WP209 (v1.0) January 12, 2004

Virtex Variable-Input LUT Architecture

By: Ralf Krueger and Brent Przybus

The variable-input look-up table (LUT) architecture has been a fundamental component of the Xilinx Virtex™ architecture first introduced in 1998. This unique architecture enables flexible implementation of any function with eight variable inputs, as well as implementation of more complex functions. In addition to being optimized for 4-, 5-, 6-, 7-, and 8-input LUT functions, the architecture is designed to support 32:1 multiplexers and Boolean functions with up to 79 inputs.

The Virtex architecture enables users to implement these functions with minimal levels of logic. By collapsing levels of logic, users can achieve superior design performance. This performance leadership is validated by benchmarks that show Virtex with an average 38% performance leadership over alternative programmable logic architectures.

Xilinx Virtex Architecture

The Virtex architecture introduced in 1998 is now in its third generation. While incremental improvements have been made, the Virtex architecture is still based on the same fundamental building block, the Configurable Logic Block (CLB). Each CLB contains two slices, which in turn contain two look-up tables (LUTs) each.

Beyond the Four-Input LUT

The current generation (Virtex-II architecture) includes dedicated multiplexers for combining LUTs, allowing devices to support up to eight inputs. These specialized multiplexers improve the performance, density, and size of wide logic that can be implemented in each CLB. The Virtex-II variable-input LUT architecture enables users to implement a 32:1 multiplexer, as well as Boolean logic functions with up to 79 inputs, in just one level of logic.

The Virtex-II architecture contains dedicated two-input multiplexers (one MUXF5 and one MUXFX per slice). These multiplexers combine the four-input LUT outputs or the outputs of other multiplexers. Using the multiplexers; MUXF5, MUXF6, MUXF7, and MUXF8, users can create 2-, 4-, 8-, and 16-input LUTs. Specific routing resources are associated with these two-input multiplexers to guarantee a fast implementation of any combinatorial function built upon LUTs and MUXFX.

The combination of LUTs and the MUXFX offers a unique solution for wide-input function designs. Any Virtex-II slice can implement a 4:1 multiplexer, any CLB can implement a 16:1 multiplexer, and two CLBs can implement a 32:1 multiplexer. The same logic resources also can be used for wide, general-purpose logic functions. For applications like comparators, encoder/decoders, or case statements, these resources provide an optimal solution.

In addition, the Virtex-II slices also contain a dedicated two-input multiplexer (MUXCY) and a two-input OR gate (ORCY) to perform operations involving wide AND and OR gates. These combine the four-input LUT outputs. These gates can be cascaded in a chain to provide wide AND functionality across slices. The output from the cascaded AND gates can then be combined with the dedicated ORCY to produce a Sum of Products (SOP) function.

Achieving Reduced Levels of Logic Without Compromising Flexibility

Additional logic available in the latest generation Virtex architecture provides the flexibility to implement functions optimized for four-input LUTs as well as more complex functions that require additional logic. The Virtex architecture's uniqueness enables users to implement a full range of functions while at the same time reducing the levels of logic required to implement them.

Wide Dedicated Multiplexers

The largest mux that a single LUT supports is a 2:1 mux, with the fourth input available as a possible enable. One method for constructing larger muxes is to cascade multiple LUTs. For example, a 4:1 mux can be built by combining the outputs of two LUTs into a third LUT. However, this method adds two full levels of logic delay plus an additional routing delay between the LUTs. To increase multiplexer speed and density, the Virtex-II architecture provides a dedicated 2:1 mux following every LUT, replacing additional levels of LUT-based logic. One of these LUTs, called the F5MUX, combines adjacent LUTs to create a 4:1 mux. The other mux, called the FXMUX, where the index "X" equals 6, 7, or 8, follows every pair of LUTs and combines muxes into even wider functions, with different capabilities, depending on its location in the CLB.

