Cost and Integration Drivers

- Moore’s Law is about cost
- Increased integration, decreased cost → more possibilities for semiconductor-based products
- Pentium 4 die shot:

2.2cm

HW Q1: Estimate (a) the number of SRAM bitcells per square millimeter, and (b) the number of logic gates per square millimeter, in a 90nm CMOS process. Explain how you obtained your answers.
MOS Transistor Scaling (1974 to present)

S = 0.7
[0.5x per 2 nodes]

Source: 2001 ITRS - Exec. Summary, ORTC Figure
Half Pitch (= Pitch/2) Definition

(Typical DRAM)

(Typical MPU/ASIC)

Source: 2001 ITRS - Exec. Summary, ORTC Figure
Sense of Scale (Scaling)

- What fits on a VLSI Chip today?
- State of the art logic chip
  - 20mm on a side (400mm²)
  - 0.13μm drawn gate length
  - 0.5μm wire pitch
  - 8-level metal
- For comparison
  - 32b RISC processor
    - 8K λ x 16Kλ
  - SRAM
    - about 32λ x 32λ per bit
    - 8K x 16K is 128Kb, 16KB
  - DRAM
    - 8λ x 16λ per bit
    - 8K x16K is 1Mb, 128KB

Slide courtesy of Ken Yang, UCLA
Sub-Wavelength Optical Lithography

What are implications of this picture?
...Complexity of Photomasks

How many wafers, on average, are printed with a mask set?
Silicon Complexity Challenges

- Silicon Complexity = impact of process scaling, new materials, new device/interconnect architectures
- Non-ideal scaling (leakage, power management, circuit/device innovation, current delivery)
- Coupled high-frequency devices and interconnects (signal integrity analysis and management)
- Manufacturing variability (library characterization, analog and digital circuit performance, error-tolerant design, layout reusability, static performance verification methodology/tools)
- Scaling of global interconnect performance (communication, synchronization)
- Decreased reliability (SEU, gate insulator tunneling and breakdown, joule heating and electromigration)
- Complexity of manufacturing handoff (reticle enhancement and mask writing/inspection flow, manufacturing NRE cost)
  - If you don’t know a term, ask…
System Complexity Challenges

- System Complexity = exponentially increasing transistor counts, with increased diversity (mixed-signal SOC, …)

- Reuse (hierarchical design support, heterogeneous SOC integration, reuse of verification/test/IP)

- Verification and test (specification capture, design for verifiability, verification reuse, system-level and software verification, AMS self-test, noise-delay fault tests, test reuse)

- Cost-driven design optimization (manufacturing cost modeling and analysis, quality metrics, die-package co-optimization, …)

- Embedded software design (platform-based system design methodologies, software verification/analysis, codesign w/HW)

- Reliable implementation platforms (predictable chip implementation onto multiple fabrics, higher-level handoff)

- Design process management (team size / geog distribution, data mgmt, collaborative design, process improvement)
Required Performance for Multi-Media Processing

GOPS: Giga Operations Per Second
In a PDA...

- **Reference Design**: personal digital assistant (PDA)

- Composed of CPU, DSP, peripheral I/O, and memory
...Implemented With an SoC

0.18um / 400MHz / 470mW (typical)

MM Application
MP3
JPEG
Simple Moving Picture

Specification
Available Time
6-10Hr

Peripheral Area
4 – 48MHz

Data Transfer Area
100MHz

Processor Area
6.5MTrs.
Max 400MHz

- If the PDA must have 200h standby time with a 120g battery… ?

ECE 260B – CSE 241A Intro and ASIC Flow .12
Andrew B. Kahng, UCSD
Class Objectives

- Learn about ASIC implementation flow: Verilog → GDSII
  - Semi-custom implementation of CMOS digital circuits, and optimization with respect to different constraints: area, speed, power, reliability, cost
  - Understand impact of constraints, tradeoffs, technology scaling
  - Get some feel for each phase of the implementation flow

- Learn about building blocks: wires, gates, memories

- Prepare for future design experiences
  - Get some feel for industry-standard design tools, libraries
    - Will mostly use Cadence BuildGates and SOC Encounter, and Artisan TSMC 0.18/0.13um libraries
  - Synthesize small cores from RTL into GDSII
Design Levels

- **Specification**
  - what the system (or component) is supposed to do

- **Architecture**
  - high-level design of component
    - state defined
    - logic partitioned into major blocks

- **Logic**
  - gates, flip-flops, and the connections between them

- **Circuit**
  - transistor circuits to realize logic elements

- **Device**
  - behavior of individual circuit elements

- **Layout**
  - geometry used to define and connect circuit elements

- **Process**
  - steps used to define circuit elements

Can describe design at many different levels of abstraction
Abstractions and Disciplines

