

Projetos de Circuitos Digitais em VHDL e FPGA

Cap. 7 - Simulação de Circuitos Digitais com VHDL

Prof. Erivelton Geraldo Nepomuceno

Engenharia Elétrica
UFSJ - Universidade Federal de São João del-Rei

13 de fevereiro de 2019

Objetivo

Objetivo Principal

Descrever os fundamentos para a criação de testbenches utilizando VHDL.

Objetivos Específicos

- Explicar os tipos de testbenches e ensinar a desenvolvê-los.
- Geração e teste de estímulos.
- Ensinar a fazer simulações usando o ModelSim.

- Síntese é o processo de tradução de um código-fonte em um conjunto de estruturas de hardware que implementam as funcionalidades descritas no código.
- Simulação é um procedimento de teste usado para verificar se o circuito **sintetizado** de fato implementa o comportamento pretendido.

Simulação

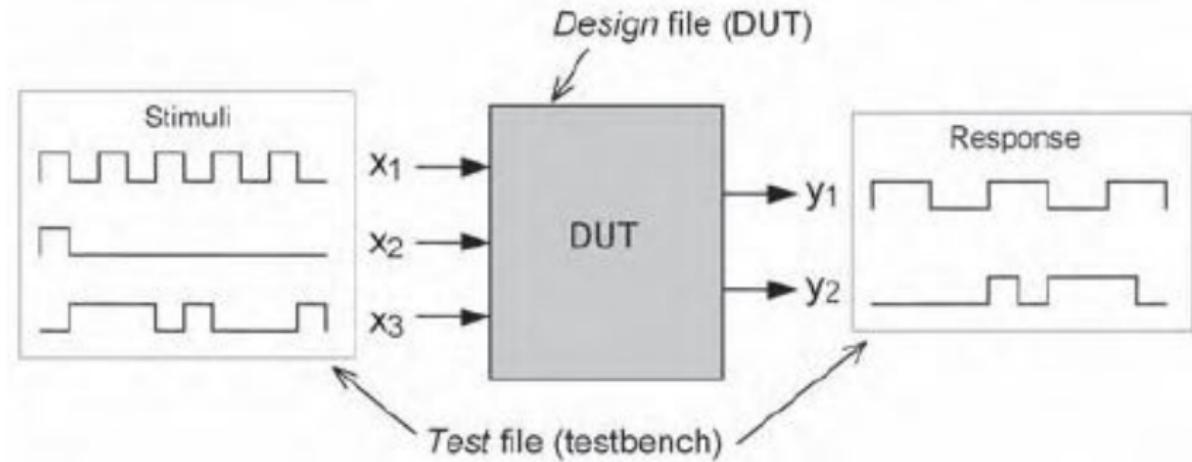


Figura 1: Resumo da simulação de projetos em VHDL.

Tipos de Testbenches

Testbenches	Retardos de propagação	Análise do sinal na saída
I	Não incluídos	Inspeção visual
II	Incluídos	Inspeção visual
III	Não incluídos	Inspeção com VHDL
IV	Incluídos	Inspeção com VHDL

Tabela 1: Quadro que indica os quatro tipos de simulações.

- Instruções de tempo em VHDL:
 - AFTER (para código concorrente).
 - WAIT FOR (para código sequencial).
 - NOW
- Arquivos necessários para efetuar as simulações:

Para realizar análise funcional:

- Arquivo do projeto (extensão .vhd).
- Arquivo do testbench (extensão .vhd).

Para realizar análise temporal:

- Arquivo do projeto (extensão .vhd).
- Arquivo do testbench (extensão .vhd).
- Arquivo pós-síntese (extensão .vho).
- Arquivo com anotações temporais (extensão .sdo).

