29. Aritmetické obvody (výpočet aritmetických operací pomocí číslicových obvodů)

Polyadické číselné soustavy - opakování

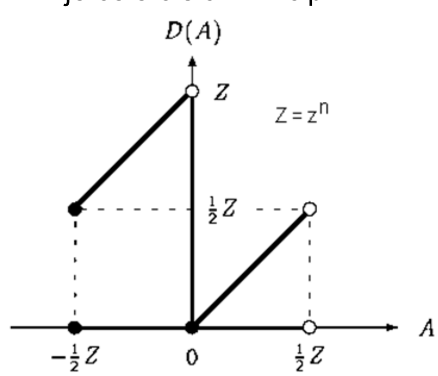
z-adické číslo v z-adické soustavě:
$A = a_n z^n + \ldots + a_0 z^0 + a_{-1} z^{-1} + \ldots + a_{-m} z^{-m}$
z je základ polyadické soustavy $z>=2$
(nejčastěji z = 2, 8, 10, 16)
a_i je z-adická číslice $0<=a_i<z$ („z“ různých a_i)
$a_{min} = 0$ $a_{max} = z – 1$
$A_{min} = 0$ $A_{max} = z^{n+1} – z^{-m}$
mezi řádem „0“ a „-1“ píšeme desetinnou čárku
standardně uvažujeme jen nezáporná čísla

Zobrazování záporných čísel

- Přímý kód – přidáme znaménkový bit
  + … 0 - … 1 (např.: –6 … 1 0110)
  složité aritmetické operace – nepoužívá se
- Jednotkový doplněk (inverzní kód)
  $1D(A) = z^n – A –1$
  např. A = 10101, z = 2, n = 5, 1D(A) = 01010
  problém dvou nul: 0 = (0000)2, -0 = (1111)1D
- Dvojkový doplněk
  $2D(A) = z^n – A = 1D(A) +1$
  např. A = 356, z = 10, n = 4, 2D(A) = 9644

Doplňkový kód

Vychází z dvojkového doplňku: D(A) = A (pro A>=0) a z^n - A (pro A<0). Ve dvojkové soustavě bude mít obraz nezáporného čísla v nejvyšším řádu 0, obraz záporného čísla 1.
Nutno vyloučit stejné obrazy (dvojznačnost) - např.: D(534) = 543, D(-457) = 1000 – 457 = 543.
Proto předpokládáme, že A splňuje podmínku: -0,5 * z^n <= A < 0,5 * zn, „z“ uvažujeme sudé (2, 8, 10, 16) z toho vyplývá že 0,5 * z^n je celé číslo. Princip:



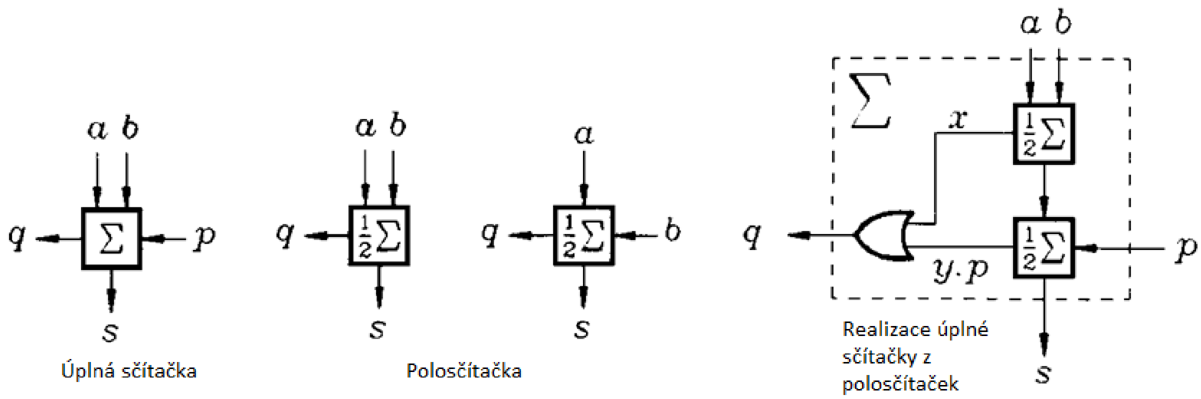
Sčítání a sčítačky

Sčítání se provádí stejně jako jsme zvyklí v desítkové soustavě:

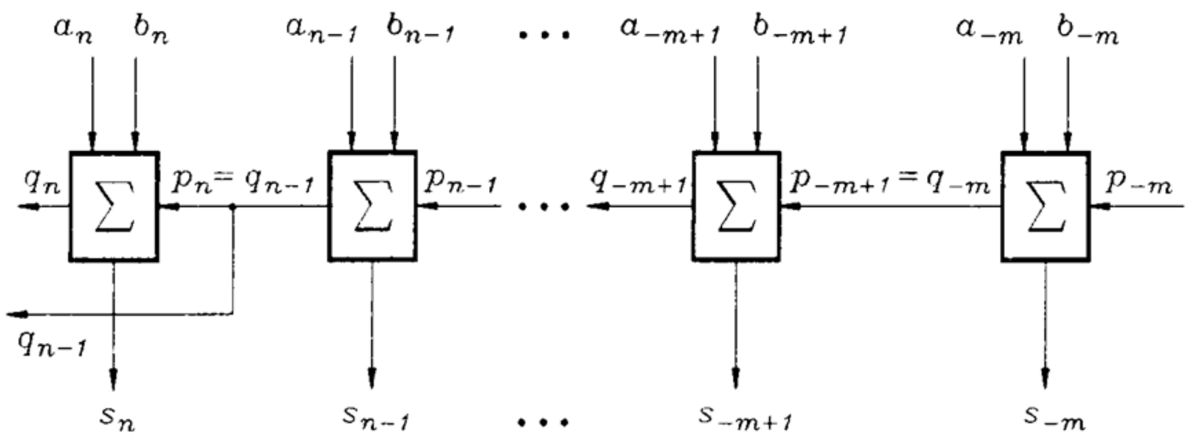
	1684_{10}		11_{10}		01011_2
+	0684_{10}	+	06_{10}	+	00110_2
-----------------		-----------------		-----------------	

= 2368_{10} nebo 17_{10} tedy = 10001_2

Značení dvojkové sčítačky a polosčítačky:

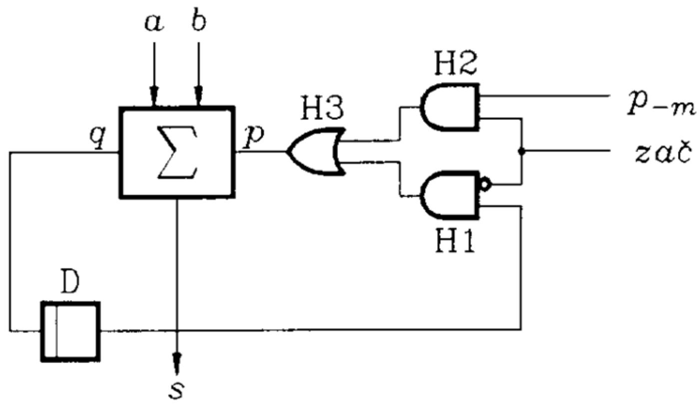


Paralelní dvojková sčítačka:
Na každé sčítačce jsou sečteny dva bity a přenos je předán další sčítačce.
Z obvodu jsou vyvedeny q_n a q_{n-1}, důvod bude vysvětlen později.

