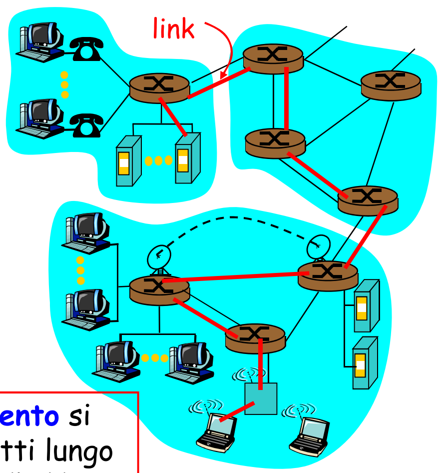
Lo strato di collegamento
Parte 1

Strato di collegamento (Data Link)

- **Gli host e i router sono i** nodi
- **i canali di comunicazione che collegano nodi adiacenti lungo un cammino sono i** collegamenti (link)
  - collegamenti cablati
  - collegamenti wireless
  - LAN
- **Le unità di dati scambiate dai protocolli a livello di link sono chiamate** frame.



I protocolli di strato di collegamento si occupano del trasporto dei pacchetti lungo un singolo canale di comunicazione (link)

3

Strato di collegamento

- **Un pacchetto può essere gestito da diversi protocolli su collegamenti differenti**
  - Es., un pacchetto può essere gestito da Ethernet sul primo collegamento, da PPP sull'ultimo e da un protocollo WAN nel collegamento intermedio
- **I servizi erogati dai protocolli del livello di link possono essere differenti**
  - Ad esempio, non tutti i protocolli forniscono un servizio di consegna affidabile (controllo d'errore)

4

Servizi offerti dallo strato di link

- **Framing**
  - I protocolli incapsulano i pacchetti del livello di rete all'interno di un **frame** a livello di link
  - Se necessario (reti ad accesso multiplo) il **protocollo MAC** controlla l'accesso al mezzo
    - Per identificare origine e destinatario vengono utilizzati indirizzi "MAC"
- **Rivelazione e correzione degli errori**
  - Gli errori sono causati dal transito del segnale nel mezzo trasmissivo
  - Il nodo ricevente individua la presenza di errori
    - è possibile grazie all'inserimento, da parte del nodo trasmittente, di bit di controllo di errore all'interno del frame
  - Il nodo ricevente oltre a rivelare l'errore e lo corregge

5

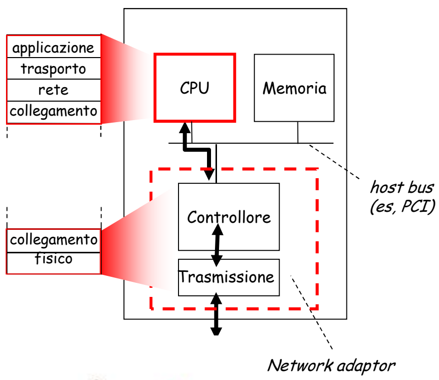
Servizi offerti dal livello di collegamento

- **Controllo di flusso**
  - Evita che il nodo trasmittente saturi quello ricevente
- **Consegna affidabile dei dati e ritrasmissione**
  - Nel caso i requisiti dell'applicazione impongano una consegna affidabile dei dati il protocollo di link può effettuare la ritrasmissione delle frame affette da errore
    - Questa funzione può essere eseguita anche nello strato di trasporto (es. TCP)
  - È normalmente utilizzata nei collegamenti soggetti a elevati tassi di errori (es.: collegamenti wireless)
- **Half-duplex e full-duplex**
  - Nella modalità full-duplex gli estremi di un collegamento possono trasmettere contemporaneamente
  - nella modalità half-duplex la trasmissione nei due versi è alternata

