Capitolo 2
Codifica binaria dell'informazione

2.1 – Rappresentazione dell'informazione
2.2 – Codifica di caratteri
2.3 – Codifica dei numeri
2.4 – Trasmissione dell'informazione
2.5 – Protezione dell'informazione

2.1
Rappresentazione dell'informazione

Simbolo, alfabeto e stringa

Informazione - **Stringa** di lunghezza finita formata da **simboli** s_i appartenenti ad un **alfabeto** di definizione A:

$s_1 s_2 s_3 \ldots s_i \ldots\ldots s_{n-1} s_n$ con $s_i \in A: \{a_1, a_2, .., a_m\}$

Alfabeto A: insieme di informazioni "elementari"

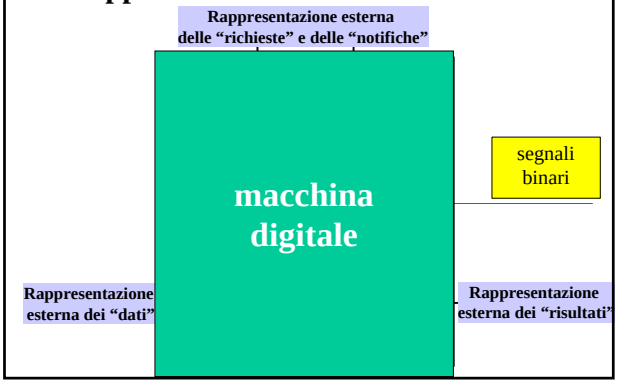
Esempi:

"testo" e caratteri | "numero" e cifre | "musica" e note

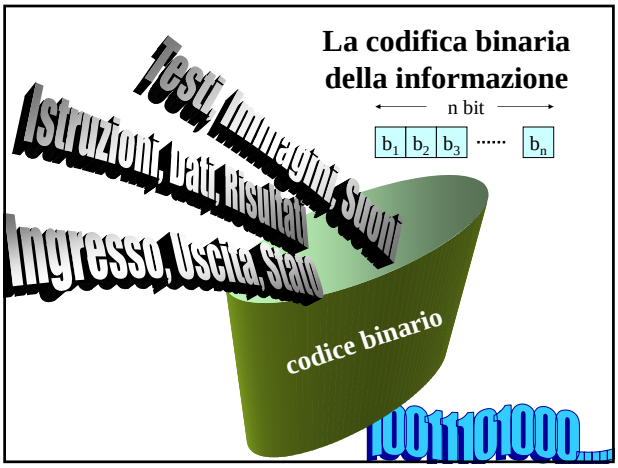
"immagine", pixel e toni di grigio | "disegno" e pend./lung. di tratti

"misura" e posizione di un indice | "parlato" e fonemi

Macchine digitali e rappresentazione dell'informazione



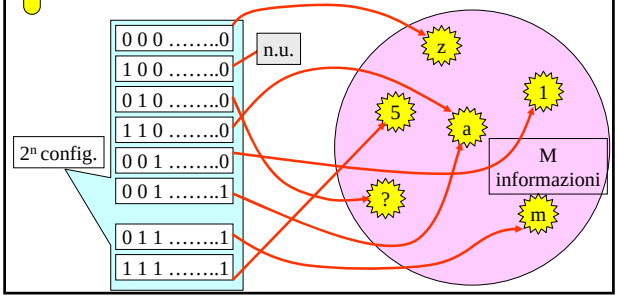
La codifica binaria della informazione



Codici binari

Codice binario

Codice binario - Funzione dall'insieme delle 2^n configurazioni di n bit ad un insieme di M informazioni (*simboli alfanumerici, colori, eventi, stati interni,* ecc.).
Condizione necessaria per la codifica: $2^n \geq M$



Proprietà di un codice

Il codice è una rappresentazione convenzionale dell'informazione.

