

Úvod do informačních technologií

Jan Outrata



KATEDRA INFORMATIKY
UNIVERZITA PALACKÉHO V OLOMOUCI

přednášky

Binární logika

Počítač = počítaací stroj . . . počítání s čísly

Člověk:

- deset hodnot (deset prstů na rukách), deset symbolů (číslic, **0** až **9**)
- použití **desítkové (dekadické) poziční číselné soustavy**: číslo jako součet mocninné řady o základu (radixu) 10, zápis = posloupnost symbolů pro koeficienty řady, pozice (pořadí) symbolu určuje mocninu (řád)

$$(1024)_{10} = 1 \cdot 10^3 + 0 \cdot 10^2 + 2 \cdot 10^1 + 4 \cdot 10^0$$

- jiné číselné soustavy: dvanáctková (hodiny), šedesátková (minuty, sekundy), dvacítková (dřívější platidla) aj.

Věta (O reprezentaci přirozených čísel (včetně 0))

Libovolné přirozené číslo N (včetně 0) lze vyjádřit jako součet mocninné řady o základu $B \geq 2, B \in \mathbb{N}$:

$$N = a_{n-1} \cdot B^{n-1} + a_{n-2} \cdot B^{n-2} + \dots + a_1 \cdot B^1 + a_0 \cdot B^0,$$

kde $0 \leq a_i < B, a_i \in \mathbb{N}$ jsou koeficienty řady.

Číslo N se (v poziční číselné soustavě o základu B) zapisuje jako řetěz symbolů (číslic) S_i pro koeficienty a_i zleva v pořadí pro i od $n-1$ k 0:

$$(S_{n-1}S_{n-2} \dots S_1S_0)_B$$

Získání (hodnoty) čísla N z jeho zápisu $(S_{n-1}S_{n-2} \dots S_1S_0)_B$ postupným přičítáním:

```
 $N = a_0$   
 $B' = B$   
for  $i = 1$  to  $n - 1$  do  
     $N = N + a_i * B'$   
     $B' = B' * B$ 
```

Získání zápisu $(S_{n-1}S_{n-2} \dots S_1S_0)_B$ čísla N (dané hodnoty) postupným odečítáním:

```
 $B' = i = 1$   
while  $B' * B \leq N$  do  
     $B' = B' * B$   
     $i = i + 1$   
for  $i = i - 1$  to  $0$  do  
     $a_i = N / B'$   
     $N = N - a_i * B'$   
     $B' = B' / B$ 
```

Získání (hodnoty) čísla N z jeho zápisu $(S_{n-1}S_{n-2} \dots S_1S_0)_B$ postupným přičítáním:

```
 $N = a_0$   
 $B' = B$   
for  $i = 1$  to  $n - 1$  do  
     $N = N + a_i * B'$   
     $B' = B' * B$ 
```

Získání zápisu $(S_{n-1}S_{n-2} \dots S_1S_0)_B$ čísla N (dané hodnoty) postupným odečítáním:

```
 $B' = i = 1$   
while  $B' * B \leq N$  do  
     $B' = B' * B$   
     $i = i + 1$   
for  $i = i - 1$  to  $0$  do  
     $a_i = N / B'$   
     $N = N - a_i * B'$   
     $B' = B' / B$ 
```

$$\begin{aligned} N &= a_{n-1} \cdot B^{n-1} + a_{n-2} \cdot B^{n-2} + \dots + a_1 \cdot B + a_0 \\ &= (\dots (a_{n-1} \cdot B + a_{n-2}) \cdot B + \dots + a_1) \cdot B + a_0 \end{aligned}$$

Získání (hodnoty) čísla N z jeho zápisu $(S_{n-1}S_{n-2} \dots S_1S_0)_B$ postupným násobením:

```
 $N = a_{n-1}$   
for  $i = n - 2$  to  $0$  do  
     $N = N * B + a_i$ 
```

Získání zápisu $(S_{n-1}S_{n-2} \dots S_1S_0)_B$ čísla N (dané hodnoty) postupným dělením:

```
 $a_0 = N \bmod B$   
 $i = 1$   
while  $N \geq B$  do  
     $N = N / B$   
     $a_i = N \bmod B$   
     $i = i + 1$ 
```

$$\begin{aligned} N &= a_{n-1} \cdot B^{n-1} + a_{n-2} \cdot B^{n-2} + \dots + a_1 \cdot B + a_0 \\ &= (\dots (a_{n-1} \cdot B + a_{n-2}) \cdot B + \dots + a_1) \cdot B + a_0 \end{aligned}$$

Získání (hodnoty) čísla N z jeho zápisu $(S_{n-1}S_{n-2} \dots S_1S_0)_B$ postupným násobením:

```
 $N = a_{n-1}$   
for  $i = n - 2$  to  $0$  do  
     $N = N * B + a_i$ 
```

