

Digital Design: An Embedded Systems Approach Using Verilog

Chapter 1 Introduction and Methodology

Portions of this work are from the book, *Digital Design: An Embedded Systems Approach Using Verilog*, by Peter J. Ashenden, published by Morgan Kaufmann Publishers, Copyright 2007 Elsevier Inc. All rights reserved.

Digital Design

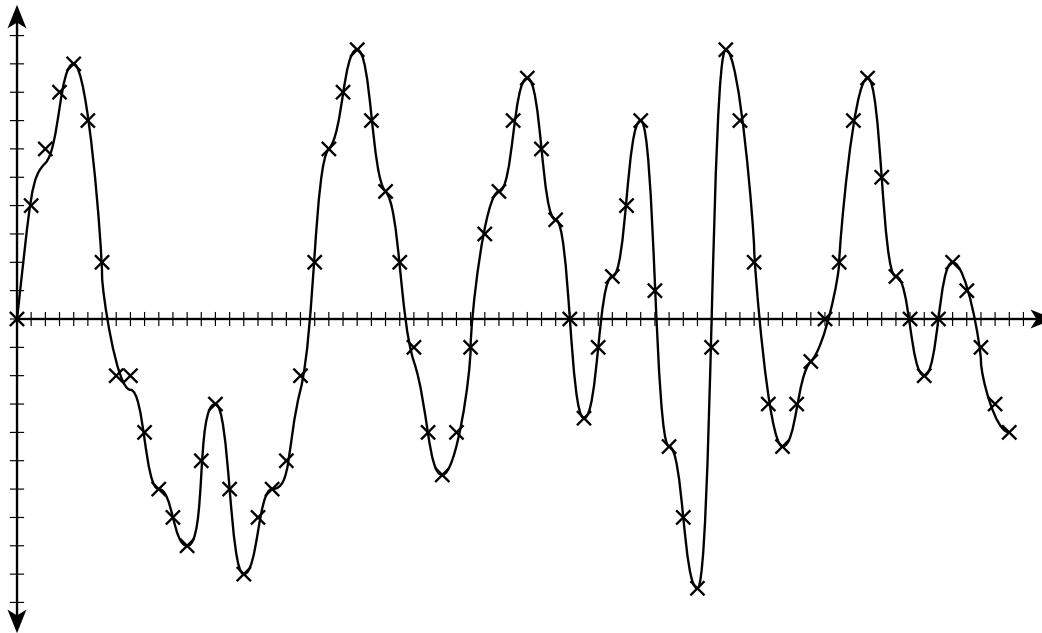
- *Digital*: circuits that use two voltage levels to represent information
 - *Logic*: use truth values and logic to analyze circuits
- *Design*: meeting functional requirements while satisfying constraints
 - Constraints: performance, size, power, cost, etc.

Design using Abstraction

- Circuits contain millions of transistors
 - How can we manage this complexity?
- Abstraction
 - Focus on aspects relevant aspects, ignoring other aspects
 - Don't break assumptions that allow aspect to be ignored!
- Examples:
 - Transistors are on or off
 - Voltages are low or high

Digital Systems

- Electronic circuits that use discrete representations of information
 - Discrete in space and time



Embedded Systems

- Most real-world digital systems include embedded computers
 - Processor cores, memory, I/O
- Different functional requirements can be implemented
 - by the embedded software
 - by special-purpose attached circuits
- Trade-off among cost, performance, power, etc.

Binary Representation

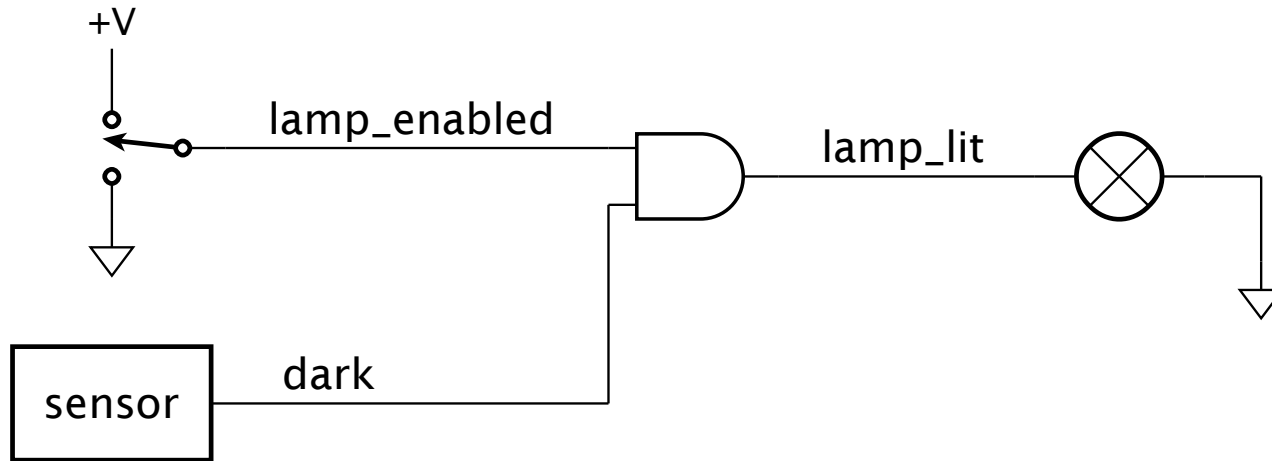
- Basic representation for simplest form of information, with only two states
 - a switch: open or closed
 - a light: on or off
 - a microphone: active or muted
 - a logical proposition: false or true
 - a binary (base 2) digit, or bit: 0 or 1

