

# Algorismes de detecció d'errors

Teòric – Pràctic 5

# Algorismes de detecció d'errors

- Quan es transmet informació per una línia s'inclou informació per detectar possibles errors.
- En base a les dades rebudes el detector estableix (amb una alta probabilitat) si hi ha hagut errors
- Hi ha dos tipus d'errors:
  - Errors de bit, que es mesuren amb el BER (bit error rate)
  - Errors de ràfega que modifiquen grups de bits

# Algorismes de detecció d'errors

- Si tenim un BER (Bit Error Rate) de  $10^{-3}$  significa que, en mitjana, un de cada 1000 bits serà erroni.
- Exemple:
  - Es transmeten bytes individuals amb un bit d'inici i un de final (10 bits) amb un BER de  $10^{-3}$
  - La probabilitat d'error en un bit serà  $P_{\text{err}} = 10^{-3}$
  - La probabilitat de que el bit sigui correcte serà

$$P_{\text{success}} = 1 - P_{\text{err}} = 1 - 10^{-3}$$

# Algorismes de detecció d'errors

- Si el caràcter és de 10 bits la probabilitat de que el caràcter sigui correcte serà:

$$P_{\text{success\_caràcter}} = P_{\text{success\_bit0}} \times P_{\text{success\_bit2}} \times \dots \times P_{\text{success\_bit9}}$$

$$P_{\text{success\_caràcter}} = (1 - P_{\text{err}})^{10}$$

- Per tant la probabilitat d'error del caràcter serà:

$$P = 1 - (1 - P_{\text{err}})^{10} \approx 10^{-2}$$

- Si transmetem 125 bytes, és molt probable que un bit de cada trama sigui erroni

# Algorismes de detecció d'errors

- Una probabilitat d'error de 1 bit en cada trama és molt elevada, per tant s'ha de reduir la longitud de trama de forma que ajusti el BER
- És important establir els tipus d'errors que afecten la línia
- Els diferents algorismes per a detectar-los permeten identificar diferents tipus d'errors
- En funció de la línia s'escollirà un algoritme o un altre

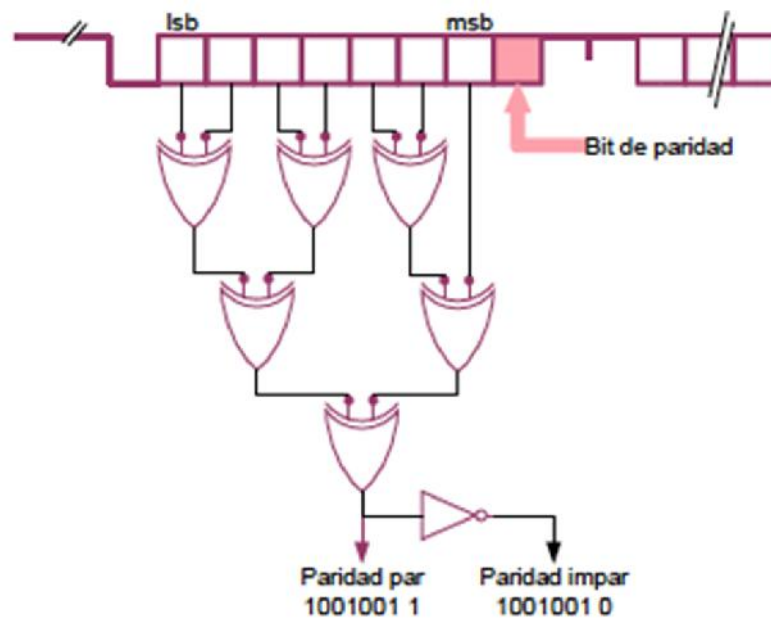
# Algorismes de detecció d'errors

## **Paritat**

- És el mètode més simple per detectar errors de bit en transmissions asíncrones orientades a caràcter
- El bit de paritat és una funció de tots els bits que formen el caràcter
- Quan es reben els bits, el receptor aplica la mateixa funció i si el resultat divergeix s'estableix que hi ha hagut un error de transmissió
- Per establir el càlcul de la paritat es realitza la suma en mòdul 2 de tots els bits del caràcter
- Si es desitja paritat parell, el nombre de bits transmesos amb valor 1 (incloent el bit de paritat) ha de ser parell
- Si es desitja senar, el nombre total de 1s serà senar.

