**ARHITEKTURA**

**SISTEMI 1**

* Arhitektura so atributi, vidni programerju
* Nabor ukazov, število bitov za predstavitev podatkov, V / I mehanizmi, tehnike naslavljanja.

**ORGANIZACIJA**

* Organizacija je način same izvedbe funkcionalnosti
* Krmilni signali, vmesniki, pomnilniška tehnologija.

**ARHITEKTURA IN ORGANIZACIJA**

* Vsa družina Intel x86 ima enako osnovno arhitekturo
* Družina IBM System / 370 ima enako osnovno arhitekturo
* To omogoča združljivost kode

• Vsaj za nazaj

* Organizacija se razlikuje med različicami procesorja

**Control Mechanism**

**VLOGE IN SESTAVA**

* Vloge so definirane z delovanjem posameznih komponent kot del sestave
* Sestava določa način, kako so komponente povezane med seboj

**Data Processing Facility**

**Data Storage Facility**

**VLOGE**

Vse vloge računalnika so:

* OBDELAVA PODATKOV
* SHRANJEVANJE PODATKOV
* PRENOS PODATKOV
* NADZOR

**PREGLED SESTAVE**

**Data MovementApparatus**

**Operating Environment**

**(source and destination data)**

**ŠTEVILSKI SISTEMI**

**DECIMALNI SISTEM**

* Sistem, ki temelji na decimalnih števkah za prikaz števil

**0, 1, 2, 3, 4, 5, 6, 7, 8, 9**

* Številka 83 na primer pomeni osem desetic plus tri:

83 =

* Številka 4728 pomeni štiri tisočice, sedem stotic, dve desetici in osem:

4728 =

* **OSNOVA**
* Desetiški sistem ima osnovo 10. To pomeni, da se vsaka števka števila pomnoži z 10 na potenco, ki ustreza položaju te števke:

83 =

* **DECIMALNI ULOMKI**
* Enako načelo velja za decimalne ulomke. A se pri njih se uporabljajo negativne potence 10. Decimalni ulomek 0,256 pomeni tako 2 desetini plus 5 stotin plus 6 tisočin:

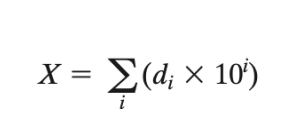
0.256 =

* Številka s celim in delnim delom ima števke pomnožene s pozitivnimi in negativnimi potencami števila 10:

442.256 =

* **POMEMBNOST ŠTEVK**
* Najpomembnejša števka
* Najbolj leva števka (nosi največjo vrednost)
* Najmanj pomembna številka
* Skrajna desna števka
* **POZICIJSKA RAZLAGA DECIMALNEGA ŠTEVILA**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **4** | **7** | **2** | **2** | **5** | **6** |
| 100s  102  position 2 | 10s  101  position 1 | 1s  100  position 0 | tenths  10-1  position -1 | hundredths  10-2  position -2 | thousandths  10-3  position -3 |

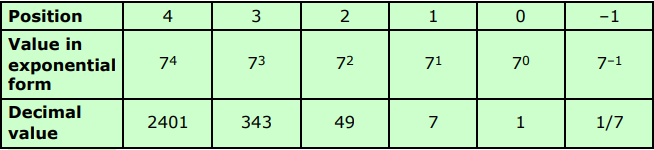


**POZICIJSKI ŠTEVILČNI SISTEMI**

* Vsako število je predstavljeno z nizom števk, v katerem ima števka na položaju i utež ri , kjer je r osnova številčnega sistema.
* Splošna oblika števila v takšnem sistemu z osnovo r je:

( . . . a3a2a1a0.a-1a-2a-3 . . . )r

* kjer je vrednost poljubne števke ai celo število v območju 0 < ai < r.
* Pika med a0 in a-1 se imenuje **TOČKA OSNOVE**.
* Pozicijska razlaga števila z osnovo 7:



**BINARNI SISTEM**

* Samo dve števki, 1 in 0
* Predstavljene v osnovi 2
* Števki 1 in 0 v binarnem zapisu imata enak pomen kot v decimalnih zapisih:

02 = 010

12 = 110

* Za predstavljanje večjih številk ima vsaka številka v binarnem številu vrednost, odvisno od svojega položaja:

102 = = 210

112 = =310

1002 = =410

* in tako naprej. Ponovno so frakcijske vrednosti predstavljene z negativnimi potencami osnove:

1001.101 = = 9.62510

* **PRETVARJANJE MED BINARNIM IN DECIMALNIM ŠTEVILOM**
* Binarni zapis v decimalni zapis:

• Vsako binarno števko pomnožite z ustrezno močjo 2 in prištevamo rezultate

* Decimalni zapis v binarni zapisu:

• Celi del in delež se obdelujeta ločeno

**CELA ŠTEVILA**

* V binarnem zapisu celo število predstavimo z zapisom:



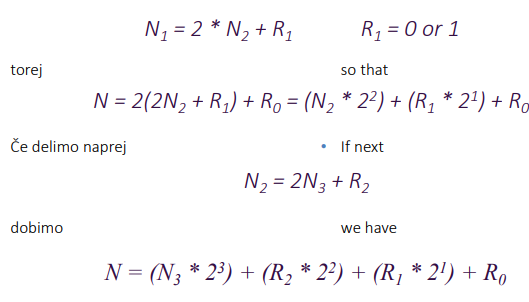
* in ima v desetiškem vrednost



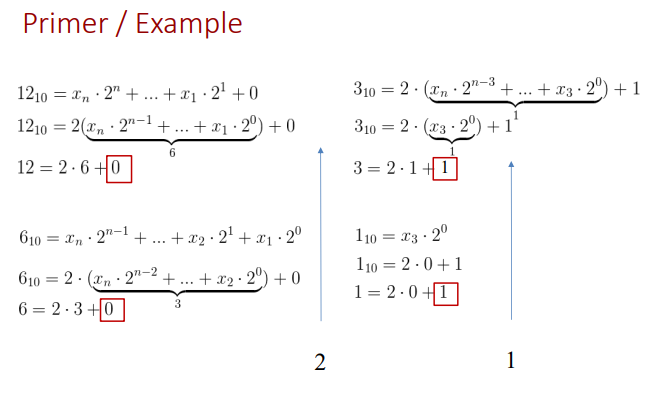
* Recimo, da je treba pretvoriti decimalno celo število N v binarno obliko. Če v decimalnem sistemu delimo N z 2, dobimo količnik N1 in preostanek R0, kar lahko zapišemo:



* Nato delimo dobljeni količnik N1 z 2. Predpostavimo, da je novi količnik N2, nov preostanek R1. Potem

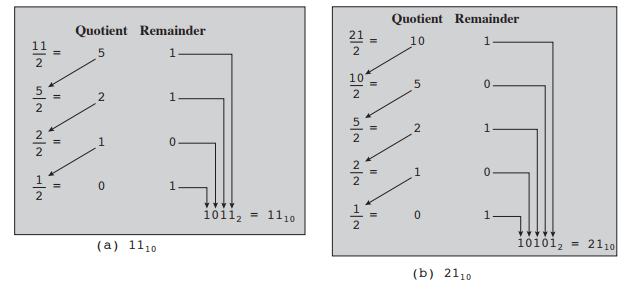


* Ker je N > N1 > N2. . . , bo nadaljevanje tega zaporedja sčasoma ustvarilo količnik Nm-1 = 1 (razen decimalnih celih števil 0 in 1, katerih binarni ekvivalenti sta 0 in 1) in preostanek Rm-2, ki je 0 ali 1. Potem je

binarna oblika N. Zato pretvorimo iz osnove 10 v osnovo 2 s ponavljanjem deljenja z 2. Ostanek in končni količnik 1 nam dajo (po naraščajočem pomenu) binarne števke N.****

**PRIMER**

* Primeri pretvorbe iz decimalnih zapisov v dvojiški zapis za celi števili:



**ULOMKI**

* Za delež se spomnite, da je v binarnem zapisu število z vrednostjo med 0 in 1 predstavljeno s:

****

* in ima vrednost:



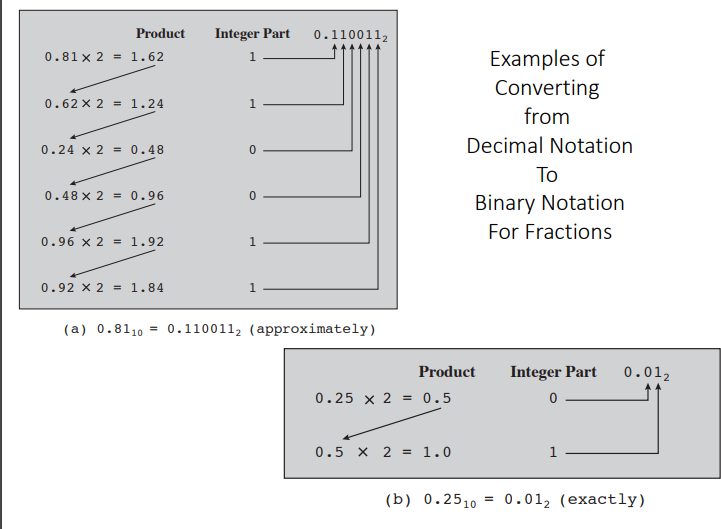
* To lahko zapišemo kot:



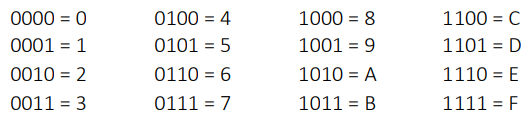
* Recimo, da želimo število F (0 < F < 1) pretvoriti iz decimalnega v binarni zapis. Vemo, da se lahko F izrazi v obliki:



* Če F pomnožimo z 2, dobimo: ****
* Iz te enačbe vidimo, da je celi del (2 \* F), ki mora biti bodisi 0 ali 1, ker je 0 < F < F1, preprosto b-1. Torej lahko rečemo (2 \* F) = b-1 + F1, kjer je 0 < F1 < 1 kjer : 
* Da bi našli b−2, ponovimo postopek. Na vsakem koraku se delež števila iz prejšnjega koraka pomnoži z 2. Števka na levi strani decimalne pike v rezultatu bo 0 ali 1 in prispeva k binarni predstavitvi, začenši z najpomembnejšo številko. Delež rezultata se v naslednjem koraku uporabi za množenje z 2.

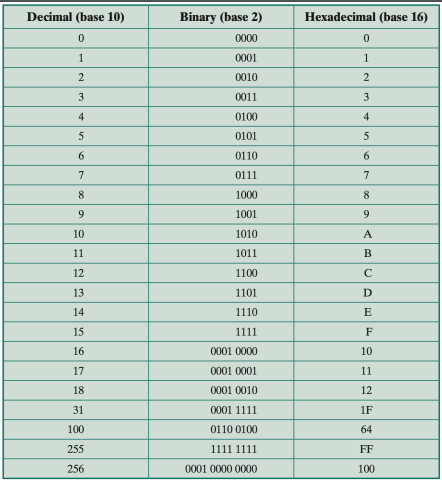


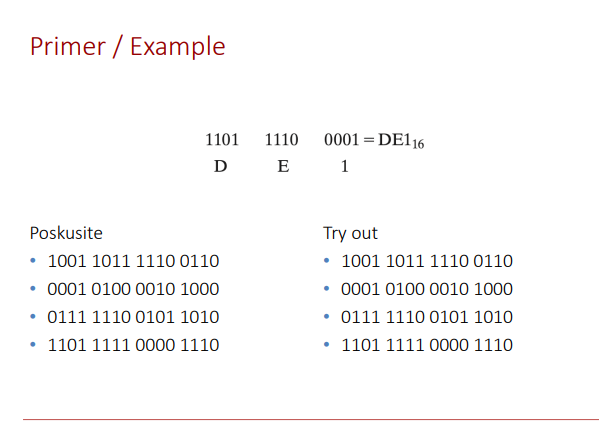
**ŠESTNJSTIŠKA PREDSTAVITEV**

* Binarne števke so združene v sklope štirih bitov, ki se imenujejo POLBAJTI ali POLZLOGI.
* Vsaka možna kombinacija štirih binarnih števk dobi simbol:
* Ker je uporabljenih 16 simbolov, se notacija imenuje šestnajstična, 16 simbolov pa šestnajstična. Tako:

****

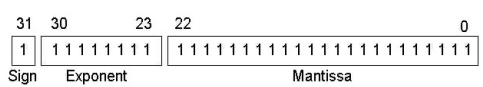
* Razlogi za uporabo šestnajstiškega zapisa so naslednji:
* Je bolj kompakten kot binarni zapis.
* Na večini računalnikov so binarni podatki predstavljeni z večkratnikom 4 bitov in s tem večkratnikom šestnajstiških števk.
* Izjemno enostavno je pretvoriti med binarnim in šestnajstnim zapisom.



