WG14 N1493
INCITS PL22.11/10-001
Reply To The Attention Of: Barry Hedquist
Email: beh@peren.com

ISO/IEC JTC1 SC22/WG14 AND INCITS PL22.11
APRIL 19-23, 2010 MEETING MINUTES

Meeting Location:

Dept.Sistemi e Informatica, Università di Firenze
3, via santa Marta, 50139, Firenze, Italy
++39 055 4796526

Host Contact information:

Host Contact information:
Host web site.
Email: Enrico Vicario

Meeting Host:
Meeting Dates: 19-23 April, 2010
Meeting Times:
19 April 2010 15:00–18:00
20 April 2010 15:00–19:00
21 April 2010 15:00–18:00

1. OPENING ACTIVITIES

1.1 Opening Comments (Benito, Vicario)

Attendance at this meeting was severely impacted by the Iceland volcano ash plume that drifted over Europe and resulted in cancellation of many flights. Consequently, with many were unable to make the trip. The first order business Monday morning was to set up some type of teleconference capability and set up a schedule that allowed as many as possible to participate via this teleconference. There will be no proposal adoption at this meeting, but hopefully we can make progress on the proposals. We will also need to discuss the possibility for revising the meeting schedule to limit the impact of this on the revision schedule.

1.2 Introduction of Participants/Roll Call

Name	***Organization***	***NB***	***Comments***
John Benito	Blue Pilot		WG14 Convener
Blaine Garst	Apple	USA	
David Keaton	CERT	USA	PL22.11 Chair
Tana L. Plauger	Dinkumware, Ltd	USA	
P. J. Plauger	Dinkumware, Ltd	USA	
Jim Thomas	HP	USA	
Hans Boehm	HP	USA	

Paul McKenny	IBM	USA	
John Parks	Intel	USA	
Clark Nelson	Intel	USA	
Nat Hillary	LDRA	USA	
Herb Sutter	Microsoft	USA	
Larry Wagoner	NSA	USA	
Douglas Walls	Oracle	USA	HOD – USA, PL22.11 IR
Barry Hedquist	Perennial	USA	PL22.11 Secretary
Tom Plum	Plum Hall, Inc.	USA	
Fred Tydeman	Tydeman Consulting	USA	
Michael Wong	IBM	CAN	
Rajan Bhakta	IBM	CAN	HOD - CAN
Steve Montgomery	MISRA-C Liaison	UK	
Larry Jones	Siemens		WG14 Project Editor
Alessandro Pinzuti	Università di Firenze	IT	HOST

1.3 Procedures for this Meeting (Benito)

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

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

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

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

Straw polls: No straw polls will be taken at this meeting.

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

The meeting times were modified to allow participation via teleconference. Meeting times were 1500 – 1900 hours, local time, to allow teleconference participation from the USA West Coast.

Teleconference participation was conducted over Skype using a network bridge provided by Intel. Telephone numbers and pass codes are posted on the WG14 Wiki provided by Dinkumware. Static IP addresses for those in Florence were provided by our host. The system worked remarkably well.

Barry Hedquist is Meeting Secretary.

1.4 Approval of Previous Minutes (N1422) (Hedquist)

Several comments for typos, etc.
Notes:

Attendees - change Walls to PL22.11 IR v. PL22.16

Minutes were modified per Fred's email, and editorial changes and approved.

Final Minutes are N**1456.**

1.5 Review of Action Items (Hedquist)

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

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

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

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

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

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

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

ACTION: Nick Stoughton and JB will work on getting TR 24732-2 to ITTF.
CLOSED – JB is working with Open Group to make this happen.

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

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

ACTION: Nick and Fred to come up with new wording for N1416, Errno at Thread Create.
DONE - N1427

ACTION: Fred, Jim Thomas, Doug Gwyn to come up with new wording for N1419, Creation of Complex Value.
DONE - N1431

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

1.6 Approval of Agenda (N1434) (Benito)

Revisions to Agenda:
Added Items:

2.1.1 – WG14 Study Group, C Secure Coding Guidelines Liaison Report (N1450) (Keaton)
5.19 – Blocks Proposal (N1451) (Garst)
5.20 – Atomic Proposal (N1452) (Garst)
5.21 – GC Florence (Garst)

Deleted Items: None
Agenda approved as modified.

