Le Macchine Combinatorie

Calcolatori Elettronici

L'algebra di Boole - richiami

Operazioni fondamentali sui bit:

x	y	x AND y
0	0	0
0	1	0
1	0	0
1	1	1

Congiunzione

x AND y si indica anche con $x \cdot y$

x	y	x OR y
0	0	0
0	1	1
1	0	1
1	1	1

Disgiunzione

x OR y si indica anche con $x + y$

x	NOT x
0	1
1	0

Negazione

NOT x si indica anche con $\bar{x}$

L'algebra di Boole - alcune proprietà (1)

- **Proprietà commutativa**:
  x AND y = y AND x   x OR y = y OR x

- **Proprietà associativa:**
  (x AND y) AND z = x AND (y AND z)
  (x OR y) OR z = x OR (y OR z)
  per la propr. associativa posso definire AND e OR a più di due operandi (es. x AND y AND z)

- **Proprietà di idempotenza e assorbimento:**
  x AND x = x   x OR x = x
  x AND (x OR y) = x x OR (x AND y) = x

L’algebra di Boole - alcune proprietà (2)

- **Proprietà distributiva**

  x AND (y OR z) = (x AND y) OR (x AND z)

  x OR (y AND z) = (x OR y) AND (x OR z)

- **Proprietà di convoluzione**

  NOT (NOT x) = x

- **Proprietà del minimo e del massimo:**

  x AND 1 = x x AND 0 = 0

  x OR 0 = x x OR 1 = 1

- **Leggi di De Morgan:**

  NOT (x AND y) = (NOT x) OR (NOT y)

  NOT (x OR y) = (NOT x) AND (NOT y)

Funzioni booleane

$y = f(x_1, x_2, \ldots, x_n)$ è una funzione booleana se ad ogni ennupla di valori booleani $x_1,\ldots,x_n$ associa un valore booleano y

Due esempi:

x_1	x_2	$f(x_1,x_2)$
0	0	0
0	1	1
1	0	1
1	1	0

Questa funzione è detta OR esclusivo, o XOR

x_1	x_2	$f(x_1,x_2)$
0	0	1
0	1	0
1	0	0
1	1	1

Questa funzione è detta equivalenza, o EQU

Anche AND, OR e NOT sono funzioni booleane. Esse vengono dette *funzioni fondamentali* dell'algebra

Insiemi funzionalmente completi

Si può dimostrare che qualsiasi funzione booleana può essere calcolata applicando le funzioni AND, OR, e NOT.
Ad esempio:

x XOR y = (x AND NOT y) OR (y AND NOT x)

Per questo, l'insieme {AND, OR, NOT} si dice *funzionalmente completo*.

Esistono altri insiemi funzionalmente completi. Si noti che grazie alle leggi di De Morgan si può costruire la AND da {OR, NOT}, oppure la OR da {AND, NOT}. Quindi anche {AND, NOT} e {OR, NOT} sono insiemi funzionalmente completi.

Reti logiche

I valori booleani possono essere rappresentati da grandezze elettriche. Ad esempio:

0 <=> tensione di 0 Volt

1 <=> tensione di +5 Volt

In tal caso le funzioni booleane possono essere realizzate mediante circuiti elettronici detti *reti logiche*.

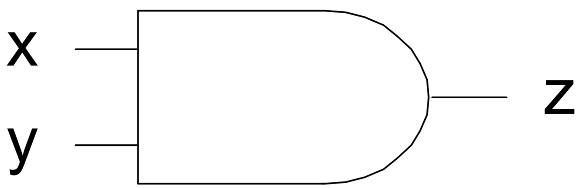
Nelle reti logiche *unilaterali*, leuscite della rete corrispondono a valori di grandezze elettriche misurate in opportuni punti del circuito; il flusso dell’elaborazione procede fisicamente in un’unica direzione, dai segnali di ingresso verso i segnali di uscita.

Nelle reti logiche *bilaterali*, invece, l’uscita della rete è determinata dalla presenza o dall’assenza di “contatto” tra due punti della rete.

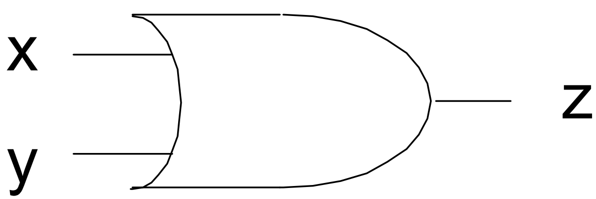
Porte logiche (*gates*)

Circuiti logici elementari che realizzano le operazioni fondamentali. Le reti logiche si costruiscono connettendo più porte logiche.

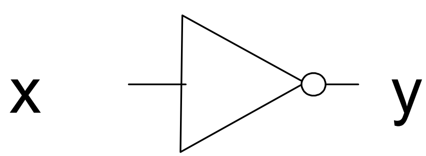
Simboli delle principali porte logiche:



z = x AND y



z = x OR y



y = NOT x

(se connesso a un’altra porta, il NOT si indica talora con un semplice pallino)

Operatori logici generalizzati (1)

Dato un vettore di variabili booleane $X = (x_1, x_2, \ldots, x_n)$, e una variabile booleana α, indicheremo con la notazione:

$Y = \alpha\ OP\ X$ (dove *OP* è un operatore booleano)

l'operazione che produce il vettore booleano Y così definito:

$Y = (y_1, y_2, \ldots, y_n)$ con

$y_1 = \alpha\ OP\ x_1$

. . . . . .

$y_n = \alpha\ OP\ x_n$

Esempio:

α AND X ha come risultato il vettore formato da:

(α AND x_1, α AND x_2, …, α AND x_n)

Operatori logici generalizzati (2)

Dati due vettori di variabili booleane $X = (x_1, x_2, \ldots, x_n)$ e $Y = (y_1, y_2, \ldots, y_n)$ indicheremo con la notazione:

$Z = X\ OP\ Y$ (dove *OP* è un operatore booleano)

l'operazione che produce il vettore booleano Z così definito:

$Z = (z_1, z_2, \ldots, z_n)$ con

$z_1 = x_1\ OP\ y_1$

. . . .

$z_n = x_n\ OP\ y_n$

Esempio:

X OR Y ha come risultato il vettore formato da:

$(x_1 \text{ OR } y_1, x_2 \text{ OR } y_2, \ldots, x_n \text{ OR } y_n)$

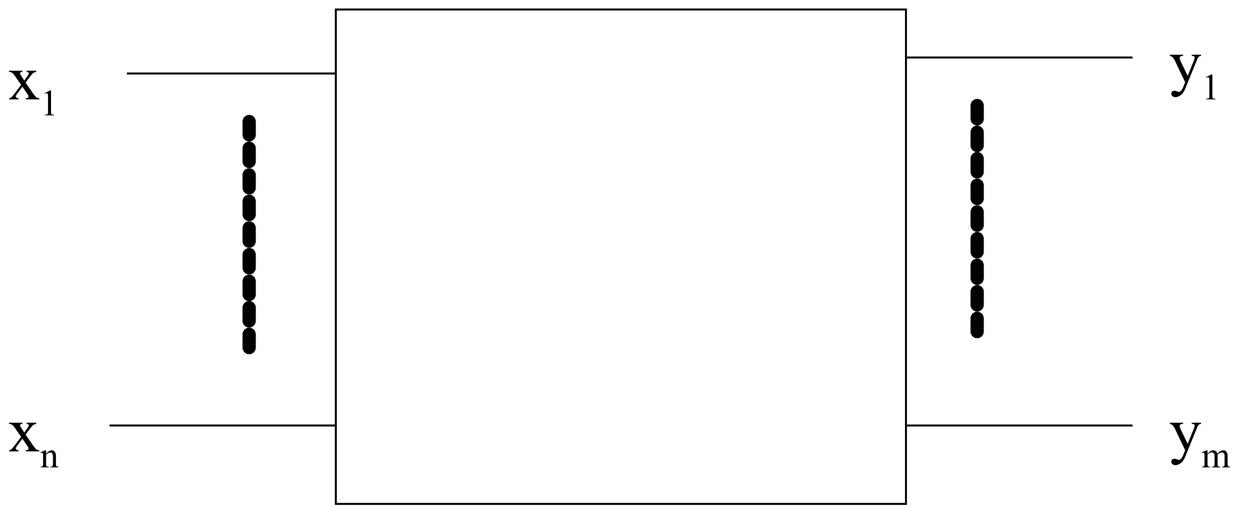
Macchine combinatorie (1)

