

Minimalizace logických funkcí a jejich realizace pomocí hradel, kombinační log. obvody

Booleova algebra, Karnaughovy mapy, realizace funkcí pomocí hradel, kombinační logické obvody

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

Minimalizace logických funkcí, realizace – metody minimalizace

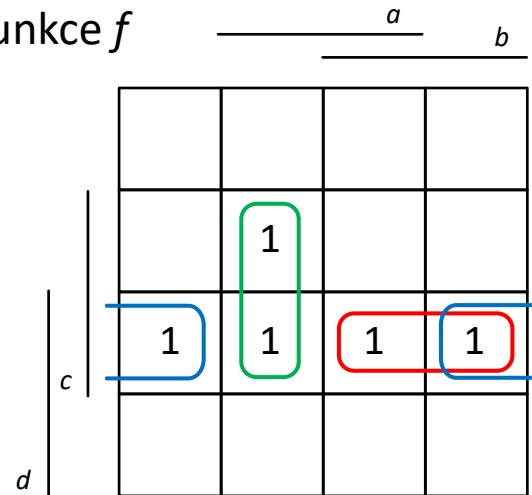
Minimalizace logických funkcí – metody, definice

- **opakování** – co je ÚNDF, ÚNKF? Co je term? Co je minterm a co je maxterm?
- **implikant funkce f** – term, který nabývá hodnoty logické 1 pro stejné stavy (kombinace vstupních proměnných), ve kterých i funkce f nabývá hodnoty logická 1
- **prostý implikant funkce f** – implikant funkce f , ze kterého když vyloučíme kteroukoliv z jeho proměnných, přestane být implikantem funkce f

- příklad – funkce f :

$$f = a\bar{b}c \vee \bar{a}cd \vee bcd$$

- je implikantem funkce f : $a\bar{b}cd$
- je prostým implikantem funkce f : cd
- není implikantem funkce f : bc



- **minimální normální disjunktí, konjunktí forma – MNDF, MNKF** – disjunktí či konjunktí forma funkce obsahující nejmenší počet prostých implikantů funkce f
 - **minimalizace** = proces hledání prostých implikantů funkce
1. **algebraická metoda minimalizace** – aplikace zákonů Booleovy algebry
 2. **Karnaughovy mapy** – grafická metoda minimalizace
 3. **Algoritmus Quine-McCluskey** – vhodný pro počítačové řešení minimalizace funkce

Minimalizace logických funkcí, realizace – metody minimalizace

▪ Realizace logických funkcí pomocí hradel

- proč je vhodné minimalizovat (optimalizovat) tvar logických funkcí?
 - v praxi realizujeme zadané funkce pomocí diskrétních logických hradel – NAND, NOR, AND, OR, XOR, NOT...
 - pro realizaci minimalizované (optimalizované) formy funkce je obvykle potřeba méně hradel – obvykle menší doba zpoždění, spotřeba, plocha obvodu...
 - optimalizace – zejména z hlediska počtu typů hradel, počtu vstupů hradel...
- procesem minimalizace obdržíme buď **MNDF** či **MNKF**
- co znamená **minimální úplný soubor funkcí**? Jaké minimální soubory znáte?
- funkce **NAND** a **NOR** – každá z nich sama o sobě tvoří minimální úplný soubor funkcí
- pro zopakování:

<i>a</i>	<i>b</i>	NAND	NOR
0	0	1	1
0	1	1	0
1	0	1	0
1	1	0	0

$$\text{NAND} - f = \overline{ab} = \bar{a} \vee \bar{b}$$

$$\text{NOR} - f = \overline{a \vee b} = \bar{a} \bar{b}$$

- **MNDF** forma – vycházíme z **jednotkových bodů a neurčitých stavů funkce** – realizujeme pomocí hradel **NAND**
- **MNKF** forma – vycházíme z **nulových bodů a neurčitých stavů funkce** – realizujeme pomocí hradel **NOR**

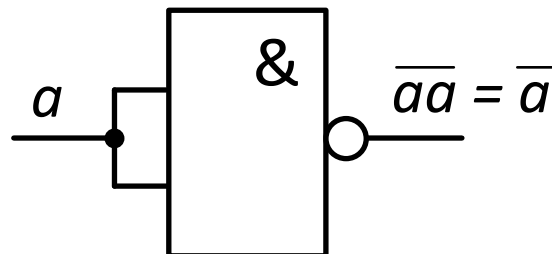
Minimalizace logických funkcí, realizace – metody minimalizace

▪ Realizace negace pomocí hradel NAND, NOR

- v jednom obvodu často chceme použít jen jeden typ hradel – jak vytvořit negaci?

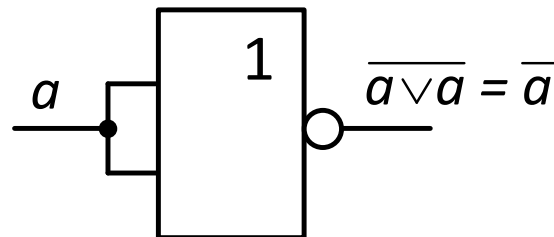
1. negace NOT pomocí hradla NAND – spojíme oba vstupy hradla

a	b	NAND	$a = b$	NAND
0	0	1	0	1
0	1	1	-	-
1	0	1	-	-
1	1	0	1	0



2. negace NOT pomocí hradla NOR – spojíme oba vstupy hradla

a	b	NOR	$a = b$	NOR
0	0	1	0	1
0	1	0	-	-
1	0	0	-	-
1	1	0	1	0



Minimalizace logických funkcí, realizace – minimalizace algebraickou metodou

▪ Minimalizace logických funkcí algebraickou metodou

- přímá aplikace zákonů a pravidel Booleovy algebry
- obvykle komplikované, náročné na čas, vhodné pouze ve specifických případech, použitelné pro malý počet proměnných, do jisté míry intuitivní

▪ Minimalizace logických funkcí grafickými metodami

- grafická reprezentace funkce a způsob její minimalizace
- Karnaughovy mapy, Svobodovy mapy, n-rozměrná tělesa a jejich rovinný rozvoj
- do nižšího počtu proměnných (5-6) rychlé, jednoduché, přehledné, pro vyšší počty proměnných nepřehledné a nepoužitelné
- Intuitivní hledání smyček (oblastí) v mapách a tělesech – nerealizovatelné programově, nelze zkontrolovat správnost výsledku

▪ Minimalizace logických funkcí pomocí algoritmů

- aplikace zákonů a pravidel Booleovy algebry (distributivní z., z. absorpce a abs.negace)
- Algoritmus Quine-McCluskey
- použitelné i pro vyšší počet proměnných (v závislosti na použitém HW)
- vhodné pro PC zpracování a realizaci minimalizace pomocí PC programu
- možnost kontroly řešení, nalezení všech možných řešení

Minimalizace logických funkcí, realizace – minimalizace pomocí Karnaughovy mapy

▪ Minimalizace logických funkcí pomocí Karnaughových map

1. pro získání:

MNDF = realizace hradly **NAND** – vyplníme do mapy **logické 1** a **neurčité stavy**

MNKF = realizace hradly **NOR** – vyplníme do mapy **logické 0** a **neurčité stavy**

2. vytvoříme smyčky (podmapy) z políček podle pravidel:

a) **můžeme spojovat pouze sousední políčka** – viz minulá přednáška

b) **smíme spojovat jen políčka se stejnými hodnotami:**

- logické 1 + neurčité stavy, logické 0 + neurčité stavy

c) **počet spojených políček ve smyčce musí být vždy mocnina 2:** 1, 2, 4, 8, 16, 32...

d) **smyčky volíme co největší** – jen tak získáme co nejvíce minimalizovanou formu

e) **výsledný počet smyček co nejmenší** – tím méně implikantů bude ve výsledku

f) **smyčky se mohou překrývat** – políčko může být zahrnuto v libovolném počtu smyček

3. tímto postupem – **musíme vždy pokrýt (zahrnout) do smyček:**

při hledání MNDF formy – všechny logické 1 musí být zahrnuty vždy alespoň v 1 smyčce

při hledání MNKF formy – všechny logické 0 musí být zahrnuty vždy alespoň v 1 smyčce

neurčité stavy (v obou formách) – můžeme, ale nemusíme je použít (pokrýt) ve smyčkách, použijeme každý pouze pokud se nám „hodí“ pomocí něho vytvořit větší smyčku, získat menší celkový počet smyček apod.

