Unidad 2. Circuitos digitales

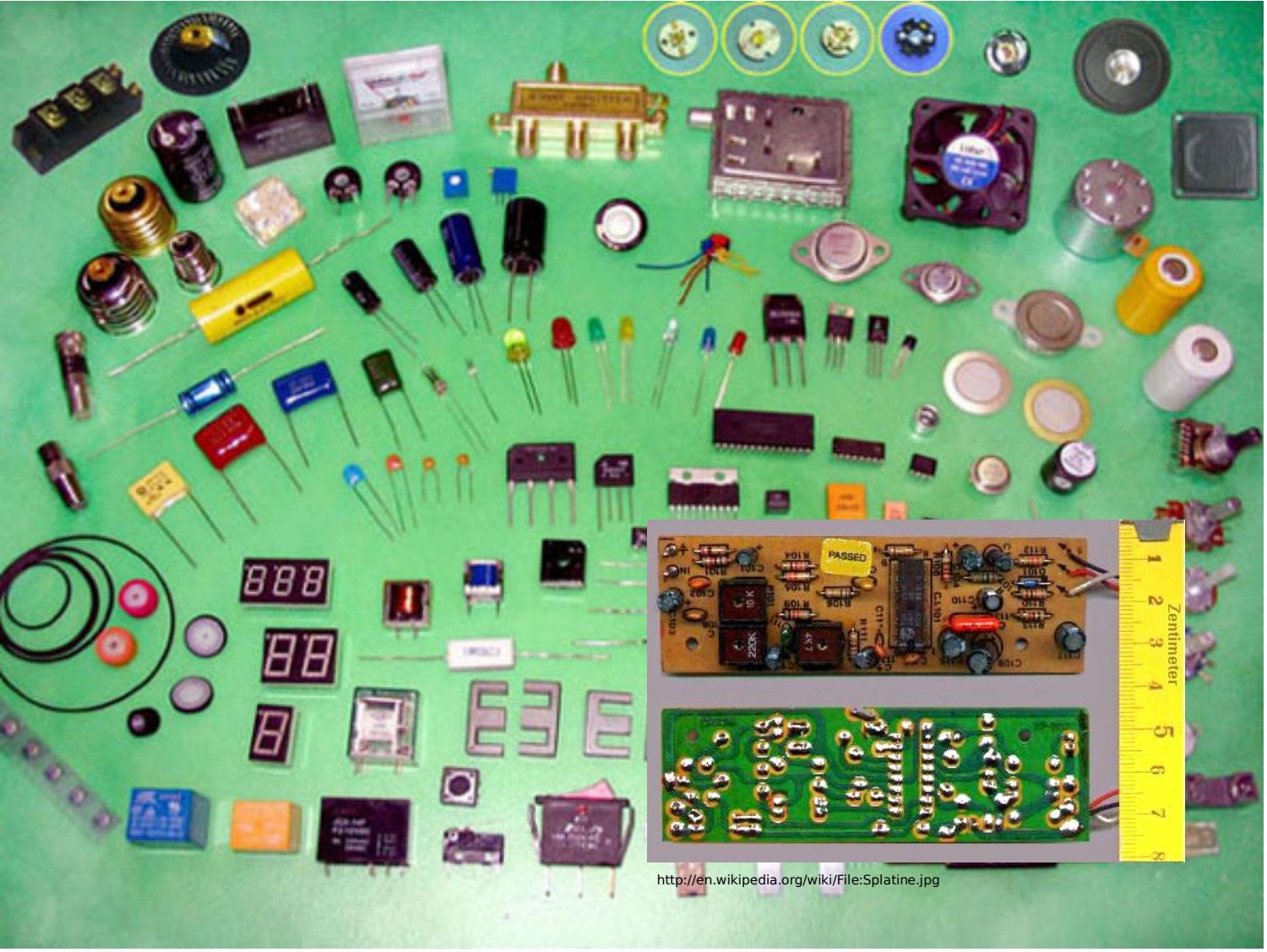
DAPA
E.T.S.I. Informática
Universidad de Sevilla
Oct. 2015

Jorge Juan <jjchico@dte.us.es> 2010-2016

Contenido

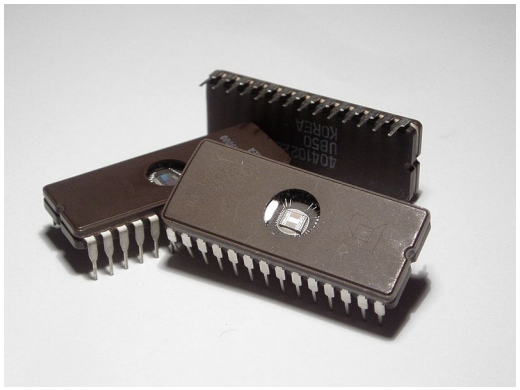
- **Circuitos digitales**
- Funciones lógicas
- Álgebra de Boole
- Diseño combinacional
- Análisis funcional
- Análisis temporal

