שפת תכנון חומרה
ורילוג - Verilog

Verilog – Hierarchical Modeling Concept

Dr. Avihai Aharon

Objectives

- **Understand top-down and bottom-up design methodologies for digital design**
- **Explain differences between modules and module instances in Verilog**
- **Describe four levels of abstraction – behavioral, data flow, gate level, and switch level – to represent the same module**
- **Describe components required for the simulation of a digital design**

Traditional vs. Hardware Description lang.

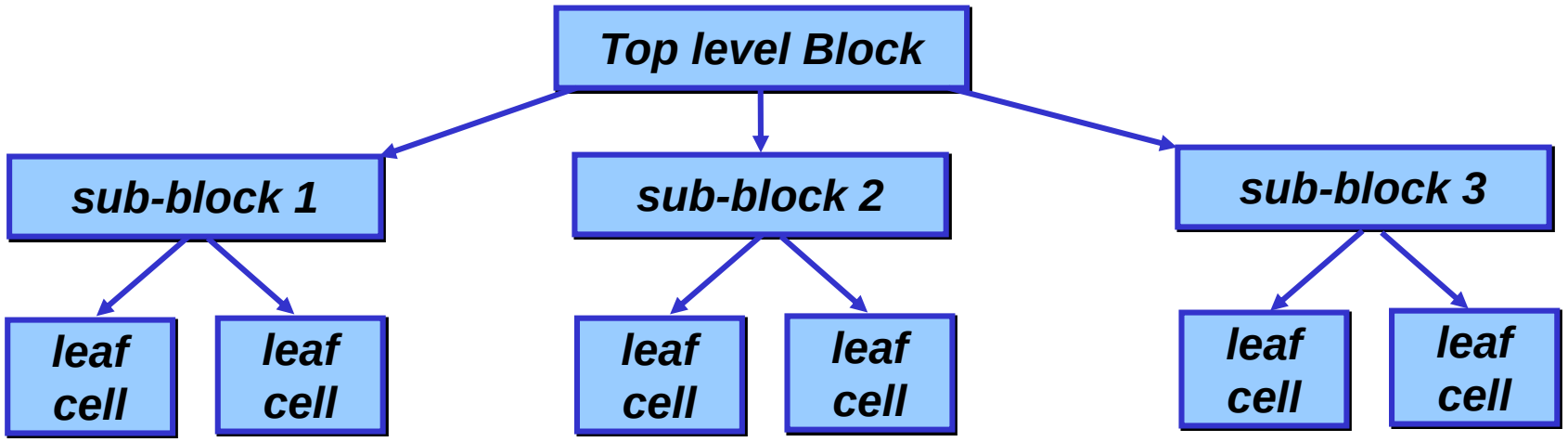
- Procedural programming languages provide the *how* or recipes

  שפות פרוצדורליות כגון C מתארות פרוצדורות לביצוע באופן סדרתי. התכנה מתבצעת (רצה) על מעבד (חומרה גנרית)

- Hardware description languages (VHDL) *describe* a system השפה מתארת חומרה הפועלת באופן מקבילי
  - Systems can be described from many different points of view
    - Behavior: what does it do, i.e., what function does it compute?
    - Structure: what is it composed of?
    - Functional properties: how do I interface to it?
    - Physical properties: how fast is it? How much power does it generate?

