

Técnicas digitales 1

Nieto Tiago

March 23, 2025

Contents

Prefacio	vii
1 Algebra de Bool	1
1.0.1 Teoremas	5
2 Funciones lógicas y su minimización	9
3 Tecnologia	11
4 Circuitos combinacionales básicos	17
4.1 Multiplexores	17
4.2 Demultiplexores	17
4.3 Decodificadores	17
4.4 Codificadores con y sin prioridad	17
4.5 Circuito conversor de binario a Gray y de Gray a binario . . .	18
4.6 circuitos detectores y correctores de error: paridad, código de Hamming	18
4.6.1 Bit paridad	18
4.6.2 Código de Hamming	18
4.7 . Decodificador BCD a 7 segmentos	19
4.8 Desplazadores	19
4.9 Uso del decodificador y del multiplexor como generador de funciones	19
4.10 El operador ternario en Verilog para implementación de mul- tiplexores	19
4.11 Modelado estructural	19
4.12 Diferencia entre una variable y una signal	19
4.13 Código concurrente versus secuencialx	19
5 Circuitos aritméticos	21
5.1 Complemento a la Base y Complemento a la Base-1	21

5.2	Operaciones Aritméticas en CA2	21
5.3	Implementación de Circuitos Aritméticos	21
5.4	Sumador Completo de 1 Bit	21
5.5	Sumador por Propagación de Acarreo	21
5.6	Sumadores Rápidos	21
5.7	Circuitos Detectores de Todos Unos o Ceros	21
5.8	Circuitos Comparadores de Magnitud y de Igualdad	21
5.9	Evaluación de Funciones Matemáticas mediante Tablas	21
5.10	Operaciones con un Operando Constante	21
6	Logica secuencial sincrona	23
6.1	Sistemas Secuenciales	24
6.2	Conceptos de Memoria y Tiempo	24
6.3	Autómata de Mealy y de Moore	24
6.4	Latches SR y D	24
6.5	Flip-Flop Maestro-Esclavo	24
6.6	Flip-Flop Activado por Flanco	24
6.7	Procedimiento de Análisis	24
6.8	Diagrama de Estados	24
6.9	Asignación de Estados	24
6.10	Diseño con Flip-Flop D	24
6.11	Reducción de Estados	24
6.12	Máquinas de Estado en Verilog	24
6.13	Simulación de Máquinas de Estado	24
6.14	Problemas de Aplicación	24
7	Registro y contadores	25
7.1	Introducción	26
7.2	Registros	26
7.3	Registro con Carga en Paralelo	26
7.4	Registro de Desplazamiento	26
7.5	Registro de Desplazamiento con Carga en Paralelo	26
7.6	Registro de Desplazamiento Bidireccional	26
7.7	Estudio de los Registros Según su Capacidad de Almacenamiento	26
7.8	Registros Individuales y Bancos de Registros	26
7.9	Contadores Binarios Sincrónicos Up/Down	26
7.10	Contadores Basados en Registros	26
7.11	Contadores de Anillo	26
7.12	Contadores Johnson	26
7.13	Contadores LFSR (Linear Feedback Shift Register)	26
7.14	Descripción en Verilog	26

8	Logica programable	27
8.1	Lógica Programable: ROM, PLA, PAL, GAL, CPLD y FPGAs	27
8.2	Diferencia entre una CPLD y FPGA	27
8.3	Introducción y Estructura de una FPGA	27
8.4	Arquitectura Básica	27
8.5	LUTs	27
8.6	Memorias de Configuración	27

Prefacio

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetur id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Chapter 1

Algebra de Bool

Desarrollado por George Bool en el 1854. Este método de calculo se utiliza para describir la interconexión de compuertas digitales. En el cual las clases o conjuntos de elementos que pueden tomar los valores '0' o '1'. Dichos elementos estan relacionados con operaciones binarias.

