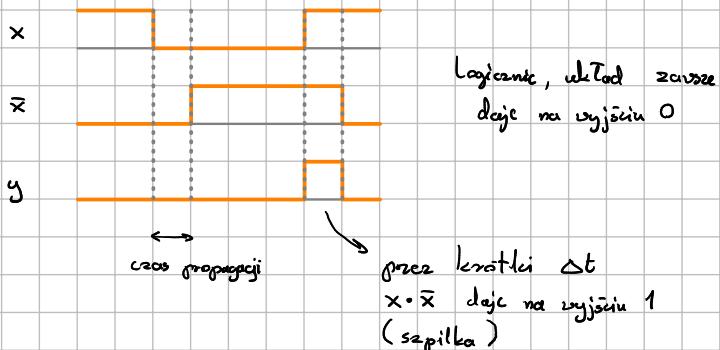


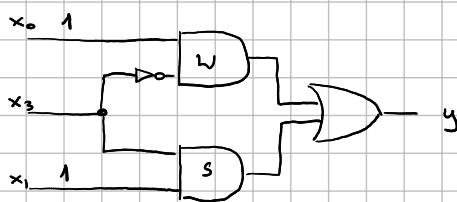
Hazard statyczny - zjawisko fizyczne



fizyczne elementy charakteryzujące czas propagacji (rzędu 1 na)



Impulsy hazardowe prawidłowo pojawiają się w układach kombinacyjnych



x_3, x_0	00	01	11	10
x_1, x_2	00	01	11	10
00	0	1	1	0
01	0	1	1	0
11	0	0	1	1
10	0	0	1	1

mijające wystąpienia szpilków widać w mapie Karnaugha

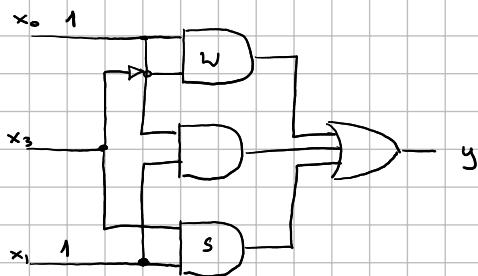
$$y = \overline{x}_3 x_0 + x_3 x_1$$

Rezygnacja z postaci minimalnej pozwala na uniknięcie hazardu

x_3, x_0	00	01	11	10
x_1, x_2	00	01	11	10
00	0	1	1	0
01	0	1	1	0
11	0	0	1	1
10	0	0	1	1

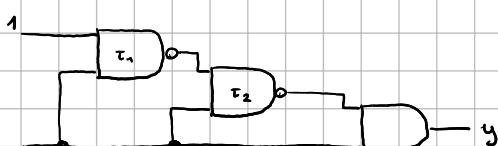
$$y = \overline{x}_3 x_0 + x_3 x_1 + x_1 x_0$$

dodatkowy implicant zapobiega powtarzaniu hazardu



Hazard dynamiczny

ten sam sygnał dodany do kolejnych 3 poziomów układu



należy uniknąć faktoryzacji

Układy wielowarstwowe



$$Y = F(X)$$

$$X = (x_{m-1}, \dots, x_0) \rightarrow m$$

$$Y = (y_{n-1}, \dots, y_0) \rightarrow n$$

$$F = (f_{n-1}, \dots, f_0) \rightarrow n$$

$$Y = \begin{cases} y_0 = f_0(x_{m-1}, \dots, x_0) \\ y_1 = f_1(x_{m-1}, \dots, x_0) \\ \dots \\ y_{n-1} = f_{n-1}(x_{m-1}, \dots, x_0) \end{cases}$$

