DODATEK II - Inne sposoby realizacji funkcji logicznych

W kolejnych podpunktach zaprezentowano sposoby realizacji przykładowej funkcji (tej samej co w instrukcji do ćwiczenia "Synteza układów kombinacyjnych") w innych wersjach.

Mamy zatem tabelę taką samą jak poprzednio:

l.p.	A	B	C	OUT
0	0	0	0	1
1	0	0	1	1
2	0	1	0	0
3	0	1	1	x
4	1	0	0	x
5	1	0	1	0
6	1	1	0	1
7	1	1	1	0

ⓘ *Znak X pojawiający się w tabeli oznacza iż określona kombinacja wejść nie ma prawa w układzie wystąpić lub że z pewnych powodów nie interesuje nas wyjście układu dla danej kombinacji wejść.*

1.1. Układ niezminimalizowany na bramkach AND i OR – trzy przykładowe sposoby realizacji

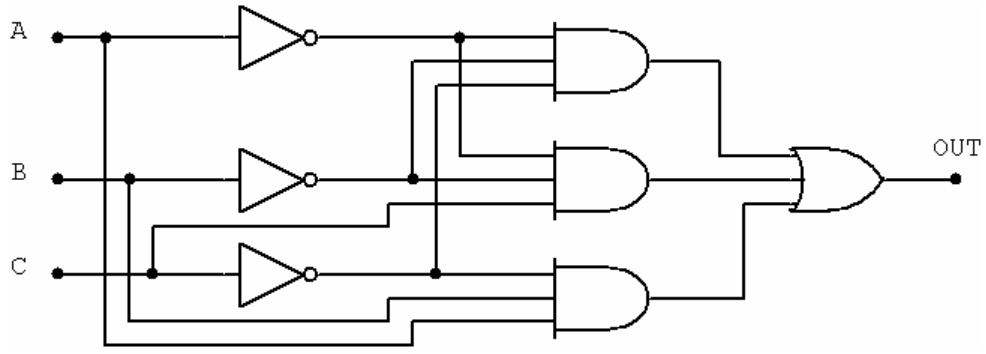
Na podstawie tabeli, tak jak we wstępie do ćwiczenia, uzyskujemy opis kanoniczny projektowanego układu w postaci sumacyjnej:

$$OUT = \sum(0,1,6(3,4))$$

i odpowiadający mu opis kanoniczny w zapisie Boola:

$$OUT = \overline{A} \cdot \overline{B} \cdot \overline{C} + \overline{A} \cdot \overline{B} \cdot C + A \cdot B \cdot \overline{C} \,.$$

Na podstawie tego zapisu możemy wprost zrealizować funkcję logiczną, jak na rysunku:



Przy założeniu, że realizujemy układ w technologii CMOS, można do tego celu użyć jednego układu 4075 (trzy 3-wejściowe bramki OR, z których wykorzystujemy jedną), jednego 4073 (trzy 3-wejściowe bramki AND) i jednego 4069 (sześć inwerterów (bramek NOT), z których wykorzystujemy trzy).

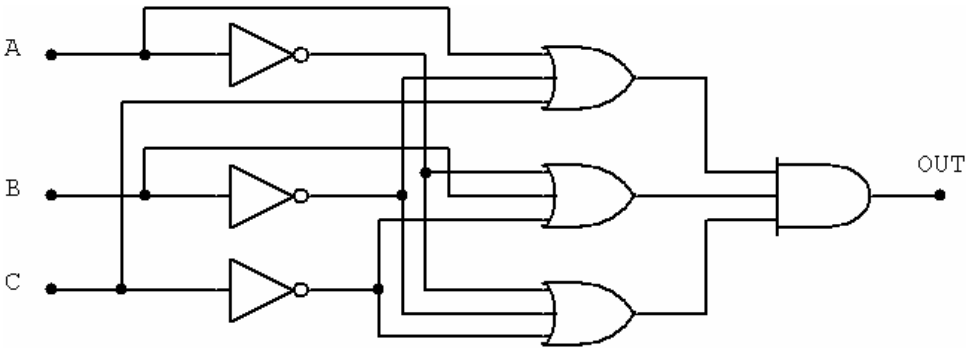
Na podstawie tabeli można też pozyskać opis kanoniczny projektowanego układu w postaci iloczynowej:

$$\boldsymbol{OUT = \prod(2,5,7(3,4))}.$$

Pozyskuje się ją odwrotnie do sumacyjnej, tamta wynika z jedynek układu, ta z zer, stany dowolne się oczywiście powtarzają. Opis kanoniczny w zapisie Boola w tym przypadku wygląda tak (zamiast sumy iloczynów mamy iloczyn sum):

$$OUT = \left(A + \overline{B} + C\right) \cdot \left(\overline{A} + B + \overline{C}\right) \cdot \left(\overline{A} + \overline{B} + \overline{C}\right).$$