Since each LUT optionally implements a 2:1 multiplexer, the F5MUX and two LUTs can implement a 4:1 multiplexer. The F6MUX and two slices implement an 8:1

multiplexer. The F7MUX and the four slices of any CLB implement a 16:1 multiplexer, and the F8MUX and two CLBs implement a 32:1 multiplexer. Shown in Figure 1 is the implementation of an 8:1 and a 16:1 multiplexer in a single CLB.

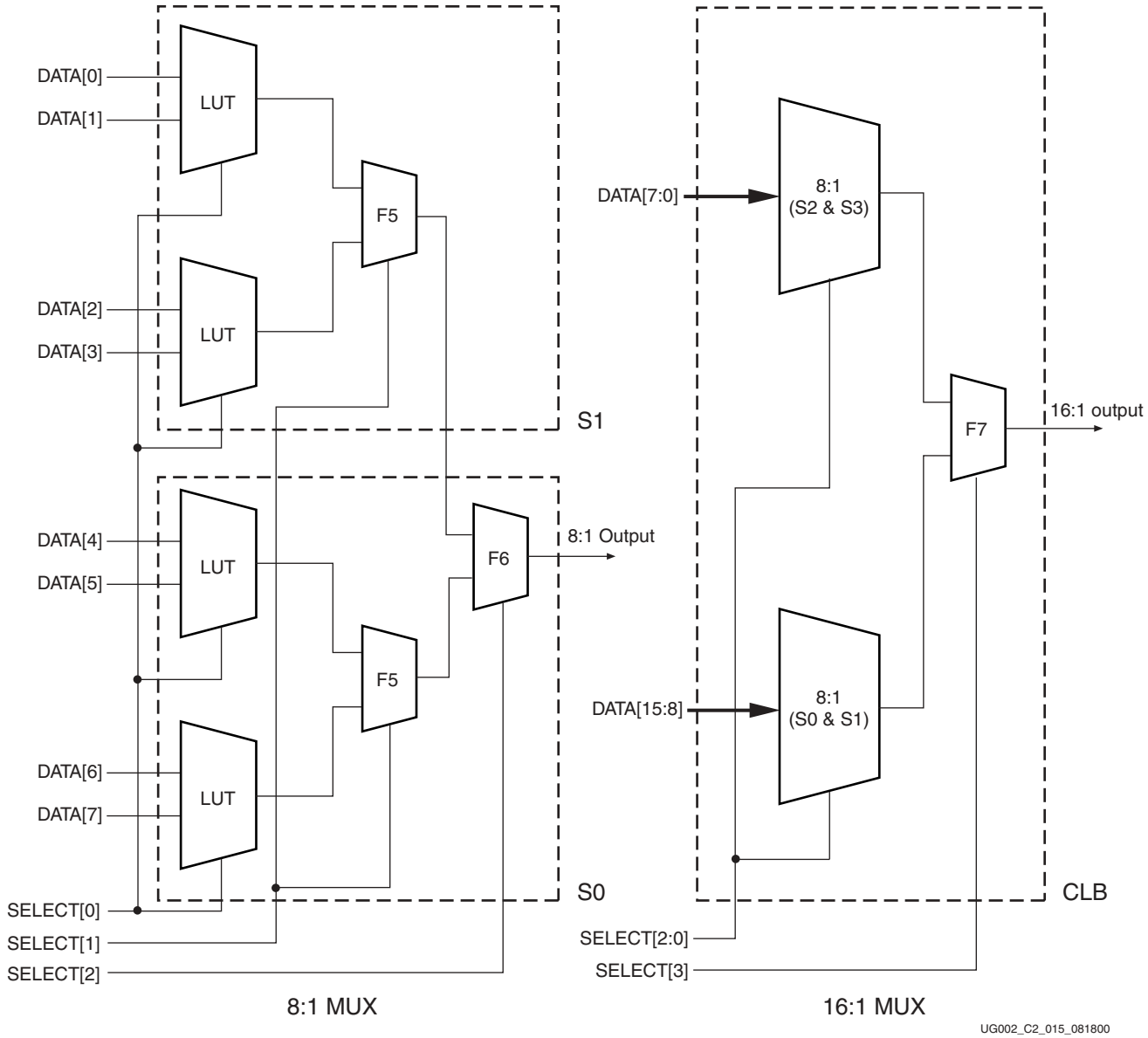

Figure 1: **8:1 and 16:1 Multiplexers in the Virtex-II Pro™ Architecture**

The entire range of multiplexers is fully supported in the Xilinx tool suite. Library elements exist for MUX2:1 through MUX31:1. The most common way to implement them is to simply have them inferred by synthesis tools when appropriate for a design. The library primitives can be used to instantiate specific multiplexers. Also, the CORE Generator system includes the Bus Multiplexer and Bit Multiplexer functions, and many other CORE solutions take advantage of the dedicated multiplexers.

Wide Gate Input Functions

FXMUX logic used to build the wider multiplexers can also implement other types of functions. The F6MUX is so named because it creates any function with six inputs. Similarly, the F7MUX generates any function with seven inputs, and the F8MUX generates any function with eight inputs.

Furthermore, FXMUX logic can be used to form limited custom Boolean functions, which can be either 39 inputs wide in a single CLB or 79 inputs wide in two CLBs with dedicated connections in a single level of logic. Figure 2 illustrates the possible implementation of a custom Boolean function with 39 inputs in a single CLB.