4. každou vytvořenou **smyčku pak zapíšeme jako logický součin těch proměnných, které se pro všechna políčka dané smyčky nemění**

5. výsledné řešení je pak **logický součet všech smyček (MNDF) nebo jeho negace (MNKF)**

6. někdy můžeme získat **více řešení** (rovnocenně minimálních)

Minimalizace logických funkcí, realizace – minimalizace pomocí Karnaughovy mapy

▪ Minimalizace logických funkcí pomocí Karnaughových map

- grafické hledání implikantů a prostých implikantů funkce v mapě
- libovolná smyčka v mapě – **představuje implikant funkce f**
- největší smyčka(y) v mapě (nelze vytvořit větší) – **je prostý implikant funkce f**
- výsledná MNDF či MNKF forma – **vytvořena jen z prostých implikantů**

Minimalizace logických funkcí, realizace – minimalizace pomocí Karnaughovy mapy

Minimalizace logických funkcí pomocí Karnaughových map

- **příklad 1** – najděte obě formy MNDF i MNKF funkce zadané: $f = (1), 2, 3, 5, 6, 7$

1. **MNDF forma** – vyplníme logické 1 a neurčitý stav – realizace NAND

		a		b
		<hr/>		<hr/>
c	0	X	1	1
	1			
	4	1	1	1
	5			

- **červená smyčka** – spojíme čtyři logické 1 v políčkách: 2, 3, 6, 7, protože jsou sousední – vyjádříme smyčku pomocí proměnných = b (obě proměnné a i c se mění)
- **modrá smyčka** – spojíme neurčitý stav a tři logické 1 v políčkách: (1), 3, 5, 7 vyjádříme smyčku pomocí proměnných = a (obě proměnné b i c se mění)
- obě smyčky se překrývají – nevadí
- použili jsme neurčitý stav – v tomto případě nám pomohl vytvořit větší smyčku
- všechny logické jedničky v mapě jsou pokryty – můžeme zapsat výsledek
- výsledná MNDF forma: $f = b \vee a$
- pro realizaci hradly NAND upravíme její vyjádření (viz předchozí příklady):

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

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

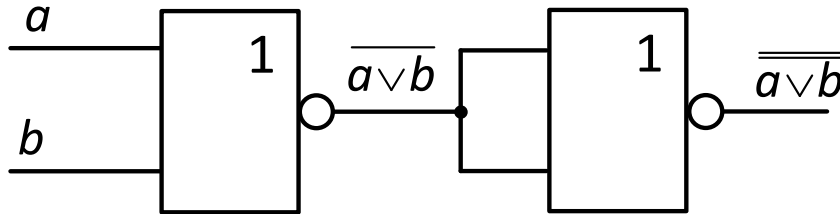
Minimalizace logických funkcí, realizace – minimalizace pomocí Karnaughovy mapy

Minimalizace logických funkcí pomocí Karnaughových map

příklad 2 – pokračování

- upravíme opět výsledek pro realizaci pomocí NOR: $f = a \vee b = \overline{\overline{a \vee b}}$
- prvním hradlem NOR vytvoříme výraz: $\overline{a \vee b}$
- druhým hradlem NOR pak vytvoříme jeho negaci (spojíme oba vstupy), abychom získali požadovaný výsledek:

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

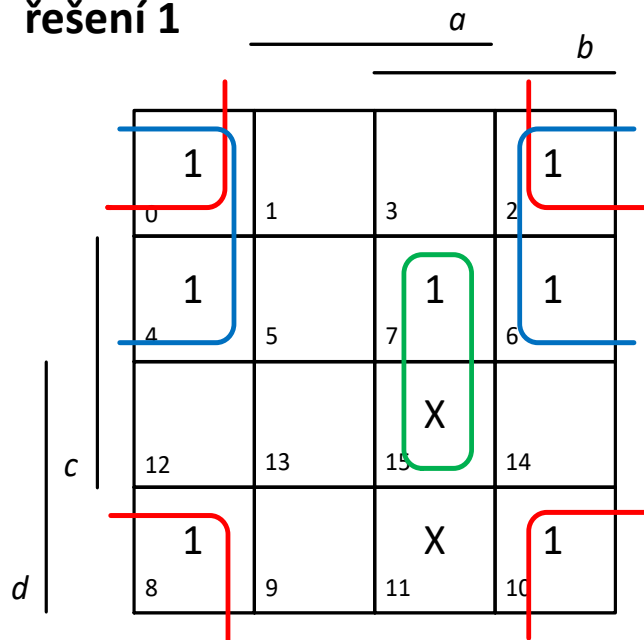


- **příklad 2** – nalezněte a realizujte MNDF a MNKF funkce f zadané indexy: $f = 0, 2, 4, 6, 7, 8, 10, (11), (15)$

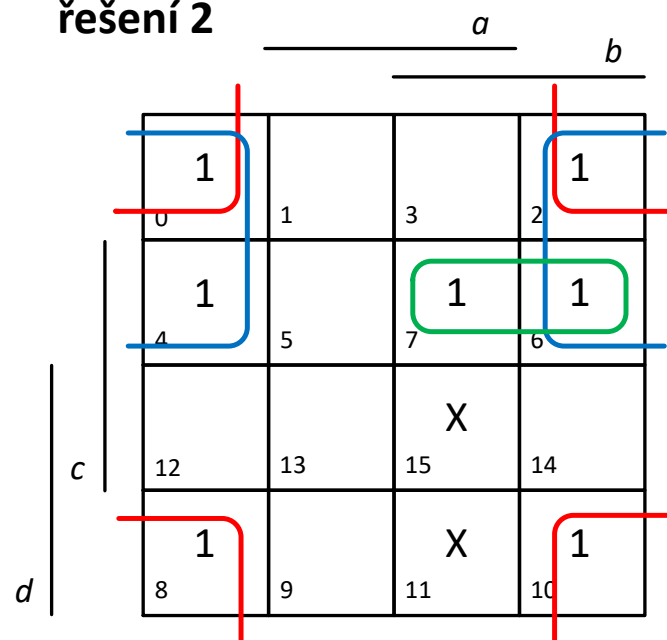
1. MNDF forma

- 2 možná (rovnocenná) řešení – obvykle preferujeme nevyužití neurčitých stavů

řešení 1



řešení 2



- **červená smyčka** – políčka s indexy 0, 2, 8, 10 jsou sousední (rohy mapy): $\bar{a}\bar{c}$
- **modrá smyčka** – políčka s indexy 0, 2, 4, 6 jsou sousední: $\bar{a}\bar{d}$
- **zelená smyčka** – řešení 1, políčka 7, (15): abc – řešení 2, políčka 6, 7: bcd
- **řešení 1:** $f = \bar{a}\bar{c} \vee \bar{a}\bar{d} \vee abc$
- **řešení 2:** $f = \bar{a}\bar{c} \vee \bar{a}\bar{d} \vee bcd$

Minimalizace logických funkcí, realizace – minimalizace pomocí Karnaughovy mapy

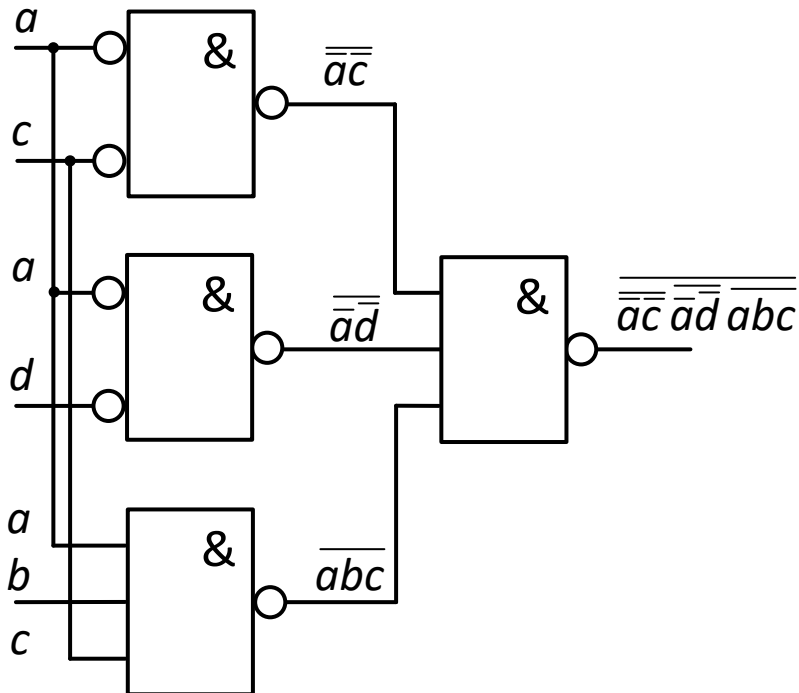
příklad 2 – realizace obou řešení MNDF

- upravíme opět nejprve výsledky pro realizaci pomocí hradel NAND

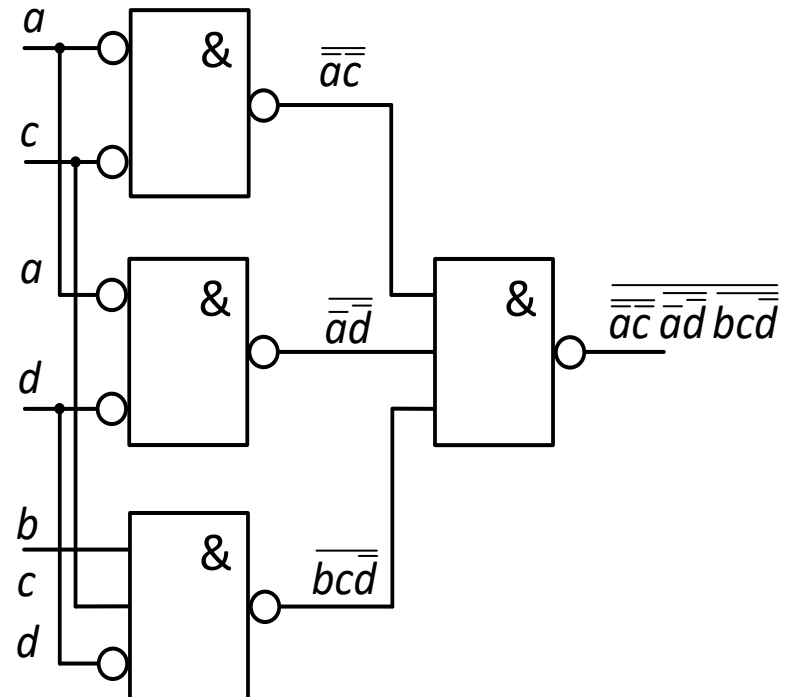
- **řešení 1:** $f = \bar{a}\bar{c} \vee \bar{a}\bar{d} \vee abc = \overline{\overline{\bar{a}\bar{c}} \vee \overline{\bar{a}\bar{d}} \vee \overline{abc}} = \overline{\overline{\bar{a}\bar{c}} \cdot \overline{\bar{a}\bar{d}} \cdot \overline{abc}}$

