norme française

NF C 96-036
Janvier 1992
HD 592 S1

arithmétique binaire en virgule flottante pour systèmes à microprocesseurs.

E : Binary floating-point arithmetic for microprocessor systems.

D : Duale Gleitkommaarithmetik für Mikroprozessor-Systeme.

Norme française homologuée par décision du Directeur Général de l'afnor le 20 Décembre 1991 pour prendre effet à compter du 20 Janvier 1992.

correspondance

Cette norme reproduit le document d'harmonisation HD 592 S1 adopté le 15 Mars 1991 par le CENELEC basé sur la publication 559 deuxième édition (1989-01) de la CEI.

analyse

L'objectif est qu'une réalisation d'un système à virgule flottante conforme à la présente norme puisse être effectuée entièrement par logiciel, entièrement par matériel, ou par une combinaison quelconque de logiciel et de matériel. C'est l'environnement que le programmeur ou l'utilisateur voit qui est conforme ou non conforme à cette norme. Les composants matériels qui nécessitent un support logiciel pour devenir conformes ne doivent pas être qualifiés de conformes indépendamment d'un tel logiciel.

descripteurs

Arithmétique binaire en virgule flottante pour systèmes à microprocesseurs.

modifications

corrections

Editée et diffusée par l'Union technique de l'Electricité, cedex 64 - 92052 paris la défense - tél. (1) 46 91 11 11

diffusée également par l'association française de normalisation (afnor) tour europe, cedex 7 - 92080 paris la défense - tél. (1) 42 91 55 55

Ets Busson, impr., 75018 Paris

1er tirage 92-01

SOMMAIRE

Ce document a été adopté par le Comité de Direction de l'UTE, le 27 Novembre 1991.

ARITHMETIQUE BINAIRE EN VIRGULE FLOTTANTE POUR SYSTEMES A MICROPROCESSEURS

---

1. Domaine d'application

1.1 *Objectifs de réalisation*

L'objectif est qu'une réalisation d'un système à virgule flottante conforme à la présente norme puisse être effectuée entièrement par logiciel, entièrement par matériel, ou par une combinaison quelconque de logiciel et de matériel. C'est l'environnement que le programmeur ou l'utilisateur voit qui est conforme ou non conforme à cette norme. Les composants matériels qui nécessitent un support logiciel pour devenir conformes ne doivent pas être qualifiés de conformes indépendamment d'un tel logiciel.

1.2 *Inclusions*

Cette norme spécifie:

1) les formats de base et étendu des nombres en virgule flottante;
2) les opérations d'addition, de soustraction, de multiplication, de division, de calcul d'une racine carrée, du calcul d'un reste et de comparaison;
3) les conversions entre nombres entiers et nombres en virgule flottante;
4) les conversions entre différents formats en virgule flottante;
5) les conversions entre les nombres en virgule flottante en format de base et les chaînes décimales, et
6) la détection et le traitement des conditions d'exception pour les nombres en virgule flottante y compris les non-nombres ("NaN").

1.3 *Exclusions*

Cette norme ne spécifie pas:

1) les formats des chaînes décimales et des entiers;
2) l'interprétation des champs de signe et de mantisse des non-nombres ("NaN"), ou
3) les conversions de binaire à décimal et réciproquement pour les formats étendus.

2. Définitions

Exposant avec excédent

Somme de l'exposant et d'une constante (excédent ou biais) choisie de manière à rendre non négatif le domaine de l'exposant avec excédent.

Nombre binaire en virgule flottante

Chaîne de bits caractérisée par trois éléments: un signe, un exposant signé et une mantisse. Sa valeur numérique, si elle existe, est le produit signé de sa mantisse par deux élevé à la puissance de son exposant. Dans la présente norme, une chaîne de bits n'est pas toujours distinguée du nombre qu'elle représente.

Nombre dénormalisé

Nombre en virgule flottante non nul dont l'exposant a une valeur réservée, d'habitude la valeur minimale du format et dont le bit significatif de la mantisse, explicite ou implicite, est nul.

Destination

Emplacement devant contenir le résultat d'une opération binaire ou unaire. La destination peut être soit désignée explicitement par l'utilisateur ou fournie de manière implicite par le système (pour les résultats intermédiaires dans les sous-expressions ou les arguments de procédures par exemple). Certains langages placent les résultats des calculs intermédiaires dans des emplacements non accessibles par l'utilisateur. Néanmoins, cette norme définit le résultat d'une opération en termes du format de cette destination aussi bien que des valeurs des opérandes.

Exposant

Elément d'un nombre binaire en virgule flottante qui représente normalement la puissance entière à laquelle deux est élevé pour déterminer la valeur du nombre représenté. Occasionnellement, l'exposant est appelé exposant signé ou exposant sans excédent.

Partie fractionnaire

Partie de la mantisse située à droite de sa virgule correspondante.

Mode

Variable qu'un utilisateur peut positionner, tester, sauvegarder et restaurer pour diriger l'exécution des opérations arithmétiques ultérieures. Le mode par défaut est le mode valable tant qu'une instruction contraire explicite n'est pas incluse dans le programme ou sa spécification.

Les modes suivants doivent être mis en place:

1) arrondi, pour commander la direction des erreurs d'arrondi, et dans certaines réalisations;

2) précision de l'arrondi, pour diminuer la précision des résultats. Le réalisateur peut fournir optionnellement les modes suivants:

3) déroutements désactivés/activés, pour gérer les conditions d'exception.

NaN (non-nombre)

Non-nombre; entité symbolique codée selon le format virgule flottante. Il existe deux types de non-nombres (voir 6.2). Les non-nombres indicateurs indiquent une condition d'exception concernant une opération invalide (voir 7.1) lorsqu'ils apparaissent en tant qu'opérandes. Les non-nombres muets se propagent à travers presque toutes les opérations arithmétiques sans signaler d'exceptions.

Résultat

Chaîne de bits (représentant généralement un nombre) qui est livrée à la destination.

Mantisse

Elément d'un nombre binaire en virgule flottante constituée d'un bit significatif explicite ou implicite placé à gauche de la virgule et d'un champ fractionnaire à droite de la virgule.

Doit

Le mot "doit" recouvre les parties obligatoires de toute réalisation conforme.

