НОВ БЪЛГАРСКИ УНИВЕРСИТЕТ

Компютърни архитектури
CSCB008

Приложения на комбинационните логически схеми в компютърните архитектури

доц. д-р Ясен Горбунов

2021

Съдържание

Елементи на микроархитектурата

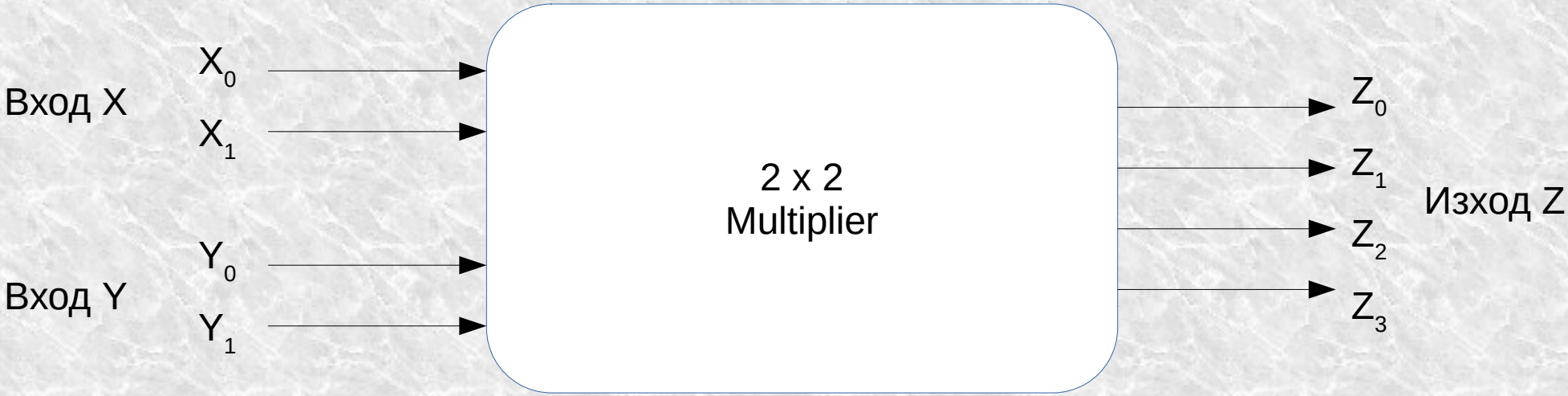
- Полусуматор
  - Пълен суматор
    - Суматор със знак
- Умножител
- Мултиплексор
- Демултиплексор
- Декодер

част 1

част 2

Умножител

Проектиране на 2-битово устройство за умножение



Проектиране на 2-битово устройство за умножение

таблица на истинност

Номер на набора	X x Y = Z	Входове				Изход			
		X		Y		Z			
		X_1	X_0	Y_1	Y_0	Z_3	Z_2	Z_1	Z_0
0	0 x 0 = 0	0	0	0	0	0	0	0	0
1	0 x 1 = 0	0	0	0	1	0	0	0	0
2	0 x 2 = 0	0	0	1	0	0	0	0	0
3	0 x 3 = 0	0	0	1	1	0	0	0	0
4	1 x 0 = 0	0	1	0	0	0	0	0	0
5	1 x 1 = 1	0	1	0	1	0	0	0	1
6	1 x 2 = 2	0	1	1	0	0	0	1	0
7	1 x 3 = 3	0	1	1	1	0	0	1	1
8	2 x 0 = 0	1	0	0	0	0	0	0	0
9	2 x 1 = 2	1	0	0	1	0	0	1	0
10	2 x 2 = 4	1	0	1	0	0	1	0	0
11	2 x 3 = 6	1	0	1	1	0	1	1	0
12	3 x 0 = 0	1	1	0	0	0	0	0	0
13	3 x 1 = 3	1	1	0	1	0	0	1	1
14	3 x 2 = 6	1	1	1	0	0	1	1	0
15	3 x 3 = 9	1	1	1	1	1	0	0	1

