Logické funkce, Booleova algebra

Logické proměnné, elementární logické funkce, Booleova algebra, vyjadřování logických funkcí

Ing. Pavel Lafata, Ph.D.
lafatpav@fel.cvut.cz

Logické funkce, Booleova algebra – logické proměnné, logické funkce

- **Logické proměnné, funkce a jejich vyjadřování**
  - **logická proměnná** – obecně nezávislá matematická proměnná, která může nabývat konečného počtu hodnot – **dvouhodnotová logika** – **logická 1** nebo **logická 0**
    - logické proměnné obvykle zapisujeme malými písmeny: a, b, c, d…
  - **logická funkce** – představuje soubor pravidel pro jednoznačné přiřazení hodnot závisle proměnných (funkčních hodnot) k jednotlivým kombinacím nezávislých proměnných (logických proměnných)
    - logické funkce obvykle zapisujeme písmenem ***f*** s indexem nebo velkými písmeny
    - pro ***n*** proměnných máme $\boldsymbol{2^n}$ různých kombinací a $\boldsymbol{2^{2^n}}$ různých logických funkcí
    - **určitá funkce** – jednoznačně určena hodnota pro všech 2^n kombinací proměnných
    - **neurčitá funkce** – pro alespoň jednu kombinaci proměnných není určena funkční hodnota
    - tím pádem logická funkce může nabývat – **logická 1**, **logická 0**, **X** – neurčitý stav
    - v jazyce VHDL definován datový typ std_logic – 9 hodnot (viz VHDL přednášky)
  - **pravdivostní tabulka** – tabulkový výčet kombinací vstupních proměnných a jim odpovídající funkční hodnota – úplná či neúplná (viz dále)
  - **úplný soubor funkcí** – soubor (skupina) elementárních logických funkcí, pomocí nichž lze vyjádřit libovolnou logickou funkci
  - **minimální úplný soubor funkcí** – minimální soubor elementárních logických funkcí, pomocí nichž lze vyjádřit libovolnou logickou funkci

Logické funkce, Booleova algebra – logické proměnné, logické funkce

1. Logické funkce 1 proměnné – vstupní proměnná *a*

- pro 1 vstupní proměnnou (*a*) – 2 kombinace hodnot – 4 různé logické funkce

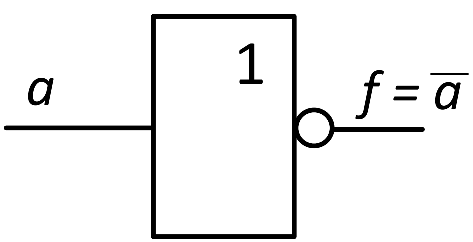
a	f_0	f_1	f_2	f_3
0	0	1	0	1
1	0	0	1	1

- f_0 – nulová funkce, hodnota logická 0 nezávisle na hodnotě vstupní proměnné *a*
- f_3 – jednotková funkce, hodnota logická 1 nezávisle na hodnotě vstupní proměnné *a*
- f_2 – funkce identita, hodnota vždy stejná jako vstupní proměnná *a*:
  $f_2 = a$
- f_1 – **funkce negace**, nabývá vždy přesně opačné hodnoty než proměnná *a*:

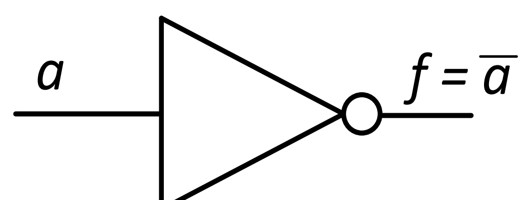
$$\boxed{f_1 = \overline{a}; \quad f_1 = \neg a; \quad f_1 = \mathrm{NOT}\, a}$$

- v literatuře několik způsobů značení, v jazyce VHDL klíčové slovo **NOT**
- v elektronickém obvodu – **hradlo invertor** – negace znázorněna „kroužkem“

značení ČSN EN 60617-12 (TNI 01 3760)



značení ANSI/IEEE Std 91-1984, 91-1991a



2. **Logické funkce 2 proměnných** – vstupní proměnné ***a***, ***b***

- pro 2 proměnné (*a, b*), vytvoříme 4 kombinace hodnot, 16 různých logických funkcí

a	*b*	f_0	f_1	f_2	f_3	f_4	f_5
0	0	0	1	0	1	0	1
0	1	0	1	1	0	1	0
1	0	0	1	1	0	1	0
1	1	1	0	1	0	0	1

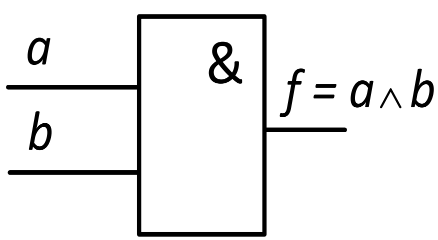
- tabulka obsahuje jen nejdůležitější vybrané funkce

- f_0 – hodnota funkce je logická 1 pouze pokud obě vstupní proměnné *a*, *b* nabývají hodnoty logická 1, jinak je její hodnota logická 0
  - jedná se o **logický součin** (**konjunkci**), značíme obvykle symboly – "$\wedge$", "$\cdot$", "AND"

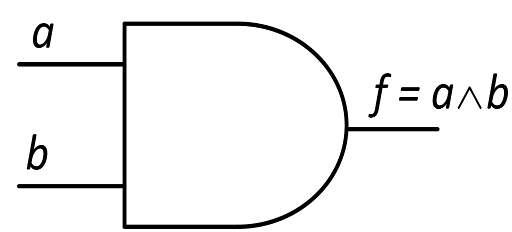
$$\boxed{f_0 = a \wedge b = a \cdot b = ab = a \,\mathrm{AND}\, b}$$