Geração de Estímulo

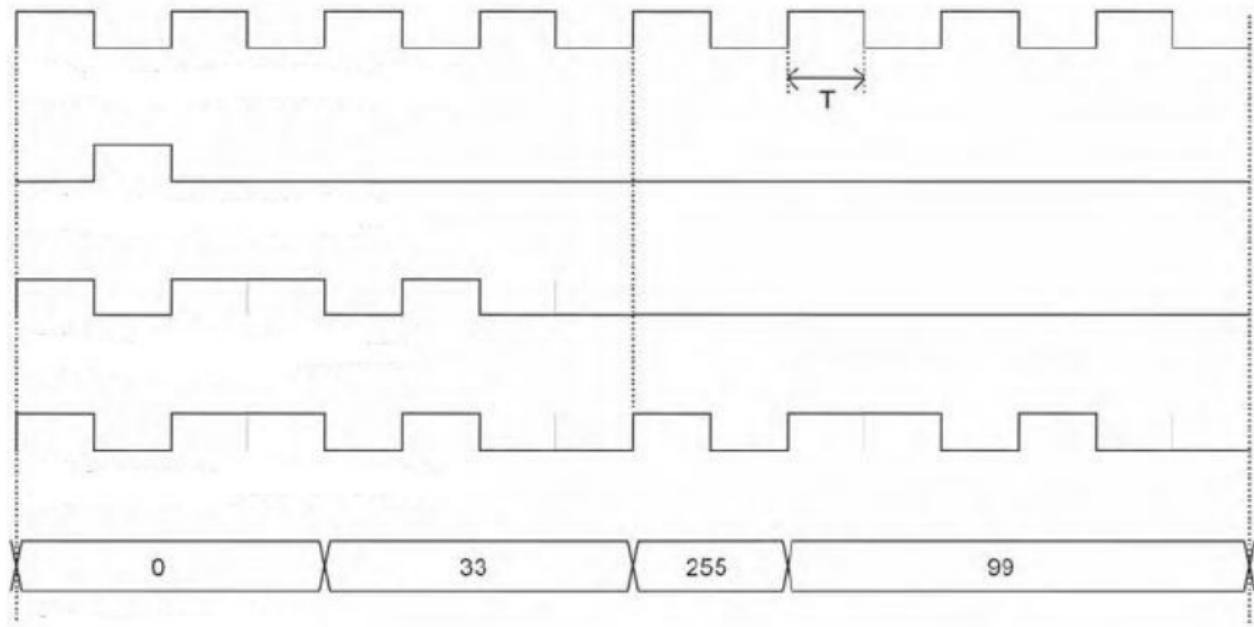


Figura 2: Exemplos de formas de onda.

Geração de Estímulo

- Caso 1: Geração de uma forma de onda repetitiva regular.

(a)

```
SIGNAL clk: BIT := '1';  
-----  
clk <= NOT clk AFTER 30 ns;
```

(b)

```
SIGNAL clk: BIT := '1';  
-----  
WAIT FOR 30 ns;  
clk <= NOT clk;
```

(c)

```
SIGNAL clk: BIT := '1';  
-----  
WAIT FOR 30 ns;  
clk <= '0';  
WAIT FOR 30 ns;  
clk <= '1';
```

Figura 3: Descrição de geração de forma de onda $n^o 1$.

Geração de Estímulo

- Caso 2: Geração de uma forma de onda de um só pulso.

```
-----  
SIGNAL rst: BIT := '0';  
-----  
WAIT FOR 30 ns;  
rst <= '1';  
WAIT FOR 30 ns;  
rst <= '0';  
WAIT;  
-----
```

Figura 4: Descrição de geração de forma de onda *nº2*.

Geração de Estímulo

- Caso 3: Geração de uma forma de onda não repetitiva irregular.

```
-----  
CONSTANT wave: BIT_VECTOR(1 TO 8) := "10110100";  
SIGNAL x: BIT := '0';  
-----  
FOR i IN wave'RANGE LOOP  
    x <= wave(i);  
    WAIT FOR 30 ns;  
END LOOP;  
WAIT;  
-----
```

Figura 5: Descrição de geração de forma de onda $n^o 3$.

Geração de Estímulo

- Caso 4: Geração de uma forma de onda repetitiva irregular.

```
-----
CONSTANT wave: BIT_VECTOR(1 TO 8) := "10110100";
SIGNAL y: BIT := '0';
-----
FOR i IN wave'RANGE LOOP
    y <= wave(i);
    WAIT FOR 30 ns;
END LOOP;
-----
```

Figura 6: Descrição de geração de forma de onda $n^o 4$.

