I CODICI

I codici

Codice (C): insieme di **simboli** (**alfabeto**) e di regole per generare **parole** che rappresentano gli elementi di un insieme di entità (C').

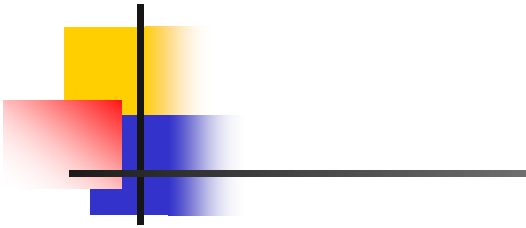
Simboli di C	**Parole di C**	**Entità di C'**
Cifre decimali (0,1,2,..9)	Numeri naturali (p.e. 12)	I I I I I I I I I I I I
Bit (0, 1)	Stringhe di 4 bit (p.e. 1100)	I I I I I I I I I I I I
Lettere dell' alfabeto italiano (a, b, ..., z)	Parole italiane (p.e. dodici)	I I I I I I I I I I I I

I codici (cont.)

Codifica: operazione per cui ad una parola di C viene associato un elemento di C’.

Decodifica: operazione per cui ad un elemento di C’ si fa corrispondere una parola di C.

Codice non ambiguo: codice in cui la corrispondenza tra le parole di C e gli elementi di C' è univoca.



Se si indica con

b: il numero di simboli differenti usati per identificare le parole di C (nel caso di uso di un sistema di numerazione, b identifica la sua base);

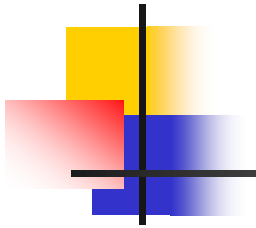
n: la lunghezza (costante) delle parole di C;

m: il minimo valore di n che rende **non ambiguo** il codice C per codificare gli elementi di C';

N = |C'|

allora $b^m >= N$

(dove > = indica maggiore o uguale)



Un codice si dice:

irridondante se $n = m$

ridondante se $n > m$

ambiguo se $n < m$

Codifica binaria: distanza di Hamming

Si definisce **distanza di Hamming d(x,y)** fra due parole (x,y) di un codice (C)

il numero di posizioni (bit) per cui differiscono

d(10010 , 01001) = 4

d(11010 , 11001) = 2

La **distanza minima** di un codice e' allora

$d_{min} = \min(d(x,y))$ per ogni x e y appartenenti a C e diversi tra loro

Ambiguità e ridondanza

codici irridondanti	h = 1 (e n = m)
codici ridondanti	h >= 1 (e n > m)
codici ambigui	h = 0

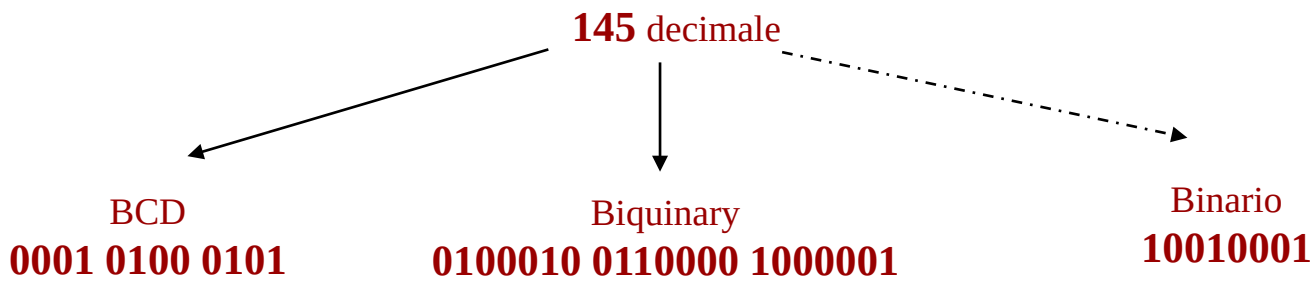
Esempi di calcolo distanza di Hamming

Parole di C	Prima codifica	Seconda codifica	Terza codifica	Quarta codifica	Quinta codifica
alfa	000	0000	00	0000	110000
beta	001	0001	01	0011	100011
gamma	010	0010	11	0101	001101
delta	011	0011	10	0110	010110
mu	100	0100	00	1001	011011
	h = 1 Irr.	h = 1 Rid.	h = 0 Amb.	h = 2 Rid. *Rivela errori*	h = 3 Rid. *Rivela e corregge errori*