1.7 Distribution of New Documents (Benito)
None

1.8 Identification of National Body Delegations (Benito)
US
Canada

1.9 Identification of PL22.11 Voting Members (Tydeman)

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

1.9.1 PL22.11 Members Attaining Voting Rights at this Meeting

Nat Hillary, LDRA Technology

1.9.2 Prospective PL22.11 Attending Their First Meeting

Larry Waggoner, National Security Agency

1.10 Next Meeting Schedule (N1436) (Benito)

We will meet again as early as the end of May to make up for the lack of physical attendance at this meeting.

The Fall 2010 meeting and lodging will be held in Geneva, Chicago, USA.

Comfort Inn & Suites Geneva
1555 East Fabyan Parkway
Geneva, ILL
USA
Tel: +1.630.280.8811
Fax: +1.630.208.7844
Email: sales@comfortinngeneva.com
Web Site: www.ComfortInnGeneva.com

Special Rates: $75/night - Standard Room; $81/night - Small Suite
The Special Rate is NOT available if reservations are made on the website.
Includes a Continental Breakfast, and Internet access.

WG21 and PL22.16 will be meeting the following week at Fermi National Accelerator Laboratory in nearby Batavia, ILL.

2. LIAISON ACTIVITIES

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

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

2.2 WG21 / PL22.16 (N1432) (Plum, Benito, Sutter)

WG21 has published a Final Committee Draft, FCD 14882, and is now in FCD Ballot. The ballot closes 26 July, 2010.

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

Herb Sutter mentioned an alignment proposal, but nothing we've seen.

2.3 PL22 (Plum)

No Report

2.4 Linux Foundation (Stoughton)

No Report

2.5 Austin Group (Stoughton)

Santa Cruz: TR balloted and approved with comments. Ballot resolution underway for the next 1-2 meetings. Several Issues raised via Austin Group Aardvark reports requesting WG14 action. ACTION: Nick will put together a summary of Austin Group issues requesting WG14 input. No immediate time constraint.

2.6 WG11

Santa Cruz: No Report

2.7 WG23 (Benito)

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

Met last week, TR balloting is complete, and should go to ITTF soon. Next meeting is in Hawaii.

2.8 MISRA C

No Report

2.9 Floating-point Study Group

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

information. Working documents being developed. Goal is to produce a proposal for a Type 2 Technical Report.

2.10 Secure Coding Study Group (N1450) (Keaton)

Florence Introduction: At the Santa Cruz meeting, WG 14 formed the C Secure Coding Guidelines Study Group (CSCG SG) to study the problem of producing analyzable guidelines for C99 and C1x. The CSCG SG has been meeting by teleconference every two to four weeks.

Anyone interested in this report, or this SG, should contact David Keaton.

2.11 Other Liaison Activities

None

3. EDITOR REPORTS

3.1 Report of the Rationale Editor (Benito)
No rationale has been developed for items being added to the WP. The proposal paper's author is responsible for writing the rationale.

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

3.2 Report of the Project Editor (N1426) (Jones)
Open Issues:
1. The N1371 21.5 changes need to be completed.
2. The index probably needs work, particularly for the newly added material.
3. The list of changes from the previous edition needs to be populated.
4. The library synopses need to be examined and extraneous headers removed.

3.3 Status of TR 24731-1, Bounds Checking, Static (Meyers)
Santa Cruz: Do we need a new Editor? Yes.

3.4 Status of TR24732, Decimal Floating Point (Kwok)
TR 24732 is now published. Type 2, must be paid for.

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

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

Convenor is coordinating with Andrew Josey, Austin Group, for editor support so we can make the final changes need to get this published.

4. FUTURE

4.1 Future Meeting Schedule

Spring 2011 - Open
Fall 2011 - Portland (tentative)

4.2 Future Agenda Items

4.3 Future Mailings

Post Florence	10 May 2010
Pre Bolder	10 May 2010

5. DOCUMENT REVIEW

5.1 Extensible Generic Math Functions (N1340) (Plauger)

Santa Cruz Notes:
This is the same paper we discussed Santa Cruz, along with N1401, Type Generic Macro Support. N2401 builds on N1340. When that paper was presented, Ulrich Drepper mentioned that the GNU compiler also implements features that can be used to implement type-generic macros. For the committee's consideration, those features are described in this paper as well as an alternate approach.

