Booleova algebra - výrokový počet

Michal Šerý

Principy počítačů

Obsah

Obsah

Výrok

– tvrzení, kterému lze jednoznačně přiřadit pravdivostní hodnotu. Při zápisu označujeme obvykle velkým písmenem A, B, ...X, Y, Z.

Pravdivostní hodnota

- F/T, FALSE/TRUE, Pravda/Nepravda
- 0/1, L/H, Low/High

Příklad

- Auto je dopravní prostředek. (TRUE) - je výrok
- Meloun není jedlý. (FALSE) - je výrok
- Jaké je počasí? (???) - není výrok

George Boole

Matematik zavedl v polovině 19. století zvláštní druh algebry, který našel uplatnění až koncem třicátých let 20. století při studiu spínacích obvodů.

Booleovu algebru budeme chápat jako nauku o operacích na množině B, která obsahuje dvě logické konstanty 0 a 1 a dále logické proměnné, které označujeme např. $A, B, ...X, Y, Z$.
Booleova algebra je univerzální algebra typu: $\{\vee, \wedge, \neg, 0, 1\}$ ($\{+, \cdot, \bar{}, 0, 1\}$; {OR, AND, NOT, 0, 1}).

Základní logické funkce

- negace
- logický součet
- logický součin

Některé možné způsoby vyjádření logické funkce

- Slovně

Některé možné způsoby vyjádření logické funkce

- Slovně
- Pravdivostní tabulkou

Některé možné způsoby vyjádření logické funkce

- Slovně
- Pravdivostní tabulkou
- Algebraickým výrazem

Některé možné způsoby vyjádření logické funkce

- Slovně
- Pravdivostní tabulkou
- Algebraickým výrazem
- Množinou indexů stavu

Některé možné způsoby vyjádření logické funkce

- Slovně
- Pravdivostní tabulkou
- Algebraickým výrazem
- Množinou indexů stavu
- Mapou

Slovní zadání logické funkce

Generátor liché parity

Vytvořte obvod, který bude generovat paritní bit a doplní počet jedniček ve slově na liché číslo.

Slovní zadání logické funkce

Generátor liché parity

Vytvořte obvod, který bude generovat paritní bit a doplní počet jedniček ve slově na liché číslo.

Bezpečnostní obvod

Pokud je zmáčknuto více jak jedno tlačítko spusť poplach.

Slovní zadání logické funkce

Generátor liché parity

Vytvořte obvod, který bude generovat paritní bit a doplní počet jedniček ve slově na liché číslo.

Bezpečnostní obvod

Pokud je zmáčknuto více jak jedno tlačítko spusť poplach.

Sčítačka

Navrhni bitovou sčítačku.

Struktura pravdivostní tabulky

index stavu | vstupní proměnné | výstupní hodnota

i	C	B	A	Y
0	0	0	0	
1	0	0	1	
2	0	1	0	
3	0	1	1	
4	1	0	0	
5	1	0	1	
6	1	1	0	
7	1	1	1	

Index stavu

Definice

Index stavu je dekadické vyjádření „binární“ kombinace vstupních hodnot.

Vyjádření logické funkce jako množinu indexů stavu

Je to množina, do které zahrneme ty indexy, pro které funkce nabývá hodnoty **1**.

Algebraický výraz

Definice

Je to zápis vytvořený pomocí proměnných, konstant, základních logických funkcí (AND, OR, NOT) a závorek. Např.:

$$A = (C + D) \cdot \overline{B}$$

Negace - inverze

Definice

Logická negace proměnné nabývá hodnoty **1** pokud proměnná nabývá hodnoty **0** a nabývá hodnoty **0** pokud proměnná nabývá hodnoty **1**. Vrací opačnou hodnotu.

Pravdivostní tabulka

A	Y
0	1
1	0

Algebraický výraz: $Y = \overline{A}$

Logický součet - disjunkce

Logický součet proměnných nabývá hodnoty **1** pokud alespoň jedna z proměnná nabýva hodnoty **1**.

Logický součet

Pravdivostní tabulka:

B	A	Y
0	0	0
0	1	1
1	0	1
1	1	1

Algebraický výraz: $Y = A + B$

Logický součin - konjunkce

Logický součin proměnných nabývá hodnoty **1** pokud všechny proměnné nabývají hodnoty **1**.

Logický součin

Pravdivostní tabulka:

B	A	Y
0	0	0
0	1	0
1	0	0
1	1	1

Algebraický výraz: $Y = A \cdot B$

Věta o reprezentaci logických funkcí

Jakoukoliv logickou funkci libovolného počtu proměnných lze vyjádřit pomocí logických funkcí dvou proměnných.

$$f(x, y, z) = x \cdot f(\mathbf{1}, y, z) + \bar{x} \cdot f(\mathbf{0}, y, z)$$

Zákony Booleovy algebry

Zákon asociativní

$$A + B + C = A + (B + C) = (A + B) + C = (A + C) + B$$

$$A \cdot B \cdot C = A \cdot (B \cdot C) = (A \cdot B) \cdot C = (A \cdot C) \cdot B$$

Zákon komutativní

$$A + B = B + A$$

$$A \cdot B = B \cdot A$$

Zákon distributivní

$$A \cdot (B + C) = (A \cdot B) + (A \cdot C)$$

$$A + (B \cdot C) = (A + B) \cdot (A + C)$$

Zákony Booleovy algebry

Zákon dvojité negace

$$\overline{\overline{A}} = A$$

Zákon vyloučeného třetího

$$A + \overline{A} = 1$$

$$A \cdot \overline{A} = 0$$

Zákony Booleovy algebry

Zákon agresivnosti

$$A + 1 = 1$$

$$A \cdot 0 = 0$$

Zákon neutrálnosti

$$A + 0 = A$$

$$A \cdot 1 = A$$

Zákon absorbce

$$A + A = A$$

$$A \cdot A = A$$

také

$$A + A \cdot B = A$$

$$A \cdot (A + B) = A$$

$$(A + B) \cdot (A + \overline{B}) = A$$

$$A \cdot B + A \cdot \overline{B} = A$$

Zákony Booleovy algebry

Zákon absorbce negace

$$A + \overline{A} \cdot B = A + B$$

$$A \cdot (\overline{A} + B) = A \cdot B$$

Zákon o vytvoření negace - De Morganovy zákony

$$\overline{(A + B)} = \overline{A} \cdot \overline{B}$$

$$\overline{(A \cdot B)} = \overline{A} + \overline{B}$$

Zákon absorbce negace

$$A + \overline{A} \cdot B = A + B$$

$$A \cdot (\overline{A} + B) = A \cdot B$$

Zákon o vytvoření negace - De Morganovy zákony

$$\overline{(A + B)} = \overline{A} \cdot \overline{B}$$

$$\overline{(A \cdot B)} = \overline{A} + \overline{B}$$

Zákony Booleovy algebry - shrnutí

Axiomy a odvozené vlastnosti

- idempotence: $A + A = A, A \cdot A = A$
- komutativita: $A + B = B + A, A \cdot B = B \cdot A$
- asociativita: $A + (B + C) = (A + B) + C, A \cdot (B \cdot C) = (A \cdot B) \cdot C$
- absorpce: $A \cdot (A + B) = A, A + (A \cdot B) = A$
- distributivita: $A \cdot (B + C) = (A \cdot B) + (B \cdot C), A + (B \cdot C) = (A + B) \cdot (A + C)$
- neutralita 0 a 1: $A + 0 = A, A \cdot 1 = 1$
- agresivita 0 a 1: $A + 1 = 1, A \cdot 0 = 0$
- komplementarita: $\overline{A} + A = 1, \overline{A} \cdot A = 0$
- absorpce negace: $A \cdot (\overline{A} + B) = A \cdot B, A + (\overline{A} \cdot B) = A + B$
- DeMorganovy zákony: $\overline{A + B} = \overline{A} \cdot \overline{B}, \overline{A \cdot B} = \overline{A} + \overline{B}$
- dvojitá negace: $\overline{\overline{A}} = A$

Co je logická funkce

Je funkce, která přiřazuje výstupní logickou hodnotu kombinaci vstupních logických proměnných:

$$Y = f(A, B, C, D, ...)$$

Počet logických funkcí

Pro n vstupních logických proměnných lze definovat 2^{2^n} logických funkcí.

Lichá parita - pravdivostní tabulka

i	C	B	A	Y
0	0	0	0	1
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	0

Logická funkce - příklad

Lichá parita - množina indexů stavu

i	C	B	A	Y
0	0	0	0	1
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	0

$$Y = \{0; 3; 5; 6\}$$

Libovolnou funkci $f(A, B, C, \ldots)$ lze zapsat ve dvou základních tvarech.

Základní součtový tvar (též úplná normální disjunktivní forma)

Funkci zapsanou v základním součtovém tvaru dostaneme jako součet základních součinů přímých nebo negovaných proměnných.
Zapisujeme pouze základní součiny (mintermy) u těch kombinací přímých nebo negovaných proměnných, u kterých má funkce hodnotu **1**; v mintermu zapisujeme proměnné, které nabývají v příslušné kombinaci hodnoty **1** jako přímé, a proměnné, které nabývají hodnoty **0** jako negované.

Například

$$f = A\overline{B}C + AB\overline{C}$$

Základní součinový tvar (též úplná normální konjunktivní forma)

Funkci zapsanou v základním součinovém tvaru dostaneme jako součin základních součtů přímých nebo negovaných proměnných. Zapisujeme pouze základní součty (maxtermy) u těch kombinací přímých nebo negovaných proměnných, u kterých má funkce hodnotu **0**; v maxtermu zapisujeme proměnné, které nabývají v příslušné kombinaci hodnoty **0** jako přímé, a proměnné, které nabývají hodnoty **1** jako negované.

Například

$$f = (A + \overline{B} + C)(A + B + \overline{C})$$

Logická funkce - příklad

Lichá parita - algebraický výraz (disjunktivní forma)

C	B	A	Y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

$$Y = \bar{A}\bar{B}\bar{C} + AB\bar{C} + A\bar{B}C + \bar{A}BC$$

Tato forma se používá se nejčastěji.