- **řešení 2:** $f = \bar{a}\bar{c} \vee \bar{a}\bar{d} \vee bcd = \overline{\overline{\bar{a}\bar{c}} \vee \overline{\bar{a}\bar{d}} \vee \overline{bcd}} = \overline{\overline{\bar{a}\bar{c}} \cdot \overline{\bar{a}\bar{d}} \cdot \overline{bcd}}$

▪ řešení 1

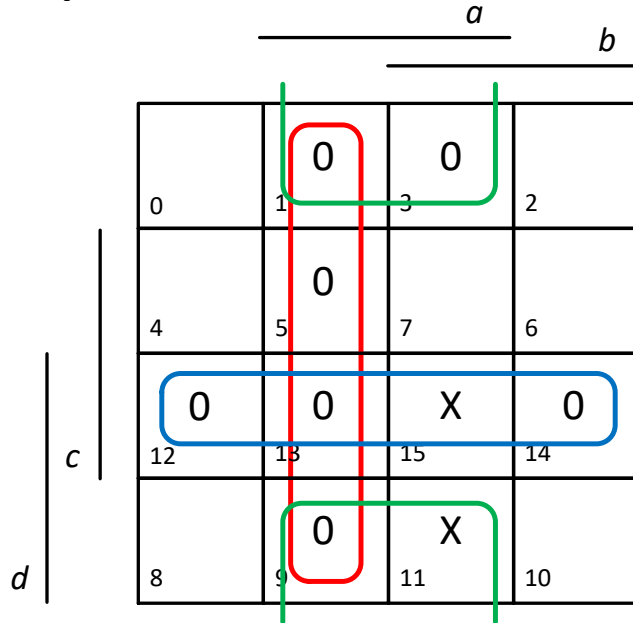


▪ řešení 2



Minimalizace logických funkcí, realizace – minimalizace pomocí Karnaughovy mapy

2. příklad 2 – MNKF

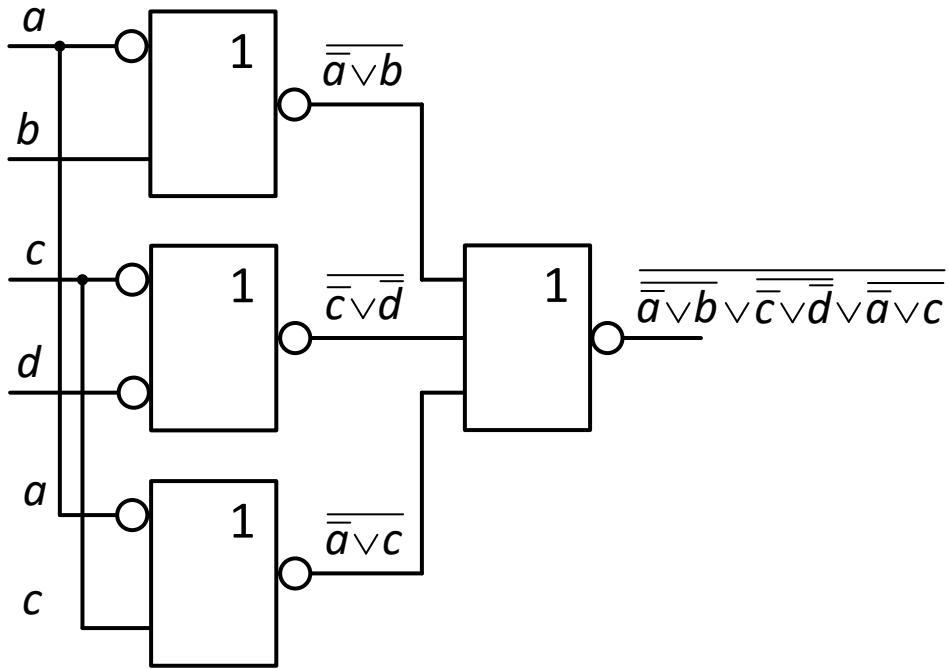


- **červená smyčka** – políčka s indexy 1, 5, 9, 13: $a\bar{b}$
- **modrá smyčka** – políčka s indexy 12, 13, 14, (15): cd
- **zelená smyčka** – políčka s indexy 1, 3, 9, (11): $a\bar{c}$
- výsledná MNKF forma: $f = \overline{a\bar{b} \vee cd \vee a\bar{c}} = (\bar{a} \vee b) \cdot (\bar{c} \vee \bar{d}) \cdot (\bar{a} \vee c)$
- opět upravíme pro potřeby realizace pomocí hradel NOR:

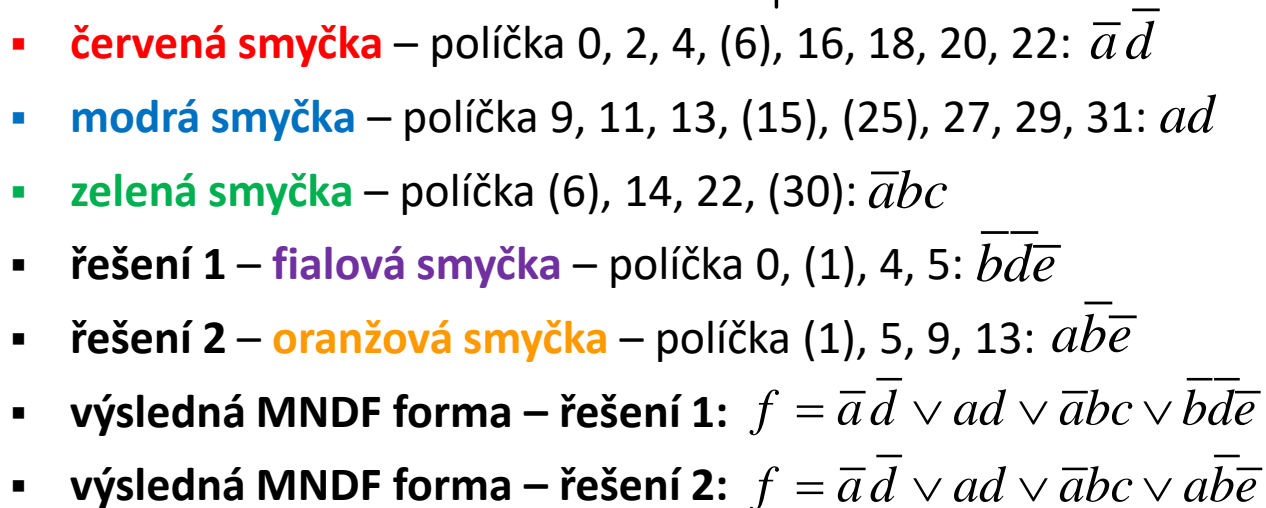
$$f = \overline{(\bar{a} \vee b) \cdot (\bar{c} \vee \bar{d}) \cdot (\bar{a} \vee c)} = \overline{(\bar{a} \vee b)} \vee \overline{(\bar{c} \vee \bar{d})} \vee \overline{(\bar{a} \vee c)}$$

