FUNDAMENTOS DE LOS COMPUTADORES: Guillermo (coordinador y profesor)

DESPACHO: C-233

guillermo.gdrivera@uam.es

CIRCUITOS/SEÑALES ANALÓGICAS: Señal que adopta un valor entre un mínimo y un máximo.

Ej: La temperatura

CIRCUITOS/SEÑALES DIGITALES: Señal que adopta un valor: el valor máximo o el valor mínimo.

Ej: los 1 y 0.

- T1 - Álgebra de Boole y Diseño Lógico
- T2 - Circuitos combinacionales
- T3 - Elementos básicos de la lógica secuencial.
- T4 - Circuitos secuenciales
- T5 - Componentes de memorización
- T6 - Representación digital de los números.

Fundamentos de Sistemas Digitales. Thomas L. Floyd. Prentice Hall.

EVALUACIÓN

- Teoría y Práctica sobre 10 (no son independientes)
- Calificación final: $\boxed{0'4 \cdot Not\_Lab + 0'6 \cdot Not\_Teoría}$
- Para aprobar → 5 en ambas

EVALUACIÓN CONTINUA

- CASO A → Not_Teo ≥ 5 ⇒ APROBADO siempre que no se haya suspendido una de las dos pruebas parciales propuestas con una nota inferior a 4.

$\boxed{Not\_Teo : 0'4 \cdot ExaParcial1 + 0'5 \cdot ExaParcial2 + 0'1 \cdot RestoActividades}$

CASO B: si no se aprueba ⇒ Examen de ...

FOTO Moodle

(T1) ÁLGEBRA DE BOOLE. DISEÑO LÓGICO

1.1. Analógico vs Digital
1.2. Sistema numérico binario. Conversión entre sistemas
1.3. Propiedades y teoremas básicos del álgebra de Boole
1.4. Funciones lógicas
1.5. Mapas de Karnaugh

ANALÓGICO VS DIGITAL

SISTEMAS ANALÓGICOS

- Trabajan con variables analógicas. (infinitos valores)
- Señales físicas para representarlas: Señales analógicas
- Señal analógica: toma valores infinitos
- Ejemplo: la temperatura [termómetro de mercurio]

SISTEMAS DIGITALES

- Trabajan con variables digitales
  - toma valores entre dos posibles
  - los valores se expresan por sentencias declarat...
  - los dos valores son excluyentes entre ellos.
- Variables físicas para representarlas: señales digit...
- Señales digitales. Toma valores discretos.
- Ejemplo: interruptor de la luz, bombilla

SISTEMA NUMÉRICO ~~POSICIONAL~~ BINARIO. CONVERSIÓN ENTRE SISTEMA

SISTEMA NUMÉRICO POSICIONAL: el valor del dígito depende de su posición en el número.

• SISTEMA DECIMAL

Columna 1000's	Columna 100's	Columna 10's	Columna 1's	BASE SISTEMA
5	3	7	4	10

• SISTEMA BINARIO

Columna 8's	Columna 4's	Columna 2's	Columna 1's	BASE SISTEMA
1	1	0	1	2

• Decimal ⟶ Binario

37 | 2
1   18 | 2
     0   9 | 2
          1   4 | 2
               0   2 | 2
                    0   1

$37_{(10)} = 100101_{(2)}$

Binario ⟶ Decimal

1 0 1 1

$1 \times 2^0 = 1$

$1 \times 2^1 = 2$

$0 \times 2^2 = 0$

$1 \times 2^3 = 8$

$1 + 2 + 8 = 11$

$1011_{(2)} = 11_{(10)}$

Ejemplos de conversión

Binario → Decimal el $10101_{(2)}$

$1\ 0\ 1\ 0\ 1$

$1 \times 2^0 = 1$

$1 \times 2^2 = 4$

$1 \times 2^4 = 16$

$\left.\right\} 1 + 4 + 16 = 21$

Decimal → Binario el $47_{(10)}$

47 | 2
1 23 | 2
1 11 | 2
1 5 | 2
1 2 | 2
0 1

101111

RANGO DE REPRESENTACIÓN DEL SISTEMA BINARIO

- Número n dígitos decimales $\{0, 9\}$, se representan 10^n números diferentes en el rango $[0, 10^n - 1]$

