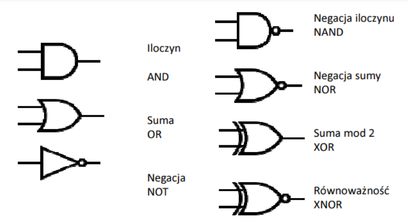
Synteza układów kombinacyjnych

Paweł Nowik

9 grudnia 2019

Układy kombinacyjne są pozbawione właściwości pamiętania stanów, realizują funkcje logiczne w oparciu o bramki i inne proste układy cyfrowe. Bramkami nazywamy kombinacyjne układy cyfrowe realizujące proste funkcje logiczne jednej lub wielu zmiennych. Zmienną logiczną jest sygnał elektryczny występujący na wejściach i wyjściach tych układów. Działanie bramek jest opisane za pomocą tablicy prawdy zawierającej stany logiczne 0/1 lub poziom wielkości fizycznej (np. napięcia) niski(L) lub wysoki (H).

Bramki logiczne



W celu dokonania syntezy układy kombinacyjnego należy:
- określić funkcję logiczną rozpatrywanego problemu za pomocą tablicy prawdy,
- przeprowadzić minimalizację formuły opisującej funkcję logiczną, np. wykorzystując prawa algebry logiki,
- sporządzić schemat na bramkach, realizujący zminimalizowaną funkcję logiczną.

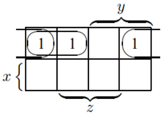
Przykład - za pomocą bramek NAND zrealizować układ opisany tablicą prawdy pokazaną na rysunku.

Wejścia			Wyjścia
x	y	z	f
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

Kroki minimalizacji z użyciem tablicy Karnaugha:
1. Zapisujemy wartości funkcji w odpowiedniej tabeli Karanaugh'a.
2. Szukamy grup "1" co wielkość 2^n czyli, (1,2,4,8,...) sąsiadujących ze sobą.
3. Sklejamy jak największe grupy "1" logicznych i zapisujemy odpowiadjące im wyrażenie alternatywne.
4. Szukamy minimalnej liczby obszarów, zawierających jak największe grupy "1" logicznych (wszystkie jedynki muszą wchodzić w skład chociaż jednego obszaru).
5. Zapisujemy wyrażenie alternatywne.

Na podstawie tablicy prawd otrzymujemy funkcję logiczną
$f(x,y,z) = \Sigma(0,1,2)$

która po minimalizacji



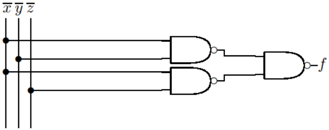
przyjmuje postać

$$f(x,y,z) = \sum(0,1,2) = \overline{x}\,\overline{y} + \overline{x}\,\overline{z}$$

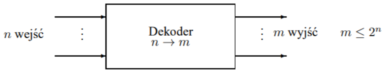
Korzystając z praw de Morgana, otrzymujemy funkcję

$$f(x,y,z) = \overline{x}\,\overline{y} + \overline{x}\,\overline{z} = \overline{\overline{\overline{x}\,\overline{y} + \overline{x}\,\overline{z}}} = \overline{\overline{\overline{x}\,\overline{y}} \cdot \overline{\overline{x}\,\overline{z}}}$$

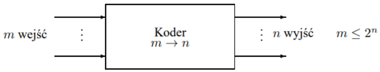
Na podstawie tej funkcji rysujemy schemat logiczny układu.



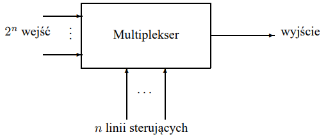
Dekoderem nazywamy układ kombinacyjny mający n wejść i m wyjść $m \leqslant 2^n$, w którym każdej kombinacji zmiennych wejściowych odpowiada pojawienie się jedynki logicznej tylko na jednym wyjściu, na pozostałych wyjściach pojawiają się wtedy zera logiczne.



Koderem nazywamy układ kombinacyjny działający odwrotnie od dekodera. Koder ma m wejść i n wyjść, $m \leqslant 2^n$.



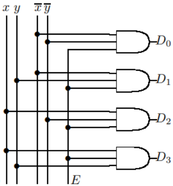
Multiplekserem nazywamy układ kombinacyjny wybierający informację dwójkową na jednej z linii wejściowych i kierujących ją na jedną linię wyjściową. Wybór linii wejściowej jest określany przez linie sterujące. Schemat blokowy multipleksera o 2^n liniach wejściowych i n liniach sterujących.



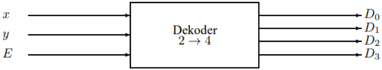
a)

Wejścia			Wyjścia			
E	x	y	D_0	D_1	D_2	D_3
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	0	1
0	x	x	0	0	0	0

b)



c)

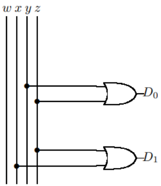


Rys. 3.24. Dekoder $2 \to 4$ z wejściem E (*enable*): tablica prawdy (a)), schemat logiczny (b)) i schemat blokowy (c))

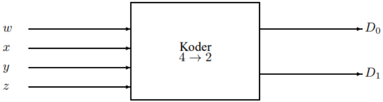
a)

Wejścia				Wyjścia	
w	x	y	z	D_0	D_1
1	0	0	0	0	0
0	1	0	0	0	1
0	0	1	0	1	0
0	0	0	1	1	1

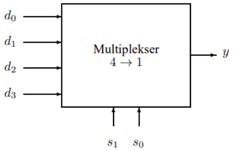
b)



c)



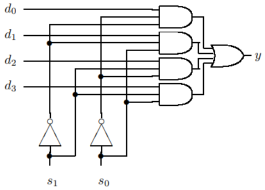
Rys. 3.27. Koder $4 \rightarrow 2$: tablica prawdy (a)), schemat logiczny (b)) i schemat blokowy (c))



a)

b)

s_1	s_0	y
0	0	d_0
0	1	d_1
1	0	d_2
1	1	d_3



c)

Rys. 3.29. Multiplekser $4 \to 1$: schemat blokowy (a)), tablica prawdy (b)) i schemat logiczny (c))