WG14 N1643
INCITS PL22.11/12-002
Reply To The Attention Of: Barry Hedquist
PL22.11 Secretary
Email: beh@peren.com

MINUTES (DRAFT) FOR October 22-26, 2012
MEETING OF ISO/JTC 1/SC 22/WG 14 AND INCITS PL22.11

Meeting Location:
DoubleTree by Hilton Hotel Portland
1000 NE Multnomah Street
Portland OR 97232
Meeting information:
N 1592
Local contact information:
Clark Nelson
Phone: +1 503 712-8433
E-Mail: clark.nelson@intel.com

Scheduled Meeting Times:
22 October 2012 09:00 – 12:00 13:30 – 16:30
23 October 2012 09:00 – 12:00 13:30 – 16:00
24 October 2012 09:00 – 12:00 13:30 – 16:30
25 October 2012 09:00 – 12:00 13:30 – 16:30
26 October 2012 09:00 – 12:00

Teleconference information:
Topic: WG 14 October 2012
Date: Every Monday, Tuesday, Wednesday, Thursday, Friday, from Monday, October 22, 2012 to Friday, October 26, 2012
Time: 09:00 am, Pacific Standard Time (San Francisco, GMT-07:00)
Meeting Number: 954 880 418
Meeting Password: wg14

To start or join the online meeting, go to iso-meetings

Audio conference information:
To receive a call back, provide your phone number when you join the meeting, or call the number below and enter the access code.

- Switzerland toll free: 0800-894627
- USA/Canada toll free: 1-855-299-5224

Having trouble dialing in? Try these backup numbers:

- Call-in toll-free number (UK): 0800-051-3810
- Call-in toll number (UK): +44-20-310-64804
- Global call-in numbers: call-in numbers
- Toll-free dialing restrictions: tollfree restrictions

Access code: *954 880 418*

For assistance:

1. go to ios meeting support

2. On the left navigation bar, click "Support".

To add this meeting to your calendar program (for example Microsoft Outlook), click this link: iso meeting to calendar

To check whether you have the appropriate players installed for UCF (Universal Communications Format) rich media files, go to ios meeting diagnostics

http://www.webex.com/

1. Opening Activities

1.1 Opening Comments (Nelson, Benito)

John Benito and Clark Nelson welcomed us to the Doubletree by Hilton Hotel Portland, and described the meeting facilities. Several local restaurants are within walking distance of the hotel. Lunch break will be from ***12:00-13:30.*** We are connected on WebEx to allow folks to call in. This meeting is hosted by ANSI and Intel Corporation. Refreshments are available in the hallway.

1.2 Introduction of Participants/Roll Call

Name	***Organization***	***NB***	***Comments***
Dave Prosser	Bloomberg	USA	
John Benito	Blue Pilot	USA	WG14 Convener
Jim Thomas	Blue Pilot	USA	
David Keaton	CMU/SEI/CERT	USA	PL22.11 Chair
Robert Secord	CMU/SEI/CERT	USA	
Daniel Plakosh	CMU/SEI/CERT	USA	
Roger Scott	Coverity	USA	
Daniel Plakosh	CMU/SEI/CERT	USA	
Tana L. Plauger	Dinkumware, Ltd	USA	
P. J. Plauger	Dinkumware, Ltd	USA	
Rajan Bhakta	IBM	Canada	HoD - Canada
Clark Nelson	Intel	USA	
John Parks	Intel	USA	
Robert Geva	Intel	USA	
Marius Cornea	Intel	USA	
Mike Hennell	LDRA	USA	
Clive Pygott	LDRA	USA	
Douglas Walls	Oracle	USA	HoD – USA, PL22.11 IR
Darryl Gove	Oracle	USA	
Deepankar Bairayi	Oracle	USA	
Liang Chen	Oracle	USA	
Barry Hedquist	Perennial	USA	PL22.11 Secretary
Tom Plum	Plum Hall, Inc.	USA	
Fred Tydeman	Tydeman Consulting	USA	
Blaine Garst	Self	USA	
Bill Seymour	Seymour	USA	
Larry Jones	Siemens	USA	WG14 Project Editor
Jacob Navia	Self	FRANCE	

1.3 Procedures for this Meeting (Benito)

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

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

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

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

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

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

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

1.4 Approval of Previous Minutes (Kona) (N 1604).

Several comments for typos, etc.

Minutes were modified per editorial changes and approved.

Final Kona Minutes are **N1642**

1.5 Review of Action Items and Resolutions (Hedquist)

ACTION: Convener to process an NP (NWI) through SC22 for a Technical Specification: Floating-Point Extensions for C: Part 1, Binary floating-point arithmetic.
DONE. Presented at SC22 Plenary. Proposed NP is on the Wiki.

ACTION: Convener to request N1598 be processed as needed. Also make it a Defect Report, DR 411.
Note: In the new procedures for ISO, there is no way to change an IS for simple errata. See 2.10.1 of the ISO/IEC Directives, Part 1.
DONE

ACTION: Clark Nelson to write up some new words for DR 405.
OPEN

ACTION: Clark Nelson to check with Hans Boehm to see it there was any discussion in WG21 (C++) on DR 406, and what, if any, resolution was reached.
OPEN

ACTION: Clark Nelson to check into the status of the concepts in DR 407 within WG21.
OPEN

ACTION: Willem, Blaine and Doug to work on further proposed words for DR 413.
DONE – Blaine has a proposal on the Wiki, N1655.

1.6 Approval of Agenda (N 1639).

Revisions to Agenda: Updated

Added Items: None

Deleted Items: None

Agenda approved by unanimous consent.

1.7 Identify National Body Delegations
US, Canada

1.8 Identify PL22.11 voting members

See PL22.11 TAG Minutes, following these minutes. 14 of 16 members present.

2. Reports on Liaison Activities

2.1 SC 22

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

2.3 PL22.16/WG 21 (Plum)

WG21 met last week. Asked if C, WG14, had an opinion had an opinion on constexpr. WG21 has not settled the issue. WG21 plans on putting out an Amendment to their Standard in 2014 that will consist of corrigenda and a small collection of new features. We need to stay on top of this. Same is true for issues approved by the Core Working Group, since all of them are being treated as corrigenda to C++11.

PL22.16 took a meeting vote to recommend a US position to the INCITS/EB on two Technical Reports in systematic review. The US voted to withdraw the TR on Library Extensions, and Confirm the TR on C++ Performance.

