Rozdział 2. Układy kombinacyjne - Zadania

Zadanie 2.1

Ile jest różnych działań 0, 1, 2 , n-argumentowych w algebrze Boole'a 2-elementowej

Rozwiązanie

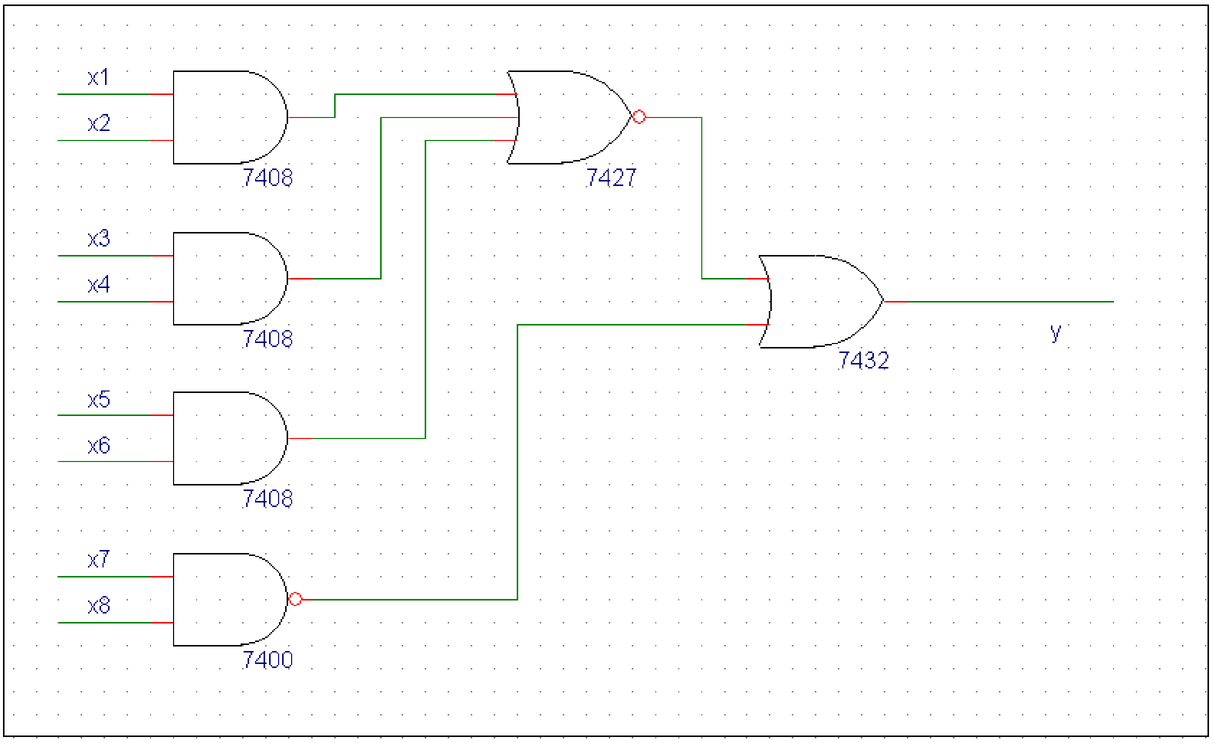
Działanie zeroargumetowe to wyróżniony element algebry. Mamy tylko 2 różne takie elementy w 2 elementowej algebrze Boole'a 0 i 1.

Działanie 1 argumentowe to odwzorowanie $f:\{0,1\} \to \{0,1\}$. Liczba wszystkich różnych takich odwzorowań jest liczbą wszystkich wariacji 2 elementowych z powtórzeniami ze zbioru 2 elementowego a więc 2^2. Jednym z tych odwzorowań jest oczywiście negacja.

Podobnie działanie n-argumentowe to odwzorowanie $f:\{0,1\}^n \to \{0,1\}$. Liczba wszystkich różnych takich odwzorowań jest liczbą wszystkich wariacji 2^n elementowych z powtórzeniami ze zbioru 2 elementowego a więc 2^{2^n}. W szczególności dla n=2, czyli dla działań 2-argumentowych, mamy $2^{2^4} = 16$ różnych działań.

Uwaga: Nie wszystkie działania n-argumentowe dają się równie łatwo zrealizować w postaci układów elektronicznych (nazywanych bramkami). Typowe bramki to NOT, OR, AND, NOR, NAND, EXOR z liczbą wejść ograniczoną na ogół do kilku. By można było z danego zestawu bramek tworzyć dowolną funkcję boolowską (metodą łączenia bramek), bramki powinny stanowić tzw. układ funkcjonalnie pełny. Na przykład układami funkcjonalnie pełnymi są { NOT, OR, AND}, {NOT,OR}, {NOT, AND} oraz {NAND} i {NOR}.

