FUNCIONAMIENTO DIGITAL

DE UN SISTEMA.

EL SISTEMA BINARIO

Fr. Casares

Sistema Digital

-Emplea dispositivos en los que solo son posibles dos estados

Elemento	Situación	
	0(Falso)	1(Verdadero)
Relé	Desactivado	Activado
Válvula	Cerrada	Abierta
Línea	Sin Tensión	Con Tensión
Presostato	Sin Presión	Con Presión
Bomba	Apagada	Encendida

Sistema Digital

-Emplea dispositivos en los que solo son posibles dos estados

Elemento	Situación	
	0(Falso)	1(Verdadero)
Relé	Desactivado	Activado
Válvula	Cerrada	Abierta
Línea	Sin Tensión	Con Tensión
Presostato	Sin Presión	Con Presión
Bomba	Apagada	Encendida

Estado normal **Estado Excitado**

Sistema Digital

- Estos dos estados se pueden designar de varia formas, siendo las mas corrientes las siguientes:

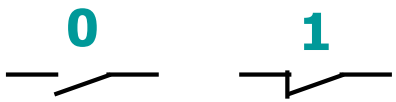
“1”	Alto	Verdadero	Con tensión	Encendido
“0”	Bajo	Falso	Sin tensión	Apagado

Lógica Positiva

- Los sistemas electrónicos se adaptan perfectamente al sistema binario utilizando la notación “**0**” y “**1**” para los estados de los elementos y tambien para representar los numeros, cadenas de texto, variables, combinaciones lógicas, aritmetica, etc. .

Sistemas Electrónicos → Sistemas Digitales

Estado de un elemento



Numeración binaria

$$110_2 = 6_{10}$$

Representación cadenas de Texto

"0100 0001 0110 0010" ="Ab"

Aritmética

$$\begin{array}{rl} & 0110 \rightarrow 6 \\ + & 0001 \rightarrow 1 \\ \hline & 0111 \rightarrow 7 \end{array}$$

Necesitamos un sistema de codificación

Concepto de codificación: Ejemplo→ Representación de números.

Existen varios sistemas de representación de números.

La no posicional: El peso del dígito se representa por el propio símbolo.

Ejemplo: la romana: MMCM = M + M - C + M = 2900

IIII = 4

La posicional: *r* **símbolos diferentes por dígito**. El peso del dígito se representa por la posición. Cada dígito tiene un peso r^i si *i* es la posición.

$2979 = 2x10^3 + 9x10^2 + 7x10^1 + 9x10^0$ (decimal)

La base de un sistema de numeración es el numero de símbolos distintos utilizados para la representación de las cantidades de los mismos.

Concepto de codificación: Ejemplo→ Representación de números.

Numeración posicional:

	Símbolos diferentes por dígito	
Decimal	→ 10	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
Hexadecimal	→ 16	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
Octal	→ 8	0, 1, 2, 3, 4, 5, 6, 7
Binario	→ 2	0, 1

r^7	r^6	r^5	r^4	r^3	r^2	r^1	r^0
0	0	0	0	0	0	3	1

$2\ r^2 + 3\ r^1 + 1r^0$

r = nº de símbolos de la base

El peso del dígito se representa por la posición

Concepto de codificación: Ejemplo→ Representación de números.

Numeración posicional:

Sistemas de Numeración

Numeración posicional o arábiga

Símbolos diferentes por dígito

10 → Decimal

16 → Hexadecimal

8 → Octal

2 → Binario

El peso del dígito se representa por la posición

Dec	Hex	Oct	Bin
0	0	000	00000000
1	1	001	00000001
2	2	002	00000010
3	3	003	00000011
4	4	004	00000100
5	5	005	00000101
6	6	006	00000110
7	7	007	00000111
8	8	010	00001000
9	9	011	00001001
10	A	012	00001010
11	B	013	00001011
12	C	014	00001100
13	D	015	00001101
14	E	016	00001110
15	F	017	00001111

Concepto de codificación: Ejemplo→ Representación de números.

Código	Cantidad								Nº dígitos necesar.
Numero decimal						1	2	6	3
Numero binario	0	1	1	1	1	1	1	0	7
Numero hexadecimal							7	E	2

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	1	1	1	1	1	1	0

$= 0 \times 2^7 + 1 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 +$
$+ 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 126$

Cada dígito se denomina BIT

BIT : Es la menor unidad de información en el sistema binario (0,1)

16^7	16^6	16^5	16^4	16^3	16^2	16^1	16^0
0	0	0	0	0	0	7	E

$= 7 \times 16^1 + 14 \times 16^0 = 126$

Codificación de datos en sistemas digitales:

Para poder transmitir y manejar información es necesario codificarla, representarla mediante un conjunto de símbolos que constituye un código.

Con un vector binario de **n componentes** tendremos 2^n combinaciones distintas y se podrá codificar hasta un alfabeto con 2^n elementos .

Código o palabra			Eq. Decimal	Nº de códigos diferentes
Dos Variables	00 11	→ →	0 3	2^2=4
Tres Variables	000 111	→ →	0 7	2^3=8
Cuatro Variables	0000 1111	→ →	0 15	2^4=16
Seis Variables	000000 111111	→ →	0 63	2^6=64
Ocho Variables	0000 0000 1111 1111	→ →	0 255	2^8=256
Dieciséis Variables	00000000 00000000 11111111 11111111	→ →	0 65535	2^{16}=65536

Nº decimales:
10 números → código de 4 variables

Letras:
26 letras (sin distinguir mayúsculas/minúsculas)
52 letras (mayúsculas y minúsculas)

Letras + nº decimales:

52+10 = 62 elementos → código de 8 variab

Un grupo de varios bits (vector) que tengan un determinado significado es una información, palabra o código.

Codificación de datos en sistemas digitales:

A.- Códigos de números (datos) en binario:

Códigos de 4 variables (o dígitos):

- BCD natural
- Código gray
- Código BCD exceso a tres

Códigos de 8, 16, 32, 64 variables (o dígitos):

- Codificación de enteros sin signo (8, 16, 32, 64)
- Codificación de enteros con signo (16, 32, 64)
- Codificación de números reales (32, 64)

B.- Codificación alfanumérica: ASCII

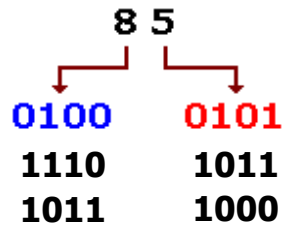
Codificación de números en binario:

Códigos de 4 variables o dígitos (4 BIT):

Sistema BCD (Binario Código Decimal)

Código Decimal	BCD natural	BCD Aiken 2421	BCD 5421
0	0000	0000	0000
1	0001	0001	0001
2	0010	0010	0010
3	0011	0011	0011
4	0100	0100	0100
5	0101	1011	1000
6	0110	1100	1001
7	0111	1101	1010
8	1000	1110	1011
9	1001	1111	1100

Es un método para expresar un dígito de un número decimal en notación binaria. Cada dígito decimal se expresa por cuatro bits traduciéndose así en forma aislada.



