



UD 2 Representación de la información

Desarrollo de Aplicaciones Multiplataforma

Contenidos

1. Introducción
2. Representación numérica
3. Conversión entre bases
4. Operaciones lógicas
5. Representación alfanumérica
6. Múltiplos

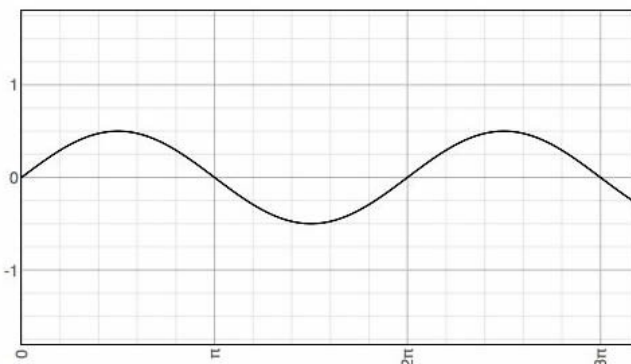
1. Introducción

1. Introducción

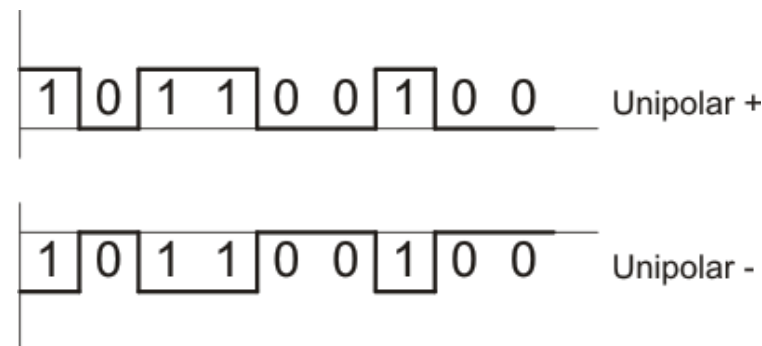
- La transmisión de información entre el ser humano y los ordenadores puede hacerse de muchas formas:
 - Mediante **caracteres alfanuméricos** (letras y números). Por ejemplo, los introducidos al ordenador mediante un teclado.
 - Mediante **sonidos**: como los introducidos al ordenador a través de un micrófono, o que salen del ordenador por los altavoces.
 - Mediante **vídeos**: como las imágenes obtenidas a través de una cámara de vídeo.
 - Mediante **gráficos e imágenes**: por ejemplo, una imagen introducida por un escáner o fotografías descargadas de una cámara de fotos digital.
- En general, cualquier tipo de dato enviado por un periférico capaz de tomar datos y enviarlos al ordenador, o a la inversa.

1. Introducción

- Las señales que se transmiten dentro de un ordenador son digitales. Una señal digital toma una serie de valores discretos, no continuos.



Señal analógica

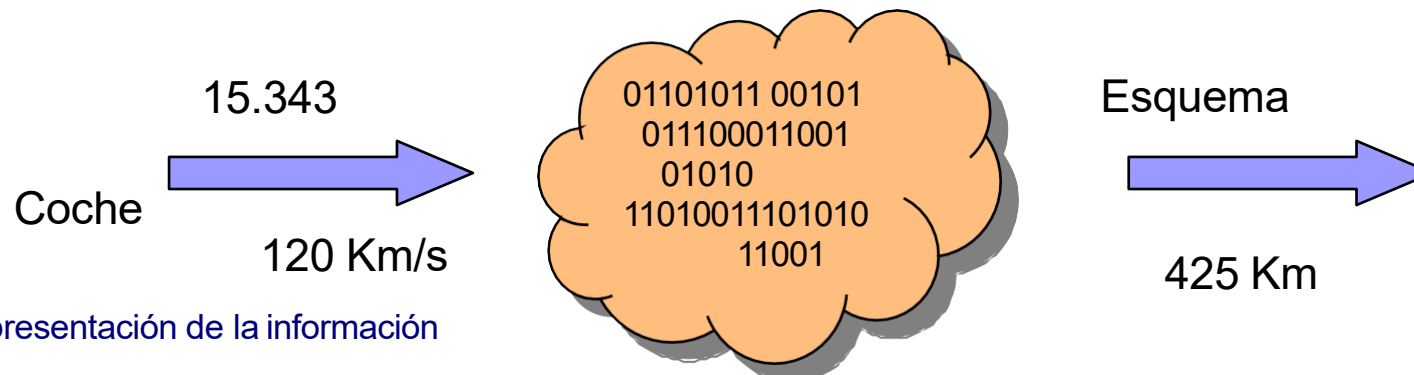


Señal digital

- La representación interna de la información en los ordenadores ha de darse en forma de **impulsos eléctricos**.
- Esto se efectúa empleando **biestables**, circuitos con dos posibles estados: activado - desactivado, encendido - apagado, abierto - cerrado, tensión - no tensión. Es decir, hay impulso o no lo hay.

