Paper Reading (schedule):

April 24, 2015, Satoshi Matsushita

INVYSWELL: A HYBRID TRANSACTIONAL MEMORY FOR HASWELL'S RESTRICTED TRANSACTIONAL MEMORY

http://cs.brown.edu/~irina/papers/invyswell.pdf

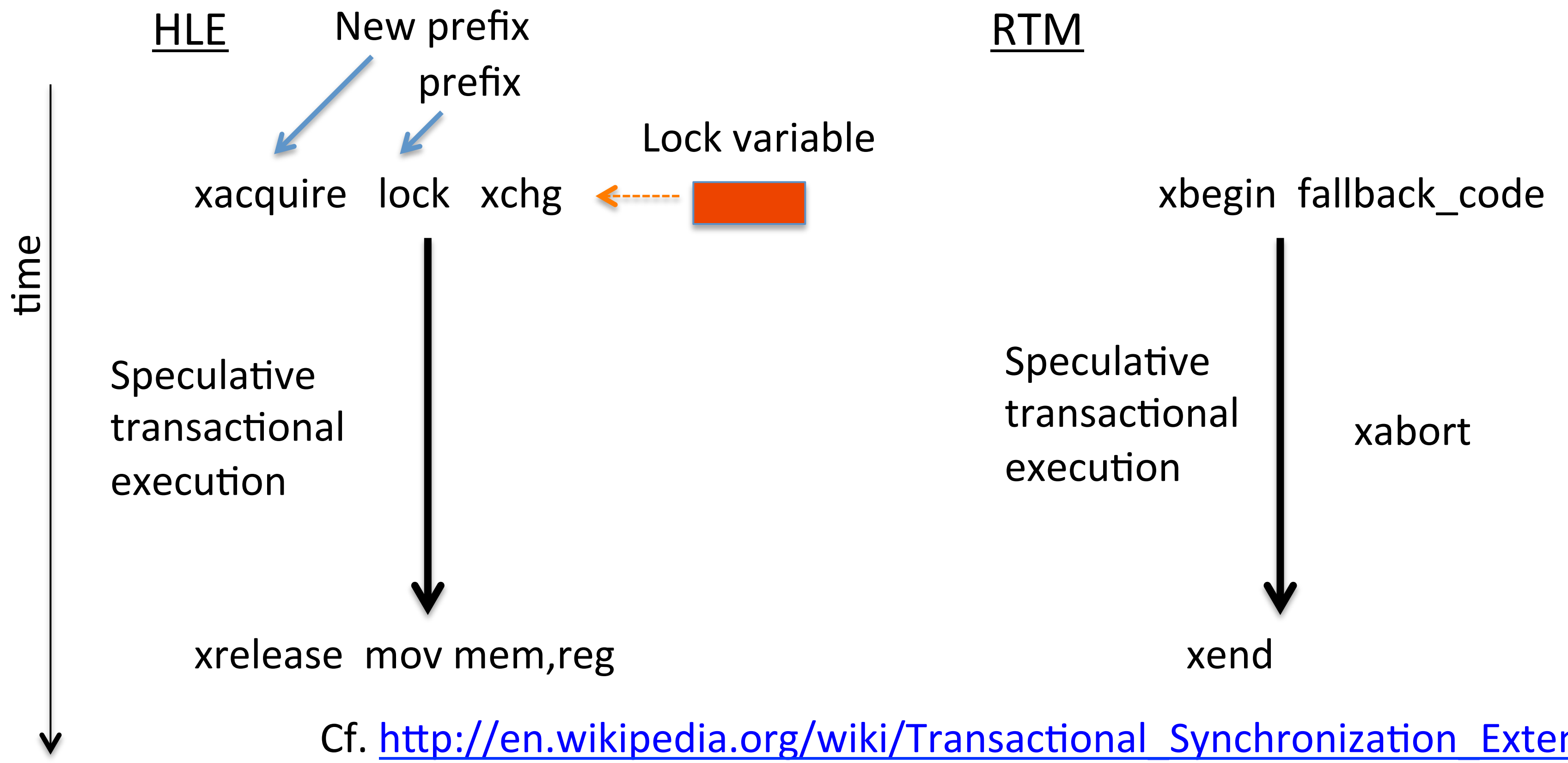
PACT 2014

Authors:

1. Irina Calciu, Brown Univ. http://cs.brown.edu/~irina/
2. Justin Gottschlich, Intel labs : http://justingottschlich.com/
3. Tatiana Shpeisman, Intel Labs
4. Gilles Pokam, Intel Labs
5. Maurice Herlihy, Brown Univ.: https://cs.brown.edu/people/faculty/mph.html

Intel Transactional Synchronization Extensions (TSX)

- Hardware Lock Elision (HLE)
- Restricted Transactional Memory (RTM)



HLE

New prefix

prefix

Lock variable

xacquire lock xchg

time

Speculative transactional execution

xrelease mov mem,reg

RTM

xbegin fallback_code

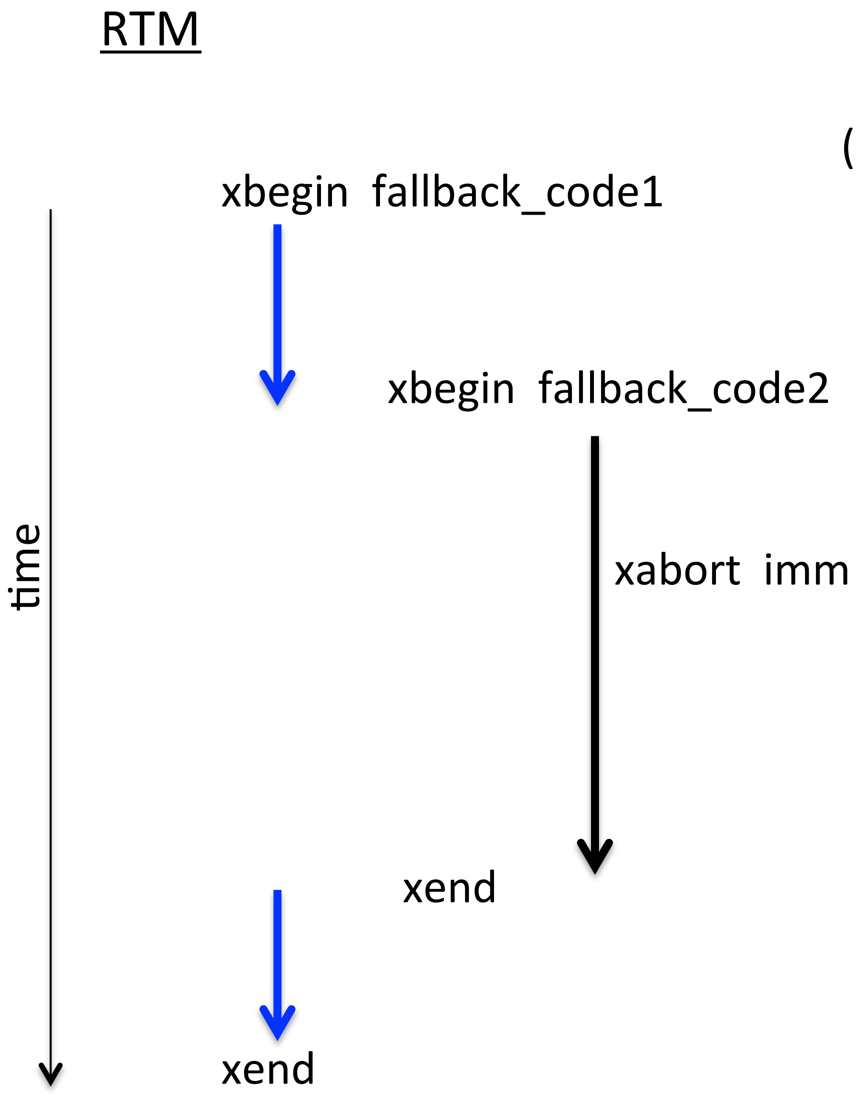
Speculative transactional execution

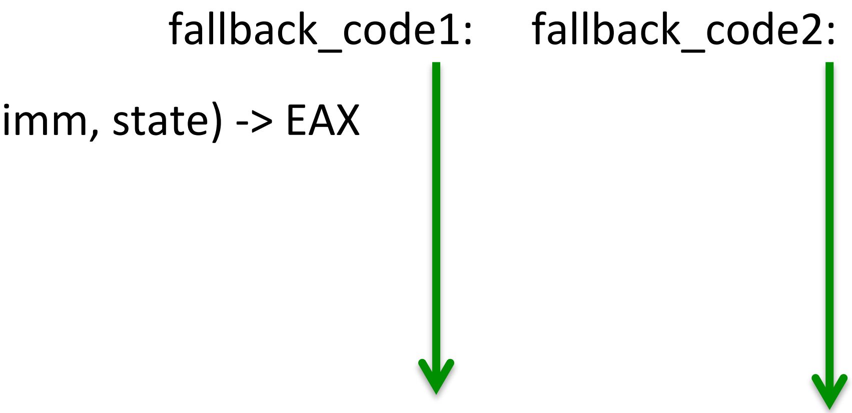
