Draft Minutes for 23 April–27 April, 2018, MEETING OF ISO/IEC JTC 1/SC 22/WG 14 including PL22.11 meeting April 24, 2018

WG 14/N 2239

Dates and Times

23 Apr 2018	09:00 -12:00	Lunch	13:30 -16:30
24 Apr 2018	09:00 -12:00	Lunch	13:30 -16:30
25 Apr 2018	09:00 -12:00	Lunch	13:30 -16:30
26 Apr 2018	09:00 -12:00	Lunch	13:30 -16:30
27 Apr 2018	09:00 -12:00		

Meeting Location

Red Hat
Building TPB/B
Purkyňova 111
Brno 61200
Czech Republic

Meeting information

Venue information: N 2181

Local contact information

Martin Sebor `msebor@redhat.com`

1 Opening Activities

1.1 Opening Comments (Sebor)

Martin welcomed us to Brno, and WG14.

1.2 Introduction of Participants/Roll Call

Name	Organization	NB	Comments
David Keaton	Keaton Consulting	USA	WG14 Convener
Daniel Plakosh	CERT/SEI/CMU	USA	WG14 ISO eCommittee Secretary
Blaine Garst	The Planet Earth Society	USA	
Rajan Bhakta	IBM	CA	PL22.11 Chair
John Parks	Intel	USA	
Fred Tydeman	Tydeman Consulting	USA	PL22.11 Vice Chair
Barry Hedquist	Perennial	USA	dialed in
Tom Plum	Plum Hall	USA	dialed in
Martin Sebor	Red Hat	USA	host
Larry Jones	Siemens PLM Software	USA	WG 14 Project Editor, dialed in
Aaron Ballman	GrammaTech	USA	
Clive Pygott	LDRA	UK	
Jens Gustedt	INRIA	France	WG 14 fallback Editor
Lars Gullik Bjønnes	Cisco	USA	
Florian Weimer	Red Hat	USA	
Arjun Shankar	Red Hat	USA	
Freek Wiedijk	Plum Hall	USA	
Robert Seacord	NCC Group	USA	
Philipp Krause	Albert-Ludwigs-Universität Freiburg	Germany	
Victor Gomes	University of Cambridge	UK	
Kayvan Memarian	University of Cambridge	UK	
Peter Sewell	University of Cambridge	UK	
Keld Simonsen		Denmark	dialed in

1.3 Procedures for this Meeting (Keaton)

The Meeting Chair and WG14 Convener, David Keaton, announced that procedures would be as per normal. Everyone was encouraged to participate in the discussion and straw polls.
Straw polls are an informal WG14 mechanism used to determine if there is consensus to pursue a technical approach or possibly drop a matter for lack of consensus. They are voted on by a show of hands for people that approve, reject or abstain, respectively (denoted by #approved/#reject/#abstain in the minutes) on the poll question. Straw polls are not formal votes, and do not in any way represent any National Body position. National Body positions are established in accordance with the procedures established by

each National Body.
INCITS PL22.11 members reviewed the INCITS Anti-Trust and Patent Policy Guidelines at: `http://www.incits.org/standards-information/legal-info`
All 'N' document numbers in these minutes refer to JTC1 SC22/WG14 documents unless otherwise noted.
The primary emphasis of this meeting was to review the progress of our subgroups, work on Clarification Reports and review documents that were presented to WG14.
Participants are requested to register on ISO's site for this meeting.
David Keaton is the meeting Chair. Jens Gustedt is the Recording Secretary.

1.4 Approval of Previous Minutes N 2187

The previous minutes have been amended for typos, etc. The previous minutes are approved by unanimous consent. The final approved Albuquerque minutes are N 2238 The draft Brno minutes are N 2239

1.5 Review of Action Items and Resolutions

ACTION: Blaine to reconcile N 2019 and N 2026 for DR469.

on hold

ACTION: Convener to coordinate with WG21 on the mechanics adding a 'C' or 'P' designated papers for proposals to WG14.

open

ACTION: Convener to update SD3 to go through the obsolescent features and future directions to decide what to remove for C2X.

done, this is now document N 2232

ACTION: Martin to split N 2145 into two proposals. One to define Generic functions, another to work on compound literals.

done

ACTION: C++ liaison members to indicate to WG21 that WG14 is good with N 2160 (Comma omission and deletion).

done

ACTION: Larry to write a paper with any objections to N 2161 (left shift behavior).

open

ACTION: Convener to update SD3 to point to N 2165 for attributes.

done

ACTION: Convener to add N 2186 to SD3.

done

ACTION: Convener to write a response for N 2174 as an N document.

open

ACTION: Martin to write papers for the other items in DR496 that need attention or that he disagrees with.

done

ACTION: Clark to Supply new words for DR499 based on reflector message 14632.

open

ACTION: Rajan to invite Clark specifically to the CFP meetings (not just in the reflector) to help resolve the cbrt example issue (CFP DR16).

done

ACTION: Convener to send the CPLEX document to the national bodies that agreed to participate on the work item.

open

1.6 Approval of Agenda N 2233

- Added:
  - 8.1: Discussion of "Reentrant APIs" N 2241
  - 8.2: Creation of a "memory model" working group
- Refused because considered too late by some participants:
  - Discussion of documents N 2245 and N 2246
- Deleted: None

approved

1.7 Identify National Bodies Sending Experts

UK, USA, Canada, Germany, Denmark

2 Reports on Liaison Activities

2.1 sc22

nothing major

2.2 PL22.11/WG 14

1. New PL22.11 Chair - Rajan Bhakta

   two ballots out

2. WG 14

   - Updated version of Standing Document 3 is N 2232
   - update on C17:

     C17 was sent to ISO on 2017-12-17.

     The editors had to exchange several versions back and forth with ISO.

     C17 FIDS is now in ballot.

     The process was quite tedious: they pushed hard on modifications that are not even covered by written ISO documents.

     Until the ballot is closed, we are not allowed to vote introducing new things into the standard, so people know what they are voting on.

     The document with diffmarks to our previous internal document for the LaTeX transformation is N 2240. This document is password protected.
   - N-documents

     There is an ongoing discussion with ISO about our strategy to publish working drafts.

     We are leaving the procedures in place until the discussion with them is closed.

2.3 PL22.16/WG 21

Barry: there is a document on specific differences between C++ and C, hopefully will make people aware of what's going on.
Action: Barry to make this document available to us.
Aaron: more papers to come for WG14 from WG21.

