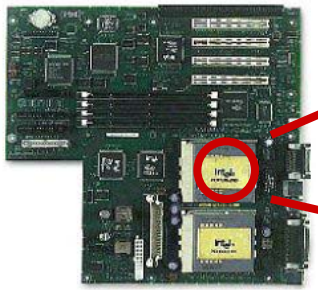
Digital Design & FPGA Workshop

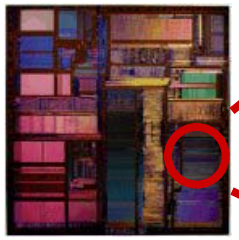
- Week 1 – Workshop kickoff
  - Goals
  - Expectations
  - Format
  - Formalities
  - Introduction into the topics we'll be covering
  - VM Setups, time permitting

How do you build systems with >1G components?

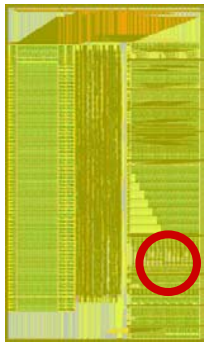
Personal Computer:
Hardware & Software



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

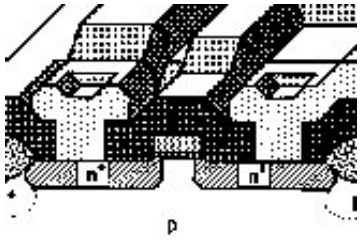


Integrated Circuit:
≈8-16 / PCB
.25M-16M devices

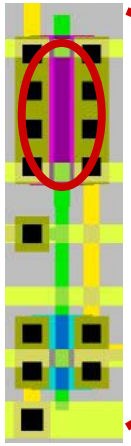


Module:
≈8-16 / IC
100K devices

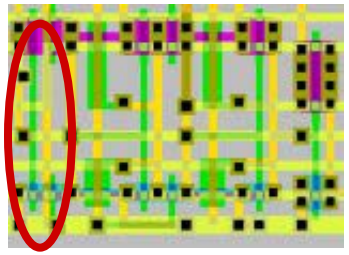
MOSFET



Scheme for
representing
information



Gate:
≈2-16 / Cell
8 devices



Cell:
≈1K-10K / Module
16-64 devices

What do we see?

- Structure
  - hierarchical design:
    - limited complexity at each level
    - reusable building blocks
- Interfaces
  - Key elements of system engineering; typically outlive the technologies they interface
  - Isolate technologies, allow evolution
  - Major abstraction mechanism
- What makes a good system design?
  - "Bang for the buck": minimal mechanism, maximal function
  - reliable in a wide range of environments
  - accommodates future technical improvements

Our plan of attack…

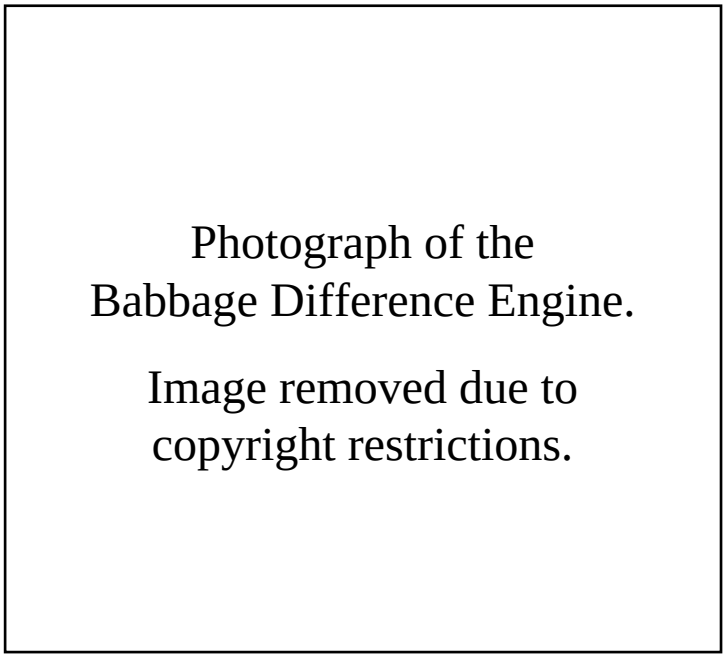
- Understand how things work, *bottom-up*
- Encapsulate our understanding
  using appropriate abstractions
- Study organizational principles:
  abstractions, interfaces, APIs.