Zadanie 2.2



Rys. 1. Przykład układu kombinacyjnego realizującego pewną funkcję boolowską $f:\{0,1\}^8 \to \{0,1\}$

Znaleźć funkcję boolowską realizowaną przez układ z Rys.1.

Rozwiązanie:
Na wyjściu bramki 7427 mamy sygnał logiczny opisywany funkcją boolowską

$$f_1(x_1,x_2,x_3,x_4,x_5,x_6)=\overline{x_1x_2+x_3x_{4+}x_5x_6}$$

który podawany jest na wyjściową bramkę 7432 sumy logicznej.

Na wyjściu tej sumy czyli na wyjściu układu mamy więc

$$y=f_1(x_1,x_2,x_3,x_4,x_5,x_6)+\overline{x_7x_8}=\overline{x_1x_2+x_3x_4+x_5x_6}+\overline{x_7x_8}$$

Zadanie 2.3

Dana jest funkcja boolowska 3 zmiennych $f(x_1,x_2,x_3)=x_1x_2\bar{x}_3+x_1x_3$ podać tabelkę prawdy dla tej funkcji.

Rozwiązanie:

x_1 x_2 x_3	$f(x_1,x_2,x_3)$
0 0 0	0
0 0 1	0
0 1 0	0
0 1 1	0
1 0 0	0
1 0 1	1
1 1 0	1
1 1 1	1

Zadanie 2.4

Wykazać prawa de Morgana w 2 elementowej algebrze Boole'a

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

Rozwiązanie:
Sprawdzimy pierwszą tożsamość. Drugiej dowodzi się analogicznie.

x	y	$\overline{(x+y)}$	$\bar{x}\cdot\bar{y}$
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	0

Zadanie 2.5

Pokazać, że:

a) zbiór działań w dwuelementowej algebrze Boole'a złożony tylko z jednego 2-argumentowego działania NAND jest układem funkcjonalnie pełnym

b) zbiór działań w dwuelementowej algebrze Boole'a złożony tylko z jednego 2-argumentowego działania NOR jest układem funkcjonalnie pełnym.

Jak uzyskać w przypadku a) i b) funkcje stałe $f:\{0,1\}^n \to \{0,1\}$ tzn. takie, że $f(x_1,x_2,\ldots,x_n)=1$ dla każdego $x_1,x_2,\ldots,x_n \in \{0,1\}$ lub $f(x_1,x_2,\ldots,x_n)=0$ dla każdego $x_1,x_2,\ldots,x_n \in \{0,1\}$.

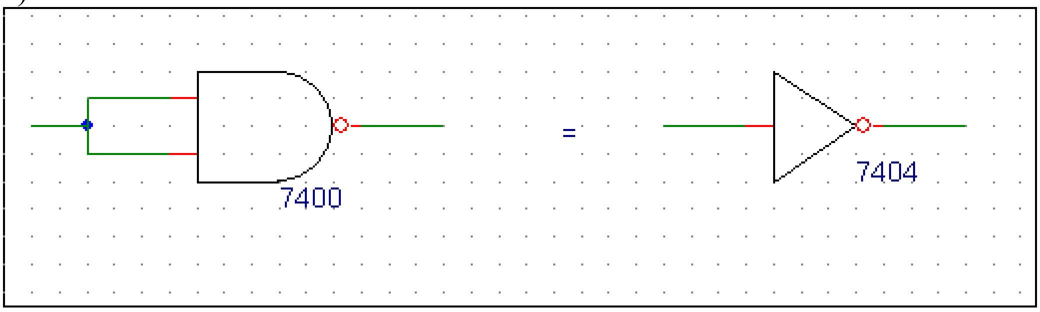
Rozwiązanie:

Wiadomo (por. twierdzenie o postaci kanonicznej sumacyjnej funkcji boolowskiej), że układem funkcjonalnie pełnym jest układ {NOT, OR, AND}, gdzie OR, AND są bramkami dwuwejściowymi. Wystarczy zatem pokazać, że można zrealizować bramki NOT, OR, AND za pomocą bramek NAND. Odpowiadające sobie układy pokazane są na Rys. 2.2 a), b) i c). Ostatnia równoważność z Rys. c) wynika z prawa de Morgana mamy bowiem

