WG14 N1884
INCITS PL22.11/14-004
Date: 2014-10-31
Reply To The Attention Of: Barry Hedquist
PL22.11 Secretary
Email: beh@peren.com

MINUTES (Draft)
Oct 27-31, 2014
MEETING OF ISO/IEC JTC 1 SC 22/WG 14 AND INCITS PL22.11

Meeting Location

St. Louis Union Station Hotel by DoubleTree
1820 Market St.
One Union Station,
St Louis, Missouri, 63103, USA
Phone: +1 314 621 5262
FAX: +1 314 923 3970

Meeting Information

N1861

Local Contact Information

Bill Seymour (william.a.seymour@usps.gov)
Phone: +1 314 923 2638

Scheduled Meeting Times

27 Oct 2014 09:00 – 12:00 Lunch 13:30 – 16:30
28 Oct 2014 09:00 – 12:00 Lunch 13:30 – 16:00
29 Oct 2014 09:00 – 12:00 Lunch 13:30 – 16:30
30 Oct 2014 09:00 – 12:00 Lunch 13:30 – 16:30
31 Oct 2014 09:00 – 12:00

1. Opening Activities

1.1 Opening Comments (Keaton, Seymour)

David Keaton and Bill Seymour welcomed us to St. Louis and described the meeting facilities. The meeting was hosted by Bill Seymour and ANSI.

1.2 Introduction of Participants/Roll Call

Name	***Organization***	***NB***	***Comments***
David Keaton	CERT/SEI/CMU	USA	WG14 Convener
John Parks	Intel	USA	PL22.11 Acting Chair
Daniel Plakosh	CERT/SEI/CMU	USA	
Blaine Garst	Garst	USA	
Rajan Bhakta	IBM	Canada	
Clark Nelson	Intel	USA	
Barry Hedquist	Perennial	USA	Recording Secretary
Clive Pygott	LDRA	USA	
Douglas Walls	Oracle	USA	
Tom Plum	Plum Hall, Inc.	USA	
Martin Sebor	Cisco	USA	
Fred Tydeman	Tydeman	USA	PL22.11 Vice Chair
Max Abramson	Student	USA	
Bill Seymour	Seymour	USA	
Larry Jones	Siemens PLM Software		WG14 Project Editor

1.3 Procedures for this Meeting (Keaton)

The Meeting Chair and WG14 Convener, David Keaton, announced that procedures would be as per normal. Everyone was encouraged to participate in the discussion and straw polls.

Straw polls are an informal WG14 mechanism used to determine if there is consensus to pursue a particular technical approach or possibly drop a matter for lack of consensus. Straw polls are not formal votes, and do not in any way represent any National Body position. National Body positions are established in accordance with the procedures established by each National Body.

INCITS PL22.11 members reviewed the INCITS Anti-Trust and Patent Policy Guidelines at:

http://www.incits.org/standards-information/legal-info

All 'N' document numbers in these minutes refer to JTC1 SC22/WG14 documents unless otherwise noted.

The primary emphasis of this meeting was to review the progress of our subgroups, consider proposals for new work, and work on Defect Reports.

Barry Hedquist was the Recording Secretary for the meeting.

1.4 Approval of Previous Minutes (N1820)

Several typos from were reported by various members and corrected.

The minutes were approved by unanimous consent with those changes. (Hedquist/Garst)

Final Minutes from Parma will be N1883.
Draft Minutes from St. Louis will be N1884.

1.5 Review of Action Items and Resolutions (Hedquist)

ACTION: Clark to take N1777 Part 5 to WG21.
DONE - N1857

ACTION: Clark to investigate what WG21 has done about DR 406/Core Issue 1466.
DONE – In C++ 2014

ACTION: Clark to investigate what WG21 has done about DR 407 / WG21 Library Issue 2130.
DONE - In C++ 2014.

ACTION: Blaine to write paper with proposed TC for DR 423
DONE – N1863

ACTION: Blaine to go back to Shao Miller for more input on DR 427
DONE

ACTION: Blaine to write paper on DR 431
DONE – N1864

ACTION: Benito to ask Nick for more input and a new TC for DR 437
OPEN

ACTION: Blaine to write a Proposed Committee Response to DR 442
DONE

ACTION: Blaine to write a Proposed Committee Response to DR 443

DONE

ACTION: Blaine to add link to N1804 to Committee Discussion in DR 444
DONE

ACTION: Blaine to add link to N1804 to Committee Discussion in DR 445
DONE

ACTION: Martin to provide better words for DR 450
DONE – N1873

ACTION: Blaine to write Proposed Committee Response to DR 451
DONE

ACTION: Martin Sebor to rewrite N1812/DR 461 to remove const, add allocated storage.
DONE – N1874

ACTION: Rajan to propose new words for DR 453.
DONE – N1853

ACTION: Clive to propose Secure C rule about arrays.
DONE – N1860

1.6 Approval of Agenda (N1878)

Revisions to Agenda: posted on the Wiki
Added Items: Three items, N1881, 1882, 1824

Deleted Items: None

Agenda approved by unanimous consent. (Hedquist/Garth)

1.7 Identify National Bodies Sending Experts

US, Canada.

2. Reports on Liaison Activities

2.1 SC 22 (Plum)

SC22 will meet in Hawaii in September 2015, hosted by Plum Hall / ANSI. David Keaton officially named WG14 Convener.

2.2 PL22.11/WG 14 (Parks/Keaton)

This is David Keaton's first meeting of many as Convener of WG14. Congratulations! He covered the 'new' rules for JTC1, i.e. no delegations; NB representatives are 'experts'. The NB voting process is unchanged. Convener's Report was presented at SC22. David walked us through the Livelink page for WG14.

2.3 PL22.16/WG 21 (Plum)

The C++ 2014 ballot is now closed and approved. WG21 and PL22.16 will meet in Champaign-Urbana next week. The C++14 revised standard is expected to be published in 2014.

2.4 PL22 (Plum)

Nothing relevant to WG14 to report.

2.5 WG 23 (Plum)

WG 23 is not being disbanded. Do we have a volunteer to work with WG 23 for the C Annex. Clive Pygott volunteered.

2.6 MISRA C (Pygott)

No new items

2.7 Other Liaison Reports

None

3. Reports from Study Groups

3.1 C Floating Point Activity Report (Rajan)

Parts 3 & 4 should be ready soon. Part 5 is in process.