  - v jazyce VHDL klíčové slovo **AND**, obvodová značka – **hradlo logický součin**

značení ČSN EN 60617-12 (TNI 01 3760)



značení ANSI/IEEE Std 91-1984, 91-1991a



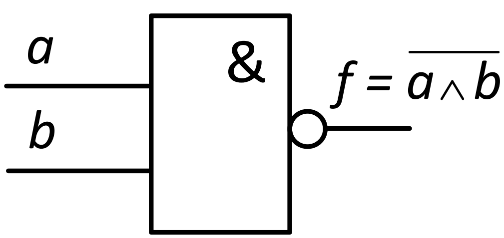
2. **Logické funkce 2 proměnných** – vstupní proměnné ***a***, ***b***

- $\boldsymbol{f_1}$ – funkce negace logického součinu, hodnota funkce je logická 0 pokud obě proměnné *a*, *b* nabývají hodnoty logická 1, v opačném případě je výstup logická 1
  - přesně opačná funkce k funkci logický součin – **negace logického součinu**, v literatuře také nazývána Shefferova funkce
  - NOT AND = **NAND**

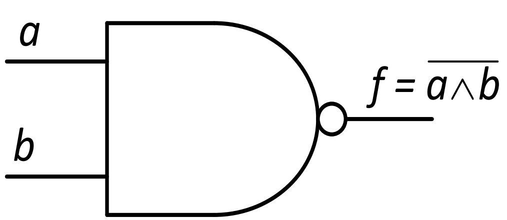
$$\boxed{f_1 = \overline{a \wedge b} = \overline{a \cdot b} = \overline{ab} = a\,\mathrm{NAND}\,b}$$

  - **hradlo negovaného logického součinu** – k výstupu log. součinu přidáme negaci

značení ČSN EN 60617-12 (TNI 01 3760)



značení ANSI/IEEE Std 91-1984, 91-1991a



  - pozor na správný zápis negace, **negace logického součinu *a* a *b* není rovna logickému součinu negace *a* a negace *b*!** – dostaneme se k tomu později

$$\boxed{\overline{a \wedge b} \neq \overline{a} \wedge \overline{b}}$$ **!!!**

Logické funkce, Booleova algebra – logické proměnné, logické funkce

2. **Logické funkce 2 proměnných** – vstupní proměnné ***a***, ***b***

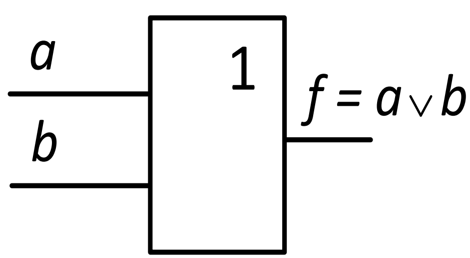
a	***b***	f_0	f_1	f_2	f_3	f_4	f_5
0	0	0	1	0	1	0	1
0	1	0	1	1	0	1	0
1	0	0	1	1	0	1	0
1	1	1	0	1	0	0	1

- f_2 – funkce nabývá logické 0 pouze když obě proměnné *a*, *b* jsou logické 0, pokud alespoň jedna z nich má hodnotu logická 1, hodnota funkce je logická 1
  - funkce se nazývá **logický součet** (**disjunkce**), značíme symboly – "$\vee$", "$+$", "OR"
  - v jazyce VHDL klíčové slovo **OR**

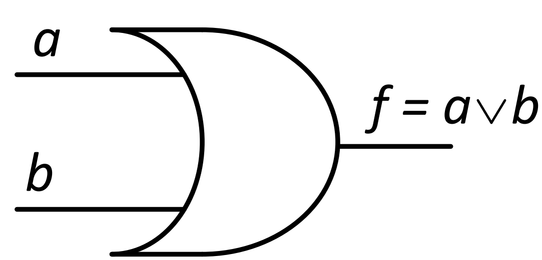
$$\boxed{f_2 = a \vee b = a + b = a\,\mathrm{OR}\,b}$$

  - obvodová značka – **hradlo logický součet**

značení ČSN EN 60617-12 (TNI 01 3760)



značení ANSI/IEEE Std 91-1984, 91-1991a



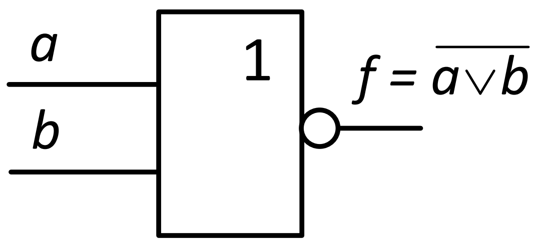
2. **Logické funkce 2 proměnných** – vstupní proměnné ***a***, ***b***

- ***f₃*** – opačná funkce k funkci logický součet, její funkční hodnota je logická 1 pouze když obě proměnné *a*, *b* jsou logické 0, jinak je její hodnota logická 0
  - je to **negace logického součtu** – NOT OR = **NOR**, v literatuře též Piercova funkce
  - v jazyce VHDL klíčové slovo **NOR**

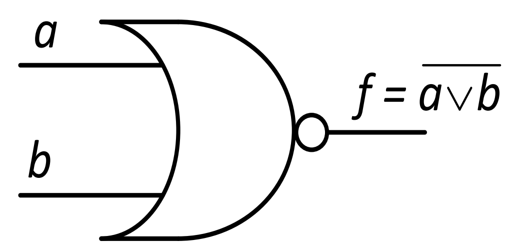
$$\boxed{f_3 = \overline{a \vee b} = \overline{a + b} = a \,\mathrm{NOR}\, b}$$

  - **hradlo negovaného logického součtu** – k výstupu log. součtu přidáme negaci

značení ČSN EN 60617-12 (TNI 01 3760)



značení ANSI/IEEE Std 91-1984, 91-1991a



  - opět, pozor na negaci, **negace logického součtu *a* a *b* není rovna logickému součtu negace *a* a negace *b*!** – viz dále

$$\boxed{\overline{a \vee b} \neq \overline{a} \vee \overline{b}}$$ **!!!**

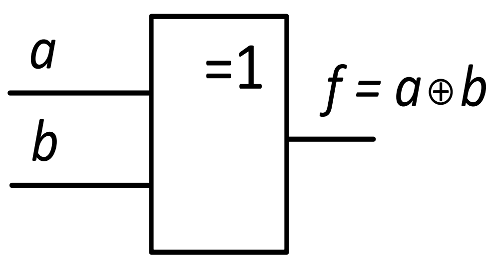
2. **Logické funkce 2 proměnných** – vstupní proměnné ***a***, ***b***