1. Introducción

- Por eso, se tendrá que **codificar la información utilizando un código con dos únicos símbolos** que representen los dos estados, por ejemplo, el **1** para indicar que hay impulso y el **0** para indicar que no lo hay.
- Todo se transcribirá a combinaciones de ceros y unos, aunque, en realidad, no existen ni los ceros ni los unos, es otra manera de representación para que los humanos podamos entenderlo y trabajar con él, con el **código binario**.
- En las entradas/salidas (E/S) se efectúa la transformación.



1. Introducción

- Para proceder a la comunicación de los datos es necesario cambiar la forma en que estos se representan. Por lo tanto, los datos deben ser traducidos o **codificados**.
- **Representación externa:**
 - La utilizada por las personas: sistema decimal, caracteres alfanuméricos, etc.
- **Representación interna:**
 - La utilizada por el ordenador: sistema binario, códigos ASCII para los caracteres, etc.
- **Codificación** es una transformación que representa los elementos de un conjunto mediante los de otro, de forma tal que a cada elemento del primer conjunto le corresponda un elemento distinto del segundo.

2. Representación numérica

2. Representación numérica

Sistema de numeración:

- Definición: Conjunto de símbolos o dígitos utilizados para la representación de cantidades, así como las reglas que rigen dicha representación. Ejemplos: decimal, binario, octal.
- El número de símbolos utilizado para la representación viene determinado por la **base**.
 - Decimal \rightarrow base = 10
 - Binario \rightarrow base = 2
- Sistema de numeración es **posicional** \rightarrow Los dígitos tienen un significado distinto en función de la posición que ocupan.

2. Representación numérica

- El Teorema fundamental de la numeración relaciona una cantidad expresada en cualquier sistema de numeración, con la misma cantidad expresada en el sistema decimal.
- El valor de un número N en un sistema posicional de base b:
 - Se expresa como una secuencia de dígitos de la base.
 - Se calcula de la siguiente forma:

$$N = \sum_{i=0}^n a_i b^i = a_n b^n + \dots + a_1 b^1 + a_0 b^0$$

- Si se extiende a números reales:


$$R = \sum_{i=-p}^n a_i b^i = a_n b^n + \dots + a_1 b^1 + a_0 b^0 + a_{-1} b^{-1} + \dots + a_{-p} b^{-p}$$

2. Representación numérica

■ Sistema decimal:

- Base: 10.
- Dígitos: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.
- La posición del dígito representa una potencia de 10.
- Ejemplos:

$$123_{10} = 1 \times 10^2 + 2 \times 10^1 + 3 \times 10^0 =$$

Base 

$$100 + 20 + 3$$

$$45,612_{10} = 4 \times 10^1 + 5 \times 10^0 + 6 \times 10^{-1} + 1 \times 10^{-2} + 2 \times 10^{-3} =$$
$$40 + 5 + 0,6 + 0,01 + 0,002$$

2. Representación numérica

■ Sistema binario:

- Base: 2.
- Dígitos: 0, 1 (denominados **bits**).
- Una cantidad N se representa mediante una secuencia de bits:

de bits: $N = 1010_2$

← Base

↖ MSB (Most Significant Bit)

↖ LSB (Least Significant Bit)

- La posición del dígito representa una potencia de 2.
- Conversión de binario a decimal:

$$1101_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 =$$

$$8 + 4 + 0 + 1 = 13_{10}$$

$$101,11_2 = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2} =$$

$$4 + 0 + 1 + 0,5 + 0,25 = 5,75_{10}$$

2. Representación numérica

■ Ejercicio: Paso de binario a decimal

- ☐ $011_2 \rightarrow X_{10}$
- ☐ $10101_2 \rightarrow X_{10}$
- ☐ $1100_2 \rightarrow X_{10}$
- ☐ $11101_2 \rightarrow X_{10}$
- ☐ $100_2 \rightarrow X_{10}$
- ☐ $1101_2 \rightarrow X_{10}$

2. Representación numérica

- Además del binario se utilizan los sistemas octal y hexadecimal:
 - Por su facilidad de conversión a/desde binario (ya que sus bases son potencia de 2).
 - Porque permiten representar largas secuencias de bits con menos dígitos.

