Topics

– Design Strategies
Standard Cell Libraries

• How do you decide on the composition of the cell library?
  – Number of inputs
  – Transistor sizing for varying capacitive loads
  – Pullup/pulldown ratio
Standard Cell Libraries

AMI5HS 0.5 micron CMOS Standard Cell

Description
AA3x is a family of 3-input gates which perform the logical AND function.

Logic Symbol

<table>
<thead>
<tr>
<th>Truth Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
</tr>
<tr>
<td>L</td>
</tr>
<tr>
<td>X</td>
</tr>
<tr>
<td>X</td>
</tr>
<tr>
<td>H</td>
</tr>
</tbody>
</table>

HDL Syntax

Verilog: `AA3x inst_name (G, A, B, C);
VHDL: `inst_name A3x port map (X, A, B, C);`
Standard Cell Libraries

<table>
<thead>
<tr>
<th>Pin Loading</th>
<th>Equivalent Loads</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pin Name</td>
<td>AA31</td>
</tr>
<tr>
<td>A</td>
<td>1.0</td>
</tr>
<tr>
<td>B</td>
<td>1.0</td>
</tr>
<tr>
<td>C</td>
<td>1.0</td>
</tr>
</tbody>
</table>

Size And Power Characteristics

<table>
<thead>
<tr>
<th>Cell</th>
<th>Equivalent Gates</th>
<th>Power Characteristics²</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Static $I_{DD}$ (T J = 85°C) (nA)</td>
<td>$EQL_{PD}$ (Eq-load)</td>
</tr>
<tr>
<td>AA31</td>
<td>1.7</td>
<td>TBD</td>
</tr>
<tr>
<td>AA32</td>
<td>2.0</td>
<td>TBD</td>
</tr>
<tr>
<td>AA34</td>
<td>4.0</td>
<td>TBD</td>
</tr>
<tr>
<td>AA36</td>
<td>5.2</td>
<td>TBD</td>
</tr>
</tbody>
</table>

1. See page 2-15 for power equations.
Standard Cell Libraries

### AMI5HS 0.5 micron CMOS Standard Cell

**Propagation Delays (ns)**

Conditions: $T_j = 25^\circ C$, $V_{DD} = 5.0V$, Typical Process

<table>
<thead>
<tr>
<th>Number of Equivalent Loads</th>
<th>1</th>
<th>3</th>
<th>6</th>
<th>10</th>
<th>13 (max)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>AA31</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>From: Any Input To: $Q_L$</td>
<td>0.20</td>
<td>0.28</td>
<td>0.36</td>
<td>0.43</td>
<td>0.55</td>
</tr>
<tr>
<td>From: Any Input To: $Q_H$</td>
<td>0.26</td>
<td>0.28</td>
<td>0.36</td>
<td>0.43</td>
<td>0.53</td>
</tr>
<tr>
<td><strong>AA32</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>From: Any Input To: $Q_L$</td>
<td>0.20</td>
<td>0.28</td>
<td>0.39</td>
<td>0.48</td>
<td>0.55</td>
</tr>
<tr>
<td>From: Any Input To: $Q_H$</td>
<td>0.26</td>
<td>0.28</td>
<td>0.42</td>
<td>0.48</td>
<td>0.61</td>
</tr>
<tr>
<td><strong>AA34</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>From: Any Input To: $Q_L$</td>
<td>0.31</td>
<td>0.35</td>
<td>0.37</td>
<td>0.48</td>
<td>0.56</td>
</tr>
<tr>
<td>From: Any Input To: $Q_H$</td>
<td>0.26</td>
<td>0.28</td>
<td>0.35</td>
<td>0.48</td>
<td>0.55</td>
</tr>
<tr>
<td><strong>AA35</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>From: Any Input To: $Q_L$</td>
<td>0.26</td>
<td>0.25</td>
<td>0.36</td>
<td>0.44</td>
<td>0.51</td>
</tr>
<tr>
<td>From: Any Input To: $Q_H$</td>
<td>0.25</td>
<td>0.25</td>
<td>0.36</td>
<td>0.44</td>
<td>0.53</td>
</tr>
</tbody>
</table>

Delay will vary with input conditions. See page 218 for interconnect estimates.
Compiled Cells

- **Standard cells**
  - Must be redesigned for every new process technology
  - Design options are limited because of discrete set of cells

- **Customized cells would provide more flexibility**
  - Automatic layout generation for design-specific requirements
Automatic Cell Generation

Initial transistor geometries

Placed transistors

Routed cell

Compacted cell

Finished cell

Courtesy Acadabra
Macrocells

- Hard Macros
  - Predetermined physical design
  - Fixed transistor and wiring locations
  - Dense layout, optimized performance and power characteristics
Cell-based Design (or standard cells)

Routing channel requirements are reduced by presence of more interconnect layers.
Macrocells

256×32 (or 8192 bit) SRAM
Generated by hard-macro module generator
Macrocells

• Soft Macros
  – Physical design is done automatically
  – Easily ported across many different technologies and processes
  – Macro cell compiler will take a functional and parameterized description and generate a netlist of standard cells
Macrocells

```
string mat = "booth";
directive (multtype = mat);
output signed [16] Z = A * B;
```
Intellectual property