Il convient - Il est recommandé - Il y a lieu

Ces termes recouvrent les parties qui sont fortement recommandées comme étant dans l'esprit de cette norme, bien que des contraintes architecturales ou autres, hors du domaine de cette norme, puissent à l'occasion rendre ces recommandations peu pratiques.

Indicateur d'état

Variable qui peut prendre deux états, actif (valeur 1) ou inactif (valeur 0). Un utilisateur peut désactiver un indicateur, le copier, ou le remettre dans un état antérieur. Lorsqu'il est actif, un indicateur peut contenir des informations supplémentaires dépendant du système et éventuellement inaccessibles à certains utilisateurs. Les opérations définies par cette norme peuvent avoir comme effet secondaire l'activation de certains des indicateurs suivants: résultat inexact, dépassement de capacité inférieur, dépassement de capacité supérieur, division par zéro et opération invalide.

Utilisateur

Toute personne, tout matériel ou logiciel non spécifié lui-même dans cette norme, ayant accès aux opérations de l'environnement de programmation spécifiées dans cette norme et qui les commande.

3. Formats

Cette norme définit quatre formats de virgule flottante en deux groupes, de base et étendu, chacun admettant deux largeurs, simple et double précision. Les niveaux de réalisation standard se distinguent par les combinaisons de formats supportés.

3.1 *Ensembles de valeurs*

Ce paragraphe concerne seulement les valeurs numériques représentables dans un format, et non leur codage qui fait l'objet des paragraphes suivants. Les seules valeurs représentables dans un format donné sont celles qui sont spécifiées selon les trois paramètres entiers suivants:

P = nombre de bits significatifs (précision)

E_{max} = valeur maximale de l'exposant, et

E_{min} = valeur minimale de l'exposant

Les paramètres de chaque format sont regroupés dans le tableau 1. Pour chaque format, les seules entités qui doivent être fournies sont:

Nombres de la forme $(-1)^s 2^E (b_0 b_1 b_2 \ldots b_{p-1})$

où:

s vaut 0 ou 1;
E est un entier compris entre E_{min} et E_{max} bornes incluses, et
chaque b_i vaut 0 ou 1.

Deux valeurs infinies, $+\infty$ et $-\infty$;
au moins un non-nombre indicateur, et
au moins un non-nombre muet.

Tableau 1 - Résumé des paramètres du format

Paramètre	Format			
	Simple	Simple Etendu	Double	Double Etendu
P	24	≥32	53	≥64
E_{max}	+127	≥+1 023	+1 023	≥+16 383
E_{min}	-126	≤-1 022	-1 022	≤-16 382
Excédent de l'exposant	+127	Non spécifié	+1 023	Non spécifié
Largeur de l'exposant (bits)	8	≥11	11	≥15
Largeur du format (bits)	32	≥43	64	≥79

La description précédente énumère certaines valeurs de manière redondante, par exemple:

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

Cependant, le codage de telles valeurs non nulles peut être redondant seulement pour les formats étendus (voir 3.3). Les valeurs non nulles de la forme $\pm 2^{E_{min}} (0 \cdot b_1 b_2 \ldots b_{p-1})$ sont appelées

"dénormalisées". Des valeurs réservées d'exposants peuvent être utilisées pour coder les non-nombres, ±∞, ±0, et les nombres dénormalisés. Pour toute variable ayant la valeur zéro, le bit de signe *s* fournit un bit supplémentaire d'information. Bien que tous les formats aient des représentations distinctes pour +0 et -0, les signes sont significatifs en certaines circonstances, comme la division par zéro, et non dans d'autres. Dans cette norme, 0 et ∞ sont écrits sans signe lorsque ce dernier n'a pas d'importance.

3.2 *Formats de base*

Les nombres en format simple et double précision se composent de trois champs:

un signe *s* de 1 bit,

un exposant avec excédent e = E + excédent, et

une partie fractionnaire $f = .b_1 b_2 \ldots b_{p-1}$.

Le domaine des valeurs de l'exposant sans excédent E doit inclure tout entier placé entre les bornes E_{min} et E_{max} incluses, ainsi que deux autres valeurs réservées: $E_{min}-1$ pour coder ±0 et les nombres dénormalisés, et $E_{max}+1$ pour coder ±∞ et les non-nombres. Les paramètres ci-dessus apparaissent dans le tableau I. Chaque valeur numérique non nulle possède un codage unique. Les champs sont interprétés comme suit:

3.2.1 *Simple précision*

Un nombre *X* de 32 bits en format simple précision est constitué comme indiqué sur la figure 1. La valeur v de *X* se déduit de ses champs constitutifs soit:

1) Si $e = 255$ et $f \neq 0$, alors v est un non-nombre quel que soit *s*
2) Si $e = 255$ et $f = 0$, alors $v = (-1)^s \infty$
3) Si $0 < e < 255$, alors $v = (-1)^s \, 2^{e-127} \, (1,f)$
4) Si $e = 0$ et $f \neq 0$, alors $v = (-1)^s \, 2^{-126} \, (0,f)$ (nombres dénormalisés)
5) Si $e = 0$ et $f = 0$, alors $v = (-1)^s \, 0$ (zéro)

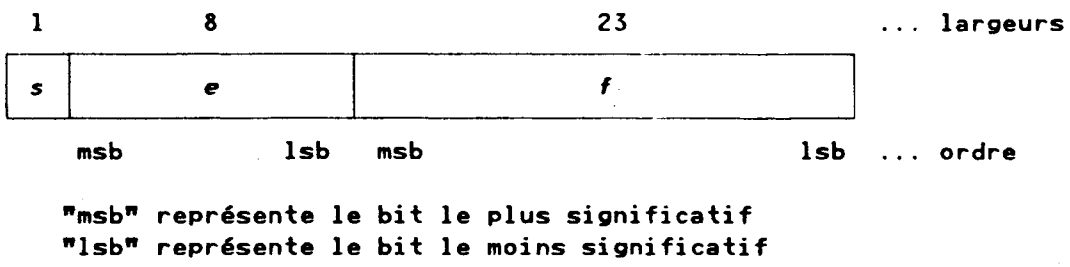

Figure 1 - Format simple précision

3.2.2 *Double précision*

Un nombre X de 64 bits en format double précision est constitué comme indiqué sur la figure 2. La valeur v de X se déduit de ses champs constitutifs, soit:

1) Si $e = 2\,047$ et $f \neq 0$, alors v est un non-nombre quel que soit s
2) Si $e = 2\,047$ et $f = 0$, alors $v = (-1)^s \infty$
3) Si $0 < e < 2\,047$, alors $v = (-1)^s\, 2^{e-1\,023}\, (1,f)$
4) Si $e = 0$ et $f \neq 0$, alors $v = (-1)^s\, 2^{-1\,022}\, (0,f)$ (nombres dénormalisés)
5) Si $e = 0$ et $f = 0$, alors $v = (-1)^s\, 0$ (zéro)

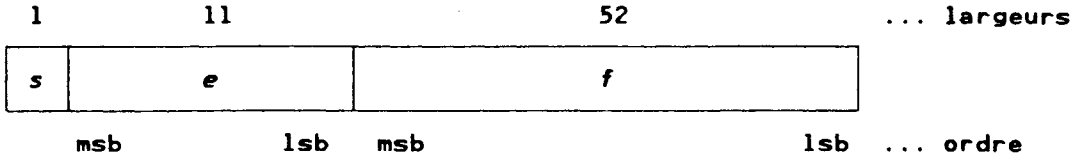

Figure 2 - Format double précision

3.3 *Formats étendus*

Les formats étendus simple précision et double précision assurent le codage de manière dépendante de la réalisation pour les ensembles de valeurs de 3.1, assujetties aux contraintes du tableau 1. Cette norme permet à une réalisation de coder certaines valeurs de manière redondante pourvu que cette redondance soit transparente à l'utilisateur selon le sens suivant: soit une réalisation doit coder chaque valeur non nulle de manière unique, soit elle ne doit faire aucune distinction entre les codages redondants des valeurs non nulles. Une réalisation peut également réserver quelques chaînes de bits pour des utilisations en dehors du domaine de cette norme; lorsqu'une telle chaîne de bits réservée est utilisée comme opérande, le résultat n'est pas spécifié par cette norme.

Une réalisation de cette norme n'est pas tenue de faire en sorte (et il y a lieu que l'utilisateur ne présume pas) que le format étendu simple précision possède un domaine plus grand que le format étendu double précision.

3.4 *Combinaisons de formats*

Toutes les réalisations conformes à cette norme doivent supporter le format simple précision. Il convient que les réalisations supportent le format étendu correspondant au format de base le plus grand qu'elles supportent et elles ne sont pas tenues de supporter d'autres formats étendus.*

---

* C'est seulement au cas ou la compatibilité ascendante et la vitesse de calcul sont importantes qu'il convient qu'un système supportant le format étendu double précision supporte aussi le format étendu simple précision.

4. Arrondi

L'arrondi considère un nombre comme infiniment précis et, si nécessaire, le modifie pour l'adapter au format de la destination tout en signalant l'exception d'inexactitude (voir 7.5). Sauf pour la conversion binaire-décimale (dont les conditions plus faibles sont spécifiées en 5.6), chaque opération spécifiée dans l'article 5 doit être effectuée comme si elle produisait tout d'abord un résultat intermédiaire correct avec une précision infinie et avec un domaine illimité, puis exécutait une opération d'arrondi sur ce résultat selon un des modes de cet article.

Les modes d'arrondi affectent toutes les opérations arithmétiques, sauf la comparaison et le calcul du reste. Les modes d'arrondi peuvent affecter les signes des sommes de valeur nulle (voir 6.3) et affectent les seuils au-delà desquels un dépassement (voir 7.3) et un dépassement inférieur (voir 7.4) peuvent être signalés.

4.1 *Arrondi au plus près*

Une réalisation de cette norme doit fournir un arrondi optimal (au plus près) comme mode d'arrondi par défaut. Dans ce mode, elle doit fournir la valeur représentable la plus proche du résultat théorique infiniment précis; si les deux valeurs représentables les plus proches sont à égale distance du résultat théorique, la valeur fournie sera celle dont le bit de poids faible est égal à zéro. Cependant, un résultat infiniment précis avec une valeur au moins de $2^{E_{max}} (2 - 2^{-P})$ doit être arrondi à ∞ sans changement de signe; dans ce cas, E_{max} et P sont déterminés par le format de destination (article 3), à moins que ce dernier ne soit annulé par un mode de précision d'arrondi (voir 4.3).

4.2 *Arrondis orientés*

Une réalisation doit aussi fournir trois modes d'arrondi orientés au choix de l'utilisateur: arrondi vers $+\infty$, arrondi vers $-\infty$ et arrondi vers 0.

Lors de l'arrondi vers $+\infty$, le résultat doit être la valeur du format (éventuellement $+\infty$) la plus proche du résultat théorique de précision infinie et non inférieure à ce résultat théorique. Lors de l'arrondi vers $-\infty$, le résultat doit être la valeur du format (éventuellement $-\infty$) la plus proche du résultat théorique de précision infinie et non supérieure à ce résultat théorique.

Lors de l'arrondi vers 0, le résultat doit être la valeur du format la plus proche du résultat théorique de précision infinie, et non supérieure en valeur absolue à ce résultat théorique.

4.3 *Précision d'arrondi*

En général, un résultat est arrondi avec la précision correspondant au format de sa destination. Cependant, certains systèmes donnent des résultats en format double précision ou en format étendu. Sur un tel système, l'utilisateur, qui peut être un compilateur de langage évolué, doit pouvoir spécifier qu'un résultat doit être arrondi en simple précision, même lorsqu'il est mémorisé en format double précision ou

étendu avec un exposant ayant un domaine plus étendu*. De la même manière, un système qui fournit des résultats seulement sous le format étendu double précision doit permettre à l'utilisateur de spécifier l'arrondi en simple ou double précision. Il faut remarquer que pour satisfaire aux spécifications de 4.1, le résultat ne peut pas être entaché de plus d'une erreur d'arrondi.

