

Advanced VLSI Design

Instructor:

Professor Jim Plusquellic

Text:

Digital Integrated Circuits: A Design Perspective,
by Jan M. Rabaey, Prentice Hall (1996).

Further Info:

<http://www.cs.umbc.edu/~plusquel/>



Purpose of the Course

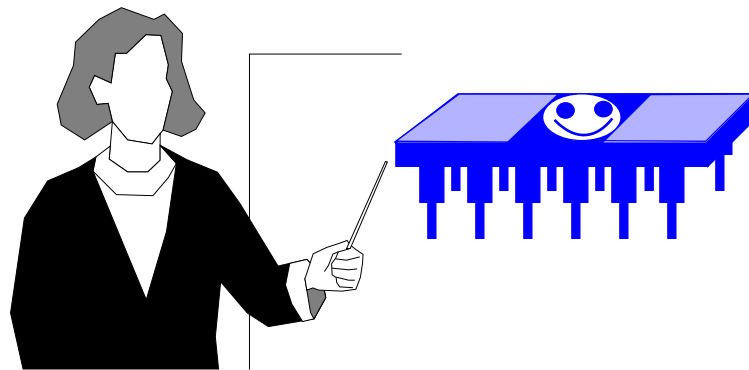
- To introduce the concepts and techniques of modern integrated circuit design (CMOS VLSI).
- To provide experience designing integrated circuits using Commercial Computer Aided Design (CAD) Tools (CADENCE).



The VLSI Design Process

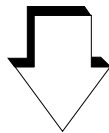
The Design Process: An iterative process that refines an “idea” to a manufacturable device through at least five levels of design abstraction.

Top level: The “idea” $\xrightarrow{\text{refined into a set of requirements called}}$



Specification:

- What does the chip do?
- How fast does it need to operate in order to be competitive?
- How much power will it consume?
- How big will it be?



Design Constraints:

Speed, power and area.

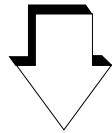
Abstraction: A very effective means of dealing with design complexity. Creating a model at a higher level of abstraction involves replacing detail at the lower level with simplifications.

The VLSI Design Process

Simulation: The functional behavior of the design (or a parameter such as power) is determined by applying a set of excitation vectors to a circuit model.

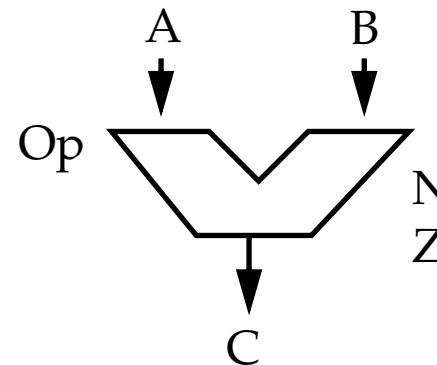
VHDL (Hardware Description Language)

