ním grafu do šířky, zatímco my jsme v naší úloze pracovali s ohodnoceným grafem a využili jsme proto Dijkstrův algoritmus. Zadání i řešení původní soutěžní úlohy uvádí internetový archív úloh MO kat. P uložený na adrese [4] a podrobněji ho popisuje článek [1].

Literatura

[1] *Töpfer, P.*: Úlohy z MO – kategorie P. MFI, roč. 9 (1999–2000), č. 6, s. 375–379.

[2] *Töpfer, P.*: Algoritmy a programovací techniky. Prometheus, Praha, 1995 a 2007.

[3] http://ksp.mff.cuni.cz/kucharky/halda-a-cesty/

[4] http://mo.mff.cuni.cz/p/archiv.html

Jak počítal první programovatelný počítač?

INGRID NAGYOVÁ

Pedagogická fakulta OU, Ostrava

Prvenství v konstrukci programovatelného počítače je přisuzováno německému inženýrovi Konrádovi Zuse, který v roce 1938 sestrojil první elektromechanický stroj. Tento stroj s názvem Z1 pracoval s čísly ve dvojkové soustavě a byl ovladatelný pomocí programu zadávaného na děrné pásce. Počítač Z1 byl velmi poruchový a proto nevhodný pro praktické využití. Odstartoval však vývoj dalších, daleko sofistikovanějších počítacích strojů a vedl nakonec ke konstrukci počítače tak, jak jej známe dnes. Počítače Konráda Zuse byly využity ve válce a před koncem války byly zničeny při náletu. Archiv prací Konráda Zuse lze dnes najít i na internetu [1]. K záznamům o konstrukci počítačů se o několik desetiletí později vrátil profesor Raúl Rojas, který ukázal, že počítače Z1 a Z3 jsou i přes absenci podmíněných skoků Turingovsky úplné. Zaměřil se také na rekonstrukci počítacích strojů Konráda Zuse [2]. Replika počítače Z1 se dnes nachází v Technickém muzeu v Berlíně. Panoramatickou vizualizaci počítače a simulaci výpočetního procesu lze také sledovat na [3].

Dále se seznámíme se stavbou a principy práce prvních počítačů. Jejich konstrukce je velice jednoduchá, lze ji vysvětlit žákům sedmých ročníků základní školy. I přesto, že předpokládá znalosti počítání ve dvojkové soustavě, kromě převodů čísel z a do dvojkové soustavy není pro simulaci práce prvních počítačů potřeba dalších matematických znalostí. Poznání principů práce prvních počítačů a výpočty na schématech reprezentujících model počítače mohou naopak objasnit spojitost počítačů s počítáním ve dvojkové soustavě.

Na začátku naznačíme nutné úpravy algoritmu pro sčítání čísel ve dvojkové soustavě s ohledem na konstrukci počítačů. Algoritmus bude dále aplikován na modelu prvního počítače. V závěru článku naznačíme, jak uvedené informace použít při seznámení žáků základní nebo střední školy s historií vzniku počítačů a s jeho základními principy, které platí dodnes.

Sčítání čísel ve dvojkové soustavě

Sčítání čísel ve dvojkové soustavě se řídí následujícími pravidly:

$$0 + 0 = 0 \tag{1}$$

$$1 + 0 = 0 + 1 = 1 \tag{2}$$

$$1 + 1 = 10 \tag{3}$$

Vztah (3) znamená, že jedna plus jedna je nula, s tím, že jednička v čísle $(10)_2$ se přenáší do vyššího řádu.

Pomocí vztahů (1) až (3) lze sečíst libovolná čísla ve dvojkové soustavě běžným způsobem odzadu. Výpočet je jednoduchý, pokud nemusí být uplatněno pravidlo (3), nebo pokud je toto pravidlo použito pouze v nejvyšším řádu.

$$\begin{array}{r} 1001 \\ 0010 \\ \hline 1011 \end{array} \qquad \begin{array}{r} 1010 \\ 100 \\ \hline 1110 \end{array} \qquad \begin{array}{r} 1001 \\ 1100 \\ \hline 10101 \end{array}$$

Další postup sčítání při uplatnění pravidla (3) představíme a upravíme tak, aby byl později realizovatelný na jednoduchém počítacím stroji. Navíc v informatice je strukturou paměti omezena práce s čísly jednotlivými bajty. Pro jednoduchost využijeme velikost jednoho bajtu, tj. 8 bitů a jednotlivá čísla ve výpočtu budeme zapisovat jako osmici čísel 0 nebo 1 – dané číslo ve dvojkové soustavě doplněné zleva vedoucími nulami.