2.4 PL22

2.5 WG 23

Clive: The C language supplement for the WG23 report is in the final stages of editing prior to going out for vote. He thanks the WG14 committee for helpful comments during and after the Albuquerque meeting, and input from the UK C panel. The next version of the language independent "/main body/" is also getting close to going out for vote, probably by the end of 2018 or early 2019. The main additions are parallelism and inheritance. WG23 has started working with WG21 on a C++ supplement.

2.6 MISRAC

Clive: progressing slowly

2.7 Other Liaison Activities

none reported

3 Reports from Study Groups

3.1 C Floating Point activity report

- Continuing to work on DR's to the TS's.
- Continuing to work on IEEE 754:2018 binding.
- Looking for other possible issues when comparing binding to 754:2008 to validate we are binding correctly.

Blaine: What is the process to fold changes to the TS into C?

- Modifications to parts 1 and 2 are aimed to go into C2x.
- Parts 3, 4, and 5 should be rebased on C17 and new TS should be created.

3.2 CPLEX activity report

- There hasn't been a lot of participation from experts in the field.
- The contribution of the OpenMP community has increased via Tom's participation, though.
- It is suggest to wait before sending out the document until changes have completely been integrated.

3.3 C Safety and Security Rules Study Group

- There are fortnightly meetings to discuss Misra.
- *Blaine:* process issues, are change requests expressed as TC?
- *Clive:* What is discussed is essentially a new document, but there is disagreement what is "safety and security".
  - is it a set of rules for new code?
  - or is it a set of rules to assess existing code?

4 Teleconference Meeting Reports

4.1 Report on any teleconference meetings held

The C17 editorial review committee met on Dec 6, 2018. It agreed that all changes required by WG14 had been integrated by the editors.

5 Future Meetings

5.1 Future Meeting Schedule

- Oct 15-18 2018, Pittsburgh
- spring 2019, Denmark (tentative)

5.2 Future Mailings

- post-Brno, May 21, 2018
- pre-Pittsburgh, Sep 17, 2018
- post-Pittsburgh, Nov 13, 2018

6 Document Review

6.1 Harmonizing `static_assert` with C++ N 2197

Presented by Aaron.
Blaine: How is that implemented?
Aaron: not very difficult in clang
Rajan: Likes it. A minor editorial remark, there should be better wording for the optional argument. Considered editorial.
Florian: There may be intersection with Martin's proposal on `assert`.
ACTION: Aaron will look into that.
Blaine: What is the procedure to integrate this request into C?
Rajan: Suggest to move it into SD3.
Martin: Why not informally move it into C2x?
David: At beginning of the Pittsburgh meeting will be going through SD3.

Straw poll: Shall we add the proposed changes of N 2197 to SD3?
17 approved, 0 reject, 1 abstain

ACTION: David to add the proposed changes of N 2197 to SD3.

6.2 Adding the `u8` character prefix N 2198

Presented by Aaron. A problem is to formulate this the same way as for C++.
Blaine: How can we be sure that implementations choose the right encoding?
Aaron: There are macros for UTF16 and UTF32.
Rajan: He would not want have all of UTF8 possible in `u8` characters.
Jens: There is no ambiguity for `u8`, this talks about one byte code presentation in UTF8. This is just the intersection of Unicode and ASCII.
Florian: What happens if the character doesn't fit into one byte?
Aaron: If this is something that doesn't fit into one byte then we should have UB.
Philipp: This should be implementation defined just as it is for plain character constants.
Florian: `u8` should not be in the paragraph for wide strings or characters, because these are not wide characters.
Jens: Is this `char` or `unsigned char`?
Aaron: Wants to check which one is really needed for compatibility with C++.
Rajan: Would we need a macro as for `u` and `U` character constants?

Aaron: He doesn't think so.
Action: Aaron to present new revision that integrates the discussion.

6.3 Effects of repeated calls to `mtx_init` with the same mutex N 2190

Martin presents.
Blaine: Likes to see a more general fix to the threads section, because there are similar changes to make for `cnd_t`.
Florian: Not clear if `mtx_t` are just handles or the mutex objects themself. The interface should allow both types of implementations.
David: Doesn't want it to have explicit UB.
Jens: It is currently defined behavior, because `mtx_init` has to accept any valid pointer to a `mtx_t` as an argument.
Blaine: Put the proposal into SD3.
Martin: Wants to make progress.
Action: Blaine to write up changes to threads that are necessary.
Rajan: Do we want incremental improvements like this or to we want have a global solution.
Jens: wants both.
Rajan: This paper in isolation is not enough.

- It needs the modification of terminology from "create" to "initialize".
- It needs the corresponding change to `mtx_destroy` in N 2193.
- It doesn't talk about static `mtx_t` that are default initialized.

6.4 Effects of copying a `mtx_t` object N 2191

Martin presents.
Jens: This is the operational difference that he wants to have.
Blaine: We already have a phrase for a similar purpose in 7.21.3 p6 for `FILE`.
Rajan: Allows that some implementation may still allow for copying.
Aaron: In C++, a `mtx_t` can't be copied.
Martin: It has to be UB because it is so in POSIX and we want to allow thin wrappers.
Jens: Is it allowed to copy mutex and initialize the copy afterwards?
Martin: In POSIX this is not allowed.
Jens: So it should state "copying is undefined".
Martin is ok with that change.
David: Likes the analogy to the FILE type.

6.5 Effects of calling `mtx_init` with an invalid or null pointer N 2192

Martin presents: Makes it consistent with all other functions in the C library.
Blaine: The current resolution of DR493 leaves the return value for the case of a null pointer unspecified.
Rajan: This is not what is written in the Albuquerque minutes.
Martin: Concludes that everyone is happy with the proposed change. Only highlighted changes are proposed, no overwrite of C11 into C17 changes is intended.