Operaciones binarias

suma logica

producto logico

complementacion o inversion

Postulados

1. Ambas operaciones son conmutativa es decir que si a y b son elementos del álgebra de Bool, se verifica:

$$a + b = b + a$$

$$a \cdot b = b \cdot a$$

2. Posee 2 elementos neutros, el 0 y el 1, que cumplen con la propiedad de identidad
3. Cada operacion es distributiva con respecto a la otra


$$a.(b + c) = a.b + a.c$$

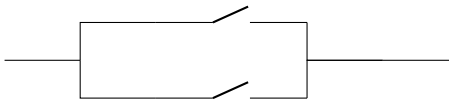
$$a + b.c = (a + b).(a + c)$$

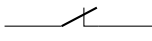
4. Para cada elemento 'a' del algebra de Bool, existe un elemento 'a' que se lo llama negado, tal que:

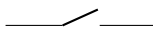
$$a + \bar{a} = 1$$

$$a \cdot \bar{a} = 0$$

AND Operation: 

OR Operation: 

Closed Switch: 

Open Switch: 

Funcion del alg. de Bool:

Termino Canonico= Es cuando aparecen todas las variables, de las cuales la funcion depende ya sea de forma directa o negada.

Dada la función booleana $F(A, B, C)$ verdadera para los minitérminos 1, 2 y 5, su forma canónica es:

$$F(A, B, C) = (A' \cdot B' \cdot C) + (A' \cdot B \cdot C') + (A \cdot B' \cdot C)$$

Suma canonica

Si la función booleana es verdadera para los minitérminos 0, 3 y 7, la forma canónica de la suma sería:

$$F(A, B, C) = (A' \cdot B' \cdot C) + (A' \cdot B \cdot C') + (A \cdot B' \cdot C)$$

Función canónica: Cuando todos sus términos son canónicos, y puede ser expresada como suma de productos o productos de suma

Expresión como Suma de Productos:

$$F(A, B, C) = \sum_{m \in \{1, 2, 5\}} m(A, B, C)$$

Donde cada producto canonico representa un 1

Expresion como Producto de Sumas:

$$F(A, B, C) = \prod_{m \in \{0, 3, 7\}} (A_m + B_m + C'_m)$$

Donde cada suma conica representa un 0

Compuerta OR

Aca dibujo compuerta or

A	B	$A \text{ OR } B$
0	0	0
0	1	1
1	0	1
1	1	1

$$A \text{ OR } B = A + B$$

Compuerta AND

compueta and

A	B	$A \text{ AND } B$
0	0	0
0	1	0
1	0	0
1	1	1

$$A \text{ AND } B = A \cdot B$$

Compuerta NOT

Compuerta NAND

Compuerta NOR

Compuerta XOR

Compuerta XNOR

Acá va el diagrama de la compuerta XNOR.

A	B	$A \text{XNOR } B$
0	0	1
0	1	0
1	0	0
1	1	1

$$A \text{XNOR } B = \overline{A \oplus B}$$

Acá va el diagrama de la compuerta XOR.

A	B	$A \text{XOR } B$
0	0	0
0	1	1
1	0	1
1	1	0

$$A \text{XOR } B = A \oplus B$$

Acá va el diagrama de la compuerta NOR.

A	B	$A \text{NOR } B$
0	0	1
0	1	0
1	0	0
1	1	0

$$A \text{NOR } B = \overline{A + B}$$

Acá va el diagrama de la compuerta NAND.

A	B	$A \text{NAND } B$
0	0	1
0	1	1
1	0	1
1	1	0

$$A \text{ NAND } B = \overline{A \cdot B}$$

Acá va el diagrama de la compuerta NOT.

A	NOT A
0	1
1	0

$$\text{NOT } A = \overline{A}$$

1.0.1 Teoremas

A continuación, se presentan los principales teoremas y propiedades de las compuertas lógicas:

Teoremas Básicos

1. Ley de Identidad:

$$A \cdot 1 = A \quad \text{y} \quad A + 0 = A$$

Esta ley establece que cualquier operación lógica AND con 1 o cualquier operación OR con 0 no altera el valor de la variable.

