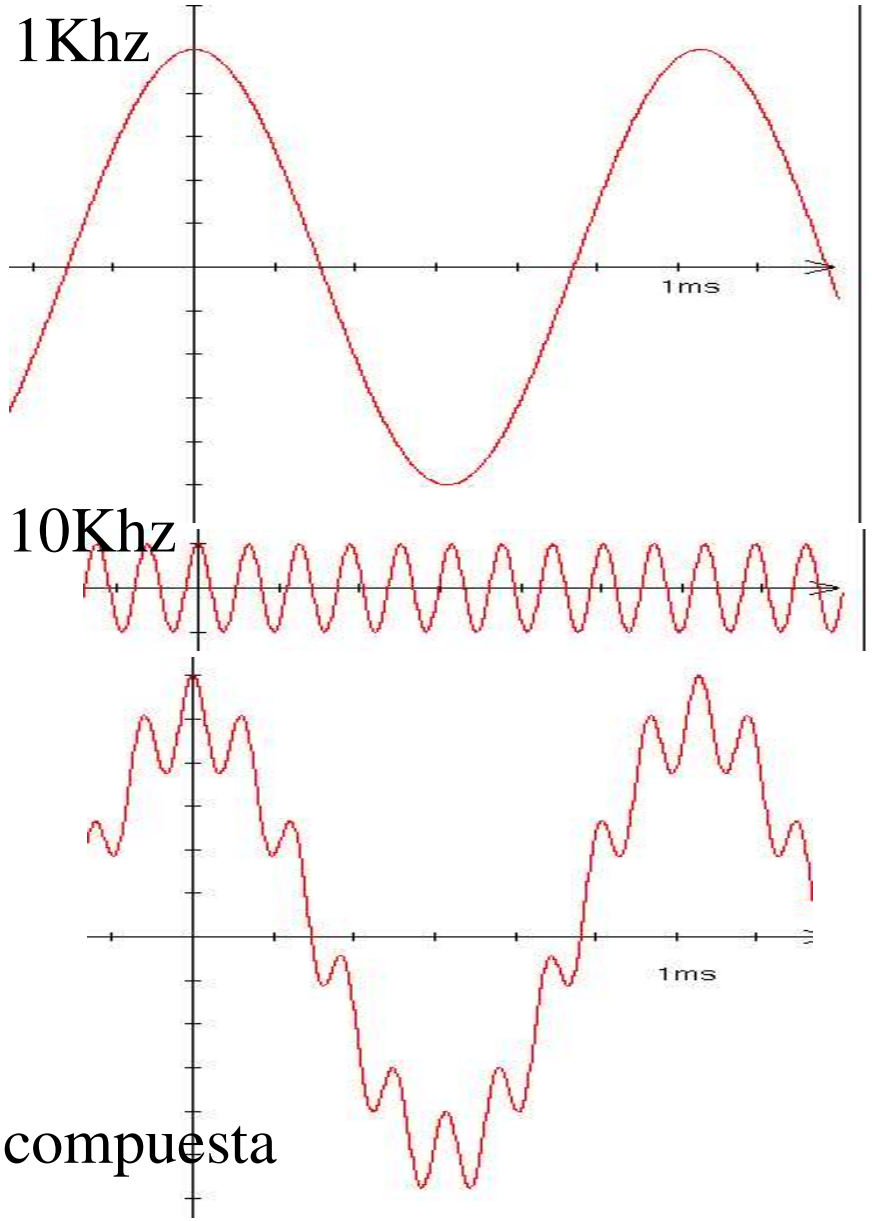
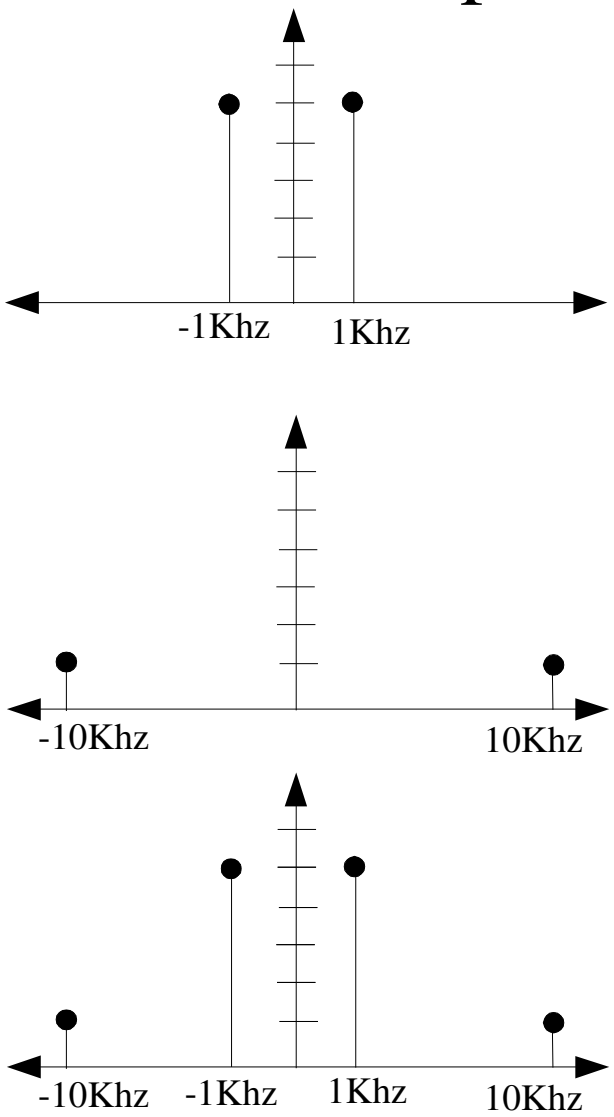
Codificación

- Moduladora analógica => modulación por impulsos
  - Objetivo: Transmisión digital de señales analógicas
  - Proceso (conversión Analógico-Digital):
    - Muestreo -> discretización en amplitud => señal discreta en el tiempo. No hay pérdida de información
    - Cuantificación -> discretización en amplitud => señal digital. Pérdida de información
    - Codificación => formato de representación binaria
  - Tipos: PAM, PWM, PPM, delta, MIC....
- Moduladora digital => codificación
  - Objetivos:
    - Reducir ancho de banda de la señal
    - Eliminar componente continua
    - Sincronización
    - Detección de errores
    - Mejorar la tasa de error
  - Tipos: bifásica, multinivel, manchester, NRZ, 5B6B, HDB3, etc.

