Islamic University of Technology

EEE 4483
Digital Electronics & Pulse Techniques

Lecture- 6

Introduction to Logic Circuits & Logic Design with VHDL (1st Edition)

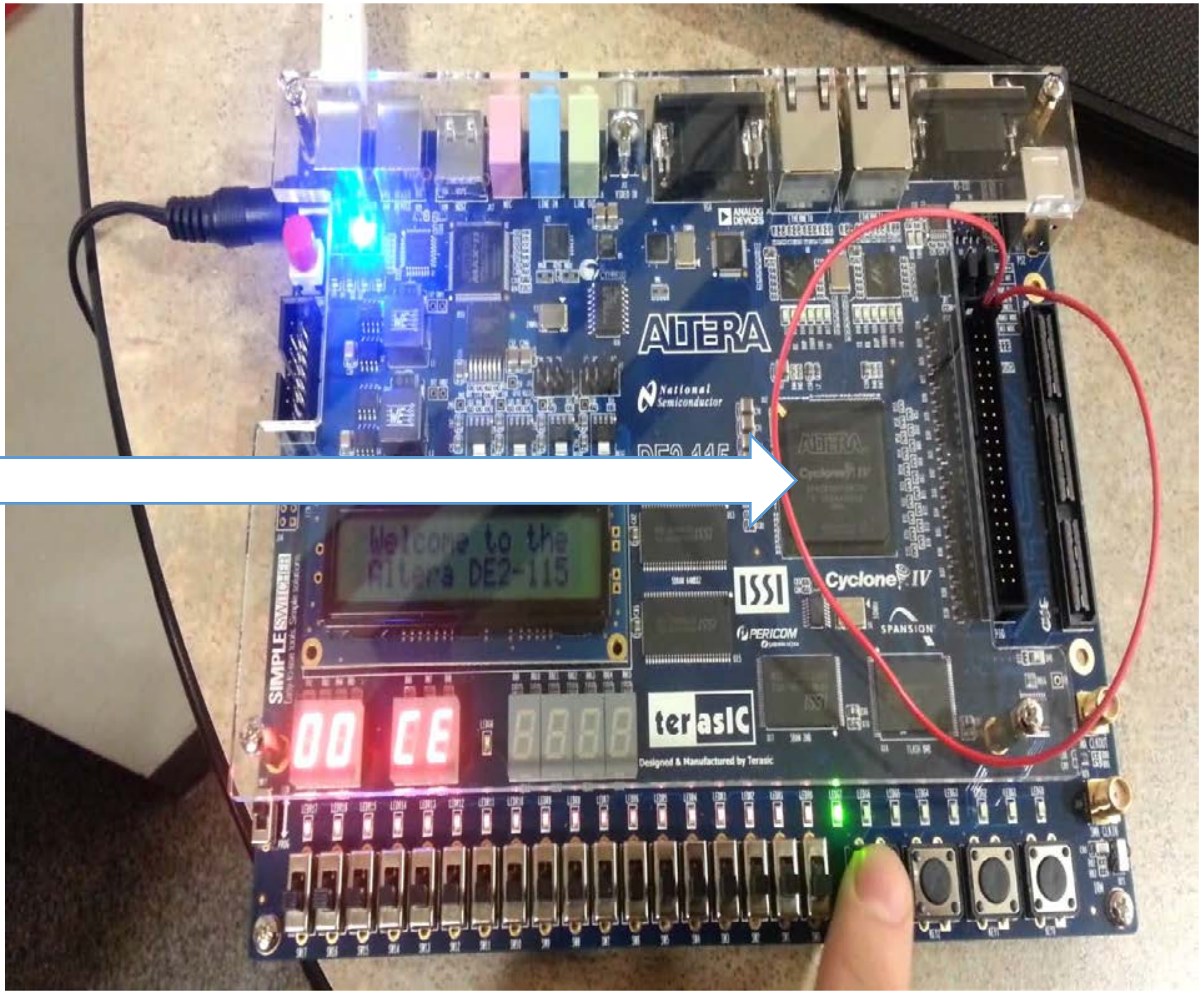
by Brock J. LaMeres

- **Page No -** 143
- **Article No** – 5.2, 5.4, 5.4.1, 5.4.2, 5.4.3, 5.4.4, 5.5, 5.5.1, 5.5.2, 5.5.3

