Subject: Comments on Section 13
From: Van Snyder

1 Edits

Edits refer to 01-007r3. Page and line numbers are displayed in the margin. Absent other instructions, a page and line number or line number range implies all of the indicated text is to be replaced by immediately following text, while a page and line number followed by + (-) indicates that immediately following text is to be inserted after (before) the indicated line. Remarks are noted in the margin, or appear between [and] in the text.

Unless otherwise specified, the inquiry intrinsic functions accept array arguments for which the shape need not be defined. The shape of array arguments to transformational and elemental intrinsic functions shall be defined.	272:15-22
[Simplification: Editor: Insert "of type logical" after "argument" and delete [272:29].]	272:24, 29
[How can procedures be a set? Editor: Insert "consists of a set of" ⇒ "are".]	272:31
[Editor: "this section" ⇒ "13.7".]	273:33
[Editor: Move [275:31-32] to here, to put the list into alphabetical order.]	275:26+
[Where are the "other" type conversion functions that these are different from? Editor: "Other" ⇒ "Miscellaneous".]	275:33
[Number doesn't agree.] A specific intrinsic function marked with a bullet (•) shall not be used as an actual argument.	279:29
[ISO doesn't like the term "section".] Detailed specifications of the standard generic intrinsic procedures are provided here in	279:31
[Editor: "has value" ⇒ "has the value".]	281:6
[The term *target* has been split into *data-target* and *proc-target*, and an article is needed. Editor: "*target*" ⇒ "the *data-target* or *proc-target*".]	283:23
[It is inconsistent to describe the result in terms of a prohibited pattern. Editor: "AIMAG(X)" ⇒ "AIMAG(X,KIND)"; insert "X were real with the value REAL(X,KIND) and" before "Y".]	286:30
Example. See 13.7.42.	286:42+
[Editor: "*i*" ⇒ "*ii*" (probably requires changing the paragraph tag). Fixed in LaTeX.]	288:7
[One can't compare the timing of different algorithms on the same computer if different compilers are used. Editor: "computer" ⇒ "processor". Where else than "on a computer" would it be interesting to invoke CPU_TIME to measure the time taken by a computation? Editor: Delete "on a computer".]	288:30
[Editor: Remove the first "and"; delete "following"; Put "1985 ... 500" between "(/" and "/)" and move to be after "value"; delete the ":" after "DATE_TIME".]	290:32-33
[Improve precision of cross references: Editor: "7.1.4"⇒ "7.1.4.2" twice.]	291:31, 33
[Editor: Insert a comma before "which".]	293:5
[The dynamic type of *every* disassociated pointer and unallocated allocatable is the declared type.]	294:24-25

The dynamic type of a disassociated pointer or unallocated allocatable is its declared type.

[Editor: Insert "a" between "has" and "value".]	294:34
[Editor: "teh"⇒ "the".]	296:43
[Simplification (of the same sort as used elsewhere): Editor: "INT(A) ... A" ⇒ "INT(A) = INT (REAL (A, KIND(A)))".]	300:34-35
[It's confusing to say "1 if A else *n* if B else 1".]	302:23-29
Case (i): If ARRAY is a whole array or array structure component and either ARRAY is an assumed-size array of rank DIM or dimension DIM of array has nonzero extent, LBOUND (ARRAY, DIM) has a value equal to the lower bound of subscript DIM of ARRAY. Otherwise the result value is 1.	
[Doesn't work because MATMUL is not an *op* listed in 7.1.4.2. If it were, this could be simplified. It can be simplified (and made to work) anyway: Editor: at [306:11] "according to 7.1.4.2" ⇒ "as specified in 7.1.4.2 for the * operator"; at [306:13] "according to 7.1.4.2" ⇒ "as specified in 7.1.4.2 for the .AND. operator".]	306:10-14
["extended" is used in 7.2.3. Editor: "padded" ⇒ "extended".]	307:11
[Editor: "Note ... 1" ⇒ "Note that this is independent of the declared lower bounds for A" at [308:26] and ⇒ "Note that this is independent of the declared lower bounds for B" at [308:30].]	308:26,30
[Editor: Delete the extraneous blank after "zero".]	309:3
[Editor: ")" ⇒ "))".]	309:10
["extended" is used in 7.2.3. Editor: "padded" ⇒ "extended".]	310:13
[Editor: Insert "or allocatable" after "pointer" because we want to allow allocatable arguments but it's not allowed to have an allocatable actual argument associated with a pointer dummy argument; "Its" ⇒ "If MOLD is a pointer its"; Insert "If MOLD is allocatable its allocation status may be currently allocated or not currently allocated." before "If". Insert "or allocated" before ", the target".]	314:38-40
[Number doesn't agree. Editor: "type parameters" ⇒ "a type parameter".]	315:9
[There are two examples. Editor: "**Example**" ⇒ "**Examples**". Start new paragraphs with "REAL" and "The".]	315:25-26
[HUGE and TINY aren't defined for complex arguments. Editor: Delete "or complex".]	319:9
Case (iii): For complex argument the result has the value RANGE(REAL(X)).	319:10+
[We don't use this clumsy circumlocution anywhere else. Editor: "the processor-dependent ... the" ⇒ "that of".]	319:22-23
[The dynamic type of *every* disassociated pointer and unallocated allocatable is the declared type.] The dynamic type of a disassociated pointer or unallocated allocatable is its declared type.	321:11
[Editor: "criteria" ⇒ "criterion".]	322:35
[Not quite right, now that we can detect negative zero.] **Description.** Magnitude of A with the sign of B.	323:43