There was considerable discussion on UTF character sets. There is ongoing work, but no conclusions yet.

We may want to look at the work going on in a future C++ TS being developed on File System support, as well as the work in Numerics.

Further discussion lead to a conclusion that WG14 participants that normally attend WG21 meetings should be assigned specific areas to monitor for C/C++ compatibility issues.

> Tom Plum, Plum Hall (US) - Core, Evolution, Modules (SG 2), Numerics (SG 6) and Transactional Memory (SG 5).
>
> Bill and Tana Plauger, Dinkumware (US), Library, File System (SG 3).
>
> Clark Nelson, Intel (US), Core, Concurrency.
>
> Bill Seymour, self (US), Library, Database, Numerics (SG 6)
>
> Barry Hedquist, Perennial (US), Evolution, Networking (SG 4)

2.4 PL22 (Plum)

PL22 meets twice a year via teleconference. Next teleconference will be in June 2013. See Tom Plum, PL22 Chair, for details.

2.5 WG 23 (Benito)

Revision to their TR will be published soon with bindings to several languages, Meets in conjunction with SC22 plenary, and others.

2.6 MISRA C (Montgomery)

Mike and Clive reported that the document for the latest release has been reviewed, and is now in final editing. The new MISRA C is based on C99. Expect publication in 2012

2.7 Floating-point SG (N 1616) (Thomas)

The Study Group continues to work on a C binding for IEEE 754-2008 / IEC 60559:2011, to be delivered as a five-part Technical Specification. Teleconferences are held monthly, along with email discussions and wiki postings. Two documents were included in the pre-Portland mailing:

N1631 – Part 1, Binary Floating Point
N1632 – Part 2, Decimal Floating Point

For further information, contact Jim Thomas.

2.8 Other Liaison Reports - none

3. Teleconference Meeting Reports

3.1 Report on any teleconference meetings held (Benito)

June – meeting to handle comments for Secure Coding, republished the document, reviewed again, and is on the agenda for this session.

4. Future Meetings & Mailings

4.1 Future Meeting Schedule

- Spring 2013 – 23 – 26 April 2013, Delft, NL – starts on a Tuesday – Friday (see N 1240)
- Fall 2013 – Chicago, IL, USA, Sept 23 - Oct 5, 2013, with WG21. Exact split is TBD.
- Spring 2014 – OPEN
- Fall 2014 – St. Louis – Dates TBD.

4.2 Future Mailings

- Post Portland – 19-November-2012
- Pre Delft – 26-March-2013

5. Document Review

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

1. N1614 ***Typos in 6.27 Threads <threads.h>*** **(Plum)**

N1614 is a proposed Technical Corrigenda (**DR 414**) for 7.26.4 for two items:

7.26.1;p5. mtx_plain, and 7.26.4.2;p2, mtx_init (should this be two DRs?)

7.26.1;p5
Summary

The enumeration constants are

mtx_plain

which is passed to **mtx_init** to create a mutex object that supports neither timeout nor test and return; the "test and return" is referring to **try_lock**, **try_lock** is not optional, therefore the "test and return" should be removed.

Suggested Technical Corrigendum

Change 7.26.1 paragraph 5 to

The enumeration constants are

mtx_plain

which is passed to **mtx_init** to create a mutex object that does not support timeout;

7.26.4.2;p2

Summary

In 7.26.4.2 paragraph 2

The **mtx_init** function creates a mutex object with properties indicated by **type**,
which must have one of the six values:
mtx_plain for a simple non-recursive mutex,
mtx_timed for a non-recursive mutex that supports timeout,
mtx_plain | mtx_recursive for a simple recursive mutex, or
mtx_timed | mtx_recursive for a recursive mutex that supports timeout.
There are not **six** values listed, "six" should be changed to "these".

Suggested Technical Corrigendum

Change 7.26.4.2 paragraph 2 to

The **mtx_init** function creates a mutex object with properties indicated by **type**,
which must have one of the these values:

2. N1617 *Missing divide by zero entry in Annex J.2* (Seacord)

N1617 is a proposed Defect Report (**DR 415**) stating that a specific undefined behavior in the body of the Standard (6.5.5;p6) is not included in Annex J.2, and should be added.

Suggested Technical Corrigendum

Add a bullet with the following text to J.2 after bullet 45

If the quotient a/b is not representable, the behavior of both a/b and a%b is undefined (6.5.5).

3. N1624 *WD TS 17961, C Secure Coding Rules,* (Seacord)

N1624 is the Working Draft of a Technical Specification for C Secure Coding Rules, Dated 6/26/2012. Comments have been submitted on this draft.

4. N1625 *A container library for C* (Navia)

N1625 is a proposal on adding a container library to the C Standard.

Does an existing implementation exist? Yes, and is freely available.

Jacob Navia is the compiler implementer, and presented his proposal.

Bill Seymour: Why do this in C? C is a good language that will be around for a long time. Having a container library can solve many problems for those who use C by providing facilities that do not exist today. C is not an “overcomplicated moving target” language.

Blaine sees the proposal as inventing objects. Does C need objects? Blaine does not think so. Is there a way to have a simple object system? Is there anything the language to do to help this out? This proposal does not address changes to the languages.

David K is concerned about the implementation. If C++ can do this faster, why do it?

PJ: Since the implementation is freely available, why standardize it?

Rajan asked if Jacob would be willing to make this work a TS? Jacobs sees a real need to standardize it, and sees a lot of work still needed. He sees creating a standard as only a starting point to create a much better implementation.

Blaine believes it would be very difficult to implement this and get performance improvements from one compiler vendor to another w/o having language support.

Prosser points out that the proposal is really insufficient to do what he already does today.

Parks points out that for his environment, C and C++ need to be able to work together, and see this as a divergent path for his projects.

In general, the thrust of this proposal is to provide a C library for containers so that a user does not have to use C++.

Should Jacob continue this work?

Blaine sees this as a good solution for those who do not want to use C++. It's a large library, and will have to meet other concerns, such as threading.

Roger: Is there a large audience in pure C solutions?

Clark: If this model becomes popular in the next ten years, that would be great, but we would like to see a demand before we jump into it.

Which comes first: A standard approach, or an existing market approach? Jacobs sees common approaches to problems such as this will not be adopted by the market without involvement by the Standards Committee. Everyone will continue to develop their own approach because there is no standard.