- Digital abstraction
  - signals are 1 or 0

- Switch abstraction
  - MOSFETs as simple switches

- Gate abstraction
  - Unidirectional elements
  - Separable timing

- Synchronous abstraction
  - Race free logic
  - Function does not depend on timing

- Constrain the design space to simplify the design process
  - Balance between design complexity and performance
  - E.g., standard-cell methodology

- Orthogonalize concerns
  - Architecture and implementation
  - Logic and timing
  - Logic and embedding

- Partition the problem (hierarchy)
  - Module is a box with pins
  - Apply recursively
Design Procedure and Tools

- Concept
  - divider

- Architecture
  - subtract/compare

- Logical Implementation
  - \( ab+bc+ac \)
  - xor

- Circuit Implementation
  - transistors

- Physical layout + Verify
  - mask layers (rectangles)

- C-modeling

- Behavior modeling
  - Verilog or VHDL

- Logic synthesis
  - SNPS DC, CDN BG, …
  - Verification of synthesis
    - Static timing analysis

- Place and route
  - SNPS Astro, CDN SOCE, …
  - Verification of layout
    - Dynamic timing analysis
Design Methodologies (+ business models)

- Full-Custom (high effort, leading-edge performance, high-volume)
- Semi-Custom (strong infrastructure, economical in lower volumes)
  - ASIC (Application-Specific Integrated Circuit)
  - COT (Customer-Owned Tooling)
  - ASIC vs. COT: “Who pays for the scrap?”
- FPGA
- System-on-Chip (System-in-Package)
  - Larger components, often from outside of design team
- Special
  - Analog (custom layout, I/Os and sense amps)
  - Mixed-Signal / RF (unique to each process, no scaling)
Flow

- Wire Model
  - 3-D RLC Modeling Tool
    - $\rho, \sigma, \mu$
    - Layers
    - Layout rules

- Standard Cell Library
- Device model
- Cell Characterization
- Schematic Entry
- Layout Entry

- Parasitic Extraction Library
- Synthesis Library (Timing/Power/Area)
- Place & Route Library (Ports)

- C-Model
- Verilog Behavioral Model
  - Functional
- Structural Model
  - Verilog Structural RTL
  - Functional
  - Static Timing
- Block Layout
  - Floorplan
  - P & R
- Global Layout
  - Floorplan
  - DRC/ERC/LVS
  - Static/Dynamic Timing w/extract
  - Power/Area
  - Scan/Testability
  - Clock Routing/Analysis
Traditional Taxonomy

Front End

Behavioral Level Design

Logic Design and Simulation

Logic Partitioning

Die Planning

Simulation

Floorplanning

Design Verification

Timing Verification

Test Generation

Back End

IO Pad Placement

Power/Ground Stripes, Rings Routing

Global Placement

Detail Placement

Clock Tree Synthesis and Routing

Global Routing

Detail Routing

Extraction and Delay Calc. Timing Verification

LVS

DRC

ERC
Another Version (Back-End Flow)

- Architectural optimization (timing)
- Inter-group buses, bandwidth
- Clock, SI, test; validation

- Floorplanning and custom WLM
- Power distribution (Internal, I/O)
- I/O driver, padring design
- Board-level timing, SI

- Row definitions
- Placement of cells
- Congestion analysis

- Placement-based re-synthesis
- Noise minimization, isolation
- Clock distribution

- Full routing
- Scan stitching, re-ordering

- Full RC back-annotation
- Hierarchical timing, electrical and SI analysis and IPO/ECO
Generic Flow Steps

- Library preparation
  - Library data preparation
  - Design data preparation

- Logic design
  - Specification to RTL
  - RTL simulation
  - Hierarchical floorplanning
  - Synthesis
  - Formal verification
  - Gate level simulation
  - Static timing analysis

- Physical design
  - Physical floorplanning
  - Place and route
  - RC extraction
  - Formal verification
  - Physical verification
  - Release to manufacturing

- Design for test

- Engineering change order
Library and Design Data

- Models and technology data required to execute the design flow

- Power, timing: ALF, DCL, OLA, .lib, STAMP

- Layout: LEF, DEF, GDSII

- Delays and path timing, parasitics: SDF, GCF, SDC, DSPF, RSPF, SPEF, SPICE

- Layout rules: Dracula, Calibre “deck”
RTL Simulation

- RTL code, written in Verilog, VHDL or a combination of both, is simulated to verify functional correctness
- Testbenches apply input stimulus to the design
- Several methods are used to verify the outputs
  - Self-checking testbenches automatically verify output correctness and report mismatches
  - Results can be stored in a file and compared to previous results
  - Waveform displays can be used to interactively verify the outputs
Floorplanning

