DK 621.382.049.77.037.372.2 DEUTSCHE NORM Januar 1992

Binäre Gleitpunkt-Arithmetik für Mikroprozessor-Systeme

(IEC 559 : 1989)
Deutsche Fassung HD 592 S1 : 1991

DIN IEC 559

Diese Norm enthält die Deutsche Fassung des Europäischen Harmonisierungsdokumentes HD 592 S1

Binary floating-point arithmetic for microprocessor systems (IEC 559 : 1989);
German Version HD 592 S1

Arithmétique binaire en virgule flottante pour systèmes à microprocesseurs (CEI 559 : 1989);
Version allemande HD 592 S1

Diese Norm enthält das Europäische Harmonisierungsdokument HD 592 S1 : 1991. HD 592 S1 : 1991 enthält unverändert die Internationale Norm IEC 559 : 1989.

Nationales Vorwort

Die deutsche Übersetzung für diese Norm hat das zuständige nationale Gemeinschaftsunterkomitee 631.10 „Mikroprozessor-Systeme" der Deutschen Elektrotechnischen Kommission im DIN und VDE (DKE) und des Normenausschusses Informationsverarbeitungssysteme (NI) im DIN angefertigt.

Das Europäische Komitee für Elektrotechnische Normung CENELEC hat IEC 559, 2. Ausgabe 1989, als Harmonisierungsdokument 592 angenommen. Die vorliegende Norm stellt die Übernahme dieses Europäischen Harmonisierungsdokuments in das nationale Normenwerk dar.

Der Entwurf zu dieser Norm war veröffentlicht als DIN IEC 47B(CO)19, Ausgabe 05.89.

Internationale Patentklassifikation

G 06 F 7/54

Fortsetzung 11 Seiten HD-Dokument

Deutsche Elektrotechnische Kommission im DIN und VDE (DKE)
Normenausschuß Informationsverarbeitungssysteme (NI) im DIN Deutsches Institut für Normung e.V.

HARMONISIERUNGSDOKUMENT
HARMONIZATION DOCUMENT
DOCUMENT D'HARMONISATION

HD 592 S1

Mai 1991

DK 621.382.049.77.037.372

Deskriptoren: Binäre Gleitpunkt-Arithmetik, Mikroprozessor-Systeme

Deutsche Fassung

Binäre Gleitpunkt-Arithmetik für Mikroprozessor-Systeme

(IEC 559 : 1989)

Binary floating-point arithmetic for microprocessor systems (IEC 559 : 1989)

Arithmétique binaire en virgule flottante pour systèmes à microprocesseurs (CEI 559 : 1989)

Dieses Harmonisierungsdokument wurde von CENELEC am 1991-03-15 angenommen.

Die CENELEC-Mitglieder sind gehalten, die CEN/CENELEC-Geschäftsordnung zu erfüllen, in der die Bedingungen für die Übernahme dieses Harmonisierungsdokuments auf nationaler Ebene festgelegt sind.

Auf dem letzten Stand befindliche Listen dieser nationalen Übernahmen mit ihren bibliographischen Angaben sind beim Zentralsekretariat oder bei jedem CENELEC-Mitglied auf Anfrage erhältlich.

Dieses Harmonisierungsdokument besteht in drei offiziellen Fassungen (Deutsch, Englisch, Französisch).

CENELEC-Mitglieder sind die nationalen elektrotechnischen Komitees von Belgien, Dänemark, Deutschland, Finnland, Frankreich, Griechenland, Irland, Island, Italien, Luxemburg, Niederlande, Norwegen, Österreich, Portugal, Schweden, Schweiz, Spanien und dem Vereinigten Königreich.

CENELEC

EUROPÄISCHES KOMITEE FÜR ELEKTROTECHNISCHE NORMUNG
European Committee for Electrotechnical Standardization
Comité Européen de Normalisation Electrotechnique

Zentralsekretariat: rue de Stassart 35, B-1050 Brüssel

Vorwort

Das CENELEC-Fragebogenverfahren zur unveränderten Annahme der Internationalen Norm IEC 559 : 1989 ergab, daß für die Annahme als Harmonisierungsdokument keine gemeinsamen CENELEC-Abänderungen notwendig waren.

Das Referenzdokument wurde den CENELEC-Mitgliedern zur formellen Abstimmung vorgelegt und von CENELEC am 15. März 1991 als HD 592 S1 genehmigt.

Nachstehende Daten wurden festgelegt:

– spätestes Datum der Ankündigung des HD auf nationaler Ebene	(doa)	1991-09-01
– spätestes Datum der Veröffentlichung einer harmonisierten nationalen Norm	(dop)	1992-03-01
– spätestes Datum für die Zurückziehung entgegenstehender nationaler Normen	(dow)	1992-03-01

Anerkennungsnotiz

Der Text der Internationalen Norm IEC 559 : 1989 wurde von CENELEC als Harmonisierungsdokument ohne irgendeine Abänderung genehmigt.

Binäre Gleitpunkt-Arithmetik für Mikroprozessor-Systeme

Inhalt

Vorwort

1. Die offiziellen Beschlüsse oder Vereinbarungen der IEC über technische Fragen, die in Technischen Komitees von Vertretern aller an dem behandelten Thema besonders interessierten nationalen Komitees erarbeitet werden, bringen das höchstmögliche Maß internationaler Übereinstimmung für das behandelte Sachgebiet zum Ausdruck.
2. Sie stellen Empfehlungen zur internationalen Anwendung dar und sind als solche von den nationalen Komitees angenommen.
3. Um die internationale Vereinheitlichung zur fördern, wünscht die IEC, daß alle nationalen Komitees den Text der IEC-Empfehlungen so weit in ihre nationalen Regeln übernehmen, wie es die Gegebenheiten im jeweiligen Land gestatten. Jede Abweichung zwischen der IEC-Empfehlung und der entsprechenden nationalen Regel sollte in dieser, soweit möglich, deutlich gekennzeichnet werden.

Einleitung

Diese Norm wurde vom Unterkomitee 47B „Mikroprozessor-Systeme" des Technischen Komitees Nr 47 „Halbleiter-Bauelemente" erarbeitet. (Dieses Unterkomitee wurde von ISO/IEC JTC 1 übernommen.)

Diese zweite Ausgabe der Norm IEC 559 ersetzt die erste Ausgabe aus dem Jahr 1982.

Der Text dieser Norm basiert auf folgenden Schriftstücken:

Sechsmonatsregel	Abstimmungsbericht
47B(CO)19	47B(CO)26

Ausführliche Angaben über die Abstimmung zur Annahme dieser Norm können dem in der obigen Tabelle angegebenen Bericht über das Abstimmungsergebnis entnommen werden.

1 Anwendungsbereich

1.1 Gegenstand

Es wird davon ausgegangen, daß die Realisierung eines Gleitpunkt-Systems nach dieser Norm vollständig in Software, vollständig in Hardware oder in beliebiger Kombination von Software und Hardware erfolgt. Der Programmierer oder Anwender sieht das Gesamtsystem, das die Norm entweder erfüllt oder nicht. Hardware-Teile, die zur Übereinstimmung Software-Unterstützung erfordern, dürfen nicht allein, abgelöst von dieser Software, als normkonform bezeichnet werden.

1.2 Einschließungen

Diese Norm gilt für:

1. Grund- und erweiterte Gleitpunkt-Zahlenformate;
2. Addition, Subtraktion, Multiplikation, Division, Quadratwurzel, Rest- und Vergleichsoperationen;
3. Umwandlungen zwischen ganzen Zahlen und Gleitpunkt-Zahlen;
4. Umwandlungen zwischen verschiedenen Gleitpunkt-Formaten;

5. Umwandlungen zwischen Gleitpunkt-Zahlen in Grundformaten und Dezimaldarstellungen; und
6. Gleitpunkt-Ausnahmen und ihre Behandlung einschließlich NaNs (Nichtzahlen).

