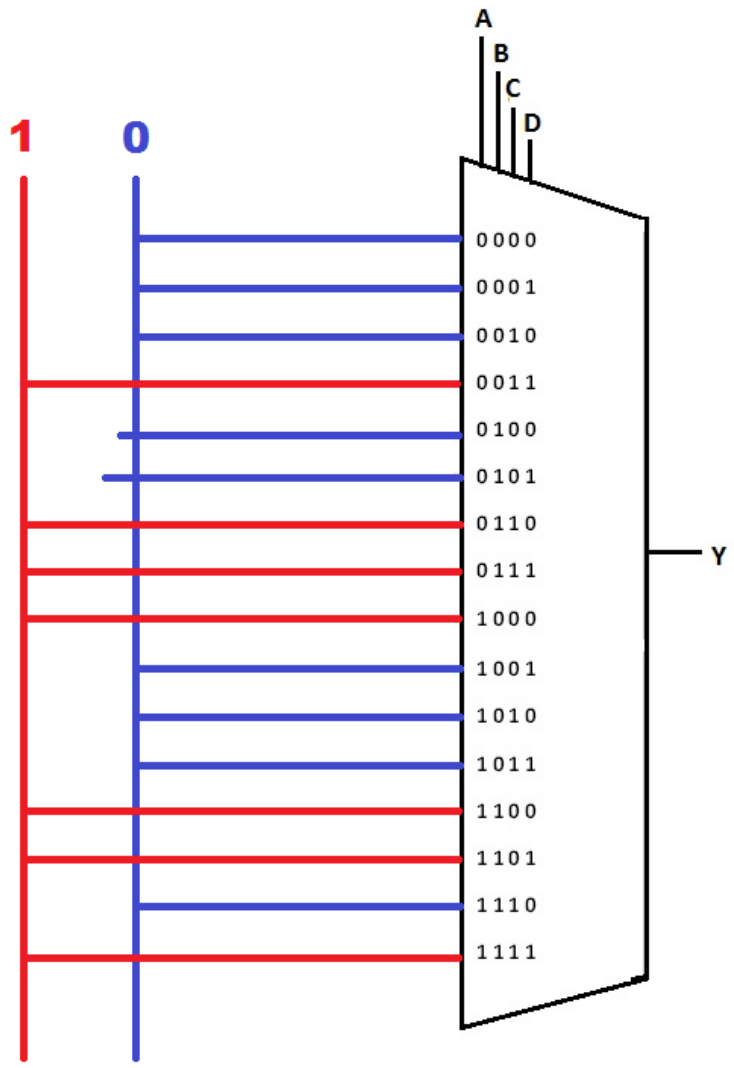
Realizacja funkcji logicznych za pomocą multiplekserów 16/1, 8/1 i 4/1.

Realizacja funkcji logicznej $Y = 3,6,7,8,12,13,15$. Realizacja funkcji na multiplekserze 16/1 jest najprostszym spobem realizacji powyższej funkcji logicznej. Funkcja na wyjściu ma podac jedynkę, gdy jest podana jakaś liczba, z warunków, które spełniają tę funkcję. Do wejść które odpowiadają warunkom, czy też argumentom (nie wiem jak to nazwać) funkcji podłączamy jedynkę, a do reszty zero. W wyniku, wybierając na wejściach adresowych daną liczbą, np.: 7 (0111) to otrzymamy na wyjściu jedynkę.

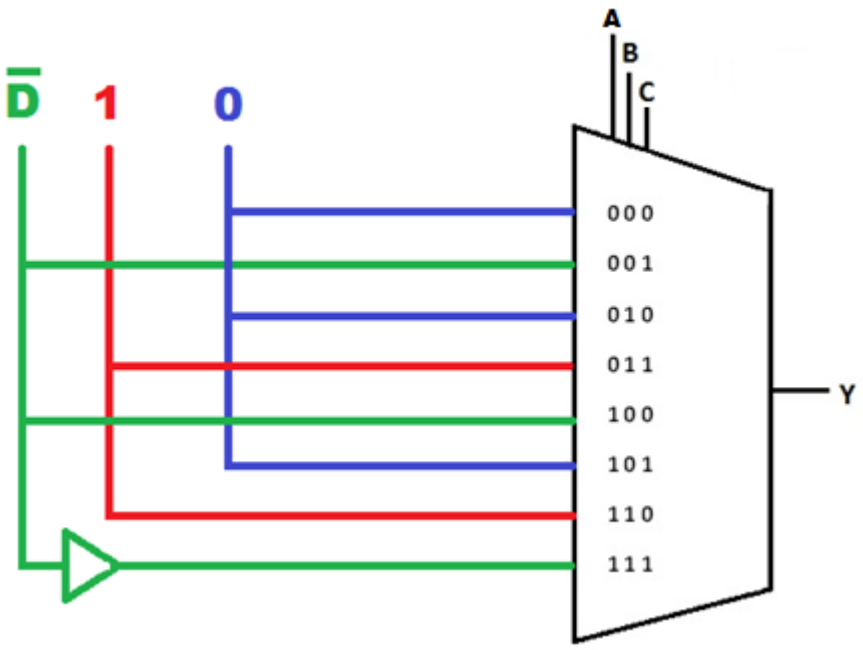


Powyższą funkcję z multipleksera 16/1 przekonwertuję (zminimalizuję?) do multipleksera 8/1.