- Roll up our sleeves and design at
  each level of hierarchy
- Learn engineering tricks
  - history
  - systematic approaches
  - algorithms
  - diagnose, fix, and avoid bugs

6.111 Goals and Prerequisite

- Design and Implement Complex Digital Systems
  - Fundamentals of logic design : combinational and sequential blocks
  - System integration with multiple components (memories, discrete components, FPGAs, etc.)
  - Use a Hardware Design Language (Verilog) for digital design
  - Interfacing issues with analog components (ADC, DAC, sensors, etc.)
  - Understand different design metrics: component/gate count and implementation area, switching speed, energy dissipation and power
  - Understand different design methodologies and mapping strategies (discrete logic, FPGAs vs. custom integrated circuits)
  - Design for test
  - Demonstrate a large scale digital or mixed-signal system
- Prerequisite
  - Prior digital design experience is NOT Required
  - 6.004 is not a prerequisite!
    - Take 6.004 before 6.111 or
    - Take 6.004 after 6.111 or
    - Take both in the same term
  - Must have basic background in circuit theory
  - Some basic material might be a review for those who have taken 6.004

The First Computer



The Babbage Difference Engine (1834)

25,000 parts

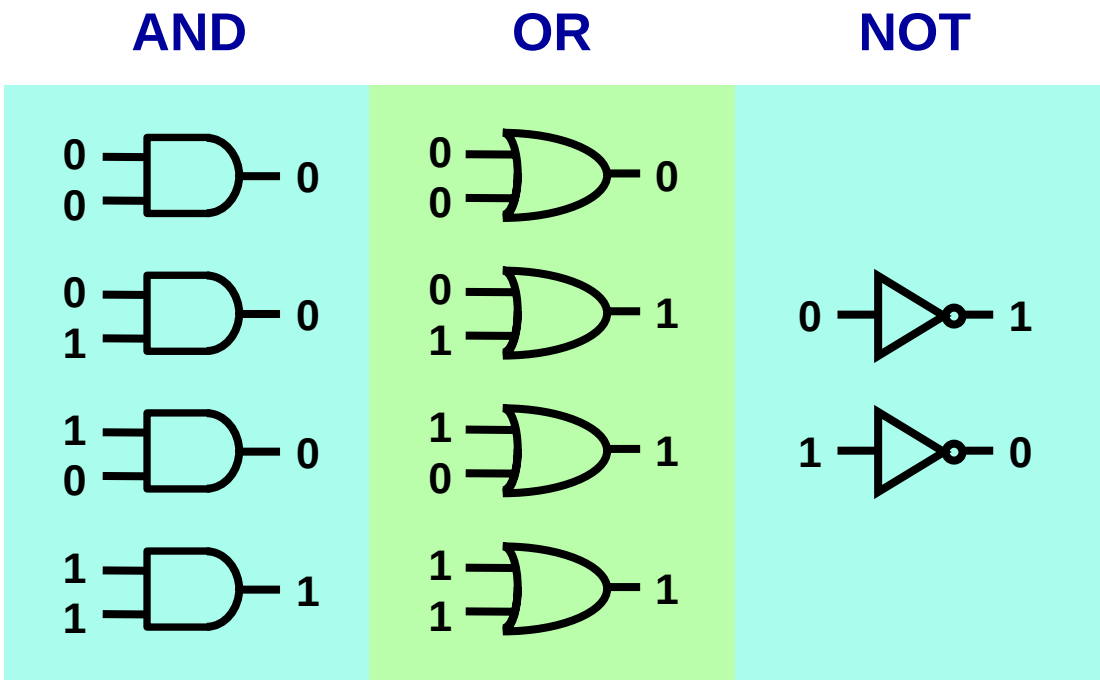
cost: £17,470

- **The first digital systems were mechanical and used base-10 representation.**
- **Most popular applications: arithmetic and scientific computation**

Meanwhile, in the World of Theory…

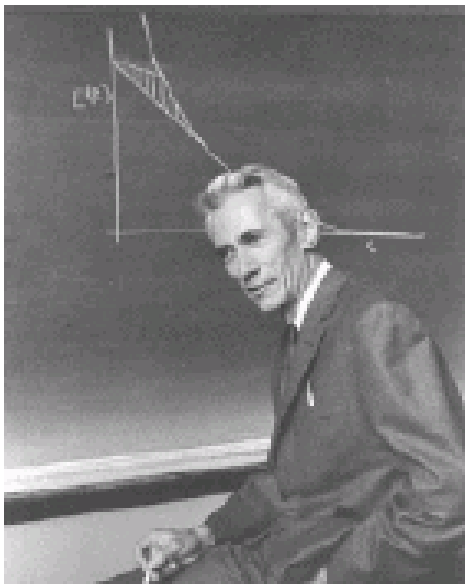
Photograph of George Boole.

Image removed due to copyright restrictions.