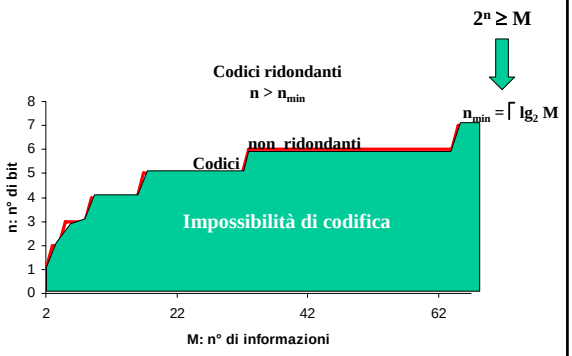
La scelta di un codice è condivisa da sorgente e destinazione ed ha due gradi di libertà:
- **il numero di bit (qualsiasi, a patto che sia $2^n \geq M$)**
- **l'associazione tra configurazioni e informazioni**

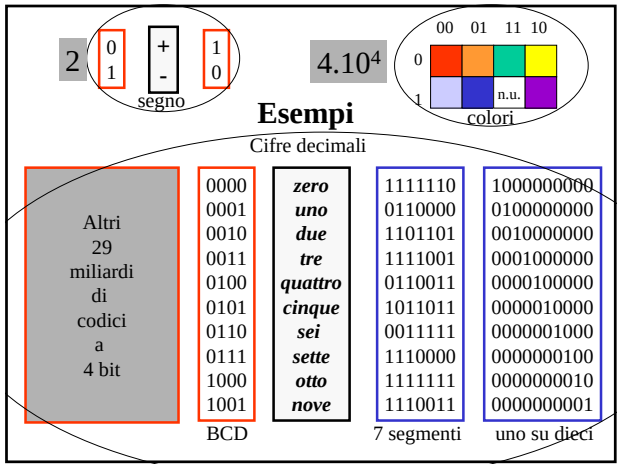
A parità di n e di M le associazioni possibili sono

$$C = 2^n! / (2^n - M)!$$

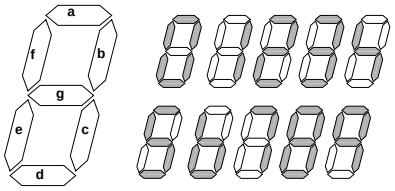
n = 1, M = 2	C = 2
n = 2, M = 4	C = 24
n = 3, M = 8	C = 64.320
n = 4, M = 10	C = 29.000.000.000

Codici ridondanti e non ridondanti



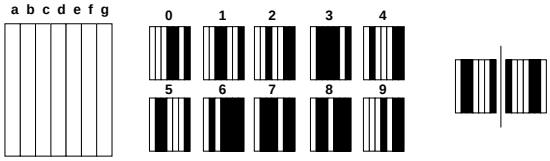


Codice a 7 segmenti



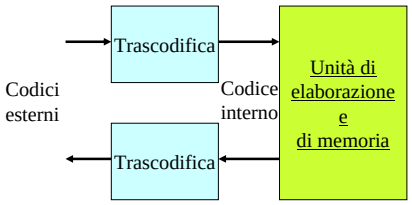

```
      a b c d e f g
zero  1 1 1 1 1 1 0
uno   0 1 1 0 0 0 0
ecc.
```

Universal Product Code



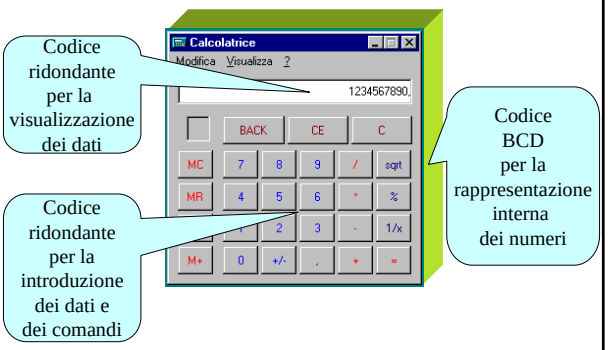
Trascodifica

La trascodifica sul percorso dei dati

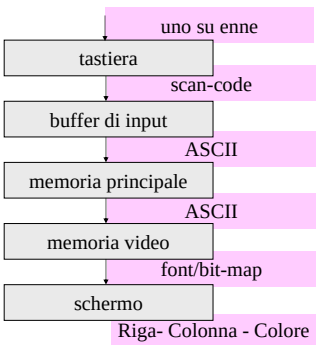


Il codice interno è di norma non ridondante per minimizzare il n° di bit da elaborare e da memorizzare.
Il codice esterno è di norma ridondante, per semplificare la generazione e la interpretazione delle informazioni, e standard, per rendere possibile la connessione di macchine (o unità di I/O) fatte da Costruttori diversi.