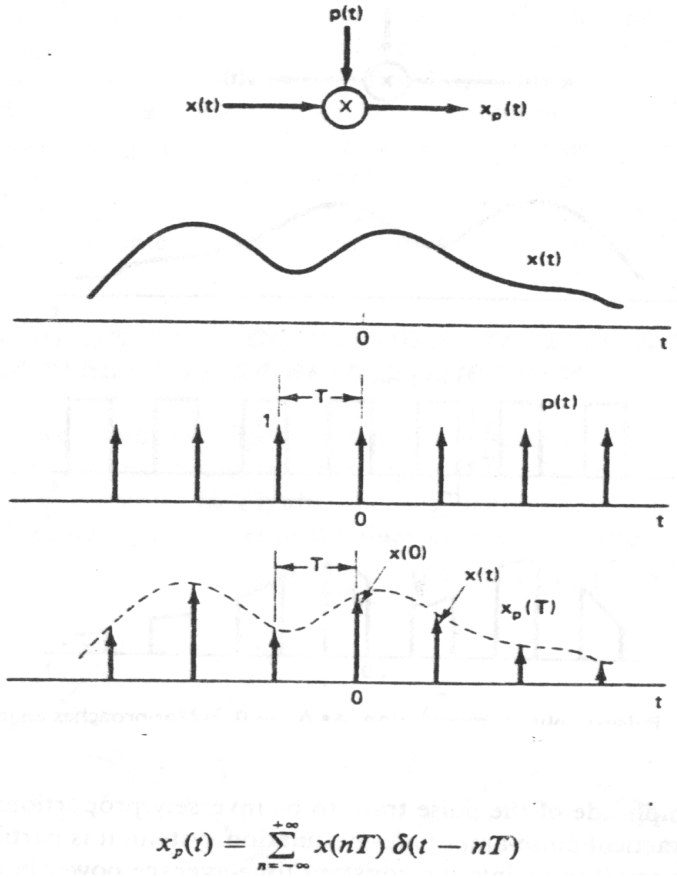
Representación tiempo-frecuencia. El espectro





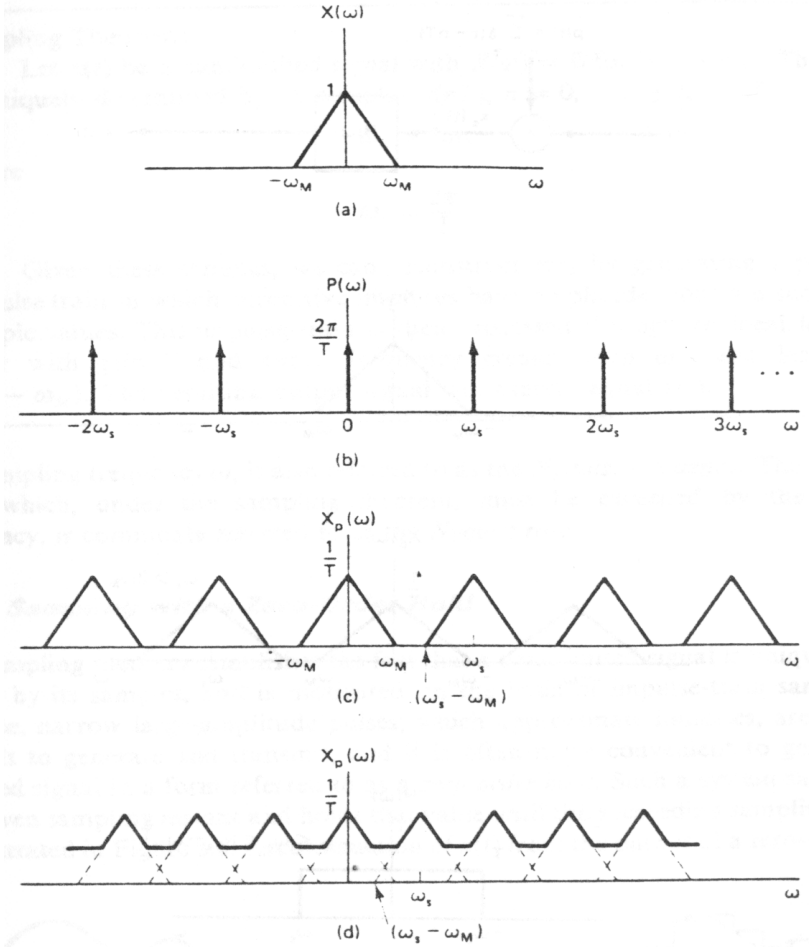
El muestreo

- Muestreo = discretizar en el tiempo señal analógica
- No se pierde información si fm ≥ 2W



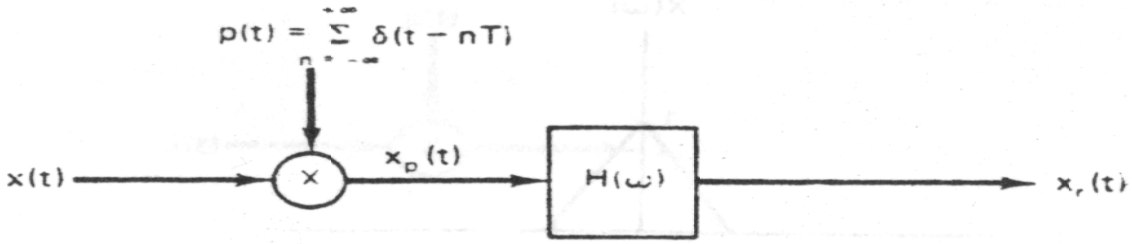
$$x_p(t) = \sum_{n=-\infty}^{+\infty} x(nT)\,\delta(t - nT)$$

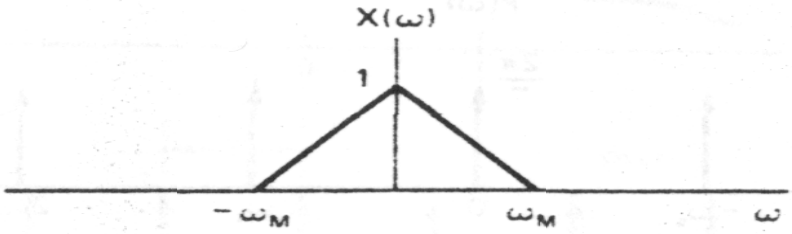
Muestreo visto en el tiempo

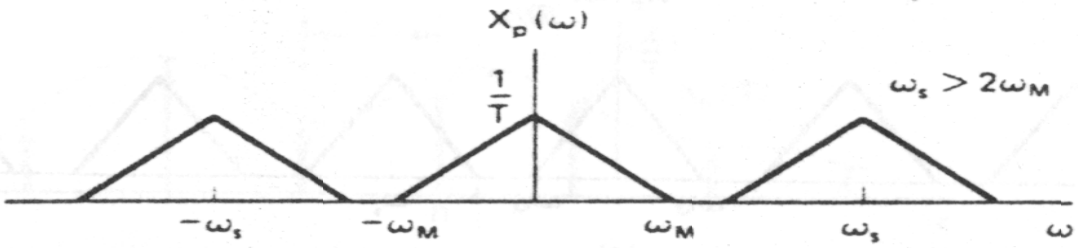


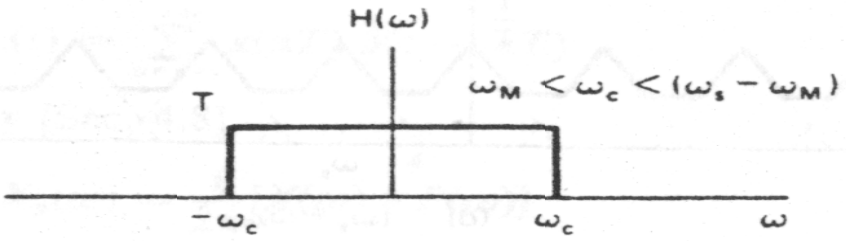
Muestreo visto en la frecuencia

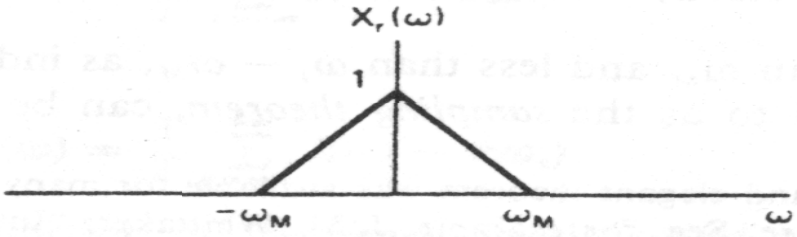
recuperación de la señal original con un filtro ideal



