

# IEEE 754-2008 standard za aritmetiku realnih brojeva u pokretnom zarezu

Nenad Mitić, Matematički fakultet Univerziteta u Beogradu  
email: nenad@matf.bg.ac.rs

## 1 Realni brojevi u pokretnom zarezu

Realni brojevi se zapisuju u fiksnom zarezu kada pripadaju intervalu oblika  $[-d, d]$  gde je  $d$  neka vrednost. Ovaj način zapisa nije pogodan za jako velike ili jako male realne brojeve, kao ni za realne brojeve sa velikim brojem cifara u razlomljenom delu. U takvim slučajevima se koristi drugačija reprezentacija realnih brojeva.

Kroz istoriju razvoja računarstva je predloženo i korišćeno nekoliko različitih zapisa realnih brojeva, ali je danas u širokoj upotrebi samo **zapis realnih brojeva u pokretnom zarezu** (eng. *floating point*).<sup>1</sup> Realni brojevi u pokretnom zarezu se predstavljaju pomoću osnove  $\beta$  (koja je uvek parna) i preciznosti  $p$ . Na primer, ako je  $\beta=10$  i  $p=4$  tada se broj 0.4 predstavlja kao  $4.000 \times 10^{-1}$ , a ako je  $\beta=2$  i  $p=10$  tada se dekadni broj 0.4 može predstaviti samo u aproksimativnom obliku  $1,100110011 \times 2^{-2}$ . Predstavljanjem brojeva u pokretnom zarezu omogućuje se zapisivanje vrlo velikih ili jako malih brojeva pomoću malog broja cifara. Na primer, broj 564,000,000,000,000,000,000,000 se predstavlja kao  $5.640 \times 10^{26}$ , dok se broj 0.0000000000000000564 predstavlja kao  $5.640 \times 10^{-16}$ .

U opštem slučaju broj u pokretnom zarezu se predstavlja u obliku

$$\pm d_0, d_{-1}d_{-2}\dots d_{-(p-1)}\beta^e$$

gde se  $d_0, d_{-1}d_{-2}\dots d_{-(p-1)}$  naziva **značajni deo** (eng. *significand*),<sup>2</sup>  $\beta$  **osnova**,  $e$  **eksponent** i  $p$  **preciznost**. Niz cifara u razlomljenom delu broja se naziva **frakcija**. Značajni deo se zapisuje u brojanom sistemu sa osnovom  $\beta$ , tj.  $0 \leq d_i < \beta$ . Zapis broja za koji važi da je  $d_0 \neq 0$  se naziva **normalizovan**.

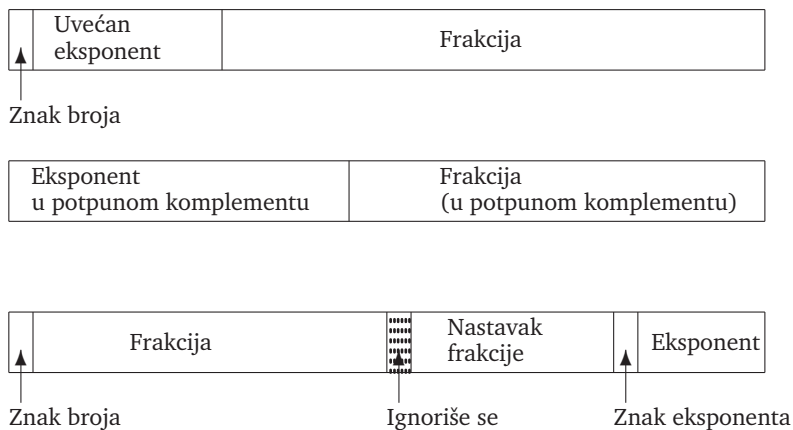
Za predstavljanje realnih brojeva u savremenim računarima se koriste vrednosti  $\beta=2$ ,  $\beta=10$  ili  $\beta=16$ . U sva tri slučaja eksponent se kodira u binarnom brojanom sistemu dok se vrednost za  $\beta$  ne čuva eksplicitno jer je ista za sve brojeve koji se zapisuju. Ako se za zapis brojeva koriste registri veličine 32 bita, tada se kaže da je broj zapisan sa jednostrukom preciznošću, ako se koriste dva registra (64 bita) tada je broj zapisan sa dvostrukom preciznošću, a ako se koriste četiri registra (128 bita) tada je broj zapisan sa četvorostrukom preciznošću. Rad sa preciznošću većom od dvostruke nisu u prethodnom periodu podržavali svi procesori. Ova mogućnost postaje šire dostupna tek sa pojavom 64-bitnih mikroprocesora, a sa usvajanjem standarda IEEE 754-2008 postaje obaveza proizvođača mikroprocesora. I sami termini *jednostruka*, *dvostruka*, *četvorostruka*... preciznost vode poreklo iz vremena kada je dužina registra procesora bila 32 bita. Danas, kada su 64-bitni procesori preuzeli primat na tržištu nazivi su ostali u upotrebi, mada ne daju ispravnu sliku o dužini polja koja se koristi za zapis podataka. U tom smislu, ispravnije je koristiti broj bajtova (4, 8, 16, ...) kao reper za veličinu prostora.

## Zapis brojeva u pokretnom zarezu

Kroz istoriju računarstva korišćeno je više oblika zapisa brojeva u pokretnom zarezu. U opštem slučaju, svi zapisi se mogu podeliti u tri grupe (slika 1). Kao najpogodniji za rad na procesorima opšte namene pokazao se prvi od navedenih formata. Detaljnije informacije o karakteristikama sva tri načina zapisa, kao i računarima koji su ih koristili ili ih koriste se može naći u [8].

<sup>1</sup>Primeri drugih načina zapisa su pokretna kosa crta (eng. *floating slash*) i označeni logaritam (eng. *signed logarithm*). Zbog teškoća pri implementaciji i sporosti izvođenja aritmetičkih operacija ovi zapisi su napušteni i koriste se samo na malom broju sistema ([3]).

<sup>2</sup>Ovaj termin, koji su uveli 1967. godine Forsythe i Moler je u potpunost zamenio stari termin *mantisa* (preciznije, ako je broj  $X$  bio predstavljen u pokretnom zarezu kao  $X = m\beta^e$ , tada se  $m$  nazivalo *mantisa*, a  $e$  eksponent u zapisu broja  $X$ ).



Slika 1: Formati zapisa realnih brojeva u pokretnom zarezu na različitim računarima kroz istoriju

## 2 IEEE standard 754-2008

Kao što je rečeno, realni brojevi su u računarima zapisivani na više različitih načina. Uz to, proizvođači računara su primenjivali različite algoritme pri zaokruživanju vrednosti, izračunavanju elementarnih funkcija i izvodenju osnovnih aritmetičkih operacija. Zbog toga je prenosivost numerički intenzivnih programa između različitih računarskih sistema bila slaba, često uz razliku u rezultatima izvršavanja takvih programa. To su bili glavni razlozi usvajanja standarda za predstavljanje i zapis realnih brojeva u računarima. Prva verzija standarda je usvojena 1985. godine. Standard je dobio naziv IEEE 754, prema nazivu organizacije koja se najviše angažovala na njegovom definisanju i usvajanju.<sup>3</sup> Standard je kroz vreme doživeo više revizija. Sa usvajanjem nove verzije standarda, kao sufiks je dodata godina usvajanja, tako da se prva doneta verzija označava kao IEEE 754-1985, a novousvojena verzija kao IEEE 754-2008. Oba IEEE 754 standarda propisuju algoritme za operacije sabiranja, oduzimanja, množenja, deljenja i izračunavanja kvadratnog korena, kao i način njihove implementacije (npr. načine zaokruživanja, ponašanje u graničnim slučajevima, ...). Zbog toga, rezultati osnovnih operacija su isti na svim računarima koji podržavaju ovaj standard, čime se pojednostavljuje prenošenje programa. Detaljna diskusija o problemima koji se javljaju pri izračunavanjima sa realnim brojevima je data u [3], [9]; u daljem tekstu su navedeni neki od njih.

1. **Zaokruživanje i greška zaokruživanja.** Beskonačno mnogo realnih brojeva može da se zapiše konačnim brojem kombinacija bitova u registru jedino pomoću njihovih približnih reprezentacija zapisanih u unapred fiksiranom broju bitova. Sa druge strane, bez obzira na broj bitova koji je određen za zapis realnih brojeva, rezultati aritmetičkih operacija se ne mogu uvek tačno zapisati u tako izabranom broju bitova. Zbog toga je potrebno njihovo zaokruživanje do dužine koja je pogodna za zapis. Greška koja se pri tome javlja se meri na dva načina: pomoću *ulp*-a i pomoću *relativne greške*.

Neka za format u pokretnom zarezu važi  $\beta=10$  i  $p=3$ . Neka je rezultat izračunavanja  $5,76 \times 10^{-2}$ , a matematički tačna vrednost računata sa beskonačnom preciznošću 0,0574. Očigledno je greška veličine dvostruke jedinične vrednosti na poslednjem mestu zapisa broja. Slično, ako se realan broj 0,0574367 zapiše kao  $5,74 \times 10^{-2}$  greška je 0,367 jedinica na poslednjem mestu. Veličina "jedinica na poslednjem mestu" se označava sa *ulp* prema akronimu engleskog naziva *unit in the last place*. U opštem slučaju, ako realni broj  $d_0, d_{-1} \dots d_{-(p-1)} \times \beta^e$  predstavlja broj  $z$ , tada je greška u zapisu  $|d_0, d_{-1} \dots d_{-(p-1)} - (z/\beta^e)| \times \beta^{p-1}$  jedinica na poslednjem mestu.

Drugi način merjenja razlike između realnog broja i njegove reprezentacije je *relativna greška*. Relativna greška je apsolutna vrednost razlike između realnog broja i njegove reprezentacije podeljena sa apsolutnom vrednošću realnog broja. Na primer, relativna greška pri aproksimaciji 5,74367 sa  $5,74 \times 10^0$  je  $0,00367/5,74367 \approx 0,0006$ .

I *ulp* i relativna greška zavise od tzv. mašinske tačnosti (*mašinskog  $\epsilon$* ):  $\epsilon = (\beta/2)\beta^{-p}$ . Relativna greška se uvek zapisuje kao faktor od  $\epsilon$ . Na primer, u prethodnom slučaju je  $\epsilon = (\beta/2) \cdot \beta^{-p} = 5 \cdot (10)^{-3} = 0,005$ . Tako

<sup>3</sup>IEEE je skraćena od *Institute of Electrical and Electronics Engineers*.

se relativna greška može izraziti kao

$$((0,00367/5,74367)/0,005)\epsilon \approx 0,12\epsilon$$

Ako je rezultat izračunavanja vrednosti realnog broja zaokružen najbliže tačnom rezultatu, još uvek može da postoji greška od najviše 0,5 *ulp*. Relativna greška koja postoji u tom slučaju je uvek ograničena sa  $\epsilon$  ([3]).

Sledeći primer može da posluži za ilustraciju razlike između *ulp*-a i relativne greške. Neka je  $x = 12,35$  i neka je njegova aproksimacija  $x_a = 1,24 \times 10^1$ . Greška je 0,5 $ulp$ -a, a relativna greška je 0,8 $\epsilon$ . Ako se posmatra proizvod  $8 * x_a$ , tačna vrednost je  $8 * x = 98,8$ , dok je izračunata vrednost  $8 * x_a = 9,92 \times 10^1$ . Greška merena u *ulp*-ima je 8 puta veća, dok je relativna greška ista.

*Ulp* i relativna greška se koriste u različite svrhe: *ulp* za određivanje greške zaokruživanja brojeva, a relativna greška za analiziranje grešaka izračunavanja prema različitim formulama. Ako je potreban samo red veličine greške tada se mogu koristiti i *ulp* i relativna greška jer se razlikuju za najviše faktor  $\beta$ . Takođe, ako je realan broj zapisan sa greškom od  $n$  *ulp*-a, tada je broj cifara obuhvaćenih tom greškom  $\log_{\beta} n$ . Ako je pak relativna greška pri izračunavanju  $n\epsilon$ , tada je broj zahvaćenih cifara  $\approx \log_{\beta} n$ .

Prema pravilima, zaokruživanje se vrši na najbližu vrednost. Tako se npr. 1,26 zaokružuje na 1,3 dok se 1,24 zaokružuje na 1,2. U prošlosti zaokruživanje broja  $d_0d_{-1}d_{-2}\dots d_{-k}$  na  $k$ -tu cifru nije bilo jedinstveno rešeno na svim računarskim sistemima za  $d_{-k} = 5$ . Tako su DEC VAX računari u tom slučaju vršili zaokruživanje naviše, odnosno vrednost 1,25 zaokruživali na 1,3. Međutim, pokazuje se ([7]) da najbolje rezultate daje zaokruživanje na parnu cifru. Tako bi se po tom pravilu 1,25 zaokružilo na 1,2 (jer je 2 parno).

2. **Cifre čuvari.** Jedan od metoda za računanje razlike između dva realna broja je izračunavanje tačne vrednosti razlike i zatim njeno zaokruživanje na najbliži broj u pokretnom zarezu. Ova operacija može da bude vrlo skupa (u smislu neophodnih računarskih resursa) kod brojeva koji se razlikuju za nekoliko redova veličine jer ih je potrebno dovesti na isti eksponent. Zbog toga i zbog ograničenja broja bitova koji se koriste za zapis, izračunavanje tačnih vrednosti razlike nije moguće, tj. dobijeni rezultati mogu da budu nekorektni. Neka je npr.  $p = 3$ ,  $\beta = 10$  i neka treba naći razliku 10,1-9,93:

$$\begin{array}{r} x = 1,01 \times 10^1 \\ y = 0,99 \times 10^1 \\ \hline x-y = 0,02 \times 10^1 \end{array}$$

Tačan odgovor je 0,17 tako da izračunata vrednost ima grešku od oko 30 *ulp*-a. Međutim, ako se operacije izvode sa dodatnim ciframa koje su uključene da bi nadgledale situaciju (*cifre čuvari*), tada se manji broj zapisuje sa  $p + 1$ -om cifrom i tek posle izračunavanja se rezultat zaokružuje na  $p$  cifara:

$$\begin{array}{r} x = 1,010 \times 10^1 \\ y = 0,993 \times 10^1 \\ \hline x-y = 0,017 \times 10^1 \end{array}$$

U ovom slučaju se dobija korektan rezultat. Cifre čuvari su se pokazale toliko značajne u izračunavanju da ih je IBM 1968. godine uključio pri izračunavanju dvostruke tačnosti računara serije S/360 (već su prethodno bili uključeni pri izračunavanju jednostruke tačnosti) i modifikovao sve prethodno objavljene modele. Nisu svi računari imali uključene cifre čuvaru pri izvodjenju aritmetičkih operacija. Najpoznatiji primer je računar CRAY I. Zbog toga na njemu mogu da se u graničnim slučajevima (npr. kada se operacije vrše sa vrednostima koje su blizu najveće vrednosti koja može da se zapiše) dobiju neočekivani (i pogrešni) rezultati. Pokazuje se da je u nekim slučajevima za dobijanje korektnog rezultata potrebno čuvati više od jednog cifarskog mesta ([3]). Zbog toga se u savremenim mikroprocesorima koriste cifre čuvari na dva cifarska mesta.