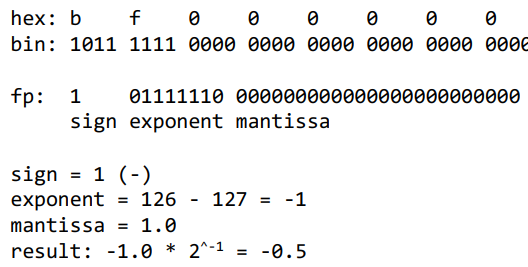


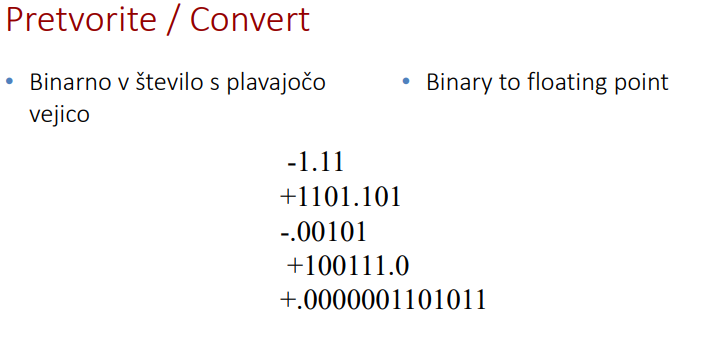
**PRIMER**

**PLAVAJOČA VEJICA**

* IEEE kratko realno: 32 bitov za znak, 8 bitov za eksponent in 23 bitov za mantiso. Imenuje se tudi enojna natančnost.
* IEEE dolgo realno: 64 bitov 1 bit za znak, 11 bitov za eksponent in 52 bitov za mantiso. Imenuje se tudi dvojna natančnost.
* Predznak binarne številke s plavajočo vejico predstavlja en bit. 1 bit pomeni negativno število, 0 bit pa pozitivno število.
* Eksponent lahko izračunamo iz bitov 24-31 tako, da odštejemo 127
* Mantisa (znana tudi kot signifikand ali ulomek) je shranjena v bitih 1-23. Nevidni vodilni bit (dejansko ni shranjen) z vrednostjo 1.0 je postavljen spredaj, nato ima bit 23 vrednost 1/2, bit 22 ima vrednost 1/4 itd
* Podkoračitev: Če eksponent doseže -127 (binarna 00000000 ali najmanjša vrednost (vse nič)), se upoštevajo posebna pravila za denormalizirane vrednosti. Vrednost eksponenta je nastavljena na 2-126 in "nevidni" vodilni bit za mantiso se ne uporablja več.

**Kako pretvorimo 0xbf000000 v decimalni zapis?**





**RAČUNALNIŠKA ARTIMETIKA**

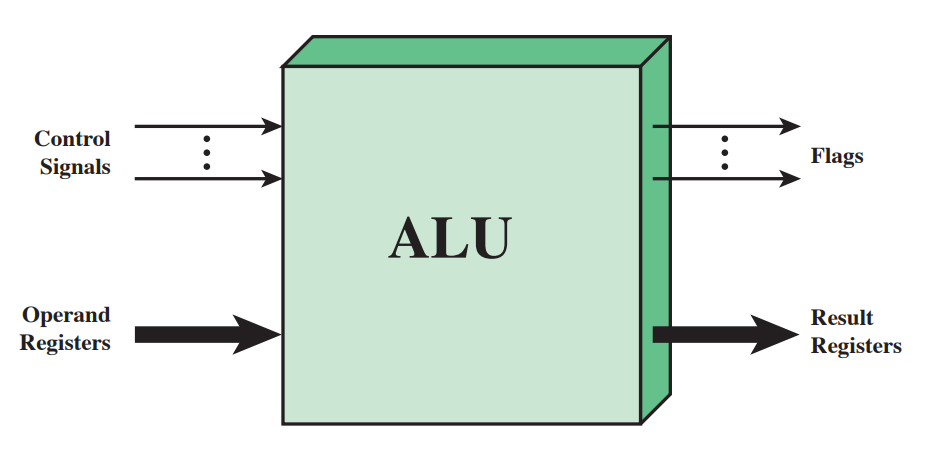
**ARTIMETIČNO LOGIČNA ENOTA (ALE)**

* Izvaja računske operacije
* Vse ostale enote v računalniku strežejo tej enoti
* Temelji na uporabi preprostih digitalnih logičnih naprav, ki lahko shranjujejo dvojiške števke in izvajajo preproste Booleanove logične operacije
* Operira nad celimi števili (angl. integer)
* Lahko dela tudi z realnimi (plavajoča vejica – angl. floating point) števili
* Lahko obstaja kot ločena enota za delo z realnimi števili (FPU) – npr. matematični koprocesor (486DX+).