2. Representación numérica

■ Sistema octal:

- Base 8 (2^3).
- Dígitos: 0, 1, 2, 3, 4, 5, 6 y 7.
- Una cantidad N se representa mediante una secuencia de dígitos del 0 al 7:

$$N = 2471_8$$

- La posición del dígito representa una potencia de 8.
- Conversión de octal a decimal:

$$\begin{aligned} 746_8 &= 7 \times 8^2 + 4 \times 8^1 + 6 \times 8^0 = \\ &448 + 32 + 6 = 486_{10} \end{aligned}$$

2. Representación numérica

■ Sistema hexadecimal:

- Base 16 (2^4).
- Dígitos: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E y F.
- Una cantidad N se representa mediante una secuencia de dígitos del 0 a la F:

$$N = A3FC_{16}$$

- La posición del dígito representa una potencia de 16.
- Conversión de hexadecimal a decimal:

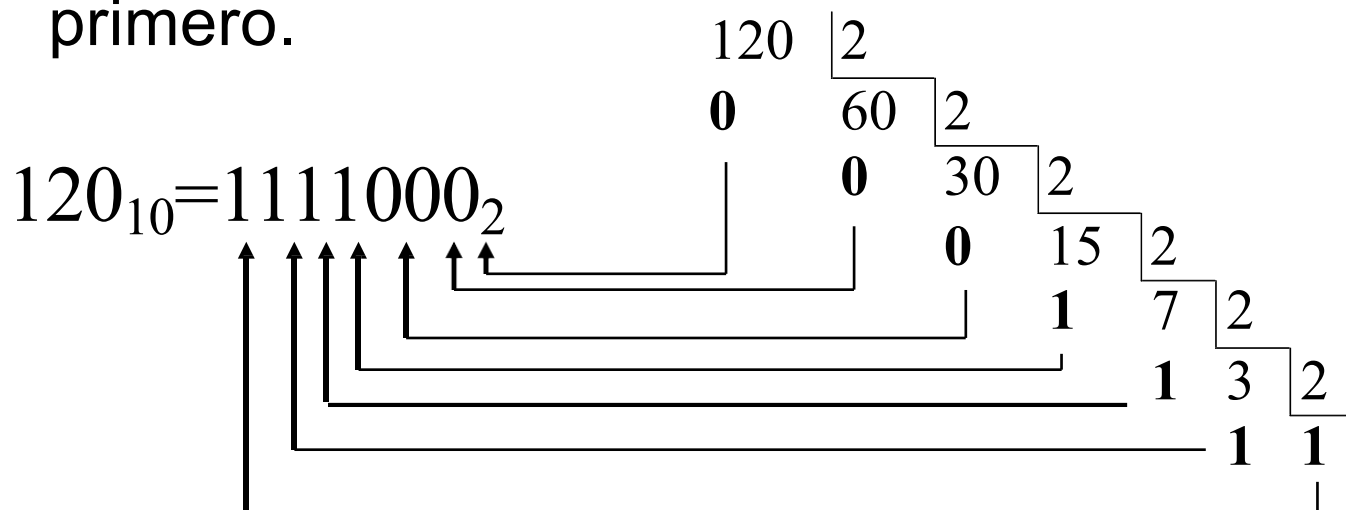
$$\begin{aligned} B23A_{16} &= B \times 16^3 + 2 \times 16^2 + 3 \times 16^1 + A \times 16^0 = \\ &= 11 \times 16^3 + 2 \times 16^2 + 3 \times 16^1 + 10 \times 16^0 = \\ &= 45056 + 512 + 48 + 10 = 45626_{10} \end{aligned}$$

3. Conversión entre bases

3. Conversión entre bases

■ Conversión de decimal a binario:

- Se divide la cantidad entre la nueva base (2).
- Mientras el cociente sea > 2 , se sigue dividiendo el cociente.
- El número en binario se obtiene concatenando el último cociente y los restos de las divisiones, empezando por el último resto y terminando por el primero.



3. Conversión entre bases

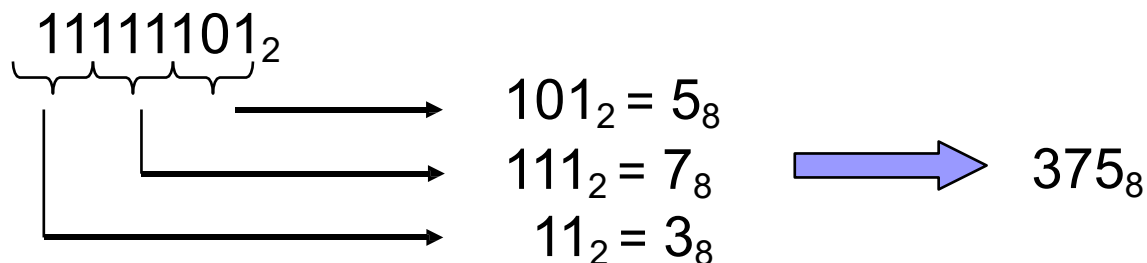
- Tabla de conversión:

Binario	Decimal
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

3. Conversión entre bases

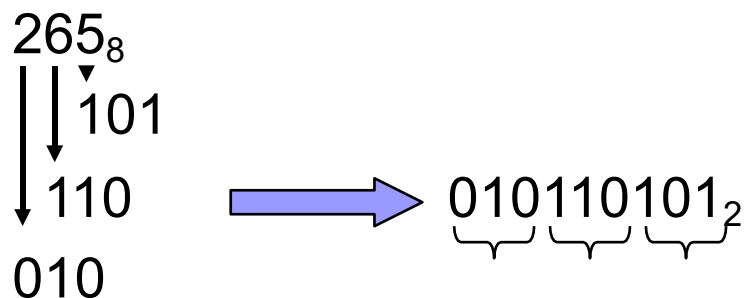
■ Conversión de binario a octal:

- Se calcula agrupando de tres en tres los bits empezando por la derecha y sustituyendo cada grupo de tres por su equivalente en octal:



■ Conversión de octal a binario:

- Se calcula sustituyendo cada dígito octal por su representación binaria utilizando tres dígitos binarios:



3. Conversión entre bases

- Tabla de conversión:

Binario	Decimal	Octal
000	0	0
001	1	1
010	2	2
011	3	3
100	4	4
101	5	5
110	6	6
111	7	7

3. Conversión entre bases

■ Conversión de binario a hexadecimal:

- Se calcula agrupando de cuatro en cuatro los bits empezando por la derecha y sustituyendo cada grupo de cuatro por su equivalente en hexadecimal:

$$\begin{array}{rcl}
 101111011_2 & & \\
 \underbrace{}_{1011_2} & \longrightarrow & 1011_2 = B_{16} \\
 \underbrace{}_{0111_2} & \longrightarrow & 0111_2 = 7_{16} \\
 \underbrace{}_{1_2} & \longrightarrow & 1_2 = 1_{16}
 \end{array}
 \quad \longrightarrow \quad 17B_{16}$$

■ Conversión de hexadecimal a binario:

- Se calcula sustituyendo cada dígito hexadecimal por su representación binaria utilizando cuatro dígitos binarios:

$$\begin{array}{rcl}
 FA1_{16} & & \\
 \downarrow & \downarrow & \downarrow \\
 1111 & 1010 & 0001
 \end{array}
 \quad \longrightarrow \quad
 \underbrace{1111}_F \underbrace{1010}_A \underbrace{0000}_0 \underbrace{0001}_1_2$$

3. Conversión entre bases

■ Tabla de conversión:

Binario	Decimal	Octal	Hexadec.
0000	0	0	0
0001	1	1	1
0010	2	2	2
0011	3	3	3
0100	4	4	4
0101	5	5	5
0110	6	6	6
0111	7	7	7

Binario	Decimal	Octal	Hexadec.
1000	8	10	8
1001	9	11	9
1010	10	12	A
1011	11	13	B
1100	12	14	C
1101	13	15	D
1110	14	16	E
1111	15	17	F

3. Conversión entre bases

- Conversión de octal a hexadecimal:

- ☐ Se realiza un **paso intermedio**, primero se pasa de octal a binario y después de binario a hexadecimal.

