UDS

LIBRO

DISEÑO LÓGICO

INGENIERÍA EN SISTEMAS COMPUTACIONALES.

SEXTO CUATRIMESTRE

Marco Estratégico de Referencia

Antecedentes históricos

Nuestra Universidad tiene sus antecedentes de formación en el año de 1979 con el inicio de actividades de la normal de educadoras "Edgar Robledo Santiago", que en su momento marcó un nuevo rumbo para la educación de Comitán y del estado de Chiapas. Nuestra escuela fue fundada por el Profesor Manuel Albores Salazar con la idea de traer educación a Comitán, ya que esto representaba una forma de apoyar a muchas familias de la región para que siguieran estudiando.

En el año 1984 inicia actividades el CBTiS Moctezuma Ilhuicamina, que fue el primer bachillerato tecnológico particular del estado de Chiapas, manteniendo con esto la visión en grande de traer educación a nuestro municipio, esta institución fue creada para que la gente que trabajaba por la mañana tuviera la opción de estudiar por las tardes.

La Maestra Martha Ruth Alcázar Mellanes es la madre de los tres integrantes de la familiaAlbores Alcázar que se fueron integrando poco a poco a la escuela formada por su padre, el Profesor Manuel Albores Salazar; Víctor Manuel Albores Alcázar en julio de 1996 como chofer de transporte escolar, Karla Fabiola Albores Alcázar se integró en la docencia en 1998, Martha Patricia Albores Alcázar en el departamento de cobranza en 1999.

En el año 2002, Víctor Manuel Albores Alcázar formó el Grupo Educativo Albores Alcázar S.C. para darle un nuevo rumbo y sentido empresarial al negocio familiar y en el año 2004funda la Universidad Del Sureste.

La formación de nuestra Universidad se da principalmente porque en Comitán y en toda laregión no existía una verdadera oferta Educativa, por lo que se veía urgente la creación de una institución de Educación superior, pero que estuviera a la altura de las exigencias de los

jóvenes que tenían intención de seguir estudiando o de los profesionistas para seguir preparándose a través de estudios de posgrado.

Nuestra Universidad inició sus actividades el 18 de agosto del 2004 en las instalaciones de la 4ª avenida oriente sur no. 24, con la licenciatura en Puericultura, contando con dos grupos de cuarenta alumnos cada uno. En el año 2005 nos trasladamos a nuestras propias instalaciones en la carretera Comitán – Tzimol km. 57 donde actualmente se encuentra el campus Comitán y el corporativo UDS, este último, es el encargado de estandarizar y controlar todos los procesos operativos y educativos de los diferentes campus, así como de crear los diferentes planes estratégicos de expansión de la marca.

Misión

Satisfacer la necesidad de Educación que promueva el espíritu emprendedor, aplicando altos estándares de calidad académica, que propicien el desarrollo de nuestros alumnos, Profesores, colaboradores y la sociedad, a través de la incorporación de tecnologías en el proceso de enseñanza-aprendizaje.

Visión

Ser la mejor oferta académica en cada región de influencia, y a través de nuestra plataforma virtual tener una cobertura global, con un crecimiento sostenible y las ofertas académicas innovadoras con pertinencia para la sociedad.

Valores

- Disciplina
- Honestidad
- Equidad
- Libertad

Escudo

El escudo del Grupo Educativo Albores Alcázar S.C. está constituido portres líneas curvas que nacen de izquierda a derecha formando los escalones al éxito. En la parte superior está situado un cuadro motivo de la abstracción de la forma de un libro abierto.

Eslogan

"Mi Universidad"

ALBORES

Es nuestra mascota, un Jaguar. Su piel es negra y se distingue por ser líder,trabaja en equipo y obtiene lo que desea. El ímpetu, extremo valor y fortaleza son los rasgos que distinguen.

DISEÑO LÓGICO

Objetivo de la materia:

Especificar sistemas secuenciales síncronos como autómatas de estados finitos. Implementar sistemas secuenciales síncronos utilizando bloques funcionales secuenciales (biestables, registros y contadores) y componentes combinacionales (puertas lógicas, memorias no volátiles y circuitos programables). Calcular el retardo de propagación y la frecuencia máxima de funcionamiento de una implementación dada de un sistema secuencial síncrono. Especificar circuitos combinacionales y secuenciales mediante el lenguaje de descripción de hardware VHDL.

UNIDAD I

INTRODUCCION

1.1.- Circuitos digitales.

1.2.- Circuitos combinacionales vs. Circuitos secuenciales.

1.3.- Sistemas secuenciales síncronos.

1.4.- El lenguaje de descripción de hardware VHDL.

1.4.1.- Origen y utilidad del lenguaje.

1.4.2.- Elementos básicos del lenguaje: tipos de datos y operadores

1.4.3.- Concepto y definición de Entity.

1.4.4.- Concepto y definición de Architecture.

1.4.5.- Sentencias concurrentes.

1.4.6.- Sentencias secuenciales: process.

UNIDAD II

SISTEMAS SECUENCIALES SÍNCRONOS

2.1.- Autómatas de estados finitos: Melay vs. Moore

2.2.- Especificación del sistema mediante diagramas y tablas de estados

2.3.- Representación comportamental del sistema mediante VHDL

2.4.- Implementación restructured de sistemas secuenciales síncronos

2.4.1.- Codificación de estados: Random, One-hot y Salidas igual a variables de estado

2.4.2.- Cálculo del circuito combinacional de excitación y salida: tabla de excitación y salida.

2.4.3.- Implementación de sistemas secuenciales síncronos mediante biestables y puertaslógicas.

2.4.4.- Retardo de propagación.2.4.5.-
Frecuencia máxima.

2.4.6.- Inicialización del sistema.

2.4.7.- Representación estructural del sistema mediante VHDL.

UNIDAD III

BLOQUES FUNCIONALES SECUENCIALES

3.1.- Contadores.

3.2.- Registros de desplazamiento.

3.3.- Registros conectados en anillo.

3.4.- Representación comportamental de bloques funcionales secuencial mediante VHDL.

3.5.- Implementación de sistemas secuenciales síncronos mediante bloques funcionalessecuenciales y puertas lógicas.

3.6.- Representación estructural de la implementación del sistema, basada en bloquesfuncionales secuenciales, mediante VHDL.

3.7.- Memorias no volátiles.

3.8.- Tipos de memorias no volátiles.

3.9.- Implementación de circuitos combinacionales mediante memorias no volátiles

3.10.- Representación comportamental de memorias no volátiles mediante VHDL.

3.11.- Implementación de sistemas secuenciales síncronos mediante bloques funcionalessecuenciales y memorias no volátiles.

3.12.- Representación estructural de la implementación del sistema, basado en memorias novolátiles, mediante VHDL.

UNIDAD IV

CIRCUITOS PROGRAMABLES

4.1. - Circuitos full custom y semicustom.

4.2.- Tipos de circuitos lógicos programables: Standard Cell, PLA/PAL, CPLD y FPGA

4.3.- Implementación de circuitos combinacionales mediante circuitos lógicos programablesde tipo PLA y PAL.

4.4.- Implementación de sistemas secuenciales síncronos mediante circuitos lógicosprogramables de tipo PLA y PAL.

Tabla de contenido

UNIDAD I

INTRODUCCION

1.1.- CIRCUITOS DIGITALES.

Los circuitos integrados son la base fundamental del desarrollo de la electrónica en la actualidad, debido a la tendencia a facilitar y economizar las tareas del hombre. Por esto es fundamental el manejo del concepto de circuito integrado, no sólo por aquellos que están en contacto habitual con este, sino también por las personas en general, debido a que este concepto debe de quedar inmerso dentro de los conocimientos mínimos deuna persona.

Un circuito integrado es una pieza o cápsula que generalmente es de silicio o de algún otro material semiconductor, que utilizando las propiedades de los semiconductores, es capaz de hacer las funciones realizadas por la unión en un circuito, de varios elementos electrónicos, como: resistencias, condensadores, transistores, etc.

Clasificación De Los Circuitos Integrados

Existen dos clasificaciones fundamentales de circuitos integrados(CI): los análogos y los digitales; los de operación fija y los programables; en este caso nos encargaremos delos circuitos integrados digitales de operación fija. Estos circuitos integrales funcionan con base en la lógica digital o álgebra de Boole, donde cada operación de esta lógica, es representada en electrónica digital por una compuerta.

La complejidad de un CI puede medirse por el número de puertas lógicas que contiene. Los métodos de fabricación actuales de fabricación permiten construir Cis cuya complejidad está en el rango de una a 105 o más puertas por pastilla.

Según esto los Cis se clasifican en los siguientes niveles o escalas de integración :SSI (pequeña escala) : menor de 10 puertas.

MSI (media escala): entre 10 y 100 puertas
LSI (alta escala): entre 100 y 10.000 puertas.
VLSI (muy alta escala): a partir de 10.000 puertas.

La capacidad de integración depende fundamentalmente de dos factores:

El ÁREA ocupada por cada puerta, que depende a su vez del tipo y del número de transistores utilizados para realizarla. Cuanto menor sea esta área mayor será la capacidad deintegración a gran escala.

El CONSUMO de potencia. En un circuito integrado se realizan muchas puertas en un espacio reducido. El consumo total del chip es igual al consumo de cada puerta por el número de puertas. Si el consumo de cada puerta es elevado se generará mucho calor en el chip debido al efecto Joule, de forma que si este calor no es disipado convenientemente se producirá un aumento de temperatura que puede provocar un funcionamiento anómalo de los circuitos.

¿Para qué los usamos?

Una de las ventajas aplicables al uso de circuitos de tipo digital es que tienen una aplicación directa en todo tipo de tecnologías. **Esto abarca la más amplia serie de recursos**, tantola electromecánica, como la magnética, la óptica, la mecánica o la microelectrónica. Para entender lo fundamental de su uso es bueno hacer especial hincapié en que este es el único tipo de circuito que da la oportunidad de que se integren, dentro de un mismo organismo,miles y miles de dispositivos de muy distinta índole. La cifra no solo abarca miles de terminales, sino millones, y en todos los casos se consigue un funcionamiento conjunto sinalterar el rendimiento, garantizando en todos casos que la velocidad de acceso será la necesaria.

¿Cómo deben ser?

Los circuitos digitales tienen una serie de rasgos coincidentes en todos los casos que los denominan y representan. Por ejemplo, dependiendo de qué tipo de valor binario se aplica enla entrada, la salida estará **representada por una serie de valores binarios específicos**.Otra de las características de estos circuitos se encuentra en que sus valores se cuantificande forma que se pueda llegar a los valores mencionados con anterioridad de 0 y 1. También es recomendable que los voltajes que salgan de las zonas delimitadas tengan un valor que resulte reducido, mientras que siempre debe darse el caso de que la contabilización de entrada y salida en cuanto a tensión produzca una ganancia que no sea superior a uno en el caso de los rangos válidos y superior a esta cantidad para el valor fuera de la zona de límite. Como uno de los últimos aspectos es necesario que cualquier cambio que se produzca en

una salida no produzca ningún tipo de efecto en las entradas y que cuando se produzca lasalida en uno de los circuitos a una cantidad de entradas superior a una.

¿Bajo qué categorías se clasifican?

Podemos clasificar los circuitos en dos grandes bloques: **por el diseño y por el tamaño**. En el primer grupo la filosofía que se adopta a la hora de dar forma a los circuitos deriva en que podamos ver distintos tipos de circuito, como los PLDs (dispositivos lógicos programables), los circuitos electrónicos MSI/SSI, las Standard Cells y los Gate Arrays, como principales exponentes.

La clasificación teniendo en cuenta al tamaño divide los circuitos de una manera distinta y produce una mayor variedad de ellos. En este grupo **se integran los circuitos digitales tipo SSI**, que se representan por disponer de una cantidad de puertas lógicas siempre inferior a diez. También es donde están los circuitos LSI, grupo en el cual la capacidad aumenta, dado que la cantidad de puertas lógicas puede llegar a ser incluso de 1000. Y en el medio está el tipo MSI, que se establece en un mínimo de diez, pero un máximo de cien puertas lógicas. Para los entornos más complejos y ambiciosos hay un cuarto tipo dependiendo de esta capacidad, tratándose del VLSI, circuitos en los que las puertas lógicas pueden superar las 1000 que se establecía como tope de los LSI. Eso sí, a través de los circuitos VLSI ya no se han impuesto limites, dado que las cifras de puertas pueden ser inluso de millones.

1.2.- CIRCUITOS COMBINACIONALES VS. CIRCUITOSSECUENCIALES.

Circuito combinacional. Está formado por funciones lógicas elementales (AND, OR, NAND, NOR, etc.), que tiene un determinado número de entradas y salidas. Es un circuito cuya salida depende solamente de la "combinación" de sus entradas en el momento que se está realizando la medida en la salida.

Los circuitos de lógica combinacional son hechos a partir de las compuertas básicas compuerta AND, compuerta OR, compuerta NOT. También pueden ser construidos con compuertas NAND, compuertas NOR, compuerta XOR, que son una combinación de las tres compuertas básicas.

Clasificación

Entre los circuitos combinacionales clásicos tenemos:

Lógicos

Generador/Detector de paridad

Multiplexor y Demultiplexor

Codificador y Decodificador

Conversor de código Comparador

Aritméticos

Aritméticos y lógicos

Unidad aritmética lógica

Estos circuitos están compuestos únicamente por puertas lógicas interconectadas entre sí.

Operación

La operación de los circuitos combinacionales se entiende escribiendo las ecuaciones booleanas y sus tablas de verdad.

Todos los circuitos combinacionales pueden representarse empleando álgebra de Boole a partir de su función lógica, generando de forma matemática el funcionamiento del sistema combinacional. De este modo, cada señal de entrada es una variable de la ecuación lógica desalida. Por ejemplo, un sistema combinacional compuesto exclusivamente por una puertaAND tendría dos entradas A y B. Su función combinacional seria F = A \cdot B, para una puerta OR sería F = A + B \,. Estas operaciones se pueden combinar formando funciones más complejas. Así, el siguiente esquema se define por la función indicada debajo del mismo.

Ejemplo de ecuación booleana: F = A . B + A . B.

Análisis

El comportamiento de los circuitos combinacionales sólo depende de las señales de entrada en un instante determinado, y no de la secuencia de entradas, es decir, de la historia del circuito. Este hecho no quiere decir que el comportamiento temporal no sea importante, de hecho, una de las principales características de los circuitos que se tienen en cuenta es la velocidad de operación o el retraso de propagación. En función de este retraso, podemos encontrar dos zonas temporales de operación bien diferenciadas: estado estacionario y estado transitorio.

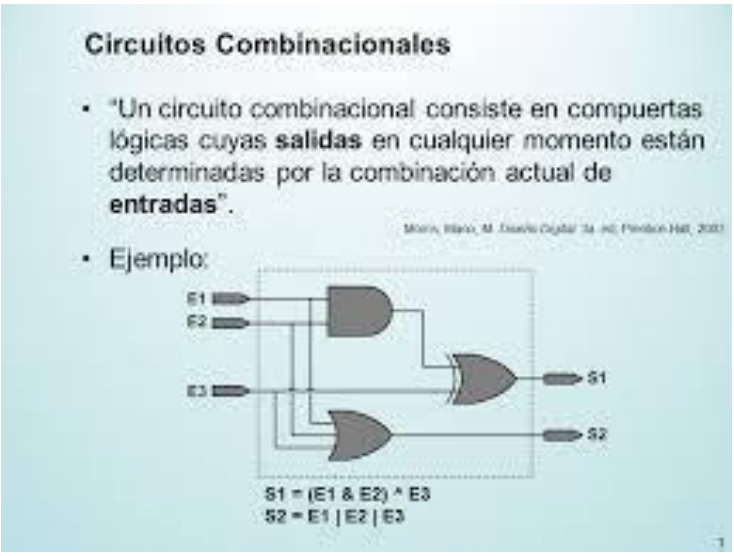
Una posible definición de estos estados sería la siguiente:

que la salida se estabilice.

En este estado, tanto las señales internas como las de salida pueden sufrir cambios (no necesariamente uno solo, sino que pueden ser varios), aunque las señales de entrada no cambien. Estos posibles cambios son los necesarios para que el circuito busque su estabilización.

El estado estacionario es aquel espacio temporal que va desde la estabilización del circuito lógico hasta que las entradas vuelvan a cambiar.

En este estado, ninguna de las señales del circuito puede sufrir ningún cambio, a no ser que sean las señales de entrada. Es decir, en el estado transitorio se producen todos los cambios necesarios en las señales de salida (e internas) hasta conseguir la estabilización del circuito. En cambio, en el estado estacionario, las señales de salida (e internas) están estables a su valor correcto. Por lo tanto, el comportamiento lógico hay que observarlo en el estado estacionario, en el cual no se producirá ningún cambio adicional debido al cambio actual de las señales de entrada.



entrada, sino también de la historia de las entradas anteriores se denomina Circuito Secuencial. Es decir, aquellos circuitos en que el contenido de los elementos de memoria sólo puede cambiar en presencia de un pulso del reloj. Entre pulso y pulso de reloj, la información de entrada puede cambiar y realizarse operaciones lógicas en el circuito combinacional, pero no hay cambio en la información contenida en las célulasde memoria.

Funcionalidad

Modelo clásico de un sistema secuencial

El circuito secuencial debe ser capaz de mantener su estado durante algún tiempo, para ello se hace necesario el uso de dispositivos de memoria. Los dispositivos de memoria utilizados en circuitos secuenciales pueden ser tan sencillos como un simple retardador (inclusive, se puede usar el retardo natural asociado a las compuertas lógicas) o tan complejos como uncircuito completo de memoria denominado multivibrador biestable o Flip Flop.

La salida del elemento de retraso es una copia de la señal de entrada retraso un determinado tiempo; mientras que la salida del elemento de memoria copia los valores de la entrada cuando la señal de control tiene una transición de subida, por lo que la copia no es exacta, sino que sólo copia lo que interesa. Por lo tanto, el modelo clásico de un sistema secuencial consta de un bloque combinacional, que generará la función lógica que queramos realizar, y un grupo de elementos de memoria con una serie de señales realimentadas.

Clasificación de los circuitos secuenciales

Los circuitos secuenciales se clasifican de acuerdo a la manera como manejan el tiempo:Circuitos secuenciales

sincrónicos

Circuitos secuenciales sincrónicos

En un circuito secuencial asíncrono, los cambios de estado ocurren al ritmo natural marcado por los retardos asociados a las compuertas lógicas utilizadas en su implementación, es decir, estos circuitos no usan elementos especiales de memoria, pues se sirven de los retardos propios (tiempos de propagación) de las compuertas lógicas usados en ellos. Esta manera de operar puede ocasionar algunos problemas de funcionamiento, ya que estos retardos naturales no están bajo el control del diseñador y además no son idénticos en cada compuerta lógica.

Circuitos secuenciales asincrónicos

Los circuitos secuenciales síncronos, sólo permiten un cambio de estado en los instantes marcados por una señal de sincronismo de tipo oscilatorio denominada reloj. Con esto sepueden evitar los problemas que tienen los circuitos asíncronos originados por cambios deestado no uniformes en todo el circuito.

Características de los circuitos secuenciales

Poseen uno o más caminos de realimentación, es decir, una o más señales internas o de salida se vuelven a introducir como señales de entradas. Gracias a esta característica se garantiza la dependencia de la operación con la secuencia anterior.

Como es lógico, existe una dependencia explícita del tiempo.

Esta dependencia se produce en los lazos de realimentación antes mencionados. En estoslazos es necesario distinguir entre las salidas y las entradas realimentadas. Esta distinción semediante dos elementos:

- ☐ Elementos de retraso, ya sean explícitos o implícitos debido al retraso de la lógicacombinacional. Este retraso es fijo e independiente de cualquier señal.
- ☐ Elementos de memoria, que son dispositivos que almacena el valor de la entrada enun instante determinado por una señal externa y lo mantiene hasta que dicha señal ordene el almacenamiento de un nuevo valor.

La diferencia de comportamiento entre ambos elementos radica en que la salida del elementode retraso es una copia de la señal de entrada; mientras que el elemento de memoria copia determinados instantes de la entrada (determinados por una señal externa), y no la señal completa, el resto del tiempo la salida no cambia de valor.

Aplicaciones de sistemas secuenciales

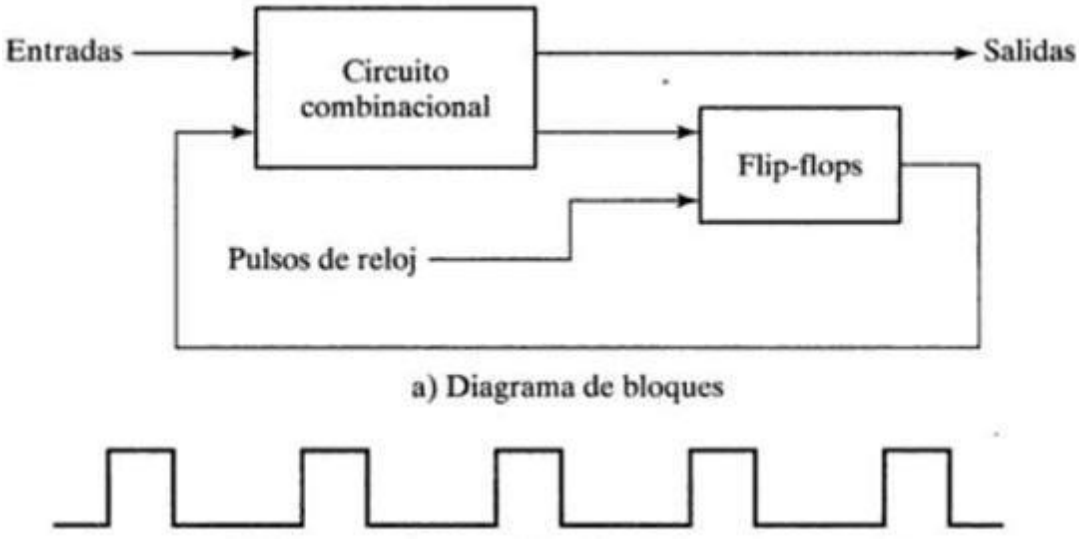
Como ya hemos comentado, los sistemas secuenciales forman un conjunto de circuitos muy importantes en la vida cotidiana. En cualquier elemento que sea necesario almacenar algún parámetro, es necesario un sistema secuencial. Así, cualquier elemento de programación (olo que es lo mismo, con más de una función) necesita un sistema secuencial.