5. Opérations

Toutes les réalisations conformes à cette norme doivent fournir les opérations suivantes: addition, soustraction, multiplication, division, extraction de racine carrée, calcul du reste, arrondi vers entier en format virgule flottante, conversion entre différents formats de virgule flottante, conversion entre formats virgule flottante et entiers, conversion binaire-décimale et comparaisons. Le fait de considérer la copie sans changement de format comme une opération est une option de la réalisation. Excepté pour la conversion binaire-décimale, chacune des opérations doit être effectuée comme si elle produisait tout d'abord un résultat intermédiaire correct avec une précision infinie et un domaine de valeurs illimité, puis contraignait ce résultat intermédiaire à s'adapter au format de destination (articles 4 et 7). L'article 6 étend les spécifications suivantes de manière à couvrir ± 0, $\pm\infty$ et les non-nombres; l'article 7 énumère les cas d'exception causés par des opérandes et des résultats hors gamme.

5.1 *Arithmétique*

Une réalisation doit offrir les opérations d'addition, de soustraction, de multiplication, de division et de calcul du reste pour n'importe quel couple d'opérandes de même format, et ce pour chaque format qui est supporté. Il y a lieu qu'elle offre aussi ces opérations pour des opérandes de format différents. Le format de destination (sans tenir compte de la commande de la précision d'arrondi de 4.3) doit être au moins aussi étendu que le format le plus étendu des opérandes. Tous les résultats doivent être arrondis comme spécifié dans l'article 4.

Lorsque $y \neq 0$, le reste $r = x$ REM y est défini quel que soit le mode d'arrondi par la relation mathématique $r = x - y \times n$, où n est l'entier le plus proche de la valeur exacte x/y; lorsque $|n - x/y| = \frac{1}{2}$, alors n est pair. Ainsi, le reste est toujours exact. Si $r = 0$, son signe doit être celui de x. La commande de précision (voir 4.3) ne doit pas s'appliquer à l'opération de calcul du reste.

---

* La commande de la précision d'arrondi a pour but de permettre à des systèmes pour lesquels les résultats sont toujours en format double précision ou étendu de simuler, en l'absence de dépassement/dépassement inférieur, la précision des systèmes opérant en simple ou double précision. Il ne convient pas qu'une réalisation fournisse des opérations qui combinent des opérandes de format double précision ou étendu pour produire un résultat en simple précision, ni des opérations qui combinent des opérandes de format étendu double précision pour produire un résultat en double précision, avec seulement une opération d'arrondi.

5.2 *Racine carrée*

L'opération d'extraction de la racine carrée doit être offerte pour tous les formats supportés. Le résultat est défini et possède un signe positif pour tous les opérandes ≥0 à l'exception près que $\sqrt{-0}$ doit être égal à -0. Le format de destination doit être au moins aussi étendu que celui des opérandes. Le résultat doit être arrondi comme spécifié dans l'article 4.

5.3 *Conversions des formats virgule flottante*

Il doit être possible de convertir des nombres en virgule flottante entre tous les formats supportés. Si le résultat de la conversion a une précision moindre, le résultat doit être arrondi comme spécifié dans l'article 4. La conversion vers un format de précision supérieure, doit être exacte. Il n'y a pas d'exception.

5.4 *Conversion entre virgule flottante et entier*

Il doit être possible d'effectuer des conversions entre tous les formats en virgule flottante supportés. La conversion vers des entiers doit s'effectuer par arrondi comme spécifié dans l'article 4. Les conversions entre formats de nombres entiers en virgule flottante et formats de nombres entiers doit être exacte, à moins qu'une exception ne se produise comme il est spécifié en 7.1.

5.5 *Arrondi de nombres en virgule flottante vers une valeur entière*

Il doit être possible d'effectuer une opération d'arrondi sur un nombre en virgule flottante pour obtenir une valeur entière dans le même format virgule flottante. L'arrondi doit être conforme à la spécification de l'article 4, avec la condition que lors d'un arrondi au plus près, si la différence entre l'opérande non arrondi et le résultat arrondi est exactement un demi, le résultat arrondi est pair.

5.6 *Conversion binaire-décimale*

La conversion entre des chaînes décimales dans un format au moins et des nombres binaires en virgule flottante dans tous les formats de base supportés doit être offerte pour tous les nombres couvrant les domaines spécifiés dans le tableau 2. Les entiers M et N des tableaux 2 et 3 sont tels que les chaînes décimales ont pour valeur $\pm M \times 10^{\pm N}$. En entrée, les zéros de queue doivent être ajoutés ou ôtés de M (dans les limites spécifiées dans le tableau 2) de manière à minimiser N. Lorsque la destination est une chaîne décimale, il y a lieu que son digit le moins significatif soit positionné selon les spécifications du format concernant l'arrondi.

Lorsque l'entier M a une valeur située en dehors du domaine spécifié dans les tableaux 2 et 3, c'est-à-dire lorsque $M \geq 10^9$ pour la simple précision et 10^{17} pour la double précision, le réalisateur peut, en option altérer tous les digits significatifs situés après le neuvième en simple précision et après le dix-septième en double précision en leur donnant d'autres valeurs, en général 0.

Les conversions doivent être arrondies correctement, comme spécifié dans l'article 4 pour les opérandes inclus dans les domaines spécifiés dans le tableau 3. Sinon, dans le cas d'arrondi au plus près, l'erreur du résultat converti ne doit pas excéder de plus de 0,47 unités du digit le moins significatif du résultat l'erreur qui résulterait de l'application des spécifications d'arrondi de l'article 4, pour autant qu'il n'y ait aucun dépassement de capacité concernant l'exposant. Pour les modes d'arrondi orientés, l'erreur doit avoir le signe correct et ne doit pas dépasser 1,47 unités de plus faible poids.

Les conversions doivent être monotones. C'est-à-dire que l'augmentation de la valeur d'un nombre binaire exprimé en virgule flottante ne doit pas se traduire par une diminution de sa valeur après conversion en une chaîne décimale; et une augmentation de la valeur d'une chaîne décimale ne doit pas se traduire par une diminution de sa valeur après conversion en un nombre binaire en virgule flottante.

Lorsqu'une opération d'arrondi au plus près est effectuée, la conversion de binaire à décimal et réciproquement doit être l'identité tant que la chaîne décimale possède la précision maximale spécifiée dans le tableau 2, c'est-à-dire 9 digits en simple précision et 17 digits en double précision*.

