The Spartan 3e FPGA

What's inside the chip?
- How does it implement random logic?
- What other features can you use?
- What do all these things mean?
  - LUT, Slice, BRAM, DCM, IOB, CLB...
- Two important documents
  (linked to the class web site)
  - Spartan3e Family Complete Data Sheet
  - Spartan3e User Guide

What's on the chip?
- CLB (Configurable Logic Blocks)
  - Logic and flip flops
  - 1,164 CLBs on our chip
  - Each CLB is 4 Slices
  - 500k total “system gates”
- IOB (Input Output Blocks)
  - Communicate off chip
  - Our chip has 232 total pins in a 320 BGA package
- BRAM (Block RAM)
  - On-chip SRAM
  - 18k bits per block
  - 20 blocks on our chip
What’s on the chip?

- Multiplier
  - Custom 18x18 multiplier
  - One per RAM block...

DCM (Digital Clock Manager)
- Clock generation and distribution
- Four on our chip

Programmable Interconnect
- Connect everything together
- Perhaps the most critical part of the chip!

CLB: Configurable Logic Block
- 4 “Slices” per CLB
- The slices work together to make logic, flip flops, distributed RAM, or shift registers
- Connected to other CLBs through Switch Matrix

Left and Right Slices
- SRL16 = 16-bit shift register
- RAM16 = 16-bit RAM (16x1 bit memory)
- LUT4 = four-bit lookup table (16x1 bit memory)
- SLICE = slice that can be memory or logic
- SLICEL = slice that can only be logic

What’s Really in a Slice?
LUT 4 – Basic Building Block

- RAM memory: 4-bit input, 1-bit output
- Can implement any logic function of up to 4 inputs

Slice Muxes extend LUT4

Once CLB – up to LUT7
Logic-only (combinational)

Logic + register (sequential)

Just register

Fast Carry Path (arithmetic)
Mapping to CLBs

- Each LUT can go through a flip flop
- So, these circuits map to the same number of Slices

How about these?

CLB Summary

- Each CLB = 4 slices
- Each slice contains
  - 2 LUT-4
    - LUT can be random logic, or 16x1bit RAM or SR
  - 2 flip flop
  - MUXs
  - Carry logic
  - ISE reports how many slices you use
  - among lots of other things...

IO Blocks

- Connections to the outside world
  - Each pin can be configured a large number of ways
  - Different signaling voltages and drive currents

NOTE! No 5v!
Actually the most important part of the FPGA!
- Consumes the most area on the die
- Consumes the most power on the die
- In most cases, wires limit the performance
- But, hardly mentioned in the datasheet
  - People are more impressed with logic

RAM-programmable switches
- 2,270,208 bits of configuration RAM!
- Compare to 368,640 total bits of Block RAM
- or 74,752 total bits of Distributed RAM (LUTs)
- Hierarchical organization
  - Many fast, short wires with small drive
  - Fewer longer wires with high drive
  - LOTS of work goes into picking just the right mix!

Clock Routing
- Routed on a separate dedicated network
- Another reason to avoid gated clocks
- Recursive “Fish bone” network that minimizes clock skew
- Clocks come from off-chip, or from a DCM

Four types of wires

Interconnect
We’ve seen details of these already...

Behavioral Template

```
parameters: RAM_WIDTH = data_width;
parameters: RAM_ADDR_WIDTH = (data_addr_size);
```

```
begin(
    // The following code in case someone is going to customize the RAM
    // contents via an external file (use freadw() for binary data)
    freadw((data_file_name), &data);
    for (i = 0; i < data_width; i++) {
        for (j = 0; j < data_addr_size; j++) {
            // read from file
            data_addr[i][j] = data[j][i];
        }
    }
end
```

```
endmodule
```

Structural Template

```
// The example is for a 32-bit data width and 32-bit address size. 
// Configuration for different values will require adjusting the 
// quantities of the RAM blocks and the number of ports.
// The number of ports depends on the system requirements.
// The address inputs can be either single or dual.

module RAM(
    input [31:0] Address,
    input [31:0] Data_In,
    output [31:0] Data_Out
);
```

```
// The following code is responsible for creating the RAM
// objects based on the configuration parameters.
RAM_OBJECTS = (data_width, data_addr_size, number_of_ports);
```

```
endmodule
```

Distributed RAM

```
// Table 1: Single-Port and Dual-Port Distributed RAMs

<table>
<thead>
<tr>
<th>Type</th>
<th>Width</th>
<th>Single-Port Distributed RAMs</th>
<th>Dual-Port Distributed RAMs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Config1</td>
<td>32</td>
<td>Single Port</td>
<td>Dual Port</td>
</tr>
<tr>
<td>Config2</td>
<td>16</td>
<td>Single Port</td>
<td>Dual Port</td>
</tr>
<tr>
<td>Config3</td>
<td>8</td>
<td>Single Port</td>
<td>Dual Port</td>
</tr>
</tbody>
</table>
```

```
// Table 2: Dual-Port RAM Function

<table>
<thead>
<tr>
<th>Function</th>
<th>D</th>
<th>SPOK</th>
<th>DPOK</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>
```

Spartan XC3E500S

```
// Spartan XC3E500S is a member of the Virtex family of FPGAs. It is
// ideal for applications requiring high performance and low power.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>LUT</td>
<td>1,000</td>
</tr>
<tr>
<td>Slice</td>
<td>32</td>
</tr>
<tr>
<td>Slice pins</td>
<td>12</td>
</tr>
<tr>
<td>Slice pins</td>
<td>12</td>
</tr>
<tr>
<td>Slice pins</td>
<td>12</td>
</tr>
</tbody>
</table>
```

```
// Spartan XC3E500S is a member of the Virtex family of FPGAs. It is
// ideal for applications requiring high performance and low power.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>LUT</td>
<td>1,000</td>
</tr>
<tr>
<td>Slice</td>
<td>32</td>
</tr>
<tr>
<td>Slice pins</td>
<td>12</td>
</tr>
<tr>
<td>Slice pins</td>
<td>12</td>
</tr>
<tr>
<td>Slice pins</td>
<td>12</td>
</tr>
</tbody>
</table>
```
Distributed RAM

Digital Clock Manager (DCM)

DCMs integrate advanced clocking capabilities directly into the FPGA’s global clock distribution network. Consequently, DCMs solve a variety of common clocking issues, especially in high performance, high-frequency applications:

- Eliminate Clock Skew: either within the device or to external components, to improve overall system performance and to eliminate clock distribution delays.
- Phase Shift: a clock signal, either by a fixed fraction of a clock period or by incremental amounts.
- Multiply or Divide an Incoming Clock Frequency or synthesize a completely new frequency by a mixture of clock multiplication and division.
- Mirror, Forward, or Rebuffer a Clock Signal, often to deskew and convert the incoming clock signal to a different I/O standard—for example, forwarding and converting an incoming LVDS clock to 1500.
- Any or all of the above functions, simultaneously.
Conclusion

- FPGAs are complex beasts!
  - Made to be very general and flexible

ASIC vs. FPGA?
- Rule of thumb, FPGA about 5 times slower clock than ASIC
- FPGAs consume more power
- FPGAs are bigger for the same function
- ASICs are much more expensive to develop
  - NRE – Non-Recurring Engineering