Lecture 3: CMOS Layout, Floorplanning & other implementation styles

Mark McDermott
Electrical and Computer Engineering
The University of Texas at Austin
Layout

- Describes actual layers and geometry on the silicon substrate to implement a function

- Need to define transistors, interconnection
  - Transistor widths (for performance)
  - Spacing, interconnect widths, to reduce defects, satisfy power requirements
  - Contacts (between poly or active and metal), and vias (between metal layers)
  - Wells and their contacts (to power or ground)

- Layout of lower-level cells constrained by higher-level requirements: “floorplanning”
Chips are specified with set of masks

Minimum dimensions of masks determine transistor size (and hence speed, cost, and power)

Feature size $f = \text{distance between source and drain}$
  - Set by minimum width of polysilicon

Feature size improves 30% every 3 years or so

Normalize for feature size when describing design rules

Express rules in terms of $\lambda = f/2$
  - E.g. $\lambda = 0.3 \, \mu m$ in 0.6 $\mu m$ process
CMOS Inverter Layout

Note: the N- and P- wells are not shown here
Another CMOS Inverter Layout
CMOS Inverter with Wider Transistors
Buffer with Two Inverters
Buffer with Stacked Inverters
Efficient Buffer with Stacked Inverters
Simplified Layout of NAND Gate
“Stick” Diagram for NAND Gate

- Identifies actual layers, can be annotated with transistor sizes
Simplified Design Rules

- Conservative rules to get you started
Inverter Layout

- Transistor dimensions specified as Width / Length
  - Minimum size $4\lambda / 2\lambda$, sometimes called 1 unit or standard pitch
  - In $f = 0.6 \, \mu m$ process, this is $1.2 \, \mu m$ wide, $0.6 \, \mu m$ long
Typical Layout Densities

- Typical numbers of high-quality layout
- Derate by 2 for class projects to allow routing and some sloppy layout.
- Allocate space for big wiring channels

<table>
<thead>
<tr>
<th>Element</th>
<th>Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random logic (2 metal layers)</td>
<td>1000-1500 $\lambda^2$ / transistor</td>
</tr>
</tbody>
</table>
| Datapath                 | 250 – 750 $\lambda^2$ / transistor  
Or 6 WL + 360 $\lambda^2$ / transistor |
| SRAM                     | 1000 $\lambda^2$ / bit       |
| DRAM                     | 100 $\lambda^2$ / bit        |
| ROM                      | 100 $\lambda^2$ / bit        |
Area Calculation Example: NAND3

- Horizontal N-diffusion and p-diffusion strips
- Vertical polysilicon gates
- Metal1 $V_{DD}$ rail at top
- Metal1 GND rail at bottom
- 32 $\lambda$ by 40 $\lambda$
Cell Flipping

- Flip every other cell
- Cells share VDD & GND
- Cells share N-WELL and substrate connections
- Reduces cell height
  - Measure contact center to contact center
A *wiring track* is the space required for a wire
- \(4\ \lambda\) width, \(4\ \lambda\) spacing from neighbor = \(8\ \lambda\) pitch

Transistors also consume one wiring track
Well spacing

- **Wells must surround transistors by 6 $\lambda$**
  - Implies 12 $\lambda$ between opposite transistor flavors
  - Leaves room for one wire track
Area Estimation

- Estimate area by counting wiring tracks
  - Multiply by 8 to express in $\lambda$
Example: O3AI

- Sketch a stick diagram for O3AI and estimate area

\[ Y = (A + B + C) \land D \]
Example: O3AI

- Sketch a stick diagram for O3AI and estimate area

\[ Y = (A + B + C) \times D \]
Example: O3AI

- Sketch a stick diagram for O3AI and estimate area

\[ Y = (A + B + C) \times D \]
The MOSIS Scalable CMOS Rules

- λ-based rules
- Designs using these rules are fabricated by a variety of companies
- Multiple designs are put on a single die
  - Each chip wired to a particular design
- Support for submicron digital CMOS, analog (buried poly layer for capacitor), micromachines, etc.