Získání zápisu $(S_{n-1}S_{n-2} \dots S_1S_0)_B$ čísla N (dané hodnoty) postupným dělením:

```
 $a_0 = N \bmod B$   
 $i = 1$   
while  $N \geq B$  do  
     $N = N / B$   
     $a_i = N \bmod B$   
     $i = i + 1$ 
```


$$\begin{aligned} N &= a_{n-1} \cdot B^{n-1} + a_{n-2} \cdot B^{n-2} + \dots + a_1 \cdot B + a_0 \\ &= (\dots (a_{n-1} \cdot B + a_{n-2}) \cdot B + \dots + a_1) \cdot B + a_0 \end{aligned}$$

Získání (hodnoty) čísla N z jeho zápisu $(S_{n-1}S_{n-2} \dots S_1S_0)_B$ postupným násobením:

```
 $N = a_{n-1}$   
for  $i = n - 2$  to  $0$  do  
     $N = N * B + a_i$ 
```

Získání zápisu $(S_{n-1}S_{n-2} \dots S_1S_0)_B$ čísla N (dané hodnoty) postupným dělením:

```
 $a_0 = N \bmod B$   
 $i = 1$   
while  $N \geq B$  do  
     $N = N / B$   
     $a_i = N \bmod B$   
     $i = i + 1$ 
```

- 1 Pro několik čísel zjistěte (hodnotu) čísla ze zápisů ve dvojkové, osmičkové, desítkové a šestnáctkové soustavě.
- 2 Pro několik čísel zjistěte zápis čísla (dané hodnoty) ve dvojkové, osmičkové, desítkové a šestnáctkové soustavě.

Počítač:

- první mechanické počítací stroje dekadické, tj. používající desítkovou soustavu
- mechanické součásti mající 10 stabilních stavů = deset hodnot
- elektromechanické a elektronické součásti: nejnadhěji realizovatelné **2 stabilní stavy** (relé sepnuto/rozepnuto, elektronkou či tranzistorem proud prochází/neprochází, mezi částmi integrovaného obvodu je/není napětí) = 2 hodnoty, 2 symboly (číslíce, **0** a **1**)
→ **digitální zařízení**
- použití **dvojkové (binární) poziční číselné soustavy**: číslo jako součet mocninné řady o základu 2, zápis = posloupnost symbolů pro koeficienty, pozice symbolu určuje mocninu

$$(11)_{10} = (1011)_2 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0$$

Dlaší typy dat (čísla s řádovou čárkou, znaky), odvozeny od (celých) čísel → **binární reprezentace** všech typů dat.

Počítač pro člověka:

- použití pozičních číselných soustav o základu 2^k ($k \in \mathbb{N}$):
 - **osmičkové (oktalové)**: symboly (číslíce) **0** až **7**
 - **šestnáctkové (hexadecimální)**: symboly (číslíce) **0** až **9** a **A** až **F**
- jednoduchý převod mezi soustavami:

Převod zápisu čísla v soustavě o základu B^k ($k \in \mathbb{N}$) na zápis v soustavě o základu B (a naopak):

každý symbol soustavy o základu B^k zapisující nějaké číslo nahradíme k -ticí symbolů soustavy o základu B zapisující stejné číslo (a naopak, k -tice symbolů v zápisu brány zprava, chybějící symboly nahrazeny 0)

Základní operace v počítači = logické operace

- formální základ = **výroková logika** – zkoumá pravdivostní hodnotu výroků (pravda/nepravda, spojky/operátory “neplatí, že” \rightarrow operace negace \neg , “a současně platí” \rightarrow konjunkce \wedge , “nebo platí” \rightarrow disjunkce \vee , “jestliže platí, pak platí” \rightarrow implikace \Rightarrow aj.)
- výroky = **logické výrazy** vyhodnocované na hodnoty pravda/nepravda, 1/0
- matematický aparát pro práci s log. výrazy: **Booleova algebra (binární, dvoustavová, logika)**
- fyzická realizace – **logické elektronické obvody** – základ digitálních zařízení
- binární logika: univerzální, teoreticky zvládnutá, efektivně realizovatelná logickými el. obvody

Logická proměnná x

- veličina nabývající dvou možných diskretních logických hodnot: 0 (nepravda) a 1 (pravda)
- definice: $x = 1$ jestliže $x \neq 0$ a $x = 0$ jestliže $x \neq 1$

Logická funkce $f(x_1, \dots, x_n)$

- funkce n logických proměnných x_1, \dots, x_n nabývající dvou možných diskretních hodnot 0 (nepravda) a 1 (pravda)
- logická proměnná = logická funkce identity proměnné, skládání funkcí
- základní = **logické operace**

Booleova algebra (binární logika)

- algebra logických proměnných a logických funkcí
- dvouhodnotová algebra, algebra dvou stavů
- relace rovnosti: $f = g$, právě když $(f = 1 \wedge g = 1) \vee (f = 0 \wedge g = 0)$

3 základní:

Negace (inverze)

- pravdivá, když operand nepravdivý, jinak nepravdivá

x	\bar{x}
0	1
1	0

- operátory: \bar{x} , NOT x , $\neg x$ (výrokově negace, algebraicky negace), \bar{X} (množinově doplněk)

Logický součin (konjunkce)

- pravdivá, když oba operandy pravdivé, jinak nepravdivá

x	y	$x \cdot y$
0	0	0
0	1	0
1	0	0
1	1	1

- operátory: $x \cdot y / xy$ (prázdný), x AND y , $x \wedge y$ (výrokově konjunkce, algebraicky průsek), $X \cap Y$ (množinově průnik)