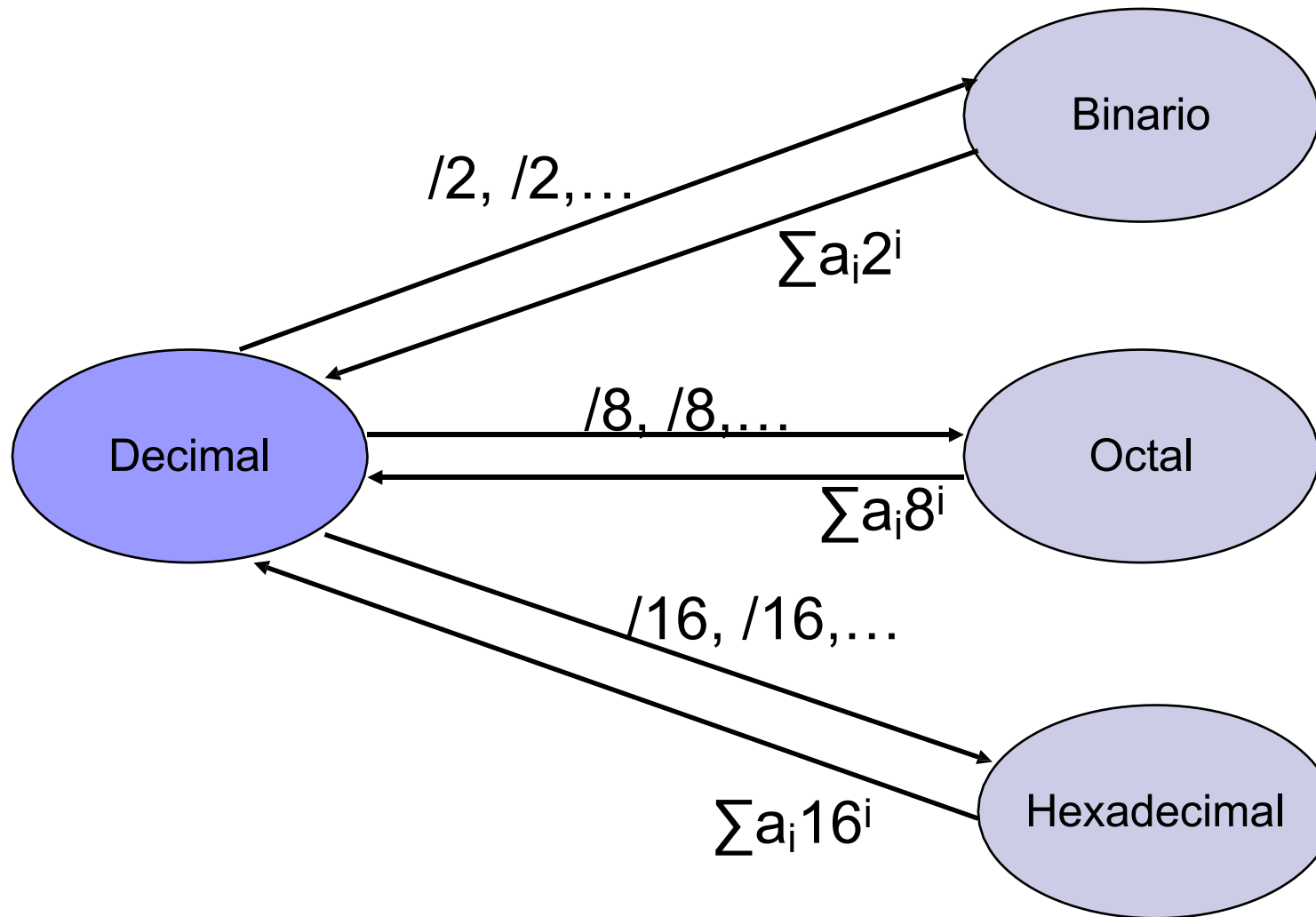
- Conversión de hexadecimal a octal:

- ☐ Se realiza un **paso intermedio**, primero se pasa de hexadecimal a binario y después de binario a octal.

- Conversión de decimal a binario usando posiciones:

- ☐ 128 64 32 16 8 4 2 1
- ☐ 1 1 1 1 1 1 1 1

Cambio de base. Resumen



Tablas de conversiones útiles

2^0	00000001	1
2^1	00000010	2
2^2	00000100	4
2^3	00001000	8
2^4	00010000	16
2^5	00100000	32
2^6	01000000	64
2^7	10000000	128

10000000	128
11000000	192
11100000	224
11110000	240
11111000	248
11111100	252
11111110	254
11111111	255

Tablas de conversiones útiles

Hacemos la práctica 2-1



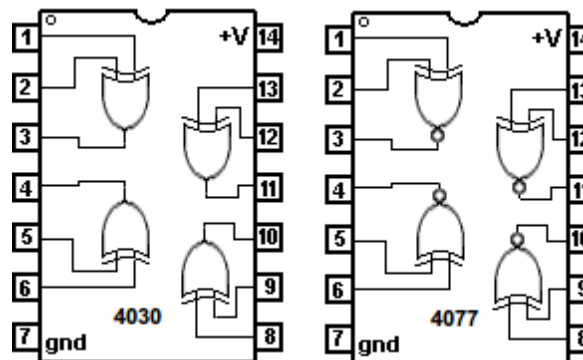
4. Operaciones lógicas

4. Operaciones lógicas

- *¿Una máquina puede pensar?*
- *¡Naturalmente! ¡Usted y yo somos máquinas y vaya si pensamos!*
- Vamos a aprender en qué se basa la parte “pensante” de un procesador, la parte que ejecuta la lógica y toma decisiones.
- La lógica se basa en una parte de las matemáticas y de la informática, llamada **álgebra booleana** o **álgebra de Boole**. **George Boole** (1815-1864) fue un matemático inglés que, basándose en lo que ya se conocía como la lógica y la toma de decisiones, creó una manera esquemática de realizar operaciones no con números, sino con lógica.

4. Operaciones lógicas

- Su creación, fue la base para que **Claude Shannon** (1916-2001), un ingeniero y matemático de los laboratorios Bell, en EEUU, gracias a su interés por la lógica y por los circuitos telefónicos, se diera cuenta de que éstos podrían servir para hacer cálculos lógicos.
- Así nacieron los **circuitos lógicos**, uno de los mayores avances en la tecnología en el último siglo. Shannon no dejó de desarrollar la aplicación de la lógica en numerosos ámbitos, como por ejemplo, la lógica necesaria para que una máquina pudiera jugar al ajedrez.



