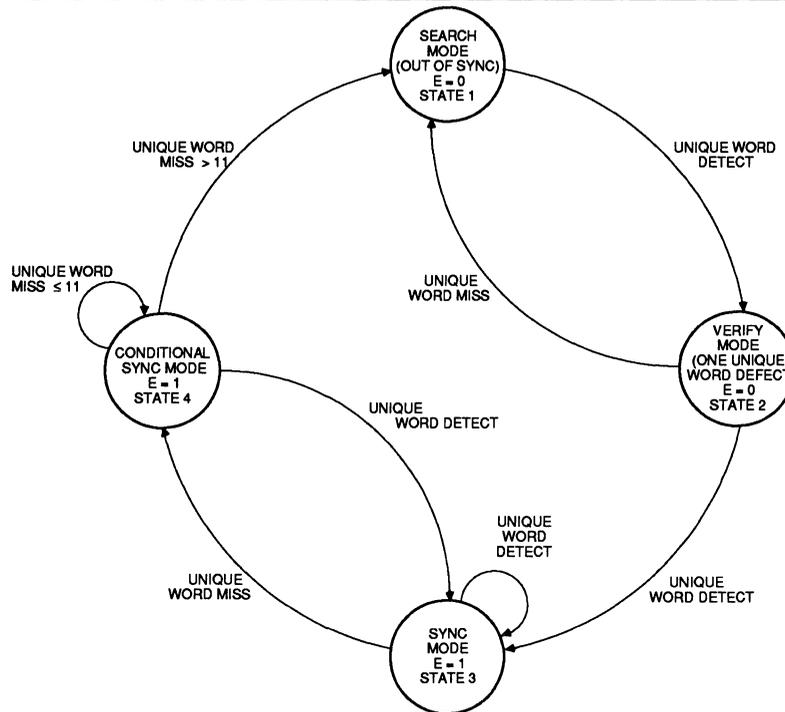
A single channel is isolated from the others by demultiplexing the descrambled data stream. The demultiplexing function is performed through a pair of counters that count the bits between unique words and tell the demultiplexer when data is available.

Once the incoming data stream has been descrambled and demultiplexed, it moves on to the M/A-Com proprietary convolutional decoder, a custom chip where error detection and correction is done on a per-channel basis. Decoded data is passed on to a microprocessor for data extraction.

## TESTING THE LCA

To test the TDM synchronizer, the LCA was loaded via the Xilinx in-circuit emulator and set into the test bed. We tested with a satellite simulator and found one design error. Both isolation and remedy of the fault were simple to perform, due to the reconfigurability of the part. Fault location was eased by choosing internal test nodes and connecting them to I/O pads. This technique made it possible to find the fault very quickly.

By using a satellite simulator we were able to insert errors into the data stream. We measured the time to lose sync and the time to acquire sync, and determined that the ripple counter was a little too slow for the required function. Since we were using an in-circuit emulator, it was very easy to reprogram the device. After the design was debugged, we left the simulator on-line for a week to ensure a thorough test of the Xilinx part under operational conditions. Our concern was how well the LCA would retain its configuration, since this information is stored by RAM cells. However, in our environment, it performed flawlessly.



1148 11

Figure 4. State Machine for the Time-division Multiplexer.

## OPEN-END DEVELOPMENT

Late into the design cycle we began to add additional planned functions to the LCA. Because we knew we could add these extra features, we finished the PCB layout and ordered PC boards without waiting for the final design. Then the process of adding putting functions into the LCA was begun.

Normally this time would have been used to design a test fixture. Instead, another LCA design was created to support a test implementation. Before the PCB was delivered, the test fixture simulating the system was built, primarily around the second Xilinx part. In the process of building the fixture, we discovered an error in the PCB layout, even before it was delivered. It was possible to fix the error by reconfiguring the LCA.

When the board was delivered, a new version of our logic design had been implemented in the Xilinx LCA, including the demultiplexing and descrambling functions.

## CONCLUSIONS

The flexibility of the Xilinx LCA lowers design costs, reduces project schedule risks, and reduces inventory risks. Using the LCA does not require much design sophistication, but rather a good general knowledge of

basic digital circuitry. For example, designers must be able to recognize the worst-case timing scenarios of their networks. Delay and system-speed considerations can now be checked with the Xilinx simulator, but at the time of our design, the simulator was still in beta test; we calculated the circuit behavior with preliminary timing software. Since then the simulator has been revised and its present version would have spotted our timing error.

Rather than packing complete design into the front end of an ASIC development, as is required for conventional gate arrays, the LCA offers the flexibility to indicate roles for the part. Designers can specify the I/O pins for the LCA then send the PC board to fabrication. While the board is in fabrication, designers can build into the LCA the gate-level logic they want and continue to make changes up until, and even after, the PCB is delivered.

After final product delivery, the on-board logic can still be reconfigured to match specific customer needs—without having to cast custom silicon for a few dozen units or changing the PC artwork. Great NRE savings are passed back to the customer. In summary, the LCA has proved to be an extremely efficient, useful, and cost-effective extension to our semicustom design capabilities.

Reprinted with permission from *VLSI System Design*.

## Faster Turnaround for a T1 Interface

by Carl Erite, Teltrend Inc., St. Charles, IL

Important design considerations for an interface system to a digital T1 network (which carries voice, data, video and fax traffic at rates up to 56 Kbytes/sec) include conserving board space, improving throughput and reducing power consumption. The user interface is achieved via a conventional four-wire loop providing independent transmit and receive capabilities. In designs that Teltrend Inc. initially considered for a single-user T1 interface, 5000 gates of conventional SSI/MSI glue logic were to be integrated using two custom gate arrays. However, a short development cycle and low market risks were also desired. This led to a search for an alternative to the time-consuming process of casting two gate arrays.

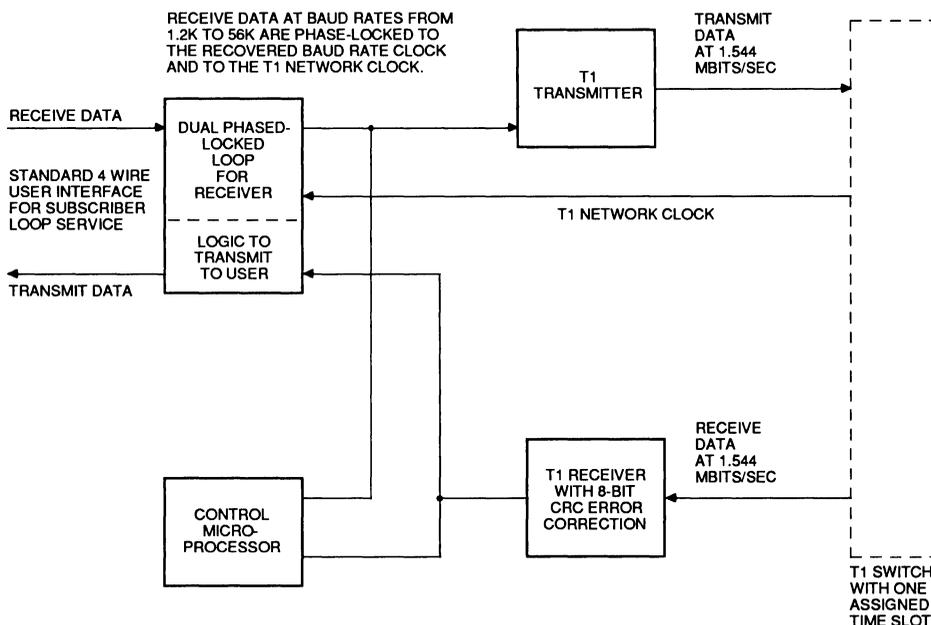
Upon completing the initial circuit design, a breadboard was built using CMOS SSI/MSI logic components. After the breadboard was working, integration path decisions were needed. Instead of hard-tooling two custom gate arrays, designers determined that three standard, programmable Xilinx Logic Cell Arrays (LCAs) met all of the

design requirements—high integration, high density, high performance, low cost, low risk and quick time-to-market.