3.2 CPLEX Activity Report (Nelson)

Working draft exists, and is in the pre meeting mailing N1862. Clark's availability precluded any discussion of this item. Deferred to next session.

4. Teleconference Meeting Reports

4.1 Report on Any Teleconference Meetings Held

5. Future Meetings and Mailings

5.1 Future Meeting Schedule

- Spring 2015 – Lysaker, Norway, 13–17 April 2015
- **ACTION Convener to provide meeting info for Lysaker in post meeting mailing.**

- Fall 2015 – Kona HI, USA, 26-30 Oct 2015

If anybody wants to host a future meeting please contact David Keaton. We are looking for a host in Europe for the Spring 2016 meeting. We have some volunteers for 2016. Spring in Canada, Markham (IBM), Fall in US, Pittsburgh (CERT) . Clive may have a venue for the UK in Spring 2016 (BSI). Move IBM to Spring 2017.

5.2 Future Mailings

- Post St. Louis: 01-Dec-2014
- Pre Lysaker : 16-Mar-2015
- Post Lysaker: 04-May-2015
- Pre Kona: 28-Sept-2015
- Post Kona: 30-Nov-2015

6. Document Review

6.1 Proposed Responses to PDTS 18661-3 Ballot Comments, [N 1868]

Rajan walked us through the proposed responses.
GB-3 – leave as is.

DECISION – Adopt NB Comments to PDTS 18661-3 , go to DTS ballot.

6.2 Proposed Responses to PDTS 18661-4 Ballot Comments [N 1869]

DECISION – Adopt NB Comments to PDTS 18661-4 , go to DTS ballot.

6.3 Alternate Exception Handling Syntax for PDTS 18661-5 [N 1841]

What syntax should be adopted to express IEEE 754-2008 alternate exception handling in C? Two basic approaches are discussed: try/catch as in C++, or #pragma STDC_CATCH_FE exception. The exception handling mechanism here is more like a 'trap' rather than exception handling used C++.

ACTION: Blaine to write up an approach for the FP Group that goes a bit beyond the approaches discussed in N1841.

Martin is concerned about introducing a 'major' new language feature to C to handle a special case. WG14 is not asking the FP group to start exploring a new mechanism for exception handling.

David pointed out exception handling has been voted down more times than classes, mostly for performance reasons. Coming up with a way to do this for FP and extending that to a general case for C, is not a direction we want to go.

The FP group plans on using #pragmas in addressing this issue because that's an existing C mechanism.

6.4 Integer Precision BIts [N1848] (Svoboda)

This paper is a proposal to have a reliable way to determine the number of bits needed for integers. This is an issue only on platforms that use padded bits because the number and value of precision bits cannot be determined from the size of the integer. The paper proposes amending the standard with macros that indicate the number of precision bits for unsigned integer types.

The Floating Point TS, part 1, has this feature. Add this to the 'features to include in the next revision of the C Standard' paper. What about extended integer types? Those are not covered in the TS. Include those as well.

ACTION: Convener to communicate our intent to include the issues discussed in N1848 into the 'future revisions' document to David Svoboda.

6.5 C11, Annex G: Corrections and Feature Requests [N1867] (Tydeman)

Three of the items here are DRs.

Paragraph 2. Annex G.6.2.1, cacosh
1. cacosh(0.0 + I*NaN) is NaN + I*pi/2 (not NaN + I*NaN)

Reasons: Mathematically, cacosh(0.0+I*y) = asinh(y) + I*pi/2. Also, C requires cacos(0+I*NaN) to be pi/2+I*NAN, which along with the mathematically identity cacosh(z) = +/-I * cacos(z), means cacosh(0.0 + I*NaN) is NaN + I*pi/2.

Paragraph 7. Annex G.6.2.6, ctanh
1. ctanh(+0.0+I*NaN) is 0.0 + I*NaN (not NaN+I*NaN)
2. ctanh(+0.0+I*INF) is 0.0 + I*NaN w/ invalid (not NaN+I*NaN w/ invalid)

Reason for above two: Since ctanh(x+I*y) = (sinh(2x) + I*sin(2y)) / (cosh(2x) + cos(2y)), for any rational number y, cos(2y) cannot be exactly -1, so no 0/0, so no NaN for the real component of the result.

All of the others are new extensions identifying special cases that Fred believes complete the intent of Annex G. There is no action for us to take on these items at this time.

ACTION: Fred to write up a DR for the three items above. DONE DR 471 (N1886)

6.6 Proposed new rule for TS 17961 [N1860] (Pygott)

This proposal started as a DR for TS 17961, rule 5.21. The solution proposed did not meet the intent of the rule. N1860 does.

Clive updated N1860 as follows:

Replace the rule 5.21 statement:

"Any attempt to use this array in a manner that causes its array bound to be violated shall be diagnosed"

with

"Any allocation where N == 0 shall be diagnosed (i.e. where n < sizeof(T)). Also, any attempt to use this array in a manner that causes its array bound to be violated shall be diagnosed.

Replace the single example with: (example 2 is the previous example)

Example 1:

```
struct S1 {
        unsigned int x;
        float        y;
        struct S1    *z;
};


struct S1 *f1(void) {
        struct S1 *p = (struct S1*)malloc(sizeof(p));  // diagnostic required
                                                       // sizeof(struct S1) > sizeof(struct S1 *)
        return p;
}
```

Example 2:

```
wchar_t *f2(void) {
        const wchar_t *p = L"Hello, World!";
        const size_t n = (wcslen(p) + 1); // n == 14
        wchar_t *q = (wchar_t *)malloc(n);
        wcscpy(q, p); // diagnostic required
                      // q is treated as wchar_t q[7];
                      // but 14 character are to be copied
        return q;
}
```

ACTION: Blaine to update TS 17961 DR Document to reflect updated N1860.

Note: CSCR DR 1 as the numbering convention.

Leave **OPEN**

6.7 Adding Classes to C [N1875] (Abramson)

The notes here apply to Items 6.7, as well as 6.8 and 6.9 below. Proposal to add C++ style classes to C. Max Abramson gave a presentation on adding features to C. Methods to structures, access specifiers, and single chain inheritance +link. College grads today are focused on object oriented programming.
Today, the basic workaround is to use structures in place of classes. Java, C# and C++ are becoming too hard to learn. The issue revolves around inheriting someone

else's code, rather than using features of a language well understood by the programmer. Maintenance and reuse of legacy code.