w funkcji, każda od tego samego argumentu

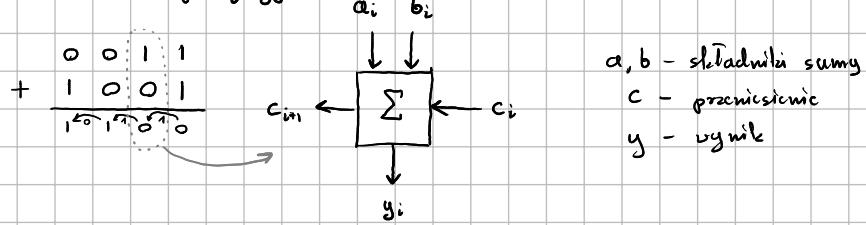
Minimizacja rozłączna i roztaczna

Minimizacja rozłączna - oddzielne każdą z funkcji składowych

Przeprowadzając minimizację każdej funkcji oddzielnie otrzymamy układ, który może jeszcze uproszczyć wykorzystując implikanty powtarzające się między funkcjami (i ostatecznie bramki)

- 1) Minimizacja każdej z funkcji y_i na mapie Karnaugha
- 2) Znalezienie wszystkich kombinacji iloczynów y_i (dla 3 $\rightarrow y_1y_2, y_1y_3, y_1y_2y_3, y_2y_3$) i minimizacja na mapie Karnaugha
- 3) Wpisanie wszystkich implikantów do tabeli Quine'a z podziałem na funkcje składowe
- 4) Znalezienie minimalnego połączenia i zapisanie każdej z funkcji składowych

Sumowanie jedno pozycyjne



a, b - składowe sumy
 c - przesunięcie
 y - wynik

tabela prawdy

a_i	b_i	c_i	y_i	c_{in}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

→ zespół funkcji

$$y_i = a_i \oplus b_i \oplus c_i$$

$$c_{i+1} = a_i b_i + a_i c_i + b_i c_i$$

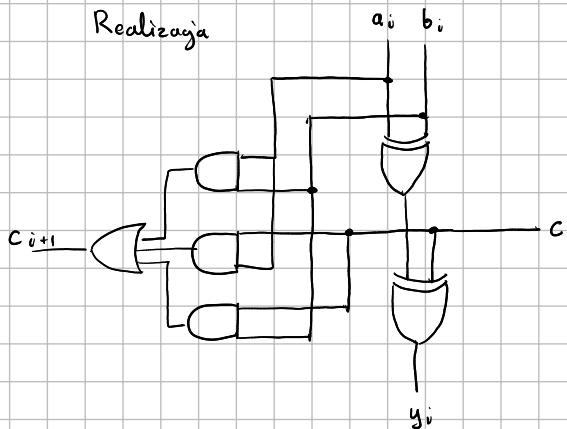
$a_i b_i$	00	01	11	10
c_i	0	0 1 0 1		
	1	1 0 1 0		

y_i

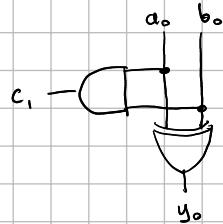
$a_i b_i$	00	01	11	10
c_i	0	0 0 1 0		
	1	0 1 1 1		

c_{i+1}

Realizacja

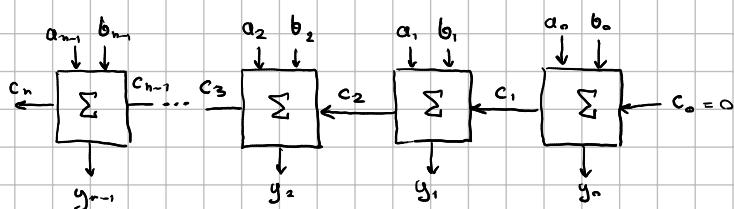


przyjmując brzegowy kiedy $c_0 = 0$
można znacznie uprościć



Sumator iteracyjny n - pozycyjny

Gatunek blok wytwarzający portówkę od pośrednich liczb razy podając na wejścia kolejne bity i Tocząc sygnały przesunięcia