Basic Types of Design Methodology

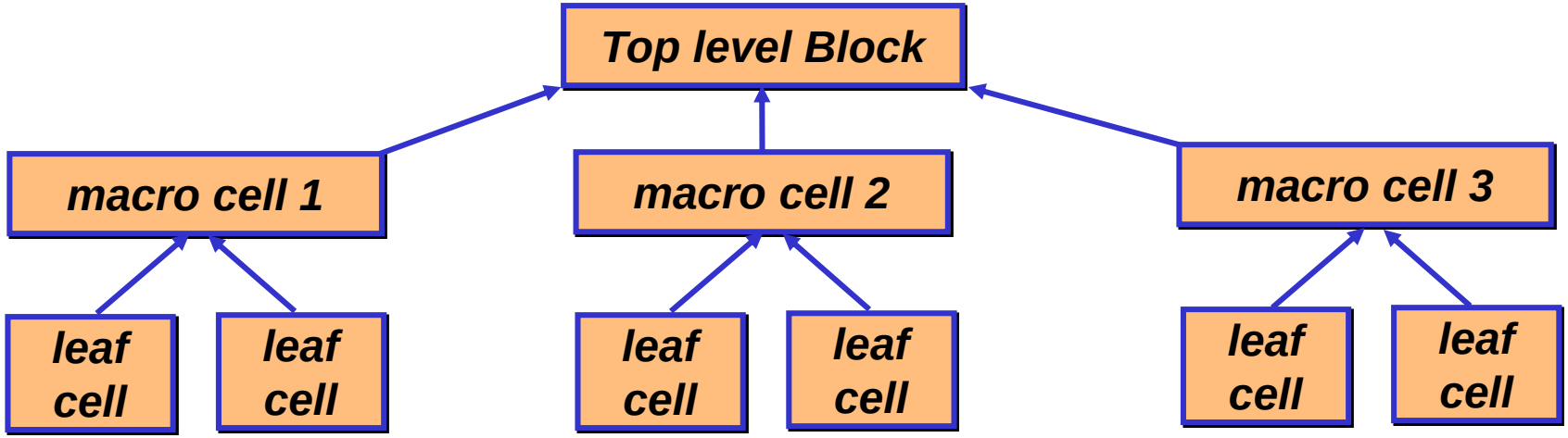
- A **top-down** design methodology



✓ **Define the top-level block and identify the sub-blocks necessary to build the top-level block. Subdivide the sub blocks until will come to leaf cells (can't be divided)**

Basic Types of Design Methodology

- A **bottom-up** design methodology



✓ **Identify the the building blocks that are available and build bigger cells using these building blocks. These cells are then used for higher-level blocks until the top-level block in the design will be built.**

Verilog Hierarchical Modeling Concept

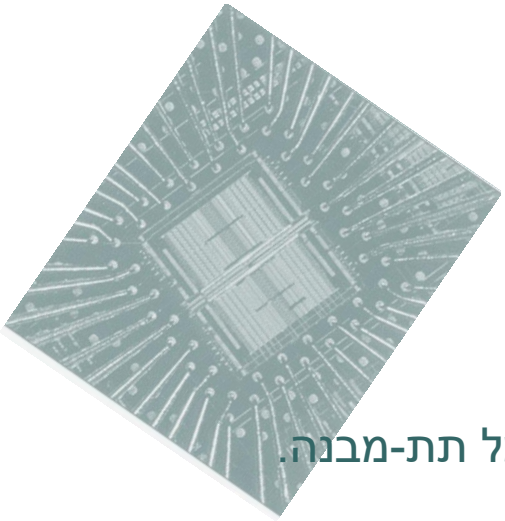
- Verilog is both behavioral and structural language
- Internals of each module can be defined at four levels of abstraction (depending on the needs of the design)
  - ✓ Behavioral or algorithmic level
  - ✓ Structural level
  - ✓ Gate level
  - ✓ Switch level
- The level of abstraction to describe a module can be changed without any change in the environment
- Verilog allows the designer to mix and match all four levels of abstraction in a design

Verilog Hierarchical Modeling Concept

תיאור מבני:

בשיטת התיאור המבני, מפרקים את התכנון כולו ליחידות משנה (הנקראות תתי-בלוקים) קטנות יותר, לפי הפרדה פונקציונאלית. כך מתאפשר לנו לפרק בעיה גדולה וסבוכה, לבעיות קטנות יותר ופשוטות יותר. ניתן להמשיך ולפרק את יחידות המשנה שוב ושוב על מנת לפשט עוד יותר את הבעיה.