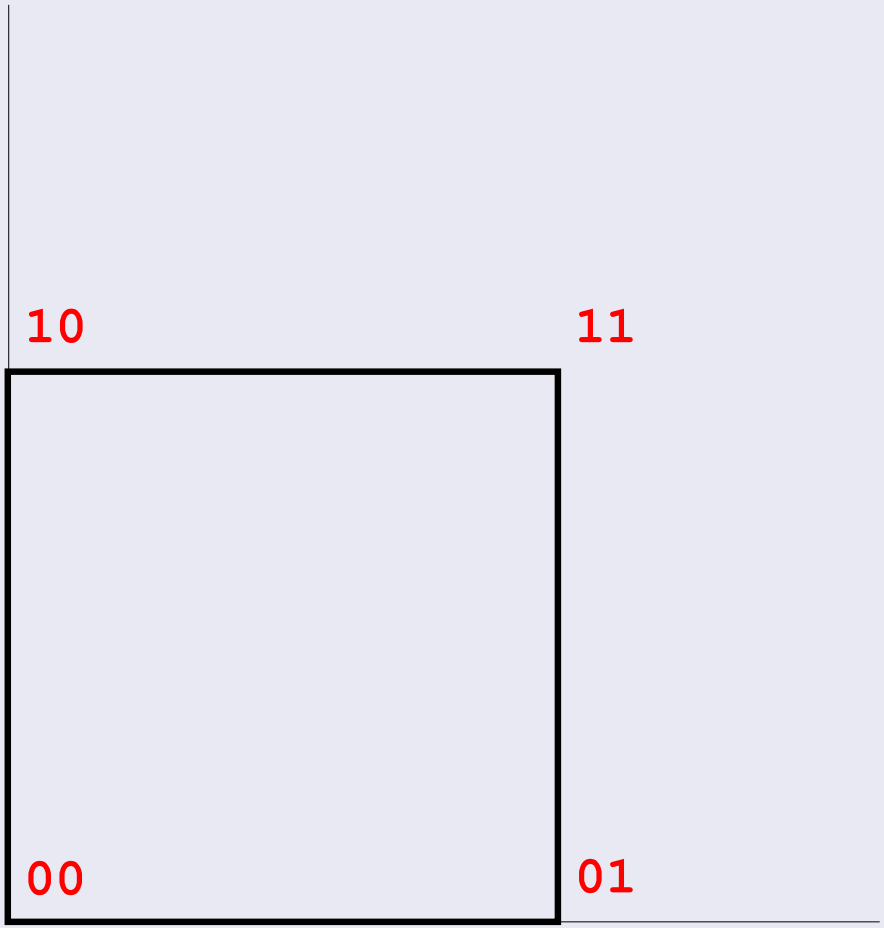
Logická funkce - příklad

Lichá parita - algebraický výraz (konjunktivní forma)

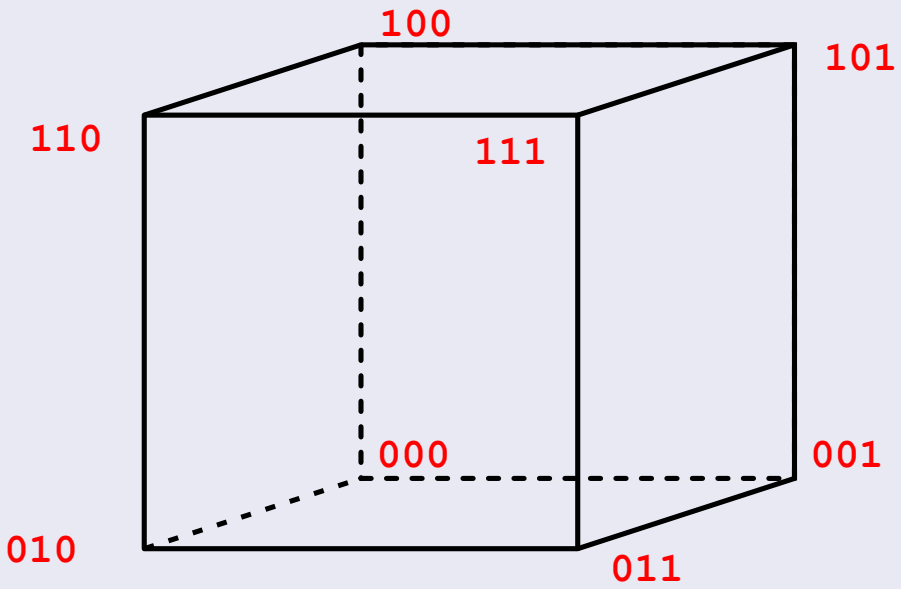
C	B	A	Y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