2. Ley de Anulación:

$$A \cdot 0 = 0 \quad \text{y} \quad A + 1 = 1$$

En esta ley, cualquier operación AND con 0 siempre dará como resultado 0, mientras que cualquier operación OR con 1 siempre dará como resultado 1.

3. Ley de Idempotencia:

$$A \cdot A = A \quad \text{y} \quad A + A = A$$

El valor de una variable AND o OR consigo misma no cambia.

4. Ley de Complementación:

$$A \cdot \overline{A} = 0 \quad \text{y} \quad A + \overline{A} = 1$$

Un valor AND con su complemento siempre da como resultado 0, mientras que un valor OR con su complemento siempre da 1.

5. Ley de Conmutación:

$$A \cdot B = B \cdot A \quad \text{y} \quad A + B = B + A$$

El orden de las variables no afecta el resultado de la operación AND u OR.

6. Ley de Asociatividad:

$$A \cdot (B \cdot C) = (A \cdot B) \cdot C \quad \text{y} \quad A + (B + C) = (A + B) + C$$

El resultado de las operaciones lógicas AND u OR no depende de cómo se agrupan las variables.

7. Ley de Distributividad:

$$A \cdot (B + C) = (A \cdot B) + (A \cdot C) \quad \text{y} \quad A + (B \cdot C) = (A + B) \cdot (A + C)$$

La ley de distributividad establece cómo se distribuyen las operaciones lógicas entre AND y OR.

Teoremas de las Compuertas Lógicas**8. Ley de De Morgan:**

$$\overline{A \cdot B} = \overline{A} + \overline{B} \quad \text{y} \quad \overline{A + B} = \overline{A} \cdot \overline{B}$$

Esta ley establece la relación entre la negación de una operación AND y OR.

9. Teorema de la Absorción:

$$A \cdot (A + B) = A \quad \text{y} \quad A + (A \cdot B) = A$$

El teorema de la absorción dice que una operación AND u OR con una variable y su combinación con otra no altera el resultado.

10. Teorema de la Involución:

$$\overline{\overline{A}} = A$$

La doble negación de una variable da como resultado la variable original.

Ejemplo de Aplicación

Supongamos que tenemos la siguiente expresión booleana que representa una red de compuertas lógicas:

$$Y = (A \cdot B) + \overline{C}$$

Queremos simplificar esta expresión utilizando los teoremas lógicos. Comenzamos aplicando la ley de distribución:

$$Y = (A \cdot B) + \overline{C}$$

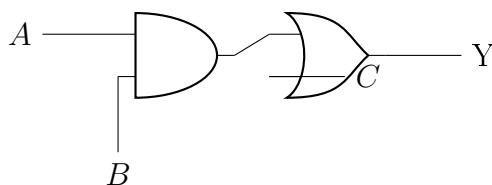
Aplicando la ley de involución en \overline{C} :

$$Y = (A \cdot B) + \overline{\overline{C}} = (A \cdot B) + C$$

Y hemos simplificado la expresión de la siguiente forma:

$$Y = A \cdot B + C$$

Finalmente, podemos representar esta expresión con un circuito lógico utilizando compuertas AND y OR, lo que nos da la siguiente representación:



La expresión booleana $Y = A \cdot B + C$ está representada por un circuito lógico que utiliza una compuerta AND y una compuerta OR. Este es un ejemplo básico de cómo simplificar y representar una expresión lógica utilizando teoremas.

