Document Number: P2524R0
Date: 2022-01-15
Authors: Michael Wong
Project: Programming Language C++, SG14 Games Dev/Low Latency/Financial Trading/Banking/Simulation/Embedded
Reply to: Michael Wong <michael@codeplay.com>

SG14: Low Latency/Games Dev/Embedded/Financial Trading/Banking/Simulation Meeting Minutes 2020/12/09-2022/01/12

Contents

Minutes for 2020/12/09 SG14 Conference Call

On Wed, Dec 9, 2020 at 9:54 AM Michael Wong <fraggamuffin_at_[hidden]> wrote:

> Thanks both, here is the adjusted agenda for today's call
>
> Topic: SG14 Low Latency Monthly
>
> This meeting is a special meeting to gather interest in Safety and
> Security proposed by Matthew Butler in May 2020 Sg14 meeting.
>
> Hi,
>
> Michael Wong is inviting you to a scheduled Zoom meeting.
>
> Topic: SG14 monthly Dec 2020-Feb 2021
> Time: Dec 9, 2020 02:00 PM Eastern Time (US and Canada)
> Every month on the Second Wed, until Feb 10, 2021, 3 occurrence(s)
> Dec 9, 2020 02:00 PM
> Jan 13, 2021 02:00 PM
> Feb 10, 2021 02:00 PM
> Please download and import the following iCalendar (.ics) files to
> your calendar system.
> Monthly:
> https://iso.zoom.us/meeting/tJcscuiqqD8pHNESxi1bJ9ClURVqr_ZAvmv1/ics?icsToken=98tyKuCrrz4rEtKRsx-CRowqBY_4d_zwpilego14rwfsUiJ5OyD6A9B0I6BAKvnG
>
> Join from PC, Mac, Linux, iOS or Android:
> https://iso.zoom.us/j/93151864365?pwd=aDhOcDNWd2NWdTJuT1loeXpKbTcydz09
> Password: 789626
>
> Or iPhone one-tap :
> US: +12532158782,,93151864365# or +13017158592,,93151864365#
> Or Telephone:
> Dial(for higher quality, dial a number based on your current location):
> US: +1 253 215 8782 or +1 301 715 8592 or +1 312 626 6799 or +1
> 346 248 7799 or +1 408 638 0968 or +1 646 876 9923 or +1 669 900 6833
> or 877 853 5247 (Toll Free)
> Meeting ID: 931 5186 4365
> Password: 789626
> International numbers available: https://iso.zoom.us/u/abRrVivZoD
>

> Or Skype for Business (Lync):
> https://iso.zoom.us/skype/93151864365
>
> Agenda:
>
> 1. Opening and introduction
>
> ISO Code of Conduct
>
<https://isotc.iso.org/livelink/livelink?func=ll&objId=20882226&objAction=Open&nexturl=%2Flivelink%2Flivelink%3Ffunc%3Dll%26objId%3D20158641%26objAction%3Dbrowse%26viewType%3D1>
>
> ISO patent policy.
>
> https://isotc.iso.org/livelink/livelink/fetch/2000/2122/3770791/Common_Policy.htm?nodeid=6344764&vernum=-2
>
>
> WG21 Code of COnduct:
>
>
> https://isocpp.org/std/standing-documents/sd-4-wg21-practices-and-procedures
>
> 1.1 Roll call of participants
>
John Macfarlane
Gaby Dos Reis
Matthew Butler
Aaron Ballman
Aleksandar Veselinovic
Alex Christensen
Anreas weisz
Andrew Lumsdaine
Antony Peacock
Arthur O'Dwyer
Ben Saks
Conor Horman
Daniel Papke
Edward Catchpole
erhard Ploederreder

Geoffrey Viola
Guy Davidson
Henry Miller
Herb Sutter
Inbal Levi
Jan Babst
JF Bastien
Ken Dinne
Kim Nillson
Mateusz Pusz
Matthew Bentley
Miguel Ojeda
Paul Bendixen
Rene Ferdinand Rivera Morell
Roberto Bagnara
Ronan Keryell
Roonen riedman
Ryan McDougall
Scott Schurr
Sophia Poirier
Staffan Tjernstrom
Stephanie Even
Michael Wong

> 1.2 Adopt agenda
>
Approved

> 1.3 Approve minutes from previous meeting, and approve publishing
> previously approved minutes to ISOCPP.org
>
> 1.4 Action items from previous meetings
>
> 2. Main issues (125 min)
>
> 2.1 General logistics
>
> Future meeting plans
>
> Dec 9, 2020 02:00 PM ET/1900 UTC: Safety Security

> Jan 13, 2021 02:00 PM ET/1900UTC: Games
> Feb 10, 2021 02:00 PM ET/1900 UTC: Embedded
>
> 2.2 Paper reviews
>
> Matthew Butler to review Safety Security Proposal: Safety & Security
> Review Group proposal for tomorrow:
> https://docs.google.com/document/d/e/2PACX-1vQSH4wLATLo3bLHjU71v4GwB0Ztr0UScV_hEyTFYTFzo0vt7euXdDkrilI2W-EulNL8ATvhWKVl9Hvp/pub
> .
>

P2272 R1 SSRG proposal
perception problem
bloated code
they use smaller language, but they always grow
many moving parts
RG not an SG
like ABI RG
help language and library evolution
C++ experts who are also safe and secure and say I know how to exploit this
define a language language subset and show you the technique for using them
also covers education
all safety security systems have to address the same issues
memory, undefined, exception
should be D R0 paper
para for what is safety critical : any device with a human life attached to it
security issue with cloud download
safety can be confused memory safety. life critical use, safety is overloaded
C had a safety security in WG14, had very little consensus between security and safety practitioners: security want code that was defect code, safety need prove through analysis that it was correct, not as much overlap
how to handle that set of requirements in a single language ? it is done through CERT C and MISRA C (which moves some to security space),
C standard has Annex K on Bounds checking, Annex L on analyzability related Annex K, overlap to SG12 SG14
SG would not be invitation only
and also be passive on demand
attract experts, and help committee understand concerns, committee may

ignore that feedback
SG12 does not overlap WG23, SG12 is a partner that works with these concerns, they have not address issues its more about enumerating vulnerabilities
Safety security world thinks the e committee is not helping them,
mixin concerns by calling safety security, then split later

a list of security safety concerns, how to solve the is where WG14 ran into problems

concern should be not undefined behaviour, SSRG would modify it for expertise

SG12 questions? point out problem the nsolvin is easier

issues will go back and forth between EWG and SSRG: watch out for this APi

Safety Security is cross cutting, SG12 does not cover everything but combined with WG23 helps
we don't want SGs competing with each other,
community is confused: Is C++ about performance, or security and safety

dynamic memory, and exception is very complicated and not a single SG can solve the problem

my users do not have the technical capacity the way WG21 works, they don't have the time to drive forward their own proposals and may not have the knowhow of WG21 processes

we can be proactive on reaching out, or papers to run through the SG

Tool developers: is there a disconnect with them? solution may be in tooling, and not just language, we need better tooling to help with safety and security

different industry have different needs, also approach OS and chip design, one that cannot be solved just by C++ language, library, or tools

who would reviewing papers with the ability to call the authors of those papers into the meeting, and security group could have that mandate to interview that mandate directly to reduce noise

scope and possible high workload, safety and security tends be slow, you do have to take your time, part of the load balancing is not have everyone review everything

do we have a critical mass of wg21 attendees who are familiar with our process and have domain expertise.

SG12: topic is fundamental, and lower the confusion

Wg21 people with expertize:
Billy Baker
Ben Saks
Ryan MacDougal
Mateusz Pusz
John MacFarlane
Stephanie Even
Aaron Ballman
Andreas Weisz
Gaby Dos Reis
JF Bastien
Michael Wong
Matthew Butler

Not WG21 but experts

Ken Dunne
Jan Babst
Staffan Tjerstrom:

P1315: Secure clear Miguel
clear memory without optimizer removing the memory store

C has consensu that this can work
will it be imported to C++, need a C++ expert
SG22, SG12 saw it

optimizing away the call in C++ is the major problem,
any efor to say something should not be removed.

> P1315R5 - Miguel Ojeda - secure_clear
>
P1315: Secure clear Miguel
clear memory without optimizer removing the memory store

C has consensu that this can work
will it be imported to C++, need a C++ expert to help
WG142, SG12 saw it

> P1705R1 - Shafik Yaghmour - Enumerating Core Undefined Behavior
>
P1705: Shafik
list each UB in the core behaviour
reviewed by SG12: write a mock annex, 80% done
SG12 has seen it from Michael Spencer
presented to EWG
practical goal,
any proposal UB should be added to an annex

most people think UB that compiler to just translate C++ to assembly statements one by one, that is not the case

is it a defect report to highlight the omission if it is not there already?
annex is nonnormative, then we just go to normal DR process

> P1861R1 - Alex Christensen - Secure Networking in C++
> P2234 - Scott Schurr -"Consider a UB and IF-NDR Audit"
>
P2234 consider a UB and IF-NDR audit by Scott Shur
find UB very hard to reason about, been looking for leverage
illorm and no diagnostic required MDR, there are over 900 including whats in the library
examining the UB we have and how much we really need
if UB is non-essential can we re examine them and can we make it safer
such an audit can be useful
UB makes the language hard to reason about,

will schedule SG 12 review or this,

been wanting this since 2014, like signed integer, but there could be contentions
if there are things that could be push back to the compilation stage then all for it but UB can be positive for safety and security

uninit variables is a source of security issues, but having all init var 0 init is not the best either as 0 could be bad,UB gives leeway for a number of things, can't really standardize that, Core of the problem is that there is bug in the language, how can we have a language without those pitfalls

signed integer arith overflow from lawrence Crowl is another similar example; because optimizer has become very aggressive

How do you reconcile the 2 things if goal is to reduce UB? we categorize UP and give rationale and can we achieve the same goal by down grading to impl specified , or impl defined

people equate UB with discoverability because it allows sanitizer to discover it,

ubsan can trap for overflow detections but these tools are hard to deploy e.g. Donald Knuth has an overflow in this book but not found for years

unsigned overflow, if you remove UB it removes the overflow detection; so can UB discussion be combined with contracts; these are things that optimizers are allowed to make assumptions about? right contracts want focus on safety

30 years of saying UB is bad and rebranding is hard

UB is not really for developers, its for implementers, yes agree

Scott will work on revision R1,

> 2.2.1 any other proposal for reviews?
>
> SG14/SG19 features/issues/defects:
>
>

> https://docs.google.com/spreadsheets/d/1JnUJBO72QVURttkKr7gn0_WjP--P0vAne8JBfzbRiy0/edit#gid=0
>
>
> 2.3 Domain-specific discussions
>
> 2.3.1 SIG chairs
>
> - Embedded Programming chairs: Ben Craig, Wouter van Ooijen and Odin
> Holmes, John McFarlane
>
> - Financial/Trading chairs: Staffan Tjernström, Carl Cooke, Neal
> Horlock,
> Mateusz Pusz, Clay Trychta,
> - Games chairs: Rene Riviera, Guy Davidson and Paul Hampson
> - Linear Algebra chairs: Bob Steagall, Mark Hoemmen, Guy Davidson
>
> 2.4 Other Papers and proposals
>
> 2.5 Future F2F meetings:
>
> 2.6 future C++ Standard meetings:
> https://isocpp.org/std/meetings-and-participation/upcoming-meetings
>
> - 2020-11: (New York, tentative) Cancelled.
> - 2021-02-22 to 27: Kona, HI, USA Cancelled
>
> 3. Any other business
> Reflector
> https://lists.isocpp.org/mailman/listinfo.cgi/sg14
> As well as look through papers marked "SG14" in recent standards committee
> paper mailings:
> http://open-std.org/jtc1/sc22/wg21/docs/papers/2015/
> http://open-std.org/jtc1/sc22/wg21/docs/papers/2016/
>
> Code and proposal Staging area
> https://github.com/WG21-SG14/SG14
> 4. Review
>
> 4.1 Review and approve resolutions and issues [e.g., changes to SG's
> working draft]

>
> 4.2 Review action items (5 min)
>
> 5. Closing process
>
> 5.1 Establish next agenda
>
> 5.2 Future meeting
>
> Dec 9, 2020 02:00 PM Eastern Time (1900 UTC) : Security: Mathew
> Butler
>
> Jan 13, 2021 02:00 PM Eastern Time (1900 UTC): Games: Rene
>
> Feb 10, 2021 02:00 PM Eastern Time (1900 UTC): Embedded :
>
> Kind Rgds
>
> On Tue, Dec 8, 2020 at 11:20 PM Matthew Butler via SG14 <
> sg14_at_[hidden]> wrote:
>
>> Thanks, John. I will try to get this on the agenda for tomorrow if at all
>> possible,
>>
>> All, here is the Safety & Security Review Group proposal for tomorrow:
>> https://docs.google.com/document/d/e/2PACX-1vQSH4wLATLo3bLHjU71v4GwB0Ztr0UScV_hEyTFYTFzo0vt7euXdDkrill2W-EulNL8ATvhWKVl9Hvp/pub.
>> It's very short so I will put it last on the agenda for tomorrow.
>>
>> See you all then.
>>
>> Thanks,
>> Matt
>>
>> On Tue, Dec 8, 2020 at 10:36 AM John McFarlane <john_at_[hidden]>
>> wrote:
>>
>>> If we're looking at P1705, should we also review P2234 "Consider a UB
>>> and IF-NDR Audit"?
>>>

>>> On Tue, 8 Dec 2020 at 03:56, Matthew Butler via SG14 <
>>> sg14_at_[hidden]> wrote:
>>>
>>>> We have 3 other papers to review as well:
>>>>
>>>> P1315R5 - Miguel Ojeda - secure_clear
>>>> P1705R1 - Shafik Yaghmour - Enumerating Core Undefined Behavior
>>>> P1861R1 - Alex Christensen - Secure Networking in C++
>>>>
>>>>
>>>>
>>>> On Mon, Dec 7, 2020 at 8:49 PM Michael Wong via SG14 <
>>>> sg14_at_[hidden]> wrote:
>>>>
>>>>> Topic: SG14 Low Latency Monthly
>>>>>
>>>>> This meeting is a special meeting to gather interest in Safety and
>>>>> Security proposed by Matthew Butler in May 2020 Sg14 meeting.
>>>>>
>>>>> Hi,
>>>>>
>>>>> Michael Wong is inviting you to a scheduled Zoom meeting.
>>>>>
>>>>> Topic: SG14 monthly Dec 2020-Feb 2021
>>>>> Time: Dec 9, 2020 02:00 PM Eastern Time (US and Canada)
>>>>> Every month on the Second Wed, until Feb 10, 2021, 3 occurrence(s)
>>>>> Dec 9, 2020 02:00 PM
>>>>> Jan 13, 2021 02:00 PM
>>>>> Feb 10, 2021 02:00 PM
>>>>> Please download and import the following iCalendar (.ics) files to
>>>>> your calendar system.
>>>>> Monthly:
>>>>> https://iso.zoom.us/meeting/tJcscuiggD8pHNESxi1bJ9ClURVqr_ZAvmv1/ics?icsToken=98tyKuCrrz4rEtKRsx-CRowqBY_4d_zwpilego14rwfsUiJ5OyD6A9B0I6BAKvnG
>>>>>
>>>>> Join from PC, Mac, Linux, iOS or Android:
>>>>> https://iso.zoom.us/j/93151864365?pwd=aDhOcDNWd2NWdTJuT1loeXpKbTcydz09
>>>>> Password: 789626
>>>>>

>>>>> Or iPhone one-tap :
>>>>> US: +12532158782,,93151864365# or +13017158592,,93151864365#
>>>>> Or Telephone:
>>>>> Dial(for higher quality, dial a number based on your current
>>>>> location):
>>>>> US: +1 253 215 8782 or +1 301 715 8592 or +1 312 626 6799
>>>>> or +1 346 248 7799 or +1 408 638 0968 or +1 646 876 9923 or +1 669 900
>>>>> 6833 or 877 853 5247 (Toll Free)
>>>>> Meeting ID: 931 5186 4365
>>>>> Password: 789626
>>>>> International numbers available: https://iso.zoom.us/u/abRrVivZoD
>>>>>
>>>>> Or Skype for Business (Lync):
>>>>> https://iso.zoom.us/skype/93151864365
>>>>>
>>>>> Agenda:
>>>>>
>>>>> 1. Opening and introduction
>>>>>
>>>>> ISO Code of Conduct
>>>>>
<https://isotc.iso.org/livelink/livelink?func=ll&objId=20882226&objAction=Open&nexturl=%2Flivelink%2Flivelink%3Ffunc%3Dll%26objId%3D20158641%26objAction%3Dbrowse%26viewType%3D1>
>>>>>
>>>>> ISO patent policy.
>>>>>
>>>>> https://isotc.iso.org/livelink/livelink/fetch/2000/2122/3770791/Common_Policy.htm?nodeid=6344764&vernum=-2
>>>>>
>>>>>
>>>>> WG21 Code of COnduct:
>>>>>
>>>>>
>>>>> https://isocpp.org/std/standing-documents/sd-4-wg21-practices-and-procedures
>>>>>
>>>>> 1.1 Roll call of participants
>>>>>
>>>>> 1.2 Adopt agenda
>>>>>

>>>>> 1.3 Approve minutes from previous meeting, and approve publishing
>>>>> previously approved minutes to ISOCPP.org
>>>>>
>>>>> 1.4 Action items from previous meetings
>>>>>
>>>>> 2. Main issues (125 min)
>>>>>
>>>>> 2.1 General logistics
>>>>>
>>>>> Future meeting plans
>>>>>
>>>>> Dec 9, 2020 02:00 PM ET/1900 UTC: Safety Security
>>>>> Jan 13, 2021 02:00 PM ET/1900UTC: Games
>>>>> Feb 10, 2021 02:00 PM ET/1900 UTC: Embedded
>>>>>
>>>>> 2.2 Paper reviews
>>>>>
>>>>> Matthew BUtler to review Safety Security Proposal
>>>>>
>>>>> 2.2.1 any other proposal for reviews?
>>>>>
>>>>> SG14/SG19 features/issues/defects:
>>>>>
>>>>>
>>>>> https://docs.google.com/spreadsheets/d/1JnUJBO72QVURttkKr7gn0_WjP--P0vAne8JBfzbRiy0/edit#gid=0
>>>>>
>>>>>
>>>>> 2.3 Domain-specific discussions
>>>>>
>>>>> 2.3.1 SIG chairs
>>>>>
>>>>> - Embedded Programming chairs: Ben Craig, Wouter van Ooijen and Odin
>>>>> Holmes, John McFarlane
>>>>>
>>>>> - Financial/Trading chairs: Staffan Tjernström, Carl Cooke, Neal
>>>>> Horlock,
>>>>> Mateusz Pusz, Clay Trychta,
>>>>> - Games chairs: Rene Riviera, Guy Davidson and Paul Hampson
>>>>> - Linear Algebra chairs: Bob Steagall, Mark Hoemmen, Guy Davidson
>>>>>

>>>>> 2.4 Other Papers and proposals
>>>>>
>>>>> 2.5 Future F2F meetings:
>>>>>
>>>>> 2.6 future C++ Standard meetings:
>>>>> https://isocpp.org/std/meetings-and-participation/upcoming-meetings
>>>>>
>>>>> - 2020-11: (New York, tentative) Cancelled.
>>>>> - 2021-02-22 to 27: Kona, HI, USA Cancelled
>>>>>
>>>>> 3. Any other business
>>>>> Reflector
>>>>> https://lists.isocpp.org/mailman/listinfo.cgi/sg14
>>>>> As well as look through papers marked "SG14" in recent standards
>>>>> committee
>>>>> paper mailings:
>>>>> http://open-std.org/jtc1/sc22/wg21/docs/papers/2015/
>>>>> http://open-std.org/jtc1/sc22/wg21/docs/papers/2016/
>>>>>
>>>>> Code and proposal Staging area
>>>>> https://github.com/WG21-SG14/SG14
>>>>> 4. Review
>>>>>
>>>>> 4.1 Review and approve resolutions and issues [e.g., changes to SG's
>>>>> working draft]
>>>>>
>>>>> 4.2 Review action items (5 min)
>>>>>
>>>>> 5. Closing process
>>>>>
>>>>> 5.1 Establish next agenda
>>>>>
>>>>> 5.2 Future meeting
>>>>>
>>>>> Dec 9, 2020 02:00 PM Eastern Time (1900 UTC) : Security: Mathew
>>>>> Butler
>>>>>
>>>>> Jan 13, 2021 02:00 PM Eastern Time (1900 UTC): Games: Rene
>>>>>
>>>>> Feb 10, 2021 02:00 PM Eastern Time (1900 UTC): Embedded :
>>>>>

>>>>> Kind Rgds
>>>>> _____
>>>>> SG14 mailing list
>>>>> SG14_at_[hidden]
>>>>> https://lists.isocpp.org/mailman/listinfo.cgi/sg14
>>>>>
>>>> _____
>>>> SG14 mailing list
>>>> SG14_at_[hidden]
>>>> https://lists.isocpp.org/mailman/listinfo.cgi/sg14
>>>>
>>> _____
>> SG14 mailing list
>> SG14_at_[hidden]
>> https://lists.isocpp.org/mailman/listinfo.cgi/sg14
>>
>

Minutes for 2021/01/13 SG14 Conference Call

Thanks to Rene Ferdinand Riviera Morell for chairing.
Matthew Bentley, Maxime Laine, patch cam, Paul Bauman, Paul bendixen, Pedro Oliveira, Roberto Bagnara, Ronen Friedman, Staffan Tjernstrom
Jonas Jensen, John Mcfarlaine, JF Basttien, Jeffrey Mendelsohn, Jan Babst, Inbal Levi, Henry Miller, Edward Catchpole, Dave Bartolomeo, Conor Horman, Charles Bay , Carlos Chacon, Billy Baker, Ben Saks, Ben Craig, Andrew Lumsdaine, Andreas Vaselinovic, Guy Davidson, Rene Riviera, Michael Wong, Emmanuel, Damon McDougall, Sophia Poirier

Matthew Bentley presenting
been forwared to LEWG, after SG14
LEWG review Feb 2nd
general purpose unordered insertion container
Matt has worked with Jens to address wording,
hard to get through 23 now, due to ipeline being full
similar to flat containers and that is already in the pipeline
any wording review done early will help LWG
any perception this has no impl experience then people will be mre conservative, needs broad impl expereince,
Billy Baker LEWGI
Ben Craig, Vice chair Library Evolution
Inbal Library evolution
JF Language Evolution

proxy=Guy Davidson

2.2.1 any other proposal for reviews?

SG14/SG19 features/issues/defects:

https://docs.google.com/spreadsheets/d/1JnUJBO72QVURttkKr7gn0_WjP--P0vAne8JBfzbRiy0/edit#gid=0

LA syntax: Klaus implementing
LA BLAS: in LEWG
intrusive containers: Isabella may restart, Hal's paper is similar
Ring buffer: dont wait for Lawrence, get a list of differences
Trivially relocatable connect with gaspar

member layout control: similar to Hal using attributes, reflection related

2.3 Domain-specific discussions

2.3.1 SIG chairs

- Embedded Programming chairs: Ben Craig, Wouter van Ooijen and Odin Holmes, John McFarlane

Wouter's concern on atomics with , Paul Bendixen may collaborate on this
a third party macro, not controlled or discussed in committee, not to be done on reddit
write down the concern, may not need a paper

- Financial/Trading chairs: Staffan TjernstrÃ¶m, Carl Cooke, Neal Horlock, Mateusz Pusz, Clay Trychta,

paper on ring buffer use case
not just games
any messaging ,or network card is a ring buffer

- Games chairs: Rene Riviera, Guy Davidson and Paul Hampson

No Games meeting in December in future!

- Linear Algebra chairs: Bob Steagall, Mark Hoemmen, Guy Davidson

2.4 Other Papers and proposals

Security has a paper submitted by Matthew, and now there will be a discussion by the chairs
suggestion is through reflector, mail.