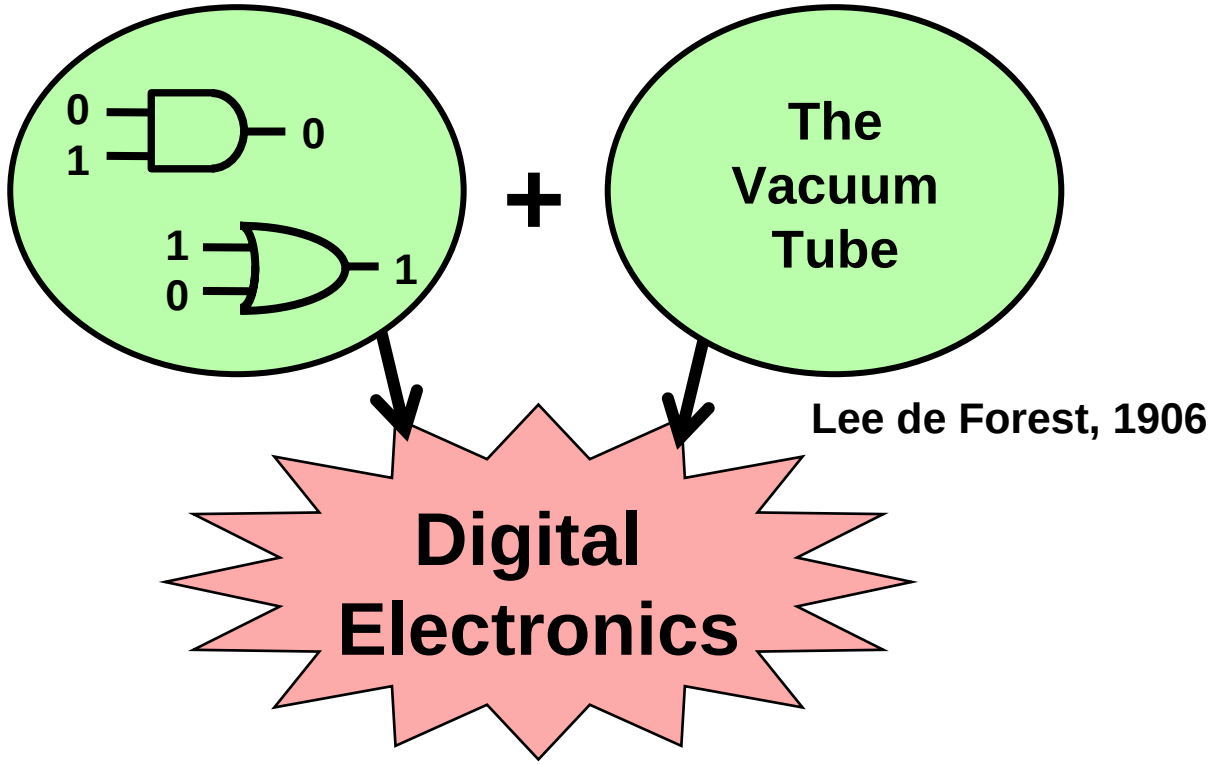
- **1854: George Boole shows that logic is math, not just philosophy!**
- **Boolean algebra: the mathematics of binary values**

Key Link Between Logic and Circuits



Claude Shannon

Courtesy of Jonah Sacks. Used with permission.



- Despite existence of relays and introduction of vacuum tube in 1906, ***digital*** electronics did not emerge for thirty years!
- Claude Shannon notices similarities between Boolean algebra and electronic telephone switches
- Shannon's 1937 MIT Master's Thesis introduces the world to binary digital electronics

