InterNational Committee for Information Technology Standards (INCITS)
Secretariat: Information Technology Industry Council (ITI)
1101 K Street NW, Suite 610, Washington, DC 20005
www.INCITS.org

WG14 2187
INCITS PL22.11-2017-00004
Date: 2017-10-31
Reply To: Rajan Bhakta
PL22.11
Email: rbhakta@us.ibm.com

Draft Minutes for 30 October–3 November, 2017
MEETING OF ISO/IEC JTC 1/SC 22/WG 14 AND INCITS PL22.11
WG 14/N 2187

Dates and Times

30 Oct 2017	09:00 - 12:00 - Lunch -13:30 - 16:30
31 Oct 2017	09:00 - 12:00 - Lunch -13:30 - 16:30
1 Nov 2017	09:00 - 12:00 - Lunch -13:30 - 16:30
2 Nov 2017	09:00 - 12:00 - Lunch -13:30 - 16:30
3 Nov 2017	09:00 - 12:00

Meeting Location

Albuquerque Marriott
2101 Louisiana Boulevard NE
Albuquerque, New Mexico 87110 USA
http://www.marriott.com/hotels/travel/abqnm-albuquerque-marriott/

Meeting information

Venue information: N 2155

Local contact information

David Keaton <dmk@dmk.com>

1. Opening Activities

1.1 Opening Comments (Keaton)

David welcomed us to ABQ, 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	
John Parks	Intel	USA	PL22.11 Chair
Clark Nelson	Intel	USA	
Fred Tydeman	Tydeman Consulting	USA	PL22.11 Vice Chair
Barry Hedquist	Perennial	USA	PL22.11 IR
Tom Plum	Plum Hall	USA	dialed in
Martin Sebor	Red Hat	USA	
Larry Jones	Siemens PLM Software	USA	WG 14 Project Editor
Aaron Ballman	GrammaTech	USA	
Clive Pygott	LDRA	UK	
Michael Wong	Codeplay / ISOCPP	CA	
Jens Gustedt	INRIA	France	
Tao Schardl	MIT	USA	
Tom Skogland	LLNL	USA	

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 and work on Defect Reports.

Participants need to register on ISO's site for this meeting.

David Keaton is the meeting Chair.
Rajan Bhakta is the Recording Secretary.

1.4 Approval of Previous Minutes [N 2142]

The previous minutes have been amended for typos, etc.
The previous minutes are approved by unanimous consent.

The final approved Markham minutes are N2185
The draft Albuquerque minutes are N2187

1.5 Review of Action Items and Resolutions

ACTION: Blaine to reconcile N2019 and N2026 for DR 469.
Open – More work to do.

ACTION: Convener to have venue information for Albuquerque in post meeting mailing.
Done

ACTION: Convener to add N2098, along the lines of Option 3, to SD 3.
Done – N2182

ACTION: Convener to add N2101 to SD 3.
Dome – N2182

ACTION: Convener to add TS 18661, Part 3 to the SD 3.
Done – N2182

ACTION: Convener to add TS 18661, Part 4 to the SD 3.
Done – N2182

ACTION: Convener to add TS 18661, Part 5, evaluation format pragmas, to the SD 3.
Done – N2182

ACTION: Convener to add TS 18661, Part 5, optimization control pragmas, to the SD 3.
Done – N2182

ACTION: Convener to add TS 18661, Part 5, reproducible results, to the SD 3.
Done – N2182

ACTION: Convener to add something along the lines of TS 18661, Part 5, alternate exception handling, to the SD 3.
Done – N2182

ACTION: Convener to add TS 18661, Part 5, rounding direction macro, to the SD3
Done – N2182

ACTION: DR 467. Fred to write up a model and definition for normalized double double.
Done – N2151

ACTION: DR 496. Clark to write a draft of a Proposed Technical Corrigenda.
Open

ACTION: DR 497. Fred to write a Suggested TC.
Done - SC22WG14.14657 (this is not a STC).

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

ACTION: Blaine to draft a PCR for DR 502
Done

ACTION: Larry & Jens to examine the use of a server to maintain the Standard.
Done

ACTION: Clark to take code examples as an N document back to CPLEX, and come back with recommendations on how to proceed.
Done

ACTION: Jens to publish an improved LaTeX version NLT the post meeting mailing.
Done

1.6 Approval of Agenda [N 2184]
Added: 8.1: Discussion of replacement for DR process.
Deleted: None
Modified/Corrections: Correcting link to SD3 by replacing the extension pdf with htm.
Approved

1.7 Identify National Bodies Sending Experts
Canada
US
France
UK

2. Reports on Liaison Activities

2.1 SC 22
David Keaton now SC22 Chair

2.1.1 SC 22 Officer appointments

- New WG 5 (Fortran) convener - Steve Lionel
- New SC 22 chair - David Keaton

2.1.2 JTC 1 Plenary results affecting SC 22

2.1.2.1 JTC 1 Document access rules

- We had to protect every draft of the standard. But now we can open up everything up to and including the CD.
    - After that (DIS, FDIS), it needs to be protected. For C2X it is open until this point.
    - For C17 it is different (minor revision) which starts at FDIS (single vote).
    - There can be any number of drafts that are not voted on which can be open. When it goes to vote we need to close it off.
    - We can do a draft after each meeting in WG14.

2.1.2.2 JTC 1 Editors' Forum

- SC22 and others having problems with ISO rules for editing. Editor's forum is fighting for us.
- Rules about documents are being looked at.

- Goal is to move everything over to XML (although in a proprietary format).

2.1.2.3 Change to relieve pressure on Technical Corrigenda [N 2183]

- David has not talked with Herb about this, but he will be talking with all WG Conveners.
- Average standard is 50 pages at ISO.
- Only 3 TC's per standard, and none past 2 years.
- ISO allowed write access to a portion of their website to publish things.
    - We can update it as often as we want.
    - We can communicate directly with customers of the standard.
- ISO/IEC consider a defect to be something so urgent that it must be turned around within 1.5 months even if there is no meeting. This is finally now public (the definition). Given that, we can change our work flow to not have to judge whether a paper is a defect report or not.
    - We've only had one like that: C11 date macro.
    - Our "defects" were using the English definition. Called “Interpretation Requests” in ANSI. We moved those to defect reports when moved to ISO.
- These old "defect reports" can be published on the website as Future changes to the standard or Interpretation Requests or Clarifications or whatever.
- N2183 talks about ways to deal with it. Clarifications could be interpretations of the published standards.
- No change to C17. We can decide what to do with the defect log (7.1).
- We may be able to update the ISO website as soon as the post meeting mailing.

2.1.4 C17 minor revision approved - FDIS due on December 30

- Straight to FDIS by December 31, 2017 at the latest.
- Any technical comment may cause the ballot to fail and we don't get another one.

2.2 PL22.11/WG 14

Item 23 in SD3 was the first new one added.

2.2.1 Convener's Report and Business Plan [N 2162]

2.2.2 WG 14 Standing Document 3 [N 2182]

2.3 PL22.16/WG 21

The Ballot for ISO/IEC DIS 14882:2017 has been approved. The document is in final drafting with ITTF and WG21. Work on C++20 has begun. WG21 and PL22.16 will meet in Albuquerque next week.

2.4 PL22
Chris Tandy new PL22 Chair.

2.5 WG 23

- The document in mailing is for C (N2169). It is a separate document from the main language independent document.
- The main body (language independent vulnerabilities) is moving towards a new publication (~18 months).
- Meeting at this location next week.

2.6 MISRA C

- Moving towards a new release in about 18 months.
- Trying to focus on security more.
- Moving to base it off of C11.
- Not much interaction between the Safety/Security study group (Robert Seacord) and this one.

2.7 Other Liaison Activities

3. Reports from Study Groups

3.1 C Floating Point activity report

- Monthly teleconferences ongoing
- Working on:
    - DR's for published TS's
    - Proposals for C2X
    - New IEEE 754 revision discussion

3.2 CPLEX activity report

- Not very active due to some absences
- Made some changes to address the worst of the OpenMP concerns

3.3 C Safety and Security Rules Study Group

- Looking at MIRSA material, making progress w/o MISRA participation.

4. Teleconference Meeting Reports

4.1 Report on any teleconference meetings held

- Editorial committee, scheduled June 7, was resolved by email.