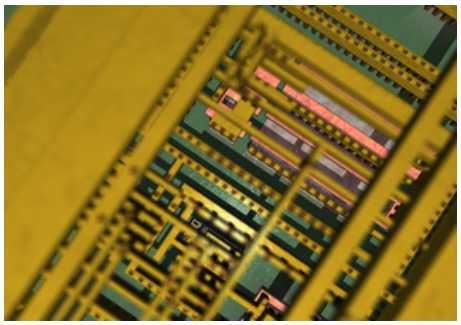
Departamento de Tecnología Electrónica – Universidad de Sevilla



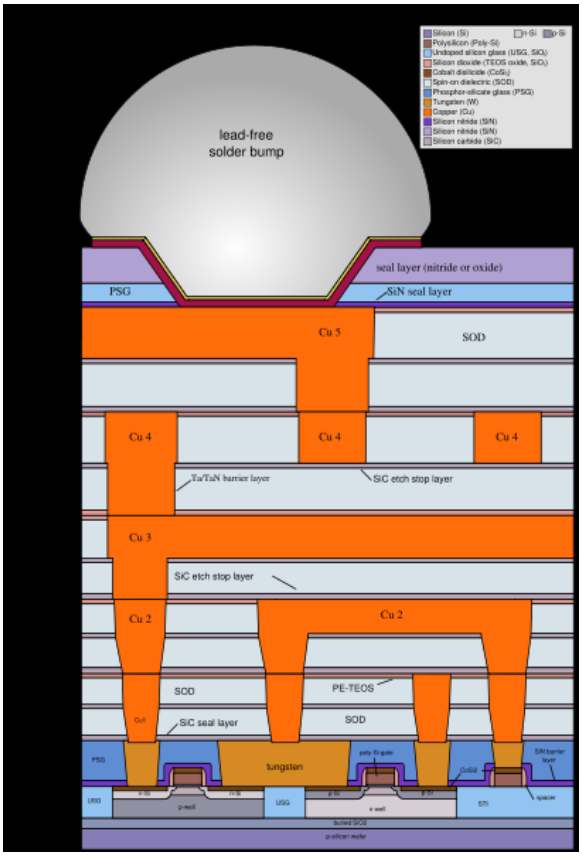
http://en.wikipedia.org/wiki/File:Splatine.jpg

