Universidade Federal de Santa Catarina
Centro Tecnológico – CTC
Departamento de Engenharia Elétrica

“Circuitos e Técnicas Digitais”

Prof. Eduardo Augusto Bezerra

Eduardo.Bezerra@ufsc.br

Florianópolis, agosto de 2015.

Plano de Aula

"Projeto de Sistemas Digitais com VHDL"

- Objetivos:
  - Apresentar uma visão geral de VHDL
  - Exemplo de descrição VHDL
  - Introdução ao Quartus II – ferramentas de desenvolvimento
  - Estudo de caso / exercício

VHDL - Visão Geral

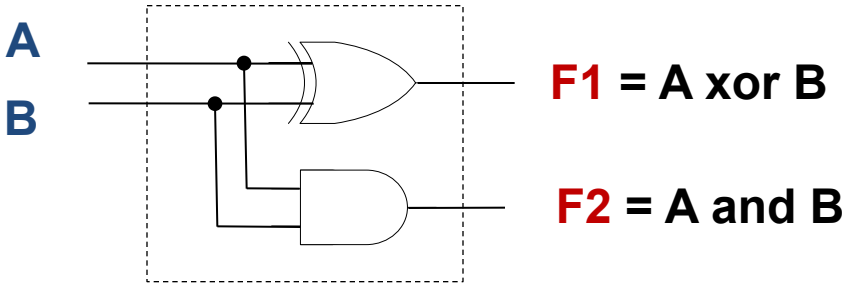
- **VHDL - linguagem para descrição de hardware**
- **VHDL = VHSIC Hardware Description Language**
- **VHSIC = Very High Speed Integrated Circuits. Programa do governo dos USA do início dos anos 80.**
- **No final da década de 80, VHDL se tornou um padrão IEEE (Institute of Electrical and Electronic Engineers).**
- **Existem diversas ferramentas para simular e sintetizar (gerar hardware) circuitos descritos em VHDL.**
- **Outras linguagens de descrição de hardware: Verilog, SystemC, AHDL, Handel-C, System Verilog, Abel, Ruby, ...**

VHDL - Visão Geral

- **O projeto de um circuito digital pode ser descrito em VHDL em diversos níveis de abstração (estrutural, comportamental).**
- **Descrições no nível de transferência entre registradores (RTL, Register Transfer Level) são bastante utilizadas.**
- **VHDL NÃO é uma linguagem de programação, e as ferramentas de síntese (não são de "compilação") não geram códigos executáveis a partir de uma descrição VHDL.**
- **Descrições em VHDL podem ser simuladas (executadas em um simulador).**
- **Descrições em VHDL podem ser utilizadas para gerar um hardware (arquivo para configuração de um FPGA, por exemplo).**
- **A geração de estímulos para simulação VHDL é realizada por intermédio de testbenches.**
- **Um testbench define os estímulos externos a serem utilizados como entrada para o circuito (definição do comportamento externo ao circuito sob teste).**
- **O testbench pode ser escrito em VHDL ou em diversas outras linguagens (ex. C, C++, ...).**

Descrição de circuito digital em VHDL

ENTITY



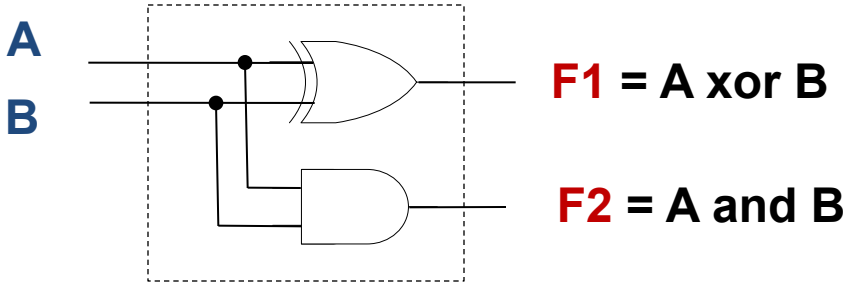
A	B	F1	F2
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

```
entity halfadd is
port (A: in std_logic;
      B: in std_logic;
      F1: out std_logic;
      F2: out std_logic
      );
end halfadd;
```

ENTITY – define os "pinos" do circuito digital (sinais), ou seja, a interface entre a lógica implementada e o mundo externo.

Descrição de circuito digital em VHDL

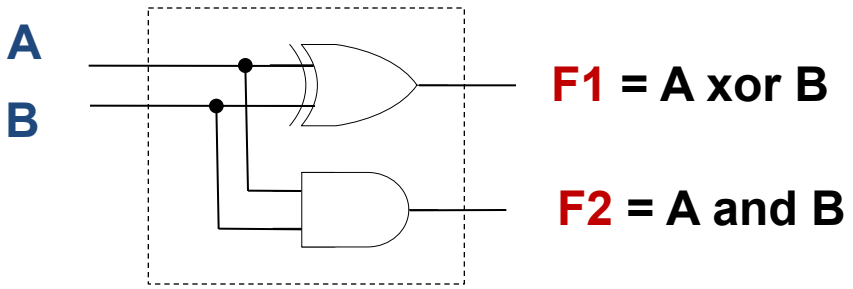
ARCHITECTURE




```
architecture circuito_logico of halfadd is
begin
   F1 <= A xor B;
   F2 <= A and B;
end circuito_logico;
```

ARCHITECTURE – define a funcionalidade do circuito digital, utilizando os "pinos" de entrada e saída listados na ENTITY em questão. Uma ENTITY pode possuir diversas implementações diferentes (diversas ARCHITECTURES).

