Ievads Datoru Arhitektūrā

Peldošā punkta skaitļu attēlošana

Tēmu saraksts

- Pāreja no decimālās sk. sistēmas uz bināro
- Pāreja no binārās sk. sist. uz decimālo
- Zinātniskais pieraksts
- IEEE 754 attēlojuma standarts
- Skaitļošana ar FP
- FP kļūdas

Pāreja no decimālās sk. sistēmas uz bināro

- Pieņemsim ka mums jāpārveido 252.390625 binārajā kodā.
  1. Pārveidojam 252 binārajā formā. 252 = 11111100
  2. Nākamā darbība ir pārveidot 0.390625 binārajā formā. Lai to veiktu mēs ***reizināsim*** ar 2. un rezultātā pierakstīsim to kas ir kreisajā pozīcijā.

0.390625 * 2 = 0.78125   0 <- svarīgākais bits (kreisā puse)
0.78125 * 2 = 1.5625   1 <- nākamā bita pozīcija
0.5625 * 2 = 1.125   1
0.125 * 2 = 0.25   0
0.25 * 2 = 0.5   0
0.5 * 2 = 1.0   1
0

- Tiekot līdz 0 esam visu izdarījuši:
  11111100.011001

Piemēri

Kā attēlojas 3.625 binārajā formā?

011,101

Kā attēlojas 0.1 binārajā formā?

0.1*2=0.2

0.2*2=0.4

0.4*2=0.8

0.8*2=1.6

0.6*2=1.2

0.2*2=

Tā varam turpināt līdz... kamēr sasniedzam vēlamo precizitāti.

Tāda pati doma ir visās parējās skaitīšanas sistēmās piem. 16. sistēmā tikai tad reizinām ar doto skaitīšanas bāzi.

Pāreja no binārās sk. sist. uz decimālo

- Tagad uz otru pusi. Ja mums ir dots binārais kods: 1100.011001, kā iegūt decimālās sistēmas skaitli?
- Ar 1100 problēmām būt nevajadzētu:
  1100 = 0 * 2^0 + 0 * 2^1 + 1*2^2 + 1*2^3 = 4+8 = 12
- Ar daļu .011001 savukārt vajag reizināt un summēt bet ***darīt to sākot ar pakāpi 2^-1***
  . 0 1 1 0 0 1 = 0*2^-1+1*2^-2+1*2^-3+0*2^-4+0*2^-5+1*2^-6
- Derētu atcerēties ka 2^-1 ir 1/2, 2^-2 ir 1/4, 2^-3 ir 1/8, utt.
- Sasummējot iegūstam:
  = 1 / 4 + 1 / 8 + 1 / 64
  = .25 + .125 + .015625
  = .390625
- Tātad rezultāts ir 12.390625.

Piemēri

Kāda ir 100.1111 decimālā vērtība?

100=4

.1111 = ½ +1/4 +1/8 +1/16 = 0.5 + 0.25 + 0.125 + 0.0625 = 0.9375

Kopā:....

Zinātniskais pieraksts

- Tagad Jūs ziniet kā pārveidot daļas uz bināro kodu bet parasti jau dati netiek uzglabāti vai nerodas šādā formā... Pirms veikt konvertāciju vajadzētu normalizēt skaitļus izmantojot zinātnisko pierakstu.
- Pārveidojot decimālo skaitli 0201.0900
  - Vispirms noskaidrojam kuri cipari ir zīmīgi. Formāli noteikumi lai noteiktu kurš cipars ir zīmīgs ir:
    1. Ne 0 cipars vienmēr ir zīmīgs
    2. 0 cipars nekad nav zīmīgs ja tas atrodas pirms zīmīga cipara
  - Pielietojot šos noteikumus varam atmest ievadošo 0 bet atstājam nobeiguma nulles jo mēs nezinām vai tās ir zīmīgas vai nē (pagaidām ir skaidrs tikai tas ka tās ir papildinājums)
- Lai pārveidotu šādu skaitli uz zinātnisko formu mums jāpārvieto decimālo punktu uzreiz aiz pirmā kreisās puses zīmīgā cipara reizinot rezultātu ar 10^n:

  2.010900 * 10^2
- Dotajā gadījumā pārvietojums ir divas decimālās pozīcijas un tāpēc reizinājam ar 10^2
- Ja mums būtu daļa tad mēs darītu tāpat tikai reizinātu ar kādu 10 daļu.

