Universidad Nacional Autónoma de México
Facultad de Ingeniería
Diseño Digital VLSI

Introducción a los PLD

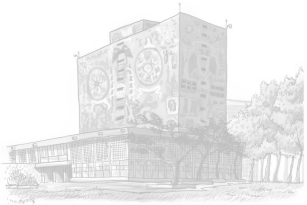
Uso del *VHDL*

M.I. Bryan Emmanuel Alvarez Serna

“Me lo contaron y lo olvidé; lo vi y lo entendí; lo hice y lo aprendí.”

Confucio

Contenido

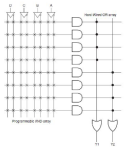


¿Qué es un PDL?

- PLD son siglas en inglés de *Programmable Logic Device*.
- Es un dispositivo sin función especifica y se puede reconfigurar después de su fabricación.

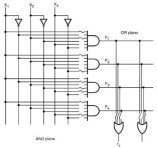
1970

Memoria PROM
Funciones lógicas especificas.



1978

PAL
Mayor conexión entre I/O



1981

PLA
Matriz AND y OR

1985

FPGA
Xilinx lanza el primer FPGA.

1990

CLPD
Altera lanza el CPLD.

PLD

Los PLD se clasifican por escala de integración; i.e., por el número de componentes.

Escala	Componentes
Small Scale Integration (SSI)	1 - 100
Medium Scale Integration (MSI)	100 - 1k
Large Scale Integration (LSI)	1k - 10k
Very Large Scale Integration (VLSI)	10k - 1M
Ultra Large Scale Integration (ULSI)	1M - 10M
Giga Scale Integration (GSI)	+ 10M

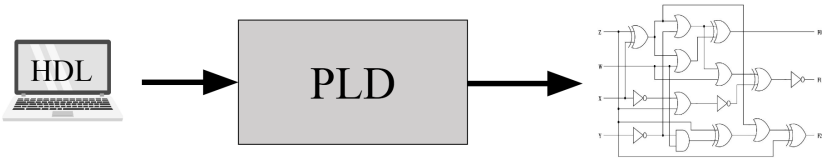
Ventajas de un PLD

- ✓ Flexibilidad y reconfigurabilidad.
- ✓ Personalización.

Los PLD se pueden configurar y reconfigurar con lenguajes de descripción de *hardware* (HDL por sus siglás en inglés de *Hardware Description Languaje*.

Algunos HLD

VHDL.
Verilog.
System Verilog.
SystemC.
AHDL.
MyHDL.
Handel-C.



¿Qué es describir?

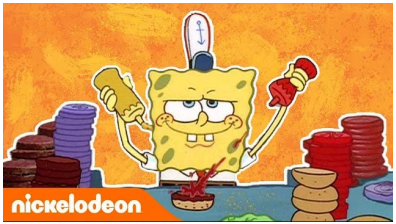
Descripción de *hardware*

- Describir: Especificar el comportamiento y la estructura de un sistema.
- Programar: Dar instrucciones, en un lenguaje de programación, para que un sistema realice tareas previamente definidas.

Descripción de *hardware*

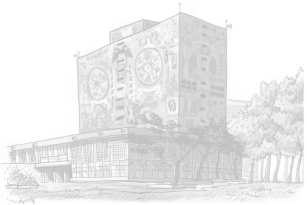
- Describir: Especificar el comportamiento y la estructura de un sistema.
- Programar: Dar instrucciones, en un lenguaje de programación, para que un sistema realice tareas previamente definidas.

Secuencial



Concurrente

Contenido



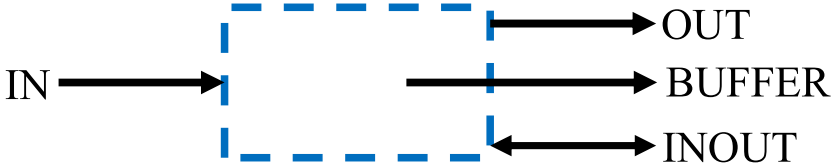
VHDL

VHDL: **V**ery High Speed Integrated Circuit (VHSIC) **H**ardware **D**escription **L**anguage.