Objectives

- Quick introduction to VHDL
  - basic language concepts
  - basic design methodology
  - examples

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_signed.all;
use ieee.numeric_std.all;


entity HALF_ADDER is
  port(
    A    :       in    std_logic;
    B    :       in    std_logic;
    sum  :       out   std_logic;
    carry:       out   std_logic
);
end HALF_ADDER;

architecture Behavioral of HALF_ADDER is
begin
    SUM   <= A xor B;
    CARRY <= A and B;
end Behavioral;
```



What does VHDL stand for ?

- VHSIC Hardware Description Language

- VHSIC:
  Very High Speed Integrated Circuits

VHDL is a programming language that allows one to model and develop complex digital systems in a dynamic environment.

VHDL --

- VHDL is a programming language that allows one to model and develop complex digital systems in a dynamic envirornment.
- Object Oriented methodology for you C people can be observed -- modules can be used and reused.
- Allows you to designate in/out ports (bits) and specify behavior or response of the system.

Modeling of Digital System

- VHDL is for coding models of a digital system...
- Reasons for modeling
  - requirements specification
  - documentation
  - testing using simulation
  - formal verification
  - synthesis
  - class assignments
- Goal
  - most ‘reliable’ design process, with minimum cost and time
  - avoid design errors!

Basic VHDL Concept

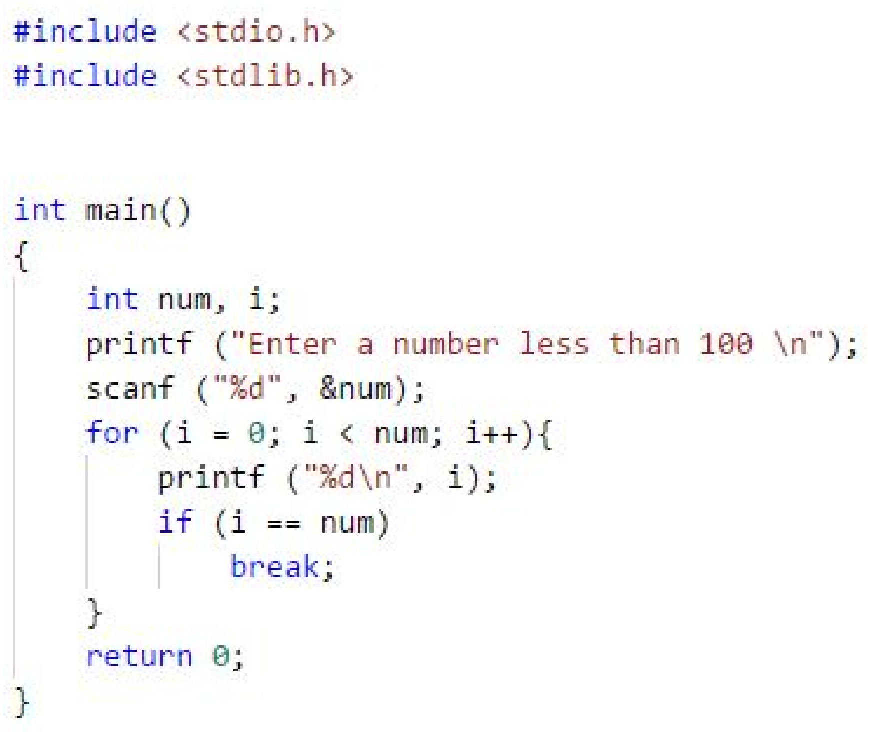
- Interfaces -- i.e. ports
- Behavior
- Structure
- Test Benches
- Analysis, simulation
- Synthesis

HDLs vs. Software Languages

Concurrent (parallel) Statements

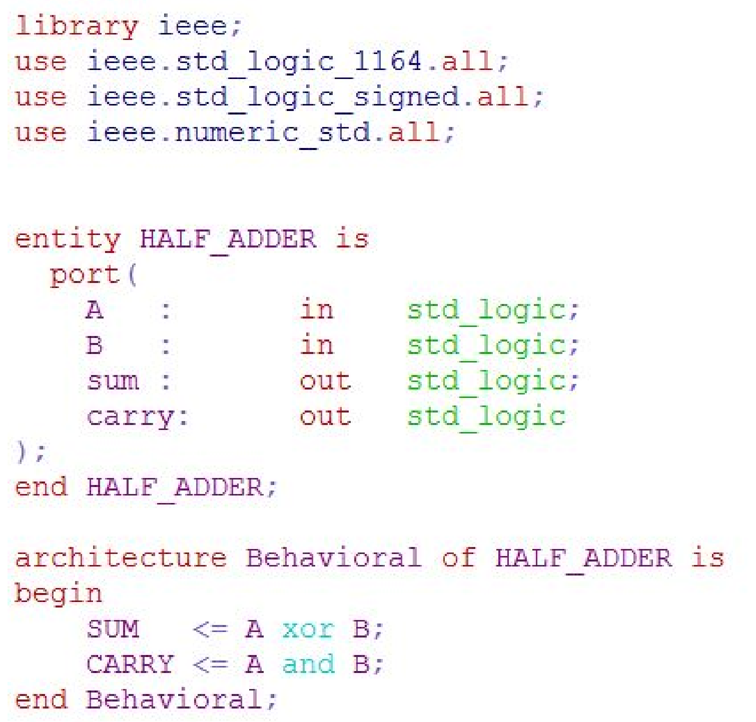
vs.

Sequential Statements