1.3 Ausschließungen

Diese Norm gilt nicht für:

1. Formate von Dezimalfolgen und ganzen Zahlen;
2. Interpretation des Vorzeichens und der signifikanten Felder von NaNs, oder
3. Binär ⟷ Dezimal-Umwandlungen zu und von erweiterten Formaten.

2 Begriffe

Transformierter Exponent (Biased exponent)
Die Summe von Exponent und einer Versatzkonstanten (Bias), die so gewählt ist, daß der transformierte Exponent nicht negativ wird.

Binäre Gleitpunkt-Zahl
Eine Bitfolge ist durch drei Teile gekennzeichnet: einem Vorzeichen, einem vorzeichenbehafteten Exponenten und einer Mantisse. Ihr numerischer Wert, wenn vorhanden, ist das mit Vorzeichen versehene Produkt ihrer Mantisse mit der dem Exponenten potenzierten Basis 2.

In dieser Norm wird eine Bitfolge nicht immer von der Zahl, die sie darstellen soll, unterschieden.

Unnormalisierte Zahl
Eine Gleitpunkt-Zahl ungleich Null, deren Exponent einen reservierten Wert hat, üblicherweise das Minimum des Formats, und deren explizites oder implizites Führungsbit Null.

Ziel
Der Ort für das Ergebnis einer binären oder unären Operation. Das Ziel kann entweder explizit durch den Anwender bestimmt oder implizit durch das System geliefert werden (z. B.: Zwischenergebnisse in Unterausdrücken oder Argumente für Prozeduren). Einige Sprachen plazieren die Ergebnisse von Zwischenrechnungen in Ziele außerhalb des Anwenderzugriffes. Dessen ungeachtet definiert diese Norm, abhängig von den Werten der Operanden, das Ergebnis einer Operation bezüglich des Formates im Ziel.

Exponent
Der Teil einer binären Gleitpunkt-Zahl, der üblicherweise die ganzzahlige Potenz zur Basis 2 darstellt, die benutzt wird, um den Wert der Gleitpunkt-Zahl festzulegen. Gelegentlich wird der Exponent „vorzeichenbehafteter" oder „nichttransformierter" (unbiased) Exponent genannt.

Bruch
Der Teil der Mantisse, der sich rechts vom impliziten Binärpunkt befindet.

Modus
Eine Variable, die ein Anwender zur Steuerung von nachfolgenden arithmetischen Operationen Setzen, Abfragen, Sichern und Wiederherstellen kann. Der voreingestellte Modus (Default Mode) ist der Modus, den ein Programm vorgeben kann und der wirksam ist, solange nicht explizit eine entgegengesetzte Festlegung entweder im Programm oder seiner Spezifikation getroffen wird.

Der folgende Modus muß implementiert sein:

1. Rundung – zur Steuerung der Richtung von Rundungsfehlern;
   und in bestimmten Implementierungen
2. Rundungsgenauigkeit – um die Genauigkeit von Ergebnissen zu verringern.
   Der Ersteller kann eigenständig den folgenden Modus zusätzlich realisieren:
3. Traps, gesperrt/freigegeben – zur Behandlung von Ausnahmefällen.

Nichtzahl (NaN)
Eine im Gleitpunkt-Format kodierte symbolische Größe. Es gibt 2 Arten von NaNs (siehe Abschnitt 6.2). Anzeigende NaNs führen zum Ausnahmefall „UNZULÄSSIGE OPERATION" (siehe Abschnitt 7.1), wenn sie als Operanden auftreten. Nichtanzeigende NaNs laufen durch fast alle Rechenoperationen, ohne Ausnahmefälle herbeizuführen.

Ergebnis
Die Bitfolge (die gewöhnlich eine Zahl darstellt), die in dem Ziel abgelegt wird.

Mantisse
Der Teil einer binären Gleitpunkt-Zahl, die aus einem expliziten oder impliziten Anfangsbit links von ihrem impliziten Binärpunkt und einem Feld für den Bruch rechts vom Binärpunkt besteht.

Muß
Das Wort „muß" bezeichnet das, was in jeder normkonformen Implementierung obligatorisch vorgeschrieben ist.

Sollte
Das Wort „sollte" bezeichnet das, was nachdrücklich empfohlen wird, um die Vorgaben der Norm einzuhalten. Architektonische oder andere Sachzwänge außerhalb des Anwendungsbereichs dieser Norm können gelegentlich die Anwendung dieser Festlegungen unmöglich machen.

Status Flag
Eine Variable, die die Zustände Gesetzt und Gelöscht annehmen kann. Ein Anwender kann ein Flag löschen, es kopieren oder seinen früheren Zustand wiederherstellen. Wenn es gesetzt ist, kann ein Flag zusätzliche, systemabhängige, möglicherweise für einige Anwender nicht zugängliche Informationen enthalten.

Die Operationen nach dieser Norm können als Nebeneffekt einige der folgenden Flags setzen: UNGENAUES ERGEBNIS, UNTERLAUF, ÜBERLAUF, DIVISION DURCH NULL und UNZULÄSSIGE OPERATION.

Anwender
Im Sinne dieser Norm versteht man darunter jede Person, jede Hardware und jedes Programm außerhalb des Definitionsbereiches dieser Norm mit Zugang zu den Operationen der Programmierumgebung, die Bestandteil dieser Norm sind oder diese Operationen steuern.

3 Formate

Diese Norm definiert vier Gleitpunkt-Formate in zwei Gruppen: Grundformate und erweiterte Formate. In jeder Gruppe gibt es ein Format einfacher Länge und doppelter Länge. Gebräuchliche Implementierungen unterscheiden sich durch die unterstützten Formate.

3.1 Wertemengen

Dieser Abschnitt betrifft nur die numerischen Werte, die innerhalb eines Formats darstellbar sind, nicht die Kodierungen, die Gegenstand der folgenden Abschnitte sind. In einem gewählten Format sind genau diejenigen Werte darstellbar, die durch die folgenden drei ganzzahligen Parameter bestimmt sind:

p die Anzahl der signifikanten Bits (Genauigkeit)
E_{max} der maximale Exponent, und
E_{min} der minimale Exponent

Die Parameter jedes Formats sind in Tabelle 1 dargestellt. Innerhalb jedes Formats müssen genau die folgenden Größen darstellbar sein:

Tabelle 1. **Zusammenfassung der Format-Parameter**

Parameter	Format			
	einfach lang	erweitert einfach lang	doppelt lang	erweitert doppelt lang
p	24	≥ 32	53	≥ 64
E_{max}	+ 127	$\geq +1023$	+ 1023	$\geq +16383$
E_{min}	− 126	≤ -1022	− 1022	≤ -16382
Versatzkonstante (Bias)	+ 127	nicht spezifiziert	+ 1023	nicht spezifiziert
Exponentenbreite in Bits	8	≥ 11	11	≥ 15
Formatbreite in Bits	32	≥ 43	64	≥ 79

- Zahlen der Form $(-1)^s \cdot 2^E (b_0 b_1 b_2 \ldots b_{p-1})$, wobei
  s gleich 0 oder 1
  E jede ganze Zahl zwischen E_{min} und E_{max} einschließlich und wobei jedes b_i gleich 0 oder 1 ist;
- Zwei Unendlichkeiten, $+\infty$ und $-\infty$;
- mindestens eine anzeigende NaN und
- mindestens eine nichtanzeigende NaN.

