

Apostila de Lógica Reconfigurável com VHDL: Fundamentos e Aplicações

Prof. Felipe Walter Dafico Pfrimer

8 de dezembro de 2025

Sumário

1	Introdução	1
1.1	O que são Dispositivos Lógicos Programáveis?	1
1.2	Linguagens de Descrição de Hardware	2
1.3	O que é VHDL?	2
1.4	Fluxo de Projeto (Design Flow)	3
2	Introdução	5
2.1	Primeiro Exemplo	5
2.2	Conclusão	6

Capítulo 1

Introdução

Esta apostila apresenta conceitos fundamentais de VHDL utilizados na disciplina, introduzindo a base tecnológica e a linguagem de descrição.

1.1 O que são Dispositivos Lógicos Programáveis?

Em um futuro distante, um marceneiro chega ao seu galpão para mais um dia de trabalho. No interior do espaço, repousam alguns cubos perfeitos — cada um com exato um metro de lado — feitos de um material misterioso. O marceneiro pega um desses cubos, leva-o ao centro do galpão e, com um pequeno dispositivo em mãos, emite comandos que dão vida à matéria inerte. Diante de seus olhos, o cubo se reconfigura: transforma-se primeiro em uma mesa robusta, depois em uma cadeira ergonômica, em seguida em uma estante organizada — tudo de acordo com a necessidade daquele momento.

O segredo está na divisão de tarefas: o dispositivo armazena os *blueprints* (os planos detalhados de cada móvel), enquanto o cubo é feito de um material inteligente, capaz de se moldar dinamicamente, respeitando apenas os limites de sua massa e volume.

Embora soe como ficção científica, essa metáfora ilustra o funcionamento dos **Dispositivos Lógicos Programáveis** (PLDs – *Programmable Logic Devices*): componentes eletrônicos cuja arquitetura interna pode ser configurada pelo usuário para executar funções lógicas específicas, simplesmente carregando um novo “projeto” — ou seja, um novo arquivo de configuração.

No contexto dos PLDs, o “material inteligente” é representado por uma matriz de blocos lógicos e interconexões programáveis. Esses blocos podem ser configurados para realizar operações lógicas variadas, enquanto as interconexões permitem que os blocos se comuniquem, formando circuitos com-

plexos. O *blueprint*, por sua vez, representa o projeto digital. Esse projeto pode ser descrito tanto por diagramas esquemáticos quanto por **Linguagens de Descrição de Hardware** (HDLs).

Existem diversas categorias de PLDs. Entre as mais comuns estão os **FP-GAs** (*Field-Programmable Gate Arrays*) e os **CPLDs** (*Complex Programmable Logic Devices*). Os FPGAs são conhecidos por sua alta densidade lógica e flexibilidade, permitindo a implementação de sistemas digitais complexos. Já os CPLDs costumam ser mais simples, possuem arquitetura menos granular e são ideais para lógicas de controle e aplicações que exigem previsibilidade temporal rigorosa.

1.2 Linguagens de Descrição de Hardware

Uma HDL (*Hardware Description Language*) **não é uma linguagem de programação**. Como o próprio nome sugere, trata-se de uma linguagem utilizada para **descrever** o hardware. Diferentemente das linguagens de software (como C ou Python), que geram instruções sequenciais para um processador, as HDLs permitem que engenheiros definam a estrutura e o comportamento de circuitos eletrônicos, onde muitas coisas acontecem simultaneamente (concorrência).

Atualmente, as HDLs mais populares para configurar PLDs são o **VHDL** (*VHSIC Hardware Description Language*), o **Verilog** e o **SystemVerilog**. Todas permitem a descrição de circuitos em diferentes níveis de abstração, desde o comportamental (o que o circuito faz) até o estrutural (quais portas lógicas compõem o circuito). Esta apostila foca no uso do **VHDL**.

1.3 O que é VHDL?

A sigla VHDL significa **VHSIC Hardware Description Language**, onde VHSIC é o acrônimo para *Very High Speed Integrated Circuit*. Desenvolvida na década de 1980 pelo Departamento de Defesa dos Estados Unidos, a linguagem foi criada para documentar e simular o comportamento de circuitos integrados complexos. Com o tempo, o VHDL evoluiu e se tornou um padrão internacional (IEEE 1076), amplamente adotado na indústria eletrônica para o design de sistemas digitais.

Para exemplificar, considere o trecho de código VHDL abaixo, que descreve uma porta AND de duas entradas:

Observe no Código 1.3.1 a estrutura básica de um arquivo VHDL:

Código 1.3.1 Exemplo de uma porta AND em VHDL

```
1 library ieee;
2 use ieee.std_logic_1164.all;
3
4 entity AND2 is
5     port (
6         A, B : in  std_logic;
7         Y    : out std_logic
8     );
9 end AND2;
10
11 architecture rtl of AND2 is
12 begin
13     Y <= A and B;
14 end rtl;
```

- **Bibliotecas:** As linhas iniciais importam a biblioteca `ieee`, necessária para usar o tipo de dado padrão `std_logic`.
- **Entidade (Entity):** Define a interface do componente, ou seja, suas entradas (A, B) e saídas (Y). É a "caixa preta" vista de fora.
- **Arquitetura (Architecture):** Descreve o comportamento interno. Neste caso, a saída Y recebe o resultado da operação lógica AND entre A e B.

Além da descrição para síntese (criação do circuito físico), o VHDL é amplamente utilizado para **simulação**. Isso permite que os desenvolvedores validem o comportamento do circuito no computador antes de implementá-lo no PLD, economizando tempo e garantindo a confiabilidade do projeto.

O código a seguir apresenta um exemplo simples de testbench em VHDL, utilizado para simular o comportamento da porta AND definida anteriormente:

1.4 Fluxo de Projeto (Design Flow)

Código 1.3.2 Exemplo de Testbench para a porta AND

```
1 library ieee;
2 use ieee.std_logic_1164.all;
3 use ieee.numeric_std.all;
4 entity tb_AND2 is
5 end tb_AND2;
6 architecture behavior of tb_AND2 is
7     signal A, B : std_logic := '0';
8     signal Y      : std_logic;
9
10    component AND2
11        port (
12            A, B : in  std_logic;
13            Y    : out std_logic
14        );
15    end component;
16 begin
17    uut: AND2 port map (A => A, B => B, Y => Y);
18    process
19    begin
20        A <= '0'; B <= '0'; wait for 10 ns;
21        A <= '0'; B <= '1'; wait for 10 ns;
22        A <= '1'; B <= '0'; wait for 10 ns;
23        A <= '1'; B <= '1'; wait for 10 ns;
24        wait;
25    end process;
26 end behavior;
```

Capítulo 2

Introdução

Esta apostila apresenta conceitos fundamentais de VHDL utilizados na disciplina.

2.1 Primeiro Exemplo

A seguir temos um exemplo simples de entidade e arquitetura em VHDL.

Código 2.1.1 Exemplo de entidade e arquitetura

```
1 library ieee;
2 use ieee.std_logic_1164.all;
3
4 entity AND2 is
5     port (
6         A, B : in  std_logic;
7         Y    : out std_logic
8     );
9 end AND2;
10
11 architecture rtl of AND2 is
12 begin
13     Y <= A and B;
14 end rtl;
```

Podemos referenciar o código acima como Código 2.1.1.

2.2 Conclusão

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetur id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.