WG14/N477
X3J11/95-078
16 November 1995

Unapproved Draft WG14/X3J11 Meeting Minutes
16-20 October 1995

Clarion Somerset Hotel
2 Somerset Parkway
Nashua, New Hampshire 03063
USA

Legend:

The following symbols in the left margin of these minutes have the indicated meaning:

A	General approval
SV	Straw vote
FVP	X3J11-only formal vote that passed
FVF	X3J11-only formal vote that failed to pass
WGV	WG14-only consensus vote
***	Action item

Formal votes are reported as:

In-favor/Opposed/Abstaining/Absent/Total-eligible

1 Opening Activities

1.1 Opening Comments

Plauger convened the co-located WG14/X3J11 meeting at 9 AM, 16 October 1995, and immediately turned the meeting over to Jaeschke, chair of X3J11.

Jaeschke stated that the goals of the meeting were to process defect reports and technical proposals for C9x.

1.2 Introduction of Participants

Attendees introduced themselves. Attending the meeting were: John Benito, Jutta Degener, Frank Farance, Doug Gwyn, Rex Jaeschke, Larry Jones, David Keaton, John Kwan, Tom Lynch, Tom MacDonald, Neil Martin, Randy Meyers, Dave Mooney, P. J. Plauger, Tom Plum, Jim Thomas, Ted Van Sickle, Douglas Walls, Jonathan Ziebell.

Throughout the week, various observers, presenters, or alternates

7

attended the meeting: Byan Higgs, Scott Jameson, Bob Jervis, Andy Johnson, Ed Johnston, Alan Martin, William McKeeman, John Parks, Craig Schaffert, Jeff Zeeb.

1.3 Selection of Meeting Chair

A Jaeschke was selected as the meeting chair.

1.4 Host Facilities/local information

Meyers (Digital Equipment Corporation) was the meeting host. He distributed a map of restaurants within walking distance and stated that a PC running Windows and laser printer would be made available. He also announced that after the meeting session on Tuesday that Bill McKeeman would present his research on Dynamic Differential Testing.

1.5 Procedures for this Meeting

Meyers was secretary.

1.6 Approval of Previous Minutes [N434]

Plum requested that Section 27 of the minutes be dropped as the section reported an informal discussion that occurred after the meeting was adjourned, and the section misquoted Plum regarding BSI validation and ISO 9000 certification.

Plum also proposed that future minutes be less detailed than in the past: minimal minutes are less likely to be a source of controversy.

MacDonald/Plauger: The discussions captured in the minutes provide useful rationale for committee decisions, and have paid off in the past.

There was a brief discussion on whether there should be two sets of minutes: the official minutes that reported only the required information, and a "trip report" from the secretary that reported the discussions of committee.

Gwyn/Farance: The discussions in the minutes are not the problem. We should have faith in the process.

Martin requested that the draft minutes clearly indicate that they have not yet been approved by the committee.

Meyers summarized his sense of the committee: future minutes should continue to include discussions, indicate that they are only a draft, and should not report on any occurrences outside of committee meetings.

8

The minutes from the Copenhagen meetings were accepted with the following corrections:

Page 2, Section 1.2, Degener's affiliation should be DIN.

Page 7, Section 4.1, Farance's deadline for being able to translate the standard into HTML is October 1996.

Page 17, Section 16.1, Add the following straw vote immediately after the straw vote in the minutes: "In favor of adding some sort of complex arithmetic to C9x? 8 Yes. 0 No. 8 Abstain."

Page 27, Section 27, Delete this section.

1.7 Review of Action Items and Resolutions

Simonsen created a home page for WG14. The URL is http://www.dkuug.dk/JTC1/SC22/WG14

The following action items from the previous minutes were determined to be pending:

*** Gwyn will draft a proposal for deprecating implicit int.

*** Benito will investigate new minimum translation limits for a 500k conceptual machine.

*** Simonsen will produce a proposal to harmonize C and POSIX.

*** Keizer, Farance, Plauger will review Simonsen's C and POSIX proposal.

*** Simonsen will write a proposal for culturally correct uppercase and lowercase string conversion functions. Farance will review.

*** Mooney will produce an updated proposal on __FUNC__.

1.8 Approval of Agenda [N476]

After minor adjustment, the agenda was approved. The revised agenda is attached to these minutes.

1.9 Distribution of New Documents

New documents were assigned WG14/X3J11 numbers and will appear in the next mailing.

9

1.10 Information on Next Meeting

Ziebell announced that Unisys will host the next meeting during 5-9 February 1996 in Irvine, CA. The details are in N482 (distributed during the Nashua meeting and in the post-Nashua mailing).

The Orange County John Wayne Airport is the closest airport to the meeting.

*** Ziebell and Benito will issue the invitation through ANSI for WG14 to attend the meeting.

1.11 Identification of National Bodies/J11 voting members

After the attendance list was circulated, MacDonald announced 15 out of 17 eligible voting members of X3J11 were present, and that constituted a quorum.

A copy of the attendance sheet of X3J11 members is attached to these minutes.

WG14 members in attendance were: P. J. Plauger (WG14 convenor), John Benito (United States), Jutta Degener (Germany, DIN), Neil Martin (United Kingdom, BSI).

2 Reports on Liaison Activities

2.1 X3J11

Jaeschke reported that the letter ballot on the Numerical C Extensions Technical Report passed and that the announcement of the publication of the tech report should be made soon.

X3 approved the IR project.

We have been requested to provide a vocabulary representative. Jaeschke called for volunteers.

2.2 WG14 and SC22

Plauger and Plum attended the SC22 Plenary Meeting in Annapolis. They were happy to report no problems.

The steps necessary to register the C9x draft will follow.

Plum reported SC22 was very sensible on the issues of Web sites and electronic document distribution, and was critical of the JTC1 electronic document rules. Within SC22, ASCII sent by email or postscript on restricted FTP sites could be used to distribute documents instead of paper. Additional formats could be used in conjunction with these two.

, D

Farance noted the planned display and text forms of the C9x draft meet these requirements.

Plauger reported the Germans and Dutch raised concerns about C and C++ drifting apart. Our charter and previous resolutions diffused tensions on this subject.

Farance has been approved as project editor.

2.3 WG21/X3J16 (C++)

Plum reported that there were many "No" votes on the CD ballot to approve C++ as a DIS. Even before the end of the balloting period, WG21/X3J16 acknowledged that additional work was needed, and had resolved to have a second CD ballot in about a year. Due to a change in JTC1 procedure, the text of a DIS can not be changed. This meant that the C++ draft needed to be in a more mature form than originally planned when the standardization schedule was first drawn up.

There is now a concept of "Final CD", which is similar to the old DIS. The document is believed to be in a stable form, but could still be changed.

Plum/Plauger: The C++ committee seems to be putting more emphasis on no more new inventions in order to stabilize the document.

Plum: Banning of implicit int went though. There is a controversy on whether "const n = 20;" should be allowed. The C++ CD allows this since const and volatile are type specifiers in C++.

SV As long as C++ has banned implicit int, we feel it is appropriate for C++ to disallow const and volatile from appearing as the only type specifiers when a type is needed.
16 Yes. 0 No. 1 Abstain.

Plauger: BSI complained about the C++ draft changing during the balloting period. Voting periods can constrain what business a committee can address (perhaps even whether the committee can meet). It is important to follow proper procedures.

Plum stated overloading in C++ raises several issues about bool freely converting to int. He was not sure if these issues apply to C.

Gwyn asked if the problem could be avoided by disallowing bool from converting to an int argument.

Plum was not sure, and thought that would be an extreme rule.

Farance asked if X3J11/WG14's comments on the C++ CD are being listened to.

Plum stated that everything editorial that could be changed was changed. A group met before the Monterey C++ meeting and either closed an issue or put the issue on a C++ subgroup's issue list.

Farance complained that it was hard not getting any feedback after six months.

Gwyn pointed out that it took X3J11 a year in some cases to reply to public comments made during the public review period.

Plauger stated that this experience should give us more sympathy for the people who submitted public comments on C back in 1987. C++ is a bigger language with more fuzz.

Benito stated that C++ had a lot of work left, and that there was no rule that two CD ballots would be enough.

2.4 WG15 (POSIX)

Jaeschke has been notifying people about the C9x revision.

2.5 WG20 (Internationalization)

Plum complained to John Hill that WG20 has been making decisions that impact C without informing us. Now Plum is on the WG20 mail reflector.

Plum reported that WG20 is working on an ambitious plan for one collating algorithm for the ISO 10646 character set. There already exists a three pass collating algorithm for English and French that is a Canadian Standard. strxfrm() might be able to implement such an algorithm if control-A was used as a separator between the collation sequences generated during the different passes of the algorithm.

Plauger thought that there should be a commitment from WG20 that their collating algorithm be implementable using strxfrm(). So far, WG20 has only said that they think it should be possible to do a strxfrm() implementation. C, C++, and POSIX need strxfrm() to work.

SV Require that any universal collating algorithm provide a commercially reasonable implementation using strxfrm() and wcsxfrm().
17 Yes. 0 No. 0 Abstain.

Farance discussed extending identifiers, string literals, and character constants to contain characters from extended character sets, which might include characters not even in the source character set. One proposal is to encode such characters using \U followed by a "name" for the character. ISO 10646 includes names and encodings for all characters.

Admittedly, such a representation is ugly: it is trigraphs with a different prefix. However, such representations are useful for email and interchange of source.

Extended identifiers raise linker issues.

Plum reported that SC22 has made some recommendations for programming language to follow. The hex encoding of a character in ISO 10646 should be used as the character's "name." Programming languages should treat each simple character in ISO 10646 as a separate character and need not support combining characters or overstrike characters as a single character. WG20 has proposed a list of identifier characters from 10646: the list is about a dozen ranges of hex encodings.

2.6 Other Liaison Activities

Farance will coordinate reviews of other proposed standards. He asked how long should he give people to review documents. Jaeschke recommended ten days, but would check to see if there is a more appropriate deadline.