xabort

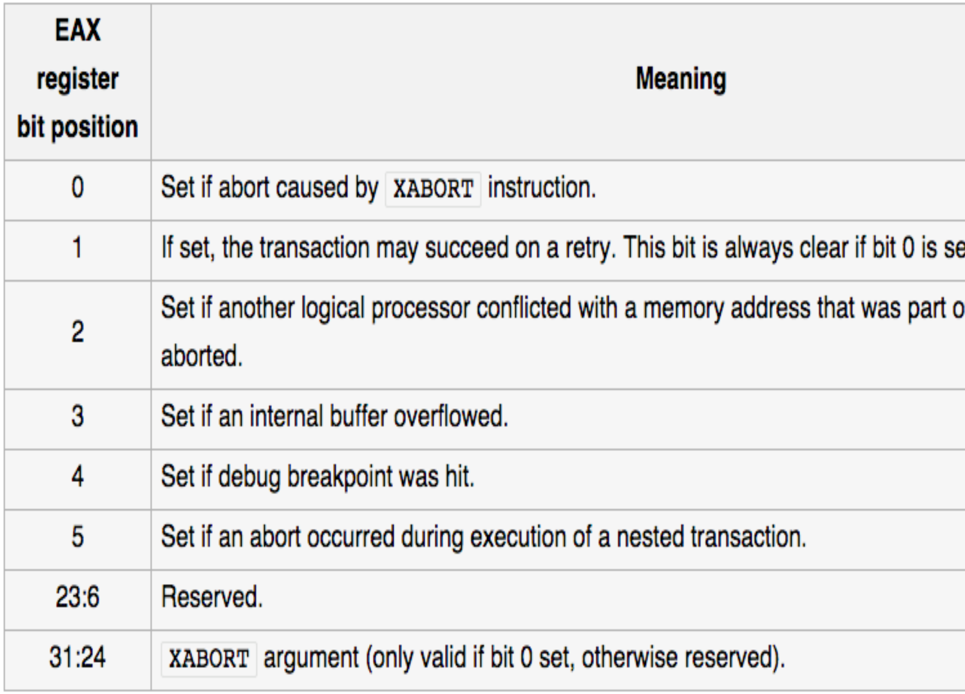
xend

Cf. http://en.wikipedia.org/wiki/Transactional_Synchronization_Extensions

TSX: nest, abort







EAX register bit position	Meaning
0	Set if abort caused by `XABORT` instruction.
1	If set, the transaction may succeed on a retry. This bit is always clear if bit 0 is se
2	Set if another logical processor conflicted with a memory address that was part o aborted.
3	Set if an internal buffer overflowed.
4	Set if debug breakpoint was hit.
5	Set if an abort occurred during execution of a nested transaction.
23:6	Reserved.
31:24	`XABORT` argument (only valid if bit 0 set, otherwise reserved).