Reti logiche con *n* ingressi $x_1, x_2, \ldots, x_n$ e *m* uscite $y_1, y_2, \ldots, y_m$ che realizzano la corrispondenza:

$$y_1 = f_1(x_1, x_2, \ldots, x_n)$$

$$\cdots\cdots$$

$$y_m = f_m(x_1, x_2, \ldots, x_n)$$



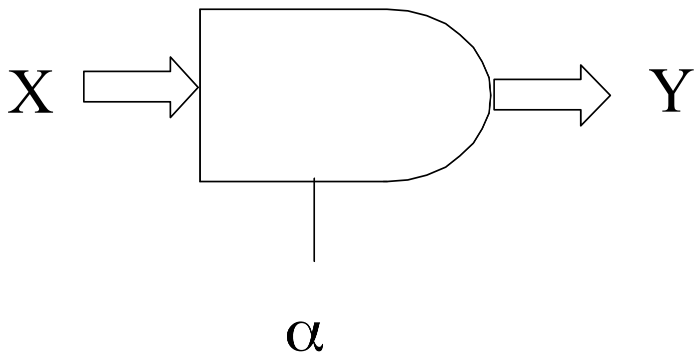
Macchine combinatorie (2)

In una macchina combinatoria i valori delle uscite dipendono esclusivamente dai valori degli ingressi.

In una macchina combinatoria ideale tale dipendenza è istantanea; in una macchina reale c'è sempre un ritardo tra l'istante in cui c'è una variazione in uno degli ingressi e l'istante in cui l'effetto di questa variazione si manifesta sulle uscite.

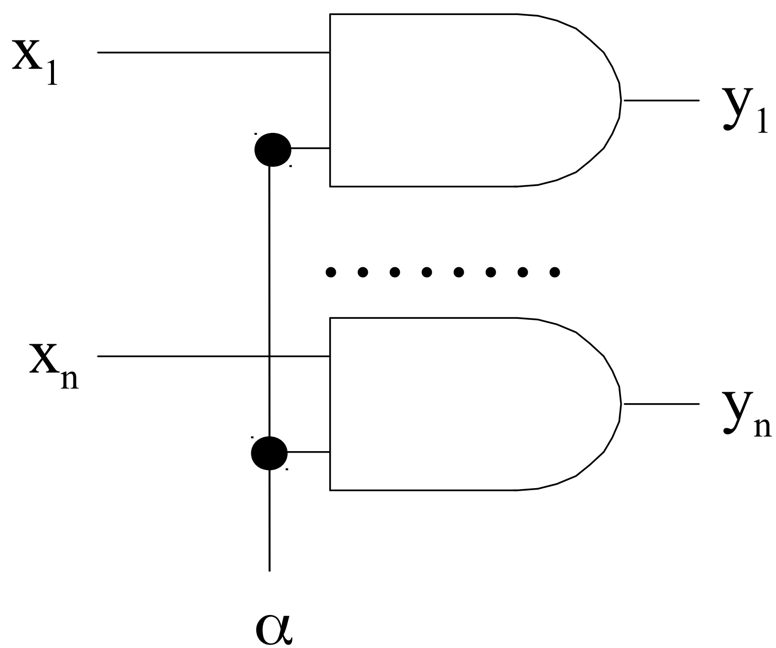
Porte logiche generalizzate (1)

Rappresentazione simbolica:



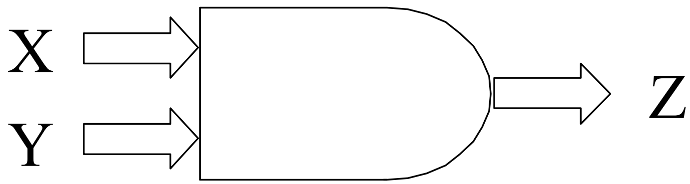
$Y = \alpha$ AND X

Circuito equivalente:



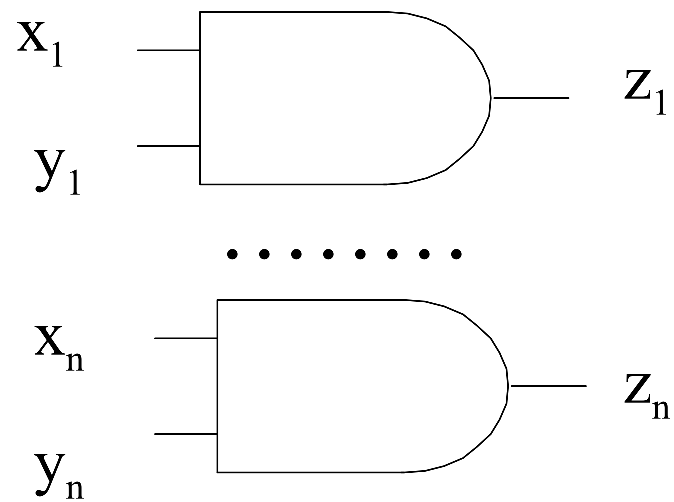
Porte logiche generalizzate (2)

Rappresentazione simbolica:



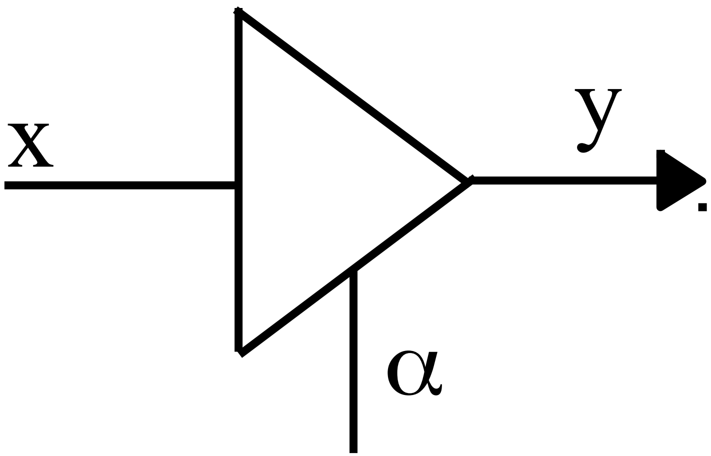
Z = X AND Y

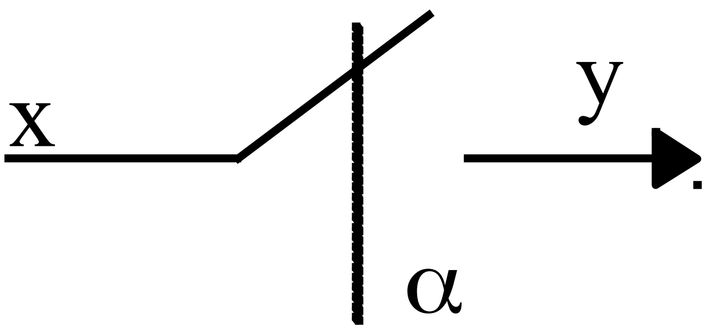
Circuito equivalente:



AND tristate

- È la tecnica più diffusa per realizzare il collegamento di più registri "sorgenti" verso un bus comune





α	x	y
0	1	z
1	1	1
0	0	z
1	0	0
0	z	z
1	z	z

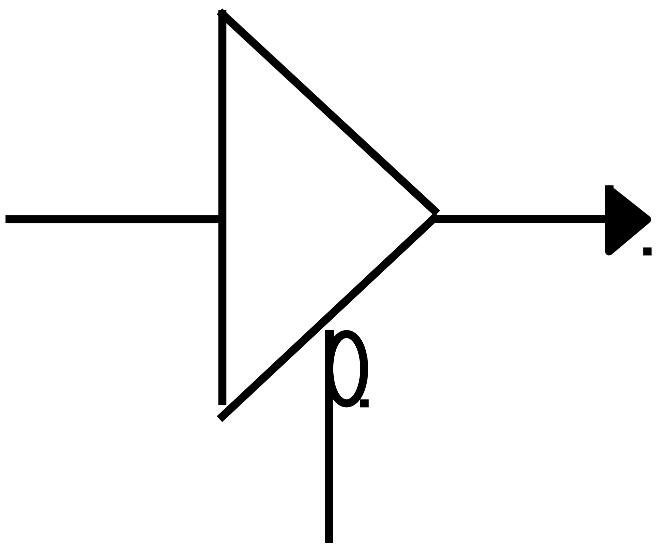
Codifica in Binario

- Dato un insieme di N elementi il numero minimo m di bit necessario alla codifica è dato da