Проектиране на 2-битово устройство за умножение

$$Z_0 = \sum_0^{15}(5,7,13,15)$$

$$Z_1 = \sum_0^{15}(6,7,9,11,13,14)$$

$$Z_2 = \sum_0^{15}(10,11,14)$$

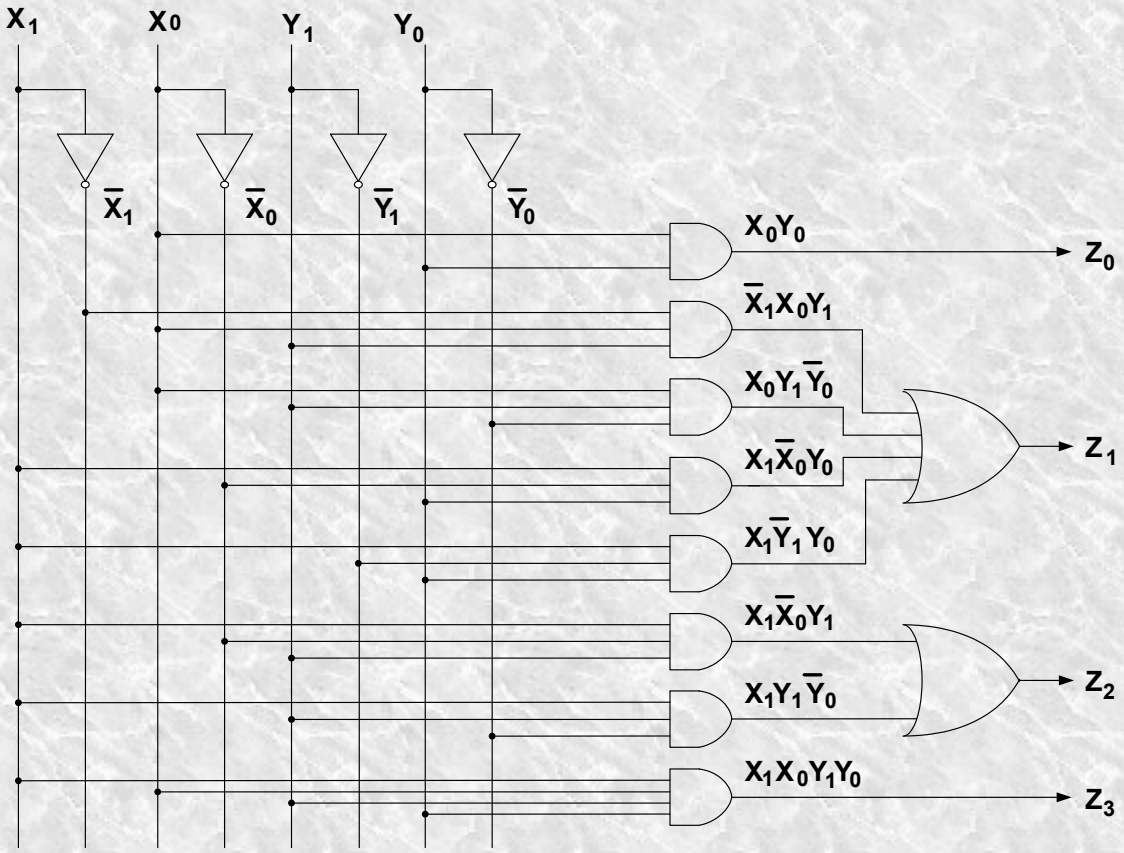
$$Z_3 = \sum_0^{15}(15)$$

$$Z_0 = X_0 \cdot Y_0$$

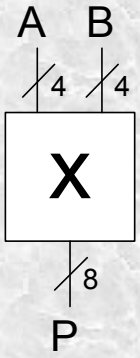
$$Z_1 = \overline{X_1} \cdot X_0 \cdot Y_1 + X_0 \cdot Y_1 \cdot \overline{Y_0} + X_1 \cdot \overline{X_0} \cdot Y_0 + X_1 \cdot \overline{Y_1} \cdot Y_0$$

$$Z_2 = X_1 \cdot \overline{X_0} \cdot Y_1 + X_1 \cdot Y_1 \cdot \overline{Y_0}$$

