WG14 N1764
INCITS PL22.11/13-002
Date: 2013-10-03
Reply To The Attention Of: Barry Hedquist
PL22.11 Secretary
Email: beh@peren.com

MINUTES (Draft)
Sept 30 – Oct 3, 2013
MEETING OF ISO/IEC JTC 1 SC 22/WG 14 AND INCITS PL22.11

Meeting Location

Renaissance Chicago Downtown Hotel
1 West Wacker Drive
Chicago, IL 60601
+1-312-372-7200
Marriott Chicago
Meeting information

N 1686 - Local contact information

Nevin Liber
Tel: +1 312 623-5420
E-Mail: nliber@drw.com

Scheduled Meeting Times:

30 September 2013	09:00 – 12:00	Lunch	13:30 – 16:30
01 October 2013	09:00 – 12:00	Lunch	13:30 – 16:00
02 October 2013	09:00 – 12:00	Lunch	13:30 – 16:30
03 October 2013	09:00 – 12:00	Lunch	13:30 – 16:30

Teleconference information:

Topic: WG 14 Oct. 2013
Date: Every 1 day, from Monday, September 30, 2013 to Thursday, October 3, 2013
Time: 9:00 am, Central Daylight Time (Chicago, GMT-05:00)
Meeting Number: ***958 277 350***
Meeting Password: ***wg14***

To join the online meeting (Now from mobile devices!)

1. Go to this meeting.
2. If requested, enter your name and email address.
3. If a password is required, enter the meeting password: ***wg14***
4. Click "Join".

- To view in other time zones or languages, please click this link.

To join the audio conference only

- To receive a call back, provide your phone number when you join the meeting, or call the number below and enter the access code.
    - Call-in toll-free number (UK): 0800-051-3810
    - Call-in toll number (UK): +44-203-478-5289
- Global call-in numbers
- Toll-free dialing restrictions are here.
- Access code: ***958 277 350***

For assistance

1. Go to help.
2. On the left navigation bar, click "Support".

To add this meeting to your calendar program (for example Microsoft Outlook)

- Follow this link.

1. Opening Activities

1.1 Opening Comments (Liber, Benito)

John Benito and Nevin Liber, DRW, welcomed us to Chicago and described the meeting facilities. Several local restaurants are within walking distance of the meeting. Lunch break will be from ***12:00-13:30.*** This meeting is hosted by ANSI and DRW. Refreshments and coffee are available during the breaks right outside the room. Lunch will be in the adjoining room.

1.2 Introduction of Participants/Roll Call

Name	***Organization***	***NB***	***Comments***
John Benito	Blue Pilot	USA	WG14 Convener
Jim Thomas	Blue Pilot	USA	
Martin Sebor	Cisco	USA	
David Keaton	CERT/SEI/CMU	USA	PL22.11 Chair
Daniel Plakosh	CERT/SEI/CMU	USA	
Tana L. Plauger	Dinkumware, Ltd	USA	
P. J. Plauger	Dinkumware, Ltd	USA	
Blaine Garst	Garst	USA	
Rajan Bhakta	IBM	Canada	HoD - Canada
Michael Wong	IBM	Canada	
Clark Nelson	Intel	USA	
John Parks	Intel	USA	

Clive Pygott	LDRA	USA	
Herb Sutter	Microsoft	USA	
Douglas Walls	Oracle	USA	HoD – USA, PL22.11 IR
Barry Hedquist	Perennial	USA	PL22.11 Secretary
Tom Plum	Plum Hall, Inc.	USA	
Bill Seymour	Seymour	USA	
Fred Tydeman	Tydeman Consulting	USA	
Nevin Liber	DRW	USA	Host
Freek Wiedijk	Radboud Univ. Nijmegen	Netherlands	
Larry Jones			WG14 Project Editor
Marc Moreno Maza	Univ. Western Ontario	Canada	
Yuzhen Xie	Univ. Western Ontario	Canada	
Robbert Krebbers	Radboud Univ. Nijmegen	Netherlands	

1.3 Procedures for this Meeting (Benito)

The Meeting Chair, John Benito, WG14 Convener, announced the procedures are as per normal. Everyone is encouraged to participate in the discussion and straw polls.

Straw polls are an informal mechanism used to determine if there is consensus within the meeting to pursue a particular technical approach or even drop a matter for lack of consensus. Participation by everyone is encouraged to allow for a discussion of diverse technical approaches. Straw polls are not formal votes, and do not in any way represent any National Body position. National Body positions are only 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.

Emphasis for this meeting is to consider defect reports and future Technical Specifications for WG14.

Barry Hedquist, PL22.11 Secretary, is the Recording Secretary for the meeting.

1.4 Approval of Previous Minutes (Hedquist) (N 1693).

Several comments for typos, etc.

Minutes were modified per editorial changes and approved by unanimous consent.

Final Delft Minutes are **N1763 (1764 for this session draft)**

1.5 Review of Action Items and Resolutions (Hedquist)

ACTION: Bill Seymour will communicate the work of WG14 on PDTS 18661, Part 3, to WG21/SG6, Numerics.
DONE

ACTION: Small editorial committee (Tydeman, Benito, Thomas, Keaton) to review the changes to N1676 made by the accepted comments contained in N1702.DONE

ACTION: Convener to submit the revised version of PDTS 18661, Part 1, (N1676) to SC22 for PDTS Ballot. (90 day Ballot) DONE

ACTION: Blain Garst to submit the material discussed in item 5, Defect Reports, Discussion of Other Items, memcpy, as a Defect Report. N1736 DONE

ACTION: Douglas Walls to write a Suggested Technical Corrigenda for DR 433
DONE N1733

ACTION: Blain Garst to write a Proposed Technical Corrigenda for DR 431
DONE

ACTION: Douglas Walls to contact the submitter of N1671 for clarification. DONE There will be further discussion on his matter.

ACTION: David Keaton to write a Proposed Technical Corrigenda for DR 413. N1749
DONE

ACTION: Convener to forward PDTS 17961, Secure Coding, to DTS Ballot after review by a small editorial committee (Roberto, Clive, Willem, Douglas, Benito, Secord, Keaton).
DONE N1760 summary of voting.

1.6 Approval of Agenda (N1693).

Revisions to Agenda: Agenda is posted to the Wiki.

Added Items: None

Deleted Items: None

Agenda approved by unanimous consent.

1.7 Identify National Body Delegations

US, Canada

1.8 Identify PL22.11 voting members

See PL22.11 TAG Minutes, following these minutes. 14 of 17 members present.

2. Reports on Liaison Activities

2.1 SC 22 (Benito, Plum)

Plenary session in Tokyo. Nothing bad happened.