Si la conversion décimale-binaire donne lieu à un dépassement supérieur ou inférieur, la réponse sera comme spécifié dans l'article 7. Il convient que les conditions de dépassement supérieur et inférieur, les non-nombres et les valeurs infinies apparaissant lors des conversions binaire à décimal soient signalées à l'utilisateur par des chaînes appropriées. Cette norme ne donne aucune information à propos du traitement des non-nombres codés sous forme de chaînes décimales.

Pour éviter des incohérences, il convient que les procédures utilisées pour la conversion binaire-décimale fournissent les mêmes résultats, que la conversion soit effectuée lors de la traduction du langage (interprétation, compilation ou assemblage) ou lors de l'exécution du programme (exécution et entrée/sortie interactive).

Tableau 2 - Domaine de conversion décimale

Format	Décimal-binaire		Binaire-décimal	
	$N_{max.}$	$N_{max.}$	$M_{max.}$	$N_{max.}$
Simple précision	$10^9 - 1$	99	$10^9 - 1$	53
Double précision	$10^{17} - 1$	999	$10^{17} - 1$	340

* Les propriétés spécifiées pour les conversions résultent des limites d'erreur qui dépendent du format (simple ou double précision) et du nombre de chiffres décimaux qui interviennent; la valeur 0,47 mentionnée correspond à la limite pour le pire des cas. Pour une discussion détaillée de ces bornes d'erreur et des algorithmes de conversion économiques qui utilisent le format étendu, se référer à "Accurate Yet Economical Binary ↔ Decimal conversions" par Jerome T. Coonen (Ph.D. Dissertation, Université de Californie, Berkeley).

Tableau 3 - Domaine d'arrondi exact pour la conversion décimale

Format	Décimal-binaire		Binaire-décimal	
	$M_{max.}$	$N_{max.}$	$M_{max.}$	$N_{max.}$
Simple précision	$10^9 - 1$	13	$10^9 - 1$	13
Double précision	$10^{17} - 1$	27	$10^{17} - 1$	27

5.7 *Comparaison*

Il doit être possible de comparer des nombres en virgule flottante dans tous les formats supportés, même si les formats des opérandes sont différents. Les comparaisons sont exactes et ne créent jamais de dépassement de capacité. Il existe quatre relations possibles, qui s'excluent mutuellement: "plus petit que", "égal", "plus grand que" et "non ordonné". Ce dernier cas survient lorsqu'un des opérandes au moins est un non-nombre. Chaque non-nombre doit donner "non ordonné" comme résultat de la comparaison avec tout élément y compris lui-même. Les comparaisons doivent ignorer le signe de zéro (ainsi +0 = -0).

Le résultat d'une comparaison doit être fourni selon l'une des deux manières suivantes: soit comme un code condition identifiant l'une des quatre relations indiquées ci-dessus, ou comme une réponse binaire (vrai-faux) à un prédicat qui identifie la comparaison spécifique désirée. En supplément à la réponse vrai-faux, une condition d'exception d'opération invalide (voir 7.1) doit être signalée quand, comme indiqué dans la dernière colonne du tableau 4, des opérandes "non ordonnés" sont comparés en utilisant un des prédicats mettant en oeuvre "<" ou ">" mais pas "?". Le symbole "?" signifie ici "non ordonné".

Le tableau 4 montre les vingt-six prédicats fonctionnellement distincts identifiés, dans la première colonne, au moyen de trois notations: ad hoc, FORTRAN et mathématique. Il montre comment ils sont obtenus à partir des quatre codes conditions et indique quels prédicats sont la cause d'une exception d'opération invalide lorsque la relation est "non ordonné". Les entrées V et F indiquent si le prédicat est vrai ou faux lorsque la relation correspondante est vérifiée.

Tableau 4 - Prédicats et relations

Prédicats			Relations				Exceptions
Ad hoc	FORTRAN	Math	Plus grand que	Plus petit que	Egal	Non ordonné	Invalide si non ordonné
=	.EQ.	=	F	F	V	F	Non
?<>	.NE.	≠	V	V	F	V	Non
>	.GT.	>	V	F	F	F	Oui
>=	.GE.	≥	V	F	V	F	Oui
<	.LT.	<	F	V	F	F	Oui
<=	.LE.	≤	F	V	V	F	Oui
?	non ordonné		F	F	F	V	Non
							Oui
<>	.LG.		V	V	F	F	
<=>	.LEG.		V	V	V	F	Oui
?>	.UG.		V	F	F	V	Non
?>=	.UGE.		V	F	V	V	Non
?<	.UL.		F	V	F	V	Non
?<=	.ULE.		F	V	V	V	Non
?=	.UE.		F	F	V	V	Non
NON(>)			F	V	V	V	Oui
NON(>=)			F	V	F	V	Oui
NON(<)			V	F	V	V	Oui
NON(<=)			V	F	F	V	Oui
NON(?)			V	V	V	F	Non
NON(<>)			F	F	V	V	Oui
NON(<=>)			F	F	F	V	Oui
NON(?>)			F	V	V	F	Non
NON(?>=)			F	V	F	F	Non
NON(?<)			V	F	V	F	Non
NON(?<=)			V	F	F	F	Non
NON(?=)			V	V	F	F	Non

Remarquer que les prédicats s'associent par paires, chacun étant la négation logique de l'autre; l'application d'un préfixe comme "NON" pour effectuer la négation d'un prédicat dans le tableau 4 échange le sens des valeurs vrai et faux des entrées associées, mais laisse la dernière colonne inchangée*.

Les réalisations qui offrent des prédicats doivent fournir les 6 premiers prédicats du tableau 4 et il convient qu'elles fournissent aussi le septième, ainsi que le moyen d'effectuer la négation logique des prédicats.

---

* Il peut sembler qu'il existe deux façons d'écrire la négation logique d'un prédicat, l'une utilisant "NON" de manière explicite et l'autre inversant l'opérateur relationnel. Par exemple, la négation logique de (X=Y) peut s'écrire soit NON(X=Y) ou (X?<>Y); dans ce cas, les deux expressions sont fonctionnellement équivalentes à (X≠Y). Cependant, cette coïncidence ne se produit pas pour les autres prédicats. Par exemple, la négation logique de (X<Y) est seulement NON(X<Y); le prédicat inverse (X?>=Y) est différent en ce sens qu'il ne signale pas d'exception d'opération invalide lorsque X et Y sont "non ordonnés".