Geração de Estímulo

- Caso 5: Geração de uma forma de onda multibit.

```
-----  
SIGNAL z: INTEGER RANGE 0 TO 255;  
-----  
z <= 0;  
WAIT FOR 120 ns;  
z <= 33;  
WAIT FOR 120 ns;  
z <= 255;  
WAIT FOR 60 ns;  
z <= 99;  
WAIT FOR 180 ns;  
-----
```

Figura 7: Descrição de geração de forma de onda $n^o 5$.

Teste de Estímulos

- Importante para ter certeza que os sinais que serão aplicados ao circuito são, de fato, pretendidos.
- Tal teste é feito no ModelSim.

```
1 -----  
2 LIBRARY ieee;  
3 USE ieee.std_logic_1164.all;  
4 -----  
5 ENTITY test_testbench IS  
6 END test_testbench;  
7 -----
```

Library declarations {

Entity (empty) {

Figura 8: Código VHDL padrão para teste do próprio testbench (Parte I).

Teste de Estímulos

Architecture

```
8 ARCHITECTURE testbench OF test_testbench IS
9     SIGNAL a: STD_LOGIC := '1';
10    SIGNAL b: STD_LOGIC := '0';
11 BEGIN
12     ----- generate a: -----
13     PROCESS
14     BEGIN
15         WAIT FOR 30 ns;
16         a <= NOT a;
17     END PROCESS;
18     ----- generate b: -----
19     PROCESS
20     BEGIN
21         WAIT FOR 30 ns
22         b <= '1';
23         WAIT 30 ns;
24         b <= '0';
25         WAIT;
26     END PROCESS;
27 END testbench;
28 -----
```

Signal declarations

Stimuli generation

Figura 9: Código VHDL padrão para teste do próprio testbench (Parte II).

Código padrão para testbenches

```
1  LIBRARY ieee;
2  USE ieee.std_logic_1164.all;
3
4
5  ENTITY test_mycircuit IS
6  END test_mycircuit;
7
8  ARCHITECTURE full_testbench OF test_mycircuit IS
9    ---- component declaration: -----
10   COMPONENT mycircuit IS
11     PORT (clk, x: IN STD_LOGIC;
12           y: OUT STD_LOGIC);
13   END COMPONENT;
14   ---- signal declarations: -----
15   SIGNAL t_clk: BIT := '0';
16   SIGNAL t_x: BIT := '1';
17   SIGNAL t_y: BIT;
```

Library declarations {

Entity (empty) {

} Declarations {

Figura 10: Código VHDL padrão para simulações de projetos (Parte I).

Código padrão para testbenches

Architecture

```
18 BEGIN
19     ---- component instantiation: -----
20     dut: mycircuit PORT MAP (clk=>t_clk,
21                             x=>t_x, y=>t_y);
22     ---- generate clk: -----
23 PROCESS
24 BEGIN
25     WAIT ...
26 END PROCESS;
27     ---- generate x: -----
28 PROCESS
29 BEGIN
30     WAIT ...
31 END PROCESS;
32     ---- verify y: -----
33 PROCESS
34 BEGIN
35     WAIT ...
36     ASSERT ...
37 END PROCESS;
38 END full_testbench;
39 -----
```

Component instantiation

Stimulus generation

Output verification
(optional)

Figura 11: Código VHDL padrão para simulações de projetos (Parte II).

Desenvolvendo Testbenches Tipo I e Tipo II

- Para o Tipo I, os tempos de propagação não incluídos, caso contrário se trata de um testbench Tipo II.
- Resultados da saída são examinados por inspeção visual.

Ex.: Divisor de Clock.

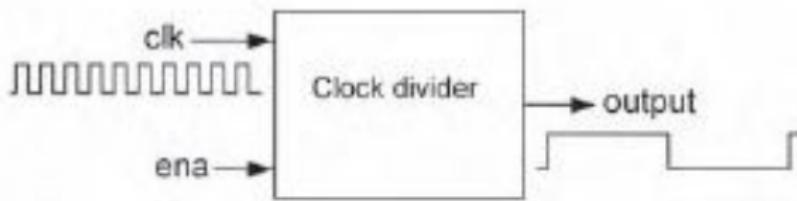


Figura 12: Exemplo de divisor de clock.

