Field Programmable Gate Arrays

- Configuration Memory
- Programmable Logic Blocks (PLBs)
- Programmable Input/Output Cells
- Programmable Interconnect

Typical Complexity = 5 million – 1 billion transistors
Basic FPGA Operation

- Writing configuration memory ⇒ defines system function
  - Input/Output Cells
  - Logic in PLBs
  - Connections between PLBs & I/O cells

- Changing configuration memory data ⇒ changes system function
  - Can change at anytime
  - Even while system function is in operation
Combinational Logic Functions

- Gates are combined to create complex circuits
- Multiplexer example
  - If $S = 0$, $Z = A$
  - If $S = 1$, $Z = B$
  - Very common digital circuit
  - Heavily used in FPGAs
    - $S$ input controlled by configuration memory bit
    - We’ll see it again
Look-up Tables

- Recall multiplexer example
- Configuration memory holds outputs for truth table
- Internal signals connect to control signals of multiplexers to select value of truth table for any given input value

<table>
<thead>
<tr>
<th>S</th>
<th>A</th>
<th>B</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Look-up Table Based RAMs

- Normal LUT mode performs read operations
- Address decoder with write enable generates clock signals to latches for write operations
- Small RAMs but can be combined for larger RAMs
Basic PLB Architecture

- Look-up Table (LUT) implements truth table
- Memory elements:
  - Flip-flop/latch
  - Some FPGAs - LUTs can also implement small RAMs
- Carry & control logic implements fast adders/subtractors
A Simple PLB

- Two 3-input LUTs
  - Can implement any 4-input combinational logic function

- 1 flip-flop
  - Programmable:
    - Active levels
    - Clock edge
    - Set/reset

- 22 configuration memory bits
  - 8 per LUT
    - C0-7
    - S0-7
  - 6 control bits
    - CB0-5

C. Stroud 8/06
Xilinx FPGAs

- **Virtex and Spartan 2**
  - Array of 96 to 6,144 PLBs
    - 4 LUTs/RAMs (4-input)
    - 4 FF/latches
  - 4 to 32 4K-bit dual-port RAMs

- **Virtex II, Virtex II Pro**
  - Array of 352 to 11,204 PLBs
    - 8 LUTs/RAMs (4-input)
    - 8 FF/latches
  - 12 to 444 18K-bit dual-port RAMs
  - 12 to 444 18×18-bit multipliers
  - 0 to 2 PowerPC processor cores

- **Virtex 4**
  - Array of 1,536 to 22,272 PLBs
    - 4 LUTs/RAMs (4-input)
    - 4 FF/latches
  - 48 to 552 18K-bit dual-port RAMs
    - Also operate as FIFOs
  - 32 to 512 DSP cores include:
    - 0 to 2 PowerPC processor cores

- **Spartan 3**
  - Array of 192 to 8,320 PLBs
    - 4 LUTs/RAMs (4-input)
    - 4 LUTs (4-input)
    - 8 FF/latches
  - 48 to 552 18K-bit dual-port RAMs
  - 4 to 104 18K-bit dual-port RAMs
  - 4 to 104 18×18-bit multipliers
# Ranges of FPGA Resources

<table>
<thead>
<tr>
<th>FPGA Resource</th>
<th>Small FPGA</th>
<th>Large FPGA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logic</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PLBs per FPGA</td>
<td>256</td>
<td>25,920</td>
</tr>
<tr>
<td>LUTs and flip-flops per PLB</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>Routing</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wire segments per PLB</td>
<td>45</td>
<td>406</td>
</tr>
<tr>
<td>PIPs per PLB</td>
<td>139</td>
<td>3,462</td>
</tr>
<tr>
<td>Specialized Cores</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bits per memory core</td>
<td>128</td>
<td>36,864</td>
</tr>
<tr>
<td>Memory cores per FPGA</td>
<td>16</td>
<td>576</td>
</tr>
<tr>
<td>DSP cores</td>
<td>0</td>
<td>512</td>
</tr>
<tr>
<td>Other</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Input/output cells</td>
<td>62</td>
<td>1,200</td>
</tr>
<tr>
<td>Configuration memory bits</td>
<td>42,104</td>
<td>79,704,832</td>
</tr>
</tbody>
</table>
What is Built-In Self-Test?

- **Basic idea:** Add circuitry to IC or PCB to facilitate testing itself
  - Only power and clock needed during BIST sequence
  - Pass/Fail result reported at end of BIST sequence
    - No need for external test equipment

- **Necessary components:**
  - Test Pattern Generator (TPG)
  - Output Response Analyzer (ORA)
  - For system level use:
    - Test controller
    - Input isolation

- **Benefits:** low testing time & cost

- **Penalties:** area overhead, performance

---

C. Stroud 8/06

Overview of FPGAs
BIST for FPGAs

- **Basic idea:** reprogram FPGA to test itself
- BIST logic disappears after test
  - No area overhead or performance penalties
- Applicable to all levels of testing
  - A generic test for a generic component
  - Independent of system function
- Good diagnostic resolution
  - **Logic:** Look-Up Table (LUT) or flip-flop
  - **Routing:** wire segment or switch
  - Reconfigure system function for fault-tolerance
- **Cost:** memory to store BIST configurations
Built-In Self-Test of FPGA Logic

- Program PLBs to functions as
  - Test Pattern Generators (TPGs)
    - 6 to 12-bit counter
  - Output Response Analyzers (ORAs)
    - Comparator and a latch
  - Blocks Under Test (BUTs)
- Flip BIST architecture to complete testing

Test Session 1

Test Session 2
Fault Injection Emulator

- Faulty FPGA are difficult to find
  - 1 FPGA with faulty PLB & 2 FPGAs with faulty routing

- We created a Fault Injection Emulator
  - Intercepts & modifies configuration bits prior to download
  - Fault Emulator can create multiple faults in:
    - PLBs: LUTs, flip-flops, etc.
    - Interconnect: shorts and opens in wires

System or BIST config

Fault mask

Download file

FPGA

C. Stroud 8/06

Overview of FPGAs 13
FPGA BIST Demonstration

- **Graphic User Interface**
  - Shows what is happening inside FPGA during BIST

- **Fault Injection Emulator**
  - Inserts faults into configuration data file
  - Emulated faults are downloaded with BIST configuration

- **Diagnostic algorithm applied to BIST results**
  - Identifies faulty resource