Document Number:	P0937R0
Date:	2018-02-12
Authors:	Michael Wong
Project:	Programming Language C++, SG5 Transactional Memory
Reply to:	Michael Wong <michael@codeplay.com>

SG5: Transactional Memory (TM) Meeting Minutes 2017/10/23-2018/1/29

Contents

Minutes for 2017/10/23 SG5 Conference Call

Minutes by Hans

1.2 Adopt agenda
1.3 Approve minutes from previous meeting, and approve publishing previously approved minutes to ISOCPP.org
1.4 Review action items from previous meeting (5 min)
1.5 Call schedules
Aug 14 DONE
Aug 28 DONE
Sep 11 Cancelled
Sep 25 Michael away
Oct 9 Mailing deadline Oct 16 DONE
Oct 23
Nov 6 C++ Meeting Albuquerque
2. Main issues (50 min)
2.1 Future of TM Discussion with Herb
Herb, if you like to send any pre-call material or discussion, please go ahead.
Herb is not here. Michael W and Hans had a phone call with Herb. He is interested
Herb not here. Herb is interested in making TM more acceptable, possibly by focussing on it as a replacement for small memory-only code segments that would otherwise use lock-free code..
MW and HB had an email exchange with him.
2.2: Interaction with Executors and Synchronized proposal
https://groups.google.com/a/isocpp.org/forum/#!topic/tm/jG9XPJetNkc
The last discussion has us considering an alternative lambda form.
See Paper emailed out on Lambda proposal
https://docs.google.com/document/d/1ICmcrCdigq3ataoM2Jl7m19h_Sa3aE3KfU6AVkPyT-4/edit#
Discussed some changes, particularly to discussion points, in the above lambda proposal.
MW suggested phrasing issues as straw polls.
Need some discussion of whether everyone is OK with losing static atomicity guarantee.
MW: Say something about implementation status?
MSpear: Alpha quality. Exceptions currently break things.
MSpear: Is removal of static checks a deal-breaker? What about transaction deferral?
HB: Seems to be desired to accommodate output.
Discussion of shared_ptr, unique_ptr
MSpear: shared_ptr should work correctly with x86 hardware implementation.
MSpear: Do we need a lock that doesn't alway spin?
MSpear: Do we need to worry about in-the-kernel locks?
Probably not.
Relationship to Olivier's "synchronic" etc. proposals. Probably independent.
2.3 future issues list:
1. llvm synchronized blocks
2. more smart ptrs?how fast can atomics and smart ptrs be outside tx if they have to interact with tx (for world that does not care about tx), the atomic nature of smart ptrs as a way towards atomics inside atomic blocks
3. more papers?
4. Issue 1-4 paper updates to current TM spec
5. std library
2.4 Discuss defects if any work done since last call
Issue 1: https://groups.google.com/a/isocpp.org/forum/#!topic/tm/SMVEiVLbdig
Issue 2: https://groups.google.com/a/isocpp.org/forum/#!topic/tm/Th7IFxFuIYo

Issue 3: https://groups.google.com/a/isocpp.org/forum/#!topic/tm/CXBycK3kgo0
Issue 4: https://groups.google.com/a/isocpp.org/forum/#!topic/tm/Ood8sP1jbCQ
3. Any other business
4. Review
4.1 Review and approve resolutions and issues [e.g., changes to SG's working draft]
N4513 is the official working draft (these links may not be active yet until ISO posts these documents)
http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2015/n4513.pdf
N4514 is the published PDTS:
http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2015/n4514.pdf
N4515 is the Editor's report:
http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2015/n4514.html
Github is where the latest repository is (I have updated for latest PDTS published draft from post-Leneaxa):
https://github.com/cplusplus/transactional-memory-ts
Bugzilla for filing bugs against TS:
https://issues.isocpp.org/describecomponents.cgi
4.2 Future backlog discussions:
4.2.1 Write up guidance for TM compatibility for when TM is included in C++ standard (SG5)
4.2.2 Continue Retry discussion
https://groups.google.com/a/isocpp.org/forum/?hl=en&fromgroups#!topic/tm/qB1Ib__PFfc
https://groups.google.com/a/isocpp.org/forum/#!topic/tm/7JsuXIH4Z_A
4.2.3 Issue 3 follow-up
Jens to follow up to see if anything needs to be done for Issue 3.
4.2.5 Future C++ Std meetings:
http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2017/n4633.pdf

2017-11 Albuquerque WG21 meeting information

Then Jacksonville, Rapperswil, ...