Farance complained that the Language Independent Procedure Call review period was only four days.

*** Jaeschke will investigate the short review periods and how to get copies of documents.

Farance pointed out that reviewers sometimes needed related documents in order conduct a review. Martin can provide some such documents.

Farance stated that Language Independent Datatypes Standard causes problems for C. He will follow up, but is not an official liaison.

*** Meyers will talk to Craig Schaffert about getting copies of the Language Independent standards for the mailing.

Farance and Keaton are following the Information Infrastructure Task Force (IITF).

MacDonald reviewed a document on FORTRAN calling C. This document does not place any additional requirements on C implementations.

13

3 Redactor Reports

3.1 Rationale (Benito)

Benito has converted the Rationale to use the section numbers from the ISO C Standard, and can produce the Rationale in a variety of formats. The Rationale is now a WordPerfect document.

Meyers asked if the long term goal was for the Rationale to be an SGML document. Benito said yes.

*** Jaeschke will put an item on the agenda to adopt the new Rationale.

Benito requested that people review the Rationale when it appears in an upcoming mailing and that they provide him with comments, including any bugs that they know of in the original Rationale.

3.2 Standard [N457] (Farance)

Farance reported that he is using the DSSSL doc style format in SGML.

Draft 3 included ISO Amendment 1 and TC1. Draft 4 includes TC2 and paragraph numbers.

Plauger reported that expects TC2 to be approved within the next few weeks.

Gwyn stated that Farance should have the license to use editorial judgment to produce a consistent document. (There was general agreement with this position.)

*** Benito, Keaton, Martin, and Walls will review the next C9x draft.

*** Farance will update the C9x draft and change to a 14 point font.

4 Defect Reports [N441, N455]

4.1 N441 DR142

Walls lead the discussion, and reported that the response to this DR had been approved in Copenhagen.

This lead to a discussion of how DRs should be handled as the committee shifts its primary focus to the development of the new C9x Standard. Plauger pointed out that although we have a statutory obligation to respond to a national body's DR, we are not obligated to produce a TC/RR immediately.

There was general agreement that when answering a DR that exact wording should be drawn up and any required changes to the

11

Standard should be communicated to the project editor (Farance). A formal TC/RR would be issued at a more leisurely pace.

Gwyn and others requested that the corrigendum to Subclause 7.1.3 also add a footnote that said that the reserved macro names also include the reserved function names because of the liberty given in Subclause 7.1.7 to provide macro implementations of functions.

4.2 N441 DR143

Walls lead the discussion, and reported that the response to this DR had been approved in Copenhagen.

Several people asked, "When does the project editor actually incorporate changes that result from a DR?"

Farance suggested, "When the committee asks him to do so."

Plauger proposed that the committee make a resolution at the end of each meeting week to instructing the project editor to make the changes.

Gwyn asked that each answer clearly state whether it is a edit to the existing Standard, an edit to the C9x draft, or just a response.

4.3 N441 DR144

Walls lead the discussion. The response contains the redundant phrase "a directive consists of" that should be dropped. The response should state that it is an edit to Subclause 6.8 in the C9x draft.

4.4 N441 DR145

Jones lead the discussion, and stated that the wording in the Standard was a bit sloppy here, but not so bad that a corrigendum was required. The wording should be improved for C9x.

&x[5] - &x[2] is not a constant.

4.5 N441 DR146

Jones lead the discussion, and reported that the DR is correct in pointing out that a TC has made the constraint in Subclause 6.1.2 redundant, and Jones recommends dropping the constraint from C9x.

Several people expressed concern that the Standard should be clear as well as complete. Challenges to the Standard can be headed off when there is a clear statement of a principle in the standard, even when the statement is just a consequence of other rules in the Standard.

15

Jones pointed out that the text of the TC occurs in the same Subclause, and so the constraint does not aid in understanding the Standard.

The constraint will be dropped from C9x.

4.6 N441 DR146

Jones lead the discussion. The committee reaffirmed its position that C standard library functions contain a sequence point even if not written in C. Jones will provide words for C9x.

4.7 N388 UK Defect Reports

Jones lead the discussion. Defect reports UK 14 through UK 16 should be fixed in C9x. No change is required for defect report UK 17: the current Standard is clear as is.

4.8 N441 DR148

Gwyn lead the discussion. The committee adopted the response in N441 after changing "reader" to "programmer" and dropping the reference to DR 142.

4.9 N441 DR149

Gwyn lead the discussion. The committee adopted the response in N441.

4.10 N441 DR150

Degener lead the discussion. The committee adopted the response in N441.

4.11 N441 DR151

Degener lead the discussion. The committee adopted the response in N441.

4.12 N441 DR152

Degener lead the discussion of what can be done from a signal handler triggered asynchronously (not by calling abort or raise).

Jones recalled that this question was investigated extensively when the original C Standard was developed, and the answer was "almost nothing." Supporting library calls from asynchronous signal handlers requires that libraries be asynchronously reentrant at great potential cost. For example, unless all I/O operations in a library are atomic, you can not even call exit() from an asynchronous signal handler. exit() flushes buffers and closes files.

16

[Consider if fclose() was interrupted. It might have closed the file but not yet marked the FILE * as closed. exit() would attempt I/O to a closed FILE *.]

Jones stated there is no contradiction when Subclause 7.7.1.1 says it is undefined for asynchronous signal handlers to call anything but signal() and Subclause 7.6.2.1 says signal handlers call call longjmp(): It is undefined for asynchronous handlers to call longjmp(); it is defined for synchronous handlers to call longjmp().

The committee decided the Standard is clear as it is.

5 LIA presentation (Schaffert)

Craig Schaffert presented an overview of the Language Independent Standards being developed by WG11, which are:

1. LIA, Language Independent Arithmetic
2. LID, Language Independent Datatypes
3. LIPC, Language Independent Procedure Calls

LIA is divided into three parts:

1. LIA-1, integer and floating point arithmetic, approved as an IS.
2. LIA-2, Elementary numerical functions, up for CD registration
3. LIA-2, Complex arithmetic, just starting

LIA is primarily a documentation standard. Conformance consists of selecting from one of its defined alternatives and providing an indication which implementation alternative was chosen. (For example, providing a header file similar to float.h with macros describing the implementation choices.) Schaffert's slides, which are included in this mailing, contained several proposals for C. Items marked with a dagger are items from LIA missing in Standard C.

Schaffert pointed out that LIA-1 tried to be friendly to C, and, based on feedback from the C community, added quietly wrapping signed integer arithmetic as one of the LIA-1 implementation alternatives.

Schaffert strongly encouraged more participation from the C community in the Language Independent Standards. Ideally, C experts should attend meetings, but failing that, they should participate on the WG11 mail reflector:

17

```
sc22wg11-request@dkuug.dk   To get on list
sc22wg11@dkuug.dk           To send a message
```

*** Farance will follow up on Language Independent Standards.

Plauger stated that SC22 does not mandate that language committees bind to the Language Independent Standards, but strongly encourages this. It is fair for us to honor the cross language standards.

There was a discussion of how LIA support should appear in C9x. Plauger expressed the opinion that at a minimum it should appear in an Annex.

6 Defect Reports

6.1 N439 DR139

Kwan lead the discussion of struct, union, and enum type compatibility across modules, particularly when the type in one module is an incomplete type.

MacDonald mentioned the importance to C9x of using the name of the tag when determining compatibility. Type compatibility and the aliasing information it provides is important to highly optimizing compilers.

Meyers pointed out that in Subclause 6.3 the committee wrote aliasing rules to support optimizing C. The cross module issues recently raised by DRs may have uncovered a hole:

Module a.c:

```
#include <stdio.h>
struct INCOMPLETE *p
extern struct INCOMPLETE *get(void);
extern int process(struct INCOMPLETE *p1, struct INCOMPLETE *p2);
int main()
{
    p = get();
    printf("%d\n", process(p, p));
    return 0;
}
```

Module b.c:

```
struct COMP1 {int x;};
struct COMP2 {int x;};

struct COMP1 *get(void)
{
    static struct COMP1 comp1;
```

18

```
        return &comp1;
    }

    int process(struct COMP1 *p1, struct COMP2 *p2)
    {
        p1->x = 4;
        p2->x = 5;      /* Can't change what p1 points to? */
        return p1->x;   /* Compiler free to return 4? */
    }
```

When analyzing process() for aliasing, the compiler could determine that the assignment to p2->x could not change the value of p1->x because the type pointed to by p1 is not compatible with the type pointed to by p2. Thus, the compiler could change the return statement to return the constant 4 since that was the last value stored in p1->x. However, one interpretation of the response to DR139 would be the compiler has to assume that a third incomplete type in another module might cause *p1 to alias *p2. This would be a disastrous blow to optimizing C.

The committee discussed the issues of making the tag name play a part in type compatibility for the cases of struct, union, or enum types without tags or who are only named via a typedef. Meyers pointed out that a popular coding style avoids tag names and uses typedef names for all struct types. Plum stated that the C++ committee's solution to this problem was to consider the typedef name to be the tag name for the purposes of type compatibility.

SV It is in the best interests of implementors and users that the tag name be important in determining type compatibility in C9x.
13 Yes. 1 No. 2 Abstain.

Jones pointed out that the response to DR 139 does not make Meyers's possible aliasing hole any wider: you can write the same example using a complete struct type in module a.c.

The committee decided to let the response to DR139 in N439 stand and to pursue name compatibility in C9x.

6.2 N439 DR73

Farance pointed out that DR73 in N439 and DR51 have conflicting answers. DR51 says it strictly conforming to access a partial object if the accesses are limited to the part that exists. DR73 says it is undefined behavior to access any part of a partial object.

[The issue concerns the best way to program a data structure that contains a variable sized array as its last member. Should the array be declared with only one element (zero if the extension is supported), or should the array be declared with a very large

19

number of elements of which only a few are actually allocated?]

