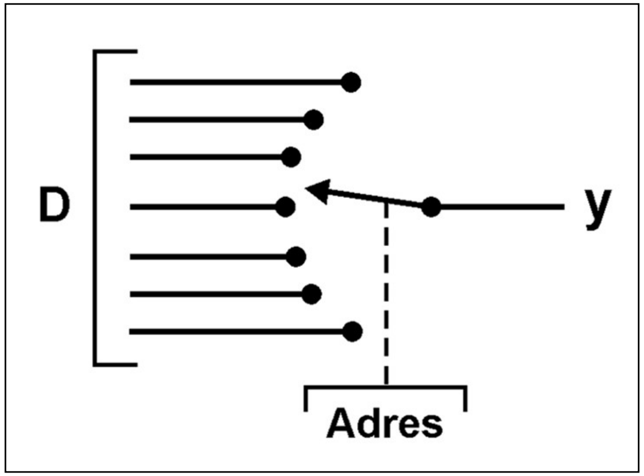
CYFROWE BLOKI KOMBINACYJNE

- ***Multipleksery***
- ***Demultipleksery***
- ***Kodery, dekodery***
- ***Konwertery kodów***
- ***Komparatory***
- ***Bloki arytmetyczne***

Multipleksery (MUX, MX)

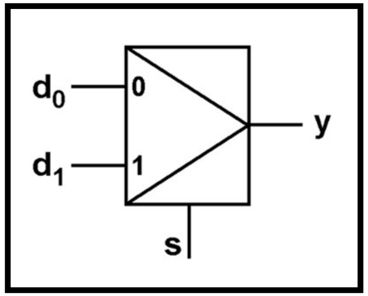
Układy pełniące funkcję selektora (komutatora) umożliwiając wybór i połączenie jednego z wielu wejść do jednego wyjścia.

Wybór wejścia określany jest przez adres.

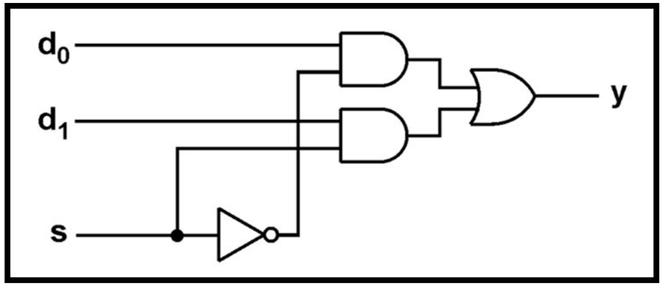


Multipleksery

Multiplekser 2-wejściowy (2-na-1)



s	y
0	d0
1	d1



Tablica prawdy

s	d1	d0	y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Siatka Karnough...

s \ d1 d0	00	01	11	10
0	0	1	1	0
1	0	0	1	1

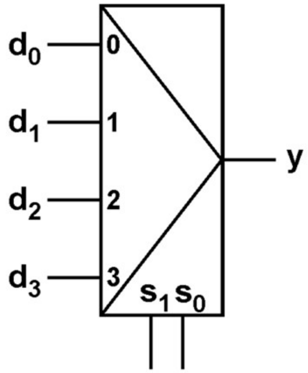
y

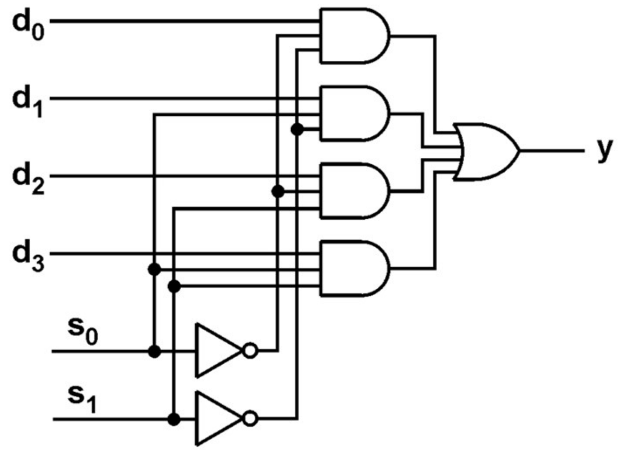
Funkcja logiczna...

$$\boxed{y = d_1 s + d_0 \overline{s}}$$

Multipleksery

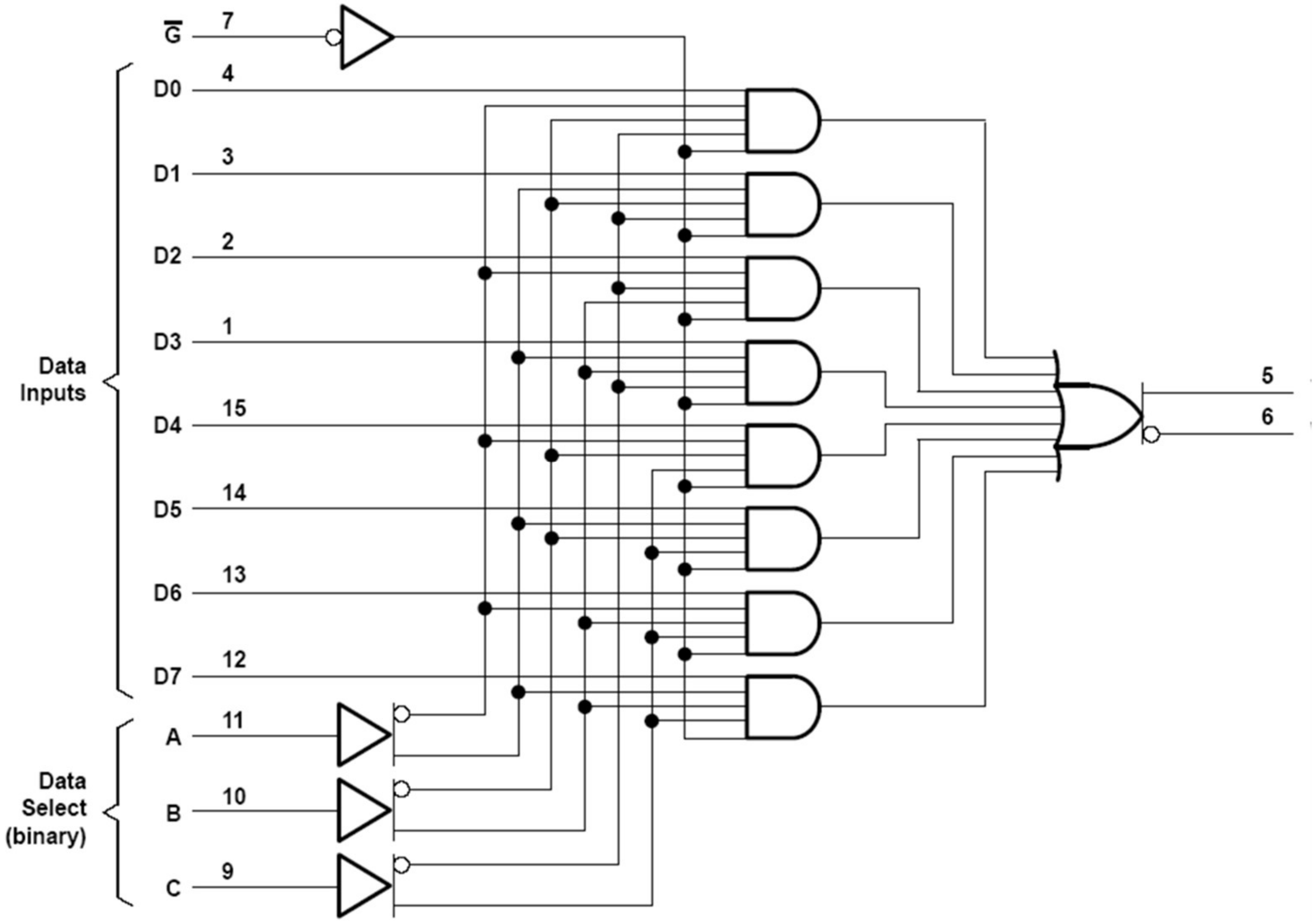
Multiplekser 4-wejściowy (4-na-1)





