

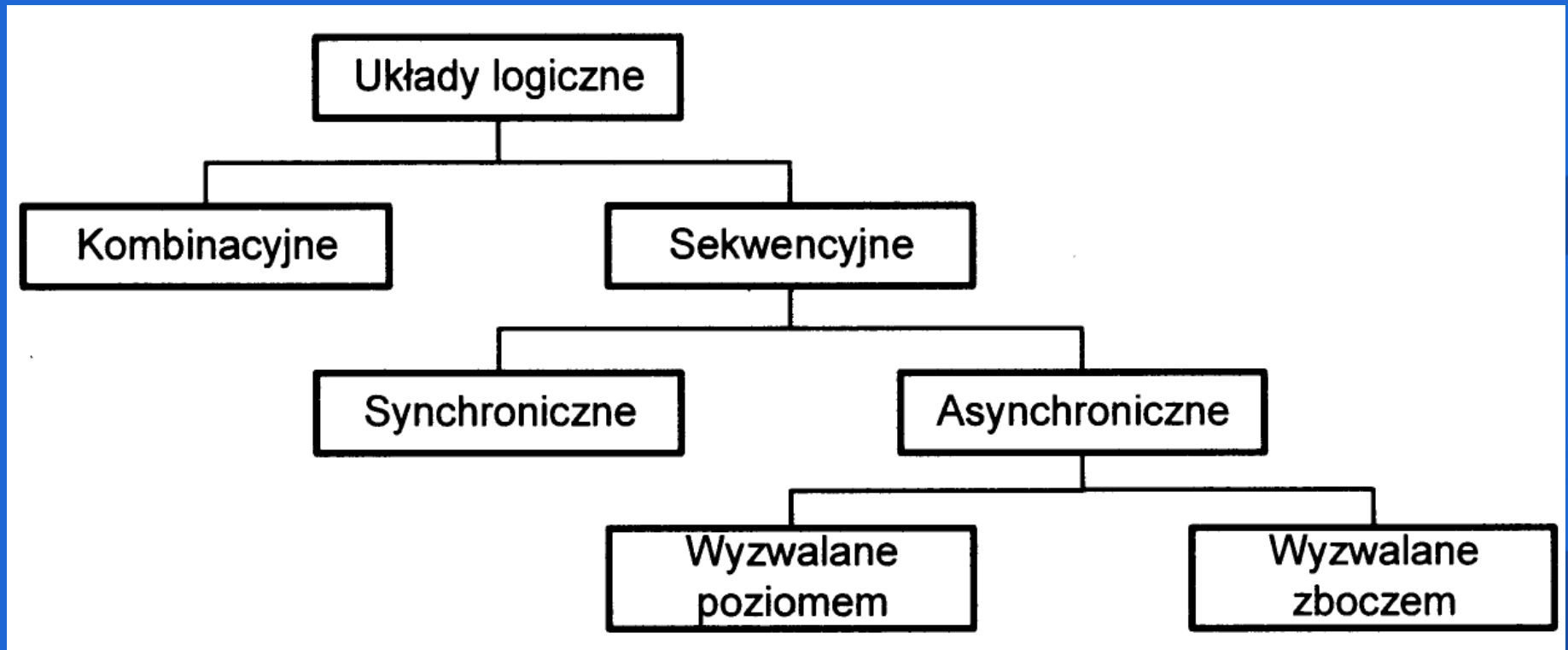
Sławomir Kulesza

Technika cyfrowa Synteza układów kombinacyjnych

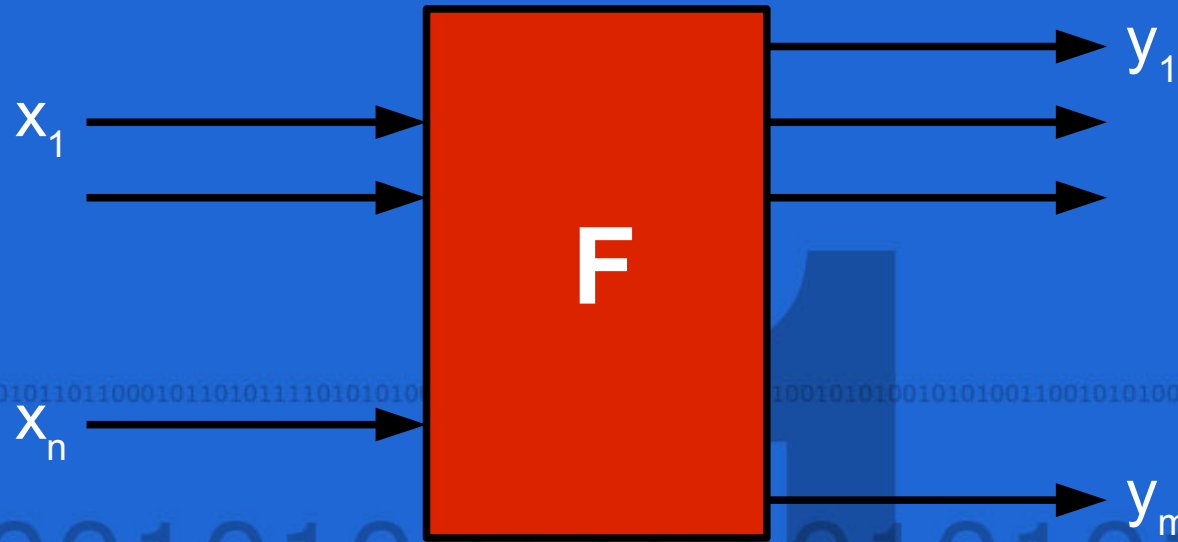
Wykład dla studentów III roku Informatyki

Wersja 2.0, 05/10/2011

Podział układów logicznych



Opis funkcjonalny układów logicznych



$x_i, y_j \in \mathbf{B}$ – sygnały binarne

$X = (x_1, x_2, \dots, x_n)$ – wektor wejściowy

$Y = (y_1, y_2, \dots, y_m)$ – wektor wyjściowy

X_k – k-ty stan wejściowy (konkretna postać X)

Y_j – j-ty stan wyjściowy (konkretna postać Y)

Stany układu logicznego

Zbiór **X** występujących stanów układu X_j jest zawsze podzbiorem B^n i zawiera maksymalnie $N = 2^n$ elementów (stanów układu).