Binary Representation: Example



- Signal represents the state of the switch
 - high-voltage => pressed,
low-voltage => not pressed
- Equally, it represents state of the lamp
 - lamp_lit = switch_pressed

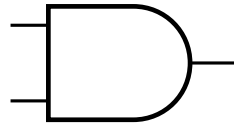
Binary Representation: Example



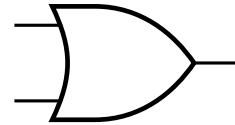
- dark: it's night time
- lamp_enabled: turn on lamp at night
- lamp_lit: lamp_enabled AND dark
- Logically: day time \Rightarrow NOT lamp_lit

Basic Gate Components

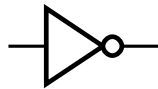
- Primitive components for logic design



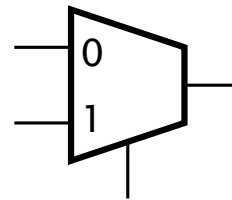
AND gate



OR gate



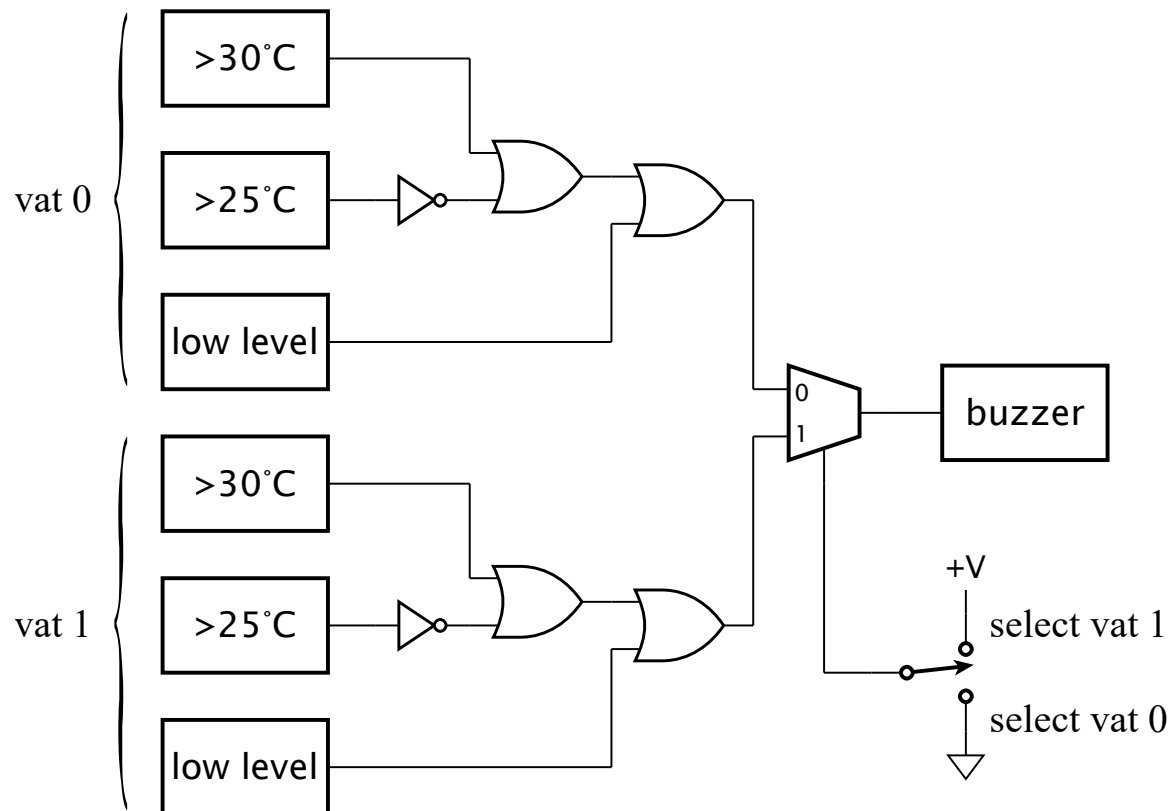
inverter



multiplexer

Combinational Circuits

- Circuit whose output values depend purely on current input values

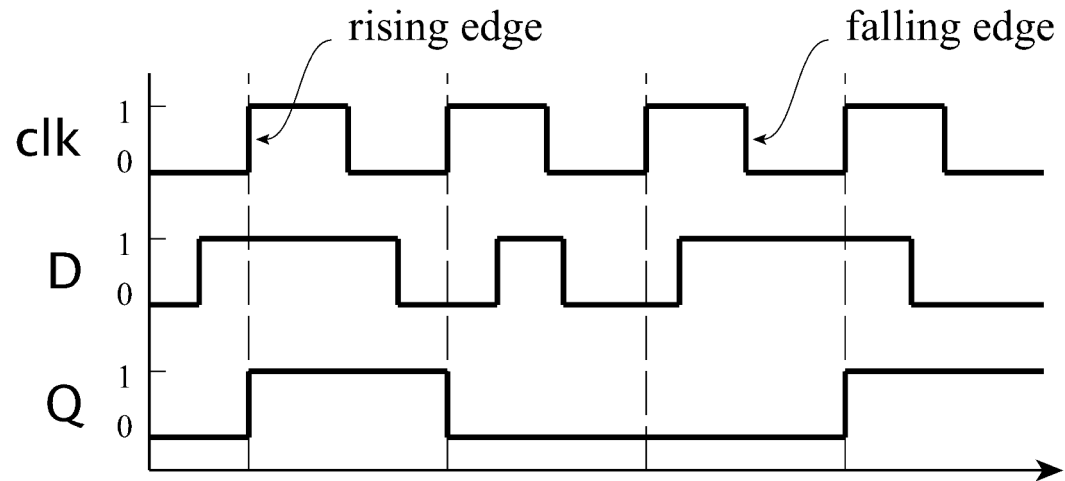
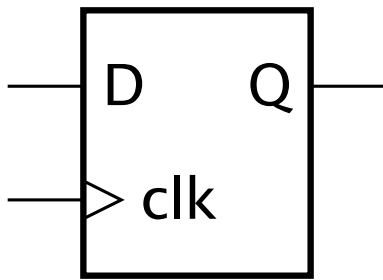


Sequential Circuits

