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

Elektronika (konspekt)

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

www.pe.ifd.uni.wroc.pl

Wykład 12

Podstawy elektroniki cyfrowej
(kody i układy logiczne kombinacyjne)

Dwa znaki wystarczają aby w układach cyfrowych i komputerach zapisywać wszelaką informację - liczby, słowa, instrukcje itp.

Kilka elementarnych bramek logicznych wystarcza aby budować urządzenia cyfrowe i komputerowe zdolne wykonywać różnorodne zdania i pełnić rozmaite funkcje.

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 (pozycji).

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 $c_1=1$)

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

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

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

KODY

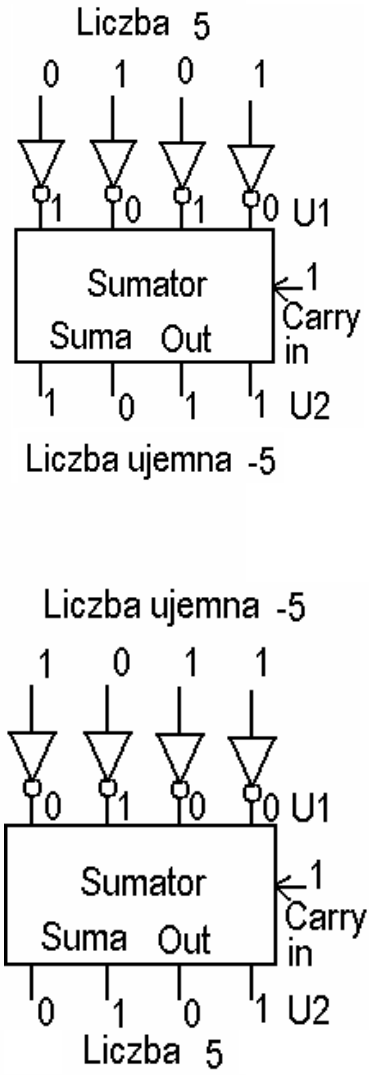
Kodem nazywamy zbiór symboli razem z zasadami stosowania.

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 +3 np. 10_{10}= 0100 0011 $_{(BCD)}$) , Należy zauważyć, że notacje BCD nie są identyczne z zapisem binarnym. Kod BCD wykorzystywany jest w układach z wyświetlaczami cyfr dziesiętnych.

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] =

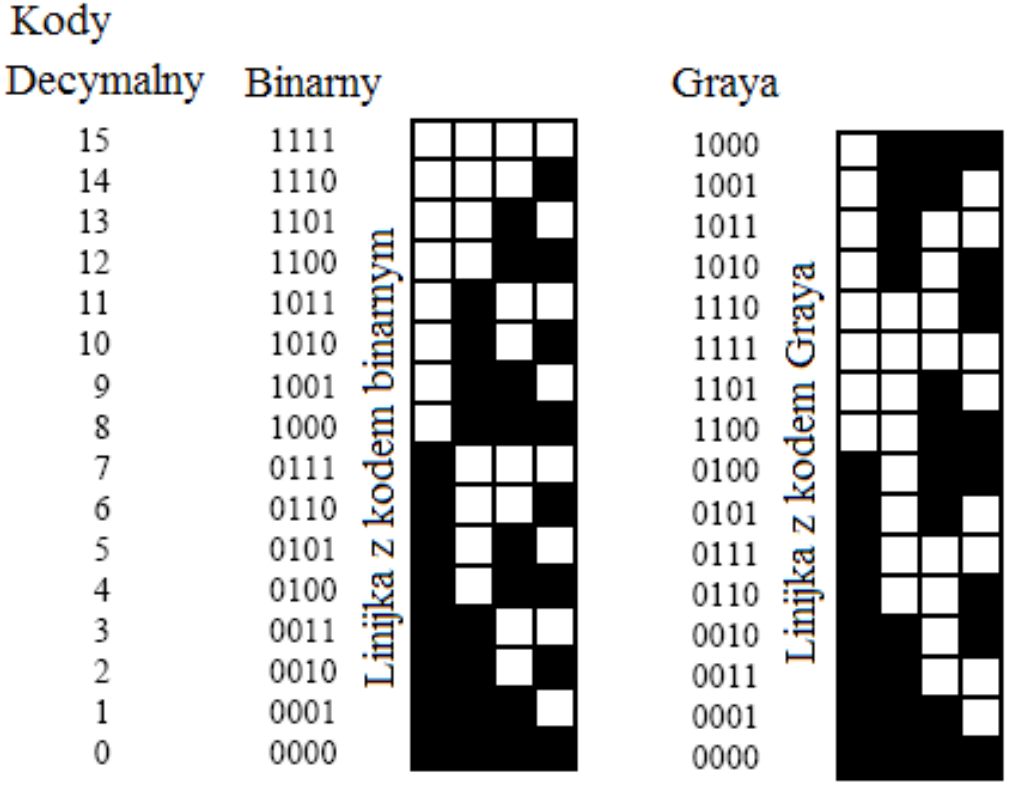
00110 + [**1**1011 +1] = **0**0110 + **1**1100 = 1**0**0010 = **0**0010

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).

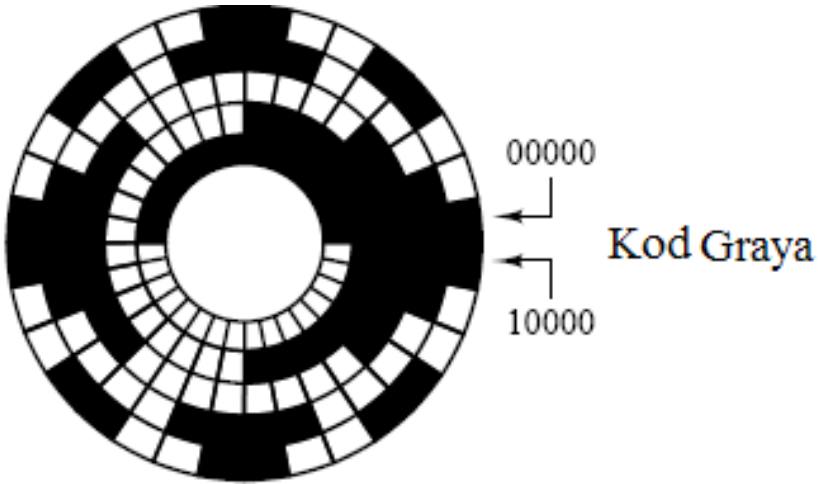
Wśród innych kodów o wzmocnionej odporności na błędy można wymienić kody ze **stałą liczbą jedynek** oraz z tzw. **bitem parzystości**.

Zakładając, że przykładowe linijki pokazane na rysunku mają długość 10 mm i są czterobitowe to jeden segment (z $2^4 = 16$) takiej linijki zapewni rozdzielczość: (10 mm)/16 = 0,625 mm.

Przykłady linijek enkoderów pozycji w ruchu liniowym.



Enkoder kątowy na rysunku dolnym jest enkoderem 5 bitowym i jego rozdzielczość kątowa wynosi: $360°/(2^5) = 360°/32 = 11{,}25°$. Oczywiście dla zwiększenia rozdzielczości zwiększamy liczbę bitów. Kod Graya w takim zastosowaniu poprawia odporność na zakłócenia bo przemieszczenie na sąsiednią pozycję oznacza zamianę na przeciwny tylko jednego bitu!



Enkodery pozycji mają duże zastosowanie przede wszystkim w robotyce, precyzyjnych obrabiarkach i w obszarze rozmaitych urządzeń intensywnie rozwijającej się mechatroniki.

Formaty liczb binarnych zmiennopozycyjnych

(Floating point standard IEEE-P754)

[Znak: 1 bit][(Wykładnik z przesunięciem: 8, 10 lub 15 bitów] [Ukryta jedynka mantysy: 0 bitów][Mantysa: 23, 52 lub 63 bity]. 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 = $1x2^{-127}$)

Obok wymienionych już kodów (binarny, ósemkowy – oktalny, szesnastkowy – heksadecymalny, BCD) jest jeszcze tzw. kod ASCII przyjęty jako standard przez wszystkich producentów sprzętu komputerowego. Kod ASCII definiuje znaki graficzne i kontrolne (2^7 - znaków alfanumerycznych) związane z tekstami używanymi w programowaniu.

Kod ASCII jest zamieszczony na następnej stronie.

Należy jeszcze wymienić kody instrukcji dla procesorów znane jako opkody (opcode – operation codes, czyli są to kody rozumiane przez procesory). Niestety opkody okazują się różne dla różnych typów procesorów od różnych producentów.

Dla ułatwienia programistom identyfikację i pamiętanie znaczenia poszczególnych opkodów stosowane są tzw. mnemoniki. W efekcie programista programujący na najbardziej elementarnym poziomie stosuje mnemoniki a napisany program jest przetwarzany (translated) na kod maszynowy zawierający opkody i dane przy pomocy programu zwanego assemblerem.

Kod ASCII

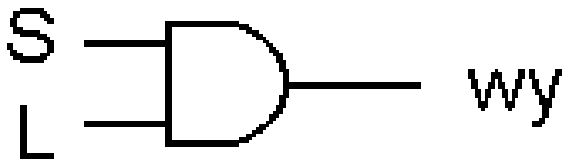
American Standard Code for Information Interchange.

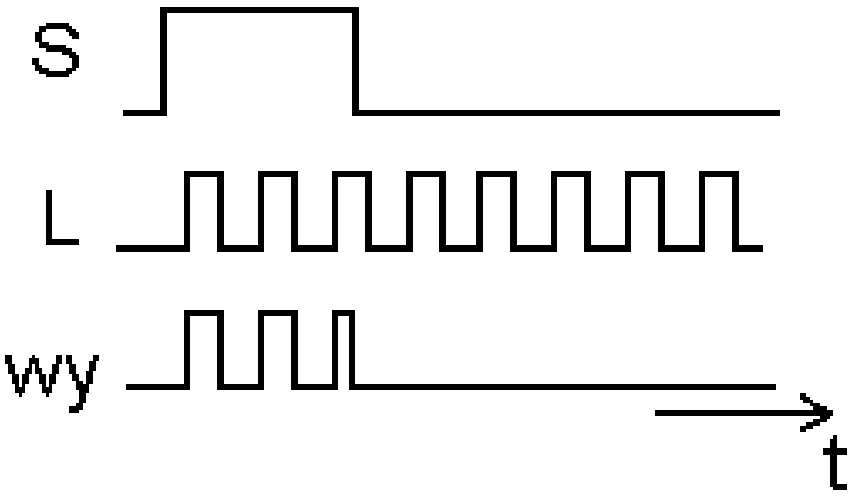
Graphic or control	ASCII (hex)	Graphic or control	ASCII (hex)	Graphic or control	ASCII (hex)	Graphic or control	ASCII (hex)	Graphic or control	ASCII (hex)	Graphic or control	ASCII (hex)
NUL	00	SYN	16	,	2C	B	42	X	58	n	6E
SOH	01	ETB	17	−	2D	C	43	Y	59	o	6F
STX	02	CAN	18	.	2E	D	44	Z	5A	p	70
ETX	03	EM	19	/	2F	E	45	[	5B	q	71
EOT	04	SUB	1A	0	30	F	46	\	5C	r	72
ENQ	05	ESC	1B	1	31	G	47	]	5D	s	73
ACK	06	FS	1C	2	32	H	48	↑	5E	t	74
BEL	07	GS	1D	3	33	I	49	←	5F	u	75
BS	08	RS	1E	4	34	J	4A	`	60	v	76
HT	09	US	1F	5	35	K	4B	a	61	w	77
LF	0A	SP	20	6	36	L	4C	b	62	x	78
VT	0B	!	21	7	37	M	4D	c	63	y	79
FF	0C	"	22	8	38	N	4E	d	64	z	7A
CR	0D	#	23	9	39	O	4F	e	65	{	7B
SO	0E	$	24	:	3A	P	50	f	66	|	7C
SI	0F	%	25	;	3B	Q	51	g	67	}	7D
DLE	10	&	26	<	3C	R	52	h	68	~	7E
DC1	11	'	27	=	3D	S	53	i	69	DEL	7F
DC2	12	(	28	>	3E	T	54	j	6A		
DC3	13	)	29	?	3F	U	55	k	6B		
DC4	14	*	2A	@	40	V	56	l	6C		
NAK	15	+	2B	A	41	W	57	m	6D		

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

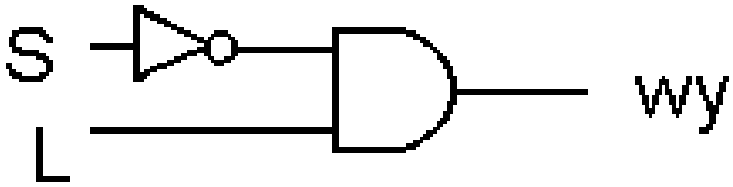
Proste układy z bramkami cyfrowymi.

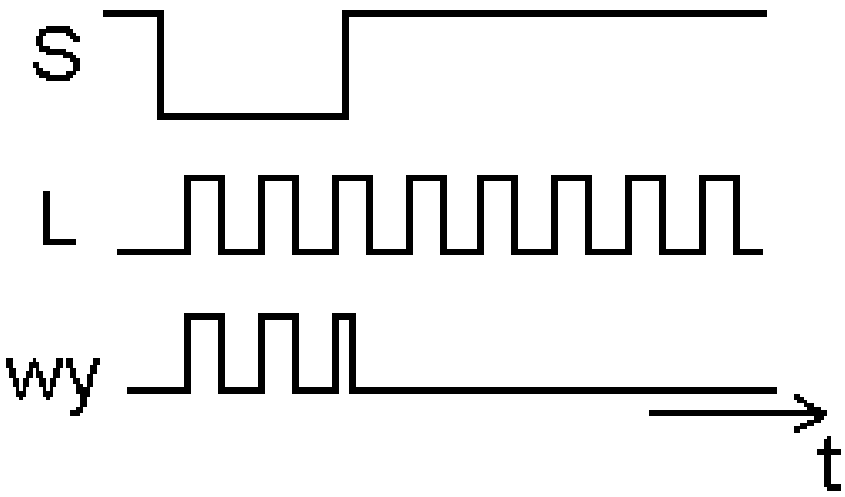
Układ koincydencyjny



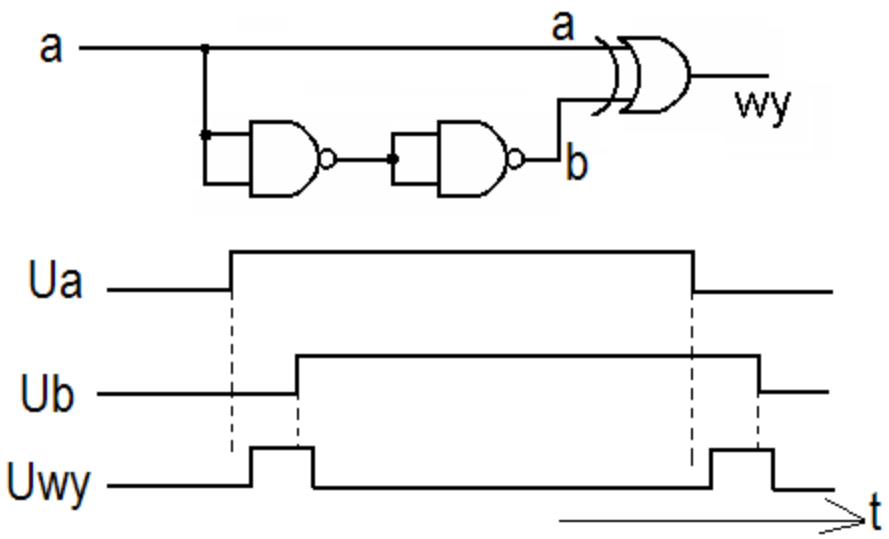


Układ antykoincydencyjny

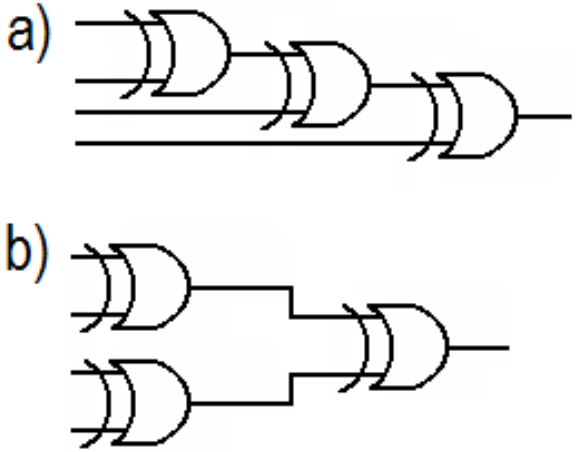




Efektem różnych czasów propagacji wzdłuż różnych ścieżek sygnału może być generowanie wąskich impulsów czasem zamierzone i pożądane a czasem szkodliwe.

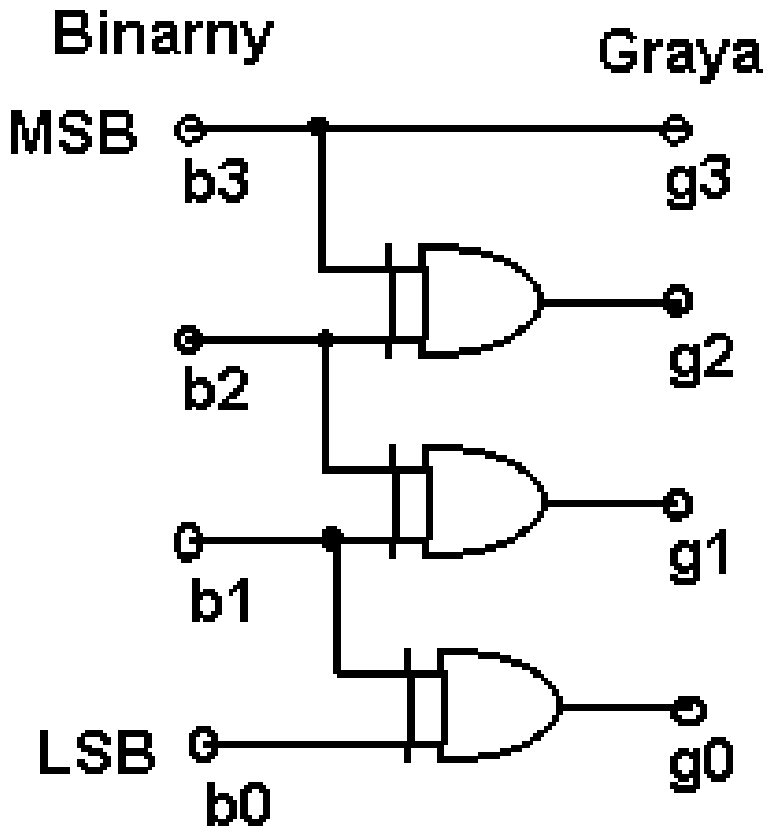


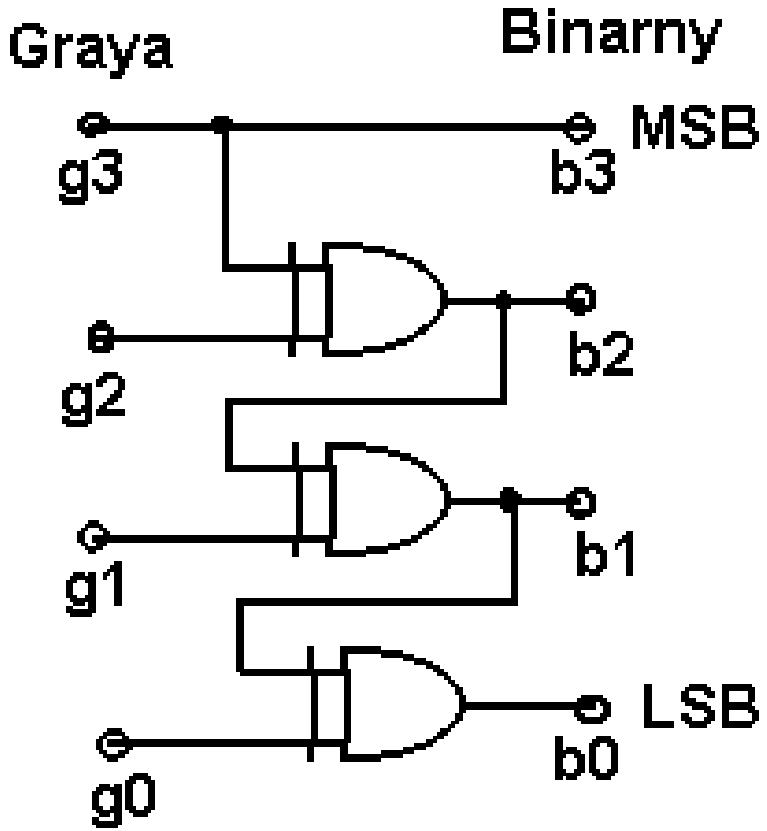
Z dwóch pokazanych na rysunku układów do generowania bitu parzystości lepszy jest wariant „b”, w którym czas ustalania stanu wyjściowego jest o 1/3 krótszy od czasu ustalania stanu w wariancie „a”



Przykłady:

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





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.

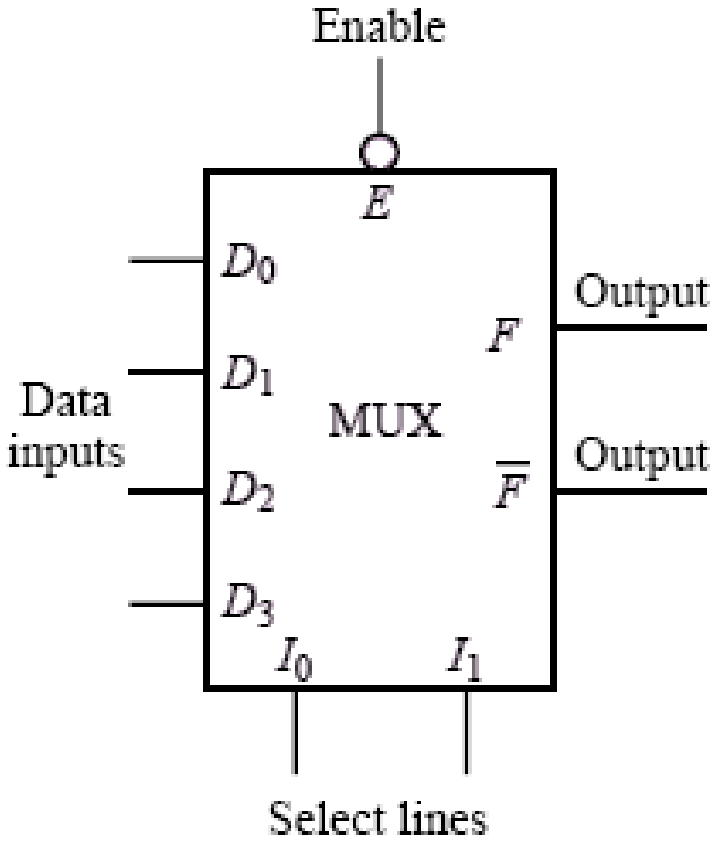
Multipleksery stosowane są np. na wejścia przetworników analogowo-cyfrowych (AD). Multipleksery i demultipleksery mogą realizować multipleksowany system przesyłania danych, mogą też być stosowane do realizacji innych układów kombinacyjnych realizujących złożone funkcje np. linijka świetlna.

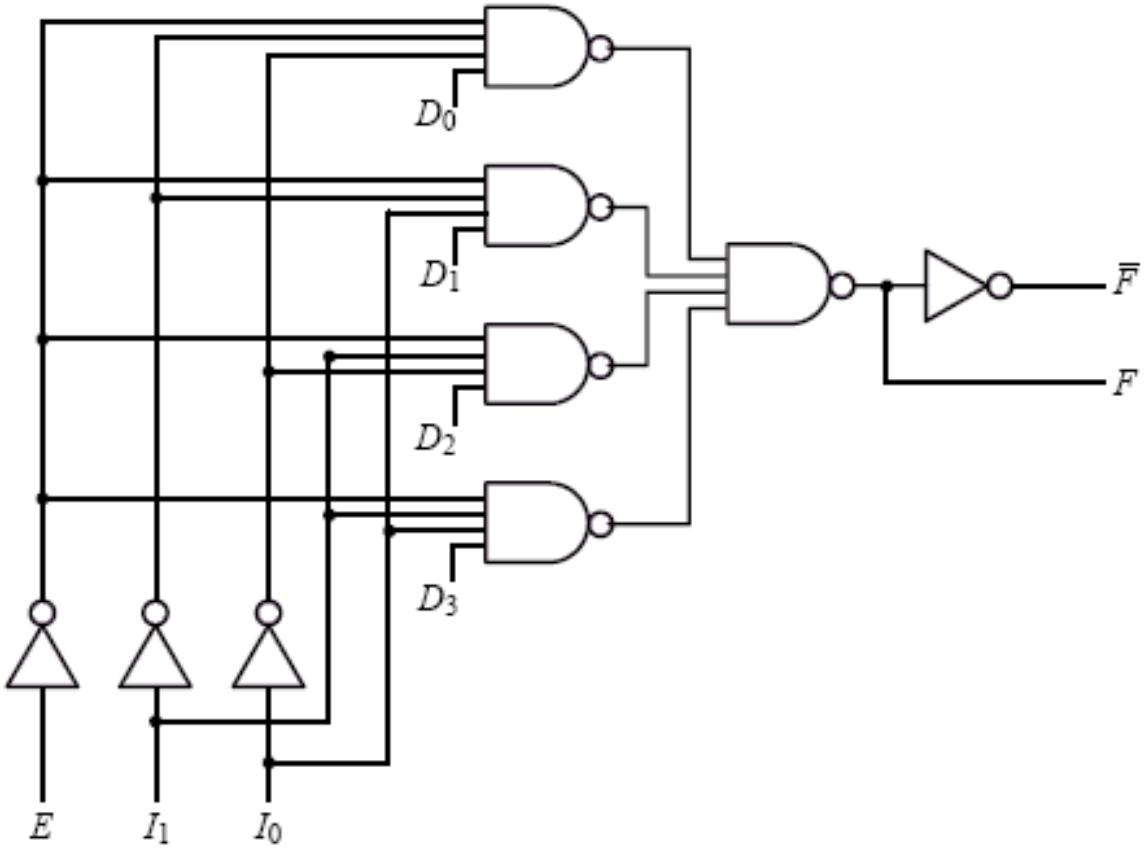
Multiplekser

Przykładowa struktura

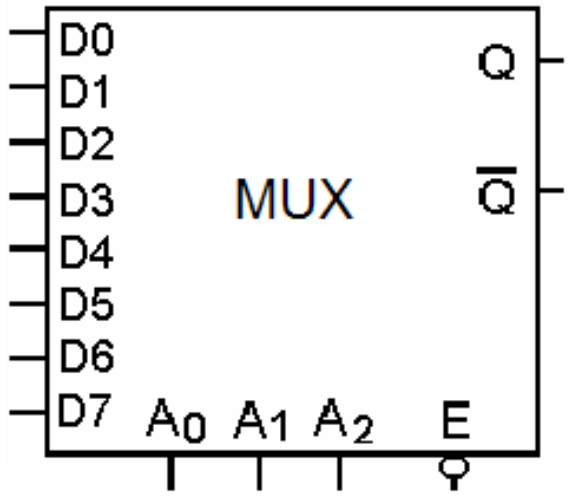
4 – bitowego multipleksera

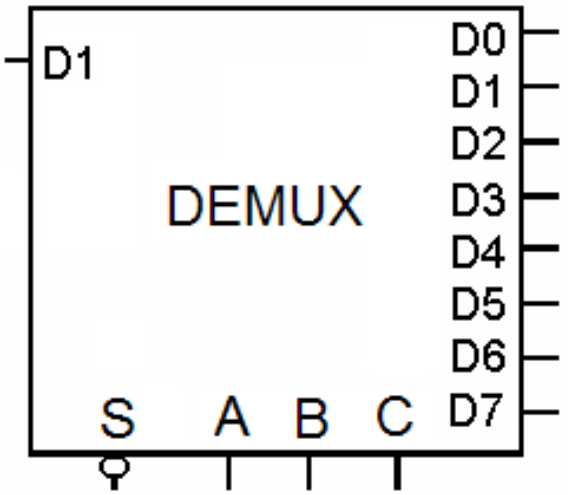
i jej symbol:

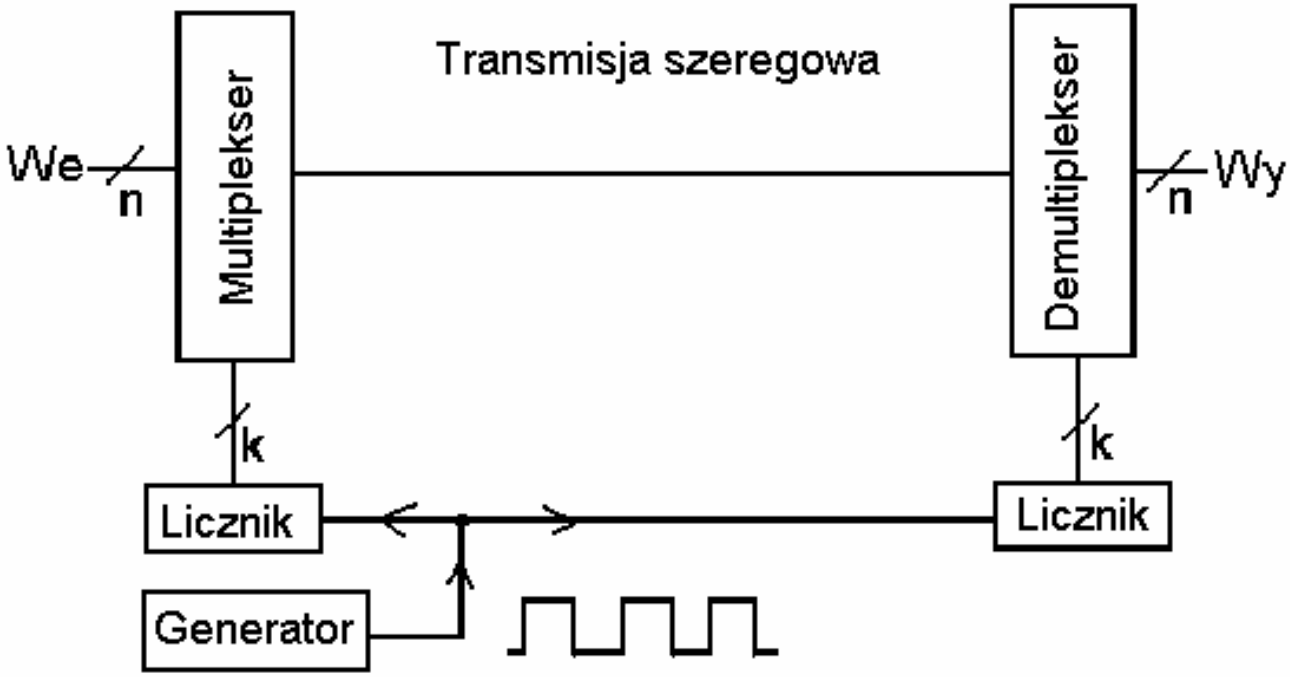




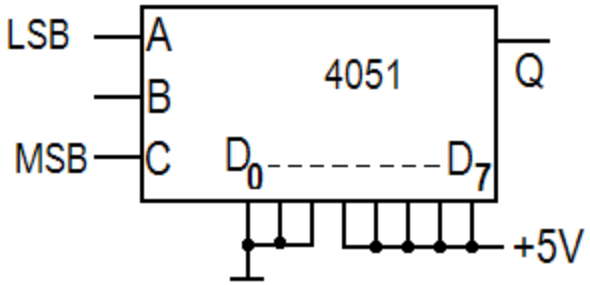
Na rysunku zamieszczono przykład multipleksera i demultipleksera oraz uproszczony układ zamiany transmisji równoległej na szeregową i ponownego powrotu do transmisji równoległej (związek między n i k: $n = 2^k$) . Symbole D i Q oznaczają linie danych, A,B i C – linie adresowe, S – Strobe, E – enable,





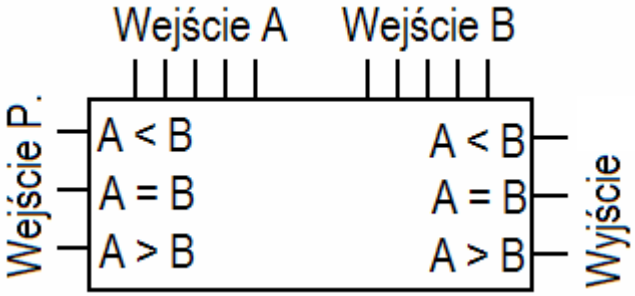


Realizacja tabeli prawdy przy pomocy multipleksera. Układ obok wyróżnia liczby większe od 2 podawane na 3-bitowe wejście ABC.



Komparator cyfrowy

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

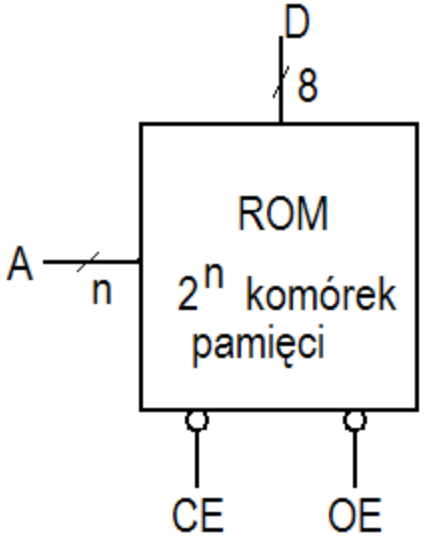


Pamięć ROM jako przykład układu kombinacyjnego

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,



Przykład.

Pamięć ROM jako układ kombinacyjny.

Pamięć z m = 2 liniami adresowymi zawiera 2^m słów 4 bitowych pokazuje rys. obok.

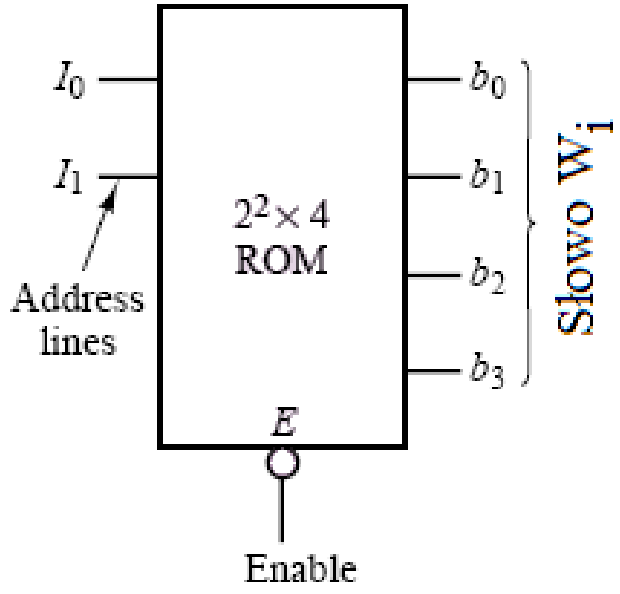
Jeżeli na liniach pojawi się adres 01 to na wyjściu (przy aktywującym stanie na wejściu E) pojawi się słowo: 1001.

W zależności od rozmiarów ROM możemy zaimplementować bardziej lub mniej złożoną funkcję logiczną.

W pewnym sensie ROM można traktować jak MUX, który na wyjściu zamiast pojedynczego bitu wystawia słowo n - bitowe.

Niestety w ROM raz zapisane funkcje nie można zmienić.

ROM address		Zawartość				Słowa
I_1	I_0	b_3	b_2	b_1	b_0	
0	0	0	1	1	0	← W_0
0	1	1	0	0	1	← W_1
1	0	0	1	1	0	← W_2
1	1	1	1	1	1	← W_3



Pod tym względem lepszym rozwiązaniem są pamięci EPROM (erasable programmable read-only memory), których zawartość można zaprogramować wielokrotnie.

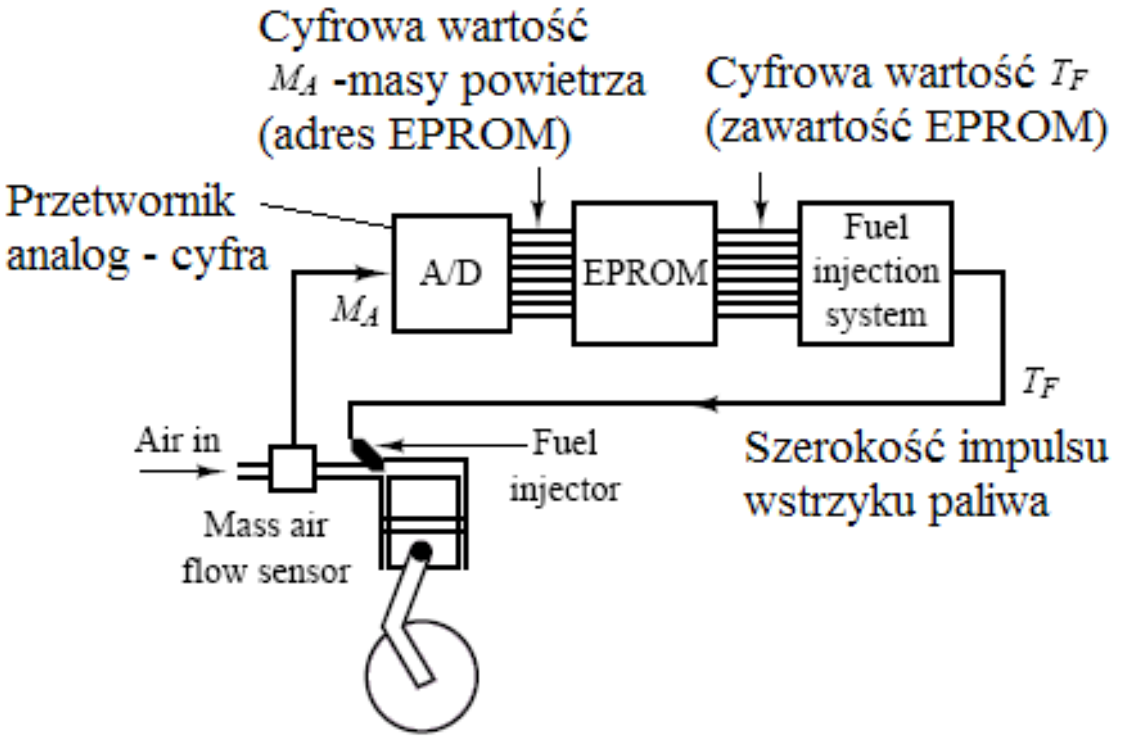
Przykład.

System wstrzykiwania paliwa do cylindrów silnika.

Dla optymalnego działania silnika spalinowego ważna jest proporcja między masą porcji powietrza M_A i masa porcji paliwa M_F:

Powinno być: $M_A/M_F = 14,7$.

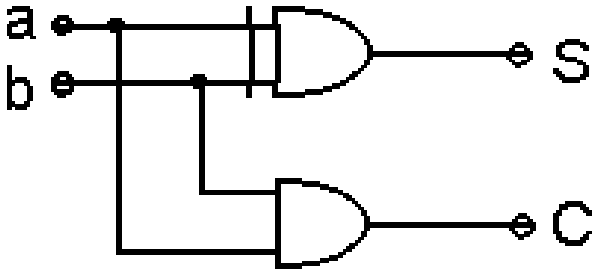
Zatem sensor mierzący ilość wpływającego powietrza dostarcza sygnał analogowy do przetwornika analogowo-cyfrowego, z którego wartość cyfrowa jest adresem odpowiedniej komórki pamięci EPROM zawierającej przeliczoną wartość czasu wstrzykiwania paliwa: $T_F = M_A/(14,7 \times K_F)$ [0,1 ms]. K_F jest szybkość wstrzykiwania paliwa w odpowiednich jednostkach.

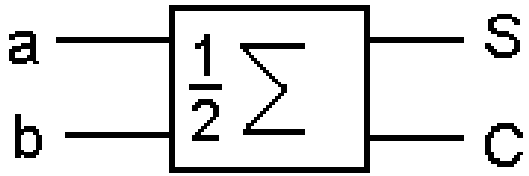


Sumatory

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.

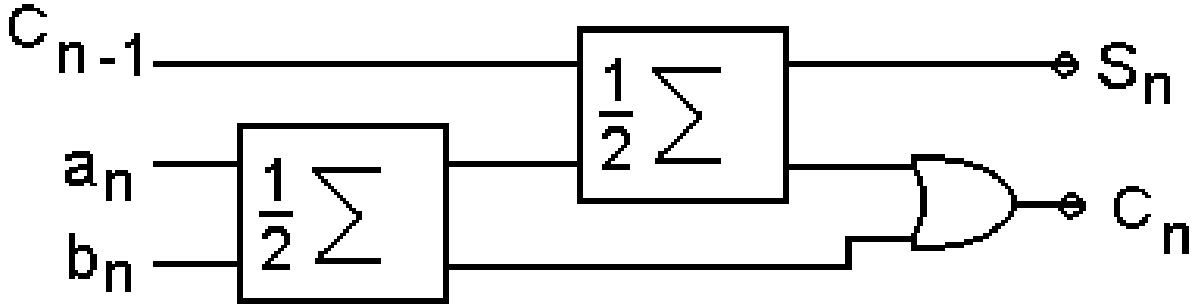


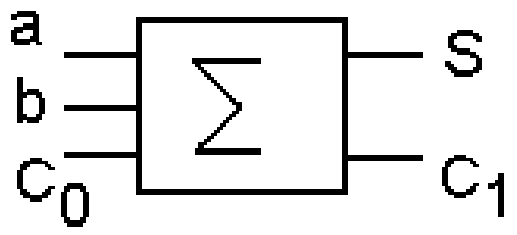


Sumator

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

Schemat i symbol sumatora





Elektronika. Lista – 12

1) Przedstaw realizację tabeli prawdy przy pomocy multipleksera 4051, który będzie wyróżniał liczby parzyste podawane na 3-bitowe wejście ABC.

2) Zaproponuj schemat multipleksera (z łożonego z bramek logicznych) o dwóch liniach adresowych i czterech liniach wyjściowych.

3) Zapisać w pamięci EPROM takie wartości aby uzyskać zamianę liczb od 0 do 3 na liczby od 8 do 11.

I_1	I_0	b_3	b_2	b_1	b_0	
0	0					W_0
0	1					W_1
1	0					W_2
1	1					W_3

4) Narysuj układ bramek logicznych generujący bit parzystości dla kodu ASCII (7 linii kod, 8-linia bit parzystości).