Minimalizace logických funkcí, realizace – minimalizace pomocí Karnaughovy mapy

příklad 2 – realizace MNKF pomocí hradel NOR



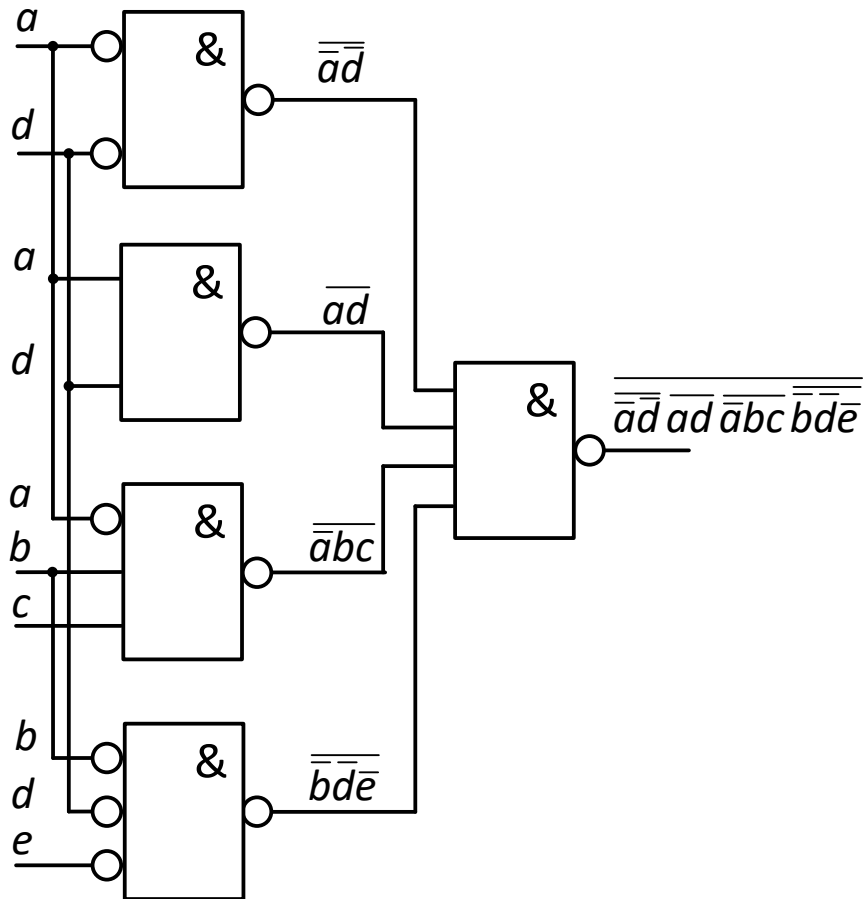
1. MNDF forma – 2 možná řešení



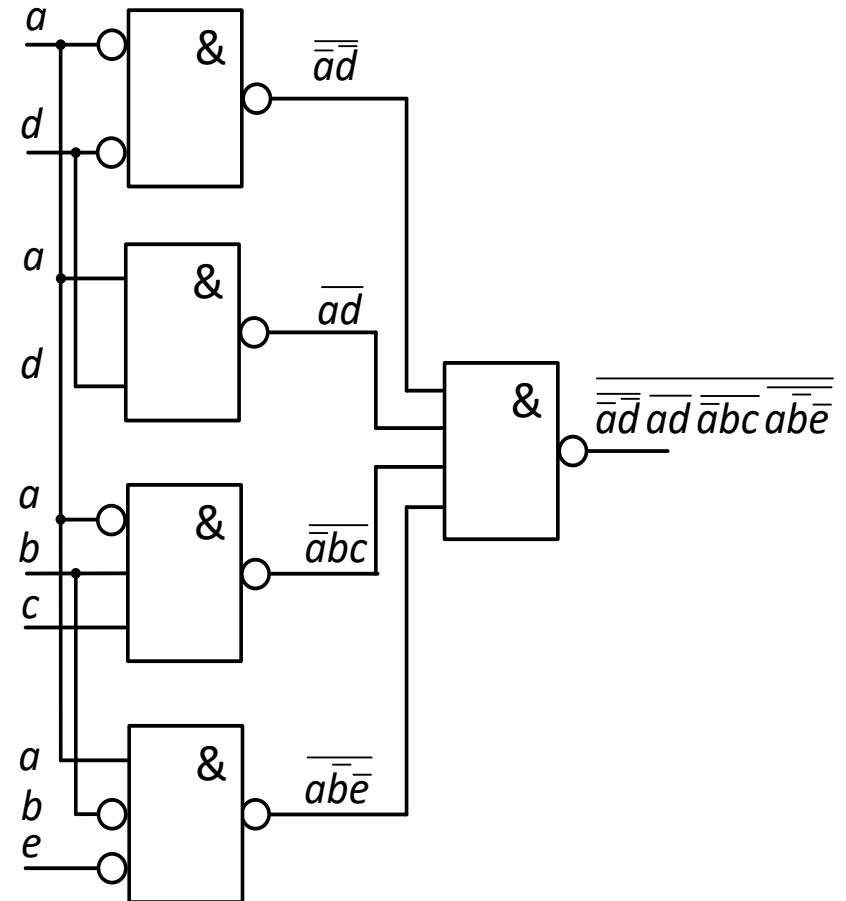
Minimalizace logických funkcí, realizace – minimalizace pomocí Karnaughovy mapy