  0.0020109 = 2.0109 * 10-3

IEEE 754 attēlojuma standarts

- Lai attēlotu skaitli IEEE 754 formātā vispirms tas jāpārveido binārajā zinātniskajā formātā.
- Piemēram ir iegūts 1.001011 * 2^3.
- Vispārīgi šī vērtība satur: (Zīme) * (Mantisa) * 2(Eksponente)
- Saglabājot šo skaitli tiek pieņemts ka mēs lietosim 2. pakāpes un tad mums jāsaglabā:
  - Zīme
  - Eksponente (3)
  - Mantisa (1.001011)
- SP IEEE 754 formātā tam tiek atvēlēti 32 biti:

  S EEEEEEEE MMMMMMMMMMMMMMMMMMMMMMM

  1 8 23

  1 bits ir zīmes bits (kreisā pusē)

  8 biti ir atvēlēti eksponentes saglabāšanai.

  23 biti ir atvēlēti mantisas saglabāšanai.

IEEE 754 attēlojuma standarts

Zīmes lauks

- Vai nu 0 vai 1.
- O – pozitīvs skaitlis
- 1 – negatīvs skaitlis
- Mūsu piemērā 1.001011* 2^3 tas ir -> 0

IEEE 754 attēlojuma standarts

Eksponentes lauks

- Ja jau mums ir 8. biti tad attēlot varētu pakāpes no 0 – 255. Bet lai varētu attēlot negatīvas pakāpes ir izvēlēta nobīdītā (biased) eksponente kura tiek aprēķināta kā:

  NobīdītāEksponente = 127 + ReālāEksponente

- Tabulas veidā tātad::

Decimālā	NobīdītāDecimālā	NobīdātāBinārā
0	127 + 0 = 127	01111111
1	127 + 1 = 128	10000000
2	127 + 2 = 129	10000001
128	127 + 128 = 255	11111111*
-1	127 – 1 = 126	01111110
-127	127 – 127 = 0	00000000*

- Mūsu piemērā 1.001011* 2^3, būtu jāattēlo 3. Pielietojot nobīdi 127 iegūstam: 127+3 = 130 vai 10000010 binārajā kodā.

IEEE 754 attēlojuma standarts

Mantisas lauks (significand)

- Ņemot vērā to ka 2. pakāpes tiek izmantotas pēc noklusējuma viss kas atliek ir pierakstīt mūsu gadījumā 1.001011
- Ņemot vērā to ka mantisai tiek atvēlēti 23 biti varētu padomāt ka saglabāts tiks sekojoša bitu virkne 10010110000000000000000.
- Tomēr ņemot vērā to ka zinātniskajā pierakstā nevar skaitlis sākties ar nesvarīgu ciparu... Pieņem to ka pirmā pozīcija ir 1(*hidden bit)*.
- Šī ideja dod iespēju saglabāt vienu papildus bitu precizitātei.
- Mantisa 1.001011* 2^3 tad būtu 00101100000000000000000

Piemēri

- Saliekot visu kopā šim piemēram:
- Zīmes bits = 0
- Eksponente = 10000010
- Mantisa = 00101100000000000000000
- Vai: 0100 0001 0001 0110 0000 0000 0000 0000
- Vai: 41160000 sešpadsmitnieku sistēmā
- Attēlojiet –10.4375 IEEE 754 formātā.
- s1e10000010m01001110000000000000000
- Attēlojiet 0 IEEE 754 formātā

IEEE 754 attēlojuma standarts

- DP FP sastāv no 64 bitiem kas sadalās sekojoši 1, 11, un 52.
- Eksponentes lauks satur vērtību kas ir par 1023 lielāka par reālo vērtību.