Gwyn agreed that the two DRs were addressing the same case.

MacDonald stated that DR73 was wrong.

Gwyn liked the answer to DR73 and thought that DR51 was a kludge.

Farance agreed that DR73 was undefined behavior, but was a common extension.

Plum suggested that the committee move slowly in interpreting the existing Standard in this area. This programming technique either works or does not work in existing implementations. DR51 is an example of what the committee thinks is a good working solution [declare the array with too many elements], which is a better solution than a zero sized tail.

Someone suggested that perhaps it was too draconian to outlaw zero size arrays in the first place.

Keaton suggested that no DR be produced and that the issue be resolved in C9x.

Jones pointed out we have two existing DRs with different answers.

SV Should we bring DR73 forward?
8 Yes. 4 No.

SV Who is in favor of the following as the answer to DR73:
7 Undefined behavior. 2 Strictly conforming. 6 Don't know.

6.3 N455 NNI-1 DR153

Gwyn lead the discussion and reported that DR3 answered this question. No constraint is violated and no diagnostic is required when a function like macro of one argument is called with an empty argument list. However, the behavior is undefined.

6.4 N388 DR UK 14 through 17

Jones lead the discussion.

DR UK 14 on page 7 correctly points out that for an expression to really be an lvalue it must designate a real object. However, it is frequently impossible to prove at compile time that an lvalue expression is designating a real object, and yet this determination must be made to check the constraints on some operators. Jones said that this is a wording problem that should be fixed in C9x. Perhaps using a phrase like "has the syntactic form of an lvalue" should be used in the Standard when describing

20

the constraints on operands of operators like ++.

DR UK 15 and DR UK 16 point out edits in TC1 that should also be applied in other sections of the Standard. These edits should be done in C9x.

DR UK 17 complained that the statement in Subclause 5.2.1.1 that says "Each ? that does not begin one of the trigraph listed above is not changed." conflicts with the fact that the second "?" in a trigraph is replaced when the trigraph is recognized.

Gwyn said he did not think many people would be confused by the wording in the Standard.

SV Who is in favor of making some sort of wording change to fix this apparent contradiction?
3 Yes. 10 No.

6.5 N388 DR UK 20

Gwyn lead the discussion.

DR UK 20 Part 1 alleges that Subclause 6.3.8 and Subclause 6.2.2.3 contradict each other when comparing a null pointer. Subclause 6.3.8 says that it is undefined to compare two pointers unless they are members of the same aggregate or union. Subclause 6.2.2.3 says a null pointer always compares unequal to a pointer to object or function. [Note DR UK 20 erroneously references Subclause 6.2.2.1 instead of 6.2.2.3.]

There is no contradiction since 6.2.2.1 discusses the equality operators and not the relational operators in 6.3.8.

Plum stated that C++ users want relational operators on null pointers. Does this cause problems on some machines (traps, slow execution)?

Plauger stated that inline code for stream buffers in some C++ implementations can be speeded up if a relational on a possibly null pointer is used. In particular, you want NULL < NULL to be false. This is a hot spot in the library worth optimizing.

Mooney stated he was not sure if relationals on null pointers was a problem on his machines.

Gwyn hypothesized that some tagged architectures might pay a penalty to compare a null pointer.

Jones wondered if a special case for NULL < NULL is all that is needed.

As for DR UK 20 Part 2, Plauger said this is an example of where

21

the Standard should be saying "only if" instead of "if". A pass through the C9x draft should be made looking for places "only if" should be used.

7 Revision of ISO/IEC 9899:1990 (Jaeschke)

Jaeschke asked what rules should be followed in forming the US position for WG14.

Meyers, Plum, and Benito suggested that since WG14 works by consensus, the US position should be determined a simple majority of X3J11.

SV In favor of the informal US position being determined by a simple majority of X3J11 members voting?
9 Yes. 4 No. 1 Abstain.

FVP (Meyers/Farance) Move that the informal US position be determined by a simple majority of X3J11 members voting.
11/3/0/3/17

Jaeschke said that the committee needed rules to provide clear guidance on whether it was interested in a proposal. He suggested that people avoid abstaining on votes to give committee agenda time.

Farance suggested that subgroups be organized to integrate the separate proposals and insure common solutions. One of the problems that some proposals struggle with is they do not know if they can use features from other proposals, for example, overloading and the math library.

There was a discussion of what such subdivisions might be made, whether subgroups were a good idea, and whether integration should be attempted now or later.

Jaeschke suggested that at least one-half of the people in the room have to vote in favor of giving a topic agenda time before the topic gets on the agenda for the next meeting.

Thomas stated that a vote for agenda time should be taken at the logical end of the session on a proposal. If there are multiple alternative proposals, all should be heard before a vote. Likewise, if one proposal is split across multiple sessions on the agenda, the vote should occur after the last session.

SV At the logical end of a session for a proposal before the committee, a vote will be taken to give the proposal agenda time at the next meeting. At least one-half of the members in the room must vote "Yes" in order for the topic to be given agenda time. [This vote was retaken later in the week.]
17 Yes. 2 No.

22

SV Should we at this time create subgroups to consolidate proposals?
3 Yes. 8 No. 6 Abstain.

8 Electronic Distribution Policy [N454] (Keaton)

Keaton lead the discussion of N454.

A spirited discussion broke out on the question of making the SGML source of the Standard available to committee members in addition to the derived HTML and text formats.

Plauger stated that ISO claims a copyright on the DIS and IS, but not on earlier forms of the standard. ANSI only claims a copyright on the final standard. The status of earlier drafts of the standard is muddled. We have an obligation to protect the eventual property of ANSI and ISO. Plauger reported that X3J16/WG21 posted the HTML and text versions of the C++ CD during its public review period, but the C++ project editor has very carefully restricted distribution of the troff source.

The committee discussed whether a password guarded FTP site is secure enough to protect the SGML source.

Plum pointed out that JTC1 has adopted sending diskettes in the mail as a form of electronic distribution. This is not totally secure as floppies can be stolen. If our security is as good as floppies in the mail, then we fulfill the obligation to secure the source.

Gwyn stated that passwords provide a good enough level of security.

SV Shall the SGML source to the Standard be made available on the secure FTP site?
8 Yes. 7 No. 3 Abstain.

Since there was no clear direction, Plauger exercised his rights as convenor: The SGML source will be made available only by request of national bodies. Farance will keep a list of who has received the SGML.

SV Shall the derived forms (e.g., HTML, text) of the Standard be made available on the secure FTP site?
18 Yes. 0 No. 0 Abstain.

There was general agreement that draft minutes should not be made publicly available.

Plauger stated that he is responsible for telling SC22 who maintains the WG14 pubic web site. SC22 prohibits publishing on the site commercial messages or membership lists (to protect privacy), but encourages making available agendas, meeting

20

schedules, and information on published standards.

SV Shall proposed changes to the C language and author-authorized documents be available on the unrestricted FTP site?
16 Yes. 0 No. 0 Abstain.

9 Designated Initializers [N472, N473, N474] (Keaton)

9.1 Designated Initializers

Keaton presented his proposal on designated initializers N472 and N473.

SV Shall designated initializers as described in N472 be added to C9x?
15 Yes. 0 No. 1 Abstain.

FVP (Keaton/Meyers) Move that designated initializers as described in N472 be added to C9x.
15/0/0/2/17

WGV Shall designated initializers as described in N472 be added to C9x? 3 Yes. 0 No.

Meyers pointed out this was the first change to C9x.

Walls requested that page 2, line 19 be removed. Keaton agreed.

*** Keaton will supply Benito with rationale for designated initializers.

*** Walls, Mooney, Gwyn will be the editorial review board for designated initializers.

*** Keaton will ask Thompson if he wishes to be named as prior art for designated initializers.

9.2 Initializer Repetition Counts

Keaton presented his proposal N474.

Lynch asked why "implied for loops" were not proposed instead. Keaton replied that loops raise more language issues: what is the scope of the index, what is its value after the loop, ...

Jaeschke requested that the examples show what got initialized and with what values.

There was a brief discussion of making the feature more programmable, but there was a fear that led to a compile-time subset of C.

Farance stated he was in favor of the semantics, but not the syntax.

Jutta stated that she objected to the entire feature as it was not powerful enough.

Mooney stated that the syntax was hard to understand, hard to teach, hard to implement.

Plauger stated that Whitesmiths C has a similar feature, and that embedded programmers loved it.

Kwan asked about interaction with partially elided braces. Keaton said that it works just like designated initializers. Kwan asked for examples.

SV In favor of agenda time at the next meeting for initializer repetition counts?
14 Yes. 2 No. 2 Abstain.

10 Compound Literals [N475] (Keaton)

Keaton presented his proposal N475. The following example generated much discussion:

```
void example()
{
    int i;
    for (i = 0; i < 10; i++) {
        f( (struct FOO){.a = i, .b = 42} );
    }
}
```

when contrasted to the same code fragment with no {} around the call to f(). With the {}, the compound literal has scope of the loop body, and the "a" member of the compound literal takes on the successive value of "i". Without the {}, the scope of the compound literal is the function body. The value of "a" might be frozen on the first value of i when the compound literal is first executed, depending on the decisions made by the committee.

Gwyn and Farance expressed concern about the execution overhead associated with some of the possible semantics for compound literals.

Plum pointed out that compound literals were similar to temporaries in C++, and that temporaries in C++ only endure to the end of the expression. It is annoying that C temporaries have different lifetimes than C++ temporaries.

Keaton replied that if compound literals have temporary lifetime, then you rule out taking the address of a compound literal and

25

manipulating it over the course of several statements.

Farance thought pointers to compound literals could be useful.

Jones observed that compound literals were declarations hanging in space. C traditionally has not allowed declarations intermixed with statements.