```c
#include <stdio.h>
#include <stdlib.h>


int main()
{
    int num, i;
    printf ("Enter a number less than 100 \n");
    scanf ("%d", &num);
    for (i = 0; i < num; i++){
        printf ("%d\n", i);
        if (i == num)
            break;
    }
    return 0;
}
```

C-code



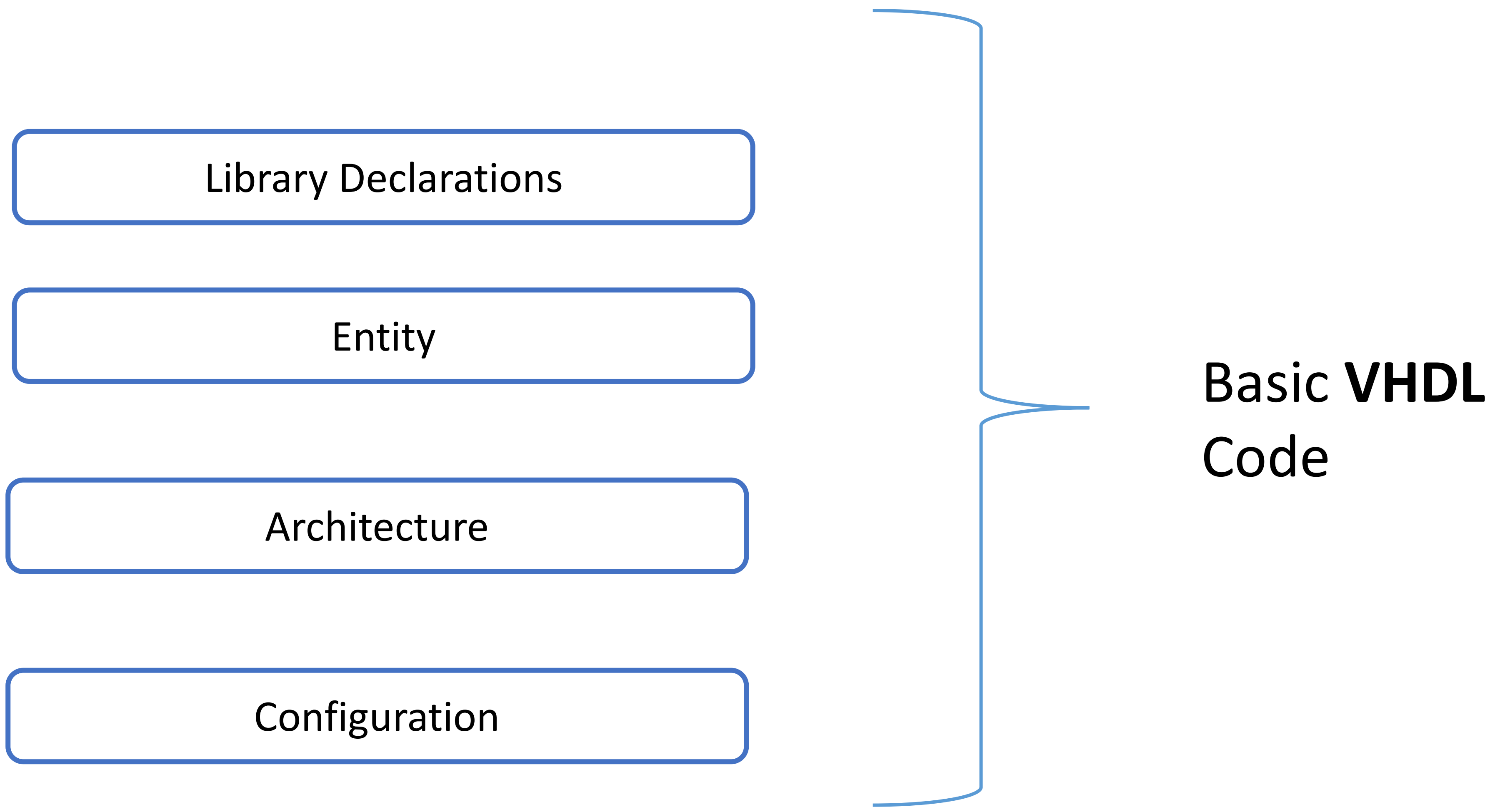
```vhdl
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_signed.all;
use ieee.numeric_std.all;


entity HALF_ADDER is
  port(
    A   :       in    std_logic;
    B   :       in    std_logic;
    sum :       out   std_logic;
    carry:      out   std_logic