Specification



Functional Design

Behavioral Simulation

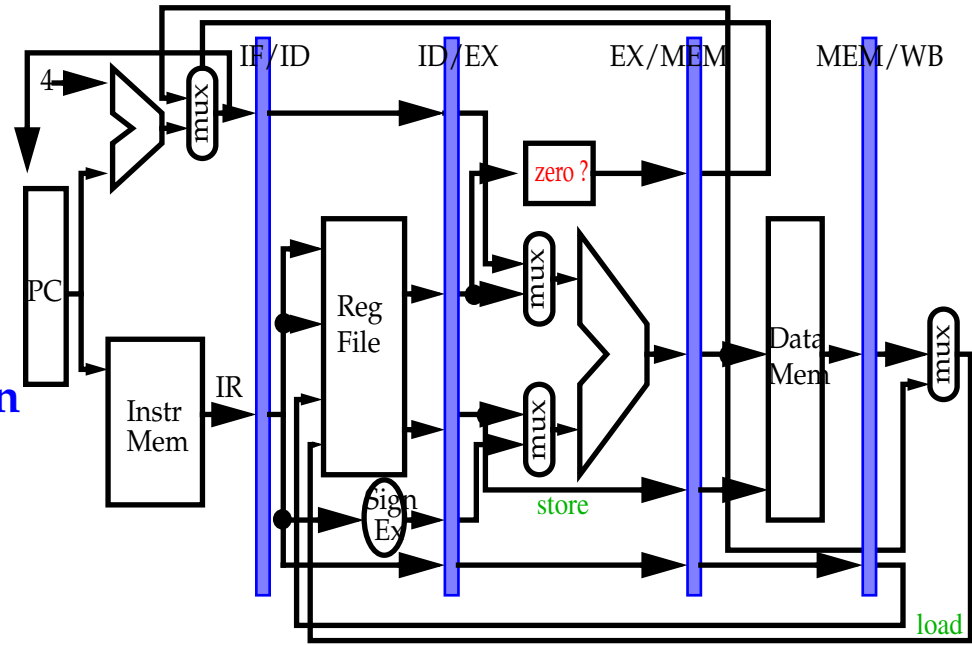


```
entity ALU32 is port(
  A, B: in bit_vector( 31 downto 0);
  C: out bit_vector( 31 downto 0);
  Op: in bit_vector( 5 downto 0);
  N, Z: out bit);
end half_adder;
```