PJ points out that for practical purposes, most shops are using a mix of C and C++, and that is the most likely scenario going forward. More and more use of a mix of C and C++ is the trend being taken by industry. As such, there is not likely a large audience for a C only solution.

Jacob believes his approach can achieve the same performance as C++.

Parks is not convinced the market of C programmers is as concerned about this problem as much as they do about other problems we are working on.

Mike points out that his experience is that those who are writing in C only write in C.

In summary, most believe the proposal is very thorough and well thought out. The question is the value of doing something is C solely for the sake of providing a competing approach to C++ that can be used by those who prefer C over C++.

Do we want to encourage Jacob to continue working on this effort?

Are we willing to help with this work? Jacob cannot do all this work alone.

Are we interested in seeing this work progress? That implies we want to see it come back.

Straw Poll: Does the committee want to pursue a library based container proposal in the near term?

Yes - 4, No – 12, Abstain – 3

5. **N1626** *Convener's Report and Business Plan* (Benito)

   This report was given at the SC22 Plenary. No questions or comments.

6. N1627 *Proposed defect report regarding tss_t* (Shepherd)

   N1627 is a Proposed Defect Repot (**DR 416**), stating that the Standard does not specify if, or when, destructors for thread specific data keys, created with tss_create, are invoked, and proposes to resolve that by aligning the behavior with that specified in POSIX for pthread_key_t

   Q: Is the lack of specifics intentional? How does this affect C++ compatability? See DR 416 discussion.

7. N1628 *Missing entries in Annex J* (Benito)

   N1628 is a propose Defect Report (**DR 417**) that identifies instances of implementation defined, undefined and unspecified behavior listed in the body of the C Standard but missing in Annex J.

   Suggested Technical Corrigendum

   Change bullet 42 of J.1 to:

   — The order and contiguity of storage allocated by successive calls to the **calloc**, **malloc**, **realloc**, and **aligned_alloc** functions (7.22.3).

   Change bullet 43 of J.1 to:

   — The amount of storage allocated by a successful call to the **calloc**, **malloc**, **realloc**, or **aligned_alloc** function when 0 bytes was requested (7.22.3).

   Change bullet 166 of J.2 to

   — A non-null pointer returned by a call to the **calloc**, **malloc**, **realloc**, or **aligned_alloc** function with a zero requested size is used to access an object (7.22.3).

   Change bullet 37 of J.3.12 to

   — Whether the **calloc**, **malloc**, **realloc** and **aligned_alloc** functions return a null pointer or a pointer to an allocated object when the size requested is zero (7.22.3).

8. N1629 *Discussion on DR 410: f(inf) is inf being a range error* (Tydeman)

   N1629 presents a number of issues to consider regarding DR 410.

9. N1630 *Recommendations for math library implementers* (Tydeman)

   N1630 is a set of recommendations to implementers of math libraries. This is a proposal.

   Fred presented N1630, and he believes it is a defect in the C Standard. It adds nothing new. It's all recommended practice. Others don't see a body of recommended practice as a defect report. Such work could be applied to the entire Standard. There is no support for this as a DR.

10. N1656 *Floating-point extensions for C - Part 1: Binary floating-point arithmetic* (Thomas)

    N1656 is an update to N1631, and was presented by Jim Thomas.

Notes for N1656 presentation.

Numbered paragraphs for inclusion in C11.

Added line numbering.

Minor corrections and clarifications.

Content changes …

p7: Changed conformance macro form **__STDC_IEC_60559__** to **__STDC_IEC_60559_BFP__**

p7: Added conformance for freestanding implementations - by

- adding strfromflt function (10.2 – p17) to numeric conversion functions in stdlib.h and
- requiring freestanding implementations to support numeric conversion function in stdlib.h

p7: Added want macro (and settled on the name) **__STDC_WANT_IEC_00000_EXT1__** where 00000 will be the ISO/IEC document number

p8: ISSUE 1: The **00000** in **__STDC_WANT_IEC_00000_EXT1__** is to be replaced with the ISO/IEC TS number when it becomes available.

P8: ISSUE 2: Do we need a specification for the WANT macro along the lines of K.3.1.1 for **__STDC_WANT_LIB_EXT1__**?

p11: Clarify that canonical-ness does not determine a separate value (TS 7.2, Std 5.2.4.2#5)

p11 and others: Replace misuses of “encoding” with “representation” or “value”

p13: Added references in Table 1 – Operation binding

p20: Recast specification of calls affected by constant rounding directions (per discussion with DP and DG)

Table 2 for which macro replacement has not been suppressed (Std 7.6.1a#5)

For goal to allow implementations to support signaling NaNs (and so declare by defining FE_SNANS_ALWAYS_SIGNAL in <fenv.h>) …

various: Made some small changes to clarify whether certain operations may (and do if FE_SNANS_ALWAYS_SIGNAL is defined) or are not allowed to raise “invalid” for signaling NaN operands (JM comment)

- comparison operations may
- negate, fabs, copysign, totalorder, totalordermag, inquiry macros (including new iszero) are not allowed to

p30: Renamed math rounding direction macros, e.g. FP_CEIL to FP_INT_UPWARD, so as not to suggest conversion to floating types.

p31: Clarified that math.h does not define intmax_t and uintmax_t, types returned by fromfp and ufromfp functions (JM comment)

Fixed or footnoted sample code regarding signaling NaNs

p26: round - added footnote saying this implementation doesn’t satisfy FE_SNANS_ALWAYS_SIGNAL

p33: fmax – avoid returning snan after invalid signal

p34: fmaxmag – ditto

p38: Fixed domain error spec for **iseqsig** macro (7.12.14.7) by saying "a domain error occurs for the macro, as if a domain error occurred for a function (7.12.1)"

p40: Added iszero() macro (14.7), to assure IEEE-required non-signaling behavior

p46: Added specification to <tgmath.h> (16.0) to

- handle new functions introduced by Part 1
- clarify that tgmath macros are affected by constant rounding directions
- give sample implementation for type-generic cbrt that handles constant rounding directions

Tom: Do signaling Nans trap? No, by default.

Benito – there are a number of editorial things that need to be done to this document prior to sending it out as a NWI. Given that, are we comfortable with the content as something we are ready to send out, or do we want a review. Move forward.