# Algorismes de detecció d'errors

## Paritat



# Algorismes de detecció d'errors

## **Paritat**

- El conjunt de informació i els bits de detecció d'errors es denomina paraula de codi
- El mínim número de bits en que difereixen dos paraules de codi es denomina distància Hamming
- Aquesta distància permet establir el número de errors que es poden detectar amb un determinat codi

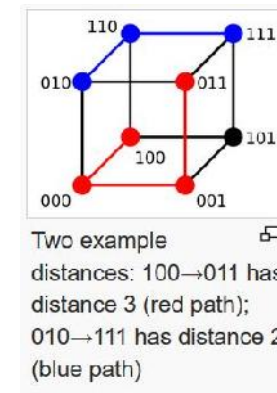


# Algorismes de detecció d'errors

## Paritat

- Si analitzem una llista de paraules de codi:

- 0000000 0
- 0000001 1
- 0000010 1
- 0000011 0



- En aquest cas la distància Hamming és dos
- És capaç de detectar errors de un bit o de un número senar de bits


# Algorismes de detecció d'errors

## **Verificació de suma de bloc**

- Quan es transmet un conjunt de caràcters o bytes, és possible que hi hagi un error en un d'ells
- La probabilitat de que un bloc de caràcters o de bytes tingui un error es coneix com Tassa d'errors de bloc
- Quan es transmet un bloc d'informació es pot millorar la detecció d'errors incloent una detecció de columnes i una de files

# Algorismes de detecció d'errors

Paridad de fila (impar)	$P_R$	$B_6$	$B_5$	$B_4$	$B_3$	$B_2$	$B_1$	$B_0$	
	0	0	0	0	0	0	1	0	=STX
	1	1	0	1	0	1	0	1	
	0	0	0	1	1	0	0	1	
	0	0	1	1	0	1	0	0	
	1	0	0	1	0	1	0	0	
	1	0	0	1	0	0	0	1	
	1	1	0	0	1	0	0	0	
	1	0	0	0	0	0	1	1	=ETX
	0	0	1	1	0	1	0	0	=BCC
Paridad de fila (par)									

 Ejemplo de combinación no detectada

$P_R$  Bit de paridad de fila

BCC Carácter de control de bloque

# Algorismes de detecció d'errors

## **Verificació de suma de bloc**

- A l'exemple es veu que tot i que dos errors en una fila poden passar desapercebuts, aquests són detectats en la columna
- També es cert que si es modifiquen dos bits en una fila i els mateixos en una altra fila, el error passa inadvertit, tot i que aquesta situació és molt menys probable

# Algorismes de detecció d'errors

## Codis de redundància cíclica

- Els esquemes anteriors són útils en situacions on els errors són només d'un bit
- Si tenim un error de ràfega es precisa un mètode més segur i rigorós.
- Un error de ràfega es defineix per el nombre de bits entre bits erronis, incloent aquests

1011110011101111001111001

1011100010100000101111001

- A més s'ha de satisfer la condició següent:

***Si la ràfega anterior era de B bits, fins al següent error han d'haver uns B bits correctes per considerar que no és la mateixa ràfega***

# Algorismes de detecció d'errors

## **Codis de redundància cíclica**

- El codi es calcula a partir de les dades de la trama i s'afegeix en la cua de la mateixa
- El numero de dígit utilitzat per trama es selecciona en base als tipus d'errors de transmissió esperats
- En general es solen fer servir 16 ó 32 bits
- Els dígit de verificació es denominen Seqüència de Verificació de Trama (FCS: Frame Check Sequence) o dígit de Codi de Redundància Cíclica (CRC)

