ECE 222 Fall 2019 Lecture 1
Oct 22, 2019
Syllabus (see the course website)
https://ece222-fall19-01.courses.soe.ucsc.edu

webcast.ucsc.edu
U: ee-222-1
P: Kang lecture
S. Kang, Office: BE-239
http://kang.soe.ucsc.edu

Source:
Introduction
Andreas Burg
Telecommunications Circuits Laboratory,
EPFL, Switzerland

History of Digital Circuits

- The abacus: the first computation device
- Invented around 2400 BCE
- Still in use today
- Napier’s bones:
  - Invented by John Napier (~1590)
  - Addition, multiplication, logarithms
- Slide rule:
  - Introduced in 1620
  - Analog Computer

Binary Logic
- Pingala discovered the Binary Numeral System (~300 BCE India)
- Leibniz described Binary Logic (~1650 Germany)
- Boolean Algebra was published by George Boole in 1854
- Mechanical Calculators
  - First calculator by Schickard (1623), followed by Pascal and Leibniz.
  - First mass-produced calculator by Thomas (1820)
History of Digital Circuits

Punch Cards
- In 1725, Beamon developed an *Automatic Loom* based on holes in paper.
- In 1801, Jacquard enabled using punch cards to control such a loom.
- In 1822, Charles Babbage described the *Difference Engine*, which is considered the first real computer design, though it was only made in 1991 (it is still operational at the London Science Museum).
- In 1834, Babbage described the *Analytical Engine* based on punch cards and a steam engine. It was the first general purpose programmable computer.

20th Century Milestones
- 1906 - The *Electronic Valve* is invented (De Forest). This is the switch that enabled the development of the digital computer.
- 1919 - The *Flip Flop* was proposed (Eccles, Jordan).
- 1937 - Alan Turing publishes paper describing the "Turing Machine" and sets the basis for computer theory. Turing is considered "The Father of Modern Day Computing".

20th Century Milestones
- 1939 - First machine to calculate using *vacuum tubes* developed.

History of Digital Circuits

20th Century Milestones
- 1896 - Herman Hollerith establishes the *Tabulating Machine Company*, later to become IBM (1924).
The first “Bug”

The first integrated circuit was invented by Jack Kilby of TI. The first silicon IC was invented by Robert Noyce of Fairchild half a year later.

In 1958, my goals were simple: to lower the cost, simplify the assembly, and make things smaller and more reliable. And although I do not consider myself responsible for all the activity that has followed, it has been very satisfying to watch the IC’s evolution. I’m pleased to have had even a small part in helping turn the potential of human creativity into practical reality.”

First Successful Operation of MOS Transistor

Synchronous Operation of MOS Transistor

- Surface Germanium Code (SGC)
- Silicon on Insulator (SOI)
- Self-Align Lithography (SAL)
- Gate All Around (GAA)

Kilby’s Integrated Circuit

- First silicon IC was invented by Robert Noyce of Fairchild half a year later.
- In 1958, my goals were simple: to lower the cost, simplify the assembly, and make things smaller and more reliable. And although I do not consider myself responsible for all the activity that has followed, it has been very satisfying to watch the IC’s evolution. I’m pleased to have had even a small part in helping turn the potential of human creativity into practical reality.”

Synchronous Operation of MOS Transistor

- Surface Germanium Code (SGC)
- Silicon on Insulator (SOI)
- Self-Align Lithography (SAL)
- Gate All Around (GAA)

Kilby’s Integrated Circuit

- First silicon IC was invented by Robert Noyce of Fairchild half a year later.
- In 1958, my goals were simple: to lower the cost, simplify the assembly, and make things smaller and more reliable. And although I do not consider myself responsible for all the activity that has followed, it has been very satisfying to watch the IC’s evolution. I’m pleased to have had even a small part in helping turn the potential of human creativity into practical reality.”

First Successful Operation of MOS Transistor

- Sodium (Na), Ohio State Univ. (Ph.D. 1959)
- Dr. Kahng, with M. Alaila, fabricated a MOSFET using a gate insulator formed from high quality SiO2 grown by a new high-pressure steam oxidation process at Bell Labs (1960)
- First successful demonstration of MOSFET was a major milestone in semiconductor technology
- Co-invented in 1967 the first nonvolatile silicon memory (floating gate memory) with Simon Sze
- Became Founding President of NEC, Princeton, NJ in 1988
1961 Fairchild Flip Flop

4T 5R result from Planar process which was soon used to fabricate RTL. The volume production era started, both at TI and Fairchild.