Codificación de números en binario:

Códigos de 4 variables o dígitos (4 BIT):

Decimal	BCD	BIN
10	0001 0000	0000 1010
11	0001 0001	0000 1011
12	0001 0010	0000 1100
13	0001 0011	0000 1101
14	0001 0100	0000 1110
21	0010 0001	0001 0101

Con un vector de 4 bits → de 0 a 9 → de 0 a 16

Con un vector de 8 bits → de 0 a 99 → de 0 a 255

Con un vector de 16 bits → de 0 a 9999 → de 0 a 65535

Codificación de números en binario:

Códigos de 4 variables o dígitos (4 BIT):

Nº Decimal:	**160**
BCD natural	**0001 0110 0000**
BCD aiken	**0001 1101 0000**
BCD 5421	**0001 1001 0000**

Binario → 0000 1010 0000

Codificación de datos en sistemas digitales:

A.- Códigos de números (datos) en binario:

Códigos de 4 variables (o dígitos):

- **BCD natural**
- **Código gray**
- **Código BCD exceso a tres**

Códigos

Codif

Codif

Codif

Código Gray: Es un código continuo porque las combinaciones correspondientes a números decimales consecutivos difieren solamente en un bit

***Código BCD exceso tres*:** Se obtiene de sumar tres a cualquiera de las combinaciones del código BCD natural

B.- Codificación alfanumérica: ASCII

DECIMAL	BCD	Exceso 3
	8421	
0	0000	0011
1	0001	0100
2	0010	0101
3	0011	0110
4	0100	0111
5	0101	1000
6	0110	1001
7	0111	1010
8	1000	1011
9	1001	1100

Decimal	Binario	GRAY
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

Codificación de datos en sistemas digitales:

A.- Códigos de números (datos) en binario:

Códigos de 4 variables (o dígitos):

- BCD natural
- Código gray
- Código BCD exceso a tres

⇨ Códigos de 8, 16, 32, 64 variables (o dígitos):

- Codificación de enteros sin signo (8, 16, 32, 64)
- Codificación de enteros con signo (16, 32, 64)
- Codificación de números reales (32, 64)

B.- Codificación alfanumérica: ASCII

Codificación de números en binario:

Códigos de 8, 16, 32, 64 variables o dígitos

- → Codificación de enteros sin signo (8, 16, 32, 64)
- Codificación de enteros con signo (16, 32, 64)
- Codificación de números reales (32, 64)

---

8 bit

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	0	0	0	1	1	0	1

256 Combinaciones = 2^8

Numero entero: 0-255

16 bit

2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	1

65536 Combinaciones = 2^{16}

Numero entero: [0,65535]

32 bit } ***4.294.967.296 Combinaciones = 2^{32}***

Numero entero: 0- 4.294.967.296

Codificación de números en binario :

Códigos de 8, 16, 32, 64 variables o dígitos

- Codificación de enteros sin signo (8, 16, 32, 64)
- → Codificación de enteros con signo (16, 32, 64)
- Codificación de números reales (32, 64)

- Signo más valor absoluto → SM
- Complemento a 1 →1C
- Complemento a 2 →2C

Bit de signo

8 bit

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	0	0	0	1	1	0	1

256 Combinaciones = 2^8

Numero: (-127,127)

16 bit

2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	1

65536 Combinaciones = 2^{16}

Numero: (-32767,32767)

Codificación de números en binario :

Códigos de 8, 16, 32, 64 variables o dígitos

- Codificación de enteros sin signo (8, 16, 32, 64)
- → Codificación de enteros con signo (16, 32, 64)
- Codificación de números reales (32, 64)

- Signo más valor absoluto → SM
- Complemento a 1 →1C
- Complemento a 2 →2C

En numeración binaria tenemos dos tipos de complementos:

Complemento a 1: Se obtiene escribiendo el bit de estado opuesto.

Numero:	0	1	101110
Complemento a uno:	1	0	010001

Complemento a dos: Se obtiene hallando primero el complemento a 1 y después sumándole 1.

Numero:	101110
Complemento a dos:	010010

Codificación de números en binario :

Códigos de 8, 16, 32, 64 variables o dígitos

Codificación de enteros sin signo (8, 16, 32, 64)

→ Codificación de enteros con signo (16, 32, 64)

Codificación de números reales (32, 64)

- Signo más valor absoluto → SM
- Complemento a 1 →1C
- Complemento a 2 →2C

	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	
	0	0	0	0	0	1	0	0	⇨ 4_{10}
SM	1	0	0	0	0	1	0	0	⇨ -4_{10}
C1	1	1	1	1	1	0	1	1	⇨ -4_{10}
C2	1	1	1	1	1	1	0	0	⇨ -4_{10}

Codificación de números en binario :

Códigos de 8, 16, 32, 64 variables o dígitos

Codificación de enteros sin signo (8, 16, 32, 64)

→ Codificación de enteros con signo (16, 32, 64)

Codificación de números reales (32, 64)

- Signo más valor absoluto → SM
- Complemento a 1 →1C
- Complemento a 2 →2C

Ejemplo con 8 bits (2C):

-25	→	1	1	1	0	0	1	1	1	+25	→	0	0	0	1	1	0	0	1
-3	→	1	1	1	1	1	1	0	1	+3	→	0	0	0	0	0	0	1	1
-1	→	1	1	1	1	1	1	1	1	+1	→	0	0	0	0	0	0	0	1

Codificación de números en binario :

Códigos de 8, 16, 32, 64 variables o dígitos

Codificación de enteros sin signo (8, 16, 32, 64)

Codificación de enteros con signo (16, 32, 64)

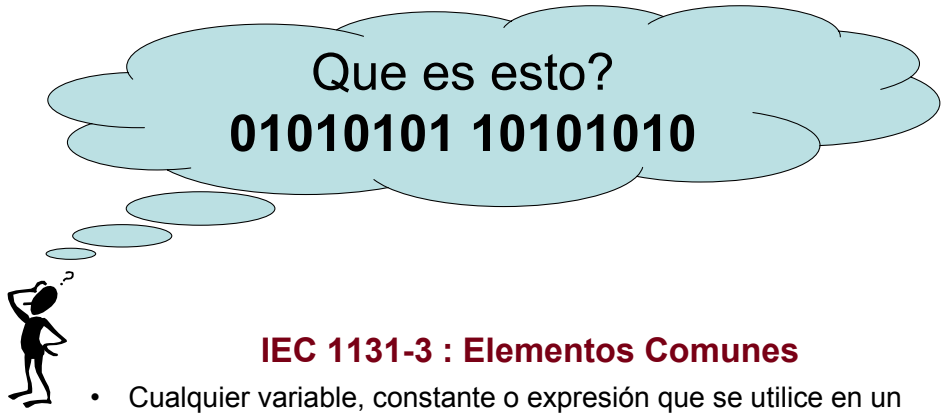
→ Codificación de números reales (32, 64)

Coma flotante

- El código binario se divide en dos campos: Mantisa y exponente
- Se necesitan 32 bits: 1 de signo + 23 de mantisa + 8 exponente
- El numero real equivalente es igual a: x = Mantisa • $2^{\text{exponente}}$
- Estandar IEEE 7544

Coma fija

- El punto decimal ocupa una posición fija
- 100111,11 → 159 x 2^{-2} → 39,75



IEC 1131-3 : Elementos Comunes

- Cualquier variable, constante o expresión que se utilice en un programa (escrito en cualquier lenguaje) debe estar caracterizado por un tipo de dato. La coherencia de tipos deberá mantenerse en las operaciones gráficas y sentencias literales.
- Desde el principio se conoce si un dato es un String, una fecha o un Entero, y por tanto, no hay confusión cuando diferentes personas trabajan en un proyecto usando la representación textual (el nombre de la variable).

IEC 61131

Que es esto?
01010101 10101010

IEC 1131-3 : Elementos Comunes

- Ejemplos de tipos de datos estándar son: **Bool, Byte Integer, Real**, los cuales conocemos. Pero aparecen otros como: **Date, Time_of_day, String**
- El tipo de dato lo que refleja en realidad es la forma de almacenamiento en la memoria del autómata: en binario (numero enteros), BCD (Fechas, números), complemento a dos(números enteros con signo), Números en coma flotante según el estándar IEEE (para los reales).

