Lucrarea 2

Reprezentarea Numerelor în Virgulă Mobilă

Lucrarea întroduce conceptul de reprezentare a numerelor în virgulă flotantă. Apoi, sunt prezentate două exemple semnificative de format în virgulă flotantă: IEEE 754 şi IBM S30/370.

1 Reprezentarea Numerelor în Virgulă Mobilă

Există multe aplicaţii care necesită acoperirea unui domeniu valoric mare şi care, în general, nu poate fi acoperit prin reprezentarea în virgulă fixă. În aceste cazuri se apelează la cel de-al doilea tip de reprezentare, corespunzător aşa numitei notaţii
tiin ific (nu este o nota ie pozi ional), în conformitate cu care un num r este reprezentat prin 3 numere în forma:

$$N = M \times B^E \qquad (2.1)$$

unde E este exponentul, B este baza, iar M este mantisa.

Mantisa M este reprezentată în semn-mărime sau în complement de 2, iar exponentul E este reprezentat în semn-mărime sau în cod exces (pentru a permite reprezentarea numerelor negative ca întregi fără semn). Baza B este uzual 2, sau o putere a lui 2, dar – foarte rar – poate fi şi 10.

Se spune despre bază că este „built-into-the-circuit“ deoarece ea nu este reprezentată în mod explicit.

Reprezentarea în virgulă flotantă se confruntă cu două probleme principale:

1. Problema mantisei: în conformitate cu care, un număr poate fi reprezentat în mod redundant. De exemplu, următoarele două reprezentări corespund aceluiaşi număr :

$$N = 0.3 \times 10^5$$
$$N = 30 \times 10^3 \qquad (2.2)$$

În vederea efectuării unor operații în virgulă flotantă şi în vederea obținerii rezultatelor în virgulă flotant se apelează la o formă normală de reprezentare a numerelor sau, altfel spus, se apelează la operația de normalizare a numerelor în felul următor:

- dacă numărul de virgulă flotantă este reprezentat în semn-mărime, atunci se consideră un număr normalizat dacă prima cifră binară normală succesivă virgulei este '1' şi nu '0', adică se elimină zerourile nesemnificative (leading 0's);

- dacă numărul în virgulă flotantă este reprezentat în complement de 2 atunci se spune că este normalizat dacă bitul de semn şi bitul cel mai semnificativ al mantisei) diferă valoric;

- operația de normalizare implică deplasări ale vigulei la dreapta, cu incrementarea exponentului, şi la stânga, cu decrementarea exponentului;

- prin normalizare, mantisa obține valori în câmpul restrâns $\frac{1}{2} \le |M| < 1$;

2. Problema exponentului: este legată de problema lui 0, adică $0 \times B^E$. Datorită unei erori de rotunjire sau a unei scăderi între numere de valoare foarte apropiată, mantisa în loc să fie 0 este un număr foarte mic $\neq 0$. Este posibil ca acest număr foarte mic să nu poată fi reprezentat cu ajutorul formatului de virgulă flotantă, decât dacă se permite operația de

denormalizare a mantisei. Semnalizarea cazurilor speciale, cum este denormalizarea, revendică coduri care în mod normal ar fi reprezentat valori ale exponentului. În principiu, formatele de virgulă flotantă care iau în considerație situațiile de excepție folosesc o reprezentare ca întreg în exces a exponentului, cu valorile extreme folosite pentru a semnaliza situațiile speciale.

Dacă avem k biți alocați exponentului, din care un bit este atribuit semnului, atunci rezultă că în câmpul de exponent pot fi reprezentate $2k$ numere: în semn-mărime în intervalul $\left[-2^{k-1}+1; 2^{k-1}-1\right]$, iar în complement de 2 în intervalul $\left[-2^{k-1}; 2^{k-1}-1\right]$, diferența fiind datorată anomaliei complementului de 2 (cu o singură reprezentare pentru „0“). În calculator apare necesitatea testării lui 0 care este inclus ca microoperație în mai multe instrucții. Pentru a uniformiza aceste testări pentru numere întregi şi pentru numere în virgulă flotantă, se cere ca pattern-ul (modelul) de 0-uri să fie constituit dintr-un şir de biți de '0', dar, din precizarea anterioară rezultă că la exponent ar trebui s ă ne folosim de o reprezentare în excess de $2^{k-1}-1$ sau 2^{k-1} .