Descrição completa do circuito em VHDL (Entity e Architecture)




```
library IEEE;
use IEEE.Std_Logic_1164.all;
```

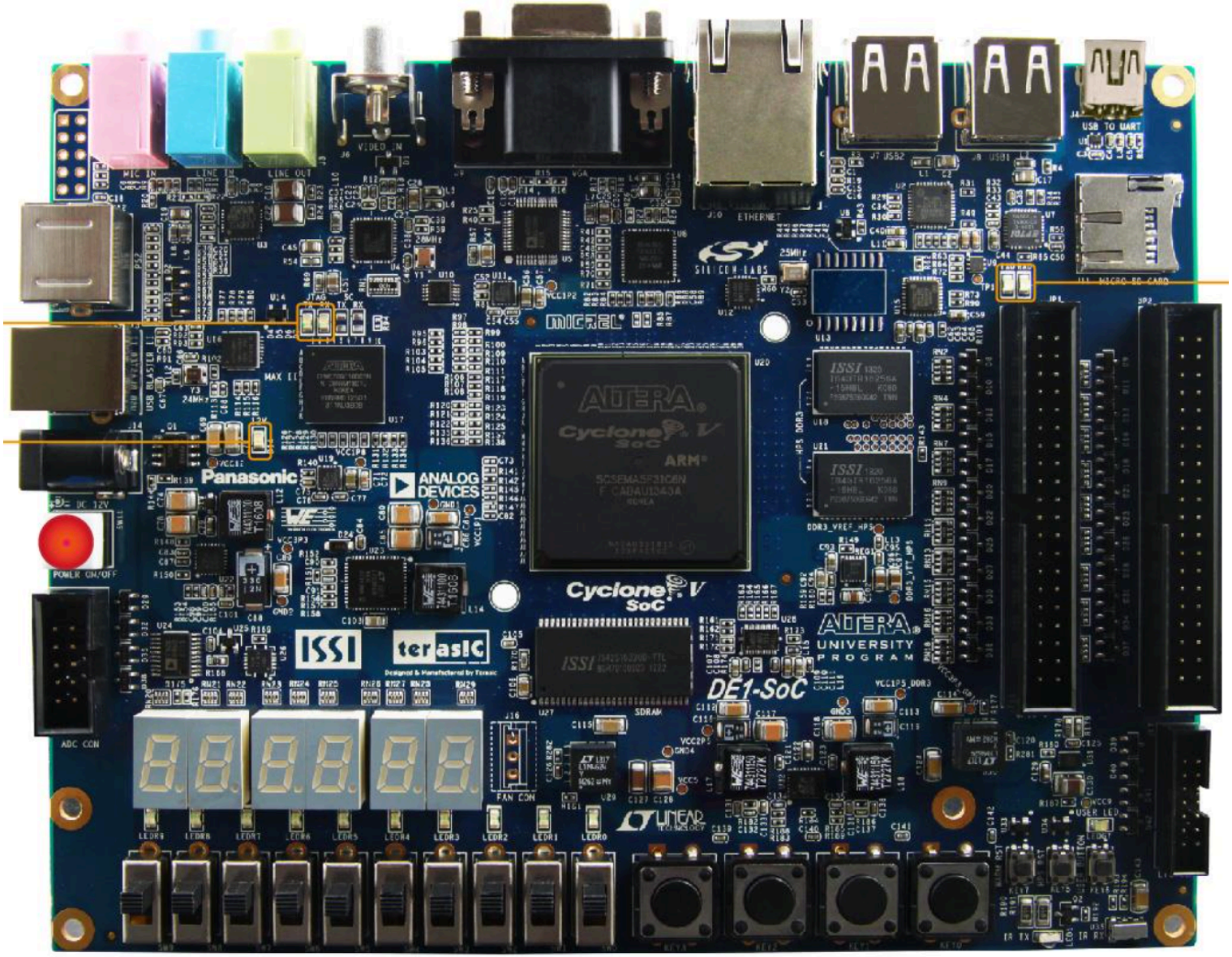
```
entity halfadd is
port (A: in std_logic;
      B: in std_logic;
      F1: out std_logic;
      F2: out std_logic
      );
end halfadd;
```

Para utilizar o tipo std_logic é necessário incluir um pacote da biblioteca IEEE.

```
architecture circuito_logico of halfadd is
begin
  F1 <= A xor B;
  F2 <= A and B;
end circuito_logico;
```

Plataforma de prototipação FPGA Altera – DE1-SoC

Kit DE1-SoC da Altera



Interface com o usuário (entrada e saída)

- **Placa DE1-SoC possui 10 LEDs vermelhos denominados LEDR$_{9-0}$ e 10 chaves denominadas SW$_{9-0}$**
- **As conexões entre esses componentes e os pinos do FPGA da placa estão definidas no arquivo *pinos.qsf***
- **São utilizados "vetores" para facilitar o acesso aos LEDs e chaves da placa**
- **Exemplo: SW[0] é o elemento 0 do vetor SW, e está conectado ao pino PIN_AB12 do FPGA**
- **No código em VHDL, usar sempre os nomes definidos no arquivo *pinos.qsf,* disponível no site da disciplina**

Interface com o usuário (entrada e saída)

- **Código VHDL para "leitura" das chaves e "escrita" nos LEDs**

```
library ieee;
use ieee.std_logic_1164.all;
entity part1 is
      port ( SW     : in std_logic_vector(9 downto 0);
             LEDR : out std_logic_vector(9 downto 0)
           );
end part1;
architecture behavior of part1 is
begin
      LEDR(3 downto 0) <= SW(7 downto 4);
      LEDR(7 downto 4) <= "0101";
      LEDR(9) <= (SW(9) AND SW(0)) OR (SW(8) AND '1');
end behavior;
```