In terms of what this committee can do with this, the proposed features are new to C, and would require a revision to the Standard. As of now, there are no C implementations that have these features. That suggests an approach of developing a TS. What is the interest of implementers to do so? Does the C Community want a C language with OOP features? These features will work with gcc, clang, right now. Relatively clean, easy to implement.

Are there any C/C++ compatibility issues? None have been found. The goal is to be compatible with a subset of C++.

Are we interested in seeing more work on these topics in six months?

Straw Polls – Would we be interested in:

1. Add namespaces in C: 1-5-lots: No

2. Member functions: 1-6-lots: No

The straw polls above indicate there is no interest by the Committee to persue further development of the proposal presented. However, we would encourage the submitter to develop an implementation that gains user experience with the concepts presented, i.e. Modification of gcc or clang.

6.8 Access specifiers for structures in C. [N1876] (Abramson)

Proposal to add C++ style access specifiers for data members (instance variables) of C structures.

See 6.7 above

6.9 Single chain plus link inheritance for C. [N1877] (Abramson)

Proposal to add single chain inheritance to C structures.

See 6.7 above.

6.10 CPLEX: Extensions for Parallel Programming [N1862] (Nelson)

6.11 Lock Ownership vs. Thread Termination [N1881] (Riegel)

See Sec 7.2, POSSIBLE DEFECT REPORTS., Item #8.

6.12 mtx_trylock should be allowed to fail spuriously [N1882] (Reigle) (Boehm)

See Sec 7.2, POSSIBLE DEFECT REPORTS., Item #9

6.13 N1824, ATOMIC_VAR_INIT (Garst)

This document was withdrawn by the author.

6.14 N1866, thread safety of set_constraint_handler_s, (Sebor)

Martin presented an issue with the existing function **set_constraint_handler_s()** (TR 247310). This was initially presented as a proposed defect report, but the Committee decided it is a new feature rather than a DR.

Douglas asked if anyone can make use of the existing function. In general, the answer seems to be NO. We would like Martin to further develop a resolution to solving the issues presented.

7. Defect Reports

7.1 Discussion of the Defect Report Process

There was discussion about the minutes from our Defect Report sessions. Some felt it was important that they capture committee sentiment and not fine details of the conversation, lest they discourage people from freely expressing their opinions and changing their minds for fear of being viewed as inconsistent.

David reviewed some aspects of our current process:

- Any time a DR is changed it moves back to Open. During the next meeting, if it isn't changed, it may move from Open to Review. The meeting after that, if it isn't changed, it may move from Review to Closed.

- DRs can be submitted by: national bodies, the Project Editor, or the Convener. David noted, however, that we do not want to stifle input and generally treat any defect as a DR.
- When the committee changes its mind on a Technical Corrigendum, it replaces the existing words "below the line" and does not, in general, save the history. There has been no need for that complexity.
- Proposed TC's that come from the committee are generally presented as separate documents. They are not written directly into the Defect Reports.

For this meeting, Blaine chaired the DR session.

DRs that apply to Technical Specifications do not get added to the DR log. They are added to TS Specific Standing Document in the same format as a normal DR, and processed in the same way.

7.2 ISO/IEC 9899:2011 Defect Reports

POSSIBLE DEFECT REPORTS

1. Possible Defect Report: Clarifying the Behavior of the #line Directive [N1842]

   Accepted DR 464

2. Possible Defect Report: Fixing an inconsistency in atomic_is_lock_free [N1847]

   Accepted DR 465 (C++ Compatibility Issue)

3. Possible Defect Report: Scope of a for loop control declaration [N1865]

   Accepted DR 466 (C++ Compatibility Issue)

4. Possible Defect Report: Thread safety of set_constraint_handler_s [N1866]

   NAD – This was never an intended feature. Move to Document Review.

   Oct 2014

5. Possible Defect Report: Maximum (normalized) numbers [N1870]

   Accepted DR 467

6. Possible Defect Report: Epsilon numbers [N1871]

Accepted – Add to DR 467

7. Possible Defect Report: strncpy_s clobbers buffer past null [N1872]

   Accepted DR 468

8. Possible Defect Report: Lock ownership vs. thread termination past null [N1881]

   Accepted DR 469 (C++ Compat issue?)

9. Possible Defect Report: trylock semantics [N1882]

   Accepted DR 470

Discussion of Defect Reports in REVIEW Status

DR 413 – REVIEW

Moved to CLOSED

DR 416 – REVIEW

Moved to CLOSED

DR 424 – REVIEW

Moved to CLOSED

DR 426 – REVIEW

Moved to CLOSED

DR 429 – REVIEW

Moved to CLOSED

DR 433 – REVIEW

Moved to CLOSED

DR 434 – REVIEW

Moved to CLOSED

DR 435 – REVIEW

Moved to CLOSED

DR 436 – REVIEW

Moved to CLOSED

DR 441 – REVIEW

Moved to OPEN

ACTION: Fred to write a new Proposed TC, DR 441. DONE

Oct 2014 Discussion:

Words for Committee discussion on DR 441 (Tydeman)

"The committee regards the existing definition of FLT_ROUNDS as intended to apply to the three types: float, double and long double. However, if all three types cannot support the same set of rounding modes, the implementation needs to set FLT_ROUNDS to -1 meaning indeterminable. This provides very little information to the programmer.

As has been pointed out, in Annex F, only the types float and double need be IEC 60559 types. If long double is not an IEC 60559 type (for example, a pair of doubles), it may not support the same set of rounding modes as float and double. We assume that is the issue you have encountered in real implementations [a committee member has also run into this problem]. In this case, having FLT_ROUNDS apply to float and double (but not long double) would result in a value of 0, 1, 2, or 3 and would provide useful information to the programmer.

It is possible that restricting FLT_ROUNDS to apply to just the float type would cause some existing implementations to change from -1 to one of 0, 1, 2, or 3, and that would be of benefit to programmers."

There was no consensus to adopt the words above, so Blaine offered to rewrite them.

ACTION: Blaine to reword the Proposed Committee Response for DR 441.