**PREDSTAVITEV CELIH ŠTEVIL**

V sistemu binarnih števil lahko poljubna števila predstavljamo s:

* Števkami 0 in 1 npr. 4110 = 001010012
* Minusom za negativna števila
* Vejico za realna števila.
* -13.312510 = -1101.01012
* A hkrati na računalniku nimamo posebnega znaka za minus ali decimalno vejico
* Za prikaz števil lahko uporabljajo samo binarne števke (0,1)



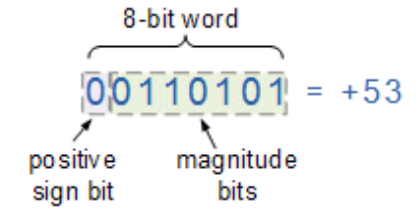
* **2 predstavitvi podrobno**

1. Predznak-velikost
2. Dvojiški complement

Ostale:

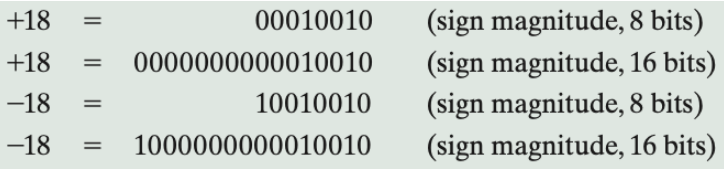
1. Z odmikom
2. Eniški complement
3. Baza -2

* **Predznak-velikost (P-V)**
* Najbolj levi bit predstavlja bit predznaka
* 0 pomeni pozitivno
* 1 pomeni negativno
* ¡ +18=00010010
* ¡ -18 = 10010010



**RAZŠIRITEV OBSEGA P-V**

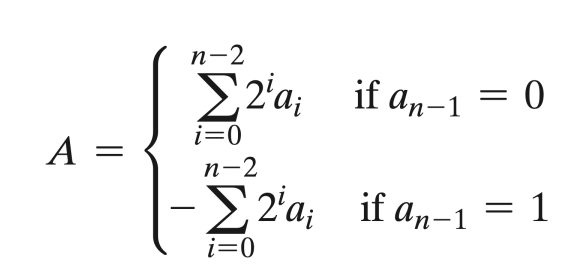
* Obseg števil, ki jih lahko izrazimo, se poveča s povečanjem dolžine bitov
* V zapisu P-V se to doseže s premikanjem predznaka na skrajno levo lego in z dopolnjevanjem ničel

****

**TEŽAVE P-V**

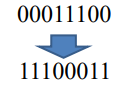
* Težave:
* Pri računanju je potrebna ločena obravnava predznaka in velikosti števila
* Dve predstavitvi ničle: +0 in -0
* Zaradi teh pomanjkljivosti se pri računanju celih števil ALU redko uporablja izvedba predznak +velikost.

**SPLOŠNA PREDSTAVITEV P-V**

****

**Eniški komplement / Dvojiški komplement**

* Negacijo števila v dvojiški notaciji dobimo tako, da invertamo vse števke
* 0 postanejo 1 in obratno
* Primer -28



* Težave:
* 2 ničli
* Seštevanje in odštevanje sta potrebna za dodajanje kakršnega koli "prenosa nazaj" (ali končnega prenosa) k rezultatu

**DVOJIŠKI KOMPLEMENT**

* Za zapis negativnega celega števila v dvojiškem komplementu, napišemo številko v dvojiški notaciji, invertamo (Eniški komplement) in prištejemo 1.
* Primer -28

00011100

* 0 postanejo 1 in obratno:

11100011

* prištejemo 1

11100100

**Obratni postopek**

* Obratni postopek

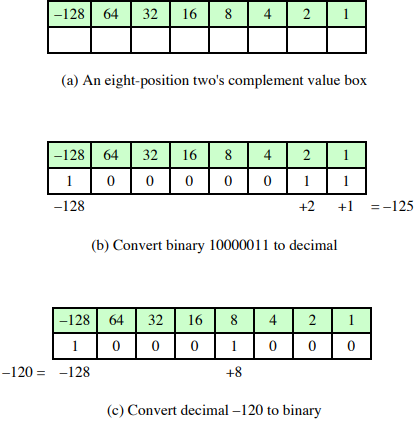
11100100

* 0 postanejo 1 in obratno (eniški komplement)