- $\boldsymbol{f_4}$ – tato funkce nabývá hodnoty logická 1 pokud hodnoty obou proměnných jsou navzájem různé, a hodnoty logická 0 pokud jsou stejné
  - podobná funkci logického součtu, ale na rozdíl od něj dává funkce logickou 0 pokud *a* = *b* = 1 (logický součet dává logickou 1 pro tuto vstupní kombinaci)
  - může platit jen jedna vstupní podmínka (buď jen *a* = 1 nebo jen *b* = 1) – „vyloučené nebo“ = **exclusive OR** = **XOR** (někdy též funkce neekvivalence)
  - značíme pomocí symbolů – $"\oplus", "\nleftrightarrow", "\not\equiv", "\mathrm{XOR}"$
  - a můžeme ji definovat pomocí předchozích funkcí:

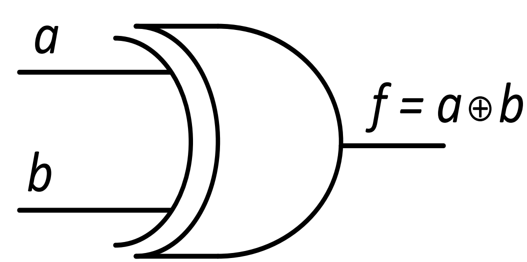
$$\boxed{f_4 = a \oplus b = \overline{a} \cdot b \vee a \cdot \overline{b}}$$

  - obvodová značka – vychází ze značky logického součtu

značení ČSN EN 60617-12 (TNI 01 3760)



značení ANSI/IEEE Std 91-1984, 91-1991a



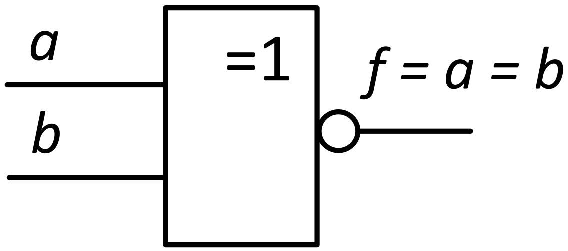
2. **Logické funkce 2 proměnných** – vstupní proměnné ***a***, ***b***

- $\boldsymbol{f_5}$ – opačná funkce k funkci XOR, její hodnota je logická 1 pokud hodnoty obou vstupních proměnných *a*, *b* jsou shodné, pokud *a*, *b* jsou různé, dává logickou 0
  - negace XOR = **XNOR**, někdy též funkce ekvivalence
  - značíme ji pomocí symbolů – $"=", "\leftrightarrow", "\Leftrightarrow", "\mathrm{EQ}", "\mathrm{XNOR}"$
  - v jazyce VHDL klíčové slovo **XNOR**
  - můžeme ji vyjádřit jako:

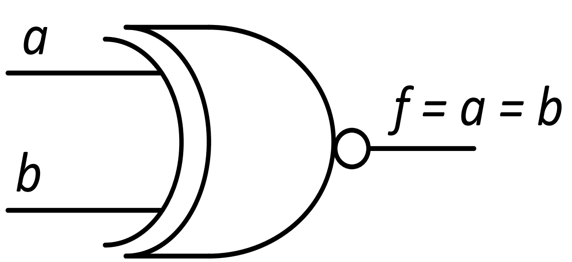
$$\boxed{f_5 = \overline{a \oplus b} = a = b = \overline{a} \cdot \overline{b} \vee a \cdot b}$$

  - obvodová značka – doplnění negace do symbolu hradla XOR

značení ČSN EN 60617-12 (TNI 01 3760)



značení ANSI/IEEE Std 91-1984, 91-1991a



Logické funkce, Booleova algebra – logické proměnné, logické funkce

Elementární logické funkce – pravdivostní tabulka, shrnutí

a	***b***	**NOT *a***	**NOT *b***	**AND**	**NAND**	**OR**	**NOR**	**XOR**	**XNOR**
0	0	1	1	0	1	0	1	0	1
0	1	1	0	0	1	1	0	1	0
1	0	0	1	0	1	1	0	1	0
1	1	0	0	1	0	1	0	0	1

- **Logické funkce, úplný soubor funkcí**
  - na počátku jsme definovali **úplný soubor funkcí** a **minimální úplný soubor funkcí** – pomocí něho můžeme vyjádřit libovolnou logickou funkci
  - příklady minimálních úplných souborů:
    1. **funkce NAND** – negovaný logický součin sám o sobě je minimálním úplným souborem funkcí a lze pomocí něho vyjádřit libovolnou logickou funkci
    2. **funkce NOR** – i negovaný logický součet je minimálním úplným souborem funkcí
    - tyto 2 funkce jsou tedy důležité při realizaci logických funkcí pomocí hradel
    3. **funkce AND + NOT** – logický součin a negace
    4. **funkce OR + NOT** – logický součet a negace
  - příklad (neminimálního) úplného souboru:
    1. **funkce AND**, **OR**, **NOT** – trojice funkcí: logický součin, logický součet, negace, pomocí nich lze vyjádřit libovolnou logickou funkci

- **Booleova algebra**
  - algebra = obecně matematický prostor obsahující definice symbolů, operací a pravidel pro manipulaci s těmito symboly
  - Booleova algebra – část algebry zabývající se logickými proměnnými a funkcemi
  - pojmenována na počest po Georgeovi Booleovi, formulace základů matematické logiky a informatiky: The Laws of Thought (1854)

Logické funkce, Booleova algebra – logické funkce, Booleova algebra

- ## Booleova algebra
  1. **definovány dva základní prvky** – logická 0 (nepravda), logická 1 (pravda)
  2. **Booleova algebra založena na úplném souboru funkcí** – AND, OR, NOT