Chapter 2

Funciones lógicas y su minimización

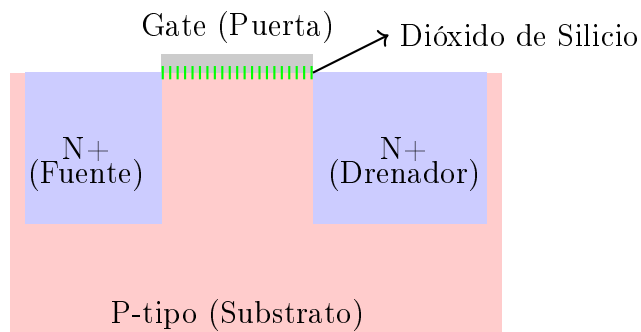
Chapter 3

Tecnologia

Las compuertas logicas estan diseñadas mayormente con transistores, los cuales estan echo de silicio, un pobre conductor, por lo que se dopa:

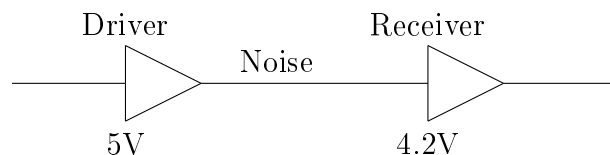


Transistores CMos



Los voltajes se representan con 1(Vcc) y 0 (Gnd)

Ruido: se le define como toda cosa que degrada la señal, como la resistencia, la fuente, cables vecinos, etc.



Donde:

1. VOH : Minimo valor de salida.

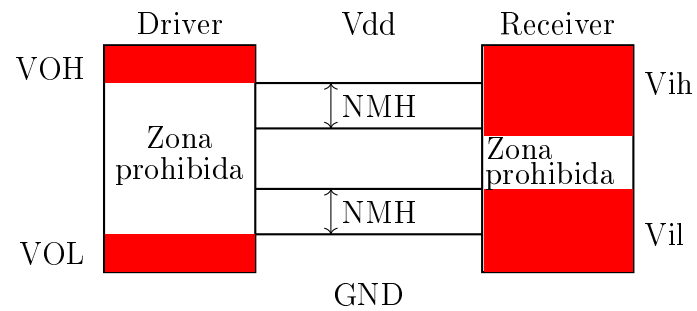


Figure 3.1: Noise Margins Diagram with Driver and Receiver

2. VOL : Maximo valor salida
 3. VIL : Maximo valor de entrada
 4. VIL : Minimo valor de entrada
 5. NMH : Margen de ruido a nivel alto
 6. NML : Margen de ruido a nivel bajo
- $$NML = VOH - VIH \quad NML = VIL - VOL$$