Die vorhergehende Beschreibung läßt redundante Werte zu, z. B.:

$$2^0(1.0) = 2^1(0.1) = 2^2(0.01) = \ldots$$

Die Kodierungen solcher Werte ungleich Null dürfen jedoch nur in erweiterten Formaten (siehe Abschnitt 3.3) redundant sein. Die Werte ungleich Null der Form $\pm 2^{E_{min}} (0.b_1 b_2 \ldots b_{p-1})$ werden als „unnormalisiert" bezeichnet. Reservierte Exponenten können zur Kodierung von NaNs, $\pm\infty$, ± 0 und unnormalisierten Zahlen benutzt werden. Für jede Variable, die den Wert Null hat, liefert das Vorzeichen s ein zusätzliches Informationsbit. Obwohl alle Formate festgelegte Darstellungen für +0 und −0 haben, sind die Vorzeichen in einigen Fällen von Bedeutung, wie bei Division durch Null, und in anderen nicht. In dieser Norm werden 0 und ∞ ohne Vorzeichen geschrieben, wenn das Vorzeichen keine Rolle spielt.

3.2 Grundformate

Zahlen in den einfach langen und doppelt langen Formaten setzen sich aus drei Feldern zusammen:

Einem 1-Bit-Vorzeichen s

Einem transformierten Exponenten $e = E + \text{Bias}$, und

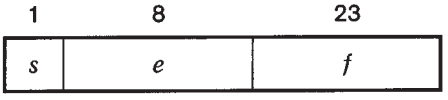
Einem Bruch $f = .b_1 b_2 \ldots b_{p-1}$.

Der Bereich des nicht transformierten Exponenten E muß jede Ganze Zahl zwischen zwei Werten E_{min} und E_{max} einschließlich beinhalten. Weiterhin müssen zwei andere feste Werte beinhaltet sein: $E_{min} - 1$, um ± 0 und unnormalisierte Zahlen zu kodieren, und $E_{max} + 1$, um $\pm\infty$ und NaNs zu kodieren. Die oben genannten Parameter sind in Tabelle 1 angegeben. Jeder numerische Wert ungleich Null hat genau eine Kodierung. Die Felder werden in den folgenden Abschnitten beschrieben.

3.2.1 Einfach langes Format

Eine 32-Bit-Zahl X im einfach langen Format wird aufgeteilt wie in Bild 1 gezeigt. Der Wert v von X ergibt sich dabei aus seinen konstituenten Feldern wie folgt:

1. Wenn $e = 255$ und $f \neq 0$, dann ist $v =$ NaN unabhängig von s.
2. Wenn $e = 255$ und $f = 0$, dann ist $v = (-1)^s \cdot \infty$.
3. Wenn $0 < e < 255$, dann ist $v = (-1)^s \cdot 2^{e-127}(1.f)$.
4. Wenn $e = 0$ und $f \neq 0$, dann ist $v = (-1)^s \cdot 2^{-126}(0.f)$ (unnormalisierte Zahlen).
5. Wenn $e = 0$ und $f = 0$, dann ist $v = (-1)^s \cdot 0$ (Null).



„msb" bedeutet „Bit höchster Wertigkeit"
„lsb" bedeutet „Bit niedrigster Wertigkeit"

Bild 1. Einfach langes Format

3.2.2 Doppelt langes Format

Eine 64-Bit-Doppel-Format-Zahl X wird aufgeteilt wie in Bild 2 gezeigt. Der Wert v von X ergibt sich dabei aus seinen konstituenten Feldern wie folgt:

1. Wenn $e = 2047$ und $f \neq 0$, dann ist $v =$ NaN unabhängig von s.
2. Wenn $e = 2047$ und $f = 0$, dann ist $v = (-1)^s \cdot \infty$.
3. Wenn $0 < e < 2047$, dann ist $v = (-1)^s \cdot 2^{e-1023}(1.f)$.
4. Wenn $e = 0$ und $f \neq 0$, dann ist $v = (-1)^s \cdot 2^{-1022}(0.f)$ (unnormalisierte Zahlen).
5. Wenn $e = 0$ und $f = 0$, dann ist $v = (-1)^s \cdot 0$ (Null).

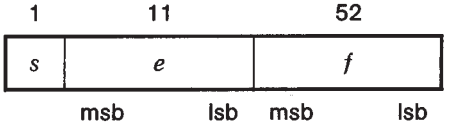


Bild 2. Doppelt langes Format

3.3 Erweiterte Formate

Die erweitert einfach langen und erweitert doppelt langen Formate kodieren in einer von der Implementierung abhängigen Weise die Wertemengen in Abschnitt 3.1 abhängig von den Beschränkungen der Tabelle 1. Diese Norm erlaubt eine Implementierung, um einige Werte redundant zu kodieren, vorausgesetzt, daß die Redundanz für den Anwender transparent im folgenden Sinne ist: eine Implementierung muß entweder jeden Wert ungleich Null eindeutig kodieren oder sie darf redundante Kodierungen von Werten ungleich Null nicht unterscheiden. Eine Implementierung darf auch einige Bitfolgen für Operationen außerhalb des Anwendungsbereiches dieser Norm reservieren. Tritt eine solche reservierte Bitfolge als Operand auf, ist das Ergebnis durch diese Norm nicht spezifiziert.

Von einer Implementierung dieser Norm wird nicht erwartet (und der Anwender sollte das nicht annehmen), dafür zu sorgen, daß erweitert einfach lange Formate einen größeren Zahlenbereich haben als erweitert doppelt lange Formate.

3.4 Kombinationen von Formaten

Alle Implementierungen, die dieser Norm entsprechen, müssen das einfach lange Format unterstützen. Implementierungen sollten das erweiterte Format entsprechend dem breitesten unterstützten Grundformat unterstützen und brauchen keine anderen erweiterten Formate zu unterstützen [1]).

4 Rundung

Bei der Rundung wird eine Zahl als unendlich genau betrachtet und, wenn notwendig, so modifiziert, daß sie in das Format des Zieles paßt, wobei die Ausnahme UNGENAU angezeigt wird (siehe Abschnitt 7.5). Mit Ausnahme der Binär ⟷ Dezimal-Umwandlungen (bei denen schwächere Bedingungen in Abschnitt 5.6 festgelegt sind), muß jede Operation, die in Abschnitt 5 festgelegt ist, so durchgeführt werden, als ob sie zuerst ein korrektes unendlich genaues Zwischenergebnis liefert, das einen unbegrenzten Bereich hat und das dann nach einem der Rundungsverfahren in diesem Abschnitt gerundet wird.

Die Rundungsverfahren wirken sich auf alle Rechenoperationen aus, ausgenommen Vergleichs- und Restoperationen. Die Rundungsverfahren können die Vorzeichen von „Null-Summen" (siehe Abschnitt 6.3) beeinflussen. Sie beeinflussen die Grenzen, außerhalb derer ÜBERLAUF (siehe Abschnitt 7.3) und UNTERLAUF (siehe Abschnitt 7.4) angezeigt werden können.

4.1 Rundung auf den nächstgelegenen Wert

Eine Implementierung dieser Norm muß die Rundung auf den nächstgelegenen Wert als den voreingestellten Rundungs-Modus zur Verfügung stellen. In diesem Modus muß der darstellbare Wert, der dem unendlich genauen Ergebnis am nächsten kommt, geliefert werden. Sind die beiden nächsten darstellbaren Werte gleich nah, muß derjenige geliefert werden, dessen Bit niedrigster Wertigkeit gleich Null ist. Jedoch muß ein unendlich genaues Ergebnis mit einer Größe von mindestens $2^{E_{\max}}(2-2^{-p})$ ohne eine Änderung des Vorzeichens auf ∞ gerundet werden; $E_{\max}$ und p sind hier durch das Format im Ziel (Abschnitt 3) festgelegt. Diese Festlegung kann durch Rundungsgenauigkeits-Vereinbarungen (siehe Abschnitt 4.3) aufgehoben werden.

