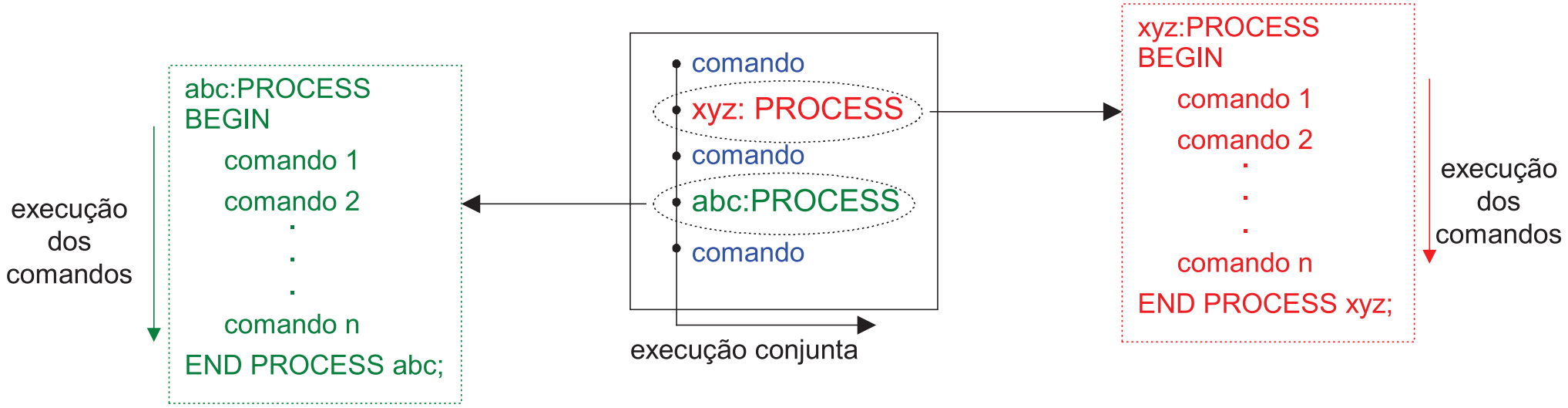
Comandos sequenciais básicos

Tópicos

- **Lista de sensibilidade em processos**
- **Atribuição de valor para um sinal – região sequencial**
- **Construção IF ELSE**
- **Construção CASE WHEN**
- **Comando WAIT**
- **Cuidados na descrição**
  - Comparações entre construções WHEN ELSE e IF ELSE
  - Construções IF ELSIF ELSE e CASE WHEN aninhadas
  - Emprego da construção IF ELSE e CASE WHEN

Comandos sequenciais

- **Comandos sequenciais podem ocorrer em:**
  - processos
  - subprogramas
- **PROCESS**
  - é um comando concorrente
  - delimita uma região contendo código sequencial
- **Lembrando**: uma descrição é composta de comandos concorrentes
  - todos são executados conjuntamente



Lista de sensibilidade em processos

- **Após o comando PROCESS**:
  - é possível declarar a lista de sensibilidade
- **Lista de sensibilidade**:
  - define quais sinais causam a execução do processo
- **A execução do processo ocorre se**:
  - um sinal da lista tem valor alterado
- **Iniciada a execução**:
  - os comandos são avaliados na sequência

```
   abc: PROCESS (lista de sensibilidade)
        BEGIN
           comando_1;
           comando_2;
              ..
           comando_n;
 END PROCESS abc;
```

