ACT™ Family Field Programmable Gate Array DATABOOK

April 1990

ACT™ Family Field Programmable Gate Array Databook

April 1990

ACT, Action Logic, Activator, Actionprobe and PLICE are trademarks of Actel Corporation.
Apollo, LCA, Mentor Graphics, NETED, OrCAD, OrCAD/SDT, OrCAD/VST, PAL, QuickSim, Sun, Sun 3, Sun 4, SYMED, Valid, ValidGED, ValidSIM, Viewdraw, Viewlogic, Viewsim, and Workview are all trademarks or registered trademarks of their respective manufacturers.

Actel Corporation reserves the right to make changes to any products or services herein at any time without notice. Actel does not assume any responsibility or liability arising out of the application or use of any product or service described except as expressly agreed to in writing by Actel.

© 1990 Actel Corporation

ACT™ Family Field Programmable Gate Arrays

Product Data 1

Reliability Report 2

Applications 3

Article Reprints 4

ACT™ Family Field Programmable Gate Array Databook

Order of Contents

Section 1: Product Data

ACT 1 Field Programmable Gate Arrays 1-1
ACT 1 Military Field Programmable Gate Arrays 1-27
Action Logic System FPGA Design Environment 1-41
ACT 2 Field Programmable Gate Arrays (Preliminary) 1-47
Activator 2 Programmer/Tester/Debugger (Preliminary) 1-49

Section 2: Reliability Report

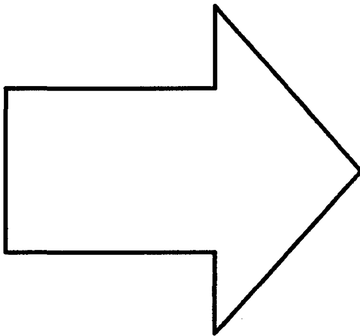
Reliability Report 2-1

Section 3: Applications

Which Actel array should you use? 3-1
Efficient Logic Conversions 3-3
The Actel Timer 3-15
Using Actionprobe Diagnostic Tools 3-17
Metastability 3-19
Using the Actel Debugger as a Functional Tester 3-21
Calculating Power in Actel's ACT 1 Arrays 3-25
Three-Stating ACT 1010/1020 Designs 3-29
Multi-ASIC and System Simulation with Actel FPGAs 3-31
Constructing RAM with ACT 1 Macros 3-33
Using the DRAM Controller Supermacro 3-39
Using the DMA Controller Supermacro 3-43
Using the SCSI Interface Controller Supermacro 3-45
UART Design 3-49
Actel TA181 ALU 3-61
Actel Fast Adders 3-63
Eight-Bit Twos Complement Multiplier 3-65

Section 4: Article Reprints

AR-1: Analyze FPLD Architectures, Performance, and Development Tools to Optimize Design-Dependent Selection 4-1
AR-2: A CMOS Electrically-Configurable Gate Array 4-11
AR-3: An Architecture for Electrically-Configurable Gate Arrays 4-23
AR-4: Dielectric-Based Antifuse for Logic and Memory ICs 4-29
AR-5: Field Programmable Gate Array with Non-Volatile Configuration 4-35
AR-6: Testing Antifuse-Based FPGAs 4-45
AR-7: Compare ASIC Capacities with Gate Array Benchmarks 4-49



Product Data 1

Reliability Report 2

Applications 3

Article Reprints 4

ACT 1 Field Programmable Gate Arrays 1-1

ACT 1 Military Field Programmable Gate Arrays 1-27

Action Logic System FPGA Design Environment 1-41

ACT 2 Field Programmable Gate Arrays 1-47

Activator 2 Programmer/Tester/Debugger 1-49

ACT™ 1 Field Programmable Gate Arrays

ACT 1010
ACT 1020

Features

- High Gate Count
  - ACT™ 1010: 1200 gate array gates (3000 PLD/LCA equivalent gates)
  - ACT 1020: 2000 gate array gates (6000 PLD/LCA equivalent gates)
- Field Programmability Allows Instant Prototypes and Production
- Toggle Rates to 70 MHz
- I/O Drive to 8 mA
- System-Level Performance to 40 MHz
- Gate Array Architecture Allows Completely Automatic Place and Route
- Non-Volatile, Permanent Programming
- Built-In Clock Distribution Network
- Built-In Diagnostic Probe Pins
- Low-Power CMOS Technology
- Fully Supported by Actel's Action Logic™ System

Description

ACT 1010 and ACT 1020 devices are the first members of a family of field programmable gate arrays (FPGAs) offered by Actel. These devices are implemented in silicon gate, 2-micron, two-level metal CMOS, employing Actel's PLICE™ antifuse technology. The unique architecture offers gate array flexibility, high performance, and instant turnaround through user programming. Utilizations as high as 95% permit designs of 1200 gates for the ACT 1010, and 2000 gates for the ACT 1020.

ACT 1010/1020 devices also provide system designers with unique on-chip diagnostic probe capabilities, allowing convenient testing and debugging. Additional features include an on-chip clock driver with a hardwired distribution network. The network provides efficient clock distribution with minimum skew.

The user-configurable I/Os are capable of driving at both TTL and CMOS drive levels. The ACT 1010/1020 devices are available in 44-, 68-, and 84-pin plastic leaded chip carrier and J-leaded cerquad packages, and an 84-pin pin grid array package.

A security fuse may be programmed to disable all further programming and protect the design from being copied or reverse engineered.

The Action Logic System

ACT 1010/1020 devices are supported by Actel's Action Logic System, allowing logic design implementation with minimum effort. The Action Logic System (ALS) interfaces with the resident CAE system to provide a complete gate array design environment: schematic capture, simulation, fully automatic place and route, timing verification, and device programming. The Action Logic System is available for 386 PC and Apollo™ and Sun™ workstations, running Viewlogic®, Mentor Graphics™, Valid™, and OrCAD™.

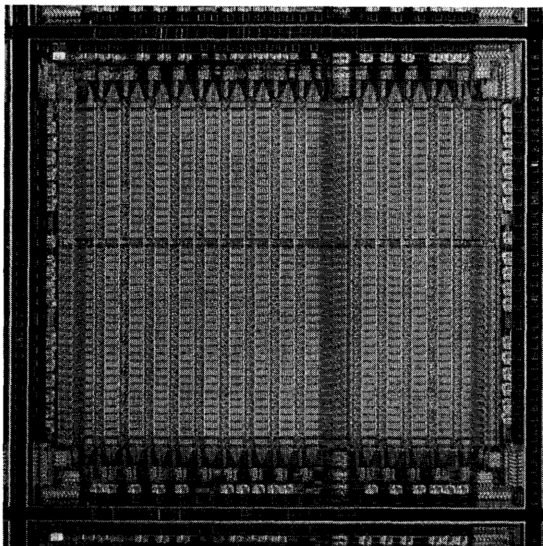


Figure 1. ACT 1020 Die

© 1990 Actel Corporation

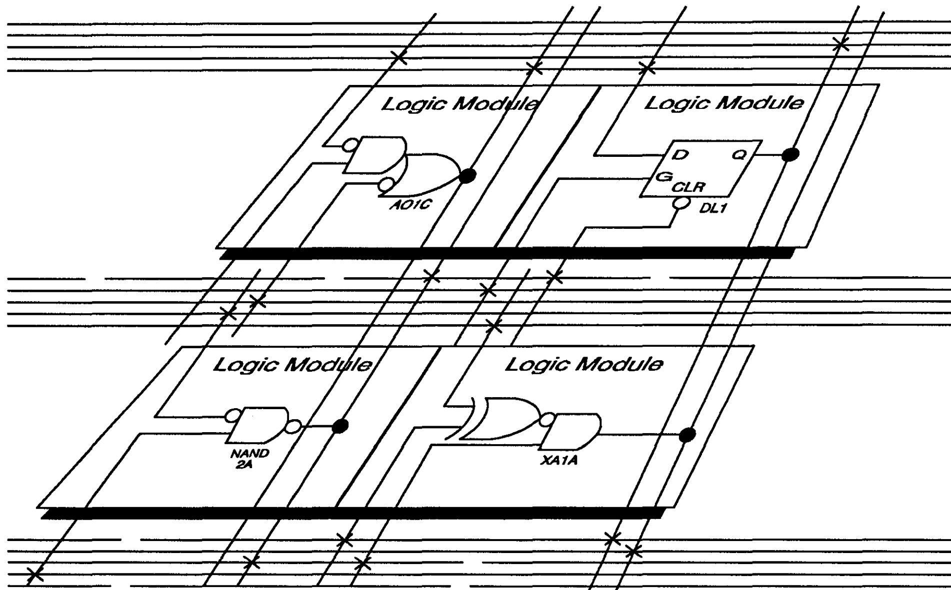


Figure 2. Partial View of an ACT Device

ACT Device Structure

A partial view of an ACT device (Figure 2 above) depicts four logic modules and distributed horizontal and vertical interconnect tracks. PLICE antifuses, located at intersections of the horizontal and vertical tracks, connect logic module inputs and outputs. During programming, these antifuses are addressed and programmed to make the connections required by the circuit application.

The Actel Logic Module

The Actel logic module is an eight-input, one-output configurable logic circuit chosen for the wide range of functions it implements and for its efficient use of interconnect routing resources.

The logic module can implement the four basic logic functions (NAND, AND, OR, and NOR) in gates of two, three, or four inputs. Each function may have many versions, with different combinations of active-low inputs. The logic module can also implement a variety of D-latches, exclusivity function, AND-ORs, and OR-ANDs. No dedicated hardwired latches or flip-flops are required in the array since latches and flip-flops may be constructed from logic modules wherever needed in the application.

I/O Buffers

Each I/O pin is configurable as an input, output, three-state, or bidirectional buffer. Input and output levels are compatible with standard TTL and CMOS specifications. Outputs sink or source 4 mA at TTL levels. See Electrical Specifications for additional I/O buffer specifications.

Device Organization

ACT devices consist of a matrix of logic modules arranged in rows separated by wiring channels (see Figure 1). This array is surrounded by a ring of peripheral circuits including I/O buffers, testability circuits, and diagnostic probe circuits providing real-time diagnostic capability. Between rows of logic modules are routing channels containing sets of segmented metal tracks with PLICE antifuses. Each channel has 22 signal tracks. The resulting network allows arbitrary and flexible interconnections between logic modules and I/O modules.

Probe Pin

ACT 1010/1020 devices have two independent diagnostic probe pins. These pins allow the user to observe any two internal signals by entering the appropriate net name in the diagnostic software. Signals may be viewed on a logic analyzer using Actel's Actionprobe™ diagnostic tools. The probe pins can also be used as user-defined I/Os when debugging is finished.

ACT Array Performance

Temperature and Voltage Effects

Worst-case delays for ACT arrays are calculated in the same manner as for masked array products. A typical delay parameter is multiplied by a derating factor to account for temperature, voltage, and processing effects. However, in an ACT array, temperature and voltage effects are less dramatic than with masked devices.

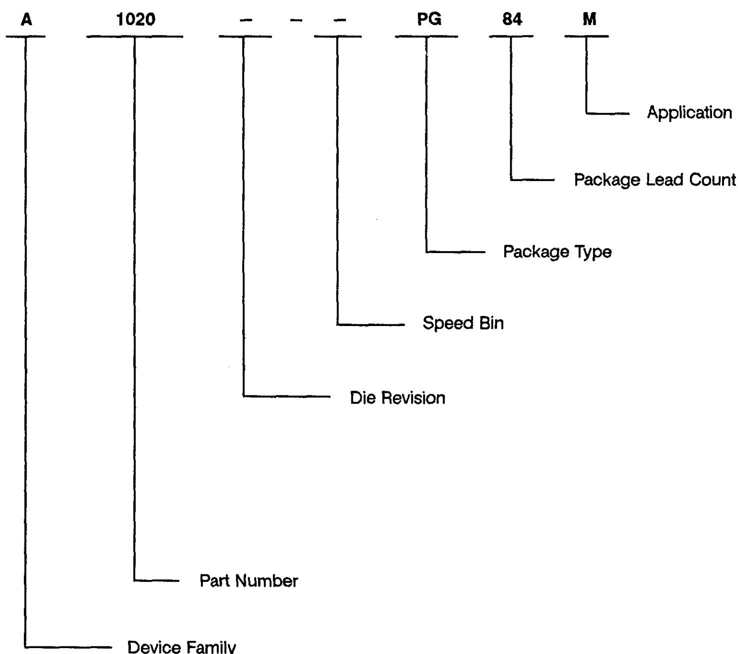
The electrical characteristics of module interconnections on ACT devices remain constant over voltage and temperature fluctuations.

As a result, the total derating factor from typical to worst case for an ACT array is only 1.19 to 1, compared to 2 to 1 for a masked gate array.

Logic Module Size

Logic module size also affects performance. A mask programmed gate array cell with four transistors usually implements only one logic level. In an ACT array's more complex logic module (similar to the complexity of a gate array macro), implementation of multiple logic levels within a single module is possible. This eliminates interlevel wiring and associated RC delays. The effect is termed "net compression."

Ordering Information



Packages

	Type	Lead Count
PL	Plastic Leaded Chip Carrier	44, 68, 84
JQ	J-Leaded Cerquad Chip Carrier	44, 68, 84
PG	Ceramic Pin Grid Array	84

Applications

C	Commercial
I	Industrial
M	Military
B	883 B*

*Contact your Actel representative for current status.

Device Resources

Device	Modules	Gates	User I/Os 84 LCC	68 LCC	44 LCC	84 PGA
1010	295	1200	N/A	57	34	57
1020	546	2000	69	57	34	69

Recommended Operating Conditions

Parameter	Commercial	Industrial	Military	Units
Temperature Range (Note 1)	0 to +70	-40 to +85	-55 to +125	°C
Power Supply Tolerance	±5	±10	±10	$\%V_{CC}$

Note:

1. Ambient temperature (T_A) used for commercial and industrial; case temperature (T_C) used for military.

Absolute Maximum Ratings

Free air temperature range

Symbol	Parameter	Limits	Units
V_{CC}	DC Supply Voltage1	-0.5 to +7.0	Volts
V_I	Input Voltage	-0.5 to V_{CC} +0.5	Volts
V_O	Output Voltage	-0.5 to V_{CC} +0.5	Volts
I_{IK}	Input Clamp Current	±20	mA
I_{OK}	Output Clamp Current	±20	mA
I_{OK}	Continuous Output Current	±25	mA
T_{STG}	Storage Temperature	-65 to +150	°C

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. Exposure to absolute maximum rated conditions for extended periods may affect device reliability. Device should not be operated outside the Recommended Operating Conditions.

Note:

1. $V_{PP} = V_{CC}$, except during device programming.

Power Dissipation

The following formula is used to calculate total device dissipation.

Total Chip Power (mW) = 0.41 N*F1 + 0.17 M*F2 + 1.62 P*F3

Where:

F1 = Average logic module switching rate in MHz.

F2 = Average clock pin switching rate in MHz.

F3 = Average I/O switching rate in MHz.

M = Number of logic modules connected to the clock pin.

N = Total number of logic modules used on the chip. (including M)

P = Number of outputs used loaded with 50 pF.

The second term, variables F2 and M, may be ignored if the CLKBUF macro is not used in the design.

Electrical Specifications

Parameter		Commercial Min.	Commercial Max.	Industrial Min.	Industrial Max.	Military Min.	Military Max.	Units
V_{OH}[1]	(I_{OH} = −4 mA)	3.84						V
	(I_{OH} = −3.2 mA)			3.7		3.7		V
V_{OL}[1]	(I_{OL} = 4 mA)		0.33		0.40		0.40	V
V_{IL}		-0.3	0.8	-0.3	0.8	-0.3	0.8	V
V_{IH}		2.0	V_{CC} + 0.3	2.0	V_{CC} + 0.3	2.0	V_{CC} + 0.3	V
Input Transition Time t_R, t_F[2]			500		500		500	ns
C_{IO} I/O Capacitance[2, 3]			10		10		10	pF
Standby Current, I_{CC}[4]			10		20		25	mA
Leakage Current[5]		-10	10	-10	10	-10	10	µA
I_{OS} Output Short Circuit Current[6]	(V_O = V_{CC})	20	140	20	140	20	140	mA
	(V_O = GND)	-10	-100	-10	-100	-10	-100	mA

Notes:

1. Only one output tested at a time. V_{CC} = min.
2. Not tested, for information only.
3. Includes worst-case 84-pin PLCC package capacitance. V_{OUT} = 0 V, f = 1 MHz.
4. Typical standby current = 3 mA. All outputs unloaded. All inputs = V_{CC} or GND.
5. V_O, V_{IN} = V_{CC} or GND.
6. Only one output tested at a time. Min. at V_{CC} = 4.5 V; Max. at V_{CC} = 5.5 V.

Package Thermal Characteristics

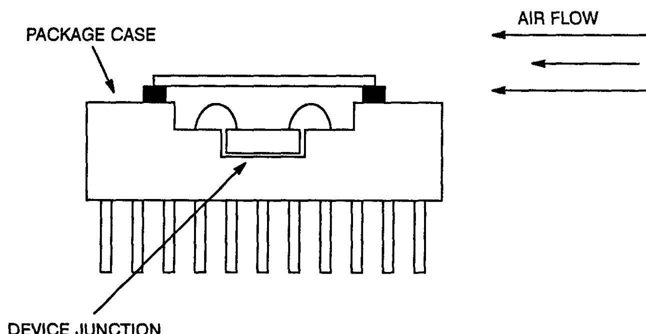
The device junction to case thermal characteristic is θjc, and the junction to ambient air characteristic is θja. The thermal characteristics for θja are shown with two different air flow rates.

Maximum junction temperature is 150°C.

A sample calculation of the maximum power dissipation for a ceramic PG84 package at military temperature is as follows:

150 (Max) - 125 (Max Mil.) = 25/33 C/W (PG84 - still air) = 0.76 Watts.

Package Type	Pin Count	θjc	θja Still air	θja 300 ft/min.	Units
Plastic J-leaded (PL)	44	15	52	40	°C
	68	13	45	35	°C
	84	12	44	33	°C
Cerquad J-leaded (JQ)	44	8	38	30	°C
	68	8	35	25	°C
	84	8	34	24	°C
Ceramic Pin Grid Array (PG)	84	8	33	20	°C



Functional Timing Tests

AC timing for logic module internal delays is determined after place and route. The ALS Timer utility displays actual timing parameters for circuit delays. ACT 1 devices are AC tested to a "binning" circuit specification.

The circuit consists of one input buffer + n logic modules + one output buffer (n = 16 for the ACT 1010; n = 28 for the ACT 1020). The logic modules are distributed along two sides of the device. These modules are configured as inverting and non-inverting buffers. The modules are connected through programmed antifuses with typical capacitive loading.

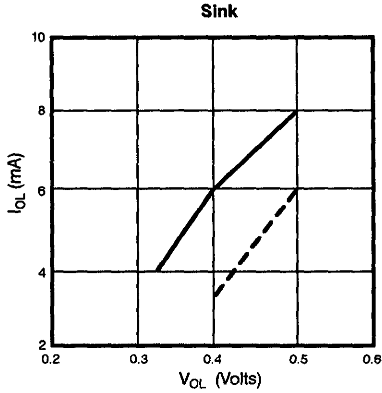
Propagation delay [$t_{PD} = (t_{PLH} + t_{PHL})/2$] is tested to the following AC test specifications.

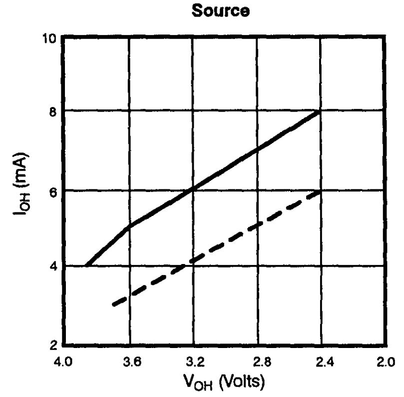
AC Test Specifications

V_{CC} = 5.0 V; T_A = 25°C

Parameter	Min.	Max.	Units
Binning Circuit Delay (t_{PDB})			
1010		87	ns
1020		135	ns

Output Buffer Performance Derating





- - - - Military, worst case values at 125°C, 4.5 V.
——— Commercial, worst case values at 70°C, 4.75 V.

Note:
The above curves are based on characterizations of sample devices and are not completely tested on all devices.

Timing Characteristics

Timing is design-dependent; actual delay values are determined after place and route of the design using the ALS Timer utility. The following delay values use statistical estimates for wiring delays based on 85% to 90% module utilization. Device utilization above 95% will result in performance degradation.

With Actel's place and route program, the user can assign a net's criticality level, based on timing requirements. Delays for both typical and critical (speed-sensitive) nets are given below. Most nets will fall into the "typical" category.

Less than 1% of all routing in a design requires the use of "long tracks." Long tracks, long vertical or horizontal routing paths, are used by the autorouter only as needed. Delays due to the use of long tracks range from 15 ns to 35 ns. Long tracks may be used to route the least-critical nets in a given design.

In the following tables, timing is characterized over the recommended operating conditions, unless specified otherwise.

Module Timing

V_{CC} = 5.0 V; T_A = 25°C; t_{PD} = 4.0 ns @ FO = 0

Single Logic Module Macros (e.g., most gates, latches, multiplexors)[1]

Parameter	Output Net	FO = 1	FO = 2	FO = 3	FO = 4	FO = 8	Units
t_{PD}	Critical	7.0	7.5	8.0	11.0	Note 2	ns
t_{PD}	Typical	8.2	8.7	10.0	11.2	14.0	ns

Dual Logic Module Macros (e.g., adders, wide input gates)[1]

Parameter	Output Net	FO = 1	FO = 2	FO = 3	FO = 4	FO = 8	Units
t_{PD}	Critical	12.0	12.5	13.0	16.0	Note 2	ns
t_{PD}	Typical	13.2	13.7	15.0	16.2	19.0	ns

Sequential Element Timing Characteristics

Parameter		Fan-out					Units
		FO = 1	FO = 2	FO = 3	FO = 4	FO = 8	
t_{SU}	Set Up Time, Data Latches	4.6	5.0	5.4	5.8	5.8	ns
t_{SU}	Set Up Time, Flip-Flops	5.0	5.0	5.0	5.0	5.0	ns
t_H	Hold Time	0	0	0	0	0	ns
t_W	Pulse Width, Minimum[3]	10.0	10.0	10.0	10.0	10.0	ns
t_{PD}	Delay, Critical Net	7.0	7.5	8.0	11.0	Note 2	ns
t_{PD}	Delay, Typical Net	8.2	8.7	10.0	11.2	14.0	ns

Notes:

1. Most flip-flops exhibit single module delays.
2. Critical nets have a maximum fan-out of 6.
3. Minimum pulse width, t_W, applies to CLK, PRE, and CLR inputs.

I/O Buffer Timing

V_{CC} = 5.0 V; T_A = 25°C

INBUF Macros

Parameter	From – To	FO = 1	FO = 2	FO = 3	FO = 4	FO = 8	Units
t_{PHL}	Pad to Y	9.0	9.9	11.6	13.9	18.6	ns
t_{PLH}	Pad to Y	7.7	8.4	10.0	10.9	16.1	ns

CLKBUF (High Fan-out Clock Buffer) Macros

Parameter	FO = 20	FO = 50	FO = 200	Units
t_{PLH}	11	15	25	ns
t_{PHL}	12	16	25	ns

Notes:

1. A clock balancing feature is provided to minimize clock skew.
2. There is no limit to the number of loads that may be connected to the CLKBUF macro.

OUTBUF, TRIBUFF & BIBUF Macros

C_L = 50 pF

Parameter	From – To	CMOS	TTL	Units
t_{PHL}	D to Pad	5.1	6.4	ns
t_{PLH}	D to Pad	9.4	7.4	ns
t_{PHZ}	E to Pad	6.7	4.4	ns
t_{PZH}	E to Pad	8.4	6.3	ns
t_{PLZ}	E to Pad	8.9	6.7	ns
t_{PZL}	E to Pad	6.4	7.7	ns

Change in Propagation Delay with Load Capacitance

Parameter	From – To	CMOS	TTL	Units
t_{PHL}	D to Pad	0.04	0.06	ns/pF
t_{PLH}	D to Pad	0.09	0.05	ns/pF
t_{PHZ}	E to Pad	0.10	0.06	ns/pF
t_{PZH}	E to Pad	0.09	0.05	ns/pF
t_{PLZ}	E to Pad	0.09	0.05	ns/pF
t_{PZL}	E to Pad	0.04	0.05	ns/pF

Notes:

1. The BIBUF macro input section exhibits the same delays as the INBUF macro.
2. Load capacitance delay delta can be extrapolated down to 15 pF minimum.

   Example:

   Delay for OUTBUF driving a 100-pF TTL load:

   $t_{PHL} = 6.4 + (0.06 * (100–50)) = 6.4 + 3.0 = 9.4$ ns

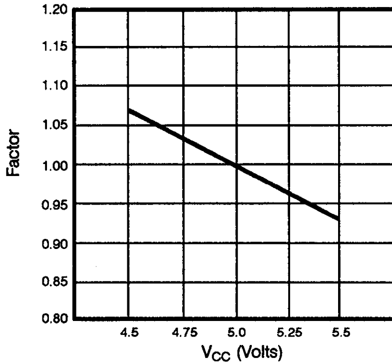
   $t_{PLH} = 7.4 + (0.05 * (100–50)) = 7.4 + 2.5 = 9.9$ ns

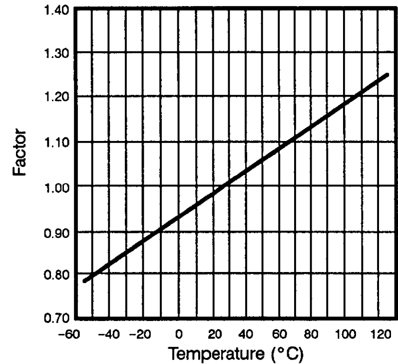
Timing Derating

Operating temperature and voltage and device processing conditions account for variations in array timing characteristics. These variations are summarized into a derating factor for ACT array typical timing specifications. Derating factors are shown below: best-case reflects minimum operating voltage and temperature and best-case processing; worst-case reflects maximum operating voltage and temperature and worst-case processing. Best-case derating is based on sample data only and is not guaranteed.

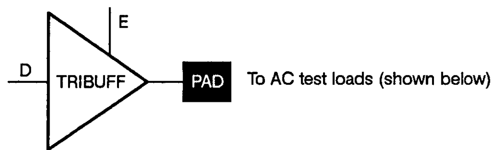
Factor (x typical)

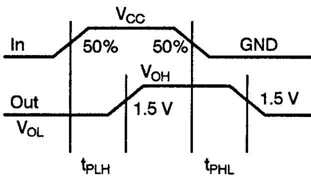
	Commercial	Industrial	Military
Best-case	0.55	0.47	0.40
Worst-case	1.19	1.27	1.38

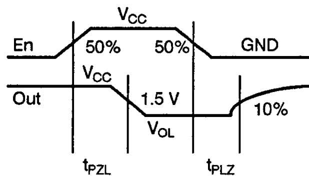


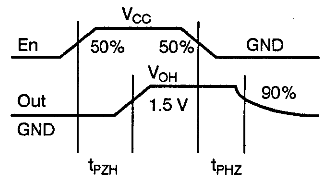


Output Buffer Delays

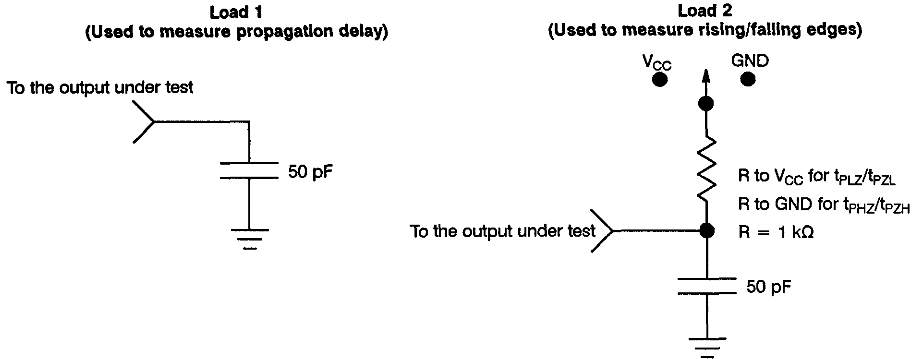




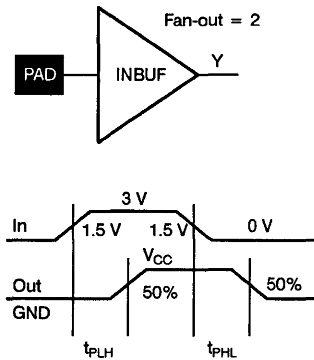




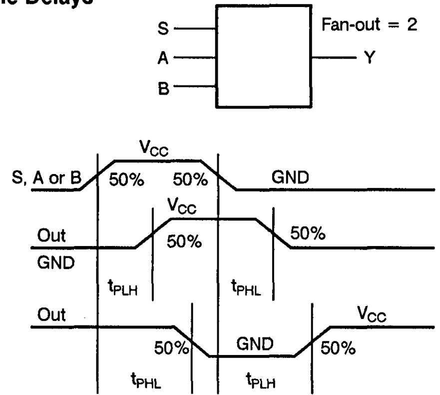
AC Test Loads



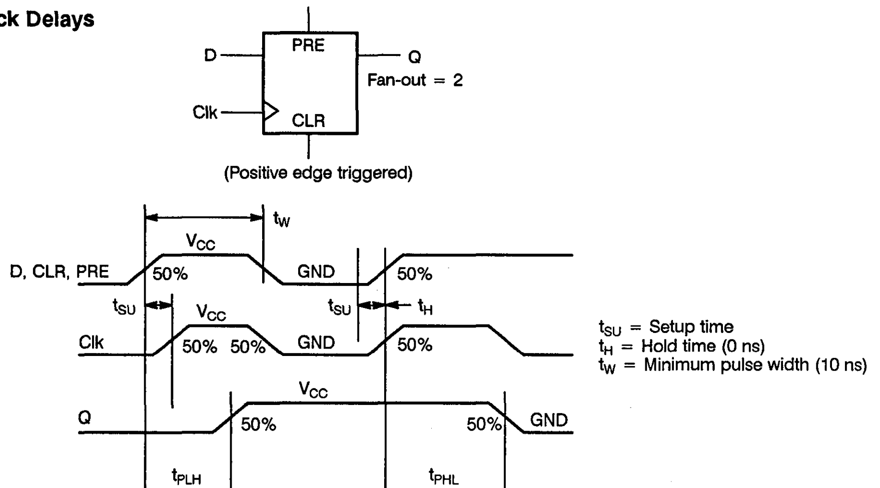
Input Buffer Delays



Module Delays



D-Type Flip-Flop and Clock Delays



t_{SU} = Setup time
t_H = Hold time (0 ns)
t_W = Minimum pulse width (10 ns)

Soft Macro Library Overview

Macro Name	Modules Req'd	Description	Levels of Logic
Counters			
CNT4A	17	4 bit loadable binary counter with clear	4
CNT4B	15	4 bit loadable bin counter w/ clr, active low carry in & carry out	4
UDCNT4A	24	4 bit up/down cntr w/ sync active low load, carry in & carry out	6
Decoders			
DEC2X4	4	2 to 4 decoder	1
DEC2X4A	4	2 to 4 decoder with active low outputs	1
DEC3X8	8	3 to 8 decoder	1
DEC3X8A	8	3 to 8 decoder with active low outputs	1
DEC4X16A	20	4 to 16 decoder with active low outputs	2
DECE2X4	4	2 to 4 decoder with enable	1
DECE2X4A	4	2 to 4 decoder with enable and active low outputs	1
DECE3X8	11	3 to 8 decoder with enable	2
DECE3X8A	11	3 to 8 decoder with enable and active low outputs	2
Latches and Registers			
DLC8A	8	Octal latch with clear	1
DLE8	8	Octal latch with enable	1
DLM8	8	Octal latch with mulitplexed inputs	1
REGE8A	20	Octal register with preset and clear, active high enable	2
REGE8B	20	Octal register w/ active low clock, preset & clear, active high en.	2
Adders			
FA1	3	One bit full adder	3
FADD8	37	8 bit fast adder	4
FADD12	62	12 bit fast adder	5
FADD16	78	16 bit fast adder	5
FADD24	120	24 bit fast adder	6
FADD32	160	32 bit fast adder	7
Comparators			
ICMP4	5	4 bit identity comparator	2
ICMP8	9	8 bit identity comparator	3
MCMP16	93	16 bit magnitude comparator	5
MCMPC2	9	2 bit magnitude comparator with enables	3
MCMPC4	18	4 bit magnitude comparator with enables	4
MCMPC8	36	8 bit magnitude comparator with enables	6
Multiplexors			
MX8	3	8 to 1 multiplexor	2
MX8A	3	8 to 1 multiplexor with an active low output	2
MX16	5	16 to 1 multiplexor	2
Multipliers			
SMULT8	235	8 x 8 two's complement multiplier	Varies

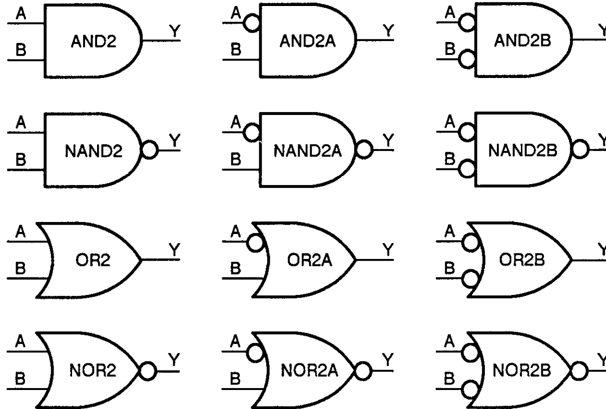
Soft Macro Library Overview (continued)

Macro Name	Modules Req'd	Description	Levels of Logic
Shift Registers			
SREG4A	8	4 bit shift register with clear	2
SREG8A	18	8 bit shift register with clear	2
TTL Replacements			
TA138	12	3 to 8 decoder with 3 enables and active low outputs	2
TA139	4	2 to 4 decoder with an enable and active low outputs	1
TA151	5	8 to 1 multiplexor with enable, true, and complementary outputs	3
TA153	2	4 to 1 multiplexor with active low enable	2
TA157	1	2 to 1 multiplexor with enable	1
TA161	22	4 bit sync counter w/ load, clear, count enables & ripple carry out	3
TA164	18	8 bit serial in, parallel out shift register	1
TA169	25	4 bit synchronous up / down counter	6
TA181	31	4 bit ALU	4
TA194	14	4 bit shift register	1
TA195	10	4 bit shift register	1
TA269	50	8 bit up/down cntr w/ clear, load, ripple carry output & enables	8
TA273	18	Octal register with clear	1
TA280	9	Parity generator and checker	4
TA377	16	Octal register with active low enable	1
UARTs			
UART	189	Universal Asychronous Receiver / Transmitter	7-Tx 4-Rx

Hard Macro Library Overview

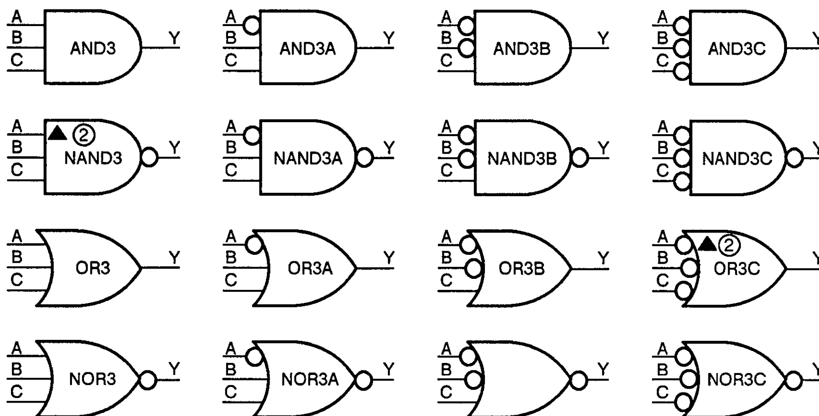
The following illustrations show all the available Hard Macros.

2-Input Gates (Module Count = 1)



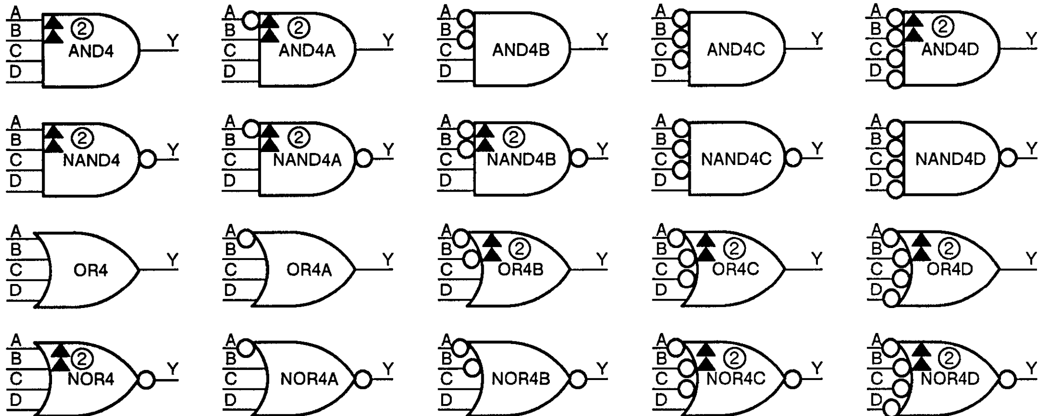
3-Input Gates (Module Count = 1, unless indicated otherwise)

② Indicates 2-module macro
▲ Indicates extra delay input

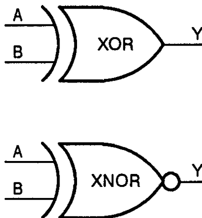


4-Input Gates (Module Count = 1, unless indicated otherwise)

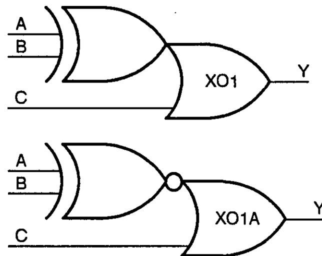
② Indicates 2-module macro
▲ Indicates extra delay input



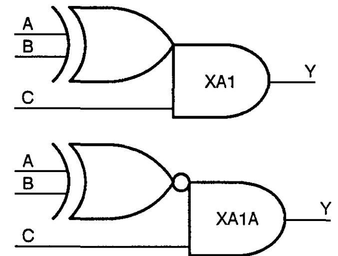
XOR Gates
(Module Count = 1)



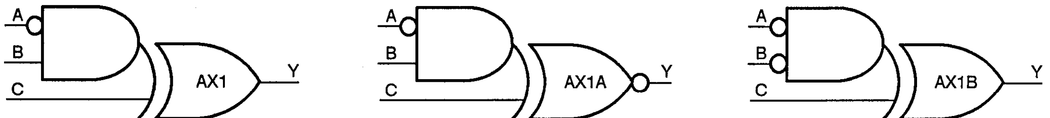
XOR OR Gates
(Module Count = 1)



XOR AND Gates
(Module Count = 1)

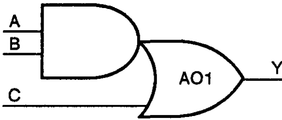


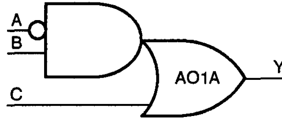
AND XOR Gates
(Module Count = 1)

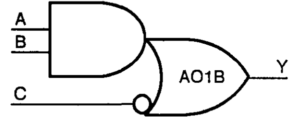


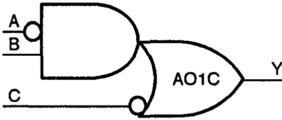
AND OR Gates (Module Count = 1)

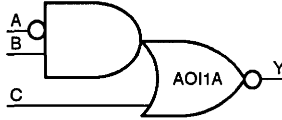
② Indicates 2-module macro
▲ Indicates extra delay input

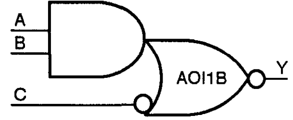


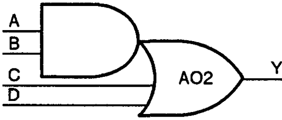


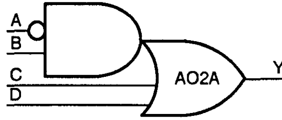


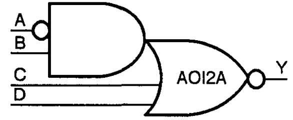


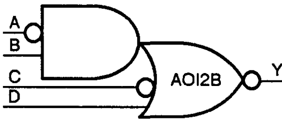


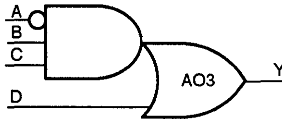


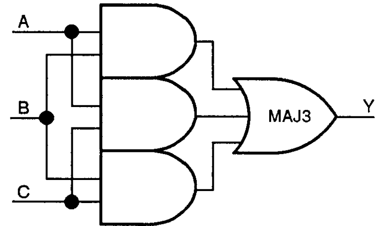


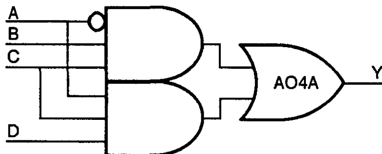


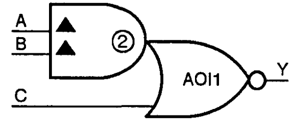


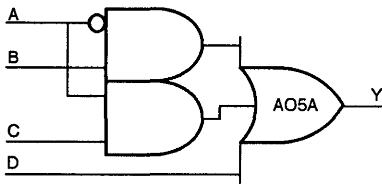


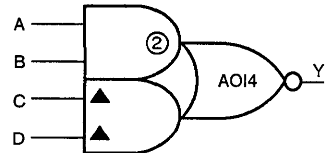


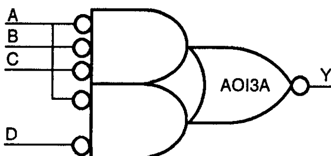






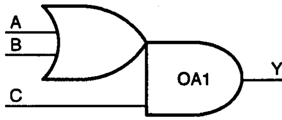


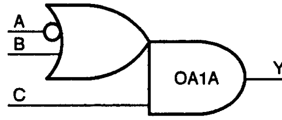


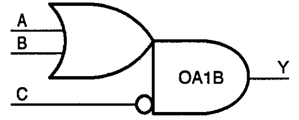


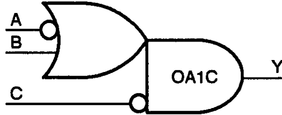
OR AND Gates (Module Count = 1)

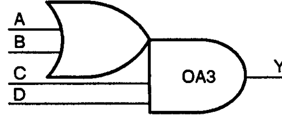
② Indicates 2-module macro
▲ Indicates extra delay input

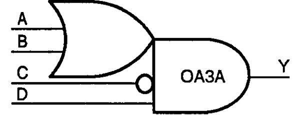


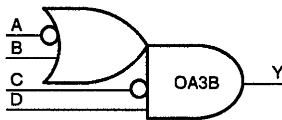


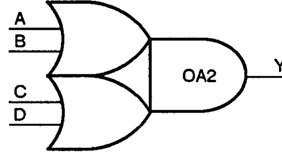


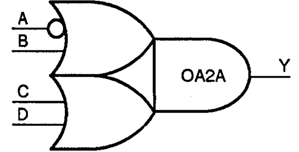


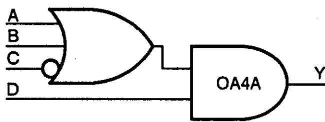


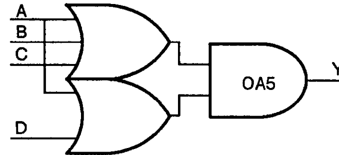


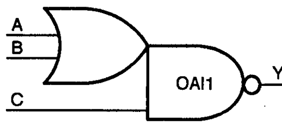


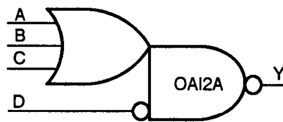


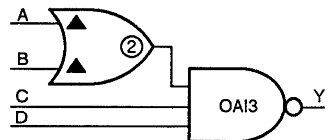




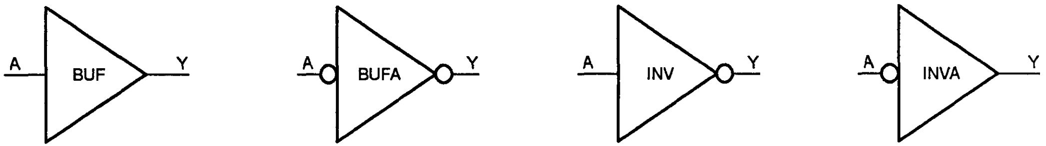




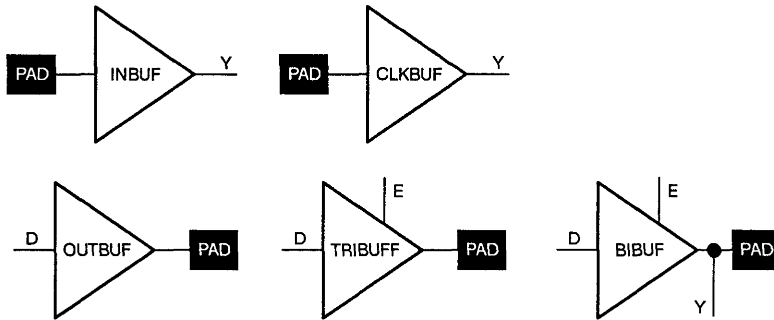




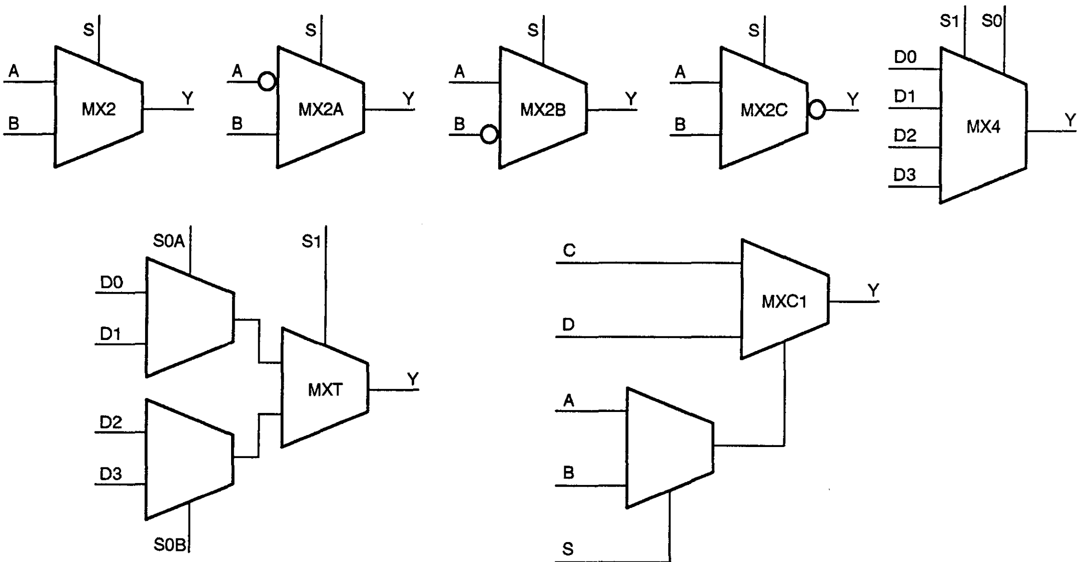
Buffers (Module Count = 1)



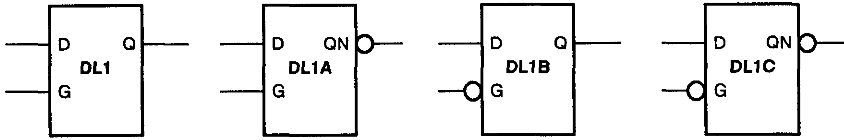
I/O Buffers (I/O Module Count = 1)



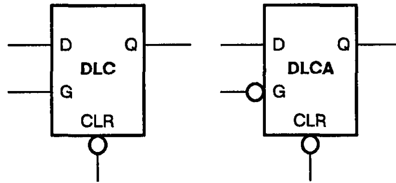
Multiplexors (Module Count = 1)



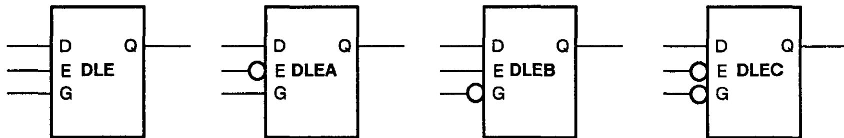
Latches (Module Count = 1)



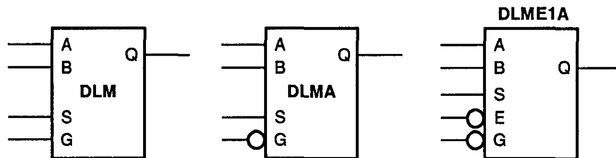
D Latches with Clear (Module Count = 1)



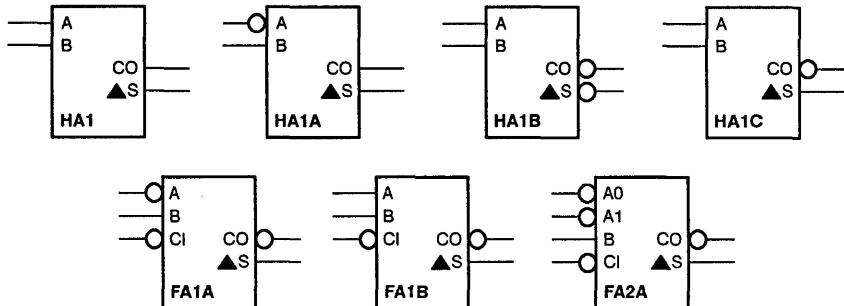
D Latches with Enable (Module Count = 1)



Mux Latches (Module Count = 1)

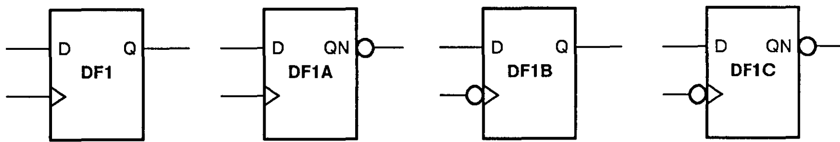


Adders (Module Count = 2)

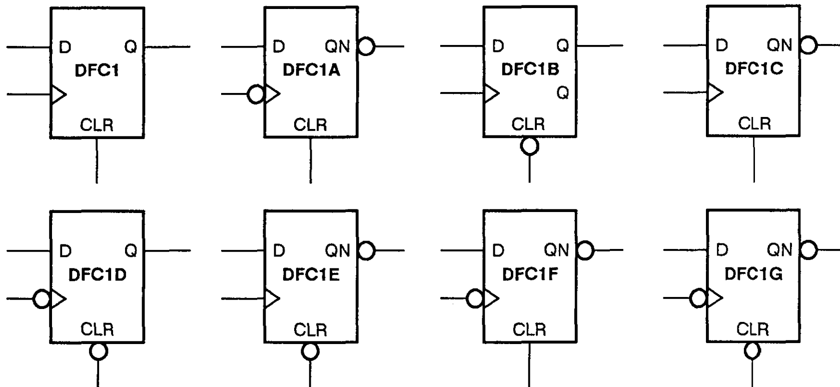


Macros FA1A, FA1B, and FA2A have two level delays from the inputs to the S outputs, as indicated by the ▲

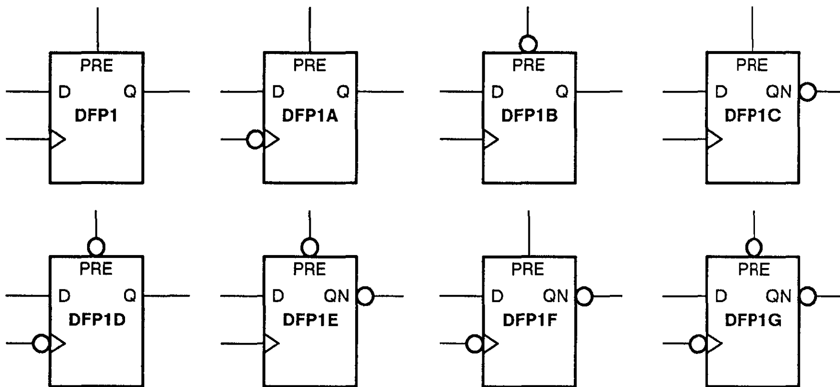
D Flip-Flops (Module Count = 2)



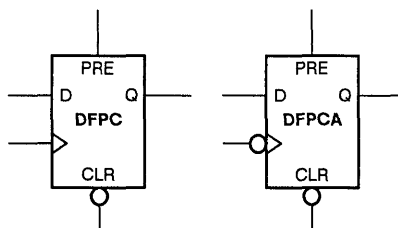
D Flip-Flops with Clear



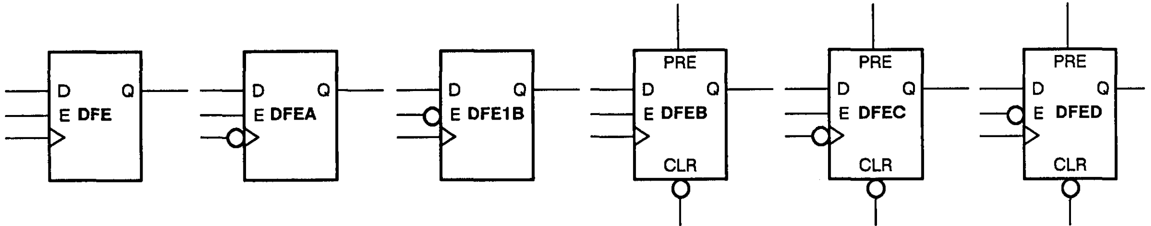
D Flip-Flops with Preset



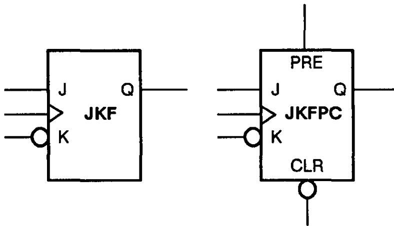
D Flip-Flops with Preset and Clear



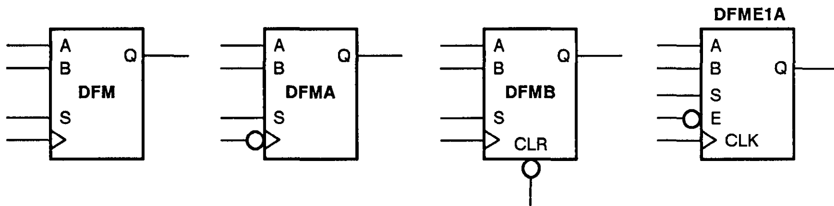
D Flip-Flops with Enable (Module Count = 2)



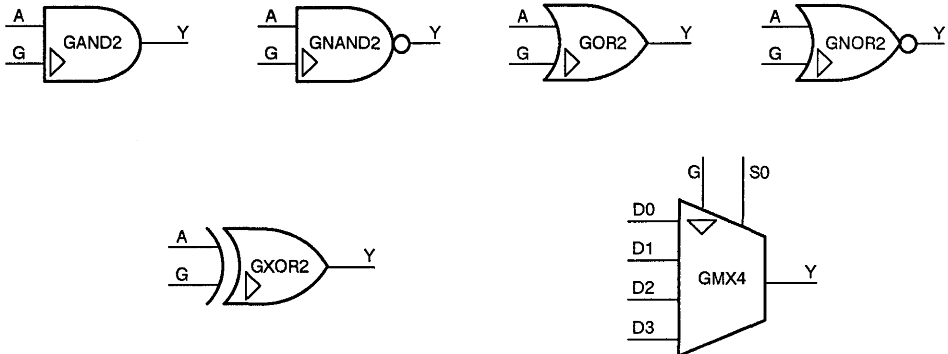
JK Flip-Flops (Module Count = 2)



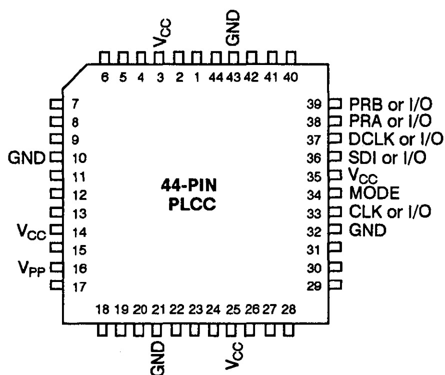
Mux Flip-Flops (Module Count = 2)

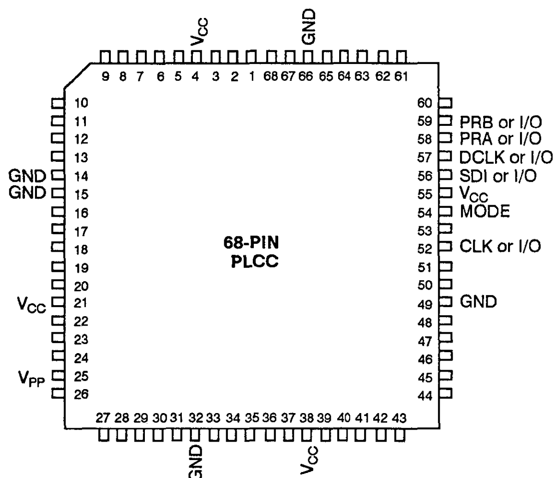


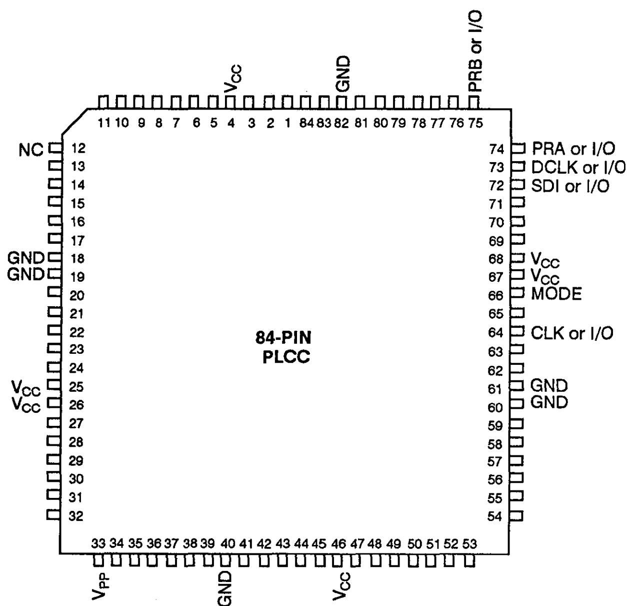
CLKBUF Interface Macros (Module Count = 1)



Packages
(Top View)



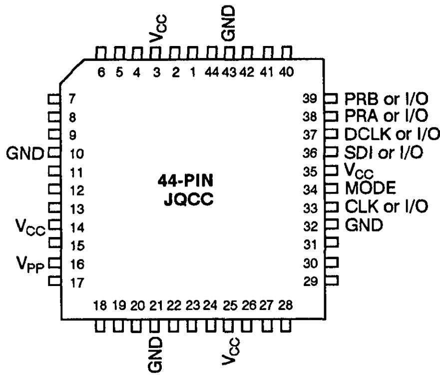


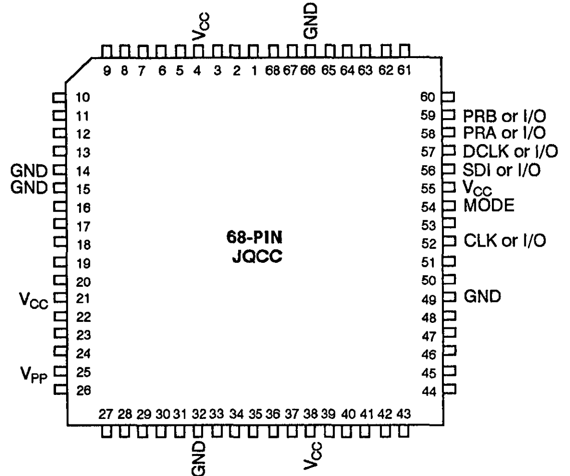


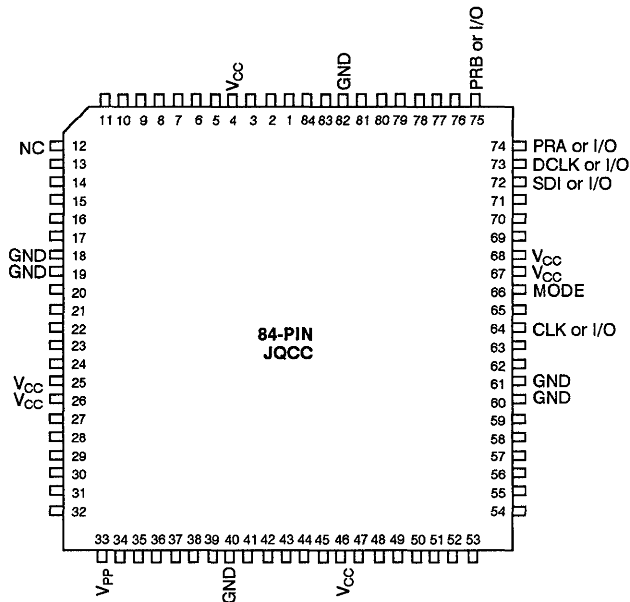
Notes:

1. V_{PP} must be terminated to V_{CC}, except during device programming.
2. MODE must be terminated to circuit ground, except during device programming or debugging.
3. Unused I/O pins are designated as outputs by the Action Logic System and are driven low.
4. All unassigned pins are available for use as I/Os.