Tipos de datos & Variables

IEC 61131

Que es esto?
01010101 10101010

Tipos de datos & Variables

Enteros sin signo

8 bit → 0-255 → **USINT** Unsigned Short Integer

16 bit → 0-65535 → **UINT** Unsigned Integer

32 bit → 0-2^{32} → **UDINT** Unsigned double Integer

64 bit → → **ULINT** Unsigned Long Integer

Enteros con signo

16 bit → - 32768 a +32768 → **INT** Integer

32 bit → -2147483648 a 2147483647 → **DINT** Double Integer

64 bit → - 2^{64} a (2^{64}-1) → **LINT** Long Integer

Números reales

32 bit → → **REAL** real de precision simple

64 bit → → **LREAL** real de precision doble

IEC 61131

Que es esto?
01010101 10101010

Tipos de datos & Variables

TIPO DE DADO	TAMANHO (em memória)	INTERVALO
BOOL	1 bit	TRUE e FALSE
INT	16 bits	-32768 a +32767
UINT	16 bits	0 a 65535
WORD	16 bits	0 a FFFF
DINT	32 bits	-2147483648 a +2147483647
UDINT	32 bits	0 a 4294967295
DWORD	32 bits	0 a FFFFFFFF
REAL	32 bits	-3.40282346638528860e+38 a 3.40282346638528860e+38 Underflow: 1.1754943508222875e-38
DATE	32 bits	01/01/2000 a 31/12/2080
TIME	32 bits	0 a 49d17h2m47s290ms
TIME_OF_DAY	32 bits	00:00:00 a 23:59:59
DATE_AND_TIME		
STRING		caracteres ASCII
ARRAY		
STRUCT		

Que es esto?
01010101 10101010

Configuración

```
CONFIGURATION EJEMPLO
  VAR_GLOBAL
    C1: INT;
    C2: BYTE AT %IB0;
    C3: INT AT %IW5;
  END_VAR
  RESOURCE SIGNALPROCESSING ON I386
    VAR_GLOBAL
      R1: INT AT %QW4;
      R2: BOOL AT %IX15;
      R3: INT AT %IW3;
    END_VAR
    TASK SLOW(INTERVAL:=TIME#20MS,
              PRIORITY:=2);
    (* OTRAS TAREAS NO MOSTRADAS *)
    PROGRAM P1 WITH SLOW:
             FILTRO(X:=C3, Y=>R3);
    (* OTROS PROGRAMAS NO MOSTRADOS *)
  END_RESOURCE
  (* OTROS RECURSOS NO MOSTRADOS *)
END CONFIGURATION
```

Direccionamiento según el IEC61131-3

Esta tabla representa el direccionamiento según el estándar para cualquier recurso del PLC.

Dirección IEC	Descripción
%	Identificador de direccionamiento IEC
I	Entrada
Q	Salida
M	Área de Memoria
X	Dato tipo BOOL (1 bit)
W	Dato tipo WORD (16 bits)
D	Dato tipo DOUBLE WORD (32 bits)
No_1	a.) Para I y Q: No_1 = Nº de palabra b.) Para M: No_1 = Referencia de Memoria Interna Relé internos, relés especiales R/WR/DWR ⇒ 0 Temporizador T ⇒ 1 Contador C ⇒ 2 Valor de preselección cont/temp SV/DSV ⇒ 3 Valor actual cont/temp EV/DEV ⇒ 4 Registro de Datos, registros especiales ST/DDT ⇒ 5 Registro índice IX,IY ⇒ 6 Relé de enlace L/WL/DWL ⇒ 7 Registro de enlace Ld/DLd ⇒ 8 Registro de datos FL/DFL ⇒ 9 Relé de alarma de Error E ⇒ 10 Relé de pulsos P ⇒ 11
.	Separador
No_2	a.) Para I y Q: No_2 ⇒ Posición del bit en la palabra b.) Para M: Tipo bit: No_2 ⇒ Nº de contacto Otro tipo: No_2 ⇒ Nº de palabra
.	Separador
No_3	Posición del bit en la palabra para R, L ó P (si No_1 = 0, 7 ó 11)

Ejemplo:

X0	%IX 0.0
X2F	%IX 2.15
Y0	%QX 0.0
Y30	%QX 3.0
R0	%MX 0.0.0
R5	%MX 0.0.5
R200	%MX 0.20.0
DT0	%MW 5.0
DT200	%MW 5.200
T1	%MX 1.1

AGRUPACIÓN DE NÚMEROS BINARIOS: BIT, BYTE, WORD, DOUBLE WORD

En sistemas digitales:

BIT → Nº binario compuesto por un digito.

(la menor unidad de informacion: "1", "0")

BYTE → Nº binario compuesto por 8 digitos

WORD → Nº binario compuesto por 16 digitos (2 BYTE)

DOUBLE WORD → Nº binario compuesto por 32 digitos(4 BYTE)

AGRUPACIÓN DE NÚMEROS BINARIOS: BIT, BYTE, WORD, DOUBLE WORD

En sistemas digitales:

BIT → Nº binario compuesto por un digito.

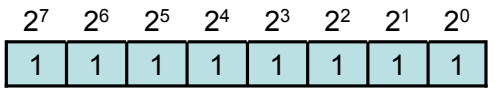
(la menor unidad de informacion: "1", "0")

BYTE → Nº binario compuesto por 8 digitos

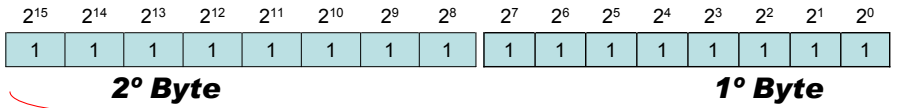
WORD → Nº binario compuesto por 16 digitos (2 BYTE)

DOUBLE WORD → Nº binario compuesto por 32 digitos(4 BYTE)

Nº máximo de valores en numeración binaria que pueden representar las combinaciones de bit:



1 BYTE ***256 Combinaciones***



1 Palabra ***65536 Combinaciones***

Palabra ***Palabra***

1 Doble Palabra ***+4.000 Millones Comb***

Codificación alfanumérica en sistemas digitales:

Letras: 26 letras (sin distinguir mayúsculas y minúsculas)

52 letras (mayúsculas y minúsculas)

Nº decimales: 10 números

Total: 62 combinaciones diferentes

Código de:	Nº de códigos diferentes
4 variables	2^4=16
6 variables	2^6 =64
7 variables	2^7=128
8 variables	2^8=256

ASCII

American Standard Code for Information Interchange

Estándar Americano de Codificación para el Intercambio de Información

ASCII

- Código de 7 bits (128 caracteres diferentes).
- ASCII ampliado o completo: Código de 8 bits.
- Significado por contexto.