Numeracja stanów X_j : $j \in \{0, 1, 2, \dots, N - 1\}$, np.:
 $n = 2$, $\mathbf{X} = \{X_0, X_1, X_2, X_3\} = \{00, 01, 10, 11\}$.

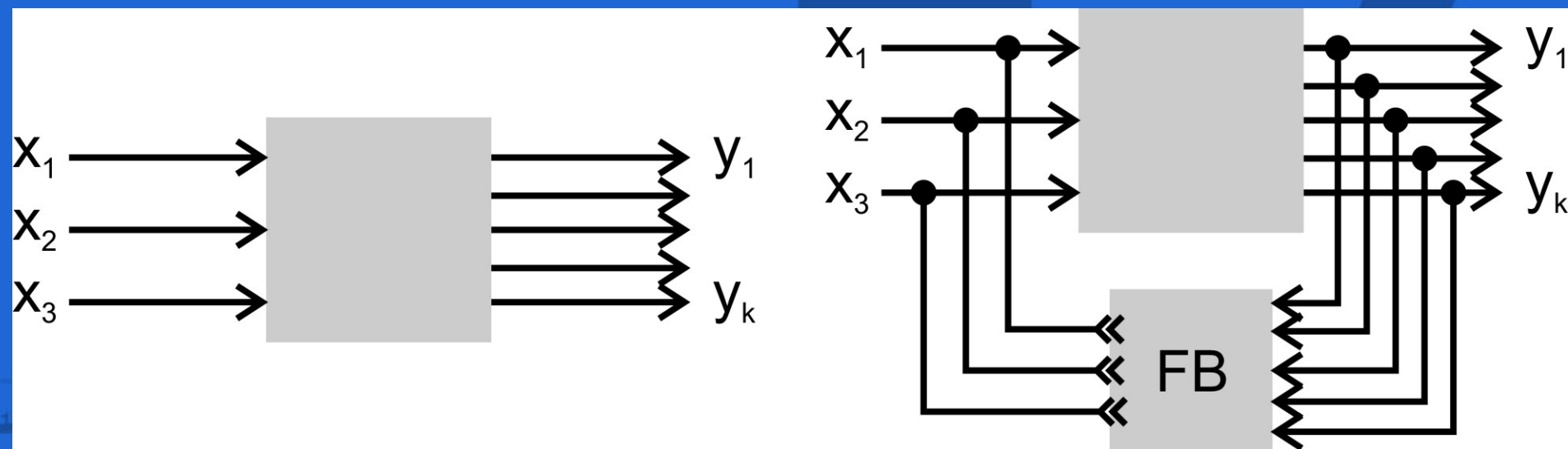
W realnych układach: $\mathbf{X} \subset B^n$, $\mathbf{Y} \subset B^m$

Układy kombinacyjne a sekwencyjne

Układy kombinacyjne: $Y^t = F(X^t)$

Układy sekwencyjne: $Y^t = F(X^t, X^{t-1}, X^{t-2}, \dots X^{t-p})$

1010010100110101010101101100010110101111010101001010010101010010010101010010101001010100110010101001



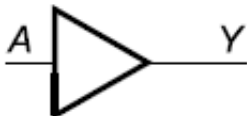
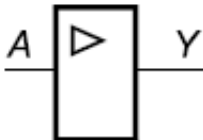
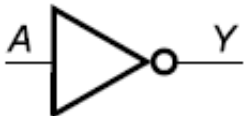
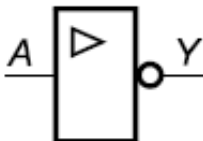

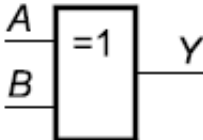

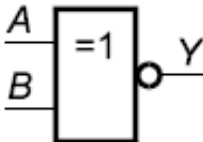
Funkcja przełączająca

Jeśli w dowolnej chwili t stan Y^t układu logicznego zależy wyłącznie od stanu X^t , czyli $Y^t = f(X^t)$, to jest to układ kombinacyjny (bez pamięci), opisywany funkcją $f: X \rightarrow Y$.