3. **Tačno zaokružene operacije.** Kada se operacije brojeva u pokretnom zarezu izvode korišćenjem cifara čuvara dobijeni rezultati nisu tako precizni kao kada se izračunaju tačne vrednosti i zatim rezultat zaokruži na najbliži broj u pokretnom zarezu. Operacije koje se izvode na ovakav način se nazivaju *tačno zaokružene*.

## Opis standarda

IEEE standard 754-1985 je razvijen sa ciljem da omogući prenosivost programa kao i sa ciljem da se ohrabri razvoj sofisticiranih numerički orijentisanih programa i izračunavanje u računarima što više približi izračunavanju u realnoj matematici. O reviziji sa ciljem poboljšanja i proširenja je počelo da se razmišlja 2000. godine. Postavljeni ciljevi revizije su bili

- Da se otklone neki uočeni nedostaci poznati samo nekolicini eksperata
- Uvodjenje veće preciznosti zapisa (uključujući i četvorostruku preciznost)
- Uključivanje specifikacija navedenih u IEEE 854 standardu (tj. zapisa realnih brojeva pomoću dekadne osnove)
- Uključivanje formata podataka opisanih u standardu IEEE 1596.5 iz 1993. godine kome je isticao rok važenja. Ovaj standard je opisivao više različitih formata podataka (uključujući i IEEE 754 formate) za različite računare koji više nisu postojali (stari modeli Cray računara, VAX-11/780, CDC-6000, itd.)
- Dodavanje transcendentnih funkcija
- Dodavanje operacije spojenog višestrukog sabiranja
- Uvodjenje malih promena u specifikaciju operacija za konverziju između celobrojnih vrednosti i realnih brojeva u pokretnom zarezu
- Da ciljni rezultat ne označi nekorektnim ni jedan od načina izračunavanja koji se koristio na tada postojećim računarima.

Poslednji od navedenih uslova je bio jako značajan. Situacija je bila znatno drugačija nego 1985. godine kada nije postojao nikakav standard za operacije sa realnim brojevima. Formati zapisa brojeva predviđeni standardom iz 1985. godine su u trenutku otpočinjanja revizije korišćeni skoro na svim procesorima i koji nisu mogli da budu promenjeni da bi zadovoljili revidirani standard. Zbog toga je komitet za standarde usvojio pravilo da nove mogućnosti mogu da budu dodavane, ali bez narušavanja postojećih. Posle nekoliko godina razmatranja revizije, nova verzija standarda je usvojena 2008. godine. Formati zapisa brojeva pomoću dekadne osnove predviđeni verzijom iz 2008. godine su trenutno potpuno hardverski implementirani na relativno malom broju mikroprocesora (IBM POWER 6 i z serija). U daljem tekstu će za označavanje verzije standarda iz 2008. godine neće biti korišćen sufiks 2008, tj. biće označavan samo kao IEEE 754.

IEEE 754 standard propisuje:

- $\beta=2$  i  $\beta=10$  kao osnove koje se koristi za zapis brojeva u pokretnom zarezu.
- Osnovne formate za zapis podataka u binarnoj i dekadnoj osnovi
- Formate za razmenu podataka zapisanih u binarnoj i dekadnoj osnovi
- Proširene i proširive formate za zapis podataka
- Način izvođenja operacija sabiranja, oduzimanja, množenja, deljenja, spojenog višestrukog sabiranja,<sup>4</sup> dobijanja kvadratnog korena, ostatka pri deljenju, izvođenja poredjenja, itd.
- Način konverzije između celobrojnih i vrednosti u pokretnom zarezu
- Način konverzije između različitih formata brojeva u pokretnom zarezu
- Način konverzije između formata brojeva u pokretnom zarezu i njihove spoljašnje reprezentacije u obliku niski karaktera
- Vrste izuzetaka koji se javljaju pri radu sa brojevima u pokretnom zarezu i načine njihove obrade

U daljem tekstu će detaljnije biti prikazan samo deo standarda koji se odnosi na načine zapisa i formate brojeva, i neke od operacija sa njima. Detaljnije informacije o standardu se mogu naći u [5] (standard iz 2005. godine) i [6] (standard iz 2008. godine).

Formalno, broj u pokretnom zarezu (bez obzira na osnovu koja se koristi za zapis) može da bude predstavljen na jedan od sledeća tri načina:

<sup>4</sup>Operacija spojenog višestrukog sabiranja je pri izvođenju u računarima značajno brža od operacije višestrukog izvođenja operacije sabiranja. Zaokruživanje kod ove operacije se vrši samo jednom, na kraju sabiranja, a ne pri svakom sabiranju dve vrednosti.

1. Kao uredjena trojka (*znak*, *eksponent*, *značajan\_deo\_broja*). U osnovi  $\beta$  broj u pokretnom zarezu predstavljen na ovaj način ima vrednost  
 $(-1)^{\text{znak}} \times \beta^{\text{eksponent}} \times \text{značajan\_deo\_broja}$ .
2.  $+\infty$ ,  $-\infty$
3. qNaN (tihi NaN) i sNaN (signalni NaN)

Enkodiranje preslikava ove reprezentacije u niske bitova. Pri tome, enkodiranje može da bude takvo da jednu vrednost u pokretnom zarezu preslikava u više različitih niski bitova.

Konačan skup brojeva u pokretnom zarezu koji koji su predstavljivi pojedinačnim formatom zapisa je određen sledećim celobrojnim parametrima:

- Osnovom  $\beta=2$  ili  $\beta=10$
- Brojem cifara u značajnom delu  $p$ . Broj cifara u značajnom delu broja određuje *preciznost (tačnost) zapisa*.
- Najvećom vrednošću eksponenta *emax*.
- Najmanjom vrednošću eksponenta *emin*. Pri tome za sve formate mora da važi  $emin=1-emax$ .

## Specijalne vrednosti

Prema standardu IEEE 754 određene bit kombinacije u zapisu realnih brojeva označavaju specijalne vrednosti. Specijalne vrednosti se koriste radi obezbeđivanja korektnosti IEEE aritmetike i efikasnije programske obrade izuzeća i specijalnih stanja. Podaci u IEEE 754 zapisu se dele u klase koje uključuju numeričke i odgovarajuće nenumeričke entitete. Klase podataka propisane IEEE standardom su:

- **Normalni brojevi.** Realni brojevi u intervalu  $[\beta^{emin}, \beta^{emax} \times (\beta - \beta^{1-p})]$  se nazivaju *normalni*.
- **Subnormalni brojevi.** Za normalne brojeve u pokretnom zarezu koji se nalaze blizu  $\beta^{emin}$  (npr. kod zapisa sa binarnom osnovom za  $x = 1,0002 \times 2^{-125}$  i  $y = 1,0001 \times 2^{-125}$ ) ne važi

$$x = y \Leftrightarrow x - y = 0 \quad (1)$$

što može da dovede do grešaka u programu. Na primer, naredba

```
if (x <> y) then z = 1/(x-y)
```

može da izazove prekid programa zbog deljenja sa nulom. Da bi se ovakvi i slični problemi eliminisali, IEEE standard uvodi *subnormalne brojeve*. Brojevi koji su po apsolutnoj vrednosti manji od  $\beta^{emin}$  (ali veći od 0) se nazivaju *subnormalni*.<sup>5</sup> Subnormalni brojevi uvek imaju manje od  $p$  značajnih cifara. Svaki konačan broj u pokretnom zarezu predstavlja umnožak subnormalnog broja sa najmanjom apsolutnom vrednošću  $\beta^{emin} \times \beta^{1-p}$ . Bez subnormalnih brojeva, svaki broj manji od  $\beta^{emin}$  je zamenjivan nulom. Sa subnormalnim brojevima granica je pomerena bliže ka 0 čime se postiže tzv. *postepeno potkoračenje* i obezbeđuje da relacija 1 uvek važi.

- **Beskonačno.** Kao što NaN-ovi omogućuju nastavak izračunavanja u slučaju nailaska na  $\sqrt{-1}$ , tako i beskonačno predstavlja način za nastavak izvršavanja programa u slučaju pojave prekoračenja. U zavisnosti od znaka dobija se vrednost  $+\infty$  ili  $-\infty$ .
- **Označena nula.** Kako znak može da ima dve različite vrednosti, to u standardu postoje dva zapisa za nulu:  $+0$  i  $-0$ . Da ne bi bilo razlike pri poredjenju  $+0$  i  $-0$  (jer bi npr. naredbe programa tipa `if (x=0) . . . . vraćale nepredvidive rezultate`), standardom je definisano da važi  $+0 = -0$ .

Iako postojanje dve nule ima nedostataka<sup>6</sup> IEEE komitet je procenio da su koristi od postojanja označene nule veće od njenih nedostataka. Tako, kada množenje ili deljenje sadrži označenu nulu, njen znak utiče na znak krajnjeg rezultata:  $+45*(+0) = +0$ ,  $+0/(-45) = -0$ . Takodje, u slučaju neoznačene nule ne bi važila relacija  $1/(1/x)=x$  za  $x=\pm\infty$ . Razlog je što i  $1/-\infty$  i  $1/+\infty$  kao rezultat daju 0, a  $1/0=+\infty$ , pri čemu se gubi informacija o znaku.

<sup>5</sup>U standardu IEEE 754-1985 normalni brojevi su nazivani *normalizovani*, a subnormalni *denormalizovani*.

<sup>6</sup>Npr. kada je  $x = +0$ ,  $y = -0$  ne važi ekvivalencija  $x = y \Leftrightarrow 1/x = 1/y$ .

Drugi primer upotrebe označene nule tiče se potkoračenja i funkcija koje imaju prekid u 0, kao što je npr. logaritam. U IEEE aritmetici prirodno je da se definiše  $\log 0 = -\infty$  i  $\log x = \text{NaN}$  kada je  $x < 0$ . Ako  $x$  predstavlja mali negativan broj koji teži ka 0, zahvaljujući označenoj nuli  $x$  će biti negativan i  $\log$  će vratiti vrednost NaN; bez označene nule ne bi mogla da se napravi razlika i dobila bi se vrednost  $-\infty$ . Dodatni primeri upotrebe označene nule su dati u [3].

- **NaN.** Tradicionalno, izračunavanje  $0/0$  ili  $\sqrt{-1}$  se smatralo nepopravivom greškom koja je dovela do zaustavljanja izračunavanja. Međutim, u nekim slučajevima ima smisla nastaviti sa izračunavanjem u takvoj situaciji. Na primer, neka potprogram nula određuje nulu funkcije  $f$ . Uobičajeno je da korisnik unese interval  $[a, b]$  na kome je funkcija definisana i na kome se traži nula. Većina takvih potprograma radi tako što menja vrednost argumenta i određuje odgovarajuće vrednosti funkcije  $f$ . Ako je vrednost argumenta van domena  $f$ , tada izračunata vrednost može biti  $0/0$  ili  $\sqrt{-1}$  što prouzrokuje nepotrebno zaustavljanje izvršavanja potprograma.

Ovakvi problemi se izbegavaju uvođenjem specijalnih vrednosti koje nisu brojevi. Te vrednosti su nazvane NaN, (skraćena od eng. *not-a-number*, nije broj). U standardu su definisane dve grupe NaN-ova:

1. Signalni NaN (eng. *Signaling NaN, SNaN*) signalizira izuzeto stanje kod aritmetičkih operacija, poređenja i konverzije kod operacija koje su deo standarda.
2. Tihi NaN (eng. *Quiet NaN, QNaN*) predstavlja pojavu nedozvoljene operacije u programu. QNaN se propagira kroz izračunavanje tako da ostaje vidljiv na njegovom kraju. Pri tome se ne signalizira pojava izuzeća.

Specijalne vrednosti se zapisuju kao različite kombinacije vrednosti bitova u zapisu broja. Način zapisa specijalnih vrednosti zavisi od osnove koja se koristi za zapis realnog broja u pokretnom zarezu.

## Osnovni formati zapisa

IEEE 754 propisuje pet osnovnih formata zapisa:

1. Tri formata zapisa sa binarnom osnovom sa preslikavanjem vrednosti u 32, 64 i 128 bita.
2. Dva formata zapisa sa dekadnom osnovom sa preslikavanjem vrednosti u 64 i 128 bita.

Podaci zapisani u osnovnim formatima se koriste u aritmetičkim operacijama. Da bi implementacija bila u saglasnosti sa standardom, mora da bude implementiran bar jedan od ovih formata. U narednoj tabeli su prikazani preciznost i najveća vrednost eksponenta koji definišu osnovne formate u zapis brojeva u pokretnom zarezu.