1.3.- SISTEMAS SECUENCIALES SÍNCRONOS.

Un circuito secuencial síncrono utiliza señales que afectan a los elementos de almacenamiento únicamente en instantes discretos. La sincronización se logra con un dispositivo de temporización llamado generador de reloj, el cual produce un tren periódico de pulsos de reloj. Los pulsos d reloj se distribuyen por todo el sistema de modo que los pulsos de reloj se aplican con otras señales que especifican el cambio requerido en los elementos de almacenamiento.

Los circuitos secuenciales síncronos que usan pulsos síncronos que usan pulsos de reloj en las entradas de sus elementos de almacenamiento. Los circuitos secuenciales síncronos que usan pulsos de reloj en las entradas de sus elementos de almacenamiento se denominan circuitos secuenciales con reloj, y son del tipo que se usan más comúnmente en la práctica. Casi nunca manifiestan problemas de estabilidad y es fácil dividir su temporización en pasos discretos independientes, cada uno de los cuales se puede considerar por separado.

Los elementos de almacenamiento empleados en los circuitos secuenciales con reloj se llaman flip-flops. Un flip-flop es un dispositivo binario de almacenamiento que puede almacenar un bit de información. Un circuito secuencial podría usar muchos flip-flops para almacenar tantos bits como sea necesario: En la figura 1-2 se ilustra el diagrama de bloques de un circuito secuencial sincrónico con reloj.



a) Diagrama de bloques

Un Sistema Secuencial es un Sistema Digital cuyos vectores de salida dependen no sólo delvector de entrada actual sino también del anterior o los anteriores. En otras palabras, un Sistema Secuencial debe ser capaz de "memorizar" la evolución de los vectores de entrada y determinar el vector de salida en función de la misma.

1.4.- EL LENGUAJE DE DESCRIPCIÓN DE HARDWAREVHDL.

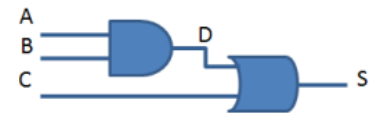
VHDL es un lenguaje de descripción de circuitos electrónicos digitales que utiliza distintosniveles de abstracción. El significado de las siglas VHDL es VHSIC (Very High Speed Integrated Circuits) Hardware Description Language. Esto significa que VHDL permite acelerar el proceso de diseño.

VHDL no es un lenguaje de programación, por ello conocer su sintaxis no implica necesariamente saber diseñar con él. VHDL es un lenguaje de descripción de hardware, que permite describir circuitos síncronos y asíncronos. Para realizar esto debemos:

- ☐ Pensar en puertas y biestables, no en variables ni funciones.
- ☐ Evitar bucles combinacionales y relojes condicionados.
- ☐ Saber qué parte del circuito es combinacional y cuál secuencial.
- ☐ ¿Por qué usar un lenguaje de descripción hardware?
- ☐ Poder descubrir problemas en el diseño antes de su implementación física.
- ☐ La complejidad de los sistemas electrónicos crece exponencialmente, es necesaria unaherramienta que trabaje con el ordenador.
- ☐ Permite que más de una persona trabaje en el mismo proyecto.

En particular VHDL permite tanto una descripción de la estructura del circuito (descripción a partir de subcircuitos más sencillos), como la especificación de la funcionalidad de un circuito

La misión más importante de un lenguaje de descripción HW es que sea capaz de simular perfectamente el comportamiento lógico de un circuito sin que el programador necesite imponer restricciones (ver ejemplo 1). En el ejemplo, una ejecución del código utilizando lasreglas básicas de cualquier lenguaje de programación al uso daría dos resultados diferentes sobre la misma descripción del circuito. Esto es debido a que en HW todos los circuitos trabajan a la vez para obtener el resultado (todo se ejecuta en paralelo) mientras que ensoftware el orden de las instrucciones delimita la actualización de las variables (ejecución secuencial de las instrucciones). Un lenguaje de descripción HW, VHDL o cualquier otro de los existentes en el mercado, nos debe dar el mismo resultado en simulación para los dosprogramas del ejemplo 1.



Prog 1	Prog 2
D = A and B; S = D or C;	S = D or C; D = A and B

Simulación Hardware		
t = 0ns	**t = 5ns**	**t = 10ns**
A = 0	A = 0	A = 1
B = 0	B = 1	B = 1
C = 0	C = 0	C = 0

	Simulación Software	
	Prog 1	Prog 2
S(5 ns)	0	0
S(10 ns)	1	0

Los circuitos descritos en VHDL pueden ser simulados utilizando herramientas de simulación que reproducen el funcionamiento del circuito descrito. Para la realización de la simulación existe un estándar aprobado por el ieee, en el cual se explican todas las expresiones propias de VHDL y cómo se simulan. Además, existen herramientas que transforman una descripción VHDL en un circuito real (a este proceso se le denomina síntesis). La sintaxis para síntesis ysu implementación final, aunque sigue unas normas generales, depende en gran medida de laherramienta de síntesis seleccionada.

1.4.1.- ORIGEN Y UTILIDAD DEL LENGUAJE.

Los estudios para la creación del lenguaje VHDL (VHSIC HDL) comenzaron en el año 1981, bajo la cobertura de un programa para el desarrollo de Circuitos Integrados de Muy Alta Velocidad (VHSIC), del Departamento de Defensa de los Estados Unidos. En 1983 las compañías Intermetrics, IBM y Texas Instruments obtuvieron la concesión de un proyecto para la realización del lenguaje y de un conjunto de herramientas auxiliares para su aplicación. Finalmente, en el año 1987, el lenguaje VHDL se convierte en la

norma IEEE-1076 –como todas las normas IEEE, se somete a revisión periódica, por lo que en 1993 sufrió algunas leves modificaciones.

El lenguaje VHDL fue creado con el propósito de especificar y documentar circuitos y sistemas digitales utilizando un lenguaje formal. En la práctica se ha convertido, en un gran número de entornos de CAD, en el HDL de referencia para realizar modelos sintetizables automáticamente. Las principales características del lenguaje VHDL se explican en los siguientes puntos:

- ☐ Descripción textual normalizada: El lenguaje VHDL es un lenguaje de descripción que especifica los circuitos electrónicos en un formato adecuado para ser interpretado tanto por máquinas como por personas. Se trata además de un lenguaje formal, es decir, no resulta ambiguo a la hora de expresar el comportamiento o representar la estructura de un circuito. Está, como ya se ha dicho, normalizado, o sea, existe un único modelo para el lenguaje, cuya utilización está abierta a cualquier grupo que quiera desarrollar herramientas basadas en dicho modelo, garantizando su compatibilidad con cualquier otra herramienta que respete las indicaciones especificadas en la norma oficial. Es, por último, un lenguaje ejecutable, lo que permite que la descripción textual del hardware se materialice en una representación del mismo utilizable por herramientas auxiliares tales como simuladores y sintetizadores lógicos, compiladores de silicio, simuladores de tiempo, de cobertura de fallos, herramientas de diseño físico, etc.

- ☐ Amplio rango de capacidad descriptiva: El lenguaje VHDL posibilita la descripción del hardware con distintos niveles de abstracción, pudiendo adaptarse a distintos propósitos y utilizarse en las sucesivas fases que se dan en el desarrollo de los diseños. Además, es un lenguaje adaptable a distintas metodologías de diseño y es independiente de la tecnología, lo que permite, en el primer caso, cubrir el tipo de necesidades de los distintos géneros de instituciones, compañías y organizaciones relacionadas con el mundo de la electrónica digital; y, en el segundo, facilita la actualización y adaptación de los diseños a los avances de la tecnología en cada momento.
- ☐ Otras ventajas: Además de las ventajas ya reseñadas también es destacable la capacidad del lenguaje para el manejo de proyectos de grandes dimensiones, las garantías que comporta su uso cuando, durante el ciclo de mantenimiento del proyecto, hay que sustituir componentes o realizar modificaciones en los circuitos, y el hecho de que, para muchas organizaciones contratantes, sea parte indispensable de la documentación de los sistemas.

1.4.2.- ELEMENTOS BÁSICOS DEL LENGUAJE: TIPOS DEDATOS Y OPERADORES

COMENTARIOS

Los comentarios van precedidos de dos guiones. En una línea se ignorará todo aquello quevaya después de dos guiones seguidos. Ejemplo:

--Esto es un comentario

Identificadores

Son cualquier cadena de caracteres que sirven para identificar variables, señales, procesos, etc. Puede ser cualquier nombre compuesto por letras (*aux*) o números y letras (*aux1, aux2, aux3, ...*), incluyendo el símbolo de subrayado "_". Las mayúsculas y minúsculas son consideradas iguales, por lo tanto, los identificadores *TMP* y *tmp* representan el mismo elemento. No es posible crear un identificador que coincida con alguna palabra reservada dellenguaje.

Números

Cualquier número se representa en base 10. Aunque es posible poner los números en otrasbases utilizando diferentes símbolos, como se muestra en la siguiente sección.

Bases

Para escribir un número se puede hacer en binario, octal, decimal y hexadecimal.Para vectores de bits:

"01111" binario
O"17" octal

X"F" hexadecimal

Constantes, señales y variables

En VHDL existen tres tipos de elementos: señales, constantes y variables. Estas dos últimas tienen un significado similar a cualquier otro lenguaje de programación.

Todos estos elementos son diferentes. Las variables sólo tienen sentido dentro de los procesos o subprogramas, mientras que las señales pueden ser declaradas en arquitecturas, paquetes o bloques concurrentes. Las constantes pueden ser declaradas en los mismos sitios que las variables y señales.

Constantes

Es un elemento que se inicializa con un valor determinado, el cual no puede ser modificado, es decir siempre conserva el mismo valor. Esto se realiza con la palabra reservada CONSTANT.

```
CONSTANT e : real := 2.71828;
    CONSTANT retraso : time := 10 ns;
```

Variables

Es lo mismo que una constante, pero con la diferencia que puede ser modificada en cualquier instante, aunque también es posible inicializarlas. La palabra reservada VARIABLE es la que permite declarar variables.

```
VARIABLE contador : natural := 0;
    VARIABLE aux : bit_vector(31 DOWNTO 0);
```

Señales

Las señales se declaran con la palabra reservada SIGNAL, a diferencia con las anteriores este tipo de elementos pueden ser de varios tipos: normal, register o bus. Es posible asignarles un valor inicial.

```
SIGNAL sel : bit := '0';
SIGNAL datos : bit_vector(7 DOWNTO 0);
```

Atributos

Los elementos como señales y variables pueden tener atributos, éstos se indican a continuación del nombre, separados con una comilla simple "'" y pueden incluir información adicional de algunos objetos desarrollados en VHDL, que servirán a las herramientas de diseño para obtener información a la hora de realizar una síntesis.

Existen muchos atributos, como LEFT, RIGHT, LOW, HIGH, RANGE, LENGTH... Pero el atributo más usado es EVENT, que indica si una señal ha cambiado de valor. Por ejemplo la siguiente sentencia captura un flanco de subida de una señal (clk).

```
....
if clk'event and clk = '1' then
....
```

Operadores

Los operadores que proporciona el lenguaje son:

Lógicos: Actúan sobre los tipos *bit, bit_vector y boolean*. En el caso de utilizar este tipo deoperadores en un vector, la operación se realizará bit a bit.

Operadores: AND, OR, NAND, NOR, XOR, XNOR y NOT.

***Aritméticos*:**

+ (suma o signo positivo): Sirve para indicar una suma entre dos números. También puedeactuar como símbolo si se sitúa delante de una expresión.

- (resta o signo negativo): Sirve para indicar la resta entre dos números. Si va delante de unaexpresión modifica el signo de la expresión.

* (multiplicación): Multiplica dos números de cualquier tipo.

/ (división): Divide dos números de cualquier tipo.

** (exponencial): Eleva un número a una potencia. El número de la izquierda puede serentero y real, pero el de la derecha sólo puede ser entero. Ejemplo: 4**2 sería 4^2.

ABS() (valor absoluto): Devuelve el valor absoluto de su argumento.MOD
(módulo): Calcula el módulo de dos números.

REM (resto): Calcula el resto de la división.

Relacionales: Siempre devuelven un valor booleano (true o false).

==, /= (igualdad): El primero devuelve verdadero si los operando son iguales y falso en caso contrario. El segundo indica desigualdad, funcionando al revés que el anterior.

>, >=, <, <= (menor mayor): Poseen el significado habitual (mayor que, mayor o igual que, menor que, menor o igual que, respectivamente). La diferencia con los anteriores reside en su uso, en este caso los tipos de datos que pueden manejar son siempre de tipo escalar omatrices.

Desplazamientos: (incluidas en la revisión de 1993)

SLL (Shift Left Logic) y SRL (Shift Right Logic), desplazamiento lógico a la izquierda ydesplazamiento lógico a la derecha, respectivamente, rellenando de ceros los huecos.

SLA (Shift Left Arithmetic) y SRA (Shift Right Arithmetic), desplazamiento aritmético a laizquierda y derecha respectivamente.

ROL (ROtate Left) y ROR (ROtate Right), rotación a la izquierda y a la derecharespectivamente. En este caso los huecos son ocupados por los bits que van quedando fuera. *Otros*:

es la suma de las dimensiones de las matrices con las que se opera.

Hay que decir que no todos los operadores pueden funcionar sobre todos los tipos de datos. También hay operadores que en determinadas circunstancias no pueden ser utilizados, por ejemplo al hacer código sintetizable no es recomendable usar multiplicadores (excepto si unode los operadores es potencia de dos, puesto que se trataría de un simple desplazamiento de bits).

El orden de preferencia, de mayor a menor es:

**, ABS, NOT

*, /, MOD, REM

+, - (signo)

+, -, & (operaciones)

=, /=, <, <=, >, >=

AND, OR, NAND, NOR, XOR

1.4.3.- CONCEPTO Y DEFINICIÓN DE ENTITY.

El tipo de entidad es el bloque de creación fundamental para describir la estructura de los datos con el Entity Data Model. En un modelo conceptual, los tipos de entidad se construyena partir de propiedades y describen la estructura de conceptos de nivel superior, como clientes y pedidos en una aplicación empresarial. Del mismo modo que una definición declase en un programa es una plantilla para las instancias de la clase, un tipo de entidad es unaplantilla para las entidades. Una entidad representa un objeto concreto (como un cliente o pedido concreto). Cada entidad debe tener una clave de entidad única dentro de un conjuntode entidades. Un conjunto de entidades es una colección de instancias de un tipo de entidad concreto. Los conjuntos de entidades (y los conjuntos de asociaciones) se agrupan de forma lógica en un contenedor de entidades.Los tipos de entidad admiten la herencia: es decir, un tipo de entidad se puede derivar deotro. Para obtener más información, vea Entity Data Model: herencia.

Un tipo de asociación (también denominado Asociación) es el bloque de creación fundamental para describir las relaciones en el Entity Data Model. En un modelo conceptual, una asociación representa una relación entre dos tipos de entidad (como Customer y Order). Cada asociación tiene dos extremos de asociación que especifican los tipos de entidad implicados en la asociación. Cada extremo de la asociación también especificauna multiplicidad de extremo de asociación que indica el número de entidades que pueden estar en ese extremo de la asociación. Una multiplicidad de extremo de asociación puede tener un valor de uno (1), cero o uno (0.. 1) o varios (*). Se puede tener acceso a las entidades de un extremo de una asociación a través de las propiedades de navegacióno a través de claves externas si se exponen en un tipo de entidad. Para obtener más información,consulte Foreign Key Property.

En una aplicación, una instancia de una asociación representa una asociación concreta (como por ejemplo una asociación entre una instancia Customer y una instancia Order). Las instancias de asociación se agrupan lógicamente en un conjunto de asociaciones. Los conjuntos de asociaciones (y los conjuntos de entidades) se agrupan de forma lógica enun contenedor de entidades.

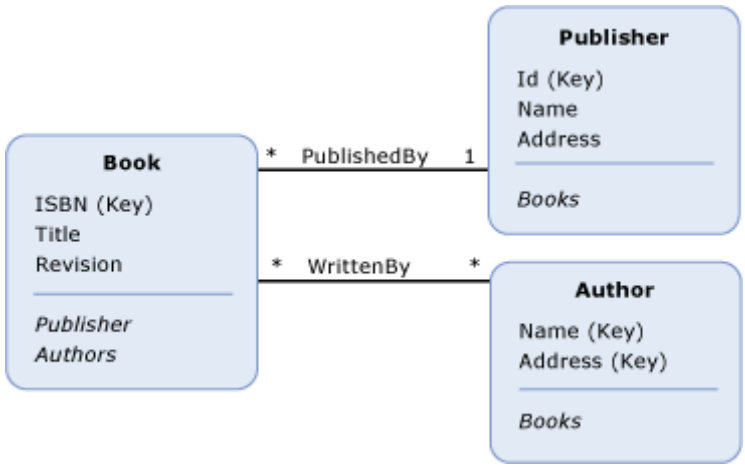
Propiedad.

Los tipos de entidad contienen propiedades que definen su estructura y suscaracterísticas. Por ejemplo, un tipo de entidad Customer puede tener propiedades como CustomerId, Name y Address.

Las propiedades en un modelo conceptual son análogas a las propiedades definidas en una clase en un programa. Del mismo modo que las propiedades en una clase definen la forma dela clase y proporcionan información sobre los objetos, las propiedades en un modelo conceptual definen la forma de un tipo de entidad y proporcionan información sobre las instancias del tipo de entidad.

Una propiedad puede contener datos primitivos (como una cadena, un entero o un valor booleano) o estructura los datos (como un tipo complejo). Para obtener más información, vea Entity Data Model: tipos de datos primitivos.

Un *modelo conceptual* es una representación específica de la estructura de algunos datos como entidades y relaciones. Una manera de representar un modelo conceptual es con un diagrama. El siguiente diagrama representa un modelo conceptual con tres tipos de entidad (Book, Publisher y Author) y dos asociaciones (PublishedBy y WrittenBy):



1.4.4.- CONCEPTO Y DEFINICIÓN DE ARCHITECTURE.

La Arquitectura la podemos definir como la estructura que describe el funcionamiento dela Entidad, de manera tal que nos permita el desarrollo de los procedimientos que se deberá llevar a cabo con la finalidad de que la Entidad cumpla con las condiciones de funcionamientoque queramos.

La principal ventaja que nos muestra VHDL para cuando deseamos definir una arquitectura, es la manera en que podemos describir nuestros diseños, es decir, mediante el algoritmo de programación que utilicemos podemos describir sistemas muy complejos. Los estilos de programación utilizados en el diseño de arquitecturas son:

- *Estilo Funcional*
- *Estilo por Flujo de Datos*
- *Estilo Estructural*

Descripción Funcional

Cuando utilizamos estilo funcional cuando exponemos la manera en que trabaja el sistema,es decir las descripciones consideran la relación que hay entre las entradas y las salidas del circuito, sin importar cómo este organizado en su interior, en este caso estudiaremos uncomparador.


```
1  library IEEE;
2  use  IEEE.STD_LOGIC_1164.ALL;
3  entity Comp is
4      Port ( A : in  STD_LOGIC_VECTOR (1 downto 0);
5             B : in  STD_LOGIC_VECTOR (1 downto 0);
6             C : out  STD_LOGIC);
7  end Comp;
8
9  architecture Funcional of Comp is
10 begin
11 compara: process (A,B)
12 begin
13  if A = B then
14   C <= '1';


15  else
16   C <= '0';
17 end if;
18 end process compara;
19 end Funcional;
```

Podemos observar que entre las líneas 1 a 7 declaramos tanto las Librerías como la Entidad, la ejecución del algoritmo (Arquitectura) se puede apreciar entre las líneas 8 a 18, este fragmento describe el funcionamiento del comparador.

Para iniciar la ejecución de la Arquitectura necesitamos definir un nombre arbitrario que podamos identificar (en nuestro caso fue la palabra Funcional) además debemos incluir la entidad a la cual lo relacionamos (Comp). En la línea 9 podemos observar el inicio de la sección donde se declaran los procesos que rigen la manera en que nuestro sistema se comporta. La declaración del proceso (línea 10) la utilizamos para la definición de algoritmos y empieza con una etiqueta opcional (Compara) seguida de dos puntos (:), la palabra reservada process enlaza la lista sensitiva (A y B) que hace referencia a las señales que determinan el funcionamiento delproceso.

Para finalizar el análisis, notamos que desde las líneas 12 a 17 el proceso ejecuta usando declaraciones secuenciales if-then-else (si-entonces-si no). Esto podemos interpretarlo de la siguiente manera: si el valor de la señal de entrada A es igual a la señal B, entonces C seráigual a '1', si no se asigna un '0' (el símbolo <= se lee como "se asigna a"). Ya terminado el proceso al usar la palabra reservada end process y opcionalmente el nombre del proceso (Compara), y similarmente añadimos la etiqueta (Funcional) al finalizar la arquitectura.

La Descripción Funcional es basada en el uso de procesos y declaraciones secuenciales, lascuales nos permiten declarar la función con rapidez.