2.2 PL22.11/WG 14 (Benito, Keaton, Walls)

1. WG 14 Standing Document 2 (N1716)

2. Convener's Report and Business Plan (N1728)

This is JBs next to last meeting as Convener; he will not serve another term.

2.3 PL22.16/WG 21 (Plum)

WG21 met last week in Chicago.

Digit separator item, does not directly affect us as it is a optional add on, but Tom believes that people will eventually expect it. Right now, we have no revision planned, so we do not have a formal mechanism to add digit separators at this time.

2.4 PL22 (Plum)

PL22 meets twice a year via teleconference. Next teleconference will be in June 2014. See Tom Plum, PL22 Chair, for details.

2.5 WG 23 (Benito)

WG 23 has basically completed its work. Some of the existing work in process will be published.

2.6 MISRA C (Pygott)

Publication of MISRA C:2012 was announced on Feb 26, 2013, based on C99. The PDF version of the document was duly published, as announced, on 18 March 2013 and is available for purchase from http://www.misra.org.uk/shop/buy_now.php. The printed version is now available.

MISRA C has a new Chairman: Andrew Banks, Andrew@andrewbanks.com, and chairman@miscr-c.org

2.7 Other Liaison Reports

None

3. Teleconference Meeting Reports

3.1 Report on any teleconference meetings held (Benito)

Floating point group is still having monthly teleconference, reviewing NB comments for part 1, and updates for parts 2 & 3.

4. Future Mailings

4.1 Future Meeting Schedule

- Spring 2014: Parma, Italy, April 7-11, 2014 (N1745)
- Fall 2014: St. Louis – Dates: 3-8 Nov C++, C last week of October 27-30. Bill Seymour will follow up.
- Spring 2015: Host needed. Europe, UK
- Fall 2015 – Kona??

4.2 Future Mailings

- Post Chicago: 04-Nov-2013
- Pre Parma: 10-March-2014
- Post Parma: 12-May-2014

5. Document Review

Several of the documents listed here are proposed Defect Reports (DR). Documents that were accepted as Defect Reports were give a DR number. Further discussion of the document as a Defect Report can be found in Section 6, Defect Reports.

5.1 N1660 Possible defect report: Missing constraint w.r.t. Atomic (Tydeman)

Accepted as a Defect Report? YES

See DR 434 in Section 6 for further discussion.

5.2 N1661 Possible defect report: Missing constraint w.r.t. Imaginary (Tydeman)

Accepted as a Defect Report? YES

See DR 435 in Section 6 for further discussion.

5.3 N1712 Proposed Defect Report: Runtime-constraint issue with sprintf family in Annex K (Walls)

Accepted as a Defect Report? NO – Superseded by N1733 & N1734

5.4 N1713 Request for interpretation of C11 6.8.5#6 (Wakker)

Discussion: Make a DR 436

C11, section 6.8.5 paragraph 6 reads:

An iteration statement whose controlling expression is not a constant expression,156) that performs no input/output operations, does not access volatile objects, and performs no synchronization or atomic operations in its body, controlling expression, or (in the case of a for statement) its expression-3, may be assumed by the implementation to terminate.157)

Question: to what does the *that* refers back to: to the *controlling expression* or to the *constant expression*?

See DR 436 in Section 6 for further discussion.

5.5 N1717 DTR 17961, C Secure Coding Rules (Secord)

This is a revision of N1669. It has change bars inserted.

Discussion: This document is at ITTF for publication.

5.6 N1617 DTR 17961, C Secure Coding Rules (Secord)

This is the same document as above, without change bars, and is the version sent to ISO for DTR Ballot.

5.7 N1719 Proposed Defect Report: Clock overflow problems (Stoughton)

Accepted as a Defect Report? YES

See DR 437 in Section 6 for further discussion.

5.8 N1720 Proposed Defect Report: ungetc / ungetwc and file position after discarding push back problems (Stoughton)

Accepted as a Defect Report? YES

See DR 438 in Section 6 for further discussion.

5.9 N1729 Issues with the definition of full expression (Nelson)

Accepted as a Defect Report? YES

See DR 439 in Section 6 for further discussion.

5.10 N1730 PDTS 18661-1: Floating Point issues in C11, UK Review (Myers)

Accepted as a Defect Report? YES It's really four items. The first is a new feature request, the remaining are potential DRs

See DR 440 - 443 in Section 6 for further discussion.

5.11 N1731 Issues with alignment in C 11 (Myers)

Accepted as a Defect Report? YES Looks like two DRs

See DR 444 – 445 in Section 6 for further discussion.

5.12 N1736 Proposed Defect Report: Use byte instead of character for memcmp(), memcpy(). (Garst)

Accepted as a Defect Report? YES

See DR 446.

5.13 N1738 Background on –yx and –y/v (Tydeman)

See DR 426

5.14 N1739 Possible Defect Report: Boolean from complex (Tydeman)

Accepted as a Defect Report? YES

See DR 447

5.15 N1740 Possible Defect Report: # non-directive (Tydeman)

Accepted as a Defect Report? YES

See DR 448

5.16 N1741 Summary of Voting on PDTS 18661 (Benito)

No negative votes. Comments submitted by Great Britain (BSI) and Netherlands (NEN).

5.17 N1742 UK National Body Comments on PDTS 18661 (Myers)

See N1754, Item 5.24 for further discussion.

5.18 N1744 What is the value of TSS_DTOR_ITERATIONS for implementations with no maximum? (Walls)

Accepted as a Defect Report? YES

See DR 449

5.19 N1746 MetaFork: A Metalanguage for Concurrency Platforms Targeting Multicores (Chen, Maza, Shekar)

This paper proposes a metalanguage, METAFORK, for multithreaded algorithms based on the fork-join parallelism model and targeting multicore architectures. A slide presentation was given by Marc Moreno Maza, one of the papers authors.

Slides will be included in post-Chicago mailing.

Where do we go with this paper? Have the CPLEX Study Group look at this, and join that Study Group.

5.20 N1747 Stability of uninitialized variables, (Wiedijk, Krebbers)

The real questions come down to ‘what does the Standard mean’?

1. Can an uninitialized variable with automatic storage duration (of a type that does not have trap values, whose address has been taken so 6.3.2.1p2 does not apply, and which is not volatile) change its value without direct action of the program?
2. If the answer to question 1 is "yes", then how far can this kind of "instability" propagate?
3. If "unstable" values can propagate through function arguments into a called function, can calling a C standard library function exhibit undefined behavior because of this?

The three proposed resolutions: a, b, or c.