Funciones

los nMos: pasan buenos 0, se conecta el source a GND

los pMOS: pasan buenos 1, se conecta el source a VDD

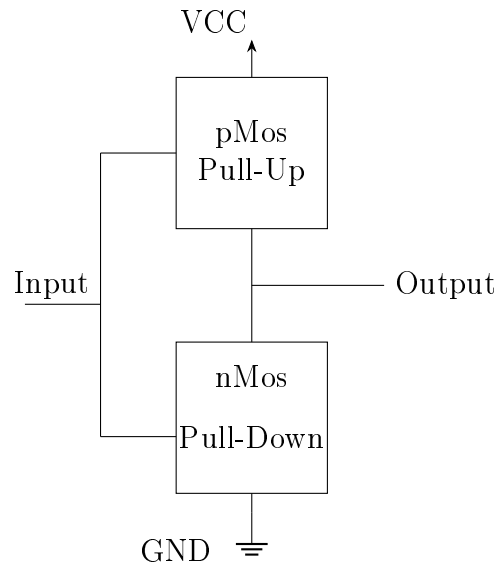
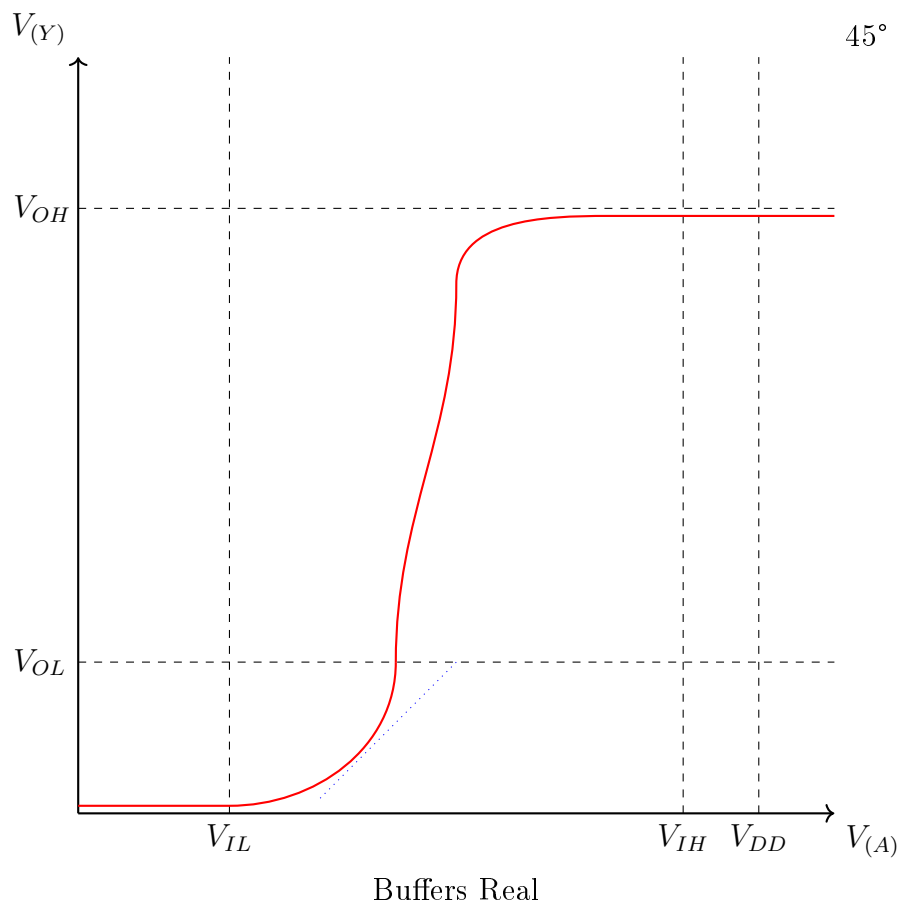
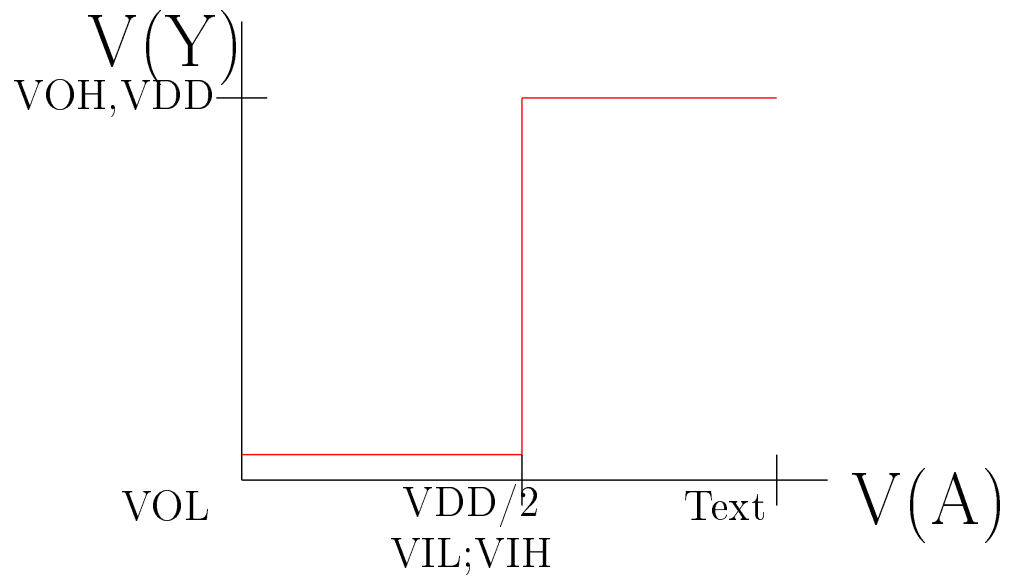
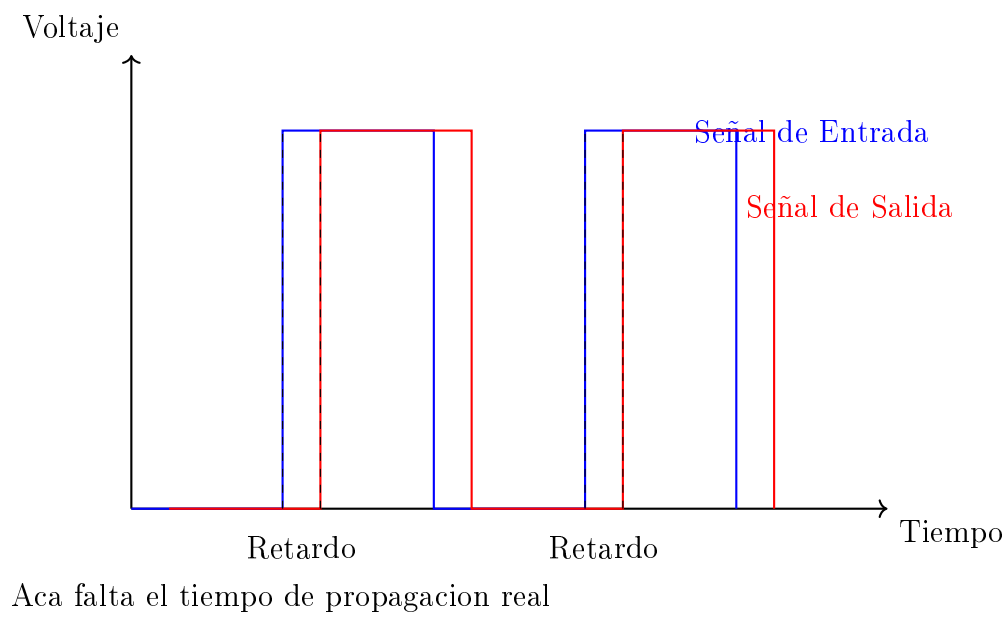