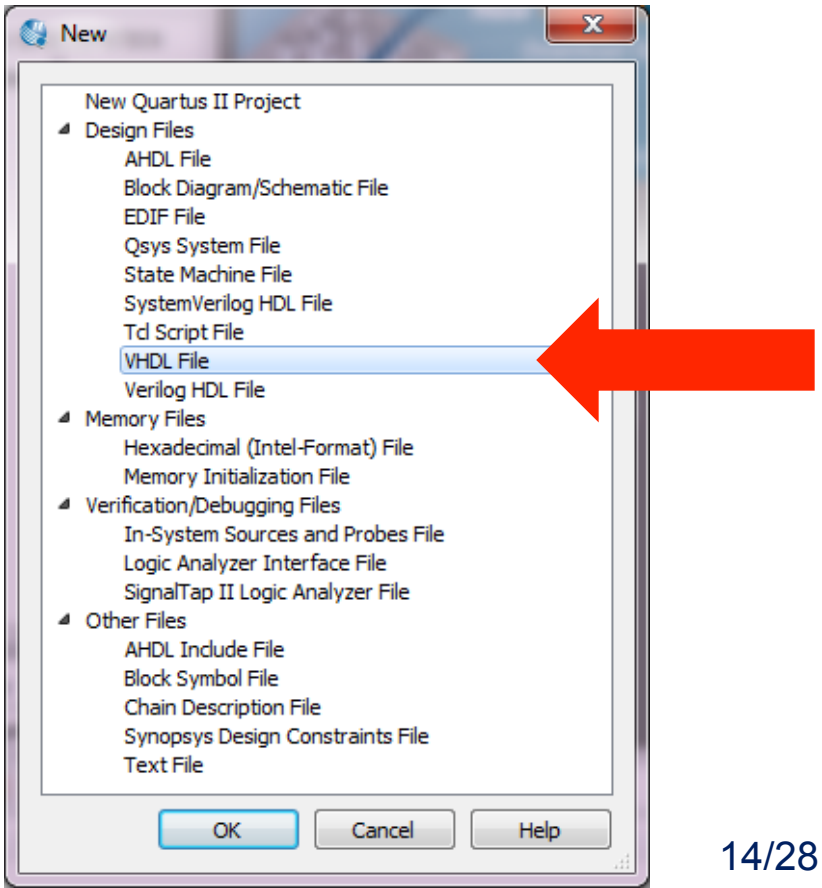
Tarefa a ser realizada na aula prática

Tarefa a ser realizada na aula prática

1. Utilizando a ferramenta Quartus II da Altera, criar um projeto VHDL que implemente o circuito apresentado no slide 7 (*and* e *xor*).
2. Realizar a simulação do circuito (VHDL) por intermédio de diagramas de formas de onda, e obter a tabela verdade.
3. Visando fixar o conhecimento do fluxo de ferramentas de projeto, seguir o tutorial descrito no livro texto, e detalhado na última aula.
4. Utilizando as dicas do slide 11, alterar o projeto de forma a realizar a entrada dos dados A e B a partir das chaves SW(0) e SW(1), e a apresentação dos resultados F1 e F2 no LEDR(0) e LEDR(1).
5. Testar o circuito no kit DE1-SoC, usando as chaves SW(0) e SW(1) para entrar com os operandos, e observar os resultados nos LEDs.

Resumo do tutorial: *Etapa 1 - Design Entry*

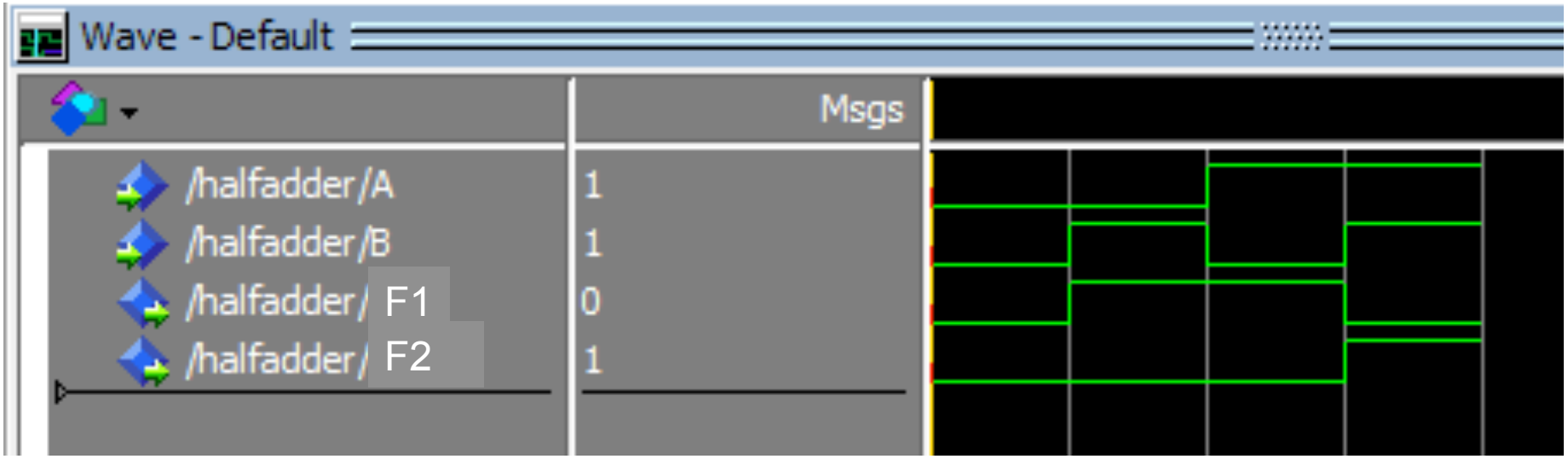
1. [Quartus II] File -> New Project Wizard
2. No "project wizard", seguir exatamente os passos do tutorial da última aula.
3. File -> New -> VHDL File (Essa é a principal diferença!).
4. Copiar o fonte VHDL do slide 7 para o novo arquivo, e salvar.