2 Standardul IEEE 754 pentru Reprezentarea Numerelor în Virgulă Flotantă

La începutul anilor '80 fabricanții de calculatoare uzitau de modalități diferite de reprezentare ale numerelor în virgulă flotantă, adică fiecare avea formatul propriu. Difereau numărul de biți pentru reprezentarea numărului şi al mantisei, difereau modalitățile de tratare ale underflow-ului şi overflow-ului, precum şi a condițiilor speciale (împărțirea la 0, extragerea de rădăcină pătratică dintr-un număr negativ, etc.).

Toate acestea făceau portabilitatea software-ului să fie afectată în mod semnificativ. Pentru a corecta această situație, la sfârşitul deceniului opt, IEEE a organizat un comitet pentru standardizarea aritmeticii în virgulă mobilă. Scopul urmărit era acela de a permite schimbul de date în vigul mobilă între calculatoare diferite şi de a furniza proiectanților hardware un model considerat corect. Rezultatele activității au condus la Standardul IEEE 754(IEEE,1985) propus de profesorul de matematică William Kahan, de la Berkley. Standardul

defineşte trei formate: precizie simplă (32 de biţi), precizie dublă (64 de biţi) şi precizie extinsă (80 de biţi).

$$N = (-1)^S \times 2^{E-127} \times (1.M) \qquad (2.3)$$

unde 0<E<255. Exponentul are alocaţi 8 biţi. În realitate E este un număr cu semn între -127 şi +128, pentru că folosim reprezentarea în exces de 127. Întrucât mantisa este reprezentată în semn-mărime şi este un număr normalizat (adicăprima cifră de '1' apare după virgulă), ştim că primul bit după virgulă este '1'. S-a trecut la deplasarea spre stânga a numărului normalizat a vechii mantise, iar '1' nu se mai reprezintă explicit, ci este „built-into-the-circuit", deci este implicit. Ca efect, se măreşte precizia de reprezentare a numărului în virgulă flotantă. Mantisa (significantul) are câmpul valoric nu $\frac{1}{2} \le |M| < 1$ ci $1 \le |M| < 2$.

În formatul pe 64 de biţi în câmpul de exponent avem 11 biţi iar restul de 52 de biţi sunt pentru mantisă.

$$N = (-1)^S \times 2^{E-1023} \times (1.M) \qquad (2.4)$$

unde 0<E<2047. Acest standard are în plus următoarele convenţii de reprezentare a unor condiţii speciale:

1. ***E*=255(2047) şi *M* ≠0** corespunde a a numitului format „NaN-not a number" şi prin acest format se semnalează procesorului apariţia pe parcursul calculelor a unor situaţii speciale de tipul împărţirii la zero sau extragere de rădăcină pătratică dintr-un număr negativ, etc. În acest caz se intră într-un „trap" prin care se testează modul în care apare această situaţie.

2. *E*=255(2047) şi *M*=0 corespunde formatului care indică situaţia de overflow $\pm\infty$.

3. *E*=0 şi *M* ≠0, acest format corespunde situaţiei de denormalizare.

$$N = (-1)^S \times 2^{-126} (0.M) \qquad (2.5)$$

4. *E*=0 şi *M*=0, este reprezentarea lui ‘0’ în virgulă flotantă IEEE 754: $N = (-1)^S \times 0$

3 Standardul IBM S/360, S/370

Standardul prezintă trei formate: precizie simplă (32 de biți), precizie dublă (64 de biți) şi precizie extinsă (80 de biți).

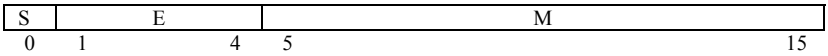
$$N = (-1)^S \times 16^{E-64}(0.M) \qquad (2.6)$$

În câmpul exponentului se reprezintă numere întregi în exces de 64 şi pentru toate versiuniile este format din 7 biți. Mantisa este un număr fracționar reprezentat în semn-mărime nefolosindu-se soluția bit ascuns („hidden“ bit - (0.M)). Baza nu mai este 2 ci 16 iar normalizarea se face în consecință, fiecare incrementare sau decrementare a exponentului mutând virgula la stânga sau la dreapta cu 4 poziții binare.