Descripción por Flujo de Datos

La descripción por Flujo de Datos nos indica de qué manera que los datos pueden transferir desde una señal a otra sin necesitar declaraciones secuenciales (if-then-else). Este tipo de descripción nos permite definir el flujo que tomarán los datos entre módulos que se encargarán de realizar las operaciones. Podemos utilizar dos formatos: mediante when-else (cuando-si no) o por medio de ecuaciones booleanas.

a) Descripción por Flujo de Datos mediante when-else:

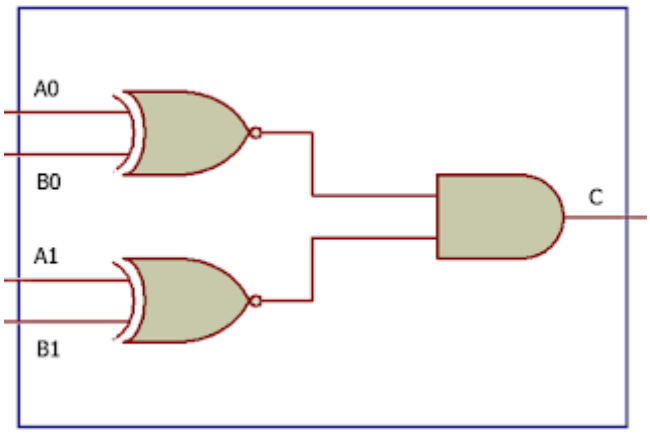
En el siguiente código podemos observar el comparador de igualdad de 2 bits descrito anteriormente. Observe que la principal diferencia es la eliminación del proceso y enla descripción sin declaraciones secuenciales.

```
1  library IEEE;
2  use IEEE.STD_LOGIC_1164.ALL;
3  entity Comp is
4      Port ( A : in  STD_LOGIC_VECTOR (1 downto 0);
5             B : in  STD_LOGIC_VECTOR (1 downto 0);
6             C : out  STD_LOGIC);
7  end Comp;
8
9  architecture F_datos of Comp is
10 begin
11   C <= '1' when (A =B) else '0'; --Se asigna a C el valor de 1
12    --cuando a=b si no sera 0
13 end F_Datos;
```

En el lenguaje VHDL manejamos dos tipos de declaraciones: las secuenciales y las concurrentes. La declaración secuencial que tiene la forma if-then-else se halla dentro del proceso en donde su ejecución debe seguir un orden específico para evitar que se pierda la lógica descrita.

b) Descripción por Ecuaciones Booleanas:

La otra forma de describir un circuito es a través de la obtención de sus ecuacionesbooleanas.

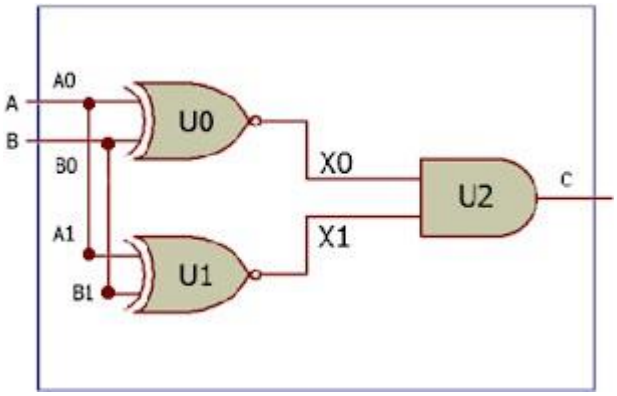


La descripción por flujo de datos en cualquiera de sus representaciones describe elcamino que los datos siguen al ser transferidos de las operaciones efectuadas entre las variables A y B a la variable C.

```
1  library IEEE;
2  use  IEEE.STD_LOGIC_1164.ALL;
3  entity Comp is
4      Port ( A : in  STD_LOGIC_VECTOR (1 downto 0);
5             B : in  STD_LOGIC_VECTOR (1 downto 0);
6             C : out  STD_LOGIC);
7  end Comp;
8
9  architecture booleana of Comp is
10 begin
11    C <= (A(1) xnor B(1))
12      and  (A(0) xnor B(0));
13 end booleana;
```

Descripción Estructural

Una descripción estructural es basada en su comportamiento en modelos lógicos establecidos (compuertas, sumadores, contadores,etc.). En la siguiente figura podemos observar un esquema del circuito comparador que hemos tomado como ejemplo, el cual esta conformado por compuertas NOR-exclusivas y una compuerta AND.



Para programar una entidad de manera estructural tenemos que dividir todo el diseño en submódulos (Jerarquizar), los cuales nos permitirán de manera más practica el circuito, esto debido a que conocemos la función de entrada/salida. En el caso del comparador, la salida delas dos compuertas XNOR es conocida (X0 y X1) estas son unidas a la compuerta AND loque nos da por resultado la salida C.

Enfatizo en la importancia de una jerarquía en VHDL se refiere al procedimiento de dividiren bloques y no a que un bloque tenga un mayor peso que otro. esto nos permite que la descripción estructural sea una manera sencilla de programar. en el contexto de diseño lógico podemos observar esto cuando analizamos por separado alguna parte de nuestro sistema.

```
1  library IEEE;
2  use IEEE.STD_LOGIC_1164.ALL;
3  entity Comp is
4      Port ( A : in  STD_LOGIC_VECTOR (0 to 1);
5             B : in  STD_LOGIC_VECTOR (0 to 1);
6             C : out  STD_LOGIC);
7  end Comp;
8  use work.compuerta.all; architecture
9  Estructural of Comp is
10 signal x: STD_LOGIC_VECTOR (0 to 1);
11 begin
12 U0: xnor2 port map (A(0), B(0), x(0);
13 U1: xnor2 port map (A(1), B(1), x(1);
14 U2: and2 port map (x(0), x(1), C);end
15 F_Datos;
```

En el código observamos que en la entidad declaramos solamente las entradas y salidas delcircuito (A,B y C). Los componentes XNOR y AND no los declaramos yaque están incluidos en el paquete de compuertas (Linea 8). En la linea 9 podemos observar que inicia la declaración de la Arquitectura Estructural. El algoritmo es propuesto describe dela siguiente manera: cada compuerta se maneja como un bloque lógico independiente (Componente) del diseño original, a estos bloques les asignamos una variable temporal(U0,

U1, U2); la salida de cada uno de estos bloques lo manejamos como una señal signal x (x0 y x1), las cuales declaramos dentro de la arquitectura y no en la entidad, debido a que no representa una terminal (pin) y y solo la utilizamos para conectar bloques de manera interna a la entidad.

1.4.5.- SENTENCIAS CONCURRENTES.

Se ejecutan de forma asíncrona unas respecto de las otras en el mismo tiempo de simulación. El orden en que se escriban es indiferente ya que no siguen un orden de ejecución predefinido. – No obstante, conviene escribir el código en el orden que mejor se pueda entender y mejor documente el programa.

- ☐ Sirven para especificar:
- ☐ Interconexiones entre componentes.
- ☐ Estructuras jerárquicas.
- ☐ Estructuras regulares.
- ☐ Transferencias entre registros.

1.4.6.- SENTENCIAS SECUENCIALES: PROCESS.

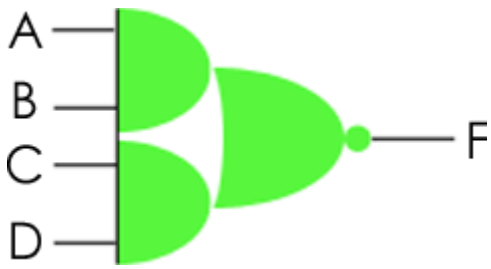
Los procesos solo se permiten dentro de una arquitectura. Las declaraciones dentro de losprocesos se ejecutan secuencialmente, no simultáneamente.

Los procesos se pueden escribir de varias maneras. El enfoque más común cuando se usanprocesos para describir diseños es usar el formulario que tiene una lista de sensibilidad.

Continuando con el tema anterior , utilizaremos dos procesos en nuestro diseño MUX_2,uno para reemplazar la compuerta AOI y el otro para reemplazar el inversor. Luego fusionaremos los dos procesos para ver cómo se puede usar un solo proceso para describirel diseño.

En primer lugar, abordemos la puerta AOI. Todo lo que hacemos es extraer la asignación de señal de la arquitectura v1 e insertarla en el proceso. En este punto, la asignación de señal esuna asignación de señal secuencial en lugar de una asignación de señal concurrente. En esta etapa, todo lo que hemos hecho es envolver un proceso alrededor de la asignación de señal.

```
v1_arch: proceso - incompleto en esta etapa
empezar
  F <= no ((A y B) o (C y D));
proceso finalizado;
```



Para completar este proceso, debemos recordar que estamos describiendo una pieza de lógica combinacional. Desde un punto de vista conceptual, las salidas de un circuito combinacional PUEDEN cambiar cuando CUALQUIERA de las entradas cambia. Esto significa que necesitamos hacer que el proceso se ejecute cuando alguna de las 'entradas' al proceso cambie. La manera de hacer esto es crear una lista de sensibilidad para el proceso de las señales A , B , C y D . La lista de sensibilidad sigue la palabra clave del proceso como se muestra:

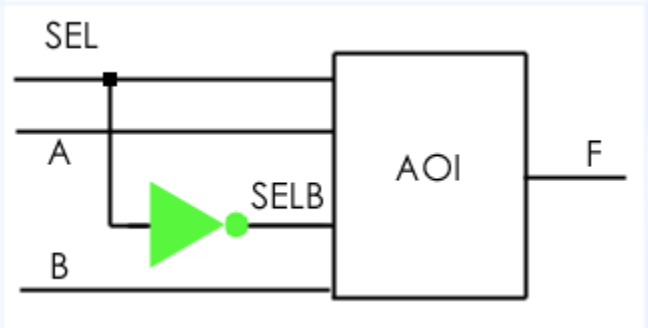
```
v1_arch: proceso (A, B, C, D)
empezar
  F <= no ((A y B) o (C y D));
proceso finalizado;
```

Sin embargo, dentro del diseño MUX_2, estamos utilizando los puertos y las señales de esediseño, por lo tanto:

```
G2: proceso (SEL, A, SELB, B)
empezar
  F <= no ((SEL y A) o (SELB y B));
proceso finalizado;
```

Del mismo modo para el inversor:

```
G1: proceso (SEL)
empezar
  SELB <= no SEL;
proceso finalizado;
```



Podemos combinar estos dos procesos en uno:

```
combinado: proceso (SEL, A, B)
empezar
  F <= no ((SEL y A) o ((no SEL) y B));
proceso finalizado;
```

Observe que no se utilizó SEL del proceso G1 para sustituir el SELB en el proceso G2 . Ahora tenemos un solo proceso con una lista de sensibilidad compuesta solo por los puertos MUX_2, sin señales internas.

En el próximo artículo de esta serie veremos cómo adoptar un enfoque más conceptual para los procesos de codificación, de modo que podamos usar más que solo asignaciones de señales para describir el comportamiento de un diseño, en este caso, el MUX_2.

Sistema Secuencial Síncrono

https://www.youtube.com/watch?v=iKRcemDeDic

Circuitos Secuenciales

https://www.youtube.com/watch?v=U3tXM5NTSrs

UNIDAD II

SISTEMAS SECUENCIALES SÍNCRONOS

2.1.- AUTÓMATAS DE ESTADOS FINITOS: MELAY VS.MOORE

En la teoría de la computación, una **Máquina de Mealy** es un tipo de máquina de estadosfinitos que genera una salida basándose en su estado actual y una entrada. Esto significa que el Diagrama de estados incluirá ambas señales de entrada y salida para cada línea de transición. En contraste, la salida de una máquina de Moore de estados finitos (el otro tipo) depende solo del estado actual de la máquina, dado que las transiciones no tienen entrada asociada. Sin embargo, para cada Máquina de Mealy hay una máquina de Moore equivalente cuyos estados son la unión de los estados de la máquina de Mealy y el Producto cartesiano de los estados de la máquina de Mealy y el alfabeto de entrada.

El nombre "Máquina de Mealy" viene del promotor del concepto: George H. Mealy, un pionero de las máquinas de estados, quien escribió *Un Método para sintetizar Circuitos Secuenciales*, Bell System Tech. J. vol 34, pp. 1045–1079, September 1955.

Las máquinas de Mealy suministran un modelo matemático rudimentario y eficiente para las máquinas de cifrado. Considerando el alfabeto de entrada y salida del alfabeto Latino, por ejemplo, entonces una máquina de Mealy puede ser diseñada para darle una cadena de letras(una secuencia de entradas), esto puede procesarlo en un string cifrado (una secuencia de salidas). Sin embargo, aunque se podría probablemente usar un modelo de Mealy para describir una Máquina Enigma, el diagrama de estados sería demasiado complejo para suministrar medios factibles de diseñar máquinas de cifrado complejas.

Una máquina de Mealy es una 6-tupla, $M=(S, S_0, \Sigma, \Lambda, T, G)$:

S es un conjunto finito de estados.

S_0 es un estado inicial, el cual es un elemento de S. $S_0 \in S$

Σ es un conjunto finito, llamado alfabeto de entrada.Λ es un conjunto finito, llamado alfabeto de salida. T es una función de transiciones ($T : S \times \Sigma \rightarrow S$)

G es una función de salida ($G : S \times \Sigma \rightarrow \Lambda$)

En la Teoría de la computación, una **Máquina de Moore** es un autómata de estados finitos para el cual la salida en un momento dado sólo depende de su estado en esemomento, mientras la transición al siguiente estado depende del estado en que se encuentrey de la entrada introducida. El diagrama de estados para una máquina Moore incluirá una señal de salida para cada estado. Comparada con la Máquina de Mealy, la cual mapea *transiciones* en la máquina a salidas.

El nombre **Moore machine** viene de su promotor: Edward F. Moore, un pionero de las máquinas de estados, quien escribió *Gedanken-experiments on Sequential Machines,* pp 129 –153, Estudios de Autómatas, Anuales de los Estudios Matemáticos, no. 34, Princeton University Press, Princeton, N. J., 1956.

La mayoría de las máquinas electrónicas están diseñadas como sistemas secuenciales síncronos. Los sistemas secuenciales síncronos son una forma restringida de máquinas de Moore donde el estado cambia solo cuando la señal de reloj global cambia. Normalmente el estado actual se almacena en Flip-flops, y la señal de reloj global está conectada a la entrada"clock" de los flip-flops. Los sistemas secuenciales síncronos son una manera de resolver problemas de meta estabilidad.

Una máquina electrónica de Moore típica incluye una cadena de Lógica combinacional para decodificar el estado actual en salidas (lambda). El instante en el cual el estado actual cambia, aquellos cambios se propagan a través de la cadena. y casi instantáneamente las salidas cambian (o no cambian). Hay técnicas de diseño para asegurar que no ocurran errores decorta duración en las salidas durante el breve periodo mientras esos cambios se están propagando a través de la cadena, pero la mayoría de los sistemas están diseñados para que los glitches durante el breve tiempo de transición sean ignorados. Las salidas entonces permanecen igual indefinidamente (por ejemplo, los LEDs permanecen brillantes, la batería permanece conectada a los motores, etc.), hasta que la máquina de Moore cambia de estado otra vez.

Una máquina de Moore puede ser definida como una 6-tupla $\{ S, S_0, \Sigma, \Lambda, T, G \}$ consistente de un conjunto finito de estados (S)un estado inicio (también llamado estado inicial) S_0 el cual es un elemento de (S)

un conjunto finito llamado alfabeto entrada (Σ)un conjunto finito llamado el alfabeto salida (Λ)

una función de transición ($T : S \times \Sigma \rightarrow S$) mapeando un estado y una entrada al siguienteestado una función salida ($G : S \rightarrow \Lambda$) mapeando cada estado al alfabeto salida.

El número de estados en una máquina de Moore será mayor o igual al número de estados enla Máquina de Mealy correspondiente.

2.2.- ESPECIFICACIÓN DEL SISTEMA MEDIANTE DIAGRAMAS Y TABLAS DE ESTADOS

Diagrama de Estado: Esta muestra la secuencia de estados por los que pasa bien un casode uso, un objeto a lo largo de su vida, o bien todo el sistema. Es una forma de representación gráfica más intuitiva de los autómatas finitos basadas en dígrafos con arcos acotados llamados transiciones en los cuales se ponen los símbolos de tránsito

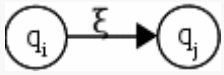
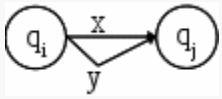
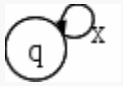
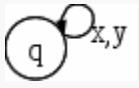
entre un vértice (estado) y otro y se identifican los estados de partida y los de aceptación del resto. Los diagramas de estados finitos son también representaciones más cómodas para su elaboración, legibilidad y comprensión de distintos tipos de abstracciones computacionales dereconocimiento como los autómatas de pila y las máquinas de Turing.

Sea un autómata finito definido por la 5-tupla $A=<Q, T, g, F, q_0>$, donde Q es el conjunto de estados, T el alfabeto de símbolos terminales, la relación de transiciones $g \subseteq Q \times (T \cup \{\xi\}) \times Q$, F son los estados finales o de llegada dentro de Q, q_0 es el estado inicial o de partida; se le llama **diagrama de estados de *A*** al grafo orientado con aristas y vértices acotados de la forma siguiente:

- ☐ Todos los estados de Q se representan por círculos en cuyo interior se designa elnombre del estado que representa.
- ☐ El estado inicial q_0 se indica agregando una saeta corta a su izquierda que no tiene origen alguno y concluye en el borde de la circunferencia, normalmente en los 180grados del mismo.
- ☐ Los estados finales se indican o bien sombreando el círculo o poniendo un borde doble al estado.
- ☐ Pueden existir estados iniciales finales, simplemente se aplican los dos casos anteriores.
- ☐ A cada transición entre los estados q_i y q_j con el símbolo terminal x o la cadenavacía ε se representa como un arco etiquetado con x ó ε según corresponda.
- ☐ Si entre los estados y en el mismo sentido hay varias transiciones con los terminales $x_1, x_2, ..., x_n$ entonces se indican en el mismo arco pero separados por espacio o comas según convenga.

Tipo	**Diagrama de estado**	**Descripción**
Estado		Círculo con el nombre del estado etiquetado dentro
Estado inicial		$q_0 \in F$ q_0: Flecha corta sin origen que apunta al estado
Estado final		$q \in F$: Estado sombreado.

Estado final		$q \in F$: Estado con doble círcunferencia.
Estado inicialy final		: Flecha sin origen que apunta al estado sombreado.

Estado inicialy final		: Flecha sin origen que apunta al estado doblemente circulado.
Transición		$<q_i,x,q_j>$ ó $g(q_i,x)=q_j$: Arco con origen en q_i y destino en q_j y acotado con el terminal *x*.
Transición vacía		$<q_i,\varepsilon,q_j>$ ó $g(q_i,\varepsilon)=q_j$: Arco con origen en q_i y destino en q_j y acotado con la cadena vacía ε.
Transición múltiple		$g(q_i,x)=q_j$, $g(q_i,y)=q_j$: Arco con origen en q_i y destino en q_j y acotado con los terminales *x,y*, separados por coma o espacio en blanco.
Lazo		$<q,x,q>$ ó $g(q,x)=q$: Arco circular con origen y destinoen *q*, acotado con el terminal *x*.
Lazo múltiple		$g(q,x)=q$, $g(q,y)=q$: Arco circular con origen y destinoen *q*, acotado con los terminales *x,y*.

Función

En el diagrama de estados se indica qué eventos hacen que se pase de un estado a otro y cuáles son las respuestas y acciones que genera. También ilustra qué eventos pueden cambiarel estado de los objetos de la clase. En cuanto a la representación, un diagrama de estados esun grafo cuyos nodos son estados y cuyos arcos dirigidos son transiciones etiquetadas conlos nombres de los eventos. Normalmente contienen: estados y transiciones. Como los estados y las transiciones incluyen, a su vez, eventos, acciones y actividades. Al igual que otros diagramas, en los diagramas de estado pueden aparecer notas explicativas y restricciones.

Definición de Estado

Identifica un periodo de tiempo del objeto (no instantáneo) en el cual el objeto está esperando alguna operación, tiene cierto estado característico o puede recibir cierto tipo de estímulos. Se representa mediante un rectángulo con los bordes redondeados, que puede tener tres compartimientos: uno para el nombre, otro para el valor característico de los atributos del objeto en ese estado y otro para las acciones que se realizan al entrar, salir oestar en un estado. También en casos mas simples se usan círculos con textos dentro para la representación de los estados, como para los autómatas finitos.

Partes que conforman el Diagrama de Estados Estado

Un estado se representa como una caja redondeada con el nombre del estado en su interior. Una transición se representa como una flecha desde el estado origen al estado destino. La caja de un estado puede tener 1 o 2 compartimentos. En el primer compartimento apareceel nombre del estado. El segundo compartimento es opcional, y en él pueden aparecer acciones de entrada, de salida y acciones internas.

Eventos

- ☐ Es una ocurrencia que puede causar la transición de un estado a otro de un objeto.Esta ocurrencia puede ser una de varias cosas:
- ☐ Condición que toma el valor de verdadero o falso
- ☐ Recepción de una señal de otro objeto en el modelo

Recepción de un mensaje

Paso de cierto período de tiempo, después de entrar al estado o de cierta hora y fecha particular. El nombre de un evento tiene alcance dentro del paquete en el cual está definido,no es local a la clase que lo nombre.

Envío de mensajes

Además de mostrar y transición de estados por medio de eventos, puede representarse elmomento en el cual se envían mensajes a otros objetos. Esto se realiza mediante una línea punteada dirigida al diagrama de estados del objeto receptor del mensaje.