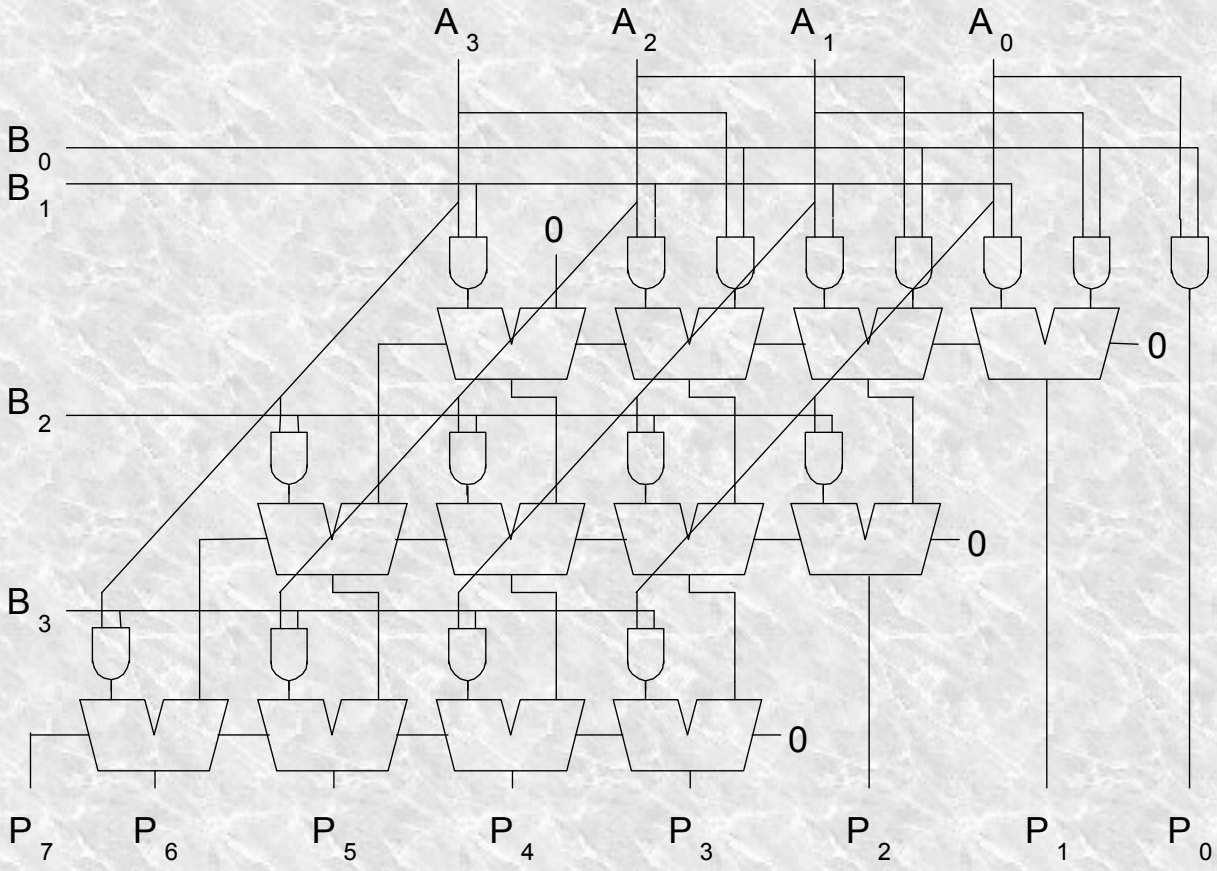
$$Z_3 = X_1 \cdot X_0 \cdot Y_1 \cdot Y_0$$



Проектиране на 4-битово устройство за умножение



$$
\begin{array}{rcccccccc}
 & & & A_3 & A_2 & A_1 & A_0 & & \\
\times & & & B_3 & B_2 & B_1 & B_0 & & \\
\hline
 & & & A_3B_0 & A_2B_0 & A_1B_0 & A_0B_0 & & \\
 & & A_3B_1 & A_2B_1 & A_1B_1 & A_0B_1 & & & \\
 & A_3B_2 & A_2B_2 & A_1B_2 & A_0B_2 & & & & \\
+ & A_3B_3 & A_2B_3 & A_1B_3 & A_0B_3 & & & & \\
\hline
P_7 & P_6 & P_5 & P_4 & P_3 & P_2 & P_1 & P_0 &
\end{array}
$$



Умножител

Други начини за проектиране и използване на умножители

1. Комбинационни логически схеми – бързи, но обемисти

2. Последователно изместване и събиране (shift & add) – крайни автомати

3. Специални алгоритми (Booth, Dadda, Wallace Tree...)