Es un lenguaje para modelar y simular sistemas digitales como μP, ASIC y PLD.

- Estándares: IEEE 1076, IEEE 1164, IEEE 1076.2 y IEEE 1076.1.
- Versiones: VHDL-87, VHDL-93, VHDL-2002, VHDL-2008 y VHDL-2019.

Entidad y puertos



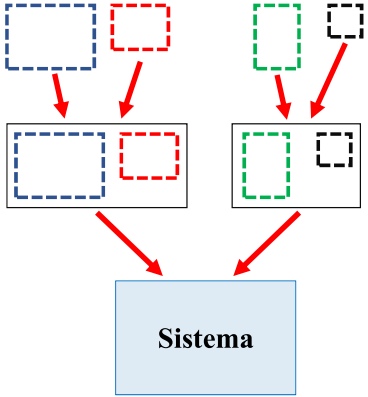
- IN: Sólo se puede leer.
- OUT: Sólo se le puede asignar un valor.
- BUFFER: Se puede leer y asignarle un valor.
- INOUT: Bidireccional.

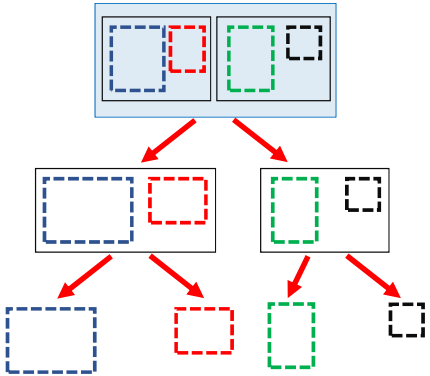
Sintaxis básica

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL; -- Operaciones básicas
USE IEEE.NUMERIC_STD.ALL; -- Operaciones aritméticas y conversión de datos

ENTITY nombre_entidad IS
        PORT(ENTRADA1: IN STD_LOGIC; -- Entrada de 1 bit
                ENTRADA2: IN STD_LOGIC_VECTOR(7 DOWNTO 0); -- Entrada de 8 bits
                ENTRADA3: IN BIT_VECTOR(0 TO 7); -- Entrada de 8 bits
                ENTRADA4: IN INTEGER RANGE 0 TO 255; -- Entero de 8 bits

                SALIDA: OUT INTEGER RANGE 0 TO 100; -- Salida de 7 bits

                BUF: BUFFER STD_LOGIC_VECTOR(4 DOWNTO 0); -- Vector de 5 bits

                IO: INOUT STD_LOGIC := '1'); -- Bidireccional de 1 bit
END ENTITY;

ARCHITECTURE nombre_aqr OF nombre_entidad IS
SIGNAL AUX: INTEGER RANGE 0 TO 127 := 0; -- Señal entera de 7 bits
SIGNAL FLAG: STD_LOGIC; -- Señal de 1 bit
CONSTANT VEL: INTEGER := 100; -- Constante
BEGIN

        -- Sentencias concurrentes o procesos

END ARCHITECTURE;
```

Diseño de sistemas en VHDL

Diseño *BOTTOM-UP*	Diseño *TOP-DOWN*





Diseño de sistemas en VHDL

Existen tres estilos para describir *hardware*.

- Flujo de datos: Descripción a nivel de transferencia entre registros (RTL siglas en inglés de *Register Transfer Level*).
- Algorítmica: También conocida como descripción comportamental, permite describir la funcionalidad en un nivel de abstracción alto.
- Estructural: Permite describir por separado y unir los bloques mediante código.

Metodología recomendada para describir.

1. Diseñar la entidad general.
2. Contemplar las subentidades necesarias.
3. Definir el tipo de descripción.

Ejemplo

Diseñar un MUX 2 a 1 de 3 bits.

Flujo de datos.