# Algorismes de detecció d'errors

## Codis de redundància cíclica

- El mètode utilitzat funciona de la següent forma:
  - $M(x)$  és un nombre de  $k$  bits (missatge)
  - $G(x)$  és un nombre de  $(n+1)$  bits anomenat divisor o generador
  - $R(x)$  és un nombre de  $n$  bits, tal que  $k > n$  (el residu de la divisió)

$$\frac{M(x) \cdot 2^n}{G(x)} = Q(x) + \frac{R(x)}{G(x)}$$

On  $Q(x)$  és el quocient

$$\frac{M(x) \cdot 2^n + R(x)}{G(x)} = Q(x)$$

Fem servir aritmètica mòdul 2

# Algorismes de detecció d'errors

- Per aprofitar això, el contingut total de la trama es desplaça cap a l'esquerra tants zeros com dígit de FCS es van a generar
- Això equival a multiplicar el missatge per  $2^n$ , on  $n$  són els nombres de dígit de FCS
- Aquest número es divideix en mòdul 2 per el polinomi generador
- El residu serà el FCS que es transmet al final de la trama



# Algorismes de detecció d'errors

## Codis de redundància cíclica

- En el receptor es divideix la trama completa, incloent el FCS obtingut amb el polinomi generador

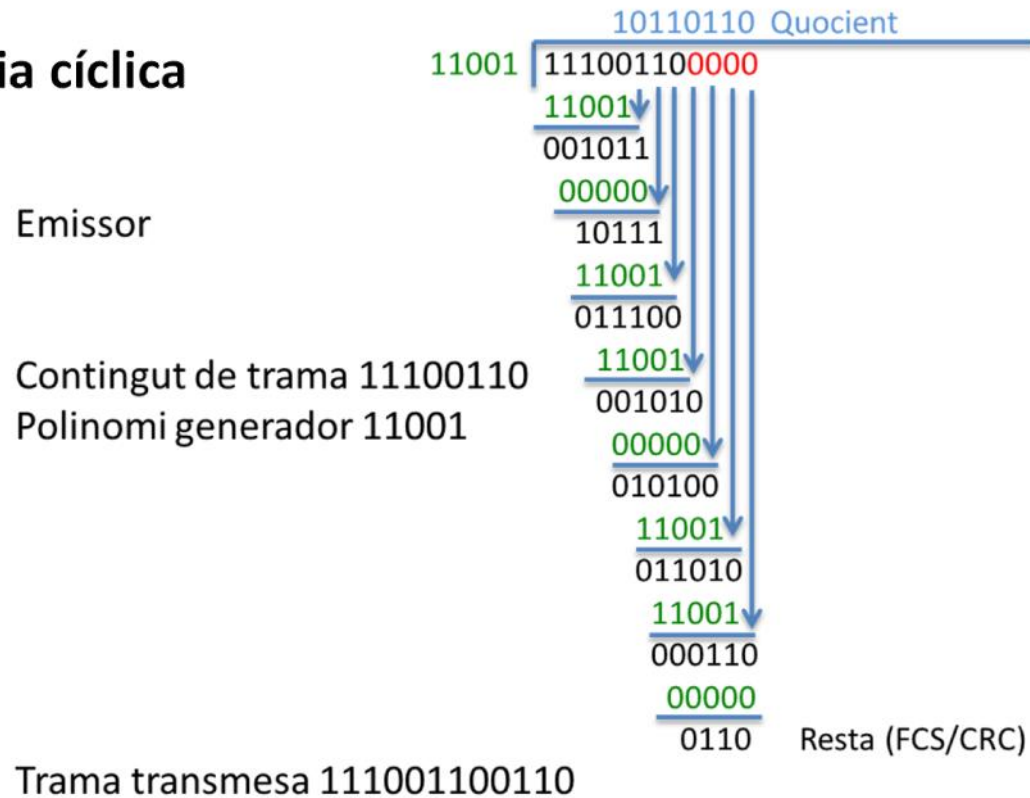
$$\frac{M(x) \cdot 2^n + R(x)}{G(x)}$$

- Si no es presenten errors, el residu serà 0
- En cas d'error, es detectarà un residu no nul

