WG14 N1475
INCITS PL22.11/09-0002
Reply to the Attention of: Barry Hedquist
Email: beh@peren.com

ISO/JTC1/SC22/WG14 AND INCITS PL22.11
OCTOBER 26-30 2009 MEETING MINUTES

Meeting Location:

University Inn & Conference Center
611 Ocean Street
Santa Cruz, CA 95060
Phone: +1 831 426-7100
Fax: +1 831 429-1044

Host Contact information:

Blue Pilot Consulting, Inc.
P. O. Box 2998
Santa Cruz, Ca 95063
Phone: +1 831 427-0528
Mobil: +1 831 600-5547
John Benito

Meeting Host: ANSI, Plantronics
Meeting Dates: 26-30 October, 2009
Meeting Times:
26 October 2008 09:00–12:00 13:30–17:00
27 October 2008 09:00–12:00 13:30–17:00
28 October 2008 09:00–12:00 13:30–17:00
29 October 2008 09:00–12:00 13:30–17:00
30 October 2008 09:00–12:00

1. OPENING ACTIVITIES

1.1 Opening Comments (Benito*)*

John Benito welcomed us to Santa Cruz.
Coffee will be in the meeting room. Lunch forms available

1.2 Introduction of Participants/Roll Call

Each of the meeting participants introduced themselves.

Tim Rentsch	Abelian Software	USA	
Blaine Garst	Apple	USA	
John Benito	Blue Pilot	USA	WG14 Convener
Robert Secord	CMU/SEI /CERT	USA	
Tana L. Plauger	Dinkumware, Ltd	USA	
P. J. Plauger	Dinkumware, Ltd	USA	

Jim Thomas	Hewlett Packard	USA	
Michael Wong	IBM	USA/CANADA	HOD - CANADA
Paul McKenney	IBM	USA	
Clark Nelson	Intel	USA	
Dan Quinlan	Lawrence Livermore National Laboratory	USA	
Nat Hillary	LDRA	USA	
Herb Sutter	Microsoft	USA	
Andrey Gusev	Oracle	USA	
Barry Hedquist	Perennial	USA	PL22.11 Secretary
Tom Plum	Plum Hall	USA	
David Keaton	Self	USA	PL22.11 Chair (Acting)
Bill Seymour	Self	USA	
Douglas Gwyn	Self	USA	
Larry Jones	Self	USA	ISO Project Editor
John Brenneise	Silicon Image / Self	USA	
Douglas Walls	Sun Microsystems	USA	PL22.16 IR, HOD - USA
Fred Tydeman	Tydeman Consulting	USA	
Nick Stoughton	Usenix	USA	

1.3 Procedures for this Meeting (Benito)

The Chair, John Benito, announced the procedures are as per normal. Everyone is encouraged to participate in straw polls. INCITS PL22.11 members are reminded of the requirement to follow the INCITS Anti-Trust Guidelines which can be viewed 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 C Standard, also known as C1X.

Barry Hedquist is Meeting Secretary.

1.4 Approval of Previous Minutes (N1380) (Hedquist)

Several comments for typos, etc. Minutes approved as modified. Final Minutes are N**1421.**

1.5 Review of Action Items (Hedquist)

ACTION: David Svoboda to write detailed attribute syntax proposal, focused on existing practice and the current list of attributes intended to be standardized.
DONE N1403

ACTION: David Keaton to rework his previous proposal on anonymous unions, adding anonymous structures as appropriate.
DONE N1406

ACTION: Randy to write examples for both DR 340 and 342 to allow vendors to evaluate what their implementations actually do in these cases. CLOSE

ACTION - PJ to put together a paper on an alternate approach to implementing type-generic macros. (picked up from Ulrich and Nick action)
DONE (Nelson) N1404

ACTION: Arjun to revise N1335 to add alignof. JB will ping.
DONE (Nelson) N1397

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.
OPEN

ACTION: Convenor to find out from SC22 what we can do / not do with the existing TR, then look at how to proceed from there w/r/t N1351 and possible changes to TR18037.
OPEN

ACTION: Tom Plum to check that the proposed definitions in N1350 do not conflict with the ISO Vocabulary standard.
DONE – They don't.

ACTION: Convenor to work with TR 24732 Project Editor to develop a lite-weight DR log and Rationale update mechanism.
OPEN

ACTION: David Svoboda to explore an approach to encode and decode pointers with Ulrich.
OPEN

ACTION: Fred and Jim will rework the words in N1378.
DONE N1384

ACTION: PJ to write a rationale for N1371.
OPEN

ACTION: PJ to write a proposal for incorporating errno, as applicable w/r/t N1371
OPEN

ACTION: PJ to write a rationale for Threads, N1372.
OPEN

ACTION: Fred to write up compatible changes to the LIA annex w/r/t DR330.
DONE N1383

ACTION: Clark write a paper to clarify the behavior that values are computed before main starts, and thread local storage get the same values as before program start up.
DONE N1387

1.6 Approval of Agenda (N1405)

Revisions to Agenda:
Added Items:
5.18 Boehm, Memory Model Rationale – N1411
5.20 Crowl, Thread Compatibility – N1414
5.21 Nelson, thread-local clarifications – N1387
5.22 Tydeman, LIA annex corrections – N1383
5.23 Tydeman, xxx_TRUE_MIN macros for <float.h> – N1384

Deleted Items:
5.4 C Secure Coding Guidelines (N1393) (Seacord) .
5.19 Plum, WG14 email message #11790 – N1412

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 List above. 15 of 15 possible are present.