Desenvolvendo Testbenches Tipo I e Tipo II - Arquivo do Projeto

```
1 ----- Design file (clock_divider.vhd): -----
2 ENTITY clock_divider IS
3     PORT (clk, ena: IN BIT;
4             output: OUT BIT);
5 END clock_divider;
6 -----
7 ARCHITECTURE clock_divider OF clock_divider IS
8     CONSTANT max: INTEGER := 5;
9 BEGIN
10    PROCESS(clk)
11        VARIABLE count: INTEGER RANGE 0 TO 7 := 0;
12        VARIABLE temp: BIT;
13    BEGIN
14        IF (clk'EVENT AND clk='1') THEN
15            IF (ena='1') THEN
16                count := count + 1;
17                IF (count=max) THEN
18                    temp := NOT temp;
19                    count := 0;
20                END IF;
21            END IF;
22            output <= temp;
23        END IF;
24    END PROCESS;
25 END clock_divider;
26 -----
```

Figura 13: Arquivo de projeto do divisor de clock.

Desenvolvendo Testbenches Tipo I e Tipo II - Arquivo de Teste

```
1  ---- Test file (test_clock_divider.vhd):-----
2  ENTITY test_clock_divider IS
3  END test_clock_divider;
4  -----
5  ARCHITECTURE test_clock_divider OF test_clock_divider IS
6      ---- component declaration: -----
7      COMPONENT clock_divider IS
8          PORT (clk, ena: IN BIT;
9                  output: OUT BIT);
10     END COMPONENT;
11     ---- signal declarations: -----
12     SIGNAL clk: BIT := '0';
13     SIGNAL ena: BIT := '0';
14     SIGNAL output: BIT;
```

Figura 14: Arquivo de teste do divisor de clock (Parte 1).

Desenvolvendo Testbenches Tipo I e e Tipo II - Arquivo de Teste

```
15 BEGIN
16     ----- component instantiation: -----
17     dut: clock_divider PORT MAP (clk=>clk, ena=>ena, output=>output);
18     ----- generate clock: -----
19     PROCESS
20     BEGIN
21         WAIT FOR 30 ns;
22         clk <= NOT clk;
23     END PROCESS;
24     ----- generate enable: -----
25     PROCESS
26     BEGIN
27         WAIT FOR 60 ns;
28         ena <= '1';
29         WAIT; --optional
30     END PROCESS;
31 END test_clock_divider;
32 -----
```

Figura 15: Arquivo de teste do divisor de clock (Parte 2).

Desenvolvendo Testbenches Tipo I e e Tipo II - Resultado

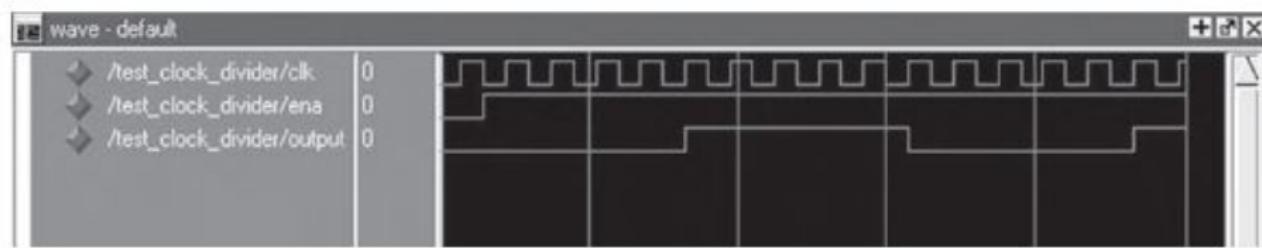


Figura 16: Resultados da simulação do divisor de clock.

Desenvolvendo Testbenches Tipo III e Tipo IV

- Tempos de propagação internos não são incluídos para o Testbench Tipo 3.
- Tempos de propagação internos são incluídos para o Testbench Tipo 4.
- Resultados da saída são examinados pelo próprio código em VHDL.
- Arquivo de Projeto do divisor de clock é o mesmo para os casos anteriores.