6. Infini, non-nombres et zéro signé

6.1 *Arithmétique de l'infini*

L'arithmétique de l'infini doit être considérée comme le cas limite de l'arithmétique réelle, avec des opérandes de valeur aussi grande que l'on veut, lorsqu'une telle limite existe. Les infinis doivent s'interpréter au sens affine, c'est-à-dire que $-\infty <$ (tout nombre fini) $< +\infty$.

L'arithmétique sur ∞ est toujours exacte et, pour cette raison, ne doit signaler aucune exception, sauf pour les opérations invalides spécifiées pour ∞ en 7.1. Les cas d'exception qui relèvent de ∞ sont signalés seulement dans les cas suivants:

1) ∞ est créé a partir d'opérandes finis par dépassement de capacité (voir 7.3), ou par une division par zéro (voir 7.2), avec le déroutement correspondant désactivé, ou

2) ∞ est un opérande invalide (voir 7.1).

6.2 *Opérations avec des non-nombres*

Deux types de non-nombres, indicateur et muet, doivent être supportés dans toutes les opérations. Les non-nombres indicateurs autorisent des valeurs pour les variables non initialisées et des améliorations de type arithmétique (telles que les infinités complexes-affines ou les domaines d'amplitude extrême) qui sont en dehors de l'objet de cette norme. Il y a lieu que les non-nombres muets, par des moyens laissés à la discrétion du réalisateur, fournissent des informations de diagnostic rétrospectif héritées des données et résultats invalides ou non disponibles. La propagation des informations de diagnostic nécessite que l'information contenue dans les non-nombres soit préservée à travers les opérations arithmétiques et les conversions de format en virgule flottante.

Les non-nombres indicateurs doivent être des opérandes réservés qui signalent les conditions d'exception d'opération invalide (voir 7.1) pour chaque opération indiquée dans l'article 5. La décision de signaler une exception d'opération invalide lors de la copie d'un non-nombre indicateur sans changement de format est laissée au réalisateur comme option.

Toute opération portant sur un non-nombre indicateur ou toute opération invalide (voir 7.1) doit, si aucun déroutement ne se produit et si un résultat en virgule flottante est attendu, fournir un non-nombre muet comme résultat.

Toute opération portant en entrée sur un ou deux non-nombres, aucun n'étant un non-nombre indicateur, ne doit pas signaler d'exception, mais, si un résultat en virgule flottante est attendu, doit fournir comme résultat un non-nombre muet, qui appartiendrait à un des non-nombres d'entrée. Remarquer qu'il est possible que les conversions de format ne puissent fournir le même non-nombre. Les non-nombres muets ont des effets semblables aux non-nombres indicateurs qui ne fournissent pas de résultat en virgule flottante; ces opérations, à savoir la comparaison et la conversion à un format dépourvu de non-nombres, sont détaillés en 5.4, 5.6, 5.7 et 7.1.

6.3 *Bit de signe*

Cette norme ne parle pas du signe d'un non-nombre. Dans les autres cas, le signe d'un produit ou d'un quotient s'obtient par le "OU exclusif" des signes des opérandes; et le signe de la somme, ou de la différence $x - y$ considérée comme la somme $x + (-y)$, diffère de l'un des signes des termes au plus. Ces règles doivent s'appliquer même lorsque les opérandes ou les résultats sont zéro ou l'infini.

Lorsque la somme de deux opérandes de signes contraires (ou la différence de deux opérandes de même signe) est exactement nulle, le signe de cette somme (ou différence) doit être "+" dans tous les modes d'arrondi, sauf l'arrondi vers $-\infty$, auquel cas ce signe doit être "-". Cependant, $x + x = x - (-x)$ conserve le même signe que x même lorsque x est nul.

A l'exception près que $\sqrt{-0}$ doit avoir la valeur -0, toute racine carrée valide doit avoir un signe positif.

7. Exceptions

Il y a cinq cas d'exceptions qui doivent être signalés lorsqu'ils sont détectés. Cette signalisation peut consister à positionner un indicateur d'état, exécuter un déroutement, ou éventuellement effectuer les deux. A chaque exception, il convient qu'un déroutement soit associé sous contrôle de l'utilisateur, comme spécifié dans l'article 8. La réponse par défaut à une exception doit être de poursuivre sans déroutement. Cette norme spécifie les résultats devant être fournis selon les deux cas, déroutement et non-déroutement. Dans certains cas, le résultat est différent lorsqu'un déroutement est activé.

Pour chaque type d'exception, la réalisation doit fournir un indicateur d'état qui doit être mis à un lors de toute occurrence de l'exception correspondante lorsqu'aucun déroutement ne se produit. Cet indicateur doit être remis à zéro seulement à la demande de l'utilisateur. Ce dernier doit pouvoir tester et modifier les indicateurs d'état individuellement et, de plus, il y a lieu qu'il puisse sauvegarder et restaurer les cinq indicateurs en bloc.

Les seules exceptions qui peuvent coïncider sont "inexact avec dépassement" et "inexact avec dépassement inférieur".

7.1 *Opérations Invalides*

L'exception d'opération invalide est signalée lorsqu'un des opérandes est invalide pour l'opération à exécuter. Le résultat, lorsque l'exception se produit sans déroutement, doit être un non-nombre muet (voir 6.2), pourvu que la destination ait un format virgule flottante. Les opérations invalides sont les suivantes:

1) Toute opération sur un non-nombre indicateur (voir 6.2).

2) Addition ou soustraction: soustraction en valeur absolue de valeurs infinies comme $(+\infty) + (-\infty)$.

3) Multiplication $0 \times \infty$.

4) Division 0/0 ou ∞/∞.

5) Calcul du reste: *x* REM *y*, où *y* est nul ou *x* est infini.

6) Racine carrée: si l'opérande est négatif.

7) Conversion d'un nombre binaire en virgule flottante en un format entier ou décimal lorsqu'un dépassement de capacité, la présence de valeurs infinies ou de non-nombres empêchent une représentation fidèle dans ce format, et que ce fait ne peut pas être signalé autrement, et

8) comparaison utilisant des prédicats impliquant "<" ou ">", sans "?", lorsque les opérandes sont "non ordonnés" (voir 5.7, tableau 4).