1.10 Information on Future Meetings and Mailings.

1.10.1 Next Meeting Schedule (N1413)

19 – 23 April 2010, Florence Italy

Nov 2010: Batavia, IL, or St Louis, MO. Likely the 1st week in Nov. (TBD)

1.10.2 Next Mailing Schedule

Post Santa Cruz mailing: 30 Nov 2009
Pre Florence mailing: 22 March 2010

2. LIAISON ACTIVITIES

2.1 WG14 / PL22.11 (N1390) (Benito, Walls, Keaton)

Major work item now is the C1X revision. N1390 is The Convenor's Report. MISRA C is now a Category C Liaison.

2.2 WG21 / PL22.16

WG21 is in ballot resolution for the CD ballot, and will be for the rest of this year. The plan is to release CD2 for ballot after the Spring 2010 meeting.

Crowl paper on threads on our agenda. N1414 See discussion at agenda item 5.20.

2.3 PL22 (Plum)

No items to report.

2.4 Linux Foundation (Stoughton)

No Report

2.5 Austin Group (N1407) (Stoughton)

TR balloted and approved with comments. Ballot resolution underway for the next 1-2 meetings. The US TAG for WG23 is PL 22, and several members of PL22.11 are on the TAG.

Several Issues raised via Austin Group Aardvark reports requesting WG14 action.
ACTION: Nick will put together a summary of Austin Group issues requesting WG14 input.
No immediate time constraint.

2.6 WG11

No Report

2.7 WG23 (Benito)

Just finished NB comments on PTR Ballot. DTR will go out in about 3 weeks. Language dependent on annexes for ADA and SPARK, which is underway. Possible work in Fortran. Work is proceeding on a C language annex with assistance from Tom Plum and Robert Seacord. Next WG23 meeting will be outside Venice, Italy the week prior to the C meeting in Florence.

2.8 Other Liaison Activities

Discussion of SC7 Study Group doing embedded programming work. Should this be in SC22? Expertise for the specific language contributions lies within the SC22/WGs.

A study group has been formed, headed up by Jim Thomas, to address Floating Point issues to provide input to the work of IEEE 754. The group meets by teleconference and has about 17 participants. A wiki is available. Contact Jim Tomas, or the Convenor for additional information.

3. EDITOR REPORTS

3.1 Report of the Rationale Editor (Benito)

No rationale has been developed for items being added to the WP. The proposal paper's author is responsible for writing the rationale.

One draft paper has been received by the Convenor with rationale information. No action taken.

3.2 Report of the Project Editor (N1402) (Jones/Stoughton)

Anyone with editorial issues can send them directly to the PE for resolution.
Should the definition of __STDC_LIB_EXT1__ be changed? Yes, a new feature was added.

3.3 Status of TR24731, Bounds Checking (Meyers)

Do we need a new Editor? Yes.

3.4 Status of TR24732, Decimal Floating Point (Kwok)

TR 24732 is now published. Type 2, must be paid for.

Do we need a new Editor ? Yes. Mike will check.

3.5 Status of Draft IS 24747, Special Math (Plauger)

Has been published.

3.6 Status of TR 24731-2 (N1388) (N1389) (Benito/Stoughton)

*DTR Ballot passed. Need **final** version to submit for publication to ITTF.*

ACTION: Nick and JB will work on getting TR 24732-2 to ITTF.

4. DOCUMENT REVIEW

4.1 C Standard Revision Schedule (N1392) (Benito)

JB presented a schedule for C1X. Is this a schedule we want to adopt? There are a number of C/C++ compatibility issues, such as threads, where it's not clear how successful we will be in reaching agreement. The C++ schedule will likely take longer than the one proposed for C. Do we want to wait for C++ to resolve all their issues? The general sense is that we may not be able to make the C++ folks happy. We previously wanted the C++ Committee to take the lead on some of the issues, but that was when we believed they would publish a revision years before we would.

Herb pointed out that mutexes, for example, will require design changes to gain compatibility. Is compatibility with C++ important to WG14? It's important, but we do not want to sit around and wait for C++ to stop inventing. They can't seem to get to an end point. Important? Yes. Wait, and wait? No. The fundamental differences between the committees in choosing what to standardize hinders convergence on compatibility.

David pointed out that C does not standardize things that do not have existing practice. Many C++ incompatibilities exist because of invention by the C++ committee. A number of implementations already exist that already contain the major features, such as threads, going into C1X.

