Počítač

Jiří Zacpal

DEPARTMENT OF COMPUTER SCIENCE
PALACKÝ UNIVERSITY, OLOMOUC

KMI/YUDIT Úvod do informačních technologií

Osnova

- Předmět KMI/YUDIT
- Historie počítačů
- Architektura osobního počítače
- Číselné soustavy
- Logické funkce a obvody
- Reprezentace čísel a znaků

Úvod do informačních technologií

- Obsah přednášek:
  1. Co je to počítač, historie vývoje počítačů. Architektura počítače (von Neumannova) a princip jeho činnosti. Číselné soustavy, logické funkce a obvody. Reprezentace čísel a znaků.
  2. Základní deska počítače a interní sběrnice počítače (PCI). Princip činnosti mikroprocesoru (CPU) a vnitřních pamětí (RAM, cache). Interní součásti počítače, přídavné karty (grafická, zvuková, síťová, multimediální), vnější paměti (pevné disky a disková pole). Periferie a externí sběrnice počítače (USB). Monitor (CRT, LCD),
     polohovací zařízení (klávesnice, myš aj.), datové mechaniky a média (floppy, CD, DVD), tiskárna a skener, modem.
  3. Operační systém a jeho funkce při ovládání počítače, z uživatelského i administrátorského pohledu. Struktura a funkce operačního systému (správa procesů, paměti a disku)
  4. Počítačové sítě, technologie a principy fungování. Celosvětová síť Internet a její služby.
  5. Základy databázových systémů a zpracování dat.

Způsob výuky

- **Prezenční výuka.** Studenti se dostaví pětkrát za semestr k organizované výuce. Ta probíhá formou přednášek, seminářů a cvičení.
- **Konzultace**
  - **Prezenční konzultace (úterý, pátek 10.00-11.00, pracovna 5.044)**
  - **Emailové konzultace (jiri.zacpal@upol.cz)**
  - **Konzultace telefonem (**585 63**4 706)**
  - Na plánovanou konzultaci a její obsah je vhodné tutora předem upozornit emailem. Po dohodě lze sjednat konzultaci individuálně i mimo stanovené konzultační hodiny.
- **Samostudium a samostatná práce**
  - Většina předmětů je zabezpečena studijními texty v elektronické podobě (http://phoenix.inf.upol.cz/esf/materialy.htm):
    - J. Hronek: Struktura počítačů
    - P. Příhoda: Počítačové sítě
    - J. Hronek: Databázové systémy
    - Vyuka/KMI_YUDIT, složka vyukovy_text_uvt
  - Studenti mají k dispozici studijní literaturu v angličtině i v češtině k zapůjčení v Knihovně Přírodovědecké fakulty

Historie počítačů

Počítač

- je **elektronické** zařízení, které zpracovává **data** pomocí předem vytvořeného **programu**
- skládá se z:
  - **hardware**, které představuje fyzické části počítače
  - **software** (operační systém a programy)
- je zpravidla ovládán **uživatelem**, který poskytuje počítači data ke zpracování prostřednictvím jeho **vstupních zařízení** a počítač výsledky prezentuje pomocí **výstupních zařízení**
- v současnosti jsou počítače využívány téměř ve všech oborech lidské činnosti

Historie počítačů – Nultá generace

- **mechanické části**, **relé**, (desítky operací/s)
- 1936 - Turingův stroj (teoretický model), Alan Turing
- 1937 - dvojková, digitální elektronika, Claude Shannon
- 1937 - Atanasoff–Berry Computer, dvojkový, neprogramovatelný (soustavy lineárních rovnic), ne turingovsky úplný
- 1938 - reléový počítací automat Z-1, Konrád Zuse, pomalý, nespolehlivý,
- 1941 - Z-3 programovatelný,
- 1943 - Colossus, kryptoanalýza Enigmy (Bletchley Park)

Harvard Mark I

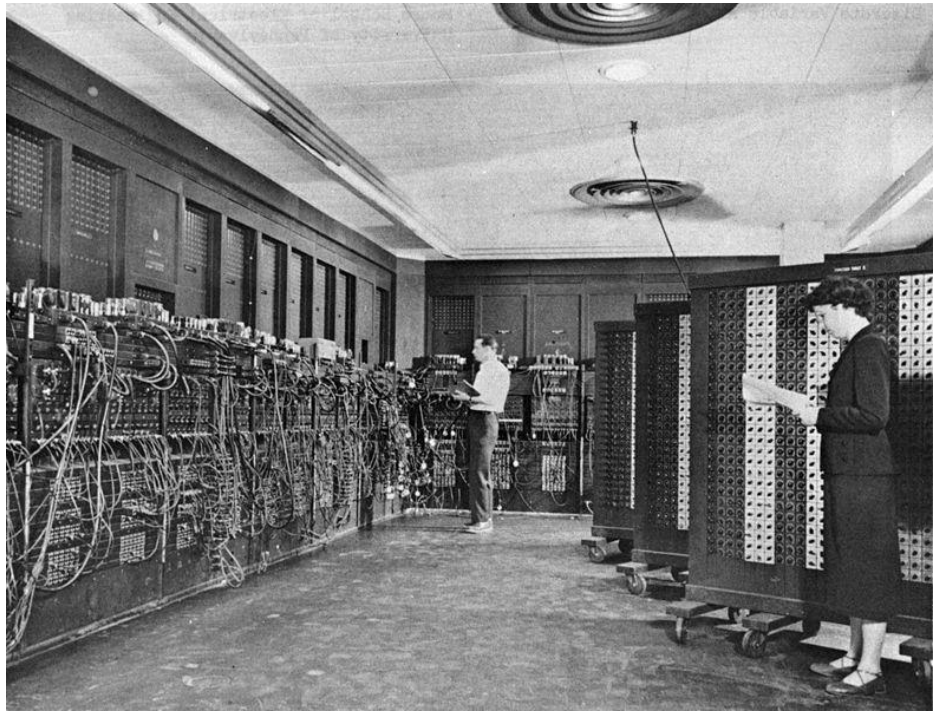
- počítač nulté generace - mechanický stroj, obsahoval však některé elektromagnetické součástky (relé)
- sestrojen během 2. světové války firmou IBM
- později převezen na Harvardskou univerzitu
- 765 000 komponent
- instrukce načítal z děrného štítku
- 0,3 s součet, 6 s násobení, 1 min výpočet jedné periody funkce sinus
- americké námořnictvo ho využívalo k výpočtu balistických tabulek

Historie počítačů – První generace

- **elektronky**, (stovky až tisíce operací/s)
- 1945 - idea řízení počítače programem uloženým v paměti, John von Neumann
- 1946 - ENIAC (Electronic Numerical Integrator and Computer),
- 1951 - EDVAC, Bellovy laboratoře, dvojkový, IAS (1952, John von Neumann), lépe navržený a univerzálnější než ENIAC - program v paměti spolu s daty, dále UNIVAC, MANIAC, JOHNNIAC, IBM 650, Strela (1953)
- paměti: magnetické bubny, děrné štítky a pásky

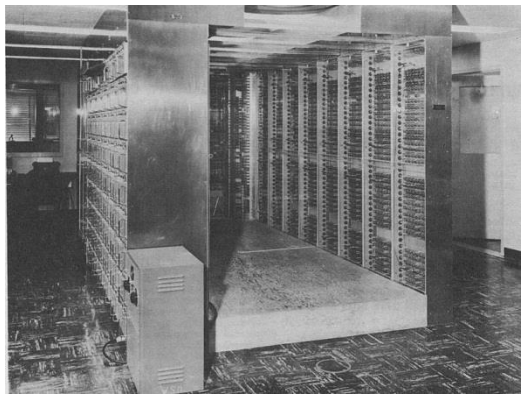
ENIAC- Electronic Numerical Integrator And Computer

- počítač první generace, kompletně elektronický
- vývoj financovaný Armádou Spojených Států Amerických během 2. světové války (University of Pennsylvania)
- přibližně 1000 krát rychlejší než Harvard Mark I
- 30 tun,15m^2 (bývalá univerzitní tělocvična),
- 17460 elektronek, 1500 relé, 174 kW (chlazení vzduchem od vrtulí dvou leteckých motorů),
- násobení v řádu ms, dekadický, programovatelný pomocí přepínačů a kabelů, výpočet konfigurace vodíkové bomby,
- 1955 - rozebrán