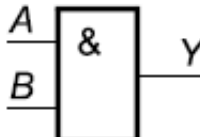

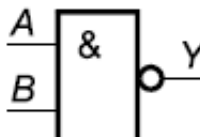

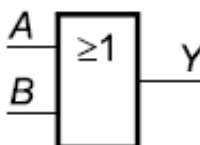

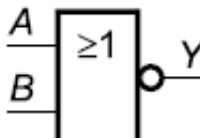
Wartość każdego sygnału wyjściowego y_i zależy od stanu wejść X i funkcji logicznej (boolowskiej, przełączającej) f_i układu:

$$y_i = f_i(X)$$

Symbole bramek logicznych

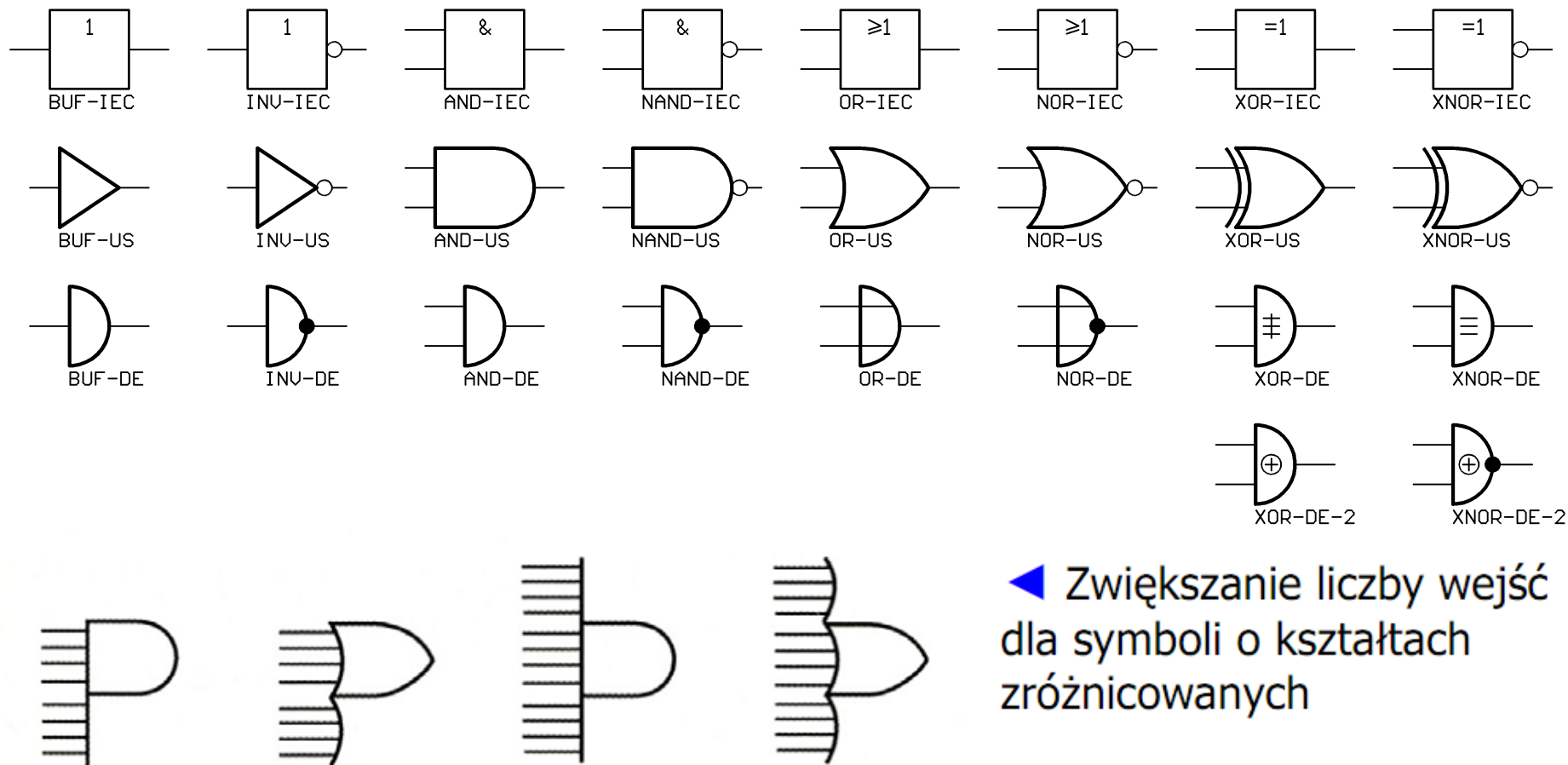
Nazwa elementu ang. / pl.	symbol ANSI/IEEE	symbol IEC, ANSI/IEEE	Tabela prawdy															
BUFFER BUFOR			<table><tr><th>A</th><th>Y = A</th></tr><tr><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td></tr></table>	A	Y = A	0	0	1	1									
A	Y = A																	
0	0																	
1	1																	
NOT NIE			<table><tr><th>A</th><th>Y = \bar{A}</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	A	Y = \bar{A}	0	1	1	0									
A	Y = \bar{A}																	
0	1																	
1	0																	
EXOR ALBO, WYŁĄCZNIE LUB			<table><tr><th>A</th><th>B</th><th>Y = $A \oplus B$</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>0</td></tr></table>	A	B	Y = $A \oplus B$	0	0	0	0	1	1	1	0	1	1	1	0
A	B	Y = $A \oplus B$																
0	0	0																
0	1	1																
1	0	1																
1	1	0																
EXNOR ALBO-NIE			<table><tr><th>A</th><th>B</th><th>Y = $A \otimes B$</th></tr><tr><td>0</td><td>0</td><td>1</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 = $A \otimes B$	0	0	1	0	1	0	1	0	0	1	1	1
A	B	Y = $A \otimes B$																
0	0	1																
0	1	0																
1	0	0																
1	1	1																