Packages (continued)



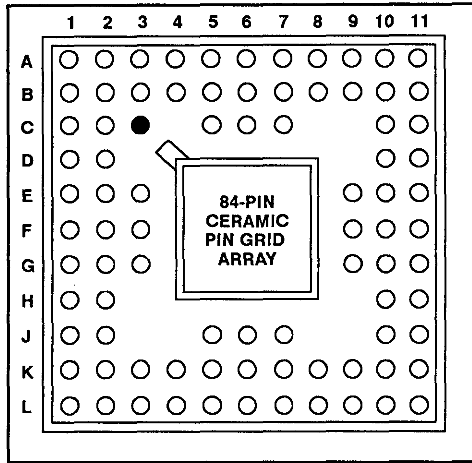




Notes:

1. V_{PP} must be terminated to V_{CC}, except during device programming.
2. MODE must be terminated to circuit ground, except during device programming or debugging.
3. Unused I/O pins are designated as outputs by the Action Logic System and are driven low.
4. All unassigned pins are available for use as I/Os.

Packages (continued)



● Orientation Pin (C3)

Pin Assignments for the 84-Pin PGA

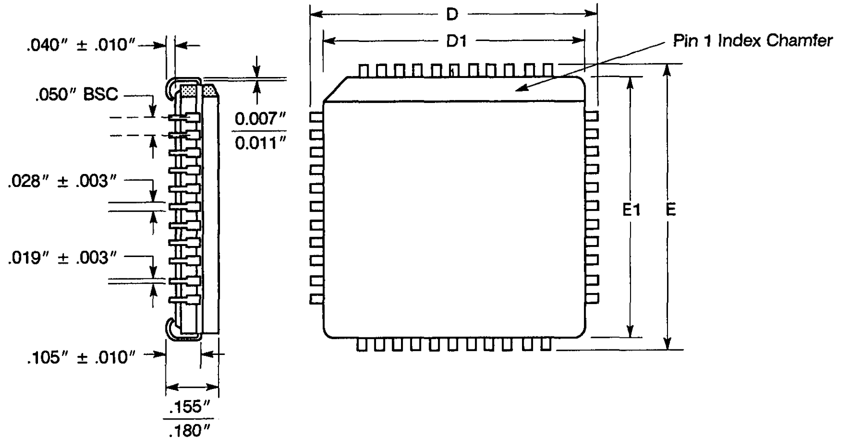
Signal	ACT 1010	ACT 1020
PRA	A11	A11
PRB	B10	B10
MODE	E11	E11
SDI	B11	B11
DCLK	C10	C10
V_{PP}	K2	K2
CLK or I/O	F9	F9
GND	B7, E2, E3, K5, F10, G10	B7, E2, E3, K5, F10, G10
V_{CC}	B5, F1, G2, K7, E9, E10	B5, F1, G2, K7, E9, E10
N/C (No Connection)	B1, B2, C1, C2, K1, J2, L1, J10, K10, K11, C11, D10, D11	B2

Notes:

1. V_{PP} must be terminated to V_{CC}, except during device programming.
2. MODE must be terminated to circuit ground, except during device programming or debugging.
3. Unused I/O pins are designated as outputs by the Action Logic System and are driven low.
4. All unassigned pins are available for use as I/Os.

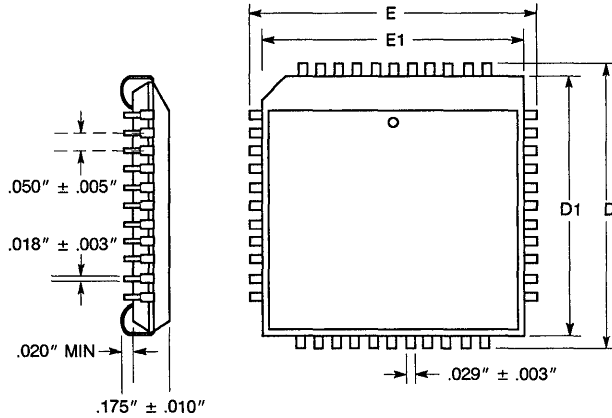
Package Mechanical Details

J-Leaded Cerquad Chip Carriers



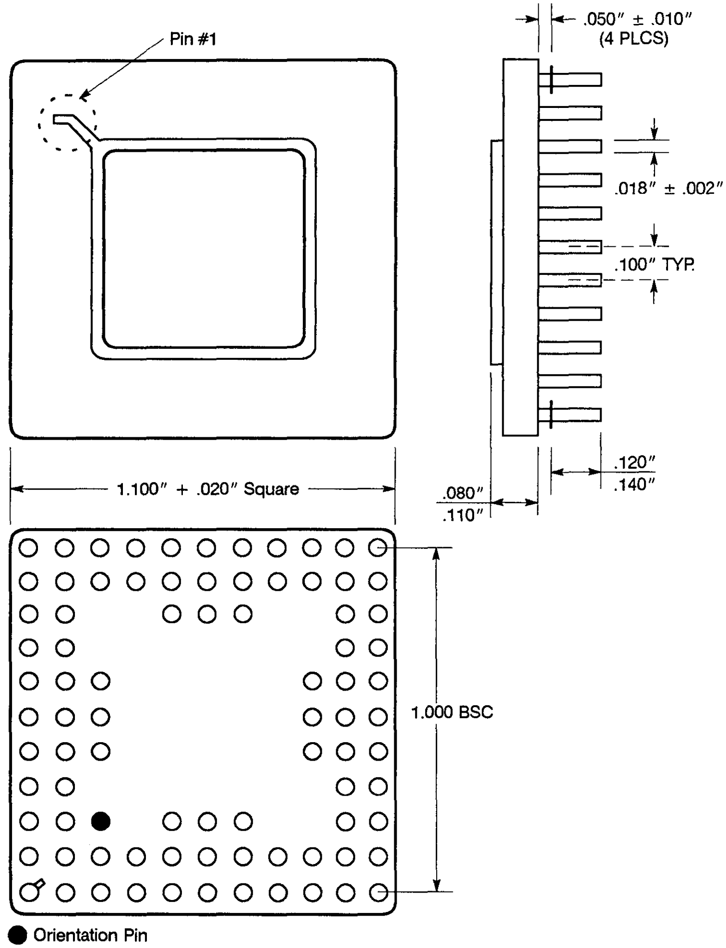
Lead Count	D, E	E1, D1
44	.690″ ± .005″	.650″ ± .008″
68	.990″ ± .005″	.950″ ± .008″
84	1.190″ ± .005″	1.150″ ± .008″

Plastic J-Leaded Chip Carriers



Lead Count	D, E	E1, D1
44	.690″ ± .005″	.655″ ± .005″
68	.990″ ± .005″	.955″ ± .005″
84	1.190″ ± .005″	1.155″ ± .005″

Ceramic Pin Grid Array



ACT™ 1 Military Field Programmable Gate Arrays

Preliminary

Features

- Military Data Sheet Devices
  - Operation Over Full Temperature Range; -55 °C to +125 °C, with ±10% Voltage Supplies
- Military STD 883-C Class B Devices*
  - Assembled to Method 5004
  - Dynamic Burn-In
  - Qualification to Method 5005
  - Group A, B, C, D Test Reports Available
- Packaging
  - 84-Pin Ceramic PGA — Available Now
  - 44-, 68- and 84-Pin J-Leaded Cerquad Chip Carrier — Available Q2, 1990
  - 84-Pin Ceramic Quad Flat Pack — Planned
- Architecture/Technology
  - Antifuse-based architecture offers design flexibility and high gate utilization
  - Architecture and antifuse provide high security for customer designs
  - CMOS process offers low power consumption
  - Complete Test Coverage
- Design Tools
  - Actel Action Logic™ System uses fully automatic place and route algorithms and offers timing analysis to military worst-case specifications
  - Interface to Viewlogic®, Mentor Graphics™, Valid™, and OrCAD™
- High Reliability
  - Nonvolatile, Permanent Programming
  - Antifuse reliability (< 10 FITS) exceeds that of the base CMOS process
  - Very low product failure rate: 91 FITS — 0.091% failures per 1000 hours at 70°C ambient temperature (90°C junction temperature)
  - Radiation tolerant CMOS antifuse process: preliminary total dose tests better than 1.5 megarads (Si)
  - Greater than 2000 V ESD on all pins (Class 2 test method 3015 MIL-STD-883C)

* Compliance to Para 1.21 MIL-STD-883 and MIL-M-38510 pending.

Benefits

Actel's ACT™ 1 field programmable gate array (FPGA) is an ideal device for military applications. Designs are easily implemented and programmed in minutes. This nonvolatile device offers a risk-free solution for almost any type of logic integration problem. The ACT 1 family includes the ACT 1010 device with a capacity of 1,200 gate array equivalent gates or 3,000 PLD/LCA™ gates, and the ACT 1020 device with a capacity of 2,000 gate array gates or 6,000 PLD/LCA gates.

No cost risk — Once you have an Action Logic System, Actel's CAE software and programming package, you can produce as many chips as you like for just the cost of the device itself, no NRE charges to eat up your development budget each time you want to try out a new design.

No time risk — After entering your design, placement and routing is automatic, and programming the device takes only about 5 minutes for an average design. You save time in the design entry process by using tools that are familiar to you. The Action Logic System (ALS) software interfaces to popular CAE software such as Mentor Graphics, Valid, OrCAD, and Viewlogic and runs on popular platforms such as Apollo™, Sun 3™, Sun 4™, and 386 PC compatible machines.

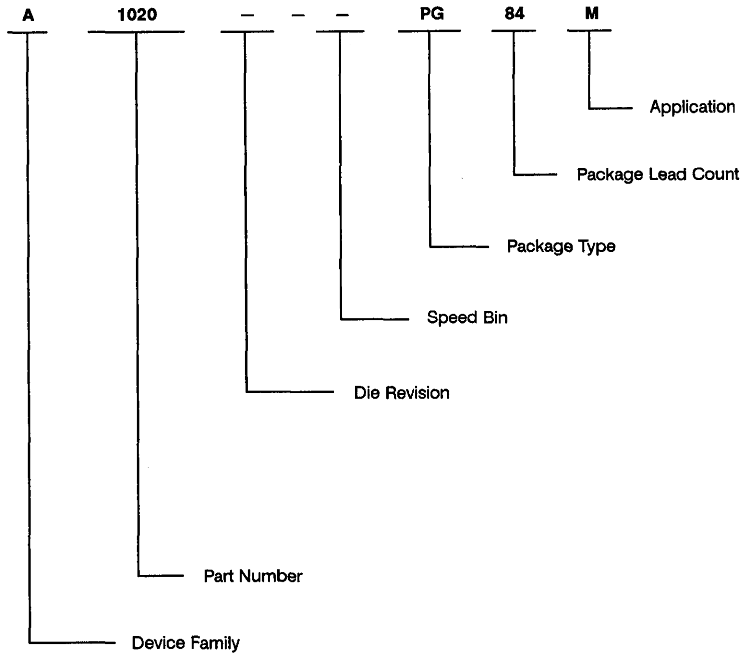
No reliability risk — The PLICE™ antifuse is a one-time programmable, nonvolatile connection. Since Actel devices are permanently programmed, no down-loading from EPROM or SRAM storage is required. Inadvertent erasure is impossible and there is no need to reload the program after power disruptions. Both the PLICE antifuse and the base process are radiation tolerant. Fabrication using a low-power CMOS process means cooler junction temperatures. Actel's non-PLD architecture delivers lower dynamic operating current. Our reliability tests show a very low failure rate of 91 FITS at 90°C junction temperature with no degradation in AC performance. Special stress testing at wafer test eliminates infant mortalities prior to packaging.

No security risk — Reverse engineering of programmed Actel devices from optical or electrical data is extremely difficult. Programmed antifuses cannot be identified from a photograph or by using a SEM. The antifuse map cannot be deciphered either electrically or by microprobing. Each device has a silicon signature that identifies its origins, down to the wafer lot and fabrication facility.

No testing risk — Unprogrammed Actel parts are fully tested at the factory. This includes the logic modules, interconnect tracks, and I/Os. AC performance is assured by special speed path tests and programming circuitry is verified on test antifuses. During the programming process an algorithm is run to assure that all antifuses are correctly programmed. In addition, Actel's Actionprobe™ diagnostic tools allow 100 percent observability of all internal nodes to check and debug your design.

© 1990 Actel Corporation

Ordering Information



Packages

	Type	Lead Count
PG	Ceramic Pin Grid Array	84
JQ*	J-Leaded Cerquad Chip Carrier	44, 68, 84

Applications

M	Military
B*	883 B

* **Important:**
Contact your Actel representative for current availability.

Actel Military Product Flow

Step	Screen	883C—Class B 883C Method	883C—Class B Requirement	Military Datasheet Requirement
1.0	Internal Visual	2010, Test Condition B	100%	100%
2.0	Temperature Cycling	1010, Test Condition C	100%	100%
3.0	Constant Acceleration	2001, Test Condition E (min), Y1, Orientation only	100%	100%
4.0	Seal	1014		
	a. Fine		100%	100%
	b. Gross		100%	100%
5.0	Visual Inspection		100%	100%
6.0	Pre Burn-in Electrical Parameters	In accordance with Actel applicable device specifications	100%	N/A
7.0	Burn-in Test	1015 Condition D 160 hours @ 125°C Min.	100%	N/A
8.0	Interim (post burn-in) Electrical Parameters	In accordance with Actel applicable device specifications	100%	100% (as final test)
9.0	Percent Defective Allowable	5%	All Lots	N/A
10.0	Final Electrical Test	In accordance with Actel applicable device specifications		
	a. Static Tests (1) 25°C (Subgroup 1, Table I, 5005) (2) -55°C and +125°C. (Subgroups 2, 3, Table I, 5005)		100%	100%
	b. Dynamic and Functional Tests (1) 25°C (Subgroup 7, Table I, 5005) (2) -55°C and +125°C. (Subgroups 8A and 8B, Table I, 5005)		100%	100%
	c. Switching Tests at 25°C (Subgroup 9, Table I, 5005)		100%	100%
11.0	Qualification or Quality Conformance Inspection Test Sample Selection (Group A)	5005	All Lots	N/A
12.0	External Visual	2009	100%	Actel specification

Device Resources

Device	Modules	Gates	User I/Os 84 JQCC	68 JQCC	44 JQCC	84 PGA
1010	295	1200	N/A	57	34	57
1020	546	2000	69	57	34	69

Absolute Maximum Ratings

Free air temperature range

Symbol	Parameter	Limits	Units
V_{CC}	DC Supply Voltage[1]	-0.5 to +7.0	Volts
V_I	Input Voltage	-0.5 to V_{CC} +0.5	Volts
V_O	Output Voltage	-0.5 to V_{CC} +0.5	Volts
I_{IK}	Input Clamp Current	±20	mA
I_{OK}	Output Clamp Current	±20	mA
I_{OK}	Continuous Output Current	±25	mA
T_{STG}	Storage Temperature	-65 to +150	°C

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. Exposure to absolute maximum rated conditions for extended periods may affect device reliability. Device should not be operated outside the Recommended Operating Conditions.

Note:

1. $V_{PP} = V_{CC}$, except during device programming.

Recommended Operating Conditions

Parameter	Military	Units
Temperature Range (T_C)	-55 to +125	°C
Power Supply Tolerance	±10	$\%V_{CC}$

Power Dissipation

The following formula is used to calculate total device dissipation.

Total Chip Power (mW) = 0.41 N*F1 + 0.17 M*F2 + 1.62 P*F3

Where:

F1 = Average logic module switching rate in MHz.

F2 = Average clock pin switching rate in MHz.

F3 = Average I/O switching rate in MHz.

M = Number of logic modules connected to the clock pin.

N = Total number of logic modules used on the chip. (including M)

P = Number of outputs used loaded with 50 pF.

The second term, variables F2 and M, may be ignored if the CLKBUF macro is not used in the design.

DC Characteristics

V_{CC} = 5.0 V ± 10%; -55 °C ≤ T_C ≤ +125 °C

Symbol	Parameter	Test Conditions	Group A Subgroups	Limits Min.	Max.	Units
V_{OL}	Output Low Voltage	V_{CC} = 4.5 V, I_{OL} = 4 mA Test one output at a time	1, 2, 3		0.4	V
V_{OH}	Output High Voltage	V_{CC} = 4.5 V, I_{OH} = -3.2 mA Test one output at a time	1, 2, 3	3.7		V
V_{IL}	Input Low Level		1, 2, 3	-0.3	0.8	V
V_{IH}	Input High Level		1, 2, 3	2.0	V_{CC} + .3	V
I_{CC}	Standby Supply Current	V_{CC} = 5.5 V, V_{IN} = V_{CC} or GND Outputs unloaded	1, 2, 3		25	mA
I_{IL}	Input Leakage Current	V_{CC} = 5.5 V, V_{IN} = V_{CC} or GND	1, 2, 3	-10	10	µA
I_{OZ}	Output Leakage Current	V_{CC} = 5.5 V, V_{OUT} = V_{CC} or GND	1, 2, 3	-10	10	µA
I_{OS}	Output Short Circuit Current V_{OUT} = V_{CC}	Test one output at a time	1, 2, 3	20	140	mA
	V_{OUT} = GND	V_{CC} = 4.5 V for min. limit V_{CC} = 5.5 V for max. limit		-10	-100	mA

Switching Characteristics

V_{CC} = 5.0 V ± 10%; -55 °C ≤ T_C ≤ +125 °C

Symbol	Parameter	Test Conditions	Group A Subgroups	Limits Min.	Max.	Units
t_{PDB}	Binning Circuit Delay ACT 1010	V_{CC} = 4.5 V	9, 10, 11		120	ns
	ACT 1020	V_{IH} = 3 V, V_{IL} = 0 V V_{OUT} = 1.5 V			186	ns

Functional and Switching Tests

ACT 1010 and ACT 1020 devices can be tested functionally by using a serial scan test method. Data is shifted into the SDI pin, and the DCLK pin is used as a clock. The data is used to drive the inputs of the internal logic and I/O modules, allowing a complete functional test to be performed. The outputs of the modules can be read by shifting out the output response. All units are tested for functionality over the military temperature range. Group A subgroups 7, 8A, and 8B are tested in the same manner.

AC timing for logic module internal delays and pin-to-pin delays is determined after place and route. The ALS Timer utility displays actual timing parameters for circuit delays. Since these delays are design-dependent and cannot be tested on unprogrammed devices, Actel tests for AC performance by measuring the input-to-output delay of a special path called the "binning circuit."

The binning circuit consists of one input buffer + n logic modules + one output buffer (n = 16 for the ACT 1010; n = 28 for the ACT 1020). The logic modules are distributed along two sides of the device. These modules are configured as inverting and non-inverting buffers. The modules are connected through programmed antifuses with typical capacitive loading.

Actel uses a special benchmark design to correlate the binning circuit delay to typical and worst-case design delays. Samples of units are programmed to this benchmark circuit and all programmed paths are measured for AC performance (including the binning circuit delay). The measured delays are then compared to the ALS Timer predictions. The binning circuit maximum delay has been set to assure conformance to the predicted delays. Units are sampled to confirm this correlation upon initial device characterization and whenever a change is made that may affect AC performance.

Package Thermal Characteristics

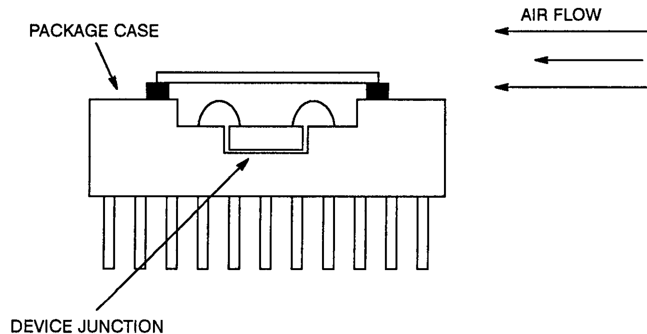
The device junction to case thermal characteristic is θjc, and the junction to ambient air characteristic is θja. The thermal characteristics for θja are shown with two different air flow rates.

Maximum junction temperature is 150°C.

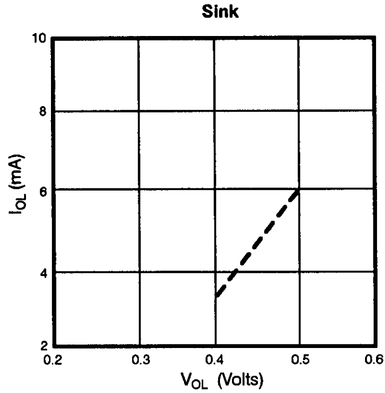
A sample calculation of the maximum power dissipation for a ceramic PG84 package at military temperature is as follows:

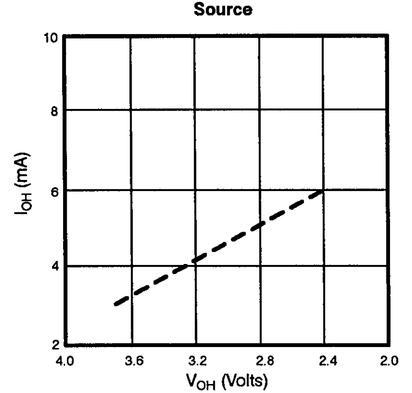
150 (Max) – 125 (Max Mil.) = 25/33 C/W (PG84 – still air) = 0.76 Watts.

Package Type	Pin Count	θjc	θja Still air	θja 300 ft/min.	Units
Ceramic Pin Grid, PG84	84	8	33	20	°C
Cerquad J lead, JQ44	44	8	40	32	°C
Cerquad J lead, JQ68	68	8	38	30	°C
Cerquad J lead, JQ84	84	8	36	25	°C



Output Buffer Performance Derating





Worst-case values at 125°C, 4.5 V.

Note:

The above curves are based on characterizations of sample devices and are not completely tested on all devices.

Timing Characteristics

Timing is design-dependent; actual delay values are determined after place and route of the design using the ALS Timer utility. The following delay values use statistical estimates for wiring delays based on 85% to 90% module utilization. Device utilization above 95% will result in performance degradation.

With Actel's place and route program, the user can assign a net's criticality level, based on timing requirements. Delays for both typical and critical (speed-sensitive) nets are given below. Most nets will fall into the "typical" category.

Less than 1% of all routing in a design requires the use of "long tracks." Long tracks, long vertical or horizontal routing paths, are used by the autorouter only as needed. Delays due to the use of long tracks range from 15 ns to 35 ns. Long tracks may be used to route the least-critical nets in a given design.

In the following tables, timing is characterized over the recommended operating conditions, unless specified otherwise.

Module Timing

V_{CC} = 5.0 V; T_A = 25 °C; t_{PD} = 4.0 ns @ FO = 0

Single Logic Module Macros (e.g., most gates, latches, multiplexors)[1]

Parameter	Output Net	FO = 1	FO = 2	FO = 3	FO = 4	FO = 8	Units
t_{PD}	Critical	7.0	7.5	8.0	11.0	Note 2	ns
t_{PD}	Typical	8.2	8.7	10.0	11.2	14.0	ns

Dual Logic Module Macros (e.g., adders, wide input gates)[1]

Parameter	Output Net	FO = 1	FO = 2	FO = 3	FO = 4	FO = 8	Units
t_{PD}	Critical	12.0	12.5	13.0	16.0	Note 2	ns
t_{PD}	Typical	13.2	13.7	15.0	16.2	19.0	ns

Sequential Element Timing Characteristics

Parameter		Fan-out					Units
		FO = 1	FO = 2	FO = 3	FO = 4	FO = 8	
t_{SU}	Set Up Time, Data Latches	4.6	5.0	5.4	5.8	5.8	ns
t_{SU}	Set Up Time, Flip-Flops	5.0	5.0	5.0	5.0	5.0	ns
t_H	Hold Time	0	0	0	0	0	ns
t_W	Pulse Width, Minimum	10.0	10.0	10.0	10.0	10.0	ns
t_{PD}	Delay, Critical Net	7.0	7.5	8.0	11.0	Note 2	ns
t_{PD}	Delay, Typical Net	8.2	8.7	10.0	11.2	14.0	ns

Notes:

1. Most flip-flops exhibit single module delays.
2. Critical nets have a maximum fan-out of 6.
3. Minimum pulse width, t_W, applies to CLK, PRE, and CLR inputs.

I/O Buffer Timing

V_{CC} = 5.0 V; T_A = 25 °C

INBUF Macros

Parameter	From – To	FO = 1	FO = 2	FO = 3	FO = 4	FO = 8	Units
t_{PHL}	Pad to Y	9.0	9.9	11.6	13.9	18.6	ns
t_{PLH}	Pad to Y	7.7	8.4	10.0	10.9	16.1	ns

CLKBUF (High Fan-out Clock Buffer) Macros

Parameter	FO = 20	FO = 50	FO = 200	Units
t_{PLH}	11	15	25	ns
t_{PHL}	12	16	25	ns

Notes:

1. A clock balancing feature is provided to minimize clock skew.
2. There is no limit to the number of loads that may be connected to the CLKBUF macro.

OUTBUF, TRIBUFF & BIBUF Macros

C_L = 50 pF

Parameter	From – To	CMOS	TTL	Units
t_{PHL}	D to Pad	5.1	6.4	ns
t_{PLH}	D to Pad	9.4	7.4	ns
t_{PHZ}	E to Pad	6.7	4.4	ns
t_{PZH}	E to Pad	8.4	6.3	ns
t_{PLZ}	E to Pad	8.9	6.7	ns
t_{PZL}	E to Pad	6.4	7.7	ns

Change in Propagation Delay with Load Capacitance

Parameter	From – To	CMOS	TTL	Units
t_{PHL}	D to Pad	0.04	0.06	ns/pF
t_{PLH}	D to Pad	0.09	0.05	ns/pF
t_{PHZ}	E to Pad	0.10	0.06	ns/pF
t_{PZH}	E to Pad	0.09	0.05	ns/pF
t_{PLZ}	E to Pad	0.09	0.05	ns/pF
t_{PZL}	E to Pad	0.04	0.05	ns/pF

Notes:

1. The BIBUF macro input section exhibits the same delays as the INBUF macro.
2. Load capacitance delay delta can be extrapolated down to 15 pF minimum.

   Example:

   Delay for OUTBUF driving a 100-pF TTL load:

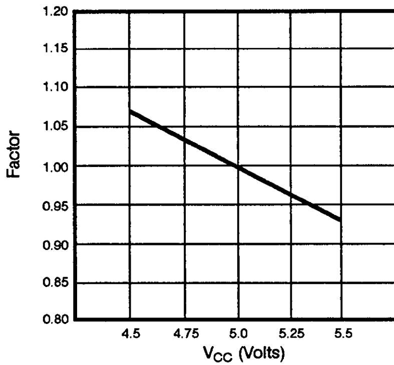
   t_{PHL} = 6.4 + (0.06 * (100–50)) = 6.4 + 3.0 = 9.4 ns

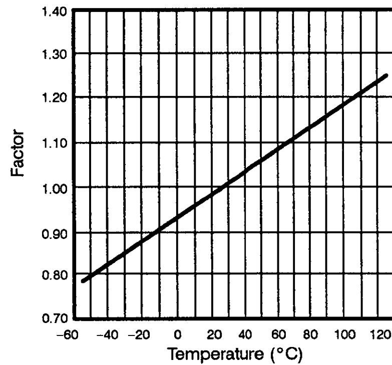
   t_{PLH} = 7.4 + (0.05 * (100–50)) = 7.4 + 2.5 = 9.9 ns

Timing Derating

Operating temperature and voltage and device processing conditions account for variations in array timing characteristics. These variations are summarized into a derating factor for ACT array typical timing specifications. Derating factors for military devices are shown below: best-case reflects minimum operating voltage and temperature and best-case processing; worst-case reflects maximum operating voltage and temperature and worst-case processing. Best-case derating is based on sample data only and is not guaranteed.

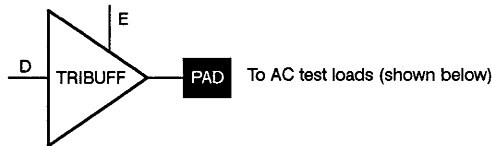
	Factor (x typical)
Best-case	0.40
Worst-case	1.38

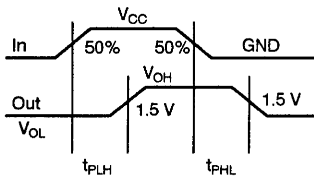


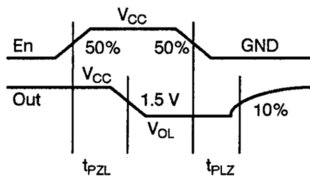


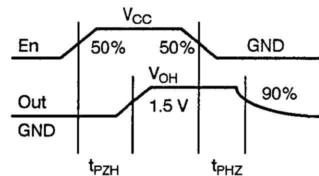
1

Output Buffer Delays

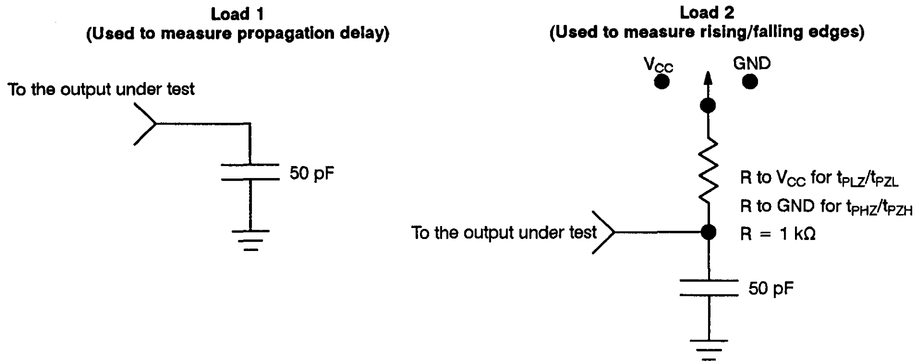




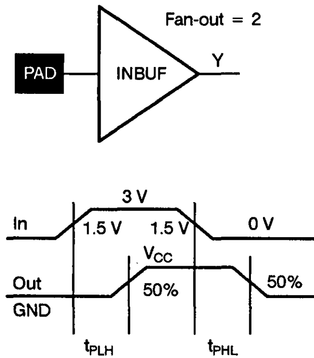




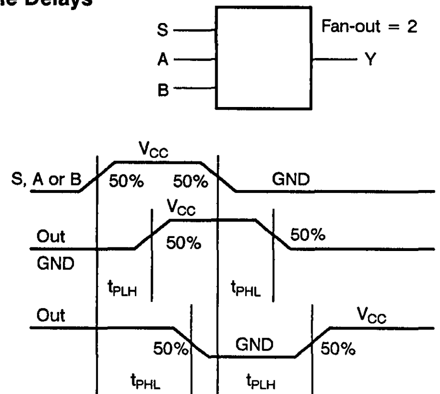
AC Test Loads



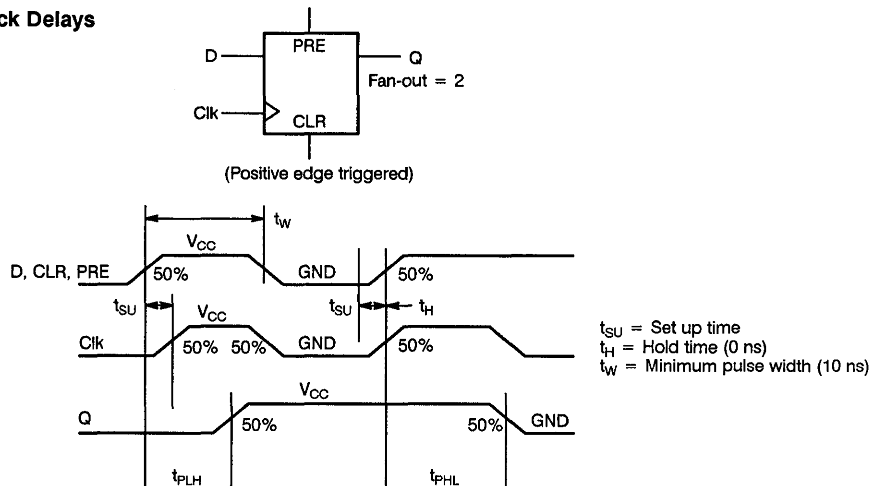
Input Buffer Delays



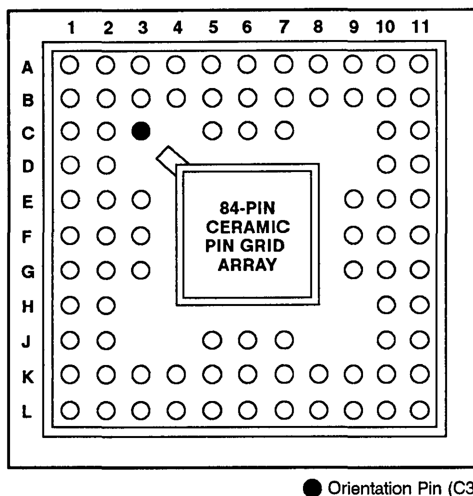
Module Delays



D-Type Flip-Flop and Clock Delays



Packages
(Top View)



● Orientation Pin (C3)

1

Pin Assignments for the 84-Pin PGA

Signal	ACT 1010	ACT 1020
PRA	A11	A11
PRB	B10	B10
MODE	E11	E11
SDI	B11	B11
DCLK	C10	C10
V_{PP}	K2	K2
CLK or I/O	F9	F9
GND	B7, E2, E3, K5, F10, G10	B7, E2, E3, K5, F10, G10
V_{CC}	B5, F1, G2, K7, E9, E10	B5, F1, G2, K7, E9, E10
N/C (No Connection)	B1, B2, C1, C2, K1, J2, L1, J10, K10, K11, C11, D10, D11	B2

Notes:

1. V_{PP} must be terminated to V_{CC}, except during device programming.
2. MODE must be terminated to circuit ground, except during device programming or debugging.
3. Unused I/O pins are designated as outputs by the Action Logic System and are driven low.
4. All unassigned pins are available for use as I/Os.

Packages (continued)

44-PIN JQCC

V_{CC} GND

6 5 4 3 2 1 44 43 42 41 40

7
8
9
GND 10
11
12
13
V_{CC} 14
15
V_{PP} 16
17

39 PRB or I/O
38 PRA or I/O
37 DCLK or I/O
36 SDI or I/O
35 V_{CC}
34 MODE
33 CLK or I/O
32 GND
31
30
29

18 19 20 21 22 23 24 25 26 27 28

GND V_{CC}

68-PIN JQCC

V_{CC} GND

9 8 7 6 5 4 3 2 1 68 67 66 65 64 63 62 61

10
11
12
13
GND 14
GND 15
16
17
18
19
20
V_{CC} 21
22
23
24
V_{PP} 25
26

60
59 PRB or I/O
58 PRA or I/O
57 DCLK or I/O
56 SDI or I/O
55 V_{CC}
54 MODE
53
52 CLK or I/O
51
50
49 GND
48
47
46
45
44

27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43

GND V_{CC}

84-PIN JQCC

V_{CC} GND PRB or I/O

11 10 9 8 7 6 5 4 3 2 1 84 83 82 81 80 79 78 77 76 75

NC 12
13
14
15
16
17
GND 18
GND 19
20
21
22
23
24
V_{CC} 25
V_{CC} 26
27
28
29
30
31
32

74 PRA or I/O
73 DCLK or I/O
72 SDI or I/O
71
70
69
68 V_{CC}
67 V_{CC}
66 MODE
65
64 CLK or I/O
63
62
61 GND
60 GND
59
58
57
56
55
54

33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53

V_{PP} GND V_{CC}

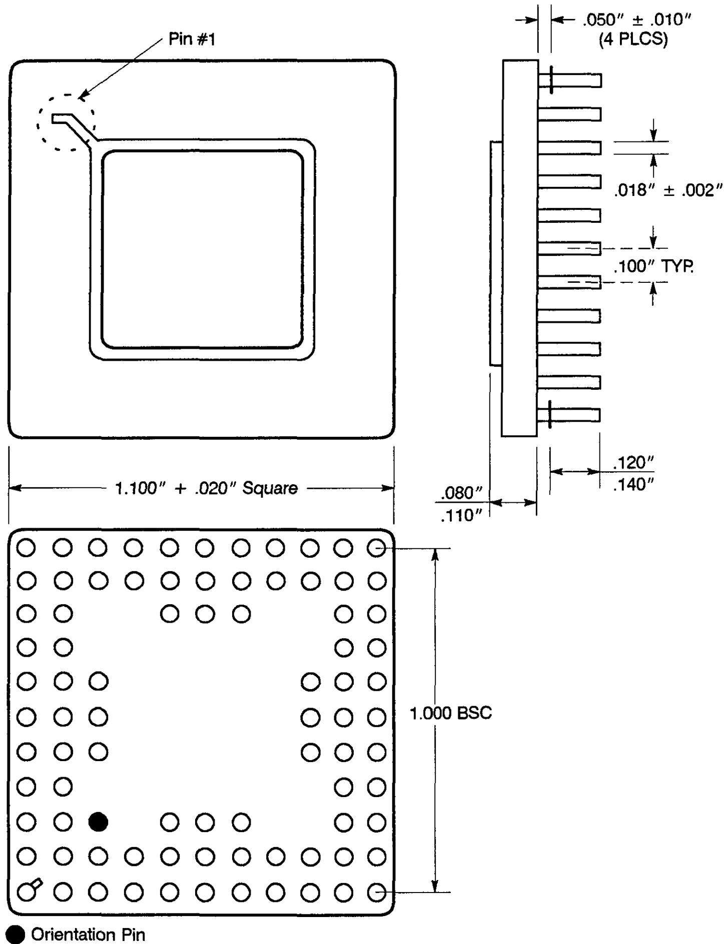
Notes:

1. V_{PP} must be terminated to V_{CC}, except during device programming.
2. MODE must be terminated to circuit ground, except during device programming or debugging.
3. Unused I/O pins are designated as outputs by the Action Logic System and are driven low.
4. All unassigned pins are available for use as I/Os.

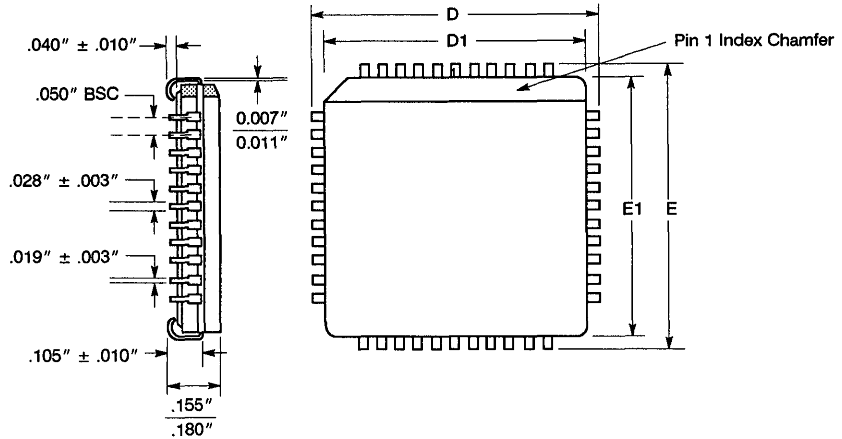
Important:

Please contact your Actel representative for current package availability.

Package Mechanical Details
Ceramic Pin Grid Array



Package Mechanical Details (continued)
J-Leaded Cerquad Chip Carriers



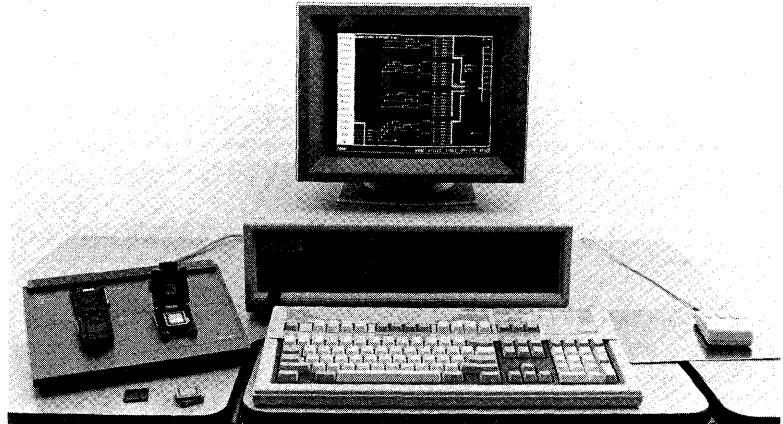
Lead Count	D, E	E1, D1
44	.690″ ± .005″	.650″ ± .008″
68	.990″ ± .005″	.950″ ± .008″
84	1.190″ ± .005″	1.150″ ± .008″

Important:
Please contact your Actel representative for current package availability.

Action Logic™ System FPGA Design Environment

Features

- Extensive Gate Array Macro Library
- Supports Popular CAE Systems
- Design Validation, Electrical Rules Check
- Fully Automatic Place and Route
- Menu-Driven Pin Editor for I/O Assignment
- Timing Analyzer Inspects All Critical Paths and AC Specifications



Overview

The Action Logic™ System is a high-productivity CAE environment for implementing logic functions with ACT™ 1 field programmable gate array devices. A menu-driven interface allows users to complete ACT 1 designs, from concept to silicon, within hours without costly NRE. The system includes a software design environment and an Activator™ programmer, tester, and debugger. Most popular CAE platforms are supported. Designers use their workstation to capture schematics, simulate, verify, place and route, perform timing analysis, program, and debug the chip. Figures 1 through 4 show the Actel interface to popular CAE design environments. Actel's on-line help screens and reference manuals speed and simplify the design process.

Gate Array Macro Library

The Actel macro library contains the logic function building blocks necessary to create a design. The library includes macros ranging in complexity from simple gates to complex functions. Each macro has a graphic symbol or icon. Hard macros also contain placement information, a netlist, and timing data.

Basic gates from the library are used to create soft macros, such as counters, adders, and decoders. The Actel library contains over 200 different macros.

Schematic Capture and Simulation

Users enter their schematic using the Actel library. When schematic capture is completed, functional simulation is performed on the design.

Pin Editor

The Actel Pin Editor is an easy-to-use, menu-driven program for user assignment of logic I/Os and package pins. The editor automatically scrolls a list of all user-designated I/Os. To assign any pin to an I/O, the user enters the desired pin number next to the I/O node name. During each pin assignment, the Editor insures each pin is a valid package pin and is not already assigned to another chip I/O.

Design Validation

The Actel Design Validator examines an ACT design for adherence to design rules specific to the ACT device chosen for the design. Validation verifies routability and performs design rules checks prior to routing. The Validator produces error and information messages and system warnings regarding electrical rules violations such as excessive fanout, shorted outputs, or unconnected inputs. A warning is issued, for example, if a net exceeds a fanout of ten; unconnected module inputs produce an error message.

© 1990 Actel Corporation

The Validator also provides statistical information about routability, logic module count, I/O count, average fanout per net, and array utilization. After passing through the validator, the design is free of electrical rules violations.

Automatic Place and Route

Fully automatic place and route software minimizes design delay by assigning macros to optimal locations in the chip. The system uses the design's netlist, critical net information, and I/O assignments to automatically place and route all the logic blocks within the circuit. No manual intervention is required, even at high device utilization.

The software provides the user with data on actual wire lengths, capacitive loading, and wiring congestion. The route program assigns the shortest possible net segments to connect the library macros with minimal delay, routing 100% of the nets automatically for 85% to 95% logic module utilization.

Timing Analysis

The timing analyzer (Timer) is an interactive tool that determines and highlights all critical and non-critical paths within a specified design. After the layout phase is completed, the Timer extracts accurate net delays. These delays are back-annotated to the netlist and evaluated with the timer or a CAE simulator. Using this information, the designer can optimize the design to meet timing specifications. The Timer accepts as input the design's netlist and delay information. The user directs the type of analysis and output. Delay reports generated by the timing analyzer using post-route numbers provide the final AC specifications for the design.

Device Programming and Functional Test

Programming is controlled by the Activator programming system. Action Logic software generates a fuse map for the ACT device, which is used by the Activator for programming. A series of internal address registers are automatically loaded; they specify the programming element (PLICE™ antifuse) to be programmed. A programming sequence is then initiated, creating a permanent link. These steps are repeated until all interconnections are made.

The Activator supports functional testing using an I/O test vector file. It also supports interactive circuit debugging.

In-Circuit Test and Debug

Once the device is programmed and functionally verified, it is ready for operation in the user's system. Actionprobe™ diagnostic tools may then be used to further evaluate circuit integrity.

Ordering Information

Part Number	Platform/ CAE Host*	Configuration: Viewlogic Viewdraw	Configuration: Place & Route, Timer	Configuration: Debugger, Actionprobes	Configuration: Activator
ALS-113	386 PC/ OrCAD™ Viewlogic®	X	X		X
ALS-110	386 PC/ OrCAD Viewlogic	X	X	X	X
ALS-031	Apollo™/ Mentor Graphics™		X	X	X
ALS-035	Apollo/ Mentor Graphics		X		
ALS-041-S3	Sun 3™/ Valid™		X	X	X
ALS-045-S3	Sun 3/ Valid		X		
ALS-041-S4	Sun 4™/ Valid		X	X	X
ALS-045-S4	Sun 4/ Valid		X		
ALS-051-S3	Sun 3/ Viewlogic		X	X	X
ALS-055-S3	Sun 3/ Viewlogic		X		
ALS-051-S4	Sun 4/ Viewlogic		X	X	X
ALS-055-S4	Sun 4/ Viewlogic		X		

OPTIONS	
ALS-016	High-Density Viewsim® for 386 PC
ALS-017	Low-Density (up to 650 logic modules) Viewsim for 386 PC
ALS-114	Boolean Entry and Optimization for 386 PC
ALS-119	Activator 1 Programmer
ALS-219, 239, 249-S3, 249-S4	Activator 2 Programmer
ALS-256-S3	SUN 3-based Workview® CAE
ALS-256-S4	SUN 4-based Workview CAE

*Includes schematic, symbol, and functional simulation libraries

Important: All systems and options are subject to availability.
Please contact your Actel representative for current status.

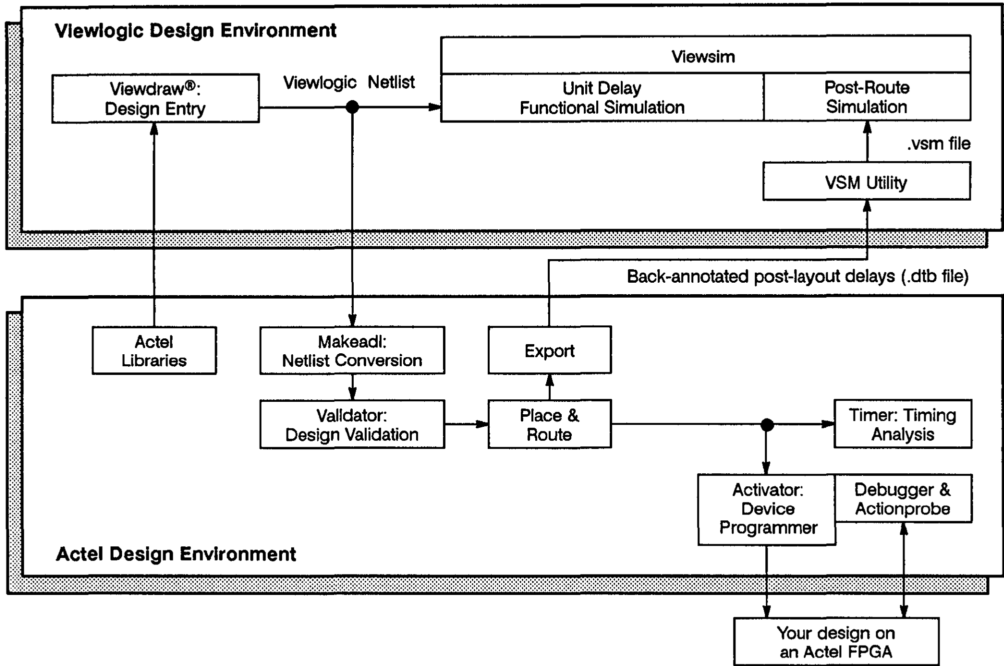


Figure 1. Actel - Viewlogic Design Flow

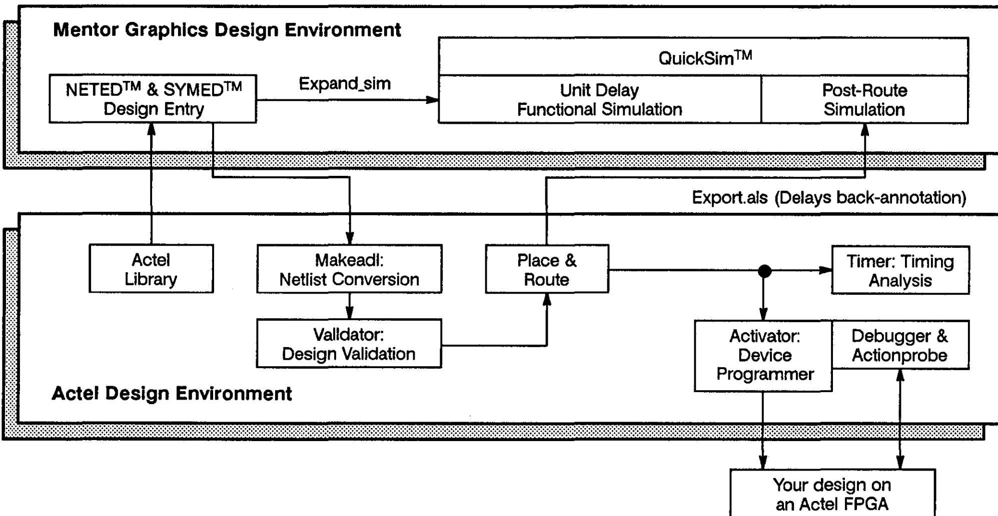


Figure 2. Actel - Mentor Graphics Design Flow

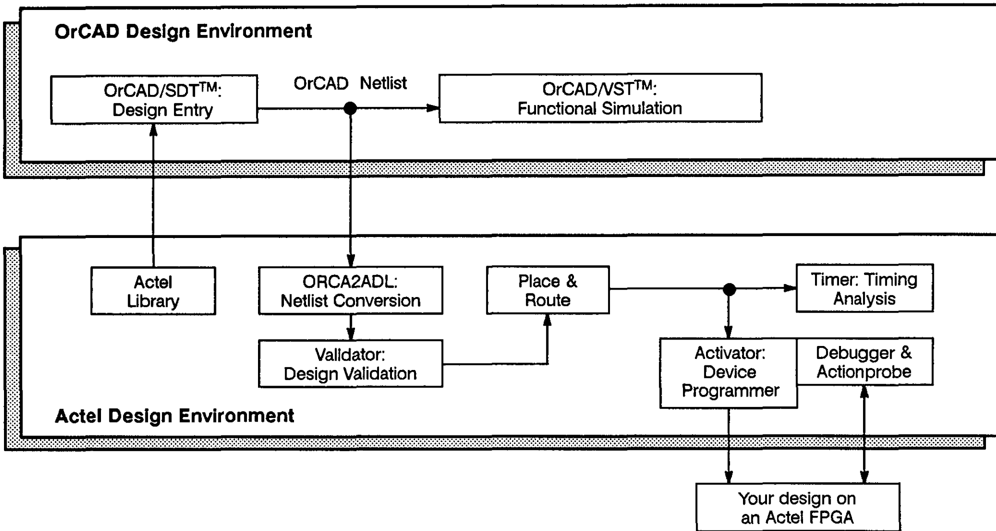


Figure 3. Actel - OrCAD Design Flow

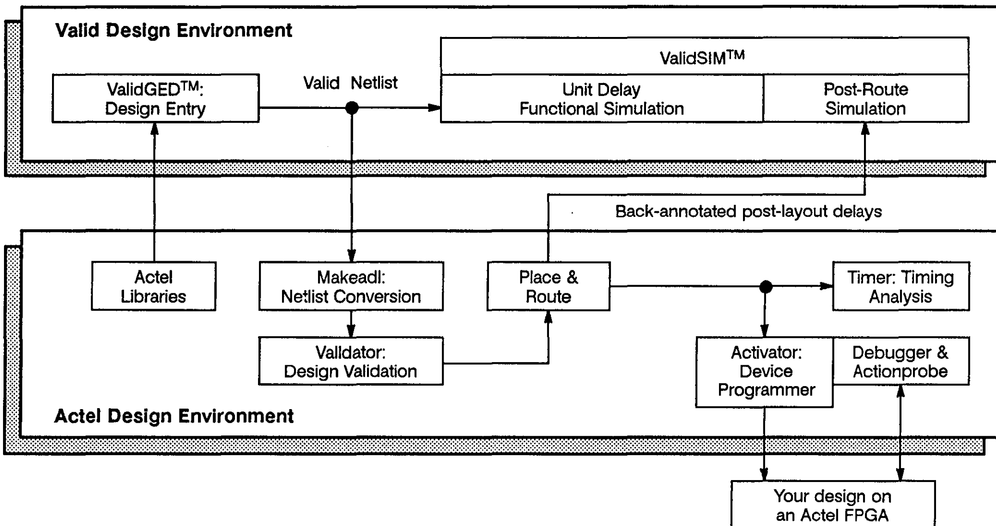


Figure 4. Actel - Valid Design Flow

ACT™ 2 Field Programmable Gate Arrays

Preliminary Product Brief

Features

- High Gate Count
  - Up to 8000 gate array gates (20,000 PLD/LCA equivalent gates)
- 1232 Logic Modules
- 140 User-Programmable I/O Pins
- 176-Pin PGA Package
- Instant Prototypes and Production
- Non-Volatile, Permanent Programming
- I/O Drive to 8 mA
- System-Level Performance to 50 MHz
- Enhanced Channeled-Array Architecture
  - Up to 750,000 Programming Elements
  - Up to 36 Horizontal and 15 Vertical Tracks
- Enhanced Logic Modules
- Supports Wide-input Combinatorial Functions (Up to five-Input AND gates)
- Supports Single-Module Sequential Functions
- Two In-circuit Diagnostic Probe Pins Support 50 MHz Analysis
- Two High-Speed Low-Skew Clock Networks
- Low-Power 1.2-µm CMOS Technology
  - Standby Current < 300 µA

Description

The ACT™ 2 family is the second generation of field programmable gate arrays from Actel. Based on Actel's patented channeled array architecture, the ACT 2 family provides significant enhancements to gate density and performance while maintaining upward compatibility from ACT 1 designs. The devices are implemented in silicon gate, 1.2-µm, two-level metal CMOS, and they employ Actel's PLICE™ antifuse technology. The unique architecture offers gate array flexibility, high performance and instant turnaround through user programming. Designs of up to 8000 gates can be implemented. The ACT 2 family is supported by the Action Logic™ System (ALS). ALS is available on Sun™, Apollo™, and 386 PC platforms, with CAE interfaces to Valid™, Viewlogic®, Mentor Graphics™, and OrCAD™.

Enhanced Logic Module

The logic module used in the ACT 2 family is an enhanced version of the ACT 1 module, allowing improved implementation of high fan-in combinatorial macros. The ACT 2 logic modules are also optimized to implement high-speed flip-flops and latches. Figure 1 depicts part of a state machine implemented with five ACT 2 modules. The critical path delay is only two module delays.

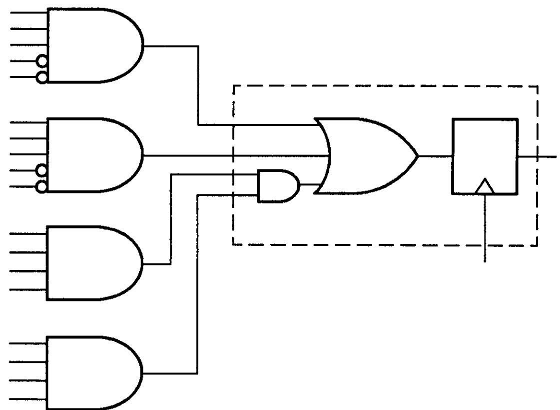


Figure 1. Part of State Machine Implemented In Five ACT 2 Modules.

Enhanced Architecture

Routing efficiency with large gate counts is accomplished with increased routing resources, increased antifuse programming elements, and architectural enhancements. Horizontal routing tracks/channel increase to 36 (vs 25 for ACT 1); vertical routing tracks/column increase to 15 (vs 13 for ACT 1). All speed-critical module-to-module connections are accomplished with only two low-resistance antifuse elements. Most connections are implemented with either two or three antifuse elements (as shown in Figure 2). No connections are allowed with more than four antifuse elements in the path. The result is predictable performance with fully automatic placement and routing. Device utilization is typically 85% to 95% of available logic modules and 80% of gates.

© 1990 Actel Corporation

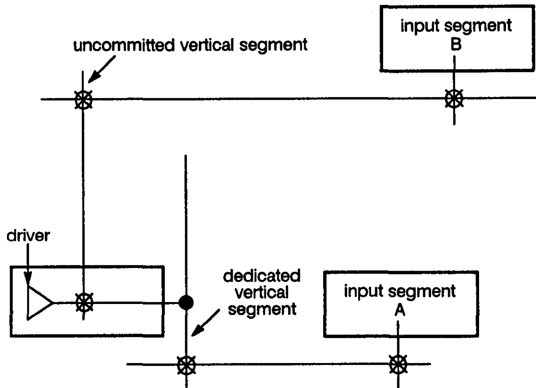


Figure 2. Enhanced Routing Architecture

Low-Skew Network

Two low-skew distribution networks are provided. Each network can be driven by either of two dedicated I/O pins or from internal logic.

Enhanced Programming and Test

The ACT 2 family provides the same guaranteed functionality as ACT 1 provides. All routing tracks, logic modules, and program, debug, and test circuits are fully tested in the factory. Verification of correct antifuse programming is performed automatically with Actel's Activator™ 2 program and debug hardware. The ACT 2 architecture implements an enhanced programming and test algorithm. Programming time for designs up to 8000 gates is comparable to programming time for 2000-gate designs implemented with the ACT 1 array.

Programmable I/O Pins

Each I/O pin can be configured as an input, output, three-state, or bidirectional buffer. Inputs are TTL- and CMOS-compatible. Output drive levels meet 8-mA TTL and 4-mA HCT standards. User-programmable slew rate is provided to limit simultaneous switching noise. Optional transparent latches at the I/O pins are provided for both inputs and outputs. I/O latches can be combined automatically with latches in the array to implement master-slave flip-flops, as depicted in Figure 3.

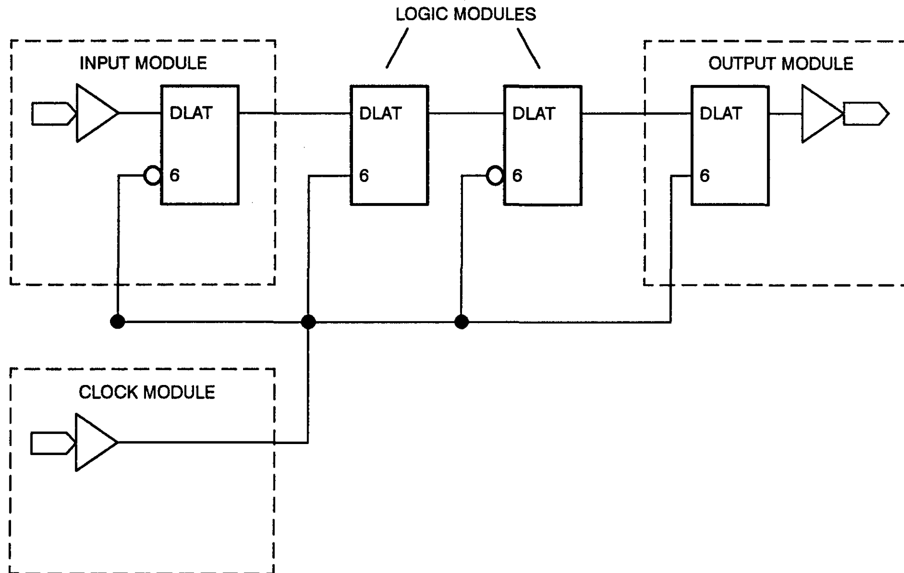


Figure 3. Latched User I/Os

Activator™ 2 Programmer/Tester/Debugger

Preliminary Product Brief

Features

- Supports ACT™ 1 and ACT 2 Device Families
- SCSI Connectors Interface to 386 PC, Sun™ and Apollo™ Workstations
- Adapter Modules for Each Package/Family Combination
- Up to Two Times Faster Than Activator™ 1 for ACT 1 Programming
- Supports Functional Verification with Actionprobe™ Diagnostic Pod
- Simultaneously Programs a Single Pattern in Up To Four Identical Devices
- In-Circuit Probing of Up To Four Devices Simultaneously

Product Description

Activator 2 is Actel's state-of-the-art desktop programmer. It utilizes Actel's Action Logic™ System (ALS) software and PLICE™ antifuse technology to program custom-engineered devices from Actel's ACT 1 and ACT 2 device families. SCSI connectors, shipped with the unit, provide a convenient connection for PC 386, Sun, and Apollo workstation support. Customized adapter modules for each device type permit easy interchange of programmable devices. Programming, Test, and Debug execute up to two times faster than on the Activator 1; users can program up to four devices at one time. The accompanying diagnostic pod and Actel debug software support observation of all internal signals.

