



# Sistemas Electrónicos Digitales

## Tema #4

### Diseño mediante Lenguajes de Descripción Hardware

Parte 4.3d



- 4.1 Ventajas de los HDL.
- 4.2 Metodología de Diseño.
- **4.3 VHDL. Sintaxis de VHDL.**
- 4.4 Codificación de circuitos lógicos en VHDL.
- 4.5 Módulos IP.
- 4.6 Codiseño SW-HW.
- 4.7 SystemC



# VHDL: descripción estructural (I)

- Es la de menor nivel de abstracción.
- Describe el sistema en forma de nodos interconectados.
- Útil para:
  - Visualizar la estructura y jerarquía de un sistema.
  - Utilizar componentes de bibliotecas.
  - Interconectar componentes entre sí.



# VHDL: descripción estructural (II)

- Requiere:
  - Declarar los componentes y sus interfaces.
  - Definir las señales que interconectarán los componentes.
  - Relacionar cada componente con una entidad y una arquitectura.
  - Crear (*instantiate*) los componentes necesarios.



# VHDL: descripción estructural (III)

- Declaración de un componente:

**component** nombre [**is**]

[**generic** ({nom\_pam: tipo [:= por\_defecto];}\_1);]

[**port** ({nom\_prto: clase tipo;}\_1);]

**end component** [nombre];

- Va en la parte declarativa de la arquitectura (antes de **begin**).



# VHDL: descripción estructural (IV)

- Ejemplo declaración de componente:

```
architecture structural of up is
```

```
  component rom is
```

```
    generic (addr_width: positive := 14);
```

```
    port (
```

```
      oe: in std_logic;
```

```
      addr: in std_logic_vector(addr_width – 1 downto 0);
```

```
      data: out std_logic_vector(7 downto 0)
```

```
    );
```

```
  end component rom;
```

```
begin
```

```
  ...
```

# VHDL: descripción estructural (V)

- Creación (*instantiation*) de componentes:

id:

**[component]** nombre\_comp |

**entity** nombre\_e[(nombre\_arqt)] |

**configuration** nombre\_conf

**[generic map**({nom\_pam => valor, }<sub>1</sub>)**]**

**[port map**({[puerto =>] señal, }<sub>1</sub>)**];**

Única sintaxis  
válida en VHDL'87

- Va dentro del bloque begin/end de la arquitectura.



## VHDL: descripción estructural (VI)

- Es al crear un componente cuando se dan valores a sus genéricos y se realizan las conexiones de sus puertos.
- Los genéricos y puertos no asignados deben asignarse en otro lugar (especificación de configuración).





# VHDL: descripción estructural (VI)

- Ejemplos de creación de componentes:
  - u1: **and2 port map**(a, b, y);
  - u2: **entity or2 port map**(a => a, b => b, y => y);
  - u3: **component rom generic map**(addr\_width => 16);
  - u4: **configuration test\_conf**
    - generic map**(addr\_width => 14)
    - port map**(
      - oe => cs,
      - addr => addr\_bus,
      - data => data\_bus);



# VHDL: descripción estructural (VII)

- El componente sólo define una interfaz.
- Para dotarlo de funcionalidad hay que asociarlo a una entidad y una arquitectura.
- Para ello existe una sintaxis específica (especificación de configuración) útil para:
  - VHDL'87 donde no existe otra alternativa.
  - Realizar la misma asociación para varios componentes a la vez.
- Esta sintaxis requiere la declaración previa del componente.



# VHDL: descripción estructural (VIII)

- Especificación de configuración:

```
for {ref,}₁ | all | others: nombre_comp  
  use entity nombre_entidad[(nombre_arq)] |  
  use configuration nombre_conf  
  [generic map({nom_pam => valor, }₁)]  
  [port map({[puerto =>] señal, }₁)];
```

- **all**: todas las referencias del componente.
- **others**: referencias no asignadas hasta ahora.