Resumo do tutorial: *Etapa 2 - Simulação*

5. **[ModelSim] Simulação Funcional – Teste do circuito**
   -> não considera informação de temporização.
   - *Seguir o tutorial de simulação da última aula.*

6. **Resultado esperado da simulação:**



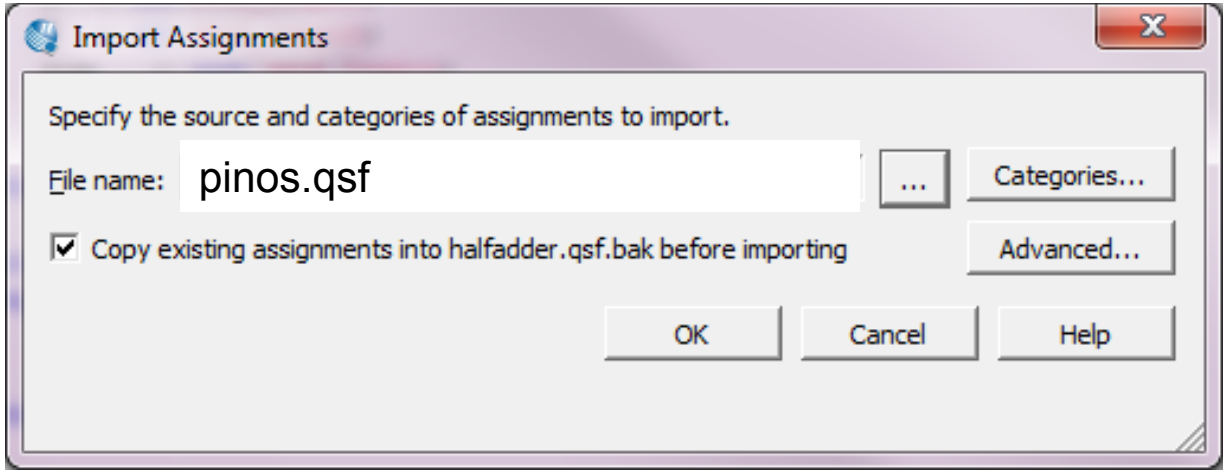
Resumo do tutorial: *Etapa 3 – prototipação FPGA*

7. Adaptar o fonte para os nomes de sinais da DE1-SoC:

```
1   library IEEE;
2   use IEEE.Std_Logic_1164.all;
3
4   entity halfadder is
5   port (                                                  -- A -> SW(0)
6       SW   : in  std_logic_vector( 9 downto 0);           -- B -> SW(1)
7       LEDR: out std_logic_vector( 9 downto 0)             -- sum   -> LEDR(0)
8       );                                                  -- carry -> LEDR(1)
9   end halfadder;
10
11  architecture ha_stru of halfadder is
12  begin
13    LEDR(0) <= SW(0) xor SW(1);       -- sum   <= A xor B
14    LEDR(1) <= SW(0) and SW(1);       -- carry <= A and B
15  end ha_stru;
```

Resumo do tutorial: *Etapa 3 – prototipação FPGA*

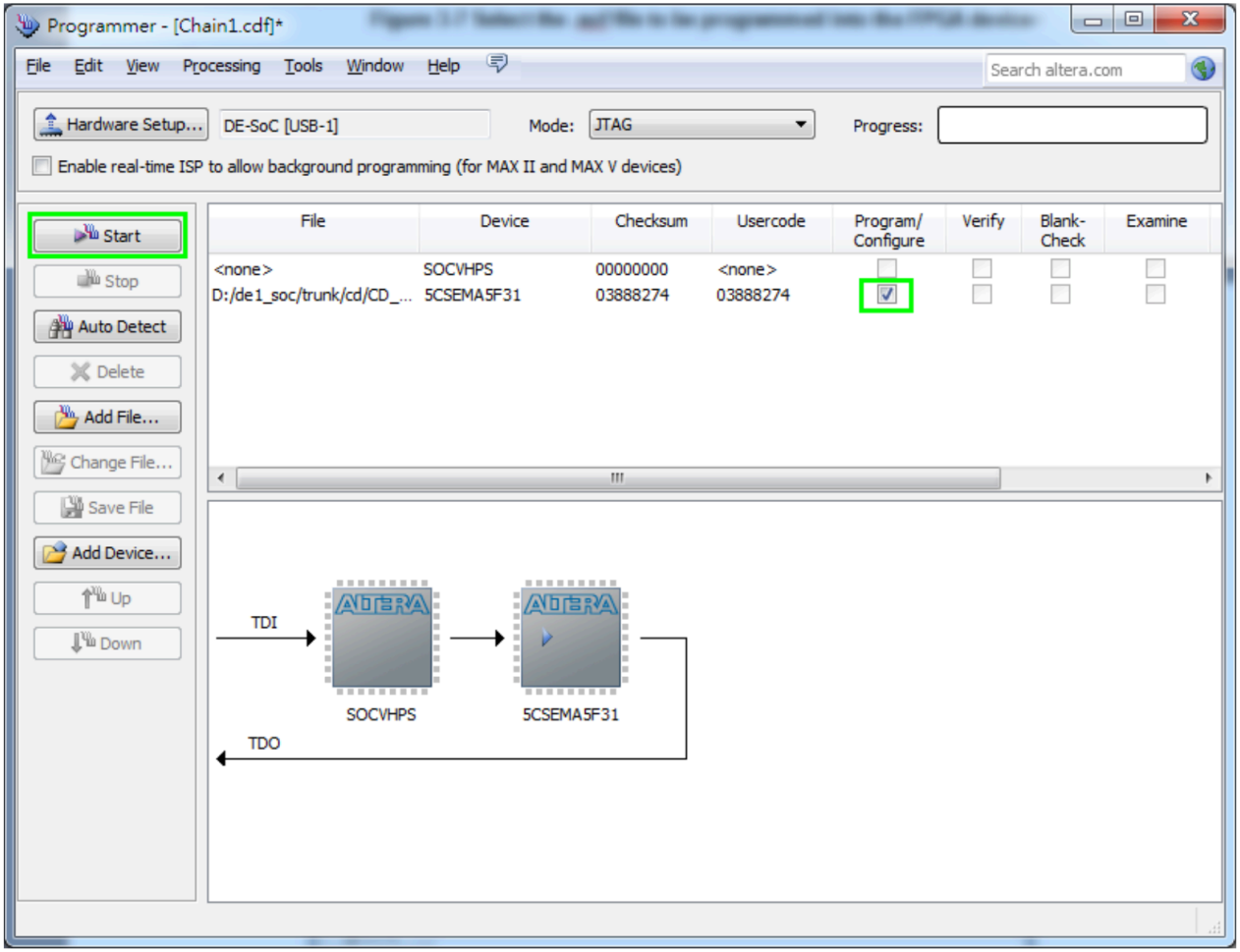
8. **Assignments -> Import Assigments (procurar no site e usar o arquivo *pinos.qsf*)**



9. **Com isso, os pinos do FPGA foram associados aos sinais da entity do VHDL**
10. **Compilar o VHDL (síntese)**
11. **ATENÇÃO!!! Verificar se o nome da entity é o mesmo nome do projeto, para evitar erros na síntese.**
12. **A compilação resulta em dezenas de warnings devido aos pinos não conectados do arquivo *.qsf***

Resumo do tutorial: *Etapa 3 – prototipação FPGA*

13. **Programação – FPGA é carregado com circuito, configurando fisicamente elementos de processamento e roteamento.**
    Tools – Programmer. Hardware Setup – USB-Blaster. Start!