);
end HALF_ADDER;

architecture Behavioral of HALF_ADDER is
begin
    SUM   <= A xor B;
    CARRY <= A and B;
end Behavioral;
```

VHDL-code

Basic VHDL Concepts

- Ports
- Behavior
- Structure
- Testbenches
- Simulation
- Synthesis

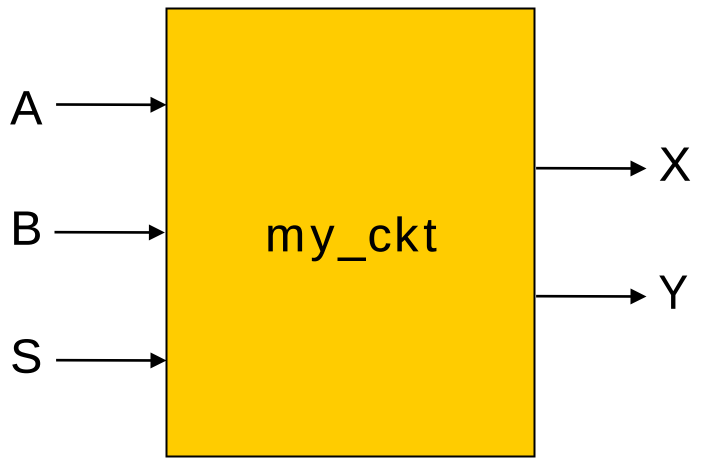
Anatomy of a VHDL code



VHDL Libraries

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_textio.all;
use IEEE.std_logic_arith.all;
use IEEE.numeric_bit.all;
use IEEE.numeric_std.all;
use IEEE.std_logic_signed.all;
use IEEE.std_logic_unsigned.all;
use IEEE.math_real.all;
use IEEE.math_complex.all;
```