6.6 Effects of calling `mtx_destroy` with a uninitialized mutex N 2193

Martin presents.
Larry: Remove the occurrence of shall in the proposed change.
Rajan: Doesn't like the term destroy.
Florian: Some implementations **require** to destroy a mutex before the end of its lifetime.
Blaine: The old phrase tells the user that there may be resources that can be wasted by not calling `mtx_destroy`.
Martin: This is not a normative requirement on neither the implementation nor the user.
Rajan: The status quo with "resources" is not satisfactory. Put the part with the resources in a footnote.
Florian: There might be a synchronization problem, between different threads.
Jens: Doesn't think so, because the standard is clear that synchronization is done accross `mtx_t` functions with release consistency on call to the `mtx_unlock` function and acquire consistency on the call to the `mtx_lock` or a successful call to `mtx_trylock`.
David: There is a missing comma after "locked" to make the phrase unambiguous.
Blaine: ok for the comma
Peter: "In use" is implying "happens before" relation. POSIX is bogus because they don't have the "happens before" relation.
ACTION: Martin to present new versions of these papers that take their interrelationship into account.

6.7 The `deprecated` attribute N 2214

Aaron: only changes on the surface in this version.

Florian: In Java such functions can in turn use other deprecated functions.
Aaron: This a question of QoI how much warning implementations would give in that case.
Martin: This would provide a escape hatch from the deprecation mechanism.
Florian: You could use a define to deactivate a specific attribute.
Blaine: Are empty double square brackets valid syntax?
Aaron: yes
Aaron: Also "deprecated" is not a keyword and so using the identifier for a macro should be allowed.
Martin: We should say something about interaction between macros and attribute identifiers.
Florian: We have to specify what we want.??
Jens: Reminds of paper N 2175. It identified this problem of attributes names invading the identifier name space that is dedicated for user code. He still is convinced that this is a bad attitude and will cause much more problems in C than it does in C++.
Aaron: QoI usually allows to add `__` in front.
Florian: The `__` approach is annoying for C++ and they even would like to get rid of it.
Aaron: Do we want attribute tokens more like keywords?
Martin: Make it more clear to implementators how we want to protect user code from macro expansion.
Aaron: Attribute tokens can be replaced by a macro.

6.8 The `nodiscard` attribute N 2215

Rajan: There is no mention for non-consistent annotations.
Florian: Is this just syntax based, or is it intended to be data flow analysis?
Aaron: This is just syntax based.
Florian: Make this more explicit.
Rajan: Is it intended to be handled consistently between different TU?
Aaron: No, each TU gets the declaration (with or without) attributes that it sees and acts accordingly.
Florian: How are function pointers handled? Does the attribute apply?
Aaron: We could do if it is considered to be useful. He will look into it and eventually coordinate with C++.
Aaron: Would we like to have a proposal that identifies functions of the C library that should be annotated with this attribute?
There seems to be consensus that this is a good idea.
Clive: Wants a pragma to switch this kind of diagnosis on and off.

6.9 The `fallthrough` attribute N 2216

Philipp: He doesn't like the `fallthrough` attribute, because he thinks that it implicitly deprecates fallthrough in `switch`.
Aaron: Is already implemented by C++ and has been useful.
Florian: Compilers will do that anyhow.
Clive: This is useful for Misra.
Aaron: Expectation is that not to fallthrough is the common case.
Jens: Seconds Philipp in that he thinks that this somehow deprecates fallthrough.
Florian: Having an "inverse" attribute to label `switch` statements would be more complicated to specify.
Philipp: doesn't see that.
JP: All these are code annotations that turn on or off diagnostics.
Philipp: No the other ones bump up diagnostic, this one is to turn them off in a specific place.
Clive: What is the cost of adding attributes, can't we have both approaches?
Aaron: Basically the cost is specification, because implementations are allowed to ignore attributes, anyhow.
Blaine: This will give a lot of warnings on valid code.
Aaron: The default in existing compilers is not to have the diagnostic on.
Florian: Static analysis tools can profit from this attribute.
David: The idea has been around since 30 years.
JP: Existing practice, this just adds as common syntax.
Rajan: (editorial) There is need to change "warnings" to "diagnostics".

6.10 The `maybe_unused` attribute N 2217

Rajan: What is happening with all these papers? Replace the papers in SD3 with these ones.
Blaine: The least interesting of these attributes.
David: Would be most used by him.
Florian: Gcc has two different attributes. There is one (`attribute(used)`) that tells the compiler to emit the symbol unconditionally, and then not to warn if it is not used.
Aaron: There is no expectation about symbol generation, so the one that is proposed here corresponds to their other one (`attribute(unused)`) that is purely a syntactic check.
ACTION: David to change to the current versions in SD3.

6.11 Removal of Bullet 67 in Annex J.2 N 2201

Martin presents. It just is redundant.
Larry: no strong opinion
Martin: It came up because people go through the Annex to produce examples, e.g for Misra.
Jens: Prefers to keep it in.
Rajan: Do we know which of the other bullets make this redundant?
No consensus to delete the bullet.

6.12 Assert Expression Problematic For C++ N 2207

Martin presents.
Jens: It is how C talks about Boolean evaluation.
Florian: They tried to use the ?: operator.
Rajan: Why can't C++ just undef and redefine?
Florian: Problems when passed to intrinsics.
Rajan: This sounds like a QoI issue.
Martin: glibc rewrites headers
Jens: would just removing `"== 0"` help?
Martin & Florian: no
David: The proposed text doesn't even help.
David: Shouldn't we just introduce a "recommended practice" section that helps avoid the implementation as explicit comparrison with `0`?
Rajan: (editorial) The macros `false`, `true` and `bool` should not be used in this part of the text, but their replacements.
Florian: We will look if C++ could work with this idea.
Larry: Wants just to add recommended practice, but not to change the normative text.
Martin: We will put together a new document that just introduces recommended practice.

6.13 Library Functions And Compound Literals N 2208

Martin presents.
Jens: He sees the need for some change, here, but is not satisfied with the direction. Why should we constrain C library implementors not to use macros, where this is possible since always? He prefers to go for a completely different solution along the following:

> Unless used in a `#ifdef`, `#ifndef` or `defined` construct, a call to a macro that replaces a library function call shall not have observable behavior that distinguishes it from such a function call. FOOTNOTE In particular, such a macro call must lexically be treated such that macro arguments that contain commas (compound literals) are handled as if passed to the corresponding function, and all sequencing properties of a function call must be observed.

Rajan: Against prohibition of macros. Inline is not the same. Other solution would be to take more care with compound literals.
David: Just add a footnote that recommends to put parenthesis around compound literals.
Martin: This doesn't address the existing code base.
Martin: Existing implementations that use macros for C library functions could change, and maybe have a less efficient implementation.
David: There are much more uses of `memcpy` in the wild than of compound literals that are used in C library function calls.
Blaine: Not sure that "compound literals" are an important issue. The cost for this change too high.
Rajan: We are focussing on "make compound literal work with the C library". High cost for minimal gain.