  - **zákony Booleovy algebry:**
    1. Asociativita logického součtu
       $$a \vee (b \vee c) = (a \vee b) \vee c$$
    2. Asociativita logického součinu
       $$a(bc) = (ab)c$$
    3. Komutativita logického součtu
       $$a \vee b = b \vee a$$
    4. Komutativita logického součinu
       $$ab = ba$$
    5. Distributivní zákon
       $$a(b \vee c) = ab \vee ac$$
       $$(a \vee b) \cdot (a \vee c) = a \vee bc$$

Logické funkce, Booleova algebra – logické funkce, Booleova algebra

- **zákony Booleovy algebry:**

6. Zákon o vyloučení třetího

$$a \cdot \overline{a} = 0$$
$$a \vee \overline{a} = 1$$

7. Zákon o idempotenci prvků

$$a \vee a = a$$
$$a \cdot a = a$$

8. Zákon agresivity nuly

$$a \cdot 0 = 0$$

9. Zákon agresivity jedničky

$$a \vee 1 = 1$$

10. Zákon neutrálnosti nuly

$$a \vee 0 = a$$

11. Zákon neutrálnosti jedničky

$$a \cdot 1 = a$$

Logické funkce, Booleova algebra – logické funkce, Booleova algebra

- **zákony Booleovy algebry:**

12. Zákon absorpce

$$a\left(a \vee b\right) = a$$
$$a \vee ab = a$$

12. Zákon absorpce negace

$$a \vee \overline{a}b = a \vee b$$
$$\overline{a} \vee ab = \overline{a} \vee b$$

13. Zákon dvojité negace

$$\overline{\overline{a}} = a$$

14. Zákon o negaci logického součinu

$$\overline{ab} = \overline{a} \vee \overline{b}$$

15. Zákon o negaci logického součtu

$$\overline{a \vee b} = \overline{a}\overline{b}$$

De Morganovy zákony

- **Booleova algebra a jazyk VHDL**
  - **Distributivní zákon dle Booleovy algebry vs. jazyk VHDL**
  - v Booleově algebře má logický součin vyšší prioritu než logický součet
  - můžeme proto zapsat bez použití závorek např.: $a \vee \overline{b}c$
  - závorky nejsou nutné: $a \vee \left(\overline{b}c\right) = a \vee \overline{b}c$
  - v jazyce VHDL je to však jinak:
    nejvyšší prioritu má – NOT (negace)
    nižší (stejnou) prioritu mají – AND, OR, NAND, NOR, XOR, XNOR (všechny ostatní)
  - logický součin a logický součet tedy mají **ve VHDL stejnou prioritu! – nutné závorky**
  - ve VHDL navíc musíme vždy **oddělit pomocí závorek všechny nestejné operace** (kromě NOT)**!**
  - uvedený výraz v jazyce VHDL tedy zapíšeme:
    $a\,\mathrm{OR}\left(\mathrm{NOT}\,b\,\mathrm{AND}\,c\right)$

Logické funkce, Booleova algebra – vyjadřování logických funkcí

- **Logické funkce – vyjadřování logických funkcí, nejčastěji pomocí:**
  1. Pravdivostní tabulka
  2. Zkrácený seznam stavových indexů
  3. Algebraický zápis
  4. Mapy

- **Pravdivostní tabulka**
  - tabulkový výčet jednotlivých kombinací nezávislých logických proměnných a jim přiřazených hodnot dané logické funkce
  - obvykle přidáváme sloupec stavových indexů *N*
  - **úplná pravdivostní tabulka** – obsahuje všechny kombinace vstupních proměnných
    - pro ***n*** vstupních proměnných obsahuje $\boldsymbol{2^n}$ řádků
  - **neúplná (zkrácená) pravdivostní tabulka** – z úplné pravdivostní tabulky vynecháme neurčité stavy (neurčité stavy zapisujeme X), počet řádků neúplné tabulky $\boldsymbol{\leq 2^n}$
  - pokud dvě logické funkce mají navzájem shodné pravdivostní tabulky, jedná se o identické logické funkce (a naopak)
  - pravdivostní tabulka je vhodná pro popis logického obvodu (výstupní funkce), ale nehodí se pro minimalizaci ani realizaci pomocí hradel

Logické funkce, Booleova algebra – vyjadřování logických funkcí

- ## **Pravdivostní tabulka**
  - příklad:
    3 vstupní proměnné – *a*, *b*, *c*, obvykle přiřazujeme *a* na LSB pozici a *c* na MSB pozici

úplná pravdivostní tabulka funkce *f*

N	***c***	***b***	***a***	***f***
0	0	0	0	0
1	0	0	1	×
2	0	1	0	1
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	1

zkrácená pravd. tabulka funkce *f*

N	***c***	***b***	***a***	***f***
0	0	0	0	0
2	0	1	0	1
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	1

  - ***N*** je sloupec stavových indexů – binární kombinace nezávislých proměnných vyjádřená v desítkové soustavě
  - s počtem proměnných *n* narůstá exponenciálně počet řádků úplné tabulky (2^n)

Logické funkce, Booleova algebra – vyjadřování logických funkcí

- **Zkrácený seznam stavových indexů**
  - zkrácený zápis pravdivostní tabulky
  - seznam indexů ***N***, ve kterých nabývá daná funkce **logické 1 nebo neurčitého stavu**
  - pro rozlišení zapisujeme neurčité stavy **do závorek**
  - předchozí funkci *f* můžeme zapsat ve formě zkráceného seznamu stavových indexů:
    f = (1), 2, 3, 5, 6, 7