Układy iteracyjne

Rozwiązywanie po krokach mniejsze problemy

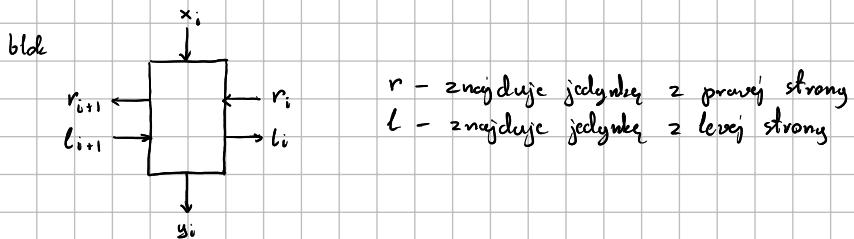
Projektowanie układu iteracyjnego

- Zdekomponować układ dla pojedynczego wejścia – jeden blok
- Określić i wyznaczyć przekształcenia pomiędzy blokami (tabela prawdy)
- Wyznaczyć funkcję wyjścia i funkcję przekształceń
- Połączycy bloki

Przykładowy układ

daje '1' na wszystkich pozycjach między sterującymi '1' wejściem

IN	0	0	1	0	1	1	0	0	0	1	0	0	0
OUT	0	0	1	1	1	1	1	1	1	1	0	0	0

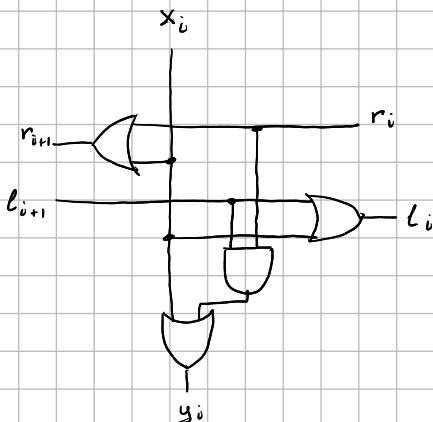


x_i	l_{i+1}	r_i	y_i	l_i	r_{i+1}
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	1	1	1
1	0	0	1	1	1
1	0	1	1	1	1
1	1	0	1	1	1
1	1	1	1	1	1

$$y_i = x_i + l_{i+1} r_i$$

$$l_i = x_i + l_{i+1}$$

$$r_{i+1} = x_i + r_i$$



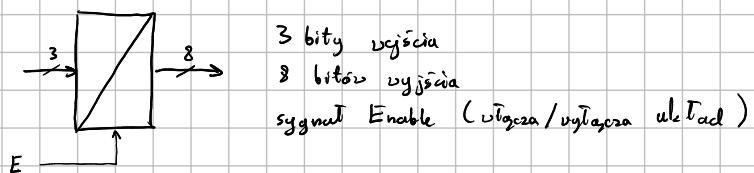
Bloki funkcyjne (kombinacyjne)

Gotowe wstędy realizujące często powtarzające się przydatne funkcje

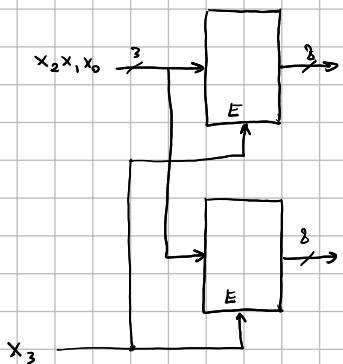
Konwertyatory kodu (kodery i dekodery)

- Kod NKB na kod Graya
- Kod Graya na kod NKB
- Kod NKB na kod '1 z n'
- itp.