Parametri	Formati				
	Sa binarnom osnovom ( $\beta=2$ )			Sa dekadnom osnovom ( $\beta=10$ )	
	binary32	binary64	binary128	decimal64	decimal128
p	24	53	113	16	34
emax	+127	+1023	+16383	+384	+6144

Tabela 1: Osnovni formati zapisa brojeva u IEEE 754-2008

Bez obzira na format zapisa, implementacija mora da omogućiti predstavljanje sledećih brojeva u pokretnom zarezu:

- Označenih nula i ne-nula brojeva u pokretnom zarezu u obliku  $(-1)^s \times \beta^e \times m$  gde važi:
  1.  $s$  je 0 ili 1
  2.  $e$  je proizvoljan ceo broj  $e_{min} \leq e \leq e_{max}$
  3.  $m$  je broj predstavljen niskom cifara oblika  $d_0.d_1d_2\dots d_{p-1}$  gde je  $d_i$  cifra  $0 \leq d_i < \beta$ . Odavde je  $0 \leq m < \beta$ .
- Dve beskonačne vrednosti  $-\infty$  i  $+\infty$ .
- Dve NaN vrednosti (qNaN i sNaN)

U nekim slučajevima je pogodno da se značajan deo broja predstavi u obliku celog umesto razlomljenog broja. U tom slučaju se konačan broj u pokretnom zarezu predstavlja kao označen nula ili ne-nula broj oblika  $(-1)^s \times \beta^q \times c$  gde važi:

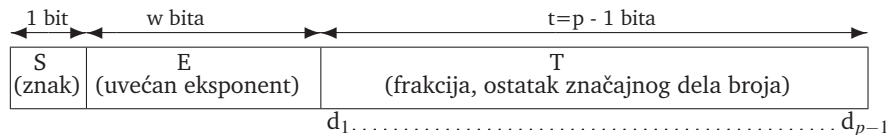
1.  $s$  je 0 ili 1
2.  $q$  je proizvoljan ceo broj  $emin \leq q + p - 1 \leq emax$
3.  $c$  je broj predstavljen niskom cifara oblika  $d_0d_1d_2\dots d_{p-1}$  gde je  $d_i$  cifra  $0 \leq d_i < \beta$ . Odavde je  $0 \leq c < \beta^p$ .

Reprezentacije i vrednosti brojeva u osnovnim formatima zapisa su iste kao i u odgovarajućim formatima za razmenu podataka.

## Formati za razmenu podataka zapisanih pomoću binarne osnove

Formati za razmenu se koriste pri razmeni podataka izmedju implementacija. U formatu za razmenu podataka zapisanih pomoću binarne osnove se svaki broj u pokretnom zarezu zapisuje na tačno jedan način. Da bi enkodiranje bilo jedinstveno u skladu sa prethodnim pravilima, vrednost značajnog dela broja  $m$  se povećava uz smanjenje vrednosti eksponenta  $e$  sve dok se ne dodje do  $e = emin$  ili do  $m \geq 1$ . Na kraju ovog procesa, ako je  $e = emin$  i  $0 < m < 1$  broj u pokretnom zarezu je subnormalan.

Realan broj u pokretnom zarezu se u formatu za razmenu podataka pomoću binarne osnove zapisuje pomoću  $k$  bitova (slika 2) podeljenih u tri polja sa sledećim značenjem:



Slika 2: Format za razmenu podataka zapisanih pomoću binarne osnove

1. Znak broja  $S$  veličine 1 bit
2. Uvećani eksponent  $E = e + uvećanje$  zapisan u  $w$  bita
3. Ostatak  $T = d_1d_2\dots d_{p-1}$  značajnog dela broja zapisan u dužini  $t = p - 1$  bita. Vodeći bit  $d_0$  značajnog dela broja je implicitno zapisan u uvećanom eksponentu  $E$ .

Standard definiše formate za razmenu podataka zapisanih pomoću binarne osnove veličine 16, 32, 64 i 128 bita. Formati binary32 i binary64 odgovaraju zapisu u jednostrukoj i dvostrukoj tačnosti prema standardu IEEE 754-1985. U opštem slučaju, veličina formata može da bude bilo koji broj veći od 128 koji je umnožak od 32. U tabeli 3 su prikazane vrednosti parametara za formate sa binarnom osnovom predviđene standardom. Klase podataka, njihova reprezentacija i vrednost predstavljene pomoću binarne osnove su prikazane u tabeli 4.

U tabeli 2 su dati primeri zapisa realnih brojeva pomoću binarne osnove u jednostrukoj tačnosti prema IEEE 754 standardu.

U slučaju da je dužina formata 64 bita ili je umnožak od 32 i  $\geq 128$  bita, vrednosti parametara se mogu odrediti na sledeći način:

$$\begin{aligned}
 k &= 1 + w + t = w + p \\
 &= 32 \times \text{ceiling}((p + \text{round}(4 \times \log_2(p + \text{round}(4 \times \log_2(p)) - 13)) - 13) / 32) \\
 w &= k - t - 1 = k - p = \text{round}(4 \times \log_2(k)) - 13 \\
 t &= kw - 1 = p - 1 = k - \text{round}(4 \times \log_2(k)) + 12 \\
 p &= k - w = t + 1 = k - \text{round}(4 \times \log_2(k)) + 13 \\
 emax &= bias = 2^{(w-1)} - 1 \\
 emin &= 1 - emax = 2 - 2^{(w-1)}
 \end{aligned}$$

pri čemu funkcija  $\text{round}()$  vrši zaokruživanje na najbliži ceo broj.

	Znak	EkspONENT	Frakcija
+15 =	0	10000010	111000000000000000000000
-15 =	1	10000010	111000000000000000000000
+1/64 =	0	01111001	000000000000000000000000
+0 =	0	00000000	000000000000000000000000
-0 =	1	00000000	000000000000000000000000
$(1 - 2^{-24}) \times 2^{+128}$ =	0	11111110	111111111111111111111111
$+1 \times 2^{-126}$ =	0	00000001	000000000000000000000000
$+1 \times 2^{-149}$ =	0	00000000	000000000000000000000001

Provera vrednosti zapisa broja +15:

$$\begin{aligned}
 & - \text{Znak} = + \\
 & - \text{EkspONENT} = 3 (=130-127) \\
 & - 1, \text{frakcija} = (1,111)_2 = 1 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-3} \\
 & - \text{Vrednost} = \text{Znak} 1, \text{frakcija} * 2^{\text{ekspONENT}} = +(+1 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-3}) * 2^3 \\
 & \quad = +1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 8 + 4 + 2 + 1 = +15
 \end{aligned}$$

Tabela 2: Zapis realnih brojeva u jednostrukoj tačnosti pomoću binarne osnove prema IEEE 754 standardu

## Formati za razmenu podataka zapisanih pomoću dekadne osnove

### Kodiranje gustim pakovanjem dekadnih cifara

Za kodiranje pojedinačne dekadne cifre pomoću 8421 BCD koda (ili nekog od kodova navedenih u prethodnoj tabeli) koriste se 4 bita. Dekadnih cifara ima 10, a sa druge strane broj mogućih kodova je 16. Očigledno je da pri kodiranju dekadnih brojeva nekim od navedenih kodova veliki deo prostora ostaje neiskorišćen. Postavilo se pitanje da li postoji neko bolje kodiranje dekadnih cifara koje neće ostavljati toliko neiskorišćenog prostora, a sa druge strane neće prilikom dekodiranja opterećivati izvršavanje programa dodatnim operacijama. Jedno od takvih kodiranja, nazvano kodiranje gustim pakovanjem dekadnih cifara (eng. *Densely Packed Decimal, DPD*) definisao je Majk Kovlišou (*Mike Cowlishaw*) ([1]) 2002. godine. Osnovna ideja DPD kodiranja je, podela dekadnih cifara na male (0 do 7) koje zahtevaju 3 bita za međusobno razlikovanje, i velike (8 ili 9) koje zahtevaju jedan bit. Takodje, pri kodiranju se razmatra potreban broj bitova za svaku od kombinacija tri cifre:

- Sve tri cifre su male (51.2% slučajeva): potrebno je 10 bitova (3+3+3 za cifre, 1 bit da označi ovu kombinaciju)
- Dve cifre su male (38.4% slučajeva): potrebno je 7 bitova za cifre (3+3+1); preostala 3 bita označavaju kombinaciju
- Jedna cifra je mala (9.6% slučajeva): potrebno je 5 bitova za cifre (3+1+1); preostalih 5 bitova označavaju kombinaciju
- Sve cifre su velike (0.8% slučajeva): potrebno je 3 bita za cifre (1+1+1); preostalih 7 (potrebno je samo 5) označavaju kombinaciju

### Shematski prikaz kodiranja

U sledećoj tabeli je prikazan način izračunavanja bitova rezultata pri kodiranju. Polazi se od toga da su cifre koje se kodiraju, npr.  $C_1C_2C_3$ , zapisane u BCD kodu i da im odgovara 12-bitna kombinacija: (abcd)(efgh)(ijkl). Kodiranjem se dobija 10-bitna kombinacija (pqr)(stu)(v)(wxy). Vrednost svakog od bitova se može predstaviti kao logička funkcija čiji su argumenti vrednosti bitova u BCD kodiranju. Funkcije na osnovu kojih se izračunavaju vrednosti su prikazane u narednoj tabeli:

aei	pqr	stu	v	wxy	Komentar
000	bcd	fgh	0	jkm	Sve cifre su male
001	bcd	fgh	1	00m	Krajnje desna cifra je velika
010	bcd	jkh	1	01m	Srednja cifra je velika
100	jdk	fgh	1	10m	Krajnje leva cifra je velika
110	jdk	00h	1	11m	Krajnje desna cifa je mala (ostale dve su velike)
101	fgd	01h	1	11m	Srednja cifra je mala (ostale dve su velike)
011	bcd	10h	1	11m	Krajnje leva cifra je mala (ostale dve su velike)
111	00d	11h	1	11m	Sve cifre su velike; dva bita se ne koriste

Parametri kodiranja	Formati			
	binary16	binary32	binary64	binary128
k, širina polja (bita)	16	32	64	128
bit za znak (bita)	1	1	1	1
w, dužina polja za eksponent (bita)	5	8	11	15
p, preciznost (bita)	11	24	53	113
t, ostatak značajnog dela broja (bita)	10	23	52	112
emax, najveća vrednost eksponenta e	15	127	1023	16383
emin, najmanja vrednost eksponenta e	-14	-126	-1022	-16382
Uvećanje E-e	15	127	1023	16383

Najveće i najmanje vrednosti u formatu	
$N_{max}$	$(1-2^{-1}) \times 2^{16}$ $(1-2^{-24}) \times 2^{128}$ $(1-2^{-53}) \times 2^{1024}$ $(1-2^{-113}) \times 2^{16384}$ $\approx 6.5504 \times 10^{+4}$ $\approx 3.4 \times 10^{+38}$ $\approx 1.8 \times 10^{+308}$ $\approx 1.2 \times 10^{+4932}$
$N_{min}$	$1.0 \times 2^{-14}$ $1.0 \times 2^{-126}$ $1.0 \times 2^{-1022}$ $1.0 \times 2^{-16382}$ $\approx 6.1 \times 10^{-5}$ $\approx 1.2 \times 10^{-38}$ $\approx 2.2 \times 10^{-308}$ $\approx 3.4 \times 10^{-4932}$
$D_{min}$	$1.0 \times 2^{-24}$ $1.0 \times 2^{-149}$ $1.0 \times 2^{-1074}$ $1.0 \times 2^{-16494}$ $\approx 5.96 \times 10^{-8}$ $\approx 1.4 \times 10^{-45}$ $\approx 4.9 \times 10^{-324}$ $\approx 6.5 \times 10^{-4966}$

$N_{max}$  – Najveći (po apsolutnoj vrednosti) predstavljiv normalan broj

$N_{min}$  – Najmanji (po apsolutnoj vrednosti) predstavljiv normalan broj

$D_{min}$  – Najmanji (po apsolutnoj vrednosti) predstavljiv subnormalan broj

Tabela 3: Vrednosti parametara formata za razmenu podataka sa binarnom osnovom

Klasa podataka	Znak	Uvećani eksponent E	Implicitni bit	Frakcija	Reprezentacija	Vrednost
Označena nula	S	0	0	T	$(S, emin, 0)$	$(-1)^S \times (+0)$
Normalni brojevi	S	$1 \leq E \leq 2^w - 2$	1	proizvoljno	$(S, (E - uve), (1 + 2^{1-p} \times T))$	$(-1)^S \times 2^{E-uve} \times (1 + 2^{1-p} \times T)$
Subnormalni brojevi	S	0††	0	$\neq 0$	$(S, emin, (0 + 2^{1-p} \times T))$	$(-1)^S \times 2^{emin} \times (0 + 2^{1-p} \times T)$
Beskonačno	S	$2^w - 1$	xxx	0	$(-1)^S \times (+\infty)$	$(-1)^S \times (+\infty)$
Nije broj	xxx	$2^w - 1$	xxx	$d_1 = 1, d_r = \text{proizvoljno} \rightarrow \text{qNaN}$ $d_1 = 0, d_r \neq 0 \rightarrow \text{sNaN}$		NaN

uve Uvećanje eksponenta

xxx Ne primenjuje se

†† U aritmetičkim operacijama se tretira kao da ima vrednost 1

Eksponent za  $E=0$  - sadržaj polja za eksponent su sve nule, a za  $E=2^w - 1$  - sadržaj polja za eksponent su sve jedinice

Frakcija  $d_1$  je krajnje levi bit frakcije;  $d_r$  predstavlja ostale bitovi frakcije

Tabela 4: Klase podataka u zapisu sa binarnom osnovom u IEEE 754

Funkcije navedene u tabeli mogu da budu realizovane hardverski. Ukoliko procesor ne poseduje tu mogućnost, tada se kodiranje može sprovesti i softverski preko sledećih logičkih funkcija:

$$\begin{aligned}
 p &= b \vee (a \wedge j) \vee (a \wedge f \wedge i) \\
 q &= c \vee (a \wedge k) \vee (a \wedge g \wedge i) \\
 r &= d \\
 s &= (f \wedge (\neg a \vee \neg i)) \vee (\neg a \wedge e \wedge j) \vee (e \wedge i) \\
 t &= g \vee (\neg a \wedge e \wedge k) \vee (a \wedge i) \\
 u &= h \\
 v &= a \vee e \vee i \\
 w &= a \vee (e \wedge i) \vee (\neg e \wedge j) \\
 x &= e \vee (a \wedge i) \vee (\neg a \wedge k) \\
 y &= m
 \end{aligned}$$