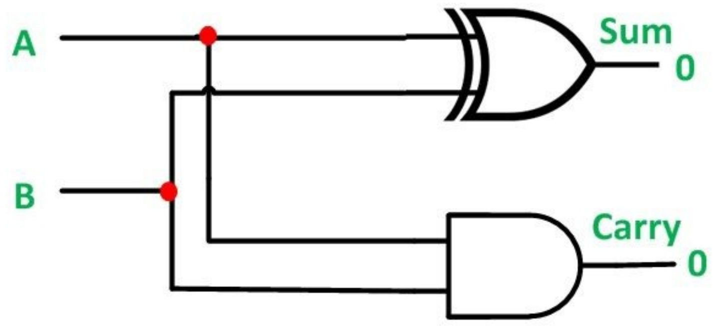
Input-Output specification of a circuit



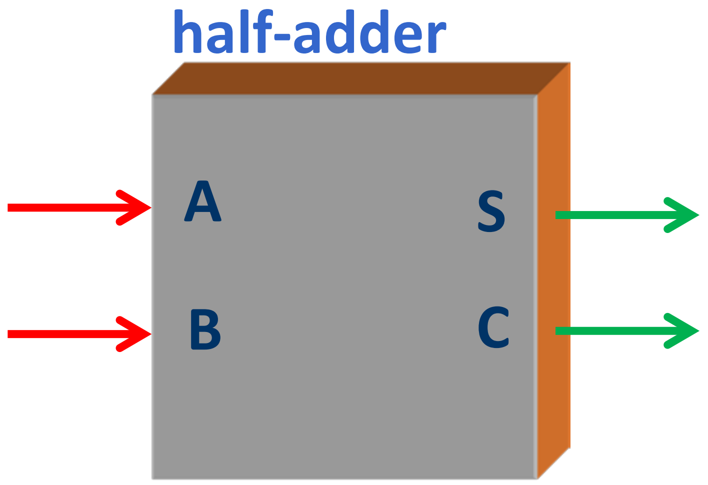
- Example: my_ckt
  - Inputs: A, B, C
  - Outputs: X, Y
- VHDL description:

```
entity my_ckt is
port (
        A: in bit;
        B: in bit;
        S: in bit;
        X: out bit;
        Y: out bit);
end my_ckt ;
```

Entity



2 input and 2 output ports



```
entity HALF_ADDER is
port (
        A: in std_logic;
        B: in std_logic;
        S: out std_logic;
        C: out std_logic
    )
end HALF_ADDER;
```