  Ejemplo → $n = 3$; se pueden representar $10^3 = 1000$ números $[0, 999]$

- Número n dígitos binarios $\{0, 1\}$, se representan 2^n números diferentes en el rango $[0, 2^n - 1]$

  Ejemplo → $n = 3$; se pueden representar $2^3 = 8$ números $[0, 7]$

Binario → hexadecimal

$$\underbrace{001}_{1}\ \underbrace{1010}_{A}\ \underbrace{1011}_{B}{}_{(2)} \longrightarrow 1AB_{(16)}$$

Hexadecimal → Binario

$$7\ F\ 2_{(16)} \longrightarrow 011111110010_{(2)}$$

$$\underbrace{111}\ \underbrace{1111}\ \underbrace{0010}$$

Hexadecimal → Decimal

$$7F2_{(16)} \longrightarrow$$

$2 \cdot 16^0$

$15 \cdot 16^1$

$7 \cdot 16^2$

Decimal → Hexadecimal

- dividiendo entre 16

Ejemplos de conversión

Hexadecimal → binario el $4AF_{(16)}$

$$\underbrace{0100}_{4}\ \underbrace{1010}_{A}\ \underbrace{1111}_{F}$$

Hexadecimal → decimal $4AF_{(16)}$; también $0 \times 4AF$

$$4 \cdot 16^2 + 10 \cdot 16 + 15 \cdot 1$$

* Programando no se puede introducir un número en base 16 empezando por una letra. ~~FA72~~ 0FA72 ✓

- BITS

$\underbrace{1}_{}$ 0 0 1 1 0 1 $\underbrace{1}_{}$

bit más significativo (msb) — bit menos significativo (lsb)

- BYTES y NIBBLES

1 Byte = 8 bits

1 0 0 1 1 0 1 1

1 Nibble = 4 bits

- BYTES

$\underbrace{\text{FC}}$ A5 C0 $\underbrace{\text{8D}}$

bit más significativo (msb) — bit menos significativo (lsb)

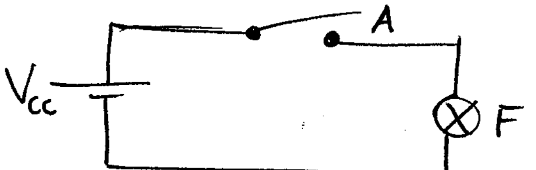
ÁLGEBRA DE BOOLE: herramienta matemática utilizada para el análisis y la síntesis de los sistemas digitales binarios.

VARIABLE BOOLEANA: es una señal digital que en un instante determinado sólo puede tomar uno de los dos valores. Los valores a tomar son mutuamente excluyentes.

- Se representan como: 0 y 1; OFF y ON; etc.

Una variable lógica solo puede tomar dos valores (y no puede tener los dos al mismo tiempo ⟹ son excluyentes). Se pueden expresar mediante una sentencia lógica declarativa.

▶ VARIABLES LÓGICAS Y CIRCUITOS ELÉCTRICOS



- Estado del interruptor A:
  - Abierto (0)
  - Cerrado (1)
- Estado de la bombilla F:
  - apagado (0)
  - encendida (1)

El estado de la variable lógica bombilla es función del estado de la variable lógica interruptor

FUNCIÓN LÓGICA: circuito que acepta valores lógicos a la entrada y produce un valor lógico a la salida.

TABLA DE VERDAD: describe el funcionamiento de las funciones lógicas.

- Especifica la salida de la puerta o función lógica para todas las posibles combinaciones de entradas.
- Son representaciones gráficas de todos los casos que se pueden dar en una relación algebraica y de todos sus posibles resultados.

