

IC 323 -Comunicación de Datos

Unidad N°6: Teoría de la Información y Codificación

Detección y Corrección de Errores



Cuatro preguntas ante un canal ruidoso

01

Medir

¿Cuántos errores hay? Tasa de error de bit (BER).

02

Detectar

¿Llegó bien el bloque? Paridad, checksum y CRC.

03

Retransmitir

Pedir de nuevo el bloque dañado: ARQ.

04

Corregir

Reparar en el receptor: FEC y códigos de Hamming.



Tasa de error de bit (BER)

$$\text{BER} = \frac{\text{bits recibidos con error}}{\text{bits transmitidos}}$$

Ejemplo: 1 bit errado en 100.000 → **BER = 10^{-5}**

Es una probabilidad

Medida sobre muchos bits, estima Pb: la probabilidad de que un bit cualquiera llegue invertido.

Empeora con la velocidad

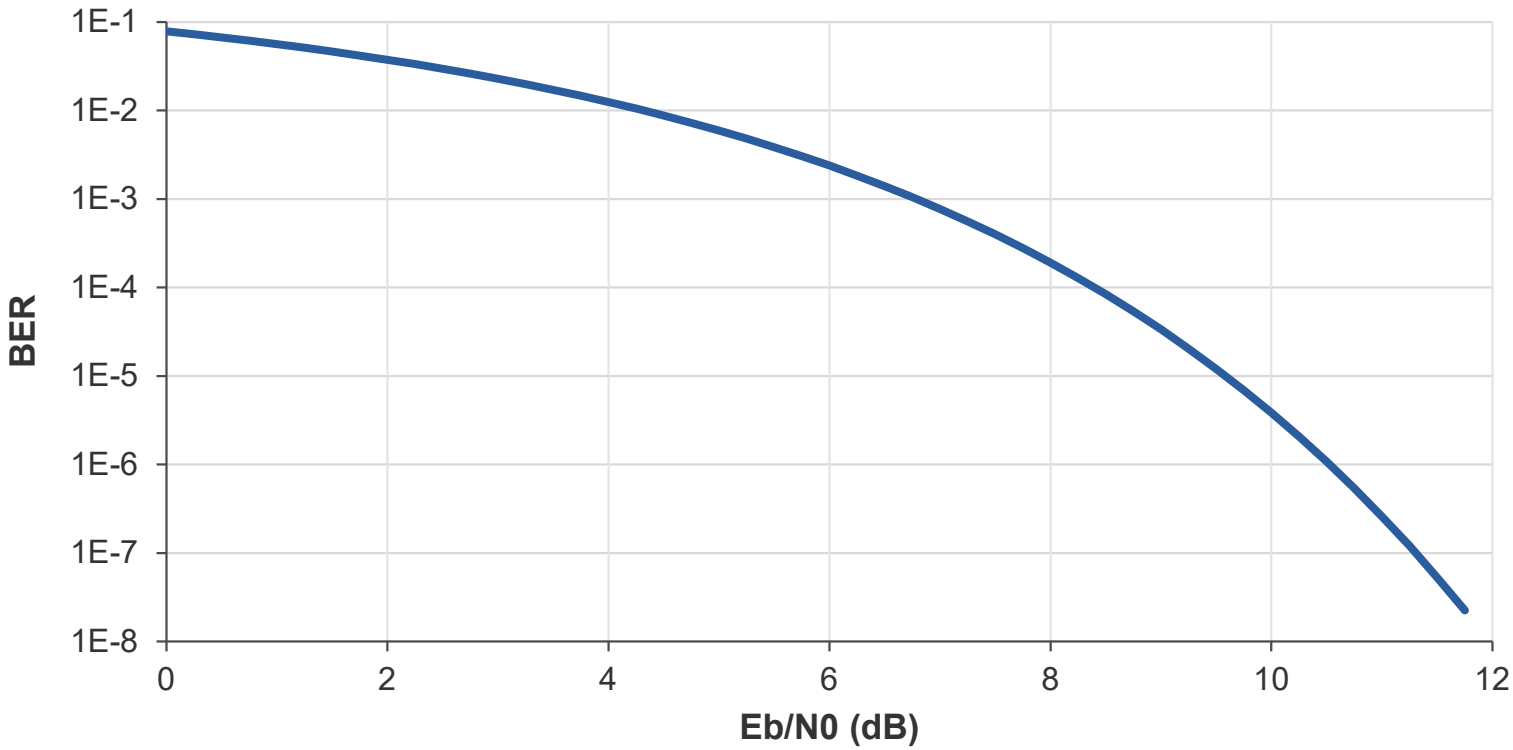
A mayor tasa Rb, menos energía por bit: Eb = S / Rb.