### Shematski prikaz dekodiranja

Dekodiranje predstavlja izračunavanje vrednosti bitova 12-bitne kombinacije bitova (abcd)(efgh)(ijkl) koje predstavljaju BCD zapis cifara  $C_1C_2C_3$  u BCD zapisu na osnovu vrednosti 10-bitne kombinacije bitova (p)(qrs)(tuv)(wxy). Funkcije na osnovu kojih se izračunavaju vrednosti su prikazane u narednoj tabeli:

vwxst	abcd	efgh	ijkl
0...	0pqr	0stu	0wxy
100..	0pqr	0stu	100y
101..	0pqr	100u	0sty
110..	100r	0stu	0pqy
11100	100r	100u	0pqy
11101	100r	0pqu	100y
11110	0pqr	100u	100y
11111	100r	100u	100y

gde tačka (.) označava da je sadržaj na toj poziciji nebitan. Analogno kodiranju, i za dekodiranje se mogu definisati logičke funkcije koje mogu da se koriste u slučaju da ne postoji hardverski implementirana podrška za DPD kodiranje:

$$\begin{aligned}
 a &= (v \wedge w) \wedge (\neg s \vee t \vee \neg x) \\
 b &= p \wedge (\neg v \vee \neg w \vee (s \wedge \neg t \wedge x)) \\
 c &= q \wedge (\neg v \vee \neg w \vee (s \wedge \neg t \wedge x)) \\
 d &= r \\
 e &= v \wedge ((\neg w \wedge x) \vee (\neg t \wedge x) \vee (s \wedge x)) \\
 f &= (s \wedge (\neg v \vee \neg x)) \vee (p \wedge \neg s \wedge t \wedge v \wedge w \wedge x) \\
 g &= (t \wedge (\neg v \vee \neg x)) \vee (q \wedge \neg s \wedge t \wedge w) \\
 h &= u \\
 i &= v \wedge ((\neg w \wedge \neg x) \vee (w \wedge x \wedge (s \vee t))) \\
 j &= (\neg v \wedge w) \vee (s \wedge v \wedge \neg w \wedge x) \vee (p \wedge w \wedge (\neg x \vee (\neg s \wedge \neg t))) \\
 k &= (\neg v \wedge x) \vee (t \wedge \neg w \wedge x) \vee (q \wedge v \wedge w \wedge (\neg x \vee (\neg s \wedge \neg t))) \\
 m &= y
 \end{aligned}$$

### Karakteristike DPD kodiranja

Neke od karakteristika DPD kodiranja su:

1. Kodiranje jedne ili dve dekadne cifre (u optimalnih 4 ili 7 bitova respektivno) se dobija kao podskup kodiranja 3-cifrenog dekadnog broja. Posledica ovoga je da se proizvoljan broj dekadnih cifara kodira efikasnije. Na primer, 38 cifara se može kodirati pomoću 127 bita, a 71 dekadna cifra u 237 bita.
2. Kodiranje jedne ili dve dekadne cifre je uvek desno poravnato u grupi od 10 bita, dok su ostali bitovi 0. Ovo omogućuje da kodirane dekadne cifre mogu da se zapišu polju veće dužine dopunjavanjem nula sa leve strane, bez potrebe za ponovnim kodiranjem.
3. Pozicija i izbor indikator bita (bit  $v$  u DPD kodiranju) dopuštaju da se svi jednocifreni brojevi (preciznije svi brojevi u intervalu  $[0,79]$  kodiraju desno poravnato na isti način kao u BCD (8421) kodu, što olakšava konverziju takvih brojeva.

U narednoj tabeli je dat primer kodiranja nekih brojeva.

Cifre	BCD	DPD
005	0000 0000 0101	000 000 0101
009	0000 0000 1001	000 000 1001
055	0000 0101 0101	000 101 0101
099	0000 1001 1001	000 101 1111
555	0101 0101 0101	101 101 0101
999	1001 1001 1001	001 111 1111

Primer: Kodirati brojeve 825 i 294 u DPD zapis.

8	2	5	2	9	4	Dekadni zapis		
abcd	efgh	ijkm	abcd	efgh	ijkm	Bitovi u BCD zapisu		
1000	0010	0101	0010	1001	0100	BCD zapis		
100	010	1	101	010	1	DPD zapis		
pqr	stu	v	wxy	pqr	stu	v	wxy	Bitovi u DPD zapisu

### Kohorte

Za razliku od zapisa pomoću binarne osnove, realan broj može da ima više reprezentacija u zapisu pomoću dekadne osnove. Skup različitih reprezentacija u koje se preslikava broj u pokretnom zarezu se naziva *kohorta*. Na primer, ako je  $c$  umnožak od 10 i  $q$  je manje od najveće dozvoljene vrednosti, tada su  $(s, q, c)$  i  $(s, q + 1, c/10)$  dve reprezentacije istog broja u pokretnom zarezu i članovi iste kohorte. Iako numerički jednaki, različiti članovi iste kohorte mogu da se različito ponašaju pri izvodjenju aritmetičkih operacija.

Ako reprezentacija konačnog ne-nula broja ima  $n$  dekadnih cifara, odgovarajuća kohorta će imati najviše  $p - n + 1$  članova gde je  $p$  broj cifara preciznosti datog formata. Na primer, jednocifren realan broj može da ima najviše  $p$  različitih reprezentacija, dok realan broj sa  $p$  cifara koji se ne završava nulama ima najviše jednu reprezentaciju. Kohorta broja u pokretnom zarezu sa  $n$  cifara može da ima manje od  $p - n + 1$  članova ako se broj nalazi jako blizu granica intervala mogućih vrednosti eksponenta. Sa druge strane, nula ima mnogo više članova kohorte: svaka od kohorti za  $+0$  i  $-0$  sadrži po jedan član za svaku moguću vrednost eksponenta.

### Kanonički zapis

Bez obzira da li se kodira konačan broj, beskonačno ili NaN ista vrednost može da se zapiše pomoću različitih kombinacija bitova. Jedna od tih kombinacija se naziva *kanonička*. Razlog uvođenja kanoničke reprezentacije je postojanje velikog broja različitih preslikavanja koja mogu da se koriste za kodiranje tri dekadne cifre (1000 mogućih kombinacija) u 10 binarnih cifara (1024 moguće kombinacije). Komitet za standarde je jednu od kombinacija proglasio kanoničkom (poželjnom, prvenstvenom). Odabrana kanonička reprezentacija je prikazana u tabelama 5 (deklet u dekadne vrednosti) i 6 (dekadne vrednosti u deklet). Kriterijumi za izbor su bili jednostavnost hardverske implementacije navedenih tabela pomoću tri nivoa logičkih i/ili flip-flop elemenata. Tabela 5 ima 1024 moguća ulaza i 1000 vrednosti na izlazu, dok tabela 6 ima po 1000 vrednosti na ulazu i izlazu. Odatle sledi da postoje 24 *nekanoničkih* bit kombinacija koje mogu da budu ulazne ali ne i izlazne vrednosti iz aritmetičkih operacija. Sva enkodiranja koja sadrže neku od nekanoničkih kombinacija bitova se nazivaju nekanonička.

$b_{(6)}, b_{(7)}, b_{(8)}, b_{(3)}, b_{(4)}$	$d_{(1)}$	$d_{(2)}$	$d_{(3)}$
0xxxx	$4b_{(0)} + 2b_{(1)} + b_{(2)}$	$4b_{(3)} + 2b_{(4)} + b_{(5)}$	$4b_{(7)} + 2b_{(8)} + b_{(9)}$
100xx	$4b_{(0)} + 2b_{(1)} + b_{(2)}$	$4b_{(3)} + 2b_{(4)} + b_{(5)}$	$8 + b_{(9)}$
101xx	$4b_{(0)} + 2b_{(1)} + b_{(2)}$	$8 + b_{(5)}$	$4b_{(3)} + 2b_{(4)} + b_{(9)}$
110xx	$8 + b_{(2)}$	$4b_{(3)} + 2b_{(4)} + b_{(5)}$	$4b_{(0)} + 2b_{(1)} + b_{(9)}$
11100	$8 + b_{(2)}$	$8 + b_{(5)}$	$4b_{(0)} + 2b_{(1)} + b_{(9)}$
11101	$8 + b_{(2)}$	$4b_{(0)} + 2b_{(1)} + b_{(5)}$	$8 + b_{(9)}$
11110	$4b_{(0)} + 2b_{(1)} + b_{(2)}$	$8 + b_{(5)}$	$8 + b_{(9)}$
11111	$8 + b_{(2)}$	$8 + b_{(5)}$	$8 + b_{(9)}$

Tabela 5: Dekodiranje 10-bitnih gusto pakovanih dekadnih brojeva zapisanih u bitovima  $b_0 \dots b_9$  u 3 dekadne cifre  $d_0, d_1$  i  $d_2$ . x Označava da je vrednost na toj poziciji nevažna. Ovim preslikavanjem se svih 1024 mogućih 10-bitnih kombinacija preslikava u 1000 mogućih trocifrenih brojeva

$d_{(1,0)}, d_{(2,0)}, d_{(3,0)}$	$b_{(0)}, b_{(1)}, b_{(2)}$	$b_{(3)}, b_{(4)}, b_{(5)}$	$b_{(6)}$	$b_{(7)}, b_{(8)}, b_{(9)}$
000	$d_{(1,1:3)}$	$d_{(2,1:3)}$	0	$d_{(3,1:3)}$
001	$d_{(1,1:3)}$	$d_{(2,1:3)}$	1	$0, 0, d_{(3,3)}$
010	$d_{(1,1:3)}$	$d_{(3,1:2)}, d_{(2,3)}$	1	$0, 1, d_{(3,3)}$
011	$d_{(1,1:3)}$	$1, 0, d_{(2,3)}$	1	$1, 1, d_{(3,3)}$
100	$d_{(3,1:2)}, d_{(1,3)}$	$d_{(2,1:3)}$	1	$1, 0, d_{(3,3)}$
101	$d_{(2,1:2)}, d_{(1,3)}$	$0, 1, d_{(2,3)}$	1	$1, 1, d_{(3,3)}$
110	$d_{(3,1:2)}, d_{(1,3)}$	$0, 0, d_{(2,3)}$	1	$1, 1, d_{(3,3)}$
111	$0, 0, d_{(1,3)}$	$1, 1, d_{(2,3)}$	1	$1, 1, d_{(3,3)}$

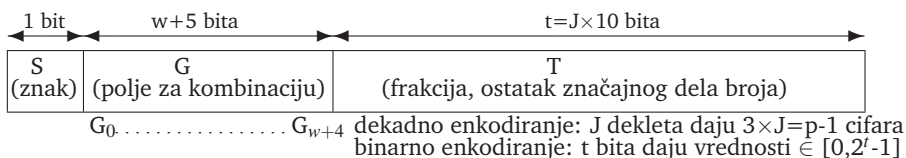
Tabela 6: Kodiranje 3 dekadne cifre u 10-bitno gusto pakovane dekadne brojeva u 3 dekadne brojeve. Bitovi u zapisu dekadnih cifara  $d_{(1)}$ ,  $d_{(2)}$  i  $d_{(3)}$  su izraženi preko drugog indeksa ( $d_{(1,0:3)}$ ,  $d_{(2,0:3)}$  i  $d_{(3,0:3)}$ ), pri čemu je bit označen sa 0 bit najveće težine. Bitovi u dekletu u koji se vrši kodiranje su označeni sa  $b_0 \dots b_9$ . Operacije formiraju samo 1000 kanoničkih dekleta koji su navedeni u ovoj tabeli. Nekanoničkih dekleta ima 24, sa bitskim zapisom oblika  $01x11x111x$ ,  $10x11x111x$ , ili  $11x11x111x$  gde 'x' označava da je vrednost na toj poziciji nevažna.

### Enkodiranje

Zbog nepostojanja saglasnosti u komitetu za standarde, u standardu su predviđene dve mogućnosti za zapis nazvane *dekadno* i *binarno* enkodiranje. Dekadno enkodiranje omogućava jednostavniju hardversku, dok binarno omogućava jednostavniju softversku implementaciju. Bez obzira na izbor kodiranja, skup realnih brojeva koji može da se zapiše je isti u oba kodiranja je isti, tako da krajnji korisnik ne mora da vodi računa o tome koje se kodiranje koristi. Implementacija, takodje, mora da obezbedi mogućnost konverzije između oba načina enkodiranja.

Realan broj u pokretnom zarezu se u formatu za razmenu podataka zapisanom pomoću dekadne osnove zapisuje pomoću  $k$  bitova (slika 3) podeljenih u tri polja čije je kanoničko značenje:

1. Znak broja  $S$  veličine 1 bit
2. Polje za kombinaciju veličine  $w + 5$  bita koje kodira:
  - klasifikaciju (da li je podatak konačan broj,  $\pm\infty$  ili NaN),
  - eksponent  $q$  u slučaju da je kodirani podatak konačan broj. Uvećani eksponent  $E$  je veličine  $w + 2$  bita i iznosi  $q +$  uvećanje.
  - cifru najveće težine značajnog dela konačnog broja (u slučaju dekadnog kodiranja) ili 3-4 bita najveće težine značajnog dela broja (u slučaju binarnog kodiranja).
3. Ostatak  $T$  značajnog dela broja koji sadrži  $t = 10 \times J$  bita. Kombinovanjem sa bitovima najveće težine značajnog dela broja koji se nalaze u polju za kombinaciju dobija se da format omogućava zapisu ukupno  $p = 3 \times J + 1$  dekadnih cifara.