Resolution (a)
1. no
2. not applicable
3. not applicable

Resolution (b)
1. yes

2. any operation performed on indeterminate values will have an indeterminate value as its result
3. no

Resolution (c)
1. yes
2. any operation performed on indeterminate values will have an indeterminate value as its result
3. yes, library functions will exhibit undefined behavior when used on indeterminate values (probably functions like memcpy and maybe fwrite should be immune from this)

We agree that we have a debatable set of issues, and this paper should be accepted as a Defect Report. Assigned DR 451

5.21 N1750 Implicit thrd_exit. (Ballman)

Accepted as a Defect Report? Already addressed in DR 416

See DR 416

5.22 N1751 Thread-specific storage destructor invocation (Ballman)

Accepted as a Defect Report? Already addressed in DR 416

See DR 416

5.23 N1752 tmpnam_s clears s[0] when maxsize > RSIZE_MAX (Sebor)

Accepted as a Defect Report? YES

See DR 450

5.24 N1754 Teleconference Group Responses to GB Comments on N1711 (Thomas)

Jim Thomas led a review of N1754, responses to NB comments from GB on PDTS 18661, as documented in N1754. Great Britain was the only National Body that submitted comments.

Joseph Myers submitted comments on the proposed responses to GB comments. See N1770

The comment responses will be addressed by the group producing the DTS, and made available to WG14 in a 30 day review.

5.25 N1755 Editor's Comments on N1711 (Thomas)

Jim Thomas led a review of the Project Editor's comments on N1711, PDTS 18661.

5.26 N1756 Update to N1711 with N1754 & N1755 incorporated (Thomas)

As stated above. Our goal now is to decide what the next step is for the DTS. There is one minor change to add. There is no objection to forwarding this as DTS. Small editing committee formed to check that changes are per responses to NB comments.

ACTION: Small editing group: Jim Thomas, Fred Tydeman, David Keaton, John Benito, to review N1756, and modifications as needed, for incorporation of agreed to responses to NB comments.

ACTION: Convener to forward N1756, modified as needed by the small editing group, to SC22 for DTS ballot.

5.27 N1757 Update to TS 18661-2 (Draft) (Thomas)

Slides are available on the WG14 website (N1766). Jim walked through the key changes to the draft TS as presented in the slides. What's next? Oct 10 teleconference meeting of the Study Group is scheduled. We could have a PDTS ballot done by the end of the year, and be ready to review proposed responses to ballot comments in Parma. Are there any existing implementations? There is no reference implementation; however none is required as this document is a TS.

Any objecting to getting a PDTS ready, have a 30 day WG14 review, followed by a teleconference if needed? No objection, so we will proceed with that plan.

5.28 N1758 Update to TS 18661-3 (Draft) (Thomas)

Slides are available on the WG14 website (N1765). Jim walked through these slides and answered questions as they came up. This document is not ready for PDTS, and has ongoing work.

5.29 N1762 Possible DR: Effective Type in Loop Invariant (Miller)

The definition for "effective type" does not appear to apply to non-lvalue expressions. This can cause a behavioral difference, in loops.

Accepted as a Defect Report? YES DR 452

We want more time to look at this.

5.29 N1769 Round to Narrow Issue (Nagy) (Thomas)

This paper identifies a number of issues with new operations that round a result to a narrower type in TS 18661 – Part1 (N1756). The following evaluation can actually cause two rounding errors:

r = x / y;

once in the divide, another in the assignment. Jim Thomas has a proposed solution adding new material and deleting existing text. The proposed resolution will be added to the final version of DTS 18661 and be available for review by everyone prior to the document going out for DTS ballot

6. Defect Reports (DR)

6.1 Defect Reports in Review Status

The following DRs in REVIEW Status were either moved to Closed, or left in Review, as noted.

DR 402 – REVIEW

Chicago Discussion

The Proposed Technical Corrigenda is adopted unchanged. Move to CLOSED.

DR 405 – REVIEW

The Proposed Technical Corrigenda is adopted unchanged. Move to CLOSED.

DR 407 – REVIEW

The Proposed Technical Corrigenda is adopted unchanged. Move to CLOSED.

Clark says the CWG item related to this was closed w/o C being accounted for, meaning we are further out of sync with C++ than we were.

This change has been applied to the C++ working draft:

Change 29.3 [atomics.order] paragraph 7 as indicated: *[Drafting note: Note that the wording change intentionally does also replace the term atomic operation by atomic modification]*

~~-7-~~ ~~For atomic operations *A* and *B* on an atomic object *M*, if there are memory_order_seq_cstfences *X* and *Y* such that *A* is sequenced before *X*, *Y* is sequenced before *B*, and *X* precedes *Y* in *S*, then *B* occurs later than *A* in the modification order of *M*.~~ 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*.

The corresponding paragraph reference for C is 7.17.3p11.

DR 409 – REVIEW

The Proposed Technical Corrigenda is adopted unchanged. Move to CLOSED.

DR 419 - REVIEW

The Proposed Technical Corrigenda is adopted unchanged. Move to CLOSED.

DR 425 - REVIEW

The Proposed Technical Corrigenda is adopted unchanged. Move to CLOSED.

6.2 DRs in OPEN Status

The following DRs in **OPEN** status were either moved to **REVIEW**, or left in **OPEN**, as indicated.

DR 406 - OPEN

Chicago Discussion:

CWG 1466 has not been acted on.
ACTION: Clark to flag CWG 1466 back to WG21 for resolution - DONE

These changes have been proposed for the C++ working draft:

Change 1.10 paragraph 14 as follows:

The visible sequence of side effects on an atomic object M, with respect to a value computation B of M, is a maximal contiguous sub-sequence of side effects in the modification order of M, where the first side effect is visible with respect to B, and for every side effect, it is not the case that B happens before it. The value of an atomic object M, as determined by evaluation B, shall be the value stored by some operation in the visible sequence of M with respect to B side effect A that modifies M, where B does not happen before A. [Note: It can be shown that the visible sequence of side effects of a value computation is unique given The set of side effects that a given evaluation might take its value from is also restricted by the rest of the rules described here, and in particular, by the coherence requirements below. —end note]

The corresponding paragraph reference for C is 5.1.2.4p22.

1.10p20 should be changed as follows:

[Note: The visible sequence of side effects value observed by a load of an atomic depends on the “happens before” relation, which depends on the values observed by loads of atomics, which we are restricting here. 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]

The corresponding paragraph reference for C is 5.1.2.4p24.

I think 1.10p22 should be changed as follows:

[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 "visible sequence" coherence rules. —end note]
The corresponding paragraph reference for C is 5.1.2.4p27.

I believe the 29.3p3 wording should change as follows:

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 in the visible sequence of side effects with respect to B 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 in the visible sequence of side effects with respect to B that is not memory_order_seq_cst.

The corresponding paragraph reference for C is 7.17.3p6.

Leave OPEN

DR 413 - OPEN

Committee Discussion-Delft

6.7.9 Paragraphs 17-18 specify that each designator list affects only the smallest subobject to which the designator list refers. As a result, the second clause of paragraph 19 occurs once for the greater object as a whole, filling in only those parts of the whole object that were never initialized explicitly.