V prvním kroku sečteme jednotlivá dvojková čísla běžným způsobem uplatněním pravidel (1) až (3) odzadu.

$$\begin{array}{cccccccc}
0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\
0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\
\hline
0 & 0 & 0 & 10 & 1 & 1 & 10 & 10
\end{array} \qquad \begin{array}{c} \text{(A)} \\ \\ \text{(4)} \end{array}$$

Výsledek kroku (A) tvoří posloupnost $c_1 \dots c_8$ čísel ve dvojkové soustavě, kde c_1 je první číslice zleva (0), c_2 je druhá číslice (opět 0) atd., c_8 je poslední číslice $(10)_2$.

Dále převedeme jedničky z čísel $(10)_2$ v posloupnosti $c_1 \dots c_8$ do vyššího řádu jejich přičtením k aktuálnímu výsledku. V první řádku následujícího výpočtu se nachází cifry posloupnosti $c_1 \dots c_8$ v řádu jednotek (viz posloupnost $c_1 \dots c_8$, pouze místo čísel $(10)_2$ jsou zde uvedeny 0). Do druhého řádku sepíšeme jedničky z čísel $(10)_2$ posloupnosti $c_1 \dots c_8$, které ale připočítáme o řád výše, tj. zapisujeme je posunuté o jedno místo vlevo.

$$\begin{array}{cccccccc}
0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
 & & 1 & & 1 & 1 & & \\
\hline
0 & 0 & 1 & 0 & 1 & 10 & 1 & 0
\end{array} \qquad \begin{array}{c} \text{(B)} \\ \\ \text{(5)} \end{array}$$

Výsledek kroku (B) tvoří posloupnost $a_1 \dots a_8$ čísel ve dvojkové soustavě. Tato posloupnost nemusí být konečným výsledkem sčítání dvou čísel ve dvojkové soustavě. Mohou se zde vyskytovat čísla $(10)_2$ vyžadující přenos do vyššího řádu (viz výsledek kroku (B)). Opakovaně proto uplatníme pravidlo z kroku (5). Pro získání konečného výsledku je v našem případě nutné uplatnit toto pravidlo ještě dvakrát.

$$\begin{array}{cccccccc}
0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\
 & & & & 1 & & & \\
\hline
0 & 0 & 1 & 0 & 10 & 0 & 1 & 0
\end{array} \qquad \begin{array}{cccccccc}
0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\
 & & & 1 & & & & \\
\hline
0 & 0 & 1 & 1 & 0 & 0 & 1 & 0
\end{array}$$

Počet opětovných úprav výsledku podle pravidla (5) může být i vyšší (než třikrát v našem případě). Opakování je sice přirozenou činností počítače, snížení počtu opakování vede ale k úspoře času výpočtu a paměťového místa v počítači. Vraťme se proto k posloupnosti $a_1 \dots a_8$, která vznikla jako výsledek kroku (B) a sledujme její vlastnosti.

Věta o úpravě výsledku

Pro posloupnost $a_1 \dots a_8$ platí následující tvrzení:

a) Hodnota $a_n \neq (10)_2$.

b) Pokud $a_i = (10)_2$ pro $i \neq n$, pak $a_{i-1} \neq (10)_2$ a $a_{i+1} \neq (10)_2$.

c) Pokud $a_i = (10)_2$ a $a_j = (10)_2$ pro $i < j$, pak existuje k, $i < k < j$ tak, že $a_k = 0$.

Při důkazu tvrzení je potřeba vyjít z konstrukce posloupnosti $a_1 \dots a_n$, viz kroky (A) a (B).

ad a) Tvrzení vyplývá přímo z konstrukce posloupnosti. Hodnota a_n odpovídá nultému řádu výsledného čísla. V kroku (B) se zde již nic nepřipočítává. Hodnota se upravuje pouze v případě, že $c_n = (10)_2$, tehdy je po úpravě hodnota $a_n = 0$.

ad b) Hodnota $a_i = (10)_2$ vznikne v kroku (B) pouze v případě, že $c_i = 1$ a $c_{i+1} = (10)_2$.