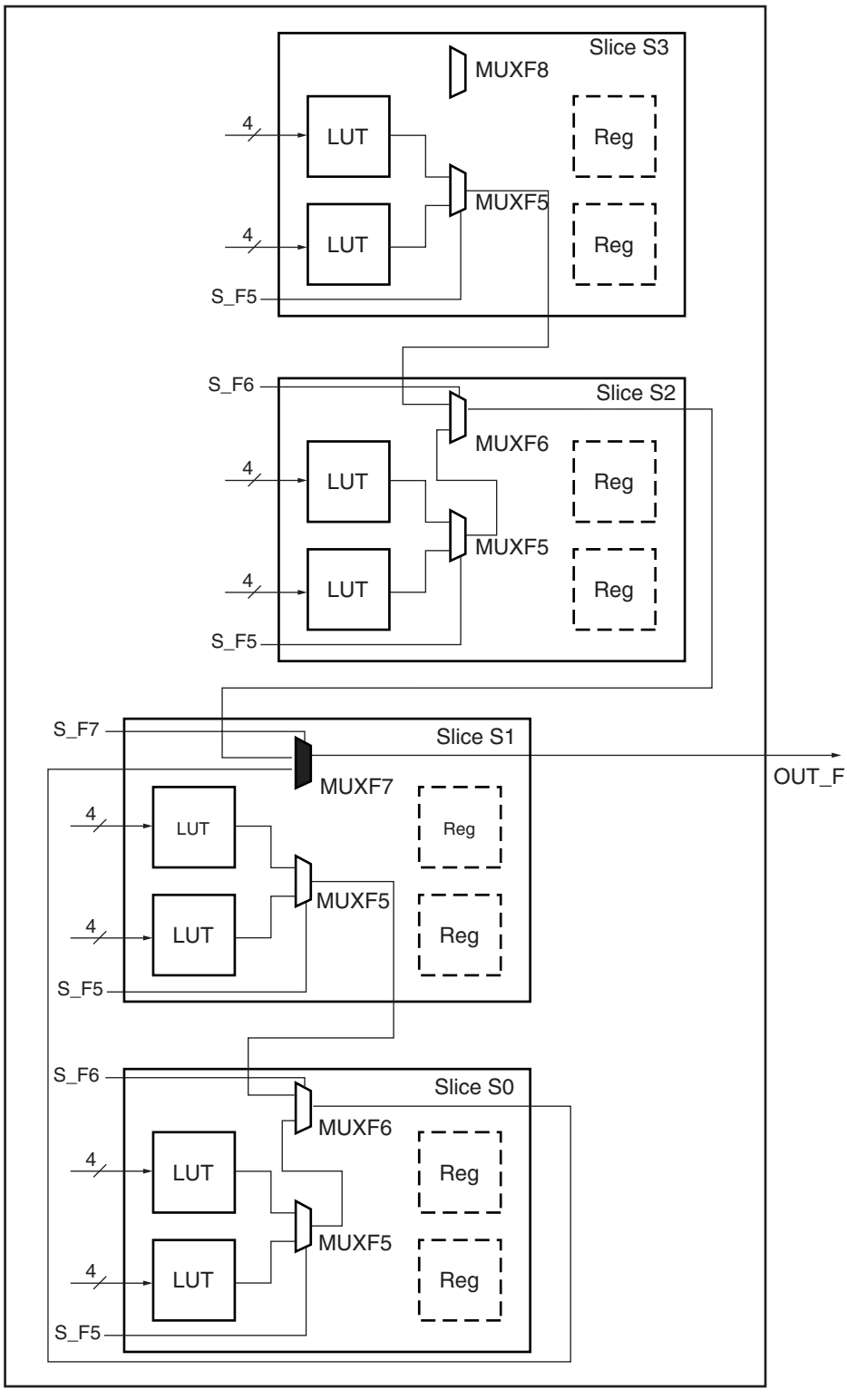

Figure 2: **LUTs (and MUXF5, MUXF6, and MUXF7) in One Virtex-II Pro CLB**

In addition to FXMUX capabilities, each Virtex-II slice contains dedicated two-input multiplexers (MUXCY) and a two-input OR gate (ORCY) to perform operations involving wide AND and OR gates. These combine the four-input LUT outputs. These gates can be cascaded in a chain to provide wide AND functionality across slices. The output from the cascaded AND gates can then be combined with the dedicated ORCY to produce a Sum of Products function.

The MUXCY uses the output from the LUTs as a SELECT signal. Depending on the width of data desired, several slices can be used to provide the sum output. Figure 3 shows the logic involved in designing a 16-input AND gate. It utilizes the four-input LUT to provide the necessary SELECT signal for the MUXCY. Only when all of the input signals are high can the VCC at the bottom reach the output. This use of carry logic helps to perform AND functions at high speed and saves logic resources.