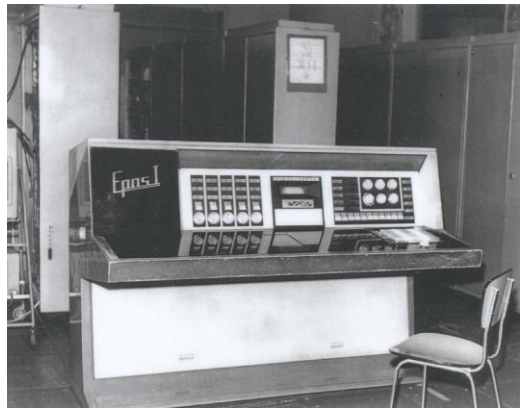
MANIAC – Matemathical Analyser Numerical Integrator And Computer

- počítač první generace
- inspirovaný ENIACem
- uveden do provozu John von Neumannem
- v projektu Manhattan byl použit k vývoji první jaderné bomby



EPOS 1, EPOS 2

- EPOS 1 byl zkonstruován pod vedením prof. Antonína Svobody
- roku 1960 ve Výzkumném ústavu matematických strojů
- počítač první generace, plně elektronkový
- následník EPOS 2 již osazený tranzistory (počítač druhé generace)



Historie počítačů – Druhá generace

- **tranzistory**, desítky až stovky tisíc operací/s
- 1947 - polovodičový **tranzistor**, Bellovy laboratoře, Bardeen-Brattain-Shockley
- 1956 - **TX** (“tixo”, MIT, 18-bitová slova), další Univac, IBM 7XXX
- 1963 - PDP-6 (DEC, jen 23 kusů), time sharing, 36-bitová slova
- paměti: feritové, magnetické disky a pásky
- různý nekompatibilní hardware
- (nižší) programovací jazyky: strojový kód, “assemblery”, Fortran, Algol, COBOL

Historie počítačů (1) – Třetí generace

- **integrované obvody**, miliony operací/s
- 1959 - integrovaný obvod (s více tranzistory na křemíkovém čipu)
- míra integrace v počtu tranzistorů na čipu: SSI (desítky), MSI (stovky, konec 60. let)
- 1964 - IBM System/360, počátek rodiny mainframů, 32-bitová slova, 8 bitů = byte, adresace bytů
- 1968 - PDP-10 na univerzitách (MIT, Stanford, Carnegie Mellon), „hackerský“
- 1970 - mikroprocesor, Intel 4004 (1971, 4-bit), 8008 (1972, 8-bit), 8080 (1974), {8086} (1978, 16-bit), Motorola 6800 (1974, 8-bit),{68000} (1979, 16/32-bit)

Historie počítačů (2) – Třetí generace

- 1975 - mikropočítače ALTAIR 8800 a IMSAI 8080, další Apple I (1976)
- 80. léta - Sinclair ZX 80, Commodore C64, IBM PC (1981), **ZX Spectrum**, Apple Lisa (1983, GUI), IBM PC/XT (1983), Apple Macintosh (1984), IBM PC/AT (1984), Atari ST (1985), Commodore Amiga (1985), IBM PS/2 (1987)
- paměti: magnetické disky a pásky, elektronické kompatibilní hardware, modulární architektury
- (vyšší) programovací jazyky: Lisp, BASIC, Pascal, C, Smalltalk, ...
- terminální sítě a počítačové sítě

Historie počítačů - Současnost

- **integrované obvody**, miliardy operací/s
- míra integrace: LSI (desetitisíce, 70. léta),
- VLSI (stovky tisíc až miliardy, od 80. let)
- paměti: magnetické a optické disky, elektronické (FLASH)
- (víceúčelové) programovací jazyky: Python, Visual Basic, Java, C#
- počítačové clustery

Architektura osobního počítače

Kategorie počítačů z hlediska hardware (1)

- **mikropočítač (osobní počítač)**
  - mikroprocesor
  - na 1 čipu
  - typy: workstation, desktop, server, laptop, notebook, palmtop, PDA, embedded, 1 uživatel, všeobecné použití
- **minipočítač (midrange)**
  - terminálové serverové počítače, větší diskový prostor,
  - více periferií, hotswap hardware, spolehlivé, více uživatelů (I/O zařízení),
  - použití v obchodní systémech, průmyslu, např. DEC PDP, VAX, IBM
  - HP 3000, Sun SPARC Enterprise,
  - v pol. 80 let (nahrazeny sítěmi) serverů a pracovních stanic

Kategorie počítačů z hlediska hardware (2)

- **mainframe (sálový počítač)**
  - velký diskový prostor, mnoho periferií,
  - paralelní architektury, vysoký výkon, použití pro výpočty (průmysl),
  - zpracování hromadných dat (statistiky, banky), např. IBM System/360,
  - System z10
- **superpočítač**
  - paralelní a distribuovaná architektury, velmi vysoký
  - výkon, náročné
  - výpočty nad rozsáhlými daty, použití pro výzkum, meteorologii, seismologii apod. - simulace, např. Cray, IBM Blue Gene, Roadrunner

Osobní počítač

- vychází z minipočítačů z konce 70. let
- příbuznost a (částečná nebo úplná) kompatibilita s počítači IBM PC a Apple Macintosh
- IBM PC (od roku 1981), procesor 8088
- IBM PC XT (8-bitový) -> IBM PC AT (16, 32, dnes 64-bitový)
- základní koncepce technického provedení počítače - „skládačka“:
  - základní deska s procesorem, pamětí a přídavnými kartami,
  - vstupní a výstupní zařízení

von Neumannova koncepce počítače (1)

- 1946 - Princeton Institute for Advanced Studies
- řízení počítače programem uloženým v paměti
- architektura:
  - procesor (CPU): řadič + aritmeticko-logická jednotka (ALU)
  - operační paměť: lineárně organizovaná, rozdělená na stejně velké buňky, přístup pomocí adres
  - vstupně/výstupní (I/O) zařízení

von Neumannova koncepce počítače (2)

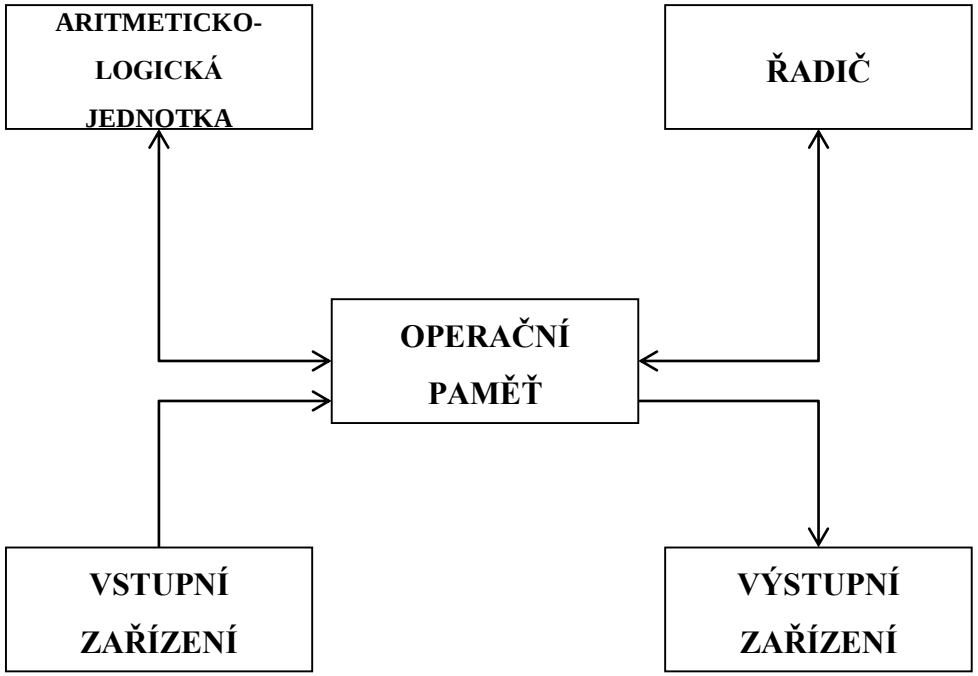


Schéma počítače dle von Neumanna

von Neumannova koncepce počítače (3)

- **program** = předpis pro řešení úlohy = posloupnost elementárních kroků, tzv. instrukcí
- **instrukce** = interpretovaná binární data se speciálním významem
- (proměnná) **data a program** načtené do jedné společné operační paměti
- činnost počítače řídí **řadič**: s využitím ALU zpracovává instrukce programu nad daty čtenými z paměti nebo vstupního zařízení, výsledná data se zapisují do paměti nebo výstupního zařízení

von Neumannova koncepce počítače (4)

- instrukce programu vykonávány sekvenčně, výjimku tvoří instrukce skoků
- ALU: základní početní operace (sčítání, násobení, logické, bitové posuvy)
- von Neumann bottleneck: rychlost zpracování instrukcí vs. rychlost komunikace s pamětí

von Neumannova koncepce počítače (5)