Transición simple

Una transición simple es una relación entre dos estados que indica que un objeto en el primer estado puede entrar al segundo estado y ejecutar ciertas operaciones, cuando unevento ocurre y si ciertas condiciones son satisfechas. Se representa como una línea sólida entre dos estados, que puede venir acompañada de un texto con el siguiente formato:

Transición interna

Es una transición que permanece en el mismo estado, en vez de involucrar dos estados distintos. Representa un evento que no causa cambio de estado. Se denota como una cadena adicional en el compartimiento de acciones del estado.

Acciones

Se puede especificar la solicitud de un servicio a otro objeto como consecuencia dela transición. Se puede especificar el ejecutar una acción como consecuencia de entrar, salir,estar en un estado, o por la ocurrencia de un evento.

Generalización de Estados

Se puede reducir la complejidad de estos diagramas usando la generalización de estados. Se distingue así entre superestado y subestados. Un estado puede contener varios subestados disjuntos. Los subestados heredan las variables de estado y las transiciones externas. La agregación de estados es la composición de un estado a partir de varios estadosindependientes. La composición es concurrente por lo que el objeto estará

en alguno de los estados de cada uno de los subestados concurrentes. La destrucción de un objeto es efectiva cuando el flujo de control del autómata alcanza un estado final no anidado. La llegada a un estado final anidado implica la subida al superestado asociado, no el fin del objeto.

Subestados

Un estado puede descomponerse en subestados, con transiciones entre ellos y conexiones al nivel superior. Las conexiones se ven al nivel inferior como estados de inicio o fin, los cuales se suponen conectados a las entradas y salidas del nivel inmediatamente superior.

Transacción Compleja

Una transición compleja relaciona tres o más estados en una transición de múltiples fuentes y/o múltiples destinos. Representa la subdivisión en threads del control del objeto o una sincronización. Se representa como una línea vertical de la cual salen o entran varias líneas detransición de estado.

Transición a estados anidados

Una transición de hacia un estado complejo (descrito mediante estados anidados) significa la entrada al estado inicial del subdiagrama. Las transiciones que salen del estado complejo se entienden como transiciones desde cada uno de los subestados hacia afuera (a cualquier nivelde profundidad).

Transiciones temporizadas

Las esperas son actividades que tienen asociada cierta duración. La actividad de espera se interrumpe cuando el evento esperado tiene lugar. Este evento desencadena una transición que permite salir del estado que alberga la actividad de espera. El flujo de control se transmite entonces a otro estado.

Ventajas y DesventajasVentajas

El Diagrama de Estados tiene éxito en sistemas interactivos, ya que expresa la intención quetiene el actor (su usuario) al hacer uso del sistema.

Como técnica de extracción de requerimiento permite que el analista se centre en las necesidades del usuario, qué espera éste lograr al utilizar el sistema, evitando que la gente especializada en informática dirija la funcionalidad del nuevo sistema basándose solamente encriterios tecnológicos.

A su vez, durante la extracción (elicitation en inglés), el analista se concentra en las tareas centrales del usuario describiendo por lo tanto los casos de uso que mayor valor aportan al negocio. Esto facilita luego la priorización del requerimiento.

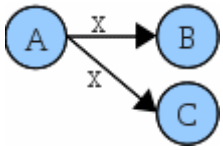
Desventajas

La inclusión de estas relaciones hace que los diagramas sean más difíciles de leer, sobre todopara los clientes.

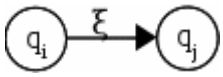
Importancia

Los diagramas de estado en el caso de los automatas finitos, además de mejorar su legibilidad,comprensión, e incluso visualizar una especie de primera aproximación material a su implementación física o computacional; también ayudan a visibilizar las propiedades del AF más intuitivamente que en la notación de la 5-tupla o la de la tabla de transiciones.

Por ejemplo, un autómata finito no determinista se reconoce con más prontitud viendo si en su esquema aparecen las transiciones de la forma:

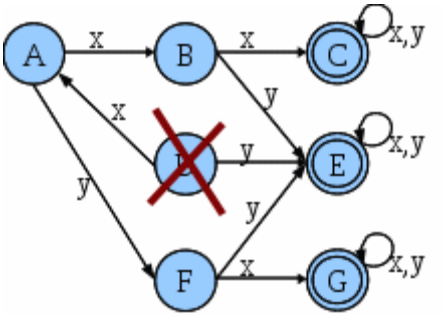


(Transición no determinista).



(Transición vacía).

O si existe un estado inaccesible desde el inicio como en el caso:



En teoría de autómatas y lógica secuencial, una **tabla de transición de estados** es una tabla que muestra qué estado se moverá un autómata finito dado, basándose en el estado actual y otras entradas. Una *tabla de estados* es esencialmente una tabla de verdad en la cual algunas de las entradas son el estado actual, y las salidas incluyen el siguiente estado, junto con otras salidas.

Una tabla de estados es una de las muchas maneras de especificar una *máquina de estados*, otras formas son un diagrama de estados, y una *ecuación característica*.

Cuando se trata de un autómata finito no determinista, entonces la tabla de transición muestra todos los estados que se moverá el autómata.

Tablas de estados de una dimensión

También llamadas **tablas características**, las tablas de estados de una dimensión son más como tablas de verdad que como las versiones de dos dimensiones. Las entradas son normalmente colocadas a la izquierda, y separadas de las salidas, las cuales están a la derecha.Las salidas representarán el siguiente estado de la máquina. Aquí hay un ejemplo sencillo de una máquina de estados con dos estados, y dos entradas combinacionales:

A	B	Estado Actual	Siguiente Estado	Salida
0	0	S_1	S_2	1
0	0	S_2	S_1	0
0	1	S_1	S_2	0

0	1	S_2	S_2	1
1	0	S_1	S_1	1
1	0	S_2	S_1	1
1	1	S_1	S_1	1
1	1	S_2	S_2	0

S_1 y S_2 representarían probablemente los bits individuales 0 y 1, dado que un simple bit solotiene dos estados.

- ☐ Tablas de Estados de dos dimensiones
- ☐ Las tablas de transición de estados son normalmente tablas de dos dimensiones. Haydos formas comunes para construirlas.

La dimensión vertical indica los Estados Actuales, la dimensión horizontal indica eventos, y las celdas (intersecciones fila/columna) de la tabla contienen el siguiente estado si ocurre un evento (y posiblemente la acción enlazada a esta transición de estados).

Tabla de Transición de Estados

Events / State	E_1	E_2	...	E_n
S_1	-	A_y/S_j	...	-
S_2	-	-	...	A_x/S_i

…	…	…	…	…
S_m	A_z/S_k	-	…	-

(S: estado, E: evento, A: acción, -: transición ilegal)

La dimensión vertical indica los Estados Actuales, la dimensión horizontal indica los siguientesestados, y las intersecciones fila/columna contienen el evento el cual dirigirá al siguiente estado particular.

Tabla de Transición de Estados

next / current	S_1	S_2	…	S_m
S_1	A_y/E_j	-	…	-
S_2	-	-	…	A_x/E_i
…	…	…	…	…
S_m	-	A_z/E_k	…	-

(S: estado, E: evento, A: acción, -: transición imposible)

2.3.- REPRESENTACIÓN COMPORTAMENTAL DEL SISTEMA MEDIANTE VHDL

El lenguaje de descripción en hardware VHDL se estructura en módulos o unidades funcionales, identificados mediante una palabra reservada y particular de este lenguaje (véasefigura 1.2). En tanto, a su vez, cada módulo tiene una secuencia de instrucciones o sentencias,las cuales, en conjunto con las declaraciones de las unidades involucradas en el programa,

permiten la descripción, la comprensión, la evaluación y la solución de un sistema digital. Al interior de la estructura de un programa, las unidades Entidad (Entity) y Arquitectura (Architecture) —en conjunto— forman la columna vertebral de este lenguaje. Por su parte,los módulos restantes, no necesariamente utilizados en la búsqueda de una solución, sirven entre otras cosas para optimizar y generalizar la aplicación en futuros desarrollos, como se verá cuando la ocasión se presente. Sin embargo, en este momento nuestra atención se centra en describir la función de la entidad y la arquitectura.

En una descripción funcional lo más importante es el conocimiento global del sistema, razón por la cual las entidades diseñadas bajo este estilo son programadas como una caja negra; es decir, no importa la organización o la estructura interna de la entidad, solo se requiere que elprogramador conozca lo que espera obtener en la salida y la forma en que operan las pins de entrada. Por ejemplo, considérese una entidad en la cual existe una salida C que adopta el valor C=1 cuando las señales de entrada, A y B, son iguales; en caso contrario, la salida Ctoma el valor de cero C=0. Entonces, la síntesis del problema sería la siguiente:

```
Si A=B entonces C=1, sino C=0
if A=B   then   C=1, else C=0
```

A continuación se muestra el código que representa la descripción del circuito.

```
1  -- Ejemplo de una descripción funcional
2  library ieee;
3  use ieee.std_logic_1164.all;
4  entity funcional is
5  port (A,B: in std_logic;
6      c: out std_logic);
7  end funcional ;
8  architecture caja of funcional is
9  begin
10 process (A,B)
11 begin
12   if A = B then        -- Si A=B entonces
13      C <='1';
14   else                 -- sino
15      C <='0';
16 end if;
17 end process;
18 end caja;
```

Listado 1.13 Arquitectura funcional de un comparador de igualdad de 2 bits.

Nótese que la declaración de la entidad (entity) se encuentra descrita entre las líneas 4 y 7; en tanto que de las líneas 8 a la 18 se desarrolla el algoritmo (architecture) que describe el funcionamiento de la entidad. Para iniciar la declaración de la arquitectura (línea 8), es necesario definir un nombre arbitrario con el que esta declaración de la arquitectura pueda ser referenciada. En este caso, el nombre asignado es caja, además de incluir a la entidad con la cual está relacionada (funcional). Por su parte, en la línea 9 se observa el inicio (begin) de la sección donde se comienzan a declarar los procesos que rigen el comportamiento del sistema. Una declaración funcional utiliza una nueva sentencia denominada process, la cual seaprecia en la línea 10. La declaración del proceso (process) está acompañada de una lista sensitiva entre paréntesis (A,B), que hace referencia a las señales que determinan el funcionamiento del proceso —el cual debe entenderse como el hecho de que el algoritmo es sensible al cambio de valor de estas variables—.

En la línea 11 inicia le ejecución del proceso mediante la palabra reservada begin. Siguiendo con el análisis, se puede advertir que de la línea 12 a la 16 el proceso se ejecuta mediante la declaración del tipo if-then-else (si- entonces-sino).

Esto se interpreta como sigue (línea 12): si el valor de la señal A es igual alvalor de la señal B, entonces el valor de '1' se asigna a la variable C, sino se asigna a C con la palabra reservada end process y la arquitectura se cierra de la forma acostumbrada end caja. Como se puede observar, la descripción funcional se basa de manera principal en eluso de procesos y de declaraciones secuenciales, las cuales permiten de forma rápida el modelado de la función. Con la finalidad de reforzar este tipo de programación, considérese la figura 1.30, en la cual se ha detallado la compuerta OR, cuya función de transferencia es bien conocida, además de que puede distinguirse en la tabla de verdad que se muestra en la figura 1.29. De esta manera, una descripción funcional sería de la forma siguiente:

a	b	f1
0	0	0
0	1	1
1	0	1
1	1	1

Figura 1.29

Figura 1.30 Función conocida de una compuerta OR.

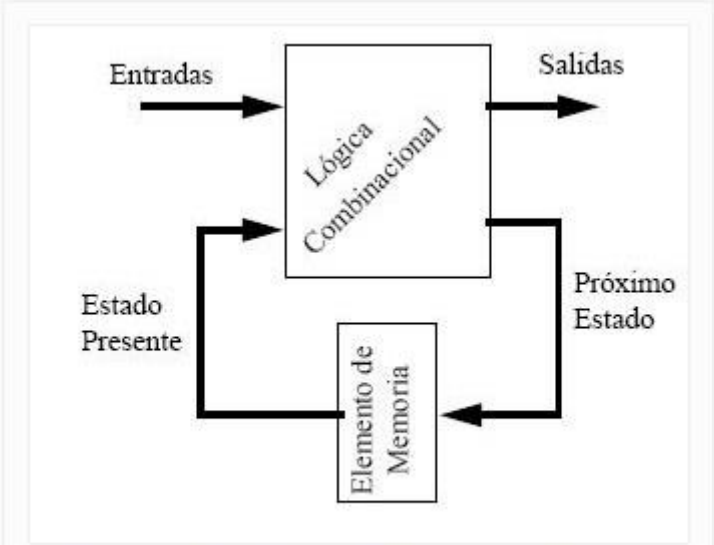
```
1  -- Declaración funcional
2  library ieee;
3  use ieee.std_logic_1164.all;
4  entity com_or is
5  port (a,b: in std_logic;
6      f1: out std_logic);
7  end com_or;
8  architecture funcional of com_or is
9  begin
10 process (a,b) begin
11    if (a = '0' and b = '0') then
12        f1 <= '0';
13    else
14       f1 <= '1';
15 end if;
16 end process;
17 end funcional;
```

Listado 1.14

2.4.- IMPLEMENTACIÓN ESTRUCTURADA DE SISTEMAS SECUENCIALES SÍNCRONOS.

El circuito secuencial debe ser capaz de mantener su estado durante algún tiempo, para ello se hace necesario el uso de dispositivos de memoria. Los dispositivos de memoria utilizados en circuitos secuenciales pueden ser tan sencillos como un simple retardador (inclusive, se uede usar el retardo natural asociado a las compuertas lógicas) o tan complejos como uncircuito completo de memoria denominado multivibrador biestable o Flip Flop.

La salida del elemento de retraso es una copia de la señal de entrada retraso un determinado tiempo; mientras que la salida del elemento de memoria copia los valores de la entrada cuando la señal de control tiene una transición de subida, por lo que la copia no es exacta, sino que sólo copia lo que interesa. Por lo tanto, el modelo clásico de un sistema secuencial consta de un bloque combinacional, que generará la función lógica que queramos realizar, y un grupo de elementos de memoria con una serie de señales realimentadas.



Modelo clásico de un sistema secuencial

2.4.1.- CODIFICACIÓN DE ESTADOS: RANDOM, ONE- HOT Y SALIDAS IGUAL A VARIABLES DE ESTADO

- One Hot: codificación de estados donde hay solamente un uno por código y seutilizan tantos flip-flops como estados existan.
- Mínimo cambio de Bit (código Grey): codificación de estados donde solamente hay un cambio de bit entre códigos adyacentes.
- Conteo enumerado (binario): codificación de estados que utiliza la numeración binariacomo asignación de códigos. Codificación de estados realizada directamente por el diseñador:
- Aleatorio (Random): codificación de estados sin orden aparente.
- Two Hot: codificación de estados donde hay dos unos por código. También utilizatantos Flip-Flops como estados existan.
- Igual a las salidas: codificación de estados donde el código del estado es igual a lasalida de la máquina de estados.

2.4.2.- CÁLCULO DEL CIRCUITO COMBINACIONAL DEEXCITACIÓN Y SALIDA: TABLA DE EXCITACIÓN YSALIDA.

Flip-flop SR

Q	Q (sig)	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

Flip-flop JK

Q	Q (sig)	J	K

0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Flip-flop D

Q	Q (sig)	D
0	0	0
0	1	1
1	0	0
1	1	1

Flip-flop T

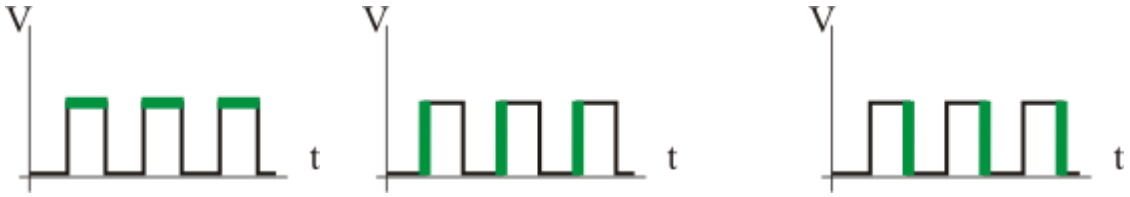
Q	Q (sig)	T
0	0	0
0	1	1
1	0	1
1	1	0

2.4.3.- IMPLEMENTACIÓN DE SISTEMAS SECUENCIALES SÍNCRONOS MEDIANTE BIESTABLES Y PUERTAS LÓGICAS.

Las puertas lógicas y los circuitos creaos con ellas constituyen la lógica **combinacional**, llamada así porque la salida únicamente depende de la combinación de las variables de entrada que haya.

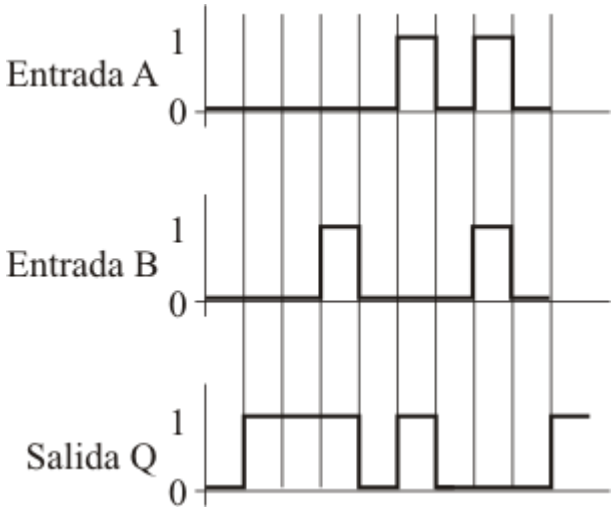
Existe un segundo grupo de circuitos lógicos denominados **secuenciales** llamados así porque la salida depende del valor de salida anterior, además de las variables de entrada. Éstosignifica que estos circuitos están dotados de memoria. Además, una gran parte de los circuitos secuenciales sólo se activan con una señal cíclica o de reloj, y se denominan circuitos secuenciales **síncronos**.

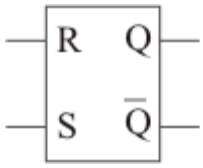
Además, el tiempo de activación puede durar mientras dure la señal de reloj o el instante decambio de 0 a 1.



Activación por pulso *Activación por flanco (ascendente o descendente)*

Para el estudio de los circuitos secuenciales se hace uso del **cronograma**, la representacióngráfica de cómo evolucionan en el tiempo tanto las entradas como la salida del circuito:





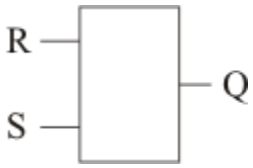
Los circuitos secuenciales básicos se denominan biestables o flip-flops, y entre ellos tenemos:

BIESTABLE R-S ASÍNCRONO

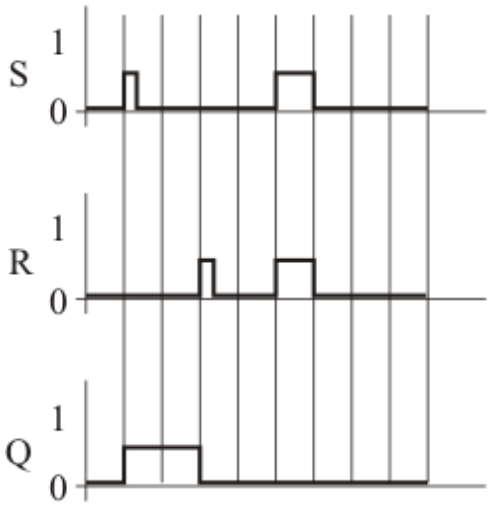
Consiste en un elemento con dos entradas, una denominada R (del inglés *RESET*) que cuando tiene voltaje pone a cero la salida, y otra entrada S (de *SET*) que pone la salida a uno cuandotiene voltaje.

Cuando las dos entradas tienen valor 0, no se producen cambios en la salida, pero si ambas tienen valor 1, el biestable no tiene claro qué debe hacer, y la salida se queda sin cambios.

Símbolo



Cronograma

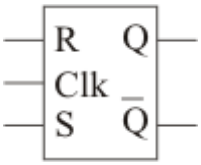


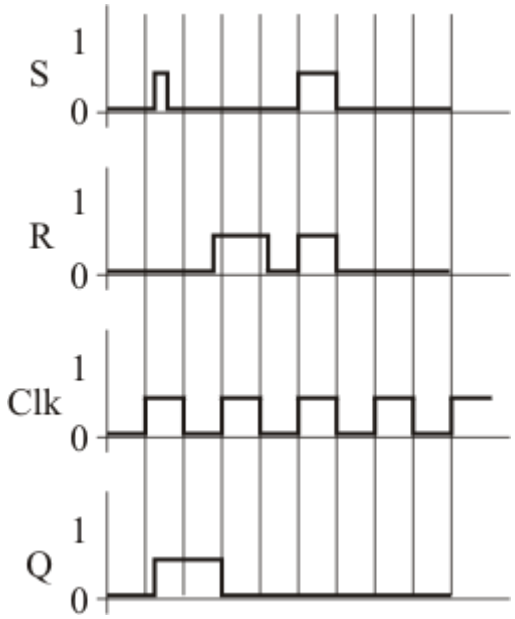
Los biestables se les suele poner dos salidas, tanto la Q, como su negada:

BIESTABLE R-S SÍNCRONO

Además de las dos entradas, R y S, en este caso hay otra entrada marcada como C o Clk (del inglés *CLOCK*). Funciona exactamente igual que su equivalente asíncrono, pero 55olo funciona durante el pulso de reloj:

Símbolo Cronograma



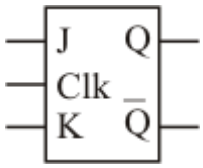


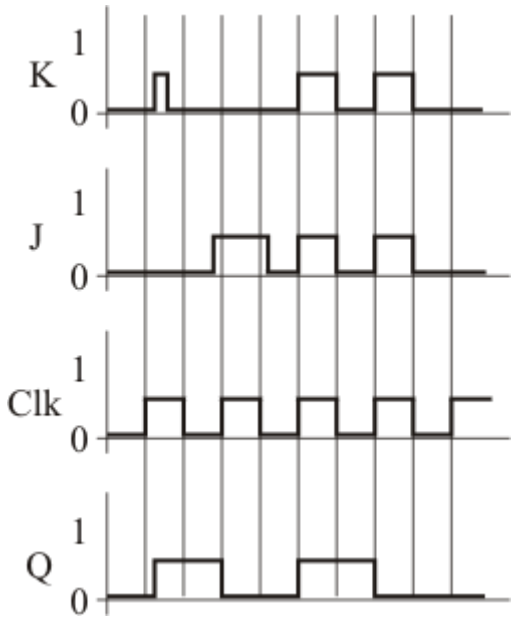
BIESTABLE J-K SÍNCRONO

Estos elementos se desarrollan para obviar el problema de que ambas entradas R y S tengan una salida indeterminada cuando ambas valen 1. En el biestable J-K, la entrada J tiene la función de poner a cero (*RESET*) y la entrada K pone a uno la salida (*SET*). Pero cuando ambas tienen valor 1, la salida invierte el valor que tuviera.

