Uniwersytet Wrocławski

Wydział Fizyki i Astronomii
Instytut Fizyki Doświadczalnej

pl. M. Borna 9
50-204 Wrocław
tel. +48 71 375 93 02, +48 71 328 73 65
fax +48 71 328 73 65
e-mail: sekr@ifd.uni.wroc.pl
www.ifd.uni.wroc.pl

Elektrotechnika i elektronika (konspekt)

Franciszek Gołek **(golek@ifd.uni.wroc.pl)**

www.pe.ifd.uni.wroc.pl

Wykład 13

Bramki logiczne.
Wstęp do elektroniki cyfrowej.

Elektronika cyfrowa

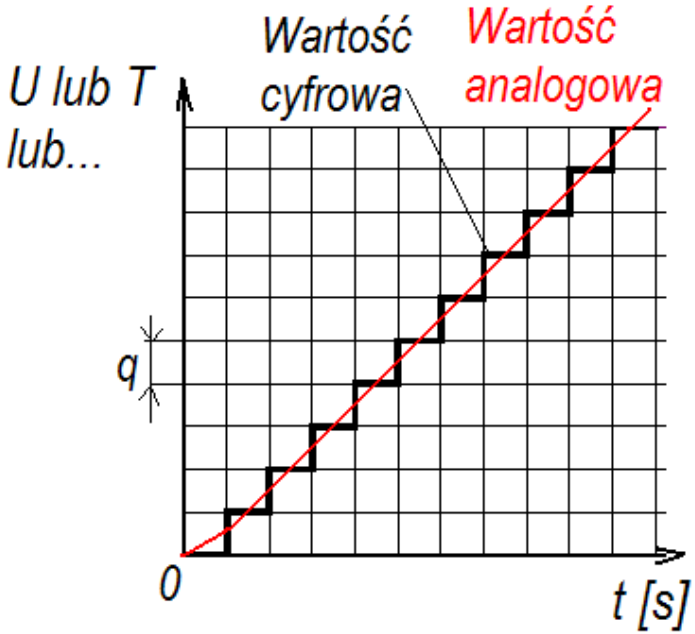
Przekaz informacji w postaci elektrycznego sygnału analogowego wykazuje zasadniczą wadę jaką jest ograniczona precyzja. Dominujący wpływ na ograniczenie precyzji sygnałów analogowych mają tzw. szumy elektryczne, których wielkość choć można obniżać to o ich całkowitej eliminacji mowy nie ma.

Sytuacja radykalnie się poprawia, gdy informacja jest kodowana w postaci elektrycznego sygnału cyfrowego. W tym przypadku zwykły szum nie stanowi poważnej przeszkody bo sygnały cyfrowe (nawet transmitowane na znaczne odległości) są łatwo oczyszczane z szumu. Istotne jest aby szum nie przekroczył wartości różnicy między stanami niskimi i wysokimi reprezentującymi zera i jedynki (– jedyne elementarne znaki w elektronice cyfrowej).

W układach cyfrowych rozróżniamy stany: wysoki (H – High) i stan niski (L – Low). Dokładna wartość stanu jest tu mniej istotna byle tylko mieściła się w odpowiednim dopuszczalnym przedziale wartości. W układach cyfrowych sygnały są ciągami zer i jedynek. Można nimi kodować dowolną informację, nawet przebiegi analogowe stosując przetworniki A/C (analogowo-cyfrowe) i ponownie przywracać pierwotną postać analogową stosując przetworniki C/A (cyfrowo-analogowe).

Dzięki ciągle postępującej miniaturyzacji i swoistej odporności na zakłócenia systemy cyfrowe pozwalają na przetwarzanie i długotrwałe magazynowanie olbrzymich ilości informacji.

W przypadku cyfryzacji sygnałów analogowych należy mieć na uwadze efekt kwantyzacji wartości w pomiarze, zapisie czy też odczycie.
Waga „q” najmniej znaczącej cyfry określa minimalną różnicę sygnałów (wielkości fizycznych), którą dany układ cyfrowy rozróżnia.



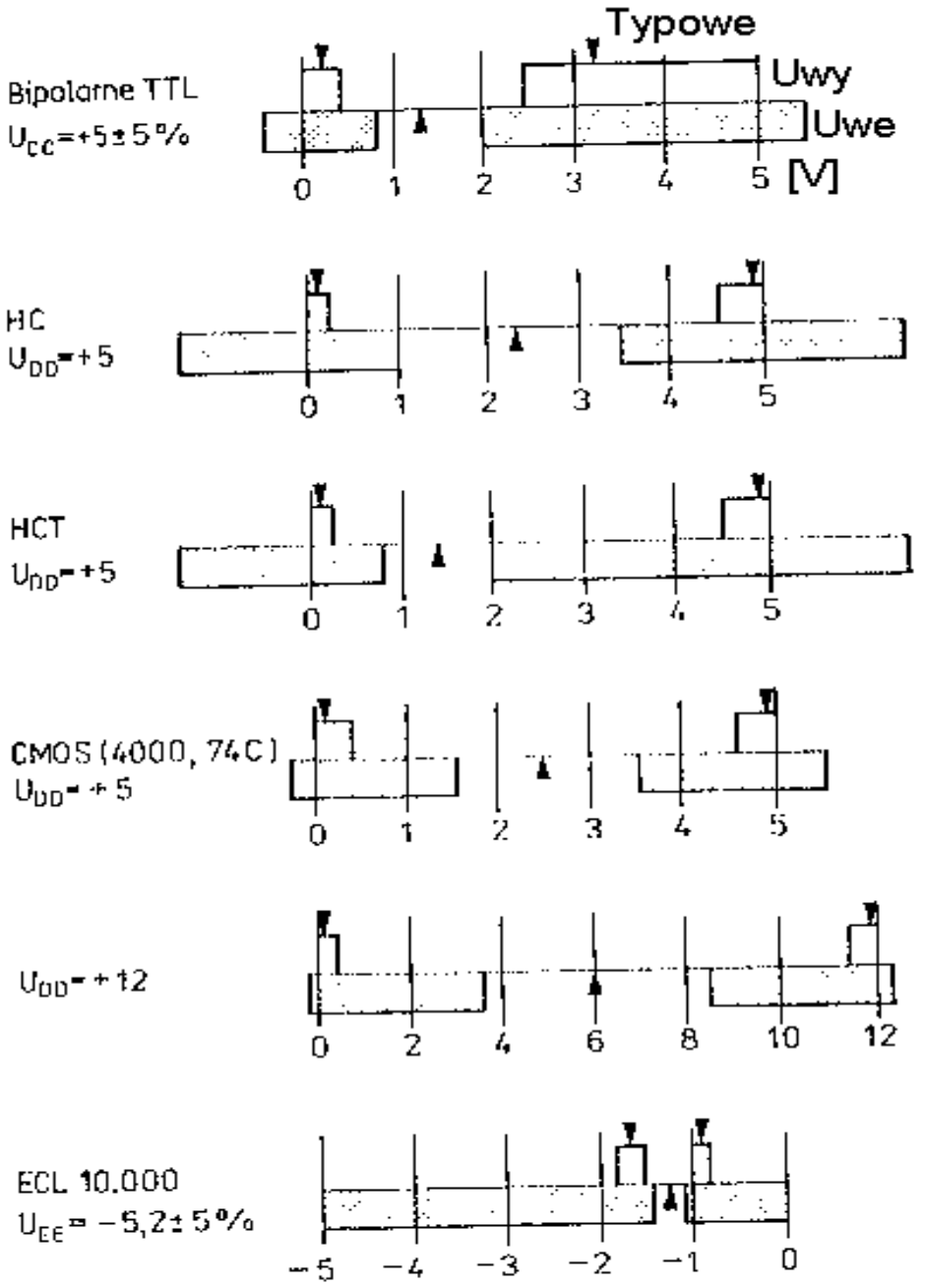
Wartości napięć stanów logicznych L i H (L - stan niski H - stan wysoki)

Przedziały nad osiami to przedziały napięć wyjściowych (wystawianych). Pod osiami zaznaczono przedziały rozpoznawania stanów pojawiających się na wejściach.

Górne strzałki pokazują wartości typowe.

Dolne strzałki pokazują granice miedzy L i H.

(P. Horowitz, W. Hill, *Sztuka elektroniki*)

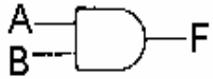
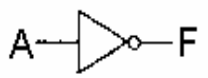
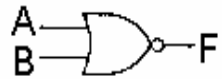
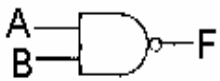
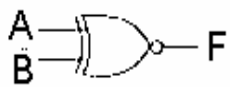
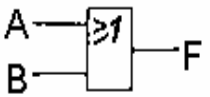
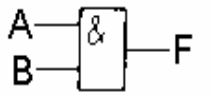
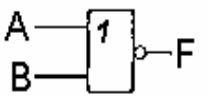
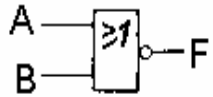
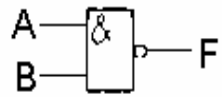
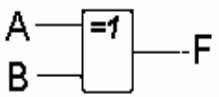
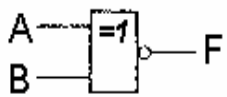


Bramki logiczne – to inaczej funktory realizujące proste operacje logiczne.

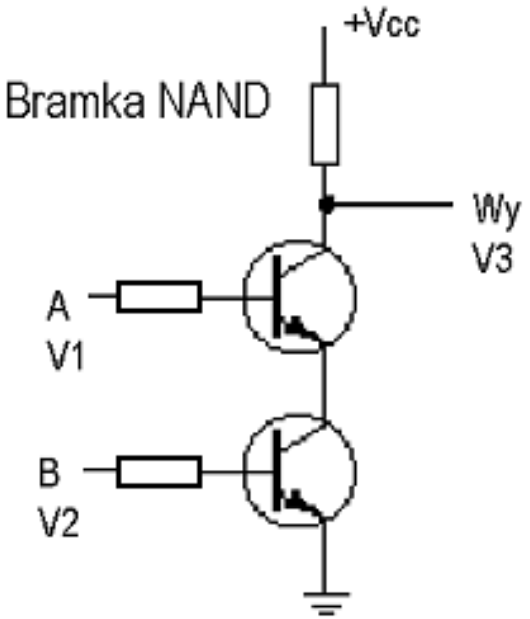
Działanie bramek definiują tzw. tablice (tabele) prawdy!

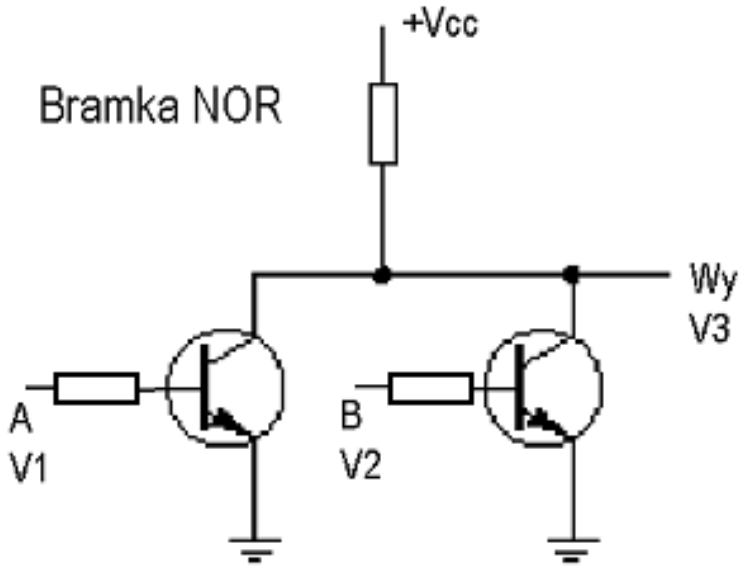
Tabela prawdy jest zestawieniem wszystkich wartości wyjściowych bramki (układu) dla wszystkich możliwych kombinacji wartości wejściowych.

Bramki — Najważniejsze funkcje logiczne, ich nazwy, tablice prawdy i symbole graficzne

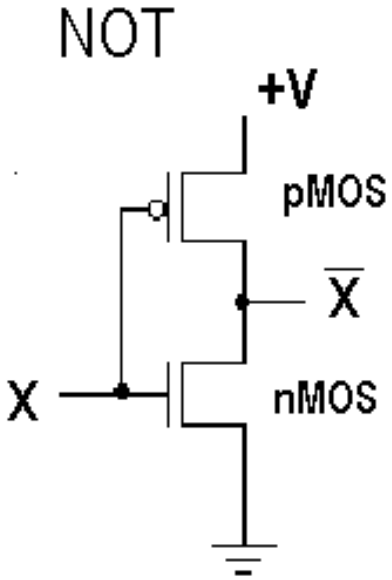
Argumenty		Wartości funkcji F						
A	B	Suma	Iloczyn	Negacja	Negacja sumy	Negacja iloczynu	Nietożsamość	Tożsamość
0	0	0	0	1	1	1	0	1
0	1	1	0	1	0	1	1	0
1	0	1	0	0	0	1	1	0
1	1	1	1	0	0	0	0	1
Wyrażenie logiczne		$F = A+B$	$F = AB$	$F = \overline{A}$	$F = \overline{A+B} = \overline{A}\,\overline{B}$	$F = \overline{AB} = \overline{A}+\overline{B}$	$F = A \oplus B = A\overline{B} + \overline{A}B$	$F = \overline{A \oplus B} = AB + \overline{A}\,\overline{B}$
Nazwy skrótowe		LUB, OR	I, AND	NIE, NOT	LUB – NIE, NOR	I – NIE, NAND	ALBO, EX-OR	ALBO – NIE, EX-NOR
Symbolr graficzne								
								

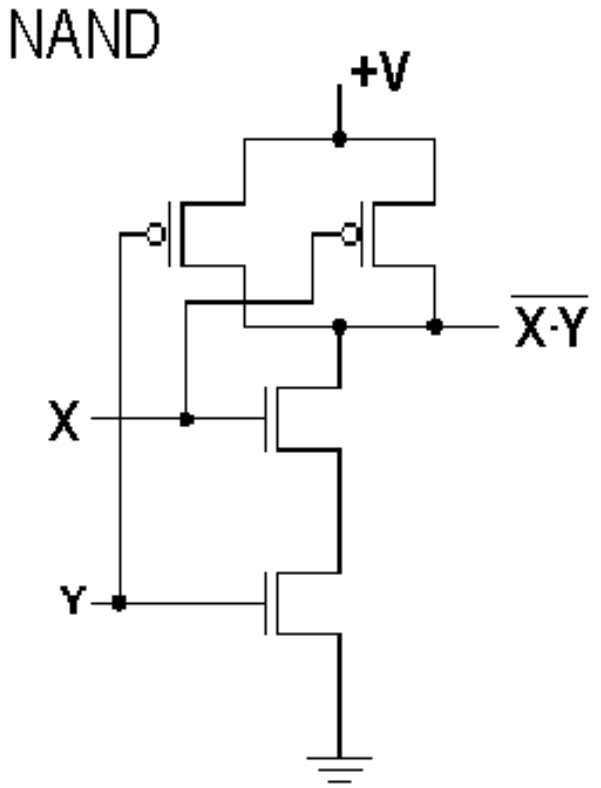
Bramki TTL:





Bramki CMOS:





Wybrane reguły algebry Boolea

1. $0 + X = X$
2. $1 + X = 1$
3. $X + X = X$
4. $X + \overline{X} = 1$
5. $0 \cdot X = 0$
6. $1 \cdot X = X$
7. $X \cdot X = X$
8. $X \cdot \overline{X} = 0$
9. $\overline{\overline{X}} = X$
10. $X + Y = Y + X$ } Komutacja
11. $X \cdot Y = Y \cdot X$ } Komutacja
12. $X + (Y + Z) = (X + Y) + Z$ } Asocjacja
13. $X \cdot (Y \cdot Z) = (X \cdot Y) \cdot Z$ } Asocjacja
14. $X \cdot (Y + Z) = X \cdot Y + X \cdot Z$
15. $X + X \cdot Z = X$
16. $X \cdot (X + Y) = X$
17. $(X + Y) \cdot (X + Z) = X + Y \cdot Z$
18. $X + \overline{X} \cdot Y = X + Y$
19. $X \cdot Y + Y \cdot Z + \overline{X} \cdot Z = X \cdot Y + \overline{X} \cdot Z$

Prawa De Morgan'a

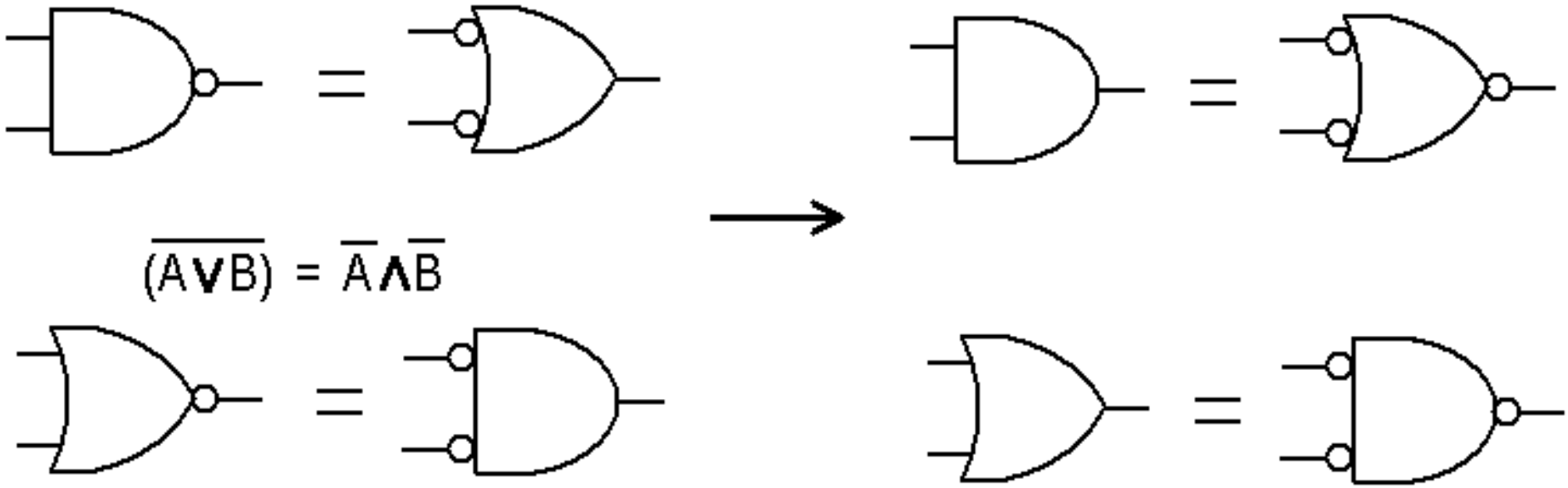
$$\overline{(X + Y)} = \overline{X} \cdot \overline{Y}$$

$$\overline{(X \cdot Y)} = \overline{X} + \overline{Y}$$

Wskazują, że dowolną funkcję logiczną można otrzymać stosując tylko bramki OR i NOT albo tylko AND i NOT.

Prawa De Morgana

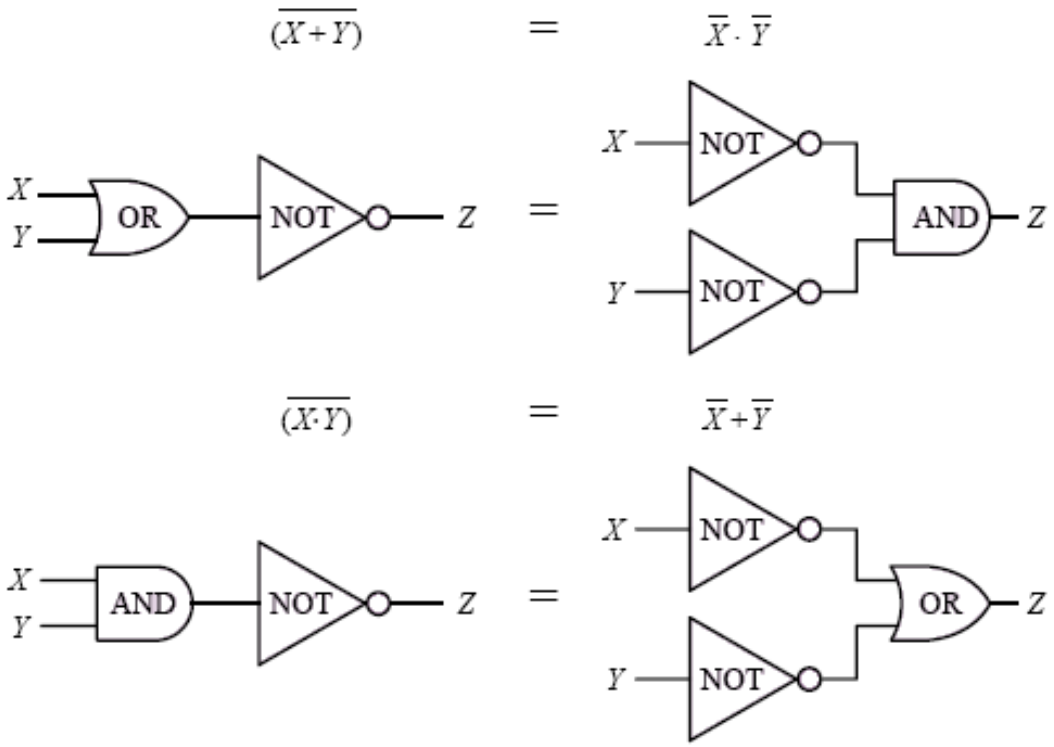
$\overline{(A \wedge B)} = \overline{A} \vee \overline{B}$



$\overline{(A \vee B)} = \overline{A} \wedge \overline{B}$

Należy pamiętać, że bramka AND jest iloczynem (AND-em) dla stanów wysokich traktowanych jako stany aktywne, a dla stanów niskich jest sumą logiczną. Podobnie bramka OR, dla stanów niskich (będących stanami aktywnymi) działa jak iloczyn logiczny.

Ilustracja praw De Morgana



Tablice prawdy

X	Y	$Z=\overline{(X+Y)}=\overline{X}\cdot\overline{Y}$
0	0	1
0	1	0
1	0	0
1	1	0

X	Y	$Z=\overline{(X\cdot Y)}=\overline{X}+\overline{Y}$
0	0	1
0	1	1
1	0	1
1	1	0

Przykład: przedstaw funkcję logiczną, która zezwala na start samolotu gdy co najmniej dwóch z trzech pilotów wykazują aktywność (X – 1-pilot siedzi za sterami, Y – 2-pilot siedzi za sterami, Z – autopilot jest czynny).

Rozw. $f = X\cdot Y + X\cdot Z + Y\cdot Z$; gdy $f = 1$ mamy zezwolenie na start. Warto odnotować, że (stosując prawa De Morgana) stosując negację funkcji f zamieniamy sumę iloczynów na iloczyn sum dostajemy funkcję g, która dla wartości $g = 1$ oznacza zakaz startu!

$$g = \overline{f} = \overline{X\cdot Y + X\cdot Z + Y\cdot Z} = (\overline{X}+\overline{Y})\cdot(\overline{X}+\overline{Z})\cdot(\overline{Y}+\overline{Z})$$

Ważna oczywistoś

Iloczyn daje jedynkę dla tylko jednej kombinacji czynników!

$Y = \bar{A} \cdot B \cdot C = 1$ tylko dla jednej kombinacji czynników: $A = 0$, $B = 1$ i $C = 1$.

Suma natomiast wyróżnia jedyną kombinację dla wyniku zero!

$Y = \bar{A} + B + C = 0$ tylko dla jednej kombinacji Składników: $A = 1$, $B = 0$ i $C = 0$.

Przykład: Zbudować układ z bramek logicznych realizujący funkcję Y = Y(A,B,C) zdefiniowaną poprzez tablicę prawdy:

A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Rozw. Należy zacząć od zamiany tablicy na wyrażenie logiczne.
Zaczynamy od drugiej linii bo dla linii pierwszej Y = 0 urządzenie jest zbyteczne (wybieramy linie gdzie Y = 1) i piszemy iloczyny dające wartość 1.

$$Y = \overline{A}\cdot\overline{B}\cdot C + \overline{A}\cdot B\cdot C + A\cdot\overline{B}\cdot\overline{C} + A\cdot\overline{B}\cdot C + A\cdot B\cdot\overline{C} + A\cdot B\cdot C$$
$$= \overline{A}\cdot C\cdot(\overline{B}+B) + A\cdot\overline{B}\cdot(\overline{C}+C) + A\cdot B\cdot(\overline{C}+C)$$
$$= \overline{A}\cdot C + A\cdot\overline{B} + A\cdot B$$
$$= \overline{A}\cdot C + A\cdot(\overline{B}+B)$$
$$= \overline{A}\cdot C + A$$
$$= A + C.$$

Czyli rozwiązaniem jest pojedyncza bramka OR podłączona tylko do źródeł sygnałów A i C!

Kilka rodzajów bramek logicznych stanowi wystarczającą bazę aby budować urządzenia cyfrowe i komputerowe zdolne wykonywać różnorodne zadania i pełnić rozmaite funkcje.

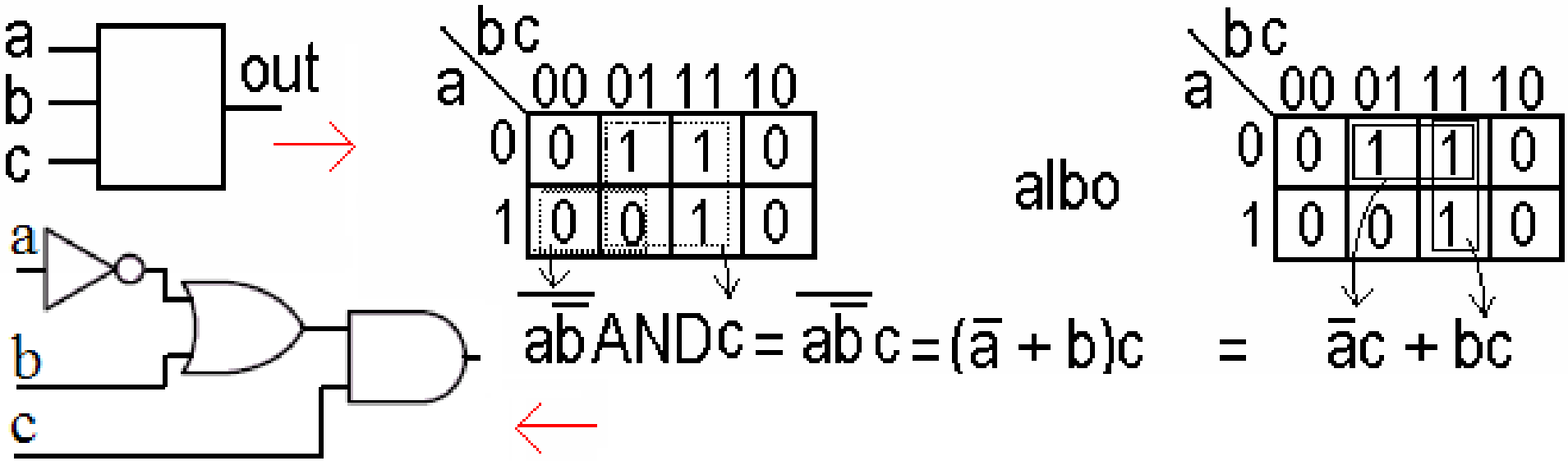
Metoda Karnaugh

Jest to metoda znajdywania minimalnej formuły (minimalnej ilości bramek logicznych) dla zadanej funkcji Boolowskiej przy małej liczbie zmiennych. Metoda ta nie wymaga takiego sprytu jak przy przekształceniach i stopniowym upraszczaniu wyrażeń Boolowskich. Metoda polega na zapisaniu mapy Karnaugh'a, która jest w zasadzie tabelą prawdy projektowanego i minimalizowanego układu kombinacyjnego a następnie zastosowaniu następujących reguł i czynności:

1) Pogrupować „jedynki" w czworokątne bloki zawierające 2^n jedynek (1, 2, 4, 8 itd.).

2) Starać się aby te bloki były możliwie duże.

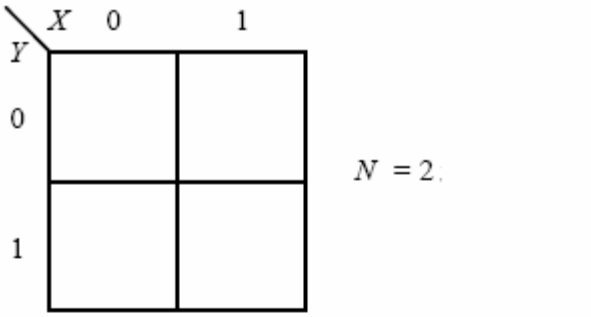
3) Odczytać zmienne - współrzędne bloków i stany wyjściowe w blokach i to one zostają ważnymi zmiennymi, reszta jest zbędna.

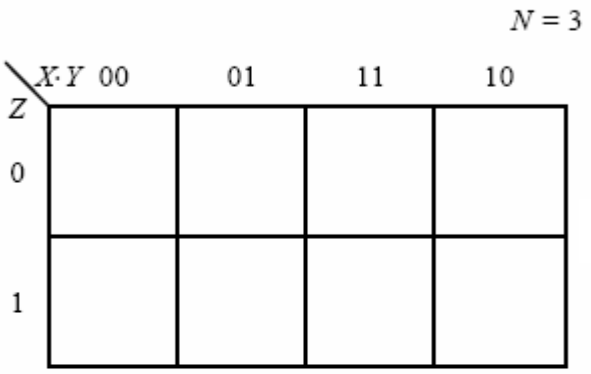


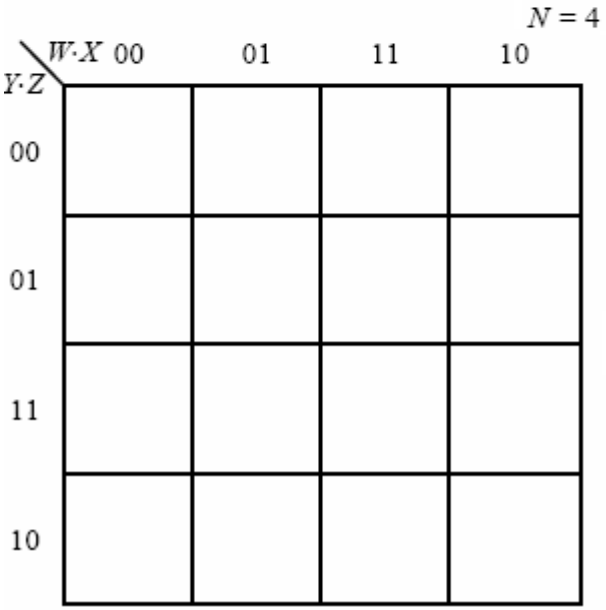
Układanie map Karnaugh polega na zestawieniu 2^N ramek dla N zmiennych w sposób przedstawiony na rysunku. Ramki zawierają pożądane wartości funkcji dla odpowiednich kombinacji zmiennych.

Kombinacje zmiennych (czyli – „współrzędne”) dla sąsiednich ramek różnią się tylko jednym bitem. Ponadto należy przyjąć, że zewnętrzne brzegi są ze sobą „sklejane” (co pozwala na „rolowanie mapy w pionie lub w poziomie i nadal „współrzędne” dla sąsiednich ramek różnią się tylko jednym bitem).

Mapa Karnaugh przedstawia wartości funkcji w dogodnej postaci graficznej.







X	Y	Z	Wartość funkcji
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Tabela prawdy

	$\overline{X}\overline{Y}$	$\overline{X}Y$	XY	$X\overline{Y}$
$\overline{Z}$	0	1	1	0
Z	0	1	1	0

Mapa Karnaugha

W tabeli prawdy w zasadzie możemy interesować sie albo tymi kombinacjami zmiennych, które skutkują wartościami = 1 na wyjściu albo tymi, które skutkują wartościami funkcji = 0.

Dla wartości funkcji 1 wyróżniony jest iloczyn logiczny (daje 1 tylko dla jednej kombinacji zmiennych – samych jedynek).

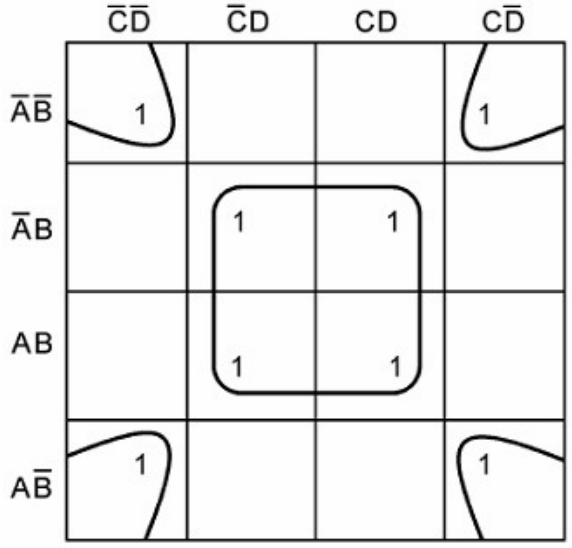
Dla wartości funkcji 0 wyróżnioną jest suma logiczna (daje zero dla jednej tylko kombinacj zmiennych tj. samych zer)

Inputs			*Output*	*Product terms*	*Sum terms*
A	*B*	*C*	*Y*		
0	0	0	0		A + B + C
0	0	1	0		A + B + C′
0	1	0	1	A′BC′	
0	1	1	0		A + B′ + C′
1	0	0	1	AB′C′	
1	0	1	1	AB′C	
1	1	0	1	ABC′	
1	1	1	0		A′ + B′ + C′

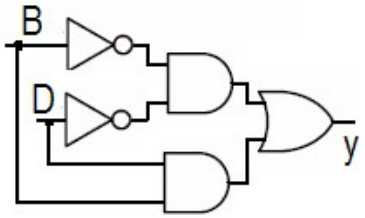
Mapa Karnaugh jest graficzną reprezentacją układu logicznego (lub systemu układów logicznych) i jest stosowalna do małej ilości zmiennych praktycznie do 6 zmiennych. (generalnie funkcja logiczna dla n zmiennych może być reprezentowana przez 2^{n-4} map Karnaugh). Mapa Karnaugh może służyć do bezpośredniego napisania funkcji logicznej w postaci sumy iloczynów („minterm") lub w postaci iloczynu sum („maxterm"). W przypadku mapy Karnaugh typu „minterm" wstawiamy „1" w te pola, dla których stan wyjściowy wynosi „1" a „0" tam gdzie wyjściowy stan wynosi „0". W przypadku mapy Karnaugh typu „maxterm" wyróżniamy te pola, dla których wyjście przyjmuje wartość „0" (a nawet możemy wstawiać „1" w pola, dla których wyjście przyjmuje wartość „0" a „0" w pola, które odpowiadają stanom wyjściowym „1").

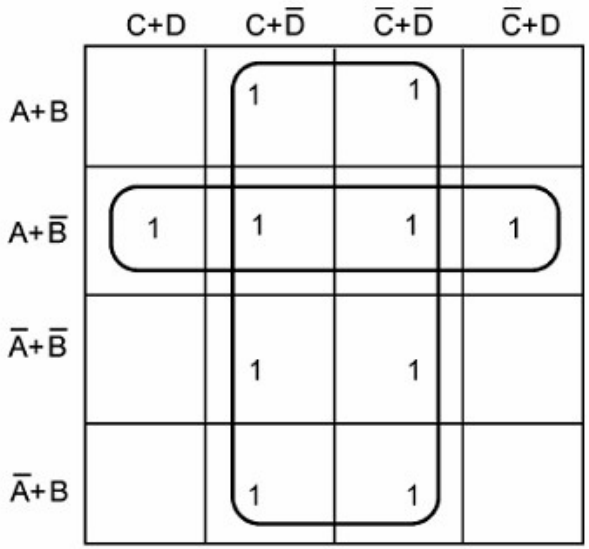
W obu przypadkach koncentrujemy sie na wyróżnionych polach.

Przy układaniu funkcji logicznej każda „1" – czyli wyróżnione pole musi być uwzględniona (przynajmniej raz). Rysunek pokazuje przykładowy minterm i przykładowy maxterm.

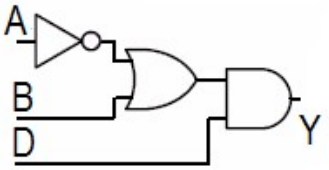


$y = \bar{B}\bar{D} + BD$



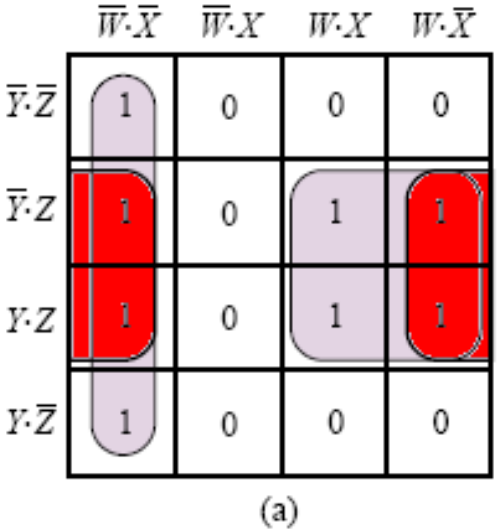


$Y = \bar{D}.(A+\bar{B})$

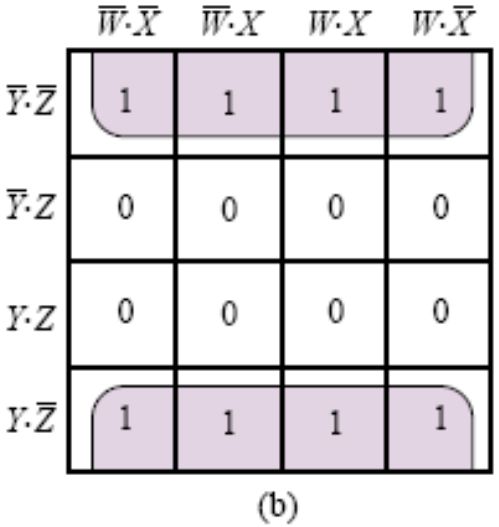


Po utworzeniu mapy Karnaugh zakreślamy obszary (możliwie duże) przylegających do siebie „jedynek”. Zasada minimalizacji wyrażenia funkcji logicznej polega na stosowaniu algebry Boolea do eliminowania niektórych elementów wyrażenia jak np.:

$$Y \cdot X + Y \cdot \overline{X} = Y \cdot (X + \overline{X}) = Y, \text{ bo } X + \overline{X} = 1 \text{ zawsze.}$$



(a)



(b)

Przykłady 4-polowych i 8-polowych grup jedynek. (przylegających do siebie jedynek!)

Przykłady z 4-polowym i 8-polowym obszarem jedynek. Z przykładów widać, że im większe obszary jedynek tym prostszy układ do realizacji funkcji!

	$\overline{W}\cdot\overline{X}$	$\overline{W}\cdot X$	$W\cdot X$	$W\cdot\overline{X}$
$\overline{Y}\cdot\overline{Z}$	0	0	0	0
$\overline{Y}\cdot Z$	1	1	1	1
$Y\cdot Z$	0	0	0	0
$Y\cdot\overline{Z}$	0	0	0	0

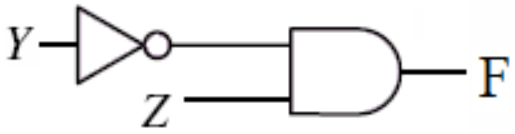
$$F =$$

$$\overline{W}\cdot X\cdot\overline{Y}\cdot Z+\overline{W}\cdot\overline{X}\cdot\overline{Y}\cdot Z+W\cdot\overline{X}\cdot\overline{Y}\cdot Z+W\cdot X\cdot\overline{Y}\cdot Z$$

$$=\overline{W}\cdot Z\cdot\overline{Y}\cdot(X+\overline{X})+W\cdot\overline{Y}\cdot Z\cdot(\overline{X}+X)$$

$$=\overline{W}\cdot Z\cdot\overline{Y}+W\cdot\overline{Y}\cdot Z$$

$$=\overline{Y}\cdot Z\cdot(\overline{W}+W)=\overline{Y}\cdot Z$$



	$\overline{W}\cdot\overline{X}$	$\overline{W}\cdot X$	$W\cdot X$	$W\cdot\overline{X}$
$\overline{Y}\cdot\overline{Z}$	1	1	1	1
$\overline{Y}\cdot Z$	1	1	1	1
$Y\cdot Z$	0	0	0	0
$Y\cdot\overline{Z}$	0	0	0	0

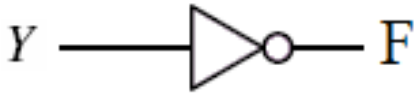
$$F =$$

$$\overline{W}\cdot\overline{X}\cdot\overline{Y}\cdot\overline{Z}+\overline{W}\cdot X\cdot\overline{Y}\cdot\overline{Z}+W\cdot X\cdot\overline{Y}\cdot\overline{Z}+W\cdot\overline{X}\cdot\overline{Y}\cdot\overline{Z}$$

$$+\overline{W}\cdot\overline{X}\cdot\overline{Y}\cdot Z+\overline{W}\cdot X\cdot\overline{Y}\cdot Z+W\cdot X\cdot\overline{Y}\cdot Z+W\cdot\overline{X}\cdot\overline{Y}\cdot Z$$

$$=\overline{W}\cdot\overline{Y}\cdot\overline{Z}+W\cdot\overline{Y}\cdot\overline{Z}+\overline{W}\cdot\overline{Y}\cdot Z+W\cdot\overline{Y}\cdot Z$$

$$=\overline{Y}\cdot\overline{Z}+\overline{Y}\cdot Z=\overline{Y}$$

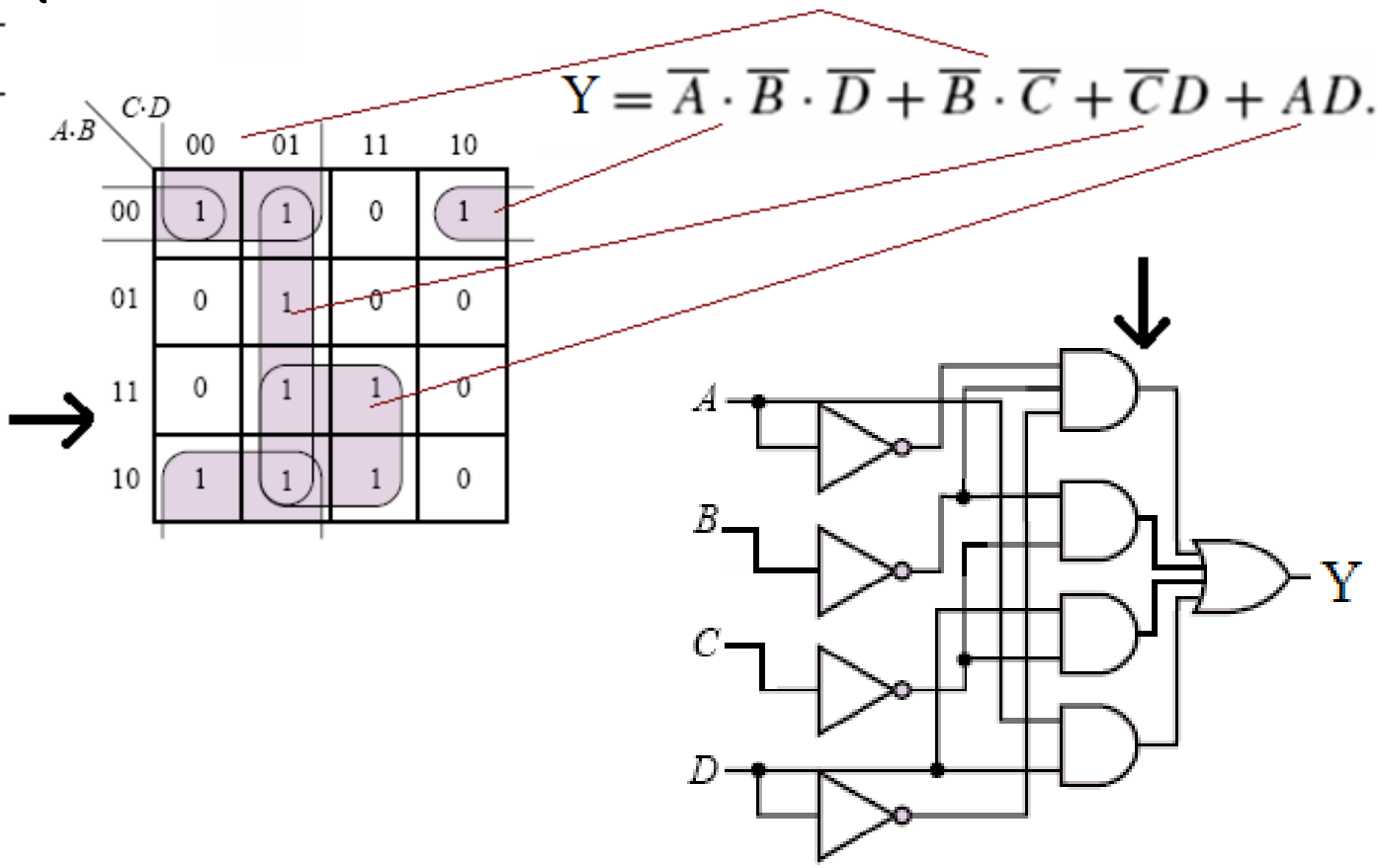


W powyższych przykładach startowaliśmy od sum iloczynów.

W takim podejściu obowiązują zasady: 1) Zaczynać od izolowanych jedynek, tu nic się nie upraszcza, 2) Wybrać odizolowane pary jedynek, 3) Następnie wybieramy 4-polowe, 8-polowe obszary itd.. 4) Minimalne wyrażenie formujemy przez zebranie najmniejszej liczby maksymalnych obszarów (obejmujących jedynki).

Przykład: Zminimalizować funkcję gdy dostępne są wielowejściowe bramki (2,3 i 4 wejściowe). Rozwiązanie:

A	B	C	D	Y
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1



Przykład: Mając do dyspozycji bramki a) ADN i NOT albo b) tylko NAND zrealizować funkcję:

$$f = (\overline{x} + \overline{y}) \cdot (y + \overline{z})$$

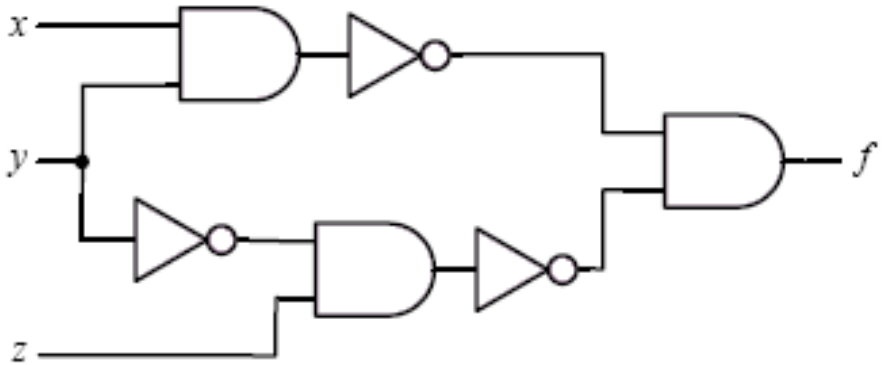
Rozwiązanie: Stosując prawa De Morgana otrzymamy:

$$\overline{x} + \overline{y} = \overline{x \cdot y}$$

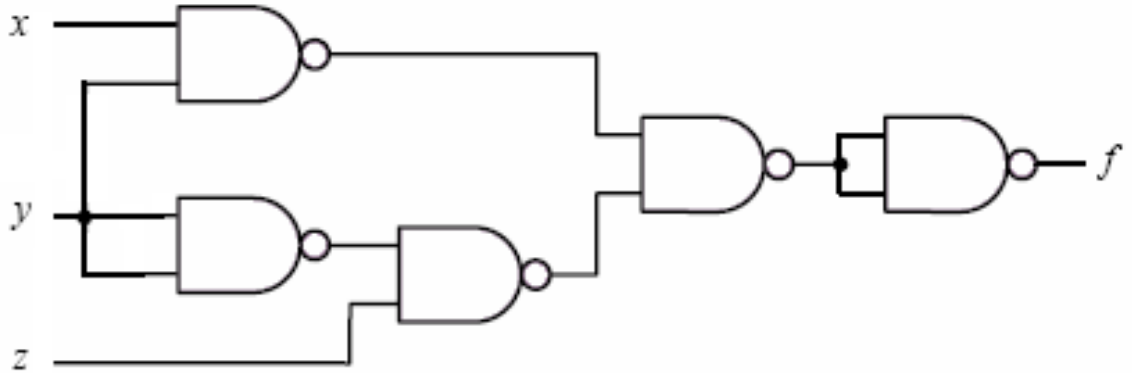
$$y + \overline{z} = \overline{\overline{z} \cdot \overline{y}}$$

$$f = (\overline{x \cdot y}) \cdot (\overline{\overline{z} \cdot \overline{y}})$$

a)



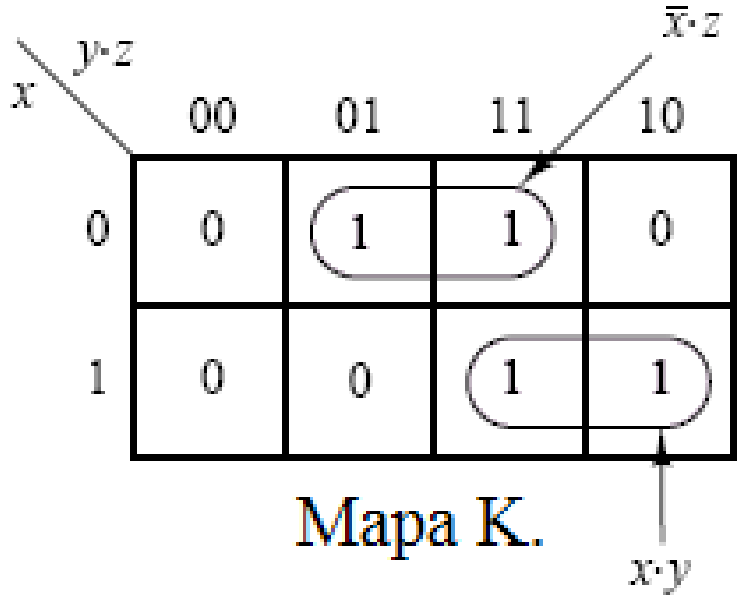
b)



Przykład: Z pomocą mapy K uprościć funkcję:

$$f = x \cdot y + \overline{x} \cdot z + y \cdot z$$

Rozwiązanie: Wykonujemy mapę K ->



Mapa K.

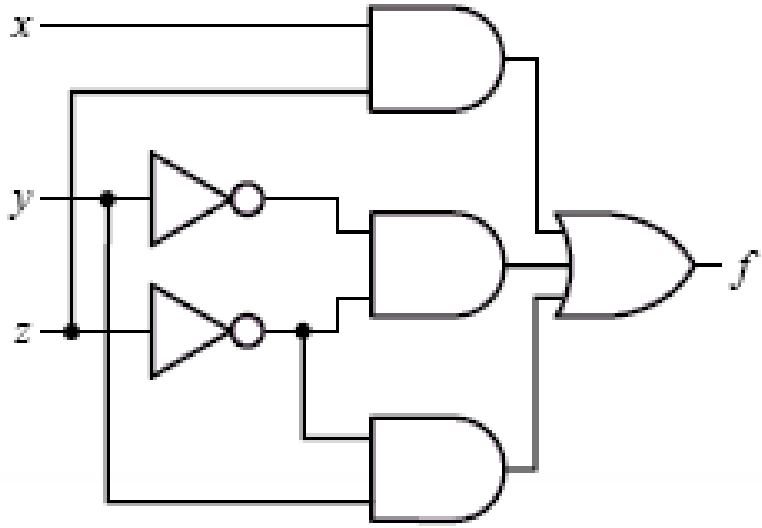
Z mapy odczytujemy:

$$f = x \cdot y + \overline{x} \cdot z.$$

Zatem y·z znika z wyrażenia funkcji gdyż jedynki z tego iloczynu są już wygenerowane dwoma poprzedzającymi iloczynami

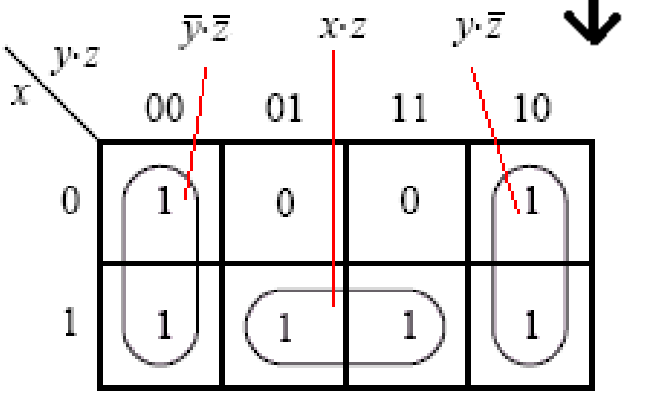
(trzeci pionowy obszar jedynek jest „załatwiony” dwoma poziomymi).

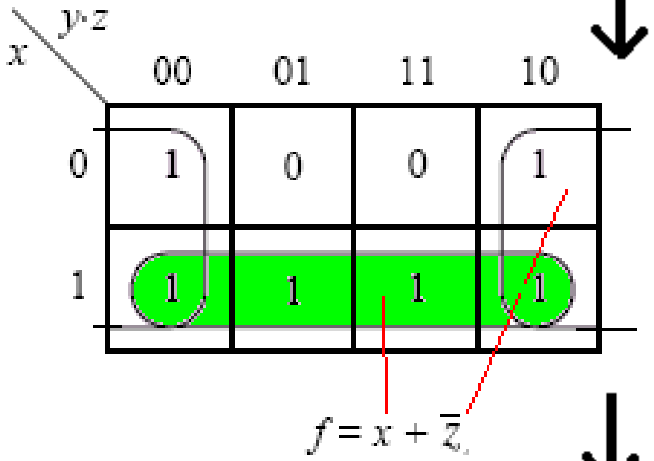
Przykład: Uprościć układ:

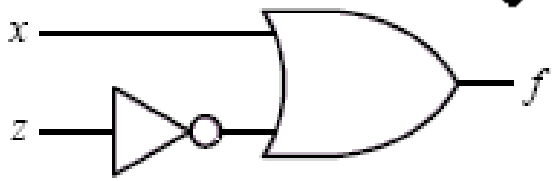


Rozwiązanie: Znajdujemy funkcję:

$$f = (x \cdot z) + (\bar{y} \cdot \bar{z}) + (y \cdot \bar{z})$$







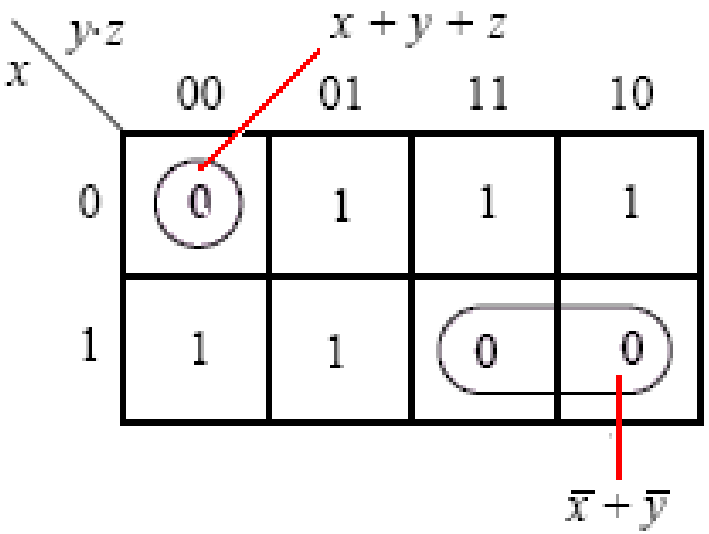
Metoda stosowania iloczynu sum. W poprzednich przykładach stosowaliśmy metodę sumy iloczynów. Z praw De Morgana wynika, że równie dobre mogą być iloczyny sum (czasem mogą prowadzić do mniejszej ilości bramek)!

W tej metodzie zakreślamy obszary sąsiadujących zer (nie jedynek) i kompletujemy wynikające z nich wyrażenie.

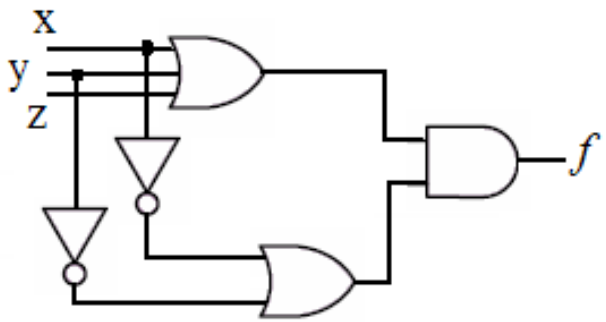
Przykład. Zrealizować funkcję przedstawioną za pomocą tabeli.

Rozwiązanie: Budujemy mapę K, zakreślamy obszary zer i znajdujemy wyrażenie na funkcję w postaci iloczynu sum:

x	y	z	f
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0



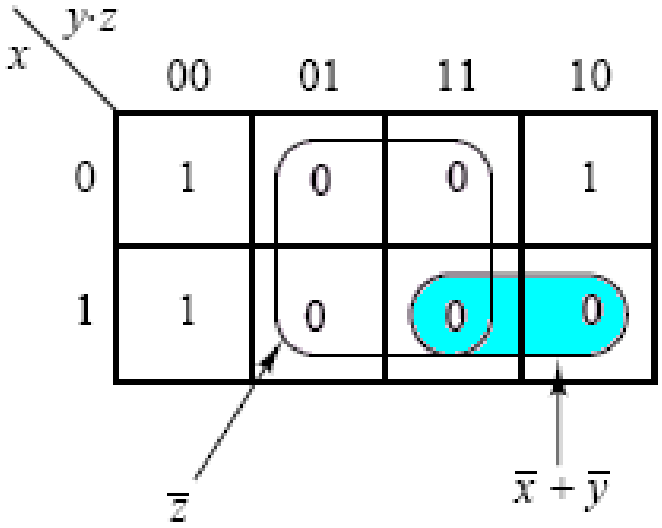
$$f = (x + y + z) \cdot (\overline{x} + \overline{y})$$



Przykład. Zrealizuj funkcję logiczną opisaną przez tablicę prawdy. Funkcja ma być w formie iloczynu sum.

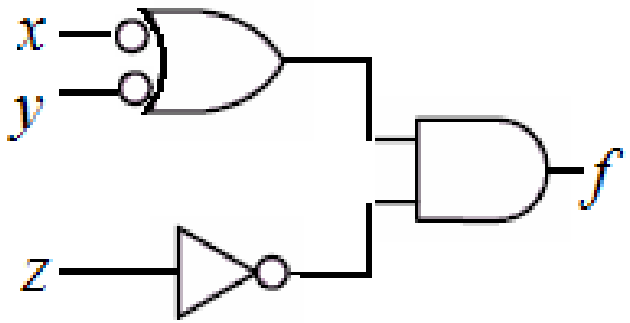
x	y	z	f
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

Rozwiązanie: budujemy mapę K i zakreślamy obszary zer.



Z mapy odczytujemy następującą funkcję:

$$f = \overline{z} \cdot (\overline{x} + \overline{y})$$



Istotą techniki cyfrowej jest wytwarzanie cyfrowych sygnałów wyjściowych jako odpowiedzi na cyfrowe sygnały wejściowe realizując odpowiednią funkcję logiczną lub arytmetyczną.

Układy kombinacyjne

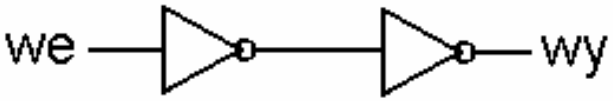
Układy, dla których sygnały (stany) wyjściowe zdeterminowane są aktualnymi stanami wejściowymi nazywamy układami kombinacyjnymi.

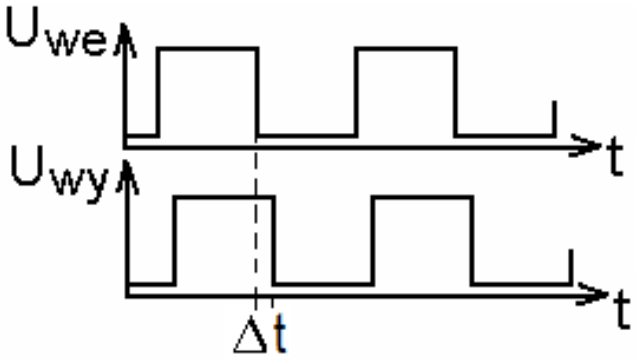
Należy jednak pamiętać, że stan wyjściowy ustala się dopiero po tzw. czasie propagacji (przejścia sygnału przez dany układ) od momentu zmiany stanów wejściowych. Bramki logiczne są układami kombinacyjnymi. **Czas propagacji** przez pojedynczą bramkę może wynosić od 1ns do 10ns - oznacza to, że szeregowe łączenie bramek zwiększa czas propagacji do znacznych wartości szkodliwych dla działania szybkich układów cyfrowych.

Układy sekwencyjne

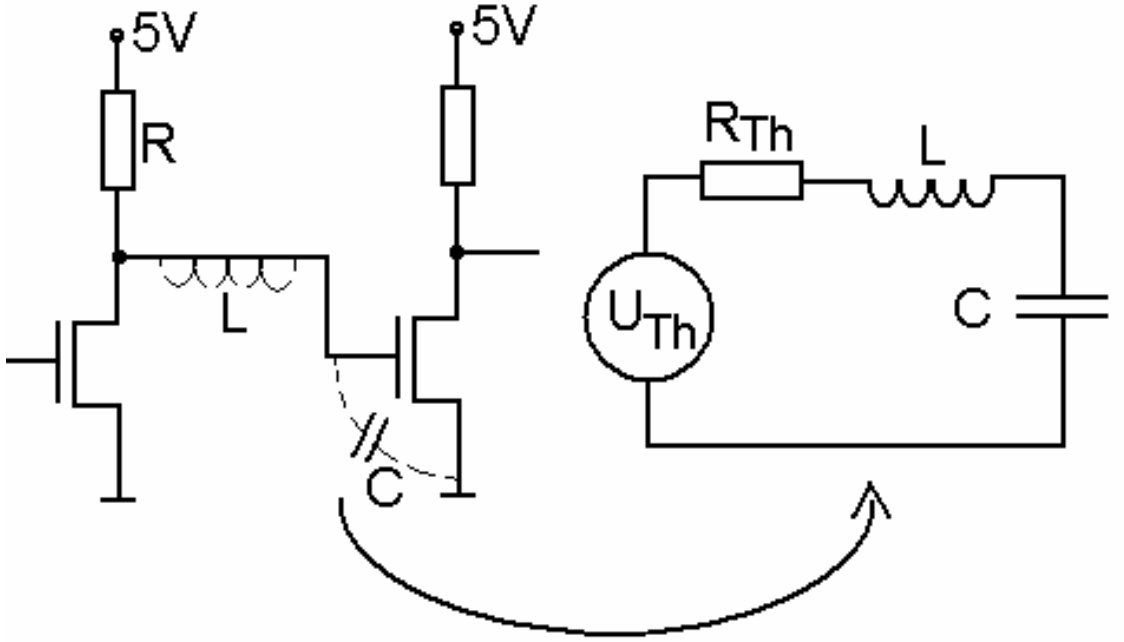
Układy, dla których sygnały (stany) wyjściowe zdeterminowane są nie tylko aktualnymi stanami wejściowymi ale zależą od stanów poprzednich (występuje pamięć) nazywamy układami sekwencyjnymi. W tych układach czas propagacji też odgrywa istotną rolę.

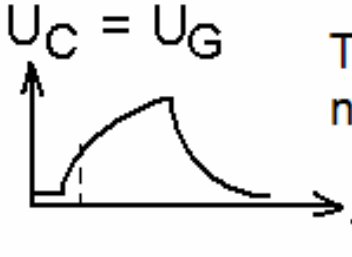
Szybkość przełączania Szybkie działanie (szybkie i częste przełączania) układów cyfrowych ograniczają takie czynniki jak: a) wydzielana moc (duża ilość ciepła). b) skończony czas propagacji sygnału wynikający z wielu przyczyn, np. resztkowe (pasożytnicze) pojemności i indukcyjności, długość połączenia itp.



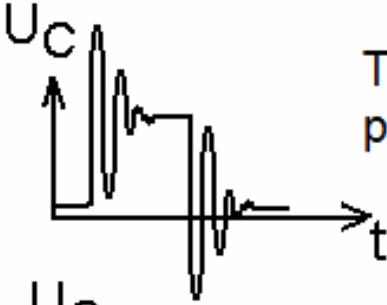


Δt - czas propagacji





Tłumienie nadkrytyczne
Duże R_{Th}



Tłumienie podkrytyczne
Małe R_{Th}

Tłumienie okołokrytyczne
$Q \gtrapprox 1$

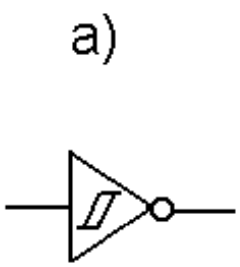
Uwaga o zakłóceniach w elektronice cyfrowej

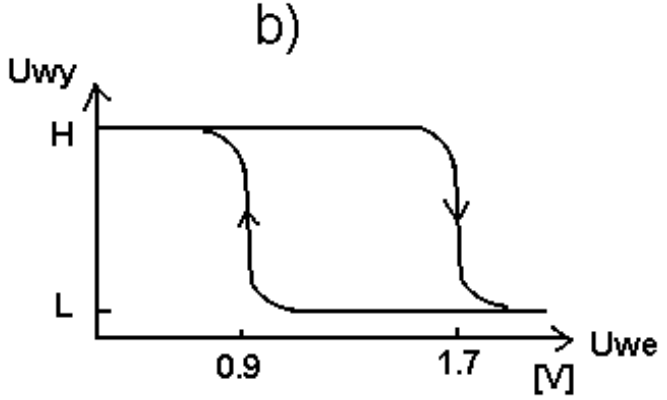
Jeżeli narosty impulsów są tak krótkie, że wynoszą około 1ns (10^{-9}s przy szybkości transmisji sygnału około $3x10^8$m/s) to połączenia o długości zaledwie kilku cm należy traktować jako linie długie. Przyczynami zakłóceń mogą być: A) Odbicia sygnału od niedopasowanych impedancji połączonych ze sobą odcinków linii sygnałowych. B) Pojawianie się szpilek napięciowych na liniach sygnałowych. Napięcie to powstaje jako skok nawet ponad 1V na indukcyjności przewodu gdy szybkie przełączenie stanu wymaga przesłania określonej porcji ładunku na pojemność wejściową układu odbierającego sygnał. Takie szpilki napięciowe w przewodach masy (i zasilania) mogą powodować niepożądane przełączenia „pobliskich" układów (np. pamięci). Dlatego przewody masy wykonywane są jako maksymalnie szerokie (i grube) a kondensatory filtrujące napięcie zasilania stosowane są obficie.

Bramka Schmitta

a) symbol, b) charakterystyka

Bramka Schmitta stosowana jest np. do oczyszczania sygnałów zakłóconych i osłabionych.
Podając na wejście bramki Schmitta napięcie sinusoidalne otrzymamy na jej wyjściu przebieg prostokątny.





Układy scalone o dużej skali integracji

Procesory

CPU, DSP, Controllers

Układy pamięci

RAM, ROM, EEPROM

Układy analogowe

Mobile communication,
audio/video processing

Układy programowalne

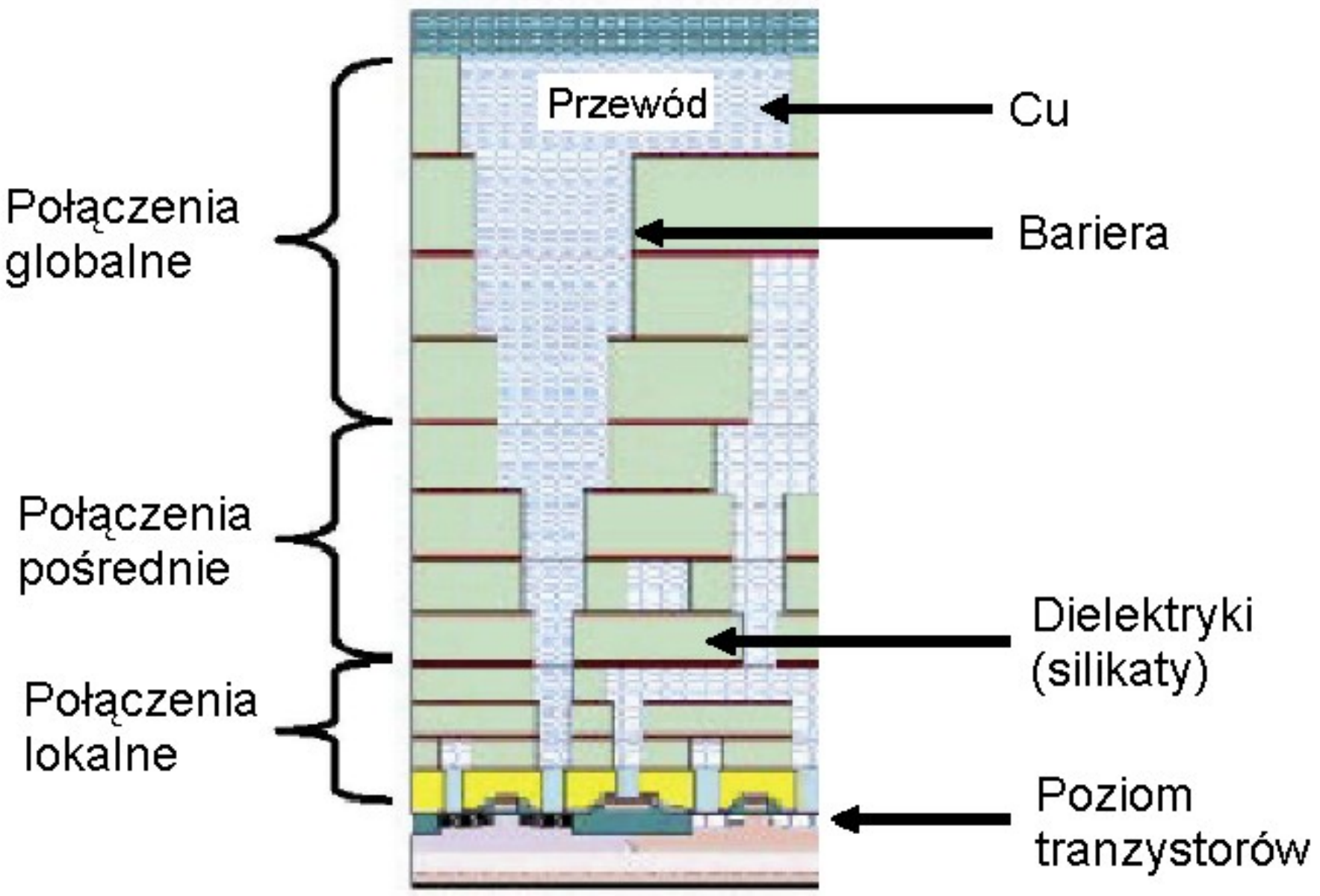
PLA, FPGA

Systemy wbudowane

Układy kontroli w samochodach, fabrykach
Network cards

System-on-chip (SoC)

Przekrój wielopoziomowej struktury układu scalonego



Systemy liczbowe i kody

Powszechnie stosowany, dziesiętny system liczbowy opiera się na zbiorze dziesięciu znaków: 0, 1, 2 ...9. W elektronice stosowane są ponadto systemy oparte na zbiorach zawierających: 2 elementy, 8 oraz 16 elementów. Zapis w tych systemach nazywamy pozycyjnym, gdyż waga cyfry zależy od jej miejsca.

Dwójkowy (binarny) system liczbowy wykorzystuje tylko dwa symbole: 0 i 1. W systemie tym podstawą jest liczba 2. Na przykład $1101_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 13_{10}$.

Poszczególne jedynki i zera nazywane są bitami (cyframi binarnymi). W systemie ósemkowym mamy 8 znaków (0,1,2 ... 7) i podstawą jest liczba 8. Szesnastkowy (heksadecymalny) system liczbowy wykorzystuje symbole: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. W systemie tym podstawą jest liczba 16, jest wygodny przy skrótowym zapisie długich ciągów cyfr (zwłaszcza binarnych). Na przykład $707_{10} = 1011000011_2 =$ (10 1100 0011 = 2 C 3) = $2C3_{16} = 2C3_H$. **Wagami** w systemie dziesiętnym są: od przecinka w lewo – 10^0, 10^1, 10^2 itd. a od przecinka w prawo – 10^{-1}, 10^{-2}, 10^{-3}. W systemie binarnym wagami są: 2^0, 2^1, 2^2, 2^3 itd. I odpowiednio 2^{-1}, 2^{-2}, 2^{-3} itd.

Przykład zamiany liczby dziesiętnej na binarną: $13_{10} = 1101_2$ bo

$$13/2 = 6 \text{ i } r_1 = 1$$

$$6/2 = 3 \text{ i } r_2 = 0$$

$$3/2 = 1 \text{ i } r_3 = 1$$

$$1/2 = 0 \text{ i } r_4 = 1$$

$$13_{10} = r_4 r_3 r_2 r_1 = 1101_2$$

Przykład zamiany liczby dziesiętnej ułamkowej na binarną.

$0.625_{10} = 0.101_2$ bo

$0.625 \times 2 = 1.25$ (całość $c_1=1$)

$0.25 \times 2 = 0.5$ ($c_2 = 0$)

$0.5 \times 2 = 1$ ($c_3 = 1$)

$0.625_{10} = 0.c_1c_2c_3 = 0.101$

System binarny wystarcza aby w układach cyfrowych i komputerach zapisywać wszelaką informację (liczby, słowa, instrukcje itp.)

KODY

Kodem nazywamy zbiór symboli razem z zasadami stosowania.

W elektronice funkcjonuje wiele kodów, poniżej podamy tylko kilka z nich.

Kody BCD (*binary coded decimal*). Te kody kodują każdą cyfrę liczby dziesiętnej osobną czwórką bitów. W zwykłym kodzie BCD mamy wagi 8421 i na przykład $1998_{10} = (1\ 9\ 9\ 8) = 0001\ 1001\ 1001\ 1000_{(BCD)}$. Inne kody BCD to: BCD Aikena o wagach 2421, BCD z nadmiarem 3 (do każdej cyfry dodajemy +3 np. $10_{10} = 0100\ 0011_{BCD\ EX\text{-}3}$)

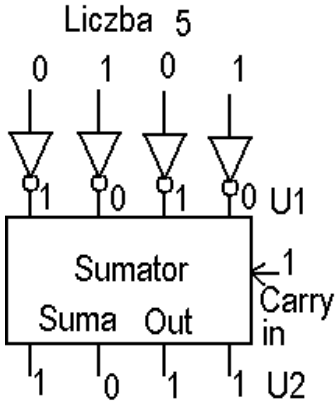
Należy zauważyć, że notacje BCD nie są identyczne z zapisem binarnym. Kod BCD wykorzystywany jest w układach z wyświetlaczami cyfr dziesiętnych.

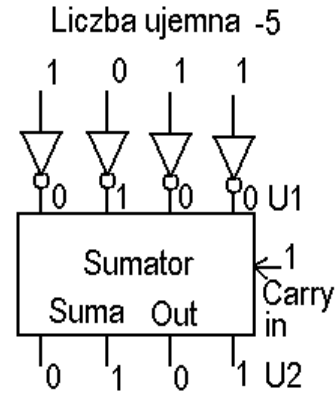
Kod z nadmiarem 3

0	0011	5	1000
1	0100	6	1001
2	0101	7	1010
3	0110	8	1011
4	0111	9	1100

Porównanie kodów: znak-moduł, binarny-przesunięty, U1 i U2.

Liczba całkowita	Znak-moduł	Binarny przesunięty	Z uzupełnieniem do 1 (U1)	Z uzupełnieniem do 2 (U2)
+7	0111	1111	0111	0111
+6	0110	1110	0110	0110
+5	0101	1101	0101	0101
+4	0100	1100	0100	0100
+3	0011	1011	0011	0011
+2	0010	1010	0010	0010
+1	0001	1001	0001	0001
+0	0000	1000	0000	0000
-1	1001	0111	1110	1111
-2	1010	0110	1101	1110
-3	1011	0101	1100	1101
-4	1100	0100	1011	1100
-5	1101	0011	1010	1011
-6	1110	0010	1001	1010
-7	1111	0001	1000	1001
-8	—-	0000	—-	1000
(-0)	1000	—	1111	—-





Przykład:

a) Wykonać odejmowanie liczb: 0010 – 0010 (czyli 2 - 2) stosując kod U2.

Rozw. 0010 – 0010 = **0**0010 + [negacja z **0**0010 +1] =

00010 + [**1**1101 +1] = **0**0010 + **1**1110 = 1**0**0000 = **0**0000

b) Wykonać odejmowanie liczb: 0010 – 0100 (czyli 2 - 4) stosując kod U2.

Rozw. 0010 – 0100 = **0**0010 + [negacja z **0**0100 +1] =

00010 + [**1**1011 +1] = **0**0010 + **1**1100 = **1**1110

c) Wykonać odejmowanie liczb: 0110 – 0100 (czyli 6 - 4) stosując kod U2.

Rozw. 0110 – 0100 = **0**0110 + [negacja z **0**0100 +1] =

00110 + [**1**1011 +1] = **0**0110 + **1**1100 = 1**0**0010 = **0**0010

Kod Graya jest kodem o wzmocnionej odporności na powstawanie błędów transmisji. Wynika to z faktu, iż w tym kodzie sąsiednie liczby różnią się tylko jednym bitem. Kod Graya stosowany jest gdy kodowany jest sygnał analogowy, nie skokowy, np. przy kodowaniu kąta obrotu wału: kąt-liczba. Wartość zero reprezentuje tu układ zer $0_{10} = 0000$, aby uzyskać każdą następną wartość, zmieniamy zawsze jeden, możliwie najbardziej na prawo stojący bit, którego zmiana daje nowy (dotąd nie wykorzystany układ). Czyli: $1_{10} = 0001$, $2_{10} = 0011$, $3_{10} = 0010$, $4_{10} = 0110$, $5_{10} = 0111$ itd. Kod Graya jest tzw. kodem niewagowym tj. położenie znaku (w przeciwieństwie do np. kodu binarnego) nie oznacza wagi (czyli potęgi liczby 2)[1].

Wśród innych kodów o wzmocnionej odporności na błędy można wymienić kody ze **stałą liczbą jedynek** oraz z tzw. **bitem parzystości**.

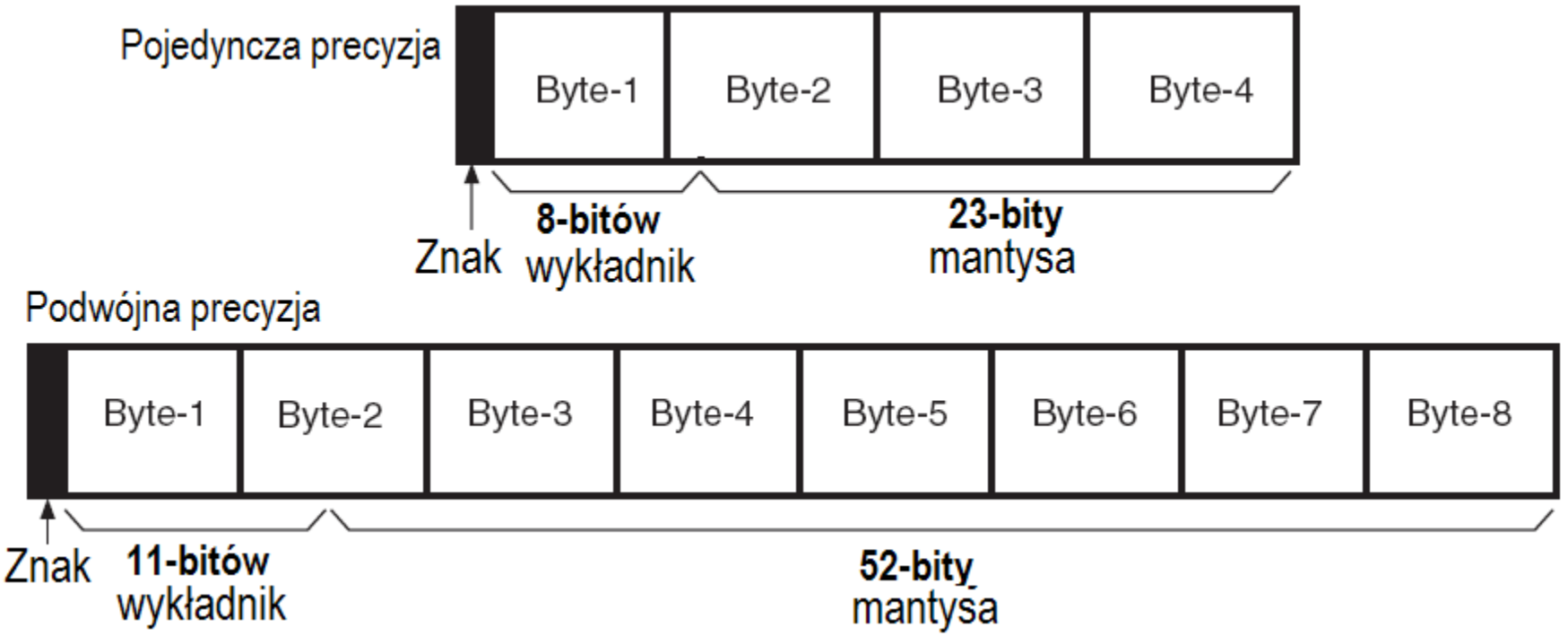
---

[1] Matematycy odkryli wieloznakowe kody Graya, np. kod 3-znakowy złożone ze znaków: 0, 1 i 2 wyglada dla kodu dwubitowego nastepująco: 00, 01, 02, 12, 11, 10, 20, 21, 22.

Formaty liczb binarnych zmiennopozycyjnych

(Floating point standard IEEE-P754)

[Znak: 1 bit][(Wykładnik z przesunięciem: 8, 11, 15 lub więcej bitów] [Ukryta jedynka mantysy: 0 bitów] [Mantysa: 23, 52, 63 lub więcej bitów]. Mantysa ma wartość od 1 do 2 ale zapisywana jest bez pierwszej (oczywistej) jedynki.



Bit znaku 0-liczba dodatnia, 1-liczba ujemna. Wykładnik: 01111111 (127) oznacza, że wykładnik = 0, poniżej wartości (127) mamy wykładniki ujemne a powyżej (127) dodatnie.

Przykłady:

-1.11_2 ---> 1 01111111 11000000000000000000000
(127+0)

$+1101.101_2$ ---> 0 10000010 10110100000000000000000
(127+3)

-0.001011 ---> 1 01111100 01100000000000000000000
(127-3)

(0 zapisane jako ciąg 0000..... jest niestety liczbą = $1x2^{-127}$)

Przykład. Przedstawić rprezentację zmiennopozycyjną liczby $(-142)_{10}$.

Rozwiazanie. Binarny równoważnik wartości bezwzględnej wynosi: $(142)_{10} = (10001110)_2$.

$(10001110)_2 = 1.0001110 \times 2^7 = 1.0001110 \times 2^{+0111}$.

Mantysa = 0001110 00000000 00000000 (bez oczywistej jedynki!).

Wykładnik bez przesunięcia = 00000111.

Wykładnik z przesunięciem = 00000111 + 01111111 = 10000110.

Znak liczby: 1. Zatem $(-142)_{10} = (1_{znak}\ 10000110_{wyk.}\ 0001110\ 00000000\ 00000000_{mant})$ = 11000011 00001110 00000000 00000000.

Przykład. Zamienić na postać dziesiętną liczbę zmiennopozycyjną:

00111111 01000000 00000000 00000000.

Rozwiązaie. Znak liczby jest dodatni (bo pierwsze jest „0”). Wykładnik z przesunięciem: 01111110 (czyli -1 bo brakuje 1 do 01111111). A formalnie bez przesunięcia: $01111110 - 01111111 = (01111110 + 10000001_{U2}) = 11111111_{U2} = -1$

Zatem wykładnik = -1. Bitami mantysy są (po dodaniu oczywistej 1-ki):

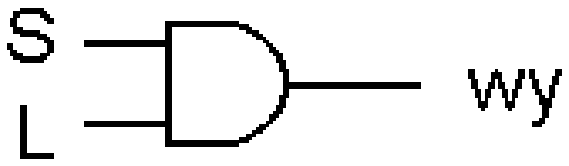
11000000 00000000 00000000. Mantysa = 1.1000000 00000000 00000000.

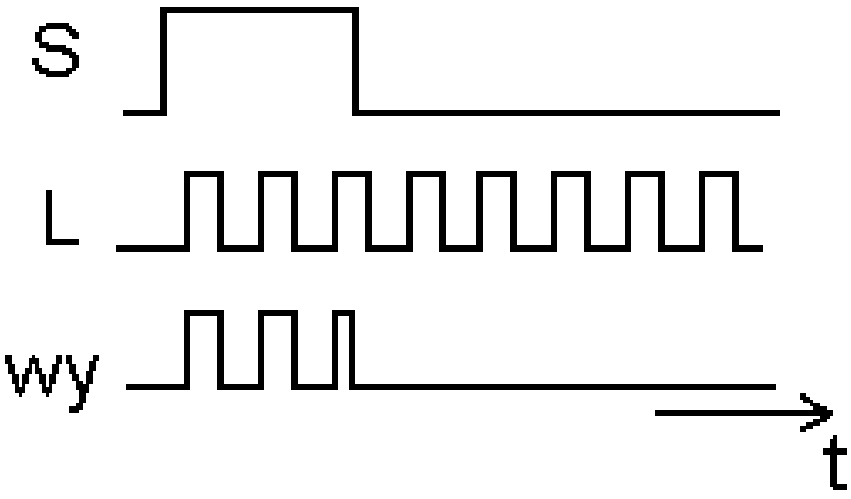
$=(1,1)_2 \times 2^{-1} = (0,11)_2 = (0,75)_{10}$.

Układy kombinacyjne to takie układy, w których stan wyjścia zależy od aktualnej kombinacji stanów wejściowych.

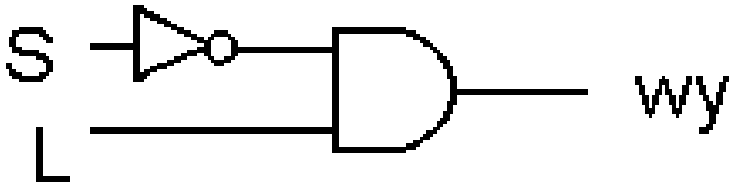
Proste układy z bramkami cyfrowymi.

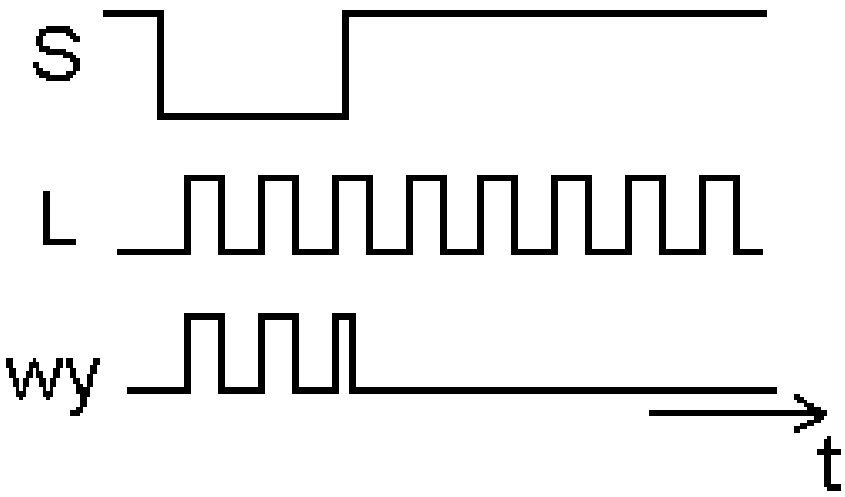
Układ koincydencyjny



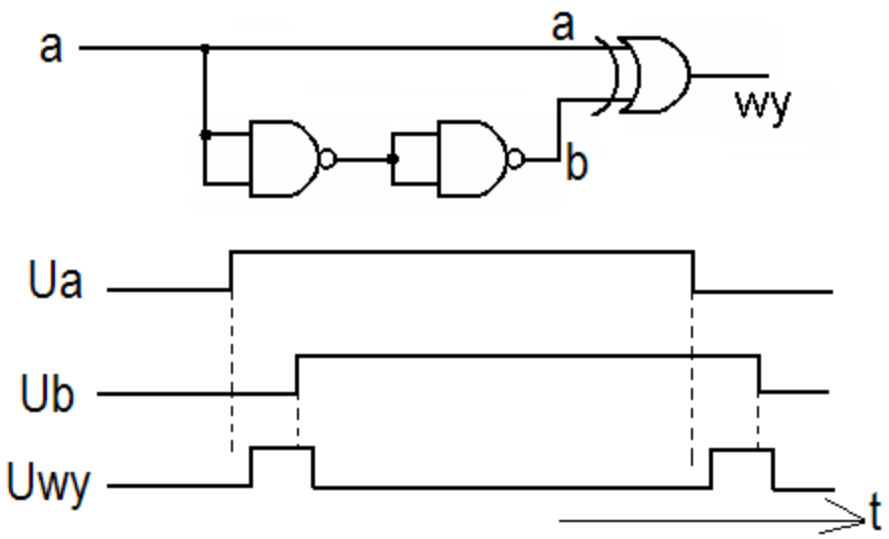


Układ antykoincydencyjny

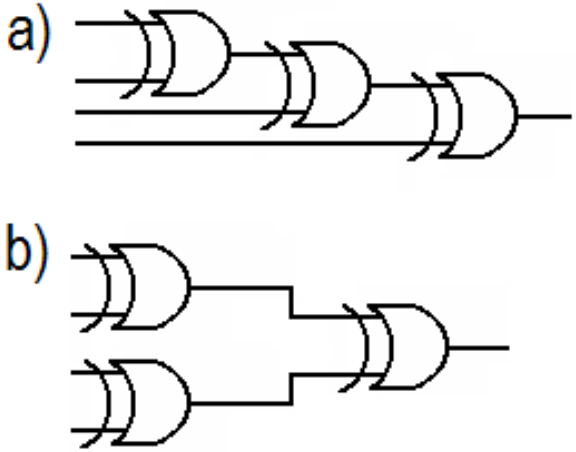




Efektem różnych czasów propagacji wzdłuż różnych ścieżek sygnału może być generowanie wąskich impulsów czasem zamierzone i pożądane a czasem szkodliwe.

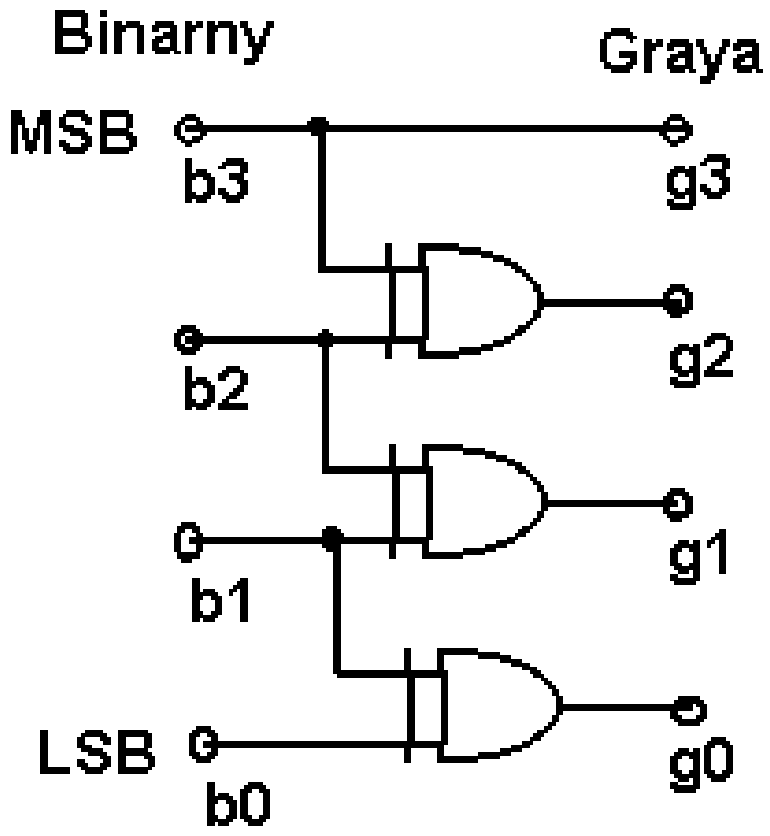


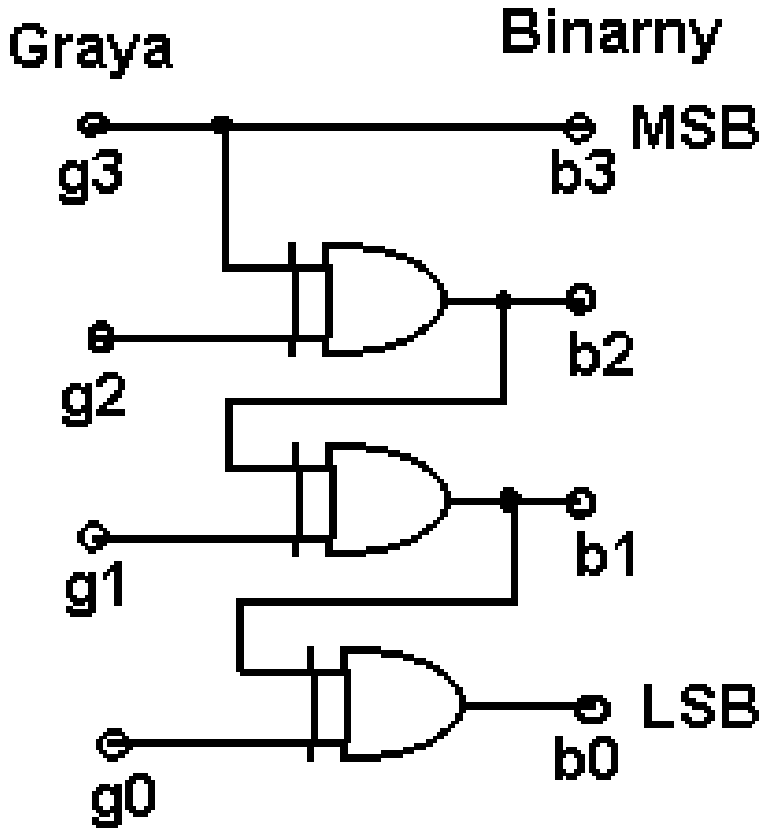
Z dwóch pokazanych na rysunku układów do generowania bitu parzystości lepszy jest wariant „b”, w którym czas ustalania stanu wyjściowego jest o 1/3 krótszy od czasu ustalania stanu w wariancie „a”



Przykłady:

Układ zamiany kodu binarnego na kod Graya i układ zamiany kodu Graya na binarny.

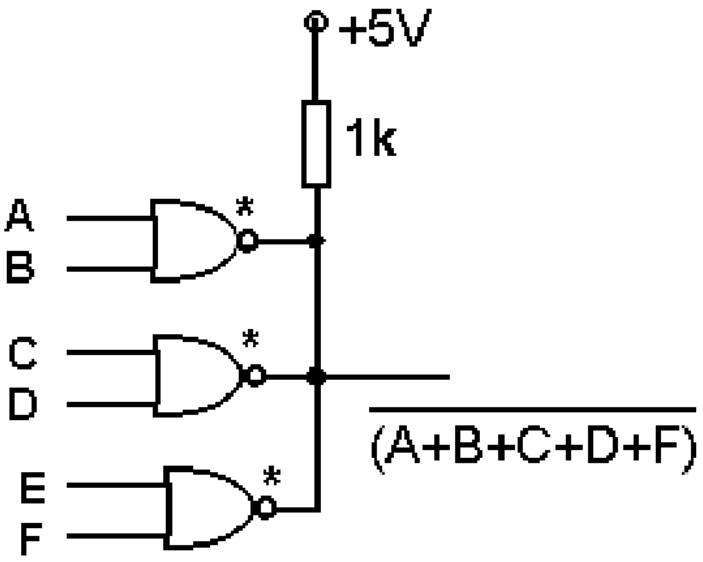




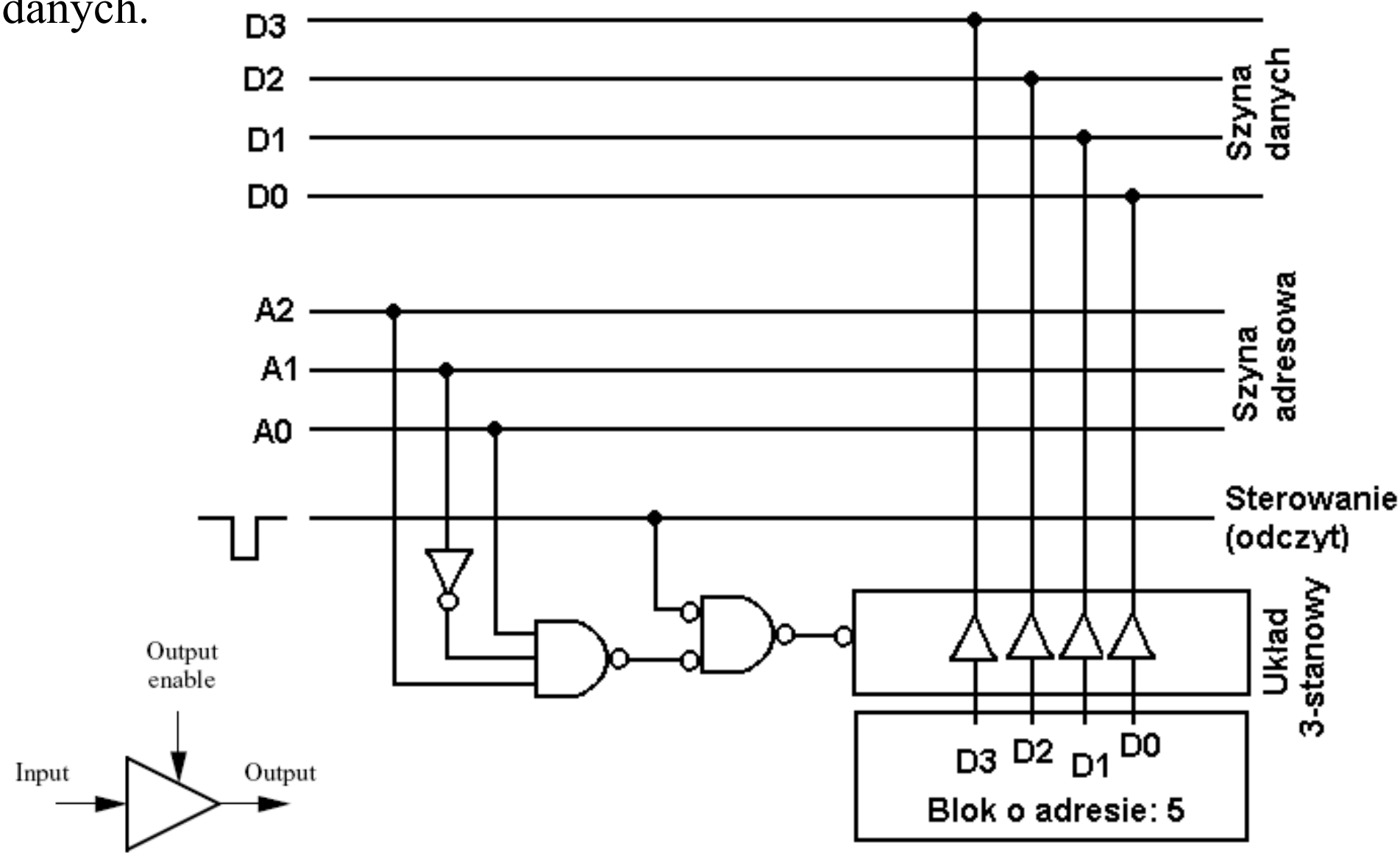
Bramki z otwartym kolektorem (OC) na wyjściu.

Bramka OC przejawia aktywność gdy na jej wyjściu ma być stan niski bo tylko wtedy zwiera ona kolektor (dren) wyjściowego tranzystora do masy. Znak gwiazdki przy symbolu bramki oznacza bramkę typu OC.

Wyjścia bramek z otwartym kolektorem mogą być i są łączone ze sobą bezkonfliktowo. Wyjścia bramek - kolektory połączone do jednej linii zasilanej przez opornik stanowią przewodowe LUB (ang. Wired-OR) czyli tzw. sumę montażową. Pojawienie się na tej linii stanu niskiego oznacza, że co najmniej jeden (LUB więcej) kolektorów zwiera tę linię do masy! W układzie pokazanym na rysunku, stan wysoki na wyjściu oznacza, że na wszystkich wejściach od A do F są stany niskie. Układ ten służy do sygnalizacji, że co najmniej jedno urządzenie chce na siebie zwrócić uwagę. Zastosowania: linia przerwań w komputerze, magistrale na zewnątrz komputera np. interfejs IEC 625, (w USA IEEE-488, znany też jako HPIB lub GPIB).



Układy z trzema stanami wyjściowymi (HIGH, LOW, Odłączony - tj. stan wysokiej impedancji) są konieczne w rozbudowanych układach z 3-magistralową architekturą. Przykład: fragment szyny do przekazu danych.



Dekodery i Kodery

Dekoderem nazywa się element, którego wektor wejściowy ma n współrzędnych, a wektor wyjściowy ma $k = 2^n$ współrzędnych, przy czym dana współrzędna wektora wyjściowego może okazać się stanem aktywnym (1 lub 0) dla tylko jednego wektora wejściowego (jednej kombinacji zer i jedynek). Jeżeli $k = 2^n$ to dekoder nazywa się dekoderem pełnym, jeżeli $k < 2^n$ to mamy dekoder niepełny.

Koder

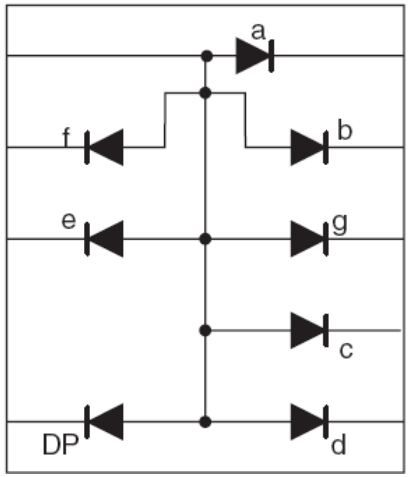
Koderem nazywa się element, którego wektor wejściowy ma $k = 2^n$ współrzędnych, a wektor wyjściowy ma n współrzędnych i jest kodem numeru tego (jedynego) wejścia, na które wprowadzono wyróżniony sygnał (sygnał aktywności, stan wysoki - w logice dodatniej, stan niski - w logice ujemnej).

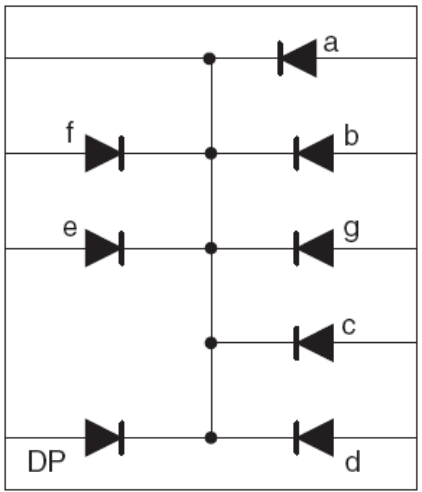
Koder priorytetu

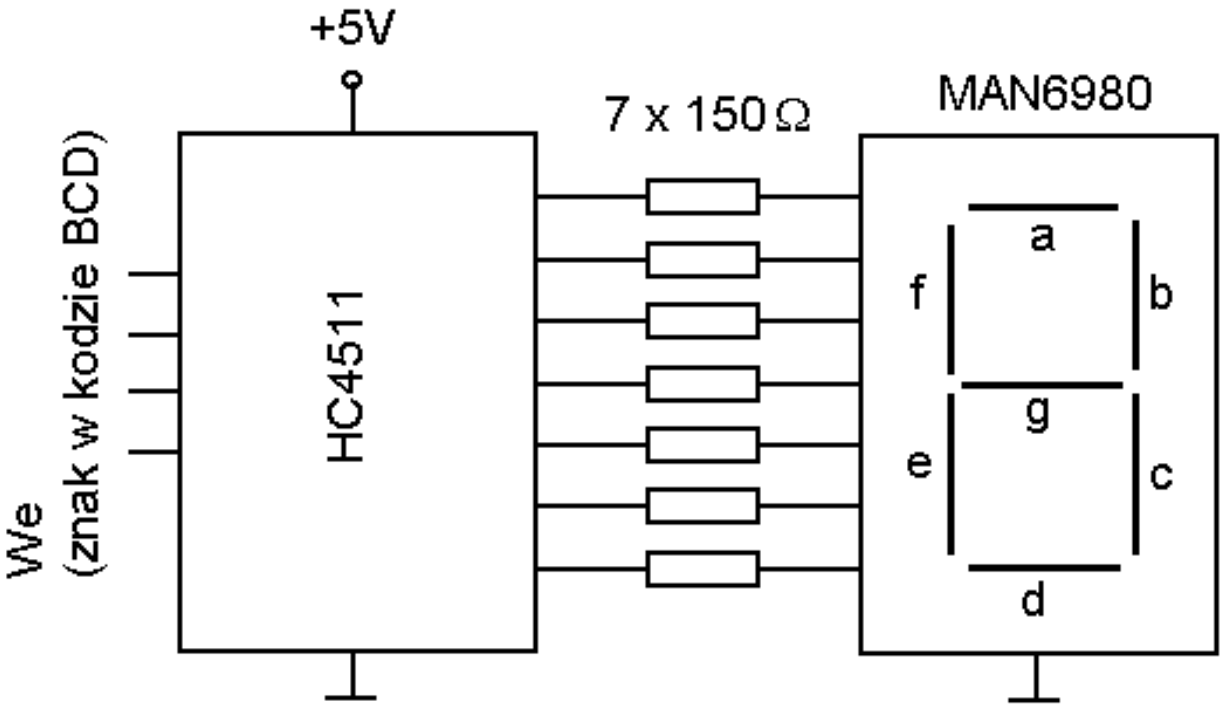
Koderem priorytetu nazywamy koder, którego wektor wyjścia jest zawsze kodem najwyższego numeru wejścia spośród wszystkich wejść, na które podano wyróżniony sygnał aktywności (jedynkę logiczną).

Przykład dekodera

Dekoder HC4511 jest dekoderem kodu BCD przeznaczonym do sterowania jednocyfrowym wyświetlaczem 7-segmentowym LED ze wspólną katodą. Wewnątrz układu dodatkowo (oprócz dekodera) znajduje się rejestr zatrzaskowy (pamięć) i wzmacniacze do sterowania segmentami, których stopnie wyjściowe mogą generować prądy do 15mA przy napięciach wyjściowych 4,5V.







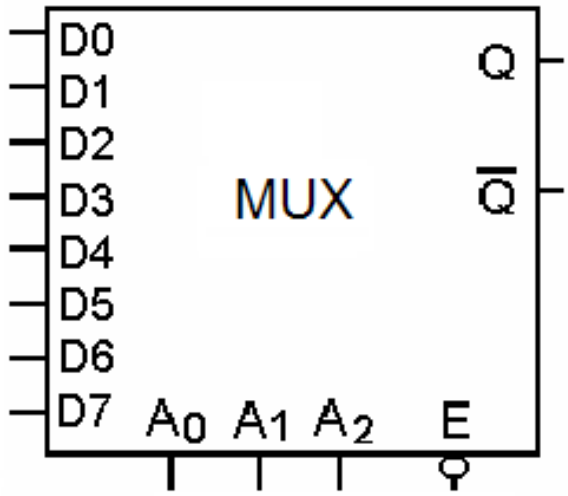
Multipleksery i demultipleksery

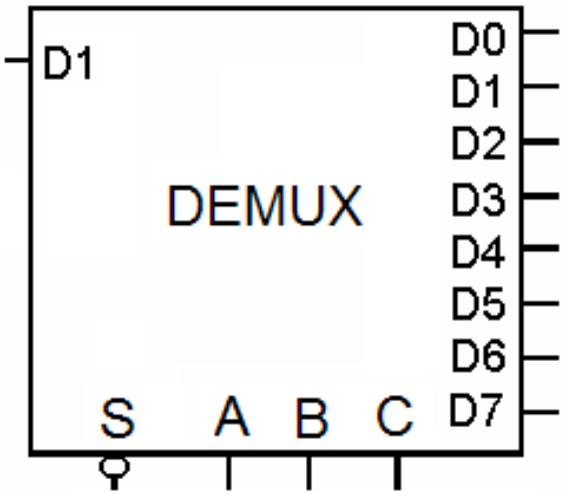
Multipleksery i demultipleksery zaliczane są do takich układów kombinacyjnych, które umożliwiają komutację (tj. przełączanie) sygnałów cyfrowych. Multipleksery są to układy pozwalające na skierowanie informacji z wielu wejść na jedno wyjście. Wyjście jest połączone (sterowane) tym wejściem, które wybieramy przy pomocy wejść adresowych. Demultipleksery realizują funkcję odwrotną tj. sygnał z jedynego wejścia kierują na „zaadresowane" jedno z wielu wyjść.

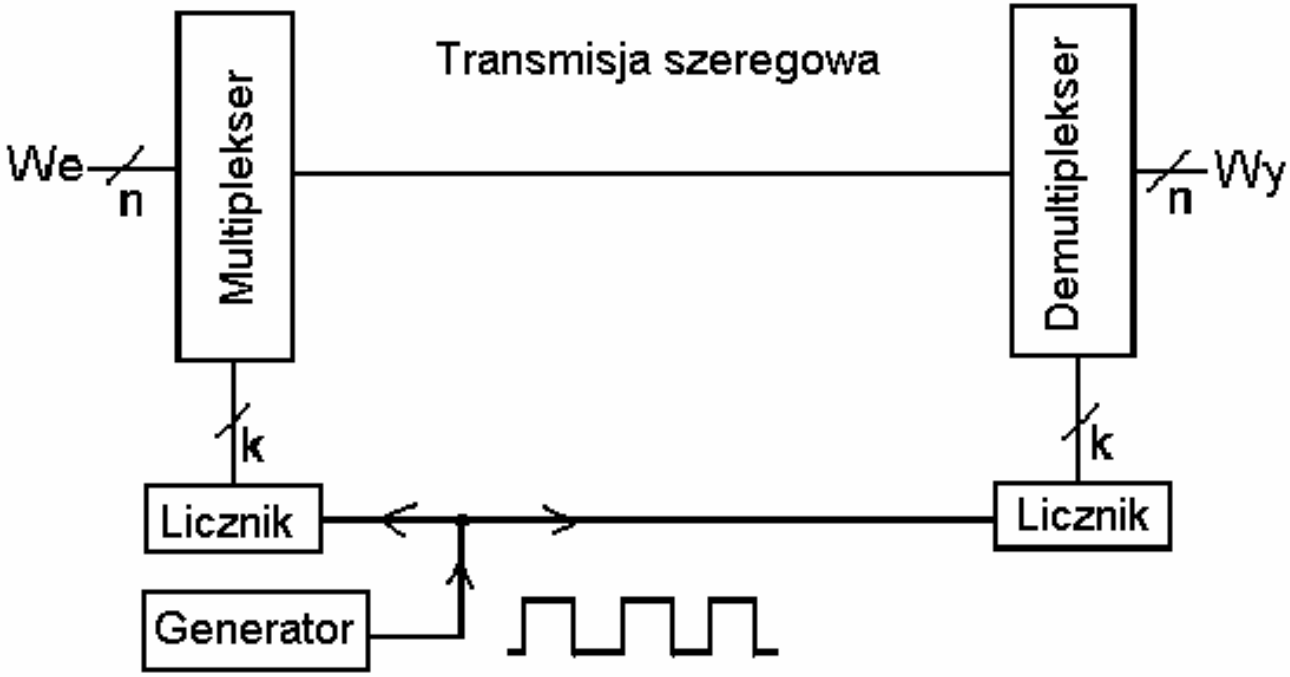
Multipleksery podobnie jak i demultipleksery mogą być ze sobą łączone dając możliwość zwiększenia liczby przełączanych linii.

Multipleksery stosowane są np. na wejścia przetworników analogowo-cyfrowych (AD). Multipleksery i demultipleksery mogą realizować multipleksowany system przesyłania danych, mogą też być stosowane do realizacji innych układów kombinacyjnych realizujących złożone funkcje np. linijka świetlna.

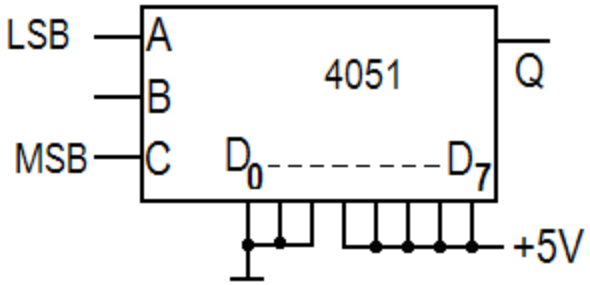
Na rysunku zamieszczono przykład multipleksera i demultipleksera oraz uproszczony układ zamiany transmisji równoległej na szeregową i ponownego powrotu do transmisji równoległej (związek między n i k: $n = 2^k$) . Symbole D i Q oznaczają linie danych, A,B i C – linie adresowe, S – Strobe, E – enable,





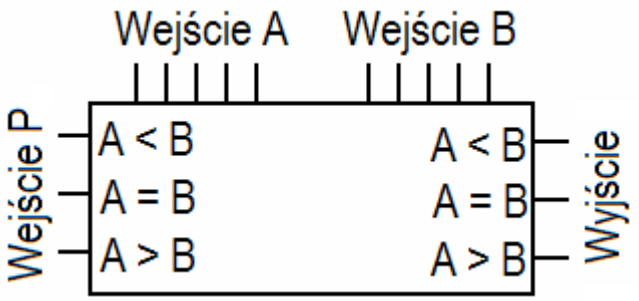


Realizacja tabeli prawdy przy pomocy multipleksera. Układ obok wyróżnia liczby większe od 2 podawane na 3-bitowe wejście ABC.



Komparator cyfrowy

Dodatkowe wejście porównania (wejście P.) umożliwia porównywanie większych liczb A i B.

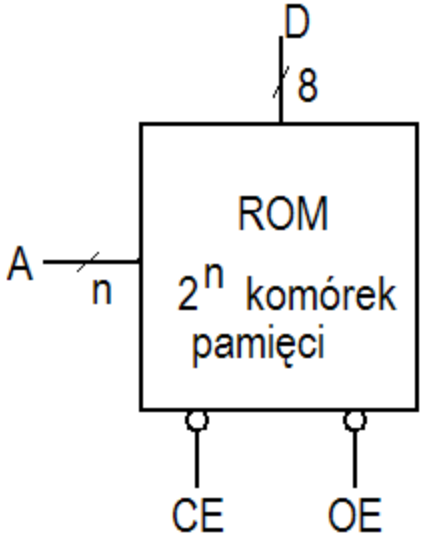


Pamięć ROM jako przykład układu kombinacyjnego

Układy pamięci ROM (read-only-memory) będąc w zasadzie układem z pamięcią po jednorazowym zaprogramowaniu staje się układem kombinacyjnym.

Przykładowo na rysunku obok

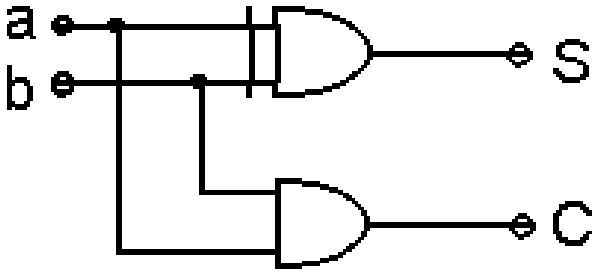
n wejść adresowych A pozwala na zaadresowanie 2^n komórek pamięci. Zawartość zaadresowanej 8 – bitowej komórki może być wystawiona na 8 wyjściach danych D w momencie gdy na wejściach CE i OE pojawią się stany niskie. Zatem na wyjściu 8 – bitowym D pojawia się zestaw stanów jako funkcja stanów na wejściach CE, OE i A,

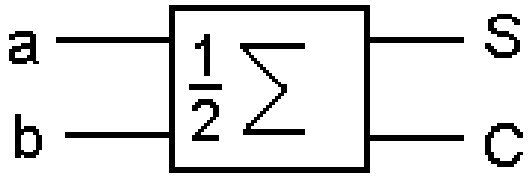


Sumatory

Sumatory są układami dodającymi dwie liczby binarne. Najprostszymi i elementarnymi są te, które dodają dwie liczby jednobitowe. Półsumator może dodawać dwa najmłodsze bity liczb. Bit przeniesienia występuje tu tylko na jednym z wyjść (oznaczonym przez C).

Schemat i symbol półsumatora.

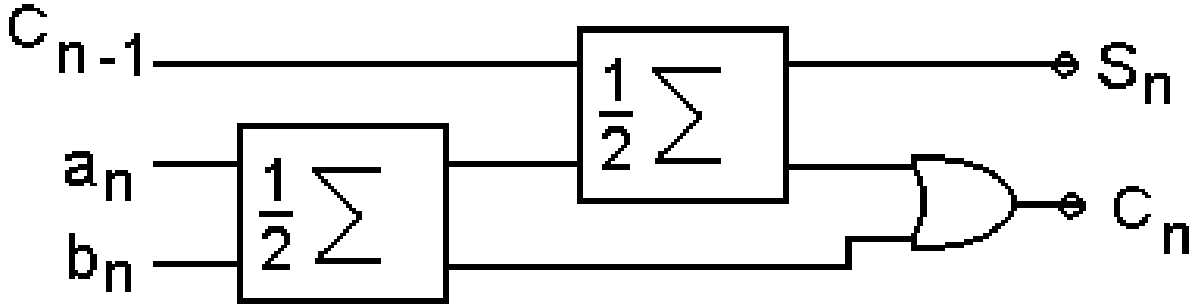


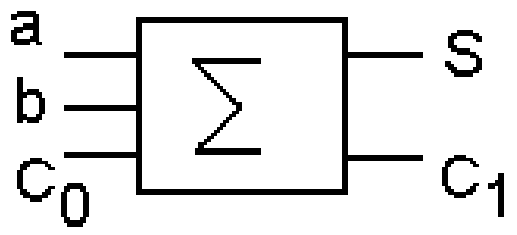


Sumator

Pełny sumator może dodawać dowolnie usytuowane części liczb, gdyż dodaje również bit przeniesienia z młodszej części liczb.

Schemat i symbol sumatora

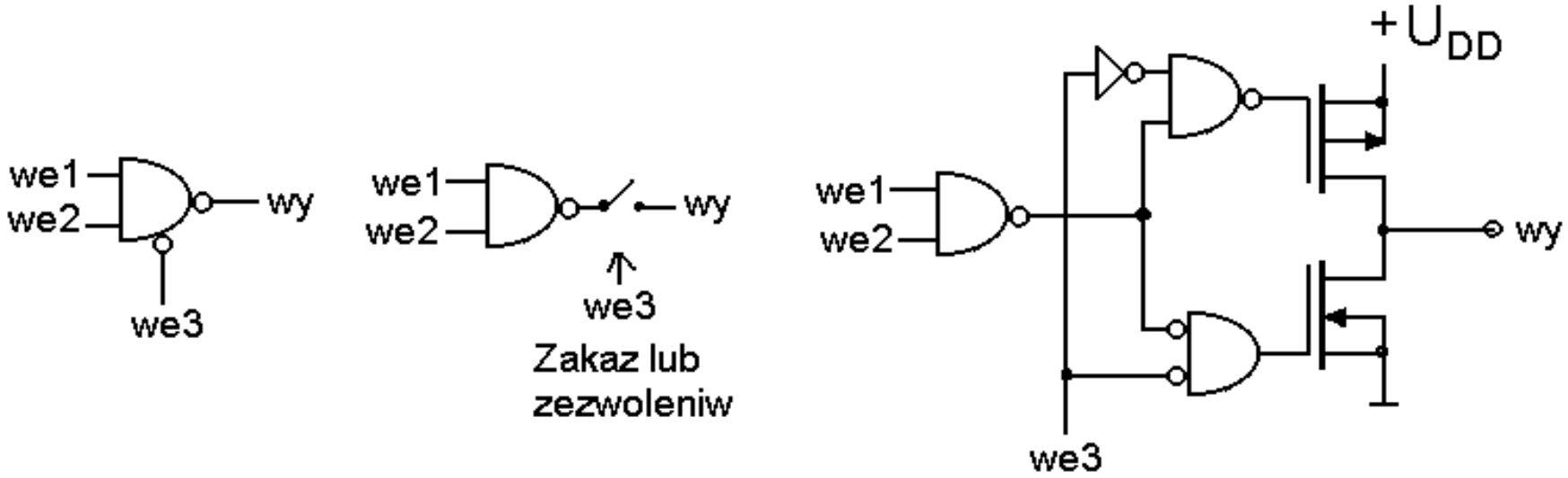




Układy trójstanowe (logika trójstanowa)

W elektronice cyfrowej często spotykamy sytuacje (np. w systemach komputerowych), w których wiele bloków musi wymieniać dane wykorzystując jedną wspólną szynę. Układy z wyjściami dwustanowymi nie mogą być podłączone bezpośrednio do takiej szyny "bezkonfliktowo" (nie można uniknąć zdarzeń gdy na jednym przewodzie część bloków próbuje wymusić stan wysoki a inna część bloków stan niski!). Rozwiązaniem problemu jest zastosowanie układów trójstanowych. Przykład bramki trójstanowej NAND CMOS:

Symbol Zasada działania Realizacja



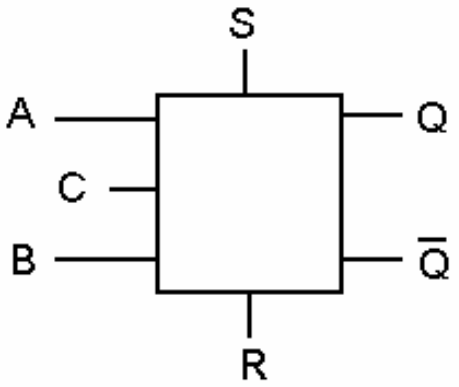
Układy sekwencyjne W tych układach stan wyjścia zależ nie tylko od aktualnej kombinacji stanów wejściowych ale również od wcześniejszych kombinacji (od historii) czyli są to takie układy, które mogą pamiętać.

Przerzutniki bistabilne.

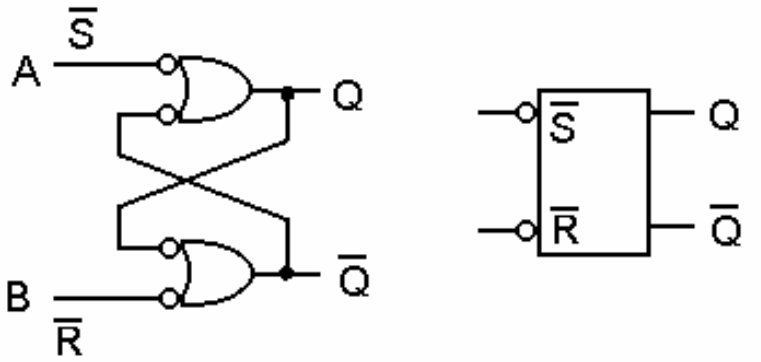
Stanowią osobna grupę układów cyfrowych i są najprostszymi elementami pamięci (układy sekwencyjne). Mogą pamiętać jeden bit informacji. Przerzutniki mają po dwa wyjścia Q i Q^*. Na wyjściu Q^* pojawia się zawsze stan przeciwny do stanu na wyjściu Q. Poza tym przerzutniki mają dwa wejścia asynchroniczne: jedno ustawiające S (set) i jedno kasujące R (reset), jedno wejście zegarowe (taktujące) C i zwykle dwa wejścia informacyjne A i B.

Wymuszanie stanów logicznych na wyjściach za pomocą wejść S i R charakteryzuje się najwyższym priorytetem: zachodzi niezależnie od sytuacji na innych wejściach. Natomiast gdy na wejściach S i R są zera logiczne, stan wyjściowy przerzutnika określany jest przez wejścia A i B ale dopiero po pojawieniu się jedynki logicznej na wejściu C jako odpowiedniego impulsu zegara. Należy podkreślić, że rozmaite przerzutniki reagują na różne zbocza tego impulsu: zbocze narastające lub opadające.

Ogólny schemat

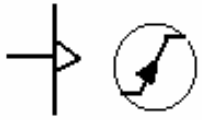


Przerzut RS i jego symbol

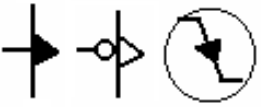


Przejście wymuszane zboczem:

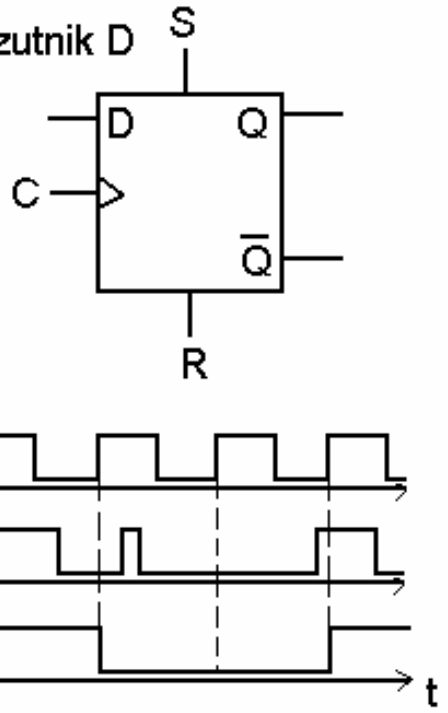
narastającym,



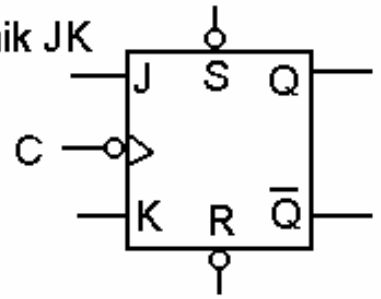
opadającym



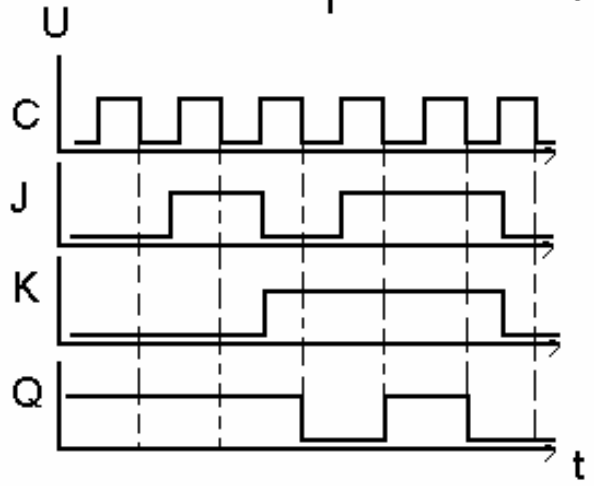
Przerzutnik D



Przerzutnik JK



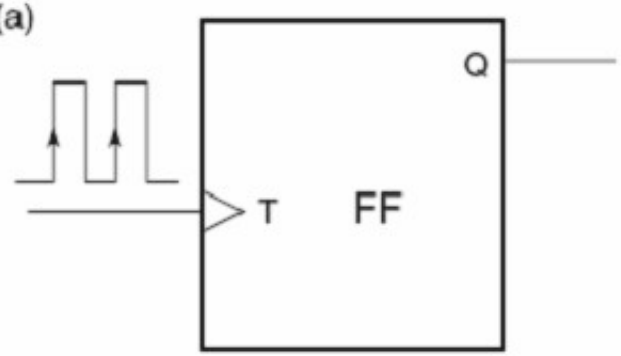
J K	Q_{n+1}
00	Q_n
01	0
10	1
11	$\overline{Q}_n$



Przerzutniki (Flip-Flop: FF) typu T (Taggle)

Tu aktywne zbocze zawsze „przestawia” stan wyjściowy na przeciwny: Q_{n+1} = Negacja Q_n

(a)

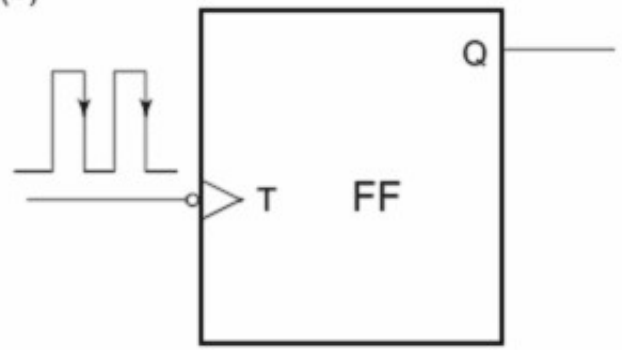


T	Q_n	Q_{n+1}
↑	0	1
↑	1	0

Q_n	T	Q_{n+1}
0	0	0
0	1	1
1	0	1
1	1	0

$$Q_{n+1} = T.\overline{Q_n} + \overline{T}.Q_n$$

(b)



T	Q_n	Q_{n+1}
↓	0	1
↓	1	0

Q_n	T	Q_{n+1}
0	0	1
0	1	0
1	0	0
1	1	1

$$Q_{n+1} = \overline{T}.\overline{Q_n} + T.Q_n$$

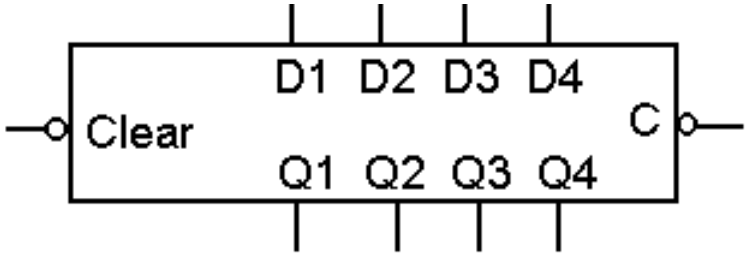
Rejestry

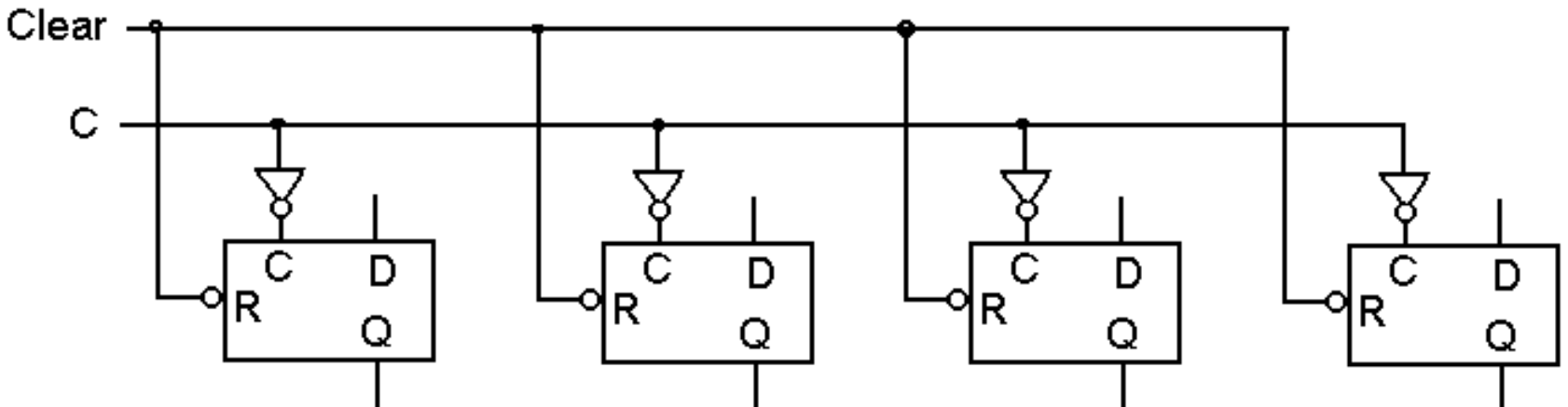
Rejestry należą do układów sekwencyjnych (pamiętających)

Podstawowym przykładem rejestru jest rejestr buforowy.

Rejestr buforowy (w skrócie rejestr) jest zespołem przerzutników synchronicznych o wspólnym wejściu taktującym i wspólnym wejściu zerującym, przeznaczony jest do chwilowego przechowania wektora informacji. Wprowadzanie wektora informacji odbywa się równolegle (wszystkie bity składowe jednocześnie). Wszystkie bity wektora informacji są dostępne jednocześnie i mogą być odczytane równolegle.

Przykład 4-bitowego rejestru buforowego (i jego schemat).

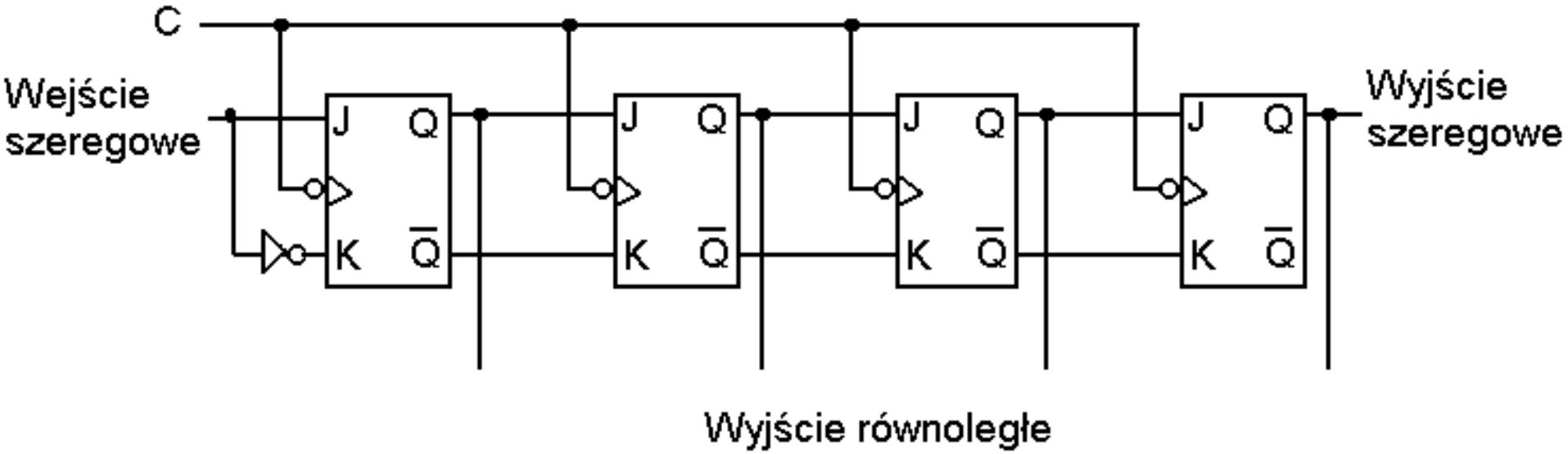




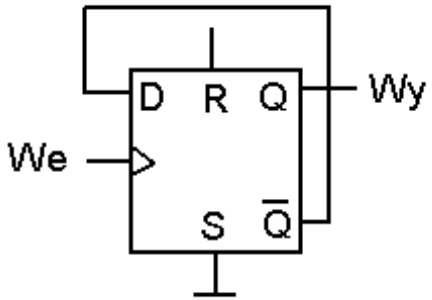
Rejestr przesuwający

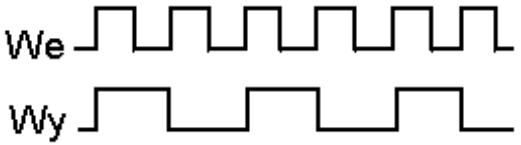
Innym typem rejestru jest rejestr przesuwający.

Jest nim zespół przerzutników synchronicznych, umożliwiający wprowadzanie i wyprowadzanie wektorów informacji cyfrowej w sposób bitowo-szeregowy w czasie. Pokazuje to rysunek 4-bitowego rejestru przesuwającego:

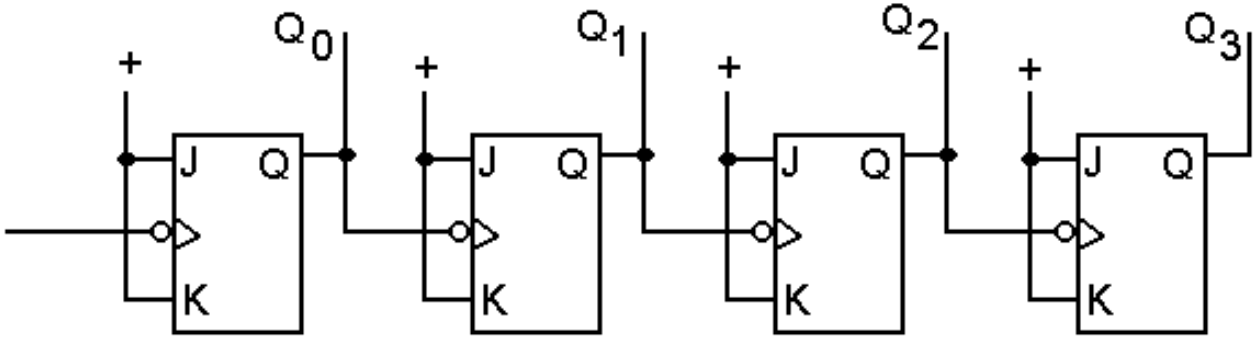


Liczniki. Licznikiem nazywa się rejestr, którego stan jest kodem numeru impulsu wprowadzonego na jego wejście liczące (licznik zaczyna pracę od wyróżnionego stanu początkowego a całkowita liczba impulsów wprowadzonych nie przekracza pojemności licznika). Na rysunku pokazano elementarne liczniki na przerzutnikach D oraz JK. Przerzutniki mogą i często są dzielnikami częstotliwości przez 2. Połączenie szeregowe n takich jednostek elementarnych daje licznik zliczający w kodzie dwójkowym o pojemności 2^n. Jako przykład, na poniższym rysunku, przedstawiony jest licznik 4-bitowy (dzielnik przez 16). Zliczane impulsy podawane są na wejście zegarowe.

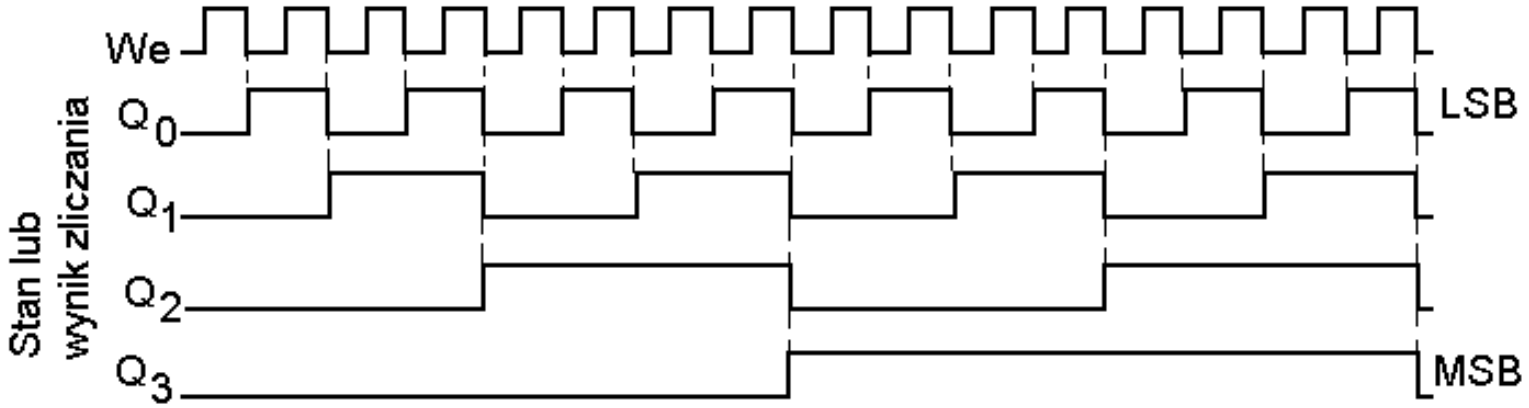




Schemat:



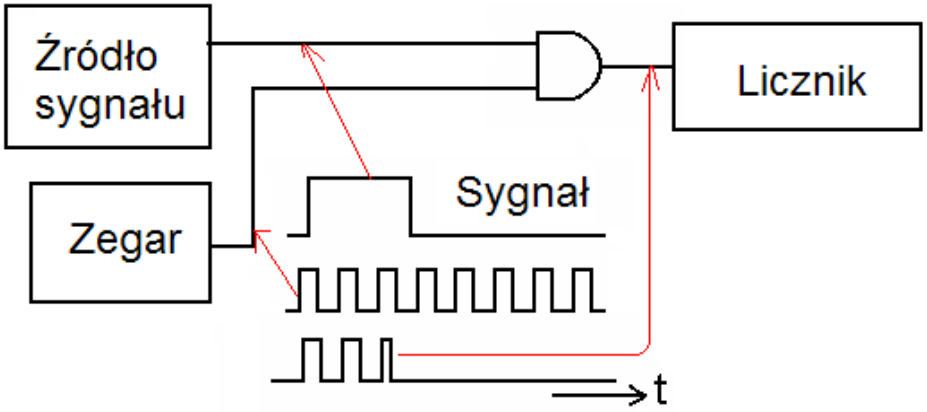
i przebiegi czasowe



Zastosowania Liczników (czasomierzy)

Pomiar czasu trwania impulsu

Licznik przed pomiarem jest wyzerowany. Badany impuls jest tu użyty jako impuls bramkujący licznik przy zliczaniu cykli sygnału zegara. Czas trwania impulsu T_i jest dany przez: $T_i = N/f_z$, gdzie N – liczba zliczeń, f_z – częstotliwość zegara.



Pomiar odstępu czasu między dwoma zdarzeniami

W tym przypadku pierwsze zdarzenie włącza początek impulsu bramkującego a zdarzenie drugie kończy ten impuls.

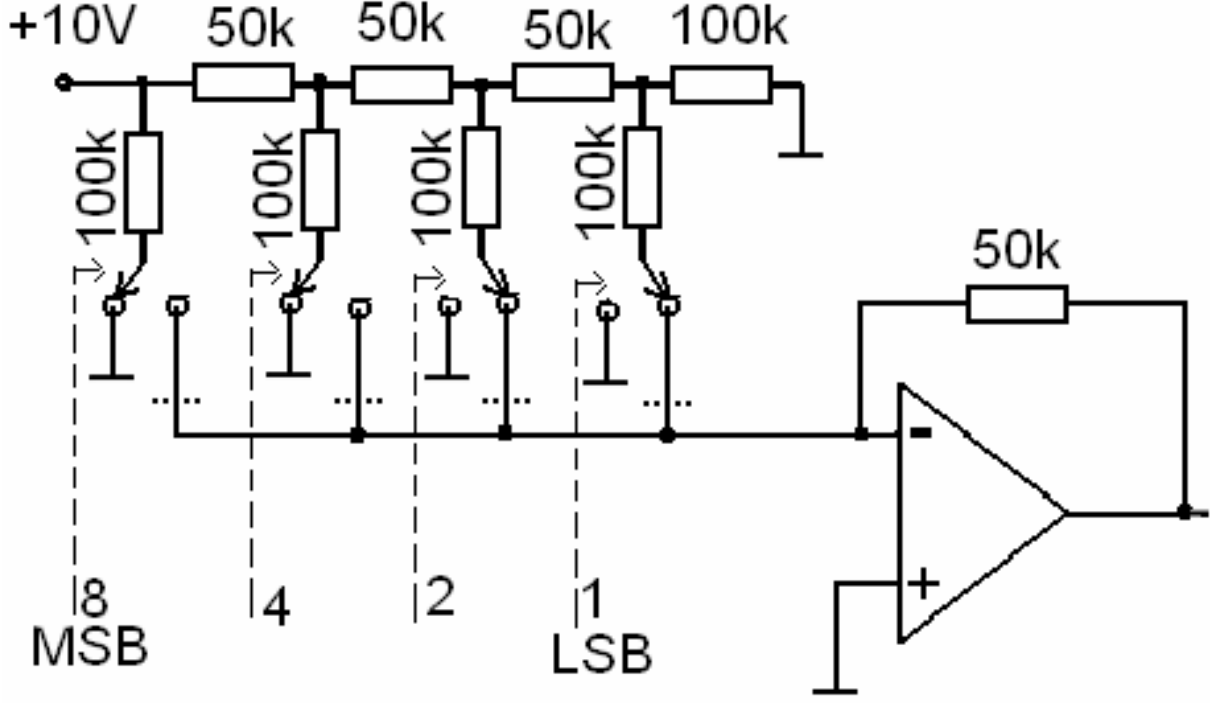
Generowanie impulsu o określonej długości (czasowej).

Licznik ustawiany jest na N zliczeń np. po załadowaniu liczby N liczy w dół impulsy zegara aż do zera. Sygnał wyjściowy jest wysoki w czasie liczenia i niski po pojawieniu się zera. Czas trwania impulsu T_i jest dany przez: $T_i = N/f_z$, gdzie N – liczba zliczeń, f_z – częstotliwość zegara.

Przetworniki D/A

Zadaniem przetworników cyfrowo analogowych (DAC) jest zamiana liczb (w kodzie binarnym) na napięcia proporcjonalne do wartości tych liczb.

Na rys. pokazano ideę jednego z wielu typów przetworników.



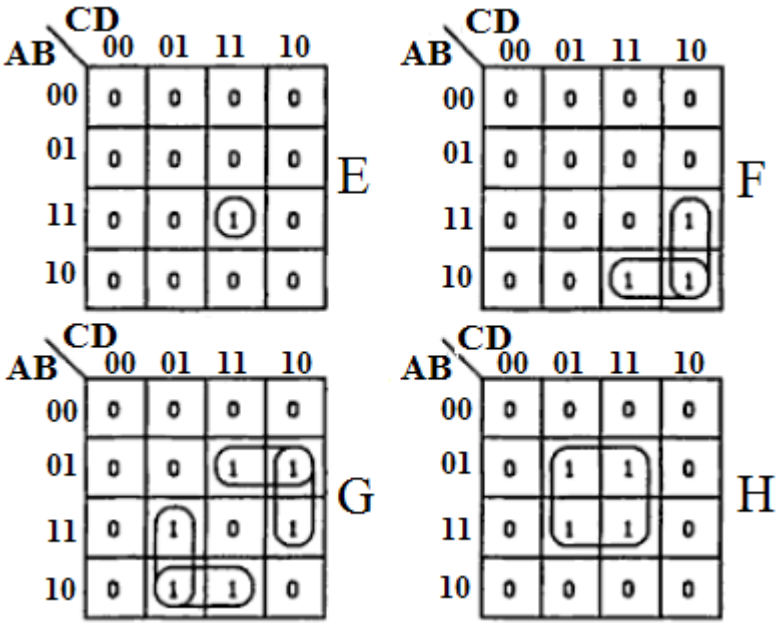
Jest to tzw. drabinka R-2R. Stany 1 i 0 na poszczególnych liniach szyny (tu 4-bitowej) decydują o włączeniu bądź nie, odpowiedniego przełącznika. Przez rezystory 100k płyną stałe prądy (niezależnie od położenia przełączników) o wartościach proporcjonalnych do wagi poszczególnych bitów. Suma tych prądów, które są włączone do wejścia wzmacniacza operacyjnego oczywiście musi przepływać przez opornik 50k nad wzmacniaczem i na wyjściu wzmacniacza mamy już napięcie proporcjonalne do wartości przetwarzanej „liczby”.

Przetworniki takie sterowane mikroprocesorami mogą generować rozmaite przebiegi napięciowe.

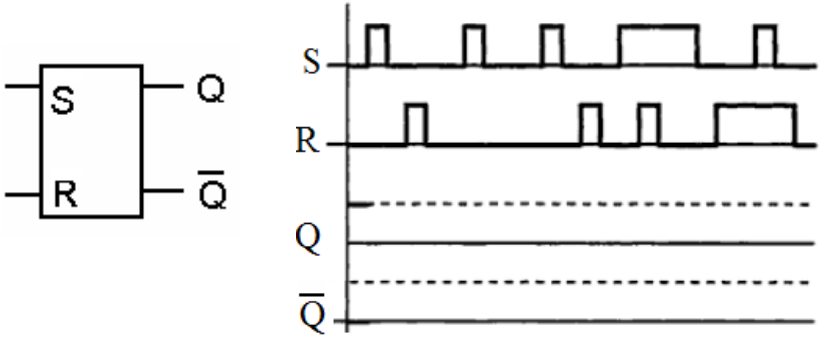
Lista zadań – 13

1) Wykonać działanie $93{,}5_{10} - 42{,}75_{10}$ stosując kody U2.

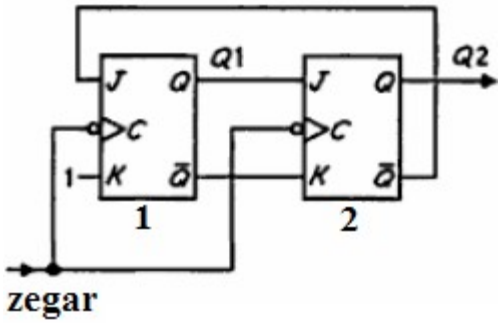
2) Do czterech linii A,B,C i D należy podłączyć kombinacyjny układ logiczny, który na czterech wyjściach E, F, G i H będzie generował stany według tabel Karnaugh na rys. obok.



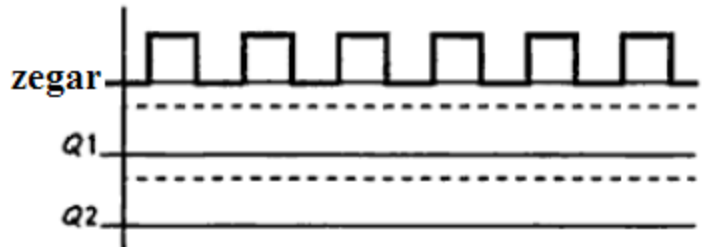
3) Naszkicować przebieg stanów wyjściowych przerzutnika RS dla podanych przebiegów wejściowych S i R.



4) Dla podanego układu uzupełnić tabelę prawdy i narysować przebiegi napięć na wyjściach Q 1 i Q2.



impulsy zegara	Q1	Q2	J1	K1	J2	K2
0	0	0				
1						
2						
3						



Lista zadań – 13

5) Zbudować przerzutnik typu T z przerzutnika typu JK.

6) Zbudować z przerzutników i bramek licznik liczący do 10.