Slika 3: Format za razmenu podataka zapisanih pomoću dekadne osnove

Standard definiše format za razmenu podataka zapisanih pomoću dekadne osnove najmanje veličine 32 bita. U opštem slučaju, veličina formata može da bude bilo koji broj koji je umnožak od 32. U tabeli 7 su prikazane vrednosti parametara za neke formate sa dekadnom osnovom. U slučaju da je dužina formata umnožak od 32,

vrednosti parametara se mogu odrediti na sledeći način:

$$\begin{aligned}
 k &= 1 + 5 + w + t = 32 \times \text{ceiling}((p + 2)/9) \\
 w &= kt - 6 = k/16 + 4 \\
 t &= kw - 6 = 15 \times k/16 - 10 \\
 p &= 3t/10 + 1 = 9 \times k/32 - 2 \\
 emax &= 3 \times 2^{w-1} \\
 emin &= 1 - emax \\
 \text{uvećanje} &= emax + p - 2
 \end{aligned}$$

Parametri kodiranja	Formati			
	decimal32	decimal64	decimal128	decimal{k} (k ≥ 32)
k, širina polja (bita)	32	64	128	umnožak od 32
p, preciznost (cifara)	7	16	34	$9 \times k/32 - 2$
emax, najveća vrednost eksponenta e	96	384	6144	$3 \times 2^{k/16+3}$
emin, najmanja vrednost eksponenta e	-95	-383	-6143	$1 - (3 \times 2^{k/16+3})$
Uvećanje E-q	101	398	6176	emax + p - 2
bit za znak (bita)	1	1	1	1
w+5, dužina polja za kombinaciju (bita)	11	13	17	k/16+9
t, ostatak značajnog dela broja (bita)	20	50	110	15 × k/16 - 10

Najveće i najmanje vrednosti u formatu				
$N_{max}$	$(10-1.0 \times 10^{-6}) \times 10^{+96}$	$(10-1.0 \times 10^{-15}) \times 10^{+384}$	$(10-1.0 \times 10^{-33}) \times 10^{+6144}$	$\approx (10-1.0 \times 10^{-p+1}) \times 10^{emax+1}$
$N_{min}$	$1.0 \times 10^{-95}$	$1.0 \times 10^{-383}$	$1.0 \times 10^{-6143}$	$1.0 \times 10^{1-emax}$
$D_{min}$	$1.0 \times 10^{-101}$	$1.0 \times 10^{-398}$	$1.0 \times 10^{-6176}$	$1.0 \times 10^{emax+p-2}$

$D_{min}$  Najmanji (po apsolutnoj vrednosti) predstavljiv subnormalan broj

$N_{max}$  Najveći (po apsolutnoj vrednosti) predstavljiv normalan broj

$N_{min}$  Najmanji (po apsolutnoj vrednosti) predstavljiv normalan broj

Tabela 7: Vrednosti parametara formata za razmenu podataka sa dekadnom osnovom

Reprezentacija i vrednost podatka u pokretnom zarezu zapisanog pomoću dekadne osnove se određuju na sledeći način:

1. Ako su 5 bitova najveće težine kombinacije ( $G_0G_1G_2G_3G_4$ ) 11111, vrednost broja je NaN. Ako je bit  $G_5 = 1$  tada je vrednost sNaN; u suprotnom je qNaN. Ostalih w-1 bitova polja za kombinaciju se ignoriše, dok se frakcija T koristi radi bliže specifikacije NaN vrednosti. U kanoničkoj reprezentaciji NaN vrednosti svi bitovi  $G_6 \dots G_{w+4}$  su jednaki 0.
2. Ako su 5 bitova najveće težine kombinacije ( $G_0 \dots G_4$ ) jednaki 11110, reprezentacija i vrednost broja su jednaki  $(-1)^S \times (+\infty)$ . Ostalih w bitova u polju za kombinaciju i frakcija se ignorišu. Kanoničke reprezentacije  $+\infty$  i  $-\infty$  poseduju bitove od  $G_5$  do  $G_{w+4}$  jednake 0, i T=0. U nekim implementacijama preostalih w bitova polja za kombinaciju mogu da se koriste radi bliže specifikacije beskonačnosti u operacijama, ali nemaju uticaja na određivanje vrednosti broja koja je jednaka beskonačno.
3. Ako su 5 bitova najveće težine kombinacije ( $G_0G_1G_2G_3G_4$ ) u intervalu 00000-11101, tada je u pitanju konačan broj. Reprezentacija konačnog broja je jednaka  $r = (S, E - \text{uvećanje}, C)$ , dok je njegova vrednost jednaka  $v = (-1)^S \times 10^{E - \text{uvećanje}} \times C$  gde je C dobijeno dopisivanjem cifre najveće težine ili bitova iz polja za kombinaciju G na frakciju T (ostatka značajnog dela broja). Pri tome je uvećani eksponent E zapisan u polju za kombinaciju. Enkodiranje podataka u polju za kombinaciju zavisi od toga da li implementacija za zapis značajnog dela broja koristi dekadno ili binarno enkodiranje.

(a) U slučaju dekadnog enkodiranja bitovi od  $G_5$  do  $G_{w+4}$  čine w bita najmanje težine eksponenta, dok se dva bita najveće težine eksponenta i dekadne cifre  $d_0d_1 \dots d_{p-1}$  određuju na sledeći način:

- i. Ako su 5 bitova kombinacije u intervalu 0xxxx-10xxx, tada su dva bita najveće težine kombinacije  $G_0G_1$  dva bita najveće težine eksponenta. Vrednost cifre najveće težine frakcije  $d_0$  je jednaka  $4G_2 + 2G_3 + G_4$  i pripada intervalu  $[0,7]$ .

ii. Ako su 5 bitova kombinacije u intervalu 110xx-1110xx, tada je dekadna vrednost cifre najveće težine frakcije  $d_0$  jednaka  $8+G_4$  (i iznosi 8 ili 9 u zavisnosti od vrednosti  $G_4$ ), dok su  $G_2G_3$  cifre najveće težine eksponenta.

Ako je  $T=0$  i pet cifara najveće težine u polju za kombinaciju je jednako 00000, 01000, ili 10000 tada je vrednost broja  $v = (-1)^S \times (+0)$ .

Frakcija sadrži  $3 \times J = p-1$  dekadnih cifara  $d_1 \dots d_{p-1}$  enkodiranih dekadnih cifara pomoću gustog pakovanja dekadnih cifara (DPD kodiranje). Kanonički oblik značajnog dela broja sadrži jedino kanoničke deklete. Računske operacije generišu samo 1000 kanoničkih dekleta, ali takodje prihvataju i preostala 24 ne-kanonička dekleta kao operande.

(b) U slučaju binarnog enkodiranja (BID, eng. *binary-integer decoding*) dekadne cifre u značajnom delu broja su zapisane kao celobrojne vrednosti u binarnoj osnovi. Cifre eksponenta i cifra najveće težine frakcije  $d_0$  se određuju na sledeći način:

- i. Ako su bitovi  $G_0G_1$  00, 01 ili 10 tada se uvećani eksponent E formira od cifara  $G_0$  do  $G_{w+1}$ , dok se značajni deo broja formira počev od bita  $G_{w+2}$  pa do kraja zapisa (uključujući i frakciju T).
- ii. Ako su bitovi  $G_0G_1$  11, a  $G_2G_3$  jedna od vrednosti 00, 01 ili 10 tada se uvećani eksponent E formira od cifara  $G_2$  do  $G_{w+3}$ , dok se značajni deo broja formira dodajući 4 bita koji imaju dekadnu vrednost  $8+G_{w+4}$  (tj. binarno  $100G_{w+4}$ ) na početak frakcije T.

Najveća vrednost u ovom načinu enkodiranja značajnog dela je ista kao i u slučaju dekadnog enkodiranja i iznosi  $10^{3 \times J + 1} - 1$  (odnosno  $10^{3 \times J} - 1$  kada se frakcija T koristi radi bliže specifikacije NaN vrednosti. Ako vrednost prelazi maksimum, značajni deo  $c$  je nekanonički i  $c$  dobija vrednost 0.

Računske operacije generišu samo 1000 kanoničkih dekleta, ali takodje prihvataju i preostala 24 ne-kanonička dekleta kao operande.

Enkodiranje nije imalo uticaja na izbor veličine uvećanja eksponenta i broja cifara u značajnom delu broja. Ove vrednosti su određene na osnovu veličine registara računara i poredjenja zapisom pomoću binarne osnove. Kako binary32 omogućava zapis dekadnog broja sa 6-7 cifara, ista tačnost je trebala da bude omogućena i pomoću dekadnog zapisa. Za zapis dekadnih 7 cifara u slučaju dekadnog kodiranja potreban su dva dekleta i dodatni prostor za zapis još jedne cifre (maksimalno 4 bita), dok je u slučaju binarnog kodiranja potrebno 24 bita. Kako je jedan bit potreban za znak, za moguće vrednosti eksponenta ostaje još 7 bitova (tj. najviše 128 različitih vrednosti). Medjutim, kako bi korišćenje sva 24 bita (u oba kodiranja) predstavljalo nepotreban utrošak prostora (jer u slučaju DPD-a mala cifra se kodira za 3 a velika sa 1 bitom, dok se u slučaju binarnog zapisa sa 24 bita zapisuje broj 16777216 koji je veći od 9999999 što je najveći 7-cifren broj) to je dodatna veličina eksponenta dobijena mešanjem bitova koji označavaju cifru najveće težine sa bitovima eksponenta u polje za kombinaciju. Način određivanja vrednosti eksponenta i njegovog uvećanja opisan je u [2].

U tabelama 8 i 9 prikazani su zapisi brojeva pomoću dekadne osnove u jednostrukoj tačnosti pomoću dekadnog i binarnog enkodiranja.

	Znak	Kombinacija	Nastavak frakcije
+15	= 0	01100101000	00000000000000001111
-15	= 1	01100101000	00000000000000001111
+15.0	= 0	01100100000	0000000000010010110
+1/64	= 0	01011111000	00000011110100001001
+0	= 0	01100101000	00000000000000000000
-0	= 1	01100101000	00000000000000000000
+0.0	= 0	01100100000	00000000000000000000
+0.00	= 0	01100011000	00000000000000000000
$+(10^7 - 1) \times 10^{+90}$	= 0	11101111111	10001001011001111111
$+1 \times 10^{-95}$	= 0	00000110000	00000000000000000001
$+1 \times 10^{-101}$	= 0	00000000000	00000000000000000001

Provera vrednosti zapisa broja +15:

- Znak = +

- Eksponent = 0 ( $(01100101)_2$  - uvećanje =  $(101)_{10}$  - uvećanje =  $101 - 101 = 0$ )

- Frakcija =  $(15)_{10} = (0000000000000000010101)_2$  (kodirana u 23 bita)

- Vrednost = Znak frakcija \*  $10^{\text{eksponent}}$  =  $+15 * 10^0 = +15_{10}$

Tabela 9: Primer zapisa realnih brojeva pomoću dekadne osnove /BID kodiranje

	Znak	Kombinacija	Nastavak frakcije
+15 = 0	0	01000 100101	0000 0000 0000 0001 0101
-15 = 1	1	01000 100101	0000 0000 0000 0001 0101
+15.0 = 0	0	01000 100100	0000 0000 0000 1101 0000
+1/64 = 0	0	01000 011111	0000 0101 0111 0010 0101
+0 = 0	0	01000 100101	0000 0000 0000 0000 0000
-0 = 1	1	01000 100101	0000 0000 0000 0000 0000
+0.0 = 0	0	01000 100100	0000 0000 0000 0000 0000
+0.00 = 0	0	01000 100011	0000 0000 0000 0000 0000
$+(10^7 - 1) \times 10^{+90}$ = 0	0	11101 111111	1111 1111 1111 1111 1111 nk.d.
$+(10^7 - 1) \times 10^{+90}$ = 0	0	11101 111111	0011 1111 1100 1111 1111 k.d.
$+1 \times 10^{-95}$ = 0	0	00000 000110	0000 0000 0000 0000 0001
$+1 \times 10^{-101}$ = 0	0	00000 000000	0000 0000 0000 0000 0001

Broj  $+1 \times 10^{-101}$  je zapisan na dva načina: pomoću nekanoničkog i kanoničkog dekleta.

Provera vrednosti zapisa broja +15:

- Znak = +

- EkspONENT = 0 (=101 - 101)

- Frakcija =  $(15)_{10} = 0000000000 0000010101$  (kodirana u dva dekleta)

- Vrednost = Znak frakcija \*  $10^{\text{ekspONENT}} = +15 * 10^0 = +15_{10}$

Tabela 8: Primer zapisa realnih brojeva pomoću dekadne osnove /DPD kodiranja

## Prošireni i proširivi zapis

Standard preporučuje korišćenje *proširenog* i *proširivog* zapisa radi povećanja preciznosti u operacijama koje zahtevaju veću preciznost od one koju pružaju osnovni formati. Pri tome, format zapisa sa proširenom preciznošću se definiše kao format koji proširuje podržane osnovne formate kako po preciznosti tako i po opsegu eksponenta, dok se proširivi format zapisa definiše kao format zapisa kod koga korisnik određuje veličine preciznosti i opsega. U standardu se ne navode stroge granice za parametre koji određuju ove formate zapisa, već se daju specifikacije njihovih donjih granica. Takodje, standard ne zahteva implementaciju ni jednog od ovih formata zapisa, ali u slučaju da se implementiraju moraju da zadovoljavaju karakteristike odgovarajućih formata za razmenu. U suštini, prošireni formati zapisa su uvedeni da bi se podržalo obavljanje numeričkih izračunavanja. Ako u implementaciji za neki osnovni format  $X$  postoji odgovarajući prošireni format  $X_p$ , tada će se sve operacije nad podacima zapisanim o osnovnom formatu  $X$  izvršavati na podacima zapisanim u proširenom formatu  $X_p$ . Kako se radi sa većom preciznošću manja je mogućnost pojave grešaka u krajnjem rezultatu koji se prikazuje pomoću osnovnog formata. Takodje, zbog većeg opsega eksponenta manje je prekida zbog prekoračenja u medjukoracima izračunavanja kod kojih je krajnji rezultat korektan. Parametri koji definišu proširene formate zapisa su prikazani u tabeli 10.

Parametri	Prošireni formati pridruženi formatu				
	binary32	binary64	binary128	decimal64	decimal128
p cifara $\geq$	32	64	128	22	40
emax	+1023	+16383	+65535	+6144	+24576

Tabela 10: Parametri proširenih formatia zapisa brojeva u IEEE 754-2008

## Zaokruživanje

Po IEEE 754 standardu zaokruživanje uzima broj za koga se smatra da je zapisa sa beskonačnom tačnošću, i, ako je potrebno, modifikuje ga tako da se ukalupi u ciljni format. Tom prilikom može da dodje do generisanja signala za porekoračenje, potkoračenje ili netačnu operaciju. Sem u slučaju da je drugačije naznačeno, svaka operacija se obavlja kao da je prvo formiran medjurezultat sa beskonačnom tačnošću i neograničenim opsegom, a zatim se dobijena vrednost zaokruži prema odgovarajućim atributima. Zaokruživanje se ne odnosi na NaN vrednosti.