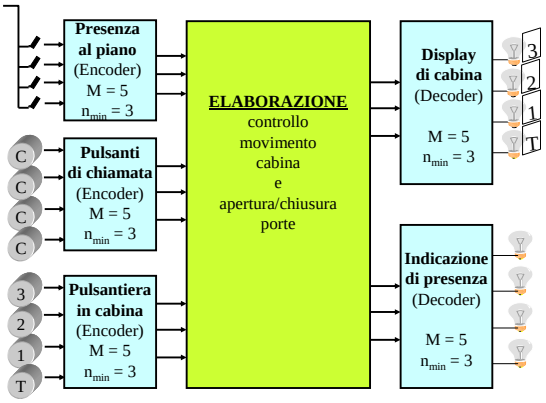
La calcolatrice tascabile



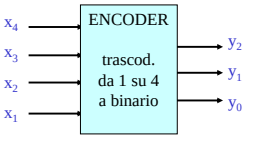
Esempio : un editor



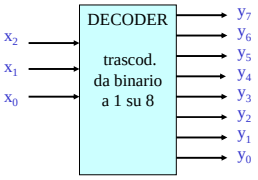
Input/output di un ascensore



Le due trascodifiche



$x_4\,x_3\,x_2\,x_1$	$y_2\,y_1\,y_0$
0 0 0 0	0 0 0
0 0 0 1	0 0 1
0 0 1 0	0 1 0
0 1 0 0	0 1 1
1 0 0 0	1 0 0



$x_2\,x_1\,x_0$	$y_7\,y_6\,y_5\,y_4\,y_3\,y_2\,y_1\,y_0$
0 0 0	0 0 0 0 0 0 0 1
0 0 1	0 0 0 0 0 0 1 0
0 1 0	0 0 0 0 0 1 0 0
0 1 1	0 0 0 0 1 0 0 0
1 0 0	0 0 0 1 0 0 0 0
1 0 1	0 0 1 0 0 0 0 0
1 1 0	0 1 0 0 0 0 0 0
1 1 1	1 0 0 0 0 0 0 0

Codici proprietari e standard

Codice proprietario - Codice fissato da un Costruttore per mettere in comunicazione apparati da lui realizzati
• L'uso di codici proprietari ottimizza le prestazioni e protegge il mercato di certe apparecchiature.

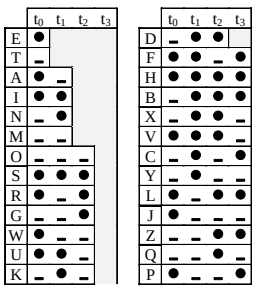
Esempi: Linguaggio Assembler, Periferiche, Telecomando TV

Codice standard - Codice fissato da norme internazionali (***de iure***) o dal costruttore di una macchina utile per tutti gli altri (***de facto***).
• L'uso di codici standard nelle unità di I/O consente di collegare macchine fatte da costruttori diversi

Esempi: Stampanti e Calcolatori, Calcolatori e Calcolatori

2.2 La codifica dei caratteri

La codifica Morse (1830)



Caratteristiche:

•Lunghezza variabile

•Stringhe separate da pause

•Efficiente per l'uso da parte di operatori umani

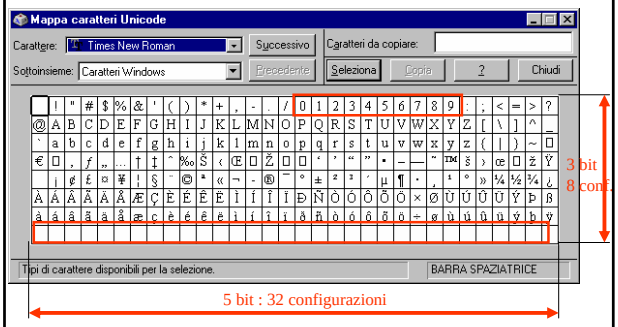
•Difficoltoso il progetto di ricetrasmettitori automatici

Stringhe di uguale lunghezza - Baudot (1874): 5 bit

I 95 simboli di “testo”: ASCII a 7 bit (1967)

	000	001	010	011	100	101	110	111
0000	caratteri di controllo		SP	0	@	P	'	p
0001			!	1	A	Q	a	q
0010			"	2	B	R	b	r
0011			#	3	C	S	c	s
0100			$	4	D	T	d	t
0101			%	5	E	U	e	u
0110			&	6	F	V	f	v
0111			'	7	G	W	g	w
1000			(	8	H	X	h	x
1001			)	9	I	Y	i	y
1010			*	:	J	Z	j	z
1011			+	;	K	[	k	{
1100			,	<	L	\	l	|
1101			-	=	M	]	m	}
1110			.	>	N	^	n	~
1111			/	?	O	_	o	DEL

Codice ASCII esteso (8 e 16 bit)

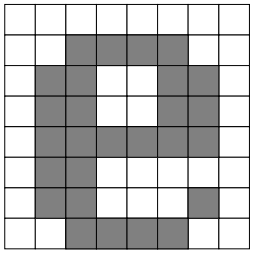


Lo standard Unicode (16 bit) codifica
in binario i caratteri di tutte le lingue!