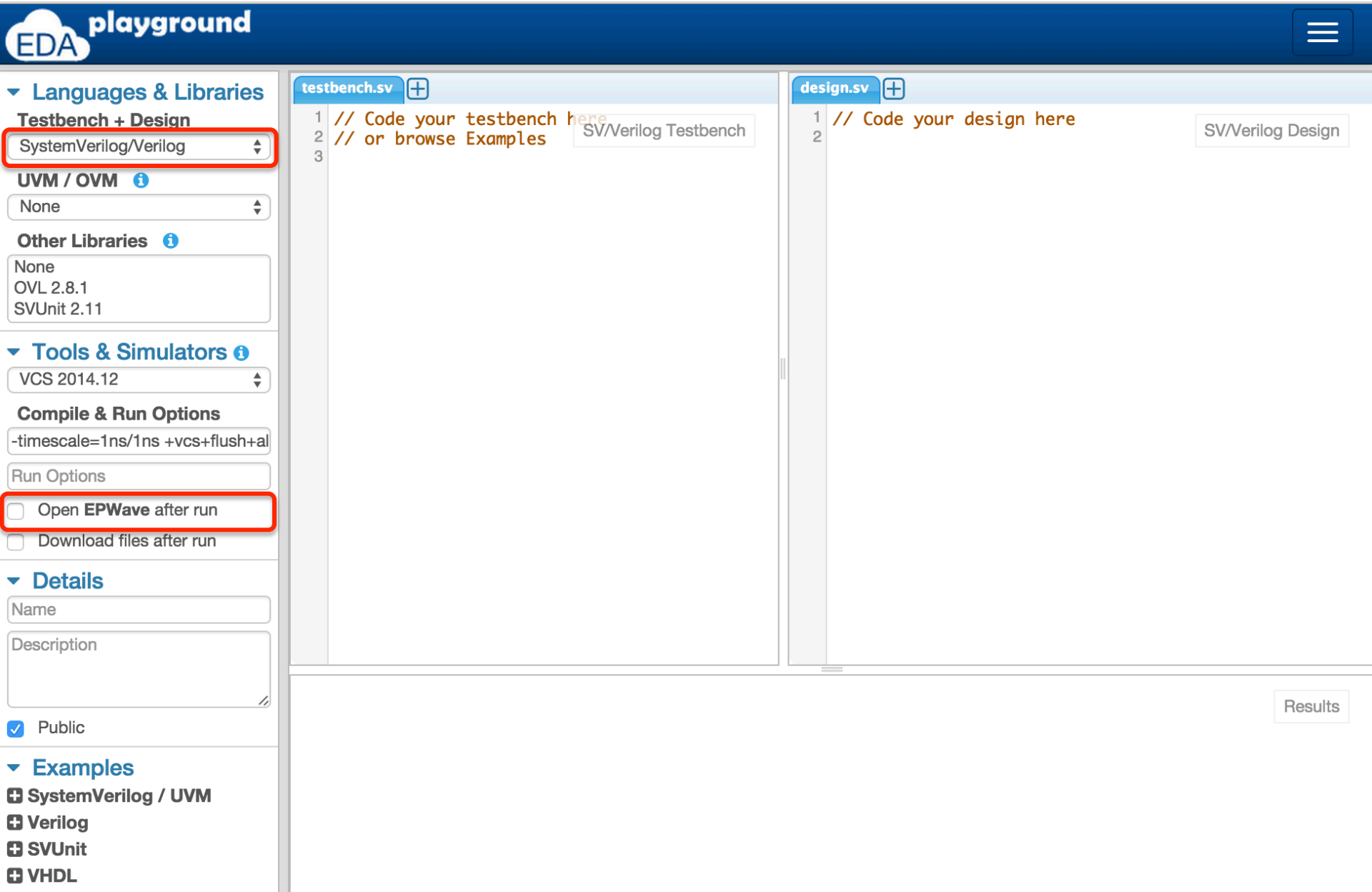
Dica: Ferramenta de simulação na web

Outra opção para simulação: http://www.edaplayground.com/

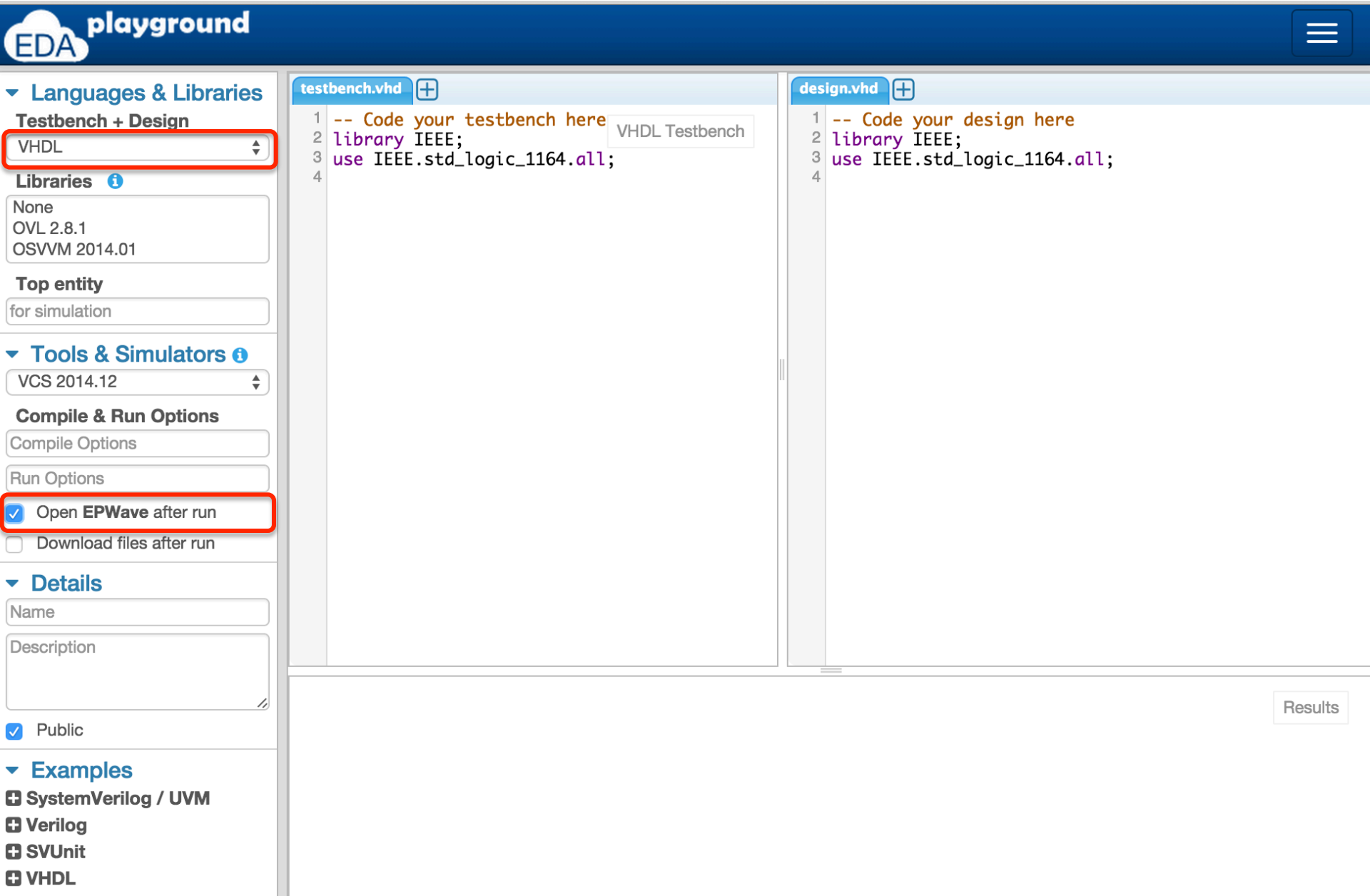
As etapas para usar essa ferramenta são:

1. **Usar um projeto exemplo (template) como base**