Aunque también existe una variedad asíncrona, este biestable suele ser síncrono, es decir, todos estos cambios se producen únicamente durante el pulso de reloj:

Símbolo Cronograma

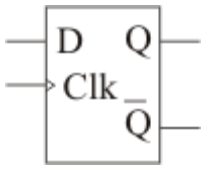




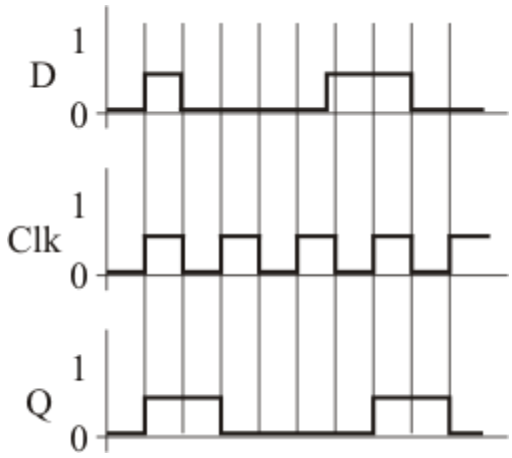
BIESTABLE D

En este último biestable, la señal de entrada D (del inglés *DATA*) es transferida a la salida conel pulso de reloj. La activación de estos biestables suele ser por flanco, por lo cual en lapatilla Clk se dibuja un pequeño triángulo:

Símbolo



Cronograma

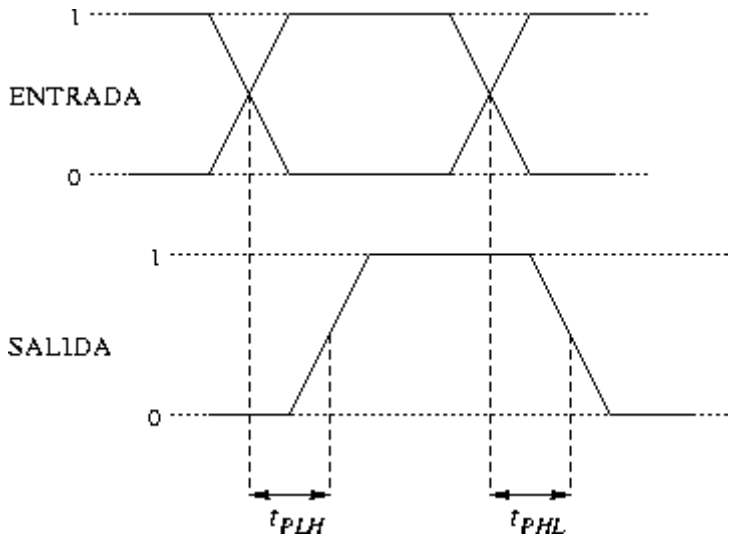


Todos los biestables se comercializan en froma de circuitos integrados, y además de las patillas propias de cada tipo, existen dos patillas para alimentación (+ y -) y dos patillas más,una PRESET que pone a uno la salida y otra RESET que pone a cero la salida.

2.4.4.- RETARDO DE PROPAGACIÓN.

Cuando una señal digital pasa a través de un circuito lógico, siempre experimenta un retardo temporal llamado *tiempo de retardo de propagación*. Este tiempo es muy importante porque limita la frecuencia máxima a la que es posible trabajar.

Figure 1.4: Retardos de propagación.



Existen dos tiempos de retardo de propagación:

1. t_{PLH} (tiempo de propagación de bajo a alto): es el tiempo entre un determinado punto delimpulso de entrada y el correspondiente impulso de salida, cuando la salida cambia de 0 a 1.

2. t_{PHL} (tiempo de propagación de alto a bajo): es el tiempo entre un determinado punto delimpulso de entrada y el correspondiente impulso de salida, cuando la salida cambia de 1 a 0.

2.4.5.- FRECUENCIA MÁXIMA.

Comúnmente se expresa que un sistema puede correr satisfactoriamente a 100MHz, o a 133MHz o cualquier otra frecuencia. ¿Qué significa esto? ¿Porqué a esa frecuencia como máxima frecuencia? ¿Quién fija ese límite en un circuito digital? ¿¿y Cómo se obtiene 6 calcula, ese número?? Muchas preguntas para contestarlas en un simple blog, pero... haré lo que pueda para contestarlas . . . :) .

Periodo Mínimo

En primer lugar, recordemos que el periodo de reloj de un sistema es el que 'marca el paso' en el sistema. Si el periodo es largo, el sistema da pasos (entrega resultados/valores) a pasosgrandes (sistema lento), si el periodo del reloj es pequeño, el sistema entregará resultados a pasos más cortos, es decir más rápidamente.

Recordemos también que la frecuencia de funcionamiento de un sistema es la inversa del periodo de ese mismo sistema: F = 1/T (periodo). Por ello para encontrarla máxima frecuencia debemos encontrar el periodo mínimo . . .

¿Cómo calcular u obtener el periodo mínimo?

Bien, en un sistema digital el periodo mínimo está compuesto por los retardos MÁXIMOS delos componentes sincrónicos, de los componentes combinacionales, y de las interconexiones entre ellos.

Veamos que significa esto: primero, en un sistema hay muchos caminos sincrónicos (camino de conexión entre flip-flops), pero el que es de interés para el cálculo del periodo mínimo es el camino (path) MAS LENTO de todos, es decir el que tiene los retardos MÁXIMOS. Elcamino más lento también es llamado camino CRITICO (critical path).

Componentes del periodo mínimo:

1. *Retardo Sincrónico*: cuanto se demora en estar estable el dato a la salida de un flip-flip luego del flanco activo del reloj.
2. *Retardo combinacional*: está compuesto por los componentes no sincrónicos, pueden ser compuertas lógicas, mux, decodificadores, LUTs (en el caso de un FPGA), etc, del camino crítico. Puede haber más de uno de estos componentes; el retardo decada uno se suma para lograr el retardo total.

La cantidad de componentes combinacionales en el camino critico es normalmente informado en el reporte defrecuencia máxima como '*niveles lógicos*' (logic levels). Si se informa que el 'logic level' es 5, significa que en el camino critico hay 5 componentes lógicos combinacionales.

3. *Retardo de ruteo o de conexiones en el camino critico*:
    1. Entre la salida del componente sincrónico (flip-flop) y la entrada a loscomponentes combinacionales.
    2. Entre los componentes combinacionales.
    3. Entre la salida de los componentes combinacionales y la entrada delflip-flop.
4. *Tiempo de establecimiento del flip-flop donde termina el camino crítico*. El dato deentrada tiene que estar un tiempo de establecimiento (dado por la hoja de datos)antes de la llegada del flanco activo del reloj, a fin de evitar problemas de metaestabilidad.

Margen de periodo: normalmente los valores de los retardos máximos de los puntos anteriores pueden estar afectados por lo que se llama PVT (process, volume, temperature) y los valores dados por la hoja de datos no son los que realmente tenga el circuito en funcionamiento. Por ello se agrega un cierto margen para asegurarse la estabilidad del sistema.

Como bien se dice,una imagen vale más que mil palabras, la siguiente figura muestra gráficamente lo anteriormente descrito en palabras:

En esta figura los tiempos detallados son los siguientes:

- *Tco*: retardo sincrónico, comúnmente llamado 'clock to output'.
- *Rr* : retardo de ruteo.
- *Rc* : retardo total de la lógica combinacional, que puede tener varios niveles.
- *Tsu*: tiempo de establecimiento del flip-flop.

Con todo esto podemos ahora describir la fórmula para calcular el periodo mínimo que nosva dar la frecuencia máxima de funcionamiento del sistema:

$$Tmin \cong Tco_{max} + RetRuteoTotal_{max} + (RetCombTotal + RetRuteo)_{max} + Tsu_{max} + 10\%Margen$$

El 10% de margen se calcula sumando los otros retardos y sacando el 10% de esa suma, elvalor obtenido se usa como margen de seguridad del periodo mínimo

Periodo Mínimo en FPGAs

Los softwares provistos por los fabricantes de FPGAs ofrecen herramientas que realizan el análisis de todos los caminos sincrónicos del sistema. Estas herramientas realizan lo que sellama un Static Timing Analysis (STA) y reportan el camino más crítico y los siguientes casos no tan críticos. Por ejemplo, Altera Quartus provee TimeQuest, Xilinx ISE ofrece Timing Analyzer, Libero de Actel ofrece SmartTime. En un anterior post detallé la información dada por Timing Analyzer.

A modo de ejemplo de la informacion proviste por una herramienta STA, se puede ver a continuación un ejemplo de la información de un camino critico dada por la herramientaTimeQuest de Altera.

Analicemos esta información: La primera columna muestra los tiempos de ya sea de unacelda lógica o de un ruteo. La tercera columna detalla el componente o conexión motivo del retardo: el tiempo referenciado como CELL es el retardo combinacional de la celda lógica detallada en le última columna de la derecha. El tiempo IC se refiere al retardo de ruteo o de interconexión entre las celdas lógicas combinacionales. Como puede verse algunos retardos de IC es igual a 0.000; esto se debe al ruteo dedicado que existe dentro de cada bloque básico del FPGA. El retardo total de este camino más crítico, que es el periodo mínimo al que puede correr satisfactoriamente este sistema, es de 2.331ns. Solo como referencia: La información de la quinta columna detalla el componente lógico usado, y la última columna el nombre dado por la herramienta de Place and Route para asociar elcomponente lógico con el nombre usado en el código VHDL.

Esta información es más fácil de interpretar viendo el esquemático de este camino. Tanto Altera, como Actel y Xilinx ofrecen esta vista. He preparado una figura que relaciona todo lo visto hasta acá, asociando el esquemático generado por la herramienta de análisis de tiempo y la figura anteriormente presentada.

2.4.6.- INICIALIZACIÓN DEL SISTEMA.

Un aspecto que un administrador de un sistema Linux debe considerar en ocasiones, es lasecuencia de inicialización del sistema. Esto puede servir para iniciar junto con el sistema servicios importantes (especialmente si el computador se emplea como servidor), para eliminar servicios que no se usen y así aprovechar mejor los recursos o para afinar algunos detalles que afectarán a todos los usuarios del sistema. Por otra parte, algunas fallas que pueden ocurrir en un sistema Linux pueden deberse a una secuencia de inicialización errada ypara solucionarlas puede ser necesario seguirla de cerca.

Secuencia de inicio

Al iniciar un computador con Linux ocurren gran cantidad de procesos, deben ocurrir en unorden preciso porque unos dependen de otros (e.g. el cargador de arranque requiere la BIOS, el kernel requiere la inicialización que hace el cargador de arranque, los módulos requieren el kernel operando, las librerías y programas requieren el kernel y los módulospara interactuar con el usuario).

A grandes rasgos la secuencia de inicialización es:

Programa que está en memoria ROM que inicializa y maneja a bajo nivel el hardware y que inicia un cargador de arranque.

Programa que permite al usuario escoger el sistema operativo al cual entrar y carga el kernel del mismo.Nombre en español de un programa que controla un tipo específico de hardware o un servicio. En inglés sería *driver*.

2.4.7.- REPRESENTACIÓN ESTRUCTURAL DEL SISTEMA MEDIANTE VHDL.

VHDL fue diseñado con base a los principios de la programación estructurada. La idea es definir la interfaz de un módulo de hardware mientras deja invisible sus detalles internos. La entidad (ENTITY) en VHDL es simplemente la declaración de las entradas y salidas de un módulo mientras que la arquitectura (ARCHITECTURE) es la descripción detallada de la estructura interna del módulo o de su comportamiento. En la siguiente figura se ilustra el concepto anterior. Muchos diseñadores conciben la Entity como una funda de la arquitectura dejando invisible los detalles de lo que hay dentro (architecture). Esto forma la base de un sistema de diseño jerárquico, la arquitectura de la entidad de más nivel (top level) puede usarotras entidades, dejando invisible los detalles de la arquitectura de la identidad de menos nivel. En la figura las entidades B, E y F no utilizan otras entidades. Mientras que la entidad Autiliza todas las demás. A la pareja entidad-arquitectura se la llama modelo. En un fichero texto VHDL la entidad y la arquitectura se escriben separadas, por ejemplo, a continuación, se muestra un programa muy simple en VHDL de una compuerta de 2 entradas. Como otros programas, VHDL ignora los espacios y saltos de líneas. Los comentarios se escriben con 2 guiones (--) y terminan al final de la línea. En la figura siguiente se muestra la estructura de unmodelo en VHDL.

SINTAXIS PARA LA DECLARACIÓN DE LA ENTIDAD VHDL

define muchos caracteres especiales llamados “palabras reservadas”. Aunque las palabras reservadas no son sensibles a las mayúsculas o minúsculas, en el ejemplo que sigue las utilizaremos en mayúsculas y negrita para identificarlas.

```
ENTITY Nombre_entidad IS
PORT ( Nombre de señal: modo tipo de señal;
            . . .
            Nombre de señal:  modo       tipo de señal ) ;
END nombre_entidad ;
```

Además de darle nombre a la entidad el propósito de la declaración es definir sus señales (o ports) de interfaz externa en su declaración de ports. Además de las palabras reservadas o claves **ENTITY**, **IS**, **PORT** and **END**, una ENTITY tiene los siguientes elementos.

- ☐ Nombre entidad; es un identificador seleccionado por el usuario para seleccionar laentidad.
- ☐ Nombre de señal; es una lista de uno o más identificadores separados por una coma yseleccionados por el usuario para identificar las señales externas de la interfaz.
- ☐ MODO es una de las 4 siguientes palabras reservadas para indicar la dirección de la señal:

Modo	Descripción
IN	En este modo las señales solo entran en la entidad
OUT	Las señales salen de la entidad
BUFFER	Este modo se utiliza para las señales que además de salir de la entidad puedenusarse como entradas realimentadas
INOUT	Este modo se utiliza para señales bidireccionales. Se emplea en salida con tres estados. Se puede asignar como sustituto de los tres modos anteriores, pero no se aconseja pues dificulta la comprensión del programa.

Cuando se omite el modo de una señal en la declaración de la entidad se sobreentiende quees de entrada.

- ☐ Tipo de señal; en VHDL, hay varios tipos de señales predefinidas por el lenguaje, talescomo:

TIPO	**Características**
BIT	En este tipo las señales solo toman los valores de "1" y "0"
Booleana	En este tipo las señales solo toman los valores de True y False
Std_logic	En este tipo las señales toman 9 valores, entre ellos tenemos: "1", "0",
	"Z" (para el 3.er estado), "-" (para los opcionales).
Integer	En este tipo las señales toman valores enteros. Los 1 y los 0 se escribensin “
Bit_Vector	En este tipo los valores de las señales son una cadena de unos y ceros. Ejemplo: “1000”
Std_Logic_Vector	En este tipo los valores de las señales son una cadena de los nuevevalores permisibles para el tipo std_logic.
Character	Contiene todos los caracteres ISO de 8 bits, donde los primeros 128 sonlos caracteres ASCII.

Ejemplo: “1-0Z” -231 + 1 231 - 1 Integer -2 147 483 647 2 147 483 647

Bit Character Severity_level Bit_vector Integer String Boolean Real time

Operadores	
Tipo	**Std_logic**
U	Uninitialized (Sin inicializar)
X	Forcing Unknown (Forzar valor desconocido)

	Forcing 0 (Forzar un cero)
1	Forcing 1 (Forzar un uno)
Z	High Impedance (Alta impedancia)
W	Weak Unknown (Valor débil desconocido)
L	Weak 0 (Cero débil)
H	Weak 1 (uno débil)
-	Don’t care (Cualquier valor)

UNIDAD III

BLOQUES FUNCIONALES SECUENCIALES

El **diagrama de bloques de funciones**, o **Function Block Diagram** (**FBD**) es unlenguaje gráfico para controladores de lógica programable, que describe la función entre variables de entrada y variables de salida, misma que puede ser descrita como un conjunto debloques. Las variables de entrada y salida están conectadas a bloques por líneas de conexión.

Las entradas y salidas de los bloques están conectadas mediante enlaces, los cuales pueden usarse para conectar dos puntos lógicos del diagrama, ya sea una variable de entrada con una entrada del bloque, una salida de un bloque con una entrada de otro bloque, o una salida de un bloque con una variable de salida.

El FBD es uno de los cinco lenguajes especificados en el estándar IEC 61131-3.

3.1.- CONTADORES

El PLC Simatic S7-200 ofrece al usuario un conjunto de instrucciones que permiten llevar la cuenta de cuántos enventos se producen durante la ejecución del programa de control, esto es, cuántas veces una señal (una marca, una entrada, una variable, etc.) cambia de valor.

Para poder almacenar cuántos eventos se han producido hasta un determinado momento, el PLC ofrece al usuario una zona de la memoria predefinida. Esta zona de la memoria se identifica con la letra "C" seguida de un número "XXX" que debe estar comprendido en elintervalo [0..255]. Se puede decir por tanto, que el usuario dispone de hasta 256 contadores distintos.

El valor actual (VA) del número de eventos producidos se almacena en una variable del tipo CXXX cuyo tamño es una palabra (una WORD) y cuyo tipo es INT. Además, por cada contador CXXX, el PLC ofrece al usuario una variable de tipo bit también identia como CXXX que tomará el valor "1" ó "0" dependiendo de ciertas condiciones detalladas más adelante.

El S7200 ofrece tres tipos de contadores como se muestra a continuación:

- Contador de contaje adelante (CTU). Para cuentas ascendentes.
- Contador de contaje atrás (CTD). Para cuentas descendentes.
- Contador de contaje adelnate/atrás (CTUD). Para cuentas ascendentes y descendentes

La siguiente tabla muestra de manera resumida las acciones que las distintas operaciones de contaje llevan a cabo sobre el valor actual de la cuenta y el valor del bit del contador, enfunción del valor del los principales parámetros de entrada de dicha instrucción. (Yo quitaía lacolumna Alimentación/primer ciclo porque no entiendo lo que aporta, tú qué harías?).

Tipo de contador	Valor actual (VA)	Flanco en entrada	Entrada R/LD
CTU	Si el VA>=PV el bit del contador se pone a ON. El contador seguirá contando hastaalcanzar los 32.767.	Un flanco en CU provoca el incremento del VA del contador.	Si el bit R delcontador se activa, bit del contador OFF y VA=0.
CTD	Si el VA<=0 el bit del contador se pone a ON. El contador seguirá contando hasta llegar a -32.767.	Un flanco en CD provoca el decremento del VA del contador.	Si el bit Ld del contador de activa, bit del contador OFF, y se carga el valor PV al VA.
CTUD	Si el VA>=PV el bit del contador se pone a ON. El contador seguirá contando hasta32.767 ó -32.767.	Un flacon en CU provoca el incremento del VA del contador, mientras que unflanco en CD provoca eldecremento del VA del contador.	Si el bit R delcontador se activa, bit del contador OFF y VA=0.

Debido a los tiempos de procesamiento y de ciclo de scan, estas instrucciones de contajesólo son adecuadas para contar eventos con frecuencias limitadas. Para contar eventos de altas frecuencias (hasta 20KHz) se dispone de las instrucciones de contadores rápidos (HC).

3.2.- REGISTROS DE DESPLAZAMIENTO.

Los registros de desplazamiento son circuitos secuenciales formados por biestables o flip-flops generalmente de tipo D conectados en serie y una circuitería adicional que controlará lamanera de cargar y acceder a los datos que se almacenan. En los de desplazamiento se transfiere información de un flip-flop hacia el adyacente, dentro del mismo registro o a la entrada o salida del mismo. La capacidad de almacenamiento de un registro es el númerototal de bits que puede contener.

El funcionamiento se realiza de manera síncrona con la señal de reloj. Gran parte de los registros de desplazamiento reales incluyen una señal RESET o CLEAR asíncrona, que permite poner simultáneamente todas las salidas en "0" o estado bajo, sin necesidad de introducir ceros seguidos. Esto permite limpiar rápidamente el registro de desplazamiento lo cual es muy importante a nivel práctico.

Sus funciones dentro del sistema digital son:

- Servir de almacenamiento temporal de un conjunto de bits sobre los que se estárealizando una labor de procesamiento.
- Desplazamiento de datos a lo largo de los flip-flops.
-

3.3.- REGISTROS CONECTADOS EN ANILLO.

Utilización del protocolo de redundancia en anillo "Media Redundancy Protocol" (MRP) para crear una comunicación PROFINET redundante con SIMATIC S7 y SCALANCE X.