The Xilinx devices implement a digital phase-locked loop, as well as the T1 transmitter and receiver. A Hitachi microprocessor provides overall intelligence to handle T1 controls, network code manipulation and other tasks.

The dual digital phase-lock loop provides the key function of the system. Data on the user interface is encoded with the clock signals, a process that may occur at various send/receive data rates. Data extraction from the user interface must be phase-locked and, at the same time, data must be synchronized with the T1 network clock. A Xilinx LCA implements the phase-locked loop that synchronizes both the interface and the T1 network.

The second LCA transmits data onto the T1 network. Here, data transmits serially at 1.544 Mbits/sec in one of the 24 assigned time slots. A unique data word to be



1148 12

**Figure 7.** Teltrend's digital T1 interface is built around three user-programmable Xilinx Logic Cell Arrays in lieu of two conventional gate arrays. One LCA implements a dual digital phase-lock loop around four-wire loop; other LCAs form both the transmitter and receiver logic circuits, including error correction.

transmitted is held in the LCA while logic synchronization determines the start of the first time slot or the beginning of the data frame. The assigned time slot is found by counting time slots from the start of a complete frame. After locating the assigned time slot, data is transmitted onto the T1 network.

A third LCA, complementary to the transmitter function, receives data. It also furnishes complete error correction for incoming data. Time-slot detection logic determines the start of data for the assigned channel. Serial data comes from the T1 network. After the LCA performs 8-bit error correction, the data passes to the processor and user interface.

The first iteration of the design was extracted directly from the CMOS breadboard schematics using the Xilinx XACT system running on an IBM PC/AT. The working design for the first device was completed in two weeks, with some time-critical elements moved off the chip. Designs for the second and third parts took about the same time, but additional interaction during the design process resulted in

higher performance in critical timing paths and higher overall device utilization. In all three designs, LCA logic resource utilization exceeded 95%.

All three designs are flip-flop intensive, involving multiple counters, shifters, registers and other memory-oriented functions. The LCAs provide more flip-flops per device than any other programmable logic alternative. Only a few simple 8-bit registers were implemented externally with octal devices. Next-generation designs will use Xilinx's compatible higher density devices to achieve greater logic density in the same socket.

Overall, the ability to enter the original design using the Xilinx LCA XACT design system ensured that all the integrated logic functioned as desired before the part was placed in the system. With a conventional gate array, the design might still be waiting for silicon, since turnaround times for production quantity gate arrays typically range from 8 to 16 weeks (production quantities).

Reprinted with permission from *ESD: The Electronic System Design Magazine*.

By Tom Liehe, Principal Design Engineer, Test Instrument Division, Honeywell Inc., Denver, Colo.

---

**Designers at Honeywell picked the RAM-based Xilinx LCA for its short development cycle, and realized savings in board real estate through its dynamic reprogrammability.**

Advances in one technology often lead to improvements in other, more dated design and manufacturing practices. A recent example of this occurred at Honeywell during the development of a high-capacity digital tape recorder.

Honeywell's original objective was to design the VLDS (very large data storage) recorder, taking maximum advantage of the available analog technology currently being used in standard VCRs for home use. The recorder developed under this program uses digital rotary technology to record large amounts of data on a standard VHS video cassette. It transfers data at a rate of 4 Mbauds, and is able to store 5.2 Gbauds of information on a single BHS tape. Its major planned application is in capturing and storing digital medical images, such as those produced by a CAT scanner.

When this recorder was in the prototype stage, it became apparent that the addition of an error-correction circuit would significantly enhance system performance. This requirement dictated the design of an entirely new and major logic circuit to accomplish the desired error correction.

Design of this circuitry would not normally be a problem, but at this stage of development, there were several challenges. First, the design allowed almost no circuit board space for addition of the error correction code (ECC) circuitry. Second, very tight deadlines were being faced if the promised delivery date was to be met.

The entire system was housed in a 19-inch-wide by 20-inch-deep rack-mounted cabinet. The cabinet already contained eight separate circuit boards, and there was room enough for only one additional board to incorporate the ECC circuitry. Space was at a premium. The goal was to design and manufacture a  $10^{-12}$  corrected bit error rate circuit that could be contained on one circuit card. The targeted time for completion of this work was three months.

Errors on tape typically are caused by tape defects, dirt, head clogs, etc. Because these error bursts can be thousands of bits long, sophisticated ECC techniques are required. Initially, two basic circuits, using Reed-Solomon algorithms and TTL technology, were designed. These were the ECC encoder and decoder.

The write portion of the circuitry (the encoder) uses a byte-wide linear feed-back shift register (LFSR) to create a 68-byte code word from each 64-byte incoming message block. During operation, parity check bits are computed based on the data within a block of the message to be encoded. These check bits are appended to the block to create the code word.

During decoding, the code words are checked for errors by regenerating the parity bits which are then compared with the check bits. If they match, it is assumed that no errors have occurred. If they do not, the pattern of mis-matches (called the syndrome of the error) is used to compute the corrected form of the message block.

The ECC decoder (the read circuit) required a partial syndrome generator and the solution of a set of simultaneous non-linear equations to determine error locations and values. This error-determination step is performed by a special-purpose processor with a microinstruction sequencer, a finite field arithmetic unit, two discrete registers and an eight-word memory. The correction step is then accomplished in circuitry whereby the error values are exclusive-ORed with the message at the address given by the previously computed error locations.

Using wrapped-wire techniques, a working prototype of the ECC circuitry was developed. However, it was quickly recognized that the long lead time required to design and fabricate a factory-programmed gate array to replace this prototype TTL circuit was not practical with the tight delivery schedule.

An option considered, but not chosen, was to design and fabricate a conventional gate array. The considerable design time required, together with the inherent risks associated with masking and manufacturing a custom logic circuit, made this an unattractive alternative.

## XILINX'S LCA

Finally, the search for an alternative solution led to the discovery of a programmable gate array known as the logic cell array (LCA), designed and manufactured by Xilinx Inc. (San Jose, Calif.). The LCA is a standard, off-the-shelf device that is custom configured to the customer's requirements by means of the Xilinx development system. This development package consists of a personal-computer-based software system combined with an in-circuit emulator.