Meyers agreed with Jones, and stated support for the temporary model for compound literals. The temporary model allows for the compound literal to take on the value of its initializer expressions every time the compound literal is executed, which is the natural definition. The temporary management in the compiler is similar to the temporaries managed by C compilers today for the return value of functions that return structs.

Farance said he could not tolerate the two examples acting differently depending upon whether the loop body was enclosed in braces.

Lynch said constructors are starting to look good in contrast.

Keaton pointed out that compound literals also support arrays, unlike constructors.

Plum stated that he did not like any of this, and that he must oppose any departure from C++ without sufficient justification. Inline functions could be used to create structures on the fly. The only case not handled by inline functions is arrays.

Degener stated there is an entire style of small struct programming that could benefit from compound literals. Expression lifetime is good enough, since if you are going to declare a pointer to point at a compound literal, you might as well declare a struct or array instead of using the pointer.

SV Should the compound literal in the above example take on the same sequence of values regardless of the presence or absence of braces around the call to f()?
16 Yes. 0 No.

SV Should the lifetime of a compound literal in a function body match that of an auto variable in the enclosing scope?
4 Yes. 5 No. 9 Abstain.

SV Should it be possible to take the address of a compound literal?
10 yes. 0 No. 7 Abstain.

SV In favor of agenda time at the next meeting for compound literals? [This vote was retaken later in the week]
8 Yes. 6 No. 4 Abstain.

26

11 Restricted Pointers [N448] (MacDonald)

MacDonald presented his proposal N448.

Plum observed that this is prior art in some C and C++ implementations.

MacDonald reported that when restrict was used in the Sandia Class Library (written in C) the performance exceeded the FORTRAN version of the library.

MacDonald stated that aliasing is rare in real programs. If a program stops working after the restrict keyword is added, a few techniques can help discover if aliasing caused the problem. The first step is to turn off the optimizer. If the problem goes away, try turning the optimizer back on and then #defining restrict away.

MacDonald went on to say that restrict is mostly used on parameters.

Walls reported restrict greatly speeds up some benchmarks. The feature is hard to document; examples are the best way to explain it.

MacDonald stated that a command line switch to add restrict to all pointer parameters is useful.

Degener believes that the assignment compatibility rules for type qualifiers should not apply to restrict.

*** Degener will write a paper on assignment compatibility rules for the restrict qualifier.

SV In favor of adding restricted pointers to C9x?
17 Yes. 0 No. 0 Abstain.

FVP (Walls/MacDonald) Move that restricted pointers as described by N448 be added to C9x.
15/0/0/2/17

WGV In favor of adding restricted pointers to C9x?
3 Yes. 0 No.

*** MacDonald will supply Benito with rationale for restricted pointers.

SV In favor of agenda time at the next meeting for Degener's paper on assignment compatibility rules for the restrict qualifier?
16 Yes. 0 No. 0 Abstain.

*** Keaton, Walls, MacDonald will be the editorial review board for

27

restrict.

12 Variable-Length arrays [N451] (MacDonald)

MacDonald presented his proposal N451. [VLA means Variable Length Array]

Jaeschke asked what is the composite type of two VLAs? MacDonald answered the composite type is VLA since the size is not part of the type of a VLA at compile time. [N451 Section 6.1.2.6]

Jaeschke asked what is the scale factor with doing pointer arithmetic on a pointer to a VLA. MacDonald answered the runtime size of the type.

Degener asked about sequence points and VLA declarations. MacDonald replied there was a sequence point after the declarator. Degener requested this information be added to Subclause 6.3 (page 40 of C9x draft 4).

Martin requested that the comment /* Size Fixed Here */ be added to N451, Page 5, Example 3, Line 10.

Thomas asked if lexical scoping turned out OK. MacDonald replied that Example 14 on Page 11 shows the coding style that allows old FORTRAN parameter passing order to work with lexical scoping.

Degener stated opposition to N451 Section 6.5.6: typedefs should fix the size when the typedef is used, not when the typedef is declared.

Meyers worried about N451 Section 6.5.4.2: Whether a type is a VLA or a normal array is a big distinction in the type system, yet it depends on whether the array bounds is a constant expression or not. This is not a big distinction since most implementations extend constant expressions. Thus, int x[(int)(2.0 + 2.0)] might be a VLA on one implementation but not on another. All implementations extend integral constant expressions in some way since it is the only way to implement offsetof. The current C Standard was careful to allow extending integral constant expressions by defining them outside of a syntax or constraint section. In many compilers, it is hard to tell if an constant was a true constant or the result of evaluating an extended constant expression.

MacDonald will come up with wording that allows implementations leeway to recognize extended constant expressions as bounds of normal arrays.

Plum asked if some allowances could be made to allow VLAs to be passed by dope vector. This might allow some implementations to simplify their interlanguage calling mechanisms with FORTRAN 90.

28

SV In favor of adding variable-length arrays to C9x?
10 Yes. 4 No. 3 Abstain.

FVP (MacDonald/Van Sickle) Move that variable-length arrays as described in N451 be added to C9x.
11/4/0/2/17

Meyers explained his "No" vote: A dope vector scheme for VLAs is superior to the old FORTRAN scheme of passing the array bounds as separate parameters. However, Meyers complemented MacDonald for coming up with a solution that preserves lexical scoping.

WGV In favor of adding variable-length arrays to C9x?
2 Yes. 1 No. The vote failed consensus.

SV In favor of agenda time at the next meeting for variable-length arrays?
17 Yes. 0 No. 0 Abstain.

13 Inline Functions and Function Overload

Plum reported that he had led a lunch time discussion of inline functions and function overloading. Inline functions were supported by the lunch time discussion group; overloading was more controversial.

Plum said the discussion had included several alternative ideas for overloading: Only allow inline functions to be overloaded in order to avoid name mangling, allow external functions to be overloaded, the possibility of overloading rules that modeled the usual arithmetic rules for operators, and wild card overloads.

Several people expressed opposition to overloading as being very complex for both programmers and implementors. Meyers observed that C++ compilers still differ in how they do overload resolution, and different compilers decide to call different functions in user programs.

Several people expressed support for overloading. Thomas pointed out it was very useful for numeric programming.

SV Interested in seeing a proposal to add inline functions to C9x?
17 Yes. 1 No. 0 Abstain.

*** Plum will write a proposal on inline functions.

SV Interested in seeing a proposal to add function overloading to C9x?
7 Yes. 6 No. 4 Abstain.

*** Farance will write a paper on function overloading.

29

Thomas observed that the rule adopted earlier in the week requiring one-half of all votes to be "yes" in order to get agenda time did not apply to new proposals not yet seen by the committee. There was general agreement.

14 Classes in C [N424, N445, N446, N447] (Walls, Jervis)

Jervis presented his proposals. He explained that the proposal was in four parts at the request of the committee so that they could chose the features that they wanted. Part 1 is basic to any features and is a prerequisite for the other three proposals. Part 2 depends upon Part 1, and Part 3 depends upon Part 2. Part 4 only depends upon Part 1.

Jaeschke asked how to initialize const class objects with private members if there are no constructors? Jervis answered that you initialize them with a non-const version of the object.

There was a long discussion on constructors and destructors.

One issue was whether constructors and destructors should only be allowed on objects with particular storage classes. For example, not allowing constructors on file-level static objects avoids having to extend the linker.

Keaton was uncomfortable with the implicit nature of constructors and destructors given the traditional focus on explicit operations in C.

Gwyn asked if constructors and destructors should only be allowed on objects created with new and delete: it makes the operations explicit and under the control of the programmer. Higgs pointed out this this could lead too much use of dynamic storage in order to get the feature. Meyers stated that a traditional advantage of C++ over other object oriented languages is that C++ allows non-dynamic objects, and this shows up as a big performance advantage.

MacDonald raised the issue of longjmp() and destructors on autos.

Jervis and Meyers pointed out that implementations of longjmp() could unwind the stack and call destructors.

Plum pointed out in C++ you use exception handling to make sure that destructors are called when exiting nested function calls. In C++ it is undefined behavior to longjmp() past destructor calls: some implementations of C++ exception handling do a pass to call destructors and then do a non-unwinding longjmp() as a second pass. Those implementations require longjmp() not to honor destructors.

MacDonald expressed that it was important for code not using a

30

feature to have no additional overhead because the unused feature exists in the language. Legacy code should not slow down or get bigger because of new, but unused, features.

Meyers reported that the combination of destructors on autos and exception handling did carry additional overhead, since tables of information must be generated so that destructors can be called as the stack unwinds. However, code with no destructors has no additional overhead, and most implementations support a command line option that says to assume no exception handling so even code with destructors has no additional overhead.

Plauger explained the C++ style of programming with constructors and destructors: it allows the resources associated with an object to managed properly, and for the books to balance. Constructors and destructors on autos should be fully supported or not supported. There should be no half steps or undefined behavior.

Gwyn and Kwan also stated that undefined behavior should be avoided.

Jervis pointed out that the feature could be left out; that is why it is a separate proposal. The alternative is for programmers to add their own "open" and "close" functions to class types and call them explicitly to "initialize" and "finalize" objects.

Plum worried that the more C++ features appear in C9x, the louder the complaints may be from the C++ committee. We owe Jervis a great debt for producing separable proposals.

Degener observed that constructors and destructors allow programmers to extend the language, and gives programmers a great sense of power. Unfortunately, this seduces them into using lots of hidden machinery, which is bad.

Meyers agreed with Degener: C++ has a goal of allowing user defined data types to be first class citizens. As a consequence, there is language support for operator overloading, copy constructors, etc. However, most user defined data structures do not benefit from being first class citizens. In most of his programming, Meyers defines the assignment operator and copy constructor as private member functions, and never provides implementations. Thus, the compiler does not treat his classes as first class types. It is not useful to make a copy of a search table by assigning it to another search table variable.

Meyers also pointed out that the current C programming style with explicit management of resources can lose resources during a longjmp(), and the new variable length array proposal allows the space for the VLA to be lost during a longjmp().