- koncepce, až na drobné odlišnosti, používaná dodnes:
  - rozšíření o koncepci přerušení od I/O a dalších zařízení
    - umožňuje efektivně zpracovávat více programů „zároveň” i na jednom CPU (multitasking)
  - více než jeden procesor (radič, ALU), zpracovávání více programů zároveň
  - postupné načítání programu do paměti podle potřeby

Harvardská koncepce počítače

- podle počítače MARK I (program na děrné pásce, data na elektromechanických deskách)
- Architektura podobná von Neumannově, až na:
  - dvě oddělené paměti pro program a pro data
  - paměť programu často jen pro čtení
  - paralelní přístup do pamětí
- modifikovanou ji interně používají moderní CPU (instrukční a datová cache)
- DSP procesory v audio/video technice, jednoúčelové (programovatelné) mikrokontroléry (Atmel AVR), kalkulátory

Informace

Informace

- Jednotkou informace je **bit** = 2 různé hodnoty (ano-ne, 0-1)
- Osminásobek jednoho bitu se nazývá **byte** (označení 1B) = 256 různých hodnot (0-255)

$$2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 = 2^8 = 256$$

Základní jednotky

Jednotka	Značka	B
Kilobyte	kB	1 000
Kibibyte	KiB	1 024
Megabyte	MB	1 000 000
Mebibyte	MiB	1 048 576
Gigabyte	GB	10^9
Gibibyte	GiB	$\sim 1{,}074 \cdot 10^9$
Terabyte	TB	10^{12}
Tebibyte	TiB	$\sim 1{,}1 \cdot 10^{12}$

Kódování dat

Kódování dat

- Jakákoliv informace uložená v počítači musí být nejprve převedena do tvaru, kterému počítač rozumí. Tomuto převodu říkáme **kódování**.
- Zpětnému převodu do podoby, která je čitelná pro člověka, naopak říkáme **dekódování**.

Kódování čísel

- Obecně se pro zápis čísel používají tzv. **číselné soustavy**.
  - číslo dané soustavy je posloupností symbolů, které se nazývají **číslice** (nebo cifry)
  - každá číselná soustava je určena **základem** *z*, což je nenulové přirozené číslo, které udává maximální počet použitelných číslic
  - skutečná hodnota každé číslice je pak dána pozicí ve zmíněné posloupnosti symbolů.

Kódování čísel

- číslo ***A*** v číselné soustavě o základu ***z*** můžeme napsat jako posloupnost

$$A = a_n a_{n-1} a_{n-2} \dots a_1 a_0,$$

  - kde $a_n a_{n-1} a_{n-2} \dots a_1 a_0$ jsou jednotlivé číslice čísla *A*, přičemž a_n je nejvýznamnější číslice a a_0 je nejméně významná číslice
- hodnota čísla *A* se pak určí jako součet mocnin základu, které jsou vynásobené jednotlivými číslicemi:

$$A = a_n \cdot z^n + a_{n-1} \cdot z^{n-1} + a_{n-2} \cdot z^{n-2} + \cdots + a_1 \cdot z^1 + a_0 \cdot z^0.$$

- takovému zápisu říkáme polynomiální

Kódování čísel

- Příklad:
  - číslo 1234 v desítkové soustavě je možné napsat jako
$$(1234)_{10} = 1 \cdot 10^3 + 2 \cdot 10^2 + 3 \cdot 10^1 + 4 \cdot 10^0.$$
  - číslo 110101 ve dvojkové soustavě zapíšeme takto:
$$(110101)_2 = 1 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0.$$

Kódování záporných čísel

- **Přímé kódování** - první bit je vyhrazen pro znaménko
  - číslo 00001001 ve dvojkové soustavě je 9 v desítkové, a proto 10001001 představuje číslo -9
- **Doplňkový kód** - záporné číslo je zaznamenáno jako binární negace (záměna všech 0 za 1) původního čísla zvětšená o 1.
  - pokud 00001101 je binární vyjádření čísla 13, pak -13 se vypočte jako 11110010 + 1 = 11110011

Kódování záporných čísel

- **Aditivní kód** – výsledná binární reprezentace představuje nezáporné číslo, které vznikne součtem kódovaného čísla a domluvené konstanty (většinou polovina maximálního kladného čísla).
  - číslo $(-10)_d$ reprezentujeme pomocí 1 B jako $118 = (-10)_d + (256)_d / 2 = -10 + 128$
- **Inverzní kód** - kladná čísla se vyjadřují normálním způsobem, záporná čísla se vyjadřují binární negací čísla
  - například -3 vyjádříme kódem 11111100

Kódování čísel s fixní řádkovou čárkou

Příklad: kódování čísla $(0,625)_{10}$

$0,625 \cdot 2 = 1,250$	celočíselná část = 1	b_{-1}
$0,250 \cdot 2 = 0,500$	celočíselná část = 0	b_{-2}
$0,500 \cdot 2 = 1,000$	celočíselná část = 1	b_{-3}

Odtud je číslo $(0,625)_{10} = (0,101)_2$

$$\underline{\quad\quad\quad}\,,\,\underline{\quad\quad}$$

$$101{,}01$$

$$(101{,}01)_2 = 1 \,.\, 4 + 1 \,.\, 1 + 1 \,.\, 0{,}25 = (5{,}25)_{10}$$

Kódování čísel s pohyblivou řádkovou čárkou

- definované normou IEEE 754
- formáty
  - jednoduchá přesnost (single) – 32 bitů
  - dvojnásobná přesnost (double) – 64 bitů

$$X=(-1)^s \times 2^{exp-bias} \times m$$

  - 2 je báze, někdy také nazývaná radix. U IEEE 754 je to vždy dvojka, protože výpočty s bází dvě jsou pro číslicové obvody nejjednodušší. V minulosti se používaly i jiné báze, například 8, 16 nebo i 10.
  - exp je vždy kladná hodnota exponentu posunutého o hodnotu bias
  - bias je hodnota, díky které je uložený exponent vždy kladný. Tato hodnota se většinou volí dle vztahu: $bias=2^{eb-1}-1$, kde eb je počet bitů vyhrazených pro exponent.
  - m je mantisa, která je u formátů IEEE 754 vždy kladná
  - s je znaménkový bit nabývající hodnoty 0 nebo 1. Pokud je tento bit nulový, je reprezentovaná hodnota kladná, v opačném případě se jedná o zápornou hodnotu. Vzhledem k tomu, že je jeden bit vyhrazen na uložení znaménka, je možné rozlišit kladnou a zápornou nulu.

Jednoduchá přesnost

$$X = (-1)^s \times 2^{E-127} \times (1 + Q)$$

$$Q = m_1 \times 2^{-1} + m_2 \times 2^{-2} + \ldots + m_{22} \times 2^{-22} + m_{23} \times 2^{-2}$$

- 127 = $2^{eb-1}-1 = 2^{8-1}-1 = 2^7-1 = 128-1$
- exponent
  - od –127 do 128 (od -126 do 127)
    - -127 (00000000) a 128 (11111111) jsou použity pro speciální účely
- mantisa – ukládají do ní normalizovaná čísla v intervalu <1;2>
  - vzhledem k tomu, že první bit umístěný před binární tečkou vždy 1, není ho zapotřebí ukládat, což znamená, že ušetříme jeden bit z třicetidvoubitového slova

bit	31	30 29 … 24 23	22 21 … 3 2 1 0
význam	s	exponent (8 bitů)	mantisa (23 bitů)

Jednoduchá přesnost

- mezní hodnoty exponentu

podmínka	hodnota	poznámka
E = 1 až 254	$X = (-1)^s \times 2^{E-127} \times (1 + Q)$	základní formát
E = 0, Q ≠ 0	$X = (-1)^s \times 2^{-126} \times Q$	denormalizovaná čísla
E = 0, Q = 0, s = 0	X = 0	kladná nula
E = 0, Q = 0, s = 1	X = 0	záporná nula
E = 255, Q = 0, s = 0	X = +∞	kladné nekonečno (výsledek byl příliš vysoký)
E = 255, Q = 0, s = 1	X = -∞	záporné nekonečno (výsledek byl příliš nízký)
E = 255, Q > 0	X = NaN	není číslo

Jednoduchá přesnost

- příklad: 123,456
  1. $123{,}456 = 1{,}929 \times 2^6$
  2. s=0
  3. $E - 127 = 6 \rightarrow E = (133)_{10} = (10000101)_2$
  4. mantisa $(0{,}929)_{10} = (11101101110100101111001)_2$

	s	exponent								mantisa																					
bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
hodnota	0	1	0	0	0	0	1	0	1	1	1	1	0	1	1	0	1	1	1	0	1	0	0	1	0	1	1	1	1	0	0	1