Leave **OPEN**

DR 446 – REVIEW

Moved to CLOSED

ACTION: Convener to add the issues in DR 446 to the future revisions document.

DR 447 – REVIEW

Moved to CLOSED

DR 448 – REVIEW

Moved to OPEN

ACTION: Douglas to rewrite proposed TC for DR 448, also add TC to Annex J.2. - DONE

Proposed Technical Corrigendum

Add new paragraph 6.10 paragraph 9:

The execution of a non-directive preprocessing directive results in undefined behavior.

Add to annex J.2:

The execution of a non-directive preprocessing directive (6.10)

Discussion of Defect Reports in OPEN Status

DR 406 – OPEN Visible sequences of side effects are redundant

Clark Nelson was asked to check on the status of WG21 Core Issue 1466. A check of the DIS for C++2014 shows the proposed wording was approved and incorporated into C++2014 as shown below. Are these words applicable to C?

C++2014:

10.1 [intro.multithread]

16 The value of an atomic object M, as determined by evaluation B, shall be the value stored by some side effect A that modifies M, where B does not happen before A. [Note:

The set of such side effects is also restricted by the rest of the rules described here, and in particular, by the coherence requirements below. —end note]

22 [Note: The value observed by a load of an atomic depends on the "happens before" relation, which depends on the values observed by loads of atomics. The intended reading is that there must exist an association of atomic loads with modifications they observe that, together with suitably chosen modification orders and the "happens before" relation derived as described above, satisfy the resulting constraints as imposed here. —end note]

25 [Note: Compiler transformations that introduce assignments to a potentially shared memory location that would not be modified by the abstract machine are generally precluded by this standard, since such an assignment might overwrite another assignment by a different thread in cases in which an abstract machine execution would not have encountered a data race. This includes implementations of data member assignment that overwrite adjacent members in separate memory locations. Reordering of atomic loads in cases in which the atomics in question may alias is also generally precluded, since this may violate the coherence rules. —end note]

29.3 [atomics.order]

3 There shall be a single total order S on all memory_order_seq_cst operations, consistent with the "happens before" order and modification orders for all affected locations, such that each memory_order_seq_cst operation B that loads a value from an atomic object M observes one of the following values:

— the result of the last modification A of M that precedes B in S, if it exists, or

— if A exists, the result of some modification of M that is not memory_order_seq_cst and that does not happen before A, or

— if A does not exist, the result of some modification of M that is not memory_order_seq_cst.

[Note: Although it is not explicitly required that S include locks, it can always be extended to an order that does include lock and unlock operations, since the ordering between those is already included in the "happens before" ordering. —end note]

N1856 submitted by Clark has a Proposed Technical Corrigenda. Add Clark's proposed words to the DR, and discuss when we are able.

Leave OPEN

DR 407 – OPEN memory_order_seq_cst fence sequencing rules

C++2014:

Clause 29.3, paragraph 7 & 8.

7 For atomic modifications A and B of an atomic object M, B occurs later than A in the modification order
of M if:

*— there is a **memory_order_seq_cst** fence X such that A is sequenced before X, and X precedes B in S,*
or

*— there is a **memory_order_seq_cst** fence Y such that Y is sequenced before B, and A precedes Y in*
S, or

*— there are **memory_order_seq_cst** fences X and Y such that A is sequenced before X, Y is sequenced before B, and X precedes Y in S.*

*8 [Note: **memory_order_seq_cst** ensures sequential consistency only for a program that is free of data races and uses exclusively **memory_order_seq_cst** operations. Any use of weaker ordering will invalidate this guarantee unless extreme care is used. In particular, **memory_order_seq_cst** fences ensure a total order only for the fences themselves. Fences cannot, in general, be used to restore sequential consistency for atomic operations with weaker ordering specifications. —end note]*

Add Clark's Proposed TC to the DR, discuss when able.

Leave **OPEN**

DR 423 – OPEN under specification for qualified rvalues

Oct 2014

ACTION: Blaine to write paper with proposed TC for DR 423

Leave **OPEN**.

DR 427 – OPEN Function Parameter and Return Value Assignments

Oct 2014:

Blaine proposes to adopt the words contained in the Committee Discussion, Oct 2013. No objection.

Leave OPEN.

DR 431 – OPEN atomic compare exchange: what does it mean to say 2 structs compare equal?

Oct 2014:

Blaine submitted N1864 with the following Proposed Technical Corrigenda.

Append to 7.17.7.4 p 2

When these operations are applied to pointers to objects of atomic struct or atomic union type the behavior is undefined.

Tom objects to adopting these words because they differ from what C++ is doing. Douglas suggests these words are narrow enough to not affect C++. Martin believes the proposed TC does not go far enough. The note 1 at 7.17.7.4 is intended for implementations w/o padding bits.

After further discussion, and examining the words in C++, it's clear there are issues we did not consider. What does it mean to compare structures? WE need to shift our focus to align with C++ on that issue. C++ allows structure comparison (in many cases) and cites a { pointer, counter } lock-free example where its useful to solve ABA issues. C++ is worded to absolutely require bit comparison, C11 compares values.

ACTION: Douglas to examine the issue and come up with a Proposed TC for DR 431.

Leave OPEN

ACTION: Blaine to open a DR to clarify the memcmp reference in note 1 in 7.17.7.4; para 3.

DR 437 – OPEN clock overflow problems

The committee felt like it needed input from Nick before proceeding. Leave **OPEN**.

Oct 2014:

We have no further input from Nick

ACTION: Martin to contact Nick for further input for DR 437

Leave OPEN

DR 438 – OPEN ungetc/ungetwc file position after discarding push back

Oct 2014:

Move to REVIEW

DR 439 – OPEN Issues with the definition of "full expression"

The committee is waiting for more input from Clark. Leave **OPEN**.

Oct 2014: Defer until Clark is available.

DR 440 – OPEN Floating-point issues in C11 from PDTS 18661-1 UK review, Issue 1

Oct 2014:

Moved to REVIEW

DR 442 – OPEN Floating-point issues in C11 from PDTS 18661-1 UK review, Issue 3

N1804 from Blaine addresses this DR.

Committee sentiment was that this is not a defect and the normative requirements relative to Annex F are clear enough. Blaine offered to write a Proposed Committee Response that says that. Leave **OPEN**.