- **Algebraický zápis**
  - zápis ve formě funkčního vyjádření pomocí trojice elementárních logických funkcí – log. součtu, log. součinu a negace a s použitím pravidel a zákonů Booleovy algebry
  - nejprve několik potřebných definicí:
  - **nezávislá proměnná** – jedna vstupní proměnná funkce v přímém či negovaném tvaru
  - **podstatný bod** – stav, kdy daná funkce nabývá hodnoty logická 1
  - **nepodstatný bod** – stav, kdy funkce nabývá logické 0 nebo neurčitého stavu
  - **term** – jeden výraz obsahující každou nezávislou proměnnou přesně jedenkrát
  - **součinový term** – term obsahující jen operaci logického součinu – vytvořený jako logický součin vstupních nezávislých proměnných
  - **součtový term** – obdobně, term obsahující jen operaci logický součet
  - **minterm** – součinový term obsahující všechny vstupní nezávislé proměnné právě jednou v přímém či negovaném tvaru
  - **maxterm** – obdobně, součtový term obsahující všechny proměnné právě jednou

Logické funkce, Booleova algebra – vyjadřování logických funkcí

- ## Algebraický zápis
  - **úplná normální disjunktní forma** (**ÚNDF**) – forma funkce obsahující logické součty mintermů – **součet součinů** – v literatuře též **Sum of Products**, **SoP**
  - obsahuje tolik součinů, kolik má daná funkce podstatných (jednotkových) bodů

$$\boxed{f_D = \bigvee_{N=0}^{2^n-1} f_N C_N = \sum_{N=0}^{2^n-1} f_N C_N}$$

  - **kde:**
    V, Σ – logický součet (OR), ***N*** – index, f_N – hodnota funkce ***f*** pro daný index ***N***,
    C_N – logický součin nezávislých proměnných pro daný index ***N***
  - **minimální normální disjunktní forma** (**MNDF**) – minimalizovaná forma ÚNDF dané funkce ***f***, obsahující nejmenší možný počet mintermů
  - vyjádření funkce ***f*** pomocí ÚNDF:
  - pro každý řádek úplné pravdivostní tabulky dané funkce vytvoříme **logický součin funkční hodnoty f_N a vstupních proměnných *a*, *b*, *c*** – pokud obsahuje pro daný řádek tabulky proměnná log. 1, zapíšeme ji přímo, v případě log. 0 zapíšeme její negaci
  - využijeme zákony o neutrálnosti jedničky a agresivity nuly
  - výslednou ÚNDF formu zapíšeme jako **součet jednotlivých mintermů**
  - předchozí příklad funkce *f* – neurčitý stav považujeme za logickou 1:

$$f_D = 0\cdot\overline{a}\,\overline{b}\,\overline{c} \vee 1\cdot a\overline{b}\,\overline{c} \vee 1\cdot\overline{a}b\overline{c} \vee 1\cdot ab\overline{c} \vee 0\cdot\overline{a}\,\overline{b}c \vee 1\cdot a\overline{b}c \vee 1\cdot\overline{a}bc \vee 1\cdot abc$$

$$f_D = a\overline{b}\,\overline{c} \vee \overline{a}b\overline{c} \vee ab\overline{c} \vee a\overline{b}c \vee \overline{a}bc \vee abc$$

Logické funkce, Booleova algebra – vyjadřování logických funkcí

- ## Algebraický zápis
  - **úplná normální konjunktní forma** (**ÚNKF**) – forma funkce obsahující logické součiny maxtermů – **součin součtů** – v literatuře též **Product of Sums**, **PoS**
  - obsahuje tolik součtů, kolik má daná funkce nepodstatných bodů

$$\boxed{f_K = \bigwedge_{N=0}^{2^n-1} f_N \vee D_N = \prod_{N=0}^{2^n-1} f_N \vee D_N}$$

  - **kde:**
    Λ, $\prod$ – logický součin (AND), ***N*** – index, $\boldsymbol{f_N}$ – hodnota funkce ***f*** pro daný index ***N,***
    $\boldsymbol{D_N}$ – logický součet nezávislých proměnných pro daný index ***N***
  - **minimální normální konjunktní forma (MNKF)** – minimalizovaná forma ÚNKF dané funkce ***f***, obsahující nejmenší možný počet maxtermů
  - vyjádření funkce ***f*** pomocí ÚNKF:
  - pro každý řádek úplné pravdivostní tabulky dané funkce vytvoříme **logický součet funkční hodnoty $\boldsymbol{f_N}$ a <u>negace</u> logického součinu proměnných *a*, *b*, *c*** – proměnné zapisujeme v přímém či negovaném tvaru stejně jako v případě ÚNDF
  - využijeme zákony o neutrálnosti nuly a agresivity jedničky a De Morganovy zákony
  - výslednou ÚNKF formu zapíšeme jako **součin jednotlivých maxtermů**
  - předchozí příklad funkce *f* – neurčitý stav považujeme za logickou 0:

$$f_K = \left(0 \vee \overline{\bar{a}\bar{b}\bar{c}}\right) \cdot \left(0 \vee \overline{a\bar{b}\bar{c}}\right) \cdot \left(1 \vee \overline{\bar{a}b\bar{c}}\right) \cdot \left(1 \vee \overline{ab\bar{c}}\right) \cdot \left(0 \vee \overline{\bar{a}\bar{b}c}\right) \cdot \left(1 \vee \overline{a\bar{b}c}\right) \cdot \left(1 \vee \overline{\bar{a}bc}\right) \cdot \left(1 \vee \overline{abc}\right)$$

$$f_K = (a \vee b \vee c) \cdot (\bar{a} \vee b \vee c) \cdot (a \vee b \vee \bar{c})$$

Logické funkce, Booleova algebra – vyjadřování logických funkcí

- ## Algebraický zápis
  - **konjunktní formu funkce** získáme z těch řádků pravdivostní tabulky, kdy funkce *f* dává hodnotu **logická 0 nebo neurčitý stav** – zákon o neutrálnosti nuly
  - **disjunktní formu funkce** získáme z těch řádků pravdivostní tabulky, kdy funkce *f* dává hodnotu **logická 1 nebo neurčitý stav** – zákon o neutrálnosti jedničky
  - **každou funkci *f*** proto můžeme vždy zapsat v její **disjunktní i konjunktní formě**

