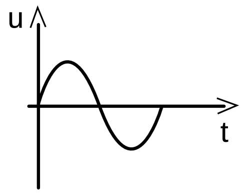
Obsah

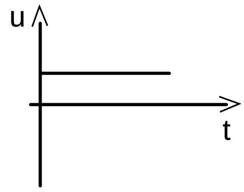
1. Základy číslicovej techniky

Logické systémy

Doteraz ste sa učili obvody, ktoré boli charakterizované spojitými časovými priebehmi. Boli to väčšinou lineárne, konštantné, prípadne nelineárne priebehy a pod. (napr. sínusové priebehy s prípadnými fázovými posuvmi, ktoré boli závislé od druhu obvodu).

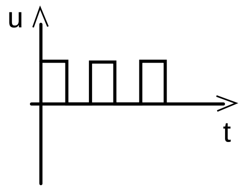
Tieto priebehy nazývame aj **analógové signály**. Analógové signály sa menia priebežne alebo mierne.





Vývojom elektroniky sa ľudstvo dopracovalo k zdokonaleniu elektronických zariadení, v ktorých sa čoraz viac a v dnešnej dobe väčšinou používajú súčiastky pracujúce na princípe číslicových metód.

Číslicové obvody sú obvody, v ktorých sa signál v čase mení nespojito (skokom). Využívajú sa napríklad v počítačoch, digitálnych meracích prístrojoch, televízií, hodinkách, telefónnych ústredniach a v mnohých ďalších zariadeniach.



Dôvody pre používanie číslicovej techniky:

- použitím číslicových integrovaných obvodov sa zmenšila váha, veľkosť a cena obvodov
- IO (integrované obvody) nahradili zložité obvody (v niektorých prípadoch aj celé prístroje)
- zväčšila sa presnosť a výkonnosť v prezentácií výsledkov aj pri samotnom vykonávaní merania
- zväčšil sa dynamický rozsah
- zväčšila sa stabilita
- pohodlnosť (priame desiatkové odčítanie dát vylučuje nesprávne odčítanie meraných hodnôt)
- automatizácia (pomocou IO alebo PC – programovanie operácií)
- jednoduchá konštrukcia
- nové prístupy pri riešení konštrukčných problémov elektronických zariadení

V tomto školskom roku začneme sa zaoberať už spomínanou časťou elektroniky - číslicovou technikou. Základom číslicovej techniky sú obvody s nespojitými ovládacími súčiastkami, ktoré sú charakteristické tým, že nadobúdajú len dva stavy - vypnutý, zapnutý (vedie - nevedie, otvorený - zatvorený).

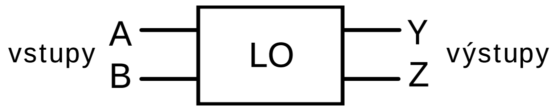
Takéto súčiastky s nespojitým časovým priebehom nazývame **logické súčiastky**. Obvody, ktoré pozostávajú s týchto logických súčiastok a uskutočňujú logickú funkciu nazývame **logické obvody**.

Pod pojmom **logický systém** rozumieme systém, pri ktorom každá vstupná a výstupná veličina sa nemôže meniť spojito, ale môže nadobúdať len dva stavy, hodnoty označované 0 a 1.

Pokiaľ chceme skúmať logický systém, musíme sa zaoberať:

- správaním logického systému
- jeho štruktúrou (súčiastky a ich väzby)

Bloková schéma logického obvodu:



Na opis správania logických systémov používame matematický model pre prácu s dvojhodnoto-vými (binárnymi) veličinami, tento model sa nazýva **Boolova algebra.**

Spôsoby vyjadrenia logických funkcií:

- názov funkcie
- slovné vyjadrenie
- pravdivostná tabuľka
- algebraické vyjadrenie
- Karnughová mapa
- kontaktová realizácia schémy
- realizácia logickými členmi

Základné operácie Boolovej algebry

Boolová algebra je dvojhodnotová logická algebra, ktorá používa disjunkciu (logický súčet), konjunkciu (logický súčin) a negáciu (logická negácia) ako úplný súbor základných logických funkcií a slúži na matematický opis zákonov a pravidiel výrokovej logiky, ktoré riešia vzťahy medzi pravdivými a nepravdivými výrokmi:

- **pravdivý výrok** - priradená hodnota logická 1
- **nepravdivý výrok** - priradená hodnota logická 0.

V Boolovej algebre sú definované **3 základné operácie**, pomocou ktorých môžeme vyjadriť ľubovoľnú logickú operáciu:

- logický súčet - **disjunkcia**
- logický súčin - **konjunkcia**
- logická negácia - **negácia**

Hodnoty závislej premennej Y, závislej od jednotlivých kombinácií nezávisle premenných A**,** B pre základné logické operácie znázorníme v **pravdivostnej tabuľke**. V ľavej časti tabuľky sú zapísané všetky kombinácie nezávislé premenných. V pravej časti tabuľky sú zapísané všetky stavy funkčných hodnôt výstupnej premennej Y.

Logický súčet

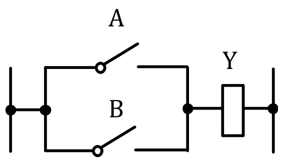
Logický súčet je operácia disjunkcie alebo zjednotenia a označuje sa znakom ∨. V praxi sa často používa algebraický zápis:

$$Y = A + B$$

Pre operáciu logického súčtu je charakteristická spojka „alebo“ (anglicky „or“). Logický člen, ktorý realizuje logický súčet, sa nazýva **člen OR**. Logický súčet sa **rovná jednotke**, keď **aspoň jedna** nezávisle premenná má hodnotu 1.

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

pravdivostná tabuľka pre logický súčet



riadková schéma kontaktovej realizácie funkcie logického súčtu

Logický súčin

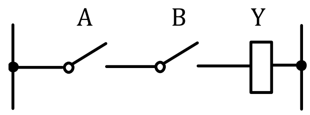
Logický súčin je operácia konjunkcie alebo prieniku, označuje sa znakom ∧. V praxi sa často používa algebraický zápis:

$$Y = A \cdot B$$

Pre operáciu logického súčinu je charakteristická spojka „aj“, „i“(anglicky „and“, „&“). Logický člen, ktorý realizuje logický súčin, sa nazýva **člen AND**. Logický súčin sa **rovná jednotke len vtedy**, keď **všetky** nezávisle premenné majú hodnotu 1. Ak aspoň jedna vstupná premenná má hodnotu 0, výsledná hodnota sa rovná 0.

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

pravdivostná tabuľka pre logický súčin



riadková schéma kontaktovej realizácie funkcie logického súčinu

Logická negácia

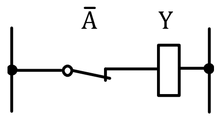
Logická negácia je operácia negácie, záporu. V praxi sa často používa algebraický zápis:

$$Y = \bar{A}$$

Pre operáciu logickej negácie je charakteristický zápis „nie“, („not“). Logický člen, ktorý realizuje logický súčin, sa nazýva **člen NOT, inventor** alebo **negátor**. Logický negácia mení hodnotu nezávislej premennej na opačnú. Jej výsledok je rovný 1, ak $A = 0$ a opačne.

A	Y
0	1
1	0

pravdivostná tabuľka pre logickú negáciu



riadková schéma kontaktovej realizácie funkcie logickej negácie

Pre Boolovu algebru platia nasledovné zákony a pravidlá:

1. Zákon komutatívny	$A + B = B + A$	$A.B = B.A$
2. Zákon asociatívny	$A + (B + C) = (A + B) + C$	$A.(B.C) = (A.B).C$
3. Zákon distributívny	$A.(B + C) = A.B + A.C$	$A + B.C = (A + B).(A + C)$
4. Zákon idempotentný	$A + A = A$ $A + 0 = A$ $A + 1 = 1$	$A.A = A$ $A.0 = 0$ $A.1 = A$
5. Zákon doplnku	$A + \bar{A} = 1$	$A.\bar{A} = 0$
6. Zákon involúcie	$A = \bar{\bar{A}}$	
7. Zákon de Morganov	$\overline{A + B} = \bar{A}.\bar{B}$	$\overline{A.B} = \bar{A} + \bar{B}$
8. Zákon absorpcie	$A.(A + B) = A$	$A + A.B = \bar{A} + B$
9. Zákon absorpcie negácie	$A + \bar{A}.B = A + B$ $\bar{A} + A.B = \bar{A} + B$	$A.(\bar{A} + B) = A.B$ $\bar{A}.(A + B) = \bar{A}.B$

Všetky zákony a pravidlá možno dokázať zostavením pravdivostnej tabuľky ľavej a pravej strany rovnosti a porovnaním.