4. Памети – lookup tables

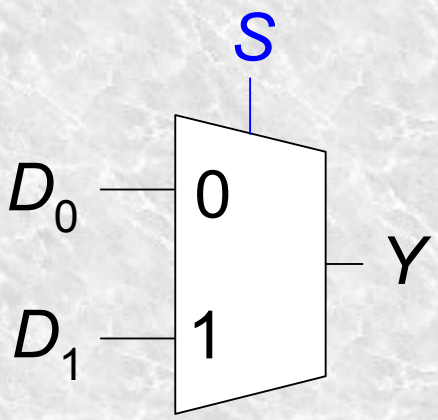
5. Вградени в схемите умножителни блокове (еднотактови 18x18 и др.)

2-битов мултиплексор

- Свързване на един от няколко възможни входа към единствен изход
- Селекторен вход (S), чрез който се избира активния вход – $\log_2 N$ бита

Приложение

синтез на логически функции
избор на логически условия (превключване)
извличане на флагове от регистър



S	D_1	D_0	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

S=0

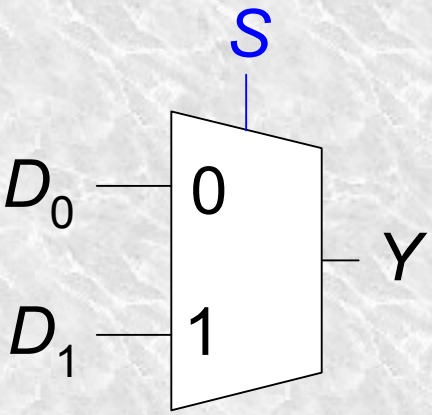
S	D_1	D_0	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1

S=1

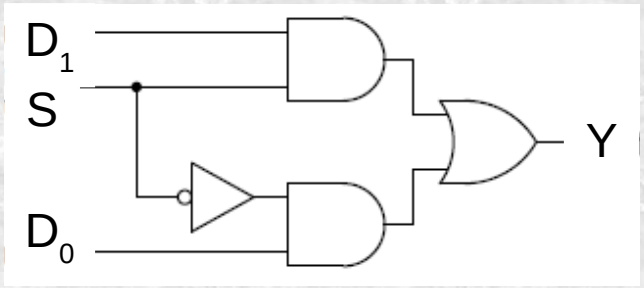
S	D_1	D_0	Y
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Мултиплексор