www.mosis.org/Technical/Designrules/scmos/
Floorplanning
Floorplanning 101

- **Determine block sizes**
  - Function of SC pitch, Cell Placement, RLM, SC/SDP, Custom/Memory Block Sizing and Block Routing Overhead (Signals, Clocking, Power)
Floorplanning 101

- **Determine block sizes**
  - Function of SC pitch, Cell Placement, RLM, SC/SDP, Custom/Memory Block Sizing and Block Routing Overhead (Signals, Clocking, Power)

- **Determine core size**
  - Function of #Blocks, Block sizes, Block Aspect Ratios, Global Routing Overhead (Signals, Clocking, Power)

- **Determine I/O ring size**
  - Function of the number of I/O, Number
Floorplanning

- **How do you estimate block areas?**
  - Begin with block diagram
  - Each block has
    - Inputs
    - Outputs
    - Function (draw schematic)
    - Type: array, datapath, random logic

- **Estimation depends on type of logic**
  - RLM: Random Logic Macro
  - Datapath
  - Array
Area Estimation

- **Arrays:**
  - Layout basic cell
  - Calculate core area from # of cells
  - Allow area for decoders, column circuitry

- **Datapaths**
  - Sketch slice plan
  - Count area of cells from cell library
  - Ensure wiring is possible

- **Random logic**
  - Compare complexity do a design you have done
Metal Planning

- **Metal layer, width, spacing and shielding are negotiable**
  - “Negotiable” means you have to plead your case to the integration leader
  - All of these impose a physical constraint for layout

- **Typical 8 layer metal layer allocation**
  - M1, M2: Local routing (standard cell)
  - M3, M4, M5, M6: Data and control
  - M7, M8: Power, Ground, Clock, Reset, etc
  - Assume HVH routing:
    - Metal-1: Horizontal
    - Metal-2: Vertical
    - Metal-3: Horizontal
    - Metal-4: Vertical
    - ....

- Use standard 'HALO' cells to make the resulting 'floor-plannable' objects 'snap' to the desired power and routing grids.
  - Added to the boundary of all custom layouts (as well as synthesized blocks).
Chip & Block Level Clock Routing

- Watch out for the clock, it’s your most critical net
- Make sure the physical design treats it accordingly
- Help reduce clock power by eliminating unnecessary load
- Make sure the clock net has enough via coverage
- Use a combination of Global (Chip) and Block Level Clock distribution
Chip level power routing

- Power busses are a combination of rings and/or grids.
  - Rings are generally in the I/O ring.
  - Grids are used at the chip and block level.
  - Grid pitch is set by horizontal and vertical routing resource requirements.

- Special consideration needs to be taken for multiple power domains.
  - There can be any number of power domains depending on the system architecture.
  - Analog blocks require isolation rings.
  - Interfaces between blocks require level shifters.
Eye candy: Floorplan examples
Apple A8 SOC (for iPhone)
Apple A8X SOC (for iPad)
Flip chip power mesh for AMD Jaguar
SPARC Multicore Processor
Xilinx XC2C32A CPLD
Analog Devices LNA
Implementation Techniques
Implementation Choices

Digital Circuit Implementation Approaches

- Custom
  - Cell-based
    - Standard Cells
    - Compiled Cells
  - Semicustom
    - Macro Cells
- Array-based
  - Pre-diffused (Gate Arrays)
  - Pre-wired (FPGA's)
Path from RTL to structural netlist

**RTL Database**

- **LEC**
  - LEC proves equivalence of RTL and Schematics

- **RLM**
  - Use RLM library
  - Create with logic synthesis
  - May “tweak” output by hand

- **Structured Datapath**
  - Use any existing cell from the library
  - Create with text editor or schematic capture

- **CUSTOM**
  - Create new layout cells
  - Create new schematics
  - Use new layout cells and schematic in “Datapath” flow to the left

**Schematic or Gate level RTL**

- “I want to design control logic…”
- “I want to design datapath logic…”
- “I want to design an array, complex dynamic gate, etc…”

**Automatically Generated Low-Level Netlist**

**Cell Library**

8/26/18
The Custom Approach

Intel 4004

© Digital Integrated Circuits 2nd

Courtesy Intel
Transition to Automation and Regular Structures

Intel 4004 (‘71)