Boolovské funkcie

Boolovské funkcie sú také, pri ktorých závislé aj nezávislé premenné môžu nadobúdať len hodnoty 0 alebo 1. Vo všeobecnosti zápis tejto funkcie môže mať tvar:

$$Y = f(A, B, C, \dots)$$

kde :

- A,B,C, ... sú nezávislé premenné (vstupne veličiny)
- Y je závislé premenná (funkčná hodnota).

Funkciu s n nezávisle premennými možno určiť pre všetky možné kombinácie hodnôt n premenných, t.j. pre $\boldsymbol{N = 2^n}$. Táto funkcia sa nazýva **úplne zadaná.**

Existujú ešte **neúplne zadané funkcie**, také, ktoré **nie sú definované vo všetkých $\boldsymbol{2^n}$ bodoch** definičného oboru. Býva to vtedy, ak niektoré kombinácie vstupných veličín neexistujú fyzikálne, alebo pri niektorých kombináciách nám nezáleží na hodnote výstupu.

Pre n premenných existuje maximálne 2^{2^n}, t.j. 2^N logických funkcií.

Príklad:

Máme 2 nezávislé premenné t.j. n = 2.

Pre 2 nezávislé premenné môžeme určiť N možných kombinácií hodnôt $\boldsymbol{N = 2^n}$ t.j. $\boldsymbol{N = 2^2 = 4}$.

Pre 2 nezávislé premenné môžeme teda určiť 4 možné kombinácie hodnôt a existuje pre ne maximálne 2^N logických funkcií, t.j. 16 ($2^N = 2^4 = 16$) logických funkcií.

Boolovská funkcia jednej premennej

Pre $n = 1$ je boolovská funkcia $Y = f\ (A)$ definovaná pre $N = 2^1 = 2$ kombinácií hodnôt A. Týchto funkcií môže byť $2^2 = 4$.

Všetky možné funkcie sú uvedené v nasledujúcej tabuľke:

A	Y1	Y2	Y3	Y4
0	0	0	1	1
1	0	1	0	1

- funkcie $Y1 = 0$ a $Y4 = 1$ sú konštantné funkcie
- funkcia $Y2 = A$ je funkcia opakovaná
- funkcia $Y3 = \bar{A}$ je funkcia negácie

Boolovská funkcie dvoch premenných

Pre $n = 2$ je boolovská funkcia $Y = f\ (A, B)$ definovaná pre $N = 2^2 = 4$ kombinácií hodnôt A, B. Týchto funkcií môže byť $2^4 = 16$.

Všetky možné funkcie sú uvedené v nasledujúcej tabuľke:

A	B	Y1	Y2	Y3	Y4	Y5	Y6	Y7	Y8	Y9	Y 10	Y 11	Y 12	Y 13	Y 14	Y 15	Y 16
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

- funkcie Y8 - logický súčet
- funkcia Y2 - logický súčin
- funkcia Y15 - negácie logického súčinu

Definičný obor logických funkcií štyroch premenných má 8 kombinácií vstupov, z čoho môžeme dostať $2^8 = 256$ funkcií.

Logické funkcie

Logická funkcia je predpis, ktorý kombinácií, prípadne aj postupnosti hodnôt jednej alebo viac (nezávislých) logických premenných jednoznačne priraďuje hodnoty jednej (závislej) premennej.

Tieto funkcie môžeme rozdeliť podľa počtu vstupných premenných na funkcie jednej, dvoch, troch a viac premenných. Ak označíme počet týchto premenných písmenom n, potom počet možných kombinácií je $C = 2^n$ a počet funkčných hodnôt, ktorým je určený aj počet logických funkcií je $C_t = 2^C$.

Prehľad kombinačných logických funkcií

1. **Negácia** (**NOT**) je taká funkcia jednej premennej, v ktorej má závislé premenná Y vždy opačnú hodnotu, ako nezávisle premenná A. Výrok je vyjadrený logigkou spojkou „**nie je pravda, že platí**“. V technickej praxi je to situácia, keď činnosť jedného člena je určená nečinnosťou druhého člena.

$$Y = \bar{A}$$

algebraický zápis negácie

A	Y
0	1
1	0

pravdivostná tabuľka negácie

2. **Konjunkcia** (**AND**) alebo logický súčin je taká funkcia dvoch premenných A, B, kde nezávisle premenná Y nadobúda hodnotu 1 iba vtedy, ak A a B majú súčasne hodnotu 1. V ostatbých prípadoch nadobúda Y hodnotu 0. Na vytvorenie uvedeného zloženého výroku používame logickú spojku „**a**“ alebo „**i**“.

$$Y = A \,.\, B$$
$$Y = A \wedge B$$

algebraický zápis konjunkcie

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

pravdivostná tabuľka konjunkcie

3. **Disjunkcia** (**OR**) alebo logický súčet je taká funkcia dvoch premenných A, B, kde závisle premenná Y nadobúda hodnotu 1 vtedy, ak je buď A alebo B alebo A aj B súčasne rovné 1. Na vytvorenie uvedeného zloženého výroku používame logickú spojku „**alebo**“.

$$Y = A + B$$
$$Y = A \vee B$$

algebraický zápis disjunkcie

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

pravdivostná tabuľka disjunkcie

4. **Shefferova funkcia** (**NAND**) je taká funkcia dvoch premených A, B, kde závisle premenná Y nadobúda hodnotu 0 iba vtedy, ak majú A i B súčasne hodnotu 1. V ostatných prípadoch nadobúda premenná Y hodnotu 1. Na vytvorenie uvedeného výroku používame logickú spojku „ **a ak nie je**“.

$$Y = \overline{A.B}$$
$$Y = A \uparrow B$$

algebraický zápis Shefferovej funkcie

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

pravdivostná tabuľka Shefferovej funkcie

5. **Pierceova funkcia** (**NOR**) je taká funkcia dvoch premenných A, B, kde závisle premenná Y nadobúda hodnotu 1 iba vtedy, ak majú súčasne A i B hodnotu 0. V ostatných prípadoch nadobúda premenná Y hodnotu 0. Na vytvorenie uvedeného výroku používame logickú spojku „**ani**“.

$$Y = \overline{A + B}$$
$$Y = A \downarrow B$$

algebraický zápis Pierceovej funkcie

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

pravdivostná tabuľka Pierceovej funkcie

6. **Totožnosť** (**ekvivalencia**) je taká funkcia dvoch premenných A, B kde závisle premenná Y nadobúda hodnotu 1 iba vtedy, ak majú súčasne A aj B zhodné (totožné) hodnoty. Na vytvorenie uvedeného výroku používame logickú spojku „**vtedy, keď**".

$$Y = A \equiv B$$
$$Y = A <=> B$$

algebraický zápis ekvivalencie

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

pravdivostná tabuľka ekvivalencie

7. **Rôznoznačnosť** (**nonekvivalencia**) je taká funkcia dvoch premenných A, B, kde závisle premenná Y nadobúda hodnotu 1 iba vtedy, ak majú súčasne A aj B rôzne hodnoty. Na vytvorenie uvedeného výroku používame logickú spojku „**buď..., alebo**".

$$Y = A \neq B$$
$$Y = A <\neq> B$$

algebraický zápis nonekvivalencie

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

pravdivostná tabuľka nonekvivalencie

8. **Implikácia** je taká funkcia dvoch premenných A, B, kde závisle premenná Y nadobúda hodnotu 0 iba vtedy, ak platí, že A má hodnotu 1 a súčasne B hodnotu 0. V ostatných prípadoch nadobúda premenná Y hodnotu 1. Na vytvorenie uvedeného výroku používame logickú spojku „**ak...., potom**".

$$Y = A \rightarrow B$$

algebraický zápis nonekvivalencie

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

pravdivostná tabuľka nonekvivalencie

Formy zápisu logických funkcií

Najpoužívanejšie formy zápisu logických funkcií sú:

- algebraický zápis
- pravdivostná tabuľka
- Karnaughova mapa

V predošlých statiach sme sa už stretli s algebraickými zápismi a pravdivostnými tabuľkami pri základných logických operáciách (logický súčet, logický súčin, negácia).