Evolution of Digital Electronics

Vacuum Tubes

ENIAC, 1946

UNIVAC, 1951

1900 adds/sec

Transistors

First Transistor
Bell Labs, 1948

IBM System/360, 1964

500,000 adds/sec

VLSI Circuits

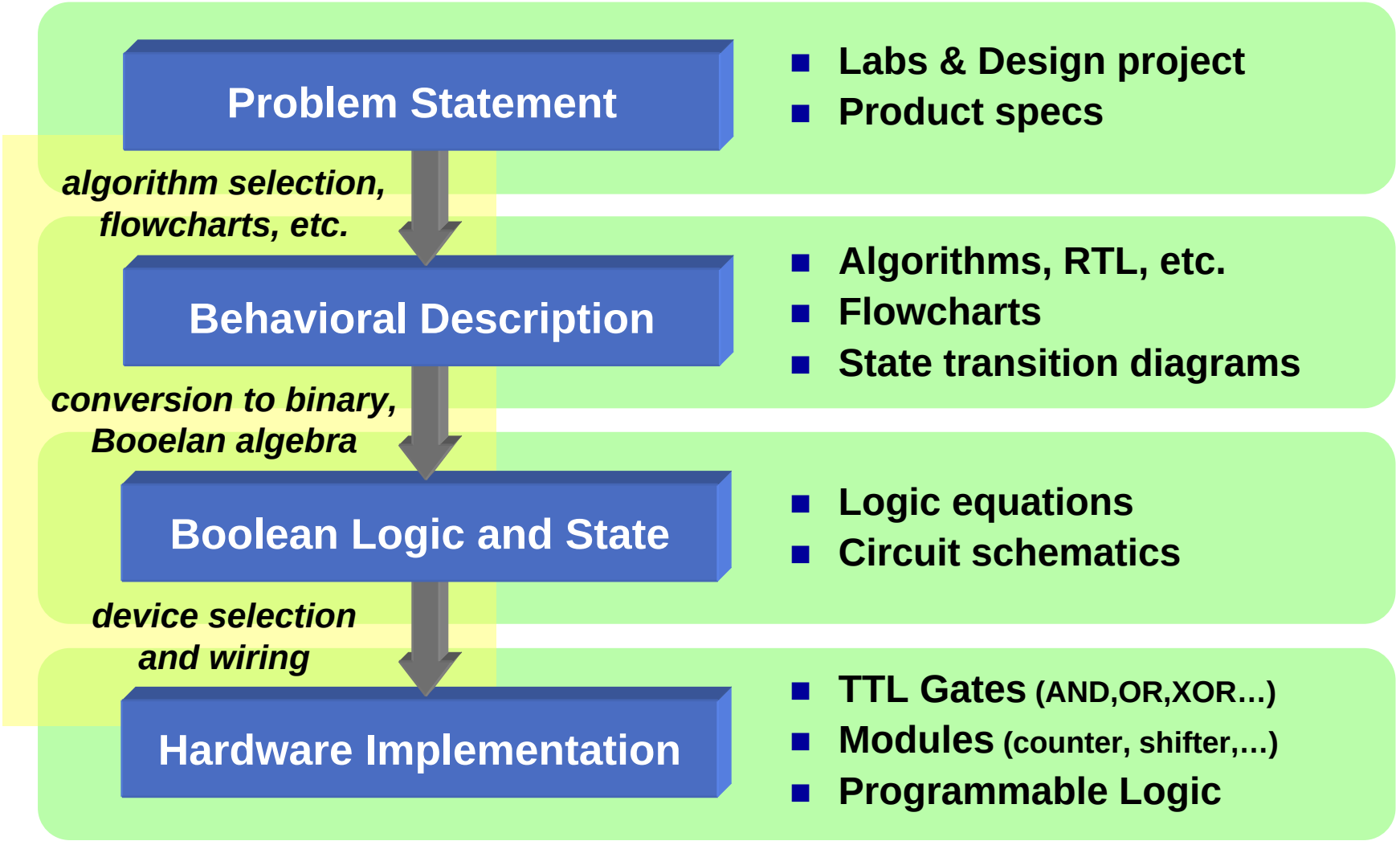
4004, 1971

Intel Itanium, 2003

2,000,000,000 adds/sec

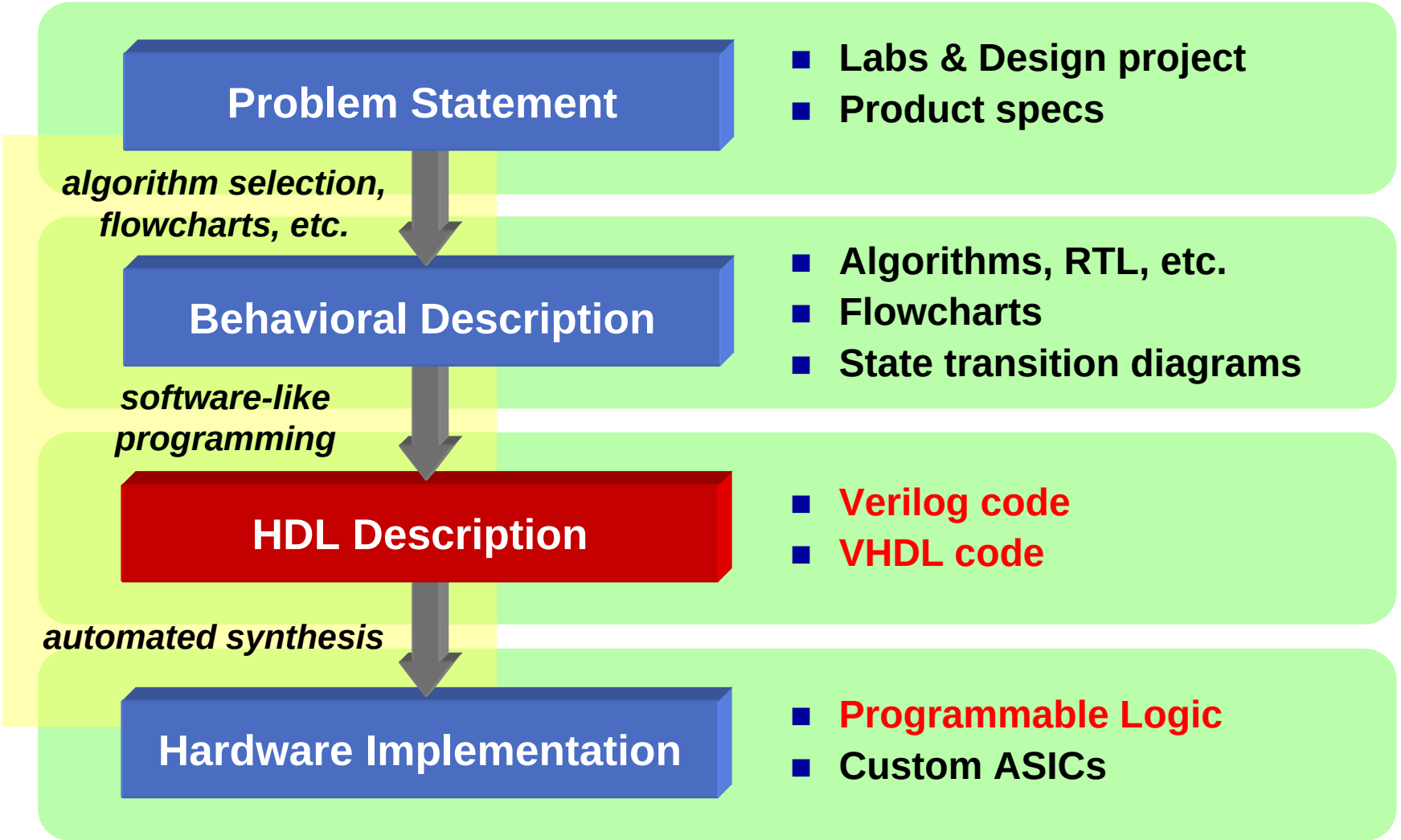
Building Digital Systems

- **Goal of 6.111: Building binary digital solutions to computational problems**



Building Digital Systems with HDLs

- Logic synthesis using a Hardware Description Language (HDL) automates the most tedious and error-prone aspects of design



Verilog and VHDL

VHDL	Verilog
Commissioned in 1981 by Department of Defense; now an IEEE standard	Created by Gateway Design Automation in 1985; now an IEEE standard
Initially created for ASIC synthesis	Initially an interpreted language for gate-level simulation
Strongly typed; potential for verbose code	Less explicit typing (e.g., compiler will pad arguments of different widths)