Kódování textu

- Text je počítačem zpracováván jako posloupnost znaků. Problém kódování textu se tedy redukuje na problém kódování jednotlivých znaků dané abecedy
- **ASCII** (American Standard Code for Information Interchange) určený pro reprezentaci písmen anglické abecedy. Kromě písmen (velkých i malých) ASCII kóduje i číslice, speciální symboly (např. !,@,\#,\$) a několik tzv. netisknutelných znaků (mezera, tabulátor, posun na nový řádek apod.)
  - používá 7 bitů = 128 znaků

Kódování české abecedy

- Prvním řešením, které se nabízelo, bylo využít osmý bit ASCII kódování. Tím byl získán prostor pro dalších 128 znaků.
- Brzy se však ukázalo, že i celkový počet 256 znaků je žalostně málo. Pro každou abecedu tak postupně vznikalo mnoho různých vzájemně nekompatibilních kódování.
- Jen pro českou abecedu bylo vytvořeno alespoň šest kódování. Nejpoužívanějšími byla jednobytová kódování:
  - Windows-1250 v operačních systémech Windows,
  - ISO 8859-2 v operačních systémech Unix,
  - Kamenických v operačním systému DOS.

Unicode

- Každý znak v Unicode má jednoznačný **číselný kód** a svůj **název**.
- Tabulka Unicode poskytuje prostor pro 1 114 112 znaků.
  - Tento prostor se dělí na 17 částí, každý o velikosti 216.
  - První část se nazývá Basic Multilingual Plane (BMP) a obsahuje znaky běžně používaných abeced.
  - Původní 16bitový návrh Unicode počítal jen s BMP, následně se ale ukázalo, že pro pokrytí všech používaných abeced to nestačí.
  - Prvních 128 znaků (tj. sedmibitové kódy) obsahuje znakovou sadu ASCII.

UTF

- Existuje několik různých způsobů, jak znaky Unicode kódovat.
- UTF-8 (UCS Transformation Format):
  - nejpoužívanější zobrazení Unicode znaků
  - pokud se znak v ASCII-7, zobrazí se beze změny v 1. bajtu
  - pokud není v ASCII, je zadán dvěma až šesti bajty:
    - 1. bajt: počet jedniček zleva vyjadřuje délku sekvence, nula je oddělovač,
    - další bajty: v nejvyšších dvou bitech vždy 10
  - pro českou abecedu (všechny znaky jsou v rozsahu U+0080 – U+07FF) stačí 2 B pro znaky s diakritikou, 1 B pro znaky bez diakritiky
  - Příklad: slovo „Příliš“

50	c5 99	c3 ad	6c	69	c5 a1
P	ř	í	l	i	š

ř: **110**0 0101 **10**01 1001 „ř“ v Unicode U+$(0159)_{16}$ =101011001_2

- UTF-16 a UTF-32
  - rozšíření základní šířky z 8 na 16 a 32 bitů
- UTF-7
  - pro sedmibitový přenos e-mailem

Binární logika

Binární logika (1)

- Základní operace v počítači = logické operace
- Formální základ = výroková logika (zkoumá pravdivostní hodnotu výroků -> pravda/nepravda ->1/0
- Matematický aparát pro práci s log. výrazy: Booleova algebra (binární, dvoustavová, logika)
- Fyzická realizace : **logické elektronické obvody** - základ digitálních zařízení

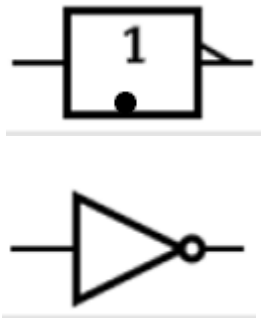
Binární logika (2)

- Logická proměnná x
  - veličina nabývající dvou možných diskrétních logických hodnot: 0 (nepravda) a 1 (pravda)
- Logická funkce $f(x_1,\dots , x_n)$
  - funkce n logických proměnných $x_1,\dots,x_n$ nabývající dvou možných diskrétních hodnot 0 (nepravda) a 1 (pravda)
- Booleova algebra (binární logika)
  - algebra logických proměnných a logických funkcí
  - dvouhodnotová algebra, algebra dvou stavů

Negace (inverze)

- pravdivá, když operand nepravdivý, jinak nepravdivá

x	$\bar{x}$
0	1
1	1



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

Logický součin (konjunkce)

- pravdivá, když oba operandy pravdivé, jinak nepravdivá

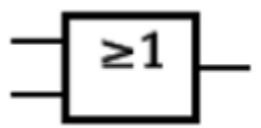
x	y	$x \cdot y$
0	0	0
1	0	0
0	1	0
1	1	1

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

Logický součet (disjunkce)

- nepravdivá, když oba operandy nepravdivé, jinak pravdivá

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



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

Implikace

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

x	y	$x \to y$
0	0	1
1	0	0
0	1	1
1	1	1

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

Ekvivalence

- pravdivá, když operandy mají stejnou hodnotu, jinak nepravdivá

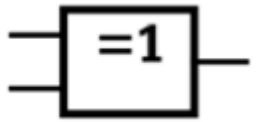
x	y	$x \equiv y$
0	0	1
1	0	0
0	1	0
1	1	1

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

Nonkvivalence

- pravdivá, když operandy mají různou hodnotu, jinak nepravdivá

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



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

Shefferova funkce (negace logického součinu)

- nepravdivá, když oba operandy pravdivé, jinak pravdivá

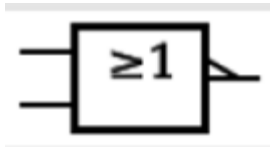
x	y	$x \uparrow y$
0	0	1
1	0	1
0	1	1
1	1	0

- operátory: $x \uparrow y$, x NAND y

Piercova funkce (negace logického součtu)

- pravdivá, když oba operandy nepravdivé, jinak nepravdivá

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



- operátory: $x \downarrow y$, x NOR y

Logické funkce (1)

- 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
  - NEekvivalentní 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):

$$1 \cdot x = x \quad 0 + x = x$$

- **komplementárnost**:

$$x \cdot \bar{x} = 0 \quad x + \bar{x} = 1$$

Vlastnosti základních logických operací

- nula a jednička:

$$0 \cdot x = 0 \quad 1 + x = 1$$

- 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 (dvojí negace):

$$\bar{\bar{x}} = x$$

- De Morganovy zákony:

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

- absorpce:

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

Logické funkce

- zadání pravdivostní tabulkou:
  - úplně - funkční hodnota f (x_i) definována pro všech 2^n možných přiřazení hodnot proměnným x_i ; 0 < 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)

$$(x_0 + \cdots + x_{n-1}) \cdot \cdots \cdot (x_0 + \cdots + x_{n-1}), X_i = x_i \; nebo \; \overline{x_i},$$

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

$$(x_0 \cdot \cdots \cdot x_{n-1}) + \cdots + (x_0 \cdot \cdots \cdot x_{n-1}), X_i = x_i \; nebo \; \overline{x_i}$$

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

- ekvivalentními úpravami a doplněním chybějících proměnných nebo jejich negací
- tabulkovou metodou:
  1. pro řádky s $f(x_i) = 0(1)$ sestroj log. součet (součin) všech x_i pro $x_i = 0(1)$ nebo $\overline{x_i}$ pro $x_i = 1(0)$ ·
  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	$x+y+\bar{z}$	
0	1	0	0	$x+\bar{y}+z$	
0	1	1	1		$\bar{x}\cdot y\cdot z$
1	0	0	0	$\bar{x}+y+z$	
1	0	1	1		$x\cdot\bar{y}\cdot z$
1	1	0	1		$x\cdot y\cdot\bar{z}$
1	1	1	1		$x\cdot y\cdot z$

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

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

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, . . .)
- metody:
  - algebraické úpravy
  - Karnaughova metoda (Karnaughova mapa)

Karnaughova metoda

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

Karnaughova metoda

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

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

- výpočetně náročné

Úplný systém logických funkcí

= množina log. funkcí, pomocí kterých je možné vyjádřit jakoukoliv log. funkci (libovolného počtu proměnných)

→množina log. funkcí dvou proměnných

- (1) negace $\bar{x}$, log. součin $x \cdot y$ a log. součet $x + y$
- (2) negace$\bar{x}$ a implikace $x \to y$
- 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) $\bar{x}$, log. součin $x \cdot y$
  - (4) $\bar{x}$, log. součet $x + y$
  - a další

