Universidade de São Paulo
Escola de Engenharia de São Carlos
Depto. de Engenharia Elétrica e de Computação

Introdução a VHDL

Aula 5

Professora Luiza Maria Romeiro Codá

Aula 5: Introdução a VHDL

Conteúdo:

- Esquemas de Iteração:

  Região de códigos concorrentes: comando GENERATE

  1. FOR GENERATE
  2. IF GENERATE

  Iteração de Componentes: FOR GENERATE

- Prática nº8: somador completo de 4 bits, usando os comandos FOR GENERATE e IF GENERATE

Esquemas de Iteração

Em VHDL é possível criar **esquemas iterativos** de geração. Com eles é possível repetir uma série de comandos, tanto concorrentes como sequenciais.

- ✓ Para região de códigos **concorrentes**, é utilizado o comando GENERATE,
- ✓ para região de códigos **sequenciais**, é utilizado o comando LOOP.

Para ambos os comandos de iteração, há dois esquemas:

- Um repete os comandos um número determinado de vezes
- o outro repete os comandos caso uma expressão de condição seja atendida.

É importante lembrar que os esquemas de iteração, assim como qualquer outro comando em VHDL, não são avaliados em **tempo de execução** ("*run-time*"), isto é, são avaliados apenas no momento da síntese da descrição e portanto, geram um circuito fixo.

Obs.: O termo "tempo de execução" é aqui usado figurativamente. Os dispositivos lógicos programáveis não executam nenhum código. As descrições são interpretadas pelo *software* e transformadas em um circuito físico real.

Esquemas de Iteração :GENERATE

O comando **concorrente** GENERATE utiliza dois esquemas de iteração para repetir comandos concorrentes:

- Esquema FOR
- Esquema IF

FOR GENERATE

Usado para: replicar circuitos;
replicar componentes.

O esquema FOR repete um conjunto de comandos uma quantidade determinada de vezes. É fornecida uma variável local e os limites para esta variável.

Sintaxe :

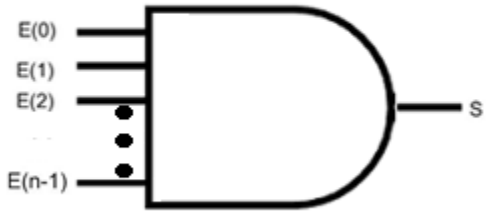
```
rótulo_obrigatório: FOR <variável_local> IN <limites_da_variável> GENERATE
                      -- Comandos concorrentes
                       END GENERATE rótulo_obrigatório;
```

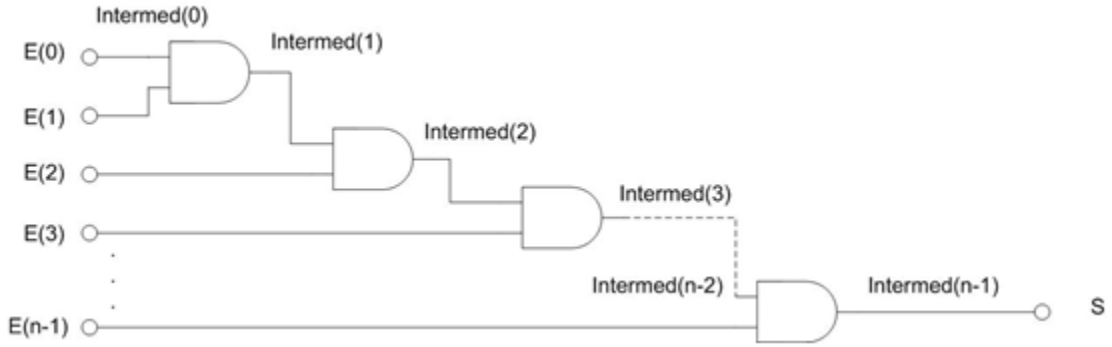
Por exemplo, o código abaixo repete os comandos concorrentes 4 vezes:

```
abc: FOR i IN 0 TO 3 GENERATE
    -- Comandos concorrentes
     END GENERATE abc;
```

