**WG14 N1541**
**INCITS PL22.11/10-002**
**Reply To The Attention Of: Barry Hedquist**
**Email: beh@peren.com**

**ISO/IEC JTC1 SC22/WG14 AND INCITS PL22.11**
**NOV 1 - 5, 2010 MEETING MINUTES (DRAFT)**

Meeting Sponsor: USA (ANSI ) / Fermi National Accelerator Laboratory

Meeting Location:
Comfort Inn & Suites Geneva
1555 East Fabyan Parkway
Geneva, Illinois
Phone: +1.630.208-8811
Fax: +1.630.208-7844

Host Contact information:

Walter E. Brown
FPE Dept., Computing Division
Fermi National Accelerator Laboratory
Batavia, IL 60510-0500 USA
E-mail: wb@fnal.gov

Meeting Dates: 1-5 Nov, 2010

Meeting Times:
01 November 2010 09:00–12:00 13:30–16:30
02 November 2010 09:00–12:00 13:30–16:00
03 November 2010 09:00–12:00 13:30–16:30
04 November 2010 09:00–12:00 13:30–16:30
05 November 2010 09:00–12:00

## 1. OPENING ACTIVITIES

### 1.1 Opening Comments (N1436) (Brown, Benito)

Walter Brown welcomed us to the Chicago/Geneva area, described the meeting and hotel facilities. One local restaurant is within walking distance of the hotel. Lunch break will be from 12:00 - 13:30. Breakfast served every morning starting at 6:30 AM. Morning and afternoon breaks will be at 10:30, and 14:30.

Walter presented an overview of Fermi National Accelerator Laboratory, and invited us to have lunch and tour there on Wednesday.

### 1.2 Introduction of Participants/Roll Call

| *Name* | *Organization* | *NB* | *Comments* |
|---|---|---|---|
| | | | |

| | | | |
|---|---|---|---|
| **John Benito** | Blue Pilot | | WG14 Convener |
| **Blaine Garst** | Apple | USA | |
| **David Keaton** | CMU/SEI/CERT | USA | PL22.11 Chair |
| **Tana L. Plauger** | Dinkumware, Ltd | USA | |
| **P. J. Plauger** | Dinkumware, Ltd | USA | |
| **Jim Thomas** | HP | USA | |
| **Rajan Bhatka** | IBM | Canada | HOD - Canada |
| **Clark Nelson** | Intel | USA | |
| **Douglas Walls** | Oracle | USA | HOD – USA, PL22.11 IR |
| **Barry Hedquist** | Perennial | USA | PL22.11 Secretary |
| **Tom Plum** | Plum Hall, Inc. | USA | |
| **Fred Tydeman** | Tydeman Consulting | USA | |
| **Larry Jones** | Siemens | | WG14 Project Editor |
| **Nick Stoughton** | Usenix | USA | Austin Group Liaison |
| **Walter Brown** | Fermi National Accelerator Laboratory | USA | |
| Paul E McKenney | IBM | USA | |
| | | | |

### 1.3 Procedures for this Meeting (Benito)

The meeting Chair, John Benito, announced the procedures are as per normal. Everyone is encouraged to participate in straw polls.

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

http://www.incits.org/inatrust.htm.

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

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.

The term "WP" means Working Paper, the latest draft version of the revision to the ISO/IEC 9899 C Standard, also known as C1X.

Barry Hedquist is Meeting Secretary.

### 1.4 Approval of Previous Minutes - Boulder (N1490) (Hedquist)

Several comments for typos, etc.

Minutes were modified per editorial changes and approved.

Final Minutes are **N1542.**

**1.5 Review of Action Items and Resolutions (Hedquist)**

ACTION: Larry Jones: Review use of headers required for the function declaration shown in the synopsis, and any others required to make the declaration valid.
DONE - WP

ACTION: David Svoboda to explore an approach to encode and decode pointers with Ulrich.
CLOSE - dropped

ACTION: Clark will ask Hans Boehm to add an explanation of memory sequencing to the rational re: N1411, Memory Model Rationale.
DONE N1521

ACTION: Clark to work on a clearer formulation of what the rules are in N1409, Aliasing and Effective Type.
DONE N1520

ACTION: Convener to discuss the resolution of N1448 with the Submitter, Steve Adamczyk.
OPEN

ACTION: David Keaton to work with Joseph Myers and write a proposal w/r/t SC22WG14.12205, Anonymous Structures.
OPEN

ACTION: Larry Jones to correspond with Joseph and make changes to the WP as needed w/r/t SC22WG14.12207, Static Assertions in 'for' loops.
DONE

ACTION: Convener to discuss N1471, Further Subsetting the C Standard, in light if the new work we've done this week for the C1X with Steve Montgomery / MISRA.
DONE

**1.6 Approval of Agenda (N1474) (Benito)**

Revisions to Agenda: None
Added Items: None
Deleted Items: None

Agenda approved as modified.

**1.7 Distribution of New Documents (Benito)**
None

**1.8 Identification of National Body Delegations (Benito)**
US, Canada

**1.9 Identification of PL22.11 Voting Members (Tydeman)**

See PL22.11 Minutes, following these minutes. 12 of 15 members present.

### 1.10 Next Meeting Schedule (N1436) (Benito)

The Spring 2011 meeting will be held at BSI, March 14-18, 2011, the week prior to the C++ meeting in Madrid.

## 2. LIAISON ACTIVITIES

### 2.1 WG14 / PL22.11 (Benito, Walls, Keaton)

Batavia: No Report

### 2.2 WG21 / PL22.16 (Nxxxx) (Plum, Benito, Sutter)

The ballot for the WG21 C++ Final Committee Draft, FCD 14882, closed on 26 July, 2010. The committee will meet next week to begin Ballot Resolution, and again in March 2011, in Madrid. The plan is to complete Ballot Resolution and vote out an FDIS no later than July 2011.

WG14 published a Liaison Statement to WG21 (N1432), reaffirming its intention to remain on schedule for C1X (N1392).

WG21 provided a liaison statement on Atomics to WG14. See agenda item 6.3.

Batavia: WG21 meets next week in Batavia, Fermi National Accelerator Laboratory.

### 2.3 PL22 Programming Languages (Plum)

PL22 meets twice a year via teleconference. A meeting will be held in late July 2011 to establish US positions to SC22 Plenary.

### 2.4 WG11

No Report – WG11 is going away, so this item will be dropped.

### 2.5 WG23 (Benito)

*Boulder: TR passed, revision in process.*

Batavia: TR is published, revision is in process. Will meet in Dec, San Diego. Meets four time a year, and has teleconferences. For further information, contact John Benito.

### 2.8 MISRA C

Category C Liaison. No Report.

