Subject: Comments on Section 9
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.

[Editor: "*file-unit number*" ⇒ "*file-unit-number*" – i.e., insert a hyphen.]	168:16
[Editor: Delete the extraneous "." at the end of [192:3] and add "." at the end of [192:4].]	192:3,4

2 Subclauses 9.5.3 and 9.9 ought to be combined

Subclauses 9.5.3 and 9.9 contain some duplicative material, some contradictory material and some incomplete material (mostly incomplete with respect to wait operations). There is no subclause not in 9.5 that addresses error conditions, but error conditions can occur in other than data transfer statements. To address these problems, subclauses 9.5.3 and 9.9 ought to be combined.

[Editor: Delete.]	183:30-8
[Editor: "9.5.3" ⇒ "9.9" twice.]	166:20,21
[Editor: "9.5.3" ⇒ "9.9".]	166:36
[Editor: "9.5.3" ⇒ "9.9".]	184:28
[Editor: "9.5.3" ⇒ "9.9".]	185:6
[Editor: "9.5.3" ⇒ "9.9".]	188:2
[Editor: "9.5.3" ⇒ "9.9".]	191:38
[Editor: "9.5.3" ⇒ "9.9".]	192:3
[Editor: "9.5.3" ⇒ "9.9".]	193:31
[Editor: Move [183:16-36] to replace the subclause heading and this paragraph. This paragraph is useless and repetitive introductory waffle, so it's not a loss to just replace it.]	203:3-7
[Editor: Delete "(9.5.3)" twice.]	203:22,37
[Editor: Delete "(9.5.3)" twice.]	204:5
[Editor: Delete "(9.5.3)" thrice.]	204:16-17

9.9.3 Error conditions and the ERR= specifier 203:36-38

If an error condition occurs during execution of a data transfer the position of the file becomes indeterminate. [This is here because it occurs independently of whether the statement contains an ERR= or an IOSTAT= specifier. The lack of "statement" after "transfer" is intentional, to cover the case of an error condition occurring during a wait operation.]

If an error condition occurs during execution of an input/output statement that contains neither an ERR= specifier nor an IOSTAT= specifier, execution of the program is terminated. If an error condition occurs during execution of an input/output statement that contains either an ERR= specifier or an IOSTAT= specifier

---

[Editor: Insert ", if any," after "list".] 203:39

---

(2) If the statement is a data transfer statement or the error condition occurs during a wait operation, all implied DO variables in the statement that initiated the transfer become undefined. 203:40

($5\frac{1}{2}$) If the statement is a READ statement or the error condition occurs in a wait operation for a transfer initiated by a READ statement, all input list items or namelist group objects in the statement that initiated the transfer become undefined. [Is the "in" part good enough to cover namelist group objects?] 204:2+

[Editor: "Execution" ⇒ "If an ERR= specifier appears, execution".] 204:3

9.9.4 End-of-file conditions and the END= specifier 204:4-6

If an end-of-file condition occurs during execution of an input/output statement that contains neither an END= specifier nor an IOSTAT= specifier, execution of the program is terminated. Otherwise if an error condition does not occur [we don't need the "If an end-of-file condition occurs during execution of an input/output statement..." part that we needed for error conditions, because end-of-file conditions can only occur during execution of input/output statements]

($1\frac{1}{3}$) All implied DO variables in the statement that initiated the transfer become undefined. 204:7+

($1\frac{2}{3}$) If the statement is a READ statement or the end-of-file condition occurs in a wait operation for a transfer initiated by a READ statement, all input list items or namelist group objects in the statement that initiated the transfer become undefined.

[Editor: "input" ⇒ "input/output" twice to cover the case of an end-of-file condition occurring during a wait operation.] 204:10,12

[Editor: "Execution" ⇒ "If an END= specifier appears, execution".] 204:14

9.9.5 End-of-record conditions and the EOR= specifier 204:15-17

[This needs to be checked carefully because the material from which it is taken at [183:37-42] and [1884:7-8] is difficult to follow, and may be contradictory.]

If an end-of-record condition occurs during a data transfer initiated by a nonadvancing input statement, the statement in which the condition is detected has neither an EOR= nor an IOSTAT= specifier, and the pad mode has the value NO, execution of the program is terminated. Otherwise if neither an error nor an end-of-file condition occurs

---

($2\frac{1}{2}$) All implied DO variables become undefined. 204:21+

[Editor: "input" ⇒ "input/output" twice to cover the case of an end-of-record condition occurring during a wait operation.] 204:23,25

[Editor: "Execution" ⇒ "If an EOR= specifier appears, execution".] 204:29

[Editor: Move [203:8-35] to here.] 204:29+

[Editor: "9.5.3" ⇒ "9.9".] 384:24

---

3 Don't know what to do

The term "permanently" is a bit extreme.	169:9
This constraint is rather fatuous, given that one can do a nonadvancing asynchronous read, and then put an EOR= specifier in a WAIT statement.	1777:19
The explanation that a variable may be a pending input/output storage sequence affector in some situations and not others is schizophrenic. First it discusses different scoping units, and then different instants of time. Which is it?	179:27-29
Appears to contradict [183:37-42]. This paragraph is deleted by edits proposed in section 2.	184:7-8
Is there a set of specifiers that is allowed for inquire by name but not inquire by unit, or vice-versa? If so, a summary here would be useful.	197:49+
Should *io-implied-do variable* be *io-implied-do-object*?	205:2,5

4 Very Minor Technical Enhancement

[There is no obvious reason why backspacing over records written using list-directed or namelist formatting is prohibited. Records are perfectly well defined in these cases. The problem is that one might not know where a record boundary is. But this also applies in the case of "ordinary" formatted output that has slash edit descriptors or in which format reversion takes place.] 195:31

NOTE 9.49$\frac{1}{2}$

Backspacing over records written by list-directed or namelist formatting should only be done with extraordinary care. The positions of record boundaries in output produced by list-directed or namelist formatting are not specified by this standard.