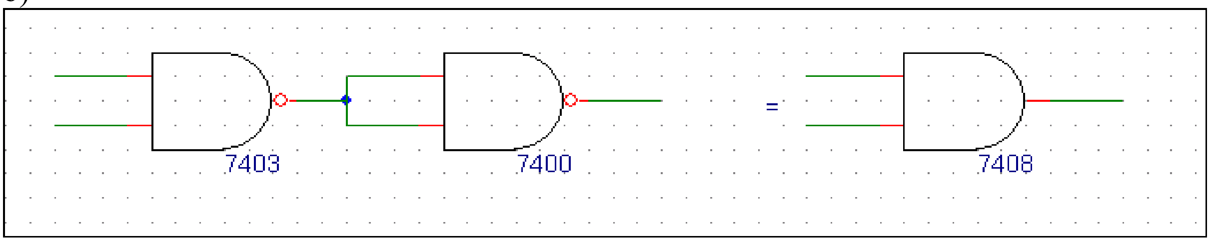
$$\overline{\overline{x}\cdot\overline{y}} = \overline{\overline{x}} + \overline{\overline{y}} = x + y$$

Analogicznie można wykazać, że układ {NOR} jest układem funkcjonalnie pełnym. Stałe 0 i 1 łatwo można uzyskać ze zrealizowanej za pomocą bramek NAND (odpowiednio NOR) sumy modulo 2.

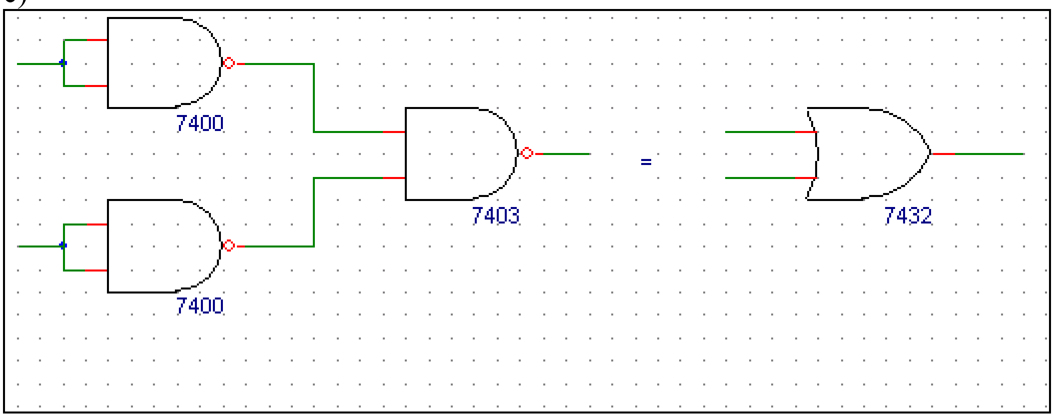
a)



b)



c)



Rys. 2. Realizacja układów NOT, OR i AND za pomocą bramek NAND

Zadanie 2.6

Jak zmienia się realizowana przez układ kombinacyjny funkcja przy zmianie konwencji logicznej.

Rozwiązanie:

Jeśli w konwencji dodatniej układ realizuje funkcję $f(x_1, x_2, ..., x_n)$, to w konwencji ujemnej ten sam układ realizuje funkcję $\overline{f(\bar{x}_1, \bar{x}_2, \bar{x}_3, ..., \bar{x}_n)}$.

Podobnie, jeśli w konwencji ujemnej układ realizuje funkcję $f(x_1, x_2, ..., x_n)$, to w konwencji dodatniej realizuje funkcję $\overline{f(\bar{x}_1, \bar{x}_2, \bar{x}_3, ..., \bar{x}_n)}$.

Na przykład układ NAND zrealizowany w konwencji dodatniej po zmianie konwencji staje się układem NOR a układ NOR staje się układem NAND. Istotnie korzystając z praw de Morgana dostajemy:

$$\overline{\overline{\bar{x}_1 \cdot \bar{x}_2}} = \bar{x}_1 \cdot \bar{x}_2 = \overline{x_1 + x_2}$$

oraz

$$\overline{\overline{\bar{x}_1 + \bar{x}_2}} = \bar{x}_1 + \bar{x}_2 = \overline{x_1 \cdot x_2}$$

Podobnie można wykazać, że po zmianie konwencji układ AND przechodzi na OR, a układ OR przechodzi na AND.

Zadanie 2.7

Funkcja boolowska zadana jest poniższą tabelą prawdy. Zaprojektować układ realizujący tę funkcję.