Codici binari per rappresentare decimali

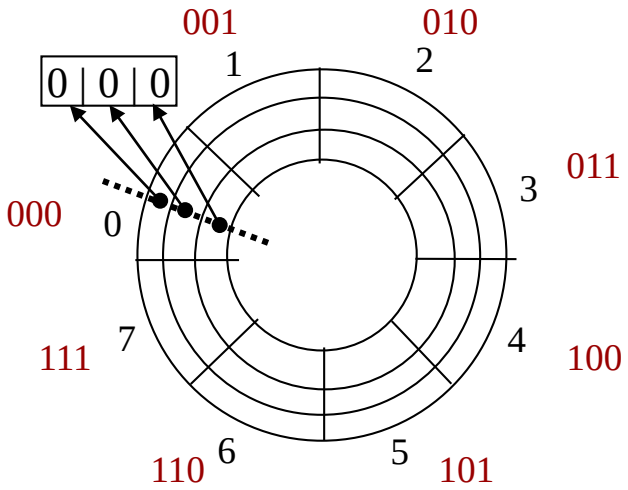
Si usano per rappresentare le dieci cifre decimali in binario
dato che $2^3 < 10 < 2^4$ occorrono almeno 4 bits

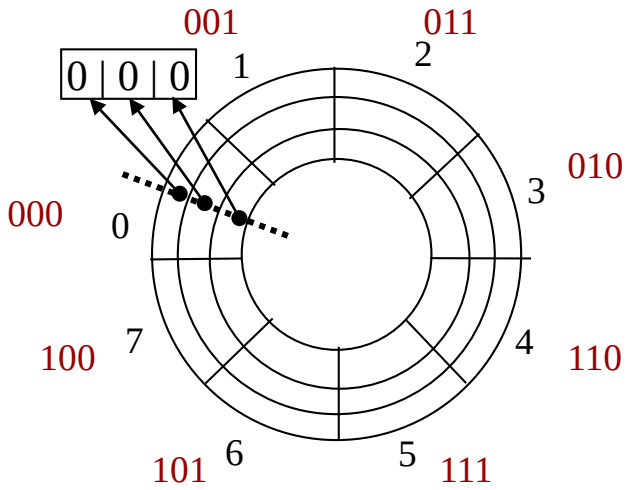
Decimale	Binario	BCD	Eccesso-3	Biquinary	1 di 10
0	0	0000	0011	0100001	0000000001
1	1	0001	0100	0100010	0000000010
2	01	0010	0101	0100100	0000000100
3	11	0011	0110	0101000	0000001000
4	100	0100	0111	0110000	0000010000
5	101	0101	1000	1000001	0000100000
6	110	0110	1001	1000010	0001000000
7	111	0111	1010	1000100	0010000000
8	1000	1000	1011	1001000	0100000000
9	1001	1001	1100	1010000	1000000000



Codice Gray *(codice riflesso)*

Codici binari in cui le rappresentazioni di valori consecutivi variano per un solo bit





Dec.	Binario	GRAY-2	GRAY-3
0	000	0 0	0 00
1	001	0 1	0 01
2	010	1 1	0 11
3	011	1 0	0 10
4	100		1 10
5	101		1 11
6	110		1 01
7	111		1 00

E' possibile costruire ricorsivamente un codice gray ad *n+1* bits partendo da uno ad *n* bits utilizzando i seguenti passi:

- Le prime 2^n parole del codice ad n+1 bits sono uguali a quelle del codice ad *n* bits estese (MSB) con lo '0'
- Le seconde 2^n parole del codice ad n+1 bits sono uguali a quelle del codice ad *n* bits ma scritte in ordine ***inverso*** (riflesso) ed estese (MSB) con '1'

Codice ASCII - codici carattere

I codici carattere vengono usati per rappresentare in binario i simboli non numerici usati nella scrittura (ALFABETO , punteggiatura, parentesi ...) ed anche comandi standard provenienti componenti di I/O (tastiera, stampante ,...)