Intel 8080

Intel 8085

Intel 8286

Intel 8486

Courtesy Intel

© Digital Integrated Circuits2nd

8/26/18
Cell-based Design (or standard cells)

Routing channel requirements are reduced by presence of more interconnect layers.
Standard Cell — Example
Standard Cell – The New Generation

Cell-structure hidden under interconnect layers

© Digital Integrated Circuits

8/26/18
Standard Cell - Example

3-input NAND cell
(from ST Microelectronics):

C = Load capacitance
T = input rise/fall time

<table>
<thead>
<tr>
<th>Path</th>
<th>1.2V - 125°C</th>
<th>1.6V - 40°C</th>
</tr>
</thead>
<tbody>
<tr>
<td>In1→t_{pLH}</td>
<td>0.073+7.98C+0.317T</td>
<td>0.020+2.73C+0.253T</td>
</tr>
<tr>
<td>In1→t_{pHL}</td>
<td>0.069+8.43C+0.364T</td>
<td>0.018+2.14C+0.292T</td>
</tr>
<tr>
<td>In2→t_{pLH}</td>
<td>0.101+7.97C+0.318T</td>
<td>0.026+2.38C+0.255T</td>
</tr>
<tr>
<td>In2→t_{pHL}</td>
<td>0.097+8.42C+0.325T</td>
<td>0.023+2.14C+0.269T</td>
</tr>
<tr>
<td>In3→t_{pLH}</td>
<td>0.120+8.00C+0.318T</td>
<td>0.031+2.37C+0.258T</td>
</tr>
<tr>
<td>In3→t_{pHL}</td>
<td>0.110+8.41C+0.280T</td>
<td>0.027+2.15C+0.223T</td>
</tr>
</tbody>
</table>

© Digital Integrated Circuits

8/26/18 VLSI-1 Class Notes 47
Automatic Cell Generation

Initial transistor geometries
Placed transistors
Routed cell
Compacted cell
Finished cell

© Digital Integrated Circuits 2nd

Courtesy Cadabra
MacroModules

$256 \times 32$ (or 8192 bit) SRAM
Generated by hard-macro module generator

© Digital Integrated Circuits$^{2\text{nd}}$
“Soft” MacroModules

string mat = “booth”;
directive (multtype = mat);
output signed [16] Z = A * B;

© Digital Integrated Circuits2nd
“Intellectual Property” (IP) Cores

A Protocol Processor for Wireless

© Digital Integrated Circuits 2nd
Semicustom Design Flow

Design Iteration

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

Design Capture

Behavioral

Structural

Physical

© Digital Integrated Circuits 2nd

8/26/18
The “Design Closure” Problem

Iterative Removal of Timing Violations (white lines)

© Digital Integrated Circuits 2nd

Courtesy Synopsys
Integrating Synthesis with Physical Design

RTL (Timing) Constraints

Physical Synthesis

Macromodules
Fixed netlists

Place-and-Route Optimization

Netlist with Place-and-Route Info

Artwork

© Digital Integrated Circuits 2nd
Late-Binding Implementation

Array-based

- Pre-diffused (Gate Arrays)
- Pre-wired (FPGA's)

© Digital Integrated Circuits2nd
Gate Array — Sea-of-gates

Uncommitted Cell

Committed Cell (4-input NOR)

© Digital Integrated Circuits 2nd
Sea-of-gate Primitive Cells

Using oxide-isolation

Using gate-isolation

© Digital Integrated Circuits 2nd
Example: Base Cell of Gate-Isolated GA

From Smith97
Example: Flip-Flop in Gate-Isolated GA

© Digital Integrated Circuits 2nd From Smith97
Sea-of-gates

Random Logic

Memory Subsystem

LSI Logic LEA300K (0.6 µm CMOS)

© Digital Integrated Circuits 2nd

Courtesy LSI Logic
Prewired Arrays

Classification of prewired arrays (or field-programmable devices):

- Based on Programming Technique
  - Fuse-based (program-once)
  - Non-volatile EPROM based
  - RAM based
- Programmable Logic Style
  - Array-Based
  - Look-up Table
- Programmable Interconnect Style
  - Channel-routing
  - Mesh networks