Strong support for package management and large designs	No special extensions for large designs

Hardware structures can be modeled effectively in either VHDL and Verilog. Verilog is similar to c and a bit easier to learn.

Levels of Modeling in Verilog

- **Behavioral or Algorithmic Level**
  - Highest level in the Verilog HDL
  - Design specified in terms of algorithm (functionality) without hardware details. Similar to “c” type specification
  - Most common level of description
- **Dataflow Level**
  - The flow of data through components is specified based on the idea of how data is processed
- **Gate Level**
  - Specified as wiring between logic gates
  - Not practical for large examples
- **Switch Level**
  - Description in terms of switching (modeling a transistor)
  - No useful in general logic design – we won’t use it

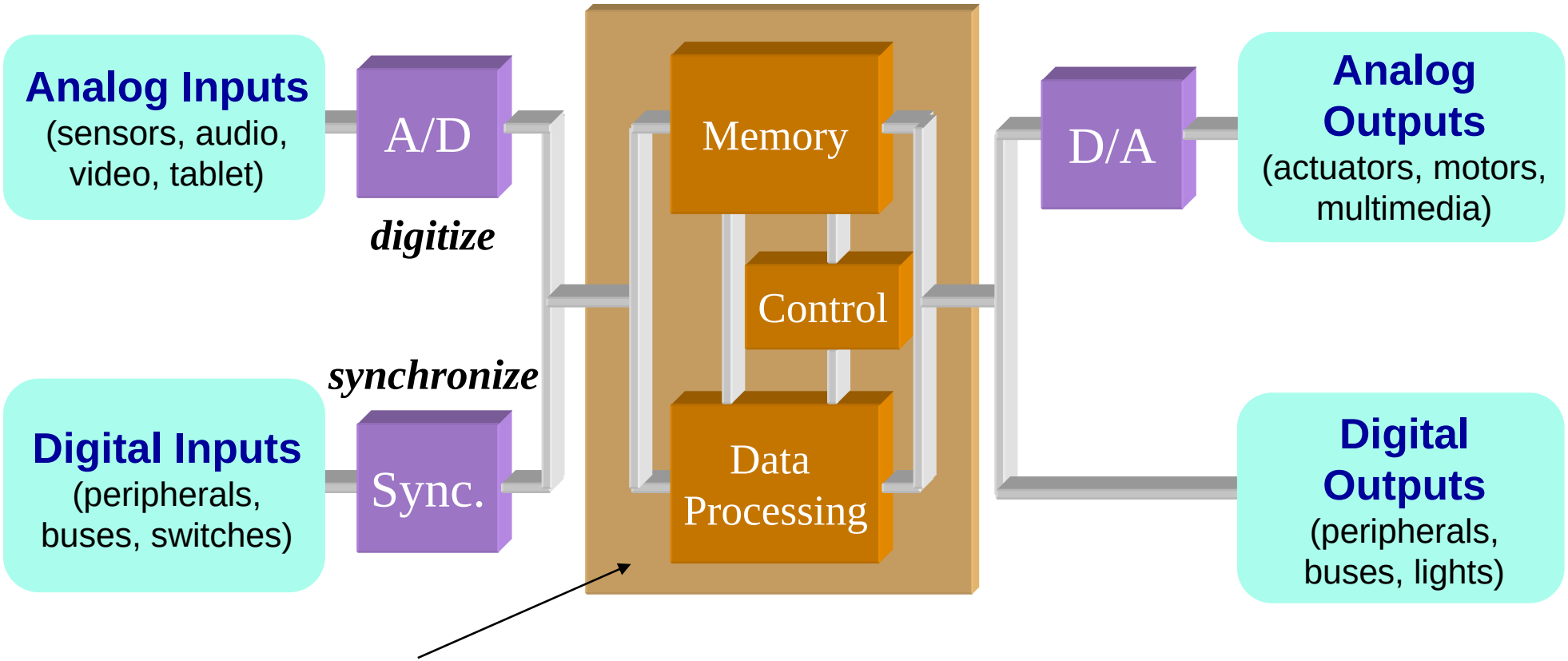
A design mix and match all levels in one design is possible. In general Register Transfer Level (RTL) is used for a combination of Behavioral and Dataflow descriptions

Verilog HDL

- **Misconceptions**
  - The coding style or clarity does not matter as long as it works
  - Two different Verilog encodings that simulate the same way will synthesize to the same set of gates
  - Synthesis just can't be as good as a design done by humans
    - Shades of assembly language versus a higher level language
- **What can be Synthesized**
  - Combinational Functions
    - Multiplexors, Encoders, Decoders, Comparators, Parity Generators, Adders, Subtractors, ALUs, Multipliers
    - Random logic
  - Control Logic
    - FSMs
- **What can't be Synthesized**
  - Precise timing blocks (e.g., delay a signal by 2ns)
  - Large memory blocks (can be done, but very inefficient)

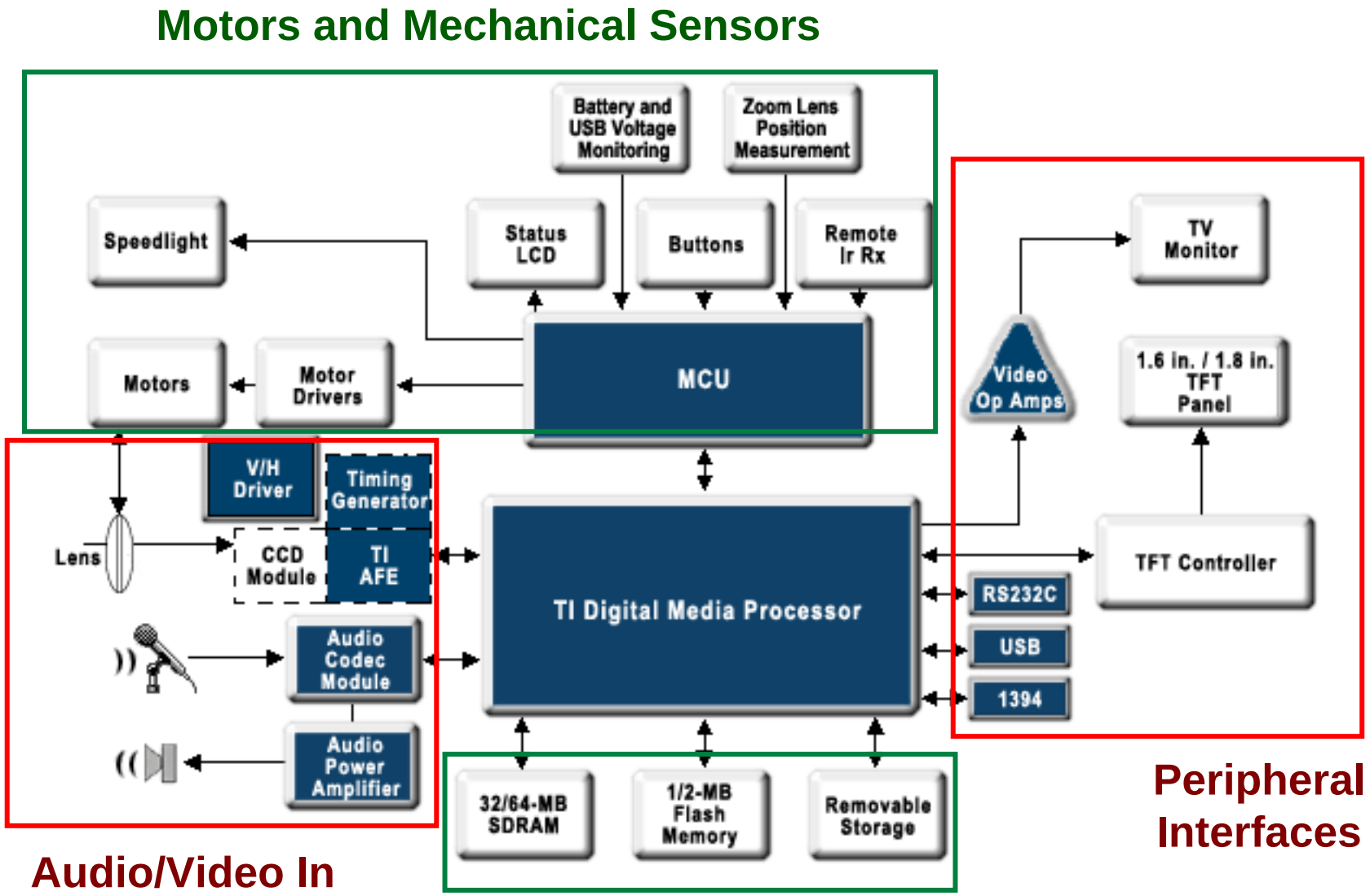
Understand what constructs are used in simulation vs. hardware mapping

Embedded Digital System



- **Digital processing systems consist of a datapath, memory, and control. Early machines for arithmetic had insufficient memory, and often depended on users for control**
- **Today's digital systems are increasingly embedded into everyday places and things**
- **Richer interaction with the user and environment**

TI Digital Camera Controller



Courtesy of Texas Instruments. Used with permission.

Real-World Performance Metrics

Cost	Speed	Energy
commodity products	scientific computing, simulation	portable applications

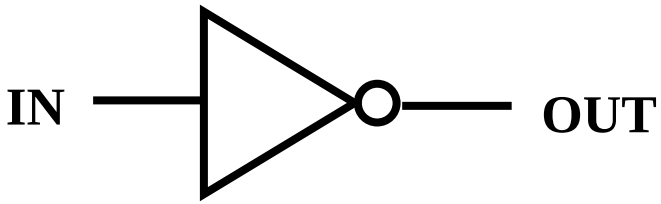
- **Commercial digital designs seek the most appropriate trade-offs for the target application…**
- **…keeping time-to-market in mind**

Verification and Testing

- Design can be fun. Verification/testing is hard work.
- Verification by simulation (and formally through test benches) is a critical part of the design process.
- The physical hardware must be tested to debug the mapping process and manufacturing defects.
- Physical realizations often do not allow access to internal signals. We will introduce formal methods to observe and control internal state.

Verification and Design for Test (DFT) are important components of digital design

The Inverter: Voltage Transfer Characteristic

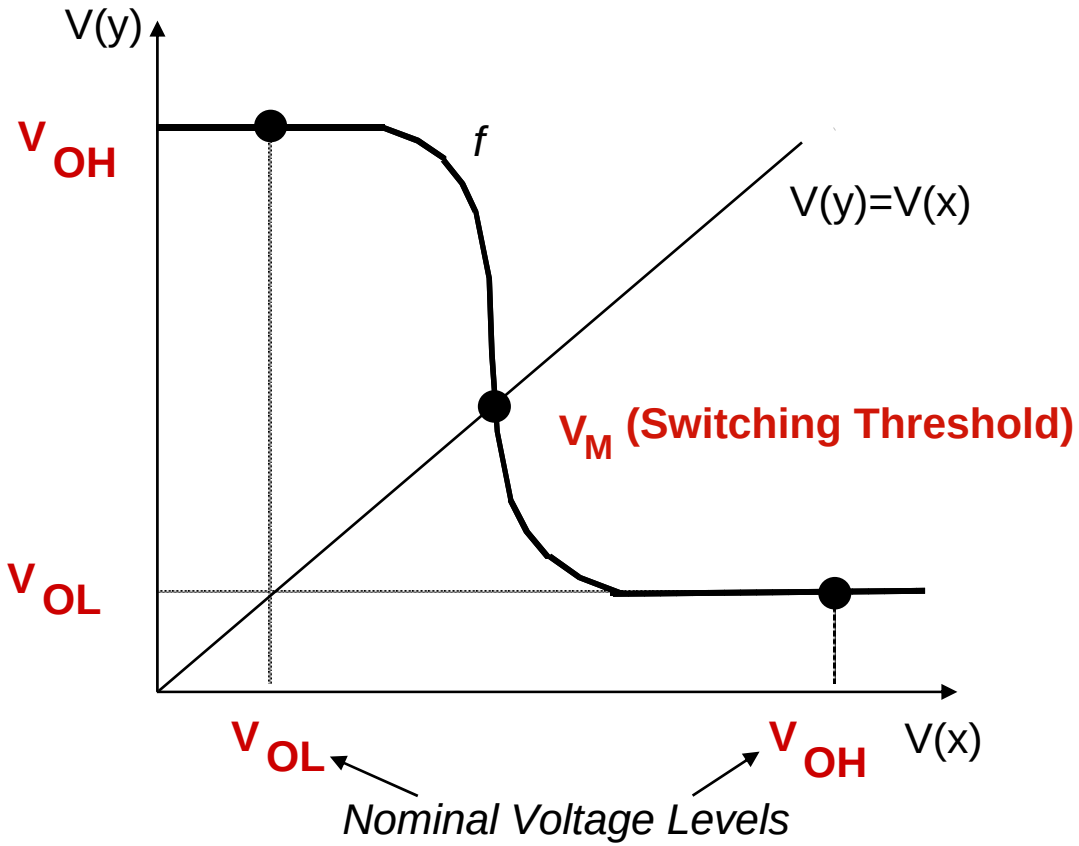


Digital circuits perform operations on logical (or Boolean) variables

A logical variable is a mathematical abstraction. In a physical implementation, such a variable is represented by an electrical quantity