### 2.9 Floating-point Study Group (Thomas)

*Boulder Discussion: Jim Thomas. Nothing new. Making slow but steady progress, meeting via teleconference twice a month. Contact Jim Thomas for details.*

Batavia: Working on a C binding for the floating point standard. See Jim Thomas for more info. Meetings are by teleconference.

**2.10 Secure Coding Study Group (Keaton)**

*Boulder Discussion: Plans to ask WG14 for a Type 2 TR, after C1X is complete. Tentative meeting in Mid July. Contact David for details.*
Batavia: Work is progressing. Will propose a TS by Sept next year. Meeting here on Friday.

**2.11 Other Liaison Activities**

Austin Group has submitted a Liaison Report. See Agenda Item 6.41, N1529. A revision to the POSIX Standard was published last year. Work on a TC is progressing.

## 3. EDITOR REPORTS

**3.1 Report of the Rationale Editor (Benito)**
*The proposal paper's author is responsible for writing the rationale*.

Convener is adding text for the rational.

**3.2 Report of the Project Editor (N1517) (Jones)**
There are a number of non-editorial issues that need to be addressed, and the items are in N1517. Items to review.

Non-editorial issues:
1. Should char be a standard integer type? Does it make a difference?
ACTION – Tom Plum to research.

2. Footnote 57 in 6.2.8p3 notes that every over-aligned type is, or contains, a structure or union type with a member that has an extended alignment. It was not clear to the editorial review committee where, if anywhere, this is stated normatively.

WG21 is also working this issue, and may have a resolution next week. There are compilers that allow this. Leave it as is.

3. The prohibition in 6.5.2.3p5 against accessing members of an _Atomic-qualified structure or union would seem to make them rather difficult to use. Should this say that undefined behavior *may* occur rather than that it *does* occur?

The concern here is over the occurrence of data races. We left it *undefined* intentionally. There is no real consensus.
ACTION – Blain will look into this, and prepare words.

4. The ‘‘editorial’’ change made to 6.7.6.2p2 in response to DR 320 and incorporated as part of TC3 had an unintended normative effect that needs to be reversed.

The original words were:
*Only an ordinary identifier (as defined in 6.2.3) with both block scope or function prototype scope and no linkage shall have a variably modified type.*
*This clearly restricts the kinds of things that can be declared with a variably modified type. In particular, it forbids declaring structure and union members as is noted in footnote 121 in 6.7.2.1.*

The revised wording from DR 320 no longer contains this restriction:

An ordinary identifier (as defined in 6.2.3) that has a variably modified type shall have either block scope and no linkage or function prototype scope.

I propose the following rewording to restore the restriction:

*If an identifier is declared as having a variably modified type, it shall be an ordinary identifier (as defined in 6.2.3) and have both block scope or function prototype scope and no linkage.*

Move 'no linkage' first.

5. The description of initialization in 6.7.9 was not updated to take the new sequencing model into account. I propose updating 6.7.9p23 to:

*The evaluations of the initialization list expressions are indeterminately sequenced with respect to one another and thus the order in which any side effects occur is unspecified.*

OK

Does this need to be tweaked for compound literals? No. It should be taken care of during initialization.

6. The description of signal handling in 7.14.1.1p5 should be harmonized with the description in 5.1.2.3p5.

The description of a signal handler in p2 should probably note that functions called indirectly via standard library functions like **abort** (when **SIGABRT** is being caught) are also considered to be part of the handler.

ACTION - Larry to look into this item further.

7. ISO/IEC TR 10176 *Information technology — Guidelines for the preparation of programming language standards* (the basis for Annex D) has been updated (twice!) since the 1998 edition that we reference. We should probably update our reference and Annex D to the current edition.

See N1518

8. ISO 4217, *Codes for the representation of currencies and funds*, ISO 8601, *Data elements and interchange formats — Information interchange — Representation of dates and times*, ISO/IEC 9945, *Information technology — Portable Operating System Interface (POSIX)*, and ISO/IEC 10646, *Information technology — Universal Multiple-Octet Coded Character Set (UCS)* have also been updated. We should investigate referencing the newer editions.

ITTF will make us do this anyway. Nick believes that POSIX Standard reference (9945) can be updated. Others need to be looked at.

9. The index probably needs work, particularly for the newly added material.

If anyone has an item to add to the index, let Larry know.

**3.3 Status of TR 24731-1, Bounds Checking, Static (Benito)**

No Report

### 3.4 Status of TR24732, Decimal Floating Point (Benito)

No Report
Fred Tydeman volunteered to be the PE.

### 3.5 Status of TR 24731-2, Bounds Checking, Dynamic (Benito)

The TR is nearly ready for publication, awaiting proof from ITTF.

## 4. EDITORIAL MEETING REPORTS

### 4.1 Report on Editorial Meetings Held (Benito, Jones)

Held at Apple, teleconference capability, everything done is in the current WP, N1516.

## 5. FUTURE

### 5.1 Future Meeting Schedule

Spring 2011 – BSI, London.
Fall 2011 - Portland (tentative) October? September looks better. Week of 26th

*Future meetings could be at INCITS, Santa Cruz, or we can get sponsors for Europe, but right now we have no meetings scheduled for 2011. What does the group want to do? How can we get the experts we want to attend meetings. Teleconferencing is economical, but technical discussions can get lost. Jim noted that teleconference seems to work best if everyone is conferencing. Tom works with a group that is split 50/50 present/teleconference.*

### 5.2 Future Agenda Items

Per Normal

### 5.3 Future Mailings

| | |
|---|---|
| Post Batavia: | Dec 3, 2010 |
| Pre London: | Feb 14, 2011 |

## 6. DOCUMENT REVIEW

### 6.1 Convenor's Report and Business Plan (N1506) (Benito)

No action needed.

### 6.2 C and C++ Alignment Compatibility (N1507) (Crowl)

Batavia:

N1507 is a revision to N1447. The revision incorporates C++ core language subcommittee changes that make the *alignment-specifier* occupy the syntactic position of an attribute rather than a *decl-specifier*. This change has no effect on C compatibility. The revision also incorporates 'pack expansions' inadvertently dropped in the reversion to pre-attribute syntax. This change has no effect on C compatibility.

Are there any changes in this paper, proposed wording for C that differs from the proposed wording in N1447? No. We've already adopted the changes recommended for C in N1447.

### 6.3 C++ liaison statement to C on Atomics (N1508) (Crowl)

*The WG21 C++ Concurrency Working Group reviewed the Atomics proposal for C, and presents their planned courses of action for C++for a number of issues. WG21 has not adopted this paper.*

Is there any action requested of or required by WG14? Clark says there are only two points in the paper that we need to address:

1. **The C++ committee recommends that the C committee not support atomic floating-point operations.**

The current C draft includes support for atomic arithmetic operations on floating-point types. The C++ draft does not. C++ shall not support floating-point arithmetic operations.