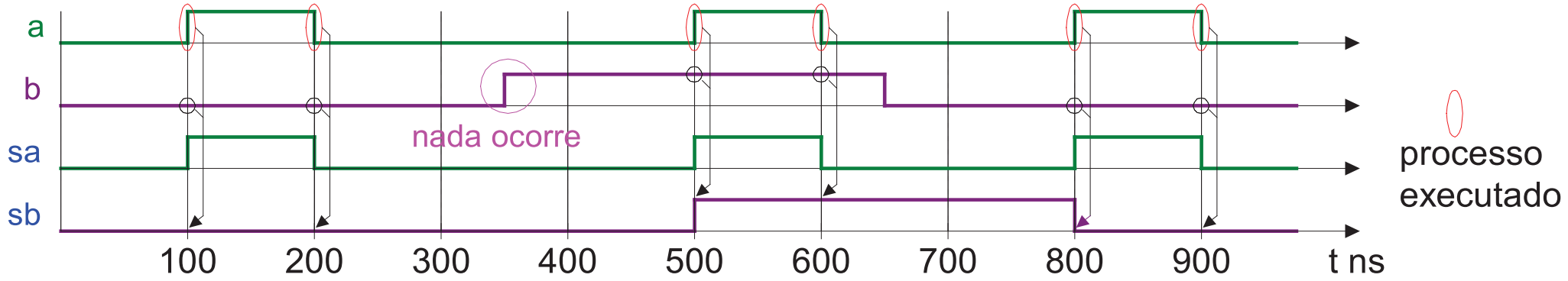
Exemplo: lista de sensibilidade

- **Valores dos sinais a e b transferidos para os sinais sa e sb**
- **Nesta descrição**: unicamente o sinal a na lista de sensibilidade
- **Consequência**:
  - alteração do valor no sinal a:
    execução do processo → valores de a e b transferidos para sa e sb
  - alteração do valor no sinal b:
    o processo não é executado → os valores de sa e sb são mantidos

```
 1 ENTITY sens_tes IS
 2   PORT (a, b :  IN  BIT;
 3         sa, sb  :  OUT BIT);
 4 END sens_tes;
 5
 6 ARCHITECTURE teste OF sens_tes IS
 7 BEGIN
 8   abc: PROCESS (a) -- executado na alteracao do valor de "a"
 9   BEGIN
10     sa <= a;
11     sb <= b;
12   END PROCESS abc;
13 END teste;
```

Exemplo: lista de sensibilidade

- **Simulação da entidade**




```
 1 ENTITY sens_tes IS
 2   PORT (a, b :  IN  BIT;
 3         sa, sb  :  OUT BIT);
 4 END sens_tes;
 5
 6 ARCHITECTURE teste OF sens_tes IS
 7 BEGIN
 8   abc: PROCESS (a) -- executado na alteracao do valor de "a"
 9   BEGIN
10     sa <= a;
11     sb <= b;
12   END PROCESS abc;
13 END teste;
```

Atribuição de valor para um sinal – região sequencial

- **A atribuição de valor para um sinal pode ocorrer em**:
  - regiões de código concorrente
  - regiões de código sequencial

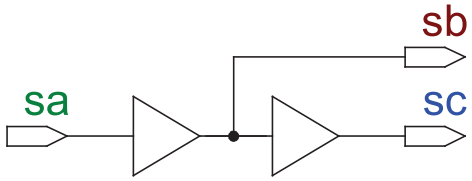
- **Região de código concorrente**
  - a atribuição é executada como um comando concorrente:
    - na ocorrência de uma alteração de valor de um sinal da expressão

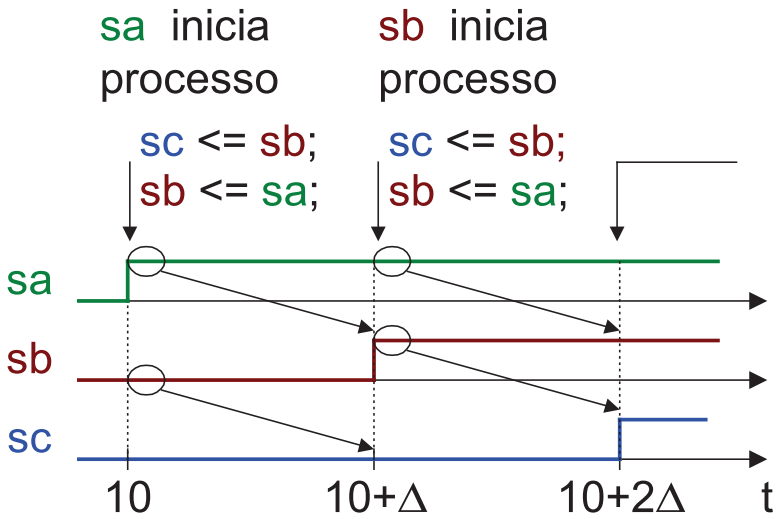
- **Região de código sequencial** (um processo, por exemplo)
  - a atribuição sempre ocorre quando o comando é executado