# VHDL: descripción estructural (IX)

- Ejemplos de especificaciones de configuración:

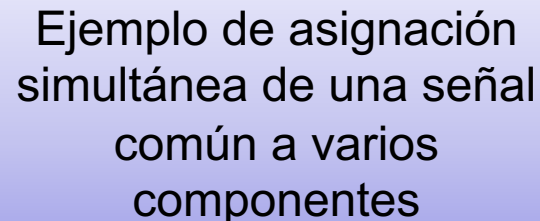
**for all:** inv **use entity** work.inverter;

**for** u8, u23: and2 **use entity** work.andgate(dataflow);

**for others:** and2 **use entity** work.andgate(behavioral);

**for all:** d\_reg **use entity** work.ffd(behavioral)

**port map**(clr\_n => global\_reset);



Ejemplo de asignación  
simultánea de una señal  
común a varios  
componentes



# VHDL: descripción estructural (X)

- Estructuras repetitivas:  
id:  
**for** param **in** rango **generate**  
  [declaraciones  
**begin**]  
  {sentencias\_concurrentes}  
**end generate** [id];



# VHDL: descripción estructural (XI)

- Repeticiones singulares:

id:

**if** condición **generate**

[declaraciones

**begin**]

{sentencias\_concurrentes}

**end generate** [id];



# VHDL: descripción estructural (XII)

- Ej.: registro cuádruple

```
entity reg_quad is
```

```
  port(clk: in bit; din: in bit_vector(3 downto 0);
```

```
        dout: out bit_vector(3 downto 0));
```

```
end;
```

```
architecture structural of reg_quad is
```

```
  component ffd is port(clk, d: in bit; q: out bit);
```

```
begin
```

```
  reg_d: for i in 0 to 3 generate
```

```
    ff: ffd port map(clk => clk, d => din(i), q => dout(i));
```

```
  end generate reg_d;
```

```
end;
```



# VHDL: descripción estructural (XIII)

- Ej.: contador binario asíncrono

```
entity async_cntr is
```

```
    port(clk: in bit; dout: out bit_vector(3 downto 0));
```

```
end;
```

```
architecture structural of async_cntr is
```

```
    component fft is port(clk: in bit; q: out bit); -- T flip-flop
```

```
    signal dout_i: std_logic_vector(dout'range);
```

```
begin
```

```
    cntr: for i in 0 to 3 generate
```

```
        ff0: if i = 0 generate -- First ff is different
```

```
            ff: fft port map(clk => clk, q => dout_i(i));
```

```
        end generate ff0;
```

```
        ffn: if i > 0 generate
```

```
            ff: fft port map(clk => dout_i(i-1), q => dout_i(i));
```

```
        end generate ffn;
```

```
    end generate cntr;
```

```
    dout <= dout_i;
```

```
end;
```





# VHDL: descripción estructural (XIV)

- Declaración de configuración:
  - Con las construcciones vistas hasta ahora la configuración de los componentes se realiza en la arquitectura.
  - Esto implica modificar el código de la arquitectura si se quieren usar distintas configuraciones de sus componentes.
  - El bloque de configuración saca la configuración fuera de la arquitectura de forma que cambiar la configuración de sus componentes no requiera modificar la arquitectura.



# VHDL: descripción estructural (XV)

- Declaración de configuración:  
**configuration** nombre **of** nombre\_entidad **is**  
[use\_clause | especificn\_atributo | declaración\_grupo]<sub>n</sub>  
{configuración\_bloque}<sub>1</sub>  
**end [configuration] [nombre];**



# VHDL: descripción estructural (XVI)

- Configuración bloque:

```
for nom_arq |  
    etiq_bloque |  
    etiq_generate [(rango | expresión estática)]  
    [use_clause]n  
    [configuración_bloque | especificación_configuración]n  
end for;
```



# VHDL: descripción estructural (XVII)

- Ejemplo de configuración:

**architecture** structural **of** processor **is**

**component** ALU **port**( ... ); **end component**;

**component** MUX **port**( ... ); **end component**;

**component** Latch **port**( ... ); **end component**;

**begin**

A1: ALU **port map**( ... );

M1: MUX **port map**( ... );

M2: MUX **port map**( ... );

M3: MUX **port map**( ... );

L1: Latch **port map**( ... );

L2: Latch **port map**( ... );

**end** structural;



# VHDL: descripción estructural (XVIII)

- Ejemplo configuración:

```
library TTL, work;  
configuration ASIC of processor is  
  use work.all;  
  for structural  
    for A1: ALU  
      use configuration TTL.SN74LS181;  
    end for;  
    for M1,M2,M3: MUX  
      use entity multiplex4 (Behavior);  
    end for;  
    for all: latch  
      -- use defaults  
    end for;  
  end for;  
end configuration ASIC;
```