Logický součet (disjunkce)

- nepravdivá, když oba operandy nepravdivé, jinak pravdivá

x	y	$x + y$
0	0	0
0	1	1
1	0	1
1	1	1

- operátory: $x + y$, $x \text{ OR } y$, $x \vee y$ (výrokově disjunkce, algebraicky spojení), $X \cup Y$ (množinově sjednocení)

Logický výraz

- = korektně vytvořená posloupnost (symbolů) logických proměnných a funkcí (operátorů) spolu se závorkami
 - priority sestupně: negace, log. součin, log. součet
 - např. $x \cdot \bar{y} + f(x, z) = (x \cdot \bar{y}) + f(x, z)$
- = zápis logické funkce

Logické rovnice

- ekvivalentní úpravy: negace obou stran, logický součin/součet obou stran se stejným výrazem, . . . , log. funkce obou stran se stejnými ostatními operandy funkce
- NEkvivalentní úpravy: “krácení” obou stran o stejný (pod)výraz, např. $x + y = x + z$ není ekvivalentní s $y = z$

Axiomy (Booleovy algebry)

■ komutativita:

$$x \cdot y = y \cdot x \qquad x + y = y + x$$

■ distributivita:

$$x \cdot (y + z) = x \cdot y + x \cdot z \qquad x + y \cdot z = (x + y) \cdot (x + z)$$

■ identita (existence neutrální hodnoty):

$$\mathbf{1} \cdot x = x \qquad \mathbf{0} + x = x$$

■ komplementárnost:

$$x \cdot \overline{x} = \mathbf{0} \qquad x + \overline{x} = \mathbf{1}$$

Vlastnosti základních logických operací

■ nula a jednička:

$$0 \cdot x = 0 \qquad \mathbf{I} + x = \mathbf{I}$$

■ idempotence:

$$x \cdot x = x \qquad x + x = x$$

■ asociativita:

$$x \cdot (y \cdot z) = (x \cdot y) \cdot z \qquad x + (y + z) = (x + y) + z$$

■ involuce (dvojitá negace):

$$\overline{\overline{x}} = x$$

■ De Morganovy zákony:

$$\overline{x \cdot y} = \overline{x} + \overline{y} \qquad \overline{x + y} = \overline{x} \cdot \overline{y}$$

■ absorpce:

$$x \cdot (x + y) = x \qquad x + x \cdot y = x$$

■ a další

Vlastnosti základních logických operací – použití

- důkazy: s využitím axiomů a již dokázaných vlastností, rozbořem případů (dosazením všech možných kombinací hodnot **0** a **1** za proměnné)
- ekvivalentní úpravy (pro zjednodušování) logických výrazů
- ...

Další operace

Implikace

- nepravdivá, když první operand pravdivý a druhý nepravdivý, jinak pravdivá

x	y	$x \rightarrow y$
0	0	1
0	1	1
1	0	0
1	1	1

- operátory: $x \rightarrow y$, $x \rightarrow y$ (výrokově i algebraicky implikace), $X \subseteq Y$ (množinově podmnožina)

Ekvivalence

- pravdivá, když operandy mají stejnou hodnotu, jinak nepravdivá

x	y	$x \equiv y$
0	0	I
0	I	0
I	0	0
I	I	I

- operátory: $x \equiv y$, $x \text{ XNOR } y$, $x \equiv y$ (výrokově i algebraicky ekvivalence), $X \equiv Y$ (množinově ekvivalence nebo rovnost)

Nonekvivalence (negace ekvivalence, aritmetický součet modulo 2)

- pravdivá, když operandy mají různou hodnotu, jinak nepravdivá

x	y	$x \oplus y$
0	0	0
0	1	1
1	0	1
1	1	0

- operátory: $x \oplus y$, $x \text{ XOR } y$, $x \not\equiv y$ (výrokově i algebraicky negace ekvivalence), $X \not\equiv Y$ (množinově negace ekvivalence)

Shefferova funkce (negace logického součinu)

- nepravdivá, když oba operandy pravdivé, jinak pravdivá

x	y	$x \uparrow y$
0	0	1
0	1	1
1	0	1
1	1	0

- operátory: $x \uparrow y$, $x \text{ NAND } y$

Piercova funkce (negace logického součtu)

- pravdivá, když oba operandy nepravdivé, jinak nepravdivá

x	y	$x \downarrow y$
0	0	1
0	1	0
1	0	0
1	1	0

- operátory: $x \downarrow y$, $x \text{ NOR } y$

■ zadání **pravdivostní tabulkou**:

- úplně – funkční hodnota $f(x_i)$ definována pro všech 2^n možných přiřazení hodnot proměnným $x_i, 0 \leq i < n$
- neúplně – funkční hodnota pro některá přiřazení není definována (např. log. obvod realizující funkci ji neimplementuje)

■ **základní tvary** (výrazu):

- **součinový (úplná konjunktivní normální forma, ÚKNF)** – log. součin log. součtů všech proměnných nebo jejich negací (úplných elementárních disjunkcí, ÚED)

$$(X_0 + \dots + X_{n-1}) \cdot \dots \cdot (X_0 + \dots + X_{n-1}) \quad X_i = x_i \text{ nebo } \overline{x_i}$$