11. N1657 *Floating-point extensions for C - Part 2: Decimal floating-point arithmetic* (Thomas)

N1657 is an updated version of N1632, a prior draft of a Technical Specification for Decimal Floating Point – Part 2, Decimal Floating-point arithmetic. Note: The internal doc number on this doc is N1648, which is in error.

Tom raised a general issue about C++ compatibility w.r.t. Part 1, and Part 2, simply to raise this as a potential issue with WG21 (C++).

Notes for N1657 presentation:

Part 2 of TS – will supersede decimal FP TR.

Used same ISO template as for part 1, copied in the content from decimal FP TR.

Will note significant changes from the TR - will refer to pdf page numbers.

Hope that with changes based on responses document will be ready for detailed review at or shortly after next meeting. About 90% of the final content is here.

Showed old text for each change; updated to refer to new standards (C11, IEC 60559:2011), and made several editorial clarifications.

Added new IEC 60559 features and interfaces.

P7: Retained same conformance and want macros.

- conformance macro defined with date
- compatible with old TR
- style not consistent with Part 1 – There was some general discomfort with that approach, and the SG was asked to consider making it consistent.

p17: Do we need a specification for the WANT macro along the lines of K.3.1.1 for **__STDC_WANT_LIB_EXT1__**?

P7: Required conformance to Part 1

- covers the type independent aspects of IEEE FP (exceptions, conversions with integer types, optimization, contractions, returns, NaN support)
- facilitates specifying conversions between decimal and binary formats
- implementation without Part 1 conformance can say they follow (adhere to) the specification for decimal FP in Part 2

Some general discussion on requiring conformance to Part 1, Binary Floating Point, in order to conform to Decimal Floating Point, Part 2. Why? There are general specifications contained in Part 1 that are required for Part 2. For expediency, those basic requirements are not repeated in Part 2. There is some general discomfort with that approach. The IEEE Standard allows for independent implementations, binary or decimal. Why aren't we? Vendors at this meeting seemed to be comfortable with this approach, but we should be aware that this as a potential issue. Possible use of an 'as if' rule by citing applicable sections of Part 1 should be considered

p12: Changed DECN_SUBNORMAL_MIN to DECN_TRUE_MIN - to be consistent with C11

p12: Specified the complete operation binding for decimal by referring to the binding table in Part 1 and

- adding supplementary text as needed

p13: ISSUE 1: Should we capture in C11 the IEC 60559 recommendation for implicit inexact **conversion** to integer type ***to raise the "inexact" exception? Would this conflict with conventions that assignments and casts are equivalent?***

C11 does not refer to the 2011 version of 60559. Should this TS make that reference? Should implicit conversions raise an exception? Compilers will generally raise a diagnostic over implicit conversions, so, yes. Prosser – should we consider initialization an issue in this category as well? If the initialization is an "implicit conversion', is would be treated as such, an exception would be raised.

p15: Removed the FLOAT_CONST_DECIMAL64 pragma – error prone

p15: Removed the d and D suffixes for double – needed only because of pragma, conflicted with prior art

p17: ISSUE 2: Do we need a specification for the WANT macro along the lines of K.3.1.1 for __STDC_WANT_LIB_EXT1__? Yes.

P20: Added DEC_SNANN signaling NaN macros

p26: ISSUE 3: Should we change "If value is a decimal floating-point number, the exponent is an integral power of 10; otherwise it is an integral power of 2." to "If value is of generic floating type, the exponent is an integral power of 2. If value is of decimal floating type, the exponent is an integral power of 10"? Can we assume generic RADIX is not 10?

We don't know.

P27: ISSUE 4: For ldexp, and scalbn below, should we change "A range error may occur" to "A range error may occur for finite arguments"? Waiting for WG14 resolution of same issue for corresponding functions in C11. TBD.

p29: Added types and functions for dec*i*mal re-encoding for DPD and BID encodings. DPD is Densely Packed Decimal, BID is Binary Integer Decimal, and are used by the decimal re-encoding functions.

p31: Added disambiguating specification and examples for printf a,A edge cases (where rounding is required and were precision is 0).

p37: ISSUE 5: Clause 12.8 needs to provide suggested changes to C11 tgmath.

P37: ISSUE 6: The study group is still considering other issues with tgmath for decimal.

P37: ISSUE 7: We intend to add an example showing how generic selection can be used to define cbrt for tgmath to handle decimal as well as generic FP types.

P37: Added specification for quantum exponents (12.9)

P38: ISSUE 8: We intend to add guidance for optimization that will preserve quantum exponents.

12. N1633 *Possible defect report: fmod(0.,NaN) and fmod (NaN, infinity)* (Tydeman)

    N1633 is a proposed Defect Report, submitted by Fred Tydeman. The issues revolve around fmod() in Annex F.

    Assigned **DR 418**

13. N1634 *Clarifying Memory Allocation* (Crowl)

    N1634 states that the existing wording of the C Standard describing memory allocation lacks sufficient precision, and could result in the exclusion of macro-optimization of such allocation.

    Clark led a discussion on this paper. Should allocations be clustered or divided up? Let implementations be smart about the way they do dynamic allocations.

    Douglas: Oracle only likes the proposed change to 7.22.3, paragraph 2. They oppose the proposed change to 7.22.3;p1.

    David K: Disagrees. The proposed change changes the meaning of what is intended.

    Defer any action on this paper. We see no action to take at this time, and will wait until WG21 (C++), takes action.

    ACTION: Clark to follow the disposition of N1634 in WG21 (N3433) and report such back to WG14.

14. N1635 *What the heck is a "generic function"?* (Walls)

    N1635 is a proposed Defect Report citing a need to redefine the atomic type generic functions as type generic macros, and define the underlying functions to which such macros expand.

    Assigned **DR 419**

15. N1636 *Comments on N1624* (Walls)

    N1636 is a set of comments on N1624, Draft TS 17961, C Secure Coding Rules, submitted by Douglas Walls. Most of the changes submitted were made to the draft.

    Dave Prosser is concerned that the use of the term ‘string’ does not match the definition of string in the C Standard. Blaine Garst agrees. Dave Keaton will review the use of ‘string’.

    ACTION: Robert Secord to make the agreed to changes to N1624 for the next mailing.

