Arquitectura de ordenadores

3-1. Principios Básicos I

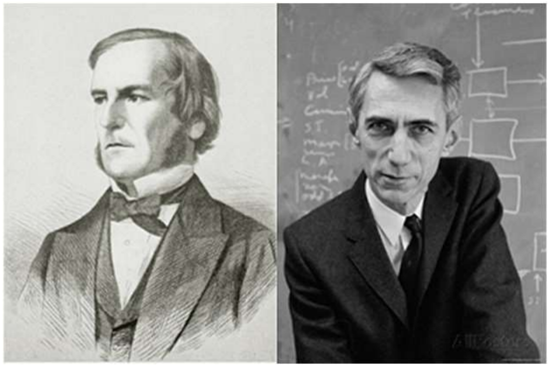
Ignacio Calles González ignacio.gonzalez@ext.live.u-tad.com
Tiago Manuel Louro Machado de Simas tiago.louro@u-tad.com
Francisco Javier García Algarra javier.algarra@u-tad.com
Carlos M. Vallez Fernández carlos.vallez@u-tad.com

2023-2024

Índice

1. Introducción

- Los ordenadores básicamente lo que entiende son 0’s y 1’s.
- Necesidad “reglas” que permitan operar con 0’s y 1’s.
- En el s.XIX un matemático llamado George Boole desarrolló una teoría matemática:
  - Principal característica: manejar variables con solo dos valores (Verdadero o Falso), o por extensión, 1 y 0.
  - Se conoce como Algebra de Boole.
  - Es la herramienta matemática que se ha empleado en el diseño de sistemas digitales (se debe a Shannon), como puede ser un ordenador.



Boole y Shannon

Índice

2. Álgebra de Boole - Definición

- Un Álgebra de Boole (B) es un conjunto finito con dos operaciones binarias: Suma (+) y multiplicación (.) que satisface las siguientes reglas:

  - Para toda variable booleana x (**que puede tomar dos valores**) se cumple que: x=1 o x=0
  - Existe la **operación complemento** o negación (NOT) tal que_

    NOT(0) =1 NOT(1)=0
  - Existe la **operación producto lógico** o booleana (AND):

    0 . 0 = 0 0 . 1 = 0 1 . 0 = 0 1 . 1= 1

    Nota : se puede sustituir el operador . por la función lógica AND para un mejor entendimiento.
  - Existe la **operación suma lógica** o boolena (OR):

    0 + 0 = 0 0 + 1 = 1 1 + 0 = 1 1 + 1 = 1

    Nota : se puede sustituir el operador + por la función lógica OR para un mejor entendimiento.
  - Por convenio la operación AND tiene una **precedencia** superior a la función OR

2.1 Postulados del Álgebra de Boole (I)

- **Postulado 1: + y . son leyes de composición interna**

  Ello quiere decir que para todo a y b que pertenecen al conjunto B, el resultado de a+b o de a.b también pertenece al conjunto B.

  Matemáticamente se expresa como:

  $$\forall a, b \in B, \qquad a + b \in B \; y \; a.b \in B$$

- **Postulado 2: Elemento neutro**

  Todas las operaciones tienen un elemento neutro

  $$\forall a \in B, \qquad a + 0 = a \; y \; a.1 = a$$

2.1 Postulados del Álgebra de Boole (II)

- **Postulado 3: Elemento inverso**

  Existe un elemento contrario o inverso, para todo elemento, que pertenece a B

  $\forall a \in B$, $\exists$ $\mathrm{NOT}(a) \in B$ | a + NOT(A) =1 y a . NOT(a) =0

  Nota : NOT(a) también se representa como: ¬a o $\bar{a}$

- **Postulado 4 Propiedad conmutativa**

  a+b= b+a y a.b = b.a

- **Postulado 5 Propiedad distributiva**

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

  a + (b.c)= (a+b) –(a+c)

2.3 Propiedades / Teoremas

Algunas de estas propiedades ya se han nombrado como postulados en el apartado anterior pero los recogemos nuevamente para tenerlas unificadas:

- **Elemento Inverso**:
- **Idempotencia**: $\bar{1} = 0 \ y \ \bar{0} = 1$
- **Involución**: $a+a = a \ y \ a.a = a$
- **Asociativa:** $NOT(NOT(a)) = a$
- **Absorción** $a + (b+c) = (a+b) + c \quad y \ a . (b.c) = (a.b) .c$
- **"Sin nombre"** $a + (a.b) = a \ y \ a.(a+b) = a$
- **Teorema de Morgan** $a + \bar{a}.b = a + b \quad y \quad a.(\bar{a} + b) = a.b$
- **Teorema de Shannon** $\overline{a + b} = \bar{a}.\bar{b} \quad y \quad \overline{a.b} = \bar{a} + \bar{b}$