- Circuit whose output values depend on current *and previous* input values
 - Include some form of storage of values
- Nearly all digital systems are sequential
 - Mixture of gates and storage components
 - Combinational parts transform inputs and stored values

Flipflops and Clocks

- Edge-triggered D-flipflop
 - stores one bit of information at a time



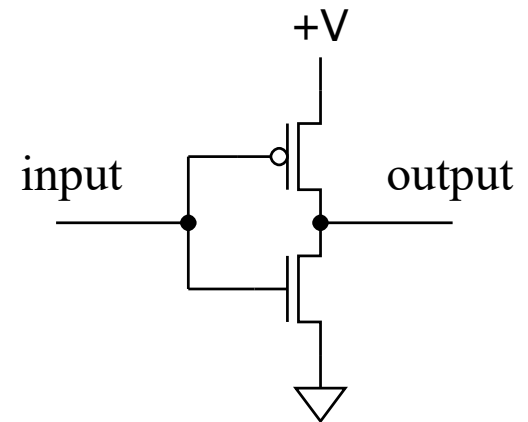
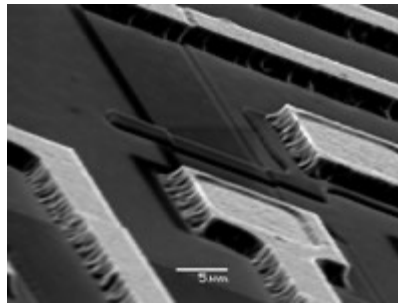
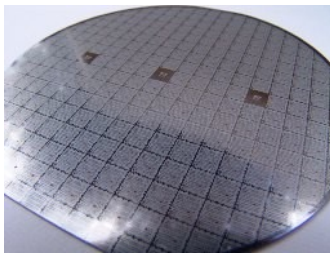
- Timing diagram
 - Graph of signal values versus time

Real-World Circuits

- Assumptions behind digital abstraction
 - ideal circuits, only two voltages, instantaneous transitions, no delay
- Greatly simplify functional design
- Constraints arise from real components and real-world physics
- Meeting constraints ensures circuits are “ideal enough” to support abstractions

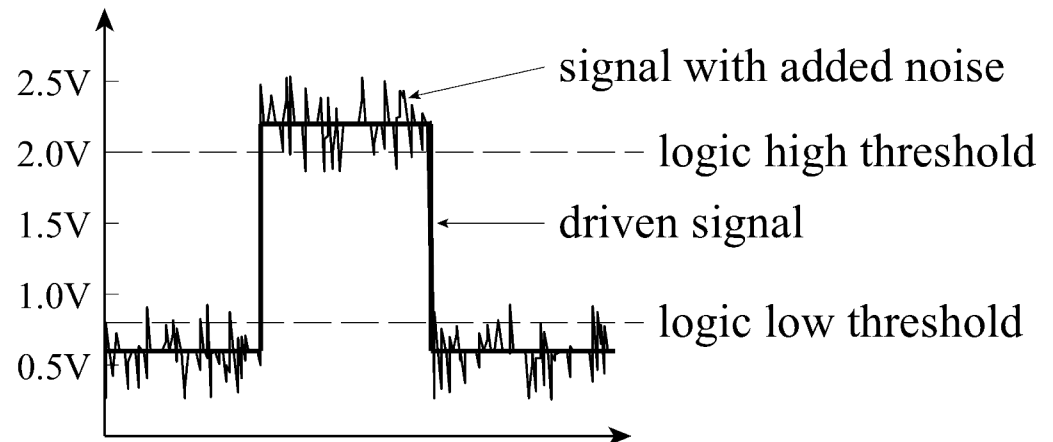
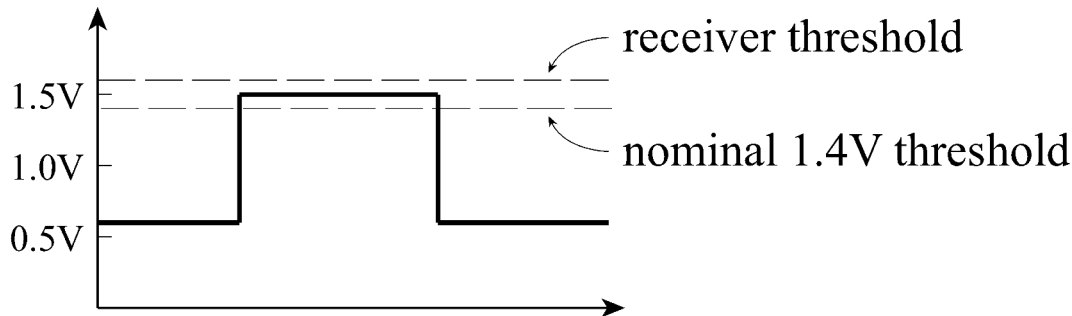
Integrated Circuits (ICs)