Editor: "b^{e-p}" ⇒ "$b^{\max(e-p,e_{MIN}-1)}$"; Delete ", provided ... range".] 325:16-17

[Why is the term "arithmetic" suddenly introduced here after not being used anywhere else in Section 13? Editor: Delete "arithmetic".] 327:3

[Editor: De-italicize the ")".] 329:4

13.8 Intrinsic modules 331:22+

Processors may provide public entities in addition to the public entities provided by intrinsic modules specified by this standard.

NOTE 14.1$\frac{1}{2}$

To avoid potential name conflicts with program entities, it is recommended that a program use the ONLY option in any USE statement that accesses an intrinsic module specified by this standard.

13.8.1 The ISO_C_BINDING module

The ISO_C_BINDING module is described in Section 15.

13.8.2 The IEEE_EXCEPTIONS, IEEE_ARITHMETIC, and IEEE_FEATURES modules

The IEEE_EXCEPTIONS, IEEE_ARITHMETIC, and IEEE_FEATURES modules are described in Section 14.

[Editor: Push 13.8 etc. down one level to make 13.8.3 etc.] 331:23

[Editor: Delete.] 331:24-30

---

2 Clarifications and simplifications

Many intrinsic functions refer to "whose model is as at the end of 13.4" It's note 13.5 to which reference is made. Why not say that? Editor: Replace "at the end of 13.4." by "in Note 13.5" at the following places:

291:10 293:30 294:8 295:3 297:22-23 307:23-24 310:25-26 314:4 316:28
317:41 319:11 320:38-39 321:22 323:26-27 325:19 328:14

A scalar is defined to have rank zero at [17:35], which allows a simplification. Delete "or ... one", "is an array of" ⇒ "has", and "of shape" ⇒ "shape" at the following places:

281:30-31 282:34-35 287:36-37 308:42-43 311:46-1 317:8-9

3 Increased functionality

3.1 Get_Command_Argument

[Editor: "Subroutine" ⇒ "Elemental subroutine".] 295:27

3.2 Get_Environment_Variable

[Editor: "Subroutine" ⇒ "Elemental subroutine".] 296:33

3.3 Huge

There is no way that it can be used in an initialization expression to know how many characters are in a character set.

[Editor: "or real" ⇒ ", real or character".] 297:17

Result Value. 297:19-21

Case (i): X is of type integer: The result has the value $r^q - 1$ where r and q are as defined in 13.4 for the model representing numbers of the same type and kind type parameter as X.

Case (ii): X is of type real: The result has the value $(1 - b^{-p})b^{e_{\max}}$ where b, p and $e_{\max}$ are as defined in 13.4 for the model representing numbers of the same type and kind type parameter as X.

Case (iii): X is of type character: The result has the value CHAR(n – 1,KIND(X)) where n is the number of characters in the representation method having the same kind as X.

3.4 ICHAR – note that this subsection passed

ICHAR doesn't work if C has a kind that occupies more storage space than a default integer.

[Editor: Insert "**[, KIND]**" after "**C**".] 299:8

[Editor: Make this the first element of a list of arguments.] 299:13-14

[Editor: Copy [294:31-33] to here.] 299:15

3.5 Merge

Several intrinsic functions permit their arguments not to be defined. The functionality of Merge would be increased if that were allowed here also. It is clear from [433:34-43] that it was intended to allow actual arguments not to be evaluated if the processor can determine that the value isn't necessary.

[Editor: Insert an additional sentence "It is not referenced and need not be defined if MASK is false."] 309:31

[Editor: Insert an additional sentence "It is not referenced and need not be defined if MASK is true."] 309:32

[Editor: Make the current example *Case(i)*. Then add:] 309:36-42

Case (ii): The value of MERGE(ORDER,10,PRESENT(ORDER)) has the value of ORDER if ORDER is present and 10 otherwise. Notice that ORDER is not defined if it is not present.

3.6 Product

It is useful to be able to carry out products in higher precision than the operands. That's the reason for DPROD. Since DPROD has only two arguments, it wouldn't be a disaster to convert them to double precision before doing the multiply. With PRODUCT, one would need to convert the array argument to double precision. This would be expected to take a lot more time and space than converting the arguments of DPROD.

PRODUCT(ARRAY,DIM[,MASK,KIND]) or PRODUCT(ARRAY[,MASK,KIND]) 316:39

[Editor: Copy [294:31] to here.] 317:6+

Result Characteristics. The type is the same as ARRAY. If KIND is present, the kind type parameter is that specified by the value of KIND; otherwise the kind type parameter is the same as the kind type parameter of ARRAY. It 317:7

Result Value. The product is formed as though ARRAY were converted to the kind of the result. 317:11