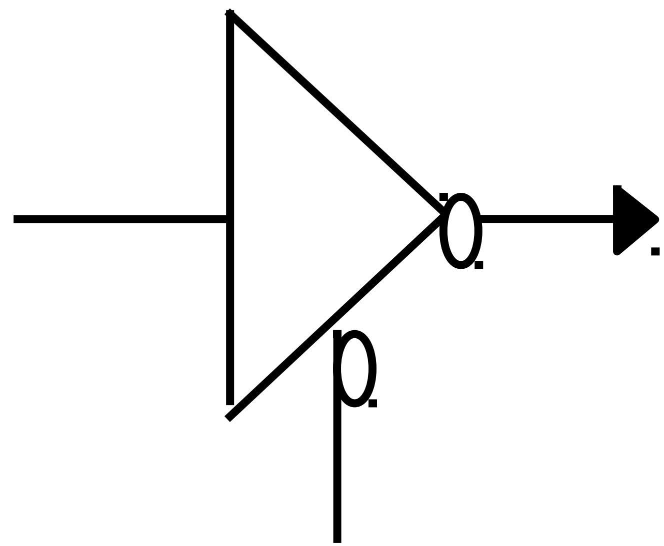
$$m \geq \{[\log_2 N]\}$$

AND tristate - Esempi

- AND tristate con abilitazione 0-attiva

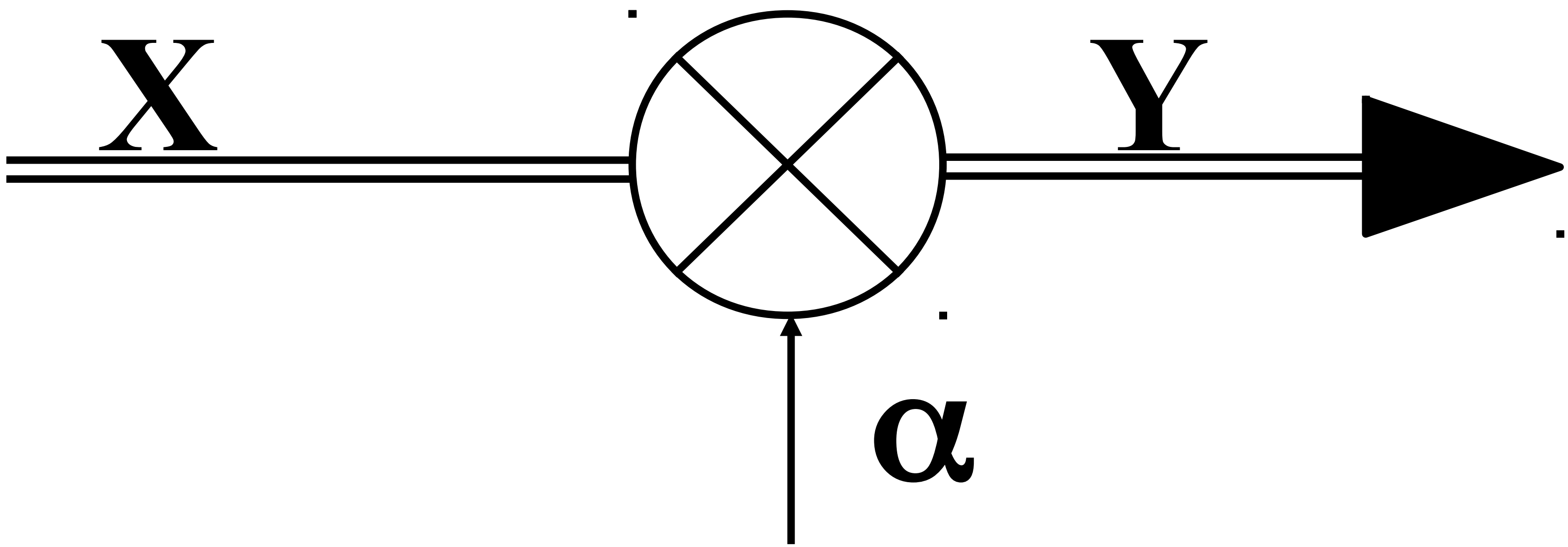


- AND tristate invertente



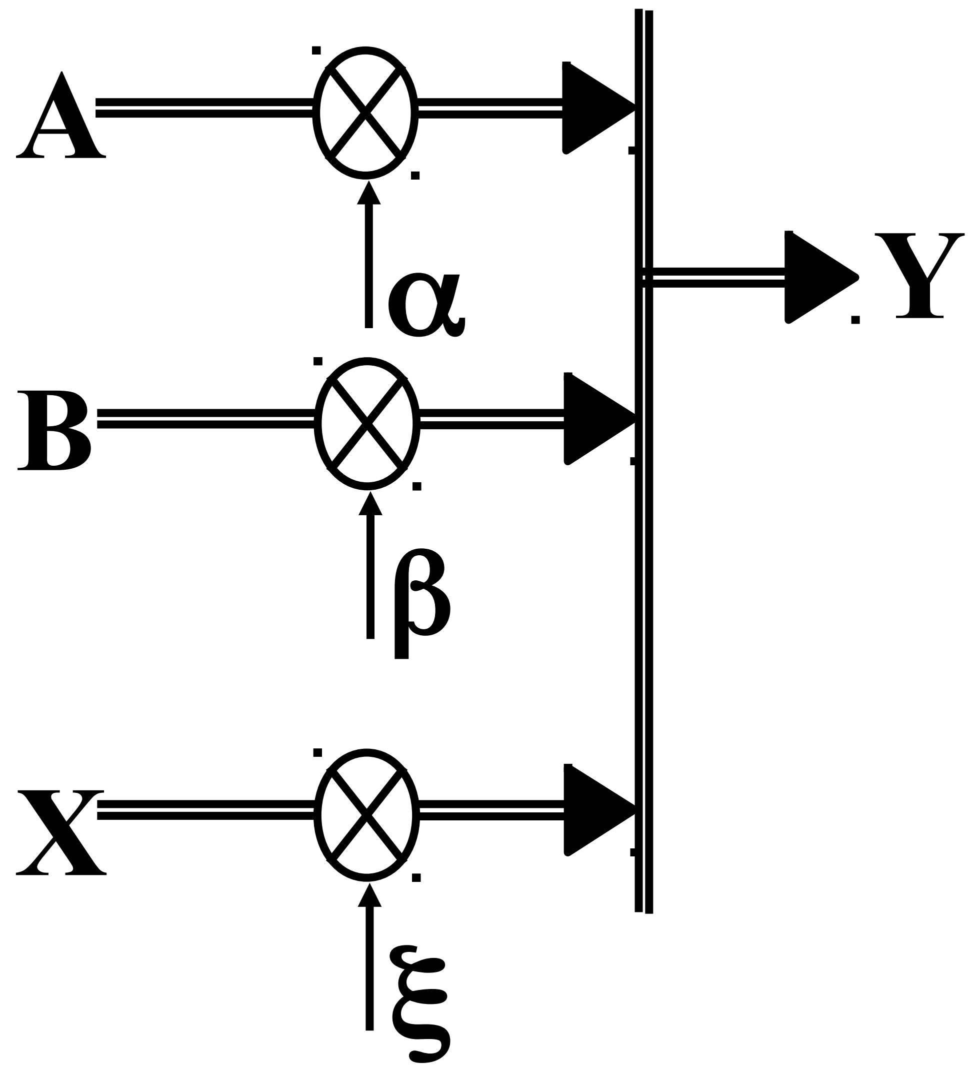
Abilitazione di un bus

$$\mathbf{Y} = \alpha\mathbf{X}$$