  - **disjunktní a konjunktní formu dané funkce můžeme navzájem převádět:**

1. nahradíme všechny logické součty (V) pomocí logických součinů (Λ) a naopak
2. ve všech výrazech nahradíme jednotlivé proměnné jejich negacemi a naopak
3. musíme vždy převádět úplné formy, tzn. obsahující všechny řádky pravdivostní tabulky funkce (s hodnotou logická 0 i 1), neurčité stavy ponecháme vždy v obou formách

- příklad – převeďte ÚNDF předchozí funkce ***f*** na její ÚNKF:

ÚNDF:
$$f_D = 0\cdot\bar{a}\bar{b}\bar{c} \vee \mathrm{X}\cdot a\bar{b}\bar{c} \vee 1\cdot\bar{a}b\bar{c} \vee 1\cdot ab\bar{c} \vee 0\cdot\bar{a}\bar{b}c \vee 1\cdot a\bar{b}c \vee 1\cdot\bar{a}bc \vee 1\cdot abc =$$
$$= \underline{a\bar{b}\bar{c} \vee \bar{a}b\bar{c} \vee ab\bar{c} \vee a\bar{b}c \vee \bar{a}bc \vee abc}$$

ÚNKF:
$$f_K = (0 \vee a \vee b \vee c)\cdot(\mathrm{X} \vee \bar{a} \vee b \vee c)\cdot(1 \vee a \vee \bar{b} \vee c)\cdot(1 \vee \bar{a} \vee \bar{b} \vee c)\cdot(0 \vee a \vee b \vee \bar{c})\cdot$$
$$\cdot(1 \vee \bar{a} \vee b \vee \bar{c})\cdot(1 \vee a \vee \bar{b} \vee \bar{c})\cdot(1 \vee \bar{a} \vee \bar{b} \vee \bar{c}) =$$
$$= \underline{(a \vee b \vee c)\cdot(\bar{a} \vee b \vee c)\cdot(a \vee b \vee \bar{c})}$$

Logické funkce, Booleova algebra – vyjadřování logických funkcí

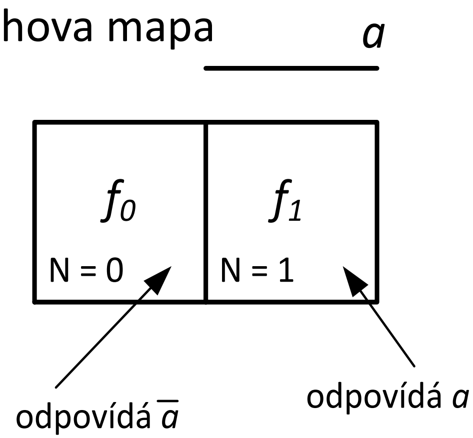
- **Karnaughovy mapy**
  - existuje více druhů map, Karnaughovy mapy do 6 proměnných nejvhodnější
  - grafická reprezentace funkce *f*
  - mapa má vždy 2^n políček – kde ***n*** je počet nezávislých proměnných funkce
  - **každé políčko v Karnaughově mapě** představuje **jeden řádek pravdivostní tabulky**
  - **každé políčko v Karnaughově mapě** představuje **jeden minterm** funkce *f*
  - políčka v Karnaughově mapě **očíslována pomocí Grayova kódu** – indexy navzájem sousedních políček vyjádřené ve dvojkové soustavě se liší jen na jedné řádové pozici
  - proměnné v mapě obvykle přiřazujeme od ***a*** – **LSB** až **„poslední“** – **MSB**
  - oblasti pokrytí mapy jednotlivými proměnnými znázorňujeme po stranách mapy

- **Karnaughova mapa 1 proměnné** – *a*

pravdivostní tabulka

N	***a***	***f***
0	0	f_0
1	1	f_1

Karnaughova mapa



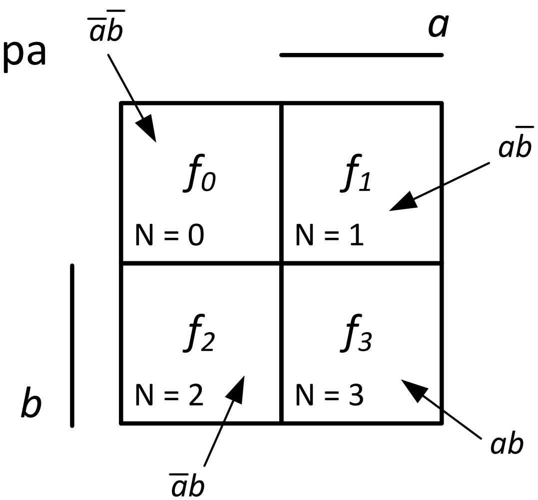
Logické funkce, Booleova algebra – vyjadřování logických funkcí