    ACTION: A small editing group review the edits made to N1624.
    The editing group will consist of:

    John Benito
    Blaine Garst
    David Keaton
    Dave Prosser
    Clive Pygott
    Robert Secord
    Willem Wakker

Douglas Walls

ACTION: Convener to either send the reviewed document to SC22 for PDTS Ballot, or not, as appropriate.

16. N1637 *Krevvers and Wiedijk Subtleties of the ANSI/ISO C standard*

    N1637 is a paper that states the C Standard does not allow Turing complete implementations; evaluation semantics does not preserve typing, and that no strictly conforming programs exist (i.e. the C Standard cannot guarantee a C program will not crash.)

17. N1638 *Updated WG 14 SD 1* (Benito)

    N1638 is an update to the WG14 SD-1, Joint Meeting and Mailing Information.

18. N1644 *New Work Item Proposal for C Binding to IEC 60559:2011*

    N1644 is a proposal for a New Work Item for a multi-part Technical Specification for a C Binding to the new Floating Point Standard, IEC 605599:2011.

19. N1645 *Strict Fork-Join Parallelism*

    N1645 is a WG14 version of WG21 N3409, proposing new constructs for C++ to support the central constructs of parallel programming for C++ through extension to the C++ language. The paper is a 'strawman' and does not propose wording at this point.

    Robert Geva, Intel, provided a presentation on C Language Constructs for Parallel Programming. In general, there are competing proposals, along with this, being considered in C++. The proposal calls for language support which can achieve performance improvements over library only approaches to parallelism. The slides will be available on the WG14 Wiki, and in the post-Portland mailing.

    Tom believes it would be good for WG14 to move forward with this. Among competing proposals being considered in C++, only the Cilk proposals meet the union of C and C++. It meets the bar for market need.

    A major CPU developer is producing ONLY multi-core CPUs. The need for parallel programming only goes away if CPU architectures revert to single-core. In terms of performance, there are no real metrics at this time that compare a library only implementation with a compiler supported approach, however, it is generally believed that a compiler (language) supported approach would have to produce better performance than a library only approach.

    Blaine believe that O/S will evolve that will solve this problem. Who? We don't know of anyone working in this direction.

    How would we approach freestanding? Accept the keywords and keep going.

    Does the committee believe that parallelism is important and we want to work on it? That seems to be the case.

    Clark – This is ether a good idea, or it isn't. Whether it's this proposal or some other proposal does not matter. This is something we should work on.

    Straw Poll:
    Do we want a formal proposal along the lines of the papers presented?
    (N1645, N1646).

    yes – 18
    no – 0
    abstain - 1

20. N1646 *Vector loops and Parallel Loops*

N1646 is a proposal for two new language constructs, vector loops and parallel loops, submitted by Robert Geva. This proposal is in concert with a WG21 (C++) proposal (WG21/N3419) for language support for parallel programming. A Study Group has been established within WG21 to research C++ language support for parallel programming.

21. N1647 *syntax error in specification of for-statement* (Gustedt)

    N1647 is a proposed Defect Report, submitted by Jens Gustedt, stating there is a syntax error in the specification of for-statement.

    Assigned **DR 420**

22. N1648 *initialization of* ***atomic_flag*** (Gustedt)

    N1648 is a proposed Defect Report, submitted by Jens Gustedt, stating that the intent of the C Standard to have atomic_flag as a primitive cannot be achieved with the current semantic for initialization.

    Assigned **DR 421**

23. N1649 *initialization of atomic types* (Gustedt)

    N1649 is a proposed Defect Report, submitted by Jens Gustedt, stating that the C Standard fails to specify the value of an atomic object if it is initialized by default.

    Assigned **DR 422**

24. N1650 *Defect Report relative to N1570, underspecification for qualified rvalues* (Gustedt)

    N1650 is a proposed Defect Report, submitted by Jens Gustedt, stating that rvalues with qualified types is underspecified throughout the C Standard.

    Assigned **DR 423**

25. N1651 *underspecification of* ***tss_t*** (Gustedt)

    N1651 is a proposed Defect Report, submitted by Jens Gustedt, stating that tss_t is underspecified.
    See also: N1627.

    Assigned **DR 424** – See also **DR 416.**

26. N1652 *The order on* ***uintptr_t*** *and* ***void*** *pointers should be compatible* (Gustedt)

    N1652 is a proposed Defect Report, submitted by Jens Gustedt, stating that the order on uintptr_t and void pointers should be compatible.

    This is a proposal, not a DR. We'll defer discussion until resubmitted as a proposal.

27. N1653 *no specification for the access to variables with temporary lifetime* (Gustedt)

    N1653 is a proposed Defect Report, submitted by Jens Gustedt, stating there is no specification for the access to variables with a temporary lifetime. Should such access be Implementation Defined, or simply forbidden?

    Assigned **DR 425**

28. N1654 *underspecification of thread functions with* ***void*** *return* (Gustedt)

N1654 is a proposed Defect Report, submitted by Jens Gustedt, pointing out that several thread functions in C11 do not have return values, implying that they cannot fail, while other implementations, such as POSIX, have return values for corresponding functions. The functions in question are: call_once, cnd_destroy, mtx_destroy, tss_delete, thrd_yield.

This is a proposal, not a DR.

6. Defect Reports

6.1 Review Status

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

DR 401 – CLOSED

DR 402 – REVIEW
ACTION - Rajan and Blaine to write up words/diagrams for DR 402. There's a typo as well, p12 s/be para 22.

We reviewed a diagram by Rajan. Clark was responsible for translating the math into English.

If the value of B is not a value provided by X, it must be possible to find a value for Y, and B gets its value from Y. Rajan sees an interpretation of the existing words where an implementer could use Y to provide a value for B, when in fact it should have come from X. Blaine believes the existing proposed words in the DR better represent, in industry standard terms, the intent than the current words in the Standard. Clark and Rajan seem to agree the DR interpretation was never intended.

Clark suggests that we take the Suggested TC in the DR, move the DR to Review, and spend some time between now and the next meeting working with C++ to find a way to further clarify them if we are able.

ACTION - Clark to examine the adoption of the Proposed TC words for DR 402, and tweek as needed.

DR 403 – CLOSED

DR 404 – CLOSED

6.2 Open Status

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

DR 400 – REVIEW

DR 405 – OPEN

ACTION - Clark Nelson had an action to write words for this – DONE

Add the following as 7.26.4 p1 and p2:

> For purposes of determining the existence of a data race, lock and unlock operations behave as atomic operations. All lock and unlock operations on a particular mutex occur in some particular total order.
>
> NOTE This total order can be viewed as the modification order of the mutex.