6

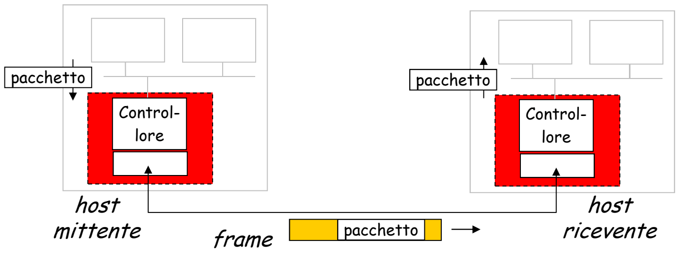
Esempio di implementazione

- **In tutti gli host**
- **È realizzato in una Network Interface Card (NIC)**
  - Es. scheda Ethernet, PCMCI, 802.11
  - Implementa il livello di collegamento e fisico
- **E' una combinazione di hardware, software e firmware**



7

Esempio di implementazione



- **Lato mittente:**
  - Incapsula un pacchetto in un frame.
  - Imposta il bit rilevazione degli errori, trasferimento dati affidabile, controllo di flusso, etc.
- **Lato ricevente:**
  - Individua gli errori, trasferimento dati affidabile, controllo di flusso, etc.
  - Estrae i pacchetti e li passa al nodo ricevente

8

Framing

Framing

- **Ha lo scopo di formare la PDU di strato (frame) incapsulando la PDU di strato superiore (pacchetto)**
- **L'entità ricevente deve essere in grado di riconoscere senza ambiguità l'inizio e la fine di ogni frame (funzione di delimitazione)**
- **Ad ogni frame viene aggiunto all'inizio e alla fine una sequenza fissa di bit, denominata flag**
  - L'entità ricevente esamina il flusso binario entrante e delimita le frame riconoscendo i flag di apertura e di chiusura
- **Problema della simulazione del flag all'interno della frame**

Esempio di funzione di delimitazione

- **Una possibile configurazione del Flag di delimitazione è**

  01111110
- **Per evitare la simulazione si utilizzano le funzioni di**
- **Bit stuffing**
  - In emissione, si aggiunge uno "0" dopo ogni sequenza di cinque "1" consecutivi all'interno della frame indipendentemente da quale sia il bit successivo
- **Bit destuffing**
  - In ricezione si contano gli "1" consecutivi
  - Quando sono ricevuti cinque "1" consecutivi, si esamina la cifra successiva
    - se è un "1": la sequenza di cifre binarie è un Flag
    - se è un "0": questo è un bit di stuffing e deve quindi essere eliminato

Esempio bit stuffing

- **Sequenza originale**

1 0 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 1 1 1 1 1 0 0 1 1 1 1 1 1 0 0

- **Sequenza trasmessa**

1 0 1 1 1 1 1 $\boxed{0}$ 1 1 1 1 1 $\boxed{0}$ 1 0 1 1 0 1 1 1 1 1 $\boxed{0}$ 0 0 1 1 1 1 1 $\boxed{0}$ 1 0 0

bit di stuffing

- **Sequenza ricevuta**

1 0 1 1 1 1 1 0 1 1 1 1 1 0 1 0 1 1 0 1 1 1 1 1 0 0 0 1 1 1 1 1 0 1 0 0

0 dopo cinque "1" consecutivi: bit di stuffing bit eliminato

bit di stuffing bit eliminato

bit di stuffing bit eliminato

bit di stuffing bit eliminato

Byte stuffing e de-stuffing

- **Utilizzata nel protocollo PPP (Point to Point Protocol)**
- **Byte stuffing (si usa una sequenza nota come «control escape» 01111101)**
  - In emissione, se in una parte della frame compare la sequenza "01111110" (ad eccezione del flag") o la sequenza "01111101" viene premesso il byte "01111101"
- **Byte destuffing**

