6.371 Intro to VLSI Systems
Fall 2002

Lecturers: Krste Asanovic, Chris Terman
Website: http://6371.lcs.mit.edu
Modern systems engineering

Personal Computer: Hardware & Software

Circuit Board: ≈8 / system
1-2G devices

Integrated Circuit: ≈8-16 / PCB
.25M-16M devices

Module: ≈8-16 / IC
100K devices

MOSFET

Scheme for representing information

Gate: ≈2-16 / Cell
8 devices

Cell: ≈1K-10K / Module
16-64 devices
So, what’s 6.371 all about?

You’ll get a top-down tour of how integrated circuits are engineered. We’ll talk about

♦ how you tackle the problem of designing circuits with millions of gates – and have everything work the first time!

♦ how architectural and implementation choices are affected by the target technology (in our case, CMOS)

♦ various design and layout techniques for creating combinational and sequential circuits, datapaths, memories, buffers, regular logic structures, …

♦ computer-aided design (CAD) tools – letting programs do the heavy lifting.

The fun part: you’ll get to try it yourself in the labs!
What's expected of you

Homework
20% of final grade
Design exercises to be done in the “lab.” Readings from Weste & Eshraghian (see course website for how to order the book).

Project
40% of final grade
Specify, architect, design, layout and verify a modest-sized CMOS circuit. Prepare a Web-page description and give a brief presentation the last week of class. Work as part of a 2- or 3-person team.

Quizzes
40% of final grade
Two 80-minute in-class, closed-book quizzes (10/11, 11/15). Meant to be duck soup if you’ve been coming to lectures and doing the homework.
What is a VLSI circuit?

**VERY LARGE SCALE INTEGRATED CIRCUIT**

Technique where many circuit components and the wiring that connects them are manufactured simultaneously into a compact, reliable and inexpensive chip.

Early (circa 1977) characterization of circuit “size” before people realized that the number of components per chip was quadrupling every 24 months (aka Moore’s Law)!
Dawn of the transistor

Bell Labs lays the groundwork:

1940: Russel Ohl develops PN junction which produces 0.5V when exposed to light.

1945: Bell sets up lab in the hopes of developing “solid state” components to replace existing electromechanical systems. William Schockley, John Bardeen, Walter Brattain: all solid-state physicists. Focus on Si and Ge.

1947: Bardeen and Brattain create point-contact transistor w/ two PN junctions. Gain = 18.

Announced in July 1948. But treated as a novelty until 1951 invention of junction transistor. Bell Labs willing to license the rights to the transistor to any company for a royalty (which was waived for hearing aid companies as a gesture to Alex. G. Bell). Transistor was good: smaller, faster, more reliable and economical but this is only half the story since the circuits, albeit smaller, were still constructed in much the same way.
Dawn of the transistor (II)

1951: Shockley develops junction transistor which can be manufactured in quantity.

1952: GWA Dummer forecasts “solid block [with] layers of insulating, conducting and amplifying materials”

1954: The first transistor radio! Also, TI makes first silicon transistor (price $2.50)

1956: Bardeen, Shockley, Brattain receive Nobel Prize.

(U.S. Patents #2,502,488, #2,524,035)
Early Integration

Jack Kilby was denied entry to MIT because of poor high school grades (went to U of I). Kilby worked on miniaturized components during the war and experimented with photolithography. Went to 1952 Bell Labs transistor course.

High labor costs at TI got Kilby thinking about “solid circuits” over the July 1959 plant closing. Built phase-shift oscillator and it worked on 9/12/59. By the end of the year, he had constructed several examples, including the flip-flop shown in the patent drawing above. Components are connected by hand-soldered wires and isolated by “shaping” and PN diodes used as resistors.

In December 2000, Kilby was awarded the Nobel Prize in physics for this work.
Making it real...

Robert Noyce experimented in the late 40’s with transistors while a physics major at college (his prof was friends with Bardeen at Bell and so had early access to transistors). He came to MIT where “much to his surprise, few people had even heard about the transistor.” After getting his PhD in 1953, he worked in industry, finally arriving at Mountain View, CA and Shockley Semiconductor Labs in 1955.

In 1957, Noyce left Shockley’s lab (Schockley wasn’t the best of managers) to form Fairchild Semiconductor with Jean Hoerni. Gordon Moore is another founder.