4.2 Gerichtete Rundungen

Eine Implementierung muß drei vom Anwender auswählbare gerichtete Rundungsverfahren zur Verfügung stellen: Runden gegen $+\infty$, Runden gegen $-\infty$ und Runden gegen Null.

Wenn gegen $+\infty$ gerundet wird, muß das Ergebnis dem Wert des Formats (möglicherweise $+\infty$) am nächsten kommen und darf nicht kleiner sein als das unendlich genaue Ergebnis. Wenn gegen $-\infty$ gerundet wird, muß das Ergebnis dem Wert des Formats (möglicherweise $-\infty$) am nächsten kommen und darf nicht größer als das unendlich genaue Ergebnis sein.

Wenn gegen 0 gerundet wird, muß das Ergebnis dem Wert des Formats am nächsten kommen und darf im Wert nicht größer als das unendlich genaue Ergebnis sein.

4.3 Rundungsgenauigkeit

Üblicherweise wird ein Ergebnis entsprechend der Genauigkeit seines Zieles gerundet. Jedoch liefern einige Systeme Ergebnisse nur an doppelt lange oder erweiterte Ziele. Mit einem solchen System muß der Anwender, der ein Hochsprachen-Compiler sein kann, in der Lage sein festzulegen, daß ein Ergebnis auf einfache Genauigkeit gerundet wird, obwohl es im doppelt langen oder erweiterten Format mit seinem größeren Exponentenbereich gespeichert sein könnte [2]). Entsprechend muß ein System, das Ergebnisse nur an erweitert doppelt lange Ziele liefert, dem Anwender erlauben, die Rundung auf einfache oder doppelte Genauigkeit festzulegen. Um den Festlegungen in Abschnitt 4.1 zu entsprechen, ist zu beachten, daß das Ergebnis nicht mehr als einen Rundungsfehler enthält.

5 Operationen

Alle Implementierungen nach dieser Norm müssen Operationen zur Verfügung stellen, die addieren, subtrahieren, multiplizieren, dividieren, Quadratwurzel ziehen, den Rest ermitteln, auf Ganze Zahlen im Gleitpunkt-Format runden, zwischen verschiedenen Gleitpunkt-Formaten umwandeln, zwischen Gleitpunkt-Formaten und Formaten Ganzer Zahlen umwandeln, Binär-Format ⟷ Dezimal-Format umwandeln und vergleichen. Ob Kopieren ohne Formatwechsel als Operation betrachtet wird, ist eine Wahlmöglichkeit bei der Implementierung. Alle Operationen, ausgenommen Binär ⟷ Dezimal-Umwandlungen, müssen so ausgeführt werden, als ob sie zuerst ein unendlich genaues Zwischenergebnis mit unbegrenztem Zahlenbereich liefern und dieses Zwischenergebnis dann in das Zielformat einpassen würden (siehe Abschnitte 4 und 7). Abschnitt 6 erweitert die folgenden Festlegungen, um ± 0, $\pm\infty$ und NaNs abzudekken; Abschnitt 7 zählt Ausnahmen auf, die durch Ausnahmefälle für Operanden und Ergebnisse verursacht werden.

5.1 Arithmetik

Eine Implementierung muß für jedes unterstützte Format und für zwei beliebige Operanden des gleichen Formats, die Additions-, Subtraktions-, Multiplikations-, Divisions- und Rest-Operationen bereitstellen. Ebenfalls sollte sie die Operationen für Operanden mit sich unterscheidenden Formaten unterstützen. Das Zielformat (ungeachtet der Kontrolle der Rundungsgenauigkeit von Abschnitt 4.3) muß mindestens so breit wie das Format des breiteren Operanden sein. Alle Ergebnisse müssen, wie in Abschnitt 4 festgelegt, gerundet werden.

Ist $y \neq 0$, ist der Rest $r = x \text{ REM } y$, unabhängig vom Rundungsverfahren definiert durch die mathematische Beziehung $r = x - y \cdot n$, wobei n die ganze Zahl ist, die dem exakten Wert x/y am nächsten kommt. Wenn $|n - x/y| = 1/2$ ist, ist n gerade. Folglich ist der Rest immer exakt. Ist $r = 0$, muß das Vorzeichen von r das von x sein. Die Genauigkeitskontrolle (siehe Abschnitt 4.3) darf keinen Einfluß auf die Restoperation haben.

5.2 Quadratwurzel

Die Quadratwurzel-Operation muß für alle unterstützten Formate bereitgestellt werden. Das Ergebnis ist definiert und hat ein positives Vorzeichen für alle Operanden ≥ 0, mit folgender Ausnahme: $\sqrt{-0} = -0$. Das Zielformat muß mindestens so breit wie das der Operanden sein. Das Ergebnis muß nach Abschnitt 4 gerundet werden.

---

[1]) Nur wenn Aufwärtskompatibilität und Geschwindigkeit wichtig sind, sollte ein System, welches das erweitert doppelt lange Format unterstützt, auch erweiterte einfach lange unterstützen.

[2]) Eine Kontrolle der Rundungsgenauigkeit hat zum Zweck, Systemen, deren Ziele immer doppelt lang oder erweitert sind, die Möglichkeit zu geben, die Genauigkeit von Systemen mit einfach und doppelt langem Ziel, bei nicht vorhandenem ÜBER- oder UNTERLAUF nachzuahmen. Eine Implementierung sollte keine Operationen ermöglichen, die mit nur einer Rundung doppelt lange oder erweiterte Operanden kombinieren, um ein einfach langes Ergebnis zu liefern oder erweiterte doppelt lange Operanden kombinieren, um ein doppelt langes Ergebnis zu liefern.

5.3 Gleitpunkt-Format-Umwandlungen

Es muß möglich sein, Gleitpunkt-Zahlen aller unterstützten Formate untereinander umzuwandeln. Wenn die Umwandlung in ein Format geringerer Genauigkeit erfolgt, muß das Ergebnis, wie in Abschnitt 4 festgelegt, gerundet werden. Umwandlung in eine vergrößerte Genauigkeit ist ohne Ausnahme genau.

5.4 Umwandlungen zwischen Gleitpunkt-Zahl und Ganzer Zahl

Es muß möglich sein, alle unterstützten Gleitpunkt- und ganzzahligen Formate untereinander umzuwandeln. Die Umwandlung in eine Ganze Zahl muß durch Rundung gemäß Abschnitt 4 durchgeführt werden. Umwandlungen zwischen Ganzen Zahlen im Gleitpunkt-Format und Ganzzahlformaten müssen genau sein, sofern nicht eine Ausnahme nach Abschnitt 7.1 auftritt.

5.5 Rundung einer Gleitpunkt-Zahl auf einen ganzzahligen Wert

Es muß möglich sein, eine Gleitpunkt-Zahl auf einen ganzzahligen Wert innerhalb desselben Gleitpunkt-Formates zu runden. Die Rundung muß, wie in Abschnitt 4 festgelegt, so erfolgen, daß das Ergebnis ganzzahlig ist, wenn auf den nächstgelegenen Wert gerundet wird, und der Unterschied zwischen dem nicht gerundeten Operanden und dem gerundeten Ergebnis genau 1/2 beträgt.

5.6 Binär ⟷ Dezimal-Umwandlung

Die Umwandlung zwischen mindestens einem Dezimalzahlenformat und allen unterstützten Grundformaten für binäre Gleitpunkt-Zahlen muß für alle Zahlen innerhalb des in Tabelle 2 festgelegten Bereiches ermöglicht werden. Die ganzen Zahlen M und N in den Tabellen 2 und 3 sind so gewählt, daß die Dezimalzahlen Werte von $\pm m \cdot 10^{\pm N}$ haben. Bei der Eingabe müssen nachfolgende Nullen M angehängt oder weggelassen werden (bis zu den Grenzen, die in Tabelle 2 festgelegt sind), um N zu minimieren. Ist das Ziel eine Dezimalzahl, sollte ihre Ziffer niedrigster Wertigkeit zum Zweck der Rundung durch die Formatspezifikation angegeben sein.