Il più diffuso e' il codice **ASCII** (American Standard Code for Information Interchange) composto da parole di lunghezza fissa a **7 bit** (128 combinazioni)

Una estensione successiva ad 8 bit del codice ASCII fu sviluppata dall'IBM e prende il nome di **EBCDIC** (Extended Binary-Coded Decimal Interchange Code)

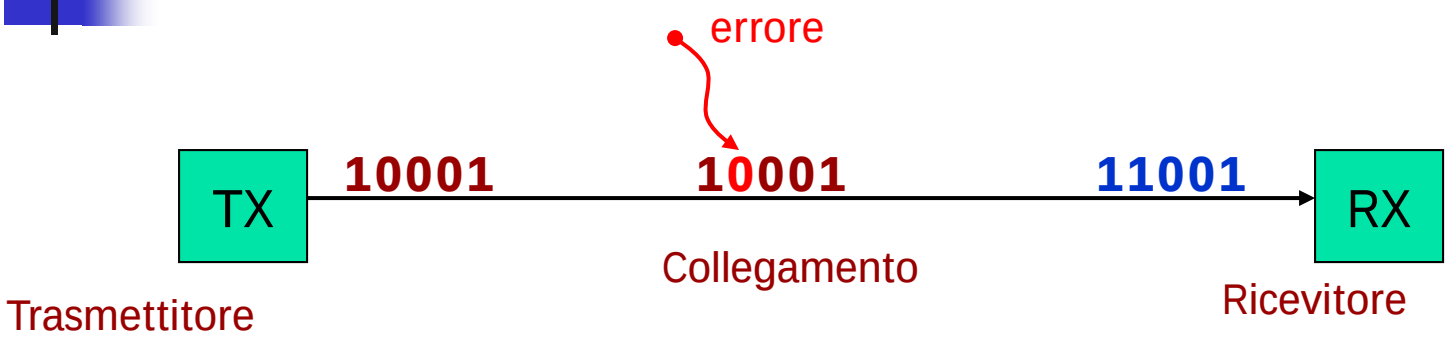
Ulteriori estensioni hanno permesso di incorporare nella rappresentazione i simboli utilizzati negli alfabeti di diverse lingue (cinese, russo..)

Codice ASCII - codici carattere

righe	colonne	0	1	2	3	4	5	6	7
b4b3b2b1	b7b6b5	000	001	001	011	100	101	110	111
0	0000	NUL	DLE	SP	0	@	P	'	p
1	0001	SHO	DC1	!	1	A	Q	a	q
2	0010	STX	DC2	"	2	B	R	b	r
3	0011	ETX	DC3	#	3	C	S	c	s
4	0100	EOT	DC4	$	4	D	T	d	t
5	0101	ENQ	NAK	%	5	E	U	e	u
6	0110	ACK	SYN	&	6	F	V	f	v
7	0111	BEL	ETB	'	7	G	W	g	w
8	1000	BS	CAN	(	8	H	X	h	x
9	1001	HT	MEM	)	9	I	Y	i	y
10	1010	LF	SUB	*	:	J	Z	j	z
11	1011	VT	ESC	+	;	K	[	k	{
12	1100	FF	FS	,	<	L	\	l	!
13	1101	CR	GS	_	=	M	]	m	}
14	1110	SO	RS	.	>	N	^	n	~
15	1111	SI	US	/	?	O	-	o	DEL

Tab. I.5 - Codice ASCII.

Codici rivelatori di errore (error detecting codes)



Per "rivelare" errori di trasmissione il sistema che invia dati introduce ridondanza nelle informazioni trasmesse.

Codice (n, k) con n> k => codice con parole di lunghezza n di cui k bit di *informazione*

Un codice *rivelatore di errore* ha la proprietà che la generazione di un errore su una parola appartenente al codice produce una *parola non appartenente al codice*

Si definisce **peso di un errore** il numero di bit "corrotti" durante la trasmissione

In sistemi binari ho due soli casi di errore
Trasmetto 0 Ricevo 1
Trasmetto 1 Ricevo 0

Codici rivelatori di errore (error detecting codes)