Bit map: un codice d’uscita ridondante per simboli alfanumerici

Stampanti ad impatto: ASCII

Stampanti laser, a getto, monitor: BITMAP



Matrice di pixel: ad es. 8x8

Bianco/nero:
1 pixel, 1 bit

Tonalità:
1 pixel, 8 bit

Colori RGB:
1 pixel, 3x8 bit

Font

2.3 La codifica dei numeri

Rappresentazione dei numeri

- Esterna: BCD, ASCII, Unicode
- Interna: Sistema di numerazione in base 2

Numeri in base B

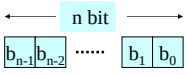
1) Rappresentazione:

$a_{n-1} \ldots\ldots a_0 , a_{-1} \ldots\ldots a_{-m}$ — $a_i \in \{0, 1, \ldots, (B-1)\}$

2) Valore:

$$(N)_B = (a_{n-1}.B^{n-1} + \ldots + a_0.B^0 + a_{-1}.B^{-1} + a_{-2}.B^{-2} + \ldots a_{-m}.B^{-m})$$

Il sistema di numerazione in base 2 (il caso dei numeri naturali < 2^n)



$$(N)_2 = b_{n-1}.2^{n-1} + b_{n-2}.2^{n-2} + \ldots + b_0.2^0$$

N_{10}	N_2	N_{10}	N_2
0	0000	8	1000
1	0001	9	1001
2	0010	10	1010
3	0011	11	1011
4	0100	12	1100
5	0101	13	1101
6	0110	14	1110
7	0111	15	1111

Lunghezza della stringa in base 2 e in base 10

Dato un numero decimale con m cifre

$$0 \le (N)_{10} \le 10^m - 1$$

per la sua rappresentazione binaria deve essere $2^n > 10^m$ e quindi

$$n = \lceil (m \times \log_2 10) \rceil \approx \lceil (3{,}32\ m) \rceil$$

Conversione di base

Conversioni da base 2 a base 10 e viceversa di numeri naturali

ESEMPIO: 100110

0 +
2 +
4 +
0 +
0 +
32 =
38

Conversione da base 2 a base 10

$(N)_{10} = (b_{n-1}.2^{n-1} + b_{n-2}.2^{n-2} + \ldots + b_1.2^1 + b_0.2^0)_{10}$

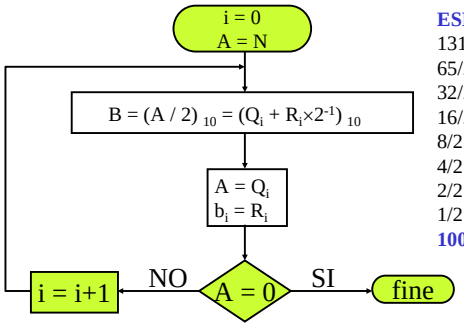
Conversione da base 10 a base 2
Osservazione preliminare:

$(N)_{10} = (b_{n-1}.2^{n-1} + b_{n-2}.2^{n-2} + \ldots + b_1.2^1 + b_0.2^0)_{10}$

$(N)_{10}/2 = (b_{n-1}.2^{n-2} + b_{n-2}.2^{n-3} + \ldots + b_1.2^0) + (b_0.2^{-1})_{10}$

$= Q \qquad + R.\ 2^{-1}$

Conversione di un numero naturale N da base 10 a base 2



ESEMPIO: 131

131/2 = 65 + 1.0,5
65/2 = 32 + 1.0,5
32/2 = 16 + 0
16/2 = 8 + 0
8/2 = 4 + 0
4/2 = 2 + 0
2/2 = 1 + 0
1/2 = 0 + 1.0,5
10000011

Altre rappresentazioni di numeri binari

- **Sistema esadecimale**: B =16
  cifre: 0,1,..,9,a,b,c,d,e,f
  codice binario: 0 = 0000, 1 = 0001, …, f = 1111
  n° di bit per cifra: 4