• Macrocells can be acquired from third-party vendors
  – Includes appropriate compilers, debuggers, test vectors, prediction models
  – Similar to reusable software libraries
  – Examples include embedded processors, bus interfaces, DSP processors, ECC coders, MPEG codecs, etc.
“Intellectual Property”

A Protocol Processor for Wireless
Semicustom Design Flow

Design Capture

- Pre-Layout Simulation
- Logic Synthesis
- Floorplanning
- Placement
- Routing
- Tape-out

Behavioral

- HDL

Structural

- Pre-Layout Simulation

Physical

- Circuit Extraction

Design Iteration
Integrating Synthesis with Physical Design

- RTL (Timing) Constraints
- Physical Synthesis
- Macromodules
  - Fixed netlists
- Place-and-Route Optimization
- Netlist with Place-and-Route Info
- Artwork
System-on-a-Chip (SoC) Design

• Embed multiple functionalities on a single chip

• SoC is a natural result of having more and more transistors available

• Managing multiple modules becomes a design challenge
System-on-a-Chip Design

IBM CoreConnect Architecture
System-on-a-Chip Design

Philips Nexperia HDTV SoC
System-on-a-Chip Design

- Embedded applications
- Mixed-mode applications (Analog/Digital)
- Heavy software component
- May have programmable and application specific components
Array-Based Design

• Cell-based and fully custom designs require a run through the full manufacturing process
  – Can take up to several months before the first chip arrives
  – Mask generation costs can make it very expensive
    • As process technologies get better, the tendency has been to use more and more masks

• Alternative is array-based implementations
Array-Based Designs

- Pre-diffused
  - Mask-programmable
  - Gate arrays
  - Sea of gates
- Pre-wired
  - Field programmable gate arrays (FPGA)
Gate arrays

- Rows of uncommitted cells
- Routing channel

Uncommitted Cell

Committed Cell (4-input NOR)
Gate arrays

- Less compact than standard cells
- Manufacturing time savings is not as significant because the design times are the most important factor now
- Availability of FPGAs
Field Programmable Gate Arrays

- Prewired arrays
- Programming techniques
  - Write-once or fuse-based
  - Memory-based
    - Non-volatile (Flash, EEPROM, etc.)
    - Volatile (RAM)
Field Programmable Gate Arrays

• Programmable Logic Style
  – Array-based
  – Cell-based
    • Function generator
    • Look-up table

• Programming Interconnect
  – Channel-routing
  – Mesh networks
Fuse-based FPGA

antifuse polysilicon

ONO dielectric

n\(^+\) antifuse diffusion
Array-Based Programmable Logic

Programmable AND array

Programmable OR array

Fixed AND array

Fixed OR array

PLA

PROM

PAL

Indicates programmable connection

Indicates fixed connection

ECE 249 VLSI Design and Simulation
Spring 2005
Lecture 19

© John A. Chandy
Dept. of Electrical and Computer Engineering
University of Connecticut
Array-Based Programmable Logic

\[ f_0 = x_0 x_1 + \overline{x_2} \]

\[ f_1 = x_0 x_1 x_2 + \overline{x_2} + \overline{x_0 x_1} \]

\( \bullet \) : programmed node
Array-Based Programmable Logic

- Lower density than custom
  - All minterms may not be actively used
  - OK if you have large fanin

- Lower performance
  - Each node has significant capacitance

- Only implements combinational logic - no registers or flip-flops
Array-Based Programmable Logic

- Macrocell-based PAL

![Diagram of Macrocell-based PAL]

programmable AND array \((2i \times j k)\)

\(k\) macrocells

\(j\)-wide OR array

macrocell

\(i\) inputs

\(D\) \(Q\) \(OUT\)

CLK
Cell-based Programmable logic

- Multiplexor as a function generator

<table>
<thead>
<tr>
<th>Configuration</th>
<th>A</th>
<th>B</th>
<th>S</th>
<th>F=</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0</td>
<td>0</td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>0 0 1</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>0 1 1</td>
<td>Y</td>
<td></td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>0 0 X</td>
<td></td>
<td>XY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>X 0 0</td>
<td></td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Y 0 X</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Y 1 X</td>
<td></td>
<td>X</td>
<td></td>
<td>X+Y</td>
</tr>
<tr>
<td>1 0 X</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>1 0 Y</td>
<td></td>
<td></td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>1 1 1</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>
Lookup Table

<table>
<thead>
<tr>
<th>In</th>
<th>Out</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>0</td>
</tr>
<tr>
<td>01</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
</tr>
</tbody>
</table>
Lookup Table

- Configurable Logic Block (CLB) - Xilinx Virtex II
Array-based Programmable Wiring

- **Interconnect Point**
- **Programmed interconnection**
- **Input/output pin**
- **Horizontal tracks**
- **Vertical tracks**

Cell

M
Array-based Programmable wiring

- Transistor method takes more space
- Use Fuse or antifuse methods
- Write-once - can not change
Mesh Network

Switch Box

Connect Box

Interconnect Point
Mesh Network
Mesh Network

- Sometimes can not route interconnect
- Pass transistor has a voltage drop
- RAM based programmable switch matrix
- Good for local wiring
- Global wiring will have large capacitive loads
Next class

• Testing and verification

• Exam 2 next Thursday