2. BOOLEOVA ALGEBRA

Booleova algebra je zvláštní matematický způsob popisu chování i struktury logických obvodů. Tato algebra pracuje s log. proměnnými, které mohou nabývat pouze dvou hodnot, log. Hodnoty '**0**' nebo log. hodnoty '**1**'.

2.1. Výroková logika a logické funkce

Logický obvod je tvořen skupinou vzájemně spojených logických členů, o kterých lze říci, že jsou zařízení uskutečňující log. funkci. S log. funkcemi se můžeme setkat ve výrokové logice, která je součástí matematické logiky.

Základním pojmem výrokové logiky je výrok, což je každé tvrzení, o kterém má smysl prohlásit, zda je pravdivé nebo nepravdivé.

Logická funkce je předpis, který kombinaci, popř. i sledu hodnot jedné nebo více (nezávislých) log. proměnných jednoznačně přiřazuje hodnoty jedné (závislé) log. proměnné.

Negace je takovou funkcí jedné proměnné, u které má závisle proměnná vždy opačnou hodnotu, než nezávisle proměnná. Algebraické vyjádření této funkce je: $A = \overline{B}$

Disjunkce nebo-li ***logický součet*** je takovou funkcí dvou proměnných A, B, že závisle proměnná Y nabývá hodnoty 1 tehdy, je-li A nebo B nebo A i B současně rovno 1.

Algebraické vyjádření této funkce je: $Y = A + B$

(v praxi - např. obvod paralelního zapojení dvou spínačů k žárovce)

Konjunkce nebo-li ***logický součin*** je takovou funkcí dvou proměnných A, B, že závisle proměnná Y nabývá hodnoty 1 pouze tehdy, mají-li současně A i B hodnotu 1. V ostatních případech nabývá proměnná Y hodnoty 0. Algebraické vyjádření této funkce je: $Y = A \cdot B$

(v praxi – např. obvod sériového zapojení dvou spínačů k žárovce)

Shefferova funkce (NAND) je takovou funkcí dvou proměnných A, B, že závisle proměnná Y nabývá hodnoty 0 pouze tehdy, mají-li současně A i B hodnotu 1. V ostatních případech nabývá proměnná Y hodnoty 1. Algebraické vyjádření této funkce je: $Y = \overline{A \cdot B}$

Pierceova funkce **(NOR)** je takovou funkcí dvou proměnných A, B, že závisle proměnná Y nabývá hodnoty 1 pouze tehdy, mají-li současně A i B hodnotu 0. V ostatních případech nabývá proměnná Y hodnoty 0. Algebraické vyjádření této funkce je: $Y = \overline{A + B}$

Přehled základních kombinačních logických funkcí je uveden v tabulce 2.

2.2. Základní pravidla a zákony Booleovy algebry.

Booleova algebra je dvouhodnotová logická algebra, používající log. součtu, součinu a negace jako úplného systému základních logických funkcí. Používá se k úpravě a zjednodušování (minimalizaci) logických funkcí. Obsahuje následující zákony a pravidla:

1. Pravidlo agresívnosti a neutrálnosti hodnot 0 a 1

$X + 1 = 1$ $\qquad$ $X + 0 = X$

$X \cdot 0 = 0$ $\qquad$ $X \cdot 1 = X$

2. Pravidlo komutativní

$X + Y = Y + X$ $\qquad$ $X \cdot Y = Y \cdot X$

3. Pravidlo asociativní

$X + (Y + Z) = (X + Y) + Z = X + Y + Z$

$X \cdot (Y \cdot Z) = (X \cdot Y) \cdot Z = X \cdot Y \cdot Z$

4. Pravidlo distributivní

$X + (Y \cdot Z) = (X + Y) \cdot (X + Z)$ $\qquad$ $X \cdot (Y + Z) = X \cdot Y + X \cdot Z$

5. Pravidlo absorpce

$X + X = X$ $\qquad$ $X \cdot X = X$

$X + X \cdot Y = X$ $\qquad$ $X \cdot (X + Y) = X$

6. Pravidlo absorpce negace

$X + \overline{X} \cdot Y = X + Y$ $\qquad$ $X \cdot (\overline{X} + Y) = X \cdot Y$

$\overline{X} + X \cdot Y = \overline{X} + Y$ $\qquad$ $\overline{X} \cdot (X + Y) = \overline{X} \cdot Y$

7. Pravidlo o vyloučeném třetím

$X + \overline{X} = 1$ $\qquad$ $X \cdot \overline{X} = 0$

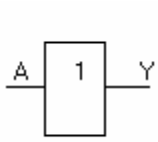
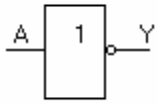
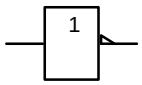
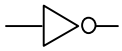
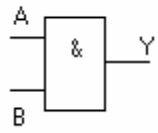
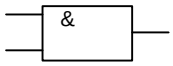
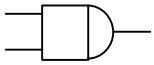
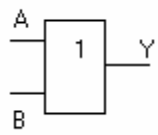
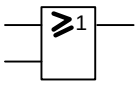
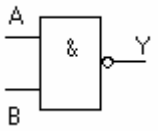
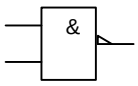
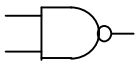
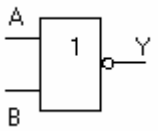
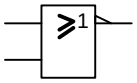
8. Pravidlo dvojité negace

$\overline{\overline{X}} = X$ $\qquad$ $\overline{\overline{X + Y}} = X + Y$ $\qquad$ $\overline{\overline{X \cdot Y}} = X \cdot Y$

9. Pravidla o vytvoření negace - De Morganovy zákony

$\overline{X + Y} = \overline{X} \cdot \overline{Y}$ $\qquad$ $\overline{X \cdot Y} = \overline{X} + \overline{Y}$

Tab. 2: Přehled kombinačních logických funkcí.

Logický člen	Logická funkce	Pravdivostní tabulka	Schematická značka	Schematická značka DIN (SW - Multisim)	Schematická značka ANSI (SW - Multisim)
Opakovač	$Y = A$	A | Y 0 | **0** 1 | **1**			
Negace NOT	$Y = \overline{A}$	A | Y 0 | **1** 1 | **0**			
Konjunkce AND	$Y = A \cdot B$	A B | Y 0 0 | **0** 0 1 | **0** 1 0 | **0** 1 1 | **1**			
Disjunkce OR	$Y = A + B$	A B | Y 0 0 | **0** 0 1 | **1** 1 0 | **1** 1 1 | **1**			
Shefferova funkce NAND	$Y = \overline{A \cdot B}$	A B | Y 0 0 | **1** 0 1 | **1** 1 0 | **1** 1 1 | **0**			
Pierceova funkce NOR	$Y = \overline{A + B}$	A B | Y 0 0 | **1** 0 1 | **0** 1 0 | **0** 1 1 | **0**			

2.3. Pravdivostní tabulka

Tabulka č.3:
Důkaz De-Morganova zákona tabulkovou metodou

A	B	$A+B$	$\overline{A+B}$	$\overline{A}$	$\overline{B}$	$\overline{A}\cdot\overline{B}$
0	**0**	0	1	1	1	1
0	**1**	1	0	1	0	0
1	**0**	1	0	0	1	0
1	**1**	1	0	0	0	0

Pozn.:
Pro **n** proměnných má tabulka $\mathbf{2^n}$ kombinací

Tabulka stavů je tabulka, která jednoznačně přiřazuje vstupní proměnné k jedné nebo několika výstupním proměnným. Má dvě části. Levá část zahrnuje všechny možné kombinace hodnot vstupních proměnných, pravá část obsahuje výslednou hodnotu výrazu pro každou kombinaci hodnot z levé části. Velikost levé části závisí na počtu vstupních proměnných, neboť počet řádků v tabulce je roven $\mathbf{2^n}$.