- Reconcile logical and physical hierarchies
- Cells that are interconnected want to be close together
  - Take advantage of RTL hierarchy
  - Generate a physical hierarchy
  - RTL hierarchy = best physical hierarchy?
- Place big blocks on chip (memories)
- Allow space for power, clock, major buses
- Give placement some initial clues to reduce complexity
Logic Synthesis

- Conversion of RTL to gate-level netlist
  - Targeted to a foundry-specific library
  - Can be performed hierarchically (block by block)

- Timing-driven
  - Clock information
  - Primary input arrival times, primary output required times
  - Input driving cells, output loading
  - False paths, multi-cycle paths

- Interconnect delay may be calculated based on a “wireload model” which uses fanout to estimate delay

- Clock parameters (insertion delay, skew, jitter, etc.) are assumed to be attainable later in place and route
Verification of Synthesis Result

- Formal verification
  - RTL description and gate level netlist are compared to verify functional equivalence, thereby verifying the synthesis results
  - Emerging technology that supplements the more traditional gate-level simulation approach
  - FV also performed after place-and-route (if gate netlist changes)

- Gate-level simulation
  - Covers both functionality and timing
  - Correctness is only as good as the test vectors used
  - Especially critical for non-synchronous designs, verification of false path and multi-cycle path constraints
  - Cell timing is included in the simulation models and interconnect delay is passed from the synthesis run
  - Worst case PVT conditions are used to analyze for setup violations, and best case PVT conditions are used to analyze for hold violations
    - PVT = Process, Voltage, Temperature
Static Timing Analysis

- Verifies that design operates at desired frequency
  - Implicitly assumes correct timing constraints (!), e.g., boundary conditions

- Timing constraints are similar to those used by logic synthesis

- Verifies setup and hold times at FF inputs; can also check timing from and to PI’s and PO’s; can also check point-to-point delay values (with blocking of pins, etc.)

- As with gate-level simulation, both best- and worst-case analysis is performed

- Typically performed on full-chip (not block) basis
  - May require modified constraints for inter-block issues: multiple clock domains, multi-cycle paths, etc.

- For compatibility with timing-driven layout flow, helps to have simple / single set of constraints
  - Other issues: incremental analysis, …
Physical Floorplanning

- Defines the basic chip layout architecture
  - Define the standard cell rows and I/O placement locations
  - Place RAMs and other macros
  - Define power distribution structures such as rings and stripes
- Often bundled within the same cockpit as the place and route tool
- Rules of thumb for cell density are used to initially calculate design size
Place and Route

- Automatically place the standard cells
- Generate clock trees
- Add any remaining power bus connections
- Route clock lines
- Route signal interconnects
- Design rule checks on the routes and cell placements
- Timing driven tools
  - Require timing constraints and analysis algorithms similar to those used during the static timing analysis step
RC(L) Extraction

- Calculate resistance and capacitance (and inductance) of interconnects
  - Based on placement of cells
  - Routing segments

- Calculate capacitive (inductive) effects of adjacent segments
  - Extract capacitance between metal segments

- RC(L) data transferred back to
  - Static timing analysis (back annotation)
  - Gate level simulation
  - Replaces wire load model used in synthesis

- Drive delay calculation, signal integrity analysis (crosstalk, other noise), static timing

- Q: How do parasitics and noise affect performance?
Physical Verification

- **DRC – Design Rule Check**
  - Polygon/Layer spacing rules
  - Verifies the design rules (DRC)

- **LVS – Layout Versus Schematic**
  - Verifies that layout and netlist are equivalent at the transistor level

- **Antenna (and other Electrical Rule Check)**
  - Manufacturing check for long nets
  - Net can accumulate charge during plasma etch and damage gate oxide

- **GDSII (Stream Format)**
  - Final merge of layout, routing and placement data for mask production
Release to Manufacturing

- Final edits to the layout are made
- Metal fill and metal stress relief rules are checked
- Manufacturing information such as scribe lanes, seal rings, mask shop data, part numbers, logos and pin 1 identification information for assembly are also added
- DRC and LVS are run to verify the correctness of the modified database
- ‘Tapeout’ documentation is prepared prior to release of the GDSII to the foundry
- Pad location information is prepared, typically in a spreadsheet
- Cadence’s Virtuoso is used for custom-manual edits of the mask layers

Manufacturing steps
- generation of masks
- silicon processing
- wafer testing
- assembly and packaging
- manufacturing test
Multiple design files are converged into one efficient Data Model
- Disk accesses are eliminated in critical methodology loops
- Verification of function, performance, testability and other design criteria all move to earlier, higher levels of abstraction followed by
  - Equivalence checking
  - Assertion-driven design optimizations
- Industry standard interfaces for data access and control
- Incremental modular tools for optimization and analysis