Teraz zovšeobecníme tvorbu algebraického výrazu z pravdivostnej tabuľky a používanie Karnaughovej mapy.

Zápis funkcie do pravdivostnej tabuľky

Logickú funkciu zapisujeme do pravdivostnej tabuľky tak, že hodnoty vstupných veličín (nezávisle premenných) prostredníctvom algebraického výrazu reprezentujú výstupnú veličinu (závisle premennú). Vstupné veličiny zapíšeme do ľavej strany tabuľky a výstupnú veličinu zapíšeme do pravej strany tabuľky. Počet stĺpcov tabuľky zodpovedá počtu vstupných veličín a výstupnej veličiny. Počet riadkov zodpovedá počtu kombinácií vstupov t.j. $N = 2^n$ Riadky v pravdivostnej tabuľke sú očíslované poradovými číslami, ktoré nazývame **stavové indexy**.

Príklad:

Vytvorte pravdivostnú tabuľku z daných logických funkcií:

$$Y_1 = (A + B + C).(A + \bar{B} + C).(\bar{A} + B + \bar{C})$$
$$Y_2 = (\bar{A}.\bar{B}.C) + (A.B.\bar{C}) + (A.\bar{B}.\bar{C})$$

Riešenie:

Obidve funkcie môžeme pre zjednodušenie zapísať do jednej tabuľky. Vytvoríme teda tabuľku so šiestimi stĺpcami (stavové indexy, premenné A, B, C a výstupné hodnoty Y_1, Y_2) a ôsmimi riadkami ($N = 2^3$).

Do stĺpcov pre premenné A, B, C vpíšeme všetky kombinácie možných vstupných hodnôt (0 a 1). Do stĺpcov pre výstupy Y1 a Y2 vpíšeme výstupnú hodnotu danej funkcie pri daných hodnotách nezávisle premenných pre každý riadok tabuľky (nezávisle premenné dosadíme do funkcie a pomocou základných operácií boolovej algebry určíme výslednú hodnotu).

N	A	B	C	Y_1	Y_2
0	0	0	0	0	0
1	0	0	1	1	1
2	0	1	0	0	0
3	0	1	1	1	0
4	1	0	0	1	1
5	1	0	1	0	0
6	1	1	0	1	1
7	1	1	1	1	0

Výpis funkcie z pravdivostnej tabuľky

Zo zadanej pravdivostnej tabuľky sa najčastejšie vypisuje funkcia ako logický súčet základných logických súčinov. **Základný logický súčin (minterm)** je súčin všetkých nezávisle premenných, pričom vstupné premenné, ktoré majú stav 0, sa vyjadrujú negované. Vyberáme len tie súčiny z tých riadkov, kde hodnota funkcie nadobúda stav 1.

Niekedy môže byť výhodnejšie (napríklad ak je výstupná hodnota funkcie vo väčšine riadkov tabuľky rovná 0) zapísať funkciu v tvare logického súčinu základných logických súčtov. **Základný logický súčet (maxterm)** je súčet všetkých nezávisle premenných, pričom vstupné premenné, ktoré majú stav 1, sa vyjadrujú v negovanom stave. Vyberáme len tie súčty z tých riadkov, kde hodnota funkcie nadobúda stav 0.

Príklad:

Zapíšte logickú funkciu danú pravdivostnou tabuľkou v algebraickom tvare použitím súčtu základných logických súčinov a súčinu základných logických súčtov.

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

Riešenie:

Pre súčet základných logických súčinov z tabuľky vypíšeme kombinácie vstupných premenných z tých riadkov, kde je hodnota funkcie logická 1. Kombináciám nezávisle premenných v týchto riadkoch zodpovedajú logické súčiny:

- riadok č. 2: $A = 0,\ B = 1,\ C = 0$ - základný logický súčin: $\bar{A} . B . \bar{C}$
- riadok č. 3: $A = 0,\ B = 1,\ C = 1$ - základný logický súčin: $\bar{A} . B . C$
- riadok č. 6: $A = 1,\ B = 1,\ C = 0$ - základný logický súčin: $A . B . \bar{C}$
- riadok č. 7: $A = 1,\ B = 1,\ C = 1$ - základný logický súčin: $A. B. C$

Výsledná logická funkcia potom bude súčet týchto základných logických súčinov:

$$Y = \bar{A}. B. \bar{C} + \bar{A}. B. C + A. B. \bar{C} + A. B. C$$

Pre súčin základných logických súčtov vypíšeme z tabuľky tie riadky, v ktorých má funkcia hodnotu 0:

- riadok č. 0: $A = 0, B = 0, C = 0$ - základný logický súčet: $A + B + C$
- riadok č. 1: $A = 0, B = 0, C = 1$ - základný logický súčet: $A + B + \bar{C}$
- riadok č. 4: $A = 1, B = 0, C = 0$ - základný logický súčet: $\bar{A} + B + C$
- riadok č. 5: $A = 1, B = 0, C = 1$ - základný logický súčet: $\bar{A} + B + \bar{C}$

Výslednú logickú funkciu zapíšeme ako súčin týchto základných logických súčtov:

$$(A + B + C). (A + B + \bar{C}). (\bar{A} + B + C). (\bar{A} + B + \bar{C})$$

Karnaughova mapa, zápis logickej funkcie do Karnaughovej mapy

Prehľadný zápis všetkých stavov logickej funkcie umožňuje **Karnaughova mapa**.

Keď porovnáme pravdivostné tabuľky, napríklad funkcie troch premenných, zistíme, že ľavá strana, kde sú jednotlivé kombinácie nezávisle premenných, je vždy rovnaká a neprináša žiadne nové informácie. Každá vstupná premenná ma v polovičnom počte z celkového 2^n riadkov hodnotu 0 a v druhej polovici hodnotu 1.

Každej kombinácií nezávisle premenných je v Karnaughovej mape pridelený jeden štvorec, do ktorého sa vpisuje stav výstupnej funkcie. Teda počet štvorcov v Karnaughovej mape sa musí rovnať počtu riadkov v pravdivostnej tabuľke a naopak, každému riadku pravdivostnej tabuľky zodpovedá jeden štvorec v Karnaughovej mape.

Priradenie vstupných premenných pre jednotlivé riadky a stĺpce sa vykonáva:

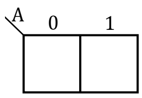
- **algebraickým označením** - po obvode Karnaughovej mapy sa napíšu hodnoty nezávisle premenných, ktoré prislúchajú danému riadku/stĺpcu. Tento spôsob sa používa len zriedkavo.
- **svorkami** - po okrajoch Karnaughovej mapy sa vyznačia čiarami označenými príslušnou nezávisle premennou tie riadky alebo stĺpce, pre ktoré má daná nezávisle premenná hodnotu 1.

Karnaughove mapy sa takto môžu označovať rôznymi spôsobmi, no treba dodržať tieto pravidlá:

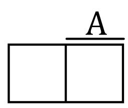
- každá premenná musí mať polovicu štvorčekov Karnaughovej mapy priradenej k hodnote 1 a druhú polovicu hodnote 0.
- v mape musí existovať štvorec, v ktorom majú všetky nezávisle premenné hodnotu 1 a štvorec v ktorom majú všetky nezávisle premenné hodnotu 0.

Karnaughova mapa jednej nezávislej premennej:

Z predchádzajúceho vyplýva, že Karnaughova mapa jednej nezávisle premennej bude mať dve políčka (dva riadky v pravdivostnej tabuľke).