- **součtový (úplná disjunktivní normální forma, ÚDNF)** – log. součet log. součinů všech proměnných nebo jejich negací (úplných elementárních konjunkcí, ÚEK)

$$(X_0 \cdot \dots \cdot X_{n-1}) + \dots + (X_0 \cdot \dots \cdot X_{n-1}) \quad X_i = x_i \text{ nebo } \overline{x_i}$$

Převod log. funkce $f(x_i)$ na základní tvar (normální formu)

- ekvivalentními úpravami a doplněním chybějících proměnných nebo jejich negací
- **tabulkovou metodou:**
 - 1 pro řádky s $f(x_i) = 0(I)$ sestroj log. součet (součin) všech x_i pro $x_i = 0(I)$ nebo $\overline{x_i}$ pro $x_i = I(0)$
 - 2 výsledná ÚKNF (ÚDNF) je log. součinem (součtem) těchto log. součtů (součinů)

x	y	z	$f(x, y, z)$	ÚED	ÚEK
0	0	0	0	$x + y + z$	
0	0	1	0	$x + y + \overline{z}$	
0	1	0	0	$x + \overline{y} + z$	
0	1	1	1		$\overline{x} \cdot y \cdot z$
1	0	0	0	$\overline{x} + y + z$	
1	0	1	1		$x \cdot \overline{y} \cdot z$
1	1	0	1		$x \cdot y \cdot \overline{z}$
1	1	1	1		$x \cdot y \cdot z$

$$\text{ÚKNF}(f(x, y, z)): (x + y + z) \cdot (x + y + \overline{z}) \cdot (x + \overline{y} + z) \cdot (\overline{x} + y + z)$$

$$\text{ÚDNF}(f(x, y, z)): \overline{x} \cdot y \cdot z + x \cdot \overline{y} \cdot z + x \cdot y \cdot \overline{z} + x \cdot y \cdot z$$

Převeďte několik log. funkcí se třemi a více proměnnými do ÚKNF a ÚDNF.

Věta (O počtu log. funkcí)

Existuje právě $2^{(2^n)}$ logických funkcí s n proměnnými.

Funkce f^1 jedné proměnné

x	f_0	f_1	f_2	f_3
	0	x	\bar{x}	I
0	0	0	I	I
I	0	I	0	I

Funkce f^2 dvou proměnných

x	y	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}
		0	\cdot		x		y	\oplus	$+$	\downarrow	\equiv	\bar{y}		\bar{x}	\rightarrow	\uparrow	I
0	0	0	0	0	0	0	0	0	0	I	I	I	I	I	I	I	I
0	I	0	0	0	0	I	I	I	I	0	0	0	0	I	I	I	I
I	0	0	0	I	I	0	0	I	I	0	0	I	I	0	0	I	I
I	I	0	I	0	I	0	I	0	I	0	I	0	I	0	I	0	I

Funkce více než dvou proměnných

pro $n = 3$:

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

a podobně pro $n > 3$

Věta (O reprezentaci log. funkcí)

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

Funkce více než dvou proměnných

pro $n = 3$:

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

a podobně pro $n > 3$

Věta (O reprezentaci log. funkcí)

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

Zjednodušení výrazu logické funkce

- = optimalizace za účelem dosažení co nejmenšího počtu operátorů (v kompromisu s min. počtem typů operátorů)
 - důvod: méně (typů) log. obvodů realizujících funkci (menší, levnější, nižší spotřeba, ...)

Algebraická minimalizace

$$f = \bar{x} \cdot y \cdot z + x \cdot \bar{y} \cdot z + x \cdot y \cdot \bar{z} + x \cdot y \cdot z$$

// dvakrát přičteme $x \cdot y \cdot z$ (idempotence)

$$f = (\bar{x} \cdot y \cdot z + x \cdot y \cdot z) + (x \cdot \bar{y} \cdot z + x \cdot y \cdot z) + (x \cdot y \cdot \bar{z} + x \cdot y \cdot z)$$

// distributivita

$$f = y \cdot z \cdot (\bar{x} + x) + x \cdot z \cdot (\bar{y} + y) + x \cdot y \cdot (\bar{z} + z) \quad // \text{komplementárnost}$$

$$f = x \cdot y + y \cdot z + x \cdot z$$

- pro složitější výrazy náročná

Zjednodušení výrazu logické funkce

Karnaughova metoda (Veitch diagram)

- nahrazení algebraických ekvivalentních úprav geometrickými postupy
 - nalezení minimálního výrazu
- 1 k výrazu v základním součtovém tvaru se sestaví tzv. **Karnaughova mapa** = tabulka vyplněná **I** v buňkách reprezentující log. součiny, součiny reprezentované sousedními buňkami se liší v 1 proměnné
 - 2 hledání smyček (minterm) v mapě splňujících jisté podmínky (min. počet, max. obdélníková oblast vyplněná **I**, počet políček mocnina 2, mohou se překrývat, pokrytí všech **I**)
 - 3 smyčky po vyloučení komplementárních proměnných a jejich negací reprezentují log. součiny výsledného součtového tvaru