Tabla ASCII

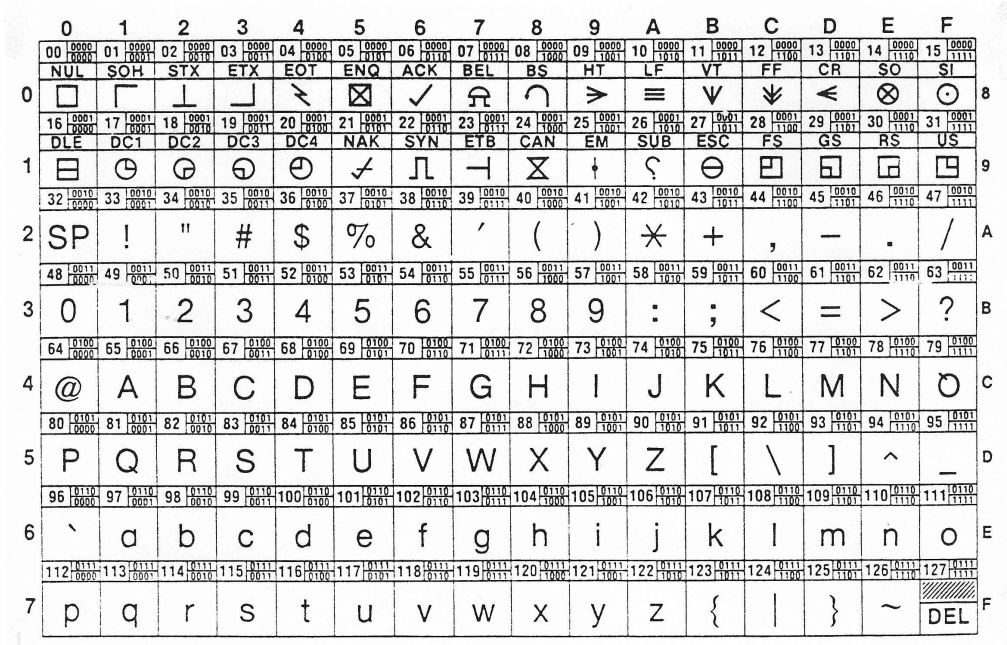
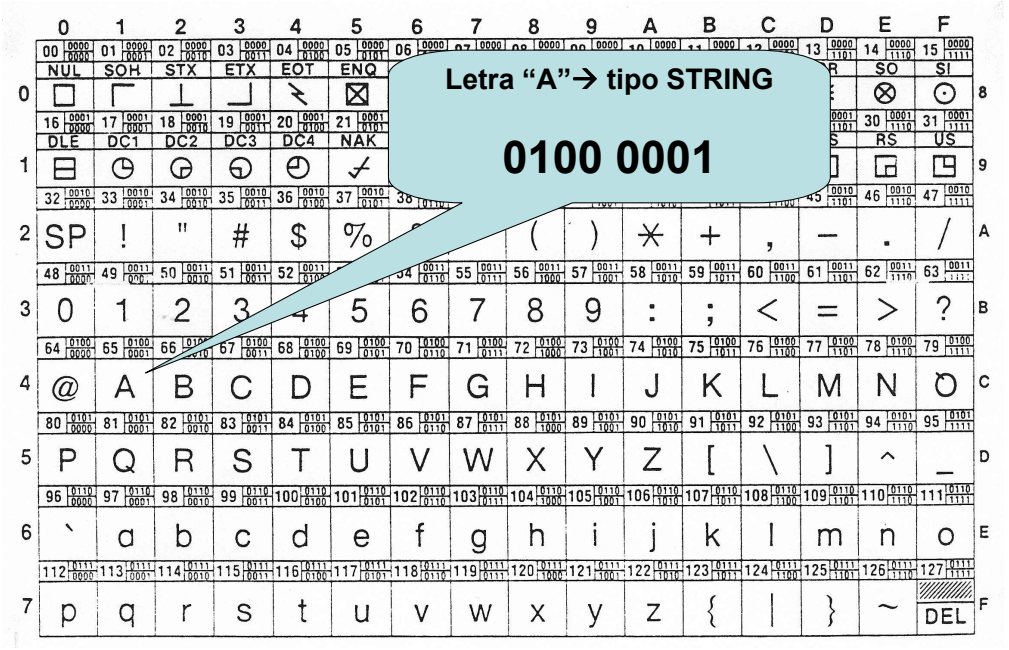
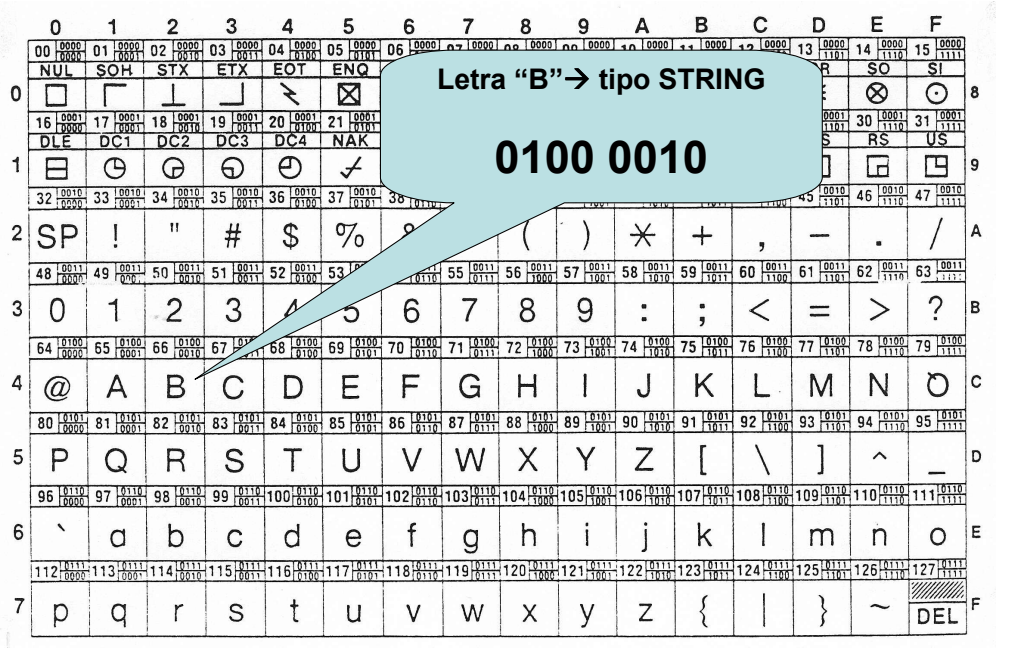