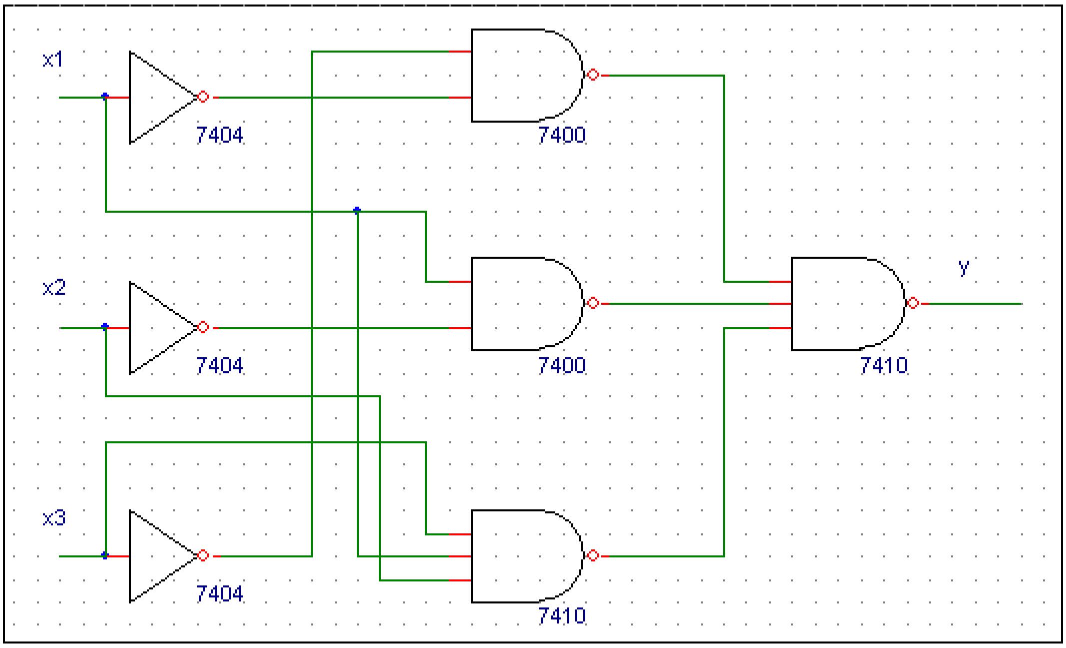
$x_1\ x_2\ x_3$	$f(x_1, x_2, x_3)$
0 0 0	1
0 0 1	0
0 1 0	1
0 1 1	0
1 0 0	1
1 0 1	1
1 1 0	0
1 1 1	1

Rozwiązanie

Na mocy twierdzenia o postaci kanonicznej dysjunkcyjnej mamy,

$$f(x_1, x_2, x_3) = \bar{x}_1 \cdot \bar{x}_2 \cdot \bar{x}_3 + \bar{x}_1 \cdot x_2 \cdot \bar{x}_3 + x_1 \cdot \bar{x}_2 \cdot \bar{x}_3 + x_1 \cdot \bar{x}_2 \cdot x_3 + x_1 \cdot x_2 \cdot x_3 =$$
$$= \bar{x}_1 \cdot \bar{x}_3 + x_1 \cdot \bar{x}_2 + x_1 \cdot x_2 \cdot x_3$$

Układ realizujący powyższą funkcję pokazany jest na Rys. 3.

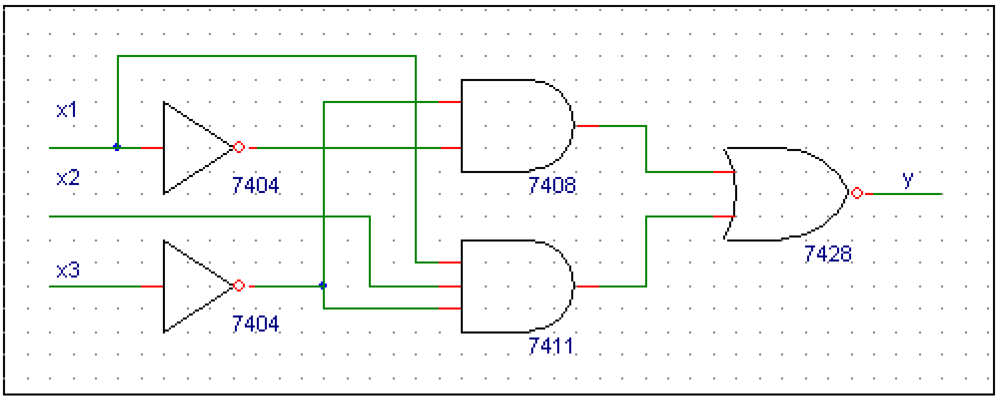


Rys. 3. Realizacja funkcji boolowskiej opisanej zadana tabelą prawdy

Warto zauważyć, że oszczędniej jeśli chodzi o liczbę bramek byłoby najpierw „wygenerować" zera, a potem użyć negacji co prowadzi do przedstawienia funkcji f w postaci

$$f(x_1,x_2,x_3)=\overline{\overline{x_1}\cdot\overline{x_2}\cdot x_3+\overline{x_1}\cdot x_2\cdot x_3+x_1\cdot x_2\cdot\overline{x_3}}=\overline{\overline{x_1}\cdot x_3+x_1\cdot x_2\cdot\overline{x_3}}$$

