Definition binärer Gleitpunktzahlen

Die Logik für Gleitpunktzahlen wurde mit **b = 2** in die binären Arithmetik übernommen.

Nach dem Standard **IEEE 754** ist eine **binäre Gleitpunktzahl z** mit **Vorzeichen-Bit s**, **binärer Mantisse 1.f** (in Normalform) und **Exponent e** wie folgt definiert:

$$z = (-1)^s \cdot 1.f \cdot 2^e$$

Hierbei steht **s=0** wegen $(-1)^0 = 1$ für **positives** und **s = 1** wegen $(-1)^1 = -1$ für **negatives Vorzeichen** von **z**. Das **Vorzeichen-Bit** bildet das höchstwertige Bit (MSB) der binären Gleitpunktzahl.

Danach folgen die Bits des **Exponenten**, der jedoch nicht direkt als **e**, sondern in Form der sog. **Charakteristik c = e + B** gespeichert wird.

Zum tatsächlichen **Exponenten e** wird also eine **Verschiebung (Bias) B** addiert, die so gewählt ist, dass der Nullpunkt für **e** in die Mitte des zur Verfügung stehenden Wertebereichs **[0, 2B+1]** verschoben wird.

Auf diese Weise können **Exponenten** zwischen **e = -B** (entsprechend **c = 0**) und **e = B + 1** (entsprechend **c = 2B+1**) dargestellt werden.

Anschließend folgen die binären Nachkommastellen **f** der **Mantisse**.

Die **führende 1** muss **nicht** gespeichert werden (verborgene Eins, Hidden Bit), da diese nach Definition konstant ist.

Bei einer kurzen Gleitpunktzahl werden **32 Bit** verwendet, wobei **8 Bit** für die **Charakteristik c= e +127** mit Wertebereich **[0, 255]** und **Bias B=127** zur Verfügung stehen.

Die Addition einer Verschiebung **B** bezeichnet man allgemein als **Excess-Code** und speziell mit **B=127** als **127-Exzess-Code**.

Darstellung des **Exponenten e** und der **Charakteristik c** im **127-Excess-Code**:

$c=(00000000)_2 = (0)_{10}$ **$c=(01111111)_2 = (127)_{10}$** **$c=(11111111)_2 = (255)_{10}$**

e = - 127 **e = 0** **e = 128**

Die Mantisse einer kurzen Gleitpunktzahl mit 23 Bit für die Nachkommastellen lautet:

$$\mathbf{m = 1.f_0 f_1 \ldots . f_{22}}$$

Daraus resultiert eine Genauigkeit von $\mathbf{2^{-24}}$, entspricht 7 signifikanten Dezimalstellen.

Aufbau einer kurzen Gleitpunktzahl nach dem IEEE 754 Standard

s	c c c c c c c c	f f f f f f f f f f f f f f f f f f f f f f f
Vorzeichen der Mantisse	Exponent (8 Bit) im 127-Exzess-Code	← Mantisse in Normalform **ohne führende 1** (23 Bit) →

Eine **lange Gleitpunktzahl** umfasst **64 Bit** :

Bit 0 (MSB): Vorzeichen-Bit, **0** entspricht **positiv** oder Null, **1** entspricht **negativ**

Bit 1 bis 11: **11-Bit** für die **Charakteristik c = e + 1023**

Bit 12 bis 63: **52 Bit** für die **Mantisse** in **Normalform** $\mathbf{m = 1.f_0 f_1 \ldots . f_{53}}$

Genauigkeit: ca. 15 signifikanten Dezimalstellen.

Umwandlung einer Dezimalzahl in eine kurze binäre Gleitpunktzahl:

1. Die Dezimalzahl wird in eine Binärzahl umgewandelt, ggf. mit Nachkommastellen.

2. Das Komma wird so weit nach links oder rechts verschoben, bis die Normalform erreicht ist. Bei Verschiebung um je eine Stelle nach links wird der Exponent e der Basis 2 um eins erhöht, bei Verschiebung nach rechts um eins erniedrigt.

3. Vorzeichen der Zahl (positiv: 0, negativ: 1) wird in das MSB des ersten Byte geschrieben.

4. Zum Exponenten e wird 127 addiert, das Ergebnis wird in binäre Form mit 8 Stellen umgewandelt. Ist der Exponent positiv, so hat das führende Bit den Wert 1, sonst hat es den Wert 0. Das Ergebnis wird im Anschluss an das Vorzeichen-Bit in die letzten 7 Bit des ersten und in das MSB des zweiten Byte eingefügt.

5. In die Bytes 2 (ohne das bereits für den Exponenten verwendete MSB), 3 und 4 werden die Nachkommastellen $f_0 f_1 \ldots f_{22}$ der Mantisse eingefügt.

Beispiel: **148.625** ist in eine binäre Gleitpunktzahl umzuwandeln.