bit

std_logic

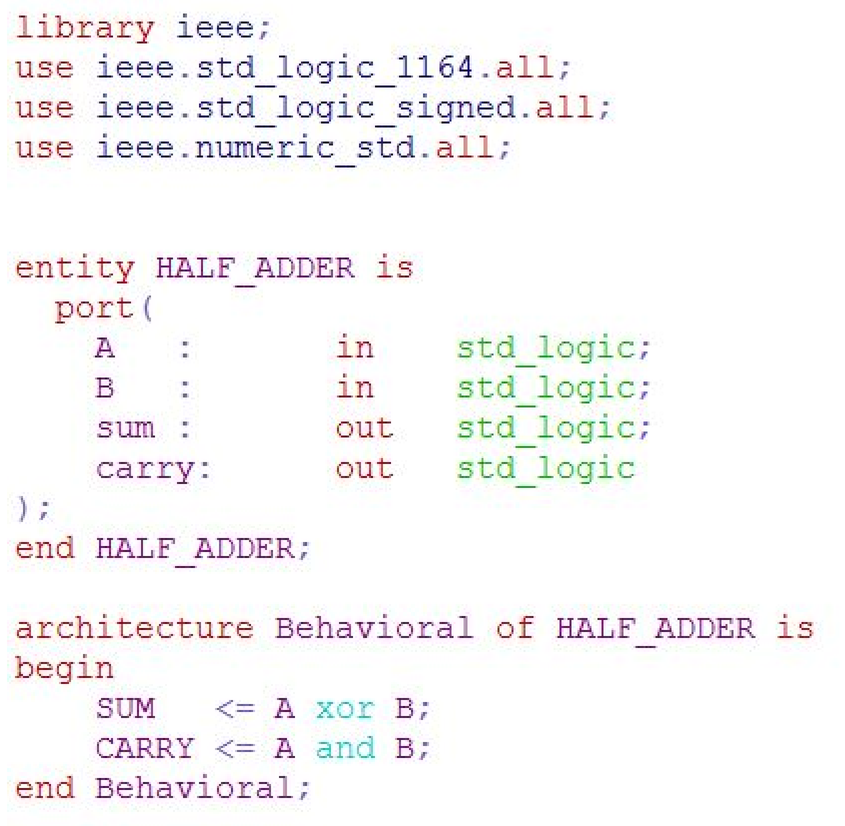
bit is a predefined type and only can only have the value 0 or 1. The bit type is an idealized value.

```
type Bit is ('0', '1');
```

std_logic is part of the std_logic_1164 package and provides more realistic modeling

```
TYPE std_ulogic IS (
         '0',   -- Forcing  0
         '1',   -- Forcing  1
         'Z',   -- High Impedance

       // total nine different values
possible
);
```

Jumping right in to a Model…..



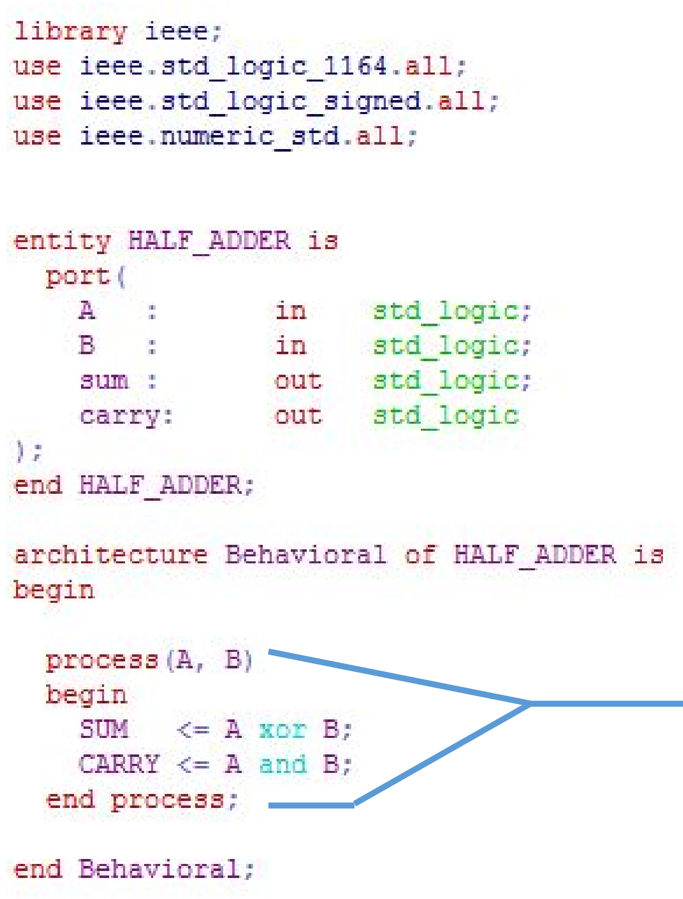
```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_signed.all;
use ieee.numeric_std.all;


entity HALF_ADDER is
  port(
    A   :       in    std_logic;
    B   :       in    std_logic;
    sum :       out   std_logic;
    carry:      out   std_logic
);
end HALF_ADDER;

architecture Behavioral of HALF_ADDER is
begin
    SUM   <= A xor B;
    CARRY <= A and B;
end Behavioral;
```