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

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

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

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

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

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

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

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

Yes – 12
No – 4

Abs - 8
--- End Santa Cruz Discussion ----

Florence Discussion:
See 5.9, N1441

5.2 C and C++ Thread Compatibility (N1447) (Crowl)

N1447 presents what is believed to be the consensus proposal of the C and C++ compatibility mailing list, and is a revision to N1414 discussed in Santa Cruz. It incorporates detailed proposed wording for both C and C++ standards.

The revision to C++ is now in FCD Ballot, and none of the proposed C++ wording in this document, N1447, is contained in the C++ FCD.

Santa Cruz Notes:

[From the paper's Introduction (N1414)] The compatibility between the C and C++ threading facilities is important to many members of the respective communities. There are at least three levels of compatibility: critical, important, and desirable. Of these, the C and C++ committees should commit to achieving critical and important compatibility. This paper analyses the compatibility between current draft standards, and recommends several actions to improve that compatibility.

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

[Discussion]

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

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

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

Florence Discussion:
Paul McKenny – alignas Operator. Paul covered the proposed changes to C. Blaine concerned about not having a more complete solution, although he supports the concepts of this paper. There are no specific objections to this paper. No decision at this meeting.

5.3 Mutexes in C and Compatibility with C++ (N1437) (Williams)

Background: From N1473
The current C1x draft (WG14 N1425) has a single mutex type (mtx_t), the properties of which are specified when each instance is initialized. On the other hand, the C++0x draft (WG21 N3000) has 4 distinct mutex types (std::mutex, std::timed_mutex, std::recursive_mutex and std::recursive_timed_mutex), each with predefined properties.

This discrepancy was highlighted by Lawrence Crowl in his paper on C and C++ compatibility in the thread library (WG14 N1423, WG21 N2985), and has since led to much discussion on the C and C++ compatibility reflector, and a series of informal teleconferences. This paper has come out of these discussions.

Two primary issues are discussed:

1. *Sharing of mutexes between C and C++ code, and*
2. *Performance of mutexes in C code.*

Florence Discussion:
David Keaton walked us through N1473, which contains proposed wording for the revision. This paper does not reflect an implementation. It is standardese. PJ does not believe there is anything here that is markedly better than existing practice. Without evidence of this being a better way, PJ is opposed to this paper. Blaine would like to take a look at the performance issues, and sample implementation in the next day, and come back to this later. Herb proposed a teleconference on this paper at a later date, which was agreeable. Herb is available after May 10. Tom pointed out that WG14 is highly unlikely to approve anything for which there is no implementation, or field experience.

5.4 Updates to the C++ Memory Model Based on Formalization (N1446) (McKenney)

This paper is a WG14 version of WG21 N3074. The proposed wording for C++, Core Issue 2, 1.10p14, has NOT been adopted in the C++ FCD. ???? (Why - check BR doc).

Florence Discussion: Paul McKenny presented N1446. Blaine expressed concerned over some of the wording. David K concerned about the proposed wording as well, re: paper changed to "might be" changed to "is". (5.1.2.4p2). Paul believes N1425 addresses Blaine's concern. David's concern the proposed words Invalidate the base memory. Clarify the scope as *normal memory*? Then we have to define *normal memory*. What happens to other forms of legal C declared memory? Needs to be addressed. Who will do that? Blaine thinks several of these papers that deal with atomics needs to be better understood and viewed w/r/t Blaine's N1462. Mike Wong believes a better understanding is needed w/r/t what atomics bring to C. Blaine suggest deferring discussing these papers until tomorrow, but Paul sees nothing in Blaine's paper that resolves the items they have been discussing.