Wenn die Ganze Zahl M außerhalb des in den Tabellen 2 und 3 festgelegten Bereiches liegt, d. h. wenn $M \geq 10^9$ für einfach lange Formate oder 10^{17} für doppelt lange Formate gilt, kann der Programmierer nach seiner Wahl alle signifikanten Ziffern, beim einfach langen Format nach der neunten und beim doppelt langen Format nach der siebzehnten, in andere Dezimalziffern abändern, üblicherweise in 0.

Für Operanden, die innerhalb der in Tabelle 3 festgelegten Bereiche liegen, müssen Umwandlungen gemäß Abschnitt 4 korrekt gerundet werden; andernfalls darf der Fehler des Ergebnisses, der bei der Umwandlung und der Rundung auf den nächstgelegenen Wert entsteht, um nicht mehr als 0,47 Einheiten der niedrigstwertigen Stelle des Zieles den Fehler übersteigen, der bei der Rundung nach Abschnitt 4 entstehen würde. Dabei wird vorausgesetzt, daß kein Exponenten-Unter-/Überlauf auftritt. Bei den gesteuerten Rundungsverfahren muß der Fehler das korrekte Vorzeichen haben und darf nicht 1,47 Einheiten der letzten Stelle übersteigen.

Umwandlungen müssen monoton sein; das bedeutet, die Erhöhung des Wertes einer binären Gleitpunkt-Zahl darf bei der Umwandlung in eine Dezimalzahl keine Werteverminderung zur Folge haben. Ebensowenig darf eine Erhöhung des Wertes einer Dezimalfolge keine Werteverminderung bei der Umwandlung in eine binäre Gleitpunkt-Zahl zur Folge haben.

Bei der Rundung auf den nächstgelegenen Wert muß die Umwandlung von Binär in Dezimal und zurück in Binär Identität ergeben, wenn die Dezimalzahl mit der maximalen Genauigkeit verarbeitet wird, nämlich (wie in Tabelle 2 festgelegt) 9 Stellen für Einzelformate und 17 für Doppelformate [3]).

Tritt bei der Umwandlung vom Dezimalformat in das Binärformat ÜBER-/UNTERLAUF auf, erfolgt die Meldung nach Abschnitt 7. Wenn Über-/Unterlauf, NaNs und Unendlichkeiten bei der Umwandlung von Binär- in Dezimalzahlen auftreten, sollte dieses dem Anwender durch entsprechende Zeichenfolgen angezeigt werden. Diese Norm sagt nichts darüber aus, wie mit NaNs zu verfahren ist, die in Dezimalfolgen kodiert sind.

Um Widersprüche zu vermeiden, sollten die für die Binär/Dezimal-Umwandlung verwendeten Prozeduren zu denselben Resultaten führen, unabhängig davon, ob die Umwandlung während der Sprachübersetzung (Interpretation, Kompilierung oder Assemblierung) oder während der Programmausführung (Laufzeit und interaktive Ein-/Ausgabe) durchgeführt wird.

Tabelle 2. **Dezimale Umwandlungsbereiche**

Format	Dezimal in Binär		Binär in Dezimal	
	max M	max N	max M	max N
Einfach lang	10^9-1	99	10^9-1	53
Doppelt lang	$10^{17}-1$	999	$10^{17}-1$	340

Tabelle 3. **Dezimaler Umwandlungsbereich für genaue Rundung**

Format	Dezimal in Binär		Binär in Dezimal	
	max M	max N	max M	max N
Einfach lang	10^9-1	13	10^9-1	13
Doppelt lang	$10^{17}-1$	27	$10^{17}-1$	27

5.7 Vergleich

Es muß möglich sein, Gleitpunkt-Zahlen aller unterstützter Formate zu vergleichen, auch dann, wenn die Formate der Operanden sich unterscheiden. Vergleiche sind genau und führen niemals zu UNTERLAUF oder ÜBERLAUF. Vier sich gegenseitig ausschließende Relationen sind möglich: „kleiner als", „gleich", „größer als" und „nichtgeordnet". Der letzte Fall tritt auf, wenn mindestens ein Operand eine NaN ist. Jede NaN ergibt beim Vergleich „nichtgeordnet", auch wenn sie mit sich selbst verglichen wird. Vergleiche müssen das Vorzeichen von Null ignorieren (also $+0 = -0$).

Das Ergebnis eines Vergleichs muß auf eine von zwei Arten geliefert werden: entweder durch einen Code, der eine der vier oben aufgeführten Relationen kennzeichnet oder als eine TRUE/FALSE Antwort auf ein Prädikat, das den gewünschten spezifischen Vergleich angibt. Zusätzlich zu der TRUE/FALSE Antwort muß ein Ausnahmefall „UNZULÄSSIGE OPERATION" (siehe Abschnitt 7.1) angezeigt werden, wenn unter Verwendung einer der Aussagen „<" oder

[3]) Die für die Umwandlungen spezifizierten Eigenschaften werden durch Fehlergrenzen beeinflußt, die vom Format (einfach oder doppelt lang) und der Anzahl der Dezimalstellen abhängen. Die erwähnten 0,47 sind nur die Grenze im ungünstigsten Fall. Eine detaillierte Diskussion dieser Fehlergrenzen und ökonomischen Umwandlungslogarithmen, die das erweiterte Format ausnutzen, ist zu finden in „Accurate Yet Economical Binary ⟷ Decimal Conversions" von Jerome T. Coonen (Ph. D. Dissertation, University of California, Berkeley).

Tabelle 4. **Prädikate und Relationen**

Prädikate			Relation				Ausnahmefall
ad hoc	FORTRAN	math	größer als	kleiner als	gleich	nichtgeordnet	ungültig, wenn nichtgeordnet
=	.EQ.	=	F	F	T	F	nein
?<>	.NE.	≠	T	T	F	T	nein
>	.GT.	>	T	F	F	F	ja
>=	.GE.	≥	T	F	T	F	ja
<	.LT.	<	F	T	F	F	ja
<=	.LE.	≤	F	T	T	F	ja
?	nichtgeordnet		F	F	F	T	nein
<>	.LG.		T	T	F	F	ja
<=>	.LEG.		T	T	T	F	ja
?>	.UG.		T	F	F	T	nein
?>=	.UGE.		T	F	T	T	nein
?<	.UL.		F	T	F	T	nein
?<=	.ULE.		F	T	T	T	nein
?=	.UE.		F	F	T	T	nein
NICHT (>)			F	T	T	T	ja
NICHT (>=)			F	T	F	T	ja
NICHT (<)			T	F	T	T	ja
NICHT (<=)			T	F	F	T	ja
NICHT (?)			T	T	T	F	nein
NICHT (<>)			F	F	T	T	ja
NICHT (<=>)			F	F	F	T	ja
NICHT (?>)			F	T	T	F	nein
NICHT (?>=)			F	T	F	F	nein
NICHT (?<)			T	F	T	F	nein
NICHT (?<=)			T	F	F	F	nein
NICHT (?=)			T	T	F	F	nein

„>", aber nicht „?" wie in der letzten Spalte von Tabelle 4 angegeben, „nichtgeordnete" Operanden verglichen werden. (Das Symbol „?" bedeutet hier „nichtgeordnet").