Use of the LCA seemed to be the ideal solution to the time constraints. So, a Xilinx XC2064 LCA was selected. In this device, any logic function having up to four variables can be implemented in any one of the 64 configurable logic blocks (CLBs). Optionally, results can be stored in either a latch or a flip-flop. Thus, implementation of the design can be constrained by a fixed set of standard logic elements.

The I/O pins of this device also can be configured as registered inputs. The large number of flip-flops, plus the ability of each CLB to function as four-input exclusive-ORs, made this LCA ideal for ECC circuit implementation.

## MULTIFUNCTION CAPABILITY

One of the real benefits of this LCA is its multifunction capability. The capability of performing a number of functions with the same device provides optimum utilization of circuit board space. This was a real bonus with the VLDS recorder. At any given time, the VLDS operates in only the read or the write mode—it is never required to do both simultaneously. Consequently, the same LCA could be reconfigured electronically to perform one function in the write mode, and a completely different function in the read mode. This versatility eliminated the need for two separate circuits, and thereby conserved space.

The LCA has a usable density of 1,000- to 1,500-gate equivalents, and is capable of replacing up to 75 SSI/MSI devices, five to 15 PAL-type devices, or some combination of both. In the VLDS, the entire ECC encoder and the partial syndrome generator portion of the ECC decoder were replaced by the LCA. The initial encoder circuit used eight identical PALs, each of which implemented a 1-bit slice of the shift register, and four PROMs. The original partial syndrome generator design used six PALs and four 74LS374 tri-state octal flip-flops to store the four syndromes. Thus, the LCA replaced a total of 14 PALs, four 256k x 8 PROMs and four 74LS374s, or a total of 22 20-pin DIPs.

## ANOTHER BENEFIT

Another significant benefit derived from the use of the Xilinx LCA was reduced power consumption. The original bipolar IC design consumed approximately 12 W of power. Through the use of CMOS technology, the replacement LCA consumes only 50 mW of power. It should be pointed out that the bipolar version was capable of operating at a much higher clock rate than the LCA. However, the clock rate used in this particular design was only 2 MHz. The speed of the LCA was, therefore, adequate for the VLDS application.

Because the required logic circuitry was already designed and tested, the development of the configuration program for the LCA went very smoothly. It took only two days to configure the circuit using the Xilinx XACT LCA development system running on a standard, IBM-compatible personal computer. The primary effort involved was the partitioning of the logic to match the capabilities of the LCA.

For a regular, repetitive design, a small portion of the logic was defined. This portion was then copied and minor modifications were made to complete the design. The byte-oriented nature of the RS ECC circuitry lent itself to easy entry. Starting with tables showing the bits to be exclusive-ORed, the entire circuit was entered in a few hours.

The software simulation capability, which enabled the modeling of physical delays and logic functions, resulted in a very high design confidence factor before the first hardware checkout. The simulator provided both tabular output and logic analyzer style waveforms, which aided considerably in the visualization of the circuit performance. A high-level language program was used to generate expected results of the encoder, and to perform partial syndrome generator simulations. This greatly aided the evaluation of the simulation output.

By using the in-system emulation feature, configurations were loaded directly from the PC to an LCA mounted in the target system. Thus, the usual step of programming an EPROM from which the LCA can boot itself was eliminated. Initial design checkout of the ECC circuitry was performed using the emulator connected to the wrapped-wire board containing the discrete IC version.

There was a problem with the encoder circuit that was delaying data for an extra byte. Correcting this problem required removing the input flip-flops on the LCA. The entire process of reentering the LCA editor, removing the mouse and reloading the new configuration took no more than five minutes.

Compared with the time required to rework any other type of hardware, the LCA is the only way to go. Also, taking into consideration the high costs associated with reworking a factory-programmed gate array, or even a semi-custom PLD, the LCA technology is an extremely cost-effective alternative.

In summary, the Xilinx part was well suited for our application because of its high flip-flop count and its ability to be configured in exclusive-OR trees. Additionally, its capability of being electronically reconfigured while in the system (when switching from write to read) offered significant savings.

Further, power consumption was much lower than when

using equivalent discrete ICs. And finally, the ability to perform design entry, simulation, emulation and in-system testing through the software development system facilitated quick and easy implementation of the user's ideas.

Today, the Honeywell VLDS offer error correction as powerful as most major computer tape subsystems. It is ideally suited for the newly developing imaging technologies used in electronic office documents, advanced geophysical analysis and computer-aided graphic arts. Without the Xilinx logic cell array however, Honeywell could still be waiting for a custom gate array.

Reprinted with permission from *Electronic Engineering Times*.

by Rusty Woodbury, Interactive Educational Video, Salt Lake City, UT.

---

Reprinted with permission from *ESD: The Electrical System Design Magazine*.

The market for tools and overlay products for video pictures generated from laser disks is in its infancy. Applications for this emerging video-based technology can require high resolution and high performance, and the wide variety of video disk players employed means that problems associated with varied noise characteristics must be overcome. What works with one particular brand and model in the factory may falter with another brand in the field.

The Xilinx Logic Cell Array (LCA) helps to solve the problem of meeting different system requirements because the device elevates hardware to the same level of programmability as software. Before the LCA, once a design had been committed to hardware, revisions to the design could only be implemented via software changes.

Interactive Educational Video (IEV) has implemented three separate designs and logic replacements with the LCA. These functions reside on IBM PC expansion cards, where space limitations would ordinarily preclude such a design. Although application-specific video ICs could perform similar functions, they cost more than the LCA and offer lower performance.

The first application is a graphics engine that uses four LCAs. Here, the LCAs replace over 50 SSI/MSI chips, including four traditional programmable logic devices (PLDs).

One LCA functions as the address generator for the video memory. By relying on a pair of high-speed counters to locate horizontal and vertical coordinates, memory write functions (which implement line drawing) can perform at high rates. Given the slope, starting point and length of a line, the logic simply increments counters that point to video memory locations. Scanning and writing to the screen are interleaved. The data written to memory corresponds to a particular color and, by simple incremental additions to the slope of the address pointer counters, powerful line drawing functions are easily implemented.

Important to the design is the decision logic, which deter-

mines how to increment the counter. All of these functions, plus logic to generate the read/modify/write cycle timing, are implemented in a single LCA that replaces nine MSI parts, four of which are PLDs.

