Aula 11 – Configurações

Tópicos da aula

- **Uso de configuração**
- **Definição de componentes**
- **Uso de componentes**

Configuração (*configuration*)

- **Usada para fazer associações em modelos**
  - Associar uma entidade a um arquitetura
  - Associar um componente a um par entidade-arquitetura
- **Largamente usada em ambientes de simulação**
  - Facilita a exploração do espaço de projeto
- **Uso limitado em ambientes de síntese (nem sempre suportada)**

```
CONFIGURATION <identifier> OF <entity_name> IS
        FOR <architecture_name>
        END FOR;
END; -- (1076-1987 version)
END CONFIGURATION; -- (1076-1993 version)
```

Configuração (*configuration*)

- **Exemplos**

```
CONFIGURATION config OF mux_2x1 IS
  FOR structural
  END FOR;
END;
```

```
CONFIGURATION config OF mux_2x1 IS
  FOR behavior
  END FOR;
END;
```

Exemplo de modelo

- **Este modelo possui duas arquiteturas**
  - Estrutural
  - Comportamental
- **A configuração seleciona a arquitetura a ser usada**

```
ENTITY mux_2x1 IS
PORT (a, b : IN  BIT;  -- data inputs
      sel  : IN  BIT;  -- selector
      s    : OUT BIT); -- data output
END mux_2x1;
-------------------------------------
ARCHITECTURE structural OF mux_2x1 IS
BEGIN
  s <= (a AND NOT sel) OR
       (b AND sel);
END structural;
-------------------------------------
ARCHITECTURE behavior OF mux_2x1 IS
BEGIN
  PROCESS(a,b,sel)
  BEGIN
    IF (sel='0') THEN
      s <= a;
    ELSE
      s <= b;
    END IF;
  END PROCESS;
END behavior;
-------------------------------------
CONFIGURATION config OF mux_2x1 IS
  FOR behavior
  END FOR;
END;
```

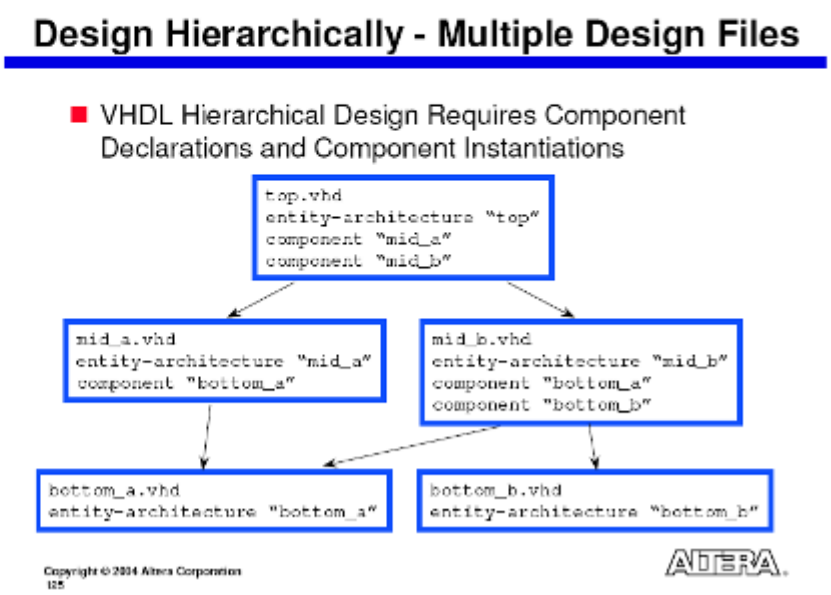
Exemplo de modelo

- **Este modelo possui duas arquiteturas**
  - Estrutural
  - Comportamental
- **A configuração seleciona a arquitetura a ser usada**

```
ENTITY mux_2x1 IS
PORT (a, b : IN  BIT;  -- data inputs
      sel  : IN  BIT;  -- selector
      s    : OUT BIT); -- data output
END mux_2x1;
-------------------------------------
ARCHITECTURE structural OF mux_2x1 IS
BEGIN
  s <= (a AND NOT sel) OR
       (b AND sel);
END structural;
-------------------------------------
ARCHITECTURE behavior OF mux_2x1 IS
BEGIN
  PROCESS(a,b,sel)
  BEGIN
    IF (sel='0') THEN
      s <= a;
    ELSE
      s <= b;
    END IF;
  END PROCESS;
END behavior;
-------------------------------------
CONFIGURATION config OF mux_2x1 IS
  FOR behavior
  END FOR;
END;
```