Kombinace metodou vstupních proměnných je nejlépe vyjádřit pomocí dvojkových čísel. Tabulku stavů používáme nejen pro snadné popsání činnosti log. obvodu, ale můžeme ji použít i pro zjištění, jsou-li dva výrazy shodné. Příklad pro důkaz De Morganova zákona je uveden v tabulce 3. V levé části jsou uvedeny vstupní proměnné A, B a ve výstupní části je názorně ukázán celý postup důkazu.

2.4. Minimalizace

Při návrhu log. obvodu dbáme na to, aby jej bylo možno realizovat pokud možno co nejmenším počtem co nejjednodušších logických členů. To znamená, že algebraický výraz má být složen z co nejmenšího počtu členů, z nichž každý obsahuje nejmenší počet proměnných. Postup pomocí kterého nahradíme složitější algebraický výraz jednodušším výrazem, nazýváme minimalizací.

Způsob úpravy algebraického výrazu, který využívá všech známých zákonů a pravidel Booleovy algebry se nazývá ***přímá minimalizace***. Nyní si na několika příkladech ukážeme zjednodušování výrazů.

Příklad 1:

$$Z = AB\overline{C}D + \overline{A}B\overline{C}D + ABCD + \overline{A}BCD$$
$$Z = BD(A\overline{C} + \overline{A}\overline{C} + AC + \overline{A}C)$$
$$Z = BD[\overline{C}(A + \overline{A}) + C(A + \overline{A})]$$
$$Z = BD(C + \overline{C})$$
$$Z = BD$$

Příklad 2:

$$Z = \overline{\overline{X}} + XY$$
$$Z = \overline{X}(Y + \overline{Y}) + XY$$
$$Z = \overline{X}Y + \overline{X}\overline{Y} + XY$$
$$Z = \overline{X}Y + \overline{X}Y + \overline{X}\overline{Y} + XY$$
$$Z = \overline{X}(Y + \overline{Y}) + Y(X + \overline{X})$$
$$Z = \overline{X} + Y$$

Příklad 3:

Z = (A +B)(A + C)
Z = A + AB +AC + BC
Z = A(1 + B + C) + BC
Z = A + BC

Příklad 4:

$$Z = \overline{A}BC + A\overline{B} + \overline{A} \cdot \overline{B} + ABC$$
$$Z = BC(A + \overline{A}) + \overline{B}(\overline{A} + A)$$
$$Z = BC + \overline{B} \quad \text{(pravidlo absorbce negace)}$$
$$Z = \overline{B} + C$$

2.5. Zápis funkce z pravdivostní tabulky

Co je to pravdivostní tabulka, jsme si již popsali dříve. Nyní si ukážeme, jak pomocí této tabulky budeme řešit určitý problém a jak z ní sestavíme log. funkci.

Zadání:

Sestavte log. funkci pro ovládání žárovky ze dvou míst (případ schodišťového přepínače)

Řešení: Nejprve sestavíme pravdivostní tabulku pro dvě proměnné, kde spínače budou označeny **A**, **B** a žárovka **Z**.

A	B	Z
0	0	**0**
0	1	**1**
1	0	**1**
1	1	**0**

Z tabulky zapíšeme logickou funkci:

$$Z = \overline{A} \cdot B + A \cdot \overline{B} \qquad \text{(součet součinů)}$$

nebo $$Z = (A + B) \cdot (\overline{A} + \overline{B}) \qquad \text{(součin součtů)}$$

Algebraický výraz můžeme dostat z tabulky ve dvou tvarech. A to buď jako (1) součet součinů nebo jako (2) součin součtů.

(1) Výraz ve tvaru součtu součinů dostaneme, napíšeme-li pro každé políčko, které obsahuje 1 pro výstupní proměnnou, kombinaci odpovídajících vstupních proměnných ve formě součinu. Tyto jednotlivé součiny pak spojíme v součet.

(2) Výraz ve tvaru součinu součtů dostaneme, napíšeme-li pro každé políčko, které obsahuje 0 pro výstupní proměnnou, kombinaci odpovídajících vstupních proměnných v podobě součtu. Tyto součty pak spojíme v součin.

Příklad 4: Z následující tabulky sestavte log. funkci.

A	B	C	Z
0	0	0	**0**
0	0	1	**1**
0	1	0	**0**
0	1	1	**1**
1	0	0	**1**
1	0	1	**0**
1	1	0	**0**
1	1	1	**0**

Řešení:

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

tuto funkci můžeme dále zjednodušit:

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

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

Často také musíme řešit problém opačný, a to sestavení tabulky z logické funkce.

Příklad 5:

A	B	C	Z
0	0	0	**0**
0	0	1	**1**
0	1	0	**0**
0	1	1	**0**
1	0	0	**1**
1	0	1	**0**
1	1	0	**0**
1	1	1	**1**

Z dané logické funkce sestavte tabulku stavů:

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

$$1\cdot1\cdot1 \quad + 1\cdot0\cdot0 \quad + \quad 0\cdot0\cdot1$$

Protože se jedná o součet součinů, musí se každý součin rovnat 1. Najdeme tedy odpovídající kombinace vstupních proměnných a k výstupní proměnné napíšeme 1. K ostatním kombinacím napíšeme nuly.

Příklad 6:

A	B	C	Z
0	0	0	**1**
0	0	1	**0**
0	1	0	**0**
0	1	1	**1**
1	0	0	**1**
1	0	1	**0**
1	1	0	**0**
1	1	1	**0**

Z dané logické funkce sestavte tabulku stavů:

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

Funkci můžeme také rozepsat takto:

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

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

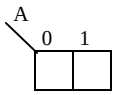
$$0\cdot1\cdot1 \quad + \quad 1\cdot0\cdot0 \quad + \quad 0\cdot0\cdot0$$

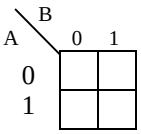
2.6. Karnaughova mapa

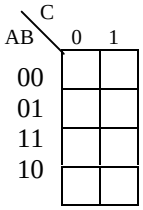
Pro zobrazení log. funkce se kromě tabulky stavů používá také Karnaughova mapa. Slouží hlavně pro minimalizaci funkce. Její nevýhodou je, že ji můžeme použít nejvýše pro čtyři proměnné. Použití pro více proměnných je již poměrně složité.

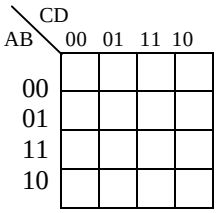
Na rozdíl od tabulky stavů se hodnoty vstupních proměnných zapisují po straně mapy a nad mapou a výstupní proměnná se zapisuje do políček mapy (tab. 4).

Tab.4: Karnaughova mapa pro jednu, dvě, tři a čtyři proměnné



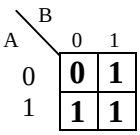






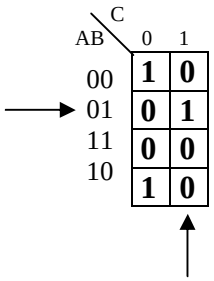
Mapa funkce dvou proměnných má čtyři políčka. Jako příklad si můžeme uvést tabulku stavů pro logickou funkci OR a k této tabulce odpovídající Karnaughovu mapu.

A	**B**	**Z**
0	0	**0**
0	1	**1**
1	0	**1**
1	1	**1**



Mapa funkce tří proměnných má osm políček, stejně jako má osm řádků tabulka pro tři proměnné. Pořadí sloupců v mapě neodpovídá pořadí, v němž píšeme řádky tabulky. To proto, aby se v sousedních sloupcích změnila hodnota pouze jedné proměnné. Důvodem je minimalizace z mapy.