Zjednodušení výrazu logické funkce

Karnaughova metoda (Veitch diagram)

$$f = \bar{x} \cdot y \cdot z + x \cdot \bar{y} \cdot z + x \cdot y \cdot \bar{z} + x \cdot y \cdot z$$

	$\bar{x} \cdot \bar{y}$	$\bar{x} \cdot y$	$x \cdot y$	$x \cdot \bar{y}$
\bar{z}			1	
z		1	1	1

Obrázek: Karnaughova mapa

$$f = x \cdot y + y \cdot z + x \cdot z$$

- výpočetně náročná (hledání smyček)

Další algoritmické metody: tabulační (Quine-McCluskey), branch-and-bound (Petrick), Espresso logic minimizer aj.

Pokuste se minimalizovat log. funkce z přechozího úkolu.

Úplný systém logických funkcí

- = množina log. funkcí, pomocí kterých je možné vyjádřit jakoukoliv log. funkci (libovolného počtu proměnných)
- množina log. funkcí dvou proměnných (Věta o reprezentaci log. funkcí)
 - (1) negace \bar{x} , log. součin $x \cdot y$ a log. součet $x + y$
 - (2) negace \bar{x} a implikace $x \rightarrow y$
 - a další

Minimální úplný systém logických funkcí

- = úplný systém, ze kterého nelze žádnou funkci vyjmout tak, aby zůstal úplný
 - (1) NENÍ: $x \cdot y = \overline{\bar{x} + \bar{y}}$, $x + y = \overline{\bar{x} \cdot \bar{y}}$ (De Morganovy zákony)
 - (2) je
 - (3) negace \bar{x} a log. součin $x \cdot y$
 - (4) negace \bar{x} a log. součet $x + y$
 - a další

Úplný systém logických funkcí

- = množina log. funkcí, pomocí kterých je možné vyjádřit jakoukoliv log. funkci (libovolného počtu proměnných)
- množina log. funkcí dvou proměnných (Věta o reprezentaci log. funkcí)
 - (1) negace \bar{x} , log. součin $x \cdot y$ a log. součet $x + y$
 - (2) negace \bar{x} a implikace $x \rightarrow y$
 - a další

Minimální úplný systém logických funkcí

- = úplný systém, ze kterého nelze žádnou funkci vyjmout tak, aby zůstal úplný
 - (1) NENÍ: $x \cdot y = \overline{\bar{x} + \bar{y}}$, $x + y = \overline{\bar{x} \cdot \bar{y}}$ (De Morganovy zákony)
 - (2) je
 - (3) negace \bar{x} a log. součin $x \cdot y$
 - (4) negace \bar{x} a log. součet $x + y$
 - a další

Minimální úplný systém logických funkcí

Jediná funkce:

- **Shefferova** \uparrow (negace log. součinu)
- **Piercova** \downarrow (negace log. součtu)
- důkaz: vyjádření např. negace a log. součinu (součtu)

Vyjádření logické funkce pomocí Shefferovy nebo Piercovy funkce

- 1 vyjádření funkce v základním součtovém tvaru
- 2 zjednodušení výrazu funkce, např. pomocí Karnaughovy metody
- 3 aplikace De Morganových zákonů pro převedení výrazu do tvaru, který obsahuje pouze Shefferovy nebo pouze Piercovy funkce

Vyjádření logické funkce pomocí Shefferovy nebo Piercovy funkce

$$f = \bar{x} \cdot y \cdot z + x \cdot \bar{y} \cdot z + x \cdot y \cdot \bar{z} + x \cdot y \cdot z$$

$$f = x \cdot y + y \cdot z + x \cdot z$$

$$f = \overline{\bar{x} \cdot \bar{y} \cdot \bar{y} \cdot \bar{z}} + x \cdot z$$

$$f = \overline{\overline{\overline{\overline{x \cdot y \cdot y \cdot z}} \cdot \overline{x \cdot z}}}$$

$$f = \overline{\overline{\overline{\overline{\overline{x \cdot y \cdot y \cdot z} \cdot \overline{x \cdot y \cdot y \cdot z}} \cdot \overline{x \cdot z}}}}$$

$$f = (\bar{x} + y + z) \cdot (x + \bar{y} + z) \cdot (x + y + \bar{z}) \cdot (x + y + z)$$

$$f = (x + y) \cdot (y + z) \cdot (x + z)$$

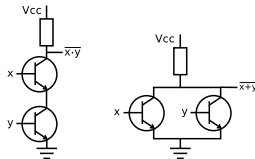
$$f = \overline{\overline{x + y + y + z}} \cdot (x + z)$$

$$f = \overline{\overline{\overline{\overline{x + y + y + z} + \overline{x + z}}}}$$

$$f = \overline{\overline{\overline{\overline{\overline{x + y + y + z} + \overline{x + y + y + z}} + \overline{x + z}}}}$$

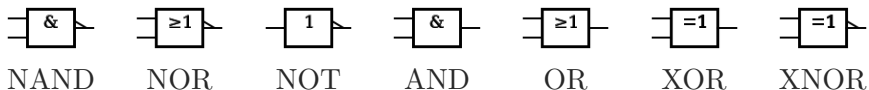
Vyjádřete log. operace negace, log. součin, log. součet, implikace, ekvivalence a nonekvivalence pomocí (1) Shefferovy funkce a (2) Piercovy funkce.