  In ricezione

  Se si ricevono due byte consecutivi "01111101" uno dei due viene eliminato

  Se si riceve un "01111101" seguito da un "01111110" il primo viene eliminato

  Se si riceve un solo "01111110" viene riconosciuto come flag

- **Esempio**
  - Sequenza originale

  1 0 1 1 1 1 1 1 1 0 1 1 $\boxed{0\ 1\ 1\ 1\ 1\ 1\ 0\ 1}$ 0 1 1 1 1 1 0 0

  - Sequenza trasmessa

1 0 1 1 1 1 1 1 1 0 1 1 $\boxed{0\ 1\ 1\ 1\ 1\ 1\ 0\ 1}$ $\boxed{0\ 1\ 1\ 1\ 1\ 1\ 0\ 1}$ 0 1 1 1 1 1 0 0

13

Rivelazione e correzione d'errore

14

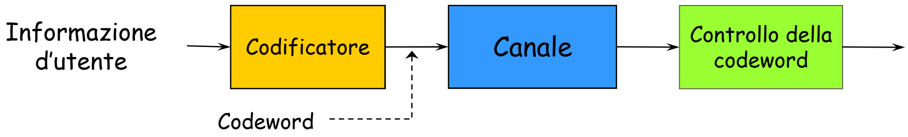
Controllo d'errore

- **La trasmissione introduce errori**
  - Bit Error Rate (BER)
- **Il controllo d'errore si usa quando il livello trasmissivo non soddisfa i requisiti dell'applicazione**
- **Il controllo d'errore assicura un determinato livello di accuratezza nella trasferimento di uno stream dati**
- **Due approcci possibili**
  - Error detection & retransmission (ARQ)
  - Forward Error Correction (FEC)

15

Principio base del controllo d'errore

- Si organizza la trasmissione in modo da trasformare i blocchi di dati trasmessi in particolari "parole di codice" (codeword)
- Se il blocco ricevuto non è una parola di codice è considerato in errore
- E' necessaria una ridondanza (overhead) costituita da un insieme di bit di controllo da aggiungere al blocco dati d'utente
- E' possibile che il canale trasformi la parola di codice trasmessa in una stringa di bit che è ugualmente una parola di codice



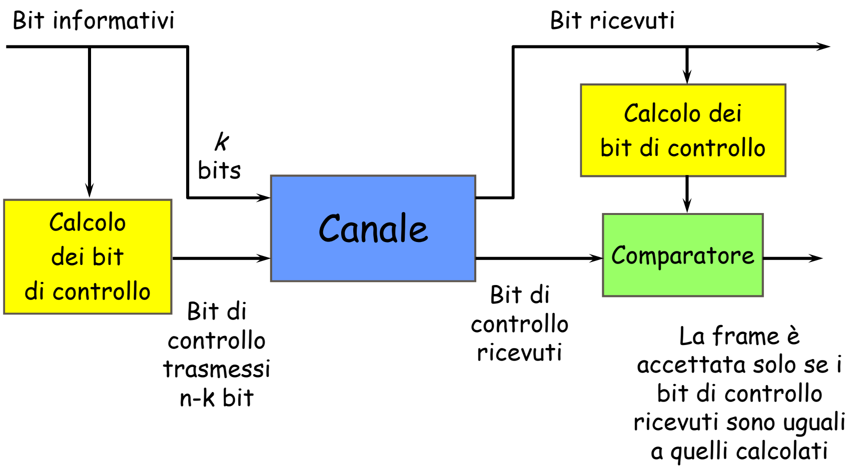
16

Rivelazione di errore (2/4)

- Se
  - *k* è la lunghezza del blocco da proteggere;
  - *n-k* è il numero di bit di controllo
- Le codeword sono di lunghezza uguale a n bit
- Se una PDU è colpita da errore e se questi sono in configurazione tale da non essere rivelati (sostituzione di codeword), si verifica l'evento di "errori non rivelati"
- I metodi di codifica per rivelare errori rientrano usualmente nella categoria dei codici con controllo di parità (parity check codes)
  - codici a parità singola
  - codici a parità a blocchi
  - codici a ridondanza ciclica (CRC, Cyclic Redundancy Check)