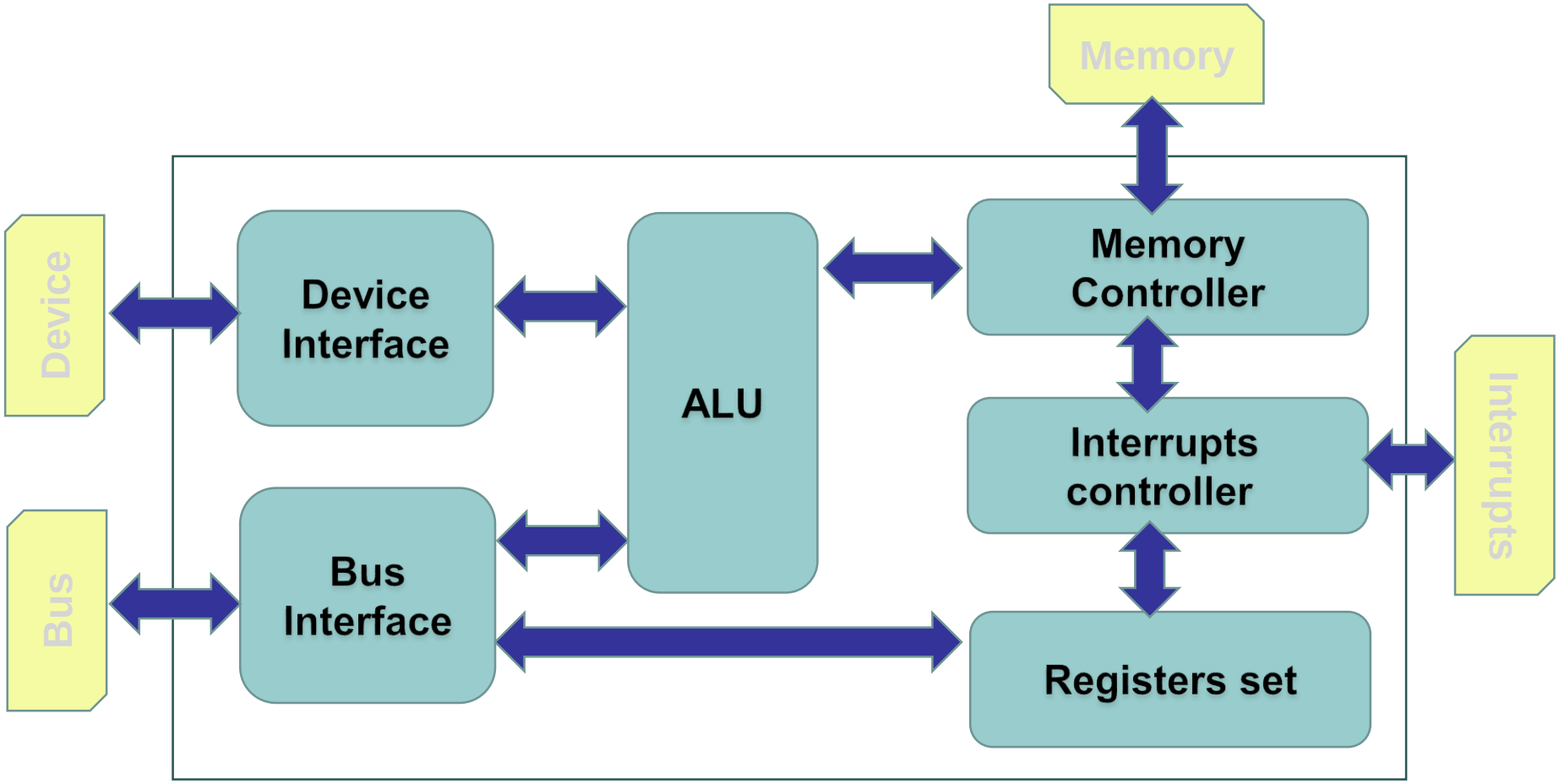
בשיטת תיאור מבני ניתן להשיג את המטרות הבאות:



- פישוט הבעיה למרכיביה.
- אפשרות לחלוקת העבודה לצוות.
- בידוד בעיות ע"י ישויות.
- שימוש חוזר במודולים.
- שיפור יכולת הסימולציה לכל תת-מבנה.
- שיפור ביצוע הסינתזה.