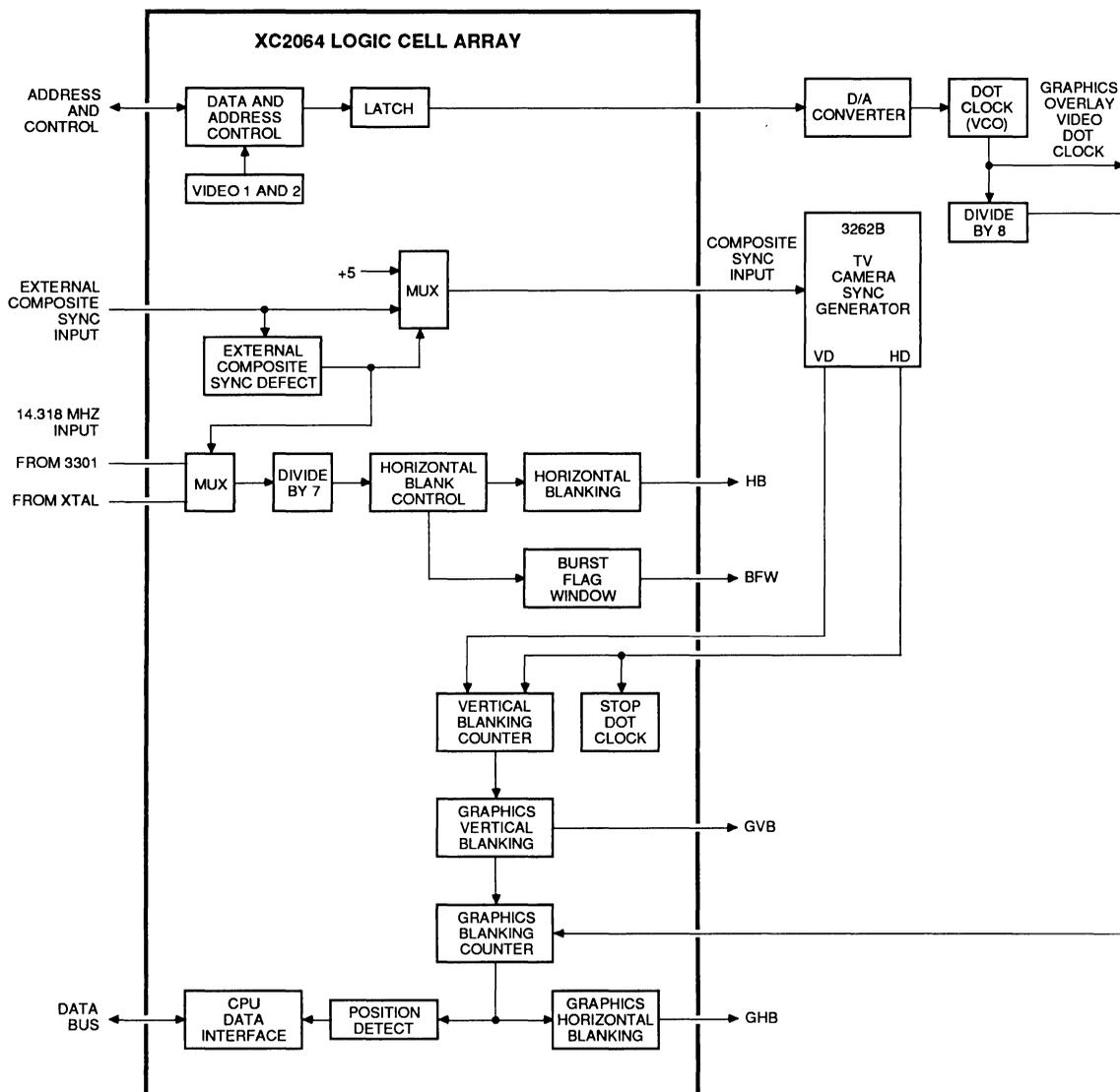
Two more LCAs implement a three-bit ALU. This technique achieves ultra-high-speed screen writes for both horizontal and vertical lines. For many applications, these are the most common lines drawn, so a special control bit is used to simultaneously modify pixels. Horizontal lines can be written at 14 Mpixels/sec instead of the normal 2 Mpixels/sec—a seven-fold improvement. Though more logic could be placed in these two devices, a bit-sliced logic approach permits continuous enhancements. Moreover, a board layout can be defined at the beginning of the product cycle while logic enhancements are made internally in the LCA. Nearly 30 SSI/MSI devices were integrated into the LCAs.

A fourth LCA fully implements the graphics engine. To read out data to the screen, scan counters point to memory. A shift register serializes at a rate of 14 Mpixels/sec. Using traditional MSI devices, these functions require about 10 chips.

The second design fabricated by IEV is a graphics controller (Figure 6). Using an external genlock IC, the LCA relies on an NTSC composite sync signal to generate timing signals for the CRT display. Instead of using PLDs, IEV uses the LCA to implement digital counter and timers. The result is higher performance and reduced complexity. The previous generation board has only half the functionality and demands four times the board space. To further reduce complexity, the same hardware can be used with a different configuration program to match a particular video disk player's noise characteristics. Without the LCA, this design needs eight PALs plus 12 to 15 MSI devices.

In another IEV design, a PC serial port emulator integrates a subset of the IBM PC serial port functions onto the graphics card, making an IBM serial card unnecessary. With the given space restrictions, this implementation proves particularly cost-effective. Seven PLDs are required to match this design.

Reprinted with permission from *ESD: The Electronic System Design Magazine*.



1148 13

**Figure 6.** IEV implemented an intelligent Graphics Overlay Controller microprocessor peripheral with one XC2064 Logic Cell Array, replacing eight PALs and 12 MSI devices. The controller generates all timing for a video graphics overlay by deriving the necessary timing from the underlying video disk signal.

---

---

**1 Programmable Gate Arrays**

**2 Product Specifications**

**3 Quality, Testing, Packaging**

**4 Technical Support**

**5 Development Systems**

**6 Applications**

**7 Article Reprints**



# Index

---

Index .....	8-1
Sales Office Listing .....	8-3

---

---

abort .....	2-15, 2-67	electrostatic discharge .....	3-8
adder .....	6-22	error correction .....	6-45, 7-19
alpha particle .....	3-8	FIFO .....	6-35
ASIC .....	1-9	flat pack .....	3-18
Automatic Place & Route .....	5-17	frequency counter .....	6-29
bar code reader .....	6-43	FutureNet .....	5-38
battery back-up .....	2-27	gate count .....	6-7
BIDI .....	2-60, 2-7	in-circuit verification .....	5-36
Boolean equations .....	5-35, 6-9	initialization .....	2-14, 2-66
buffers .....	2-10	input .....	(see IOB)
bulletin board .....	3-1, 4-4	input protection .....	2-27, 3-8
burn-in .....	2-100	interconnect .....	2-7, 2-13, 2-59, 2-60
busing .....	2-11	IOB .....	2-3, 2-56, 6-14
CLB .....	2-5, 2-57	IOB timing .....	2-42, 2-86, 6-14
CLB timing .....	2-40, 2-84, 6-16	latches .....	6-25
code conversion .....	6-30	latch-up .....	3-9
combinatorial functions .....	2-6, 2-58	length count .....	2-17, 2-67
comparator .....	6-22	library of macros .....	5-25
configuration .....	2-14, 2-66	logic analyzer .....	6-51
configuration data .....	2-16	logic synthesis .....	5-13, 5-34, 6-9
configuration memory .....	2-2, 2-55, 3-7, 3-11	long lines .....	2-11, 2-60
configuration pins .....	2-32	macros .....	5-25
cost .....	1-9	magic box .....	2-9, 2-60
counters .....	6-26, 7-22	majority logic .....	6-20
crystal oscillator .....	2-13, 2-62, 6-15	MAKEBITS .....	2-17, 5-21
Daisy .....	5-40	MAKEPROM .....	2-17, 5-21
daisy chain .....	2-20	master mode timing .....	2-45, 2-88
diagnostic .....	6-38	master parallel mode .....	2-14, 2-19, 2-66, 2-68
direct inputs .....	2-10	master serial mode .....	2-14, 2-18, 2-66, 2-69, 2-115
disk controller .....	7-1	memory cell .....	2-2, 2-55, 3-7, 3-11
DONE .....	2-23, 2-72	memory requirements .....	5-1
download .....	5-21	Mentor .....	5-42
DRAM controller .....	6-45	metastability .....	6-19
EditNet .....	2-7, 2-60	Micro Channel .....	6-41