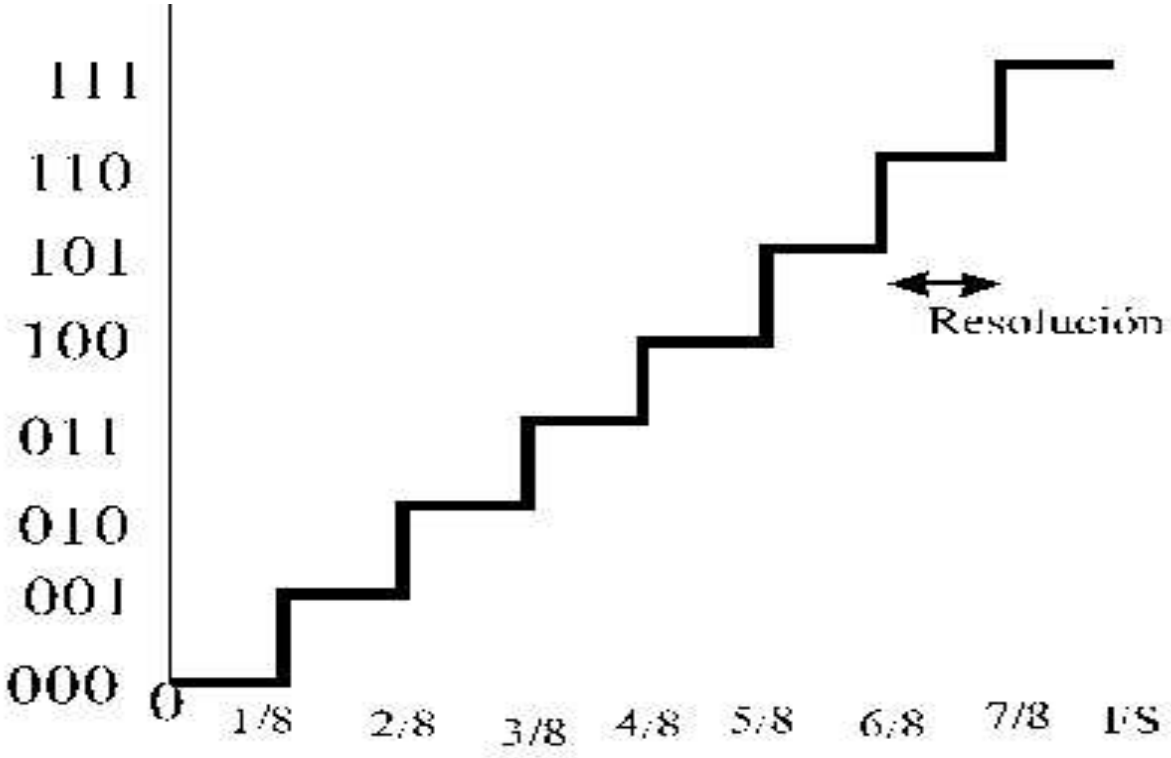








La cuantificación



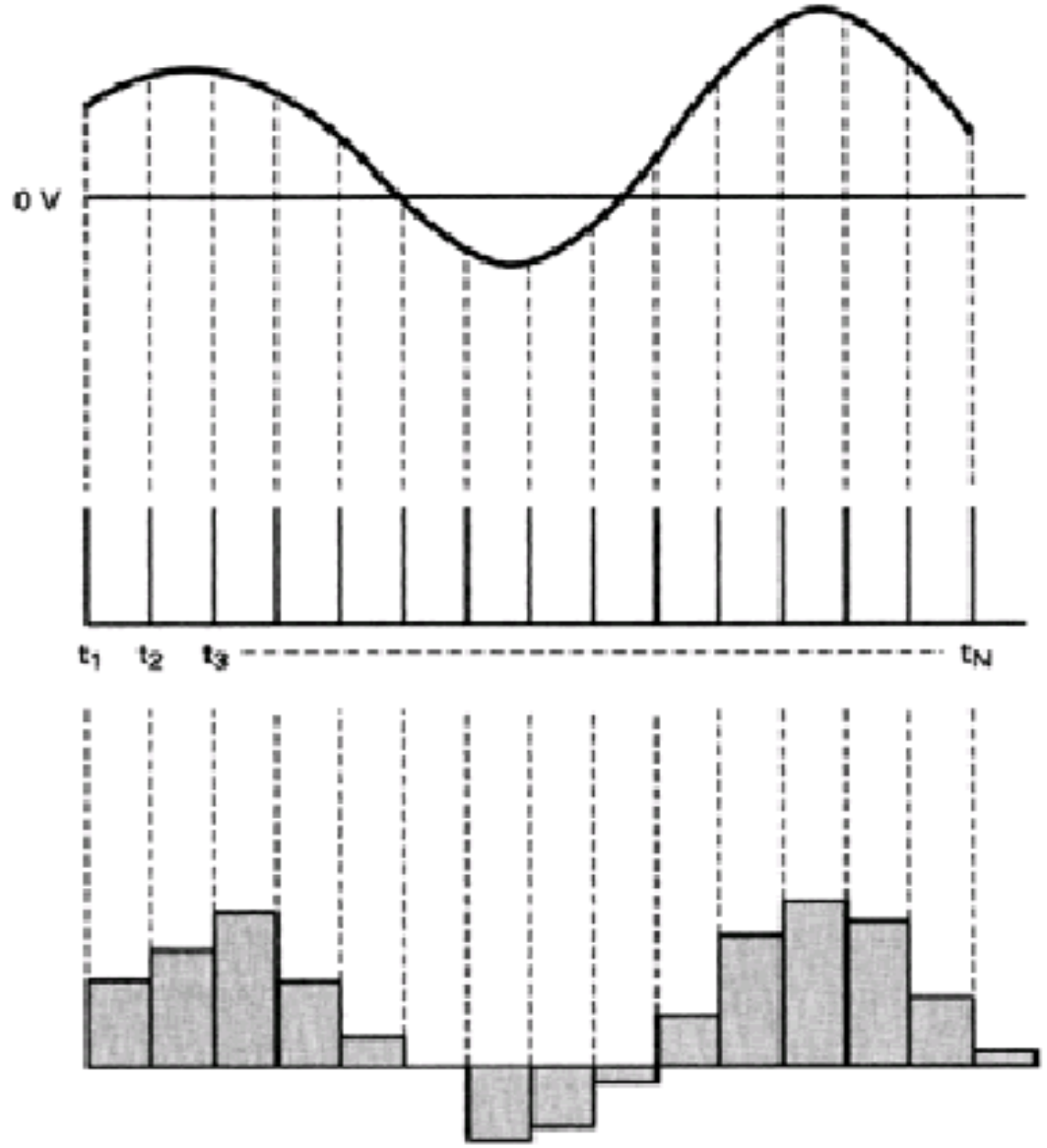
- Muestra -> cualquier amplitud
- N valores normalizados de aplitud => aproximación
  - Redondeo -> error = ±½Δ
  - Truncamiento -> error = Δ
- Codificación -> n bits, siendo $N = 2^n$