Minimální úplný systém logických funkcí

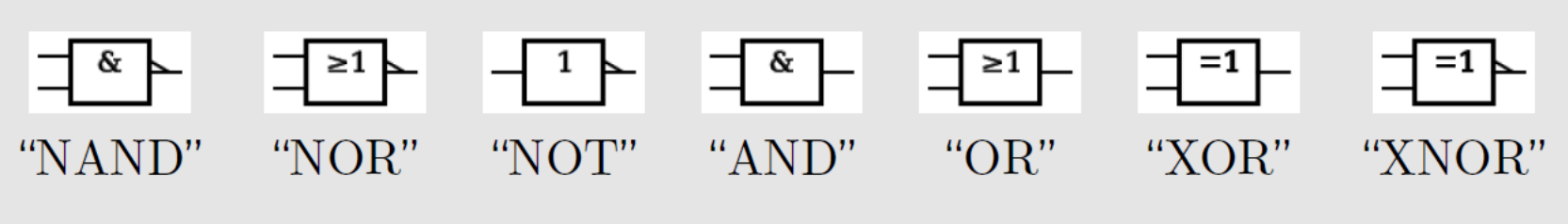
- Jediná funkce:
  - Sheerova $\uparrow$ (negace log. součinu)
  - Piercova $\downarrow$ (negace log. součtu)
  - důkaz: vyjádření negace a log. součinu (součtu)
  - Vyjádření logické funkce pomocí Sheerovy nebo Piercovy funkce
    1. vyjádření funkce v základním součtovém tvaru
    2. zjednodušení výrazu funkce, např. pomocí Karnaughovy metody
    3. aplikace De Morganových zákonů pro převedení výrazu do tvaru, který obsahuje pouze Sheerovy nebo pouze Piercovy funkce

Logické členy

Fyzická realizace logických funkcí

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

Značky logických členů - hradel

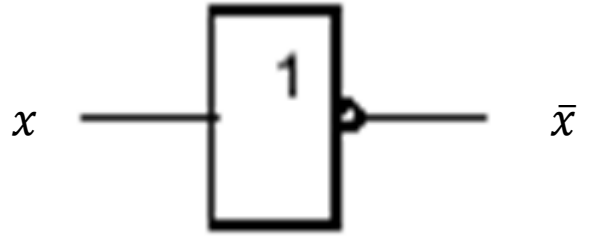


Obrázek: Symbolické značky logických členů (podle normy IEC)

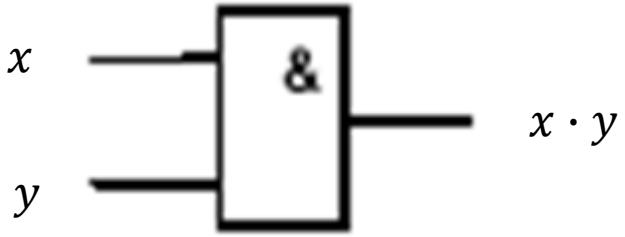
Obrázek: Symbolické značky logických členů (tradiční, ANSI)

Použité logické obvody - značky

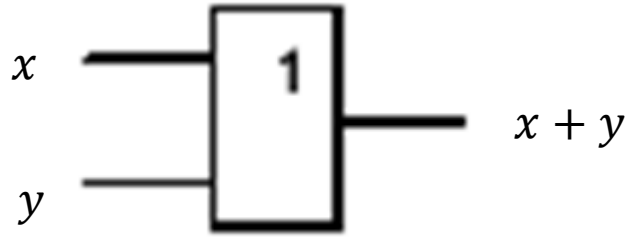
- negace



- konjunkce (logický součin)



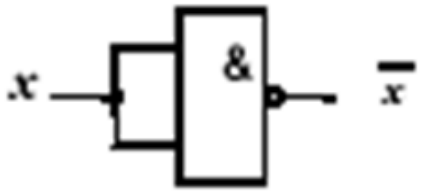
- disjunkce (logický součet)



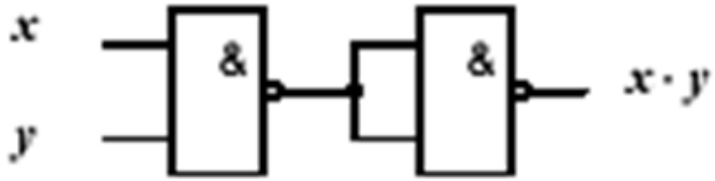
Minimální úplný systém log. funkcí - NAND

Realizace operátorů NOT, AND a OR pomocí NAND

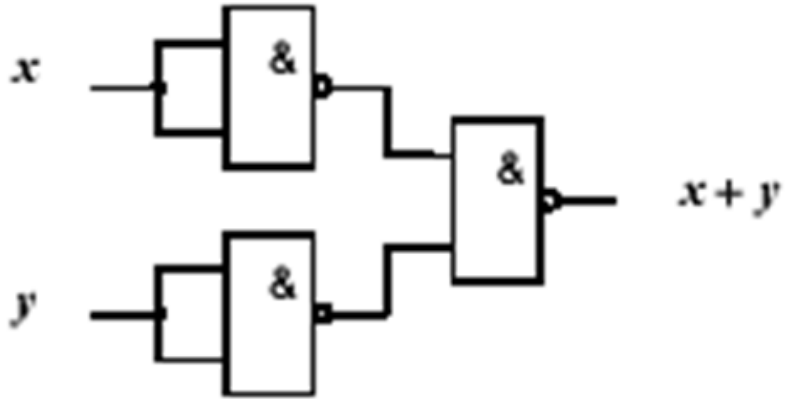
- NOT: $\bar{x} = \overline{x \cdot x}$



- AND: $x \cdot y = \overline{\overline{x \cdot y}} = \overline{\overline{x \cdot y} \cdot \overline{x \cdot y}}$



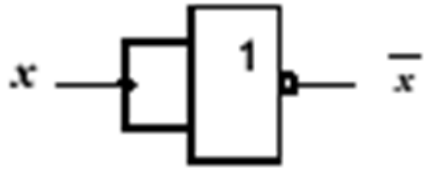
- OR: $x + y = \overline{\overline{x + y}} = \overline{\overline{x \cdot x} \cdot \overline{y \cdot y}}$



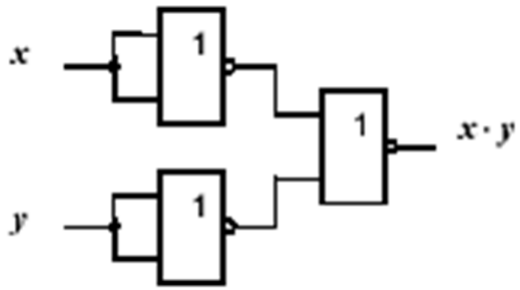
Minimální úplný systém log. funkcí - NOR

Realizace operátorů NOT, AND a OR pomocí NOR

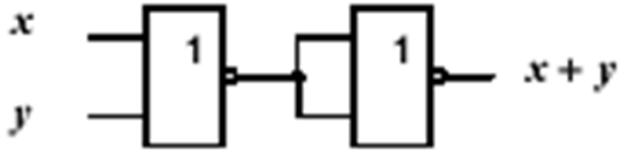
- NOT: $\bar{x} = \overline{x + x}$



- AND: $x \cdot y = \overline{\overline{x \cdot y}} = \overline{\bar{x} + \bar{y}} = \overline{(\overline{x + x}) + (\overline{y + y})}$



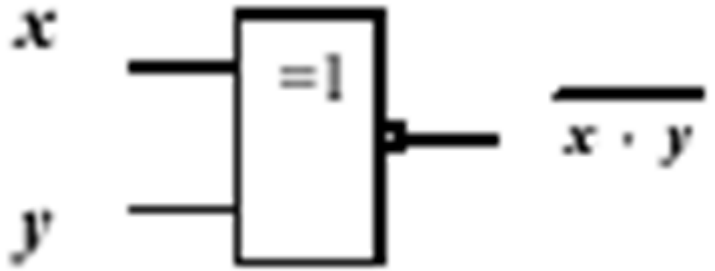
- OR: $x + y = \overline{\overline{x + y}} = \overline{\overline{(x + y)} + \overline{(x + y)}}$