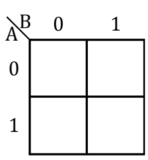
algebraické označenie



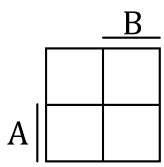
označenie svorkami

Karnaughova mapa dvoch nezávisle premenných:

Karnaughova mapa dvoch nezávisle premenných bude mať štyri políčka (štyri riadky v pravdivostnej tabuľke).



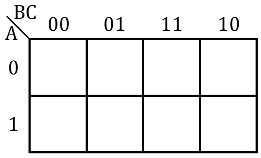
algebraické označenie



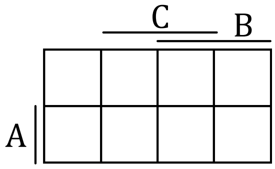
označenie svorkami

Karnaughova mapa troch nezávisle premenných:

Karnaughova mapa troch nezávisle premenných bude mať osem políčok (osem riadkov v pravdivostnej tabuľke).



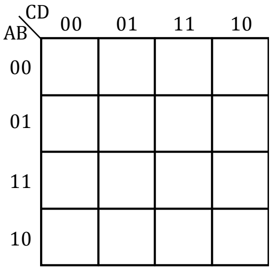
algebraické označenie



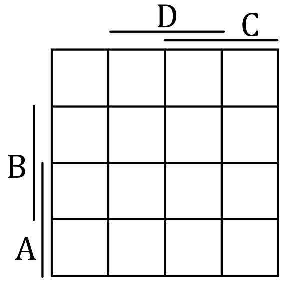
označenie svorkami

Karnaughova mapa štyroch nezávisle premenných:

Karnaughova mapa štyroch nezávisle premenných bude mať šestnásť políčok (šestnásť riadkov v pravdivostnej tabuľke).



algebraické označenie



označenie svorkami

Ako zapísať logickú funkciu do Karnaughovej mapy si ukážeme na nasledujúcom príklade.

Príklad:

Zapíšte funkciu danú pravdivostnou tabuľkou do Karnaughovej mapy:

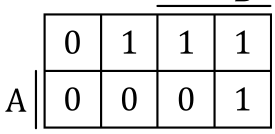
N	A	B	C	Y
0	0	0	0	0
1	0	0	1	1
2	0	1	0	1
3	0	1	1	1
4	1	0	0	0
5	1	0	1	0
6	1	1	0	1
7	1	1	1	0

Riešenie:

Tabuľka má 8 riadkov, teda Karnaughova mapa bude mať 8 štvorcov.

Výstupnú hodnotu funkcie Y pre daný riadok pravdivostnej tabuľky zapíšeme do prislúchajúceho štvorca Karnaughovej mapy, ktorý je na jej obvode označený tak, že toto označenie zodpovedá hodnotám nezávisle premenných v danom riadku tabuľky.

A \ BC	00	01	11	10
0	0	1	1	1
1	0	0	0	1



Výpis logickej funkcie z Karnaughovej mapy

Logickú funkciu znázornenú v Karnaughovej mape môžeme algebraicky zapísať podobne ako z pravdivostnej tabuľky.

Ak hovoríme a tom, že si všímame štvorčeky, v ktorých je stav výstupne funkcie 1, funkciu popíšeme súčtom súčinov vstupných premenných. Hovoríme, že vytvárame **Úplnú normálnu disjunktívnu formu** (**ÚNDF**) funkcie Y.

Ak si všímame len štvorčeky Karnaughovej mapy kde funkcia Y nadobúda hodnotu 0, funkciu popíšeme súčinom súčtov negácií vstupných premenných. Hovoríme, že vytvárame **Úplnú normálnu konjunktívnu formu** (**ÚNKF**) funkcie Y.

Príklad:

Zapíšte algebraicky funkciu danú Karnaughovou mapou.

		B
	1	0
A	0	1

Riešenie:

1. ÚNDF

Funkciu vypíšeme podobne ako pri pravdivostnej tabuľke pomocou súčtu základných logických súčinov. Vypisujeme štvorčeky, v ktorých má funkcia hodnotu 1:

- 1. štvorček: $A = 0, B = 0$ - základný logický súčin: $\bar{A}.\bar{B}$
- 4. štvorček: $A = 1, B = 1$ - základný logický súčin: $A.B$
- výsledná funkcia: $\bar{A}.\bar{B} + A.B$

2. ÚNKF

Tentoraz pozeráme na štvorčeky, v ktorých má funkcia hodnotu 0 a používame zápis pomocou súčinu základných logických súčtov.

- 2. štvorček: $A = 0, B = 1$ - základný logický súčet: $A + \bar{B}$
- 3. štvorček: $A = 1, B = 0$ - základný logický súčet: $\bar{A} + B$
- výsledná funkcia: $(A + \bar{B}).(\bar{A} + B)$

Minimalizácia logických funkcií

Pri minimalizácií logickej funkcie bude našou snahou postupovať tak, aby sme dostali čo najjednoduchšie vyjadrenie logickej funkcie.

Algebraická minimalizácia

Algebraická minimalizácie logických funkcií predstavuje algebraické upravovanie logického výrazu podľa zákonov a pravidiel Boolovej algebry. Výsledkom je funkcia s rovnakým správaním, vyjadrená však jednoduchšie, čo pri realizácii znamená jednoduchšiu štruktúru logického obvodu.

Príklad:

Upravte funkciu

$$Y = A.B + A.\bar{B} + \bar{A}.\bar{B}$$

Riešenie:

z prvého a druhého súčinu môžeme vybrať A pred zátvorku:

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

na výraz $B + \bar{B}$ môžeme použiť zákon doplnku, po úprave dostaneme:

$$Y = A + \bar{A}.\bar{B}$$

nakoniec použijeme zákon absorpcie negácie a dostaneme výsledný výraz:

$$Y = A + \bar{B}$$

Minimalizácia pomocou Karnaughovej mapy

Máme takúto Karnaughovu mapu:

		C	C, B	B
	0	0	0	0
A	1	1	0	0

Pomocou ÚNDF z nej môžeme vypísať logickú funkciu a algebraicky ju minimalizovať:

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

Zistíme, že sa nám však ponúka aj možnosť minimalizácie pomocou Karnaughovej mapy. Pretože dve jednotky spolu susedia, môžeme pristúpiť okamžite k napísaniu jedného výrazu pre obe políčka a tak dostaneme priamo minimalizovaný výraz. Uvedený postup môžeme rozšíriť na väčší počet susedných políčok. Ak robíme minimalizáciu funkcie pomocou jednotiek, hovoríme o **Normálnej disjunktívnej forme – NDF**, alebo o **Normálnej konjunktívnej forme – NKF**, ak minimalizujeme z núl.

Postup pri minimalizácií z Karnaughovej mapy pre NDF (NKF) je nasledovný:

1) Jednotky (nuly) v mape uzatvárame pomocou slučiek, ktoré obsahujú 2^n (teda jedno, dve, štyri, osem, atď.) susedných políčok. Pritom si treba zapamätať, že čím väčšie slučky vytvoríme, tým jednoduchšie budú zodpovedajúce algebraické výrazy.
2) Karnaughovu mapu si treba pri vytváraní slučiek predstaviť napr. ako guľu, pretože do jednej slučky môžu patriť aj políčka z náprotivných strán Karnaughovej mapy. Do jednej slučky môžu patriť aj políčka v rohoch Karnaughovej mapy.

3) Každá jednotka (nula) musí byť zahrnutá aspoň v jednej slučke.
4) Slučky sa môču navzájom pretínať.
5) Každej slučke zodpovedá iba jeden výraz, ktorý neobsahuje tie premenné, pre ktoré daná slučka nadobúda hodnotu 0 **aj** 1.
6) Slučky s jednotkami zodpovedajú výslednému výrazu v podobe súčtu súčinov, zatiaľ čo slučky s nulami výrazu v podobe súčinu súčtov.

Na predchádzajúcej mape budú slučky vyzerať nasledovne:

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

Daná slučka obsahuje výraz $Y = A.\overline{B}$, keďže v celej slučke má premenná A hodnotu logickej 1 a premenná B hodnotu logickej 0. Vidíme, že výraz získaný minimalizáciou z Karnaughovej mapy je zhodný s výsledkom algebraickej minimalizácie.