There was ongoing discussion of the Proposed TC, which will result in more changes. Will not solve today, table for now.

Leave OPEN.

Delft Discussion: Willem says there are two conflicting directives in the Standard. At least seven compilers give the result as 0, six from IBM, and GCC. Rajan pointed out there would be some reluctance in making a change. However, those who are doing it 'right' need to be able to answer those who claim the 'right' implementations are 'wrong'.

Leave OPEN

Chicago, Oct 2013

From Douglas Walls, N1749

The committee discussions of the DR for three meetings has been along the lines that the intent of the standard for the example given in the DR is 42. The committee however has been unable to come up with any improvement to the words to make it any clearer. N1659 was one such attempt, that included an example.

I'm suggesting in this paper that we give a committee response to the question raised, and use the example from N1659 for a Technical Corrigendum.

Suggested Committee Response for DR 413
The value of l.t.k is 42.

Suggested Technical Corrigendum for DR 413
Add the following example to 6.7.9:

```
typedef struct {
   int k; int l;
   int a[2];
   } T;

typedef struct {
   int i;
   T t;
} S;


T x = {.l = 43, .k = 42, .a[1] = 19, .a[0] = 18 };

void f(void)
{
   S l = { 1, .t = x, .t.l = 41, .t.a[1] = 17};
}
```

The value of l.t.k is 42, because implicit initialization does not override explicit initialization.

October 2013 Discussion
See papers N1659 and N1749.
Accept last two items from N1659

Add the following to the end of paragraph 21:
Implicit initialization does not override explicit initialization.

Add the following example to 6.7.9.

```
typedef struct {
int k;
int l;
int a[2];

} T;
typedef struct {
int i;
```

```
T t;
} S;
T x = {.l = 43, .k = 42, .a[1] = 19, .a[0] = 18 };

void f(void)
{
S l = { 1, .t = x, .t.l = 41, .t.a[1] = 17};
}
```

The value of l.t.k is 42, because implicit initialization does not override explicit initialization.

Leave OPEN

DR 416 - OPEN

Committee Discussion:

Chicago 2013 - DR 416

See N1750 and N1751 for proposed changes that address this DR. The proposed changes are basically in line with what POSIX does. However, we wrote the threads material to be able to work with either Windows or POSIX, and need to be consistent with that approach. **Douglas has detailed notes of changes to proposed changes.**

ACTION: Douglas will post changes to N1750 & N1751 on the Wiki. – DONE – see below.

Proposed Technical Corrigenda for DR 416

After 7.26.5.1 paragraph 2, add:

Returning from func shall have the same behavior as invoking **thrd_exit** with the value returned from func.

Change 7.26.5.5, replace paragraph 2 with:

For every thread-specific storage key which was created with a non-null destructor and for which the value is non-null, **thrd_exit** shall set the value associated with the key to **NULL** and then invoke the destructor with its previous value. The order in which destructors are invoked is unspecified.

If after this process there remain keys with both non-null destructors and values, the implementation shall repeat this process up to **TSS_DTOR_ITERATIONS** times.

Following this, the **thrd_exit** function terminates execution of the calling thread and sets its result code to **res**.

After 7.26.6.1 paragraph 2, add the following new paragraphs:

The value **NULL** shall be associated with the newly created key in all existing threads. Upon

thread creation, the value associated with all keys shall be initialized to **NULL**.

Destructors associated with thread-specific storage are not invoked at program termination.

A call to **tss_create** from within a destructor results in undefined behavior.

7.26.6.2 paragraph 2, add the following new second sentence:

A call to **tss_delete** function results in undefined behavior if the call to tss_create which set **key** completed after the thread commenced executing destructors.

After 7.26.6.2 paragraph 2, add the following new paragraphs:

If **tss_delete** is called while another thread is executing destructors, whether this will affect the number of invocations of the destructor associated with key on that thread is unspecified.

Calling **tss_delete** will not result in the invocation of any destructors.

7.26.6.3 paragraph 2, add the following new second sentence

A call to **tss_get** function results in undefined behavior if the call to **tss_create** which set **key** completed after the thread commenced executing destructors.

7.26.6.4 paragraph 2, add the following new second sentence:

A call to **tss_set** function results in undefined behavior if the call to **tss_create** which set **key** completed after the thread commenced executing destructors.

After 7.26.6.4 paragraph 2, add the following new paragraph:

This action will not invoke the destructor associated with the key on the value being replaced.*)

*) This clarifies whether or not a destructor will be invoked for storage created after a thread has already begun executing destructors: because tss_set is an undefined operation, a value may never be associated with the storage and therefore the destructor may never be invoked.

Martin has some reservations, take up after lunch.

Martin reviewed POSIX, and is satisfied that the proposed TC is OK. Adopt the wording above, leave OPEN.

DR 421 - OPEN

Chicago – 2013

This DR has a Proposed Committee Response. Are we satisfied with it? Yes.

Move to REVIEW

DR 422 - OPEN

Chicago - 2013

Proposed CR exists, no comments. Moved to REVIEW

DR 423 - OPEN

Issue: The dealing of rvalues with qualified types is largely underspecified in all versions of the C standard. This didn't surface as a problem until C11, since until then the type of an expression was not observable but only its value.

Committee Discussion

This paper is new enough that a thorough examination of its contents has not been made. It's not clear whether it's a DR or a proposal. If implementers don't know what to do, it's a defect. We really need more time to examine this. Dave Prosser – there are inherent problems with what the Standard says now. Handling of the atomic type qualifier may be the most likely defect, if there is one.

Leave OPEN.

Delft Discussion: The Standard is not exactly clear. We know what it is supposed to say, but it does not seem to say it. Rajan knows of one group who has interpreted the Standard different from what we would expect. Clark believes that the intent of the Standard is stated as Proposal 5 in this DR. Does Proposal 5, Drop all values of rvalues, cover all we would want to say about this. Tom agrees.

ACTION: Clark Nelson to review the applicability of Proposal 5, and write a Propose Technical Corrigenda. – DONE, see below.

Chicago – 2013

Joseph Myers email, WG14 13037, 8/13/2013

> DR#423 has committee discussion with proposed changes to avoid rvalues having qualified type, so that qualifiers do not need handling when using _Generic. It also has a note "Atomic types may or may not be subject to distinct generic selection and this needs to be resolved.".
>
> The issue with atomic types applies to more than just generic selections.
> If sizeof is applied to an expression with atomic type, then as per 6.3.2.1#2 lvalue conversion does not apply, so the size returned is the size of the atomic type, which may differ from that of the corresponding non-atomic type. Thus, it matters for sizeof, and not just for _Generic, whether casts to atomic type result in an expression with that type or with the corresponding non-atomic type, and likewise it also matters for functions returning an atomic type.
>
> I think this can most sensibly be dealt with as part of the resolution of DR#423 rather than having a separate overlapping DR.
>
> Should generic selection apply to atomic types? Blaine thinks it should. Clark suggests that Blaine write that up.