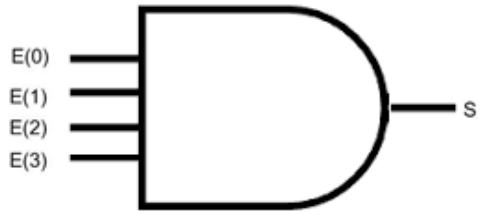
FOR GENERATE : Exemplo

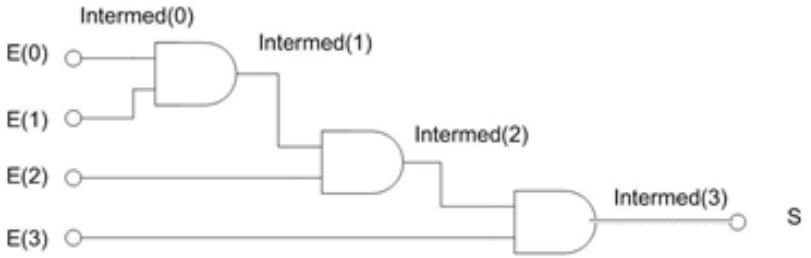
Porta AND descrita com quantidade de entradas variável utilizando uma declaração de GENERIC n



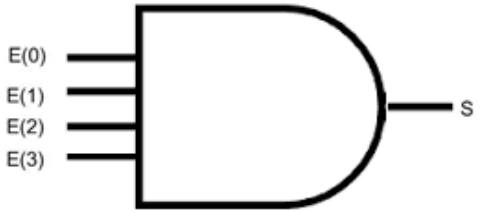


Porta AND descrita com 4 entradas

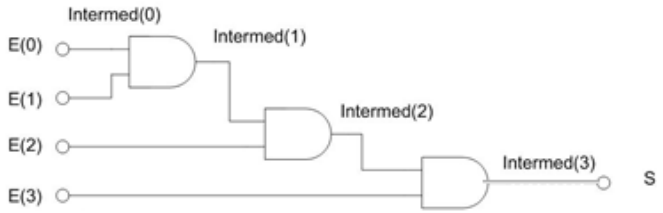




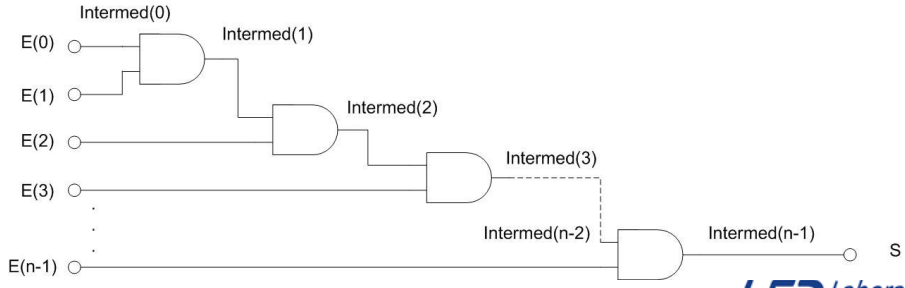
FOR GENERATE : Exemplo : Projeto de Porta AND descrita com quantidade de entradas variável




```
LIBRARY IEEE;
USE IEEE.std_logic_1164.ALL;

ENTITY AND_n_entradas IS
    GENERIC(n : NATURAL := 4); -- Números de pinos de entradas
    PORT(E : IN  STD_LOGIC_VECTOR(n-1 DOWNTO 0); -- vetor de 4 bits (3 DOWNTO 0)
         S : OUT STD_LOGIC);
END AND_n_entradas ;
```




```
ARCHITECTURE a OF AND_n_entradas IS
    -- Sinal intermediário para receber os resultados parciais
    SIGNAL Intermed : STD_LOGIC_VECTOR(n-1 DOWNTO 0); -- mesmo tamanho da entrada
BEGIN
    Intermed(0) <= E(0); -- Primeiro pino que é uma entrada

    and_for: FOR i IN 1 TO n-1 GENERATE -- Cria (n-1) portas AND

      Intermed(i) <= Intermed(i-1) AND E(i); --Intermed(1) <= Intermed(0) AND E(1);
                                             -- Intermed(2) <= Intermed(1) AND E(2);
                                              -- Intermed(3) <= Intermed(2) AND E(3);

    END GENERATE and_for;
      S <= Intermed(n-1); -- Saída

END a;
```