17

Funzione di Error Detection



18

Controllo di parità singola

- **Aggiunge un bit di parità a k bit informativi**

Info Bit	$b_1, b_2, b_3, \ldots, b_k$
Check Bit	$b_{k+1} = (b_1 + b_2 + b_3 + \ldots + b_k)$ modulo 2
Codeword	$(b_1, b_2, b_3, \ldots, b_k, b_{k+1})$

- **Un blocco dati trasmesso ha un numero pari di "1"**
- **Il ricevitore controlla se il numero di "1" è pari**
  - E' rivelabile una qualsiasi configurazione di errore che modifica un numero dispari di bit
  - Tutte le configurazione di errore che modificano un numero pari di bit non sono rilevabili

19

Esempio

- **Bit informativi (7 bit): (0, 1, 0, 1, 1, 0, 0)**
- **Bit di parità: b_8 = 0 + 1 + 0 + 1 +1 + 0 + 0 = 1**
- **Codeword (8 bit): (0, 1, 0, 1, 1, 0, 0, 1)**

- **Errore singolo nel bit 3 : (0, 1, 1, 1, 1, 0, 0, 1)**
  - Numero di "1" è uguale a 5 (dispari)
  - Errore rivelato
- **Errore nei bit 3 and 5: (0, 1, 1, 1, 0, 0, 0, 1)**
  - Numero di "1" = 4 (pari)
  - Errore non rivelato

20

Prestazioni del controllo di parità

- **Ridondanza**
  - Il controllo di parità aggiunge 1 bit di ridondanza ogni k bit informativi
  - overhead = $1/(k+1)$
- **Errori rivelati**
  - Una <u>configurazione di errore</u> è una stringa binaria composta da (n=k+1) bit [(k+1)-tuple], in cui sono presenti bit "1" nelle posizioni in cui si sono verificati gli errori, mentre gli altri bit sono uguali a "0"
  - Tutte le configurazioni di errore con un numero dispari di bit modificati sono rivelati
  - Tra tutte le 2^{k+1} $(k+1)$-tuple binarie, $\frac{1}{2}$ hanno un numero dispari di "1"
  - Solo il 50% delle configurazioni di errore possono essere rivelate

21

Prestazioni del controllo di parità

- **Normalmente si assume l'ipotesi che i canali introducono errori sui bit in modo indipendente con probabilità p**
  - una statistica più attendibile prevede **errori a burst**
- **Alcune configurazioni di errore sono più probabili di altre**

$$P[10000000] = p(1-p)^7 = (1-p)^8 \frac{p}{1-p}$$

$$P[11000000] = p^2(1-p)^6 = (1-p)^8 \left(\frac{p}{1-p}\right)^2$$

- **Poichè si può assumere p<0.5 si ha p/(1 - p)<1, le configurazioni con 1 solo errore sono più probabili delle configurazioni con 2 errori e così via**
- **Qual è la probabilità di non rivelare gli errori ?**

22

Prestazioni del controllo di parità

- **Gli errori non rivelabili**
  - Configurazione d'errore con un numero pari di "1"

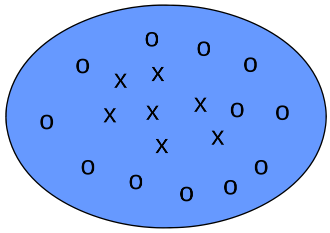
$$\Pr\{errore\ non\ rivelabile\} = \Pr\{config.\ di\ errore\ con\ \#\ pari\ di\ 1\} =$$
$$= \binom{n}{2} p^2(1-p)^{n-2} + \binom{n}{4} p^4(1-p)^{n-4} + \ldots$$