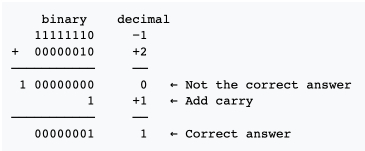
00011011

* prištejemo 1

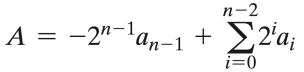
00011100



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_



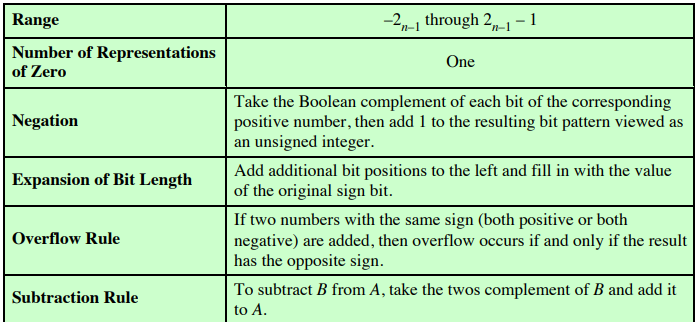
**Splošna predstavitev DK**



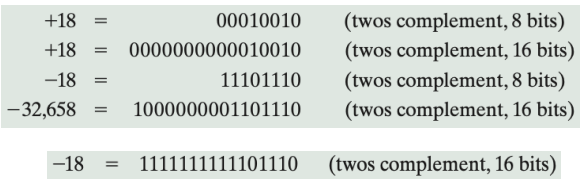
Dokaz: A + dvojiški komplement(A) = 0

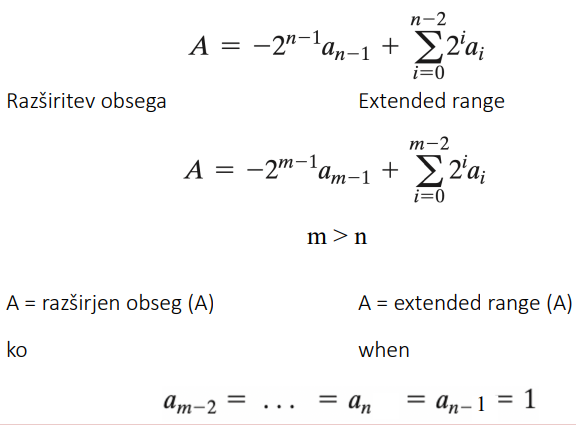


1. Značilnosti dvojiškega komplementa in aritmetike:



**Razširitev obsega DK**

* Ta postopek ne bo deloval pri dvojiškem komplementu za negativna cela števila ¡
* Pravilo je, da premaknemo bit predznaka na novo skrajno levo lego in ¡
* Za pozitivna števila vnesemo ničle, za negativna števila pa enice ¡ To imenujemo razširitev predznaka. 
* Dokaz:



**Predstavitev z odmikom**

* Predznačeno število n je predstavljeno z bitnim vzorcem, ki ustreza nepredznačenemu številu n+K, pri čemer K predstavlja odmik.

-310 (n) => -3+7 (n+K) = 4 => biased 0100

* Tako je 0 predstavljen z K, −K pa je predstavljen s samimi ničlami.
* Ni standarda za binarni odmik, vendar se najpogosteje za K pri n-bitnem številu uporablja K = 2n−1 ali K = 2n−1-1 (slednji uporabljan za eksponent v predstavitvi s plavajočo vejico).

**Baza -2**

* V običajnem dvojiškem številskem sistemu je osnova 2; tako skrajni desni bit predstavlja 20, naslednji bit predstavlja 21, naslednji bit 22 itd. Vendar pa je možen tudi dvojiški številski sistem z bazo −2. Skrajni desni bit predstavlja (−2)0 = +1, naslednji bit predstavlja (−2)1 = −2, naslednji bit (−2)2 = +4 in tako naprej, z izmeničnim predznakom.
* Obseg številk, ki jih je mogoče predstaviti, je asimetričen.
* Če ima številka sodo število bitov, je velikost največjega negativnega števila, ki ga je mogoče predstaviti, dvakrat večja od največjega pozitivnega števila, ki ga je mogoče predstaviti, in obratno, če ima beseda liho število bitov.