5.5 Dependency Ordering for C Memory Model (N1444) (McKenney)

N1444 proposes an addition to the C memory model and atomics library for inter-thread data-dependency ordering. This proposal is based on a similar proposal for C++ (WG21 N2664), most of which was adopted for the revision to C++. This proposal admits a trivial implementation, limiting significant implementation investment to those compilers and platforms where that investment will be recovered.

Douglas Walls: If I understand the proposal correctly it adds: explicit program support for inter-thread data-dependency ordering. Programmers will explicitly mark the root of tree of data-dependent operations, and implementations will respect that ordering.
It appears to add another enum to memory order that may be supplied to operations for which data-dependency semantics are permitted. As I've noted in the past I do not fully understand all of these various forms memory ordering operations.

As this potentially aids performance for TSO I do support it. Though I'd rather see vast simplification to only one memory ordering for the standard.
--- End SC22WG14.12019 ---

Florence Discussion: Tuesday
Paul McKenney presented an overview of N1444, which evolved into a discussion of details with Blaine regarding the types of hardware that could be supported, and memory ordering schemes. Why was 'lock' changed to 'mutex'? To be consistent with the same change made to C++0X. The term is specific to the facilities provided in C and C++. Blaine thinks the atomics section needs to be reviewed simply for clarity. Clark wants to know if everyone is on board about the concepts presented in this paper. David K thinks it looks good. Blaine thinks his atomic proposal is an add on. Tana points out that C++ is moving forward, so if we have a comment on it, we should make it. Tom suggests we have no other alternative approaches before us. Blaine thinks the level of detail is fine for those who need it, but not for the C Standard, but also agrees with Tom. Tom generally agrees with the paper presented. Walking through the proposed wording is likely to be unproductive. If anyone has comments, forward them to Paul and Clark.

5.6 Explicit Initializers for Atomics (N1445) (McKenney)

Florence Introduction: C/C++ Compatibility Issue

N1445 describes a compatibility issue between C++ and C for atomics. Unlike C++, C has no mechanism to force a given variable to be initialized. Therefore, if C++ atomics are going to be compatible with those of C, either C++ needs to tolerate uninitialized atomic objects (ed - that's not going to happen), or C needs to require that all atomic objects be initialized (ed - that's what we have to do). There are three basic cases to consider: 1) C static variables, 2) C on-stack auto variables, and 3) C dynamically allocated variables.

Florence Discussion: [Wed]
Paul McKenny presented and overview of N1445. Any objections to the general concept – all atomics in C be initialized? None.

5.7 General support for type-generic macros (N1441) (Nelson)

Santa Cruz Introduction:

N1401, Type-generic Macro Support, builds on N1340. When that paper was presented, Ulrich Drepper mentioned that the GNU compiler also implements features that can be used to implement type-generic macros. For the committee's consideration, those features are described in this paper as well as an alternate approach.

Santa Cruz Discussion:

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

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

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

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

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

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

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

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

Yes – 12
No – 4
Abs - 8

--- End Santa Cruz Discussion ---

Florence Introduction: N1441 contains proposed wording to specify the _Generic construct proposed in N1404. In Santa Cruz, there was consensus (12-4-8) to adopt that approach.

SC22WG14.12021:
Douglas Walls, indicated support for this paper over N1340.

SC22WG14.12011:
Douglas Gwyn, n1441; what about use on types for which no default or….
I would urge the Committee to minimize invention of new linguistic features for C. Indeed, C99 type-generic macros ought to be revisited to determine if they have been sufficiently useful to offset their difficulty in complete specification and implementation. Similarly for sizeof(VLA), which is harder to make good use of than simply using the expression that determined the VLA length in the first place.

Some invention may be needed when there is a demonstrated need for a feature, not portably available under C99, across a wide variety of platforms. But it should not add substantial complexities.
[end SC22WG14.12011]

SC22WG14.12015
Herb Sutter: Strongly supports DG position above.

Florence Discussion:
John Parks presented N1441, proposed wording for type generic macros. The general sense is that all of this stuff (type generic macros) is a kludge, but we seem to be stuck with them. PJ noted that the approach in N1441 was more complete that that of N1340.