Tabla ASCII



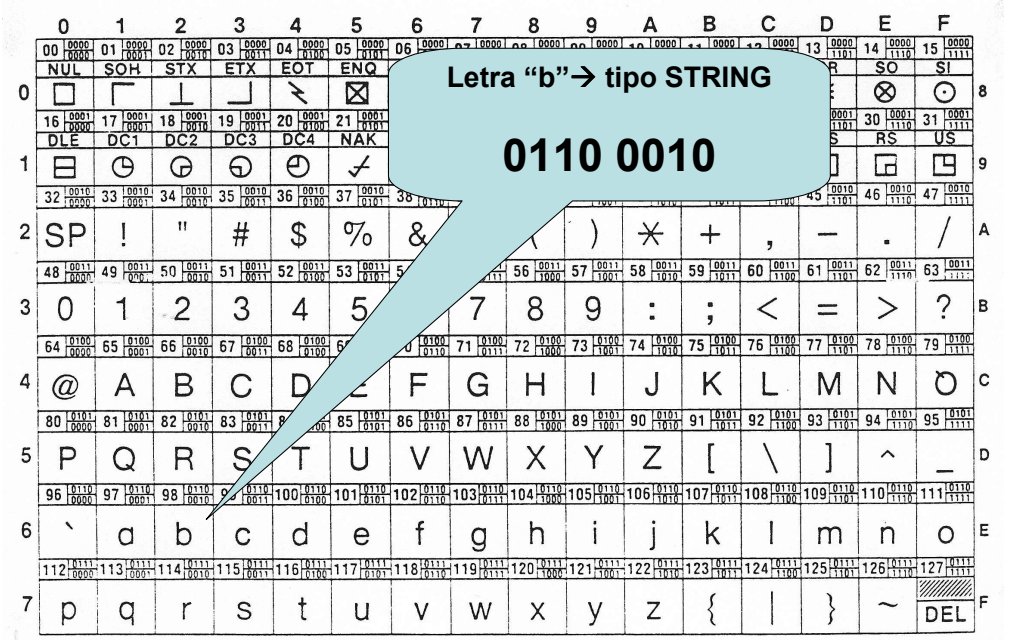
Caracteres imprimible: del 32 al 127

Tabla ASCII



Caracteres imprimible: del 32 al 127

Tabla ASCII



Caracteres imprimible: del 32 al 127

Tabla ASCII

Caracteres imprimible: del 32 al 127

Tabla ASCII

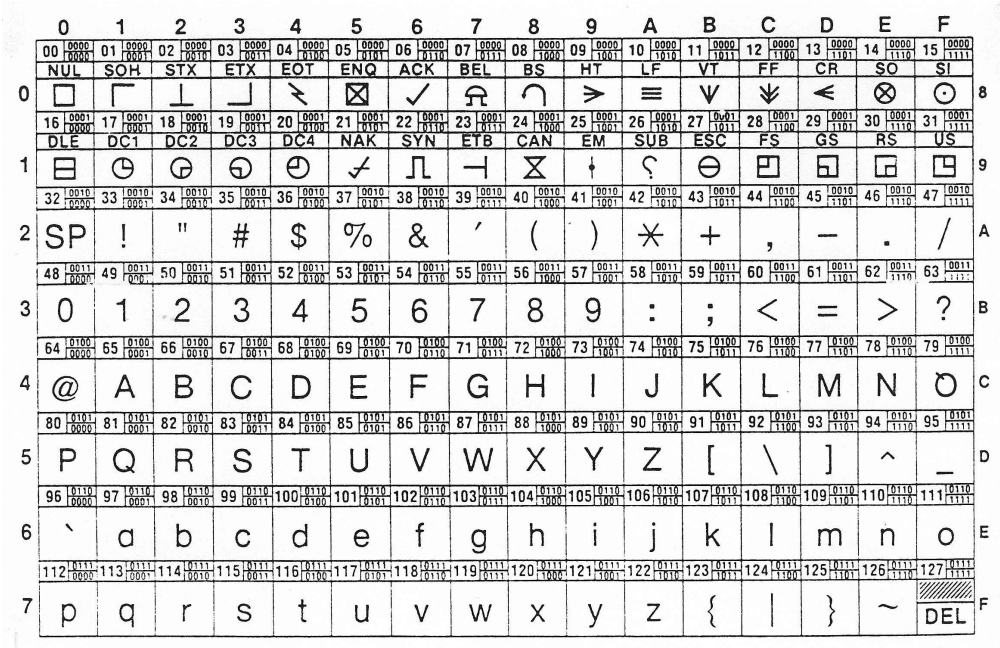


Tabla ASCII

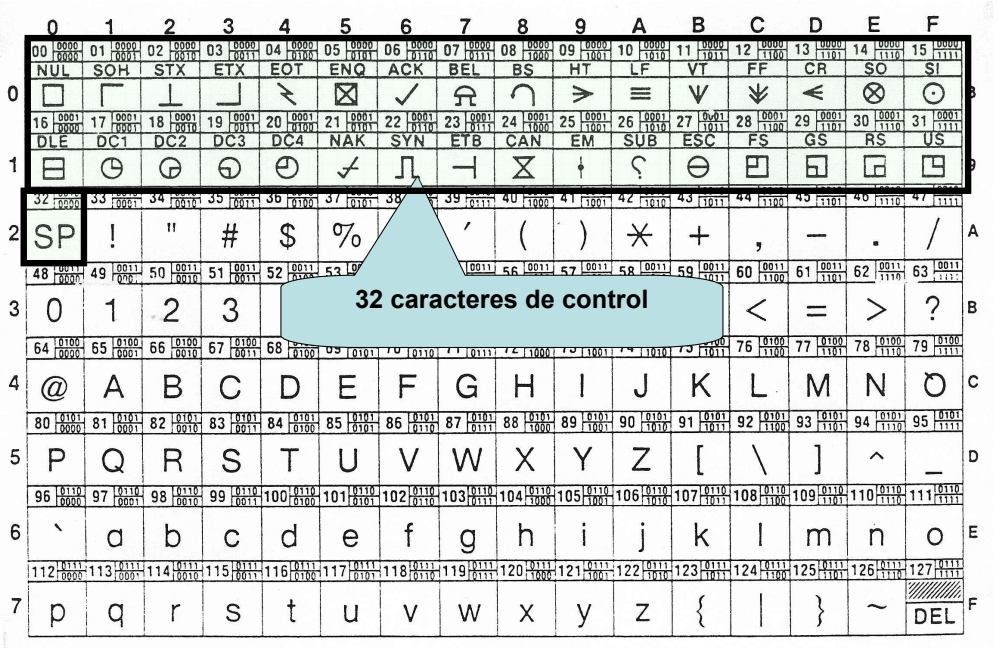
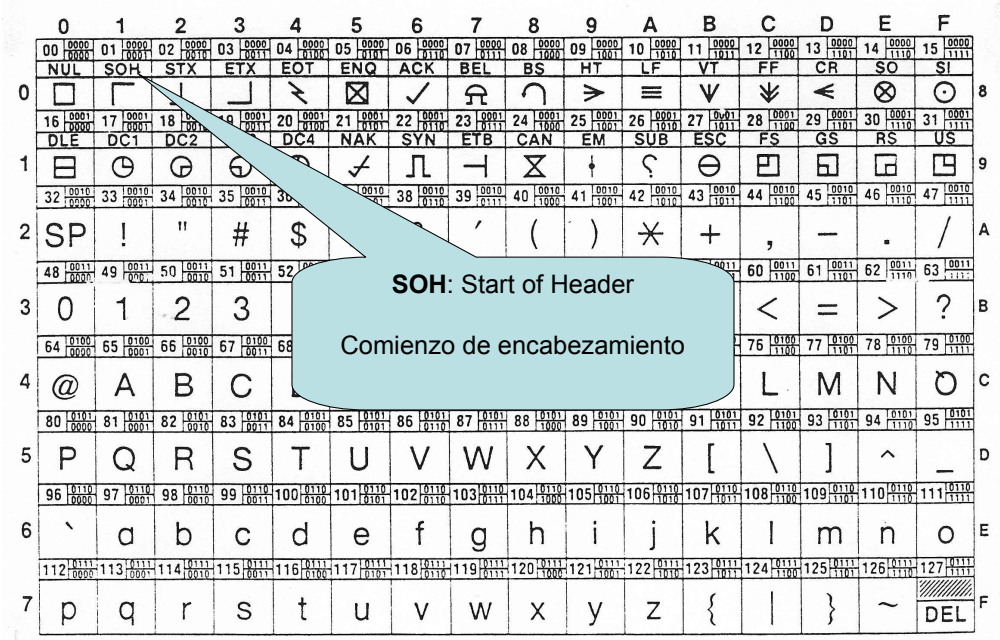
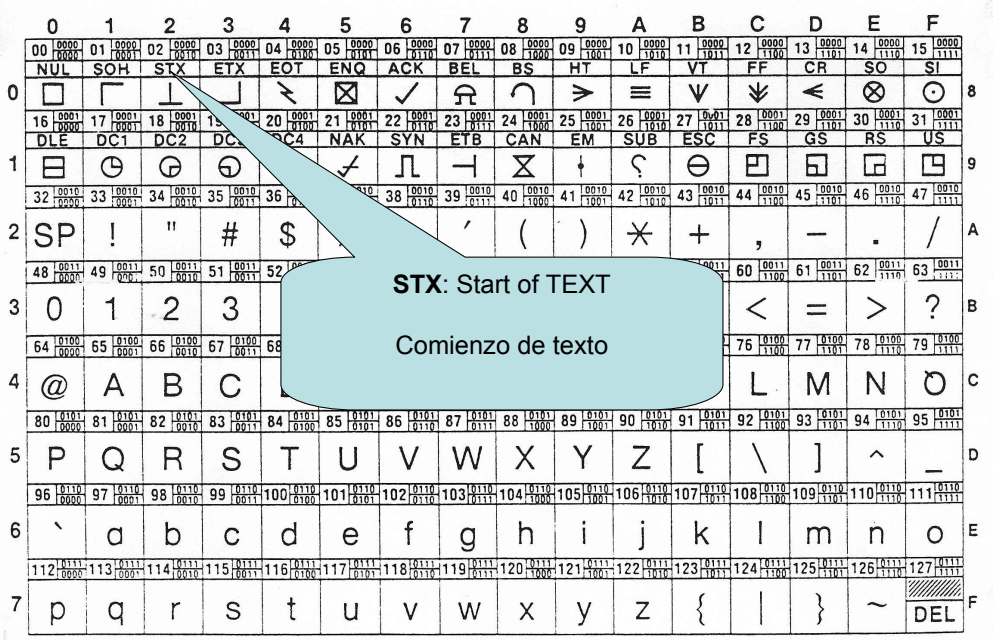