4.3 Review action items (5 min)
5. Closing process
5.1 Establish next agenda
5.2 Future meeting
Next call: TBD post ABQ meeting

Minutes for 2018/01/29 SG5 Conference Call

Minutes by Michael Scott, SG5
29 January 2018

Start Time: Monday, Jan 29 2018, 12:00 PM US Pacific Time (07:00 PM in GMT)
End Time: 1:00 PM US Pacific Time (duration: one hour)

Notes by Michael Scott.
The current secretary rota list is (the person who took notes at the
last meeting is moved to the end)

Michael Spear, Jens Mauer, Victor Luchangco, Michael Wong,
Hans Boehm, Maged Michael, Michael Scott

Agenda:

1. Opening and introductions

1.1 Roll call of participants

Victor Luchangco, Mike Spear, Hans Boehm, Michael Scott,
Piotr Balcer [Intel], Tom Kapela [Intel]

1.2 Adopt agenda

Interaction of TM w/ persistence

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

1.4 Review action items from previous meeting (5 min)

NA

1.5 Call schedules (please add your away days)

Jan 29: Michael away
Feb 12 : mailing deadline is 10 am ET today
Feb 26: Michael Scott away
March 12 C++ Meeting JAX

2. Main issues (50 min)

2.1 Persistent memory by piotr.balcer@intel.com

<< See slide deck:
https://docs.google.com/presentation/d/e/2PACX-1vTsbdYXg4Rh6HAoFHldYy-OY5RxeykN8Z5VEWc42aloKRIkOfxU0K5lDp4JfyDMvmx6mi569_s2Sjj2/pub
Ran through slides 1-18. >>

Misc. notes:
Asynchronous DRAM refresh important to semantics.
Atomicity guaranteed at 8-byte granularity only.
- Writes-back to different words of the same cache line may therefore reach memory out of order.

HW guaranteed that memory controller buffers will be flushed on power fail.
Anticipate mmap-ing files directly, w/out intervening kernel buffers.
- No need for msync()
- File system must leave page alone after it is mmap-ed()

May have 100s of GB on a single NVDIMM.

libpmemobj
- Provides transactions and atomic updates.
  - Implementation is undo-log based. Uses thread-local storage.
- malloc and free within file are failure atomic.
- Have to deal w/ fact that malloc and adding to data structure are separate steps; not atomic.
- Scoped wrapper for transactions.
- Snapshotting of basic data types
  - p<> property
  - Note that this envisions a static partitioning between persistent and nonpersistent memory. SG5 STM envisions operating on "ordinary" C++ data.
- Position-independent persistent pointers
  - Implementation is a 16B quantity containing UID for file and offset.

Challenges
- libstdc++ containers works w/ LLVM libc++ but not GNU libstdc++ or MSVC.
  - Latter 2 aren't yet C++11 compliant.
- Lack of standard layouts for data structures means you can get memory corruption if you write from code using one version and read or write from code using another.
  - Probably need some sort of version tagging.
- vptrs (and thus RTTI) don't work across program invocations!
  - Currently limited to POD objects.
- code of standard library operations and algorithms doesn't know about persistent pointers, and is compiled w/out necessary instrumentation.
  - Hope to build upon HTM and Intel libitm.
  - Envision an interface that implements operations to begin, commit, rollback txn; load, store word. Could then provide this

interface to a transaction, to be used inside, thereby avoiding
the need for the compiler to understand or have access to the
library.
Looking forward to future architectures that will flush _caches_
on power failure.

Discussion
What about accesses _outside_ one of these persistent-atomic blocks?
Might our newer ideas regarding executor-based transactions be a
better fit than the full technical specification.

NB: HTM provides isolation but not (failure) atomicity

Adjourned at 4pm.

Next call: Feb 12

(Didn't get to anything below here.)

2.2: Interaction with Executors and Synchronized proposal

https://groups.google.com/a/isocpp.org/forum/#!topic/tm/jG9XPJetNkc

The last discussion has us considering an alternative lambda form.

See Paper emailed out on Lambda proposal

https://docs.google.com/document/d/1ICmcrCdigq3ataoM2Jl7m19h_Sa3aE3KfU6AVkPyT-4/edit#

2.3 Future of TM Discussion with Herb

Herb, if you like to send any pre-call material or discussion, please go ahead.

2.4 future issues list:

1. llvm synchronized blocks