Issue 127 - End-of-File in Formatted Stream Files

To: J3
From: Craig Dedo
Date: March 22, 2001
Subject: Issue 127 - End-of-File in Formatted Stream Files

Issue

I'm not convinced that end-of-file conditions are fully covered for formatted streams. Note that there is no endfile record in a formatted stream (and I doubt we want there to be one). A strict reading of the 2nd sentence of 9.2.3.2 would tell me that it didn't apply because the endfile wasn't a result of reading an endfile record, but that's subtle. I'd suggest explicitly adding something about sequential; didn't do that myself in case someone thinks that this should apply.

What happens when reading a partial record at the end of a file? We say that there may be partial records, but I don't see where we ever say what the effects of such a thing are. If there is a partial record at the end of a file, is it possible to position after it so that it is the previous record? Should, perhaps, reading past the end of a partial record be an error instead of an EOF or EOR condition? Does padding apply to partial records? Some of these questions are probably best answered elsewhere than in 9.2.3.2, but I'll lump them all into one J3 note.

Analysis

This analysis makes use of the ideas developed in paper 01-119, "Design Considerations for Stream I/O".

It appears that the existing normative text of the Fortran 2000 draft already answers these questions. It may be appropriate to add some notes in some places in order to make the existing text clear.

Part of this issue is addressed by Interpretation 24. This paper assumes that Interpretation 24 is approved as proposed in paper 01-158. This paper adds the edit of Interpretation 24 to the Fortran 2000 draft.

Following is a separate analysis of each question in this issue.

Q1. What happens when reading an endfile record while using formatted steam I/O?

A1. The second sentence of the current version of 9.2.3.2 starts out with, "For sequential access on input, . . .". Apparently, the sentence that the first paragraph of this issue refers to is now the third sentence of this section, which starts with , "Otherwise, . . .". From the context of the entire section 9.2.3.2, it is clear that the second and third paragraphs [165:31-36] apply only to sequential access on input. The section then goes on in subsequent paragraphs to discuss sequential access on output, direct access, and stream access, in that order. Thus, this sentence [165:32-33] refers to sequential access on input and does not apply to formatted stream input. The rules for file positioning in stream access are in [165:43-44].

By definition, an endfile record is immediately after the last data record of the file. Thus, it is after the terminal point of a file, whether that file is connected for sequential or stream access. Thus, a program which reads a file that is opened for formatted stream access will never reach the endfile record; it will reach the terminal point and generate an EOF condition before it ever reaches the endfile record.

Q2. What happens when reading a partial record at the end of a file?

A2. Reading a partial record has the same effect as reading a complete record. The only difference between a partial record and a complete record is that the partial record does not have an end-of-record marker after it.

Q3. If there is a partial record at the end of a file, is it possible to position after it so that it is the previous record? Should, perhaps, reading beyond the end of a partial record be an error instead of an EOF or EOR condition?

A3. Yes. Section 9.2.3, "File position", states [165:18-19], "For a file connected for stream access, the file position is either between two file storage units, at the initial point of the file, at the terminal point of the file, or is undefined." Earlier [165:4-5], it states, "The terminal point is the position just after the last record or file storage unit."

If a program reads beyond the end of a partial record, the usual rules for reading beyond the end of record or end of file apply (9.5.3) [183:23-33]. The current language is without qualification; it does not distinguish between full and partial records.

Q4. Does padding apply to partial records?

A4. Padding applies to partial records just as it applies to full records, for the same reasons.

Edits

[165:27] In subclause 9.2.3.1, add at the end of the paragraph:
If a nonadvancing output statement leaves a file positioned within the current record and no further output statement is executed for the file before it is closed or a BACKSPACE, ENDFILE, or REWIND statement is executed for it, the effect is as if the output statement were the corresponding advancing output statement.

References

01-007, Fortran 2000 Draft
01-102, Changes to List of Unresolved Issues
01-119, Design Considerations for Stream I/O
01-140, Issue 128 - Empty Incomplete Record
01-158, Interpretation 24 - Termination of a partial record by a CLOSE, BACKSPACE, ENDFILE, or REWIND statement.
[End of J3 / 01-139r1]