The VLSI Design Process

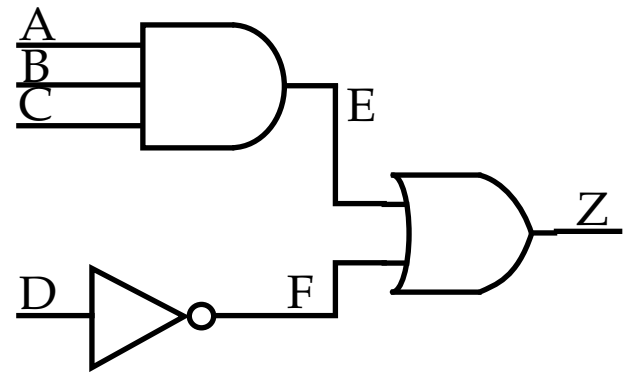
Register Transfer Level Design

RTL Simulation

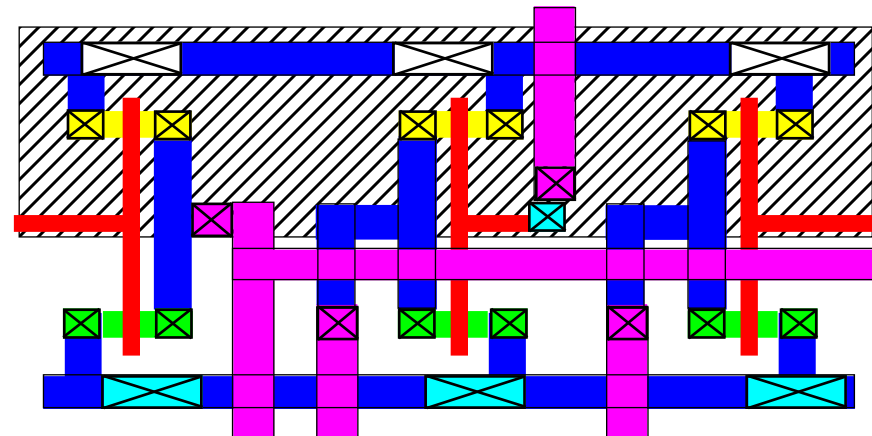
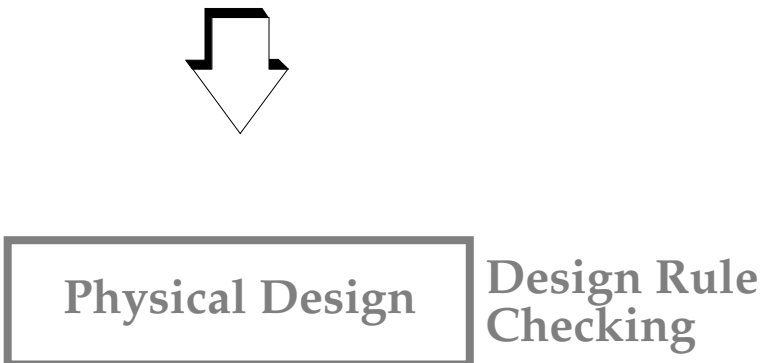
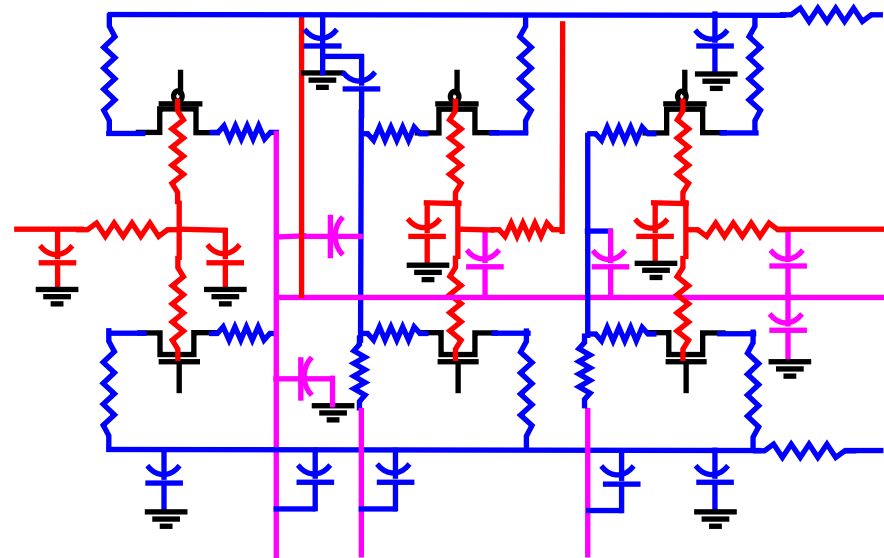
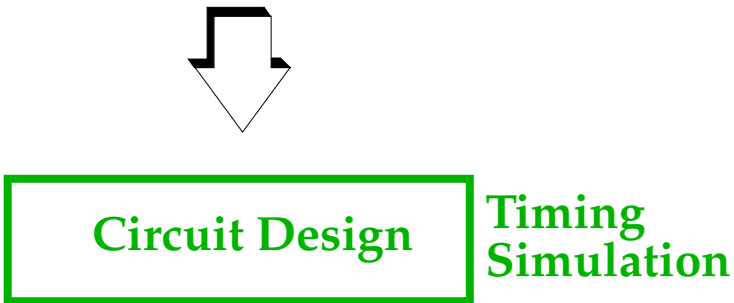


Logic Design

Logic Simulation



The VLSI Design Process



Brief History

TTL (Transistor-Transistor logic).

First successful IC logic family.

Composed largest fraction of digital IC market until 80's.

Power consumption per gate set upper limit on integration density.

I²L (Integrated Injection Logic):

An attempt to provide a high integration density, low power bipolar family of logic.

MOS (Metal-Oxide-Silicon): Actually, we use polysilicon for gates now.

Gate stability problems solved in 60's.

CMOS was first !

Complexity of manufacturing process delayed use until 80's.

PMOS-only used through early 70's.

In 1974, the 8080 microprocessor was implemented using faster NMOS-only.

Late 70's, NMOS-only started suffering from same problem as high density bipolar technology -- **power consumption**.

Brief History

Since early 80's, CMOS remains the technology of choice.

However, **power consumption** is now becoming a problem.

And there is no new technology around the corner to alleviate the problem.

When performance is the main issue, other technologies are used:

BiCMOS: High speed memory and gate arrays.