ACTION: Blaine to do so. See Below.

Resolving DR 423 - Blaine Garth

At this point I'm just assuming the submitter has in fact identified all the issues where the standard is insufficiently clear that non-lvalues do not have qualified type; I haven't had time to search for search for others.
Following are the submitter's proposed edits for what he calls Proposal 5, each followed by my commentary.

6.5.1.1, modify as follows:
EXAMPLE The cbrt type-generic macro could be implemented as follows. Here the prefix operator + in the selection expression ensures that lvalue conversion on arithmetic types is performed such that e.g. lvalues of type float const select cbrtf and not the default cbrt.

```
#define cbrt(X) _Generic (+(X), \
                          long double: cbrtl,      \
                          default: cbrt,           \
                          float: cbrtf             \
                          ) (X)
```

This edit is not necessary. The controlling expression of a generic selection was very carefully not added to the list of contexts in which lvalue conversion is not done and type qualification is discarded; see 6.3.2.1p2. So the controlling expression of a generic selection cannot have qualified type. So, for example, by a careful reading of the standard, a qualified type name in a generic association can never be matched.

The submitter believes that “The intention is clearly ... to distinguish all 8 different forms of qualifications of a type”, and that it is important to be able to do so. However, he is simply mistaken about WG14's intention. (Only time will tell whether this is in fact important.)

6.5.4, add after p2: The type of a cast expression of a qualified scalar type is the scalar type without any qualifiers.

The effect of this proposed edit is desirable. My proposed edit to achieve that effect is instead to change 6.5.4p5:

Preceding an expression by a parenthesized type name converts the value of the expression to the unqualified version of the named type. This construction is called a *cast*. A cast that specifies no conversion has no effect on the type or value of an expression.

Footnote 104 should also be deleted.

6.5.2.2, add after p1: The type of a function call is the return type of the function without any qualifiers.

6.7.63, change p15, first sentence: For two function types to be compatible, the unqualified versions of both return types shall be compatible.

The combined effect of these edits is desirable: a qualified function return type doesn't make sense. However, I think it would be simpler just to remove any qualifier from the declared return type of a function; that would make the submitter's changes unnecessary. To do that, I propose to change paragraph 5:

…and the type specified for *ident* in the declaration "T D" is "*derived-declarator-type-list T*", then the type specified for *ident* is "*derived-declarator-type-list* function returning the unqualified version of *T*".

After some discussion, Blaine decided more work was needed w.r.t. atomic.

ACTION: Blaine to write up a new Suggested TC for DR 423

DR 424 - OPEN

Section 7.26.6 "Thread-specific storage functions" of C11 is severely underspecified since it uses terms that are not introduced (so far) in the context of C. This is really a pity, since POSIX also has pthread_key_t that is completely feature equivalent and for which the specification is much more complete.

Adopt resolution for DR 416, leave OPEN.

Committee Discussion

Tied to DR 416 – and 5.9 See discussion there.

Leave OPEN

Chicago 2013 – same as above, adding: The TC for DR 416 resolves the issues in DR 424.

DR 426 – OPEN

Committee Discussion

Chicago – 2013

See N7238. Fred submitted a suggestion to add to the committee discussion.

Typos in above:

- (-y)/v should be (-y)*v
- -(y/v) should be -(y*v)

Cases 1 and 2 above about NaNs are not required by either C11 or IEEE-754.

Negate is not affected by rounding, but gives the correct sign of the final result. The product or division is then done, which is affected by rounding. If you do negate after rounding, you end up with the wrong result for asymmetric rounding.

The Committee agrees that the third case in DR 426 is the driving reason for this change:

> "All operands are non-NaN, the result is inexact and non-NaN, and a rounding that is not symmetric about zero is in effect."

Proposed Technical Corrigendum

In the table in G.5.1 #2, change

-yv

to

(-y)v

in three places.

In the table in G.5.1 #3, change

-x/v

to

(-x)/v

in two places.

The above does not reflect what we really want. Several believe this is NOT a DR, and deserves a Record of Response. Accept Fred's words 12-5-1

Leave OPEN.

DR 427 – OPEN

Committee Discussion

Chicago – Oct 2013

Blaine submitted the following:

Summary

As best I understand the issues in DR427, there is a clear defect with respect to specifying the initial values of the parameters to a function call. The standard is written in terms of assignment when initialization is meant, this blatantly forbids const arguments since they cannot be assigned to (as well as aggregates containing const members).

I proposed a simple resolution to this in message 13034 on the reflector to which Larry Jones in follow up message 13035 kindly improved. Here it is:

Suggested Technical Corrigendum

In 6.5.2.2p2 change:

If the expression that denotes the called function has a type that includes a prototype, the number of arguments shall agree with the number of parameters. Each argument shall have a type such that its value may be assigned to an object with the unqualified version of the type of its corresponding parameter.

to:

If the expression that denotes the called function has a type that includes a prototype, the number of arguments shall agree with the number of parameters. Each argument shall have a type such that its value may be used to initialize an object having the type of its corresponding parameter.

In 6.5.2.2p4, change

An argument may be an expression of any complete object type. In preparing for the call to a function, the arguments are evaluated, and each parameter is assigned the value of the corresponding argument.

to:

An argument may be an expression of any complete object type. In preparing for the call to a function, the arguments are evaluated, and each parameter is initialized to the value of the corresponding argument.

Douglas: What's the diff between initialization and assignment w.r.t. conversions? David K asked that a specific example be looked at.

Add as Suggested TC, Leave OPEN

DR 428 – OPEN

Committee Discussion:

Chicago 2013

Douglas Walls has an additional suggested/proposed Technical Corrigendum:

It was pointed out at the October 2012 meeting in Portland that there were additional runtime constraints that needed corrections with respect to arguments not being greater than RSIZE_MAX.

Upon investigation, all of those runtime constraints are for wide character functions which also need to be corrected in DR 433. In Document N1733 I've provided suggested technical corrigendum that would correct those runtime constraints for both DR 428 and DR 433 using DR 433. Here is that list of functions:

K.3.9.1.3 The snwprintf_s function
K.3.9.1.4 The swprintf_s function
K.3.9.1.8 The vsnwprintf_s function
K.3.9.1.9 The vswprintf_s function
K.3.9.3.2.1 The mbsrtowcs_s function
K.3.9.3.2.2 The wcsrtombs_s function

One additional change we might consider making is to K.3.5.1.2 tmpname_s, if only so the wording is consistent with all of the other runtime constraints about arguments not being greater than RSIZE_MAX being corrected by DR 428 and 433:

K.3.5.1.2 The tmpnam_s function

K.3.5.1.2p2 replace "less than or equal to RSIZE_MAX" with "not greater than RSIZE_MAX".