U standardu postoje pet moguća načina zaokruživanja koji su podeljeni u dve grupe:

### 1. Zaokruživanje na najbližu vrednost.

- **Zaokruživanje na parnu cifru.** Medjurezultat izračunavanja se zaokružuje na najbližu predstavljivu vrednost, uz zaokruživanje na parnu cifru kada je izračunati medjurezultat na sredini intervala između dve predstavljive vrednosti. Ovaj način zaokruživanja je predefinisano za zapis sa binarnom osnovom, dok je za zapis sa dekadnom osnovom preporučen (ai ne i obavezan) kao predefinisani način zaokruživanja.
- **Zaokruživanje na udaljeniju vrednost.** Medjurezultat izračunavanja se zaokružuje na najbližu predstavljivu vrednost, uz zaokruživanje na veću (po apsolutnoj vrednosti) vrednost kada je izračunati medjurezultat na sredini intervala između dve predstavljive vrednosti.

### 2. Usmereno zaokruživanje

- **Zaokruživanje ka pozitivnom.** Medjurezultat se zaokružuje na veću vrednost (moguće i  $+\infty$ ). Ovaj način zaokruživanja se govori s naziva i zaokruživanje ka  $+\infty$ . Naziv “prema  $+\infty$ ” potiče od činjenice da ovakvim zaokruživanjem izračunata vrednost ne može da se smanji, ali može da se poveća (do  $+\infty$ ). Realizacija ovog načina zaokruživanja može da se izvede u dva koraka:
  - Ako je broj pozitivan i ako postoji bar jedna jedinica na nekoj poziciji desno od poslednje pozicije koja se čuva u zapisu, na tom mestu se dodaje jedinica.
  - Bez obzira na znak broja, odbace se sve cifre desno od poslednje pozicije koja se čuva u zapisu.
- **Zaokruživanje ka negativnom.** Medjurezultat se zaokružuje na manju vrednost. Ovaj način zaokruživanja se govori s naziva i zaokruživanje prema  $-\infty$ . Naziv “prema  $-\infty$ ” potiče od činjenice da ovakvim zaokruživanjem izračunata vrednost ne može da se poveća ali može da se smanji (do  $-\infty$ ). Realizacija ovog načina zaokruživanja može da se izvede u dva koraka:
  - Ako je broj negativan i ako postoji bar jedna jedinica na nekoj poziciji desno od poslednje pozicije koja se čuva u zapisu, na tom mestu se oduzima jedinica.
  - Bez obzira na znak broja, odbace se svi bitovi desno od poslednje pozicije koja se čuva u zapisu.
- **Zaokruživanje ka 0.** Rezultat je broj u pokretnom zarezu koji je najbliži broju, ali ne i veći po apsolutnoj vrednosti od broja koji se zaokružuje. Ova tehnika je najjednostavnija i sastoji se u odbacivanju svih cifara (binarnih ili dekadnih) desno od poslednje pozicije koja se čuva u zapisu.

U svakom od prethodnih načina zaokruživanja broj se zaokružuje ka  $\infty$  bez promene u znaku ako je po apsolutnoj vrednosti jednak najmanje  $\beta^{emax} \times (\beta - \beta^{1-p}/2)$ , gde su  $\beta$  i  $p$  određeni formatom zapisa.

## 3 Aritmetika u pokretnom zarezu

Osnovne aritmetičke operacije u pokretnom zarezu se vrše u skladu sa pravilima aritmetičkih operacija za stepene jednakih osnova. Pri izvođenju operacija eksponenti i frakcije brojeva se tretiraju kao brojevi u nepokretnom zarezu i to eksponenti kao celi neoznačeni brojevi a frakcije kao razlomljeni brojevi (za zapis sa binarnom osnovom) odnosno kao celi brojevi (za zapis sa dekadnom osnovom). Pri tome se eksponent i frakcije razdvajaju i drže u različitim registrima. Po završetku operacije izračunate vrednosti frakcije i eksponenta se ponovo spajaju i formiraju broj u pokretnom zarezu koji je rezultat operacije. U opisima algoritama aritmetičkih operacija pretpostavka je da su brojevi u pokretnom zarezu zapisani u skladu sa IEEE 754 standardom. Slični algoritmi važe i za operacije nad brojevima koji su zapisani u pokretnom zarezu na način prikazan u delu 4.

Postoje određene specifičnosti koje se javljaju kod brojeva zapisanih pomoću dekadne osnove. Jedna od njih je izračunavanje dekadnog eksponenta. Kako realan broj u pokretnom zarezu zapisan pomoću dekadne osnove može da ima više reprezentacija, aritmetika sa dekadnim brojevima uključuje i izbor odgovarajućeg člana kohorte. U slučaju da operacija sa dekadnim brojevima<sup>7</sup> ne proizvodi tačan broj kao rezultat da bi se dobio najveći broj značajnih cifara koristi se član kohorte sa najmanjim mogućim eksponentom. Ako je rezultat tačna vrednost, bira se član kohorte zasnovan na ciljanom (poželjnom, očekivanom) eksponentu za rezultat operacije. Ciljani eksponent je funkcija od eksponenata ulaznih operanada. Tako, za konačno  $x$ , u zavisnosti od reprezentacije nule,  $0+x$  može kao rezultat da proizvede različite elemente kohorte. Ako kohorta rezultata ne uključuje član sa ciljanim eksponentom, za rezultat se bira član čiji je eksponent najbliži ciljanom.

<sup>7</sup>Sem operacija kvantizacije i roundToIntegralExact.

## Totalno uredjenje predstavljenih vrednosti

Druga specifičnost je vezana za poredjenje brojeva. Standard definiše totalno uredjenje izmedju vrednosti koje mogu da se predstave. Poredak je zasnovan na vrednostima (bit kombinacijama) znakova, eksponenata i frakcije. Svaka pozitivna vrednost je veća od negativne. Ukoliko su vrednosti istog znaka, vrši se leksikografsko predjenje (bit po bit) vrednosti eksponenata i frakcija. Pošto je predstavljanje pomoću binarne osnove jedinstveno tu se ne javljaju nedoumice oko uredjenja brojeva. Kod zapisa pomoću dekadne osnove, zbog mogućnosti različitog predstavljanja iste vrednosti i načina poredjenja dobija se poredak izmedju istih vrednosti koje su zapisane sa različitim brojem značajnih cifara (tj. nulama na kraju). Tako je, na primer,  $12.5 > 12.50 > 12.500$  jer se radi zapisa predstavljaju kao  $125 \times 10^{-1}$ ,  $1250 \times 10^{-2}$  i  $12500 \times 10^{-3}$ . Kako izmedju eksponenata važi da je  $-1 > -2 > -3$  to odlučuje i redosled izmedju njihovih zapisa.

Posledica totalnog uredjenja je da važi poredak (gledajući samo eksponente)

$$\begin{aligned} (+)qNaN > (+)sNaN > +\infty > N_{max} > +\text{konačan broj} > +N_{min} > +D_{min} > 0 > \\ > -D_{min} > -N_{min} > -\text{konačan broj} > -N_{max} > -\infty > (-)sNaN > (-)qNaN \end{aligned}$$

Minus je u zagradi ispred oznaka za NaN jer, formalno gledano, NaN-ovi ne predstavljaju vrednosti već samo oznake da nije u pitanju broj, pa se i ne koriste u poredjenjima sa ostalim vrednostima.

## Specijalne vrednosti u aritmetičkim operacijama

Podaci koji učestvuju u aritmetičkim operacijama mogu da pripadaju bilo kojoj od IEEE klasa podataka. Ako podatak predstavlja normalan broj, subnormalan broj ili nulu, rezultat se dobija u skladu sa uobičajenim načinom izračunavanja. Pravilo za odredjivanje rezultata operacije koja ima beskonačno kao operand je jednostavno: zameni se  $\infty$  sa konačnim brojem  $x$  i odredi granica kada  $x \rightarrow \infty$ . Tako je  $45 / +\infty = 0$  jer  $\lim_{x \rightarrow +\infty} 45/x = 0$ .

Slično je i  $100 - \infty = -\infty$  i  $\sqrt{\infty} = \infty$ . Ako pak granica kada  $x \rightarrow \pm\infty$  ne postoji, tada je rezultat operacije NaN. Ako je jedan od argumenata izraza NaN, tada je vrednost celog izraza NaN, a ako je argument  $\pm\infty$  tada vrednost izraza može da bude važeći broj u pokretnom zarezu, kao npr.  $1 / \pm\infty = 0$ .

Arithmetičke operacije i operacije poredjenja sa beskonačnim vrednostima uvek kao rezultat daju važeću vrednost u skladu sa matematičkim pravilima. Kao što je prethodno već rečeno, važi

$$-\infty < \text{bilo koji konačan broj} < +\infty$$

Operacije sa beskonačnostima su občno tačne i ne generišu izuzetke. Sa izuzetkom operacija koje proizvode qNaN (tabela 11) svaka aritmetička operacija koja uključuje  $\infty$  takodje daje  $\infty$  kao rezultat. Na primer,

$$\begin{aligned} x \pm (+\infty) &= \pm\infty \text{ (za konačno } x) \\ x \pm (-\infty) &= \mp\infty \text{ (za konačno } x) \\ x \times (\infty) &= \infty \text{ (za konačno ili beskonačno } x \neq 0) \\ (+\infty) + (+\infty) &= +\infty \\ (+\infty) - (-\infty) &= +\infty \\ (-\infty) + (-\infty) &= -\infty \\ (-\infty) - (+\infty) &= -\infty \\ (-\infty) - (+\infty) &= -\infty \\ \sqrt{+\infty} &= +\infty \end{aligned}$$

ostatak pri deljenju  $x$  sa  $\infty$  ( $x \text{ REM } \infty$ ) za konačno i normalano  $x$  konverzija  $\infty$  u  $\infty$  u različitom formatu

Obe NaN vrednosti su podržane u svim aritmetičkim operacijama. Signalni NaN označava prisustvo neinicijalizovanih promenljivih, pogrešne operacije ili proširenja u aritmetici koja nisu deo standarda. Tihi NaN može (u zavisnosti od implementacije) da označi postojanje informacija o pogrešnim ili nepostojećim podacima i rezultatima. Preporuka je da se što je moguće više takvih dijagnostičkih informacija prosledjuje kroz operacije tako što će rezultat operacija u kojima učestvuje NaN takodje biti NaN. Dijagnostičke informacije se zapisuju u preostalim ciframa frakcije koje ne utiču na vrstu NaN-a. Neke od operacija koje proizvode tihi NaN su prikazane u tabeli 11 Ukoliko je qNaN bar jedan od argumenata operacije koja treba da kao rezultat da broj u pokretnom zarezu, a sama operacija nije odredjivanje maksimuma ili minimuma, preporuka je da i rezultat te operacije bude jedan od ulaznih qNaN-ova.

Operacija	QNaN formiran pomoću
Sabiranje, oduzimanje	$(\pm\infty) \pm (\mp\infty)$
Množenje	$0 \times \infty$
Deljenje	$0/0, \infty/\infty$
Ostatak	$x \text{ REM } 0, \infty \text{ REM } y$
Kvadratni koren	$\sqrt{x}$ kada je $x < 0$

Tabela 11: Operacije koje proizvode Tihi NaN

## Sabiranje i oduzimanje

Neka su brojevi  $x$  i  $y$  zapisani u pokretnom zarezu kao  $x = x_s \times \beta^{x_e}, y = y_s \times \beta^{y_e}$  i neka treba izračunati njihov zbir ili razliku. U opštem slučaju dobijena vrednost će biti jednaka

$$x \pm y = \begin{cases} (x_s \times \beta^{x_e - y_e} \pm y_s) \times \beta^{y_e} & \text{ako } x_e \leq y_e \\ (x_s \pm y_s \times \beta^{y_e - x_e}) \times \beta^{x_e} & \text{ako } y_e < x_e \end{cases}$$

Pri izvodjenju operacija treba voditi računa o generisanju i propagaciji specijalnih vrednosti. Osnovni koraci u algoritmu za sabiranje i oduzimanje su:

1. Provera postojanja specijalnih vrednosti. Ukoliko je neki od argumenata operacije specijalna vrednost, rezultat se određuje prema pravilima navedenim u delu 3.
2. Oduzimanje  $x - y$  se se realizuje kao  $x + (-y)$  uz prethodnu promenu znaka argumenta  $y$ .
3. Ukoliko je jedan od sabiraka jednak nuli, vrednost drugog sabirka je rezultat sabiranja.
4. Svodjenje sabiraka na jednake eksponente. Svodjenje se vrši povećavanjem manjeg eksponenta uz istovremeno pomeranje cifara frakcije udesno. Pomeranje udesno se vrši za onoliko mesta za koliko je povećana vrednost eksponenta.<sup>8</sup> Ako pri pomeranju značajan deo broja postane nula, tada vrednost drugog sabirka predstavlja vrednost rezultata.
5. Sabiraju se značajni delovi sabiraka, pri čemu se uzimaju u obzir njihovi znaci. Sabiranje se vrši po pravilima za sabiranje celih brojeva u zapisu znak i apsolutna vrednost. Ukoliko je dobijeni rezultat nula, tada je ukupan zbir nula. Ako je pak pri sabiranju došlo do prekoračenja, dobijeni rezultat se pomera za jedno mesto udesno uz povećanje vrednosti eksponenta za jedan. Ako ovo povećanje vrednosti eksponenta dovede do prekoračenja (vrednosti eksponenta), ukupan rezultat sabiranja je  $+\infty$  ili  $-\infty$  u zavisnosti od znaka broja.
6. Traženi zbir predstavlja broj u pokretnom zarezu čiji su znak i značajan deo jednaki znaku i zbiru značajnih delova sabiraka, a eksponent jednak eksponentu sabiraka. Ako u rezultatu sabiranja značajan deo nije normalizovan (u binarnom zapisu) ili nije predstavljen pomoću ciljanog eksponenta (u dekadnom zapisu), pokušava se njegova normalizacija odnosno traži ciljani eksponent. Ukoliko je tom prilikom došlo do potkoračenja vrednosti eksponenta, ukupan zbir je nula, dok se u ostalim slučajevima po potrebi vrši zaokruživanje i formira traženi zbir.