5. Future Meetings

5.1 Future Meeting Schedule

- Spring, 2018 – Brno, CZ, 23-26 April (N2181)
- Fall, 2018 – TBD, preferably US – Pittsburgh?
- Spring, 2019 – TBD, typically non-US – open to Proposals, Denmark?

5.2 Future Mailings

- Post Albuquerque – 04-December-2017

6. Document Review

1. Proposed C17 FDIS [N 2176], diff marked [N 2177]

Summary:

Josephs comments have been incorporated in the newer doc (not the one we have). We should vote this out for review and submittal to SC22 for FDIS Ballot. We'll have an ongoing review of the document and revisit that effort daily. On Thurs we'll vote the document out for FDIS. DRs that are CLOSED as of Markham (last meeting).

Thursday – C17 draft seems to be clean at this point.
There are some edits that Larry and Jens will make

SP: In favor of printable spaces?
Yes - 2
No - 12
Abstain – 1

Editorial Review Committee – to review any edits between the end of this meeting and FDIS.
Aaron Ballman
Clive Pygott
Blaine Garst
David Keaton
Joseph Myers

SP: In favor of sending draft C17 to editorial committee thence to FDIS?
Yes - 14
No - 0
Abstain – 0

Dec 6, 2017, 8 am PST, meeting of editorial committee. 2 hours

Details:

- Joseph's messages have been addressed in a newer version of the draft in the git repository.
- Minor changes like typo's and missing commas were also corrected (TC or otherwise).
- Foreword #7: Principal changes include technical corrections and clarifications is sufficient. No change needed.
- We need review of the changes outside of this meeting as well. Joseph has done it, but others should as well.
- There were separate commits for each DR.
- *AI* Jens to email the details of the web server to David Keaton to bring up in the meeting along with the updated document (addressing Joseph's comments).
- Goal for Thursday is to have a list of any edits required and a decision to send the result out for FDIS.
- An editorial review committee will look over any changes that come out of this meeting and meet via teleconference before sending it to FDIS.
- Other than the forward and editorial changes, all changes should have a DR associated with it.
- Each person should do that rigorously and bring it forward to the committee if there are issues.
- No new DR additions this meeting as the DR cutoff has already happened last meeting. About 60 DR's (See N2148 Closed DR's for a list of possible changes).
- Each person should focus on what they are interested in first, then last names that start with A-M do top down (DR 400 downwards) while others do bottom up (DR 503 upwards towards 400).
- Aaron: A change between TRoff and LaTeX where bookmarks do not link functions to the part in the standard.
    - Larry: On the list of things to fix. Likely not by C17.
- Aaron: Diff markings are not always correct.
    - Jens: Technical problem which I don't know how to solve.
- Blaine:
    - DR413 changes: Seems OK.
    - DR416 changes: Says shall not be called.
        - Larry: Means the same thing.
    - *AI* Jens: 7.26.6.2#2: Missing the new second sentence.
    - *AI* Jens: DR473: Mislabeled function name (lgamma).
- DR485:
    - Jens: The minutes are what matters, and the changes were made to reflect that.
    - It would be good to deprecate it but it was not voted on.
    - Martin was going to supply words for the future directions (14645).
    - We never formally adopted words.
    - The future directions is a non-normative change and we expected it to be present in C17.
    - We will take what is present in 14645.

    - Martin: Is updating the annex something that needs exact words?
        - Larry: Not necessary but it helps.
    - May change "will be removed" into "may be removed".
- Jens: For C2X we should remove things we listed as being removed.
- *AI* David Keaton to update SD3 to go through the obsolescent features and future directions to decide what to remove for C2X.
- Clive:
    - DR448: Words are in a different order.
        - Valid editorial change.
    - DR460: Text was put in and then struck out except for the first word.
        - Larry: May be due to the tool that does the diff marks. Always compare to the non-diff marked version.
        - It was supposed to be removed so it is correct.
    - DR454: The Suggested TC was not present.
        - Expected since it was not proposed.
- Aaron: Everything posted on the reflector by me have been addressed by Jens and Larry.
- David: Are we good with putting this forward as the FDIS document?
    - Jens: There are a number of editorial things that should be fixed.
        - Larry: Not urgent, can wait.
- Aaron: The printable spaces are distracting.
    - Jens: I like them. For things like scanf, they are really important.
    - Barry: ITTF may say something about this as well.
    - Clark: Would it be hard to achieve having it in some places but not others? i.e. Where it matters.
    - Jens: It is a bit of work.
    - Rajan: Since it is limited to scanf, we can just put a note or comment on the parts where it is significant.
    - Tao: Can put both examples, one without the printable space, and where it matters another example saying "this shows the spaces".
    - Martin: I saw an example in token pasting as well.
    - Larry: These examples have been there forever including a mistake that had a missing space, and no one has complained.
    - Blaine: It was confusing at the start.
    - Jens: It is always good to think and be surprised.
    - Straw Poll: All in favor of printable spaces?
        - 2/12/1. We will not have printable spaces showing up in the document.
- Joseph, Blaine, Aaron have gone through the entire review.

- Straw Poll: All in favor of submitting this document for FDIS subject to any edits from the review committee?

    - 14/0/0. We are good with this document for FDIS after editorial review.

- Editorial review committee (to review any edits from the end of this meeting to FDIS): Aaron, Blaine, Clive, David, Jens, Larry, Joseph
- Will meet via teleconference on December 6th, 2017 at 8am PST, 2 hours.

2. Properties of Generic Functions [N 2145]

Summary

The term 'generic function' is not defined. This paper proposes a definition for clause 3, defining generic function, as well as modifications to clause 7.17.1 for Atomics.

Should compound literals be supported or excluded? They may not be addressed/supported across the library in general. This proposal is for a new feature? Compound literals are usable if parenthesized – (…).

Martin will write two papers addressing the issues discussed.

Details:

- Martin: No implementations support compound literals. It can't be done by type generic macros. Some compiler magic is needed.
- We should define the term "generic function".
- Normative change, requires all implementations change to support compound literals.
- Could also require parenthesizing to make it work with macros.
- Does WG14 want compound literals to be supported?
- Change 2 does not force it to be a macro.
- Clark: Whatever we do with atomics should be like tgmath.
- Martin: tgmath is a closed set of types, atomics is unbounded (user defined types like the Shorts type in the example).
- Clark: Prefer to see a general description and make tgmath a subset of that. Do not want them to be described differently.
- Martin: They are very different and implemented differently by everyone.
- Jens: Can be a closed set. load, store and exchange are the only ones that are unbounded, and they can be done with individual functions.
    - Martin: The problem with that is that rvalues cannot be passed in since they are not pointers.
- Clark: Since no implementation supports compound literals it is not major motivation here.
- Martin: The cost will be in the specification. Most implementations have some form of type generic intrinsics which could make it work for compound literals easily.
- Blaine: This change would take away macro implementations since with comma's it is not possible.

- Martin: Correct. For compound literals, this is true.
- Clark: It would require compiler changes since there cannot be macros anymore.
- Martin: Since this applies to the standard library already, existing functions have this issue. We don't say anything about that right now.
- We should change the implementations to support compound literals throughout the library.
- Clark: Is there a customer who is requesting it?
    - Martin: No. No bug report on GCC for this. Atomics are not used that much now, but are starting to be used. It will come up sooner or later. The only way to initialize an atomic object is full initialization, not piecemeal.
- Blaine: Compound literals being addressed throughout the library is the problem that needs to be solved in general instead of focusing on this particular part.
- Jens: This is new. Causes implementation changes for everyone. Need a general phrase for library with regards to possibly being a macro and hence need parenthesized compound literals.
- Rajan: Do the generic function definition and then deal with compound literals separately.
- Blaine: The atomic functions do not have to be functions (can be small snippets of assembly).
- Clark: There does not seem to be a case that takes a struct by value in the library.
- Martin: Can pass a member of a compound literal to a function that takes a scalar. The other case is taking the address of a compound literal.
- *AI*: Martin to split N2145 into two proposals. One to define Generic functions, another to work on compound literals.

3. double double and C [N 2151]

    - No proposal, no action. No change.
    - David: There are gaming GPU's that do float float too. It is not unique to IBM.
    - Martin: Worried about maintaining the floating-point model in the future if there is lack of expertise later on.
    - Martin: What is WG21's position on this?
        - Jens: And Fortrans?
        - No idea.
    - Clark: If this was to be worked on, it should be presented to other people working on double double before coming to this group.