příklad 3 – realizace obou řešení MNDF

řešení 1

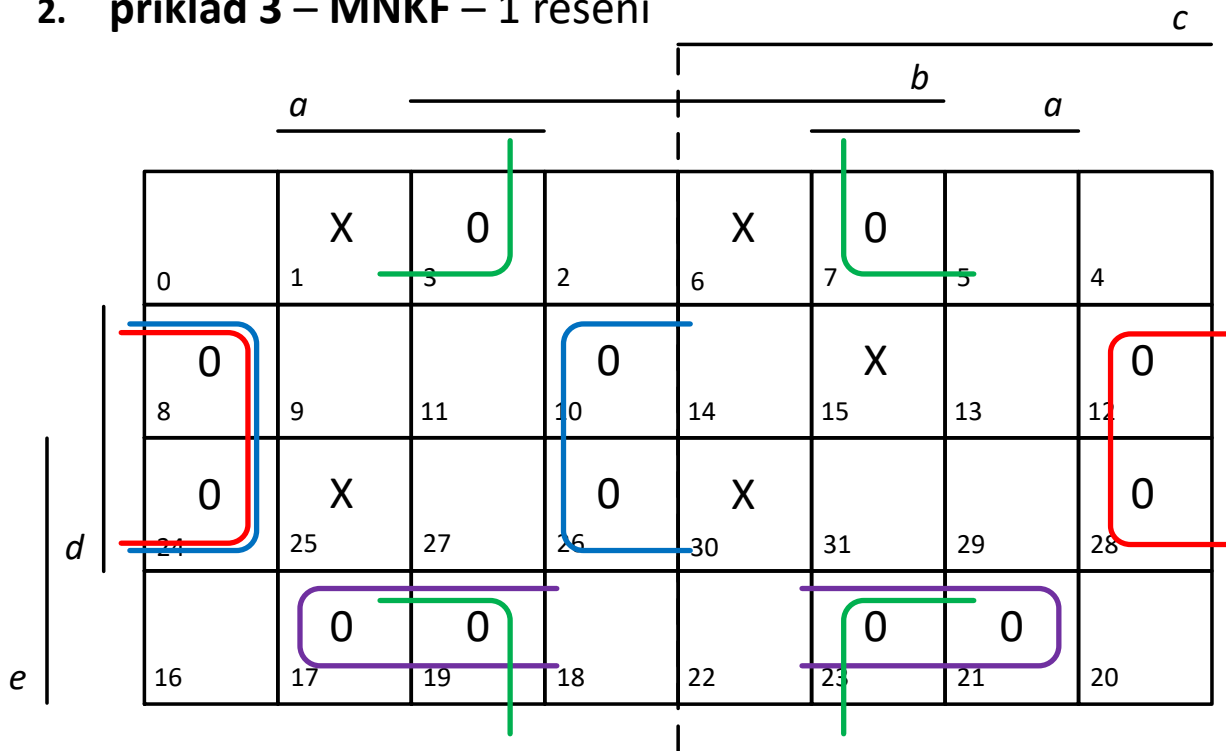


řešení 2



Minimalizace logických funkcí, realizace – minimalizace pomocí Karnaughovy mapy

2. příklad 3 – MNKF – 1 řešení

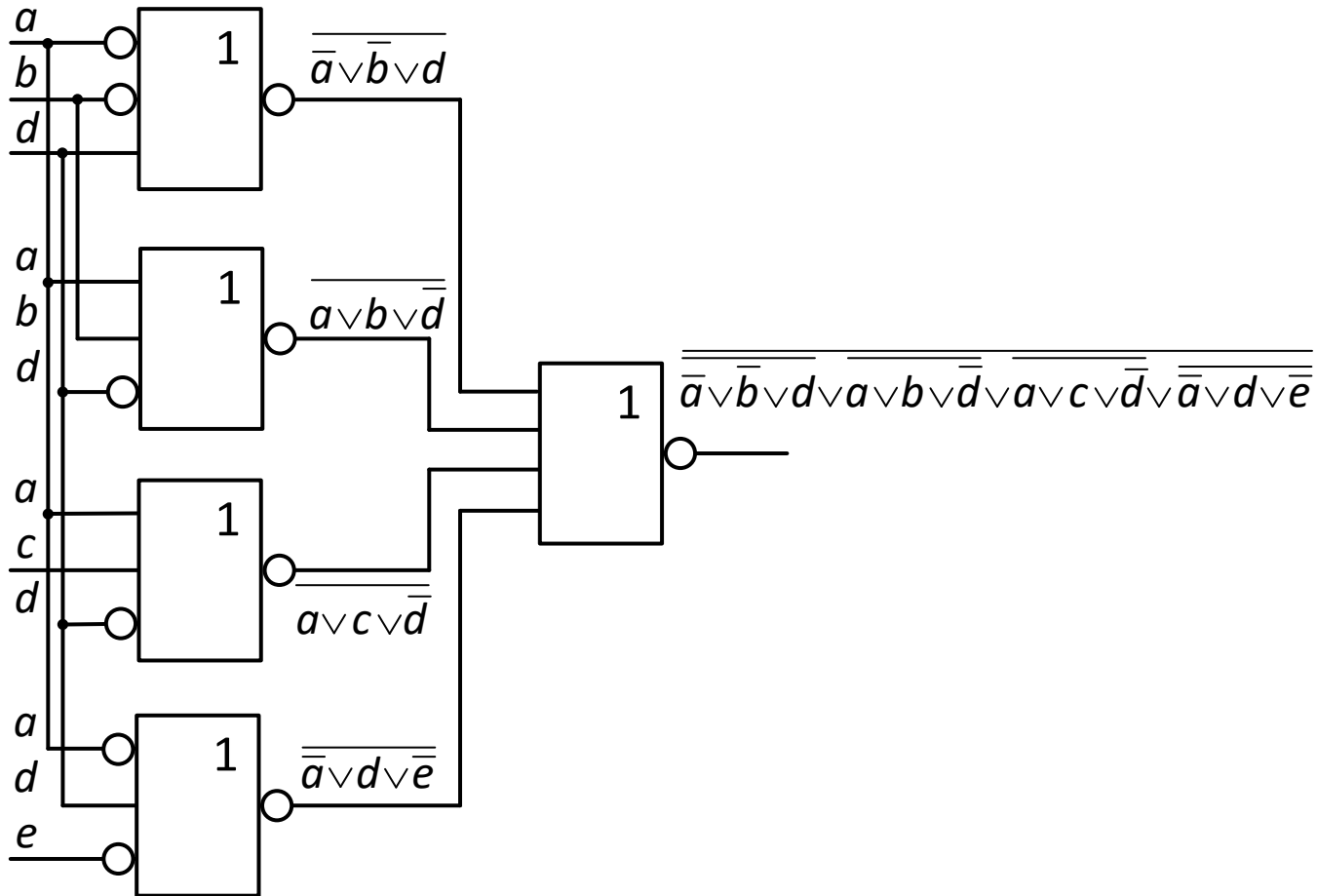


- **červená smyčka** – políčka 8, 12, 24, 28: $\bar{a}\bar{b}\bar{d}$
- **modrá smyčka** – políčka 8, 10, 24, 26: $\bar{a}\bar{c}\bar{d}$
- **zelená smyčka** – políčka 3, 7, 19, 23: $ab\bar{d}$
- **fialová smyčka** – políčka 17, 19, 21, 23: $a\bar{d}e$
- **výsledná MNKF forma:**

$$f = \bar{a}\bar{b}\bar{d} \vee \bar{a}\bar{c}\bar{d} \vee ab\bar{d} \vee a\bar{d}e = (a \vee b \vee \bar{d}) \cdot (a \vee c \vee \bar{d}) \cdot (\bar{a} \vee \bar{b} \vee d) \cdot (\bar{a} \vee d \vee \bar{e})$$

Minimalizace logických funkcí, realizace – minimalizace pomocí Karnaughovy mapy

příklad 3 – realizace MNKF



Minimalizace logických funkcí, realizace – minimalizace pomocí Karnaughovy mapy

▪ Minimalizace logických funkcí pomocí Karnaughových map

- Karnaughova mapa pro **5 proměnných** – **osa symetrie** – 2 symetrické poloviny
- každé políčko v mapě má vždy **n sousedních políček** kde n je počet proměnných – v mapě pro **5 proměnných** má tedy každé políčko **5 sousedních** – **4 ve stejné polovině** mapy a **5. políčko osově symetricky umístěné ve druhé polovině mapy**

▪ Použití Karnaughovy mapy pro převod disjunktční a konjunktční formy funkce

- převod disjunktční z/do konjunktční formy – algebraická metoda, předchází přednáška
- pomocí Karnaughovy mapy je však obvykle převod jednodušší a rychlejší

▪ **postup:**

1. ze zadané formy funkce postupně vyplníme do mapy políčka odpovídající jednotlivým smyčkám (implikantům), ze kterých zadaný tvar funkce vznikl:
 - v případě převodu **disjunktční forma → konjunktční forma** vyplňujeme do mapy **logické 1** přímo, **každému implikantu odpovídá daná smyčka logických 1**
 - **konjunktční → disjunktční forma** vyplňujeme do mapy **logické 0**, nejprve ale **znegujeme** zadaný konjunktční tvar, upravíme a **každý implikant zapíšeme smyčkou logických 0**
 - **neurčité stavy neuvažujeme** – pouze logické 1 a 0
2. nakonec doplníme **opačnou hodnotu do všech zbylých políček**, vytvoříme **smyčky z těchto políček mapy** a postupujeme stejně jako při procesu minimalizace
 - postup je v zásadě opačný procesu minimalizace

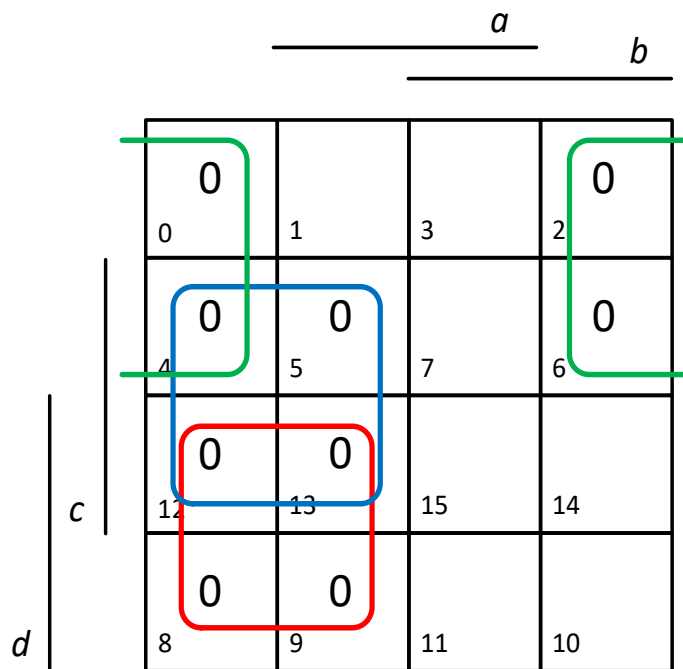
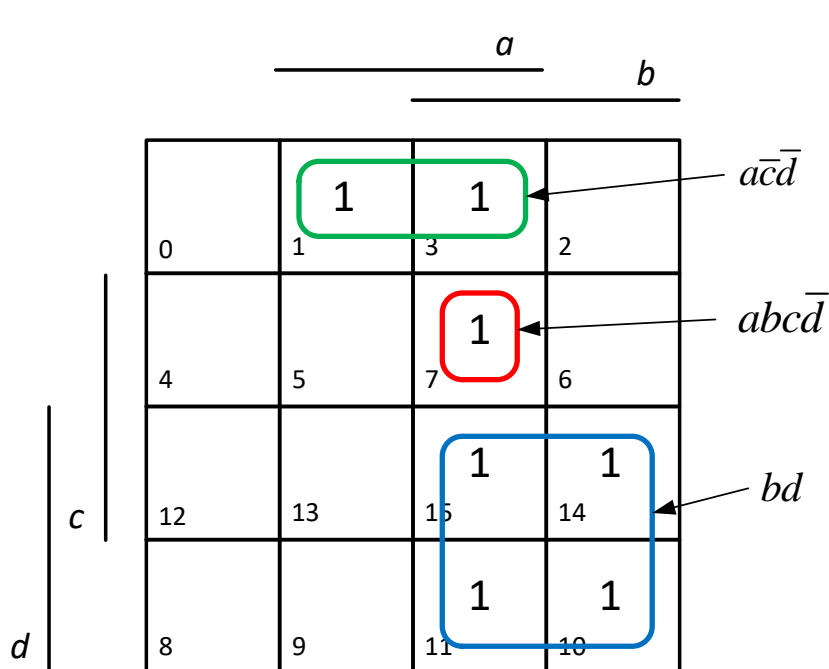
Minimalizace logických funkcí, realizace – převod formy pomocí Karnaughovy mapy

▪ Převod disjunktční forma → konjunktční forma

- **příklad 1** – převedte disjunktční tvar funkce f na MNKF:

$$f = abcd\bar{d} \vee a\bar{c}\bar{d} \vee bd$$

1. f obsahuje 4 proměnné (a, b, c, d) – použijeme Karnaughovu mapu pro 4 proměnné
2. vyplníme logické 1 do smyček dle implikantů funkce f
3. doplníme logické 0 na zbylá políčka, vytvoříme smyčky a výslednou MNKF



$$f = \overline{\bar{b}\bar{d} \vee \bar{b}\bar{c} \vee \bar{a}\bar{d}} = (b \vee \bar{d}) \cdot (b \vee \bar{c}) \cdot (a \vee d)$$