$$\mathbf{y} = \mathbf{d_3 s_1 s_0} + \mathbf{d_2 s_1 \overline{s}_0} + \mathbf{d_1 \overline{s}_1 s_0} + \mathbf{d_0 \overline{s}_1 \overline{s}_0}$$

Multiplekser scalony 74151

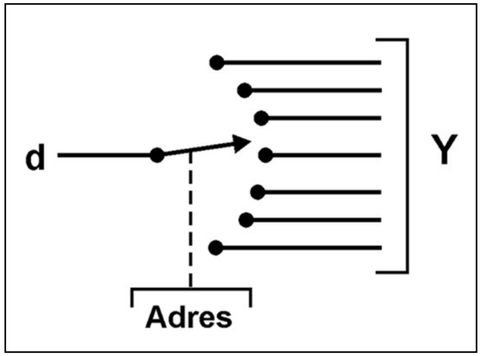


Demultipleksery (DMUX, DX)

Układy pełniące funkcję odwrotną niż multipleksery.

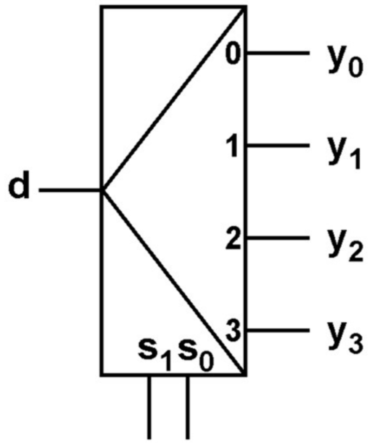
Umożliwiają połączenie jednego sygnału wejściowego do jednego z wielu wyjść.

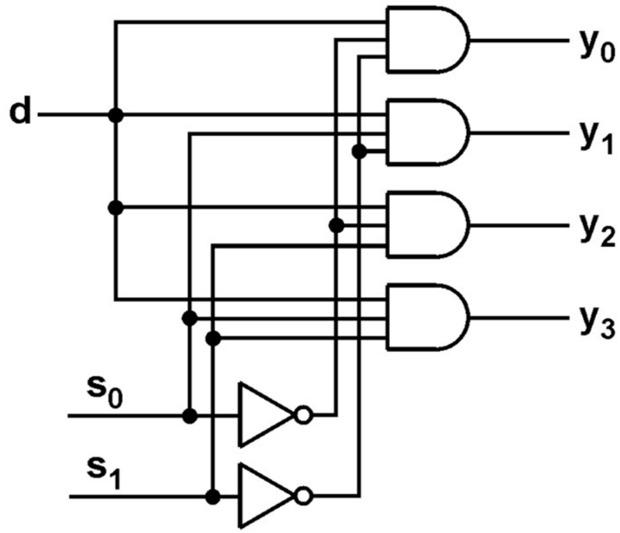
Wybór wyjścia określany jest przez adres.



Demultipleksery

np. **Multiplekser 4-wyjściowy (1-na-4)**

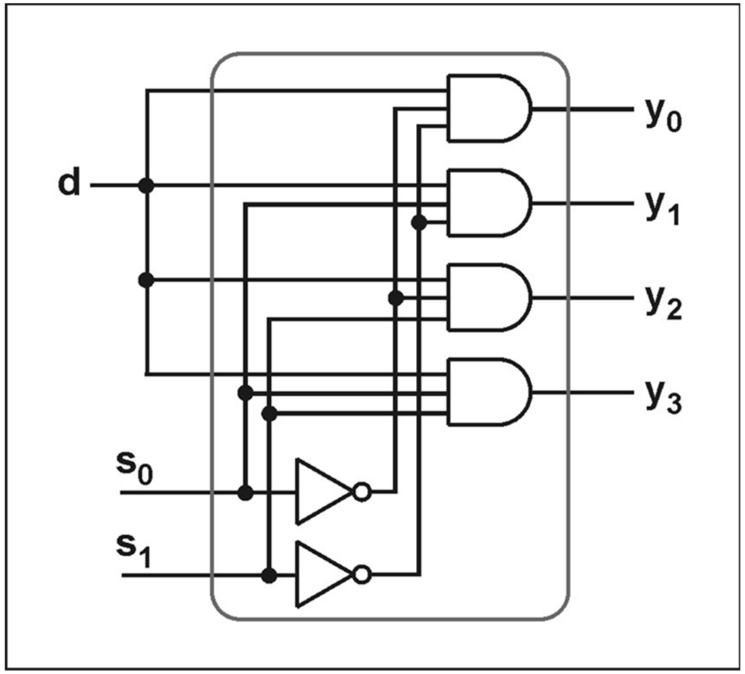




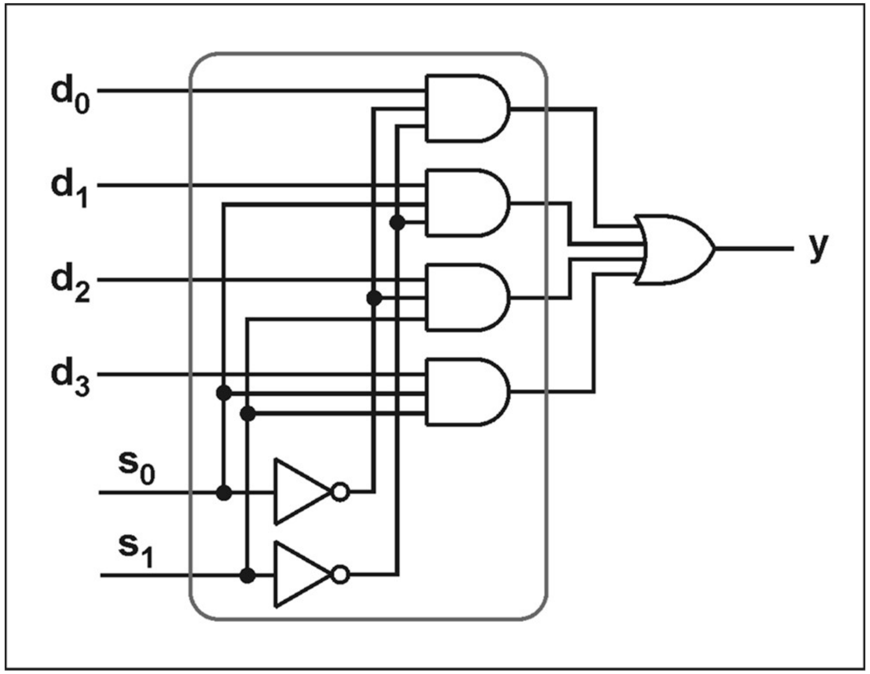
$$\boxed{y_0 = d\overline{s}_1\overline{s}_0} \quad \boxed{y_1 = d\overline{s}_1 s_0}$$

$$\boxed{y_2 = d s_1\overline{s}_0} \quad \boxed{y_3 = d s_1 s_0}$$

Demultiplekser



Multiplekser



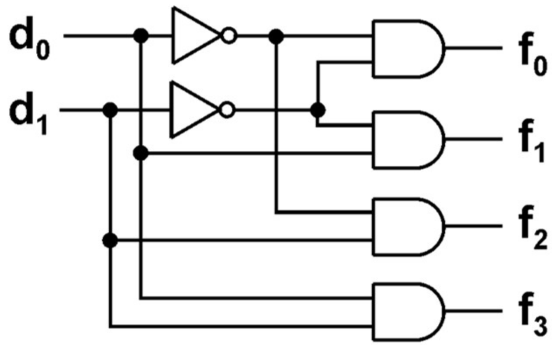
Takie same „układy” wewnątrz obu struktur...

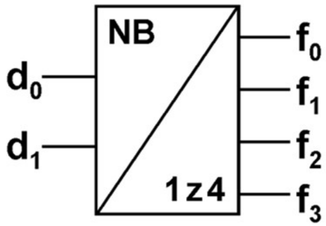
Dekodery

Służą do przetwarzania dowolnego kodu binarnego na kod 1–z–*N*.

np.

NB	1-z-4
00	0001
01	0010
10	0100
11	1000



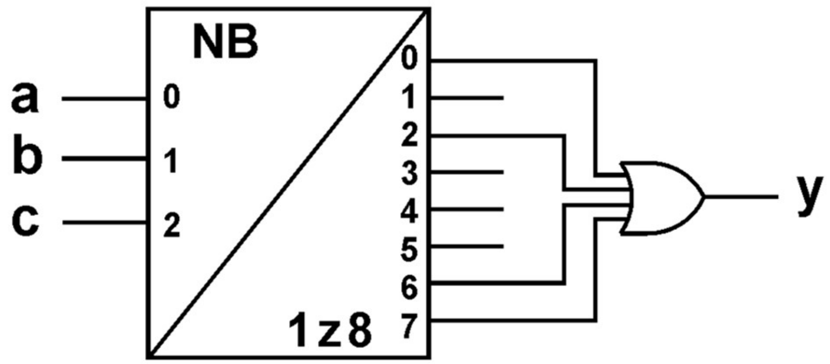


$$f_0 = \overline{d_1}\,\overline{d_0}$$
$$f_1 = \overline{d_1}\,d_0$$
$$f_2 = d_1\,\overline{d_0}$$
$$f_3 = d_1 d_0$$

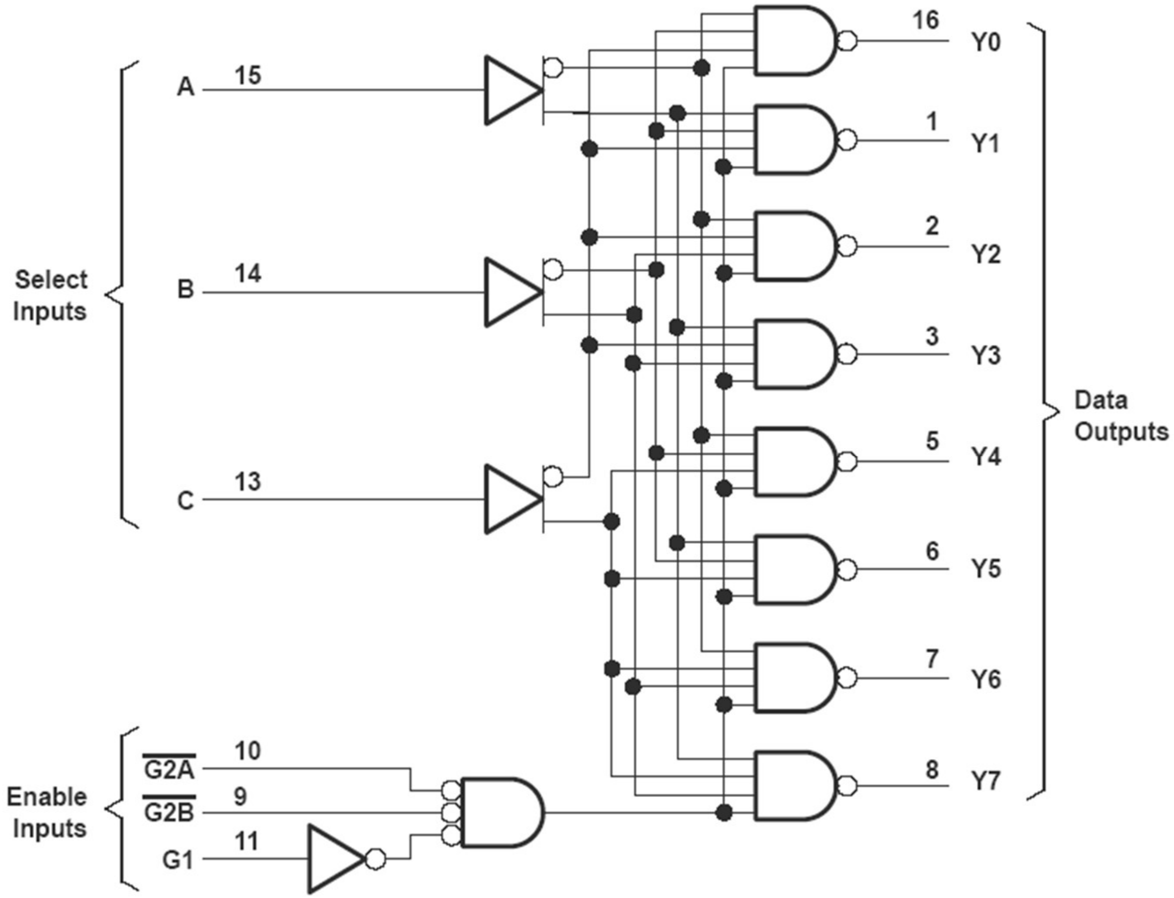
Dekodery – realizacja funkcji logicznej !!!

np. $T = \{ 0, 2, 6, 7 \}_{cba}$

$$y = \bar{c}\bar{b}\bar{a} + \bar{c}b\bar{a} + cb\bar{a} + cba$$



Dekoder scalony 74138

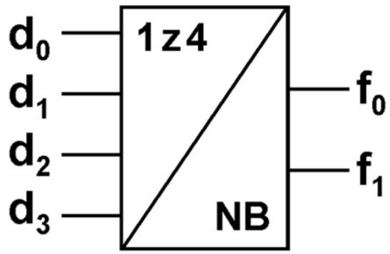


Kodery

Służą do przetwarzania kodu 1–z–*N* na kod binarny.

np.

1-z-4	NB
0001	00
0010	01
0100	10
1000	11

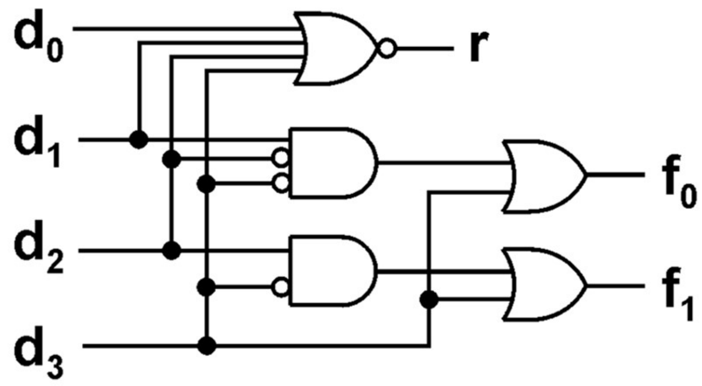


???: 0000, 0011, 1010, 1101, itd…

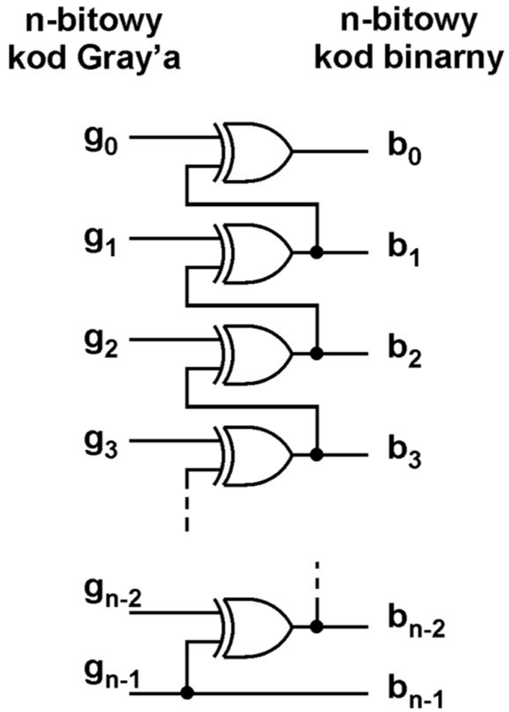
Kodery priorytetowe

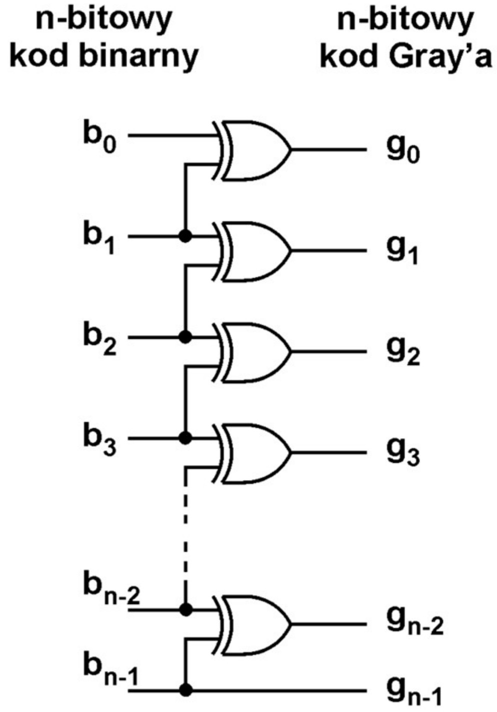
W przypadku obecności na kilku wejściach stanów aktywnych, kodowaniu podlega tylko stan na wejściu o najwyższym priorytecie, zazwyczaj zgodnie z kierunkiem indeksowania...

1-z-4	NB	r
0000	00	1
0001	00	0
001-	01	0
01- -	10	0
1- - -	11	0



Konwertery kodu Gray’a (układ iteracyjny...)





$$b_{n-1} = g_{n-1}$$
$$b_i = g_i \oplus b_{i+1}$$

$$g_{n-1} = b_{n-1}$$
$$g_i = b_i \oplus b_{i+1}$$

Pozostałe dekodery...

Do sterowania cyfrowych wskaźników informacyjnych stosuje się również dekodery kodu BCD-8421 na kod wskaźnika 7-segmentowego, np. 7447

ARYTMETYKA DWÓJKOWA

Podstawowe operacje arytmetyki dwójkowej:

- dodawanie arytmetyczne,

- odejmowanie arytmetyczne,

- mnożenie arytmetyczne,

- dzielenie arytmetyczne.

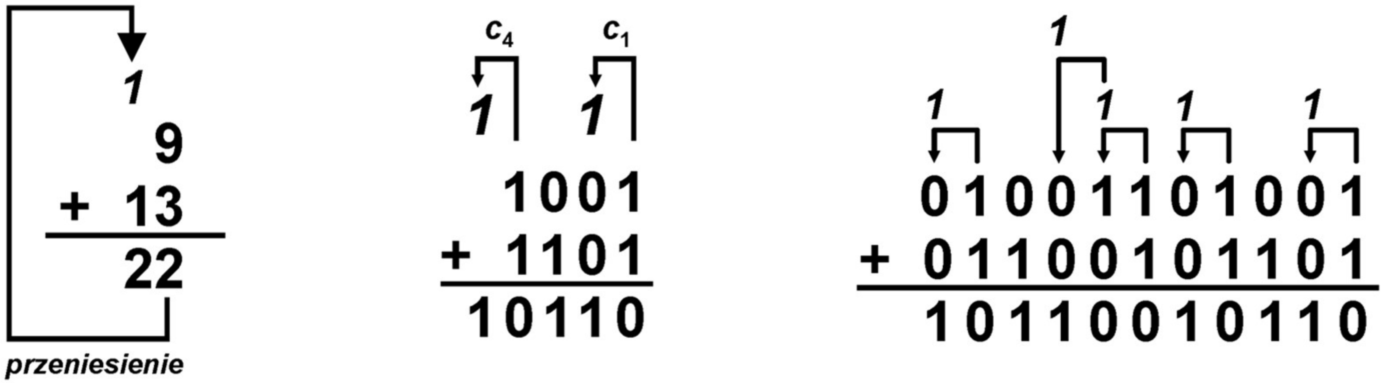
Operacje wykonywane są na liczbach kodowanych dwójkowo.

DODAWANIE

Aby dodać dwie liczby dwójkowe należy dokonać sumowania par bitów na poszczególnych pozycjach, rozpoczynając od najmniej znaczącego bitu (LSB).

Cyfrowy układ dodający to sumator.

Podobnie jak w arytmetyce dziesiętnej, należy przy sumowaniu bitów na każdej *i*-tej pozycji uwzględniać bit przeniesienia c_i z niższej pozycji.



Dodawanie kilku liczb dwójkowych

Liczby dwójkowe dodajemy parami!

Np.: 0101 + 1100 + 0010 + 1010

$$
\begin{array}{r}
0101 \\
+\ 1100 \\
\hline
10001
\end{array}
\qquad
\begin{array}{r}
10001 \\
+\ 00010 \\
\hline
10011
\end{array}
\qquad
\begin{array}{r}
10011 \\
+\ 01010 \\
\hline
11101
\end{array}
$$

TO JEST SUMOWANIE AKUMULACYJNE!

UZUPEŁNIENIA LICZB

Każdą liczbę naturalną można zapisać w odpowiednim kodzie uzupełnieniowym.

Dla każdego kodu liczbowego o podstawie *p* istnieją dwa rodzaje uzupełnień:

– uzupełnienie do *p* – 1, oznaczane U(*p* – 1)

– uzupełnienie do podstawy *p*, oznaczane U(*p*)

Np.:

Dla liczb o podstawie *p* = 10 (czyli dziesiętnych), istnieją uzupełnienia U9 i U10.

Dla liczb o podstawie *p* = 2 (czyli dwójkowych), istnieją uzupełnienia U1 i U2.

Uzupełnienia liczb U(p – 1)

W praktyce uzupełnienie U(p – 1) liczby nieujemnej otrzymuje się poprzez odjęcie każdej cyfry tej liczby od p – 1.

Np.:

U9 (378) = 621 **ponieważ 999 – 378 = 621**

U9 (13,345) = 86,654 **ponieważ 99,999 – 13,345 = 86,654**

W przypadku liczb dwójkowych operacja uzupełnienia do 1 (czyli U1) określana jest *operacją dopełnienia*, a liczba dwójkowa w kodzie U1 nazywana jest *dopełnieniem*. W praktyce jest to negacja wszystkich bitów!

Np.:

U1 (101) = 010

U1 (0) = 1

U1 (11.01101) = 00.10010

Uzupełnienia liczb U*p*

W praktyce uzupełnienie U*p* liczby nieujemnej otrzymuje się poprzez dodanie 1 do jej uzupełnienia U(*p* – 1).

Np.:

U10 (378) = 621 + 1 = 622

U10 (13,345) = 86,654 + 0,001 = 86,655

albo…

U10 (378) = 1000 – 378 = 622

U10 (13,345) = 100 – 13,345 = 86,655

Uzupełnienia liczb U2

W przypadku liczb dwójkowych operacja uzupełnienia do 2 (czyli U2) wykonywana jest przez dodanie arytmetyczne jedynki do uzupełnienia U1 tej liczby.

Np.:

U2 (101) = 010 + 001 = 011

U2 (0) = 0

U2 (11.01101) = 00.10010 + 00.00001 = 00.10011

ZAPIS LICZB DWÓJKOWYCH ZE ZNAKIEM

W systemie dziesiętnym liczby ujemne mają znak minus (–) a dodatnie plus (+).

W dwójkowym systemie liczbowym znaki te mogą być wprowadzone tylko za pomocą odrębnego bitu znaku, którego wartość równa 1 oznacza znak „ – ” , a wartość 0 odpowiada znakowi „+”.

Zatem istnieją trzy sposoby kodowania liczb dwójkowych ze znakiem:

– znak-moduł (ZM),

– znak-uzupełnienie do 1 (ZU1),

– znak-uzupełnienie do 2 (ZU2).

ZM $+12_{10} \rightarrow 0.1100_2$

$-12_{10} \rightarrow 1.1100_2$

ZU1 $+12_{10} \rightarrow 0.1100_2$ *– jak w ZM*

$-12_{10} \rightarrow 1.0011_2$

ZU2 $+12_{10} \rightarrow 0.1100_2$ *– jak w ZM*

$-12_{10} \rightarrow 1.0100_2$

ODEJMOWANIE

W technice cyfrowej odejmowanie liczb dwójkowych wykonuje się poprzez dodawanie uzupełnionego odjemnika:

$$P - Q = P + (- Q)$$

To pozwala użyć zwykłe sumatory.

Najpierw odjemnik przedstawia się w jednym z kodów uzupełnieniowych a następnie wykonuje się dodawanie.

Czasami cyfrowe układy odejmujące nazywane są subtraktorami.

ODEJMOWANIE LICZB DZIESIĘTNYCH BEZ ZNAKU

$$\begin{array}{r} 54 \\ -\,30 \\ \hline 24 \end{array}$$

U9 (30) = 69

$$\begin{array}{rl} 54 & \text{ND} \\ +\;\;69 & \text{U9} \\ \hline 123 & \\ +1 & \\ \hline 24 & \text{ND} \\ & \\ +24 & \end{array}$$

dwa sumowania

U10 (30) = 70

$$\begin{array}{rl} 54 & \text{ND} \\ +\;\;70 & \text{U10} \\ \hline 124 & \\ \cancel{+1} & \\ 24 & \text{ND} \\ & \\ +24 & \end{array}$$

jedno sumowanie!

ODEJMOWANIE LICZB DZIESIĘTNYCH BEZ ZNAKU

$$\begin{array}{r} 30 \\ -\,54 \\ \hline -24 \end{array}$$

U9 (54) = 45

$$\begin{array}{rl} 30 & \text{ND} \\ +\ \ 45 & \text{U9} \\ \hline 075 & \\ +0 & \\ \hline 75 & \text{U9} \\ & \\ -24 & \end{array}$$

U10 (54) = 46

$$\begin{array}{rl} 30 & \text{ND} \\ +\ \ 46 & \text{U10} \\ \hline 076 & \text{U10} \\ & \\ -24 & \end{array}$$

ODEJMOWANIE LICZB DWÓJKOWYCH BEZ ZNAKU – *U1*

U1

NB(54) = 110110	NB(30) = 011110
U1(54) = 001001	U1(30) = 100001

54 – 30 = ...

```
   110110    NB
 + 100001    U1
 --------
 1←010111
       +1
 --------
 + 011000    NB

 + 24
```

30 – 54 = ...

```
   011110    NB
 + 001001    U1
 --------
 0←100111
       +0
 --------
 - 100111    U1

 - 24
```

ODEJMOWANIE LICZB DWÓJKOWYCH BEZ ZNAKU – *U2*

U2

NB(54) = 110110

U1(54) = 001001

U2(54) = 001010

NB(30) = 011110

U1(30) = 100001

U2(30) = 100010

54 – 30 = ...

```
        110110   NB
       +100010   U2
      ---------
    1 ← 011000

      + 011000   NB

      + 24
```

30 – 54 = ...

```
        011110   NB
       +001010   U2
      ---------
    0 ← 101000

      - 101000   U2

      - 24
```

ODEJMOWANIE LICZB DWÓJKOWYCH ZE ZNAKIEM – *U1*

	NB(37) = 100101	NB(12) = 001100
	U1(37) = 011010	U1(12) = 110011
ZU1	ZU1(–37) = 1.011010	ZU1(–12) = 1.110011

12 – 37 = ...

```
    0.001100
  + 1.011010
  ----------
  0←1.100110
          +0
  ----------
    1.100110  ZU1
        -25
```

37 – 12 = ...

```
    0.100101
  + 1.110011
  ----------
  1←0.011000
          +1
  ----------
    0.011001  ZM
        +25
```

ODEJMOWANIE LICZB DWÓJKOWYCH ZE ZNAKIEM – *U2*

NB(37) = 100101 NB(12) = 001100

U2(37) = 011011 U2(12) = 110100

ZU2 ZU2(–37) = 1.011011 ZU2(–12) = 1.110100

12 – 37 = ...

$$\begin{array}{r} 0.001100 \\ +\,1.011011 \\ \hline 1.100111 \end{array} \quad \text{ZU2}$$

-25

37 – 12 = ...

$$\begin{array}{r} 1.110100 \\ +\,0.100101 \\ \hline \cancel{1}\;0.011001 \end{array} \quad \text{ZM}$$

+25

PEWIEN PROBLEM SUMOWANIA...

NADMIAR – może wystąpić, gdy sumowane liczby są jednocześnie dodatnie lub ujemne.

$$
\begin{array}{rr}
 & +69 \\
+ & +103 \\
\hline
 & +172
\end{array}
\qquad
\begin{array}{rr}
 & 0.1000101 \\
+ & 0.1100111 \\
\hline
 & 1.0101100
\end{array}
$$

$c_z = 0$

$c_m = 1$

„– 84"
w kodzie ZU2 !

Stąd test nadmiaru:

$$c_z \oplus c_m = 1$$

Wówczas wiadomo, że błędny wynik!

PEWIEN PROBLEM SUMOWANIA...

Gdy test nadmiaru jest równy 1 to należy zwiększyć długość dodawanych liczb dwójkowych o jeden bit!

$$
\begin{array}{rr}
+69 & 0.01000101 \\
+103 & +\ 0.01100111 \\
\hline
 & 0.10101100
\end{array}
\qquad \text{„+ 172” w ZU2 !}
$$

$c_z = 0$

$c_m = 0$

UKŁADY ARYTMETYCZNE

– komparatory do porównywania dwóch lub więcej słów dwójkowych (liczb),

– sumatory (np. 7483),

– bloki arytmetyczno-logiczne (np. 74181),

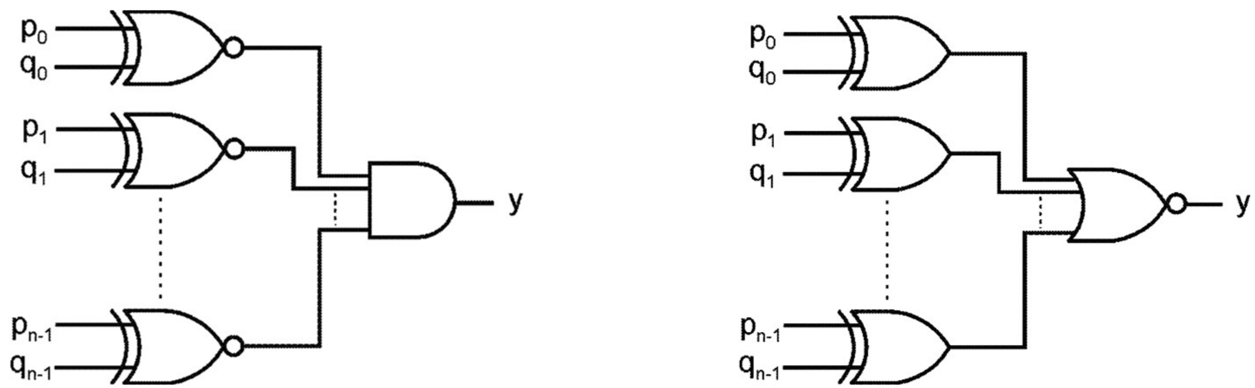
– bloki mnożące.

Komparatory

Wynikiem porównania może być sygnał wyjściowy równy 1 gdy porównywane liczby są równe, różne, jedna mniejsza od drugiej lub jedna większa od pozostałej.

Najprostszy komparator jednobitowy to bramka XNOR.
Wyjście ma stan 1 dla równych stanów na obu wejściach.

Komparator dwóch słów wielobitowych *P* i *Q* porównuje *i*-te bity:



Sumatory

Układy dodające liczby dwójkowe. Struktura zależy od rodzaju kodu, w którym są zapisane dodawane liczby – zazwyczaj w kodzie NB (sumator dwójkowy) oraz BCD (sumator dziesiętny).

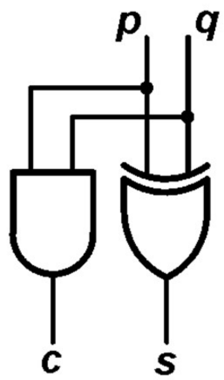
Sumator liczb dwójkowych bez znaku w kodzie NB

Najprostszy układ zwany *półsumatorem* operuje na dwóch liczbach jednobitowych *p* i *q* generując bit sumy *s* i przeniesienia *c*:

p	*q*	*s*	*c*
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$$s = p \oplus q$$

$$c = pq$$

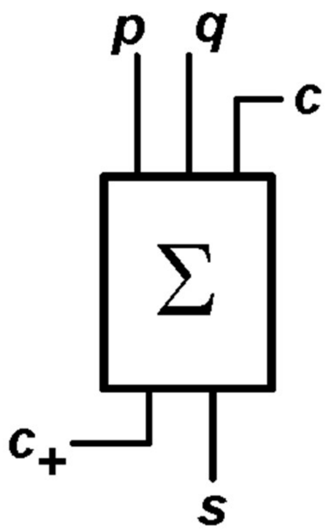


Sumator liczb dwójkowych bez znaku w kodzie NB

Aby realizować sumowanie liczb wielobitowych układ sumatora musi mieć dodatkowe wejście przeniesienia z pozycji poprzedzającej.

Dla *i*-tej pozycji taki układ nazywany jest *sumatorem jednobitowym*.

p	q	c	s	c_+
0	0	0	0	0
0	1	0	1	0
1	0	0	1	0
1	1	0	0	1
0	0	1	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1

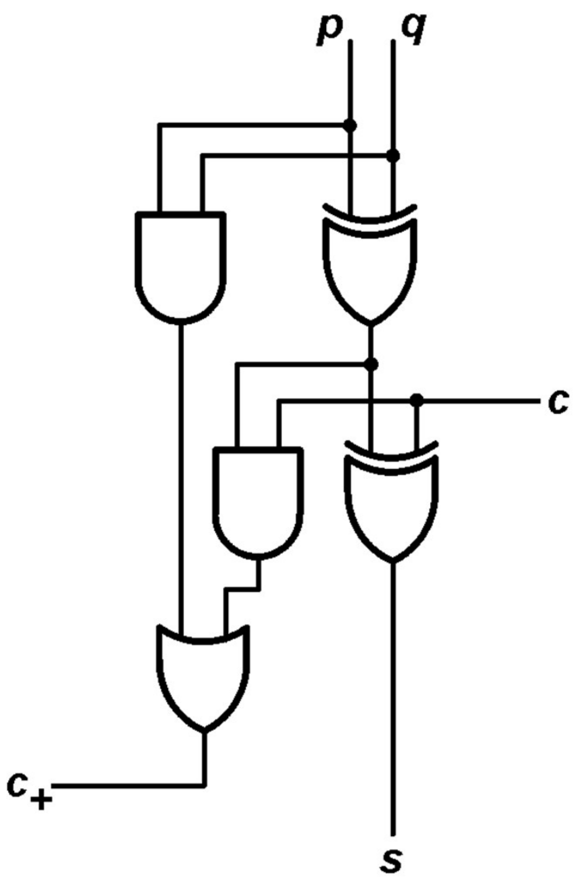


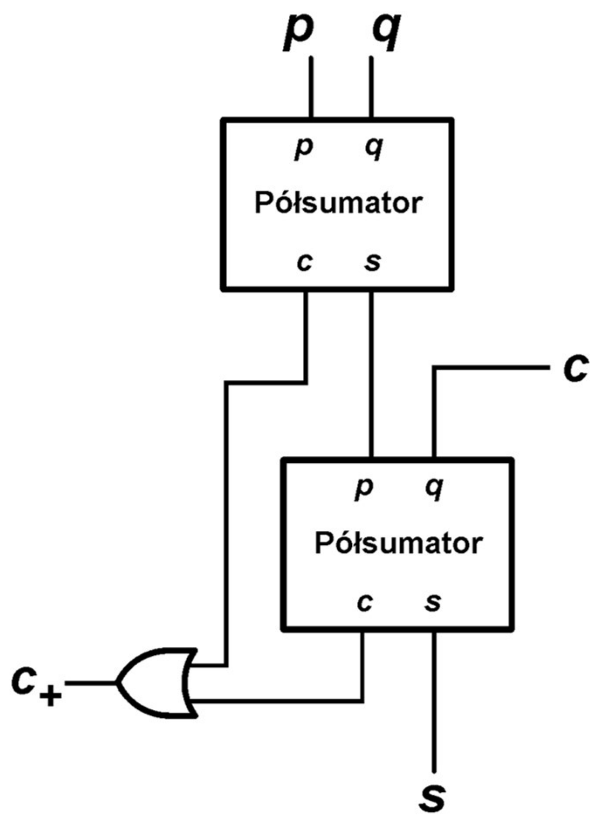
Sumator jednobitowy (tzw. pełny)

$$s = p \oplus q \oplus c$$

$$c_+ = pq + c(p \oplus q)$$

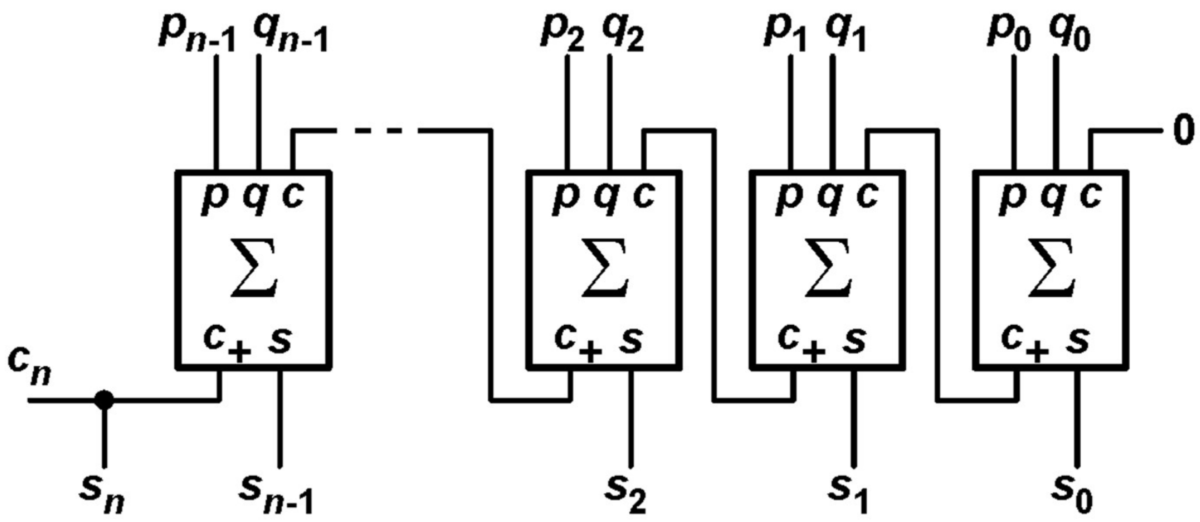
$$c_+ = pq + c(p + q)$$





Sumator *n*-bitowy

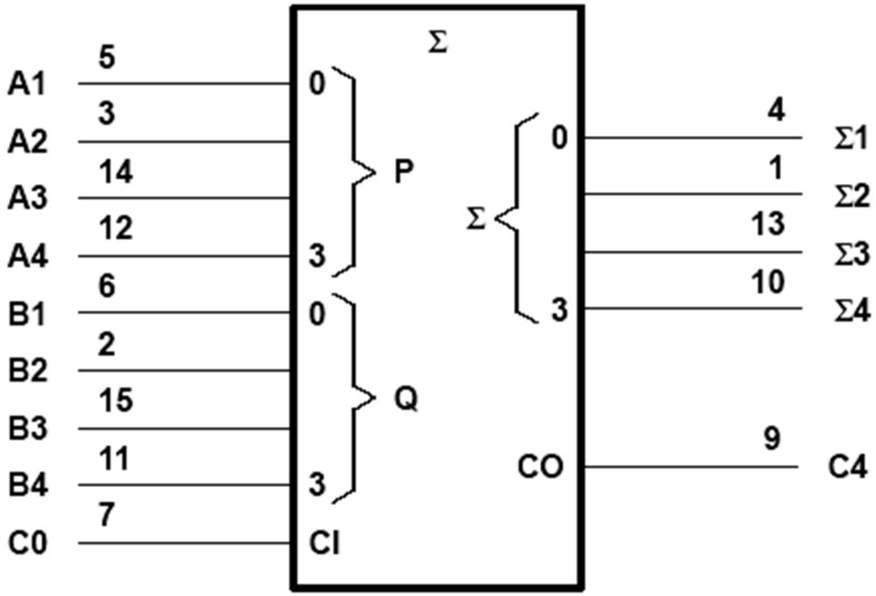
Otrzymywany przez połączenie sumatorów jednobitowych:



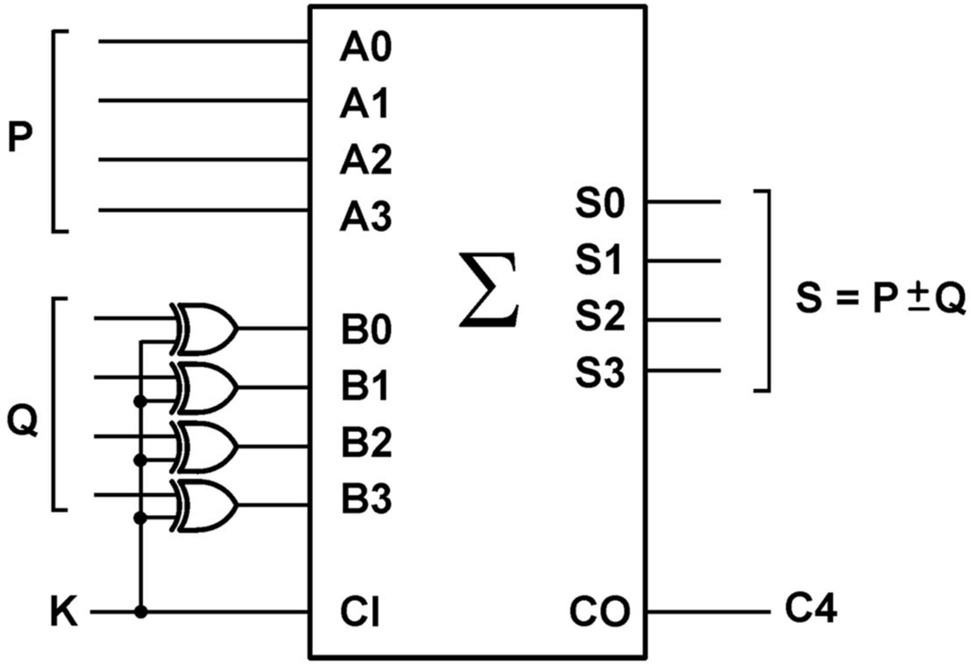
Dla dwóch liczb n-bitowych wynik ma n+1 bitów.

Wada: długi czas otrzymywania wyniku z powodu propagacji przeniesień

Sumator 4-bitowy – symbol graficzny (7483)



Sumator/subtraktor 4-bitowy w kodzie U2



K = 0 – sumator

K = 1 – subtraktor

UKŁADY MNOŻĄCE

– Przesuwanie w lewo (każde przesunięcie o 1 bit to mnożenie przez 2)

0000 0110	6
0000 1100	12
0001 1000	24
0011 0000	48
0110 0000	96

– **Sumowanie wielopoziomowe (matrycowe)**

			a_2	a_1	a_0
		*	b_2	b_1	b_0
			a_2b_0	a_1b_0	a_0b_0
		a_2b_1	a_1b_1	a_0b_1	
+	a_2b_2	a_1b_2	a_0b_2		
	Σ	Σ	Σ	Σ	Σ

– Sumowanie iteracyjne (akumulacyjne)