```vhdl
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;

ENTITY mux21 IS
        PORT(A,B: INTEGER RANGE 0 TO 7; -- Entradas
             SEL: IN STD_LOGIC; -- Selección
                  Z: OUT INTEGER RANGE 0 TO 7); -- Salida
END ENTITY;

ARCHITECTURE BEAS OF mux21 IS
BEGIN

        WITH SEL SELECT -- Asgno dependiendo SEL
                Z <= A WHEN '0',
                     B WHEN '1';
                     -- B WHEN OTHERS; -- También podría usar WHEN OTHERS


END ARCHITECTURE;
```

```vhdl
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;

ENTITY mux21 IS
        PORT(A,B: INTEGER RANGE 0 TO 7; -- Entradas
             SEL: IN STD_LOGIC; -- Selección
                Z: OUT INTEGER RANGE 0 TO 7); -- Salida
END ENTITY;

ARCHITECTURE BEAS OF mux21 IS
BEGIN

        Z <= A WHEN SEL = '0' ELSE
             B WHEN SEL = '1';

END ARCHITECTURE;
```

Ejemplo

Diseñar un MUX 2 a 1 de 3 bits.

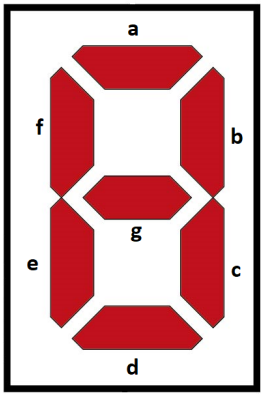
Algorítmica.

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;

ENTITY mux21 IS
        PORT(A,B: INTEGER RANGE 0 TO 7; -- Entradas
             SEL: IN STD_LOGIC; -- Selección
                  Z: OUT INTEGER RANGE 0 TO 7); -- Salida
END ENTITY;

ARCHITECTURE BEAS OF mux21 IS
BEGIN

        PROCESS(SEL)
        BEGIN
           IF SEL = '0' THEN
                Z <= A;
           ELSE
                Z <= B;
           END IF;
        END PROCESS;

END ARCHITECTURE;
```

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;

ENTITY mux21 IS
        PORT(A,B: INTEGER RANGE 0 TO 7; -- Entradas
             SEL: IN STD_LOGIC; -- Selección
               Z: OUT INTEGER RANGE 0 TO 7); -- Salida
END ENTITY;

ARCHITECTURE BEAS OF mux21 IS
BEGIN

        PROCESS(SEL)
        BEGIN
           CASE SEL IS
                WHEN '0' => Z <= A;
                WHEN '1' => Z <= B;
           END CASE;
        END PROCESS;

END ARCHITECTURE;
```

Ejercicio

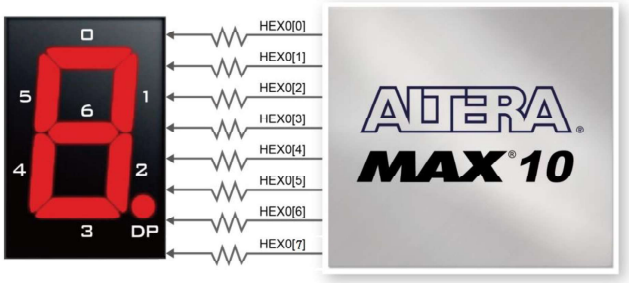
Diseñar un decodificador para las vocales e implementarlo en el HEX2 de la tarjeta DE10 Lite.



Ejercicio

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;

ENTITY decovoc IS
        PORT(VOCAL: IN INTEGER RANGE 0 TO 4;
                SEG: OUT STD_LOGIC_VECTOR(0 TO 7));
END ENTITY;

ARCHITECTURE BEAS OF decovoc IS
BEGIN

        WITH VOCAL SELECT
                SEG <= "00010001" WHEN 0, -- A
                       "01100001" WHEN 1, -- E
                       "00111111" WHEN 2, -- I
                       "00000011" WHEN 3, -- O
                       "10000011" WHEN 4, -- U
                       "11111111" WHEN OTHERS;

END ARCHITECTURE;
```

Ejercicio



HEX20	PIN_B20	S
HEX21	PIN_A20	S
HEX22	PIN_B19	S
HEX23	PIN_A21	S
HEX24	PIN_B21	S
HEX25	PIN_C22	S
HEX26	PIN_B22	S
HEX27	PIN_A19	S