31

Jaeschke asked if there was any interaction between classes and inline. Jervis replied that they were separable features. Meyers reported that some implementations use the definition of the first non-inline member function to determine the module that contains the definition of the the virtual function table. Inline can complicate this by allowing all member functions to be inlined.

Farance worried that classes were a slippery slope leading to an explosion of features, and that C9x was five years behind C++.

Jervis stated that the years of experience with C++ is our advantage: We can see the pitfalls. We use this subset at Sun.

Plauger pointed out the differences in approach taken by X3J11 and X3J16. J16 has a tendency to say that "if a feature is useful in some cases, it should be put in." J11 has a tendency to say "if you can live without a feature, then leave it out." The C committee is more concerned about standardizing existing practice. Choosing the right balance point is the risk. We do not have to get caught up in adding feature after feature.

Gwyn expressed some concern that the public perception might be that the C committee was just second guessing the C++ committee. However, C89 code does not appear to pay any penalty for these new features: this proposal does not cross the line.

Farance observed that people really do want features like this.

Jaeschke asked who is the audience for C9x? Presumably C will be more suitable for embedded systems than C++. But in the long term, C++ will probably pick up C9x features making the languages more similar.

MacDonald pointed out that C's traditional customer base would benefit greatly from the namespace control allowed by classes. [There was a suggestion that Jervis point this out in his rationale.]

Gwyn stated that systems implementors would benefit from a language with C's low runtime overhead and some features from C++.

Meyers reported that inside of DEC there was a great deal of interest in a simpler C++.

Meyers also pointed out that there is a style of C programming that tries to simulate inheritance using macros to define the bodies of "derived" structs (this was discussed on the mail reflector). This style of programming violates the aliasing constraints in Subclause 6.3. It would be better to give people inheritance to avoid optimization problems.

Farance stated that classes and inheritance were a good solution to interface and namespace problems. Compilers should do what they can to help the programmer.

Jervis reported that he had always thought of C as a systems implementation language. However, while preparing an article, he discovered that Ritchie had always defined C as a general purpose programming language. Currently, the SUN C and C++ compilers swap back and forth as to which is more popular. C++ features do not automatically cause overhead: the C subset of C++ is just as efficient as C. C is a different language from C++, but we owe our users C++ features when good. Classes fix namespace problems, but also help organize APIs. Virtual functions help make more extensible libraries and subsystems. The result is better checking, and the ability to extend the interface without needing the original source code.

Gwyn asked if we should add operator overloading? MacDonald replied that there is a point where we should tell people to use C++ instead.

MacDonald reported that migration to C++ has been redirected towards C inside of Cray because of the different dialects of C++.

Gwyn reported that most of the people he works with are not moving towards C++. But, he could see them moving to C9x. However, if they were PC programmers, they would be forced to C++ by windowing support.

SV In favor of some sort of class feature in C9x?
10 Yes. 0 No. 7 Abstain.

14.1 N424 Basic Classes

At the Copenhagen meeting, there was a discussion of minimizing the number of keywords added to C9x by not adding "public" and "class". Jaeschke expressed support for having both public and private keywords, as it allows the order of public and private members to be mixed. Also, the class keyword is easy to teach, and has the right default access.

Meyers pointed out it would be useful to have "public" and "class" to maximize the common subset of C and C++. This allows a fighting chance to have common C/C++ header files.

Jervis stated one of his goals was allow new features only in classes to avoid changing any promises about the layout of structs.

Plum stated that the C++ committee had defined the term "POD" for Plain Old Data. A POD is a struct, class, or union that has the

33

same layout predictability as a C89 struct or union. Classes in C++ can be PODs, if they meet the restrictions. POD Classes in C++ should have the same predictability in C9x.

MacDonald asked about offsetof and classes. Plum replied that in C++ offsetof only works on PODs.

Thomas asked if static members could be added. Jervis replied that static members were cheap to add. Meyers agreed that they were cheap, and pointed out that static members imply support for the :: scope operator.

Plum asked about tags as type names as in C++. Meyers pointed out that C++ had to partially put back in the tag namespace in order to support the popular coding style of "typedef struct stat stat;" and to resolve ambiguities in templates.

SV If classes are added to C9x, they should look as proposed in N424.
12 yes. 0 No. 5 Abstain.

Plum observed that different projects programming in C++ adopt different local style rules blessing different subsets of C++. Perhaps C9x could jell a common subset.

14.2 N445 Inheritance

Plauger stated that single inheritance was a lightweight feature with high payoff.

MacDonald asked for a rationale.

Jervis mentioned organizing APIs and supporting extendibility.

Plum enumerated the advantages of type safety, less casts, support for a common prefix between data structures.

Gwyn asked why public inheritance and private inheritance?

Meyers explained with public inheritance a derived class can be thought of as just a special case of its base class. Any function that can process the base class can also process the derived class, and programmers can take advantage of that fact. With private inheritance, it is only an implementation detail that the derived class is a kind of its base class. For example, it is only an implementation detail subject to change that class store_inventory is derived from class linked_list. By using private inheritance, users of a derived class can not take advantage of knowing what the base class is. This allows the base class to change.

This lead to a discussion of the "protected" access specifier

31

from C++. Jervis decided to add this keyword to the proposal.

SV If C9x contains basic classes as in N424, then inheritance as proposed in N445 should be added.
13 Yes. 0 No. 5 Abstain.

SV In favor of adding the "protected" keyword if inheritance is added to C9x?
6 Yes. 0 No. 10 Abstain.

14.3 N446 Virtual Functions

Plauger and Plum pointed out array new and delete should be added for compatibility with C++. Destructors imply that the delete[] operator be supported so that destructors can be called on all of the elements of an array.

Degener and Meyers asked for placement new and delete to support user controlled heap management.

Plauger pointed out that placement new and delete raise thorny issues with lifetimes and the object model that the C++ committee has not fully addressed yet.

Keaton worried about the additional overhead in virtual function calls. Meyers pointed out it was a programmer decision to use or not use virtual functions.

SV If classes (N424) and inheritance (N445) are added to C9x, then virtual functions as proposed in N446 should be added.
8 Yes. 1 No. 9 Abstain.

14.4 N447 Constructors and Destructors

There was a discussion of partially initializing a class object using a brace enclosed initializer list having the side effect of zeroing members not explicitly initialized. This could allow simple initializations of private and protected members.

There was a discussion of whether copy constructors were needed. Plauger pointed out that copy constructors were important if you wanted user defined types to be first class citizens. However, writing user defined types with first class status is error prone.

Plum observed that frequently constructors in C++ are overloaded. If you allow constructors to take multiple arguments, then you need for new to take multiple arguments also.

Jervis stated he would change the proposal not to require an empty set of parenthesis when no arguments were being passed to the constructor.

35

SV If C9x contains basic classes as in N424, then constructors and destructors as proposed in N447 should be added.
5 Yes. 4 No. 8 Abstain.

SV In favor of agenda time at the next meeting for classes?
17 Yes. 0 No. 1 Abstain.

There was a brief discussion on strategy for proceeding.

Plum expressed concern that the committee's work on classes not be over-represented, under-represented, or misrepresented to the public, as it could stir up bad publicity.

Mooney stated that he wanted basic classes (N424), inheritance (N445), and virtual functions (N446) together or nothing.

Plauger stated that adding classes needed to be done incrementally in a cohesive fashion. He called for high attendance at the meetings.

The next meeting will vote on at least basic classes, inheritance, and virtual functions.

15 TC2 and RR2 News (Plauger)

Plauger reported that TC2 has been approved.

On the voting on RR2, the UK complained that the responses were not complete, and the Netherlands pointed out that DR87 and DR117 give conflicting responses.

Plauger asked the committee what it wished to do about the conflict.

Gwyn suggested that the behavior be undefined as suggested by the Netherlands.

16 Complex Arithmetic

16.1 Complex Arithmetic [N450] (MacDonald)

MacDonald presented his proposal N450.

Meyers made some comment about freestanding implementations and complex, but the secretary failed to record it properly. The issue probably concerned whether the rules against using the complex keyword and complex constants before including <complex.h> was a sufficient guarantee that freestanding implementations need not provide complex.

16.2 Complex Arithmetic [N426, N470, N471] (Thomas)

Thomas presented his proposals and papers N426, N470, N471.

A discussion ensued of whether complex infinities were useful.

Lynch stated that the only applications he was aware of where signed zeros and infinities work well are those which use directed rounding for bounding values. The proposal under discussion did not include a requirement for directed rounding.

Lynch stated that in the expression (x + x) / x, if x + x overflows, a result of infinity was a sign of infinite relative error over the true result of 2.

Thomas asserted that infinities and signed zeros have proven their usefulness in IEEE arithmetic.

Gwyn stated that the right thing for complex infinities may not usually happen, but the proposed behavior could be useful in some cases.

Thomas said that naive code is more likely to work with the proposed behavior.

Lynch pointed out the advantages of interval arithmetic over IEEE and other conventional floating point formats.

MacDonald raised the issue of the imaginary type. The benefits are not worth its cost.

Plum stated that C9x has broad requirements for IEEE and non-IEEE floating point support. Perhaps imaginary should only be for IEEE systems?

Thomas and Farance stated pluses for the imaginary type: it can save storage and save operations at runtime since an operation on two imagery operands does not promote to complex.

Gwyn asked if there is one or many complex infinities in the proposal. Thomas replied there were multiple infinities, but with care you can program to the one infinity model.

Kwan asked if the proposal was compatible with FORTRAN.

Thomas replied that the differences in this proposal are improvements over FORTRAN.

Gwyn added that the differences were extensions: you could ignore the imaginary type for greater FORTRAN compatibility.

MacDonald stated that the Cray complex proposal was prior art.

It had received no complaints and had aided in several conversions from FORTRAN to C.

Thomas stated that his proposal had been implemented as a C++ class library.

Meyers suggested that perhaps LIA-3 was a better place to work out the model differences between the two proposals for complex.