The presented schedule is very aggressive, and will require the participation and engagement of everyone on the committee. However, it is workable.

We need a list of features, if possible, by the end of the week to make a decision on what will be in the revision. Revisit this topic at EOW.

4.2 Netherlands Contribution on the Growth of Programming Language Standards (N1395) (Wakker)

Proposal to subset language standards. Something more flexible to the user community, for example, than what's done with 'freestanding'. Doug is sympathetic to the problem, and would be willing to make use of feature test macros to better define the subsets. Many, including Plauger, expressed support for the concept, some expressed concern of the practicality in being able to define a subset(s) for C1X.

What do we do with this paper? Put the guidance contained in the paper in an annex in the revision? Come up with a list of features that could be optional? Committee created subsets? User created subsets? We are not the language police. Subsetting in this way would create different levels of conformance, like 'freestanding'.

4.3 Committee Working Draft (N1401) (Jones)

No further comments.

4.4 Removed

4.5 Floating-point to int/_Bool conversions (N1391) (Tydeman)

Clause 6, and Annex G make explicit exceptions for _Bool type, but Annex F does not. Fred proposed changes to Annex F to correct that over site.

Does this change affect what implementations are already doing?

Straw Poll: Adopt N1391, as is, to the WP

Yes – 14
No – 0
Abs – 8

DECISION: Adopt N1391 to the WP

4.6 Analyzability (N1394) (Plum)

Analyzability is not meant to be the name of the Bounds Checking Library.
What do we want to do with this paper?

Straw Poll: Add something along the lines of N1394 into the WP as an Annex.

Yes - 15
No - 6
Abs – 3

DECISION: Add N1394, modulo corrections by the PE, into the WP as an Annex.

4.7 Problems with wide function return values (N1396) (Thomas)

Evolution of N1362, N1382, DR290, N1353. The Standard, Annex F, currently allows for widened function returns, which undermines the requirements of IEEE 754, i.e. C99 does not requite a function return to be narrowed to the type of the function. There is a similar issue in the type returned by an expression using functions of a different type. It proposes that return types of expressions be consistent.

Tom: We loosened these rules at the request of floating point experts who came to a meeting in Kona.

There are two basic options proposed: 1) make the change standard wide, and 2) make the change only to Annex F.
Doug expressed concern that there are implementations that don't already do this, so he recommends the proposed changes apply only to Annex F. Mike Wong agreed. Bill can live with making the change only to Annex F. Clark opposes the first, can live with the second.

Straw Poll:
Adopt N1396 Recommendation #1 to the C1X WP.
Yes – 4
No – 10
Abs – 6

Straw Poll:
Adopt N1396 Recommendation #2 to the C1X WP.
Yes – 9
No – 4
Abs – 7

DECISION: Adopt N1396 Recommendation #2 (Alternate Recommendation) to the WP.

4.8 Alignment Proposal (N1397) (Nelson)

SC22/WG21/N2341 proposes wording to the C++ language and libraries to support alignment, and the language aspects of alignment in that paper are largely applicable to C. This paper proposes their adoption in a semantically equivalent manner (syntax aside, for the moment). The aspects of alignment intended primarily to support generic libraries in C++ are not included for adoption.

Some implementations are not aligned in 4, 8,16, etc sequences, but we decided long ago that trying to satisfy those was not worth the trouble.

Doug: Doesn't C99 already have implied alignment requirements for objects? He'll check.

Does an align action require a reallocation? There is existing practice at Apple, but it does not seem to be required anywhere?

Robert S: Should the functions take an rsize_t type, rather than a size_t type? rsize_t is conditional normative, so using it would be 'tricky'.
Tom: Maybe we could move rsize_t to the main standard?
Doug: Not in favor of doing that. Would lead to abuse.

Clark wants a list of changes needed for the next revision of this paper.
Tom would rather vote it in, and correct the few items that need to be fixed later, such as the syntax of [[ ]], which we do not like. General support for that idea. Voting it in does not shut off the avenue for future proposals/papers to make changes to the feature.

Two grammar rules in the paper, one uses C++ syntax that we don't like. The other has to do with where it goes. In C++ it can go before or after, and each means something different.

Treat the attribute as a declaration specifier.

Spelling of the keyword. Options: #1) __STDC_ALIGN , #2) _Align.
The first spelling seems to suggest it's a macro.

Straw Poll:

#1 __STDC_ALIGN – 1
#2 _Align – LOTS

Straw Poll: Adopt N1397 with the exception of the square brackets. Replace them with _Align.

Yes – 17
No – 0
Abs – 3

DECISION: Adopt N1397, with the exception of the square brackets to the WP. Replace the square brackets with _Align.

4.9 Treatment of Math Error Conditions (N1398) (Tydeman)

Fred believes that the effect of DR330 may prohibit math functions from "raise(SIGFPE)", and has proposed changes to correct that situation. Fred is specifically thinking of IEEE 754 trap handler. Jim Thomas pointed out that there is no way to do that in the Standard. Tom: People in C++ do not want to see math errors handled by signals. Doug: If people are using errno, they need some sort of promise about how it is set.