2-битов мултиплексор



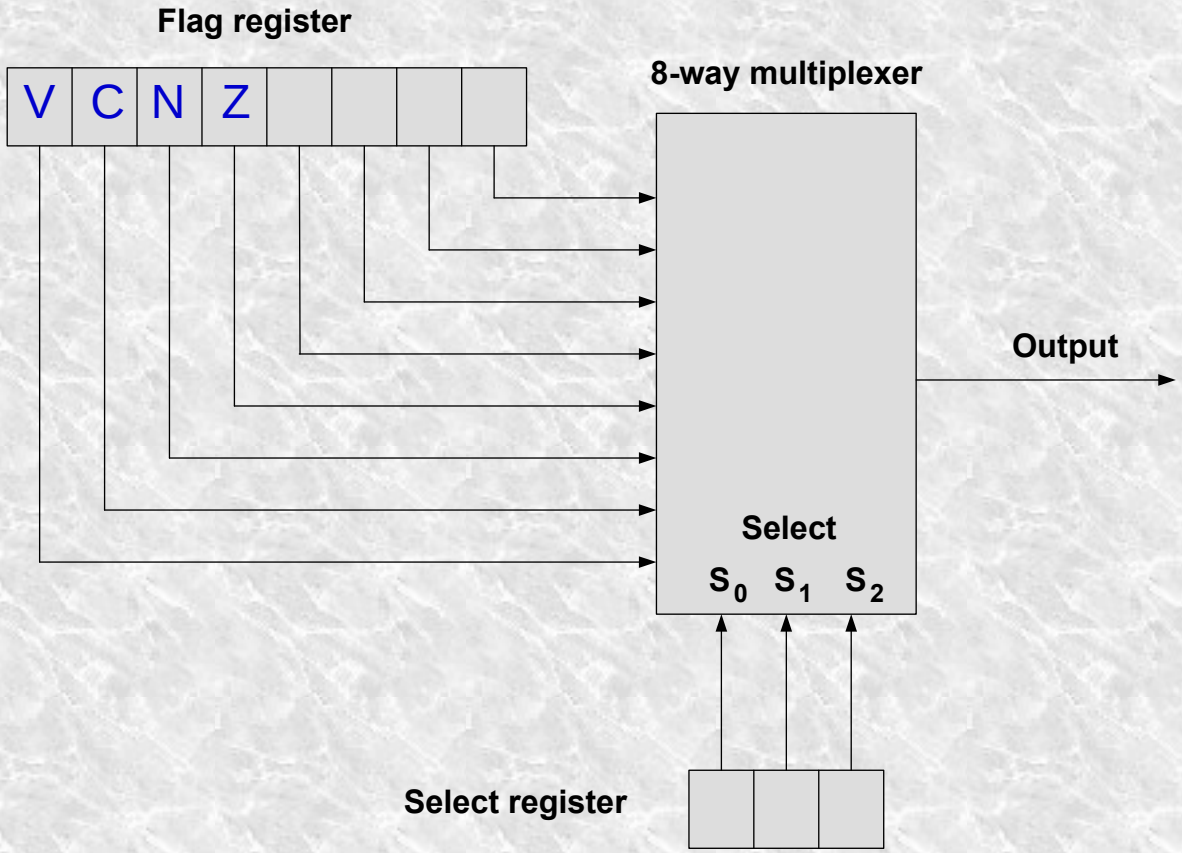
S	Y
0	D_0
1	D_1



$$Y = S \cdot D_1 + \overline{S} \cdot D_0$$

8-битов мултиплексор – извличане на флагове от регистър

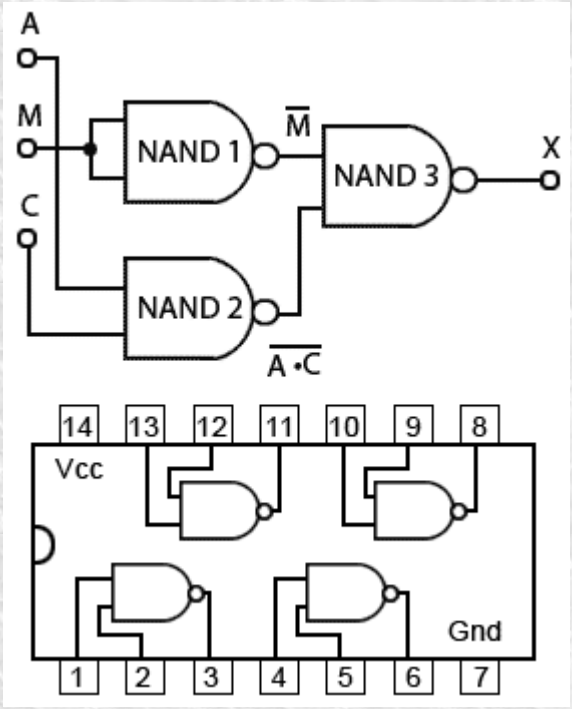
V – o**V**erflow
C – **C**arry
N – **N**egative
Z – **Z**ero