Modulación por amplitud de pulsos (PAM)

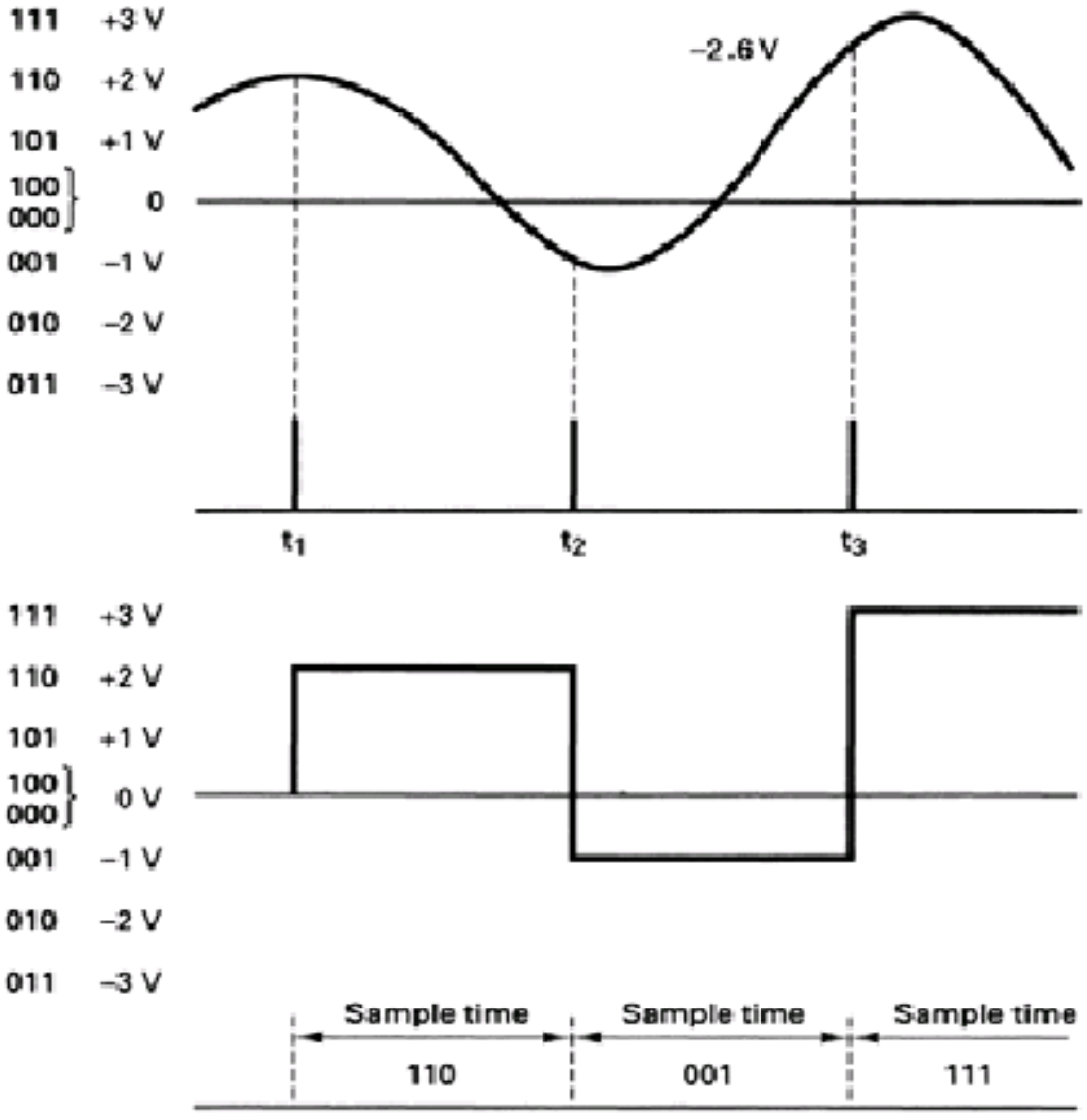
(a) input signal;

(b) Sampling signal

(c) PAM signal



Modulación por pulsos codificados (MIC o PCM)



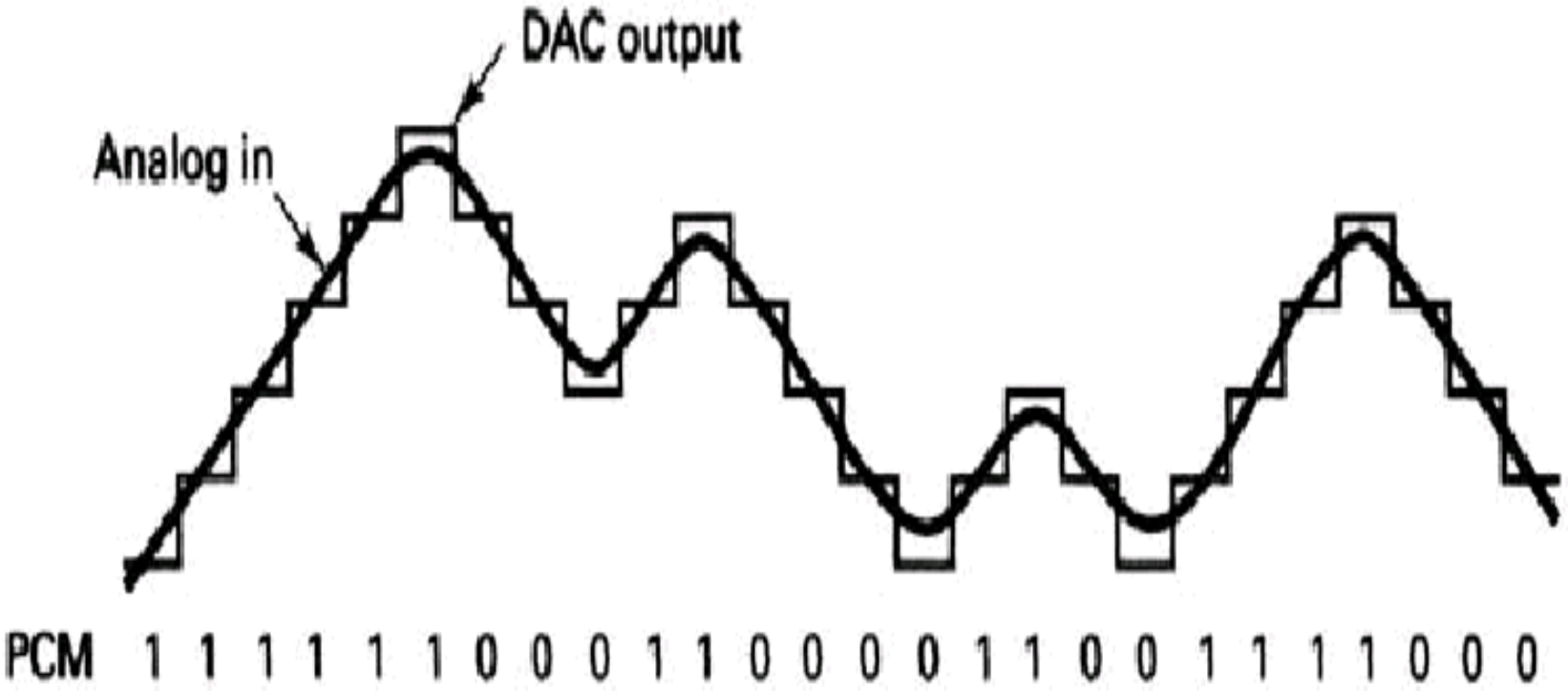
Analog input signal

Sampling signal

PAM signal

PCM code

Modulación delta (diferencial)