---

MIL-STD-883 .....	3-1	RESET .....	2-15, 2-23, 2-67
military .....	2-97	retention .....	2-27
modes .....	2-14, 2-66	Schema II .....	5-39
multiplexer .....	2-12	schematic capture .....	5-10, 5-37
OrCAD .....	5-43	self test .....	6-38
ordering information .....	2-49, 2-91, 5-47, 5-49	serial PROM .....	2-113
oscillator .....	2-13, 2-23, 2-62, 6-15	SILOS .....	5-21, 5-33
output .....	(see IOB)	simulation .....	5-21
output inversion .....	2-4	sizing .....	6-3
output slew rate .....	2-3	slave mode .....	2-14, 2-20, 2-66, 2-71
package dimensions .....	2-50, 2-92	slave mode timing .....	2-48, 2-90
package pins .....	2-33	sockets .....	2-95
packages .....	2-49, 2-91, 3-17	soft errors .....	3-8
PALASM .....	5-13, 5-34, 6-9	specifications .....	2-37, 2-82
part numbers .....	2-49	start-up .....	2-17
pattern detector .....	6-32	state machine .....	6-36
PC requirements .....	5-1	subtractor .....	6-22
PC-SILOS .....	5-33	supply voltage .....	2-26, 2-76
PDS2XNF .....	5-34, 6-9	switch matrix .....	2-9, 2-60
performance .....	2-24, 2-74	T1 Interface .....	7-17
peripheral mode .....	2-14, 2-20, 2-66, 2-70	temperature .....	2-26, 2-76, 3-9
peripheral mode timing .....	2-47, 2-89	testing .....	3-11
pin description .....	2-30, 2-78, 2-114	three-state buffers .....	2-12
pinouts .....	2-80, 2-33	threshold .....	2-15, 2-71
PIP .....	2-8, 2-60	tie option .....	2-17
power dissipation .....	2-26, 2-28, 2-64, 2-77, 6-16	timing .....	2-40, 2-84, 6-16
power distribution .....	2-24, 2-64	timing calculator .....	2-24
power-down .....	2-15, 2-27, 2-72	timing, CLB .....	2-25, 2-84
power-on .....	2-15, 2-67	timing, interconnect .....	2-26, 2-75
preamble .....	2-15, 2-67	timing, IOB .....	2-25, 2-86
process .....	1-14	timing, PROM .....	2-119
PROM programmer .....	5-46	toggle frequency .....	2-72, 2-23
PROM programming .....	2-116	TTL - MSI library .....	5-44
quality .....	3-1	video control .....	7-22
radiation .....	3-9	wired-AND .....	2-12
re-program .....	2-23, 2-72	XACT Design Editor .....	5-6, 5-19, 5-32
readback .....	2-22, 2-72	XACTOR .....	5-6, 5-36
readback timing .....	2-48, 2-90	XNF2LCA .....	5-16
reconfigure .....	7-19	XNFMERGE .....	5-17, 5-34, 6-9
reliability .....	3-1	XNFOPT .....	5-16, 5-34, 6-9



# Sales Offices

## SALES OFFICES

### EUROPE

XILINX, Ltd.  
Station House  
Bepton Road, Midhurst  
Sussex GU 29 2RE  
England  
Tel: 073081 6725  
FAX: 073081 4910

### JAPAN

XILINX JAPAN  
3-6, Ginza Nichome  
Okura Shoji Bldg.  
Chuo-ku, Tokyo 104, Japan  
Tel: 03-561-7763  
FAX: 03-563-0784

### NORTH AMERICA

XILINX, INC.  
2069 Hamilton Avenue  
San Jose, CA 95125  
(408) 559-7778  
TWX: 510-600-8750  
FAX: 408-559-7114

XILINX, INC.  
1270 Oakmead Parkway  
Suite 201  
Sunnyvale, CA 94086  
(408) 245-1361  
FAX: 408-245-0517

XILINX, INC.  
2659 Townsgate Road  
Suite 101  
Westlake Village, CA 91361  
(805) 494-5026  
FAX: 805-496-0239

XILINX, INC.  
20 Mall Road, Suite 469  
Burlington, MA 01803  
(617) 229-7799  
FAX: 617-273-0228

XILINX, INC.  
65 Valley Stream Parkway  
Suite 140  
Malvern, PA 19355  
(215) 296-8302  
FAX: 215-296-8378

XILINX, INC.  
919 North Plum Grove Road  
Suite A  
Schaumburg, IL 60173  
(312) 605-1972  
TLX: 510-601-5973  
FAX: 312-605-1976

## DOMESTIC SALES

### ALABAMA

Technology Marketing  
Associates, Inc.  
3315 So. Memorial Pkwy.  
Bldg. 100  
Huntsville, AL 35801  
(205) 883-7893  
FAX: 205-882-6162

### ARIZONA

Quatra Associates  
4645 S. Lakeshore Dr.,  
Suite 1  
Tempe, AZ 85282  
(602) 820-7050  
TWX: 910-950-1153  
FAX: 602-820-7054

### ARKANSAS

Bonsler-Philhower Sales  
4614 S. Knoxville Avenue  
Tulsa, OK 74135  
(918) 744-9964  
TWX: 510-600-5274

### CALIFORNIA

SC Cubed  
468 Pennsfield Place  
Suite 101A  
Thousand Oaks, CA 91360  
(805) 496-7307  
FAX: 805-495-3601

SC Cubed  
17862 17th. St.  
Tustin, CA 92680  
(714) 731-9206  
FAX: 714-731-7801

Quest-Rep Inc.  
9444 Farnham St., Suite 107  
San Diego, CA 92123  
(619) 565-8797  
FAX: 619-565-8990

Norcomp  
3350 Scott Blvd., Suite 24  
Santa Clara, CA 95054  
(408) 727-7707  
TWX: 510-600-1477  
FAX: 408-986-1947

### COLORADO

Front Range Marketing  
3100 Arapahoe Rd.,  
Suite 404  
Boulder, CO 80303  
(303) 443-4780  
TWX: 910-940-3442  
FAX: 303-447-0371

## CONNECTICUT

Lindco Associates, Inc.  
Cornerstone  
Professional Park  
Suite C-101  
Woodbury CT, 06798  
(203) 266-0728  
FAX: 203-266-0784

## DELAWARE

Micro Comp, Inc.  
1421 S. Caton Avenue  
Baltimore, MD 21227  
(301) 644-5700  
TWX: 510-600-9460  
FAX: 301-644-5707