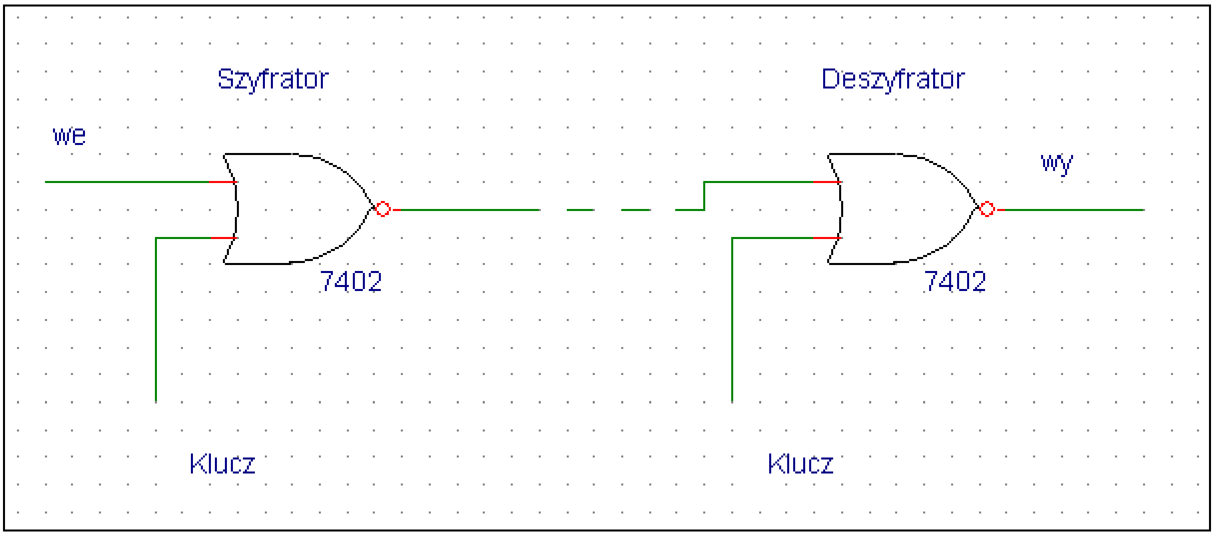
Odpowiedni schemat pokazany jest na Rys. 4.



Rys. 4 Realizacja funkcji boolowskiej opisanej zadaną tabelą prawdy

Zadanie 2.8

Wiadomo, że Alicja i Bob posługują się szyfrem Vernama (koder i dekoder są sumą modulo 2 (por. Rys.5.). Przechwycono wiadomość jawną $m=m_1m_2...m_r$, gdzie $m_i\in\{0,1\}$ i szyfrogram $c=c_1c_2...c_r$, gdzie $c_i\in\{0,1\}$, znaleźć klucz $k=k_1k_2...k_r$; $k_i\in\{0,1\}$ jakim posłużyli się Alicja i Bob.



Rys.5. System kryptograficzny Vernama wykorzystuje 2 sumy modulo 2

Rozwiązanie:
Szyfrogram $c = c_1 c_2 ... c_r$ tworzony jest tak, że dla każdego $i = 1,2,...r$ przyjmujemy

$$c_i = m_i \oplus k_i \tag{1}$$

Mnożąc obie strony równości (1) przez m_i i korzystając z łączności działania sumy modulo 2 w Z_2, dostajemy

$$\begin{aligned} m_i \oplus c_i &= m_i \oplus (m_i \oplus k_i) \\ m_i \oplus c_i &= (m_i \oplus m_i) \oplus k_i \\ m_i \oplus c_i &= 0 \oplus k_i \\ m_i \oplus c_i &= k_i \end{aligned}$$

Uwaga: Warto zauważyć, że suma modulo 2 w Z_2 definiowana dla $a, b \in Z_2$ jako reszta z dzielenia $a+b$ przez 2, czyli $a \oplus b \overset{df}{=} (a+b) \pmod 2$, jest tym samym działaniem co suma modulo 2 zdefiniowana równościami $0 \oplus 0 = 0$, $0 \oplus 1 = 1$, $1 \oplus 0 = 1$, $1 \oplus 1 = 0$.