Príklad:

Minimalizujte funkciu danú Karnaughovou mapou:

		D	D, C	C
	1	0	0	1
B	1	1	1	1
B, A	0	1	1	0
A	1	0	0	1

Riešenie pomocou NDF:

Na mape vytvoríme slučky obsahujúce jednotky. Slučky robíme čo najväčšie pre zjednodušenie výrazu.

		D	D, C	C
	1	0	0	1
B	1	1	1	1
B, A	0	1	1	0
A	1	0	0	1

Vypíšeme jednotlivé slučky ako súčet základných súčinov:

$$Y = B.D + \overline{B}.\overline{D} + \overline{A}.B$$

Riešenie pomocou NKF:

Vytvoríme slučky z núl. Opäť sa snažíme vytvoriť čo najväčšie slučky.

		D	D, C	C
	1	0	0	1
B	1	1	1	1
B, A	0	1	1	0
A	1	0	0	1

Vypíšeme jednotlivé slučky ako súčin základných súčtov, pričom nezabúdame, že výstupy sú negované:

$$\boldsymbol{Y = (B + \overline{D}).(\overline{A} + \overline{B} + D)}$$

Príklady na precvičovanie:

Nasledujúce funkcie zapíšte do Karnaughovej mapy, následne ich minimalizujte algebraicky a pomocou Karnaughovej mapy za použitia NDF a NKF.

$$\boldsymbol{Y = \overline{A}.\overline{B}.\overline{C} + \overline{A}.\overline{B}.C + \overline{A}.B + C}$$

$$\boldsymbol{Y = (A + \overline{B} + C).(A + \overline{C}).(A + C)}$$

Kontaktová realizácia logickej funkcie

Posledným bodom postupu pri návrhu kombinačného logického obvodu je jeho schéma, ktorá je podkladom pre jeho technickú realizáciu. Východiskom pre nakreslenie schémy je minimalizovaný algebraický výraz. Skôr ako začneme schému kresliť, treba vopred zvážiť, aké technické prostriedky (logické členy) použijeme na jej realizáciu.

Zatiaľ sa budeme zaoberať realizáciou logického obvodu pomocou kontaktových členov (relé, stýkače) – nakreslíme **kontaktovú schému.**

Kontaktová schéma je historicky najstaršia, tak ako aj prostriedky na jej realizáciu, ale používa sa dodnes. Ako sme hovorili predtým, vychádzame z minimalizovaného algebraického výrazu. Pri podrobnom preskúmaní tohto výrazu zistíme, že obsahuje nezávislé premenné v základných logických funkciách.

Kontaktové zapojenia základných logických funkcií sme si uviedli v stati „Základné operácie Boolovej algebry“ ako riadkové schémy. Tieto zapojenia využijeme teraz pri návrhu kontaktového zapojenia logickej funkcie.

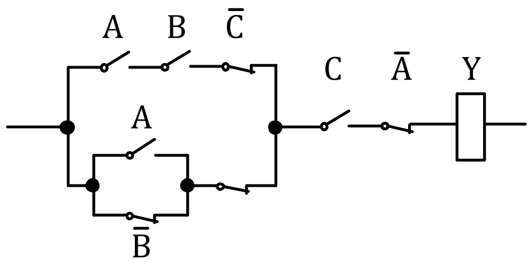
Príklad:

Navrhnite kontaktovú realizáciu funkcie

$$Y = \left(A.B.\bar{C} + \overline{A + \bar{B}}\right).\bar{A}.C$$

Riešenie:

V zátvorke je medzi dvoma výrazmi $\boldsymbol{A.B.\bar{C}}$ a $\boldsymbol{A + \bar{B}}$ logický súčet, teda tieto dva výrazy zakreslíme paralelne. Negovaný súčet zakreslíme tak, že za súčet pridáme negátor.



Príklady:

Navrhnite kontaktové realizácie pre tieto logické funkcie:

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

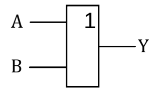
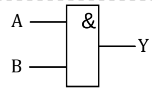
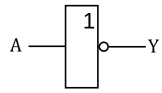
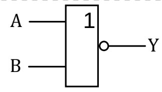
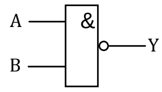
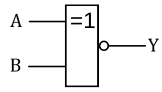
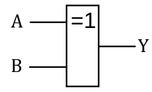
$$Y = (A.B + \bar{A} + B.\bar{C}).(A + \overline{\bar{C} + B})$$

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

Realizácia logickej funkcie logickými členmi

Aj v tomto prípade vychádzame z minimalizovaného algebraického výrazu. Pri podrobnom preskúmaní tohto výrazu zistíme, že obsahuje nezávislé premenné v základných logických funkciách. Z toho dôvodu najprv spomenieme značky logických členov.

Základné logické členy (kombinačné):

Názov	Skrátený názov	Výraz	Symbol
Logický súčet	OR	$Y = A + B$	
Logický súčin	AND	$Y = A.B$	
Logická negácia	NOT	$Y = \bar{A}$	
Negovaný logický súčet	NOR	$Y = \overline{A + B}$	
Negovaný logický súčin	NAND	$Y = \overline{A.B}$	
Ekvivalencia (rovnoznačnosť)	XNOR	$Y = \overline{A \oplus B}$	
Non ekvivalencia (rôznoznačnosť)	XOR	$Y = A \oplus B$	

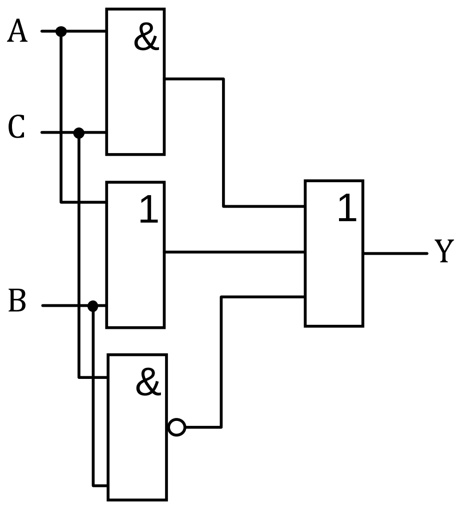
Pomocou týchto logických členov môžeme realizovať logické funkcie. Nezabúdajme, že realizácia sa uskutočňuje z minimalizovaného výrazu.

Príklad:

Realizujte danú funkciu logickými členmi

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

Riešenie:



Príklady:

Logickými členmi realizujte funkcie:

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

$$Y = A.\bar{B}.C + A.\bar{C} + \overline{B.C}$$

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

			C	B
	1	0	1	1
A	1	1	0	0

V praxi sa využívajú najmä integrované obvody pozostávajúce z členov NAND alebo z členov NOR. Pre použitie takýchto obvodov je potrebné si funkciu, ktorú chceme realizovať, upraviť do tvaru vhodného na realizáciu tými členmi, ktoré máme k dispozícií. Na úpravy sa využívajú hlavne de Morganove zákony.

Realizácia logickej funkcie pomocou členov NAND

Máme funkciu

$$Y = A.\bar{B}.C + A.B.\bar{C} + \overline{B.C}$$

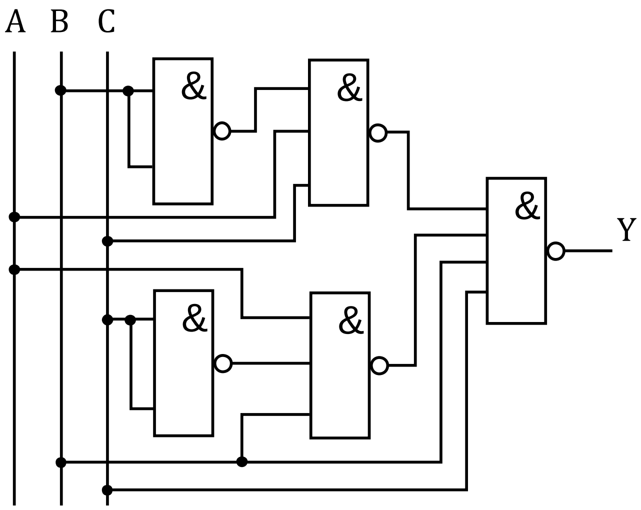
a chceme ju realizovať členmi NAND. Vidíme však, že ak by sme túto funkciu neupravili, potrebovali by sme aj iné logické členy. Preto funkciu upravíme de Morganovým zákonom nasledovným spôsobom:

$$Y = \overline{\overline{A.\bar{B}.C + A.B.\bar{C} + \overline{B.C}}}$$

$$Y = \overline{\overline{A.\bar{B}.C}.\overline{A.B.\bar{C}}.\overline{\overline{B.C}}}$$

$$Y = \overline{\overline{A.\bar{B}.C}.\overline{A.B.\bar{C}}.B.C}$$

Na realizáciu takto upravenej funkcie nám vystačia členy NAND.