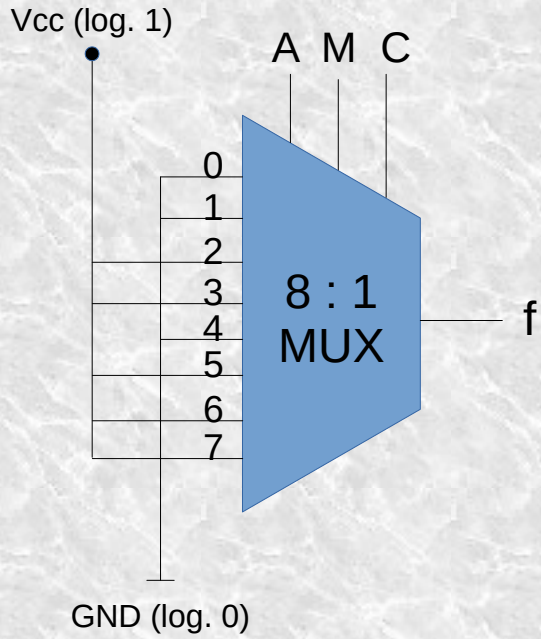


Синтез на логически функции чрез мултиплексор

A	M	C	f
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

$$f = \sum (2,3,5,6,7)$$

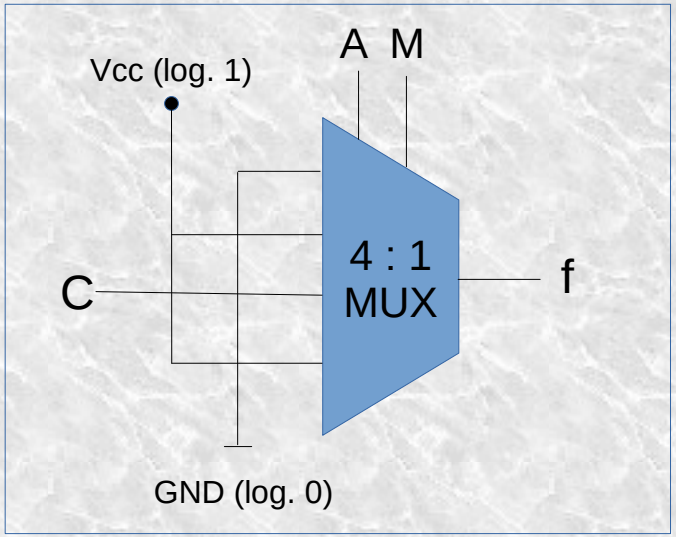




Синтез на логически функции чрез мултиплексор

A	M	C	f
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

A	M	f
0	0	0
0	1	1
1	0	C
1	1	1



$$f = \sum (2,3,5,6,7)$$

Демултиплексор

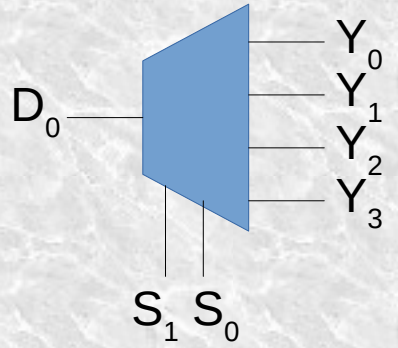
- функция, обратна на мултиплексора
  - свързване на единствен вход към един от няколко възможни изхода
  - селекторен вход (S), чрез който се избира активния изход – $\log_2 N$ бита

Приложение
- избор на устройство (device selector)
- реализация на логически схеми
- **декодиране на инструкции**

таблица на истинност – демултиплексор 1:4

D_0	S_1	S_0	Y_0	Y_1	Y_2	Y_3
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	0
0	1	1	0	0	0	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	0	1