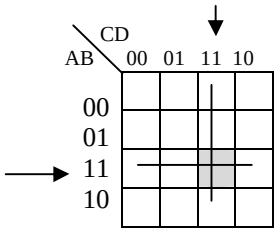
A	B	C	Z
0	0	0	**1**
0	0	1	**0**
0	1	0	**0**
0	1	1	**1**
1	0	0	**1**
1	0	1	**0**
1	1	0	**0**
1	1	1	**0**



V tabulce je uvedena ukázka jak určit polohu příslušného políčka pro dané hodnoty proměnných:
A=0, B=1, C=1

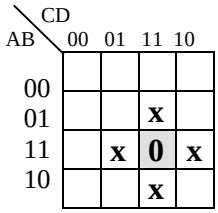
Mapa funkce čtyř proměnných má šestnáct políček rozdělených do čtyř řádků a čtyř sloupců. Pořadí řádků a sloupců je stejné: 00, 01, 11, 10. Je to opět proto, aby se v sousedních řádcích a sousedních sloupcích změnila hodnota pouze jedné proměnné. Polohu políčka na mapě čtyř proměnných určujeme stejným způsobem, jako jsme ji určovali na mapě pro jiný počet proměnných.

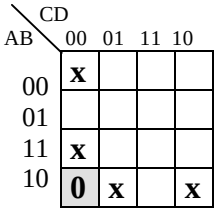
A	B	C	D	Z
0	0	0	0	**1**
0	0	0	1	**0**
0	0	1	0	**1**
0	0	1	1	**1**
0	1	0	0	**0**
0	1	0	1	**0**
0	1	1	0	**1**
0	1	1	1	**1**
1	0	0	0	**1**
1	0	0	1	**1**
1	0	1	0	**0**
1	0	1	1	**1**
1	1	0	0	**1**
1	1	0	1	**1**
1	1	1	0	**1**
1	1	1	1	**1**

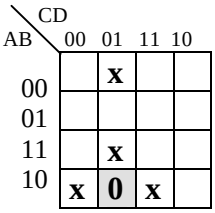


Na mapě je určena poloha políčka pro hodnoty proměnných: A=1, B=1, C=1, D=1

Příklady odvození **sousedních** políček pro políčko zadané:



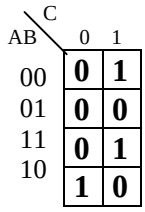




2.7. Zápis algebraického výrazu funkce z Karnughovy mapy

Z mapy lze přímo určit odpovídající výraz ve formě součtu součinů nebo součinu součtů, přičemž se budeme snažit o co nejjednodušší a nejvhodnější zápis.

Příklad 7: Z dané mapy určete algebraický výraz ve tvaru součtu součinů.



Zápis je obdobný jako z tabulky stavů:

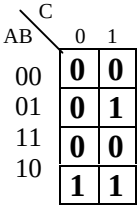
$$Z = \overline{A} \cdot \overline{B} \cdot C + A \cdot B \cdot C + A \cdot \overline{B} \cdot \overline{C}$$
$$0 \cdot 0 \cdot 1 \quad + 1 \cdot 1 \cdot 1 \quad + \quad 1 \cdot 0 \cdot 0$$

2.8. Sestavení Karnaughovy mapy z algebraického výrazu

Příklad 8: Nakreslete mapu a zapište do ní hodnoty fce Z, která je dána vztahem: $Z = \overline{A}BC + A\overline{B}$

Řešení:

$$Z = \overline{A}BC + A\overline{B}(C + \overline{C}) = \overline{A}BC + A\overline{B}C + A\overline{B} \cdot \overline{C}$$
$$0 \cdot 1 \cdot 1 \quad + 1 \cdot 0 \cdot 1 \quad + \quad 1 \cdot 0 \cdot 0$$



2.9. Minimalizace pomocí Karnaughovy mapy

AB \ C	0	1
00		
01		
11		**1**
10		**1**

Na uvedené mapě jsou zapsány dvě jedničky v sousedních políčkách, která mají tu vlastnost, že různé hodnoty má pouze proměnná **B**. Napíšeme nyní výraz za obě jedničky:

$$Z = A\overline{B}C + ABC$$

Vidíme, že výraz se skládá ze dvou členů a lze jej dále zjednodušit.

$$Z = AC(B + \overline{B})$$

$$Z = AC$$

Pokusme se nyní sestavit obecné pravidlo, které by určilo, jakým způsobem můžeme za sousední políčka v mapě napsat přímo jeden člen. Jak již bylo uvedeno, mění se při přechodu mezi sousedními políčky jedna proměnná - v našem případě proměnná **B**. Proto tento člen nezávisí na této proměnné a je určen jen ostatními proměnnými, v našem případě výrazem **AC**.

Toto pravidlo platí i pro větší počet políček.

Při minimalizaci pomocí Karnaughovy mapy volíme následující postup:

1. ***Jedničky v mapě uzavíráme pomocí smyček, které obsahují dvě, čtyři nebo osm sousedních políček. Přitom je třeba mít na paměti, že čím větší smyčku vytvoříme, tím je odpovídající výraz jednodušší.***
2. ***Smyčky se mohou navzájem protínat.***
3. ***Za každou smyčku zapíšeme pouze jeden výraz, který neobsahuje ty proměnné, jejichž hranice smyčku protínají.***
4. ***Algebraické výrazy za jednotlivé smyčky sečteme.***

Závěrem lze říci, že smyčky s jedničkami odpovídají výrazu ve tvaru součtu součinů. Obdobným způsobem můžeme vytvářet i smyčky s nulami. V tom případě však získáme vztah ve tvaru součinu součtů.

Nyní si celý postup minimalizace pomocí Karnaughovy mapy objasníme na několika příkladech, kde bude naším úkolem napsat algebraické výrazy v obou tvarech pro uvedené mapy.

Příklad 9:

a) součin součtů:

AB \ C	0	1
00	**1**	**0**
01	**1**	**0**
11	**1**	**1**
10	**1**	**1**

$$Z = A + \overline{C}$$

b) součet součinů:

AB \ C	0	1
00	**1**	**0**
01	**1**	**0**
11	**1**	**1**
10	**1**	**1**

AB \ C	0	1
00	**1**	**0**
01	**1**	**0**
11	**1**	**1**
10	**1**	**1**

*V této části mapy se mění stavy proměnných **A** a **B**. Tyto proměnné vyloučíme a Zůstane proměnná **C** (ovšem negovaná)*

*V této části se mění stavy proměnných **B** a **C**. Proto se tyto proměnné vyloučí a zůstane proměnná **A**.*

Výsledný zápis je opět: $Z = A + \overline{C}$

Příklad 10: *sestavte minimalizovaný výraz za pomocí Karnaughovy mapy.*

AB \ CD	00	01	11	10
00	**0**	**0**	**1**	**0**
01	**0**	**1**	**1**	**0**
11	**0**	**1**	**1**	**0**
10	**0**	**1**	**1**	**0**

Řešení:

AB \ CD	00	01	11	10
00	**0**	**0**	**1**	**0**
01	**0**	**1**	**1**	**0**
11	**0**	**1**	**1**	**0**
10	**0**	**1**	**1**	**0**

V této části tabulky se mění stavy proměnných ***A*** *a* ***C****. Proto se vyloučí a zůstane proměnná* ***B*** *a* ***D****.*

AB \ CD	00	01	11	10
00	**0**	**0**	**1**	**0**
01	**0**	**1**	**1**	**0**
11	**0**	**1**	**1**	**0**
10	**0**	**1**	**1**	**0**

Zde se mění stavy proměnné ***B*** *a* ***C****. Po vyloučení těchto proměnných zůstanou proměnné* ***A*** *a* ***D****.*

AB \ CD	00	01	11	10
00	**0**	**0**	**1**	**0**
01	**0**	**1**	**1**	**0**
11	**0**	**1**	**1**	**0**
10	**0**	**1**	**1**	**0**

Zde se mění stavy proměnných ***A*** *a* ***B****. Po jejich vyloučení zůstanou proměnné* ***C*** *a* ***D****.*

Výsledný výraz je dán součtem jednotlivých částí tabulky:

$$Z = BD + AD + CD$$

Sestavení výrazu pomocí součinu součtů:

AB \ CD	00	01	11	10
00	**0**	**0**	**1**	**0**
01	**0**	**1**	**1**	**0**
11	**0**	**1**	**1**	**0**
10	**0**	**1**	**1**	**0**

AB \ CD	00	01	11	10
00	**0**	**0**	**1**	**0**
01	**0**	**1**	**1**	**0**
11	**0**	**1**	**1**	**0**
10	**0**	**1**	**1**	**0**