  (Morgan generalizado) $\overline{a_1 + a_2 + \dots + a_n} = \bar{a}_1 . \bar{a}_2 . \dots . \bar{a}_n$

  $\overline{a_1 . a_2 . \dots . a_n} = \bar{a}_1 + \bar{a}_2 + \dots + \bar{a}_n$

2.4 Representación de los sistemas digitales (I)

Todo sistema digital puede ser representado de diversas formas:

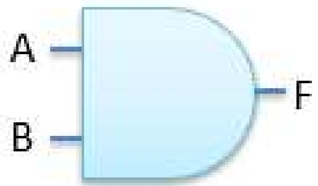
- **Fórmulas** Ej: f(x)= x.y
- **Tablas de verdad:** Muestra lo que ocurre a la salida del sistema digital en función de los valores de las variables de entrada.

x	y	f(x)= x.y
0	0	0
0	1	0
1	0	0
1	1	1

Cualquier función booleana puede ser definida de forma explícita con una tabla de verdad y cualquier tabla de verdad tiene su función booleana equivalente

- **Circuitos lógicos** creados a partir de puertas lógicas.

2.4 Representación de los sistemas digitales (II)

<table>
<tr><th>Nombre</th><th>Símbolo</th><th>Operación</th><th>Tabla de verdad</th></tr>
<tr><td>AND</td><td>
</td><td>F = A • B

F = AB

F = A AND B</td><td><table>
<tr><th>A</th><th>B</th><th>F</th></tr>
<tr><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td></tr>
</table></td></tr>
</table>

Índice

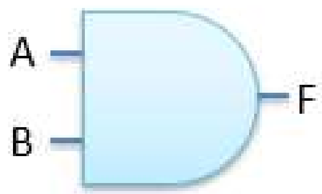
3. Puertas Lógicas

- Puerta lógica : circuito pequeño normalmente e integrado en un chip.
- Implementa una función lógica elemental. Es el elemento básico empleado en la construcción de circuitos digitales lógicos.
- Variables sus entradas (señales que entran al circuito) .
- Salida: el resultado de aplicar la función lógica que implementan sobre las entradas.
- Pueden tener una o varias entradas, pero solo una salida.
- La salida puede tomar le valor 0 o 1
- El circuito que forma una puerta lógica está formado por una serie de resistencias, diodos, transistores, etc.
- Una puerta lógica se define mediante su nombre, su símbolo, su notación algebraica y su tabla de verdad.

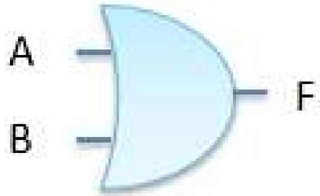
3.1 Puertas Lógicas Básicas (I)

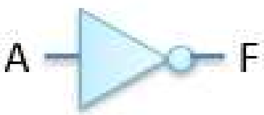
- Las puertas lógicas más usadas son las puertas NOT , OR y AND.
- En la siguiente diapositiva se muestra una tabla con dichas puertas lógicas mostrando su nombre, el símbolo más comúnmente empleado en la representación de circuitos, la notación algebraica o fórmula lógica (nótese que se emplean los símbolos de las operaciones del álgebra de Boole) y finalmente la tabla de verdad.

3.1 Puertas Lógicas Básicas (II)

<table>
<tr><th>Nombre</th><th>Símbolo</th><th>Operación</th><th>Tabla de verdad</th></tr>
<tr><td>AND</td><td>
</td><td>$F = A \cdot B$

F = AB

F = A AND B</td><td><table><tr><th>A</th><th>B</th><th>F</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table></td></tr>
<tr><td>OR</td><td>
</td><td>$F = A + B$

F = A OR B</td><td><table><tr><th>A</th><th>B</th><th>F</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table></td></tr>
<tr><td>NOT</td><td>
</td><td>$F = \bar{A}$

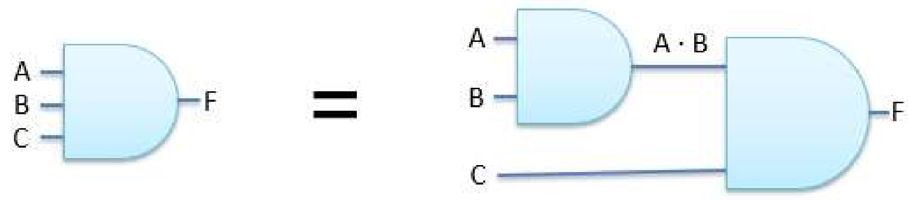
F = NOT(A)</td><td><table><tr><th>A</th><th>F</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table></td></tr>
</table>

3.1 Puertas Lógicas Básicas (III)

- Las puertas OR y AND podrían tener más de dos entradas.
- Las operaciones que realizan son las que como “operaciones del álgebra de Boole”.
- Por tanto las puertas lógicas tienen por tanto las mismas propiedades vistas del Algebra de Boole.
- Por ejemplo, si pensamos en la propiedad asociativa podemos expresar una puerta AND de tres entradas como un circuito formado por dos puertas AND de dos entradas.