Speciālie gadījumi

- FP attēlojumā ir daži skaitļi kas paredzēti speciālu gadījumu sagabāšnai piemēram : dalīšna ar 0, precīzi 0 attēlošana... Šie gadījumi tiek attēloti ar visiem 1 vai 0 eksponentes laukā.
- Pirmais gadījums ir nenormalizēti skaitļi (*denormalized* numbers). Šie gadījumi tiek attēloti ar eksponentes lauku kas satur visās pozīcijās 0
- Pēc iepriekš teiktā būtu 0-127=-127 bet tā nav jo mēs šoreiz attēlojam nevis 1.mantissa * 2^-127 bet gan 0.mantissa * 2^-126.
- Tas tiek darīts lai varētu iekodēt precīzi 0 (bez iedomātā 1. jo 0*2^-126==0) Tāpat šāds režīms ļauj iekodēt ļoti mazus skaitļus no 0 līdz 1*2^-126.
- Šo režīmu dēvē arī par pakāpenisku izzudi (*gradual underflow)*.
- Kā redzams ir iespējami gadījumi kad skaitļa vērtība vienalga būs pārāk maza lai to varētu iekodēt bet šādi vismaz var palielināt vērtību lauku.
- Otrs speciālo skaitļu gadījums ir tad kad eksponente satur visās pozīcijās 1. Atkal jau iepriekš tka teikts ka normāli tas būtu 255 – 127 jet eksponente 128. Tomēr šis ir speciāls gadījums kā attēlot bezgalību. Bezgalība tiek kodēta kās eksponente visi 1 un mantisa visas 0
- Var būt gan + gan – bezgalība . Bezgalība tiek iegūta pārpildīšanās gadījumā.
- Speciālais gadījums kad eksponentes laiukā ir visi 1 bet mantisa neastur tikai 0 tiek saukts par NaN (*Not a Number)*
- Šajos gadījumos zīme netiek ņemta vērā .
- NaN iegūst dalot 0 ar 0 vai citos nedefinētas aritmētikas gadījumos (dalot kaut ko ar 0 iegūstam bezgalību)

FP kļūdas

- Ja mums ir vienāds bitu skaits tad FP būs *neprecīzāks* attēlojums nekā veselo skaitļu aritmētikai ar tādu pat bitu skaitu.
- Piemēram 8 bitu gadījumā ar nenobīdītu eksponenti un bez spec. gadījumiem:

  EEE MMMMM

  3 5
- Gadījumā ja eksp. un mantisa satur tikai 0 tad tas ir 1.0*2^0=1
- Gadījumā ja eksp. un mantisa satur tikai 1 tad tas ir 1.11111*2^7=11111100 jeb 252
- Protams FP var attēlot daļas ko nevar veselie skaitļi...

FP kļūdas

- Vēl viena problēma. Kāds ir nākamais mazākais skaitlis pēc 252?
- Binārajā kodā tas būtu 111 11110 jeb 1.11110 * 2^7 jeb 11111000 jeb 248.
- Kļūdas FP gadījumā rodas no mantisas bitu nepietiekamības kas ir maksimālas lielu skaitļu gadījumā (tās tiek vēl pastiprinātas ar eksponenti).
- Mazām vērtībām tā nav problēma (piemēram var attēlot precīzi 2)

Skaitļošana ar FP

- Kā redzams dēļ noapaļošanas mantisas un eksponentes laukos ir iespējami gadījumi kad FP skaitļi nebūs identiski
- Tas ir līdzīgi kā gadījumā ar 0.2 attēlošanu binārā kodā
- Piemēram pieskaitot 0.2 kādam skaitlim 100,000 reizes rezultātā neiegūs pieaugumu par 20 000
- Šī iemesla dēļ lietojot FP aritmētiku nāksies pārbaudīt skaitļus kādās robežās (piem., >=19999.99 un <= 20000.01).

Skaitļošana ar FP

1. Pirmais darbs ir pārbausdīt vai kāds no operandiem nav 0. Ja ir tad otrs ir rezultāts.
2. Jāsaskaņo mantisas jo nevar saskaitīt ja eksponentes ir dažādas.
3. Saskaitīt mantisas.
4. Normalizēt rezultātu. Pārbaudot vai nav iegūts kāds izņēmums – izzude (underflow) vai pārpildīšanās (overflow).

- Reizinot ir vieglāk: saskaitīt eksponentes un sareizināt mantisas.
- Dalot: atņemt eksponentes un izdalīt mantisas.

Mājās

- Izlasīt:
  - http://en.wikipedia.org/wiki/Floating_point

- Kāds ir nākamais mazākais skaitlis (pēc 1) ko var attēlot 15. slaida piemērā?
- Pieņemiet ka 8. bitu binārā skaitlī ir iekodēta decimālā vērtība 3. Kā izmantojot bīdes darbības var iegūt decimālo vērtību 12?
- Pieņemiet ka 8. bitu binārā skaitlī ir iekodēta decimālā vērtība 48. Kā izmantojot bīdes darbības var iegūt decimālo vērtību 3?
- Parādiet kā teik reizinātas 9 un 5 vērtības 4. bitu datorā.