- **Karnaughova mapa 2 proměnných** – *a, b*

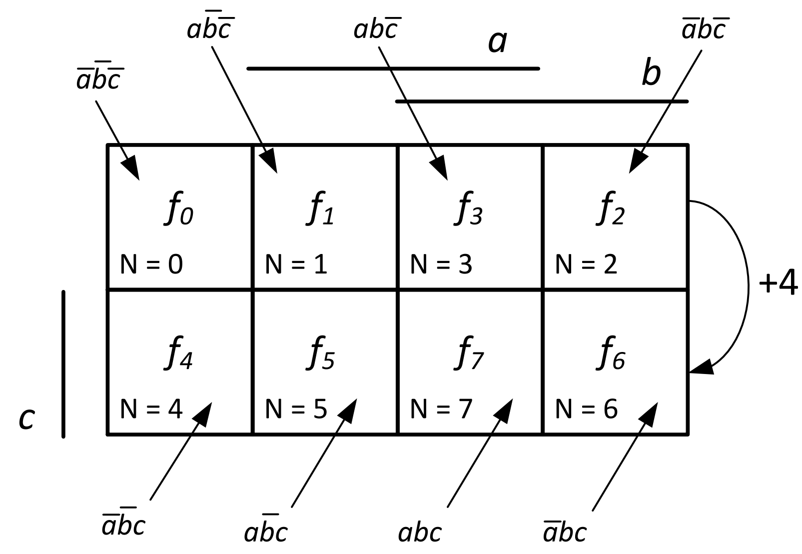
pravdivostní tabulka

N	***b***	***a***	***f***
0	0	0	f_0
1	0	1	f_1
2	1	0	f_2
3	1	1	f_3

Karnaughova mapa

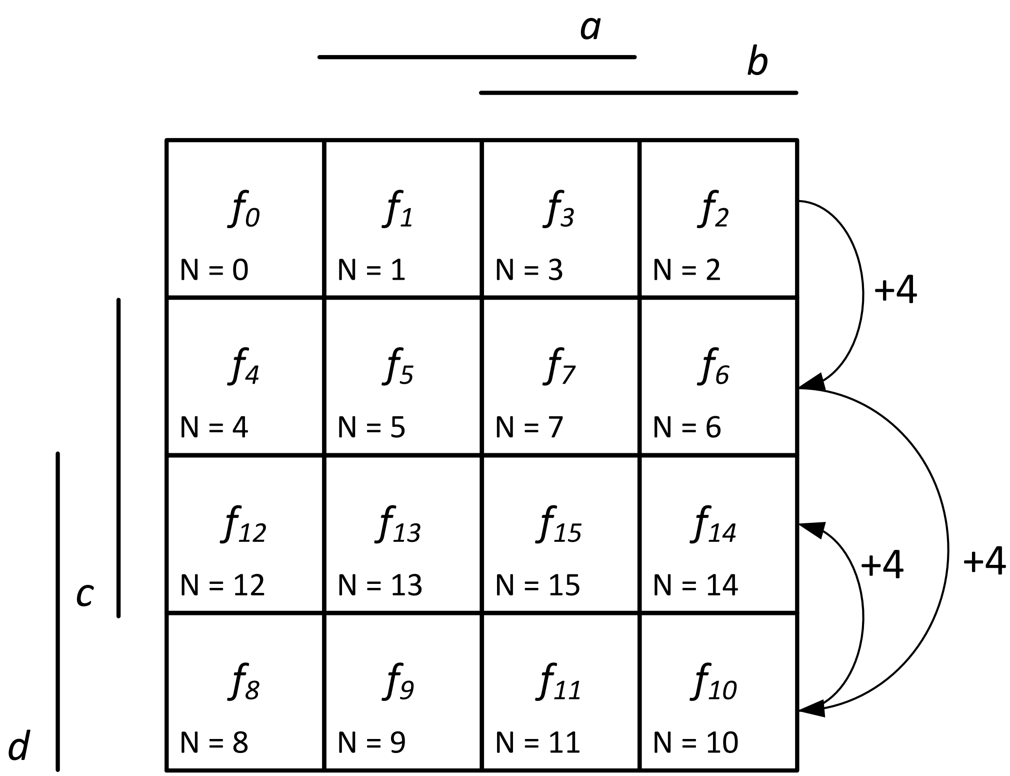


- **Karnaughova mapa 3 proměnných** – *a, b, c*
  - první řádek očíslujeme 0, 1, 3, 2 (Grayův kód)
  - druhý řádek je zvětšen o +4



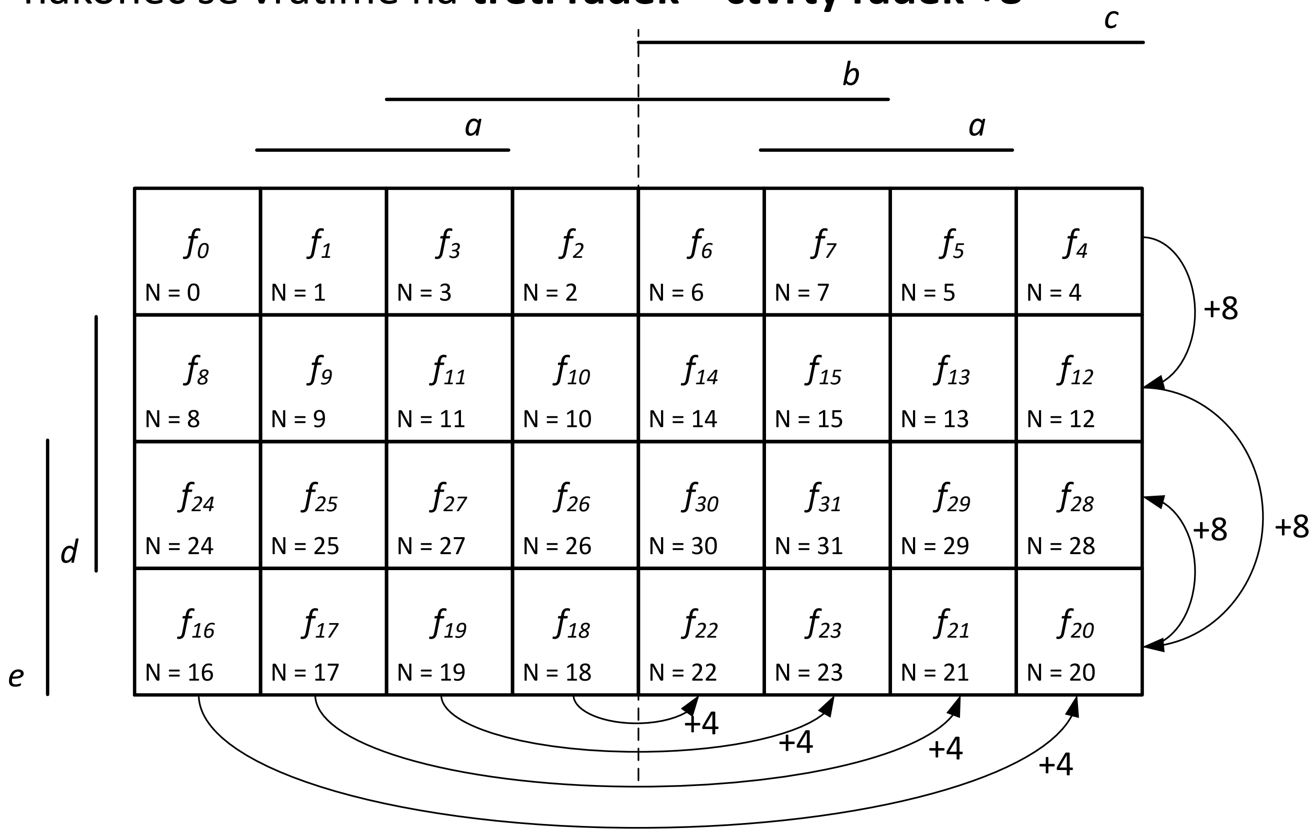
Logické funkce, Booleova algebra – vyjadřování logických funkcí