7.2 *Division par zéro*

Si le diviseur est nul et le dividende un nombre fini non nul, alors l'exception division par zéro doit être signalée. Le résultat, lorsqu'aucun déroutement n'a lieu, doit être la valeur ∞ avec le signe correct (voir 6.3).

7.3 *Dépassement de capacité*

L'exception de dépassement de capacité doit être signalée lorsque la valeur du plus grand nombre fini représentable dans le format de destination est dépassée par le résultat théorique en virgule flottante avec arrondi qui serait obtenu si le domaine de l'exposant était illimité (article 4). Le résultat lorsqu'aucun déroutement ne se produit, doit être déterminé par le mode d'arrondi et le signe du résultat intermédiaire comme suit:

1) Dans le mode d'arrondi au plus près, tous les dépassements donnent le résultat ∞ avec le signe du résultat intermédiaire.

2) Dans le mode d'arrondi vers 0, tous les dépassements donnent comme résultat le plus grand nombre fini du format avec le signe du résultat intermédiaire.

3) Dans le mode d'arrondi vers -∞, tous les dépassements positifs donnent comme résultat le plus grand nombre fini du format et les dépassements négatifs donnent -∞ comme résultat.

4) Dans le mode d'arrondi vers +∞, tous les dépassements négatifs donnent comme résultat le nombre fini négatif le plus petit du format et les dépassements positifs donnent +∞ comme résultat.

Les dépassements donnant lieu à un déroutement doivent fournir à la routine de traitement de déroutement associée et pour toutes les opérations sauf les conversions, le résultat obtenu en divisant le résultat infiniment précis par 2^{α} puis en effectuant l'arrondi. L'ajustement d'excédent α est de 192 pour le format simple précision, de 1 536 en double précision, et de $3 \times 2^{n-2}$ dans le format étendu où n repré-

sente le nombre de bits du champ de l'exposant*. Les dépassements avec déroutements sur conversion de format binaire en virgule flottante doivent fournir à la routine de traitement de déroutement un résultat dans ce même format ou un format supérieur, avec éventuellement un ajustement de l'excédent de l'exposant, mais avec un arrondi correspondant à la précision de la destination. Les dépassements avec déroutements sur conversion décimale-binaire doivent fournir à la routine de traitement de déroutement un résultat dans le format le plus grand supporté, éventuellement avec un ajustement de l'excédent de l'exposant, mais avec un arrondi correspondant à la précision de la destination; lorsque le résultat a une valeur telle que l'exposant ne peut être ajusté, le dépassement doit fournir un non-nombre muet.

7.4 *Dépassement de capacité inférieur*

Deux événements corrélés contribuent à un dépassement inférieur. L'un est la création d'un résultat infinitésimal non nul de valeur comprise entre $\pm 2^{E_{min}}$ qui, à cause de sa petitesse, peut être la cause de quelque autre exception ultérieure, telle qu'un dépassement de capacité par division. L'autre est la perte considérable de précision lors de l'approximation par des nombres dénormalisés de tels nombres infinitésimaux. Le réalisateur peut choisir la manière dont ces événements sont détectés, mais doit détecter ces événements de la même manière pour toutes les opérations. La petitesse peut être détectée soit:

1) "après arrondi" lorsqu'un résultat non nul, calculé comme si le domaine de l'exposant était illimité, est compris strictement entre $\pm 2^{E_{min}}$,

ou

2) "avant arrondi" lorsqu'un résultat non nul, calculé comme si le domaine de l'exposant ainsi que la précision étaient tous deux illimités, serait compris strictement entre $\pm 2^{E_{min}}$,

Une perte de précision peut se détecter soit comme:

3) une perte de dénormalisation: lorsque le résultat fourni diffère de la valeur qui aurait été obtenue si le domaine de l'exposant avait été illimité,

ou

4) un résultat inexact: lorsque le résultat fourni diffère de la valeur qui aurait été obtenue si le domaine de l'exposant ainsi que la précision avaient été illimités. (C'est la condition appelée inexactitude en 7.5.)

---

* L'ajustement d'excédent est choisi de manière à décaler les valeurs correspondant aux dépassements supérieur et inférieur aussi près que possible du milieu du domaine de l'exposant, afin que, si on le désire, ces valeurs puissent être utilisées lors d'opérations ultérieures avec moins de risque de production de nouvelles exceptions.

Lorsqu'un déroutement associé à un dépassement de capacité n'est pas réalisé ou n'est pas activé (le cas par défaut), le dépassement de capacité doit être signalé (par l'intermédiaire de l'indicateur de dépassement inférieur) seulement lorsque la petitesse du résultat et la perte de précision ont été toutes deux détectées. La méthode de détection de la petitesse et de la perte de précision n'affecte pas le résultat fourni qui peut être zéro, dénormalisé ou $\pm 2^{E_{min}}$. Lorsqu'un déroutement pour dépassement de capacité inférieur a été réalisé et se trouve activé, le dépassement de capacité inférieur doit être signalé lorsque la petitesse est détectée quelle que soit la perte de précision. Pour toutes les opérations, sauf la conversion, les dépassements de capacité inférieurs avec déroutement doivent fournir à la routine de traitement de déroutement le résultat obtenu en multipliant le résultat théorique infiniment précis par 2^{α} puis en effectuant un arrondi. L'ajustement d'excédent α est de 192 en simple précision, de 1 536 en double précision, et de $3 \times 2^{n-2}$ en format étendu, ou n est le nombre de bits dans le champ de l'exposant*. Les dépassements de capacité inférieurs avec déroutement sur conversion doivent être traités de manière similaire au traitement des dépassements de capacité supérieurs sur conversion.

7.5 *Inexactitude*

Si le résultat arrondi d'une opération n'est pas exact ou s'il donne lieu à un dépassement de capacité, alors l'exception d'inexactitude doit être signalée. Le résultat arrondi ou avec dépassement de capacité doit être fourni à la destination ou, si un déroutement sur inexactitude se produit, à la routine de traitement de cette exception.