Straw poll: Is the committee interested in the resolution proposed in N 2208?
2 approved, 10 reject, 5 abstain

6.14 Clarifying the C Memory Object Model: Introduction to N 2219 - N 2222 N 2223

Peter presents.

6.15 Clarifying Pointer Provenance (Q1-Q20) v3 N 2219

Peter presents an updated version that can be found at `http://www.cl.cam.ac.uk/~pes20/cerberus/notes97-2018-04-21-provenance-v4.html`
Martin: Using "padding bits" in pointer values is undefined in gcc.
Florian: There are extensions in gcc to support arithmetic in integers that are derived from pointer values.

- Q6 in this document:

  Robert: Does this suggest to integrate more operations on pointers into the C standard?

  Aaron: We should try to validate existing practice, so this wouldn't help.

  Blaine: We can't assume that `intptr_t` is available.

  Philipp: It would be be surprising to have bit operations on pointers that are not consistent with pointer addition or subtraction.

- The no-provenance option:

  Martin: What is the scope of the nop-providence option?

  Jens: Why make provenance optional?

  Peter: Linux e.g may depend on no-provenance.

- Q11, XOR linked list idiom:

  Jens: How to handle re-constructed pointers?

  Peter: Wildcard provenance was a first try to do that.

  Blaine: Please add an example that masks pointers with XOR.

  Florian: Intel MPX has a pointer check that uses sideline information and thus a simple byte-wise copy loop could not work. But this is dead anyhow.

- Q17, type punning

  Martin: Do you want to make the answer to Q17 implementation defined?

- Q19, pointers via IO

  Forian: This is basically done for dynamic linking.

  Rajan: Thinks that Q19 has UB if the value in the file changed in the mean time.

- Q20, concrete address

  Blaine: What about memory maps?

  Blaine: There are hardwares that have different "kinds" of memories with different properties.

  Philipp: The embedded C standard may have provisions for that.

- general questions

  Peter: What is the granularity of provenance?

  Jens: He would like to have it on the level of definition/allocation.

  Peter: Has the impression that there is agreement on the provenance approach.

  Florian: Concerned about transfer of provenance between pointers.

  Peter: This is possible through some weird arithmetic.

  Martin: Likes to have stricter rules such that require some kind of annotation that enforce break outs from the provenance model.

6.16 Clarifying Trap Representations (Q47) v3 N 2220

Handled together with the following.

6.17 Clarifying Unspecified Values (Q48-Q59) v3 N 2221

Peter presents.
Martin: Who thinks that copying a partially initialized struct is UB? 3 participants.
Florian: Is `x->a = 0` reading `*x`?
David: No the standard does not define `->` via the reference operator `*`.
Blaine: Wants default initialization and promote the use of `calloc` instead of `malloc`, such that objects are generally initialized.
Rajan: This does not necessary help for everything, since even with `calloc` pointers or floating point member may then be initialized but still have a trap representation.
Peter: Let us move to mail discussion, working group or off schedule discussion.

6.18 Further Pointer Issues (Q21-Q46) N 2222

Not discussed because lack of time.

6.19 Atomic pointers in expressions N 2209

Martin presents.
Jens: What about `void _Atomic*` for 6.5.15?
Aaron: Clang doesn't allow for that. They seem to diverge here, because they think that atomic types have to be complete.

David: Formulation for simple assignment should talk about pointers *to* atomic type.
Philipp: Pointer comparison should be handled consistent with assignment. We can assign from an unqualified pointer to a qualified pointer. But we can't assign from an unqualified pointer to an atomic qualified pointer. Since we can't assign these types of pointers, such pointers also should not compare equal.
ACTION: Martin to integrate the raised issues and present a new proposal.

6.20 Signed integers are two's complement N 2218

Aaron presents.
Aaron: This is an attempt to re-standardize C on a world where only two's complement is available.
Philipp: There are many C implementations out there that we don't know of.
Florian: This is just about the type, not about overflow handling.
Aaron: This is proposing changes where C++ has found consensus, not proposing changes where there wasn't.
Clive: How many of these compilers are C11?
Fred: For atomics the C standard alread requires to have two's complement.
Robert: We already had that discussion for C11.
Martin: How much effort to we have to go through to know if there are still implementations that have different integer types?
Philipp: Is there an advantage in only allowing two's complement?
Aaron: Many people just already assume that.
David: It is also about adding slack for future implementations.
Blaine: The same question was asked in 1993 on comp.lang.c. He supports the change.
Freek: Should we also require similar properties about endianess?
Florian: Users make a lot of effort to accommodate different integer representations. Such code is a lot of work and it is effectively never tested because users don't have test platforms for it. Live would easier.
Rajan: What is the intent? Remove it, make it obsolencent?
Aaron: For the moment it is fact finding.
Blaine: Wants to clear that up for C2x.
Robert: Would this change 6.3.1.3 p3?
Peter: Unisys is based on C89.
Florian: TS 18661 part 1 defines a bunch of macros with `"_WIDTH"` names. So there is influence of this proposal to the TS 18661.

Jens: The presented document has a problem with the formulation for `bool`.
Florian: This is overselling, this is only part of the way. The removal of overflow provision are missing in his POV.
Rajan: Wants to see a vote just on the integer representations.

Straw poll: Is WG14 comfortable in removing *ones complement* and *sign and magnitude* from the C standard.
14 approved, 0 reject, 3 abstain

Aaron: Are there other issues that we want to bring back to WG21?
Rajan: `bool` padding and overflow.
Robert: To have right shift defined to be arithmetic would be problematic.

6.21 Equality With `true` N 2229

Martin presents.
Jens: In favor. This is a simple and clean solution.
David: Add words to the footnote to explain the casting trick.
Rajan: Some code could break with this change.
Aaron: Maybe it would be better to make it "recommended practice" than a footnote.
Blaine: For the example this could now introduce a warning for the inverse `"!= false"` .
Blaine: `sizeof(true)` may change
Martin: This is the simplest change to accommodate the problem with the `ctype.h` functions.
Freek: Type change for the two constants. So users can know if they have some implicit conversions from integer to `bool`.
Blaine: This is invention. So we should be sure about the motives.