- **Karnaughova mapa 4 proměnných** – *a, b, c, d*
  - **první řádek** očíslujeme **0, 1, 3, 2** (Grayův kód)
  - **druhý řádek = první řádek +4**
  - pak přeskočíme na **čtvrtý řádek = druhý řádek +4**
  - nakonec se vrátíme na **třetí řádek = čtvrtý řádek +4**



Logické funkce, Booleova algebra – vyjadřování logických funkcí

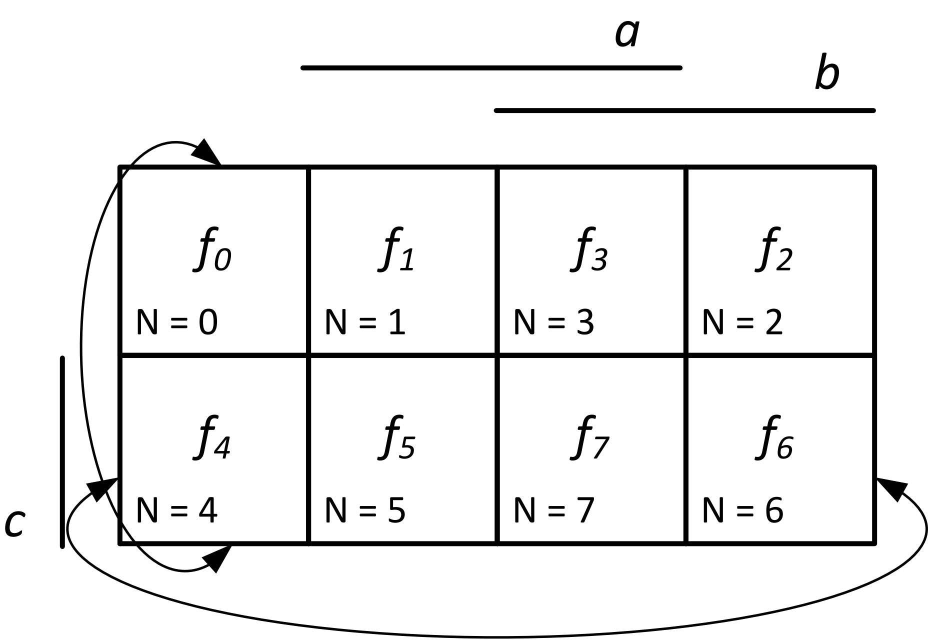
- **Karnaughova mapa 5 proměnných** – *a, b, c, d, e*
  - **osa symetrie rozděluje mapu na dvě poloviny** – levá + pravá část (osově symetrické)
  - vždy **políčko v pravé polovině** mapy = **políčko v levé polovině +4**
  - první řádek levá část 0, 1, 3, 2 (Grayův kód) – pravá část = 6, 7, 5, 4
  - **druhý řádek = první řádek +8**
  - pak přeskočíme na **čtvrtý řádek = druhý řádek +8**
  - nakonec se vrátíme na **třetí řádek = čtvrtý řádek +8**



- oblast pokrytí proměnnou ***a*** **rozdělena na dvě části** (osově symetrické)

Logické funkce, Booleova algebra – vyjadřování logických funkcí

- **Karnaughovy mapy – sousední políčka**
- indexy sousedních políček v Karnaughově mapě zapsané ve dvojkové soustavě – změna jen na jedné řádové pozici – vždy jen jedna proměnná se liší (přímý vs. negovaný tvar)
- každá **Karnaughova mapa může být přeložena zleva-doprava, seshora-dolů**, mapy pro 5 a více proměnných navíc **osy symetrie** – **políčka z okrajů po složení navzájem sousední**



- **příklad** – mapa pro 3 proměnné
- **složení mapy** – políčka č. 6 a 4 jsou sousední, políčka č. 0 a 2 jsou sousední, políčka č. 0 a 4 jsou sousední a políčka č. 2 a 6 jsou sousední

- **jiný příklad**, Karnaughova mapa pro 4 proměnné, jsou políčka č. 12 a 14 sousední?
řešení – rozepíšeme dané indexy ve dvojkové soustavě:
$\mathbf{12}_{(10)}$ -> $\mathbf{1100}_{(2)}$ -> $dc\overline{b}\overline{a}$
$\mathbf{14}_{(10)}$ -> $\mathbf{1110}_{(2)}$ -> $dcb\overline{a}$
obě čísla se navzájem liší jen na jedné řádové pozici (proměnná *b*) = **políčka jsou sousední**

Logické funkce, Booleova algebra – vyjadřování logických funkcí

- **Karnaughovy mapy – zápis funkce pomocí mapy**
- napíšeme “**1**” (nebo čáru) do políčka s indexem, ve kterém nabývá funkce logické 1, “**X**” do políčka, ve kterém je hodnota funkce neurčitý stav a “**0**” do políčka s indexem, ve kterém dává funkce hodnotu logická 0
- příklad – Karnaughova mapa předchozí funkce *f*:

			a	
				b
	0 N = 0	X N = 1	1 N = 3	1 N = 2
c	0 N = 4	1 N = 5	1 N = 7	1 N = 6

- Karnaughovy mapy pro více než 6 proměnných nepřehledné
- Karnaughovy mapy užitečné hlavně pro **minimalizaci tvaru funkce**