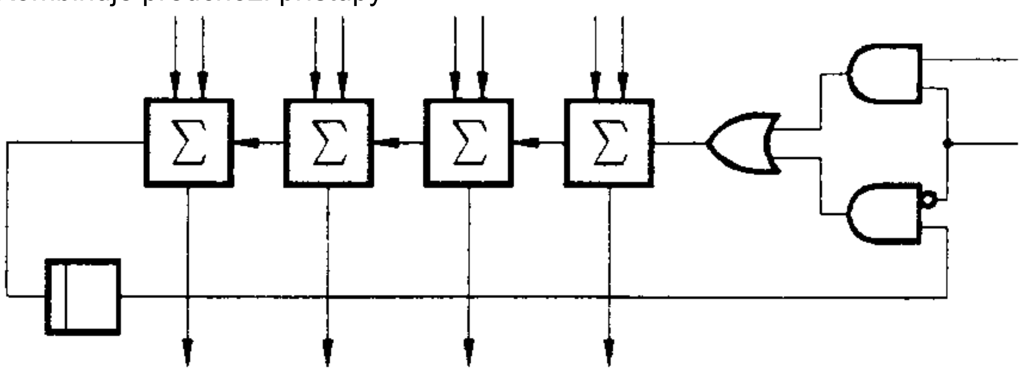


p_i .... přenos do řádu i ....carry
q_i .... přenos z řádu i

Sériová dvojková sčítačka:
Vstup “zač” by měl být aktivní (log. 1) pouze na začátku = sčítačka akceptuje první přenos. Poté by měl být změněn na log. 0 a sčítačka si přenos přebírá z registru D do kterého poté předá svůj přenos.



Serio-paralelní dvojková sčítačka:
Kombinuje předchozí přístupy.



Sčítačka pro nezáporná čísla:
Pokud chceme sčítat pouze kladná čísla je možné realizovat to stejnou sčítačkou.
Pouze je potřeba kontrolovat přenos do dalšího řádu.

Pokud je přenos 0 je na výstupu výsledek sčítání.

Pokud je přenos 1 došlo k přeplnění (overflow) a na výstupu výsledek - M, kde M je 2^n, kde n je počet bitů sčítačky (n-bitová sčítačka).

Např. máme 3bitovu sčítačku a chceme sečíst 111 + 001 výsledkem bude 0 a přenos 1.
Správným výsledkem je tedy $0+M = 0+2^3 = 8 = 1000_2$

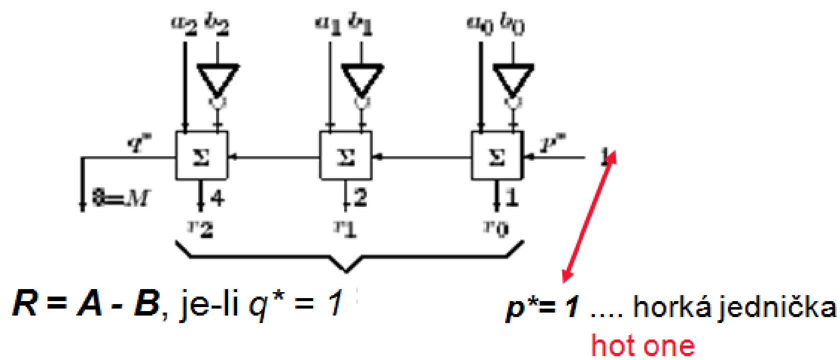
Odečítání

Odečítání je převedeno na sčítání doplňků.

1. Jednotkový doplněk = u jednoho čísla se 1 přemění na 0 a 0 na 1 (inverzní kód) a výsledný přenos je nutné k výsledku přičíst.
2. Dvojkový doplněk = změníme nuly na jedničky (a naopak) a přičteme jedničku, přenos se ztrácí (je li výsledek záporný, jeho absolutní hodnota se získá opět dvojkovým doplňkem).

