

Sistemas Digitales

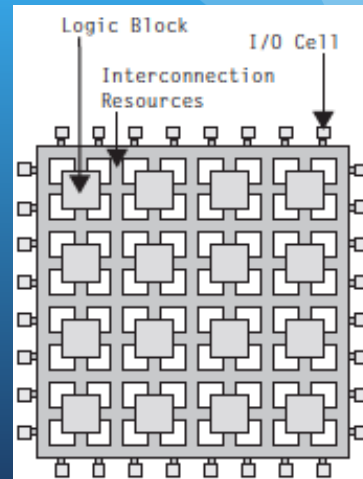
Lenguaje de Modelamiento de Hardware

FPGA

- Un FPGA (Field Programmable Gate Array) es un arreglo de compuertas reconfigurable.
- En estos dispositivos se pueden implementar tanto circuitos combinacionales como secuenciales y funciones lógicas multinivel.
- Algunos FPGAs incluyen funciones integradas como multiplicadores, elementos de lógica combinacional y arreglos de RAM.

FPGA - Arquitectura

- Cada fabricante tiene su propia arquitectura para el FPGA, pero en términos generales son sólo variantes de la versión base como se muestra en la figura.
- Los principales elementos de la arquitectura son:
 - Configurable Logic Blocks
 - Configurable I/O Blocks
 - Programmable Interconnect



FPGA

- Existen muchos fabricantes de FPGAs como Xilinx, Altera, Lattice, Microsemi, etc.
- En el desarrollo de este curso se empleará un FPGA de la marca Xilinx (familia Spartan).
- Debido a que los FPGAs son arreglos programables, se requiere de un lenguaje para poder “programar” los circuitos que deseamos implementar. El lenguaje que se empleará es VHDL (aunque también podemos emplear Verilog)

Sistemas Digitales

VHDL

VHDL - Conceptos

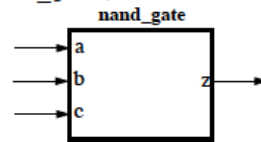
- Lenguaje de Modelamiento vs Programación
 - Código VHDL → FPGA
 - Código de Lenguaje de Programación → Microprocesador.
 - Sentencias concurrentes vs sentencias secuenciales.
- Características sobre VHDL
 - Entidad
 - Arquitectura
 - Objetos y tipos de datos
 - Procesos

VHDL - Entidad

- Establece las señales de Entrada/Salida del circuito a diseñar.

```
ENTITY entity_name IS
  [GENERIC (generic_list);]
  [PORT (port_list);]
END ENTITY [entity_name];
```

```
ENTITY nand_gate IS
  PORT(
    a : in  std_logic;
    b : in  std_logic;
    c : in  std_logic;
    z : out std_logic);
END ENTITY nand_gate;
```



VHDL - Entidad

- Tener en cuenta las palabras reservadas: ENTITY, IS, END, PORT
- No sensible a las mayúsculas/minúsculas
- El nombre de la entidad debe ser único
- Las sentencias acaban en punto y coma (;)
- Cada puerto debe tener un modo de funcionamiento: IN, OUT, INOUT y debe ser de algún tipo de dato (std_logic, std_logic_vector, etc).

VHDL - Arquitectura

- Describe la operación del circuito.

```

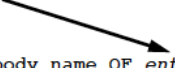
ARCHITECTURE body_name OF entity_name IS
  --this is the ->declarative area<-
  --declare signals, variables, components,
  --subprograms
BEGIN
  --this is the ->statement area<-
  --in here go statements that describe
  --organization or functional operation of
  --the component
  --this is the "execution part" of the model
END [body_name]

```

```

ENTITY entity_name IS
  --
ARCHITECTURE body_name OF entity_name IS

```



VHDL - Arquitectura

- Las palabras reservadas: ARCHITECTURE, OF, BEGIN
- El nombre de la arquitectura es único y no es sensible a mayúsculas/minúsculas. Todas las sentencias acaban en punto y coma (;)
- El área de declaraciones donde se pueden especificar, si se requiere, objetos como señales, constantes, variables.
- El área de sentencias, entre Begin y End, contiene la descripción del funcionamiento del circuito.
- Los comentarios se colocan antecediendo doble guion (--)

VHDL - Declaración de objetos

- Existen varios tipos de objetos que se pueden declarar dentro de la arquitectura: señales, constantes y variables.
- Por ejemplo:
 - CONSTANT retardo : TIME:= 10ns
 - VARIABLE suma : REAL
 - SIGNAL clock : STD_LOGIC
- Los objetos se pueden inicializar al efectuar la declaración.