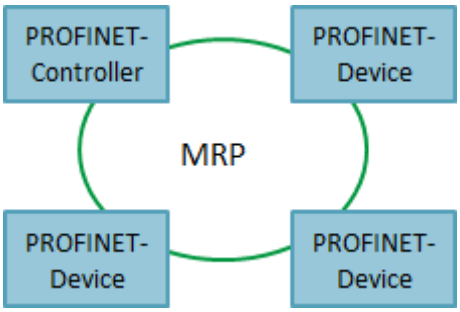
La seguridad ante fallos es un requisito básico muy importante para la automatización de procesos / industria manufacturera y el abastecimiento de energía. Para aumentar la disponibilidad se han desarrollado redes de comunicación industriales con rutas de conexión físicas redundantes entres los equipos de la red.

Solución

Una solución muy utilizada es crear una topología en anillo. Ofrece muchas ventajas:

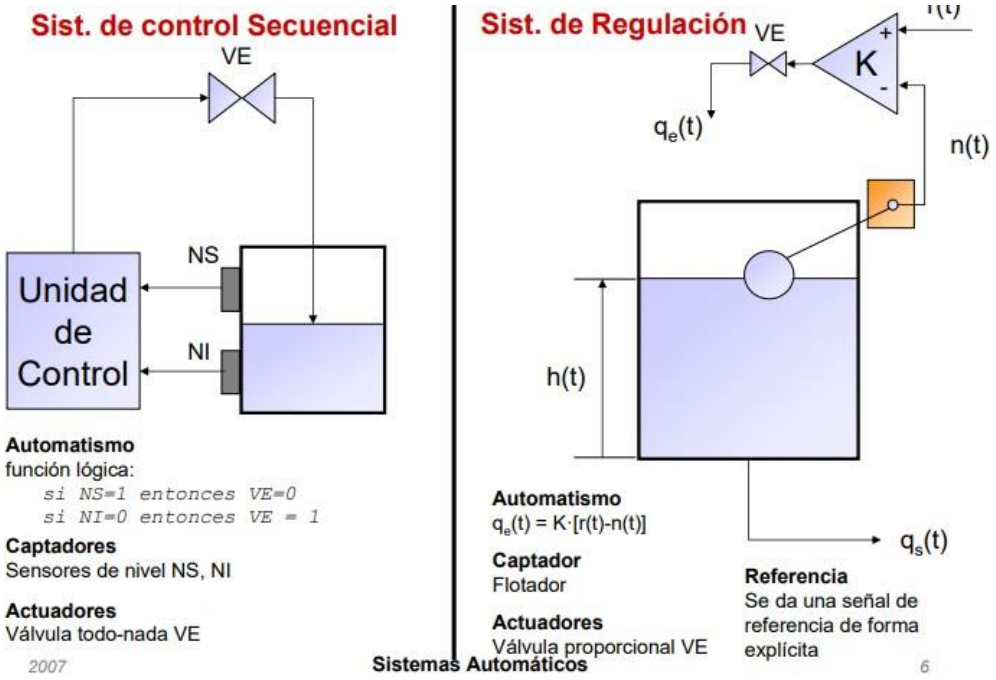
- ☐ Detección rápida de perturbaciones en la red y reconfiguración de la misma.
- ☐ Aplicable a redes pequeñas y redes muy grandes.
- ☐ Económico por la reducción del coste del cableado.
- ☐ Cableado estructurado claro y sencillo.
- ☐ La instalación es ampliable en funcionamiento.
- ☐ El protocolo estandarizado permite la compatibilidad entre equipos de diferentes fabricantes.

Esta aplicación ejemplo muestra el funcionamiento del protocolo "Media Redundancy Protocol" (MRP) y cómo poder implementar una comunicación PROFINET redundante utilizando componentes SIMATIC y SCALANCE.



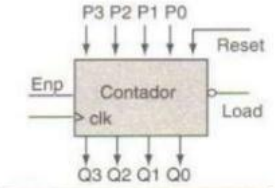
MRP sirve para aumentar la disponibilidad de la red con topología en anillo y para tener una comunicación libre de bucles.

3.4.- REPRESENTACIÓN COMPORTAMENTAL DE BLOQUES FUNCIONALES SECUENCIAL MEDIANTE VHDL.



3.5.- IMPLEMENTACIÓN DE SISTEMAS SECUENCIALES SÍNCRONOS MEDIANTE BLOQUES FUNCIONALES SECUENCIALES Y PUERTAS LÓGICAS.

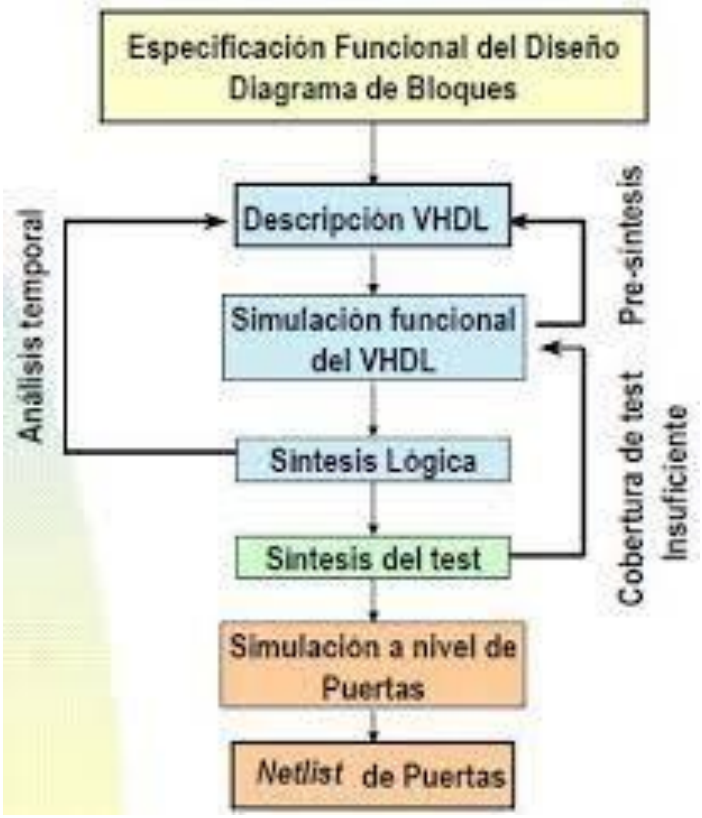
Ejercicio: Describa en VHDL un contador síncrono con reset asíncrono y carga en paralelo (load). Y simule su funcionamiento.



Enp	Load	Accion
0	0	Carga
0	1	Mantiene Estado
1	0	Carga
1	1	Cuenta

```
--EJEMPLO DE UN CONTADOR CON EVENTOS
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
--USE WORK.STD_ARITH.ALL;
USE IEEE.STD_LOGIC_unsigned.ALL;
ENTITY contador IS PORT
( p: IN STD_LOGIC_VECTOR(3 DOWNTO 0);
  clk, load, enp, reset: IN STD_LOGIC;
  q: INOUT STD_LOGIC_VECTOR(3 DOWNTO 0) );--SE TIENE QUE
--DECLARAR LA SEÑAL Q COMO INOUT DEBIDO A QUE SE UTILIZA
--COMO ENTRADA Y SALIDA
END contador;
ARCHITECTURE ejemplo of contador IS
BEGIN
        PROCESS (clk, reset, load, enp) BEGIN
            IF(reset='1') THEN
                q<="0000";
            ELSIF (clk'EVENT AND clk='1') then
                    IF (load='0' and enp='0')THEN
                        q<=p;
                    ELSIF (load='0' and enp='1')THEN
                        q<=p;
                    ELSIF(load='1' and enp='0')THEN
                        q<=q;
                      ELSIF(load='1' and enp='1')THEN
                        q<=q+1;
                      END IF;
            END IF;
        END PROCESS;
END ejemplo;
```

64


Especificación Funcional del Diseño
Diagrama de Bloques
Descripción VHDL
Simulación funcional del VHDL
Sintesis Lógica
Sintesis del test
Simulación a nivel de Puertas
Netlist de Puertas
Análisis temporal
Pre-síntesis
Cobertura de test Insuficiente

3.7.- MEMORIAS NO VOLÁTILES.

Son dispositivos electrónicos enchufables en la CPU, destinados a guardar información demanera provisi permanente. Se cuentan con dos tipos de memorias, volátiles (RAM) yno volátiles (EPROM Y EEPROM), según requieran o no de energía eléctrica para la conservación de la información.

La capacidad de memoria de estos módulos se diseña para diferentes tamaños, las más típicasson: 2, 4, 8, 16, 32, 64, 128, 256 Kb, y más, excepcionalmente

3.8.- TIPOS DE MEMORIAS NO VOLÁTILES.

Memorias EEPROM

Las memorias EEPROM (Electrically Erasable Programmable Read Only Memory) pertenecen al grupo de memorias no volátiles. Las soluciones de este tipo se utilizan con mayor frecuencia en aplicaciones que requieren la presencia de áreas ROM reprogramables, especialmente en relación con el almacenamiento de datos de configuración del sistema. Dada la interfaz, las EEPROM pueden ser en serie o en paralelo. Las memorias serial (serie24xx con interfaz I2C, serie 25xx con interfaz SPI, serie 93xx con interfaz Microwire) se producen con mayor frecuencia en carcasas DIP y SOIC. Su capacidad suele ser de decenas de kB. Gracias a la interfaz en serie y al pequeño tamaño y la baja demanda de energía, talesmemorias se utilizan a menudo para almacenar datos sobre el número de serie del dispositivo o la configuración y los datos de producción. También hay memorias en serie con una dirección única preprogramada de 48 o 64 bits. que se puede utilizar como la dirección MAC del dispositivo. Las memorias en paralelo son series 28xx.

Tenga en cuenta el hecho de que, en términos de funcionalidad de lectura y derivaciones, es compatible con la serie EPROM 27xxx. El espectro de aplicación de la memoria EEPROM incluye principalmente su presencia en la electrónica industrial: dispositivos de medición y sistemas de control, sistemas de protección y alarma, sensores y cargadores de baterías.También puedes encontrarlos en dispositivos IoT. La memoria EEPROM también se utiliza en dispositivos médicos y en el segmento automotriz. La memoria EEPROM tampoco carece de electrónica de consumo, es decir, hardware de computadora, electrónica de consumo y electrodomésticos. El soporte de Microchip en la producción de chips de memorias EEPROM fabricadas con la tecnología más antigua – 1.2µm – 0.7-0.5- 4.4-0.25 – 0.18-0 13um – juega unpapel importante para garantizar la continuidad de la producción de equipos.

La dirección deldesarrollo de las memorias EEPROM incluye principalmente la reducción de la demanda deenergía y la introducción del servicio de nuevas interfaces. Vale la pena tener en cuenta aquí el bus asíncrono UNI/O desarrollado por Microchip en 2008 (serie 11xx). Se basa en una línea de datos SCIO bidireccional (del inglés: Single Connection I/O), que proporciona untotal de 3 salidas que permiten el uso de gabinetes SOT23 y TO92. La última solución es

lamemoria con la interfaz Single-Wire (serie 21CS), en la que el sistema se alimenta a través deuna línea de datos bidireccional, lo que reduce el número de pines del sistema a dos (SI/O + GND).

Memorias FLASH

Las memorias FLASH no volátiles en relación con la memoria EEPROM se caracterizan portiempos de escritura y lectura más cortos, lo que, sin embargo, está asociado con la falta de capacidad para escribir y leer bytes individuales. Aquí, la lectura y la escritura se llevan a cabo en áreas más grandes de la memoria, las llamadas páginas (128/256 bytes). La memoria Flashofrecida por Microchip tiene una interfaz paralela (serie SST39) o una interfaz serial (SPI en laserie SST25, SQI en la serie SST26). Los parámetros importantes de la memoria Flash son:capacidad de memoria (4 Mbit), frecuencia de operación (por ejemplo, 40 MHz), voltaje de operación (p. ej., 2.3 – 3.6V), tipo de carcasa (por ejemplo, TDFN8), método de ensamblaje (por ejemplo, SMD) y temperatura de trabajo (p.ej., -40-85°C). Vale la pena mencionar la tecnología SuperFlash utilizada en los sistemas que garantiza un consumo de energía reducidocon una eliminación muy breve de los datos. A su vez, la interfaz SQI garantiza una transmisión de datos rápida utilizando el número mínimo de pines.

Memorias EERAM

EERAM es una conexión rápida de la memoria SRAM (Static Random-Access Memory) y dela memoria no volátil EEPROM, almacenando una copia de memoria SRAM (I2C, serie 47x).Una conexión de este tipo significa que, en caso de problemas de alimentación, el contenido de la memoria caché se puede restaurar desde una copia de seguridad. Por lo tanto, la memoria EERAM se basa en un condensador externo, que es una fuente de respaldo de energía durante el tiempo necesario para copiar el contenido de la memoria. Vale la pena mencionar la similitud de los sistemas NVSRAM (Non-volatile Static Random-Access Memory.— seria 23XX), que también tienen la función de mantener el contenido de RAM. La diferencia es que para que funcionen correctamente se requiere la presencia de una fuente de alimentación adicional: una batería o batería, innecesaria en el caso de la memoria EERAM,lo que tiene un impacto en los costos de producción del dispositivo. Es importante destacar que el número de operaciones para guardar y leer datos es ilimitado.

Dependiendo de las necesidades de la aplicación, se selecciona la memoria EERAM con unacapacidad de 4kb o 16kb. Durante el trabajo, la lógica interna es responsable de monitorear el estado de la alimentación en tiempo real. Como resultado, cualquier corte de energía y disminución se detectan teniendo en cuenta el umbral adoptado (Vtrip). Si se detecta alguno de estos estados, se inicia la copia del contenido de la SRAM a la EEPROM. El condensadorexterno conectado al circuito Vcap del sistema es importante aquí. Con la tensión de alimentación nuevamente por encima del nivel Vtrip, el contenido de EEPROM se copia a la SRAM. Se debe enfatizar que el contenido de SRAM se puede restaurar en cualquier momento mediante la activación del programa. En resumen, las memorias EERAM se adaptan perfectamente al uso en aplicaciones donde se requiere una actualización frecuente y rápida del contenido de las celdas

de la memoria, al tiempo que se garantiza que los datos almacenados allí se guardarán en caso de pérdida de energía. Por lo tanto, pueden encontrarse idealmente en electrónica de medición (energía, gas, líquido), electrónica industrial y de consumo (terminales de pago POS, quioscos de información, impresoras) y soluciones automotrices (registradores de datos, sensores).

3.9.- IMPLEMENTACIÓN DE CIRCUITOS COMBINACIONALES MEDIANTE MEMORIAS NO VOLÁTILES

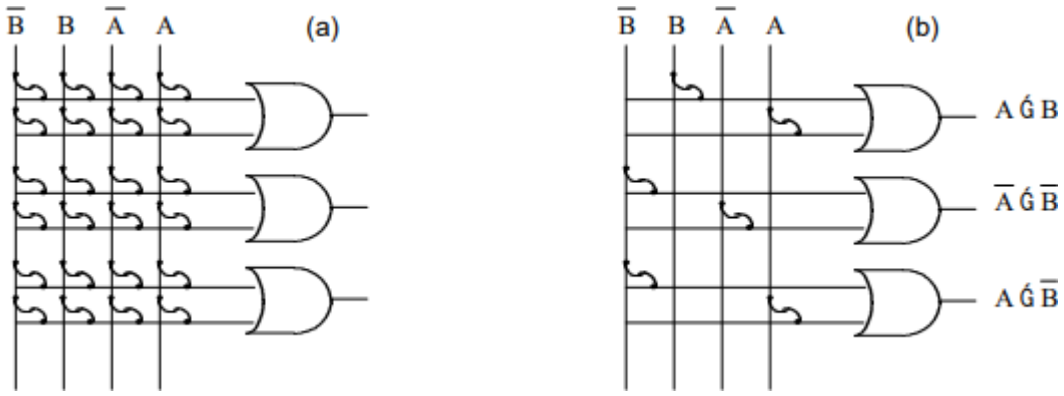
Todos los PLDs están formados por matrices programables. Una matriz programable es unared de conductores distribuidos por filas y columnas con un fusible en cada punto de intersección.

Matriz OR.

Está formada por una serie de puertas OR conectadas a una matriz programable con fusibles en cada punto de intersección de una columna y una fila, como muestra la figura 5.1.

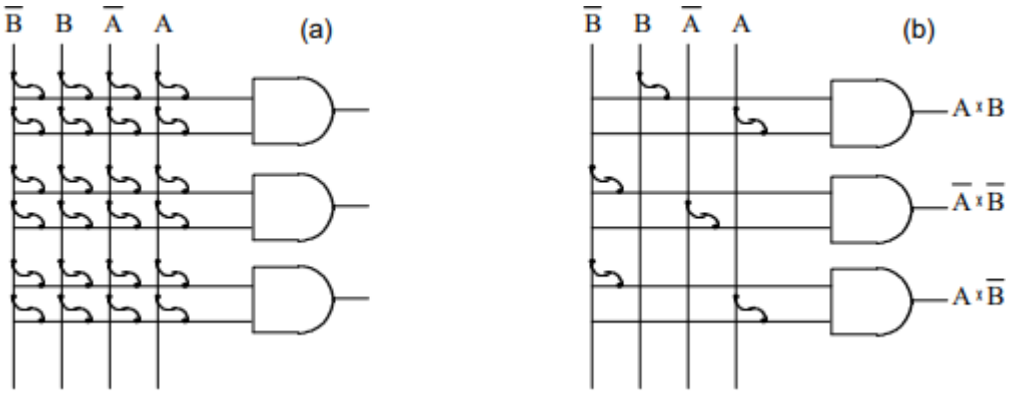
La matriz se programa fundiendo los fusibles para eliminar las variables seleccionadas de las

funciones de salida, como ilustra la parte (b). Para cada una de las entradas de la puerta OR sólo queda intacto un fusible que conecta la variable deseada a la entrada de la puerta. Una vez que el fusible está fundido no se puede volver a conectar.



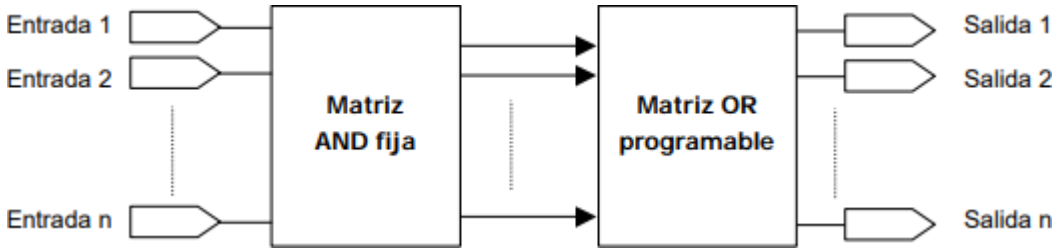
Matriz AND.

Este tipo de matriz está formado por puertas AND conectadas a una matriz programable con fusibles en cada punto de intersección como muestra la figura 5.2. Al igual que la matriz OR se programa fundiendo los fusibles para eliminar las variables de las funciones de salida, comomuestra la parte (b).

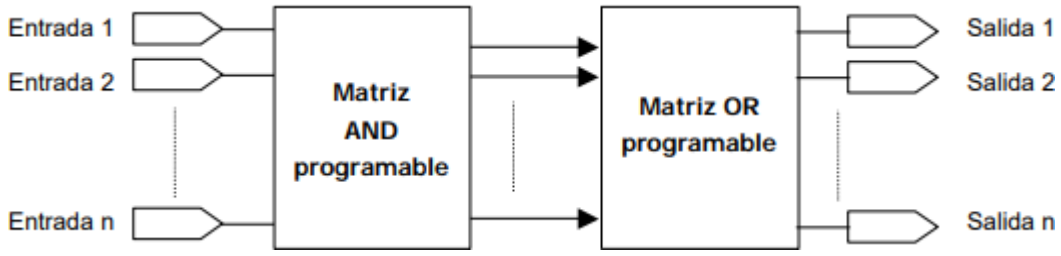


3.10.- REPRESENTACIÓN COMPORTAMENTAL DE MEMORIAS NO VOLÁTILES MEDIANTE VHDL.

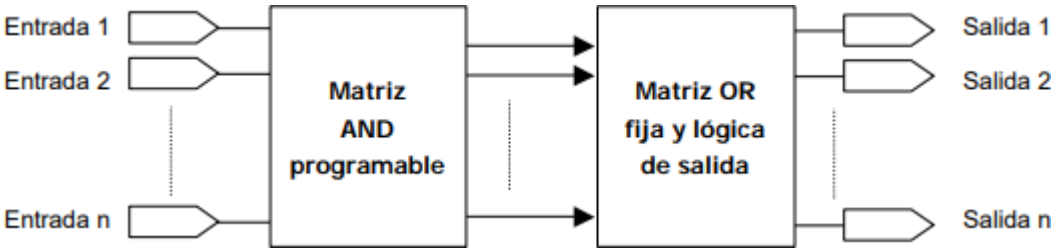
Memoria programable de sólo lectura (Programmable Read Only Memory, PROM). Está formada por un conjunto fijo de puertas AND (no programable) conectadas como decodificador y una matriz programable OR, como muestra la figura 5.3. Se utiliza como memoria direccionable y no como dispositivo lógico.



Matriz lógica programable PLA (Programmable Logic Array). Es un PLD formado por una matriz AND programable y una matriz OR programable. También se denomina FPLA (FieldProgrammable Logic Array) debido a que es el usuario y no el fabricante el que la programa.