Circuitos integrados



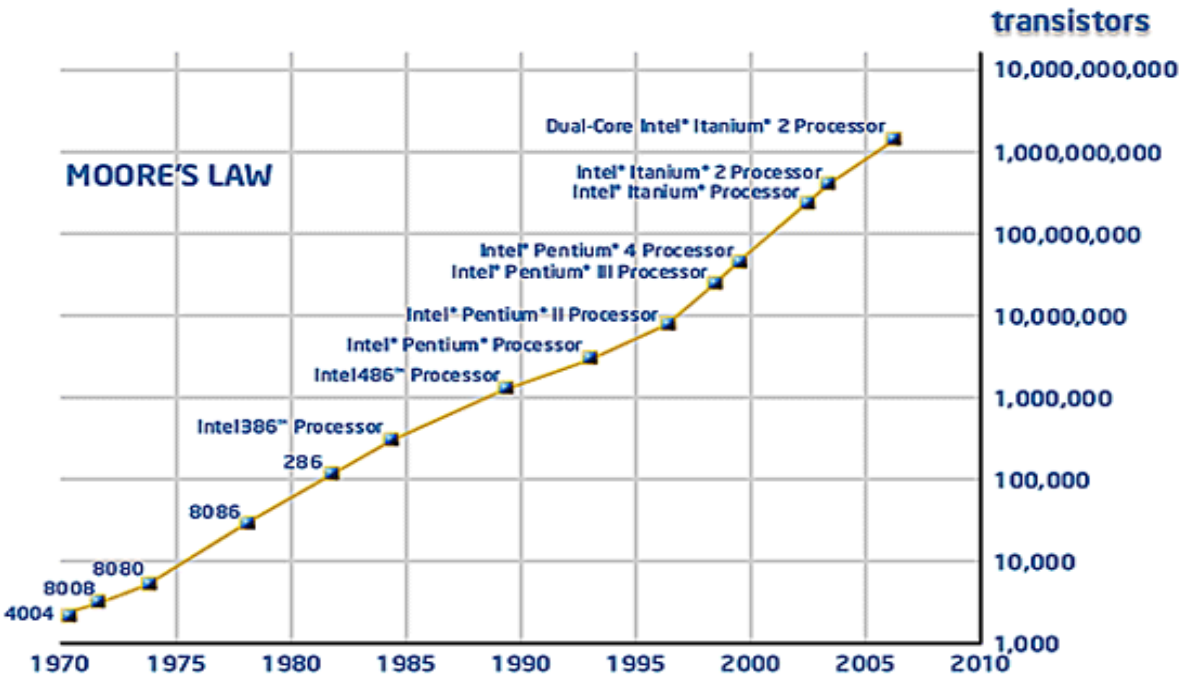


http://en.wikipedia.org/wiki/Integrated_circuit

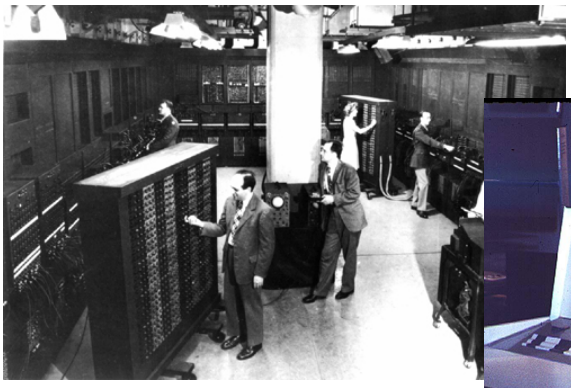


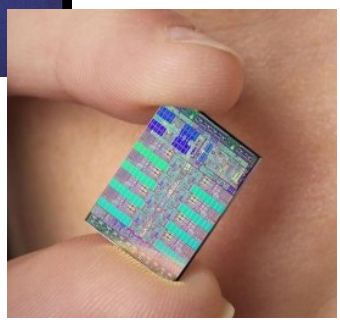
Circuitos integrados. Evolución

- Ley de Moore (1964): “La densidad de integración se duplica cada 18 meses”.



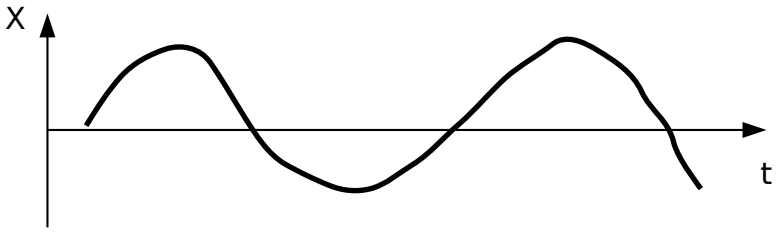
Evolución



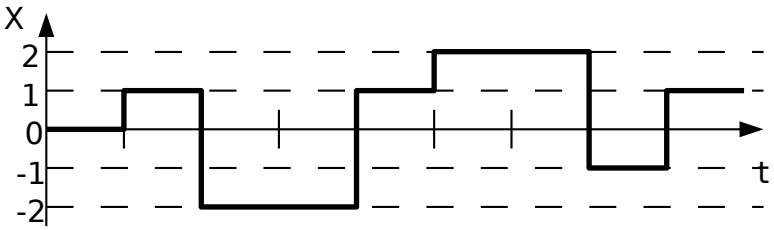


Circuitos digitales: 0s y 1s

Analógico

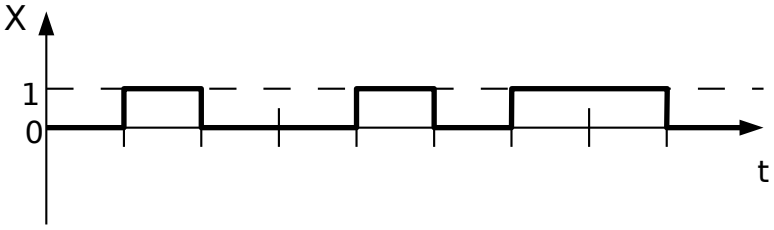


Digital
(5 valores)



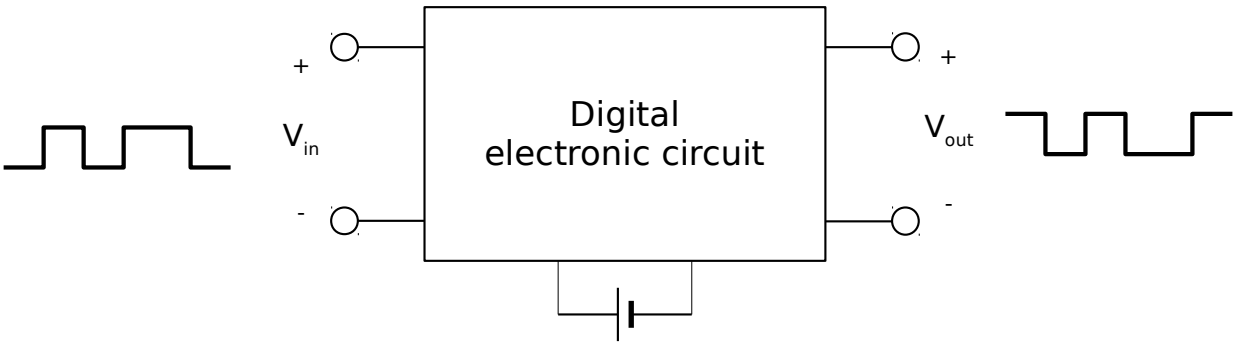
Sequence: 0, 1, -2, -2, 1, 2, 2, -1, 1

Digital
(2 valores)



Sequence: 0, 1, 0, 0, 1, 0, 1, 1, 0

Circuito digital

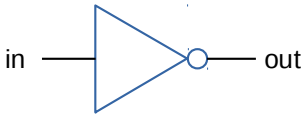


Valor eléctrico (V_X) vs valor lógico (X)

