

## Zapisy funkcji boolowskiej przydatne przy projektowaniu

- tabela prawdy → pomocna przy przechodzeniu od zapisu słownego do formalnego
- postać dziesiętna → tabelaryczny zapis
- Mapa Karnaugh → pomocna przy minimalizacji
- postać algebraiczna → zapis boolowski
- schemat logiczny na bramkach

## Zestaw funkcjonalnie pełny

Każdą funkcję logiczną można wyrazić stosując wyłącznie NAND albo wyłącznie NOR

## Postać sumacyjna APN

dla funkcji 3 zmiennych  $y = \Sigma(3, 4, 5, 6, 7)$

$$y = \bar{x}_2 x_1 x_0 + x_2 \bar{x}_1 \bar{x}_0 + x_2 \bar{x}_1 x_0 + x_2 x_1 \bar{x}_0 + x_2 x_1 x_0$$

Implikant (np.  $\bar{x}_2 x_1 x_0$ ) – taki iloczyn zmiennych, że dla wszystkich wektorów binarnych, dla których przyjmuje wartość 1, funkcja też przyjmuje wartość 1

## Postać iloczynowa KPN

dla funkcji 3 zmiennych  $y = \Pi(0, 1, 2)$

$$y = (x_2 + x_1 + x_0)(x_2 + x_1 + \bar{x}_0)(x_2 + \bar{x}_1 + x_0)$$

Implikant (np.  $x_2 + \bar{x}_1 + x_0$ ) – taka suma zmiennych, że dla wszystkich wektorów binarnych, dla których przyjmuje wartość 0, funkcja też przyjmuje wartość 0

## Minimalizacja funkcji

przebiegi twierdzenia algebraiczne

$$y = \bar{x}_2 x_1 x_0 + x_2 \bar{x}_1 \bar{x}_0 + x_2 \bar{x}_1 x_0 + x_2 x_1 \bar{x}_0 + x_2 x_1 x_0$$

$$y = \bar{x}_2 x_1 x_0 + x_2 \bar{x}_1 + x_2 x_1$$

$$y = \bar{x}_2 x_1 x_0 + x_2$$

$$y = x_1 x_0 + x_2$$

prosto pozbawiania

minimalizacja na mapie Karnaugh

$$y = \sum(3, 4, 5, 6, 7)$$

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

Kod Grupa

2 sąsiadujące słowa kodowe różniące się tylko 1 bitem

przeruszc i ostatnie słowo kodowe różni się tylko 1 bitem

→ Sklejanie jedynek → zapisuje te bity, które nie ulegają zmianie

pola mapy sąsiadujące przez ścianę (np. 4 z 6)

$x_1, x_0$   
 $x_2$  } afirmacje

$x_2 + x_1 x_0 \rightarrow$  minimalna postać funkcji

po użyciu wszystkich jedynek zostaje postać minimalna

Mapa Karnaugh 4 zmiennych

$$y = \sum(1, 3, 5, 6, 9, 11, 12, 13, 14)$$

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

$\bar{x}_1, x_0$

$x_2 x_1, \bar{x}_0$

$x_3 x_2, \bar{x}_1$

$\bar{x}_2 x_0$

→ sąsiadstwo jest też przez krawędzie (torus)

grupować można maksymalnie  $2^n \times 2^m$

$$y = \bar{x}_1 x_0 + x_2 x_1 \bar{x}_0 + x_3 x_2 \bar{x}_1 + \bar{x}_2 x_0$$

Dla KPN staleja się zero

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

$$(\bar{x}_2 + \bar{x}_1 + \bar{x}_0)$$

$$(x_3 + x_1 + x_0)$$

$$(x_2 + x_0)$$

KPN:  $y = (\bar{x}_2 + \bar{x}_1 + \bar{x}_0)(x_3 + x_1 + x_0)(x_2 + x_0)$

Dla 5 zmiennych

$x_2, x_1, x_0$ $x_4, x_3$	000	001	011	010	110	111	101	100
00								
01								
11								
10								

symetria względem osi  
symetrii na środku

Dla więcej niż 5 zmiennych mapa robi się zbyt skomplikowana

## Metoda Quine'a - McCluskey'a

→ powszechnie implementowany algorytm

- 1) Grupowanie implikantów prostych  
wg. liczby jedynek
- 2) Sklejanie implikantów
- 3) Tabela Quine'a
- 4) Wybór minimalnego zbioru  
implikantów pokrywającego  
wszystkie jedynki funkcji

Dla  $y = \sum(3, 4, 5, 6, 7)$

- 1) Zapisuję binarnie jedynki funkcji

3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

grupuję wg. liczby jedynek i zapiskę

4	1	0	0
3	0	1	1
5	1	0	1
6	1	1	0
7	1	1	1

- 2) Sklejanie implikantów

✓ 4	1	0	0
✓ 3	0	1	1
✓ 5	1	0	1
✓ 6	1	1	0
✓ 7	1	1	1

✓ 4,5	1	0	-
✓ 4,6	1	-	0
3,7	-	1	1
✓ 5,7	1	-	1
✓ 6,7	1	1	-

4,5,6,7	1	-	-
<del>4,5,6,7</del>	1	-	-

→ duplikat

Każdy wiersz z grupy  $n$  porównuję z każdym z grupy  $n+1$

Jeśli różnią się 1 bitem, to przepisuję dalej ze znakiem - a pierwsze wiersze odznaczam