Tabelle 4 zeigt in der ersten Spalte durch drei Schreibweisen (ad hoc, FORTRAN-angelehnt und mathematisch) die 26 nützlichen, funktional klar unterschiedenen Prädikate. Sie zeigt, wie sie durch die vier Codes erhalten werden und sagt, welche Prädikate eine unzulässige Ausnahmeoperation verursachen, wenn die Relation „nichtgeordnet" ist. Die Eintragungen T und F geben an, ob die Aussage TRUE oder FALSE ist, wenn die entsprechende Relation gilt.

Es wird darauf hingewiesen, daß Aussagen in Paaren auftreten, von denen jeweils eine die logische Negation der anderen ist. Der Gebrauch des Präfix „NICHT" zur Negation einer Aussage in Tabelle 4 dreht den TRUE/FALSE-Gehalt der zugeordneten Eintragungen um, läßt jedoch die Eintragungen der letzten Spalte unverändert [4]).

Implementierungen mit verfügbaren Prädikaten müssen die ersten 6 Prädikate in Tabelle 4 bereitstellen. Sie sollten das siebente Prädikat sowie ein Hilfsmittel für die logische Negation von Prädikaten zur Verfügung stellen.

6 Unendlich, NaNs und Null mit Vorzeichen

6.1 Arithmetik mit Unendlich

Arithmetik mit Unendlich muß als der Grenzfall der reellen Arithmetik mit Operanden beliebiger Größe aufgefaßt werden, wenn eine solche Grenze existiert. Unendlichkeiten müssen in dem Sinne interpretiert werden, daß $-\infty <$ (jede endliche Zahl) $< +\infty$ gilt.

Arithmetik mit ∞ ist immer exakt und darf deshalb keine Ausnahmefälle anzeigen, außer bei den für ∞ in Abschnitt 7.1

[4]) Es gibt zwei Möglichkeiten, um die logische Negation einer Aussage zu schreiben: Eine, die „NICHT" explizit verwendet und eine andere, die den Beziehungsoperator umkehrt. Zum Beispiel kann die logische Negation von $(x=y)$ entweder als NICHT $(x=y)$ oder $(x?<>y)$ dargestellt werden. In diesem Fall sind beide Schreibweisen funktionell äquivalent zu $(x \neq y)$. Jedoch tritt diese Koinzidenz nicht bei anderen Prädikaten auf. Zum Beispiel ist die logische Negation von $(x<y)$ nur NICHT $(x<y)$; das umgekehrte Prädikat $(x?>=y)$ unterscheidet sich davon dahingehend, daß es den Ausnahmefall UNZULÄSSIGE OPERATION nicht anzeigt, wenn x und y „nichtgeordnet" sind.

festgelegten unzulässigen Rechenoperationen. Die Ausnahmen, die sich auf ∞ beziehen, werden nur angezeigt, wenn

1. ∞ entsteht durch endliche Operanden bei ÜBERLAUF (siehe Abschnitt 7.3) oder DIVISION DURCH NULL (siehe Abschnitt 7.2), wobei der entsprechende Trap gesperrt ist oder
2. ∞ ein ungültiger Operand (siehe Abschnitt 7.1) ist.

6.2 Operationen mit NaNs

Zwei unterschiedliche Arten von NaNs, anzeigende und nichtanzeigende, müssen bei allen Operationen unterstützt werden. Anzeigende NaNs werden benutzt als Werte für nicht initialisierte Variablen und arithmetik-ähnliche Erweiterungen (wie komplex-affine Unendlichkeit oder extrem große Bereiche), die nicht Gegenstand dieser Norm sind. Nichtanzeigende NaNs sollten durch Mittel, die im Ermessen des Programmierers liegen, zurückblickende Diagnoseinformationen bieten, hervorgerufen durch ungültige oder nicht verfügbare Daten und Ergebnisse. Die Weiterleitung der diagnostischen Information erfordert, daß die in den NaNs enthaltene Information in allen arithmetischen Operationen und Gleitpunkt-Formatumwandlungen erhalten bleibt.

Anzeigende NaNs müssen reservierte Operanden sein, die die Ausnahme UNZULÄSSIGE OPERATION (siehe Abschnitt 7.1) für jede in Abschnitt 5 aufgeführte Operation anzeigen. Ob das Kopieren einer anzeigenden NaN ohne einen Formatwechsel die Ausnahme UNZULÄSSIGE OPERATION anzeigt, liegt im Ermessen des Erstellers.

Jede Operation, die eine anzeigende NaN betrifft oder jede unzulässige Operation (siehe Abschnitt 7.1) muß, wenn kein Trap auftritt und wenn ein Gleitpunkt-Resultat geliefert werden soll, eine nichtanzeigende NaN als Ergebnis liefern.

Jede Operation, die ein oder zwei Eingabe-NaNs betrifft, von denen keine anzeigt, darf keine Ausnahme anzeigen. Sie muß jedoch, wenn ein Gleitpunkt-Resultat geliefert werden soll, als ihr Resultat eine nichtanzeigende NaN liefern, die eine der Eingabe-NaNs sein sollte. Dabei ist zu beachten, daß Formatumwandlungen nicht immer die gleiche NaN liefern könnten. Nichtanzeigende NaNs wirken wie anzeigende NaNs auf Operationen, die kein Gleitpunkt-Resultat liefern. Diese Operationen, nämlich Vergleich und Umwandlung in ein Format, das keine NaNs hat, werden in den Abschnitten 5.4, 5.6, 5.7 und 7.1 behandelt.

6.3 Das Vorzeichenbit

Diese Norm erläutert nicht das Vorzeichen einer NaN. Jedoch ist das Vorzeichen eines Produktes oder Quotienten die Exklusiv-Oder-Verknüpfung der Vorzeichen der Operanden. Ebenso unterscheidet sich das Vorzeichen einer Summe oder einer Differenz $x-y$, betrachtet als eine Summe $x + (-y)$, höchstens von einem der Vorzeichen der Summanden. Diese Regeln gelten auch dann, wenn Operanden oder Ergebnisse Null oder unendlich sind.

Wenn die Summe zweier Operanden mit entgegengesetztem Vorzeichen (oder die Differenz zweier Operanden mit gleichen Vorzeichen) exakt Null ist, muß das Vorzeichen dieser Summe (oder Differenz) in allen Rundungsverfahren „+" sein, ausgenommen bei Rundung gegen $-\infty$, wobei das Vorzeichen „−" sein muß. Jedoch behält $x+x=x-(-x)$ das gleiche Vorzeichen wie x, auch dann wenn x Null ist.

Mit der Ausnahme, daß $\sqrt{-0}$ gleich -0 sein muß, muß jede gültige Quadratwurzel ein positives Vorzeichen haben.

7 Ausnahmefälle

Es gibt 5 Ausnahmefälle, die bei Auftreten angezeigt werden müssen. Diese Meldung bewirkt das Setzen eines „status flag", die Durchführung eines Trap oder möglicherweise beides. Jedem Ausnahmefall sollte ein eigener Trap zugeordnet werden, der unter Anwenderkontrolle steht, wie in Abschnitt 8 festgelegt. Die Standardreaktion auf eine Ausnahmesituation muß die Programmfortsetzung ohne Trap sein. Diese Norm legt sowohl für die Durchführung eines Trap als auch ohne Trap die Resultate fest. In einigen Fällen ist das Resultat unterschiedlich, je nachdem, ob der Trap freigegeben ist oder nicht.

Für jeden Ausnahmefall muß die Implementierung ein „status flag" zur Verfügung stellen, das bei jedem Auftreten der entsprechenden Ausnahme gesetzt werden muß, wenn kein entsprechender Trap auftritt. Es darf nur auf Anweisung des Anwenders zurückgesetzt werden. Der Anwender muß in der Lage sein, die „status flags" individuell zu überprüfen und zu ändern und sollte weiterhin in der Lage sein, alle fünf „status flags" gleichzeitig zu retten und wiederherzustellen.