The unit is software-driven, providing flexibility of application and a built-in barrier to obsolescence. Independently powered, the unit provides desktop device programming, functional test, and in-circuit debug.

The unit supports different CAE platforms using RS232 and SCSI connectors.

The Activator 2 may operate with four different adapter modules inserted, or three of the sockets may be slaves to the first, permitting simultaneous programming of four identical devices. The Activator 2 then addresses any one of the sockets for programming.

Activator 2 Base Unit

The base unit contains the control board and the analog board. The LED display on the top of the programmer shows when the unit receives power. Four adapter ports located on the top of the base unit accept the different adapter modules for each device type. The adapter ports are identical, allowing interchangeability of modules. RS232 and SCSI connectors are located on the back of the unit.

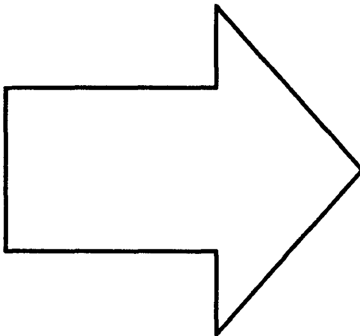
The Adapter Module

The adapter modules customize pin configurations for each device; the user need only switch adapter modules in order to program another device. All four ports may be in use simultaneously, when programming identical devices. The adapter modules are compact and sturdy. Any module may be used on any available port.

The Diagnostic Pod

The diagnostic pod supports ALS diagnostics and connects to the programmer via cable. The Activator permits the user to view any internal circuit activity through Actel's on-chip diagnostic ports. Up to four pods may be used simultaneously. A six-foot cord connects the pod to the base unit and provides debugging flexibility.

© 1990 Actel Corporation



Product Data 1

Reliability Report 2

Applications 3

Article Reprints 4

ACT 1010/1020 Reliability Report 2-1

ACT™ 1010/1020 Reliability Report

By Steve Chiang
and
Ken Hayes

ACT™ 1010/1020 devices are 1200- and 2000-gate (respectively) field programmable gate arrays (FPGAs). The programming element is an Actel-invented PLICE™ (Programmable Low-Impedance Circuit Element) antifuse. An antifuse is a normally open device in which an electrical connection is established by the application of a programming voltage. Although ACT 1010/1020 products are one-time programmable devices, their unique architecture features complete functional testability.

The ACT 1010/1020 device is processed using a standard 2 μm, double metal, CMOS process to which three additional masking steps have been added to implement the PLICE antifuse. A description of the main process parameters is shown in Table 1. Because ACT 1010/1020 devices are manufactured with a conventional CMOS process, normal CMOS failure modes will be observed. However, the addition of the antifuse adds another structure that could affect the device's reliability.

Actel has completed numerous studies in order to quantify the reliability of the antifuse. These studies lead to the conclusion that the time to failure of the antifuse is substantially more than 40 years under normal operating conditions and that the combined contribution of all antifuses to the gate array product's hard failure rate is less than 10 FITS (Failures-in-Time or 0.001% failures per 1000 hours).

Table 1. ACT 1010/1020 Process Description

CMOS, 2 μm, double metal, dual polysilicon, dual well, EPI wafer.

Dimensions		
	Width	**Space**
N+	4.0 μm	2.0 μm
P+	4.0	2.0
Cell Polysilicon	1.6	3.6
Gate Polysilicon	1.6	2.4
Metal I	4.0	2.0
Metal II	4.2	2.8
Contact	1.8 x 1.8	2.0
Via	2.0 x 2.0	2.0
Thickness		
Normal Gate Oxide		25 nm
High Voltage Gate Oxide		40
Cell Polysilicon		35
Gate Polysilicon		40
Metal I		80
Metal II		100
Passivation		1100
Composition		
Metal I	98% Al, 1% Si, 1% Cu	
Metal II	98% Al, 1% Si, 1% Cu	
Passivation	300 nm SiO_2, 800 nm SiN	

PLICE Antifuse Reliability

The antifuse is a vertical, two-terminal structure. It consists of a polysilicon layer on top, N+ doped silicon on the bottom, and an ONO (oxide-nitride-oxide) dielectric layer in-between. A Scanning Electron Microscope (SEM) cross-section of the antifuse is shown in Figure 1. On the ACT 1010/1020 device, the size of the antifuse is 1.8 μm^2. This small size, along with a low programmed on-resistance, typically 500 Ω, makes the PLICE antifuse a very attractive alternative to EPROM, EEPROM, or RAM for use as a programming element in a large programmable gate array. In the unprogrammed state, the resistance of the antifuse is in excess of 100 MΩ. The ACT 1010 and ACT 1020 contain 112,000 and 186,000 antifuses respectively. However, typical applications utilizing 85% of the available gates require programming only 2% to 3% of the available antifuses.

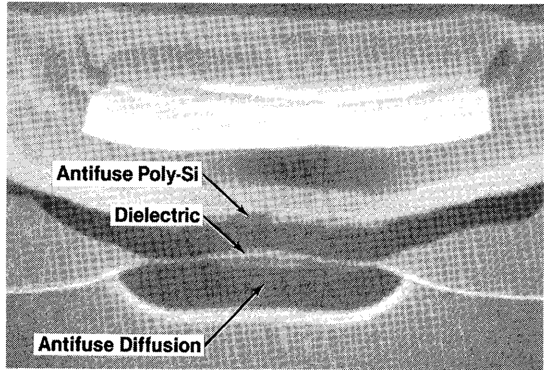


Figure 1. SEM Cross-Section of Antifuse

The Unprogrammed Antifuse

In order to evaluate antifuse reliability, Actel has developed models and collected data for both unprogrammed and programmed antifuses[1, 2]. We'll consider the unprogrammed antifuse first. Since the antifuse is a dielectric sandwiched between polysilicon and silicon, the model for its reliability, in the unprogrammed condition, is the same as that used to evaluate reliability of MOS transistor gate oxides. The parameter we wish to evaluate is the time to breakdown (t_{bd}) of the dielectric. This parameter is a function of the electric field across the dielectric as well as temperature and has the following relationship[3].

© 1990 Actel Corporation

$$t_{bd} = t_0 \cdot \exp(G/E) \quad (1)$$

where t_{bd} is the time to breakdown in seconds, t_0 is a constant, E is the electric field in MV/cm, and G is the field acceleration factor in MV/cm (G is a function of temperature).

By taking the log of both sides of equation 1 we have:

$$\ln(t_{bd}) = G \cdot (1/E) + \ln(t_0) \quad (2)$$

From experimental data, we can plot the log of the time to breakdown of the antifuse at various temperatures versus the reciprocal of the electric field across it and derive G from the slope and t_0 from the Y axis intercept. Actel has done this both on large antifuse capacitors and on arrays of 28,000 antifuses. An example is shown in Figure 2. From this we have derived a value for G of 510 MV/cm and a t_0 of 1×10^{-16} seconds. Also note the difference between the data at 25°C and 150°C.

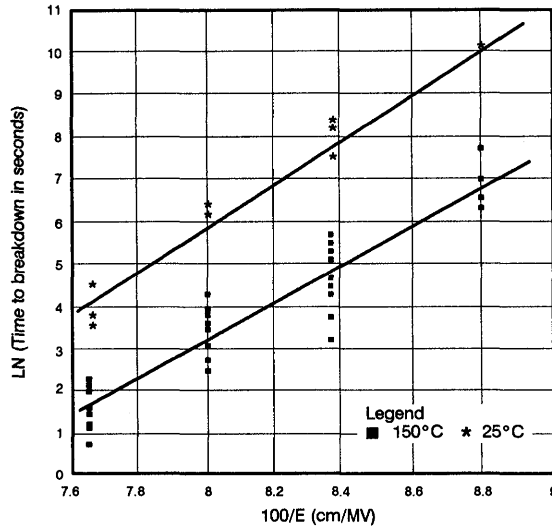


Figure 2. Field Acceleration of Antifuse (0.04 mm² Area Capacitor)

In order to quantify the temperature-dependence of the time to breakdown, we use the Arrhenius equation to determine the reaction rate (or semiconductor failure rate) of a given process (failure mode) over temperature:

$$R = R_0 \cdot \exp^{(-Ea/kT)} \quad (3)$$

where R is the reaction rate, R_0 is a constant for a particular process, T is the absolute temperature in degrees Kelvin, k is Boltzmann's constant (8.62×10^{-5} eV/°K), and E_a is the activation energy for the process in electron volts. To determine the acceleration factor for a given failure mode at temperature T_2 as compared with temperature T_1 we use equation 3 to derive:

$$A(T_1, T_2) = \exp[-(E_a/k) \cdot \{(1/T_1) - (1/T_2)\}] \quad (4)$$

where A is the acceleration factor.

For a given time to breakdown of a dielectric, the expression,

$$E_a = k \cdot d\ln(t_{bd})/d(1/T) \quad (5)$$

gives us the activation energy[2]. The field acceleration factor, G, is also temperature-dependent, i.e.

$$G(T) = G \cdot [1 + \delta/k \cdot \{1/T - 1/298\}] \quad (6)$$

where δ (in eV) characterizes the temperature-dependence of G.

E_a can be related to G (T) by:

$$E_a = G \cdot \delta/E - E_b \quad (7)$$

where δ and E_b are treated as fitting parameters between E_a and G.

From the data shown in Figure 2, as well as data on test arrays of 28,000 antifuses, we have derived a value for E_a of 0.2 eV. This value of E_a = 0.2 eV is for a very high E field of 11 MV/cm, or 10 V across the antifuse. With 5.5 V, the E field is about 6 MV/cm and E_a is approximately 0.6 eV. Values of δ and E_b were found from equation 7 to be 0.01 eV and 0.24 eV, respectively.

By combining equations 1, 5, 6, and 7, we obtain an overall equation for the time to breakdown for a given temperature and E field:

$$t_{bd} = t_0 \cdot \exp\{(G/E)[1 + (\delta/k) \cdot (1/T - 1/298)] - (E_b/k) \cdot (1/T - 1/298)\} \quad (8)$$

By applying the values for the constants as defined above, the time to breakdown for the antifuse dielectric can be derived for a given temperature and electric field. In Table 2, we have used equation 8 to solve for the acceleration factors associated with powering up a device at high voltage and/or temperature and comparing the failure rate with more typical voltages or temperatures. We can see the effect of temperature by comparing 125°C at 5.5 V with 55°C at 5.5 V in which the Actel model (equation 8) gives us an acceleration factor of 36, or 4.1 equivalent years for a 1000-hour burn-in at 125°C. Note that this acceleration factor of 36 is close to the value of 41.8 derived from the Arrhenius equation (equation 4) using an activation energy of 0.6 eV and the same temperatures. We use 0.6 eV as a general semiconductor failure mode activation energy when calculating failure rates from high-temperature operating life (HTOL) later in this report.

Table 2. Acceleration Factor vs. Operating Conditions (Unprogrammed Antifuse)

$t_0 = 1 \times 10^{-16}$ sec., G = 510 MV/cm, $\delta = 0.01$ eV.

Model	Temperature/Voltage		Acceleration Factor	Equivalent Years for 1000-Hour 125°C Burn-In
	High	Typical		
Fixed Voltage	125°C/5.5 V	55°C/5.5 V	36.0	4.1
	125°C/5.5 V	95°C/5.5 V	3.9	0.4
Fixed Temperature	25°C/5.5 V	25°C/5.25 V	48.7	5.6
	25°C/5.75 V	25°C/5.25 V	1692	193.2
	25°C/5.75 V	25°C/5.5 V	34.7	4.0
Varied Temperature and Voltage	125°C/5.5 V	55°C/5.25 V	1526	174.2
	125°C/5.75 V	55°C/5.5 V	883	100.8
	125°C/5.75 V	95°C/5.5 V	96.5	11.0
Fixed 0.6 eV (Activation Energy), Voltage-Independent	125°C/5.5 V	55°C/5.5 V	41.8	4.8
	125°C/5.5 V	95°C/5.5 V	4.2	0.5

We can also see from Table 2 that a small change in voltage is a much more effective reliability screen for the unprogrammed antifuse than is a change in temperature. For example, if we compare 25°C at 5.75 V to 25°C at 5.25 V we see that just a half volt change yields an acceleration factor of 1692, or 193.2 equivalent years per 1000 hours at 5.75 V. This strong dependence on voltage allows Actel to screen out antifuse infant mortality failures during normal wafer sort testing at Actel simply by performing a special test in which a higher than normal voltage is applied across all antifuses. Because antifuse infant mortality failures can be detected and effectively screened, ACT 1010/1020 devices have as high a level of reliability as standard CMOS processed products.

Actel has collected data on over 350 antifuse test devices representing eleven wafer runs. From this data, we have determined that the contribution of the antifuse to overall device reliability is less than 10 FITS. This conclusion is confirmed by the reliability data taken on actual ACT 1010/1020 units which will be discussed later in this paper.

The Programmed Antifuse

A Kelvin test structure as shown in Figure 3 was used to evaluate reliability of a programmed antifuse. Here, a strip of polysilicon crosses an N+ diffusion. The antifuse is located at their intersection. There are metal-to-poly contacts at nodes 1 and 3 as well as metal-to-N+ contacts at nodes 2 and 4. A four-terminal Kelvin structure is useful should a failure occur, because antifuse opens can be separated from other problems (such as polysilicon or contact opens) simply by checking for continuity on appropriate pairs of nodes.

Test devices were stressed by forcing a constant 5 mA current from polysilicon to N+ diffusion through the antifuse at 250°C. Note that this stress is far greater than what a programmed antifuse would see in a device operating under normal conditions. Because the antifuse is used to connect two networks together, there is usually no voltage across it, hence no current passes through. A voltage will appear across the antifuse only momentarily while a network switches from low-to-high or high-to-low.

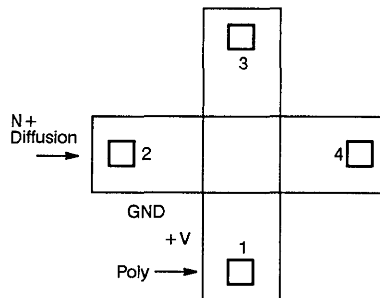


Figure 3. Antifuse Kelvin Structure

During the 5 mA, 250°C stress, the voltage across the antifuse was monitored. Figure 4 is a plot of the monitored voltage as a function of stress time. A sudden increase in voltage indicates that an open occurred. As can be seen from the figure, failures occurred at about 300 hours of stress. However, by probing on nodes 3 and 4 of the Kelvin structure, we were able to measure continuity and determine that the cause of failure was not the antifuse. The failed units were then examined on an SEM, where the cause of failure was revealed as metal-to-poly contact electromigration. This is a well-known failure mode in CMOS, which has been determined to have an activation energy of 0.9 eV. Using equation 4 we can predict a lifetime under normal operating conditions in excess of 40 years for this failure mode. The lifetime of the programmed antifuse is even longer.

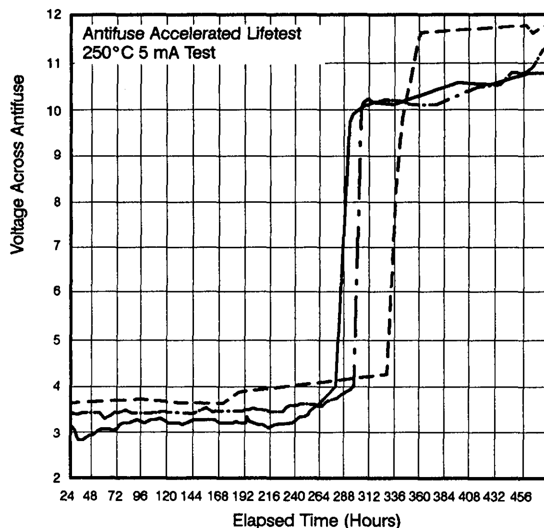


Figure 4. Voltage Across Antifuse versus Stress Time

ACT 1010/1020 Device Reliability

Device reliability was evaluated on four Actel products: a 64K PROM (PROM64), a 300-gate FPGA (1003), a 1200-gate FPGA (ACT 1010/1010A), and a 2000-gate FPGA (ACT 1020/1020A). The PROM64 product uses the same process and antifuse as the ACT 1010/1020. The 1003 is a test device, a smaller version of the ACT 1010/1020, which was used for early characterization and qualification. The ACT 1010A and ACT 1020A are 20% linear die shrink versions of the ACT 1010 and ACT 1020, respectively. The PROM64 units were packaged in 24-pin side-brazed packages; the FPGA units were in 68- and 84-pin JLCC (ceramic J-leaded chip carriers) and PLCC (plastic-leaded chip carriers) packages. Package characteristics for ACT 1010/1020 devices are shown in Table 3.

Table 3. ACT 1010/1020 Package Characteristics

PLCC	
Molding Compound	Sumitomo 6300 H
Filler Material	Fused silicon 70% by weight
Lead Frame Material	Copper
Lead Plating Composition	Tin or solder, 300 to 800 micro inches (μin)
Die Attach Material	Silver Epoxy
Die Coat	Dow Corning Q14939 silicon gel
Bond Wire	Gold, 1.3 mil diameter
Bond Attach Method	Thermosonic

JLCC	
Body Material	Ceramic
Lid and Lead Material	Alloy 42 with minimum 60-μin gold plate
Bond Wire	99% Aluminum, 1% Si, 1.25 mil diameter
Bond Attach Method	Ultrasonic

Thermal Resistance (°C/Watt)

Package	44 PLCC	44 JLCC	68 PLCC	68 JLCC	84 PLCC	84 JLCC	84 PGA
θ_{JC}	15	5	13	5	12	5	8
θ_{JA}	52	38	45	35	44	34	35

High Temperature Operating Life (HTOL) Test

The intent of HTOL is to dynamically operate a device at high temperature (usually 125°C) and extrapolate the failure rate to typical operating conditions. This test is defined by Military Standard-883 in the Group C Quality Conformance Tests. The Arrhenius relationship in equations 3 and 4 is used to do the extrapolation. To use the Arrhenius equation, we need to know the activation energy of the failure mode. Activation energies of antifuse failure modes were discussed earlier. Table 4 gives the activation energies of general semiconductor failure modes.

Table 4. Activation Energy of CMOS Failures

Failure Mechanism	Activation Energy
Ionic Contamination	1.0 eV
Oxide Defects	0.3 eV
Hot Carrier Trapping in Oxide (Short Channels)	-0.06 eV
Silicon Defects	0.5 eV
Aluminum Electromigration	0.5 eV
Contact Electromigration	0.9 eV
Electrolytic Corrosion	0.54 eV

Six different data patterns were programmed into the 64K PROMs for HTOL testing: a diagonal of zeros (98% programmed); a diagonal of ones (2% programmed); a topological checkerboard pattern (50%); all zeros (100%); all ones (0%); and an incrementing pattern (50%). During burn-in, all addresses are sequenced through at a 1 MHz clock rate. The outputs are enabled and loaded with a 100 ohm resistor to a 2 V supply. This results in an output loading of equal to or greater than the data sheet specified limits of I_{OH} = -4 mA and I_{OL} = 16 mA. In most cases, the PROMs were burned-in at V_{CC} = 5.5 V, 125°C. However, voltage acceleration experiments were also done at 7 V, 125°C as well as at 8 V, 25°C.

The PROM is useful for antifuse reliability studies for several reasons. First of all, we can program anywhere from 0% to 100% of the antifuses although we program only 2% to 3% of the antifuses for a given design on the ACT 1010/1020 device. Also, an antifuse failure on the PROM is very noticeable, since the antifuse is directly addressed. A weak antifuse would show an AC speed drift, and a failed antifuse would read the wrong data.

To evaluate the ACT 1003/1010/1020 devices, we programmed an actual design application into each device and performed a dynamic burn-in by toggling the clock pins at a 1 MHz rate. The designs selected utilized 85% to 97% of the available logic modules and 85% to 94% of the I/Os. Outputs were loaded with 1.2 kΩ resistors to V_{DD} resulting in greater than 4 mA of sink current as each I/O toggled low. Under these conditions, each ACT 1010 typically draws about 100 mA during dynamic burn-in. Most of this current comes from the output loading while about 5 mA is from the device supply current. The thermal resistance (junction to ambient) of the 68- and 84-pin PLCC packages is about 45°C/Watt; for the JLCC packages it is about 35°C/Watt. For a 125°C burn-in, this results in junction temperatures of about 150°C for plastic packages and 145°C for ceramic packages. Most burn-in was done at 5.75 V (for voltage acceleration of the antifuse) and 125°C. Some data was taken at 5.5 V, 125°C and at 5.5 V, 150°C.

A summary of the HTOL data collected by Actel is shown in Table 5. A failure is defined as any device which shows a functional failure, exceeds data sheet DC limits, or exhibits any AC speed drift. Among the parts tested, no speed drift, faster or slower, was

observed within the accuracy of the test set-up. Failure rates at 55°C and 70°C were extrapolated by using the Arrhenius equation and a general activation energy of 0.6 eV. Poisson statistics were used to derive a calculated failure rate with a 60% confidence level. Use of Poisson statistics is valid for a failure rate which is low and a failure mode which occurs randomly with time. At 55°C, the calculated failure rate with a 60% confidence level was found to be 33 FITS (0.0033% failures per 1000 hours). This number was derived from over 2.2 million device hours of data.

Both of the observed failures were normal CMOS failures and were not caused by the antifuses. The 1003 device failed after 80 hours at 150°C. The unit was functional but had high I_{DD} current (40 mA vs. 4 mA typical). Liquid crystal analysis revealed a hot spot outside the antifuse area of the chip. The other failed unit was nonfunctional, with a high I_{DD} current of about 40 mA. This unit failed after 500 hours at 125°C. Both failures are believed to be the results of junction degradation or silicon defects.

Table 5. HTOL (High Temperature Operating Life) Test

Device	Units	Runs	Device Hours at 125°C	Failures	Equivalent Device Hours at 55°C (0.6 eV)	Equivalent Device Hours at 70°C (0.6 eV)
PROM64	275	4	568,000	0	23.8 Million	9.4 Million
1003 JLCC	238	3	359,400	1	15.0	5.9
1010 JLCC	144	6	283,000	0	11.8	4.7
1020 JLCC	61	2	90,000	0	3.8	1.5
1020A JLCC	69	2	69,000	0	2.9	1.1
1010 PLCC	496	6	844,000	1	35.3	14.0
1020 PLCC	32	2	48,000	0	2.0	0.8
Totals:	**1315**	**20**	**2,261,400**	**2**	**94.6 Million**	**37.4 Million**
Summary:						
Observed FITS at 55°C:			21			
Observed FITS at 70°C:			53			
Calculated FITS to 60% confidence at 55°C:			33			
Calculated FITS to 60% confidence at 70°C:			83			
(Summary of Data as of February 20, 1990)						

Unbiased Steam Pressure Pot Test

This test is used to qualify products in plastic packages. Units are placed in an autoclave (pressure pot) and exposed to a saturated steam atmosphere at 121°C and 15 psi. Problems with bonding, molding compounds, or wafer passivation can cause metal corrosion to occur in this atmosphere. Metal corrosion is detected during a full electrical test of the device following exposure to the autoclave environment.

A total of 426 units from five wafer runs and six assembly lots were used. Read points were taken at 96, 168, and 336 hours. There were a total of four failures (Table 6). All four failures were caused by bond wires lifting off bond pads. This was an assembly problem that occurred only on our first lot of plastic units. The failures were caused by high temperature, not by metal corrosion. The assembly problem was corrected, with no further failures observed.

Table 6. Unbiased Steam Pressure Pot Test
121°C, 15 psi

				Number of Failures		
Product	Run Number	Package	Number of Units	96 Hours	168 Hours	336 Hours
1010	JB13	84 PLCC	34	0	3	0
1010	JB13	68 PLCC	71	1	0	0
1010	JB14	68 PLCC	71	0	0	0
1010	JB22	68 PLCC	71	0	0	0
1010	JB27	68 PLCC	50	0	0	0
1010A	TI24	68 PLCC	129	0	0	0

Biased Moisture Life Test (85/85)

In this test, the units are placed in a chamber at a temperature of 85°C and a relative humidity of 85%. A voltage of 5.5 V is applied to every other device pin while other pins are grounded. 5.5 V is applied to V_{DD} while V_{SS} is grounded. This test is effective at detecting die related and plastic package related problems.

As shown in Table 7, a total of 288 units were stressed. There were three failures. One failure was caused by two lifted bond wires; it was from the same lot in which we saw failures in steam pressure pot test. The second unit was functional but had high I_{DD} current. The 1000-hour failure was nonfunctional.

Table 7. Biased Moisture Life Test
85°C/85% Humidity with DC Alternate Pin Bias of 0 V to 5.5 V

				Number of Failures		
Product	**Run Number**	**Package**	**Number of Units**	**500 Hours**	**1000 Hours**	**2000 Hours**
1010	JB13	68 PLCC	80	2	1	0
1010	JB14	68 PLCC	81	0	0	0
1010	JB22	68 PLCC	54	0	0	–
1010	JB26	68 PLCC	54	0	0	–
1010	JB27	68 PLCC	19	0	0	–

Temperature Cycling

This test checks for package integrity by cycling units through temperature extremes. For ceramic packages, the range of temperature is -65°C to 150°C. For plastic packages, the range is 0°C to 125°C. Both programmed and unprogrammed units are placed on temperature cycle. Data on 451 units is summarized in Table 8.

Table 8. Temperature Cycling Test
-65°C to 150°C Ceramic; 0°C to 125°C Plastic

				Failures		
Product	**Run Number**	**Package**	**Number of Units**	**100 Cycles**	**200 Cycles**	**1000 Cycles**
1010	JB13	68 PLCC	158	0	–	0
1010	JB14	68 PLCC	28	0	–	0
1010	JB26	68 PLCC	21	0	–	0
1010	JB28	68 PLCC	31	0	–	0
1020	JB22	84 PLCC	17	0	–	0
1010	1077	84 JLCC	20	0	0	
1020	JB33	84 PGA	25	0		
1010A	TI24	68 PLCC	176	0	–	0

2

Other Tests

Electrostatic Discharge (ESD)

Units were tested for sensitivity to static electricity by using the human body model as described in MIL-883C (100 pF discharged through 1.5 kΩ). Fifteen ACT 1010 units from three wafer runs were tested. Nine representative I/O pins were checked on each device. Since all I/O pins have the same layout on the chip, the nine pins tested were selected based on their proximity to V_{SS}, V_{DD}, or the corner of the chip. The MODE pin was also tested because it is the only dedicated input on the chip. In addition, the three power supplies (V_{SS}, V_{DD}, V_{PP}) were tested. Three positive and three negative pulses were discharged into each pin tested at each voltage level. For inputs and I/Os, these six pulses were applied with three different grounding conditions; V_{SS} only grounded, V_{DD} only grounded, and all other I/Os grounded. Thus each pin received a total of eighteen pulses for each test voltage. Testing began at 1000 V and continued in 500 V increments. After pulsing was completed at each voltage, the I-V characteristic of each pin was checked on a digital curve tracer. Any significant change in the I-V curve from the previous reading was considered a failure. The units were then tested on a VLSI tester. Leakage currents were datalogged at 0 V and 5.5 V. Any pin showing more than 250 nA of leakage current was also considered to be a failure. For I_{DD}, a change of more than 250 μA was cause for rejection. No failures occurred through 2000 V. At 2500 V, five of the fifteen units failed on at least one pin. Failure analysis revealed that the failures occurred on the N-channel pulldown transistor of the output driver. With no failures through 2000 V testing, the ACT 1010 (and the ACT 1020, by virtue of identical I/O layout to the 1010 device) met the requirements for the 2000 V ESD category of MIL-883C.

Latch-up

Latch-up is a well-known cause of failure in CMOS circuits. Parasitic bipolar transistors are created by the P-channel transistor, the N-channel transistors, the N-well, and the P-substrate. These transistors are connected in a manner which effectively creates an SCR. If a voltage on an external pin were to forward bias to the substrate, the parasitic SCR can be latched to the on state, creating a low-impedance path between V_{DD} and ground. A large amount of current then flows through this path. This current can, at best, make the device temporarily nonfunctional and, at worst, cause permanent damage.

Several techniques are used by CMOS designers to reduce the chance of latch-up. One of the most common techniques is the use of guard rings to isolate P-channel and N-channel transistors. The disadvantage of this method is that it requires additional silicon die area. Another method is to use a substrate bias generator. Creating a negative substrate bias means that an input must go even more negative to cause latch-up. A third technique is to use EPI wafers to achieve low substrate resistance, which lowers the chances of triggering latch-up. Actel designers use both guard ring and EPI wafer techniques for ACT 1010/1020 devices.

The latch-up test method used is defined by JEDEC Standard No. 17. Each I/O pin on a tested device was forward biased in both directions (to V_{SS} and V_{DD}) by forcing negative and positive currents ranging from ±50 mA to ±250 mA in 50 mA increments. Following each stress, the device I_{DD} current was measured. If the current exceeded the data sheet limit of 10 mA, the unit would be rejected. The device was also functionally tested.

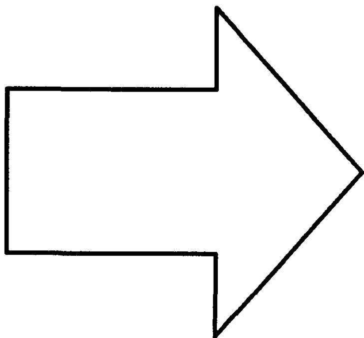
Fifteen units from three different wafer lots were tested. Testing was done both at room temperature and at a worst case temperature of 135°C. All device I/Os and power supplies were tested. No failures were detected through 250 mA.

Conclusion

The data presented in this report establishes the excellent reliability of Actel ACT 1010/1020 devices. Both the Actel models and the test devices show that the antifuse is highly reliable and that it detracts negligibly from overall product reliability.

References

1) E. Hamdy, et al, *"Dielectric Based Antifuse for Logic and Memory ICs."* IEDM paper, p. 786-789, 1988.
2) S. Chiang, et al, *"Oxide-Nitride-Oxide Antifuse Reliability."* To be published, Proc. Int. Rel. Phys. Symp., 1990.
3) J. Lee, I-Chen, and C. Hu, *"Modeling and Characterization of Gate Oxide Reliability,"* IEEE Trans. of Elec. Dev., Dec. 1988.



Product Data 1

Reliability Report 2

Applications 3

Article Reprints 4

Which Actel array should you use? 3-1

Efficient Logic Conversions 3-3

The Actel Timer 3-15

Using Actionprobe Diagnostic Tools 3-17

Metastability 3-19

Using the Actel Debugger as a Functional Tester 3-21

Calculating Power in Actel's ACT 1 Arrays 3-25

Three-Stating ACT 1010/1020 Designs 3-29

Multi-ASIC and System Simulation with Actel FPGAs 3-31

Constructing RAM with ACT 1 Macros 3-33

Using the DRAM Controller Supermacro 3-39

Using the DMA Controller Supermacro 3-43

Using the SCSI Interface Controller Supermacro 3-45

UART Design 3-49

Actel TA181 ALU 3-61

Actel Fast Adders 3-63

Eight-Bit Twos Complement Multiplier 3-65

Contributors:

Sam Beal
John Day
Dennis McCarty
Faysal Sohail
Quat Tran

Which Actel array should you use?

Applications Note

Determining which array to use is an important part of design implementation. This applications note shows how to easily select the correct Actel ACT™ 1 array for your design.

The ACT 1 family of gate arrays contains the ACT 1010, a 1200-gate array, and the ACT 1020, a 2000-gate array.

The ACT 1010 contains 295 logic modules and 57 I/O modules. The ACT 1020 contains 547 logic modules and 69 I/O modules. Logic modules can be configured as gates, flip-flops, and other logic functions. An I/O module can be designated as an input, output, or bidirectional buffer.

Each array contains a master clock buffer. This buffer can drive flip-flops or latches without fan-out restriction.

To determine the correct device for your design, first consider I/O requirements—including simultaneously switching outputs (SSOs)—then consider logic requirements.

In determining I/O requirements, consider the package limitations, as shown in the table below. Power and ground are not included in the limits given; separate bond pads and package pins are provided for power and ground. SSOs are outputs that switch within 10 ns of each other.

I/O Limits

	ACT 1010		ACT 1020	
Package	I/Os	SSOs	I/Os	SSOs
44-pin chip carrier	34	16	34	16
68-pin chip carrier	57	24	57	24
84-pin chip carrier	N/A	N/A	69	32
84-pin grid array	57	32	69	32

To calculate the number of logic modules required by a design, add up the individual logic module requirements of each macro used. The following table gives typical logic module requirements for various hard macro functions. Consult the Actel Macro Library for a complete listing of macro types and actual logic module requirements. For calculation purposes, convert complex functions (e.g, 74163) into gate and flip-flop schematic equivalents. Refer to the manufacturer's databook for this information.

Note that both true and inverting inputs are provided on all 2-, 3-, and 4-input functions, so inverters are rarely needed in Actel designs.

Design limits for Actel devices are 295 logic modules for the ACT 1010, and 547 logic modules for the ACT 1020. But device utilization, the percentage of available modules used, must be considered also. Designs using 85% or less of available logic modules can be placed and routed easily. Device utilization above 85% may lead to problems, especially when designing for fast AC performance.

Logic Module Requirements

Macro Function	Logic Modules
2-input gate	1
3-input gate	1
4-input gate	2
XOR gate	1
XOR OR gate	1
XOR AND gate	1
AND XOR gate	1
AND OR gate	1
OR AND gate	1
Buffer/Inverter	1
2- or 4-input Multiplexor	1
Adder	2
D-latch	1
D-latch with clear	1
D-latch with enable	1
Multiplexed latch	1
D flip-flops	2
Multiplexed D flip-flops	2
D flip-flop with enable	2
JK flip-flop	2

© 1990 Actel Corporation

Efficient Logic Conversions

Applications Note

Introduction

In order to obtain the most efficient conversion of a logic design into Actel's format, the designer should become familiar with the available Actel macros. The Actel Macro Library manual lists available Action Logic™ System macros, with AC performance, fan-in, and levels of logic given.

Most common logic functions are readily implemented using Actel macros. This applications note describes several common functions created from Actel macros, and serves as a designer's guide to efficient logic conversions. New macros are frequently added to the Actel Macro Library. Always check the current software library shipped with the Action Logic System before creating a macro.

All Actel macros are created with logic modules. Each logic module is an eight-input, one-output circuit, which consists of three 2-to-1 multiplexors and one 2-input OR gate. See Figure 1.

This circuit can implement a wide range of functions, and it interconnects efficiently through routing resources as required by the circuit application.

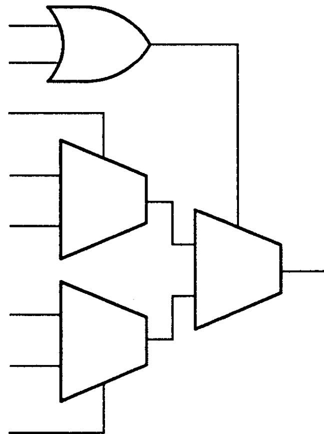


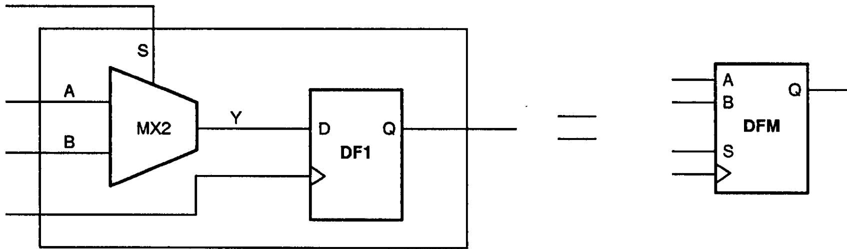
Figure 1. Logic Module Schematic

Actel macros fall into two categories, hard macros and soft macros. Hard macros consist of one or two logic modules. Soft macros were created for logic functions requiring more than two logic modules. Hard macros have fixed logic module positions and wiring interconnections to maintain specified AC characteristics. Soft macro logic modules are placed on the array as needed, and the wiring path is routed to complete the interconnection. Soft macro AC performance is determined after place and route.

© 1990 Actel Corporation

Combinatorial Functions Existing In The Macro Library

Since the basic macro building block (the logic module) is created from multiplexors, the multiplexor can be viewed as the most "cost effective" function. Many macros have multiplexor functions built in; Figure 2 illustrates how several logic functions are combined, saving space and power.



Equivalent schematic of a DFM macro

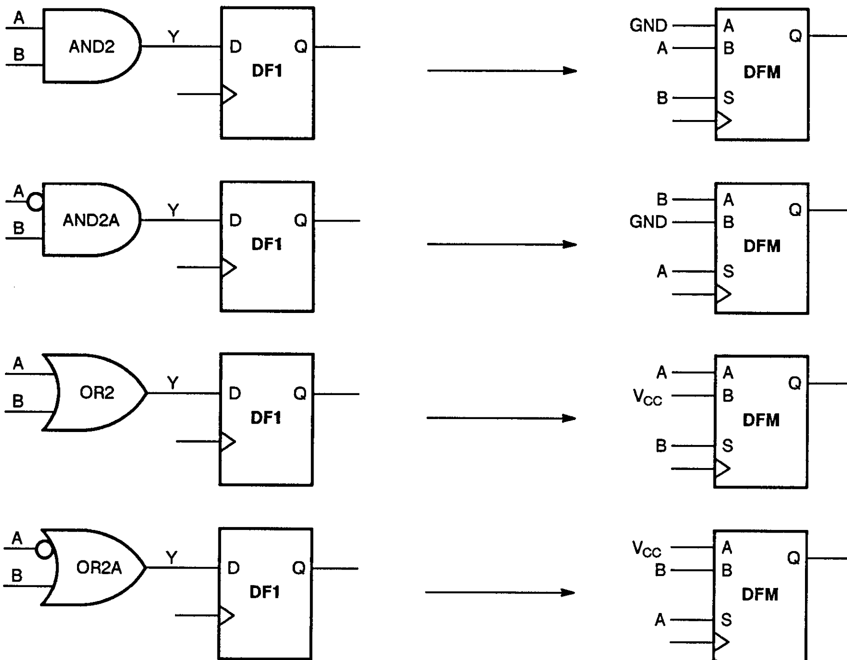


Figure 2. Sample Logic Function Equivalents

Note from Figure 2 that the DFM multiplexed D-input flip-flop makes use of its inputs to achieve other functions by simply connecting V_{CC} and GND as shown. The original circuitry consisting of a gate and flip-flop would have used three logic modules, while the more efficient version uses only two. This is a very powerful technique, and can be extended to other functions in the library.

Actel provides every combination of 2-, 3- and 4-input functions with true and inverting inputs, so inverters are almost never needed. Figure 3 illustrates the logic savings realized using Actel macros in a standard circuit.

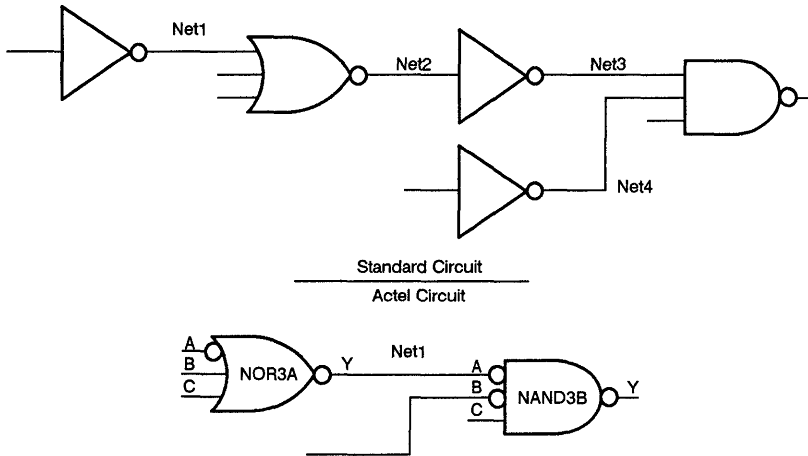


Figure 3. Sample Logic Conversion

The standard circuit requires five logic elements. The Actel circuit requires only two logic modules; it runs faster and uses less power than the standard circuit.

Counting Logic Modules

When converting a logic design into an Actel array, do not count the number of gates in the original design; count logic module requirements. Many standard logic functions can be implemented with fewer Actel logic modules than expected, as shown in the examples in this application note. Table 1 gives typical logic module requirements for various hard macro functions. Many complex functions, such as TTL replacements, exist as macros in the Actel library. Consult the Actel Macro Library for a complete listing of macro types and actual logic module requirements.

Both logic module and I/O module requirements must be considered in selecting an array for a given design. Refer to the ACT 1 data sheet for array specifications. For more information on choosing an array for your design, refer to the applications note "Which Actel array should you use?"

Table 1. Logic Module Requirements

Macro Function	Logic Modules	Notes
2-input gate	1	Includes AND, NAND, OR and NOR gates.
3-input gate	1	NAND3 and OR3C require two logic modules.
4-input gate	2	See Note 1.
XOR gate	1	
XOR OR gate	1	XOR/XNOR followed by an OR gate.
XOR AND gate	1	XOR/XNOR followed by an AND gate.
AND XOR gate	1	AND gate followed by an XOR gate.
AND OR gate	1	12 types available.
OR AND gate	1	9 types available.
Buffer/Inverter	1	Least efficient of all macro types.
2- or 4-input Multiplexor	1	Most efficient macro. 6 types available.
Adders	2	7 types available
D-latch	1	
D-latch with clear	1	
D-latch with enable	1	
Multiplexed latch	1	2 input Multiplexed D-latch.
D flip-flop	2	22 types available. Single output only.
Multiplexed D flip-flop	2	
D flip-flop with enable	2	Allows multiple clocked systems.

Note:

1. The designer could use an average of 1.5 logic modules for the four-input gate conversions, as 50% of these gates use only one logic module.

Creating Logic Functions With Hard Macros

One of the most efficient logic functions in the Actel Library is the MXT macro. Figure 4 illustrates this function. The MXT macro uses one logic module.

Many logic functions can be created using this macro, as illustrated in Figures 4a, 4b, 4c, and 4e. Figure 4d illustrates set-to-1 flip-flops created from Actel macros DLC and DLCA.

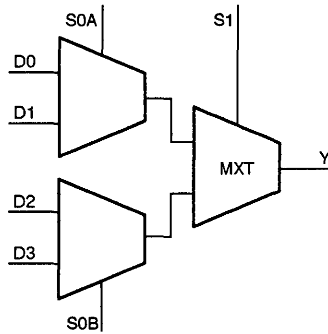


Figure 4. MXT Macro

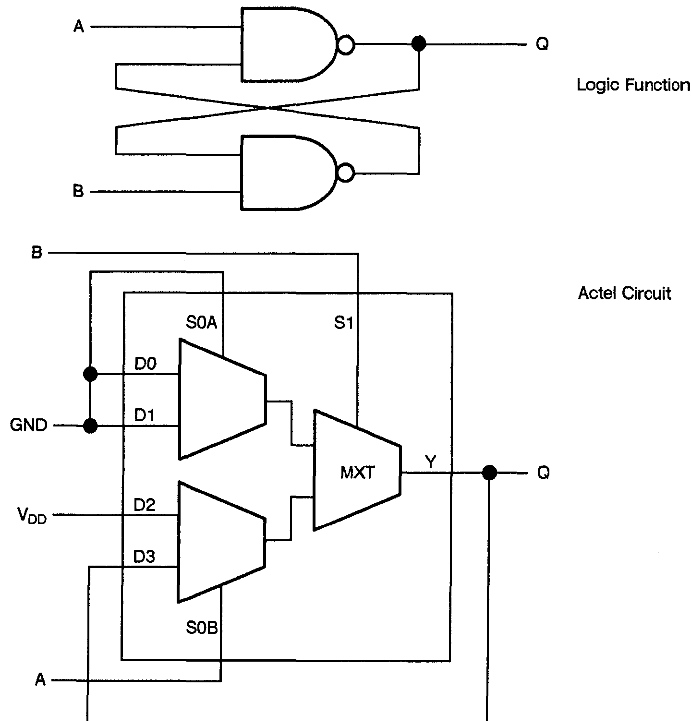


Figure 4a. NAND Gate Set – Reset Latch

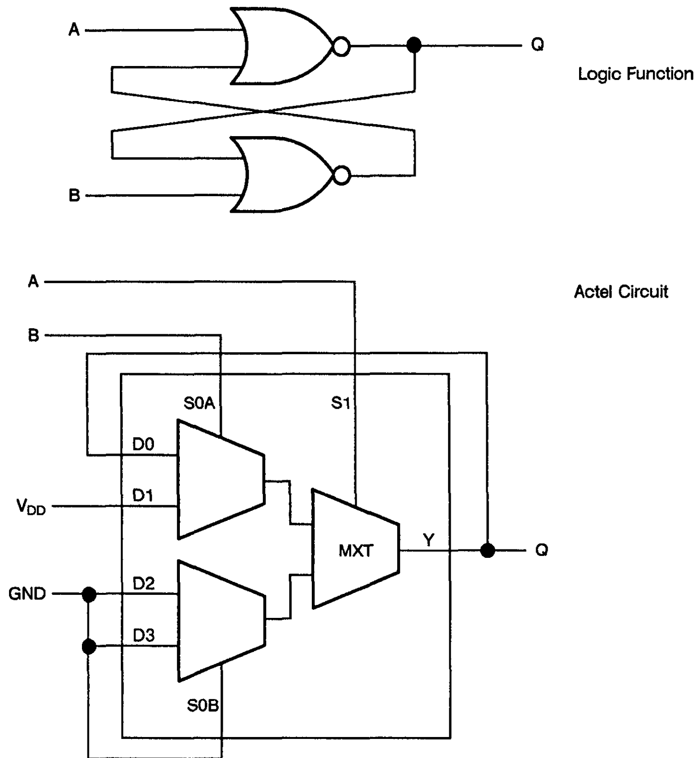
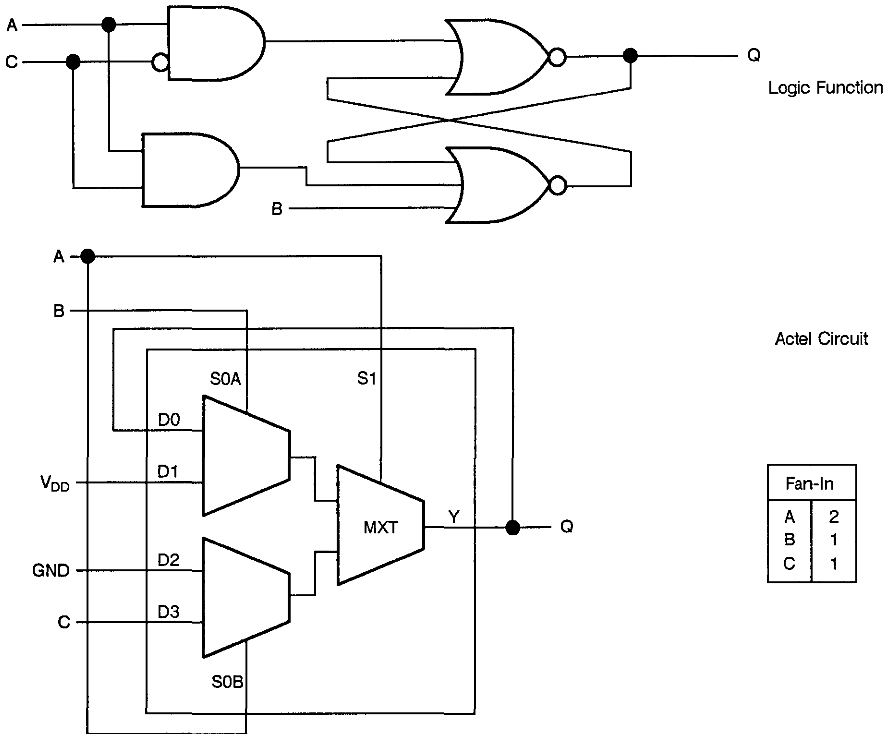


Figure 4b. NOR Gate Set - Reset Latch

Figures 4a and 4b illustrate how two types of S–R latches can be implemented using the MXT macro (one logic module) in place of two logic elements. Using the MXT macro, the Actel circuit provides improved speed and reduced power dissipation over the standard circuit.

By manipulating the inputs tied to GND and V_{DD}, further logic functions can be built in. Figure 4c illustrates how additional gates can be added to the NOR gate S–R latch using existing logic. Note that two levels of logic in the standard circuit, from the A and C inputs to the Q output, have been reduced to one level of logic, improving AC performance.



Fan-In	
A	2
B	1
C	1

Figure 4c. NOR Gate Set – Reset Latch With Both Inputs Gated

The circuit in Figure 4c can prevent the illegal state (when both A and B are active true) if the designer connects the B input to the C input. B will override A, and when both inputs are removed simultaneously, a logic 1 will be maintained at the output. Note that the fan-in is not always represented by the schematic, as noted by input C.

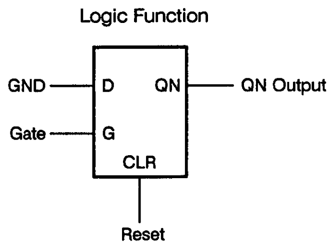
Figures 4d and 4e illustrate another set of useful logic functions: Set-to-1 and Set-to-0 flip-flops with reset. The Set-to-1 functions are created with Actel DLC and DLCA macros, while the Set-to-0 functions, shown in Figure 4e, are created with the MXT macro.

For the DLC macro, a logic 1 on the gate input will set the Q output to a logic 1. For the DLCA macro, a logic 0 on the gate input will set the Q output to a logic 1.

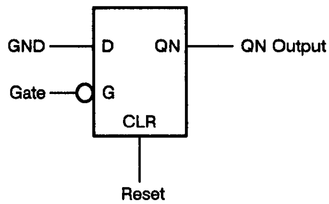
Figure 4d. Set-to-1 Flip-Flops

Logic Function

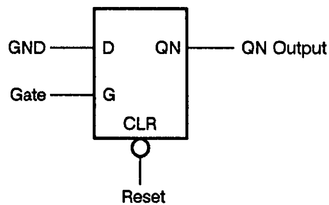
Actel Circuit



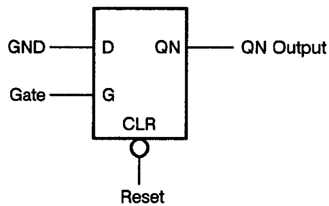
Reset
Gate
S0A
S1
D0
GND
D1
MXT
Y
QN Output
V_{DD}
D2
D3
S0B



Reset
Gate
S0A
S1
GND
D0
D1
MXT
Y
QN Output
V_{DD}
D2
D3
S0B



Reset
S0A
S1
D0
V_{DD}
D1
MXT
Y
QN Output
D2
GND
D3
S0B
Gate



Reset
S0A
S1
D0
V_{DD}
D1
MXT
Y
QN Output
GND
D2
D3
S0B
Gate

Figure 4e. Set-to-0 Flip-Flops

Implementing Logic Functions Using Multiplexors

Many functions can be implemented using the simple two-input multiplexor. Figure 5a illustrates four common functions and their multiplexor equivalents.

The designer can combine several functions using one MXT macro, as shown in Figure 5b. Note the savings of one level of logic, which saves power and improves speed.

The function shown in Figure 5b exists as a macro (AO1C) in the Actel Library.

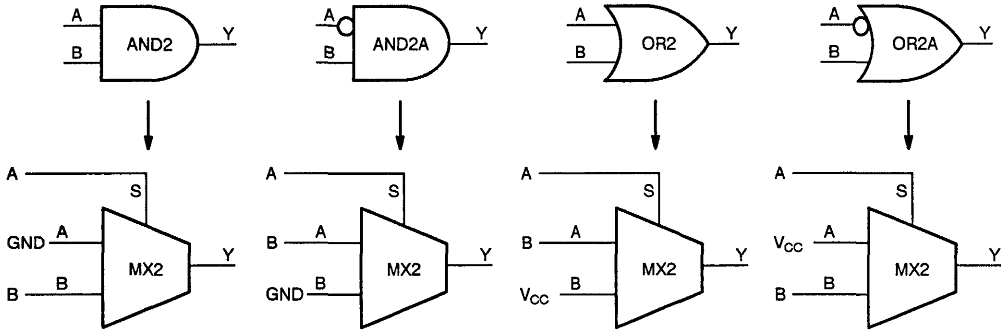


Figure 5a. Multiplexor Functional Equivalents

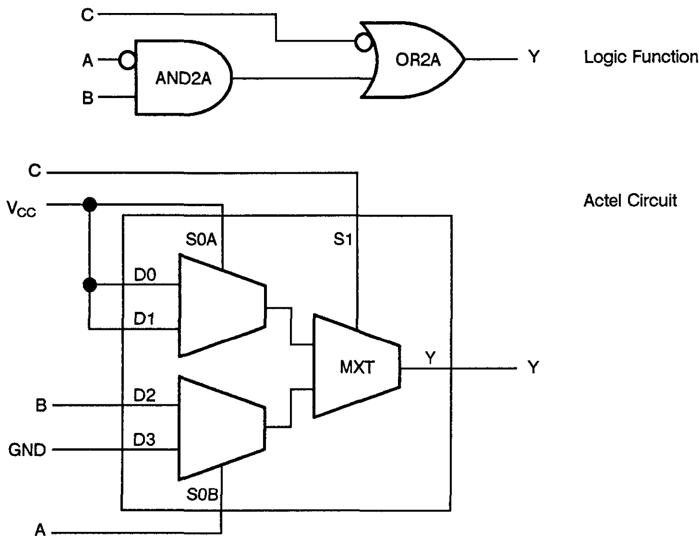


Figure 5b. Implementing Logic with the MXT Macro

Creating Toggle Flip-Flops With Q Outputs

Because Actel's flip-flops have only one output, Q or QN, creating a toggle flip-flop with a Q output usually involves the addition of an inverter (see Figure 6). Figure 6 also illustrates a way to eliminate the inverter and save one logic module. Note that the loading of the Q output remains the same for both circuits. The advantages are in eliminating one logic module, improving speed, and lowering overall power dissipation.

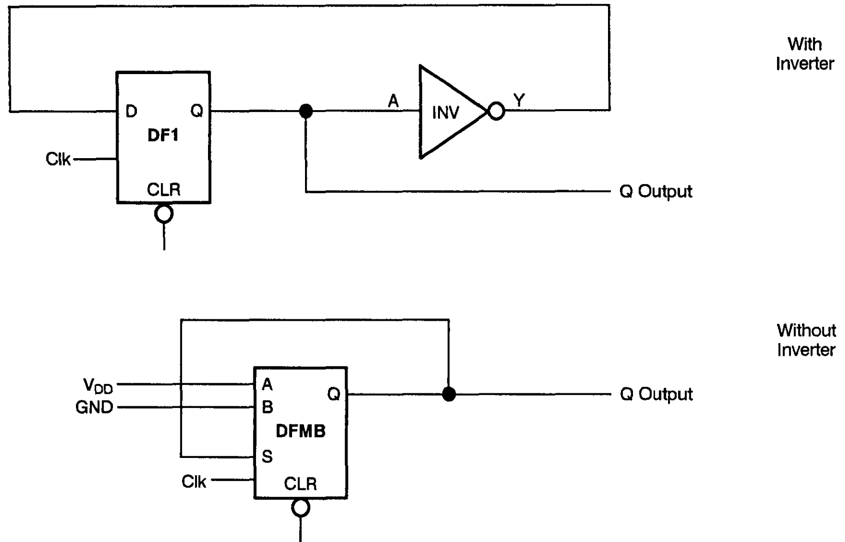


Figure 6. Creating Toggle Flip-Flops with Q Outputs

The inverter circuit can also be replaced by using the JKF JK flip-flop; however, the DFM series offers more possibilities.

The DFM flip-flop contains a multiplexor, at no additional cost, and a D flip-flop, and by connecting the inputs as shown, the need for the inverter is eliminated.

Avoiding Gated Clock Inputs

Actel does not recommend the use of gated clock inputs. Use D flip-flops with enable to emulate a gated clock configuration. Figure 7 illustrates the use of D flip-flops with enable.

Note that the use of four-input multiplexors is especially dangerous. If both select lines change simultaneously, the output almost always glitches, and false clocking of the flip-flop can occur.

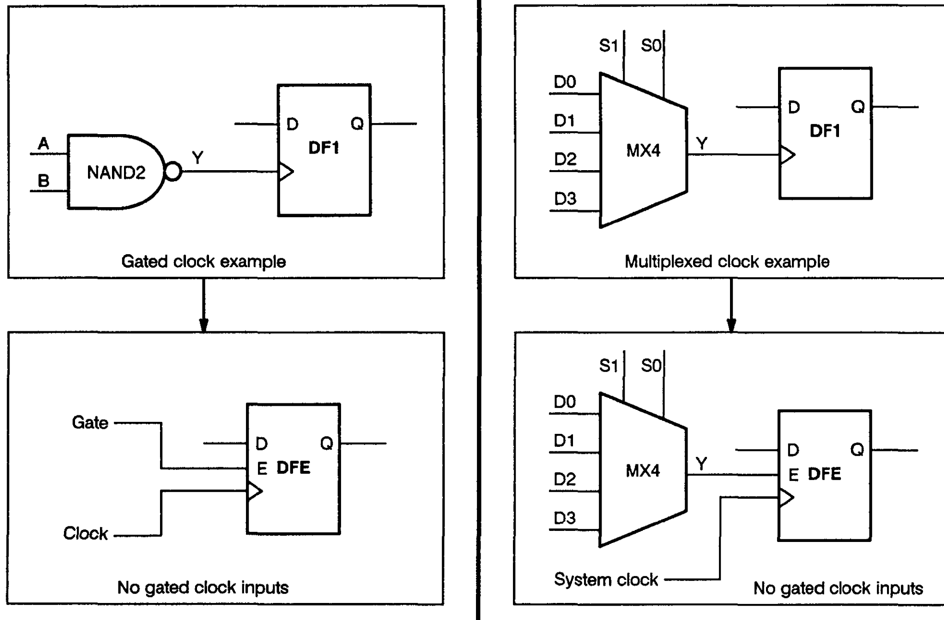


Figure 7. Avoiding Gated Clock Inputs

Important: The techniques described in this applications note make use of logical tie-offs to V_{DD} and GND. Excessive use of tie-offs should be avoided. Macro cells frequently have multiple implementations, or "mappings," that provide flexibility to the place and route software. Tie-offs restrict the number of allowable mappings.

Actel

The Actel Timer

Applications Note

Introduction

The Actel Timer is an interactive, menu-driven tool for analyzing design timing. It automatically provides delay data for all paths between clocked macros, and for any other paths specified by the designer.

The designer may review the Timer output as total path delay or break up the path to show the delay through each macro.

Setup

When first entering the Timer, the designer selects operating conditions for the device: typical, commercial, industrial, or military. The worst-case temperature and voltage specifications of each condition are shown in Table 1. The designer may also choose to inspect either post-layout or pre-layout delay data. Pre-layout delays are estimates based on statistical propagation delays and loading. Post-layout delays include placement and routing data specific to the design, and thus is the most precise timing information.

Table 1. Worst-Case Operating Conditions

Condition	Temperature (°C)	Voltage (V)
Typical	25	5.0
Commercial	+70	5.25
Industrial	+85	5.25
Military	+125	5.5

Design Analysis

The Timer automatically extracts all delays between the pins of clocked macros and places the delay information in a default pin set. This information is useful for examining delays between pins, and it provides a quick reference for the design's maximum clock frequency. In addition, the information may be manipulated easily to reveal clock line skew or any other signal skew in the design.

Pin sets can be created and edited. These pin sets can be saved and recalled; pins may be added to or removed from any pin set. Timer commands are available to logically "AND" or "OR" groups of pins to create new groups of pins.

The Timer displays delays occurring between module input pins. For example, "delay1" in Figure 1 below includes the delay of gate "G0" plus the net delay of "Net0."

The Timer provides a variety of options for design inspection and analysis. The "Worst" and "More" commands allow you to view all delays between pin sets. "Worst" shows the ten longest delays in descending order. "More" shows the next ten slowest paths, incrementally, for the entire set. The "Worst" command display appears in Figure 2. The information includes the delay rank, the delay in nanoseconds, and the path end points.