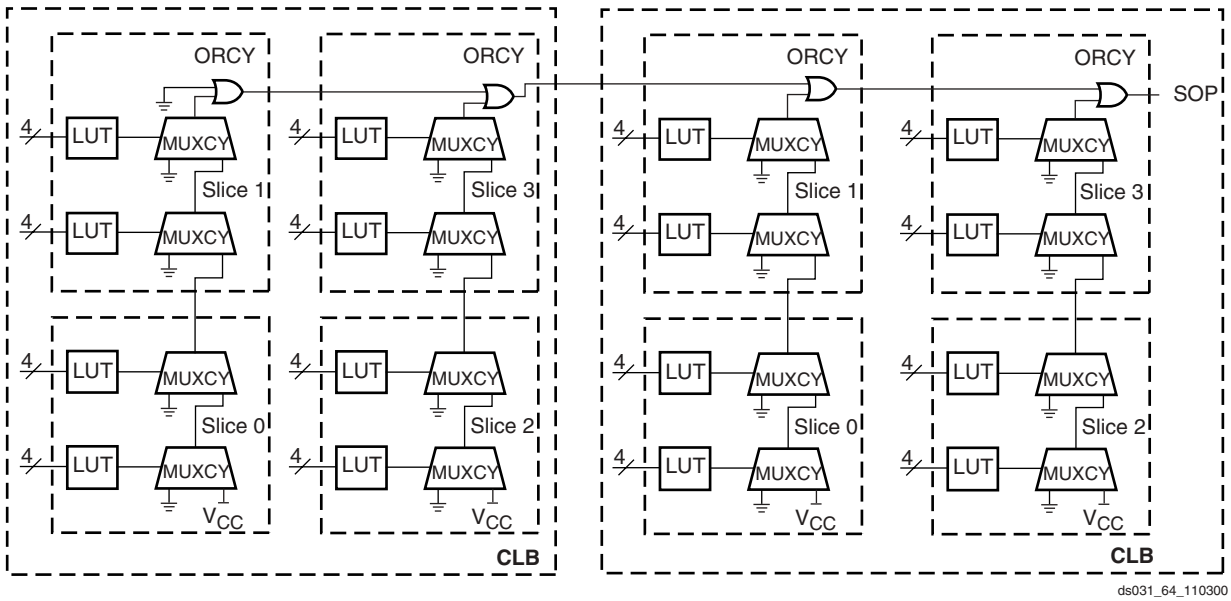

Figure 3: **Wide-Input AND Gate 16 Inputs Implemented in Virtex-II Pro Architecture**

Ease of Use

Xilinx tools allow users to take advantage of wide input gates automatically. The library contains elements such as AND16, NAND16, OR16, NOR16, XOR16, and many others. Many synthesis tools also infer wide input gates to be implemented in a single CLB. Various templates in Verilog and VHDL are available in User Guides, Applications Notes, and other documentation available at www.xilinx.com.

Achieving Higher Performance