Verilog Hierarchical Modeling Concept

תיאור מבני – דוגמא:



Verilog Hierarchical Modeling Concept

<u>תיאור התנהגותי:</u>

בשיטת זו ניתן לנו האפשרות לתאר את פתרון הבעיה ע"י תיאור פעולות ותנאים המתבצעים ברצף, כמעט ללא תלות בחומרה עצמה.

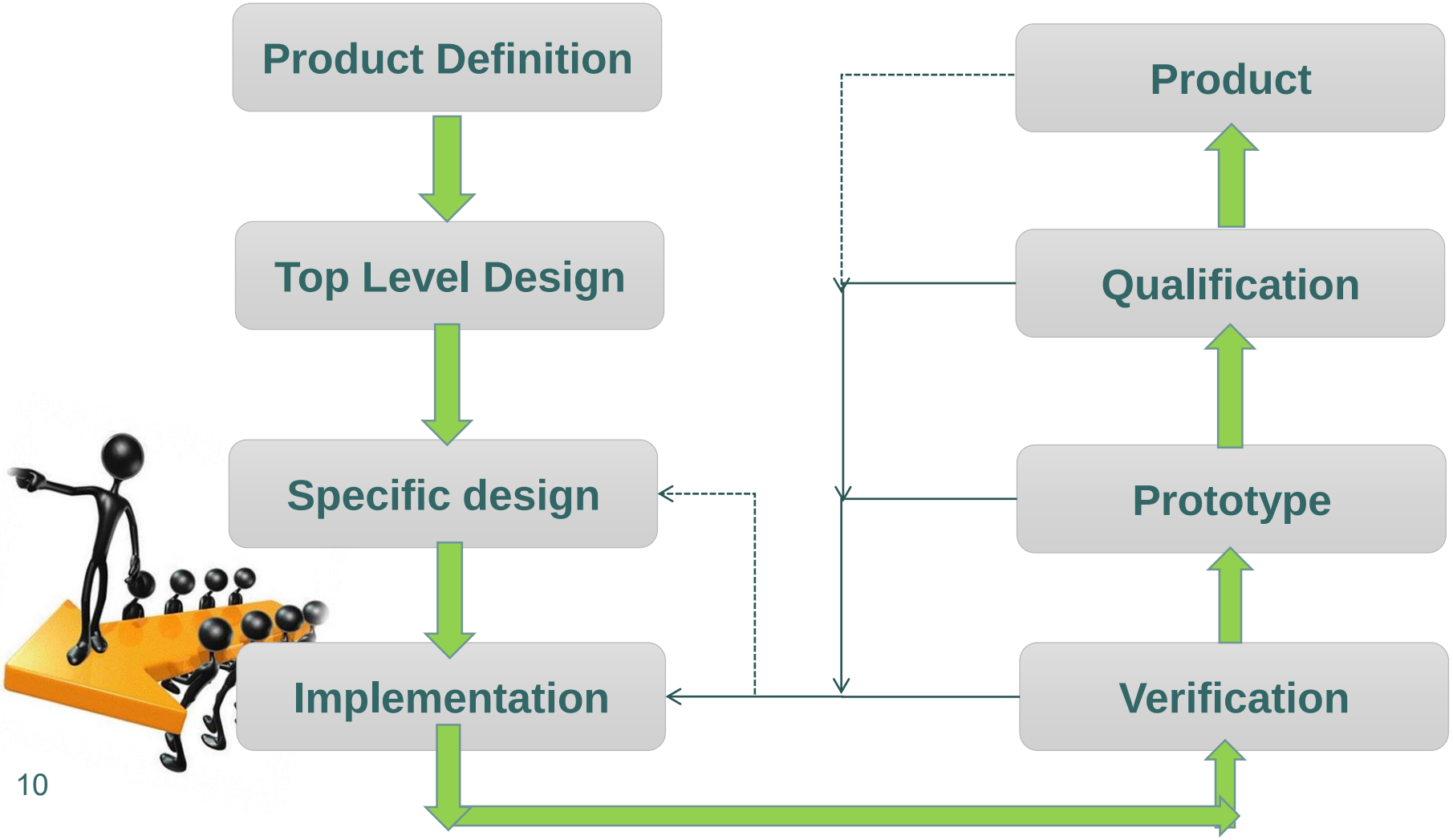
שיטה זו מאפשרת מענה במרחב הבעיה ולא במרחב הפתרון.