Straw Polls on each of the following proposals.

1. No math function shall raise **SIGFPE**.

Yes – 14
No – 2
Abs - 6

2. If no error occurs, errno shall be unaltered. Math library only. Error refers to three specific conditions ERANGE, EDOM, and EPOLE.

Yes – 16
No – 4
Abs - 4

3. If no error occurs, none of "invalid", "div-by-zero", "overflow" shall be raised.

Jim Thomas a bit nervous about this one. It may be unlikely these side affects occur. Doug believes this is already required, PJ agrees. Not an added burden. Withdrawn by Fred

4. If an error occurs and if the integer expression (math_errhandling & MATH_ERRNO) is zero [footnote], either errno is unaltered or errno acquires the value corresponding to the error condition.

[Footnote] Math errors are being indicated by the floating-point exception flags.

Yes – 13

No – 0

Abs - 9

5. If an error occurs and if the integer expression (math_errhandling & MATH_ERREXCEPT) is zero [footnote], either none of the exceptions "invalid", "divide-by-zero", or "overflow" is raised or the floating-point exception corresponding to the error condition is raised.

[Footnote] Math errors are being indicated by errno.

PJ expressed concern that this requirement may be harder for the implementer to get right. It's a 'either leave the flags alone, or set them correctly'. Agreement that this is harder to do. Clark concerned about why we are going down this road. PJ: Getting the implementations right can be done, but we shouldn't raise the bar too high. This is probably not really more difficult than any of the above items.

Yes – 5

No – 4

Abs - 12

DECISION: Adopt items 1, 2,and 4 of N1398 to the WP

4.10 Complex Multiply and Complex Divide (N1399) (Tydeman)

This paper contains three proposals to change items in Annex G of C99 (Optional).

1. C99, Annex G.5.1, contains bad examples with complex multiply and complex divide. There are LOTS of them.
Clark: fix this by deleting the examples.
Jim: This is tricky to implement, and the examples prove to be of value to implementers. Jim has a fix in mind, but it would take another paper. Jim, Fred, are free to write another paper.
Blaine: Mathematicians can always argue whatever we've done does not meet their needs. It's not the business of the C Standard to do this.

Straw Poll: Vote for only 1:

Proposal number 1: Remove C99 G.5.1, Example 1 and 2.
Yes – 6
Proposal number 2: Replace G.5.1, Example 1 and 2 with the following.
Yes - 0
Proposal number 3: Replace G.5.1, Example 1 and 2 with the following code More than #2
Yes – 4
Proposal number 4: Do none of the above.
Yes – 9

Do Nothing

4.11 Headers Not Idempotent (N1400) (Tydeman)

In C, Standard headers can be included in any order, however what happens to the macros in those headers when those macros change conditional #defines, and #includes. Well, that can

lead to a mess. The paper proposes a "guard" to protect macros that have already been included in a prior header. Situation most likely to occur in TRs containing feature test macros. The four footnotes in question are all C++ related. C++ has disclaimed them. Larry is willing to treat this as editorial, except for the footnotes.

Straw Poll: Remove the four footnotes listed in N1400 from the WP.

Yes – 20
No – 0
Abs – 4

DECISION: Remove the four footnotes listed in N1400 from the WP.

4.12 Towards Support for Attributes in C (N1403) (Svoboda)

Clark Nelson presented David Svoboda's paper on attributes. The paper basically borrowed the C++ approach to attributes, and is not w/o controversy, such as the double bracket notation [[ ]]. WG21 is still tweaking attributes, but there does not seem to be a move to remove it from the revision.

Bill S suggests that IF we add attributes to the revision, we should be compatible with C++. PJ agrees.

Tom pointed out that 8.3, Keyword syntax is a bit incomplete. This section was to be a fallback proposal in the event nothing else is adopted. Define what the names are, and it's up to the user to define what they do.

Mike Wong, author of the C++ paper on attributes, pointed out that the feature can lead us down the garden path and into a deep hole. There's danger in piling on attribute features.

Herb characterizes attributes as a portable interface for nonportable features. WG21 has already gone down the garden path, and dug itself into a hole. The grammar is a complete novelty, no one uses it. The two major vendors cited as existing implementations both oppose the grammar contained in proposal.

When WG14 first started down this road, we did not know what we should do about the syntax. We decided to wait to see what C++ did, which is how we got to where we are now. Different implementations have attributes that do the same thing, with different syntax.

Doug questions the need for attributes at all. Compilers already provide the features attributes would provide.

Mike points out that attributes are widely used, but folks would like attributes have the same meaning from compiler to compiler. Will standardizing attributes cause long-term pain, or long term innovation?

Blaine believes the proposal is too hodgepodge. Apple uses attributes extensively. Thinks it's wrong to try to come up with semantics for non-Standard extensions. Adopting the proposed syntax would cause a lot of problems for their compiler writers.
Herb pointed out that for a committee (WG14) that wants to standardize only 'existing practice', the syntax proposed is a complete novelty – it's from C++.