Straw poll: Do we want to change the types of the expansions of the macros `false` and `true` to be `_Bool`?
12 approved, 1 reject, 5 abstain

Fred: Will test this change on his test suite.

6.22 Thread-local state for `getenv`, `strtok`, and `set_constraint_handler_s` N 2225

Florian presents.

Florian: the `strtok` function started all of this.
Rajan: Why constrain it as much, that is, make it UB?
Florian: We could use similar wording as in the other papers.
Fred: Shouldn't we just say that `strtok` should not be used in more than one thread.
Philipp: Insert the following sentence between the 3rd and 4th sentence of 7.24.5.8.

> When calls in the sequence happen from different threads, the behaviour is undefined.

Rajan: We are just looking a more acceptable version of this?
Martin: Better use `strcspn` as an alternative in the footnote.
Robert: There are actually three alternatives,

- `strcspn`
- to move `strtok_s` from Annex K into the library
- replace `strtok` by `strtok_s`.

Rajan: Use the updated language with "happens before" also for `getenv` or introduce a new interface.
Martin: Personal preference is not change Annex K but remove it.
Robert: Likes to impose thread local storage for `set_constraint_handler_s`.
Aaron: Shouldn't we take this out and delegate this to the secure coding group?
Robert: They are not the right group to study this.
Blaine: Revisit this topic when we review SD3 for C2x.
Rajan: Drop this and wait for the changes from the review.
ACTION: David to add N 2225 to the bullet 12 of SD3
ACTION: Florian to update the wording.

6.23 Thread-local state for the program locale N 2226

Florian presents.
Florian: Another solution would be to use POSIX functions.
Peter: Do you really mean thread local storage duration or just thread specific semantics?
Jens: One problem is in the locale mechanism as such.
Jens: How to inherit a locale from the creating thread of a new thread?
Keld: Inherit the locale from the "main" locale object?

Florian: This is basically the POSIX model.
Keld: Wants full support for thread safety.
Martin: Are you in favor of introducing something like `uselocale` from POSIX?
Keld: Add locale as a parameter to most functions that need it.
Rajan: This should be added to the threads option.
Keld: Prefers to introduce re-entrant APIs.
Philipp: It should be possible to come up with a simple thread safe interface just for `set_locale`.
Florian: Consensus seems to be to use something along the lines of `uselocale` from POSIX.
Martin: Will this preserve the current property that any thread can change the global locale?
Florian: yes
ACTION: Florian to propose a paper that features something in the line of `uselocale` from POSIX.

6.24 Thread-local state for library functions with internal state N 2227

Florian presents.
Philipp: `rand` should be handled careful because this affects the sequence of random numbers that a thread sees.
Peter: Thinks the actual text is ambiguous who is responsible for avoiding races: the implementation or the users.
Florian: The intent seems to be to force applications to have a locking mechanism around the calls.
Rajan: The second change effectively allows to use thread local state such that observed sequences can be different.
Jens: Not a good idea. The random numbers will strongly be correlated between threads.
Philipp: The srand function would seed all threads to `1` that don't do that explicitly. So all random sequences would be the same.
Florian: Probably thread local for `rand` is a bad idea. What about forcing synchronization?
Martin: Are there existing implementations that use thread local storage for `rand`?
Jens: Adding locks to `rand` has increased costs.
Philipp: `srand` introduces reliable sequence of pseudo-random numbers.
Blaine: We need a new random function.

Rajan: We should not remove the "there can be data races" phrase because this would penalize applications that don't use threads.
Jens: There is the `rand48` family of functions from POSIX that has re-entrant versions. These are well experienced and could just be used without pain.
Aaron: What do MS compilers in that case?
Keld: Copy re-entrant functions from other standards.
Philipp: We could losen the constraints that are enforced (e.g by an explicit algorithm in POSIX) to allow for better quality pseudo-random numbers.
Martin: Where is the problem with irreproducibility?
Philipp: Because every thread's state would be initialized with `srand(1)`.
Jens: You can't protect yourself against that by external synchronization.
Martin: What are the consequences for the proposal?
Blaine: Maybe we should make `rand` obsolecent.

Discussion moves to the functions other than `rand`
Peter: Is this considered to be an improvement for the other functions because user code should have locking to be portable?
Florian: Make the applications as before, or add other interfaces.
Blaine: Require that a new interface is thread safe.
Jens: There should be feature test macro for each of these implementation defined functions.
Florian: Should such feature test macros be per function?
Philipp: Should perhaps be per group of functions.
Peter: Avoid thread concurrency problems scattered all around the standard.
Robert: Combination of states is not clear.
Martin: What is the benefit of making it implementation defined?
Florian: There is a benefit in not reserving global state if it is never needed by the application.
Martin: Users should not use the call with `0` as argument in a thread context. Make the `0` argument for the `mbstate_t` obsolecent.
Rajan: Leave this as is.
Keld: ok with obsolencing
Robert: This would make it UB.
ACTION: Florian to send something to the reflector.

6.25 Thread-local state for library functions returning pointers to internal state N 2228

Florian presents.

Martin: The pointer becomes indeterminate?
Peter: Should be formulated with "*happens before*".
ACTION: Florian to change the wording such that it uses "*happend before*".

6.26 Generic function pointer N 2230

Martin presents.
Philipp: Likes the proposal.
Peter: Why does `void*` not work?
Martin: Function pointers might be wider than data pointers.
Philipp: Even POSIX deprecates `void*` for that.
Arian: The conversion back should also be an implicit conversion.
Blaine: Why is the proposal to use a header type?
Jens: Why does this `typedef` a pointer type and not the function prototype?
Martin: He wouldn't be opposed.
Blaine: Is this used for storage of the pointers?
Rajan: Likes the idea. Prefers language and not library.
Blaine: Proposes to use existing keyword.
Jens: Why the attribute in the example code? Is this to allow for the implicit conversion?
Martin: yes.
Rajan: Prefers explicit conversion for the case such a pointer is called directly.
Florian: Prefers implicit conversion for assignment, because once we have a cast the compiler doesn't diagnose if the source pointer type is not a generic function pointer.
Florian: Direct calls of a generic function pointer would still need a cast, anyhow, because otherwise the calling prototype would not be available.
Arian: Do exactly the same thing as for `void*` to be consistent, that is implicit conversion to and from the generic pointer type.
Florian: If we chose the "language" version it should not use the empty `()` syntax to stay compatible with C++.
Rajan: It seems to be consensus that we want such pointers.