3

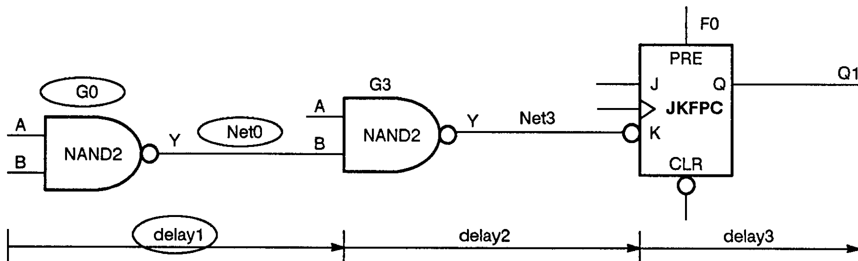


Figure 1. Input-to-Input Delay Measurement

© 1990 Actel Corporation

1st longest path to all endpin

Rank	Total	Startpin	First Net	End Net	End Pin
0	84.2	F4:CLK	NET1	NET2	F1:J
1	83.4	F17:CLK	NET3	NET4	F8:J
			•		
			•		
			•		
8	77.0	F5:CLK	NET30	NET21	F0/U0:K
9	76.1	F6:CLK	NET32	NET32	F0/U0:J

Figure 2. "Worst" Command Display

The "Path" command shows the delay along a specified path in more detail. As shown in Figure 3, "Path" shows how each module contributes to the overall delay of the path. The information displayed includes the cumulative delay, the delay through the macro, delay type, number of loads, macro name, and net name.

Using the "Path" command, the designer can pinpoint delays in a design, then fine-tune design timing. For example, the designer may pinpoint where a heavily loaded macro is creating excessive delay, then use parallel logic in the design to reduce the load and improve performance.

1st longest path to U0/U7/F0/U0:K (falling)

Total	Delay	Src	Load	Macro	Starting pin	Net name
82.1	6.5	Tsu	0	JKF_0	G12:B	NET X
75.6	7.0	Tpd	1	NAND2	G3:A	NET Y
68.8	11.8	Tpd	2	XOR	G1:A	NETA
56.8	10.8	Tpd	1	NOR2A	G14:A	NETB
46.0	10.0	Tpd	1	NOR4A	G13:C	NETC
36.0	16.4	Tpd	7	AX1	G5:C	NETD
19.6	7.1	Tpd	1	AND2A	G4:A	NETE
12.5	12.5	Tco	6	JKF_1	U1:CLK	NETF
0.0	0.0	Skw	16		U0:CLK	NETG

Figure 3. "Path" Command Display

Using Actionprobe™ Diagnostic Tools

Applications Note

Introduction

Actel's probe pin circuitry permits external monitoring of ACT™ 1 device internal signals *after* device programming. This unique diagnostic feature allows 100 percent observability of a device. Observability reduces the time required for design verification and test vector generation; it also facilitates system troubleshooting. One hundred percent observability of all internal device signals is unique to Actel field programmable gate arrays; this feature is not available in conventional masked gate arrays or programmable logic devices.

Two dedicated probe pins, PRA and PRB, provide this observability on ACT 1 family devices. Actel's Debugger software and Actionprobe™ hardware permit the connection of any two signal nodes on the device to the probe pins. Signal node assignments may be changed freely, under software control.

Setup

Actionprobe hardware consists of a towerprobe with a footprint of the selected package. A socket on top of the tower probe holds the programmed ACT device in place. The Actionprobe unit, with a programmed ACT device, can be plugged directly onto a system board. The device can be verified and debugged as it receives real-time stimuli from the system. See Figure 1.

The Actel Activator™ bus board drives the control signals SDI, DCLK, MODE, and GND via a flat ribbon cable. The MODE pin determines whether the device is in debug mode. SDI receives the serial addresses of the internal nodes from the Activator bus board. DCLK clocks the serial address into the device. When the device is being debugged in-circuit, SDI, DCLK, and MODE are terminated to ground through a > 10 kΩ resistor. Probe pins may be connected directly to a logic analyzer or oscilloscope.

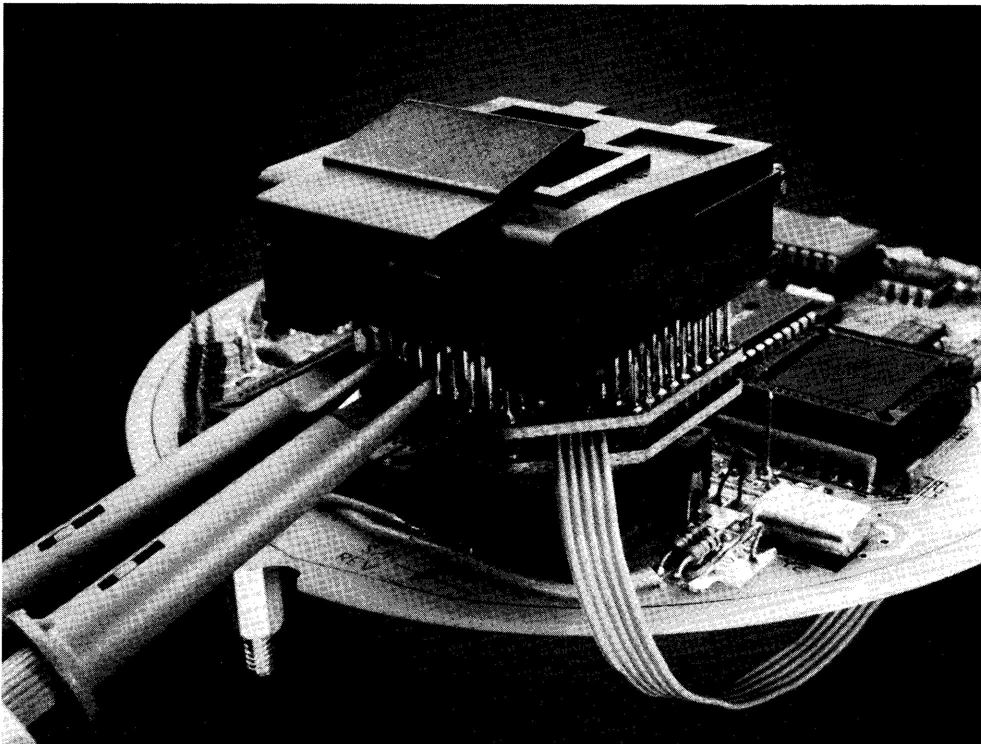


Figure 1. In-Circuit Debug Setup

© 1990 Actel Corporation

In-Circuit Probing

Changing the signal nodes is done simply by changing node names with the Debugger software. The newly assigned signals are connected automatically to the probe pins. Internal signals up to 10 MHz can be monitored externally. The internal signal passes through an inverting buffer before reaching the probe pin.

The "ICP" (In-Circuit Probing) command connects the probe pins to internal nodes. The syntax is :

ICP node_1 node_2

where "node_1" (node name) is connected electrically to PRA, and "node_2" is connected electrically to PRB.

Probe Calibration

The probe circuitry does not introduce any additional loading to the design, so the AC characteristic of the observed internal nodes remains unchanged. And, because probe propagation delay is independent of layout, probe delay remains unchanged for all points in the device.

The skew of the probe pins can be measured, then used to calibrate for accurate measurement of propagation delay. When both probe pins are assigned to the same point on the device, the delay difference measured is the skew of the probe pins. This skew is subtracted from subsequent delay measurements in the circuit. In Figure 2a, for example, both PRA and PRB are connected electrically to node Net0. The delay difference is the skew, calculated as $t_{SK} = t_{PRA} - t_{PRB}$. Using the Debugger software, the slower probe (PRA) is assigned to node Net3. Figure 2b shows this configuration. In this example, actual propagation delay is the measured delay time between the output of G0 and the output of G3, minus the probe skew time. Actual delay is calculated as: $t_{PD} = t_{PRA} - t_{PRB} - t_{SK}$

Note: Due to the propagation delay difference between rising and falling signals, both probe pin signals must be either rising or falling when they are used to calibrate for delays measurement.

Pin Assignment for Dedicated I/O Pins

During device verification, dedicated pins SDI, DCLK, PRA, and PRB are assigned as special I/O pin. These pins should be assigned as non-critical, so the design is functional without them. After device verification, these pins can be assigned as regular I/Os by programming the security fuses. This disables the probe pins to prevent unauthorized device probing.

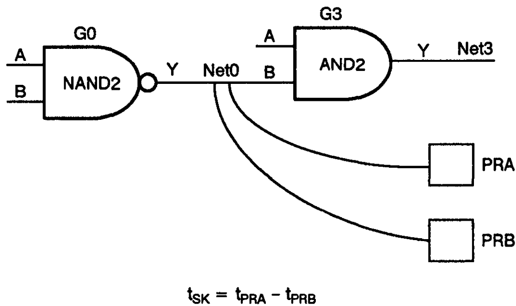


Figure 2a. Measuring Skew of Probe Pins

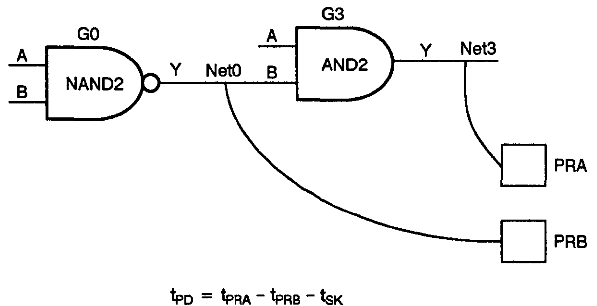


Figure 2b. Calibrating for Accurate Propagation Delay Measurement

Metastability

Applications Note

Actel Metastability Characteristics

Designers often have asynchronous signals coming into synchronous systems. Typically a flip-flop is used to synchronize the incoming signal with the system clock.

If the asynchronous incoming signal does not meet the setup time requirement for the flip-flop, there exists a window of time where the incoming signal may cause the flip-flop to develop an unknown, or metastable, logic condition. Figure 1 shows this window as t_W. Actual clock setup time of the flip-flop is shown as t_{SU}; propagation delay of the flip-flop is shown as t_{cq}. Resolution time (t_{res}) between flip-flop output and the next clocked device is the amount of time required for the metastable condition to stabilize.

The duration of a metastable condition is probabilistic. But the designer can calculate how often a metastable state will last longer than a given duration. Mean time between failures (MTBF) can be calculated from the following equation:

$$MTBF = \frac{1}{f_{clk} * f_{dat} * C1\, e^{-C2 \cdot t_{res}}}$$

where the constants depend on the ACT™ 1 device characteristics, and:

MTBF = Mean time between failures (s)

f_{clk} = System clock frequency (Hz)

f_{dat} = Incoming data rate (Hz)

e = Natural log base

t_{res} = Resolution time (ns)

C1 = 10^{-9}/Hz

C2 = 4.6052/ns

(Value of C2 derived from circuit simulations.)

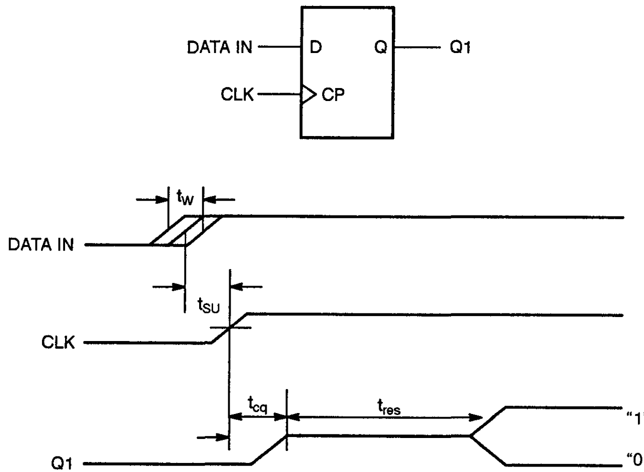


Figure 1. Metastable Condition

3

© 1990 Actel Corporation

Sample Calculation

Using the MTBF equation for a design with a system clock frequency of 10 MHz and a data rate of 1 MHz, various resolution times produce the results shown in Figure 2. Linear increases in resolution time cause exponential increases in MTBF.

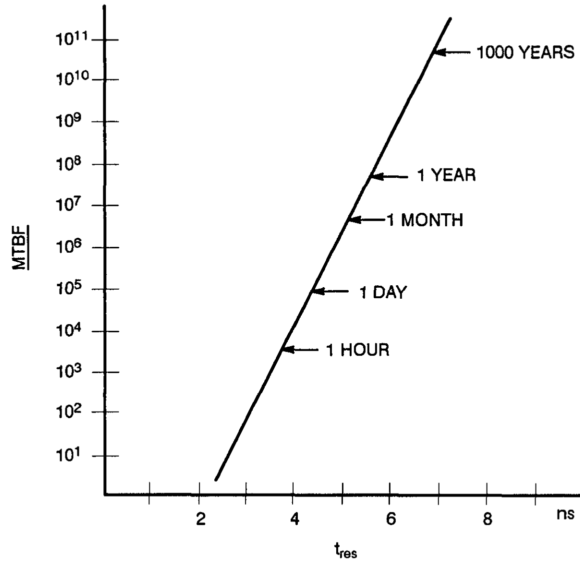


Figure 2. Metastable MTBF as a Function of Resolution Time

Using the Actel Debugger as a Functional Tester

Applications Note

Introduction

Actel's Activator™ programming and diagnostics unit, together with Actel's Debugger software, provide powerful tools to functionally test an ACT™ device. Device debugging begins after design configuration and device programming. Debugging is performed with the device inserted in the Activator unit. The user accesses Debugger software from the Action Logic™ System (ALS) main menu.

Debugger functional test allows the user to observe any internal node of the device. The user defines the device inputs with Debugger menu commands or with a command file, or any combination of the two. Command files and test vector files are created with an ASCII text editor, then loaded into the Debugger. User-defined macros may be created in a command file, then executed in the Debugger.

This applications note shows Debugger commands for a sample design. The sample design is Actel's TT269 (TTL 74269), an eight-bit binary loadable up/down counter with count enable. Figure 1 shows the sample design; Table 1 shows the truth table for the part. P0 through P7 are parallel load inputs; Q0 through Q7 are counter outputs; CLK is the counter clock; UD is the up/down count selector. Internal nodes (nets) should be labeled during design capture for easy reference during debugging.

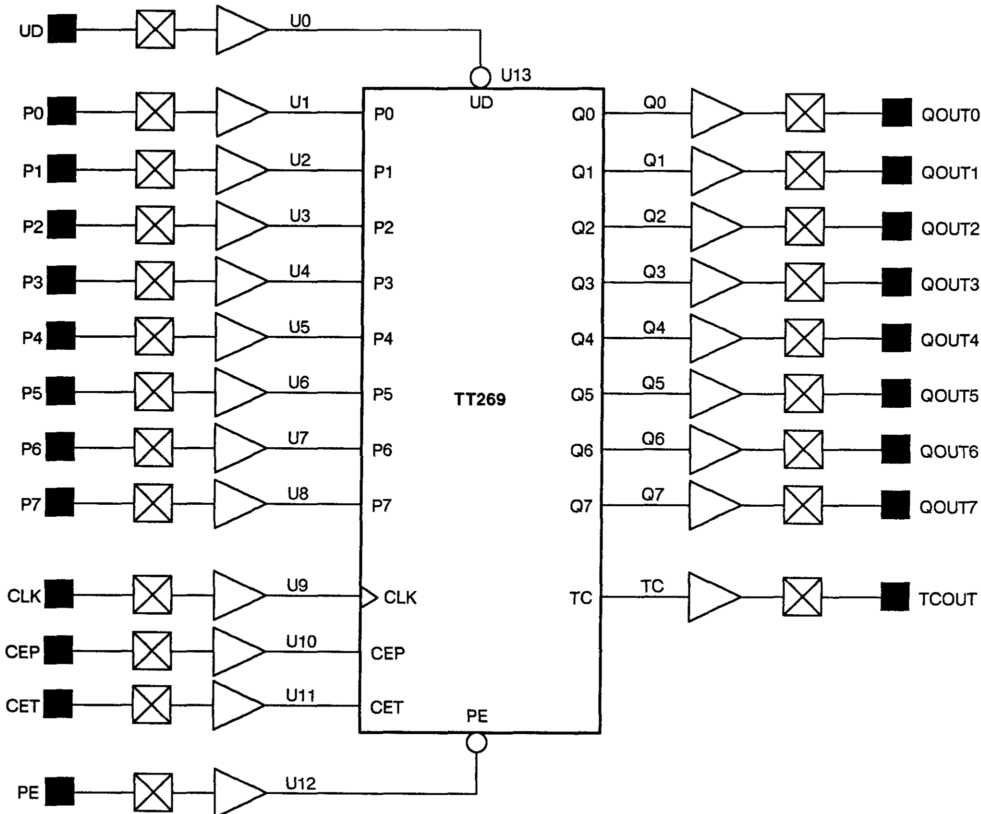


Figure 1. TT269

© 1990 Actel Corporation

Table 1. TT269 Truth Table

Inputs						Outputs	
PE	CEP	CET	UD	P[7:0]	CLK	Q[7:0]	TC
0	X	X	X	0	↑	0	1
0	X	X	X	FF	↑	FF	0
1	X	1	X	X	↑	Hold	
1	1	X	X	X	↑	Hold	
1	0	0	1	X	↑	Increment	
1	0	0	0	X	↑	Decrement	

Assigning Test Vectors

The default input radix for all test vectors is decimal. To specify a binary, hex, or octal radix, add a 0b, 0h, or 0o prefix, respectively, to the vector (e.g., 0b1010 or 0h7e). Outputs are in binary format. To interactively define input test vectors, use the Debugger menu. Alternatively, use input command files to define test vectors. To view outputs and internal nodes, print them to the PC screen display or to an output file.

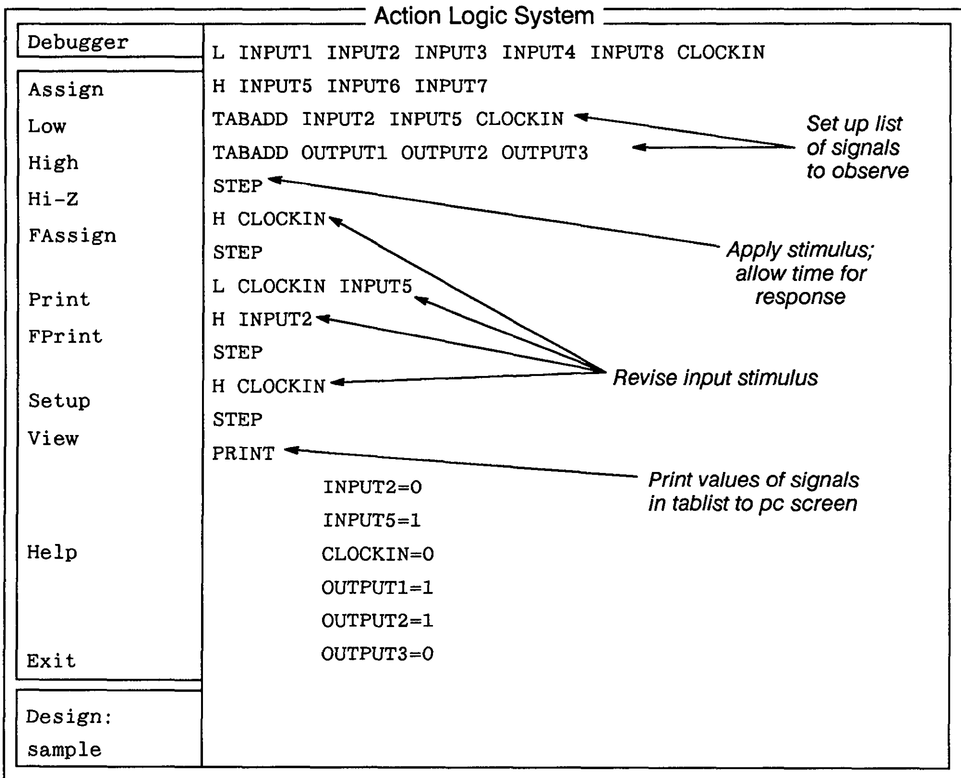


Figure 2. Typical Debug Command Sequence

Defining User Macros

You may save time by creating user-defined macros for the Debugger. These macros may contain a series of basic Debug commands or may nest any combination of basic commands and user-defined macros.

The sample macro in part A of Figure 3, clk10, provides 10 clock pulses to the pin CLK and prints the value of internal vector Q to a specified output file after each clock pulse. The OUTFILE command specifies the output file.

Part B of Figure 3 shows a nested macro, clk100, which executes the clk10 macro 10 times, providing 100 clocks to the CLK pin.

A
```
define (clk10) (repeat 10 (l CLK) (step) (h CLK) (step) (fprint Q))
```

B
```
define (clk100) (repeat 10 (clk10))
```

Figure 3. Sample Command Macros

Creating a Command File (TT269.cmd)

The command file (see Figure 4) applies test vectors to the TT269 counter. It redirects results to an output file, TT269.out and compares the output vector Q of the counter to an existing results file, TT269.cmp. The following notes correspond to each line in the command file.

Lines 1 and 2: The vector command defines eight parallel load input bits as vector P, and counter output as vector Q.

Line 3: The tabadd command defines the internal or external nodes to be displayed or printed when the print or fprint command is executed.

Lines 4, 5, and 6: The infile, outfile, and compfile commands define input and output files. infile opens a file containing input test vectors. outfile contains the output results. compfile contains the data to be compared against the current device status. Use the full path name of the file, and enclose it in question marks.

Line 7: The define commands create user-defined macros. In this example, the clk10 macro provides 10 clock pulses to the CLK input, fprint prints all nodes in the tabadd command to a file defined by outfile. fcomp compares the status of vector Q to the file specified by compfile.

Lines 8, 9, and 10: Defines three user macros: up, down, and load.

Loading a Command File

The loadfile command loads a defined command file into the Debugger. Select setup | loadfile from the Debugger menu, then enter the full path name of the command file. For example:

```
/designs/TT269/TT269.cmd
```

Executing User-Defined Macros

To execute any previously defined macro, type the macro at the command line while in the Debugger.

```
Line
  1      (vector P P0 P1 P2 P3 P4 P5 P6 P7)
  2      (vector Q Q0 Q1 Q2 Q3 Q4 Q5 Q6 Q7)
  3      (tabadd PE CEP CET UD P CLK Q TC)
  4      (infile "/designs/tt269/tt269.pat")
  5      (outfile "/designs/tt269/tt269.out")
  6      (compfile "/designs/tt269/tt269.cmp")
  7      (define (clk10) (repeat 10 (l CLK)(step)(h CLK)(step)(fprint)(fcomp Q)))
  8      (define (up) (l CEP CET)(h PE UD)(step)(clk10))
  9      (define (down) (h PE)(l CEP CET UD)(step)(clk10))
 10      (define (load) (l PE CLK)(step)(fassign P)(h CLK)(step))
```

Figure 4. Debug Command File (TT269.cmd)

Running the Sample Command File

In the following example, we will assign input test vectors from an input pattern file named TT269.pat, and write output results to a file named TT269.out. The command file (TT269.cmd) compares the counter's output vector Q to an existing results file, TT269.cmp.

Output Results File (TT269.out)

```
 S     P C C U         P C         Q T
 T     E E E D           L           C
 E       P T             K
 P
00005: 1 0 0 1 00000000 1 10000000 0
00007: 1 0 0 1 00000000 1 01000000 0
00009: 1 0 0 1 00000000 1 11000000 0
00011: 1 0 0 1 00000000 1 00100000 0
00013: 1 0 0 1 00000000 1 10100000 0
00015: 1 0 0 1 00000000 1 01100000 0
00017: 1 0 0 1 00000000 1 11100000 0
00019: 1 0 0 1 00000000 1 00010000 0
00021: 1 0 0 1 00000000 1 10010000 0
00023: 1 0 0 1 00000000 1 01010000 0
00028: 1 0 0 0 10000000 1 00000000 1
00030: 1 0 0 0 10000000 1 11111111 0
00032: 1 0 0 0 10000000 1 01111111 0
00034: 1 0 0 0 10000000 1 10111111 0
00036: 1 0 0 0 10000000 1 00111111 0
00038: 1 0 0 0 10000000 1 11011111 0
00040: 1 0 0 0 10000000 1 01011111 0
00042: 1 0 0 0 10000000 1 10011111 0
00044: 1 0 0 0 10000000 1 00011111 0
00046: 1 0 0 0 10000000 1 11101111 0
00051: 1 0 0 1 01000000 1 11000000 0
00053: 1 0 0 1 01000000 1 00100000 0
00055: 1 0 0 1 01000000 1 10100000 0
00057: 1 0 0 1 01000000 1 01100000 0
00059: 1 0 0 1 01000000 1 11100000 0
00061: 1 0 0 1 01000000 1 00010000 0
00063: 1 0 0 1 01000000 1 10010000 0
00065: 1 0 0 1 01000000 1 01010000 0
00067: 1 0 0 1 01000000 1 11010000 0
00069: 1 0 0 1 01000000 1 00110000 0
00074: 1 0 0 0 11000000 1 01000000 0
00076: 1 0 0 0 11000000 1 10000000 0
00078: 1 0 0 0 11000000 1 00000000 1
00080: 1 0 0 0 11000000 1 11111111 0
00082: 1 0 0 0 11000000 1 01111111 0
00084: 1 0 0 0 11000000 1 10111111 0
00086: 1 0 0 0 11000000 1 00111111 0
00088: 1 0 0 0 11000000 1 11011111 0
00090: 1 0 0 0 11000000 1 01011111 0
00092: 1 0 0 0 11000000 1 10011111 0
```

Input Pattern File (TT269.pat)

```
0b00000000
0b10000000
0b01000000
0b11000000
```

Output Compare File (TT269.cmp)

```
0b10000000
0b01000000
0b11000000
0b00100000
0b10100000
0b01100000
0b11100000
0b00010000
0b10010000
0b01010000
0b00000000
0b11111111
0b01111111
0b10111111
0b00111111
0b11011111
0b01011111
0b10011111
0b00011111
0b11101111
0b11000000
0b00100000
0b10100000
0b01100000
0b11100000
0b00010000
0b10010000
0b01010000
0b11010000
0b00110000
0b01000000
0b10000000
0b00000000
0b11111111
0b01111111
0b10111111
0b00111111
0b11011111
0b01011111
0b10011111
```

Calculating Power in Actel's ACT™ 1 Arrays

Applications Note

During the design of any system, the question, "How much power will the device use?" always comes up. This is especially important in battery-operated or other systems with limited available power. This applications note will help you answer the power consumption question.

Actel's ACT™ 1 array is CMOS-based, and so draws very little I_{DD} current. The amount of power supply current drawn is directly related to the input clock frequency and the amount of logic switching. Use the following formula for typical dynamic power dissipation.

Note that power is calculated for a typical interconnect length and a fan-out of three for each of the logic modules used in the design.

Total Chip Power (mW) = 0.41*N*F1 + 0.17*M*F2 + 1.62*P*F3

Where:

- F1 = Average logic module switching rate in MHz.
- F2 = Average clock pin switching rate in MHz.
- F3 = Average I/O switching rate in MHz.
- M = Number of logic modules connected to the clock pin.
- N = Total number of logic modules used on the chip. (including M)
- P = Number of I/O pairs used (input + output) with 50 pF of output loading.

F1 term example: If a 4-bit ripple counter has a 10 MHz clock input, the four outputs would be at 5, 2.5, 1.25, and 0.625 MHz, with the average frequency of 2.1875 MHz ((5 - 0.625) / 2) used for F1.

F2 is associated with the CLKBUF high fan-out clock buffer. F2 and M can be deleted if you are not using the CLKBUF macro in your design.

F3 is the average switching rate for all the I/Os in the design.

N is the total number of logic modules used in the design, including those that are connected to the CLKBUF macro.

Example of Dynamic Power Calculation

If an ACT 1010 design used 200 logic modules of which 40 were connected to the high fan-out clock buffer running at 20 MHz, and the rest were running at 4 MHz, plus 50 I/Os (25 outputs, 25 inputs) running at an average of 4 MHz, it would dissipate the following amount of power:

Total Chip Power = 0.41N*F1 + 0.17M*F2 + 1.62P*F3
= 0.41(200*4) + 0.17(40*20) + 1.62(25*4)
= 503.6 mW

Power Dissipation Characteristics

The graphs in Figure 1 illustrate the power dissipation for both the ACT 1010 and ACT 1020 devices as the operating frequency and device utilization increases. These graphs assume that the logic modules and the I/Os are operating at 1/4 the input frequency, and the CLKBUF macro is used. The percent utilization applies to both the I/Os and the number of logic modules.

Package Limitations

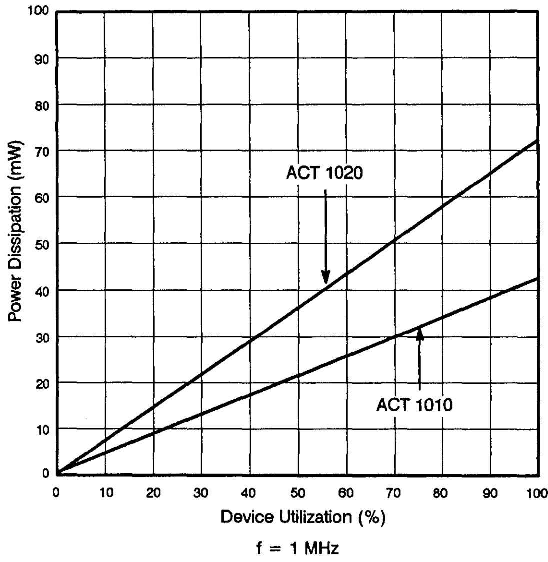
The power dissipation values given in Figure 1 do not take into account the power handling limitations of the package. The following maximums apply:

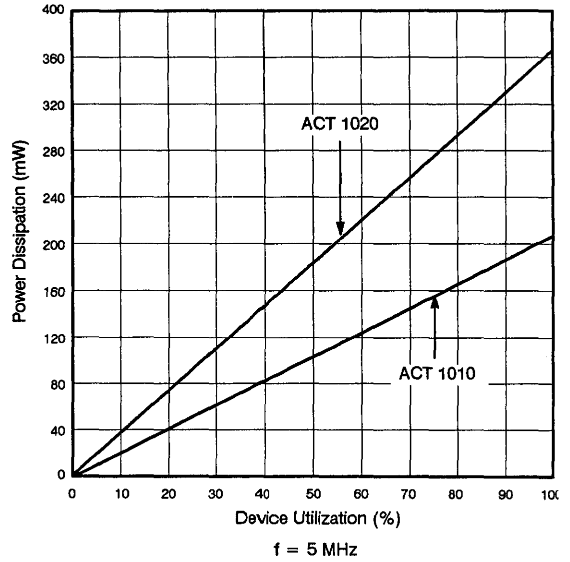
Plastic J lead packages = 1.0 watts
Ceramic J lead packages = 1.5 watts
Ceramic pin grid array = 2.0 watts

Quiescent Current

Quiescent, or standby, current ranges from 3 mA, typical, to 10 mA, maximum. A voltage pump on the device runs constantly and consumes power. All power dissipation calculations should include standby current.

© 1990 Actel Corporation





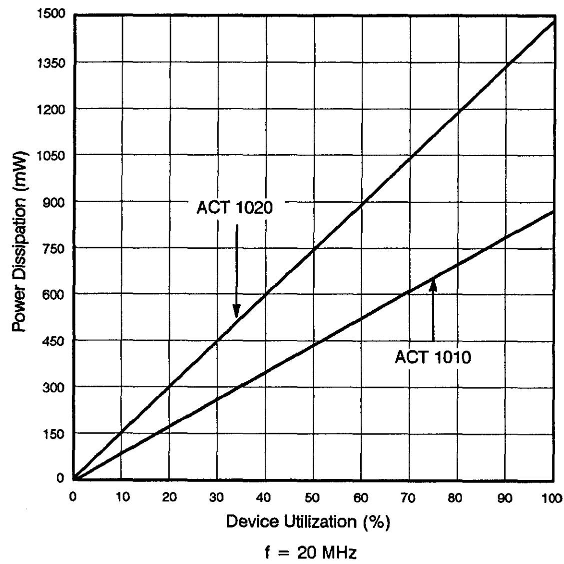


Figure 1. Power Dissipation Characteristics

Voltage and Temperature Derating

When the power supply or the ambient temperature deviates from typical, derating factors must be applied to the total power dissipation of the device.

System power supply derating factor:

V_{DD} (V)	4.50	4.75	5.00	5.25	5.50
Derating Factor:	0.87	0.94	1.00	1.07	1.13

Ambient temperature derating factor:

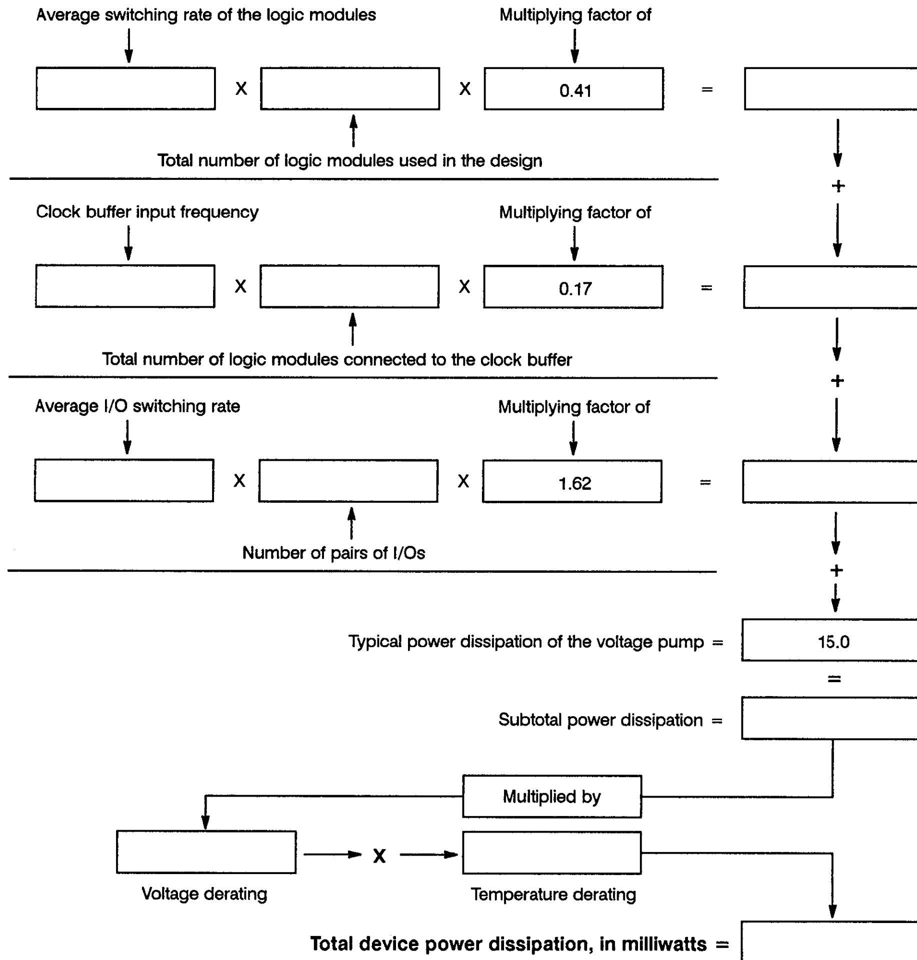
Temperature (°C)	-55	0	25	70	125
Derating Factor	0.98	0.99	1.00	1.01	1.03

In the example design, power dissipation was calculated as 503.6 milliwatts at 25°C and 5.0 V. If the voltage were taken down to 4.5 volts and the temperature elevated to 70°C, the example would then dissipate the following amount of power:

$$503.6 \text{ X } 0.87 \text{ X } 1.01 = 442.5 \text{ mW.}$$

An additional 15 milliwatts of power (nominal conditions) should be added to account for standby current. This number is also derated by the factors given for voltage and temperature.

ACT 1 Power Calculation Worksheet



Three-Stating ACT™ 1010/1020 Designs

Applications Note

During board test and debug it is frequently desirable to place all chip I/Os into a three-stated condition. This provides isolation from other three-stating circuit devices connected to signals common to the ACT™ device. The three-stated condition also allows board test for trace integrity or insertion damage to ACT device pins.

Three-stating a design is easy using the unique debug features of ACT device architecture. Three special pins on the ACT device facilitate three-stating: MODE, SDI, and DCLK.

Three-State Pin Assignments

	PLCC/JQCC			PGA
	44-pin	68-pin	84-pin	84-pin
MODE	34	54	66	E11
SDI	36	56	72	B11
DCLK	37	57	73	C10

Pins SDI and DCLK should remain unassigned by the user or should be defined as input-only.

You may three-state all user-defined pins regardless of their normal mode definition: input-only, output-only, or three-stateable. Each pin can be temporarily three-stated for test and debug purposes.

Figure 1 shows the sequence of three-stating. Seven data bits are clocked into the device, using the SDI pin as data input, and DCLK as clock. The MODE pin distinguishes "test" mode from "normal." The data sequence clocked is {0001011}. After clocking the seventh bit, all user-defined pins enter a three-state condition until MODE is taken low.

Actel Actionprobe™ diagnostic tools allow 100% observability of internal nodes; ACT devices may also be isolated from external board circuitry. Together, these two features provide a powerful debugging feature previously unavailable in custom devices.

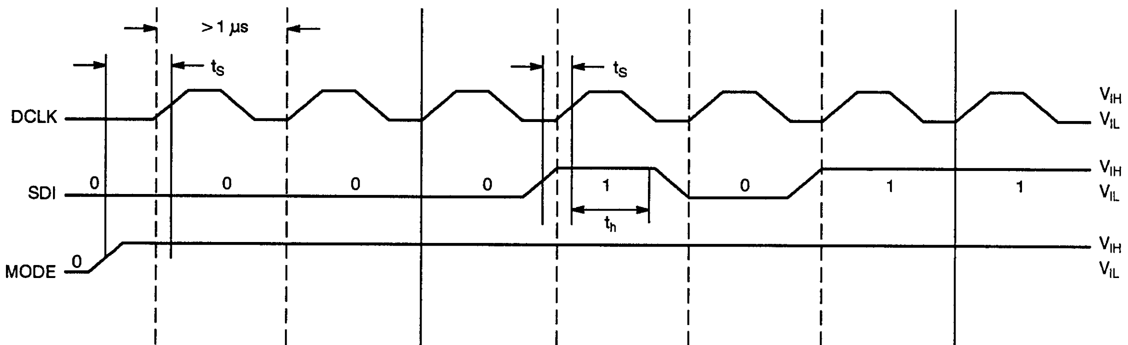


Figure 1. Three-state Timing

Notes:

1. $0 \text{ V} \leq V_{IL} \leq 0.5 \text{ V}$; $3.0 \text{ V} \leq V_{IH} \leq V_{CC}$
2. Test mode configuration is a low frequency (< 1.0 MHz) operation.
3. All setup and hold conditions (t_h, t_s) ≥ 250 ns.

3

© 1990 Actel Corporation

Multi-ASIC and System Simulation with Actel FPGAs

Applications Note

Multiple ASIC and board-level design simulation produces a dilemma: while individual simulation of component ASIC designs is possible, there is no easy system-level simulation process for designs consisting of ASICs and other component types such as TTL.

Actel offers complete timing simulation using Viewlogic® Viewsim® for board-level designs with an unlimited number of Actel field programmable gate arrays (FPGAs).

The technique is as follows:

- Design and simulate timing for each FPGA. This process includes Viewsim functional simulation, place and route using Actel's Action Logic™ System (ALS) software, and back-annotated timing simulation.
- From a DOS window in the \WORKVIEW directory, execute the following command for each FPGA design:

```
VSM YOURCHIP -W -DYOURCHIP.DTB
```

The command causes the Viewsim compiler to create a flat WIR file for the design to the Builtin or Viewlogic simulation primitive level. The .WIR file includes delay information from the file YOURCHIP.DTB.

- Copy each .WIR file to the WIR subdirectory. The VIEWDRAW.INI file specifies the location of the WIR subdirectory (on the line beginning with "DIR").
- Modify each .WIR file (using non-document mode), adding the lines:

```
G GND
G VDD
```

between the last AP/AS line and the first M line in the .wir file, as shown in Figure 1.

```
•
•
•
AP MUX41 PINTYPE=IN
APMUX41 7 PINTYPE=IN
AP MUX41 8 PINTYPE=IN
|Message: 100 component, 745 pin, and 0 net records found in table
"memtiming".
AS DELAY PINORDER=OUT IN
AP DELAY 1 PINTYPE=OUT
AP DELAY 2 PINTYPE=IN
|Add the following 2 lines between the AP/AS lines and the first M line.
G GND
G VDD
|The first M line...
M DELAY?
I ? DELAY $1172/D $IN133 DELAYMOD=TRANSPORT TPLH=100'TPHL=109
M DELAY ?
I ? DELAY $1I45\$1I29\C $1N133 DELAYMOD=TRANSPORT
TPLH=120'TPHL=135'
M DELAY ?
I ? DELAY $1I45\$1N133 DELAYMOD=TRANSPORT TPLH=101'TPHL=127'
•
•
•
```

Figure 1. Modifying the .WIR File

© 1990 Actel Corporation

- Edit the file \WORKVIEW\VIEWDRAW.INI to comment out the Actel library pointer and to add pointers for the other libraries to be used in the design. An example is shown in Figure 2 below.
- Create symbols for each Actel FPGA in Workview®.
- Capture the system-level schematics and simulate the design.

VIEWDRAW.INI (OLD)		VIEWDRAW.INI (NEW)
. . . DIR 0 C: \DESIGNS DIR 6 C: \ACTVIEW DIR 7 C: \WORKVIEW\BUILTIN . . .	becomes	. . . DIR 0 C: \DESIGNS | DIR 6 C:\ACTVIEW DIR 7 C: \WORKVIEW\BUILTIN DIR 11 C:\WORKVIEW\74LS . . .

Figure 2. Modifying the VIEWDRAW.INI File

Constructing RAM with ACT™ 1 Macros

Applications Note

Introduction

This applications note illustrates an efficient way to construct RAM of varying width and depth on ACT™ 1 field programmable gate arrays. Actel has created a number of soft macros for this purpose. Below is a listing of macros available for RAM construction. Details on these soft macros can be found in the Actel Macro Library.

Soft Macro	Function
DEC4	2 Line Decoder with Write Control
DEC8	3 Line Decoder with Write Control
DEC12	4 Line Decoder with Write Control
DEC16	4 Line Decoder with Write Control
RAM4	4 Bit Deep by 1 Bit Wide RAM Macro
SRAM4	4 Bit Deep by 1 Bit Wide Synchronous RAM Macro
DPRAM4	4 Bit Deep by 1 Bit Wide Dual Port RAM Macro
SDPRAM4	4 Bit Deep by 1 Bit Wide Synchronous Dual Port RAM Macro
RAM8	8 Bit Deep by 1 Bit Wide RAM Macro
SRAM8	8 Bit Deep by 1 Bit Wide Synchronous RAM Macro
DPRAM8	8 Bit Deep by 1 Bit Wide Dual Port RAM Macro
SDPRAM8	8 Bit Deep by 1 Bit Wide Synchronous Dual Port RAM Macro
RAM4X4	4 Bit Deep by 4 Bit Wide RAM Macro with Separate Read and Write Select
RAM4X4A	4 Bit Deep by 4 Bit Wide RAM Macro with Common Read and Write Select
RAM4X9	4 Bit Deep by 9 Bit Wide RAM Macro with Separate Read and Write Select
RAM4X9A	4 Bit Deep by 9 Bit Wide RAM Macro with Common Read and Write Select
RAM8X8	8 Bit Deep by 8 Bit Wide RAM Macro with Separate Read and Write Select
RAM8X8A	8 Bit Deep by 8 Bit Wide RAM Macro with Common Read and Write Select

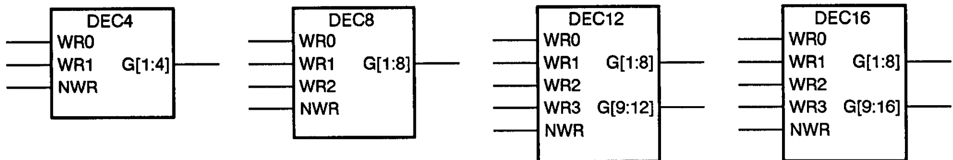
The soft macros can be used individually or combined to create RAM of any desired width and depth. The last six macros on the above list are complete RAM functions, containing address decoders, write control, and a reset. They can be used independently of the other RAM macros, or they may be combined as needed.

RAM macros are constructed from transparent data latches, therefore the outputs will follow the inputs for as long as the write line is held true. This minimizes the silicon area required for each RAM function.

© 1990 Actel Corporation

Decoder Functions

The DEC4, DEC8, DEC12, and DEC16 decoders have been created to interface easily to Actel RAM macros. Each decoder contains the address inputs, a low-true write input, and the RAM bit select outputs.



Inputs			Outputs
WR0	**WR1**	**NWR**	**G[1:4]**
X	X	1	0000
0	0	0	1000
1	0	0	0100
0	1	0	0010
1	1	0	0001

Sample Truth Table for the DEC4 Macro

Figure 1a. Decoder Macros

Figure 1a illustrates the four DECx macros, and shows a sample truth table for the DEC4 macro. Note that the Gx outputs are selected only when the write input, NWR, is at a logic zero. The NWR write pulse must occur only while the address, determined by the WRx signals, is valid. All of the DECx macros function similarly.

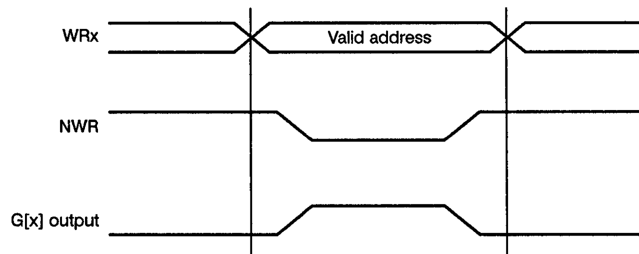


Figure 1b. Decoder Macro Timing

Figure 1b illustrates the timing requirements for the DECx macros. Soft macro timing is extracted after place and route of the design.

RAM Functions

All of the RAM building blocks are organized as 1-bit wide by either 4 or 8 bits deep. These macros are "stacked" to create RAM of the desired width. In addition, single or dual port, and asynchronous or synchronous versions are available, as shown below:

Depth	Asynchronous		Synchronous	
	Single Port	Dual Port	Single Port	Dual Port
4 Bits	RAM4	DPRAM4	SRAM4	SDPRAM4
8 Bits	RAM8	DPRAM8	SRAM8	SDPRAM8

The RAM4 and RAM8 macros have a reset input as an additional feature.

Figure 2 illustrates the schematic diagram and symbol for the SDPRAM4 macro. Note that every input presents a single unit load to the driving source. This is true for every input on all RAM macros. The G[x] inputs enable the clocking source to load data into the latch whenever both the CLK and appropriate G[x] inputs are at a logic 1. Multiple bits can be written to by controlling the G[x] inputs; however, only one latch output can be read at a time.

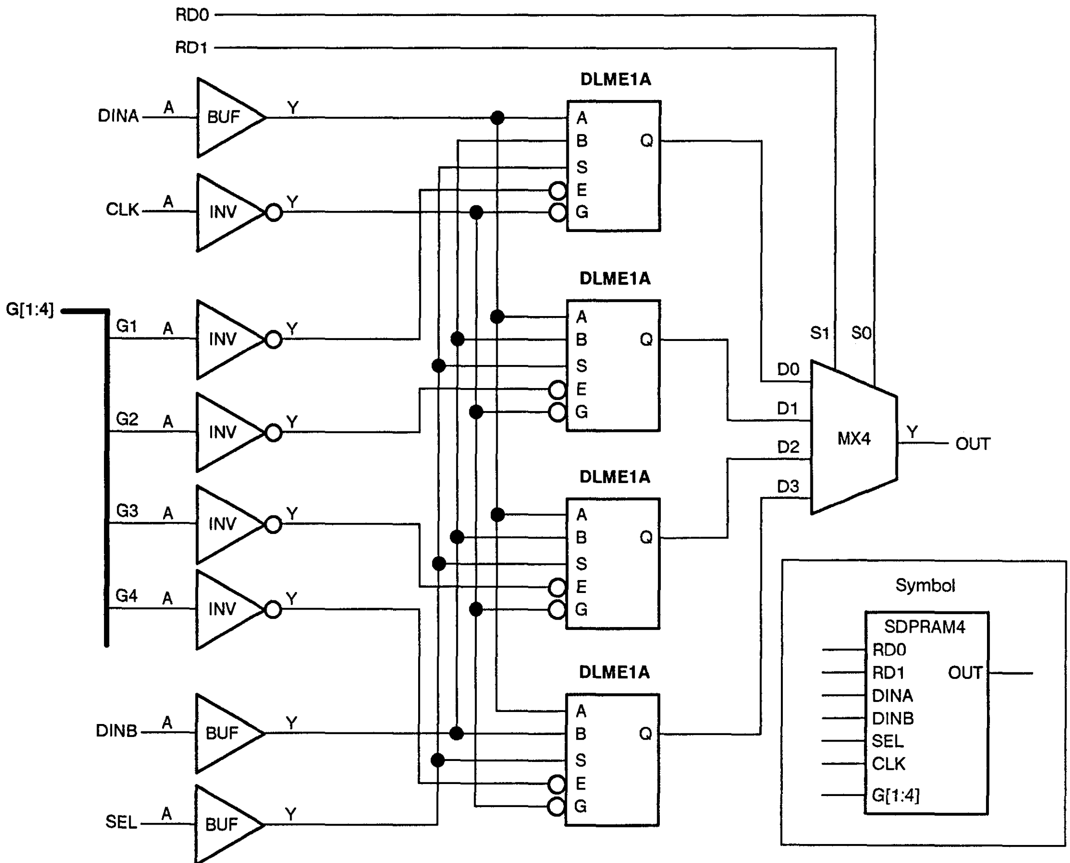


Figure 2. SDPRAM4 Macro Schematic

3

Constructing RAM Modules

By combining RAM soft macros, complete RAM modules can be constructed. Figure 3 illustrates the RAM4X4 function as constructed from Actel RAM and decoder soft macros.

Each RAM4 macro is 4 bits deep. Using four RAM4 macros as shown creates a 4-bit wide, 4-bit deep RAM module. Note that the DEC4 macro can drive up to 24 RAM4 macros. A RAM module of any size can be constructed in similar fashion.

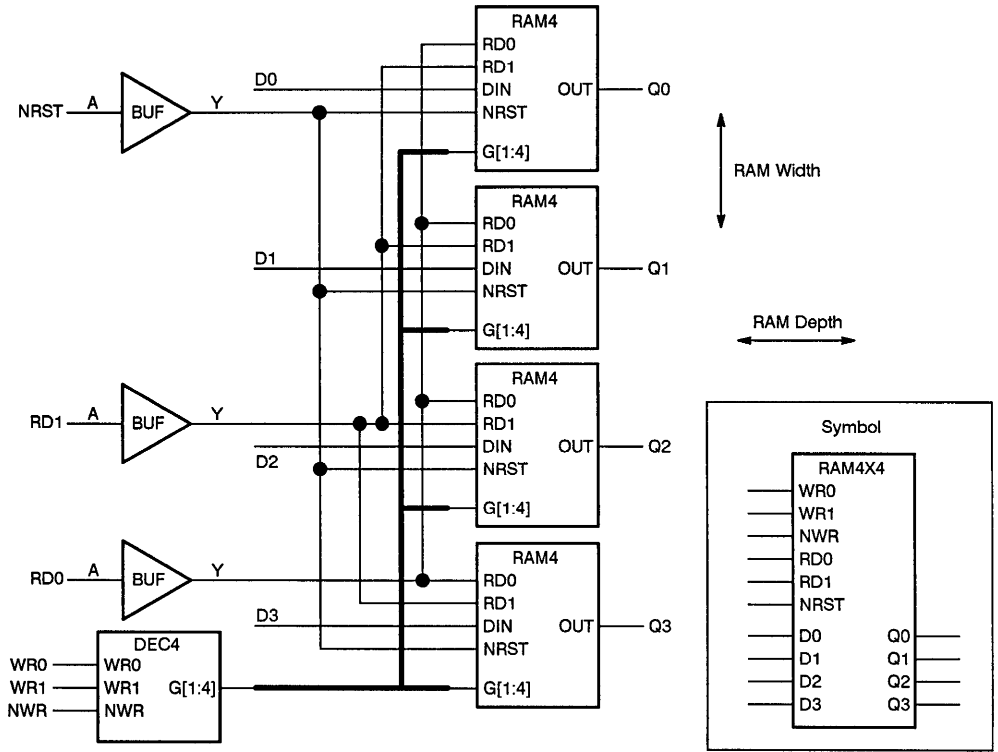


Figure 3. RAM Module Example

Constructing Deep RAM Modules

The DEC12 and DEC16 four line decoder macros allow the construction of RAM modules of 12 or 16 bits deep, respectively.

Figure 4 illustrates a 12-bit deep RAM module constructed from RAM and decoder macros.

The same technique applies to 16-bit deep RAM modules, using the DEC16 macro and two RAM8 macros.

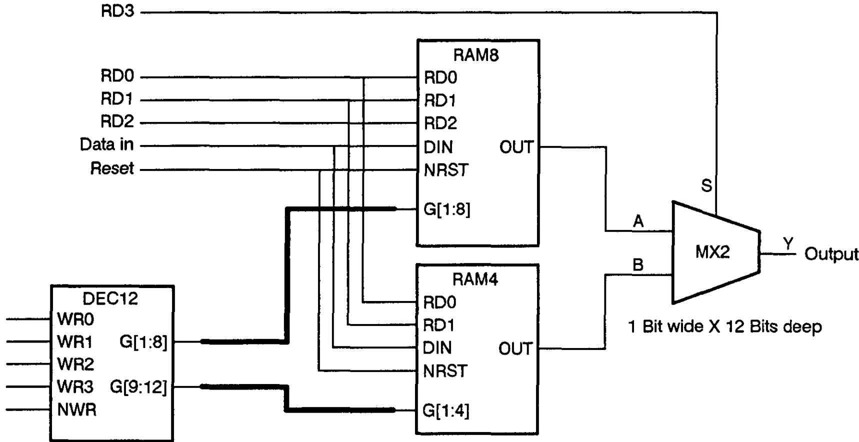


Figure 4. Deep RAM Module Example

All common inputs are tied together on the macros, completing a 1-bit wide by 12- or 16-bit deep RAM module. Up to 24 RAM macros can be stacked without additional buffering.

Actel

Using the DRAM Controller Supermacro

Applications Note

Introduction

Processor-based systems require control logic for DRAM accesses and refresh. System designers can use a VLSI DRAM controller chip with a built-in processor interface. The chips are easy to use, but slow, and they operate in a limited number of configurations.

One alternative is to build a controller out of discrete logic, including PAL®, delay lines, and standard components. Such a controller can be customized to the system, but it requires several components with the attendant assembly, board space, and failure rate costs.

Another way to implement the controller is to use a masked gate array. Although this provides a single-chip custom solution, large NRE costs and lengthy delays are inherent problems in developing masked gate arrays.

The ideal solution is to use an Actel field programmable gate array (FPGA.) The Actel library contains a DRAM controller macro that may be customized for the target system and then combined with other logic required to implement the system.

This note explains the operation of the macro and how to use it in developing a system. The design has been tested at a clock rate of 16 MHz and can operate easily with both 120 ns and 100 ns DRAMs.

The sample implementation has two address channel inputs and decodes two address bits to four RAS drivers. It has no byte decoding so that it addresses four megabytes on a 32-bit boundary.

Architecture

The architecture of the controller is shown in Figure 1. The three major blocks are the Refresh Timer, Memory Access Arbitrator, and Memory Timing Control.

The refresh timer generates refresh requests on intervals set by decoding counter outputs. The arbitrator is a state machine that prioritizes memory requests and sends signals to the timing control to initialize memory cycles. The timing controller asserts the DRAM and address multiplexor control lines at the appropriate times for the cycle being executed. A detailed operational description follows.

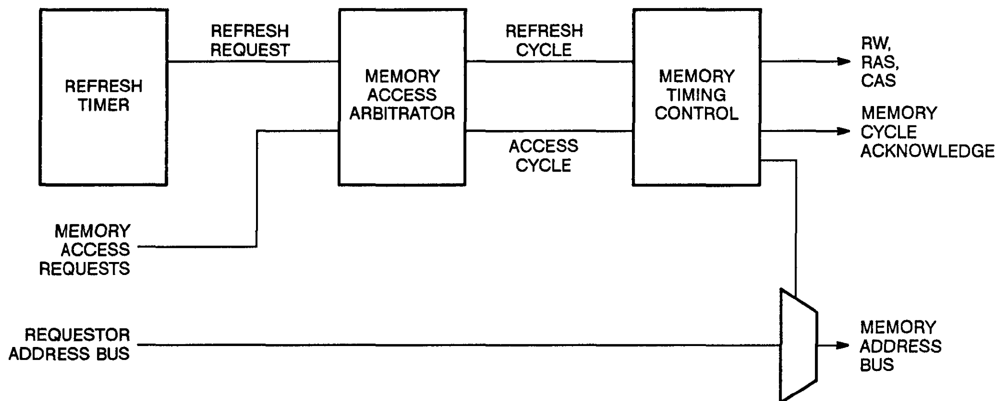


Figure 1. DRAM Controller Supermacro Architecture

© [illegible] Actel Corporation

Refresh Timer

The refresh timer consists of two counters, a five-bit binary counter, and a three-bit Gray counter. The binary counter generates a pulse every two microseconds. Its carry-out is used to enable the three-bit Gray counter. The Gray counter outputs are decoded to set the refresh request flip-flop and to reset the counters. The Gray counter is used to avoid race conditions on state decodes, which can cause spurious refresh requests.

The decode selected from the Gray counter determines the period of refresh requests. The macro's range is from 2 to 14 microseconds. Longer periods could be achieved by adding a bit to the binary counter with a loss of granularity in selecting the refresh period.

The refresh request flip-flop remains set until the arbitrator begins a refresh cycle. The memory controller uses CAS-before-RAS refresh so no refresh address counter is required.

Arbitrator

The arbitrator accepts requests for memory cycles from the system processor, I/O channel, or the refresh flip-flop. If the memory is idle, the arbitrator goes to the state appropriate for the requestor and proceeds through the states of the cycle as shown in Figure 2.

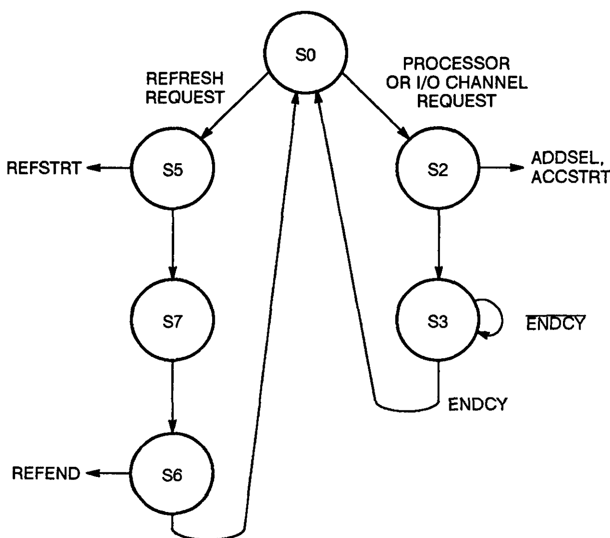


Figure 2. Memory Access Arbitrator State Graph

If the memory is not idle, the new request will wait until the current cycle is complete. The processor has priority over the I/O channel, and the channel has priority over the refresh. Processor or I/O channel cycles begin by setting a flip-flop whose output (ADDSEL) selects the appropriate address bus and asserting a signal (ACCSTRT) to the timer to start the memory access. Refresh cycles assert a signal (REFSTRT) that begins the refresh cycle.

DRAM precharge timing requirements are not violated by adjacent memory cycles because RAS is turned off prior to the end of a cycle. After that, the arbitrator returns to the idle state (S0) before it can begin another cycle and reassert RAS. At a clock frequency of 16 MHz, this pattern ensures a precharge time of at least 90 ns. Faster clocks could require unused states to be employed as timing buffers.

Memory Timing Control

The timing controller begins memory cycles when a pulse is received from the arbitrator.

A refresh cycle causes the CAS signal to be asserted first and then all the RAS lines a half cycle later. The refresh cycle is ended by the arbitrator reaching a state and issuing a pulse (REFEND).

Processor or I/O memory cycles begin by asserting the RAS line that is decoded from the two most significant address bits. The following clock sets a flip-flop that causes the address multiplexor to switch from the RAS address bits to the CAS. The multiplexor control then sets a flip-flop to cause the CAS line to be set a short time later.

The multiplexor line also goes to a flip-flop that will be set on the next clock and causes both the arbitrator and the DRAM control flip-flops to end the memory cycle.

The timing controller may require more customization than the other modules in the macro to integrate it into other systems. Faster clocks require timing adjustments to the memory control signals. These adjustments can be made with additional flip-flops, clocked on either edge of the clock, to minimize the length of the cycle without violating any timing constraints.

Using the DMA Controller Supermacro

Applications Note

Introduction

DMA controllers are used to supply main memory addresses and to handle transfers for I/O channels in systems. They are frequently implemented with standard VLSI controllers, but they may not be adapted easily to some systems. Discrete logic offers a custom solution, but it is expensive in terms of the number of devices required.

Actel offers a field programmable gate array (FPGA) solution with a DMA controller supermacro. The supermacro can be customized to accommodate any system, and can be combined with other logic in the FPGA to integrate other system functions.

This note explains the DMA controller design and its use to control the flow of data between an I/O channel and memory in a processor-based system.

Architecture

The architecture of the controller is shown in Figure 1. It consists of an address counter, a transfer counter, and a state machine controller.