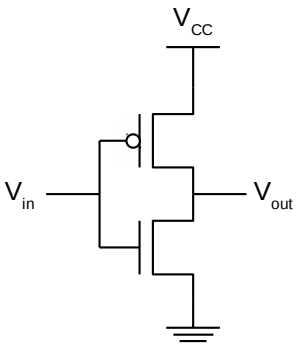
V_X	X
0V	0
3,3V	1

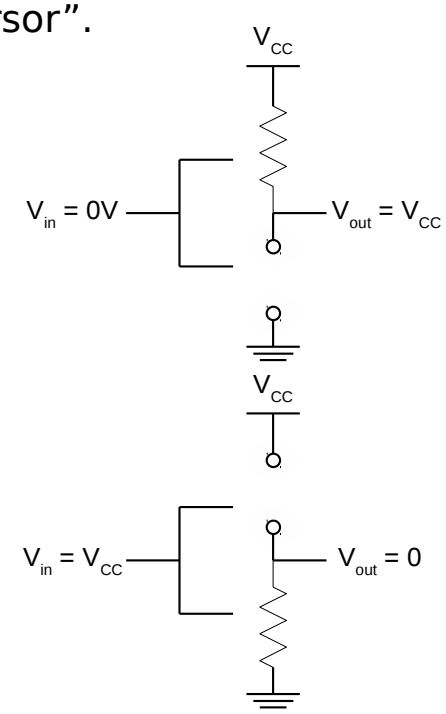
Puertas lógicas. Inversor CMOS

- Las puertas lógicas realizan operaciones simples sobre señales digitales.
- La operación digital más simple (salvo la identidad) es la inversión, implementada por el “inversor”.

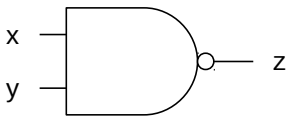


in	out
0	1
1	0

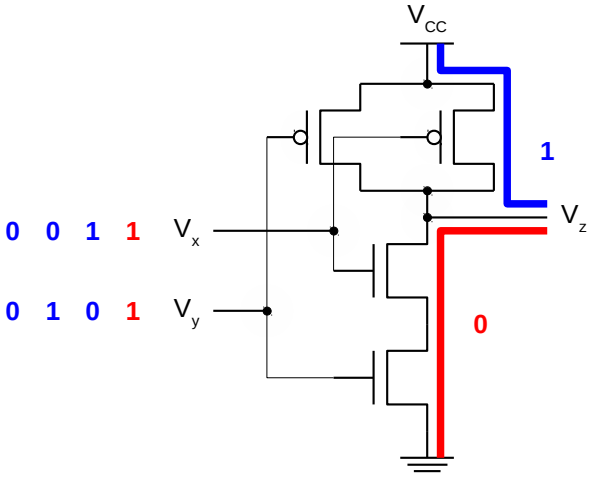




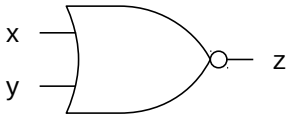
Puertas lógicas. NAND CMOS

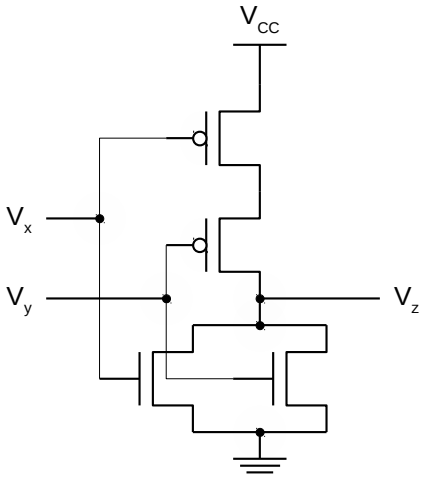


Las puertas CMOS son “naturalmente” inversoras (INV, NAND, NOR, …)



Puertas lógicas. NOR CMOS



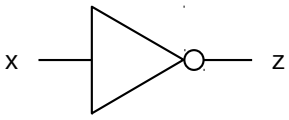


Operadores lógicos básicos

NOT

x	z
0	1
1	0

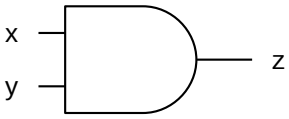
$z = \overline{x}$



AND

x y	z
0 0	0
0 1	0
1 0	0
1 1	1

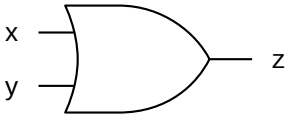
$z = x \cdot y$



OR

x y	z
0 0	0
0 1	1
1 0	1
1 1	1

$z = x + y$

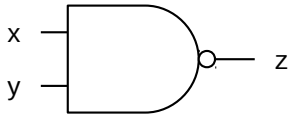


Otros operadores lógicos

NAND

x y	z
0 0	1
0 1	1
1 0	1
1 1	0

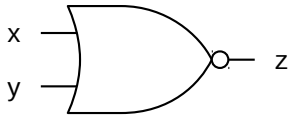
$z = \overline{x \cdot y}$



NOR

x y	z
0 0	1
0 1	0
1 0	0
1 1	0

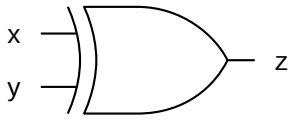
$z = \overline{x + y}$



XOR

x y	z
0 0	0
0 1	1
1 0	1
1 1	0

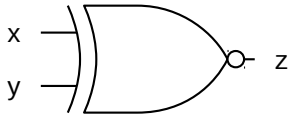
$z = x \oplus y = \overline{x}y + x\overline{y}$