Modulaciones PWM y PPM

(a) analog signal;

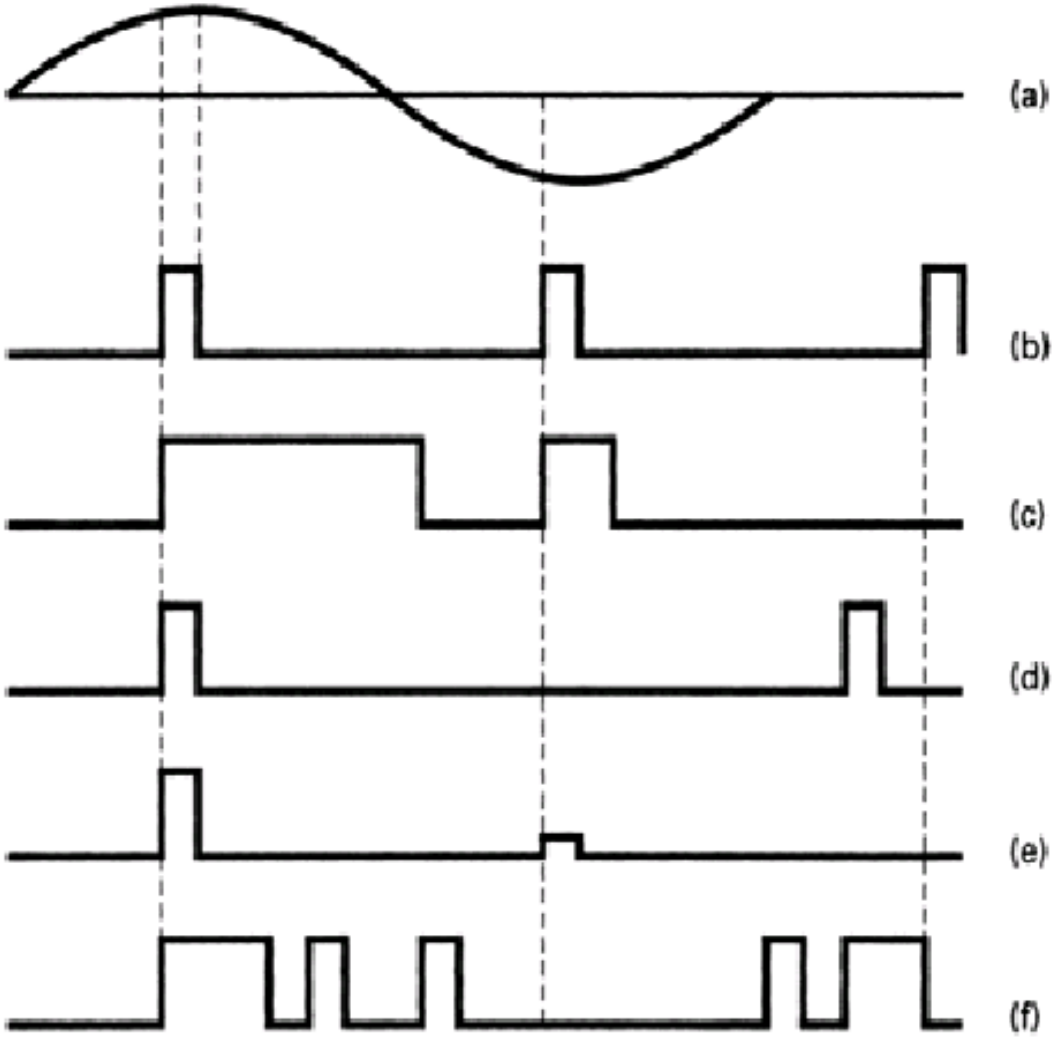
(b) sampling signal;

(c) PWM;

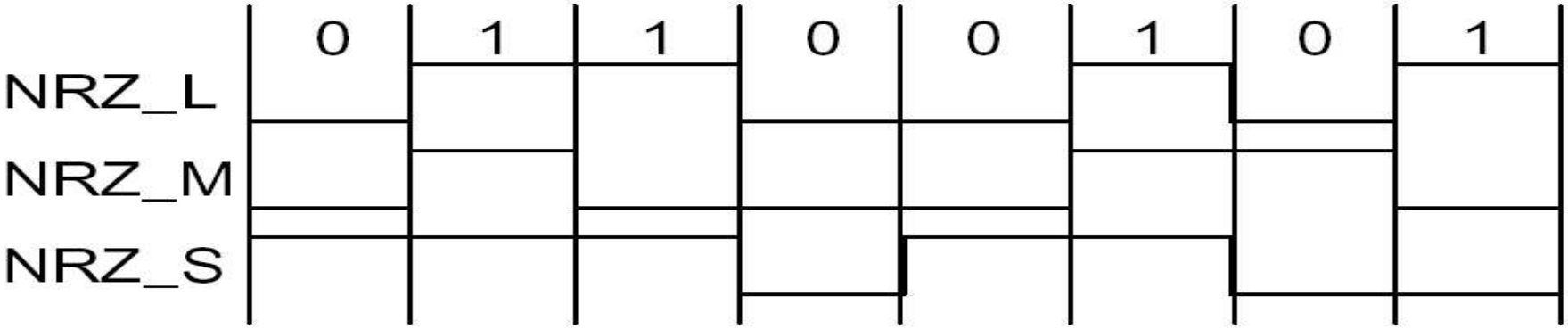
(d) PPM;

(e) PAM;

