

Esercizi svolti sul livello data-link

Esercizio 1 – Parità pari (even parity)

Dato il seguente byte incompleto, aggiungi il bit di parità **pari** in modo che il numero totale di bit a 1 sia **pari**: 1101010_

Soluzione:

Il byte ha 7 bit. Contiamo quanti 1 ci sono:

$$1 + 1 + 0 + 1 + 0 + 1 + 0 = \text{**4 bit a 1**}$$

Poiché il numero di 1 è già pari (4), il bit di parità deve essere 0 per non alterare la parità.

Risultato finale: 11010100

Esercizio 2 – Parità dispari (odd parity)

Hai ricevuto il seguente byte completo con bit di parità **dispari**. Verifica se il messaggio è corretto oppure se c'è stato un errore: 10111011

Soluzione:

Separiamo il bit di parità dal messaggio:

- **Bit di parità:** 1
- **Dati:** 1011101

Ora contiamo i bit a 1 nei dati:

$$1 + 0 + 1 + 1 + 1 + 0 + 1 = \text{**5 bit a 1**}$$

Il numero totale di 1 (incluso il bit di parità):

$$5 (\text{dati}) + 1 (\text{parità}) = 6 \rightarrow \text{pari}$$

Ma la parità doveva essere dispari \rightarrow quindi c'è un errore nella trasmissione.

Risposta:

Messaggio errato – la parità è pari ma doveva essere dispari.

Esercizio 3 – Checksum binario a 8 bit

Il mittente deve inviare due byte: 11001010 01101100

Calcola il checksum

Soluzione:

Passo 1 – Somma binaria

11001010
+ 01101100

100101110 (9 bit → overflow)

Il risultato ha 9 bit. Si somma il bit di overflow (riporto) alla parte inferiore a 8 bit:

00101110 (parte bassa) +
1 (bit di overflow)

00101111

Passo 2 – Complemento a uno (inversione di ogni bit)

Checksum = 11010000

Verifica (in ricezione):

00101111 (somma) +
11010000 (checksum)

11111111 → ok

Esercizio 4 – Checksum esadecimale (somma modulo 256)

Il mittente invia tre byte esadecimali: A3, 7F, 1D

Calcola il checksum modulo 256 (somma degli interi modulo 256) e verifica se la trasmissione è corretta.

Soluzione:

Passo 1 – Conversione decimale:

A3 = 163

7F = 127

1D = 29

Passo 2 – Somma:

$$163 + 127 + 29 = 319$$

Passo 3 – Modulo 256 e calcolo checksum:

$$319 \bmod 256 = 63$$

$$\text{Checksum} = 256 - 63 = \text{**193**} \rightarrow \text{in esadecimale: **C1**}$$

Il mittente trasmetterà: A3, 7F, 1D, C1

Verifica (lato ricevente):

$$\text{A3 (163)} + \text{7F (127)} + \text{1D (29)} + \text{C1 (193)} = 512$$

$$512 \bmod 256 = 0 \rightarrow \text{somma valida} \rightarrow \text{messaggio corretto}$$

ESERCIZIO 5 — VRC+LRC (Vertical Redundancy Check + Longitudinal Redundancy Check)

Data la seguente sequenza di **3 byte** da trasmettere, si usi il metodo **VRC** (bit di parità verticale) per aggiungere un bit di parità a ciascun byte.

Byte 1: 10110011

Byte 2: 01101101

Byte 3: 11001010

a) Calcolare i bit di parità per ciascun byte usando la parità pari e i bit LRC

b) Si supponga che durante la trasmissione il bit in posizione Byte 2, bit 5 venga invertito.

Stabilire se il VRC è in grado di rilevare e correggere l'errore.

Soluzione

a) Calcolo dei bit di parità e di LRC

Byte	b1	b2	b3	b4	b5	b6	b7	b8	Parità (bit 8)
Byte 1	1	0	1	1	0	0	1	1	?
Byte 2	0	1	1	0	1	1	0	1	?
Byte 3	1	1	0	0	1	0	1	0	?