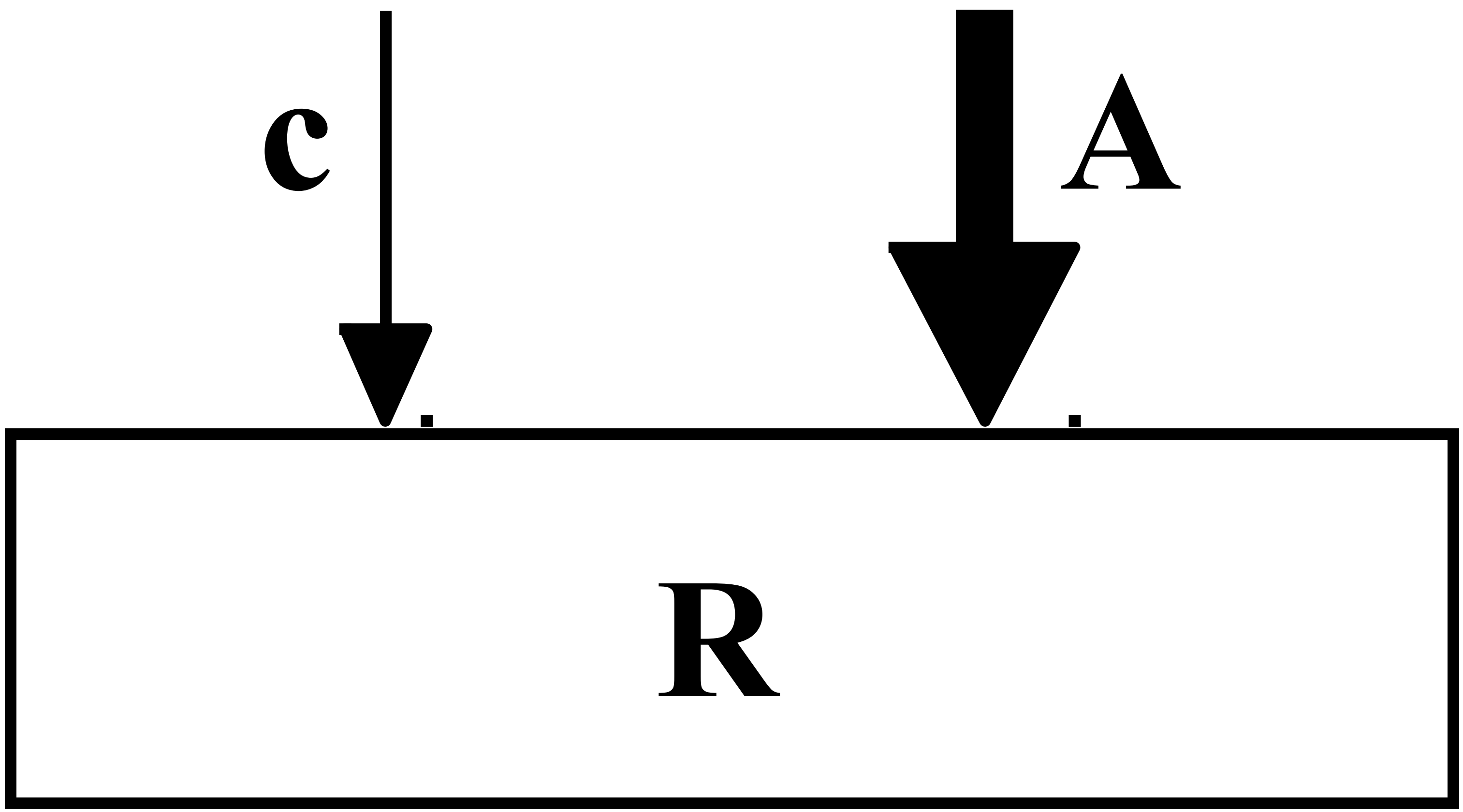
OR di bus

- Connessione delle sorgenti, ciascuna sostenuta da un buffer tristate
- Realizza il collegamento di più registri in uscita verso un bus



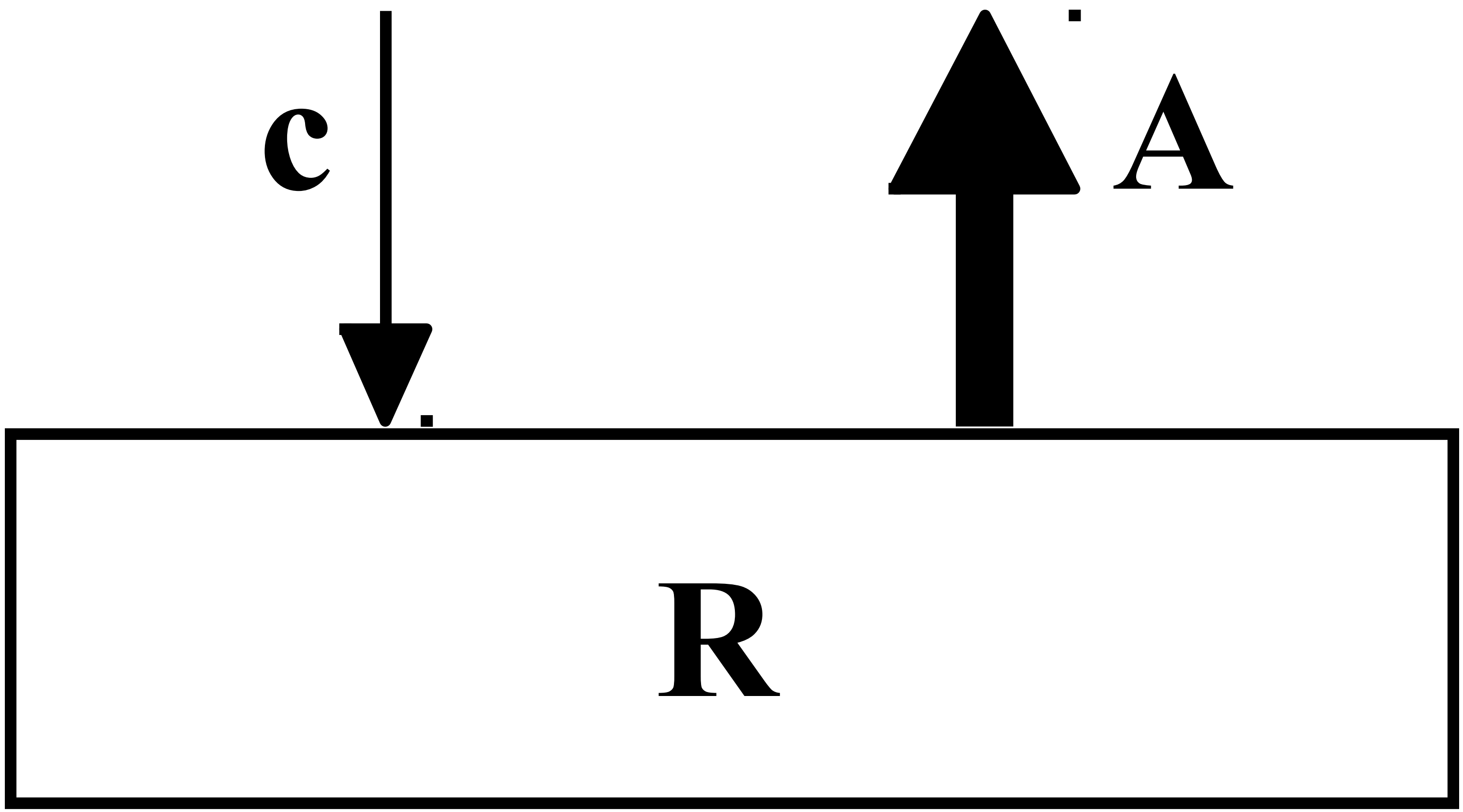
Trasferimento dati su bus unico -1/2

- Trasferimento da bus a registro (caricamento di un registro)