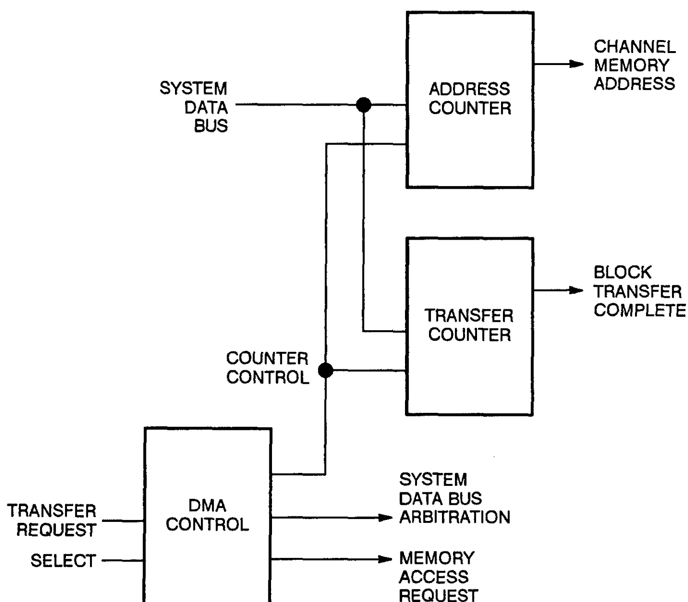


Figure 1. DMA Controller Supermacro Architecture

The address counter provides addresses for channel memory accesses. The transfer counter counts the number of transfers in a block and interrupts the processor when transfers are complete. The state machine controls access to the processor data and address bus. It also serves as the interface between the system memory controller and the I/O channel. Figure 2 shows the state machine state graph.

© 1990 Actel Corporation

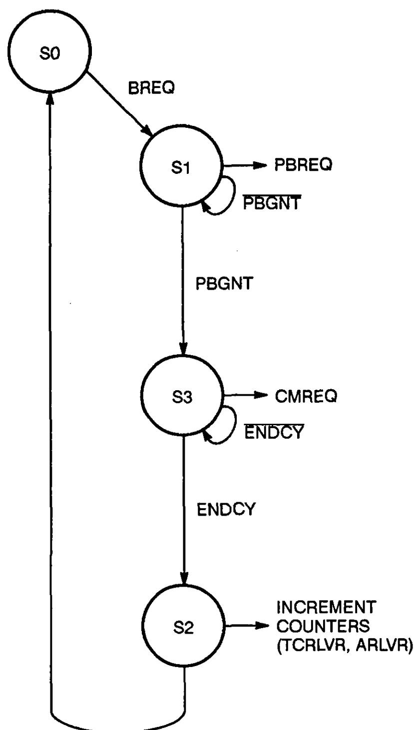


Figure 2. DMA Bus Interface State Graph

Operation

The processor writes the beginning address of the block transfer to the address counter and memory index register. The counter has 20 bits to access 1 M bit memories. The register has two bits to allow for accesses within one of four 1 M words. The counter and register may be used with a 4 X 1 M word memory configuration, or may be modified for any other configuration.

Loading the address counter/register is a two-step process, since the data port on the DMA macro is 16 bits and the counter/register is 22 bits. Upper and lower load controls are provided to load each part of the address. A single control is used to load the twos-complement of the block size to the transfer counter.

When the channel is ready for a transfer, it asserts a memory transfer request (BREQ) to the DMA controller. The state machine on the DMA controller performs the bus arbitration cycle with the processor by asserting a processor bus request (PBREQ) until it receives a processor bus grant (PBGNT). The processor bus grant also goes to the I/O channel so it may deassert the original request and drive data on the processor bus.

Next, the state machine requests a memory cycle (CMREQ) from the memory controller. When the memory cycle is complete, the DMA controller relinquishes the bus and increments the counters for the next cycle. The cycle is repeated until the transfer counter rolls over (TCRLVR), indicating the end of the block. The rollover can be used to interrupt the processor to reload the counters for the next transfer. There is also a rollover on the address counter (ARLVR), which can be used to interrupt the processor.

Actel

Using the SCSI Interface Controller Supermacro

Applications Note

Introduction

The rapid success of the SCSI interface as a standard has caused a corresponding increase in the use of SCSI controller chips. NCR manufactures several popular SCSI controllers. The NCR family of controllers features internal FIFOs, parity pass, and a variety of system bus interfaces.

Some of the devices can do multibyte system data bus transfers; some are optimized for a standard bus. However, investments in software and hardware development, as well as differences in price, may preclude the use of these devices in some systems. In such cases, a data transfer interface is needed between an eight-bit SCSI interface controller and the system bus.

This note explains how the Actel SCSI Interface Controller supermacro controls the flow of data between the system bus and an NCR 53C90B. The design can be adapted easily to a specific system. The macro uses about 60% of an ACT™ 1010 and less than a third of an ACT 1020.

Architecture

The architecture of the supermacro is shown in Figure 1. It consists primarily of a double-buffered bidirectional transceiver and a state machine. There is also a parity checker, which can be used to interrupt a processor when a single-bit parity error is detected.

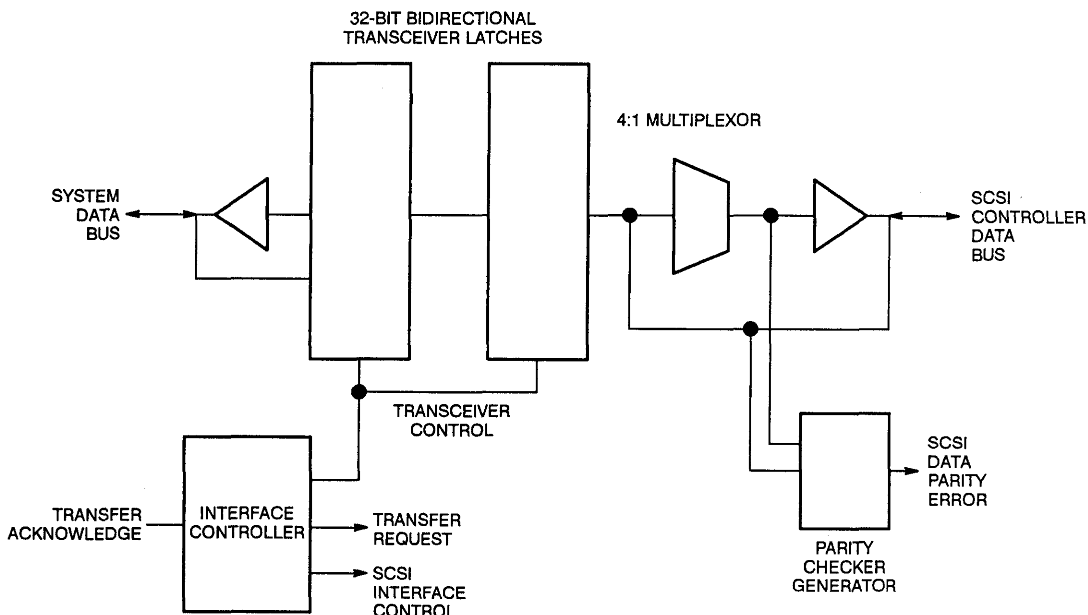
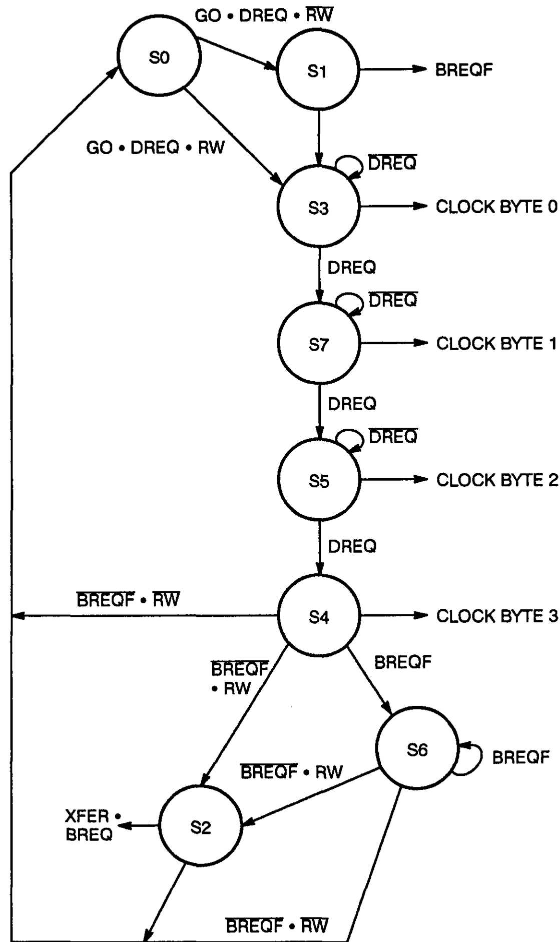


Figure 1. SCSI Interface Controller Macro

The controller cycles data through, requests the processor for data transfers, and passes control signals to the 53C90B to transfer data and access internal registers.

The heart of the design is the state machine, which controls the timing of all internal and external control lines. A state graph for the state machine appears in Figure 2.

© 1990 Actel Corporation



DREQ = SCSI CONTROLLER DATA TRANSFER REQUEST

BREQ = REQUEST FOR DATA TRANSFER OVER THE SYSTEM BUS

XFER = INTERNAL SIGNAL CONTROLLING DATA TRANSFERS BETWEEN THE TRANSCEIVERS

RW = READ/WRITE, WRITTEN INTO AN INTERNAL CONTROL REGISTER BY THE PROCESSOR

GO = READY TO TRANSFER, WRITTEN

CLOCK BYTE X = INTERNAL SIGNAL CONTROLLING BYTE TRANSFERS BETWEEN THE TRANSCEIVER AND THE SCSI CONTROLLER

Figure 2. SCSI Interface Controller State Graph

Operation

Preparing for Data Transfer

The processor accesses the 53C90B internal registers by first writing the appropriate data to the controller, then reading or writing the 53C90B.

The processor prepares the controller to transfer data by writing data to it that tells it the transfer direction and tells it to begin when the 53C90B controller asserts DREQ.

Writing

The processor writes a 32-bit word with four parity bits to the buffer and prepares the system as detailed above. When the controller gets a DREQ signal from the 53C90B, it transfers the data in the processor-side buffer to the SCSI-side buffer, then requests the processor for more data. It then multiplexes each byte with its parity bit out to the controller data bus and issues an acknowledge to the 53C90B.

If the request for data from the processor has been granted by the time the last byte is sent, the process will repeat, until the 53C90B stops requesting data.

If the 53C90B stops asserting DREQ, the state machine will wait until the signal returns before sending the next byte. If the transfer is of an odd number of bytes, or if the transfer stops on an odd-byte boundary due to a failure, the processor can initialize the state machine by activating the controller chip-select line.

Reading

The flow of data from the controller to the system bus begins when the 53C90B asserts DREQ. The controller latches each byte of a word until the buffer on the 53C90B side is full. Then the controller transfers the word to the other buffer, sets the data transfer request flag, and resumes latching bytes.

If the processor has responded to the data transfer request by the time the next word is latched, then the process continues. If not, the state machine will stop until the buffer is read.

Parity Error Detection

The parity error detection logic checks each byte that passes through when the DACK signal is asserted. If an error is detected, it sets a flag that could be used to interrupt the processor. The flag can only be cleared by the processor accessing the controller.

Actel

UART Design

Applications Note

Introduction

This applications note demonstrates the design of a Universal Asynchronous Receiver and Transmitter (UART) circuit on an Actel field programmable gate array (FPGA). The architecture of ACT™ 1010/1020 arrays provides the flexibility needed for the circuit design. Actel macros are used to speed and simplify the design process.

UART Description

The UART is a serial communication device that handles transmission and reception of information between computers and peripherals (modems, terminals, printers, etc). This note describes a general-purpose UART that utilizes 189 logic modules (approximately 700 gates) in the ACT 1010.

Figure 1 shows the UART configured as a controller between a computer and a peripheral device. Communications protocol is a standard RS232 serial interface.

The UART transmission format is shown in Figure 2. A start bit begins the transmission, followed by the data bits, and then an optional even or odd parity bit. A programmed number of stop bits concludes the transmission.

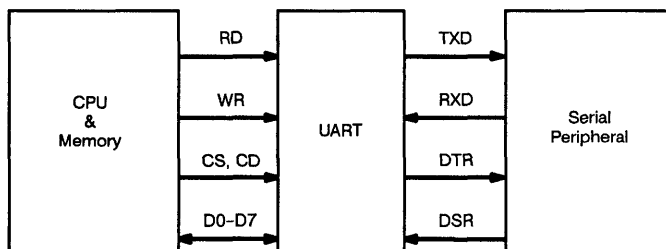


Figure 1. UART System Block Diagram

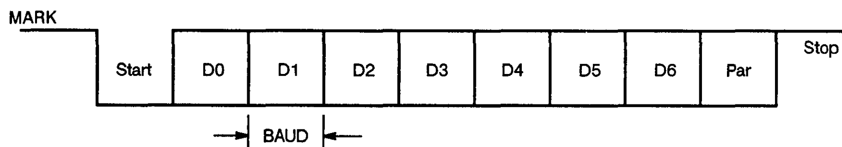


Figure 2. UART Transmission Format

© 1990 Actel Corporation

UART Design

The UART application described in this note operates similar to an INTEL 8251 operating asynchronously. See Figure 3 below.

When the CPU sends a data character, the UART automatically adds a start bit, followed by 7 (or 8) data bits, an even or odd parity bit, and a stop bit. The UART then transmits this "packet" as a serial data stream on the TXD output. It shifts out serial data at a rate equal to 1/16 the clock frequency (pin CLK16X).

When the CPU receives a data character, a falling edge on the UART's RXD input triggers the beginning of a start bit. The start bit is again checked to prevent a false start. If the start bit is valid, the data character is shifted in, followed by one stop bit. A serial stream of data is then converted to parallel format, ready for CPU retrieval.

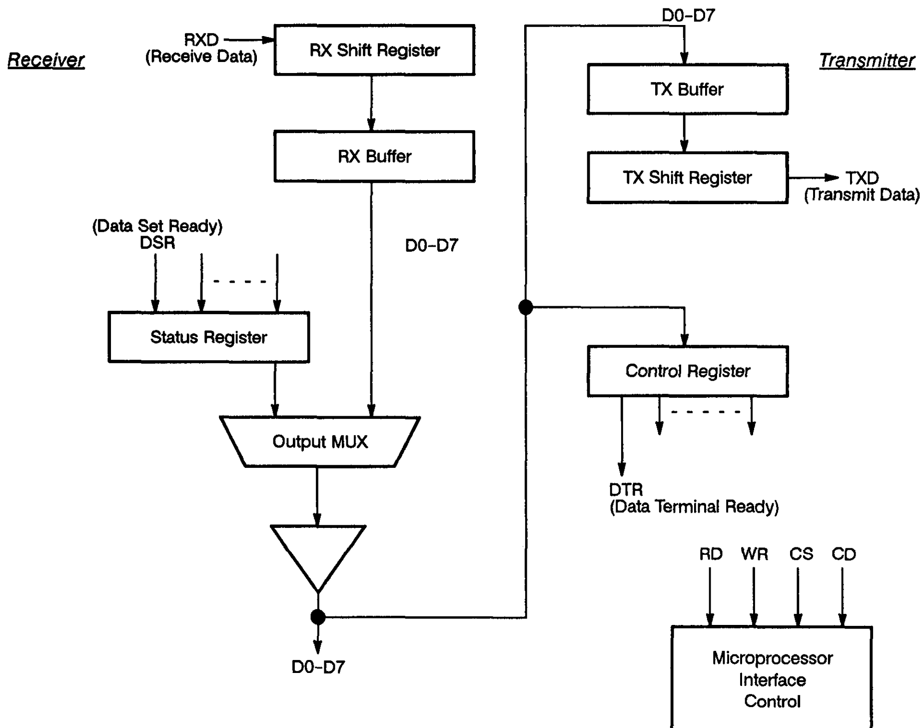


Figure 3. UART Function

Transmitter Design

Figure 4 shows the transmitter's buffer and shift register.

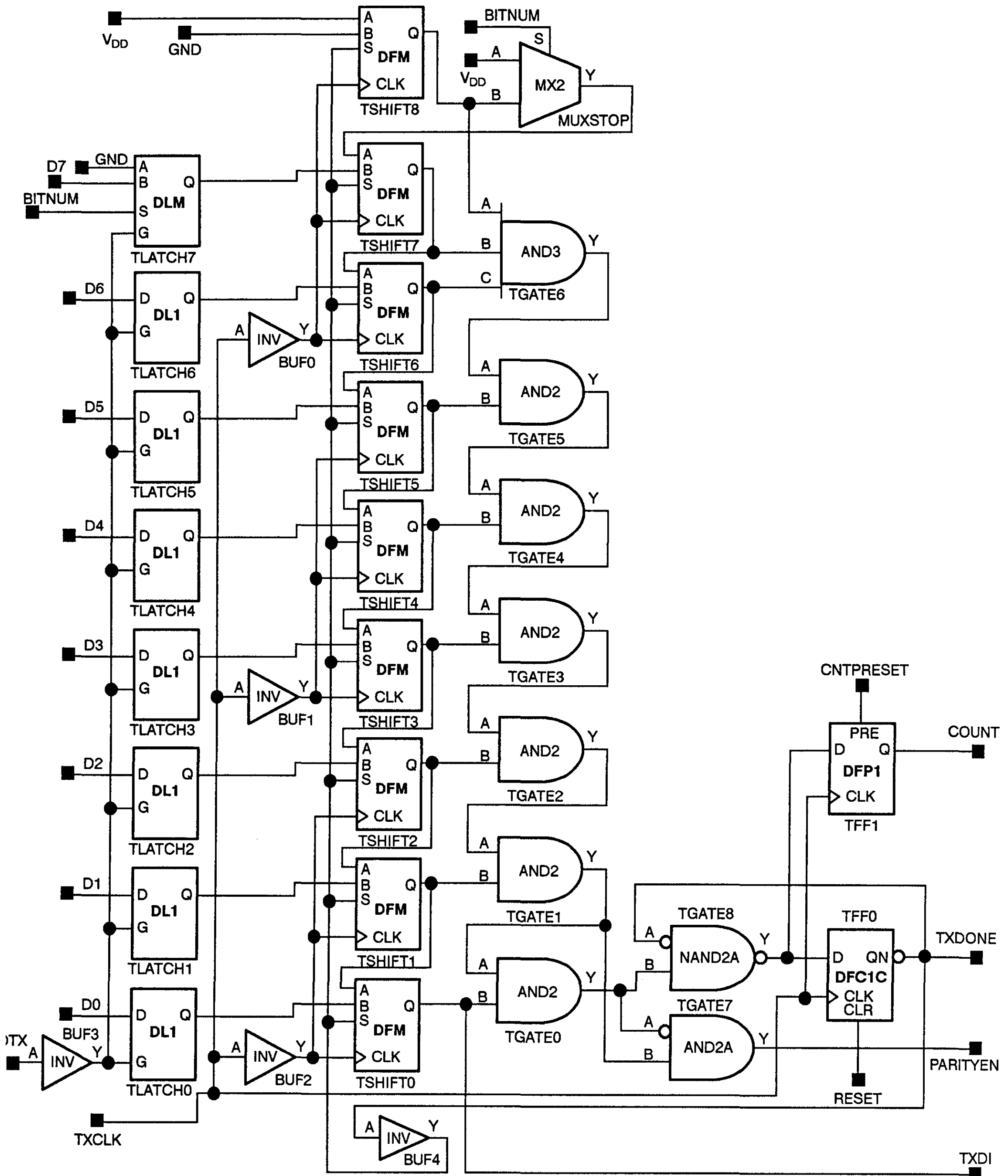


Figure 4. Transmitter Buffer and Shift Register

The transmitter first latches the parallel data word D0–D7 from the CPU in the buffer TLATCH0–TLATCH7. It then transfers the data character to the shift register TSHIFT0–TSHIFT7 through the B inputs. After the first clock (TXCLK), TXDONE goes low selecting the A inputs of TSHIFT0–TSHIFT7. TXCLK then serially shifts out the data character. A series of AND gates (TGATE0–TGATE6) detect a logic high, indicating the stop bit, propagating through the shift register.

Transmitter Buffer

The transmitter buffer TLATCH0–TLATCH7 uses transparent latches to store a data character. The level-sensitive transparent latch consumes only one ACT 1 logic module.

Because of the wide variety of latches available in the ACT 1 library, designers may build efficient registers for any application: synchronous clear, gate-enable, active high or low outputs, and active high or low gates.

Clocks

It is also possible to construct a two-phase clock system by combining active high gate latches and active low gate latches. See Figure 5 below.

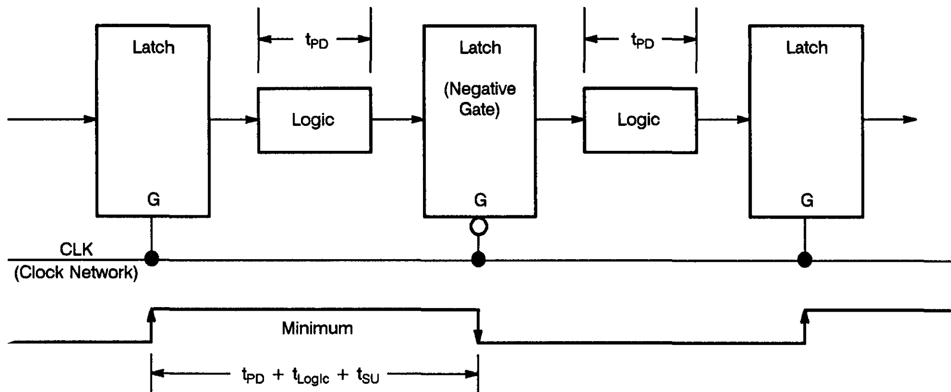


Figure 5. Two-Phase Clock Using Transparent Latches

The ACT 1 macro library includes a large selection of flip-flops, which are edge-sensitive storage elements. These flip-flops are available with asynchronous clear, asynchronous preset, and clock enable inputs. Negation bubbles are available on all inputs and outputs.

The clock enable feature allows separation of data storage and transfer while maintaining a common clock system. See Figure 6 below.

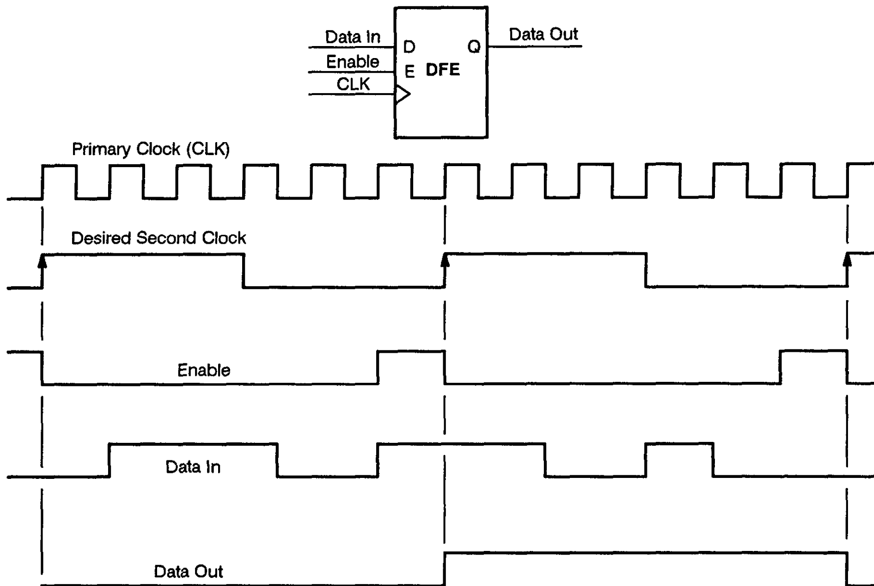


Figure 6. Clock Enable Function

Transmitter Shift Register

The transmitter's shift register TSHIFT0–TSHIFT7 uses flip-flops with a built-in 2:1 multiplexor (MUX) on the D input. The built-in MUX can emulate two-input logic functions, as shown in Figure 7.

The MUX flip-flop uses two logic modules (as does a standard flip-flop). It is thus possible to construct a parallel load/serial shift register without using additional logic.

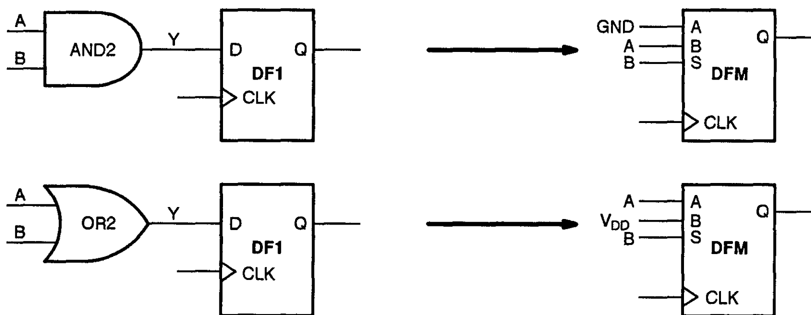


Figure 7. Reducing Logic Levels with MUX Flip-Flops

Transmitter Clock Circuit

The clock circuit is shown in Figure 8. It is implemented as a 4-bit Gray code counter with both synchronous and asynchronous reset (COUNT and RESET, respectively) and count enable (GOTX).

An external clock signal (CLK16X) drives the state machine. Gates G13 and G14 decode the outputs Q3–Q0 of the state machine and generate an internal clock (TXCLK) for the transmitter shift register. Only one bit changes at a time, eliminating glitches on the transmitter clock (TXCLK).

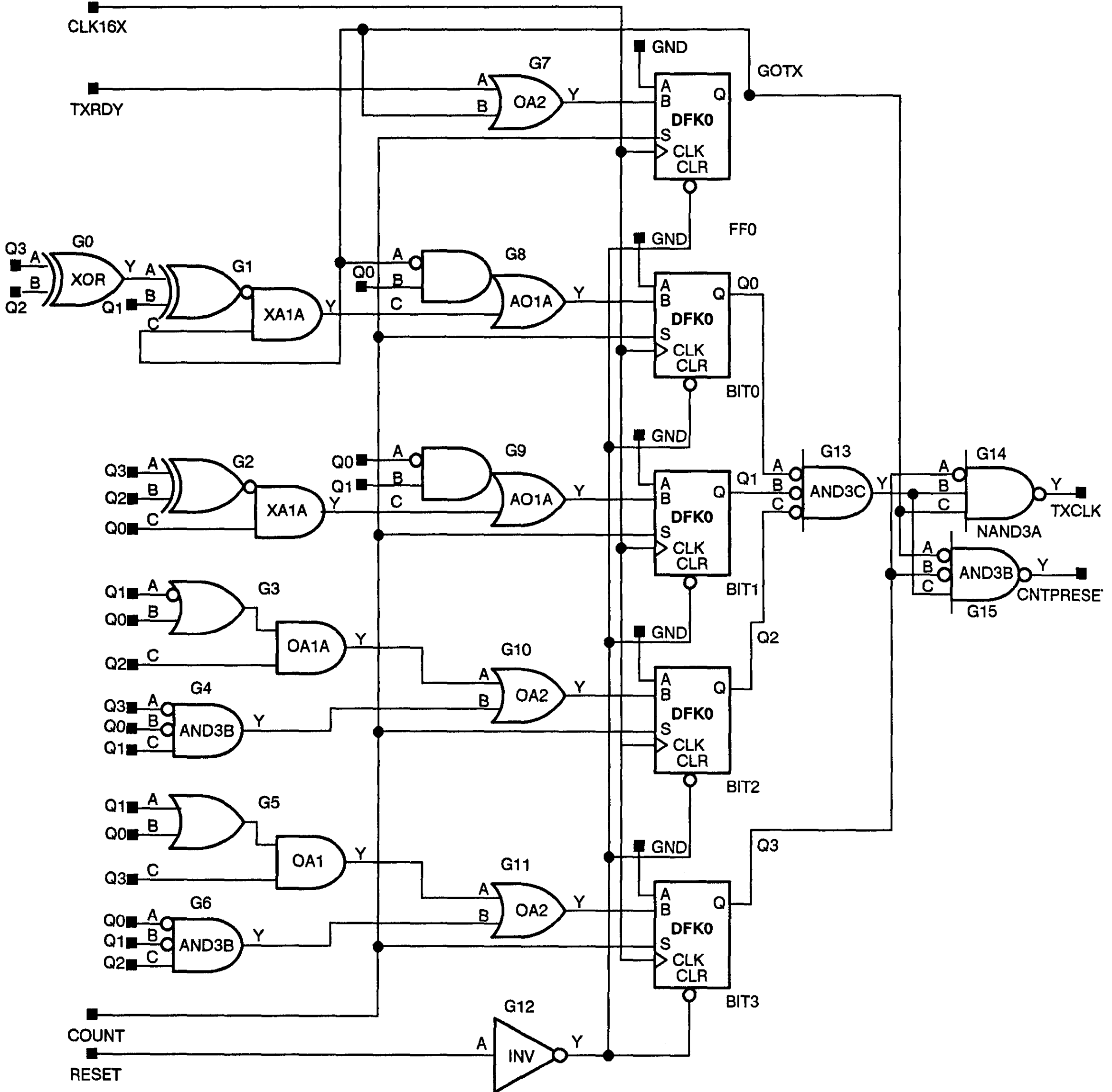


Figure 8. Transmitter Clock Circuit

Two-Level Logic

The transmitter state machine takes advantage of the OR/AND macro, a two-level logic function using only one logic module. In combinatorial logic, use of two-level logic functions can significantly reduce the module count. There are different types of OR/AND macros with a wide choice of input combinations. Refer to the ACT 1 family data sheet for a list of OR/AND and AND/OR macros.

Decoders

Decoder functions require signals and their complements. Using negation assertion levels available on Actel macros simplifies the logic path and conserves module usage. See the 2:4 decoder in Figure 9.

It is important to note that propagation delay through a logic module is independent of the implemented function. Thus the designer should use negation bubbles and two-level logic macros whenever possible.

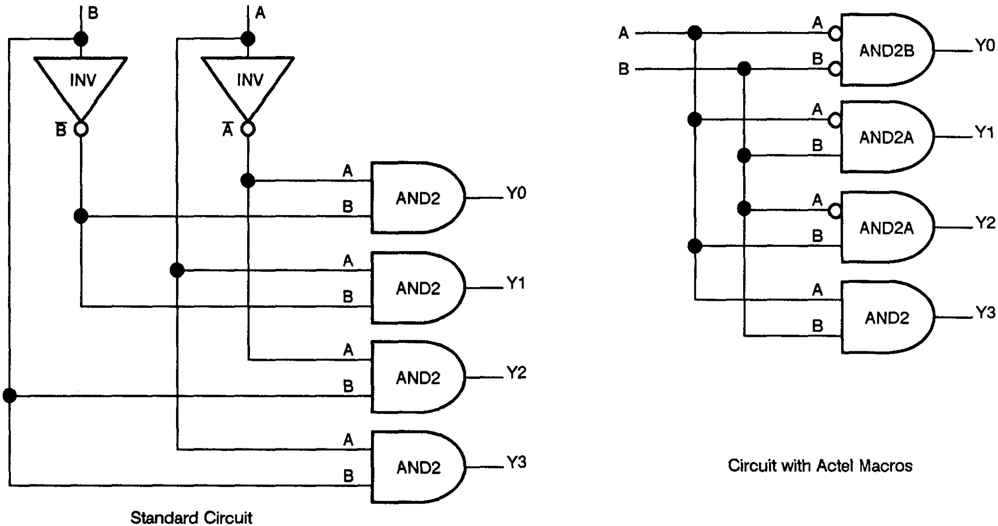


Figure 9. 2:4 Decoders

Parity Generation

The transmitter parity generator is programmable for odd or even parity. The parity generator circuit is shown in Figure 10.

From the shift register, data serially enters the parity generator on the transmit line (TXDI). The generator inserts the parity bit just prior to the stop bit. A 2:1 MUX (PMUX) selects either data or parity bit to the data transmit line TXD.

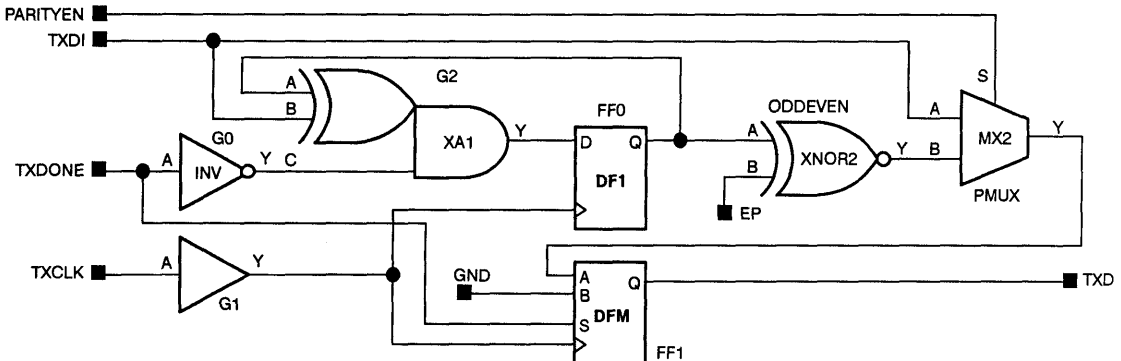


Figure 10. UART Parity Generator

Logic Module Conservation

Efficient utilization of Actel macros allows implementation of the UART transmitter section using only 80 of the 295 available logic modules on the ACT 1010 array. The parity generator uses the XOR/AND macro. The XOR/AND, XOR/OR, and AND/XOR functions each utilize only one logic module.

ACT 1 devices are particularly efficient at implementing multiplexor functions. Both 2:1 and 4:1 MUXs can be implemented in only one logic module each. MUXs implement internal bus designs in place of internal three-state drivers on ACT 1 devices, as shown in Figure 11.

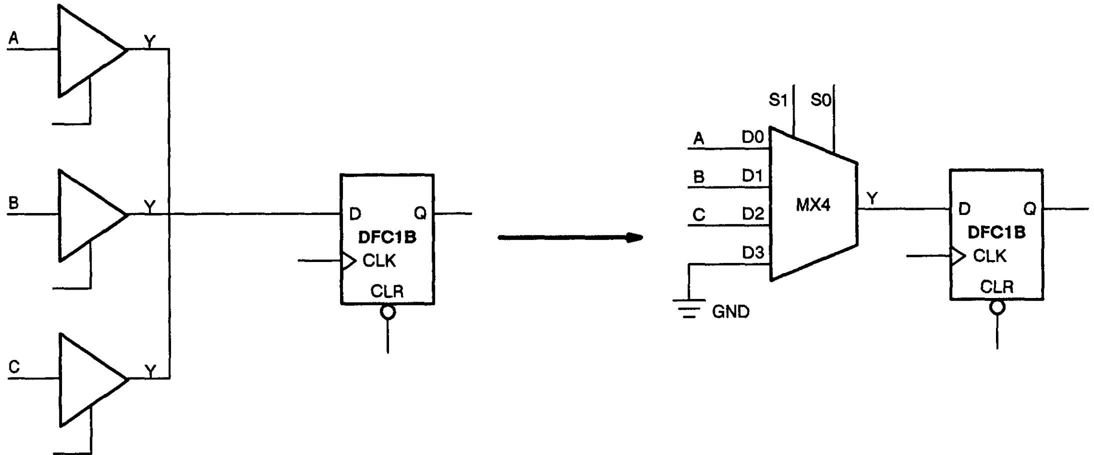


Figure 11. Internal Three-State Function Implemented with ACT 1 Multiplexor

Receiver Design

Shift Register and Buffer

Figure 12 on the next page shows the schematic of the receiver's shift register and buffer. A logical "1" is loaded into the receiver shift register RSHIFT0-RSHIFT9. After the first clock (SHFTRX), XFERRX goes low, selecting the A inputs of the MUX flip-flops. The low start bit precedes the serial data shifts into the register. When a complete serial data character shifts into the register, the parallel data character transfers to the buffer (RLATCH0-RLATCH7) for CPU access.

The receiver design takes advantage of the single-module transparent latch with MUX, minimizing required circuitry. Similarly, the built-in MUX can emulate other two-variable functions.

The MUX also can emulate an inverter, effecting a negation at the data input of the transparent latch.

Figure 12. Receiver Shift Register and Buffer

Receiver Clock Circuit

The receiver clock circuit is a 4-bit Gray code counter (similar to the transmitter clock circuit) and is driven by the same external clock signal (CLK16X). See Figure 13.

The state machine has both synchronous and asynchronous resets (RXCOUNT, RESET). A filter circuit (RXDFILTER) prevents false starts due to errors on the data line (RXDI). The receiver clock (SHFTRX) is decoded from the outputs of the state machine (Q0–Q3).

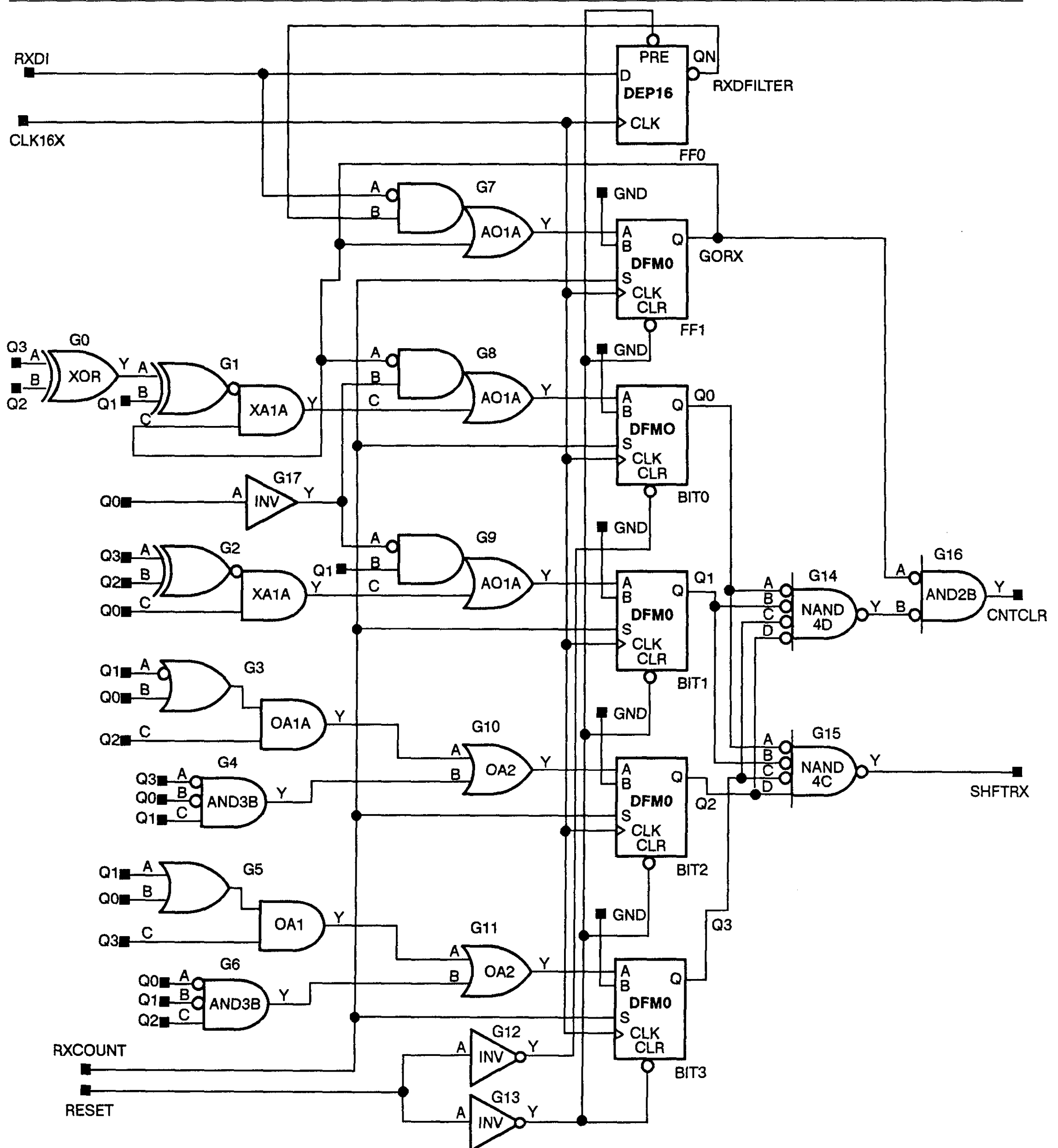


Figure 13. Receiver Clock Circuit

Parity Checker

The receiver parity checker is programmable to either odd or even parity. The parity checker generates a parity bit from the incoming serial data stream and compares it to the incoming parity bit. The parity error flag (PARITYERROR) is set accordingly. See Figure 14.

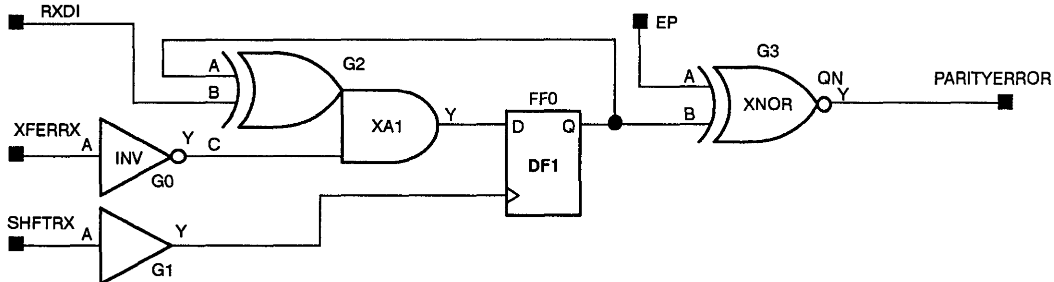


Figure 14. Parity Error Generator

Design Complexity

The UART transmitter section can be implemented using only 80 of the 295 available logic modules on the ACT 1010 device. The receiver section requires only 78 logic modules.

These figures represent 27.1% utilization of the array for the transmitter section and 26.4% utilization of the ACT 1010 for the receiver section.

Table 1 provides a breakdown of logic utilization for the UART design on Actel's FPGA.

Table 1. Logic Utilization

Transmitter	Logic Modules
Buffer and Shift Register	45
Clock Circuit	26
Parity Generator	9
Total	80
Receiver	**Logic Modules**
Receiver Shift Register and Buffer	42
Receiver Clock	30
Receiver Parity Checker	6
Total	78

Actel TA181 ALU

Applications Note

Introduction

The venerable TTL 74181 is a well-known and useful arithmetic logic unit (ALU). Its four select lines, mode, and carry-in bits combine to offer over 40 unique arithmetical and logical functions. Many designers dislike losing such functionality as they move from discrete to ASIC technology.

Designers may retain the function of the 74181 using Actel field programmable gate arrays. Actel's macro library includes "TA181," a soft macro functionally identical to the TTL component.

Implementation

The 74181 depicted in popular TTL data manuals has six logic levels (seven for A = B output). While this representation may not be accurate at the transistor level, it typifies an ASIC design implementation.

The Actel TA181 design requires only four logic levels (five for A = B output). It uses components available from Actel's library. Actel logic module flexibility enables such "logic compression" by implementing a wide variety of logic functions.

For example, each bit pair in the ALU passes through a section of logic, where select lines qualify the pair before outputting them to the remaining logic. The qualification logic, as it appears in TTL manuals, is shown in Figure 1. It uses eight gates and three logic levels.

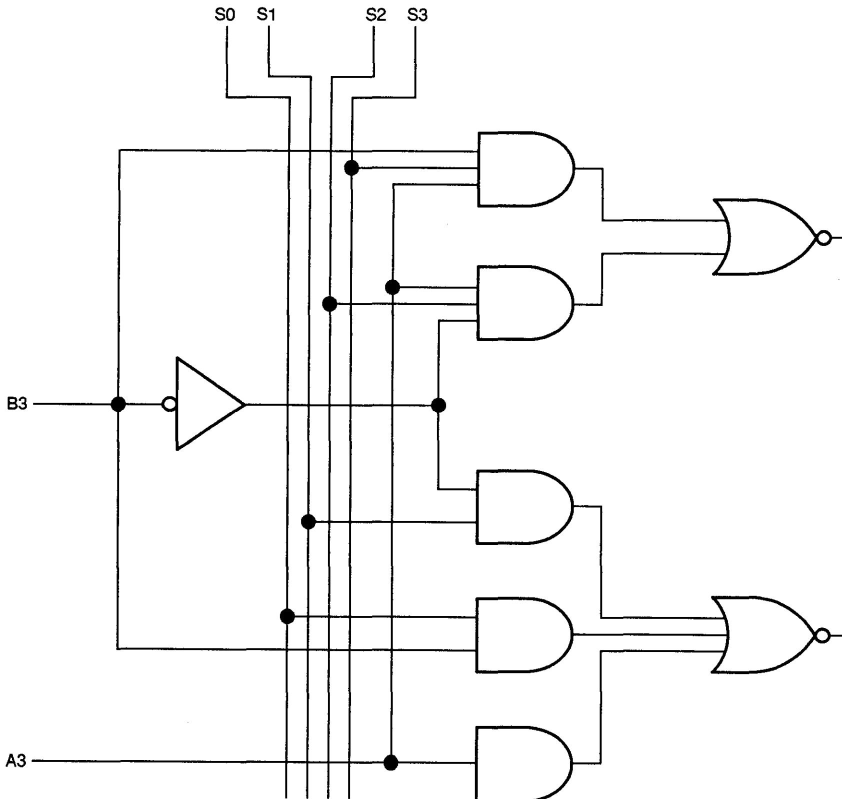


Figure 1. TTL 74181 Qualification Logic

3

© 1990 Actel Corporation

Figure 2 shows Actel's version of qualification logic. By contrast, it requires only two modules and one level of delay. The driving gate is an OR function, as opposed to NOR in the TTL diagram. Bubbling (inverting) inputs on each gate driven by the section adjusts for this difference. All gates in the Actel library have versions with bubbled inputs.

One module implements two three-input ANDs to two-input OR (AO4A), and the other module implements two two-input ANDs to three-input OR (AO5A). Combined, these modules create the qualification logic.

The remaining logic in the TA181 macro is similar to that found in the TTL 74181, with some improvements. M inputs require no inverter, because M-driven gates have bubbled inputs. Single logic module macros implement the AND-OR functions.

Another example of logic compression is in macro OA5, used in the logic generating the F3 output. The macro implements the expression:

$$Y = /((/A*/B*/C) + (/A*/D))$$

in a single module. OA5 is a clear example of the savings in delay and modules achievable with the versatile Actel logic module.

Conclusion

Actel's ALU 181 design uses only 32 logic modules, only about 10% of an ACT™ 1010 gate array. It is an excellent example of the Actel logic module's adaptability, both in implementing logic functions and in compressing logic, for efficient, high-performance designs.

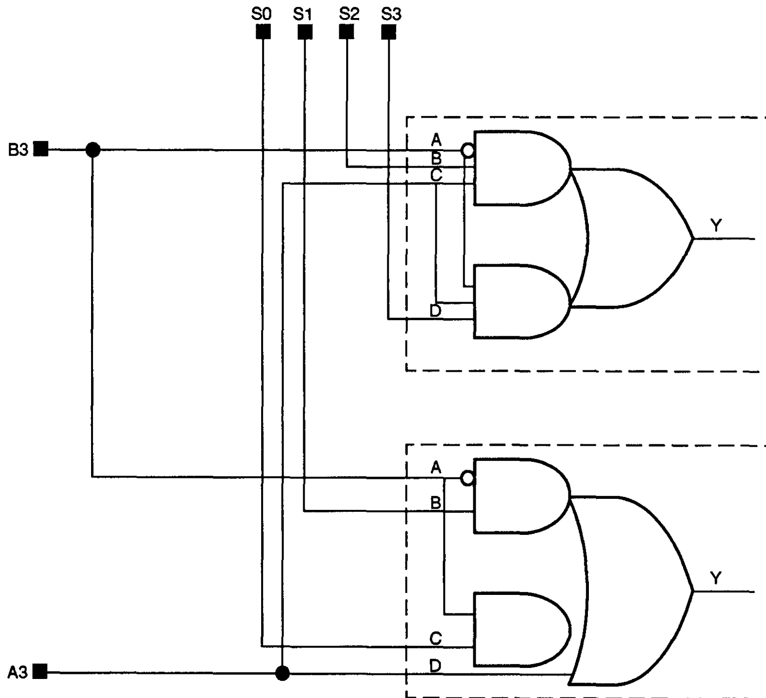


Figure 2. Actel's TA181 Qualification Logic

Actel

Actel Fast Adders

Applications Note

Introduction

Adders, in a variety of sizes, perform a very common function in digital designs. They are frequently implemented using carry-look-ahead (CLA) logic to speed carry propagation. However, as adder size increases, CLA logic requires increasingly wider gates and more logic levels; such requirements degrade adder performance.

Implementation

Actel solved the CLA logic problem by designing its family of fast adders using the Carry Select (CS) method. CS adders partition the addition into groups of bits. Two additions are performed simultaneously for all the groups. One addition is done with a carry-in of one and the other with a carry-in of zero. The two group sums and their carry-outs are connected to two-input multiplexors (MUXs).

When determined, the carry from the lower groups selects the group that supplies the sum to the outputs. The carry-out for the group selects the sum and carry for the next higher group, and so on.

The speed of group additions and carry propagation determine the speed of the CS adder.

Group Size

Since carries ripple within groups, small group size is an advantage. Small groups, however, require more groups for a given number of bits, lengthening the carry propagation path. Thus, the design trade-off is the number of delays to a group sum versus the number of delays required to propagate a carry.

By optimizing group sizes based on adder macro delay parameters, Actel adders are very fast.

Carry Propagation

As previously mentioned, carry propagation speed is a limiting factor in carry select adder performance. Group carries must be selected, just as the sums are. Unlike sums, carries select higher carries and so forth. This creates the problem of cascading carries through multiple logic levels.

The flexibility of Actel's logic module alleviates cascading carry problems by implementing a cascaded two-input MUX in a single module. The logic is shown in Figure 1. A two-input MUX drives the select for another two-input MUX. The macro allows two-level carry selection using only one module and in only one delay. All Actel adder macros use the cascade MUX macro for rapid carry propagation.

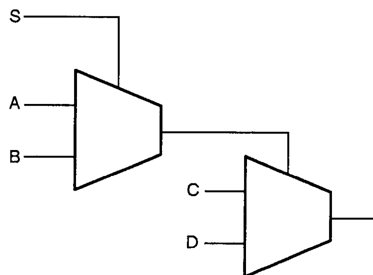


Figure 1. Cascade Multiplexor Macro

© 1990 Actel Corporation

The Actel Fast Adder Family

The following table lists Actel adders, their speeds (driving one load), and the number of logic modules used:

Macro Name	Number of Bits	Worst-Case Delay (ns)	Levels of Logic	Number of Modules
FADD8	8	36	4	37
FADD12	12	45	5	63
FADD16	16	47	5	78
FADD24	24	56	6	120
FADD32	32	65	7	160

Eight-Bit Twos Complement Multiplier

Applications Note

Introduction

Multipliers are vital to systems design. In the past, designers integrated VLSI multiplier chips into their designs. As ASICs become more popular, ASIC-based systems increasingly utilize multiplier macros. Previously, field programmable devices did not have sufficient architectural flexibility and connectivity to efficiently implement multipliers. Actel has designed an eight-bit twos complement multiplier that easily fits on an ACT™ 1010 or ACT 1020 field programmable gate array (FPGA).

Algorithm

High-performance multipliers form the product of two numbers using combinatorial logic and an array of adders. Actel's multiplier is based on the Baugh-Wooley[1] algorithm (Equation 1), and uses conventional full adders for twos complement multiplication. When the size of the multiplier and multiplicand are known, the Baugh-Wooley formula can generate a series of product terms. In this example, when both multiplier and multiplicand are eight bits, M and N equal 8 in the formula.

Each product term contains "2" raised to a power. The product terms are arranged into an array according to their binary power. Adding the columns of the array results in the product of the two original operands.

In the generic architecture, AND gates generate the product terms, and the adders sum them together. The adder array has a parallelogram shape. Dividing the parallelogram into four sections eases the design process. Each section generates a set of partial products, which are then added together.

$$P = a_{m-1} b_{n-1} 2^{m+n-2} + \sum_{i=0}^{m-2} \sum_{j=0}^{n-2} a_i b_j 2^{i+j}$$
$$+2^{m-1}(-2^n + 2^{n-1} + \bar{a}_{m-1} 2^{n-1} + a_{m-1} + \sum_{i=0}^{n-2} a_{m-1} b_j 2_i)$$
$$+2^{n-1}(-2^m + 2^{m-1} + \bar{b}_{m-1} 2^{m-1} + b_{n-1} + \sum_{i=0}^{m-2} \bar{a}_i b_{n-1} 2^i)$$

Equation 1. Baugh-Wooley Algorithm

Actel Multiplier Macro Architecture

The Actel multiplier macro architecture appears in Figure 1. It consists of three major sections: the partial product generators, Wallace tree adder, and final product adder.

The partial product generators implement the adder array sections using Actel library components. Since the partial product generators have similar structure, the design was copied and modified to create the others.

The Wallace tree adder sums the partial products and passes two binary numbers on to the final product adder. The final product adder performs arithmetic necessary to produce the product. The following is a discussion of each element in detail.

© 1990 Actel Corporation

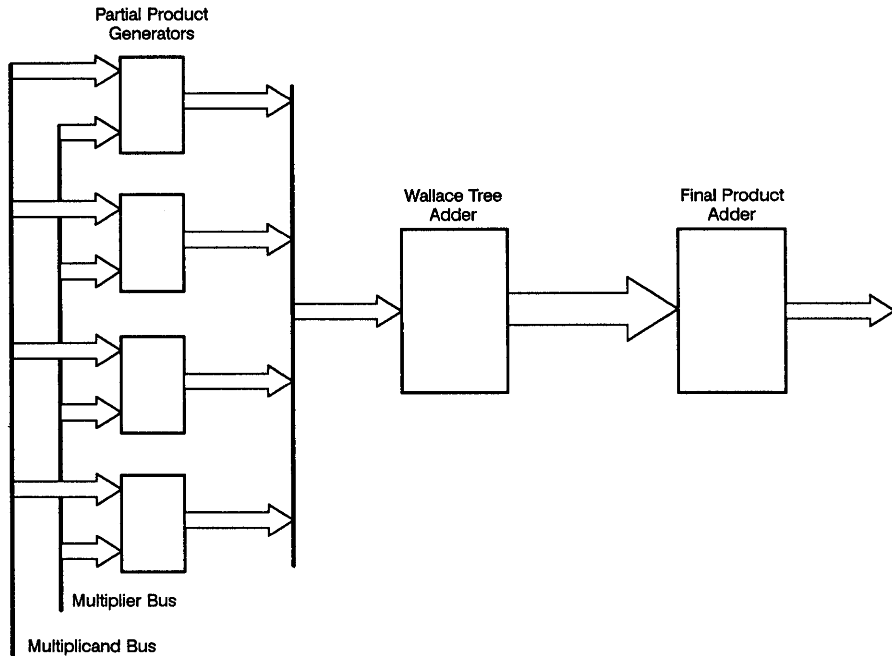


Figure 1. Actel Multiplier Macro Architecture

Partial Product Generator

Each partial product generator requires as input four bits from each operand. The bits pass through AND functions, which output through an array of adders to generate eight partial products. Because the partial product generators operate in parallel, all partial products are produced simultaneously.

The schematic has an array of adders with inputs either from an AND gate or the sum from another adder. The Actel library adder FA2A adds a two-input AND term with inverted inputs to another input and a carry-in. Most of the adders in the generator use FA2A, requiring only two logic modules to implement both the AND and full adder functions.

Wallace Tree Adder

The multiplier's Wallace tree section sums columns of terms from partial product generators. Summing a column in binary arithmetic requires tracking carries generated to the higher power-of-two positions in the number. The Wallace tree uses three-bit adders to input up to three bits of the column. It feeds the adder sums into adders taking in other bits of the column, and feeds carry-outs to the inputs of adders summing the column in the next higher bit position, and so on. The tree output is a series of sum and carry-in bits which may be added in a conventional adder as two binary numbers, producing the final product.

Because both carry-outs and carry-ins are active-low on the Actel adder macros, the adders can be used in a Wallace tree structure as a three-bit adder which outputs a sum and a carry. The carry-out from an adder in one column inputs directly to the carry-in of the next higher column. Other than the adders, no logic is required to design the Wallace tree.

Final Product Adder

The multiplier's last section is a binary adder. The Wallace tree completely sums some lower columns of bits, which pass from the tree to the product bus. The tree outputs higher bits as two binary numbers to be added. The bits comprise the sum bit for each bit position and its respective carry-in. The final adder adds them together for the remaining bits of the product.

The final adder uses the same design technique as the Actel family of fast adder macros. The architecture of the final product adder is shown in Figure 2.

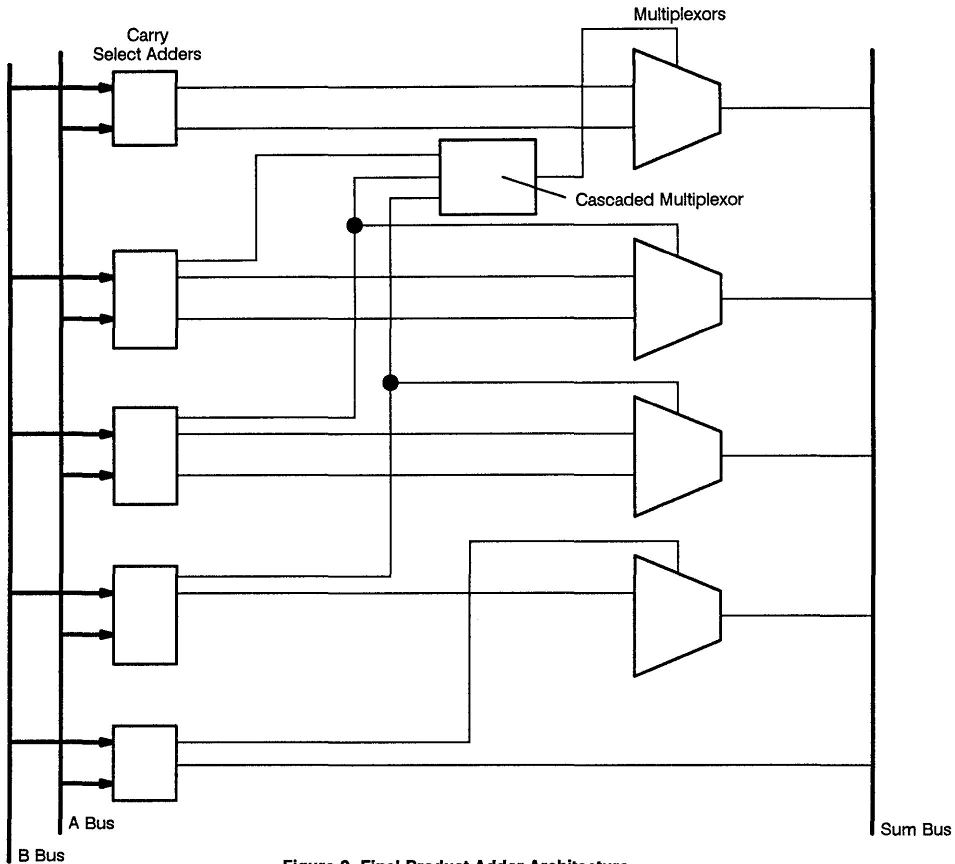


Figure 2. Final Product Adder Architecture

The Carry-Look-Ahead Problem

Adders are usually implemented using carry-look-ahead (CLA) logic to speed carry propagation. However, as the size of the adder increases, the CLA logic requires wider gates and more logic levels; such requirements degrade adder performance.

Carry Select Addition

To solve the CLA problem, Actel designs its fast adders using the Carry Select (CS) method. CS adders partition the addition into groups of two or three bits. Two additions occur simultaneously for all groups. One addition has a carry-in of one and the other a carry-in of zero. The two group sums and their carry-outs connect to two-input multiplexors.

When determined, the carry from the lower groups selects the supply that sums the outputs. Next, the carry-out for the group selects the sum and carry for the next higher group, and so on.

The CS adder speed is determined by group additions speed and carry propagation speed.

Group Size

Since carries ripple within groups, small group size is an advantage. Small groups, however, require more groups for a given number of bits, lengthening the carry propagation path. Thus, the design trade-off is the number of delays to a group sum versus the number of delays required to propagate a carry.

By optimizing group sizes based on adder macro delay parameters, Actel adders are very fast.

Carry Propagation

As previously mentioned, carry propagation speed is a limiting factor in carry select adder performance. Group carries must be selected, just as the sums are. Unlike sums, carries select higher carries and so forth. This creates the problem of cascading carries through multiple logic levels.

The flexibility of Actel's logic module alleviates cascading carry problems by implementing a cascaded two-input MUX in a single module. The logic is shown in Figure 3. A two-input MUX drives the select for another two-input MUX. The macro allows two-level carry selection using only one module and in only one delay. All Actel adder macros use the cascade MUX macro for rapid carry propagation.