Si definisce **distanza di Hamming d(x,y)** fra due parole (x,y) di un codice (C) il numero di posizioni (bit) per cui differiscono

d(10010 , 01001) = 4

d(11010 , 11001) = 2

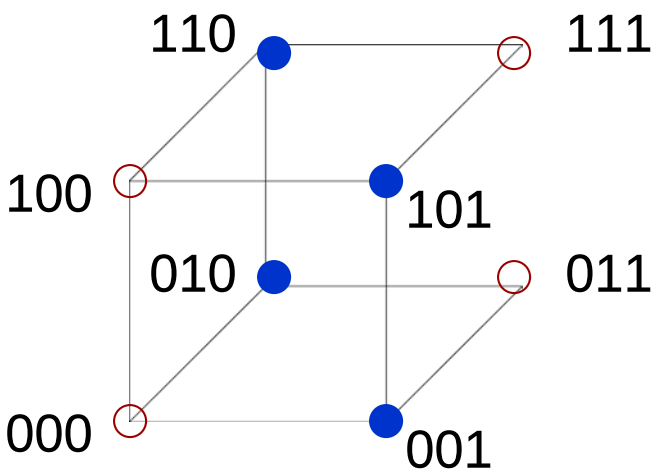
La **distanza minima** di un codice e' allora

$d_{min} = \min(d(x,y))$ per ogni x e y appartenenti a C e diversi tra loro

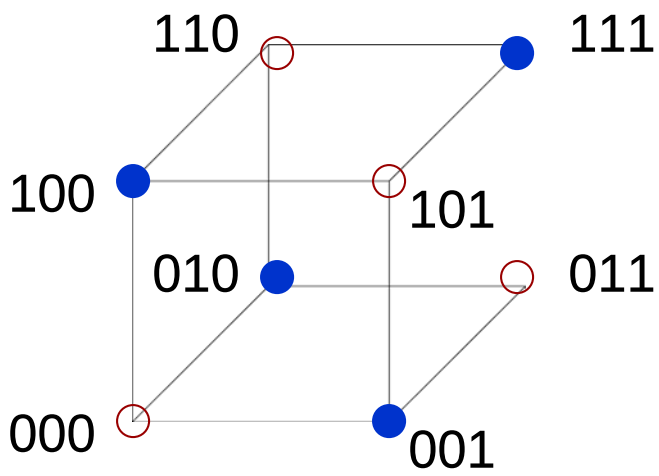
Un codice a distanza minima d
e' capace di rivelare errori di peso <= **d-1**

Codici rivelatori di errore (error detecting codes)

	Codice 1	Codice 2
A =>	000	000
B =>	100	011
C =>	011	101
D =>	111	110



$d_{min}=1$



$d_{min}=2$

○ Parole del codice (legali)

● Parole non appartenenti al codice

Codice di parità (distanza minima 2)

Posso costruire un codice a d_{min} pari a 2 utilizzando le seguenti espressioni:

$b_1 + b_2 + b_3 + ….. + b_n + p = 0$ parità oppure

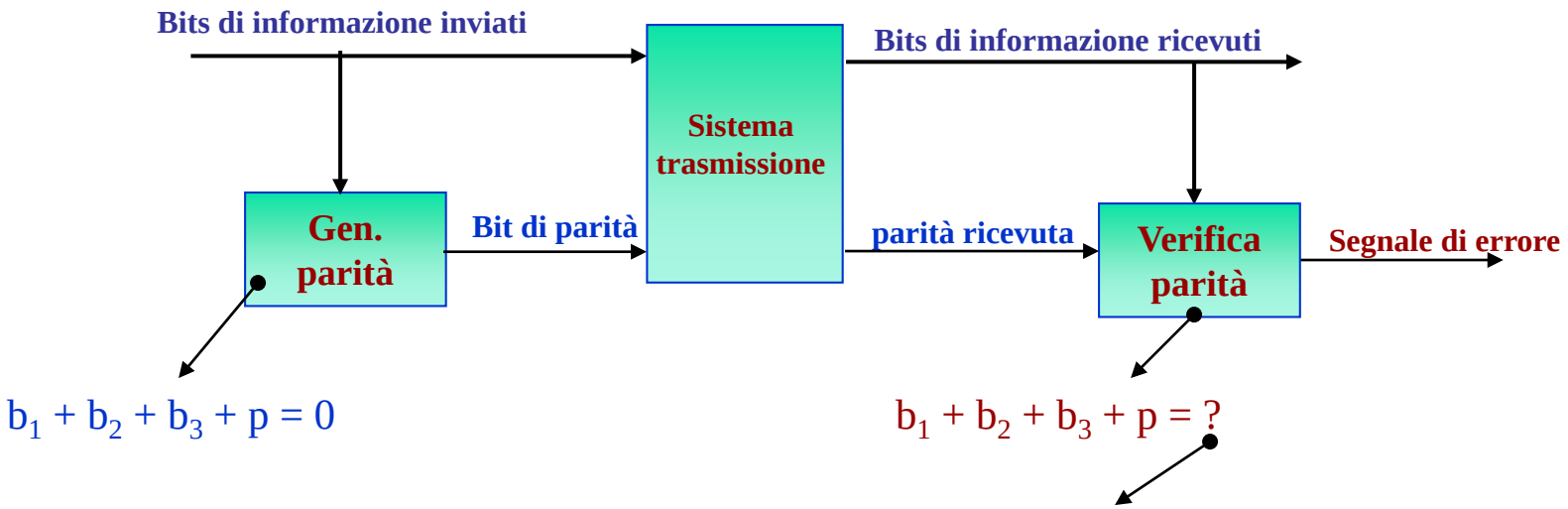
$b_1 + b_2 + b_3 + ….. + b_n + p = 1$ disparità