Zadanie 2.9

Zaprojektować układ kombinacyjny sprawdzający parzystość słowa binarnego 8-to bitowego $x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8$.

Rozwiązanie:
Przyjrzyjmy się funkcji

$$f(x_1, x_2, ..., x_n) = x_1 \oplus x_2 \oplus ... \oplus x_n$$

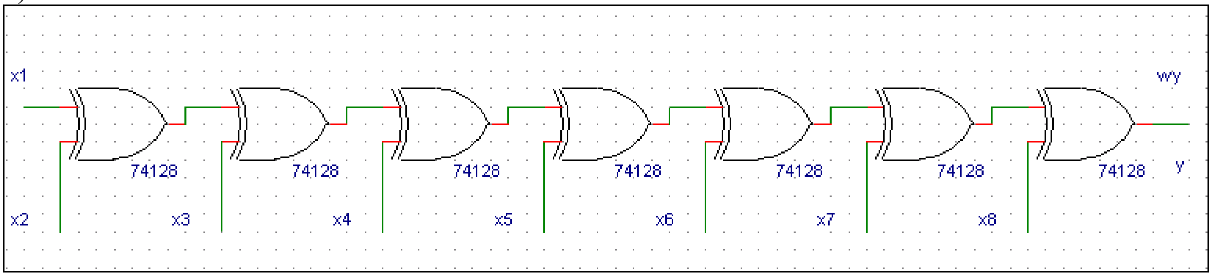
Jeśli liczba jedynek w słowie wejściowym $x_1 x_2 x_3 x_4 x_5 x_6 ... x_n$ jest parzysta to wartość funkcji $f(x_1, x_2, ..., x_n) = x_1 \oplus x_2 \oplus ... \oplus x_n$ jest równa 0 a jeśli nieparzysta to 1. Wynika to bezpośrednio z definicji sumy modulo 2 oraz przemienności i łączności sumy modulo2.

Zatem w przypadku słowa 8 bitowego musimy zrealizować funkcję

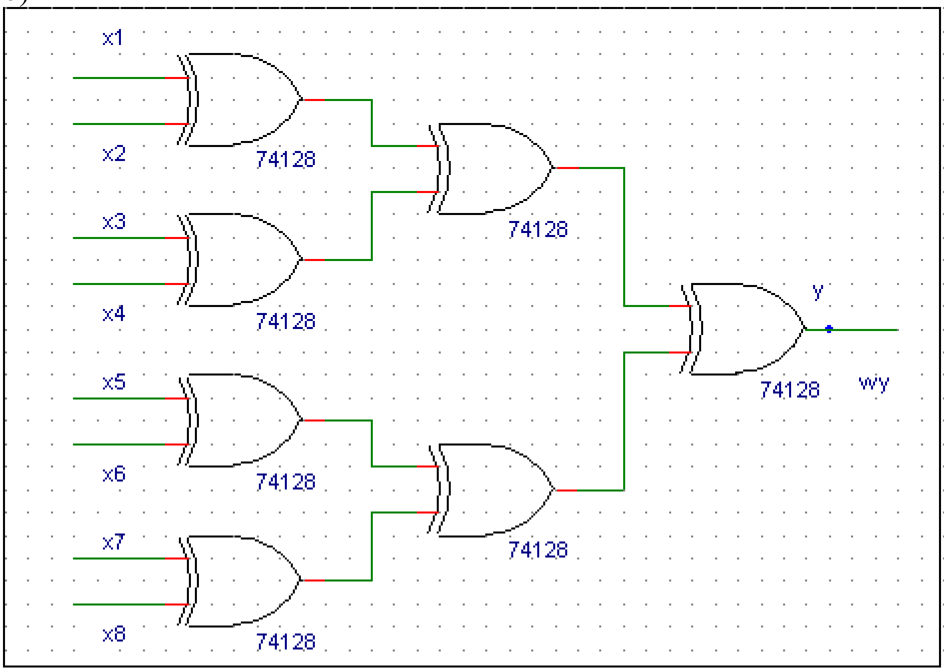
$$f(x_1,x_2,x_3,x_4,x_5,x_6,x_7,x_8) = x_1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus x_5 \oplus x_6 \oplus x_7 \oplus x_8$$

Na Rys. 6. pokazane są 2 układy realizujące tę funkcję za pomocą 2 wejściowych bramek sumy modulo 2 (układy EXOR). Czytelnik zechce zauważyć, że układ a) zawiera tyle samo bramek ale wnosi większe opóźnienie niż układ pokazany na Rys. b).