S_1	S_0	Y_0	Y_1	Y_2	Y_3
0	0	D_0	0	0	0
0	1	0	D_0	0	0
1	0	0	0	D_0	0
1	1	0	0	0	D_0



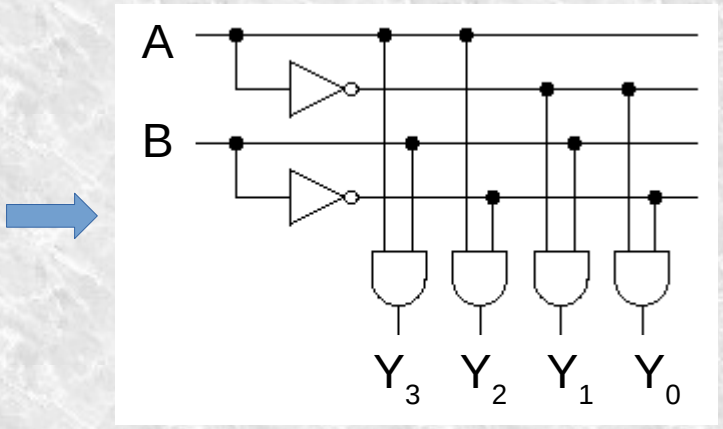
Декодер

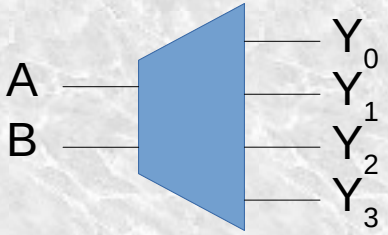
- Свързване на n входа към 2^n изхода

Приложение
- реализация на логически схеми
- **декодиране на инструкции**
- реализация на мултиплексори

Пример на декодер 2:4 – таблица на истинност

A	B	Y_0	Y_1	Y_2	Y_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

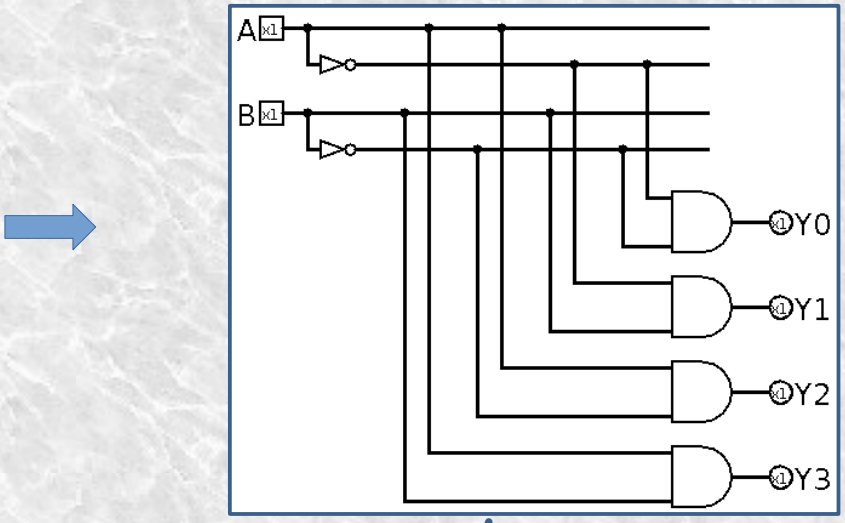




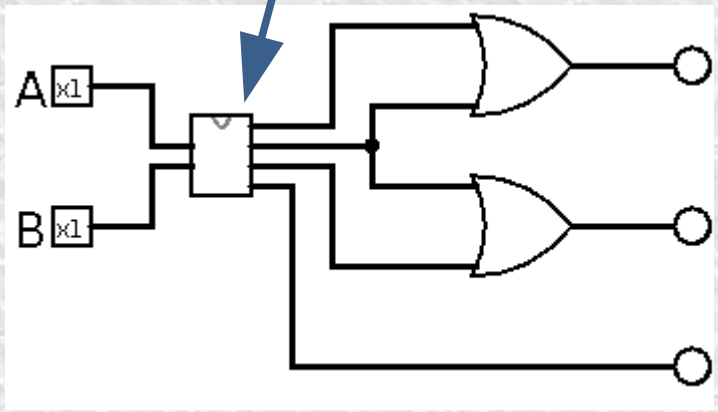
Декодер – пример за синтез на логическо управление на светофар