Hodnota a_{i+1} vzniká z $c_{i+1} = (10)_2$. Proto $a_{i+1} = 0$, pokud c_{i+2} neexistuje nebo pokud existuje a je jednociferné, nebo $a_{i+1} = 1$, pokud $c_{i+2} = (10)_2$. V obou případech je a_{i+1} jednociferné, tj. $a_{i+1} \neq (10)_2$.

Víme, že $c_i = 1$, pak $a_{i-1} = c_{i-1}$, pokud je c_{i-1} jednociferné, nebo $a_{i-1} = 0$, pokud je $c_{i-1} = (10)_2$. V obou případech je $a_{i-1} \neq (10)_2$.

ad c) Existence jednociferného čísla a_k v posloupnosti vyplývá z bodu b), pro $k = j - 1$ je $a_{j-1} \neq (10)_2$.

Pokud $a_i = (10)_2$, pak $c_{i+1} = (10)_2$. Pak existuje $n \geq 1$, že $\forall p \in \langle 1, n \rangle$, $p \in \mathbb{N}$: $c_{i+p} = (10)_2$. Pak uplatněním pravidla (4) $a_{i+n} = 0$ a $\forall p \in \langle 1, n \rangle$: $a_{i+p} = 1$. Hledané $k = i + n$. Podle předpokladu je $a_j = (10)_2$, tj. $k < j$.

Tvrzení b) říká, že v posloupnosti $a_1 \dots a_n$ se vyskytuje číslo $(10)_2$ vždy izolovaně, jako jednotlivé dvojciferné číslo. Z tvrzení c) vyplývá, že mezi dvěma hodnotami $(10)_2$ se vždy vyskytuje hodnota 0. Výpočet součtu dvou čísel v dvojkové soustavě může být dokončen tak, že posloupnost vzniklou v kroku (B) upravíme následovně:

Pokud $\exists j\colon a_j = (10)_2$, pak vyhledáme $k < j\colon a_k = 0$ a vybereme největší takové k. Výsledná posloupnost $t_1 \dots t_n$ vznikne z posloupnosti $a_1 \dots a_n$ tak, že $t_j = 0$, $t_k = 1$ a $\forall i \in \mathbb{N}$, $i \in \langle k+1, j-1 \rangle$: $t_i = 0$. (C)

Řečeno slovy – jednička se přenáší do vyššího řádu odzadu postupnou změnou všech jedniček před ní na nuly, až dokud nenarazíme na první nulu. Přepisem nuly na jedničku přenos končí.

Je důležité si uvědomit, že hodnota k nemusí obecně existovat, tj. $\forall k < j$: $a_k \neq 0$. Pokud by existovalo $a_k = (10)_2$ pro nějaké $k < j$, pak podle tvrzení c) existuje l, $k < l < j$ a $a_l = 0$. Pak bychom položili $k = l$ a pravidlo (C) by mohlo být uplatněno. Proto hledané k neexistuje pouze v případě, že $\forall k < j$: $a_k = 1$. V matematice problém řešíme přidáním dalšího číselného řádu – připsáním jedničky před výsledek. V informatice je číselné počítání omezeno strukturou paměti a „připsání jedničky před výsledek“ nemusí být z důvodu tzv. přetečení možné. V dalším budeme pro jednoduchost předpokládat, že k přetečení při výpočtech nedochází, tj. že součet čísel je omezen velikostí jednoho bajtu (číselně 0 až 255). Aplikací kroku (C) na posloupnost (5) dostáváme výsledné řešení: 00101(10)10 $\rightarrow$ 00110010.

Ukážeme si, jak lze naznačený algoritmus sčítání aplikovat při seznámení se s principy práce prvních programovatelných počítačů. Nejprve vystavíme schéma části počítače určené pro výpočty. Na schématu ukážeme proces sčítání dvou čísel v dvojkové soustavě. Současně s jednotlivými kroky algoritmu sčítání uvádíme úlohy, které lze pro nácvik algoritmu v dané etapě řešit.