XNOR

x y	z
0 0	1
0 1	0
1 0	0
1 1	1

$z = \overline{x \oplus y} = \overline{x}\,\overline{y} + xy$



Contacto con el diseño digital

- Dieseñar un circuito de alarma con un detector de presencia y un sensor de fuego
- Entradas:
  - a: alarma conectada (1) o alarma desconectada (0)
  - b: sensor de presencia (1) o no presencial (0)
  - c: sensor de fuego (1) o no fuego (0)
- Salida:
  - z: alarma activada (1) o no activada (0)
- Ejercicio:
  - Diseñar un circuito electrónico digital que genere la salida correcta en función del valor de las entradas mediante la interconexión de puertas lógicas.

Contenido

- Circuitos digitales
- **Funciones lógicas**
- Álgebra de Boole
- Diseño combinacional
- Análisis funcional
- Análisis temporal

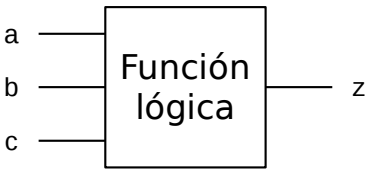
Funciones lógicas. Ejemplo

Descripción verbal

Un sistema de alarma digital tiene puede estar activado o desactivado y dispone de un sensor de presencia y otro de apertura de puerta. Cuando el sistema está desactivado, se activará una señal de alarma si se detecta presencia y la puerta está abierta (se ha olvidado cerrar la puerta). Cuando el sistema está activado, se activará la señal de alarma si se detecta presencia o se abre la puerta.

Descripción formal

a: 0-desactivado, 1-activado
b: 0-no presecia, 1-presencia.
c: 0-p. cerrada, 1-p. abierta



a b c	z
0 0 0	0
0 0 1	0
0 1 0	0
0 1 1	1
1 0 0	0
1 0 1	1
1 1 0	1
1 1 1	1

- Tabla de verdad: especificación formal. Valor de la función para cada valor de las variables (n variables: 2^n valores).
- Los circuitos digitales combinacionales implementan funciones lógicas

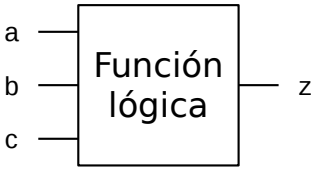
Expresiones lógicas

- Combinación de operadores AND (·), OR (+) y NOT (¯ ')
- Precedencia de · sobre +
  - x+(y·z) = x+y·z
- "·" puede ser omitido
  - x+y·z = x+yz
- Son una forma de representación de funciones lógicas

$$f(a,b,c) = (a+b+c)\,(a + b\,c) + cd\,(a+c)$$

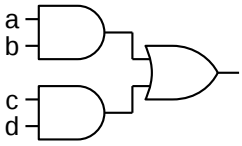
Funciones lógicas. Resumen

- Es posible construir circuitos elementales que realizan los operadores lógicos básicos (puertas lógicas)
- El objetivo del diseño combinacional es describir las funciones lógicas mediante operadores básicos que permitan construir un circuito digital que realice la función deseada



a b c	z
0 0 0	0
0 0 1	0
0 1 0	0
0 1 1	1
1 0 0	0
1 0 1	1
1 1 0	1
1 1 1	1

Circuito combinacional



Contenido

- Circuitos digitales
- Funciones lógicas
- **Álgebra de Boole**
- Diseño combinacional
- Análisis funcional
- Análisis temporal

Formalización: Álgebra de Boole

- B={0, 1}
- {B, NOT, AND, OR} forman un Álgebra de Boole
  - Caso particular: Álgebra de Conmutación
- Un Álgebra de Boole cumple los siguiente axiomas

George Boole (1815-1864)

Identidad	$x+0 = x$	$x \cdot 1 = x$
Conmutativa	$x+y = y+x$	$x \cdot y = y \cdot x$
Distributiva	$x \cdot (y+z) = (x \cdot y)+(x \cdot z)$	$x+(y \cdot z) = (x+y) \cdot (x+z)$
Complemento	$x+\overline{x} = 1$	$x \cdot \overline{x} = 0$

Dualidad: si una expresión se cumple, la expresión que resulta de intercambiar + con · y 0 con 1 también se cumple