Clark summed up.

Do we want support for alignment? Yes? Blaine – not if the syntax is unacceptable.
Do we want a framework of some sort for 'attribute' like things in C1X?
Doug Gwyn teased out a point whether we should have a declarative syntax for alignment.

Straw Polls:
Should there be a declarative syntax for alignment?
Yes – lots
No – few

Should there be a way to designate that a function not return ?
Yes – 10
No – 2
Abstain -10

Do we want to do something akin to what's proposed in 8.3 Keyword Syntax?
Yes – 14
No – 5
Abs – 3

Do you support double square brackets? [[ ]]
Yes – 4
No – 15
Abs – 3

Do we want a syntax for non-Standard extensions?
Yes – 4
No – 12
Abs – 6

To move this forward, somebody needs to write a paper on this.

ACTION: Tom Plum, along with David Svoboda, Nick, Clark, Blaine, to work up a new paper for new keywords based on N1403.

4.13 Type-generic Macro Support (N1404) (Nelson)

This paper builds on N1340. When that paper was presented, Ulrich Drepper mentioned that the GNU compiler also implements features that can be used to implement type-generic macros. For the committee's consideration, those features are described in this paper as well as an alternate approach.

Discussion:
Clark presented this paper, and does not believe the GCC approach is a direction we should go in. He believes we should go in a direction closer to Plauger's earlier paper, N1340. That paper proposes a construct that resembles a function call, including a comma separated list of associations between the type-name and a function. It proposes a new alternative for the grammar *primary-expression* called *generic-selection*, with a new language construct *_Generic.*

How general or how specific to the type expressions need to be? Unknown at this time.
Why not adopt the EDG approach? (See N1340). EDG's implementation is really only suitable for floating point, and does not extend to atomics. Clark's proposal is a plausible extension of the EDG approach to include covering atomics. It does not touch on the preprocessor.

Where do we want to go with this? The only concern expressed thus far is one of macro expansion. No one has done tgmath this way? GCC has, but it's brand new. Concern about the lack of real world use cases. Implementation of tgmath to date is basically through compiler magic. Greater use of tgmath is likely with the coming of atomics and decimal floating point. We can either come up with a mechanism that's usable to implementers, or leave it to them to figure out the compiler magic.

Blaine finds the style of overloading very troublesome, and we are inventing rather than standardizing existing practice. It's interesting, and we should pursue a solution, but perhaps use a real overloading mechanism.

PJ: Perhaps we are overreaching with atomics. Expect to have more implementation experience later. Agrees that we are venturing into invention.

How desirable is generic support? We crossed that bridge in C99. It's there. tgmath is a flamingo with no legs, it has nothing to stand on.

Where do we go from here? Pursue this or not. Tom believes that decision has already been made, and does not want to see it dropped on the floor.

Do we want to go in the direction suggested by N1340?

Yes – 12
No – 4
Abs - 8

4.14 Anonymous Member Structures and Unions. (N1406) (Keaton)

This proposal specifies the form and interpretation of a pure extension to the language portion of the C standard to permit structures and unions to contain members that are anonymous structures and unions. It is in concert with what's being done in C++. (The current C++ WP does not change anything.)

Doug supports this proposal, it is in wide use.
Mike: Does this support tags? Dave: No. That would be incompatible with C++

Straw Poll: Adopt the changes proposed in N1406, modulo the usual edits.

yes – 18
no – 0
abs – 2

DECISION: Adopt N1406, modulo corrections by the PE, to the WP

4.15 Vectorized Types for C (N1408) (Wong)

This aim of this proposal is to suggest standardized vector types for C. This can be in the form of a TR or with enough interest, with the current C1X. However, this proposal can be viewed as an introduction to vector types, and is based on the use of OpenCL.

Is there interest for C1X ?

Is there interest after C1X ?
Is there interest in a more generalized form after C1X ?

The use of 128 bit SIMD processors has become ubiquitous in commercial products such as game consoles, cell phones, etc. SIMD language extensions have inconsistencies between them. Today, unique vectorized code must be written for every SIMD architecture.

JB: There's not a lot of point of two different groups working on this at the same time. Suggested establishing a liaison with OpenCL.

No action for this committee at this time. There is general support for the approach, and we can revisit this are at a later time.

4.16 Aliasing and Effective Type as it Applies to Unions/Aggregates (N1409) (Meyers)

Surmises that the union/aggregate conditions described in 6.5p7 should apply to the effective type, not the lvalue type, and proposed changes to the Standard.

Clark: The 5th bullet of the aliasing rules is not expressed symmetrically and it should be. Believes the suggested change is correct. The example is incomplete, and does not clearly demonstrate the problem Clark is aware of.
Doug: Does anyone have a problem with the proposed solution? Does it solve the problem?