$$Y = (\bar{A}+B+C)(A+\bar{B}+C)(A+B+\bar{C})(\bar{A}+\bar{B}+\bar{C})$$

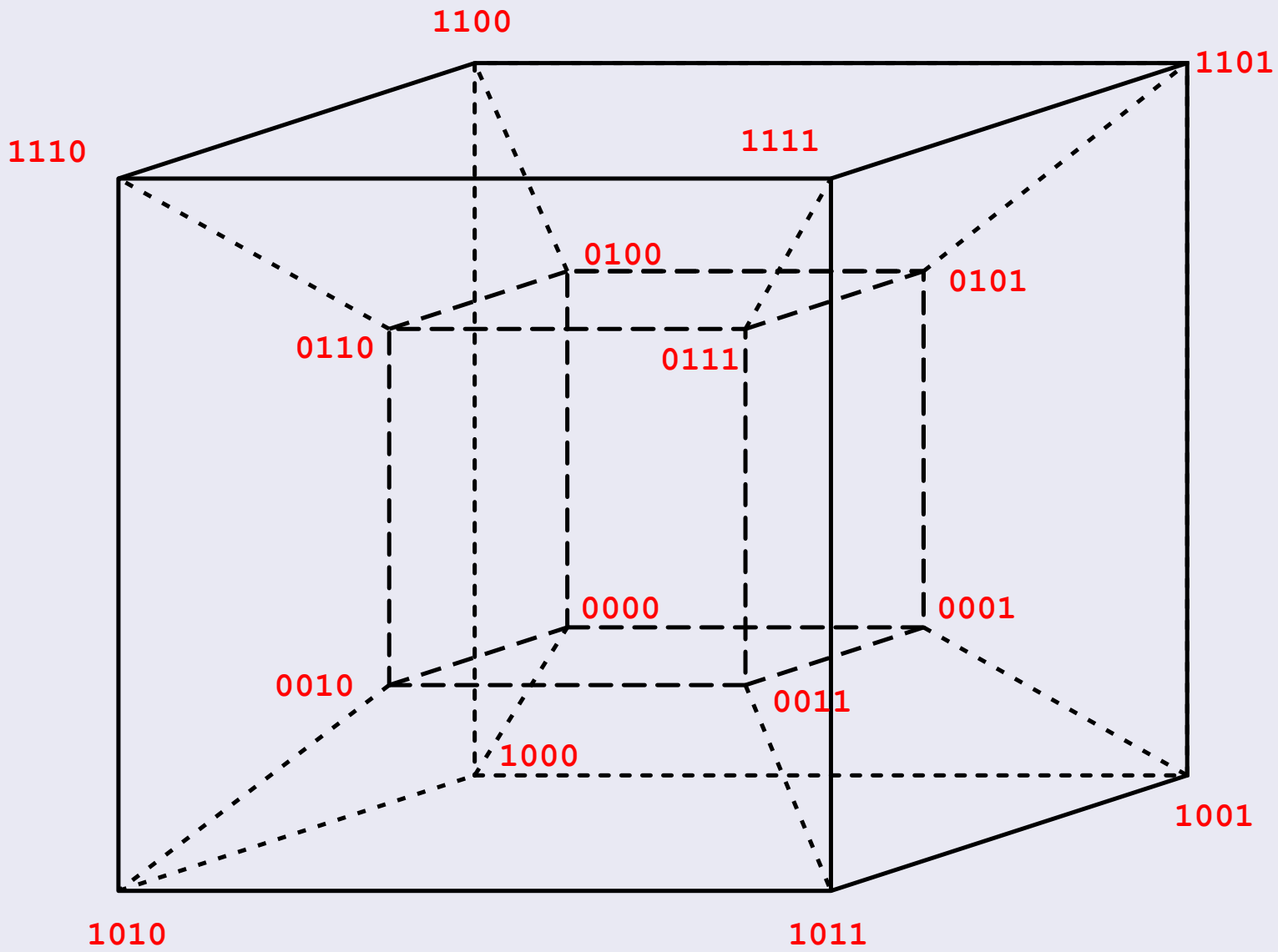
Kódová krychle 2D



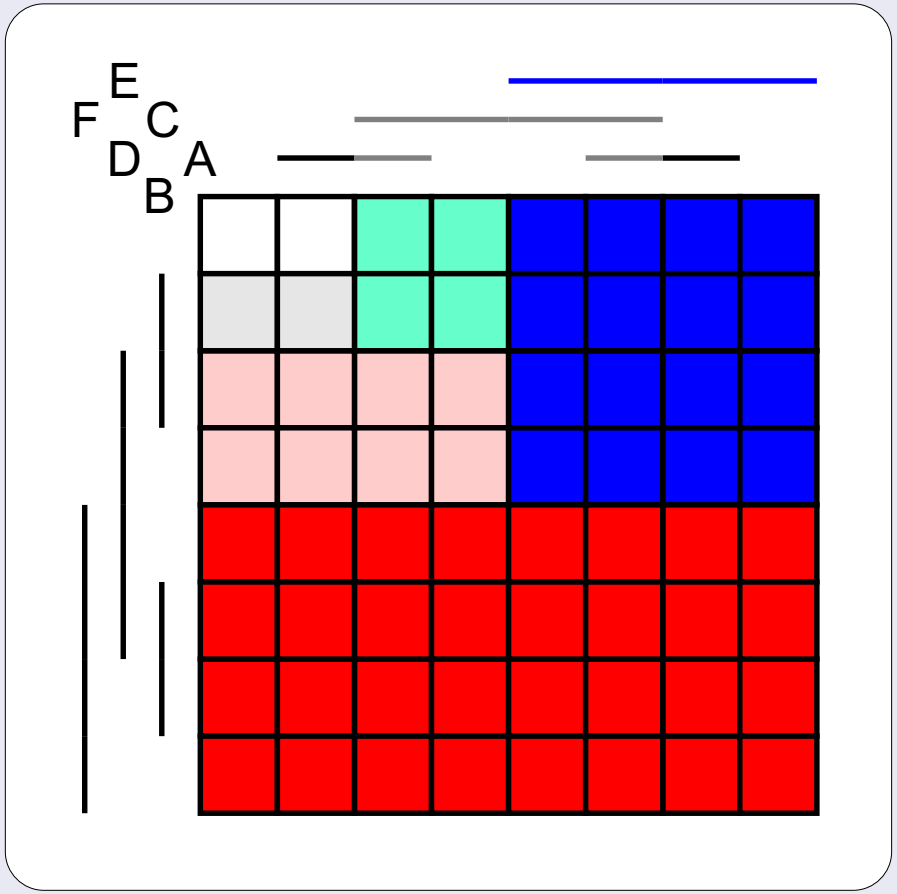
Kódová krychle 3D



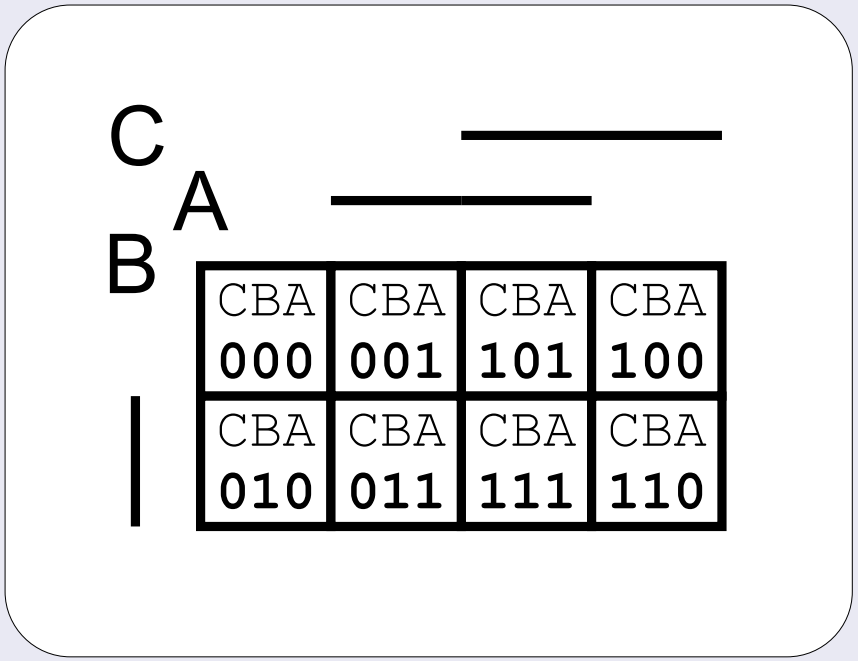
Kódová krychle 4D



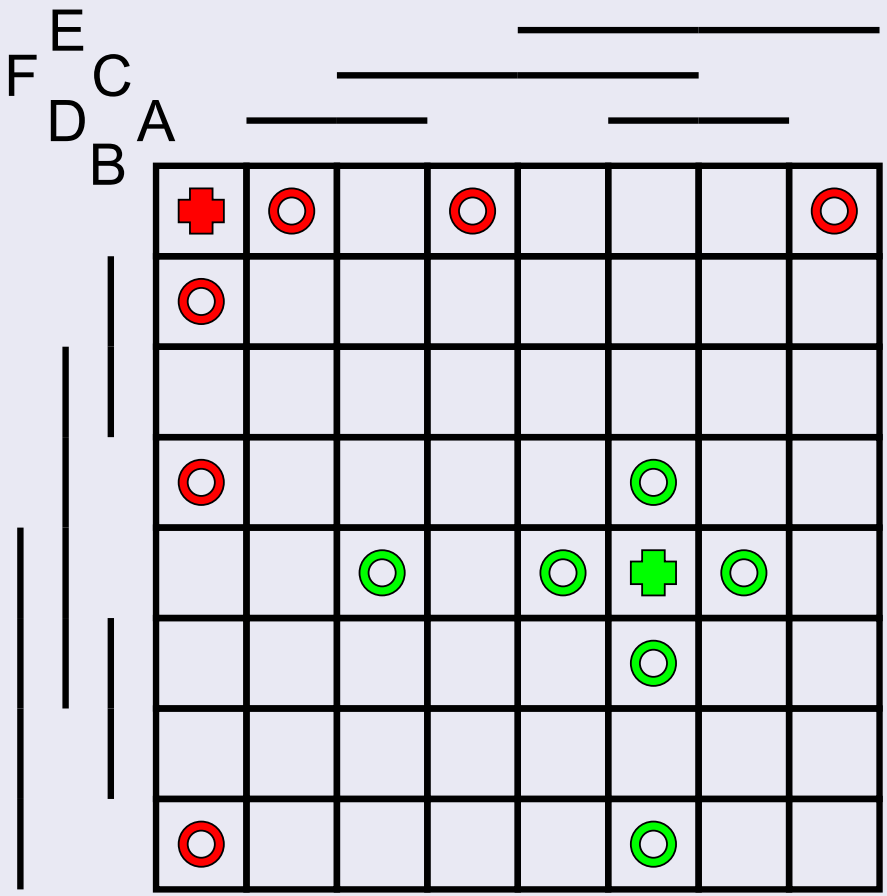
Vznik mapy



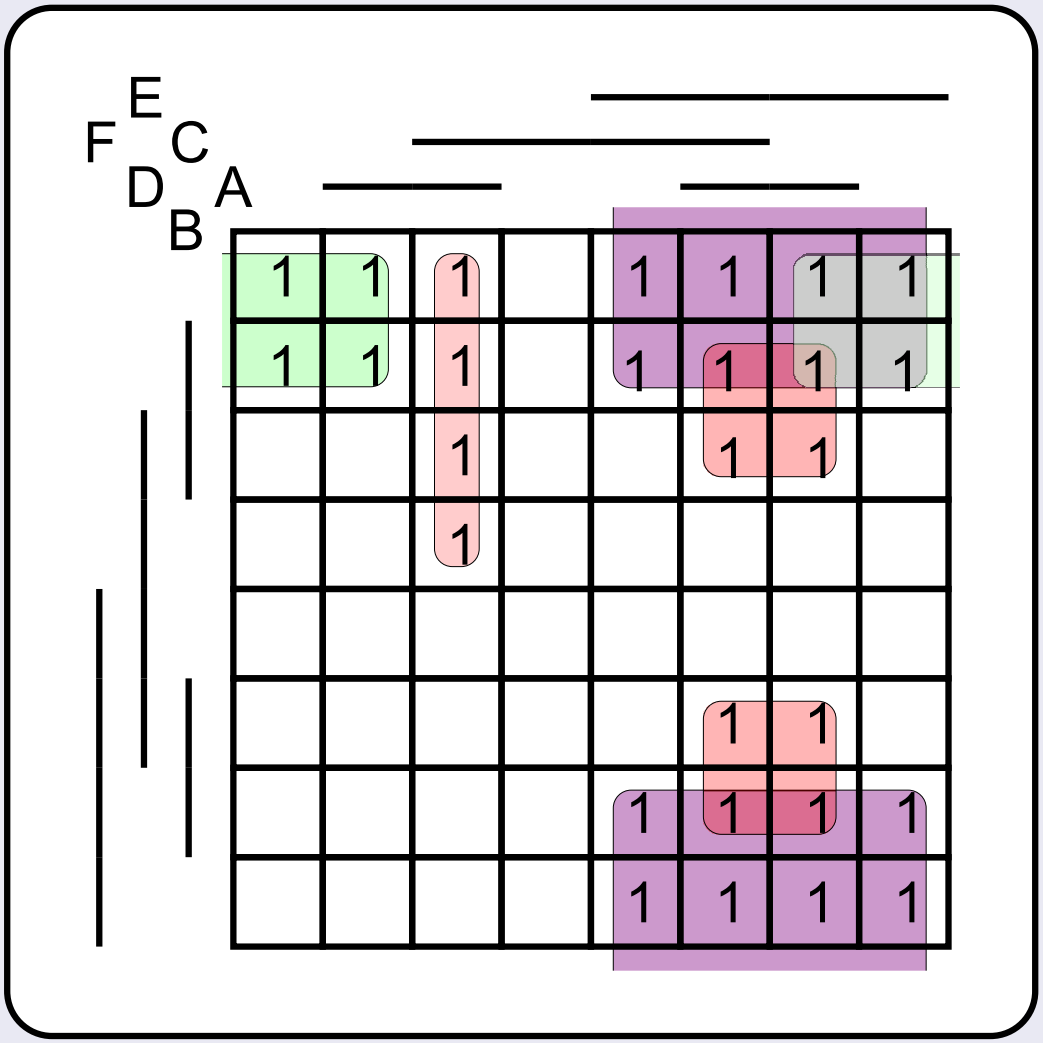
Průmět pruhů



Sousedi v mapě



Smyčky



$$f = \bar{C}\bar{D}\bar{F} +$$

$$AC\bar{E}\bar{F} +$$

$$ABE +$$

$$E\bar{D}$$

Karnaughova mapa

Pravidla pro smyček

- smyčky se tvoří pro **1**
- na konci každá **1** musí být alespoň v jedné smyčce
- zahrnuje pouze **1** nebo neurčité stavy
- velikost smyčky 2^n to jest 1, 2, 4, 8, 16 ....
- zahrnuje pouze nejbližší sousedy

Popis smyček

Do popisu zahrneme pouze takové proměnné, které na všech polích smyčky mají stejnou hodnotu. Ty proměnné, které nabývají hodnoty **1** píšeme do součinu přímo a ty které **0** píšeme s negací.

Popis smyček

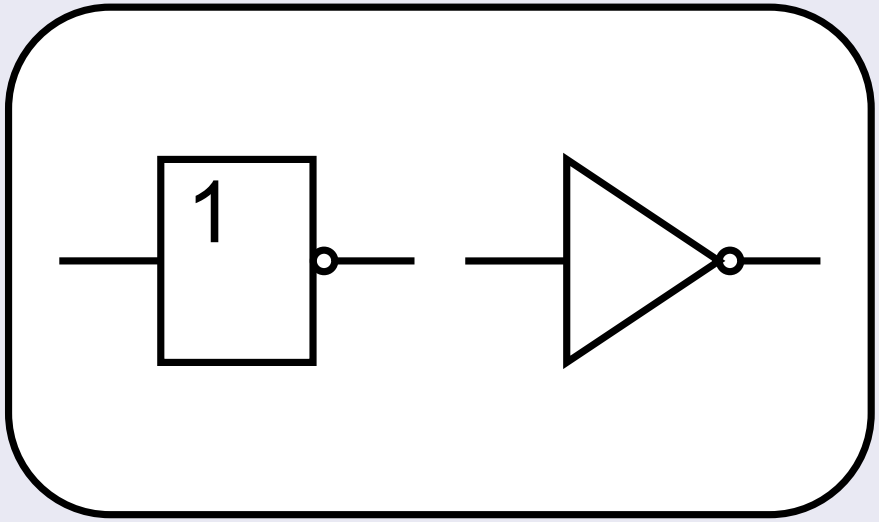
$$f = \bar{C}\bar{D}\bar{F} + AC\bar{E}\bar{F} + ABE + E\bar{D}$$

Logické funkce pro dvě proměnné

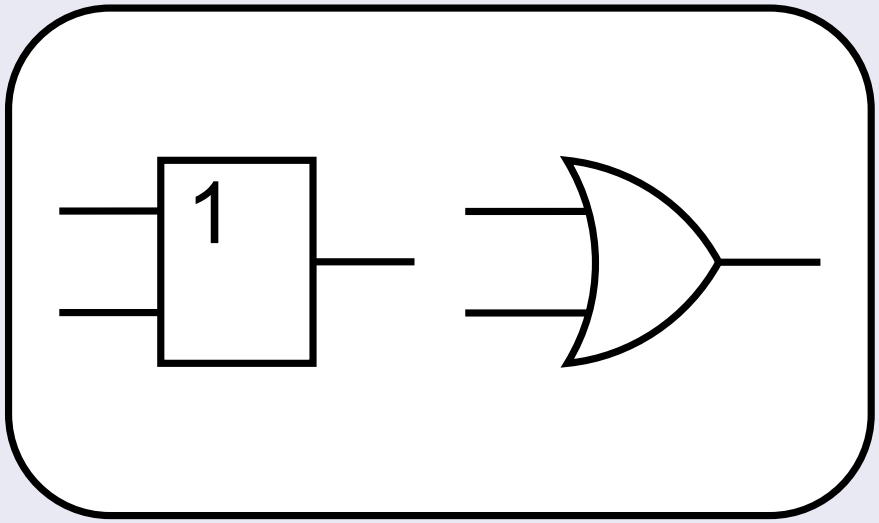
A	0	1	0	1	
B	0	0	1	1	
f_0	0	0	0	0	Nulová funkce $f_0 = 0$
f_1	1	0	0	0	NOR - Pierceova funkce $f_1 = \overline{(A + B)}$
f_2	0	1	0	0	Zpětná inhibice $f_2 = \overline{A}B$
f_3	1	1	0	0	Negace B $f_3 = \overline{B}$
f_4	0	0	1	0	Přímá inhibice $f_4 = \overline{B}A$
f_5	1	0	1	0	Negace A $f_5 = \overline{A}$
f_6	0	1	1	0	Nonekvivalence $f_6 = \overline{A}B + A\overline{B}$
f_7	1	1	1	0	NAND - Shafferova funkce $f_7 = \overline{AB}$
f_8	0	0	0	1	AND $f_8 = AB$
f_9	1	0	0	1	Ekvivalence $f_9 = \overline{AB} + AB$
f_{10}	0	1	0	1	Opakování A $f_{10} = A$
f_{11}	1	1	0	1	Přímá implikace $f_{11} = A + \overline{B}$
f_{12}	0	0	1	1	Opakování B $f_{12} = B$
f_{13}	1	0	1	1	Zpětná implikace $f_{13} = \overline{A} + B$
f_{14}	0	1	1	1	OR $f_{14} = A + B$
f_{15}	1	1	1	1	Jedničková funkce