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

Příklad 11: *sestavte minimalizovaný výraz za pomocí Karnaughovy mapy.*
metodou součtu součinů:

AB \ CD	00	01	11	10
00	**1**	**1**	**1**	**1**
01	**1**	**1**	**1**	**0**
11	**0**	**0**	**1**	**0**
10	**1**	**0**	**1**	**1**

AB \ CD	00	01	11	10
00	**1**	**1**	**1**	**1**
01	**1**	**1**	**1**	**0**
11	**0**	**0**	**1**	**0**
10	**1**	**0**	**1**	**1**

AB \ CD	00	01	11	10
00	**1**	**1**	**1**	**1**
01	**1**	**1**	**1**	**0**
11	**0**	**0**	**1**	**0**
10	**1**	**0**	**1**	**1**

Výsledný výraz je dán součtem jednotlivých částí tabulky:

$$Z = \overline{A} \cdot \overline{C} + CD + \overline{B} \cdot \overline{D}$$

metodou součinu součtů:

AB \ CD	00	01	11	10
00	**1**	**1**	**1**	**1**
01	**1**	**1**	**1**	**0**
11	**0**	**0**	**1**	**0**
10	**1**	**0**	**1**	**1**

AB \ CD	00	01	11	10
00	**1**	**1**	**1**	**1**
01	**1**	**1**	**1**	**0**
11	**0**	**0**	**1**	**0**
10	**1**	**0**	**1**	**1**

AB \ CD	00	01	11	10
00	**1**	**1**	**1**	**1**
01	**1**	**1**	**1**	**0**
11	**0**	**0**	**1**	**0**
10	**1**	**0**	**1**	**1**

Výsledný výraz je dán součinem jednotlivých částí tabulky:

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

3. SCHÉMA LOGICKÉHO OBVODU

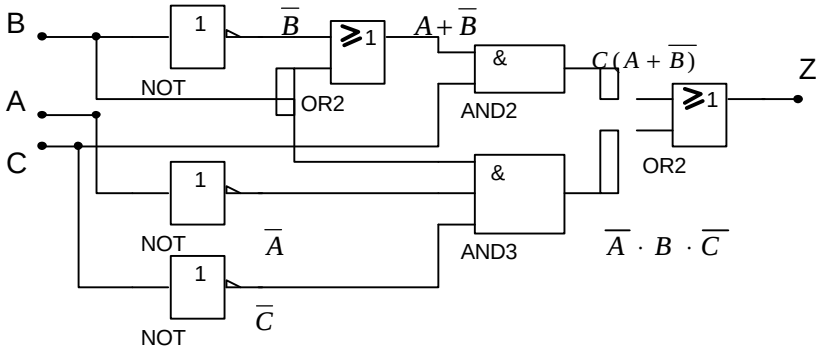
Na základě znalostí algebraického výrazu, který popisuje činnost určitého log. obvodu, můžeme nakreslit jeho schéma.

Způsob kreslení si ukážeme na příkladě č. 12.

Příklad 12: Nakreslete blokové schéma obvodu, jehož činnost je vyjádřena vztahem:

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

Řešení je následující:

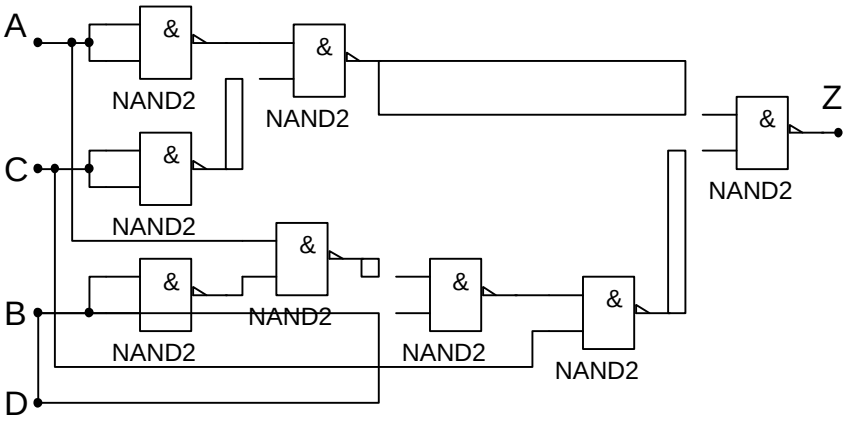


3.1. Realizace obvodu Shefferovou funkcí

V praxi se velmi často setkáváme s log. členy uskutečňujícími Shefferovu a Pierceovu funkci především proto, že jejich realizace je technickými prostředky snadná. Tyto funkce tvoří základ Shefferovy a Pierceovy algebry. Dále si ukážeme, jak je třeba postupovat, abychom získali schéma obvodu, který by byl složen jen ze členů uskutečňujících Shefferovu funkci. Celý postup bude spočívat ve vhodných úpravách následujícího výrazu.

Příklad 13: Upravte následující výraz na výraz vhodný pro realizaci pomocí Shefferových členů.

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

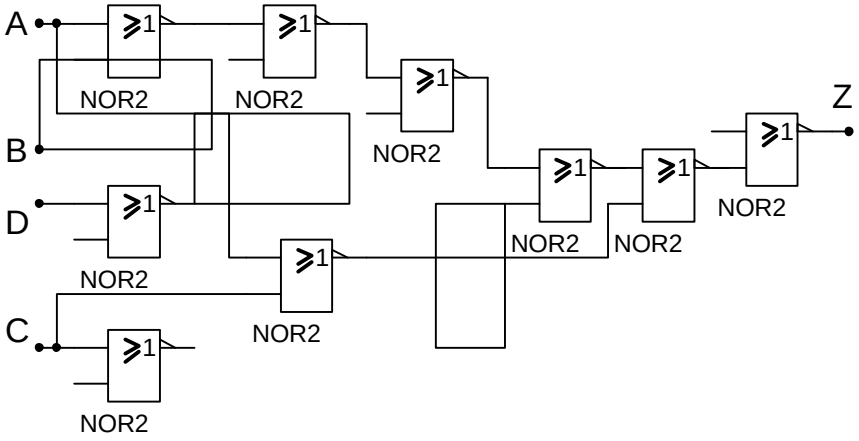


3.2.Realizace obvodu Pierceovou funkcí

Pierceův člen nám uskutečňuje negaci disjunkce nebo negaci. Abychom mohli algebraický výraz vyjádřený ve tvaru součtu součinů nebo součinu součtů realizovat Pierceovými členy, musíme jej upravit na postupné negace disjunkcí. Způsob úpravy si ukážeme na příkladě.

Příklad 14: Upravte následující výraz na výraz vhodný pro realizaci pomocí Pierceových členů.

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



4. NÁVRHY A ŘEŠENÍ KOMBINAČNÍCH LOGICKÝCH OBVODŮ

Při návrhu jednoduchých ovládacích obvodů volíme zpravidla tento postup:

1. Vyjádříme podmínky činnosti zařízení.
2. Sestavíme rovnice log. funkce a provedeme její minimalizaci.
3. Navrhneme log. členy, které použijeme pro realizaci obvodu.
4. Nakreslíme schéma log. obvodu.

4.1. Návrh log. obvodu s jedním výstupem

Příklad 15: Navrhněte obvod pro ovládání žárovky ze dvou míst.
Je-li sepnut lichý počet spínačů, žárovka svítí (sepnutý spínač má hodnotu log. 1 - tj. žárovka svítí).
Řešení: Sestavíme tabulku stavů a Karnaughovu mapu.