Die einzigen Ausnahmefälle, die gleichzeitig auftreten können, sind „UNGENAU MIT ÜBERLAUF" und „UNGENAU MIT UNTERLAUF".

7.1 Unzulässige Operation

Der Ausnahmefall „UNZULÄSSIGE OPERATION" wird angezeigt, wenn ein Operand für die durchzuführende Operation unzulässig ist. Das Ergebnis muß, wenn die Ausnahme ohne Trap auftritt, eine nichtanzeigende NaN (siehe Abschnitt 6.2) sein, vorausgesetzt, das Ziel hat ein Gleitpunkt-Format. Die unzulässigen Operationen treten auf:

1. bei jeder Operation mit einer anzeigenden NaN (siehe Abschnitt 6.2);
2. bei Addition oder Subtraktion: größenmäßige Subtraktion unendlicher Größen wie $(+\infty) + (-\infty)$;
3. bei Multiplikation: $0 \cdot \infty$;
4. bei Division; $0 : 0$ oder $\infty : \infty$;
5. bei Rest: x REM y, wobei y Null oder x unendlich ist;
6. bei Quadratwurzel, wenn der Operand kleiner als Null ist;
7. bei Umwandlung einer binären Gleitpunkt-Zahl in eine Ganze Zahl oder in ein Dezimalformat, wenn Überlauf, Unendlich oder eine NaN eine getreue Darstellung in diesem Format ausschließen und dies nicht anders angezeigt werden kann; und
8. bei Vergleichen mit den Prädikaten „<" oder „>", ohne „?", wenn die Operanden „nichtgeordnet" (siehe Abschnitt 5.7, Tabelle 4) sind.

7.2 Division durch Null

Wenn der Divisor Null und der Dividend eine endliche Zahl ungleich Null ist, muß die Ausnahme „DIVISION DURCH NULL" angezeigt werden. Das Ergebnis muß, wenn kein Trap auftritt, ein korrekt vorzeichenbehaftetes ∞ sein (siehe Abschnitt 6.3).

7.3 Überlauf

Der Ausnahmefall „ÜBERLAUF" wird immer dann angezeigt, wenn das fiktive gerundete Gleitpunkt-Ergebnis (siehe Abschnitt 4) mit unbegrenzt angenommenen Exponenten den darstellbaren Zahlenbereich des Zielformates überschreitet. Das Ergebnis muß, wenn kein Trap auftritt, durch das Rundungsverfahren und das Vorzeichen des theoretischen Ergebnisses wie folgt festgesetzt werden:

1. Rundung auf den nächstgelegenen Wert bildet alle Überläufe auf ∞, mit dem Vorzeichen des theoretischen Ergebnisses ab.
2. Rundung gegen Null bildet alle Überläufe auf die größte endliche Zahl des Formats mit dem Vorzeichen des theoretischen Ergebnisses ab.
3. Rundung gegen $-\infty$ bildet positive Überläufe auf die größte endliche Zahl des Formats und negative Überläufe auf $-\infty$ ab.

4. Rundung gegen $+\infty$ bildet negative Überläufe auf die negativste endliche Zahl des Formats und positive Überläufe auf $+\infty$ ab.

   Überläufe mit Trap müssen, ausgenommen bei Umwandlungen, bei allen Operationen der Trap-Bearbeitungsprozedur das Ergebnis liefern, das durch Dividieren des unendlich genauen Ergebnisses durch 2^{α} und anschließendes Runden erhalten wird. Die Bias-Einstellung α ist 192 im einfach langen, 1536 im doppelt langen und $3 \cdot 2^{n-2}$ im erweiterten Format, wobei n die Anzahl der Bits im Exponentenfeld ist [5]). ÜBERLAUF mit Trap bei Umwandlungen eines binären Gleitpunkt-Formates muß der Trap-Bearbeitungsprozedur ein Ergebnis in diesem oder in einem breiteren Format, möglicherweise mit transformierten Exponenten liefern, jedoch auf die Genauigkeit des Zieles gerundet. ÜBERLAUF mit Trap bei der Umwandlung vom Dezimal- in das Binärformat muß der Trap-Bearbeitungsprozedur ein Ergebnis in dem breitesten unterstützten Format, möglicherweise mit transformiertem Exponenten liefern, jedoch auf die Genauigkeit des Zieles gerundet. Wenn das Ergebnis zu weit außerhalb des Bereiches für den Bias liegt, um verschoben zu werden, muß stattdessen ein nichtanzeigendes NaN geliefert werden.

7.4 Unterlauf

Zwei verwandte Ereignisse tragen zum UNTERLAUF bei. Eines ist die Bildung eines winzigen Ergebnisses ungleich Null zwischen $\pm 2^{E_{min}}$, das, weil es so klein ist, später einen anderen Ausnahmefall verursachen kann, z. B. ÜBERLAUF nach Division. Das andere ist außergewöhnlicher Genauigkeitsverlust bei der Approximation von solchen winzigen Zahlen durch unnormalisierte Zahlen. Der Ersteller kann auswählen, wie diese Ereignisse erkannt werden, muß jedoch diese Ereignisse bei allen Operationen auf die gleiche Art handhaben. „Winzigkeit" kann entweder erkannt werden:

1. „Nach Rundung": wenn ein Ergebnis ungleich Null, so berechnet, als ob der Exponentenbereich unbegrenzt wäre, zwischen $\pm 2^{E_{min}}$ liegt, oder
2. „vor Rundung": wenn ein Ergebnis ungleich Null, so berechnet, als ob sowohl der Exponentenbereich als auch die Genauigkeit unbegrenzt wären, zwischen $\pm 2^{E_{min}}$ liegt.

Genauigkeitsverlust kann entweder erkannt werden

3. als ein Verlust bei Entnormalisierung: wenn das gelieferte Ergebnis sich von dem unterscheidet, was bei unbegrenztem Exponentenbereich errechnet worden wäre

oder

4. als ein ungenaues Ergebnis: wenn das gelieferte Ergebnis sich von dem unterscheidet, das errechnet worden wäre, wenn der Zahlenbereich des Exponenten und die Genauigkeit beide unbegrenzt gewesen wären. (Das ist die in Abschnitt 7.5 bezeichnete Bedingung „UNGENAU".)

Wenn ein Unterlauf mit Trap nicht implementiert oder nicht freigegeben ist (der Standard-Fall), darf UNTERLAUF nur angezeigt werden (durch das Unterlauf-Flag), wenn beide, „Winzigkeit" und Genauigkeitsverlust, erkannt worden sind. Das Verfahren zur Erkennung von „Winzigkeit" und Genauigkeitsverlust beeinflußt nicht das gelieferte Resultat, das Null, unnormalisiert oder $\pm 2^{E_{min}}$ sein könnte. Wenn ein UNTERLAUF-Trap implementiert und freigegeben ist, muß UNTERLAUF ohne Rücksicht auf Genauigkeitsverlust angezeigt werden, wenn „Winzigkeit" erkannt wird. Bei allen Operationen, ausgenommen Umwandlung, müssen Unterläufe mit Trap der Trap-Bearbeitungsprozedur das Ergebnis liefern, das durch Multiplizieren mit 2^{α} und anschließendem Runden des unendlich genauen Ergebnisses erhalten wird. Die Bias-Einstellung α ist 192 im einfach langen, 1536 im doppelt langen und $3 \cdot 2^{n-2}$ im erweiterten Format, wobei n die Anzahl der Bits im Exponentenfeld ist [6]). Unterläufe bei Umwandlung mit Trap müssen analog zur Handhabung von Überläufen bei Umwandlung bearbeitet werden.