Układ taki można zrealizować z wykorzystaniem tych samych układów co poprzednio, tyle że tym razem wykorzystane zostaną 3 bramki OR i jedna AND:



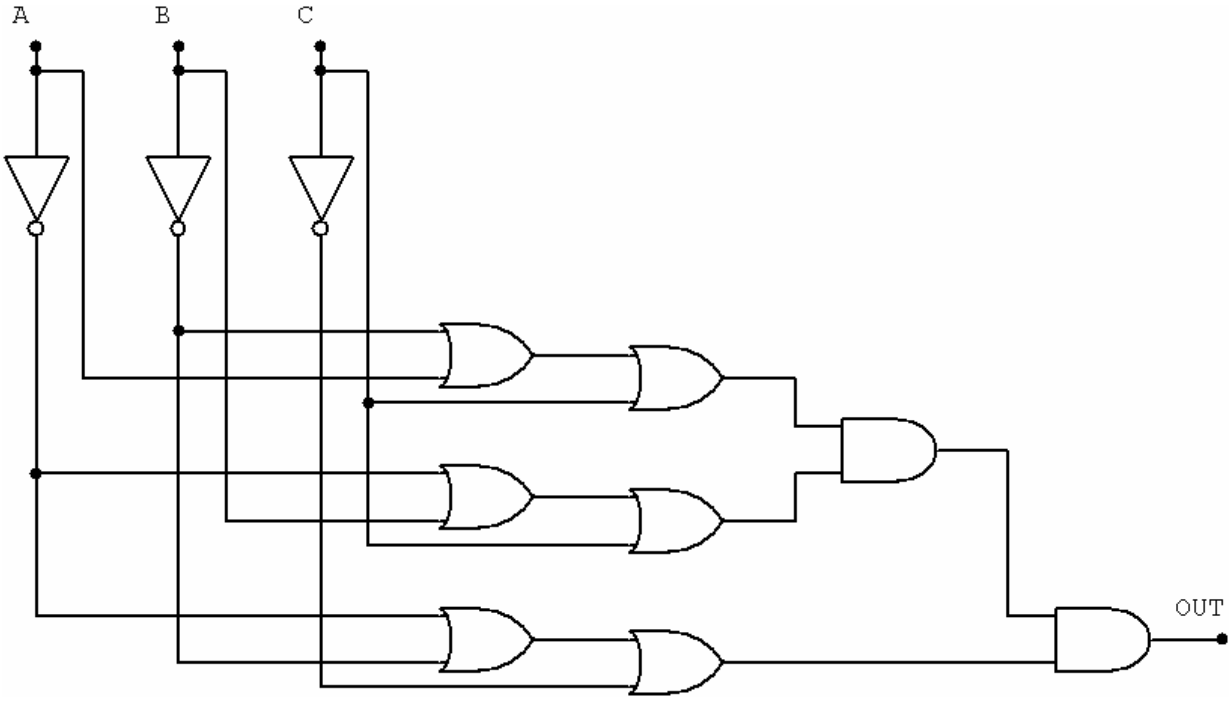
Czasem dysponujemy tylko określonym rodzajem bramek logicznych, należy wtedy (o ile to możliwe) doprowadzić opis układu do takiej postaci by była możliwa realizacja założeń projektowych. Załóżmy np. że funkcję:

$$OUT = \left(A + \overline{B} + C\right) \cdot \left(\overline{A} + B + \overline{C}\right) \cdot \left(\overline{A} + \overline{B} + \overline{C}\right).$$

musimy zrealizować korzystając tylko i wyłącznie z układów 7404, 7408 i 7432, czyli z bramek NOT i 2-wejściowych bramek AND i OR. W realizacji powyżej korzystaliśmy z bramek 3-wejściowych, takiej jaka wynika bezpośrednio z postaci funkcji. Jeżeli chcemy korzystać z bramek 2-wejściowych to funkcję tą musimy doprowadzić np. do postaci:

$$OUT = \left(\left(\left(A + \overline{B}\right) + C\right) \cdot \left(\left(\overline{A} + B\right) + \overline{C}\right)\right) \cdot \left(\left(\overline{A} + \overline{B}\right) + \overline{C}\right)$$

i wtedy da się ją zrealizować tak:

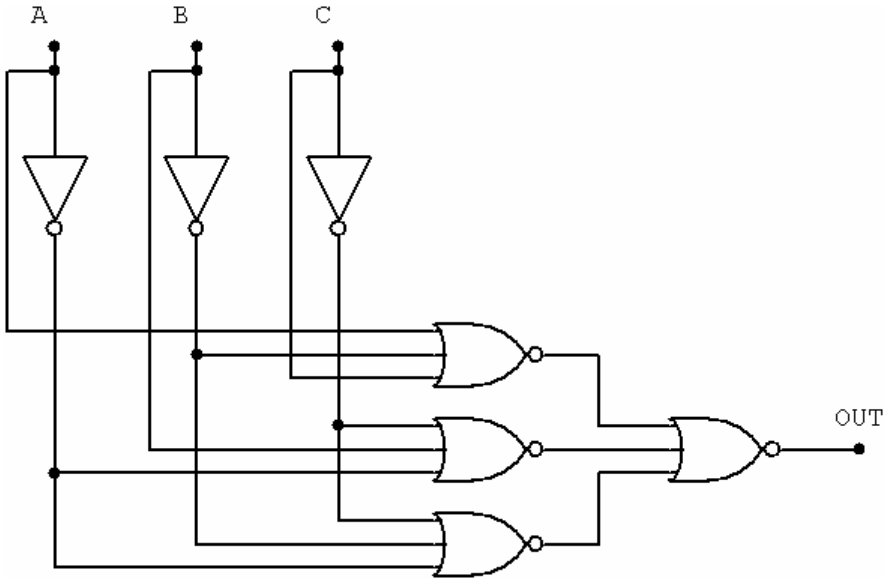


1.2. Układ niezminimalizowany na bramkach NOR

Warto wspomnieć, że analogicznie jak w przypadku realizacji na bramkach NAND pokazanej we wstępie do ćwiczenia, postać iloczynową funkcji można przekształcić z wykorzystaniem praw de Morgana do postaci:

$$OUT = \left(A + \overline{B} + C\right) \cdot \left(\overline{A} + B + \overline{C}\right) \cdot \left(\overline{A} + \overline{B} + \overline{C}\right) = \overline{\overline{\left(A + \overline{B} + C\right) \cdot \left(\overline{A} + B + \overline{C}\right) \cdot \left(\overline{A} + \overline{B} + \overline{C}\right)}} =$$
$$= \overline{\left(A + \overline{B} + C\right) + \left(\overline{A} + B + \overline{C}\right) + \left(\overline{A} + \overline{B} + \overline{C}\right)},$$

dzięki czemu uzyskujemy postać do realizacji z wykorzystaniem 3-wejściowych bramek NOR (np. układów 4025 w technologii CMOS lub 7427 w technologii TTL). Układ wygląda wtedy tak:

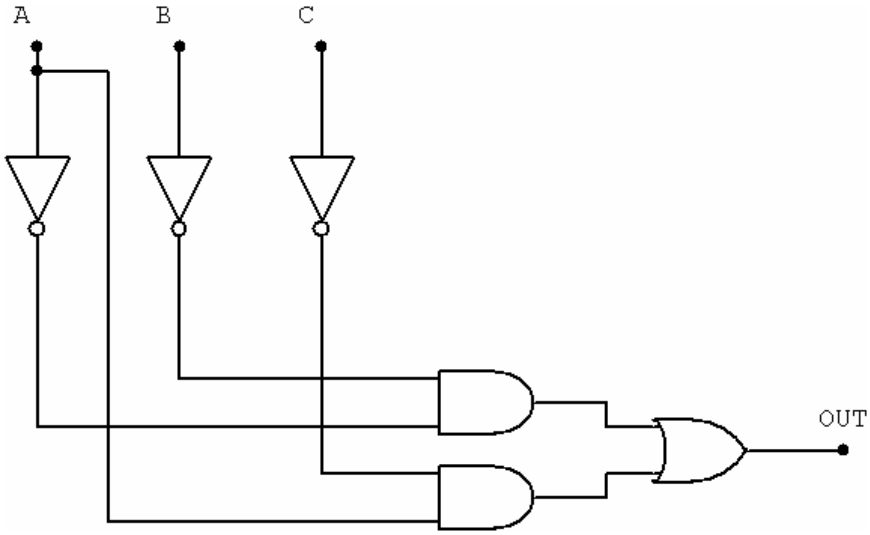


1.3. Układ zminimalizowany na bramkach AND I OR – dwie wersje

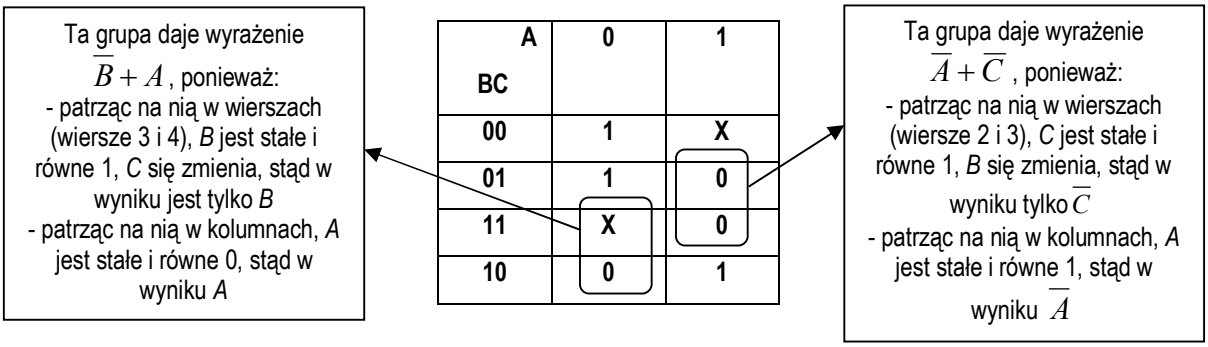
Funkcja zminimalizowana w punkcie 2.2 ma postać jak niżej:

$$OUT = \overline{A} \cdot \overline{B} + A \cdot \overline{C}$$

zatem podobnie jak poprzednio można ją zrealizować na bramkach AND i OR, układ będzie wyglądał tak:



Niemniej podobnie jak postać nieminimalną funkcji można pozyskać tak z jedynek jak i z zer, tak samo minimalizować układ można na dwa sposoby, względem jedynek lub zer. Jeżeli zadaną funkcję chcemy zminimalizować korzystając z zer, to tablica Karnaugh będzie wyglądać tak samo, zmienią się grupy, tym razem łączymy zera:



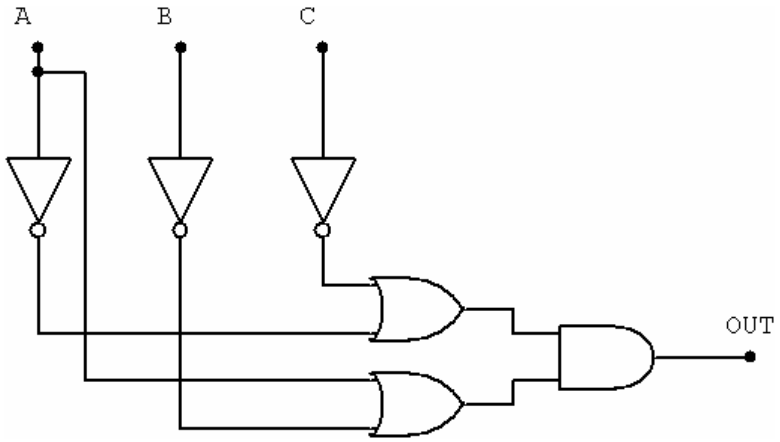
Zasady tworzenia grup przy minimalizacji względem zer:

- *łączymy w grupy wszystkie zera (jedno zero to też grupa!),*
- *X-y mogą być uznane za 1 lub 0, jak nam jest wygodniej, w przykładzie powyżej jeden X został uznany za 0 i wykorzystany w minimalizacji, drugi za 1 i niewykorzystany.*
- *grupa powinna być "zwarta", ale może się symetrycznie „zawijać" (podobnie jak w minimalizacji według 1 w poprzednim przykładzie),*
- *grupę mogą stworzyć następujące ilości czynników: 1, 2, 4, 8, 16… (czyli potęgi dwójki), przy czym pożądane jest by grupy były jak największe.*

Postać minimalna iloczynowa wygląda więc tak:

$$OUT = \left(A + \overline{B}\right) \cdot \left(\overline{A} + \overline{C}\right)$$