- Circuits formed on surface of silicon wafer
 - Minimum feature size reduced in each technology generation
 - Currently 90nm, 65nm
 - Moore's Law: increasing transistor count
 - CMOS: complementary MOSFET circuits



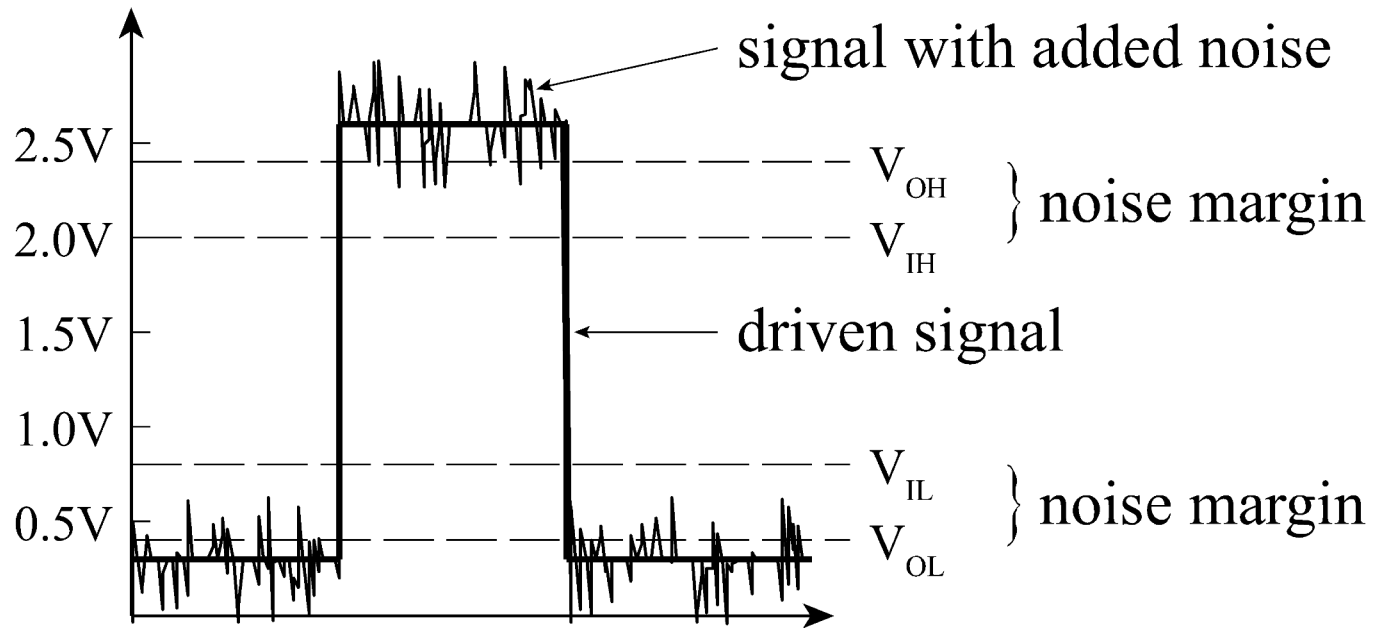
Logic Levels

- Actual voltages for “low” and “high”
 - Example: 1.4V threshold for inputs



Logic Levels

- TTL logic levels with noise margins



V_{OL} : output low voltage

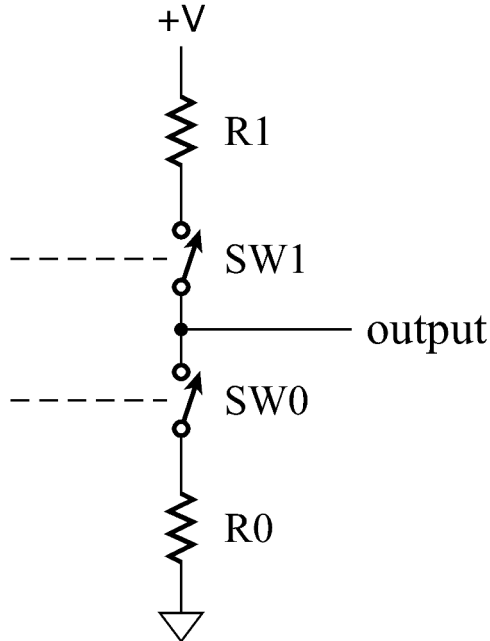
V_{IL} : input low voltage

V_{OH} : output high voltage

V_{IH} : input high voltage

Static Load and Fanout

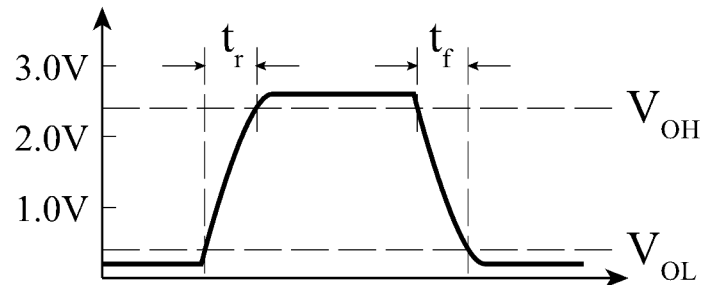
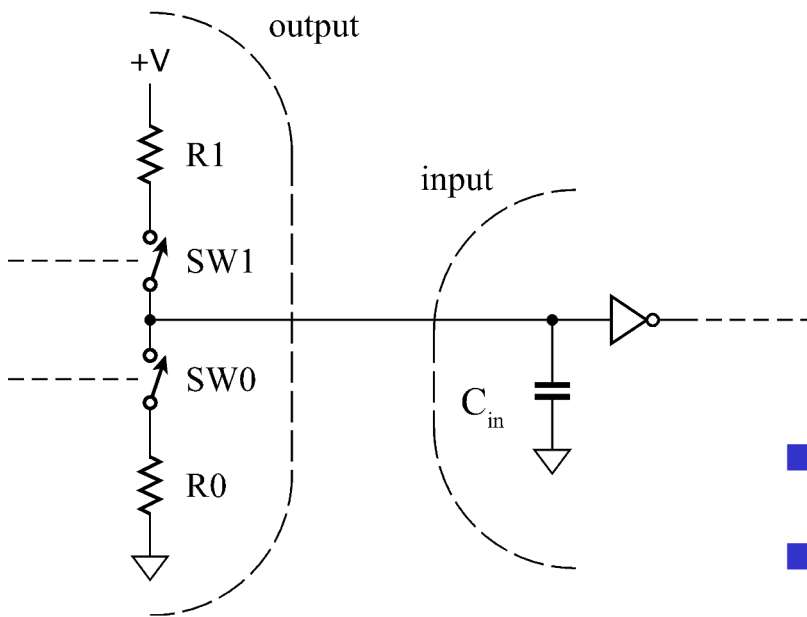
- Current flowing into or out of an output



- High: SW1 closed, SW0 open
 - Voltage drop across R1
 - Too much current: $V_O < V_{OH}$
- Low: SW0 closed, SW1 open
 - Voltage drop across R0
 - Too much current: $V_O > V_{OL}$
- Fanout: number of inputs connected to an output
 - determines static load

Capacitive Load and Prop Delay

- Inputs and wires act as capacitors