VHDL - Declaración de objetos

- Los tipos de datos que se pueden utilizar son:
 - BIT, BIT_VECTOR, BOOLEAN, CHARACTER, STRING, SEVERITY_LEVEL, INTEGER, NATURAL, POSITIVE, REAL, TIME
 - STD_LOGIC, STD_LOGIC_VECTOR
 - Tipo numerado, es un tipo de datos donde el usuario asigna los valores.
 - Tipo compuesto, es un tipo de datos formados por datos de otros tipos. Se clasifican en:
 - ARRAY, es una colección de datos del mismo tipo.
 - RECORDS, es una colección de datos de distinto tipo.

VHDL - Estilos de Descripción

- Estilo Flujo de Datos.
 - Sentencias Concurrentes.
- Estilo Algorítmico (secuencial).
 - Procesos.
- Estilo Estructural.
 - Componentes.

Estilo Flujo de Datos

- Las asignaciones concurrentes son asignaciones (\leq) de valores a señales, fuera de proceso, que permiten modelar de una manera muy compacta lógica combinacional. Existen tres tipos de asignaciones:

- Simples: `s <= (a and b) + c;`
- Condicionales: `s <= a when c='1' else b;`
- Selectivas:

```
With a+b select
  s <= d when "0000",
    e when "1010",
    '0' when others;
```

Estilo Algorítmico (Procesos)

- Un proceso describe el comportamiento de un circuito y se considera una sentencia concurrente en si misma.
 - Su estado cambia cuando cambian las señales.
 - Utilizan construcciones secuenciales como `if...then...else, case`.
- Los procesos se “disparan” cuando una de las señales de su lista sensible cambia. Todas las entradas deben estar en la lista de sensibilidad.
- Las instrucciones dentro del proceso se ejecutan de manera secuencial, pero sin dar lugar a que avance el tiempo (proceso cuasi instantáneo).

Estilo Algorítmico (Procesos)

- Un proceso inicia con la siguiente estructura:

```
process (lista sensible)
begin
    (expresiones);
end process;
```

- En la lista sensible se incluyen todos los objetos que provocan el funcionamiento del proceso.
- En el área de expresiones se describe la secuencia de eventos que llevan a un resultado en el proceso.

Estilo Algorítmico (Procesos)

- Sentencia IF
 - Es similar a la asignación condicional si se usa una asignación por cada condición.
 - Es conveniente para múltiples asignaciones por condición.

```
IF condición then
    (sentencias si condición es verdadera);

[ELSIF condición THEN]
    (sentencias alternativas);

[ELSE]
    (sentencias si condición es falsa);

END IF;
```

Estilo Algorítmico (Procesos)

- Sentencia CASE
 - Se emplea para ejecutar un conjunto de sentencias secuenciales de acuerdo con el valor del selector.
 - Es parecido a una asignación selectiva.
 - Se deben evaluar todos los valores del selector.

```
CASE selector IS
    WHEN opción_1 =>
        (sentencias secuenciales);

    WHEN opción_2 =>
        (sentencias secuenciales);

    WHEN opción_3 =>
        (sentencias secuenciales);
    ...
    WHEN OTHERS =>
        (sentencias secuenciales);
END CASE;
```

Memoria implícita

- El estándar de VHDL establece que una señal permanecerá en su valor previo si no es asignada en un proceso.
- Durante el proceso de síntesis esto implica un estado interno o un elemento de memoria.
- Para evitar estos elementos se deben seguir las siguientes reglas:
 - Incluir todas las señales de entrada en la lista sensible.
 - Incluir `else` en una sentencia `if`
 - Asignar un valor a cada señal en cada caso.

Estilo Estructural (Componentes)

- Se implementan circuitos digitales a partir de módulos (componentes), efectuando las conexiones entre ellos y con la entidad que los agrupa.
- Los circuitos componentes deben haber sido procesados antes de ser empleados como parte de la estructura.
- Para emplear este estilo, se deben efectuar los siguientes pasos:
 - Declaración del componente en la arquitectura.
 - Instanciación del componente declarado.

Estilo Estructural (Componentes)

- La declaración de un componente emplea la siguiente sintaxis:

```
COMPONENT nombre_componente IS
  PORT (
    nombre_puerto: modo tipo;
    nombre_puerto: modo tipo;
    nombre_puerto: modo tipo;
    ...);
END COMPONENT;
```

- Como se puede apreciar, la estructura es similar a la creación de una entidad.
- Los detalles deben ser igual a la entidad del componente.

Estilo Estructural (Componentes)

- La instanciación de un componente se realiza en el cuerpo de la arquitectura

```
Etiqueta: nombre_componente PORT MAP(lista_puertos_conectados);
```

- La etiqueta proporciona una identificación para la instancia.
- El nombre del componente es el establecido en la declaración.
- La lista de puertos establece la conexión entre cada puerto del componente con la señal correspondiente en la arquitectura. Se deben conectar todos los puertos del componente, ninguno debe quedar al “aire”.