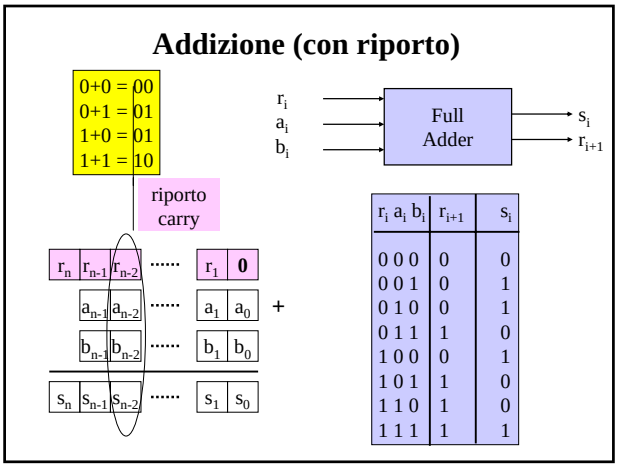
ESEMPIO: 11000100 → 1100-0100 → C4

- **Sistema ottale**: B = 8,
  cifre: 0, 1, …,7
  codice OCTAL: 0 = 000, …, 7 = 111
  n° di bit per cifra: 3

ESEMPIO: 11000100 → 11-000-100 → 304

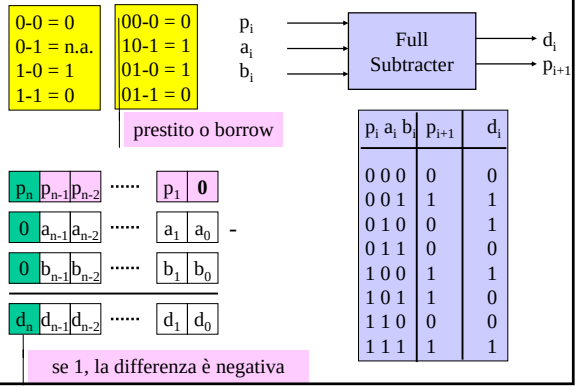
Operazioni aritmetiche

Addizione (con riporto)



r_i a_i b_i	r_{i+1}	s_i
0 0 0	0	0
0 0 1	0	1
0 1 0	0	1
0 1 1	1	0
1 0 0	0	1
1 0 1	1	0
1 1 0	1	0
1 1 1	1	1

Sottrazione (con prestito)



p_i a_i b_i	p_{i+1}	d_i
0 0 0	0	0
0 0 1	1	1
0 1 0	0	1
0 1 1	0	0
1 0 0	1	1
1 0 1	1	0
1 1 0	0	0
1 1 1	1	1

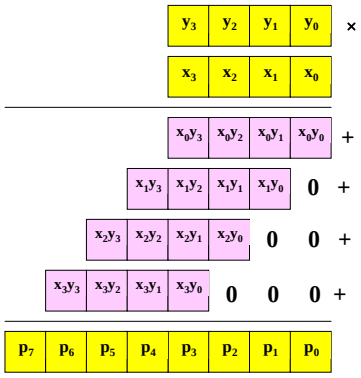
Moltiplicazione

0.0 = 0
0.1 = 0
1.0 = 0
1.1 = 1

P = Y.X
$= Y.(x_{n-1}.2^{n-1} + \ldots + x_1.2^1 + x_0.2^0)$
$= (Y.x_{n-1}).2^{n-1} + \ldots + (Y.x_1).2^1 + (Y.x_0).2^0$
$= P_{n-1}.2^{n-1} + \ldots + P_1.2^1 + P_0.2^0$

$P_i = (Y.x_i) = (y_{n-1}.x_i).2^{n-1} + \ldots + (y_0.x_i).2^0$

Moltiplicazione (shift and add)



Rappresentazione dei numeri razionali

•Come coppia di interi (più un bit per il segno)

i_n	i_{n-1}	...	i_0	f_1	f_2	...	f_n

s	i_n	i_{n-1}	...	i_0	f_1	f_2	...	f_n

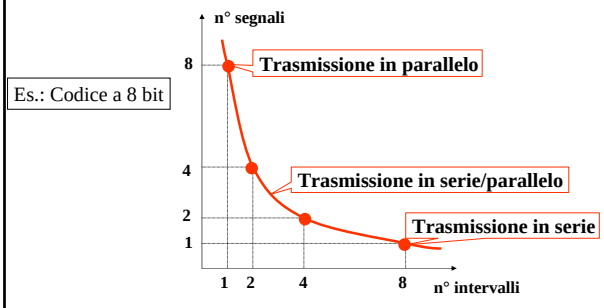
•Notazione *scientifica*

s	m_n	m_{n-1}	...	m_0	s	e_n	e_{n-1}	...	e_0

2.4 Trasmissione

Modalità

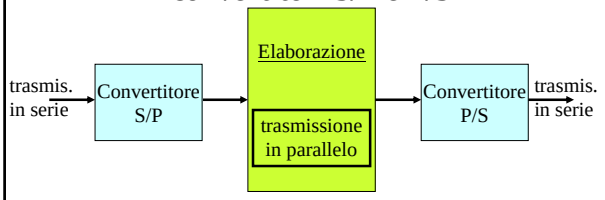
Modalità di trasmissione dei bit: compromesso spazio/tempo