Minutes for 2021/02/10 SG14 Conference Call

On Tue, Feb 9, 2021 at 5:55 PM Michael Wong <fraggamuffin@gmail.com> wrote:

Topic: SG14 Low Latency Monthly

This meeting is focused on Embedded. It will be chaired by John MacFarlane.

Hi,

Michael Wong is inviting you to a scheduled Zoom meeting.

Topic: SG14 monthly Dec 2020-Feb 2021
Time: Jan 13, 2020 02:00 PM Eastern Time (US and Canada)
Every month on the Second Wed, until Feb 10, 2021, 3 occurrence(s)
Dec 9, 2020 02:00 PM
Jan 13, 2021 02:00 PM
Feb 10, 2021 02:00 PM
Please download and import the following iCalendar (.ics) files to your calendar system.
Monthly:
https://iso.zoom.us/meeting/tJcscuigqD8pHNESxi1bJ9ClURVqr_ZAvmv1/ics?icsToken=98tyKuCrrz4rEtKRsx-CRowqBY_4d_zwpilego14rwfsUiJ5OyD6A9B0I6BAKvnG

Join from PC, Mac, Linux, iOS or Android:
https://iso.zoom.us/j/93151864365?pwd=aDhOcDNWd2NWdTJuT1loeXpKbTcydz09
Password: 789626

Or iPhone one-tap :
US: +12532158782,,93151864365# or +13017158592,,93151864365#
Or Telephone:
Dial(for higher quality, dial a number based on your current location):
US: +1 253 215 8782 or +1 301 715 8592 or +1 312 626 6799 or +1 346 248 7799 or +1 408 638 0968 or +1 646 876 9923 or +1 669 900 6833 or 877 853 5247 (Toll Free)
Meeting ID: 931 5186 4365
Password: 789626
International numbers available: https://iso.zoom.us/u/abRrVivZoD

Or Skype for Business (Lync):
https://iso.zoom.us/skype/93151864365

Agenda:

1. Opening and introduction

ISO Code of Conduct
<
https://isotc.iso.org/livelink/livelink?func=ll&objId=20882226&objAction=Open&nexturl=%2Flivelink%2Flivelink%3Ffunc%3Dll%26objId%3D20158641%26objAction%3Dbrowse%26viewType%3D1
>

ISO patent policy.
https://isotc.iso.org/livelink/livelink/fetch/2000/2122/3770791/Common_Policy.htm?nodeid=6344764&vernum=-2

WG21 Code of COnduct:

https://isocpp.org/std/standing-documents/sd-4-wg21-practices-and-procedures

1.1 Roll call of participants

1.2 Adopt agenda

1.3 Approve minutes from previous meeting, and approve publishing
previously approved minutes to ISOCPP.org

1.4 Action items from previous meetings

2. Main issues (125 min)

2.1 General logistics

Future meeting plans

Apr 14, 2020 02:00 PM ET/1800 UTC: Finance focus on low latency
May 12, 2021 02:00 PM ET/1800UTC: Games
June 9, 2021 02:00 PM ET/1800 UTC: Embedded

2.2 Paper reviews

Compound assignment

C++ 20 deprecates compound assignment to volatile
memory map devices is common and uses this and create larger rift between C and C++ world
harder to get c project to just recompile in C++, now you have to replace compound assignment
header files also contain macros like this
its only deprecation but it is an intent to remove
difficult to pinpoint semantics problems, one read, one writeback

code that he does not control, and vendor headers in macros is the problme
the semantics of solution in paeper are the same

can we get C to deprecate this ?

most of volatile is unspecified what it means
we do a read of register and or it with something
reading a device register dont give you any sensible value back
for some HW it might work or not work
if deprecation is not good, and attach proper semantics, then write a paper

main way for a bit set

impedance mismatch of the fundamental unit of byte
hard to read |= in a data race
the scenario can exasperate it during times when no memory order

C/C++ liasion group to deprecate this in C
fix with replacing compound with non-compound, this just impose a style from C++ without
benefit, so may not want to deprecate compound ops

this is a well known idiom on some embedded HW
fix in embedded world with a style guide with MISRA
changes a lot of old code prevents moving to C++ in embedded, so recommend style guide to fix this

might not be funneling people down a path at all, but using operation provided to them by HW vendor
here is a piece of code that works in C and now they want me to try C++
; now this make people move back to C

a|=b
a=a|b

volatile atomics that can give slightly better semantics in some use case, but it is implementation specific

so specific semantics=volatile is used when using HW or signal, these need specific instructions
compiler might only honor the size, and not CAS

could this be a compiler warning? could be lots of false positive, what used to be there is fine

looks annoying to have this deprecation, will be no go for C++ in our tools
what are instructions emitted by volatile access, we sue C++ on CPU side and in FPGA, and replace C++ program with equivalent, and is volatile is emitting memory access
need to keep this as it is simpler for end user

support the other which is we see so many time in our code, assume that when you write compound that you are doing RMW , was always a bug
so obviously misleading and hard t keep safeguard

what would be good outcome here, easiest is to undeprecate, but what do you say? you dont say what they do and it is implementiaton specified, valid syntax, but you dont know what it means
std should define semantics of compound operation but that might make it worst

counter argument is it also has no defined semantcis in C either, C does not warn you but C++ does, so we make it appear to have different behavior when there is no difference

define volatile? yes there is another paper that is trying to define meaning of volatile load and store, and atomic volatile

ask C/C++ liasion about deprecatign for C23, if NO then we know we have input

JF to email aaron

hard to get a good feelign about each other's interest; see no consensus from the discussion, mixed on both sides

Freestanding?

P2268R0 is a roadmap

P2013 Language optional on EWG EP
P1642 Easy utilities in LEWG EP

Low cost deterministic exception for next Embedded call

P0829 is Std lib omnibus

beyond are still being discussed and needs implementation experience

Asking for volunteers

This is a future directions paper on embedded which is more then most desktops

PB has a fork of this paper

Minutes for 2021/04/14 SG14 Conference Call

Meeting notes.

On Tue, Apr 13, 2021 at 10:53 AM Michael Wong <fraggamuffin_at_[hidden]> wrote:

> Topic: SG14 Low Latency Monthly
>
> This meeting is focused on Finance/Low latency.
>
> Hi,
>
> Michael Wong is inviting you to a scheduled Zoom meeting.
>
> Topic: SG14 monthly
> Time: 2nd Wednesdays 02:00 PM Eastern Time 1800 UTC (US and Canada)
> Every month on the Second Wed,
> Apr 14, 2020 02:00 PM 1800 UTC
> May 12, 2021 02:00 PM 1800 UTC
> June 9, 2021 02:00 PM 1800 UTC
>
>
> Join from PC, Mac, Linux, iOS or Android:
> https://iso.zoom.us/j/93151864365?pwd=3DaDhOcDNWd2NWdTJuT1loeXpKbTcydz09
> Password: 789626
>
> Or iPhone one-tap :
> US: +12532158782,,93151864365# or +13017158592,,93151864365#
> Or Telephone:
> Dial(for higher quality, dial a number based on your current location=
):
> US: +1 253 215 8782 or +1 301 715 8592 or +1 312 626 6799 or +1
> 346 248 7799 or +1 408 638 0968 or +1 646 876 9923 or +1 669 900 6833
> or 877 853 5247 (Toll Free)
> Meeting ID: 931 5186 4365
> Password: 789626

> *International numbers available: https://iso.zoom.us/u/abRrVivZoD*
>
> *Or Skype for Business (Lync):*
> *https://iso.zoom.us/skype/93151864365*
>
> *Agenda:*
>
> *1. Opening and introduction*
>
> *ISO Code of Conduct*
> *<*
>
> *https://isotc.iso.org/livelink/livelink?func=3Dll&objId=3D20882226&objAct=*
ion=3DOpen&nexturl=3D%2Flivelink%2Flivelink%3Ffunc%3Dll%26objId%3D201586
41%=
26objAction%3Dbrowse%26viewType%3D1
> >
>
> *ISO patent policy.*
>
> *https://isotc.iso.org/livelink/livelink/fetch/2000/2122/3770791/Common_Po=*
licy.htm?nodeid=3D6344764&vernum=3D-2
>
> *WG21 Code of COnduct:*
>
>
> *https://isocpp.org/std/standing-documents/sd-4-wg21-practices-and-procedu=*
res
>
> *1.1 Roll call of participants*
>
Andreas Weis, Ben Saks, Billy Baker, Conor Horman, Henry Miller, Jakob Lovhall, Michael Adams, Piotr Grygorczuk, Puya Daravi, Rene Ferdinand Rivera Morell, Ronan Keryell, Sean Middleditch, Sophia Porier, Jeffrey Olkin, Matthew Butler, Michael Wong, Jens Maurer , Guy Davidson, Max Gardner, John McFarlane

> *1.2 Adopt agenda*

>
> 1.3 Approve minutes from previous meeting, and approve publishing
> previously approved minutes to ISOCPP.org
>
> 1.4 Action items from previous meetings
>
> 2. Main issues (125 min)
>
> 2.1 General logistics
>
> Future meeting plans
>
> Apr 14, 2020 02:00 PM ET/1800 UTC: Finance focus on low latency
>
> May 12, 2021 02:00 PM ET/1800UTC: Games
> June 9, 2021 02:00 PM ET/1800 UTC: Embedded
>
> 2.2 Paper reviews
>
> Compound assignment review
>
thank Wouter and SG14 in acknowledgment

intro should say embedded community (who never update their 3rd party header) that was overlooked in initial review which focused on SG1
so a different interface will not fix the problem for future
some compatibility layer from vendors for future? that is a future proposal

make sure +=3D does not appear from header analysis

mixed C and C++ impact: add SG22 lukewam reception
also porting of C to C++ impact

restore status quo in existence for 30 years

[depr.volatile.type] annex might need to be updated

de-deprecate is split in 3.2 make it a newline

> Low Latency topic brainstorm
>
fpga offload computation suing VHDL/verilog
nanosecond resolution, lock free programming
low latency, control of generated assembly
can I access hardware features, memory card,
C like subset of C++ for this,
memory barriers
no support for I/O control, DMA memory, whether device is there, but these may be out of scope

what does low latency mean? is it an upper bound? is it determinism?

also need thread priorities Patrice roy
also need cache controlling
also needing scheduling periodic thread intervals, by some deadline (flagging a real-time thread, deal with conflicts)
get a processor onto a cpu when its idle
we also need cpu pinning, like madvise
also mange memory mapped files carefully,
but most of these are owned by the OS

standardize filesystem, but many freestanding systems that don't have this, offer standard cross platform apis for this, is there standard policy for this
support some real time thread with a deadline, allows you to associate n-number of threads as working on same real time constraint, grouping api
should we add threading attribute that support this
Vulkan has a capability model, standard model, but capability fits specific hardware
standard api to query capability

use encoding like unicode api

what about Os vs bare metal / library only profiles

conditionally support is a kind of segmentation already
concern about SQL profiles which seem to fail
features with non-local
vft, I asked for it so thats ok
embedded small devices stuck with C89,

similar for templates which don't generate object code

we have compilers that have flags to subset; this leaves people out in the cold and segmented language, hwo to help a single C++ instead of subset dialects

floating point hardware is not needed for floating point, avoid use at runtime with a lookup table

annotate and enforce real-time constraint on a call path to say a function needs real-time, or trap at runtime if unable to satisfy; say threads is allowed or not allowed to call synchronous apis but aim to fail early

Freestanding?

>
> 2.2.1 any other proposal for reviews?
>
> SG14/SG19 features/issues/defects:
>
>
> https://docs.google.com/spreadsheets/d/1JnUJBO72QVURttkKr7gn0_WjP--P0vAne=
8JBfzbRiy0/edit#gid=3D0
>
> 2.3 Domain-specific discussions
>
> 2.3.1 SIG chairs
>
> - Embedded Programming chairs: Ben Craig, Wouter van Ooijen and Odin
> Holmes, John McFarlane

>
> - Financial/Trading chairs: Staffan Tjernstr=C3=83=C6=92=C3=86=E2=80=
=99=C3=83=E2=80=9A=C3=82=C2=B6m, Carl Cooke, Neal
> Horlock,
> Mateusz Pusz, Clay Trychta,
> - Games chairs: Rene Riviera, Guy Davidson and Paul Hampson
> - Linear Algebra chairs: Bob Steagall, Mark Hoemmen, Guy Davidson
>
> 2.4 Other Papers and proposals
>
> 2.5 Future F2F meetings:
>
> 2.6 future C++ Standard meetings:
> https://isocpp.org/std/meetings-and-participation/upcoming-meetings
>
> -
>
> 3. Any other business
> Reflector
> https://lists.isocpp.org/mailman/listinfo.cgi/sg14
> As well as look through papers marked "SG14" in recent standards committe=
e
> paper mailings:
> http://open-std.org/jtc1/sc22/wg21/docs/papers/2015/
> http://open-std.org/jtc1/sc22/wg21/docs/papers/2016/
>
> Code and proposal Staging area
> https://github.com/WG21-SG14/SG14
> 4. Review
>
> 4.1 Review and approve resolutions and issues [e.g., changes to SG's
> working draft]
>
> 4.2 Review action items (5 min)
>
> 5. Closing process
>
> 5.1 Establish next agenda

>
> *5.2 Future meeting*
>
> *Apr 14, 2020 02:00 PM 1800 UTC*
> *May 12, 2021 02:00 PM 1800 UTC*
> *June 9, 2021 02:00 PM 1800 UTC*
>
>
> *Kind Rgds*

Minutes for 2021/05/12 SG14 Conference Call

HI all here is the attendance and notes from the call: Sophia Poirier {she}
Michael Wong
connor horman
16175122142
Billy Baker
Jeffrey Olkin
René Ferdinand Rivera Morell
Derek Haines
Ronan Keryell
Brett Searles
Matthew Butler
Izzy Muerte
Staffan Tjernström
Daniel Papke
Sean Middleditch
Lee Mracek
Andrew Lumsdaine
Damon McDougall
Sophia Poirier {she} (Sophia)
Henry Miller
Guy Davidson
Ronen Friedman
Piotr Grygorczuk
Kim Nilsson
John McFarlane
Paul M. Bendixen
Matheus Izvekov
Inbal Levi
Marco Foco
Tristan Sizemore
John Law
Lucas

And the notes which are partial as I had a machine crash in the middle:
15:14:12 Yep. So, what I'm trying to say is that, if anyone wants to like make sure that these will reach the library, etc. So please, you know,

contact us and tell us this one's important currently there's sort of in the end of the queue, because no one's like
15:14:12 insistent on them.
15:14:15 Right.
15:14:17 Yeah, I think.
15:14:18 Su Su six is quite relaxed. This approach to getting players into the library. Yeah, absolutely fine.
15:14:29 Football pornos is a new idea. We don't even have a paper here, and meals not hearing me.
15:14:34 Okay.
15:14:35 Jon Jon's away improving debug builds.
15:14:40 Is this maybe I mean it kind of gets
15:14:46 brought brought up in relation to the slow builds as well. I think can be some fundraising it also relate to anything that has a lot of enlightening.
15:14:57 A lot of deep Call, call trees.
15:14:59 Yeah, now it's just one thing that I tried floating in.
15:15:08 the then sweep didn't proceed with it from there. There was some work to try and prototype some of this in clang briefly.
15:15:17 But I sort of didn't, I didn't chase up with that.
15:15:20 But, I mean, we were looking at different features, things that we can get in but then we also some of the things we're talking about our sort of broader topics and I error handling bill times and stuff like that.
15:15:34 I wonder if we want to have a list of, you know, broad topics, alongside a list of individual features and and papers that we have, because you know like shoulder, shoulder some mentioning earlier, take a lot of people will have still put off by slow
15:15:51 template code.
15:15:54 No. Read people are finding ranges, is another good example of something that just makes compiles slow, the, you know, and exhaust some memory as well.
15:16:21 Yes, I'm very sad about ranges looks great but I can't offending you're doing better with the language between the library features Sure.
15:16:17 Yeah that's hard to re mentioned I mentioned the paper earlier cargo regard to debugging but I think there's also an effect here author and chat it's p 1221 the parametric expressions proposal by Jason rice.
15:16:33 He has a demo of it on DOD bolt and I think a client branch, I don't know that here else has ever done like a deep performance analysis to

see how much this type of approach would actually help things like debug bills and iterations.
15:16:48 Like, conceptually in my head, it sounds like a good help but obviously that needs to be tested verified.
15:16:53 So I just wanted to like pass it out there and say, hey thing to consider putting the putting it in chat, but I just want to mention it.
15:17:01 Okay, yeah, that didn't occur to me that that that work might have a benefit there. So that's good to know. Thanks.
15:17:10 Yeah, I mean, it sounds, it sounds like, I mean, this is the topics that are raising a feeler.
15:17:17 Yeah, definitely not just mean they're a pain point for game developers, in particular, perhaps, but I think just generally for the, for the health of the language.
15:17:29 We can't just keep sort of leading hit more and more every on enlightening.
15:17:35 I mean I maybe want to see how how things are affected by modules, because you know I think everybody's in a holding pattern to see how they affect build times.
15:17:47 Nobody sounds particularly optimistic that you get anything other than that, you know, you know, why having a bell times that kind of thing.
15:18:04 But yeah, I mean ranges that the linear algebra, the and the, the proposals that I put forward, which, you know, required rapping types and types and types know a lot of angle brackets there when I when I took any of that to committee and say, and was
15:18:17 you know tentative tentatively saying you know is this too complex is this, I mean that it makes you It makes your eyes, boggle the, the, the amount of nesting going on there.
15:18:27 And I was getting, you know, encouragement to just go full steam ahead and, you know, increase that
15:18:35 without really any concern for you know Bill times, or complexity.
15:18:40 So, yeah, I just wonder if there's going to be a general solution. Anything library specific I feel.
15:18:49 Okay. Izzy.
15:18:51 I just wanted to comment that both the SIG and rust have adopted.
15:18:59 You know, an intermediate format to classical to more easily get metadata, and to bug, but would have been enlightened on like a per function basis like the Bitcoin support isn't entirely there because they do have to interact with c++, you know based

15:19:18 tools like GDP lol to be.
15:19:21 But there are now two buggers being written in rust that support, talking to Mary, for instance, directly, which is the, the interpreter for the intermediate format that Russ uses.
15:19:39 Whereas like right now, you know, lol VM, and clang have a byte code interpreter for constant mountain concepts for stuff but there's no way to debug that.
15:19:49 And I think that anything that like could be checked at compile time should also be the bubble right now that's not really an option.
15:19:59 Yeah, I just wanted to comment on them.
15:20:04 Right. Well, the other direction stuff that we, we might want to debug but can't, which is another thing that causes difficulty.
15:20:15 Yeah.
15:20:18 Okay, we have Ben's pledges next on Exception Handling size and speed.
15:20:25 With any luck hubs papers will
15:20:31 render these interesting historical artifacts, rather than critical point is to the pain of using exceptions.
15:20:40 Statistical functions I'm actually on this one myself.
15:20:45 I don't know that it's necessarily an sg 14 related paper, but it is a it is a 19 but does it have. Yeah, he was usage.
15:20:56 It is a test in 19 but does it have us usage. So it does is, it, it, it now enables things like means, median, mode variance and deviation and ketosis. Yes, I'm sure.
15:21:09 Yes, certainly, we use. Well, certainly the meeting.
15:21:13 Meeting men mean mode and medium, all the time.
15:21:22 Yes, it's not, it's nice to have an establishment, because then we don't end up writing our own incorrectly, which is what many people do with love and mid points.
15:21:28 I've noticed job.
15:21:31 Yes, guess what I'm going to talk about.
15:21:34 So, so this is this is the example I didn't actually mention it, but this was what I was thinking about earlier.
15:21:41 When I said a lot of some of these some functions, they will take two ranges, a little bit like for examples to transform can.
15:21:50 And if, if the ranges don't match in certain ways. Exceptions are thrown.
15:21:55 Yeah, I think somebody has actually, possibly even written a

paper, sort of question this was certainly got got in touch with the the authors to say Do you really want to be thrown sections for something that usually if you're using this right this simply
15:22:12 isn't going to happen.
15:22:14 And if it does, if it does happen then then presumably it's a bug, it's not the sort of thing that might happen and then you want to catch it and continue the best an energy
15:22:28 correcting each of these things on a per paper basis, basically following some advice from I think it's chapter three of the sections paper.
15:22:40 I mean it just seems a bit wasteful for, for people to have to repeat themselves, so often.
15:22:51 I,
15:22:51 Michael.
15:22:57 What was I gonna say okay yeah you're right there. That is that that objection has been raised and we're going to address that in tomorrow's meeting at this exact same time in SG 19, but thank you.
15:23:07 Yeah.
15:23:09 Yeah. I'll be as to 19 tomorrow as well.
15:23:15 differentiable programming that's also SG 19.
15:23:29 we can skip these if they don't, we don't think they're valuable This is essentially giving people a way of doing different differentiation. Oh Michaels here goes there. Yeah, Monica.
15:23:33 Please take the floor. No, no, oh yeah it's automatic differentiation paper. The first one, the second one we abandoned the idea, because we don't have the time to follow that that numeric differentiation part.
15:23:44 So if someone wants to pick it up. Yeah. You're welcome. But for me, kid. Right. Okay.
15:23:52 And for the financial world programming, so we we changed one of the others because now we have two implementations. So, one is at higher level, and one is at a higher level.
15:24:06 So that's called transformational level, it's quite interesting but I don't know if it's related to see 14 now so it's back to what I had on other things because I wanted to ask a question.
15:24:21 I don't think it wasn't bad festering Prague, we had a presentation of someone I don't remember who doing presenting a faster way of doing exceptions.
15:24:33 But I couldn't see anything here. It was not a p Baker that's why

I kind of find it.
15:24:42 Michael, you 21. Yes.
15:24:46 Yes. So that was a paper by a student, a PhD student who now works at on.
15:24:53 And it what there was no p paper and I still have to ask him to submit it as a piece of paper, but it was called.
15:25:00 It's called them determine is the exception for embedded devices, and it's built on work from hopes paper but extends it in a way that I think could work better for this community, it's still on my to do list.
15:25:16 And sorry I haven't done it but that is potentially another thing that we, we need to do for us to 14, and I was going to do that in the next call because the next call is embedded.
15:25:25 So I'm just going to get James, what's it was, I can't remember his last name. Yeah. But now I've got it right yeah it's a guy from arm, I don't know if you remember his name but yeah, I will do that.
15:25:39 No, I can't remember.
15:25:42 Okay.
15:25:44 If you can send me his name I can get in touch with him because I'm also interested in that.
15:25:48 Yeah, quite a few people were interested in. Okay, because he has a good help of this proposal and.
15:25:57 And he was hoping to get people to help them experiment with it. So I think that's, I know it's been over a year but that's what the pandemic busy. Yeah, exactly.
15:26:07 Sorry about that.
15:26:11 Thank you. Okay, thank you. But James Randi okay yeah someone like Michael I presume that's an old hand.
15:26:22 Michael I presume that's an old hand. But I was going to say for the next one also SG 19, but gradually lung stain i think is on and he might be able to say something about it.
15:26:30 Yes.
15:26:37 He has under is no longer in the meeting with us, I think, is supposed to be to gain coin.
15:26:43 And I'll say something about it. It's essentially a paper to do to build a standardized graph data structure would have Jason see nodes and vertices.
15:26:54 And that way, it can give you a path towards graph execution, which is something we've been talking about

15:27:03 building call graph executions.
15:27:05 It's, it's, it's a very nice piece of work I think and I think we should be able to leverage off of that, for people who are deeply involved or interested in building grass grass, it builds off the boost graph library, which you guys all know from BTL,
15:27:21 but now Andrew has like the fifth iteration of that bill and it's much improved, and he's been working with people from SAS, who are also involved as you know in them in machine learning, that's just everything there is it's all about graphs, it's about
15:27:37 convolution grass. So, as a result, it's a very very useful piece of work, but it's got more than most of the night, people are looking at it to use it for things like call graph building a called Building a car graph, like what we talked about.
15:27:55 Splendid.
15:27:59 Yeah, Yeah.
15:28:01 The comments show I'm saying 90% of my job is rising graph so you can be quite useful. Yep. Graph crop up a lot reverberations are assuming it's one game programming, whether it's a messenger application or logic.
15:28:14 Sounds great to me as a hobbyist Yeah, I'm looking forward to this paper and looking forward to this proposal.
15:28:17 So anyone wants to discuss it. Su 19 calls tomorrow.
15:28:22 And you can find out from the VSG 19 reflector of the call in number, same same same time as this one.
15:28:29 I like to get both both things done within one week. Yeah, I would command the su 90 meetings in general, I've been to a few now, and I'm finally quite stimulating exhilarating and refreshing, they appeal to different ways the SDR team meetings.
15:28:46 Right physical units British
15:28:51 six and 16.
15:28:57 Does anybody want to speak to this paper.
15:28:59 It's quite a simple paper simply as propose it proposing
15:29:04 unit three on two seconds, Michael, please save me. okay yeah sure so I'll talk about it.
15:29:11 I've had contact with Matusow about this and his paper is good, it's proposing something I've been talking about since I helped create user defined liberals, and the idea.
15:29:22 The idea of using some kind of user defined literal to create a first class citizen for unit physical unit programming, so that you can

mess up units and cause like the Mars lander incident.
15:29:36 The Air Canada game league lie to incident so many incidents in the past, it's great for security and safety of course. And I think the newspaper is stalled because he's been asked to survey every buddy else's techniques which are like template techniques
15:29:52 runtime techniques.
15:29:54 And he's just kind of getting stuck. So, the directions good actually talked a little bit about this too because we kind of really want this, and we wanted to progress, but he, but it's, but he matures, it is getting overwhelmed by by people asking him
15:30:08 to to get a state of the art so
15:30:15 I wonder if he just needs more.
15:30:18 If he needs more bodies. I think he should just ignore the request for the state of the art survey and just get down to designing his paper I entire paper, and I couldn't figure out what he was proposing even though I knew exactly what he was proposing,
15:30:33 so they can tell from the paperwork this joint, because he had 20 pages, talking about other people's proposals, which was we've already decided we're not going to do units as a template.
15:30:43 Because that's too hard to read. Right.
15:30:57 Do it as a runtime quantity, because that's really hard to detect trap at compile time, we already know the problem of all these other techniques now. Yeah, just in literature literature review the same unnecessary certainly. He's some his implementation
15:31:04 i think is maybe where he's putting some effort, and he certainly getting some help there as well.
15:31:11 There may be a bit a little bit of a bubble there in terms of designing it might be worth him you know coming with something preliminary, just to sort of test the water in the committee again but yeah.
15:31:24 Well, we can always invite him to present.
15:31:28 Right. I mean he wouldn't have been able to present to present.
15:31:32 He probably needs. I mean, it was more of a let's let's do units was that paper. He probably now needs to adhere my units, paper, I think, maybe that would be breaking the rule of separating the design rationale but that that paper.
15:31:51 I don't think it seen any any activity for purity now.
15:31:55 Okay.
15:31:58 Michael, would you like to give them a nudge.

