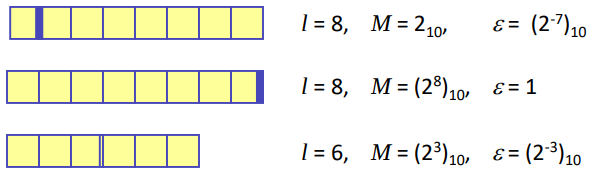
**BI-SPOL-29 Kódy pro zobrazení čísel se znaménkem a realizace aritmetických operací (paralelní sčítačka/odčítačka, realizace aritmetických posuvů, dekodér, multiplexor, čítač). Reprezentace čísel v pohyblivé řádové čárce**

BI-SAP

**Řádová mřížka**

* určuje formát zobrazitelných čísel (tj. definuje nejvyšší řád n a nejnižší řád -m)
* Délka ř.m. (l) – počet řádů obsažených v ř.m.
* Jednotka ř.m. (e) – nejmenší číslo zobrazitelné v ř.m. (nezáporné!)
* Modul ř.m. (M) – nejmenší číslo, které již v ř.m. zobrazitelné není
* první bit se většinou nechává pro znaménko
* (to tlustý je řádová čárka)



-m = -7, n = 0

-m = 0, n = 7

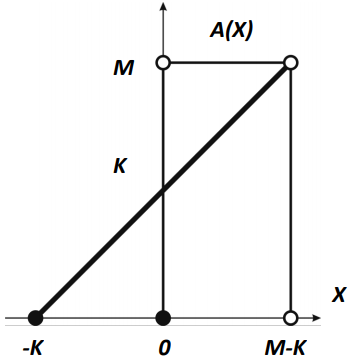
-m = -3, n = 2

* vhodné třeba pro zobrazení čísel bez znaménka (všechny bity využity pro číslo)

### Kódy zobrazení čísel se znaménkem

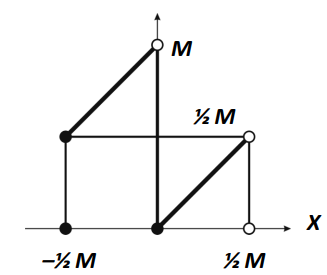
**Přímý kód**

* nejvyšší řád představuje znaménko (+ 0, - 1)
* jinak zůstávají kladná i záporná čísla stejné ( např. 1101 = -5, 0101 = 5)
* existuje záporná nula



**Aditivní kód**

* kód s posunutou nulou (tzn. jako 0, počátek, se bere nejnižší číslo)
* většinou se interval dělí na dva stejně velké v záporném i kladném intervalu (záporný je o 1 větší)
* např. pokud máme rozsah -4...3, tak -4 = 000 a 3 = 111

**Doplňkový kód**

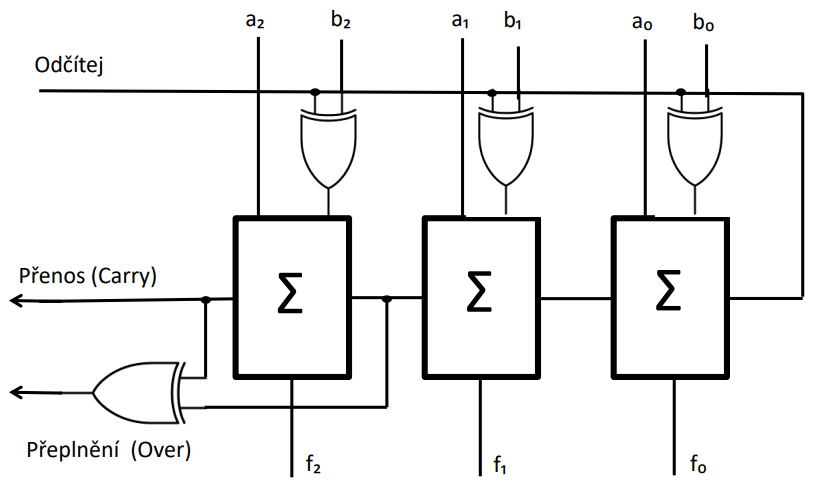
* znaménko je určeno prvním bitem zleva (opět +0, -1) ale je zároveň součástí obrazu
* pokud chceme získat zápornou hodnotu daného čísla, tak znegujeme a přičteme 1 (např. 3 = 011, neg = 100 + 1, -3 = 101)
* kladná čísla klasicky, ale jakmile je první bit 1, tak to spadne do nejnižšího záporného čísla

(např. -4 =100 a -3 = 101)