Straw poll: Do we prefer a core language solution to introduce generic function pointers?
14 approved, 0 reject, 5 abstain

6.27 `char8_t`: A type for UTF-8 characters and strings N 2231

Presented by Aaron.
Jens: Macros for atomic are not usefull because they are per conversion rank, anyhow.
Philipp: Is this proposal usefull?
Rajan: Has not seen problems with compilers.
Florian: Only change of type to `unsigned char` is problematic.
Aaron: Locale and threads are not well specified.
Rajan: His users deal with it through locale. The new proposal breaks on types.
Florian: Change `uchar8_t` to be a "new" character type.
Blaine: Would that be an integer type?
Jens: Compile time handling of these coding translations is good. Hopefully the changes concerning the type are compile time detectable, so there are no implicit semantic changes that would go undetected.
Aaron: There could be problems with type generic expressions, that all of a sudden chose a different branch because a different type is presented to them.
Aaron: Is anybody really oposed to that proposal?
Blaine: There is no known implementation.
Rajan: There seems to be a C++ implementation.
Aaron: An example implementation should have a wide range such that we can test a large code base for occurrence of these features.
Martin: Do we want features that have no implementations to go into C2x?
David: Traditionally we asked for at least two implementations. Having a proposal voted into C++ is not enough.
Blaine: C should not be experimental.
Florian: The main concern is backwards compatibility. C++ has a different mechanisms here. E.g for functions that receive such strings, function overloading could provide an "old" version for a "`u8` strings are `char`" implementation and a "new" version for a "`u8` strings are `unsigned char`" implementation.
Lars: Backwards compatibility is not an issue for C++ because this feature is rarely used.
Rajan: C++ is a different language and has a different user base, so there might be problems in C that wouldn't appear in C++.
Blaine: There could be ABI issues that we don't know of.
Aaron: Is there over my dead body POV? Critical points are:

- There is no implementation.

- We can't assess backwards compatibility, yet.

Rajan: There is a dynamic solution (with locale) for the problem, so he doesn't see the need of the proposed changes.
Jens: This dynamic solution (with locale) has problems with multi-threading.
Blaine: Seconds Rajan, because there is a working solution.
Florian: This solution can't use mutexes because libraries may have multiple origins.
David: Aaron should note the connection of this issue with iteraction between locale and threads, see point 6.23 and document N 2226.

6.28 `deprecated` Attribute Existing Practice N 2236

Martin presents.
Florian: Is the return type of a function deprecated or is the function interface deprecated? This is ambiguous in the gcc implemenation.
Aaron: The attribute syntax is more regular than gcc's version.
Martin: Is the function `fi` in the example a possible use case that should provoque a diagnostic?
Aaron: Yes, that is how he understands this.
Martin: Is there a use case where the behavior of g++ is useful?
Florian: We should do it as the majority of implementation does, here.
Aaron: There are two-level usages that don't deprecate the public interface but deprecate it internally.
Jens: There is a possibility to distinguish according to the argument to the deprecated attribute.
Florian: But the users would have to know the string.
Martin: Do we want examples to include into the specification such that we can give guidance to users and implementors?
Aaron: Finds that useful.
Blaine: Not keen to have examples but would like to have normative text.
Aaron: This would be tricky because attributes can be ignored.
Rajan: Only problem is that are too many diagnostic. This is just unimportant noise.
Martin: The `f1` function has no portable way to specify that it should not issue an diagnosis.
Aaron: All this is non-normative. He suspects that implementors wouldn't diagnose `f1` because users wouldn't want it. He wants to add some "recommended practice" section.
Rajan: Maybe we need to add another mechanism to *not* to warn.

Blaine: We already see divergence. So convergence will not happen on its own. So we have to say something.
Florian: (1) It is hard to issue warnings from system headers. (2) gcc has a mechanism to deprecate an identifier (independed of its definition) and to add `_Pragma` warnings to macro expansions.
Martin: Does glibc use deprecate attribute for system headers and types?
Florian: No they don't deprecate types.
Martin: Has enough input to go forward to create examples.

7 Clarification Requests

7.1 Discussion on the Clarification Request Process

Blaine:

- We don't do "defect reports" (anymore) but clarification requests.
- What will be the process for documents?
- We have been very strict for changes on C, but don't have to do that this way anymore. How will we have to track the changes for C2x?

Rajan: Small versus big changes is not a major issue. The important difference lies between new features and changes.

- DR → corrigendum
- feature → proposals

Aaron: This is how WG21 works basically. A corrigendum is a change where we wished we could go back in time and correct the text as it was written, then, such that it better reflects the intent.
Blaine: Wants some kind of web based tracking based on the working draft, with a complete set of changes.
David: N 1540 is an example of an Editor's Report.
Florian: Are there diffs to the LaTeX document?
Jens: Yes there are.
Philipp: Is there (or will there be) a rationale document?
Blaine: This is a question of processing time. We currently don't have the work force to provide this.
Jens: A working solution could be to have a rationale document that changes incrementally with all modifications that are voted into C2x. Therefore we

would need a separate "rationale" section for every proposal that could be maintained along with the proposed (suggested) TC.
Blaine: There is objection from ISO to give access to the git, but read-only access for the committe is ok.
(Note from the editor. I don't think that there is any ISO requirement there for our internal procedure. We could give write access to any collaborator we want, as long as this is not public, traceable and the editors have the last word.)
Blaine: The documents for the latest progress reports, are N 2242 and N 2243.

7.2 IS 9899:2011 Clarification Requests (successive revisions N 2148, N 2243, N 2244)

- DR 432 no further discussion, moved to C2x
- DR 467 no further discussion, moved to C2x
- DR 476 no further discussion, moved to C2x
- DR 488 no further discussion, moved to C2x
- DR 494 no further discussion, moved to C2x
- DR 497 no further discussion, moved to C2x
- DR 498

  Philipp: for C2x we could require more, if there is no shift state to keep track of.

  Still, the general feeling is to make progress.

  moved to C2x
- DR 502

  moved to close

  Martin is invited to join the work with Jens and Clark on flexible arrays.
- DR 486 and DR 495

  Blaine: wants to identify a set of people to work on these

  Jens: taking ownership on these two.