- **Esempio: n=32, p=10^{-3}**

$$\Pr\{errore\ non\ rivelabile\} =$$
$$\binom{32}{2}(10^{-3})^2(1-10^{-3})^{30} + \binom{32}{4}(10^{-3})^4(1-10^{-3})^{28} + \ldots =$$
$$= 496(10^{-6}) + 35960(10^{-12}) \approx 4.96 \cdot 10^{-4}$$

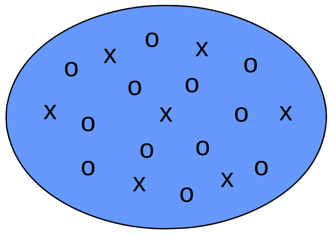
Quanto è “buono” un codice ?

- **In molti canali le configurazioni di errore più probabili sono quelle con un numero basso di bit errati**
- **Questi errori trasformano le codeword trasmesse in n-tuple “vicine”**
- **Se le codeword sono “vicine” tra loro allora la funzione di rivelazione può fallire**
- **I buoni codici massimizzano la “distanza” tra le codeword trasmesse**



Distanza bassa: Codice non buono

x = codewords
o = noncodewords



Distanza elevata: Codice buono

Controllo di parità bi-dimensionale

- **Un numero maggiore di bit di parità aumentano le prestazioni del codice**
  - Si struttura la sequenza di bit informativi in colonne
  - Si aggiunge un bit di parità per ogni colonna
  - Si aggiunge una “colonna di parità”

1	0	0	1	0	0
0	1	0	0	0	1
1	0	0	1	0	0
1	1	0	1	1	0
1	0	0	1	1	1

La colonna finale è formata dai bit di parità di ogni riga

La riga finale è formata dai bit di controllo di ogni colonna

Capacità di rivelazione d'errore

1	0	0	1	0	0
0	0	0	0	0	1
1	0	0	1	0	0
1	1	0	1	1	0
1	0	0	1	1	1

1 errore

1	0	0	1	0	0
0	0	0	0	0	1
1	0	0	1	0	0
1	0	0	1	1	0
1	0	0	1	1	1

2 errori

- Configurazioni con 1, 2, o 3 errori possono essere sempre rivelate.

1	0	0	1	0	0
0	0	0	1	0	1
1	0	0	1	0	0
1	0	0	1	1	0
1	0	0	1	1	1

3 errori

1	0	0	1	0	0
0	0	0	1	0	1
1	0	0	1	0	0
1	0	0	0	1	0
1	0	0	1	1	1

4 errori
(non rivelabile)

- Non tutte le configurazioni di >4 errori possono essere rivelate

Altri codici di rivelazione d'errore

- **I codici a parità singola hanno scarse prestazioni**
  - Elevata probabilità di non rivelare errori
- **I codici bi-dimensionali hanno overhead elevato**
  - richiedono un numero elevato di bit di controllo
- **I codici più usati sono**
  - Internet Checksums
    - Strato di trasporto (implementazione software)
  - Codici polinomiali a ridondanza ciclica (CRC)
    - Strato di collegamento (implementazione hardware)

27

Internet Checksum

- Molti protocolli usati in Internet (es. IP, TCP, UDP) usano bit di controllo (checksum) per rivelare errori nell'header IP (o nell'header e nel campo dati delle unità dati TCP/UDP)
- Il checksum è inserito in uno specifico campo dell'header delle PDU (RFC 1071)
- Il checksum è ricalcolato in ogni router e quindi deve essere di facile implementazione in software
- Si considera che la stringa di bit da proteggere sia composta da L parole di 16 bit

$$b_0, b_1, b_2, \ldots, b_{L-1}$$

- Il checksum è una stringa b_L di 16 bit

28

Calcolo del Checksum

- Il checksum b_L è calcolato come segue
- Ciascuna stringa di 16-bit è considerata un intero