ACTION: Clark to develop a clearer example of the problem he believes N1409 is trying to resolve. Done, see below. Three examples:

Example 1:

```
void f1(int *pi, double *pd, double d)
{
        for (int i = 0; i < *pi; i++) {
                *pd++ = d;
        }
}
```

In f1, *pi can be assumed to be loop-invariant, because the type-based aliasing rules don't allow *pd to be an alias for *pi. Therefore, it is valid to transform the loop such that *pi is loaded only once, at the top of the loop.

Example 2:

```
struct S { int a, b; };
void f2(int *pi, struct S *ps, struct S s)
{
        for (int i = 0; i < *pi; i++) {
                *ps++ = s;
        }
}
```

In f2, *pi cannot be assumed to be loop-invariant. *pi (having type int) can be accessed (presumably including modification) by an lvalue that has aggregate type including a member of type int. struct S is of course such a type, and the lvalue *ps has that type. Therefore, f2 could be called as follows:

```
struct S a[10] = { [9].b = 1000 };
f2(&a[9].b, a, (struct S){ 0, 0 });
```

Despite the fact that the initial value of *pi in the loop would be way too large for the array, this usage would nevertheless be safe; *pi would have to be reloaded for each iteration, and on the tenth iteration, the value of *pi would become zero, and the loop would terminate before the loop writes past the end of the array.

Example 3:

```
struct S { int a, b; };
void f3(int *pi, struct S *ps1, struct S const *ps2)
{
        for (*pi = 0; *pi < 10; ++*pi) {
                *ps1++ = *ps2;
        }
}
```

The question here is whether the object *ps2 may be accessed (and especially modified) by assigning to the lvalue *pi — and if so, whether the standard actually says so. It could be argued that this is not covered by the fifth bullet of 6.5p7, since the lvalue in question does not have aggregate type at all, much less one containing a member with the object type in question.

Perhaps the intention is that the question should be turned around: is it allowed to access the value of the object *pi by the lvalue *ps2. Obviously, this case would be covered by the fifth bullet.

All Clark can say about this interpretation is that it never occurred to him a possibility until this meeting, even though he thought about these rules in considerable depth over the course of many years. Even if this case is considered to be covered by the existing wording, he suggests that it might be worth looking for a less opaque formulation.

Is there anybody that thinks the rules are clear enough? No one is really able to interpret them. So it seems that they are not clear enough. The problem is how to state them.

Doug: It seems we are allowing users to have overlapping objects. How useful is that?

Bill: Optimizers push the rules to the wall

Do we want better guarantees, or fewer guarantees? Yes

ACTION: Clark to work on a clearer formulation of what the rules are, re: N1409.

4.17 #macro proposal (N1410) (Brenneise)

John Brenneise presented his proposal on the use of new macros. The new macros are:

#macro & #endmacro
#for & #endfor
#integer
#printf

These macros are existing practice in the Microsoft macro assembler. Has not been implemented in a C compiler. They provides additional flexibility to the programmer over the existing macros.

Bill sees this as a large chance in the preprocessor, and lacks implantation in C. Might be better as a TR, rather than adding it to the IS.
Tom: This proposal goes beyond that which we agreed to as criteria for additions to C1X, but is open to hearing the argument that this is really needed, but suspects that argument does not apply here. Also sees a potential C++ compatibility issue.
PJ: Wish we had this 30 years ago, but adding this to C1X would likely create preprocessor problems we would be solving over the next 30 years.

Defer action for now, may come back to this later.

later – where do we want to go with this, if anywhere. Thought we had said we don't want to do anything for C1X in this area (we did). We are not considering C1X.
There are lots of macro processing tools in the world already. If it's a TR, how many folks will actually implement it? It may be worth trying. John B tried to make his work as much like the rules for #define in C99 as he could, that's why its in the preprocessor. Doug Gwyn volunteered to work with John on doing further work on this paper.

4.18 Memory Model Rationale (N1411) (Boehm)

This paper is only rationale.
Doug: memory sequencing is not explained anywhere, and needs to be added to the rationale.
ACTION: Clark will ask Hans Boehm to add an explanation of memory sequencing.

4.19 Removed

4.20 C and C++ Thread Compatibility. (N1414) (Crowl)

[From the paper's Introduction] The compatibility between the C and C++ threading facilities is important to many members of the respective communities. There are at least three levels of compatibility: critical, important, and desirable. Of these, the C and C++ committees should commit to achieving critical and important compatibility. This paper analyses the compatibility between current draft standards, and recommends several actions to improve that compatibility.

The most useful kind of compatibility is when an application header using thread facilities can be included by and used from both languages. Furthermore, it is desirable for C++ programs to be able to use C++ syntax with objects from that header. The recommendations within this paper support that goal. [end of intro material]

[Discussion]

Paul McKenney presented the paper from Lawrence Crowl. The paper proposes a number of recommendations for specific issues. Herb Sutter has put together a teleconference to discuss and attempt to resolve issues of incompatibility. There will likely be several teleconferences. JB encouraged those who have concerns or interests in the issues presented here to get involved with the teleconferences.