Es.: Codice a 8 bit

Esempio: processori Intel

Modalità di trasmissione dei bit: convertitori S/P e P/S

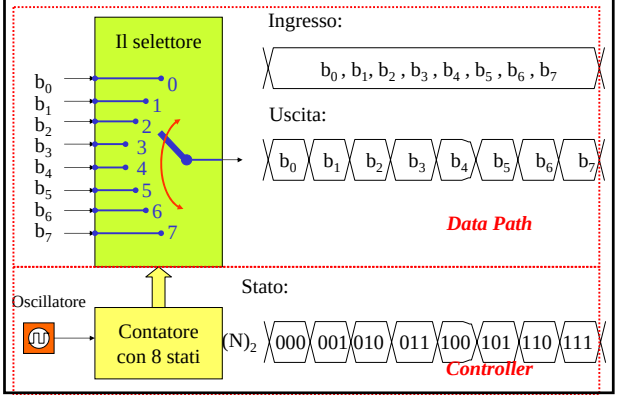


La <u>modalità di trasmissione all'interno</u> della macchina è di norma in parallelo (per massimizzare la velocità di elaborazione)

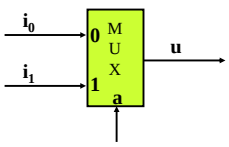
La <u>modalità di trasmissione all'esterno</u> della macchina è di norma in serie (per minimizzare la complessità del supporto fisico)

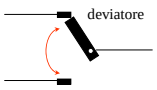
Esempi: interfaccia di tastiera, interfaccia video

La conversione P/S di un byte



La serializzazione di due bit

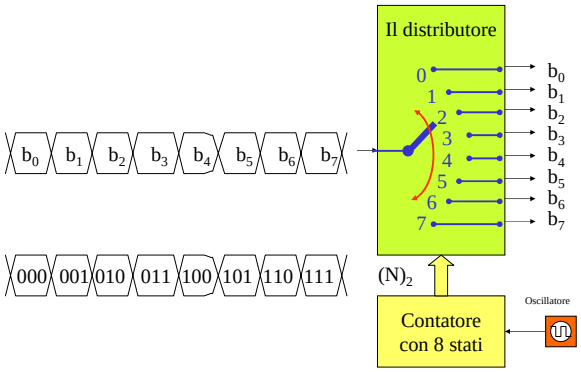




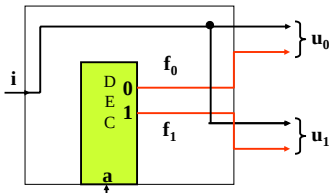
a	i_0	i_1	u
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

***se* a=0 *allora* u=i_0**
***altrimenti* u=i_1**

Conversione S/P di un byte



La distribuzione di due bit



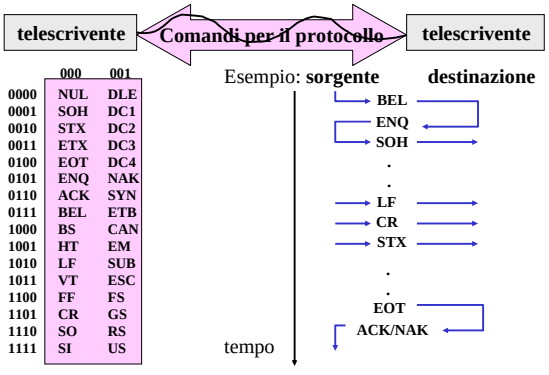
a	f_0	f_1
0	1	0
1	0	1

Contatore con 2 stati

Il Decoder genera **2** "flag di validità", di cui uno solo alla volta ha valore 1.
L'uscita che riceve tale valore è la destinazione del bit d'ingresso **i**