      Straw Poll: Does the committee want to support a model for double double?
      Yes - 1

No - 10
Abstain – 5
Result: Fails

4. Flexible array members may take unspecified values [N 2159]

Summary:

The trailing padding discussed in this paper is not what the author thought it was. There is no padding. This paper is a proposed DR with a Suggested Technical Corrigendum. Blaine does not see issues resolved by the STC. Clark wants to spend more time looking at this paper to determine if there's anything there. Rajan does not see a real practical problem here.

Revisited on Wed morning, afternoon. The paper really needs additional study, but we are gaining closure on understanding the concerns expressed.

Explicit initialization of a flexible array member should be ill-formed, but there are implementations that allow it. The Standard does not say that such initialization is not allowed.

Straw Poll: Should the Standard make initialization of a flexible array member a constraint violation.
Yes - 11
No - 1
Abstain – 1

The other four points we have consensus on.

Details:

- Clark: "So the trailing part ... of a structure is considered to be padding".
  - The padding that the paragraph is talking about is the padding used to align the Flexible Array Member (FAM).
  - Larry: It is talking about the imaginary structure, not the actual structure.
  - Jens: It can be misunderstood.
  - Clark: It's been a number of years and hasn't been misunderstood.
  - Blaine: It seems that there is no padding since the array after dummy takes the bytes and hence there is no padding to take unspecified bytes.
  - Jens: Everything after the normal structure members are padding which may or may not be interpreted as part of the FAM.
- Jens: There are other concerns beyond padding. For example questions about effective type.
  - Clark: For Aa: There is no object so the question does not make sense.

    - Jens: If it was automatic, not malloc'd for example, the type is fixed and it will never change.
    - Clark: That is right. It is intended to be that way.
    - Jens: That should be clarified.
    - Blaine: Can be done through alignas.
    - Aaron: The standard does say you can't access it in the same paragraph.
    - Jens: If there is space you can do it.
- Aaron: The example in the introduction, 'array' is after padding between dummy and the end of the structure.
- Martin: You should be able to access array[0] etc. if the structure was allocated on the stack (automatic).
- Aaron: It can fall at the end of the structure, but does not have to.
    - Jens: Most implementations I have seen have it (the FAM) inside the structure.
- Clark: We agree we would like to see what would otherwise be padding not be padding if it overlaps with a FAM.
- Martin: Do you see any implementations do anything surprising?
- David: There are some implementations that don't have alignment requirements > 1.
- Clark: Is a default minimum size of a FAM available for programmers to take advantage of?
- Jens: With malloc, it does not necessarily have an object. With calloc, I would like to have access to it since it is initialized to zero.
    - Martin: Assumptions is that this works and there are lots of code that depend on it.
    - The alternative is to force array sizes > 0 to have space.
- Rajan: Stack allocation or malloc/calloc and accessing inside the base structure is not common usage so this is not a real problem in the field.
- Clark: The example in the standard uses sizeof + bytes needed. It should be offsetof(FAM) + size needed.
    - The standard is missing an example that shows this.
- This can be a proposal or an issue.
- Blaine: Doesn't seem to be enough motivation to do anything here. Not a practical issue.
- Clark: There might be potential for something that needs fixing, but need more time.

- Clark: I think the statement that 'the trailing part of a structure can be considered padding' is not true.
- Jens: The standard is not clear on when the padding goes away.
    - You can't know how big the flexible array is.
- Martin: When you allocate the object at the same size of the struct, there may be padding, but if you allocate a larger amount, then the padding goes away and you have a flexible array.
- Jens: If you know that you can fit at least one element in the padding, it can be used.

- Martin: As soon as you store in it, that part of the padding goes away.
- John: The purpose of the padding is to align the FAM.
- Blaine: There is two kinds of padding, one for alignment assuming no FAM, and the other is for the FAM.
- Martin: If you have long long (8 bytes), char, then a char FAM, you have 7 bytes of padding. Once you store in the first element of the FAM, you end up with 7 bytes of padding and the FAM is a one element array.
    - Jens: This is not how the effective type rule is defined.
- Jens: Automatic allocation has fixed type since the effective type is the declared type all the time for that object. The type of the bytes must be clearly defined and fixed and hence not ever padding.
- Clark: If the FAM was not present, it would be padding, but once it is present it is not padding.
    - Martin: The standard says there padding.
    - Clark: It says there *may* be padding. The reason it is there since the FAM may be larger than any other member of the structure and hence needs to be larger.
    - Martin: The FAM must be initialized for auto's if there is an initializer.
    - Clark: The statement is that in most situations the FAM is ignored. We don't have a crisp definition of when it is ignored or when it is not. What happens to the FAM? Is it zero initialized or not?
    - Blaine: If the padding overlaps the FAM, then are the overlapped bytes initialized to zero?
    - Clark: We hope the compiler would not overwrite the FAM.
    - Martin: The compiler can optimize an overwrite the padding/FAM if it knows there is no access to it.
- Blaine: There is no requirement to initialize the FAM.
- Clark: I think the committee intended FAM to only deal with dynamic allocation, not auto or assignment.
- Jens: The standard explicitly says you cannot initialize FAM. It also says assignment only copies the non-FAM members. It also says it is undefined whether it can be initialized.
- Martin: I believe since you don't copy the FAM on assignment they are padding.
- Clark: The statement about FAM acting as if it wasn't there, may mean that none of the elements are copied. We do want to allow it though.
- Larry: They are padding so the compiler can do it.
- Clark: Don't like it being sometimes there or sometimes not. Perhaps say it may copy them or not explicitly.
- Jens: I do want a guarantee that changing an element in the FAM does not clobber the remaining elements.
- Clark: Document concerning N2159
    - Is this a strictly-conforming program? Should it be?

    - Not sure if this was thought about when FAM was written up.

    - Martin: It illustrates the problem, but more if count was assigned to after the loop.
- Clark: Taking it for granted that assigning to any member of the struct will not overwrite the FAM.
    - Jens: Is this guaranteed by the standard though?
    - Clark: To me it is so obviously wrong we shouldn't bother with it.
    - Martin: You are focusing on auto, not dynamic. Not a problem if malloc'd.
    - Clark: Yes, that's true.
- Martin: If this doesn't work, it will encourage programmers to use a bounded array instead of using FAM's.
    - It happens in GCC a lot with an array bound of 1 and for automatics.
    - Jens: It is mainly for malloc, but in some cases it is used as an auto.
- Blaine: FAM is only useful in allocated storage.
- Jens: In my example, knowing that the first character is allocated statically, then it is valid, ex. memcpy.
- Clark: Does anyone think there is a problem with the computation of count. No. Does anyone think there is undefined behavior in the loop body? No.
- Martin: I don't think the standard says it is strictly conforming. Most of the time the member is ignored for example.
    - Clark: Do you mean the freedom for padding to change arbitrarily? I don't consider that interesting. Assuming that padding can not change, is there any other issue with this example?
    - Martin: No. There are other issues like copying to another struct X.
- Clark: Does anyone think the standard should guarantee one way or another if the data is copied via struct assignment.
    - Blaine: Strong opinion that it is unspecified.
    - Martin: It should say it explicitly.
    - Aaron: My opinion is it should be the sizeof struct copy.
- Clark: The standard is written such that you can do be element by element copy. If you want bit by bit you can do memcpy.
- Jens: I think we should not force implementations to do a copy of the FAM.
- Aaron: Doesn't that say that FAM is not a member of the structure.
- Fred: I would like to see words added to 6.7.9#19 for implicit initialization to explicitly exclude FAM.
- David: If a compiler did use the latitude to overwrite padding for auto it would likely do the same for allocated storage and hence would have been broken anyways.
- Aaron: The compiler is not required to initialize or assign FAM's. A user would find this confusing.