7.5 Ungenau

Wenn das gerundete Ergebnis einer Operation nicht genau ist, oder wenn es ohne ÜBERLAUF-Trap überläuft, muß der Ausnahmefall UNGENAU angezeigt werden. Das gerundete oder übergelaufene Ergebnis muß an das Ziel geliefert werden oder im Fall des UNGENAU-Trap zu der Trap-Bearbeitungsprozedur.

8 Traps

Einem Anwender sollte es möglich sein, für jede der fünf Ausnahmefälle einen Trap einzurichten, indem er eine Bearbeitungsprozedur für ihn festlegt. Es sollte ihm möglich sein zu bestimmen, daß eine vorhandene Bearbeitungsprozedur gesperrt, gerettet oder wiederhergestellt wird. Er sollte ebenfalls in der Lage sein zu ermitteln, ob eine bestimmte Trap-Bearbeitungsprozedur für einen bestimmten Ausnahmefall freigegeben wurde. Wenn ein Ausnahmefall, dessen Trap gesperrt ist, angezeigt wird, muß er in der in Abschnitt 7 festgelegten Weise bearbeitet werden. Wenn ein Ausnahmefall, dessen Trap freigegeben ist, angezeigt wird, muß die Ausführung des Programms, in der der Ausnahmefall auftrat, ausgesetzt, die vorher vom Anwender festgelegte Trap-Bearbeitungsprozedur aktiviert und ihr ein Ergebnis, soweit in Abschnitt 7 festgelegt, zur Verfügung gestellt werden.

8.1 Trap-Bearbeitungsprozedur

Eine Trap-Bearbeitungsprozedur sollte die Fähigkeiten eines Unterprogramms haben, das einen Wert zurückgeben kann, welcher anstelle des Ergebnisses der Ausnahmefall-Operation verwendet werden kann. Dieses Ergebnis ist undefiniert, wenn es nicht durch die Trap-Bearbeitungsprozedur geliefert wird. Ebenso kann (können) das (die) „flag(s)" der Ausnahmefälle, die mit ihnen zugeordneten, freigegebenen Traps angezeigt werden, undefiniert sein, soweit sie nicht von der Trap-Bearbeitungsprozedur gesetzt oder zurückgesetzt worden sind.

Wenn in einem System ein Trap auftritt, sollte die Trap-Bearbeitungsprozedur in der Lage sein zu ermitteln,

1. welche Ausnahme(n) trat(en) bei dieser Operation auf;
2. die Art der Operation, die ausgeführt wurde;
3. das Format des Zieles;
4. für die Ausnahmefälle ÜBERLAUF, UNTERLAUF und UNGENAU das korrekt gerundete Ergebnis, einschließlich der Information, die nicht in das Format des Zieles passen würde, und
5. bei UNZULÄSSIGER OPERATION und bei DIVISION DURCH NULL die Werte der Operanden.

8.2 Vorrang

Wenn freigegeben, haben die Traps für ÜBERLAUF und UNTERLAUF Vorrang vor einem zusätzlichen UNGENAU-Trap.

---

[5]) Die Einstellung der Versatzkonstanten wird gewählt, um über-/untergelaufene Werte so nahe wie möglich in die Mitte des Exponentenbereiches umzusetzen, so daß, wenn gewünscht, diese in nachfolgend skalierten Operationen verwendet werden können, mit geringerem Risiko, weitere Ausnahmefälle zu verursachen.

[6]) Es ist zu beachten, daß ein System, dessen unterlegte Hardware bei UNTERLAUF immer einen Trap auslöst und ein gerundetes, skaliertes Ergebnis produziert, anzeigen muß, ob die Mantisse dieses Ergebnisses aufgerundet ist. Somit kann das korrekte, unnormalisierte Ergebnis durch Systemsoftware erzeugt werden, wenn der Anwender-UNTERLAUF-Trap gesperrt ist.

Anhang A

Empfohlene Funktionen und Prädikate

Dieser Anhang ist nicht Bestandteil dieser Norm für Binäre Gleitpunkt-Arithmetik, sondern dient nur der Information.

Die folgenden Funktionen und Prädikate werden als Hilfe zur Portierbarkeit von Programmen auf verschiedene Systeme empfohlen, deren Arithmetik vielleicht sehr unterschiedlich durchgeführt wird. Sie werden ganz allgemein beschrieben, d.h. die Typen der Operanden und Ergebnisse sind den Operanden inhärent.

Sprachen, die explizite Typvereinbarung erfordern, werden entsprechende Gruppen von Funktionen und Prädikaten haben.

Einige der untenstehenden Funktionen, wie die Kopieroperation $y := x$ ohne Wechsel des Formates, können nach Ermessen des Erstellers als nichtarithmetische Operationen behandelt werden, die für anzeigende NaNs den Ausnahmefall UNZULÄSSIGE OPERATION nicht melden. Die in Frage kommenden Funktionen sind 1., 2., 6. und 7.

1. copysign (x, y) ergibt x mit dem Vorzeichen von y. Somit gilt abs (x) = copysign $(x, 1.0)$, sogar wenn x eine NaN ist.
2. $-x$ ist x mit umgekehrtem Vorzeichen, nicht $0-x$; die Unterscheidung ist relevant, wenn x eine NaN oder ± 0 ist. Folglich wäre es ein Fehler, das Vorzeichenbit zur Unterscheidung zwischen anzeigenden und nichtanzeigenden NaNs zu verwenden.
3. scal b (y, N) ergibt $y \cdot 2^N$ für ganzzahlige Werte von N, ohne 2^N zu berechnen.
4. log$b(x)$ ergibt den nichttransformierten Exponenten von x, eine Ganze Zahl mit Vorzeichen im Format von x, außer daß log$b$ (NaN) eine NaN ist, log$b(\infty)$ gleich $+\infty$ und log$b(0)$ gleich $-\infty$ ist, mit der Anzeige des Ausnahmefalls DIVISION DURCH NULL. Wenn x positiv und endlich ist, liegt der Ausdruck scal $b\,(x, -\log b(x))$ genau zwischen 0 und 2; er ist nur kleiner als 1, wenn x unnormalisiert ist.
5. nextafter (x, y) ergibt den nächsten darstellbaren Nachbarn von x in der Richtung auf y zu. Folgende Spezialfälle ergeben sich: Wenn $x = y$ ist, ist das Ergebnis gleich x, ohne daß eine Ausnahme angezeigt wird; hingegen, wenn entweder x oder y eine nichtanzeigende NaN ist, ist das Ergebnis die eine oder die andere der Eingabe-NaNs. ÜBERLAUF wird angezeigt, wenn x endlich ist, aber nextafter (x, y) unendlich ist; UNTERLAUF wird angezeigt, wenn nextafter (x, y) genau zwischen $\pm 2^{E_{\min}}$ liegt, in beiden Fällen wird UNGENAU angezeigt.
6. finite (x) ergibt den Wert TRUE, wenn $-\infty < x < +\infty$ und andernfalls FALSE.
7. isnan (x), oder äquivalent $x \neq x$, ergibt den Wert TRUE, wenn x eine Nichtzahl (NaN) ist und andernfalls FALSE.
8. $x <> y$ ist nur TRUE, wenn $x < y$ oder $x > y$, und ist verschieden zu $x \neq y$, das NICHT $(x = y)$ bedeutet (siehe Tabelle 4).
9. unordered (x, y) oder $x ? y$, ergibt den Wert TRUE, wenn x „nichtgeordnet" mit y ist, andernfalls FALSE (siehe Tabelle 4).
10. class (x) sagt aus, unter welche der folgenden zehn Klassen x fällt: anzeigende NaN, nichtanzeigende NaN, $-\infty$, negativ normalisiert ungleich Null, negativ unnormalisiert, -0, $+0$, positiv unnormalisiert, positiv normalisiert ungleich Null, $+\infty$. Diese Funktion führt niemals zu einem Ausnahmefall, auch nicht für anzeigende NaNs.