Minimalizace logických funkcí, realizace – převod formy pomocí Karnaughovy mapy

▪ Převod konjunktvní forma → disjunktvní forma

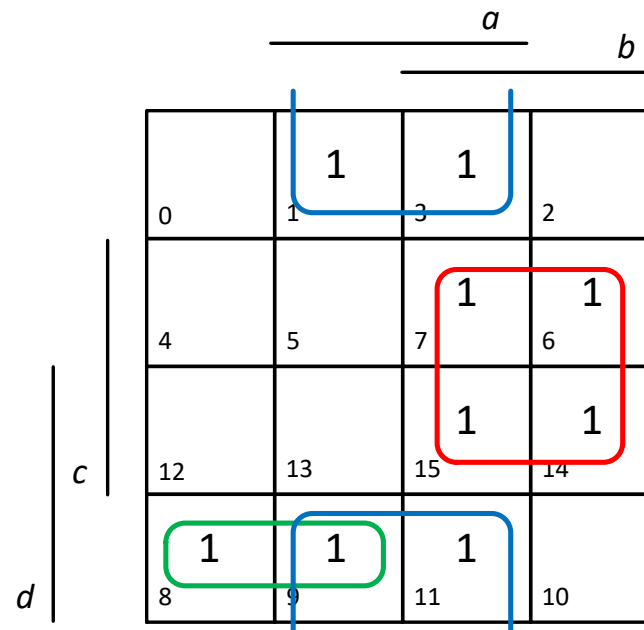
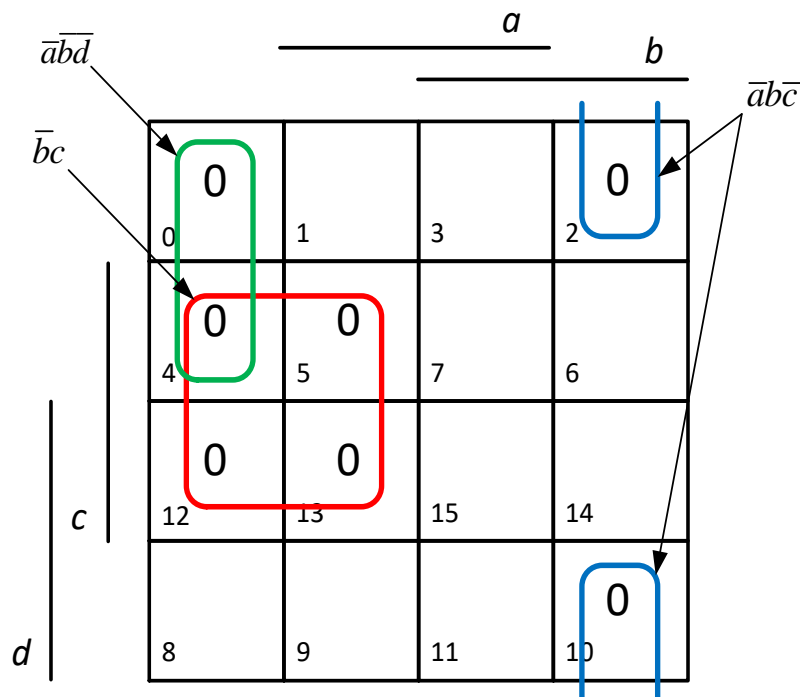
- **příklad 2** – převedte konjunktvní tvar funkce f na MNDF:

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

1. f obsahuje 4 proměnné (a, b, c, d) – použijeme Karnaughovu mapu pro 4 proměnné
2. znegujeme konjunktvní tvar a upravíme negaci (opačná funkce k funkci f):

$$\bar{f} = \overline{(a \vee \bar{b} \vee c) \cdot (a \vee b \vee d) \cdot (b \vee \bar{c})} = \bar{a}b\bar{c} \vee \bar{a}\bar{b}d \vee \bar{b}c$$

3. vyplníme logické 0 do mapy dle implikantů
4. vyplníme zbylá políčka logickými 1, vytvoříme z nich smyčky a výslednou MNDF

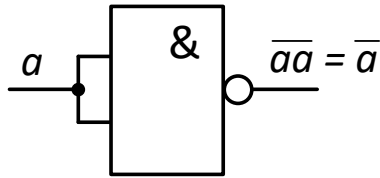
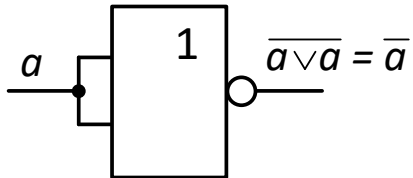
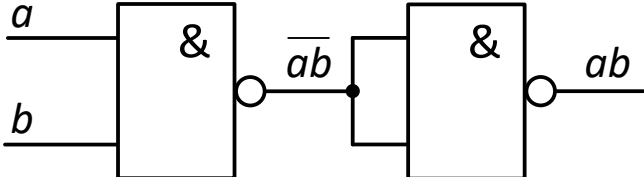
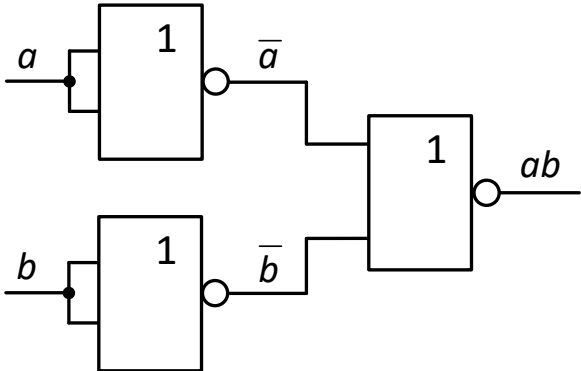
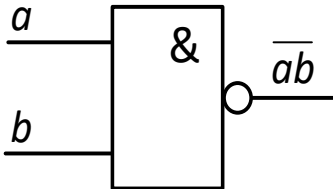
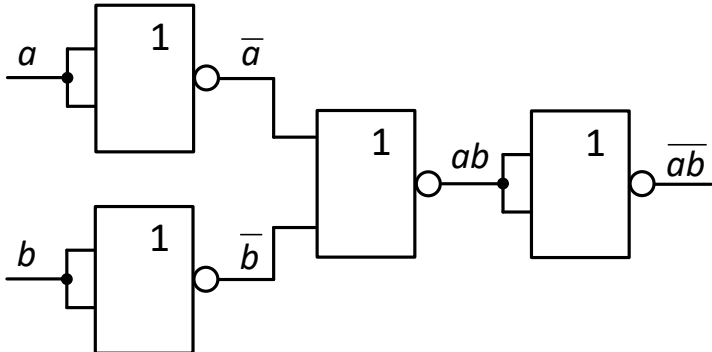
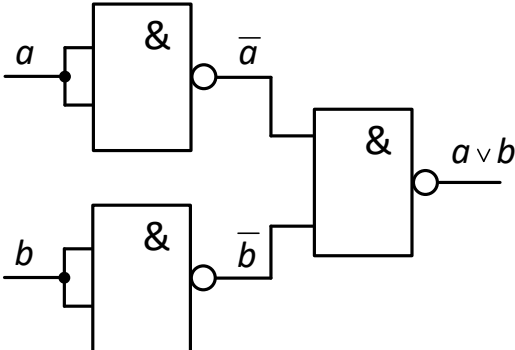
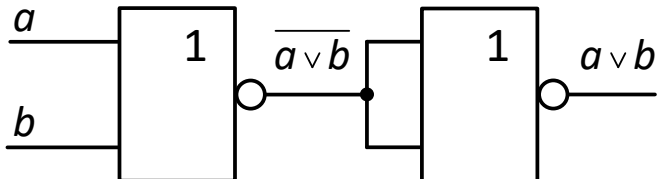