History of Digital Circuits

The Transistor Era

- 1960 – First MOSFET Fabricated
- 1962 – TTL Invented
- 1963 – CMOS Invented (solve TTL Power issue)
- 1964 – 1-inch silicon wafers introduced
- 1965 – Moore’s Law (more in a minute...)
- 1967 – Floating Gate invented
- 1970 – First commercial DRAM (1Kbit)
- 1971 – Microprocessor invented
- 1978 – Intel 8086/8088
- 1981 – IBM PC is introduced

The Xerox Alto (1973)
- Mouse
- Graphical Display
- LAN
- WYSIWYG Editor
- Drawing Program
- Windows UI

Cray Supercomputer (1976)

“Small” Computers

The Apple 1

DEC PDP-8

The first “minicomputer”

The IBM PC 5150 (1981)
Moore’s Law
In 1965, Gordon Moore noted that the number of transistors on a chip doubled every 18 to 24 months. He made a prediction that semiconductor technology will double its effectiveness every 18 months.

Moore’s Law Guides the Industry R&D
“The complexity for minimum component costs has increased at a rate of roughly a factor of two per year. Certainly over the short term, this rate can be expected to continue, if not to increase. Over the longer term, the rate of increase is a bit more uncertain, although there is no reason to believe it will not remain nearly constant for at least 10 years. That means by 1975, the number of components per integrated circuit for minimum cost will be 65,000.”

Gordon Moore, Cramming more Components onto Integrated Circuits, (1965).

Moore’s Law – Transistor Count
The more devices on a single IC, the more functions we can sell for the same price.

Moore’s Law – Transistor Cost
Moore’s Law is, at its base, an Economical law.

Study: Moore’s Law Really Works After
Illustration: Evolution of Microprocessors

Intel Core 2 Duo Microprocessor
Introduced 2006
Clock speed: 2.5 GHz
291 Mio Transistors
65nm Technology
Power: 65W

Quad-Core Intel Xeon Microprocessor
Introduced 2007
Clock speed: >3 GHz
820 Mio Transistors
45nm Technology
Power: 45W

Teraflops Research Chip
Introduced 2006
65nm Technology
80 Processor cores
- 3.16 GHz 62W 1.0 TFlops
- 5.1 GHz 175W 1.6 TFlops
- 5.7 GHz 265W 1.8 TFlops

For comparison: ASCI Red was the first supercomputer to reach TFlops in 1996. That system used nearly 10,000 Pentium® Pro processors running at 200Mhz and consumed 500kW of power plus an additional 50kW just to cool the room that housed it.

The Thermal Power Limit

Power Density Increases

Slide 1.28
There exist very compelling reasons why a further increase in power density should be avoided at all costs. As shown in an earlier slide for the PowerPC 5, power densities on chips can become excessive and lead to degradation or failure, unless extremely expensive packaging techniques are used. To drive the point home, power density levels of some well-known processors are compared to a collection of real-world examples, such as hot plates, nuclear reactors, rocket nozzles, or even the sun’s surface. Surprisingly, high-performance IC’s are not that far off from some of these extreme heat sources! Classic wisdom dictates that power densities above 150W/cm² should be avoided for the majority of designs, unless the highest performance is an absolute must and cost is not an issue.
Moore's Law is Alive and Well

Overall Design Flow

Architecture Design

The golden rule: ALWAYS START WITH A BLOCK DIAGRAM!

- Identify blocks
- What do we need to perform the functionality
- Visualize structure
- How are blocks connected
- Find critical paths
- Which block is the most critical (speed, area, power)?
- Divide and Conquer
- Use hierarchy, i.e., draw sub-block diagrams

Use hierarchy also in your block diagram

Architecture Design: Structured Design Principle

Hierarchy: “Divide and conquer” technique involves dividing a module into submodules and then repeating this operation on the sub-modules until the complexity of the smaller parts becomes manageable.

Regularity: The hierarchical decomposition of a large system should result in not only simple, but also similar blocks, as much as possible. Regularity usually reduces the number of different modules that need to be designed and verified, at all levels of abstraction.

Modularity: The various functional blocks which make up the larger system must have well-defined functions and interfaces.

Locality: Internal details remain at the local level. The concept of locality also ensures that connections are mostly between neighboring modules, avoiding longdistance connections as much as possible.

VLSI Design Styles for Logic/Circuit/Physical Design

VLSI Design Styles – Full Custom

Advantages
- Everything is done on device/transistor level
  - Full control over all device parameters
  - Full flexibility w.r.t. circuit topology
- Excellent performance
- Very high accuracy in simulations
- No strict separation between analog and digital parts
- Still the only option for analog design

Disadvantages
- Everything is done manually
  - Limited design capacity (size) to few hundred devices
  - Long design time and limited re-use
- No abstraction/approximation
  - Slow

Schematic Entry
Analog Simulation
Manual Layout
Layout Check/Extract
Post-Layout Simulation
**VLSI Design Styles – Full Custom**

**Schematic Entry**

![Schematic Entry Image]

**Simulation using Spice or Spectre**

![Simulation Image]

**VLSI Design Styles – Full Custom**

**Layout: layers represent the masks for production**

![Layout Image]

**VLSI Design Styles – Full Custom**

In full-custom style, the designer has many degrees of freedom to optimize a circuit design:

- Adjust individual transistor dimensions (width, length, aspect ratio, etc.) to satisfy:
  - DC specifications (voltage levels, switching thresholds)
  - Transient specifications (delay times, rise- and fall-times)

- Freely choose the most appropriate topology (placement and routing) for each circuit block.