Symbole bramek logicznych

Nazwa elementu ang. / pl.	symbol ANSI/IEEE	symbol IEC, ANSI/IEEE	Tabela prawdy															
AND I			<table><tr><th>A</th><th>B</th><th>$Y = A \cdot B$</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	$Y = A \cdot B$	0	0	0	0	1	0	1	0	0	1	1	1
A	B	$Y = A \cdot B$																
0	0	0																
0	1	0																
1	0	0																
1	1	1																
NAND I-NIE			<table><tr><th>A</th><th>B</th><th>$Y = \overline{A \cdot B}$</th></tr><tr><td>0</td><td>0</td><td>1</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>0</td></tr></table>	A	B	$Y = \overline{A \cdot B}$	0	0	1	0	1	1	1	0	1	1	1	0
A	B	$Y = \overline{A \cdot B}$																
0	0	1																
0	1	1																
1	0	1																
1	1	0																
OR, LUB			<table><tr><th>A</th><th>B</th><th>$Y = A + B$</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	$Y = A + B$	0	0	0	0	1	1	1	0	1	1	1	1
A	B	$Y = A + B$																
0	0	0																
0	1	1																
1	0	1																
1	1	1																
NOR, LUB-NIE			<table><tr><th>A</th><th>B</th><th>$Y = \overline{A + B}$</th></tr><tr><td>0</td><td>0</td><td>1</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>0</td></tr></table>	A	B	$Y = \overline{A + B}$	0	0	1	0	1	0	1	0	0	1	1	0
A	B	$Y = \overline{A + B}$																
0	0	1																
0	1	0																
1	0	0																
1	1	0																

Symbole bramek logicznych

(inne standardy)



◀ Zwiększanie liczby wejść
dla symboli o kształtach
zróżnicowanych

Formy boolowskie

Forma boolowska w analityczny sposób przedstawia funkcję przełączającą układu wykorzystując sygnały (zmiennne boolowskie), stałe 0/1 oraz operatory boolowskie AND / OR / NOT.



Forma sumacyjna:
(suma iloczynów literałów)

$$x_1 \cdot x_3 + \bar{x}_1 \cdot x_2 \cdot \bar{x}_3$$

Forma iloczynowa:
(iloczyn sum literałów)

$$(x_1 + x_2 + x_3) \cdot (\bar{x}_1 + x_2)$$

Równoważność form boolowskich

Dwie formy boolowskie w_1 , w_2 są równoważne, gdy określają tę samą funkcję przełączającą f , tzn.:

$$\forall_{x \in x} (w_1(X) = f(X)) \wedge (w_2(X) = f(X))$$

Każda funkcja przełączająca może być przedstawiona przez nieskończenie wiele równoważnych form boolowskich, np.:

$$f(X) = X, w_1(X) = X \vee X, w_2(X) = X \wedge X, \text{ itd..}$$

Mnogość form boolowskich

Minimalizacja logiczna:

- redukcja liczby literałów (wejść bramek),
- redukcja liczby bramek,
- redukcja liczby poziomów bramek.

- 1) Technologie bramek cyfrowych ograniczają liczbę wejść sygnałowych,
- 2) Mniejsza liczba bramek to: mniejsze opóźnienia propagacji sygnałów, niższa moc zasilania, niższe zakłócenia, prostszy układ itd.
- 3) Liczba użytych bramek wpływa na koszt wytworzenia układu.

Wzór Shannona

Uogólnienie prawa De Morgana na formy boolowskie:

$$f'(X, \cdot, +) = f(X', +, \cdot)$$

Obowiązuje przy tym kolejność działań:
(1) nawiasy, (2) negacje, (3) iloczyny, (4) sumy

Wzór Shannona pozwala m.in. konwertować formy sumacyjne na iloczynowe i *vice versa*.

Przykład konwersji

Niech: $f = x_1 \cdot x_2 + x_1'$ ← forma sumacyjna

Wówczas:

$$f' = (x_1 \cdot x_2 + x_1')' = (x_1' + x_2') \cdot x_1 = x_1' \cdot x_1' + x_1' \cdot x_2' = x_1' \cdot x_2'$$

Ponadto:

$$f = (f')' = (x_1' \cdot x_2')' = x_1' + x_2' \leftarrow \text{forma iloczynowa}$$

Tablica prawdy

Tablica prawdy jest macierzową reprezentacją funkcji przełączającej. W przypadku funkcji n -zmiennych $y = f(x_1, \dots, x_n)$ zawiera ona 2^n -wierszy i $(n+1)$ -kolumn.

Liczba wszystkich funkcji przełączających n -zmiennych wynosi 4^n , np. dla $n = 1$:

$$(1) f_1(a) = a,$$

$$(2) f_2(a) = a',$$

$$(3) f_3(a) = 0,$$

$$(4) f_4(a) = 1.$$

$y = a$		$y = a'$		$y = 1$		$y = 0$	
a	y	a	y	a	y	a	y
0	0	0	1	0	1	0	0
1	1	1	0	1	1	1	0

Funkcje logiczne dwóch zmiennych

x_1	x_2	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
Symbol operatora		\cdot \wedge	Δ /			Δ /		\oplus	\vee +	\downarrow	\odot	$-$ /	\Rightarrow \supset	$-$ /	\Rightarrow \supset	\mid	