Álgebra de Boole. Teoremas

Idempotencia	$x+x = x$	$x \cdot x = x$
Unicidad del complemento	$\overline{x}$ es único	
Elementos dominantes	$x+1 = 1$	$x \cdot 0 = 0$
Involución	$\overline{(\overline{x})} = x$	
Absorción	$x+xy = x$	$x \cdot (x+y) = x$
Consenso	$x+\overline{x}y = x+y$	$x \cdot (\overline{x}+y) = x \cdot y$
Asociativa	$x+(y+z) = (x+y)+z$	$x \cdot (y \cdot z) = (x \cdot y) \cdot z$
De Morgan	$\overline{x \cdot y} = \overline{x}+\overline{y}$	$\overline{x+y} = \overline{x} \cdot \overline{y}$
Reducción	$xy+x\overline{y} = x$	$(x+y)(x+\overline{y}) = x$

Conversión. SOP/POS a tabla de verdad

$z(a,b,c) = a + a\,\overline{b}\,c + ac + \overline{b}c$

- z = 1 si y sólo si
  - a=1, o bien
  - a=1 y b=0 y c=1, o bien
  - a=1 y c=1, o bien
  - b=0 y c=1

$z(a,b,c) = (a+b+c)(a+\overline{b})(a+c)$

- z = 0 si y sólo si
  - a=0 y b=0 y c=0, o bien
  - a=0 y b=1, o bien
  - a=0 y c=0

Contenido

- Circuitos digitales
- Funciones lógicas
- Álgebra de Boole
- **Diseño combinacional**
- Análisis funcional
- Análisis temporal

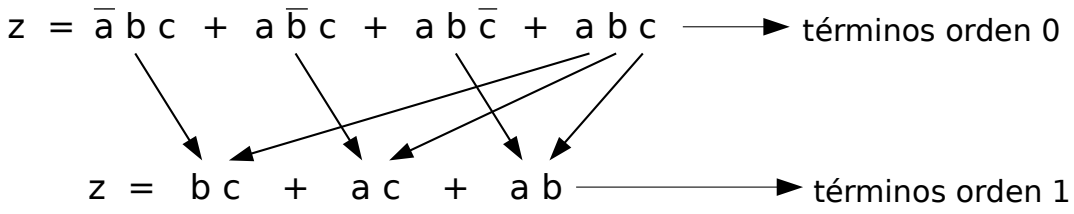
Diseño de circuitos combinacionales

- Basado en la síntesis de circuitos a partir de formas normalizadas (SOP o POS)
- Una expresión más simple dará un circuito más simple
- Las formas normalizadas pueden ser simplificadas de forma sistemática (automatizable)
- Criterios de simplificación básicos
  - Expresiones equivalentes: mismos valores para las mismas entradas.
  - Mínimo número de términos → mínimo número de puertas
  - Mínimo número de literales en cada término → mínimo número de entradas en cada puerta
- Otros criterios
  - Tipos de puertas disponibles
  - Facilidad para hacer complementos (inversores)
  - Consumo de potencia, velocidad de operación, etc.

Simplificación de formas normalizadas

$$xy + x\bar{y} = x$$

"x" puede ser cualquier expresión



Cada término puede ser usado más de una vez (x+x=x)

Implicante de una función

- Término producto que puede ser parte de una expresión en forma de SOP de la función.
- Contiene o “cubre” varios mintérminos de la función.

Algoritmo de minimización Quine-McCluskey (simp.)

1. Comenzar con la lista de los mintérminos de la función
2. Buscar implicantes de primer orden (TO-1)

   Aplicando el teorema de reducción a mintérminos adyacentes
3. Eliminar mintérminos cubiertos de la expresión final

   No hay que preocuparse de los mintérminos cubiertos porque están incluídos en los TO-1
4. Buscar implicantes de segundo orden (TO-2)

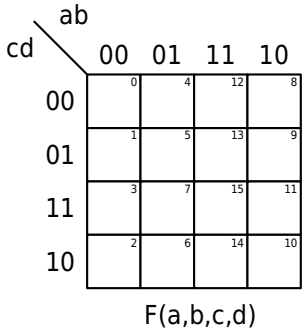
   Aplicando el teorema de reducción a los TO-1 adyacentes
5. Eliminar los TO-1 cubiertos de la expresión final
6. Continuar hasta obtener implicantes del mayor orden posible (implicantes primos)

   Las implicantes primos son los términos más simples que contienen todos los mintérminos de la función.

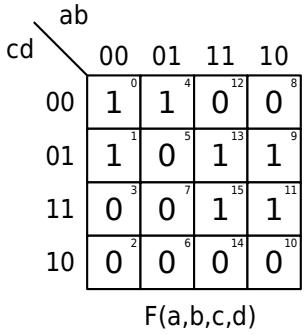
   Las implicantes primos esenciales son los que contienen mintérminos no contenidos en ningún otro implicante primo.
7. Selecciona todas las implicantes primos esenciales y/o un número mínimo de ellos que cubra todos los mintérminos de la función.