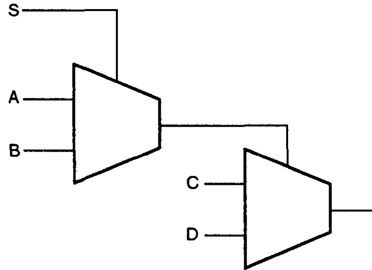


Figure 3. Cascade Multiplexor Macro

The Actel Advantage

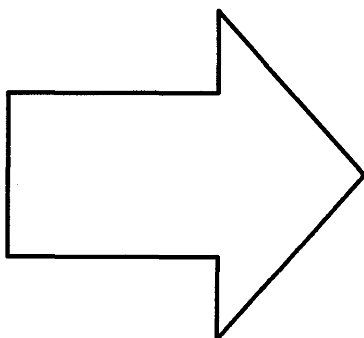
The eight-bit twos complement multiplier illustrates the type of macros that can be implemented on Actel field programmable gate arrays. The design method detailed here suits multipliers of many different configurations. This technique also applies to 3X7, 9X4, or 6X5 multipliers, depending on system requirements.

Other multiplier algorithms also apply. A magnitude array multiplier design using the Cavanagh method[1] requires less logic than the eight-bit multiplier above. Other alternatives include the Booth and Bit-Pair Recording methods[1].

The multiplier macro may be designed as part of the logic in a field programmable gate array. The remaining gate array modules are available for other design purposes, either other arithmetic functions or glue logic. For example, Actel's 8X8 multiplier macro uses 241 of 546 modules on an ACT 1020. Nearly 60% of the chip is still available for other purposes.

In any design, Actel provides the technology to complete the entire gate array development process including schematic capture, simulation, place and route, timing analysis, and device programming.

1. Cavanagh, Digital Computer Arithmetic Design and Implementation, McGraw-Hill, New York. 1984.



Product Data 1

Reliability Report 2

Applications 3

Article Reprints 4

Article Reprint 1:

Analyze FPLD Architectures, Performance, and Development Tools to Optimize Design-Dependent Selection 4-1

Article Reprint 2:

A CMOS Electrically-Configurable Gate Array 4-11

Article Reprint 3:

An Architecture for Electrically-Configurable Gate Arrays 4-23

Article Reprint 4:

Dielectric-Based Antifuse for Logic and Memory ICs 4-29

Article Reprint 5:

Field Programmable Gate Array with Non-Volatile Configuration 4-35

Article Reprint 6:

Testing Antifuse-Based FPGAs 4-45

Article Reprint 7:

Compare ASIC Capacities with Gate Array Benchmarks 4-49

Article Reprint 1

ANALYZE FPLD ARCHITECTURES, PERFORMANCE AND DEVELOPMENT TOOLS TO OPTIMIZE DESIGN-DEPENDENT SELECTION

by Dennis McCarty
Sr. Applications Engineer, Actel Corporation
Sunnyvale, California

4

© 1990 Actel Corporation

Analyze FPLD Architectures, Performance and Development Tools to Optimize Design-Dependent Selection

by Dennis McCarty, Sr. Applications Engineer
Actel Corporation, Sunnyvale, California

Over the past decade, field-programmable logic devices (FPLDs) have undergone dramatic changes. The devices of the late 1970s and early 1980s primarily consisted of fuse-programmable PALs, pioneered by Monolithic Memories Inc. (now part of Advanced Micro Devices), and a handful of alternative versions from competitive manufacturers. The devices were used by circuit designers to integrate discrete TTL logic into higher density packages. The designer's goal was to take four or five discrete logic chips and replace them with one. And while the existing device architectures were not that sophisticated, that modest goal was easily accomplished using fairly inexpensive devices and programming hardware.

Consider, for example, the device choices available to the designers of ten years ago. Clearly, bipolar was the most prevalent technology in use with devices being "hard" programmed by blowing some sort of a fuse — PALs, FPLAs and alternatives were good examples of that genre. The differences between competing devices related to the number of I/O, the number of product terms and packaging options.

Today, FPLD density levels have increased as have the number and types of devices that compete in the logic-integration arena. Indeed, it's common to find a dozen FPLDs on the market today that can easily accommodate the functionality of 20 random TTL logic devices. And CMOS has replaced bipolar as the dominant technology of choice for FPLDs.

Perhaps the biggest change has taken place in the area of device programming and the enabling technology. While conventional bipolar fuse technology continues in use today, EPROM, EEPROM, static RAM (SRAM) and antifuse programming technologies have quickly been adopted by the majority of new FPLD companies entering the industry. And device programmers have been replaced with sophisticated workstation-based development environments that combine powerful new hardware and software that speeds the design time for complex circuit designs.

But as the complexity of the new breed of FPLDs has increased for designers, so has the difficulty in determining the suitability of the various devices for particular applications. Moreover, with new manufacturers emerging each year with devices of different architectures and design methodologies, the 1990s system designer's selection task is likely to become more difficult as the decade progresses. Even device names have gotten more sophisticated and, as a result, more confusing. Today, the terms PLD, EPLD, FPGA, PGA, LCA, PMD, PAL, PLA, EEPLD as well as others are all generally being used to refer to devices that fall into the FPLD category.

Interestingly, once armed with the proper set of criteria, system designers can accurately and safely select preferential devices. The guidelines that follow take aim at achieving that goal for designers. Specifically, the criteria relates to comparing device technologies, understanding density issues and evaluating development tools.

Comparing Device Technologies is Key

Because memory-based technologies are now in use, the question of device volatility becomes an issue. Specifically, the volatility of the programming element used to interconnect the logic on device determines whether programming is permanent or must be performed each time power is applied. One-time programmable devices use fuses or antifuses. Reprogrammable devices are either electrically reconfigurable (EEPROM or SRAM technology), or optically reconfigurable (EPROM technology).

Only the SRAM technology is volatile in that, if power is interrupted, the device must be reloaded. SRAM or EEPROM devices must be used if the design requires that the device be dynamically reconfigured in operation.

If reconfiguration is not required, then the advantages each technology offers need to be considered. Some designers are attracted to reprogrammability during the development cycle because it allows them to try design iterations without wasting components. That is a consideration, but it must also be noted that there is a design time and production cost penalty for the software and hardware support required to boot the device. As always, when evaluating systems, all the relevant costs should be considered. For example, losing parts to design iterations is not nearly as costly as placement and routing (P&R) that require manual intervention.

In certain applications, such as military, nonvolatility is preferred. In other applications, such as aerospace/satellite environments, radiation-tolerant nonvolatile technology may be the only option due to the threat of a loss of program information from radiation. The vendors of devices being considered for use in a hostile environment should be asked what conditions could adversely affect their products.

Beyond device volatility, performance becomes another element in the selection criteria. But how is it possible to determine performance and functionality without testing or, at least, simulating the device? Actually, the performance a design achieves depends on how it is implemented within a device. A qualified estimate of how the design might be implemented usually can be made once the architecture of the device is understood. But while some manufacturers claim that their device performance is entirely predictable, estimates are valid only insofar as the P&R performs as expected. It may be easy to anticipate how the FPLD software will implement a state machine or counter. It is quite another thing to guess how it will accommodate an adder or other large combinatorial function.

The maximum clock frequency is often determined by the number of logic levels required between flip-flops, the pad-to-flip-flop time, or the flip-flop-to-pad time. The last two times may be found in the device data sheet. Estimating the number of levels of logic in the longest path in a design requires more thought. The number of terms and/or variables driving the input to a flip-flop is one consideration.

Devices such as PALs, that have AND-OR array feeding flip-flops, can implement terms containing large numbers of variables as easily as terms containing only one. Other devices have a set number of variables that its logic elements can accept. Terms that exceed that number require more than one element, causing elements to be cascaded to implement the logic. Cascading logic increases the delay path. Moreover, some AND-OR array devices have a fixed number of terms in their arrays. If this number is exceeded, the extra terms must be implemented elsewhere and brought to the array plane. Bringing in terms from outside the array causes additional delay.

Routing delays are another consideration. Devices that fall into the FPGA (field-programmable gate array) camp have delays that increase with fanout. The amount of additional delay due to fanout is published by the manufacturers in tabular or equation format. Other, more fixed architectures like PALs, don't exhibit fanout-dependent delays, but have the problem of fixed routing paths whose delays cannot be improved.

In either case, however, post P&R timing analysis is required to determine the anticipated delays. For example, a path for one application might pass through blocks that were not needed for a different circuit, thus adding path delays. Thus, designers need to carefully anticipate path differences and resultant delays for each application.

The Path to a Masked Device

While FPLD devices have proven themselves as cost-effective production vehicles in all but extremely high volume applications (greater than 10,000 units per month), many designers are taking advantage of the FPLD's fast device turnaround times and are employing them as prototyping vehicles as well. Indeed, it is often easier and faster to develop an FPGA for a prototyping environment than it is to implement the equivalent system design using discrete logic devices on a printed circuit board. As such, many designers today use programmable devices for design prototyping before migrating the design to a masked array or standard cell for production. Use of an FPLD as a temporary solution is an attractive alternative to the time and expense of developing a masked device. The programmable device may even be used to sustain production until demand for the product justifies the masked solution. This strategy can effectively shorten product introduction time by months.

The ease with which the design may be migrated from one technology to another determines the risk, cost, and time involved in making the transfer. In general, the closer the architecture of the device to that of a masked device, the easier the migration path. Some gate array vendors have software that converts netlists of programmed devices to masks. The impact that the choice of a programmable device would have on a conversion should be considered as early in the life cycle of the product as possible.

The Gate-Density Confusion

Perhaps no other criterion has created as much controversy as has the issue of device density of FPLDs. Considering the difficulty of comparing different technologies, the issue is understandable. Indeed, accurate assessments of gate densities is crucial in not only design but also in comparative cost analysis.

Because FPLD device manufacturers often use conventional masked gate arrays and related density measurements as points of comparison, any similarities that they can make are often drawn, and the resulting marketing hype adds to the confusion. Specifically, many FPLD manufacturers equate the density of their devices in terms of "number of gates." Unfortunately, manufacturers all don't count gates the same way. Other FPLD manufacturers don't refer to gates at all. As a result, FPLD device comparisons using density measurements are error prone.

The problem doesn't exist when comparing conventional gate arrays. Masked gate array vendors specify their devices as having a certain

number of two-input NAND gates and the percentage that can be actually utilized in a design. With masked arrays, higher-input gates and other logic functions are made from specified numbers of these two-input gates. Such comparisons cannot be made with FPLDs because each manufacturer uses a unique architecture and functional logic module, or block within the device to implement various circuit functions. Indeed, it is very difficult for designers who are used to designing with conventional logic, and even those who have had experience with masked gate arrays, to understand what they can do with a more complex type of logical element or module. Clearly, designers need a more accurate way to gauge FPLD device densities.

Only Two Basic Device Types

The first step toward achieving that goal is to understand that despite what vendors claim, there are only two basic types that fall into the FPLD category — programmable logic devices (PLDs) and field-programmable gate arrays (FPGAs). Understanding which devices are one or the other is critical to proper device selection. For example, PLDs have a fixed interconnect architecture, comprise AND-OR array planes that feed flip-flops and usually exhibit static power dissipation. Devices in this category include those with labels such as PALs, FPLAs, EPLDs and EEPLDs.

The distinguishing characteristics among the different vendors, aside from programming specifics (fuse, EPROM, etc), is the number of flip-flops that the device contains. An FPGA, on the other hand, has a flexible interconnect technology with no fixed AND-OR plane, but does incorporate an array of logical building blocks that are often structured loosely around a masked gate array architecture. PGAs and LCAs are names some manufacturers use to describe their specific type of FPGAs.

It's important for designers to keep in mind that there are areas of similarities between FPGAs and PLDs that often add to the confusion. For example, both categories of devices incorporate combinatorial logic that feed other logical devices (flip-flops, gates, etc). The combinatorial logic that exists with FPGAs, however, is more flexible and more easily configurable than that within the typical PLD. In addition, the FPGA's combinatorial logic is in more of an array configuration, so it is interconnected differently.

The Elusive "Equivalent Gate"

Once the PLD/FPGA distinction is made, the second step in gauging FPLD device densities involves understanding device architectures to determine capacities rather than relying on manufacturers' stated density claims. To be specific, the question of gate densities or capacities with either category can be reduced to analyzing the amount of logic that can be realized from the logic building blocks in the device, and the amount of blocks on the device that actually can be used in a design.

To facilitate the analysis, the concepts of Gate Array Equivalent Gates and PLD/LCA Equivalent Gates needs to be defined. A gate array equivalent gate is the two-input NAND gate mentioned earlier and can be considered as a real or usable gate.

A PLD Equivalent Gate corresponds to the number of gates that are assigned to the device by the manufacturer. The number may reflect the total number of gates on the device, or it may be based on some assumptions about the effective use of gates. Each building block on an FPGA contains an inherent number of gates. The way the building block is used in a design, as well as the architecture of the device itself, determines how many of the gates can actually be used to implement logic. The distinction doesn't arise with masked gate arrays because the lowest level of logic is the gate itself. When the level of granularity of the logic element is higher, as it is in all PLDs and FPGAs, the question of how many of the gates on the device can actually be used is important.

One Gate, Two Gates, Three Gates, . . .

With this understanding in mind and armed with information derived from material supplied by manufacturers, consider the counting methods and derived device densities of two representative FPGA vendors - Actel and Xilinx - and PLD supplier, Altera.

Actel determines the gate capacities of its FPGAs by comparing actual designs and library macros with the same logic implemented on a masked gate array. Actel's ACT™ 1 architecture is similar to a channeled gate array in nature. Because the gate counts for masked gate array designs are easily determinable, Actel uses the masked arrays as a convenient standard to characterize its devices in gate-array equivalent gates. The Actel architecture consists of an array of logic modules with the rows and columns separated by routing tracks. While it is more complex than a gate, the Actel logic module is much simpler than the basic element of competing architectures. Based on comparisons to masked gate arrays and analysis of actual designs over the past two years, in practice Actel's module is equivalent to 3.2 gates.

The FPGAs from Xilinx, meanwhile, are based on a Configurable Logic Block (CLB) architecture and the vendor counts gates using another technique. Each CLB contains two dedicated flip-flops and a combinatorial function generator. The generator is a static RAM that is loaded during the configuration process. Five inputs are used to address the SRAM so that it may implement any five-variable function or two four-variable functions with some restrictions.

Xilinx I/O Blocks (IOBs) contain an input latch, an output flip-flop and a three-state output buffer. When

Xilinx calculates the densities of its devices, the company counts all the gates in all the IOBs on a device as usable gates. The total number of gates in an IOB is said to be 13. Xilinx determines the density of their devices by multiplying the number of CLBs and IOBs in the device by the number of gates they are claimed to represent. The function generator on the CLB is evaluated as two four-input generators. Each is specified as having a maximum gate count of between 20 to 22 gates and a minimum of one. There would thus be between 2 and 44 gates per CLB function generator. The vendor then assigns an "average number of gates" to the function generator based on an assumed 50 percent utilization efficiency.

The CLB is rated as having a total of 42 gates, 30 gates from the function generator and 12 gates from the two flip-flops. These gates account for nearly 88% of the total number of gates on the XC3000 series devices. After multiplying all possible gates in the CLBs and IOBs on the device the products are summed to give the total number of gates. A fraction of the total (between about 50% and 65%) is given as the "usable gate count" by the manufacturer. This arbitrary approach is similar to that applied to PLD architectures.

Altera's MAX architecture represents perhaps the most architecturally sophisticated PLD and the subject of this comparison. The largest device in that family, the EPM5128, incorporates 128 flip-flops (the typical PAL has only eight) and a bus structure that differs from conventional PLDs. The firm's PIA (programmable interconnect array) bus allows the company to interconnect more cells together than conventional PLDs, but the architecture of the basic unit that designers must deal with has not changed. It is the AND-OR macrocell. The architecture of the device consists of Logical Array Blocks (LABs) that can be interconnected via the PIA bus. The bus acts to connect LAB outputs to other LAB inputs.

Each LAB contains 16 macrocells and an expander. The macrocell contains an AND-OR structure feeding into an exclusive OR gate which, in turn, drives a flip-flop. There are also inverter/buffers in the macrocell to allow for true or complementary logic. The expander produces additional terms which may be fed to any of the macrocells in its LAB. Altera doesn't specify gate counts for its device but rather uses macrocells to characterize capacity.

Trust Designs, Not Vendor Claims

Although the accuracy of the gate densities derived from the counting schemes itemized above will not be disputed, designers are well advised to disregard manufacturers' claims. Instead, by employing the analysis techniques and equations that follow, system designers will be able to accurately gauge competitive device densities by understanding the various architectural distinctions and by analyzing their own designs. Because the ultimate goal is to measure the number of gates a designer can expect to be able to use on an FPLD/LCA device rather than basing this number on assumptions or small-scale macro implementations, prior to the evaluation that follows, it is interesting to review the results of actual designs that were analyzed relative to Actel, Xilinx and Altera devices as to gate usage. The results will serve as a check on the gate-counting equations being presented later in the discussion. The results of these tests are open to scrutiny by any who request it.

For Actel, nine customer designs were randomly selected and a group of TTL macros from the macro library for analysis. The designs included a DRAM controller, 32-bit accumulator, graphics controller, data transfer controller, and a SCSI interface controller. The TTL macros included counters, multiplexors, and decoders. Table 1 lists the macros analyzed.

TABLE OF TTL MACROS ANALYZED IN ACTEL AND XILINX LIBRARIES
3-to-8 Decoder
8-to-1 MUX
4-to-1 MUX
74161
74164
74194
74377
74139

Table 1.

Four customer designs were similarly randomly selected and data for Xilinx along with an AMD (a Xilinx second source) published application article and the same TTL macros selected from the Actel library. The Xilinx designs included a keyboard interface controller, an RLL2,7 coder, and a video controller.

Counting gates for Altera devices was a different situation. When the Altera MAX+PLUS development system compiles a design it uses the resources on the device to implement it, but the details of the implementation are hidden. When a macrocell is reported by the development system as having been used there is no way to know how much of its logic was used. It could range from a few gates to all the logic and the flip-flop in the macrocell. What is required to accurately measure the capacity of the device would be designs whose schematic gate counts were known and, more importantly, that are known to have utilized virtually all the resources on the chip.

Fortunately, four such examples exist. These are some capacity benchmark studies performed by Altera. The four designs were examples of common logic circuits that were chosen so as not to favor any type of architecture over another. The designs include a Data Path, Timer Counter, State Machine and Arithmetic functions. The benchmark measures how many of each type of design could be accommodated by the device being tested. Because the

number of gates in each of the designs is known, as well as the number of the designs that could be placed on a fully-loaded device, the capacity of the device can be found by simply multiplying the gates per design by the number of designs.

Test Results Analyzed

The designs were analyzed by counting the number of gates and building blocks in them. The gates were next categorized according to whether they were combinatorial gates, flip-flop (i.e. flip-flop or sequential latch) gates, or combinatorial gates associated with a flip-flop. The last category refers to combinatorial gates that immediately precede the D input of a flip-flop in a design. The classification is significant because it enabled the equations that follow to be developed that allow designers to estimate the number of gates that could be accommodated on a device. The estimate would be based on the number of flip-flops in the design as a percentage of the total. The designs were used to verify the formula as a predictive tool.

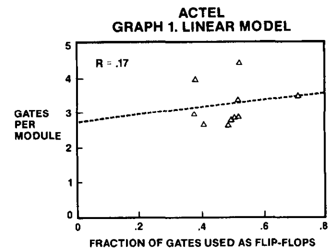
For consistency, individual logical elements were counted (source: LSI Logic HCMOS Design Manual, 1986) as follows:

Element	Gates
Inverter	.5
Buffer	.5
Gate, two-input*	1.0
Gate, three-input	1.5
Gate, four-input	2.0
Gate, five-input	2.5
Gate, exclusive-Or	3.0
MUX, 2:1	3.0
MUX, 4:1	6.0
Latch	4.0
Flip-flop, D	6.0
Flip-flop, JK	10.0
Adder, full	10.0

(* Note: Bubbled inputs on gates were counted as .5 gates unless all inputs were bubbled which counted for none.)

Table 2 shows the gate counts for the various Actel designs. The designs are ordered from the lowest number of gates per module to the highest as calculated by simply dividing the total number of gates in the design by the total number of modules. As may be seen in Graph 1, there is variation in the utilization in the modules with the average being about 3.2 gates per module.

Table 3 shows the gate counts for the various Xilinx designs. Due to the large number of flip-flops in the Xilinx architecture (there are two per LCA which is why the fraction



Design	FFs	FF Gates	Combinatorial Gates	Total Gates	Modules Used	Gates per Module
Design 1	24	144	116	260	99	2.63
Design 2	143	672	846	1518	512	2.96
Design 3	171	835	461	1296	451	2.87
Design 4	162	962	572.5	1534.5	436	3.52
Design 5	42	252	628.5	880.5	220	4.00
Design 6	22	132	131	263	98	2.68
Design 7	127	714	622	1336	477	2.80
Design 8	178	1103	486.5	1589.5	549	2.90
Design 9	127	740	833.5	1573.5	469	3.36
Design 10	84	356	412	768	172	4.47
				Average		3.22

Table 2. Actel Design Gate Counts

Design	FFs	FF Gates	Combinatorial Gates	Total Gates	CLB Used	Gates per CLB
Design 1	38.00	228.00	186.50	414.50	58.50	7.09
Design 2	24.00	144.00	123.00	267.00	34.00	7.85
Design 3	74.00	448.00	372.00	820.00	104.00	7.88
Design 4	32.00	192.00	149.00	341.00	44.00	7.75
Design 5	99.00	594.00	319.00	913.00	97.00	9.41
Design 6	167.00	1136.00	399.50	1535.50	136.00	11.29
				Average	.76	8.55

Table 3. Xilinx Design Gate Counts

of flip-flops to CLBs ranges from zero to two) it was expected that designs would achieve densities in proportion to the number of flip-flops used. As may be seen in Graph 2, there is a strong (98%) correlation between the fraction of the total gates in the design that are used to implement flip-flops and the number of gates realized per CLB. The average number of gates per CLB for the designs analyzed was 8.5. Table 4 shows the number of gates for each Xilinx device at 8.5 gates per CLB. It also includes CLB utilizations that are typical according to the Xilinx design guide.

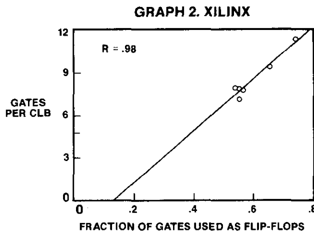
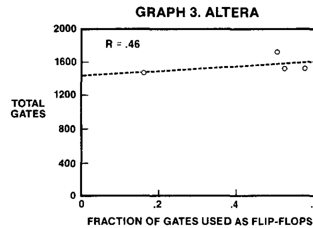


Table 5 shows the gate counts for the Altera-generated benchmarks. The average was 1559 gates. Graph 3 plots the total number of gates in the benchmark against the fraction of gates used as flip-flops. Another data point provided by Altera comes from the data sheet for the EPM5128, where it claims that an equivalent to a 74161 counter occupies 3% of the device. The counter requires 54 gates to implement, which would put the capacity of the device at 1800 gates.



Architectures Determine Actual Device Density

Because there are almost as many ways to count gates as there are manufacturers, had additional devices been analyzed, alternate gate counting methods would have been described. With these test results as a guide, the analysis techniques and formulas that follow can be used by designers to determine how their application-specific design will fit on the various devices available, regardless of density claims.

PLDs will be considered first, to see how much logic they can accommodate and what types of designs fit best on that architecture. In general, the typical PLD architecture contains a huge number of combinatorial functions and only relatively few flip-flops. The limiting factor with this technology is the number of flip-flops.

Designers must first analyze their own designs to determine the ratio of combinatorial two-input gates to flip-flop gates, keeping in mind that a flip-flop function represents six two-input gates. For simplicity, treat a three-input gate as 1½ two-input gates, and a four-input gate as two gates. Next, the following formula can be used to determine if the intended PLD will accommodate the design. The equation accurately determines the density of any PLD device and is more precise than gate count claims because it accounts for flip-flop limitations of PLDs.

Device	CLBs	Typical Gates per CLB	Total Gates	Device Utilization	Usable Gates	IOB	Gates per IOB	IOB Gates	Total Gates	Advertised Gates
XC3020	64	8.5	544	0.85	462	64	3.25	208	670	2000
XC3030	100	8.5	850	0.8	680	80	3.25	260	940	3000
XC3042	144	8.5	1224	0.75	918	96	3.25	312	1230	4200
XC3064	224	8.5	1904	0.7	1333	120	3.25	390	1723	6400
XC3090	320	8.5	2720	0.65	1768	144	3.25	468	2236	9000

Table 4. Typical Capacity of Xilinx Devices

Design	FF	FF Gates	Combinatorial Gates	Total Gates	Fraction FF Gates	Designs per Device	Gates per Device
Arithmetic	8	48	245	293	0.16	5	1465
Timer Counter	24	144	143	287	0.50	6	1722
State Machine	12	72	67	139	0.52	11	1529
Data Path	16	96	73	169	0.57	9	1521
						Average	1559

Table 5. Altera Benchmark Gate Counts for EPM5128

Gt = ((6 gates x #FF)/PercentFF))* (Df), where:

Gt = upper gate limit on the PLD
#FF = the total number of flip-flops in the PLD
PercentFF = the fraction of gates that reside in flip-flops of the actual circuit
Df = a derating factor, or maximum device utilization factor available on manufacturers' data sheets. For PLDs, Df is equal to approximately 0.9.

The term, PercentFF, is a useful way of distinguishing among designs and is relatively easy to estimate based on the number of flip-flops in a design and the approximate total gates. Independent research conducted by Stanford University on scores of logic designs indicates that approximately 45% of all gates used in logic designs are used to implement flip-flops and the remainder is used for combinatorial logic. According to the LSI Logic handbook, six two-input gates are required to implement a flip-flop. Thus, designers can use 45% as a representative number if knowledge about their own designs is absent. Thus, using the equation and this 45% figure, the upper bound on the capacity of a PLD containing 128 flip-flops may be found to be:

Gt = ((6x128)/(.45))(0.9) = 1700(0.9) = 1530 gates.

Thus, a PLD design that uses all the flip-flops available could expect to achieve densities of no greater than 1700 gates (Df = 1.0), regardless of what the manufacturer states when PercentFF is 0.45. For values of PercentFF between 0.35 and 0.55, Gt ranges from 1975 to 1250. This number is about a third the density claimed for such devices by some PLD vendors and consistent with the actual test results detailed in Table 5. While it might be possible to find a design that uses the fixed PLD architecture more effectively by creating flip-flops from combinatorial logic, such circuits are generally much slower than built-in flip-flops and are not generally usable.

The FPGA architectures can be evaluated using similar equations to determine capacities. As indicated, the two types of FPGA architecture available are the LCA and ACT 1 family of channeled gate arrays.

LCAs that employ dedicated flip-flops have a better balance between combinatorial logic and flip-flops than PLDs, but for most designs, there are more flip-flops than necessary and combinatorial logic resources, thus, become the limiting factor in logic density instead of flip-flops as with PLDs.

The designer can determine how well his circuit will fit on an LCA device by examining his schematics to see how many inputs and how many gates were used on the logic feeding a flip-flop. For example, if there is a four-input block, how many gates are typically feeding that block : 3, 4, 5? In the selection of vendor-generated TTL macros analyzed in the previous section, there is an average of 3.6 combinatorial gates per CLB. If four or fewer inputs were used most of the time, then the average number of gates prior to flip-flops can be used in the following formula to find the upper bound for the gate utilization on an LCA (if you need more than four, say, five, you get fewer gates and you need two blocks):

Gt = ((#ff x #CLB x #pre-FFgates)/(1-PercentFF)) (Df), where:

Gt = upper gate limit on the LCA
#ff = number of flip-flops per CLB
#CLB = number of logic blocks in the device
#pre-FFgates = number of two-input gates before the flip-flop (e.g. 3.6)
(1- PercentFF) = fraction of gates in the user's combinatorial logic
Df = a derating factor, or maximum device utilization factor available on manufacturers' data sheets. For LCAs, Df ranges between .55 to .85.

(Note: The formula ignores gates in I/O blocks. Flip-flops in I/O generally improve performance, and effectively increase routing resources rather than gate count because there are no combinatorial functions in the I/O for use with those flip-flops. For designs with PercentFF in the range 0.35 to 0.55, LCAs are usually combinatorial gate limited.)

Using this formula on an LCA with 144 CLBs and assuming an average of three pre-FFgates, 45% of the gates reside in the flip-flops, and a utilization of 75%, a sample calculation would be as follows:

Gt = ((2x144x3)/.55).75 = 1180 gates.

Thus, an LCA with 144 CLBs and a claimed gate density of 4200 gates has a maximum 1180 usable gates, and only 1570 if fully utilized (Df = 1.0, under the above conditions). The results of this equation show results somewhat higher than found in the actual design of Table 4.

Similar capacities can be calculated with ACT 1 devices. The ACT1 architecture uses a large number of relatively small building blocks and has no dedicated flip-flops. Therefore, neither the PLD nor LCA formulas detailed above can be used. The ACT 1 architecture isn't constrained either by flip-flops or combinatorial logic. However, certain assumptions can be made and a formula derived that will also give designers consistently accurate capacity calculations. As in the previous cases, assume 45% of all gates reside in device flip-flops. Again, other values may be used based on actual design characteristics.

Consider, for example, the ACT1020, a 2000-gate FPGA. This device incorporates a total of 546 logic modules and two are used to implement one flip-flop. Most gates, multiplexors and latches use one module. With these architectural considerations in mind, the following formulas can be used to give designers a fairly accurate capacity reading on an ACT 1 device:

#LM = #MF + #MC   Eq. 1

#MFx3.4/(#MFx3.4 + #MCx3.5) = .45   Eq. 2

Gt = (#MFx3.4 + #MCx3.5)(Df)   Eq. 3

where:

#LM = total number of logic modules on an Actel device
#MF = number of modules used for flip-flops
#MC = number of modules used for combinatorial functions
Gt = upper gate limit on the LCA
Df = the module utilization factor.

Equation 1 states that the sum of the combinatorial and sequential modules equals the total number of modules.

Equation 2 states that the assumption that 45% of the gates in an average design are used as flip-flops.

Equation 3 computes the expected number of gates used in an ACT 1 device.

The logic module can implement one to five combinatorial gates with 3.5 being typical. Df is listed by Actel as ranging from .85 to .95, so .90 will be used for these calculations.

Two logic modules can be used to implement a flip-flop and some combinatorial logic. The sequential modules in the designs in the previous section attained 3.4 gates/module.

Therefore, the 2000-gate ACT 1 device with 546 modules:

Solving Eq. 2 yields:
#MF = .84#MC.
Substituting this result into Eq. 1 gives:
546 = 1.84#MC
#MC = 296 and #MF = 546-296 = 250.
Finally:
Gt = (296x3.5 + 250x3.4)(.9) = (1886)(.9) = 1697 gates.

It should be noted that the results of this equation are also consistent with the tested gate counts determined for Actel devices in Table 2 when the 3.4 gates/module utilization (approx.) factor is used.

Using these formulas for PLDs, the results indicated in Table 6 were calculated for a representative number of devices from eight different manufacturers.

Evaluating Development Tools

While the equations enable designers to evaluate architectures, a hands-on approach is often preferred by engineers. And although device manufacturers supply development systems that allow designers to analyze the design and program the devices here too, as with technology and density comparisons, care must be taken to achieve accurate results. For example, when doing a competitive analysis of development systems, beware of trying to save time by doing an identical small design on each. A small design is one that uses only a small fraction of the chip. To see what the system and the device can really do, it is important to test them to their limits. A simple example may make device development and programming tools appear to be stronger than they actually are.

Most systems vary a great deal in cost and capability. Without becoming familiar with a system it is difficult to understand its capabilities. Some of the most important features to examine are the place and route software, timing analysis tools and platform support.

The P&R software is the heart of

Vendor	Device	Dedicated Flip-Flop[2]	Fraction Utilized[1]	No Gates[4]	No I/O	Reprogrammable
Actel	1010	0	.90	916	57	N
	1020	0	.90	1697	69	N
Altera	5064	64	.90	770	36	Y
	5128	128	.90	1530	60	Y
AMD	16R8	8	1.0	105	16	N
	22V10	10	1.0	135	22	N
Atmel	2500	48	1.0	640	38	Y
ICT	7024	40	1.0	530	22	Y
	7040	48	1.0	640	30	Y
Plus Logic	2020	144	.9	1255	72	Y[3]
Signetics	2552	52	1.0	575	53	Y[3]
Xilinx	3020	128	.9	630	64	Y
	3030	200	.85	930	80	Y
	3042	288	.80	1180	96	Y
	3064	448	.70	1860	120	Y
	3090	640	.65	2270	144	Y

Table 6. Usable Gates

the development system. The software assigns the logical resources of the device to implement logic functions described by the design netlist. It uses the routing resources (i.e. tracks and interconnect elements) to connect the logic functions together. The extent to which the software succeeds will determine the logic capacity of the device and, to a large extent, its performance. All FPGA/ PLD vendors include P&R software with their systems, but the software's ability to implement designs is constrained by the power of its algorithms, the architecture of the device, and the amount of interconnect resources available.

Place and route on a PLD is simpler and faster than on an FPGA due to the less flexible architecture of PLD devices. PLDs usually place and route successfully unless the design exceeds the resources of the device.

The FPGA architecture allows for great P&R flexibility, but requires more of the software because there is more to do. The router may fail to complete successfully requiring that design be changed or, if allowed, routed by hand. The designer has much more control over FPGA implementations than PLDs by being able to assign placement directly or attaching criticality to nets in the design.

FPGAs require that less of the device area be devoted to wiring resources than PLDs. As devices scale to higher levels of integration, PLDs will require a larger proportion of the device for wiring than FPGAs. For this reason, FPGAs will be better able to sustain larger device densities.

The speed with which the router operates is also important. If each design iteration requires many hours to run the P&R tool, then it significantly impacts the design cycle time. PLD routing times are short as there is relatively little for the software to do. Routing times for FPGAs may range from minutes to hours depending on the routing resources available, the sophistication of the software and the complexity of the design. The problem is exacerbated if the software runs for a long time and still fails to route some nets. At that point the designer may be forced to P&R the entire design manually.

Manual P&R is tedious and time consuming. When manual intervention is required, every design iteration requires the same amount of time as the first and the development time is multiplied by the number of iterations. While manual P&R allows the designer complete control over the design implementation, it requires him to have detailed knowledge of the physical architecture of the device. Nevertheless, FPGA architectures and tools are available that permit fully automatic placement and routing in under 30 minutes.

Designers have little control over the delays from the P&R in PLDs. Moreover, the fixed path delays limit any opportunities for timing optimization. PLD delays do not vary with load, but are determined by the individual delays of the device elements that make up the path. There's more variation in FPGA delays than in PLDs and the delays are load dependent. Architectural flexibility allows for delay optimization which may be done manually or by using software tools.

The timing analysis tools available with the various systems vary considerably in both the quality and quantity of the information they provide about the system. The timing analysis tools that come with the development system should allow the designer to examine timing details of the design. If there is a timing problem, the timing analyzer should be specific enough to allow the problem to be pinpointed so the design or the P&R may be modified. The more commands available and the more powerful the commands are the faster the designer can analyze the design.

If the vendor-specific software supports other CAE systems, there should be a way to back-annotate the delay information to the CAE system simulator. Back-annotation allows the design timing information to be seen graphically and under simulated operating conditions.

Finally, designers need to consider platform support. Designers who already have invested in hardware for a CAE system may also want to use it to develop programmable devices. Designers need to make sure the vendor's interface package for his workstation is complete. For example, does it have back-annotation available for effective use of the simulator.

The End of the Line

To be sure, PLDs have changed dramatically. Fortunately, with a little common sense and the aid of the guidelines and formulas detailed here, designers can wade through the hype and mounds of vendor literature to determine the best FPLD for the application. It's important to keep in mind that all devices are different and can suit a variety of applications. Successfully choosing one category of device for Design A doesn't mean it will be the right device for Design B. Every application requires a recalculation and close scrutiny of the circuit. By closely examining the device technologies, development tools and, above all, gate capacities, the optimum device can easily be selected.

Notes:

1. Based on design experience. Smaller devices can usually use all available flip-flops. Larger devices cannot always take advantage of embedded flip-flops.
2. Actel devices contain no dedicated flip-flops. Any logic module may be used as a latch and any two modules may be used as a flip-flop.
3. These devices contain input flip-flops which cannot be driven by internal combinatorial logic. The input flip-flops are counted only for the gates they contain.
4. All gate counts assume 45% of users' gates are used for flip-flops.

Actel

Article Reprint 2

A CMOS ELECTRICALLY CONFIGURABLE GATE ARRAY

by Khaled A. El-Ayat, Abbas El Gamal, Richard Guo,
John Chang, Ricky K. H. Mak, Frederick Chiu,
Esmat Z. Hamdy, John McCollum, Amr Mohsen

Reprinted from
IEEE JOURNAL OF SOLID-STATE CIRCUITS
Vol. 24, No. 2, April 1989

4

© 1990 Actel Corporation

Actel

A CMOS Electrically Configurable Gate Array

KHALED A. EL-AYAT, ABBAS EL GAMAL, SENIOR MEMBER, IEEE, RICHARD GUO, JOHN CHANG, RICKY K. H. MAK, FREDERICK CHIU, ESMAT Z. HAMDY, MEMBER, IEEE, JOHN McCOLLUM, AND AMR MOHSEN, SENIOR MEMBER, IEEE

***Abstract* —A novel CMOS electrically configurable gate array is described. The chip combines the flexibility, efficiency, extendability, and performance of mask-programmed gate arrays with the convenience of user programmability. The implementation is facilitated by a novel two-terminal antifuse programmable element and a new configurable interconnect technology. The chip has been fabricated using 2-μm n-well CMOS technology with two-layer metallization.**

I. Introduction

Mask-programmed gate arrays have gained wide acceptance and are used in an increasingly larger number of customer-specific applications. This is primarily due to a) their architectural flexibility that allows efficient implementation of a wide range of applications, b) large gate densities of 100K gates or more [1], [2] which permits high levels of integration, and c) high performance. The architectural flexibility is due to the use of primitive basic cells and a routing capability that allows the construction of a large set of macrocell building blocks and general wiring interconnect between the macrocells. Gate arrays, however, suffer from long and costly development cycles which are barriers towards wider usage in many applications, especially those with low-volume requirements or restricted to short development cycles.

Programmable logic devices (PLD's) [3]–[6], on the other hand, offer instant turnaround in the design cycle. However, their architecture and technologies limit their efficient integration of many applications to only hundreds of gates. Some PLD architectures [7], [8] have integration levels of several thousand gates, but due to architectural and interconnect inefficiencies, the number of usable gates is a small fraction of the total available gates.

This paper describes a CMOS electrically configurable gate array that combines the architectural flexibility and efficiency of gate arrays with the convenience of user programming available from PLD's. The implementation is facilitated by a new configurable interconnect technology [9] based on a novel one-time, two-terminal, programmable, low-impedance circuit element [10]. The electrical characteristics of this element are more suitable for on-chip integration than other previously published antifuses (e.g., [11]). The chip has been fabricated using a 2-μm n-well CMOS technology with two layers of metallization.

Manuscript received July 28, 1988; revised November 14, 1988.

K. A. El-Ayat, A. El Gamal, R. Guo, J. Chang, E. Z. Hamdy, J. McCollum, and A. Mohsen are with Actel Corporation, Sunnyvale, CA 94086.
R. K. H. Mak and F. Chiu are with Data General Corporation, Sunnyvale, CA 94086.

IEEE Log Number 8926932.

II. Chip Architecture

A. Overview

The chip has a channeled gate array architecture (Fig. 1) consisting of configurable logic modules organized in rows and columns and separated by wiring channels (vertical wiring channels were omitted from Fig. 1 for clarity). Unlike gate arrays, however, the channels contain predefined segmented metal tracks of different segment lengths to accommodate the routing requirements. Antifuse elements are located at the intersection of the horizontal and vertical wire segments and between adjacent track segments. Circuit connections and module configuration are established by programming the appropriate antifuse element which then forms a low-impedance connection between the required two metal segments. The concept is similar to vias between the first and second level of metallization in double-metallization technology. The logic module is configurable and was carefully selected to implement a large set of combinatorial logic cells as well as latches and flip-flops for sequential circuit applications.

The chip also has configurable I/O buffers that can be configured as input, output, or bidirectional I/O. Also shown in Fig. 1 are the periphery circuits necessary for addressing, programming, and testing the array.

B. Wiring

Wiring channels in this architecture consist of a set of segmented tracks of various lengths. A typical channel track segmentation scheme is shown in Fig. 2, with vertical tracks mostly omitted for clarity of the diagram. The lengths of the track segments vary from a minimum of two

0018-9200/89/0600-0752$01.00 ©1989 IEEE

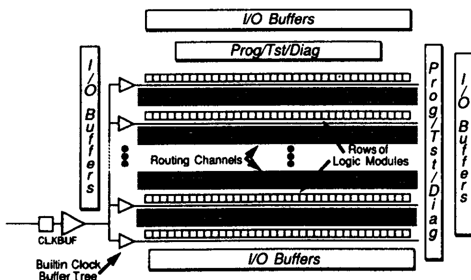


Fig. 1. Block diagram of the chip.

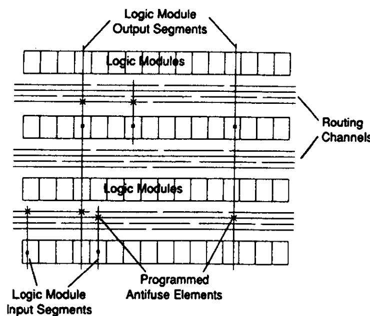


Fig. 2. Wiring channels and track segmentation.

module spans and progressively increase until they span the entire length of the array. Short segments offer more routing flexibility because their granularity affords a large number of their type. Their disadvantage is that longer nets suffer a degradation in performance due to the addition of interconnect antifuses in the circuit path. Careful choice of the lengths of the horizontal and vertical wire segments as well as the number of tracks in each channel provides routing flexibility comparable to gate arrays [12]. The trade-offs between routing flexibility, performance, and silicon area was carefully weighed in designing the channel and its track segmentation. The number of tracks per channel in this architecture is only slightly higher than mask-programmed gate arrays with the same gate density.

The similarity of this architecture to gate arrays can be best understood by drawing a parallel between the programmable antifuse elements and the vias introduced during the fabrication of gate arrays. The small size of the via allows gate arrays to contain an extremely large number of potential via sites, of which a very small fraction is needed to implement any application. Similarly, the size of the antifuse elements permit packing them as close as the metal pitch of the employed technology. Thus, a very large number of these elements can be included in the chip. As an example, a typical 2000-gate mask gate array has approximately 90 000 potential via sites compared with 200 000 antifuse sites in the chip family described in this paper. Again, only a few percent of these antifuse elements need to be programmed to implement any application. This allows gate array-like extension of the chip architecture, by adding more logic modules and incrementally increasing the number of wiring tracks in each channel.

The interconnect architecture of this chip is further illustrated by Fig. 3, which shows a small section of the array. Two partial rows of modules (each row containing three modules) with vertical and horizontal track segments are shown. Antifuse elements are located at the intersection of the track segments as well as between adjacent horizontal and vertical wire segments. To program an antifuse element, 18 V is applied across its terminals while all other elements are subjected to half that voltage. Two examples of circuit path connections are shown in Fig. 3. Path $P1$ is a short path in which a vertical segment is connected to a horizontal track by programming antifuse $F1$. The path is completed by programming a second antifuse $F2$ which connects the horizontal segment to another vertical segment. The second example path $P2$ illustrates how arbitrarily longer nets can be constructed in this architecture whenever the desired single segment length is not available. Antifuses $F3$ and $F5$ are programmed in a similar fashion to path $P1$. Antifuse $F4$, however, is programmed to connect two adjacent horizontal segments to form a new longer horizontal segment. The isolation n-channel transistor $T1$ is part of the addressing and decoding circuitry used to address, decode, and program the required antifuse elements. When programming antifuse $F4$, $T1$ must be turned OFF to supply the full programming voltage across its terminals.

C. Logic and I/O Module

The interconnect architecture described in this paper can be used with a variety of logic modules. The choice of the module involves trade-offs between logic capability, routability, performance, and silicon areas. The configurable logic module used in this architecture is shown in Fig. 4. It has eight inputs and one output, and was chosen for its efficiency in implementing both combinatorial and sequential circuits and for its optimum utilization of routing resources. The module implements a modified 4:1 multiplexor function with inputs $A-D$, select inputs SA, SB, $S0$, and $S1$, and output Y. To implement the required logic function, the module is configured as the desired macrocell by simply programming the appropriate antifuses at its input terminals to connect the inputs to the required nets or to V_{DD}/V_{SS}.

The module is capable of implementing all two- and three-variable functions and some four-variable functions such as a 4:1 multiplexor. It can also be configured to implement latches and flip-flops. One module is needed to implement a latch while two modules are needed to implement a flip-flop. No predetermined hardwired latches or flip-flops are implemented or needed in this gate array. Latches and flip-flops may be implemented anywhere in this array to suit the requirements of the application. The required macrocell connections are simply placed and in-

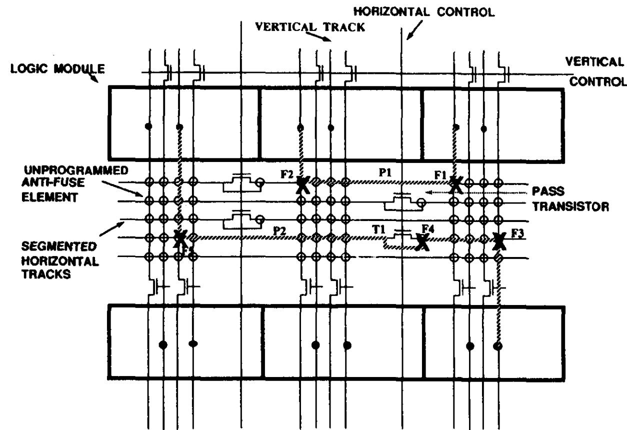


Fig. 3. Interconnect scheme with typical routing examples.

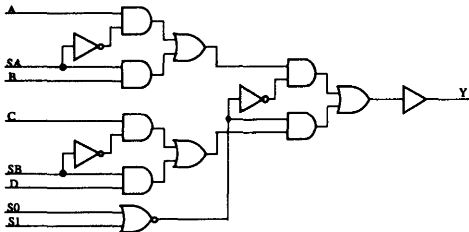


Fig. 4. Logic module schematic.

TABLE I
TYPICAL MACROS AND THEIR EQUIVALENT MODULE IMPLEMENTATION

Logic Function	Modules
3-input OR	1
4-input AND	1
2-input XNOR	1
D latch with clr	1
DFF with Pre & Clr	2

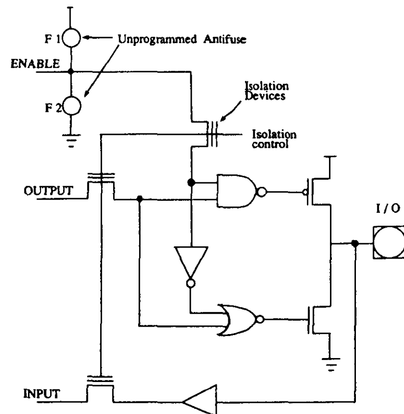


Fig. 5. Bidirectional programmable I/O module.

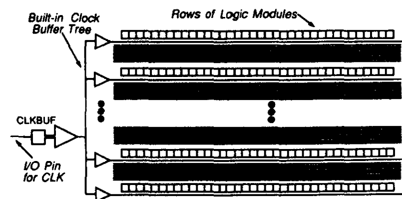


Fig. 6. Low-skew clock distribution network.

terconnected wherever needed in the array. A sampling of typical macros is shown in Table I. Also shown are the required number of modules needed to implement the macro.

The I/O architecture is also quite flexible as shown in Fig. 5. Each I/O module may be configured as input, output, or bidirectional I/O by simply programming the appropriate antifuse elements. The I/O module is configured as an input buffer or output buffer by simply programming antifuses $F2$ or $F1$, which connect the *ENABLE* input to V_{SS} or V_{DD}, respectively. To configure it as a bidirectional I/O the *ENABLE* line is driven from the appropriate net in the array and the two antifuses are not programmed. Also, any I/O module can be driven from virtually any net in the array. Naturally, appropriate placements of the I/O relative to the application circuit would always result in more efficient routing.

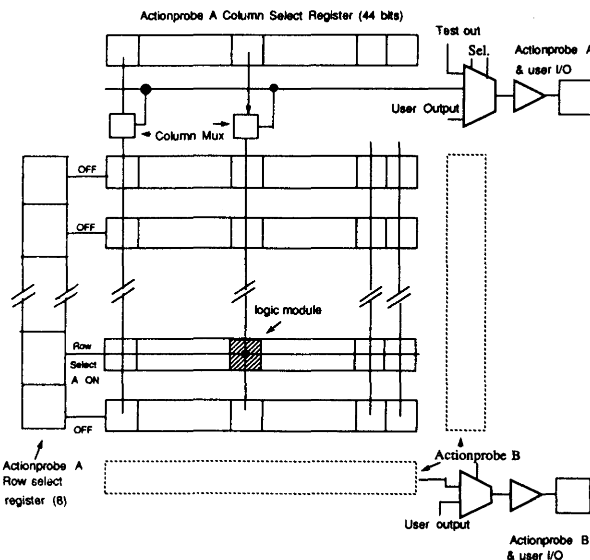


Fig. 7. In-circuit probing scheme.

D. Clock Distribution

Another important feature of this architecture is a high-speed clock distribution network which is routed to every module in the array (Fig. 6). This provides the user with a minimal skew distributed clock network that can be used to map high-speed synchronous applications in this array. It also relieves the user from the added burden of clock network design in his gate array application. Since latches and flip-flops can be built using any module in the array, building high-speed synchronous sequential networks is accomplished by using the clock network and the required latch, flip-flop, or combinatorial macrocells. The speed of the clock network is determined by the critical paths in the application. For example, the frequency of a simple toggle flop-flop has been measured at more than 70 MHz whereas in a complex application the clock network will operate at speeds up to 40 MHz.

E. In-Circuit Diagnostic Capability

One of the key features of this chip is a novel probing scheme that allows 100-percent observability of any node in the array. This feature allows designers to access and observe signals at any node in their schematic in real time, which eases and simplifies the development and debugging cycle of a user circuit.

The probing scheme which is shown in Fig. 7 consists of two independent probing channels to allow simultaneous real-time observance of two independent nodes in the circuit. This is internally implemented by two separate sets of column and row select registers, multiplexor channels, and multiplexed I/O ports. The probe mode is used in the following manner. The chip is first switched into the required probe mode which selects one or both of the two probing channels. The desired node address, consisting of row and column select fields, is then serially shifted into the chip and loaded into the required column and row select registers. This column and row select address is used to uniquely select the desired node in the circuit and feed its logic value through a multiplexor and onto a designated I/O port. Any test instrument such as an oscilloscope or logic analyzer probe may then be used to observe the required node in real time, while the chip is running in the actual system environment. The diagnostic capability described is supported by a development system which provides the user with a convenient interface to his circuit and eases the verification and debug cycle.

III. Implementation

A. Technology

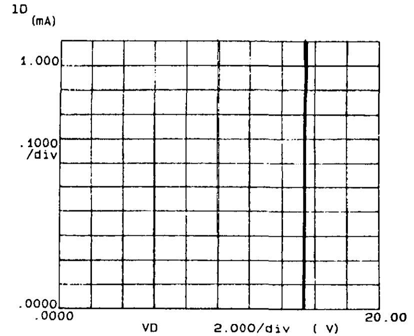
The chip is implemented using a 2-μm n-well CMOS process with two layers of metallization (Table II) and four transistor types. Two low-voltage/high-speed 250-A oxide transistors are used in all performance-sensitive circuits, such as logic modules and I/O buffers. They are also used in other low-voltage circuit applications such as address latches and control logic where their tighter design geometries are better suited. The two high-voltage 400-A oxide transistors are used in all the high-voltage circuit applications such as decoders, level shifters, programming path switching transistors, as well as isolation transistors to

TABLE II
TECHNOLOGY OVERVIEW

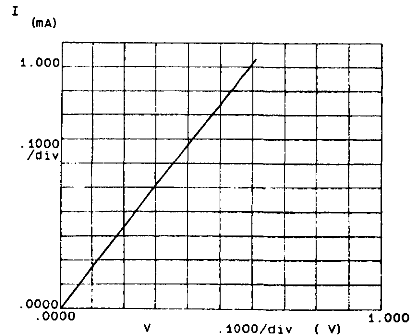
Transistor Type	Gate Oxide	Leff	Voltage
N Low Voltage	250 A	1.1u	5V
P Low Voltage	250 A	1.2u	5V
N High Voltage	400 A	1.5u	20V
P High Voltage	400 A	1.8u	20V

TABLE III
ANTIFUSE ELECTRICAL CHARACTERISTICS

Parameter	Value
Programming Voltage VPP (volts)	18
Programming Time (ms)	< 10
Programming Current (ma)	< 10
On Resistance (ohms)	< 1K
Off Resistance (ohms)	> 100M



(a)



(b)

Fig. 8. Antifuse $I-V$ characteristic (a) before programming, and (b) after programming.

isolate high-performance circuits from high-voltage circuits. The chip has 40 000 transistors evenly divided between low-voltage and high-voltage transistors.

The electrical characteristics of the antifuse element used in this technology are summarized in Table III. The two-terminal antifuse [13] is a dielectric between an n^+ diffusion and polysilicon and is compatible to CMOS, BIMOS, and bipolar technologies. The antifuse normally exhibits a very high impedance (>100 MΩ) in the OFF (unprogrammed) state. When an 18-V programming voltage is applied across its terminals, a bidirectional low-resistance (<1 kΩ) connection is established. The programming current requirement is less than 10 mA and the programming time is less than 10 ms. Since this is an antifuse technology, only a few percent of the total available antifuse elements need to be programmed and therefore, total application programming time is fairly short. Fig. 8 further illustrates the characteristics of the antifuse. Fig. 8(a) shows the $I-V$ characteristic of the antifuse as it transitions from the unprogrammed state ($I=0$) to the programmed state. In this example the antifuse programmed at 15.5 V. The maximum current through the antifuse is determined by the antifuse impedance as well as the impedance of the external circuit. Fig. 8(b) shows the $I-V$ characteristics of several programmed antifuses. The slope of the $I-V$ curves is the impedance of the programmed antifuse.

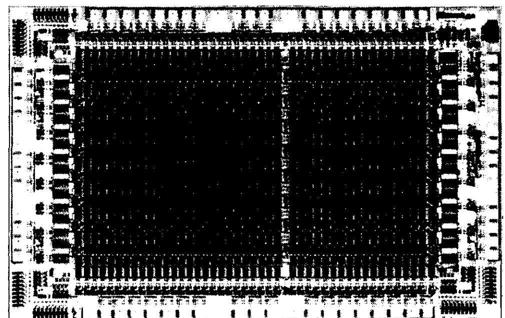


Fig. 9. Chip photomicrograph.

B. Circuit Design

A chip photomicrograph is shown in Fig. 9, and measures 240 × 360 mils in 2-μm technology. The logic module rows and the horizontal channels are illustrated. Each channel contains matrices of antifuse elements at the intersections of the horizontal and vertical track segments implemented as two layers of metallization. Surrounding the array are all the periphery circuits: address latches, decoders, high-voltage programming circuits, testability circuits, and I/O modules necessary to program and test the array.

The addressing scheme used to select and program a particular antifuse consists of a serial address stream which is shifted into the chip and latched. A distributed two-level decode scheme is then used to select the unique track(s)

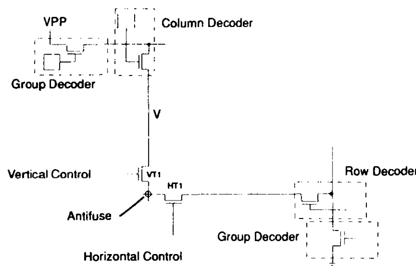


Fig. 10. Chip programming path.

that must be addressed to program the required antifuse. This is best illustrated by the programming path example shown in Fig. 10. To program the indicated antifuse element, the programming voltage must be transmitted and applied across its two terminals. This is accomplished by a two-level decode scheme consisting of a group decoder followed by a column decoder, which transmits the programming voltage onto vertical track *V*. The voltage is then propagated along the track by switching transistor *VT*1 to apply the high voltage on one terminal of the antifuse. Similarly, the ground potential is switched by another group decoder and row decoder and through switching transistor *HT*1 to the other terminal of the antifuse. The number of switching transistors that must be switched in the vertical and horizontal tracks varies with the location of the antifuse. All decoders and switching transistors are driven from the address that was serially shifted into the chip.

A circuit schematic of the decoder cell is shown in Fig. 11. Each decoder cell consists of a flip-flop, a low-to-high voltage level shifter, high-voltage decoder transistor, and miscellaneous control gates. The flip-flop is part of the address latch used in programming and testing the chip. The output of the flip-flop *OUT* is fed to the *IN*1 input of the adjacent decoder cell, thus forming the serial address shift register. The control gates are used to disable the decoder, precharge the decoder output, or enable the flip-flop output to drive the decoder. The gated signal then passes through a low-to-high voltage level shifter which drives the gate of the decoder high-voltage output transistor. The *IN*2 input of the flip-flop is used in test modes to sense and latch the result of each test. The decoder cell and some variations of it are used to implement all group, column, and row decoders in the chip.

A summary of chip characteristics is shown in Table IV. The chip has 295 configurable logic modules, 55 configurable I/O modules, and over 112K anitfuse element sites. This illustrates the routing flexibility of this chip and technology. The module delay of 5 ns is for a typical net configuration implementing a two- to four-variable function.

Another circuit technique used in the design is the appropriate use of high-voltage and low-voltage transistors. All high-performance circuits such as logic and I/O module use low-voltage high-performance transistors.

TABLE IV
CHIP CHARACTERISTICS
Delays measured at 5 V, 25°C, fan-out = 3

Parameter	Value
Chip Size (mils)	240 x 360
Package type (PLCC)	84
Standby current (ma)	< 5
Number of Modules	295
Number of programmable elements	112,000
Clock network skew (ns)	3
Module performance (ns)	5
Latch setup time (ns)	6
Latch hold time (ns)	2
Number of user defined pins	55
Configurable I/O buffers	TTL, Bidirectional
Input delay (ns)	6
Output delay (ns)	12
Output drive (ma)	4

However, since the terminals of the modules are exposed to high voltage during programming, high-voltage isolation transistors are used to isolate the high-performance part of the circuit from the programming path (Fig. 5). During programming the isolation transistors are turned OFF. During normal operating mode, the gates of isolation transistors are driven to a high voltage to lower the source/drain resistance of the transistors and minimize the delay penalty through the transistor.

IV. Testability

Testing a one-time programmable gate array presents some challenging problems [14]. Fortunately, testing is facilitated by the normally open antifuse elements. All circuit connections are normally open—net connections, module configuration, I/O module connections—and will be established as the user-application circuit is programmed into the chip. All active circuits such as modules, I/O buffers, latches, and decoders must be tested and their functionality and performance guaranteed. In addition, all passive circuit elements such as wiring channels must be free of defects such as opens and shorts.

Consequently, one of the critical design objectives was to ensure that the architecture and the design are fully testable [14]. This is critical in assuring high-quality chips and high programming yield. Testability circuits and techniques were implemented to permit complete testing of:

- every logic and I/O module,
- all vertical and horizontal tracks,
- addressing and decoding circuits,
- all programming and high-voltage circuits,
- integrity of all antifuses in the array, and
- ac performance of the chip.

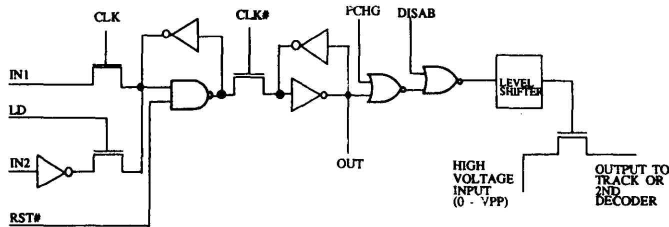


Fig. 11. Decoder cell used in the two-level decode scheme.

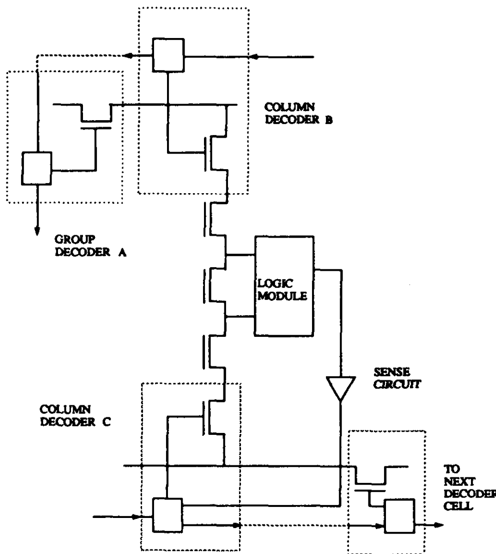


Fig. 12. Typical drive, sense and latch circuit path.