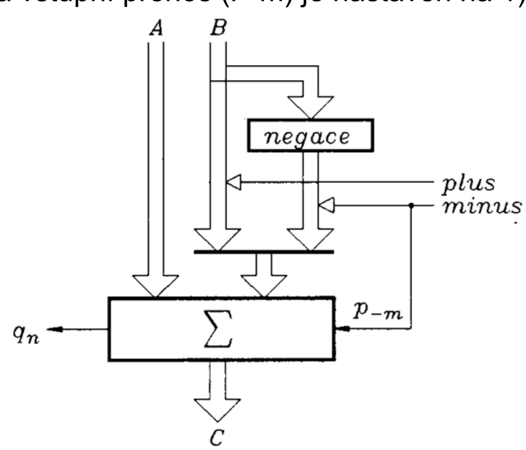
U odečítání narozdíl od sčítání neuvádíme přenos ale výpůjčku.
Sčítání: q = carry = přenos
Odečítání v = borow = výpůjčka
Výpůjčka je negací přenosu tedy pokud je přenos 1, výpůjčka je 0 a výsledek je větší roven 0, pokud je přenos 0 (výpůjčka = 1) výsledek je menší jak 0.

Odčítačka pro nezáporná čísla: (“horká jednička” kvůli dvojkovému doplňku, kde se neguje a **přičítá 1**)



Sčítačka odčítačka

(pokud chceme sčítat vstup B je přiveden normálně, pokud chceme odčítat je vstup B znegován a vstupní přenos (P-m) je nastaven na 1)



Zde je nutné ošetřit tzv. “přeplnění” (“přeplnění” není to stejné jako přenos nebo výpůjčka).
Přeplnění může nastat ve dvou případech:

- záporné číslo + záporné číslo = kladné číslo

- kladné číslo + kladné číslo = záporné číslo

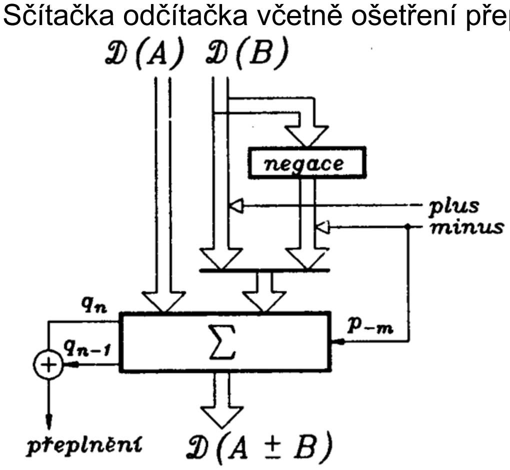
Jak ovšem přeplněnídetekovat? Odpovědí je pravdivostní tabulka nejvyššího řádu:

vstup A	vstup B	přenos do řádu P	přenos z řádu Q	výstup
0 (kladné číslo)	0 (kladné číslo)	0	0	0 (kladné číslo)
0 (kladné č.)	**0 (kladné č.)**	**1**	**0**	**1(záporné č.)**
0 (kladné číslo)	1 (záporné číslo)	0	0	1 (záporné číslo)
0 (kladné číslo)	1 (záporné číslo)	1	1	0 (kladné číslo)
1 (záporné číslo)	0 (kladné číslo)	0	0	1 (záporné číslo)
1 (záporné číslo)	0 (kladné číslo)	1	1	0 (kladné číslo)
1 (záporné č.)	**1 (záporné č.)**	**0**	**1**	**0 (kladné č.)**
1 (záporné číslo)	1 (záporné číslo)	1	1	1 (záporné číslo)

přeplnění lze tedy detekovat 2 způsoby:

1. porovnání přenosů
2. porovnání vstupů a výstupů

Sčítačka odčítačka včetně ošetření přeplnění



Sčítačka odčítačka pro nezáporná čísla

Je náročná na konstrukci protože detekce přeplnění je závislá na operaci

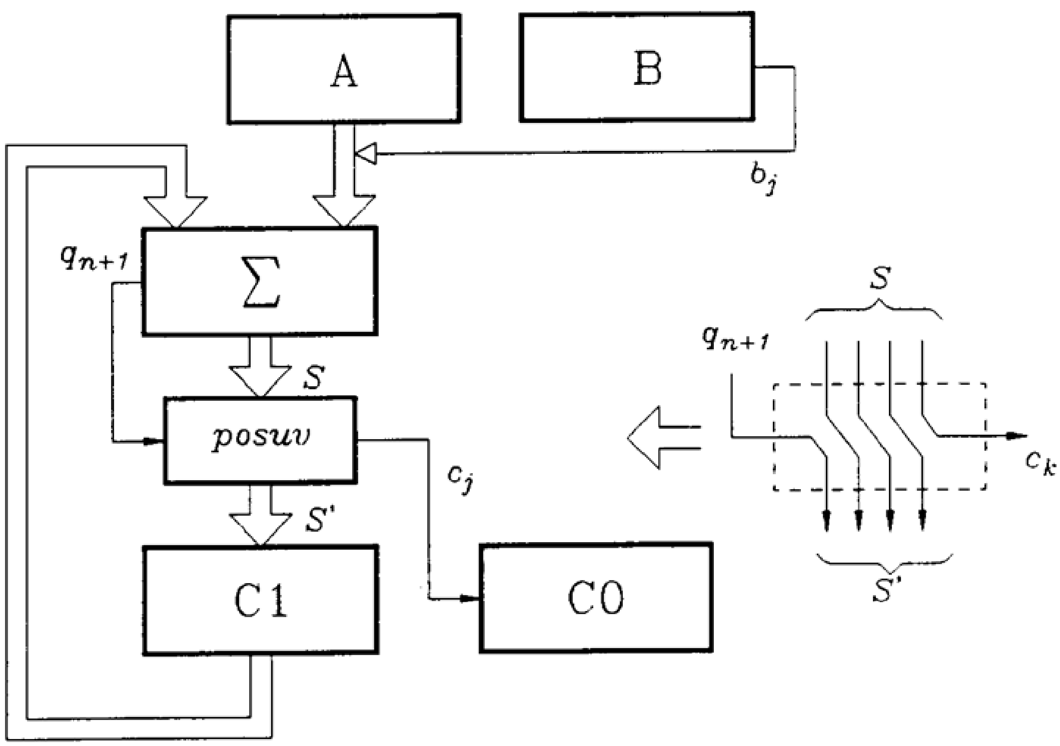
- sčítání - přeplnění je detekováno přenosem tedy přenos = 1
- odčítání - přeplnění je detekováno výpůjčkou tedy přenos = 0

Násobení ve dvojkové soustavě

Lze realizovat postupnými součty a posuvy.

	07_{10}			000111_2
*	05_{10}		*	000101_2
------------------			-------------------	
	35_{10}	tedy	=	100011_2

operace	C0	C1
počátek	0000	000
přičtení 111	0111	000
posuv	0011	100
přičtení 000	0011	100
posuv	0001	110
přičtení 111	1000	110
posuv	0100	011



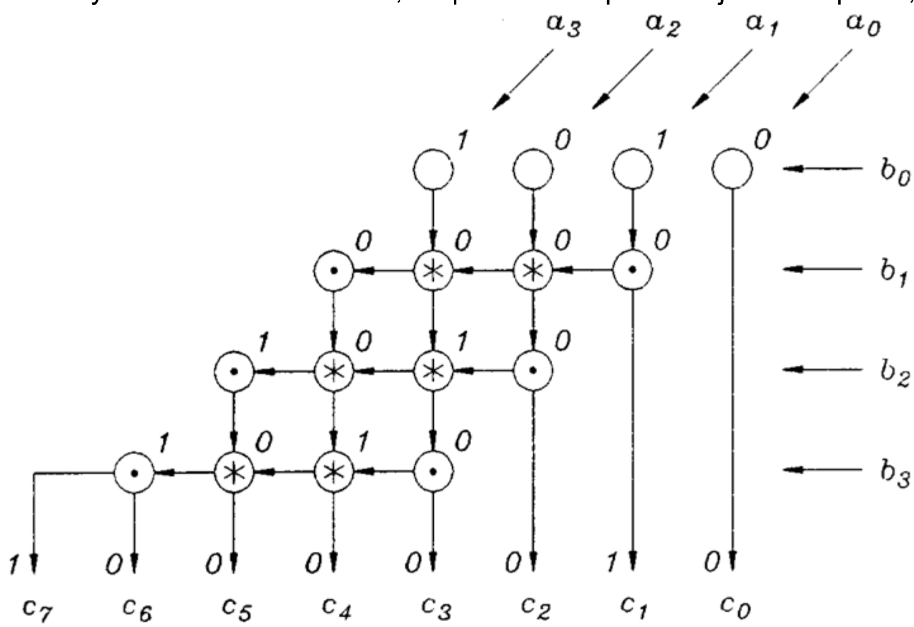
Násobení záporných čísel:
Lze realizovat dvěma způsoby:

1. Násobit absolutní hodnoty a vyhodnotit znaménko
2. Násobit přímo v doplňkovém kódu (obrázek pro dobrovolníky viz CIT přednáška 10 - klidně pošlu)

Paralelní násobičky
Násobení se provádí během jednoho cyklu podle vzorce:

$$C = \sum_{i=0}^{m} \sum_{j=0}^{n} a_i b_j . 2^{i+j}$$

a_i * b_i lze realizovat jako logický součin
2^{i+j} udává posuv o i+j míst
Vzniká pravidelná struktura se součinovými hradly (kroužky)
- kroužky označené hvězdičkou, resp. tečkou reprezentují navíc úplnou, resp. poloviční sčítačku



Dělení des. čísel ve dvojkové soustavě

Lze realizovat dvěma způsoby:

1. S návratem přes nulu - odečítá se příslušně posunutý dělitel dokud nedostaneme záporný výsledek např:

$$
\begin{array}{lll}
0{,}100 : 0{,}101 = 0{,}110\ldots & & \\
\underline{-\ 101} & & \rightarrow \\
\quad -1 & 0, & \\
\underline{+101} & & \\
\ 1000 & & \\
\ \underline{-101} & & \\
\quad 110 & 1 & \\
\ \ \underline{-101} & & \\
\qquad 10 & 1 & \\
\quad \underline{-101} & & \\
\quad\ -11 & 0 & \\
\quad \underline{+101} & & \\
\qquad 10 & &
\end{array}
$$

2. Bez návratu přes nulu - bez restaurace nezáporného zbytku např:

$$
\begin{array}{lll}
0{,}100 : 0{,}101 = 0{,}110\ldots & & \\
\underline{-\ 101} & & \rightarrow \\
\quad -10 & 0, & \\
\ \underline{+101} & & \\
\quad 110 & 1 & \\
\ \ \underline{-101} & & \\
\qquad 10 & 1 & \\
\quad \underline{-101} & & \\
\quad\ -11 & 0 & \\
\quad \underline{+101} & & \\
\qquad 10 & &
\end{array}
$$

Zvýšení rychlosti dělení

Dělení převrátíme na násobení převrácenou hodnotou a k násobení se použije rychlá paralelní násobička.

$$C = \frac{A}{B} = A.B^{-1}$$

B^{-1} lze získat iteračního vzorce a limity jdoucí k nekonečnu.... (v každém iteračním kroku získáme dvojnásobný počet platných bitů)

Plovoucí řádová čárka

Pro zobrazení reálných nebo příliš velkých celých čísel se využívá uložení pomocí mantisy a exponentu.

Mantisa = přesnost čísla

Exponent = rozsah

Rozlišují se dvě přesnosti:

1. Jednoduchá = znaménko mantisy + 23b mantisa (přímý kód) + 8b exponent (kód s posunutou nulou)
2. Dvojnásobná = znaménko mantisy + 52b mantisa (přímý kód) + 11b exponent (kód s posunutou nulou)

Druhá odmocnina
Využívá se vztahu, že odmocnina je větší než maximální počet nejmenších přirozených lichých čísel, která lze od A odečíst
Postupné odečítání po sobě jdoucích lichých čísel dokud nedostaneme nulu nebo záporný zbytek
Př:
odmocnina z 25 = 5
výpočet $25 – 1_1 – 3_2 – 5_3 – 7_4 – 9_5 = 0$