כמו כן, שיטה זו יעילה מאד לצרכי סימולציה, כגון התייחסות להשהיות, קריאה מקבצים (כגון זיכרון) ועוד.

בשיטת תיאור התנהגותי מתקבלת בד"כ סימולציה מהירה בהרבה מאשר סימולציה ברמת שערים לוגיים.

Verilog Hierarchical Modeling Concept

Design Process



Basic Types of Design Methodology

- **Typically, a combination of top-down and bottom-up flows is used**
  - ✓ **Design architect define the specifications of the top-level block**
  - ✓ **Logic designers decide how the design should be structured by breaking up the functionality into blocks and sub-blocks**
  - ✓ **Circuit designers are designing optimized circuits for leaf-level cells and build higher-level cells by using these leaf cells**
  - ✓ **The flow meets at intermediate point where the switch-level circuit designers have created a library of leaf cells by using switches, and the logic level designers have designed from top-down until all modules are defined in terms of leaf cells**

Verilog Hierarchical Modeling Concept

- A **module** is a basic building block in Verilog
- A **module** can be element or a collection of low-level design blocks
- Elements are grouped into **modules** to provide common functionality that is used in many places in the design
- A **module** provides the necessary functionality to the higher level blocks through its port interface (inputs and outputs), but hides the internal implementation
- Designer can modify **module** internals without affecting the rest of design

Verilog Abstraction Levels

1. **Behavioral level**
   - ✓ **The highest level of abstraction provided by Verilog HDL**
   - ✓ **A module can be implemented in terms of the desired algorithm without concern for the hardware implementation details**
   - ✓ **Very similar to C language programming**

2. **RTL level**
   - ✓ **Verilog description that is acceptable by synthesis tool.**

Verilog Abstraction Levels

3. **Gate level**
   - ✓ **The module implemented in terms of logic gates and interconnections between these gates**
   - ✓ **Similar to describing a design in terms of gate-level logic diagram**
4. **Switch level**
   - ✓ **The lowest level of abstraction provided by Verilog HDL**
   - ✓ **A module can be implemented in terms of switches, storage nodes, and interconnection between them**
   - ✓ **Design at this level required knowledge of switch level implementation details**

Verilog Abstraction Levels

- **The higher (behavioral) level of abstraction makes the design more flexible and technology independent**
- **As design goes lower toward switch-level, its becomes technology dependent and inflexible**

Instances in Verilog

- A module provide a template from which you can create actual objects
- When a module is invoked, Verilog creates a unique object from the template
- Each object has it’s own name, variables, parameters and I/O interface
- The process of creating objects from a module template is called instantiation, and the objects are called instances

Instances in Verilog (example)

```
// Define the top-level module called ripple carry counter.
// It instantiates 4 T-flip-flops.
module ripple_carry_counter (q, clk, reset)

input clk, reset;    // I/O signals (explained later)
output [3:0] q;      // I/O signals and vector explanation

/*   Four instances of the module T_FF are created. Each has  a unique
     name. Each instance is passed a set of signals. Notice, that each
     instance is a copy of the module T_FF.  */
 T_FF    tff_0 (q[0],clk,reset);
 T_FF    tff_1 (q[1],q[0],reset);
 T_FF    tff_2 (q[2],q[1],reset);
 T_FF    tff_3 (q[3],q[2],reset);

endmodule
```

Simulation of Digital Systems

- **What do you do to test a software program you write?**
  - ✓ **Give it some inputs, and see if it does what you expect**