$$x = b_0 + b_1 + b_2 + \ldots + b_{L-1} \text{ modulo } 2^{16}-1$$

- Il checksum è dato da

$$b_L = -x \text{ modulo } 2^{16}-1$$

- Quindi, l'intero blocco trasmesso deve soddisfare la seguente proprietà

$$0 = b_0 + b_1 + b_2 + \ldots + b_{L-1} + b_L \text{ modulo } 2^{16}-1$$

- Il calcolo del checksum è eseguito in software

29

Esempio

- Uso di Aritmetica modulare
- Stringhe di 4 bit
- Si usa l'aritmetica mod_(2^4-1)=mod_15
  - b_0 = 1100 = 12
  - b_1 = 1010 = 10
  - b_0+b_1 = 12+10 = 7 mod_15
  - b_2 = -7 = 8 mod_15
- Quindi
  - b_2 = 1000

30

Codici CRC: i polinomi

- Le singole cifre binarie di una stringa da proteggere sono trattate come coefficienti (di valore "0" o "1") di un polinomio P(x)
- Le cifre binarie della stringa con lunghezza uguale a K sono considerate come i coefficienti di un polinomio completo di grado K-1

$$P(x) = a_{k-1}x^{k-1} + a_{k-2}x^{k-2} + \ldots + a_1x^1 + a_0$$

- In particolare, l'i-esimo bit (a_i) della stringa è il coefficiente del termine x_{i-1} di P(x)

31

Codici CRC: i polinomi

- Le entità emittente e ricevente utilizzano un polinomio comune $G(x)$, detto polinomio generatore
- il polinomio $G(x)$ gode di opportune proprietà nell'ambito della teoria dei campi algebrici
- i coefficienti di $G(x)$ sono binari, come quelli di $P(x)$, supponiamo che questo polinomio sia di grado z
  - i coefficienti di $G(x)$ di grado massimo e di grado nullo debbono entrambi essere uguali a 1
  - Es: $x^{16} + x^{12} + x^{5} + 1$ $(z = 16)$

32

Codici CRC: i polinomi

- La entità emittente utilizza $G(x)$ come divisore del polinomio $x^{z}P(x)$

$$\frac{x^{z}P(x)}{G(x)} = Q(x) + \frac{R(x)}{G(x)}$$

- $Q(x)$ è il polinomio quoziente
- La particolarità della divisione risiede nel fatto che
  - i coefficienti di dividendo e di divisore sono binari
  - l'aritmetica viene svolta modulo 2

Aritmetica polinomiale a coefficienti binari

- **Addizione**
  - Addizione e sottrazione sono operazioni identiche
  - Equivalgono ad un XOR sui bit degli operandi

$$(x^7 + x^6 + 1) + (x^6 + x^5) = x^7 + x^6 + x^6 + x^5 + 1 = x^7 + x^5 + 1$$

- **Moltiplicazione**
  - la moltiplicazione di una stringa binaria per 2^k equivale ad uno shift verso sinistra di k posizioni

$$(x+1)\cdot(x^2+x+1) = x^3 + x^2 + x^2 + x + x + 1 = x^3 + 1$$

Aritmetica polinomiale a coefficienti binari

- **Divisione**
  - Algoritmo di Euclide

Divisore: $x^3 + x + 1$

Quoziente [Q(x)]: $x^3 + x^2 + x$

Dividendo: $x^6 + x^5$

$$\begin{array}{r|l}
 & x^3 + x^2 + x \\
\hline
x^3 + x + 1 & x^6 + x^5 \\
 & x^6 \qquad + x^4 + x^3 \\
\hline
 & \quad + x^5 + x^4 + x^3 \\
 & \quad + x^5 \qquad + x^3 + x^2 \\
\hline
 & \qquad\quad + x^4 \qquad + x^2 \\
 & \qquad\quad + x^4 \qquad + x^2 + x \\
\hline
 & \qquad\qquad\qquad + x
\end{array}$$