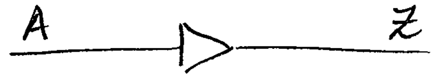
PUERTAS LÓGICAS: Implementan a las funciones lógicas más elementales.

• EL AMPLIFICADOR (o BUFFER)

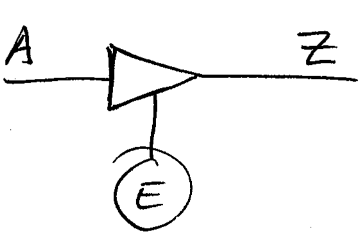
- puerta ~~lógica~~ lógica más sencilla
- una entrada (A) y una salida (Z)
- tabla de verdad

A	Z
1	1
0	0

- ecuación lógica
- representación gráfica:



BUFFER DE TRES ESTADOS



- cuando E(enable) = 1 el buffer de 3 estados funciona como una resistencia bajísima

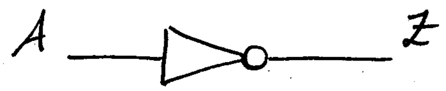
- cuando E = 0 el buffer de 3 estados funciona como una resistencia infinita

- LA PUERTA NOT o INVERSOR

- Una entrada (A) y una salida (Z).
- Tabla de verdad

A	Z
1	0
0	1

- Representación gráfica:



- Ecuación lógica: $Z = \overline{A}$

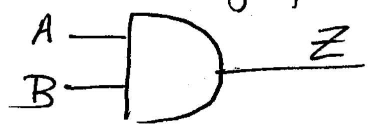
- LA PUERTA AND

La variable lógica bombilla (Z) estará activa cuando las variables independientes interruptor A y interruptor B están activos.

$Z = 1$ sólo si las dos entradas están en 1.

A	B	Z
0	0	0
0	1	0
1	0	0
1	1	1

- Ecuación lógica $Z = A.B$
- Representación gráfica



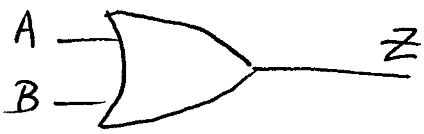
- LA PUERTA OR

La variable lógica bombilla (Z) estará activa cuando ~~las~~ una de las variables independientes interruptor A o B (o ambos) están activos.

$Z = 1$ si A, B o ambos están en 1.

A	B	Z
0	0	0
0	1	1
1	0	1
1	1	1

- Ecuación lógica $Z = A + B$
- Representación gráfica:

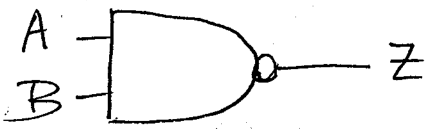


- **LA PUERTA NAND**

$Z=1$ si, al menos, una de las dos entradas vale 0.

A	B	Z
0	0	1
0	1	1
1	0	1
1	1	0

- Ecuación lógica $Z = \overline{A \cdot B}$
- Representación gráfica



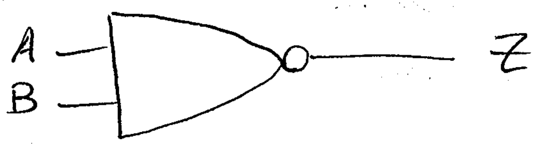
- **LA PUERTA NOR**

$Z=1$ si ambas entradas valen 0.

$Z=0$ si, al menos, una de las dos entradas vale 1.

A	B	Z
0	0	1
0	1	0
1	0	0
1	1	0

- Ecuación lógica $Z = \overline{A+B}$
- Representación gráfica

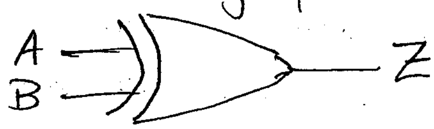


- **LA PUERTA XOR (OR-Exclusiva)**

$Z=1$ si hay un número impar de entradas en estado 1.