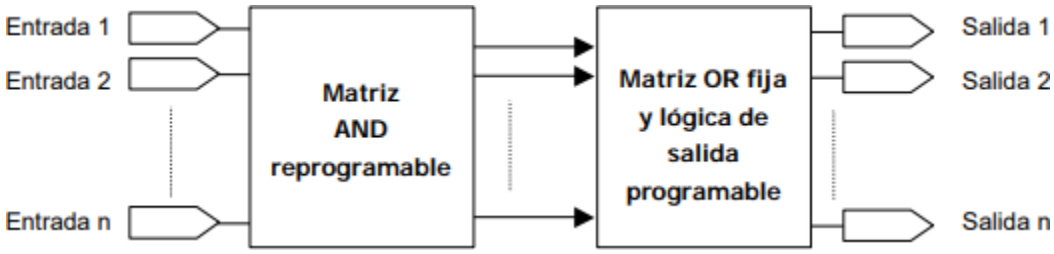


Matriz lógica programable PAL (Programmable Array Logic). Se ha desarrollado para superar ciertas desventajas de la PLA, tales como largos retardos debidos a fusibles adicionales que resulta de la utilización de dos matrices programables y la mayor complejidad del circuito. LaPAL básica está formada por una matriz AND programable y una matriz OR fija con la lógica de salida.



Matriz lógica genérica GAL (Generic Array Logic). Es el desarrollo más reciente. Al igual que la PAL se forma con una matriz AND programable y una matriz OR fija. Las dos principalesdiferencias son:

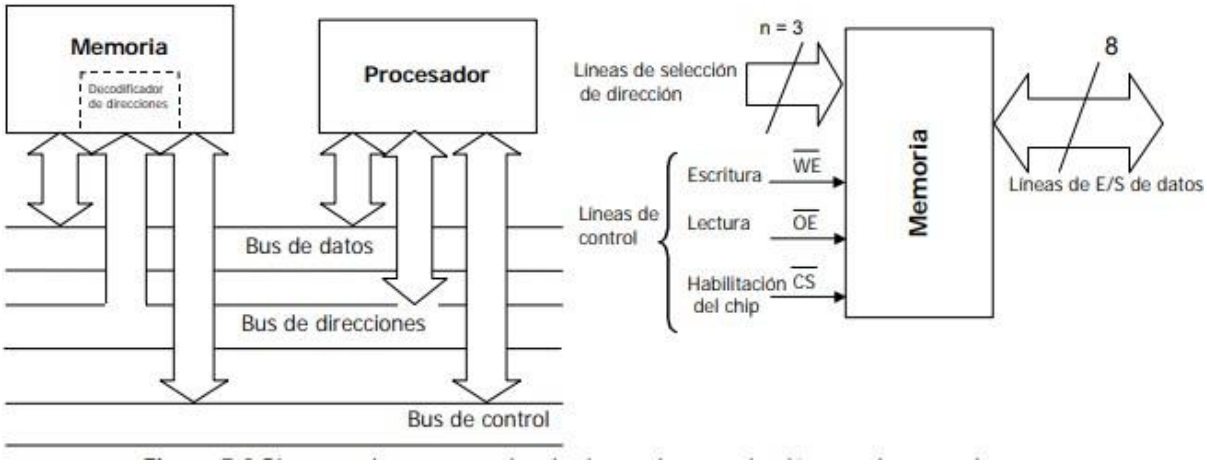
- Es reprogramable: usa la tecnología E2CMOS (Electrically Erasable CMOS) CMOSborrable eléctricamente en lugar de fusibles.
- Tiene configuraciones de salida programables.



3.11.- IMPLEMENTACIÓN DE SISTEMAS SECUENCIALESSÍNCRONOS MEDIANTE BLOQUES FUNCIONALES SECUENCIALES Y MEMORIAS NO VOLÁTILES.

Las operaciones básicas de una memoria son las de escritura y lectura. La operación de escritura coloca los datos en una posición específica de la memoria y la operación de lectura extrae los datos de una posición específica de la memoria.

Los datos se introducen y se extraen a través de un conjunto de líneas denominado bus dedatos (figura 5.9). Además, en la operación de escritura y de lectura se tiene que seleccionaruna dirección introduciendo un código binario, que representa la dirección deseada, en un conjunto de líneas denominado bus de direcciones. El código de dirección se decodifica y de esa forma se selecciona la dirección adecuada.



Las posiciones que podemos seleccionar de una memoria, es decir su capacidad, será igual a 2n , siendo n las líneas del bus de direcciones.

Operación de escritura.

Para almacenar un byte de datos en memoria, se introduce en el bus de direcciones el códigobinario de la posición de memoria donde se quiere escribir el dato. Una vez que el código de dirección está ya en el bus, el decodificador de direcciones lo decodifica y selecciona la posición de memoria especificada. La memoria recibe entonces, del bus de control una ordende escritura y los datos almacenados en el registro de datos se introducen en el bus de datos y se almacenan en la dirección de memoria seleccionada. Cuando se escribe un nuevo bytede datos en una dirección de memoria se destruye el byte que estaba en esa dirección.

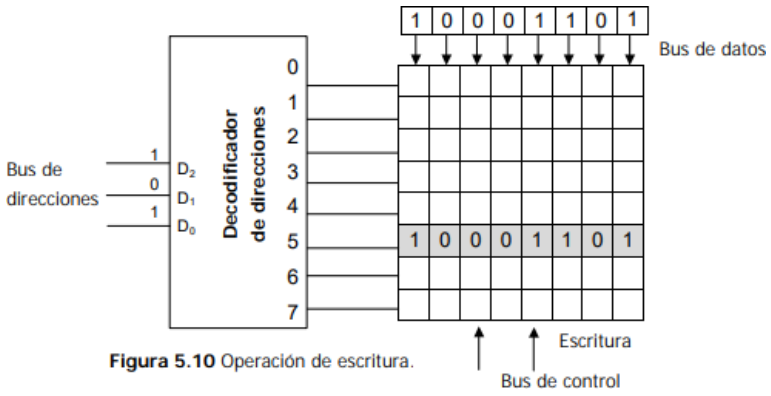


Figura 5.10 Operación de escritura.

3.12.- REPRESENTACIÓN ESTRUCTURAL DE LA IMPLEMENTACIÓN DEL SISTEMA, BASADO EN MEMORIAS NO VOLÁTILES, MEDIANTE VHDL.

El controlador hardware que hemos desarrollado ha sido diseñado para trabajar con el modelo de arquitectura objetivo descrito en la Figura 16, el cual es una extensión del modelode jerarquía de memoria comentado en la Figura 8 del capítulo anterior. Según este modelo, existen un conjunto de memorias on-chip situadas entre la memoria de configuración (off- chip) y el hardware reconfigurable. Estas memorias son conocidas habitualmente como caches, pero normalmente no son realmente caches, sino memorias de tipo SRAMcontroladas por software (es decir, scratchpads). Si se utilizan adecuadamente, pueden mejorar drásticamente las prestaciones de la jerarquía de memoria, así como su consumo de energía. El motivo es que evitan que el sistema acceda a buses externos al hardware reconfigurable, los cuales tienen una gran capacitancia [23]. Estas memorias on-chip se caracterizan por tener unos consumos de energía y tiempos de acceso mucho menores quelas memorias off-chip.

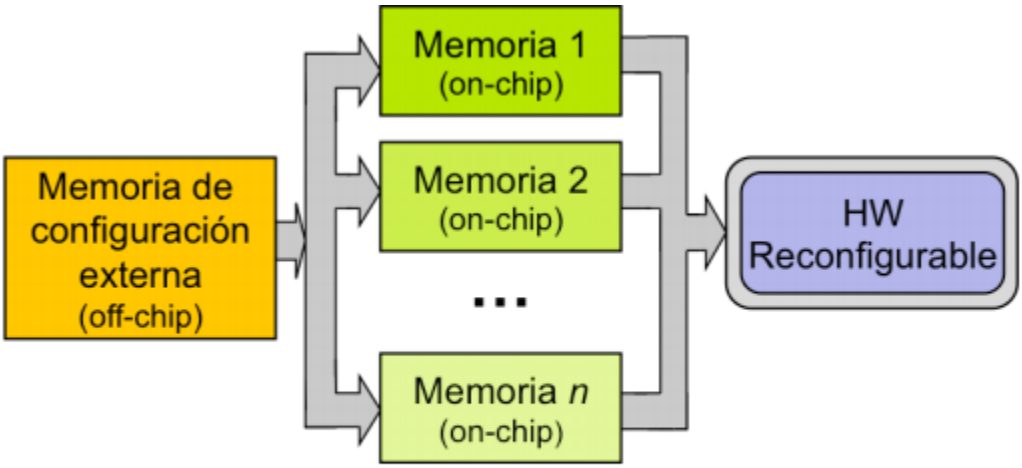


Figura 16. La jerarquía de la memoria de configuración supuesta. El HW reconfigurable también está conectado a la memoria *off-chip* por medio de una conexión dedicada, que no se muestra en la figura por simplicidad

Además, cada una de ellas puede tener diferentes propiedades. Por ejemplo, en el caso de dispositivos reconfigurables, estas memorias pueden estar construidas componiendo una o varias BRAMs. Si no hay suficientes BRAMs disponibles, otra posibilidad es utilizar la RAMque se encuentra distribuida entre todos los slices de la FPGA. En este caso, una combinación de ambos tipos de memoria constituye una jerarquía de memoria heterogénea, cuyos diferentes tipos de memorias trabajan a diferentes velocidades y tienen diferentes consumos de energía.

En esta arquitectura, se asume que los diferentes bloques en los que las configuraciones estándivididas se asignan a las diferentes memorias existentes en el sistema. Estas asignaciones (llamadas "mapeos" en el resto de la memoria) se asignan dependiendo de las restricciones de las tareas. Lo más habitual es que se asignen en tiempo de diseño; es decir, antes de laejecución de las aplicaciones. Sin embargo, también pueden actualizarse en tiempo de ejecución, en función de la carga de trabajo del sistema y de la presión que se ejerce sobre las memorias on-chip. Sea cual sea la manera en la que estos mapeos se realice, el controlador que hemos desarrollado es compatible con cualquiera de estas metodologías.

De hecho, como acabamos de comentar, hemos incorporado la posibilidad de realizar el mapeo en tiempo de ejecución, por lo que nos pareció muy interesante incorporar también un módulo de estadísticas. De esta manera, podemos consultar en tiempo de ejecución el número de aciertos y fallos que ha sufrido una determinada tarea, pudiendo recalcular el mapeo sobre dicha tarea en el caso de que el número de fallos fuese muy elevado o si seconsiderase oportuno. Esta medida ofrece la posibilidad de mejorar la eficiencia en tiempo deejecución.

Cuando el sistema arranca por primera vez, se asume que, inicialmente, todos los bloques delas reconfiguraciones están almacenados en la memoria off-chip. Sin embargo, cuando el sistema demanda la reconfiguración de una tarea en el hardware reconfigurable, el controlador que hemos desarrollado busca esta reconfiguración en la memoria off-chip, guardando una copia en la memoria on-chip correspondiente, de acuerdo a lo que el mapeo de esa tarea especifique.

Si la memoria on-chip en cuestión está llena, el controlador tambiéndebe decidir cuál de los bloques que ya están escritos en esa memoria debe ser reemplazado para dejar sitio al nuevo. Esto se explica en mayor detalle en los capítulos siguientes.

UNIDAD IV

CIRCUITOS PROGRAMABLES

4.1. - CIRCUITOS FULL CUSTOM Y SEMICUSTOM.

Los diseños electrónicos tienen que cumplir varios requisitos a nivel de prestaciones, consumo y área. Cuando trabajamos con diseños sencillos se utilizan componentes con encapsulados grandes de agujero pasante (DIP). Estos componentes se pueden coger con la mano sin problemas. Los componentes se colocan sobre una placa de prototipos o una placa de circuito impreso (PCB). Si necesitamos que el área que ocupa el circuito sea reducidapodemos emplear componentes de montaje superficial (SMD). Los componentes SMD ocupan menos espacio y mantienen las propiedades de los componentes más grandes.

A medida que aumenta la complejidad del diseño llega un momento en el que el área queocupan los componentes es mayor que el soporte que contiene el circuito. Un ejemplo que muestra esto es un teléfono móvil. Necesitamos por una parte una potencia de procesadogrande, un consumo reducido y al mismo tiempo un área pequeña. Aunque funcione perfectamente, no podemos vender un teléfono móvil si tiene un tamaño mucho mayor que los de la competencia. En estos casos es necesario integrar todas las funciones en un únicochip (SoC – System On a Chip).

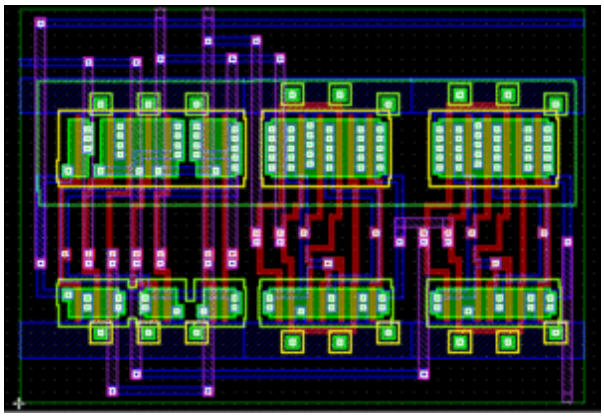
Dentro del mismo circuito integrado podemos tener varios microcontroladores / procesadores, memorias, convertidores de señal y módulos de comunicaciones. Para integrar todos los elementos podemos recurrir a dos tipos de diseño: el diseño full custom y el diseño con FPGA's. En el primero conseguimos las mejores prestaciones, así como unos consumos reducidos y un área de ocupación muy pequeña. En el segundo tipo de diseño sacrificamos algunas de estas ventajas para poder utilizar dispositivos programables.

Con lo que he comentado hasta este momento es natural pensar que, si el diseño full custom nosproporciona las mejores prestaciones, hay que usarlo siempre. Como veremos esto es cierto, pero con ciertas matizaciones.

Cuando montamos un circuito con componentes tradicionales (agujero pasante) podemoscoger los componentes sin problemas con la mano. Si estamos usando un transistor tendremos una pequeña oblea de silicio con el transistor construido físicamente, un encapsulado que lo protege y nos permite cogerlo y unos terminales de conexión que comunican el transistor con el exterior. Dependiendo de la tecnología de fabricación empleada el transistor en la oblea de silicio puede ser más o menos grande, pero se puededecir que respecto a las dimensiones del encapsulado

su tamaño es prácticamente despreciable. Hay casos en los que las dimensiones son grandes como en los trans de potencia, pero en este artículo pretendo tratar el tema de forma general.

El diseño full custom lo que hace básicamente es coger el transistor, quitarle el encapsulado ylos terminales de conexión y colocarlo directamente sobre la oblea de silicio. Al trabajar de esta forma el espacio que ocupa el transistor se reduce de forma drástica. En la siguientefigura podemos ver un circuito digital (un sumador completo) en un diseño microelectrónico.



En un diseño full custom obtenemos unos valores de prestaciones, consumos y superficieocupada óptimos. Al mismo tiempo conseguimos un coste por unidad muy reducido que nos permite ser muy competitivos. No todo son ventajas, sino que hay también algunos inconvenientes importantes.

- Es necesario contar con personal muy especializado.
- En la escala nanométrica hay efectos físicos que pueden degradar el funcionamiento del circuito si no se tienen en cuenta.
- Los tiempos de diseño, fabricación y verificación son largos (desde varios meses a un año).
- El proceso sólo es rentable si superamos un número mínimo de unidades fabricadas.
- Si aparece un error en el diseño la única alternativa es tirar la oblea de silicio y empezar de nuevo.

A partir de los beneficios e inconvenientes podemos decir que realizar un diseño full custom no es algo que una empresa pueda hacer de forma sencilla. Si queremos evitar estos inconvenientes podemos utilizar un diseño basado en FPGA's. No conseguiremos todos losbeneficios del diseño full custom pero si evitaremos muchos de sus inconvenientes. El primero de ellos es que la complejidad del diseño que se reduce de forma importante.

Elsegundo es que el diseño es reprogramable permitiendo corregir los errores que se detecten sin tener que fabricar todo de nuevo.

Las FPGA's son circuitos integrados que se venden comercialmente. El fabricante pone unconjunto de recursos genéricos como puertas lógicas o memorias y el diseñador debe conectarlos de forma adecuada para que realicen una función. Un microcontrolador incluye de serie recursos de alto nivel como memorias o módulos de comunicaciones. En la FPGApodemos usar recursos de ese tipo, pero no estamos sujetos a las limitaciones del microcontrolador. Por ejemplo, si queremos usar dos CPU en un microcontrolador tenemos que comprar dos integrados y comunicarlos entre ellos.

En la FPGA simplemente *colocamos* dos microcontroladores en el diseño y los conectamos en el mismo integrado. Lo mismo se aplica si queremos añadir más memoria o insertar bloques hardware dedicados con diseños a medida. Una operación que de forma secuencial puedecostar 10 ciclos de ejecución en un microcontrolador se puede paralelizar (ejecutar en paralelo) en una FPGA y realizarse únicamente en 1 ciclo. Esta flexibilidad obliga a que el diseñador tengo unos niveles de conocimiento superiores a los necesarios para usar un microcontrolador, pero le permite obtener prestaciones similares a las de un diseño full custom sin sus inconvenientes.

Para trabajar con una FPGA necesitamos una PCB que incluya tanto la FPGA como los componentes auxiliares que son necesarios para que funcione. Si estamos realizando un prototipo de un producto lo más sencillo es comprar un entrenador que incluye todo lo necesario para trabajar. Comercialmente podemos encontrar entrenadores que valen unos60 € para las FPGA's de gama baja. Las FPGA's de gama alta también tienen entrenadores, pero su precio aumenta de forma importante pudiendo superar los 1000 €. Una vez tenemos el entrenador, podemos programar la FPGA usando lenguajes de descripción de hardware como VHDL o Verilog o lenguajes de alto nivel como C si hemos insertado un microcontrolador como el Nios2. No estamos limitados a microcontroladores, sino que podemos añadir una CPU ARM e incluso tarjetas gráficas embebidas.

4.2.- TIPOS DE CIRCUITOS LÓGICOS PROGRAMABLES: STANDARD CELL, PLA/PAL, CPLD Y FPGA

- Las celdas estándar están diseñadas en función de la potencia, el área y el rendimiento.
- El primer paso es la arquitectura celular. La arquitectura de la celda se trata de decidir la altura de la celda en función de los requisitos de tono y biblioteca. Primero tenemos que decidir la pista, el tono, la relación β, el posible ancho de PMOS y el ancho de NMOS.
- Track: *Track* generalmente se usa como una unidad para definir la altura de la celda

estándar. *La pista* puede estar relacionada con carriles, por ejemplo, como decimos camino de 4 carriles, implica que 4 vehículos pueden correr en paralelo. Del mismo modo, labiblioteca de 9 *pistas* implica 9 *pistas de* enrutamiento disponibles para enrutar 9 cables en paralelo con un tono mínimo.

- Paso: la distancia entre dos pistas se llama paso.
- Vía: Las vías se usan para conectar dos capas metálicas diferentes como se muestra en la Fig. I (a). En la figura I (b), estamos conectando M1 y M2 usando una Vía. No hacemos pistas con un espaciado mínimo, ya que obtendremos un error de DRC si hay alguna vía saliente.

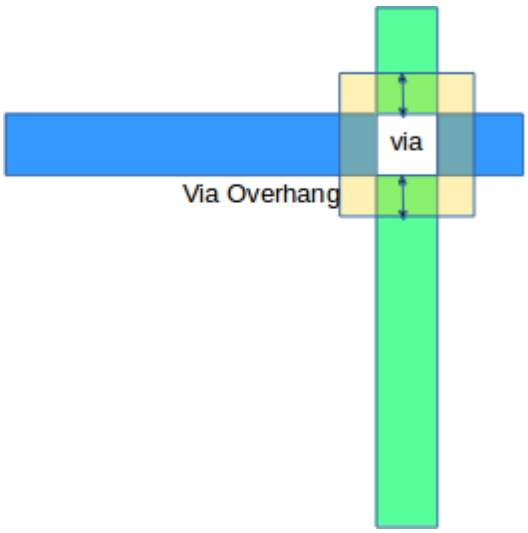


Fig. 1 (a) A través de la conexión de metal 1 y metal 2.

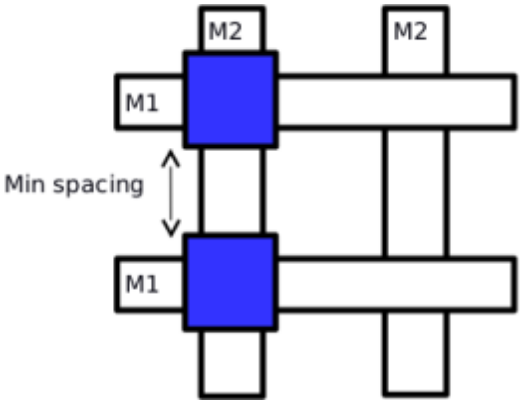


Fig. 1 (b) Cálculo de la inclinación, incluso mediante voladizo

- Veamos cómo calcular la altura de celda estándar, el tono, el tamaño de PMOS y NMOS para una biblioteca de 9 pistas.
- Deje que el ancho del metal sea de 4 unidades, el espacio mínimo de metal a metal es de 3 unidades y a través del voladizo sea de 2.
- P picor = 2 [1/2 (metal ancho) + Via voladizo] + metal-metal Spacin g . Usando esta fórmula, Pi tch = 11 unidades.
- Altura de celda estándar = Paso * (N-1) donde N representa el número de pistas. Esto suma88 unidades.
- En un diseño, las celdas se organizarán una encima de la otra, de manera que puedan compartir un VDD y VSS común. Fig. 2 representan dos células s (puede ser cualquier célula) ab ut ted de tal manera que comparten el mismo VDD.