Atribuição de valor para um sinal – região sequencial

- **A atribuição do sinal ocorre Δ após a execução do comando**
- **A iteração dos comandos concorrentes ocorre na suspensão do processo**
- **Exemplo**: duas iterações ocorrem neste caso

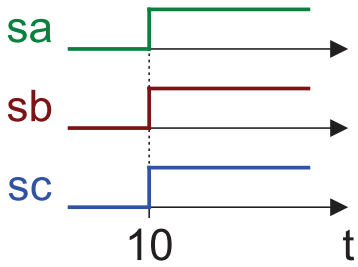
```
 6 ARCHITECTURE teste OF sig_tes0 IS
 7 BEGIN
 8   PROCESS (sa, sb)
 9   BEGIN
10    sc <= sb;
11    sb <= sa;
12   END PROCESS;
13 END teste;
```





fim das iterações:
sc não consta na
lista de sensibilidade

como
Δ=0



Construção IF ELSE

(similar: construção WHEN ELSE)

- **Execução condicional de um ou mais comandos sequenciais**
- **Teste**: definido por uma lista de condições
- **Primeira condição verdadeira**: define os comandos executados
- **Condição de teste**: qualquer expressão que retorne BOOLEAN
- **Início da construção**: comando **IF**
- **Cláusulas ELSIF e ELSE**: opcionais
- **Formato da construção**:

```
IF   condicao_1 THEN
  comando_sequencial;
  comando_sequencial;
ELSIF condicao_2 THEN          -- clausula ELSIF opcional
  comando_sequencial;
  comando_sequencial;
ELSIF condicao_3 THEN
  comando_sequencial;
ELSE                          -- clausula ELSE opcional
   comando_sequencial;
END IF;
```

Construção IF ELSE

- **É possível aninhar várias construções IF ELSE**
  - IF é também um comando sequencial

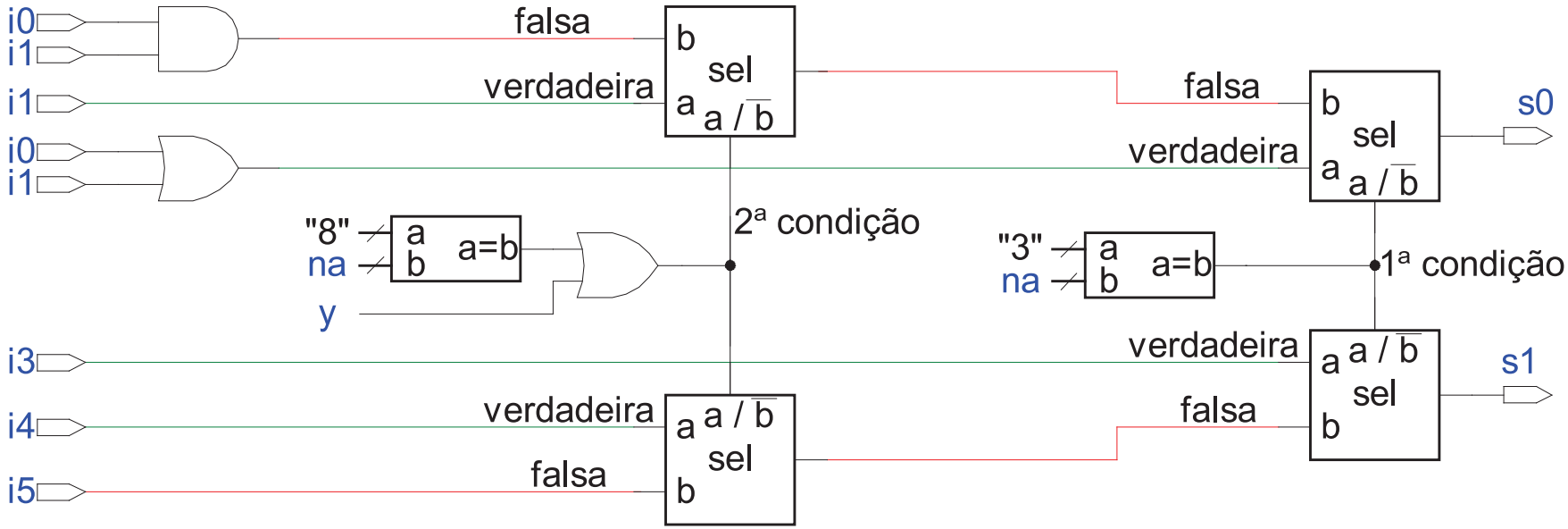
```
IF condicao_1 THEN
  IF condicao_2 THEN
    comando_sequencial;
  ELSE
    comando_sequencial;
  END IF;
ELSE
  IF condicao_3 THEN
    comando_sequencial;
  ELSE
    comando_sequencial;
  END IF;
END IF;
```