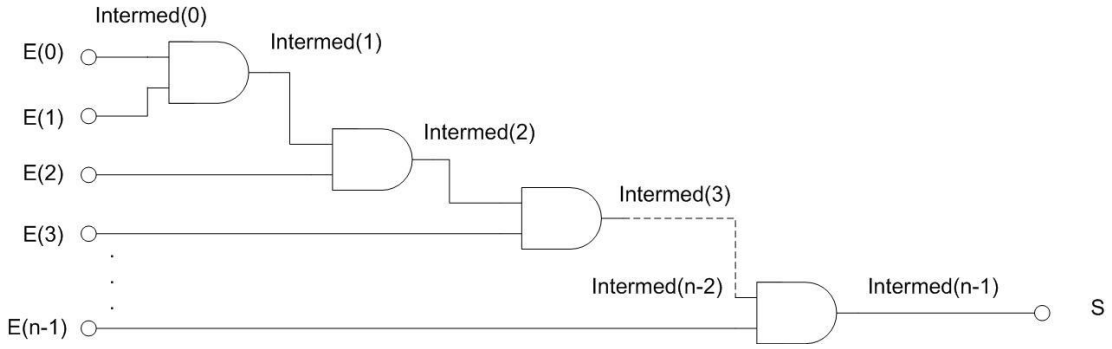


GENERATE – FOR : Exemplo

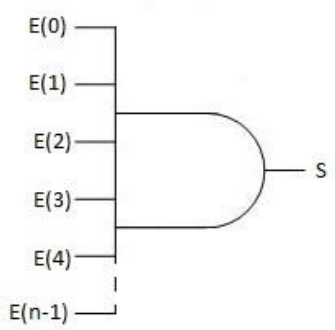
É necessário atribuir separadamente os valores de `Intermed(0)` e `S` através das linhas:

```
Intermed(0) <= E(0) ; -- Primeiro pino
S <= Intermed(n-1);  -- Saída
```

Isto é devido ao fato de que as operações para `i = 0` e `i = n-1` não seguem a mesma regra de geração, pois não existe **`Intermed(-1)`** e **`Intermed(n-1)`** é desnecessário. Utilizando o esquema de geração IF é possível incluir os casos citados no comando GENERATE.



Circuito Descrito



Circuito Sintetizado

IF GENERATE

O esquema IF insere uma réplica de um conjunto de comandos caso a condição contida após a palavra reservada IF seja satisfeita. Sintaxe:

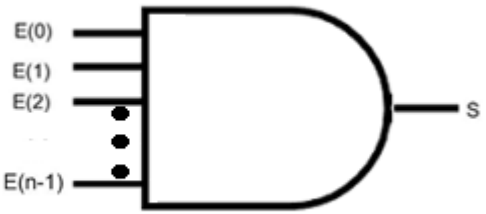
Sintaxe:

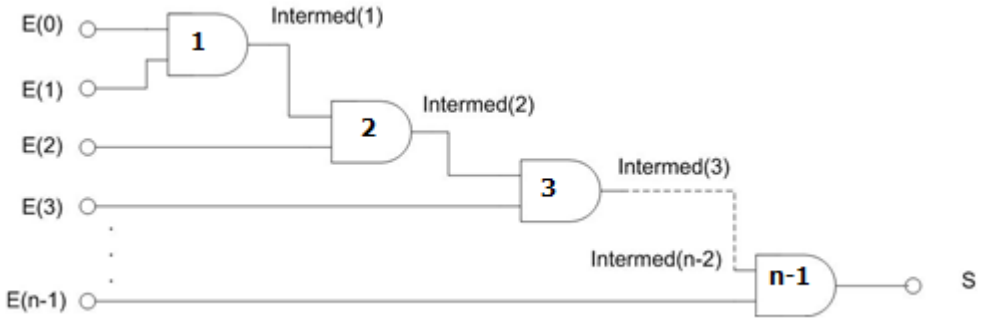
```
<rótulo_obrigatório>: IF <condição> GENERATE
                        -- Comandos concorrentes
                          END GENERATE < rótulo_obrigatório>;
```

IF GENERATE

Na criação de (n – 1) portas AND (figura abaixo), utilizando o esquema IF GENERATE é possível descrever a primeira e a última porta, as quais apresentam condições especiais, o que nâo é possível fazer utilizando o esquema FOR GENERATE.

- A porta 1 as entradas são dois pinos, E(0) e E(1) e a saída é um vetor intermed (intermed(1)),
- as portas 2 até a porta (n-2) apresentam entradas uma sendo pino a outra vetor intermed e sua saída sendo vetor intermed;
- A última porta também tem a condição diferente das outras pois a saída é um pino.