Thomas stated that LIA is more of a documentation standard than a place to specify an implementation model. LIA-3 might define a macro to test if an imaginary type is supported, but LIA-3 would not settle model issues.

SV In favor of some kind of complex in C9x?
8 Yes. 1 No. 9 Abstain.

Several people explained their votes:

Jaeschke has no opinion.

Martin is concerned about the impact on embedded systems.

Walls believe that more experience was needed.

Lynch was concerned that complex was hard to get right and optimize, that interval arithmetic would be more useful, and that the justifications for the features was thin.

Mooney did not believe that there was enough interest among his customers for complex in C. Leave complex alone as part of the NCEG tech report until some demand shows up.

Keaton thought complex might attract new programmers.

Degener said that complex would not receive wide enough use to justify adding it to the language.

SV In favor of agenda time at the next meeting for complex?
13 Yes. 0 No. 5 Abstain.

17 Floating-point extensions [N466, N467, N468, N469] (Thomas)

Thomas presented his proposals.

Plum observed that the overload mechanism described in N467 Section 7.x.2 is a placeholder for a mechanism that has not been standardized yet.

Plum complained that hex floating point constants are proposed so that people can get exact constants, but we do not require decimal floating point constants to be exactly translated. This

38

means people must translate constants themselves into hex, instead of letting the compiler do the work.

Gwyn, Keaton, Meyers, and MacDonald worried about the complexity of exact translation of decimal floating point. Keaton raised the cross compiler issue.

Gwyn, Lynch, Plauger discussed arbitrary precision compile time constant expression evaluation. It was observed that most floating point constant evaluation could be deferred until program startup.

There was a suggestion that the proposed pragmas be allowed between statements as well as external declarations. Meyers suggested that the pragmas be allowed between any statements or declarations and have effect until canceled.

Thomas pointed out that under such a scheme there is no way to push and pop pragmas, and so it is hard to know what pragma state to restore after changing the setting for a section of code.

Meyers suggested the lack of a stack support in the pragmas was the problem, and such support should be added. DEC's C and C++ compilers have several such pragmas.

Plum and MacDonald suggested alternative forms of pragmas, such as in N449.

Thomas asked what should be the relationship between math.h and fp.h.

Meyers suggested that fp.h include math.h.

MacDonald suggested that including fp.h be a hook for dropping errno.

Gwyn suggested that it be undefined to include both fp.h include math.h, and that math.h eventually be dropped.

Degener wanted including both fp.h include math.h to work.

Plum pointed out there could be issues with the function overloading in fp.h. Currently, you can take the address of functions in math.h. This may not work if if the function is overloaded.

Degener found "shoulds" in the proposal annoying, and proposed the text be rewritten to say "highly recommended."

Lynch stated that some of the "shoulds" should be "musts".

Mooney stated that "should" is Ok when directed towards the

39

implementor, but not when directed towards the user.

Gwyn said that there was a recommended practice for writing standards. "Should" is for recommendation; "shall" is for requirements.

There was a discussion of how IEEE floating point specific requirements could be incorporated.

Plauger reported that SC22 permits conditional standards. An Annex can be either informative or normative. The Standard could say that an Annex is normative if, for example, an implementation set some macro in limits.h to 1. An implementation that did not define the macro to 1 would not have to obey the rules in the Annex.

MacDonald recommended that there be one way to do things, and that the IEEE Annex be informative only.

SV In favor of agenda time at the next meeting for floating point extensions?
18 Yes. 0 No. 0 Abstain.

*** The review committee for floating point extensions is Farance, Lynch, and Johnston (from HP).

SV In favor of making any IEEE support in conditionally normative?
8 Yes. 3 No. 7 Abstain.

18 Specification Based Extended Integers [N459] (Farance)

Farance presented his proposal N459.

After a brief discussion on integral promotions, Keaton decided Farance correctly modeled the existing rules regarding signed versus unsigned.

Keaton observed that the "p" suffix conflicts with hex floating point.

Gwyn asked what was the meaning of an exact 18 bit type.

Farance replied that it probably would be a 32 bit container with 18 bits extracted from it when operated upon.

MacDonald expressed a level of discomfort with the amount of prior art for this sort of feature.

Thomas asked if attributes like "fast" were still part of results. Farance answered not anymore: the goal is minimal information associated with a temporary.

40

Plum stated he was opposed to the entire direction. The combinatorial scheme introduces thousands of integer types, which swamps overloading. This would lead to a feature that C++ could not adopt back from C9x.

Meyers expressed his opinion that the world didn't really need exact 19 bit types, and shared Plum's concern for overloading.

Plum stated that he use to give much more complicated advice about choosing integer types, but now believes in simpler rules. Use int if you are not concerned about portability. Use int and then long (for big data) if you want to be more portable. A simpler, somewhat clumsy, approach suffices.

SV In favor of agenda time at the next meeting for specification based extended integer types? [This vote was retaken later in the week.]
8 Yes. 7 No. 3 Abstain.

19 inttypes.h [N403] (Kwan)

Kwan presented his proposal N403.

Degener observed that stroumax() when truncated to six characters conflicts with the name of another function.

Farance asked what were the promotion rules for the types defined by the typedefs.

Meyers replied that the promotion rules were the same as the underlying built in types. Since the underlying types would vary from implementation to implementation, the promoted types would as well. However, programmers could rely on the normal C rules to choose a promoted type appropriate for the result.

Degener stated that the atleast, fast, and exact typedefs should be dropped.

Farance stated that those attributes are needed by programmers.

Gwyn observed that the atleast types help out machines that do not organize data into units of 8, 16, and 32 bits.

FVP (Kwan/Meyers) Move that inttypes.h as described in N403 be added to C9x.
9/6/0/2/17

WGV In favor of adding inttypes.h to C9x?
1 Yes. 2 No. The vote failed consensus.

SV In favor of agenda time at the next meeting for inttypes.h?
15 Yes. 3 No. 0 Abstain.

Mooney stated that the printf and scanf macros should be dropped and that the number of typedefs be reduced.

20 Empty Macro Arguments [N418]

There was general agreement to give defined meaning to macro arguments consisting of no tokens. MacDonald recalled that Keizer had objected to F() being recognized as a macro call with no tokens for its argument.

*** Gwyn will champion a proposal to give defined meaning to macro arguments consisting of no tokens.

SV In favor of agenda time at the next meeting for macro arguments consisting of no tokens?
17 Yes. 0 No. 0 Abstain.

21 Boolean support [N477] (Plum)

Plum presented his proposal N477.

There was general agreement that there should be some sort of macro defined that tells you if bool is supported. MacDonald observed that you could use #ifdef true to test for the support, if true was a macro.

Plum mentioned that some people wanted true and false to be defined for the preprocessor. Otherwise, #if true would act like #if 0, a surprising result.

Degener asked what was the payoff in this proposal? Plum said greater compatibility with C++. Gwyn, Meyers, Jones saw utility in just standardizing the names.

SV In favor of agenda time at the next meeting for bool?
14 Yes. 2 No. 2 Abstain.

Degener and Meyers stated that to avoid breaking all of the current user definitions of bool, the names should be defined in a new header file.

SV In favor of bool as a keyword?
3 in favor of a pure keyword
8 opposed to a pure keyword
7 Abstain

SV In favor of allowing implementations to choose whether to define a new keyword or define in a header?
3 in favor of allowing implementations to choose
12 opposed to allowing implementations to choose
3 Abstain

42

SV In favor of defining bool in a new header?
11 Yes. 4 No. 3 Abstain.

22 Signed integer division [N452] (MacDonald)

MacDonald presented his proposal N452 to adopt FORTRAN's definition of signed integer division.

Keaton asked if any machines are affected by the change. Plauger reported that there are few such machines left.

Gwyn observed that the definition of the % operator was changed.

MacDonald replied that the new definition was just an algebraic transformation of the old definition.

Jones observed that the proposed definition for divide matched the div() function, but the words were different.

MacDonald will reconcile the wording.

Gwyn made a comment about divide by zero.

FVP (MacDonald/Walls) Move that the definition of signed integer division in N452 be added to C9x?
14/1/0/2/17

WGV In favor of adopting the definition of signed integer division in N452 into C9x?
3 Yes. 0 No.

*** MacDonald will supply Benito with rationale for the definition of signed integer division.

*** Thomas, Degener, Gwyn will be the editorial board for signed integer division.

23 Tag compatibility [N453] (MacDonald)

MacDonald presented his proposal N453.

[The secretary's notes are not clear. It appears that some discussion of the compatibility of complete types and incomplete types across modules occurred. N453 does not reflect changes to Subclause 6.1.2.6 made by a TC on this subject.]

Jones stated that this was a quiet change.

Plum pointed out that the proposal increased C++ compatibility.

Jones and Plum proposed dropping the special rule allowing the compatibility of tagless structs.

Gwyn and Meyers pointed out one popular coding style avoids tags by defining all struct types in typedefs.

Plum reported that C++ had adopted the policy of using the typedef name as the tag for such cases.

SV In favor of agenda time at the next meeting for tag compatibility?
17 Yes. 0 No. 1 Abstain.

SV In favor of tagless structs being incompatible with tagged structs across compilation units?
10 Yes. 1 No. 6 Abstain.

SV In favor of tagless structs being incompatible with other tagless structs across compilation units?
3 Yes. 4 No. 11 Abstain.

24 Ordering/alignment/data representation [N464, N465] (Farance)

Farance presented his proposals N464 and N465.

A long discussion occurred on whether endian-ness applied to bytes or bits and bytes.

Farance stated that current practice for handling endian-ness translation is the host to network long function, named htonl(), whose use is error prone.

Jaeschke asked if there were problems with the extensions proposed and vendor specific controls such as #pragma pack.

Meyers answered that such interactions could be worked out.