5.8 Comparison Macros (N1442) (Plauger)

Florence Introduction: C/C++ Compatibility Issue
The words in the current C1X WP say nothing about whether or not the two arguments in a comparison macro must have the same real-floating type. The same macros in the C++ FCD require the types be the same. We should either change the C1X WP to require the same types, or file a comment against the C++ FCD to not require the same type. PJ prefers the first choice.

Reflector: WG14.12023: Douglas Walls argues that C99 places no requirement that the two types be the same, and prefers the second option, filing a comment against the C++ FCD. Oracle's implementation allows different types.

Florence Discussion: [Wed]
PJ presented the background of the comparison macros, and the incompatibility between C++0X and C99. Bill is willing to file a comment, if that's what we want to do. HP and Oracle want to be able to use different types, which is what the C Standard presently allows. PJ views doing so requires a degree of compiler magic, but agreed to let the types be different after further discussion. Agreement that the C Standard may not be 'clear'. If the types are mixed, they can promoted to the common type. Tom: maybe we need to say the usual type balancing rules apply. It was never intended that compile errors result if the types are different. PJ regarded this as sufficient direction.

5.9 Subsetting the Standard (N1443) (Plauger)

Santa Cruz: N1395, Wakker, Netherlands Contribution on the Growth of Programming Language Standards, SC22 N4484.

N4484 discusses the growing set of features being added to Standards for programming languages, and the problems created by requiring the implementation of those features on all systems, particularly specialized embedded processers. Creating a conformance category called a Conforming Subset Implementation, which would allow conformance to a subset of the entire standard, would help. An example in the C Standard is 'freestanding', however some believe more flexibility is needed.

Santa Cruz Discussion: In Santa Cruz, Plauger expressed support for the concept, others expressed concern of its practicality. No action was taken.

At the SC22 Plenary, the paper was discussed, and the following resolution (09-16) adopted:

Netherlands Contribution on Growth of Programming Language Standards JTC 1/SC 22, noting the discussion on SC 22 N 4484 (Netherlands Contribution on the Growth of Programming Language Standards), recommends its Working Group Conveners to distribute SC 22 N 4484 on the Growth of Programming Language Standards to their Working Groups as a JTC 1/SC 22 recommended document on guidance to address the issues identified in this document.

Florence Introduction:
N1443 proposes three feature test macros be added to the C1X WP:

*__STDC_NO_COMPLEX_*
*__STDC_NO_THREADS_*
*__STDC_NO_VLA_*

and suggests the Committee also consider a similar macro for atomics either as part of threads, or separately.

Florence Discussion: Subsetting can allow more implementations to achieve conformance to C99 in general, without having to implement features not needed by an implementer's customer base. The three cited above were the ones that immediately came to mind when PJ put this paper together. In doing so, PJ tried to follow the guidelines presented in Wakker's paper (N1395). How does this affect conformance w/r/t strictly conforming? Conformance can be achieved using these macros. The lack of any feature, such as threads, would have to be identified. Consider adding 'freestanding' to the above list. Atomics? General support of the concept of subsetting. MISRA C says don't use complex, or VLAs. Expect to see a more flushed out proposal at the next meeting.

5.10 Completeness of Types (Feather) (N1439)

N1439 proposes changes to the definitions of types to clarify that completeness of a type is a 'scoped property of a type', it can vary from place to place in a translation unit, without altering the type itself. While that's clear from the current text, it's not clear from the definition.

Florence Discussion:
Clive is not available. Tom: In most places where we say an object type, we should be saying a complete object type. Completeness is a scope property, once a type is completed, it is a complete type. There are function types and object types. Within an object type, it may be complete of not. Once it is completed, it is a complete type. Larry believes Clive is correct, but it's a lot of work.

5.11 Constant Expressions (Adamczyk) (N1448)

Florence Introduction:
N1448, asks if the following code is valid in C

```
int i = 1 || (2, 3);
int main() {
  return 0;
}
```

A literal reading of the C99 Standard suggests the comma operator is valid because it appears in an subexpression that is not evaluated, which differs from C89.

Florence Discussion: Douglas Walls is opposed to this change. Tom: The net of Steve's proposal is the compiler's would have to issue a diagnostic where in the past they did not. Tom suggested that the expression above may not even be a 'constant expression'. If it's not, the comment about the comma operator does not matter. Steve suggests that since the initialization is static, it must be a constant expression. Tom changed his mind. David K points out there is a footnote in 6.6;p11 that addresses this.