Пример на декодер 2:4 – таблица на истинност

A	B	Y_0	Y_1	Y_2	Y_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1



A	B	Цвят
0	0	червен
0	1	червен + жълт
1	0	жълт
1	1	зелен



червен

жълт

зелен

Демултиплексор

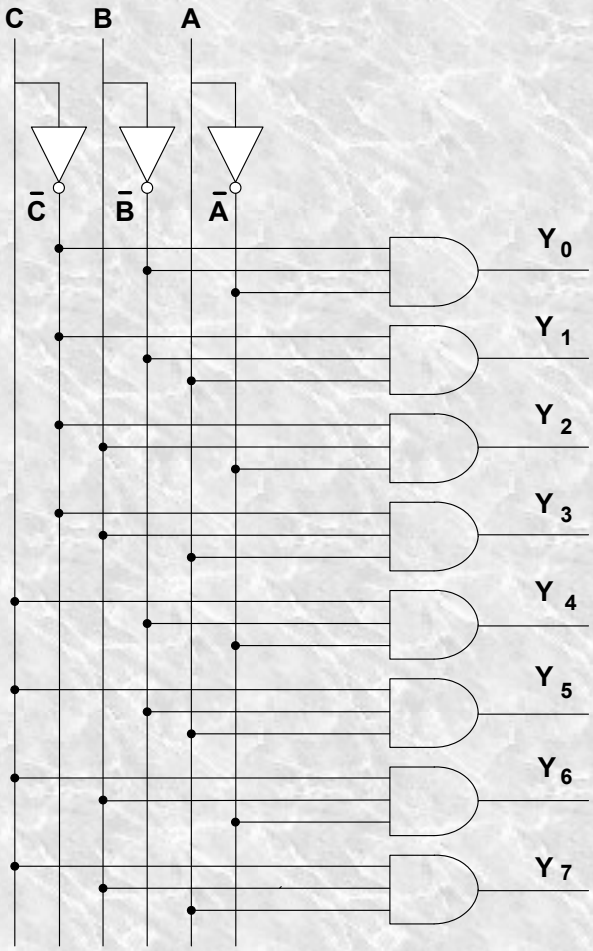
1 от 8 демултиплексор (3:8 декодер)

таблица на истинност

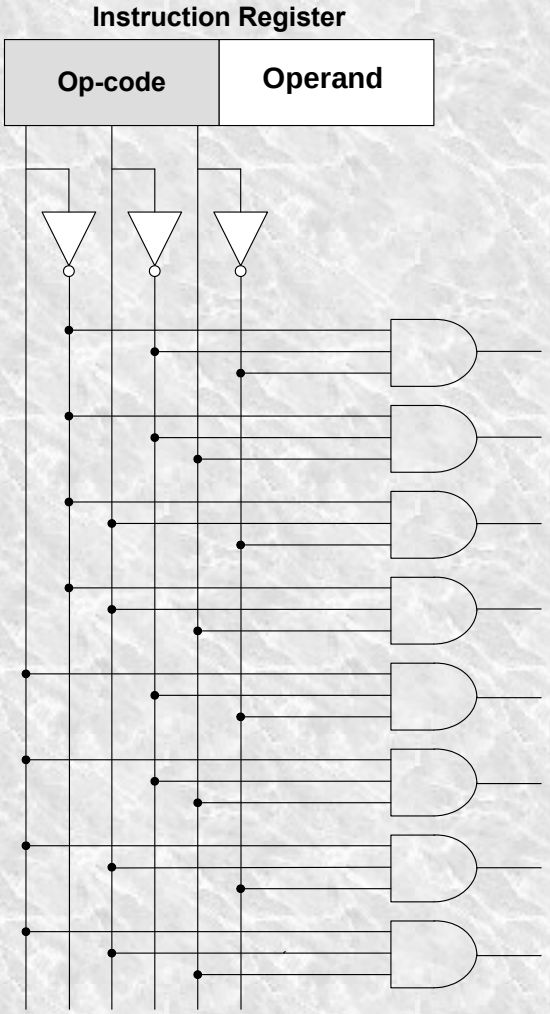
Номер на набора	Входове (селектор)			Изходи							
	A	**B**	**C**	Y_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7
0	**0**	**0**	**0**	**1**	0	0	0	0	0	0	0
1	**0**	**0**	**1**	0	**1**	0	0	0	0	0	0
2	**0**	**1**	**0**	0	0	**1**	0	0	0	0	0
3	**0**	**1**	**1**	0	0	0	**1**	0	0	0	0
4	**1**	**0**	**0**	0	0	0	0	**1**	0	0	0
5	**1**	**0**	**1**	0	0	0	0	0	**1**	0	0
6	**1**	**1**	**0**	0	0	0	0	0	0	**1**	0
7	**1**	**1**	**1**	0	0	0	0	0	0	0	**1**

Демултиплексор

1 от 8 демултиплексор (3:8 декодер)



декодиране на компютърна инструкция



Инструкции

- Add
- Subtract
- Load
- Store
- Branch on zero
- Branch on not zero
- Branch unconditionally
- Stop