ACTION: Jens to work on a more complete document that addresses inconsistencies concerning atomics (language and library).

- DR 469, 479, 493

  Blaine: There are similar issues for `cnd_t` as for `mtx_t`. He will try to identify all the issues concerning threads.

  Florian: Isn't this missing the total lock order for `mtx_t`?

  Blaine: Wants to improve the text to use vocabulary of acquire-release for `mtx_t` more consistently.

  David: We should be careful with existing implementations.

  Florian: We need semantics that existing implementations may currently not provide.

  Martin: POSIX has not yet adapted C11 threads.

  Blaine: Maybe implementations will need to add memory barriers in their wrappers to comply to C11. But the semantics should be implementable inside the wrappers.

  Blaine: Will make a note in committee discussion that these three DRs are to be resolved jointly and he is volunteering to produce a paper for that.

  ACTION: Blaine to write a paper that identifies more apparent problems for threads (language and library).

  Rajan: There is a truncation in Blaine's document that is missing a 2016 discussion.

  Martin: Is worried that we miss out necessary changes that are already identified.

  Blaine: Plans to go through these systematically such that none of the items that had already been identified will be missed.

- DR 496

  Rajan: editorial, some date issue in the write up of DR discussion.

  Martin: should we add words that no new types can be added inside `offsetof`?

  Straw poll: Do we want that defining a new type inside a call to the `offsetof` macro is a constraint violation?

9 approved, 0 reject, 4 abstain

ACTION: Martin to write up a proposal that forbids new types to appear in calls to `offsetof`.

Florian: Is there another place where defining a new type causes the same kind of problems?

Martin: Wants to make progress on this one.

- DR 499

  Blaine: We should find an owner for this.

  Rajan: There is no record on community discussion why his original solution doesn't work?

  David: May have been related to DR502.

  Martin: Do you mean "structure layout"?

  Rajan: new version on the reflector

  Blaine: Anonymous struct keeps layout **and** size.

  Rajan: This DR only asks about overlap.

  Peter: Strike out "existing".

  Rajan: There are other potential issues here, e.g tail padding and relation to flexible array. But this is progress, so let us have this as PTC.

  Blaine: Consensus to have this "without existing" as PTC.

  leave the DR open

- DR 501

  Blaine: This impacted by the floating point FP-CR20. We need to split DR's and pull the C changes suggested in FP-CR20 into this one.

  ACTION: Rajan and FP committee to propose a new SC in coordination with FP-CR20.

  Left open.

7.3 TS 17961:2013+Cor 1:2016 Clarification Requests N 2150

no business

7.4 TS 18661 Clarification Requests N 2149

- DR9 no further discussion, moved to closed
- DR11 no further discussion, moved to closed
- DR12 no further discussion, moved to closed
- DR14 no further discussion, moved to closed
- DR15 no further discussion, moved to closed
- DR13, two papers submitted, N 2213 has the new TC. Rajan has some changes to it.

  Rajan presenting. There are only some last comments by Joseph with updated changes to make it compatible with TS part 2. Keep TS part 2 valid.

  Blaine will try to fold the changes it.

  left open
- DR16, N 2212

  Rajan: There has been a removal of the example from parts 2 and 3 and other minor editorial changes.

  moved to review
- DR17, N 2203

  Rajan presents. The problem here is that there might not be a common format for two such floating point types. Therefore this is explicit UB such that implementations may provide comparisons if they want to, but they are not forced to provide such comparisons.

  moved to review
- DR18, N 2204

  Rajan presents.

  moved to review
- DR19, N 2210

  Rajan presents. This is against changes to C11 that are currently in TS part 1.

  moved to review

- CR20, N 2211

  This are changes to TS part 1 **and** to C11.

  Blaine: There are more mentions of `DECIMAL_DIG` in C17 (and C11). Please clarify with paragraphs numbers, give a complete list of changes. (**ACTION** Rajan)

  Pull C parts into a resolution of DR501 **or** close DR501 in reference to that one.

  Rajan: Is the reference to typed `DECIMAL_DIG` ok? yes

  Jens: Please verify that C11 and C17 are no different w.r.t all of that.

  remaining open

- CR21 N 2224

  Fred presents.

  An updated version of N 2224 was sent during the meeting.

  left open

8 Other Business

8.1 Reentrant APIs (N 2241)

Keld presents.
David: Did the text come from another document
Keld: A previous document WG22.
David: Are all the interfaces in the other document thread safe?
Keld: He wants a new TS that also includes the functions from Florian's papers.
David: What is `setmedia`?
Keld: An extension to possibly chose audio output. Could be removed the proposal.
Florian: A lot of these functions are not in glibc. Also a lot of these functions are not even thread safe.
Keld: Some of them are, maybe 50%.
Martin: Floating an idea, and the question is if we support this as an idea?
Keld: Yes. I am aware that a lot is not implemented.
Jens: Concentrate on the re-entrant properties of the functions.
Philipp: Most freestanding implementations would not do that.

David: Suggest to have a discussion on the reflector to know which of the functions should be retained.
Rajan: He would prefer to get more of a C focus. E.g `setmedia` should be removed. It would be good to have a first filter for what is useful in our context. He would like to have a TS.
David: This came out of the work for CPLEX.
Florian: It is not clear how a caller would use `newencoding`
Jens: Be carefull to chose a name prefix such that this doesn't interact with the user identifiers.
ACTION: Keld to update document N 2241 in the lines as suggested during discussion.

8.2 Creation of a C Memory Object Model Study Group

proposed charter:

> This group will study the memory object model issues of C, including pointer provenance, uninitialized values, and related questions, aiming to reconcile mainstream usage, implementation practice, the ISO C standard text, and harmonization with ISO C++. It will produce papers and proposals for C2x.

David: Is there any objection to create such a SG?
There is none.
Peter: proposes to create a mailing list and monthly teleconference meetings. Also perhaps some physical meeting.

9 Resolutions and Decisions reached

9.1 Review of Decisions Reached

None

9.2 Review of Action Items

1. Carry over items

   ACTION: Blaine to reconcile N 2019 and N 2026 for DR469. Superseeded by new action item, see below.

ACTION: Convener to coordinate with WG21 on the mechanics adding a 'C' or 'P' designated papers for proposals to WG14. open

ACTION: Larry to write a paper with any objections to N 2161 (left shift operator), on hold, see new action item for Aaron, below.