A	B	Z
0	0	**0**
0	1	**1**
1	0	**1**
1	1	**0**

A \ B	0	1
0	**0**	**1**
1	**1**	**0**

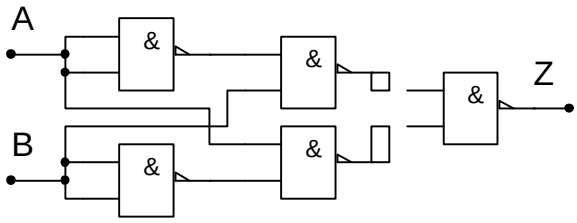
Z mapy vyplývá, že nemůžeme minimalizovat
Napíšeme algebraický výraz:

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

Daný výraz převedeme do úplné Schefferovy funkce:

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

Z uvedené rovnice navrhneme schéma obvodu:



Příklad 16: Navrhněte obvod (schodišťový vypínač) pro ovládání žárovky ze tří míst.
Je-li sepnut lichý počet spínačů, žárovka svítí (sepnutý spínač má hodnotu log. 1)
Řešení: Sestavíme tabulku stavů a Karnaughovu mapu.

A	B	C	Z
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**

AB \ C	0	1
00	**0**	**1**
01	**1**	**0**
11	**0**	**1**
10	**1**	**0**

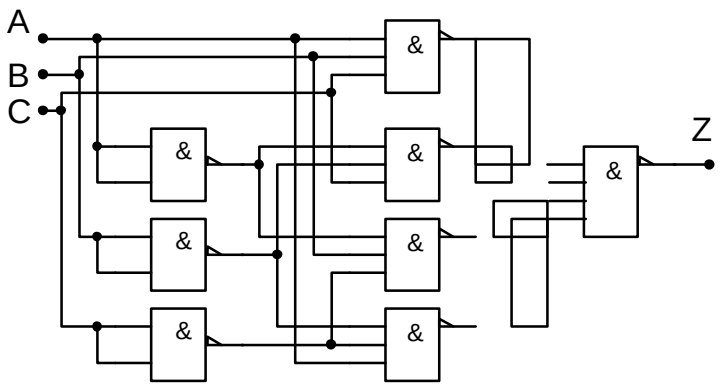
Z mapy vyplývá,že nemůžeme minimalizovat.
Napíšeme algebraický výraz:

$$Z = \overline{A} \cdot \overline{B} \cdot C + \overline{A} \cdot B \cdot \overline{C} + ABC + A \cdot \overline{B} \cdot \overline{C}$$

Převod do Schefferovy funkce:

$$Z = \overline{\overline{A} \cdot \overline{B} \cdot C + \overline{A} \cdot B \cdot \overline{C} + ABC + A \cdot \overline{B} \cdot \overline{C}}$$

$$Z = \overline{\overline{A} \cdot \overline{B} \cdot C} \cdot \overline{\overline{A} \cdot B \cdot \overline{C}} \cdot \overline{ABC} \cdot \overline{A \cdot \overline{B} \cdot \overline{C}}$$

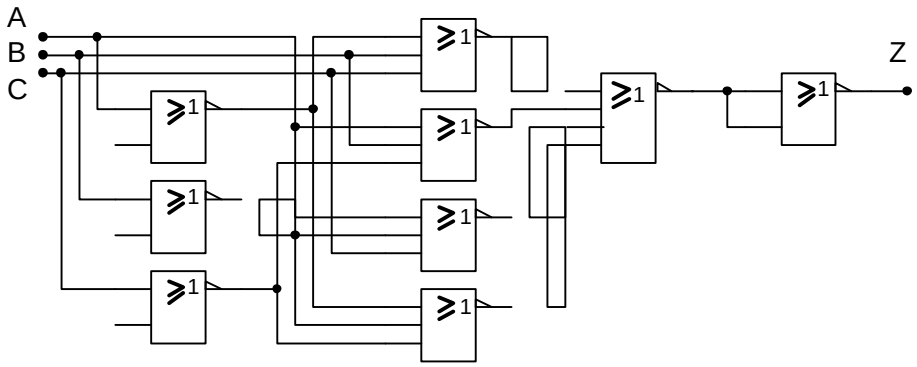


Převod do Pieceovy funkce:

$$Z = \overline{\overline{\overline{A}\cdot\overline{B}\cdot C} + \overline{\overline{A}\cdot B\cdot\overline{C}} + \overline{AB\overline{C}} + \overline{A\cdot\overline{B}\cdot\overline{C}}}$$

$$Z = \overline{\overline{A+B+\overline{C}} + \overline{A+\overline{B}+C} + \overline{\overline{A}+\overline{B}+\overline{C}} + \overline{\overline{A}+B+C}}$$

$$Z = \overline{\overline{\overline{A+B+\overline{C}} + \overline{A+\overline{B}+C} + \overline{\overline{A}+\overline{B}+\overline{C}} + \overline{\overline{A}+B+C}}}$$



4.2. Návrh logického obvodu s větším počtem výstupů

Při návrhu log. obvodů se často vyskytují obvody, které mají větší počet výstupů. Funkci, kterou musí realizovat navrhovaný obvod, budeme zadávat tabulkou stavů a tuto tabulku přepíšeme do Karnaughových map. Z nich získáme minimalizované výrazy, podle kterých následně navrhneme schéma logických obvodů s jedním výstupem. Log. obvod s větším počtem výstupů pak dostaneme sloučením jednotlivých log. obvodů s jedním výstupem do jediného schématu. Jestliže ve schématech s jedním výstupem existuje log. člen, který realizuje stejnou funkci v několika z nich, pak ve výsledném schématu bude tento člen nakreslen pouze jednou.

Příklad 17: Navrhněte log. obvod, který má sčítat tři dvojková čísla o jednom řádu.

Řešení: Při aritmetickém sčítání tří dvojkových čísel o jednom řádu bude výsledek vyjádřen dvojkovým číslem o dvou řádech. Z tabulky stavů sestavíme Karnaughovy mapy pro jednotlivé výstupní proměnné a z nich napíšeme příslušné algebraické výrazy.

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

→ Zde algebraicky sčítáme dvojková čísla: 0+1=01
1+1=10
1+1+1=11

Y

AB \ C	0	1
00	0	0
01	0	1
11	1	1
10	0	1

Z

AB \ C	0	1
00	0	1
01	1	0
11	0	1
10	1	0

$$Y = AB + AC + BC$$

$$Z = \overline{A}\cdot\overline{B}\cdot C + \overline{A}\cdot B\cdot\overline{C} + ABC + A\cdot\overline{B}\cdot\overline{C}$$

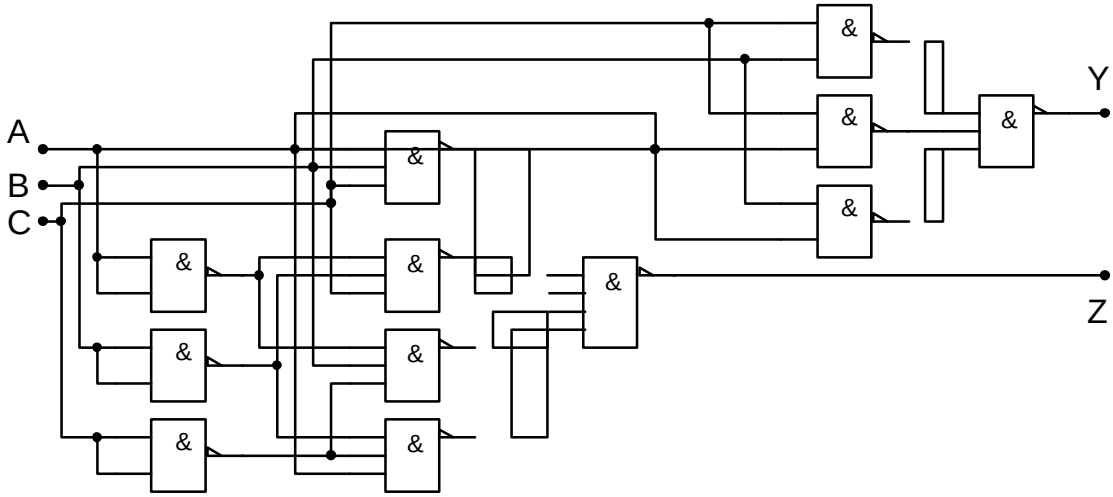
Převod do Shefferovy funkce:

$$Y = \overline{AB + AC + BC}$$

$$Y = \overline{\overline{AB}\cdot\overline{AC}\cdot\overline{BC}}$$

$$Z = \overline{\overline{\overline{A}\cdot\overline{B}\cdot C + \overline{A}\cdot B\cdot\overline{C} + ABC + A\cdot\overline{B}\cdot\overline{C}}}$$

$$Z = \overline{\overline{\overline{A}\cdot\overline{B}\cdot C}\cdot\overline{\overline{A}\cdot B\cdot\overline{C}}\cdot\overline{ABC}\cdot\overline{A\cdot\overline{B}\cdot\overline{C}}}$$