Tabla ASCII



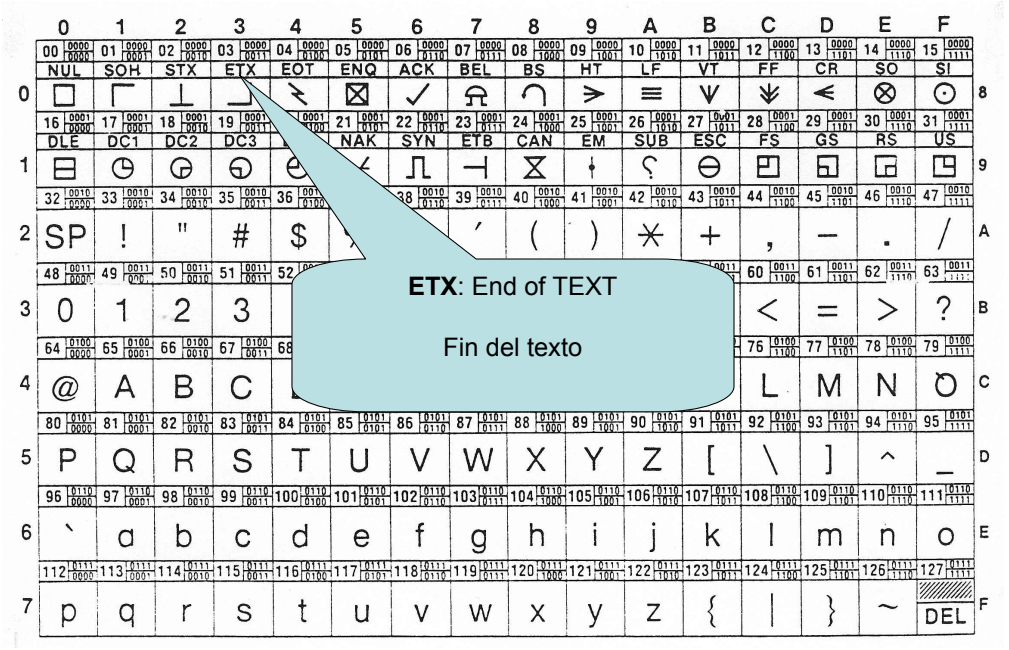
Caracteres de Control de las comunicaciones lógicas

Tabla ASCII



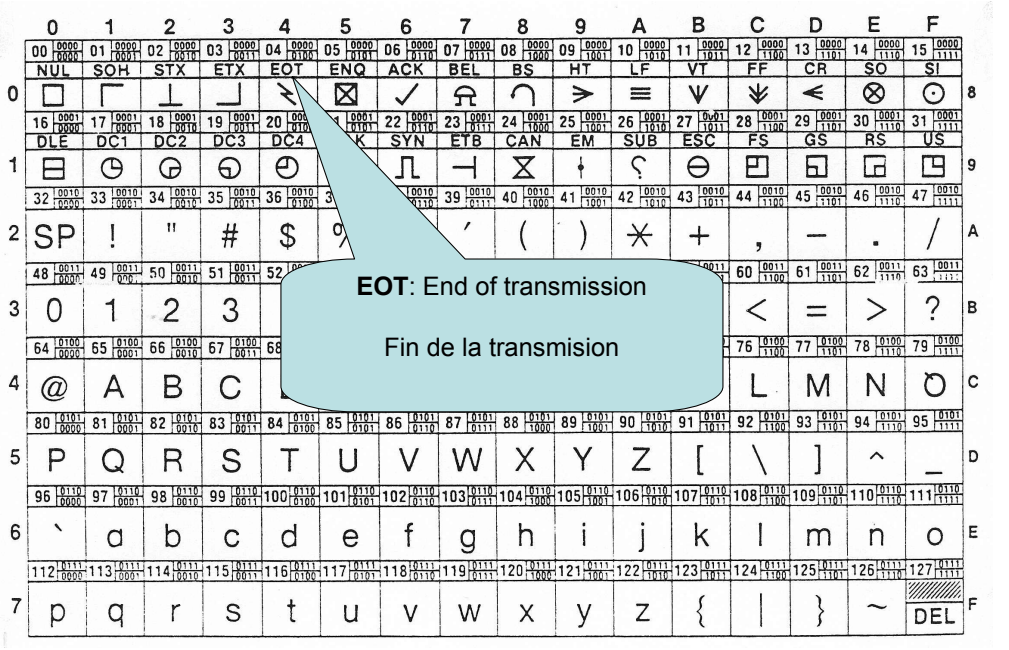
Caracteres de Control de las comunicaciones lógicas

Tabla ASCII



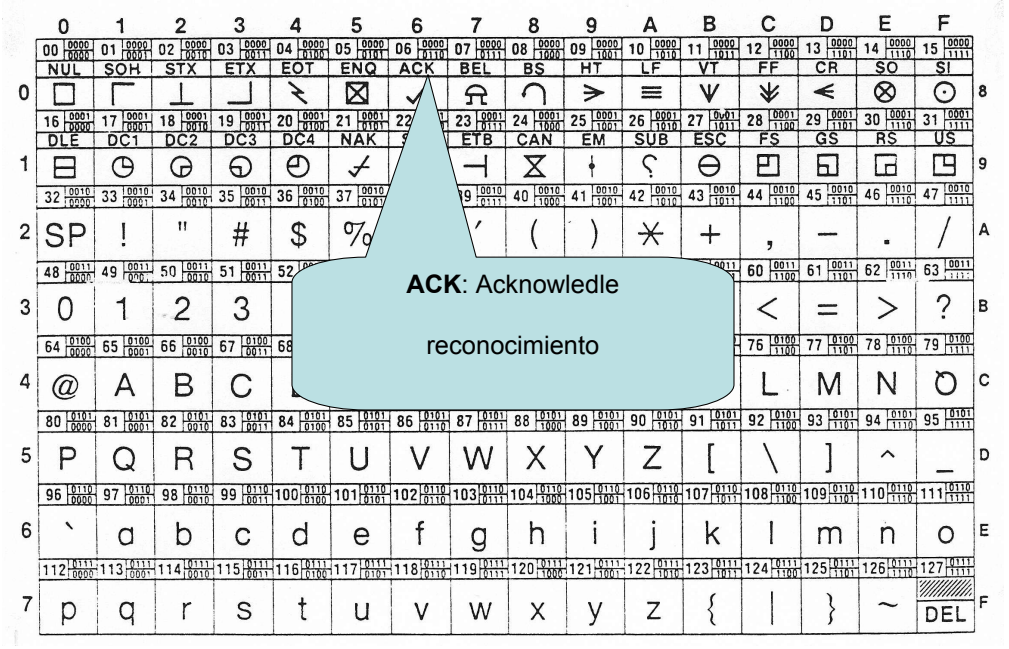
Caracteres de Control de las comunicaciones lógicas