## FLORIDA

Technology Marketing  
Associates, Inc.  
8000 Orange Ave., Suite 100  
Orlando, FL 32809  
(407) 857-3760  
TWX: 510-600-4721  
FAX: 407-857-6412

Technology Marketing  
Associates, Inc.  
1280 S.W. 36th Ave.,  
Suite 201  
Pompano Beach, FL 33069  
(305) 977-9006  
FAX: 305-977-9044

Technology Marketing  
Associates, Inc.  
1300 S. Harbor City Blvd.  
Suite 14  
Melbourne, FL 32901  
(407) 676-3776  
FAX: 407-676-4231

Technology Marketing  
Associates, Inc.  
11100 66 St. No., Suite 25  
Largo, FL 34643  
(813) 541-1591  
FAX: 813-545-8617

## GEORGIA

Technology Marketing  
Associates, Inc.  
6855 Jimmy Carter Boulevard  
Suite 2420  
Norcross, GA 30071  
(404) 446-3565  
FAX: 404-446-0569

## IDAHO (Southwest)

Thorson Company Northwest  
12301 N.E. 10th Place  
Bellevue, WA 98005  
(206) 455-9180

## ILLINOIS

Beta Technology Sales, Inc.  
501 Mitchell Road  
Glendale Heights, IL 60139  
(312) 790-9886  
TWX: 62885853

## INDIANA

Arete Sales Inc.  
2260 Lake Ave Suite 250  
Fort Wayne, IN 46805  
(219) 423-1478  
FAX: 219-420-1440

Arete Sales Inc.  
918 Fry Road Suite B  
Greenwood, IN 46142  
(317) 882-4407  
FAX: 317-888-8416

## IOWA

Advanced Technical Sales  
375 Collins Road N.E.  
Cedar Rapids, IA 52402  
(319) 365-3150  
FAX: 319-393-7258

## KANSAS

Advanced Technical Sales  
610 N. Mur-Len, Suite 8  
Olathe, KS 66062  
(913) 782-8702  
FAX: 913-782-8641  
TWX: 910-350-6002

## LOUISIANA (Northern)

Bonsler-Philhower Sales  
689 W. Renner Rd., Suite III  
Richardson, TX 75080  
(214) 234-8438  
TWX: 910-867-4752  
FAX: 214-437-0897

## LOUISIANA (Southern)

Bonsler-Philhower Sales  
10700 Richmond, Suite 150  
Houston, TX 77042  
(713) 782-4144  
TWX: 910-350-3451

## MAINE

Mill-Bern Associates, Inc.  
2 Mack Road  
Woburn, MA 01801  
(617) 932-3311  
TWX: 710-332-0077  
FAX: 617-932-0511

## MARYLAND

Micro Comp, Inc.  
1421 S. Caton Avenue  
Baltimore, MD 21227  
(301) 644-5700  
TWX: 510-600-9460  
FAX: 301-644-5707

## MASSACHUSETTS

Mill-Bern Associates, Inc.  
2 Mack Road  
Woburn, MA 01801  
(617) 932-3311  
TWX: 710-332-0077  
FAX: 617-932-0511

## MICHIGAN

A.P. Associates  
810 E. Grand River  
Brighton, MI 48116  
(313) 229-6550  
TWX: 816-287-310  
FAX: 313-229-9356

## MINNESOTA

Com-Tek  
6525 City West Parkway  
Eden Prairie, MN 55344  
(612) 941-7181  
TWX: 310-431-0122  
FAX: 612-941-4322

## MISSOURI

Advanced Technical Sales  
1810 Craig Road, Suite 125  
St. Louis, MO 36146  
(314) 878-2921  
FAX: 314-878-1994

## NEVADA

Norcomp  
(Excluding Clark County)  
3350 Scott Blvd., Suite 24  
Santa Clara, CA 95054  
(408) 727-7707  
TWX: 510-600-1477

Quatra Associates  
(Clark County)  
4645 S. Lakeshore Dr.,  
Suite 1  
Tempe, AZ 85282  
(602) 820-7050  
FAX: 602-820-7054

## NEW HAMPSHIRE

Mill-Bern Associates, Inc.  
2 Mack Road  
Woburn, MA 01801  
(617) 932-3311  
TWX: 710-332-0077  
FAX: 617-932-0511

## Sales Offices

### NEW JERSEY (Northern)

Parallax  
734 Walt Whitman Road  
Melville, NY 11747  
(516) 351-1000  
FAX: 516-351-1606

### NEW JERSEY (Southern)

Delta Technical Sales, Inc.  
3901 Commerce Avenue  
Suite 180  
Willow Grove, PA 19090  
(215) 657-7250  
TWX: 510-601-1856  
FAX: 215-657-3781

### NEW MEXICO

Quatra Associates  
9704 Admiral Dewey N.E.  
Albuquerque, NM 87111  
(505) 821-1455

### NEW YORK (Metro)

Parallax  
734 Walt Whitman Road  
Melville, NY 11747  
(516) 351-1000  
FAX: 516-351-1606

### NEW YORK

Gen-Tech Electronics  
4855 Executive Drive  
Liverpool, NY 13088  
(315) 451-3480  
TWX: 710-545-0250  
FAX: 315-451-0988

Gen-Tech Electronics  
41 Burning Tree Lane  
Penfield, NY 14526  
(716) 381-5159

Gen-Tech Electronics  
70 Sandoris Circle  
Rochester, NY 14622  
(716) 467-5016

Gen-Tech Electronics  
5 Arbutus Lane  
Binghamton, NY 13901  
(607) 848-8833

### NORTH CAROLINA

The Novus Group, Inc.  
5337 Trestlewood Lane  
Raleigh, NC 27610  
(919) 833-7771  
TWX: 510-600-0558  
FAX: 919-856-1644

### NORTH DAKOTA

Com-Tek  
6525 City West Parkway  
Eden Prairie, MN 55344  
(612) 941-7181  
TWX: 310-431-0122  
FAX: 612-941-4322

### OHIO

Bear Marketing, Inc.  
P.O. Box 427  
3623 Brecksville Road  
Richfield, OH 44286-0177  
(216) 659-3131  
FAX: 216-659-4823  
TWX: 810-427-9100

Bear Marketing, Inc.  
240 W. Elmwood Drive  
Suite 1002  
Centerville, OH 45459  
(513) 436-2061  
FAX: 513-436-9137

### OKLAHOMA

Bonser-Philhower Sales  
4614 S. Knoxville Avenue  
Tulsa, OK 74153  
(918) 744-9964  
TWX: 510-600-5274

### OREGON

Thorson Company Northwest  
6700 S.W. 105th Ave.,  
Suite 104  
Beaverton, OR 97005  
(503) 844-5900  
FAX: 503-644-5919

### PENNSYLVANIA (Eastern)

Delta Technical Sales, Inc.  
3901 Commerce Avenue  
Suite 180  
Willow Grove, PA 19090  
(215) 657-7250  
TWX: 510-601-1856  
FAX: 215-657-3781

### PENNSYLVANIA (Western)

Bear Marketing, Inc.  
300 Mt. Lebanon Blvd.  
Pittsburg, PA 15234  
(412) 531-2002  
FAX: 412-531-2008