Straw Poll – Accept the words above as Proposed TC, leave OPEN.

DR 406 - OPEN
This item has also become WG21 Core issue 1466.

DR 407 – OPEN

Committee Discussion

This item has also become WG21 Library Issue 2130. Rajan believes we can do the first part of this DR as proposed for 29.3p7. If the first part is accepted, the second part may be redundant. Prosser disagrees, and is not comfortable with dropping the second part.

ACTION: Rajan to write up with words/diagram for DR 407.

Proposed Resolution:

After 7.17.3 paragraph 11 add the following:

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

Clark – This is a Library Issue for C++. We might want to liaison with C++ on this. Do we want the corollaries in the C Standard? Once we decide that, we should communicate that to C++.

The first part of the DR seems to be understood. Wait for C++ to make a decision on the issues they have already opened. Leave OPEN.

Straw Poll: (13-1-0) YES
Adopt the words in the Proposed Resolution given above for the first part.

Straw Poll: (3-10-3) - NO
Adopt the following derivatives following 7.17.3

Note that the following derivations do fall through the statement above, and as such, do not need to be stated explicitly:
For atomic modifications A and B of an atomic object M, if there is a memory_order_seq_cst fence X such that A is sequenced before X, and X precedes B in S, then B occurs later than A in the modification order of M.
For atomic modifications A and B of an atomic object M, if there is a memory_order_seq_cst fence Y such that Y is sequenced before B, and A precedes Y in S, then B occurs later than A in the modification order of M.

DR 408 - OPEN

Committee Discussion
This is already Undefined Behavior in C, and the submitter agreed. Add a Committee Response, moved to Ready.

ACTION: Blaine to draft a Committee Response for DR 408

DR 409 - Review

Feb, 2012 Meeting

- The committee rejected the Suggested Change in the main body of this defect report.
- The committee considered the following, but rejected it (as just being a restatement of 7.12.1 paragraphs 4 and 5).

  If the result overflows, a range error shall occur.

- A question arose as to why these range error cases are listed in the individual functions (instead of just being covered by the blanket 7.12.1 paragraphs 4, 5, and 6)

7.12.1 paragraph 1 has the answer:

The behavior of each of the functions in <math.h> is specified for all representable values of its input arguments, except where stated otherwise.

- Several other approaches were discussed, without any consensus reached
    1. Add a footnote to 7.12.1 paragraph 5, first sentence:

    In an implementation that supports infinities, a range error may happen for functions that map an infinity argument into an exact infinity or exact zero result.

    2. Add to end of 7.12.1 paragraph 4:

    Recommended practice

    In an implementation that supports infinities, a range error should not happen for functions that map an infinity argument into an exact infinity or exact zero result.

    3. Add to 7.12.1 paragraph 4:

    An implementation may define additional range errors, provided that such errors are consistent with the mathematical definition of the function.

Oct 2012 meeting

Committee Discussion
Fred wrote a paper, N1629. There are two parts.

1. Fixing a contradiction.

    The Committee rejected the words for the Suggested Change in Feb 2012

    But, this change looks OK to Jim Thomas. Break out 1-5, 10-12, as OK, Rajan disagrees. This change leaves infinity out for those who treat infinity as too large. Making this change will create work for implementers. Jim's fine reading of the definition range error 7.12.1;p4 of would indicate infinity is excluded, thus no change is needed.

2. Taking care of infinity.
    Implemented defined, or undefined. Rajan would prefer to see undefined behavior, so that implementers don't have to document the feature. Make it 'bounded' undefined behavior. That does not solve the problem, because it can still trap. Also, 'bounded' applies to Annex L. Not an option.

ACTION: Fred to write up words as a proposed Record of Response – DONE

Fred's words follow:

> Some on the committee believe that there is a contradiction. However, no agreement could be reached on how to fix it.
>
> Making the behavior implementation defined for infinity arguments requires implementers to document their behavior and some implementers do not wish to do that documentation.
>
> Some on the committee believe that exp(+infinity) being a range error is prohibited and is a bug in those implementations that treat it as such.
>
> exp(+infinity) is +infinity. Since the input +infinity is representable, then the output +infinity is representable in an object of the specified type. By, 7.12.1#4, a range error has not happened.
>
> Also, by 7.12.1#5, since the result is not finite, a range error has not happened.

A number of folks expressed concern over these words, by they are left to the Maintainer of the Defects to rework if needed.

Fred presented another possible solution using 'unspecified' , that would not require implementers document anything. Jim opposes this, since we had already decided yesterday that the existing words are correct, and no change is needed. There is no consensus to adopt Fred's new proposal.

Leave OPEN.

DR 410 - Review

Clark sees no reason to impose consistency between these two functions. They work differently

This DR has a Proposed technical Corrigendum

> In 7.12.6.5 paragraph 2, change the last sentence to:
>
> If the correct value is outside the range of the return type, the numeric result is unspecified and a domain error or range error may occur.

Straw Poll: Move to Review? No Objection.

DR 411 – CLOSED and published as a TC.

DR 412 - REVIEW

Feb 2012 Meeting

Proposed Technical Corrigendum

> In 6.10.1p6, change:
>
> Only the first group whose control condition evaluates to true (nonzero) is processed.
>
> to:
>
> Only the first group whose control condition evaluates to true (nonzero) is processed; any following groups are skipped and their controlling directives are processed as if they were in a group that is skipped.

DR 413 - OPEN

Feb 2012 Meeting

Committee Discussion

- *It was noted that this is basically the same issue as dr 253.*
- *The following was proposed, but there was no consensus for adoption.*

> *The initialization shall occur in initializer list order, each initializer provided for a particular subobject overriding any previously listed initializer for the same subobject [151]. Subsequently, all subobjects that are not initialized explicitly previously shall be initialized implicitly the same as objects that have static storage duration.*

Oct 2012 Meeting

Committee Discussion

Blaine believes the Standard is clear as written. Does anyone else read it differently? Clark believes there is some ambiguity. The intent for the value given in the example is 42. Rajan expected 0. It seems that we did not nail down words in the beginning to make it clear. Does anybody depend on '0' – not likely? This is really an edge case. If we make a

clarification, it's not likely we will get anybody upset. Adding an example would be non-normative. It's also likely that the proposed new text can also be misread. GCC also gives '0'.