The testability technique consists of three phases: 1) driving phase, 2) sense and latching phase, and 3) shift-out and compare phase. Fig. 12 illustrates a typical drive, sense and latch circuit path used in the various test modes. In the driving phase, the chip is first driven into the required testability mode by loading a control byte into the chip. A unique test pattern is then serially loaded into the address latches which drive the two-level decode mechanism explained above. The driving circuit in this test example consists of group decoder A and column decoder B. This technique permits driving any track with any desired test voltage to suit the test pattern. Further, any number of tracks may be simultaneously driven including, if needed, all tracks in the array. Also shown in this example is a logic module under test whose inputs are driven by the vertical tracks excited by decoders A and B above. In the sense and latching phase, sensing circuits implemented in the array and in the periphery are used to sense the required signals and latch them into the same periphery address latches (data are latched on a different phase of the clock to prevent contention with the test pattern). In the example shown in Fig. 12 sensing and latching is accomplished by column decode C and group decode D. Also, each module has a sense circuit to sense its output voltage during test. Finally, the shift-out and compare phase consists of simply shifting the test result out on one of the I/O pins and comparing it with the expected test results. By using the above technique and the appropriate decoders for driving and sensing the entire chip can be tested. For example, the fuse integrity test is done by simply driving vertical tracks and sensing horizontal tracks. AC performance of the chip is tested by programming a typical path in the chip consisting of redundant logic modules and testing its speed.

The overhead of the testability circuits is minimized by sharing the periphery circuits and switching transistors inside the array between programming, testability, and diagnostic modes. Appropriate test patterns are applied that thoroughly exercise the chip before configuration. All aspects of the chip are tested, including programming of redundant antifuses and integrity of unprogrammed antifuses (antifuse programming is verified by the programming hardware as the application is programmed). This assures high-quality chips and good programming yield.

V. Applications

Two application examples are included to illustrate the flexibility and performance of the chip. The first example (Fig. 13) is a standard 4×4 array multiplier. The input vectors are $X(0-3)$, $Y(0-3)$ and the product term is $P(0-7)$. The basic macros used to implement the multiplier are the full-adder macros $FA1A$ and $FA2A$. Both macros implement the standard full-adder logic functions. The second macro $FA2A$ implements a full adder with inputs $A0$ and $A1$ going through a NOR function before the sum and carry functions are generated. This special macro is useful in multiplier functions. Each macro uses one logic module to implement the desired adder function and is configured by antifuses as explained in Section II-C.

The second example (Fig. 14) is a seven-decade frequency counter. The application simply measures the total number of clocks on the "Frequency_in" pin being measured, that occur during a particular reference time interval. The reference time interval is generated by the oscillator and the six-decade counter. The eight-decade counter is used as the frequency counter and the control circuits

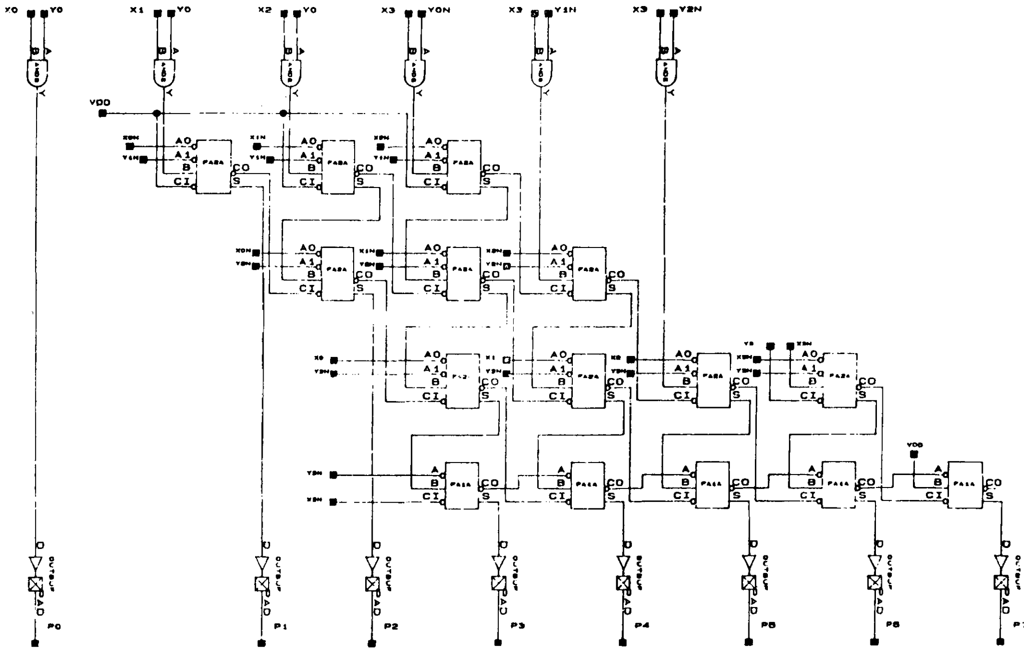


Fig. 13. 4×4 multiplier.

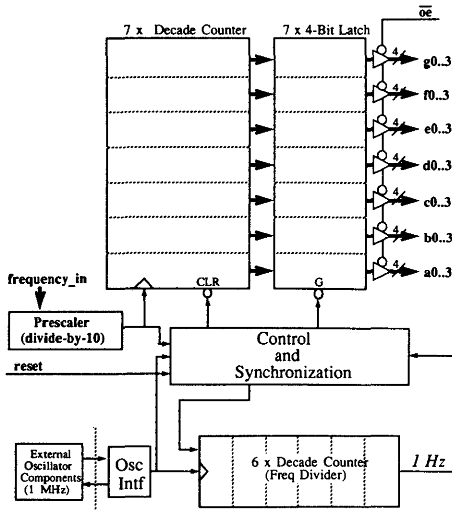


Fig. 14. Seven-decade frequency counter with prescaler.

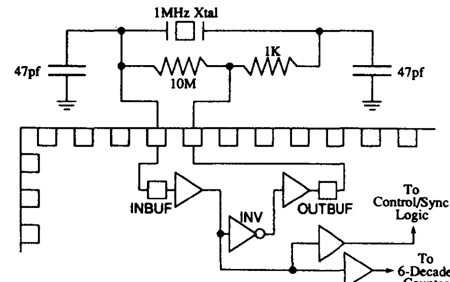


Fig. 15. Crystal oscillator using on-chip inverter.

generate the required timing and load signals. The frequency count result is then loaded into a set of output latches. The oscillator circuit is shown in Fig. 15. It uses an input buffer, an inverting logic module, and an output buffer as well as external crystal and resistor components.

Fig. 16(a) illustrates the front-end prescaler used to scale the Frequency_in pin being measured. The prescalar is simply a divide-by-10 counter implemented as a divide-by-2 toggle flip-flop followed by a divide-by-5 shift counter. Fig. 16(b) shows the logic diagram of the flip-flop macro DFM used in the shift counter. Fig. 17 shows the captured waveforms at the input and output of the prescaler. The Frequency_in pin in this waveform is running at 80 MHz and the prescaler output is running at 8 MHz. A prescaler is used in this application to vary the frequency range being measured. This application uses 284 out of 295 logic modules, 37 I/O buffers, 60 flip-flops, and 32 latches.

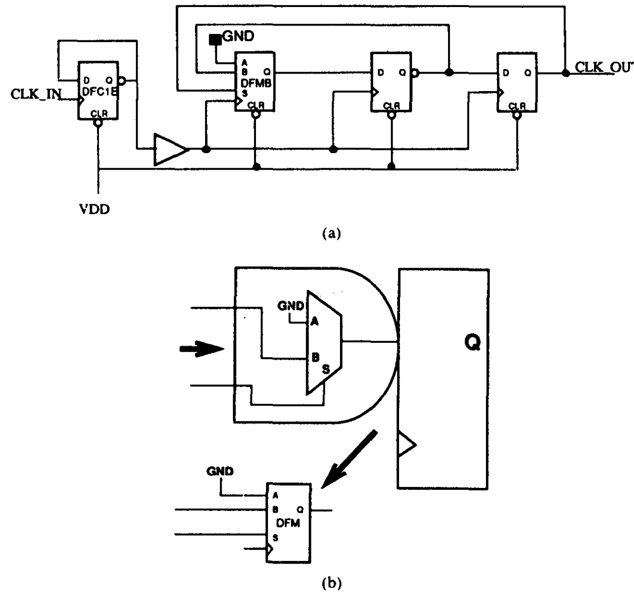


Fig. 16. (a) Divide-by-10 prescaler. (b) Configuring flip-flops with MUX inputs to perform logic functions.

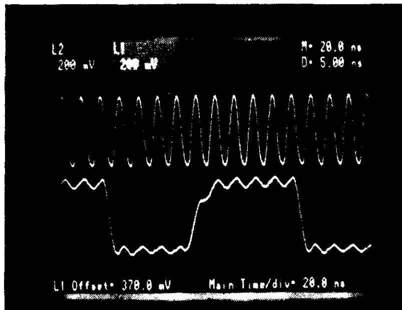


Fig. 17. Prescaler input and output waveforms.

When mapped into a conventional mask-programmed gate array this application would require 1200 gates.

The chip is supported with a computer-aided design and development system specifically developed for this chip family [12]. Designs are entered into the system as schematics or net lists using a macrocell library. The designs are then automatically placed and routed using fully automated placement and routing software capable of consistently mapping applications utilizing up to 95 percent of the array modules. Net delays are accurately computed by the system from the post-layout data. The system also contains development and diagnostic tools that greatly simplify the design process.

VI. Conclusion

A CMOS electrically configurable gate array with a channeled gate array architecture has been described. The chip is made feasible by a new interconnect technology and a novel two-terminal antifuse element. The array has 295 configurable logic modules and over 112K antifuse elements. The wiring channels consist of segmented tracks with antifuses at their intersections and between adjacent segments. The I/O modules are also configurable. The chip is implemented in a 2-μm n-well CMOS process with two layers of metallization. The implementation of the chip ensures that the design is fully testable.

Acknowledgment

The authors would like to acknowledge contributions by K. Bushey, L. Chan, J. Chen, S. Chiang, S. Eltoukhy, J. Greene, K. Hayes, H. Nguyen, B. Osann, J. Reyneri, E. Rogoyski, A. Samie, F. Sohail, and E. Takyu. They would like to acknowledge A. Chatterjee, J. Dorosti, and M. Ghafghaichi of Data General for their excellent support in manufacturing the chips.

References

[1] M. Takechi *et al.*, "A 630K transistor CMOS gate array," in *ISSCC Dig. Tech. Papers*, Feb. 1988, pp. 72–73.

[2] R. Blumberg *et al.*, "A 640K transistor sea of gates 1.2 μm HCMOS technology," in *ISSCC Dig. Tech. Papers*, Feb. 1988, pp. 74–75.

[3] J. Pathak *et al.*, "A 50MHZ CMOS programmable logic device," in *ISSCC Dig. Tech. Papers*, Feb. 1988, pp. 144–145.
[4] S. H. Bowden *et al.*, "A 9ns electrically erasable CMOS programmable logic device," in *ISSCC Dig. Tech. Papers*, Feb. 1988, pp. 146–147.
[5] S. Wong *et al.*, "CMOS erasable programmable logic with zero standby power," in *ISSCC Dig. Tech. Papers*, Feb. 1986, pp. 242–243.
[6] H. Hsieh *et al.*, "A second generation user-programmable gate array," in *Proc. CICC*, May 1987, pp. 515–521.
[7] H. Hsieh *et al.*, "A 9000-gate user-programmable gate array," in *Proc. CICC*, May 1988, pp. 15.3.1–15.3.7.
[8] K. H. Gudger *et al.*, "A 2500 gate programmable logic device with subdivisable macrocells," in *Proc. CICC*, May 1988, pp. 15.2.1–15.2.4.
[9] K. El-Ayat *et al.*, "A CMOS electrically configurable gate array," in *ISSCC Dig. Tech. Papers*, Feb. 1988, pp. 76–77.
[10] A. Mohsen *et al.*, patent pending, 1988.
[11] H. Stopper, "A wafer with electrically programmable interconnections," in *ISSCC Dig. Tech. Papers*, Feb. 1985, pp. 268–269.
[12] A. El Gamal *et al.*, "An architecture for electrically configurable gate arrays," in *Proc. CICC* May 1988, pp. 15.4.1–15.4.4.
[13] E. Hamdy *et al.*, "Dielectric based antifuse for logic and memory ICs," in *IEDM Tech. Dig.*, Dec. 1988, pp. 786–789.
[14] K. El-Ayat *et al.*, patent pending, 1989.

Khaled A. El-Ayat received the B.Sc. degree in electrical engineering from the University of Cairo, Egypt, in 1968, the M.Sc. degree in electrical engineering and computer science from the University of Toronto, Canada, in 1971, and the Ph.D. degree in electrical engineering and computer science from the University of California, Santa Barbara in 1977.

In May 1977 he joined Intel Corporation, Santa Clara, CA, to work in the Microprocessor Design Group, where he worked on the definition and development of industry standard microprocessor families such as 8086, 80186, and 80386. As a Project Manager at Intel, he was responsible for the design of the control structures of the 80386 Microprocessor. After leaving Intel, he cofounded Actel Corporation in Sunnyvale, CA, and was the Program Manager responsible for development of electrically configurable gate arrays. His present research interests include configurable logic and application-specific architectures, VLSI design, and microprocessor architectures. He has authored many articles and holds patents covering Actel's architecture and testability techniques. Presently he is a Chief Engineer working on the definition of the next generation of products.

Abbas El Gamal (S'71–M'73–SM'83) was born in Cairo, Egypt, on May 30, 1950. He received the B.S. degree in electrical engineering from Cairo University, Cairo, Egypt, in 1972 and the M.S. degree in statistics and Ph.D. degree in electrical engineering both from Stanford University, Stanford, CA, in 1977 and 1978, respectively.

From 1978 to 1980 he was an Assistant Professor of EE systems at the University of Southern California. Since 1980 he has been with the Electrical Engineering Department at Stanford University where he is currently an Associate Professor. From 1984 to 1986 he was the Director of the Systems Research Lab, LSI Logic Corporation, Milpitas, CA. He is a co-founder and Chief Scientist of Actel Corporation, Sunnyvale, CA.

Richard Guo received the B.S. degree in electrical engineering from National Taiwan University, Taiwan, Republic of China, in 1978, and the M.S. degree in electronics engineering from the University of Southern California in 1981.

In 1981 he joined Intel Corporation where he worked as a Design Engineer on 256K CMOS dynamic RAM's. In 1985 he joined Samsung where he worked as Senior Engineer in the design of a 1M CMOS DRAM. In 1986 he joined Actel Corporation, Sunnyvale, CA, as a Senior Design Engineer where he worked on the design of electrically configurable gate arrays. At present he is a Project Manager working on the next generation of products.

John Chang received the B.S. degree in electronic physics from National Chiao-Tung University, Taiwan, Republic of China, in 1977, and the M.S. degree in electronics engineering from the University of Iowa in 1981.

In 1981 he joined Advanced Micro Devices where he worked as a Design Engineer in MOS static RAM's. In 1986 he joined Actel Corporation, Sunnyvale, CA, as a Senior Design Engineer where he worked on the design of electrically configurable gate arrays.

Ricky K. H. Mak was born in Hong Kong on January 13, 1959. He received the B.S. degree in electrical engineering and computer sciences from the University of California, Berkeley, in 1983.

He joined Data General Corporation, Sunnyvale, CA, in 1983 as an IC Design Engineer. From 1986 to 1988 he was involved in the development of the configurable gate array. He is currently working on 64-bit CPU chip design.

Frederick Chiu was born in 1960 and received the B.S. degree in electrical engineering and computer science from the University of California, Berkeley, in 1983.

He joined Data General Corporation, Sunnyvale, CA, in 1984 as a Design Engineer. Currently he is working on the design of a VLSI CPU chip.

Esmat Z. Hamdy (S'76–M'80) was born in Cairo, Egypt, on February 2, 1950. He received the B.Sc. and M.Sc. degrees from Cairo University, Cairo, Egypt, in 1971 and 1975, respectively, and the M.A.Sc. and Ph.D. degrees from the University of Waterloo, Waterloo, Ont., Canada in 1977 and 1980, respectively, all in electrical engineering.

From 1971 to 1975 he served as an Instructor in the Department of Electrical Engineering, Helwan University, Helwan, Egypt. From 1975 to 1980 he was a Research and Teaching Assistant in the Department of Electrical Engineering, University of Waterloo, where he was engaged in the analysis, modeling, and design of high-density bipolar and MOS integrated structures for LSI/VLSI technologies. He joined Intel Corporation, Aloha, OR, in 1981, where he was a Senior Staff Engineer/Project Manager in the Technology Department involved in device physics and circuit design of dynamic RAM's and submicrometer CMOS. In 1985 he cofounded Actel Corporation, Sunnyvale, CA, where he is now Director of the Technology Development. He has authored or co-authored over 15 papers on LSI/VLSI circuits including contributions to I^2L, single-device-well (SDW) MOSFET's, and CMOS latch-up. He also has eight pending or granted patents.

Actel

Dr. Hamdy was the winner of the Best Student Paper at the 1979 International Electron Device Meeting (IEDM).

John McCollum received the B.A. degree in physics from the University of California at Berkeley.

He is currently employed at Actel Corporation, Sunnyvale, CA, where he is the Manager of Process Development. He previously worked at Intel Corporation where he worked on both design and process development. His interests include astronomy and electronics.

Amr Mohsen (S'72–M'74–SM'84) received the Ph.D. degree in electrical engineering and applied physics from the California Institute of Technology, Pasadena.

He is a founder and Chairman of the Board of Actel Corporation, Sunnyvale, CA, and has more than 20 years of experience in the semiconductor industry. Before founding Actel, he was a Senior Engineering Manager in the Technology Development Division at Intel Corporation. He also worked on charge-coupled device development at Bell Laboratories and served as a consultant. He has authored more than 45 articles relating to semiconductors and is responsible for inventions covered by 20 patents.

Actel

Actel Corporation 955 East Arques Avenue Sunnyvale, CA 94086 408.739.1010

Actel

AN ARCHITECTURE FOR ELECTRICALLY CONFIGURABLE GATE ARRAYS

by Abbas El Gamal, Jonathan Greene, Justin Reyneri,
Eric Rogoyski, Khaled A. El-Ayat, Amr Mohsen

Reprinted from
IEEE JOURNAL OF SOLID-STATE CIRCUITS
Vol. 24, No. 3, 1989

© 1990 Actel Corporation

Actel

An Architecture for Electrically Configurable Gate Arrays

ABBAS EL GAMAL, SENIOR MEMBER, IEEE, JONATHAN GREENE, JUSTIN REYNERI, ERIC ROGOYSKI, KHALED A. EL-AYAT, AND AMR MOHSEN, SENIOR MEMBER, IEEE

***Abstract* — An architecture for electrically configurable gate arrays using a two-terminal anti-fuse element is described. The architecture is extensible, and can provide a level of integration comparable to mask-programmable gate arrays. This is accomplished by using a conventional gate array organization with rows of logic modules separated by wiring channels. Each channel contains segmented wiring tracks. The overhead needed to program the anti-fuses is minimized by an addressing scheme that utilizes the wiring segments, pass transistors between adjacent segments, shared control lines, and serial addressing circuitry at the periphery of the array. This circuitry can also be used to test the device prior to programming and observe internal nodes after programming. By providing sufficient wiring tracks segmented into carefully chosen lengths and a logic module with a high degree of symmetry, fully automated placement and routing is facilitated.**

I. Introduction

MASK-programmable gate arrays offer the architectural flexibility and efficiency to integrate thousands of gates, but require long development time and high nonrecurring engineering costs. On the other hand, the convenience of field programming is available with programmable logic device (PLD) technologies, but their architectures have not allowed integration of a wide variety of applications exceeding a few hundred gates [1], [2].

We describe a novel gate array architecture [3] which combines the flexibility of mask-programmable arrays with the convenience of field programmability. Its implementation is made possible by a two-terminal electrically programmable anti-fuse offering low resistance in its conducting state and small area.

The architecture supports a design style similar to conventional gate arrays, including fully automatic placement and routing algorithms attaining 85–95-percent utilization. This required considerable emphasis on symmetry and routability, which we touch on below.

The anti-fuse is so called because it irreversibly changes from high to low resistance when "blown" by applying a programming voltage across it. The anti-fuse, or fuse for short, has an ON-state resistance of approximately 500 Ω. The layout area of the fuse cell is generally limited by the pitch of the first- and second-level metal lines that connect to it; it is about the same size as a via.

Manuscript received August 22, 1988; revised December 2, 1988.
The authors are with Actel Corporation, Sunnyvale, CA 94086.
IEEE Log Number 8826132.

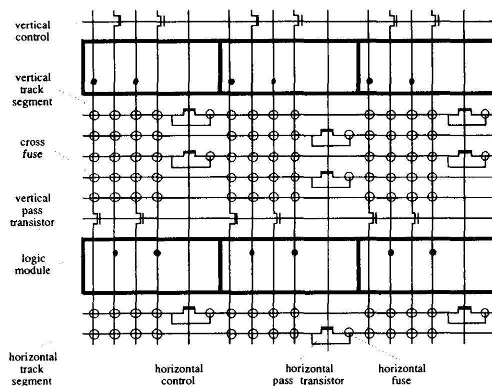


Fig. 1. Interconnect architecture.

This paper focuses on the architecture itself, which is fairly independent of the exact details of the particular CMOS technology and the anti-fuse. Other papers describe more fully the anti-fuse [4], a CMOS circuit implementing the architecture [5], and a study comparing the architecture's logic density to that of conventional gate arrays [6].

II. Programmable Interconnect Architecture

The general architecture, shown in Fig. 1, exhibits the familiar gate array organization: rows of logic cells interspersed with routing channels. There are, of course, several key differences.

The tracks in the channels are not simply empty areas in which metal lines can be arranged for a specific design. Rather, they contain predefined wiring "segments" of various lengths. Other wiring segments pass through the channels vertically. Each input and output of a logic module is connected to a dedicated vertical segment. Other vertical segments just pass through the modules, serving as feedthroughs between channels. (The number and lengths of segments in Fig. 1 are only suggestive.)

0018-9200/89/0400-0394$01.00 ©1989 IEEE

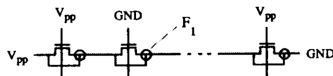


Fig. 2. Horizontal fuse programming.

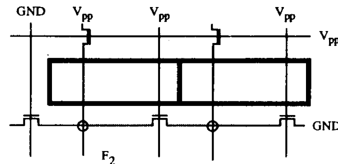


Fig. 3. Cross-fuse programming.

A fuse is located at each crossing of a horizontal and vertical segment. Programming one of these "cross fuses" provides a low-resistance bidirectional connection between the segments. Other fuses are located between adjacent horizontal segments within a track. When blown, these "horizontal fuses" connect the two segments to form a longer one. (Although not shown in the diagram, fuses may also be provided to connect adjacent vertical segments.)

In order to program a fuse, we need to apply high voltage across it. This is accomplished by an efficient addressing scheme that uses the wiring segments themselves, pass transistors connecting adjacent segments, and control logic at the periphery of the array. Fuse addresses are shifted into the chip serially.

As shown in Fig. 1, each column of "horizontal pass transistors" connecting horizontal tracks is controlled by a shared "horizontal control" line running across the array. Each row of "vertical pass transistors" is controlled by a "vertical control" line. The peripheral circuitry can drive the control lines and the segments at the end of each track.

Horizontal fuse programming is quite simple. In the example of Fig. 2, we apply programming voltage V_{PP} across the fuse F_1. All horizontal control lines except the one in the column containing F_1 are turned on by connecting them to V_{PP}, and the appropriate track segments are driven to *GND* and V_{PP} as shown. (Vertical fuses, if present, are programmed similarly.) Cross-fuse programming uses both horizontal and vertical control lines as shown in Fig. 3. Segments not driven to either *GND* or V_{PP} are left precharged to $V_{PP}/2$. Thus the voltage across fuses not being programmed is either zero or $V_{PP}/2$.

Some care is required to assure that a unique fuse is addressed. Fig. 4 shows how previously blown fuses can divert current along a "sneak path," in this case programming fuse F_5 through previously blown fuses F_3 and F_4 instead of programming F_2. Fortunately, we are not interested in blowing an arbitrary pattern of fuses (this is not a PROM!). For example, we are not concerned with programming a pattern that connects two outputs together since this does not form a useful net. If we consider only relevant patterns, it can be shown that programming the fuses in a carefully chosen order eliminates sneak paths. In general, fuses must be programmed starting from the center of the chip and moving outward, channel by channel. Determining the proper order is a bin sort operation, and can be done by software in linear time.

The pass transistors and peripheral control logic are also used to test the chip; this is discussed in detail later.

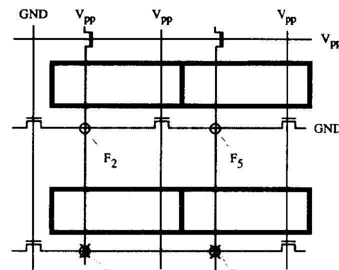


Fig. 4. A sneak path.

TABLE I

macro	4 transistor cells	modules
3 input NOR	2	1
4:1 mux, non-inverting	6	1
D latch with clear	4	1
D flip-flop with clear/set	7	2
full adder	10	2

III. Choice of the Logic Module

As outlined so far, the programmable interconnection architecture could be used with a variety of logic modules. Which would be best? This turned out to be a very difficult question, involving subtle trade-offs among routability, the logical capability of the module as perceived by the user, and delays due to capacitive loading in the routing segments.

The complexity of the module must be balanced with the routing overhead. Mask-programmed gate arrays provide very flexible and efficient routing. They therefore use a simple four-transistor cell. On the other hand, routing is very expensive in both area and delay with present programmable logic arrays. These generally use a module capable of implementing more complex functions [2]. The architecture outlined here has a cost of routing closer to a conventional gate array, suggesting a logic module of intermediate size. Because this is about the same complexity as conventional gate array hard macros, the designer can use a library like the familiar gate array cell libraries; there is no need to map logic into a more complex module. Table I lists several typical gate array macros and the numbers of four-transistor cells and logic modules required to implement them.

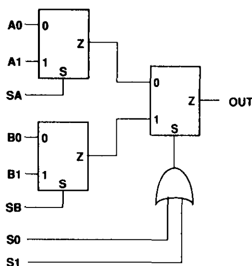


Fig. 5. Module function.

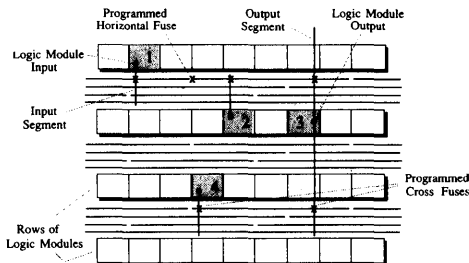


Fig. 6. A routing path.

Our chosen module, shown in Fig. 5, has eight inputs and a single output. It is composed of three two-to-one multiplexors, with an OR gate on the last stage's select input. Various macros, such as those in the table, are implemented by using an appropriate subset of the inputs and tying the remaining inputs high or low. Thus the module can implement all macros with two inputs, most with three inputs, many with four inputs, etc.

The module's output is connected to a vertical segment spanning several channels. Each input is connected to a short vertical segment spanning one channel. Four of these span the channel above the module, four the channel below. The use of short segments for the inputs reduces parasitic capacitance and hence delay.

Note that each input is accessible from either the channel above or below but not both. At first, this would appear to limit routability compared to a conventional "double-entry" gate array cell, in which signals may enter from either channel. However, there is nearly always more than one way to implement a macro. For example, there may be up to four distinct ways to implement a two-input gate: with both signals connecting to inputs in the top channel, with both signals connecting to inputs in the bottom channel, with one signal in the top and the other in the bottom channel, or vice-versa.

By letting the router choose an implementation that uses inputs accessed from convenient channels, the benefits of full double-entry symmetry are approached or sometimes attained. The degree of symmetry possible for a particular macro m implemented in a given module is reflected in the following measure S:

$$S(m) = \log_2(N(m))$$

where $N(m)$ is the number of distinct possible implementations of the macro m. Full double-entry symmetry would correspond to a value $S(m)$ equal to the fan-in of the macro. To evaluate the overall symmetry of a module, we average $S(m)$ over the macro library, weighted by relative macro usage $U(m)$ and the fan-in $F(m)$:

$$\frac{\sum_m U(m)S(m)}{\sum_m U(m)F(m)}.$$

This is the effective fraction of macro inputs in a typical design that have double-entry symmetry, and is an important criterion for choosing a module.

IV. Routing

Fig. 6 illustrates the routing of a net. The vertical segment connected to the driving module's output is connected by cross fuses to horizontal segments, which in turn connect to the segments associated with module inputs. In the top channel, a horizontal fuse is used to link two segments into a longer one.

The resistance of the blown fuses and the parasitic capacitance of the segments used form an RC tree, with the driver of the net as the root. Note that each input is driven through a maximum of three and generally two fuses to limit the delay. (If the number of series stages in the RC tree were allowed to increase further, the delay through the routing would increase rapidly.) The maximum number of fuses and the segment lengths (hence capacitances) can be altered to suit the chip dimensions and the resistance of the fuse technology.

In rare instances, it is not possible to place the macros so that all inputs on the net lie within the channels spanned by the output segment of the driving module. To handle this case, a few additional uncommitted long vertical segments are provided. The net is then routed from the output segment to a horizontal segment, then to the long vertical segment, then to another horizontal segment, and finally to the necessary input segments. To keep the number of fuses in series limited to four, no horizontal fuses are allowed in such nets. (If necessary, the architecture can be extended to provide a special fuse connecting the output directly to a long vertical segment passing over the driving module, thus eliminating the first horizontal segment and reducing the total number of fuses in series back to three.)

A means must also be provided to connect internal signals to the bonding pads of the chip. Each pad has a dedicated bidirectional buffer, which connects to the array through an associated I/O module. The I/O modules fit in the outer columns and rows of the array next to the logic modules. Each I/O module has two inputs, data and enable, and an output. The data and enable signals are

sent to the output buffer of the associated bonding pad, and the module's output comes from the input buffer of the pad. Thus the I/O module can be configured to provide input, output, tristate, or bidirectional capability.

To minimize clock skew due to differential routing delay, one entire track (or more if needed) in each channel is set aside for clock distribution. These tracks are connected directly to buffers, so that each input presents a similar load driven through exactly one fuse.

An interesting theoretical question is whether more horizontal tracks are needed in each channel here (where the lengths of the wiring segments must be predetermined) than in mask-programmed routing (where the wiring is customized for the design). Surprisingly, a high probability of routability is obtained with only a few tracks above channel density.

This requires a careful choice of the lengths of the segments, based on statistics from an extensive suite of design examples. This was done by first determining the distribution of net lengths, i.e., the length each net would run along each channel if the constraint of fixed segmentation were absent (as in a conventional gate array). The distribution of physical segment lengths provided on the chip was chosen to obey similar statistics. Then the segmentation was "tuned" manually based on actual routings which obeyed the constraints it imposed.

To obtain good routing performance it is also necessary to take advantage of the symmetry of the macros where possible. For example, observe that if macro 4 in Fig. 6 permits its input to be routed from either the upper or lower channel, there is a better chance of finding a free horizontal segment to connect it.

V. TESTING

To assure high programming yield, it is necessary to thoroughly test the chip for defects in the modules and fuses prior to programming. With a simple addition, the addressing circuitry used for programming suffices for this purpose as well.

Continuity of the tracks is easily verified by turning on all vertical and horizontal pass transistors, and using the peripheral circuits to drive the tracks from one end and read them back from the other. Testing for the absence of shorts between adjacent tracks is done in a similar way by applying a pattern of alternating ZERO's and ONE's.

Shorted or weak cross fuses are detected by turning on all horizontal and vertical pass-transistor lines, grounding all horizontal segments, and driving all vertical segments to some stress voltage. Horizontal fuses are tested column by column, with the same addressing method that is used to program them.

To verify the functional operation of the modules, we need to apply test vectors to their inputs and read their outputs. A vector is applied simultaneously to an entire row of modules by turning on all vertical pass transistors except those in the row being tested. Data are applied to the inputs in the channel above the row from the periphery at the top of the array, and to the inputs in the channel below the row from the bottom of the array.

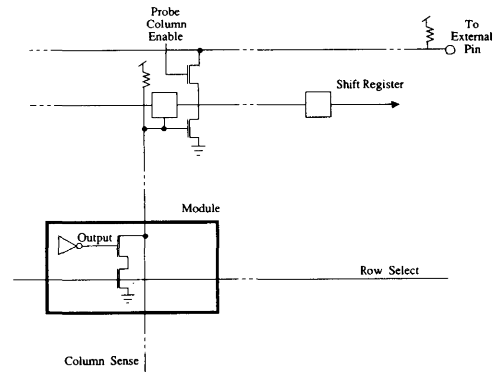


Fig. 7. Probe circuit.

Since the outputs of the modules share a vertical track with outputs of other modules above and below them, some other means is required to read the module outputs at the array periphery. As shown in Fig. 7, a select line is provided along each row of modules, and a sense line along each column. Activating the select line for the row of modules under test gates their output values onto the sense lines. The sense lines are loaded into a shift register at the top of the array.

This ability to read the output of any module at the array periphery is highly useful *after* programming as well. Only a small amount of extra circuitry is required to select one of the sense lines and make its value available at an external pin of the chip. Thus by shifting in the appropriate address, the user can observe any internal node of his design externally in real time. This virtual probe can be used and its address changed even as the programmed chip is operating in the user's system.

VI. IMPLEMENTATION: SILICON AND SOFTWARE

The architecture has been implemented in a CMOS device. For details, including the speed of the module in isolation and in an application, see [5].

Computer-aided design tools have been developed to support the architecture. Designs are entered as schematics or net lists using a cell library.

The placement and routing algorithms are specific to the architecture. As usual these are time consuming, taking up to a few hours on a low-cost workstation. They achieve 100-percent routing completion. (Even expert users have never been able to improve manually on the automatic router.) The probability of successful routing can be predicted by analyzing some statistics of the design.

Because the nets are *RC* trees, delays are not a simple function of capacitive load as with mask-programmed gate

arrays. Nevertheless, we are able to quickly calculate precise delays at each input for post-layout simulation and timing verification.

Acknowledgment

The authors gratefully acknowledge the technical contributions of J. Chang, D. Gluss, R. Guo, D. How, and F. Sohail.

References

[1] S. Wong, H. So, C. Hung, and J. Ou, "CMOS erasable programmable logic with zero standby power," in *ISSCC Dig. Tech. Papers*, Feb. 1986, pp. 242–243.

[2] H. Hsieh *et al.*, "A second generation user programmable gate array," in *Proc. Custom Integrated Circuits Conf.*, May 1987, pp. 515–521.

[3] A. El Gamal, K. El-Ayat, and A. Mohsen. "Programmable interconnect architecture," pending U.S. patent.

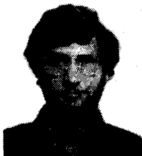
[4] E. Hamdy *et al.*, "Dielectric based antifuse for logic and memory ICs," in *IEDM Tech. Dig.* (San Francisco, CA), 1988, pp. 786–789.

[5] K. El-Ayat *et al.*, "A CMOS electrically configurable gate array," in *ISSCC Dig. Tech. Papers*, Feb. 1988, pp. 76–77.

[6] B. Osann and A. El Gamal, "Compare ASIC capacities with gate array benchmarks," *Electron. Des.*, vol. 36, no. 23, pp. 93–98, Oct. 13, 1988.

Abbas El Gamal (S'71–M'73–SM'83) received the B.Sc. degree in electrical engineering from Cairo University, Egypt, in 1972, and the M.Sc. degree in statistics and the Ph.D. degree in electrical engineering both from Stanford University, Stanford, CA, in 1977 and 1978, respectively.

From 1978 to 1980 he was an Assistant Professor of electrical engineering at the University of Southern California, Los Angeles. Since 1980 he has been with the Electrical Engineering Department of Stanford University where he is currently an Associate Professor. From 1984 to 1986 he was Director of the Systems Research Laboratory, LSI Logic Corporation, Milpitas, CA. He is a co-founder and Chief Scientist of Actel Corporation, Sunnyvale, CA.



Jonathan Greene received the Sc.B. degree in biology from Brown University, Providence, RI, and the Ph.D. degree in electrical engineering from Stanford University, Stanford, CA, in 1983, where he performed research on configurable VLSI arrays, VLSI complexity, and information theory.

During 1984 he was with Hewlett-Packard Laboratories. From 1984 to 1986 he worked on cell design automation and module compilation at the LSI Logic Systems Research Laboratory in Palo Alto, CA. He is currently Manager of System Architecture at Actel Corporation, Sunnyvale, CA.

Justin Reyneri received the B.S. and M.S.E.E. mathematics degrees from Stanford University, Stanford, CA, in 1978, and the Ph.D. degree in electrical engineering, also from Stanford, in 1985, having done research in cryptology and information theory.

He is now Manager of System Development at Actel Corporation, Sunnyvale, CA, where he has worked since 1986. Prior to that he worked at LSI Logic's System Research Laboratory on automated data-path layout, and at Hellman Associates on communications security systems.

Eric Rogoyski received the B.S. degree in mathematics from the State University of New York at Stony Brook in 1972.

He was at IBM Corporation from 1974 to 1982 with his last position in EDS at East Fishkill, NY. He held various positions in physical design from 1982 to 1984 at the company he co-founded, California Automated Design Inc., and from 1984 to 1986 at Mentor Graphics Corporation. He is currently a software consultant at Actel Corporation, Sunnyvale, CA, supporting the architectural development and physical design of Actel's configurable technology.

Khaled A. El-Ayat received the B.Sc. degree in electrical engineering from the University of Cairo, Egypt, in 1968, the M.Sc. degree in electrical engineering and computer science from the University of Toronto, Canada, in 1971, and the Ph.D. degree in electrical engineering and computer science from the University of California, Santa Barbara, in 1977.

In May 1977 he joined Intel Corporation, Santa Clara, CA, to work in the Microprocessor Design Group, where he worked on the definition and development of industry standard microprocessor families such as 8086, 80186, and 80386. As a Project Manager at Intel, he was responsible for the design of the control structures of the 80386 Microprocessor. After leaving Intel, he cofounded Actel Corporation in Sunnyvale, CA, and was the Program Manager responsible for development of electrically configurable gate arrays. His present research interests include configurable logic and application-specific architectures, VLSI design, and microprocessor architectures. He has authored many articles and holds patents covering Actel's architecture and testability techniques. Presently he is a Chief Engineer working on the definition of the next generation of products.

Amr Mohsen (S'72–M'74–SM'84) received the Ph.D. degree in electrical engineering and applied physics from the California Institute of Technology, Pasadena.

He is the founder, President, and Chief Executive Officer of Actel Corporation, Sunnyvale, CA, and has more than 20 years of experience in the semiconductor industry. Before founding Actel, he was a Senior Engineering Manager in the technology division at Intel Corporation. He also worked on charge-coupled device development at Bell Laboratories and served as a consultant. He has authored more than 45 articles relating to semiconductors and is responsible for inventions covered by 20 patents.

Article Reprint 4

DIELECTRIC BASED ANTIFUSE FOR LOGIC AND MEMORY ICS

by Esmat Hamdy, John McCollum, Shih-ou Chen,
Steve Chiang, Shafy Eltoukhy, Jim Chang,
Ted Speers, Amr Mohsen

4

A paper presented at the
ternational Electron Devices Meeting, 1988

© 1990 Actel Corporation

DIELECTRIC BASED ANTIFUSE FOR LOGIC AND MEMORY ICs.

Esmat Hamdy, John McCollum, Shih-ou Chen, Steve Chiang,
Shafy Eltoukhy, Jim Chang, Ted Speers, Amr Mohsen

Actel Corporation, 955 East Arques Ave., Sunnyvale CA 94086

ABSTRACT

This paper describes a *P*rogrammable *L*ow *I*mpedance *C*ircuit *E*lement (PLICE), which is a dielectric based antifuse for use in both logic and memory ICs. The antifuse element offers significant size and performance improvement compared to other programmable cells. A simple thermal model has been developed to predict the antifuse resistance. Each antifuse occupies an area of 1.5 um^2 using 1.2um technology. It can be programmed within 1 msec, and has a tight resistance distribution centered around 500 ohms. The reliability of both the programmed and unprogrammed states is demonstrated to be better than 40 years. The antifuse was used in the design of the first family of desktop-configurable channeled gate arrays and a 64K PROM device.

INTRODUCTION

PROMs and programmable logic devices commonly employ programmable elements such as fusible links, EPROM or EEPROM cells. These programmable elements are either large in area, require high programming current, or are three terminal devices. This paper describes a Programmable Low Impedance Circuit Element (PLICE), which is a dielectric based antifuse that offers significant size and performance improvements over other programmable elements. The antifuse structure, technology, characteristics, thermal model and reliability are described below.

ANTIFUSE STRUCTURE AND TECHNOLOGY

The PLICE antifuse is a dielectric between an n+ diffusion and poly-Si as shown in the SEM cross-section (Fig. (1)) and is compatible with CMOS and other technologies such as bipolar and BIMOS. The PLICE element was integrated in a standard 12 mask, double layer metal twin tub CMOS technology. Three additional masks were required; n+ antifuse diffusion, antifuse poly, and 40 nm oxide mask for the high voltage transistors, bringing the total to 15 masks. Four transistor types (TABLE 1) are used; the low voltage high speed transistors are used for the logic and signal path while the high voltage transistors are used for the programming path. Fig. (2) is a photomicrograph of a 2000 gate programmable gate array utilizing 186,000 antifuses. There are roughly one hundred antifuses for each logic gate to achieve a high degree of interconnectivity and gate utilization. Each antifuse occupies an area equivalent to a contact or via. The antifuses are incorporated into a cell structure such that when the antifuse is programmed, it connects a metal 1 and metal 2 line [1,2]. The cell structure size is thus limited by the metal 1 and metal 2 pitch. Fig. (3) is a photomicrograph of a 64K PROM designed using the same technology with a typical access time of 35 nsec.

ANTIFUSE CHARACTERISTICS

When 18 volts is applied across the antifuse through X-Y select transistors [1,2], the antifuse is programmed to the conductive state in about 200 usec as shown in Fig. (4). Fig. (5) shows a tight resistance distribution of antifuses programmed with a current of 5 mA.

ANTIFUSE THERMAL MODEL

Once the dielectric is ruptured, the resistance R_l of the conductive state is determined by the size of the conductive conduit (link). The size of the link is determined by the amount of power dissipated in the link which melts the dielectric. Since the temperature of the molten core varies inversely with its radius, the molten core will expand until its temperature drops below the melting point of the dielectric. Since the size of the link is much smaller than the thickness of the conductive silicon layers on both sides of the dielectric, the temperature gradient and the resultant heat conduction can be modelled by a simple sphere. The resultant equation for the link temperature T_l:

$$T_l = I_p V_l / 4\pi k_{th} r_l \quad \ldots\ldots\ldots\ldots\ldots (1)$$

Where I_p is the programming current, V_l is the voltage across the link, r_l is the radius of the link, and k_{th} is the thermal conductivity of silicon. The calculation however becomes

CH2528-8/88/0000-0786 $1.00 © 1988 IEEE

complicated if the thermal conductivity of silicon is taken into consideration as a function of its temperature [3]. Furthermore, the power dissipation is distributed throughout the sphere. Hence a computer simulation that breaks the sphere into 1 nm thick conductive shells was used to calculate the link temperature T_l and resistance R_l vs programming current I_p. The resultant calculations are plotted in Fig. (6). The link radius r_l is determined by the equilibrium of power dissipation and melting temperature, as expressed in the following equation [4]:

$$r_l = 2 I_p (\sqrt{\rho_l + \rho_{si}/2}) / 87\pi \quad \ldots\ldots\ldots (2)$$

where ρ_l, ρ_{si} are the resistivities of the link and silicon respectively during programming and the relation between link resistance R_l and link radius r_l can be obtained as follows[4]:

$$R_l = (\rho_l + \rho_{si}) / \pi r_l \quad \ldots\ldots\ldots\ldots\ldots (3)$$

From eqn.(2) and eqn.(3) we derive that R_l is thus inversely proportional to I_p (Fig. (6)), given by the following equation:

$$R_l = (87/2 I_p) * (\rho_l + \rho_{si}) / \sqrt{\rho_l + \rho_{si}/2} \quad \ldots (4)$$

This can be further simplified to:

$$R_l = 2.5 / I_p \quad \ldots\ldots\ldots\ldots\ldots\ldots (5)$$

At 5 mA, for example, the link radius is about 20 nm and the fuse resistance is 500 ohms - (approximately the impedance of a 20um wide EPROM cell) in its conductive state. The programmed antifuse resistance is a function of the reading current as shown in Fig. (7). The resistance rises at a current just before the programming current, due to heating of the conductive channel; drops when the silicon in the conductive channel melts; and continues to decrease beyond the original resistance as the current exceeds the programming current. This is due to the permanent widening of the conductive link.

ANTIFUSE RELIABILITY

Programmed antifuse reliability was evaluated using discrete antifuses incorporated in a Kelvin structure (Fig. (8)). The structures were stressed with a 5 mA current between two of the terminals 1, 2 at a temperature of 250C. Fig. (9) is a plot of the monitored voltage across the programmed antifuse as a function of stress time. Failure is indicated by an increase in voltage as the structure becomes open. Measurement of the antifuse through the other two terminals 3, 4 indicated that the programmed antifuse did not fail or exhibit any measurable resistance change. Investigation of the failure using SEM indicated that the failure is due to poly contact electromigration. With 0.9 eV activation energy [5] for contacts, the predicted lifetime is more than 40 years. Fig. (10) is a plot of time to breakdown vs 1/electric-field for unprogrammed discrete antifuses. Extrapolation of Fig. (10) indicates that the lifetime of an unprogrammed antifuse under continuous 5.5v stress exceeds 100 years, with a failure rate less than 1 FIT. In addition to the accelerated discrete antifuse device reliability data, 364 product units have accumulated a total of 571,000 device hours of dynamic burn-in at 5.5 volt and 125C (with some units reaching 2500 hours) with no failures or change in a.c. characteristics. This is equivalent to a failure rate of less than 100 FIT.

SUMMARY

In summary we have presented and modelled a reliable high *performance dielectric* based antifuse which can be programmed with relatively low current and is compatible with CMOS and other technologies. The cell read current, 2 mA at 1 volt, is an order of magnitude higher than that of an EPROM of comparable size. There is no data (charge) retention concern. The unique combination of small size and low resistance has enabled the development of the first family of desktop-configurable channeled gate arrays and a 64K PROM device.

ACKNOWLEDGEMENTS

The authors would like to express their appreciation to Prof. Chenming Hu of the University of California, Berkeley for valuable discussions, to F. Issaq, and M. Rafinejad for collecting the data, and to the technology staff of Data General for their excellent support in manufacturing the chips.

REFERENCES

1) K. El-Ayat, et al, "A CMOS ELECTRICALLY CONFIGURABLE GATE ARRAY", ISSCC Digest of Technical Papers, p.76-77; Feb. 1988.

2) A. El Gamal, et al, "AN ARCHITECTURE FOR ELECTRICALLY CONFIGURABLE GATE ARRAYS", *Proceedings of the CICC, 15.4.1; May 1988.*

3) THERMOPHYSICAL PROPERTIES OF MATTER, vol.1, Thermal Conductivity-- Metallic Elements and Alloys, ed. by Y. Touloukian, et al (IFT/Plenum, New York, 1970), p.326.

4) to be published

5) ACCELERATED TESTING HANDBOOK, Technology Associates ,D. S. Pech and O. D. Trapp , 1987, p. 5.36-37

TABLE 1
TECHNOLOGY OVERVIEW

PROCESS:TWIN TUB DOUBLE LAYER METAL CMOS 15 MASKS

TRANSISTOR	OXIDE	LEFF	VOLT
N LOW VOLT	25 nm	1.1 um	5 V
P LOW VOLT	25 nm	1.2 um	5 V
N HIGH VOLT	40 nm	1.5 um	20 V
P HIGH VOLT	40 nm	1.8 um	20 V

ANTIFUSE CHRACTERISTICS:

PROGRAMMING VOLTAGE	18 V
PROGRAMMING TIME	< 1 mS
PROGRAMMING CURRENT	< 10 mA
ON RESISTANCE	< 1K OHMS
OFF RESISTANCE	> 100M OHMS

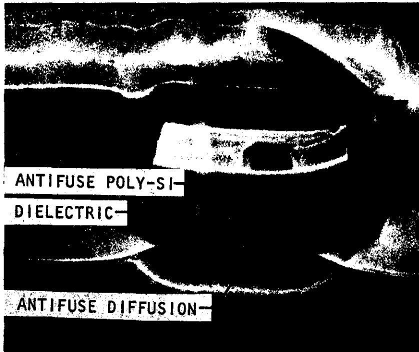


Fig.(1) SEM cross-section of antifuse.

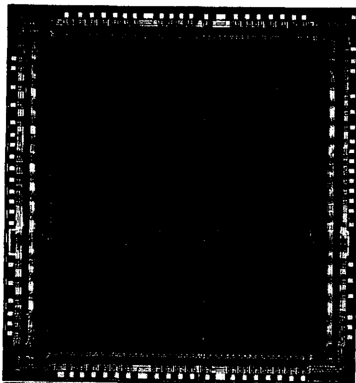


Fig.(2) Photomicrograph of 2000 gate programmable gate array.

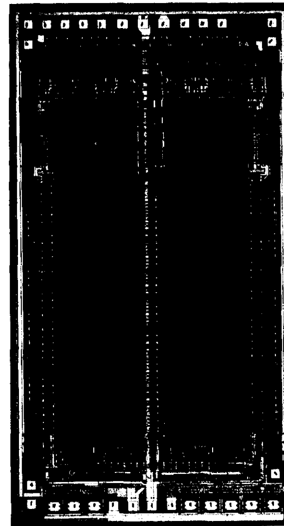


Fig.(3) Photomicrograph of 64K PROM

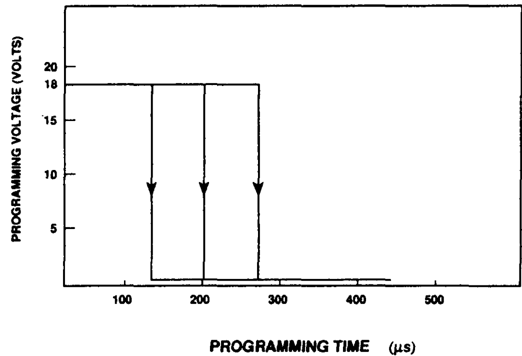


Fig.(4) Programming time of antifuse with 18V programming voltage.

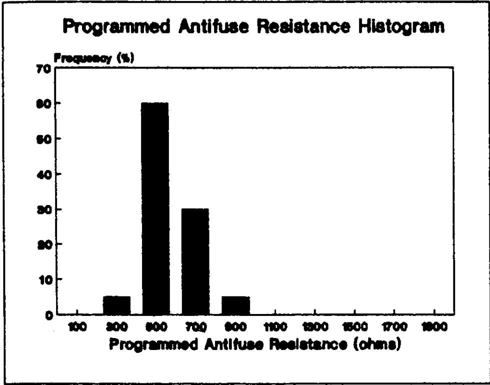


Fig.(5) Programmed antifuse resistance histogram, programmed with 5 mA current.

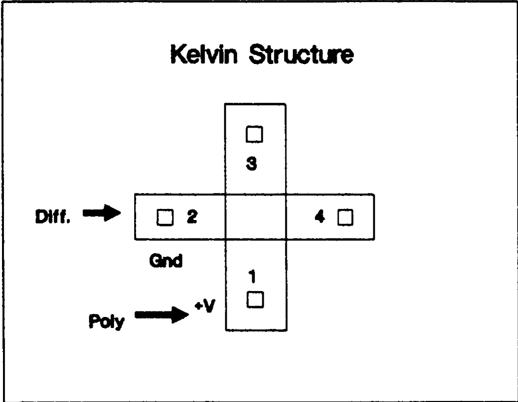


Fig.(8) Antifuse Kelvin structure.

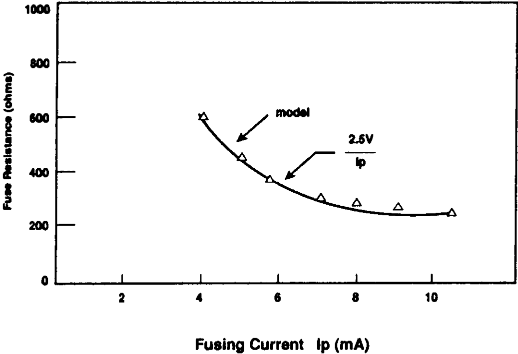


Fig.(6) Programmed antifuse resistance versus programming current.

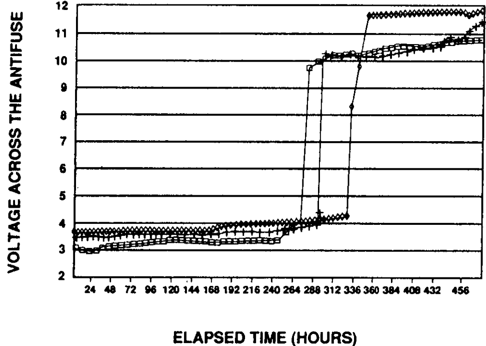


Fig.(9) Voltage across antifuse versus stress time, with 5 mA stress current.

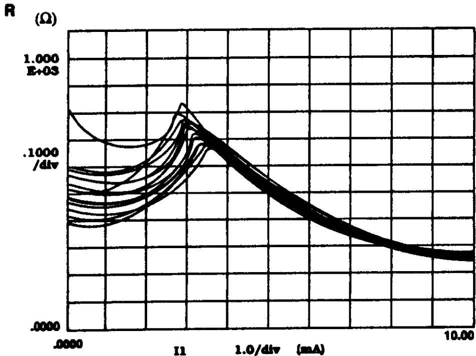


Fig.(7) Programmed antifuse resistance versus reading current, with 100 ohms of parasitic resistance in series.

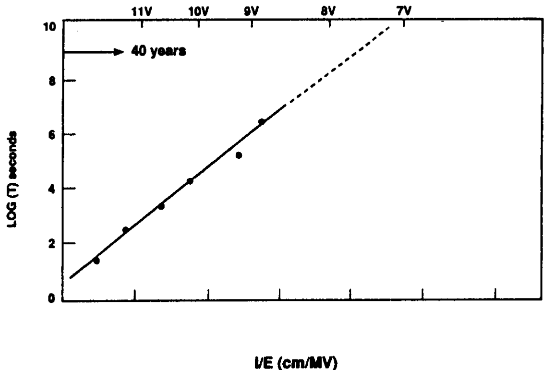


Fig.(10) Time to breakdown versus inverse electric field.

Article Reprint 5

FIELD-PROGRAMMABLE GATE ARRAY WITH NON-VOLATILE CONFIGURATION

by Andrew Haines

Reprinted from
MICROPROCESSORS AND MICROSYSTEMS
Volume 13, No. 5, June 1989

4

© 1990 Actel Corporation

Field-programmable gate array with non-volatile configuration

Field-programmable gate arrays (FPGAs) allow fast turnaround; mask-programmable gate arrays offer flexibility. **Andrew Haines** describes the new FPGA family from Actel which boasts speed and flexibility

Actel Corporation has developed a family of field-programmable, channelled gate arrays based on proprietary architecture and interconnect technologies. The new semicustom devices combine the flexibility of mask-programmable, channelled gate arrays and the convenience of fast-turnaround, electrically programmable logic devices. A new programming element, the PLICE antifuse, which can be produced in a standard CMOS fabrication facility, permits significant architectural innovation.

microsystems gate arrays logic modules PLICE antifuse

In July 1988, Actel introduced the ACT 1 family of field-programmable gate arrays (FPGAs). Ranging from 1200 to 6000 gates, these devices have a number of important attributes that clearly distinguish them from previous approaches: low cost; fast prototype development; masked array density; and applications flexibility. Using a familiar schematic capture and macro library design entry procedure, comprehensive CAE support includes automatic placement and routing at 85–95% gate utilization.

The underpinning of the ACT 1 family is a new PLICE (programmable low-impedance circuit element) antifuse programming element. The antifuses, non-volatile, two-terminal elements that exhibit low 'on' resistance when programmed, provide the same wire-to-wire interconnect functions as 'vias' provide in mask-programmable gate arrays, and that transistor-based EPROM and RAM cells and metal fuses provide in previous programmable logic devices.

The very small size (1.25 μm per side) and low programmed-state resistance of the PLICE antifuse facilitate the creation of a new field-programmable, channelled gate-array architecture, with gate density, application flexibility and performance comparable with masked arrays, but with short development cycles.

The paper first reviews the key elements of FPGA architectures and then, in this context, examines the ACT 1 architecture. Next the PLICE programming element which underlies the architecture is considered and finally the characteristics of ACT 1 family devices are described from the point of view of the user.

Actel Corporation, 955 E. Arques Avenue, Sunnyvale, CA 94086, USA
Paper received: 27 January 1989. Revised: 10 March 1989

ARCHITECTURAL CONSIDERATIONS IN FIELD-PROGRAMMABLE GATE ARRAYS

Consider the key elements of field-programmable gate array architecture — the logic modules, the number of potential interconnect elements, and the delay introduced when interconnecting logic modules. Gate arrays consist of an array of cells or logic modules separated by routing channels[1]. The logic capacity of a device is determined by the number and size of the logic modules, the effectiveness of the modules in implementing diverse logic functions (such as flip-flops, latches and gates), and the percentage of modules that can be wired together in a given application. This percentage utilization is dependent on the amount of wiring or routing resources that are available on the chip. Performance is determined by the intrinsic delay of the logic modules, the delay introduced by interconnect wiring and the number of programming elements required in a programming path.

A major consideration in the choice of a logic module is the module size which can range from a few gates to tens of gates. The ideal logic module is very small, i.e. made up of a few gates which can be combined in many ways to provide suitable logic functions. Choice of a small logic module is constrained, however, because the smaller the module the greater the wiring resources required on-chip with which to tie them together. Adequate routing resources may be impossible to achieve if the size of the programmable interconnect element consumes too much silicon area. While the choice of a large logic module relieves routing resource requirements, larger module sizes may be less efficient in logic function implementation if a module containing tens of gates must be sacrificed to perform simple AND, OR or INVERT functions. Logic applications vary with respect to the relative amount of combinatorial and sequential logic that they contain[2]. Regardless of its size, the logic module

0141-9331/89/05305-08 $03.00 © 1989 Butterworth & Co. (Publishers) Ltd

must be designed such that it can implement a wide variety of logic function with high efficiency.

While the size of the logic module contributes to the routing resource requirements, equally important are the routing resource requirements intrinsic to the user's application. Functions with a regular structure and serial data transmission, such as shift registers and counters, have relatively low fan-out per gate and modest routing requirements. Most random logic applications, such as multipliers, accumulators and ALUs, have higher effective fan-out and demand more routing resources for the same number of gates. Bus-oriented architectures can exhibit very high fan-out and are notorious for presenting difficult routing problems.

Providing sufficient routing resources on chip to meet a wide range of application requirements is a crucial architectural problem. If resources are inadequate, users will find that effective device capacity varies greatly depending on the application. If this is so, the supplier's designated capacity of '2000 gates' may actually materialize only infrequently, if ever. Placement and routing, a key part of the design cycle, is also adversely affected. Under these conditions, automatic software may perform poorly, forcing the user into frustrating and time-consuming interactive placement and routing.

For field-programmable devices of all types, performance in specific applications is dominated by the delay associated with interconnecting logic modules. Field-programmable devices must have a programming element in the interconnect pathways. Since these elements do not have zero resistance, and capacitance is present in the circuits, they introduce delays. Hence, with respect to device performance, programming elements can be thought of as delay elements. The lower the resistance of the programming element, the faster the device will be. Therefore, another critical architectural consideration is the minimization of the number of delay elements in any given interconnect path. A large or indeterminate number of programming elements in signal paths leads to intolerably slow or unpredictable performance.

In reality, the selection of a logic module, routing resources, and device performance are closely coupled. The trade-offs between choices for these elements find a common focus in the choice of a programming element. Once a programming element has been selected, many possible architectures are immediately ruled out. For example, if a large, medium resistance element (approximately 1k ohm) such as a static RAM cell is chosen, then the choice of a small logic module is precluded because the large number of interconnect elements that would be called for cannot fit on any reasonably sized chip. The desired programming element must exhibit the smallest possible size, low on-state resistance and compatibility with conventional CMOS double-layer metal manufacturing processes.

ACT 1 FAMILY ARCHITECTURE

Actel has developed a new architecture that takes advantage of the small size and low resistance of the PLICE antifuse elements for electrically configuring devices. The architecture can be extended to high gate counts through the use of sub-micron geometries, providing a level of integration comparable with mask-programmable, channelled gate arrays. In fact, the device architecture exhibits the familiar, channelled gate array organization — rows of logic cells interspersed with routing channels (see Figures 1 and 2). The architecture differs from that of a traditional gate array in the details of the horizontal and vertical wiring tracks and, of course, the antifuse programming elements.