Protocolli

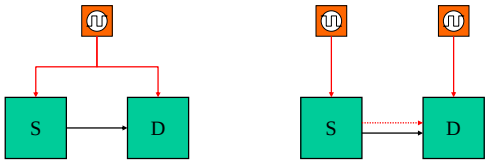
Modalità di controllo (ASCII a 7 bit) : codifica dei comandi e protocollo di scambio



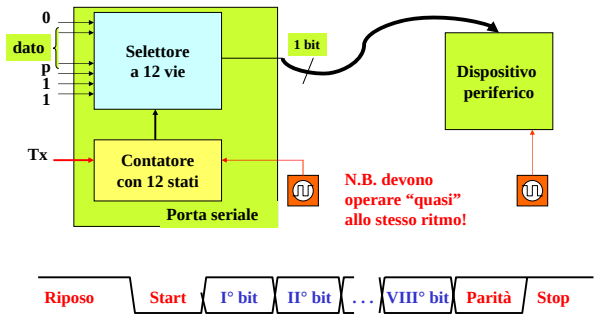
Sincronizzazione

La <u>destinazione</u> deve sapere in quali istanti di tempo i valori presenti sul canale sono significativi.
Si hanno due casi:

"accoppiamento stretto" ***"accoppiamento lasco"***



Comunicazione asincrona di un byte: il protocollo RS232



2.5 Protezione

Disturbi e Guasti

- *linea di trasmissione*
- *unità di memoria*

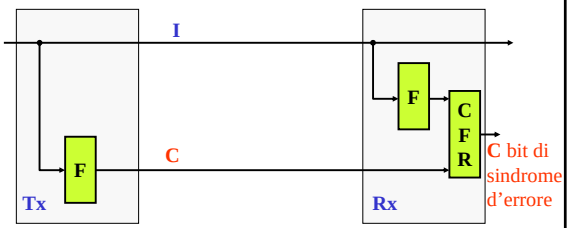
Obiettivo: riconoscere alla destinazione le configurazioni modificate.
Condizione necessaria: **il codice deve essere ridondante**

- se arriva una configurazione "non utilizzata" la destinazione ha la **certezza** della sua non integrità;
- se arriva una configurazione "utilizzata" la destinazione ha solo una certa **probabilità** che sia integra.

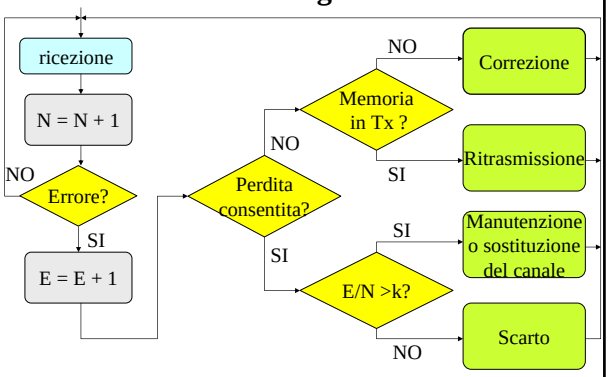
Ulteriore condizione - Le configurazioni non utilizzate devono essere le modifiche più probabili delle configurazioni utilizzate.

Codici separabili: rilevazione di errori

bit di informazione (*information bits*) | bit di controllo (*check bits*)



Gestione degli errori



Codici con rilevazione di errori

L'ipotesi degli errori indipendenti

Consideriamo una stringa di ***n*** bit e supponiamo che l'evento di modifica di un bit (o *errore*) da parte di un disturbo
a) sia indipendente dalla posizione del bit nella stringa;
b) si verifichi con probabilità pari a ***p*** *(tasso di errore)*.
La probabilità che la stringa ricevuta contenga ***e*** errori è data da:

$$P_e = \binom{n}{e} \cdot p^e \cdot (1-p)^{n-e}$$

Esempio:
p = *1 %*
N.B. molto alto!

n	P_0	P_1	P_2	P_3
8	*92,27 %*	*7,46 %*	*0,26 %*	*0,005 %*
16	*85.14 %*	*13,76 %*	*1,04 %*	*0,049 %*

Per n = 8 le modifiche più probabili riguardano un solo bit
Per n = 16 le modifiche più probabili riguardano uno o due bit

Distanza minima di un codice

Distanza fra due configurazioni binarie di n bit: D(A,B) - Numero di bit omologhi che hanno valore diverso.

Esempi: D(100,101) = 1; D(011,000) = 2; D(010,101) = 3

Distanza minima di un Codice C: DMIN (C) - Valore minimo della distanza tra due qualsiasi delle configurazioni utilizzate.