Somma dei bit in ciascun byte:

- Byte 1: $1+0+1+1+0+0+1+1 = 5 \rightarrow$ dispari \Rightarrow bit di parità = 1
- Byte 2: $0+1+1+0+1+1+0+1 = 5 \rightarrow$ dispari \Rightarrow bit di parità = 1
- Byte 3: $1+1+0+0+1+0+1+0 = 4 \rightarrow$ pari \Rightarrow bit di parità = 0

Byte	b1	b2	b3	b4	b5	b6	b7	b8	VRC
Byte 1	1	0	1	1	0	0	1	1	1
Byte 2	0	1	1	0	1	1	0	1	1
Byte 3	1	1	0	0	1	0	1	0	0
LRC	0	0	0	1	0	1	0	0	0

Quindi:

Bit di parità di riga: 1100

Bit di parità di colonna: 000101000

b) Inversione del bit in posizione Byte 2, bit 5 (da 1 a 0):

Nuovo Byte 2: 01100101

Bit di parità di riga: 1000

Bit di parità di colonna: 000111000

Quindi alla riga 2 e alla colonna 5 il bit è stato invertito

Conclusione: Il metodo VRC+LRC è in grado di rilevare e correggere l'errore

Esercizio 6 – Generazione della Codeword (Hamming)

Data la sequenza di 4 bit 101010111, calcola:

1. Il numero di bit di ridondanza da inserire
2. Le posizioni dei bit di parità
3. Il valore dei bit di parità
4. La codeword da trasmettere secondo il codice di Hamming

Soluzione

- Messaggio: 1 0 1 0 1 0 1 1 1
- Numero di bit di ridondanza r tale che $2^r \geq m + r + 1$
 - $m = 9 \rightarrow r = 4 \rightarrow$ totale = 8 bit
- Posizioni:
 - Ridondanza: posizioni 1, 2, 4, 8
 - Dati: posizioni 3, 5, 6, 7, 9, 10, 11, 12, 13

- Inserimento preliminare:

P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13
		1		0	1	0		1	0	1	1	1

Posizione	Binario
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101

Calcolo bit di parità (bit in posizione potenza di 2):

- p1 (pos 1): controlla posizioni 1,3,5,7,9,11,13 → bit: [p1,1,0,0,1,1,1] → $1+1+1+1=4$ (pari) → $p1=0$
- p2 (pos 2): controlla posizioni 2,3,6,7,10,11 → [p2,1,1,0,0,1] → $1+1+1=3$ (dispari) → $p2=1$
- p4 (pos 4): controlla posizioni 4,5,6,7,12,13 → [p4,0,1,0,1,1] → $1+1+1=3$ (dispari) → $p4=1$
- p8 (pos 8): controlla posizioni 8,9,10,11,12,13 → [p8,1,0,1,1,1] → $1+1+1+1=4$ (pari) → $p8=0$

La codeword è: 0 1 1 1 0 1 0 0 1 0 1 1 1

Esercizio 7 – Verifica correttezza della Codeword (Hamming)

Data la codeword 1 1 1 1 0 1 0 0 1 0 1 1 1 determinare se ci sono stati errori in trasmissione e in che posizione

Soluzione

Calcolare la parità

Per ciascun bit di parità, verifichiamo la parità **sui bit controllati** da quella posizione.

◆ **Parità in posizione 1 (p1):**

Controlla posizioni: 1, 3, 5, 7, 9, 11, 13

Valori: 1, 1, 0, 0, 1, 1, 1 → somma = 5 (dispari) ⇒ parità **1**

◆ **Parità in posizione 2 (p2):**

Controlla posizioni: 2, 3, 6, 7, 10, 11

Valori: 1, 1, 1, 0, 0, 1 → somma = 4 (pari) ⇒ parità **0**

◆ **Parità in posizione 4 (p4):**

Controlla posizioni: 4–7, 12–13

Valori: 1, 0, 1, 0, 1, 1 → somma = 4 (pari) ⇒ parità **0**

◆ **Parità in posizione 8 (p8):**

Controlla posizioni: 8–13

Valori: 0, 1, 0, 1, 1, 1 → somma = 4 (pari) ⇒ parità **0**

Parità calcolata vs. ricevuta:

- p8: 0 (ok)
- p4: 0 (ok)
- p2: 0 (ok)
- p1: 1 (**errore**)

La sequenza binaria è 0001 → **errore in posizione 1**