Expresado en fórmula

$A \cdot B \cdot C = (A \cdot B) \cdot C$

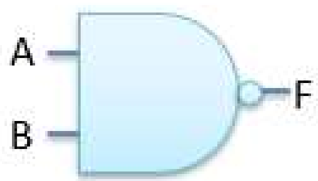


3.2 Puertas Lógicas No Básicas (I)

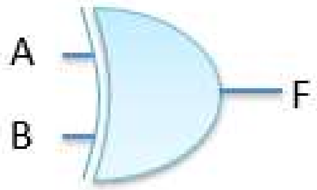
- Además de las puertas lógicas vistas en el apartado anterior existen las puertas :
  - NOR (que es la negación de la OR)
  - NAND (que es la negación de AND)
  - XOR (que es la función OR exclusivo el cual toma el valor 1 cuando las entradas son distintas).

Es conveniente fijarse que por ejemplo el símbolo del NAND es como el del AND + “la bolita” del NOT

3.2 Puertas Lógicas No Básicas (II)

<table>
<tr><th>Nombre</th><th>Símbolo</th><th>Operación</th><th>Tabla de verdad</th></tr>
<tr><td>NAND</td><td>
</td><td>$F = \overline{(A \cdot B)}$

F = A NAND B</td><td>
<table>
<tr><th>A</th><th>B</th><th>F</th></tr>
<tr><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>0</td></tr>
</table></td></tr>
<tr><td>NOR</td><td>
</td><td>$F = \overline{(A + B)}$

F = A NOR B</td><td>
<table>
<tr><th>A</th><th>B</th><th>F</th></tr>
<tr><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>0</td></tr>
</table></td></tr>
<tr><td>XOR</td><td>
</td><td>$F = (A \oplus B)$

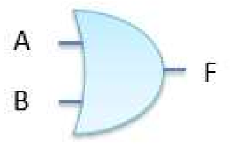
$F = A \cdot \overline{B} + \overline{A} \cdot B$

F = A XOR B</td><td>
<table>
<tr><th>A</th><th>B</th><th>F</th></tr>
<tr><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>0</td></tr>
</table></td></tr>
</table>

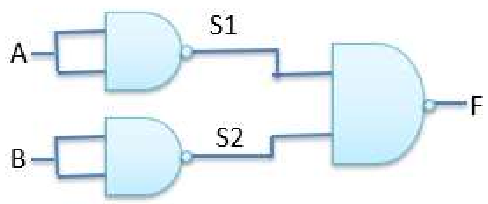
3.3 Agrupación de Puertas Lógicas (I)

- La agrupación y el encadenado de puertas lógicas da lugar a la generación de los circuitos electrónicos complejos que conforman un computador.

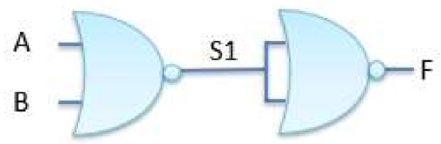
- Es posible implementar el funcionamiento una puerta lógica empleando otras.
- Por ejemplo: vamos a ver cómo podríamos implementar el funcionamiento de una puerta OR sólo con puertas NAND y por otro lado sólo con puertas NOR



A	B	F
0	0	0
0	1	1
1	0	1
1	1	1



A	B	S1= A NAND A	S2=B NAND B	F= S1 NAND S2
0	0	1	1	0
0	1	1	0	1
1	0	0	1	1
1	1	0	0	1



A	B	S1= A NOR B	F=S1 NOR S1
0	0	1	0
0	1	0	1
1	0	0	1
1	1	0	1

3.3 Agrupación de Puertas Lógicas (II)

- Algunas de estas puertas, al agruparse, componen **un conjunto de puertas funcionalmente completo**.
- Con las puertas de un conjunto funcionalmente completo se puede implementar el funcionamiento de cualquier otra puerta lógica.
- Ejemplo:
  - AND, OR, NOT: Es el conjunto funcionalmente completo básico, ya que contiene las tres funciones lógicas básicas.
  - AND, NOT: Es un conjunto funcionalmente completo, ya que con ellas se puede implementar la puerta lógica básica restante (OR) .