ECL (Emitter-coupled logic): Even higher performance.

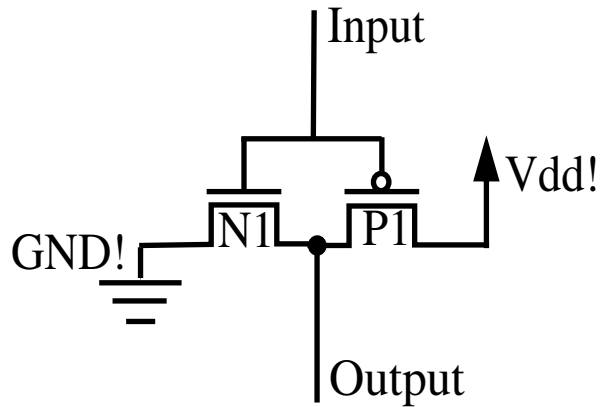
Galium-Arsenide

Silicon-Germanium

Superconducting Technologies

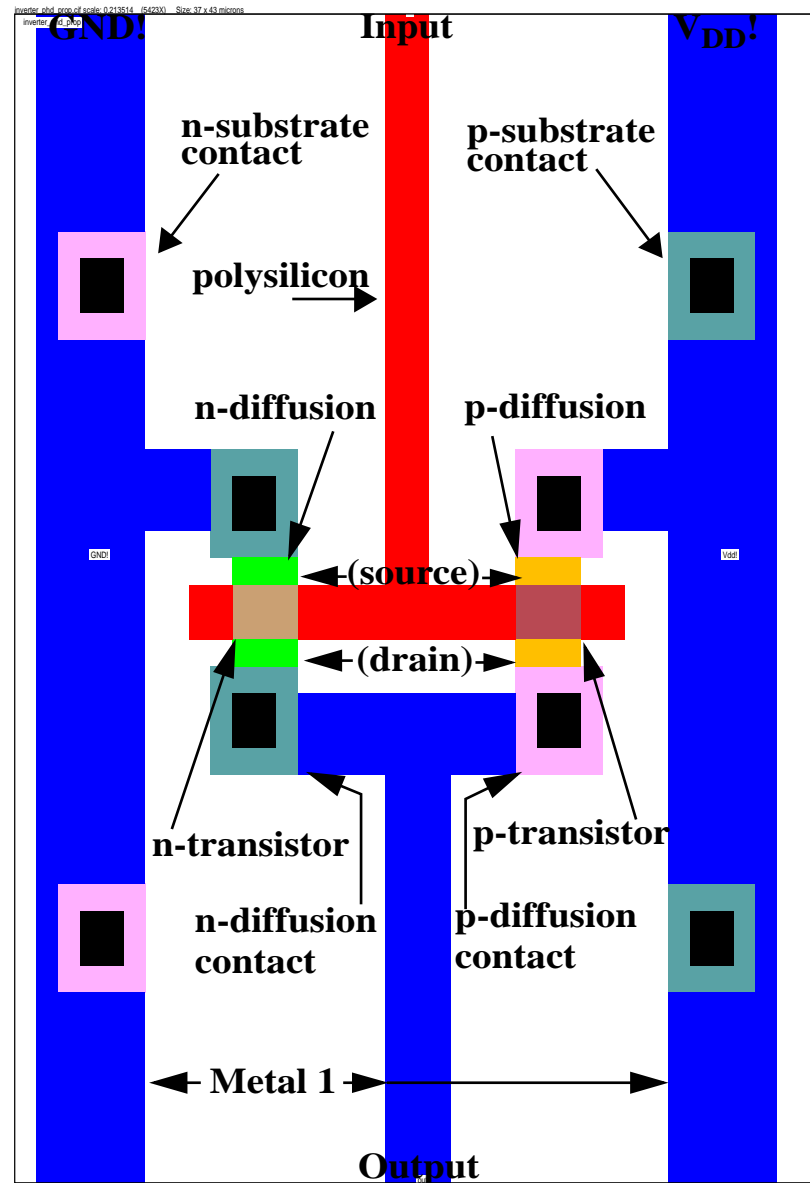
What is CMOS?