ACTION: Convenor to discuss this with Steve Adamczyk.

5.12 threads and errno (N1427) (Tydeman)

Florence Introduction:
N1427 proposed to specify, for threads, each thread has its own errno.

Florence Discussion:
The initial state of the errno is indeterminate. The proposed footnote tells us nothing. Suggested removing it.

5.13 ilogb (N1428) (Tydeman)

Florence Introduction: Contains three proposals regarding ilogb.
Background: As a function that maps reals to reals, sqrt(-1.0) has no usefully definable result, and the operand is invalid, so is a domain error. In like vain, as a function that maps reals to integers, ilogb(0.0) has no usefully definable result, so it should be a domain error. In addition, IEEE-754-2008 requires ilogb(NaN or infinity or 0) to raise invalid.

Florence Discussion:
We do we really want to do about new 754 standard. Do we really want to put 754 requirements in the C standard? Clark considers this the nose of the camel. PJ does not want to tinker until we get a final report, in the works, w/r/t 754 impacts on the C Standard. No support for Proposal 1 or 2

Proposal 3 in this paper addresses an incorrect statement in C99 (really a DR). F.9.3.5. PJ thinks the case is so unlikely, has not happened in any machine he knows about. Jim T believes the statement is OK as is.

5.14 fabs (N1429) (Tydeman)

Florence Introduction: Defect Report
Background: IEEE-754-1985 suggests and IEEE-754-2008 requires fabs(NaN) to affect just the sign bit.

Florence Discussion:
PJ disagrees. As with the earlier remark, PJ does not want to tinker with this. Different nose of the same camel. Meta rule: Changes between 754-1985, and 754-2008 will wait until the FP Study Group is finished with their report.

5.15 negate (N1430) (Tydeman)

Florence Introduction: Defect Report
Background: IEEE-754-1985 suggests and IEEE-754-2008 requires negate(NaN) to affect just the sign bit.

Florence Discussion:
Same meta rule as above. Wait on FP Study Group. PJ prefers to leave it alone.

5.16 Creation of complex value (N1431) (Tydeman)

Florence Introduction: Three proposals on the creation of complex values.

Problem:
*(x + y*I)*
will NOT do the right thing if "I" is complex and "y" is NaN or infinity. It does work fine if "I" is imaginary. Users and library implementers have noticed this deficiency in the standard and have been surprised that there is no easy to use portable way to create a complex number that can be used in both assignment and static initialization. Three solutions are proposed.

Florence Introduction:
SC22WG14.12018
Douglas Walls: I support this proposal.
I prefer solution #3 macros over proposal #2 due to the ambiguous case talked about in the footnote. I think that footnote will need to become normative, not a footnote. But maybe I don't understand :-) I dislike proposed solution #1 as type punning can lead to aliasing issues.
--- end SC22WG14.12018 –

Florence Discussion:
Proposal 1: uses type punning
Proposal 2: invention from EDG
Proposal 3: adds new macros. HP has been shipping this solution for several years.

PJ: we need a notation. Tom: likes #3, and the names are as good as anything. Blaine, consider an operator over a macro. Douglas support the macro solution, but could support an operator solution as well. Jim T: If compilers supported the imaginary type, operators would not be needed. PJ: That's the problem, Imaginary is optional.

5.17 Assumed types in F.8.2, Expression Transformations, (N1435) (Tydeman)

Florence Introduction: Defect Report

Problem: The integer constants should be double constants.

Add a paragraph to F.8.2 Expression transformations

*While the following code shows (and assumes) **double** type, the same issues apply to all floating types. Change existing F.8.2 integer constants (0, 1, 2) to double constants (0.0, 1.0, 2.0).*

Florence Discussion:

The reference should be F.9.2 in the current WP. Clark: See no motivation for this change. PH agrees with Clark. Fred has worked with people that have found this confusing.

5.18 Ballot Resolutions, SC22 N4491, DTR 24731-2 (N1440) (Benito)

Florence Introduction: Two comments were submitted by Japan, both of which were editorial, and have been accepted. The document and responses have been sent to SC22.