a)



b)



Rys 6. Układy sprawdzające parzystość słowa binarnego $x_1x_2x_3x_4x_5x_6x_7x_8$

Zadanie 2.10

Funkcja boolowska 3 zmiennych zadana jest poniższą tabelą prawdy. Zaprojektować układ realizujący tę funkcję używając tylko 2- lub 3-wejściowych bramek NAND.

x_1 x_2 x_3	$f(x_1,x_2,x_3)$
0 0 0	1
0 0 1	0
0 1 0	0
0 1 1	0
1 0 0	0
1 0 1	0
1 1 0	0
1 1 1	1

Zadanie 2.11

Zaprojektować układ kombinacyjny wykrywający liczbę -100 (D) w 8-mio bitowym zapisie U2.

Zadanie 2.12

Pokazać, że $(Z_2, \oplus, \cdot)$ gdzie $\oplus$ jest sumą modulo 2 a działanie mnożenia jest zwykłym iloczynem boolowskim jest ciałem. Jak wyglądają wielomiany o współczynnikach w Z_2.

Zadanie 2.13

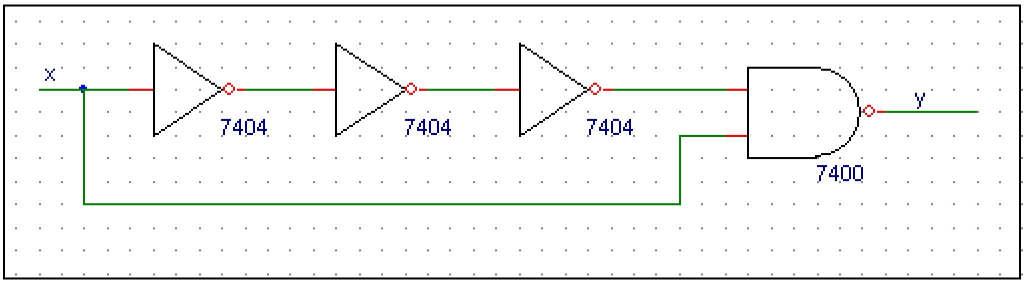
Pomnożyć 2 wielomiany boolowskie 3 stopnia przez siebie. Zaproponować układ kombinacyjny do mnożenia wielomianów boolowskich 3 stopnia przez siebie.

Zadanie 2.14

Rozważmy układ kombinacyjny pokazany na Rys. 2.1 . Załóżmy, że słowo wejściowe zmienia się w chwili 0. Po jakim czasie na wyjściu układu otrzymamy na pewno prawidłową wartość napięcia przy założeniu, że opóźnienie każdej bramki jest jednakowe i równe $\Delta t = 1ns$.

Zadanie 2.15

Na Rys. 7. pokazany jest tzw. układ różniczkujący na bramkach. Naszkicować impuls wyjściowy układu w odpowiedzi na skok napięcia wejściowego z L na H. Jak można regulować szerokość impulsu wyjściowego.

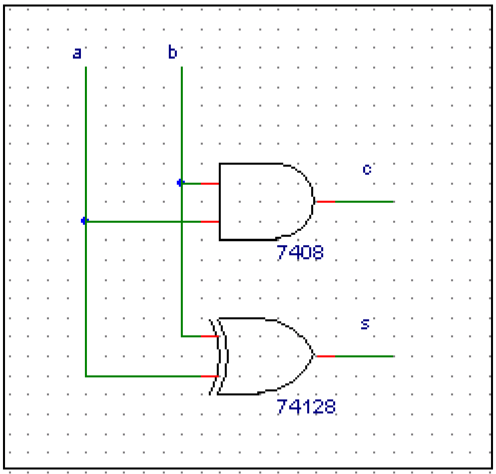


Rys. 7. Układ różniczkujący na bramkach

Zadanie 2.16

Zrealizować układ kombinacyjny dodający w zapisie NKB 2 bity a i b bez uwzględnienia przeniesienia z poprzedniej pozycji. Taki układ nosi nazwę półsumatora.

Rozwiązanie:
Układ dodający 2 bity powinien mieć 2 wyjścia: wyjście sumy s i wyjscie przeniesienia c. Łatwo zauważyć, że układ półsumatora opisują 2 funkcje boolowskie $s = a \oplus b$ oraz $c = a \cdot b$, gdzie s jest sumą a c przeniesieniem. Układ realizujący półsumator pokazany jest na Rys 8.