A CMOS Inverter



Inverter schematic diagram

Inverter layout



Hierarchy and Abstraction

Moore's Law: Number of transistors that can be integrated onto a single die increases exponentially with time.

Integration complexity doubles every 1-2 years.

For example, Microprocessors:

The million transistor/chip barrier crossed in '88 with the 486.

Impact of this revolution on design:

Hand crafting not possible anymore (as was done for the 4004).

Hierarchy is used in the design of the Pentium.

The processor is a collection of modules each composed of cells.

Re-use of cells reduces design effort and increases the chance of a first-time right implementation.

The use of hierarchy is a key ingredient to the success of the digital circuit.

Reason why large analog designs never caught on.

Hierarchy and Abstraction

Abstraction is also possible in digital designs.

And difficult to apply effectively to analog designs.

Critical element in dealing with complexity.

A multiplier, for example, can be designed and treated like a black box.

The performance of the multiplier is only marginally influenced by the way it is used in a larger system.

This **divide and conquer** (*hierarchical*) approach allows the designer to deal with a much smaller number of well characterized modules (or *abstractions*).

Abstraction levels:

Physical level: Rectangles, design rules.

Circuit level: Transistors, R and C, analog voltage/current values.

Switch level: Transistors, R and C, multi-valued logic.

Logic level: Boolean logic gates, binary valued logic.

Register Transfer Level: Adders, datapaths, binary valued words.

Functional level: Processors, programs and data structures.



Hierarchy and Abstraction

Entire CAD design frameworks are based on this design philosophy.

These have made it possible to achieve current design complexity.

Design tools include:

Simulation at various complexity levels.

Design verification.

Layout generation.

Design synthesis.

Standard cells are a popular design style that makes layout generation easy.

Layouts of basic gates such as AND, OR, NAND, NOR, and NOT as well as arithmetic and memory modules are provided as input.

These cells are designed with similar characteristics, such as constant height, and can be manipulated easily to generate a layout.

More on this later.

Digital Circuit Design

If design automation solves all the problems, why be concerned with digital circuit design?

Reality is more complex and a knowledge of digital circuit design will be important for some time to come.

- Someone has to design and implement the module libraries.
 - Porting from technology generation to technology generation (different feature sizes) is NOT automatic.
 - This occurs approximately every two years!
- Creating an adequate *model* of a cell/module requires an in-depth understanding of its internal operation.
 - A significant part of digital circuit design focuses on analysis of internal circuit operation.

Digital Circuit Design

- The library-based approach does NOT work for all situations, i.e. high performance designs like microprocessors.

For Application Specific Integrated Circuits (*ASICs*), library-based approach works well since the design constraints (speed, power, cost and area) are reduced.