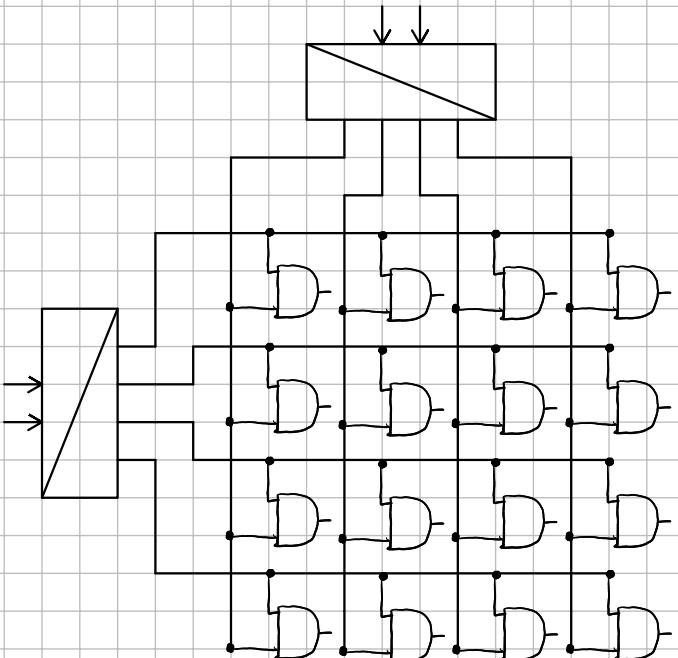
oznaczenia



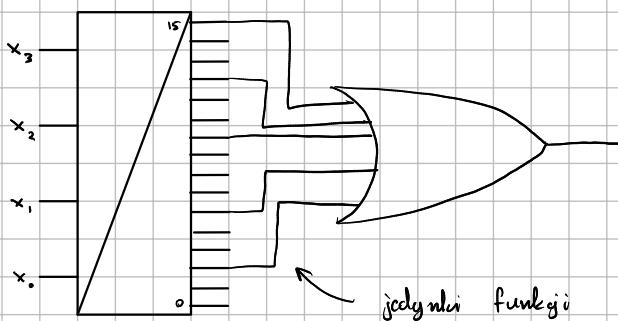
Dekoder 4 na 16 złożony z dwóch 3 na 8



Dekoder wsportrzędnościowy



Realizacja funkcji logicznej h z 4-bitowych na dekoderze 1:16

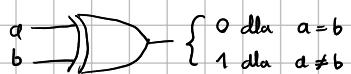


Wyświetlacz 7-segmentowy

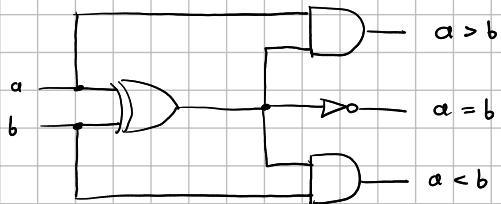
f	a	0	1	2	3	
	b	4	5	6	7	
c	c	8	9	A	B	
	d	C	D	E	F	

→ dekoder z 4-bitowym wejściem

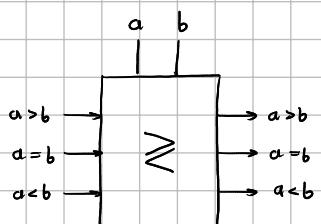
Komparator



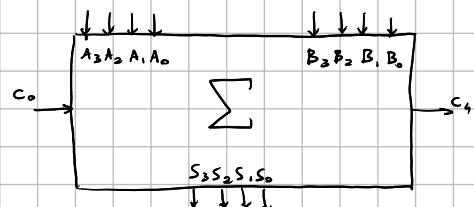
Komparator jednabitowy



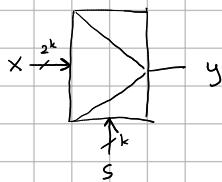
Blk interacyjny komparatora



Sumator 4-bitowy

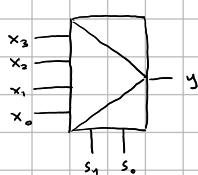


Multiplexer



Wybiera jedno z wejść (x)
za pomocą sygnału sterującego (s)
i wyprowadza je na wyjście (y)