Tabla ASCII



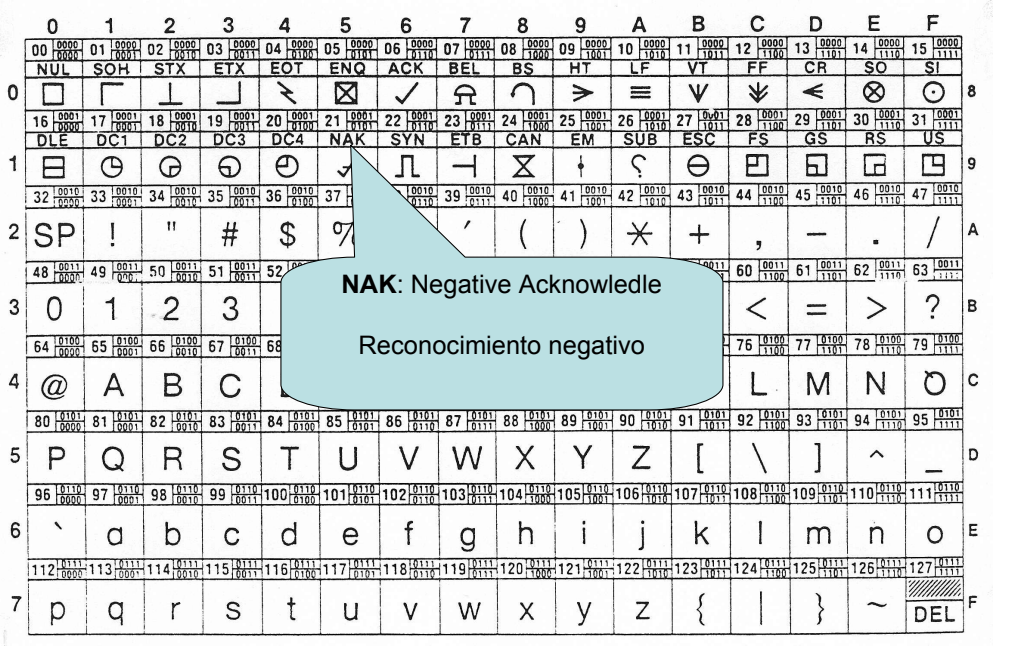
Caracteres de Control de las comunicaciones lógicas

Tabla ASCII



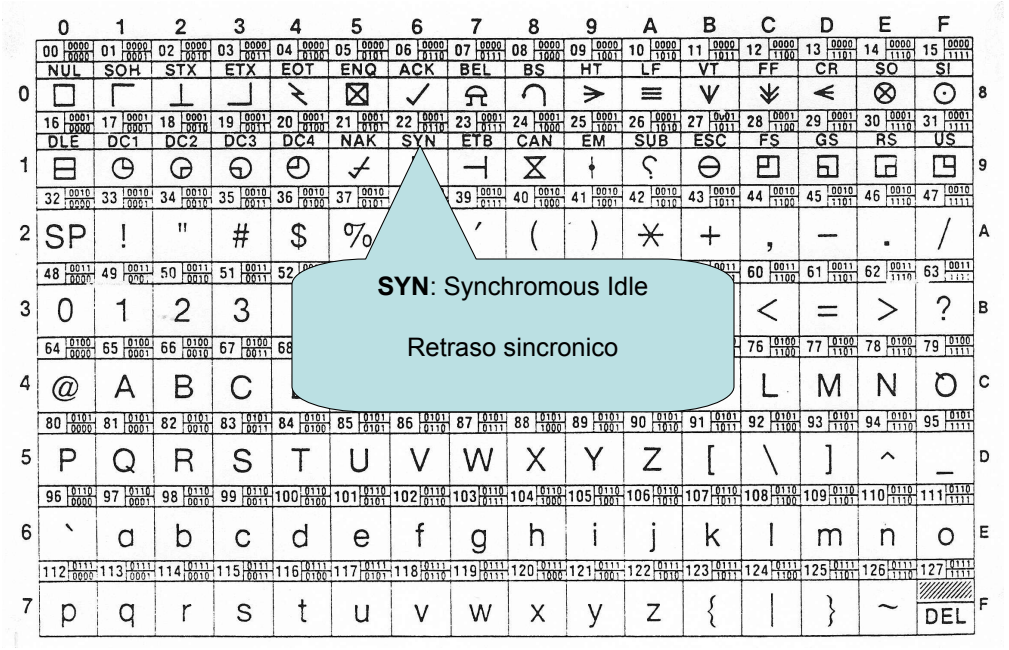
Caracteres de Control de las comunicaciones lógicas

Tabla ASCII



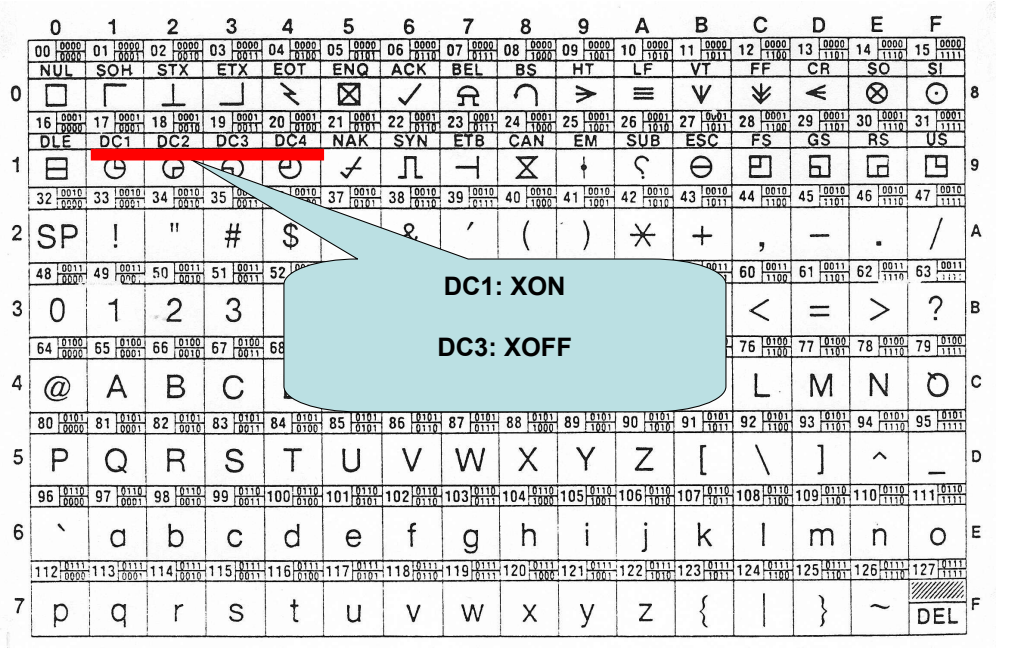
Caracteres de Control de las comunicaciones lógicas