15:32:03 Oh, yeah.
15:32:09 talking about flights.
15:32:06 Sorry, Michael.
15:32:09 Yeah, but if you give me too much and see if he wants to see if there's any help that we can offer or if, see what see what the state of the art is sorry the stage of the paper is on the side to the other side of town is Kim.
15:32:26 Hello, can you hear me. Yeah.
15:32:28 All right. Just a quick comment on that, I think, whatever implementation he goes for it has to be faster compile. Otherwise, I don't think, at least, the things I've seen won't be using it.
15:32:42 The, the class of bugs that catches is not that big.
15:32:45 And if it explodes or compile times it's really going down the drain.
15:32:50 Yeah.
15:32:51 Yes, I'm already composing a tweets for Twitter after this.
15:32:56 I'll be sending out after this meeting about compile times because I think it's still overlooked by large swathes of the committee.
15:33:03 So, if, if this some library was just doing the checking just constraining what was done with the, the calculations, you could you could type it all out for instance floats.
15:33:16 You know, for general development.
15:33:19 If these things were women to operate exactly like the the fundamental types that they were wrapping that that's something that that I was thinking we should aim for if we have ever safe type of traps overflow.
15:33:35 Just be able to swap that out for a for a regular integer type, and then you get really fast builds but with no checking. And then you sort of maybe change the change the type def one of these super expensive steadily checked unit types.
15:33:51 And then suddenly you have just a slow build it says oh no overhead, you've made. You've gotta go to your units and interest.
15:33:58 Yeah. That's one of his design considerations.
15:34:01 Okay.
15:34:02 It's interesting.
15:34:07 Um, okay. So Michael, oh no, sorry not Michael beyond has a paper on c++ exceptions alternatives. I haven't read this paper.
15:34:18 It's yeah it's actually, it's an old paper. I right, it's, it's really him justify why exceptions still exists.

15:34:29 And it's worth a read, it's not really a proposal because he and I was talking in the direction group and we're kind of tired of the fact that there's all this proposal for, you know, there are many proposals will alternate exceptions like error and I'm
15:34:46 not blaming anybody but we are now. And the question the thing that he was trying to say was, people have potentially have forgotten why we needed exceptions.
15:34:57 And that's because you can't throw an error code in a constructor.
15:35:03 You can't throw an error code in an assignment, operate at low any operators.
15:35:08 Right.
15:35:10 And so, and there are other reasons why and it explains the in depth back history of why he designed exceptions. He's trying to say that he's seen all of these other techniques and methods before and he discussed them with Bell Labs.
15:35:27 And we're just rediscovering all the same reasons, and he's trying to point out that exception is really only useful when you have, I think, You talked about this.
15:35:39 So, correct me my numbers are he says it's like is it that if you have 1010 levels or is that 100 levels of a function call.
15:35:49 Sorry, sorry, the frequency of it coming up. Yeah, I think inbound will will be able to correct me on this one is a 10 or 100 frequencies will do you remember which one it is.
15:36:02 No, sorry.
15:36:03 Yeah. So you try to say that you were supposed to use exceptions. If you Eric and Eric happens like less than 10 times.
15:36:11 Right, right. Yeah, something like that you know I'm trying to say right and but, yeah, yeah, yeah. You shouldn't. It's really related to what are we saying not to use exceptions for, you know, things that are.
15:36:23 Yeah.
15:36:25 Sorry.
15:36:27 Yeah, cuz he did the the mechanism would overwhelm your error. Right.
15:36:32 Yeah.
15:36:33 So, so he was just mostly complaining, not complaining but trying to re inject into the world what people forgotten about why exception was an important design.
15:36:45 And I really should put in James some paper now as well to his

paper on deterministic exception for embedded systems. If people who are looking for it you can just Google that paint that name.
15:36:58 c++ deterministic exception for embedded systems, and you'll find James's paper, and I'm going to put that on this list as soon as I get off the phone right now.
15:37:10 Right. Thank you, Michael.
15:37:15 A lot of assumptions tonight isn't here.
15:37:20 I have no idea what this one's about. No, I'm not sure why this is my related to us.
15:37:28 Right. This yes so sorry for that.
15:37:33 That's a pre WG 21 kind of contracts related thing there.
15:37:39 This would you be able to inject assumptions into the code that can maybe help you optimize it produce faster code.
15:37:49 And that's probably why it's with us. Right, right.
15:37:52 Thank you.
15:37:55 Mr.
15:37:57 Thank you. I just wanted to add a small thing regarding the exceptions and error mechanisms, just that, as Michael said that it's true that are like, few different error handling mechanisms which are, which, which, at least to my opinion is good, we might
15:38:14 want to address the multiple ways of handling errors or, if we want multiple ways in which ways of handling error in the library so that would be an interesting topic to explore, regardless of adding the mechanism so so like JS proposal is moving forward.
15:38:34 First is expected, etc.
15:38:37 You might want to address that. Regardless of the of the mechanisms that exist. So if anyone wants to know, I'd love to corporate and that as well.
15:38:50 Okay.
15:38:51 I'm sure next.
15:38:54 Sure. Next, I just wanted to add for the portable assumptions, like john said I mean, once we have contracts that removes like 9095 99% of the uses for these like what you currently use the built in assume or assume macros for or up.
15:39:09 Yeah, intrinsic sport.
15:39:22 There's still a handful of cases you might use them in the middle of a function that I don't know contracts would be the right fit for that at all.

15:39:22 Sorry, sorry to interrupt you, but assumptions and assumptions.
15:39:26 I think there's
15:39:28 another paper by have actually saying that that assertion search should not be assumptions.
15:39:36 And so that's kind of like a hot topic again in, in the contract study group.
15:39:43 Yeah, I guess I was like I was like I think a lot of the uses for for assume would would be very related to contracts, but I think there's independent of that discussion I think there's still some corner cases where it can be independently useful but
15:39:58 I would assume that if we have contracts and if they do have that having all that conversation goes I guess it's one another assumptions that would really, really decrease the pressure on meeting portable assumptions and think.
15:40:10 Right. Yes, and I'd really like to see. Contrast be assumed report.
15:40:15 But I think there's there's a lot of fear around, you know, time travel up,
15:40:21 type, type concerns people that you bring up.
15:40:27 People get instantly quite concerned that lots of terrible things are going to happen to their programs and so it's quite difficult to get a, like a calm stream of thought into some of those conversations.
15:40:53 We have 20 minutes left, and there are still other things to do after this I'm going to rattle through these may return value coffee, religion,
15:41:01 No patriots but obviously that sounds magical math word from the lovely yes thunder, Sean little bit absolutely need to
15:41:13 threats constructor attributes,
15:41:17 can be changed our creation is a stack size. Do you have anyone here, Billy, you're on this one.
15:41:24 Oh, but he's left.
15:41:27 Okay,
15:41:34 high
15:41:38 priority inheritance.
15:41:41 And we have no paper for this.
15:41:48 Another new idea. Right now we have artist. Yes, it's a question, raise your data structures.
15:42:01 And
15:42:01 I don't know about these I think we start I think we're going on

to more recent papers now I can easily excited here let's.
15:42:12 Okay.
15:42:15 Yes, not strictly game or related but I can definitely see its application in our low latency projects we have stumbled upon this problem where we want to take a ratio, but the performance is way, way too large.
15:42:29 So if we could have some sort of fixed time that we pass by copying instead of by application or something like that. That wouldn't be very nice.
15:42:38 Yeah. Okay,
15:42:43 new random number generator.
15:42:44 That's just, that is the God, the random number generation is not portable across platforms.
15:42:55 Ah, involves something in the chats to find a coach when a simple performance test to go and levels deep into a call sequence and then it was an error.
15:43:04 If the air is rare say one 2001 to 10,000 and the cornerstone is deep say 100 or 1000 exception handling is much faster than explicit tests.
15:43:15 Yeah.
15:43:18 Izzy.
15:43:20 I would just like to point out that in languages like rust. The compiler is actually able to signal bubbling up between the two locations, if necessary, of like where, you know, the error state of a result is first try and found, and it will automatically
15:43:39 propagate up.
15:43:42 And using relocation it does not actually have to do any checks in between that, it may, you know, have to unwind or, you know, I'm sorry, drop lifetimes of objects and whatnot but in terms of speed it is as fast if not faster than an exception in the
15:44:03 same case that was stated by that statement.
15:44:07 Because you're not having to do a test at every level inside of every function, because every single crate is compiled against each other, and one Craig is one translation unit so even if you have 300 files with 1000 calls deep, that's all one translation
15:44:24 unit, the compiler can see that.
15:44:26 Right.
15:44:28 I've had learning rust near the top of most do this for about five years now.
15:44:32 Every time I hear something about get more interesting.

15:44:35 Maybe one day I'll be tipped over the edge.
15:44:39 I must points verification explicit so now even to chocolate yes that is nice. Yes, please.
15:44:45 Okay.
15:44:47 Rodney random number generator I don't know anything about this time.
15:44:51 Can anybody lightness.
15:44:54 We certainly need the random number generators that affordable
15:45:00 is a whole bunch of proposed.
15:45:04 Sorry john is he was first sorry No I put the hand up from earlier, thank you john.
15:45:11 Yeah, I mean I can pick up a number of some.
15:45:18 I don't know if they were generators or not but yeah there was a paper proposing things. Some generators that maybe had appealing performance characteristics.
15:45:30 Yeah.
15:45:31 Okay.
15:45:33 to. to st six.
15:45:35 The year before last.
15:45:37 Right.
15:45:39 Sure.
15:45:45 Just since there's no details here I was just wondering if they would do or to just plant the seed, like it might be useful to have noise generation, rather than just r&g in a standard release generation.
15:45:53 Yes.
15:45:55 Yeah.
15:45:57 Speaking mostly Nice, nice segue into audio. So this is my paper.
15:46:02 So, myself.
15:46:16 We just don't have the time to finish it, and the problem has become larger, more complex and more subtle than me than we ever realized.
15:46:27 So, unless somebody else wants to take us out. We are moving on to other projects.
15:46:33 Was this is disappointing.
15:46:35 It's, It's clearly just too much for us. And there are too many.
15:46:40 I think the problem possibly these businesses, specifying better.
15:46:44 As I say what all three of us are too busy to take this on into our other word.
15:46:50 There was, we're so busy that the paper that we were planning to

rise to explain why we're not doing this is not 15 months overdue.
15:46:59 So, it's.
15:47:02 Many apologies.
15:47:04 Execute executor moving forward is moving toward pipeline channels off.
15:47:16 CMOS points elastic numbers
15:47:21 like Mathias I'm so deep in the.
15:47:26 I'm, and also I think you said there with their matrices as well. I'm now going to come to terms with the c++ 20 Well, having to rethink an awful lot of the decisions that I came up with.
15:47:37 Yes, yes it was 20 really has got very big stone into a very large, very large pool.
15:47:45 And this is generally paper so that takes us to our list, and we have 13 minutes.
15:47:51 Yeah, that's the paper that people asked about, which is the one that we lost reviewed in StG 14 in in Prague.
15:47:59 So yeah, thank you for going through all this I've been making notes along the way and and updating it, so we needed that. So I'm afraid so maybe in the last five minutes we can now talk about, you know, what we what people want that they don't see on
15:48:13 this list.
15:48:15 Renee your hand.
15:48:18 Right, so I promised something earlier on about something that not on this list.
15:48:24 So I actually got this one from Twitter since I asked publicly says if anyone had any ideas, and the one thing that came up with a better support for that oriented structures.
15:48:36 You know, goes along with the layout stuff but repacking that a structure so that they, you know, but the one of the other some other languages repacking the data structure so that the members are individually members are in segment that buffers as opposed
15:48:50 to, you know, each object on next to each other.
15:48:56 Yeah.
15:48:59 Okay john. So well just in response to that that any mentioned that usually is created by reflection will so fat or two,
15:49:13 which makes sense.
15:49:21 Well, we're hoping that reflection, you know, addresses that but we won't really know. Because, I mean how far along is the generative side

of reflection going.
15:49:27 Well, I mean, yeah, I don't know what the progress is, I think, but stately described things being things are being prototype, the moment. Yes.
15:49:39 You know this stuff in compilers that you can play around with. But, as well as the, the data oriented thing.
15:49:48 Just reflection in, in game engines generally serialization and tools to allow game designers to manipulate game objects that kind of thing is traditionally somewhere where reflection solutions have been reinvented many times by different game companies,
15:50:08 often using macros.
15:50:12 And again reflection is the hope would be that reflection caters for those needs.
15:50:20 I can see how it might, but I wouldn't like to lay money on guaranteeing that it will.
15:50:27 Well, you really ought to. I mean, yeah, it's it's reflection I mean it's pretty. I think even the same word is used it, we want to, we want to know about different types, but in a, in a general way so that we can, we can present that information sterilize
15:50:43 it decentralize it expose it and graphical editors, the missing thing you need to be able to do that is to be able to talk about the shape of types.
15:50:57 Yes.
15:50:57 Yes, about that.
15:51:08 I'm sure.
15:51:02 Yes, your next couple different things about what is that I, I love the idea of reflection like exactly like john said like we implement all this from scratch and almost every game engine as I'm sure you're aware.
15:51:26 We actually do a code generation approach on the current project I'm working on and part of that is managing reflection will handle which is that we want to Koh Gen language bindings for some of these data structures similar to like when you're generating
15:51:29 Protocol Buffers or something like that, like we want a single definition of a single source of truth. That's going to spit out the c++ code all of the metadata that we use for like the editor and all that, and spit out you know bindings to other languages
15:51:42 so there's.
15:51:45 I'm a little concerned that reflections not going to solve, as

many of the things as some people think it will, especially for studios that care about multi language interoperability.
15:51:54 But the other part that I wanted to note about reflection is that just like ranges I'm really scared about the idea that we're using heavy weight libraries that have to be quote unquote interpreted by the compiler to do, potentially very simple transformations
15:52:08 like we're talking about the data oriented stuff.
15:52:11 I have as yet unfounded. But deep seated fear that reflection is going to end up being too costly, have a mechanism to use for some of the things that people are saying to use it for like the data oriented approach in time to share your concerns.
15:52:32 Great Marco
15:52:36 about the layout, I think that is be 1112 still on the table.
15:52:42 So I don't know if you want to have a look at it is that has to seven.
15:52:49 Okay.
15:52:51 We have eight minutes before we before the meeting expires.
15:52:58 So I'm going to return to the minutes, unless anybody wants to make any more points about any of these papers.
15:53:08 No.
15:53:12 I was gonna say, looking at beyond spade wrong exceptions, this looks to me like a 2530 minute readings. And I think I want to commend you all to give it a moment.
15:53:25 If you would send to these minutes you'll see where we do it, that's the that's the spreadsheet that we've been ever since this is, you can get it from the agenda today.
15:53:37 Let's tattoo the rest of these segments specific discussions, jazz, well, this is what we're doing today.
15:53:44 Next, next time it's going to be embedded programming is it Michael.
15:53:49 Yes, it is. Okay.
15:53:56 So these are always the second Tuesday. Second Wednesday, so that will be tonight. And I'm going to ask one of the embedded chairs to to chair and that might be the time to talk about the deterministic.
15:54:09 The low cost you two minutes exception for embedded systems that James right and in fact I'll try to invite James Renwick to come on and talk about that.
15:54:18 Come on. Last year in April. So, and we parked it for the whole

year so it's probably time to get him to come on again.
15:54:26 Yeah.
15:54:27 Okay, um, other proposals, no future face to face meetings we are still
15:54:35 obviously all suspended according to our parent organizations, Portland, Oregon, I believe is our next tentative meeting, or is it New York, I think, is it important to New York.
15:54:54 Portland. It is Portland. Yeah.
15:54:54 Yeah.
15:54:56 So upcoming meetings Well, Yes, for New York right any other business.
15:55:06 I don't understand why these are here, Michael, could you these stale.
15:55:13 Those are all GitHub links. I remember, so don't worry about those those are just places where people have put code.
15:55:20 So maybe you're right I should probably remove them, but that's okay.
15:55:24 We can probably just close the meeting and but thank you very much to guy Davidson for chairing a masterful job. That was very, very, it's what it was fun, and it was really informative to.
15:55:36 And I hope I've been watching the translation. They are actually particularly good, they haven't been, it actually goes back and change the words. Once it understands what you were saying, so I'm pretty impressed with it I hope, I hope we were allowed
15:55:51 to keep using this. Yeah, so so that's. This isn't a list of actions, I will. I'll contact her sequel Africa clarification. Yeah.
15:55:59 But yes, this is a great improvement.

On Fri, May 14, 2021 at 10:01 AM Michael Wong <fraggamuffin_at_[hidden]> wrote:

> HI all, if anyone saved any notes from Wednesday's call from the
> transcript, would you please forward it to me? I am trying to evaluate the
> quality. Thank you.
>
> On Wed, May 12, 2021 at 9:41 AM Michael Wong <fraggamuffin_at_[hidden]>

> wrote:
>
>> HI all, today's SG14 session will be chaired by Guy Davidson. Unless
>> there are specific papers, Guy will lead a discussion into what Games needs
>> from C++ , as we did last month on Low Latency.
>>
>> Thanks Guy.
>>
>> On Tue, May 11, 2021 at 10:48 AM Michael Wong <fraggamuffin_at_[hidden]>
>> wrote:
>>
>>> Topic: SG14 Low Latency Monthly
>>>
>>> This meeting is focused on Games.
>>>
>>> Hi,
>>>
>>> Michael Wong is inviting you to a scheduled Zoom meeting.
>>>
>>> Topic: SG14 monthly
>>> Time: 2nd Wednesdays 02:00 PM Eastern Time 1800 UTC (US and Canada)
>>> Every month on the Second Wed,
>>>
>>> June 9, 2021 02:00 PM 1900 UTC
>>>
>>> Join from PC, Mac, Linux, iOS or Android:
>>> https://iso.zoom.us/j/93151864365?pwd=aDhOcDNWd2NWdTJuT1loeXpKbTcydz09
>>> Password: 789626
>>>
>>> Or iPhone one-tap :
>>> US: +12532158782,,93151864365# or +13017158592,,93151864365#
>>> Or Telephone:
>>> Dial(for higher quality, dial a number based on your current
>>> location):
>>> US: +1 253 215 8782 or +1 301 715 8592 or +1 312 626 6799 or +1
>>> 346 248 7799 or +1 408 638 0968 or +1 646 876 9923 or +1 669 900 6833
>>> or 877 853 5247 (Toll Free)

>>> Meeting ID: 931 5186 4365
>>> Password: 789626
>>> International numbers available: https://iso.zoom.us/u/abRrVivZoD
>>>
>>> Or Skype for Business (Lync):
>>> https://iso.zoom.us/skype/93151864365
>>>
>>> Agenda:
>>>
>>> 1. Opening and introduction
>>>
>>> ISO Code of Conduct
>>> <
>>>
>>> https://isotc.iso.org/livelink/livelink?func=ll&objId=20882226&objAction=Open&nexturl=%2Flivelink%2Flivelink%3Ffunc%3Dll%26objId%3D20158641%26objAction%3Dbrowse%26viewType%3D1
>>> >
>>>
>>> ISO patent policy.
>>>
>>> https://isotc.iso.org/livelink/livelink/fetch/2000/2122/3770791/Common_Policy.htm?nodeid=6344764&vernum=-2
>>>
>>> WG21 Code of COnduct:
>>>
>>>
>>> https://isocpp.org/std/standing-documents/sd-4-wg21-practices-and-procedures
>>>
>>> 1.1 Roll call of participants
>>>
>> from Zoom report
>
>> 1.2 Adopt agenda
>>>
>>> 1.3 Approve minutes from previous meeting, and approve publishing
>>> previously approved minutes to ISOCPP.org
>>>

>>> 1.4 Action items from previous meetings
>>>
>>> 2. Main issues (125 min)
>>>
>>> 2.1 General logistics
>>>
>>> Future meeting plans
>>>
>>> Apr 14, 2020 02:00 PM ET/1900 UTC: Finance focus on low latency
>>>
>>> May 12, 2021 02:00 PM ET/1900UTC: Games
>>> June 9, 2021 02:00 PM ET/1900 UTC: Embedded
>>>
>>> 2.2 Paper reviews
>>>
>>> Compound assignment review
>>>
>>> Low Latency topic brainstorm
>>>
>>> Freestanding?
>>>
>>> 2.2.1 any other proposal for reviews?
>>>
>>> SG14/SG19 features/issues/defects:
>>>
>>>
>>> https://docs.google.com/spreadsheets/d/1JnUJBO72QVURttkKr7gn0_WjP--P0vAne8JBfzbRiy0/edit#gid=0
>>>
>>> 2.3 Domain-specific discussions
>>>
>>> 2.3.1 SIG chairs
>>>
>>> - Embedded Programming chairs: Ben Craig, Wouter van Ooijen and Odin
>>> Holmes, John McFarlane
>>>
>>> - Financial/Trading chairs: Staffan TjernstrÃƒÆ’Ã†â€™Ãƒâ€šÃ‚Â¶m, Carl
>>> Cooke, Neal