Mejora con la relación S/N

Más potencia de señal frente al ruido, menos errores. En digital se usa Eb/N0.



BER frente a Eb/N0



- **BPSK en canal AWGN:** Eje vertical en escala logarítmica.
- En la zona útil, **1 dB más** reduce la BER casi un orden de magnitud.
- Un código de canal desplaza la curva hacia la izquierda: es la **ganancia de codificación**.



Ejemplo: errores en 8 paquetes de 512 bytes

BER = $2 \cdot 10^{-4}$ ¿Cuántos errores pueden esperarse en promedio?

32.768

bits transmitidos

$8 \times 512 \times 8$

≈ 6,55

errores en promedio

$32.768 \times 2 \cdot 10^{-4}$

44 %

de paquetes sin errores

$(1 - 2 \cdot 10^{-4})^{4096} \approx 0,44$

Más de la mitad de los paquetes llega con al menos un bit errado: el receptor **necesita** una forma de detectarlo.



Errores de bit simple y en ráfaga

Error simple

enviado

`010001010100`

recibido

`01001`**`1`**`1010100`

Cambia un único bit del bloque. Típico del ruido térmico, aleatorio e independiente.

Ráfaga de longitud 5

enviado

`010001000100`

recibido

`010`**`111`**`0`**`1`**`0100`

Se mide del primer al último bit errado; los intermedios pueden estar bien. Típica del ruido impulsivo: cuanto mayor la velocidad, más bits abarca.



La idea central: redundancia controlada

Detectar o corregir errores exige agregar bits que no llevan información nueva, pero permiten verificar el bloque.

k bits de datos	r de control

Longitud de la palabra

$$n = k + r$$

Tasa del código

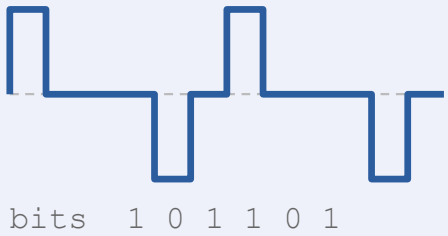
$$R = k / n$$

Repetir el mensaje completo (R = 1/2) es la forma más simple y la menos eficiente. Los códigos que siguen buscan más protección con menos bits de control.



Detección por reglas de la señal y del código

Violación bipolar (AMI)



Los unos alternan polaridad. Dos unos seguidos con la **misma** polaridad revelan un error.

Códigos de enrejado

Solo ciertas secuencias de símbolos son válidas: las que siguen un camino del enrejado (trellis).

Si llega una combinación que ningún camino produce, hubo un error. El decodificador de Viterbi además elige el camino válido más cercano.

Código de cuenta exacta

Cada símbolo usa 7 bits con exactamente 3 unos: C(7,3) = 35 combinaciones para letras, números y signos.

`1011000`

3 unos: válido

`1111000`

4 unos: error



Bit de paridad

Paridad par: el bit extra deja un total par de unos.

Paridad impar: el bit extra deja un total impar de unos.

Ejemplo: carácter ‘C’	datos | paridad	unos	resultado
ASCII original	1000011 |	3	—
con paridad par	1000011 | 1	4	se transmite
llega con 1 error	1010011 | 1	5	error detectado
llega con 2 errores	1010001 | 1	4	no se detecta

Detecta cualquier cantidad **impar** de errores; un número par deja la paridad intacta.