2. **Copiar o fonte VHDL a ser simulado para a partição da direita**
3. **Alterar o "testbench" para testar o VHDL desejado**
4. **Executar a simulação**

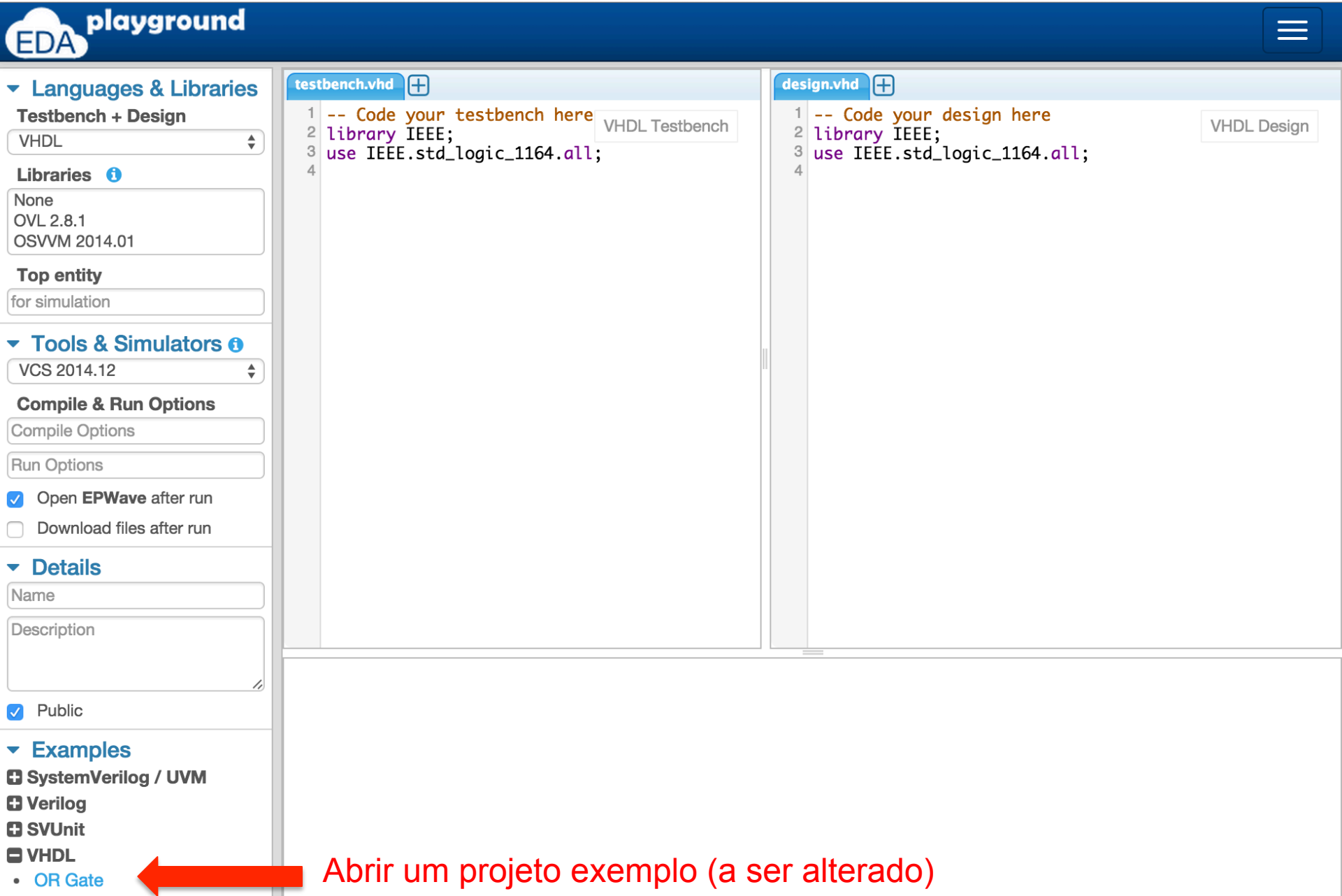
Simulador na web: *http://www.edaplayground.com/*