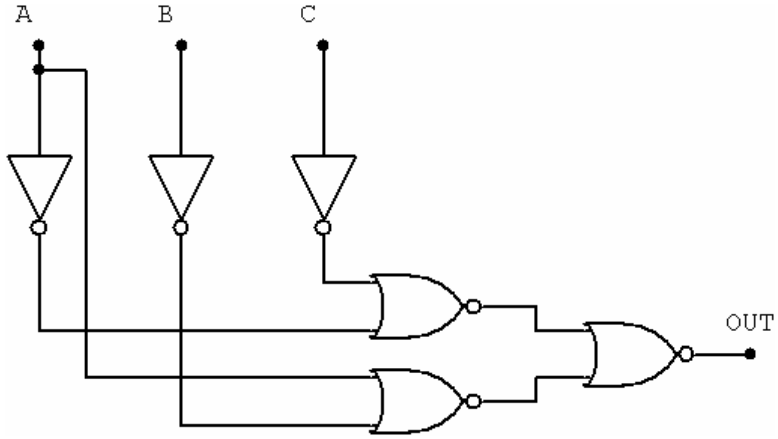
natomiast jej realizacja tak:



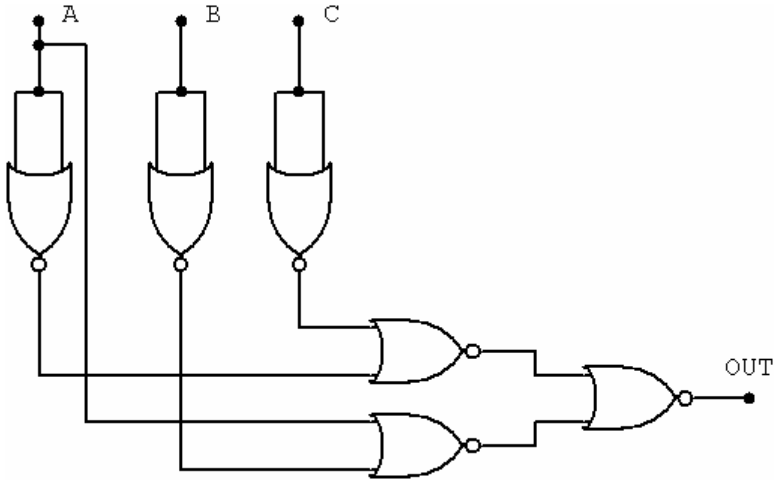
Opis minimalny można oczywiście przekształcić z pomocą praw de Morgana do postaci:

$$OUT = \left(A + \overline{B}\right) \cdot \left(\overline{A} + \overline{C}\right) = \overline{\left(A + \overline{B}\right) \cdot \left(\overline{A} + \overline{C}\right)} = \overline{\left(A + \overline{B}\right)} + \overline{\left(\overline{A} + \overline{C}\right)},$$

która pozwala na realizację funkcji z wykorzystaniem wyłącznie bramek NOR:



a w przypadku gdy nawet bramki NOT chcemy zastąpić bramkami NOR:



1.4. Podsumowanie

W dodatku pokazano 8 sposobów bramkowych realizacji przykładowej funkcji logicznej, kolejne 2 sposoby pokazano we wstępie do ćwiczenia "Synteza układów kombinacyjnych". Oczywiście nie są to wszystkie możliwości, ale w ogólności pokazują one jakie możliwości ma projektujący tego typu układy.

Jeżeli chodzi o realizację pokazanych w instrukcji układów w strukturach układów programowalnych, to dostępność poszczególnych elementów logicznych zależy od wykorzystanego układu FPGA/CPLD, niemniej znakomita większość układów zawiera bardzo wiele możliwych do wykorzystania elementów logicznych, niezmiernie rzadko zdarza się, by w projekcie układu logicznego pojawił się element, który nie ma swojego odpowiednika w strukturze wykorzystywanego układu.

Warto też wspomnieć, że układy programowalne można programować nie tylko poprzez projektowanie schematu układu (jak w ćwiczeniu), ale przy użyciu dwóch języków opisu sprzętu - VHDL lub Verilog, z których popularniejszy jest ten pierwszy. W przypadku takiego podejścia zadaniem projektanta jest prawidłowy opis projektowanego układu, przełożeniem opisu na strukturę logiczną zajmuje się kompilator.