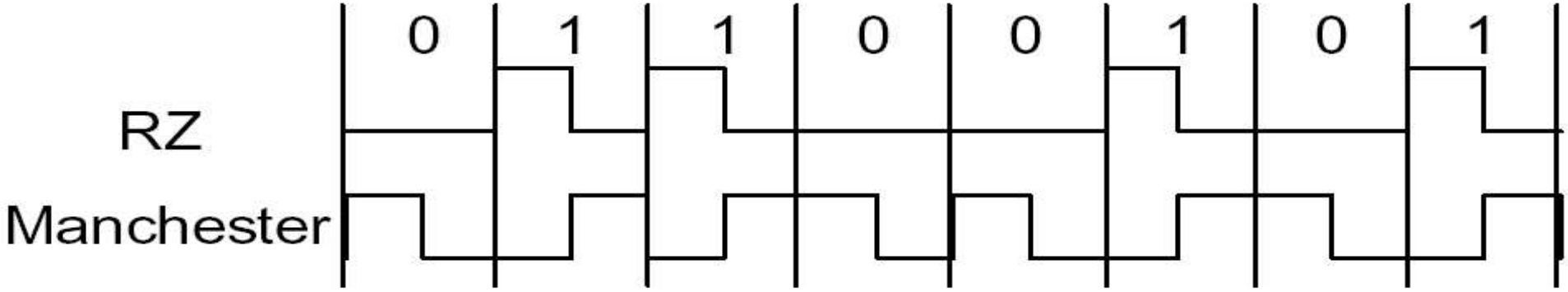
(f) PCM



Datos digitales – señales digitales

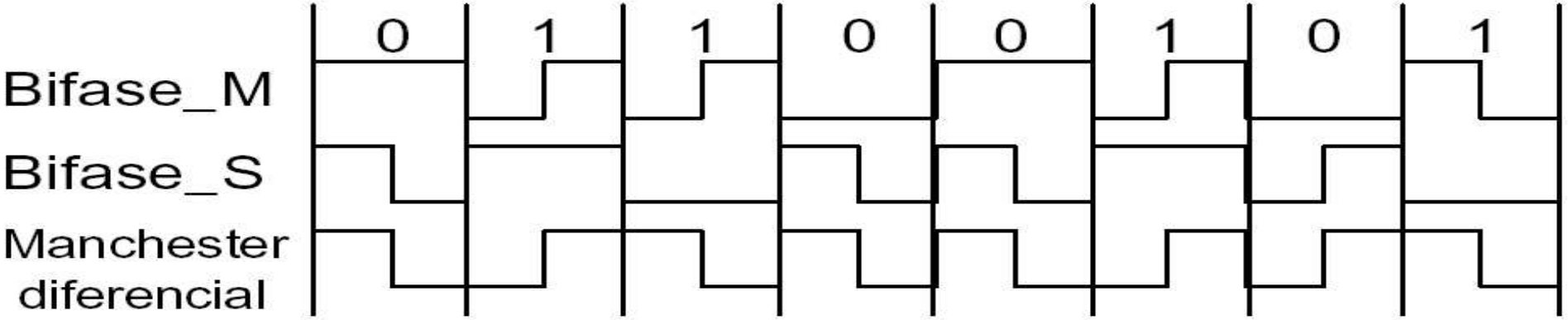


- NRZ_L = bipolar “normal”
- NRZ_M -> “1” = transición al principio del intervalo
- NRZ_S -> “0” = transición al principio del intervalo

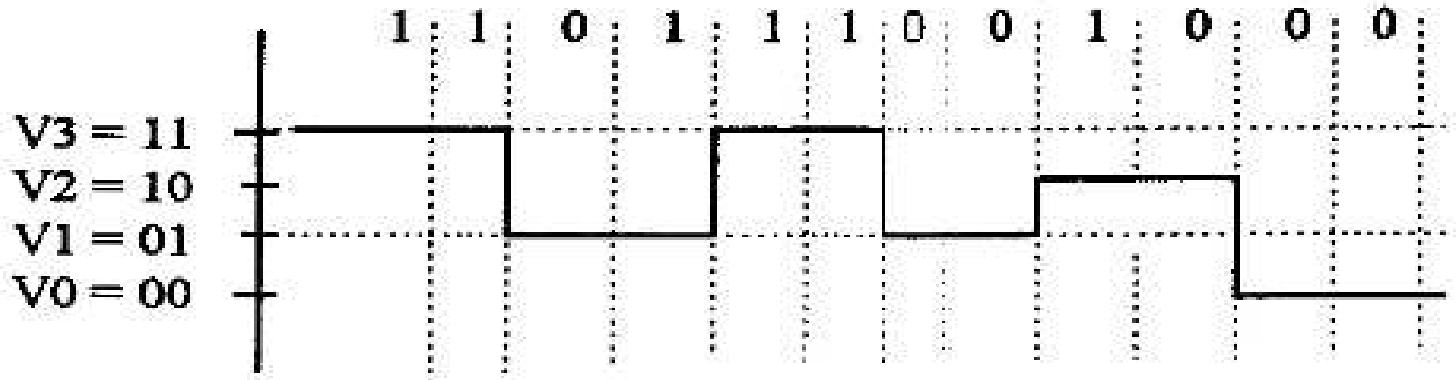


- RZ -> valor del bit en ½ periodo + retorno a cero en el otro medio
- Manchester -> flancos en el centro del bit: “1” = flanco subida, “0”=flanco bajada. Garantiza reloj. Duplica ancho de banda.

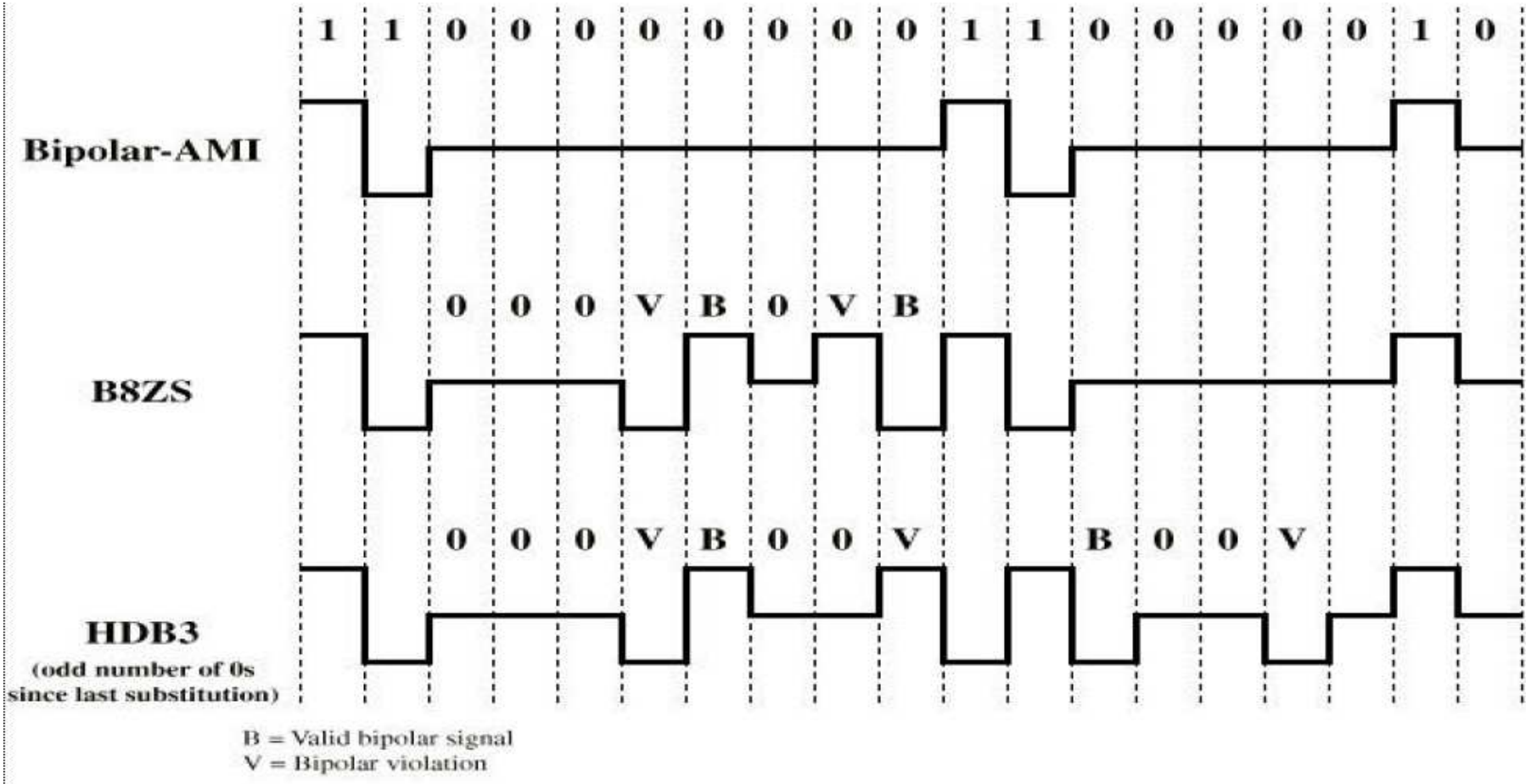
Datos digitales – señales digitales



- Bifase_M -> “1” = flanco de subida
- Bifase_S -> “0” = flanco de bajada
- Manchester diferencial -> siempre flanco en medio. “1” sin flanco al principio, “0” flanco al principio.