## Množenje i deljenje

Implementacija množenja i deljenja je jednostavnija od implementacije sabiranja i oduzimanja jer se ne vrši svodjenje na isti eksponent. Neka su brojevi  $x$  i  $y$  zapisani u pokretnom zarezu kao  $x = x_s \times \beta^{x_e}, y = y_s \times \beta^{y_e}$  i neka treba izračunati njihov proizvod ili količnik. U opštem slučaju dobijena vrednost je

$$x * y = (x_s * y_s) \times \beta^{x_e + y_e}$$

$$x / y = (x_s / y_s) \times \beta^{x_e - y_e}$$

Pri izvodjenju operacija potrebno je voditi računa o formiranju i propagaciji specijalnih vrednosti. Osnovni koraci u algoritmu za množenje su:

<sup>8</sup>Ako se brojevi pišu pomoću heksadekadne osnove svako povećanje vrednosti eksponenta za jedan znači pomeranje 4 bita frakcije udesno.

1. Proverava se postojanje specijalnih vrednosti. Ukoliko neki od argumenata operacije predstavlja specijalnu vrednost, rezultat se određuje prema pravilima navedenim u delu 3.
2. Ukoliko je bar jedan od činilaca jednak nuli, rezultat je 0.
3. Saberu se vrednosti eksponenata i od dobijenog zbira oduzme uvećanje. Ako je došlo do prekoračenja pri ovom sabiranju, krajnji rezultat je  $\pm\infty$  u zavisnosti od znaka brojeva  $x$  i  $y$ . Ako je pak došlo do potkoračenja vrednosti eksponenta, krajnji rezultat je pozitivna ili negativna (u zavisnosti od znaka brojeva  $x$  i  $y$ ) nula.
4. Pomnože se značajni delovi brojeva. Množenje se vrši prema pravilima za množenje celih brojeva zapisanih pomoću znaka i apsolutne vrednosti.
5. Dobijeni rezultat se normalizuje, odnosno odredi se ciljani eksponent sličnim postupkom kao kod sabiranja.
6. Broj cifara u proizvodu je dvostruko veći od broja cifara vrednosti koje su pomnožene; cifre koje su višak se odbacuju u procesu zaokruživanja.

Osnovni koraci u algoritmu za deljenje su:

1. Proverava se postojanje specijalnih vrednosti. Ukoliko je neki od argumenata operacije specijalna vrednost, rezultat se određuje prema pravilima navedenim u delu 3.
2. Ako je delilac nula, tada
  - Ako je deljenik  $\neq 0$ , količnik je  $\pm\infty$  u zavisnosti od znaka  $x$ .
  - Ako je deljenik  $= 0$ , tada je rezultat NaN.
3. Oduzmu se vrednosti eksponenata i na dobijenu razliku doda uvećanje. Ako je došlo do prekoračenja pri ovom sabiranju, krajnji rezultat je  $\pm\infty$  u zavisnosti od znaka brojeva  $x$  i  $y$ . Ako je pak došlo do potkoračenja vrednosti eksponenta, krajnji rezultat je pozitivna ili negativna (u zavisnosti od znaka brojeva  $x$  i  $y$ ) nula.
4. Podele se značajni delovi brojeva. Deljenje se vrši prema pravilima za deljenje celih brojeva zapisanih pomoću znaka i apsolutne vrednosti.
5. Dobijeni rezultat se normalizuje, odnosno odredi se ciljani eksponent sličnim postupkom kao kod sabiranja.
6. Dobijeni količnik se zaokružuje prema pravilima za zaokruživanje.

### Izuzeta stanja, zastavice i zamke

Pri pojavi izuzetog stanja (kao npr. deljenja sa nulom ili prekoračenja) prema IEEE standardu predaje se rezultat i nastavlja sa radom. Tipični predefinisani rezultati su NaN za  $0/0$ ,  $i/\infty$  ili prekoračenje za  $1/0$ . IEEE 754 standard deli izuzeta stanja u 5 klasa: prekoračenje, potkoračenje, deljenje sa nulom, pogrešna operacija i netačnost.<sup>9</sup> Za svako od ovih izuzeća se postavlja posebna zastavica (eng. *flag*) koju korisnik može programski da ispituje.

Standard takodje omogućuje upotrebu programskih rutina za rad sa zamkama (eng. *trap handler*). Pored omogućivanja rada programa napisanih u prethodnom periodu koji su koristili zamke, upotreba zamki je neophodna i da bi se prevazišle situacije kao na primer:

```
do    uslov
until (x >= 100)
```

Kako upotreba relacije  $<$ ,  $<=$ ,  $>$ ,  $>=$ ,  $=$  za poredjenje NaN sa brojem uvek vraća vrednost netačno, prethodni kod će bez upotrebe zamki ući u beskonačni ciklus ako  $x$  postane NaN.

<sup>9</sup>Netačnost se javlja npr. kada zaokruženi rezultat operacije u pokretnom zarezu nije jednak izračunatom medjurezultatu, kao i kada pokušaj zaokruživanja dovede do prekoračenja.

## Primeri zapisa brojeva pomoću dekadne osnove

**Primer1:** Zapisati broj  $+123.4$  u pokretnom zarezu pomoću dekadne osnove u jednostrukoj tačnosti, koristeći

1. dekadno (DPD) kodiranje
2. binarno (BID) kodiranje

**Rešenje:** Broj koga treba zapisati predstavimo u obliku  $+1234 \cdot 10^{-1}$ . Vrednosti cifara u zapisu broja su:

1. DPD kodiranje

- (a) Broj koga treba zapisati je pozitivan  $\rightarrow$  znak broja je 0.
- (b) Značajan deo broja je 1234. Dopunimo ga do maksimalnog broja cifara za jednostruku tačnost (7). Dobijeni broj koga treba da kodiramo je 0001234. Kodiraju se 6 cifara manje težine u dva dekleta, dok se cifra najveće težine kodira u polju za kombinaciju. Primenom DPD kodiranja dobija se (npr. na osnovu tabele shematskog prikaza kodiranja):

0	0	1	2	3	4	Dekadni zapis		
abcd	efgh	ijklm	abcd	efgh	ijklm			
0000	0000	0001	0010	0011	0100	BCD zapis		
000	000	0	001	010	011	0	100	DPD dekleti
pqr	stu	v	wxy	pqr	stu	v	wxy	

- (c) Eksponent je jednak -1, i uz uvećanje 101 dobija se 100. Prevod dobijene vrednosti u binarni brojni sistem je 01100100. Odavde je nastavak eksponenta 100100, dok se prva dva bita 01 kodiraju u kombinaciji.
- (d) Kombinacija treba da kodira cifru 0 (cifra najveće težine značajnog dela) i dva bita najveće težine eksponenta (binarne cifre 01). Kako je 0 mala cifra, na osnovu pravila za predstavljanje sledi da je kombinacija = 01000.

Odavde, zapis broja  $+123.4$  je 0 01000 100100 0000 0000 0101 0011 0100

2. BID kodiranje

- (a) Broj koga treba zapisati je pozitivan  $\rightarrow$  znak broja je 0.
- (b) Značajan deo broja je 1234. Dopunimo ga do maksimalnog broja cifara za jednostruku tačnost (7). Dobijeni broj koga treba da kodiramo je 0001234. Prevod ovog broja u binarni sistem je 10011010010. Posto se u jednostrukoj tačnosti značajan deo broja zapisuje u bar 23 bita, traženi prevod se dobija dodavanjem nula sa leve strane i iznosi 00000000000010011010010.
- (c) Eksponent je jednak -1, i uz uvećanje 101 dobija se 100. Prevod dobijene vrednosti u binarni brojni sistem je 01100100.
- (d) Kombinacija treba da kodira klasifikaciju, bitove eksponenta i cifru najveće težine značajnog dela broja. Kako prevod značajnog dela broja u binarni sistem pocinje nulom (tj. može da se zapiše u 23 bita bez menjanja vrednosti) to su prva dva bita kombinacije ili 0x ili 10. U tom slučaju se eksponent nalazi na početku kombinacije, a za njim slede 3 bita koja se dodaju na zapis značajnog dela. Posto su tri bita najveće težine zapisa značajnog dela jednaka 000, kombinacija ima oblik 01100100000.

Odavde, zapis broja  $+123.4$  je 0 01100100000 00000000010011010010

**Primer 2:** Zapisati broj  $+982.5294 \cdot 10^{42}$  u pokretnom zarezu pomoću dekadne osnove u jednostrukoj tačnosti, koristeći oba načina kodiranja.

**Rešenje:** Dati broj zapišimo u obliku  $+9825294 \cdot 10^{38}$ . Vrednosti cifara u zapisu broja su:

1. DPD kodiranje

- (a) Broj koga treba zapisati je pozitivan  $\rightarrow$  znak broja je 0.

- (b) Značajan deo broja je 9825294. Šest cifara manje težine se kodiraju u dva dekleta, dok se cifra najveće težine kodira u polju za kombinaciju. Primenom DPD kodiranja dobija se (npr. na osnovu tabele shematskog prikaza kodiranja):

8	2	5	2	9	4	Dekadni zapis		
abcd	efgh	ijklm	abcd	efgh	ijklm			
1000	0010	0101	0010	1001	0100	BCD zapis		
100	010	1	101	010	101	1	010	DPD dekleti
pqr	stu	v	wxy	pqr	stu	v	wxy	

- (c) Eksponent je jednak +38, i uz uvećanje 101 dobija se 139. Prevod dobijene vrednosti u binarni brojni sistem je 10001011. Odavde je nastavak eksponenta 001011, dok se prva dva bita 10 kodiraju u kombinaciji.
- (d) Kombinacija treba da kodira cifru 9 (cifra najveće težine značajnog dela broja) i dva bita najveće težine eksponenta (binarne cifre 10). Kako je 9 velika cifra, na osnovu pravila za predstavljanje sledi da je kombinacija = 11101.

Odavde, zapis broja  $+982.5294 \cdot 10^{42}$  je 1 11101 001011 1000 1011 0101 0101 1010

## 2. BID kodiranje

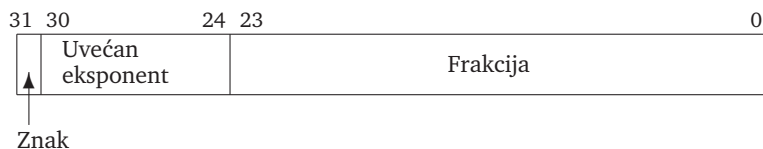
- (a) Broj koga treba zapisati je pozitivan  $\rightarrow$  znak broja je 0.
- (b) Značajan deo broja je 9825294. Njegov prevod u binarni sistem je 100101011110110000001110. Prevod već ima 24 binarne cifre i ne treba da se dopunjava nulama sa leve strane.
- (c) Eksponent je jednak +38, i uz uvećanje 101 dobija se 139. Prevod dobijene vrednosti u binarni brojni sistem je 10001011.
- (d) Kako eksponent ima 24 bita i počinje sa 1001, prva dva bita kombinacije su 11. Pošto je u pitanju konačan broj, eksponent se nalazi u produžetku kombinacije, a na kraju sledi cifra 1 (jer je  $8 + G_{w+4} = (1001)_2$  odakle sledi da je  $G_{w+4} = 1$ ). Dobijena kombinacija je 11100010111. Kod za značajan deo čini preostalih 20 bitova prevoda.

Odavde, zapis broja  $+982.5294 \cdot 10^{42}$  je 0 11100010111 0101110110000001110

## 4 Zapis brojeva u pokretnom zarezu pomoću heksadekadne osnove

Na računarima danas se, pored prethodno opisanih zapisa propisanih IEEE 754 standardom, koristi još i zapis realnih brojeva pomoću osnove 16. Tipičan predstavnik računara koji koriste (ili su koristili) ovakav način zapisa su računari familija S/360, S/370, S/390 i z serija firme IBM. Značajan deo broja se zapisuje u obliku *znak i apsolutna vrednost*; znak značajnog dela broja je istovremeno i znak broja; upisuje se u bit najveće težine i ima vrednost 0 za pozitivne i 1 za negativne brojeve. Značajan deo broja je takođe normalizovan (u obliku  $0.d_0 \dots d_{(p-1)}$ ). Njegova frakcija se zapisuje sa 6 heksadekadnih cifara u 24 bita.

Eksponent se zapisuje u 7 bita na pozicijama 24–30. Vrednosti eksponenta se zapisuju u potpunom komplementu uz uvećanje za 64. Vrednosti eksponenta koje mogu da se zapišu pripadaju intervalu  $[-64, +63]$  odnosno  $[0, 127]$  posle uvećanja za 64. Zapis u registru dužine 32 bita pomoću heksadekadne osnove je prikazan na slici 4. I u ovom slučaju osnova  $\beta=16$  se ne zapisuje u registru. Pravilo je da se nula



Slika 4: Format zapisa realnog broja u jednostrukoj tačnosti pomoću heksadekadne osnove

zapisuje kao sve nule u registru. U tabeli 12 su dati neki primeri zapisa realnih brojeva u 32-bitnoj reči uz upotrebu heksadekadne osnove. Interval kome pripada realan broj  $x$  koji može da se zapiše uz korišćenje 32 bita se određuje na osnovu formata zapisa. Za veličinu eksponenta  $e$  važi

$$-2^6 \leq e \leq 2^6 - 1$$

Za vrednost  $s$  kojom se predstavlja frakcija važi

$$16^{-1} \leq |s| \leq 1 - 16^{-6}$$

	Znak	Eksponent	Frakcija
+15 =	0	1000001	111100000000000000000000
-15 =	1	1000001	111100000000000000000000
+1/64 =	0	0111111	010000000000000000000000
0 =	0	0000000	000000000000000000000000
+16 <sup>-1</sup> × 16 <sup>-64</sup> =	0	0000000	000100000000000000000000
(1 - 16 <sup>-6</sup> ) × 16 <sup>+63</sup> =	0	1111111	111111111111111111111111
+16 <sup>-6</sup> × 16 <sup>-64</sup> =	0	0000000	000000000000000000000001

Provera vrednosti zapisa broja +15:  
 - Znak = +  
 - Eksponent = 1 (=65 - 64)  
 - Frakcija = (F)<sub>16</sub> = F × 16<sup>-1</sup>  
 - Vrednost = Znak frakcija \* 16<sup>eksponent</sup> = +(F × 16<sup>-1</sup>) \* 16<sup>1</sup> = +F<sub>16</sub> = +15<sub>10</sub>