We, WG14, do not see an issue here, so more understanding is needed via liaison. If we keep atomic float, then more work is needed in bit compare and exchange functions.

ACTION – Liaison w/WG21 for clarification.

2. **The C++ committee recommends that the C committee not support (forbid) atomic bitfields.**

The C draft includes support for atomic bitfields. The C++ committee knows of no use cases requiring atomic bitfields. Conservatively, then, the languages should not require them.

Not all bitfield types are required to be supported. Clark would like to give implementations the choice. Tom believes that doing this could be extremely difficult for some implementations. Blain believes this works, but not conceptually difficult case. JB, in any case, we would need words. That can come from further liaison, or from a proposal. We could require it for every bitfield, or not allow it (C++) for any bitfield. Or make it optional. Is anybody willing to work on this? The same folks that work on the type qualifier issue.

NO POLL – more info required

Straw Poll: Adopt recommendation #2 above, forbidding, of N1508, to the C1X WP.
result: 3, 6, 4
Decision: No Consensus to Adopt recommendation #2 above, forbidding, of N1508, to the C1X WP.

Straw Poll: Adopt a direction allowing support for atomic bitfields, to the C1X WP.

result: 11, 0
**Decision: Adopt a direction allowing support for atomic bitfields, to the C1X WP. Needs Words.**

Straw Poll: Adopt requiring support for atomic bitfields, to the C1X WP.
result: 2, 9
Decision: No consensus to Adopt requiring support for atomic bitfields, to the C1X WP.

On Wednesday we reviewed a new paper, **N1530, Garst, Atomic Bitfields Implementation Defined.**

Straw Poll: Adopt N1530 to the C1X WP.
result: 12, 2, 0
Decision: Adopt N1530 to the C1X WP.

There are a number of items listed as open issues.

**6.4 Optimizing away infinite loops (N1509) (Walls)**

N1509 proposes to eliminated 6.8.5p6 in the current WP as the author believes it to be incompatible with C99. See also N1528, Item 6.30, Response to N1509.
N1528 covers why the rules were added. N1509 was written to show a compiler can determine the opposite as well. N1528 breaks existing conforming programs. Adding an exception may solve the intent of both papers. That approach should be run by Hans. We can defer a decision and not affect getting a CD out, then respond to an NB comment. We do want to be compatible with WG21.

ACTION-Douglas to work with Larry to come up with words.
DONE

The proposed words are:

Change 6.8.5p6 as follows:
An iteration statement **whose controlling expression is not a constant expression (or omitted),** 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.

Straw Poll: Adopt N1509, as modified above, to the C1X WP.
result: 14, 0, 1
Decision: Adopt N1509, as modified above, to the C1X WP.

**6.5 Register has no effect on behavior (N1510) (Tydeman)**

N1510 points out that the use of 'register float' does not require the same precision and range as the use of 'auto float', and proposes additional clarification text for the Standard and Rationale. The lack of words was intentional, but it's not clear anyone cares. Storage class does not affect the type. PJ believes that a register variable could expect widened variables to be returned. If that was the intention, do we want to continue doing so. If there are compilers that are doing that, do we want to let it continue. Larry believe that register has no effect, there is nothing in the standard that says so. PJ preference is to leave it alone. People stopped using register some

time ago, but why mess up anyone who is. Tom: C++ compatibility. Should not perpetuate that 'register double' might look different.

Straw Poll: Adopt N1510 to the C1X WP.
result: 3, 7, 5
Decision: No Consensus to Adopt N1510 to the C1X WP.

**6.6 Clarifications for wide evaluation (N1511) (Thomas)**

1) 6.3.1.4 #1, first sentence: change "When a finite value of real floating type is converted to an integer type …" to "When a real finite floating-point value is converted to an integer type …"

The terms 'real floating type' and 'finite value' have specific meaning. This section is talking converting about converting from one 'type' to another, so the term type needs to remain. Perhaps a footnote. Most seem comfortable with the words that exist.

No Straw Poll

2) Conversions (promotions and demotions)

6.3.1.5 #1, #2 change

When a **float** is promoted to **double** or **long double**, or a **double** is promoted to **long double**, its value is unchanged (if the source value is represented in the precision and range of its type).

When a **double** is demoted to **float**, a **long double** is demoted to **double** or **float**, or a value being represented in greater precision and range than required by its semantic type (see 6.3.1.8) is explicitly converted (including to its own type), if the value being converted can be represented exactly in the new type, it is unchanged. If the value being converted is in the range of values that can be represented but cannot be represented exactly, the result is either the nearest higher or nearest lower representable value, chosen in an implementation-defined manner. If the value being converted is outside the range of values that can be represented, the behavior is undefined.

to

When a real floating-point value is converted to a floating type, if the value being converted can be represented exactly in the new type, it is unchanged. If the value being converted is in the range of values that can be represented but cannot be represented exactly, the result is either the nearest higher or nearest lower representable value, chosen in an implementation-defined manner. If the value being converted is outside the range of values that can be represented, the behavior is undefined.

The change seems to delete the requirement of the first paragraph in the original text. No it doesn't.

Table for now.

Jim produced a new paper, Clarifications for wide evaluation (N1531). Discussion. Clark wants more time (beyond this meeting) to think about this paper, and not make a decision about it at this meeting. Question: When an extended float (i.e. has more bits that a double) is cast to a double, what happens to the extra bits? Jim believes they should be scraped off. The standard implies that the extra bits remain. Jim wants the cast to work the same as an assignment. Larry

and Rajan believe the extra bits should remain. Why throw away the extra precision, and the Standard says the 'value' remains unchanged.

Thursday
Jim Thomas presented a newer paper (N1537) that does not require promotion to remove any extra precision in the main body of the Standard, but to require it in Annex F.

More discussion. Some are concerned about the change to Annex F, and the possible impact on customers. The approach to Annex F does improve predictability of the result obtained. Jim believes the WP, as written, is clearly wrong. Clark does not object to the clarifications to the main body, but not to Annex F. David believes the proposed change to Annex F meets the spirit and intent of Annex F. Walter points out that they have been burned several times by this issue (extra bits) due to the lack of predictability.

Straw Poll: Adopt the changes to the main body of the Standard, as proposed in N1538 to the C1X WP.
result: 10, 0, 5
Decision: Adopt the changes to the main body of the Standard, as proposed in N1538 to the C1X WP.

Straw Poll: Adopt the changes to Annex F of the Standard, as proposed in N1538 to the C1X WP.
result: 9, 2, 5
Decision: Adopt the changes to Annex F of the Standard, as proposed in N1538 to the C1X WP.