Sčítání na prvních počítačích

Základní pravidla (1) až (3) pro sčítání dvojkových čísel lze vyjádřit pomocí dvou logických funkcí – exklusivního logického součtu (XOR) a logického součinu (AND) – viz tabulka 1. Funkce XOR určuje hodnotu výsledku sčítání a funkce AND případný přenos do vyššího řádu.

Tab. 1 Součet dvojkových čísel a funkce AND a XOR

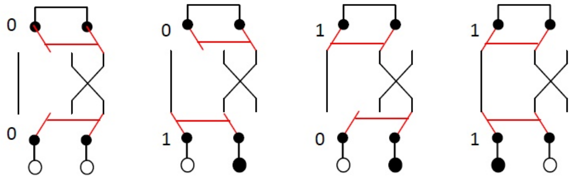
A	B	+	A AND B	A XOR B
0	0	0	0	0
0	1	1	0	1
1	0	1	0	1
1	1	10	1	0

Logickou funkci AND mohou zastoupit dva spínače řazené za sebou (obr. 1 vlevo). Proud obvodem prochází pouze v případě, že jsou oba spínače zapnuté. Logickou funkci XOR dostaneme překřížením spojení mezi spínači (obr. 1 vpravo) (křížení bez vodivého spojení). Proud obvodem prochází pouze v případě, že je jeden spínač v poloze nahoře a druhý v poloze dole, tj. jeden spínač je zapnutý a druhý vypnutý.

Obr. 1 Zapojené spínače – logické funkce AND a XOR

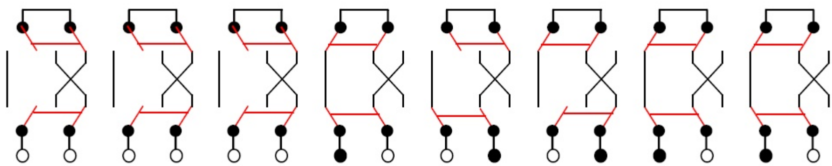
Kombinací propojení pro logické funkce AND a XOR lze získat hradlo, které dokáže sečíst dvě jednociferná dvojková čísla (obr. 2). Jedno hradlo obsahuje čtyři přepínače, vždy dva a dva jsou spřažené – spojují a rozpojují se současně. Každá dvojice spínačů umožňuje nastavit dva stavy:

0 (vypnuto) a 1 (zapnuto). Vlevo je propojení pro funkci AND – její výsledek je vlevo dole. Propojení vpravo odpovídá logické funkci XOR (výsledek funkce je vpravo dole). Výsledky funkcí jsou vypočteny automaticky na základě propojení spínačů.



Obr. 2 Hradlo pro sčítání dvou jednociferných dvojkových čísel

Podle předpokladu práce s jedním bajtem budeme sčítat maximálně osmiciferná dvojková celá čísla, tj. čísla v rozmezí 0 až 255 (jeden bajt). K provedení kroku (A) podle naznačeného algoritmu budeme potřebovat osm hradel (obr. 3), součet čísel $(101)_2 = (5)_{10}$ a $(11010)_2 = (26)_{10}$ (na obrázku jsou čísla doplněna zleva vedoucími nulami). Dolu na hradlech (větší kolečka) získáváme výsledek výpočtu. Pokud výpočet nevyžaduje přenos do vyššího řádu, je výsledek na hradlech konečný a lze ho odečíst podle hodnot na hradlech vpravo: $(00011111)_2 = (31)_{10}$.



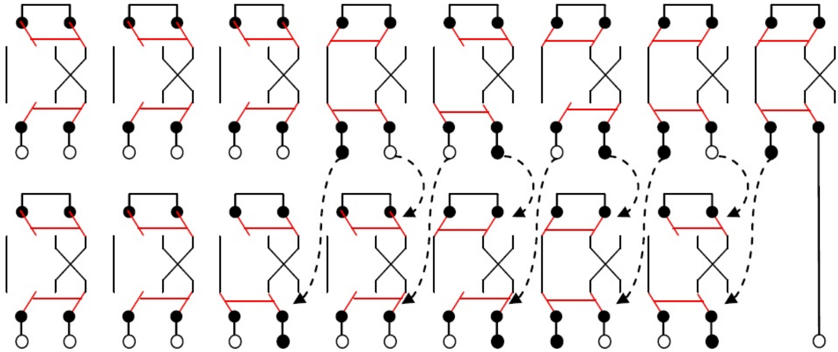
Obr. 3 Sčítání dvojkových čísel na řadě hradel