Paridad bidimensional

Bloque recibido (paridad par)

c1	c2	c3	c4	par.	verif.
1	0	1	1	1	ok
0	1	0	0	0	falla
1	1	0	0	0	ok
0	1	0	1	0	ok
0	1	0	0	1	
ok	ok	falla	ok		

Fila y columna

Cada fila y cada columna llevan su propio bit de paridad (en azul).

Ubica y corrige

Un error simple hace fallar una fila y una columna: el cruce señala el bit, que se corrige invirtiéndolo.

Su límite

Detecta todos los errores de 1, 2 y 3 bits; falla con 4 en las esquinas de un rectángulo.



CRC: verificación por redundancia cíclica

Los bits son un polinomio

110101 ↔ $x^5 + x^4 + x^2 + 1$. Emisor y receptor acuerdan un generador G(x) de grado r.

Emisor

Agrega r ceros, divide en módulo 2 por G(x) y anexa el resto R(x) como CRC.

Receptor

Divide la trama recibida por G(x): resto 0, se acepta; resto distinto de 0, hay error.

Con un G(x) bien elegido detecta

- todos los errores simples y dobles
- toda cantidad impar de errores, si (x + 1) divide a G(x)
- toda ráfaga de longitud ≤ r
- casi todas las ráfagas más largas (≈ $1 - 2^{-r}$)

CRC-16-CCITT
$x^{16} + x^{12} + x^5 + 1$ · HDLC, Bluetooth

CRC-32
grado 32 · Ethernet, Wi-Fi, ZIP, PNG



CRC paso a paso: M = 110101, G = 1011

```
110101000
1011
011001000
 1011
001111000
  1011
000100000
   1011
00000 1100
     1011
000000111
```

1. G = 1011 ↔ $x^3 + x + 1$, grado r = 3: se agregan 3 ceros → 110101**000**.

2. Donde el bit líder es 1, se hace XOR con G (en azul), sin acarreos ni préstamos.

3. El resto es **111**: es el CRC que se anexa al mensaje.

110101 111

4. El receptor divide 110101111 entre 1011: resto 000, trama aceptada.



CRC paso a paso: M = 110101, G = 1011

```
110101000
1011
011001000
 1011
001111000
  1011
000100000
   1011
00000 1100
     1011
000000111
```

1. G = 1011 ↔ $x^3 + x + 1$, grado r = 3: se agregan 3 ceros → 110101**000**.

2. Donde el bit líder es 1, se hace XOR con G (en azul), sin acarreos ni préstamos.

3. El resto es **111**: es el CRC que se anexa al mensaje.

110101 111

4. El receptor divide 110101111 entre 1011: resto 000, trama aceptada.



Métodos de detección comparados

Método	Bits de control	Qué detecta	Dónde se usa
Paridad simple	1 por carácter	cualquier cantidad impar de errores	UART, memorias
Paridad 2D	1 por fila y por columna	todo error de 1, 2 o 3 bits; corrige 1	bloques de caracteres
Checksum de Internet	16 por segmento	la mayoría de los errores aleatorios; no ve palabras reordenadas	IPv4, UDP, TCP
CRC-32	32 por trama	toda ráfaga de hasta 32 bits y casi todas las demás	Ethernet, Wi-Fi, ZIP

Más bits de control permiten más cobertura. El CRC domina en la capa de enlace porque se calcula en hardware con XOR y registros de desplazamiento.



Dos estrategias ante un error

	ARQ: solicitud automática de repetición	FEC: corrección de errores hacia adelante
El receptor…	detecta y pide la retransmisión	detecta y corrige por sí mismo
Canal de retorno	imprescindible	no hace falta
Bits extra	pocos; el costo son las retransmisiones	fijos, siempre presentes
Latencia	variable: un viaje de ida y vuelta por error	constante
Conviene en	retardo corto y BER baja: TCP, Wi-Fi, HDLC	enlaces largos, difusión o tiempo real: satélite, TV digital, sondas espaciales

Híbrido (HARQ): FEC corrige los errores frecuentes y la retransmisión cubre el resto. Es el esquema de LTE y 5G.



Tres variantes de ARQ

Parada y espera

Envía una trama y espera su ACK; ante NAK o fin de temporizador, la repite.