Esempi: DMIN (Codice ASCII) = 1; DMIN (Codice semaforo) = 2

- **I codice non ridondanti hanno DMIN=1.**
- **I codice ridondanti possono avere DMIN > 1.**

Esempio: DMIN (UPC) = 2

Distanza minima e rilevazione degli errori

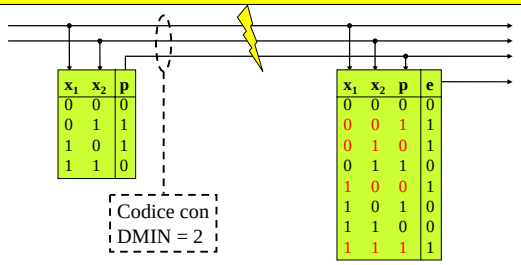
• Un codice per la rilevazione di tutti i possibili errori singoli, o **SEDC (Single Error Detection Code)**, deve non utilizzare tutte le configurazioni che distano "uno" da ciascuna delle configurazioni utilizzate.

Un codice SED deve dunque avere almeno **DMIN = 2** .

• Un codice per la rilevazione di modifiche su k bit deve avere almeno **DMIN = k+1** .

Il bit di parità : una semplice modalità per ottenere la rilevazione di errori singoli

Bit di parità p - bit che la sorgente aggiunge ad una stringa di bit di codifica al fine di renderne **pari** il n° di "uni".
Errore di parità e - bit che la destinazione pone a 1 se e solo se riceve una configurazione con un numero **dispari** di "uni".

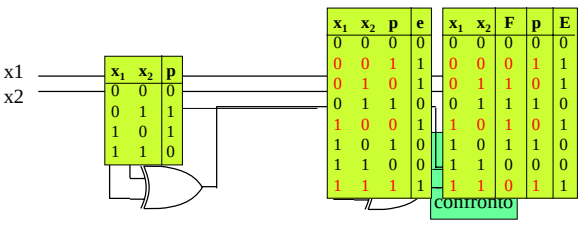


x_1	x_2	p
0	0	0
0	1	1
1	0	1
1	1	0

Codice con DMIN = 2

x_1	x_2	p	e
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Calcolo del bit di parità



x_1	x_2	p
0	0	0
0	1	1
1	0	1
1	1	0

x_1	x_2	p	e
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

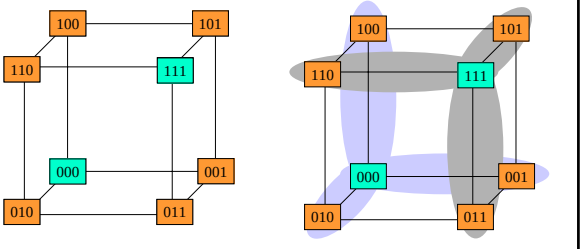
x_1	x_2	F	p	E
0	0	0	0	0
0	0	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1

p = F(x1, x2)

e = E(x1, x2, p)
= F(F(x1, x2), p)
•Funzione composta
•Disposizione in serie

Codici con correzione di errori

La correzione di errori singoli (esempio)



Tx trasmette
o NO = 000 o SI = 111

Se $P_1 >> P_2$ ogni configurazione errata può essere corretta

A causa dei disturbi Rx può ricevere una terna qualsiasi

Distanza minima e correzione degli errori

Il codice dell'esempio precedente ha DMIN=3.

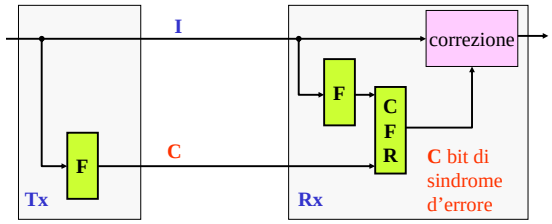
• Ogni **SECC** (Single Error Correction Code) deve avere **DMIN ≥ 3**.

• Un codice con **DMIN = 2k+1** rileva **2k** errori e può correggerne fino a **k** .

Di solito si corregge un solo bit e si usa la ridondanza introdotta per valutare la "qualità" del canale (manutenzione/sostituzione)

Codici separabili: correzione di errori

Hamming (Bell Labs, 1950) ha dimostrato che per correggere gli errori singoli su informazioni codificate con **I** bit occorrono **C** bit di controllo tali che $2^C \geq I + C + 1$.



Le 2^C configurazioni delle **sindromi di errore** devono indicare se non c'è errore (1 situazione) e se c'è, dov'è (I + C situazioni).