ACTION: David Keaton to draft words for DR 413.
DONE – N1659

From David's example, what does 'maximally-enclosing' mean? It's not a defined term in the Standard?

Make the 'new footnote' part of the normative proposed text in place of the 'maximally-enclosing'.

ACTION – Dave K to update N1659 (not yet published). - DONE

Dave Prosser does not see how the new proposed text clarifies the initial question asked – which sub-object is being initialized? Back to the kitchen.

Updated N1659:

Committee Discussion

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

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

Leave OPEN.

DR 414 (N1614)

Suggested Technical Corrigendum

Change 7.26.1 paragraph 5 to

The enumeration constants are

mtx_plain

which is passed to **mtx_init** to create a mutex object that does not support timeout;

Change 7.26.4.2 paragraph 2 to

The **mtx_init** function creates a mutex object with properties indicated by **type**,
which must have one of the these values:

Committee Discussion

This is a simple set of typos. Accept the suggested TC. Move to Review.

Proposed Technical Corrigendum

Change 7.26.1 paragraph 5 to

The enumeration constants are

mtx_plain

which is passed to **mtx_init** to create a mutex object that does not support timeout;

Change 7.26.4.2 paragraph 2 to

The **mtx_init** function creates a mutex object with properties indicated by **type**,
which must have one of the these values:

DR 415 (N1614) - Review

Summary

The undefined behavior defined in paragraph 6 of 6.5.5 is missing in J.2 and should be added.

Suggested Technical Corrigendum

Add a bullet with the following text to J.2 after bullet 45

If the quotient **a/b** is not representable, the behavior of both **a/b** and **a%b** is undefined (6.5.5).

Committee Discussion

This is just an editorial miss, and should be corrected. Adopt the Suggested Technical Corrigenda as Proposed Technical Corrigendum and move to Review.

DR 416 (N1627) - OPEN

Committee Discussion

We should look at this in unison with DR 424 (N1651), since they address the same basic issues, i.e. underspecification of threads features in the C11. Basically, C11 assumes that the reader already understands pthreads, otherwise the reader is basically lost. Tom is running the papers by Pete Becker to get some feedback. N1651 is a more thorough paper is addressing the issues.

Becker responded; see SC22/WG14 message 12813.

The DRs are basically about underspecification. It was the Committee intent to make it the specification very limited. Those who understand pthreads or Windows threads can work with the specification as is. The suggest changes in both DRs are really a proposal for a revision to the Standard, rather than a DR.

DR 417 (N1628) - REVIEW

More missing items in Annex J

Committee Discussion

Same as an earlier item, editorial misses of items to add to Annex J. Adopt the Suggested Change as Proposed, move to Review.

DR 418 (N1633) – REVIEW

Committee Discussion

Unless stated otherwise, if the argument is a NaN, the result is a NaN, and there is no floating point exception. Are these in the 'stated otherwise' category? There is no Suggested Technical Corrigenda. It's not clear that there's a problem here, as the existing words seem to be clear. Is anyone getting this confused? Unknown. Remove the source of the confusion, the parenthetical comment. Rajan disagrees. Removing it make it more confusing. Add explicit? Preference is leaning to making NO change. That's OK w/Fred. F.10;p11 takes precedence. Returns a Nan, which does not preclude it from returning the same NaN.

Second part is taken care of by the first. No Change. Move to Review

DR 419 (N1635) - OPEN

Committee Discussion

Suggested Technical Corrigendum

7.17.1 add a new paragraph after paragraph 5:

> It is unspecified whether any generic function declared in stdatomic.h is a macro or an identifier declared with external linkage. If a macro definition is suppressed in order to access an actual function, or a program defines an external identifier with the name of a generic function, the behavior is undefined.

J.2 add:

> The macro definition of a generic function is suppressed in order to access an actual function (7.17.1)

These words allow it to work. Jim – why not just call them macros? Would that confuse them with 'generic macros'? 'type generic macros'? The proposed words allow for a lot of flexibility. However, they do not really define anything. Adopt Suggested as Proposed, keep OPEN.

DR 420 (N1647) - REVIEW

Committee Discussion

Add Joseph's words (12778) to the discussion.

Regarding N1647, the second form of for-statement is *not* "Obviously" a typo. The syntax for "declaration", in 6.7 paragraph 1, includes an optional init-declarator-list and a trailing semicolon. It's the proposed change that fails to conform to existing practice.

Use the above as a response, move to Review.

DR 421 (N1648) - OPEN

Committee Discussion

Clark believes the intent of the Standard is exactly what it says, because different architectures have different conventions. It is not an oversight, as the DR suggests. It could be argued that this is a proposal, not a DR.

DR 422 (N1649) – OPEN

Committee Discussion

The current version of the standard doesn't specify to which value an atomic object should be initialized if it is initialized by default.

Are atomic qualified ints treated the same as normal ints, i.e. all are initialized to '0', atomic qualified or not ?

The DR is requesting a change, that has a big impact on the model. It's really a proposal.

DR 423 (N1650) – OPEN

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

Committee Discussion

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

Leave OPEN.

DR 424 (N1651) – OPEN

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

Committee Discussion

Tied to DR 416 – See discussion there.

Leave OPEN

DR 425 (N1653) – OPEN

Section 6.2.4 in p4 and p5 requires implementation defined behavior for accessing objects with thread local or automatic storage from different threads than where they are defined. No such mention is done for objects with *temporary lifetime* in p8. Can they be accessed by other threads? Is this property handled similar to the property for automatic storage duration? Or should this simply be forbidden?

Committee Discussion

Is there any way to access such an object? 6.2.4;p8 says it has automatic storage duration. So, this is not a defect.

Leave OPEN

7. Resolutions

7.1 Review of Decisions Reached

7.2 Review of Action Items

ACTION: Clark Nelson to write up some new words for DR 405.
DONE

ACTION: Clark Nelson to check with Hans Boehm to see it there was any discussion in WG21 (C++) on DR 406, and what, if any, resolution was reached.
DONE

ACTION: Clark Nelson to check into the status of the concepts in DR 407 within WG21.
DONE

ACTION: Clark Nelson to follow the disposition of N1634 in WG21 (N3433) and report such back to WG14.