In early 1958, Hoerni invents technique for diffusing impurities into the silicon to build planar transistors and then using a SiO<sub>2</sub> insulator. In spring of 1959, Kurt Lehovec at Sprague Elec. Co. here in North Adams, MA invents isolation technique using back-to-back pn junctions.

In mid 1959, Noyce develops first true IC using planar transistors, back-to-back pn junctions for isolation, diode-isolated silicon resistors and SiO<sub>2</sub> insulation wired using his innovation: using metal deposited by evaporation through a mask to form the interconnect -- keeping the IC flat and easy to build.
1960's: era of integration (social and electrical!)

1961: TI and Fairchild introduced the first logic IC's (cost ~$50 in quantity!). This is a dual flip-flop with 4 transistors.

1963: Densities and yields are improving. This circuit has four flip flops.

1966: Robert Dennard invents 1-T DRAM at IBM TJ Watson Research Center.

1967: Fairchild markets this semi-custom chip. Transistors (organized in columns) could be easily rewired using a two-layer interconnect to create different circuits. This circuit contains ~150 logic gates. Masks are laid-out, cut and checked by hand... beginnings of a design flow but no computer automation.
1968: Noyce and Moore leave Fairchild and found Intel. No business plan, just a promise to specialize in memory chips. They and Art Rock raise $2.5MM in two days and move to Santa Clara. By 1971 Intel had 500 employees; by 1983 it had 21,500 employees and $1.1B in sales.

In 1970, making good on its promise to its investors Intel (Joel Karp, Les Vadasz, John Reed) starts selling a 1K bit PMOS RAM, the 1103. It was a bear to interface to, but its density and cost make it the only game in town. Core memory dies…

In 1971 Intel introduces the first microprocessor, designed by Ted Hoff. The 4004 had 4-bit buses and a clock rate of 108KHz. It had 2300 transistors and was built in a 10u process. It never captured much interest in the market and was soon eclipsed by its more capable brethren.

"Moore Noyce" was trademarked for a hotel chain!
Exponential Growth

Introduced in 1972, the 8008 had 3,500 PMOS transistors supporting a byte-wide data path. Despite its limitations, the 8008 was the first microprocessor capable of playing the role of computer CPU as demonstrated on the cover of the July ‘74 issue of Radio-Electronics.

Last, but not least, on our tour is the 8080. Introduced in 1974, the 8080 had 6,000 NMOS transistors fab’ed in a 6u process. The clock rate was 2Mhz, more than enough to ignite the personal computer industry. At least Paul Allen and his partner thought so when they wrote a BASIC interpreter for the 8080 in 1975. They would later collaborate in another, more profitable, venture...
Moore's Law...


Shown with approximate relative sizes

- Intel 8080A, 1974
  - 3Mhz, 6K transistors, 6u

- Intel 8086, 1978, 33mm²
  - 10Mhz, 29K transistors, 3u

- Intel 80286, 1982, 47mm²
  - 12.5Mhz, 134K transistors, 1.5u

- Intel 386DX, 1985, 43mm²
  - 33Mhz, 275K transistors, 1u

- Intel 486, 1989, 81mm²
  - 50Mhz, 1.2M transistors, .8u

  - 66Mhz, 3.1M transistors, .8u/.6u/.35u

- Intel Pentium II, 1997, 203mm²/104mm²
  - 300/333Mhz, 7.5M transistors, .35u/.25u
The Modern Era

• Multiple functional blocks (some general purpose) stitched together at the top level to make video encoder/decoder. Lots of modeling at architectural level to ensure that functional goals could be met.

• Different teams worked on the different blocks. Note that each block is itself composed of sub-blocks, and so on for many levels of hierarchy. Lots of iteration and reuse...

• Many different architectural choices: RISC controller, SIMD compute engine, special purpose motion estimator. Each choice made to meet some performance and area goal. Used lots of clever implementation tricks: the “obvious” implementation derived from the spec would require a chip many times this size (and hence impossible to build today).

• 2.5MM transistors required lots of support from CAD tools for assembling the blocks, doing the routing and VERIFYING that everything was hooked up okay.

AVP-III Video Codec from Lucent Technologies
Architectural choices

- Reducing marketing and budget input to an engineering specification that meets the constraints

- Deciding between general-purpose (programmable) vs. special-purpose (hard-wired) implementations for each major functional block.

- Choosing the appropriate architecture for each block trading off latency, throughput, area, power, design time, verifiability, wire densities, ...