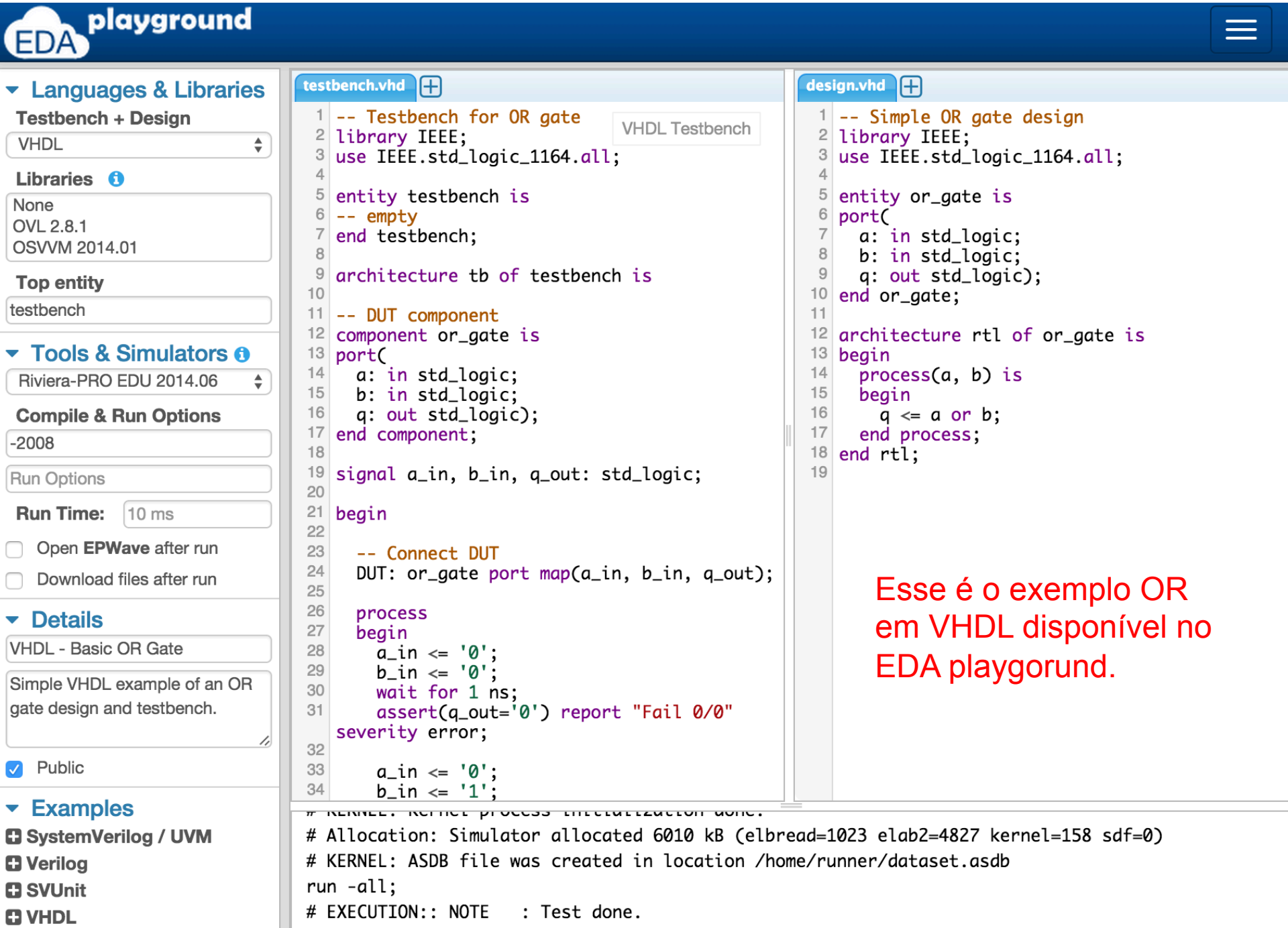


Simulador na web: *http://www.edaplayground.com/*



Simulador na web: http://www.edaplayground.com/




EDA playground
Languages & Libraries
Testbench + Design
VHDL
Libraries
None
OVL 2.8.1
OSVVM 2014.01
Top entity
testbench
Tools & Simulators
Riviera-PRO EDU 2014.06
Compile & Run Options
-2008
Run Options
Run Time:
10 ms
Open EPWave after run
Download files after run
Details
VHDL - Basic OR Gate
Simple VHDL example of an OR gate design and testbench.
Public
Examples
SystemVerilog / UVM
Verilog
SVUnit
VHDL
testbench.vhd
VHDL Testbench
-- Testbench for OR gate
library IEEE;
use IEEE.std_logic_1164.all;
entity testbench is
-- empty
end testbench;
architecture tb of testbench is
-- DUT component
component or_gate is
port(
a: in std_logic;
b: in std_logic;
q: out std_logic);
end component;
signal a_in, b_in, q_out: std_logic;
begin
-- Connect DUT
DUT: or_gate port map(a_in, b_in, q_out);
process
begin
a_in <= '0';
b_in <= '0';
wait for 1 ns;
assert(q_out='0') report "Fail 0/0"
severity error;
a_in <= '0';
b_in <= '1';
design.vhd
-- Simple OR gate design
library IEEE;
use IEEE.std_logic_1164.all;
entity or_gate is
port(
a: in std_logic;
b: in std_logic;
q: out std_logic);
end or_gate;
architecture rtl of or_gate is
begin
process(a, b) is
begin
q <= a or b;
end process;
end rtl;
Esse é o exemplo OR em VHDL disponível no EDA playgorund.
Allocation: Simulator allocated 6010 kB (elbread=1023 elab2=4827 kernel=158 sdf=0)
KERNEL: ASDB file was created in location /home/runner/dataset.asdb
run -all;
EXECUTION:: NOTE : Test done.