A favor: lógica mínima, un solo buffer.

En contra: el canal queda ocioso mientras espera.

Retroceso N

Envía hasta N tramas sin esperar. Ante un error, repite esa trama y todas las siguientes.

A favor: el receptor no necesita reordenar.

En contra: reenvía tramas que habían llegado bien.

Repetición selectiva

Solo se repiten las tramas con error; el receptor guarda y reordena las demás.

A favor: la mayor eficiencia.

En contra: buffers y lógica en ambos extremos.

Eficiencia de parada y espera sin errores: **U = 1 / (1 + 2a)**
con a = tiempo de propagación / tiempo de trama



Distancia de Hamming: cuánto puede un código

Cantidad de posiciones en que difieren dos palabras.

```
1  0  1  1  0
1  1  1  0  0
```

d = 2

La **distancia mínima** dmín es la menor entre todos los pares de palabras válidas del código.

Detecta hasta

dmín − 1 errores

Corrige hasta

⌊(dmín − 1) / 2⌋ errores

Paridad: dmín = 2 → detecta 1, no corrige.

Repetición {000, 111}: dmín = 3 → detecta 2, corrige 1 por mayoría (010 → 000).



Código de Hamming (7,4): corrige un error

posición	1	2	3	4	5	6	7
función	p1	p2	d1	p4	d2	d3	d4
enviado	0	1	1	0	0	1	1
recibido	0	1	1	0	0	0	1

Dato **1011**. Cada paridad cubre las posiciones que la incluyen en binario:

p1 → 1, 3, 5, 7
p2 → 2, 3, 6, 7
p4 → 4, 5, 6, 7

s1 = 0⊕1⊕0⊕1 = 0
s2 = 1⊕1⊕0⊕1 = 1
s4 = 0⊕0⊕0⊕1 = 1

Síndrome s4 s2 s1 = 110 = **6**: se invierte el bit 6.