Desenvolvendo Testbenches Tipo III e Tipo IV - Arquivo de Teste

```
1 ---- Test file (test_clock_divider.vhd):-----
2 ENTITY test_clock_divider IS
3 END test_clock_divider;
4 -----
5 ARCHITECTURE test_clock_divider OF test_clock_divider IS
6     ---- component declaration: -----
7     COMPONENT clock_divider IS
8         PORT (clk, ena: IN BIT;
9                 output: OUT BIT);
10    END COMPONENT;
11    ---- signal declarations: -----
12    SIGNAL clk: BIT := '0';
13    SIGNAL ena: BIT := '0';
14    SIGNAL output: BIT;
```

Figura 17: Arquivo de teste para Tesbenches do Tipo 3 e Tipo 4 (Parte 1).

Desenvolvendo Testbenches Tipo III e Tipo IV - Arquivo de Teste

```
15      SIGNAL template: BIT := '0'; --for output verification
16 BEGIN
17      ----- Component instantiation: -----
18      dut: clock_divider PORT MAP (clk=>clk, ena=>ena, output=>output);
19      ----- generate clock: -----
20      PROCESS
21      BEGIN
22          WAIT FOR 30 ns;
23          clk <= NOT clk;
24      END PROCESS;
25      ----- generate enable: -----
26      PROCESS
27      BEGIN
28          WAIT FOR 60 ns;
29          ena <= '1';
30      END PROCESS;
```

Figura 18: Arquivo de teste para Tesbenches do Tipo 3 e Tipo 4 (Parte 2).

Desenvolvendo Testbenches Tipo III e Tipo IV - Arquivo de Teste

```
31      ----- generate template: -----
32      PROCESS
33      BEGIN
34          WAIT FOR 343 ns;
35          WHILE ena='1' LOOP
36              template <= NOT template;
37              WAIT FOR 300 ns;
38          END LOOP;
39      END PROCESS;
40      ----- verify output: -----
41      PROCESS
42      BEGIN
43          WAIT FOR 1 ns;
44          ASSERT (output=template)
45              REPORT "Output differs from template!"
46              SEVERITY FAILURE;
47              --SEVERITY WARNING;
48      END PROCESS;
49  END test_clock_divider;
50 -----
```

Figura 19: Arquivo de teste para Tesbenches do Tipo 3 e Tipo 4 (Parte 3).

Sugestões de Exercícios

- Escreva um código VHDL que produza os sinais da figura abaixo. Adote $T = 5\text{ns}$.

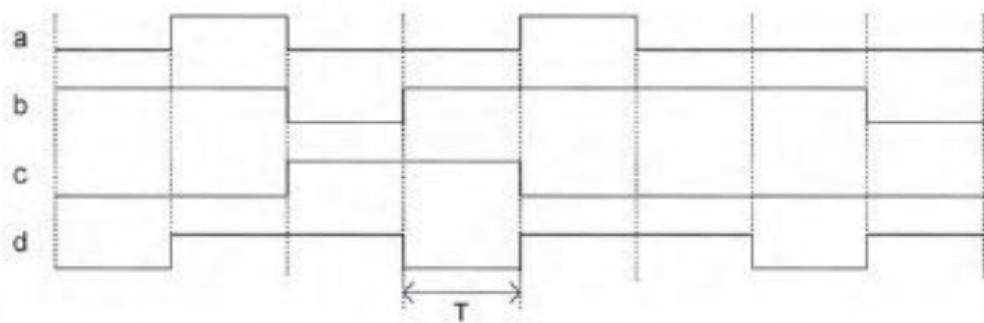


Figura 20: Sinais do exercício sugerido.

- Projete um somador de números inteiros positivos, o qual as entradas variam apenas de 0 a 255. Desenvolva os Testbenches baseados nos Tipos I e IV.

Referências

- PEDRONI, Volnei A. **Digital electronics and design with VHDL.** Morgan Kaufmann, 2008.
- PEDRONI, Volnei A. **Eletônica digital moderna e VHDL.** Rio de Janeiro, RJ: Elsevier, 2010.