>>> Horlock,
>>> Mateusz Pusz, Clay Trychta,
>>> - Games chairs: Rene Riviera, Guy Davidson and Paul Hampson
>>> - Linear Algebra chairs: Bob Steagall, Mark Hoemmen, Guy Davidson
>>>
>>> 2.4 Other Papers and proposals
>>>
>>> 2.5 Future F2F meetings:
>>>
>>> 2.6 future C++ Standard meetings:
>>> https://isocpp.org/std/meetings-and-participation/upcoming-meetings
>>>
>>> -
>>>
>>> 3. Any other business
>>> Reflector
>>> https://lists.isocpp.org/mailman/listinfo.cgi/sg14
>>> As well as look through papers marked "SG14" in recent standards
>>> committee
>>> paper mailings:
>>> http://open-std.org/jtc1/sc22/wg21/docs/papers/2015/
>>> http://open-std.org/jtc1/sc22/wg21/docs/papers/2016/
>>>
>>> Code and proposal Staging area
>>> https://github.com/WG21-SG14/SG14
>>> 4. Review
>>>
>>> 4.1 Review and approve resolutions and issues [e.g., changes to SG's
>>> working draft]
>>>
>>> 4.2 Review action items (5 min)
>>>
>>> 5. Closing process
>>>
>>> 5.1 Establish next agenda
>>>
>>> 5.2 Future meeting
>>>

>>>
>>> June 9, 2021 02:00 PM 1900 UTC
>>>
>>> Kind Rgds

Minutes for 2021/06/09 SG14 Conference Call

Thank you to John MacFarlane for chairing the meeting. You did a great job. Here are the notes and attendees:
Attendees:
[image: image.png]
13:51:16 I can see the transcript.
13:51:50 Hi, Paul, How you doing.
13:51:55 Hi, I'm fine.
13:51:58 I'm, I'm a, I'm a bit sad that I seem to have missed the,
13:52:07 the SG or the WG 121 meeting this Monday.
13:52:16 Oh that's.
13:52:18 Don't worry, that's not much really happens there.
13:52:34 Because that's mostly just a plenary. There's no technical, it's just it's just a bunch of straw polls and vote that it's pretty much already been pre decided, you want to see drama. That's about the only thing you might see see there.
13:52:57 Some people even skip the plenary This is the plenary that would normally happen on the Saturday morning, I think, of the opposite a week of work, just to find out what votes are going through and because when we, because that didn't happen there's no
13:52:55 face to face their own that huge amount of votes that happened and most of them most votes at this early stage and not controversial.
13:53:04 Okay, let's just really annoyed with myself for not being aware of it. Well, you won't dumb.
13:53:14 Well, you can still do it again because I suspect we will still be virtual, by the time they do the October one, there's another one in October. Just make sure you put a calendar.
13:53:26 And then, not miss it.
13:53:27 It was, It was mostly for the deprecating a paper.
13:53:34 I thought that was where I was supposed to, you know, defended or whatever you might say, or defending on these calls on those virtual on those plenary calls they just they just very fast set of votes that happens, and a bunch of admin stuff.
13:53:53 Yeah, if you want to see how the how the other machinery imports work, then you can certainly be.
13:53:59 And I can send some people interested in calling in and seeing what actually happens and then you can just see it.
13:54:09 Never. Just do it on the next call which is October.
13:54:09 Okay.
13:54:11 I never went to a meeting. I know this is your chance so if you

want to go to one of these when they go back to the physical, because you know it's going to be money and a lot of things and I think we're going to stay virtual at least for the October
13:54:27 one, and then maybe 2022 February one will be fifth will be physical.
13:54:35 So I don't know if you know about the October ok so the one in May was one where things actually got decided and this was just a plenary.
13:54:45 I don't know which one in may have talked about there's so many of them.
13:54:48 Yeah, okay, they're always meetings going on right but they're not the one that decides things, they just things that they just mean that are preparing things to be decided.
13:54:57 Okay so nothing's getting decided at all or. Pretty much, yeah, there's a few things that's decided like you know things are creeping into c++ 23 and things like that but big decisions are hard to make when everyone's is virtual.
13:55:13 Okay, and I'm a lot less work. Yeah, but ya know I do encourage you, if you have access to getting on the call on one of the plenary, go ahead and do it.
13:55:23 And this is while it's still virtual and it's free.
13:55:29 The most the general advice is just, just look and don't say anything unless you're really sure what you're talking about.
13:55:39 Okay, thanks.
13:55:41 You can find all the October when I'm sure on the ICP website.
13:55:48 Yeah, that's the, that's where I found out that I missed the one this month. Yeah, so don't worry the October one was definitely still be virtual.
13:55:58 And then maybe February will be Portland and then, and then we start again.
13:56:03 We'll see what happens. How, how you guys doing are you guys I'm outta lockdown or.
13:56:12 Well Denmark had it pretty tough. In February, so we've had a pretty easy. Since then, and we're quite, quite careful when it comes to all of us locked down and yeah so.
13:56:30 So things have been better in Denmark, then for what I see in the rest of Europe. Okay, and lockdown is almost non existent and everybody's getting vaccines and in all of Europe.
13:56:42 So, okay, hopefully,
13:56:51 that goes, is we just left the longest lockdown in Europe, just a couple of weeks ago.
13:56:53 And more and more people getting, which country is that Peter,

Ireland. Ireland. Ireland. Yeah, yeah, yeah so hopefully it's gonna change maybe next couple of weeks but it's been tough.
13:57:09 They actually opened any restaurants that are shops are open just last Monday,
13:57:17 since Christmas, so that's long whereabouts in Ireland.
13:57:29 Any. I'm very i'm john. Yeah, no, I mean Intel.
13:57:34 Gosh. It was me with john, That's how we met with in drag.
13:57:39 Right. Okay, okay.
13:57:41 I'm coming I'm well I'm happy to miss my, my wife's, my father in law's 90th birthday in a few weeks time and we're hoping to visit. But obviously, but he's in Waterford in the south.
13:57:54 And, Yes, giving us a while and spring problematic.
13:58:04 So yeah, we are also in the longest lockdown in North North America, Canada is Ontario anyway where I am, it's been just now beginning to get out of it, because I'm.
13:58:17 We have also been locked down since December.
13:58:26 So today we're going to be doing embedded. And I believe john McFarlane is going to call in, I hope, and he's going to lead this, she will. I spoke with him today.
13:58:38 He will. He's planning anyway.
13:58:40 Okay, so you know john McFarlane.
13:58:44 He's my neighbor actually even
13:58:49 in the building or in the house. No, no, it's kind of we're in the same town so in the in the in the context of this meeting we are neighbors.
13:59:01 That's great.
13:59:02 Well, I hope, Sean can make it if not, I'm going to have to drop Ben sacks to do to chairing for this because this is the today's the embedded session.
13:59:11 And I'm hoping that James Renwick will call in and we can sort out a few things that we need to do with his paper.
13:59:25 And then while we're at it, Ben can also you can announce the CPP con
13:59:35 or put opening up for talks and things like that.
13:59:38 Yeah, that that was mainly why I was making sure to, to be here today.
13:59:44 Right.
13:59:45 I'm interested in the, in the papers and so forth. but yeah yeah that's the sort of the primary thing on my mind.
13:59:54 If James Renwick doesn't come, okay.
13:59:59 john john thank you very much for calling in. All right, no problem. Yeah, you are the chair for the embedded for today. I'm actually

in a bit of a quandary because I have to give a talk about some indeterminate time in the next two hours waiting behind
14:00:14 I'm waiting behind a long line of speakers so I will definitely be just least monitoring and paying some amount of attention. So James just got in so that's great.
14:00:24 So the only thing I wanted to do is, did you want to bring up the agenda so you can follow it.
14:00:31 JOHN.
14:00:33 So welcome everyone this is the SG 14 monthly call, which is almost, which is almost always the second week second Wednesday of every month, we rotate between embedded games and finance and I believe next one was back to finance, and low latency.
14:00:52 And I got an I'm hoping that call cook will be coming back he said he couldn't make it this week, but he'll be back next month to talk about low latency he's one of the, the, the guys really know a lot about it so
14:01:07 today we're going to have a talk about a go back to have another discussion about where we are with the, with the paper, I believe that James is working on I think one of the things we have to do is package it up for the standard submission so so that
14:01:21 it can be ready for so that we can put it in a paper form for standards mailing, so I'll, I'll. I'm going to hand it over to john McFarlane, one of the the special interest group chairs for for embedded, and he can walk through the agenda and then I've
14:01:42 tried to chime in whenever I'm not need to give a talk on the other side of this call.
14:01:48 So john, you have the co chair and I'm going to hand it over to you. So thank you very much, first of all for volunteering and help and helping to chair this.
14:01:59 No problem.
14:02:02 By the.
14:02:02 I should point out I'm not really an embedded expert. I'm not technically an embedded developer.
14:02:11 But I, I certainly work alongside embedded developers and. Yep.
14:02:16 So please, you know guide guide me as we go along.
14:02:22 Michael Did you see that the, the author was here, I forgot the right paper by the way this is the link yes he's here, but yeah so it's all good.
14:02:33 Follow the agenda, there's a few items we probably have to go to like CPP con announcements and any future so meetings and things like that.
14:02:39 Okay.
14:02:41 And this is the right, I've got the race, gender here. Yes, this

is the one, you don't have to do roll call because I will get the roll call from the zoom on the client.
14:02:51 Okay, and I'm going to try. You don't need to take notes, unless someone wants to but I'm going to try to take it from the live transcript that's going on right now.
14:02:59 Okay, sounds good.
14:03:01 And I noticed it even spelled My name right there so it's coming along leaps and bounds.
14:03:07 Okay, well, that is a big step was named. Yes.
14:03:12 Yeah Hi everyone.
14:03:16 Welcome to the SG 14.
14:03:18 If. Hopefully you received this email with the agenda in it.
14:03:23 The first thing I want to do is draw your attention to the various standing documents related to WG 21 or so, particularly the ISO code of conduct.
14:03:36 And I guess.
14:03:39 Let me know if you don't have a copy of this by the way.
14:03:42 And so okay so we're going to skip the role of participants and is everybody okay with the agenda.
14:03:54 I guess I need somebody to put forward the agenda or or second that yes toes.
14:04:06 Okay.
14:04:04 I go, I'll propose propose okay.
14:04:10 Do we need a second. Yes we do. Okay.
14:04:14 Can I second it.
14:04:16 You may not show the chat any other exempting may indeed Second, the agenda second did.
14:04:21 Okay. Thanks man. All right.
14:04:26 So, right.
14:04:27 Yes, the future meeting so so this is June 9.
14:04:33 I assume everybody can see my screen there, followed by in a month's time, we'll have finance, low latency and.
14:04:42 And then in August, will circle back to games. These all appear to be the same time slot as this one.
14:04:52 So, Michael mentioned upcoming meetings, so I know that the, the, this being the virtual plenary was that this week.
14:05:03 Apart from that, I, I guess there are some calls out for various conferences and STP con and meetings, c++ made some announcements recently also embedded meeting plus, or is it meeting, plus plus embedded.
14:05:27 I think also is possibly looking for people to give talks at the moment.

14:05:34 And
14:05:37 it is Can anybody else think of any events coming up soon, that are worth that are of interest or SG 14.
14:05:47 I can't think of any other ones but I will just chime in. And as the embedded trek chair for CPP con. This year, and say that I, I did send out an email about this yesterday so if you're on the SG 14 mailing list, you should have seen an announcement
14:06:08 about that. And, and that contains some links about where you can get more information, ask questions. And if you're if you're being great about this sign up for to do things like review talks for the embedded track.
14:06:28 If you're, if you're up for that.
14:06:30 So, uh, yeah thanks to everyone who does that if you have any questions, you can reach out to me or you can email embedded underscore track at CPP con.org.
14:06:43 Okay. Thanks, Ben. I believe it's going to be a hybrid conference this year, some people attending in person and some online as well.
14:06:52 Interesting.
14:06:53 Okay.
14:06:55 Oh Is there going to be an sg 14 meeting at CPP con.
14:07:00 Any ideas.
14:07:03 Um, I guess that's me I'm not hosting it yet. Unless it's because we seem to have so many of it because I don't know if I will be there physically.
14:07:15 Okay, I guess, watch this space.
14:07:18 Okay. Anything else before we jump into the paper review. Yes.
14:07:25 Okay.
14:07:27 Regarding this you 14 meeting I should remark that I will, I am very likely to be there physically and I'm happy to chair an sg 14 meeting if there is sufficient interest.
14:07:36 Okay, That's great.
14:07:39 can be there. They can probably help chair this then. Yeah.
14:07:43 And it might even be interesting to try and hold it in a hybrid fashion.
14:07:47 Because I imagine all the equipment will be in place for hybrid meetings to take place.
14:07:55 It's all about having good audio equipment.
14:08:15 It really really really really is. Okay. We should make sure to check on that because my understanding was that part of the reason we didn't have one at CPP con last year was because we legally couldn't that because

14:08:15 insights and the other associated organizations, had declared that there would be no in person meetings for the duration of 2020 at that point.
14:08:29 I thought that we, We actually kind of couldn't do that.
14:08:34 But okay, any something to check on sounds like something we might need clarification maybe from from herb or somebody who is, yes. So I think the thing is if I saw an insight is still banning meetings, then have this face to face as to 14 would fall
14:08:50 into that. And I think that's what Ben is saying, yeah.
14:08:56 And that's slightly to, to be continuing until the end of the year.
14:09:01 I don't know what they're there now current rolling band is, is now up to. Okay.
14:09:08 Yeah, it was to the end of the year as far as I know. Okay, and although they think they did change it to them, you may or. It wasn't a strict, you don't have, you can't last like okay well so one to keep an eye on then.
14:09:26 Maybe there's some alternative format or something that that where maybe it's not strictly a formal, you know, study group meeting.
14:09:39 I don't know, but I believe that have actually does have discretion in this regard, so we should probably raise it with.
14:09:45 There was not right padding nice right i think there is discretion available.
14:09:50 All right. You've had us to the correctly that are two dates actually one from my soul, and one from some unity United States structure and the American one is much longer time.
14:10:06 So I so could have meetings before that, but the other Association could not so they're basically bad for both.
14:10:16 Okay.
14:10:18 I definitely want for her by the sounds of it.
14:10:24 Does somebody want to just shoot him an email perhaps and ask about that. I guess I'll take that. Thank you. Okay, thanks.
14:10:39 Right.
14:10:34 Anything else.
14:10:38 Any other topics. Okay.
14:10:41 Alright so we're looking at
14:10:46 this paper.
14:10:49 And this is said, James present.
14:10:55 Today, I am welcome James.
14:10:59 Hello, good morning and evening, whichever one.
14:11:03 Yes, should say my name is terrible, so apologies in advance.
14:11:10 Your, your connection is terrible.
14:11:14 Yes. Okay. Well, it's playing up today, apparently.

14:11:19 I was going to ask if you wanted to share your screen or whether you're happy for me to scroll through it as perhaps you could try and just basically give a low down on the paper
14:11:36 that Yeah, sure. I think the easier, you share your screen.
14:11:40 Okay.
14:11:51 All right, just tell tell me what if I mean what page to go to
14:11:48 begin.
14:11:48 I think you might some some members who may recall, I think it was last year, we brought this up as well.
14:11:58 In essence, this is about the question of in 2021 hour later, with compilers better doing optimization. Is it possible to implement exceptions, the normal way without requiring any special side channel ways of stack unwinding, basically, can we actually
14:12:20 use the return channel, can we pass information back from functions via additional parameters.
14:12:29 In this case, or, indeed potentially via other API changes that could contain extra state information.
14:12:37 And in so doing, can we re implement sections in a way that is more deterministic is much simpler is much smaller. Really.
14:12:47 The idea behind this concept. And basically I create made a prototype of this, including a number of years ago, and demonstrated that, at least at the time.
14:12:57 Yeah, Absolutely. It was very possible.
14:13:01 And it, the results in very prompt.
14:13:05 In a nutshell.
14:13:12 If you'd like me to go into details I can.
14:13:15 Yes, I mean, I guess if anybody has any questions, I think Michael has his hand up at the moment. Yeah, so for some people this is a retread because we did talk about this paper both at the prod meeting, and then a month later last year in 2020 probably
14:13:30 in April.
14:13:32 And since then, you know, the pandemic has different effects on different people and it's all understandable. So, we haven't done anything about this because you know obviously events overtake whatever.
14:13:43 So I think the thing to decide here there's no problem going through this paper again for some of you I'm sure it's useful.
14:13:51 And the thing I think I'm hoping that we come to is a two things I believe one was to repackage this paper possibly with any updates or changes as a and resubmit as a standard paper.
14:14:05 It's already been published so that steady state itself shouldn't be too much of a problem, so that it's out there as a p paper. And the

other thing is, of course, the last thing we did from the last time was trying to cultivate a group to help with the
14:14:20 GitHub changes that could be used to experiment with this paper.
14:14:26 So I think those are the only thing that I'm looking for. In that way, it doesn't hurt to go through this paper again, and if it doesn't annoy anybody.
14:14:33 I'm happy, I'm, I love to have a haven't done again so that's it that's why I wanted to say thanks.
14:14:40 Okay, thanks Michael. Yes.
14:14:43 Yeah, I'd certainly appreciate
14:14:48 a talk about this and if anybody has any questions, please raise your hands as we go along, if that's okay James Can you. Yeah, would you like to.
14:15:00 Yeah, sure thing. Let me know if my voice breaks up completely. Okay. Okay. yeah And I'll just repeat what I've said.
14:15:11 So I think the best thing to do really is just a scroll to the diagrams and find the first diagram.
14:15:20 Yeah, I was just going to ask.
14:15:23 Michael mentioned to that this, we were talking about making this AP paper, and I was curious that since this is describing a possible implementation, I'm assuming something that we wouldn't want to make normative text in the standard or things like that.
14:15:41 What would this look like as a, as a p paper Can you talk a little bit about what that might mean in this case,
14:15:53 I can answer but Jen's may already have that answer so he's has. Yeah.
14:15:57 Yeah.
14:15:59 Hi. So, um, I think, first of all, I don't think it is useful to repackage a sort of scientific paper that was meant for that sort of format, as a paper proceed.
14:16:21 P papers are, in principle, papers that describe desired changes to the c++ standard.
14:16:27 And it's a journey to get there to actually get the change approved.
14:16:32 It's certainly a journey of a year or so. But that's the, if you wish, the sole purpose of a pea number of paper
14:16:41 that we know that certain changes to the c++ language are are described.
14:16:48 Such a or to the library such as you know add new library facilities or change the core voting to introduce a new keyword for Medicaid or whatever. And I'm not sufficiently familiar with the paper on

the plate.
14:17:02 What changes, exactly, it would it would describe it would it would want so far.
14:17:11 If all it does is describe a new way of implementing exceptions. Within the, the framework of existing syntax and semantics of c++ exception so it's purely an implementation choice to do certain things different differently, then there is nothing that
14:17:34 comes from this paper in terms of changes to the language standard. All you can do is walk around compiler and mentors and tell them that you are a customer of them, and that you would feel that their compiler would be much better if they would be doing
14:17:49 if they would be doing exceptions your way as opposed to the old way.
14:17:54 That's why we have a standard multiple implementations are possible.
14:17:57 If however your paper proposes to change the semantics of exceptions in c++ to enable a more efficient implementation in some area, then, then we need to then that needs to go through the, you know, it was a committee and the committee needs to form an
14:18:16 opinion on whether the change in an exception semantics, in this at that corner case is justified or and whether they want it, or whether they don't want it.
14:18:29 And that's what we do.
14:18:33 Yeah, I speak to both of those.
14:18:35 The first thing is yes it does propose both it proposes a change out there's three elements to this. The first thing is there's still some unresolved stuff to do with API's and keyboard, like for example her current proposals on value exceptions.
14:18:53 So there's some kind of tie in there, but I think that's kind of some way secondary, there's a proposed change to some of the handling for the, the gardening session object, I forget I forget exactly what it's called because long afraid.
14:19:09 There's a, there's a there's a function within the standard library allows you to get the current pointer, it kind of transparent.
14:19:17 Even pointer to the current exception. Yes. And the way that that is worded, and it will have this discussion last time as well. My belief is the way that that is word it makes it very difficult exception to not exist on the heap.
14:19:37 Basically awesome heap like structure so if you stack. There are lots of challenges to do with managing the memory there, so there was some element. And the other thing is this proposed introduction of the new kind

of more or less, sort of implementation
14:19:45 type version of that effectively.
14:19:52 Yeah, so that needs to be. That's where we need to look from our standards perspective.
14:20:00 I think that the problem there with the problem there is I think what it what it could benefit from is more input really from various people.
14:20:10 I think that the.
14:20:12 If we have this one prototype implementation which I I'll be honest, I haven't released on the basis that I haven't listened to people who are specifically emailed me.
14:20:29 working on a separate another implementation which is a lot better. Currently, which is much easier to work with. And more importantly, has support now for aligning.
14:20:38 Every piece, basically. So you can actually keep everything on the stack for small enough exception objects, which is obviously have a great effect so while all of this, you know, unwinding quote unquote in otherwise returning is going on, we can just
14:20:52 keep everything on the stack.
14:20:54 And it said, obviously, not only apply certain things but it's much more efficient.
14:21:00 And I think much more suitable for bed.
14:21:02 So there's various things that I've wanted to do optimization wise which I've now in progress, and are showing lots of positive results.
14:21:12 And once that's done, I think the idea was to experiment and the wall that I've hit here is to do with the sea interrupt basically trying to compile a c++ standard library with things that messed around with the API in such a way that there's a, there's
14:21:41 interface, see, as you might imagine is a bit of a nightmare, the c++ library limitations are largely based off the standard library implementations. And that's kind of where there's been an issue with experimentation, so I have also been trying to figure
14:21:45 out some of these teething problems there as well before trying to get some real applications using this, basically.
14:21:53 Yeah, that's all nice and great experimentation and all that, and it certainly feeds into the decision of what to do with exceptions if anything there are other exceptions like proposals floating around for visitor has, for example.
14:22:12 So, so that certainly needs to factor into the decision at some point in the larger context, not the study group. But what to do about exceptions in general.
14:22:25 I guess sorry if I, if I may interrupt just one second to say, I

guess my point is that before wanting to make any more. How do I put this concrete decisions on the perfect API changes, wanted to have something that was more realistic, rather than just
14:22:37 some examples. That's why there's more of a holdup is why do you say a bi, or do you say API, because a part a, p i, okay, because a bi standard standard library, because a bi binary interface.
14:22:56 We are officially not really concerned with except that that we are because if we're breaking large quantities of API then we we get that night sleep but but from a standards perspective, API doesn't exist.
14:23:12 So, Yeah, I guess what I wanted before making serious proposals. I wanted to be able to say, this is just more than a silly thing, and that there's actually some potential real world here and here it's a real world use case and a real world application.
14:23:25 And that's a bit more tricky when there's still these some of these teething problems with the standard library and some such.
14:23:33 So that's I guess why the. There are things in this paper, and there are things have been hinted at other things are in progress.
14:23:41 So, yes, that's the motivation line.
14:23:46 If there are specific problems with a specification of exception under bar PR, for example, that make it unfriendly with your proposal, or was it was the implementation strategy that you're pursuing here, then, then, then, that is certainly something
14:24:03 I would like to hear about rather specific terms.
14:24:27 Okay.
14:24:19 Is there any particular part of the paper that we should
14:24:22 go to now.
14:24:27 So I think this is just talking about existing implementations. So I think we could try going into the next set of diagrams.
14:24:37 Okay, basically trying to find one that gives a good
14:24:41 overview of what the proposal looks like, Okay, well, stop me when you get to the point.
14:24:50 Oh yeah, there you go. There you go, is a good comparison.
14:24:53 Okay.
14:24:59 Yeah.
14:25:00 Sorry I'm getting a bit like there are three, three different things. Yeah,
14:25:16 the challenges of remote.
14:25:19 Hopefully that's rolling out. Excellent. Yes.
14:25:23 Yeah.
14:25:24 Good. Yes, exactly. So, in other words the general idea is that the each kind of introduce a concept of effectively exception context, and

that would be some basically anything that is, no, no, except for the main function, or even when you add a new
14:25:43 block, you have this new exception prospectively an area, a kind of concept that describes where we will allocate some space on the stack to store the potential exception that might be thrown.
14:25:59 An exception context propagates up the stack viral pointer, somehow, now that could be by register, it could be viral function parameter here it's by or a function parameter you might understand that, for example, I think.
14:26:13 Host proposal suggest using registers for doing something similar except, just with, you know, a kind of a peek pointer.
14:26:22 This this is this is similar except I use a parameter purely some implementation could be in period entirely in the front end of the compiler.
14:26:32 The general concept is having some space on the stack, into which to allocate some exception state, and that basically tracks exception that may or may not be thrown and contains a pointer to the exception data, and the general idea behind this is that
14:26:46 we can keep the existing syntax and we keep the same kind of standard library, and we don't necessarily have to change the implementations, there are a couple of points, which I think I mentioned, but a lot of them are kind of teething issues, not necessarily
14:27:04 breaking changes. And basically by keeping everything on the stack and by doing using normal function returning, instead of doing start
14:27:16 winding, we can, it's a lot simpler.
14:27:19 The problem of course comes with the fact that currently the way the standard library and the way that you know assumed collectively, is that the exception object lives on the heat somewhere or on a special here somewhere.
14:27:30 Whereas with this implementation. It does have it can sometimes end up in a special, something for the exception object buffer, except for the fact that it's just that the exception object itself is only moved there during the old wind process will return
14:27:45 process. So it's a temporary thing. And in many cases it's possible to determine explicitly how much space you might need. And worst case analysis basis.
14:27:56 Because this is all just standard function control flow.
14:28:00 There's nothing you know special will suck channel going on there's no register modification. It's just standard chocolate.
14:28:08 The only thing that's interesting. Aside from that, is the fact that if the exception object itself is small enough, it can be stored within the exception state.