- t_r : rise time
- t_f : fall time
- t_{pd} : propagation delay
 - delay from input transition to output transition

Other Constraints

- Wire delay: delay for transition to traverse interconnecting wire
- Flipflop timing
 - delay from clk edge to Q output
 - D stable before and after clk edge
- Power
 - current through resistance => heat
 - must be dissipated, or circuit cooks!

Area and Packaging

- Circuits implemented on silicon chips
 - Larger circuit area => greater cost
- Chips in packages with connecting wires
 - More wires => greater cost
 - Package dissipates heat
- Packages interconnected on a printed circuit board (PCB)
 - Size, shape, cooling, etc, constrained by final product



Models

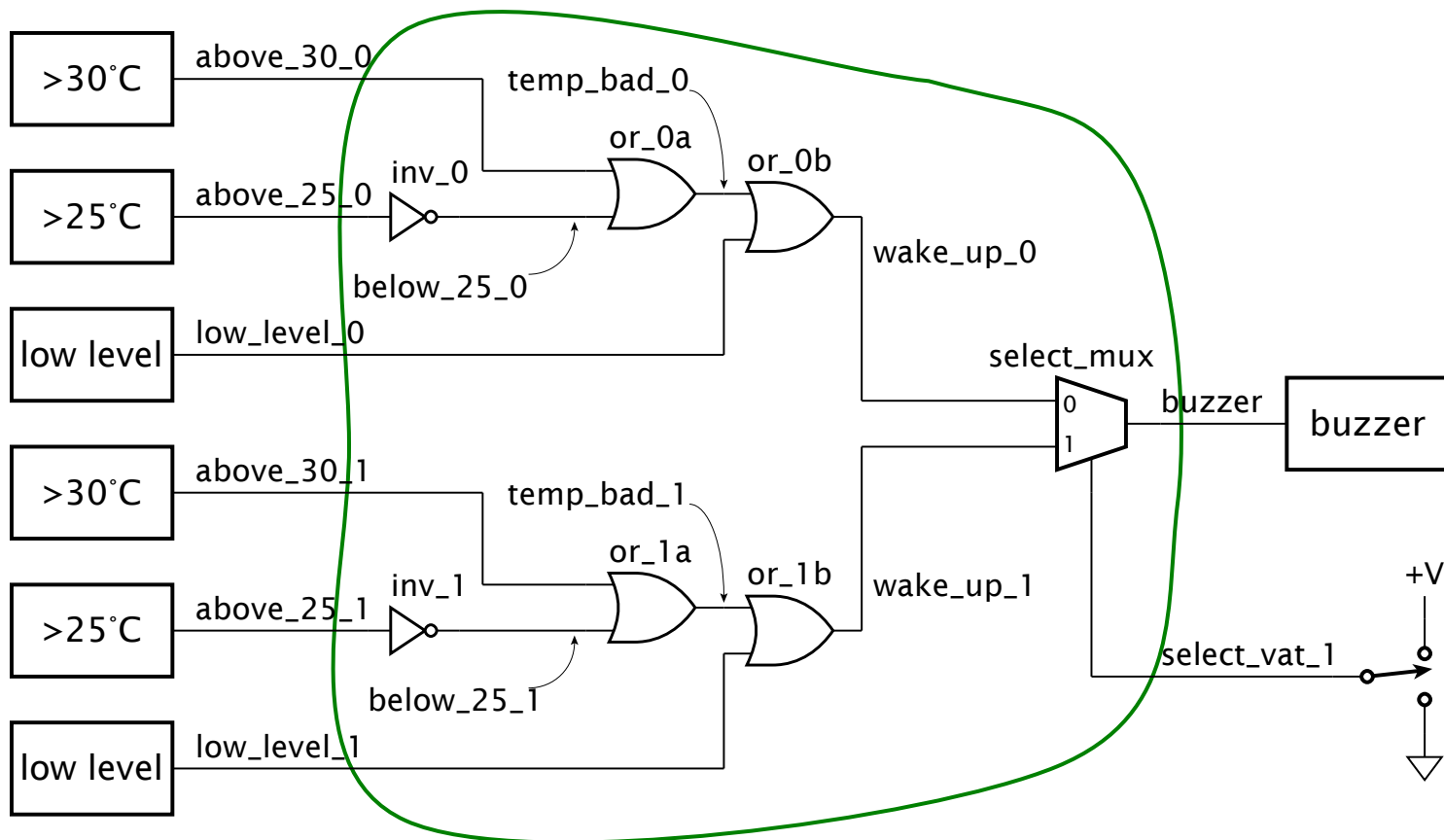
- Abstract representations of aspects of a system being designed
 - Allow us to analyze the system before building it
- Example: Ohm's Law
 - $V = I \times R$
 - Represents electrical aspects of a resistor
 - Expressed as a mathematical equation
 - Ignores thermal, mechanical, materials aspects