- dříve pomocí **spínacích relé** a **elektronek**
- dnes pomocí **tranzistorů v integrovaných obvodech**

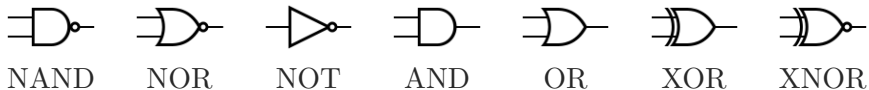


Obrázek: Realizace log. operací NAND a NOR

- realizace log. operací pomocí integrovaných obvodů – **logických členů, hradel**
 - vstupy = reprezentované log. proměnné
 - výstup = výsledek realizované log. operace
 - stavy (signály) na vstupech/výstupu = log. (binární) hodnoty **0/I** = míra informace s jednotkou **1 bit**
- symbolické značky log. členů ve schématech zapojení **logických obvodů** realizujících lib. log. funkci

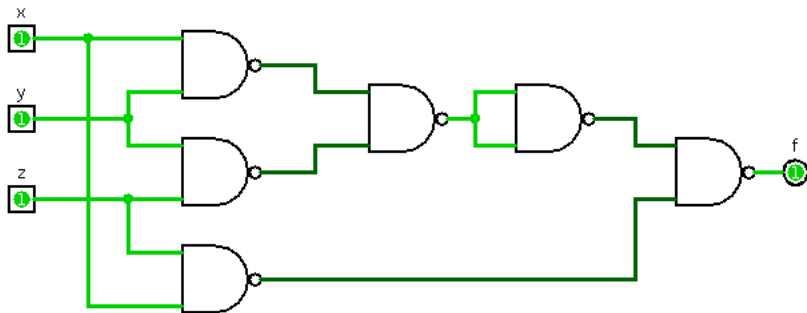


Obrázek: Symbolické značky logických členů (podle normy IEC)



Obrázek: Symbolické značky logických členů (tradiční, ANSI)

$$f = \overline{\overline{\overline{x \cdot y \cdot y \cdot z} \cdot \overline{\overline{x \cdot y \cdot y \cdot z}} \cdot \overline{x \cdot z}}}$$



Obrázek: Schéma zapojení log. obvodu realizujícího log. funkci f pomocí log. členů realizujících log. operaci NAND

Nakreslete schéma zapojení log. obvodu realizujícího log. operace NOT, AND, OR, implikace, ekvivalence a XOR pomocí log. členů realizujících operaci (1) NAND a (2) NOR.

- jeden výstup: realizace jedné log. funkce
- více výstupů: realizace více log. funkcí zároveň \rightarrow realizace **vícebitové log. funkce** ${}^n f$
- n -tice vstupů: reprezentace **vícebitových (n -bitových) log. proměnných** ${}^n x =$
vícebitový log. obvod
- **kombinační**: stavy na výstupech obvodu (tj. funkční hodnota) závisí pouze na okamžitých stavech na vstupech (tj. hodnotách proměnných)
- **sekvenční**: stavy na výstupech obvodu (tj. funkční hodnota) závisí nejen na okamžitých stavech na vstupech (tj. hodnotách proměnných), ale také na přechozích stavech na vstupech

- stavy na výstupech obvodu (tj. funkční hodnota) závisí pouze na okamžitých stavech na vstupech (tj. hodnotách proměnných)
- jedné kombinaci stavů na vstupech odpovídá jediná kombinace stavů na výstupech

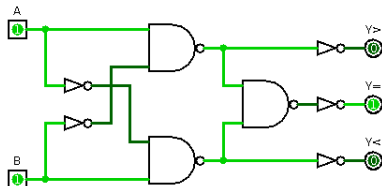
Komparátor

- provádí srovnání hodnot dvou log. proměnných A a B na vstupu
- tři výstupy udávající pravdivost vztahů: $A < B$, $A > B$ a $A = B$, realizace tříbitové log. funkce $Y_{<} = Y(A < B)$, $Y_{>} = Y(A > B)$, $Y_{=} = Y(A = B)$
- jednobitový:

$$\begin{aligned} Y_{<} &= \overline{A} \cdot B & Y_{>} &= A \cdot \overline{B} & Y_{=} &= A \cdot B + \overline{A} \cdot \overline{B} \\ Y_{<} &= \overline{\overline{\overline{A} \cdot B}} & Y_{>} &= \overline{\overline{\overline{A \cdot \overline{B}}}} & Y_{=} &= \overline{\overline{A \cdot B}} \cdot \overline{\overline{A \cdot \overline{B}}} \end{aligned}$$

