(1) Historie vývoje VT

1.1 Prehistorie výpočetní techniky

George Boole (2.11.1815 – 8.12.1864)

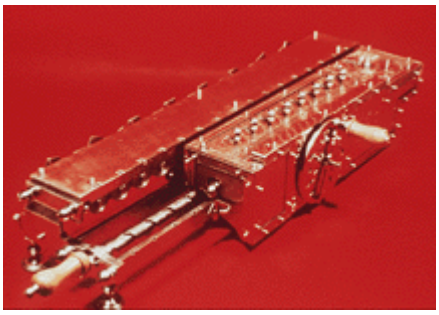
Britský matematik a filozof. Objevitel základů moderní aritmetiky (Booleova algebra)

Obr. 1: Pascalova sčítačka

1.2 Booleova algebra

Definuje vlastnosti (komutativnost, distributivnost, neutralitu (x+0=x), komplementaritu (x-x=0)) množinových a logických operací (AND, OR, NOT). Zavádí dvoustavovou logiku (pravda vs. nepravda).

Popisuje se třemi způsoby: pravdivostní tabulkou, Vennovými diagramy a matematicky



Obr. 2: Liebnitzova násobička

Pravdivostní tabulka

A	B	A + B	A x B	A'
0	0	0	0	1
0	1	1	0	1
1	0	1	0	0
1	1	1	1	0

1.3 Von Neumannova koncepce počítače (1946)

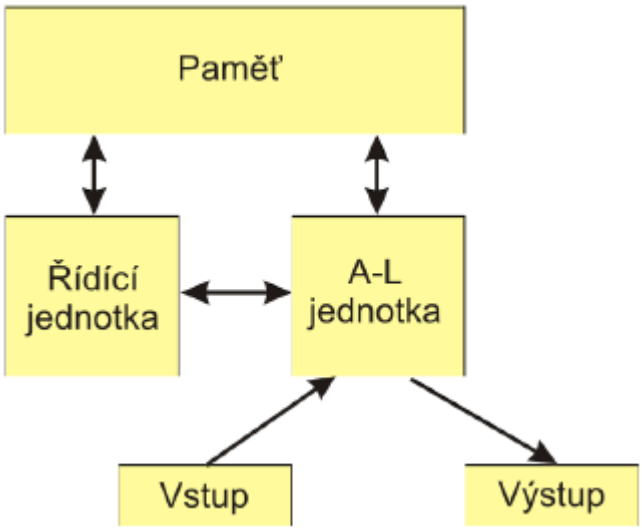
Základní moduly počítače

procesor, operační paměť, vstupní a výstupní zařízení

Základní principy

- Pracuje v dvojkové soustavě
- Programy a data jsou v operační paměti (nenačítají se z vnější paměti v průběhu výpočtu, jednotné kódování - k programům lze přistupovat jako k datům, umožnilo univerzalitu počítače, bezproblémové zavedení cyklů a podmíněného větvení)
- Programy, data, mezivýsledky a konečné výsledky se ukládají do téže paměti
- Paměť je rozdělená na stejně velké buňky, které jsou průběžně očíslované, přes číslo buňky (adresu) se dá přečíst nebo změnit obsah buňky
- Rychlost vnitřní paměti srovnatelná s rychlostí výpočetní jednotky
- Přímé adresování (přístup) - v libovolném okamžiku přístupná kterákoliv buňka paměti
- ALU tvoří pouze obvody pro sčítání čísel (ostatní operace se dají převést na sčítání)
- Struktura je nezávislá na zpracovávaných problémech, na řešení problému se musí zvenčí zavést návod na zpracování (program) a musí se uložit do paměti, bez tohoto programu není stroj schopen práce
- Po sobě jdoucí instrukce programu se uloží do paměťových buněk jdoucích po sobě, přístup k následující instrukci se uskuteční z řídící jednotky zvýšením instrukční adresy o 1
- Instrukcemi skoku se dá odklonit od zpracování instrukcí v uloženém pořadí
- Základní typy instrukcí:
  - aritmetické instrukce (sčítání, násobení, ukládání konstant,…)
  - logické instrukce (porovnání, not, and, or,…)
  - instrukce přenosu (z paměti do řídící jednotky a na vstup/výstup)

  - podmíněné skoky a ostatní (posunutí, přerušení, čekání,…)
- Všechna data (instrukce, adresy,…) jsou binárně kódované, správné dekódování zabezpečují vhodné logické obvody v řídící jednotce



Obr. 3: von Neumannovo schéma počítače

1.4 Umělé jazyky

- člověk-člověk
- člověk-stroj
  - **Programovací**
    - imperativní (procedurální) – posloupnost příkazů
      - strukturované
      - objektově orientované
    - deklarativní (neprocedurální)
      - funkcionální
      - logické
      - databázové (definiční, manipulační
  - **Značkovací**

1.5 Základní typy jazyků

Strukturované (Pascal)

Označuje programovací techniku, kdy se implementovaný algoritmus rozděluje na dílčí úlohy, které se spojují v jeden celek. K implementaci v programu se používá vybraných řídících struktur, ostatní struktury nejsou povoleny - u strukturovaného programování se např. nepoužívá řídící příkaz skoku.

```
Příklad:
program HelloWorld(output);
begin
  writeln('Hello, World!')
end.
```

Objektově orientované (Java, Ruby)

Programovací jazyky, které umožňují nebo podporují objektově orientované programovací techniky jako zapouzdření, dědičnost, modularita a polymorfismus. Za první OO jazyk je považována Simula (1967).

Příklad:

```
// Hello.java
public class Hello {
    public static void main(String[] args) {
        System.out.println("Hello World");
    }
}
```

Skriptovací (Python, Ruby)

Programovací jazyky, které slouží k jednodušší změně chování aplikací (může s nimi manipulovat uživatel, protože nejsou kompilované). V současné době se však používají i k psaní aplikací.

Příklad:

```
#!/usr/bin/env python
print 'Hello world!'
```

Funkcionální (Lisp, Scheme)

Na výpočet se dívají z pohledu matematických funkcí. Jsou založené na tzv. Lambda kalkulu. Dělí se na typované a netypované.

Příklad (faktoriál):

```
(defun factorial (n)
   (if (<= n 1)
     1
     (* n (factorial (- n 1)))))
```

Logické (Loglan)

Cílem je jednoznačnost výroků. Typicky jsou založeny na predikátové logice.

Značkovací (XML, XHTML)

Značkovací jazyk poskytuje způsob jak zkombinovat text a informace o něm. Metadata jsou vyjádřeny pomocí značek, které jsou v dokumentu samotném a „obalují“ text.

Příklad:

```
<?xml version="1.0" encoding="UTF-8"?>
<book>This is a book.... </book>
<!-- This is a comment. -->
```

1.6 Zdroje

http://linux.fjfi.cvut.cz/~sunie/akce/tcn/2003/sbornik/pdf/boolalgebra/boolalgebra.pdf

http://www.fi.muni.cz/usr/jkucera/pv109/sl1.htm

http://en.wikipedia.org/wiki/Boolean_algebra_(logic)

http://64.233.183.104/search?q=cache:q_J1lRVM460J:www.fi.muni.cz/usr/brandejs/AP/15010.html

http://cs.wikipedia.org/wiki/John_von_Neumann

http://skola.pozlovsky.net/maturitni_otazky/informatika/5.Von_Neumannovo_schema.pdf

http://en.wikipedia.org/wiki/Object_oriented_language

http://en.wikipedia.org/wiki/Markup_language

http://en.wikipedia.org/wiki/Structured_programming

http://cs.wikipedia.org/wiki/Funkcionální_programování