Document Number: P3304R0
Date: 2024-05-22
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/Embedded/Financial Trading virtual Meeting Minutes 2024/04/10

Contents

Minutes for 2024/04/10 SG14 Conference Call

Thank you. I want to offer the same to Alisdair as well. Cheers.

On Wed, Apr 10, 2024 at 7:23 PM Arthur O'Dwyer <arthur.j.odwyer_at_[hidden]> wrote:

> We raised the possibility that attendees (or anyone, really) could compile
> their own codebases with my implementation of P1144 and/or with Corentin's
> implementation of P2786, and see how they compare in terms of ergonomics.
> Both implementations are forks of the (Clang + libc++) monorepo.
> Instructions for building the P1144 compiler are here:
> https://quuxplusone.github.io/blog/2024/04/10/p1144-your-codebase/
>
> Besides the headline
> - Please compile your codebase with this compiler+library combo and report
> the results!
> I have several subordinate requests:

> - Please, if you know how to make this more convenient (e.g. for users of
> VS Code with Docker), give me teh codez!
> - Please tell me how to build the P2786 reference implementation!
> - Please contact me if you're interested in a short-term contract to add
> P1144 support to GCC and/or libstdc++!
>
> Thanks,
> Arthur
>
>
> On Wed, Apr 10, 2024 at 3:53 PM Michael Wong via SG14 <
> sg14_at_[hidden]> wrote:
>
>>
>>
>> On Tue, Apr 9, 2024 at 4:23 PM Michael Wong <fraggamuffin_at_[hidden]>
>> wrote:
>>
>>> Hi, this SG14 meeting will focus on Finance/Low Latency
>>>
>>> Michael Wong is inviting you to a scheduled Zoom meeting.
>>>
>>> Topic: SG14 monthly
>>> Time: 2nd Wednesdays 02:00 PM Eastern Time (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

>>>

>> Andre Kostur, Andrew Lumsdaine, Arthur ODWyer, Ben Sherman, Cryan St.

>> Amour, Gianluca, Delfino, Jake Favold, Josh Gebara, Lauri Vasama, Matthew

>> Butler, Phil Ratzloff, Ronen Friedman, Adarsh, Michael Wong, Jens Maurer,

>> Alisdaire Meredith

>>

>>

>>>

>>> 1.2 Adopt agenda

>>>

>>> 1.3 Approve minutes from the 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

>>>

>>> 2024 planning

>>> C++23 and C++26 status

>>>

>> Tokyo F2F

>> Contracts and Microsoft feedbacks

>> P1144 on trivially relocatable

>> May 15th Mailing deadline

>>

>> - 2024-06-24 to 29: St. Louis, MO, USA

>> <https://isocpp.org/files/papers/N4966.pdf>; Bill Seymour

>> [image: ArmsSmall.jpg]

>> - 2024-11-18 to 23: Wrocław, Poland

>> <https://isocpp.org/files/papers/N4974.pdf>; Nokia

>>

>> C++26 Deadlines

>> https://www.open-std.org/jtc1/sc22/wg21/docs/papers/2023/p1000r5.pdf

>>

>>>

>>>

>>> Future and past meeting plans

>>>

>>> * Jan 10, 2024 02:00 PM ET: Games DONE

>>> * Feb 14, 2024 02:00 PM ET: Embedded DONE

>>> * Mar 13, 2024 02:00 PM ET: Cancelled due to Tokyo 3-18-23

>>> * Apr 10, 2024 02:00 PM ET: Finance

>>> * May 8, 2024 02:00 PM ET: Games/ P3160 assigned to SG14 to discuss

>>> * June 12, 2024 02:00 PM ET: Embedded; St.louis 6-24-29

>>> * July 10, 2024 02:00 PM ET: Finance

>>> * Aug 14, 2024 02:00 PM ET: Games

>>> * Sep 11, 2024 02:00 PM ET: CPPCON Sept 15-20 so cancelled

>>> * Oct 9, 2024 02:00 PM ET: Embedded

>>> * Nov 13, 2024 02:00 PM ET: Cancelled Wroclaw F2F

>>> * Dec 11, 2024 02:00 PM ET: Finance

>>>

>>> 2.2 Paper reviews

>>> Embedded:

>>> * P3132 Accept attributes with user-defined prefixes

>>> * P3134 Attribute [[asserts_rvo]]

>>> 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 Updates

>>>

>>> Games paper review

>>>

>>> Arthur's suggestions:

>>> (1) I put in the Slack channel

>>> <https://cpplang.slack.com/archives/C3TK2M6HH/p1703947057425609> a while

>>> ago Clang PR #76596 <https://github.com/llvm/llvm-project/pull/76596>,

>>> from

>>> one Max Winkler, apparently in game dev. I don't think the PR stands much

>>> chance of getting merged into Clang; but it might still be of interest to

>>> SG14 folks. The issue description is very long and somewhat detailed, and

>>> then there's more discussion/debate in the comments

>>> <https://github.com/llvm/llvm-project/pull/76596#issuecomment-1872601156

>>> >.

>>> (I'd actually be interested in talking to Max, but he doesn't publish his

>>> email address on GitHub and I guess that might be on purpose.)

>>>

>>> (2) LEWG will be seeing my P3055 "Relax wording to permit relocation

>>> optimizations in the STL"

>>> <https://quuxplusone.github.io/draft/d3055-relocation.html> in a

>>> telecon on

>>> February 20th. (Related blog post.