4.3. Návrh obvodu zadaného neúplnou tabulkou stavů

Doposud jsme se setkali pouze s případy. kdy log. obvod byl zadám úplnou tabulkou stavů. Často se však vyskytne případ, že se některé kombinace vstupních proměnných nemohou na vstupu navrhovaného obvodu vyskytnout, nebo výstupní proměnná není pro danou kombinaci určena. V tom případě je v tabulce stavů na místě hodnoty výstupní proměnné pomlčka. Abychom mohli z této neúplné tabulky vytvořit algebraický výraz, musíme místo každé pomlčky napsat buď jedničku nebo nulu. Neúplnou tabulku stavů můžeme doplnit různým způsobem. Doplnění však provedeme tak, abychom dostali co nejjednodušší algebraický výraz. Proto je vhodné doplnit až Karnaughovu mapu se zřetelem na minimalizaci. Návrh log. obvodu zadaného neúplnou tabulkou stavů si vysvětlíme na příkladě.

Příklad 18:

Ve slévárně jsou tři pece a plní se postupně v libovolném pořadí. Při plnění pece vždy musíme otevřít její uzávěr. Navrhněte log. obvod, který z bezpečnostních důvodů signalizuje otevření uzávěru.

Řešení:

Každý uzávěr má kontakt, který vydává signál, když je otevřen. Podle podmínek sestavíme tabulku. Protože více uzávěrů nemůže být otevřeno současně, napíšeme do sloupce **Z** pro dané kombinace vstupních proměnných pomlčky. Nyní přepíšeme tabulku do Karnaughovy mapy a doplníme ji, v našem případě jedničkami. Z mapy pak sestavíme odpovídající algebraický výraz.

A	B	C	Z
0	0	0	**0**
0	0	1	**1**
0	1	0	**1**
0	1	1	-
1	0	0	**1**
1	0	1	-
1	1	0	-
1	1	1	-

AB \ C	0	1
00	**0**	**1**
01	**1**	-
11	-	-
10	**1**	-

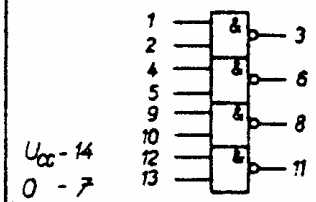
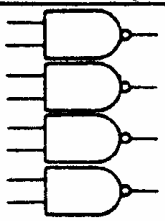
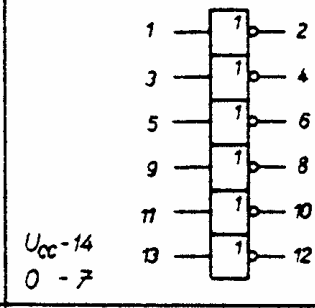
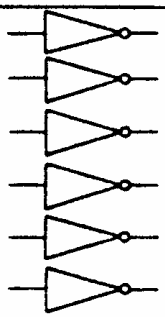
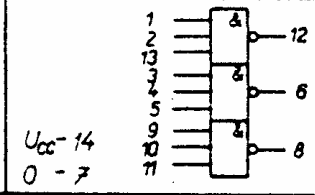
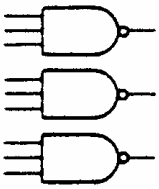
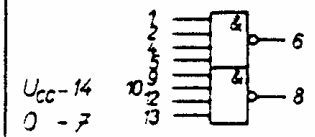
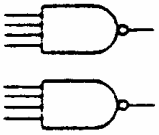
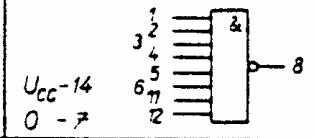
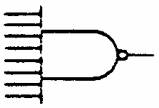
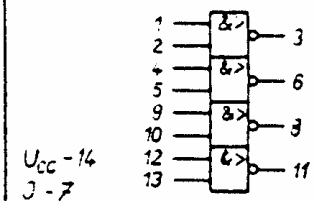
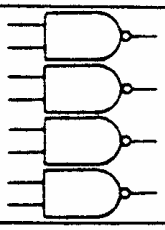
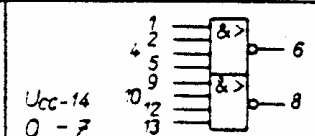
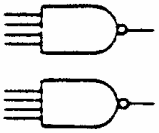
AB \ C	0	1
00	**0**	**1**
01	**1**	**1**
11	**1**	**1**
10	**1**	**1**

$$Z = A + B + C$$

5. Kombinační obvody

Obvod se nazývá kombinační, jestliže jeho výstupy závisejí pouze na vstupních kombinacích a ne na jejich předchozích hodnotách. Jediné kombinaci vstupních proměnných odpovídá jediná výstupní kombinace. Obvod nemá žádnou paměť předchozích stavů. Chceme-li navrhnout kombinační log. obvod, musíme především vědět, jakou činnost má uskutečňovat. Ze zadání bychom měli především vyčíst, kolik vstupních a výstupních proměnných má navrhovaný obvod. Tyto proměnné pak vhodně označíme písmeny a přiřadíme jim symbolické označení 0 a 1, vyjadřující, že daná činnost existuje nebo neexistuje. Příklady jednoduchých kombinačních obvodů jsou uvedeny v příkladech 15 - 18.

Základní integrované obvody

Typ	Název – funkce	Schematický znak	Schematický znak dřívější
MH7400 MH7403	čtveřice dvouvstupových členů NAND $Y = \overline{A.B}$ čtveřice dvouvstupových členů NAND s otevřeným kolektorem		
MH7404 MH7405	šestice invertorů $Y = \overline{A}$ šestice invertorů s otevřeným kolektorem		
MH7410	trojice 3 [redacted]vstupových členů NAND $Y = \overline{A.B.C}$		
MH7420	dvojice čtyřvstupových členů NAND $Y = \overline{A.B.C.D}$		
MH7430	osmivstupový člen NAND $Y = \overline{A.B.C.D.E.F.G.H}$		
MH7437 MH7438	čtveřice dvouvstupových výkonových členů NAND $Y = \overline{A.B}$ čtveřice dvojvstupových výkonových členu NAND s otevřeným kolektorem		
MH7440	dvojice čtyřvstupových výkonových členu NAND $Y = \overline{A.B.C.D}$		

5.1. Kodéry a dekodéry

Typickým představitelem kombinačních log. obvodů jsou různé převodníky kódů, které převádějí informace z jedné kódované formy do jiné kódované formy. Kódem se rozumí pravidlo, podle něhož určité kombinaci nul a jedniček (nebo stavu L a H) přiřazujeme nějaké desítkové číslo. Kódy můžeme rozdělit do dvou hlavních skupin. Jsou to dvojkový (binární) kód a kód desítkový (BCD).

Dvojkový kód: přirozenému pořadí dvojkových čísel je přiřazeno přirozené pořadí čísel desítkových.

Kód BCD (Binary Coded Decimal – tj. dvojkově kódované desítkové číslo): v těchto kódech je desítkovým číslům 0 až 9 (tj. deset hodnot) přiřazeno deset dvojkových čísel o čtyřech bitech. Čísla jsou organizována v dekádách. Např. pro vyjádření desítkového čísla řádu 10^2 je nutno použít tři čtyřbitová

čísla. Jedno (nejvíce vpravo) vyjadřuje jednotky, druhé desítky, třetí stovky. Nejjednodušším kódem **BCD** je takový kód, v němž je přirozenému pořadí dvojkových čísel přiřazeno přirozené pořadí desítkových čísel v celém rozsahu tj. do čísla 9. Tento kód se označuje jako **BCD 1248** . Např. číslo 127 bude v tomto kódu vyjádřeno číslem 0001 0010 0111.
Kodéry převádějí desítková čísla do binární soustavy a dekodéry z binární soustavy do desítkové.
Nejčastější úlohou je převod desítkového čísla na dvojkové v binárním kódu.