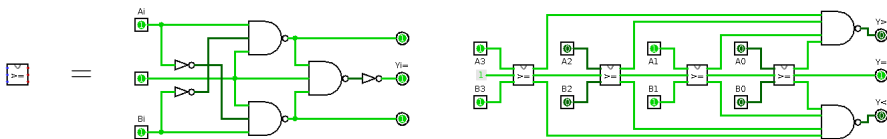
Komparátor

A	B	$Y_{<}$	$Y_{>}$	$Y_{=}$
0	0	0	0	1
0	1	1	0	0
1	0	0	1	0
1	1	0	0	1



Obrázek: Pravdivostní tabulka a schéma zapojení jednobitového komparátoru

- vícebitový: zřetěžené zapojení jednobitových pro každý řád vícebitových proměnných od nejvýznamějšího po nejméně významný



Obrázek: Schéma zapojení čtyřbitového komparátoru

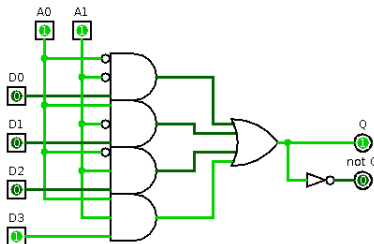
Multiplexor

- přepíná na výstup Q log. hodnotu na jednom z 2^n datových vstupů D_i vybraném na základě n -bitové hodnoty na adresním vstupu A
- kromě výstupu Q navíc ještě negovaný (invertovaný) výstup \overline{Q}
- např. čtyřvstupý (4 datové vstupy, dvoubitový adresní vstup) realizuje log. funkci

$$Q = \overline{A_0} \cdot \overline{A_1} \cdot D_0 + A_0 \cdot \overline{A_1} \cdot D_1 + \overline{A_0} \cdot A_1 \cdot D_2 + A_0 \cdot A_1 \cdot D_3$$

Multiplexor

A_0	A_1	Q
0	0	D_0
1	0	D_1
0	1	D_2
1	1	D_3



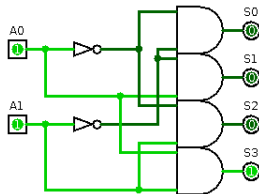
Obrázek: Pravdivostní tabulka a schéma zapojení čtyřvstupého multiplexoru

- použití: multiplexování datových vstupů na základě adresy

Binární dekodér

- nastaví (na **I**) jeden z 2^n výstupů S_i odpovídající n -bitové hodnotě na adresním vstupu A

A_0	A_1	S_0	S_1	S_2	S_3
0	0	I	0	0	0
I	0	0	I	0	0
0	I	0	0	I	0
I	I	0	0	0	I



Obrázek: Pravdivostní tabulka a schéma zapojení bin. dekodéru se čtyřmi výstupy

- použití: dekodér adresy pro výběr místa v paměti

Binární sčítačka

- čísla ve dvojkové soustavě = binárně reprezentovaná
- platí stejná pravidla aritmetiky jako v desítkové soustavě, např. (+ je zde aritmetické sčítání!):

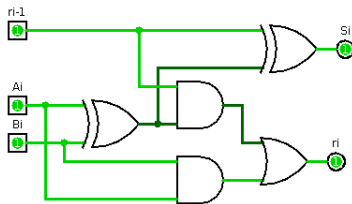
$$0 + 0 = 0 \quad 0 + 1 = 1 \quad 1 + 1 = 10$$

- sčítačka sečte binární hodnoty v každém řádu dvou n -bitových proměnných A a B podle pravidel aritmetiky pro sčítání, tj. s přenosem hodnoty do vyššího řádu
- realizuje log. funkce součtu S_i v řádu $0 \leq i < n$ a přenosu r_i z řádu i do vyššího řádu:

$$S_i = A_i \oplus B_i \oplus r_{i-1} \quad r_i = A_i \cdot B_i + (A_i \oplus B_i) \cdot r_{i-1}, \quad r_{-1} = 0$$

Binární sčítačka

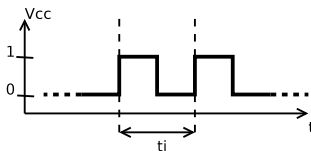
A_i	B_i	r_{i-1}	S_i	r_i
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1



Obrázek: Pravdivostní tabulka a schéma zapojení jednobitové sčítačky (pro řád i)

- vícebitová: zřetěžené zapojení jednobitových pro každý řád vícebitových proměnných od nejméně významného po nejvýznamější s přenosem do vyššího řádu
- použití: (aritmetické) sčítání binárně reprezentovaných 8-, 16-, 32-, atd. bitových čísel

- stavy na výstupech obvodu (tj. funkční hodnota) závisí nejen na okamžitých stavech na vstupech (tj. hodnotách proměnných), ale také na přechozích stavech na vstupech
- předchozí stavy na vstupech zachyceny **vnitřním stavem obvodu**
- nutné identifikovat a synchronizovat stavy obvodu v čase
- čas: periodický impulsní signál = “hodiny” (clock), diskrétně určující okamžiky synchronizace obvodu, generovaný krystalem o dané frekvenci

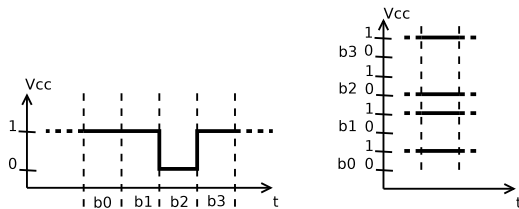


Obrázek: Časový signál “hodin” (clock)