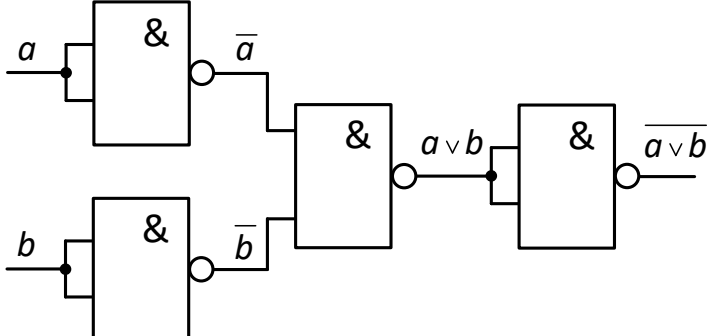
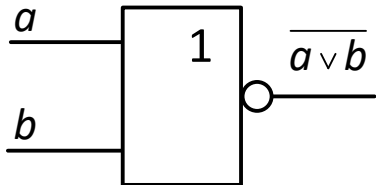
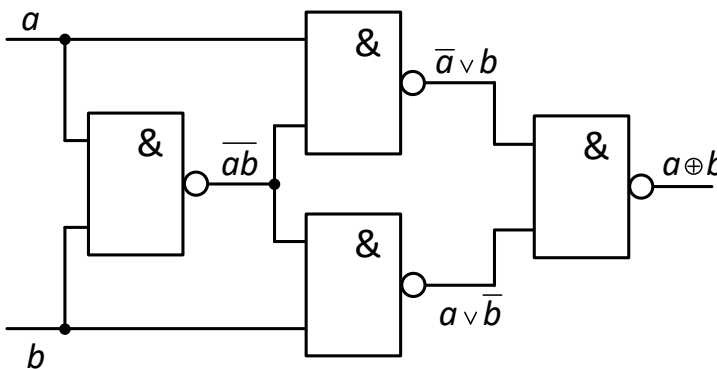
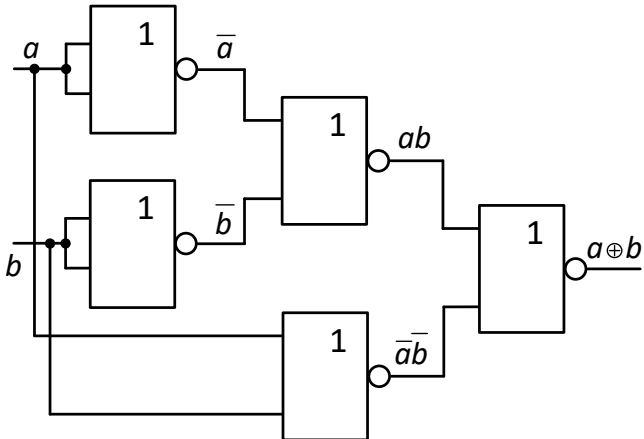
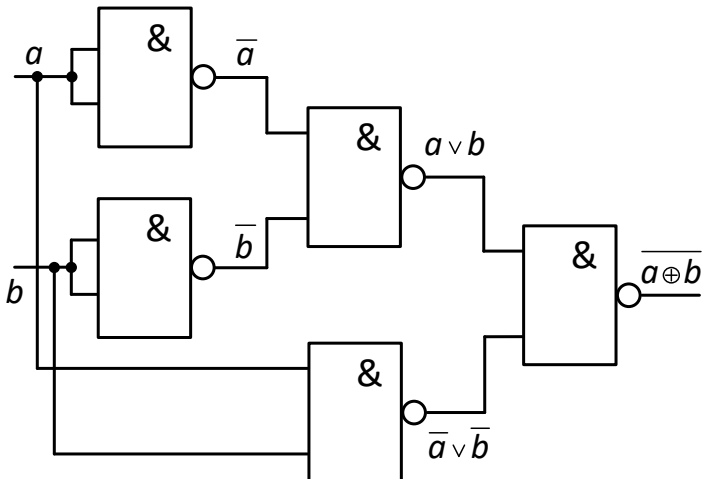
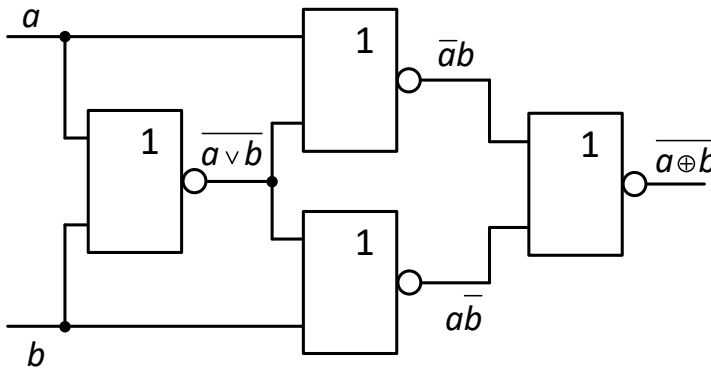
$$f = bc \vee a\bar{c} \vee \bar{b}\bar{c}d$$

Kombinační logické obvody – realizace funkcí pomocí hradel

▪ Realizace funkcí pomocí hradel

- **logický obvod** – obvod realizující danou logickou funkci
- **díky minimalizaci tvaru funkce** – menší počet použitých hradel, použití jednoho typu hradla, snížení zpoždění obvodu, snížení spotřeby obvodu, menší plocha obvodu...
- programovatelné logické obvody (FPGA, CPLD)
 - **v jazyce VHDL** definovány všechny elementární funkce – **NOT, AND, OR, NAND, NOR, XOR, XNOR** – úpravy a minimalizace forem funkce můžeme přenechat syntezátoru
 - můžeme využít hradla s různým počtem vstupů (v některých aplikacích kritické)
 - možnost pomocí parametrů ovlivňovat proces syntézy (důraz na zpoždění, důraz na počet použitých hradel, důraz na spotřebu...)
- diskrétní logická hradla
 - máme k dispozici obvykle jen omezený soubor hradel (např. jen NAND, NOR...)
 - máme k dispozici hradla s daným počtem vstupů (nejčastěji dvouvstupá, třívstupá)
 - musíme vzít v potaz fyzikální parametry a vlastnosti dané technologie realizace hradel (zpoždění, spotřeba, pracovní napětí, napěťové úrovně, logický zisk, šumová imunita...)
 - v praxi tak musíme často vhodně upravit realizaci zadané logické funkce!

Logická funkce	realizace pomocí NAND	realizace pomocí NOR
Negace, NOT		
Logický součin, AND		
Negovaný logický součin, NAND		
Logický součet, OR		

Logická funkce	realizace pomocí NAND	realizace pomocí NOR
Negovaný logický součet, NOR		
Exkluzivní součet (neekvivalence), XOR		
Negovaný exkluzivní součet (ekvivalence), XNOR		

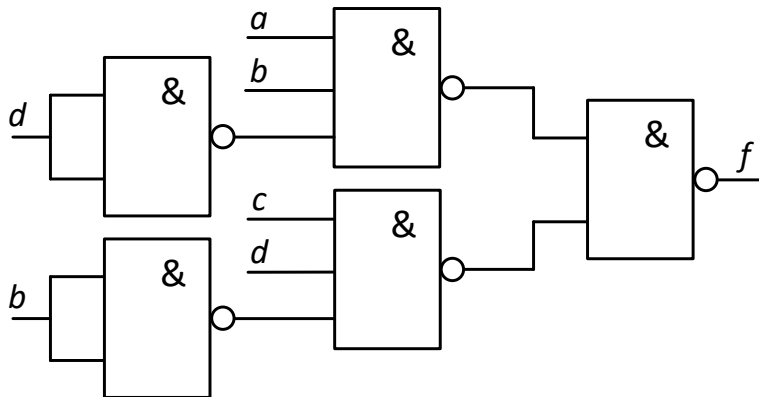
Kombinační logické obvody – realizace funkcí pomocí hradel

▪ Použití logických hradel s n -vstupy

- nejčastější (nejdostupnější) v praxi – dvouvstupá a třívstupá hradla NAND a NOR
- 2 situace – k dispozici hradla s méně vstupy, k dispozici hradla s více vstupy

a) k dispozici jsou hradla s menším počtem vstupů než je potřeba

- např. máme k dispozici dvouvstupá hradla a je potřeba realizovat minterm se 3 vstupními proměnnými
- použijeme zákon dvojité negace a De Morganovy zákony – rozdělujeme termy tak, abychom získali více termů s nižším počtem proměnných
- grafická pomůcka – Rottovy mřížky (aplikace výše uvedených zákonů)
- **příklad** – realizujte zadanou funkci f použitím pouze dvouvstupých hradel NAND
 $f = ab\bar{d} \vee \bar{b}cd$
- pokud bychom nemuseli dodržovat žádné omezení – 2x 3vstupé a 3x 2vstupé hradlo

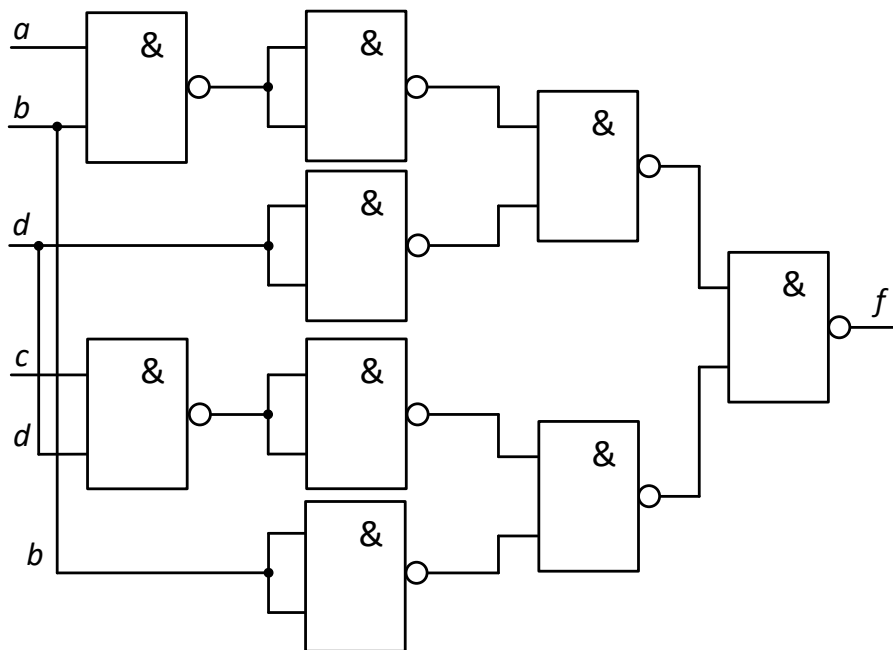


$$f = ab\bar{d} \vee \bar{b}cd = \overline{\overline{ab\bar{d}} \cdot \overline{\bar{b}cd}}$$