    F = A OR B = NOT ((NOT A) AND (NOT B))

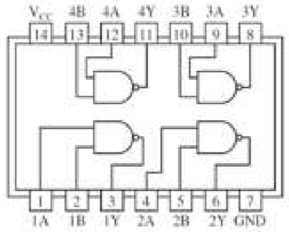
3.3 Agrupación de Puertas Lógicas (III)

- Si las puertas AND y NOT forman un conjunto funcionalmente completo como acabamos de ver, entonces :
  - La puerta NAND sola: Es un conjunto funcionalmente completo, ya que con ella se pueden implementar todas las puertas lógicas básicas (AND, OR, NOT).
  - Y la puerta NOR: Es un conjunto funcionalmente completo, ya que con ella se pueden implementar todas las puertas lógicas básicas (AND, OR, NOT).

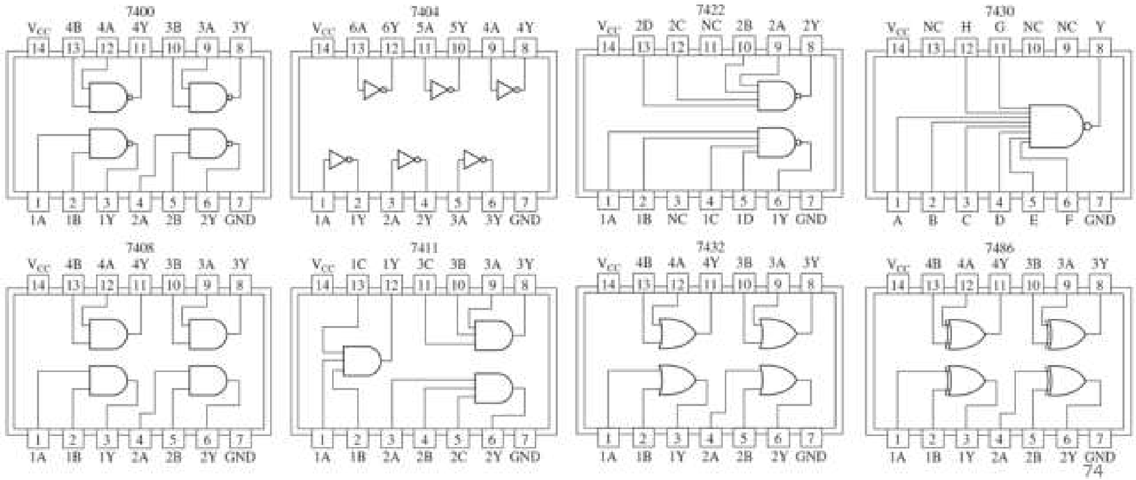
  - Estas dos puertas lógicas (NAND y NOR) son las que más frecuentemente se utilizan para construir circuitos electrónicos más complejos.

3.3 Agrupación de Puertas Lógicas (IV)

- En el mercado las puertas lógicas no se venden por separado. Las puertas se agrupan en conjuntos mixtos, específicamente diseñados y minimizados para implementar el mayor número posible de funciones lógicas booleanas.
- Esos conjuntos de puertas se implementan dentro de microchips. Ej: 7400



Chips de escala SSI: hasta 10 puertas por chip.



4 Ejercicios varios para practicar (I)

- 1 Cerrar todos los ordenadores. Sin apuntes se pide a los alumnos que en una hoja de papel creen una tabla en la cual pongan para cada puerta lógica básica y no básica:

Nombre	Básica ó no	Símbolo	Operación de las formas que conozco	Tabla de verdad

Índice

4 Ejercicios varios para practicar (II)

- 2 Se pide implementar la funcionalidad de la puerta AND de las siguientes formas y explicar por qué se puede hacer o por qué no se puede:
  - A) Con puertas OR
  - B) Con puertas OR y NOT
  - C) Con puertas NOR
  - D) Con puertas NAND

4 Ejercicios varios para practicar (II)

- 3 Se pide implementar la funcionalidad de la puerta XOR Con puertas Lógicas AND, OR y NOT. ¿y si no me dejaran usar o la puerta AND o la OR?. Se pide demostrar mediante una tabla de verdad que el conjunto de puertas propuesto es equivalente a la puerta XOR

- 4 Realizar la misma tarea pero con una puerta XNOR

4 Ejercicios varios para practicar (III)

- 5 Crear la tabla de verdad de la función lógica $F= b + c + \bar{a} \cdot \bar{b} \cdot \bar{c}$

- 6 Dibujar el circuito que representa las siguientes fórmulas con puertas lógicas:

$$F= b + c + \bar{a} \cdot \bar{b} \cdot \bar{c}$$

$F2= b + c \overline{+} a$ ¿Ves alguna similitud con la función anterior? ¿ y si haces su tabla de verdad?

$F3= a + b \cdot c$

4 Ejercicios varios para practicar (III)

- 7 Dada la siguiente fórmula lógica, aplicar las propiedades algebraicas de Boole vistas para simplificarlas y comprobar mediante las tablas de verdad que la fórmula original y la simplificada son idénticas

$$F = \bar{a} + \bar{a} \cdot b + \bar{a} \cdot \bar{b}$$