WG14/N304

X3J11/93-051

1 DEC 93

Some of my review notes for the Normative Addendum neither appeared in the final document nor did I get answers from PJP that I was happy with. Plus I have spotted a few new typos.

Clive D.W. Feather

[Editorial]
Clause 1 page 2:
"new tokens k" -> "new tokens -".

[Minor]
Clause 3:
__STDC_VERSION__ has been added. However, I note it is an (int) on some platforms and a (long) on others. Posix uses an L suffix for just this reason.

[Minor]
Clause 4:
"The identifiers with external linkage declared in either <wctype.h> or <wchar.h> are reserved ..." - insert ", and are not already reserved as identifiers with external linkage by ISO/IEC 9899:1990," before the word "are".

[Minor]
Append to footnote 2: "Note that including either of these headers in a translation unit will affect other translation units in the same program, even though they do not include either header."

[Major]
Subclause 4.5.2.1: replace the wording from "each of the following eleven functions" to the end of the paragraph with the following:

> if the argument is a wide character that corresponds (as if by a call to the wctob function) to a character (byte), then each of the following eleven functions shall return the same truth value (i.e. zero or non-zero) as the corresponding character testing function from ISO/IEC 9899:1990 subclause 7.3.1.

In footnote 8, after the first sentence add the sentence:

> Similarly, if wctob(wc) is not EOF, then if iswalpha(wc) returns true then isalpha(wctob(wc)) must also return true.

In other words, if a wide character is alphabetic AND has a corresponding one-byte character, THEN the one-byte character should be alphabetic. If it isn't, it isn't a "corresponding" character.

PJP says that Japan wanted to allow iswalpha(wc) to be true but isalpha(c) to be false. However, I claim that, in this case, wctob(wc) should be EOF (or at least not be c).

[Minor]
Subclause 4.6.2.1:
It should be noted explicitly in this subclause that each stream has an associated mbstate_t object.

[Major]
There is no way to access the mbstate_t object associated with a stream - this makes it impossible to restore a specific state; for example, to associate a state with with an fpos_t object. It would be preferable to be able to obtain the address of this object, but, if this is viewed as unacceptable, fsetpos should restore the state to that at the time of the fgetpos.

PJP says that the latter option - fpos_t holds the state - has been taken, but I can't find any wording to that effect.

[Major]
Subclause 4.6.2.3.2:
In the description of the [ specifier, what is the definition of the scanset when the l qualifier is present and bytes other than single-byte characters in the initial shift state follow the [ ? Examples might make this clearer.

[Minor]
Append to footnote 14 "without the l qualifier".

[Minor]
Subclause 4.6.2.4.1:
In the example, change the two %s specifiers to %ls.

[Editorial]
Subclause 4.6.2.4.5:
There are two typos in the first paragraph of the description.

[Editorial]
Subclause 4.6.2.5.7:
"nn" -> "an"

[Editorial]
Subclause 4.6.5.4.2:
There is some kind of font problem.

[Minor]
Subclause 4.6.5.1.1:
"valid multibyte character" -> "valid single byte character" in two places.

[Minor]
Subclause 4.6.5.1.2:

Append "of that character" to the returns.

[Major]
Subclause 4.6.5.2.1:
Since this function describes a boolean test, it would better be named "isinit" or "isinitstate" - the current name is misleading as it does not even involve any wide characters.

[Major]
Subclause 4.6.5.3.2:
I made various comments, most of which have either been accepted, or I withdraw. However, I still have one issue:

If the next n or fewer bytes consist of a shift sequence followed by a zero byte (i.e. a "trailing shift sequence"), the present wording will lose the effects of the shift sequence: it will return zero, set *pwc to a null wide character, and change *ps to the initial shift state.

This means that the information in the trailing shift state is lost.

I propose that, if the next n or fewer bytes consist of a shift sequence followed by a zero byte, but s does not point to a zero byte, then instead:
- the returned value is the number of bytes in the shift sequence, excluding the zero byte
- a null wide character is stored in *pwc
- *ps represents the state immediately following the shift sequence

Note that is s is then advanced by the returned value, another call will see the zero byte, and the effects of the two consecutive calls are the same as that of a single call with the present proposal.

[Minor]
Subclause 4.6.5.4.2:
Add somewhere to the description: "If conversion stops because a terminating null character has been reached, the bytes stored include those necessary to reach the initial shift state immediately before the null byte".