Dove
- n è il numero di bit usati per rappresentare in binario gli oggetti (informazione),
- + e' l'operatore di somma modulo 2
- p il bit di "parita/disparità" da aggiungere a quelli di informazione per costruire parole del codice

Bit di informazione	Parità	Disparità
000	000 0	000 1
001	001 1	001 0
010	010 1	010 0
011	011 0	011 1
100	100 1	100 0
101	101 0	101 1
110	110 0	110 1
111	111 1	111 0

E' un codice di distanza minima pari a 2
che permette di rivelare errori di peso 1 (single error)

Codice di parità (distanza minima 2)



- Se pari a 0 ***non*** ci sono stati singoli errori
- Se pari a 1 si è verificato un singolo errore

Es. devo trasmettere l'informazione 101
Il generatore di parità calcola il bit di parità 1 + 0 + 1 + p = 0 cioè p = 0 e trasmetto 1010

Il ricevitore riceve 1110 ne verifica la parità 1 + 1 + 1 + 0 = 1 <> da 0 quindi si è verificato un errore

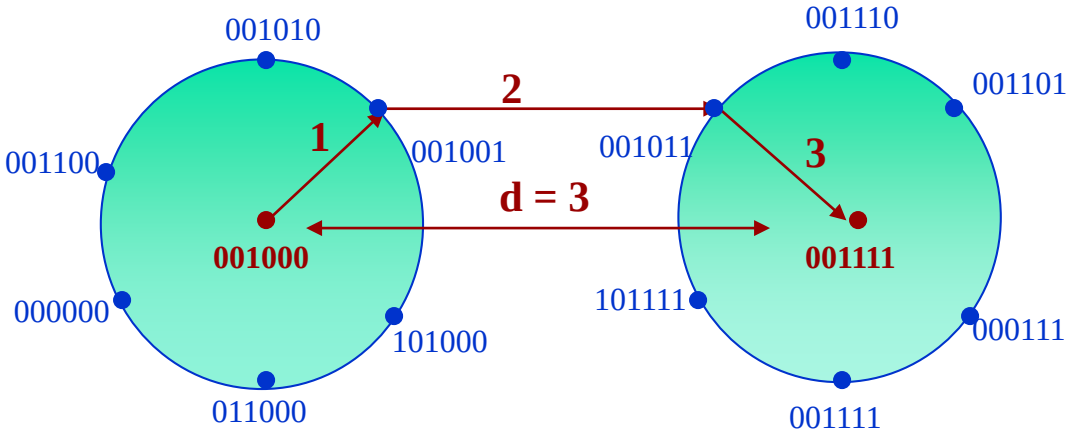
Se avessi ricevuto 1111 => 1+1+1+1 = 0 tutto OK? , niente singoli errori !! (I doppi sono sfuggiti al check)

Codici correttori di errore (error correcting codes)