1. Schritt: **148.6**25 **dez = 10010100.101 bin**

2. Schritt: **10010100.101 = 1.0010100101 * 2**^{7}
Normalform erreicht, **Exponent ist 7**

3. Schritt: **Exponent: c = 7 + 127 = 134 dez = 10000110 bin**

4. Schritt: Ergebnis: **01000011 00010100 10100000 00000000 bin = 43 94 A0 00 hex**

Byte 1	Byte 2	Byte 3	Byte 4

Das führende Bit von Byte 1 enthält das positive Vorzeichen **s** der Mantisse (MSB=0). Es folgen die zur Verdeutlichung blau gedruckten 8 Bit für den Exponenten. Die Bytes **2** (ohne MSB), **3** und **4** bilden die **Nachkommastellen der Mantisse**; die **führende 1 fällt weg**, da diese wegen der Normalformdarstellung **redundant** ist.

Gleitpunktzahlen sind **nicht gleichmäßig verteilt**. Der Abstand zwischen je zwei benachbarten Gleitpunktzahlen wird mit steigendem Betrag immer größer.

Besonderheiten und Sonderfälle

Eine nur aus 32 Nullen bestehende kurze Gleitpunktzahl hätte den endlichen Wert $\mathbf{z_{min} = 2^{-127}}$. Eine exakte **0** wäre bisher nicht darstellbar.

Es ist noch zu definieren, ab wann eine Zahl als $+\infty$ anzusehen ist:

Für **c=0**, also **e=-127** wird die Annahme der normalisierten Mantisse **1.f** fallen gelassen und durch denormalisierte Mantissen **0.f** ersetzt.

Die kleinste positive Gleitpunktzahl mit normalisierter Mantisse ist damit $\mathbf{z_{min} = 2^{-126} \approx 1.1755 x 10^{-38}}$. Daran schließen sich die **denormalisierten Gleitpunkt-zahlen** mit $\mathbf{0.f \cdot 2^{-126}}$ an, die den Wertebereich $\mathbf{\pm 2^{-149}}$ bis $\mathbf{\pm(1-2^{-23}) x 2^{-126}}$ umfassen.

Jetzt ergibt sich auch mit **f = 0** der exakte Zahlenwert **z = 0**, wenn **alle 32 Bit 0** sind.

Als ∞ wird die Zahl $\mathbf{1.0 \cdot 2^{128}}$ festgelegt. Die Nachkommastellen der Mantisse sind also alle **0**. Die größte Zahl lautet damit $\mathbf{z_{max} = (2-2^{-23}) 2^{127} \approx 3.4028 x 10^{38}}$.

Zwischen +∞ und -∞ wird durch das **Vorzeichen-Bit** entschieden. ∞ erhält man beispielsweise bei der **Division x/0** mit $|x|>0$.

Zahlen der Art $\mathbf{1.f \cdot 2^{128}}$ mit $f>0$ dienen ohne nähere Spezifizierung in der Norm zur Kennzeichnung unerlaubter Zahlenbereiche (**NaN**, Not a Number). Diese entstehen mit $\mathbf{|x|>0}$ insbesondere bei den Operationen **x / 0**, **x % 0**, **∞ + ∞**, **∞ / ∞** und $\sqrt{-|\mathbf{x}|}$

Rechnen mit Gleitpunktzahlen

Beim Rechnen mit Gleitpunktzahlen sind folgende Rechenregeln zu beachten:

Addition und Subtraktion: Als erstes werden die Exponenten angeglichen, indem die Mantisse des Operanden mit dem kleineren Absolutbetrag entsprechend verschoben wird. Dabei können Stellen verloren gehen, d.h. es entsteht dann ein Rundungs- oder Abbruchfehler.

Anschließend werden die Mantissen addiert bzw. subtrahiert.

Multiplikation: Die Mantissen der Operanden werden multipliziert, die Exponenten werden addiert.

Division: Die Mantissen der Operanden werden dividiert, der neue Exponent ergibt sich als Differenz des Exponenten des Dividenden und des Divisors.

Nach allen Operationen ist zu prüfen, ob die Ergebnisse in der Normalform vorliegen, ggf. ist durch Verschieben wieder zu normalisieren.

Außerdem sind die oben angegebenen **betragsmäßig kleinste Zahl z_{min}** und die **betragsmäßig größte Zahl z_{max}** zu berücksichtigen.

Resultate arithmetischer Gleitpunkt-Operationen sind <u>nicht notwendigerweise</u> wieder Gleitpunktzahlen; sie werden daher zu den nächstgelegenen Gleitpunktzahlen gerundet.

Wird dabei z_{max} überschritten, ergibt sich ein **Überlauf (Overflow)**. Diese Besonderheiten der endlichen Arithmetik bedeuten auch, dass die Ergebnisse arithmetischer Berechnungen von deren **Reihenfolge** abhängen können, Kommutativ-, Assoziativ- und Distributivgesetze **gelten also nicht uneingeschränkt**.