Oct 2014:

Moved to REVIEW

DR 443 – OPEN Floating-point issues in C11 from PDTS 18661-1 UK review, Issue 4

The committee agreed that the FPE (floating-point environment) is not an object but they were uncomfortable with moving footnote 205 into normative text. The sense was that there was no real need to define FPE more formally.

The committee had some sympathy with bullet 3 in the Committee Discussion (the standard does not formally define "system variable") but they had no proposed words to consider. In the end, Blaine offered to write a Proposed Committee Response conveying the sentiment that this is not a defect. Leave **OPEN**.

Oct 2014:

Proposed Committee Response exists.

Moved to REVIEW

DR 444 – OPEN Issues with alignment in C11, part 1

Joseph Myers provided a suggested TC in N1804 and the committee believes it will work.

If the committee were to adopt those changes there would be no supported way to apply _Alignas to non-aggregates. It would become a non-portable extension. Most on the committee believed that was acceptable. Some were skeptical. In the end, the committee decided to simply add a link to N1804 to the Committee Discussion and leave this **OPEN**.

Oct 2014:

ACTION: Larry to review Proposed TC, DR 444.

DR 445 – OPEN Issues with alignment in C11, part 2

Joseph Myers discusses this in N1804 as well. The committee took no action on this. Leave **OPEN**.

Oct 2014:

The issue is there is no definition for 'fundamental alignment'. Tom notes that this issue applies to C++, using a different source representation but same concepts. C++ adopted the '[[-]]' attribute designation, C did not, however the system of alignment is meant to be compatible. We'll copy-paste this into a Proposed TC, Tom/Martin will present to WG21.

ACTION: Martin Sebor to show WG21 DR 445/Proposed TC.

Leave **OPEN**

DR 449 – OPEN value of TSS_DTOR_ITERATIONS for implementations with no max

The committee agreed that the standard does not define this value intentionally and Douglas agreed to provide words to that affect.

Leave **OPEN**.

Oct 2014:

Proposed response exists.

Moved to **REVIEW**

DR 450 – OPEN tmpnam_s clears s[0] when maxsize > RSIZE_MAX

The committee agreed with the sentiment of the DR but wanted the overlong sentence in the Suggested Technical Corrigendum broken into parts to make it more readable.

Oct 2014:

N1873 - rewrite did not really improve the suggested TC.

ACTION: Larry to write a Proposed TC for DR 450 - DONE

Proposed Technical Corrigendum

> *Change K.3.5.1.2 paragraph 8 (the Returns section of tmpnam_s) to read:*
>
> *If no suitable string can be generated, or if there is a runtime-constraint violation, the tmpnam_s function writes a null character to s[0] (only if s is not null and maxsize is **both** greater than zero **and not greater than RSIZE_MAX**) and returns a nonzero value.*

The words above have conditions within conditions, rewording it is more verbose, but clearer. Make them bullet items.

ACTION: Larry to reword Proposed TC for DR 450 as bullet items. - DONE

Change K.3.5.1.2 paragraph 8 (the Returns section of tmpnam_s) from:

If no suitable string can be generated, or if there is a runtime-constraint violation, the tmpnam_s function writes a null character to s[0] (only if s is not null and maxsize is greater than zero) and returns a nonzero value.

to:

If no suitable string can be generated, or if there is a runtime-constraint violation, the tmpnam_s function:

- if s is not null and maxsize is both greater than zero and not greater than RSIZE_MAX, writes a null character to s[0]
- returns a nonzero value.

Accepted as a Proposed TC.

Leave **OPEN**

DR 451 – OPEN Instability of uninitialized automatic variables

Oct 2014

Proposed response exists.

Moved to **REVIEW**.

DR 452 – OPEN Effective Type in Loop Invariant

Oct 2014:

ACTION: Blaine to write up a clarification of the issues with DR 452 – DONE

Larry: This is why we tried to say very little about this. We've always intended these non-L value objects to be dark and mysterious. Don't mess with them.

Tom: C++ also has temporary objects, and whatever we decide w.r.t. this issue needs to be compatible with what C++ says. (COMPATIBILITY ISSUE).

ACTION: Martin to examine what C++ says about temporary objects (object with a temporary lifetime). RE: DR 452

Larry suggested that we define temporary object, and augment effective type with words to say that temporary objects behave as if they were declared with their type, and also that they need not have a unique address.

ACTION: Blaine to update DR 452 as needed.

Leave **OPEN**

DR 453 [N1853] OPEN - Atomic flag type and operations (Tydeman)

Oct 2014:

N1853 was submitted by Rajan with a Proposed TC.

More work needs to be done to clarify the conversion to _Bool, or define it in some way.

ACTION: Blaine to work with Rajan to rewrite the Proposed Technical Corrigenda for DR 453 – DONE – See Below

Proposed Technical Corrigendum

Replace

7.17.8.1 The atomic_flag_test_and_set functions
#3: Atomically, the value of the object immediately before the effects.

with:

7.17.8.1 The atomic_flag_test_and_set functions
#3: Returns false if flag value immediately before this function was called was clear, otherwise returns true.

After discussion, more work seems to be needed.

ACTION: Blaine to rework the Proposed TC for DR 453

DR 454 [N1824] OPEN - ATOMIC_VAR_INIT (issues 3 and 4) (Tydeman)

Oct 2014:

No objection to Proposed Committee Response from Parma.
Moved to **REVIEW**.

DR 455 [N1857] OPEN - ATOMIC_VAR_INIT (issue 5) (Tydeman)

Oct 2014:

N1857 was submitted by Clark Nelson that argues the requirement cited in 7.17.2.1, paragraph 2, should be omitted.

Rajan believes we should keep the requirement. Hans would like us to keep this as is, so C++ can make some changes. This is a C/C++ COMPATIBILITY ISSUE, that C++ may want to adopt. Blaine believes there may be implementation issues on PA Risc, but will check on it. David believes PA Risc is not a problem.

Adopt the April 2014 Committee Discussion as the Proposed Committee Response.

Leave **OPEN**

DR 456 [N1798] OPEN - UINT*N*_C(*value*) macro (Rajan)

Oct 2014:
Adopt the April discussion as the Proposed Committee Response.