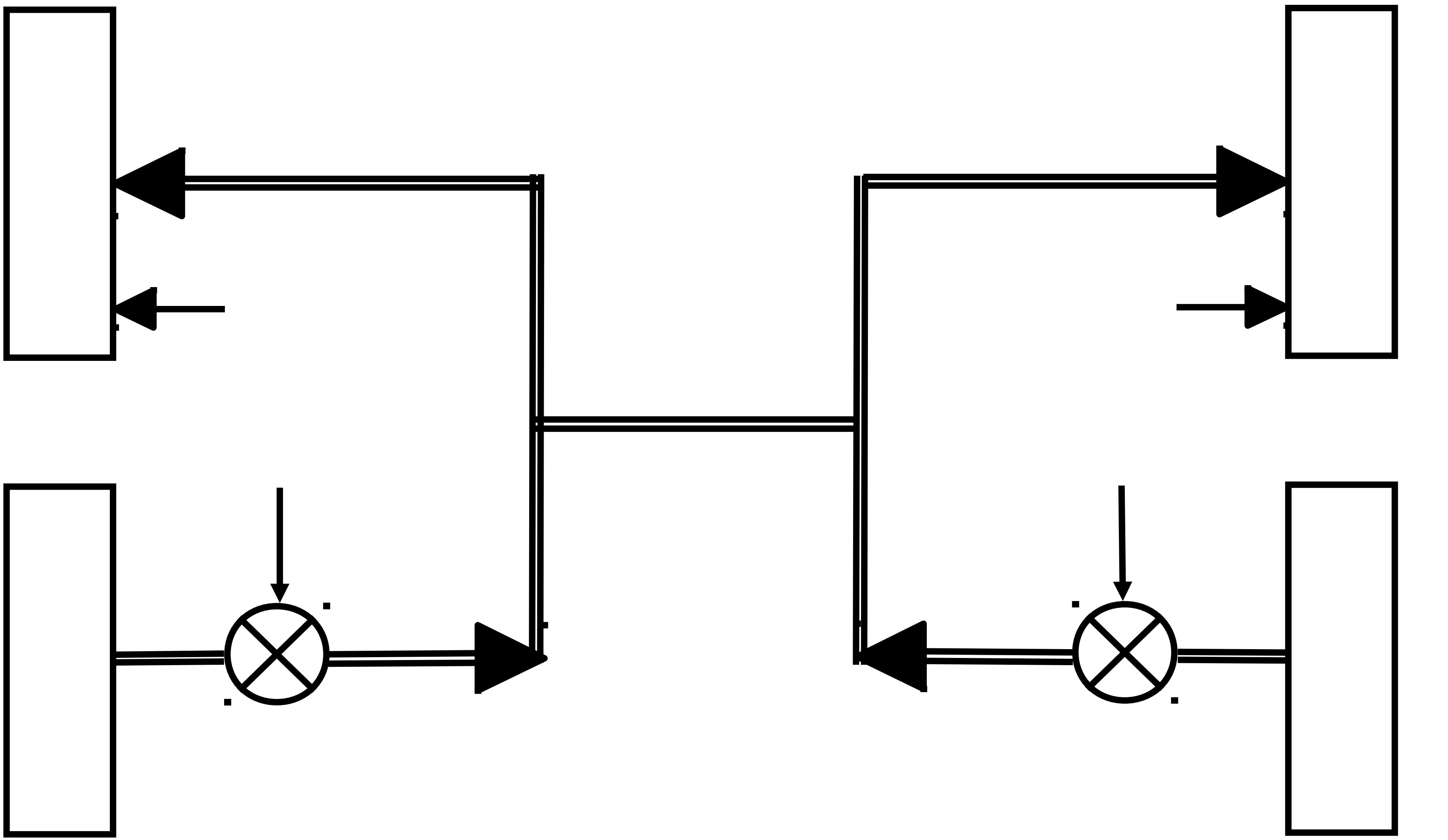


Trasferimento dati su bus unico -2/2

- Trasferimento da registro a bus (caricamento da un registro)



Bus bidirezionali



Rappresentazione di Codici mediante tabelle

$T=(x_1,x_2,\ldots,x_n)$ alfabeto origine

$E=(a_1,a_2,\ldots,a_k)$ alfabeto destinazione

Es. Codice BCD

Dato

	0	1	2	3
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

Parola Codice

Rappresentazione Decodificata

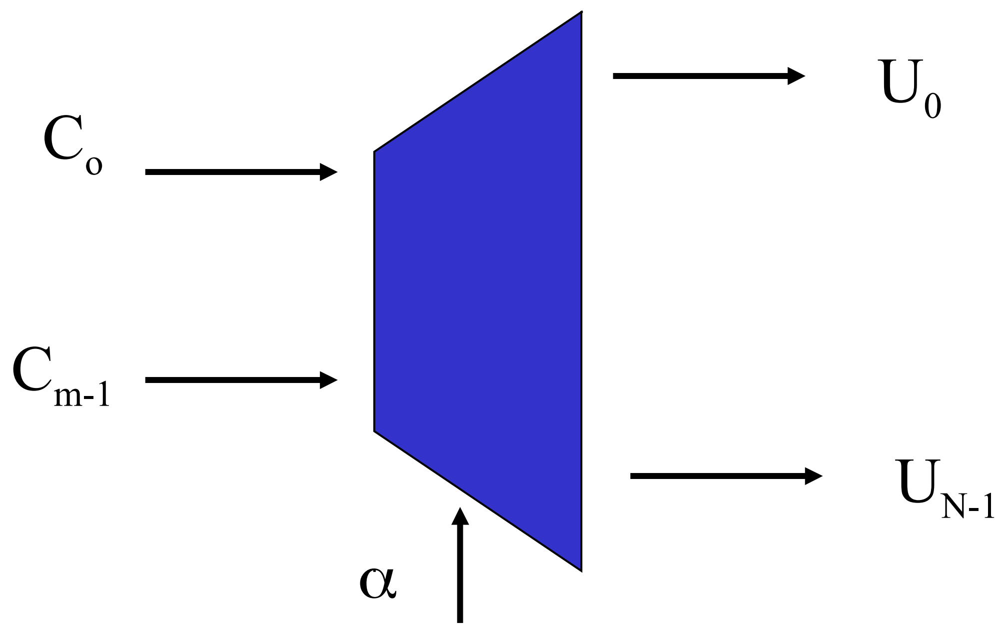
- Laddove il numero m di bit utilizzati per codificare un insieme di cardinalità N è pari ad N (m=N), ed ad ogni parola codice è associato un solo bit 1 otteniamo una rappresentazione *decodificata*

Es.

Dato	Codice BDC	Codice Decodificato
0	0000	100000000
1	0001	010000000
2	0010	001000000
3	0011	000100000
…	…	….
…		
9	1001	000000001

Decodificatore (decoder)

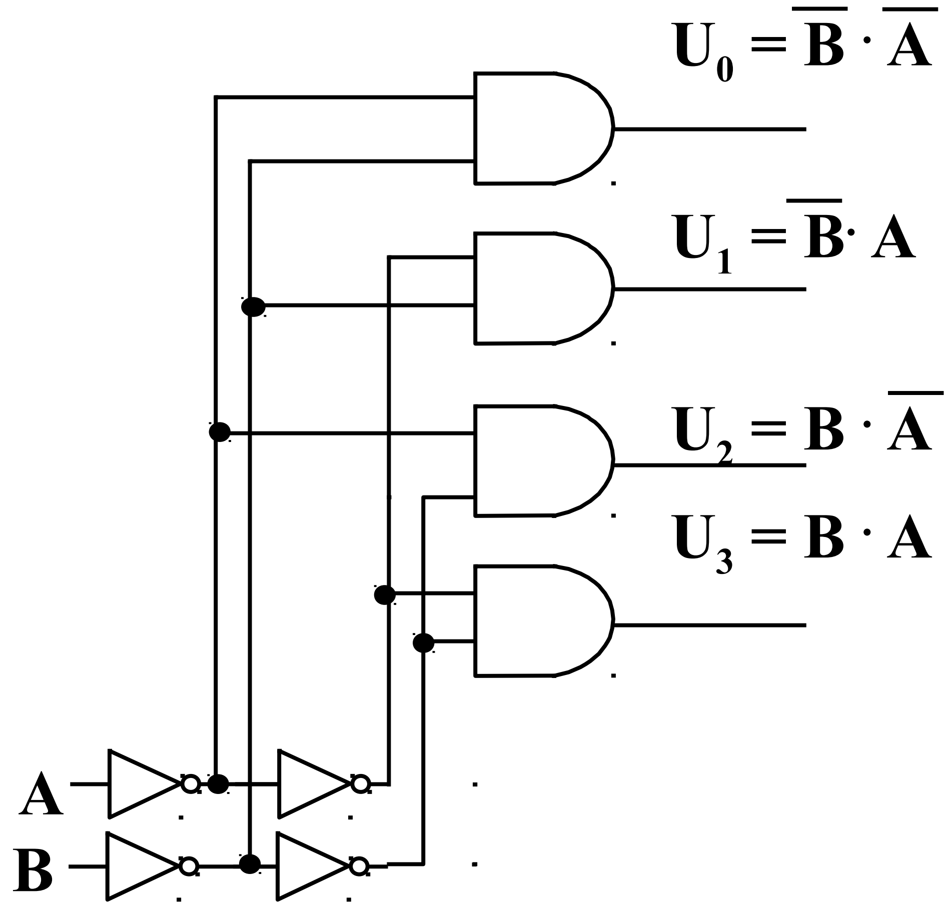
- Un decodificatore è una macchina che riceve in ingresso una parola codice (C) e presenta in uscita la sua rappresentazione decodificata (linee U_0, U_{N-1})