### Realizace aritmetických operací

**Paralelní sčítačka/odčítačka**

* pro doplňkový kód



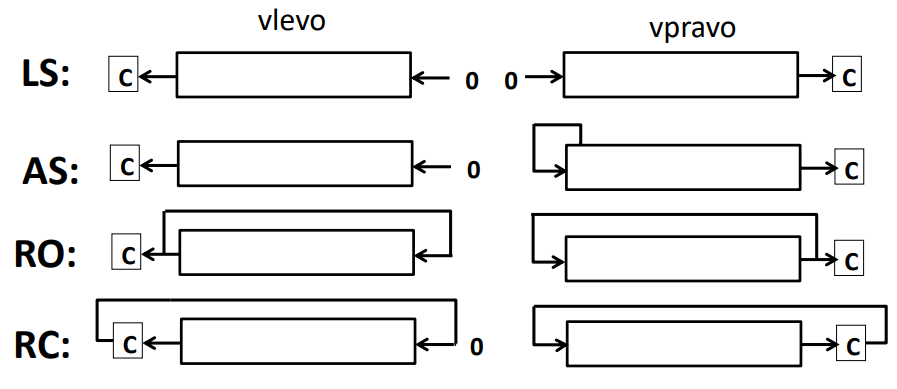
Σ – jednoduché bitové sčítačky s carry.

Odčítej (vstup) - 0/1, XOR

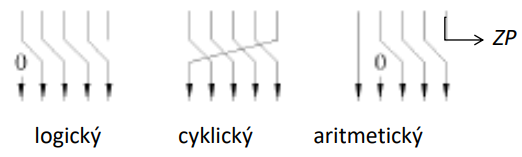
* sčítej (0): vstup B se nemění
* odčítej (1): B se invertuje a přičte 1 (viz doplňkový kód)

**Realizace aritmetických posuvů**

* *Logický posuv* (Shift, LS): posouvám pole vpravo/vlevo, na nová místa se zapisuje 0, může dojít ke ztrátě přesnosti
* *Aritmetický posuv vpravo/vlevo*: záleží na způsobu zápisu čísla (přímý, doplňkový), pro levý posuv výsledek odpovídá násobení 2, ztráta přesnosti se ORuje, přetečení XORuje
  + *přímý*: zachování znaménka, zbytek se bude posouvat (+ vkládat 0)
  + *doplňkový*: kopíruje se první bit a při násobení se bude XORovat s carry
* *Cyklický posuv* (rotace nebo rotace přes Carry flag RC): číslice, co vypadnou, se ve stejném pořadí dostanou na volné bity

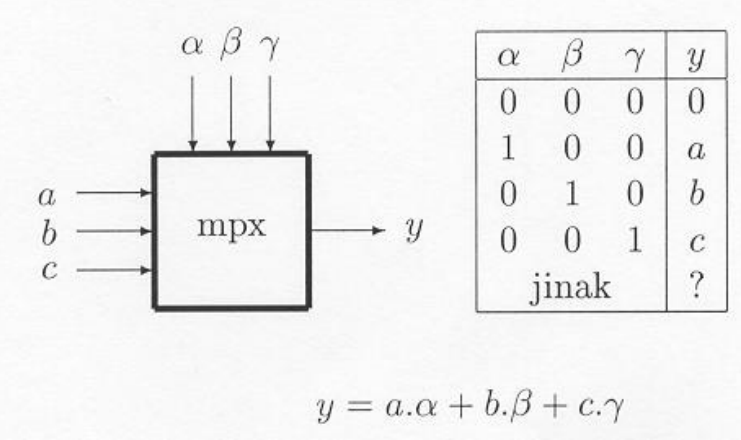


**Příklad posuvů přímého kódu o jedno místo vpravo**



**Dekodér 1 z N** - přijde zakódovaný vstup (např. číslo v binárce) a výstup 1 je pouze pro jednu žárovku, protože odpovídá jen ona (tzn. přijde 0101 a rozsvítí se 5. žárovka z 16)

**Multiplexor** - mám několik vstupů a vybírám jen jeden aby šel na výstup



**Demultiplexor** - opak multiplexoru, přesměruje 1 vstup na několik výstupů

**Čítač**

* speciální typ registru, který v sobě zahrnuje funkci inkrementu (dekrementu)
* úplné (modulo M2^n) a neúplné (modulo zbytek)
* většinou v binárním kódu
* lze sestavit Moorym nebo Mealem

### Reprezentace čísel v řádové pohyblivé čárce

Čísla s pevnou řádovou čárkou mají výrazně omezený rozsah, pro zvětšení rozsahu přidíme exponent e

X·z^e … odpovídá posunu řádové čárky v čísle X o e

z… typ soustavy (2 = binární)

Řádová mřížka má dvě podmřížky:

* *mantisa* (m) - informace o „hodnotě“ čísla, často zlomkový tvar
* *exponent* (e) - informace o pozici řád. čárky, celé číslo
* obě používají kódy pro zobrazení čísel se znaménkem

**Normalizovaný tvar čísla** - mantisa už nelze více posunout vlevo

**Princip skryté jedničky** - pokud je normalizovaný tvar čísla, tak je v nejvyšším řádu mantisy je vždy 1, ta se může vynechat a zvětšit tím rozsah