- Multivalente -> N niveles. Codificacion n bits por transición.



- AMI -> “0” = ausencia de señal. “1” = pulso positivo o negativo (alternados)
- B8ZS (EEUU)
  - no permite 8 “0” seguidos -> genera dos violaciones de AMI (invierte polaridad)
- HDB3 (UE y Japón)
  - No permite 4 “0” seguidos -> genera una violación de AMI

Codificación de la información

- Representación de un dígito binario (“0” o “1”) -> bit
- Representación de un rango mayor de símbolos => código:
  - Símbolos mensaje = cada uno de los símbolos representados
  - Palabras del código = cada una de las combinaciones de bits que representa a un símbolo.
  - N mensajes => como **mínimo** código n bits $N = 2^n$
  - ejemplo -> representación de los símbolos decimales (BCD)
- Fuentes de información
  - De memoria nula -> la probabilidad de cada símbolo depende sólo de ese símbolo
  - Con memoria -> la probabilidad de cada símbolo depende de los anteriores

Símbolo	código
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

- Códigos históricos:
  - Morse (telégrafo)
  - Baudot (teletipo) -> 5 bits + bit inicio + bit paada
- Códigos modernos
  - EBCDIC (8 bits) -> entornos IBM
  - ASCII (7 bits) -> normalizado ANSI e ISO

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

Source: www.asciitable.com

Códigos detectores y correctores de error

- Redundancia de un código
  - Redundancia = diferencia entre la información máxima que puede generar una fuente y la que realmente genera
  - Redundancia de un código -> uso de más bits de los “necesarios”
  - bits de código (cod. binario) > bits de información (Shannon)
  - Distancia de hamming
    - D. H. entre dos combinaciones binarias = nº de bits que hay que cambiar para pasar de una a otra.
    - D. H. de un código = D.H. mínima entre combinaciones
    - D.H. > 1 => redundancia
- Códigos detectores y correctores de error
  - Un error de n bits es detectable por un código con distancia n
  - Y corregible por un código de distancia 2n + 1

- Códigos m sobre n
  - Son códigos de m bits
  - Sólo son válidas las combinaciones que tienen n bits a 1
  - Distancia de Hamming = 2
- Control de paridad
  - Se añade un bit de paridad
  - Distancia de Hamming = 2
  - Paridad horizontal = para cada dato transmitido
  - Paridad vertical = para todos los bits de una secuancia de datos (columnas)
  - Paridad cruzada = combinación de las dos -> distancia de Hamming = 4
- Códigos cíclicos (CRC)
  - características
    - Detectan ráfagas de errores
    - Tratamiento de las series de bits como polinomios
    - Utilizan un polinomio generador para la comprobación de errores

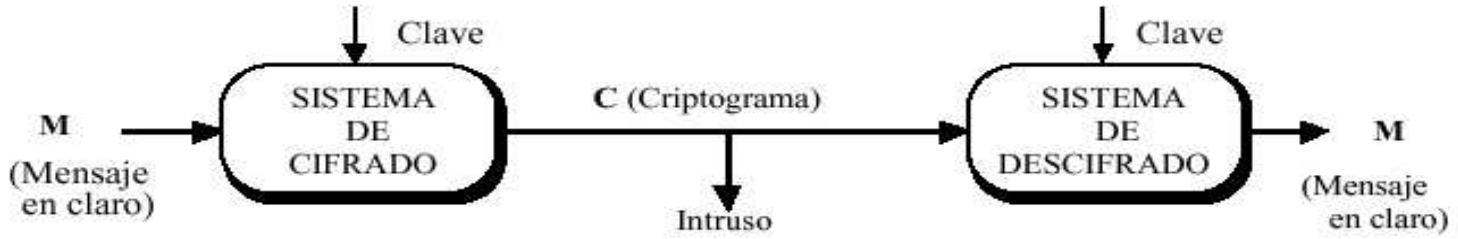
- Proceso
  - Generación
    - Se añaden al dato a transmitir tantos ceros a la derecha como el orden del polinomio generador
    - Se divide el polinomio resultante por el polinomio generador y se obtiene el resto
    - El resto se suma al dato a transmitir expandido con los ceros
  - Comprobación
    - El receptor divide el dato que le llega por el polinomio generador.
    - Si el resto es 0 no hay error
    - Si el resto no es 0 hay errores
- Polinomios cíclicos más usados
  - CRC-12 $= x^{12} + x^{11} + x^3 + x^2 + x + 1$
  - CRC-16 $= x^{16} + x^{15} + x^2 + 1$
  - CRC-CCITT $= x^{16} + x^{12} + x^5 + 1$
  - Características de los CRC16
    - Detecta 100% errores simples, y dobles
    - Detecta 100% errores en un número impar de bits
    - Detecta 100% de los paquetes con errores de longitud menor que 18 y 99'998% de los mayores

Compresión de datos

- Dos tipos de técnicas:
  - Sin pérdidas -> información almacenada = original
  - Con pérdidas -> información comprimida ≠ original
- Compresión sin pérdidas
  - Basada en eliminar la redundancia => 1bit = 1 Shannon
  - Códigos
    - Símbolos no equiprobables (p.e. letras).
    - Dependen de los anteriores.
    - Agrupaciones en bloques -> también dependen unas de otras
    - Ejemplo: “ME LLEVO EL PARAGUAS PORQUE ESTA LLOVIENDO”
  - Tipos:
    - Compresores estadísticos -> basados en la probabilidad de un símbolo: codificación con nº de bits menor según probabilidad
    - Compresores basados en diccionario -> estudian secuencias repetidas.

- Compresión con pérdidas
  - En sistemas donde se pueden tolerar diferencias (p.e. audio)
  - Basadas en:
    - Medidas de la percepción -> puede no notarse diferencia
    - Filtrado -> selección del espectro donde está la mayor parte de la potencia.
    - Redundancia temporal -> “lentitud” de variación en la imagen/señal
    - Uso de compresión sin pérdidas
- Ejemplos (algoritmos):
  - Sin pérdidas
    - Estadísticos
      - Shannon-Fano (no óptimo): Se usa en ZIP
      - Huffman (óptimo): Se usa en LZH, BZIP2
    - Basados en diccionario
      - Familia LZ78 (Lempel-Ziv 78): LZW, LZC (compress), GIF, V42bis
      - Familia LZ77 (Lempel-Ziv 77): ZIP, LZH
  - Con pérdidas: MPEG (audio), JPEG (imagen), MPEG-1, MPEG-2, MPEG-4 (video)