Tak samo z kolejną tabelą ale - muszą być na jednakowych miejscach

Postaram się tak długo aż zostanie tylko 1 albo nie ma już dalej grupować

Nieodznaczone wiersze trafiają do tabeli Quine'a

- 3)

implikanty	3	4	5	6	7
	011	100	101	110	111
3,7	-	1	1		X
4,5,6,7	1	-	-		X

Implikant zasadniczy - jedyny, który pokrywa jakąś jedynkę, musi trafić do końcowego wyniku

Zaznaczam jedynki pokrywane przez implikanty  
Znajduję minimalne pokrycie

$$y = x_1 x_0 + x_2$$

Funkcja nie u petni dzieslona

$$y = \sum(2, 4, 12, 14, 15, (1, 3, 6, 8))$$

→ wartości dla których nie jest dziesiętna

$x_1 x_0$ $x_3 x_2$	00	01	11	10
00	0	-	-	1
10	1	0	0	-
11	1	0	1	1
01	-	0	0	0

- to wartości niedziesiętna, można je skreślić ale nie trzeba

$$\bar{x}_3 \bar{x}_2 x_1 + x_1$$

metoda Quine'a - McCluskey'a dla funkcji nie u petni dziesiętna

$$y = \sum(2, 4, 12, 14, 15, (1, 3, 6, 8))$$

1	0 0 0 1	✓ 1	0 0 0 1	1, 3	0 0 - 1	4, 6, 12, 14	- 1 - 0
2	0 0 1 0	✓ 2	0 0 1 0	2, 3	0 0 1 -	<del>4, 6, 12, 14</del>	<del>1 - 0</del>
3	0 0 1 1	✓ 4	0 1 0 0	2, 6	0 - 1 0		
4	0 1 0 0	✓ 8	1 0 0 0	✓ 4, 6	0 1 - 0		
6	0 1 1 0	✓ 3	0 0 1 1	✓ 4, 12	- 1 0 0		
8	1 0 0 0	✓ 6	0 1 1 0	3, 12	1 - 0 0		
12	1 1 0 0	✓ 12	1 1 0 0	✓ 6, 14	- 1 1 0		
14	1 1 1 0	✓ 14	1 1 1 0	✓ 12, 14	1 1 - 0		
15	1 1 1 1	✓ 15	1 1 1 1	14, 15	1 1 1 -		

		2 0010	4 0100	12 1100	14 1110	15 1111
1, 3	0 0 - 1					
2, 3	0 0 1 -	X				
2, 6	0 - 1 0	X				
4, 12	1 - 0 0			X		
14, 15	1 1 1 -				X	X
4, 6, 12, 14	- 1 - 0		X	X	X	

✓ } dodatkowe litery

→ wiersz zdominowany przez - 1 - 0

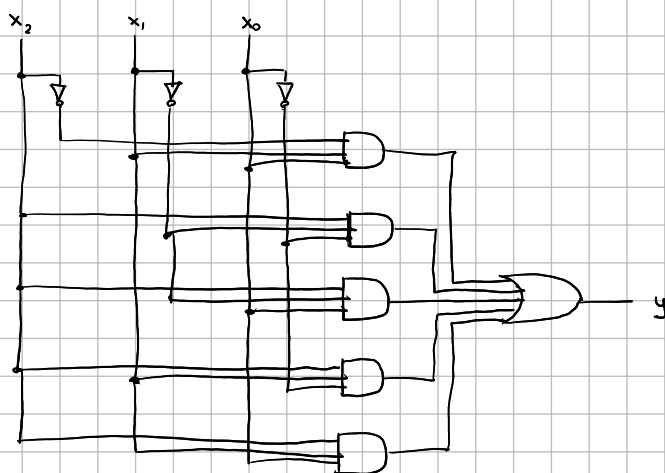
$$y = \bar{x}_3 \bar{x}_2 x_1 + x_3 x_2 x_1 + x_2 \bar{x}_0$$

## Realizacja postaci minimalnej

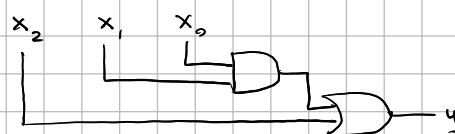
- najprostszą na bramkach logicznych
- na bramkach AND, OR, NOT - sposób podstawowy
- realizacja na zestawie funkcjonalnie pełnym
  - postać APN - na bramkach NAND
  - postać KPN - na bramkach NOR
- realizacje postaci funkcji z faktoryzacją zwykle są efektywne

$$y = \Sigma(3, 4, 5, 6, 7)$$

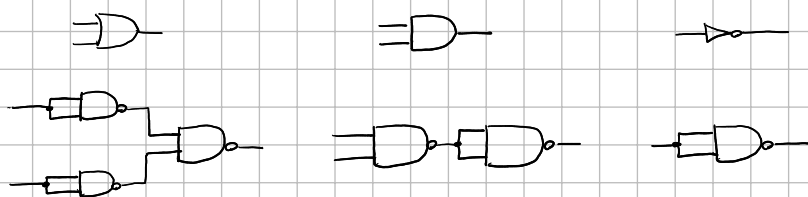
$$y = \bar{x}_2 x_1 x_0 + x_2 \bar{x}_1 \bar{x}_0 + x_2 \bar{x}_1 x_0 + x_2 x_1 \bar{x}_0 + x_2 x_1 x_0$$



$$y = x_1 x_0 + x_2 \quad \text{— postać minimalna}$$



## Zestaw funkcjonalnie pełny NAND



## Zestaw funkcjonalnie pełny NOR