Úloha 1

Na řadě osmi hradel sečtěte daná čísla. Odečtěte výsledky a zkontrolujte správnost převodem do desítkové soustavy.

a) $(1001)_2 + (10)_2$ b) $(101011)_2 + (10000)_2$ c) $(20)_{10} + (5)_{10}$

Situace je komplikovanější, pokud je v některém kroku výpočtu nutný přenos do vyššího řádu, tj. když je nutný převod jedniček z dvojkových čísel $(10)_2$ (obr. 2, hradlo vpravo). K provedení kroku (B) budeme potřebovat řadu dalších sedmi hradel (obr. 4). Výsledné hodnoty na horní řadě hradel ovlivní nastavení dolní řady hradel následovně (značeno na obr. 4 šipkami):



Obr. 4 Přenos jedniček do vyššího řádu

- Hodnota na výstupu horního hradla vpravo ovlivní horní spínač dolního hradla.
- Hodnota na výstupu horního hradla vlevo ovlivní dolní spínač dolního hradla nacházejícího se v řadě na předchozím místě (přenos do vyššího řádu).

Úloha 2

Na dvouřadové soustavě hradel sečtěte daná čísla. Odečtěte výsledky a zkontrolujte správnost převodem do desítkové soustavy.

a) $(1001)_2 + (1010)_2$ b) $(101011)_2 + (100010)_2$ c) $(24)_{10} + (19)_{10}$

Na závěr je nutné provést konečnou úpravu výsledku. Postupujeme podle závěrů vyplývajících z věty o úpravě výsledku – krok (C). Přenos provedeme v dolní řadě spínačů dolních hradel (obr. 5), přenos opět naznačují šipky. Konečný výsledek lze odečíst z dolní řady hradel podle hodnoty na hradle vpravo – označeno šedými kolečky.

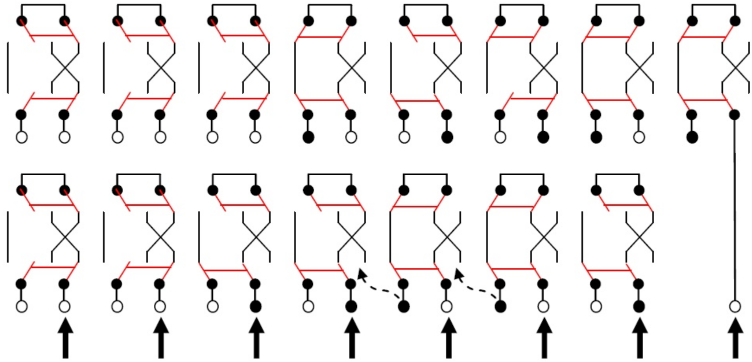
V praxi na prvním počítači Konráda Zuseho bylo nutné přenos (viz krok (C)) realizovat pomocí dalších hradel a jejich speciálního zapojení, které zde nebudeme dále rozebírat. Princip spočívá v opakovaném využití dolní řady spínačů, jak je to v úvodu naznačeno v kroku (B).

Vzhledem k omezení počtu hradel v řadě na 8 lze naznačeným algoritmem správně sčítat libovolná čísla, jejichž součet nepřekročí 255.

Úloha 3

Na dvouřadové soustavě hradel sečtěte daná čísla. Odečtěte výsledky a zkontrolujte správnost převodem do desítkové soustavy.