Nonekvivalence – exclusive OR

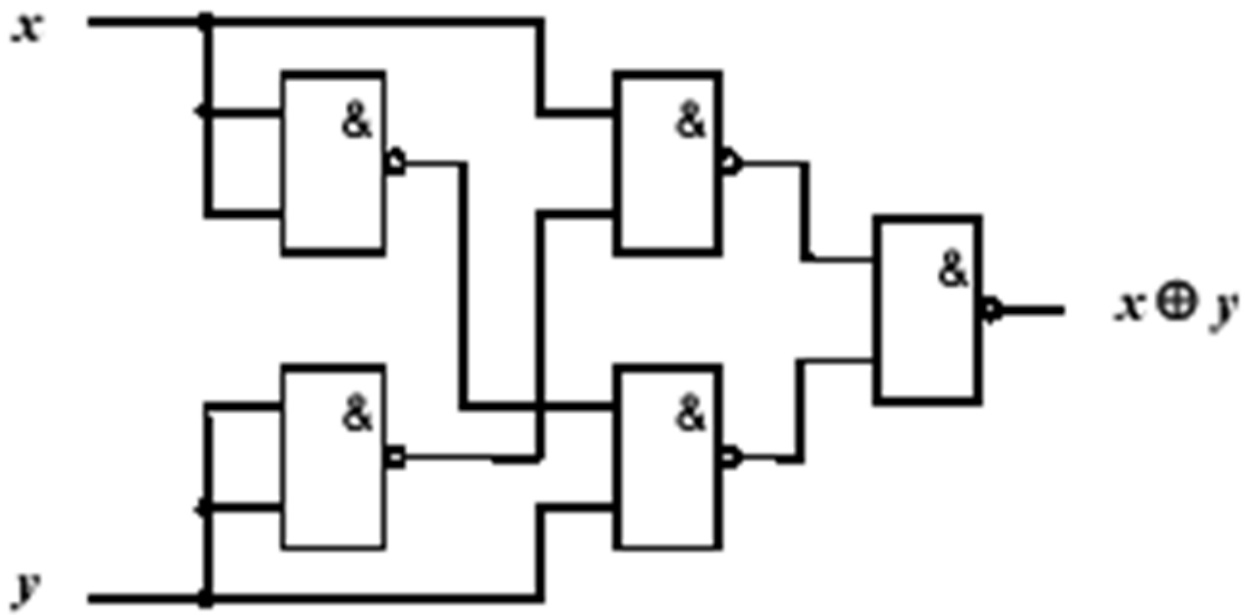
- pravdivá, když operandy mají různou hodnotu, jinak nepravdivá
- operátory: $x \oplus y, y$ algebra žinově negace ekvivalence)

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



Nonekvivalence pomocí NAND

$$x \oplus y = x \cdot \bar{y} + \bar{x} \cdot y = \overline{\overline{x \cdot \bar{y} + \bar{x} \cdot y}} = \overline{\overline{x \cdot \bar{y}} \cdot \overline{\bar{x} \cdot y}} = \overline{\overline{x \cdot \overline{y \cdot y}} \cdot \overline{\overline{x \cdot x} \cdot y}}$$



Logické obvody

Logické obvody

- jeden výstup: realizace jedné log. funkce
- více výstupů: realizace více log. funkcí zároveň realizace **vícebitové log. funkce**
- n-tice vstupů: reprezentace vícebitových (n-bitových) log. proměnných = **vícebitový log. obvod**
- druhy:
  - 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 předchozích stavech na vstupech

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:

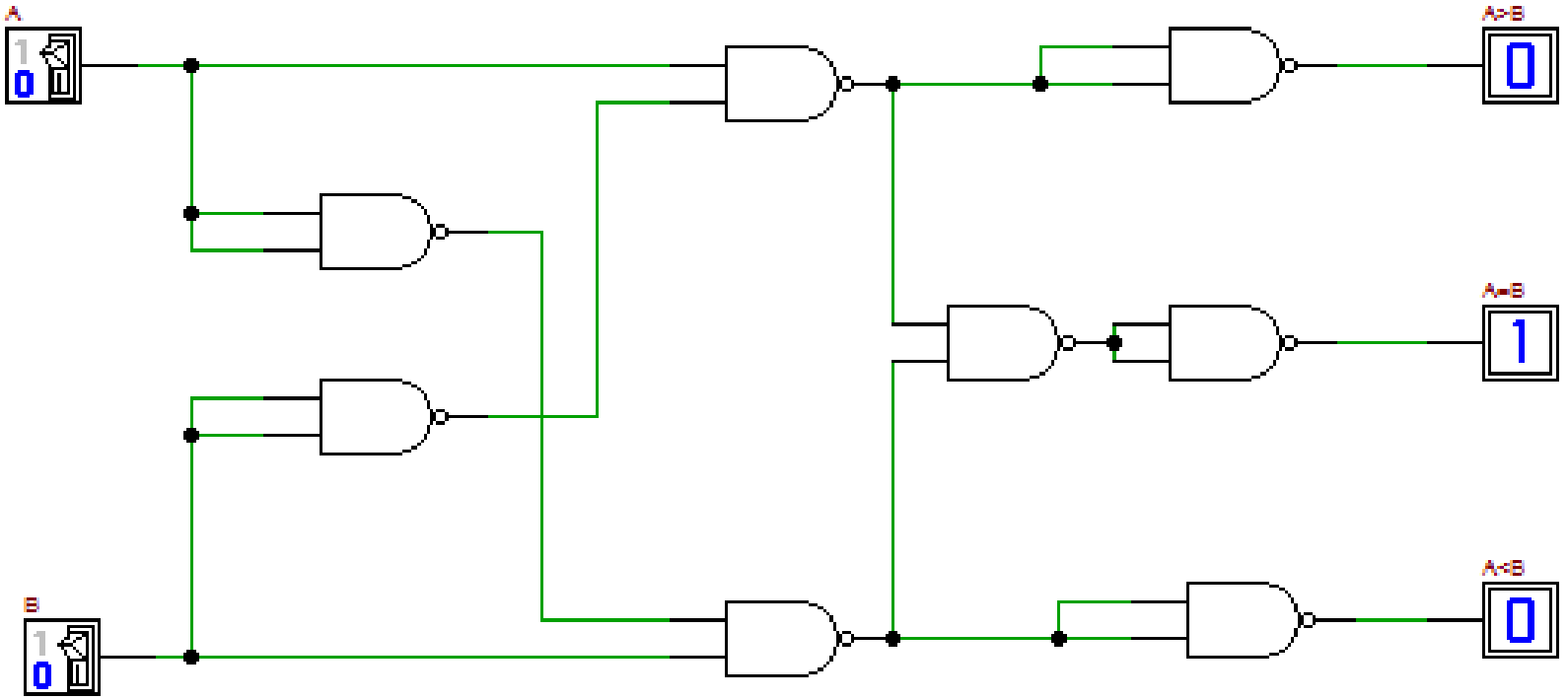
$$Y_< = Y\ (A < B);\ Y_> = Y\ (A > B);Y_= = Y\ (A = B)$$

- jednobitový:

$$Y_< = \bar{A} \cdot B \qquad Y_> = A \cdot \overline{B} \qquad Y_= = A \cdot B + \overline{A} \cdot \overline{B}$$

$$Y_< = \overline{\overline{\bar{A} \cdot B}} \qquad Y_> = \overline{\overline{A \cdot \overline{B}}} \qquad Y_= = \overline{\overline{\bar{A} \cdot B} \cdot \overline{A \cdot \overline{B}}}$$