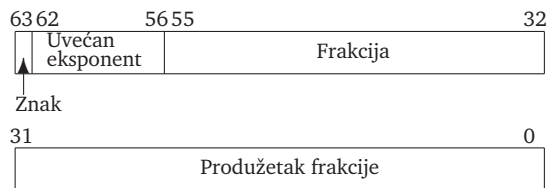
Tabela 12: Zapis realnih brojeva u jednostrukoj tačnosti / heksadekadna osnova

Kombinacijom ovih vrednosti dobija se da je interval brojeva koji mogu da se zapišu

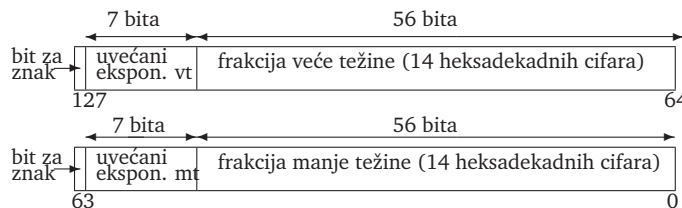
$$16^{-1} * 16^{-64} \leq |x| \leq (1 - 16^{-6}) * 16^{+63}, \quad \text{odnosno} \quad 5.4 * 10^{-79} \lesssim |x| \lesssim 7.2 * 10^{+75}$$

Pokušaj zapisa broja x gde važi  $|x| > (1 - 16^{-6}) * 16^{+63}$  dovodi do pozitivnog ili negativnog prekoračenja, u zavisnosti od znaka broja. Pre pojave računara iz familije z serija pokušaj zapisa broja  $|x| < 16^{-65}$  je dovedio do pozitivnog ili negativnog potkoračenja<sup>10</sup> Sa pojavom z serije uvedeni su subnormalni (tada denormalizovani) brojevi u zapis realnih brojeva sa heksadekadnom osnovom, ali zbog kompatibilnosti programa napisanih na računarima koji su prethodili z seriji (jer su se kao važeći brojevi koristile sve ostale kombinacije bitova) nisu uvedene i vrednosti NaN i beskonačno. Najmanji subnormalni broj (po apsolutnoj vrednosti) koji može da se zapiše u jednostrukoj tačnosti je  $16^{-64} * 16^{-6} = 16^{-70} \approx 5.1 * 10^{-85}$

I u slučaju heksadekadne osnove realan broj u dvostrukoj tačnosti se zapisuje pomoću dva uzastopna registra veličine 32 bita (slika 5). Format zapisa u prvom registru je identičan formatu zapisa u jednostrukoj tačnosti, dok se ceo drugi registar koristi za zapis dodatnih cifara



Slika 5: Format zapisa realnog broja u dvostrukoj tačnosti pomoću heksadekadne osnove



Slika 6: Format zapisa realnog broja u četverostrukoj tačnosti pomoću heksadekadne osnove

frakcije. U ovom slučaju je interval realnih brojeva koji mogu da se predstave u dvostrukoj tačnosti

$$16^{-1} * 16^{-64} \leq |x| \leq (1 - 16^{-14}) * 16^{+63}, \quad \text{odnosno} \quad 5.4 * 10^{-79} \lesssim |x| \lesssim 7.2 * 10^{+75}$$

Najmanji subnormalni broj (po apsolutnoj vrednosti) koji može da se zapiše je  $16^{-64} * 16^{-14} = 16^{-78} \approx 1.2 * 10^{-94}$

Z serija računara je donela i četverostruki format zapisa za realne brojeve sa heksadekadnom osnovom. Za zapis realnih brojeva sa heksadekadnom osnovom u četverostrukoj tačnosti se koristi 16 bajtova. Pri tome veličina prostora za eksponent koji se zapisuje u kodu višak 64 i dalje ostaje ista. Odgovarajuće pozicije za eksponent u drugoj polureči (slika 6) se koriste za zapis manje težine uvećanog eksponenta (praktično eksponenta čija je vrednost umanjena za 14 i koji predstavlja "pravi" eksponent za deo frakcije manje težine koji se nalazi u toj dvostrukoj reči). Interval brojeva koji mogu da se predstave u četverostrukoj tačnosti je

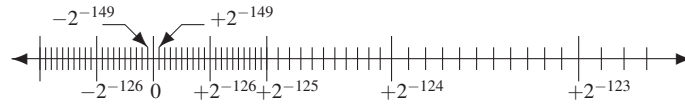
$$16^{-1} * 16^{-64} \leq |x| \leq (1 - 16^{-28}) * 16^{+63}, \quad \text{odnosno} \quad 5.4 * 10^{-79} \lesssim |x| \lesssim 7.2 * 10^{+75}$$

Najmanji subnormalni broj koji može da se zapiše je  $16^{-64} * 16^{-28} = 16^{-92} \approx 1.7 * 10^{-111}$

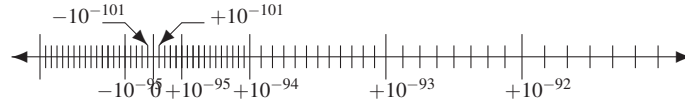
## Poredjenje zapisa realnih brojeva

Bez obzira na način zapisa realnih brojeva, za izabranu tačnost (sa binarnom, dekadnom ili heksadekadnom osnovom) broj različitih zapisa realnih brojeva je isti, ali je gustina na pojedinim delovima brojčane ose različita (slika 7). Ukoliko se poveća interval za eksponent, smanjuje se broj cifara frakcije koje je moguće zapisati i obratno. Štaviše, realnih brojeva koji mogu da se zapišu u jednostrukoj tačnosti ima isto koliko i celih (2<sup>32</sup>) jer toliko ima različitih kombinacija bitova u registru dužine 32 bita. Kao ilustracija mogućih načina zapisa brojeva u pokretnom zarezu koji se koriste u današnjim računarima može da posluži naredna tabela koja sadrži uporedni prikaz (preuzet iz [4]) nekih vrednosti zapisanih u jednostrukoj tačnosti pomoću binarne (IEEE 754), dekadne (IEEE 754, DPD kodiranje) i heksadekadne osnove.

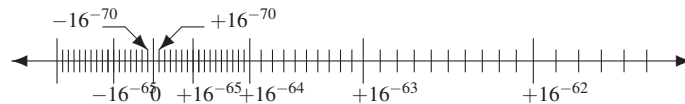
<sup>10</sup>U tom slučaju program nije prekidan, ali je prikazivana poruka o pojavi potkoračenja, i umesto navedene vrednosti u daljim operacijama je uzimana 0.



(a) Zapis pomoću binarne osnove - IEEE 754 format sa binarnom osnovom i uključenim subnormalnim vrednostima



(b) Zapis pomoću binarne osnove - IEEE 754 format sa dekadnom osnovom i uključenim subnormalnim vrednostima



(c) Zapis pomoću heksadekadne osnove sa uključenim subnormalnim vrednostima

Slika 7: Gustina zapisa realnih brojeva (za jednostruku tačnost)

Vrednost		S	BE or C	Ostatak frakcije	
1.0	B	0	011 1111 1	000 0000 0000 0000 0000 0000	[10x10 <sup>-1</sup> ]
	H	0	100 0001	0001 0000 0000 0000 0000 0000	
	D	0	010 0010 0100	0000 0000 0000 0001 0000	
0.5	B	0	011 1111 0	000 0000 0000 0000 0000 0000	[5x10 <sup>-1</sup> ]
	H	0	100 0000	1000 0000 0000 0000 0000 0000	
	D	0	010 0010 0100	0000 0000 0000 0000 0000 0101	
1/64	B	0	011 1100 1	000 0000 0000 0000 0000 0000	[15625x10 <sup>-6</sup> ]
	H	0	011 1111	0100 0000 0000 0000 0000 0000	
	D	0	010 0001 1111	0000 0101 0111 0010 0101	
+0	B	0	000 0000 0	000 0000 0000 0000 0000 0000	[+0x10 <sup>0</sup> ]
	H	0	000 0000	0000 0000 0000 0000 0000 0000	
	D	0	010 0010 0101	0000 0000 0000 0000 0000 0000	
-0	B	1	000 0000 0	000 0000 0000 0000 0000 0000	[-0x10 <sup>0</sup> ]
	H	1	000 0000	0000 0000 0000 0000 0000 0000	
	D	1	010 0010 0101	0000 0000 0000 0000 0000 0000	
-15.0	B	1	100 0001 0	111 0000 0000 0000 0000 0000	[-150x10 <sup>-1</sup> ]
	H	1	100 0001	1111 0000 0000 0000 0000 0000	
	D	1	010 0010 0100	0000 0000 0000 1101 0000	
20/7	B	0	100 0000 0	011 0110 1101 1011 0110 1110	[2857143x10 <sup>-6</sup> ]
	H	0	100 0001	0010 1101 1011 0110 1101 1011	
	D	0	010 1001 1111	1101 0111 0100 1100 0011	
2 <sup>-126</sup>	B	0	000 0000 1	000 0000 0000 0000 0000 0000	[1175494x10 <sup>-44</sup> ]
	H	0	010 0001	0100 0000 0000 0000 0000 0000	
	D	0	000 0111 1001	0011 1101 0110 0101 1010	
2 <sup>-149</sup>	B	0	000 0000 0	000 0000 0000 0000 0000 0001	[1401298x10 <sup>-51</sup> ]
	H	0	001 1011	1000 0000 0000 0000 0000 0000	
	D	0	000 0111 0010	1000 0000 0101 0101 1110	
2 <sup>128</sup> × F F = 1 - 2 <sup>-24</sup>	B	0	111 1111 0	111 1111 1111 1111 1111 1111	[3402823x10 <sup>+32</sup> ]
	H	0	110 0000	1111 1111 1111 1111 1111 1111	
	D	0	100 1100 0101	1000 0000 1001 0010 1101	
2 <sup>-260</sup>	B			Nula (broj je isuviše mali)	[5397605x10 <sup>-85</sup> ]
	H	0	000 0000	0001 0000 0000 0000 0000 0000	
	D	0	001 0101 0000	0111 1110 1111 0000 0101	
2 <sup>248</sup> × F F = 1 - 2 <sup>-24</sup>	B			Ne može da se predstavi	[4523128x10 <sup>+68</sup> ]
	H	0	111 1110	1111 1111 1111 1111 1111 1111	
	D	0	101 0010 1001	1010 1000 1100 1010 1000	

Oznake:

- S - znak broja (frakcije)
- B - Zapis sa binarnom osnovom (IEEE 754)
- D - Zapis sa dekadnom osnovom (IEEE 754, DPD kodiranje)
- H - Zapisa sa heksadekadnom osnovom (IBM z serija)
- BE ili C Uvećani eksponent B ili H broja, kombinacije kod D broja

## Efektivna širina

Rezolucija različitih formata zapisa brojeva u pokretnom zarezu može da bude opisana pomoću rastojanja između dva susedna predstavljiva broja u tom formatu, tj. pomoću vrednosti *ulp*-a. Veličina *ulp*-a je direktno zavisi od preciznosti formata. Pored *ulp*-a, drugi način za opis rezolucije formata je bliskost pojedinačne vrednosti u odnosu na relativno rastojanje što može da se predstavi vrednošću *ulp*-a/vrednost broja. Relativno rastojanje može da se konvertuje u oblik koji se naziva *efektivna širina* koji je vrlo sličan sa preciznošću. Za neku pojedinačnu normalnu vrednost  $x$  u pokretnom zarezu efektivna širina (u oznaci  $W_\beta$  je jednaka  $W_\beta = \log_\beta(x/ulp)$ ). Kada se efektivna širina meri korišćenjem osnove koja se koristi za zapis u pojedinom formatu, važi nejednakost  $p-1 \leq W_\beta \leq p$  gde je  $\beta$  osnova koja se koristi za zapis i  $p$  preciznost formata. Na osnovu prethodne jednakosti i nejednakosti mogu da se izračunaju efektivne širine normalnih brojeva u IEEE 754 binarnom i dekadnom formatu, kao i u zapisu pomoću heksadekadne osnove. Dobijeni rezultati za osnove 2, 10 i 16 (tj. broj tačnih binarnih i dekadnih cifara) su prikazane u tabeli 13.

Dužina /tačnost	Osnova	$W_2$		$W_{10}$	
		Max	Min	Max	Min
32 bita (4 bajta) /jednostruka	H	24.00	20.00	7.22	6.02
	B	24.00	23.00	7.22	6.92
	D	23.25	19.93	7.00	6.00
64 bita (8 bajtova) /dvostruka	H	56.00	52.00	16.86	15.65
	B	53.00	52.00	15.95	15.65
	D	53.15	49.83	16.00	15.00
128 bita (16 bajtova) /četvorstruka	H	112.00	108.00	33.72	32.51
	B	113.00	112.00	34.02	33.72
	D	112.95	109.62	34.00	33.00

Tabela 13: Najveće i najmanje efektivne širine predstavljene u osnovama 2, 10 i 16

## Literatura

- [1] Cowlshaw, M. F. **Densely packed decimal encoding**, *IEEE Proceedings Ő Computers and Digital Techniques*, Vol. 149 (3), pp. 102-104, May 2002
- [2] Cowlshaw M. F.: **General decimal arithmetic** <http://speleotrove.com/decimal/dbcalc.html>
- [3] David Goldberg, **What Every Computer Scientist Should Know About Floating-Point Arithmetic**, *ACM Computing Surveys*, Vol. 23, No. 1 March 1991.
- [4] **z/Architecture: Principles of Operation**, <http://publib.boulder.ibm.com/epubs>, February 2009.
- [5] **IEEE Standard for Binary Floating-Point Arithmetic**. ANSI/IEEE Std. 754-1985, reprint iz *ACM SIGPLAN Notices*, Vol. 22, No. 2, 1987.
- [6] **IEEE-2008 Standard for Floating-Point Arithmetic**. ANSI/IEEE Std. 754-2008, *IEEE*, New York, 2008
- [7] J.F. Reiser, and D.E. Knuth, **Evading the drift in floating point addition**, *Information Processing Letters*, Vol. 3, No. 3, 1975.
- [8] J. Savard: **Floating-Point Formats** <http://www.quadibloc.com/comp/cp0201.htm>
- [9] **Numerical Computation Guide**, [http://cch.loria.fr/documentation/IEEE754/numerical\\_comp\\_guide/index.html](http://cch.loria.fr/documentation/IEEE754/numerical_comp_guide/index.html)