CSE372  
Digital Systems Organization and Design Lab  
Prof. Milo Martin  
Unit 2: Field Programmable Gate Arrays (FPGAs)

Field Programmable Gate Array (FPGA)  
- An alternative to a “custom” design  
  - A high-end custom design “mask set” is expensive (millions of $!)  

Advantages  
- Simplicity of gate-level design (no transistor-level design)  
- Fast time-to-market  
  - No manufacturing delay  
  - Can fix design errors over time (more like software)  

Disadvantages  
- Expensive: unit cost is higher  
- Inefficient: slower and more power hungry  

Result: good for low-volume or initial designs

Announcements  
- Lab 0  
  - Mostly good work  
    - Biggest problem: not following directions  
    - We will grade less leniently in future  
    - Coding style matters, make it human readable  

- Lab 1  
  - First demo today  
  - Due next week  
  - Questions/comments?  

- Today’s lecture:  
  - How FPGAs work

Early Programmable Logic Device...
Modern FPGA: Xilinx Vertex II

FPGA Design Flow

- Synthesis
  - Break design into well-define logic blocks
  - Examples:
    - 2-input gates
    - Only NANDs
    - Limited set of "standard cells" with three-inputs, one output

- Place and route
  - Custom flow: position the devices and wires that connect them
  - FPGA: configure logic blocks and interconnect

- Goals:
  - Reduce latency (performance)
  - Reduce area (cost)
  - Reduce power (performance and/or cost)

For Comparison: FGPA vs Pentium 4

Review: Logical Completeness

- AND, OR, NOT can implement ANY truth table

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th>S</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>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</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>

1. AND combinations that yield a "1" in the truth table
2. OR the results of the AND gates

Mechanical process, but many optimizations
Our Old Friend, The Full Adder

- Add two bits and carry-in produce one-bit sum and carry-out

\[
\begin{array}{cccc}
A & B & C_{in} & S & C_{out} \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 \\
0 & 1 & 0 & 1 & 0 \\
0 & 1 & 1 & 0 & 1 \\
1 & 0 & 0 & 1 & 0 \\
1 & 0 & 1 & 0 & 1 \\
1 & 1 & 0 & 0 & 1 \\
1 & 1 & 1 & 0 & 1 \\
1 & 1 & 1 & 1 & 1 \\
\end{array}
\]

A Better Full Adder

\[
\text{module full adder}(s, \text{cout}, a, b, \text{cin});
\text{output} s, \text{cout};
\text{input} a, b, \text{cin};
\text{xor}(t1, a, b);
\text{xor}(s, t1, \text{cin});
\text{and}(t2, t1, \text{cin});
\text{and}(t3, a, b);
\text{or}(\text{cout}, t2, t3);
\text{endmodule}
\]

A Simple (Fake) FPGA Substrate

How Do We “Route” Signals?

- Switch matrix
  - Each junction has 6 “switches”
  - Each switch is a pass gate

- Programming
  - Each pass gate controlled by 1-bit flip-flop
  - 0/1 value of flip-flop set at configuration

- Programmable “interconnect”
  - Allows for arbitrary routing of signals
  - Each segment adds delay
  - Takes up lots of chip area

\[
\begin{array}{ccc|c}
C & \text{In} & O \\
0 & 0 & 0 \\
0 & 1 & 0 \\
1 & 0 & 0 \\
1 & 1 & 1 \\
\end{array}
\]
On-Chip Wires

More Wires

Configure This As a Full Adder

A Better FPGA

• Replace gates with general “CLB”
  • Combinational logic block
Combinational Logic Block

- Simple example CLB
  - Configure as any two-input gate
  - Use 2-input, 4-bit RAM to implement function
    - LUT - Lookup Table
    - Simple lookup operation
- Add sequential state
  - Add a latch/flipflop or two
  - One option: repurpose 4-bit memory
    - Configure CLB as either a LUT or RAM

Two 4-input LUTs
- Any 4-input function
- Limited 5-input functions
- Two flip-flops
- Fast carry logic (direct connect from adjacent CLBs)
- LUTs can be configured as RAM:
  - 2x16 bit or 1x32 bit, single ported
  - 1x16 bit dual ported
- Routing
  - Short and long wires (skip some CLBs)
  - Clocks have dedicated wires
- Also has IOBs (input/output blocks)
  - Specialized for off-chip signals, one per pin on package

The Xilinx 4000 CLB

The Xilinx 4000 CLB

Two 4-input functions, registered output
5-input function, combinational output

From UC-Berkeley CS150 slides

CLB Used as RAM

From UC-Berkeley CS150 slides

Fast Carry Logic

Xilinx 4000 Interconnect

From UC-Berkeley CS150 slides
Switch Matrix

Xilinx 4000 Interconnect Details

FPGA Design Issues

Our FPGAs: Virtex-2 Pro XC2VP30
FPGA vs Custom Designs

- Downside of configurability
  - Wires are much slower on FPGAs
  - Logic is much slower on FPGAs

- However, FPGAs are “real” logic (not software)
  - Great for our prototyping

- “Synthesis to chip” an option ($$$)
  - Standard cell design (also called “ASIC flow”)
  - Hard coded, but based on synthesis design flow
  - Not as good as “full custom” as used by Intel, AMD, IBM

---

Vertex II Pro - Embedded Structures

- FPGA Fabric
- Embedded memories
- Hardwired multipliers
- Xilinx Vertex-II Pro

---

FPGA vs Custom Designs

Not to exact scale

---

Looking Forward...

- Lab work
  - Lab 1 due next week (demo and writeup)

- Next lecture (Feb 16)
  - Thoughts on the design process
  - P37x datapath discussion

- After that (Feb 23)
  - CSE371 midterm