- zpětné vazby z (některých) výstupů na (některé) vstupy

Přenos dat (hodnot vícebitových log. proměnných):

- **sériový**: bity (hodnoty 0/I) přenášeny postupně v čase za sebou po jednom datovém vodiči
- **paralelní**: bity přenášeny zároveň v čase po více datových vodičích
- úlohy transformace mezi sériovým a paralelním přenosem



Obrázek: Sériový a paralelní přenos dat

Klopné obvody

- nejjednodušší sekvenční obvody

druhy:

- **astabilní**: nemají žádný stabilní stav, periodicky (např. podle hodinových impulsů) překlápí výstupy z jednoho stavu do druhého; použití jako generátory impulsů
- **monostabilní**: jeden stabilní stav na výstupech, po vhodném řídicím signálu je po definované dobu ve stabilním stavu; použití k vytváření impulsů dané délky
- **bistabilní**: oba stavy na výstupech stabilní, zůstává v jednom stabilním stavu dokud není vhodným řídicím signálem překlopen do druhého; použití pro realizaci **paměti**

Řízení:

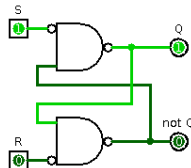
- **asynchronně** signály (**0** nebo **1**) na datových vstupech
- **synchronně** hodinovým signálem
- **hladinou** signálu: horní (**1**) nebo dolní (**0**)
- **hranami** signálu: nástupní (**0** → **1** u horní hladiny) nebo sestupní (**0** → **1** u dolní hladiny)

Klopný obvod (typu) RS

- nejjednodušší bistabilní, základ ostatních
- **jednobitový paměťový člen**
- asynchronní vstupy R (Reset) pro nulování log. hodnoty na výstupu Q (v čase i) a S (Set) pro nastavení hodnoty
- kromě výstupu Q navíc ještě negovaný (invertovaný) výstup \overline{Q}

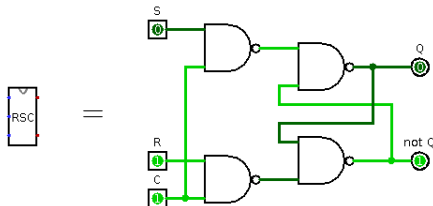
Klopný obvod (typu) RS

R	S	Q_i	$\overline{Q_i}$
0	0	Q_{i-1}	$\overline{Q_{i-1}}$
0	1	1	0
1	0	0	1
1	1	N/A	N/A

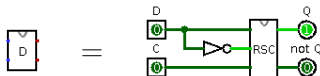


Obrázek: Pravdivostní tabulka a schéma zapojení klopného obvodu RS

- varianta se synchronizačním vstupem C s hodinových signálem



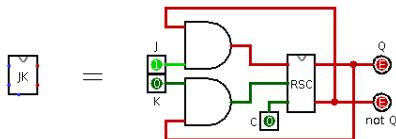
Klopný obvod (typu) D



Obrázek: Schéma zapojení klopného obvodu D

- realizace pomocí klopného obvodu RS, navíc mohou být vstupy R a S
- typ Latch: asynchronní řízení stavu vstupu D hladinou signálu na vstupu C
- typ D: synchronní (flip-flop) řízení stavu vstupu D nástupní hranou hodinového signálu na vstupu C
- typ Master-Slave: dvoufázový (master, slave), synchronní řízení stavu vstupu D nástupní i sestupní hranou hodinového signálu na vstupu C , rozšíření = **klopný obvod (typu) JK**

Klopný obvod (typu) D

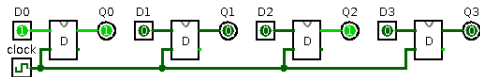


Obrázek: Schéma zapojení klopného obvodu JK

- implementace ve formě integrovaných obvodů, např. MH 7472, MH 7474, MH 7475

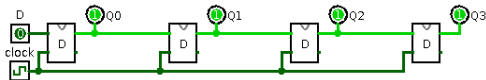
Obvody v počítačích:

- **paralelní registr (střádač):** vícebitová paměť pro hodnotu dodanou paralelně na více vstupů, paralelní zapojení klopných obvodů D



Obrázek: Schéma zapojení čtyřbitového paralelního registru

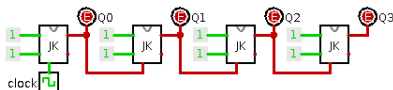
- **sériový (posuvný) registr:** vícebitová paměť pro hodnotu dodanou sériově na vstupu, sériové zapojení klopných obvodů D, použití pro transformaci sériových dat na paralelní



Obrázek: Schéma zapojení čtyřbitového sériového registru

Obvody v počítačích:

- **čítač**: paměť počtu impulsů na hodinovém vstupu, binárně reprezentovaný počet na vícebitovém výstupu, zřetěžené zapojení klopných obvodů JK



Obrázek: Schéma zapojení čtyřbitového čítače

- **sériová sčítačka**: (aritmetické) sčítání log. hodnot dodávaných na vstupy v sériovém tvaru po jednotlivých řádech