However, the current wording is not defective.

A Proposed Technical Corrigenda exists. Add material to Discussion on the 'other places'.

Move to REVIEW.

DR 429 – OPEN

Committee Discussion:

Chicago - 2013:

Paper from Douglas Walls, N1748:

The original question asked in DR 429 was

The runtime-constraint violation here can be caused by a null "s" pointer.

Should we discard the next input line even if (s == NULL) ?

When I wrote DR 429, I had not taken footnote 404) into account.

404) The gets_s function, unlike the historical gets function, makes it a runtime-constraint violation for a line of input to overflow the buffer to store it. Unlike the fgets function, gets_s maintains a one-to-one relationship between input lines and successful calls to gets_s. Programs that use gets expect such a relationship.

I now believe the answer to the question I posed in DR 429 is *yes*.

The other minor issue pointed out in the DR is that that s[0] cannot be
set to the null character when s==NULL. The following correction is offered.

Suggested/Proposed Technical Corrigendum for DR 429

Annex K.3.5.4.1, replace paragraph 3 with the following:

If there is a runtime-constraint violation, characters are read and discarded from stdin until a new-line character is read, or end-of-file or a read error occurs, and if s is not a null pointer s[0] is set to the null character.

We do not have consistent behavior in the field.

Accept Proposed TC in N1748, leave OPEN.

DR 430 – OPEN

Chicago: We have an acceptable Proposed TC.

Moved to REVIEW

DR 431 – OPEN

What does it mean to say two structs compare equal?

Chicago 2013

Blaine will address Rajan's concerns regarding applicability of types. Blain will work towards normative words to make memcmp and memcpy the implementation used for atomic_compare_exchange.

Leave OPEN

DR 432 – OPEN

Chicago: We have an acceptable Proposed TC.

Move to REVIEW.

DR 433 – OPEN

Committee Discussion:

Chicago, Sept 2013:

We have a Suggested/Propose Technical Corrigendum from Douglas Walls.

Rajan has identified a number of errors in the list provided, which were reviewed. So, the Suggested TC needs to be revised. DONE. Douglas submitted a revised Suggested TC, N1771.

Leave OPEN

6.3 New DRs established at this meeting.

DR 434 – OPEN (N1660)

Missing Constraint w.r.t. Atomic. There is a constraint for atomic type specifiers in 6.7.2.4, but no similar constraint for atomic type qualifiers in 6.7.3.

Suggested Technical Corrigendum

Add to 6.7.3 Type qualifiers, a new paragraph after paragraph 3,

Atomic type qualifiers shall not be used if the implementation does not support atomic types (see 6.10.8.3).

Add to 7.16.6 Atomic integer types, a new paragraph before paragraph 1:

Constraints

Atomic type names shall not be used if the implementation does not support atomic types (see 6.10.8.3).

We don't need the second one due to 7.17 p2

Leave OPEN

DR 435 – OPEN (N1661)

6.7.2 Type specifiers, has in paragraph 3:

The type specifier _Complex shall not be used if the implementation does not support complex types (see 6.10.8.3).

But, G.2 Types, has no similar constraint with respect to _Imaginary.

This is not a defect. Annex G requires _Imaginary be supported, so there is no need to cite a requirement w.r.t. it NOT being supported.

Needs Committee Response.

DR 436 – OPEN (N1713)

Request for Interpretation

C11, section 6.8.5 paragraph 6 reads:

An iteration statement whose controlling expression is not a constant expression,156) that performs no input/output operations, does not access volatile objects, and performs no synchronization or atomic operations in its body, controlling expression, or (in the case of a for statement) its expression-3, may be assumed by the implementation to terminate.157)

Question: to what does the "that" refers back to: to the controlling expression or to the constant expression?

It refers to 'iteration statement'. Reword

ACTION: Douglas to provide rewording of a Suggested TC for DR 436. DONE See below.

Suggested Technical Corrigendum for DR 436

Replace 6.8.5p6 with:

An iteration statement may be assumed by the implementation to terminate when all of the following are true:

- its controlling expression is not a constant expression [156)]
- it performs no input/output operations
- it does not access volatile objects
- it performs no synchronization or atomic operations in its body, controlling expression or (in the case of a for statement) its expression-3. [157)]

Clark has some issues with the exact words. The word 'it' applies to more than that listed here.

ACTION: Clark will rewrite the Suggested TC for DR 436.
DONE

Suggested TC
Replace 6.8.5p6 with:
An iteration statement may be assumed by the implementation to terminate if its controlling expression is not a constant expression *156), and none of the following operations is performed in its body, controlling expression or (in the case of a for statement) its expression *157):

- o input/output operations
- o accessing a volatile object
- o synchronization or atomic operations.

Leave OPEN

DR 437 – OPEN (N1719)

clock overflow problems
Tied to Austin Group Defect #686. The basic question revolves around what happens if more processor time elapses than can be fit into a variable of type clock_t. That seems to imply that the function must fail and return (clock_t)-1. However many implementations ignore that, and simply truncate the value, returning the lowermost bits of the actual value.

Committee Discussion: The overflow issue does not seem to be discussed in the Suggested TC, and it needs more work.

Leave OPEN

DR 438 – OPEN (N1720)

ungetc / ungetwc and file position after discarding push back problems.

Ref Austin Group Defect #701, says that "or discarding" makes no sense.
Larry Jones pointed out that the Standard is correct as written because the intent is that the specified file position indicator is an intermediate state *inside* the file positioning function after the pushed-back characters are discarded but before the actual seek. That gives you a reliable file position from which to do the seek. It's not intended that the file positioning function doesn't set the file position indicator.

Needs a Committee Response rather than a Suggested Change –or- possible footnote to explain why.

Bill Seymour proposed the following Suggested TC.

Add a footnote after "pushed-back characters" in the second sentence of 7.21.7.10p5:

> After the file positioning function discards the characters but before it actually does the repositioning.

Add a footnote after "pushed-back wide characters" in the second sentence of 7.29.3.10p5:

> After the file positioning function discards the characters but before it actually does the repositioning.

After some discussion, we decided that the above words needed some editing.

ACTION: Larry Jones to reword the Suggested TC footnote for DR 438.

Leave OPEN

DR 439 – OPEN (N1729)

Point A – editorial? To Larry Jones.

Point B – must yield 0, 1, and 2, but the order is not guaranteed. No change is necessary.

Point C – The Standard is not completely clear or coherent. A Suggested TC is needed.
ACTION: Clark to write a Suggested TC for DR 439, Item C.