Meyers stated the proposal was not sufficient to solve the problem. Network programmers want to send arbitrary structs down the wire as a sequence of bytes and then reassemble the bytes into a usable struct. In general, this requires knowledge of not only alignment and endian-ness, but compiler strategies for ordering bitfields, splitting bitfields across addressable units, and the pad unit for bitfields.

Kwan stated that the align specifier is too general: why support aligning on a 5 byte boundary?

SV In favor of agenda time at the next meeting for endian-ness and alignment?
5 Yes. 8 No. 3 Abstain.

SV In favor of agenda time at the next meeting for endian-ness?
[This vote was retaken later in the week.]
5 Yes. 6 No. 5 Abstain.

SV In favor of agenda time at the next meeting for alignment? [This vote was retaken later in the week.]
4 Yes. 8 No. 4 Abstain.

25 LIA, LID, LIPC [N461, N463] (Farance)

Farance presented his papers N461 and N463.

There was a discussion of using a new header for LIA extensions, and of extending float.h with LIA information.

Plum observed that LIA now permitted quietly wrapping signed integer arithmetic as one of its models.

N461, Page 9, last paragraph before 3.1.4 discusses LIA's requirements on a "hard to ignore" message for overflow. How does this fit in with C?

Farance should work with Thomas, and should check on LIA's notification requirements.

There was a discussion (similar to the discussion about IEEE arithmetic) whether LIA should be in an Annex, and whether the Annex should be normative or informative.

SV In favor of agenda time at the next meeting for LIA?
18 Yes. 0 No. 0 Abstain.

26 Voting Rules for Agenda Time

The committee revisited the rule adopted earlier in the week on voting a proposal agenda time at the next meeting.

Keaton found it strange that a proposal could get a majority of "Yes" votes for agenda time and fail to be placed on the agenda because of "Abstain" votes. He suggested that abstentions be ignored or that a "Yes" or "No" vote be forced by taking a formal vote.

Jones pointed out that only formal votes on technical matters forbid abstentions.

Plum stated that national bodies can always force items onto the agenda.

Jaeschke stated that he wanted to avoid the situation that occurred with some NCEG proposals where only lukewarm interest and abstentions kept the proposals alive.

MacDonald and Meyers requested that people stop abstaining on agenda votes.

45

Gwyn stated that he wanted to avoid wasting time revisiting proposals that lack support. However, the rule agreed to earlier in the week seems to allow a proposal to resurface every other meeting.

Keaton stated that the system does work, and we do eliminate proposals without the rule on abstentions being "No" votes.

Thomas stated that people now know the effect of abstentions. We have a lot of work, and we need to carefully manage our agenda time.

Keaton stated that the rule on abstentions is as prejudicial as phrasing votes in the negative.

Martin suggested that after votes on agenda time that Jaeschke make the final decision.

Jones stated he abstains when he honestly does not know if he wants to hear more on a topic.

SV A proposal before the committee will receive agenda time at the next meeting if more people vote "Yes" than "No" to give it agenda time.
9 Yes. 8 No. 0 Abstain.

27 Separate US TAG and WG14 admin meetings

The minutes of the US TAG are appended at the end of these minutes.

28 Extended Characters [N458] (Farance)

Farance presented his proposal N458.

Plum reported that SC22 just supported a resolution to SC2 to name characters based on their ISO 10646 code. It is also the position of SC22 that programming languages will not be required or mandated to deal with overstrike or combining characters.

Gwyn asked about combining characters in identifiers.

Plauger stated that programming languages are not obliged to take account of level 2 and level 3 characters. The character formed by "A", "backspace", "/" need not be recognized as the same as the character formed by "/", "backspace", "A". This does not mean that an identifier could not be recognized as a sequence of 10646 characters, just that identifiers that are linguistically the same but spelled differently would appear as different identifiers.

Meyers and Plum pointed out that the encoding of a named

46

character depends upon the compile-time locale.

Plum stated that compilers could just eat \U specifications in identifiers. However, a \U named character might fail to translate in a particular locale if it is not part of the character set. There are potential issues with displaying identifiers in error messages: is the \U form or a human readable form used?

Gwyn observed a similarity between trigraphs and \U. \U allows a universal source form for interchange.

Meyers asked if \U was really what people wanted. Don't they want to type in their source and see it displayed in their natural language?

Plum stated the model is for editors to allow just that, and a prepass converts extended characters into \U form for the compiler.

*** Farance will write separate proposals for extended characters and extended identifiers.

SV In favor of agenda time at the next meeting for extended identifiers and extended characters?
14 Yes. 2 No. 1 Abstain.

*** Gwyn, Farance, Meyers, Kwan, and Plum are the ad hoc committee to work with C++ on these issues.

*** Plum will get the topic discussed on the C++ C compatibility reflector.

29 Re-votes on Agenda Time

These votes supersede votes taken earlier in the week.

SV In favor of agenda time at the next meeting for compound literals?
9 Yes. 7 No. 0 Abstain.

SV In favor of agenda time at the next meeting for specification based extended integer types?
3 Yes. 13 No. 1 Abstain.

SV In favor of agenda time at the next meeting for endian-ness?
2 Yes. 12 No. 3 Abstain.

SV In favor of agenda time at the next meeting for alignment?
3 Yes. 12 No. 2 Abstain.

Several people said that if a proposal takes a new approach, a

47

vote for new agenda time could be requested.

30 X3 (Jameson)

Scott Jameson reported that X3 was trying to reduce confusion on participation fees and hold down costs. In the past, separate $300 invoices were sent out for the Secretariat and International fees. This caused confusion when people thought they were being double billed. Future invoices will include both fees. X3 did drop about 150 members for failing to pay the international participation fee.

X3 has rejected a fee increase. ITI, the X3 Secretariat, will try to reduce costs.

Farance expressed dissatisfaction with X3 cost of services, and found it hard to find out financial information about X3.

Jameson stated he had the information, and Farance could contact him.

Plum raised the issue of non-US domiciled organizations that are members of X3J11. They must pay the international fee even though they can not be part of the US TAG.

Jameson reported that the new JTC1 DIS rules were being delayed or rejected by ISO. A DIS is no longer frozen.

Jameson also reported that JTC1 has endorsed SC22's electronic document distribution strategy.

31 Business from Earlier in the Week

There was general approval to adopt the Netherlands suggestion for RR2.

*** Jones will provide DR wording to Gwyn.

*** Degener will provide wording on N441 DR152 concerning longjmp() and signal handlers.

*** Keaton is the administrator for the WG14 web page.

32 Process Miscellaneous Proposals

48

32.1 Allowing wchar_t to be a keyword [N479] (Plum)

Plum withdrew this proposal based on the committee's feedback on bool.

32.2 //-style comments [N481] (Gwyn)

Gwyn presented his proposal N481.

The committee observed that in unusual cases // comments could be a quiet change breaking some C89 programs:

```
x = a //* */ b
    + c;
```

x is assigned a / b + c under C89. With // comments, x is assigned a + c.

```
f(a//) + g(
);
```

f is a macro that stringizes its argument and returns an integer. In C89, f stringizes "a//" and g() is called. With // comments, f stringizes "a" and g is not called.

Some people expressed opposition to quiet changes. Plum stated that quiet changes for rare cases was acceptable.

Martin suggested an optional warning and that the rationale discuss the issue.

SV In favor of agenda time at the next meeting for // comments?
16 Yes. 0 No. 0 Abstain.

32.3 New Machine Characteristics macros [N478] (Tribble)

Gwyn and Meyers stated that the idea of additional characteristics macros was a good one in the abstract, but they could quibble with the ones in the proposal.

Degener stated the proposal should pay more attention to host versus target issues.

Thomas wondered about the applicability of some of the information.

MacDonald wondered what should a compiler targeting an interpreter define for some of the macros.

SV In favor of agenda time at the next meeting for new machine characteristics macros?
4 Yes. 8 No. 5 Abstain.

32.4 Replacement for sprintf [N430] (Bostic)

Degener asked what the return value should be if the buffer is too small. BSD returns the total number of characters needed. The wchar_t functions return EOF.

Gwyn expressed support for the proposal, but the old sprintf must be kept. A wide char version of the proposed function is also needed.

There was a lack of support for changing the return value of the existing wcs functions.

SV In favor of agenda time at the next meeting for new versions of sprintf?
13 Yes. 2 No. 2 Abstain.

*** Gwyn will work with Bostic on a revised proposal for new versions of sprintf.

33 Administration

33.1 Future Meetings

33.1.1 Future Meeting Schedule

Date	Location	Host
5-9 Feb 96	Irvine, California	Unisys
24-28 Jun 96	Amsterdam, the Netherlands	Vrije Universiteit
21-25 Oct 96	Toronto, Canada	IBM
3-7 Feb 97	Kona, Hawaii	Plum Hall
23-27 Jun 97	No host yet.	
20-24 Oct 97	California	Sun
Oct 98	New York, New York	Farance

BSI is the fall back host for the Jun 97 meeting.

Plauger offered to investigate having Whitesmiths Australia host the Jun 97 meeting in Sydney.

SV In favor of meeting in Sydney, Australia in Jun 97?
12 Yes. 1 No. 2 Abstain.

*** For the next meeting, Jaeschke will post to the reflector an agenda for the X3J11/WG14 meeting, an agenda for the US TAG meeting, and a list of evening sessions.

50

33.1.2 Future Agenda Items

Future agenda items appear as action items and straw votes throughout these minutes.

33.1.3 Future Mailings

Plauger thanked Cray (MacDonald) for handling the international mailing.

*** Van Sickle will handle the post-Nashua and pre-Irvine international mailings.

Plum reported WG21 has established document distribution via FTP, which several WG21 members are now using instead of the paper document distribution.

*** Farance will post to the reflector an electronic document distribution proposal.

Plauger requested that members be prepared to volunteer to handle the next international mailings.