ACTION: Robert Secord to make the agreed to changes to N1624 for the next mailing.

ACTION: A small editing group review the edits made to N1624.
The editing group will consist of:

John Benito
Blaine Garst
David Keaton
Dave Prosser
Clive Pygott
Robert Secord
Willem Wakker
Douglas Walls

ACTION: Convener to either send the reviewed document to SC22 for PDTS Ballot, or not, as appropriate.

ACTION: Clark Nelson to examine the adoption of the Proposed TC words for DR 402, and revise as needed.

ACTION: Convener to communicate discussion of submitted Defect Reports to the submitters.

8. Thanks to Host

The Committee expressed its thanks to Clark Nelson and Intel for hosting this meeting in Portland, Oregon.

9. Adjournment

Adjourned at 13:36, PDT, Thursday, 10/25/2012

INCITS PL22.11/12-00002
Reply To The Attention Of: Barry Hedquist
PL22.11 Secretary
Email: beh@peren.com

PL22.11 TAG Meeting Minutes (Draft)
23 October, 2012
Doubletree by Hilton Hotel Portland
Portland, Oregon

Meeting convened on October 23, 2012, at 4:15 pm by PL22.11 Chair, David Keaton.

Attendees:

Voting Members:		
Name:	**Organization:** **P – Primary, A - Alternate**	**Comments**
Dave Prosser	Bloomberg - P	
John Benito	Blue Pilot - P	
Jim Thomas	Blue Pilot – A	
David Keaton	CMU/SEI/CERT - P	PL22.11 Chair
Robert Seacord	CMU/SEI/CERT - A	
Daniel Plakosh	CMU/SEI/CERT - A	
Roger Scott	Coverity - P	
P. J. Plauger	Dinkumware, Ltd – P	
Rajan Bhakta	IBM - P	
Clark Nelson	Intel - A	
John Parks	Intel - P	
Clive Pygott	LDRA - P	
Mike Hennell	LDRA - 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
Blaine Garst	Self	

1. Approval of Agenda

Revisions to Agenda: None

Added Items: None

Deleted Items: None

Agenda approved by unanimous consent. (Garst/Tydeman)

2. Approval of Previous Minutes (PL22.11/12-001)

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

3. Selection and Review of US Delegation.

Motion: The US delegation for WG14 meetings during 2013 will be all members and alternates for PL22.11 with the exception of JTC1 Officers and INCITS Secretariat Ex Officio members.
(Walls/Keaton).

Motion approved by Unanimous Consent.

4. INCITS Anti-Trust Guidelines

We viewed the slides located on the INCITS web site.
http://www.incits.org/inatrust.htm

5. INCITS official designated member/alternate information.

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

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

See attendance list above.
14 PL22.11 voting members participated out of 16.

6.1 PL22.11 Members Attaining Voting Rights at this Meeting

Bloomberg

6.2 Prospective PL22.11 Members Attending Their First Meeting

None

7. Members in Jeopardy

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

None

7.2 Members in jeopardy due to failure to attend Meetings.

None

7.3 Members who lost voting rights attending this meeting.

None

8. New Business

8.1 US Position: Systematic Review ISO/IEC TR 24731-1:2007, ISO/IEC TR 18037:2008 and ISO/IEC TR24732:2009.

Three Technical Reports are in Systematic Review. PL22.11, US TAG to SC22/WG14, needs to make a recommended US Position to the INCITS/EB for each TR listed below.

RESOLUTION #1: PL22.11, the US TAG to SC22/WG14, APPROVES the following answers to the Systematic Review Questions for TR 24731-1:2007; TR 18037:2008 and TR 24732:2009.

ISO/IEC TR 24731-1:2007, Extensions to the C Library - Part 1: Bounds Checking Interfaces

1. Recommended action?
   Withdraw
   Comment: ISO/IEC TR24731-1:2007 was incorporated into the latest revision to ISO/IEC 9899:2011.

2. Has this International Standard been adopted or is it intended to be adopted in the future as a national standard or other publication?
   NO

3. Is the national publication identical to the International Standard or was it modified?
   N/A

4. Is this International Standard used in your country without national adoption or are products used in your country based on this standard?
   YES

5. Is this International Standard, or its national adoption, referenced in regulations in your country?
   NO

ISO/IEC TR 18037:2008, Extensions to support embedded processors.

1. Recommended action?
   Confirm

2. Has this International Standard been adopted or is it intended to be adopted in the future as a national standard or other publication?
   NO

3. Is the national publication identical to the International Standard or was it modified?
   N/A

4. Is this International Standard used in your country without national adoption or are products used in your country based on this standard?
   YES

5. Is this International Standard, or its national adoption, referenced in regulations in your country?
   NO

ISO/IEC TR 24732:2009, Extension for the programming language C to support decimal floating-point arithmetic.

1. Recommended action?
   Confirm

2. Has this International Standard been adopted or is it intended to be adopted in the future as a national standard or other publication?
   NO

3. Is the national publication identical to the International Standard or was it modified?
   N/A

4. Is this International Standard used in your country without national adoption or are products used in your country based on this standard?
   YES

5. Is this International Standard, or its national adoption, referenced in regulations in your country?
   NO

Discussion: If TR24731-1 is not withdrawn, there will be two ISO/IEC documents addressing the same technical material, requiring the maintenance of both documents, creating confusion in the market.

MOTION: PL22.11 APPROVES RESOLUTION #1 AS STATED ABOVE. (Garth/Benito)

Post-Motion Discussion: none

Roll Call Vote: (13-0-0)

Bloomberg LLC	Yes
Blue Pilot Consulting Inc	Yes
CERT Coordination Center	Yes
Coverity Inc	Absent
Dinkumware Ltd	Yes
Garst	Yes
Hewlett-Packard Company	Absent
IAR Systems AB	Absent
IBM Corporation	Yes
Intel Corporation	Yes
LDRA Technology Inc	Yes
Oracle	Yes
Perennial	Yes
Plum Hall Inc	Yes
Seymour	Yes
Tydeman Consulting	Yes

9. Next Meeting:

23 – 26 April 2013, Delft, Netherlands with SC22/WG14.

10. Adjournment

There being no further business, the meeting was adjourned at 4:35 PM local, October 23, 2012 (Garst/Prosser) - Unanimous Consent.