8. Déroutements

Il convient qu'un utilisateur puisse disposer d'un déroutement pour chacun des cinq cas d'exception en spécifiant une routine de traitement de déroutement pour celui-ci. Il y a lieu qu'il puisse commander qu'une routine de traitement existante soit désactivée, sauvegardée ou restaurée. Il convient aussi qu'il puisse déterminer si une routine de traitement de déroutement spécifique a été activée. Lorsqu'une exception dont le déroutement a été désactivé est signalée, elle doit être traitée de la manière spécifiée dans l'article 7. Lorsqu'une exception dont le déroutement est activé est signalée, l'exécution du programme dans lequel l'exception s'est produite doit être suspendue, la routine de traitement de déroutement préalablement spécifiée par l'utilisateur doit être exécutée, et un résultat, si spécifié dans l'article 7, doit lui être fourni.

---

* Remarquer qu'un système dont le matériel sous-jacent déclenche toujours des déroutements sur dépassement de capacité inférieur, produisant un résultat arrondi, à l'exposant ajusté, doit indiquer si un tel résultat est arrondi en valeur absolue, de manière que le résultat correctement dénormalisé puisse être produit par le logiciel du système lorsque le déroutement de dépassement de capacité inférieur de l'utilisateur est désactivé.

8.1 *Routine de traitement de déroutement*

Il y a lieu qu'une routine de traitement de déroutement possède les possibilités d'un sous-programme qui peut rendre une valeur devant être utilisée à la place du résultat de l'opération ayant donné lieu à l'exception; ce résultat est indéfini, à moins qu'il ne soit fourni par la routine de traitement de déroutement. De la même manière, le ou les indicateurs correspondant aux exceptions signalées avec leurs déroutements associés activés peuvent être indéfinis, à moins qu'ils ne soient mis à un ou à zéro par la routine de traitement de déroutement.

Lorsqu'un système effectue un déroutement, il convient que la routine de traitement du déroutement soit capable de déterminer:

1) Quelles exceptions se sont produites lors de cette opération.

2) Le type d'opération qui était effectuée.

3) Le format de destination.

4) Lors des cas de dépassement de capacité supérieur, inférieur, et pour les exceptions d'inexactitude, le résultat correct arrondi, y compris l'information qui pourrait ne pas être compatible avec le format de destination.

5) Lors des cas d'exceptions pour opérations invalides et de division par zéro, les valeurs des opérandes.

8.2 *Précédence*

Lorsqu'ils sont activés, les déroutements pour dépassement de capacité supérieur et inférieur ont la priorité sur un déroutement pour inexactitude séparé.

ANNEXE A

FONCTIONS ET PREDICATS RECOMMANDES

Cette annexe ne fait pas partie de la norme pour l'arithmétique binaire en virgule flottante, mais est incluse seulement pour information.

Les fonctions et prédicats suivants sont recommandés comme aidant à la portabilité des programmes vers différents systèmes, qui peut-être traitent de manière très différente l'arithmétique. Ils sont décrits de manière générique; c'est-à-dire les types des opérandes et des résultats sont considérés comme inhérents à ces opérandes. Les langages qui nécessitent un typage explicite auront des familles correspondantes de fonctions et de prédicats.

Certaines fonctions ci-dessous, comme l'opération de copie $y := x$ sans changement de format, peuvent, selon l'option prise par le réalisateur, être traitées comme des opérations non arithmétiques qui ne signalent pas l'exception d'opération invalide pour les non-nombres indicateurs; les fonctions en question sont 1), 2), 6) et 7).

1) copysign(x,y) rend x avec le signe de y. Donc abs(x) = copysign(x,1,0), même si x est un non-nombre.

2) $-x$ est égal à x copié avec le signe contraire, et non $0 - x$; la distinction se rapporte au cas ou x est ± 0 ou un non-nombre. Par conséquent, ce serait une erreur d'utiliser le bit de signe pour distinguer entre les non-nombres indicateurs et les non-nombres muets.

3) scal$b$(y,N) rend $y \times 2^N$ pour les valeurs entières de N, sans calculer explicitement 2^N.

4) log$b$(x) rend l'exposant sans excédent de x, un entier signé dans le format de x, sauf dans les cas log$b$(non-nombre) qui est un non-nombre, log$b$(∞) qui vaut $+\infty$, et log$b$(0) qui vaut $-\infty$ et signale l'exception de division par zéro. Lorsque x est positif et fini, l'expression scal$b$(x,-log$b$(x)) a une valeur comprise strictement entre 0 et 2; elle est inférieure à 1 seulement lorsque x est dénormalisé.

5) nextafter(x,y) rend le voisin représentable de x dans la direction de y. Les cas spéciaux suivants peuvent se produire: si $x=y$, alors le résultat est x sans qu'aucune exception ne soit signalée; dans le cas contraire, si soit x ou y est un non-nombre muet alors le résultat est un ou l'autre des non-nombres d'entrée. Un dépassement de capacité inférieur est signalé lorsque x est fini mais que nextafter(x,y) est infini; un dépassement de capacité inférieur est signalé lorsque nextafter(x,y) a une valeur strictement incluse dans $\pm 2^{E_{min}}$; dans ces deux cas l'exception d'inexactitude est signalée.

6) finite(x) rend la valeur VRAI si $-\infty < x < +\infty$, et rend FAUX dans les autres cas.

7) isnan(x) ou de manière équivalente $x \neq x$, rend la valeur VRAI si x est un non-nombre, et rend FAUX dans les autres cas.

8) $x<>y$ est VRAI seulement si $x<y$ ou $x>y$, et se distingue de $x \neq y$, qui signifie NON($x=y$) (Tableau 4).

9) unordered(x,y), ou $x?y$ rend la valeur VRAI si x est non ordonné avec y, et rend FAUX dans les autres cas (Tableau 4).

10) class(x) indique dans laquelle des dix classes suivantes x se place: non-nombre indicateur, non-nombre muet, $-\infty$, nombre négatif normalisé non nul, nombre négatif dénormalisé, -0, +0, nombre positif dénormalisé, nombre positif normalisé non nul, $+\infty$. Cette fonction ne donne jamais lieu à une exception, même dans le cas de non-nombres indicateurs.

E. B., impr., 75018 Paris (1er tr. 1992, n° 9830) - N° 9915 - NF C 96-036 - 700 - 01/92