În acest standard nu avem posibilitatea de a reprezenta situații speciale de tipul NaN-not a number, nici de overflow, nici de denormalizare, iar zero este reprezentat cu zero peste tot. Datorită bazei 16 câmpul valoric de reprezentare a numerelor este mult mai mare: $5.4 \times 10^{-79} \div 7.24 \times 10^{75}$.

4 Aplicații

Problema 2.1 Se dă un format de numere in VF, reprezentarea se face conform standardului IBM S360/370, iar baza este 8.



a) Să se precizeze domeniul valoric acoperit.
b) Reprezentarea în hexa conform formatului pentru $N = -77{,}25_{(10)}$
c) Ce număr în baza 10 reprezintă E7C6.

Soluție:

a) IBM $\rightarrow N = (-1)^S \cdot 16^{E-2^{7-1}}(0.M) = (-1)^S \cdot 16^{E-64}(0.M)$, E fiind pe 7 biti

$\Rightarrow$ formatul impus $\rightarrow N = (-1)^S \cdot 8^{E-2^{4-1}}(0.M) = (-1)^S \cdot 8^{E-8}(0.M)$, E fiind pe 4 biți $\Rightarrow N_{\max} = 8^{15-8} \cdot (0.11...1) = 8^7(1-2^{-11}) = 2^{21}(1-2^{-11}) \cdot 2^{21} \Rightarrow$

$\Rightarrow$ Intervalul $[-N_{\max}, N_{\max}]$

b) $N = -77.25 \begin{cases} 77 = 64+8+4+1 \rightarrow 1001101 \\ 0.25 = \dfrac{1}{2^2} = 0.01 \end{cases}$

$\Downarrow$

$N = -1001101.01 = -1.00110101 \cdot 2^6 = 0.00100110101 \cdot 2^9 = 0.00100110101 \cdot 8^3$

$\Rightarrow \quad E - 8 = 3 \Rightarrow E = 11$

$\Downarrow$

$N =$ | 1 | 1 0 1 1 | 0 0 1 0 0 1 1 0 1 0 1 |

$S \quad E \quad M$

$\Downarrow$

$N_{(16)} = D935$

c)

$N =$ | 1 | 1 1 0 0 | 1 1 1 1 1 0 0 0 1 1 0 |

$S \quad E \quad M$

$E = 12$

$M = 11111000110$

$\Downarrow$

$N = -8^4(0.11111000110) = -2^{12}(0.11111000110) =$

$= 1111100011 \cdot 2^2 = (2^{10} - 1 - 2^2 - 2^3 - 2^4) \cdot 2^2 =$

$= (1024 - 1 - 4 - 8 - 16) \cdot 4 = 995 \cdot 4 = 3980$

Problema 2.2 Se dă numărul 1915.40625. Se cere reprezentarea sa în standardele IEEE 754 şi IBM S360/370.

Soluție:
Pentru a reprezenta numărul în standardele impuse este necesară mai întâi conversia acestuia în binar:
$N = 1915.40625 = N_1 + N_2$, unde
$N_1 = 1915$
$N_2 = 0.40625$

Partea întreagă (N_1) se reprezintă în binar cu ajutorul împărțirii repetate la baza (2), reținând în mod sistematic resturile obținute. Numărul în binar reprezintă concatenarea resturilor parțiale generale, în ordinea inversă obținerii lor (Condiția de terminare a algoritmului este obținerea unui deîmpărțit parțial mai mic decât baza 2).

1915	2																				
1914	957	2																			
1	956	478	2																		
	1	478	239	2																	
		0	238	119	2																
			1	118	59	2															
				1	58	29	2														
					1	28	14	2													
						1	14	7	2												
							0	6	3	2											
								1	2	**1**											
									1												

$\Rightarrow N_{1_{(2)}} = 11101111011$

Conversia lui N_2 se obține prin înmulțirea repetată cu baza (2), reținând în mod sistematic partea întreagă a rezultatelor parțiale obținute, în ordinea obținerii acestora (Condiția de terminare a algoritmului este obținerea unui rezultat fără parte întreagă (sau un număr suficient de iterații)).