**6.7 Math Functions and Directed Rounding follow up (N1181, N1512) (Thomas)**

This paper proposes editorial changes to reduce the chances of confusion about the terms “exceptional conditions” and “floating-point exceptions”. There are many cases where one is use where the other in intended, or 'exception' v. 'LIA-1 exception' is intended, etc. This is editorial. Passed to the Project Editor. If there are any questions, they can be referred back to the Committee.

Decision: N1512, Exceptional conditions vs floating-point exceptions, passed on the PE for action.

**6.8 Exceptional conditions vs floating-point exceptions (N1513) (Thomas)**

Follow on to N1181 (DR 329) proposed text that was more specific about requirements for math functions with respect to exactness and the rounding direction mode. At the 2006 Portland meeting, the proposed changes were approved, though only for Annex F. This is a proposal to address a few items (in Annex F) that were missed or for which the change was incorrect. Six items are included. Jim Thomas went through the paper, explaining each of the items. DR 329 has been applied to the WP.

Straw Poll: Adopt N1513 to the C1X WP.
result: 10, 0, 5

Decision: Adopt N1513 to the C1X WP.

### 6.9 Conditional Normative Status for Annex G (N1514) (Thomas)

N1514 proposes to change Annex G, IEC 60559 compatible complex arithmetic, from "informative" to conditionally "normative". The rationale is that implementations of Annex G now exist (HP/UX and Sun compilers). The impact is that implementations that define __STD_IEC_559_COMPLEX__ would have to conform to Annex G. This is just as optional as it's always been.

Straw Poll: Adopt N1514 to the C1X WP.
result: 13, 0, 2
Decision: Adopt N1514 to the C1X WP.

### 6.10 Fix for Annex F pow special case (N1515) (Thomas)

On of the special cases for pow is wrong in C99.

**Proposed changes:**
In the second bullet in F.10.4.4 change

- pow(±0, y) returns +∞ and raises the "divide-by-zero" floating-point exception for y < 0 and not an odd integer.

to

- pow(±0, y) returns +∞ and raises the "divide-by-zero" floating-point exception for finite y < 0 and not an odd integer.

Insert another bullet after the second bullet:

- pow(±0, -∞) returns +∞.

Discussion: Doing this could mean adding a special case to check for this, which might affect performance? Yes, that's possible.

Straw Poll: Adopt N1515 to the C1X WP.
result: 3, 1, 10 – insufficient support to carry this forward.
Decision: No Consensus to Adopt N1515 to the C1X WP.

Straw Poll: Change: second bullet: add "..and may raise the divide by zero floating point exception."
result: No Objection
**Decision: Adopted as above.**

Here is suggested rationale for the committee-modified change in N1515:

The divide-by-zero floating-point exception is generally intended for operations that produce an infinite result from finite operands. In C99, pow(+-0, -inf) was covered by the pow(+-0, y) where y<0 case which was specified to raise the divide-by-zero floating-point exception. The provision to allow pow(+-0, -inf) to raise the divide-by-zero floating-point exception gives implementations

the alternatives to be consistent with the general meaning of divide-by-zero or to retain their C99 behavior.

### 6.11 Working Draft (N1516) (Jones)

No action needed.

### 6.12 Recommendations for extended identifier characters for C and C++ (N1518) (Nelson)

TR 10176:2003, Extended Identifiers, has been updated, and looks to be a moving target, but nobody is maintaining it. Use the XML approach, everything except, rather than listing those that apply? Clark has made a specific list of recommendations w/r/t citing this TR material as an Annex. Tom suggested pointing to the C++ Annex instead. That could be problematic since the C++ revision does not yet exist. We could adopt Clark's proposal, and change it later if we need to, i.e. in response to an NB comment.

Straw Poll: Adopt N1518, as applicable to C, to the C1X WP.
result: 14, 0, 1
Decision: Adopt N1518, as applicable to C, to the C1X WP.

### 6.13 Latency Reducing Memory Allocation in the C standard library (N1527, N1519) (Douglas)

N1527 replaces N1519 as of 10/20/2010. It proposes a minimal change in the dynamic memory allocation API in order to reduce system memory bandwidth usage. The change is to 7.22.3.1, aligned_alloc, and adds 7.22.3.2, aligned_realloc, 7.22.3.3, batch_alloc1, 7.22.3.4, batch_alloc2, 7.22.3.5, batch_alloc5, 7.22.3.9, malloc_usable_size, 7.22.3.11, try_aligned_realloc, 7.22.3.12, try_realloc. All designed to reduce latency.

This is all implemented and available. IPR for any of this is unknown.

Tom Plum already indicated to the submitter that this material is likely too late in our process to make it into C1X.

Not clear how much this has been implemented. PJ also believes it's too late in the process. JB pointed out that unless the author was here, getting adopted would be difficult. There are other options, i.e. TR, etc.

Straw Poll: Adopt N1527 to the C1X WP.
result: 0, 12, 4
Decision: No Consensus to Adopt N1527 to the C1X WP.

### 6.14 Fixing the rules for type-based aliasing (N1520) (Nelson)

N1520 addresses DR 236, and provides proposed wording based on our previous decision to close this DR. Tom agrees with the direction taken, but believes that further clarification is needed. Much discussion, and generation of example code on a white board.

"Am I gassing this guy, or what?" … Clark, after explaining a point to Tom.

Clark's reading of DR 236 was that the Committee wanted to treat writes to unions differently than reads. Is that still the case? Yes. Footnote 94: use of the word 'access', means only reading, not modification. Blain believes it could mean modification. No one else agrees. Larry will fix this.

Larry presented a case that contradicts the wording that Clark has proposed.

"When I misunderstood your wording, I thought you had nailed it." … Tom, responding to Clark's comment to disregard the proposed wording, and pointing out that DR 236 was now reopened.

Wednesday afternoon revisit, to address issues raised in prior discussions, formulate new issues, explore approaches, shoot them down in mid-air. A self selected group of masochists needs to continue the spelunking into the deeper darker reaches, trying to find the light. Stopped at 4:15.

Rajan brought up issues on N1520.