- Choosing the appropriate implementation strategy for each block
Implementation strategies

One could just start drawing transistors, but this quickly becomes hopeless with a chip of any size. Instead we’ll use one of the following strategies:

Standard cell: predefined gates, automatically placed and routed. In .5u → 10K fets/mm²

Full custom: custom “cells” meant to be stacked in columns to create N-bit wide datapath. Signals between columns routed across cells. In .5u → 25K/mm²

RAM Generator: one cell iterated many times perhaps surrounded by driver/sensing logic. Basic structure stays the same, only dimensions change. In .5u → 45K/mm² for multiport regfile
Implementation metrics

• Die size
  – yield, cost
• # of drawn fets
  – design time, risk
• Levels on interconnect
  – cost, signal integrity
• Density (fets/mm²)
  – yield, silicon efficiency
• Scalability
  – future costs, risk
• Verification time
  – NRE, functional first silicon
• % of chip active each cycle
  – silicon efficiency
• Power supply, noise margins
  – packaging cost, reliability
• Operating frequency
  – packaging costs, overhead
• W/Hz
  – battery life, power dissipation
CAD Tools

- organize
- generate
- verify

Tools to do the tedious, repetitive work such as routing, “tiling” a mosaic of building-block cells, or verifying that the layout and schematic match.

Automated layout tools to ease the task of physical design; mask verification to ensure manufacturability.

Standard-cell place and route for “random” logic.

Circuit analysis programs predict circuit behavior at all the process corners. Gate-level and behavioral simulators help you get it right the first time!
VLSI: The Ideal Implementation Medium?

So, what’s not to like about VLSI? VLSI

- gives the designer control over almost everything: architecture, logic design, speed, area, power, …
- densities are increasing, costs decreasing with each passing year
- is used by almost everyone: “No one gets fired for building a FPGA or ASIC”
- was the enabling technology for much of the economic growth of the 80’s and 90’s. It will no doubt continue in its starring role for some time come.

Is life really a bowl of cherries?
VLSI Fact-of-Life #1:
“So much to do, so little time”

You need a design methodology:

- budget ($, speed, area, power, schedule, risk)
- low-level building blocks, high-level architecture
- specification*
- behavioral design, verification*
- logic design, verification
- layout, verification

* Sometimes skipped in the “old days”
VLSI Fact-of-Life #2:  
“You can’t reach in and fix it”

Notice that the word “verification” kept appearing in the previous slide.

Mistakes can be costly:

| find bug(s) | ? | ?       |
| reverify    | 1wk | ?       |
| new masks   | 3da | $350k   |
| fab run     | 12wk | $1.5k/wafer |
| slip ship date | | $$$ |

There’s a lot that needs checking:

• circuit must operate at all “corners”
  – verified at building block level
• logic must be correct, operate reliably
  – verified at behavioral & RTL/gate level
• chip has to interoperate with system
  – verified at behavioral level
• chip has to be manufacturable
  – verified at mask level, at tester
VLSI Fact-of-Life #3: “You can’t find all the bugs”

The key word here is “find”:
- one can’t explore the behavior of the circuit under all possible conditions
- some of the bugs arise from unanticipated interactions which, by definition, one never thinks of testing
- it’s not clear when one is “done” looking for bugs! Time pressures mean that most searches stop too soon.

The trick is to choose some implementation rules that result in a circuit that is correct by construction*

For example:
- choose a simple clocking scheme
- module inputs must go only to fet gates
- disallow unclocked feedback
- make register t(clk-to-Q) > t(hold)+skew
- use poly only for local interconnect
- no diffusion wires
- etc., etc., etc.

* or at least avoids as many problems as possible!
VLSI Fact-of-Life #4: “Nobody’s perfect”

Plan for what happens after you turn it on and nothing happens.
- provide lot’s of observability and controlability. You’ll need to localize and then find the bug.
- have a way to run the chip slowly and/or stop it without it burning up or loosing bits.
- figure out how to track down performance problems without relying on fast I/O (tester pins are slow!)
- leave room in the budget (time, $) for debugging.
- write and run your manufacturing tests before tape out.
Coming up…

• Fri (9/6) & Wed (9/11):
  – hardware description languages
  – Synthesizable Verilog
  – Verilog examples
  – Reading
    • http://www.vol.webnexus.com/

• Fri (9/13):
  – Synthesis and logic optimization techniques