Figure 3.2: Diagrama CMOS con pMos Pull-Up y nMos Pull-Down





Chapter 4

Circuitos combinacionales básicos

Los circuitos combinacionales logicos pueden ser combinacionales o secuenciales.

Combinacional: Consiste en compuertas logicas cuyas salidas estan determinadas por la combinacion de entradas

Es decir, que la operacion que realiza se puede especificar logicamente con un conjunto de funciones booleanas.

Aceotan señales de entrada y generan señales

4.1 Multiplexores

Es un circuito combinacional que nos permite encaminar una señal de entrada entre varias posible (2 a la n) hacia una unica salida

4.2 Demultiplexores

4.3 Decodificadores

Un codigo binario de n bits puede representar hasta 2^a la n +elementos distintos de informacioin codificada

Un decodificador es un circuito combinacional que convierte informacion binaria de n lienas de entrada a un maximo de 2^n lineas de salidas distintas+

4.4 Codificadores con y sin prioridad

El codificador tiene 2 a la n (o menos) lineas de entrada y n lineas de salidas. Estas ultinmas genereal el codigo correspondiente al valor de entrada

Prioridad

Asigna un orden opriorida a las entradas, osea que siempre selecciona la entrada con mas prioridad

4.5 Circuito conversor de binario a Gray y de Gray a binario

4.6 circuitos detectores y correctores de error: paridad, código de Hamming

Distancia: Es el numero de bit que deben ser modificados para pasar de una combinacion a otra

4.6.1 Bit paridad

Es una tecinca para la deteccion de errores, que consiste en fijar una condicion para la transiicion.