5.19 Blocks Proposal (N1451) (Garst)

Florence Introduction: N1451 describes a feature called 'blocks', a form of closures, as already implemented by Apple. In C, the term 'block' is already well defined, so it suggests that the term 'closure' to avoid confusion with the existing term 'block'. Usage of terms such as _block, Block_copy, et al, in this paper, simply reflect existing practice by Apple, and would have to be changes to something like _closure, Closure_Copy, etc.

N1451 also cites N1270 as "Apple's Extensions to C" as a prerequisite. However, N1270 is a set of draft minutes, so the cite is wrong.

This paper is incomplete in that the proposed wording reflects 'block' terminology rather than something the Committee would need to decide on, such as 'closure'. Also, the cite needs to be corrected.

Florence Discussion:
[Wed]
Blaine went through a slide presentation on Blocks (On the Wiki, BlocksFlorence2010.pdf).

Tom: There is likely to be a involved discussion on how blocks relate to lambdas in C++.
Blaine: The main difference is that lambdas are not closures. It's harder to share information in lambdas. Nearly equivalent, but they are different facilities.

Then covered N1451, overview section. There are four basic elements:

1) a new form of compound type paralleling function types
2) a new closure literal construct that creates a pointer to a closure object,
3) a new storage class, __block, representing closure object scope duration, and
4) language primitives Block_copy and Block_release.

This is new material for most of us, and there is some doubt as to whether or not most even understand the paper. More discussion will take place at the next meeting. If anyone has specific questions, contact Blaine. The reference to N1270 should be N1370, Apple's Extensions to C.

5.20 Atomic Proposal (N1452) (Garst)

Florence Introduction: N1452 is an alternate proposal for atomics that differs from the one already adopted (N1349 ??) for C1X.

Blaine Garst: My N1452 is incomplete - a discussion of allocated memory w.r.t. atomic objects is necessary, and a revision should be expected before discussion. As it stands, I don't think allocated memory is fully consistent with existing C Standard atomic library objects since they seem to start out with undefined values but the process of establishing initial values before visibility elsewhere is not defined yet is essential for determinism. This seems bad. Requiring "0" initial values (for integers at least) and that calloc implement memory-store barriers seems sufficient, but may be too costly - a new "alloc" with alignment, realloc, memory-store barrier, initial values, may be the best alternative. Complete treatment may require new library support, but let it be minimal instead of maximal.

Florence Discussion:
Blaine Garst presented N1452. His view of using the language to program directly to the hardware with regard to synchronization of operations across multiple threads of control. The paper is essentially a white paper that could be a starting point for development of a more complete proposal. It consists of adding an '_atomic' type qualifier to volatile marked memory with a specified set of results. The concept described here does not have wide industry use. Tom favors a sequentially consistent model for C and C++.

Clark: Three questions. 1) if added, do we need to describe interaction w/library atomics. 2) if compiler is free to reject the request, what are the implications on strict conformance, 3) how would this impact C language schedule.

Blaine: 1) compatible with atomics library, this is just an add on. 2) Blaine is OK with that, others may not be. Tom does not support compilation error, 3) not clear.

Tom does not agree with going in the direction of a compile time error if the feature is not implemented.

Is there an implementation? No. At least not one that can be talked about.
Rajan likes the simplicity of the proposal, speaking as a programmer. Douglas expressed support as well.
Continue tomorrow.

[WED]

SC22.WG14.12062, Blaine Garth (excerpted)
One revision:

1) __atomic should apply to all objects, and not just the ones the compiler writer decides upon. This requires a strategy for atomic assigns of larger-than-word sized structs. This is not hard, I can and did sketch a commonly used algorithm for dynamically grabbing a lock, but dedicating some extra storage is another option, so there are several acceptable resolutions to make this reasonable.

There are two serious issues to resolve:

1) should += (et al.) have the strongest sequentially-ordered semantic (matching C++ atomic template use), or is a weaker ordering acceptable, as I have indicated in the proposal.