Příklad 19:
Navrhněte obvod pro převod dekadických čísel 0, 1, 2, 3 do soustavy binární.
Řešení:
Nejprve si sestavíme tabulku stavů: Jeden sloupec (des.) vyhradíme pro desítková čísla. Do dalších sloupců těmto desítkovým číslům přiřadíme kombinace stisknutého tlačítka (např. tlačítko C představuje číslo 1, tlačítko B představuje číslo 2 a tlačítko A číslo 3). Stisk tlačítka představuje log.***1*** a právě stiskem příslušného tlačítka na vstupu jednoznačně detekujeme desítkové číslo, které jsme si k tlačítku přiřadili.

Des.	A	B	C	Y	Z
0	0	0	0	0	0
1	0	0	1	0	1
2	0	1	0	1	0
	0	1	1	-	-
3	1	0	0	1	1
	1	0	1	-	-
	1	1	0	-	-
	1	1	1	-	-

Jedná se o neúplnou tabulku stavů (při minimalizaci ovšem můžeme na místo pomlček do Karnaughovy mapy zapsat hodnotu log. 1 – z důvodu co největší minimalizace) Tabulku přepíšeme do dvou map, napíšeme rovnice a navrhneme obvod.

Y

AB \ C	0	1
00	0	0
01	1	1
11	1	1
10	1	1

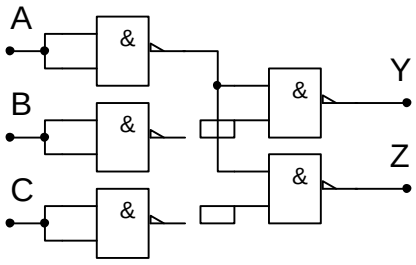
Z

AB \ C	0	1
00	0	1
01	0	1
11	1	1
10	1	1

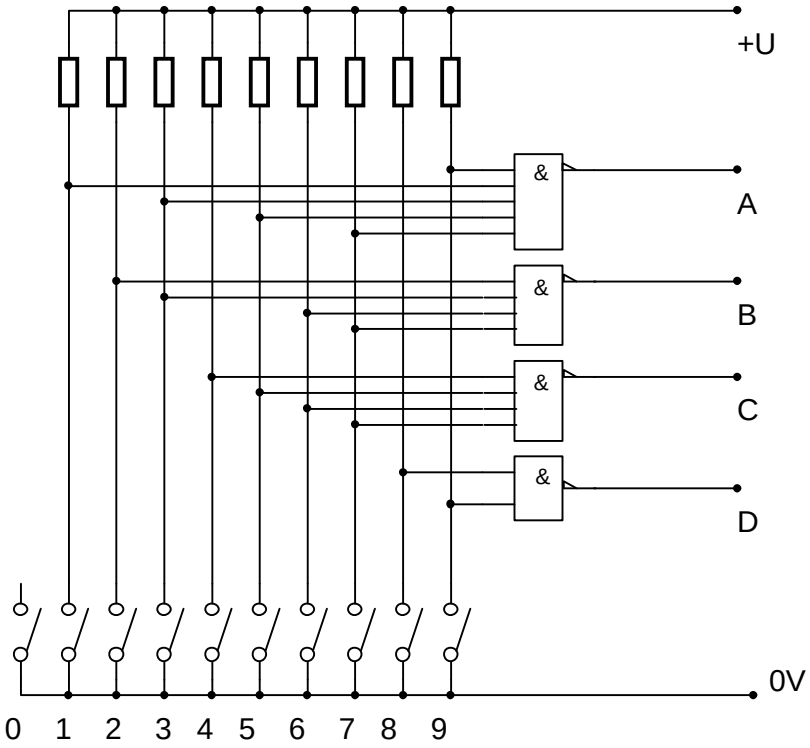
$$Y = A + B \qquad Z = A + C$$

převedeno na Shefferovu funkci:

$$Y = \overline{\overline{A} \cdot \overline{B}} \qquad Z = \overline{\overline{A} \cdot \overline{C}}$$



Příklad 20: Realizujte převodník desítkových čísel na čísla dvojková v kódu **BCD 1248**
Řešení: Budeme postupovat na základě tabulky popisující převodní vztah mezi desítkovými čísly a dvojkovými v kódu **BCD 1248** (viz tab. u schématu zapojení). desítkové číslo je převáděno na binární ve složení D, C, B, A. Symbolem D je vyznačena číslice nejvyššího řádu (též váhy), zde řádu osm. Tento převodník můžeme realizovat čtyřmi log. členy **NAND**, přičemž výstup každého z nich přísluší jednomu bitu dvojkového čísla. Vstupní obvod převodníku může být tvořen odpory, jimiž se jednotlivé vstupy log. členů udržují ve stavu ***H***. Jednotlivá desítková čísla, která chceme převádět, můžeme volit spínači (tlačítky) ***0*** až ***9***. Sepnutím spínače se příslušná úroveň ***H*** změní v úroveň ***L***. Log. členy **NAND**, jejichž vstupy takto nabyly úrovně ***L***, budou mít na výstupu stav ***H***. Výstup **A** přísluší prvnímu bitu zprava dvojkového čísla, výstup **B** druhému, výstup **C** třetímu a výstup **D** čtvrtému bitu čísla. Na spínač příslušný danému desítkovému číslu musíme připojit vstupy těch členů **NAND**, na jejichž výstupu má být podle ekvivalentního dvojkového čísla stav ***H***. Např. dvojkovým ekvivalentem čísla ***6*** je ***0110***. Ke spínači ***6*** musíme tedy přivést vstupy členů **B** a **C**. Sepneme-li spínač ***6***, bude výstup **A** ve stavu ***L***, výstup **B** ve stavu ***H***, výstup **C** ve stavu ***H*** a výstup **D** ve stavu ***L***, což vyjadřuje číslo 0110. Podle téhož pravidla je možno realizovat převodníky pro libovolný kód.



Tabulka:

des.	D 8	C 4	B 2	A 1
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

Opačnou úlohou je převod dvojkového čísla v určitém kódu na číslo desítkové.

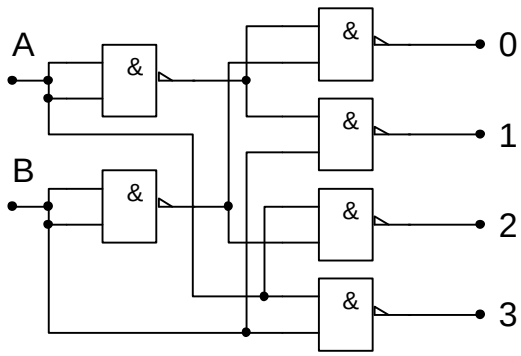
Příklad 21: Navrhněte převodník dvojkového čísla do desítkového 1 ze 4.

Řešení: Sestavíme si tabulku stavů. Každému dvojkovému číslu na vstupu zde odpovídá jeden výstup, jehož stav je odlišný od stavu výstupů ostatních, tj. je aktivní. V daném uspořádání dle tabulky je aktivní výstup ve tvaru log. ***0 (L)***, což je výhodné z hlediska součástkové základny. Výstupem ve stavu log. ***0*** lze také dobře řídit (spínat) zátěž např. zobrazovací prvek. Ty výstupy, které nejsou aktivní mají hodnotu log. ***1 (H).***

A	B	Z_0	Z_1	Z_2	Z_3
		0	1	2	3
0	0	0	1	1	1
0	1	1	0	1	1
1	0	1	1	0	1
1	1	1	1	1	0

$Z_0 = (0) = \overline{\overline{A} \cdot \overline{B}}$ $\qquad$ $Z_2 = (2) = \overline{A \cdot \overline{B}}$

$Z_1 = (1) = \overline{\overline{A} \cdot B}$ $\qquad$ $Z_3 = (3) = \overline{A \cdot B}$



Vztahy určují takové negace v součinech veličin **A**, a **B**, aby byl stav daného výstupu log. ***0 (L)***. Např. pro výstup 2 musíme nejprve hodnotu A ***(1)*** násobit negovanou hodnotou B ***(0)*** Výsledek tohoto součinu je roven ***1 (H)*** a tento pak musíme negovat, abychom dostali výsledný stav log. ***0 (L)***. Protože je ve všech vztazích použita negace součinu, použijeme k realizaci převodníku logické členy **NAND**.