- Clark: For dynamic allocation, the user would not expect it to be copied, so we would have to let them know how to compute how many elements are copied or usable. There is a difference between memcpy since the programmer gives the amount of bytes. Struct assignment does not, it is up to the compiler.
    - Martin: The compiler can memcpy only the bytes that are element based and not the rest.
- Jens: Do we want to add initialization to FAM in C2X?
- Martin: Clark seems dismissive of wobbly bits. The compiler can decide to do element copy instead of byte copy for memset. You cannot ignore those concerns.
    - Clark: We should just focus on this. Not deal with other issues.
    - Martin: It is not a useful solution unless we do deal with the wobbly aspects as well.
- Clark: If we make it clear that bytes that are a part FAM are not padding, we will have solved a problem.
    - Martin: Then struct assignment and memcpy will have to copy the FAM.
    - Clark: Unless we say differently.
- Jens: There are implementations that do not copy all the FAM bytes I believe.
    - No one disagrees with the summarized points in the paper.
    - Assignment and initialization are not specified whether FAM's are assigned/initialized or not.
- Clark: Need time to look at the suggested TC.
- Clark: Explicit initialization of a FAM should be a constraint violation (preference).
    - Martin: GCC allows it at file scope but not at local scope.
- Clark: We should have a straw poll to see if the standard should say that FAM initialization is not allowed.
- Straw poll: The standard should make an attempt to initialize a FAM a constraint violation.
    - Vote: 11/1/1. Passes.
- Clark put another paper up to address the suggested changes by Jens in N2159.
- Clark: Toying with the idea of making recommended practice that the FAM should be at the end of the struct.
- Fred: For change 2 in 6.7.2.1#18: It should be FAM is ignored unless otherwise specified.
- Jens: Completely fine with agreement of the summary and just looking at words now.
- Martin: What's default initialization for a FAM?
    - Tom S: There is nothing to initialize.
    - Clark: Change 5 of 6.7.2.1#18 shows that the FAM does exist.
- Blaine: I am underwhelmed with the initialization. The standard says it is invalid.

    - Jens: I expect it is default initialized. I have code that initializes it with a compound literal. I also use statically allocated objects as a constant of that type.
- Clark: Want assignment and initialization to be consistent. We have consensus that assignment should not force the FAM assignment and I want initialization to do the same.
    - Jens: I want it to work for static.
    - Martin: Why don't we make FAM initialized for statics and autos as well?
    - Aaron: Users would expect the FAM being initialized.
    - Clark: Nothing we do will make FAM "normal".
        - If the struct is dynamically allocated there is no way it can be copied and hence it is still weird.
    - Aaron: I wouldn't expect anything outside the struct to be copied over, but would expect the bits inside to be copied.
    - Tom S: If you are given a type, you can create a union of that type and one with the size of the array you need inside a similar structure and the problem is solved.
- Clark: It's already been mentioned that struct assignment has been implemented by member copy. Now they will need to copy the FAM up to the end of the structure as well.
    - Aaron: Or disallow placing objects there entirely.
    - Jens: The ABI fixes it.
- David: There is agreement to have a future version of this document. There is disagreement over whether assignment and DEFAULT (explicit initialization was polled to want to be a constraint violation) initialization cover FAM. Status quo is that it does not copy. This matches the straw poll as well.
- Clark: We also had a poll that said copy may or may not happen.

5. Comma omission and comma deletion rev 4 [N 2160]

Summary:

> Any comments on this document? WG21, EWG, accepted this document. Next will be for WG 21, Core, to review. We, WG14, can process the Core wording once that has been done. We can track this for now, and take a harder look at it when we have WG21 Core wording.

Details:

- Martin: If C++ adopts this we should do the same.
- Clark: Which one goes first? C++ may wait for C to adopt it first.
- Michael: Was accepted by EWG in C++. Looking to adopt it in tandem with C.

    - Thomas is attending the C++ meetings.
- *AI* C++ liaison members to indicate that WG14 is good with N2160 (Comma omission and deletion).
- If we have any issues, we need to bring it forward to C++ as well.
- We need the author to let us know he wants to go forward with this before we can do anything other than comment.
- Barry, Fred: For us, the wording will need to be changed. Ex. ill-formed.

6. Harmonizing left-shift behavior with C++ [N 2161]
   Note related document [Standing Document 3 item #2]

Abstract: C++14 changed the behavior of the left-shift operator so that a common case of undefined behavior is instead well-defined behavior. Specifically, shifting a 1 into the sign bit of a signed operand to << is only undefined behavior on architectures other than two's complement.

Prior art: C++. GCC, Clang, and MSVC do not appear to treat this as an optimization opportunity (taking advantage of the undefined behavior), at least when targeting architectures that are two's complement and compiling for C.

This item is already in the WG14 SD 3. Do we want to add this to C2X? Most likely. Discuss at next meeting.

ACTION: Larry Jones to prepare a paper on why WG 14 should not adopt N2161.

Details:

- Martin: Does C++ allow padding bits?
- Aaron: Not sure. Maybe.
- Martin: C++ doesn't normally talk about padding bits.
- Aaron: The 'representable' part handles padding bits.
- Aaron: Hugh had concerns that this does not make for good math, but it is common use.
- Larry: There are twos-complement machines where shifting into the sign bit will cause an exception. They have non-exception based instructions as well though.
- Same words as the SD3 item 2.
- Will put it in C2X.
- Clark: Corresponding unsigned type has more context in C++ than C. It is not defined as a term there.
    - Jens: It is in the C standard as well (6.2.5#6).
- Rajan: Process wise: Since this proposal was brought forward this meeting why does it have special priority over other proposals that were brought forward in other meetings?
    - We will consider this in next meeting as part of C2X proposals and SD3 review.

- Larry: Object to having this having special status in SD3.
- David: Needs to be either a paper or in SD3 for consideration for C2X.
- *AI* Larry will write a paper with any objections to N2161 (left shift behavior).
- Note related document [Standing Document 3 item #2]

7. Attributes in C [N 2165]

Summary:

> Proposal to use the same syntax as C++, [[…]]. Rajan pointed out that C uses macros much more that C++, and prefers _Attribute (or another keyword type approach) over C++ syntax to better fit C programmers. Aaron said the complexity of shared headers between C and C++ makes [[…]] a better match. This paper also includes the use of double colons to delineate-vendor specific attributes.

Details:

- Aaron: It has been added to clang trunk under a special flag and there is an online compiler that supports it.
  - No user feedback yet.
- Rajan: _Attribute has only non-C++ as the reasoning.
  - Already diverged with _Noreturn, why not here?
  - C is already more macro based than C++ (templates cover that there). More C users are probably used to macros.
- Aaron: _Noreturn was due to semantic effects since we don't want attributes that have semantic consequences.
  - Due to the C++ shared headers and the charter saying keep it the same if possible.
- Blaine: It seemed the semantic argument was the reason for C11.
  - There are cross language concepts coming in. What happens if there is an attribute in one language but not in another? We are not covering all the things that apply with cross language issues.
- Aaron: There is a C++ standing document for feature testing. There is __has_cpp_attribute which gives the version of the standard it was accepted in, and vendors have 1 or 0 to see if it is there. Intent is for Clang to do the same for C. Don't need to change the syntax itself to cover this since it can be handled with the preprocessor.
- David: For C11, attributes were new in C++ and we had said the idea was not ready for C11. In Pittsburg we decided since it was more mature we can consider it for C.
- Michael: C did decide with keywords before on purpose.

- Blaine: We should continue with semantic impact = keyword otherwise use the normal attribute syntax.
- Aaron: It is not necessarily that it has no semantic impact. It has to be that if the implantation ignores it, is the program still correct? i.e. The implementation should be free to ignore all attributes and have the same results.
- Rajan: Other implementations use :: as well and this may break them. GCC and Clang did have to change for C++ support.
- Jens: Allowing implementations to add attributes is very important.
- Aaron: There are over 200 vendor specific attributes in Clang and expected to grow.
    - GCC is also allowing all their attributes to use the [[]] syntax.
    - GCC has two different Front Ends, one for C and one for C++.
- Rajan: The C++ compat makes sense, but we need to cater to our users (C programmers) and not have that being trumped by C++ compat when a lot of headers already deal with it with #ifdef C++ in some form.
- Aaron: Haven't seen this common code with parts of attributes in macros. They cover the entire attribute by a macro.
- Blaine: We can do both ways. There are many compilers and having one implementation is not good enough.
- Concerns about C++ compat, C programming style, implementation.
- Michael: Is the pollution of multiple forms worth the cost?
- Clark: There are a lot of syntaxes for attributes, one of which is standardized by C++. We should agree to adopt one of the existing syntaxes or not adopt attributes at all. Don't want to try and create anything new.
- Aaron: The straw poll indicated support for the square bracket syntax.
- David: Everyone needs attributes to implement their OS, and without standardization they have to make their own choice.
- Blaine: They will have their own way. There is an implementation cost to doing this.
- Aaron: Haven't heard of anyone with implementation issues with [[]]. Colon's are a separate issue.
- Blaine: Just because it is done in many places doesn't mean it is easy to do.
- Martin: Not sure having __attribute or _declspec would be preferable since it would have an implementation cost and burden the one that did not have it. Similar to the users. Prefers _Attribute but the charter should be overriding concern and use the C++ method even though I don't like it.