$N_{2_{(10)}} =$

$0.40625 \bullet 2 = \mathbf{0}.81250$

$0.81250 \bullet 2 = \mathbf{1}.6250$

$0.625 \bullet 2 = \mathbf{1}.250$

$0.25 \bullet 2 = \mathbf{0}.5$

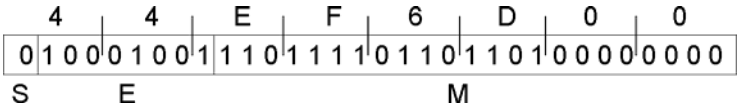
$0.5 \bullet 2 = \mathbf{1}.0$

$\Rightarrow N_{2_{(2)}} = 0.01101$

$\Rightarrow N_{(2)} = 11101111011.01101 = 1.1101111011011 01 \cdot 2^{10}$

Ținând cont de forma unui număr în IEEE 754 (normalizată)
$N = (-1)^S \cdot 2^{E-127} \cdot (1.M)$
rezultă:
$E = 127 + 10 = 137$
$M = 1101111011011010...0$

Reprezentarea conform IEEE 754 rezultă:



Cei 32 de biți ai reprezentării pot fi considerați într-o formă compactă ca şi un cod hex pe 8 simboluri:

$N = 44EF6D00$

Standardul IBM S360/370 codifică un număr sub formatul general

$N = (-1)^S \cdot 16^{E-64} (0.M)$

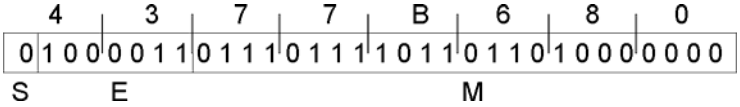
Deoarece baza reprezentării este $16 = 2^4$, mutarea virgulei pentru a obține pentru *N* formatul IBM se va face în grupuri de câte 4 cifre binare.

$N = 11101111011.01101 = 0.011101111011011 01 \cdot 16^3$
$\Downarrow$

$E = 64 + 3 = 67$

$M = 011101111011011010...0$

Reprezentarea conform IBM S360/370 rezultă:



Codul hex care rezultă pentru reprezentare este:

$N = 4377B680$

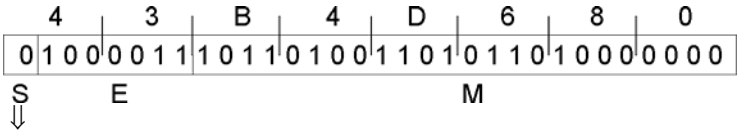
Problema 2.3 Se consideră codul de 32 de biți sub forma compactă 43B4D680, reprezentând un număr în standard IBM S360/370. Care este valoarea numărului în zecimal?

Soluție:

Desfășurarea codului pe 32 de biți oferă informații asupra semnului, mantisei și exponentului numărului, ținând cont de formula generală

$N = (-1)^S \cdot 16^{E-64}(0.M)$

Codul desfășurat este:



$\Downarrow$

$S = 0 \Rightarrow$ numarul este pozitiv

$E = 67, M = 10110100110101101000000$

$\Downarrow$

$N = 16^{67-64} \cdot (0.10110100110101101) = 101101001101.01101$

În zecimal, numărul este:

$$N = 1\cdot 2^{11} + 0\cdot 2^{10} + 1\cdot 2^{9} + 1\cdot 2^{8} + 0\cdot 2^{7} + 1\cdot 2^{6} + 0\cdot 2^{5} + 0\cdot 2^{4} +$$
$$+1\cdot 2^{3} + 1\cdot 2^{2} + 0\cdot 2^{1} + 1\cdot 2^{0} + 0\cdot 2^{-1} + 1\cdot 2^{-2} + 1\cdot 2^{-3} + 0\cdot 2^{-4} + 1\cdot 2^{-5}$$
$$N = 2048 + 512 + 256 + 64 + 8 + 4 + 1 + 0.25 + 0.125 + 0.03125$$
$$N = 2893.40625$$

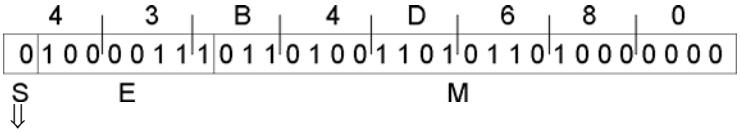
Problema 2.4 Se consideră codul de 32 de biți sub forma compactă 43B4D680, reprezentând un număr în standard IEEE 754. Care este valoarea numărului în zecimal?