Kombinační logické obvody – realizace funkcí pomocí hradel

- pokud máme jen 2vstupá hradla – dvakrát negujeme a rozdělujeme na dvojice

$$f = abd\bar{d} \vee \bar{b}cd = \overline{\overline{abd\bar{d}}} \cdot \overline{\overline{\bar{b}cd}} = \overline{\overline{abd\bar{d}}} \cdot \overline{\overline{\bar{b}cd}} = \overline{\overline{ab} \vee \overline{\bar{d}}} \cdot \overline{\overline{\bar{b}} \vee \overline{cd}} = \overline{\overline{ab} \cdot \overline{\bar{d}}} \cdot \overline{\overline{\bar{b}} \cdot \overline{cd}} = \overline{\overline{ab} \cdot \overline{\bar{d}}} \cdot \overline{\overline{\bar{b}} \cdot \overline{cd}} = \overline{\overline{ab} \cdot \overline{\bar{d}}} \cdot \overline{\overline{\bar{b}} \cdot \overline{cd}}$$

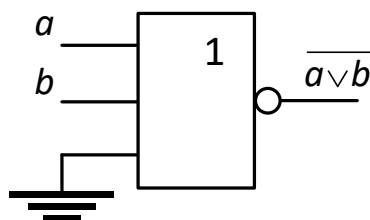
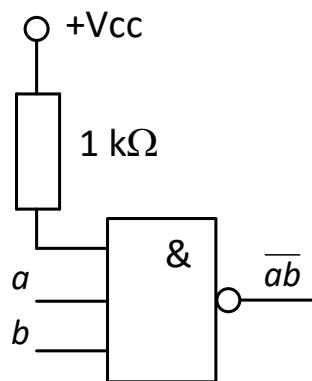


9x 2vstupé hradlo

Kombinační logické obvody – realizace funkcí pomocí hradel

b) k dispozici jsou hradla s větším počtem vstupů než je potřeba

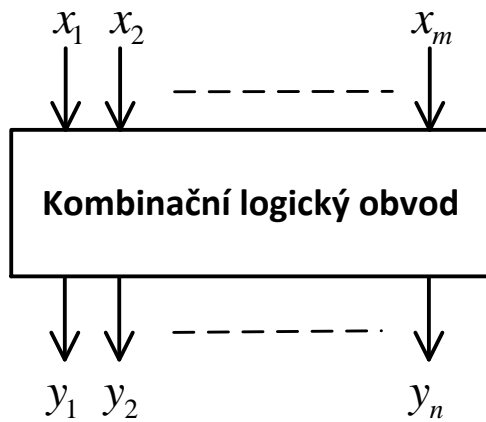
- jak ošetřit nepřipojené vstupy hradel? a proč vlastně?
- **proč?** – skrz nezapojený vstup může do hradla pronikat nežádoucí rušící signál → neočekávané (nechtěné) změny stavů na výstupu hradla → rušení v obvodu
- **jak zapojit nevyužité vstupy hradel?**
- záleží na typu hradla, neboť:
 - AND, NAND: zákon o neutrálnosti jedničky – $a \cdot 1 = a$
 - OR, NOR: zákon o neutrálnosti nuly – $a \vee 0 = a$
- součinnová hradla připojujeme přes odpor (1 k Ω) ke **zdroji napájení +Vcc** – logická 1
- součtová hradla připojujeme **na zem** – logická 0



- a co spojit vstupy navzájem? – dle zákona o idempotenci platí $a \vee a = a$, $a \cdot a = a$
- **nevýhoda** – zvyšujeme **výkonové zatížení výstupu předchozího hradla (logický zisk)**
- u součinnových hradel AND, NAND – snížení logického zisku jen o 1
- u součtových hradel OR, NOR – snížení logického zisku o počet spojených vstupů

Kombinační vs. sekvenční logické obvody – definice, příklady kombinačních obvodů

▪ Kombinační logický obvod (KLO)



$$y_1 = f_1(x_1, x_2, \dots, x_m)$$

$$y_2 = f_2(x_1, x_2, \dots, x_m)$$

$$\vdots$$

$$y_n = f_n(x_1, x_2, \dots, x_m)$$

$$\mathbf{Y} = \mathbf{F}(\mathbf{X})$$

x_1, x_2, \dots, x_m – vstupní hodnoty

množina vstupních stavů – \mathbf{X}

y_1, y_2, \dots, y_n – výstupní hodnoty

množina výstupních stavů – \mathbf{Y}

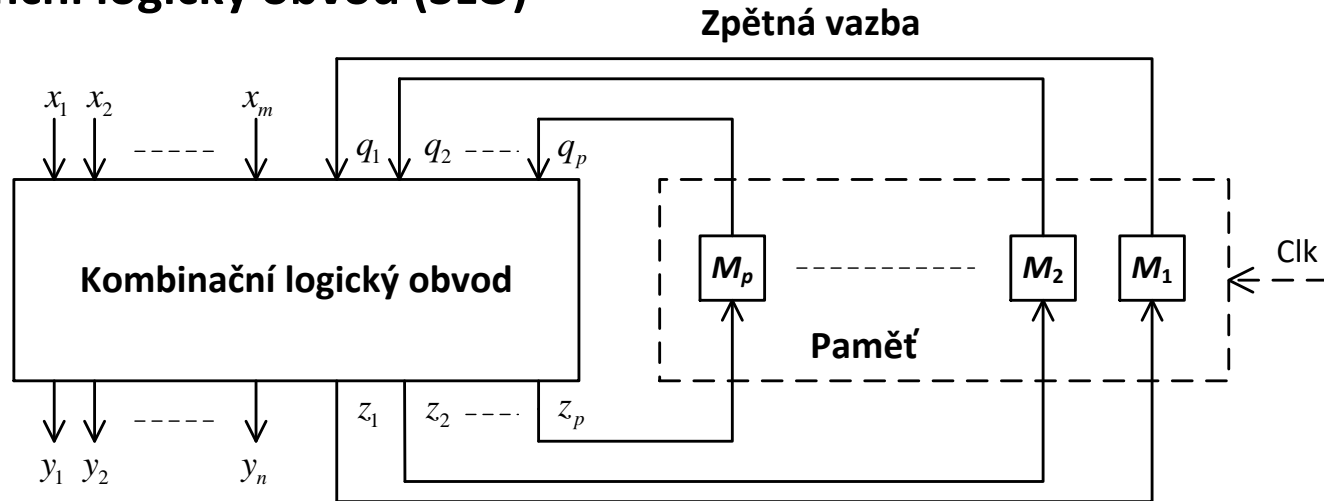
f_1, f_2, \dots, f_n – výstupní funkce

množina výstupních funkcí – \mathbf{F}

- obecně logický obvod složený z log. hradel a dalších elementárních prvků, jehož **výstupy y_1, y_2, \dots, y_n jsou jednoznačně určeny kombinací vstupních hodnot x_1, x_2, \dots, x_m**
- **obvod neobsahuje žádný paměťový člen a žádnou zpětnou vazbu!**
- **výstup obvodu y** – je v každém okamžiku jednoznačně určen pouze vstupní kombinací!
- **pro m vstupů a n výstupů** – můžeme sestavit pro popis obvodu 2^m Booleových rovnic
- **ideální kombinační logický obvod** – jakákoliv změna vstupní hodnoty vyvolá okamžitě změnu výstupní hodnoty obvodu – do další změny na vstupu je pak výstup konstantní
- **reálný (nedokonalý) kombinační logický obvod** – vlivem doby zpoždění jednotlivých hradel obvodu se změna vstupu KLO projeví na jeho výstupu se zpožděním
- **doba zpoždění** = časová prodleva mezi okamžikem změny na vstupu do nastavení správného výstupu obvodu (viz dále doba zpoždění hradel)
- nenulová doba zpoždění logických hradel – **vznik logických hazardů v obvodech**

Kombinační vs. sekvenční logické obvody – definice, příklady kombinačních obvodů

▪ Sekvenční logický obvod (SLO)



- sekvenční logické obvody:
- **asynchronní** – bez použití hodinového (taktovacího) signálu Clk, změny na vstupech obvodu jsou vyhodnocovány okamžitě, bezprostřední změny výstupních a vnitřních stavů obvodu
- **synchronní** – použití hodinového (taktovacího) signálu Clk pro výstup obvodu, změny vnitřních stavů obvodu jsou prováděny synchronně v době mezi dvěma danými okamžiky -> hladinové obvody – tzv. latch, impulzní (hranové) obvody – tzv. flip-flop
- **sekvenční logický obvod (SLO) – 3 části: kombinační (KLO), paměť, zpětná vazba**
- **výstup obvodu y** – obecně závisí jak na **současné hodnotě vstupů x**, tak i na **hodnotě vnitřních proměnných q** (respektive předchozí hodnotě vstupů)
 - více se k sekvenčním logickým obvodům vrátíme na konci přednášky