Construção IF ELSE

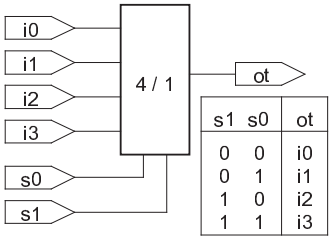
- **Exemplo**:

```
IF na =3 THEN
   s0 <= i0 OR i1;
   s1 <= i3;
ELSIF na =8 OR y ='1' THEN
   s0 <= i1;
   s1 <= i4;
ELSE
    s0 <= i0 AND i1;
    s1 <= i5;
END IF;
```

- **Circuito equivalente**:



Exemplo: circuito de seleção – IF ELSE



s1	s0	ot
0	0	i0
0	1	i1
1	0	i2
1	1	i3

- **Comando sequencial**
- **Nota**: s1 s2 são agrupados no sinal **sel**
- **Lista de sensibilidade**: sinais i0 i1 i2 i3 sel
  - remoção destes sinais: consequência?

```
 1 ENTITY mux_4a IS
 2   PORT (i0, i1, i2, i3  :  IN  BIT;  -- entradas
 3         s0, s1          :  IN  BIT;  -- selecao
 4         ot              :  OUT BIT); -- saida
 5 END mux_4a;
 6
 7 ARCHITECTURE teste OF mux_4a IS
 8   SIGNAL sel : BIT_VECTOR(1 DOWNTO 0);
 9 BEGIN
10   sel <= s1 & s0;
11   abc: PROCESS (i0, i1, i2, i3, sel) --sinal "sel" inserido na lista
12   BEGIN
13     IF    sel = "00" THEN  ot <= i0;
14     ELSIF sel = "01" THEN  ot <= i1;
15     ELSIF sel = "10" THEN  ot <= i2;
16     ELSE                   ot <= i3;
17     END IF;
18   END PROCESS abc;
19 END teste;
```

Construção CASE WHEN (similar: construção WITH SELECT)

- **Execução condicional de um ou mais comandos sequenciais**
- **Execução dos comandos**: controlada pelo valor de uma expressão
- **Todas as condições da expressão de escolha devem ser consideradas**
  - não existe uma prioridade como na construção IF ELSE
- **As opções podem ser agrupadas**: - caractere | equivale a "ou"
  - TO e DOWNTO delimitam faixas de opções
- **Opções restantes**: palavra reservada OTHERS
- **Formato da construção**:

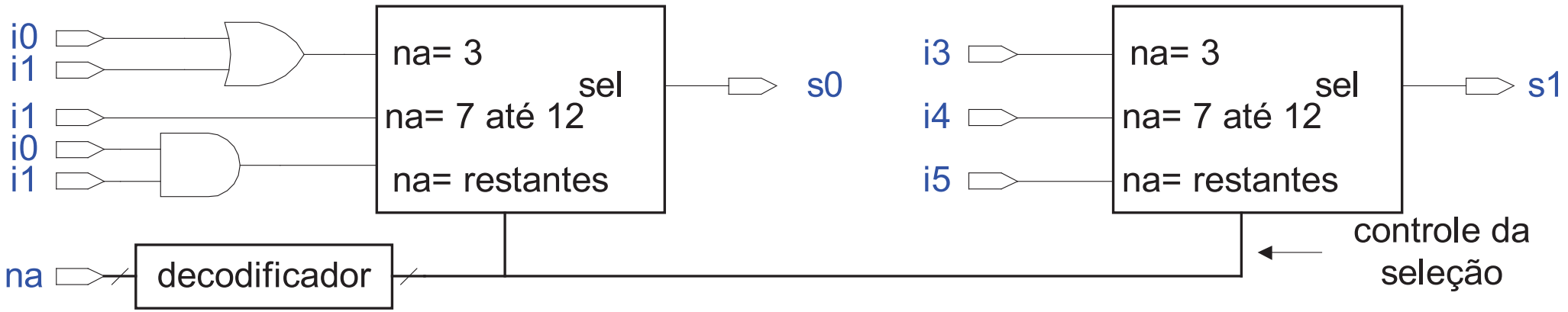
```
CASE expressao_escolha IS                                    -- expressao_escolha =

  WHEN condicao_1                 => comando_b; comando_c;          -- condicao_1

  WHEN condicao_2 | condicao_3    => comando_d;        -- condicao_2 ou condicao_3

  WHEN condicao_4 TO condicao_9 => comando_d;       -- condicao_4 ate condicao_9

  WHEN OTHERS                     => comando_e; comando_f;  -- condicoes restantes
END CASE;
```

Construção CASE WHEN

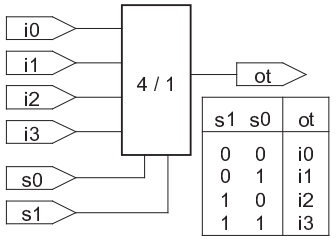
- **Exemplo**:

```
CASE na IS
  WHEN 3        =>  s0 <= i0 OR i1;    s1 <= i3;
  WHEN 7 TO 12  =>  s0 <= i1;          s1 <= i4;
  WHEN OTHERS   =>  s0 <= i0 AND i1;   s1 <= i5;
END CASE;
```

- **Circuito equivalente**:



Exemplo: circuito de seleção – CASE WHEN



- **Comando sequencial**
- **Nota**: s1 s2 são agrupados no sinal sel

```
 1 ENTITY mux_5a IS
 2   PORT (i0, i1, i2, i3  :  IN  BIT;
 3         s1, s0          :  IN  BIT;
 4         ot              :  OUT BIT);
 5 END mux_5a;
 6
 7 ARCHITECTURE teste OF mux_5a IS
 8   SIGNAL sel: BIT_VECTOR(1 DOWNTO 0);
 9 BEGIN
10   sel <= s1 & s0;
11   abc: PROCESS (i0, i1, i2, i3, sel) --sinal "sel" inserido na lista
12   BEGIN
13     CASE sel IS
14        WHEN "00"   => ot <= i0;
15        WHEN "01"   => ot <= i1;
16        WHEN "10"   => ot <= i2;
17        WHEN OTHERS => ot <= i3;
18     END CASE;
19   END PROCESS abc;
20 END teste;
```

Comando WAIT

- **WAIT**: suspende a execução de um processo
- **Opções**: 3 formas ou uma combinação delas

- **WAIT ON**: equivalente a uma lista de sensibilidade
  processo suspenso até a mudança de valor de sinais relacionados
- **WAIT UNTIL**: processo suspenso; aguarda condição booleana
- **WAIT FOR**: processo suspenso por período de tempo

```
WAIT ON lista_de_sensibilidade;
WAIT UNTIL condicao_booleana;
WAIT FOR expressao_de_tempo;

WAIT ON lista_sensibilidade UNTIL condicao_booleana FOR expressao_de_tempo;
```