4. Operaciones lógicas

- Las operaciones lógicas vienen representadas por lo que se conoce como **función lógica**.
- Una **función lógica** no es más que un conjunto de variables lógicas A, B, C, etc. relacionadas por los símbolos de las operaciones permitidas, siendo las operaciones básicas: suma, producto y negación.

Por ejemplo: $F = A + B + \bar{A} \cdot C$

- Una **variable lógica** es aquella que puede tomar únicamente dos valores: 0 y 1. En informática se entienden estos valores como *false/true*, falso/verdadero, no/sí,...
- Para representar una variable lógica vamos a utilizar las letras del abecedario en mayúsculas: A, B, C,...

4. Operaciones lógicas

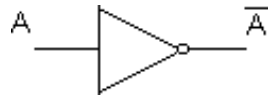
- Una función lógica acepta sólo dos posibles entradas (0 y 1) y produce un solo valor (salida).
- Para saber cuáles son las salidas de una función, se inventó lo que se conoce como la **tabla de verdad**, que es la representación de todas las combinaciones posibles de las variables lógicas, y las salidas correspondientes de la función.

A	B	$F = A \cdot B + \bar{B}$
0	0	1
0	1	0
1	0	1
1	1	1

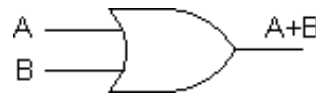
4. Operaciones lógicas

- Las operaciones lógicas básicas son:

- ☐ NOT



- ☐ OR



- ☐ AND



- Gracias a Shannon, se crearon los circuitos lógicos que pueden representar los dos valores, 0 y 1.
- Una **puerta lógica** es un circuito electrónico que tiene el mismo comportamiento que una función lógica.

4. Operaciones lógicas

■ AND:



- Equivale al producto lógico.
- El resultado es verdad (1) cuando los dos valores son verdad (1).

A	B	$S = A \cdot B$
0	0	0
0	1	0
1	0	0
1	1	1

En programación, se usa, por ejemplo, en una estructura condicional de esta forma:

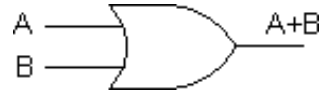
```
if (condicion_1 and condicion_2)
{se ejecutan estas ordenes si se
cumplen ambas condiciones}
else
{en caso contrario, se
ejecutarán estas ordenes}
```

<http://docs.oracle.com/javase/tutorial/java/nutsandbolts/op2.html>

http://es.wikibooks.org/wiki/Programación_en_Java/Operadores_booleanos

4. Operaciones lógicas

■ OR:



- Equivale a la suma lógica.
- El resultado es verdad (1) cuando alguno de los dos valores es verdad (1).

A	B	S = A + B
0	0	0
0	1	1
1	0	1
1	1	1

En programación, se usa, por ejemplo, en una estructura condicional de esta forma:

```
if (condicion_1 or condicion_2)
{se ejecutan estas ordenes si se
cumple alguna de las dos
condiciones}
else
{en caso contrario, se
ejecutarán estas ordenes}
```

4. Operaciones lógicas

■ **XOR:** 

- El resultado es verdad (1) cuando las dos entradas son diferentes

A	B	$S = A @ B$
0	0	0
0	1	1
1	0	1
1	1	0

4. Operaciones lógicas

■ NOT: 

□ El resultado es la negación del valor.

A	$S = \bar{A}$
0	1
1	0

4. Operaciones lógicas

- Videos:

- Video1: <https://youtu.be/8CRzrOKI96o>

- Video2: <https://youtu.be/o0736nvpys4>

5. Representación alfanumérica

5. Representación alfanumérica

- Son caracteres:

- ☐ Las letras (“a”, ..., “z”, “A”, ..., “Z”)
- ☐ Los dígitos (“0”, ..., “9”)
- ☐ Los signos de puntuación (“.”, “,”, “;”, ...)
- ☐ Y los símbolos especiales (“*”, “&”, “\$”, ...)

- Para representar los caracteres, se les asigna un código numérico a cada uno mediante una tabla.

- El ordenador siempre trabaja con los códigos, nunca con los símbolos gráficos.

- Las características de una representación son:

- ☐ Longitud en bits de los códigos.
- ☐ Cantidad de caracteres representable.
- ☐ La asignación de códigos a cada carácter (la tabla).

5. Representación alfanumérica

■ **EBCDIC:** *Extended Binary Coded Decimal Interchange Code*

- Surgió en 1964 con el sistema IBM S360.
- Longitud del código de 8 bits.
- Sólo se usa en algunos sistemas *mainframe* de IBM.