### PUERTO RICO

Mill-Bern Associates, Inc.  
2 Mack Road  
Woburn, MA 01801  
(617) 932-3311  
TWX: 710-332-0077  
FAX: 617-932-0511

Technology Marketing  
Associates, Inc.  
1280 S.W. 36th Ave.,  
Suite 201  
Pompano Beach, FL 33069  
(305) 977-9006  
TWX: 510-601-0120  
FAX: 305-977-9044

### RHODE ISLAND

Mill-Bern Associates, Inc.  
2 Mack Road  
Woburn, MA 01801  
(617) 932-3311  
TWX: 710-332-0077  
FAX: 617-932-0511

### SOUTH CAROLINA

The Novus Group, Inc.  
5337 Trestlewood Lane  
Raleigh, NC 27610  
(919) 833-7771  
TWX: 510-600-0558  
FAX: 919-839-0791

### SOUTH DAKOTA

Com-Tek  
6525 City West Parkway  
Eden Prairie, MN 55344  
(612) 941-7181  
TWX: 310-431-0122  
FAX: 612-941-4322

### TEXAS

Bonser-Philhower Sales  
8240 MoPac Expwy.,  
Suite 135  
Austin, TX 78759  
(512) 346-9186  
TWX: 910-997-8141  
FAX: 512-346-2393

Bonser-Philhower Sales  
10700 Richmond, Suite 150  
Houston, TX 77042  
(713) 782-4144  
TWX: 910-350-3451  
FAX: 713-789-3072

Bonser-Philhower Sales  
689 W. Renner Rd., Suite 101  
Richardson, TX 75080  
(214) 234-8438  
TWX: 910-867-4752  
FAX: 214-437-0897

### TEXAS (El Paso County)

Quatra Associates  
9704 Admiral Dewey N.E.  
Albuquerque, NM 87111  
(505) 821-1455

### UTAH

Front Range Marketing  
7050 Union Park Center  
Suite 440  
Midvale, UT 84047  
(801) 566-2500  
FAX: 801-566-2951

### VERMONT

Mill-Bern Associates, Inc.  
2 Mack Road  
Woburn, MA 01801  
(617) 932-3311  
TWX: 710-332-0077  
FAX: 617-932-0511

### VIRGINIA

Micro Comp, Inc.  
1421 S. Caton Avenue  
Baltimore, MD 21227  
(301) 644-5700  
TWX: 510-600-9460  
FAX: 301-644-5707

### WASHINGTON

Thorson Company Northwest  
12301 N.E. 10th Place  
Bellevue, WA 98005  
(206) 455-9180  
FAX: 206-455-9185  
TWX: 910-443-2300

### WASHINGTON (Vancouver, WA only)

Thorson Company Northwest  
6700 S.W. 105th Ave.,  
Suite 104  
Beaverton, OR 97005  
(503) 644-5900  
FAX: 503-644-5919

### WASHINGTON D.C.

Micro Comp, Inc.  
1421 S. Caton Avenue  
Baltimore, MD 21227  
(301) 644-5700  
TWX: 510-600-9460  
FAX: 301-644-5707

### WEST VIRGINIA

Bear Marketing, Inc.  
1563 East Dorothy Lane  
Suite 104  
Kettering, OH 45429  
(513) 299-5877  
FAX: 513-299-0756

### WISCONSIN (Western)

Com-Tek  
6525 City West Parkway  
Eden Prairie, MN 55344  
(612) 941-7181  
TWX: 310-431-0122  
FAX: 612-941-4322

### WISCONSIN (Eastern)

Beta Technology Sales, Inc.  
9401 Beloit, Suite 304C  
Milwaukee, WI 53227  
(414) 543-8609

### CANADA

#### BRITISH COLUMBIA (Vancouver)

Thorson Company Northwest  
12301 N.E. 10th Place  
Bellevue, WA 98005  
(206) 455-9180

#### ONTARIO

Electro Source, Inc.  
300 March Road, Suite 205  
Kanata, Ontario K2K 2E2  
(613) 592-3214  
FAX: 613-592-4256

Electro Source, Inc.  
230 Galaxy Boulevard  
Rexdale, Ontario M9W 5R8  
(416) 675-4490  
TWX: 06-989271  
FAX: 416-675-6871

### QUEBEC

Electro Source  
6600 TransCanada Hwy  
Suite 420 Point Claire  
Quebec H9R 4S2  
(514) 630-7486  
FAX: 514-630-7421

## INTERNATIONAL SALES

### AUSTRALIA

ACD/ITRONICS  
106 Belmore Rd. North  
Riverwood, N.S.W. 2210  
Tel: Sydney 534-6200  
FAX: Sydney 534-4910

ACD/ITRONICS  
Unit 2, 17-19 Melrich Road  
Bayswater VIC 3153  
P.O. Box 139  
Bayswater VIC 3153  
Tel: Melbourne (03) 762 7644  
Fax: Melbourne (03) 762 5446

ACD/ITRONICS  
55 Noreen Street  
Chapel Hill QLD 4069  
Tel: QLD 878 1488  
Fax: QLD 878 1490

### AUSTRIA

Elijapex GmbH  
Eitnergasse 6  
A-1232 Wien  
Austria  
(01) 86 3211  
FAX: (01) 86 3211 200

### BELGIUM & LUXEMBURG

Rodelco NV Electronics  
Excelsiorlaan 45, Bus 3  
1930 Zaventem  
Belgium  
(02) 720-5013  
TLX: 61415  
FAX: (02) 720-2048

### BRAZIL

International Trade  
Development Corporation  
450 San Antonio Road  
Suite 32  
Palo Alto, CA 94306  
(415) 856-6686  
Telex: 650-282-9742

### SOUTHEAST ASIA

Excel Associates, Ltd.  
1502 Austin Tower  
22-26A Austin Avenue  
Tsimshatsui, Kowloon  
Hong Kong  
Tel: 852-3-7210900  
FAX: 852-3-696826  
TLX: 30841

**DENMARK**

Saga Elektronik ApS  
Christianshusvej 4G  
DK-2970 Horsholm  
Denmark  
Tel: (02) 572677  
TLX: 37903  
FAX: (02) 572809

**FINLAND**

Sattco AB  
Pyramidvägen 9B  
S-171 36 Solna  
Stockholm, Sweden  
Tel: (08) 7340040  
TLX: 11588  
FAX: (08) 7349155

**FRANCE**

Repronc  
11, Escalier des Ulis  
91400 Orsay, France  
Tel: (16) 928 8700  
FAX: (16) 928 1750

R.T.F. Gentilly  
9, rue d'Arcueil  
94253 Gentilly Cedex,  
France  
Tel: (16) 4 66 41 11 0  
TLX: 2010169F  
FAX: (16) 4 66 44 19 9

R.T.F. (Radio Television  
Francaise S.A.)  
13, rue Lhote  
33000 Bordeaux, France  
Tel: 56 52 99 59  
TLX: 560627  
FAX: 56 48 17 83

R.T.F. Quest  
3, rue de Paris  
35510 Cesson Sevigne,  
France  
Tel: 99 83 84 85  
TLX: 741127  
FAX: 99 83 80 83