---- start of Rajan's material ----------
There is no provision in the aliasing rules for a read relying on the "common initial sequence" case presented in N1494 6.5.2.3 paragraph 6. Ignoring the common initial sequence issue implies that in a read of a u.b.y, the only lvalue that the committee is concerned with is u.b.y and not u.b. The proposed changes (N1520mod2) also explicitly take u.b into account (with an exception for the common initial sequence). As for the common initial sequence, consider:

```
union U {
   struct A { int x; } a;
   struct B { int y; } b;
} u;

int main(void) {
   u.a = (struct A){ 0 };  // established type of u is now struct A
   struct B b = u.b;  // since the entirety of u.b is in the common
initial sequence, is this a qualifying inspection?
   b.y = u.b.y;  // does the involvement of a B lvalue imply the
existence of a B object?
}
```

It is important that an expression a.m or pa->m does not generally need to alias an object which is not the "m" member of the type of "a" or *pa (respectively). Also: An evaluation of an lvalue does not necessarily lead to a read of of an object, thus the implications of the existence of an object as implied by N1494 subclause 6.3.2.1 paragraph 1 should be identified: - Where "lval", an lvalue expression, is evaluated:

```
1) must a read of a char, *(char *)&lval, be valid?
2) must memcpy((char [sizeof lval]){ 0 }, &lval, sizeof lval) be valid?
3) must a read of "lval" be valid?
```

The modification is for option (2), which implies (1) but not (3). Also:

```
struct A { int x; };
struct B { int y; };
union {
   struct A a; struct B b;
} u;
int main() {
   u.a = (struct A){ 0 };
   struct B *pb = &u.b;
   return pb->y;  // should this be allowed?
}
```

---- end of Rajan's material ----

Clark wants more time to review this material. Returned to discussion brought up by Tom, reverting to the Randy Meyers rule. On factor influencing a decision in this area is the impact on GCC compiler performance, since they make heavy use of unions. It's difficult to determine when type base aliasing rules are violated. Nothing being done here is going to make them any simpler to understand by programmers.

There does not seem to be any way that this matter will be resolved at this meeting. Each thread of proposed approaches leads to more questions. 11:48 am, Thurs, Nov 4, 2010.

ACTION – Clark do more work in N1520

### 6.15 Threads API Improvements and Issues (N1521) (Boehm)

N1521 proposes several items in an attempt to improve the threads API.

**1. Remove mtx_try**

PJ wants to check w/Becker on this item. The argument for removing mtx_try is that it can be buggy, complicates the Standard, and deviates from existing practice. Implies that try_lock must be supported. Blain disagrees with the statement that there is no benefit to keeping mtx_try. Walter points out there is an assumption of a guarantee that the standard does not make. Blain believes that paper is too much, too late, and found the arguments for it not compelling. The performance aspect is debatable. Douglas is in favor of removing mtx_try, and is persuaded by the arguments presented. Some of the items presented no longer exist in the WP as stated in the proposal, but have been rewritten. PJ views this as tinkering

Straw Poll: Remove mtx_try, as in N1521 to the C1X WP.
result: 5, 5, 5
Decision: No Consensus to remove mtx_try, as in N1521 to the C1X WP.

**2. Clarify thrd_t lifetime**

It is unclear how long the thrd_t values produced by thrd_create remain valid. There are two plausible answers:

1. Forever. thrd_t is large enough that it cannot practically wrap. 64 bits are probably enough; 128 are definitely enough. Arguably that's what the current wording requires, but I believe the requirement is way too subtle for implementors to get this right.

2. thrd_t values can be reused once a thread exits and has been joined or detached. This is the POSIX approach.

The question comes down to what we originally intended with thrd_t. PJ has no trouble with the clarification. The existing POSIX approach is error prone.

Straw Poll: Adopt the clarifications to thrd_t , as proposed in N1521 to the C1X WP.
result: 15, 0, 1
Decision: Adopt the clarifications to thrd_t , as proposed in N1521 to the C1X WP.

**3. Clarify memory model constraints.**

In a few cases, additional "synchronizes with" relationships are required to ensure that updates by one thread become visible to another. Three items are proposed, one of which has two choices. If we do the first, we are pretty much obligated to doing one of the two presented as second options.

Straw Poll: Adopt the clarifications to memory model constraints, mtx_unlock, thrd_create – 1st option, thrd_join as proposed in N1521 to the C1X WP.
result: 12, 1, 2
Decision: Adopt the clarifications to memory model constraints, mtx_unlock, thrd_create – 1st option, thrd_join as proposed in N1521 to the C1X WP.

**4. Clarify when and whether stdio functions acquire locks and can be used to communicate between threads.**

No words are proposed, no action taken

**5. Consider prohibition of cross-thread longjmps.**

No words are proposed, no action taken.

**6.16 Atomic Refinements (N1526, N1522, N1485) (Garst)**

The adoption of the Atomic Proposal N1485 by WG14 leads to an opportunity for further simplification of the C1x draft by eliminating section 7.17.6 which defines a limited set of opaque structures containing atomic elements. These elements were also correspondingly defined in C++0x as interoperable data types. Prior to N1485 these types served as the exclusive set of atomic types defined by C1x, but are now unnecessary due to the introduction of N1485 productive syntax for atomics.

N1526 replaces N1522 a/o 10/20/2010, and along with Lawrence Crowl's paper, WG21 N3164, proposes to simplify both Standards by eliminating the specific data types referenced above.

Blain presented N1526 as a revised proposal derived from discussion he had with Lawrence Crowl.

Tana pointed out that no vote was taken in the WG21 Concurrency Working Group due to the absence of two major players, however Clark believes that the group had a sense of consensus.

PJ does not want any atomic_* names removed. They are already implemented and in use, and there is no need to remove them. Blain agrees. It's just that there was no apparent benefit to leaving them in, but now, there is. Accepting PJs point essentially divides the paper into two points.

Part 1: Accept consideration of the proposed modification to 6.2.5, Types, paragraph 27, and nothing else (_Atomic can mix with const). This simply corrects an oversight.

Straw Poll: Adopt, in principle, Part 1 of N1526 as above to the C1X WP.
result: 13, 0, 2
Decision: Adopt, in principle, Part 1 of N1526 as above to the C1X WP.

Straw Poll: Adopt, in principle, Part 2 of N1526 as above to the C1X WP (remainder of paper)
result: 0, 13, 2
Decision: No consensus to Adopt, in principle, Part 2 of N1526 as above to the C1X WP (remainder of paper).

Question raised by Fred:
7.17.7.5 The atomic_fetch and modify generic functions

If a volatile int has the value INT_MAX, and a call to atomic_fetch_add() is done to add one to the int, what is the result?

The standard says: "There are no undefined results." Does the result wrap (like unsigned int)? Does it saturate? Something else?

Clark: The intention is that fetch and operate constructions have the same modular constructions as unsigned. Two's complement, wrap around as if they were unsigned. It is not really specified in the Standard. Editorial. Larry to do.
ACTION – Larry to modify the words to fetch and operate constructions as needed (7.17.7.5)

**6.17 Using specifier, not qualifier, for _Atomic (N1524) (Plum)**

N1524 proposes that the specifier '_Atomic (type-name)' be used v. the _Atomic type qualifier. Blain believes that the qualifier is the superior approach. Can we do both? Possibly. Tom believes we have two ways of saying the same thing, and that his approach preserves source compatibility with C++. Is type qualifier a bad idea? No argument is presented. Clark believes the type qualifier approach creates a wart in the Standard due to sizeof issues, i.e. the sizeof an atomic type qualifier does not have the same size or alignment as a nonqualified type. That's true, and is already stated in the WP. It exists to maintain compatibility with C++. Tom: The underlying issues are entirely one of aesthetics and taste. Clark believes that WP status quo is problematic as it mentions both type qualifier and specifier. David believes the parenthesized syntax makes C look second class. Prefers the qualifier approach. PJ believes the qualifier is a better approach. Clark's reading of qualifier rules would mean a pointer to a non-atomic qualifier could be converted to a pointer to an atomic qualifier. There are size and alignment issues here.