- Jens: The grammar in here doesn't work with __attribute or __declspec. Can't use either of them.
- Aaron: Haven't heard anything that people are looking for updates to the paper.
- Rajan: For the example, can you have 'struct S [[deprecated]];' (not in the grammar, but why was it chosen this way is)? It would fit the rule of thumb better if it was after the tag.
    - Clark: It is that way in the C++ standard that's why.
    - Michael: If it is after the named entity (in this case the struct) it applies just to the name tag, not the struct. There was no consistency between the existing attribute systems.
    - Aaron: Sometimes the compiler moves it for you too which is confusing to users.
- *AI* David to update SD3 to point to n2165 for attributes.

8. Syntax proposal for attributes in C2x [N 2175]

Summary:

> This paper proposes a 'middle ground' for the attribute syntax used for C and C++. There is some resistance to this proposal due to coding practices, changes to existing practice, difficulties that some editors will have, etc.

Details:

- Main issue is namespace invasion.
- Clark: Don't like new invention.
- Aaron: Only two kinds of attributes being added. Standard and vendor. We control the Standard. Vendor is a matter of quality of implementation.
- Jens: The new syntax is not used very often for C right now. So can't use that a basis.
    - For vendors, it is still a problem since they don't know what users have. GCC has __ prefix in the attributes.
- Aaron: Vendors can do __ for quality of implementation. C++ has not seen this issue either.
- Jens: There was a problem with the u and U strings when that happened since macros had been defined using those as identifiers.
    - This actually hit one of Jen's students where the name was not protected by __.
- Rajan: vi or similar editors will have this throw them out of the flow.
    - Not good for EBCDIC or other things including keyboards.
- Jens: The main usage is in headers, not user code so this will not be a problem.

    - Rajan: Disagree. For example, fallthrough would be mostly in user code.
- John: Most keyboards don't have this character. It would change the character of C to depend on keyboards.
- Blaine: I like bringing in Unicode characters like greater than or equals, less than or equals, etc.
    - We can perhaps have them use $ or something else to join.
    - Also the parser can join the two parts if it sees the ::.
- Aaron: Aside from being incompatible from C++, there would be problems for having something that looks like :: but being a single code point and having users typing two colons by mistake.
- Rajan: Perhaps have a footnote or recommended practice saying to not use user namespace names be sufficient?
    - Jens: No, since it is already in the field without doing the __ or anything else.
- Martin: What does Clang do for things like attribute malloc? Do you allow both __ and not?
    - Aaron: Not for the vendor name, but yes for the attribute.
- Jens: We should look into getting more unicode into the language. The description of the syntax would be easier if we could allow the <= and others as a single character.

9. Evaluation Formats [N 2166]

Summary:

A proposal to define what 'may' means in the following:
Except for assignment and cast (which remove all extra range and precision), the values yielded by operators with floating operands and values subject to the usual arithmetic conversions and of floating constants are evaluated to a format whose range and precision **may** be greater than required by the type.

What is may? When is it allowed, when is it not allowed? The four major concerns expressed in this paper were reviewed by CFP. N2186 is the CFP response to N2166.
Problem summary in N2166:

1. Reworded for clarification.
2. Reworded for clarification
3. Reworded for clarification
4. Another paper coming, but also addressed in #2 above.

ACTION: Convener to add N2186 to SD 3.

Details:

- David: The change that most resonated with me was the change to 5.2.4.2.2#9 and the change in N2186 covers that.

- Clark: Format is not defined in the standard and it overlaps with representation in at least one place. Representation is also used in a number of different senses. Format seems to refer to how a bit pattern can refer to a value. Representation can be that but also applying those rules to a specific bit pattern. Wondering how IEEE uses it and if we can use data format to talk about the more abstract view of the representation we use. Ex. Same representation and alignment would be having the same data format and alignment requirements.
- IEEE does use representation and format and the CFP group tries to do it the same way.
    - We should keep it the same way for the floating-point parts of C to keep it harmonized with IEEE.
- Rajan: We do define evaluation format here in the proposed change.
- Clark: Will take a stab at writing a paper for format/representation and pass it to Rajan and Larry to look at it.
- David: Any disagreement to adding N2186 to SD3? None.
- *AI* David: Add N2186 to SD3.

10. return and extra precision [N 2172]

Summary:

N 2172 proposes to require that return() actually return the type of the function. TC 3 to C99 added to a footnote that the return may have a wider precision than implied by the type: a cast may be used to remove this extra range and precision. N 2172 proposes to remove that clause.

In general, we do not agree with this change.
SP: Do we want to adopt N 2172?
Yes – 2
No – 11
Abstain – 0

N2172 is not adopted.

Details:

- The ABI can stay the same. They just have to go through the knothole.
- Blaine: Seems reasonable other than inline functions.
- Fred: That means if the user uses inline or not can give different results on computations.
- David: Often people are encouraged to use static inline vs macros. Losing the extra range and precision would hurt them.

- Barry: What do compilers do right now in standard conformance mode?
    - Fred: They can do whatever they want.
- David: At least some processors (Ex. 68000 and x87) like wide returns to use registers and do it. They would object to having it forced to go to memory.
- Aaron: If you don't know if the wide representation is used or not is troubling.
- Rajan: If you want to force the type you can always cast with forces stripping the extra range and precision.
- Blaine: The question is is the extra range visibly useful.
- David: Since not everyone can cast without going to memory and back, it is a performance hit for them. It is really expensive to go to memory and back.
- Fred: Integers can do this too.
- David: They generally do, but they also generally have cast instructions (Larry: Assuming AND works).
- Blaine: It's weird for _Bool since if you clear just a byte instead of a word it may have 1 bits elsewhere.
- Barry: How did it get added to C99?
- Fred: By my paper N1017 as stated in this paper.
- Clark: This is an area where there has been trash and whether we are continuing to thrash.
- Blaine: The committee has non-constant membership so can have non-constant decisions.
- Clark: Why was this brought up now?
- Fred: To address Willem's issue.
- Clark: The other paper (N1396) was explicitly for Annex F not the main body of the standard for IEEE conformance.
- David: Willem is a fixed-point expert who is only recently looking at floating point. That's why this is coming up now.
- Martin: Is this specific to the return statement. Does it apply to other things too?
- Fred: No, because elsewhere FLT_EVAL_METHOD defines how it works elsewhere.
- Larry: This is an ABI issue.
- Clark: This is not an ABI issue, since it is about the representation, not the value.
- Fred: It is an ABI issue since the result is given via registers in some cases.
- Martin: What happens if you are returning a struct with more float members than registers. This means some will have the extra range and precision (the members returned in registers) and not in others (the ones that spilled into memory).
- Blaine: We should use the register keyword!

- Fred: If the user wants to strip the extra range they need a cast after each function call vs putting the cast in the function before the return.
- Rajan: There is cost to adopting this proposal, but minimal if any benefit, we shouldn't do anything. If it matters to the programmer, they can always use a cast to remove the extra range and precision.
- David: Willem is complaining about the lack of a cross reference not that he wants to remove the extra range and precision. The CFP paper addresses that already so we don't need to scrape off the bits, especially for embedded systems which may not have cache.
- Straw poll: Do we want to adopt N2172?
    - 2/11/0. We will not adopt the paper.

11. Feature test macros and namespace for optional features [N 2174]

Summary:

Paper from Joseph Myers asking questions regarding inconsistencies within the Standard for feature test macros. We seem to believe that answer (b) in that paper is the correct answer for Annex K.

ACTION: Convener to respond to N2174 from Joseph Myers.

Details:

- David: Similar to __has_include, but from a different angle.
- Martin: Agree with the lack of consistency and clarity.
    - For Annex K the function names are always reserved regardless of the macro, though they may not be present if the macro is not defined.
    - For the floating point TS, the names are not reserved.
- This is a request for clarification.
- David: What do we want the answers to be?
- Jens: The functions may be shipped in the same shared library.
- Barry: Microsoft has the same names as in the standard mostly.
- Martin: They don't track the standard versions.
    - I don't know of any integrated implementation.
- Barry: Oracle has it implemented as well.
- Clark: The reservation rules allow a standard library function to call any other standard library function. The reason for the rule is to allow any of the Annex K functions to use any of the other Annex K functions.
    - i.e. If you use any annex K function then you can't have your own same named _s function.
- Martin: For proposals to add Annex K to glibc, there is discussion whether or not to include it directly into glibc or a separate library.