Robert pointed out that adoption of new facilities, such as threads, opens up potential for new security / vulnerability issues, implying somebody should be concerned about. Tom pointed out that adding new portability features will always increase vulnerabilities. Doug believes the only real issue would be that of one thread being able to access another thread. There is a larger issue of where the computing model is going, multi-threaded parallel processing, et al.

Herb will broadcast an invitation to C, C++, POSIX groups to participate, however such a teleconference has no official standing.

4.21 Thread-Local Clarifications (N1387) (Nelson)

This paper addresses requested clarifications for thread-local storage. Initialization of thread storage duration objects should follows the same rules as for static objects. The approach is to use zero initialization and proposes to move the description of zero initialization so that it can also be applied to thread local storage.

Straw Poll
Add the changes called for in N1387, modulo the usual edits by the PE

Yes – 17
No - 3
Abs – 3

DECISION: Adopt N1387, modulo the usual edits by the PE, to the WP.

Another clarification concerned the effect of initialization of thread-local objects on the floating-point status flags. Under the assumption that each thread has its own floating-point environment, there are two possibilities. The floating point environment of the created thread is affected, -or – the created thread affects the environment of the thread doing the creation. We determined that there was no need to make any changes to the WP over this item.

4.22 LIA Annex Correction (N1383) (Tydeman)

The response to Defect Report (DR) 330 changed 7.12.1, but forgot to make a similar change to the LIA annex. See also N1398.

Straw Poll: Adopt N1383 to the WP

Yes – 17
No – 0
Abs – 7

DECISION: Adopt N1383 to the WP

4.23 XXX__TRUE_MIN Macros for float.h (N1384) (Tydeman/Thomas)

This is an evolution of prior papers on this topic (N1303, N1317, N1352, N1378).
Doug has suggestions for some editorial changes, will work with the PE.
Bill pointed out that he has never had an occasion to use any of these features. Why do we need them? Testers use them for test cases, but that's not real world. Doug cited a couple of examples to determine if precision has been lost. Clark: These don't help. Jim: there are cases where these are useful. They can be useful in determining the characteristic of the types. Bill: C++ already has a mechanism to do this, so he won't fight it.

Straw Poll: Adopt N1383 to the WP

Yes - 13
No - 0

Abs - 6

DECISION: Adopt N1384 to the WP

4.24 Requested Clarifications for Thread-Local Storage (N1387) (Nelson)

Duplicate - See 4.21

4.25 Updates to C++ Memory Model (N1417) (McKenney)

WG21 added memory fences to their working draft to ease migration of current practice to the C++0x standard, choosing semantics that can be readily implemented on current multicore/parallel systems. Memory fences are often provided in concurrent C and C++ programming environments, usually in the form of platform-dependent inline assembly. Adding memory fences to the standard promotes portability.

The purpose of this document is informative. If there is interest, a document proposing changes will follow.

Is this a guru level API ? Yes. It is intended for very knowledgeable programmers writing systems software.

Why are atomics not enough and why are fences needed? Using atomics requires identifying an object when doing synchronization. Using fences avoids that.

Do we have interest in a follow-on document? Should C adopt fences? Fences are already in the WP. Clark needs to figure out what we really have here.

4.26 Multibyte C local (Stoughton)

Raises a question of whether or not it worth submitting an enhancement request to standardize a locale using a single-byte char set, perhaps named "C.ASCII", to guarantee that applications can select that locale when they desire to portably use byte values 128-255 in character-based APIs after an appropriate setlocale() call?

No interest in WG14 to pursue this path.

4.27 On The Removal of gets() (N1420) (Stoughton)

The gets function in the stdio.h section of the C standard has long been a source of programming errors and vulnerabilities. Many compilers issue a warning when it is used. In TC3, WG14 marked this function as obsolete.

Proposes to remove gets from the Standard.

Adopt N1420 to the WP.
Unanimous consent.

4.28 FP Exception Flags at Program Startup (N1415) (Tydeman)

Proposes to clear all floating-point exception flags at program startup

Straw Poll: Adopt N1415 to the WP.

Yes – 5
No – 8
Abs – 7

No Action

4.29 Errno at Thread Create (N1416) (Tydeman)

The floating-point environment has thread storage duration. The initial state for a thread's floating-point environment is the current state of the floating-point environment of the thread that creates it at the time of creation. Proposes similar words for errno.

The attempt to create a new thread is permitted to fail.
Nick would like to see each thread capable of setting its own errno.

ACTION: Nick and Fred to come up with new wording for N1416.

4.30 Creation of Complex Value (N1419) (Tydeman)

Problem:

(x + y*I)

will NOT do the right thing if "I" is complex and "y" is NaN or infinity. It does work fine if "I" is imaginary. Users and library Implementors have noticed this defect in the standard and have been surprised that there is no, easy to use, portable way to create a complex number.

This proposal cannot fly the way it is worded because macros cannot be used as static initializers. We should find a way to make the above expression work the way its supposed to work. If we go beyond that, we may be going down a path incompatible with C++. Tom believes the syntax in this paper could be adopted in C++.

ACTION: Fred, Jim, Doug to come up with new wording for N1419.