Resto [R(x)]: $+ x$

35

Osservazione

- Dato il grado del polinomio generatore, il grado del polinomio resto $R(x)$ è al più uguale a $Z-1$;
- Conseguentemente $R(x)$ può essere sempre rappresentato con Z coefficienti (binari), ponendo uguali a "0" i coefficienti dei termini mancanti

36

Codici CRC: l'emettitore

- Ottenuto il resto $R(x)$, l'entità emittente inserisce i coefficienti di questo polinomio in un apposito campo della PDU (campo CRC), che deve quindi avere lunghezza Z bit
- Nella PDU emessa trovano quindi posto le cifre binarie da proteggere (in numero uguale a K) e le cifre CRC (in numero uguale a Z): in totale $K+Z$ cifre binarie, che sono rappresentative di un polinomio $T(x)$ di grado $K+Z-1$

$$T(x) = x^Z P(x) + R(x)$$

e che costituiscono una parola di codice

Codici CRC: l'emettitore

- **Tenendo conto che, per definizione,**

$$x^Z P(x) = Q(x)G(x) + R(x)$$

e poiché addizione e sottrazione modulo 2 si equivalgono, si ottiene

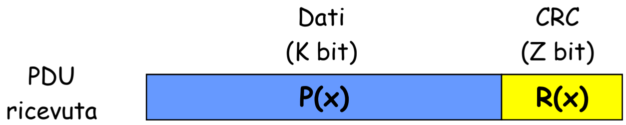
$$x^Z P(x) - R(x) = x^Z P(x) + R(x) = T(x) = Q(x)G(x)$$

cioè la stringa emessa (rappresentativa del polinomio *T(x)*) è divisibile per il polinomio generatore *G(x)*

- **Si conclude che**
  - tutte le parole di codice sono divisibili per il polinomio generatore
  - tutti i polinomi divisibili per *G(x)* sono parole di codice

Codici CRC: il ricevitore

- **La entità ricevente esegue, con il polinomio generatore, l'operazione di divisione effettuata in emissione**
- **in questo caso opera però sul polinomio rappresentato dalle K+Z cifre binarie ricevute**



Codici CRC: il ricevitore

- **Supponiamo che nel trasferimento si siano verificati errori, con una sequenza rappresentata dal polinomio *E(x)***
  - ogni errore nella PDU corrisponde ad un coefficiente non nullo in *E(x)*
- **allora le cifre binarie ricevute rappresentano il polinomio**

$$T(x)+E(x)\ ,$$

ove l'addizione è svolta modulo 2 (XOR)

Codici CRC: il ricevitore

- **Ogni bit "1" in *E(x)* corrisponde ad un bit che è stato invertito e quindi a un errore isolato**
- **un errore a burst di lunghezza *n* è caratterizzato in *E(x)* da un "1" iniziale, una mescolanza di "0" e "1", e un "1" finale per un complesso di *n* coefficienti binari**

$$E(x) = x^i\ (x^{n-1} + \ldots .+1),$$

ove *i* determina quanto il burst è lontano dall'estremità destra della PDU

41

Codici CRC: il ricevitore

- **Il ricevitore calcola il resto della divisione di *T(x)+E(x)* per *G(x)***
  - le modalità sono le stesse utilizzate nell'emettitore
- **poiché *T(x)* è divisibile per *G(x),* ne segue che**

$$Resto\left[\frac{T(x)+E(x)}{G(x)}\right]=Resto\left[\frac{E(x)}{G(x)}\right] \quad .$$

42

Codici CRC: il ricevitore

- **Conseguentemente la regola applicata dal ricevitore è la seguente:**
  - se il resto della divisione [*T(x)+E(x)*] / *G(x)* è nullo, la PDU ricevuta è assunta "senza errori"
  - in caso contrario, si sono verificati uno o più errori nel corso del trasferimento.
- **Si nota che sono non rivelabili le configurazioni di errore per le quali il relativo polinomio *E(x)* contiene *G(x)* come fattore**