- Decide on interconnection strategy between blocks.

- Decide for the global distribution of power, ground and clock.

**Limits of Full Custom Design**

Increasing integration density no longer allows for design on transistor level, neither on schematic, nor on layout level

![Limit Image]

**Principle Idea of Semicustom Design Flow**

Render the design process more efficiently by using:

- **Hierarchy**: build complex designs from a collection of smaller and much simpler components which by themselves are again hierarchical

- **Abstraction**: simplified description/characterization of components as a model (black box) to better use them on the next level of hierarchy

- **Design automation**: algorithms and tools to realize an abstract design description from components

In full-custom style, the designer has many degrees of freedom to optimize a circuit design:

- Adjust individual transistor dimensions (width, length, aspect ratio, etc.) to satisfy:
  - DC specifications (voltage levels, switching thresholds)
  - Transient specifications (delay times, rise- and fall-times)

- Freely choose the most appropriate topology (placement and routing) for each circuit block.

- Decide on interconnection strategy between blocks.

- Decide for the global distribution of power, ground and clock.

**Limits of Full Custom Design**

Increasing integration density no longer allows for design on transistor level, neither on schematic, nor on layout level

![Limit Image]

**Principle Idea of Semicustom Design Flow**

Render the design process more efficiently by using:

- **Hierarchy**: build complex designs from a collection of smaller and much simpler components which by themselves are again hierarchical

- **Abstraction**: simplified description/characterization of components as a model (black box) to better use them on the next level of hierarchy

- **Design automation**: algorithms and tools to realize an abstract design description from components
Abstraction Levels: Technology

Layers of doped semiconductor material or metal interconnect.
- Possible arrangement of these layers is determined by manufacturing processes.
- Details are one of the best guarded secrets of a foundry (years to develop).

Composition of layers determines electrical and other characteristics.
- Characterized by the complex laws of physics.
- Technology Computer Aided Design (TCAD) tools used for analysis and simulation.

Abstraction Levels: Devices

Basic building blocks for circuit design that abstract the physics of the technology layer to electrical characteristics.
- Foundries supply a Process Design Kit (PDK) which provides.
  - Devices (Transistors, Resistors, Capacitors, Diodes).
  - Layers (e.g., for interconnect).

Various “Flavors” of PDKs are available, e.g.:
- General Purpose/High Speed/Low Power.
- RF/Image Sensor.
- Flash/DRAM.

Devices are abstracted to compact electrical models for circuit simulations and design rules.
- Generated through TCAD and/or measurements.

Abstraction Levels: Macros and Standard Cells

Basic analog or digital circuits as hierarchical building blocks for more complex circuits:
- Built from devices of the PDK with full control over all device parameters.
- Characterized and optimized through circuit simulation, e.g., using SPICE.

Library and IP providers offer for example:
- Standard cells: libraries of basic digital gates.
- Digital macros such as RAMs/ROMs.
- Analog macros such as ADCs/DCDCs.

Standard cells/macro cells are described by abstract models that capture:
- Functionality.
- Interface (logical and electrical).
- Performance (e.g., delay and power).
- Physical appearance (e.g., size and shape).

Abstraction Levels: Register Transfer Level (RTL) Design

Complex digital blocks (IPs) and complete complex digital ICs.
- Specification of a digital circuit in a Hardware Description Language (HDL).
  - Defines computational logic and storage elements in an abstract way.
- Semicustom design:
  - Frontend: Automatic translation of HDL into a Gate Level netlist (a circuit built from standard cells and macro blocks).
  - Backend: Physical implementation of the design based on basic building blocks.

RTL designs can be abstracted through behavioral models.

Abstraction Levels: Architecture

Entire system (e.g., cell phone):
- Complex system comprised of individual (partially highly complex) components.
- Communicating between components often through complex interfaces and protocols.

Described either:
- Still in RTL by instantiating many other RTL components or
- In a high-level or HDL language instantiating behavioral models of the RTL components.

A Digital IC with a Semicustom Design Flow

Composed from few macros and digital logic realized with standard cells & pads.
Semicustom Design Flow: Design Entry in VHDL

Modern hardware description languages allow structural descriptions that basically determine all interconnections between well-defined blocks:

```vhdl
adder: component adder port map (a, b, ci, co, s);
```

as well as purely behavioral descriptions like:

```vhdl
sum <= a + b - c;
```

The synthesizer is responsible for converting the behavioral description into an optimized design.