Library Declarations

Entity

Architecture (main code section)

Making it **Sequential**



```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_signed.all;
use ieee.numeric_std.all;


entity HALF_ADDER is
  port(
    A   :       in    std_logic;
    B   :       in    std_logic;
    sum :       out   std_logic;
    carry:      out   std_logic
);
end HALF_ADDER;

architecture Behavioral of HALF_ADDER is
begin

  process(A, B)
  begin
    SUM   <= A xor B;
    CARRY <= A and B;
  end process;

end Behavioral;
```

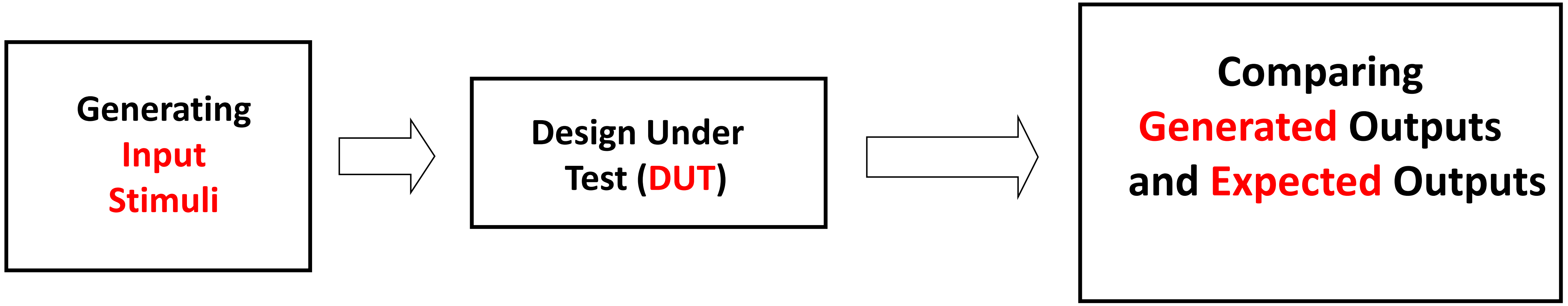
The *sensitivity list* is a compact way of specifying the set of signals, events on which may resume a **process**. A sensitivity list is specified right after the keyword **process**