EDA playground

Languages & Libraries

Testbench + Design

VHDL

Libraries

None
OVL 2.8.1
OSVVM 2014.01

Top entity

testbench

Tools & Simulators

Riviera-PRO EDU 2014.06

Compile & Run Options

-2008

Run Options

Run Time: 10 ms

- [ ] Open **EPWave** after run
- [ ] Download files after run

Details

VHDL - Basic OR Gate

Simple VHDL example of an OR gate design and testbench.

- [x] Public

Examples

- SystemVerilog / UVM
- Verilog
- SVUnit
- VHDL

testbench.vhd

VHDL Testbench

```
1  -- Testbench for OR gate
2  library IEEE;
3  use IEEE.std_logic_1164.all;
4
5  entity testbench is
6  -- empty
7  end testbench;
8
9  architecture tb of testbench is
10
11 -- DUT component
12 component or_gate is
13 port(
14   a: in std_logic;
15   b: in std_logic;
16   q: out std_logic);
17 end component;
18
19 signal a_in, b_in, q_out: std_logic;
20
21 begin
22
23   -- Connect DUT
24   DUT: or_gate port map(a_in, b_in, q_out);
25
26   process
27   begin
28     a_in <= '0';
29     b_in <= '0';
30     wait for 1 ns;
31     assert(q_out='0') report "Fail 0/0"
severity error;
32
33     a_in <= '0';
34     b_in <= '1';
```

design.vhd

VHDL Design

```
1  library IEEE;
2  use IEEE.Std_Logic_1164.all;
3
4  entity halfadd is
5  port (A: in std_logic;
6        B: in std_logic;
7        F1: out std_logic;
8        F2: out std_logic
9        );
10 end halfadd;
11
12 architecture circuito_logico of halfadd is
13 begin
14   F1 <= A xor B;
15   F2 <= A and B;
16 end circuito_logico;
17
```

Substituir o exemplo OR, pelo fonte VHDLdesejado.

```
# KERNEL: Kernel process initialization done.
# Allocation: Simulator allocated 6010 kB (elbread=1023 elab2=4827 kernel=158 sdf=0)
# KERNEL: ASDB file was created in location /home/runner/dataset.asdb
run -all;
# EXECUTION:: NOTE    : Test done.
```

EDA playground

Languages & Libraries

Testbench + Design

VHDL

Libraries

None
OVL 2.8.1
OSVVM 2014.01

Top entity

testbench

Tools & Simulators

Riviera-PRO EDU 2014.06

Compile & Run Options

-2008

Run Options

Run Time: 10 ms

Open **EPWave** after run

Download files after run

Details

VHDL - Basic OR Gate

Simple VHDL example of an OR gate design and testbench.

Public

Examples

- SystemVerilog / UVM
- Verilog
- SVUnit
- VHDL

testbench.vhd

VHDL Testbench

```
1  -- Testbench for halfadd gate
2  library IEEE;
3  use IEEE.std_logic_1164.all;
4
5  entity testbench is
6  -- empty
7  end testbench;
8
9  architecture tb of testbench is
10
11 -- DUT component
12 component halfadd is
13 port (A: in std_logic;
14       B: in std_logic;
15       F1: out std_logic;
16       F2: out std_logic
17       );
18 end component;
19
20 signal A_in, B_in, F1_out, F2_out:
   std_logic;
21
22 begin
23
24   -- Connect DUT
25   DUT: halfadd port map(A_in, B_in, F1_out,
   F2_out);
26
27   process
28   begin
29     A_in <= '0';
30     B_in <= '0';
31     wait for 1 ns;
32     assert(F1_out='0') report "Erro: F1_out
   = 1" severity error;
```

design.vhd

VHDL Design

```
1  library IEEE;
2  use IEEE.Std_Logic_1164.all;
3
4  entity halfadd is
5  port (A: in std_logic;
6        B: in std_logic;
7        F1: out std_logic;
8        F2: out std_logic
9        );
10 end halfadd;
11
12 architecture circuito_logico of halfadd is
13 begin
14   F1 <= A xor B;
15   F2 <= A and B;
16 end circuito_logico;
17
```

Alterar o testbench do exemplo OR, de forma a realizar a simulação do *halfadd*.

```
# KERNEL: Kernel process initialization done.
# Allocation: Simulator allocated 6010 kB (elbread=1023 elab2=4827 kernel=158 sdf=0)
# KERNEL: ASDB file was created in location /home/runner/dataset.asdb
run -all;
# EXECUTION:: NOTE    : Test done.
```


EDA playground
Run
Copy*
Share
Collaborate beta
Forum
Eduardo Bezerra
Pressionar o “Run” para executar a simulação.
-2008
Run Options
Run Time: 10 ms
Open EPWave after run
Download files after run
Details
VHDL - Basic OR Gate
Simple VHDL example of an OR gate design and testbench.
Public
Examples
SystemVerilog / UVM
Verilog
SVUnit
);
end component;
signal A_in, B_in, F1_out, F2_out: std_logic;
-- Connect DUT
DUT: halfadd port map(A_in, B_in, F1_out, F2_out);
process
begin
A_in <= '0';
B_in <= '0';
wait for 1 ns;
assert(F1_out='0') report "Erro: F1_out = 1" severity error;
# Allocation: Simulator allocated 6010 kB (elbread=1023 elab2=4827 kernel=158 sdf=0)
# KERNEL: ASDB file was created in location /home/runner/dataset.asdb
run -all;
Antes de pressionar o “Run”, certificar que a opção “Open EPWave after run” está selecionada.

Simulador na web: *http://www.edaplayground.com/*

Resultado esperado para a simulação: