

TECHNIKA CYFROWA

WYKŁAD (16h – 6 spotkań)

LABORATORIUM (9h – 3 spotkania po 3h)

ZALICZENIE PRZEDMIOTU – ocena wagowa:

- pisemny egzamino-test z wykładu (10 pytań) – waga oceny 0.6
warunek uczestnictwa: zaliczone laboratorium
- praca w trakcie laboratoriów – waga oceny 0.4

(50%, 60%, 70%, 80%, 90% => dst, +dst, db, +db, bdb)

dr inż. Rafał SZYMANOWSKI

R.Szymanowski@wit.edu.pl

ZAGADNIENIA (1) – cyfrowe układy logiczne

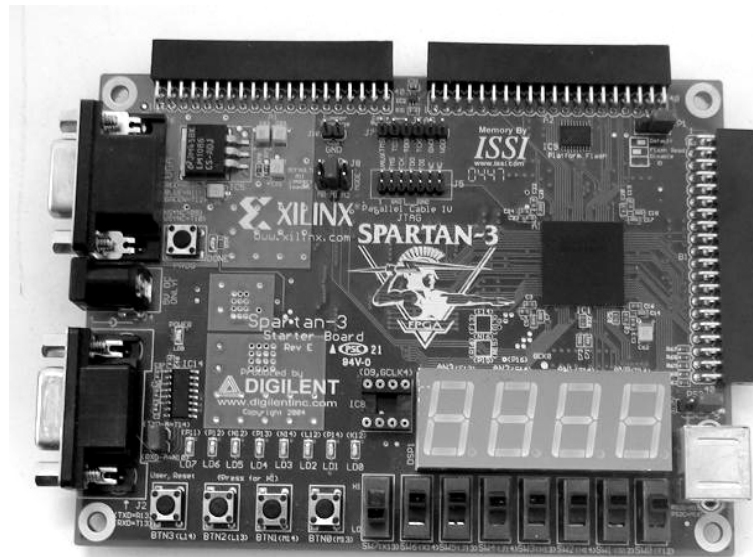
- 1) Kody.**
- 2) Algebra Boole'a dla sygnałów binarnych.**
- 3) Funkcje logiczne, formy boolowskie.**
- 4) Bramki logiczne.**
- 5) Synteza układów kombinacyjnych.**
- 6) Kombinacyjne bloki funkcjonalne.**
- 7) Układy arytmetyczne.**
- 8) Przerzutniki.**
- 9) Układy sekwencyjne.**
- 10) Synteza układów sekwencyjnych.**
- 11) Rejestry.**
- 12) Liczniki.**

ZAGADNIENIA (2) – cyfrowe układy scalone

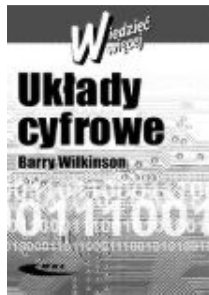
- 1) Rodzaje cyfrowych układów scalonych.**
- 2) Układy TTL, ECL i MOS.**
- 3) Układy CMOS.**
- 4) Przerzutniki scalone.**
- 7) Pamięci scalone.**
- 8) Układy programowalne (SPLD, CPLD i FPGA).**
- 9) Systemy projektowe.**
- 10) Język VHDL.**

LABORATORIUM – projekty... (FPGA, VHDL)

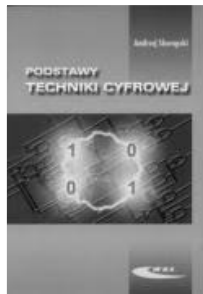
- 1) Układy kombinacyjne...
- 2) Układy synchroniczne...
- 3) Indywidualne projekciki...



LITERATURA (uzupełniająca)

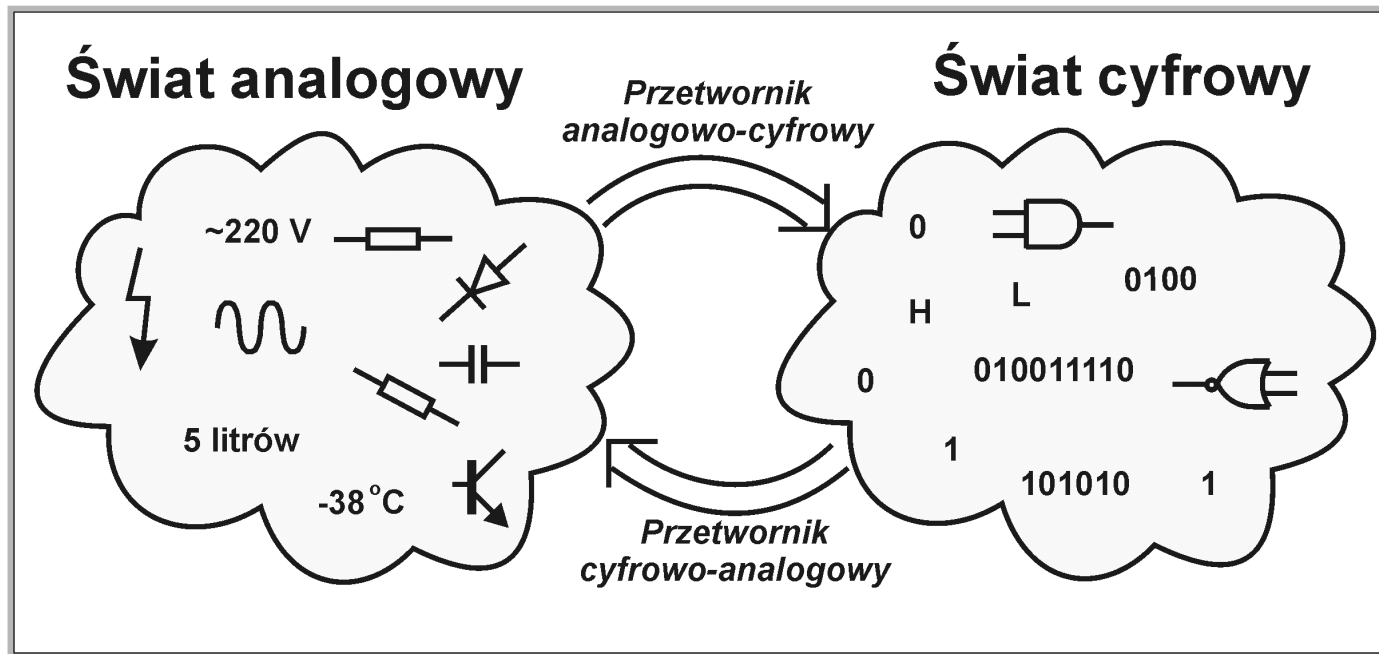


B. Wilkinson
Układy cyfrowe
WKŁ 2003, seria wydawnicza *Wiedzieć więcej*



A. Skorupski
Podstawy techniki cyfrowej
WKŁ 2004

UKŁADY CYFROWE



ANALOGOWY ZAPIS INFORMACJI

wszystkie wartości napięć i prądów jakie tylko sobie wymyślimy

CYFROWY ZAPIS INFORMACJI

wartości napięć i prądów wyrażone tylko poprzez skończony zakres liczb, które zapisane są w postaci kodów!

UKŁADY CYFROWE

Świat cyfrowy → Świat „zer i jedynek”

Zastosowanie układów cyfrowych:

- Sterowanie
- Obliczenia

Zalety układów cyfrowych:

- Łatwość przechowywania informacji
- Duże szybkości transmisji
- Łatwość detekcji stanów logicznych
(dwóch poziomów napięć)
- Odporność na zakłócenia

UKŁADY CYFROWE

Starożytni Rzymianie palec nazywali „digitus”.

**Palce służyły do liczenia,
stąd podstawy dwuwartościowości.**

digitus → digital → cyfrowy

UKŁADY CYFROWE

Ogólnie w układach elektronicznych najłatwiej można określić dwie wykluczające się sytuacje:

układ włączony

układ wyłączony

Czyli jest napięcie albo go nie ma.

... lampa świeci lub jest wyłączona, oczy otwarte lub zamknięte, bilet skasowany lub nie ...

Ponadto w takim rozumowaniu działania nie ma sytuacji pośrednich. Zawsze istnieje tylko jedna z dwóch sytuacji.

UKŁADY CYFROWE

Metody oznaczania dwóch stanów (sytuacji):

0 – stan zero

1 – stan jeden

L – *Low* – poziom L, poziom niski (napięcia)

H – *High* – poziom H, poziom wysoki (napięcia)

UKŁADY CYFROWE



Sygnały wejściowe – sygnały cyfrowe (0, 1), informacje analogowe po przetworzeniu do postaci cyfrowej

Sygnały wyjściowe – sygnały cyfrowe (0, 1)

CYFROWY ZAPIS INFORMACJI

WSTĘP...

Znak – symbol graficzny

Alfabet – zbiór znaków

np. $A = \{a, b, \dots, z\}$, $D = \{0, 1, \dots, 9\}$

Słowo – skończony i uporządkowany łańcuch znaków
określający pewną informację

np. 234, akjn34k, str36, %, @, ق, ≥, alareg87

Długość słowa – liczba znaków tworzących słowo

CYFROWY ZAPIS INFORMACJI

Jeżeli alfabet zawiera tylko dwa znaki to nosi on nazwę *alfabetu dwójkowego (binarnego)*

$$B = \{0,1\}$$

Słowo dwójkowe o długości $n = 1$ to bit

Słowo dwójkowe o długości $n = 8$ to bajt

Słowo dwójkowe o długości n ma 2^n kombinacji

CYFROWY ZAPIS INFORMACJI – *Kody*

Kod – słowo oznaczające pewną informację

Kod naturalny – to kod pozycyjny, w którym każdy znak a_i ma przypisaną pozycję i , przy czym tej pozycji przyporządkowana jest waga $w_i = p^i$, gdzie p to podstawa kodu

Kod liczbowy – kod naturalny, którego słowa oznaczają liczby

CYFROWY ZAPIS INFORMACJI – *Kody liczbowe*

NATURALNY KOD DZIESIĘTNY

Podstawa: $p = 10$

Wagi: $w_0 = 1, w_1 = 10, w_2 = 100, w_3 = 1000, \dots, w_{n-1} = 10^{n-1}$

np.

$$2364 = 2 \cdot 1000 + 3 \cdot 100 + 6 \cdot 10 + 4 \cdot 1 \Rightarrow$$

$$a_3 = 2, a_2 = 3, a_1 = 6, a_0 = 4$$

CYFROWY ZAPIS INFORMACJI – *Kody liczbowe*

NATURALNY KOD DWÓJKOWY (BINARNY)

Podstawa: $p = 2$

Wagi: $w_0 = 1, w_1 = 2, w_2 = 4, w_3 = 8, \dots, w_{n-1} = 2^{n-1}$

np.

101001 \Rightarrow odpowiada liczbie dziesiętnej 41, ponieważ:

$$1 \cdot 32 + 0 \cdot 16 + 1 \cdot 8 + 0 \cdot 4 + 0 \cdot 2 + 1 \cdot 1 = 41$$

Reprezentacja kodu dziesiętnego w kodzie binarnym (4-bitowym)

Kod dziesiętny	Kod binarny
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111

CYFROWY ZAPIS INFORMACJI – *Kody liczbowe*

NATURALNY KOD DWÓJKOWO-DZIESIĘTNY – *kod BCD* **(*BCD – Binary Coded Decimal*)**

**Każda cyfra w kodzie dziesiętnym jest reprezentowana
przez 4-bitową liczbę dwójkową**

np. 2364 \Rightarrow 0010 0011 0110 0100

CYFROWY ZAPIS INFORMACJI – *Kody liczbowe*

KOD GRAY'A

**Sąsiednie słowa kodu różnią się
tylko wartością jednego bitu**

np.

dla dwóch zmiennych: 00, 01, 11, 10

dla trzech zmiennych: 000, 001, 011, 010, 110, 111, 101, 100

CYFROWY ZAPIS INFORMACJI – *Kody liczbowe*

KOD SZESTASTKOWY (hexadecimal)

Stosowany w celu skrócenia zapisu naturalnego kodu dwójkowego.

Znaki: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

gdzie $A = 10_{10}$, $B = 11_{10}$, $C = 12_{10}$, $D = 13_{10}$, $E = 14_{10}$, $F = 15_{10}$.

Każda „czwórka” bitów zamieniana jest jeden znak:

$$100111000000010000110000_2 = 9C0430_{16}$$

CYFROWY ZAPIS INFORMACJI – *Kody liczbowe*

KOD ÓSEMKOWY (OCT)

Stosowany w celu skrócenia zapisu naturalnego kodu dwójkowego.

Znaki: 0, 1, 2, 3, 4, 5, 6, 7

Każda „trójka” bitów zamieniana jest jeden znak:

$$100\ 111\ 000\ 000\ 010\ 000\ 110\ 000_2 = 17002060_8$$

KOD ALFANUMERYCZNY ASCII

(*American Standard Code for Information Interchange*)

Siedmiobitowy kod alfanumeryczny ASCII
 ND – liczba dziesiętna, HEX – liczba szesnastkowa

ND	HEX	Znak	ND	HEX	Znak	ND	HEX	Znak	ND	HEX	Znak
0	00		32	20	SP	64	40	@	96	60	`
1	01	☺	33	21	!	65	41	A	97	61	a
2	02	☹	34	22	”	66	42	B	98	62	b
3	03	♥	35	23	#	67	43	C	99	63	c
4	04	♦	36	24	\$	68	44	D	100	64	d
5	05	♣	37	25	%	69	45	E	101	65	e
6	06	♠	38	26	&	70	46	F	102	66	f
7	07	●	39	27	,	71	47	G	103	67	g
8	08	■	40	28	(72	48	H	104	68	h
9	09	○	41	29)	73	49	I	105	69	i
10	0A	▣	42	2A	*	74	4A	J	106	6A	j
11	0B	♂	43	2B	+	75	4B	K	107	6B	k
12	0C	♀	44	2C	,	76	4C	L	108	6C	l
13	0D	♪	45	2D	–	77	4D	M	109	6D	m
14	0E	♫	46	2E	.	78	4E	N	110	6E	n
15	0F	⚙	47	2F	/	79	4F	O	111	6F	o
16	10	▶	48	30	0	80	50	P	112	70	p
17	11	◀	49	31	1	81	51	Q	113	71	q
18	12	↑	50	32	2	82	52	R	114	72	r
19	13	!!	51	33	3	83	53	S	115	73	s
20	14	¶	52	34	4	84	54	T	116	74	t
21	15	§	53	35	5	85	55	U	117	75	u
22	16	■	54	36	6	86	56	V	118	76	v
23	17	‡	55	37	7	87	57	W	119	77	w
24	18	↑	56	38	8	88	58	X	120	78	x
25	19	↓	57	39	9	89	59	Y	121	79	y
26	1A	→	58	3A	:	90	5A	Z	122	7A	z
27	1B	↑	59	3B	;	91	5B	[123	7B	{
28	1C	↵	60	3C	<	92	5C	\	124	7C	
29	1D	↕	61	3D	=	93	5D]	125	7D	}
30	1E	▲	62	3E	>	94	5E	^	126	7E	~
31	1F	▼	63	3F	?	95	5F	_	127	7F	DEL

ALGEBRA BOOLE'A

George Boole (1815 – 1864) opracował algebrę dotyczącą logiki matematycznej w oparciu o rachunek zdań i teorię mnogości (zbiorów), które to mogą być prawdziwe lub fałszywe. Inne zdania nie istnieją.

Trzy operacje logiczne

KONIUNKCJA, iloczyn logiczny (\wedge) $\rightarrow p \wedge q$ czyli „ p i q ”

ALTERNATYWA, suma logiczna (\vee) $\rightarrow p \vee q$ czyli „ p lub q ”

NEGACJA (\neg) $\rightarrow \neg q$ czyli „nie q ”, „nieprawda, że q ”

Algebra Boole'a → zmienne binarne → Shannon

W oparciu o algebrę Boole'a, w roku 1938 Shannon sformułował algebrę dla zmiennych binarnych.

Trzy operacje: suma logiczna (+)
iloczyn logiczny (•)
negacja (' , $\bar{}$, /)

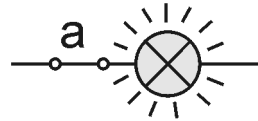
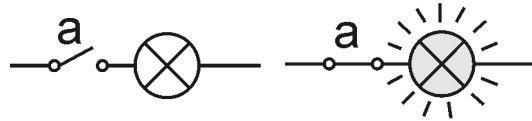
$\langle B, +, \cdot, ', 0, 1 \rangle$ gdzie $B = \{0, 1\}$

suma logiczna (dodawanie logiczne, *lub*, *OR*): $a + b$
iloczyn logiczny (mnożenie logiczne, *i*, *AND*): $a \cdot b$
negacja logiczna (dopełnienie, *NOT*): a' , $/a$, \bar{a}

Algebra Boole'a → przełączanie... (nareszcie elektronika?)

a = 0

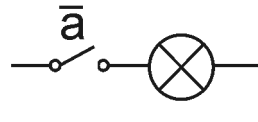
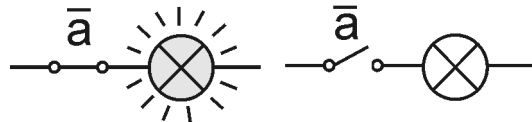
a = 1



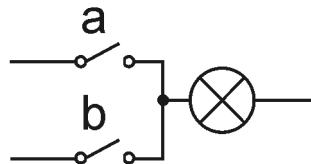
NOT

NIE

negacja



a + b

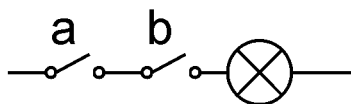


OR

LUB

suma

a • b

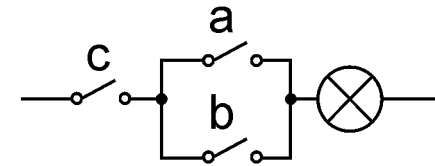
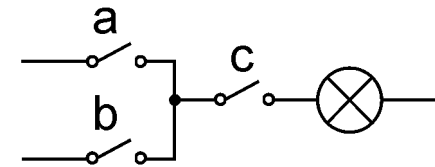


AND

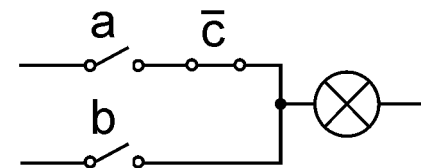
I

iloczyn

Np.: $(a + b) \cdot c = c(a + b)$



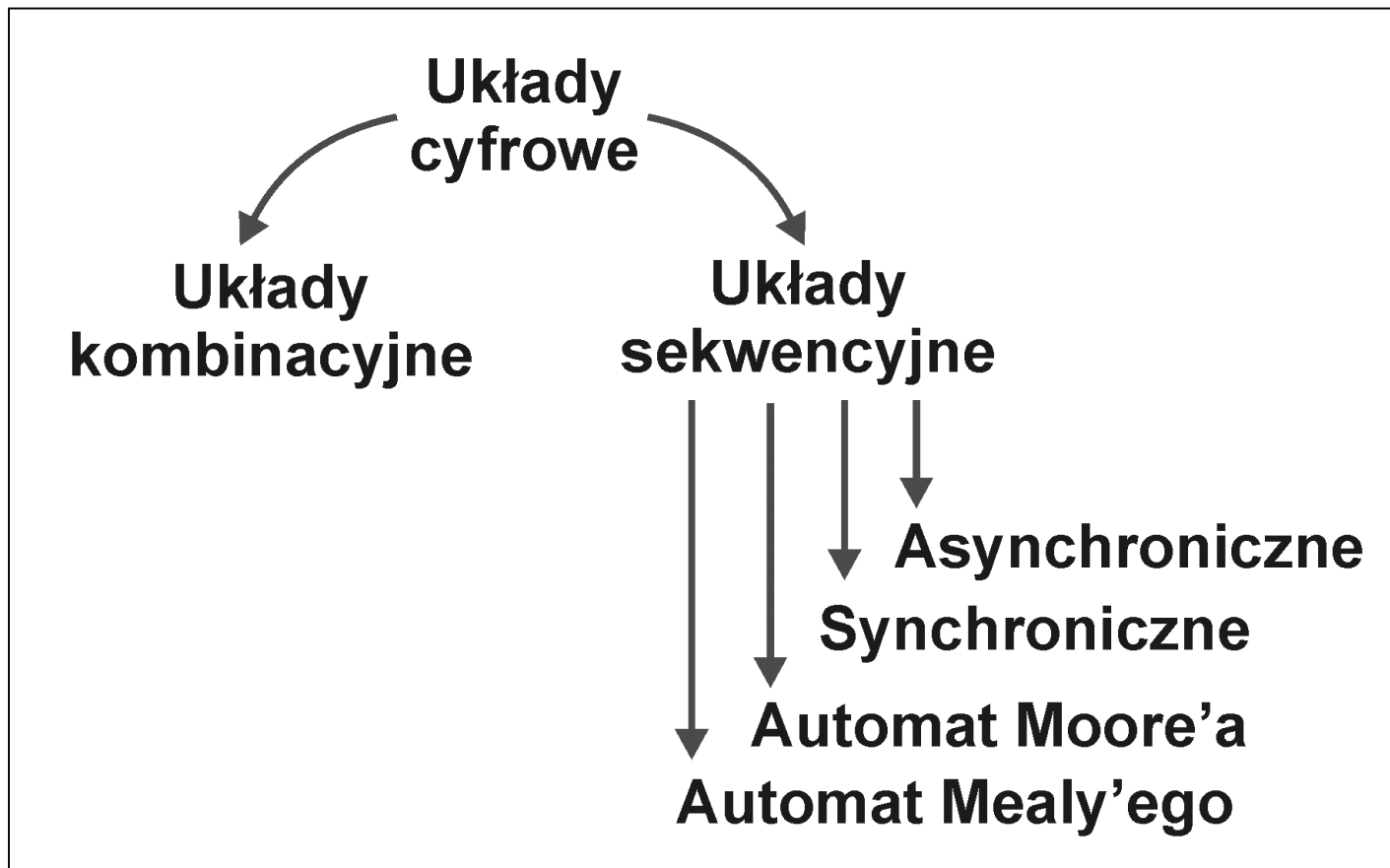
$(a \cdot \bar{c}) + b = a\bar{c} + b$



Kontrolka zamkniętych drzwi
w samochodzie...

UKŁADY CYFROWE

Podział...



UKŁADY KOMBINACYJNE



Układ kombinacyjny – układ cyfrowy realizujący poprzez swoje działanie n funkcji logicznych dla m sygnałów dwuwartościowych (0, 1)

UKŁADY KOMBINACYJNE

Do opisu działań logicznych wykonywanych przez układ kombinacyjny stosujemy algebrę Boole'a, która określa trzy podstawowe operacje logiczne:

negacja

x	$f(x) = \overline{x}$
0	1
1	0

UKŁADY KOMBINACYJNE

suma logiczna

$x_1 \ x_0$	$f(x) = x_1 + x_0$
0 0	0
0 1	1
1 0	1
1 1	1

iloczyn logiczny

$x_1 \ x_0$	$f(x) = x_1 \cdot x_0$
0 0	0
0 1	0
1 0	0
1 1	1

UKŁADY KOMBINACYJNE

Ponadto operacje sumy i iloczynu mają następujące tożsamości:

$$1. \quad x + 0 = x$$

$$2. \quad x + 1 = 1$$

$$3. \quad x \cdot 0 = 0$$

$$4. \quad x \cdot 1 = x$$

$$5. \quad x + x = x$$

$$6. \quad x + \bar{x} = 1$$

$$7. \quad x \cdot x = x$$

$$8. \quad x \cdot \bar{x} = 0$$

UKŁADY KOMBINACYJNE

oraz prawa:

sklejania $\mathbf{x}_1\mathbf{x}_2 + \mathbf{x}_1\overline{\mathbf{x}}_2 = \mathbf{x}_1$

rozdzielczości $\mathbf{x}_1(\mathbf{x}_2 + \mathbf{x}_3) = \mathbf{x}_1\mathbf{x}_2 + \mathbf{x}_1\mathbf{x}_3$
 $\mathbf{x}_1 + \mathbf{x}_2\mathbf{x}_3 = (\mathbf{x}_1 + \mathbf{x}_2)(\mathbf{x}_1 + \mathbf{x}_3)$

pochłaniania $\mathbf{x}_1(\mathbf{x}_1 + \mathbf{x}_2) = \mathbf{x}_1$
 $\mathbf{x}_1 + \mathbf{x}_1\mathbf{x}_2 = \mathbf{x}_1$

de Morgana $\overline{\mathbf{x}_1\mathbf{x}_2} = \overline{\mathbf{x}_1} + \overline{\mathbf{x}_2}$
 $\overline{\mathbf{x}_1 + \mathbf{x}_2} = \overline{\mathbf{x}_1}\overline{\mathbf{x}_2}$

UKŁADY KOMBINACYJNE

Metody opisu działania układów kombinacyjnych

1. tablica zależności, tablica prawdy, tablica wartości funkcji logicznej

x_2 x_1 x_0	y
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	0
1 1 1	1

x_1 x_0	y_1 y_0
0 0	0 0
0 1	0 1
1 0	1 0
1 1	1 1

UKŁADY KOMBINACYJNE

Metody opisu działania układów kombinacyjnych – *cd.*

2. kanoniczny zapis funkcji logicznej

x_2	x_1	x_0	y
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	0
1	1	1	1

Kanoniczna postać sumacyjna

$$y = \bar{x}_2 \bar{x}_1 \bar{x}_0 + \bar{x}_2 x_1 x_0 + x_2 x_1 x_0$$

(dla $y = 1$)

UKŁADY KOMBINACYJNE

Metody opisu działania układów kombinacyjnych – *cd.*

2. kanoniczny zapis funkcji logicznej – *cd.*

x_2	x_1	x_0	y
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	0
1	1	1	1

Kanoniczna postać iloczynowa

$$y = (x_2 + x_1 + x_0)(x_2 + x_1 + \bar{x}_0) \\ (\bar{x}_2 + x_1 + x_0)(\bar{x}_2 + x_1 + \bar{x}_0) \\ (\bar{x}_2 + \bar{x}_1 + x_0)$$

(dla $y = 0$)

UKŁADY KOMBINACYJNE

Metody opisu działania układów kombinacyjnych – *cd.*

3. zbiory iloczynów i sum

Zbiór sum (dla $y = 1$)

x_2 x_1 x_0	y
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	0
1 1 1	1

$$y = \sum (010, 011, 111) x_2 x_1 x_0$$

$$T_3 = \{ 010, 011, 111 \} \quad / \text{T – True} /$$

Zbiór iloczynów (dla $y = 0$)

$$y = \prod (000, 001, 100, 101, 110) x_2 x_1 x_0$$

$$F_3 = \{ 000, 001, 100, 101, 110 \} \quad / \text{F – False} /$$

UKŁADY KOMBINACYJNE

Metody opisu działania układów kombinacyjnych – *cd.*

4. dziesiętna postać zbiorów iloczynów i sum

Zbiór sum (dla $y = 1$)

x_2 x_1 x_0	y
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	0
1 1 1	1

$$y = \sum (2, 3, 7) x_2 x_1 x_0$$

$$T_3 = \{ 2, 3, 7 \}$$

Zbiór iloczynów (dla $y = 0$)

$$y = \prod (0, 1, 4, 5, 6) x_2 x_1 x_0$$

$$F_3 = \{ 0, 1, 4, 5, 6 \}$$

UKŁADY KOMBINACYJNE

Metody opisu działania układów kombinacyjnych – *cd.*

5. siatka Karnaugh

x_0			
x_1		0	1
	0		
1			

Kod Gray'a
jednej zmiennej x_1

$x_1 x_0$					
x_2		00	01	11	10
	0				
1					

W tej kratce wpisujemy wartość
funkcji logicznej dla $x_2 x_1 x_0 = 101$

$x_1 x_0$					
$x_3 x_2$		00	01	11	10
	00				
01					
11					
10					

Kod Gray'a dwóch zmiennych x_3 i x_2

Liczba kratek jest równa 2^n

(n – liczba sygnałów wejściowych)

UKŁADY KOMBINACYJNE

Metody opisu działania układów kombinacyjnych – *cd.*

5. siatka Karnaugh – *cd.*

		x_0	
		0	1
x_1	0	0	1
	1	2	3

		$x_1 x_0$			
		00	01	11	10
x_2	0	0	1	3	2
	1	4	5	7	6

		$x_1 x_0$			
		00	01	11	10
$x_3 x_2$	00	0	1	3	2
	01	4	5	7	6
	11	12	13	15	14
	10	8	9	11	10

Oznaczenie:

$$x' \mapsto 0$$

$$x \mapsto 1$$

$$x'_3 x'_2 x'_1 x'_0 \mapsto 0000_2 \mapsto 0_{10}$$

$$x'_3 x'_2 x'_1 x_0 \mapsto 0001_2 \mapsto 1_{10}$$

...

$$x_3 x_2 x'_1 x'_0 \mapsto 1100_2 \mapsto 12_{10}$$

...

UKŁADY KOMBINACYJNE

Metody opisu działania układów kombinacyjnych – *cd.*

5. siatka Karnaugh – *cd.*

x_2	x_1	x_0	y
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	0
1	1	1	1

$x_1 x_0$					
		00	01	11	10
x_2	0	0	0	1	1
	1	0	0	1	0

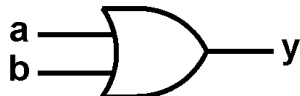
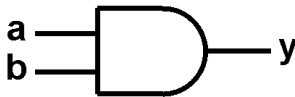
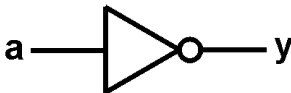
y

*Zapis funkcji y w postaci
tablicy prawdy i siatki Karnaugh*

UKŁADY KOMBINACYJNE

Bramki logiczne

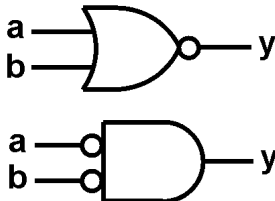
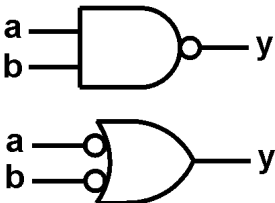

Bramki podstawowe

Nazwa bramki	Symbol graficzny	Funkcja logiczna	Tablica prawdy															
Suma (OR)		$y = a + b$	<table><tr><th>a</th><th>b</th><th>y</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	a	b	y	0	0	0	0	1	1	1	0	1	1	1	1
a	b	y																
0	0	0																
0	1	1																
1	0	1																
1	1	1																
Iloczyn (AND)		$y = a b$	<table><tr><th>a</th><th>b</th><th>y</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	a	b	y	0	0	0	0	1	0	1	0	0	1	1	1
a	b	y																
0	0	0																
0	1	0																
1	0	0																
1	1	1																
Negacja (NOT)		$y = \overline{x}$	<table><tr><th>a</th><th>y</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	a	y	0	1	1	0									
a	y																	
0	1																	
1	0																	

UKŁADY KOMBINACYJNE

Bramki logiczne – *cd.*

Bramki uniwersalne

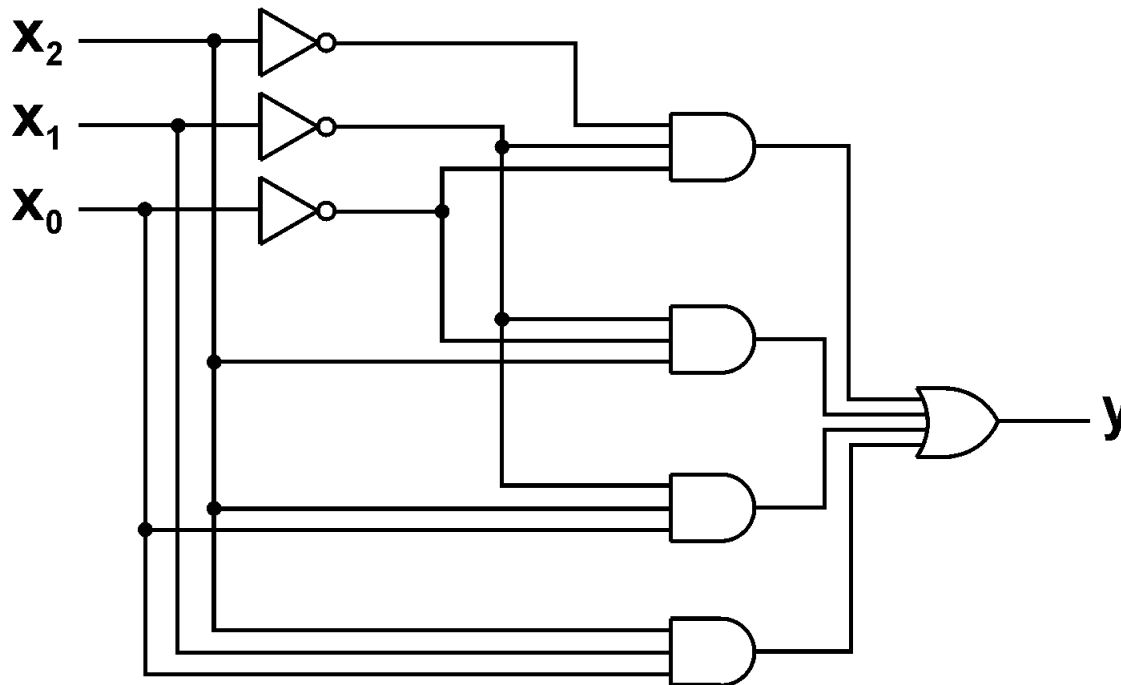
Nazwa bramki	Symbol graficzny	Funkcja logiczna	Tablica prawdy										
Negacja sumy (NOR)		$y = \overline{a + b} = \overline{a} \overline{b}$	<table><tr><th>a b</th><th>y</th></tr><tr><td>0 0</td><td>1</td></tr><tr><td>0 1</td><td>0</td></tr><tr><td>1 0</td><td>0</td></tr><tr><td>1 1</td><td>0</td></tr></table>	a b	y	0 0	1	0 1	0	1 0	0	1 1	0
a b	y												
0 0	1												
0 1	0												
1 0	0												
1 1	0												
Negacja iloczynu (NAND)		$y = \overline{a b} = \overline{a} + \overline{b}$	<table><tr><th>a b</th><th>y</th></tr><tr><td>0 0</td><td>1</td></tr><tr><td>0 1</td><td>1</td></tr><tr><td>1 0</td><td>1</td></tr><tr><td>1 1</td><td>0</td></tr></table>	a b	y	0 0	1	0 1	1	1 0	1	1 1	0
a b	y												
0 0	1												
0 1	1												
1 0	1												
1 1	0												
Suma modulo 2 (XOR)		$y = a \overline{b} + \overline{a} b = a \oplus b$	<table><tr><th>a b</th><th>y</th></tr><tr><td>0 0</td><td>0</td></tr><tr><td>0 1</td><td>1</td></tr><tr><td>1 0</td><td>1</td></tr><tr><td>1 1</td><td>0</td></tr></table>	a b	y	0 0	0	0 1	1	1 0	1	1 1	0
a b	y												
0 0	0												
0 1	1												
1 0	1												
1 1	0												

Bramki logiczne – układ kombinacyjny

Np.: $T_3 = \{0, 4, 5, 7\}$

$$y = x_2'x_1'x_0' + x_2x_1'x_0' + x_2x_1'x_0 + x_2x_1x_0$$

(postać kanoniczna
funkcji logicznej)



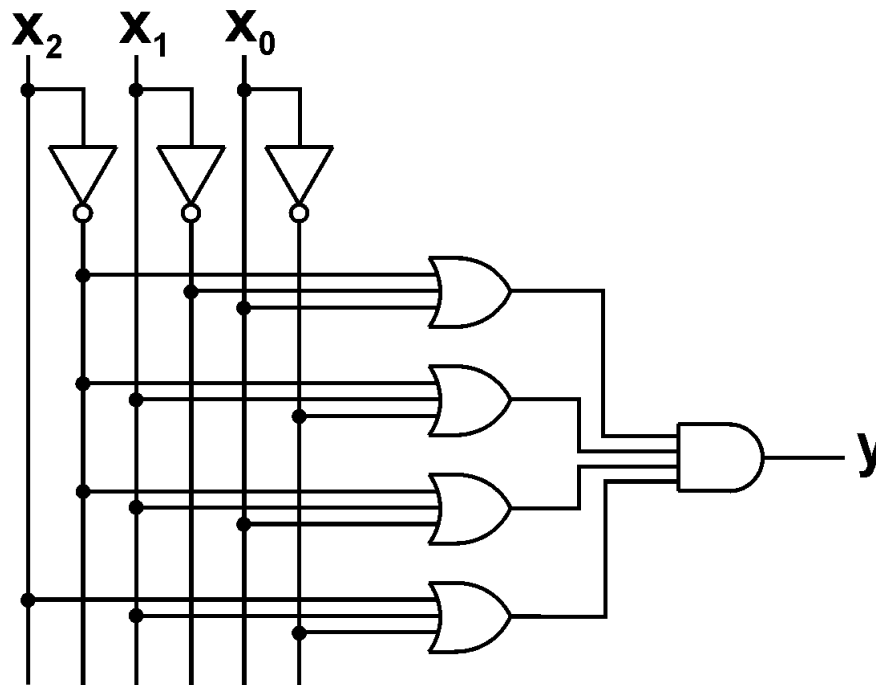
Rysujemy wejścia z lewej strony a wyjścia z prawej strony schematu!

Bramki logiczne – układ kombinacyjny

Np.: $T_3 = \{0, 4, 5, 7\} \Rightarrow F_3 = \{1, 2, 3, 6\}$

$$y = (x_2 + x_1 + x'_0)(x_2 + x'_1 + x_0)(x_2 + x'_1 + x'_0)(x'_2 + x'_1 + x_0)$$

(postać kanoniczna
funkcji logicznej)



UKŁADY KOMBINACYJNE

Minimalizacja funkcji logicznych

Celem minimalizacji jest zmniejszenie liczby bramek.

Stosujemy metodę w oparciu o prawo sklejanie algebry Boole'a:

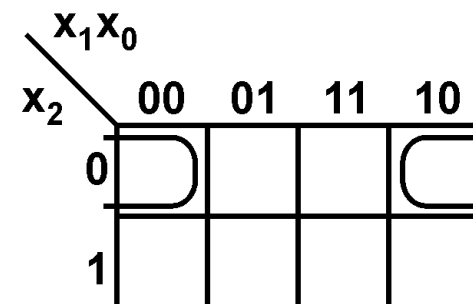
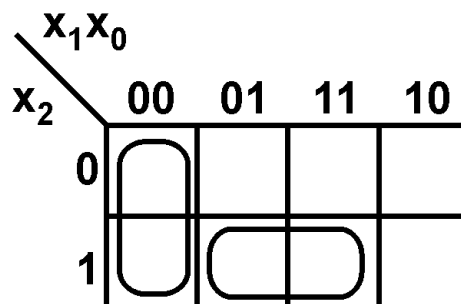
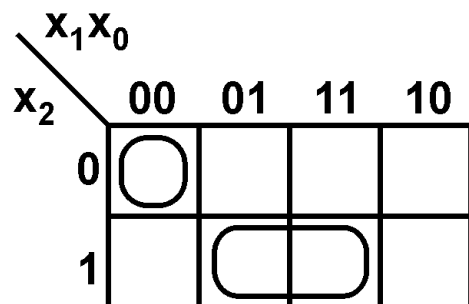
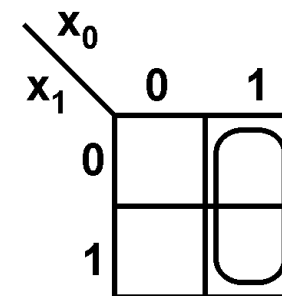
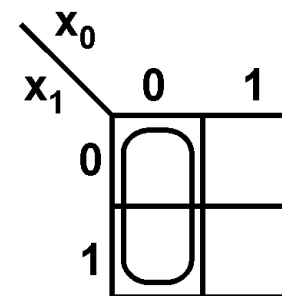
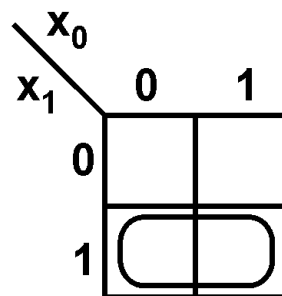
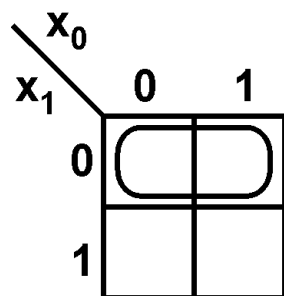
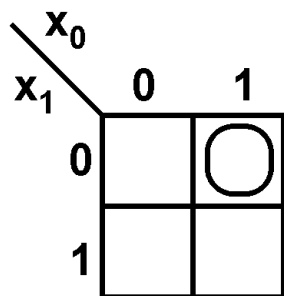
$$x_1 x_2 + x_1 \bar{x}_2 = x_1 (x_2 + \bar{x}_2) = x_1$$

> Siatki Karnaugh

Sklejamy (zakreślamy) taką liczbę krutek, która jest potęgą liczby 2, czyli 1, 2, 4, 8, 16, 32, itd.

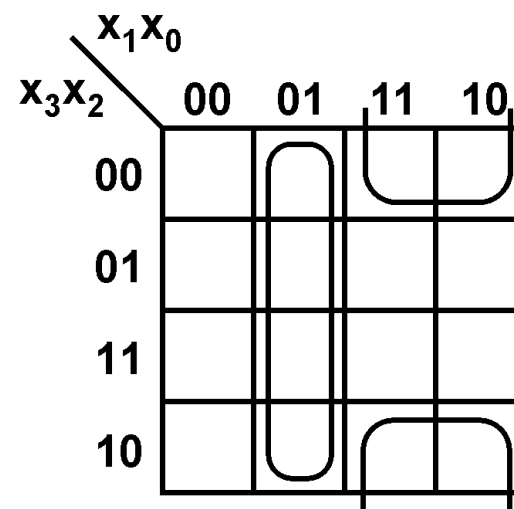
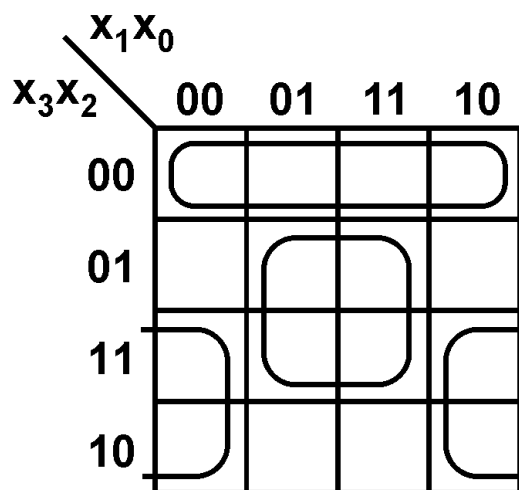
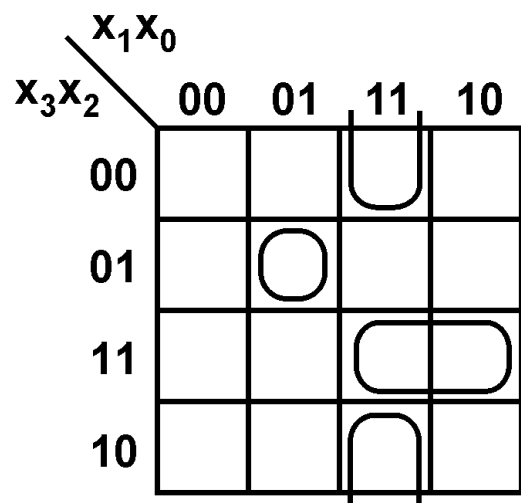
UKŁADY KOMBINACYJNE

Minimalizacja funkcji logicznych – *cd.*



UKŁADY KOMBINACYJNE

Minimalizacja funkcji logicznych – *cd.*



UKŁADY KOMBINACYJNE

Przykład

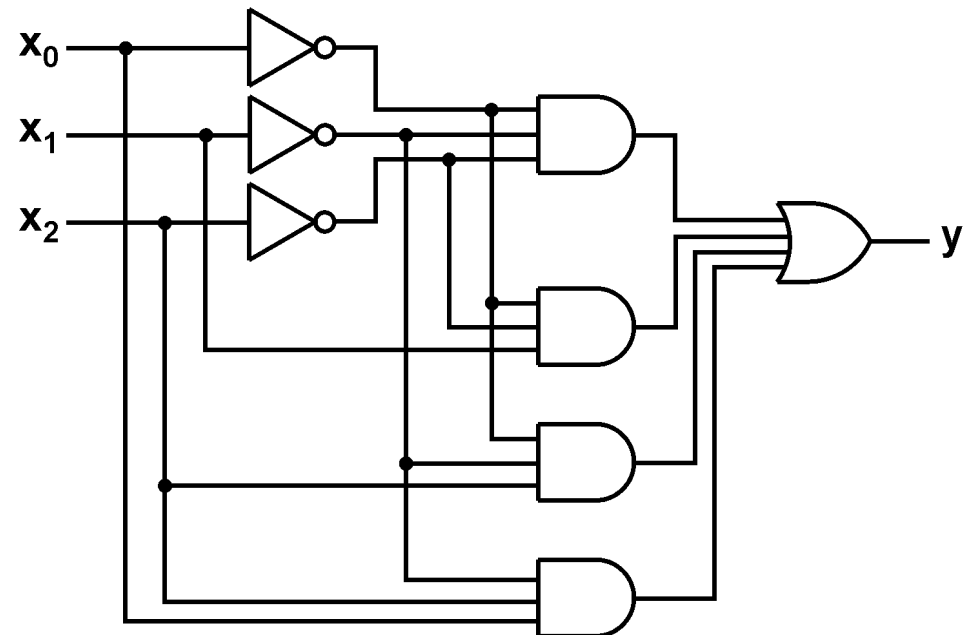
Dana jest funkcja logiczna $y = \Sigma(0, 2, 4, 5)x_2x_1x_0$

Postać sumacyjna (bez minimalizacji)

		x_1x_0			
		00	01	11	10
x_2	0	1	0	0	1
	1	1	1	0	0

y

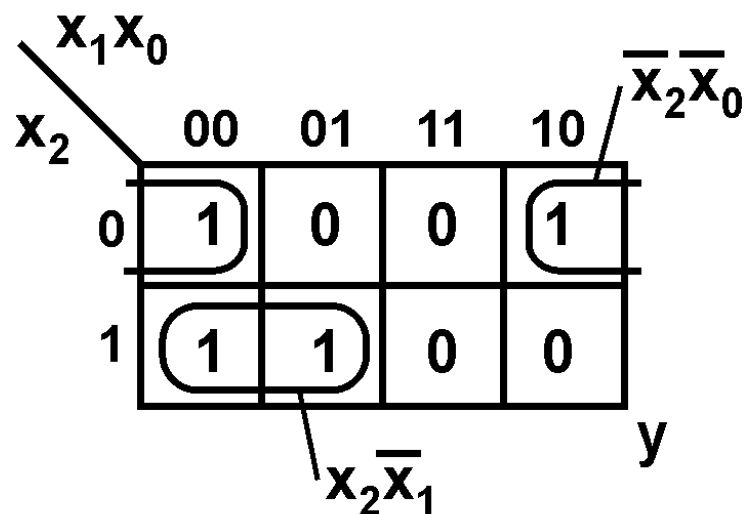
$$y = \bar{x}_2\bar{x}_1\bar{x}_0 + \bar{x}_2x_1\bar{x}_0 + x_2\bar{x}_1\bar{x}_0 + x_2\bar{x}_1x_0$$



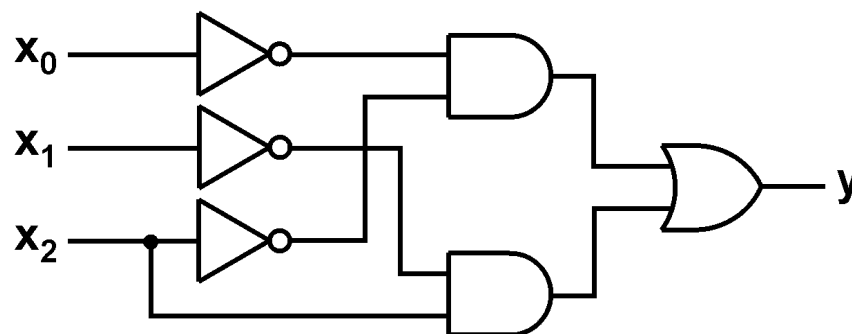
UKŁADY KOMBINACYJNE

Przykład – cd.

Minimalizacja „siatką Karnaugh”



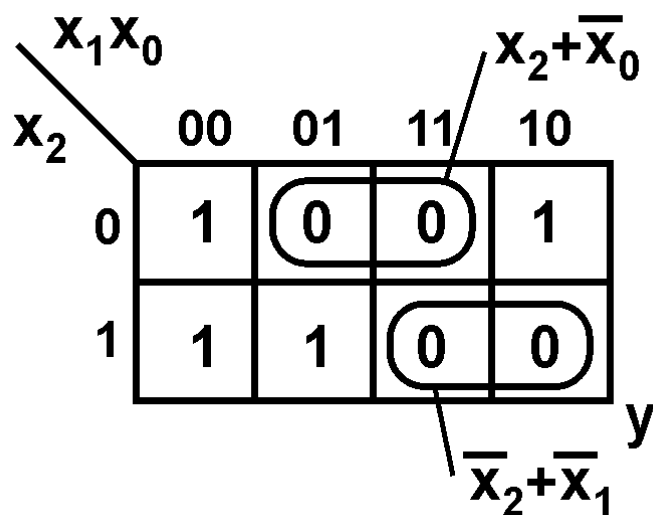
$$y = \bar{x}_2 \bar{x}_0 + x_2 \bar{x}_1$$



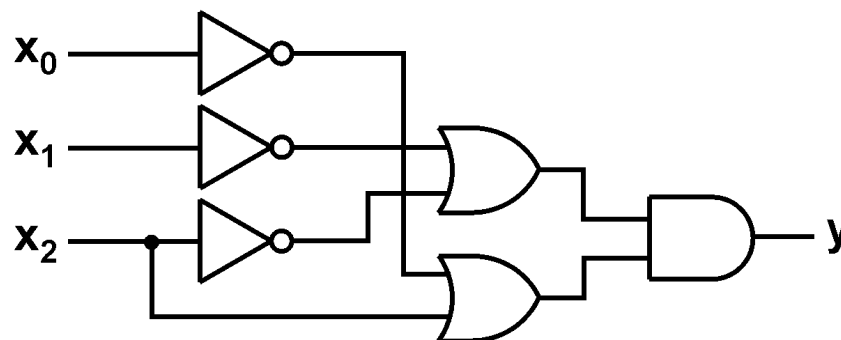
UKŁADY KOMBINACYJNE

Przykład – cd.

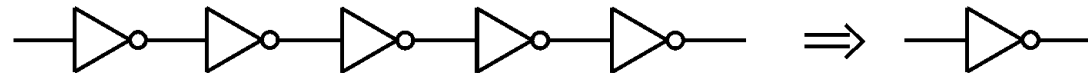
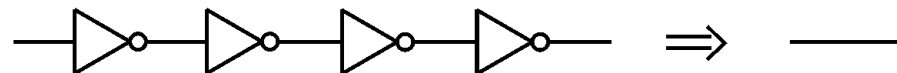
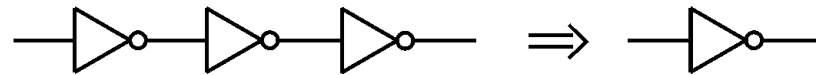
Minimalizacja „siatką Karnaugh”



$$y = (x_2 + \bar{x}_0) (\bar{x}_2 + \bar{x}_1)$$



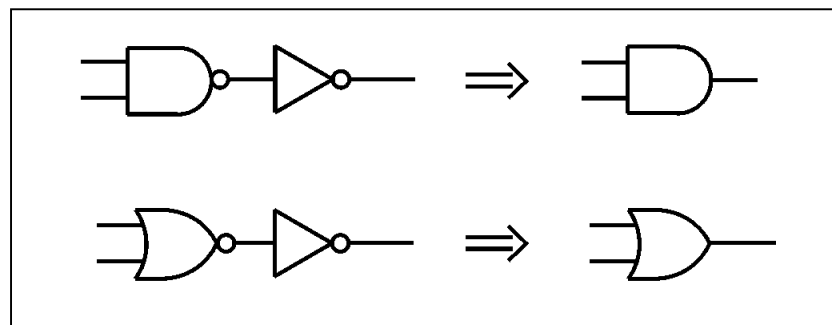
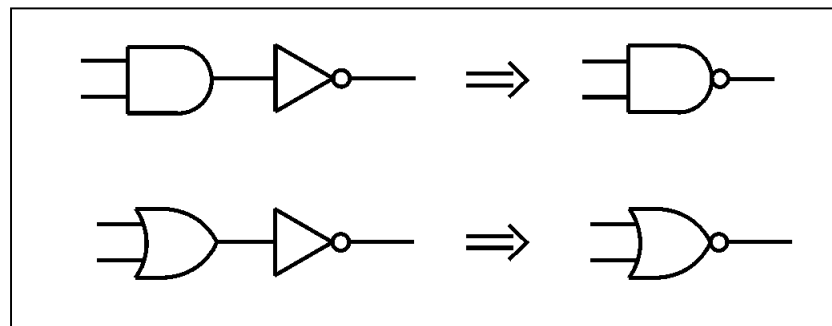
Bramka logiczna NOT



nieparzysta liczba negacji \rightarrow to pojedyncza negacja

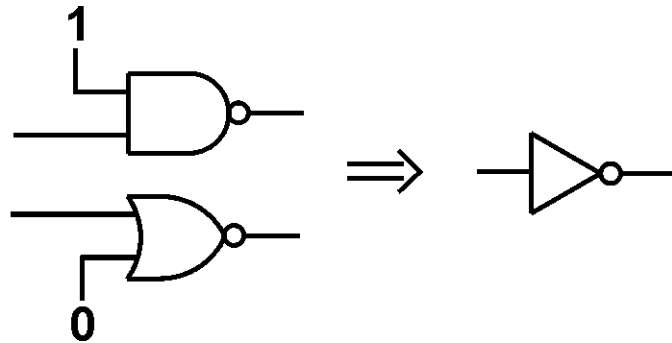
parzysta liczba negacji \rightarrow to brak negacji

Bramki logiczne NAND i NOR

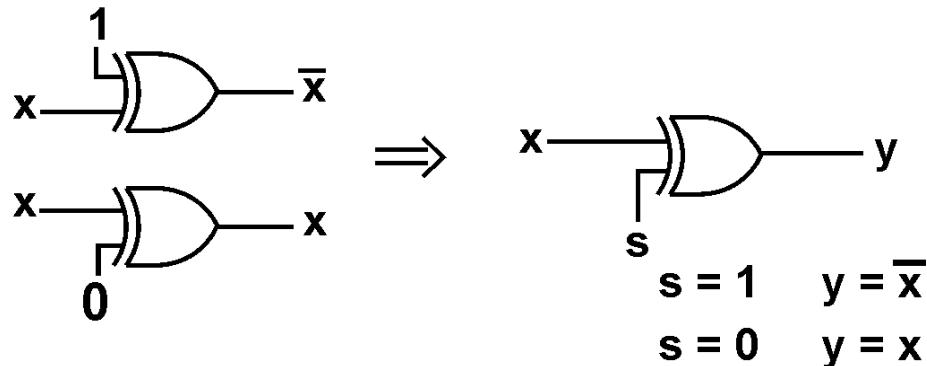


Bramki logiczne jako negatory

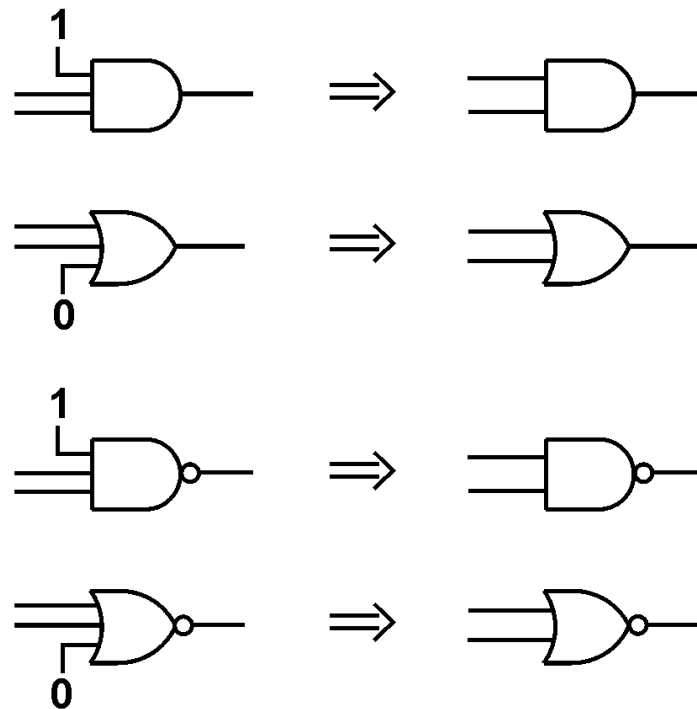
Negacja z bramek NAND i NOR



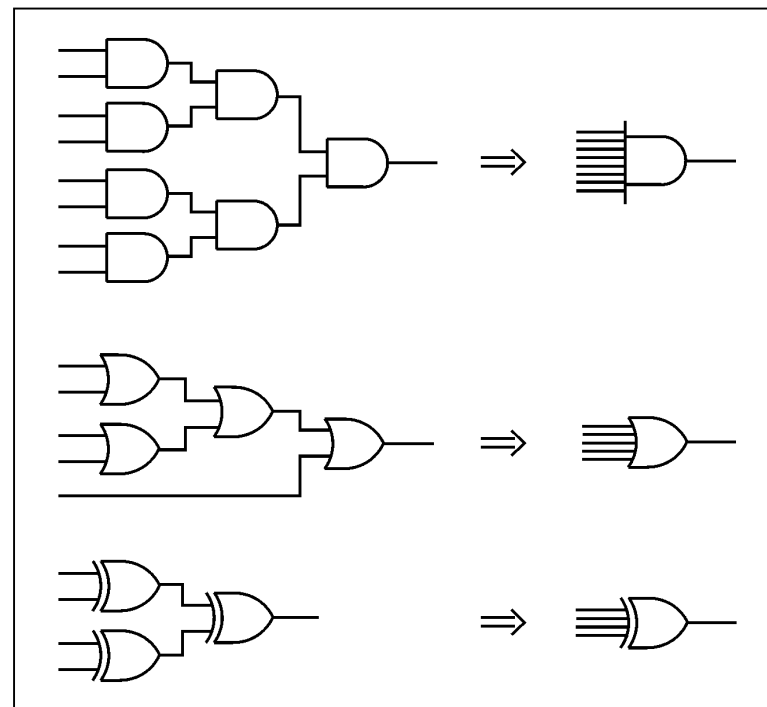
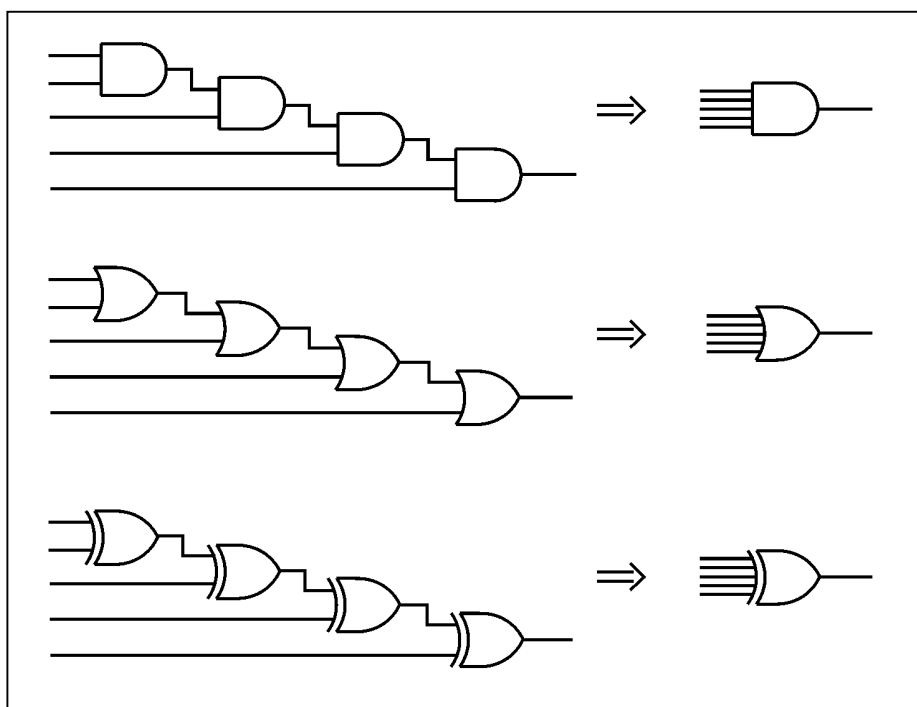
Bramka XOR jako sterowana negacja



Bramki logiczne – nie używane wejścia

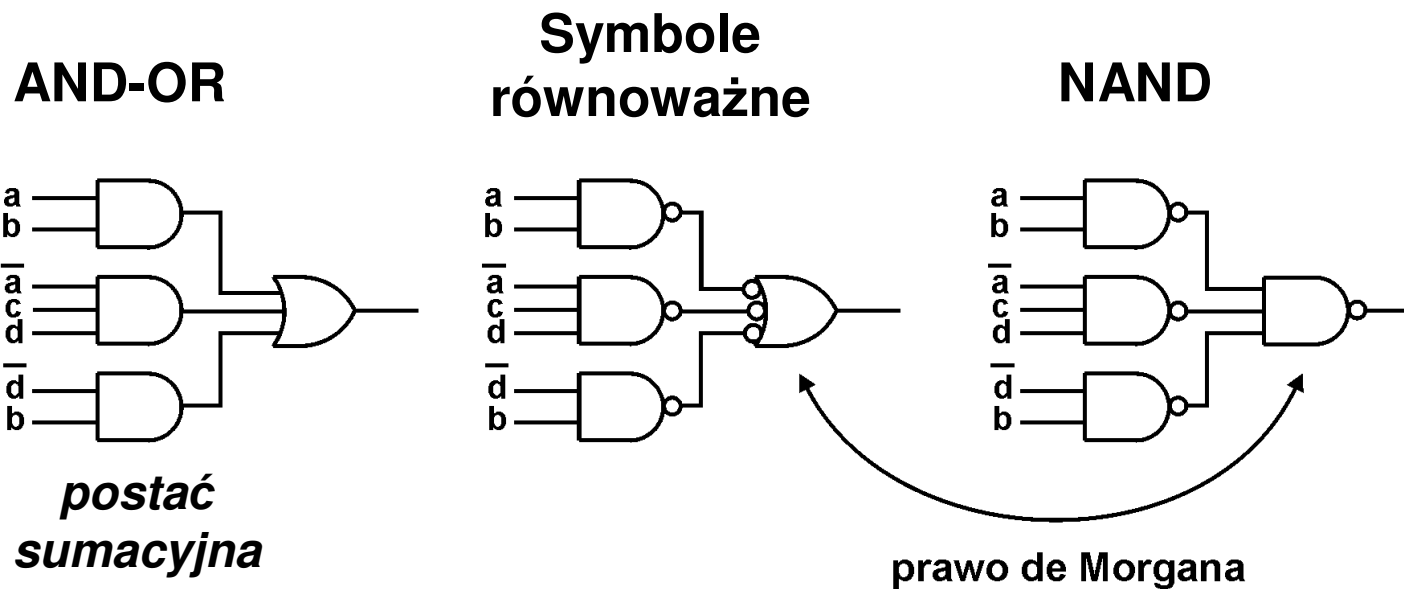


Bramki logiczne – zwiększenie liczby wejść funkcji AND, OR i XOR przy użyciu bramek dwuwejściowych



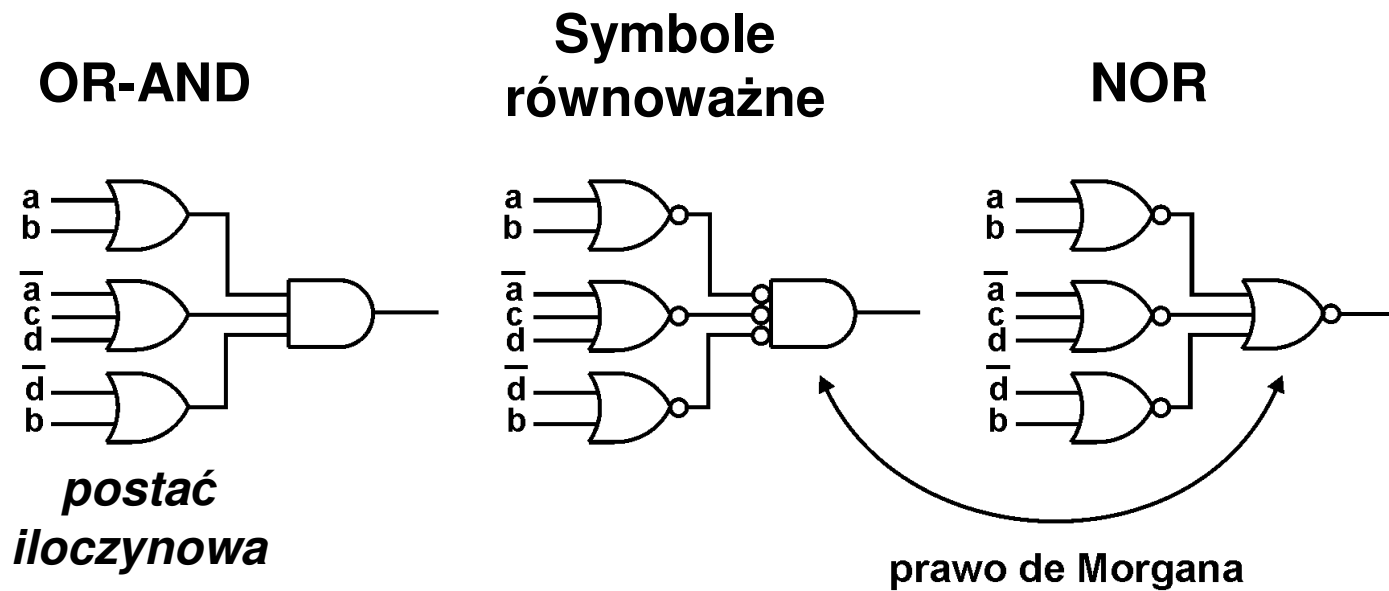
Bramki logiczne

Układy **AND-OR** można zamienić na równoważne struktury zrealizowane tylko z bramek **NAND**



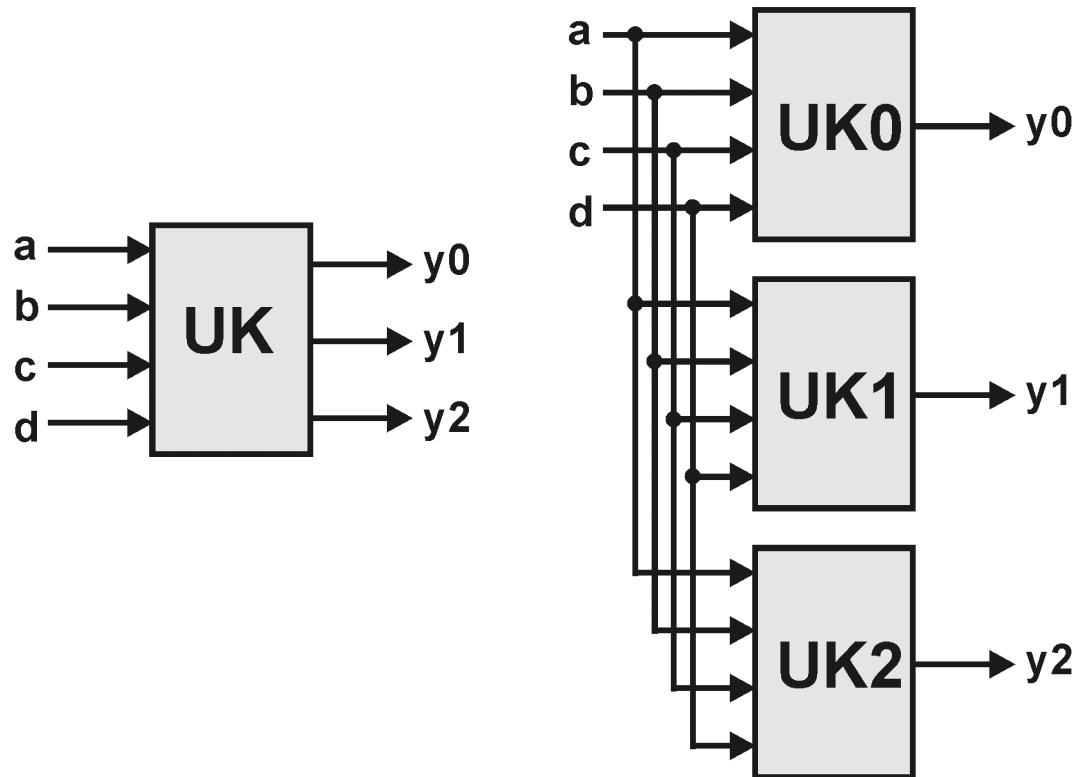
Bramki logiczne

Układy *OR-AND* można zamienić na równoważne struktury zrealizowane tylko z bramek *NOR*

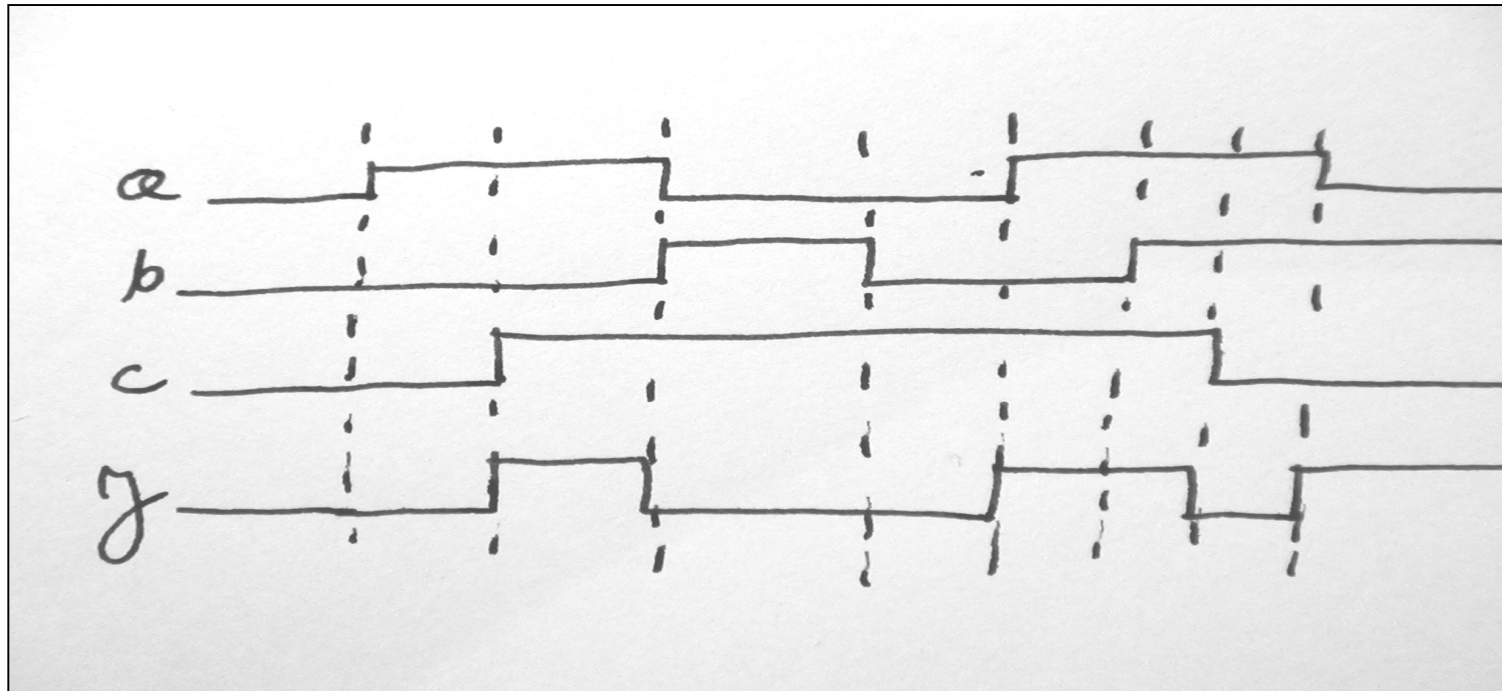


UKŁADY WIELOWYJŚCIOWE

Należy wykonać syntezę dla każdego wyjścia osobno.



Rysowanie przebiegów czasowych ilustrujących działanie układu kombinacyjnego

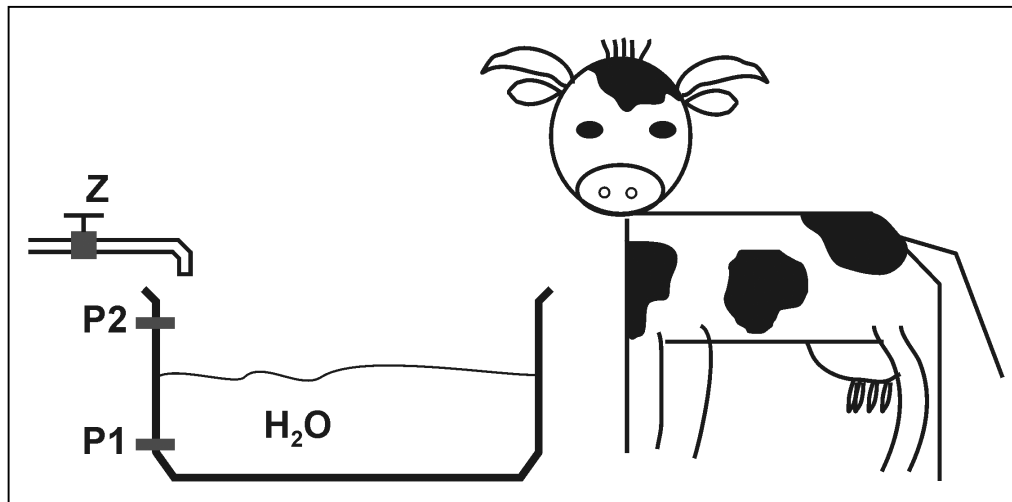


**W zakresie stałych wartości sygnałów wejściowych
określamy wartości sygnałów wyjściowych**

PRZYKŁAD SYNTEZY

KROWODOPÓJ – nalewanie wody do koryta gdy jej poziom jest niski, koniec nalewania gdy poziom wody osiągnie odpowiednią wartość...

P1, P2 – czujniki poziomu wody
Z – zawór



Stany układu:

P1=0 – woda poniżej P1

P1=1 – woda powyżej P1

P2=0 – woda poniżej P2

P2=1 – woda powyżej P2

Z=0 – zawór zamknięty

Z=1 – zawór otwarty

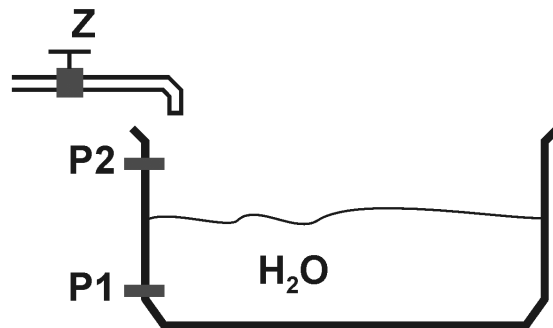
Możliwe sytuacje → Działanie:

$P1=0, P2=0$ – woda poniżej $P1$ → nalewanie wody $Z=1$

$P1=1, P2=0$ – woda pomiędzy $P1$ a $P2$ → nalewanie wody gdy koryto było puste ($Z=1$) albo woda była nalana do pełna ($Z=0$)

$P1=1, P2=1$ – pełne koryto → koniec nalewania, zamykamy zawór $Z=0$

$P1=0, P2=1$ – sytuacja nie możliwa! → brak reakcji układu sterującego

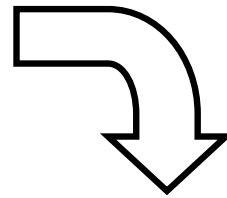


		P2 P1			
Z		00	01	11	10
	0	1	0	0	—
	1	1	1	0	—

Z

„ — ” oznacza dowolny stan logiczny

P2 P1					
Z		00	01	11	10
	Z				
0		1	0	0	—
1		1	1	0	—



$$Z = Z \cdot \overline{P2} + \overline{P1}$$