Wykorzystujemy poniższą tabelę, w której zapisujemy informacje o konkretnych stanach.

„10”	A B C D	Y	Wejścia informacyjne MUX 16/1	Wejścia informacyjne MUX 8/1
0	0 0 0 0	0	0	0
1	0 0 0 1	0	0	
2	0 0 1 0	1	1	$\overline{D}$
3	0 0 1 1	0	0	
4	0 1 0 0	0	0	0
5	0 1 0 1	0	0	
6	0 1 1 0	1	1	1
7	0 1 1 1	1	1	
8	1 0 0 0	1	1	$\overline{D}$
9	1 0 0 1	0	0	
10	1 0 1 0	0	0	0
11	1 0 1 1	0	0	
12	1 1 0 0	1	1	1
13	1 1 0 1	1	1	
14	1 1 1 0	0	0	D
15	1 1 1 1	1	1	

Przedstawienie na rysunku.



Minimalizacja polega na tym, że na wejściach multipleksera 8/1 prócz zera i jedynki pojawić się może wejście adresowane o najmniejszej wadze. W naszym przypadku jest to bit D.

Dalczego w niektórych wejściach jest D lub D nie?

Uzależnione od tego, jak ma zostać zrealizowana funkcja.

Przykład gdzie występuje D:

Liczba 15 spełnia naszą funkcję, adresujemy ją następująco w multiplekserze 8/1 111. Pozostaje nam jednak najmłodszy bit, który sprawi, że nasza funkcja będzie spełniona i poda jedynkę na wyjście. W tym przypadku by była jedynka na wyjściu i adres bitowy zgodny podajemy na wejście bit adresowy D.

Przeciwny przypadek, z zanegowanym D:

Liczba 2 spełnia naszą funkcję. Adres tej liczby to 0100. Jednocześnie w tym miejscu podajemy na wejście adresowe 010 mulitpleksera 8/1 bit adresowy D. Musimy go zanegować ponieważ, w danym adresie D = 0, więc by otrzymać jedynkę na wyjściu, musimy zanegować bit D na wejściu.

Jedynki i zera, jakie występują w niektórych miejscach, są postawione w momencie, gdzie dla dwóch liczb sąsiednich np.: 1 i 2, występuje ta sama wartość niezależna od wejścia adresowego D.

Teraz minimalizujemy powyższą funkcję na multiplekserze 8/1 do multipleksera 4/1.

Dodajemy dodatkową kolumnę do poprzedniej tabeli.

„10”	A B C D	Y	Wejścia informacyjne MUX 16/1	Wejścia informacyjne MUX 8/1	Wejścia informacyjne MUX 4/1
0	0 0 0 0	0	0	0	$C \cdot \overline{D}$
1	0 0 0 1	0	0		
2	0 0 1 0	1	1	$\overline{D}$	
3	0 0 1 1	0	0		
4	0 1 0 0	0	0	0	C
5	0 1 0 1	0	0		
6	0 1 1 0	1	1	1	
7	0 1 1 1	1	1		
8	1 0 0 0	1	1	$\overline{D}$	$\overline{CD}$
9	1 0 0 1	0	0		
10	1 0 1 0	0	0	0	
11	1 0 1 1	0	0		
12	1 1 0 0	1	1	1	$\bar{C} + D$
13	1 1 0 1	1	1		
14	1 1 1 0	0	0	D	
15	1 1 1 1	1	1		

Dokonujemy czterech minimalizacji funkcji zaznaczonych na rysunku poniżej

„10”	A B C D	Y	Wejścia informacyjne MUX 16/1	Wejścia informacyjne MUX 8/1	Wejścia informacyjne MUX 4/1
0	0 0 0 0	0	0	0	$C \cdot \overline{D}$
1	0 0 0 1	0	0		
2	0 0 1 0	1	1	$\overline{D}$	
3	0 0 1 1	0	0		
4	0 1 0 0	0	0	0	C
5	0 1 0 1	0	0		

Dokonując minimalizacji, otrzymujemy funkcję dla wejść multipleksera 4/1.

Minimalizując wyżej zaznaczony fragment otrzymujemy:

D / C	0	1
0	0	0
1	1	0

$C \cdot \overline{D}$

Wykonujemy kolejne działania dla kolejnych kombinacji bitów CD. W ten sposób otrzymujemy wejścia multipleksera 4/1.

Wysunek poniżej obrazuje nasz wynikowy, maksymalnie zminimalizowany multiplekser 16/1 do multipleksera 4/1.