Leave **OPEN**

DR 457 [N1802] OPEN - *asctime_s* (Keaton)

Oct 2014:
We have a Proposed TC.
Moved to **REVIEW**

DR 458 [N1806] OPEN - ATOMIC_XXX_LOCK_FREE macros (Sebor)

Oct 2014
Moved to **REVIEW**

DR 459 [N1807] OPEN - *atomic_load* functions missing const qualifier (Sebor)

Moved to **REVIEW**

DR 460 [N1808] OPEN - *aligned_alloc* underspecified (Sebor)

Oct 2014
We have a Proposed TC.
Moved to **REVIEW**

DR 461 [N1874] OPEN - problems with references to objects in signal handlers (Sebor)

Oct 2014:

Martin Sebor submitted N1874, which presents a case that allowing access to const qualified objects is not a new feature unless the Standard unambiguously prevents it. Since the he believes Standard is ambiguous, the Standard could be interpreted to allow such access.

A "Suggested Technical Corrigendum" is included, and then referred to as "proposed".

Q: Does the Committee agree with Martin's point of view? If so, does the committee agree with the proposed TC? What does "refers to" mean? Martin scanned the Standard to see how 'refer' is used, but that does not necessarily reflect normal English usage.

Douglas: C99 had the same wording, and it was very carefully crafted. He sees this as a new feature. Rajan, Larry agree. See it as a future revision. This is not a defect.

ACTION: Blaine to write up a Proposed Committee Response for DR 461, that also explains the intent of 'refer'.

ACTION: David to add the suggested changes in DR 461 to SD 3.

Leave **OPEN**

DR 462 [N1813] OPEN - clarifying objects accessed in signal handlers (Seacord)

Oct 2014

We have a Proposed TC. Should the Proposed TC be more general?

ACTION: David to write a Proposed TC for DR 462 to be more general. – DONE, N1887

Jens Gustedt reviewed David's paper, and made suggestions for minor changes. (See SC22WG14.13468). Others have comments as well.

Adopt David's words as a Proposed TC as discussed.

ACTION: Blaine to write a Proposed TC for DR 462 based on N1887, as discussed.

Leave OPEN

DR 463 [N1817] OPEN - harmonizing left-shift with C++14 (Ballman)

Oct 2014:
We have a Proposed Committee Response. There are no proposed changes.

Moved to REVIEW.

ACTION: David to add suggested changes in DR 436 to SD 3.

NOTE: The Reference Document should be N1817 v N1584.

DR 464 [N1842] OPEN – Clarifying the Behavior of the #line Directive

This is only a clarification change to a footnote. It's an 'off by one' error. Martin pointed out that this may not be the only place that needs clarification for the use of __LINE__. The footnote uses both.
How is __LINE__ supposed to work with #line? Tom believes it's ambiguous, and can be read in two different ways. Disambiguating the clause would be a normative change, but not likely to affect many implementations. Tom does not see any point in disambiguating the clause.

ACTION: David to rewrite the Suggested TC for DR 464 as a Proposed TC to J.1. – DONE, See Below

Proposed Committee Response

6.10.4 paragraph 2 states that “The line number of the current source line is one greater than the number of new-line characters read or introduced in translation phase 1 (5.1.1.2) while processing the source file to the current token.” Note that it does not say the number of new-line characters that exist prior to the current token; it says the number of new-line characters that have been read while processing to the current token.

In the case of the #line directive of the form

#line pp-tokens new-line

there are two possible values for the number of new-line characters that have been read when processing begins on the first pp-token. In a one-pass preprocessor, the line number at the first pp-token will be the number of new-line characters that exist prior to the #line directive, because that number of new-lines will have been read. In a preprocessor that must see the entire directive before processing it, since the directive explicitly includes a new-line, the line number at the first pp-token will be the number of new-line characters that exist prior to the #line directive plus one.

Therefore, in a #line directive of the form

#line __LINE__ “filename”

there are two possible values for __LINE__, which leads to two possible values for the line number following the #line directive. Both are valid.

Proposed Technical Corrigendum

Add the following footnote to the end of 6.10.4 paragraph 5.

Because a new-line is explicitly included as part of the #line directive, the number of new-line characters read while processing to the first pp-token may be different depending on whether or not the implementation uses a one-pass preprocessor. Therefore, there are two possible values for the line number following a directive of the form #line __LINE__ new-line.

Add the following to J.1 Unspecified behavior.

The line number following a directive of the form #line __LINE__ new-line (6.10.4).

Adopt the words above to DR 464 as Proposed Committee Response and Proposed Technical Corrigenda.

Leave **OPEN**

DR 465 [N1847] OPEN – Fixing an inconsistency in atomic_is_lock_free

C/C++ COMPATIBILITY ISSUE

The DR addresses an incompatibility in atomics between C and C++. The Suggested TC makes the current C Standard read the same as C++. Additional words are needed to address lock free types.

ACTION: David to generate a new Proposed TC for DR 465. – DONE

Proposed Technical Corrigendum

Change 7.17.5.1 paragraph 2 from:

The **atomic_is_lock_free** generic function indicates whether or not the object pointed to by **obj** is lock-free.

to:

The **atomic_is_lock_free** generic function indicates whether or not atomic operations on objects of the type pointed to by **obj** are lock-free.

Change 7.17.5.1 paragraph 3 from:

The **atomic_is_lock_free** generic function returns nonzero (true) if and only if the object's operations are lock-free. The result of a lock-free query on one object cannot be inferred from the result of a lock-free query on another object.

to:

The **atomic_is_lock_free** generic function returns nonzero (true) if and only if atomic operations on objects of the type pointed to by the argument are lock-free. In any given program execution, the result of the lock-free query shall be consistent for all pointers of the same type.

Leave **OPEN**

Martin is concerned about whether or not a NULL pointer can be a valid argument. The consequence of this TC is to allow it. That point can be added to the Committee Discussion, but we prefer to not explicitly state so as part of this DR.

ACTION: Martin will provide words to clarify that a NULL pointer can be a valid argument as a consequence of the Proposed TC for DR 465.

DR 466 [N1865] OPEN – scope of a for loop control declaration

Oct 2014:
C/C++ COMPATIBILITY ISSUE

C++ made a change to the rules of for loop control, and we did not 'adjust'. In C99, we used the rules contained in the C++ Annotated Reference Manual, which were changed by C++ for C++98. Rajan also pointed out that the Suggested TC does not go far enough to be compatible with C++ today.