I'll speak with Hans about this since he has strong opinions across years of wider experience than me. I feel pretty strongly, though, that the operations should map to the hardware and not add expensive yet rarely used value. For obvious reasons I'm going to worry about ARM as much as

Intel, while fondly remembering PPC as well - I have stories to tell about G5 instruction execution traces...

2) need to further specify structure behavior, in particular field stores. E.g. What does an inner __atomic mean to a structure declared with __atomic?

With some thought, I think that

a) operations on the whole structure are obviously atomic, as in assignment. Initialization of auto storage does not need to be atomic since it can't be shared, but that might be an optimization instead of a specification.
b) operations on field members are probably not, unless they are also specifically marked __atomic. This lets initialization happen on a per-field basis.

I hope to invite the interested-all to participate, not sure what forum. I'll make that my first item tomorrow, because I think the discussion on __atomic was all but 30 seconds from wrapping up when the hook came out :-) Feel free to sign yourself up right now if you haven't already been talking on this topic in committee.

Blaine
---- end excerpted SC22.WG14.12062 ----------

[Wed]
Q to Blaine from JB. What is it we've agreed to? Blaine covered his view of the technical aspects. JB: We want to get closer to C++ compatibility. Blaine: We are moving forward in that direction.

5.21 What's New in Objective C (Slides on Wiki: "GC Florence.pdf") (Garst)

Focus for this discussion was the evolution of Garbage Collection (GC) in Objective C at Apple. Covered GC write barrier, thread local collection, death by alloc chart, death by alloc after improving malloc, then more improvement TLC.

ACTION: Blaine to obtain Apple release for these slides to make them publically available for distribution.

JB suggested this material might be more suitable as a Type 2 TR, Blaine does not see it going into the current revision, but seemed not sure about the TR route.

6. OTHER BUSINESS
None

7. RESOLUTIONS

6.1 Review of Decisions Reached

No decisions were made at this session.

6.2 Review of Action Items

Carry Over

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

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

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

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

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

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

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

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

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

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

ACTION: Blaine Garth to obtain a release of the slides presented, GCFlorence.pdf, from Apple.

7. THANKS TO HOST

The Committee expresses its great appreciation to Alessandro Pinzuti and Professor Enrico Vacario for hosting this meeting in Florence, Italy.

Thanks also to Dinkumware for providing the Wiki.
Thanks to Intel for providing the bridge, and to John Benito for providing the Skype connection to make this meeting happen.

8. ADJOURNMENT

Meeting adjourned at 1800, 21 April, 2010.

PL22.11 Meeting
19-21 April 2010

Meeting convened at 1500, 19 April, 2010 by PL22.11 Chair, David Keaton.

Attendees:

Voting Members:		
Name:	Organization: P – Primary, A - Alternate	Comments
Blaine Garst	Apple - P	
John Benito	Blue Pilot - P	
David Keaton	CMU/SEI/CERT - P	PL22.11 Chair
Tana L. Plauger	Dinkumware, Ltd – A	
P. J. Plauger	Dinkumware, Ltd – P	
Jim Thomas	HP – P	
Rajan Bhakta	IBM - P	
Michael Wong	IBM – P	
Paul Mc Kenney	IBM – A	
Clark Nelson	Intel – A	
John Parks	Intel – P	
Nat Hillary	LDRA – P	
Herb Sutter	Microsoft – P	
Douglas Walls	Oracle - P	PL22.11 IR
Barry Hedquist	Perennial – P	PL22.11 Secretary
Tom Plum	Plum Hall – P	
Fred Tydeman	Tydeman Consulting – P	PL22.11 Vice Chair
Prospective Members		
Larry Wagoner	NSA	
Larry Jones	Siemens PLM Software	WG 14 Project Editor

Agenda Approved as modified.

1. INCITS Anti-Trust Guidelines

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

2. INCITS official designated member/alternate information.

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

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

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

3.1 PL22.11 Members Attaining Voting Rights at this Meeting

Nat Hillary, LDRA Technology

3.2 Prospective PL22.11 Attending Their First Meeting

Larry Wagoner, National Security Agency

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

Apple, HP, CERT

5. JTC1 SC22/WG14 Sessions

6. Adjournment

Adjourned at 1800 local, 21 April, 2010