Mapas de Karnaugh (K-mapa)

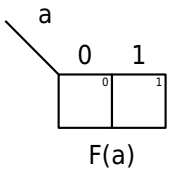
- K-mapas:
  - Tabla bidimensional
  - Valores de variables ordenados como en código gray
  - Cada celda corresponde a un mintérmino/maxtérmino
  - Localización fácil de mintérminos y términos de orden superior
  - ¡Son cíclicos!
  - Localización fácil de implicantes primos
  - Facilitan la simplificación de expresiones normalizadas mediente el método de Quine-McCluskey

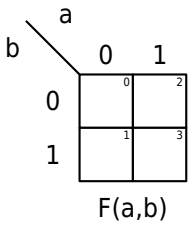


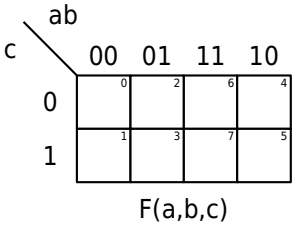
$$F(a,b,c,d) = \Sigma(0,1,4,9,11,13,15)$$

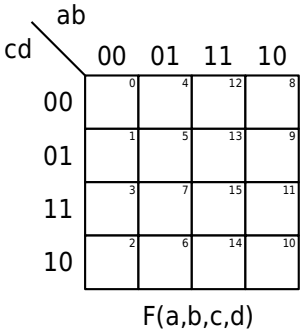


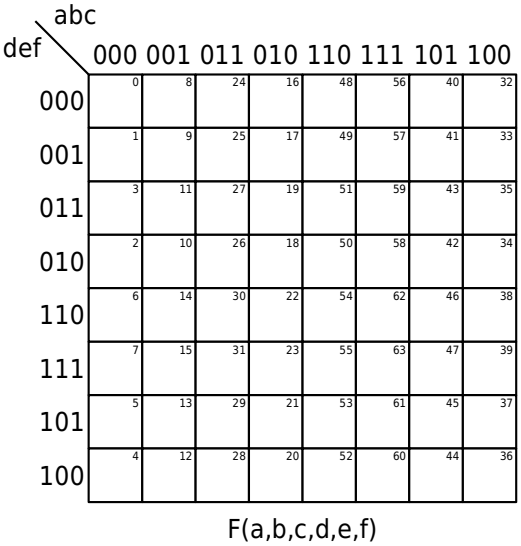
K-mapas de 1 a 6 variables

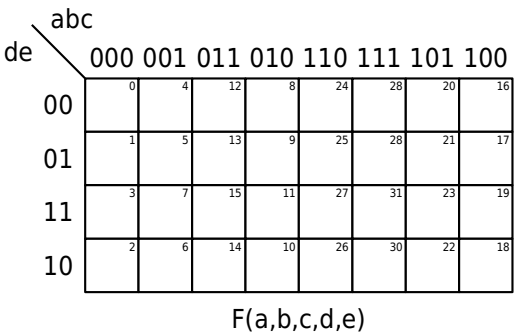




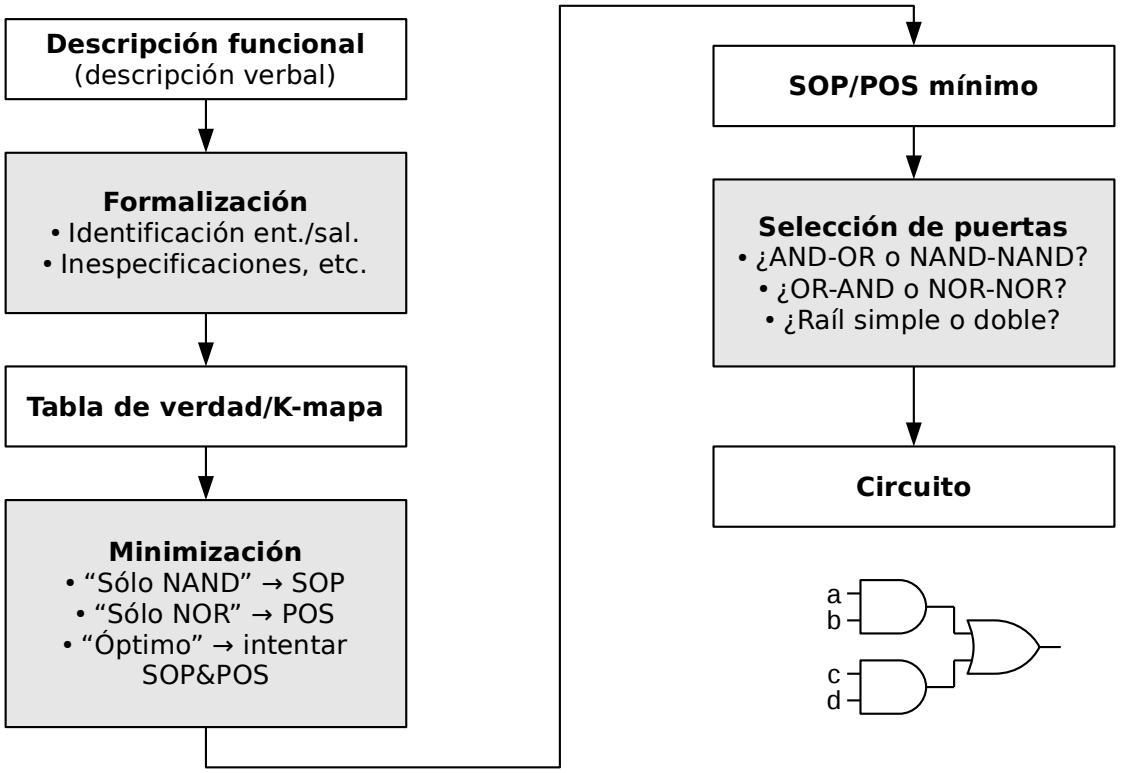








Diseño “manual” de CC (resumen)