ACTION: Convener to write a response for N 2174 as an N document, open.

ACTION: Clark to supply new words for DR499 based on reflector message 14632. Maybe obsolete by the proposed resolution to DR499, closed.

ACTION: Convener to send the CPLEX document to the national bodies that agreed to participate on the work item, open.

2. New items

ACTION: Barry to make the document on specific differences between C++ and C available to us.

ACTION: Jens to extend work on DR 486 and DR 495 for a more complete document that addresses inconsistencies concerning atomics (language and library).

ACTION: Blaine, in relation to N 2019, DRs 469, 479, and 493, to write a paper that identifies more apparent problems for threads (language and library).

ACTION: Martin to write up a proposal that forbids new types to appear in calls to `offsetof`.

ACTION: Rajan to clarify the resolution of FP-CR20 with paragraph numbers for the occurrence of `DECIMAL_DIG` in C17 and give a complete list of changes in relation to DR501.

ACTION: Rajan to in view of DR 501, the FP committee should propose a new suggested corrigendum in coordination with the resolution of CR20

ACTION: Aaron, for document N 2197, to look into the possible intersection with Martin's proposal for `assert` (N 2207).

ACTION: Convener to add the proposed changes of N 2197 to SD3.

ACTION: Aaron to revise N 2198 (`u8` character prefix) to reflect the discussion during the Brno meeting.

ACTION: Martin to write new versions of papers N 2190, N 2191, N 2192, N 2193 that take their interrelationship into account.

ACTION: Convener to update SD3 to the updated documents N 2214, N 2215, N 2216 and N 2217.

ACTION: Martin to for N 2209, integrate the issues that were raised during discussion.

ACTION: Florian to work on updates for N 2225, N 2227 and N 2228.

ACTION: Convener to add N 2225 to the bullet 12 of SD3.

ACTION: Florian, in the context of N 2226, to propose a paper that features a solution in the line of `uselocale` from POSIX.

ACTION: Keld to update document N 2241 in the lines as suggested during discussion.

ACTION: Aaron to coordinate with WG21 for left shift operator.

10 Thanks to Host

11 Adjournment

adjourned, Apr 25, 2018, 15:30

Appendix A: Straw polls

1. Do we want that defining a new type inside a call to the `offsetof` macro is a constraint violation?

   9 approved, 0 reject, 4 abstain

2. Shall we add the proposed changes of N 2197 to SD3?

   17 approved, 0 reject, 1 abstain

3. Is the committee interested in the resolution proposed in N 2208?

   2 approved, 10 reject, 5 abstain

4. Is WG14 comfortable in removing *ones complement* and *sign and magnitude* from the C standard.

   14 approved, 0 reject, 3 abstain

5. Do we want to change the types of the expansions of the macros `false` and `true` to be `_Bool`?

   12 approved, 1 reject, 5 abstain

6. Do we prefer a core language solution to introduce generic function pointers?

   14 approved, 0 reject, 5 abstain

Appendix B: Draft Minutes of PL22.11 meeting 2018/04/24

- Minutes by: Fred Tydeman
- 16:00 Start meeting. Garst/Ballman (Mover, Seconder)
- Attending at this session:

Name	Organization	Princ/Alt	Comments
Lars Bjonnes	Cisco	Principal	
Aaron Ballman	GrammaTech	Principal	
Rajan Bhakta	IBM	Principal	PL22.11 Chair
John Parks	Intel	Principal	
David Keaton	Keaton Consulting	Principal	
Clive Pygott	LDRA	Principal	
Tom Plum	Plum Hall, Inc.	Principal	
Freek Wiedijl	Plum Hall, Inc.	Alternate	
Martin Sebor	Red Hat	Principal	
Florian Weimer	Red Hat	Alternate	
Daniel Plakosh	SEI/CERT/CMU	Principal	
Blaine Garst	ThePlanet Earth Society	Principal	
Fred Tydeman	Tydeman Consulting	Principal	PL22.11 Vice Chair Secretary

- Attending at other times during week

Name	Organization	Princ/Alt	Comments
Barry Hedquist	Perennial	Principal	PL22.11 IR

- Other people attending

Name	Organization	Princ/Alt	Comments
Larry Jones	Siemens PLM Software		
Robert Seacord	NCC Group		

1. Approval of Agenda Items added:

   8.1 Two items are out for comments

   8.2 Should PL22.11 meeting be whole week?

   Items deleted: None

   The Agenda was approved by unanimous consent Garst/Parks

2. Approval previous minutes: No objections. Keaton/Garst

3. INCITS Antitrust Guidelines and Patent Policy Reviewed the Antitrust Guidelines and Patent Policy

4. INCITS official designated member/alternate information If problem, see chair. Be sure to let Lynn Barra know of any changes.

5. Identification of PL22.11 Voting Members: 12

   (a) PL22.11 Members Attaining Voting Rights at this Meeting None

   (b) Prospective PL22.11 Members Attending their First Meeting None

6. Members in Jeopardy

   (a) Members in jeopardy due to failure to return Letter Ballots None

   (b) Members in jeopardy due to failure to attend Meetings

      i. Members in jeopardy for failure to attend this meeting. None

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

      iii. Members who lost voting rights for failure to attend this meeting None

   (c) Members who previously lost voting rights who are attending this meeting None

7. Procedures for Forming a US Position Per normal.

8. New business

   (a) There are two items out for comments now.

      i. FDIS: Yes with comments; (comments should be at INCITS)

      ii. CFP 18661-2: Approve/reaffirm

   (b) Should meeting be whole week?

      C++ has been whole week, C has been 30 minutes on Tuesday afternoon Whole meeting (Mon morning & Fri afternoon) vs Tuesday afternoon

      No longer need to kick non-USA people out of the meeting.

      Bjonnes: Tuesday afternoon is good for new members

      Tydeman: Attendance issue: Miss the 30 minute session, but attend most of rest of week.

Pygott: Slight difference in minutes WG14 vs PL22.11
Straw poll: 8 for whole week, 0 against, 2 abstain
Bhakta: PL22.11 Chair will work with WG14 convenor to adjust agenda.

9. Next meeting: Pittsburgh, USA, 2018 Oct 15-18

10. Adjournment

    Meeting adjourned by unanimous consent (Garst/Ballman) at 16:16