Schématické značky logických funkcí

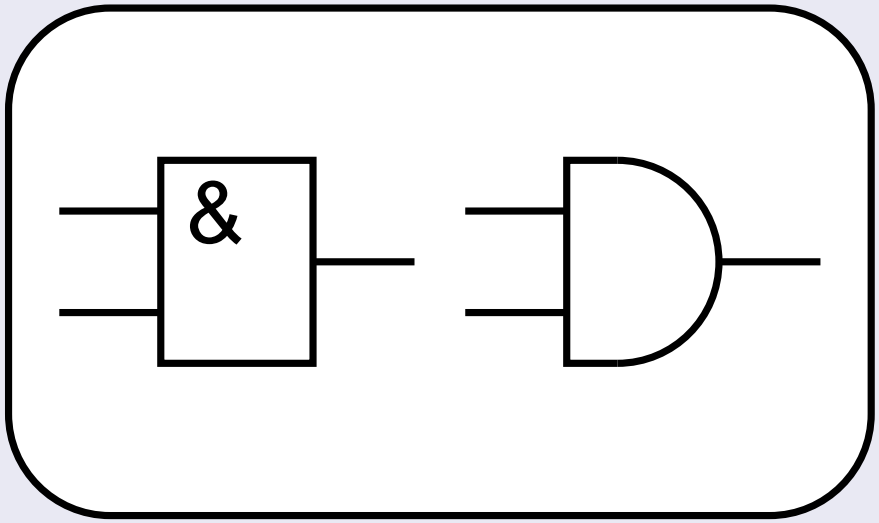
Negace: NOT



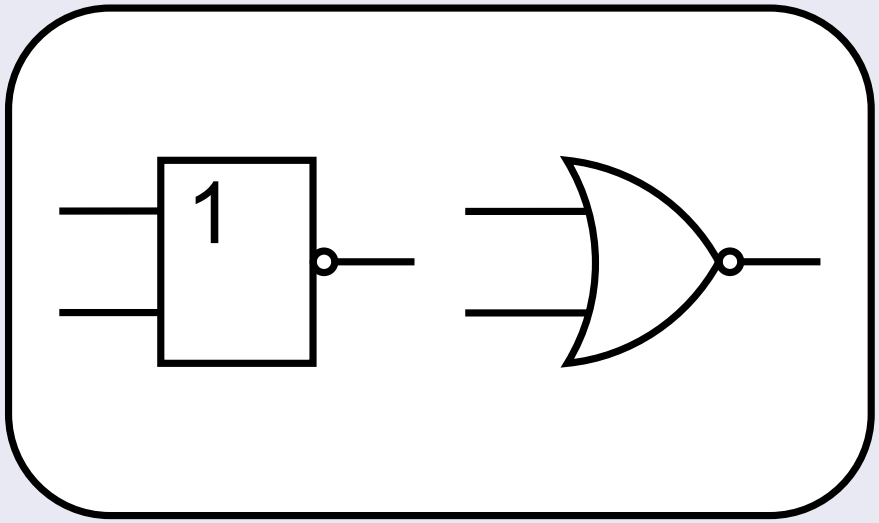
Logický součet: OR



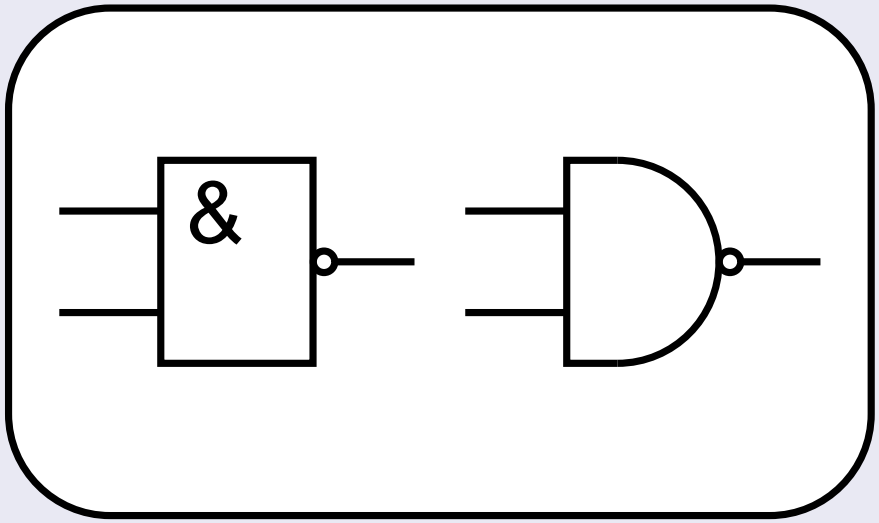
Logický součin: AND



Negovaný logický součet - Peirceova funkce: NOR

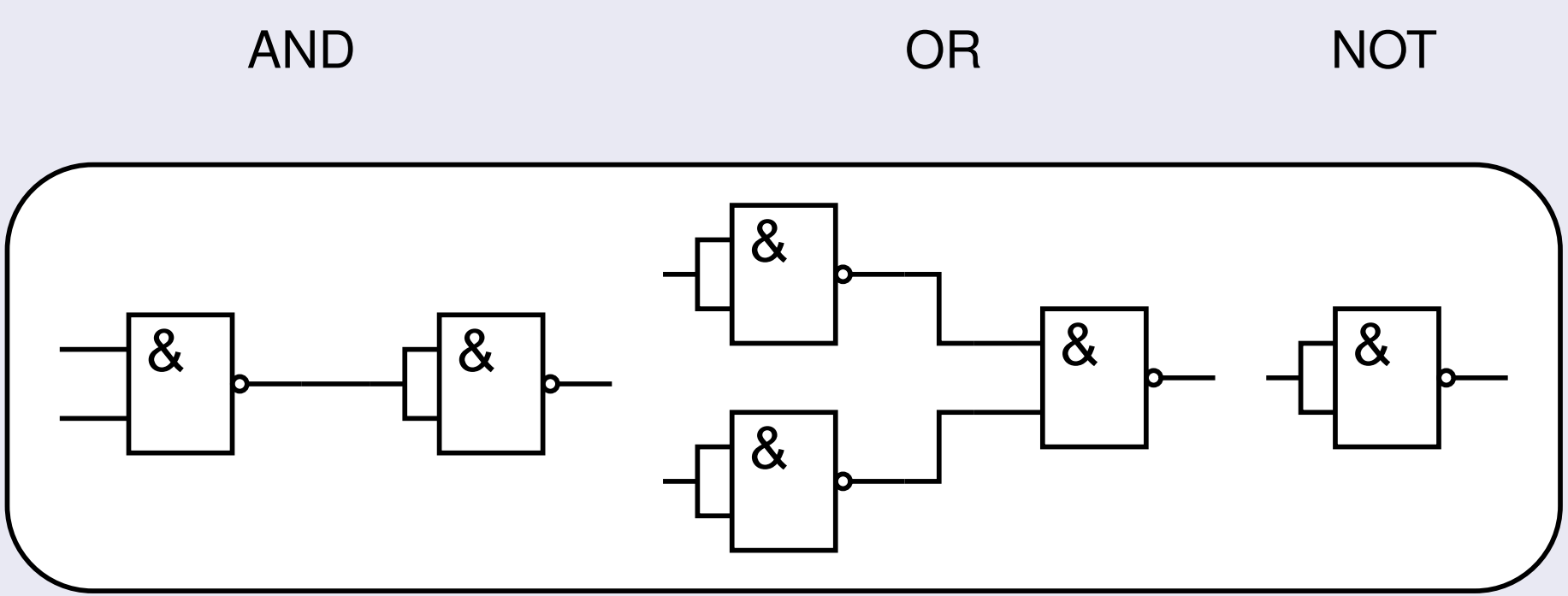


Negovaný logický součin - Shefferova funkce: NAND



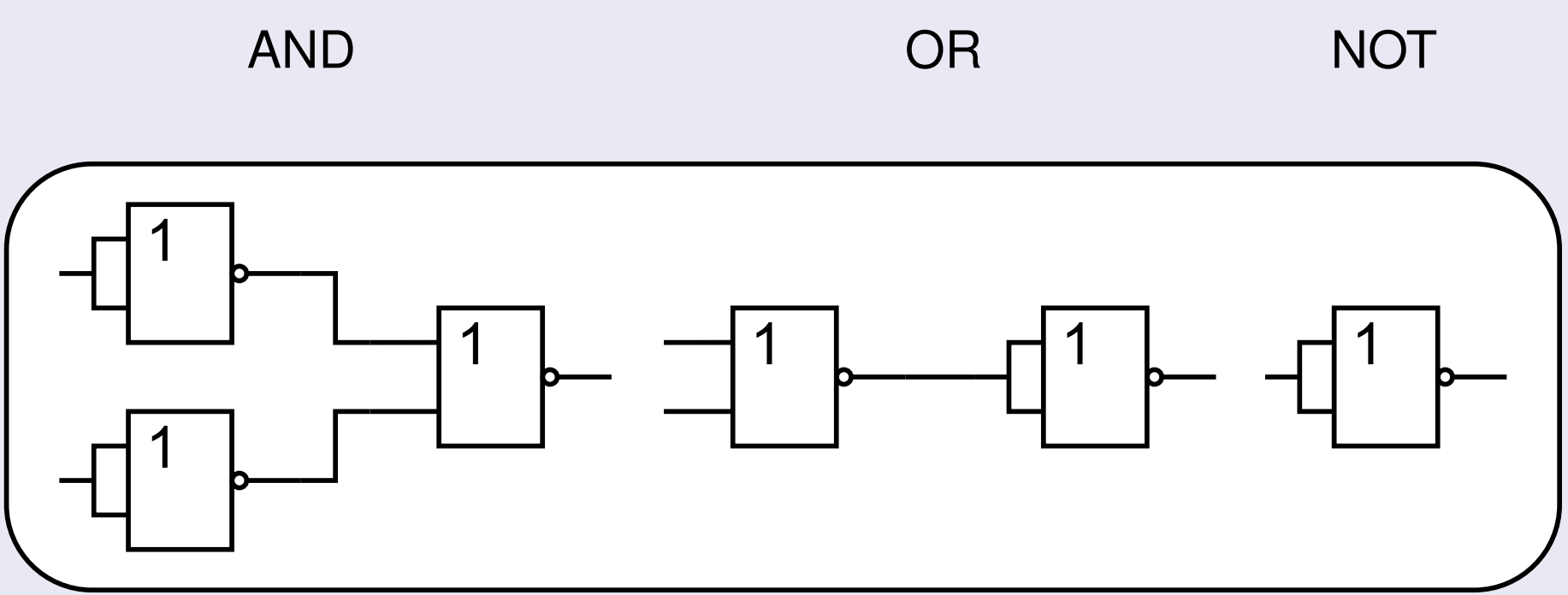
Realizace základních logivkých funkcí pomocí hradel NAND