14:28:20 And that of course is great beneficial. So we can actually just do a normal return on the exception object. So for example, if your implementation was such that you were using error codes that were just naturally be stored on the stack, and then it would
14:28:36 flag inside the exception state that indicates whether the exception is active, was a function called, which could throw returns that flag is then checked.
14:28:47 And once upon a time that would have been a major deal. And in many cases today. And a lot of the investigation that I did as part of this paper showed.
14:28:57 Actually, it's not such a big deal anymore, to have to have this function returns.
14:29:01 And then he said,
14:29:06 it actually opens up lots of natural compiler optimization here, purely by standard optimization passes.
14:29:15 So, yeah, I think there are some details in terms of implementation going down here and I, there's an example of what the exception state data structure look like.
14:29:30 If you scroll down, you can see there the coach him age number.
14:29:51 Outside, yes. Yeah, that right there. Yeah, I think it made it shows a new kind of exception context so there's an exception t being put on the staff that they represent the exception state.
14:29:52 And that's effectively what's going on behind the scenes.
14:29:56 And if you go to the next page, I think.
14:30:02 Okay.
14:30:11 life away from like crumbling yeah so so that gives an example of the sort of things that are stored.
14:30:18 Alongside the exception state that basically appointed to the pointer to the runtime type information, or it directly embedded things like the size the exception objects and whether it's active and so on so forth, and this is stuff that's been iterated
14:30:31 on sense, though, I have a number of different implementations of this to kind of do a kind of cost benefit analysis for different layouts of destruction.
14:30:41 The general idea though is that it should be as small as possible, and it sits on the stack symbolize really.
14:30:53 Obviously the biggest, the biggest problem with this or the big kind of the interesting part about this is it draws a distinction, a concrete distinction between c++ functions and C functions in that it changes the expected API.
14:31:08 So either way, no matter how right if it's by register if it's by

you know explicit parameter or, rather, implicit function parameter injection, then there is a difference now between C functions and c++ functions, whereas before that wasn't.
14:31:23 And it turns out that one theory this should be okay compilers make lots of assumptions and in any place we, you know, things like function pointers for example, it's.
14:31:33 Many, I found that clangs implementation at least at a time when I was looking at this does not actually ascribe whether a function pointer could be pointing is declared within a c++ kind of language context, or whether it's declared within a C language
14:31:50 context.
14:31:51 So we don't actually have information about things like function pointers. So, the kind of interfacing between C and c++ become very tricky to deal with.
14:31:59 And I think that's why we have this these keywords like like throws, for example, which can may indicate the change and make working between C and c++, more realistic.
14:32:12 I think when I brought this up last year.
14:32:15 Basically what we said was, we should not think too much about the see element but I.
14:32:22 My concern is that for real world applications that's a real thing. And mostly in terms of the fact that people might need two implementations of the sea standard library, without see standard by which I guess is something that herb is finding, or has
14:32:37 has found and is pursuing right if what I remember. Recently reading is correct.
14:32:51 I just going to Jen says his hand up.
14:32:48 Yeah, hi. So, instead of a new decoration, there is a way to differentiate c functions from c++ functions and that method is called language linkage. And so, all the C library functions should be decorated by x y and z, and c++ functions are on decorate.
14:33:10 And I understand that there are a lot of competitors out there that do not differentiate. For example, the function pointer type between C language functions and c++ language limited functions.
14:33:24 but some implementations do and rightfully do so, and it would behoove, everybody else to give a hard error. If people confuse function types. When forming pointers to functions and similar like that language language was exactly invented for different
14:33:41 calling conventions for different programming languages and to allow some sort of interoperability for C and c++ in space, in particular, there was no difference in in calling convention so far so compilers have

been negligent on enforcing the boundary.
14:33:57 But, but, but yeah, that's, that's what we have and compilers doesn't don't implement that properly is not an excuse for not at least, saying that this is the way to go, and go from there.
14:34:15 I mean if, in some way.
14:34:18 The second aspect of exception throwing is propagating exceptions, through the very few c functions that actually take a call back which is research and research I think doesn't work as well to certain cues or Does, does, very much so.
14:34:35 User takes a competitor, and that compared to my throat. And then what do you do with the ccq sort function, but that is a very much smaller concern than all the rest of Jewish.
14:34:53 I guess I find it. The problem is that in practice again that one kind of hits up against that, if you're if you're trying to actually prove that this is useful and isn't a complete, you know, red herring.
14:35:06 You kind of have to have a working standard library implementation, and to have a working standard library foundation need to compile it with one of the, you know, common compilers, and if all the common compilers are assuming that the same call a convention
14:35:17 in practice.
14:35:30 It makes it kind of difficult so I, you know, obviously defer to your, your knowledge on this matter but in terms of the practical reality it's very challenging at current at, you know, currently in order to actually derive anything useful is that, you
14:35:35 know, going forward.
14:35:38 That's fine.
14:35:39 From an experimentation standpoint, I'm just saying that a proposal that duplicates language linkage functionality was in as something new is not
14:35:50 probably not helpful.
14:35:53 Right.
14:35:55 I understand your point.
14:35:57 Okay.
14:36:02 Any other questions. Keep going.
14:36:05 Yeah, carry on.
14:36:08 Yes, I'm trying to think if there's anything else in particular to mentioned To be honest,
14:36:18 I don't think so. I think the only interesting thing is this this this, there are plenty obviously of interesting elements to this in, including handling multiple exceptions and handling propagation where the exception state has to then be copied back.

14:36:35 But generally, you know again we've gotten pretty good at dealing with problems like this, you know, we've been past you know returning structures for very long time now, and the kind of the, I think the performance results that I took speak for themselves.
14:36:50 I wish they had been more thorough and that's definitely part of this project in getting more variants and getting more testing and using real world stuff.
14:37:00 And again, one of the reasons why I didn't release my existing implementation, to the world as it were.
14:37:07 Because the reality is it doesn't, it's very difficult to get it working with existing programs. I think that's the point. So you can't just drop it in and then use it because of the standard library situation.
14:37:21 But in general. Turns out, cutting out all the unwind library is a massive size when as a one off.
14:37:29 And I have indeed made great strides. I can say, so far, I don't have the finished results.
14:37:34 But I haven't even made great strides in putting down the instruction cost of using these as well. Per, you know, function call and PR can throw expression, and so on, so on arm on x86 there's a massive difference because we don't we don't seem to mind
14:37:51 so much that the unwind library is huge. On arm of course there's a lot more effort put into it.
14:37:58 And did you see a significant reduction for smaller systems but it's still a one time, hit kind of thing.
14:38:12 In addition to that there are changes to RTI, which are proposed, and obviously are, how do I put this.
14:38:15 There are ways of doing RTI which can be made easier by this proposal, which, in order to actually reduce their size. I guess is my point.
14:38:27 And as a result, effectively you see both the reduction in terms of one time, you know removal of the online library, but also a reduction per kind of individual exception object used, and what would you say, kind of catch handler.
14:38:49 So, with the original exceptions there's various different types of RTI and methods and other kind of necessary Glue Code, which doesn't have to exist in my implementations.
14:39:02 Instead, it's just the standard stack return mechanism. Right.
14:39:07 And in terms of additional ICT I, there's very little that actually needs to be kept or retained specifically for exceptions, or we need to actually do is have some mechanism of comparing such that we can do, you know, catch block filtering.

14:39:22 So, with the ITT I even smaller.
14:39:28 We can actually cannot do it a like for like comparison in a way where we have more than just a one time reduction in size, but also as the binary grows.
14:39:39 There's also a chance to have a reduced binary size if you see what I mean.
14:39:44 On x86.
14:39:53 Start yeah I just say pulls out his hand up a little while ago. I actually didn't find this interesting, the one thing that struck me when I read this paper I guess about a year ago now, was to explicitly call out a place where the.
14:40:08 This approach has some failings in regards to how the exceptions are currently being used, I don't remember if it was handling multiple exceptions, or what it was really throwing something like that.
14:40:27 But it's explicitly called out in your paper on one of the fan ish pages.
14:40:36 Is that has that been fixed or addressed in some of these new work, because that really seemed like a showstopper to me at the time.
14:40:47 Yes, I do remember this I think this was effectively taking me.
14:40:53 What this was as I recall, was being able to re throw or grab the exception point for re throw fire a either a note set function or via another function I think it was Byron know except function.
14:41:07 So the idea is you can get the current exception pointer fire some kind of nested function call effectively.
14:41:15 Absolutely. And in fact, I think that this was discussed at length, I particularly remember a conversation with our two with with Ben at the time, and I think yes we, I think the point was that, since this has been identified.
14:41:34 It has been thought about. And there is a potential implementation which would solve this. Yes, I suspect that by the time I get to the final performance results.
14:41:44 that is definitely going to be in there.
14:41:47 Absolutely, I mean it's not, it's not a.
14:41:50 It's not an implementation or technical challenge in the same way to see interoperability is it's, it doesn't seem to me to be a showstopper. It's just something else that needs to be considered.
14:42:03 Basically, the real challenge, as far as I understand it is kind of how I put the selectively applying it, because this is kind of cost to tracking these objects and managing that which you might not always want.
14:42:18 So, I wouldn't say that it's a showstopper. But I would say that it's added additional complexity, which, as I said wasn't covered in this

paper but yes yeah, if that answers your question.
14:42:33 I like how much fun. Thanks, super vague answer I appreciate.
14:42:39 But yeah, I guess all I can say is I don't consider it a showstopper. I think it was an interesting case that was brought up last time.
14:42:53 Other Any other questions.
14:42:59 I mean, I guess we could dumb.
14:43:02 Unless you want to look more at maybe the results or any particular section, we could perhaps I don't know look a bit more, maybe what might go into AP number paper
14:43:18 or indeed whether we, whether that's the way to move ahead.
14:43:26 So yeah, there's fights, where you've paused is an example of two is the proposed additions to the standard library, I think,
14:43:42 as I recall, but I think that this plenty of, as I say, I mean this was.
14:43:49 I don't put this, this was kind of, I don't want to say made up everything's made up, but
14:43:55 I don't think this has been informed or thought about enough to justify taking too seriously. I think that would, I would personally want to consider that more detail about what makes sense, and how people would actually use this and so on and so forth.
14:44:16 Yes.
14:44:18 Yeah, so any changes to the exception mechanism.
14:44:23 The mechanism itself being a non local thing is, of course, touches all of c++ and potentially makes for example independently component libraries incompatible with each other and stuff like that.
14:44:37 So,
14:44:39 we have to tread a little careful here.
14:44:45 So what I most interested in terms of the number of paper first is of course.
14:44:55 Where did you find that your implementation approach was not able to implement the requirements that the c++ standard imposes on an implementation.
14:45:10 And one thing you mentioned, and where therefore you would want relaxation of rules or changes of rules or something.
14:45:27 And.
14:45:22 And that's, that's what what would be most interesting for a p number of paper.
14:45:28 And because that's where you want actual changes to the standards, and for example you claim that exception pointer intends to be on the heap.
14:46:03 And intends to be a shared something like a shared pointer

implementation. Yes, that was one possible implementation strategy that was
14:45:53 expected or planned for. When this facility came in, but it was also planned for that exception pointer just copies. The exception to someplace.
14:46:05 And, and it is essentially does not do reference counting but just copies the exceptional object and then it's there, and it's yours to keep. If you wish.
14:46:21 And so, if that wording was expressly done, I think Microsoft at that point needed that. When oh it's been ages when I drafted the voting for a certain point, so.
14:46:30 But if there is something more you want to do with the facility that exception pointer wants to provide ie the ability to extract, an exception from one threat and essentially be able to reach forward on another threat or stored away somewhere or something
14:46:47 like that, if, if you need more facilities, and the current allowances for Executive Order don't cover what you need to do, then I would like to know about that and very much detail.
14:47:02 Preferably, of course, and a p number paper.
14:47:11 Dude, do we think that maybe a p number paper might be a good way to solicit feedback and iterate on a design is maybe a little too much concern that that you're committing too much by writing a piece of paper and perhaps would just be a good generator
14:47:29 good forum for for solidifying some of these ideas.
14:47:37 I don't know what you mean. I mean, either. This is just presenting information to the committee, look here is an alternative implementation for exceptions.
14:47:46 We all look at it and say, Yeah, nice try it. boom. Go.
14:47:51 Next item next customer please, or, or there is some. And that is one very viable and possible result from looking at this academic exercise here, or we say we want.
14:48:04 We want an actual change to c++ and then I want to will need to understand what the changes.
14:48:11 And during the evolution of this project here I have understood that certain requirements that the c++ standard imposes in some.
14:48:20 Not very obvious ways ways have been have been looked at and have been implemented, or happy, or the author is looking into implementing those.
14:48:33 And that's good and yeah it's it's some of these requirements are a bit of a pain I understand that. And, but, and certainly the the the default handling of exceptions was this personality crap is kind of sub optimal.

14:48:50 But, yeah, whatever.
14:48:51 So yes, but but that is obviously a process so at some point probably there was a desire to remove that requirement from c++ that certain corner cases work a certain way.
14:49:03 Now that has been implemented.
14:49:06 So now we can maybe say okay yeah, it's not pretty but but it works and it probably doesn't doesn't make the whole scheme performed worse. As long as it's a lot of costs, we don't really care too much I guess so, yeah.
14:49:23 Of course it's part of the evolution here.
14:49:29 Sorry.
14:49:36 Okay, I mean.
14:49:38 So it sounds like a paper might might be helpful here then.
14:49:44 Again, if the desire, is that the c++ standard changes. That's the whole point of writing a paper. If there is no such desire, we don't need to paper.
14:49:56 And I'm sorry, we don't need to pee number paper to present research results were very capable of reading PDFs stored as well.
14:50:05 Okay.
14:50:12 Anyone else have any comments.
14:50:19 Okay so, so my understanding was that there were changes being posed here now I suppose if what we're saying is, James is uncomfortable proposing them, then maybe someone else needs to step up and champion this paper if we want to start getting some of
14:50:36 those changes reviewed by the committee.
14:50:39 But my understanding is that there are some, there are proposed changes being made here and if we.
14:50:52 And certainly this material a lot of this material would have to be in the paper as clear motivation to explain why things need to change. I would think.
14:51:05 Yeah, I can't help thinking that, yes, some of the implementation details or choices will help provide the context for understanding what potential changes to the, to the standard are being considered.
14:51:23 I think it's not the way I would word it myself is it's not that I'm uncomfortable proposing the changes, it's more that I, my concern is that Yeah, I'll be honest, I say, Be perfectionist about it.
14:51:35 I think that that's the. I wanted to have something realistic and working, which I could give some sort of backing to. Yeah, foundation to to what I was proposing.
14:51:47 And this does work.
14:51:49 To be clear, but it doesn't work with.

14:51:52 It works with example applications and it works with heavily modified standard libraries, and it doesn't work real code, so I couldn't take a real world application and then swap it out and say well this is how we would change this, so this is how people
14:52:04 use it, you know.
14:52:07 So I think that's the that was the main motivation so perhaps I guess what you're saying is, just get something out there anyway.
14:52:15 So there is a path towards a p number that involves a draft, or we call it the number, which isn't published in the, in the mailing, but can be shared around and feedback can be provided on that.
14:52:34 I always get the, the exact steps on how to do this wrong, but I think we maybe want to get a number, perhaps.
14:52:43 Well, before we dig into the details.
14:52:47 There is. I mean, we have. I should remind people that we do have a multi stage process in handling changes to the c++ standard. So, we can certainly iterate on a real p number of paper, for example, in this forum, which is probably a bit more cozy than
14:53:08 some other forums that the papers eventually needs to go through
14:53:13 and improve that paper and make it better and improve the motivation and make sure the presentation is so that people think this is a great idea since the greatest idea since sliced bread and all that kind of stuff.
14:53:25 And that can very, very well be done within the P number paper frame it's it's not unheard of that the first version that is seen by the rest of the comedy is revision seven or eight have a paper, because it has been iterated on a study group for a while
14:53:40 until it was good enough.
14:53:45 So, I see no problem with with doing so. And yes, There is also the number scheme, just so that drafts between mailings are clearly identified as not being the bitwise, the, the pattern of the paper that is in the mailing so that people are not confused
14:54:04 because we want the P numbers to be unique identifiers for a specific bit pattern that happens to render as a PDF or HTML.
14:54:14 Or perhaps at least for some of the more the details the boilerplate, you know, just so that it is looking roughly, roughly like up number paper that that's, yeah, that's what we have options.
14:54:31 Certainly, but that is best done by a someone relatively familiar with the process of just getting together with the actual person writing the paper and in a one on one setting and just whipping it to the right shape.
14:54:47 Okay, perhaps somebody would like to.

14:54:53 Well, I mean, I guess we should first.
14:54:56 See you. James if you're interested in willing in pursuing a paper and then perhaps we could see if anybody would like to maybe help out of it provide some early feedback, perhaps, um, I would.
14:55:12 Sorry to interrupt again.
14:55:15 I would also like to point out that this is very unlikely to happen in the c++ 23 timeframe, just because of timing and, therefore, we're not under a particular deadline rush we are usually or not anyway.
14:55:29 So if the, if James believes that he wants to make more progress on the implementation front to have more convincing arguments and all that and maybe reconsider.
14:55:44 Some of the.
14:55:46 Oh, we need to change the language like this to make it work better, the kind of things, then there is no particular rush to get a proposal out this week or next month or something, we might as well, wait for a year until things are solid solidified more
14:56:02 if people think that it's more helpful
14:56:08 to say, Yep, no that's good feedback.
14:56:12 Okay, so I wonder if there's there's maybe any feedback that you were looking forward to help guide you in your ongoing research.
14:56:25 I think the.
14:56:27 I think I would be very interested on people's opinions about the seed problem.
14:56:32 I think that's the that's the main thing, because that continues to be it.
14:56:37 I think to me that's the biggest issue, both in terms of the implementation in terms of the theory and I understand that in theory according to the standards that you know that's that shouldn't be a problem but in practice it is and if people have any
14:56:50 kind of thoughts. in addition to.
14:56:53 How do I put it in addition to the you know the fact that it shouldn't be this way, which is a very obviously very valid and important point.
14:57:01 If there are any other opinions people have around this, I'll be very grateful. And
14:57:09 I guess, for people to generally consider the algorithm. And what that might mean in terms of their own code or how they might change their behaviors, but just as a kind of like, you know, over a coffee kind of thing.
14:57:29 I think for me the meat and bones will be being able to take real applications and playing around with those. And that's why I'm working very

hard towards that, in what little spare time I have.
14:57:39 But requiring some tax changes, such as the throws annotation for c++ functions that want this extra exception context parameter is of course, slightly incompatible with applying this to real applications because real applications will need lots of changes
14:57:58 to sprinkle throws around.
14:58:02 Excellent point yes that's why I have a flag in my implementation that defaults everything to or removes the throats the net necessity for, basically, it's only really there because we discovered that this was an issue.
14:58:15 It's not necessarily there because we think it's a good idea.
14:58:19 So implementation doesn't mandate, it
14:58:24 sounds like, okay, so.
14:58:26 Okay.
14:58:29 So just to clarify, just like in everything else in c++, if it is not marked as no throw, then it is throws.
14:58:39 This this correct actually throwing Yes. Yeah.
14:58:43 Yeah.
14:58:51 Yeah. But of course, then the challenge is identifying Well, what if it See,
14:58:58 see function conference.
14:59:04 rather wouldn't accept the API change in any case,
14:59:11 I think there was previously some discussion that it might not be such an issue in the embedded case because you might not necessarily be mixing code bases, but I feel like that's not the strongest argument.
14:59:30 Yeah I at least mix code bases all the time when I'm doing low level.
14:59:41 Anyone else have any thoughts.
14:59:46 Any questions.
14:59:50 So, I'm, I'm curious, is you were talking about one of the hurdles that you are facing is a standard library that lives up to these
15:00:07 demands that you're setting.
15:00:11 Would it help getting this moving forward, to have someone do a, an attempt at modifying the standard library for clang or GCC in a way that conforms to these requirements, or is it still just whether or not it's C or c++, Colin convention.
15:00:38 I think it's almost entirely that the latter is the fact that this, for better or worse the foundations of the c++ standard library or the city standard library.
15:00:52 So getting it. I've tried many times and fat around and made it a bit of an effort to try and make that happen. But

15:01:01 yeah, the, the truth of the matter is, is the senior issue.
15:01:09 I just noticed a couple of things in the chat Ronan a ways back asked if the exception is not only does the suggested implementation support infrastructures supports infrastructures that manage or delay or redirect exceptions, with particular interest
15:01:30 in sister. I'm not sure what see stars.
15:01:36 Does that question. I saw we don't have the context for that now does that.
15:01:43 Do you understand the question.
15:01:47 Not entirely that's an interesting idea right that
15:01:53 for managing or redirecting exceptions redirecting yes to laying I'm not sure.
15:02:01 That's a very sorry.
15:02:05 If I can chime in.
15:02:17 I'm using this system it's an infrastructure that is based on the
15:02:17 scheduling program using into multiple blocks that some something that code is not exactly that.
15:02:33 The Lamb does that are scheduled in day with continuation.
15:02:36 I think in the standards we are growing and gaining, there are some efforts in creating similar things based on continuations. I think when you're talking about continuation.
15:02:49 I mostly I want to pass centers, I want to pass the exception that that was created in one such is one block or one lambda whatever to the next sometimes in the future.
15:03:04 When the exception will accrue, it will be transferred to another block, and so forth.
15:03:10 And for this currently we are using what they are using Cisco is using the exception pointer.
15:03:20 In, is this something that can be implemented with your suggestion.
15:03:26 Absolutely, I mean the exception state is effectively is an object that is just movable, there's nothing special about it. There's no special semantics it just exists as an object that's one of the kind of nice elements of baited kind of semantically.
15:03:40 So yeah, you can just pass it around, you can move it around, you can pause it through function so you can store it somewhere and then we throw it later.
15:03:49 And the, the idea is that there's a proposed actually there's a proposed ability for you to provide your own alligator. So, while the exception object itself ends up being stored in the stack within a catch block.
15:04:05 And once it's pulled into scope. So once the unwinding is complete

it sits inside your catch block so if you if you catch by, by, elbow elbow reference.
15:04:14 Then effectively. It sits inside your cache block on the stack. And so you actually, there's a proposed extension here in this paper whereby you can actually take exception objects and say I want you to copy this and I want you to copy the exception object
15:04:28 itself to some alligator. Basically, or to some location.
15:04:34 No matter what the exceptions yourself is what type it is.
15:04:37 So I think it's actually very easy, so you can specify custom alligator to make that happen.
15:04:43 And you can you carry around, in addition to the exception object is metadata, so you know what type of Was there some sort of identifier for the catch block filtering and all of the other exception state.
15:04:55 So yeah, it's not only very possible that a very serious use case, potentially for this.
15:05:01 Thank you.
15:05:06 There was another question from Connor wasn't using thread local storage for exception state previously explored.
15:05:19 Hello.
15:05:22 I take it to the rim at large.
15:05:27 or.
15:05:28 Okay.
15:05:31 Does anybody.
15:05:34 Recall, talk about als for
15:05:39 was this in was this in relation to exceptions in embedded specific, you're
15:05:48 right, I see. Yes. I think that
15:05:55 it's possible.
15:05:56 Yes.
15:05:58 All right, just possibly decide to see a big issue.
15:06:10 That's a very interesting idea. I think that's a.
15:06:10 It would be interesting to see how the optimizer would deal with that, I think, a big kind of motivator for this was that it was standard control flow.
15:06:18 And it was just, you know, all the basic abstractions of data and functions.
15:06:23 And there was nothing necessarily needed side channel The only thing was instead of returning a value that you expected as your return type, you would get something else which was affected with the exception state, right.
15:06:36 So, this would be still side channel but it's not necessarily such

an egregious side channel as a whole kind of virtual machine doing it, you know register pokey unwind.
15:06:52 says interesting thing to try.
15:06:56 Sounds like there's a fair bit of interest in in seeing the reference implementation.
15:07:02 Yes.
15:07:05 By all means don't don't feel that something has to be perfectly formed before it's presented the committee as Jen said, plenty of iteration can occur.
15:07:15 Often proposals will be transformed quite dramatically. Sometimes, multiple times before they fully proceed.
15:07:24 But yes, I don't know if it would be helpful but there is a, an sg 14.
15:07:32 github space as well.
15:07:35 Possibly we could clone something into there if there is a.
15:07:39 If there's a repo somewhere online
15:07:46 Paul.
15:07:50 Yeah, I'm just I'm just thinking of technicalities you said that the problem is that the CC library is more or less wholesale copied into the c++ library.
15:08:05 As far as our calls CCE that the c++ standard actually dictates that everything from the C library lives in namespace std, which would make it possible to have a wrapper of the c++ over the C library to, to make a track as a buffer.
15:08:35 In order to converge colon conventions.
15:08:39 Would that be a possibility.
15:08:44 As a really good idea.
15:08:51 That's really good idea because the big problem I have is linking and effectively, a lot of platforms come with a standard library as well. So, you would end up effectively having to have to build this thing.
15:09:02 And that was something I hit somebody messaged me was interested in using my implementation and I sent it to them and I said but you know the problem with you're going to have is that you need to see standard library builds effectively but a rapper might
15:09:12 solve that nicely.
15:09:21 Of course effectively nobody uses the STDs namespace but then that's a problem for another day.
15:09:30 I do, what it's worth.
15:09:32 Yeah, I expect the, the c++ code guidelines for example probably encourage people to do that, and I certainly
15:09:41 nag people in code review.