Verilog

- Hardware Description Language
 - A computer language for modeling behavior and structure of digital systems
- Electronic Design Automation (EDA) using Verilog
 - Design entry: alternative to schematics
 - Verification: simulation, proof of properties
 - Synthesis: automatic generation of circuits

Module Ports

- Describe input and outputs of a circuit



Structural Module Definition

```
module vat_buzzer_struct
  ( output buzzer,
    input above_25_0, above_30_0, low_level_0,
    input above_25_1, above_30_1, low_level_1,
    input select_vat_1 );

  wire below_25_0, temp_bad_0, wake_up_0;
  wire below_25_1, temp_bad_1, wake_up_1;

  // components for vat 0
  not inv_0 (below_25_0, above_25_0);
  or or_0a (temp_bad_0, above_30_0, below_25_0);
  or or_0b (wake_up_0, temp_bad_0, low_level_0);

  // components for vat 1
  not inv_1 (below_25_1, above_25_1);
  or or_1a (temp_bad_1, above_30_1, below_25_1);
  or or_1b (wake_up_1, temp_bad_1, low_level_1);

  mux2 select_mux (buzzer, select_vat_1, wake_up_0, wake_up_1);
endmodule
```


Behavioral Module Definition

```
module vat_buzzer_struct
  ( output buzzer,
    input above_25_0, above_30_0, low_level_0,
    input above_25_1, above_30_1, low_level_1,
    input select_vat_1 );

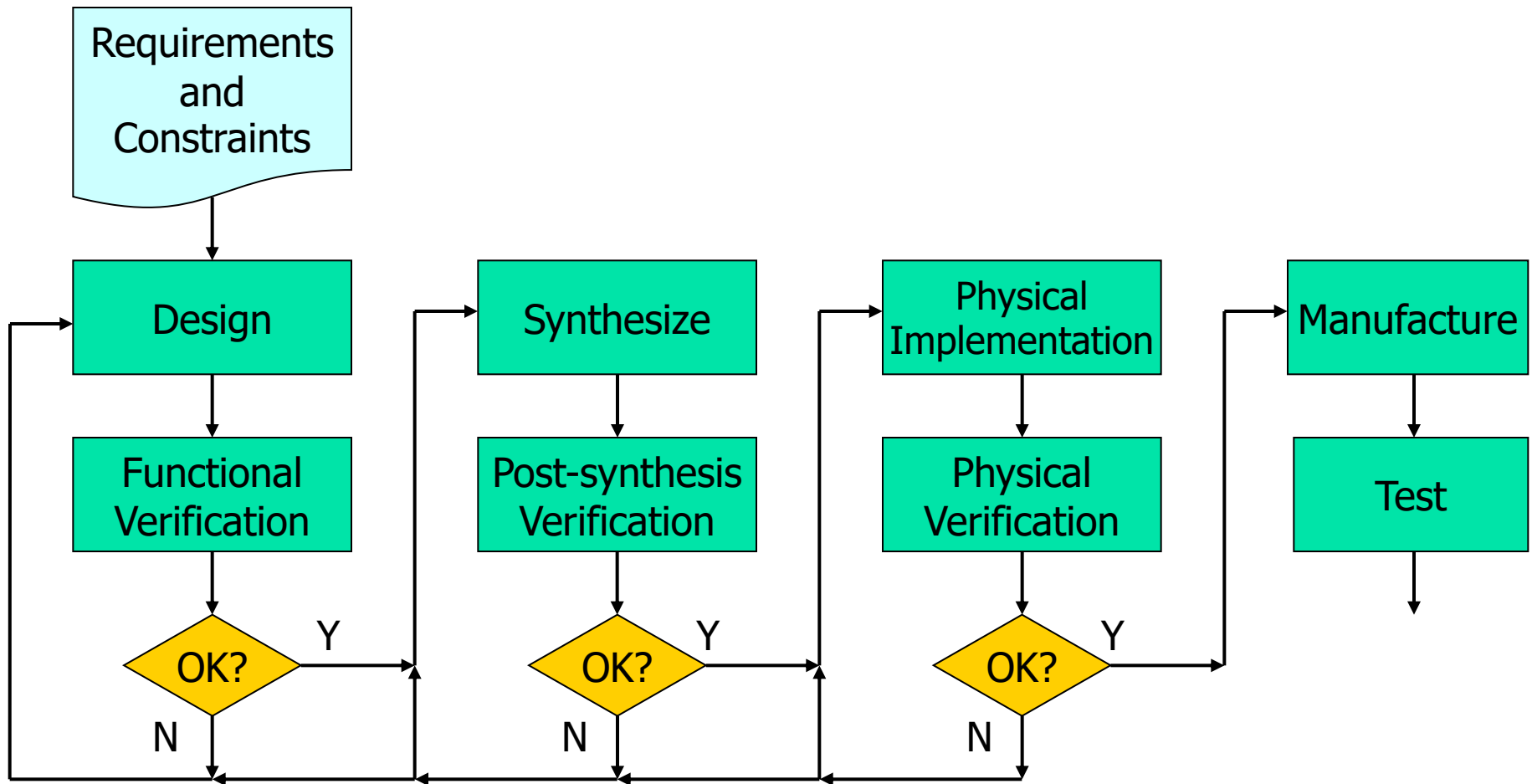
  assign buzzer =
    select_vat_1 ? low_level_1 | (above_30_1 | ~above_25_1)
      : low_level_0 | (above_30_0 | ~above_25_0);

endmodule
```

Design Methodology

- Simple systems can be design by one person using *ad hoc* methods
- Real-world systems are design by teams
 - Require a systematic design methodology
- Specifies
 - Tasks to be undertaken
 - Information needed and produced
 - Relationships between tasks
 - dependencies, sequences
 - EDA tools used