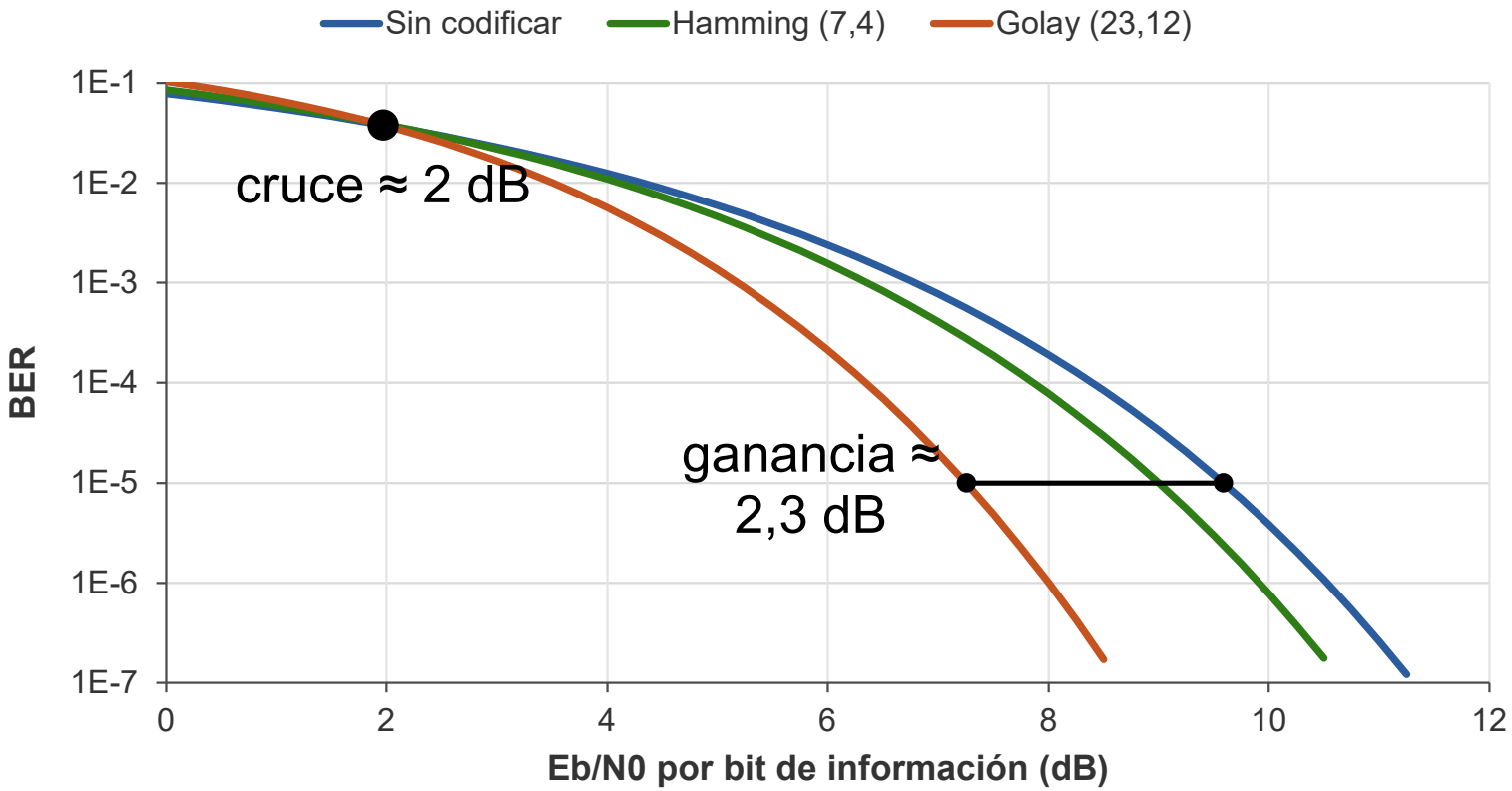


Códigos FEC en sistemas actuales

Código	Principio	Dónde se usa
Reed-Solomon	bloque sobre símbolos; fuerte ante ráfagas	CD, DVD, códigos QR, DVB
Convolucional + Viterbi	memoria de bits previos; enrejado	GSM, Wi-Fi 802.11a/g, sondas espaciales
Turbo	dos códigos concatenados, decodificación iterativa	3G (UMTS) y 4G (LTE)
LDPC	matriz de paridad dispersa, iterativo	5G (datos), Wi-Fi 6, DVB-S2
Polar	polarización del canal	5G (canales de control)

Turbo y LDPC trabajan a menos de 1 dB del límite de Shannon: la teoría de 1948 dejó de ser solo un techo teórico.

Ganancia de codificación



- **Misma BER con menos Eb/N0:** la curva codificada queda a la izquierda.
- **Cada bit codificado lleva menos energía (Ec = R·Eb):** por debajo del cruce (≈ 2 dB) codificar empeora la BER.
- **El costo:** más ancho de banda, complejidad y latencia.



Codificación de fuente frente a codificación de canal



De fuente: quita redundancia

Extrae la información esencial y la representa con la menor cantidad de bits, para almacenarla o transmitirla de forma eficiente. Ej.: Huffman, JPEG, MP3.

De canal: agrega redundancia

Suma bits de control elegidos para detectar o corregir los errores que introduce el ruido del canal. Ej.: CRC, Hamming, LDPC.

Primero se comprime, después se protege.



Para practicar

1

Un enlace de 100 Mb/s tiene BER = 10^{-6}. ¿Cuántos errores por segundo se esperan en promedio?

2

Calcule el bit de paridad impar para el carácter ‘A’ = 1000001.

3

Se recibe la palabra Hamming (7,4) 1011011. ¿Hay error? ¿Cuál es el dato de 4 bits?

4

Calcule el CRC de M = 1101011011 con $G(x) = x^4 + x + 1$ y escriba la trama transmitida.



Resumiendo...

01 La BER es una probabilidad: aun pequeña, alcanza a la mayoría de los paquetes largos.

02 Detectar cuesta pocos bits (CRC); corregir cuesta más (FEC), pero evita retransmitir.

03 La distancia mínima del código fija cuántos errores puede detectar y corregir.