A	B	Z
0	0	0
0	1	1
1	0	1
1	1	0

- Ecuación lógica $Z = A \oplus B$
- Representación gráfica



CIRCUITOS LÓGICOS

Las combinaciones de diferentes valores lógicos a la entrada hacen que aparezcan distintos valores lógicos a la salida ⟹ CIRCUITO LÓGICO

Un circuito lógico se compone de:

- entradas
- salidas
- especificación funcional
- especificación temporal

Cualquier función lógica se puede formar mediante (únicamente) puertas AND, OR y NOT.

También podemos usar solo puertas NAND para cualquier circuito.

Lógica combinacional: el estado de las ~~entradas~~ salidas depende únicamente del estado de las entradas. Sistema SIN MEMORIA

Lógica secuencial: el estado de la salida también depende del estado anterior del sistema. El circuito TIENE MEMORIA

FUNCIONES LÓGICAS

Función lógica: Expresión Booleana que relaciona variables lógicas directas o complementadas por medio de operadores AND y OR.

SUMA DE MINTERMS (CIRCUITOS SOP (Sum of products)): ~~Suma~~
uma de productos de todas las variables o sus conjugadas.

PRODUCTO DE MAXTERMS (CIRCUITOS POS (Product of sums):
Producto de sumas de todas las variables o sus conjugadas.

A	B	Y	MINTERM
0	0	0	$\bar{A}.\bar{B}$
0	1	1	$\bar{A}.B$
1	0	0	$A.\bar{B}$
1	1	1	$A.B$

SUMA DE PRODUCTOS (MINTERMS)

$$Y = F(A,B) = \bar{A}.B + A.B$$

$$Y = F(A,B) = \sum m(1,3) = \bar{A}.B + A.B$$

- Todas las ecuaciones Booleanas pueden ser descritas como suma de ~~productos~~ minterms (SOP).
- Cada fila en una tabla de verdad es un minterm.
- Un minterm es un producto (AND) de las variables y sus complementos.
- Cada minterm es VERDADERO ("1") para esa fila (y solo para esa fila).
- La función se construye por la suma (OR) de los minterms para los cuales la salida es VERDADERA.
- Se trata por tanto de una suma (OR) de productos (AND).

A	B	Y	MAXTERM
0	0	0	$A+B$
0	1	1	$A+\bar{B}$
1	0	0	$\bar{A}+B$
1	1	1	$\bar{A}+\bar{B}$

PRODUCTO DE SUMAS (MAXTERMS)

$$Y = F(A,B) = (A+B)\,(\bar{A}+B)$$

- Todas las ecuaciones Booleanas pueden ser descritas como producto de maxterms (POS).
- Cada fila en una tabla de verdad es un maxterm.
- Un maxterm es una suma (OR) de las variables y sus complementos.
- Cada maxterm es FALSO ("0") para esa fila (y sólo para esa fila).
- La función se construye por el producto (AND) de los maxterms para los cuales la salida es FALSA.
- Se trata por tanto de un producto (AND) de sumas (OR).

• Ejemplo: Desarrollo canónico de una función a partir de su tabla de verdad

Nº	A	B	C	F(A,B,C)
0	0	0	0	1
1	0	0	1	0
2	0	1	0	1
3	0	1	1	1
4	1	0	0	0
5	1	0	1	0
6	1	1	0	1
7	1	1	1	1

MINTERMS:

$$f(A,B,C) = \sum m(0,2,3,6,7) =$$

$$= \bar{A}.\bar{B}.\bar{C} + \bar{A}.B.\bar{C} + \bar{A}.B.C + A.B.\bar{C} + A.B.C$$

MAXTERMS:

$$f(A,B,C) = \prod M(1,4,5) =$$

$$= (A+B+\bar{C})(\bar{A}+B+C)(\bar{A}+B+\bar{C})$$