In summary, the Virtex architecture enables users to flexibly implement FPGA designs with the highest level of performance. Figure 4 shows the percentage performance improvement over the leading competing FPGA architecture using 50 designs based on real applications. The resulting performance advantage illustrates the flexibility of the Virtex architecture in implementing functions that require variable-input widths, while at the same time achieving superior performance through the use of the dedicated routing and logic resources unique to the Xilinx Virtex architecture.

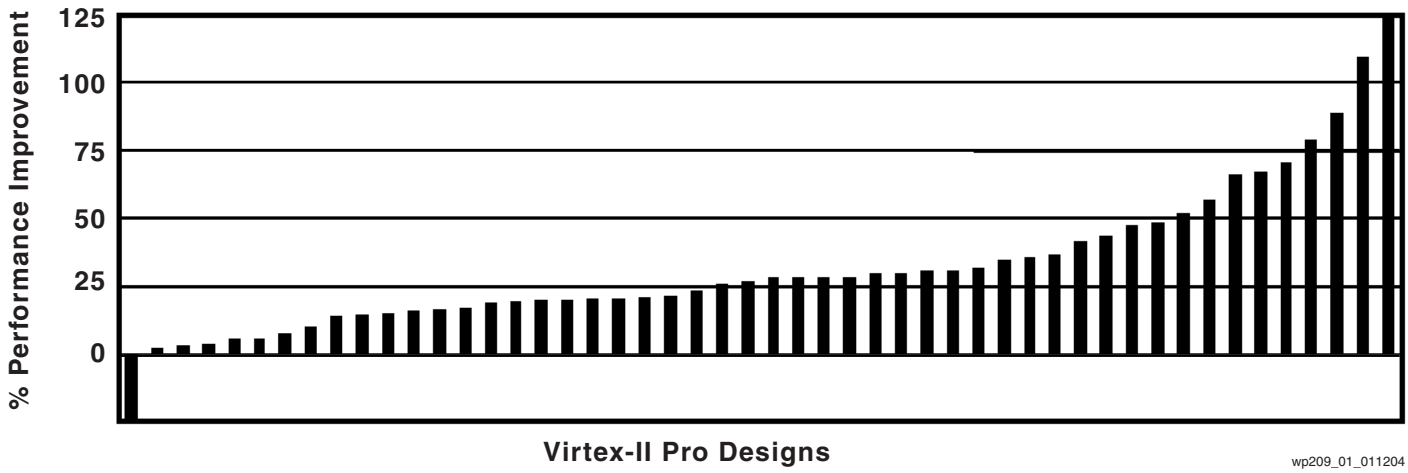

Figure 4: **Virtex-II Pro Performance Test Results**

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
01/12/04	1.0	Initial Xilinx release.