It's not clear that we can make this change as a DR. It's too far down the road, and the change is too large. However, the impact may not be high. GCC issues a warning, and moves on. Clang accepts the code w/o a warning.

This is an item for future consideration, add to SD3.

ACTION: Blaine to write up a Proposed Committee Response to DR 466.
ACTION: Convener to add the Suggested TC material in DR 466 (N1865) to SD3.

DR 467 [N1870] [N1871] OPEN – Mismatch between formulas and descriptions floating point

Both papers identify a mismatch between the words and the math formula.

N1870 – maximum representable finite (normalized) floating-point numbers in 5.2.4.2.2;p12

Suggested Technical Corrigendum

In 5.2.4.2.2#12, add 'normalized' between 'finite' and 'floating-point'.

Add a new paragraph:

12b The values given in the following list shall be replaced by constant expressions with implementation-defined values that are greater than or equal to those shown:

-- maximum representable finite floating-point number (footnote),

FLT_TRUE_MAX 1E+37
DBL_TRUE_MAX 1E+37
LDBL_TRUE_MAX 1E+3

(footnote): Need not be normalized.

N1871 – math formula for eplison floating-point numbers in 5.2.4.2.2;p13

Suggested Technical Corrigendum

In 5.2.4.2.2#13, add 'normalized' between 'least and 'value.

(note the change of #12 to #13 in the line above).

Accept the corrections re: normalize. Reject the new paragraph 12b. Add the new paragraph 12.b to SD3 for future consideration.

ACTION: Blaine to add a Proposed Committee Response and Proposed Technical Corrigenda for DR 467.

Leave OPEN.

DR 468 [N1872] OPEN – strncpy_s clobbers buffer past null

Oct 2014:
The words for this function need to be tightened up to meet the security intent of this function. Adopt the Suggested TC as Proposed TC. Douglas objects. His concern is preserving the value. 9-1-2. Adopted

ACTION: Blaine to adopt Suggested TC as Proposed for DR 468.

Leave OPEN

DR 469 [N1881] OPEN – lock ownership vs. thread termination

C/C++ COMPATIBILITY ISSUE

What happens if a thread terminates while owning a mutex. The author wants it to be specified as undefined behavior. The fact that nothing is stated already makes it undefined. Should it be? If so, let's say so. The question is where?

ACTION: Blaine to write a Proposed TC for DR 417, N1886, indicating the behavior of a program discussed in N1886. – DONE, see below.

A review of 7.26.4 Mutex functions while in search of an appropriate place to place the Suggested TC of DR469 revealed that additional repairs are needed.

Issue 1:

C11 Section 7.26.4.2 mtx_init function p2 states "must have one of the six values" and then enumerates only four, this is a typo from when we removed additional proposed functionality.

Proposed Technical Corrigendum

Replace "six" with "four" in 7.26.4.2p2

Issue 2:

Recursive locks have thread specific behaviors, namely ongoing success on locking an already locked mutex, as specified in 7.26.4.3p2. Each successive lock operation must also be matched with a call to mtx_unlock, yet this is not stated.

Proposed Technical Corrigendum

In 7.26.4.6 The mtx_unlock function p2
replace
The mtx_unlock function unlocks the mutex pointed to by mtx. The mutex pointed to by mtx shall be locked by the calling thread.

with

In 7.26.4.6 The mtx_unlock function p2
replace
The mtx_unlock function unlocks the mutex pointed to by mtx. The mutex pointed to by mtx shall be locked by the calling thread.

mtx_unlock

Issue 3:

From the DR, we wish to state explicitly that an operation on a mutex that remains locked after thread termination results in undefined behavior.

Proposed Technical Corrigendum

To 7.26.4.6 The mtx_unlock function, add a new paragraph after paragraph 2:
The behavior of a program is undefined if a thread terminates without unlocking a mutex that it has locked.

ACTION: Blaine to submit a new paper for DR 469.

DR 470 [N1882] OPEN - mtx_trylock should be allowed to fail spuriously

C/C++ COMPATIBILITY ISSUE

C11 does not appear to allow mtx_trylock to fail spuriously (i.e., return thrd_busy even thought the lock was not acquired, yet eventually acquire the lock if it is not acquired by any thread), but C++11 does (see 30.4.1.1/16):

> An implementation may fail to obtain the lock even if it is not held by any other thread. [Note: This spurious failure is normally uncommon, but allows interesting implementations based on a simple compare and exchange (Clause 29). -- end note] An implementation should ensure that try_lock() does not consistently return false in the absence of contending mutex acquisitions.

The Suggested TC needs to be clearer. Lift the words already contained in C++.

ACTION: Blaine to write a Proposed TC for DR 470.

DR 471 [N1886] OPEN – Complex math functions (cacosh (G.6.2.1) and ctanh(G.6.2.6)) are incorrectly specified.

This is an extraction from N1867 to address the DR items listed in that paper.

Suggested Technical Corrigendum

Add to G.6.2.1 cacosh before 4th bullet: cacosh(0.0+I*NaN) returns NaN + I*pi/2

Add to G.6.2.1 cacosh 4th bullet: "non-zero" so it reads: cacosh(x + iNaN) returns NaN + i*NaN and optionally raises the ''invalid'' floating-point exception, for finite non-zero x.

Add to G.6.2.6 ctanh before 3rd bullet: ctanh(0.0+I*INF) returns 0.0+I*NAN and raises the ''invalid'' floating-point exception.

Add to G.6.2.6 ctanh 3rd bullet: "non-zero" so it reads: ctanh(x + I*INF) returns NaN + i*NaN and raises the ''invalid'' floating-point exception, for finite non-zero x.

Add to G.6.2.6 ctanh before 4th bullet: ctanh(0.0+I*NaN) returns 0.0+I*NAN

Add to G.6.2.6 ctanh 4th bullet: "non-zero" so it reads: ctanh(x + I*NAN) returns NaN + i*NaN and optionally raises the ''invalid'' floating-point exception, for finite non-zero x.

Incorporate N1886 to DR 471.

ACTION: Blaine to write a Proposed TC for DR 471 based in the Suggested TC.

7.3 TS 17961:2013, C Secure Coding Rules (CSCR)

The item below was discussed in April 2014, Parma. No new items have been submitted.