Paridad par(Pp): La cantidad de 1 que tenga la palabra transmitida debe ser par

Paridad impar(PI): La cantidad de 1 que tenga la palabra transmitida debe ser impar

Para lograr esto de necesita adicionar un bit al final de cada palabra del codigo, llamado "bit de control de paridad"

Palabra codigo + bit paridad = palabra de transmicion

4.6.2 Codigo de Hamming

El codigo de hamming utiliza tamien el control de pareidad pero no sobre la palabra sino por grupos de bitd

bits de palabra codigo + bits de paridad = bits de la palabra de transmicion

Como el codigo de Hamming es corrector, cuando se produce un error, se puede conocer el bit equivocado y tenemos un grupo de controladores, que verifican la paridad de cada hrupo y su salida

Si hay NT bits en la palabra de transmicion y cualq uiera de ellos puede ffallar necesito NT+1 indicadores posibles ene la salidad e los controladorees, en el punto de recepcion

- 4.7 . Decodificador BCD a 7 segmentos
- 4.8 Desplazadores
- 4.9 Uso del decodificador y del multiplexor como generador de funciones
- 4.10 El operador ternario en Verilog para implementación de multiplexores
- 4.11 Modelado estructural
- 4.12 Diferencia entre una variable y una signal
- 4.13 Código concurrente versus secuencialx

Chapter 5

Circuitos aritméticos

- 5.1 Complemento a la Base y Complemento a la Base-1
- 5.2 Operaciones Aritméticas en CA2
- 5.3 Implementación de Circuitos Aritméticos
- 5.4 Sumador Completo de 1 Bit
- 5.5 Sumador por Propagación de Acarreo
- 5.6 Sumadores Rápidos
- 5.7 Circuitos Detectores de Todos Unos o Ceros
- 5.8 Circuitos Comparadores de Magnitud y de Igualdad
- 5.9 Evaluación de Funciones Matemáticas mediante Tablas
- 5.10 Operaciones con un Operando Constante

Chapter 6

Logica secuencial sincrona

- 6.1 Sistemas Secuenciales
- 6.2 Conceptos de Memoria y Tiempo
- 6.3 Autómata de Mealy y de Moore
- 6.4 Latches SR y D
- 6.5 Flip-Flop Maestro-Esclavo
- 6.6 Flip-Flop Activado por Flanco
- 6.7 Procedimiento de Análisis
- 6.8 Diagrama de Estados
- 6.9 Asignación de Estados
- 6.10 Diseño con Flip-Flop D
- 6.11 Reducción de Estados
- 6.12 Máquinas de Estado en Verilog
- 6.13 Simulación de Máquinas de Estado
- 6.14 Problemas de Aplicación

Chapter 7

Registro y contadores

7.1 Introducción

7.2 Registros

7.3 Registro con Carga en Paralelo

7.4 Registro de Desplazamiento

7.5 Registro de Desplazamiento con Carga en Paralelo

7.6 Registro de Desplazamiento Bidireccional

7.7 Estudio de los Registros Según su Capacidad de Almacenamiento

7.8 Registros Individuales y Bancos de Registros

7.9 Contadores Binarios Sincrónicos Up/Down

7.10 Contadores Basados en Registros

7.11 Contadores de Anillo

7.12 Contadores Johnson

7.13 Contadores LFSR (Linear Feedback Shift Register)

Chapter 8

Logica programable

- 8.1 Lógica Programable: ROM, PLA, PAL, GAL, CPLD y FPGAs
- 8.2 Diferencia entre una CPLD y FPGA
- 8.3 Introducción y Estructura de una FPGA
- 8.4 Arquitectura Básica
- 8.5 LUTs
- 8.6 Memorias de Configuración