5. OTHER BUSINESS

None

6. REVIEW

6.1 Review of Decisions Reached

The following papers have attained consensus to be added to the WP, modulo the usual edits by the project editor as indicated or appropriate:

N1391
N1398 - Items 1, 2, 4 only
N1394 – as an Annex
N1396 – Alternate Recommendation
N1397 - With the exception of the square brackets. Replace the square brackets with _Align
N1383

N1384
N1387
N1400 – remove the four footnotes listed in this paper from the WP.
N1406
N1420

6.2 Review of Action Items

Action Items Carried Over

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.

ACTION: Convenor to find out from SC22 what we can do / not do with the existing TR, then look at how to proceed from there w/r/t N1351 and possible changes to TR18037.

ACTION: Convenor to work with TR 24732 Project Editor to develop a lite-weight DR log and Rationale update mechanism.

ACTION – David Svoboda to explore an approach to encode and decode pointers with Ulrich.

ACTION: PJ to write a rationale for N1371, Thread Unsafe Standard Functions.

ACTION: PJ to write a proposal for incorporating errno, as applicable w/r/t N1371

ACTION: PJ to write a rationale for Threads, N1372.

New Action Items

ACTION: Nick and JB will work on getting TR 24732-2 to ITTF.

ACTION: Tom Plum, along with David Svoboda, Nick, Clark, Blaine, to work up a new paper for new keywords based on N1403, Support for Attributes.

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

ACTION: Nick and Fred to come up with new wording for N1416, Errno at Thread Create.

ACTION: Fred, Jim Thomas, Doug Gwyn to come up with new wording for N1419, Creation of Complex Value.

ACTION: Clark to work on a clearer formulation of what the rules are w/r/t N1409, Aliasing and Effective Type.

7. CLOSING

7.1 Thanks to Host

Many Thanks to Plantronics for hosting this meeting.
Thanks to Dinkumware for providing the Wiki.

8. ADJOURNMENT

Meeting adjourned at 1121 AM, 29 October, 2009.

PL22.11 Meeting
Tuesday, 27 Oct 2009

Meeting convened at 1630 hours by PL22.11 Chair, David Keaton

Attendees:

Voting Members:		
Blaine Garst	Apple	
John Benito	Blue Pilot	
Robert Seacord	CMU/SEI/CERT	
Tana L. Plauger	Dinkumware, Ltd	
P. J. Plauger	Dinkumware, Ltd	
Jim Thomas	HP	
Michael Wong	IBM	
Paul Mc Kenney	IBM	
Clark Nelson	Intel	
David Keaton	Keaton	PL22.11 Chair (Acting)
Andrey Gusev	Oracle	
Barry Hedquist	Perennial	PL22.11 Secretary
Tom Plum	Plum Hall	
Bill Seymour	Seymour	
Douglas Walls	Sun Microsystems	PL22.11 IR
Fred Tydeman	Tydeman Consulting	
Nick Stoughton	Usenix	
Prospective New Members of PL22,11		
Tim Rentsch	Abelian Software	
Andy Chou	Coverity	
Jacob West	Fortify	
Dan Quinlan	Lawrence Livermore National Laboratory	
Nat Hillary	LDRA	
Douglas Gwyn	Self	
John Brenneise	Silicon Image / Self	
Larry Jones	Siemens PLM Software	

Agenda Approved as modified.

1. Select US delegation for the next two meetings in 2010.

Per 5.7.12.1 of the RD-2, all attending an SC22/WG14 meeting must be US delegates, with the exception of the JTC1 Officers (Convenors, Project Editors, Ex Officio, etc.).

MOTION : Approve the US Delegation per the requirement stated above. (Walls, Hedquist) (15-0-0)

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. INCITS Automatic Billing System

INCITS has a new billing system. Expect a bill in November. If not paid by the end of Feb, you will be dropped automatically.

5. SC 7 Project

SC 7 has a project on embedded programming, and are requesting experts from C to participate in that project. John Benito volunteered. Robert Seacord also.

MOTION: Nominate John Benito and Robert Seacord as experts to participate on the SC 7 Project on Embedded Programming.

Unanimous Consent.

6. PL22.11 MX Recommendations

MOTION: INCITS/ISO/IEC 9899:1999 [R2005] is currently under revision, as such PL22.11 recommends reaffirmation of INCITS/ISO/IEC 9899:1999 [R2005], pending completion of the revision. (Walls, Stoughton)

Roll Call Vote:

Apple – yes
Blue Pilot – yes
CMU/SEI/CERT – yes
Dinkumware – yes
HP – Yes
IBM - yes
Intel - yes
Keaton - yes
Oracle – yes
Perennial - yes
Plum Hall - yes
Seymour – yes
Sun - yes
Tydeman yes
Usenix – yes

Motion Passes (15-0-0-15)

7. Members in Jeopardy due to Failure to return Letter Ballots.

Apple, HP, CERT

8. Adjournment

Adjourned at 4:48 PM, 27 Oct 2009