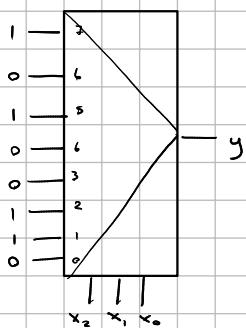
multiplexer $4 \rightarrow 1$



s_1	s_0	y
0	0	x_0
0	1	x_1
1	0	x_2
1	1	x_3

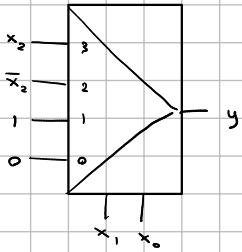
Realizacja funkcji logicznej 3 zmiennych na multipleserze $8 \rightarrow 1$

x_2	x_1	x_0	00	01	11	10
0	0	1	0	1		
1	0	1	1	0		



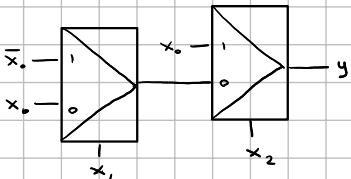
Realizacja funkcji logicznej 3 zmiennych na multipleserze $6 \rightarrow 1$

x_2	x_1	x_0	00	01	11	10
0	0	1	0	1		
1	0	1	1	0		



Realizacja funkcji logicznej 3 zmiennych na multipleserach $2 \rightarrow 1$

x_2	x_1	x_0	00	01	11	10
0	0	1	0	1		
1	0	1	1	0		



Na jednym MUX $2^n \rightarrow 1$

zawsze można zrealizować funkcję n -zmiennych
(stałe 0 albo 1 na wyjściach informacyjnych)

Na jednym MUX $2^{n-1} \rightarrow 1$

zawsze można zrealizować funkcję n -zmiennych
(stałe 0 albo 1 lub jedna ze zmiennych na
wyjściach informacyjnych)

Na jednym MUX $2^{n-2} \rightarrow 1$

nie zawsze można zrealizować funkcję n -zmiennych

Wybór wojsc sterujących

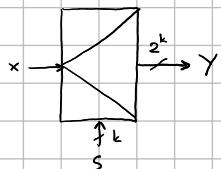
- 1) Liczyt literaly w APN
 - 2) Wybra te wojsc, które mają najczesciej
- jeśli realizacja jest możliwa to da dobrą odpowiedź

$x_3 x_2$	00	01	11	10
00	1 1 1 0	0 1 1 0	0 0 1 1	0 0 1 1
01	0 1 1 0	0 0 1 1	0 0 1 1	0 0 1 1
11	0 0 1 1	0 0 1 1	0 0 1 1	0 0 1 1
10	0 0 1 1	0 0 1 1	0 0 1 1	0 0 1 1

$$y = x_3 x_1 + \bar{x}_3 x_0 + \bar{x}_3 \bar{x}_2 \bar{x}_1$$

$x_3 \rightarrow 3$
 $x_2 \rightarrow 1$
 $x_1 \rightarrow 2$
 $x_0 \rightarrow 1$

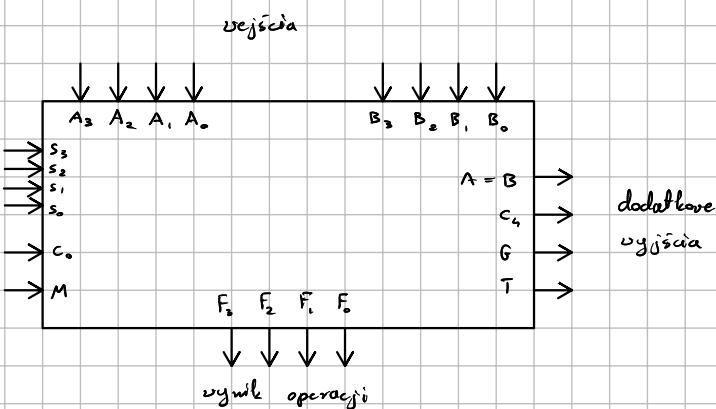
Demultiplexer



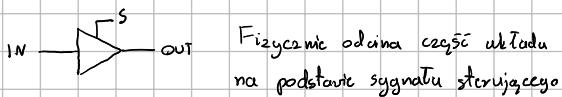
Przekształca wojsc (x) na jedno z wojsc (y) wybrane sygnałem sterującym (s)

Jednostka arytmetyczno-logiczna (ALU)

(jest układem kombinacyjnym)



Bramka trójstanowa



Pamięć ROM (read-only memory)