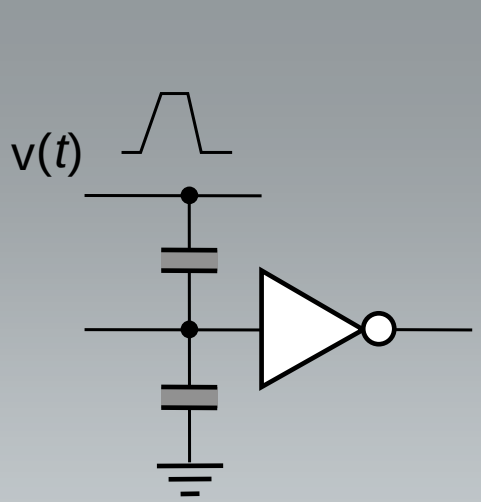
Truth Table

IN	OUT
0	1
1	0

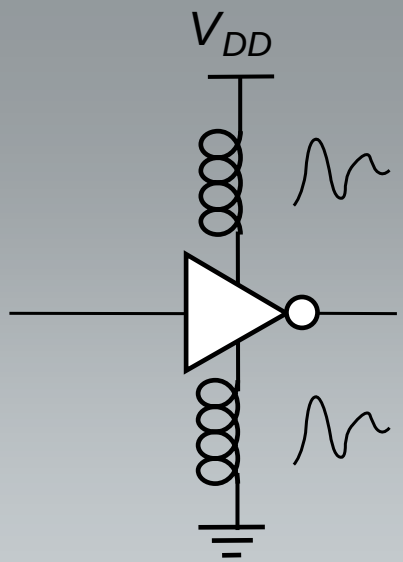


$V_{OH} = f(V_{OL})$
$V_{OL} = f(V_{OH})$
$V_M = f(V_M)$

Example Noise Sources in Digital Circuits



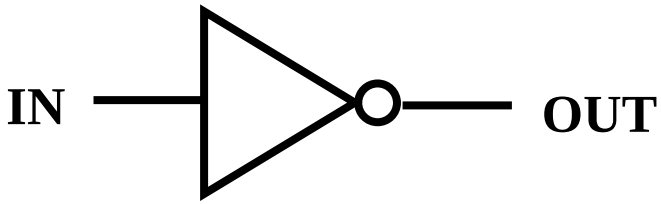
Capacitive coupling



Power and ground noise

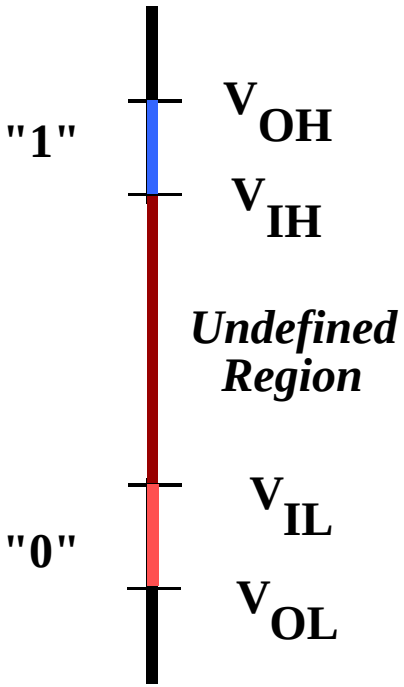
- Noise sources: coupling, cross talk, supply noise, etc.
- Digital circuits must be robust against such noise sources

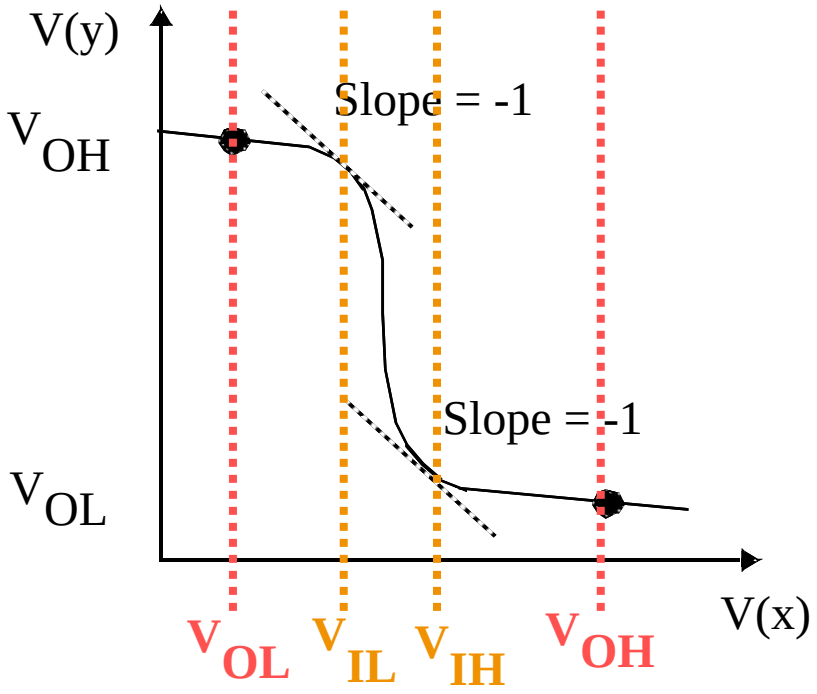
The Inverter: Noise Margin



Truth Table

IN	OUT
0	1
1	0





$$NM_L = V_{IL} - V_{OL}$$
$$NM_H = V_{OH} - V_{IH}$$

- **Large noise margins protect against various noise sources**