Diseño de circuitos combinacionales

Ejemplo 1

Diseñe un circuito combinacional con cuatro entradas (x3, x2, x1, x0) que representan los bits de una cifra BCD X, y dos salidas (c1, c0) que representan los bits de una magnitud C, donde C es el cociente de la división X/3.

Por ejemplo, si X=7 → C=2, esto es, si (x3,x2,x1,x0)=(0,1,1,1) → (c1,c0)=(1,0)

Diseñe un circuito mínimo en dos niveles de puertas, salvo inversores, con entradas en raíl simple.

Ejemplo 2

Diseñe un circuito combinacional para el ejemplo de la introducción.

Contenido

- Circuitos digitales
- Funciones lógicas
- Álgebra de Boole
- Diseño combinacional
- **Análisis funcional**
- Análisis temporal

Análisis funcional

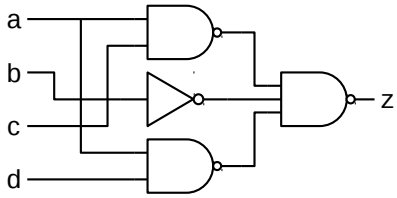
- ¿Qué es?
  - Obtener la función lógica realizada por un circuito combinacional.
- Aplicaciones:
  - Interpretar la operación y/o utilidad del circuito.
  - Rediseñar un circuito equivalente con distintos componentes.
- Método:
  - Identificar entradas y salidas.
  - Para cada puerta con entradas conocidas, calcular la expresión lógica de salida de la puerta.
  - Repetir hasta que se conocen las expresiones de todas las salidas del circuito.
  - Convertir a una representación más útil: tabla de verdad, K-mapa, expresión mínima, etc.
  - Dar una descripción verbal de la operación del circuito.

Análisis funcional. Ejemplo

El circuito de la figura corresponde a un sistema de alarma con cuatro entradas y una salida. Las entras corresponden a:
a: activación del sistema (0 - desactivado, 1 - activado)
b: sensor de fuego (0 - no hay fuego, 1 - sí hay fuego)
c: sensor de puerta de entrada (0 - puerta cerrada, 1 - puerta abierta)
d: sensor de presencia (0 - no hay presencia, 1 - sí hay presencia)

Cuando la salida z se activa (z=1) hace sonar la sirena de la alarma.

a)Analice el circuito y describa su operación con palabras: casos en los que se activará la alarma, etc.
b)Rediseñe el circuito empleando únicamente puertas NOR.



Contenido

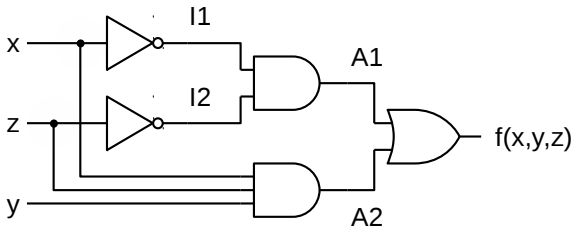
- Circuitos digitales
- Funciones lógicas
- Álgebra de Boole
- Diseño combinacional
- Análisis funcional
- **Análisis temporal**

Departamento de Tecnología Electrónica – Universidad de Sevilla

Análisis temporal

- ¿Qué es?
  - Estudiar la evolución con el tiempo de las señales internas de un circuito y de sus salidas para señales de entrada dadas.
  - Forma de onda: representación temporal del valor de una señal.
- Aplicaciones
  - Calcular o estimar el retraso de propagación de un circuito: tiempo que tarda en proporcionar un valor de salida correcto.
  - Analizar posibles fallos o comportamientos no esperados debido a características temporales: retraso excesivo, azares, etc.
- Método “manual”
  - Para cada puerta lógica, obtener la ecuación de salida en función de las entradas de la puerta.
  - Sustituir los valores de las señales de entrada que son constantes (NO SUSTITUIR NADA MÁS)
  - Dibujar las formas de onda desde las entradas primarias hasta las salidas en función de la operación de cada puerta. Aplicar posibles retrasos de cada puerta. Modelo simple: mismo retraso para todas las puertas.

Análisis temporal. Ejemplo



$I1 = \overline{x}$
$I2 = \overline{z}$
$A1 = I1\ I2$
$A2 = x\ y\ z$
$f = A1 + A2$

y = 1

$I1 = \overline{x}$
$I2 = \overline{z}$
$A1 = I1\ I2$
$A2 = x\ z$
$f = A1 + A2$