5. Representación alfanumérica

- **ASCII:** *American Standard Code for Information Interchange.*

- ☐ Pensado para el inglés.
- ☐ Utiliza 7 bits para codificar cada carácter → En total son 128 símbolos.

5. Representación alfanumérica

Caracteres
de control

Espacio en
blanco

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040	 	Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	\$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	((72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051))	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[[123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174	|	
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135]]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177		DEL

Source: www.LookupTables.com

5. Representación alfanumérica

■ ISO-8859-1 (Latin 1):

- Es una extensión del ASCII que incluye las lenguas de Europa occidental.
- Utiliza 8 bits para codificar cada carácter → En total son 256 símbolos.

5. Representación alfanumérica

Espacio en
blanco

ISO-8859-1																
	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xA	xB	xC	xD	xE	xF
0x	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	TAB	LF	VT	FF	CR	SQ	SI
1x	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2x	SP	!	"	#	\$	%	&	'	()	*	+	,	-	.	/
3x	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4x	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5x	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
6x	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7x	p	q	r	s	t	u	v	w	x	y	z	{		}	~	DEL
8x	PAD	HOP	BPH	NBH	IND	NEL	SSA	ESA	HTS	HTJ	VTS	PLD	PLU	RI	SS2	SS3
9x	DCS	PU1	PU2	STS	CCH	MW	SPA	EPA	SOS	SGCI	SCJ	CSI	ST	OSC	PM	APC
Ax	NBSP	ı	ç	£	¤	¥	ı	§	"	©	ª	«	¬	SHY	®	™
Bx	°	±	²	³	´	µ	¶	·	,	¹	º	»	¼	½	¾	¿
Cx	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï
Dx	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
Ex	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
Fx	ò	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ

5. Representación alfanumérica

■ Unicode:

- Código más moderno desarrollado como estándar.
- Las codificaciones anteriores no podían incluir caracteres suficientes.
- Es compatible con codificaciones anteriores como ASCII o ISO-8859-1.
- Hay diferentes versiones entre las que destacan UTF-16 (16 bits) y UTF-32 (32 bits).
- Norma adoptada por empresas como Apple, HP, IBM, Microsoft, Adobe, Oracle, Google,...
- Es requisito en estándares como XML, Java, ECMAScript (JavaScript), LDAP, CORBA 3.0, WML, etc.

5. Representación alfanumérica

■ Codificación de los 256 primeros caracteres:

	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00	NUL 0000	STX 0001	SOT 0002	ETX 0003	EOT 0004	ENQ 0005	ACK 0006	BEL 0007	BS 0008	HT 0009	LF 000A	VT 000B	FF 000C	CR 000D	SO 000E	SI 000F
10	DLE 0010	DC1 0011	DC2 0012	DC3 0013	DC4 0014	NAK 0015	SYN 0016	ETB 0017	CAN 0018	EM 0019	SUB 001A	ESC 001B	FS 001C	GS 001D	RS 001E	US 001F
20	SP 0020	! 0021	" 0022	# 0023	\$ 0024	% 0025	& 0026	' 0027	(0028) 0029	* 002A	+ 002B	, 002C	- 002D	. 002E	/ 002F
30	0 0030	1 0031	2 0032	3 0033	4 0034	5 0035	6 0036	7 0037	8 0038	9 0039	: 003A	; 003B	< 003C	= 003D	> 003E	? 003F
40	@ 0040	A 0041	B 0042	C 0043	D 0044	E 0045	F 0046	G 0047	H 0048	I 0049	J 004A	K 004B	L 004C	M 004D	N 004E	O 004F
50	P 0050	Q 0051	R 0052	S 0053	T 0054	U 0055	V 0056	W 0057	X 0058	Y 0059	Z 005A	[005B	\ 005C] 005D	^ 005E	_ 005F
60	` 0060	a 0061	b 0062	c 0063	d 0064	e 0065	f 0066	g 0067	h 0068	i 0069	j 006A	k 006B	l 006C	m 006D	n 006E	o 006F
70	p 0070	q 0071	r 0072	s 0073	t 0074	u 0075	v 0076	w 0077	x 0078	y 0079	z 007A	{ 007B	 007C	} 007D	~ 007E	DEL 007F
80	€ 20AC		/ 201A		// 201E	... 2026	† 2020	‡ 2021		‰ 2030	Š 0160	< 2033	Š 015A	Ť 0164	Ž 017D	Ž 0179
90		\ 2018	/ 2019	“ 201C	” 201D	• 2022	— 2013	— 2014		™ 2122	Š 0161	> 203A	Š 015B	Ť 0165	Ž 017E	Ž 017A
A0	NBSP 00A0	ˆ 02C7	˘ 02D8	Ł 0141	ł 00A4	À 0104	Á 00A6	Â 00A7	Ã 00A8	Ä 00A9	Å 015E	« 00AB	¬ 00AC	— 00AD	® 00AE	Ž 017B
B0	° 00B0	± 00B1	ℓ 02DB	ł 0142	ˆ 00B4	μ 00B5	¶ 00B6	• 00B7	„ 00B8	„ 0105	Š 015F	» 00BB	Ł 013D	“ 02DD	’ 013E	ž 017C
C0	Ř 0154	Á 00C1	Â 00C2	Ă 0102	Ä 00C4	Í 0139	Č 0106	Ç 00C7	Č 010C	É 00C9	Ê 0118	Ë 00CB	Ě 011A	Í 00CD	Î 00CE	Ď 010E
D0	Đ 0110	Ň 0143	Ň 0147	Ó 00D3	Ô 00D4	Õ 0150	Ö 00D6	× 00D7	Ř 0158	Ů 016E	Ú 00DA	Ů 0170	Ů 00DC	Ý 00DD	Ť 0162	ß 00DF
E0	í 0155	á 00E1	â 00E2	ă 0103	ä 00E4	í 013A	č 0107	ç 00E7	č 010D	é 00E9	ê 0119	ë 00EB	ě 011B	í 00ED	î 00EE	ď 010F
F0	č 0111	ň 0144	ň 0148	ó 00F3	ô 00F4	õ 0151	ö 00F6	÷ 00F7	ř 0159	ů 016F	ú 00FA	ů 0171	ù 00FC	ý 00FD	ț 0163	· 02D9