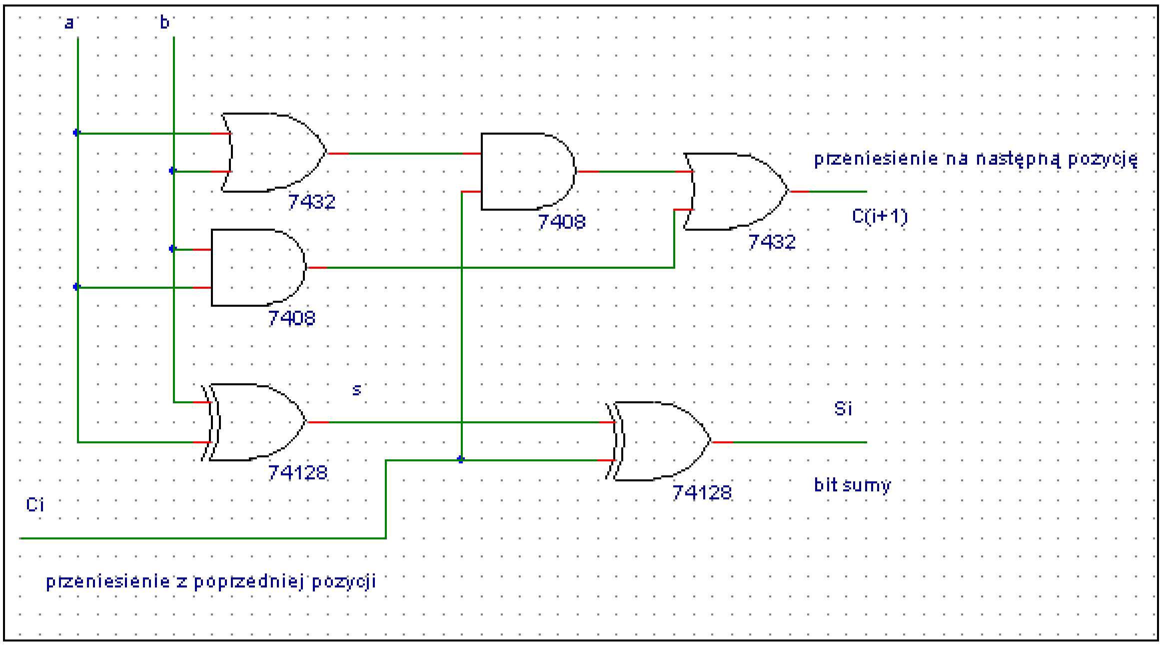
Rys 8. Układ realizujący półsumator.

Zadanie 2.17

Zrealizować układ kombinacyjny dodający (w zapisie NKB) 2 bity a_i i b_i na i tej pozycji z uwzględnieniem przeniesienia z poprzedniej pozycji. c_i. Taki układ nosi nazwę sumatora jednobitowego lub jednobitowego sumatora pełnego. Ocenić z jakim opóźnieniem otrzymujemy poprawne bity sumy i przeniesienia w sumatorze jednobitowym pełnym jeśli opóźnienie pojedynczej bramki wynosi $\Delta t = 1ns$.

Rozwiązanie

Układ dodający 2 bity ma 3 wejścia (a_i, b_i i c_i) oraz 2 wyjścia: wyjście sumy s_i i wyjście przeniesienia na następną pozycję c_{i+1}. Łatwo zauważyć, że układ sumatora opisują 2 funkcje boolowskie $s_i = a_i \oplus b_i \oplus c_i$ oraz $c_{i+1} = a_i \cdot b_i + (a_i + b_i) \cdot c_i$, gdzie s_i jest sumą, c_{i+1} przeniesieniem na pozycję $i+1$ a c_i przeniesieniem. na pozycję *i*. Układ realizujący sumator jednobitowy pełny pokazany jest na Rys 2.6. Czas po którym uzyskujemy w tym układzie poprawny bit sumy to $2\Delta t = 2ns$. Poprawny bit przeniesienia uzyskujemy po czasie $3\Delta t = 3ns$.



Rys 9. Układ realizujący jednobitowy sumator pełny

Zadanie 2.18

Narysować fragment układu przewidywania przeniesień (układ look ahead) obliczający przeniesienie z pozycji 4 na 5. Przyjmujemy numerację bitów w słowie taką, że LSB ma numer 0.

Zadanie 2.19

Zaprojektować układ kombinacyjny ustawiający znacznik OF rejestru znaczników. Rozważyć 2 przypadki

- gdy mamy dostęp do bitów przeniesień sumatora (a więc do wnętrza sumatora)
- gdy mamy dostęp tylko do argumentów i wyjścia sumatora.