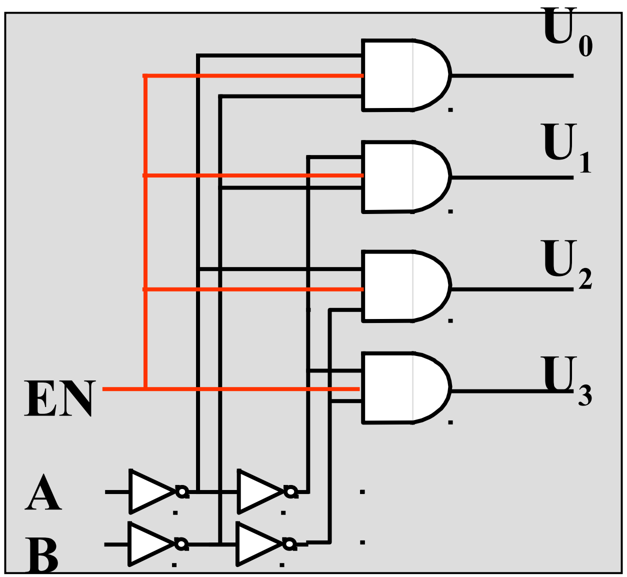
Sintesi della trascodifica da binario a 1 su N

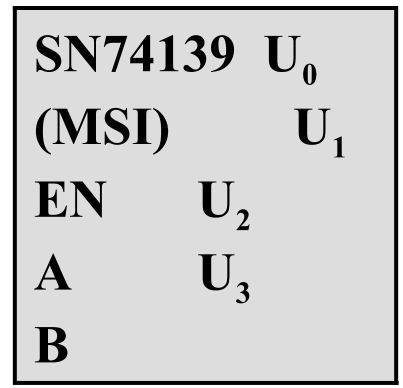
Esempio: Trascodifica 2:4

B	A	U_0	U_1	U_2	U_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1



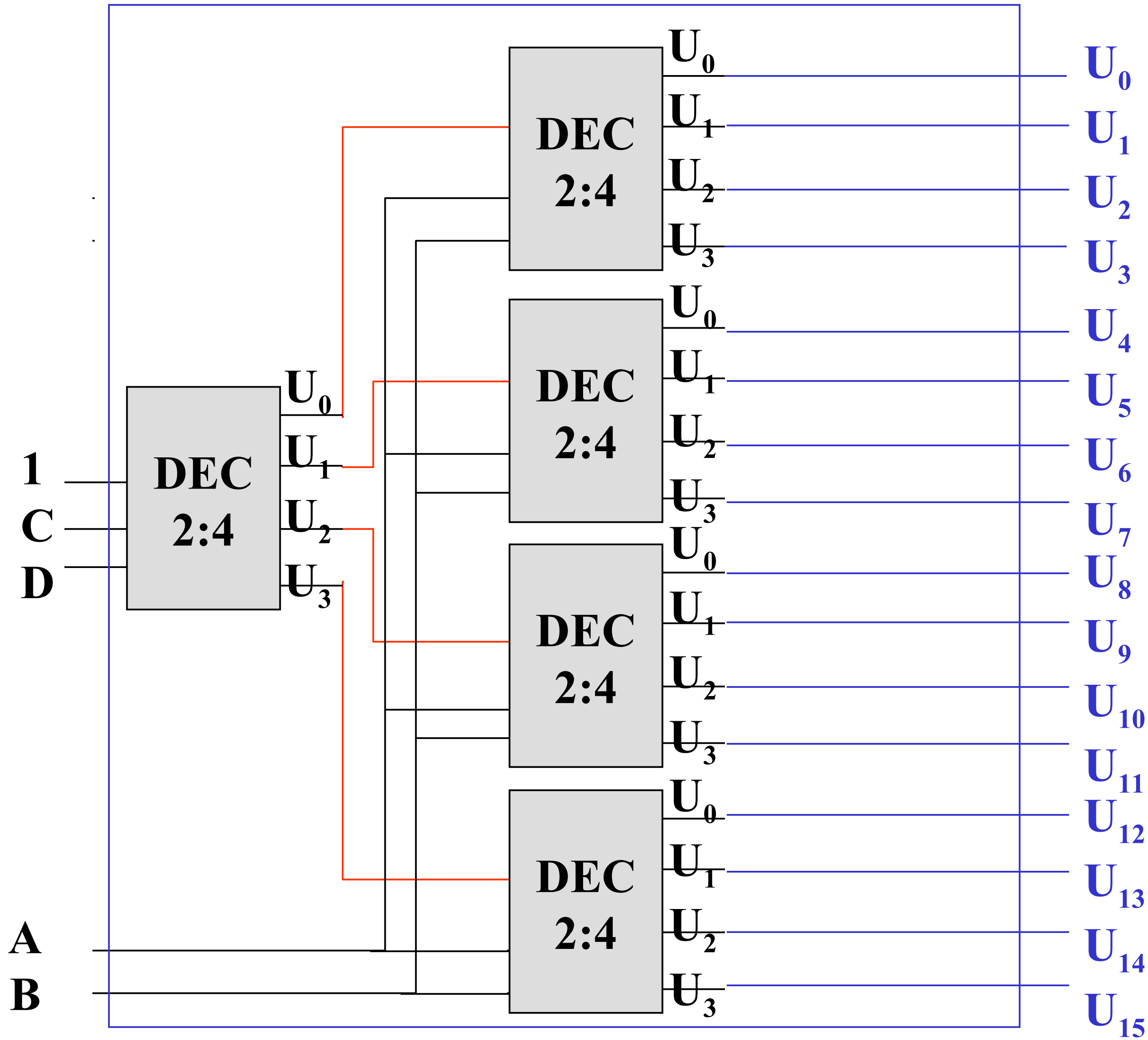
Il circuito integrato DECODER





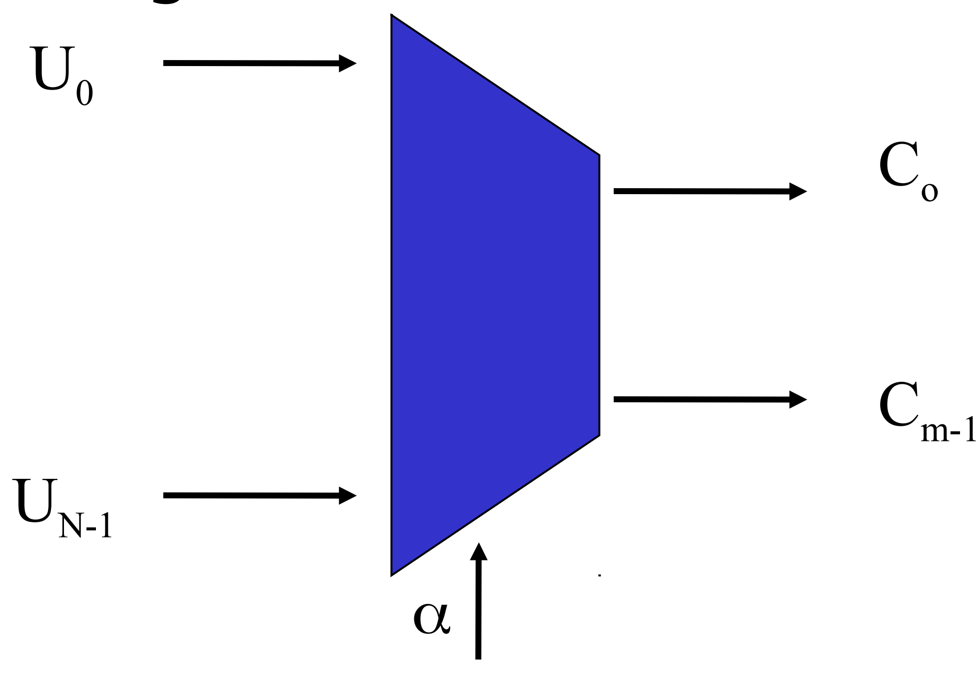
Quando EN=1 (segnale di abilitazione), vale 1 l’uscita il cui pedice, in decimale, corrisponde al numero binario in ingresso (A bit di minor peso**)**

Composizione modulare di Decoder 4:16



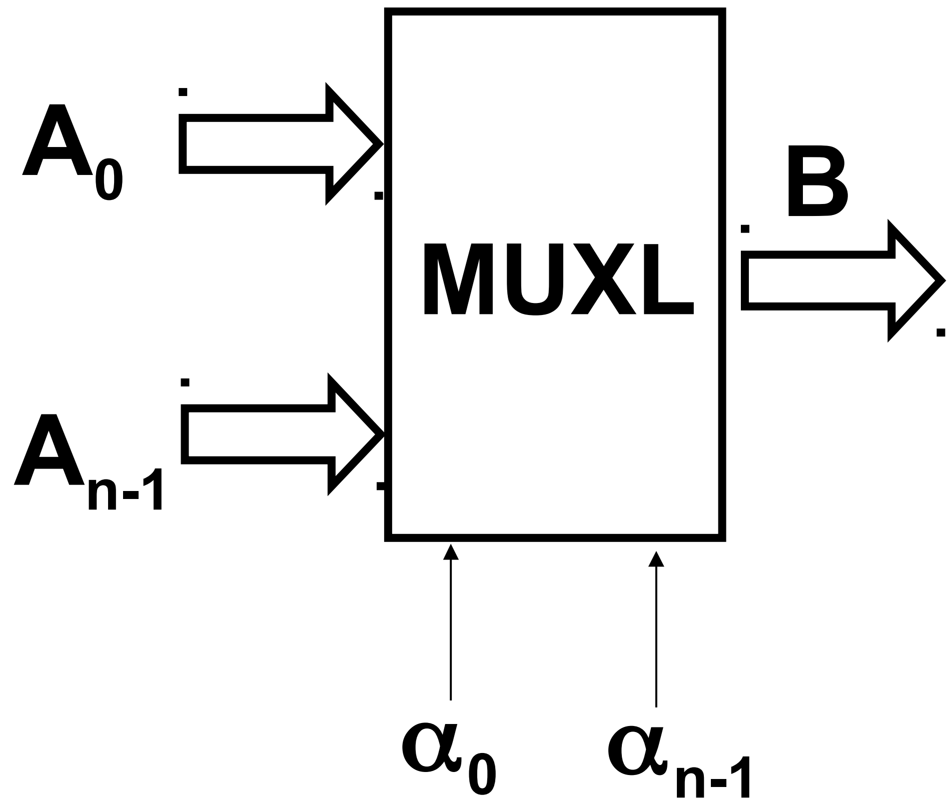
Codificatore (Encoder)

- Un codificatore è una macchina che riceve in ingresso una rappresentazione decodificata (linee U_0, U_{N-1}) e fornisce in uscita la parola codice associata a U_i se U_i=1 (più in generale se è attivo)



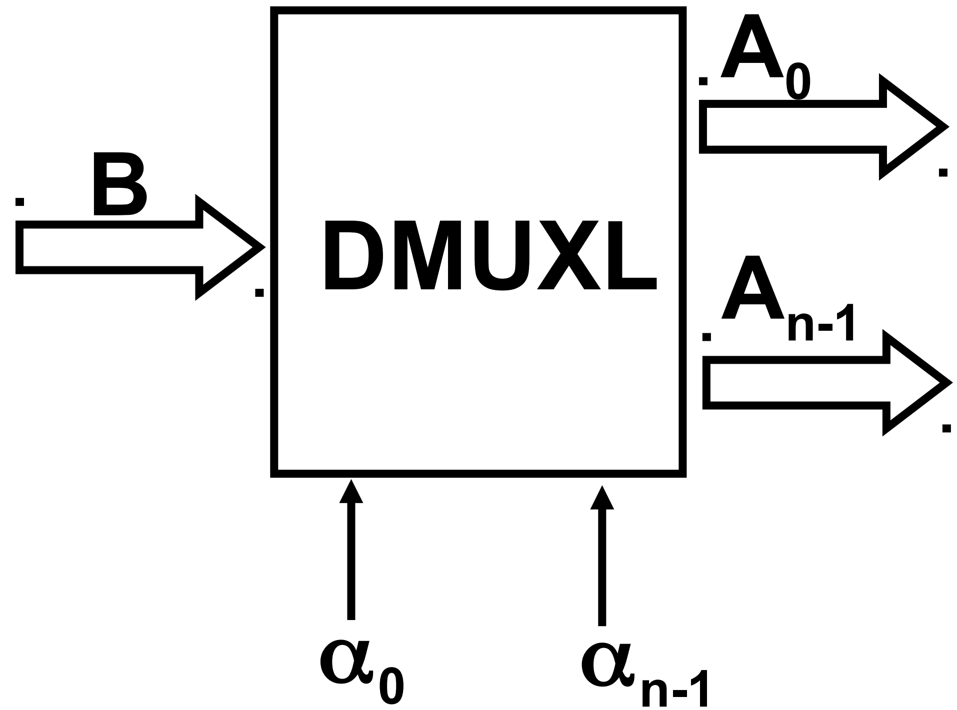
Multiplexer lineare

- Macchina con:
  - n ingressi-dati ($A_0,\ldots,A_{n-1}$)
  - n segnali binari di selezione ($\alpha_0,\ldots, \alpha_{n-1}$), dei quali al più uno è attivo
  - una uscita-dati B, che assume il valore A_i se è attivo α_i, è neutro se nessuna delle selezioni è attiva



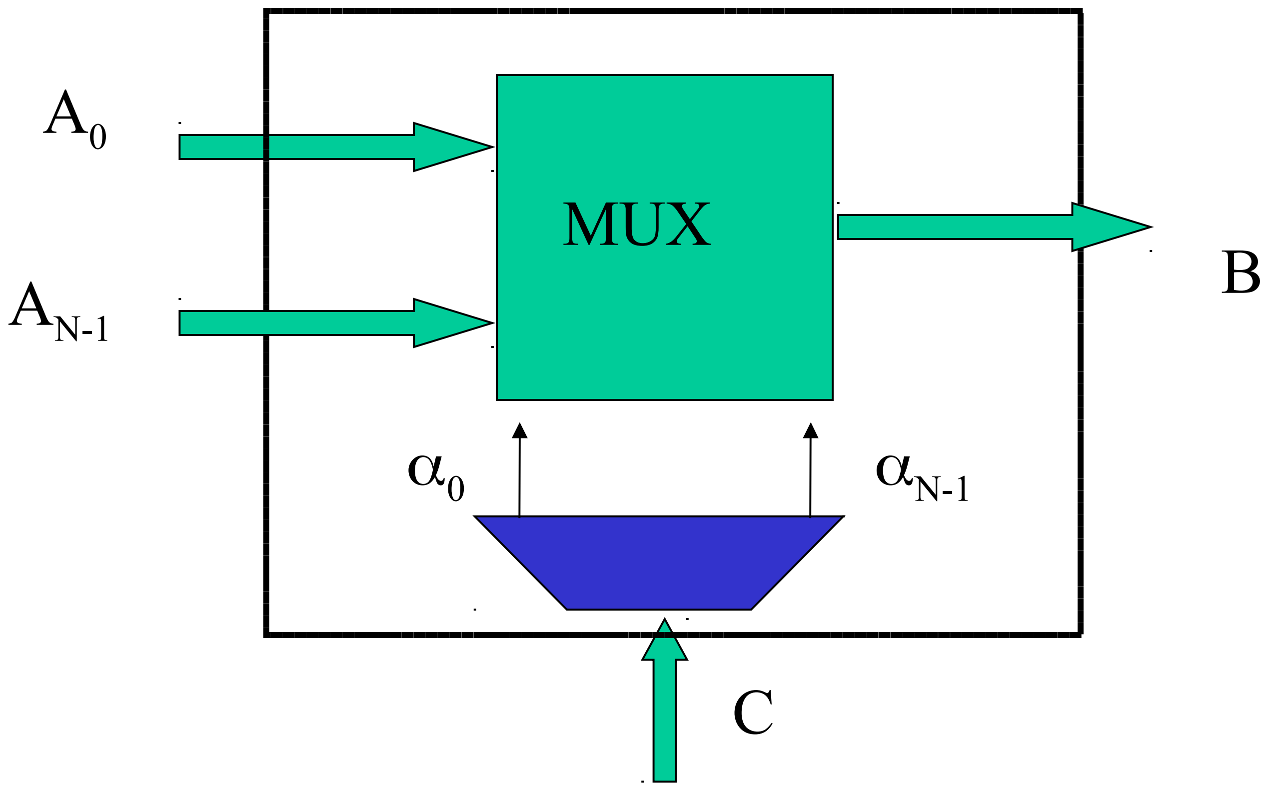
Demultiplexer lineare

- Macchina con:
  - 1 ingresso-dati B
  - n segnali binari di selezione ($\alpha_0,\ldots,\alpha_{n-1}$), dei quali al più uno è attivo
  - n uscite-dati ($A_0,\ldots,A_{n-1}$), con A_i=B se è attivo α_i, è neutro se nessuna delle selezioni è attiva