All document numbers must be obtained from Plauger:

P. J. Plauger
398 Main Street
Concord, MA
01742 USA

Phone: +1 508 369 8489
FAX: +1 508 371 9014
E-mail: pjp@plauger.com

Items for mailings should have the document number on the cover of the document, and then should be sent to Plauger. He prefers hard copy, then fax, then e-mail. When sending hard copy, please print single sided.

The deadline to get papers to Plauger for the post-Nashua mailing is Friday, 10 November 1995.

The deadline to get papers to Plauger for the pre-Irvine mailing is Friday, 22 December 1995.

33.2 Resolutions

33.2.1 Review of Decisions Reached

Meyers reviewed the votes taken at the meeting, which are marked in the left margin of these minutes.

33.2.2 Formal Vote on Resolutions

WG14 took the following votes:

WGV Move that any any universal collating algorithm provide a commercially reasonable implementation using strxfrm() and wcsxfrm().
3 Yes. 0 No.

WGV Move that we accept the proposed responses to defect reports.
3 Yes. 0 No.

WGV Move we enable the convenor to amend RR2 as requested by the Netherlands and submit the amended document to SC22 for publication.
3 Yes. 0 No.

WGV Move we enable the convenor to acknowledge the comments of the United Kingdom on RR2.
3 Yes. 0 No.

WGV Move we appoint Keaton as the administer of the WG14 World Wide Web site.
3 Yes. 0 No.

33.2.3 Review of Action Items

Meyers reviewed the action items from the meeting, which are marked in the left margin of these minutes (***).

33.2.4 Thanks to Host

The committee thanked Meyers and Digital for hosting the meeting.

The committee thanked Bill McKeeman for his talk of Dynamic Differential Testing.

The committee thanked Bruce Foster for providing computer support for the meeting.

52

33.3 Other Business

34 Adjournment

The meeting was adjourned at 12:16pm on 20 October 1995.

5-3

1 US TAG Meeting

Jaeschke convened the meeting of the US TAG at approximately 4:30pm on 19 October 1995. Meyers served as secretary.

FVP (Benito/Keaton) Benito, Keaton, Farance, Walls, Plum will serve as the US delegation.
15/0/0/2/17

Farance suggested that the committee seek ISO 9000 certification.

There was some (sometimes amused) interest in the idea of a "standard standard".

Plum pointed out that Martin is an ISO 9000 auditor, and could be consulted. Are SC22 and JTC1 considered the management of the committee for the purposes of ISO 9000 certification?

Jaeschke wondered if SC22 would allow this.

The meeting adjourned at approximately 5:00pm on 19 October 1995.

AGENDA FOR 16-20 OCT 95
MEETING OF ISO/JTC1/SC22/WG14 AND X3J11
WG14/N476 X3J11/95-077

16 Oct 95	09:00-12:00	13:30-17:30
17 Oct 95	08:30-12:00	13:30-17:00
18 Oct 95	08:30-12:00	13:30-17:00
19 Oct 95	08:30-12:00	13:30-17:00
20 Oct 95	08:30-12:00	

Clarion Somerset Hotel
2 Somerset Parkway
Nashua, New Hampshire 03063
USA

Monday Oct 16th

8:30 -- 9:00 --- Coffee ---

9:00 -- 10:15 1. Opening activities

1.1 Opening Comments
1.2 Introduction of Participants
1.3 Selection of Meeting Chair
1.4 Host Facilities/local information
1.5 Procedures for this Meeting
1.6 Approval of Previous Minutes [N434]
1.7 Review of Action Items and Resolutions
1.8 Approval of Agenda [N476]
1.9 Distribution of New Documents
1.10 Information on Next Meeting
1.11 Identification of National Bodies/X3J11 voting members

10:15 -- 10:30 --- Morning break ---

10:30 -- 11:15 2. Reports on Liaison Activities

2.1 X3J11 + ANSI
2.2 WG14 + ISO/SC22
2.3 X3J16/WG21 (C++)
2.4 WG15 (Posix)
2.5 WG20 (I18N)
2.6 Other Liaison Activities [N456, N460, N462]
(Farance)

11:15 -- 11:40 3. Redactor Reports (Farance, Benito)

11:40 -- 12:00 4. Defect Reports [N441, N455]
(overview and announce review groups)

12:00 -- 13:30 --- LUNCH ---

13:30 -- 15:15 5. Defect Reports, review by groups

55

```
15:15 -- 15:30    --- Afternoon break ---

15:30 -- 16:30   6a. Defect Reports, proposed responses

16:30 -- 17:30   6b. LIA presentation (Schaffert)


Tuesday Oct 17th

 8:00 --  8:30    --- Coffee ---

 8:30 -- 10:15   7. Defect Reports, wrap-up

10:15 -- 10:30    --- Morning break ---

10:30 -- 11:15   8. Revision of ISO/IEC 9899:1990 (Jaeschke)
                    Rules for closure and overall game plan

11:15 -- 12:00   9. Electronic Distribution Policy [N454] (Keaton)

12:00 -- 13:30    --- LUNCH ---

13:30 -- 14:15   10. Designated Initializers [N472, N473, N474] (Keaton)

14:15 -- 15:15   11. Compound Literals [N475] (Keaton)

15:15 -- 15:30    --- Afternoon break ---

15:30 -- 16:15   12. Restricted Pointers [N448] (MacDonald)

16:15 -- 17:00   13. Variable-Length arrays [N451] (MacDonald)


Wednesday Oct 18th

 8:00 --  8:30    --- Coffee ---

 8:30 -- 10:15   14. Classes in C [N424, N445, N446, N447] (Walls, Jervis

10:15 -- 10:30    --- Morning break ---

10:30 -- 12:00   15. Classes in C (continued)

12:00 -- 13:30    --- LUNCH ---

13:30 -- 14:45   16. Classes in C (continued)

14:45 -- 15:15   17. C and Posix [N___] (Simonsen)

15:15 -- 15:30    --- Afternoon break ---

15:30 -- 17:00   18. Complex Arithmetic [N426, N470, N471] (Thomas) [N450
(MacDonald)
```

56

Thursday Oct 19th

```
 8:00 --  8:30   --- Coffee ---

 8:30 -- 10:15  19. Floating-point extensions [N466, N467, N468, N469] (
omas)

10:15 -- 10:30   --- Morning break ---

10:30 -- 11:30  20. Extended Integers [N459] (Farance)

11:30 -- 12:00  21. Extended Integers [N403] (Kwan)

12:00 -- 13:30   --- LUNCH ---

13:30 -- 14:00  22. Boolean support [N477] (Plum)

14:00 -- 14:25  23. Signed integer division [N452] (MacDonald)

14:25 -- 14:45  24. Tag compatibility [N453] (MacDonald)

14:45 -- 15:15  25. Ordering/alignment/data representation [N464, N465]
               (Farance)

15:15 -- 15:30   --- Afternoon break ---

15:30 -- 16:30  26. LIA, LID, LIPC [N461, N463] (Farance)
               26b. Extended Characters [N458]

16:30 -- 17:00  27. Separate US TAG and WG14 admin meetings
```

Friday Oct 20th

```
 8:00 --  8:30   --- Coffee ---

 8:30 -- 10:15  28. Process Miscellaneous Proposals

   Addition of predefined identifier __FUNC__ [N419] (Tribble)
   Allow empty arguments in macro replacement [N418] (Tydeman)
   New Machine Characteristics macros [N478] (Tribble)
   Replacement for sprintf [N430] (Bostic)
   Replacement for strtok [N429] (Bostic)
   Allowing wchar_t to be a keyword [N479] (Plum)
   Function Literals [N480] (Long)
   New Form of Pragma [N449] (McDonald)
   //-style comments [N481] (Gwyn)

10:15 -- 10:30   --- Morning break ---

10:30 -- 11:00  29. Continue Processing Miscellaneous Proposals

11:00 -- 12:00  30. Administration

   30.1 Future Meetings

           30.1.1 Future Meeting Schedule
```

30.1.2 Future Agenda Items
30.1.3 Future Mailings (sponsor for WG14 mailings)

30.2 Resolutions

30.2.1 Review of Decisions Reached
30.2.2 Formal Vote on Resolutions
30.2.3 Review of Action Items
30.2.4 Thanks to Host

30.3 Other Business

12:00 31. Adjournment

58

Attendance Roster - X3J11

Voting members	NAME	Mon	Tue	Weds	Thu	Fri	
1 Convex							
2 Cray Research	Tom MacDonald	✓	✓	✓	✓	✓	✓
3 DEC Professional	Rex Jaeschke	✓	✓	✓	✓	✓	✓
4 Digital Equipment	Randy Meyers	✓	✓	✓	✓	✓	✓
5 Farance Inc	Frank Farance	✓	✓	✓	✓	✓	✓
6 Hewlett-Packard	JOHN KWAN	✓	✓	✓	✓	✓	✓
7 IBM Corp	Dave Mooney	✓	✓	✓	✓	✓	✓
8 Keaton	David Keaton	X	X	X	X	X	✓
9 Motorola	TED VAN SICKLE	X	X	X	X	X	✓
10 Perennial	John C. Benito	X	X	X	X	X	✓
11 Plum Hall	Tom Plum	X	X	X	X	X	✓
12 RG Consulting							
13 SDRC	LARRY JONES	✓	✓	✓	✓	✓	✓
14 Sun	Douglas Walls	✓	✓	✓	✓	✓	✓
15 Taligent	Jim thomas	✓	✓	✓	✓	✓	✓
16 Tydeman							
17 Unisys	Jonathan Ziebell	✓	✓	✓	✓	✓	✓
18 US Army	Douglas A. Gwyn	X	X	X	X	X	✓
19 Watcom							
Advanced Micro Devices	Tom Lynch	X	X	X	X		

P. J. Plauger	Convener	
Neil Martin	BSI	
Jutta Degener	DIN	
Bryan Higgs	Oracle	
Ed Johnston	HP	
Alan H. Martin	Digital	(Fri)
Scott Jameson	DIGITAL	X3/CML LIAISON FRI