NOR

NAND

Funkcje logiczne dwóch zmiennych

Wyrażenie boolowskie	Zastosowanie operatorów nieboolowskich	Nazwa operatora funkcji		Nazwa funkcji
		polska	angielska	
$f_0 = 0$				stała 0
$f_1 = x_1 x_2$		I	AND	iloczyn
$f_2 = x_1 \bar{x}_2$	$x_1 \Delta x_2$			iloczyn z zakazem
$f_3 = x_1$				przeniesienie x_1
$f_4 = \bar{x}_1 x_2$	$x_2 \Delta x_1$			iloczyn z zakazem
$f_5 = x_2$				przeniesienie x_2
$f_6 = \bar{x}_1 x_2 \vee x_1 \bar{x}_2$	$x_1 \oplus x_2$ ($x_1 \neq x_2$)	ALBO	EX-OR, XOR	funkcja nierówności
$f_7 = x_1 \vee x_2$		LUB	OR	suma
$f_8 = \bar{x}_1 \vee \bar{x}_2$	$x_1 \downarrow x_2$	LUB-NIE (skrót LUN)	NOR	funkcja (strzałka) Peirce'a
$f_9 = \bar{x}_1 \bar{x}_2 \vee x_1 x_2$	$x_1 \odot x_2 = \overline{x_1 \oplus x_2}$ ($x_1 = x_2$)	ALBO-NIE (skrót ALBON)	EX-NOR, XNOR EQUALITY, EQUIVALENCE	funkcja równości
$f_{10} = \bar{x}_2$		NIE (skrót N)	NOT	negacja
$f_{11} = x_1 \vee \bar{x}_2$	$x_2 \Rightarrow x_1$			implikacja: jeśli x_2 , to x_1
$f_{12} = \bar{x}_1$		NIE (skrót N)	NOT	negacja
$f_{13} = \bar{x}_1 \vee x_2$	$x_1 \Rightarrow x_2$			implikacja: jeśli x_1 , to x_2
$f_{14} = \overline{x_1 x_2}$	$x_1 x_2$	I-NIE (skrót IN)	NAND	funkcja (kreska) Sheffera
$f_{15} = 1$				stała 1

Podstawowe funkcje logiczne dwóch zmiennych

		AND	NAND	OR	NOR	XOR	XNOR
		$f(a,b) =$					
a	b	ab	$(ab)'$	$a + b$	$(a + b)'$	$a \oplus b$	$(a \oplus b)'$
0	0	0	1	0	1	0	1
0	1	0	1	1	0	1	0
1	0	0	1	1	0	1	0
1	1	1	0	1	0	0	1

Funkcja logiczna XOR

Funkcja XOR funkcjonuje także pod nazwami:
(1) wyłącznie-lub, (2) suma wyłączająca, (3) suma modulo 2, (4) różnica symetryczna, (5) funkcja nieparzystości:

$$f(x_1, x_2) \stackrel{\text{def}}{=} x_1 \oplus x_2 = (x_1 \neq x_2)$$

$$f(x_1, x_2) = x_1 \oplus x_2 = (x_1 + x_2)(x_1' + x_2') = x_1'x_2 + x_1x_2'$$

Podstawowe związki:

$$x_1 \oplus 0 = x_1,$$

$$x_1 \oplus 1 = x_1'$$

$$x_1 \oplus x_1 = 0,$$

$$x_1 \oplus x_1' = 1$$

$$(x_1 \oplus x_2 = x_3) \Rightarrow (x_1 \oplus x_3 = x_2) \wedge (x_2 \oplus x_3 = x_1)$$

Funkcja logiczna XNOR

Funkcja XNOR funkcjonuje także pod nazwami:

(1) wyłącznie-lub-nie, (2) funkcja komparacji,
(3) funkcja równoważności, (4) funkcja parzystości:

$$f(x_1, x_2) \stackrel{\text{def}}{=} x_1 \odot x_2 = (x_1 = x_2)$$

$$f(x_1, x_2) = (x_1 \oplus x_2)' = x_1 x_2 + x_1' x_2' = (x_1' + x_2)(x_1 + x_2')$$

Podstawowe związki:

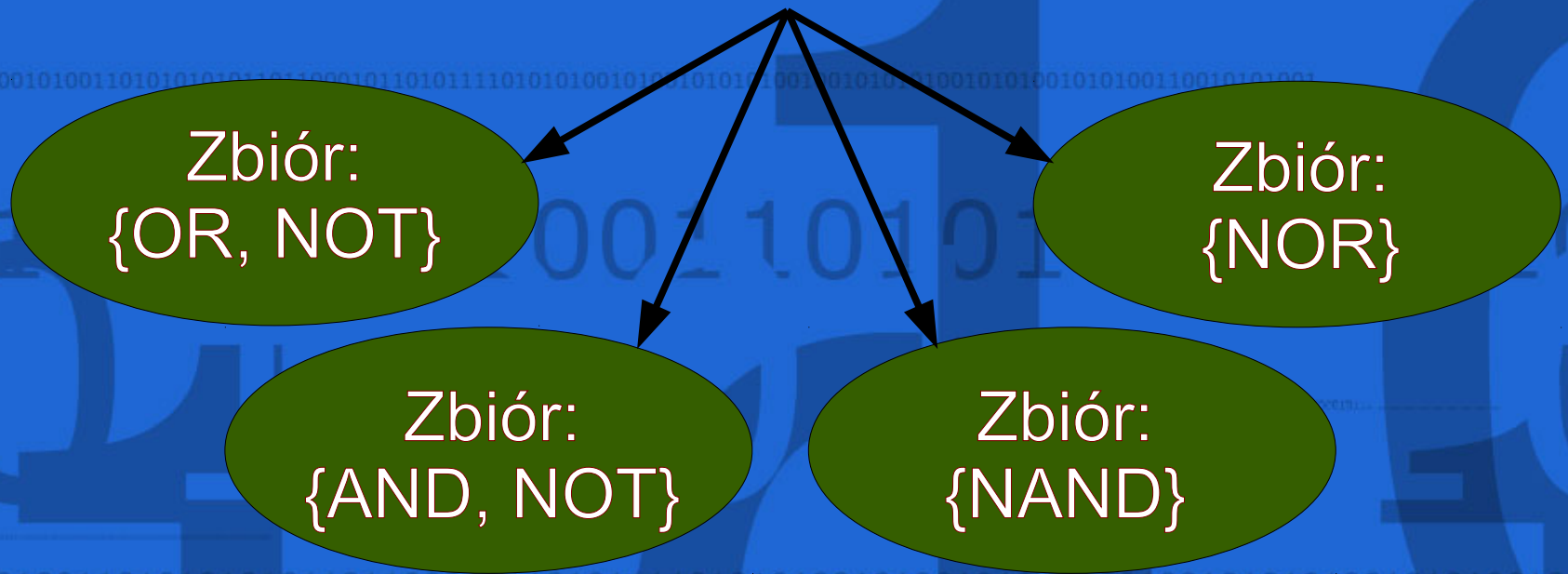
$$x_1 \odot x_1 = 1,$$

$$x_1 \odot x_1' = 0$$

$$x_1 \odot x_2 = x_1' \oplus x_2 = x_1 \oplus x_2'$$

Systemy funkcjonalnie pełne

System funkcjonalnie pełny (SFP) to każdy minimalny zbiór operatorów pozwalających zrealizować dowolną funkcję przełączającą przy użyciu tylko operatorów z tego zbioru.

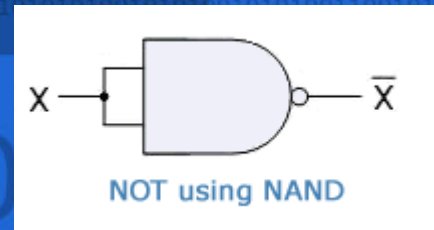


Systemy funkcjonalnie pełne

Zrealizujemy funkcje NOT oraz OR wyłącznie za pomocą bramek NAND:

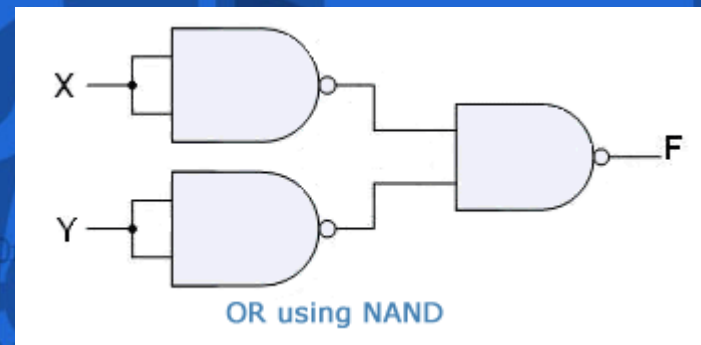
1) NOT:

$$x' = (x \cdot x)'$$



2) OR:

$$x+y = ((x+y)')' = (x' \cdot y')'$$



Formy boolowskie – definicje

Literał – symbol zmiennej lub jej negacji, np.: x_1 , x_2' ; dla n-zmiennych maksymalnie 2n-literałów.

Term iloczynowy – iloczyn literałów, np.: $x_1'x_2$.

Term sumacyjny – suma literałów, np.: x_1+x_2' .

Term pełny funkcji n-zmiennych – term iloczynowy lub sumacyjny zawierający n różnych literałów:

- minterm – term iloczynowy pełny (np.: $x_1x_2'x_3$),
- maksterm – term sumacyjny pełny (np.: $x_1+x_2'+x_3$).

Wartościowanie mintermów

Niech dany będzie uporządkowany minterm $P_n(X)$:

$$P_n(X) = x_1 x_2 x_3 \dots x_k$$

Przyporządkujemy mu liczbę binarną m w ten sposób, że: $x_n \rightarrow 1$, $x_n' \rightarrow 0$, np.:

$$x_1 x_2 x_3' \rightarrow m = 110_2 = 6_{10}$$

Wartość logiczna mintermu: $P_n(X_m) = 1 \Leftrightarrow (n=m)$

Wartościowanie makstermów

Niech dany będzie uporządkowany maksterm $S_n(X)$:

$$S_n(X) = x_1 + x_2 + x_3 + \dots + x_k$$

Przyporządkujemy mu liczbę binarną s_n w ten sposób, że: $x_n \rightarrow 0$, $x_n' \rightarrow 1$, np.:

$$x_1 + x_2 + x_3' \rightarrow s_1 = 001_2 = 1_{10}$$

Wartość logiczna makstermu: $S_n(X_m) = 0 \Leftrightarrow (n=m)$

Konwersja wartościowania minterm \Leftrightarrow maksterm

Korzystając ze wzoru Shannona dowodzi się, że:

$$P_n(X_k) = (S_n(X_k))'$$

np.: dla $P_4(X) = x_1 \cdot x_2' \cdot x_3'$ oraz $S_4(X) = x_1' + x_2 + x_3$

$$P_4(4) = P_4(100) = 1 \cdot 0' \cdot 0' = 1 \cdot 1 \cdot 1 = 1$$

$$P_4(3) = P_4(011) = 0 \cdot 1' \cdot 1' = 0 \cdot 0 \cdot 0 = 0$$

$$S_4(4) = S_4(100) = 1' + 0 + 0 = 0 + 0 + 0 = 0$$

$$S_4(3) = S_4(011) = 0' + 1 + 1 = 1 + 1 + 1 = 1$$

Makstermy i mintermy w tablicy prawdy

<i>Row</i>	<i>X</i>	<i>Y</i>	<i>Z</i>	<i>F</i>	<i>Minterm</i>	<i>Maxterm</i>
0	0	0	0	F(0,0,0)	$X' \cdot Y' \cdot Z'$	$X + Y + Z$
1	0	0	1	F(0,0,1)	$X' \cdot Y' \cdot Z$	$X + Y + Z'$
2	0	1	0	F(0,1,0)	$X' \cdot Y \cdot Z'$	$X + Y' + Z$
3	0	1	1	F(0,1,1)	$X' \cdot Y \cdot Z$	$X + Y' + Z'$
4	1	0	0	F(1,0,0)	$X \cdot Y' \cdot Z'$	$X' + Y + Z$
5	1	0	1	F(1,0,1)	$X \cdot Y' \cdot Z$	$X' + Y + Z'$
6	1	1	0	F(1,1,0)	$X \cdot Y \cdot Z'$	$X' + Y' + Z$
7	1	1	1	F(1,1,1)	$X \cdot Y \cdot Z$	$X' + Y' + Z'$

Przykład dekompozycji funkcji przełączającej

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

Kanoniczna forma sumacyjna:

$$\begin{aligned} f(X) &= P_0(X_0) \cdot f(X_0) + P_1(X_1) \cdot f(X_1) + P_2(X_2) \cdot f(X_2) + P_3(X_3) \cdot f(X_3) = \dots \\ &= x_1' \cdot x_2' \cdot 0 + x_1' \cdot x_2 \cdot 1 + x_1 \cdot x_2' \cdot 0 + x_1 \cdot x_2 \cdot 1 = \dots \\ &= x_1' \cdot x_2 + x_1 \cdot x_2 \end{aligned}$$

Poprawione twierdzenie Shannona

Każdą funkcję przełączającą $f(X)$ można przedstawić jako kanoniczną formę sumacyjną 1-mintermów:

$$f(X) = \sum_{k=0}^{2^n-1} P_k^1(X_k) \equiv \sum (p_k)$$

Przykład dekompozycji funkcji przełączającej

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

Kanoniczna forma sumacyjna:

$$f(X) = x_1' \cdot x_2 + x_1 \cdot x_2 = \Sigma(1,3)$$

Kanoniczna forma iloczynowa

Każdą funkcję przełączającą $f(X)$ można przedstawić jako kanoniczną formę iloczynową:

$$f(X) = \prod_{k=0}^{2^n-1} (s_k(X_k) + f(X_k))$$

Przykład dekompozycji funkcji przełączającej

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

Kanoniczna forma iloczynowa:

$$f(X) = (S_0(X_0) + f(X_0)) \cdot (S_1(X_1) + f(X_1)) \cdot (S_2(X_2) + f(X_2)) \cdot (S_3(X_3) + f(X_3))$$

$$\dots = (x_1 + x_2 + 0) \cdot (x_1 + x_2' + 1) \cdot (x_1' + x_2 + 0) \cdot (x_1' + x_2' + 1) = \dots$$

$$\dots = (x_1 + x_2) \cdot (x_1' + x_2')$$

Kanoniczna forma iloczynowa

Każdą funkcję przełączającą $f(X)$ można przedstawić jako kanoniczną formę iloczynową 0-makstermów:

$$f(X) = \prod_{k=0}^{2^n-1} s_k^0(X_k) \equiv \prod (s_k)$$

Zasada równoważności form kanonicznych

Formy kanoniczne: iloczynowa 1-mintermów i sumacyjna 0-makstermów danej funkcji przełączającej są sobie równoważne:

$$f(X) = \prod (s_k) = \sum (p_k)$$

Zachodzi przy tym:


$$f'(X) = \prod (p_k) = \sum (s_k)$$

Przykład dekompozycji funkcji przełączającej

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

Równoważność form kanonicznych:

$$\begin{aligned} f(X) &= \sum(1,3) = \prod(0,2) \\ f'(X) &= \prod(1,3) = \sum(0,2) \end{aligned}$$

Dekompozycja funkcji przełączającej

Dekompozycja do postaci sumacyjnej:

- $f = (ABC) + \dots$
- z tablicy prawdy wyciągamy „1”
- wartościowanie pozytywowe

Dekompozycja do postaci iloczynowej:

- $f = (A+B+C) \cdot \dots$
- z tablicy prawdy wyciągamy „0”
- wartościowanie negatywowe

Stany nieokreślone

Może zdarzyć się sytuacja, że funkcja przełączająca nie będzie zdefiniowana dla pewnych stanów wejściowych – są to tzw. stany nieokreślone d (*don't care*).

Reguły postępowania:

(1) Redukcja przeciwdziedziny funkcji przełączającej – funkcja częściowa.

(2) Rozszerzenie przeciwdziedziny funkcji przełączającej o stany nieokreślone d – funkcja niezupełna.