Shefferova funkce: NAND



Realizace základních logických funkcí pomocí hradel NOR

Piercova funkce: NOR

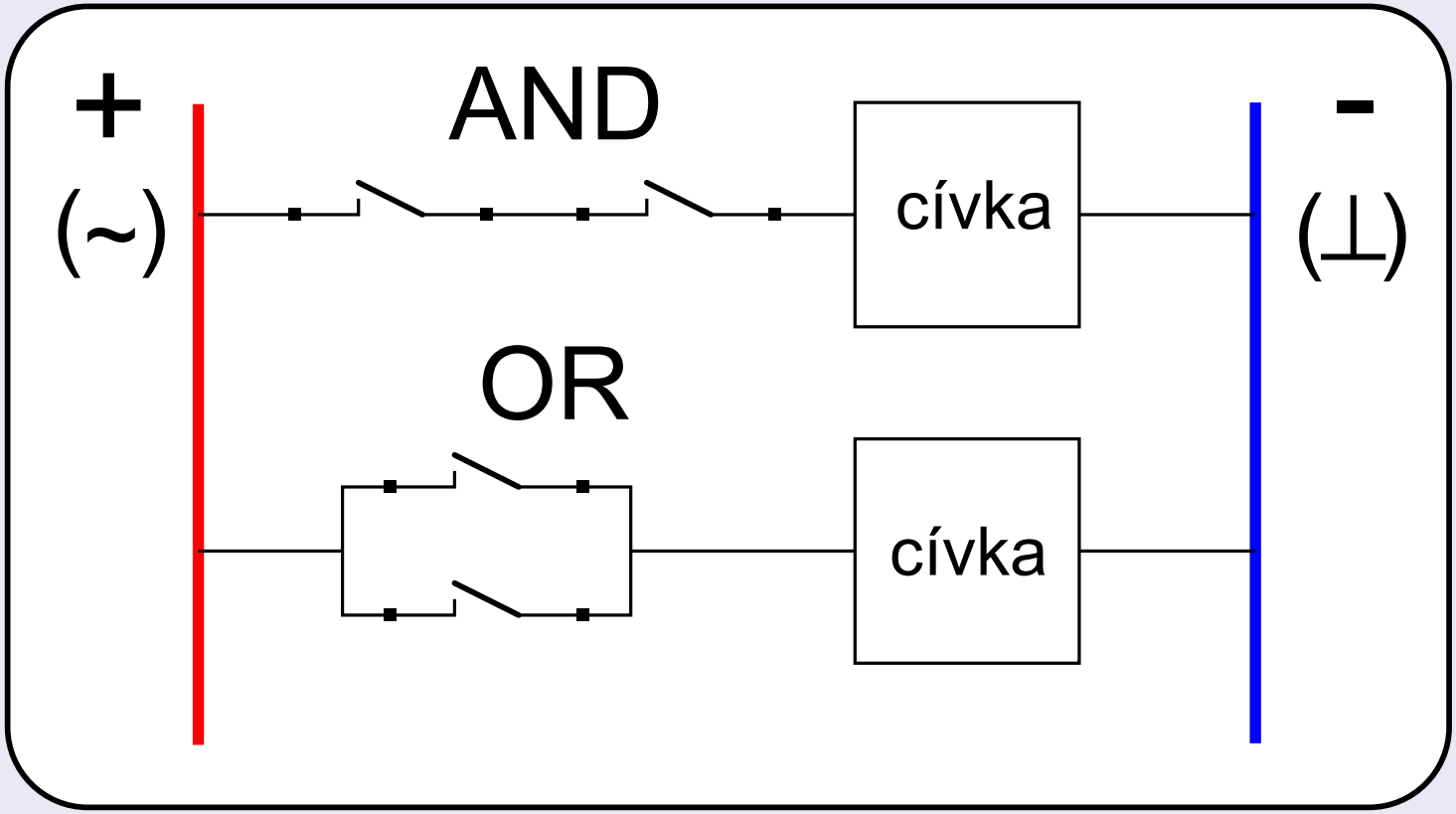


Realizace logické funkce

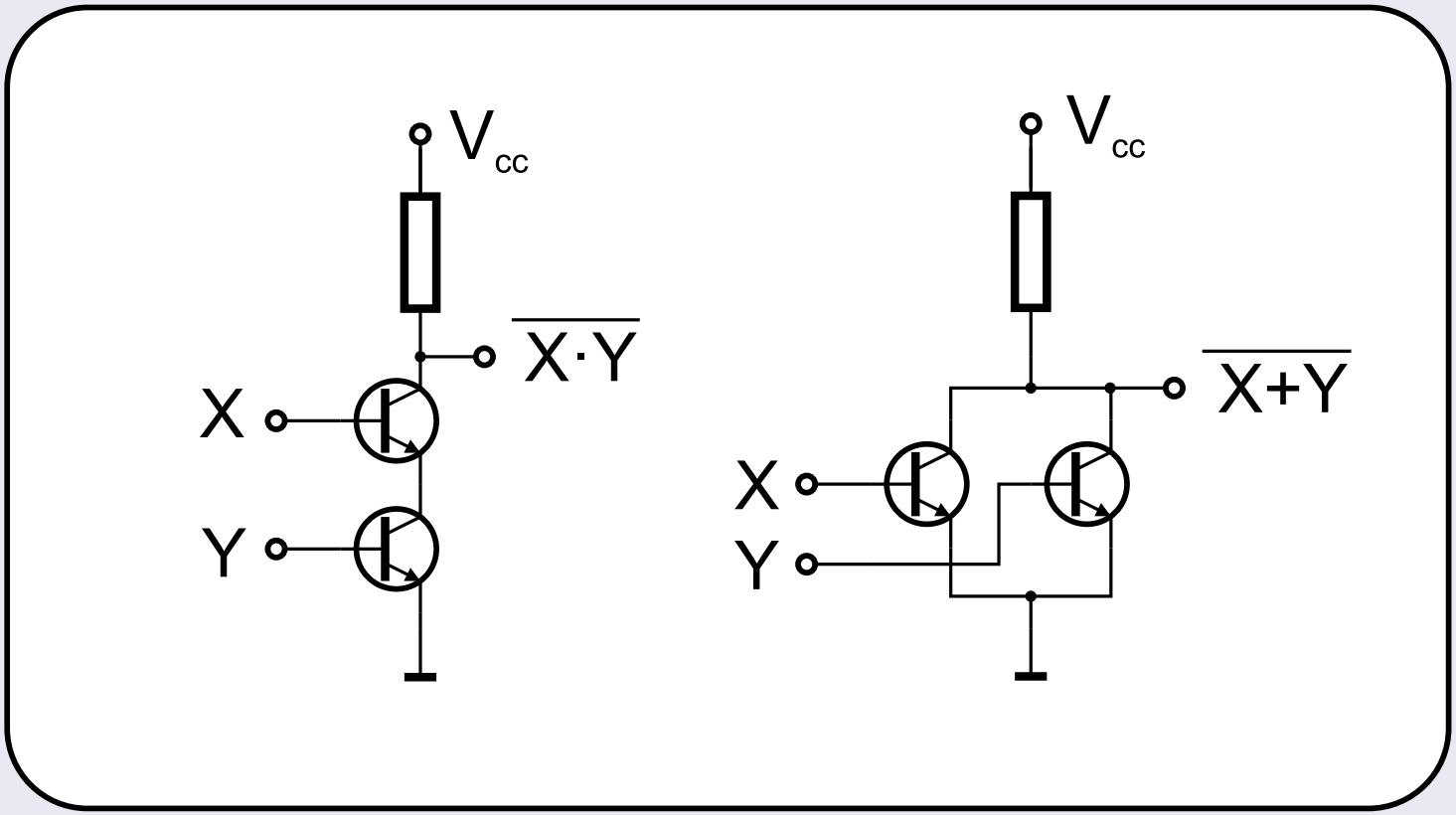
K realizaci logických funkcí se používají tzv. logické obvody. Ty mohou být například:

- releové
- pneumatické
- diodové
- tranzistorové (bipolární, CMOS)
- na bázi IO

Releová logika



Tranzistorová logika



Integrované obvody

Realizace log. operací pomocí integrovaných obvodů - logických členů, hradel.

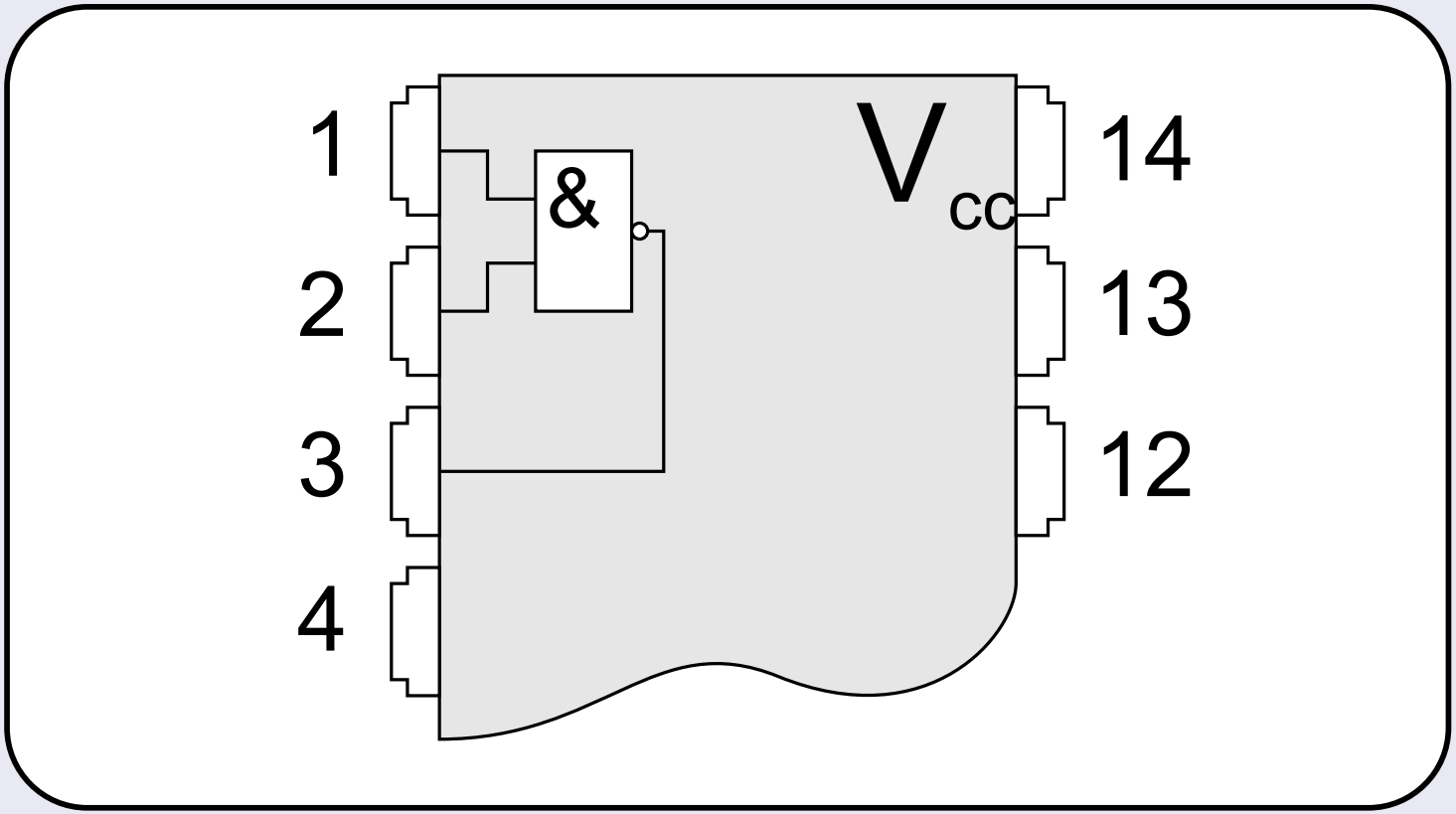
Vývody:

- Napájení
- Vstupy
- Výstupy

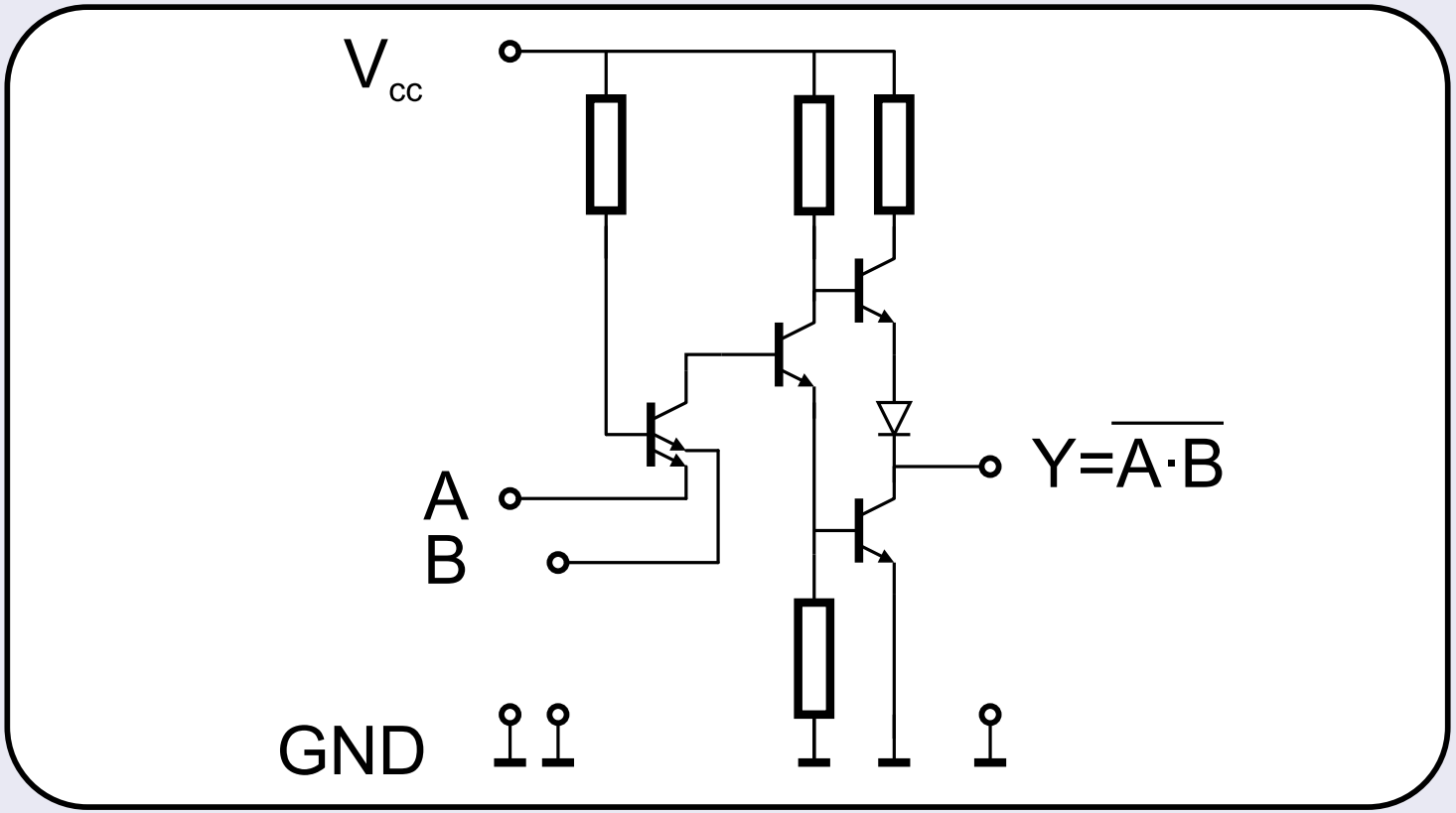
Různé typy:

- DTL
- TTL
- CMOS

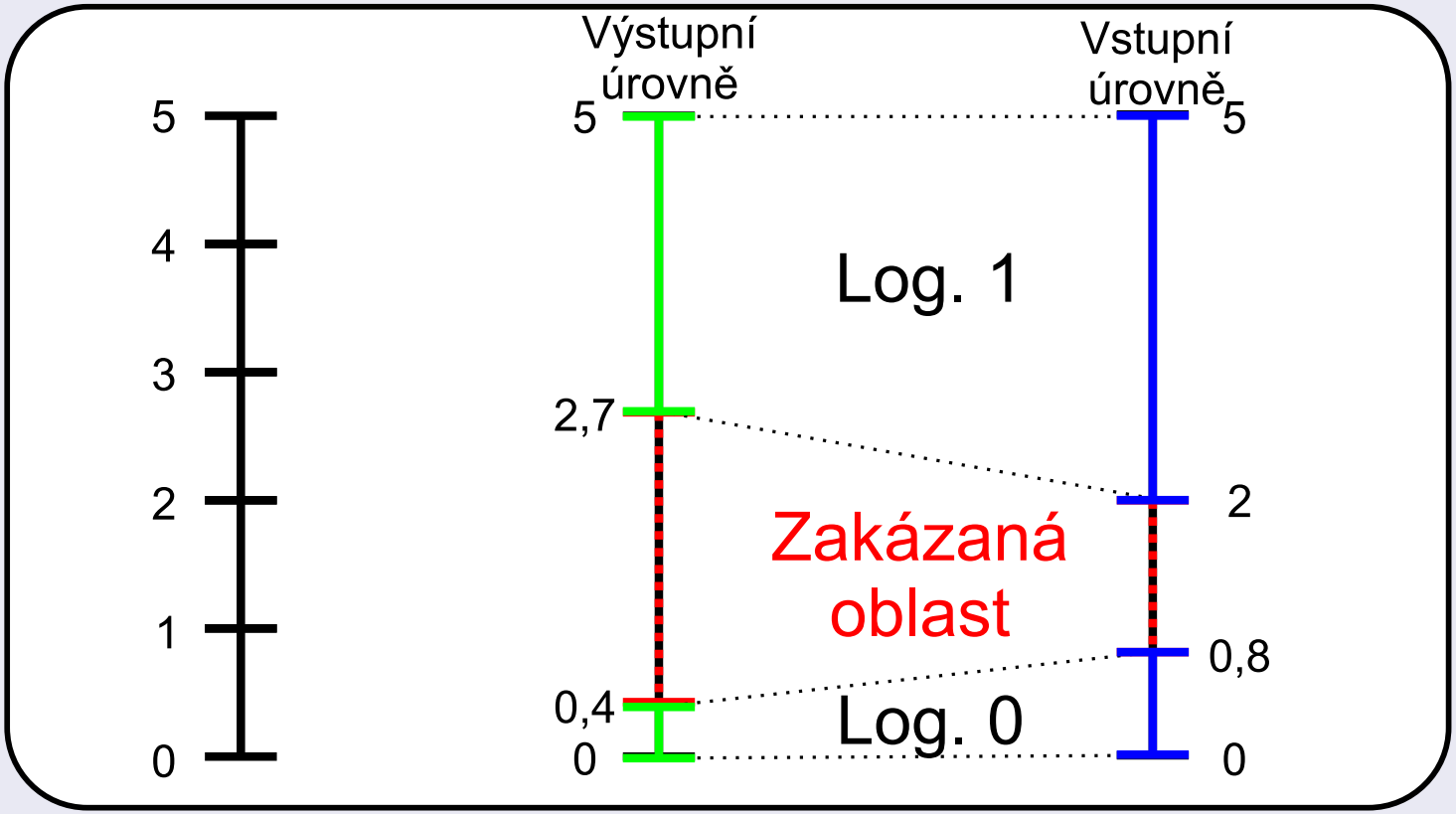
Integrované obvody - př. MH7400



TTL NAND



TTL napěťové logické úrovně



Důležité označení - napětí

Napájecí napětí

- V_{cc}
- V_{DD}
- GND - zem

Zem

- V_{SS}
- GND - zem

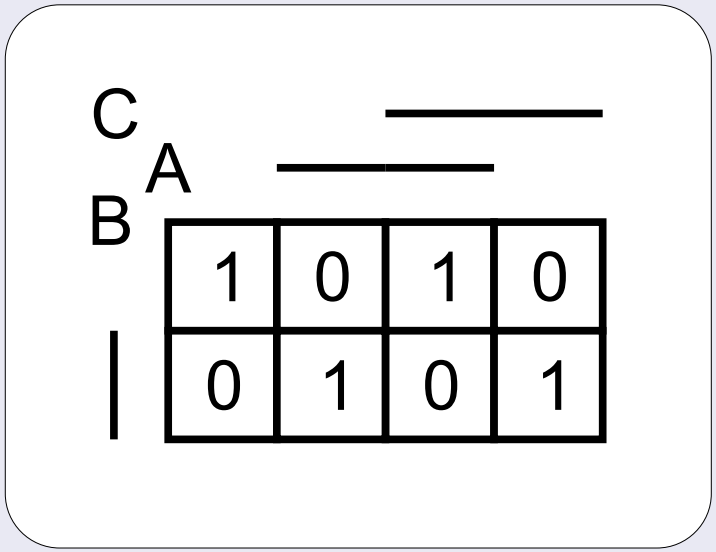
Důležité označení napětí

Logické úrovně

- $V_{OH_{min}}$ - (output hight) minimální napětí výstupní log 1
- $V_{OH_{max}}$ - (output hight) maximální napětí výstupní log 1
- $V_{IH_{min}}$ - (input hight) minimální napětí výstupní log 1
- $V_{IH_{max}}$ - (input hight) maximální napětí výstupní log 1
- $V_{OL_{min}}$ - (output low) minimální napětí výstupní log 0
- $V_{OL_{max}}$ - (output low) maximální napětí výstupní log 0
- $V_{IL_{min}}$ - (input low) minimální napětí výstupní log 0
- $V_{IL_{max}}$ - (input low) maximální napětí výstupní log 0
- V_T - (threshold) hraniční napětí