Všimnite si realizáciu člena NOT – hradlo (člen) NAND má skratované vstupy.

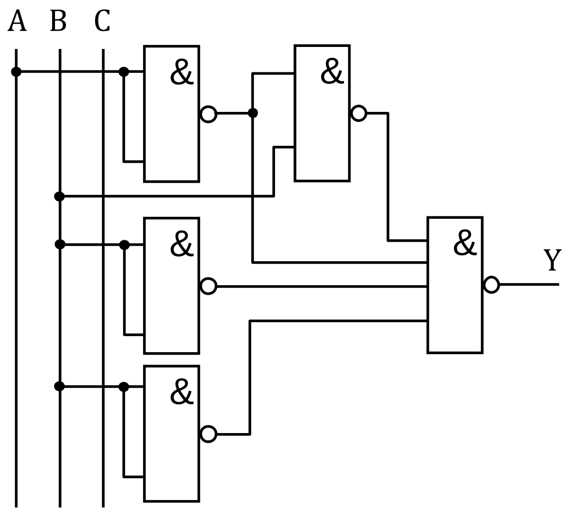
Teraz máme funkciu

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

V tomto prípade už máme požadovanú operáciu (súčin) medzi základnými súčtami. Takže tentoraz potrebujeme zmeniť znamienka len medzi jednotlivými premennými, čo by sme predošle popísaným spôsobom nedosiahli (všetky znamienka by sa zmenili na súčet). Preto použijeme de Morganov zákon nasledovným spôsobom:

$$Y = \left(\overline{\overline{\overline{A + B + C}}}\right).\left(\overline{A + \overline{\overline{B}}}\right)$$
$$Y = \left(\overline{\overline{\bar{A}.\bar{B}.\bar{C}}}\right).\left(\overline{\bar{A}.B}\right)$$
$$Y = (\bar{A}.\bar{B}.\bar{C}).\left(\overline{\bar{A}.B}\right)$$
$$Y = \overline{\overline{\bar{A}.\bar{B}.\bar{C}.\left(\overline{\bar{A}.B}\right)}}$$

Realizácia tejto funkcie:



Realizácia logickej funkcie pomocou členov NOR

Podobným spôsobom môžeme funkciu upraviť aj na realizáciu hradlami NOR. Ako príklad použijeme tie isté funkcie.

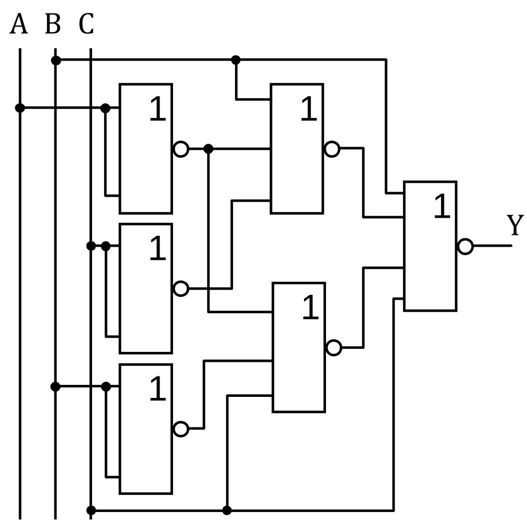
Funkciu

$$Y = A.\bar{B}.C + A.B.\bar{C} + \overline{B.C}$$

potrebujeme upraviť tak, aby sa zmenili znamienka medzi jednotlivými premennými:

$$Y = \overline{\overline{A.\bar{B}.C}} + \overline{\overline{A.B.\bar{C}}} + \overline{\overline{\overline{B.C}}}$$
$$Y = \overline{\bar{A} + \bar{\bar{B}} + \bar{C}} + \overline{\bar{A} + \bar{B} + \bar{\bar{C}}} + \overline{\overline{B + C}}$$
$$Y = \overline{\overline{\overline{\bar{A} + B + \bar{C}} + \overline{\bar{A} + \bar{B} + C} + B + C}}$$

Realizácia môže byť nasledovná:



V druhej funkcii

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

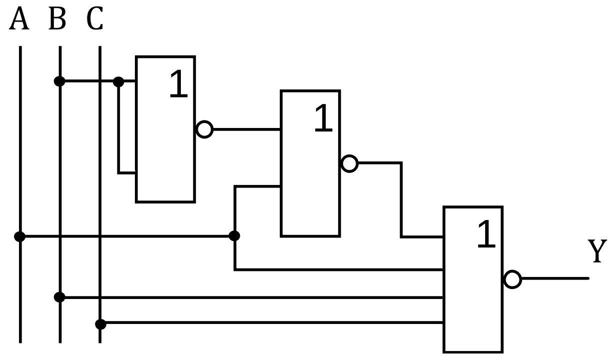
je pre túto realizáciu potrebné zmeniť znamienka medzi zátvorkami:

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

$$Y = \overline{\overline{\overline{A + B + C}} + \overline{A + \bar{B}}}$$

$$Y = \overline{A + B + C + \overline{A + \bar{B}}}$$

Realizujeme túto funkciu:



Príklad na precvičenie:

V stati **Základné operácie Boolovej algebry** sme uviedli pravdivostné tabuľky ekvivalencie a nonekvivalencie (teda XNOR a XOR). Navrhnite pomocou nich realizáciu pre XOR a XNOR členmi NAND a NOR.

2. Kombinačné logické obvody

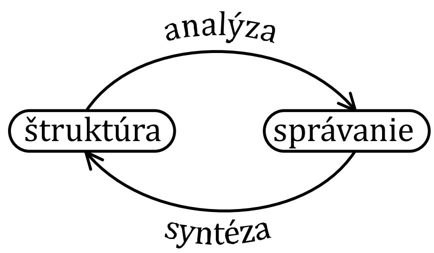
Analýza a syntéza Kombinačných logických obvodov (KLO)

Každý logický obvod ako kybernetický systém charakterizuje:

- správanie logického obvodu – môže byť určené logickou funkciou, pravdivostnou tabuľkou alebo Karnaughovou mapou
- štruktúra logického obvodu vyjadrená schémou zapojenia

Podľa toho, čo je začiatočným momentom a čo výsledkom činnosti, rozoznávame dva základné procesy (pstupy):

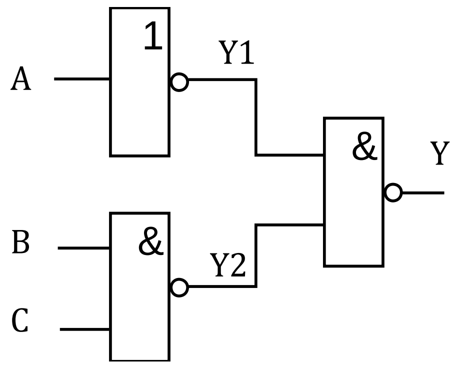
- **analýza** – rozbor činnosti už realizovaného alebo navrhnutého obvodu
- **syntéza** – postup, pri ktorom zo zadaného správania technologického systému a požiadaviek na jeho riadenie navrhujeme konkrétny logický obvod, realizujúci tieto požiadavky



Postup pri analýze:

- podľa danej štruktúry, teda schémy, určíme výstupné funkcie jednotlivých členov
- podľa vzájomných väzieb medzi jednotlivými členmi a vstupnými veličinami postupným dosadzovaním určíme výsledný výraz – algebraické vyjadrenie výstupnej funkcie
- pre výstupnú funkciu zostavíme pravdivostnú tabuľku a Karnaughovu mapu, čím je správanie logického obvodu určené