y sistemas abiertos". Fred Halsall, Addison Wesley, ISBN 968 444 331

# Algorismes de detecció d'errors

## Codis de redundància cíclica

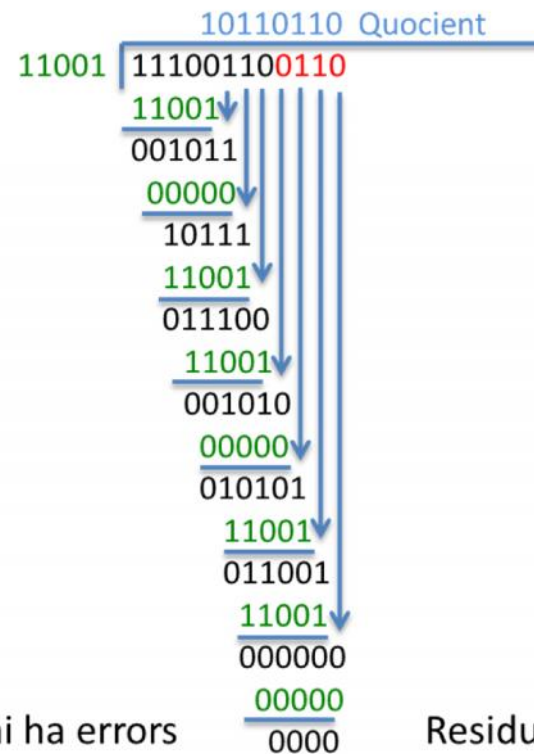


# Algorismes de detecció d'errors

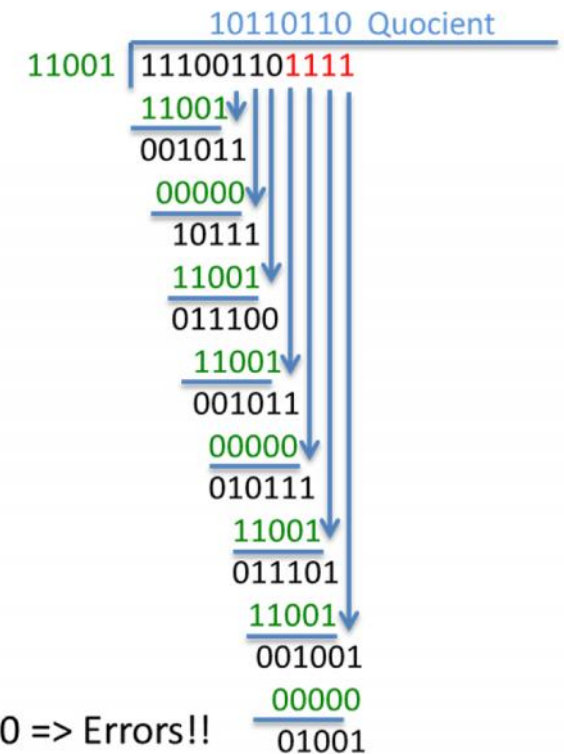
## Codis de redundància cíclica

Receptor

Residu = 0 => No hi ha errors



Residu ≠ 0 => Errors!!



# Algorismes de detecció d'errors

## Codis de redundància cíclica

- Alguns dels CRCs més utilitzats són:
  - CRC-16  $= X^{16} + X^{15} + X^2 + 1$
  - CRC-CCITT  $= X^{16} + X^{12} + X^5 + 1$
  - CRC-32  $= X^{32} + X^{26} + X^{23} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$

El CRC-16 és equivalent al número:

1 1000 0000 0000 0101

# Algorismes de detecció d'errors

## **Codis de redundància cíclica**

- El CRC-16 detecta:
  - Errors de ràfega de menys de 16 bits
- CRC-16 i CRC-CCITT es fan servir de forma extensiva en el sistema ISDN (en català XDSI), mentre que la CRC-32 es fa servir generalment en LANs
- Les divisions en mòdul-2 s'implementen en hardware dins dels circuits de comunicacions.