Lichá parita - Karnaughova mapa

C	B	A	Y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0



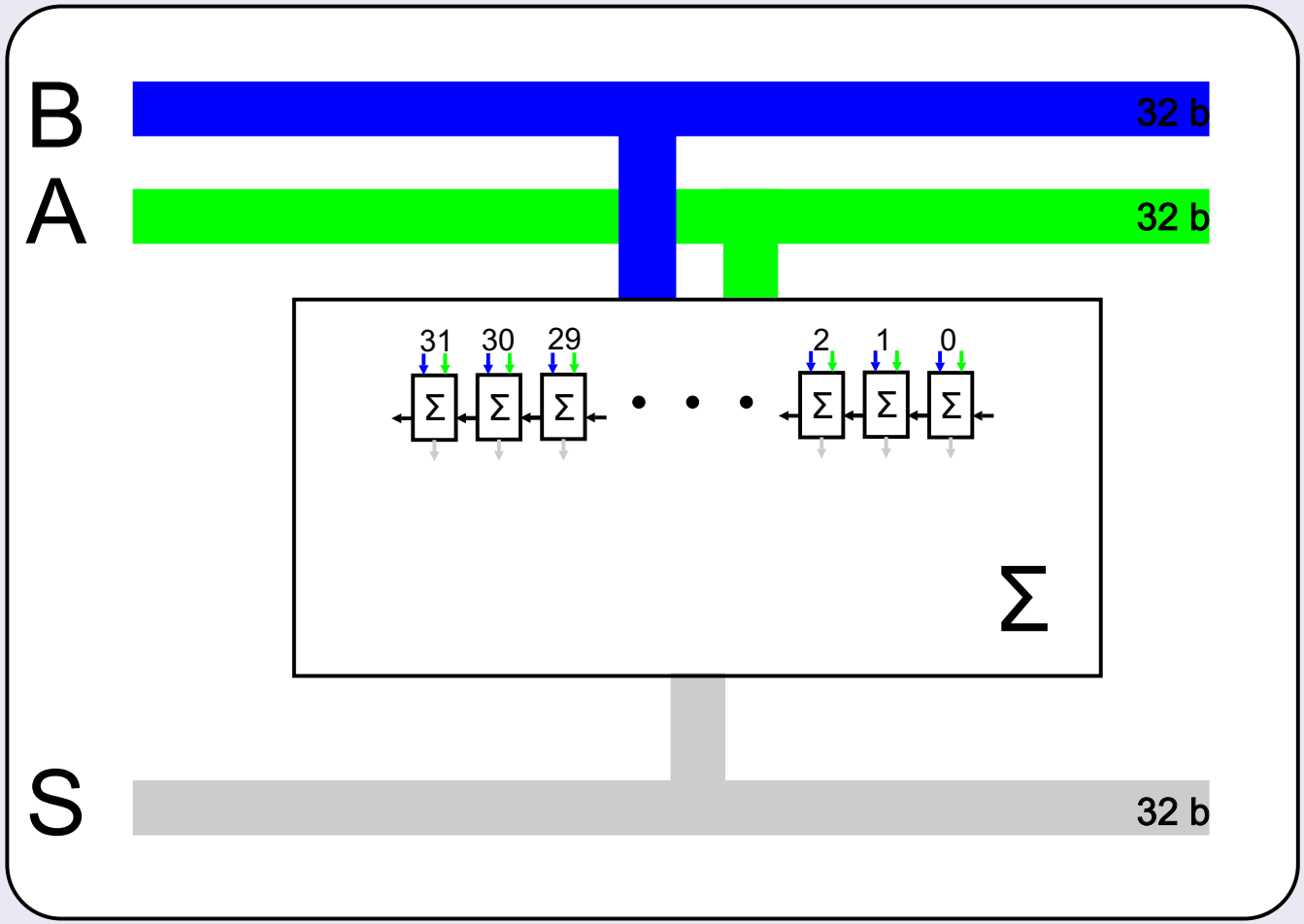
Logická funkce - příklad 2

Bitová sčítačka

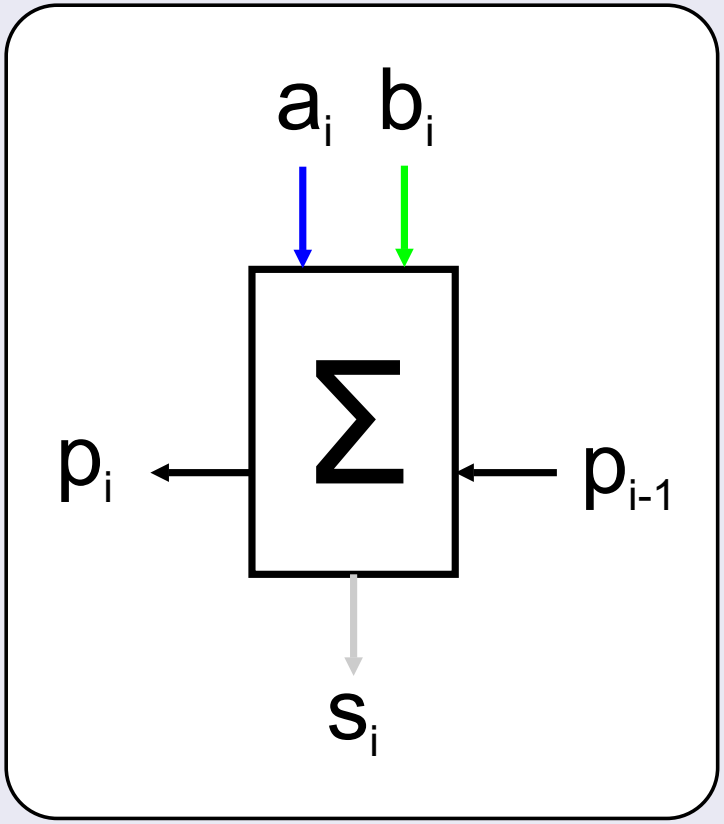
Navrhněte obvod, který bude realizovat bitovou sčítačku.

`  1 1   1`	
`1011011010`	`730`
`100011010`	`282`
`1111110100`	`1012`

Bitová sčítačka - rozbor

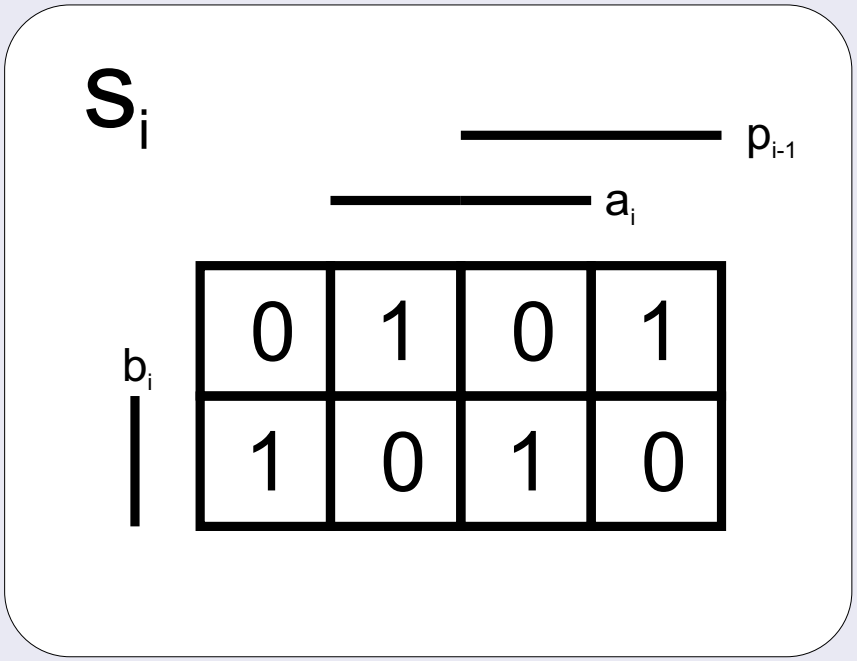


Bitová sčítačka - rozbor



Bitová sčítačka - Karnaughova mapa - součet

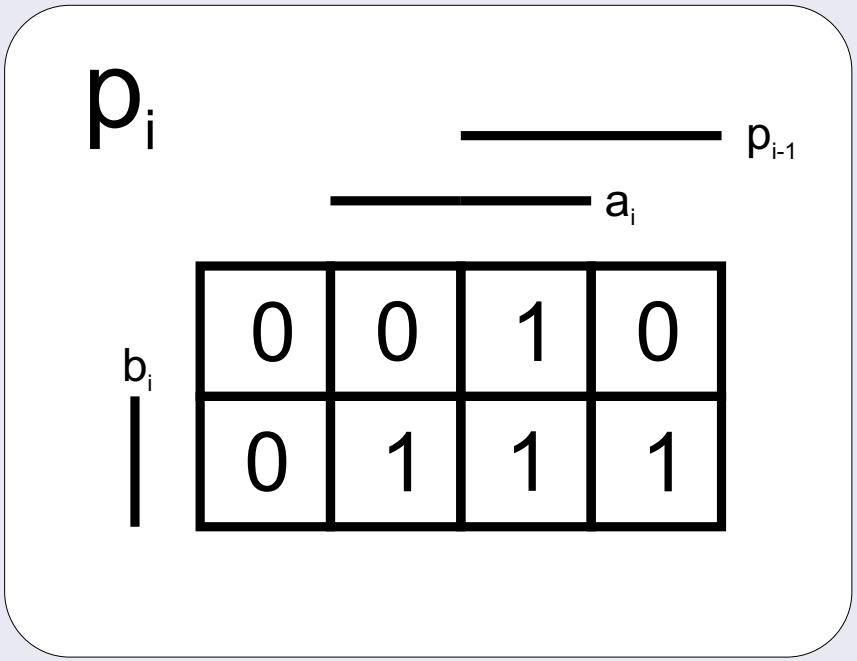
p_{i-1}	b_i	a_i	s_i
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1



$$s_i = a_i\overline{b_i}\,\overline{p_{i-1}} + \overline{a_i}b_i\overline{p_{i-1}} + \overline{a_i}\,\overline{b_i}p_{i-1} + a_ib_ip_{i-1}$$

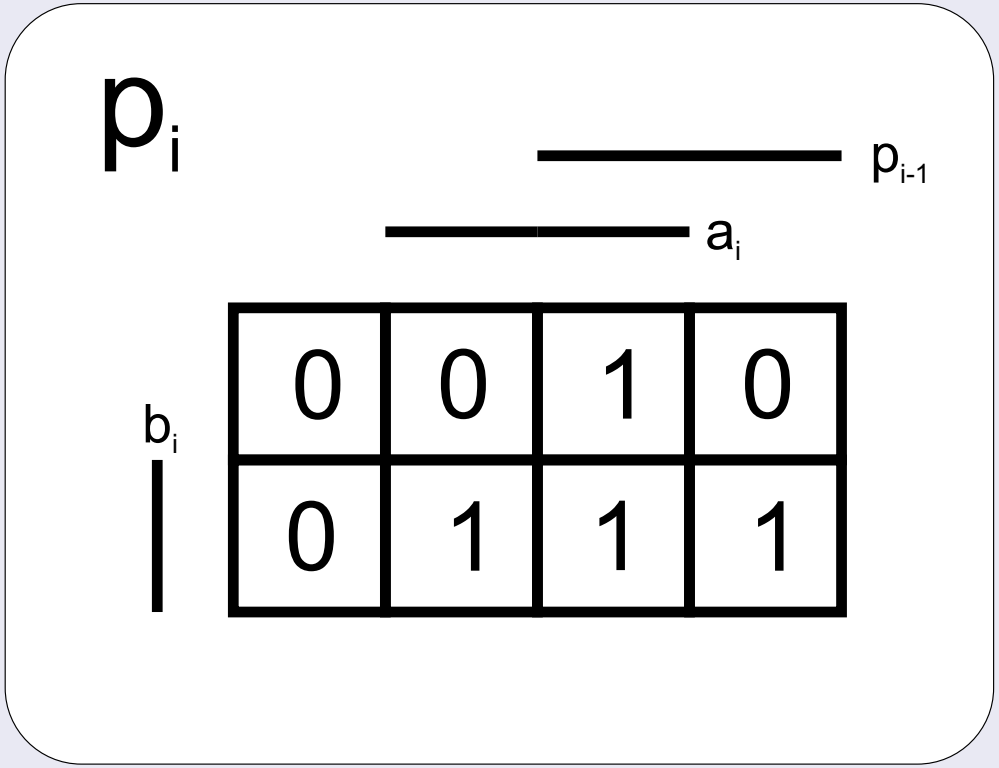
Bitová sčítačka - Karnaughova mapa - přenos do vyššího řádu