Príklad:



Jednotlivé členy majú výstup:

$$Y1 = \bar{A}$$
$$Y2 = \overline{B.C}$$
$$Y = \overline{Y1.Y2} = \overline{\bar{A}.\overline{B.C}}$$

Karnaughova mapa:

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

Postup pri syntéze:

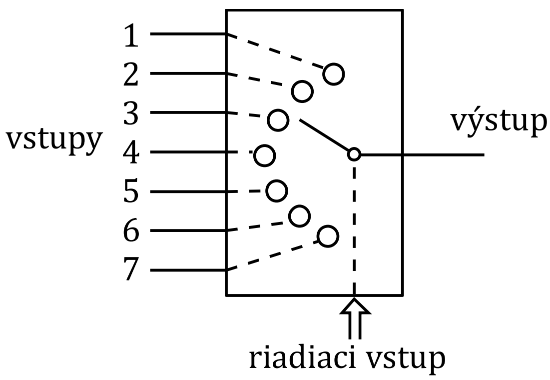
- **Etapa systémového návrhu:** Na základe znalostí celého technologického procesu ako riadeného objektu a požadovaného algoritmu riadenia, znalostí blokovacích podmienok a signalizácie si stanovíme cieľ riadiaceho procesu. Ak máme možnosť, urobíme dekompozíciu – rozdelenie zložitého systému na jednoduchšie časti, ktoré sú schopné samostatného riadenia. Z cieľovej funkcie čiastkových podsystémov formulujeme ich požadované správanie a presnú definíciu vstupov a výstupov. Správanie možno určiť pravdivostnou tabuľkou, Karnaughovou mapou, časovými diagramami alebo slovným opisom.
- **Etapa logického návrhu:** Zo zadaného správania riadiaceho podsystému vyjadríme v algebraickej forme logickú funkciu a podľa možnosti ju minimalizujeme. Zvolíme si vhodné logické členy a funkciu realizujeme – dostávame štruktúru (schému) systému.

Druhy KLO

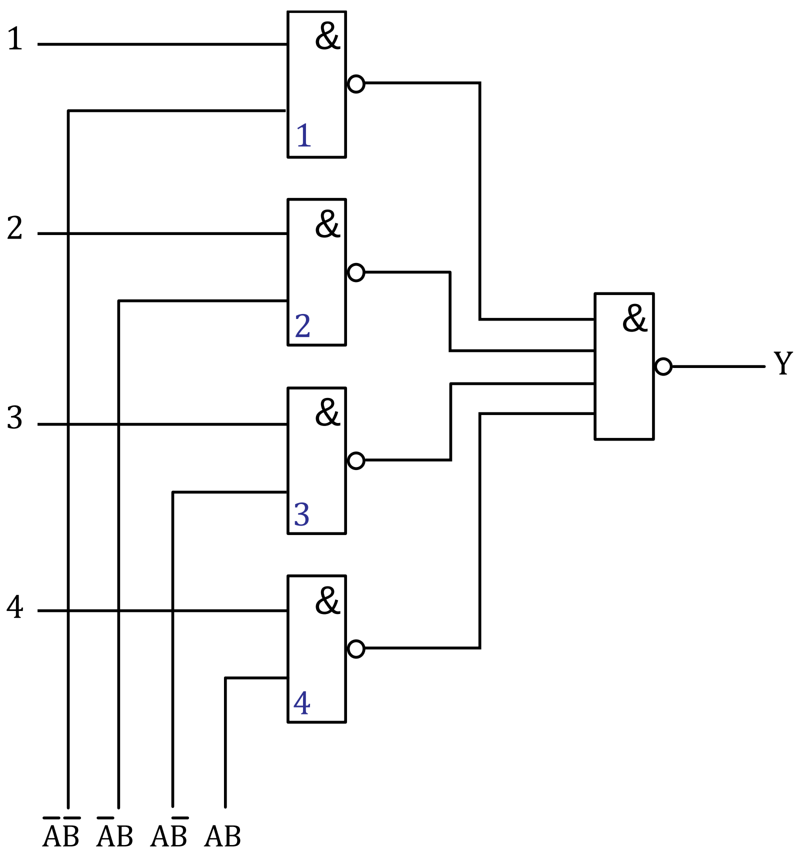
Multiplexor

Multiplexor je elektronický obvod, ktorý sa používa k výberu a k vedeniu akéhokoľvek s určitého počtu vstupných signálov na jednoduchý výstup.

Najjednoduchšia forma – viacpozičný prepínač:



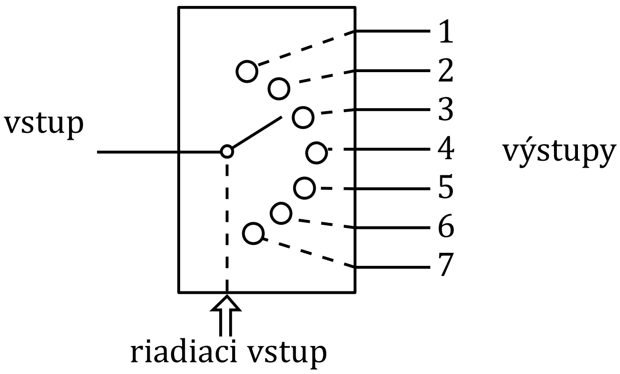
Štvorvstupový multiplexor:



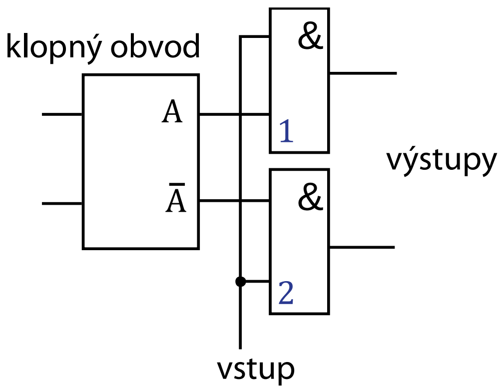
A	B	Otvorený
0	0	1
0	1	2
1	0	3
1	1	4

Demultiplexor

Je to v podstate obrátený multiplexor – jeden vstup sa prepínana viacero výstupov.



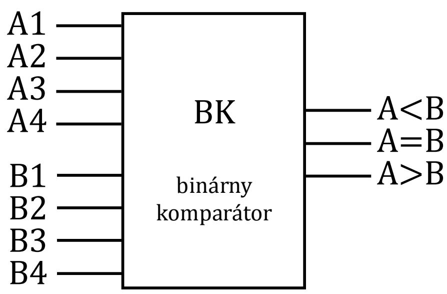
Jednoduchý dvojvýstupový demultiplexor:



A	Otvorený
0	2
1	1

Komparátor

Komparátor je KLO pre porovnávanie hodnôt na vstupe, pričom vyhodnocuje 3 základné stavy:



Najjednoduchší komparátor je logický člen XNOR – vracia log. 1 ak sa hodnoty vstupov rovnajú.

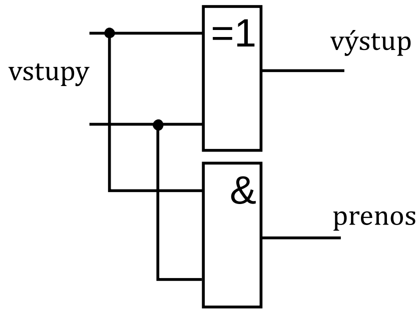
Sčítačka

Je to KLO, ktorý sčíta 2 binárne čísla. Použitie – číslicové počítače, elektronické kalkulačky, mikroprocesory a iné zariadenia využívajúce matematické operácie.