VHDL Testbench

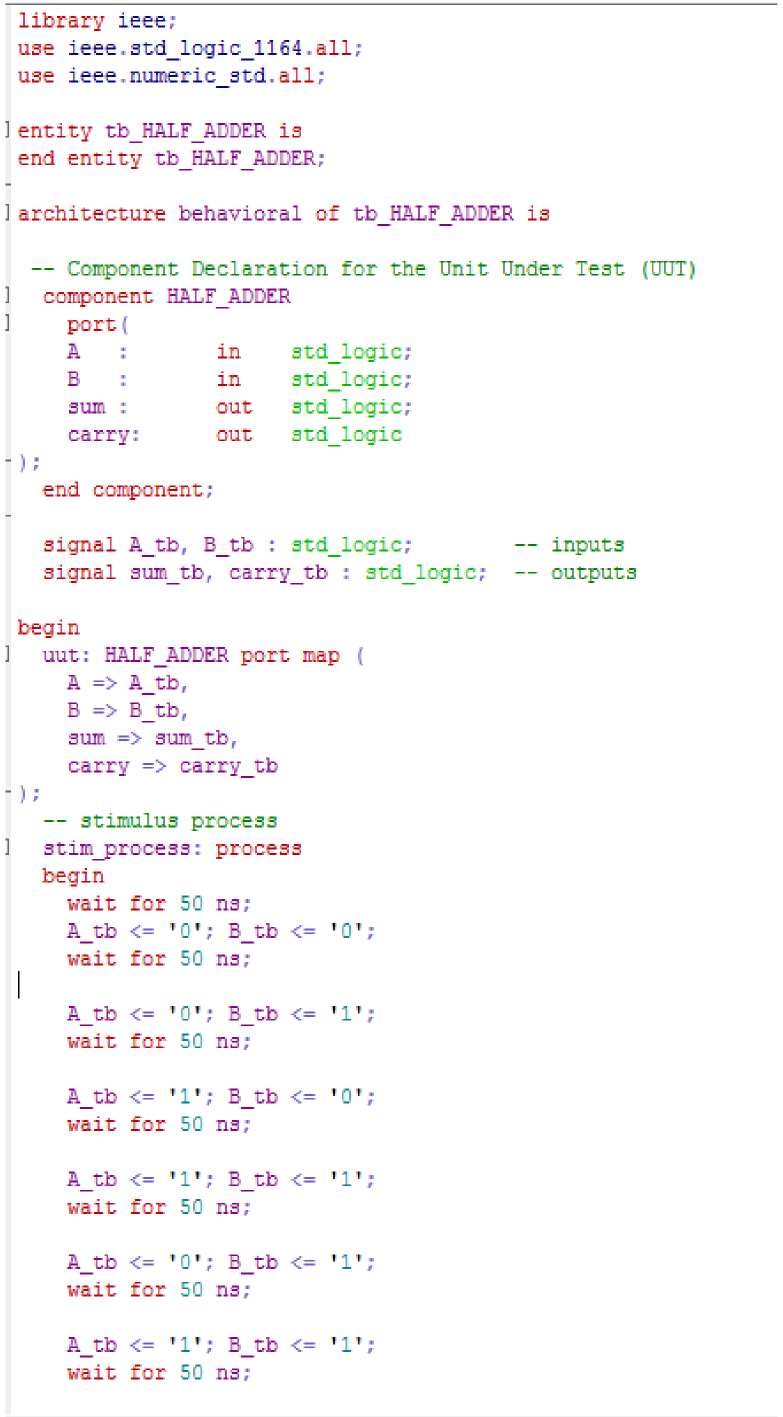
Test Bench is a program that verifies the functional correctness of the hardware design.

The **test bench** program checks whether the hardware model does what it is **supposed to do** and is not doing what it is not supposed to do.

Main functions of a Testbench



Dissection of VHDL testbench



```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity tb_HALF_ADDER is
end entity tb_HALF_ADDER;

architecture behavioral of tb_HALF_ADDER is

 -- Component Declaration for the Unit Under Test (UUT)
  component HALF_ADDER
    port(
    A   :      in    std_logic;
    B   :      in    std_logic;
    sum :      out   std_logic;
    carry:     out   std_logic
);
  end component;

  signal A_tb, B_tb : std_logic;         -- inputs
  signal sum_tb, carry_tb : std_logic;   -- outputs

begin
  uut: HALF_ADDER port map (
    A => A_tb,
    B => B_tb,
    sum => sum_tb,
    carry => carry_tb
);
  -- stimulus process
  stim_process: process
  begin
    wait for 50 ns;
    A_tb <= '0'; B_tb <= '0';
    wait for 50 ns;

    A_tb <= '0'; B_tb <= '1';
    wait for 50 ns;

    A_tb <= '1'; B_tb <= '0';
    wait for 50 ns;

    A_tb <= '1'; B_tb <= '1';
    wait for 50 ns;

    A_tb <= '0'; B_tb <= '1';
    wait for 50 ns;

    A_tb <= '1'; B_tb <= '1';
    wait for 50 ns;
```

No ports this time

Half-adder Component

Input and Output internal signals

Mapping ports with the signals of Unit Under Test (UUT)

Providing the stimuli wrapped in a process