**There are likely a number of existing qualifier rules that should not apply to an atomic qualifier.**

Straw Poll: Adopt N1524 to the C1X WP.
result: 5, 5, 5
Decision: No consensus to Adopt N1524 to the C1X WP.

Clark is strongly in favor of the type specifier approach. What does the committee wish to do to address the problems with the status quo, i.e. existing qualifier rules that impact an atomic qualifier in an undesirable way? We need to review every mention of 'qualifier' in the WP, and describe an exception for each case.

**ACTION - Clark, Blain, Tom, and Nick will work on this.**

The group researched all uses of 'qualifier' in the WP. Removing the wart could possibly create issues that are yet unknown. In eight instances, the wording for qualifier would have to be modified.

Three possible directions. 1) eliminate the wart, 2) draft away the problems that currently exist, 3) leave the problems there, go to CD knowing they are there. We may be able to do #2 here in Batavia, but maybe not. Option #2 includes possibly adopting Tom's paper on the specifier approach.

Perhaps we should just adopt the C++ solution.

In any case we are setting up 'churn' for any of these options.

Jim – perhaps we could simplify this by setting up a list of compatibly equivalent types. JB agrees.

Blain is willing to develop a proposal for wording to adopt into a CD this week. The heart of the issue revolves around compatible types. Tom – the hard part is deciding the direction. Doing that would enable is to have words this week. Tom believes that a paper will be produced along the line of his proposal, and that Blain will produce a paper along the lines of using 'qualifier', with two options, using R&A (representation and alignment), and without. Expect papers tomorrow. The action item above continues.

**6.18 Memory-Order Rationale (N1525) (McKenney)**

N1525 provides rationale for the six memory order options contained in the C1X WP, and is based on WG21 N2153. It demonstrates the purpose of each option by way of example code fragments, and also shows how some varieties of memory fences may be used. There is no real action for the committee to take with this paper.
Blain worked with Paul on putting this paper together, translating Paul's C++ examples into C. There are typo errors in the code. Blain will work with Paul to correct them.

**6.19 Proposal from Defect Report 271 (N1491) (Tydeman)**

Straw Poll: Adopt N1491 to the C1X WP.
result: 14,0,1
Decision: Adopt N1491 to the C1X WP (along the lines of).

**6.20 Proposal from Defect Report 319 (N1492) (Tydeman)**

Adopting this will break existing implementations. There is no strong motivation for this change.

Straw Poll: Adopt N1492 to the C1X WP.
result: 1, 10,3
Decision: No Consensus to Adopt N1492 to the C1X WP.

**6.21 Minimal fix to complex divide (N1496) (Tydeman)**

The existing state loses the NaN, which should show up in the result. This corrects that.

Straw Poll: Adopt N1496 to the C1X WP.
result: 9, 0, 6
Decision: Adopt N1496 to the C1X WP.

**6.22 Clarification to fmod (N1497) (Tydeman)**

There may be additional cases to consider? Fred does not think so. Clarification.

Straw Poll: Adopt N1497 to the C1X WP.
result: 6, 1, 8
Decision: Adopt N1497 to the C1X WP. Editorial as needed.

**6.23 Add missing error conditions in math functions (N1498) (Tydeman)**

These are NEW requirements on implementations.

Straw Poll: Adopt N1498 to the C1X WP.
result: 2, 9, 3
Decision: No consensus to adopt N1498 to the C1X WP.

**6.24 Remove error/no error choices in math functions (N1499) (Tydeman)**

Two proposals are presented: 1) adds recommended practices, 2) add a list of obsolescent features. This material is in Annex F, and extends that material to the main body of the C Standard. Adding as a recommended practice creates some complications.

Proposal 1:
Straw Poll: Adopt N1499, Proposal 1, to the C1X WP.
result: 1, 9, 5
Decision: No Consensus to Adopt N1499, Proposal 1, to the C1X WP.

Proposal 2:
Straw Poll: Adopt N1499, Proposal 2, to the C1X WP.
result: 1, 10, 2
Decision: No Consensus to Adopt N1499, Proposal 2, to the C1X WP

**6.25 Remove ambiguous choices in math functions (N1500) (Tydeman)**

These are new recommended practices, and are aligned with the Annex F requirements. Two proposals are presented; 1) add recommended practices, and 2) add a list of obsolescent features.

**Proposal 1**
Straw Poll: Adopt N1500, Proposal 1, to the C1X WP.
result: 1, 9, 5
Decision: No Consensus to Adopt N1500, Proposal 1, to the C1X WP.

**Proposal 2**
Straw Poll: Adopt N1500, Proposal 2, to the C1X WP.
result: 1, 12, 2
Decision: No Consensus to Adopt N1500, Proposal 2, to the C1X WP.

**6.26 EPOLE as future direction (N1501) (Tydeman)**

C1X has adopted a concept of pole error, however various standards present different approaches to handling pole errors. C1X has no unique errno value for it. N1501 proposes EPOLE

as an errno value for pole error as a 'Future Library Direction'. It's not clear what we gain by doing this.

Straw Poll: Adopt N1501 to the C1X WP.
result: 1, 9, 5
Decision: No Consensus to Adopt N1501 to the C1X WP.

**6.27 Fix for DR 334 (comparison macros) (N1502) (Tydeman)**

This Defect Report has been discussed at several meetings. The committee decided to defer this topic until after the type generic macro issue was resolved. Since the type generic macro issue was resolved at the Boulder meeting (May 2010), it is now time to fix DR 334. Some of the material from this DR has already been incorporated into the WP, except for the material that applies to the addition of a footnote to 7.12.14.

[footnote]If any argument is of integer type, or any other type that is not a real floating type, the behavior is undefined.

Straw Poll: Adopt the above footnote to 7.12.14 from N1502 to the C1X WP.
result: 9, 0, 5
Decision: Adopt the above footnote to 7.12.14 from N1502 to the C1X WP.

**6.28 Fix for DR 300 (translation-time expression evaluation) (N1503) (Tydeman)**

N1503 proposes new requirements**:** 1) the same decimal constants must result in the same value and 2) internal representation, and operators must use the same precision. Three Changes are proposed:

1) Add to 6.4.4.2 Floating constants, paragraph 5:
All floating-constants of the same source form in the same translation unit shall convert to the same internal format with the same value. Footnote: 1.23, 1.230, 123e-2, 123e-02, 1.23L are all different source forms, so need not convert to the same internal format and value.