Cifrado de datos

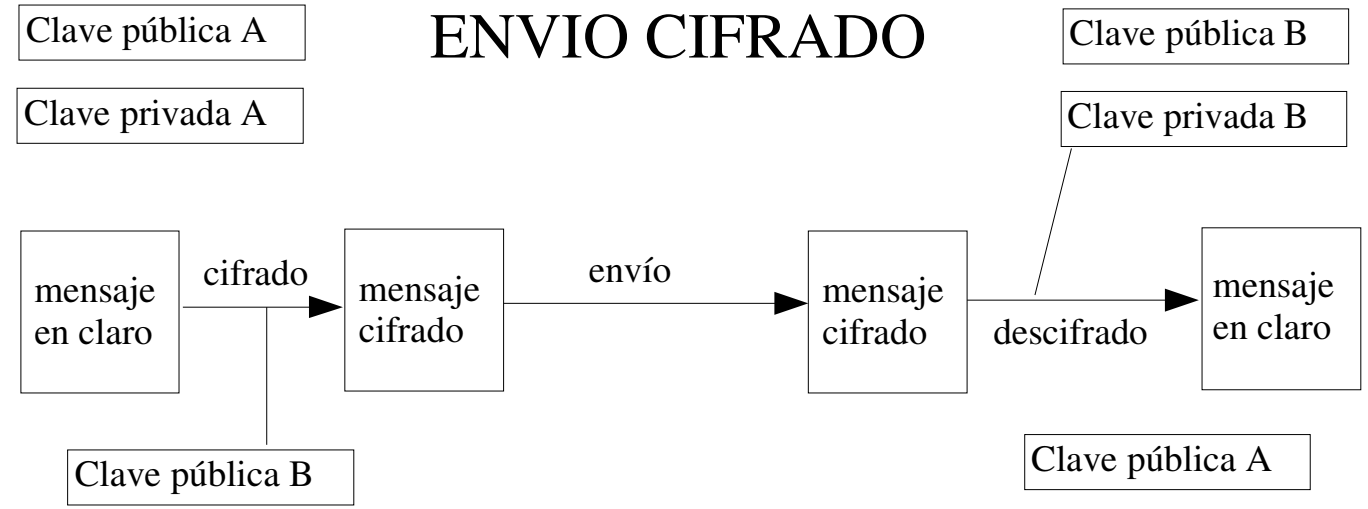


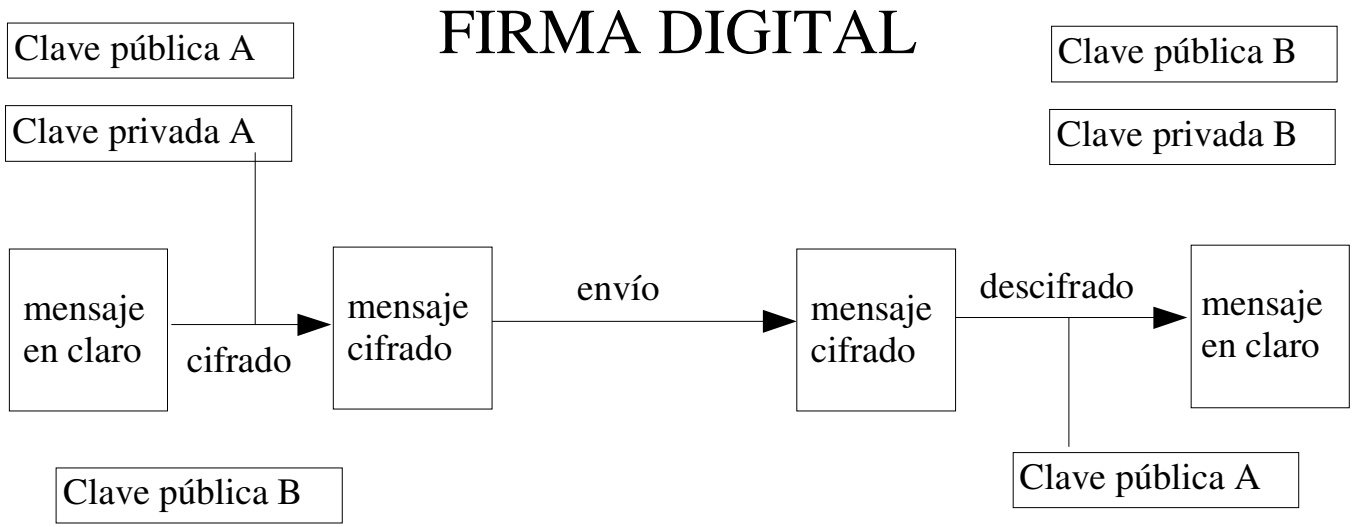
Esquema de transmisión segura de un mensaje

- Claves iguales -> Algoritmos simétricos (DES, IDEA, AES)
- Claves diferentes -> Algoritmos asimétricos (RSA, D-H, PKCS)
- Data Encryption Standard (DES)
  - Estándar americano de 1977
  - clave de 56 bits sobre bloques de datos de 64 bits-> con la tecnología de la época se tardaban 2200 años en romper la clave, hoy 3 días.
- International Data Encryption Algorithm (IDEA)
  - Tuvo su aparición en 1992.
  - Considerado por muchos el mejor y más seguro algoritmo simétrico disponible en la actualidad.
  - Trabaja con bloques de 64 bits de longitud, igual que el DES, pero emplea una clave de 128 bits.
  - Se usa el mismo algoritmo tanto para cifrar como para descifrar.

- Advanced Encryption Standard (AES)
  - Publicado el 2 de Octubre de 2000.
  - Se intuye que substituirá al actual D.E.S.
  - El tamaño de clave debe ser de, al menos, 128, 192 y 256 bits (debe admitir los tres), y el tamaño de bloque de cifrado debe ser de 128 bits.
  - Los productos que incorporen AES podrán ser exportados fuera de EE.UU.
- Algoritmos asimétricos
  - Cada usuario tiene un par de claves:
    - Clave privada -> debe ser secreta
    - Clave pública -> puede difundirse a todo el mundo.
  - Sirve para que:
    - Otros usuarios le envien documentación cifrada
    - El propietario de la clave envíe documentación “firmada”

Funcionamiento de un sistema de doble clave





Sistemas de doble clave

- Propiedades
  - Algoritmos asimétricos
  - Válidos para encriptar y firmar
  - Tiempos de cálculo muy altos => sólo se firma un extracto.
  - Necesidad de autoridades certificadoras para las firmas:
    - Fábrica Nacional de Moneda y Timbre
    - Agencia de Certificación Electrónica
    - Verisign
    - .....
- Algoritmos de cifrado
  - RSA
    - Basado en la utilización de un número producto de dos números primos grandes => producto=clave públlica, factorización=clave privada.
    - Claves de tamaño variable, típicos 512 o 1024bits.Bloques variables, menores que la clave
    - Muy seguro. Se usa en ssh

Sistemas de doble clave

- Diffie-Hellman
  - Algoritmo histórico (1976)
  - Precursor de RSA
  - Es vulnerable en algunos supuestos
- PKCS (Publick-key Cryptografy Standards)
  - 15 estándads basados en RSA.
- Funciones de hash
  - Son funciones unidereccionales de resumen -> generan una cadena de resumen de un documento (“no puede haber” dos cadenas de resumen iguales)
  - MD5 (128 bits), SHA-1(160 bits), RIPEMD(160 bits), etc.
- Protocolos de seguridad
  - Utilizan funciones de hash y sistemas de doble clave para trasnferir información de forma segura
  - PGP, SSL, SET, IPSEC, etc.