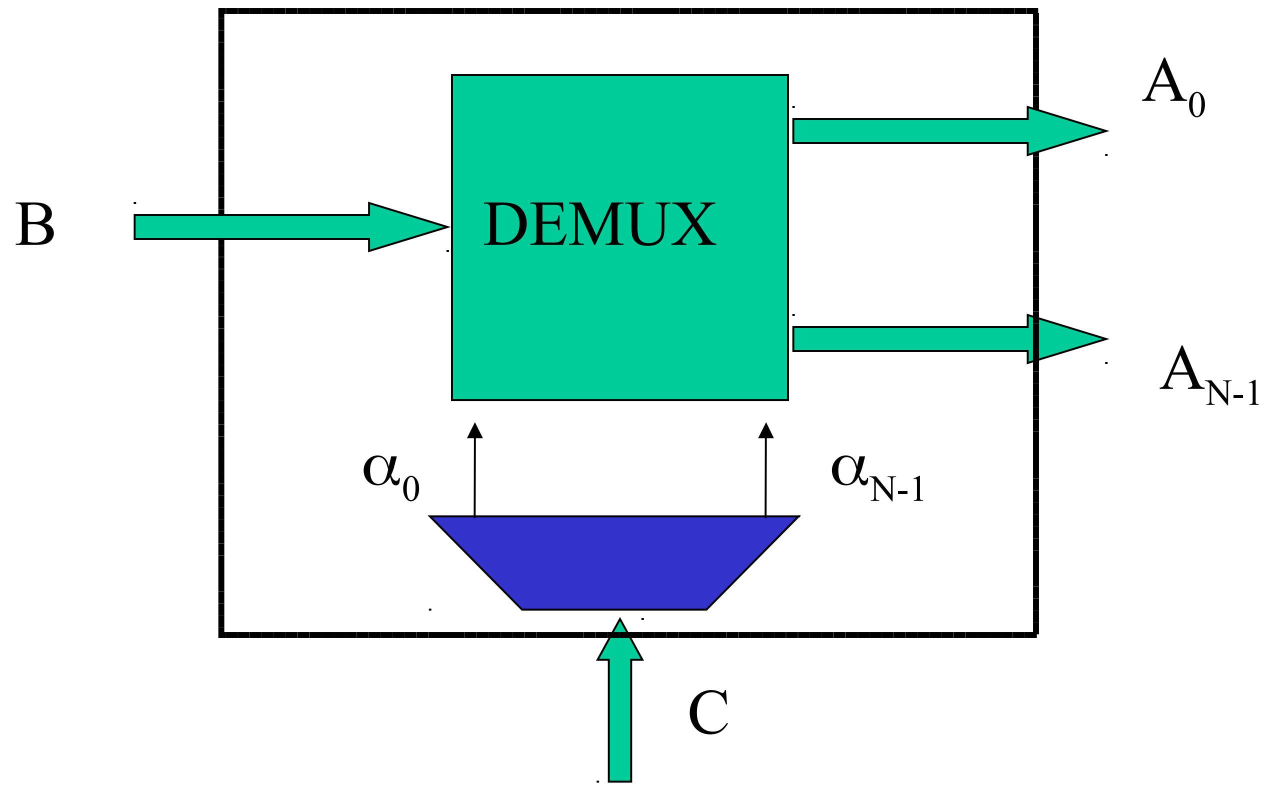
Multiplexer Indirizzabile

- E’ un multiplexer i cui segnali di abilitazione sono collegati con le uscite di un decodificatore

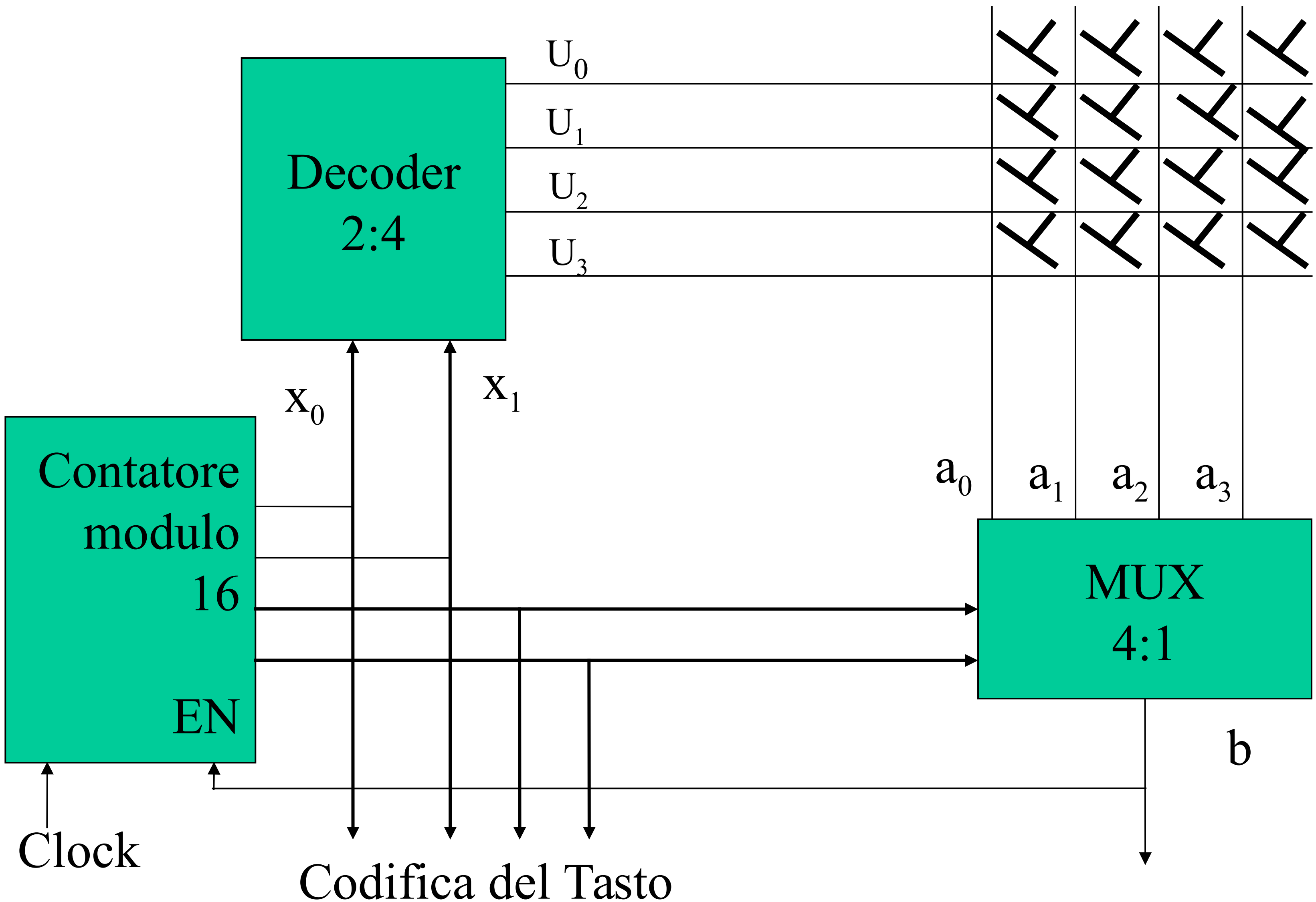


Demultiplexer Indirizzabile

- E’ un Demultiplexer i cui segnali di abilitazione sono collegati con le uscite di un decodificatore



Implementazione di una tastiera



- Le 4 uscite del contatore sono usate per scorrere la matrice 4x4 dei tasti:
  - i primi 2 bit costituiscono l’ingresso del decodificatore da cui uscirà una combinazione di 4 bit dove un solo bit è posto a 1 (tale bit identificherà la riga della matrice)
  - Gli altri 2 bit sono invece gli ingressi del multiplexer e codificano la colonna da esaminare
- Se il tasto in corrispondenza della coppia (riga, colonna) esaminata è stato premuto, allora il multiplexer restisuisce b=1. Contemporaneamente, la codifica del tasto (che corrisponde allo stato del contatore) è disponibile.
- Il contatore rimane fermo fin quando il tasto viene premuto.