R.T.F. Sud Quest  
Avenue de la Mairie  
31320 Escalquens, France  
Tel: 61 81 51 57  
TLX: 520927F  
FAX: 61 81 22 36

R.T.F. Rhone Alpes  
St. Mury, Le Vaucanson  
38240 Maylan, France  
Tel: 76 90 11 88  
TLX: 980796  
FAX: 76 41 04 09

**GERMANY**

Metronik  
Leonhardsweg 2  
Postfach 13 28  
8025 Unterhaching  
München, Germany  
Tel: (089) 611080  
TLX: 897434  
FAX: (089) 611 6468

Metronik  
Semerteichstrasse 92  
4600 Dortmund 30  
Dortmund, Germany  
Tel: (0231) 423037/38  
TLX: 8227082

Metronik  
Osterbrookweg 61  
2000 Schenefeld  
Hamburg, Germany  
Tel: (040) 8304061  
TLX: 2162488

Metronik  
Siemensstrasse 4-6  
6805 Heddesheim  
Mannheim, Germany  
Tel: (06203) 4701-03  
TLX: 465053

Metronik  
Laufamholzstr. 118  
8500 Nürnberg 30  
Nürnberg, Germany  
Tel: (0911) 590061/62  
TLX: 626205

Metronik  
Löwenstr. 37  
7000 Stuttgart 70  
Stuttgart, Germany  
Tel: (0711) 764033/35  
TLX: 7255228

**HONG KONG**

Excel Associates, Ltd.  
1502 Austin Tower  
22-26A Austin Avenue  
Tsimshatsui, Kowloon  
Hong Kong  
Tel: 3-7210900  
FAX: 3-696826  
TLX: 30841

**ISRAEL**

E.I.M International Ltd.  
8, Eric Zolz Street  
P.O. Box 4000  
Petach Tiqva  
Tel Aviv, Israel  
Tel: (3) 92 33257  
FAX: (3) 924 4857  
TLX: 3811 44 E.I.M.I.L.

**ITALY**

ACSIS S.R.L.  
Via Alberto Mario, 26  
20149 Milano, Italy  
Tel: (02) 4390832  
TLX: 326566  
FAX: (02) 4697607

Celdis Italiana S.P.A.  
Via F.lli Gracchi 36  
20092 Cinisello Balsamo  
Milano, Italy  
Tel: (02) 61 839 1  
TLX: 334887  
FAX: (02) 61 735 13

Celdis Italiana S.P.A.  
Via Massarenti 219/4  
40138 Bologna, Italy  
Tel: (051) 53 333 6

Celdis Italiana S.P.A.  
Via Savelli 15  
35100 Padova, Italy  
Tel: (049) 77 209 9

Celdis Italiana S.P.A.  
Via G. Pitre' 11  
00162 Roma, Italy  
Tel: (06) 42 897 1

Celdis Italiana S.P.A.  
Via Mombarcaro 96  
10136 Torino, Italy  
Tel: (011) 32 993 88

**JAPAN**

Okura & Co., Ltd.  
6-12, Ginza Nichome  
Chuo-Ku  
Tokyo, 104 Japan  
Tel: (03) 566 6361  
TWX: J22306  
FAX: (03) 563 5447

**KOREA**

Excel-Tech  
6th Floor, Dae Ha Bldg.  
Room 605  
14-11 Yedeuido-Dong  
Yeongdeungpo-gu  
Seoul, Korea  
Tel: 02-785 1186  
FAX: 02-784 4060

**THE NETHERLANDS**

Rodelco Electronics  
Takkebijsters 2  
P.O. Box 6824  
4802 HV Breda  
Tel: (076) 784911  
TLX: 54195  
FAX: (076) 710029

**NORWAY**

Sattco AB  
Pyramidvägen 9B  
S-171 36 Solna  
Stockholm Sweden  
Tel: (08) 7 340040  
TLX: 11588  
FAX: (08) 7 349155

**SINGAPORE**

Excel Associates Ltd.  
Singapore Representative  
Office  
111 North Bridge Road  
#11-04/06 Peninsula Plaza  
Singapore 0617  
Tel: 3366577  
FAX: 3395291  
Telex: RS 36033 WWBCS

**SPAIN**

ADM Electronica SA  
Menorea No. 3 Entrepantia  
Madrid 28009  
Spain  
Tel: (01) 409 4725  
FAX: (01) 409 5215

**SWEDEN**

Sattco AB  
Pyramidvägen 9B  
S-171 36 Solna  
Stockholm Sweden  
Tel: (08) 7 340040  
TLX: 11588  
FAX: (08) 7 349155

**SWITZERLAND**

Data Comp AG  
Silberstrasse 10  
CH-8953 Dietikon  
Tel: (01) 7405140  
Telex: 827750  
FAX: (01) 7413423

**TAIWAN**

Molecatec, Inc.  
6th Floor Hamburg Building  
260, Section 3  
Nanking East Road  
Taipei, Taiwan R.O.C.  
Tel: (02) 7219353  
FAX: (02) 7216193  
TLX: 29214 MTEX

Sertek Int'l Inc.  
15/F, 135 Sec 2  
Chien Kuo N. Road  
Taipei 10479  
Taiwan R.O.C.  
Tel: 2 501 0055  
Fax: 2 501 2521  
Telex: 23756 Sertek

**UK AND IRELAND**

Microcall, Ltd.  
17 Thame Park Road  
Thame  
Oxon OX9 3XD  
England  
Tel: (084) 421 5405  
TLX: 837457  
FAX: (084) 421 4267

Microcall, Ltd.  
The Genesis Centre  
Birchwood Science Park  
Garrett Field  
Warrington WA3 7BN  
England  
Tel: (0925) 825065

**Distributed By**

Hamilton/Avnet  
locations throughout  
the U.S. and Canada.  
1-800-HAM-ASIC  
FAX: 408-743-3003

Western Microtechnology  
12900 Saratoga Ave.  
Saratoga, CA 95070  
(408) 725-1660  
TWX: 910-338-0013  
FAX: 408-255-6491

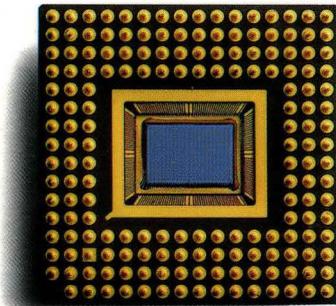
Insight Electronics  
6885 Flanders Drive  
San Diego, CA 92121  
(619) 587-9757  
TWX: 183035-UD  
FAX: 619-587-1380

Phase 1 Technology  
Corporation  
1110 Rte. 109  
N. Lindenhurst, NY 11757  
(516) 957-4900  
FAX: 516-957-4909

Nu Horizons  
Electronics Corp.  
6000 New Horizons Blvd.  
Amityville, New York 11701  
(516) 226-6000  
FAX: 516-226-6262

Marshall Industries  
locations throughout  
the U.S. and Canada.  
(818) 459-5500  
FAX: 818-459-5660





The Programmable Gate Array Company.  
2069 Hamilton Avenue, San Jose, CA 95125. (408) 559-7778