7.3.1 Error in 5.21 example [N 1801] (Pygott)

Committee Discussion, April 2014 (Parma)

The committee decided to accept this as a Defect Report. The Committee did not give it a DR number but will instead refer to it using the N1801 number. The Committee may then republish the TR to incorporate the change.

The Committee has treated defects against the TR for Embedded C in similar fashion. For reference, N 1180 is the defect log for that TR.

8. Other Business

8.1 DTS Ballots for FP TS's Parts 3 & 4

9. Resolutions and Decisions Reached

9.1 Review of Decisions Reached (Hedquist)

Send out FP DTS 18661, Parts 3 & 4, for DTS Ballot after editorial review by the Editorial Committee.

9.2 Review of Action Items (Hedquist)

- **ACTION: Convener to provide meeting info for Lysaker in post meeting mailing.**

- **ACTION: Blaine to write up an approach for the FP Group that goes a bit beyond the approaches discussed in N1841.**
- **ACTION: Convener to communicate our intent include the issues discussed in N1848 into the 'future revisions' SD3 document to David Svoboda.**
- **ACTION: Blaine to update TS 17961 DR Document to reflect the update to N1860.**
- **ACTION: Blaine to reword the Proposed Committee Response for DR 441**
- **ACTION: Convener to add the issues in DR 446 to the future revisions document SD3.**
- **ACTION: Blaine to write paper with proposed TC for DR 423**
- **ACTION: Blaine to open a DR to clarify the memcmp reference in note 1 in 7.17.7.4; para 3.**
- **ACTION: Martin to contact Nick for further input for DR 437**
- **ACTION: Larry to review the words for a Proposed TC, DR 444.**
- **ACTION: Martin Sebor to show WG21 DR 445/Proposed TC.**
- **ACTION: Martin to examine what C++ says about temporary objects (object with a temporary lifetime). RE: DR 452**
- **ACTION: Blaine to update DR 452 as needed.**
- **ACTION: Douglas to examine the issue and come up with a Proposed TC for DR 431.**
- **ACTION: Blaine to rework the Proposed TC for DR 453**
- **ACTION: Blaine to write up a Proposed Committee Response for DR 461, that also explains the intent of 'refer'.**
- **ACTION: David to add the suggested changes in DR 461 to SD 3.**
- **ACTION: Blaine to write a Proposed TC for DR 462 based on N1887, as discussed.**
- **ACTION: David to add suggested changes in DR 463 to SD 3**
- **ACTION: Blaine to write up a Proposed Committee Response to DR 466.**
- **ACTION: Convener to add the Suggested TC material in DR 466 (N1865) to SD3.**
- **ACTION: Blaine to add a Proposed Committee Response and Proposed Technical Corrigenda for DR 467.**
- **ACTION: Blaine to adopt Suggested TC as Proposed for DR 468.**
- **ACTION: Blaine to submit a new paper for DR 469.**
- **ACTION: Blaine to write a Proposed TC for DR 470.**
- **ACTION: Blaine to write a Proposed TC for DR 471 based in the Suggested TC.**
- **ACTION: Martin will provide words to clarify that a NULL pointer can be a valid argument as a consequence of the Proposed TC for DR 465.**

10. Thanks to Host

The Committee expressed its thanks to Bill Seymour for hosting the WG14 meeting in St Louis.

11. Adjournment

Adjourned at 14:31, local time, Oct 30, 2014.

INCITS PL22.11/2014-00001
Reply To The Attention Of: Barry Hedquist
PL22.11 Secretary
Email: beh@peren.com

PL22.11 TAG Meeting Minutes (Draft)
Oct 28, 2014
St Louis, MO

Meeting convened on Oct 28, 2014, at 16:00 by PL22.11 Chair, David Keaton.

Attendees:

Voting Members:		
Name:	**Organization:** **P – Primary, A - Alternate**	**Comments**
Daniel Plakosh	CERT/SEI/CMU-A	
David Keaton	CERT/SEI/CMU-P	
Blaine Garst	Garst - P	
Rajan Bhakta	IBM - P	
John Parks	Intel - P	PL22.11 Acting Chair
Clive Pygott	LDRA - P	
Douglas Walls	Oracle - P	PL22.11 IR
Barry Hedquist	Perennial, Inc - P	PL22.11 Secretary
Tom Plum	Plum Hall, Inc. – P	
Fred Tydeman	Tydeman - P	PL22.11 Vice Chair
Jim Seymour	Seymour - P	
Martin Sebor	Cisco - P	

1. Approval of Agenda

Agenda was approved by unanimous consent. (Walls/Plakosh)

2. Approval of Previous Minutes (PL22.11/13-002)

Minutes were approved by unanimous consent. (Pygott/Garth)

3. INCITS Antitrust Guidelines and Patent Policy

We reviewed the content contained in
http://www.incits.org/standards-information/legal-info

4. INCITS official designated member/alternate information.

Be sure to let INCITS know if your designated member or alternate changes, or if their email address changes. Send contact info to Lynn Barra at ITI, lbarra@itic.org.

5. Identification of PL22.11 Voting Members (Tydeman)

See attendance list above.
11 PL22.11 voting members participated out of 13.

5.1 PL22.11 Members Attaining Voting Rights at this Meeting
None

5.2 Prospective PL22.11 Members Attending Their First Meeting
None

6. Members in Jeopardy

6.1 Members in Jeopardy for failure to return letter ballots.
Seymour

6.2 Members in Jeopardy for failure to attend meetings.
Coverity

6.2.1 Members who regained voting rights by attending this meeting.
None

6.2.2 Members who lost voting rights for failure to attend this meeting.
Coverity

6.2.3 Members who previously lost voting rights who are attending this meeting.
None

7. Procedures for Forming a US Position

We were reminded that the best time to get substantial changes into our Technical Specifications is during sub-group work or during full committee meetings, not during the ballot process.

8. New Business

New procedures coming from INCITS regarding ballot duration, appointment of IR, effect of voting on member status.

9. Next Meeting: Lysaker, Norway April 13-17, Cisco

Fall 2015: Kona, HI, USA, Oct 26-30, Plum Hall, ANSI

11. Adjournment

The meeting was adjourned at 1630 local, Oct 28, 2014 by unanimous consent (Tydeman/Garst).