- K.3.1.1 says if the macro is defined to 0 then the users can use the names. If it is 1, then they can't. If it is not defined then it is implementation defined whether the names are reserved.
- There is only the Annex K category and everything else including the floating-point TS, 24731-2 and 24747.
- Martin: Removing Annex K would resolve the issue and there is a proposal to remove it that got significant support.
- We believe the answer is B.
    - This means the functions must be in a separate library.
- *AI* David to write a response for N2174 as an N document.
- Jens: Procedurally do we need to make this what used to be a DR with a committee response. It needs to be in some list of issues.
    - Perhaps Change Requests and Questions, making it CQ (one letter before DR).

12. Programming language C - Extensions for parallel programming - Part 1: Thread-based parallelism r1 [N 2170]

Summary:

Item agreed to in Pittsburg was not added to CPLEX. UK has comments.

The only new change is in Clause 6, Task Execution. Tom Skogland believes this addresses their major concerns. To ready for PDTS, there is the Pittsburg issue (reduction issue) to add. There is a significant amount of technical work to do that. Right now, Clark does not see how to proceed on that issue.

ACTION: Clark to look at adding the reduction issue to CPLEX and report back to the Committee.

ACTION: Convener to send existing document for CPLEX to those NBs that agreed to work on it for their review and comment for submittal as a PDTS.

Details:

- Rajan: What happened to the reduction object -> reduction function agreement from last time (Fall 2016)?
    - Clark: It was unintentionally dropped.
- David: We may be getting comments from the UK.
- Clark: The only techinical change was in Clause 6 (Technical Execution).
- David: Does this address the concerns raised last meeting from Tom S (Open MP).
- Tom S: It does handle the thread local storage concern. There are other concerns but they are smaller and can be handled.
- Clark: The hope is the remaining items pain level is tolerable.

- Tom S: The rest can be handled by additions or workarounds.
- What remains until it is ready for PDTS?
    - The reduction issue brought up in Fall 2016 (Reduction type is too inventive, it will be changed to function calls).
        - Tom S: Can leverage the old blocks proposal.
        - Tao: Don't like the Cilk syntax for this.
        - Clark will talk with Pablo regarding the issue and report back on the reflector.
        - Tom S: We added reductions in Open MP that could help. Will send Clark a link to that.
    - David: We have 7 countries that supported the NWIP. We can send them the draft to get feedback before the next WG14 meeting.
    - Clark: We will likely not have anything new by the post meeting mailing but can say reductions need looking at.
    - *AI* David: Send the CPLEX document to the national bodies that agreed to participate on the work item.
- Tao: What are the other concerns from Open MP?
- Tom S: The current CPLEX document is tuned to work stealing task scheduler, stack stealing approach. It is less prescriptive from a programmer point of view and more restrictive in some areas. The hints are all completely ignorable.
    - Clark: As is spawn.
    - Tom S: You can't specify any new program that couldn't have been there without it. It has potential to use cores, but doesn't add expressibility to the language.
    - Jens: Aren't reductions adding to the language?
        - Tom S: No, since they give task local memory, not thread local memory.
    - Tom S: We do use tasks sometimes as they are a useful abstraction. There are missed opportunities and possibly overoptimization for a specific solution.
- Tao: We have found it is not hard to get compiler optimizations for tasks. We found you can get the compiler to get much more work efficient code. Tasks help with stack analysis, optimization, etc. but they are not threads, and it does matter a lot. Normally talking about concurrency threads matter.
- Tom S. Threads or co-routines. Don't expect threads would ever be sufficient (too heavy).
- Tao: Didn't see how this covers marking functions that could spawn and return without syncing. Is that viral?
    - Part of me wants it to be viral.
    - Clark: Depending on the way the code is structured, you may have to put the annotation on a number of functions, so you could call that viral. But essentially it is important to know that for the programmer, so they can reason about it. It is inventive,

and may not be necessary, but it is worth getting experience. Putting it in a TS is a way to do that.
    - Tom S: We have experience requiring viral annotations and it makes people angry. By changing the names in Cilk+ and adding this, you lose ABI compatibility. If you wanted an asynchronous library, this would break the ABI and API.
    - Tao: I can see the argument if you are only talking about tasks. If you spawn tasks, sync immediately.
    - Tom S: If you want an API asynchronous, you require changing the ABI. Generally, when you think of tasks you want them to be async. Not going to fight having the annotation, but there are probably better ways like compiler optimization.
    - Clark: If a function spawns and syncs before returning, no need to change the signature. If it spawns and doesn't sync before returning, the user would want to know it.
    - Tom S: The user would not care. The programmer should read the documentation or know what they are doing. The self-documenting issue doesn't fly.
    - Clark: It is a way to know when parallel execution is happening or not.
    - Tom S: It is insufficient since you don't know if you yourself are already in a spawned block.
    - Tao: If you have a reduction variable and call the function and it spawns, if you read the reduction variable right away you don't know what you get unless you sync.
- Clark: Expecting some bike-shedding before the document goes out. Names may change.

13. Baseline Edition TR 24772-3 [N 2169]

Summary:

Clive Pygott presented. N2169 addresses C programming language vulnerabilities as part of WG23's work on identifying programming vulnerabilities for all languages. It is based on the C 11 Standard.

Details:

- Jens: The first item in Section 5 is bad (malloc casting). It gives a false sense of security.
- David: It is good if you have a static analysis tool that can check it for you. It is equivalent to using a macro to define it once and use it everywhere.
- Martin: Disagree with item 2 given the paper already given about the weaknesses and issues with it.

- Clark: If you are concerned about vulnerabilities, then providing the summary is not useful since they need the details.
- Rajan: The summary is useful to others, including people whose job is not specifically focused on vulnerabilities.
- Clive: C has well established subsets for safety and security. This document is to show how the main body of issues applies to C. OK with removing the table.
- Clark: If the purpose is just for people focusing on vulnerabilities, then this is worse than useless, but if it is for a different audience then it can make sense.
- Blaine: Can convert the table to an introduction section.
- Martin: May be less contentious if it just identifies problems instead of offering suggestions to fix them.
- David: There is an issue it may devolve into saying "this language is dangerous" which even WG23 is trying to avoid.
- Clive: The annexes of the language independent main document were split up into separate documents like this one.
- Martin: Does the main document talk about areas for compilers or tools to focus on?
- Clive: No, not for specific techniques. It does say to use other tools, not just the compiler to check.
- Jens: 6.11 mixes up concepts in weird ways. Starts off by talking about casts which are bad. Then it talks about conversions which is not the same thing.
    - It also does not point out the problem with malloc, not the casting of the void* but the size parameter being wrong. It should always use sizeof.
    - The implicit conversion for pointers are the good ones. It's the explicit casts that are often the problems.
- Clive: This started from CWE issues and rules from MISRA, CERT, etc.
- Aaron: In 6.5.2 regarding enums, the last bullet does not make sense.
    - Clive: The compiler may optimize away the default case if the rest of the enum is covered.
    - Aaron: Marking it as volatile can defeat static analysis tools.
    - Clive: There are compilers that do this.
    - Jens: Those compilers are broken.
    - Aaron: If the guidance is to work with broken compilers then you lose the static analysis potential problems caught and you are in a worse situation.
- Martin: 6.8.2: Using strncpy has serious usability issues due to misunderstanding of it's usage. Ex. chopping off the null byte. The problem is that it is becoming idiomatic instead of being used as it was intended.
- Aaron: 6.12.2: Seems impossible to avoid all pointer arithmetic. Array access is the same thing.
- Martin: No guidance on VLA's.

- Jens: Using VLA's in functions is fine (size parameter) and adds security. Allocating them is bad.

7. Defect Reports

7.1 Discussion on the Defect Report Process

Summary:

The defect report process is going to change to fit ISO's view of what is a 'defect'. Their view is a 'defect' is a near emergency change to the standard that must be processed in 1 ½ months.