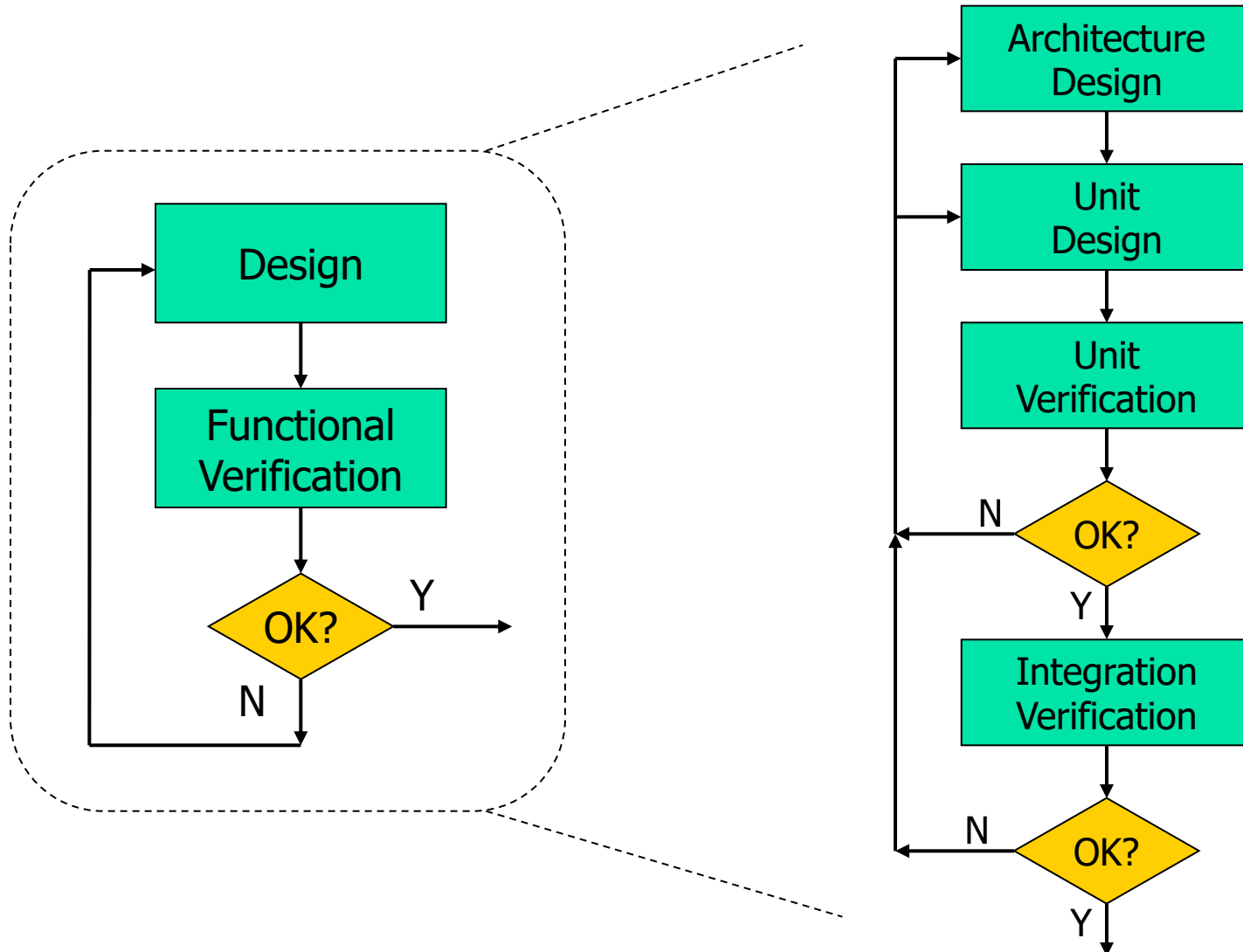
A Simple Design Methodology



Hierarchical Design

- Circuits are too complex for us to design all the detail at once
- Design subsystems for simple functions
- Compose subsystems to form the system
 - Treating subcircuits as “black box” components
 - Verify independently, then verify the composition
- Top-down/bottom-up design

Hierarchical Design



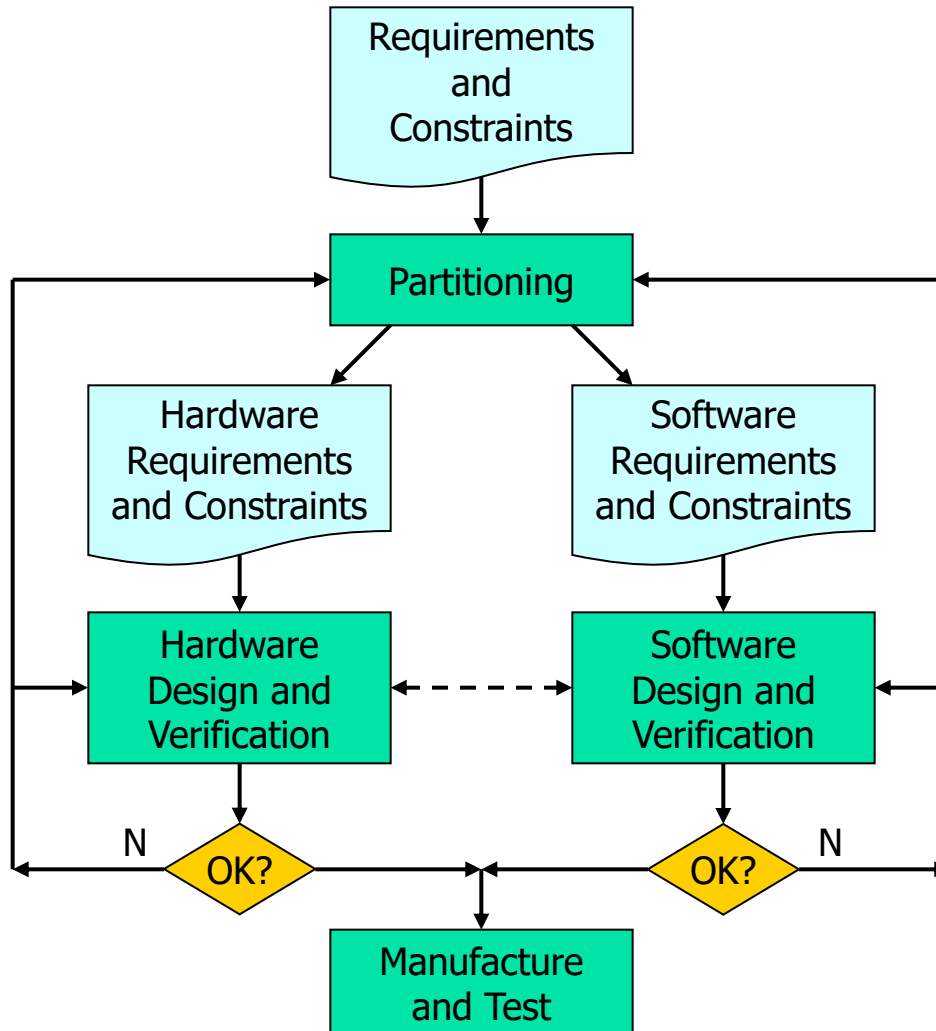
Synthesis

- We usually design using register-transfer-level (RTL) Verilog
 - Higher level of abstraction than gates
- Synthesis tool translates to a circuit of gates that performs the same function
- Specify to the tool
 - the target implementation fabric
 - constraints on timing, area, etc.
- Post-synthesis verification
 - synthesized circuit meets constraints

Physical Implementation

- Implementation fabrics
 - Application-specific ICs (ASICs)
 - Field-programmable gate arrays (FPGAs)
- Floor-planning: arranging the subsystems
- Placement: arranging the gates within subsystems
- Routing: joining the gates with wires
- Physical verification
 - physical circuit still meets constraints
 - use better estimates of delays

Codesign Methodology



Summary

- Digital systems use discrete (binary) representations of information
- Basic components: gates and flipflops
- Combinational and sequential circuits
- Real-world constraints
 - logic levels, loads, timing, area, etc
- Verilog models: structural, behavioral
- Design methodology