p_{i-1}	b_i	a_i	p_i
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

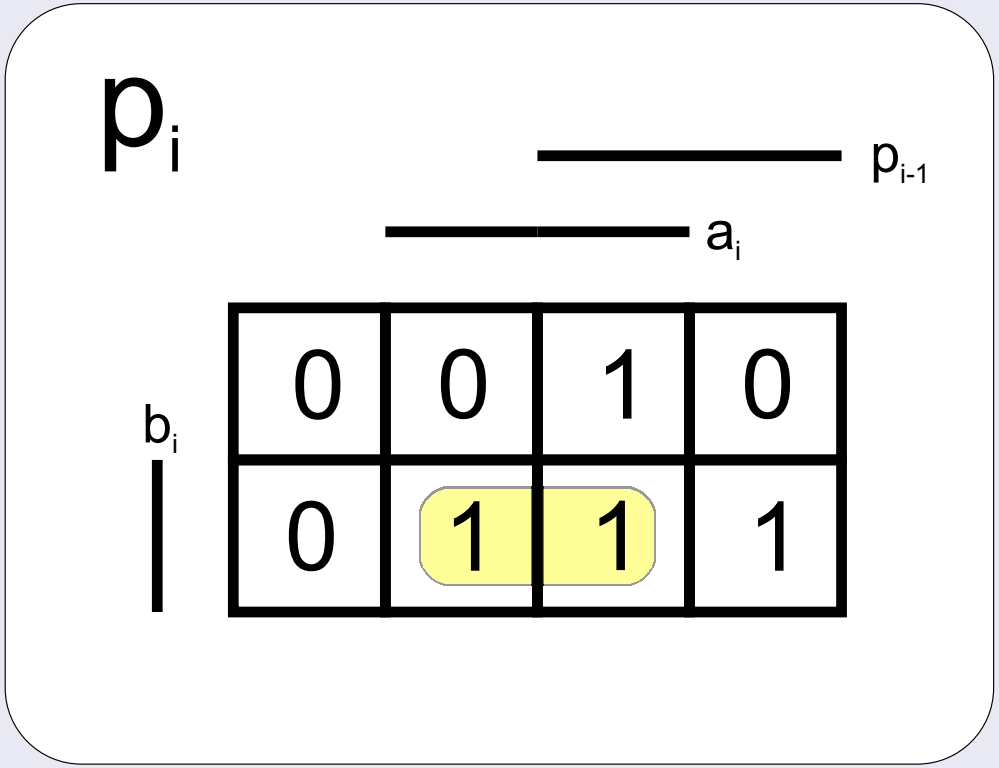


$$p_i = a_i b_i \overline{p_{i-1}} + a_i \overline{b_i} p_{i-1} + \overline{a_i} b_i p_{i-1} + a_i b_i p_{i-1}$$

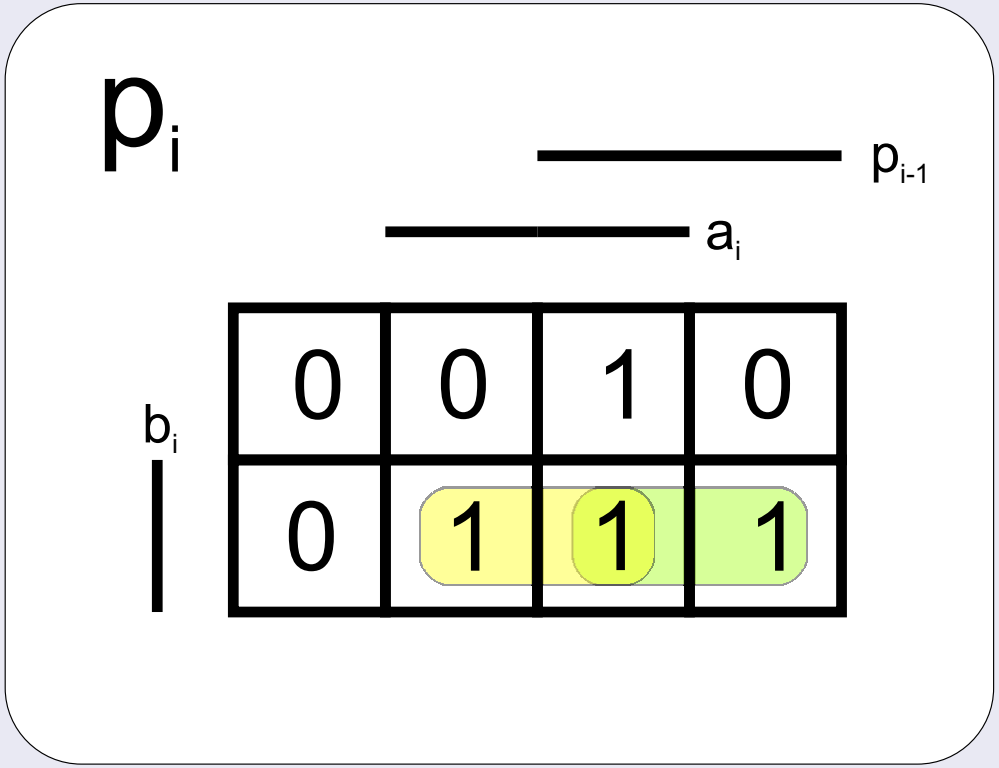
Bitová sčítačka - Karnaughova mapa - přenos do vyššího řádu



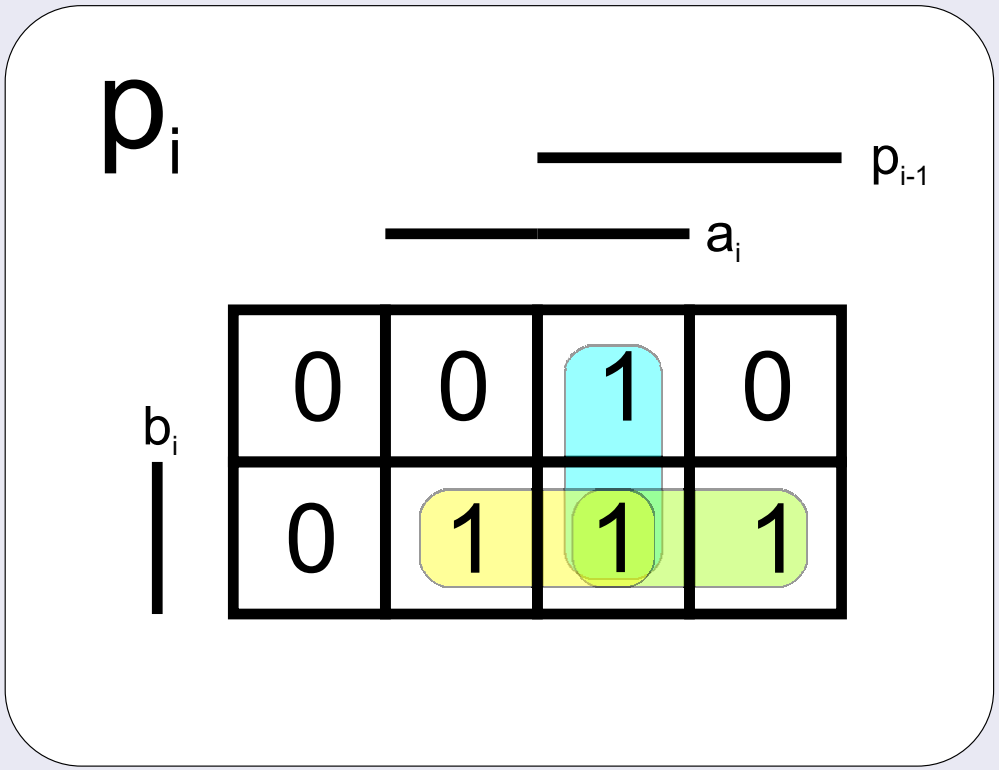
Bitová sčítačka - Karnaughova mapa - přenos do vyššího řádu



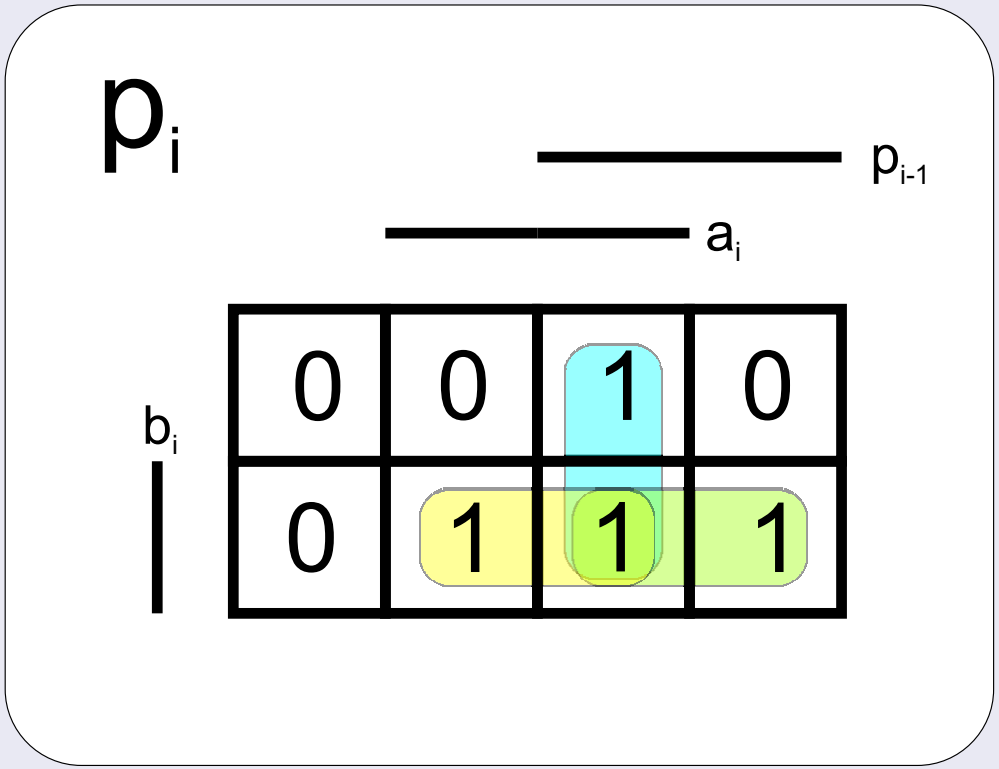
Bitová sčítačka - Karnaughova mapa - přenos do vyššího řádu



Bitová sčítačka - Karnaughova mapa - přenos do vyššího řádu



Bitová sčítačka - Karnaughova mapa - přenos do vyššího řádu



$$p_i = a_i b_i + a_i p_{i-1} + b_i p_{i-1}$$

Bitová sčítačka - realizace