Soluție:

Desfășurarea codului pe 32 de biți oferă informații asupra semnului, mantisei şi exponentului numărului, ținând cont de formula generală

$$N = (-1)^{S} \cdot 2^{E-127} \cdot (1.M)$$

Codul desfăşurat este:



$S = 0 \Rightarrow$ numarul este pozitiv

$E = 135,\ M = 01101001101011010000000$

$\Downarrow$

$$N = 2^{135-127} \cdot (1.01101001101011010000000) = 101101001.10101101$$

În zecimal, numărul este:

$$N = 1\cdot 2^{8} + 0\cdot 2^{7} + 1\cdot 2^{6} + 1\cdot 2^{5} + 0\cdot 2^{4} + 1\cdot 2^{3} + 0\cdot 2^{2} + 0\cdot 2^{1} +$$
$$+1\cdot 2^{0} + 1\cdot 2^{-1} + 0\cdot 2^{-2} + 1\cdot 2^{-3} + 0\cdot 2^{-4} + 1\cdot 2^{-5} + 1\cdot 2^{-6} + 0\cdot 2^{-7} + 1\cdot 2^{-8}$$
$$N = 256 + 64 + 32 + 8 + 1 + 0.5 + 0.125 + 0.03125 + 0.015625 + 0.00390625$$
$$N = 361.67578125$$

Problema 2.5 Seconsideră următorul format ipotetic,de la stânga la dreapta , inspirat din formatul de virgulă flotantă IBM S360/370: semnul

pe un bit, urmat de câmpul de exponent pe 4 biți, urmat de mantisă, până la completarea celor 16 biți ai formatului. Știind că baza este 8, normalizarea se face în manieră IBM S360/370, iar reprezentarea exponentului respectă, de asemenea, principiul formatului IBM, se cer următoarele:

a) Să se reprezinte numărul -99.25_{10} în acest format ipotetic.
b) Care este numărul zecimal, care în formatul ipotetic este reprezentat ca $53AC_{hex}$?
c) Care este domeniul (plaja) de valori acoperit prin acest format ipotetic?

Răspuns:

a) $-99.25_{10} = -0001100011.01_2 = -0.00110001101 \times 8^3$

Prin analogie cu formatul IBM S360/370, excesul folosit la reprezentarea exponentului este 2^3, adică 8. Prin urmare, în câmpul de exponent vom reprezenta 11_{10}.

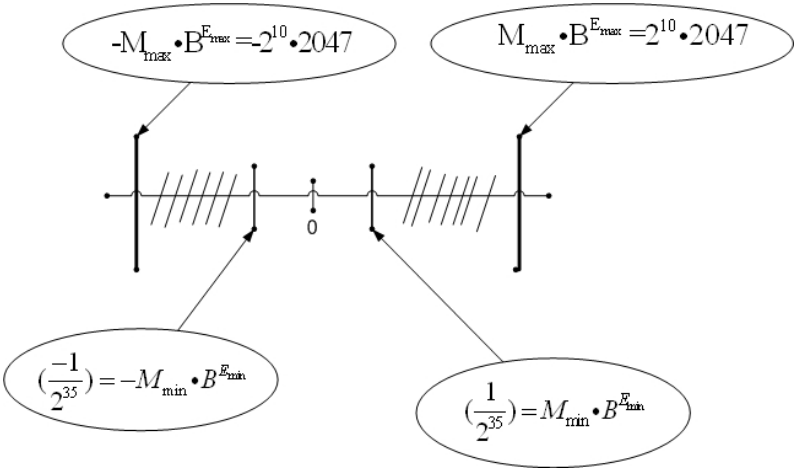
Numărul nostru va fi reprezentat, în formatul dat ca şir binar şi şir hexa astfel:

$1101100110001101 = D98D_{hex}$

b) $53AC_{hex} = 0101001110101100_2$

Numărul este pozitiv (S=0), iar exponentul este 2 în exces de 8 ($E=1010_2$). Prin urmare, numărul este $+11101.011_2 = +29.375_{10}$.

c) Exponentul minim $= E_{min} = 0000_2 = -8_{10}$
Exponentul maxim $= E_{max} = 1111_2 = +7_{10}$
Mantisa minimă nenulă $= M_{min} = 0.00000000001_2 = +2^{-11}$
Mantisa maximă $= M_{max} = 0.11111111111_2 = 1\text{-}2^{-11} = 2^{10} \times 2047$.
Plaja de valori este prezentată în figura de mai jos:



Problema 2.6 Seconsideră următorul format ipotetic,de la stânga la dreapta , inspirat din formatul de virgulă flotantă IEEE 754: semnul pe un bit, urmat de câmpul de exponent pe 5 biți, urmat de mantisă, până la completarea celor 16 biți ai formatului. Normalizarea şi rprezentarea exponentului se fac în maniera formatului de virgulă flotantă IEEE 754, inclusiv prevederea situațiilor de excepție. Se cer următoarele:

a) Să se reprezinte numărul $M=-281.75_{10}$ în acest format ipotetic.
b) Care este numărul zecimal, care în formatul ipotetic este reprezentat ca $D59B_{hex}$?
c) Care este domeniul (plaja) de valori acoperit prin acest format ipotetic?

Răspuns:

a) $M= -281.75_{10} = -100011001.11_2 = -1.0001100111 \times 2^8$

Prin analogie cu formatul IEEE 754, excesul folosit la reprezentarea exponentului este 2^4-1, adică 15. Prin urmare, în câmpul de exponent vom reprezenta 23_{10}.

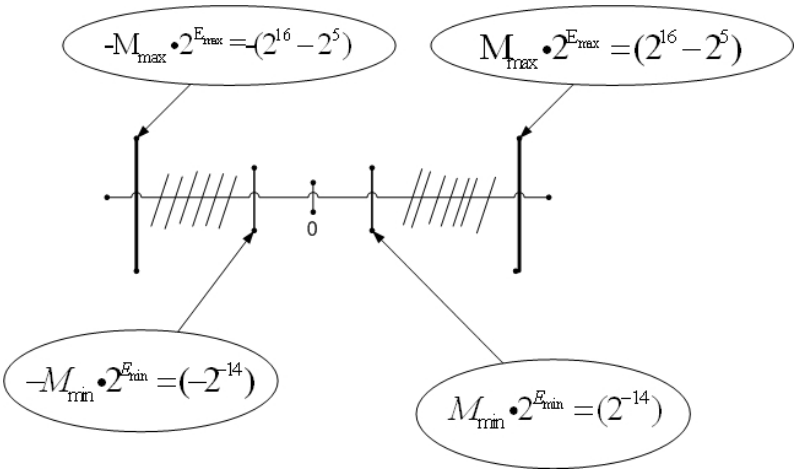
Numărul nostru va fi reprezentat, în formatul dat ca şir binar şi şir hexa astfel:

$1101110001100111 = DC67_{hex}$

b) $D59B_{hex} = 1101010110011011_2$

Numărul este negativ (S=1), iar exponentul este 6 în exces de 15 ($E=10101_2$). Prin urmare, numărul este $-1011001.1011_2 = -89.6875_{10}$.

c) Exponentul minim $=E_{min} = 00001_2 = -14_{10}$
Exponentul maxim $=E_{max} = 11110_2 = +15_{10}$
Mantisa minimă nenulă $= M_{min} = 1.0000000000_2 = +1_{10}$
Mantisa maximă $= M_{max} = 1.1111111111_2 = 2-2^{-10}$.
Plaja de valori este prezentată în figura de mai jos:



Problema 2.7 (propusă) Să se rezolve problemele precedente pentru un format ipotetic pe 24 de biți la alegerea proiectantului, cu motivarea alegerii şi cu definirea clară a mecanismului de normalizare şi reprezentare a exponentului.