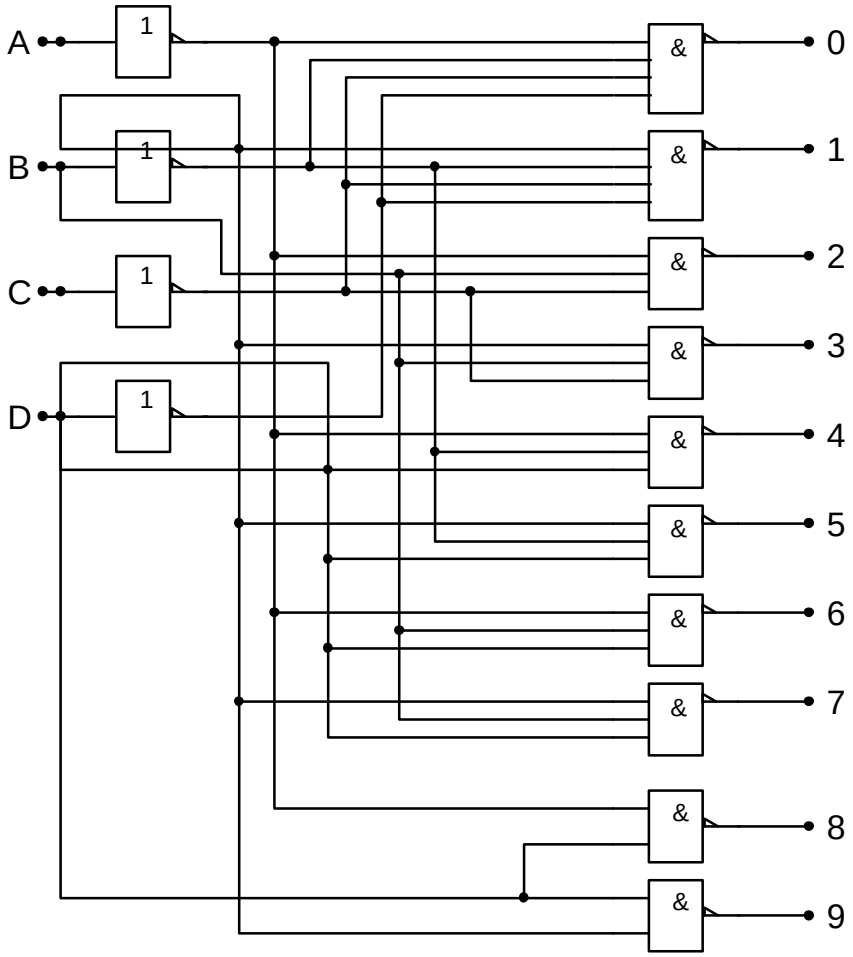
Velmi častou úlohou je převod čísel v kódu **BCD** na čísla desítková. V podstatě se jedná o převod čísla o čtyřech bitech na jedno desítkové číslo z deseti možných, tj. o převod v kódu: **1 z 10**.
Při návrhu vycházíme z příslušné tabulky (viz tab.4)

Tab.4: pravdivostní tab. převodníku z kódu: **BCD 1248** na kód: **1 z 10**

des.	D	C	B	A	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	**0**	1	1	1	1	1	1	1	1	1
1	0	0	0	1	1	**0**	1	1	1	1	1	1	1	1
2	0	0	1	0	1	1	**0**	1	1	1	1	1	1	1
3	0	0	1	1	1	1	1	**0**	1	1	1	1	1	1
4	0	1	0	0	1	1	1	1	**0**	1	1	1	1	1
5	0	1	0	1	1	1	1	1	1	**0**	1	1	1	1
6	0	1	1	0	1	1	1	1	1	1	**0**	1	1	1
7	0	1	1	1	1	1	1	1	1	1	1	**0**	1	1
8	1	0	0	0	1	1	1	1	1	1	1	1	**0**	1
9	1	0	0	1	1	1	1	1	1	1	1	1	1	**0**
10	1	0	1	0	1	1	1	1	1	1	1	1	1	1

Při sestavování log. vztahů přihlédneme k tomu, že čtyřbitové číslo dává 16 možných kombinací, z nichž využíváme jen 10. Pracujeme s kombinacemi nejvýše do čísla 9 = 1001. Odtud plynou některá zjednodušení log. vztahů. Číslo $(\mathbf{9})_{10}$ můžeme vyjádřit jen součinem $A \cdot D$, neboť jiné kombinace jako např. $A\overline{B}CD$ -$(\mathbf{13})_{10}$, $AB\overline{C}D$ -$(\mathbf{11})_{10}$ či $ABCD$ -$(\mathbf{15})_{10}$ nepřicházejí v úvahu.

K realizaci dekodéru potřebujeme 10 logických členů **NAND**. Pro jednoduchost nebudeme ve vztazích zapisovat výsledné negace součinů, které přísluší funkci **NAND**, ale jen dílčí negace proměnných (tyto vztahy jsou uvedeny u schématu) a podle nich pak vytvoříme schéma. Poněvadž se všechny proměnné **A**, **B**, **C**, **D** ve vztazích objevují i v dílčích negacích, musíme použít 4 invertory. Zapojení dekodéru je uvedeno na obr. 5. Protože jsme při návrhu dekodéru vyloučili kombinace proměnných odpovídající číslům větším než 9, nesmíme tato větší čísla na dekodér přivádět (přivedeme-li je, dá dekodér chybný výsledek). Na druhé straně je možno pořadí čísel libovolně zkrátit, tj. použít dekodér např. jen do čísla 6.



$0 = \overline{A}.\overline{B}.\overline{C}.\overline{D}$
$1 = A.\overline{B}.\overline{C}.\overline{D}$
$2 = \overline{A}.B.\overline{C}.\overline{D} = \overline{A}.B.\overline{C}$
$3 = A.B.\overline{C}.\overline{D} = A.B.\overline{C}$
$4 = \overline{A}.\overline{B}.C.\overline{D} = \overline{A}.\overline{B}.C$
$5 = A.\overline{B}.C.\overline{D} = A.\overline{B}.C$
$6 = \overline{A}.B.C.\overline{D} = \overline{A}.B.C$
$7 = A.B.C.\overline{D} = A.B.C$
$8 = \overline{A}.\overline{B}.\overline{C}.D = \overline{A}.D$
$9 = A.\overline{B}.\overline{C}.D = A.D$

nemohou nastat kombinace:

$10 = \overline{A} \cdot B \cdot \overline{C} \cdot D$
$11 = A \cdot B \cdot \overline{C} \cdot D$
$12 = \overline{A} \cdot \overline{B} \cdot C \cdot D$
$13 = A \cdot \overline{B} \cdot C \cdot D$
$14 = \overline{A} \cdot B \cdot C \cdot D$
$15 = A \cdot B \cdot C \cdot D$

Pozn.: kombinaci pro nulu je nutno ponechat v původním stavu (kryje se zčásti s kombinací pro číslo 8). Obdobně nutno ponechat kombinaci pro jedničku (kryla by se s číslem 9).

Obr. 5: Převodník čísel v kódu BCD na kód 1 z 10.

Vztahy v této formě jsou logickými součiny, tj. odpovídají funkci **AND**. Při použití log. členů **AND** bychom dostali aktivní výstupy ve stavu ***H***. Poněvadž je však žádán pro aktivní výstupy stav ***L***, musíme logické součiny negovat tj. použít log. členy **NAND**.