FPGA DESIGN RELIABILITY III
THE PROPER USE OF FPGA RESOURCES

This paper is the third in a series addressing how to build a robust FPGA-based design. This issue is intended to raise the consciousness of the reader to the proper use of various special resources in the FPGA. Each topic will be discussed briefly. Detailed explanations of each of these topics may be obtained by reading data sheets and application notes provided for a particular device by the manufacturer. A would-be FPGA designer is encouraged to thoroughly read the entire data sheet and pertinent application notes.

1. Design Guidance: Using Clock Buffers and Clock Components

The proper use of clock buffers and components has already been touched on in other tutorials. For thoroughness, guidelines will be reviewed here.

- Make use of dedicated clock buffers and clock nets for all clock domains. Avoid using general routing resources for clocks whenever possible.
- Signals crossing between clock domains must use the proper crossing circuitry. However, in some cases a designer can avoid this circuitry by using specialized clock components such as DLLs, PLLs, Phase Shifters, Frequency Synthesizers, and Clock Doublers to create one phase-locked clock domain from another.
- The use of clock enables was discussed previously. For clock enables that require high performance (i.e. high frequency enable signals), some FPGAs allow these enables to be placed on spare global clock networks.

2. Design Guidance: Using I/O Resources

FPGA manufacturers are increasingly providing specialized I/O resources on their devices, allowing the designer to interface directly to many different devices having various I/O standards and signaling schemes. This report cannot replace the detailed information on these capabilities provided by the various device manufacturers. Indeed, this is any area of rapid feature growth; information provided here would be rapidly outdated. Instead, we'll provide an overview of the choices available to the designer, and strategies to keep in mind when designing FPGA I/O.

2.1. Available I/O Standards

This section provides an overview of the I/O signaling standards available from the most prominent FPGA manufacturers. One aspect of reliable design – particularly in meeting timing requirements at the PCB level – is correctly matching I/O requirements of external devices with the capabilities of the FPGA. Some FPGA's partition their I/O pins into zones or *banks*, typically 8 banks per device. Each of these banks can be configured for a particular I/O standard. With some restrictions, it's possible to have two or more related, but different, I/O standards in a single bank. As part of the architectural planning for a design, it's essential to

determine all the interface requirements for the FPGA, and to determine how to map those requirements to the available I/O standards and banks. It is not necessary, or recommended, to pre-assign device pins at this point, but only to make sure that the I/O resources of the FPGA are sufficient.

The following is a sampling of single-ended I/O standards provided by Xilinx as of this writing:

- LVTTL (with selectable drive strengths and slew rates)
- LVCMOS (3.3V, 2.5V, 1.8V, and 1.5V)
- PCI (33 and 66 MHz)
- GTL and GTLP
- HSTL 1.5V and 1.8V (Class I, II, III, and IV)
- SSTL (3.3V and 2.5V, Class I and II)
- AGP, AGP2X

Xilinx provides, among others, the following differential I/O standards:

- LVDS and Extended LVDS (2.5V only)
- BLVDS (Bus LVDS)
- ULVDS
- LDT

A sampling of the single-ended and differential I/O Standards provided by Altera:

- 1.8-V, 2.5-V, 3.3-V, 5.0-V I/O
- 2.5-V I/O
- 3.3-V PCI and PCI-X
- 3.3-V Advanced Graphics Port (AGP, AGP2x)
- Center tap terminated (CTT)
- GTL+
- LVCMOS
- LVTTL
- True-LVDS and LVPECL data pins
- LVDS and LVPECL clock pins
- LVDS and LVPECL data pins up to156 Mbps
- HSTL Class I
- PCI-X
- SSTL-2 Class I and II
- SSTL-3 Class I and II

Some of the single-ended and differential I/O Standards provided by Actel:

- LVTTL (selectable slew rate and drive strength)
- 3.3Volt PCI
- LVCMOS2.5V

- LVCMOS1.8V
- LVCMOS1.5V (JESD8-11)
- Differential LVDS/LVPECL

Some of the single-ended I/O Standards provided by Atmel:

- LVTTL (selectable slew rate and drive strength)
- LVCMOS (selectable slew rate and drive strength)

2.2. Special I/O Architectures

In order to meet the modern demands of high-speed, high-bandwidth I/O, a designer must be familiar with the special architectural features built into I/O logic associated with the device pins. The more significant innovations are briefly reviewed here.

- Selectable input delays – User-selectable delay lines on FPGA data inputs. They guarantee adequate hold time relative to a clock signal brought into the FPGA.

- CDR – Many devices provide in-built clock and data recovery using high-speed dedicated clock PLLs and clock recovery units. The current generation of devices can receive data at rates over 3 GHz. from a differential input pair.

- I/O shift registers – high-speed serializers and de-serializers that are built into an FPGA's I/O logic to handle very high-speed serial data. They allow data to be moved between the FPGA core and its I/O logic at a much slower clock rate, in a wide, parallel format.

- Source-synchronous clocking – allows data I/O to be precisely synchronized to the clock of an external device, allowing data transmitted to that device to meet very tight timing margins.

- DDR I/O – Double data rate I/O refers to the ability to send and receive data using both edges of a clock. This typically requires extra flip-flops in the FPGA's I/O structure and precise clock shaping circuitry.

- 8b/10b conversion – A serial data coding technique that ensures sufficient transition density and limited run length—no more than five consecutive 1s or 0s—simplifying clock recovery and error detection, and eliminating direct current (DC) components from the serial stream.
- Rocket I/O – Xilinx trade name for high-speed serial differential I/O circuitry, comprised of PLLs, CDR, 8b/10b conversion, FIFOs, serializers/de-serializers, and source-synchronous clocking. It supports standards such as Infiniband, Fibre Channel, and Gigabit Ethernet.

2.3. Design Guidance: Using FPGA Memory

Every designer should become familiar with the memory resources provided within the FPGA. The resources being described are separate from the flip-flops associated with combinatorial logic in the logic blocks, logic elements, or similar units. They are separate resources provided to build relatively large memory-intensive structures within the FPGA core.

Altera provides 4096-bit and 9K (8192-bit plus parity) embedded system block memories (ESBs), which can be configured as single port RAM, dual port RAM, dual port RAM with independent read and write ports ("quad-port"), content addressable memory (CAM), and FIFOs. The CAM provides true parallel look-up operation for high-speed searches. The RAM configurations allow address depth to be traded off against data width, and different ports to have different data widths. For example, the ESB in quad port configuration can be written in ×1 mode (one bit wide) at port A, read in ×16 from port A, written in ×4 mode at port B, and read in ×2 mode from port B. All operations are synchronous, meaning that write data, address, and control are presented on each clock transition. Read data is provided from the RAM on each clock transition (following an integral number of clock delays to allow for pipelining the read address)

Xilinx provides 18 Kbit block RAMs (4.5 Kbits in original Virtex family) that can be configured as various FIFOs, single port, and dual port memories. (With the introduction of Virtex 6, a 36 Kbit variant has been added.) These, like their Altera equivalent, can be reshaped to trade off address depth against data width. Dual port configurations can have different width ports. These block RAMs can also be configured as CAMs, although the look-up operation is serial. All block RAM operations are synchronous.

Xilinx also allows their SRAM-based look-up tables (LUTs) to be used as actual RAM (this is sometimes referred to as distributed RAM or LUT-RAM). A wide variety of single port and dual port configurations, as well as FIFOs, can be built. A designer also has a lot of flexibility over depth and width, as these memories are built in increments of 16 bits. These resources work better for building small RAMs since there is a performance penalty to interconnect a large number of them. LUT-RAMs can operate in synchronous or asynchronous mode. Synchronous use is preferred, for reasons previously discussed.

2.4. Design Guidance: Other Special Features

While the features mentioned here are not integral to reliable FPGA design, knowledge of their existence and their use can make life easier for the designer by reducing his or her workload. Having more time to consider critical design practices (such as architectural planning and synchronous design) should result in a more reliable FPGA.

Some manufacturers provide dedicated hardware multiply circuitry. For example, Xilinx provides 18 x 18 multipliers spread across the die of its Virtex family, while Altera provides up to 16 x 16 multiplier primitives in its Mercury family, also spread around the die. The Xilinx logic is very fast and goes "wasted" if not used by the designer. The Altera multiplier resources are interwoven and shared with other logic components, but have high performance, and

therefore preferred over the use of general-purpose logic for multiplication. These multipliers are especially useful for DSP and similar applications.

While all FPGA manufacturers provide tri-state output pin capability, Xilinx, places tri-state buffers (TBUFs) within the interior of the FPGA. (In earlier families, these are actual tri-state buffers with relatively long turn-on times and the abilities to produce buffer fights. In later families, these shortcomings have been eliminated.) BUFTs allow the designer to build bi-directional buses and complex multiplexers within the FPGA without consuming combinatorial logic. With proper coding, the same HDL that results in BUFT structures in a Xilinx part will infer a combinatorial equivalent in devices without internal BUFTs.

Some devices allow a programmable tradeoff of power vs. speed for output pins and internal structures. This allows the designer to reduce power consumption, and thus thermal dissipation.

Both Altera and Xilinx offer an embedded logic analyzer core as a pre-compiled macro.

This provides the ability to monitor design operation over a time interval through the IEEE Std. 1149.1 JTAG circuitry. A designer can analyze internal logic at speed without bringing internal signals to the I/O pins. This feature is particularly useful for bench-top debugging if dedicated test pins have not been brought to a test header.

2.5. Design Guidance: Dedicated Arithmetic Logic

Most FPGAs provide dedicated logic to "assist" the standard combinatorial resources to create arithmetic functions – adders, comparators, and counters, for example. For the most part, using these features is transparent to the designer. Modern synthesis tools will infer them when arithmetic HDL statements are compiled. When schematic entry is used, the dedicated logic is incorporated within library macros, invoked by placing the corresponding symbols onto the schematic.

2.6. Design Guidance: Design of Very High Speed Logic

While FPGAs continue to improve their speed, and their manufacturers publish increasingly aggressive performance figures, it is unlikely that an entire FPGA, or even a significant portion of the die, can be made to operate at anything close to the published speed. Even if aggressive pipelining is used (see FPGA DESIGN RELIABILITY I), resulting in reduced combinatorial logic between register stages, the tendency for large circuits to contain a significant number of long interconnections increases the routing-related delays. However, it's frequent that only small pieces of large FPGA designs need to run at an aggressive speed, and thereby possible to compartmentalize the small amount of high-speed circuitry and clock it much faster than the bulk of the design. Because the small amount of high-speed logic can be confined to a tiny area of the die, route-based delays can be minimized.

The trick is to recognize exactly which piece, or pieces, of logic must be clocked at the high rate, and whether that logic can be successfully segmented away from the majority of the logic. This

recognition takes both cleverness and experience, but we will list a few examples to enable the reader to get the idea.

2.6.1. Very High-Speed Counters

If the purpose of a counter is simply to determine an elapsed time or digitally divide a high-speed signal, a designer might consider a counter with characteristics of a shift register, such as a Johnson counter or linear feedback shift register (LFSR). These structures have very little combinatorial logic between register stages. They can usually be packed into a small die area, minimizing routing delay.

However, if an actual binary count sequence is required, counters more than a few bits wide cannot run at or near the FPGA's peak clock rate due to the large amount of signal fan-in to the most significant flip-flop and its neighbors. As the counter becomes wider, the problem becomes more severe. However, while these most significant bits of the counter have the greatest combinatorial and routing delays, they actually toggle the least frequently. We can take advantage of this fact by designing a *pre-scaler* for the counter.

In this example, we'll consider a two bit pre-scaler. This means that first two (least significant) stages of the counter will run at the full clock rate, for instance, 400 MHz. Every time the pre-scaler rolls over (11 to 00), it outputs a terminal count pulse that is used as a clock enable for all other bits in the counter. In this case, since the pre-scaler divides by 4, the rest of the counter need only run at 400/4 = 100 MHz.

The entire counter, including the pre-scaler, is clocked at 200 MHz, but the designer specifies a 5 ns clock period constraint for the flip-flops comprising the pre-scaler, while specifying a separate 20 ns constraint for the remaining flip-flops. There is one additional requirement. The clock enable must propagate from the pre-scaler to the remaining flip-flops at the 200MHz. rate. Therefore, the designer must make sure the 5 ns constraint covers the clock enable, or must explicitly specify a 5 ns constraint for all delay paths originating in the pre-scaler and terminating at the "slow" flip-flops of the counter.

2.6.2. Very High-Speed I/O

The I/O performance of FPGAs exceeds 10 gigabits/second per serial differential pair as of this writing. Double data rate (DDR) memories have data rates over 800 MHz. It's beyond the capabilities of current FPGAs to move the data through interior logic at these rates. Handling data at this rate within the core of the FPGA requires increasing the data width, thus reducing the required clock speed. For example, a 16-bit memory interface producing data at 800 MHz could be quadrupled in width to 64 bits, and moved through the FPGA at 200 MHz, an easily managed rate.

In the case of high speed serial I/O, the data could be serialized or de-serialized with an 8-bit shift register, which can be clocked at near the maximum toggle rate of the FPGA (some manufacturers provide dedicated logic in the I/O circuitry of the FPGA to build very fast shift registers). The corresponding byte data need only move through the FPGA at 1/8 the serial rate.