15:09:44 To add the namespace qualification so you might be surprised.
15:09:55 Sorry, going back to the GitHub. I think that this new implementation that I'm doing, which is the effects of the old one, but what I basically done is instead of instead of implementing directly in the compiler front end as a silly university student
15:10:09 might do. I've actually implemented in a separate c plus plus, and source files, and I basically just have kind of entry points into that in the compiler instead, or to clarify it kind of, you know, almost like built ins are intrinsic or something along
15:10:27 those lines where it will actually just call out to these external symbols that as humans are implemented very much and similar kind of a big way right.
15:10:35 And that means that the implementation is a lot more direct so you're not going around the compiler from the final bits, and it's very easy to actually implement change.
15:10:45 So I think if I wanted to make a public implementation which I do it would probably be that one because I think it, it's much more presentable. And people can actually hack on it very easily and iterate with it.
15:10:55 and that's kind of the purpose of the rewrite, in that sense, and bringing it up to spec spec with a modern client as well say modern you know everything changes every year but
15:11:05 as the old, the old implementation does work.
15:11:08 But there you have to kind of put a bit of effort into get it, doing what you want, really.
15:11:14 And it's not, it's quite difficult to actually change and work with because it's distributed and embedded in the front end.
15:11:27 But certainly when I have a good when it when I haven't even vaguely working. I do have it vaguely working when I have it in a state which I think is easy for other people to kind of work with 100%, I would be happy on on sharing that there.
15:11:41 And I think if people messaged me and they'd like take a look The old implementation, I'd be happy to really start on a kind of, you know, individual basis.
15:12:03 I'm just checking yeah so yeah you have an email on the paper, just in case anybody wants to get in touch directly.
15:12:10 It might not be that email. Really sorry, that's not my fault, it's the university's. Okay.
15:12:18 Yes, because I left the university. It's now deactivated which is excellent.
15:12:25 But thankfully my co authors normally for emails to me so.

15:12:29 But if you're interested, I'm happy for someone to distribute my email address or I can write in here as well.
15:12:40 If that's appropriate to share information on, and I c++ chat.
15:12:47 I'm sure it's been done before.
15:12:50 Let's say you probably relatively safe.
15:12:54 Excellent. And then there's always the reflector as well.
15:13:03 Then if you're on the mailing list or the reflections, we call it. There's that as well.
15:13:07 Right. Yes, yes.
15:13:09 Okay, definitely some interest drummed up there, some good feedback,
15:13:17 whether any other
15:13:20 things anybody wanted to comment on or what questions to ask.
15:13:28 Daniel. Yes, the hand up.
15:13:31 Yeah.
15:13:34 I just wanted to say that definitely I think this see compatibility would be kind of a deal breaker for people in my area because we definitely mix, C and c++ C and c++ code quite a bit disturbingly large amount of mixing and did.
15:13:56 It's like a risk averse behavior you know if you if you said, Oh, you can't do that anymore, they'd be like, Well, we've used this code for 20 years and it's been an x y z release product so we want to keep using it right because we know it works.
15:14:11 So anything that were to break that would be like instant. Now we're not doing that.
15:14:18 Plus there's already the normal resistance to exceptions to their history, whereas I'd like to see more of them because I think they're very useful error handling tool.
15:14:27 So, anything we can do to cut out those instant knows is a good thing for me.
15:14:36 Yes, I yeah I totally agree and that would be my point of view as well which is why I'm so hesitant about it.
15:14:49 I think the thing I was going to say is I think like the rapper idea in general, I think this has been talked about before but I hadn't thought about in the context of the standard library.
15:14:54 But yes, so in theory you could have c++ wrappers for everything but obviously you know create a facade for your entire see interface seems like a giant pain.
15:15:07 The thing that I personally think is good about this implementation is the fact that with the cup with that kind of external allocation religion, you end up in a situation where if you are just

throwing error codes, you're just throwing error codes, with
15:15:23 some additional information for cash flow block filtering.
15:15:30 So that's a, it's a pleasant element, I think, for that helps make sense of this in the embedded space as well.
15:15:40 So you can kind of, you can kind of draw direct comparisons between this and an error, kind of error code approach. The only difference hopefully being was slightly more stack space required.
15:15:52 And the fact that, which I think also was brought up last time as a slight to break from very small embedded systems, although I have, you know, for what limited testing I've done on the smaller ones now, it hasn't been too much of an issue but maybe
15:16:03 I'm not smart enough.
15:16:07 Yes.
15:16:09 So that's, that's kind of the. So I think the big difference here is that that because the compiler is directly using this, it can do all optimizations necessary to say you know what this is, this isn't just unlikely.
15:16:20 this is called path like go away.
15:16:23 So it is effectively a better implementation of error codes with exceptions on top and maintaining the existing syntax of try and catch.
15:16:37 Okay, thanks. Any anyone else.
15:16:41 Before we move on.
15:16:46 If not, thank you very much James for the presentation.
15:16:52 I think we've got some good ideas going there.
15:16:55 And so,
15:16:58 very interesting avenue of exploration.
15:17:02 I look forward to seeing how this comes along. Thanks.
15:17:14 It's okay, it's fine. July just drop off now. I'm not.
15:17:15 Almost, 14, or job.
15:17:27 If, If you want to leave, that's fine.
15:17:32 That yes thanks again.
15:17:34 And, Yeah, good, good luck with it all.
15:17:41 Okay. Alright.
15:17:45 So, moving on.
15:17:49 Are there any other.
15:17:52 So are there any of the papers, people wanted to look at and then nothing else was I guess mentioned in ahead of the meeting, but is anything, sort of fair cropped up.
15:18:06 Very recently.
15:18:07 If not, we can just go on to domain specific discussions, Paul.
15:18:14 Yep.

15:18:17 Yeah, that was just a question.
15:18:22 If anyone who is currently here because it's an embedded talk.
15:18:29 That was not hear any of the earlier times. Regarding the deprecation of volatiles paper as any comments on that.
15:18:42 I'd like to hear them otherwise. Let's just move on.
15:18:46 Can you just remind me quickly what the number is for that.
15:18:55 I'm sorry I can't slightly this calculate. So, I don't remember numbers.
15:19:03 Okay.
15:19:06 That would be a two three to seven.
15:19:12 Oops.
15:19:15 Yep.
15:19:17 And this is one of the other kind of hot off the press papers.
15:19:22 At the moment, yeah so just so everybody's aware, this is another
15:19:27 paper, kind of reversing the decision to deprecate some of the operations related to follow tile.
15:19:37 And, yes, see Paul for any questions or comments on that.
15:19:44 Okay.
15:19:48 So, moving on. Are there any.
15:19:52 Is there any other papers anybody wanted to mention.
15:19:55 Are there any domain specific discussions, I guess we should give particular emphasis to any topics related to an embedded embedded programming.
15:20:10 Was there anything anybody wanted to bring up
15:20:20 I know Ben can't make it I'm not sure if it is here, or Odin, but.
15:20:30 Failing that, it's going to be, finance, next time around is there anybody from finance here.
15:20:38 Maybe somebody could bring up topics that we want to discuss next time.
15:20:46 Hello.
15:20:48 Yeah, it's just Jeff Smith, Social Security's. Um, I think it's my ideas I can put them on the reflector for things I've kind of my grab bag wish list stuff but nothing on lines of a paper proposal, quite yet.
15:21:04 So, okay, well um yes I mean, next month if you, if you have some some ideas or to float around or even a beginnings of a paper by all means.
15:21:18 Share them or put the suggested the they'd be put on the agenda for the, for the next meeting, that would be great.
15:21:26 Thanks john thanks.
15:21:30 How about games
15:21:37 guy here. Any, any news from the world of games.
15:21:41 Nothing here.

15:21:42 Okay. No, nothing is some is one of the big conferences was GTC or the other one coming up soon.
15:21:54 Do you see us coming up soon. Yes. Okay.
15:21:58 I've been asked to live on that one though.
15:22:02 Okay. Yeah, i mean i don't have.
15:22:05 I mean, I think it's just cdc.com, or GC gt comp I don't remember.
15:22:12 But anyway, yeah it's up. It's all virtual, I believe it's promote this time around and, yeah, we're not, we're actually not sending any one because most of the benefit for the game developer conferences, doing contacts and it's just really not possible.
15:22:29 Right, yeah, we have a few people that create some assembly we're giving talks but they're busy recording them at the moment in their homes.
15:22:38 Right.
15:22:38 Okay.
15:22:40 And then linear algebra, I believe there's a meeting tomorrow.
15:22:45 Is that right, there's an sg 19 meeting tomorrow But Bob and I have made no progress in the paper for a variety of reasons over the past few months.
15:22:55 It's still, it's still gonna happen. We're just we've, we've turned we're transitioning to
15:23:04 single matrix type rather than matrix and better type in treating a matrix as a simple specialization of a, of a matrix.
15:23:15 And I've taken a bit of us taking a bit of implementation.
15:23:19 But to was still rolling along with it.
15:23:24 Okay.
15:23:25 And I believe there's some, some free functions some statistical functions are going to get some interesting functions cropping up this Mean, Median median modal, all those kinds of things are good of discussion tomorrow I believe.
15:23:41 I'll certainly be there myself.
15:23:51 But, as I say, No, no linear algebra. Okay, in particular.
15:23:50 Hello. Hello.
15:23:58 Okay.
15:23:57 Okay, um, I believe exceptions will be one of the topics discussed, and maybe the quite the question of whether they're the right
15:24:07 method of of handling some of the runtime cases. So, so, moral exceptions if you didn't get enough discussion of exceptions. Today, I don't see that on the, I don't think that's on the agenda.
15:24:21 Okay. So 2376 was mentioned a few hours ago.
15:24:34 Oh yes.
15:24:34 And I think that's that's just a pretty good, like a recurring

conversation where where where error handling was sort of perhaps contract violation and error handling are
15:24:48 intertwined fair in the US.
15:25:03 It's really interesting two or three years before we get contracts isn't it. Yes. And until then, just frequently Yeah, every now and again.
15:25:00 People will propose an API where, if there's a if there's a bug on the call aside that an exception his throne, and then somebody has to write a paper saying no that's maybe not a good idea.
15:25:12 But, yeah.
15:25:14 Okay, contacts, please, please.
15:25:18 All right. Yes, Allah.
15:25:20 I'll pass on your request, thank you john.
15:25:24 Okay.
15:25:26 Anything else from the special interest groups.
15:25:31 Before we move on.
15:25:35 I think I have sort of covered it other papers.
15:25:41 Perhaps if there are other things going on in WG 21, that people might think are of particular interest to SG 14 but that haven't been raised already.
15:25:54 Yeah.
15:25:57 Do let us know.
15:26:00 And then, future face to face meetings I think also has been covered, probably unlikely in the near future.
15:26:10 And.
15:26:11 Yep.
15:26:15 Also, future standards meetings, again this this this agenda is maybe doesn't doesn't gel perfectly with the way that we do things but they are just continuous and ongoing.
15:26:30 These days, I guess.
15:26:32 Okay. So, moving on to any other business.
15:26:37 Is there anything else to discuss Oh cool.
15:26:44 Yeah, I'll just in the chat it says, How is progress on freestanding stuff I've been slightly out of the loop from Kim.
15:26:55 And if you look through the latest mailings you'll see that there is a, an update to the proposals regarding some of the freestanding stuff and it is moving slowly since Ben is more or less the only one writing any papers.
15:27:15 But it is moving forward.
15:27:18 And I still think the offer to pick up any of the papers described in the overview paper is open, so you can just pick a people you want to right Kim and bring it all forward.

15:27:37 Right, yes.
15:27:39 Bed would dearly love more help with that.
15:27:41 It's all, it's all going to happen but it's going to go at a quicker pace if there's more input.
15:27:45 So please feel free to get in touch with them directly, I guess.
15:27:51 If you're looking for direction on that.
15:27:53 But yes, presumably you're talking about this. This revision here.
15:27:58 Paul, when you talk about the, the latest mailing or.
15:28:04 Yep, I'm not sure yes yes but yeah, okay.
15:28:08 But that's right how mentioned the preview the other day. Okay.
15:28:12 All right.
15:28:14 Anything else, a short summary of contracts sometime, maybe 10 minutes and the next embedded meeting.
15:28:21 If someone can arrange it.
15:28:24 Okay.
15:28:26 I certainly be happy to.
15:28:30 Yes, give an update on contracts, there hasn't been an awful lot in there recently.
15:28:39 I think the last possibly was a paper.
15:28:46 See yes it's.
15:28:57 You have to remember exactly what study group is now.
15:28:54 Yeah, it's
15:28:57 okay yes I think defining contracts was the last thing to come up.
15:29:01 Regarding contracts, and that was because people were kind of talking past each other.
15:29:09 It seems that different developers have very different ideas about what contracts are. And so, Jessica very kindly put together a list of songs like a glossary list of of terms to use that aren't confusing or too controversial to help us talk about contracts
15:29:31 in a more constructive way going forward. But then I know one or two other papers were in draft form but didn't make it into the last mailing yet so not a lot on that in the last month or so but certainly this year there's been a lot of discussion which
15:29:50 is sort of helped and it's maybe not got us closer to a solid proposal but has helped us understand some of the, the differences and direction.
15:30:02 That may be affected the original decision to withdraw contracts from c++ 20.
15:30:11 That's, that's maybe all i have i but i can i can go into detail on some of the newer papers. Next time, if that would be.
15:30:20 That'd be appreciated. And I do think it is of particular interest

to to our study group, because the kind of contracts that go into the standard will greatly affect the how we think about the zero overhead principle and, and various other things to do
15:30:42 with performance and latency. So, yes, I'd be more than happy to talk about that next time.
15:30:48 Hope that hope that answers your question, Henry.
15:30:54 So going back to the agenda.
15:30:59 Anything else, And by the way, if you've got any suggestions for next time. Please just mention that in the reflector and it will be added to the to next month's agenda.
15:31:12 And it doesn't have to be.
15:31:17 Financial. I frequently trip, I Frequency Trading specific that obviously will be the, the, the main focus. Anything low latency via, we can chat about.
15:31:30 Okay, the topics, I'd like to maybe touch on, or shaving off costs for a thread local in some cases which probably has people beyond finance caring about so.
15:31:41 Okay.
15:31:43 Yes. Could you make sure to add that to suggest that in the agenda for next time, I'm sure.
15:31:54 Frank is being written up that's, you know at least passable before that point and you guys can decide it's where the further discussion or not.
15:32:01 Sure, please do that would be great.
15:32:03 Thanks.
15:32:05 From the finance side, we're always sort of scrambling for specific things to look at. So the more the merrier is my view.
15:32:18 Well, as always.
15:32:27 The. There may be some interesting stuff coming up in the, in the mailing every month, sure there'll be some new material. By the by the time we meet next.
15:32:32 Okay.
15:32:34 Alright.
15:32:51 So, let's see.
15:32:42 Did we okay this is links to whatever date now.
15:32:48 I don't know if there's been any activity in the, in the GitHub group at the moment I don't really follow that very closely.
15:32:55 But perhaps that that could be a somewhere where we share some of the ideas on the embedded exceptions.
15:33:05 prototype.
15:33:08 Okay,

15:33:13 moving on.
15:33:15 I'm not sure what I took action items there are, if any, I'm not sure if Michaels gone on to his, his presentation that he had to give.
15:33:26 But, yes, I guess.
15:33:29 Yeah, there's some good suggestions about things to talk about next time.
15:33:34 Let's, let's reply in in this thread with suggestions for those because I think nobody's taking notes.
15:33:42 As such, right now, if that seems like an okay idea.
15:33:48 And, yes, that's that's the right date right, July 14 say when it comes up here.
15:33:55 July 14 will be our next meeting, same time, same place.
15:34:01 And so unless there's anything else anybody would like to add.
15:34:10 Okay, I think we're. I think we're good.
15:34:14 All right.
15:34:21 Thank you everybody for showing up.
15:34:18 Look forward to seeing you next time, and have a good evening or morning or middle of the night.
15:34:27 Thank you for sharing. Thank you very much. Thank you. My pleasure. Yeah, see everyone knows you're right. Right. Okay. Take care.
15:34:34 Bye. Bye.

On Tue, Jun 8, 2021 at 10:05 AM Michael Wong <fraggamuffin_at_[hidden]> wrote:

> Topic: SG14 Low Latency Monthly
>
> This meeting is focused on Embedded.
>
> Hi,
>
> Michael Wong is inviting you to a scheduled Zoom meeting.
>
> Topic: SG14 monthly
> Time: 2nd Wednesdays 02:00 PM Eastern Time 1800 UTC (US and Canada)
> Every month on the Second Wed,
>
> June 9, 2021 02:00 PM 1800 UTC
>
> Join from PC, Mac, Linux, iOS or Android:
> https://iso.zoom.us/j/93151864365?pwd=aDhOcDNWd2NWdTJuT1loeXpKbTcydz09

> Password: 789626
>
> Or iPhone one-tap :
> US: +12532158782,,93151864365# or +13017158592,,93151864365#
> Or Telephone:
> Dial(for higher quality, dial a number based on your current
> location):
> US: +1 253 215 8782 or +1 301 715 8592 or +1 312 626 6799 or +1
> 346 248 7799 or +1 408 638 0968 or +1 646 876 9923 or +1 669 900 6833
> or 877 853 5247 (Toll Free)
> Meeting ID: 931 5186 4365
> Password: 789626
> International numbers available: https://iso.zoom.us/u/abRrVivZoD
>
> Or Skype for Business (Lync):
> https://iso.zoom.us/skype/93151864365
>
> Agenda:
>
> 1. Opening and introduction
>
> ISO Code of Conduct
> <
>
> https://isotc.iso.org/livelink/livelink?func=ll&objId=20882226&objAction=Open&nexturl=%2Flivelink%2Flivelink%3Ffunc%3Dll%26objId%3D20158641%26objAction%3Dbrowse%26viewType%3D1
> >
>
> ISO patent policy.
>
> https://isotc.iso.org/livelink/livelink/fetch/2000/2122/3770791/Common_Policy.htm?nodeid=6344764&vernum=-2
>
> WG21 Code of COnduct:
>
>
> https://isocpp.org/std/standing-documents/sd-4-wg21-practices-and-procedures
>
> 1.1 Roll call of participants
>

> 1.2 Adopt agenda
>
> 1.3 Approve minutes from previous meeting, and approve publishing
> previously approved minutes to ISOCPP.org
>
> 1.4 Action items from previous meetings
>
> 2. Main issues (125 min)
>
> 2.1 General logistics
>
> Future meeting plans
>
> June 9, 2021 02:00 PM ET/1800 UTC: Embedded
>
> July 14, 2021 02:00 PM ET/1800 UTC: Finance/Low Latency
>
> Aug 11, 2021 02:00 PM ET/1800 UTC: Games
>
> 2.2 Paper reviews
> Deterministic Exception for Embedded by James Renwick
>
> https://www.pure.ed.ac.uk/ws/portalfiles/portal/78829292/low_cost_deterministic_C_exceptions_for_embedded_systems.pdf
>
> Freestanding?
>
> 2.2.1 any other proposal for reviews?
>
> SG14/SG19 features/issues/defects:
>
>
> https://docs.google.com/spreadsheets/d/1JnUJBO72QVURttkKr7gn0_WjP--P0vAne8JBfzbRiy0/edit#gid=0
>
> 2.3 Domain-specific discussions
>
> 2.3.1 SIG chairs
>
> - Embedded Programming chairs: Ben Craig, Wouter van Ooijen and Odin
> Holmes, John McFarlane

>
> - Financial/Trading chairs: Staffan
> TjernstrÃƒÆ'Ã†â€™Ãƒâ€ Ã¢â‚¬â„¢ÃƒÆ'Ã¢â‚¬Å¡Ãƒâ€šÃ‚Â¶m, Carl
> Cooke, Neal
> Horlock,
> Mateusz Pusz, Clay Trychta,
> - Games chairs: Rene Riviera, Guy Davidson and Paul Hampson
> - Linear Algebra chairs: Bob Steagall, Mark Hoemmen, Guy Davidson
>
> 2.4 Other Papers and proposals
>
> 2.5 Future F2F meetings:
>
> 2.6 future C++ Standard meetings:
> https://isocpp.org/std/meetings-and-participation/upcoming-meetings
>
> -
>
> 3. Any other business
> Reflector
> https://lists.isocpp.org/mailman/listinfo.cgi/sg14
> As well as look through papers marked "SG14" in recent standards committee
> paper mailings:
> http://open-std.org/jtc1/sc22/wg21/docs/papers/2015/
> http://open-std.org/jtc1/sc22/wg21/docs/papers/2016/
>
> Code and proposal Staging area
> https://github.com/WG21-SG14/SG14
> 4. Review
>
> 4.1 Review and approve resolutions and issues [e.g., changes to SG's
> working draft]
>
> 4.2 Review action items (5 min)
>
> 5. Closing process
>
> 5.1 Establish next agenda
>
> 5.2 Future meeting
>

> *>*
> *>*
> *> July 14, 2021 02:00 PM ET/1800 UTC: Finance/Low Latency*
> *>*
> *> Aug 11, 2021 02:00 PM ET/1800 UTC: Games*
> *>*
> *> Kind Rgds*
> *>*

--0000000000004072da05c47b4ffc Content-Type: text/plain; charset="UTF-8"

On Thu, 10 Jun 2021 at 21:32, Tjernstrom, Staffan via SG14 <
sg14_at_[hidden]> wrote:

> *> As was mentioned by a number of people in the meeting yesterday, it'd be a*
> *> good idea to set up a list-of-topics of interest for a finance related*
> *> discussion.*
> *>*
> *> I believe from the transcript that Jeff and Henry may have some ideas, and*
> *> for myself I'm very interested in anyhthing that we can do to help out*
> *> getting the library portion of P0593 into '23.*
> *>*

Perhaps the spreadsheet tracking features, issues and defects
<https://docs.google.com/spreadsheets/d/1JnUJBO72QVURttkKr7gn0_WjP--P0vAne8JBfzbRiy0/edit#gid=0>
might provide inspiration or a place for suggestions.