Straw Poll: Adopt N1503, proposal 1, to the C1X WP.
result: 13, 2, 0
Decision: Adopt N1503, proposal 1, to the C1X WP.

2) Add to 6.4.4.2, paragraph 7 (Recommended practice):
All floating-constants with the same mathematical value and type in the same translation unit should convert to the same internal format and value. Footnote: 1.23, 1.230, 123e-2 and 123e-02 have the same mathematical value, so should convert to the same internal format and value.

Straw Poll: Adopt N1503, proposal 2, to the C1X WP.
result: 2, 7, 5
Decision: No Consensus to Adopt N1503, proposal 2, to the C1X WP.

3) Add a 12th paragraph to 6.6: Constant expression:
All operators with the same type(s) of floating-point operand(s) in the same translation unit shall use the same precision.

Straw Poll: Adopt N1503, proposal 3, to the C1X WP.
result: 4, 6, 5
Decision: No Consensus to Adopt N1503, proposal 3, to the C1X WP.

**6.29 Fix for DR 301 (meaning of FE_* macros in <fenv.h>) (N1504) (Tydeman)**

*Defect Report 301 was answered (in part) as not really a defect, but a deficiency which could be addressed in a future revision of the C Standard. The material could be added as Recommended Practice. The problem being addressed is: what is the meaning of the FE_* macros for environments that are not IEEE-754 (IEC 60559)? Since this is being added as recommended practice, it should have no affect on existing implementations.*

There are two pieces to this paper:

Straw Poll: Adopt N1504 to the C1X WP.
result: 1, 8, 6
Decision: No Consensus to Adopt N1504 to the C1X WP.

**6.30 Why Undefined Behavior for Infinite Loops - Response to N1509 (N1528) (Boehm)**

N1528 is a response to N1509, which proposes a change in wording to C1X. See agenda item 6.4.

**6.31 Austin Group Liaison Report (N1529) (Stoughton)**

N1529 presents a number of issues on the latest version of the POSIX Standard that are waiting for input from WG14 for resolution. Only the issues have an impact on WG14 are listed.

1. AG issue 00000073: wmemcpy C Conflict?
The words in the POSIX Standard follow the words used in C90 + AMD 1, rather than C99, and recognizes they may be at odds with the C1X revision. AG seems to be prepared to change their words to match C1X (and C99) if WG14 has no intention of changing them.

We are taking no action here.

2. AG Issue 000074: Pointer Types Problem.
POSIX extends the C Standard to allow a pointer to a function can be stored in a pointer to void. This is used for dynamic library functions (dlsym() in particular), and opens the door to permit conversion of a pointer to a data object into a pointer to a function.

3. AG Issue 0000090: fscanf contradiction (expects changes to C1X).
The change proposed by AG has already been adopted in C1X.

The changes expected are in the current WP.

4. AG Issue 0000105: errno change on success
C99 seemed to imply that errno could not be set except as described in the Standard, i.e. it could not be set to success unless the Standard specifically said so. POSIX has taken an approach that allows errno to be set to success for those cases, and asks that WG14 review their approach.

There is no WG14 action to take.

5. AG Issue 0000109: mblen() not thread safe.
AG believes that mblen() and mbtowc are not thread safe, and should be listed as such.

We may want to think about an equivalent statement. No action at this time.

6. AG Issue 0000110: memchr input process order.
AG has asked that WG14 review the changes to their DESCRIPTION and RETURN VALUE portions of memchr.

Does this solve the problem? Not clear. Nick will look further.

7. AG Issue 0000162: Determining System Endianess during preprocessing.
There does not seem to be a standard way to perform a conditional compilation depending on the endianess of a system, and proposes a mechanism to do so. AG has asked for WG14 input in this issue. WG14 has indicated a lack of enthusiasm for doing this in the past, but is willing to look at it if someone submits a paper.

The observation is correct. There is post C1X interest.

8. AG Issue 0000249: AG wants to adopt additional backslash escape sequences for shell use, with the intent that they be usable in the C language. \e – escape character, \cx as a control-x character. \cA is 1, control \cZ is 26.

All other items are 'for information only'.

ACTION – Nick to prepare formal proposals as needed for action by WG14.
DONE

**6.32 Aliasing Rules (N1532) (Nelson)**

N1532 addresses an issue left unresolved from an earlier discussion of N1494.

Straw Poll: Adopt N1532 to the C1X WP.
result: 13, 0, 3
Decision: Adopt N1532 to the C1X WP.

**6.33 Update to memchar from POSIX (N1533) (Stoughton)**

From the AG Liaison report, N1533 proposes an update to memchar. This paper is intended to propose similar wording to that agreed for POSIX to be applied to C1X to keep both standards in alignment. Without this change, POSIX would keep their new wording, but it would be marked as a C extension.

Straw Poll: Adopt N1533 to the C1X WP.
result: 15, 0, 1
Decision: Adopt N1533 to the C1X WP.

**6.34 Update to character constants (N1534) (Stoughton)**

From the AG Liaison report, N1534 proposes to add two new escape sequences

\e – Escape character
\cx – to mean control-x, accompanied by a list of 'x' possibilities.

These sequences are widely used in shells, and could be applicable to the C language.

The character display semantics for /e are not specified. Should they be? Yes, if we adopt this. Should we add this to Future Directions?
Douglas does not want to take action on this paper at this meeting. If in the CD, however, it's easier to remove. Future Directions is non-normative anyway. We also reserve '\<lower-case-letter> ' for future standardization.

Objection to taking up this paper at this time, so no poll will be taken.

### 6.35 Synthesis on _Atomic (N1535) (Plum)

N1535 is a synthesis of prior discussions and previous papers, on the issue of atomic type qualifier v. atomic type specifier, and is based on the following solution set:

- Keep the syntax by which _Atomic can appear as a qualifier.
- Use the phrase "atomic, qualified, or unqualified" consistently.
- Do not use the term _Atomic-qualified.

Clark believes there are presentation problems with this paper.
Blain has another paper, N1536, _Atomic Qualifier Issues, that he would consider withdrawing of we chose to go in this direction. There are some specific issues that Blain wants to review. Are there any issues we can identify now? Yes, several items near the bottom of N1535, that go beyond what was intended in N1524, that are recommended changes. Additionally, there are a list of items that are NOT recommended that were included as part of the intent of N1524. There is no objection to the material in this paper. Some revisions need to be made. Clark does not think we can make it 'perfect' today. However, the technical substance should be there.

N1537 is a revision of and replaces N1535 above.
The paper incorporates a consensus approach developed by Tom, Blain, Nick, and Clark.