Stany nieokreślone

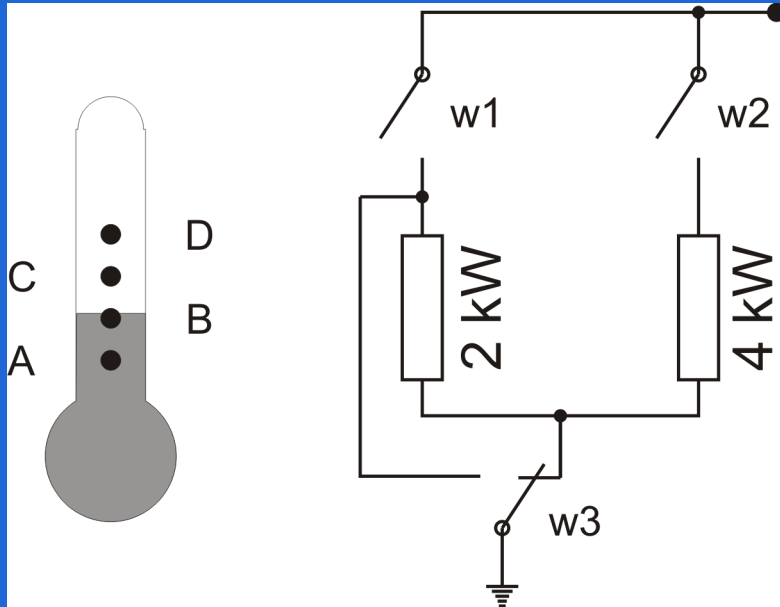
W zależności od wyboru postaci kanonicznej, stany nieokreślone d włączamy jako 1-mintermy lub 0-makstermy:

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

$$f(X) = \Sigma(1, 3, (0)) = \Pi(2, (0))$$

Układ sterowania ogrzewaniem

Zaprojektować układ kombinacyjny sterujący ogrzewaniem w/g podanego schematu:



a	b	c	d	w ₁	w ₂	w ₃
0	0	0	0	1	1	0
0	0	0	1	d	d	d
0	0	1	0	d	d	d
0	0	1	1	d	d	d
0	1	0	0	d	d	d
0	1	0	1	d	d	d
0	1	1	0	d	d	d
0	1	1	1	d	d	d
1	0	0	0	0	1	1
1	0	0	1	d	d	d
1	0	1	0	d	d	d
1	0	1	1	d	d	d
1	1	0	0	0	1	0
1	1	0	1	d	d	d
1	1	1	0	1	0	0
1	1	1	1	0	0	d

Układ sterowania ogrzewaniem

Postać funkcji logicznej dla poszczególnych przełączników:

Kanoniczna postać 1-mintermowa:

$$w_1(a,b,c,d) = a' \cdot b' \cdot c' \cdot d' + a \cdot b \cdot c' \cdot d' = \Sigma(0,14 \ (1,2,3,4,5,6,7,9,10,11,13))$$

$$w_2(a,b,c,d) = a' \cdot b' \cdot c' \cdot d' + a \cdot b' \cdot c' \cdot d' + a \cdot b \cdot c \cdot d' = \Sigma(0,8,12 \ (1,2,3,4,5,6,7,9,10,11,13))$$

$$w_3(a,b,c,d) = a' \cdot b' \cdot c' \cdot d' + a \cdot b' \cdot c' \cdot d' + a \cdot b \cdot c \cdot d' = \Sigma(8 \ (1,2,3,4,5,6,7,9,10,11,13,15))$$

Układ sterowania ogrzewaniem

Postać funkcji logicznej dla poszczególnych przełączników:

Kanoniczna postać 0-makstermowa:

$$w_1(a,b,c,d) = (a'+b+c+d) \cdot (a'+b'+c'+d) \cdot (a'+b'+c'+d') = \\ \Pi(8,12,15 \ (1,2,3,4,5,6,7,9,10,11,13))$$

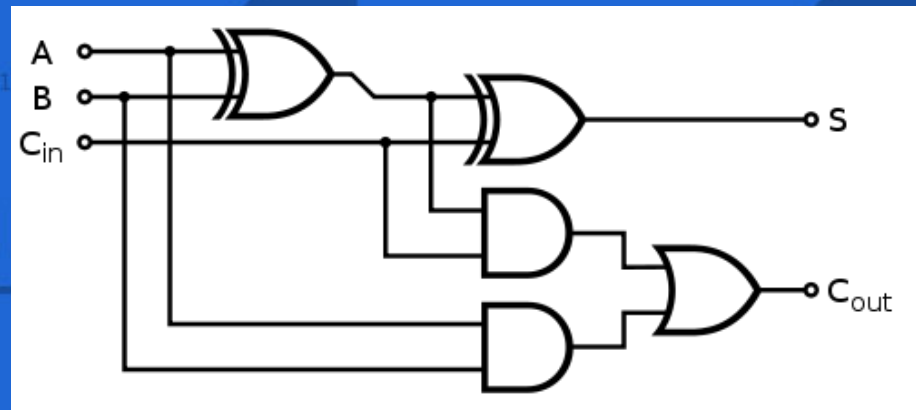
$$w_2(a,b,c,d) = (a'+b'+c+d) \cdot (a'+b'+c'+d') = \\ \Pi(14,15 \ (1,2,3,4,5,6,7,9,10,11,13))$$

$$w_3(a,b,c,d) = (a+b+c+d) \cdot (a'+b+c+d) \cdot (a'+b'+c+d) = \\ \Pi(0,12,14 \ (1,2,3,4,5,6,7,9,10,11,13,15))$$

Pełny sumator (Full Adder)

Full Adder Truth Table

<i>CARRY IN</i>	<i>input B</i>	<i>input A</i>	<i>CARRY OUT</i>	<i>SUM digit</i>
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1



Komparator 2-bitowy

A1	A0	B1	B0	Y1 (A > B)	Y2 (A = B)	Y3 (A < B)
0	0	0	0	0	1	0
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	0	1
0	1	0	0	1	0	0
0	1	0	1	0	1	0
0	1	1	0	0	0	1
0	1	1	1	0	0	1
1	0	0	0	1	0	0
1	0	0	1	1	0	0
1	0	1	0	0	1	0
1	0	1	1	0	0	1
1	1	0	0	1	0	0
1	1	0	1	1	0	0
1	1	1	0	1	0	0
1	1	1	1	0	1	0