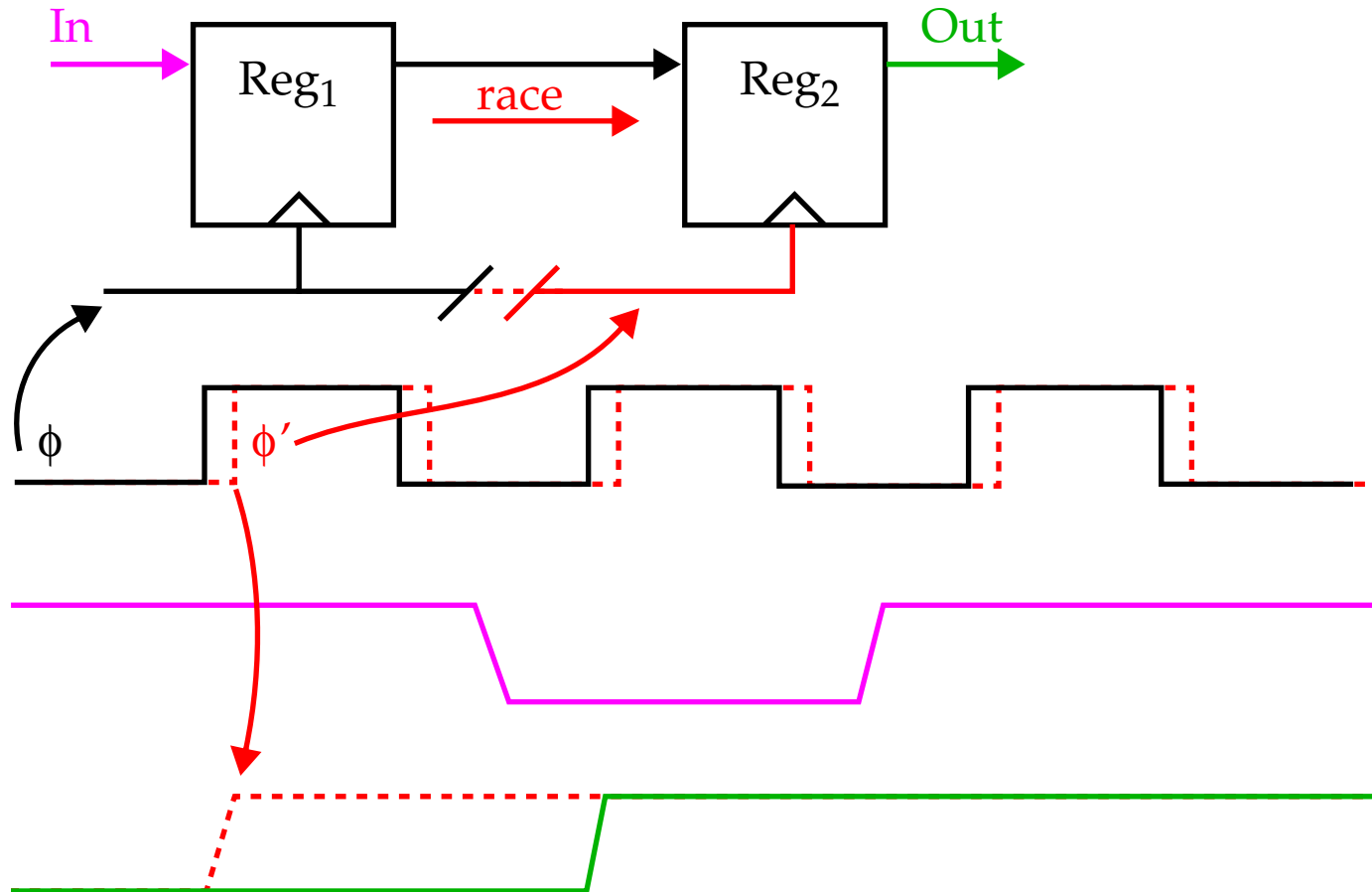
For microprocessor design, which push technology to its limits, this approach becomes less attractive.

- The abstraction-based approach is only correct to a certain degree. Performance of a module, i.e. an adder, is substantially influenced by the way it is connected in its environment (*interconnect parasitics*).

Digital Circuit Design

- Scaling tends to emphasize other deficiencies of the abstraction-based approach.
 - Global entities, such as clock signals and supply lines, are significantly affected by scaling.
 - Clock distribution, circuit synchronization and supply-voltage distribution are becoming more and more critical.
- New design issues emerge over time.
 - Power dissipation issue periodically re-emerges.
 - Recently, the ratio between device and interconnect parasitics (and consequently the appropriate delay model) is changing.
- Trouble shooting an erroneous design requires circuit expertise.



Scaling Issues: Clocks Defy Hierarchy

Clock skew creates race condition, i.e., Reg₁ value reaches Reg₂ before clk edge does, allowing Reg₁ value to appear at *Out* a clk cycle early.