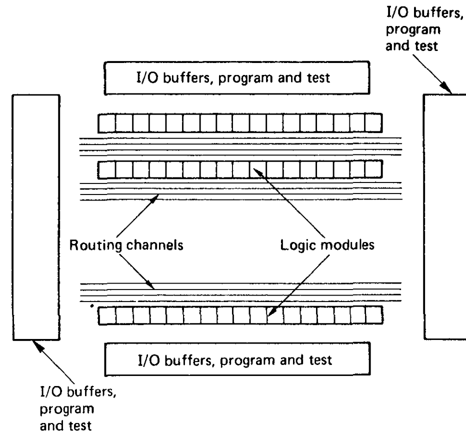


Figure 1. Block diagram of ACT 1 array showing arrangement of logic modules, wiring channels and peripheral circuits. Testability circuits and address circuitry for programming are built into the periphery

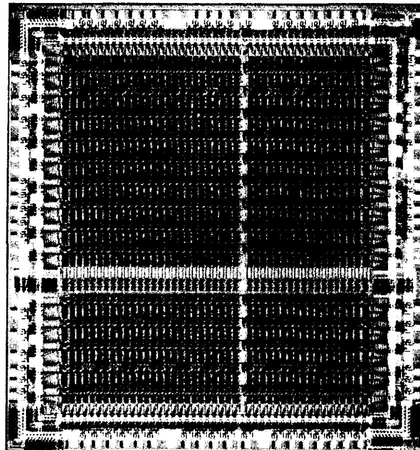


Figure 2. Die photograph of ACT1020, 2000-gate array. Channelled gate array architecture with rows of logic modules (narrow rows) separated by routing channels (wide rows) is prominently displayed

Figure 3 is a section of Actel's customer-configurable, channelled gate array showing the tracks, antifuse elements and interconnect architecture. The tracks within the channels contain predefined horizontal wiring segments of various lengths. Vertical tracks span several rows of logic modules and adjacent horizontal channels. An

4

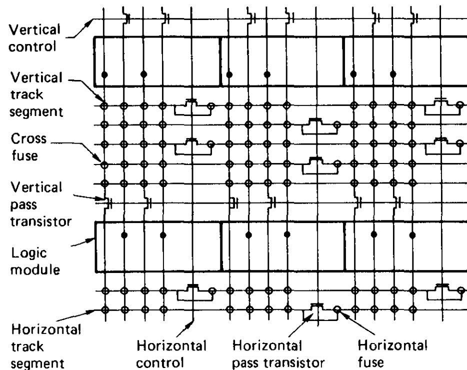


Figure 3. Simplified organizational view of ACT 1 family architecture showing logic modules, routing tracks and programming elements

antifuse programming element is located at each crossing of a horizontal and vertical segment. Programming one of these 'cross antifuses' provides a low resistance bidirectional interconnect between the segments. Other antifuses are located between adjacent horizontal segments within a track. When programmed, these 'horizontal antifuses' connect the two segments to form a longer one. Similarly, antifuse elements are also located between adjacent vertical segments.

Vertical and horizontal pass transistors are also shown in Figure 3. These pass transistors are used only during antifuse programming and, so, do not play a role in device performance. Inclusion of these control transistors allows the routing tracks themselves to carry programming voltages during the programming sequence, thus saving additional wiring. The control transistors, in turn, are controlled from a shift-register-based addressing scheme in the chip periphery. During programming, antifuse 'addresses' are shifted into the chip thus setting the pass transistors to the appropriate state to deliver the 18 V programming voltage to the selected antifuse.

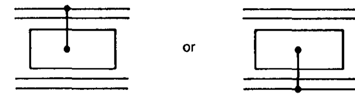
Interconnection

An antifuse element is programmed by the application of a programming voltage (V_{pp}). An efficient addressing scheme, utilizing wiring segments, pass transistors connecting adjacent segments, and control logic at the periphery of the array, provides selective application of programming voltage to antifuse elements. Antifuse addresses are shifted into the chip serially. Further information on ACT 1 family devices and architecture can be found in References 3–5.

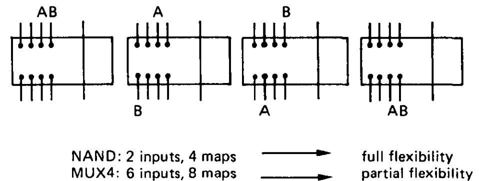
Table 1. Effectiveness of the ACT 1 logic module in implementing common logic functions

Function	Module efficiency (% of chip required)	
	ACT1010 1200 gates	LCA3020 2000 gates (from Xilinx)
1 of 8 decoder (74138)	4.1	7.8
8 : 1 MUX (74151)	1.7	6.3
3-bit binary counter	3.4	3.1
8-bit serial to parallel SR	5.4	7.8
4-bit Johnson controller w CLK EN	2.7	3.1

Mask programmed gate array cells are 'double entry'



Same effect achieved with configurable gate arrays, with several ways to map a macro into a module



NAND: 2 inputs, 4 maps → full flexibility
MUX4: 6 inputs, 8 maps → partial flexibility

Figure 4. Inputs to the ACT 1 logic module. Like conventional masked arrays, these can be routed from the top or bottom of the cell since most macro functions can be mapped into the logic module several ways. A function with two inputs, A and B, can have several mappings with various locations of A and B, thus giving the router more choices

Logic modules

A configurable logic module is the basic logic building block of Actel's channelled gate array devices. Depending on the size of the gate array, the family of field-programmable, channelled gate arrays contains between 295 (ACT1010) and 1258 (ACT1260) configurable logic modules. The flexible logic module of the ACT1010 has eight inputs and one output. With a small size, the module is similar in capability to a conventional gate array macro and can be configured into over 150 different functions. The design of the module was carefully chosen for its efficiency in implementing both combinatorial and sequential circuits and for its optimum utilization of routing resources (see Table 1). Because JK- and D-type flip-flops can be configured from the logic modules, no dedicated flip-flops are required on-chip.

Also, like a conventional gate array, the module is 'double entry' in that inputs can enter from the top or bottom of the cell (see Figure 4). Each of the eight inputs is connected to a short vertical segment spanning one routing channel. Four inputs span the channel above the module and four span the channel below. The module's output is connected to a vertical segment spanning several channels. The benefit of double-entry symmetry relates to routing efficiency: by letting the routing algorithm choose module access (from above or below), up to 95% gate utilization is attained.

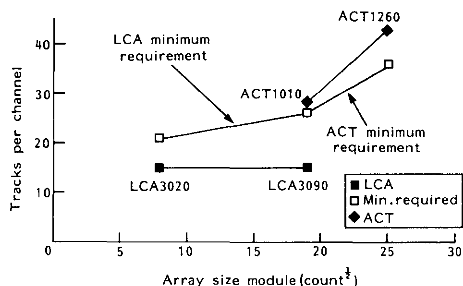


Figure 5. Routing resources (wiring tracks per channel) of field-programmable gate array architectures compared with the minimum theoretical quantity needed to achieve automatic placement and routing 90% of the time

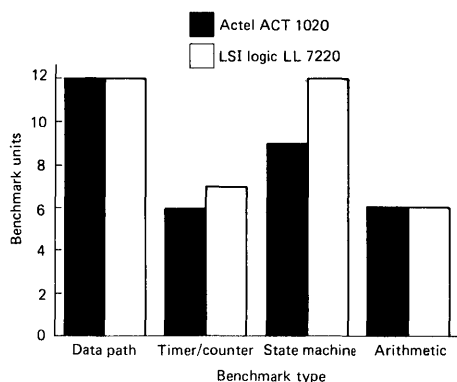


Figure 6. Benchmarking results comparing the effective gate capacity of ACT 1020, 2000-gate array with a LL7720, 2200-gate array

Routing resources

The extremely small size of the PLICE antifuse element makes a large number of interconnect elements practical, thus enabling the selection of a small, highly configurable logic module. To accommodate the resulting interconnect element requirements, 186 000 PLICE elements are included on the ACT1020 2000 gate chip. This is enough to attain conventional gate array routability in a field-programmable device. For comparison, a conventional mask-programmable array of like capacity contains roughly 90 000 potential metal-to-metal via or interconnect sites. To attain the same level of placement and routing efficiency as a masked array, a field-programmable array generally needs more interconnect resources. This results from the fact that wiring must be built in to the field-programmable device before anything is known about the specific end-user application, while with a masked array wiring is only applied by the manufacturer after a specific application has been received. Given this additional information, fewer wiring resources are needed in the masked array.

Wiring resources must be chosen so that automatic placement and routing software can achieve good results (85–95% utilization > 90% of the time), otherwise a heavy design burden is placed on the end user and the design cycle is considerably lengthened. Minimum routing requirements can be theoretically determined given the number of logic modules which must be interconnected and the average number of I/Os per module[6]. Figure 5 compares the routing resources available in ACT 1 family chips with those found in field-programmable arrays using the logic cell array (LCA) architecture and with the theoretical minimum needed to achieve 90% probability of successful completion of automatic placement and routing (APR).

Gate count

Because Actel's devices have a channelled gate array architecture with a very large interconnect capacity similar to conventional gate arrays, device gate capacity is similar. The ACT 1 family of gate arrays offers 1200, 2000, 3000 and 6000 gate sizes with up to 142 I/O pins. Unlike programmable logic devices, where 'equivalent gate' values vary dramatically depending on the application, these configurable gate arrays maintain consistent capacity and utilization across a broad variety of application types just as in conventional arrays.

In the absence of industry-standard benchmarks, Actel has developed four benchmarking standards that can be used to test achitectural capacity with respect to different application types[2]. These include a data-path circuit, a counter/timer, a state-machine function, and an arithmetic structure. Each of these benchmarks is of the order of 150–300 gates as measured in LSI logic gate array terms — an industry accepted means. Figure 6 compares the capacity (in terms of benchmark circuits) of the 2000-gate ACT1020 against the industry standard LL7220 channelled gate array. This comparison clearly indicates that the effective gate count of the field-programmable channelled gate array is comparable with conventional devices over a broad range of application types.

PERFORMANCE CHARACTERISTICS OF THE ACT 1 ARCHITECTURE

The low resistance of the PLICE antifuse (typically 500 ohm) and the ACT 1 interconnect architecture minimize the impact of programming elements on device performance. In 90% of cases the structure of the ACT 1 modules and wiring scheme limit the number of antifuses found in a propagation path to two. Routing software limits the number of antifuse elements to a maximum of four.

While specific application examples can be used to illustrate performance, a more informative view can be gained from an understanding of delays using a statistical approach. Any given path in a field-programmable array can have a range of delays depending on how it is actually routed on the chip. By taking a specific path and routing it many times, a distribution of delays can be formed. The resulting distribution illustrates the range of path delays which can be expected over a whole series of applications. The characteristics of this distribution provide a more general view of performance than that from looking at specific examples. If the distributions are very narrow, delays will predictably fall into a narrow range. Conversely,

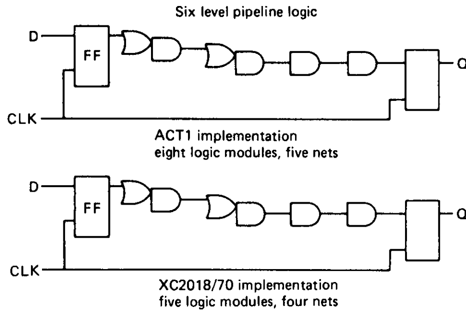


Figure 7. Segment of pipeline circuit used for performance modelling has six logic levels between flip-flop elements

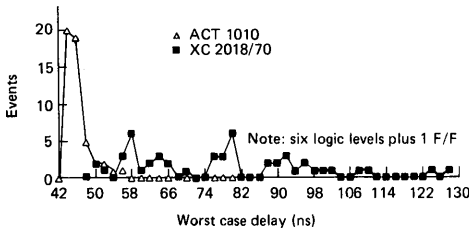
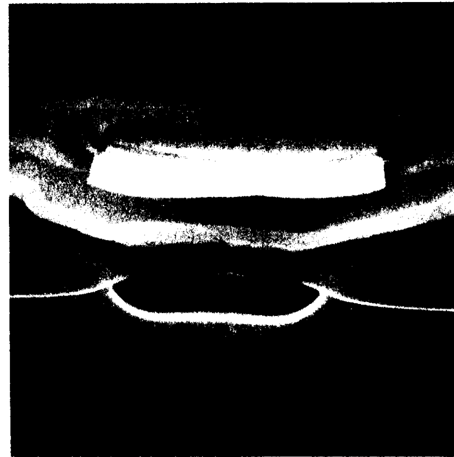


Figure 8. Results of Monte Carlo simulation showing delay distributions for 50 placement and routings of six logic-level pipeline path

the wider the distribution, the more likely it is that unexpectedly large delays will occur during application development. Such delays may not be revealed until the design process is well under way, making early preroute performance estimations of little value.

A convenient way to create such distributions is to use the Monte Carlo simulation method that is commonly used in the physical sciences. In this method, known distributions of a set of independent variables are used to create a distribution of a function of the independent variables. For FPGA performance evaluation, actual net delay statistics gathered from placement and routing a number of designs can be used to simulate delays for a complex logic path consisting of a number of nets. Figure 7 shows two paths, one implemented in a 2.0 μm ACT 1 family array and the other implemented in a 1.2 μm XC2018 LCA. Each logic function separated by a wire is implemented in a separate logic module. Such paths are common in pipelined logic structures. Differences in logic module capacity dictate the difference in the number of logic modules used to implement the two cases. In preparation for the path delay simulation, net delay statistics were gathered from actual placement and routings of nets with a fan-out of two. As in masked arrays, nets in field-programmable devices with low fan-out tend to be faster than nets with high fan-out. Figure 8 shows the delay distributions for 50 placements and routings of the two paths as generated by the Monte Carlo simulation technique.



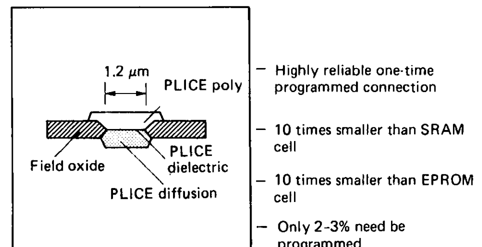


Figure 9. Scanning electron microscope cross-section of PLICE antifuse showing conductive layers of polysilicon and diffusion separated by insulating layer

PLICE antifuse

The PLICE antifuse is a programming link that has been developed for use in integrated circuit applications. Unlike a traditional fuse approach where programming creates an open circuit, programming an antifuse creates a very low resistance connection.

While many antifuse approaches have been developed to date, most need very high breakdown voltages and currents, are difficult to manufacture or do not meet the reliability requirements of state-of-the-art integrated circuits.

PLICE does not have these drawbacks and is the first antifuse that can be closely controlled during manufacture to deliver consistent and reproducible characteristics. PLICE antifuse elements consist of a dielectric sandwiched between two conductors (Figure 9). When a voltage is applied across the element, the dielectric breaks down and connects the conducting materials together. The PLICE antifuse adds only three masking steps to a conventional double-metal CMOS process and can be produced in a typical CMOS fabrication facility using conventional material and standard processing equipment and technology.

The PLICE element is a two-terminal, non-volatile, one-time programmable device that normally exhibits a very

Table 2. PLICE antifuse characteristics

Programming voltage (V_{pp})	18 V
Programming time	<10 ms
Programming current	<10 mA
On resistance	<1k ohm
Off resistance	>100M ohm

high impedance (>100M ohm) in the unprogrammed state. When an 18 V programming voltage (V_{pp}) is applied across its terminals, the bidirectional low-resistance connection is established. Programming current is <10 mA. Conveniently, in an array based on antifuse elements no more than 3% of the available antifuse elements need to be programmed. The characteristics of the PLICE antifuse are listed in Table 2. For further information on the PLICE antifuse see Reference 7.

Reliability

The reliability of the PLICE technology has been demonstrated. Because an antifuse element possesses both an 'on' state and an 'off' state, PLICE antifuse reliability depends on the reliability of each state. The reliability studies are of two types:

- the PLICE antifuse reliability by itself
- reliability of products containing PLICE antifuses.

The reliability of isolated circuit elements such as transistors, diodes or antifuses are normally studied with special test structures under operating conditions that significantly accelerate failure mechanisms. Studies of this type reveal reliability levels that would take years to demonstrate with products operating at their environmental extremes.

To confirm the reliability of the 'off' state, accelerated data were collected on the equivalent of millions of PLICE antifuse device hours. The data confirm that the time to failure for a PLICE antifuse at normal operating conditions is substantially more than 40 years and the combined contribution of all antifuses to the gate array product's hard failure rate is <1 FIT (failure in time; 0.0001%/1000 h). In the case of the reliability of the 'on' state, accelerated measurements show no intrinsic failure mechanism.

Product reliability depends not only on circuit elements such as the PLICE antifuse, but also on other features of the manufacturing and assembly process. Product reliability studies show reliability levels normally encountered with CMOS circuits.

Product reliability data from a sample of 614 devices subjected to 125 °C dynamic burn-in life testing for more than 762 000 device hours indicate no observed PLICE antifuse failures or change in AC characteristics, implying a device failure rate of 82 FIT at 70 °C.

TESTABILITY FEATURES

Testing a one-time-programmable gate array requires careful attention to on-chip testability features. All antifuse connections are normally open; no routing connections will be established until the user's application circuit is programmed into the chip. The ACT 1 architecture allows all active modules to be tested by adding appropriate testability circuits. In addition, testability circuits ensure that all passive circuit elements, such as the wiring channels, are free of defects such as opens and shorts.

Actel has implemented within its ACT gate arrays testing circuits and techniques to permit complete testing of every logic and I/O module, all vertical and horizontal tracks, addressing and decoding circuits, all programming circuits, and the integrity of all antifuses in the array. In essence, ACT 1 arrays allow 100% testability of all circuits.

The testability technique consists of a driving phase, a sense and latching phase, and a shift out and compare phase. In the driving phase, the chip is first driven into a required test mode and then loaded with a specific test pattern. This technique permits any number of tracks to be driven simultaneously with any desired voltage that suits the test pattern. In the sense and latching phase, special circuits on the periphery of the chip are used to sense the required signal and then latch them into the same circuits. During the final phase, the test results are shifted out and compared with expected test results.

Over 700 000 test patterns are applied to each 2000-gate chip by the test equipment in order to exercise and test the chip before configuration by the user. All aspects of the chips are tested including the integrity of unprogrammed antifuses. As a result of the extensive factory testing, the user is not responsible for generating test vectors to screen devices for manufacturing faults.

DESIGN AUTOMATION SYSTEM

Productive application of field-programmable gate arrays requires the availability of a sophisticated design automation system. The designer must rely on software tools for nearly every aspect of the development cycle, from design entry to performance analysis.

The design automation system, action logic system (ALS), is comprised of three major components: the logic design system; diagnostic and debug software; and the Activator hardware for programming and functional test. The capabilities of the logic design system include schematic capture with the Actel macro library, simulation, timing analysis, electrical rules checking and placement and routing.

The design automation process for the ACT 1 family is analogous to that of mask-programmed gate arrays (see Figure 10). The designer begins with a schematic capture phase that is followed by simulation. The simulated net list is then transferred to the Actel-developed portion of the system where the user makes his physical I/O

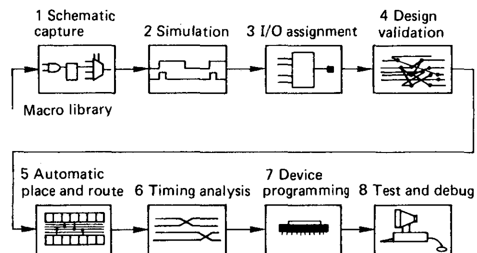


Figure 10. Design sequence for ACT 1 family field-programmable gate arrays

4

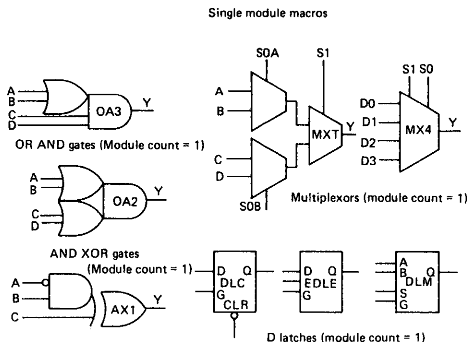


Figure 11. Selection from Actel macro library showing number of logic modules required for each function

assignment and performs an electrical rule check (validation). The next step is placement and routing, followed by timing analysis using back annotation. The remaining steps include programming with programming hardware, debug and functional test.

The initial task facing a user of the ACT 1 family is to implement his logic function using the Actel macro library. In this respect, the macro library becomes a major feature of the user interface to the Actel devices. In practice, a thorough knowledge of the macro library is more important than a detailed understanding of the underlying architectural features. The user can safely disregard details such as the number of wiring tracks per channel or the exact nature of the logic module, since they are successfully hidden from the user of ACT 1 arrays by the complete automation of the placement and routing process. This minimizes both the time required to learn the system and the length of the development cycle. ALS users claim that a finished part can be available as little as 1.5 h after completion of schematic capture[8].

Library macros are implemented by making a specific connection to the logic modules. These macros, similar in appearance and function to those of conventional gate arrays, provide the primary logic entry interface for ACT 1 devices. Each single logic module implements macros ranging in complexity from simple gates to transparent data latches and 4-to-1 multiplexers. All Actel gate arrays are supported by a macro library of over 200 standard logic functions. Also included in the library is a selection of commonly used TTL functions. Figure 11 shows some macros which can be implemented in a single logic module.

Unlike conventional gate array libraries, the Actel logic modules implement logic functions with inversions on inputs as efficiently as non-inverting inputs. More importantly, they do so with no increase in propagation delays. Actel's library includes gate macros and any combination of 'negation' bubbles on the inputs. Thus, inverters and associated wiring, generally, are no longer needed.

Perhaps the most interesting aspect of the design automation system is the placement and routing algorithm, which is specific to Actel's channelled gate array architecture. The placement and routing software which maps a logic description to the physical device resources (the logic modules, wiring segments and interconnect elements) is vital to effective use of a field-programmable gate array. Ineffective software in this step substantially lengthens the development cycle and decreases device utilization, thereby increasing the cost per function delivered.

As mentioned above, the architecture supports a design style similar to conventional mask-programmed gate arrays. The system produces fully automatic placement and routing with device utilization levels of 85–95%. In fact, Actel's algorithms frequently provide fully automatic placement and routing even at 100% module utilization for the 1200- and 2000-gate devices. Thus, designers need no longer spend additional time manually completing placement and routing for their complex designs. By eliminating this manual operation, design time can be cut by weeks, improving productivity and lowering overall system development costs.

The impressive performance of the placement and routing operation is made possible by a channelled gate array architecture with ample routing resources available through the use of PLICE antifuses as interconnect elements. This architectural approach permitted Actel to develop advanced software by building on years of algorithms development for mask-programmable channelled gate arrays.

The placement and routing algorithms operate in three phases. The first phase includes constructive placement and placement optimization similar to conventional gate arrays. The second placement optimization improves the characteristics of the placement using a simulated annealing-style algorithm prevalent today in high-end gate array routing software. The final phase, detailed routing, is unique to the Actel architecture.

Within the diagnostic software, special debug software with commands similar to those of a simulator are available to the user. Also available under the diagnostic subsystem are in-circuit diagnostic capabilities that permit the user to obtain realtime information about the state of the device while it is running in the actual product environment.

A unique feature of Actel's diagnostic and debug software is that it takes advantage of two Actionprobe diagnostic circuits within the chips. The Actionprobe circuits can connect any two points electrically in the device to external pins (Figure 12). The connections are not implemented by the one-time programmable antifuses,

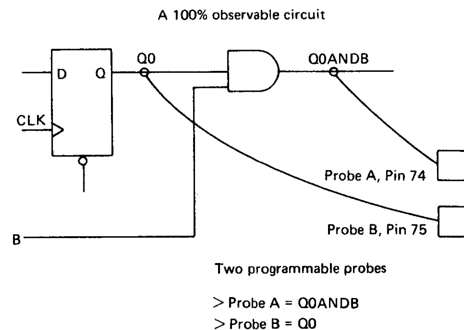


Figure 12. On-chip diagnostic probes are implemented in ACT 1 devices and are accessible through the design automation system

but instead bypass transistors in the array so that the probe points can be changed any time by commands typed at the keyboard. Probing provides 100% observability of all internal nodes so that the user actually observes electrical activity normally hidden within the gate array.

With the use of a special socket, 100% observability is available even when a device is plugged into the user's system and is operating at speed. Efficient verification and debug can take place as the system generates millions of stimuli per second for the device under test. The Actionprobe feature shortens design verification and in-system debug; it also reduces the burden of test vector generation by substituting the use of in-system stimuli.

The Activator is comprised of hardware and software for programming and functional test. The Activator facilities can be used to perform a number of specific tests in addition to programming. For example, every chip is tested by the Activator before programming to ensure that the device is actually unprogrammed. In addition, the Activator performs a verification test to ensure that antifuses have programmed correctly. Programming time, including the verification phase, is typically 5–7 min for a fully utilized 1200-gate device and 8–12 min for the 2000-gate part. Improvements in the programming algorithm and updated programming hardware will permit the 6000-gate devices to be programmed in under 10 min. Programming times are short since few of the antifuses need to be programmed to utilize a device fully. The Activator system can also be used for functional testing and capturing test vectors for simulation from known good devices.

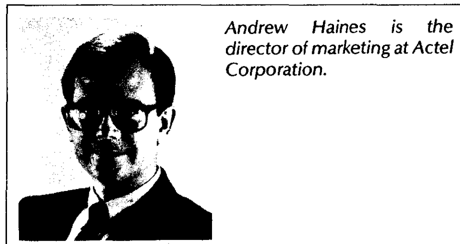
The ALS is currently available on a 80386-based PC and provides an interface to, and includes, Viewlogic's schematic capture and simulation capabilities. The recommended hardware configuration is a standard 386-based AT with 4 Mbyte of extended memory, a 40 Mbyte hard disc, one serial and one parallel port. Users will use Viewdraw for schematic capture and Viewsim for simulating their designs. Actel's logic design system, activator and diagnostic software are included with this version of the ALS. Mentor Graphics workstations are also supported as well as the OrCAD schematic capture system.

CONCLUSION

The PLICE antifuse makes possible a new field programmable device which combines the high density, speed and flexibility of mask-programmable gate arrays with the convenience of field programmability. Sophisticated design verification capabilities have been made available to users through new hardware- and software-based diagnostic features, and design productivity is increased by the development of novel placement and routing algorithms that deliver fully automatic operation.

REFERENCES

1 **York, T** 'Gate array architectures' *Microprocessors Microsyst.* Vol 12 No 6 (July/August 1988) pp 323–330
2 **Osann, R *et al.*** 'Compare ASIC capacities with gate array benchmarks' *Electronic Design* (October 13, 1988) p 93
3 **El Gamal, A *et al.*** 'An architecture for electrically configurable gate arrays' *IEEE Custom Integrated Circuits Conf.* 15.4.1 (May 1988)
4 *ACT 1 family gate arrays, product information* Actel Corporation, Sunnyvale, CA, USA (January 1989)
5 **El-Ayat, K *et al.*** 'A CMOS electrically configurable gate array' *Int. Solid State Circuits Conf. Digest of Technical Papers* (February 1988) pp 76–77
6 **IBM** *J. Design Automation and Fault Tolerant Computing* (May 1979)
7 **Hamdy, E *et al.*** 'Dielectric based antifuse for logic and memory ICs' *Proc. Int. Electron Devices Meeting* (December 1988) pp 786–789
8 **Leonard, M** 'Digital system design' *Electronic Design (January 12, 1989) p 88*
9 **Mohsen, A** 'Desktop-configurable channelled gate arrays' *VLSI Systems Design* (August 1988)



Andrew Haines is the director of marketing at Actel Corporation.

4

TESTING ANTIFUSE-BASED FPGAS

by Khaled El-Ayat, Andy Haines, Ken Hayes

4

Reprinted from
ELECTRONIC ENGINEERING TIMES
May 15, 1989

© 1990 Actel Corporation

Electronic Engineering TIMES

This week:
Programmable Logic
Technology Trends liftout, after page 42.

Monday May 15, 1989 Issue 538

A CMP Publication

THE INDUSTRY NEWSPAPER FOR ENGINEERS AND TECHNICAL MANAGEMENT

Testing antifuse-based FPGAs

By Khaled El Ayat,
Chief Engineer
and
Andy Haines,
Director of Marketing
and
Ken Hayes,
Product Engineering Manager
Actel Corp.
Sunnyvale, Calif.

With the explosion of higher density PLDs and the advent of even higher density field-programmable gate arrays (FPGAs), the issue of thorough factory testing will become evermore important. That's because inadequate device testing prior to use places a severe burden on the system manufacturer.

To try to compensate for inadequate factory testing, the system manufacturer can institute testing at incoming inspection—a difficult task for complex LSI devices. If he doesn't, he must rely on board-level testing, or worse, system-level testing. And, inevitably, some faulty devices will escape to become field failures. It is well-known the cost of finding failures rises dramatically at each of these stages.

Masked-programmed gate arrays are attractive for their low recurring costs, high gate counts and speeds. However, one little discussed disadvantage is that the quality of test coverage is the responsibility of the user. A thorough job of test-vector generation requires the use of fault grading or generation of automatic test vectors. Neither is very common in practice. Fault grading is generally expensive and adds complexity to the design process. Automatic vector generation is simplified if a scan-design technique is adopted, but few engineers are willing to conform to the required design constraints. Automatic vector generation without scanning techniques is still a developing discipline.

However, poorly tested devices are the likely result of inadequate fault coverage. It has been shown that typical masked gate arrays with 70 percent fault coverage are likely to have 0.1 to 0.2 defects per device.

Many field-programmable devices largely alleviate this problem because they can be tested at the factory. Such factory testing, characteristic of standard products, generally improves over time, resulting in high quality levels without significant involvement by the user. This has not always been the case, however. Early one-time programmable devices using fuse-based technology often disappointed users with functional defect rates of more than 10 percent for those parts that programmed successfully.

This has been improved substantially by the addition of on-chip test circuits to defect rates of 0.5 percent to 1 percent. While this represents a low defect rate for individual chips, putting 10 such chips on a single board contributes to a 5 percent to 10 percent board-failure rate. Further improvement is difficult due to the fact that in conventional one-time programmable architectures, complete functional testing of all circuits in an unprogrammed part is not possible. In such fuse-based architectures in the unprogrammed state, many circuits are shorted together by the unprogrammed fuses.

The introduction of reprogrammable logic devices based on EPROM, E²PROM and RAM programming elements has raised functional quality levels to nearly 100 percent. However, defect rates do not fall to zero because of post-testing failure mechanisms such as incorrect programming, marking, insertion and ESD damage.

Non-volatile testing

With at least one non-volatile, one-time programmable programming technology, a degree of testability comparable to reprogrammable devices is possible. Actel's ACT 1 arrays allow for special test modes that perform functional testing of unprogrammed devices at essentially 100 percent fault coverage.

This testability is independent of the large number of equivalent gates (1,200 in the ACT 1010 and 2,000 in the ACT 1020). In order to describe how this is accomplished, a few points about the ACT 1 architecture need review.

The basic building block on the chip is the logic module, which is programmable and capable of implementing all two-input logic functions, most three-input functions, and some four- and six-input functions. A single module implements a variety of latches, and two modules implement more than 32 types of D and J-K flip-flops. Each module is effectively the same complexity as a conventional gate-array macro. The ACT1020 contains 546 of these logic modules. In order to achieve 85 percent to 95 percent utilization of these numerous small building blocks, a large number of routing tracks and programming points are required, comparable to those found in a conventional array. These are arranged in horizontal rows and vertical columns.

Horizontal routing tracks alternate with the rows of logic modules. Vertical routing tracks run over the top of the logic modules and horizontal tracks. Around the outside of chip are I/O circuits, test circuits, programming circuitry and a shift-register-based antifuse addressing scheme.

The Actel-invented Plice antifuse was specifically developed for making logic interconnections. An antifuse is any programming element that is normally open and makes a connection upon the application of a programming sequence. The antifuse has a natural advantage over fuse-based programming with respect to testability. Normally open in the unprogrammed state, antifuses do not short chip circuitry together.

On the contrary, all elements of the chip can be viewed as isolated building blocks. By building in special on-chip circuitry, all architectural elements can be thoroughly tested prior to shipment and programming. Much of this circuitry is also required for programmability and normal chip operation, so that testability overhead is modest.

This multiplicity of function is especially evident for the routing tracks. Routing tracks both horizontally and vertically are broken into segments of various lengths. This makes possible the use of segments of appropriate lengths for interconnections. Pass transistors are present between these segments and can be used to form temporary connections between adjacent segments.

This allows the routing tracks to be used for carrying programming voltages, accessing chip elements for testability purposes, as well as providing their primary function as logic routing resources. The pass transistors are not used during normal device operation and do not contribute to logic-signal propagation delays. The accompanying figure shows a simplified view of logic modules, routing tracks and pass transistors.

In the test modes

The architecture of the ACT 1 arrays allows outstanding testability of unprogrammed devices. All elements of the chip can be thoroughly tested at the factory with the exception of the programmability of specific antifuses needed for a particular user-defined application. However, these can be automatically tested during the programming sequence. Thus, a complete test is performed on each device without the need for user-generated test vectors.

Testing begins with the shift-register-based addressing scheme. Through the shift register, commands can be shifted into the device, enabling special modes such as programming and functional testing. In addition, the shift register can address specific antifuses, logic module outputs, and read or drive voltage levels on specific routing tracks. The shift register can be both up-loaded and down-loaded by means of a serial shift clock and a data pin.

This setup allows various patterns to be shifted in and out again, verifying full shift-register functionality. With the shift register operational, all vertical and horizontal routing tracks can be tested for continuity and shorts. These tests also ensure that all vertical and horizontal pass transistors will turn on. Further tests on these pass transistors are performed to ensure that they have satisfactory parametric values.

The ACT 1 arrays contain a dedicated clock buffer that crosses the chip in the horizontal routing channels. This buffer can be tested by driving the chip's clock pin and reading the proper levels on the sides of the array via the shift register.

The arrays have two special pins that can be used as user I/O channels or, in the test mode, as outputs of two on-chip probes that can route the output of any logic module to the external pins. This is possible on either programmed or unprogrammed parts, and selection of probe points does not require the programming of antifuses. In fact, the probes are reprogrammable and can be steered under software control to any two points in the chip.

These probes are also available to the user through design-verification software, which is a standard part of the Actel design system. The probes allow a great many tests to be performed that would otherwise be impossible.

For example, all input buffers can be tested by driving them low on the outside of the chip and probing the internal side of

the buffer. Built-in test modes also permit driving all output buffers to high, low or high-impedance. So it is possible to test Vol, Voh, Iol, Ioh, Vil, Vih and leakage on all I/O lines.

One of the keys to thoroughness with which ACT arrays can be tested is the module test. Logic modules, in their unprogrammed state, can be viewed as eight-input/one-output combinatorial functions. By using the horizontal and vertical pass transistors, any of the eight module inputs can be forced to a high or low. The output of the module can then be read through either probe pin. In this way, every logic module on the chip can be tested with a 100 percent fault-coverage vector set. No antifuse programming is required to perform these tests. In addition, the architecture allows modules to be tested in parallel for reduced test time.

Antifuse testing

The consistently high utilization and flexibility of ACT 1 FPGAs is due in large part to the copious number of routing tracks on chip. This, in turn, requires numerous programming elements. The small size of the Plice antifuse makes sufficient numbers possible. The ACT 1020 chip, with its 2,000-gate equivalents, contains 186,000 antifuse programming elements.

Surprisingly, only 2 percent to 3 percent need be programmed to fully utilize a chip. Nonetheless, antifuse testing must play a significant part in the overall device test methodology.

The first antifuse test is the blank test. The entire array of antifuses is easily tested to ensure that antifuses are in their normally open state. The next and most important antifuse test is the accelerated life test. Dielectric breakdown in Plice antifuses is a strong function of voltage, and this effect can be used to perform an effective life-test acceleration screen in place of the more commonly used temperature-acceleration approach.

In addition, the convenience of a voltage-acceleration factor facilitates the performance of a life-test screen on every chip and its antifuses as part of the normal test sequence. The acceleration voltage is applied through the programming voltage pin (VPP). After this test, the blank test is again run to detect any antifuses that have failed.

This test is very effective at catching antifuse defects. Using devices screened in this way, more than 700,000 device-hours have been completed in 125° C dynamic burn-in reliability testing with no observed antifuse failures.

Several tests are used to confirm that programming circuitry works correctly. The first such test is a basic junction stress and leakage test. The programming mode is enabled, and the programming voltage (VPP) plus a guard band is applied to the chip. No voltage is applied to the antifuses. If programming current (IPP) exceeds a normal value, the device is rejected. The functionality of the programming circuitry can be further verified by programming various antifuses associated with special test structures and other non-user accessible features.

A number of logic modules exclusively available for test purposes are connected to each other by programming antifuses. This structure also can be used for AC-performance characterization of the chip. In addition, a number of antifuses are configured as a readable signature word. Bits in this word are programmed with factory-defined patterns. Successful completion of these tests ensures basic functionality of on-chip programming circuitry.

The description of the numerous ACT 1010 and 1020 test modes attest to the outstanding testability of these devices. As a result, statistical sampling currently shows a programming yield of better than 98 percent.

Even more important, for all sampled parts that successfully passed programming, no functional failures have been observed.

These tests were conducted with designs that utilized 92 percent to 97 percent of the logic modules. Although this utilization is high, it is not infrequent among users of Actel arrays.

955 East Arques Ave.
Sunnyvale, CA. 94086
408-739-1010

Copyright© 1989 by **CMP Publications, Inc.**, 600 Community Drive, Manhasset, NY 11030. Reprinted from Electronic Engineering Times with permission.

Article Reprint 7

COMPARE ASIC CAPACITIES WITH GATE ARRAY BENCHMARKS

by Bob Osann, Abbas El Gamal

Reprinted from
ELECTRONIC DESIGN
October 13, 1988

© 1990 Actel Corporation

DESIGN APPLICATIONS

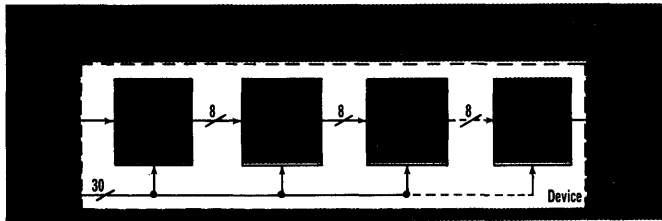
FOUR BASIC APPLICATION CIRCUITS ACCURATELY PREDICT IF AN ASIC DESIGN CAN BE IMPLEMENTED IN A SPECIFIC GATE ARRAY

COMPARE ASIC CAPACITIES WITH GATE ARRAY BENCHMARKS

When evaluating a gate array's capacity, designers typically concentrate on only the circuit they are working on. The trouble here is that this evaluation says nothing about whether the gate array will implement the designer's next circuit, or the one after that.

A functional method of testing gate array capacities must consider the characteristics of the different circuits being implemented. ASIC designers must know if a circuit can fit into a certain gate array. The solution: benchmarking, which takes routability and application differences into consideration. Toward this end, Actel has developed a set of application circuit benchmarks that give design engineers accurate predictions of ASIC capacities.

ASIC capacities are currently measured in terms of equivalent gates. An equivalent gate is a two-input NAND gate made up of an array of unconnected transistors. For example, manufacturers of CMOS gate arrays calculate the amount of equivalent gates simply by counting the number of transistors and dividing by four—the typical number of CMOS transistors per gate. This approach to calculating gate-array capacities is fairly reliable and consistent within a limited class of devices, but it doesn't relate to a meaningful standard that

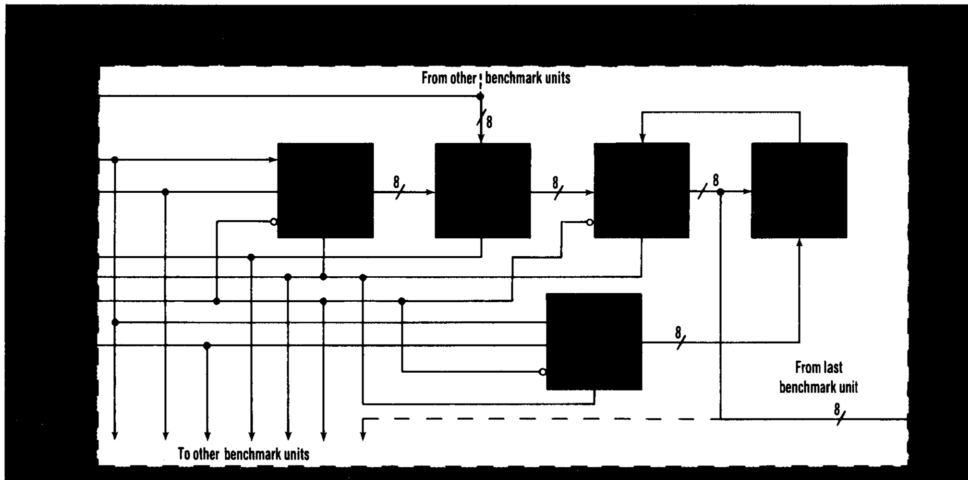


1. A MEASURE FOR a benchmark circuit unit is taken by mapping the unit into a gate array as often as possible in a step-and-repeat procedure.

BOB OSANN AND ABBAS EL GAMAL
Actel Corp., 955 East Arques Ave., Sunnyvale, CA 94086; (408) 739-1010.

DESIGN APPLICATIONS

ASIC BENCHMARKS



2. 248 GATES ARE needed to implement the timer-counter benchmark circuit. This timer is typical of microprocessor-based systems.

applies across all ASIC types.

A better, more accurate measure of ASIC capacity is a benchmark that considers application differences. Considering the broad variation of applications, creating these benchmarks isn't a trivial task. For example, some applications contain a great deal of combinatorial logic while others are mainly sequential. These inconsistencies mean that certain application circuits map into different ASIC gate-array types better than others.

The key to making efficient use of an ASIC's available gates is routability. Because all circuits aren't equally routable, numerous benchmark circuits are needed. Another factor in routability is the ratio of combinatorial to sequential logic—an important consideration for an ASIC that comes with a fixed number of flip-flop resources.

PIN-PER-GATE COUNTS

One way to roughly evaluate a circuit's routability is to consider how many macro pins-per-gate each macro function in the circuit employs. A "macro pin" refers to the input and output connection points on a macro-logic function—it has no relation to an I/O pin on the device package. Sequential logic tends to use many gates and relatively few pins per gate, while combinatorial logic uses a relatively high number of pins per gate approaching the maximum of three (two-input NAND).

In general, the greater number of pins per gate, the more difficult the circuit is to route. However, the pin-per-gate routability measure is limited: An average pin-per-gate figure doesn't convey the local congestion that can arise in certain types of applications, putting excessive strain on an ASIC's routing resources.

A useful evaluation is to actually route the various benchmark circuits into gate arrays. This approach is especially effective when applied to ASIC architectures with routing resources less abundant than those of channeled gate arrays. Specifically, the benchmark circuits must offer divergent pin-per-gate ratings to give a valid picture of routability for different applications.

Because most real-world applications include a variety of circuit types with different routability levels, filling a chip with one type of circuit might seem artificial. This is why there are four different benchmark circuits each mapped into one chip. But even one application benchmark circuit alone gives a realistic measurement as most application circuits tend to be bimodal—their sequential logic and combinatorial logic generally have very different pin-per-gate measurements. The combinatorial logic sections are commonly routing-limited, while the sequential logic sections are not. The different benchmark circuits measure the ASIC's ability to deal with each type of logic.

An ideal benchmark circuit unit should be small enough to fit into an ASIC many times without leaving a sizable amount of unused logic. On the other hand, the units must be large enough to realistically create the routability challenges of actual application circuits. Also, the circuits should be designed such that no I/O limitations are encountered, allowing the benchmark focus to remain solely on logic capacity.

Any useful benchmark circuit must relate to some type of standard. The standard against which Actel measures its units is an LSI Logic gate array. Among the most widely used gate arrays in the industry, LSI Logic parts have become de facto industry standards. With circuits constructed from LSI Logic macros, capacity requirements in

4

Reprinted with permission from ELECTRONIC DESIGN - October 13, 1988

Copyright 1988 VNU Business Publications, Inc.

terms of two-input NAND equivalents can be quickly computed in a clear way.

To take application variability into account, benchmark circuits must be designed to represent all the different types of application circuits. The CALC (comparative ASIC logic capacity) benchmark developed by Actel is actually a set of four circuits, including a data path, timer-counter, state machine, and an arithmetic unit. The CALC benchmarks are an attempt at a broader, more disciplined determination of an ASIC's capacity than an equivalent gate reading.

Step-And-Repeat

Each benchmark circuit is used to create a measure by mapping it into an ASIC as many times as possible in a step-and-repeat procedure *(Fig. 1)*. The results demonstrate how well an ASIC accommodates the type of circuit represented by the benchmark. The number of successful iterations supply a relative measure of device capacity when the same benchmark circuit and procedure is applied to several devices.

The first CALC benchmark is the data-path circuit unit. Most board-level designs—even those that heavily utilize programmable logic devices—use a significant number of data path devices. The purpose of the data-path circuit benchmark is to steer signals from one point to another, and to occasionally store the signals. Common TTL datapath functions include octal registers and latches, buffers, transceivers, and multiplexers. Shift registers used to perform parallel-to-serial and serial-to-parallel conversions are also considered a form of data-path circuit function.

This data-path benchmark circuit employs a 4-to-1 multiplexer, an 8-bit register, and an 8-bit shift register. The octal 4-to-1 multiplexer—similar to an LS153 multiplexer—selects 8 bits from a 32-bit input bus. This function approximates the many dynamic-RAM-based systems in which a 4-to-1 multiplexer selects between the processor address and the refresh address. Eight of the 32 data

BENCHMARK CIRCUITS IMPLEMENTED

bits entering the multiplexer are kept separate in order to allow data to be received from a preceding benchmark unit when placing multiple, daisy-chained circuit units into a device.

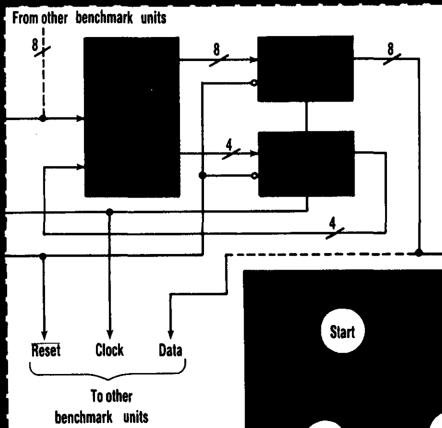
The outputs of the multiplexer feed a continuously-clocked, 8-bit holding register. The register then feeds an 8-bit parallel-in/serial-out shift register, similar to the popular LS166 register used in many video designs.

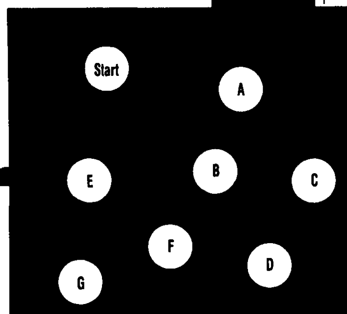
The benchmark shift register also has a parallel 8-bit output used for the daisy-chain connection to the next benchmark unit. The shift register's operation is controlled by its Shift/Load signal, which connects to the Clock Enable inputs of the register's flip-flops.

The pin-per-gate distribution indicates many sequential macros with 0.5 pins per gate, and on the other end of the scale, 4:1 multiplexers with the maximum of three pins per gate. When implemented in an LSI Logic LL7220 gate array, each data-path benchmark circuit needs about 157 gates.

The second benchmark circuit is the timer-counter *(Fig. 2)*. This particular circuit is typical of the many timer functions found in microprocessor-based systems. While dedicated timer-counter chips such as the 8253 are popular, they are too general for many ASIC applications and would require too many gates. A better solution is to build a specific timer-counter function.

The multiplexer used in this circuit is similar to an LS157 multiplexer and allows the LS161-type counter to

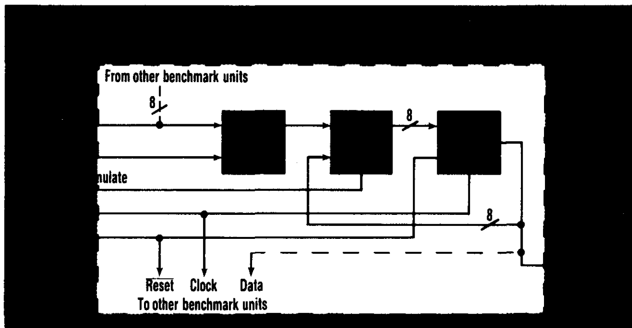




3. THIS THIRD circuit, an average-sized controller/sequencer, contains a large amount of non-standard logic. The outputs of this state machine are determined by its state diagram (see the insert).

DESIGN APPLICATIONS

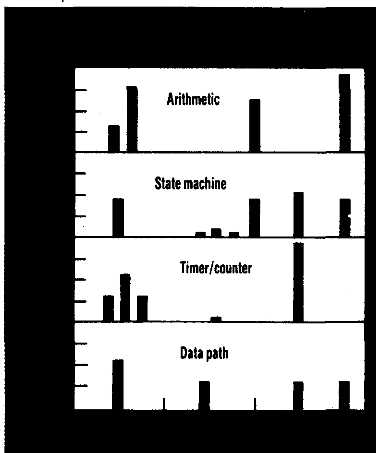
ASIC BENCHMARKS



4. THE ARITHMETIC CIRCUIT, a basic multiplier/accumulator, employs the highest number of gates out of all the benchmarks—295.

be loaded in two different ways: directly from incoming data or from the preset value register. The desired preset data is loaded into the counter when the "equal-to" comparator indicates a match between the counter's state and the value in a compare register.

Both the preset value and the compare value registers are similar to LS377 octal registers with Clock-Enables except that these also have an asynchronous reset function. The timer-counter's basic sequence of operations (count, compare, load) is repeated endlessly.



5. A COMPARISON OF pin density distributions is a good measure of routability. High pins-per-gate circuits are generally difficult to route.

The timer-counter circuit employs an intense amount of registers and exclusive-Or (XOR) gates. Registers hold the preset and compare values as well as implementing operation of the counter. The XOR gates, which are low pin-per-gate functions, are used in the counter and compare functions.

In the pin-per-gate distribution, the sequential functions have less than one pin per gate, and the simple gates have about 2.5 pins per gate. The timer-counter benchmark circuit requires about 248 gates to implement in an LL7220 gate array.

Benchmark number three is the state machine circuit representing an average-sized controller-sequencer that's found in many PLD applications *(Fig. 3)*. Because of the large amount of nonstandard logic involved, this is an important application for an ASIC.

This third benchmark circuit is a Mealy machine, whose outputs are determined from the total state of the system. The outputs are registered and clocked to produce synchronous output data. The machine has eight states and goes through 12 transitions.

The circuit's distinctive feature as a benchmark is its use of many wide decode stages (12 bits wide) in the combinatorial logic section, which demands a relatively large number of pins per gate. Consequently, this benchmark tests for how well an ASIC can handle highly congested circuits with high fan-in on each macro. Because they generally have a limited number of internal pins per module, the circuit poses a challenge to module-based ASICs.

Many LS273-type registers are included on this benchmark circuit. The combinatorial portion of a typical state-machine circuit has an extremely high pin-per-gate rating. The state-machine benchmark utilizes about 153 gates in an LL7220 gate array.

The final benchmark is the arithmetic circuit, more specifically a multiplier/accumulator (MAC) *(Fig. 4)*. This circuit consists of a multiplier, an 8-bit adder, and an 8-bit register. Such arithmetic circuits are employed in FIR filters and other DSP-type applications.

A 4-by-4-bit multiplier accepts either outside data or data from a preceding benchmark circuit unit. The 8-bit multiplication results are added to previous accumulator results, and then the current 8-bit accumulator result is moved to the output register. The 1-bit MAC input controls a gating function on the adder's input.

The MAC circuit represents a combination of circuit types that test an ASIC's ability to handle logic with both high and low pin-per-gate counts. The adders, as combinatorial functions, are high pin-per-gate sections. However, the registers used in both the multiplier and the output register are low pin-per-gate functions.

The largest pin-per-gate frequency (three pins per gate) generally occurs with this type of benchmark circuit, owing to the adder tree used to make up the array-multiplier *(Fig. 5)*. The sequential content of this benchmark has less than one pin per gate, as expected. Although one of the simplest benchmark circuits from a conceptual point of view, the arithmetic benchmark requires the most gates in the LL7220 gate array—295.

STANDARD RESULTS

The CALC benchmark results for LSI Logic gate arrays can be compared to the results for other ASICs supplying a reliable standard. The

DESIGN APPLICATIONS

ASIC BENCHMARKS

LL7220, one of the industry's most successful arrays, is commonly referred to as a 2000-gate device—though it actually contains 2224 2-input NAND gate equivalents. With an 85% usability rating, the LL7220 supplies 1890 usable gates.

Actel's ACT1020 desktop-configurable, channeled gate array scores about the same as the LSI Logic LL7220 channeled gate array for all but one of the benchmarks *(see the table)*. Therefore, even though the ACT1020 gate array implements an application circuit in logic modules instead of interconnected transistors, the device can be thought of as a 2000-gate gate array.

These results were produced by first stepping and repeating the benchmark circuit, then using an automatic router to map the circuit into the device. The use of autorouting is critical because although it's always possible to achieve better results with a manual place-and-route procedure, few designers would prefer a manual procedure for an application circuit. Consequently, these circuit benchmarks will produce typical outcomes only when autorouting is used.☐

Bob Osann, a marketing consultant to Actel Corp., was previously a vice president of P-CAD, following the acquisition of Assisted Technology where he served as president and chief executive officer. Osann holds a BSEE from Cornell University, and holds three patents in the areas of consumer and industrial electronics.

Abbas El Gamal, a cofounder of Actel Corp. and an associate professor at Stanford University, was previously director of the Systems Research Laboratory at LSI Logic Corp. He holds a PhD in electrical engineering from Stanford University and is credited with a patent for Actel's device architecture.

Actel 955 East Arques Avenue, Sunnyvale, CA 94086 • (408) 739-1010

ACTEL CORPORATION DIRECT SALES OFFICES

955 E. Arques Ave.
Sunnyvale, CA 94086
Tel: (408) 739-1010

8130 McFadden Ave., Suite 109
Westminster, CA 92683
Tel: (714) 373-4488

425 N. Martingale Rd., Suite 800
Schaumburg, IL 60173
Tel: (708) 706-3866

1740 Mass Avenue
Boxborough, MA 01719
Tel: (508) 635-0010

9101 Guilford Road, Suite 107
Columbia, MD 21046
Tel: (301) 604-0111

2350 Lakeside Blvd., Suite 850
Richardson, TX 75082
Tel: (214) 235-8944

DOMESTIC REPRESENTATIVES

ALABAMA
Rep Inc. .... (205) 881-9270

ARIZONA
Luscombe Engineering .... (602) 949-9333

CALIFORNIA
Centaur (Calabasas) .... (818) 704-1655
Centaur (Irvine) .... (714) 261-2123
Centaur (San Diego) .... (619) 278-4950
I² Inc. (Santa Clara) .... (408) 988-3400
I² Inc. (Orangevale) .... (916) 989-0843

COLORADO
Luscombe Engineering .... (303) 772-3342

CONNECTICUT
CompRep Associates .... (203) 269-1145

FLORIDA
Sales Engineering Concepts (Altamonte Springs) .... (407) 682-4800
Sales Engineering Concepts (Deerfield Beach) .... (305) 426-4601

GEORGIA
Rep Inc. .... (404) 938-4358

ILLINOIS
Carlson Electronic Sales Associates .... (708) 956-8240

INDIANA
Giesting & Associates .... (317) 844-5222

IOWA
Carlson Electronic Sales Associates .... (319) 378-1450

KANSAS
DLE Electronics .... (316) 683-6400

MARYLAND
New Era Sales .... (301) 544-4100

MASSACHUSETTS
CompRep Associates .... (617) 329-3454

MICHIGAN
Giesting & Associates (Livonia) .... (313) 478-8106
Giesting & Associates (Comstock Park) .... (616) 784-9437

MINNESOTA
Gibb Electronic Sales .... (612) 935-4600

MISSOURI
John G. Macke Co. .... (314) 432-2830

NEW JERSEY
Nexus .... (201) 947-0151

NEW MEXICO
Luscombe Engineering .... (505) 883-0333

NEW YORK
L-MAR Associates (Apalachin) .... (607) 687-1828
L-MAR Associates (E. Rochester) .... (716) 381-9100
L-MAR Associates (Poughkeepsie) .... (914) 462-8025

NORTH CAROLINA
Rep Inc. (Charlotte) .... (704) 563-5554
Rep Inc. (Morrisville) .... (919) 469-9997

OHIO
Giesting & Associates (Cincinnati) .... (513) 385-1105
Giesting & Associates (Cleveland) .... (216) 261-9705

OREGON
L² Ltd. .... (503) 629-8555

PENNSYLVANIA
Omega Sales .... (215) 244-4000

TENNESSEE
Rep Inc. .... (615) 475-4105

TEXAS
OM Associates (Austin) .... (512) 794-9971
OM Associates (Houston) .... (713) 789-4426
OM Associates (Richardson) .... (214) 690-6746

UTAH
Luscombe Engineering .... (801) 565-9885

WASHINGTON
L² LTD. .... (206) 827-8555

WISCONSIN
Carlson Electronic Sales Associates .... (414) 476-2790

CANADA
Clark-Hurman Associates (Quebec) .... (514) 426-0453
Clark-Hurman Associates (Ontario-Brampton) .... (416) 840-6066
Clark-Hurman Associates (Ontario-Nepean) .... (613) 727-5626

DOMESTIC DISTRIBUTORS

WYLE LABORATORIES

ARIZONA
Phoenix .... (602) 437-2088
CALIFORNIA
Calabasas .... (818) 880-9000
Irvine .... (714) 863-9953
Rancho Cordova .... (916) 638-5282
San Diego .... (619) 565-9171
Santa Clara .... (408) 727-2500
COLORADO
Thornton .... (303) 457-9953
MASSACHUSETTS
Burlington .... (617) 272-7300
OREGON
Hillsboro .... (503) 640-6000
TEXAS
Austin .... (512) 345-8853
Houston .... (713) 879-9953
Richardson .... (214) 235-9953
UTAH
West Valley City .... (801) 974-9953
WASHINGTON
Redmond .... (206) 881-1150

PIONEER STANDARD ELECTRONICS

CONNECTICUT
Norwalk .... (203) 853-1515
ILLINOIS
Addison .... (312) 495-9680
INDIANA
Indianapolis .... (317) 573-0880
KANSAS
Lenexa .... (913) 492-0500
MASSACHUSETTS
Lexington .... (617) 861-9200
MICHIGAN
Grand Rapids .... (616) 698-1800
Livonia .... (313) 525-1800
MINNESOTA
Eden Prairie .... (612) 944-3355
MISSOURI
St. Louis .... (314) 432-4350
NEW JERSEY
Pine Brook .... (201) 575-3510
NEW YORK
Binghamton .... (607) 722-9300
Fairport .... (716) 381-7070
Woodbury .... (516) 921-8700
OHIO
Dayton .... (513) 236-9900
PENNSYLVANIA
Pittsburgh .... (412) 782-2300
TEXAS
Austin .... (512) 835-4000
Dallas .... (214) 386-7300
Houston .... (713) 988-5555

PIONEER TECHNOLOGIES

ALABAMA
Huntsville .... (205) 837-9300
CALIFORNIA
San Jose .... (408) 954-9100
FLORIDA
Altamonte Springs .... (407) 834-9090
Clearwater .... (813) 536-0445
Deerfield Beach .... (305) 428-8877
GEORGIA
Norcross .... (404) 448-1711
MARYLAND
Gaithersburg .... (301) 921-0660
NORTH CAROLINA
Charlotte .... (704) 527-8188
Durham .... (919) 544-5400
PENNSYLVANIA
Horsham .... (215) 674-4000

INTERNATIONAL DISTRIBUTORS

AUSTRALIA
Reptechnic (Nuetral Bay, NSW) .... (2) 953.9844
DENMARK
Nordisk Electronik AS (Herlev) .... (42) 84.20.00
EGYPT
SEE (Cairo) .... (2) 665.948
ENGLAND
Gothic-Crellon Ltd. (Wokingham) .... (0734) 78.88.78
Manhattan Skyline Ltd. (Maidenhead) .... (0628) 75.85.1
FINLAND
OY Fintronic AB (Helsinki) .... (0) 69.26.022
FRANCE
ASAP (Montigny Le Bretonneux) .... (1) 30.43.8233
Scaib S.A. (Rungis Cedex) .... (1) 46.87.2313
ISRAEL
A.S.T. Ltd. (Herzelia) .... (52) 58.33.55
KOREA
Eastern Electronics, Inc. (Seoul) .... (2) 566.0514
ITALY
LASI Elettronica S.p.A.(Milan) .... (2) 66.10.1370
NETHERLANDS
Transfer B.V. (Enschede) .... (53) 33.43.81
NORWAY
Nordisk Electronik AS (Hvalstad) .... (2) 84.50.70
SPAIN
Semiconductores S.A. (Barcelona) .... (1) 742.2313
SWEDEN
Traco AB (Farsta) .... (8) 93.00.00
SWITZERLAND
Omni Ray AG (Dietlikon) .... (1) 835.21.11
TAIWAN
SEED TECH Corporation (Taipei) .... (2) 521.1100
WEST GERMANY
bit-electronic AG (Munich) .... (089) 41.80.070
ECN/Dacom GmbH (Munich) .... (089) 96.09.080

Actel Corporation
955 East Arques Avenue
Sunnyvale, CA 94086
408.739.1010

Technical Hotline 800.262.1060

5172020-0