Overview of VLSI Design
Background

- VLSI is a relatively new field
  - Started with SSI's in early 60's (a few BJTs and resistors on a chip)
  - Today's chips have millions of simple MOS transistors and perform complex functions
Why VLSI?

- Building complex electronic circuits using discrete components are difficult and expensive
  - Cost depends on # of devices
- Integrated circuits solved much of the problems
  - Print many tiny circuits on a flat surface - “easy” as taking pictures
  - Cost depends on die size
Moore’s Law

- Resolution of the printing process has improved exponentially
  - Feature size has shrunk by 0.7 times every 3 years
  - Intel’s Gordon Moore in early 80’s predicted that this trend would continue
- Cost of printing process has grown modestly
- Thus, cost per function has dropped exponentially
Moore’s Law (continued)

- At each new generation (every 3 years), the cost per function have dropped by a factor of 2
- Shrinking an existing chip makes it cheaper!
But...

- Cost of manufacturing ICs have remained flat but design cost has not.
  - Design productivity has not improved at the same rate as the complexity of the chips.
  - So the cost of chip design has grown exponentially with the complexity of the ICs.
  - Integrated circuits are attractive in terms of manufacturing cost but not in terms of design cost and risk.
What Fits On A Chip Today?

- State of the art IC
  - 18mm on each side (324 mm²)
  - 0.25um drawn gate length
  - 1um wire pitch (18,000 wire pitches)
  - 4-5 level metal
Technology Scaling

- Number of ‘grids’ per chip quadruples every 3 years
  - More functionality per chip
  - More difficult to design
  - What to do with all the space?
  - How to insure it works?
VLSI Design

- Manage complexity
  - Simplify
  - Abstract
  - Constrain
- Understand underlying technology
  - Determine what abstractions and constraints are needed
  - Determine efficient solutions (in terms of design time, area, power, and performance)
Abstraction

- Digital abstraction
  - Signals are 1 or 0

- Switch abstraction
  - MOSFETs are simple switches

- Gate abstraction
  - Unidirectional element
  - Separable timing
Design Discipline

- Constrain design space to simplify design process
  - Tradeoff between design complexity and performance
- Partition problem using hierarchy
Design Levels

- Specification
- Architecture
- Logic
- Circuit
- Device
  - Behavior of individual circuit elements
- Layout
  - Geometry used to define and connect circuit elements
- Process
DESIGN ABSTRACTIONS
What Are On ICs?

- Conducting layers which form wires
  - Many layers of wires (4-5 metal layers) which has electrical properties such as resistance and capacitance
  - Contacts and insulators between layers
- Transistors ("free" things that fit under the wires)
  - MOS transistors only for this class (voltage controlled switch)
Physical Topology

- Transistors fabricated on silicon surface
- Wires on layers of metal separated by insulators
  - Wires connect transistors
Layout

- Layout = fabrication specification
- Set of drawings (one for each layer)
  - Design rules for reliable manufacturing
  - Each layer represented by a unique color
  - Use CAD tools
Layout Example

Logical (MAGIC created)

Physical
Stick Diagram

- Simplified version of layout
  - Abstract the layout so that wires are just lines
    - No need to worry about width or spacing
  - Spacing need not be the same
    - Although it would be better to keep spacing the same for size estimation
  - Good starting point before the layout
Wiring Layers

- Wiring layers represented by different colors
  - diff: green/yellow
  - poly: red
  - metal1: blue
  - metal2: orange
  - metal3: purple

- Wires on the same layer always connect, if they touch. No way to jump wires without changing layers

- Need contacts to connect wires on different layers
Transistor

- Formed when poly (red) crosses diffusion (green or yellow)

No connection

Connection

connected  transistor
Transistor

- Voltage on gate (poly connection) controls current that flows between source and drain (diffusion terminals)
Digital Abstraction

- Rather than worrying about precise voltage levels, guarantee that voltages fall within two regions
  - One for logic ‘0’ and the other for logic ‘1’
  - Only need to compute outputs for inputs in the allowable range
    - Model transistor as on or off
  - Need to insure that output always in the allowable range (valid digital output)
    - Also want to restore voltage level
    - Allowable output range smaller than allowable input range
Digital Abstraction

- Divide voltage into discrete regions
  - Logic ‘0’ and logic ‘1’
  - X (between 0 and 1)
  - Out of range (may damage device)

- Each logic gate restores the signal
  - Noise not accumulative
  - Output range narrower than input range
  - Noise margin ($V_{OH} - V_{IH}$)
Simple MOSFET Model

- **3 terminal device**
  - Source, drain
    - Two ends of conduction path
  - Gate
    - Controls conduction
  - Operation
    - Conducts when gate high
    - Open when gate low
Transistor Examples

1
0
on

0
off

1
1
0
on

0
off

1

1
off*

1

1
off*
Switch Networks

- Structure defines logic functions
  - ‘OR’ constructed by parallel switches
  - ‘AND’ constructed by series switches

- Define function as input conditions that connect two terminals of the network
  - Func = true means terminals connected
  - Func = false means terminals not connected
General Switch Networks

![Diagram of General Switch Networks](image-url)
Switch Logic

- Use switch networks to build simple logic
  - Connects one of several inputs to output
  - Primary output must be connected to one of the inputs
  - Inputs must not be connected together
- Assume for now that both true and complement values of inputs are available
Multiplexor

- Selects one of the inputs to the output
XOR

- Using MUX structure
Parity

- $Z = A \text{ XOR } B \text{ XOR } C \text{ XOR } \ldots$
- Use iterative structure

- But need both true and complemented output
Parity Stick Diagram

- A single stage

- 4-input parity function
Inverters

To build an inverter with switch networks,

- Connect output to ground when input high - use a switch
- Connect output to $V_{dd}$ when input low - use another switch

How do we build this?
CMOS Inverters

- Use pMOS inverter
- Connects source/drain when gate low