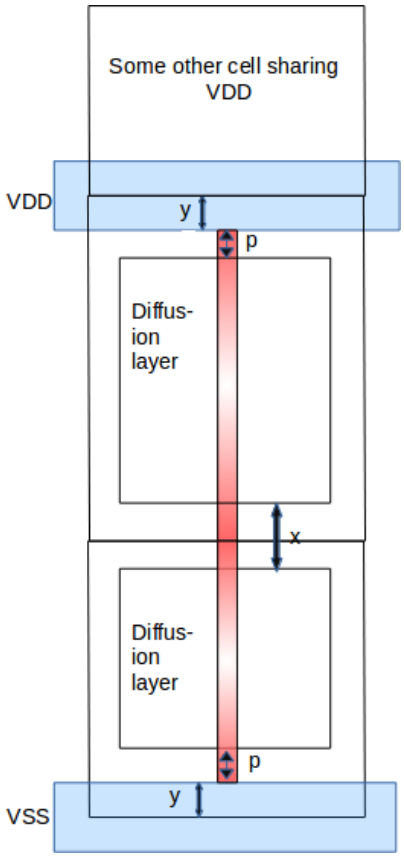


Fig. 2 Cálculo de la altura de celda estándar

Tomemos la razón β como 1.5. Por lo tanto, Wp = 1.5Wn. A continuación, se presentan las variables utilizadas para calcular la altura de celda estándar:

- p = P oly voladizo , aquí hay 2 unidades.
- x = Mínimo espacio entre pozos requerido entre las dos celdas, aquí es de 12 unidades.
- Y = W e necesidad de salir de la mitad de la del espacio entre la capa correspondiente para evitar medio violación DRC entre dos células diferentes a tope en VDD y VSS. Esto viene a1 .5 unidades.
- Wp = Ancho de PMOS.
- Wn = Ancho de NMOS.
- Altura de la celda estándar, Wp + Wn + x + 2y + 2p = 88 unidades.
- Usando esta fórmula, Wn se calcula como 27.6 unidades y Wp se calcula como 41.4 unidades. Del mismo modo, podemos calcular los valores de Wn y Wp para diferentes bibliotecas.
- Si comparamos 7T y 11T, 11T es más rápido y brindará un mejor rendimiento porque el área para 11T es mayor, de modo que podamos colocar transistores de mayor fuerza de accionamiento en él.
- Usando la biblioteca 11T podemos lograr mayores usos.
- La biblioteca 11T se utiliza para un mejor rendimiento.

- La biblioteca 7T se utiliza para mayor densidad y baja potencia.
- Celdas en biblioteca genérica

1. Puertas básicas (AND, OR, NAND, NOR, INV, EXOR, EXNOR)
2. MUX
3. HA, FA
4. Celdas especiales (Rellenos, Celdas de toma, Tapa final, De Caps)
5. Celdas de corbata
6. Células ecológicas de metal
7. AOI
8. OAI

Celdas de función booleana

9. Flops (flip flop D normal, flop capaz de escanear con set / reset)
10. Puerta del reloj
11. Células de gestión de energía

Celda de aislamiento

- U sed para aislar la salida del dominio OFF.
- Permitir que el valor de salida flotante del dominio OFF (en estado apagado) se conecte conel dominio ON dará como resultado
- Flujo de corriente de palanca, lo que resulta en un aumento del consumo de energía.
- Im correcto funcionamiento del dominio en que puede causar conocido a- estabilidad.
- También se conoce como células de sujeción, porque se utilizan para sujetar los niveles devoltaje intermedio a 0 o 1.
- Las celdas de aislamiento están diseñadas utilizando la compuerta OR (pinza 1) o lacompuerta AND (pinza 0).
- En el caso del microcontrolador, cuando el procesador pasa al modo apagado, utilizamosceldas de aislamiento para aislar el núcleo del procesador de otros módulos.
- Celdas de aislamiento pueden colocarse ya sea en OFF dominio o dominio ON.
- Cuando no son múltiple cargabilidad de salida s del dominio OFF colocando una celda de aislamiento en el dominio OFF aislará múltiples sumideros . La alimentación debe proporcionarse desde una fuente de alimentación de dominio de suministro / sumidero siempre ENCENDIDA, lo cual es un desafío.
- Me solation células si se coloca en el dominio de la función No requiere fuente de alimentación secundaria.

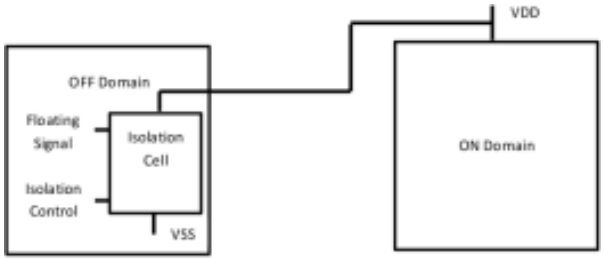


Fig2: celda de aislamiento

Desplazador de nivel

La celda de cambio de nivel se usa para cambiar el voltaje de una señal de un dominio devoltaje a otro.

- Estas celdas son necesarias cuando el chip está operando en múltiples dominios de voltaje.
- La diferencia en el rango de voltaje puede causar un funcionamiento poco confiable del dominio de destino, por lo tanto, las celdas de cambio de nivel se insertan en el cruce deldominio de voltaje.

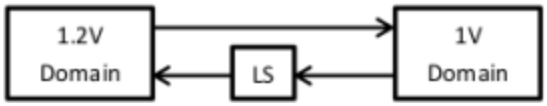


Fig3: Desplazador de nivel

Puerta de alimentación / interruptor

- Los factores que deben tenerse en cuenta al diseñar la red del interruptor de alimentaciónson:
- Cuando están ENCENDIDOS, su Vt será muy bajo, mientras que cuando estén APAGADOS,su Vt será muy alto.
- Las puertas de alimentación están diseñadas con la ayuda de CMOS de varios umbrales.
- La activación de energía es una técnica utilizada en los diseños de circuitos integrados para reducir el consumo de energía cortando la alimentación de los bloques del circuito que noestán en uso.
- Las puertas eléctricas se utilizan para la activación eléctrica.
- Corriente de acometida: La corriente de acometida es la corriente consumida por un componente durante su encendido inicial para cargar sus condensadores internos. Cuandoun dominio de energía se enciende desde el apagado, todos los condensadores en el dominio de energía comienzan a cargarse.
- La cantidad de corriente consumida será enorme, ya que todos los condensadores comienzan a cargarse, lo que dará como resultado una repentinadescarga de corriente. Esta corriente repentina puede dañar la red del interruptor de encendido. Para esto, generalmente diseñamos la red del interruptor de alimentación en forma de cadena tipo margarita.

- Corriente de fuga: el número de interruptores de alimentación utilizados para implementar lared de interru de alimentación debe ser óptimo porque si hay más interruptores decorriente, la corriente de fuga será mayor.
- Tiempo de aceleración: es el tiempo necesario para encender un componente apagado, porlo que la red del interruptor de alimentación debe diseñarse de tal manera que el tiempo de aceleración sea menor. Se puede lograr aumentando el número de interruptores de alimentación.

Flop de retención

- Los flops de retención son siempre flops ON que se utilizan para retener los datos cuando un dominio de energía pasa al modo OFF .
- La fuente de alimentación secundaria se utiliza para alimentar estos flops.
- Un flop de retención es una combinación de flop regular y pestillo de ahorro de estado.Celdas especiales

Toque las celdas

- Las células de derivación se utilizan para proporcionar una conexión de sustrato.
- Se utilizan para evitar el enganche.
- Se conectan n-well a VDD y p-sub a VSS.
- Se insertan en el diseño a intervalos regulares según las reglas de tap (distancia de tap a gate)definidas en el archivo DRC de tecnología.

Células de relleno

- Las celdas de relleno se utilizan para proporcionar continuidad ferroviaria, reduciendo así lasviolaciones de DRC creadas por la base.
- Las celdas de relleno están diseñadas de tal manera que contienen sustrato n-well y p.Celdas de metal eco-capaces
- Las celdas de relleno que se convierten para lograr cualquier funcionalidad se denominanceldas metálicas eco-capaces.
- Las capas base de las celdas de relleno y las celdas de metal eco-capaces son las mismas. Se agregarán algunas conexiones metálicas adicionales en celdas metálicas ecológicas para lograrla funcionalidad.

- Los tamaños de estas celdas son mayores en comparación con las celdas normales de la misma funcionalidad.
- Por ejemplo, considere un diseño que tiene una violación de espera después de la fabricación. Una forma de superar la violación es retrasar la ruta de datos. En este caso, podemos convertir celdas eco-capaces de metal para amortiguar el retraso. (generalmente serealiza durante el nuevo giro del chip).

Diodo de antena

- Durante la fabricación, las cargas parásitas se acumulan en capas de metal. La puerta se rompe cuando la cantidad de estas cargas es mayor al umbral. Este efecto se llamaefecto de antena. El umbral se decide por la relación entre el área de la capa de metal y el área de la puerta .
- Para superar el efecto de antena, usamos diodos de antena.
- Los diodos Zener se conectarán a las capas de metal para eliminar el exceso de cargas.

Fig4: diodo de antena

- Otra forma de superar el efecto de la antena es agregar puentes. Use capas metálicas más altas para la conexión.

Fig5: Puente

De cap cells (células de condensador de desacoplamiento)

- Las celdas De Cap son condensadores agregados en el diseño entre los rieles de potencia ytierra.
- Cuando hay una caída en el riel de alimentación, estas celdas actúan como una bateríay mantienen el voltaje a través de los rieles.
- Estas células ayudas IR gota cuestión y retire s fallos en el poder.
- En un diseño, la mayor parte del consumo de energía se realiza mediante circuitos de reloj. Suponga que todos los bloques de reloj están agrupados en un área, luego consumirán más energía, es decir, consumieron más corriente, lo que aumentará la caída de IR. En este caso se pueden usar celdas de cap .

Celda de tapa final

- Las celdas de límite final se agregan cerca del final de las filas para terminar las filas correctamente.
- Los n-pocillos de las celdas de la tapa final están debidamente terminados dentro de la celda. Celda de amarre
- Las celdas de conexión se utilizan para evitar la conexión directa de la puerta a la red eléctrica o de tierra, protegiendo así la celda del daño.
- En su diseño, algunas entradas de celda pueden requerir un valor lógico 0 o lógico 1. En lugarde conectarlos a los rieles / anillos VDD / VSS, los conecta a celdas especiales disponibles en su biblioteca llamadas celdas TIE.
- En tie high cell, nmos actúa como diodo conectado y da lógica 0 a la puerta de pmos, por lo que obtendremos la lógica 1 como salida, mientras que en tie low cell, pmos actúa comodiodo conectado y da lógica 1 a la puerta de nmos, entonces obtendremos la lógica 0 comosalida.

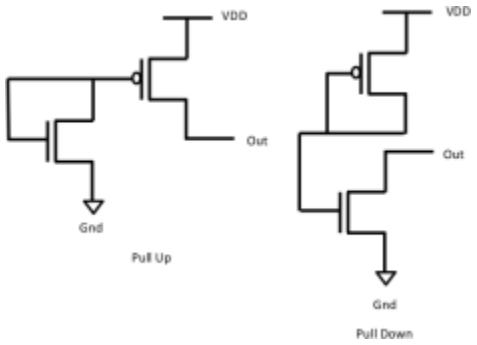


Fig6: Celda de unión

Celda de repuesto

- Las celdas de repuesto son celdas estándar normales, pero actúan como celdas redundantes, ya que se distribuyen uniformemente en el chip en previsión de futuras ECO, es decir, después de que la cinta se agote.
- Después de que se acaba la cinta, a veces es posible que tengamos que hacer algunos cambiosen el diseño para resolver un error. En estos casos, usamos las celdas de repuesto preexistentes en el diseño.
- Si llevamos a cabo los cambios de diseño con cambios mínimos de capa, se ahorrará muchocosto desde el punto de vista de la fabricación, ya que cada capa de máscara tiene un costo significativo propio.
- Las entradas de celda de repuesto se conectan a VDD / GND cuando se colocan en eldiseño y sus salidas se dejan flotando.
- Si se requiere su uso, entonces sus entradas se desconectan de VDD / GND y se conectan ala lógica funcional en modo ECO.

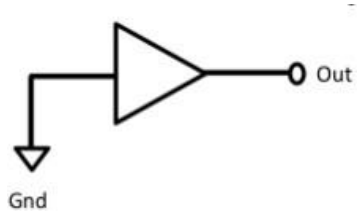


Fig7: Celda de repuesto

Caracterización

- La caracterización es la generación de archivos .lib, realizada con respecto a las esquinas PVT.
- Típicamente, la caracterización se realiza para seis cargas diferentes y seis transicionesdiferentes (rotación)
- Los modelos utilizados para generar archivos .lib son NLDM y CCS. CCS es más preciso encomparación con NLDM.

4.3.- IMPLEMENTACIÓN DE CIRCUITOSCOMBINACIONALES MEDIANTE CIRCUITOS LÓGICOS PROGRAMABLES DE TIPO PLA Y PAL.

PLAs

Un arreglo lógico programable (PLA) realiza la misma función que una ROM. Un PLA con nentradas y m salidas puede realizar m funciones de n variables. La organización interna del PLA difiere de la de la ROM, el decodificador se reemplaza por un arreglo de ANDs que realiza los términos producto seleccionados de las variables de entrada. El arreglo de ORsrealiza la operación OR a los términos producto necesarios para formar las funciones de salida.

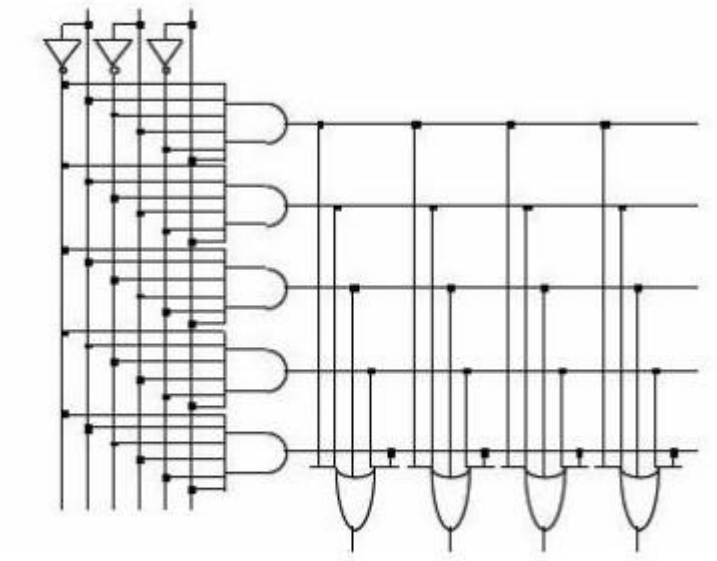


Figura 2.3 Diagrama de un PLA.

Para determinar la información que se graba en un PLA se realiza una tabla para PLA, que es diferente a una tabla de verdad para una ROM. En una tabla de verdad cada fila representa unmintérmino, por lo tanto, exactamente una fila se selecciona por cada combinación de valores de entrada, mientras que en cada fila de una tabla para PLA representa un término producto general. Por lo tanto, cero, una o más filas se seleccionan por cada combinación de valores de entrada. Para determinar el valor de la función para cierta combinación de entrada, a los valores de la función en las filas seleccionadas de la tabla para PLA se les debeaplicar la operación OR.

término producto	entradas			salidas			
	A	B	C	F0	F1	F2	F3
A'B'	0	0	_	1	0	1	0
AC'	1	_	0	1	1	0	0
B	_	1	_	0	1	0	1
BC'	_	1	0	0	0	1	0
AC	1	_	1	0	0	0	1

Figura 2.4 Ejemplo de una tabla PLA.

PALs

El PAL (Programmable Array Logic) es un caso especial del PLA en el que el arreglo de ANDs es programable y el de ORs es fijo. Sus estructuras son iguales, pero el hecho de queúnicamente el arreglo de ANDs sea programable hace más barato y fácil de programar elPAL en comparación con el PLA.

Cuando se diseña con PALs se deben simplificar las ecuaciones lógicas para que quepan enuno (o más) de los PALs existentes. Los términos AND no se pueden compartir entre dos omás compuertas OR, por lo tanto cada función puede ser simplificada por si misma sin importar los otros términos. En cualquier PAL el número de términos AND que alimentancada compuerta OR es fijo y limitado. Los PALs también pueden contener flip flops D consus entradas provenientes del arreglo combinacional. Estos se llaman PALs secuenciales. Los PALs fueron desapareciendo con el desarrollo de otros dispositivos, como GALs, CPLDs y FPGAs.

4.4.- DISPOSITIVOS LÓGICOS PROGRAMABLES/ GENERIC ARRAY LOGIC

Conforme avanzaba la tecnología de circuitos integrados, una gran variedad de dispositivos lógicos programables apareció. Los PALs tradicionales no son reprogramables, sin embargo, existen ahora PALs borrables y reprogramables con tecnología flash. A veces, a éstos se les llama PLDs

El 22CEV10 es un PLD con tecnología CMOS borrable eléctricamente que puede ser usado para hacer tanto circuitos combinacionales como secuenciales.

Además de los arreglos AND y OR, la mayoría te los PLDs tienen algún tipo de macrobloque que contiene multiplexores y otros bloques programables adicionales. Estos PLDs se llamande acuerdo a sus capacidades de entrada/ salida. Por ejemplo, el 22CEV10 tiene 12 pines de entrada más 10 pines que se pueden programar como entrada o salida (22 en total). Contiene también 10 flop flops D y 10 compuertas OR. Cada compuerta OR dirige una macrocelda lógica de salida. Cada macrocelda contiene uno de los 10 flip flops D, los cuales comparten un reloj común, un reset asícrono de entrada, y un preset síncrono de entrada.

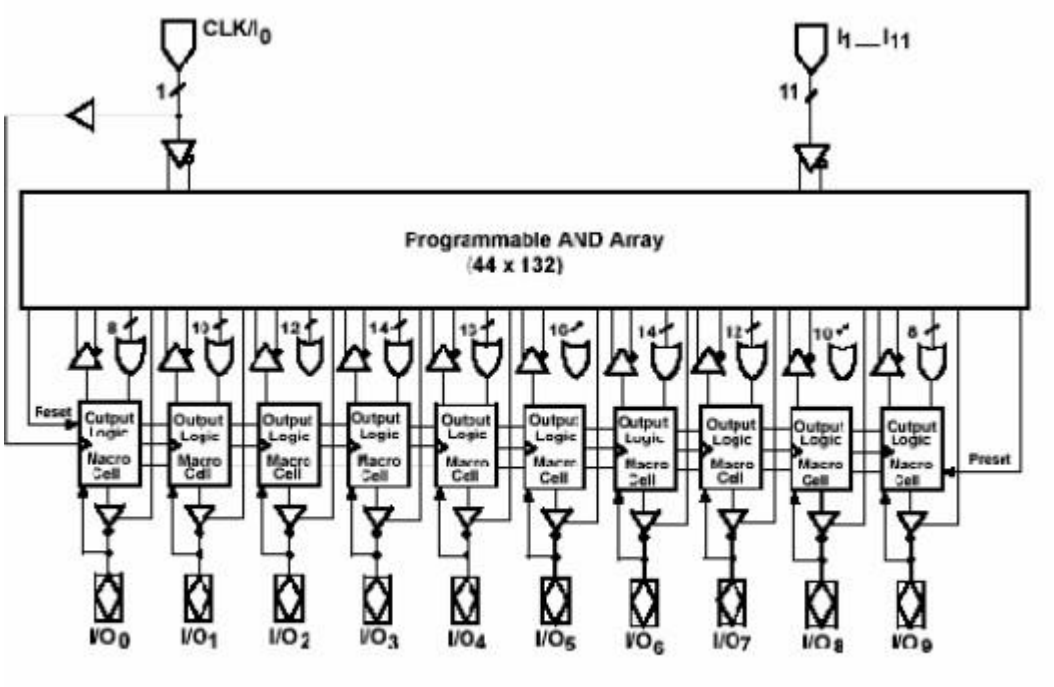


Figura 2.5 Esquema de una GAL 22V10.

La compañía Lattice Semiconductor creó dispositivos similares que tienen la capacidad de serprogramados dentro del circuito (in-circuit programming) y lo llamó Generic Array Logic (GAL). Las GALs son perferctas para implementar pequeñas cantidades de lógica de interfaz. La mayoría de los PLDs, como PALCE22V10, PALCE20V8 entre otros, tienen sus equivalentes en GAL, llamados GAL22V10, GAL20V8, etc.

Existen programas CAD (Computer Aided Design) para PALs y PLDs. Estos programas aceptan ecuaciones lógicas, tablas de verdad, gráficas de estados y demás como entrada paragenerar automáticamente los patrones de bits que se requieren. Posteriormente, un programador puede descargar dichos patrones a los dispositivos para crear las conexionesnecesarias. PALASM y ABEL son

ejemplos de lenguajes que fueron populares como lenguajede diseño para PALs y PLDs, aunque en estos días es posible hacer diseños para GALs en lenguajes como VHDL y Verilog.

BIBLIOGRAFIA

- http://www.uco.es/~el1momoc/BibElDig.html
- https://ocw.ehu.eus/mod/page/view.php?id=1793
- http://www3.uji.es/~mmarques/f47/teoria/tema7.pdf
- https://web-argitalpena.adm.ehu.es/pdf/UCWEB142021.pdf
- http://jagarza.fime.uanl.mx/general/notas/FDDSC.pdf
- https://ocw.unican.es/pluginfile.php/313/course/section/261/tema_03.pdf
- Diseño Digital /por M. Morris Mano y traducción de Julio Fournier González., Mano, M. Morris., DISEÑO LOGICO ; CIRCUITOS INTEGRADOS DIGITALES ; CIRCUITOS LOGICOS ; COMPUTADORES ELECTRONICOS DIGITALES - CIRCUITOS.
- Principios de Diseño Digital /por Daniel D. Gajski, traducción de Carlos GarciaPuntonet y otros., Gajski, Daniel D.