>>> <https://quuxplusone.github.io/blog/2024/01/02/bsl-vector-erase/>) Might

>>> be interesting to folks who do EASTL-style containers. I'd be interested

>>> in

>>> early feedback and/or telecon attendance.

>>>

>>>

>>> Discussion on Embedded:

>>> Paul's suggestions

>>> The next meeting would then be Embedded and I would be interested in

>>> knowing if people think a module std.freestanding is worth pursuing.

>>> In that context I'd like to get some feedback perhaps already for the

>>> upcoming meeting, if people have started using modules, and if so if it

>>> has

>>> brought the promised expectations or if you are holding back if you see

>>> any

>>> relevance in modules.

>>>

>>> Review latest mailings:

>>> P2532 Removing exception_ptr from the receivers concept

>>> Based on the last meeting and the discussions here.

>>> P2544 C++ Exceptions are becoming more and more problematic

>>> We might want to chime in here.

>>> /Paul

>>> P. S. P2327 de-deprecating volatile received a "consensus" straw poll.

>>>

>>>

>>> Discussion on Low Latency/Finance topics

>>>

>>> http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2022/p1839r4.pdf

>>>

>>>

>>> Who is in FInance

>>

>> Please will someone stand to be Finance SIG chair?

>>

>> Bryan St. Amour (Maystreet) ultra low latency, market feed, data handler,

>> high frequency, banks

>>

>> -data capture side, network, packets, 0 allocation, without dropping to a

>> lower language

>>

>> -cant do heap allocation, runtime eh is not fine

>>

>> Nathan Owen(Maystreet) - embedded but now Finace

>>

>> Ben Sherman (Chicago Trading) - market maker

>>

>> -compile time evaluation, value semantics, type manip,

>>

>> -strggle in type with iteration, transformation in constexpr values

>>

>> Jake Fevold (Bloomberg) - finance media company

>>

>> -contracts

>>

>> -safety

>>

>> Alisdair Meredith

>>

>> -allocators, control memory location, low latency

>>

>> Gianluca Delfino (maystreet)

>>

>> - MS before, trivially relocatable object, reinterpret cast, in-place
>> vectors to know allocation, cache locality

>>

>> -security (not on networks), memory/resource/dangling type of safety,
>> allocation, memory, locality, network facility, lifetimes

>>

>> Josh Gebara (Bloomberg) - allocators, safety

>>

>> Is there any grassroot movement to Rust? No from Maystreet. Concerned the
>> RUST hype vs C++ viability and investment. May be offer a C++ solutions to
>> memory safe RUST

>>

>>

>>

>>>

>>>

>>>

>>> 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>

>>>

>>>

>>>

>>>

>>>

>>> 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: Robin Rowe, 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

>>>

>> Relocation: What is a trivially relocatable type

>> Asking SG14 to provide feedback on both proposals usefulness. Links to be

>> sent after the call

>> P1144 Arthur

>> https://www.open-std.org/jtc1/sc22/wg21/docs/papers/2024/p1144r10.html

>> uninitialized_copy expected to get a memcpy, have a continguous range,

>> but it is not

>> if there are special functions, it is not copy constructible then you

>> have to do an actual copy constructor in a loop

>> types that are trivially relocatable, on the object same as on the value,

>> can relocate existing objects

>> e.g. of these types are vector, unique_ptr,

>>

>> can now have a Span of objects, when number is the same, then I can move

>> the objects around, copy/rotate/swap/permute, fastere because it is memove,

>> smaller , more eh safe,

>> in EWGI in Kona straw poll had no consensus

>> top down design

>> https://godbolt.org/z/o7jMo4E7e is the godbolt I was showing, btw

>>

>> https://quuxplusone.github.io/blog/2018/07/13/trivially-copyable-corner-cases/

>> Is this useful for this feature?

>>

>> P2786 Alisdair

>> Bottom up

>> https://www.open-std.org/jtc1/sc22/wg21/docs/papers/2024/p2786r4.pdf

>> volatile that is relocatble makes sense? nothing that makes it impossible

>> in core lang, volatile int is trivially copyable,

>> but there are different types where it is under 1144,and under 2786

>> protection aginst your members, like third party, or Boost

>> this one prefers to trust the user

>> AO: what does that one function do? yes there is a semantic difference:

>> trust what they read, constraint their types reasons they dont want their

>> types to to be relocatable

>> IN EWG in Tokyo, 16/2 forward to core, one open question to resolve the

>> vexing parse

>> presented to LEWG, ok with the type traits, is it a building block?

>> this is a compiler feature with small library

>> have clang branch

>>

>> can this solve their problem in this space?

>>

>>

>>

>>

>>

>>

>>

>>

>>

>>>

>>> 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 10, 2024 02:00 PM ET: Games DONE

>>> * Feb 14, 2024 02:00 PM ET: Embedded DONE

>>> * Mar 13, 2024 02:00 PM ET: Cancelled due to Tokyo 3-18-23

>>> * Apr 10, 2024 02:00 PM ET: Finance

>>> * May 8, 2024 02:00 PM ET: Games

>>> * June 12, 2024 02:00 PM ET: Embedded; St.louis 6-24-29

>>> * July 10, 2024 02:00 PM ET: Finance

>>> * Aug 14, 2024 02:00 PM ET: Games

>>> * Sep 11, 2024 02:00 PM ET: CPPCON Sept 15-20 so cancelled

>>> * Oct 9, 2024 02:00 PM ET: Embedded

>>> * Nov 13, 2024 02:00 PM ET: Cancelled Wroclaw F2F

>>> * Dec 11, 2024 02:00 PM ET: Finance