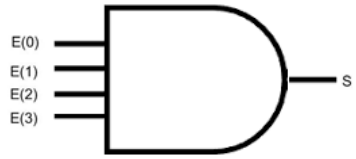


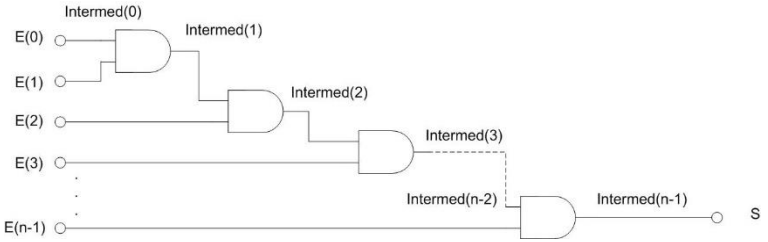


GENERATE – IF : Exemplo

Porta AND descrita com quantidade de entradas variável

```
LIBRARY IEEE;
USE IEEE.std_logic_1164.ALL;

ENTITY AND_n_entradas IS
    GENERIC(n : NATURAL := 4); -- Número de pinos de entradas
    PORT(E : IN  STD_LOGIC_VECTOR(n-1 DOWNTO 0);
         S : OUT STD_LOGIC);
END AND_n_entradas ;

ARCHITECTURE a` OF AND_n_entradas IS
    -- Sinal intermediário para receber os resultados parciais
    SIGNAL Intermed : STD_LOGIC_VECTOR(n-1 DOWNTO 0);
BEGIN

and_for: FOR i IN 0 TO n-1 GENERATE -- Cria (n-1) portas AND
        caso1: IF i = 0 GENERATE -- Bloco gerado se i = 0
            Intermed(i) <= E(i); -- Intermed(0) <= E(0)
               END GENERATE caso1;

        caso2: IF (i > 0) AND (i < n-1) GENERATE -- Se i ∈ [1,n-2]
            Intermed(i) <= Intermed(i-1) AND E(i);
               END GENERATE caso2;

        caso3: IF i = n-1 GENERATE -- Se i = n-1
            S <= Intermed(i-1) AND E(i);
               END GENERATE caso3;

         END GENERATE and_for;

END a;
```



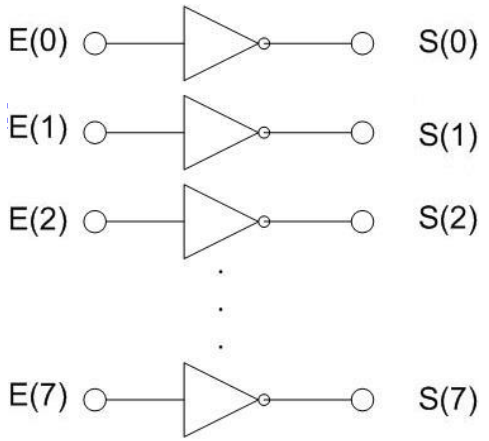


GENERATE – Instanciação Iterativa de Componentes

A chamada de componentes (PORT MAP) é também um comando concorrente e, portanto, pode-se utilizar o comando GENERATE para criar estruturas regulares utilizando componentes.

Exemplo: tomando um componente que seja uma porta NOT, pode-se criar uma estrutura que inverte todos os *bits* de um vetor utilizando comando FOR GENERATE:

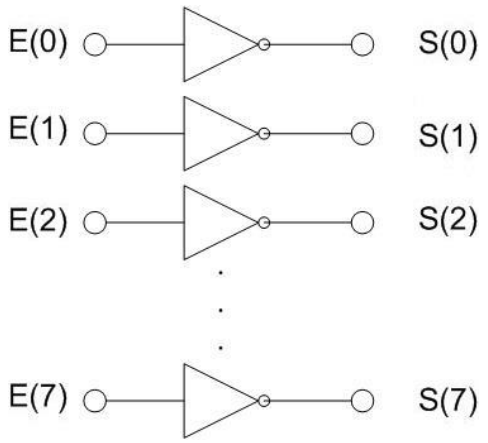
```
LIBRARY IEEE;
USE IEEE.std_logic_1164.ALL;
ENTITY not_vetor IS
    PORT(E : IN  STD_LOGIC_VECTOR(7 DOWNTO 0);
         S : OUT STD_LOGIC_VECTOR(7 DOWNTO 0));
END not_vetor ;

ARCHITECTURE a OF not_vetor IS
    COMPONENT porta_not IS
        PORT(entrada : IN  STD_LOGIC;
             saida   : OUT STD_LOGIC);
    END COMPONENT;

BEGIN
    for_not: FOR i IN 0 TO 7 GENERATE -- Cria 8 portas NOT
        x : porta_not PORT MAP(E(i),S(i));
    END GENERATE for_not;

END a;
```



