55:131
Introduction to VLSI Design

Implementation Options
Adapted from Weste and Harris notes
Implementation Options

- Use an existing chip
- Field Programmable Gate Array
- Sea of Gates/Gate Array
- Standard Cell
- Full Custom
Implementation Options

- Use an existing chip
  - Pros
    - Minimal hardware development cost – hardware is “done”!
  - Cons
    - May need a (big) software development!
    - Other features may not be a great fit
    - Beware of performance limitations
    - Beware of power consumption
    - Don’t forget about other hardware (memories, PHYs, etc)
  - Examples
    - Off-the-shelf microprocessor
    - Application Specific device
Applied Micro PPC460EX

Courtesy of Applied Micro (APM)
Implementation Options

- FPGAs or PLDs
  - Pros
    - Back-end (fabrication) is done
      - Saves time and money
  - Cons
    - Clock rate usually won’t match that of a Gate Array/Std Cell
    - Usually has higher recurring cost
    - “Fixed” feature set (RAM, PLLs, etc) may limit your options
  - Examples
    - RAM-based FPGA from Xilinx or Altera
    - Flash-based FPGA from Actel
Simplified FPGA Floorplan

Configurable Logic Block (CLB)

Routing Switch

Routing
“Configured” FPGA

Routing Switch

Routing

55:131 Introduction to VLSI Design 7
Xilinx CLB – a.k.a. “Slice”
4-input Look-up Table Implementation

Out = f (in0, in1, in2, in3)

16 SRAM Cells

Out = f (in0, in1, in2, in3)
Implementation Options

- **Sea of Gates/Gate Array**
  - **Pros**
    - Much of the fabrication is done
      - Base wafers have transistors and memories done
      - “User”-defined logic implemented in (a few) metal layers
    - Lower development costs than Standard Cell because supplier can amortize mask costs over all base array designs
    - Better clock rate and lower power than FPGA
  - **Cons**
    - Clock rate usually won’t match that of a Std Cell
    - “Fixed” feature set (RAM, PLLs, etc) may limit your options
  - **Examples**
    - LSI Logic LCA500K
Simplified Sea of Gates Floorplan
SoG and Gate Array Cell Layouts

(a) 

(b)
SoG and Gate Array 3-in NAND
Implementation Options

- Standard Cell
  - Pros
    - Smaller, faster, lower power than FPGA or SoG/GA
    - Faster development time than full custom because a standard cell library is used as the basic building blocks of the design
  - Cons
    - Custom mask set is expensive
    - Resistance to add library elements
    - Economical only for
      - High volume or
      - High cost devices
  - Examples
    - AMIS SC13 standard cell
Standard Cell Layout

- Standard Cell Rows
- Custom or Special Purpose Block
- RAM
# Standard Cell Library Example

<table>
<thead>
<tr>
<th>Gate Type</th>
<th>Variations</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inverter / buffer / tristate buffers</td>
<td></td>
<td>Wide range of power options, 1X, 2X, 4X, 8X, 16X, 32X, 64X minimum size inverter</td>
</tr>
<tr>
<td>NAND / AND</td>
<td>2–8 inputs</td>
<td>High, normal, low power</td>
</tr>
<tr>
<td>NOR / OR</td>
<td>2–8 inputs</td>
<td>High, normal, low power</td>
</tr>
<tr>
<td>XOR / XNOR</td>
<td></td>
<td>High, normal, low power</td>
</tr>
<tr>
<td>AOI / OAI</td>
<td></td>
<td>High, normal, low power</td>
</tr>
<tr>
<td>Multiplexers</td>
<td>Inverting/noninverting</td>
<td>High, normal, low power</td>
</tr>
<tr>
<td>Schmitt trigger</td>
<td></td>
<td>High, normal, low power</td>
</tr>
<tr>
<td>Adder / half adder</td>
<td></td>
<td>High, normal, low power</td>
</tr>
<tr>
<td>Latches</td>
<td></td>
<td>High, normal, low power</td>
</tr>
<tr>
<td>Flip-flops</td>
<td>D, with and without synch/asych set and reset, scan</td>
<td>High, normal, low power</td>
</tr>
<tr>
<td>I/O pads</td>
<td>Input, output, tristate, bidirectional, boundary scan, slew rate limited, crystal oscillator</td>
<td>Various drive levels (1–16 mA) and logic levels</td>
</tr>
</tbody>
</table>
Implementation Options

- Full Custom
  - Pros
    - Custom design at the physical level
    - Smallest, fastest, or lowest power circuit
      - You get exactly what you want
      - Or are capable of designing!
  - Cons
    - Design at physical level!
    - Custom mask set is expensive
    - Economical only for
      - High volume or
      - High cost devices
  - Examples
    - Microprocessor datapath, cache, IO cell
Custom Design Flow

Floorplan → Schematic or Netlist → Electrical Rule Check → Circuit Simulation → Layout Construction

Technology Rules → Technology Definition → Circuit Extract

Parasitic Extract → Schematic Backannotation → Resimulation → Design Rule Check

Reliability Check → Chip or Module OK
# CMOS Design Methods

<table>
<thead>
<tr>
<th>Design Method</th>
<th>Non-recurring Engineering</th>
<th>Unit Cost</th>
<th>Power Dissipation</th>
<th>Complexity of Implementation</th>
<th>Time to Market</th>
<th>Performance</th>
<th>Flexibility</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microprocessor/DSP</td>
<td>low</td>
<td>medium</td>
<td>high</td>
<td>low</td>
<td>low</td>
<td>low</td>
<td>high</td>
</tr>
<tr>
<td>PLA</td>
<td>low</td>
<td>medium</td>
<td>medium</td>
<td>low</td>
<td>low</td>
<td>medium</td>
<td>low</td>
</tr>
<tr>
<td>FPGA</td>
<td>low</td>
<td>high</td>
<td>medium</td>
<td>medium</td>
<td>low</td>
<td>medium</td>
<td>high</td>
</tr>
<tr>
<td>Gate Array/SOG</td>
<td>medium</td>
<td>medium</td>
<td>low</td>
<td>medium</td>
<td>medium</td>
<td>medium</td>
<td>medium</td>
</tr>
<tr>
<td>Cell Based</td>
<td>high</td>
<td>low</td>
<td>low</td>
<td>high</td>
<td>high</td>
<td>high</td>
<td>low</td>
</tr>
<tr>
<td>Custom Design</td>
<td>high</td>
<td>low</td>
<td>low</td>
<td>high</td>
<td>high</td>
<td>very high</td>
<td>low</td>
</tr>
<tr>
<td>Platform Based</td>
<td>high</td>
<td>low</td>
<td>low</td>
<td>high</td>
<td>high</td>
<td>high</td>
<td>medium</td>
</tr>
</tbody>
</table>