Komparátor

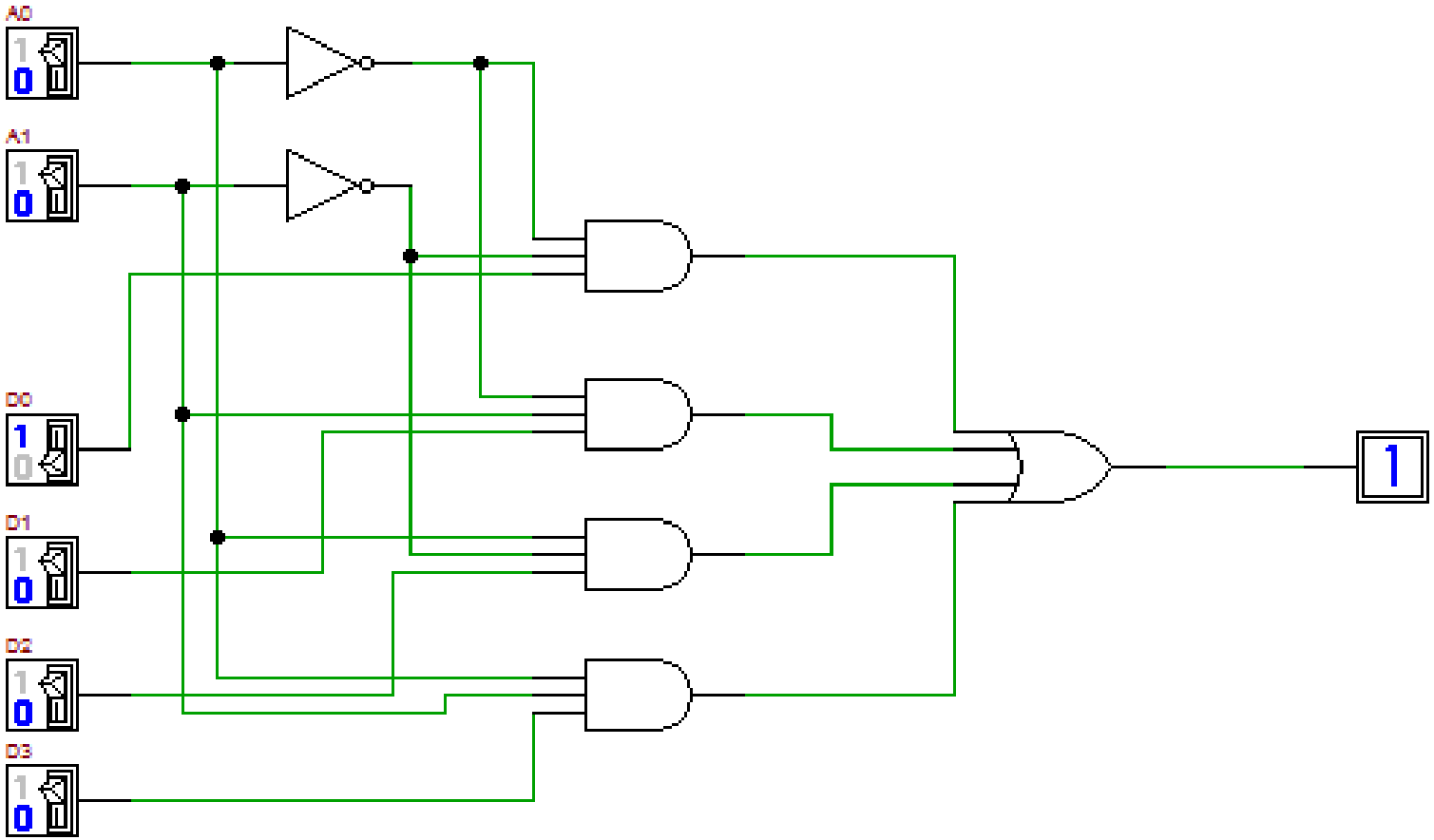


Multiplexor

- 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 Q
- např. čtyřvstupý (4 datové vstupy, dvoubitový adresní vstup) realizuje log. funkci

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

Multiplexor

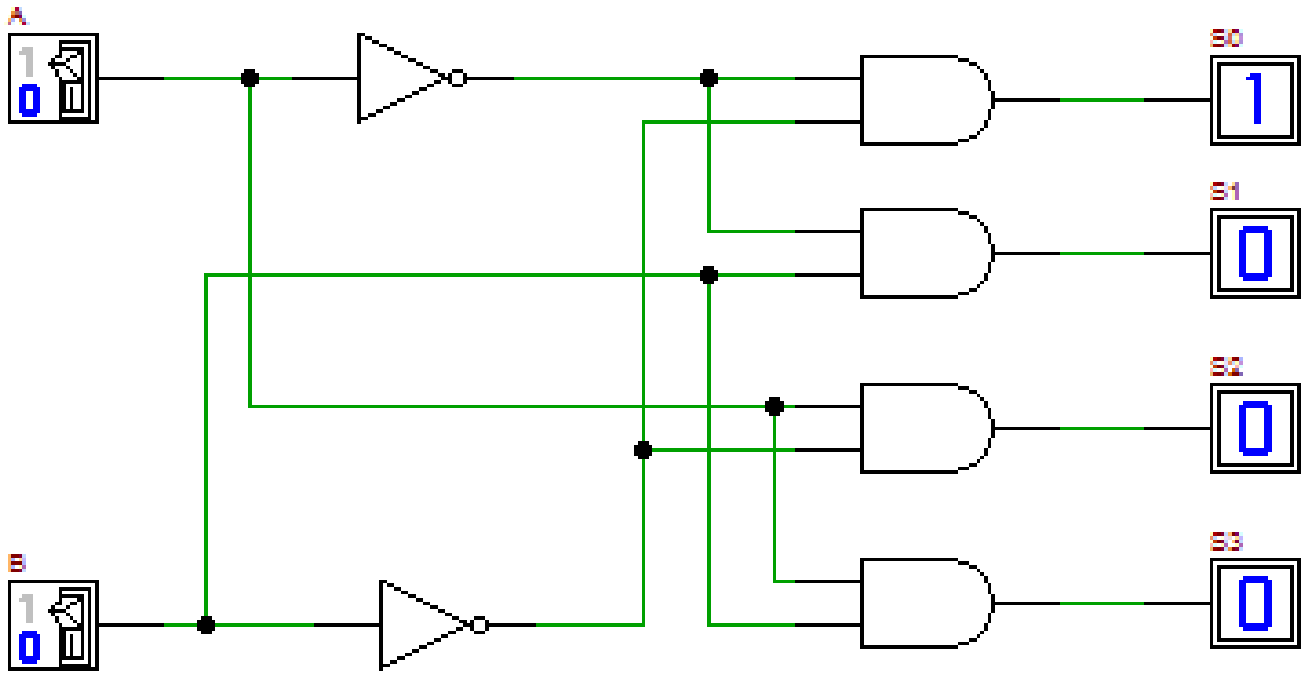


Binární dekodér

- nastaví (na I) jeden z 2^n výstupů S_i odpovídající n-bitové hodnotě na adresním vstupu A
- použití: dekodér adresy pro výběr místa v paměti

A_0	A_0	S_0	S_1	S_2	S_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

Binární dekodér

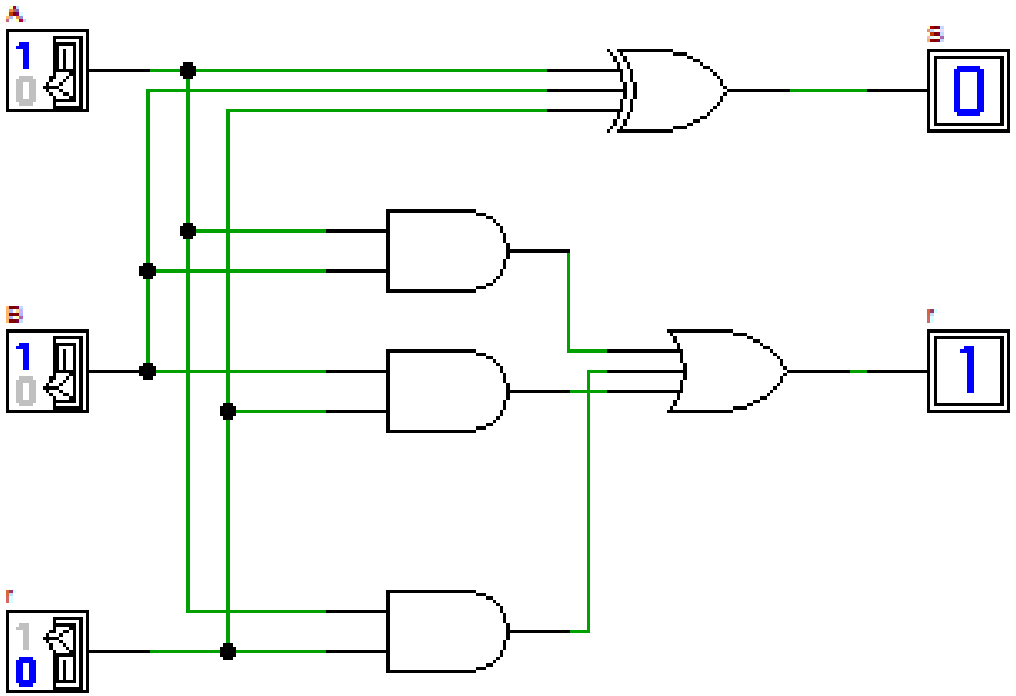


Binární sčítačka

- čísla ve dvojkové soustavě = binárně reprezentovaná
- 0 + 0 = 0 0 + I = I I + I = I0
- 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 a přenosu r_i z řádu i
- do vyššího řádu:

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

Binární sčítačka



Sekvenční logické obvody

- stavy na výstupech obvodu (tj. funkční hodnota) závisí nejen na okamžitých stavech na vstupech (tj. hodnotách proměnných), ale také na předchozích 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
- zpětné vazby z (některých) výstupů na (některé) vstupy