Componente (`component`)

- **Um modelo VHDL pode ser implementado de forma hierárquica, utilizando instâncias de componentes baseados em outros modelos VHDL**



Componente (`component`)

- **Declaração do componente**

```
COMPONENT <identifier> IS
PORT ( <signal_name> : <mode> <type>;
          ...
       <signal_name> : <mode> <type>);
END COMPONENT;
```

Componente (`component`)

- **Declaração do componente x declaração de entidade**

```
COMPONENT <identifier> IS
PORT ( <signal_name> : <mode> <type>;
         ...
       <signal_name> : <mode> <type>);
END COMPONENT;
```

```
ENTITY <identifier> IS
PORT ( <signal_name> : <mode> <type>;
         ...
       <signal_name> : <mode> <type>);
END <identifier>;
```

Componente (`component`)

- **Instanciação do componente**
  - Mapeamento dos *ports* da instância do componente com os sinais do modelo de nível mais alto
  - Mapeamento posicional (feito na ordem exata da declaração)

```
<label> : <component_identifier>
  PORT MAP(<signal_name>,..., <signal_name>);
```

  - Mapeamento nomeado (pode ser feito em qualquer ordem)

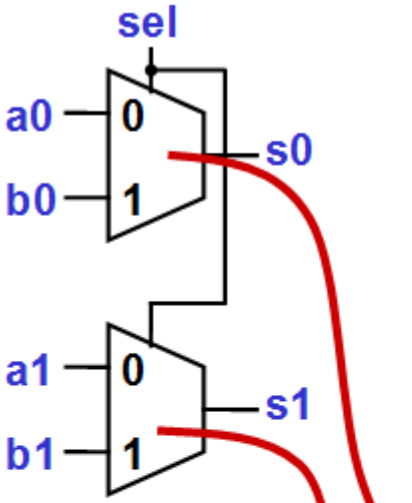
```
<label> : <component_identifier>
  PORT MAP(<component_signal_name> => <signal_name>,
           ...
           <component_signal_name> => <signal_name>);
```

Componente (component)

- **Onde declarar e instanciar o componente?**
  - Na arquitetura do modelo

```
ARCHITECTURE <identifier> OF <entity_identifier> IS
-- ...
-- Component Declarations
-- ...
BEGIN
-- ...
-- Component Instantiation Statements
-- ...
END <identifier> ; --(1076-1987 Version)
END ARCHITECTURE; -- (1076-1993 Version)
```

Exemplo: circuito com 2 mux




```
ENTITY mux_2x1_duo IS
PORT (SIGNAL a0, b0 : IN  BIT;  -- data inputs
      SIGNAL a1, b1 : IN  BIT;  -- data inputs
      SIGNAL sel    : IN  BIT;  -- selector
      SIGNAL s0, s1 : OUT BIT); -- data output
END mux_2x1_duo;

ARCHITECTURE structural OF mux_2x1_duo IS
  -- Component declaration
  COMPONENT mux_2x1
  PORT (SIGNAL a, b : IN  BIT;  -- data inputs
        SIGNAL sel  : IN  BIT;  -- selector
        SIGNAL s    : OUT BIT); -- data output
  END COMPONENT;

BEGIN
  -- Component instantiation
  u0 : mux_2x1 PORT MAP (a0, b0, sel, s0);
  u1 : mux_2x1 PORT MAP (sel => sel,
                         a   => a1,
                         b   => b1,
                         s   => s1);
END structural;
```

Resumindo....

- **Um modelo VHDL é formado por**
  - Entidade (`ENTITY`): interface
  - Arquitetura (`ARCHITECTURE`): corpo do modelo
- **A arquitetura pode ser modelada de forma**
  - Estrutural: “O que o circuito é”
  - Comportamental: “O que o circuito faz”
  - Híbrida (estrutural + comportamental)
- **Um modelo pode ter múltiplas arquiteturas (selecionadas pela configuração)**
- **Um modelo pode ser implementado usando instâncias de outros modelos (componentes)**

FIM AULA XI