Point D – Expressions in abstract declarators not mentioned at all. The Standard is not clear, but do we want to say more – as in 'these are not allowed' per se, it's really undefined. The Committee agrees that these need a committee response, but no change to the Standard.

Point E – list of full expressions is complete? Make the list of 'full expressions', a non-normative note, or something equivalent. Larry prefers a footnote.

DR 440 – OPEN (N1730)

Issue 1 –Choice of long double in Annex F. This is asking for a new feature, and cannot be handled as a DR. Needs a Committee response.

DR 441 – OPEN (N1730)

Issue 2 – Definition of FLT_ROUNDS. Is the definition inadequate? Is the intent that FLT_ROUNDS applies only to type float. No. This is asking for a change that could potential change the behavior of existing implementations. The proposed change to 'F' is already covered in F.3. We see no benefit to any of the suggested changes.

DR 442 – OPEN (N1730)

Issue 3 – Floating point exceptions and 6.5#5. This probably belongs in Annex F, rather than in the main body.
ACTION: Rajan to find a suitable place in Annex F for DR 442 (N1730, Issue 3) DONE – See below.

Rajan submittal suggests the main body of the Standard, and Annex F:

Context:

6.5p5

If an exceptional condition occurs during the evaluation of an expression (that is, if the result is not mathematically defined or not in the range of representable values for its type), the behavior is undefined.

Issue 3 request:

Append:

For implementations defining __STDC_IEC_559__, this does not apply to exceptional conditions where the behavior (such as raising a floating-point exception and returning a NaN) is defined by Annex F, directly or by reference to IEC 60559.

Suggested technical corrigendum:

In Annex F:

Add in a new F.4 (incrementing later clauses):

F.4': Exceptional conditions:

If an exceptional condition occurs during the evaluation of an expression (that is, if the result is not mathematically defined or not in the range of representable values for its type), and the behavior is not defined in this annex or by reference to IEC 60559, 6.5p5 applies and the behavior is undefined.

Since Annex F has -inf and +inf, all values are in the range of representable values.

Do we really need any of this? Larry: Put at the end of F.3. Add the long double example as a footnote? Blaine will do the right thing.

DR 443 – OPEN (N1730)

Issue 4: floating-point state not being an object. What is, or is not an object? The footnote is not normative. We do want it to be normative. JM proposes to make the footnote normative. Are we creating a possible new Undefined Behavior? No. There is general favor to do something along the lines of … JM's suggestion. Leave the normative terms in 7.6 rather than moving it to 5.1.2.3 as suggested. What is a 'system variable'? Larry – it's a code word for 'not an object'.

DR 444 – OPEN (N1731)

The paper, N1731, has been divided up into two individual DRs, based on each issue presented. Issue 1: Existence of over-aligned types: No syntax is provided: That's correct. Clark originally brought the feature to the table. We could ask Joseph for words. We need a suggested TC.

DR 445 – OPEN (N1731)

Issue 2: Contexts in which alignments are supported:

'fundamental type' is not defined in C11. Is it supposed to be 'basic type'? Yes

'context' is not defined

'valid alignment' may not be a 'fundamental alignment'. Fundamental alignment needs a definition that clarifies what we meant.

There are a lot of 'nits' that need to be tied together. Joseph Myers suggests a C99 approach, but that may not be practical, except that what worked in C99 needs to still work. We could ask Joseph to come up with some wording.

We need a paper. Leave OPEN

DR 446 – OPEN (N1736)

memcpy, memmove, memcmp. This has a Suggested TC, proposing changing 'characters' to 'bytes'. Are they equivalent? We use 'character' throughout the Standard to mean may different things, so a pervasive cleanup would be needed. Larry: That's a job somebody should take on later on when in the next revision of the standard.

DR 447 – OPEN (N1739)

What is the value of:

```
_Bool b = 0.0 + 3.0 * I;
```

Suggested TC has an exclusion for _Bool. Larry: the suggested TC is the right thing to do. Bool is a 'real' type.
Agree.

DR 448 – OPEN (N1740)

What is a directive name? What are the semantics of a # non-directive?
This is deliberately underspecified, and is essentially intended. If it starts with a directive name, it is a directive, if it does not, it isn't.
Committee Response needed. Undefined behavior. However, there are 'shalls' here that are non-normative that should be removed.

Straw Poll: Do we want this to be explicitly stated as "Undefined Behavior"? 10-7-0
If somebody wants to write the words.

ACTION: John Parks to write up words for Undefined Behavior for DR448.
DONE

Suggested Technical Corrigendum for N 1740

Add new paragraph 6.10p9:

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

Change 6.10p3 from

A text line shall not begin with a # preprocessing token. A non-directive shall not begin with any of the directive names appearing in the syntax.

to

A text line is one that does not begin with a # preprocessing token. A non-directive is one that does not begin with any of the directive names appearing in the syntax.

Straw Poll Adopt the 'Add the paragraph' portion: 14-2-1 Carried

DR 449 – OPEN (N1744)

What is the TSS_DTOR_ITERATIONS macro is supposed to expand to if the implementation imposes no restriction on the number of times that destructors will be called when a thread terminates? Proposes that -1 be used to represent no restrictions.

Setting a mechanism for allowing an unlimited number could allow users to shoot themselves in the foot, but this is C, where doing so is permitted.

Setting an unlimited maximum is not existing practice, and using -1 could break existing implementations.

ACTION: Blaine to write a Suggested Committee Response for DR 449.

DR 450 – OPEN (N1752)

`tmpnam_s` clears `s[0]` when `maxsize > RSIZE_MAX`

The majority of bounds checking functions are specified to set the first element of the destination buffer, s[0], to the NUL character when a constraint violation occurs and the s pointer is non-null and the size of the buffer is greater than zero and less than or equal to SIZE_MAX.

However, the tmpnam_s function sets s[0] to NUL even when maxsize is greater than RSIZE_MAX, making its behavior on constraint violation inconsistent with the rest.

Suggested Technical Corrigendum:

Change paragraph 8 in the Returns section of tmpnam_s to read:

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

Committee Discussion: Suggested TC needs to be broken into parts for consistency.

DR 451 – OPEN (N1747)

The real questions come down to 'what does the Standard mean'?