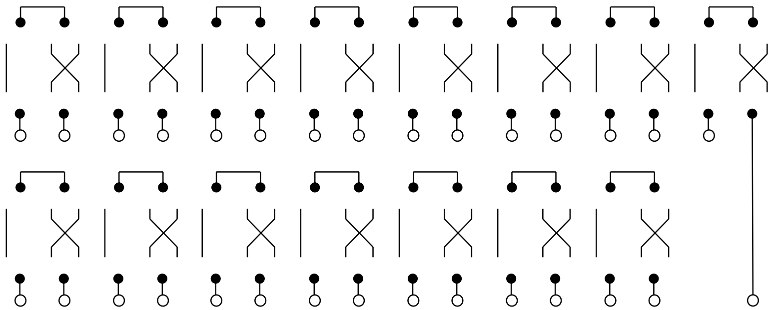
a) $(1111)_2 + (1010)_2$ b) $(101011)_2 + (100110)_2$ c) $(25)_{10} + (19)_{10}$



Obr. 5 Konečná úprava – přenos do vyššího řádu

Využití ve výuce

Kromě základních historických faktů, poměrně málo známých, lze uvedený postup použít při práci s žáky základní nebo střední školy. Žáci pracují s šablonou (obr. 6) a postupně se seznamují s postupem výpočtu. Využíváme k tomu úlohy se stoupající obtížností tak, jak jsou uvedené výše.



Obr. 6 Šablona pro práci žáků

Čísla podle zadání žáci nejprve převedou do dvojkové soustavy. Podle dvojkové reprezentace čísel vyznačí situaci na horní řadě hradel – naznačí polohu spínačů a výsledek v hodní řadě hradel (větší kolečka). Pro jednoduché příklady (viz úloha 1) zde výpočet skončí.

Složitější úlohy vyžadují využití dolní řady spínačů. Postup je naznačen v textu – od nastavení spínačů dolní řady podle výsledků v horní řadě až po přenos (viz krok (C)).

V praktické výuce se ukazuje pro žáky paradoxně nejobtížnější krok A, a to zejména zápis čísel do horní řady hradel. Pochopení práce spřažených spínačů se ukazuje značně obtížné i přesto, že se jedná o jednoduché překreslování z tabule (dataprojektoru) podle obr. 2. Žáci, ale ze zkušenosti ani studenti na vysoké škole, se s podobnými koncepty obvykle ještě nesetkali a ve výuce je nutné počítat s delším časem než lze s žáky postoupit dále. Samotný algoritmus pak už není pro pochopení natolik obtížný.

Nutno poznamenat, že při práci s žáky nezmiňujeme důvody, proč sčítání doopravdy funguje, jak to naznačuje tento článek. Zaměřujeme se pouze na zvládnutí práce podle daného algoritmu. Vybrané části algoritmu vysvětlujeme pouze žákům, kteří projeví o tyto informace zájem, a to vždy s ohledem na jejich schopnost porozumění.

Ze zkušenosti lze doložit, že práci se schématem prvního počítače podle daného algoritmu zvládnu žáci 7. třídy.

Téma prvních počítačů a jejich konstrukce je podle informací učitelů ve výuce informatiky velice časté. Učitele ho rádi zapojují, protože jako jedno z mála témat umožňuje frontální výklad. Přesto obecné povědomí o konstrukci prvních počítačů je, podle rozhovorů se studenty učitelství informatiky, poměrně zkreslené.

Domníváme se, že výklad pravdivých historických faktů doplněný o konkrétní praktickou ukázku práce prvního počítače může být dobrou a hlavně správnou motivací pro žáky a jejich vztah k informatice. Schéma prvního počítače, pochopení algoritmu výpočtu a schopnost postupovat podle tohoto algoritmu, jsou základními koncepty informatiky a informatického myšlení. Tyto koncepty, přesto, že ukazují na pravou podstatu informatiky jako vědy, jsou ve výuce informatiky na školách často opomíjeny.

V rozhovoru se studenty učitelství informatiky se často ukazuje, že práce s dvojkovou soustavou vnímají jako matematické téma, které s informatikou souvisí pouze okrajově. Práce se schématem prvního počítače dává smysl a význam výuce převodů mezi číselnými soustavami a umožňuje pochopit myšlenky, které stály u zrodu prvních počítačů.

Literatura

[1] *Konrad Zuse Internet Archive.* [online] [cit. 2016-07-15] Dostupné z: http://zuse.zib.de/

[2] *Rojas, R.*: Reconstruction of the Z1 computer. [online] [cit. 2016-07-15] Dostupné z: http://dcis.inf.fu-berlin.de/rojas/reconstruction-of-the-z1-computer/

[3] Architecture and Simulation of the Z1 Computer. [online] [cit. 2016-07-15] Dostupné z: http://zuse-z1.zib.de/index.html