Details:

- Blaine: Can rename DR's into change requests.
- Other ideas: Document reviews, Issues (C++).
- Blaine: Some DR's seem to clump. Ex. Mutexes, atomics, character definitions, etc.
- Clark: C++ has tables, one is sorted by section.
  - Blaine: Willing to do that.
- Blaine: Keep them in HTML format?
  - Jens: Making a patch from that is trivial. Not a problem. For larger changes, having real patches would help.
- Blaine: Can we put up a git repository to the website?
  - Jens: Put a pointer to the git commit in the DR.
- Blaine: How do we differentiate between change requests and a proposal?
  - Aaron: If it was clarification of what is there already published or is this a change?
  - Jens: If it is a normative change is what matters.
  - Blaine: We should tag them with normative vs non as well.
  - Clark: C++ says issues (small) vs proposals (big changes).
    - Most issues end up being clarifications with only a few being changes.
    - Can change from an issue to a proposal possibly?
- So, committee response is the clarification, and any changes can be anticipated content for a new standard version.
- David: Are we happy with the input process or do we want to go to something like github or something else?
  - Each person in the committee can create issues in github.
- Blaine: Whoever creates the issue can be responsible for maintaining it until resolution.
- Proposals will be voted on before making it into a working draft of the standard.

7.2 IS 9899:2011 Defect Reports [N 2148]
In addition to normal DR processing, the following items have new material to consider.

7.2.1 DR#471, cacosh and NaN signs [N 2173]

- The DR is already in C17.
- We can make this a one time exception to the process to allow this change to fix the incorrect PTC.
- This is not to be viewed as normal procedure (it is an exceptional case) and is not to be viewed as a precedent.
- Straw poll: Shall we make a one-time exception and accept the words from N2173 into C17?
    - Vote: 13/1/1
- Leave in closed.

7.2.2 **DRs in REVIEW status that are ready to CLOSE**

DR432: Leave in REVIEW

The DR about normalized numbers caused this one to be reopened into review status.
This was in C17 even though it should not be.
It will be pulled out of C17.
Discuss the paper for the model first then revisit this.
AI Jens to remove DR432 from C17.

DR 493 – Move to OPEN

- Martin: There is a lack of calling out Undefined Behavior, where it is called out in POSIX and C++. The Committee Response should be changed. The implementations should not have to handle a NULL pointer differently.
- Questions were answered in the committee response.
- Martin: There are still outstanding issues.
- Blaine: There are other papers that address some of the issues here.
    - This one will be rolled into a bigger issue.
    - The DR's that address the other issues are listed in the proposed committee response.
- Martin: The copying of mtx_t is not covered.
    - David: It is covered in the committee response.
- Martin may bring up any missed issues.
- Martin: We may have answers, but they are not reflected in the standard text.
    - Rajan: These are answers that are clarifications and words can be a new paper but the committee decided on the response being enough for this.
- Martin: But no words are proposed to be changed.

- David: The committee has decided what needs to be changed (in the linked DR's) and that others do not need changes which are listed here in tihs DR.
- Martin: Found one issue that has not been addressed (on reflector message 14855). Still needs to go through some of the rest of it.
- Martin: How much time should we spend on handling this as a whole instead of individual issues.
    - The issues in this DR can be tackled independently from the rest of them.
    - Easier to get consensus on little things than doing a big paper.
    - We need a way to track the issues or fix the incorrect PCR.
- Martin: I found a number of issues that are inconsistent with what is in the standard including other DR's.
    - Suggesting opening new DR's for the individual points or reopen the DR.
    - The pervasive one is the instances of undefined behavior that is not spelled out in the standard or in Annex J. They need to be specified and cross referenced in Annex J. Ex. Calling mtx_destroy twice on the same object. It should be spelled out.
- Blaine: They can be handled through DR469, DR479 and other DR's.
    - Martin: These DR's do not address the issues nor does Blaine's paper. They don't deal with these issues in particular. Beyond the undefined behavior, there is a contradictory committee response for mtx_init with a NULL pointer which is at odds with what the standard says. The committee says it should return an error, but is not spelled out in the standard. The blanket statement in the section says it is undefined behavior.
    - Rajan: For mtx_init the subsection part overrides the blanket statement.
    - Martin: I don't read it that way. I don't think that interpretation is intended or desirable. POSIX does not require detecting NULL pointer. We shouldn't do something that breaks existing practice.
- Rajan: Can close this and write papers for the disagreements while listing that in the committee discussion.
- Martin: I don't want the NULL issue to stay the way it is with the committee response since others can use the committee discussion as a direction.

- Straw poll: Is everyone content with the answer to question 7? 6/9/0.
    - Result: Make it undefined.
- *AI* Martin to write papers for the other items in DR496 that need attention or that he disagrees with.

DR 500 – Move to CLOSED

To be a part of C2X since it was closed after the deadline.

DR 503 – Move to CLOSED

7.2.3 Prior DRs in OPEN Status

DR 467 – Moved to REVIEW

DR 476 – Moved to REVIEW

DR 488 – Moved to REVIEW

DR 494 – Moved to REVIEW

DR 495 – Moved to FUTURE

- No progress. No new words or attempt at resolving it.
- The committee wants to improve the standard's handling of this topic.
- This is a part of what needs to be done for the atomics formulation so put it into future state instead?
- Jens: We didn't intend this to be used this way, so it should be in the committee response.
- Blaine: I can capture this in the committee discussion.

DR 496 – leave OPEN

- Re: offsetof(type, member-designator)
- Also, that space is the only place where 'member-designator' is used.
- Clark took an action item for this one.
    - Replacing structure member with sub-object designated by member-designator would fix the issue.
    - Sending the change to Blaine and David.
- Clark sent a document to the presenter showing structure-member being replaced by subobject.
- Martin: There was some discussion about member-designator on the reflector and whether it is the right word.
- Aaron: Subobject is not defined anywhere.
- Jens: Can have subobject-designator instead of member-designator.
- Clark: Interesting idea. Editorial.
- Martin: Not sure it does anything to the arrow designator for the member-designator.
    - Clark: Whether the arrow operator should be allowed is the same as allowing address constants.
- Clark: The words that we have allow defining a new type in the type argument. I prefer it doesn't allow it.
    - It is orthogonal to this change but it is a question raised in the DR.
    - There is no constraint that doesn't allow defining a new type in a cast.
- Martin: Can't define a new type with a comma in it.
- Clark: We should all agree this is progress and go with it. Don't hold it up for an orthogonal issue.
- Martin: Sounds like progress.

- Blaine: This can answer the first question in the DR. We can answer the second question as well with this as well.
- Clark: For the third question, we didn't intend to allow it, it fell through via the formulation. It does permit it, but not by intent.
- Martin: Joseph was implementing this in GCC with an intrinsic which allows creating new types. But it is not possible with the macro.
- Jens: For question 3 say it is permitted. For 4, we can say with a comma it is not possible, but otherwise it is.
- Aaron: Tried 6 different compilers and none of them allowed defining new types in the macro. In this case we are just standardizing existing practice.
- Clark: We can use the words provided to answer the first two questions and use a committee discussion for the remaining questions.
- Blaine: Expecting answering yes, yes, not the intent.
- Martin: No problem with that, but want to have that reflected in the text somehow. Footnote or normative doesn't matter.

DR 497 – Moved to REVIEW – subject to editorial review (per normal)

- Larry: I will probably not put it in the index since it follows right where it should be in the index.

DR 498 – Moved to REVIEW
DR 499 – Leave in OPEN

- Move the committee discussion into proposed technical corrigendum.
- Martin: There was some concerns in the reflector (14628, 14629, 14632).
- Until we get new words, not moving the discussion into a TC.
- *AI* Clark: Will supply new words for DR499 based on reflector message 14632.

DR 501 – Leave OPEN

- Move the committee discussion into a proposed technical corrigendum.

DR 502 – Moved to REVIEW

7.2.4 - New DRs in OPEN Status

None

7.2.5 - DRs with FUTURE status

None to discuss.

7.3 TS 17961:2013+Cor 1:2016 Defect Reports [N 2150]

None.

7.4 TS 18661 Defect Reports [N 2149]

DRs in REVIEW Status

DR5: Move to closed.
DR6: Move to closed.
DR7: Move to closed.
DR8: Move to closed.
DR10: Move to closed.

DRs in OPEN Status

DR9: Move to review.
DR11: Move to review.
DR12: Move to review.

- David: Prefer to have the first to changes with regards to "can be interpreted as" being a note/footnote.
    - Clark: I makes sense the way it is too. Can keep it as.

DR13: Move to review.
DR14: Move to review.