- **Simulation tests a model of the system you wish to build**

Simulation of Digital Systems

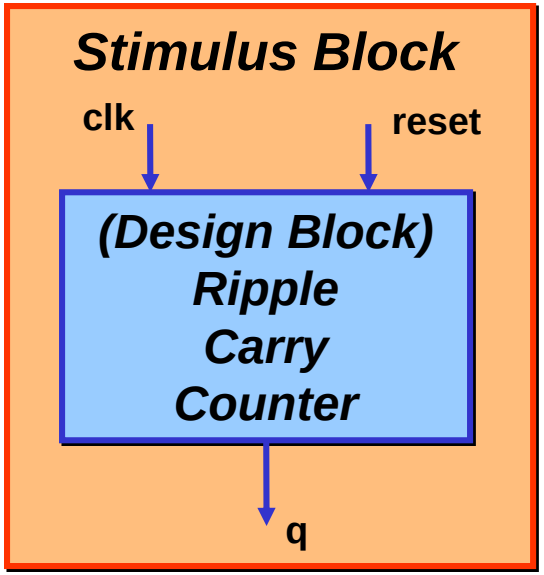
- **Simulation checks two properties**
  - ✓ **functional correctness - is the logic correct ?**
  - ✓ **timing correctness - is the logic/interconnect timing correct (e.g. are the set-up times met)?**

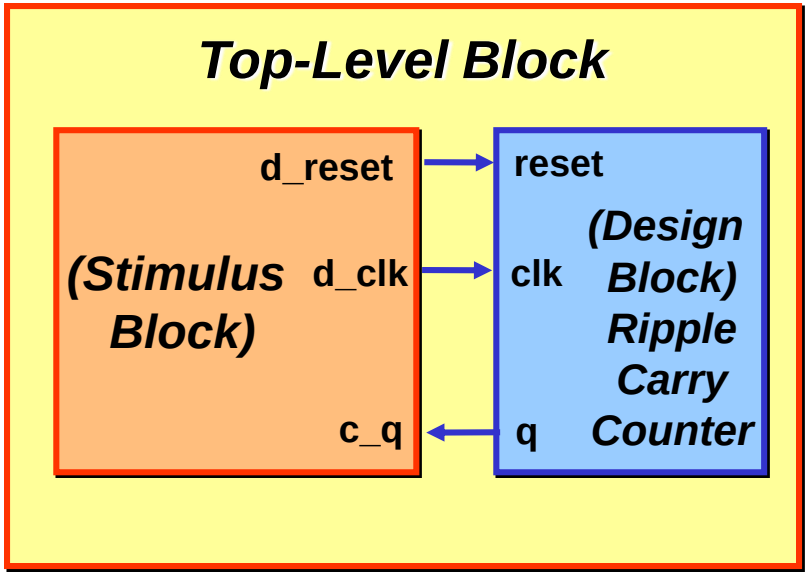
- **It has all the limitations of software testing**
  - ✓ **Have I tried all the cases?**
  - ✓ **Have I exercised every path and every option?**

Components of Simulation

- **Once a design block is completed – it must be tested**
- **The functionality of design block can be tested by applying stimulus and checking results**
- **The stimulus block (also commonly called a testbench) can be written in Verilog**
- **Two styles of stimulus application are possible:**
  - ✓ **The stimulus block instantiates the design block and directly drives the signals in the design block**
  - ✓ **Instantiate both the stimulus and design blocks in a top-level dummy model**
- **It is good practice to keep the stimulus and design blocks separate**

Two Styles of Stimulus Application





- The stimulus block instantiates the design block and directly drives the signals in the design block
- Instantiate both the stimulus and design blocks in a top-level dummy model

Hierarchical Modeling Concept Summary

- **Two kinds of design methodologies are used for digital design: top-down and bottom-up. A combination of these two methodologies is used in today's digital design. As designs become very complex, it is important to follow these structure approaches to manage the design process.**

- **Modules are the basic building blocks in Verilog. Modules are used in a design by instantiation. An instance of a module has a unique identity and is different from other instances of the same module. Each instance has an independent copy of the internals of the module. It is important to understand the difference between modules and instances.**

- **There are two distinct components in a simulation: a design block and a stimulus block. A stimulus block (usually the top-level block) is used to test the design block. There are two different styles of applying stimulus to a design block.**