Can an uninitialized variable with automatic storage duration (of a type that does not have trap values, whose address has been taken so 6.3.2.1p2 does not apply, and which is not volatile) change its value without direct action of the program? (question 1)

If the answer to question 1 is "yes", then how far can this kind of "instability" propagate? (question 2)

If "unstable" values can propagate through function arguments into a called function, can calling a C standard library function exhibit undefined behavior because of this? (question 3)

The three proposed resolutions: a, b, or c (based on the answers to questions 1, 2, 3)

Resolution (a)
no
not applicable
not applicable

Resolution (b)
yes
any operation performed on indeterminate values will have an indeterminate value as its result
no

Resolution (c)
yes
any operation performed on indeterminate values will have an indeterminate value as its result
yes, library functions will exhibit undefined behavior when used on indeterminate values (probably functions like memcpy and maybe fwrite should be immune from this)
SC22/WG14.13061 10/3/13 – also references a similar discussion going on in C++

Notice that in C (no idea about C++) the problem is not limited to unsigned chars, but applies to all types that cannot have trap representations. This is due to 3.19.2:

> indeterminate value
> either an unspecified value or a trap representation

For example, the questions of Freek and me in n1747 can also be applied to the following program.

```
uint32_t x[1];
printf("%d\n", x[0]);
printf("%d\n", x[0]);
```

(but 7.20.1.1, uint32_t does not allow trap representations.)

Chicago Discussion:

Tom believes that is no reason to peg these properties on unsigned char. Better to give advice that if you want special properties you should use volatile storage. Doing so is widely known. If it is not volatile, it does not matter whether or not it has trap representations.

Clark believes we have to invent a new category to completely say what we want to say, and that will not be easy.

Tom: This goes back to earlier discussions that led to the creation of Annex L. It was concluded then that a yet to be defined category was needed. Close to what is called an indeterminate value.

Clark: we need to be careful whether we are talking about values or objects. Modern optimizers track the sources of values. To resolve this, we need to address the property of a value rather than an object.

David K: Resolution C is the only one that allows us to keep the optimizations we want, as our answer, but without the Suggested TC.

A wobbly value stays wobbly whatever you do with it. There is no operation that turns a wobbly value into a stable value. Should accessing a wobbly value be Undefined Behavior?

The use of a wobbly value might or might not be considered an undefined behavior.

Martin: Very concerned about giving implementers the freedom to do 'whatever'.

Clark: we do not want the 'words' in Resolution C, but we do want the principle.

DR 452 - OPEN (N1762)

ACTION: Blaine to contact the submitter of N1762, DR 452, to prepare to answer his questions.

7. Resolutions

7.1 Review of Decisions Reached - NA

7.2 Review of Action Items

ACTION: Small editing group: Jim Thomas, Fred Tydeman, David Keaton, John Benito, to review N1756, and modifications as needed, for incorporation of agreed to responses to NB comments.

ACTION: Convener to forward N1756 to 30 day review by WG14. Modify the document as needed. Then submit to SC22 for DTS ballot.

ACTION: Blaine Garst to write up a new Suggested TC for DR 423

ACTION: Larry Jones to reword the Suggested TC footnote for DR 438.

ACTION: Clark Nelson to write a Suggested TC for DR 439, Item C.

ACTION: Blaine Garst to write a Suggested Committee Response for DR 449.

ACTION: Blaine Garst to contact the submitter of N1762, DR 452, to prepare to answer his questions.

8. Thanks to Host

The Committee expressed its thanks to DRW Holdings and Nevin Liber for hosting the WG14 meeting in the great city of Chicago and arranging for the great weather.

9. Adjournment

Adjourned at 1400, local time, Thursday, 10/3/2013

INCITS PL22.11/2013-00001
Reply To The Attention Of: Barry Hedquist
PL22.11 Secretary
Email: beh@peren.com

PL22.11 TAG Meeting Minutes (Draft)
1 Oct, 2013
Chicago, Ill (USA)

Meeting convened on October 1, 2013, at 16:15 pm by PL22.11 Chair, David Keaton.

Attendees:

Voting Members:		
Name:	**Organization:** **P – Primary, A - Alternate**	**Comments**
John Benito	Blue Pilot - P	
Jim Thomas	Blue Pilot – A	
David Keaton	CERT/SEI/CMU-P	PL22.11 Chair
Daniel Plakosh	CERT/SEI/CMU-A	
Martin Sebor	Cisco - P	
Tana Plauger	Dinkumware, Ltd – A	
P. J. Plauger	Dinkumware, Ltd – P	
Blaine Garst	Garst - P	
Rajan Bhakta	IBM - P	
Clark Nelson	Intel - A	
John Parks	Intel - P	
Clive Pygott	LDRA - P	
Herb Sutter	Microsoft - P	
Douglas Walls	Oracle - P	PL22.11 IR
Barry Hedquist	Perennial – P	PL22.11 Secretary
Tom Plum	Plum Hall, Inc. – P	
Bill Seymour	Seymour - P	
Fred Tydeman	Tydeman Consulting – P	PL22.11 Vice Chair
Non-Voting:		

1. Approval of Agenda

Revisions to Agenda:

Added Items: Added 9.1.

Deleted Items: None

Agenda approved by unanimous consent. (Garth/Rajan)

2. Approval of Previous Minutes (PL22.11/12-002)

Minutes were modified per editorial changes and approved by unanimous consent.
(Hedquist/Benito)

3. Selection and Review of US Delegation.

Motion: (Walls/Garth)

The US delegation for SC22/WG14 meetings during 2014 will be all Principals and Alternates for PL22.11 including new members and representatives that join the committee in-between PL22.11 meetings and before the next US delegation vote, with the exception of JTC 1 officers.

Approved (Unanimous Consent)

4. INCITS Anti-Trust Guidelines & Patent Policy

We viewed the slides located on the INCITS web site.

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

5. 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.

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

See attendance list above.
14 PL22.11 voting members participated out of 17.

6.1 PL22.11 Members Attaining Voting Rights at this Meeting

None

6.2 Prospective PL22.11 Members Attending Their First Meeting

None

7. Members in Jeopardy

7.1 Members in jeopardy due to failure to return Letter Ballots.

None

7.2 Members in jeopardy due to failure to attend Meetings.

HP, Bloomberg

7.3 Members who previously lost voting rights who are attending this meeting.

Seymour
Motion: Restore voting rights to Seymour (Plauger/Benito) Unanimous consent.

8. Procedures for Forming a US Position

Discussion on forming a US Position on a DTS ballot. Expect three phases:

1) Discussion and submit comments.
2) If there are comments, expect a straw poll on the comments.
3) Real ballot with comments. No new comments allowed.

9. New Business

9.1 INCITS New membership policy discussion.

Discussion of new requirement for new members to sign a release for copyright. General concern about this, legal review by large company legal firms, etc. etc.

10. Next Meeting: Parma, Italy

- Spring 2014: Parma, Italy With WG14 April 7-11
- Fall 2014: St. Louis –Dates: 3-8 Nov C++, C last week of October? Bill Seymour will follow up.
- Spring 2015: Host needed.

11. Adjournment

There being no further business, the meeting was adjourned at 1641 local, October 1, 2013 (Benito/Tydeman) - Unanimous Consent.