In addition to normal DR processing, the following items have new material to consider.

1. TS 18661 DR 15 - Characteristic macros for non-arithmetic formats [N 2171]

    - Blaine: The second change after the "except" seems to need its own sentence.

    Result: Copy suggested TC as Proposed TC, leave OPEN.

2. TS 18661-1 DR 16 - tgmath cbrt macro [N 2178]

    Summary:

    This fixes an example in a TS, but it seems that the proposed fix is wrong. Clark agree to attend a future meeting (telcon) with the C Floating Point Group to discuss this.

    Leave OPEN.

    Details:

    - Some messages in the reflector.
    - Clark: Feels weird since it is updating an example, and also being over specified.

- Jens: The (X) should be moved outside of the generic statement.
- Clark: The reason for this change is to show how the static rounding modes affect standard library functions.
- This change is not necessary.
- Rajan: Why is it not necessary?
- Clark: The use of a tgmath macro is affected by the static rounding mode. Using the address of the function may not follow the static rounding mode. Don't like the original fix, nor the fix to the fix. Seems to be going further in the wrong direction. There are aspects with interactions with tgmath and static rounding modes that need thought, but should go back to the drawing board.
- Blaine: Maybe have a simpler example. Perhaps don't use _Generic in the example.
- Clark: They are saying the example of _Generic needs to be fixed.
- Larry: Having the (X) in the definition can cause compiler diagnostics due to type mismatch.
- Fred: Can be editorial to move the (X) outside.
- Clark: This is the third attempted fix and it bothers me that this is for an example.
- There are a variety of issues with this.
- Rajan: I would love to see a paper showing the issues, so we can handle them.
- Clark: I can join a telecon, but probably not the next one.
- I am good with this being a DR, but not a proposed TC.
- The issue is still unresolved, and Clark will attend one of the meetings to help resolve it.
- *AI* Rajan to invite Clark specifically to the meetings (not just in the reflector) to help resolve this issue.

8. Other Business

8.1 Old DR process replacement

- Replace DR's (and TC's) with CR's: Clarification Requests
- Blaine has changed the DR log to use CR's and listed what is C17 (if it was previously closed and will go into C17) and just Closed if it is a discussion/clarification, and C2X for things that are closed and will go into C2X.
- Rajan: DR 473 should be C17.
- Clive: TC's are tied to Defect Reports, not the CR's.
- Jens: Want a link to this document despite new N documents.
- Barry: It should be on Keld's site to get the latest list.
- Aaron: This has worked well with C++.
- Blaine: We should have a rolling document. Don't need a history.
- Martin: WG21 uses Bugzilla to track issues, is the summary document automatically generated?

- Clark: Different groups in WG21 do different things. Pretty sure there is a process for libraries that is done to create the list.
- Blaine: Open to ideas. If we want to go off github or something like that is fine. My concern is whether we'll get the well thought out papers we get now.
- Barry: There are differences between the sub-groups in C++ and similar for TS's.
- Tom S: You can have the change requests being a mergeable object.
- Martin: Our process is closer to POSIX but uses a bug tracking database. Anyone can create a bug there.
- Barry: For POSIX it is resolved in meetings and they don't need papers.
- David: We want to move away from N documents as much as we can.
- Blaine: Do we want to open up the standard to bug fixes?
- Barry: In WG21 we have a working paper, and anyone can get it.
- David: I don't think it is correct to have the drafts after and including the DIS be open.
- Jens: If we make it public to everyone, anyone can put in anything. Reading c.lang and other groups, it is not good and we should have some form of moderation.
- Aaron: In WG21 core, it has not been an issue. All the requests we get seem to be professional.
- Clark: The contents of the issue list are moderated by someone, so you don't see them.
- Clive: I monitor the MISRA change request and we don't get flames. The biggest problem is people not understanding and only result in a very small number of real issues. Takes a lot of time.
- Blaine: What is the process for recording the discussions?
- Barry: It is entered by someone as a summary form.
- Blaine will talk with Jens and see what C++ is doing.
- Martin: It would open up the process to people outside the committee as well.
- Jens: FAM issue came that way (someone outside contacted him).

9. Resolutions and Decisions reached

9.1 Review of Decisions Reached

Forward C17 draft to SC22 after editorial review for FDIS Ballot.

9.2 Review of Action Items

Carry Over:

- Blaine: Reconcile N2019 and N2026 for DR469
- Clark: DR 496: Write a draft of a Proposed Technical Corrigenda.
  - Done
- Convener: Coordinate with WG21 on the mechanics adding a ‘C’ or ‘P’ designated papers for proposals to WG14.

New:

- Jens: Email the details of the web server to David Keaton to bring up in the meeting along with the updated document (addressing Joseph's comments). - Done
- Jens: C17 draft: 7.26.6.2#2: DR416: Missing the new second sentence. - Done.
- Jens: C17 draft: DR473: Mislabeled lgamma. - Done.
- David Keaton: Update SD3 to go through the obsolescent features and future directions to decide what to remove for C2X.
- Martin: Split N2145 into two proposals. One to define Generic functions, another to work on compound literals.
- C++ liaison members: Indicate to WG21 that WG14 is good with N2160 (Comma omission and deletion).
- Larry: Write a paper with any objections to N2161 (left shift behavior).
- David: Update SD3 to point to N2165 for attributes.
- David: Add N2186 to SD3.
- David: Write a response for N2174 as an N document.
- Jens: Remove DR432 from C17. - Done.
- Martin: Write papers for the other items in DR496 that need attention or that he disagrees with.
- Clark: Supply new words for DR499 based on reflector message 14632.
- Rajan: Invite Clark specifically to the CFP meetings (not just in the reflector) to help resolve the cbrt example issue (CFP DR16).
- David: Send the CPLEX document to the national bodies that agreed to participate on the work item.

10. Thanks to Host

11. Adjournment - Adjourned Thursday, 2 Nov 2017, 1200 hrs.

Minutes for the PL22.11/US TAG Meeting, Tuesday, Oct 31, 2017, 16:00

Name	*Organization*	*P/A/NB*	*Comments*
David Keaton	Keaton Consulting	P/USA	WG14 Convener
Daniel Plakosh	CERT/SEI/CMU	A/USA	WG14 ISO eCommittee Secretary
Blaine Garst	The Planet Earth Society	P/USA	
Rajan Bhakta	IBM	P/USA	
John Parks	Intel	P/USA	PL22.11 Chair
Clark Nelson	Intel	A/USA	
Fred Tydeman	Tydeman Consulting	P/USA	PL22.11 Vice Chair
Barry Hedquist	Perennial	P/USA	PL22.11 IR
Tom Plum	Plum Hall	P/USA	dialed in
Martin Sebor	Red Hat	P/USA	
Aaron Ballman	GrammaTech	P/USA	
Clive Pygott	LDRA	P/USA	
Visiting non-members			
Michael Wong	Codeplay / ISOCPP	CA	
Jens Gustedt	INRIA	France	
Larry Jones	Siemens		WG 14 Project Editor

1. Approval of Agenda (pl22.11-2017-00003) – Unanimous Consent (Ballman, Garst)

2. Approval of Previous Minutes (pl22.11-2017-00001) – Unanimous Consent (Ballman, Garst)

3. INCITS Antitrust Guidelines and Patent Policy

   Please read. No discussion. Questions to Lynn Barra.

4. INCITS official designated member/alternate information

   Please let Lynn Barra know of any changes.

5. Identification of PL22.11 Voting Members
   1. PL22.11 Members Attaining Voting Rights at this Meeting

      GrammaTech (Aaron Ballman)

      The Planet Earth Society (Blaine Garst)

   2. Prospective PL22.11 Members Attending their First Meeting

      none

6. Members in Jeopardy
   1. Members in jeopardy due to failure to return Letter Ballots

      None

   2. Members in jeopardy due to failure to attend Meetings

      None

      1. Members who retained voting rights by attending this meeting

         None

      2. Members who lost voting rights for failure to attend this meeting

         None

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

      None

7. Procedures for Forming a US Position

   per normal

8. New Business

   New individual sign in process for meeting on INCITS web site.
   Be sure to return the INCITS Membership Agreement prior to Dec 1, 2017.

9. Next Meeting

   April 24, 2018 – Brno, CZ with SC22 WG14.

10. Adjournment – unanimous consent (Keaton, Tydeman)

    Meeting adjourned at 16:30, 31 October 2017