<ctype.h> and <wctype.h> character classification functions

Title:	Return type of <ctype.h> and <wctype.h> character classification functions
Author:	Andrew Banks (MISRA Liaison, LDRA Ltd)
Date:	2020-06-10
Proposal for:	C2X
Document Ref:	WG14 N2541
Category:	Technical
References:	N2458, N2522

Summary

Clause 7.4.1 states that, for the character handling functions in <ctype.h>:

> The functions in this subclause return nonzero (***true***) if and only if the value of the argument **c** conforms to that in the description of the function.

For legacy reasons, the return type of these functions is *int* – an implicit Boolean, rather than an explicit one. Furthermore, the definition allows any non-zero return value, unlike eg the equality and inequality operators specifying *1* for *true*.

Given that C has supported the Boolean type since C99, it would make sense to tweak these functions to return *bool* rather than *int*.

> Note: The same applies to the functions of <wctype.h>

The consequences of having an implicit Boolean, rather than an explicit one, mean that attempting to enforce better type-checking produces unnecessary noise.

Furthermore, by returning int, there is potential for real-world confusion… eg, examples have been found where developers have (incorrectly?) used bitwise operators leading to incorrect determination:

`assert( isupper( 'B' ) && islower( 'a' ) )`	is OK
`assert( isupper( 'B' ) & islower( 'a' ) )`	fails on all checked implementations

Equally, given that the “true” return value is an indeterminant value, the return value cannot be compared with *true*:

`assert( isupper( 'A' ) )`	is OK – implicit type conversion to
`assert( isupper( 'A' ) != false )`	is OK
`assert( isupper( 'A' ) == true )`	fails on all checked implementations

Typically, implementations implement these as macros that mask the character against the characteristic being checked, returning the masked value. Casting to *bool* should be transparent to any exiting user code.

> Notes:
>
> - This paper reflects C18 as published, and is also intended to be compatible with Proposals N2458 and N2522 if adopted.
> - Newer (C11) additions already use bool (eg *atomic_is_lock_free* and the *atomic_compare_exchange_*xxx family) so this change brings consistency.

Proposed Change

Overview

Amend the narrative text of clause 7.4.1 to clarify the return value:

> The functions in this subclause return ~~nonzero (***true***)~~ if and only if the value of the argument **c** conforms to that in the description of the function.

Amend the narrative text of clause 7.30.2.1 to clarify the return value:

> The functions in this subclause return ~~nonzero (***true***)~~ if and only if the value of the argument **wc** conforms to that in the description of the function.

Function Definitions

In the code segment in the Synopsis for each of the following sections, replace ***int*** with ***bool***[1] as follows:

- 7.4.1.1.1 bool ~~int~~ isalnum (int c);
- 7.4.1.2.1 bool ~~int~~ isalpha (int c);
- 7.4.1.3.1 bool ~~int~~ isblank (int c);
- 7.4.1.4.1 bool ~~int~~ iscntrl (int c);
- 7.4.1.5.1 bool ~~int~~ isdigit (int c);
- 7.4.1.6.1 bool ~~int~~ isgraph (int c);
- 7.4.1.7.1 bool ~~int~~ islower (int c);
- 7.4.1.8.1 bool ~~int~~ isprint (int c);
- 7.4.1.9.1 bool ~~int~~ ispunct (int c);
- 7.4.1.10.1 bool ~~int~~ isspace (int c);
- 7.4.1.11.1 bool ~~int~~ isupper (int c);
- 7.4.1.12.1 bool ~~int~~ isxdigit (int c);

- 7.30.2.1.1.1 bool ~~int~~ iswalnum (wint_t wc);
- 7.30.2.1.2.1 bool ~~int~~ iswalpha (wint_t wc);
- 7.30.2.1.3.1 bool ~~int~~ iswblank (wint_t wc);
- 7.30.2.1.4.1 bool ~~int~~ iswcntrl (wint_t wc);
- 7.30.2.1.5.1 bool ~~int~~ iswdigit (wint_t wc);
- 7.30.2.1.6.1 bool ~~int~~ iswgraph (wint_t wc);
- 7.30.2.1.7.1 bool ~~int~~ iswlower (wint_t wc);
- 7.30.2.1.8.1 bool ~~int~~ iswprint (wint_t wc);
- 7.30.2.1.9.1 bool ~~int~~ iswpunct (wint_t wc);
- 7.30.2.1.10.1 bool ~~int~~ iswspace (wint_t wc);
- 7.30.2.1.11.1 bool ~~int~~ iswupper (wint_t wc);
- 7.30.2.1.12.1 bool ~~int~~ iswxdigit (wint_t wc);
- 7.30.2.2.1.1 bool ~~int~~ iswctype (wint_t wc, wctype_t desc);

[1] Use of ***bool*** as opposed to ***_Bool*** is deliberate, as this reflects the potential change in N2522

Annex B (Library Summary)

Consequentially, update the Library Summary, Annex B.3

- bool ~~int~~ isalnum (int c);
- bool ~~int~~ isalpha (int c);
- bool ~~int~~ isblank (int c);
- bool ~~int~~ iscntrl (int c);
- bool ~~int~~ isdigit (int c);
- bool ~~int~~ isgraph (int c);
- bool ~~int~~ islower (int c);
- bool ~~int~~ isprint (int c);
- bool ~~int~~ ispunct (int c);
- bool ~~int~~ isspace (int c);
- bool ~~int~~ isupper (int c);
- bool ~~int~~ isxdigit (int c);

Consequentially, update the Library Summary, Annex B.29

- bool ~~int~~ iswalnum (wint_t wc);
- bool ~~int~~ iswalpha (wint_t wc);
- bool ~~int~~ iswblank (wint_t wc);
- bool ~~int~~ iswcntrl (wint_t wc);
- bool ~~int~~ iswdigit (wint_t wc);
- bool ~~int~~ iswgraph (wint_t wc);
- bool ~~int~~ iswlower (wint_t wc);
- bool ~~int~~ int iswprint (wint_t wc);
- bool ~~int~~ iswpunct (wint_t wc);
- bool ~~int~~ iswspace (wint_t wc);
- bool ~~int~~ iswupper (wint_t wc);
- bool ~~int~~ iswxdigit (wint_t wc);
- bool ~~int~~ iswctype (wint_t wc, wctype_t desc);