Straw Poll: Adopt N1537 for inclusion into the C1X WP.
result: 15, 0, 1
Decision: Adopt N1537 for inclusion into the C1X WP.

## 7. OTHER BUSINESS

***JB now has the proof for our last unpublished TR.***

### *7.1 C1X Schedule*

*What do we want to do with our schedule?*
We have to get a doc in to SC22 by the end of November

Straw Poll: Request the Convenor forward the C1X WP, as modified in this meeting, to SC22 for CD Ballot.

result: 16, 0, 0
Decision: Convenor forward the C1X WP, as modified in this meeting, to SC22 for CD Ballot.

## 8. RESOLUTIONS

### 8.1 Review of Decisions Reached

The following papers achieved consensus for adoption into the C1X WP. All are modulo edits as needed by the Project Editor.

N1491 Proposal from Defect Report 271 (Tydeman) - Y
N1496 Minimal fix to complex divide (Tydeman) - Y
N1497 Clarification to fmod (Tydeman) - Y
N1502 Fix for DR 334 (comparison macros) (Tydeman) - Y
N1503 Fix for DR 300 (translation-time expression evaluation) (Tydeman) – Part 1 ONLY
N1509 Optimizing away infinite loops (Walls) – Y
N1512 Exceptional conditions vs floating-point exceptions (Thomas) – Y - Editor
N1513 N1181 follow up (Thomas) - Y
N1514 Conditional normative status for Annex G (Thomas) - Y
N1515 Fix for Annex F pow special case (Thomas)
N1518 Recommendations for extended identifier characters for C and C++ (Nelson)
N1521 Threads API Improvements and Issues (Boehm) (thrd_t; memory model (mtx_unlock, thrd_create (option 1), and thrd_join)
N1526 Atomic Refinements (Garst) – first part only
N1530 Atomic Bitfields Implementation Defined (Garst)
N1532 Aliasing Rules (Nelson)
N1533 Update to memchar from POSIX (Stoughton)
N1537 Synthesis re _Atomic (Plum, Nelson, Blain, Stoughton)
N1538 Clarifications for Wide Evaluation (Thomas)

**N1539** Convenor to forward the C1X WP, as modified in this meeting, to SC22 for CD Ballot.

### 8.2 Review of Action Items

**Carry Over**

ACTION - Convener to discuss the resolution of N1448 with the Submitter, Steve Adamczyk.

ACTION - David Keaton to work with Joseph Myers and write a proposal w/r/t SC22WG14.12205, Anonymous Structures.

**New**

ACTION – Tom Plum to research whether char should be a standard integer type.

ACTION – Blain to research whether undefined behavior 'may' occur, or 'does' occur for attempting to access members of an _Atomic qualified structure or union. re: 6.5.2.3p5 WP.
DONE – To Larry, Editorial

ACTION – Larry to look into non-editorial issue #6, harmonization of the description of signal handling.

ACTION – WG21 liaison to clarify WG21 concerns regarding non-support of atomic floating-point operations.

ACTION – Clark, Blain, Tom, and Nick to work on identifying existing qualifier and specifier rules that should not apply to atomic qualifiers and specifiers.
DONE N1537

ACTION – Clark do more work in N1520. => (N1543).

## 9. THANKS TO HOST

The Committee expresses its great appreciation to Walter Brown, Fermi National Accelerator Laboratory, for setting up the meeting arrangements for this meeting in Geneva, IL , and to Walter's wife, Carol, for providing us with additional nourishment throughout the week
Marc Paterno, Mark Fischler for providing tours of Fermi National Accelerator Laboratory.

Thanks also to Dinkumware for providing the Wiki.

## 10. ADJOURNMENT

Meeting adjourned at 3:00, Thursday, Nov 4, 2010.

# PL22.11 Meeting
# 2 Nov, 2010

Meeting convened at 1600, 2 Nov, 2010, by PL22.11 Chair, David Keaton.

Attendees:

| ***Voting Members:*** | | |
|---|---|---|
| **Name:** | **Organization:**<br>**P – Primary, A - Alternate** | **Comments** |
| Blaine Garst | Apple - P | |
| John Benito | Blue Pilot - P | |
| David Keaton | CMU/SEI/CERT - P | PL22.11 Chair |
| Tana L. Plauger | Dinkumware, Ltd – A | |
| P. J. Plauger | Dinkumware, Ltd – P | |
| Jim Thomas | HP – P | |
| Rajan Bhatka | IBM - P | |
| Clark Nelson | Intel – A | |
| Douglas Walls | Oracle - P | PL22.11 IR |
| Barry Hedquist | Perennial – P | PL22.11 Secretary |
| Tom Plum | Plum Hall – P | |
| Bill Seymour | Seymour - P | |
| Fred Tydeman | Tydeman Consulting – P | PL22.11 Vice Chair |
| Paul McKenney | IBM - A | |
| | | |
| | | |
| | | |
| ***Prospective Members*** | | |
| Walter Brown | Fermi National Accelerator Laboratory | |
| Nick Stoughton | Self | |
| Larry Jones | Siemens PLM Software | WG 14 Project Editor |

Agenda Approved as modified.

1. **Selection and Review of US Delegation – done in Boulder**

2. **INCITS Anti-Trust Guidelines**
   *We viewed the slides located on the INCITS web site.*
   http://www.incits.org/inatrust.htm

3. **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.

4. **Identification of PL22.11 Voting Members (Tydeman)**
   See attendance list above.
   12 PL22.11 voting members participated out of 15.

**4.1 PL22.11 Members Attaining Voting Rights at this Meeting**
none

**4.2 Prospective PL22.11 Attending Their First Meeting**
Nick Stoughton, self.

**5. Members in Jeopardy**

**5.1 Members in jeopardy due to failure to return Letter Ballots.**
Apple, HP, CERT

**5.2 Members in jeopardy due to failure to attend Meetings.**
Seymour – attended this meeting, must attend next meeting.
LDRA effective at the close of this meeting.

**6. JTC1 SC22/WG14 Working Sessions**
All members of PL22.11, who are not JTC1 Officers, are members of the US delegation to WG14.

7. **New Business**
See David if you want to check your roster listing. INCITS billings have gone out for 2011.

**7.1 Upcoming Ballot Process**
The going in plan is to vote YES with Comments. We'll collect comments for a 30 day comment period as well as a straw poll. That will be followed by a 30 day letter ballot. Only PL22.11 voting members will be allowed to vote and submit comments. The spread sheet ballot comment form will be used, and will be included (attached) to the announcement starting the Straw Poll and Comment Period. The announcement will come from the PL22.11 reflector, so it's important that INCITS has your correct email address.

**8. Adjournment**
Adjourned at 16:11 local, 2 Nov, 2010