Tabla ASCII



Caracteres de Control de las comunicaciones lógicas

Tabla ASCII



Caracteres de Control del flujo de la información

Tabla ASCII

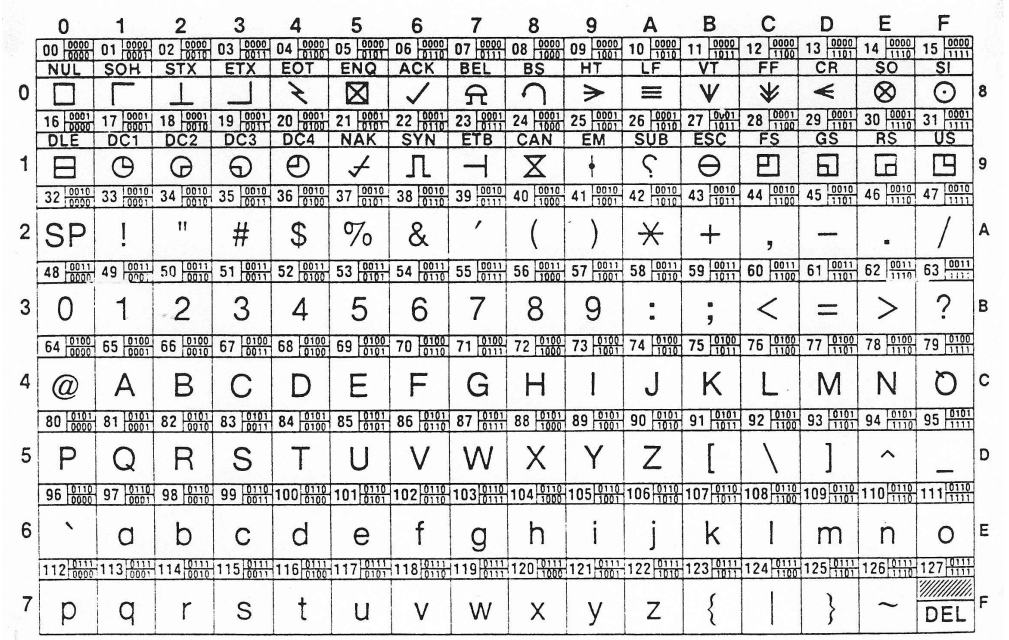
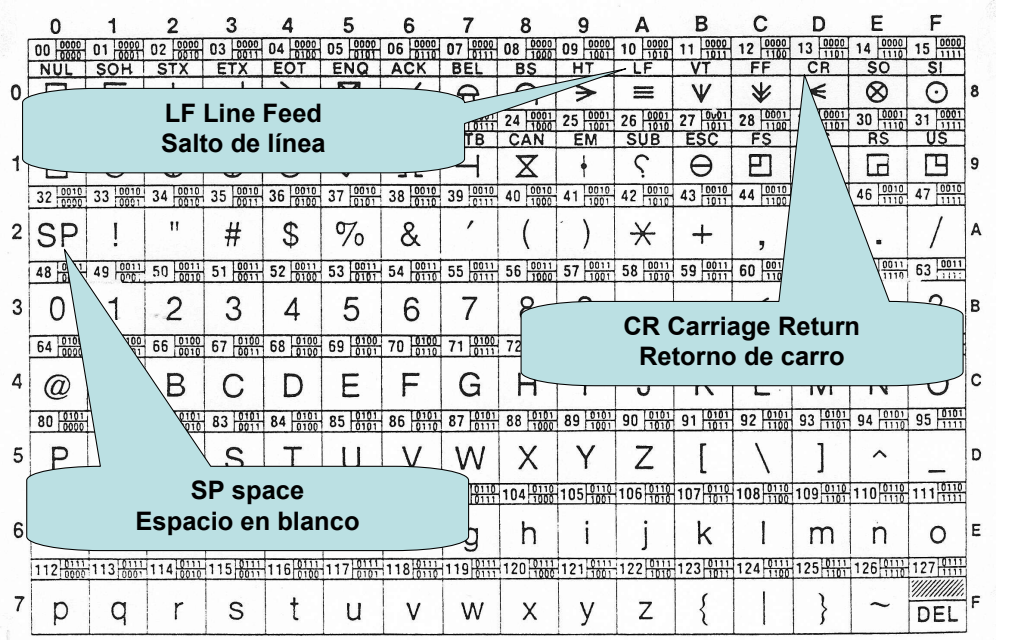
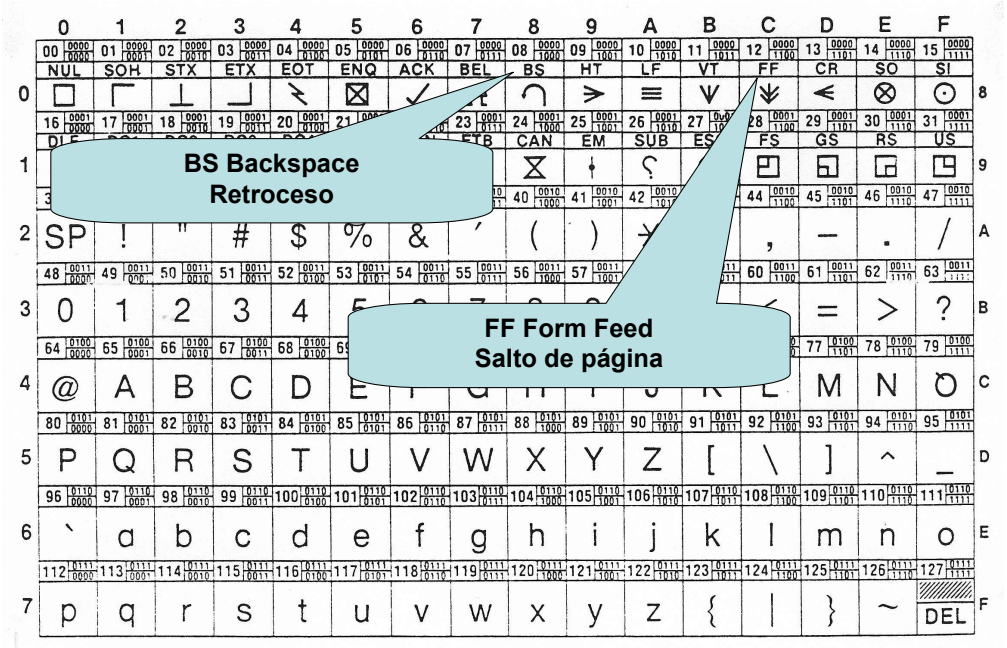


Tabla ASCII



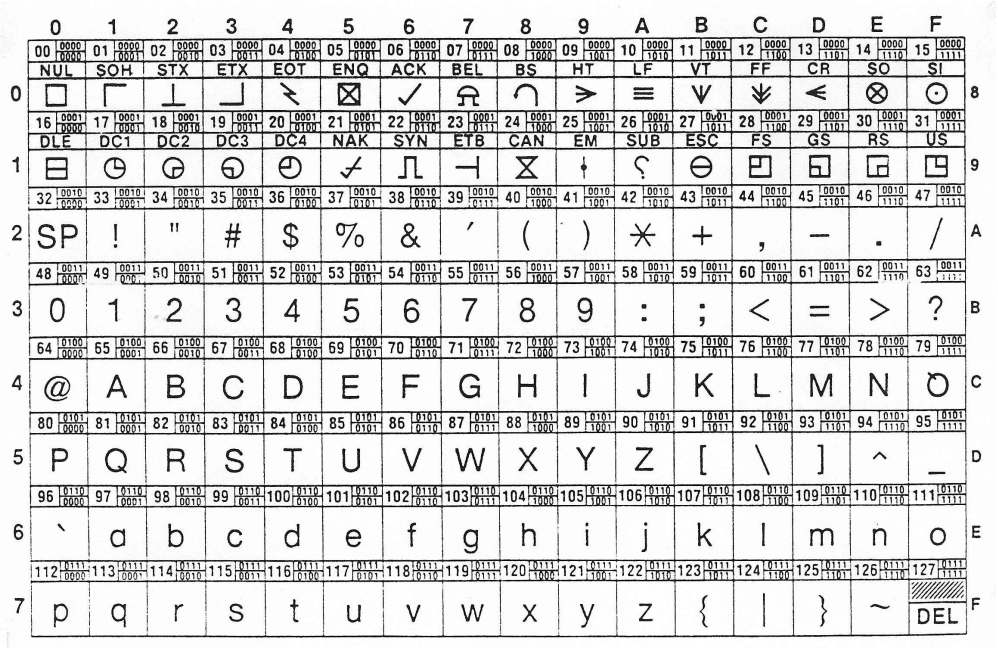
Alteradores de formato

Tabla ASCII



Alteradores de formato

Tabla ASCII



Alteradores de formato

Tabla ASCII

- **Control de transmisión:**
  - **SOH** Start Of Heading (comienzo de encabezado)
  - **STX** Start of Text (comienzo del texto)
  - **ETX** End of Text (final de texto)
  - **EOT** End Of Transmission (final de Transmisión)
  - **ENQ** ENQuiry (interrogación)
  - **ACK** Acknowledge (reconocimiento)
  - **NAK** Negative Acknowledge (reconocimiento negativo)
  - **SYN** Synchronous/idle(síncrono/parado)
  - **ETB** End of Transmission Block (final de bloque transmitido)
- **Control de formato:**
  - **BS** Back Space (retroceso de espacio)
  - **HT** horizontal Tab (Tabulación Horizontal)
  - **LF** Line Feed (avance de línea)
  - **VT** Vertical Tab (tabulación vertical)
  - **FF** Form Feed (avance de página)
  - **CR** Carriage Return (regreso del carro)

Ejemplo: