Achieving High Performance in Military and Aerospace Applications

White Paper

December 2003

Table of Contents

Performance is becoming a greater factor in military and aerospace designs. This need for speed is driven by one factor – increasing bandwidth. Next-generation optical and infrared image sensors, high-resolution cock3pit displays, and the requirements for network-centric warfare are all increasing the need for greater bandwidth. These high data-rates require the use of high-speed interfaces such as 66 MHz/64-bit PCI and 1 Gb/s Fibre Channel to enable communication within and between systems.

In addition, designers are faced with the task of achieving higher levels of integration, while working within unforgiving weight and space constraints. Only one FPGA family meets all of these needs – high performance, high density, and minimal footprint – over the full military operating range: Actel's Axcelerator family. An intrinsically fast architecture combined with high performance design techniques and the latest tools to optimize performance provide a comprehensive solution that will meet the needs of the most demanding high-performance system, even over the full military temperature range.

High Speed FPGA Architecture

A High-Performance Architecture

In order for an FPGA family to meet the needs of high-performance designs, it must be based on a high-performance architecture. The Axcelerator family of FPGAs has such an architecture, with:

- Fully-fracturable logic clusters
- High-speed embedded memory blocks with integrated FIFO controllers
- A flexible clocking system
- A non-hierarchical routing structure
- High-speed I/Os with dedicated DDR support

All of these features work in conjunction to form a high-performance architecture.

Fully-Fracturable Logic Module Cluster

Like many FPGA architectures, Axcelerator organizes its logic modules into clusters. Each logic module can be used independently of the other modules within the same logic cluster and can be combined with any other logic module in the FPGA. Therefore, each logic cluster is fully fracturable, a unique feature of the Axcelerator architecture. A fully-fracturable logic module cluster leads not only to greater levels of efficiency, enabling high logic module utilization rates, but also eases the burden on the placement software. Allowing the placer greater flexibility decreases the distance between logic modules, resulting in smaller routing delays (Figure 1 on page 4).

Embedded Memory Blocks

Clearly, a high-performance FPGA architecture requires high-speed memory blocks, but attention needs to be paid to how memory is used in the end application. For example, in optical and infrared imaging applications, large amounts of data are moved from the sensor through an image processor and out to a display or communication link. Handling this data flow is done most efficiently through the use of FIFOs. The best way to construct high-speed FIFOs is to embed the control logic into the memory block, rather than building the controller out of programmable gates (as with traditional FPGA architectures). The Axcelerator architecture includes an embedded FIFO controller in each 4.6kbit, variable-aspect ratio, dual-port SRAM block, resulting in higher speeds and increased logic efficiency. The embedded controller eliminates the concern that a FIFO controller built using programmable gates would compete with the designer's logic for device resources or system performance.

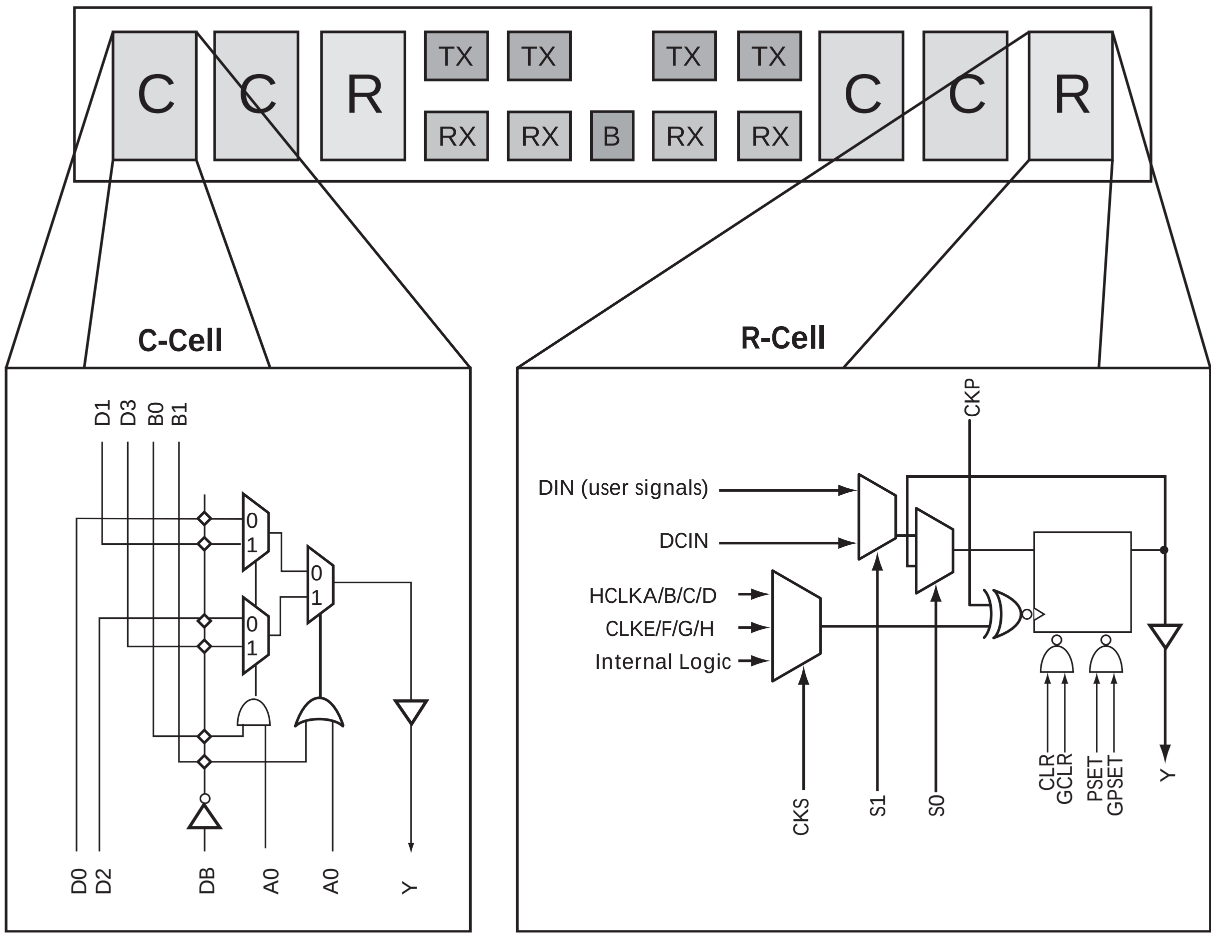


Figure 1: Fracturable Logic Module

In addition to the integrated FIFO controller, each embedded memory block inside an Axcelerator FPGA allows for the read and write clocks to operate asynchronously. This feature combined with special metastability control circuitry means that these FIFOs can be used to move data smoothly between clock and phase domains at speeds of over 450 MHz over the full military temperature range without using any internal logic.

Clocking Scheme

High-performance systems require a flexible clocking scheme that is transparent to the user and does not burden the design task. Every Axcelerator device is provided with eight segmentable, low-skew global resources. All eight global clock resources are available equally everywhere on the chip, thereby eliminating the need for clock floorplanning. Each global clock can be driven by external clock pins, its associated PLL, or by internal logic.

Moreover, the Axcelerator architecture allows for additional clock networks to be constructed out of general routing resources for maximum flexibility.

Routing Structure

The routing scheme of an FPGA is often overlooked, but it is in fact the single most important aspect of any high-performance programmable logic architecture. The routing structure determines how efficient the architecture is and ultimately determines the maximum performance possible from the FPGA.

The speed of the interconnect element is important, but size also plays a significant role. The smaller the better as a smaller element allows for an abundance of interconnect resources. Abundant interconnect resources allow the routing software to routinely achieve close to the theoretical minimum Manhattan Rule wire lengths between logic modules and therefore minimal delays. Critical to Axcelerator's high-speed architecture is the antifuse programmable interconnect element that does not take up valuable substrate real estate, allowing for shorter distances between logic modules. This is in contrast to SRAM-based FPGAs, which use SRAM cells as configurable interconnect resources and thereby use up valuable die space and incurring performance penalties. The small size of the antifuse allows for seven times the interconnect resources in Axcelerator devices when compared to equivalent SRAM FPGAs.

The Axcelerator routing structure consists of high-speed local tracks, segmented local tracks of varying lengths, as well as cross-chip routing tracks. To improve routing delays of high fanout nets, these routing tracks also have access to dedicated routing buffers.

The routing scheme of Axcelerator FPGAs is non-hierarchical – logic modules are able to access high-speed routing tracks without having to go through short-distance resources first. This non-hierarchical routing scheme provides the router with more than one type of routing resource that can be used to connect any two logic modules (Figure 2).

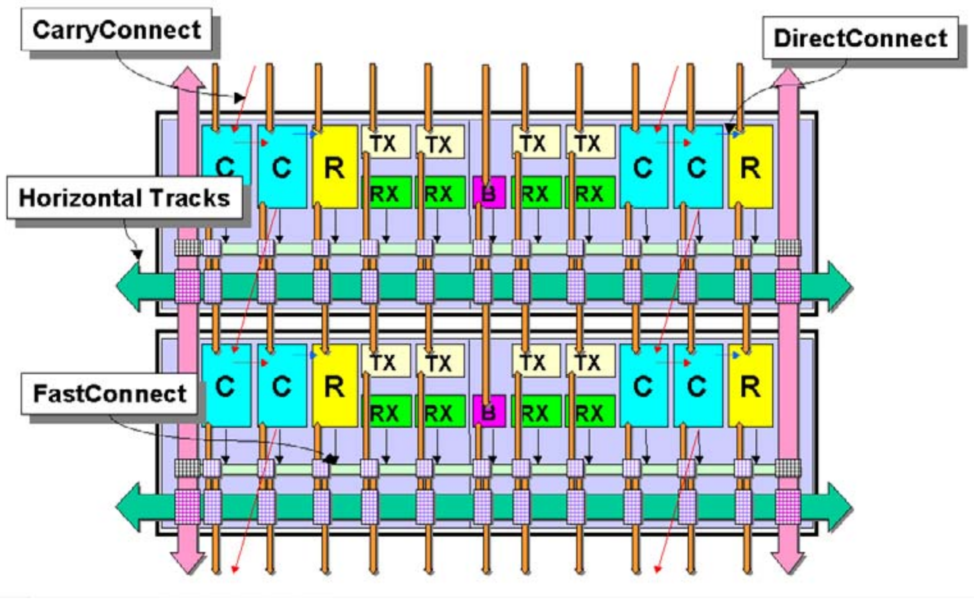


Figure 2: Routing Structure

High-Speed I/Os

High internal performance has little value if a designer cannot get data on and off the device. Each Axcelerator device is equipped with high-speed I/Os supporting a wide range of standards (such as LVDS, HSTL, PCI, etc. – 14 in all) and voltages (1.5V, 1.8V, 2.5V, and 3.3V). Each I/O can be configured to support any of the standards, and every pair of I/Os has built-in support for Double Data Rate (DDR) operation up to 630 Mbps over full military temperature, enabling the construction of high-speed interfaces to memories or other subsystems.

Performance over Mil-Temp – Real-World Examples

PCI

A common high-bandwidth interface used in today's advanced military and aerospace applications (for example, in single-board computer systems) is 66 MHz/64-bit PCI. As stated above, bandwidth in military applications is important, so an FPGA needs to meet or exceed the requirements of 66 MHz/64-bit PCI across the full military operating range, not just at commercial temperatures. Axcelerator is the only military-grade FPGA that exceeds the performance requirements for 66 MHz/64-bit PCI, even at 125ºC.

Gigabit Fibre Channel

One Gigabit per second Fibre Channel is fast becoming a de facto standard for intra-aircraft communication, linking the mission computer with various avionics subsystem or mass storage devices. In order to ensure system interoperability, the tight specifications of a Gigabit Fibre Channel must be met over the full military operating range. Again, Axcelerator is the only mil-temp FPGA that exceeds these performance requirements at 125ºC.

Military Axcelerator FPGAs offer up to 2M system gates, 288kbits of embedded memory, and 684 I/Os (Table 1).

Table 1: Axcelerator Military Product Family

	AX250	AX500	AX1000	AX2000
System Gates	250,000	500,000	1,000,000	2,000,000
ASIC Gates	30,000	60,000	125,000	250,000
R-Cells (dedicated flip-flops)	1,408	2,688	6,048	10,752
Total Modules	4,224	8,064	18,144	32,256
Embedded RAM Bits	54K	72K	162K	288K
Embedded RAM Blocks	12	16	36	64
PLLs	8	8	8	8
Global Networks	8	8	8	8
Max I/O	256	336	576	684
Plastic Packages	208-PQFP 256-FBGA 484-FBGA	208-PQFP 484-FBGA 676-FBGA	676-FBGA 896-FBGA 729-PBGA	896-FBGA 1152-FBGA
Hermetic Packages	208-CQFP 352-CQFP	208-CQFP 352-CQFP	352-CQFP 624-CCGA	352-CQFP 624-CCGA

High-Performance Design Tools and Techniques

Designers need tools and methodologies to help achieve high performance within the constraints of weight and space budgets, taking into account the resources made available by the target hardware architecture. Achieving timing closure in high performance or high complexity designs requires providing the synthesis tool with timing goals that accurately model real operating conditions. In addition, designers can take advantage of pre-optimized operator implementations offered either by the device manufacturer as a macro builder or by the EDA vendor as part of synthesis. LeonardoSpectrum™ and Precision® Synthesis from Mentor Graphics™ complement the ACTgen Macro Builder from Actel in providing an effective solution for achieving high performance (Figure 3).

```
library ieee ;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;

entity mults is
 generic (
  a_size : integer := 16 ;
  b_size : integer := 18 ;
  -- "level" dictates the amount of pipelining
  -- 1 : non-pipelined multiplier (single register at output)
  -- 2 : single-pipelined multiplier
  -- 3 : dual-pipelined multiplier
  level  : integer := 2
 ) ;
 port (
  clk     : in std_logic ;
  rst     : in std_logic ;
  a       : in std_logic_vector (a_size-1 downto 0) ;
  b       : in std_logic_vector (b_size-1 downto 0) ;
  product : out std_logic_vector (a_size + b_size - 1 downto 0)
 ) ;
end mults ;

architecture mentor of mults is

 type pipeline_stages is array (level-1 downto 0)
  of signed (a_size+b_size-1 downto 0) ;
 signal a_int       : signed (a_size - 1 downto 0);
 signal b_int       : signed (b_size-1 downto 0);
 signal product_int : pipeline_stages;

begin

 process(clk, rst)
 begin
  if rst = '1' then
   a_int   <= (others => '0');
   b_int   <= (others => '0');
   product_int(0) <= (others => '0');
   for i in 1 to level-1 loop
    product_int (i) <= (others => '0');
   end loop;
  elsif clk'event and clk = '1' then
   a_int   <= signed (a);
   b_int   <= signed (b);
   product_int(0) <= a_int * b_int;
   for i in 1 to level-1 loop
    product_int (i) <= product_int (i-1);
   end loop;
  end if;
 end process;
 product <= std_logic_vector (product_int (level-1));

end mentor ;
```

Figure 3: Sample VHDL Code for a Pipelined Multiplier

Leveraging Operator Implementations

Synthesis providers offer proven operator implementations that can take advantage of the programmable logic fabric. LeonardoSpectrum and Precision Synthesis support a wide range of operators, providing area-efficient implementations for counters, adders, pipelined and combinatorial multipliers, typically found in higher performance applications. Even further performance gains can be realized for timing-critical blocks by using pre-optimized operator implementations provided by the ACTgen Macro Builder. Using this divide and conquer approach, the designer can identify the timing critical portions of their design and provide a fast implementation for those blocks, letting synthesis provide area-efficient implementations for the remaining blocks.

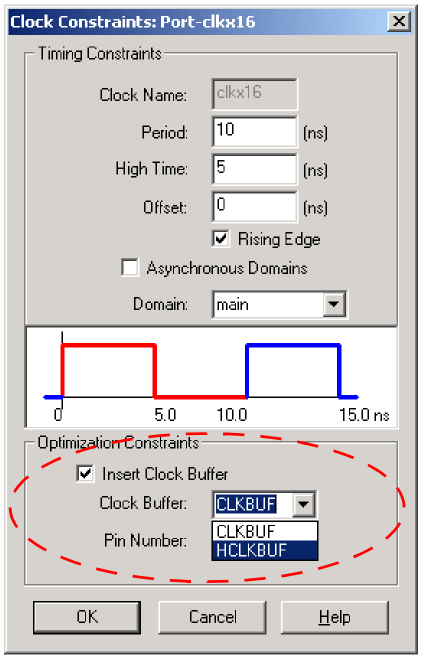
Pipelining should be considered for costly operators such as multipliers. Figure 3 on page 7 is an example circuit that synthesizes to a combinatorial or pipelined multiplier, depending on the value of the "level" generic. Synthesis can automatically extract pipelined operators and provide area-efficient implementations that better balance the delay across the pipeline stages. Table 2 demonstrates results from synthesizing a 16-bit by 18-bit multiplier targeting the AX250-FG256M in the –1 speed grade using military operating conditions. (Note that these results are from unconstrained synthesis and layout runs. Better results may be achieved with constraints). The data illustrates the potential benefit from pipelining – when R-cells are not in short supply; they can be put to good use in balancing the delay through the multiplier. Note that the number of C-cells remains unchanged, as synthesis is merely adding extra pipelining stages into the existing operator implementation. In this way, a 73% increase in overall clocking frequency is achievable with only a 19% increase in overall logic cells.

Table 2: Comparing Area vs. Delay for Various Multiplier Implementations

Pipeline Stages	Area		Delay through Multiplier
	C-Cells	**R-Cells**	
None (combinatorial)	879	68	22.17 ns
Single Stage	879	247	12.81 ns
Dual Stage	879	314	12.48 ns

Using the High-Performance Clock Network

To achieve the best performance, Actel recommends taking advantage of the eight segmentable global clocks on the device. Four of these are hardwired clocks that may only be routed to the clock pin of sequential elements; the remaining four can be used either as clock buffers or to buffer high-fanout nets such as control lines. Mapping to these high-performance clocks is accomplished either by instantiation or during synthesis. LeonardoSpectrum and Precision Synthesis automatically add global clock buffers to the highest fanout clocks in the design. Precision also allows the user to assume control over the insertion of these buffers; the user can specify which buffer is to be inserted as a clock or a net constraint. Figure 4 on page 9 illustrates specifying the clock buffer in the clock constraints dialog. The synthesis log file will indicate that the global buffer is added when performing final pad cell insertion. In the example in Figure 4 on page 9, the user has chosen to employ one of the hardwired clocks that may only be routed to the clock pin of sequential elements. Users should ensure that these clock nets are not feeding any non-clock loads. This can be accomplished with Precision Synthesis or LeonardoInsight™, either by tracing the clock net in the schematic viewer or by hovering the mouse pointer over the net and using query mode to extract the net's pin connections.



Clock Constraints: Port-clkx16

Timing Constraints

Clock Name: clkx16

Period: 10 (ns)

High Time: 5 (ns)

Offset: 0 (ns)

☑ Rising Edge

☐ Asynchronous Domains

Domain: main

Optimization Constraints

☑ Insert Clock Buffer

Clock Buffer: CLKBUF (CLKBUF, HCLKBUF)

Pin Number:

Figure 4: Specifying the Global Clock Buffer in Precision Synthesis

Embedded Memory Blocks and Memory Interfaces

Actel's ACTgen Macro Builder provides the designer with a wide variety of embedded memory configurations and memory interface pad cells such as DDR I/Os. The implementations of these can be written out either as an EDIF netlist or as structural VHDL or Verilog. One additional benefit offered by the macro builder is the ability to customize the port names of generated blocks, thus easing design-flow migration. LeonardoSpectrum and Precision Synthesis can import these structural netlists and link the macros with the remainder of the design as part of synthesis. Once the macro has been suitably customized and generated, the user should add the generated structural netlist file to the input file list or include it in their read script.

Constraining for Synthesis and Layout

Precision Synthesis fully supports the Synopsys Design Constraint language – the de facto standard method for chip designers to communicate their design constraints. In addition, the Actel Designer software supports a growing subset of this language for constraining its layout algorithms. Currently, the Actel Designer software supports two SDC constraints: “create_clock” and “set_max_delay.” Together, these two constraints can be used to constrain both the period of on-chip register-to-register paths, and the time required for signals to propagate on and off the chip through I/O pad cells.

Actel's "create_clock" implementation allows users to create real clocks at any point in their design, either at top-level ports or on internal pins. This constraint is used to check for setup violations on associated register-to-register timing paths. When setting clock constraints in Precision Synthesis, users should place all clocks in the same clock domain, "main," as depicted in Figure 4, as Actel's "create_clock" implementation does not support the "-domain" extension to SDC.

Actel's current implementation of "set_max_delay" requires fully-specifying both the "-from" and "-to" path endpoints. In the case of an input port, the "-from" argument should be the input port, and the "-to" should be the data input pin of the capturing register. In the case of an output port, the "-from" argument should be the clock input pin of the launching register, and the "-to" should be the output port.

When referencing internal netlist objects, be careful to refer to the post-compiled netlist object. The compile phase of Actel's Designer software substitutes soft macros for complex registers, thus adding an extra level of hierarchy. Refer to the design examples below for more detailed information.