Circuito Descrito

Projeto not_vetor completo (Instanciação Iterativa de Componentes)

```
LIBRARY IEEE;
USE IEEE.std_logic_1164.ALL;
ENTITY not_vetor IS
    PORT(E : IN  STD_LOGIC_VECTOR(7 DOWNTO 0);
         S : OUT STD_LOGIC_VECTOR(7 DOWNTO 0));
END not_vetor ;

ARCHITECTURE a1 OF not_vetor IS
    COMPONENT porta_not IS
        PORT(entrada : IN  STD_LOGIC;
             saida   : OUT STD_LOGIC);
    END  COMPONENT;

BEGIN
    for_not: FOR i IN 0 TO 7 GENERATE -- Cria 8 portas NOT
        x : porta_not PORT MAP(E(i),S(i));
    END GENERATE for_not;
END a!;
```



Circuito Descrito

```
 -- Descrição do projeto da porta NOT
 LIBRARY IEEE;
 USE IEEE.std_logic_1164.ALL;
 ENTITY porta_not IS
     PORT(entrada : IN  STD_LOGIC;
          saida   : OUT STD_LOGIC);
 END porta_not;
 ARCHITECTURE a2 OF porta_not IS
 BEGIN
 saida <= NOT (entrada);
 END a2;
```

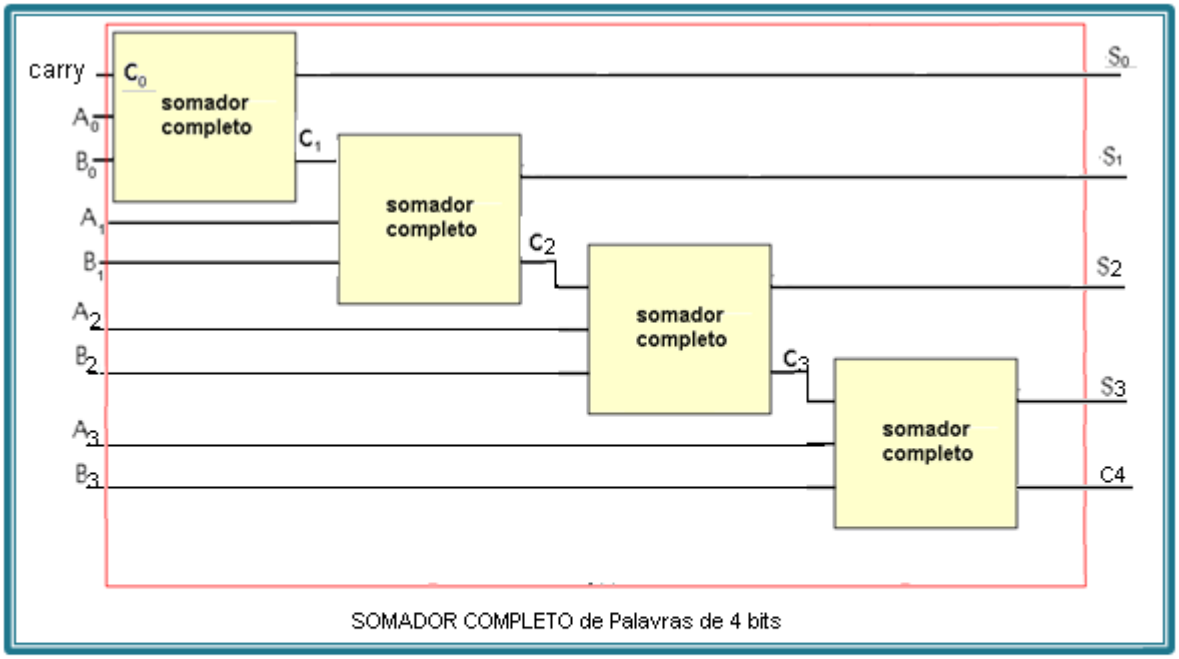
GENERATE – Aninhamento

É permitido aninhar comandos GENERATE, a fim de se criar estruturas multidimensionais: (Aula Futura)

Sintaxe:

```
linha: FOR i IN 0 TO 7 GENERATE
    coluna: FOR j IN 0 TO 7 GENERATE
        x : componente_abc PORT MAP(Entrada_a(i),Entrada_b(j),Saída(i+1));
    END GENERATE coluna;
END GENERATE linha;
```

Prática nº8
crie um projeto de um somador completo de palavras de 4 bits, utilizando GENERIC para determinar n bits.
8.1 Utilizando o comando GENERATE FOR
8.2 Utilizando o comando GENERATE IF



SOMADOR COMPLETO de Palavras de 4 bits