Přenos dat

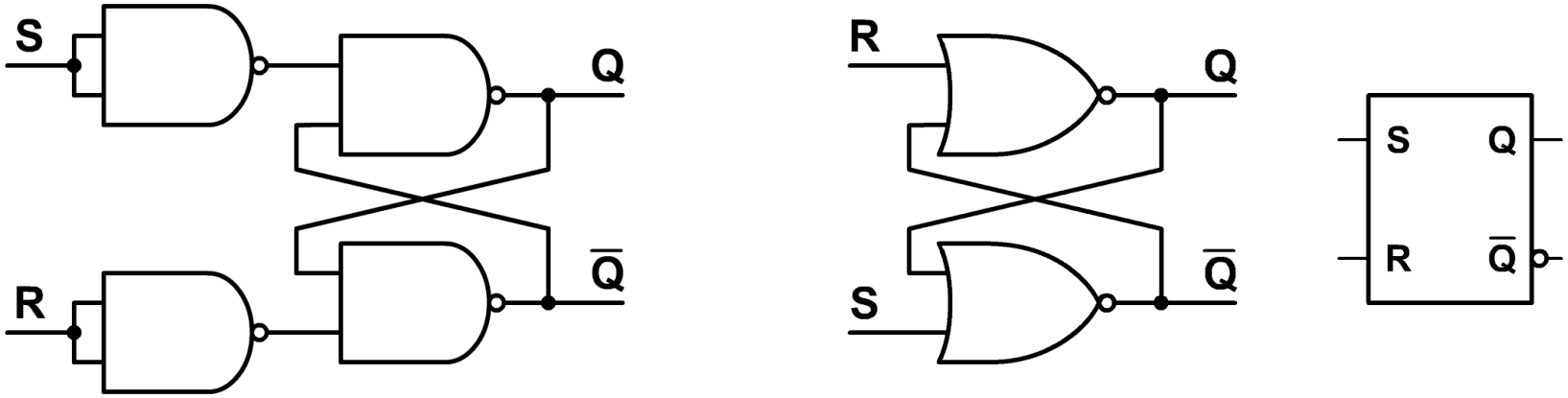
- přenos dat (hodnot vícebitových log. proměnných):
  - **sériový**: bity (hodnoty 0=I) přenášeny postupně v čase za sebou po jednom datovém vodiči
  - **paralelní**: bity přenášeny zároveň v čase po více datových vodičích
- úlohy transformace mezi sériovým a paralelním přenosem

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 řídícím signálu je po definovanou dobu ve stabilním stavu; použití k vytváření impulsů dané délky
  - **bistabilní**: oba stavy na výstupech stabilní, zůstává v jednom stabilním stavu dokud není vhodným řídícím signálem překlopen do druhého; použití pro realizaci pamětí
- řízení:
  - **asynchronně** signály (0 nebo I) na datových vstupech
  - **synchronně** hodinovým signálem
  - **hladinou signálu**: horní (I) nebo dolní (0)
  - **hranami signálu**: nástupní (0 -> I) nebo sestupní (I -> 0)

Klopné obvody (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 $\bar{Q}$

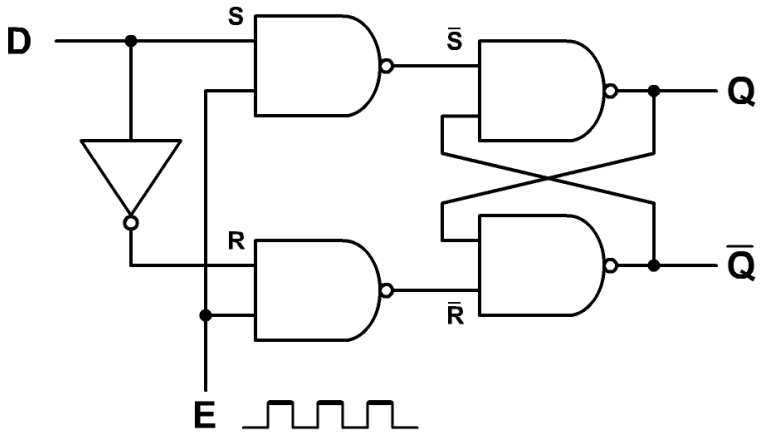


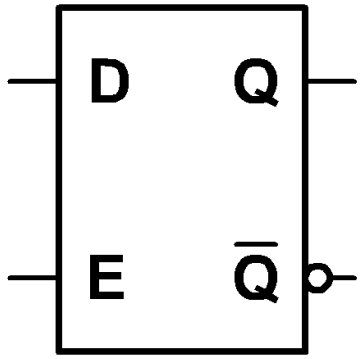
Funkce klopného obvodu RS

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

Klopné obvody (typu) D

- realizace pomocí klopného obvodu RS, navíc vstupy R a S
- signál D (data) nastavuje výstup
- signál E (*enable*, česky povolit) povoluje nebo blokuje nastavení výstupu



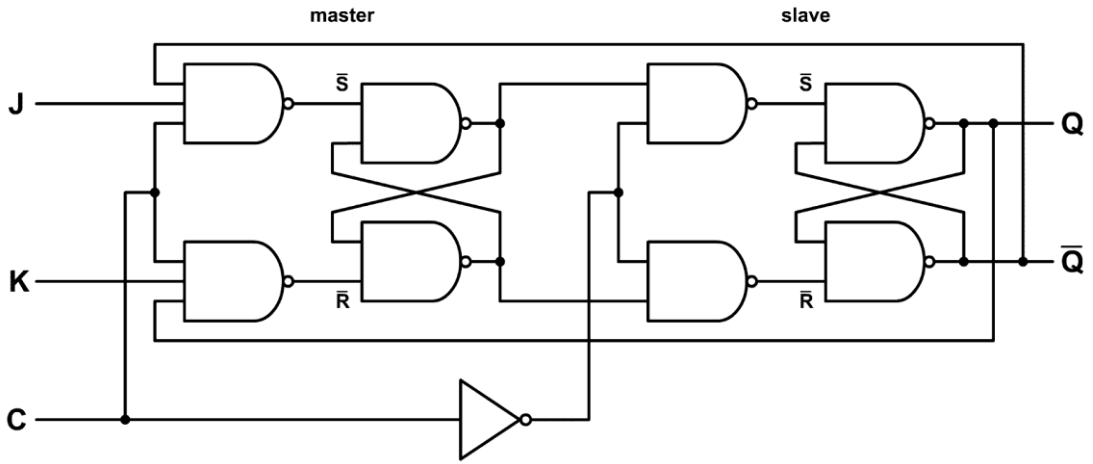


Funkce klopného obvodu D

D	E	Q_i	$\overline{Q_i}$
0	1	0	1
1	1	1	0
0, 1	0	$\boldsymbol{Q_{i-1}}$	$\boldsymbol{\overline{Q_{i-1}}}$

Klopné obvody (typu) J-K

- zachovává oba řídící signály: nastavení (J) – nulování (K)
- typ Master-Slave: dvoufázový (master, slave), synchronní řízení stavu vstupu J nástupní i sestupní hranou hodinového signálu na vstupu C



Obvody v počítačích

- **sériová sčítačka**: (aritmetické) sčítání log. hodnot dodávaných na vstupy v sériovém tvaru po jednotlivých řádech
- **paralelní registr** (střádač): vícebitová paměť pro hodnotu dodanou paralelně na více vstupů
- **sériový (posuvný) registr**: vícebitová paměť pro hodnotu dodanou sériově na vstupu, použití pro transformaci sériových dat na paralelní
- **čítač**: paměť počtu impulsů na hodinovém vstupu, binárně reprezentovaný počet na vícebitovém výstupu

Podrobnější informace

- **Od logických obvodů k mikroprocesorům : základy kombinačních a sekvenčních obvodů.** 1 / Jean-Michel Bernard, Jean Hugon, Robert le Corvec ; přeložili Vladimír Drábek, Jan Hlavička, Zdeněk Pokorný . Praha : SNTL - Nakladatelství technické literatury, 1984 204 s.
- prezentace na síti (Vyuka\KMI_YUDIT\yudit_prednaska_1_pocitac_informace.pdf)

Příště

- Základní deska počítače a interní sběrnice počítače (PCI). Princip činnosti mikroprocesoru (CPU) a vnitřních pamětí (RAM, cache). Interní součásti počítače, přídavné karty (grafická, zvuková, síťová,
multimediální), vnější paměti (pevné disky a disková pole). Periferie a externí sběrnice počítače (USB). Monitor (CRT, LCD), polohovací zařízení (klávesnice, myš aj.), datové mechaniky a média (floppy, CD, DVD), tiskárna a skener, modem.
- **Studijní texty:**
  - J. Hronek: Struktura počítačů
  - P. Tišnovský: Seriál Co se děje v počítači (http://www.root.cz/serialy/co-se-deje-v-pocitaci/)