<http://www.unicode.org>

6. Múltiplos

6. Múltiplos

■ Múltiplos del sistema internacional:

Kilo(K)	$10^3 = 1000$
Mega(M)	$10^6 = 1000 \text{ K}$
Giga(G)	$10^9 = 1000 \text{ M}$
Tera(T)	$10^{12} = 1000 \text{ G}$
Peta(P)	$10^{15} = 1000 \text{ T}$
Exa(E)	$10^{18} = 1000 \text{ P}$

■ Múltiplos del sistema binario:

KiBi(KB)	$2^{10} = 1024$
MeBi(MB)	$2^{20} = 1024 \text{ KiBi}$
GiBi(GB)	$2^{30} = 1024 \text{ MeBi}$
TeBi(TB)	$2^{40} = 1024 \text{ GiBi}$
PeBi(PB)	$2^{50} = 1024 \text{ TeBi}$
ExBi(EB)	$2^{60} = 1024 \text{ PeBi}$

6. Múltiplos

- Tabla resumen: Múltiplos en ambos sistemas

Nombre	Abrev.	Factor binario	Tamaño en el SI
bytes	B	$2^0 = 1$	$10^0 = 1$
kilo	k	$2^{10} = 1024$	$10^3 = 1000$
mega	M	$2^{20} = 1\,048\,576$	$10^6 = 1\,000\,000$
giga	G	$2^{30} = 1\,073\,741\,824$	$10^9 = 1\,000\,000\,000$
tera	T	$2^{40} = 1\,099\,511\,627\,776$	$10^{12} = 1\,000\,000\,000\,000$
peta	P	$2^{50} = 1\,125\,899\,906\,842\,624$	$10^{15} = 1\,000\,000\,000\,000\,000$
exa	E	$2^{60} = 1\,152\,921\,504\,606\,846\,976$	$10^{18} = 1\,000\,000\,000\,000\,000\,000$
zetta	Z	$2^{70} = 1\,180\,591\,620\,717\,411\,303\,424$	$10^{21} = 1\,000\,000\,000\,000\,000\,000\,000$
yotta	Y	$2^{80} = 1\,208\,925\,819\,614\,629\,174\,706\,176$	$10^{24} = 1\,000\,000\,000\,000\,000\,000\,000\,000$

6. Múltiplos

■ Ampliación



Ampliación

Bit = mínima unidad de información.

4 Bits = Nibble o cuarteto.

8 Bits = 1 Byte.

1 024 Bytes = 1 Kilobyte.

1 024 Kilobytes = 1 Megabyte (MB).

1 024 Megabytes = 1 Gigabyte (GB).

1 024 Gigabytes = 1 Terabyte (TB).

1 024 Terabytes = 1 Petabyte (PB).

1 024 Petabytes = 1 Exabyte (EB).

1 024 Exabytes = 1 Zettabyte (ZB).

1 024 Zettabytes = 1 Yottabyte (YB).

1 024 Yottabytes = 1 Brontobyte (BB).

1 024 Brontobytes = 1 Geopbyte (GeB).



Ten en cuenta

Es habitual encontrar escrito Kb o KB de forma indistinta, pero tenemos que diferenciar entre ambas escrituras. La **B** referencia **Bytes** y la **b** representa **bits**. Solamente se utilizará la b minúscula para representar medidas de transferencia de información como Kbps (Kilobits por segundo).