Codici CRC: protezione contro gli errori

- **Un codice polinomiale, in cui il polinomio generatore contiene *x+1* come fattore primo, è in grado di rivelare**
  - tutti gli errori singoli o doppi;
  - tutti gli errori isolati con una molteplicità dispari
  - tutti gli errori a burst di lunghezza $\leq Z$
- **Se la lunghezza del burst è z+1 e se tutte le combinazioni della raffica sono considerate equiprobabili, la probabilità che l'errore a raffica non sia rivelato è uguale a $2^{-(Z-1)}$.**
- **Infine, se il burst ha lunghezza maggiore di z+1, nell'ipotesi di equiprobabilità delle configurazioni di errore, la probabilità di errore non rivelato è uguale a 2^{-Z}.**

Codici CRC: polinomi generatori

- **Sono standard i seguenti polinomi generatori:**

$$G(x)= x^{16} + x^{12} + x^{5} + 1$$

$$G(x)= x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} + x + 1$$

- **Entrambi sono divisibili per *x+1* e quindi danno luogo a codici CRC con le proprietà suddette**

45

Esempio calcolo CRC

Polinomio Generatore: $G(x)= x^3 + x + 1$

Dati: (1,1,0,0) $P(x) = x^3 + x^2$

$x^3P(x) = x^6 + x^5$

$x^3 + x^2 + x$

$x^3 + x + 1 \; / \; x^6 + x^5$

$x^6 + \quad x^4 + x^3$

$x^5 + x^4 + x^3$

$x^5 + \quad x^3 + x^2$

$x^4 + \quad x^2$

$x^4 + \quad x^2 + x$

x

1110

1011 | 1100000

1011

1110

1011

1010

1011

010

Codeword trasmessa: $b(x) = x6 + x5 + x$ (1,1,0,0,0,1,0)

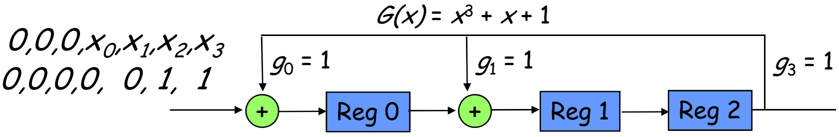
46

Implementazione tramite shift register

- **Bit informativi P(x)= $x^{k-1}, x^{k-2}, \ldots, x^2, x^1, x^0$**
- **Aggiungere z bit "0" a destra di P(x)**
  - Polinomio x^z P(x)
- **Immettere la sequenza in un circuito shift-register che esegue la divisione tra polinomi**
  - le "prese" (tap) nel circuito sono determinate dai coefficienti del polinomio generatore
- **Dopo z shifts, lo shift register contiene il resto R(x)**

47

Implementazione tramite shift register



Clock	Input	Reg 0	Reg 1	Reg 2
0	-	0	0	0
1	1 = x_3	1	0	0
2	1 = x_2	1	1	0
3	0 = x_1	0	1	1
4	0 = x_0	1	1	1
5	0	1	0	1
6	0	1	0	0
7	0	0	1	0
	bit di controllo	$r_0 = 0$	$r_1 = 1$	$r_2 = 0$

$\Longrightarrow$ $\boxed{r(x) = x}$

48

Forward Error Correction (FEC)

- Date due stringhe binarie di ugual lunghezza, X e Y e posto W(A) = *numero di bit 1 della stringa A*, si definisce distanza di Hamming tra X e Y la quantità

  HD(X,Y) = W(X *xor* Y)

- Un codice con parole di n bit può rappresentare simboli di m bit e la capacità di correzione è funzione della ridondanza $r=n-m$; il valore minimo della HD tra tutte le coppie di parole di codice è la HD del codice
- Un codice con HD=2d+1 può correggere fino a d errori binari e può rivelarne fino a 2d
- Un esempio di codice con n=10, m=2, r=8, d=2 è il seguente

  0000000000 0000011111 1111100000 1111111111