Specifying Military Operating Conditions

LeonardoSpectrum, Precision Synthesis, and Actel's Designer software all allow the user to specify operating conditions for static timing analysis. Synthesis libraries contain an appropriate process derating for military operating conditions, and physical libraries allow for more detailed specification of voltage and temperature range. With Precision Synthesis, military operating conditions can be selected by issuing the command "setup_design -cim mil" as part of the project setup.

Example: Using Synopsys Design Constraints

Figure 5 and Figure 6 on page 11 demonstrate how a small UART design might be constrained for synthesis and layout targeting Axcelerator. This basic UART design is one of the example circuits included in both LeonardoSpectrum and Precision Synthesis.

```
##################
# Clocks
##################
create_clock  { clkx16 } -name clkx16 -period 10.00 -waveform { 0.00 5.00 }
create_clock  { write } -name write -period 100.00 -waveform { 0.00 50.00 }
create_clock  { reg_txclk/U0:Q } -name reg_txclk/out -period 20.00 -waveform { 0.00 10.00 }
create_clock  { reg_rxclk/U0:Q } -name reg_rxclk/out -period 20.00 -waveform { 0.00 10.00 }

###########################
# False Paths / Multicycles
###########################
set_max_delay 10.00 -to { framingerr }
set_max_delay 10.00 -to { tx }
set_max_delay 10.00 -to { overrun }
set_max_delay 10.00 -to { txrdy }
set_max_delay 10.00 -to { parityerr }
set_max_delay 10.00 -to { rxrdy }
set_max_delay 10.00 -from { rx }
```

Figure 5: Design Constraints File for Synthesis with Precision Synthesis

```
##################
# Clocks
##################
create_clock  { clkx16 } -period 10.00 -waveform { 0.00 5.00 }
create_clock  { write } -period 100.00 -waveform { 0.00 50.00 }
create_clock  { reg_txclk/U0:Q } -period 20.00 -waveform { 0.00 10.00 }
create_clock  { reg_rxclk/U0:Q } -period 20.00 -waveform { 0.00 10.00 }

###########################
# False Paths / Multicycles
###########################
set_max_delay 10.00 -from { reg_framingerr/U0:CLK } -to { framingerr }
set_max_delay 10.00 -from { reg_tx/U0:CLK } -to { tx }
set_max_delay 10.00 -from { reg_overrun/U0:CLK } -to { overrun }
set_max_delay 10.00 -from { reg_txdatardy/U0:CLK } -to { txrdy }
set_max_delay 10.00 -from { reg_parityerr/U0:CLK } -to { parityerr }
set_max_delay 10.00 -from { reg_rxdatardy/U0:CLK } -to { rxrdy }
set_max_delay 10.00 -from { rx } -to { i2f8ex3/U0:D reg_rxstop/U0:D reg_hunt/U0:D }
```

Figure 6: Design Constraints File for Layout

Conclusion

Together, Mentor Graphics and Actel provide tools and methodologies to help designers achieve high-performance design goals, taking into account the resources made available by the Axcelerator hardware architecture. In addition, Precision Physical Synthesis and Actel Designer software provide an effective means, using industry standards, to communicate design intent for both synthesis and layout. In addition to offering the military and aerospace designer a high-performance FPGA architecture, Axcelerator gives designers a:

- Live-at-power up
- True single-chip
- Firm-error immune
- Low-power

solution. When combined with design tools from Mentor Graphics, designers have a complete platform for high-performance military/aerospace designs.

For more information, visit our website at **http://www.actel.com**

www.actel.com

Actel Corporation
2061 Stierlin Court
Mountain View, CA
94043-4655 USA
Phone 650.318.4200
Fax 650.318.4600

Actel Europe Ltd.
Dunlop House, Riverside Way
Camberley, Surrey GU15 3YL
United Kingdom
Phone +44 (0)1276.401450
Fax +44 (0)1276.401490

Actel Japan
EXOS Ebisu Building 4F
1-24-14 Ebisu Shibuya-ku
Tokyo 150, Japan
Phone +81.03.3445.7671
Fax +81.03.3445.7668

Actel Hong Kong
39th Floor, One Pacific Place
88 Queensway, Admiralty
Hong Kong
Phone +852.227.35712
Fax +852.227.35999

© 2003 Actel Corporation. All rights reserved. Actel and the Actel logo are trademarks of Actel Corporation. LeonardoSpectrum is a trademark and Precision is a registered trademark of Mentor Graphics Corporation. All other brand or product names are the property of their respective owners.

55900006-0/11.03