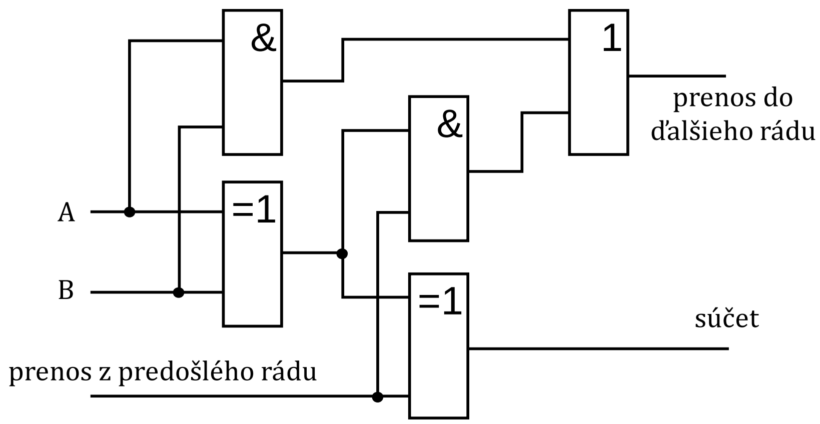
$$\begin{array}{cccc} 0 & 0 & 1 & 1 \\ +0 & +1 & +0 & +1 \\ \hline 0 & 1 & 1 & 10 \end{array}$$

prenos (carry)

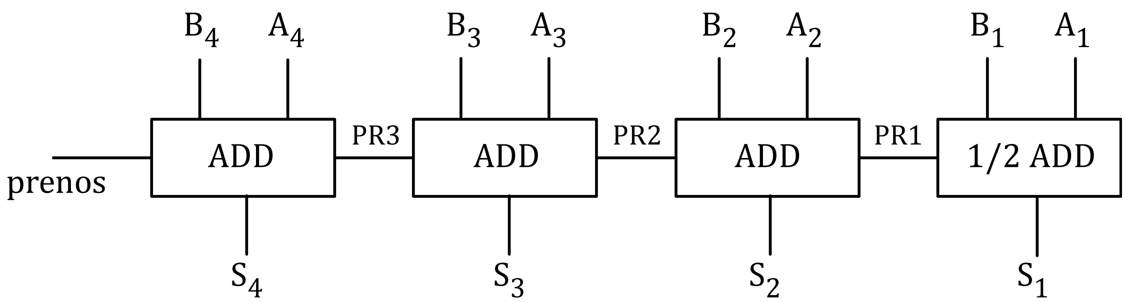
Ak by sme neuvažovali prenos do vyššieho rádu, vidíme, že takéto spočítavanie môžeme vyriešiť logickým členom NOR (výsledok je 1 ak vstupy nie sú zhodné). Prenos môžeme vyriešiť členom AND (prenos nastáva iba ak sú oba vstupy 1). Kombináciou týchto členov dostaneme Polovičnú sčítačku:



Úplná sčítačka:



Bloková schéma sčítačky:

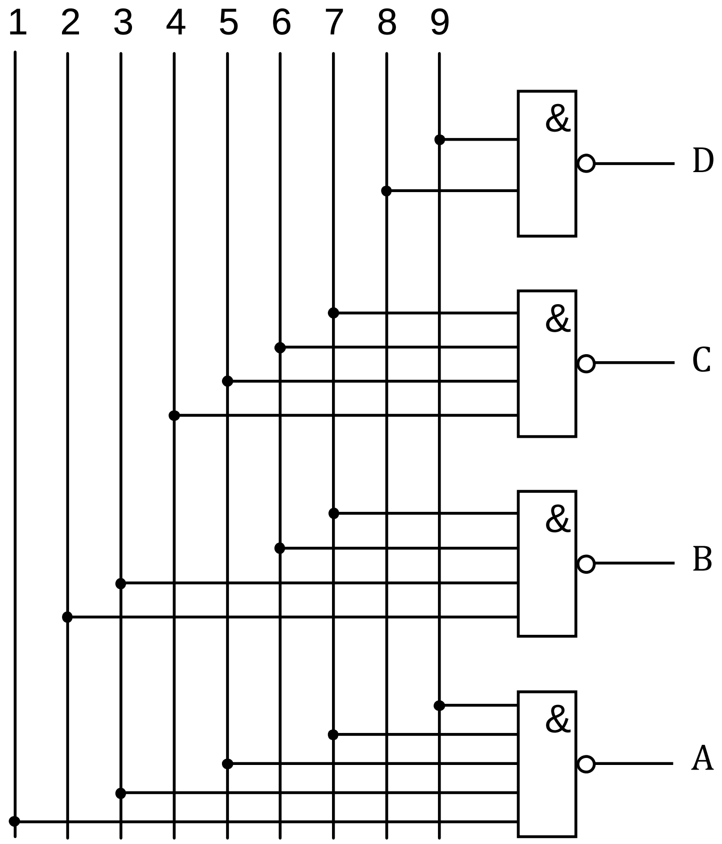


1/2 ADD - polovičná sčítačka
ADD - úplná sčítačka
PR - prenos
A, B - vstupy
S - súčet

Kóder

Je to KLO, ktorý prijíma jeden alebo viac vstupov a generuje niekoľkobitový binárny výstupný kód.

Kóder desiatkového vstupu klávesnice na kód BCD:



Princíp: po stlačení klávesy 6 sa otvorí hradlo B a C. Ostatné ostávajú zavreté. Ak si tento výstup prepíšeme do binárnej sústavy, dostaneme kód:

2^3	2^2	2^1	2^0
D	C	B	A
0	**1**	**1**	**0**

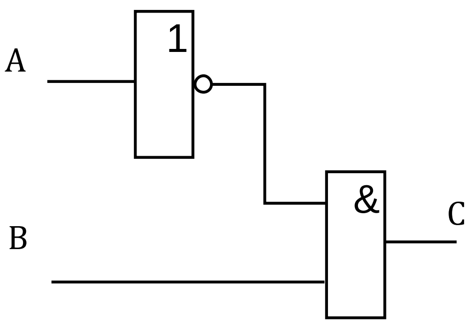
čo predstavuje číslo 6.

Dekóder

Dekóder je KLO, ktorý zisťuje prítomnosť špecifického binárneho slova. Prítomnosť tohto slova indikuje na výstupe logická 1.

Základným dekódovacím členom je AND.

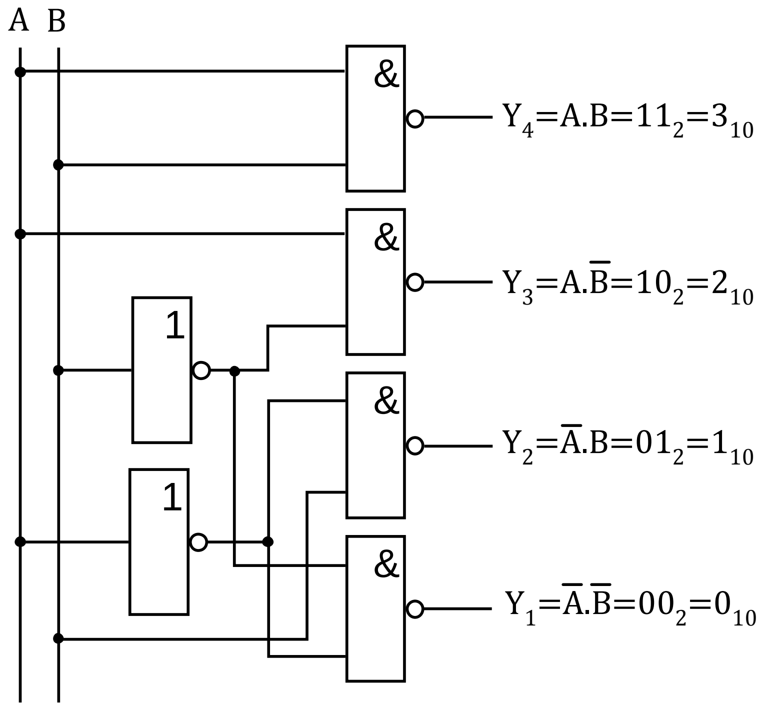
Dekóder pre vstupné číslo 01:



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

Ako vidieť v pravdivostnej tabuľke, člen AND zabezpečí, že pri tejto realizácií dostaneme na výstupe logickú jednotku len ak vstup zodpovedá požadovanému binárnemu slovu (v tomto prípade 01).

Dekóder binárneho kódu na kód jedna zo štyroch:



A	B	Y1	Y2	Y3	Y4
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1