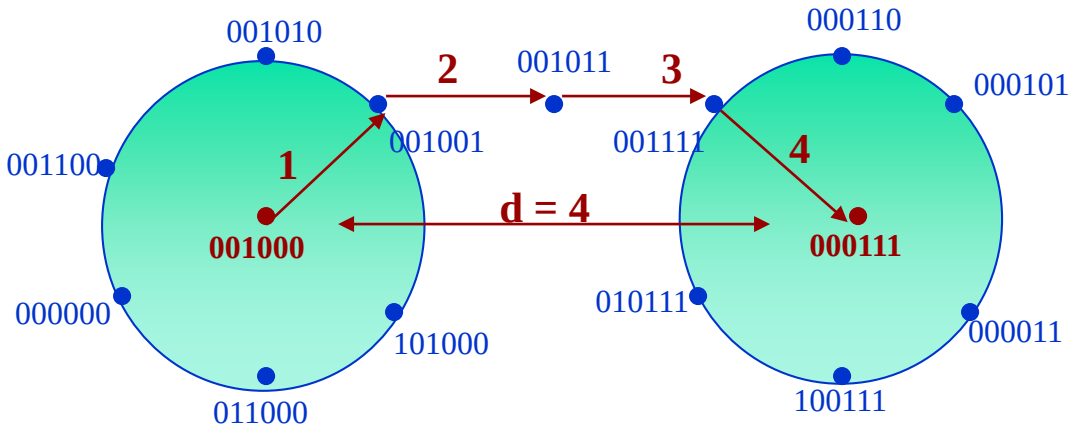
E' un codice capace di ***correggere*** gli errori generati durante la trasmissione

Dato un codice a distanza minima **d** esso ha una **capacita' di correzione** di errori di peso <= **INTINF((d-1)/2)**

Quindi un codice a distanza minima 3 può correggere errori di peso = 1

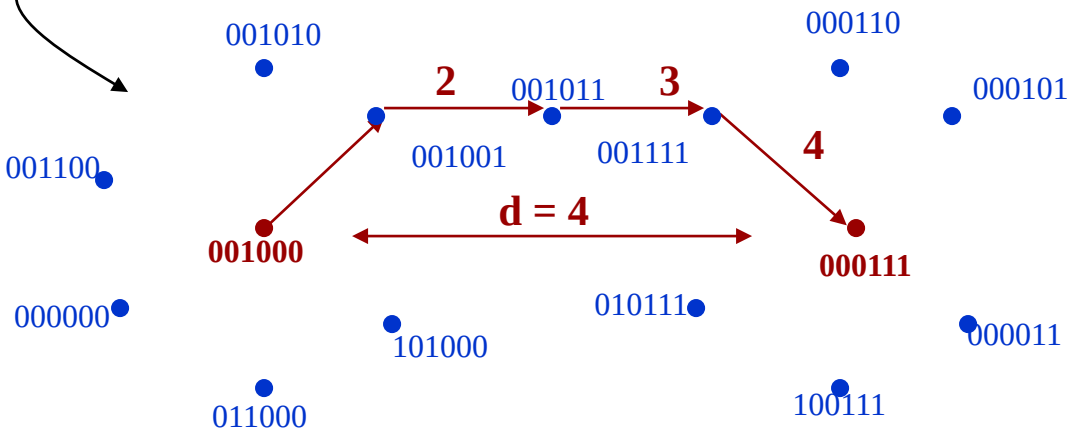


Codici di correzione e rivelazione



Un codice a distanza minima 4 può correggere errori di peso 1 (single error) e rivelare errori di peso 2 (double error).

In alternativa puo' rivelare errori di peso <= 3



Codici Hamming(1)

• Metodo per la costruzione di codici a distanza minima 3

• per ogni *i* e' possibile costruire un codice a 2^i -1 bit con *i* bit di parità (check bit) e 2^i -1-i bit di informazione.

• I bit in posizione corrispondente ad una **potenza di 2** (1,2,4,8,...) sono **bit di parità** i rimanenti sono bits di informazione

• Ogni bit di parità controlla la correttezza dei bit di informazione la cui posizione, espressa in binario, ha un 1 nella potenza di 2 corrispondente al bit di parità

Esempio con quattro bit di informazione

$p_1\ p_2\ I_3\ p_4\ I_5\ I_6\ I_7$

$p_1 + I_3 + I_5 + I_7 = 0$

$p_2 + I_3 + I_6 + I_7 = 0$

$p_4 + I_5 + I_6 + I_7 = 0$

$(3)_{10} = (0\ 1\ 1)_2$
$(5)_{10} = (1\ 0\ 1)_2$
$(6)_{10} = (1\ 1\ 0)_2$
$(7)_{10} = (1\ 1\ 1)_2$

$2^2\ 2^1\ 2^0$

Codici Hamming(2)

p_1	p_2	I_3	p_4	I_5	I_6	I_7
1	2	3	4	5	6	7

← posizione

Gruppi

			X	X	X	X
	X	X			X	X
X		X		X		X

$p_4 + I_5 + I_6 + I_7 = 0$

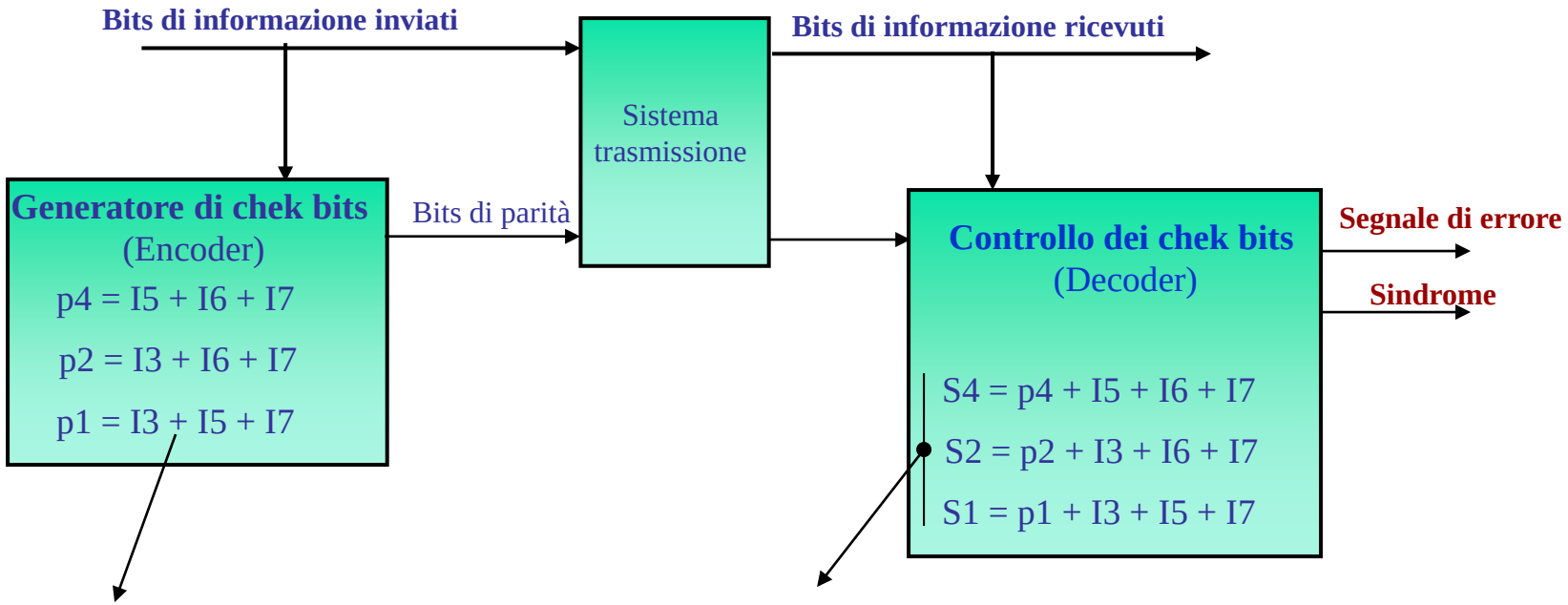
$p_2 + I_3 + I_6 + I_7 = 0$

$p_1 + I_3 + I_5 + I_7 = 0$

p_i: bit di parità

I_i: bit di informazione

Circuito di EDAC (Error Detection And Correction)



Somma modulo 2

Se i tre bit di sindrome sono pari a 0 non ci sono stati errori altrimenti il loro valore da' la posizione del bit errato