John

> *>*
> *> Kind Rgds*
> *> Staffan Tj.*

Minutes for 2021/09/08 SG14 Conference Call

> > Thanks to Rene for Chairing.
> >
> > On Tue, Sep 7, 2021 at 10:53 AM Michael Wong <fraggamuffin_at_[hidden]
> > <mailto:fraggamuffin_at_[hidden]>> wrote:
> >
> > Topic: SG14 Low Latency Monthly
> >
> > This meeting is focused on Games .
> >
> > Hi,
> >
> > Michael Wong is inviting you to a scheduled Zoom meeting.
> >
> > Topic: SG14 monthly
> > Time: 2nd Wednesdays 02:00 PM Eastern Time 1800 UTC (US and Canada)
> > Every month on the Second Wed,
> >
> >
> > Join from PC, Mac, Linux, iOS or Android:
> >
> https://iso.zoom.us/j/93151864365?pwd=aDhOcDNWd2NWdTJuT1loeXpKbTcydz09
> > <
> https://iso.zoom.us/j/93151864365?pwd=aDhOcDNWd2NWdTJuT1loeXpKbTcydz09>
> > Password: 789626
> >
> > Or iPhone one-tap :
> > US: +12532158782,,93151864365# or +13017158592,,93151864365#
> > Or Telephone:
> > Dial(for higher quality, dial a number based on your current
> > location):
> > US: +1 253 215 8782 or +1 301 715 8592 or +1 312 626 6799
> or +1
> > 346 248 7799 or +1 408 638 0968 or +1 646 876 9923 or +1 669 900 6833
> > or 877 853 5247 (Toll Free)

> > Meeting ID: 931 5186 4365
> > Password: 789626
> > International numbers available:
> > https://iso.zoom.us/u/abRrVivZoD <https://iso.zoom.us/u/abRrVivZoD>
> >
> > Or Skype for Business (Lync):
> > https://iso.zoom.us/skype/93151864365
> > <https://iso.zoom.us/skype/93151864365>
> >
> > Agenda:
> >
> > 1. Opening and introduction
> >
> > ISO Code of Conduct
> > <
> >
> https://isotc.iso.org/livelink/livelink?func=ll&objId=20882226&objAction=Open&nexturl=%2Flivelink%2Flivelink%3Ffunc%3Dll%26objId%3D20158641%26objAction%3Dbrowse%26viewType%3D1
> > <
> https://isotc.iso.org/livelink/livelink?func=ll&objId=20882226&objAction=Open&nexturl=%2Flivelink%2Flivelink%3Ffunc%3Dll%26objId%3D20158641%26objAction%3Dbrowse%26viewType%3D1
> >
> > >
> >
> > ISO patent policy.
> >
> https://isotc.iso.org/livelink/livelink/fetch/2000/2122/3770791/Common_Policy.htm?nodeid=6344764&vernum=-2
> > <
> https://isotc.iso.org/livelink/livelink/fetch/2000/2122/3770791/Common_Policy.htm?nodeid=6344764&vernum=-2
> >
> >
> > IEC Code of Conduct:
> >
> > https://www.iec.ch/basecamp/iec-code-conduct-technical-work

> > <https://www.iec.ch/basecamp/iec-code-conduct-technical-work>
> >
> > WG21 Code of Conduct:
> >
> >
> https://isocpp.org/std/standing-documents/sd-4-wg21-practices-and-procedures
> > <
> https://isocpp.org/std/standing-documents/sd-4-wg21-practices-and-procedures
> >
> >
> > 1.1 Roll call of participants
> >
> > 1.2 Adopt agenda
> >
> > 1.3 Approve minutes from previous meeting, and approve publishing
> > previously approved minutes to ISOCPP.org
> >
> > 1.4 Action items from previous meetings
> >
> > 2. Main issues (125 min)
> >
> > 2.1 General logistics
> >
> > Future meeting plans
> >
> > July 14, 2021 02:00 PM ET/1800 UTC: Finance/Low Latency
> >
> > Aug 11, 2021 02:00 PM ET/1800 UTC: Games Cancelled
> >
> > Sep 8 , 2021 02:00 PM ET/1800 UTC: Games
> > Oct 13, 2021 02:00 PM ET/1800 UTC: Embedded
> > Nov 10, 2021 02:00 PM ET/1800 UTC: DST change/cancelled
> > Dec 8, 2021 02:00 PM ET/1800 UTC: Finance/Low Latency
> >
> > 2.2 Paper reviews
> >
> > Discussion about Games topics:
> >

> > P2388R1 - Minimum Contract Support: either Ignore or Check_and_abort
> > <
> http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2021/p2388r1.html>
> >
> > P2388R1:
> > What is stopping static analyzers
> > want to allow programmers to declare good contract annotations in 23
> > so that in 26 tools can have annotations
> > min annotations
> > need from std:
> > portable notatoion
> > type system checking declared expre
> > guarantee that predicates can be rt evaluated if needed
> > ack disappearing side effects in declared expr
> > ack possibility of halting the program
> >
> > ignore mode
> > check and abort: noexcept by default
> > allow compiler to provide a way to install contract violation handler
> > but not require it: how about a compiler flag ?
> > possible to have pre/post-condition in hdr or cpp file: require from
> > first declaration; they are not part of type system
> > builtin unreachable is being standadized
> >
> > side effects in predicates as no way to be sure it is side effect free:
> > allowed but discouraged, may be evaluated, may be evaluated twice
> > ub if evaluation of one affects the other
> >
> > how does it interact with constexpr? constexpr fn with non-constexpr
> > predicate
> >
> > postconditions:
> > eval after destructors of local objects called
> > eval before destructors of function
> > referred objects values are those ...
> >
> > check value of parameter at the point it was passed, so need to make a
> > copy of the parameter in case parameter is changed during execution

> > if copy is made implicitly, then compiler can only optimize copy in some
> > cases by eliding
> > in ignore mode, you can avoid the copy
> >
> > precondition failure reported on caller side, rather then calleebut its
> > not always possible because we can not guarantee
> >
> > dropped from C++20 contracts
> > assertion levels
> > guaranteed non-evaluation of predicates (axiom)
> > continuation mode
> > interface
> >
> > Retained from C++20 contracts
> > contract annotations on first declaration
> > ODR identical annotations on overriding functions
> > access to private and protected members from predicates
> >
> > use case weher you want to widen and narroe the derived class contracts
> > and caller needs to be able to depend on the post condition being true
> > if you want to check you really need to check both the pre and post
> > conditions on the interface fn you call through so if you are calling
> > through a ptr to base , the caller is going to depend on the fact that
> > the post conditions and the pre and post conditions of the ptr you are
> > calling through are checked and the callee is going to depend on the
> > fact that the pre and post conditions it declares are going to be
> > checked. Crrectly checking the right things when they can be different
> > is complicated
> > So MVP approach is just require them to be the same for now and explore
> > what can be done later to be different
> >
> > Not addresign overriding fn issue, then this proposal will survive.
> >
> > Feedback on ignore mode, and const values
> > anything concern you on this approach?
> > in safety critical setting, might not be annotations we want them to be
> > not affecting too deeply the rest of the functions: can make it uniform
> > by referrng to a function input and output by value and are not

> const-able
> >
> > idiomatic constructor with operator plus trick would have to be
> rewritten?
> >
> >
> >
> >
> >
> >
> >
> >
> > Patrice's WIP on Games issues.
> >
> > Making C++ better for Game Developers: some Request
> > based on survey from local Montreal game companies
> > general principles:
> > simplify C++
> > make it teachable due to high turn over
> > avoid negative perf impact
> > debugging matters
> >
> > how to split these 30 proposals?
> >
> > similar to freestanding, chop into many small pieces
> > but also good to have an omnibus paper before the split
> > do a few feature per meeting, or wait until January for the next Games
> focus
> > should address Arthur's feedback and generate V2
> > some are low latency and can be in Dec meeting
> > updates coming
> >
> > Finance topics from July 14, 2021.
> >
> > https://lists.isocpp.org/sg14/2021/06/0636.php
> > <https://lists.isocpp.org/sg14/2021/06/0636.php>
> >
> > https://lists.isocpp.org/sg14/2021/07/0642.php

> > <https://lists.isocpp.org/sg14/2021/07/0642.php>
> >
> > 2.2.1 any other proposal for reviews?
> >
> > Deterministic Exception for Embedded by James Renwick
> >
> https://www.pure.ed.ac.uk/ws/portalfiles/portal/78829292/low_cost_deterministic_C_exceptions_for_embedded_systems.pdf
> > <
> https://www.pure.ed.ac.uk/ws/portalfiles/portal/78829292/low_cost_deterministic_C_exceptions_for_embedded_systems.pdf
> >
> >
> > Freestanding?
> >
> > SG14/SG19 features/issues/defects:
> >
> >
> https://docs.google.com/spreadsheets/d/1JnUJBO72QVURttkKr7gn0_WjP--P0vAne8JBfzbRiy0/edit#gid=0
> > <
> https://docs.google.com/spreadsheets/d/1JnUJBO72QVURttkKr7gn0_WjP--P0vAne8JBfzbRiy0/edit#gid=0
> >
> >
> > 2.3 Domain-specific discussions
> >
> > 2.3.1 SIG chairs
> >
> > - Embedded Programming chairs: Ben Craig, Wouter van Ooijen and
> Odin
> > Holmes, John McFarlane
> >
> > - Financial/Trading chairs: Staffan
> > TjernstrÃƒÆ’Ã†â€™Ãƒâ€ Ã¢â‚¬â„¢ÃƒÆ’Ã¢â‚¬
> > ÃƒÂ¢Ã¢â€šÂ¬Ã¢â€žÂ¢ÃƒÆ’Ã†â€™ÃƒÂ¢Ã¢â€šÂ¬
> >
> ÃƒÆ’Ã‚Â¢ÃƒÂ¢Ã¢â‚¬Å¡Ã‚¬ÃƒÂ¢Ã¢â‚¬Å¾Ã‚Â¢ÃƒÆ’Ã†â€™Ãƒâ€

Ã¢â‚¬â„¢ÃƒÆ’Ã‚Â¢Ã‚Â¢â‚¬Å¡Ã‚Â¬Ã‚Â¦Ã‚Â¡ÃƒÆ’Ã†â€™Ã‚Â¢Ã¢â€šÂ¬Ã…Â¡Ã‚Æ’Ã¢â‚¬Å¡Ã‚Â¬Ã‚Â¦Ã¢â€šÃ‚Â¶m,
> > Carl
> > Cooke, Neal
> > Horlock,
> > Mateusz Pusz, Clay Trychta,
> >
> >
> > Richard Smith to take over paper
> >
> > - Games chairs: Rene Riviera, Guy Davidson and Paul Hampson,
> > Patrice Roy
> >
> > - Linear Algebra chairs: Bob Steagall, Mark Hoemmen, Guy Davidson
> >
> > 2.4 Other Papers and proposals
> >
> > 2.5 Future F2F meetings:
> >
> > 2.6 future C++ Standard meetings:
> > https://isocpp.org/std/meetings-and-participation/upcoming-meetings
> > <https://isocpp.org/std/meetings-and-participation/upcoming-meetings
> >
> >
> > -
> >
> > 3. Any other business
> > Reflector
> > https://lists.isocpp.org/mailman/listinfo.cgi/sg14
> > <https://lists.isocpp.org/mailman/listinfo.cgi/sg14>
> > As well as look through papers marked "SG14" in recent standards
> > committee
> > paper mailings:
> > http://open-std.org/jtc1/sc22/wg21/docs/papers/2015/
> > <http://open-std.org/jtc1/sc22/wg21/docs/papers/2015/>
> > http://open-std.org/jtc1/sc22/wg21/docs/papers/2016/
> > <http://open-std.org/jtc1/sc22/wg21/docs/papers/2016/>
> >

> > Code and proposal Staging area
> > https://github.com/WG21-SG14/SG14 <https://github.com/WG21-SG14/SG14
> >
> > 4. Review
> >
> > 4.1 Review and approve resolutions and issues [e.g., changes to SG's
> > working draft]
> >
> > 4.2 Review action items (5 min)
> >
> > 5. Closing process
> >
> > 5.1 Establish next agenda
> >
> > 5.2 Future meeting
> >
> > Oct 13, 2021 02:00 PM ET/1800 UTC: Embedded
> >
> > Nov 10, 2021 02:00 PM ET/1800 UTC: DST change/cancelled
> > Dec 8, 2021 02:00 PM ET/1800 UTC: Finance/Low Latency
> >
> > Kind Rgds
> >
> >
> > _____
> > SG14 mailing list
> > SG14_at_[hidden]
> > https://lists.isocpp.org/mailman/listinfo.cgi/sg14
> >
> _____
> SG14 mailing list
> SG14_at_[hidden]
> https://lists.isocpp.org/mailman/listinfo.cgi/sg14

Minutes for 2021/10/13 SG14 Conference Call

On Mon, Oct 11, 2021 at 6:55 PM Michael Wong <fraggamuffin@gmail.com> wrote:

Topic: SG14 Low Latency Monthly

This meeting is focused on Embedded. But if time permits, we can also discuss Patrice's WIP games paper.

Hi,

Michael Wong is inviting you to a scheduled Zoom meeting.

Topic: SG14 monthly
Time: 2nd Wednesdays 02:00 PM Eastern Time 1800 UTC (US and Canada)
Every month on the Second Wed,

Join from PC, Mac, Linux, iOS or Android:
https://iso.zoom.us/j/93151864365?pwd=aDhOcDNWd2NWdTJuT1loeXpKbTcydz09
Password: 789626

Or iPhone one-tap :
US: +12532158782,,93151864365# or +13017158592,,93151864365#
Or Telephone:
Dial(for higher quality, dial a number based on your current location):
US: +1 253 215 8782 or +1 301 715 8592 or +1 312 626 6799 or +1 346 248 7799 or +1 408 638 0968 or +1 646 876 9923 or +1 669 900 6833 or 877 853 5247 (Toll Free)
Meeting ID: 931 5186 4365
Password: 789626
International numbers available: https://iso.zoom.us/u/abRrVivZoD

Or Skype for Business (Lync):
https://iso.zoom.us/skype/93151864365

Agenda:

1. Opening and introduction

ISO Code of Conduct
<
https://isotc.iso.org/livelink/livelink?func=ll&objId=20882226&objAction=Open&nexturl=%2Flivelink%2Flivelink%3Ffunc%3Dll%26objId%3D20158641%26objAction%3Dbrowse%26viewType%3D1
>

ISO patent policy.
https://isotc.iso.org/livelink/livelink/fetch/2000/2122/3770791/Common_Policy.htm?nodeid=6344764&vernum=-2

IEC Code of Conduct:

https://www.iec.ch/basecamp/iec-code-conduct-technical-work

WG21 Code of Conduct:

https://isocpp.org/std/standing-documents/sd-4-wg21-practices-and-procedures

1.1 Roll call of participants

1.2 Adopt agenda

1.3 Approve minutes from previous meeting, and approve publishing
previously approved minutes to ISOCPP.org

1.4 Action items from previous meetings

2. Main issues (125 min)

2.1 General logistics

Future meeting plans

Oct 13, 2021 02:00 PM ET/1800 UTC: Embedded
Nov 10, 2021 02:00 PM ET/1800 UTC: DST change/cancelled
Dec 8, 2021 02:00 PM ET/1800 UTC: Finance/Low Latency

2.2 Paper reviews

Discussion on Embedded topics:

update on the dedeprecating volatile paper.: p2327

Deterministic Exception for Embedded by James Renwick
https://www.pure.ed.ac.uk/ws/portalfiles/portal/78829292/low_cost_deterministic_C_exceptions_for_embedded_systems.pdf

Freestanding?

Discussion about Games topics:

P2388R1 - Minimum Contract Support: either Ignore or Check_and_abort
<http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2021/p2388r1.html>

Patrice's WIP on Games issues.

Finance topics from July 14, 2021.

https://lists.isocpp.org/sg14/2021/06/0636.php

https://lists.isocpp.org/sg14/2021/07/0642.php

2.2.1 any other proposal for reviews?

Deterministic Exception for Embedded by James Renwick
https://www.pure.ed.ac.uk/ws/portalfiles/portal/78829292/low_cost_deterministic_C_exceptions_for_embedded_systems.pdf

Freestanding?

SG14/SG19 features/issues/defects:

https://docs.google.com/spreadsheets/d/1JnUJBO72QVURttkKr7gn0_WjP--P0vAne8JBfzbRiy0/edit#gid=0

2.3 Domain-specific discussions

2.3.1 SIG chairs

- Embedded Programming chairs: Ben Craig, Wouter van Ooijen and Odin Holmes, John McFarlane, Paul bendixen

- Financial/Trading chairs: Staffan TjernstrÃm, Carl Cooke, Neal Horlock, Mateusz Pusz, Clay Trychta,

- Games chairs: Rene Riviera, Guy Davidson and Paul Hampson, Patrice Roy

- Linear Algebra chairs: Bob Steagall, Mark Hoemmen, Guy Davidson

2.4 Other Papers and proposals

2.5 Future F2F meetings:

2.6 future C++ Standard meetings:
https://isocpp.org/std/meetings-and-participation/upcoming-meetings

-

3. Any other business
Reflector
https://lists.isocpp.org/mailman/listinfo.cgi/sg14
As well as look through papers marked "SG14" in recent standards committee paper mailings:
http://open-std.org/jtc1/sc22/wg21/docs/papers/2015/
http://open-std.org/jtc1/sc22/wg21/docs/papers/2016/

Code and proposal Staging area
https://github.com/WG21-SG14/SG14
4. Review

4.1 Review and approve resolutions and issues [e.g., changes to SG's working draft]

4.2 Review action items (5 min)

5. Closing process

5.1 Establish next agenda

5.2 Future meeting

Nov 10, 2021 02:00 PM ET/1800 UTC: DST change/cancelled
Dec 8, 2021 02:00 PM ET/1800 UTC: Finance/Low Latency

Kind Rgds

14:02:26 So my name is Michael long I'm the chair of SG 14 but I have lots of CO chairs that helped me. And one of them is Paul Bendix and today he is now the co chair for the embedded, and he is going to be chairing this for today.

14:02:42 Because Today's focus is embedded, we have, we have a kind of a triple triple a triple focus between low latency and games and finance.

14:02:56 So, every day, every month I try to get somebody that helped me out. And, and share it. And so that also. They will also be, it will be good for them to step in.

14:03:07 When I'm when I, when I leave because I don't anticipate we'll be doing this forever.

14:03:12 All right. So, let me see.

14:03:16 So I see Paul Ben Dixon, I see Ben Craig, I see. Billy Baker I see Charles Bay I see Connor Holman I see Henry Miller, James Renwick Carmen Chen, Kim Nelson puter gorgeous silk running apology for mangling that name, Renee Ferdinand Rivera morale and

14:03:33 run and Friedman now Michael long.

14:03:39 Okay. So Paul could you start sharing your screen of the agenda item would be good.

14:03:59 I'm going to.

14:04:05 Is this working.

14:04:08 It is, I can see the screen quite nicely.

14:04:11 Okay so ladies.

14:04:14 So I guess.

14:04:18 So we get so it's all yours. I'm just going to stop talking now. Okay.

14:04:23 So I guess one of the very important things that we need right off the bat is someone to take the minutes.

14:04:39 Sure, give me a few minutes and I can set up unless Michael was going to do that. Okay. Thanks, Ben, I was going to. I normally do it but today I'm recording it also on the live transcript.

14:05:04 If you're doing it. That would be fantastic because you could. I always want to mix to 22 as if I can, if I push in writing notes but but knowing that there's a live transcript you, that relieves you from having to take so much detailed notes as well but the live transcript as you can see it's not 100% accurate.

14:05:10 accurate. As it sometimes it does a pretty good job but it's still, you know sometimes still misses things. So, yeah, please go ahead and take whatever notes you can, I would probably just focus on the questions and answers and rewind, our transcript to, to, to

14:05:24 take the rest. So, and I'll send you the minutes after the meetings done.

14:05:30 Yeah, thank you.

14:05:31 Okay.

14:05:32 All right.

14:05:37 Yep. Continue.

14:05:38 The first part is that we have a code of conduct, and some, and the Putin policy that everyone is record to read and follow, just to make sure that you're on page with that.

14:06:04 I guess the roll call Michael already cleared that up and people coming in will be handled by soon, somehow magically.

14:06:06 So, the next part is adopt the agenda.

14:06:10 So the agenda is more or less the default agenda, each and every month with the paper reviews.

14:06:19 This time, regarding the deprecating volatiles and deterministic exceptions, and free standing are everybody okay with the agenda.

14:06:37 Okay, not seeing anyone I'm using to scream, anything so I'll say that that was approved.

14:06:48 And regarding the minutes from previous meeting the minutes were sent out as far as I recall on the reflector.

14:06:58 And we need to approve that they can be published on cpp.org

14:07:08 not hearing any comments.

14:07:11 As far as I recall the one no action items from any of the previous meetings.

14:07:19 So, I'll be skipping that quickly.

14:07:25 So general logistics, is we need to take care of the future meetings, and the next meeting in November will be cancelled because of daylight savings time daylight saving time it'll be interesting to see what the EU comes up with in that regard next year

14:07:47 but I don't think that it will change the madness anything much. I think I think canceling it in the EU, or they love their.

14:07:57 What I've been following they have resolved to cancel the, the daylight saving, but the member states still have to agree on which way they're going to cancel it.

14:08:11 So that'll be fun. Okay.

14:08:16 And it's probably not going to change any of the madness, but at least the next meeting will be December, eighth and will be focused on finance, I don't know if we are coming up to a place where we need to take a look at having A.

14:08:36 Pacific time meeting again, but we sometimes do that I don't know if anyone knows of any. That's a good point we usually come up and do that on the game side because most of the games people on that on the, on the Australia.

14:08:49 Australia, I'll try to remember that. So that's a, that's an action item for me. Yeah. Thank you.

14:08:56 Good point good reminder.

14:08:59 Other things general logistics I didn't write down. Obviously this month is CPP con month, November, sorry the end of October, is going to be so

14:09:10 I'll see if you can is pretty strange this year is hybrid. Some people, most people are not going to be there but some people will I actually will be there because I think I have three or four hours.

14:09:21 Global committee 3d tutorials and and foundation directors meeting so I kind of have to be there. I have no choice. But a lot of you guys cannot because international travelers still cannot travel the US without being fully vaccinated.

14:09:36 With a vaccine that the US to recognize that's the key to enter phrase I believe but that's apparently is going to be changing by November, so we'll see what happens, even for me for Canadians, we couldn't drive to the US, we can only fly there.

14:09:47 See if you can make sense of that.

14:09:50 So I'm flying.

14:09:52 So that's, that's basically what we're at, we'll see how this hybrid formula goes. It's not just CPP con, a lot of other conferences, I'm going to. From this point on what it looks like is going to be.

14:10:06 Hybrid in some ways. And so that's not going to be too surprising.

14:10:10 Anyone else with any other things. So as such, there's no su 14 meeting as usual with sex CBP con, it was just too complicated for me to organize and.

14:10:20 But, but I think next year. Once it goes back to fully physical, we hope will try to do that again. Okay. And I'm not, I'm not popping for any other StG 14 meetings.

14:10:32 Any other conference again due to the complexity of it.

14:10:38 Yeah, just to make sure that. So the next meeting is going to be in December, and before that I believe that there will be the meeting c++ in Europe, and that is a completely online conference so that probably won't have any StG 14 component dialer.

14:11:01 Okay, thank you.

14:11:02 Okay, so, onto the discussion of the paper reviews. And so, I will take the chair myself and give a short update on the deprecating volatile paper.

14:11:21 At the last SG 14 embedded meeting there was some consensus that someone should write a paper, and I decided to give it a shot.

14:11:34 It went on to the reflector, and the r1 version is now been favorably received by evolution working group.

14:11:47 Thanks to you and small.

14:11:50 And the thing with the compound assignment in a volatiles that will be on dedicated are the big Weiss compound assignments. So the main concern that I heard was that the arithmetic operators gave some problems when people were erroneously using volatiles

14:12:21 in multi threaded environments. And that's the main cause for deprecating volatile compounds assignments, and by saying that we only dedicated for the big wise compound operators.

14:12:40 This means that we're dealing with bit twiddling where it's already a kind of safety of people only people who know what they're doing and trying to accomplish in an embedded setting will be doing these kinds of operations anyway.

14:13:01 And the arithmetic operations that seemed to be a problem in finance primarily still gets duplicated, and I didn't get in for a vote in the just past October meeting.

14:13:20 But I hope it will be in on the next plenary.

14:13:25 It should at least be in the mailing by the next plenary.

14:13:29 I don't know if, Michael, if you have any corrections or update son, I don't, I don't put the Can you show the paper. Do you have the paper.

14:13:38 I should have it.

14:13:43 You wrote you wrote it, so you I wrote it. So this is my presentation. I'm sorry if I'm blacking out everyone's screen right now, but

14:14:00 not catch

14:14:08 up.

14:14:16 So, I will stop sharing, and I will start sharing it again so that I share with the correct one.

14:14:26 This one.

14:14:29 So, the paper is here.

14:14:37 And the, the show was sent to the reflector.

14:14:45 And the main change is that what was called the compromise solution is now the solution.

14:14:55 Yeah, so it's all to do with these are equals and and not equals, and equals not to turn on and off bits. So we are dealing with bitwise operations, which

14:15:15 stands to reason that we should be doing been doing the bitwise compound operations. So, or equals, and equals and x equals.

14:15:27 And yen's was kind enough to help with the wording.

14:15:34 That always helps.

14:15:36 And, yes, This is the entire wording change.

14:15:47 So, yeah. And as I said, it, it seems, it got a favorable review in the wg, and as such.

14:16:02 Let's hope that it will go relatively smoothly through the rest of the process, even though the ISO CPP web page says not to expect anything until it's actually in the, in the international standard.

14:16:23 Any questions from anyone.

14:16:33 Great.

14:16:37 Then, I think we should go on to the next part, which is

14:16:50 deterministic exceptions for embedded by James Renwick, James. Do you have any updates for us.

14:17:03 Yes.

14:17:04 Nothing too.

14:17:06 Interesting, other than I had previously made some more progress, basically.

14:17:12 I've stopped all work on the solution for the exceptions outlined in the paper that I had done a few years back now I've stopped to work on that.

14:17:26 And that repository I've just made public, and I will post the links in the chat here.

14:17:34 In case anybody's interested in taking a look at that.

14:17:38 Just put the links in there.

14:17:40 So that's now been kind of not archived officially but I've certainly stopped work on that. And the rationale being that I believe it's now quite difficult to work on in terms of actually iterating and looking at it from a practical perspective and prototyping

14:17:56 things. The implementation was ok but could be a lot more modular and actually easy to change, so I'm kind of, it's also based on an older version of climbing.

14:18:08 So, I've started working on a new repository where I'm kind of porting across those changes to a newer clang and making basically making as much as I physically can in the actual c++ source file to be included as part of the compilation, to kind of like

14:18:26 a special system header that will be included everywhere.

14:18:30 implicitly. And what that means is rapid prototyping basically.

14:18:35 So, any changes that are made to kind of the data structure layouts for the exception info and, and that sort of thing that could negatively influence the code generation for this type of exceptions.

14:18:48 That is now in a c++ source file rather than spread out across various parts of compiler implementation and what that means is it's way easier to prototype things and get new performance numbers and all that sort of stuff.

14:19:01 So that's kind of under.

14:19:03 That's kind of slowly making its way towards reality.

14:19:07 But the original implementation is out there and that should come that was the one that I used roughly for the performance numbers, and that's mostly working there's some description there about the work remaining, and kind of what how it works and how

14:19:24 you could get set up to use it.

14:19:27 And there are still some no bugs and things but largely it.

14:19:31 It can compile arbitrary programs. The biggest problem still remains the standard library, and the fact that it's based on the C standard library which has a different API.

14:19:41 As a result of this change.

14:19:44 And, and so the kind of test programs and programs that I've run through this generally don't use the standard library or only use specific bits of it.

14:19:53 So that's something I'm still.

14:19:55 I've been talking to a number of people can offline about potential solutions and last call we had some ideas as well, which were worth discussing which was about, kind of, effectively rapping.

14:20:09 See library functions and some other things. and those are kind of still under investigation.

14:20:18 And those are kind of still under investigation. But the you having new repository, which is also linked there is on its way. And with that, it should be a bit more practical for actually trying out new things.

14:20:40 Great.

14:20:40 Looks very very interesting and very promising. I guess I won't be having any free time at all, the next couple of weeks.

14:20:53 Yeah.

14:20:53 Yeah, me neither.

14:20:58 Any further comments questions ideas to James's work.

14:21:06 Sure.

14:21:08 So yeah, thanks a lot for getting this out in the on GitHub.

14:21:15 You know, make it a lot easier for me to take a look at it and throw it at some of my benchmarks that I've probably need to revive.

14:21:25 In a similar vein. I have taken the very first step so gotten I've not gotten very far at all but I've taken some of the very first steps in prototyping some of the herb section stuff.

14:21:38 And when I say first steps I mean like

14:21:45 I'm doing something weird with knobs that I can talk to people about offline. And so I've just been figuring out how to generate not the right way. I haven't done anything with the front end.

14:21:58 So, I will likely be spying on James's repo quite a bit to figure out how to hook in it all of those levels as well.

14:22:08 So, the two efforts will be able to feed off of each other to some degree.

14:22:16 I hate to disappoint them, but it is, as you might expect, it's a little happy.

14:22:22 Yeah, who do we need to to find, to learn things and not necessarily to get them in an upstream will state.

14:22:32 Yes, the, the solution that I came up with was creating an empty expression, which was which was returnable, and as a result, generated nothing,

14:22:42 but that was so that's kind of front end modification rather than back end. So that so it was easier.

14:22:59 Michael's got his hand up.

14:23:03 Michael Scott. Please go on Michael.

14:23:14 If you're saying something we cannot hear you.

14:23:17 Double muted hardware and software mute sorry. Yeah, no, I agree. I think this is one of the most significant effort with we can be doing for the embedded group along with the games table that Patrice is working on I think we finally are going to be able

14:23:32 to make some headway with answering some of the key questions that we have always remember guys to go use of single, what we can do to change the standard I think that's the main thing we got out of the meetings three months ago.

14:23:47 Is that

14:23:47 what we're trying to figure out what we need to do to change the standards to support in the embedded stack based exception, so that we can create, get some sort of determinism out of, out of exception handling it we can do that I think that will be a

14:24:04 If we can do that I think that will be a great contribution to the community and to the embedded community and as you guys know I'm, I am not deeply embedded community as well too so I'm, I'm really really wanting to.

14:24:15 Thank you.

14:24:33 If there are no further answer or phrases to come up.

14:24:36 Then, I think the next part is freestanding with a nice question mark there has been some work done in the mailings I've seen, so I think that Finn would be the best one to answer any questions regarding freestanding.

14:24:59 Sure. So, Right now there are two papers in the LWGQ.

14:25:07 The optional operator new paper, and the.

14:25:16 Let me see. And I think that's p 2013.

14:25:20 And then P 1642 which is the easy utilities and a few other headers paper.

14:25:29 So those two are are have gotten approval from the respective evolution groups, but they are sitting in the LWGQ right now.

14:25:38 There are three papers that have not gotten through library evolution, and they're sitting there.

14:25:47 One for the feature test macros.

14:25:50 One for the stuff that overlaps with see as well as a few other string primitives.

14:25:59 And then the paper that has partial classes.

14:26:06 Things like array and string view where the class can conceptually work there but as some members that don't work well and freestanding.

14:26:18 And so those are all waiting for time, they might make 23 they might not.

14:26:26 So that's the state all of those papers are in.

14:26:32 I'm not currently authoring more papers to put on the library queue until at least.

14:26:40 The entirely 1642 the easy utilities paper, makes it through LWG because I don't want to deal with.

14:26:51 Updating too many papers in response to LWG feedback, so I'm waiting for some things to clear the queue before I put more on the other end.

14:27:04 Billy.

14:27:08 Yeah, I'm gay the status of where they are in the cues but you weren't given actual status on the content.

14:27:14 I mean, as far as I know there's nothing really controversial.

14:27:18 To the best of my knowledge, there isn't anything controversial.

14:27:24 There were a few rounds of questions on the Le wg mailing list for the partial classes, paper that I co authored so the main author meal, and I'm blanking on his last name.

14:27:42 He was able to comment on some of that stuff as well. But there wasn't a whole lot in terms of controversy there.

14:27:49 The C library stuff that has been seen by the sea liaison group.

14:27:59 And there was some commentary there but nothing that I could get a quorum. From the main thing that ended up changing.

14:28:08 By talking with the CVH on group is apparently and see they are deprecating or they're planning on deprecating all of the stuff.

14:28:20 And so they didn't want me to add that to freestanding, so I removed the into max overload of a few functions from my paper. There was some pushback on the W charge stuff.

14:28:34 And so I'm willing to remove that but I haven't removed that at this point.

14:28:40 And so that's what's in there and feature test macros are always exciting to talk about. So there's probably going to be some controversy there but I've played out a whole bunch of examples of what the code would look like with all of those feature Test

14:28:55 macros.

14:28:57 So hopefully that can stem the controversy.

14:29:03 Yes, Michael.

14:29:07 Can you guys hear me.

14:29:08 Yes.

14:29:09 Okay, okay. Yeah, I wanted to say to Ben to thank you for doing this. This is the other major contribution in the embedded space, but I was just going to ask, if this is on Sep 20 ones radar Do you need to present this work to sc 21, so that you can get

14:29:28 free compatibility.

14:29:29 So I did show the see compatibility one to two yeah the see liaison group.

14:29:39 I'm not sure how much of a.

14:29:45 How many rounds what what the process is for making that all work.

14:29:52 So they've seen it. There wasn't a vote or anything, but at least they're aware of it. I'm now on the reflectors I commented on some of their free standing work.

14:30:04 But to some degree, we've already got divergence.

14:30:09 And the stuff that I'm adding is mostly stuff that is already used in practice. So I mean I can talk some more with Aaron Bollman to see if he's expecting to see that paper again in the, in the sea liaison group or not.

14:30:30 It's a little tough to get in with that group right now because they're really trying to wrap up, see 23.

14:30:40 And so, they are generally focused on that stuff.

14:30:45 So, once.

14:30:50 See 23 is kind of put to bed and it will be easier to get see time at that point.

14:30:56 Thank you. I want to put people in context the context here trying to fix what we have in the present so that we can bear some resemblance to what we really do.

14:31:07 And so that in that way I really view this as being one of the most important things we're doing here. Thank you.

14:31:15 And I think that's all I've got to I'm going to try to catch up on the minute taking their.

14:31:22 Okay, um, I found out that you can't raise your hand if you're the host. So I'm going to inject myself and some Q and say, I would like your opinion been on the work I've been doing on having the ABR implementation of friends freestanding I'd like to

14:31:47 update it to follow the actual papers going through the standard, and for example in implement the feature chest macro parts, but I'm not going to bother if it's no use of the for you.

14:32:04 So, is that something that you would like me to continue to do, or would it be more prudent to just help with the paper.

14:32:18 I think that the implementation experience that the ABR implementation has provided has been very useful so far.

14:32:30 I don't know that

14:32:35 up to date.

14:32:37 experience would change things too much at this point.

14:32:44 So,

14:32:49 what in your shoes, I would say that working on exception stuff, at least for the short term is a better use of time.

14:33:01 Once a few more once a paper or to get through LWG, in particular that 1642 paper.

14:33:10 At that point, it might make more sense to jump back on the library side of things, and write the algorithms paper.

14:33:18 Though I think the main issue with the algorithms, paper, was that it really wanted to see library stuff to be there.

14:33:29 So those would be the things to look at.

14:33:38 Okay, great.

14:33:41 I'll take that into consideration.

14:33:44 So, with that, if there are no further comments or anything.

14:33:55 I guess the discussion about the games topics is more of a, an update. So, the minimum contract support.

14:34:07 Are there any updates on that.

14:34:16 I don't think I do also think you

14:34:19 will have to take whether say that for the next one.

14:34:25 Renee has a hand up.

14:34:28 Yeah, I mean i've been sitting in on contracts talks.

14:34:33 And so far it's looking like the minimum contract is not going to be part of 23.

14:34:42 That's three the extent of the.

14:34:46 update I think for them.

14:34:49 Short and sweet. Thanks for that. And I guess. Patrice is working progress.

14:34:57 He couldn't be here.

14:35:00 He wrote on the reflector so I guess we'll take that in December.

14:35:09 Right.

14:35:12 Yes, the deterministic extend a exceptions we've already gone through those.

14:35:21 And yes, we have a very nice very long list of papers that were involved with the issue 14, that is available here probably don't need to go through that.

14:35:42 If there are no other papers, we could probably end the meeting early. Yeah, I was just about to say. You've all got the, the agenda and so if you want to take a look at it, please do.

14:35:50 it's very interesting reading, reading

14:35:55 domain specific discussions.

14:35:59 I think we've already done that.

14:36:02 And, and I don't know that there are any other papers or proposals in the queue,

14:36:15 not hearing any protests.

14:36:18 So future face to face meetings are probably a ways off still.

14:36:26 So I have some update there

14:36:30 is no fully virtual Intel, they don't they didn't know the status of international travel is going in.

14:36:38 So now we're looking at the Summer 2022 which sounds like it's going to be hosted by Bloomberg in New York.

14:36:48 Okay, so summer 2022.

14:37:00 This. The Danish government has declared that code is no longer a serious health issue so there are no longer restrictions at all.

14:37:16 So interesting how how different it is all around the world.

14:37:11 And by the way, I'm Paul you felt my note I can fund that you do have a delegation, and you should contact.

14:37:17 Kill Simonson ahead of delegation. Okay.

14:37:32 I will do join them. I don't know you have to pay some countries, you have to pay, usually Commonwealth countries is free, like, anything that's associated with the Queen of England is apparently, any country that's associated with the Queen of England

14:37:33 and somehow create like Australia and Canada and UK side, the US charges like 1200 dollars for every, every working group.

14:37:42 And then $300 after the first one.

14:37:47 Yeah, I'm pretty sure Denmark is not part of the Commonwealth.

14:37:56 You only, only invade the Commonwealth.

14:38:02 Exactly.

14:38:05 I guess they did the Commonwealth to stop us from invading anyways.

14:38:10 That is the statuses of future face to face meetings.

14:38:15 And I guess the future c++ standard meeting, there will be a

14:38:23 really cold, a plenary in February, as far as I recall.

14:38:31 Yes, it will be a virtual plenary.

14:38:35 Great.

14:38:38 Any other business.

14:38:41 I don't think that we really have anything to do.

14:38:47 Else, so since this is my first time, this is the place where I kindly ask anyone to remind me if there's anything I've forgotten.

14:39:00 You did a great job. Thank you. thank you for volunteering.

14:39:04 Okay.

14:39:06 Then, I think that I will call it, and the, I guess that Ben will make sure that Michael gets the minutes so that everyone can see what we've been talking about.

Minutes for 2022/01/12 SG14 Conference Call

Name (Original Name)
Michael Wong
Henry Miller
Balachandhar TN
John McFarlane
Ronen Friedman
Jens Maurer
Paul M. Bendixen
Kim Nilsson
Guy Davidson
John McFarlane
Kim Nilsson
Ka-Ming Chan
Pedro Oliveira
Pedro André Oliveira
Inbal Levi
Matthew Bentley
Henry Miller
René Ferdinand Rivera
Morell
Staffan Tjernstrom
Ben Saks
Charles Bay
Matthew Butler
Sophia Poirier
Ronen Friedman

>> Topic: SG14 Low Latency Monthly
>> This meeting is focused on Financial/low latency. But if time permits, we
>> can also
>> discuss Patrice's WIP games pape, P2300, and Swift
>>
>>
>>
>> Hi,
>>
>> Michael Wong is inviting you to a scheduled Zoom meeting.
>>
>> Topic: SG14 monthly

>> Time: 2nd Wednesdays 02:00 PM Eastern Time 1900 UTC (US and Canada)
>> Every month on the Second Wed,
>>
>> Join from PC, Mac, Linux, iOS or Android:
>> https://iso.zoom.us/j/93151864365?pwd=aDhOcDNWd2NWdTJuT1loeXpKbTcydz09
>> Password: 789626
>>
>> Or iPhone one-tap :
>> US: +12532158782,,93151864365# or +13017158592,,93151864365#
>> Or Telephone:
>> Dial(for higher quality, dial a number based on your current
>> location):
>> US: +1 253 215 8782 or +1 301 715 8592 or +1 312 626 6799 or +1
>> 346 248 7799 or +1 408 638 0968 or +1 646 876 9923 or +1 669 900 6833
>> or 877 853 5247 (Toll Free)
>> Meeting ID: 931 5186 4365
>> Password: 789626
>> International numbers available: https://iso.zoom.us/u/abRrVivZoD
>>
>> Or Skype for Business (Lync):
>> https://iso.zoom.us/skype/93151864365
>>
>> Agenda:
>>
>> 1. Opening and introduction
>>
>> ISO Code of Conduct
>> <
>>
>> https://isotc.iso.org/livelink/livelink?func=ll&objId=20882226&objAction=Open&nexturl=%2Flivelink%2Flivelink%3Ffunc%3Dll%26objId%3D20158641%26objAction%3Dbrowse%26viewType%3D1
>> *>*
>>
>> ISO patent policy.
>>
>> https://isotc.iso.org/livelink/livelink/fetch/2000/2122/3770791/Common_Policy.htm

?nodeid=6344764&vernum=-2
>>
>> IEC Code of Conduct:
>>
>> https://www.iec.ch/basecamp/iec-code-conduct-technical-work
>>
>> WG21 Code of Conduct:
>>
>>
>> https://isocpp.org/std/standing-documents/sd-4-wg21-practices-and-procedures
>>
>> 1.1 Roll call of participants
>>
>> 1.2 Adopt agenda
>>
>> 1.3 Approve minutes from previous meeting, and approve publishing
>> previously approved minutes to ISOCPP.org
>>
>> 1.4 Action items from previous meetings
>>
>> 2. Main issues (125 min)
>>
>> 2.1 General logistics
>>
>> Future meeting plans
>>
>>
>>
>> *Jan 12, 2022 02:00 PM ET/1900 UTC: Finance/Low Latency*
>> *Feb 9, 2022 02:00 PM ET/1900 UTC: Games*
>> *Mar 9, 2022 02:00 PM ET/1900 UTC: Embedded*
>>
>> 2.2 Paper reviews
>>
>> Discussion on Loww Latency/Finance topics
>> P2300
>>
> LEWG 3.5 hours each day in the past 3 months

sender receiver and asio
one addon change to C++23 to forward to library,
still design decision discussions outstanding; need change to some of this to increase consensus
majority happy, but some improvement to the paper could improve consensus
over 6 strongly against
some concerns: too much rush at end of release cycle, should this make C++ 23
P2300, differs sender vs typed sender(signature of receiver), and calls set value
1. non-typed senders are going away, must have signature
2. tag_invoke forwards customization to a layered stack of receivers for example, the return value needs to satisfy the concept and the tag_invoke on the receiver needs to work; both these requirements need to be satisfied
3. environment introduced so receiver can be queried for its environment in the connect function, so you can ask received for its environment to give properties like async wait state; can query the environemnt
4. sender trait defines the type for the set value overloads so you can look at properties of receiver so you can query receiver for set value signature; also helper structs to help you construct the special way in which teh set value set down
against : due to rushing, any sender type erased sender; require set error channel to always forward eh ptr because set value can throw
receiver concept requires that set error handler handles eh ptr Eric is looking at it, could be a problem for asynchronous networking on a single thread? how will this support ISRsm should not be only error channel

PB tried implementing minimal parts of it and did not see eh ptr requirement
is it throw all caution to the wind to get this through

Matthew Bentley
Colony name is a point of contention
Hive is now in the lead
BS has some concern on the name? LEWG, aiming for 23
need this out of LEWG by mid year, as it will take 3-5 cycles in LWG
https://github.com/cplusplus/papers/issues/328
https://www.reddit.com/r/cpp/comments/qug17i/c23_near_the_finish_line/

> Swift
>>
> SP summarized Swift model using annotation model for concurrent safety, though it seems Swift is not specifically about that and this is more narrow
 [[lock-free]], [[real-time]] [[no-alloc]]
MW mentioned interest in RUST and C++ safety
JM ask if there are general purposed annotation models to indentify functions with labels as being correct to the label
ST mentioned wrong defaults and the annotation burden
Another language to research.. https://www.ponylang.io/

>
>>
>>
>> Discussion about Games topics:
>>
>> P2388R1 - Minimum Contract Support: either Ignore or Check_and_abort
>> <http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2021/p2388r1.html>
>>
>> Patrice's WIP on Games issues.
>>
>> Finance topics from July 14, 2021.
>>
>> https://lists.isocpp.org/sg14/2021/06/0636.php
>>
>> https://lists.isocpp.org/sg14/2021/07/0642.php

>>
>> 2.2.1 any other proposal for reviews?
>>
>> Deterministic Exception for Embedded by James Renwick
>>
>> https://www.pure.ed.ac.uk/ws/portalfiles/portal/78829292/low_cost_deterministic_C_exceptions_for_embedded_systems.pdf
>>
>> Freestanding?
>>
>> SG14/SG19 features/issues/defects:
>>
>>
>> https://docs.google.com/spreadsheets/d/1JnUJBO72QVURttkKr7gn0_WjP--P0vAne8JBfzbRiy0/edit#gid=0
>>
>> 2.3 Domain-specific discussions
>>
>> 2.3.1 SIG chairs
>>
>> - Embedded Programming chairs: Ben Craig, Wouter van Ooijen and Odin
>> Holmes, John McFarlane
>>
>> - Financial/Trading chairs: Staffan TjernstrÃm
>> Carl Cooke, Neal Horlock,
>> - Games chairs: Rene Riviera, Guy Davidson and Paul Hampson, Patrice
>> Roy
>>
>> - Linear Algebra chairs: Bob Steagall, Mark Hoemmen, Guy Davidson
>>
>> 2.4 Other Papers and proposals
>>
>> 2.5 Future F2F meetings:
>>
>> 2.6 future C++ Standard meetings:
>> https://isocpp.org/std/meetings-and-participation/upcoming-meetings
>>
>> -

>>
>> 3. Any other business
>> Reflector
>> https://lists.isocpp.org/mailman/listinfo.cgi/sg14
>> As well as look through papers marked "SG14" in recent standards committee
>> paper mailings:
>> http://open-std.org/jtc1/sc22/wg21/docs/papers/2015/
>> http://open-std.org/jtc1/sc22/wg21/docs/papers/2016/
>>
>> Code and proposal Staging area
>> https://github.com/WG21-SG14/SG14
>> 4. Review
>>
>> 4.1 Review and approve resolutions and issues [e.g., changes to SG's
>> working draft]
>>
>> 4.2 Review action items (5 min)
>>
>> 5. Closing process
>>
>> 5.1 Establish next agenda
>>
>> 5.2 Future meeting
>>
>>
>> *Jan 12, 2022 02:00 PM ET/1900 UTC: Finance/Low Latency*
>> *Feb 9, 2022 02:00 PM ET/1900 UTC: Games GUY*
>> *Mar 9, 2022 02:00 PM ET/1900 UTC: Embedded*