RTM: Restrictions

- L1 Cache (32KB 8-assoc.) for transactional state store: need one cache line for ether read_set, or write_set
- Aborted by state buff full, interrupt, context switch
- Certain (uncommon) instr.s always cause abort
- No forward progress guaranteed
- No escape actions

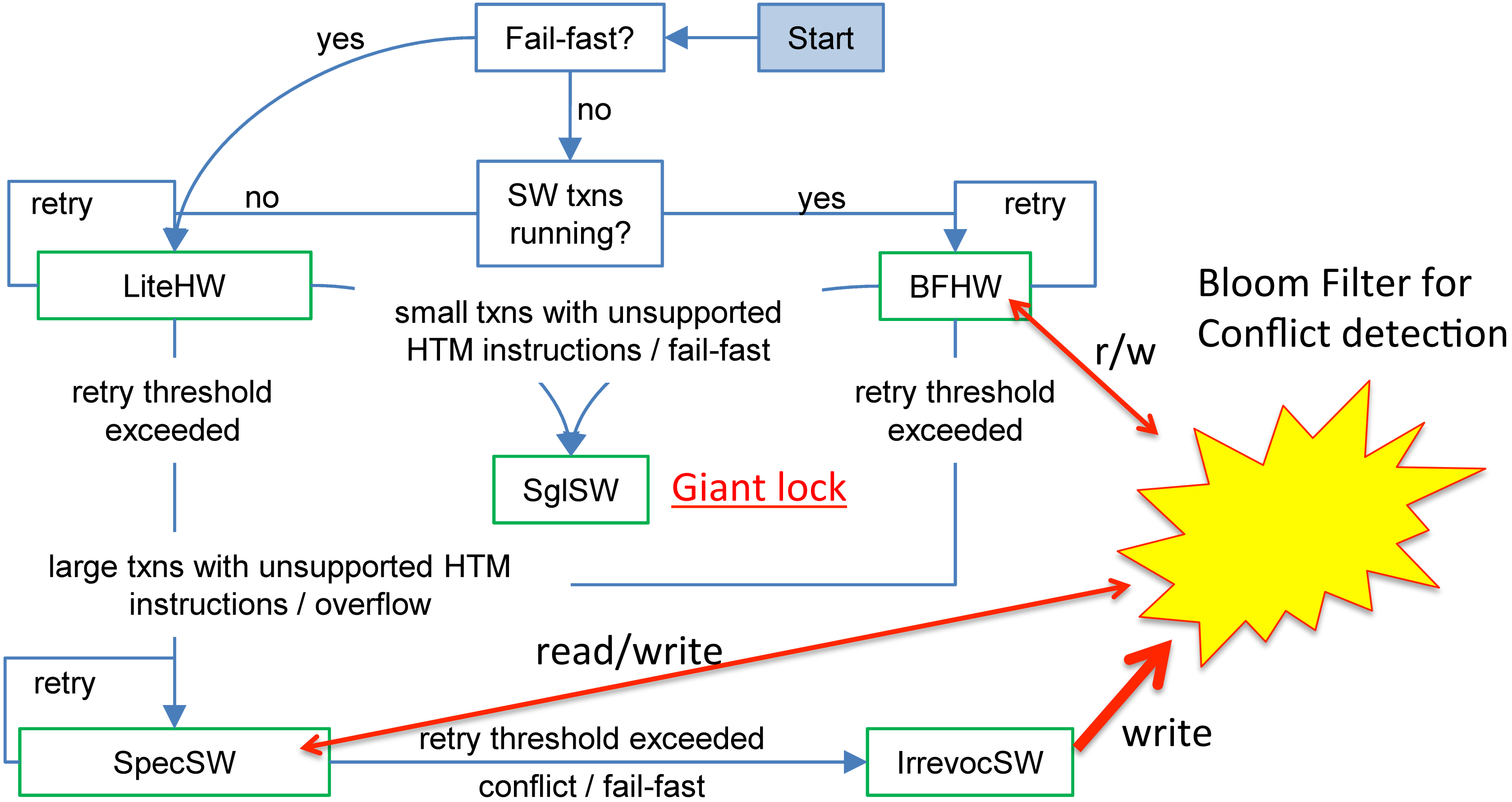
Cf. http://www.hpts.ws/papers/2013/HTM.pdf

The paper..

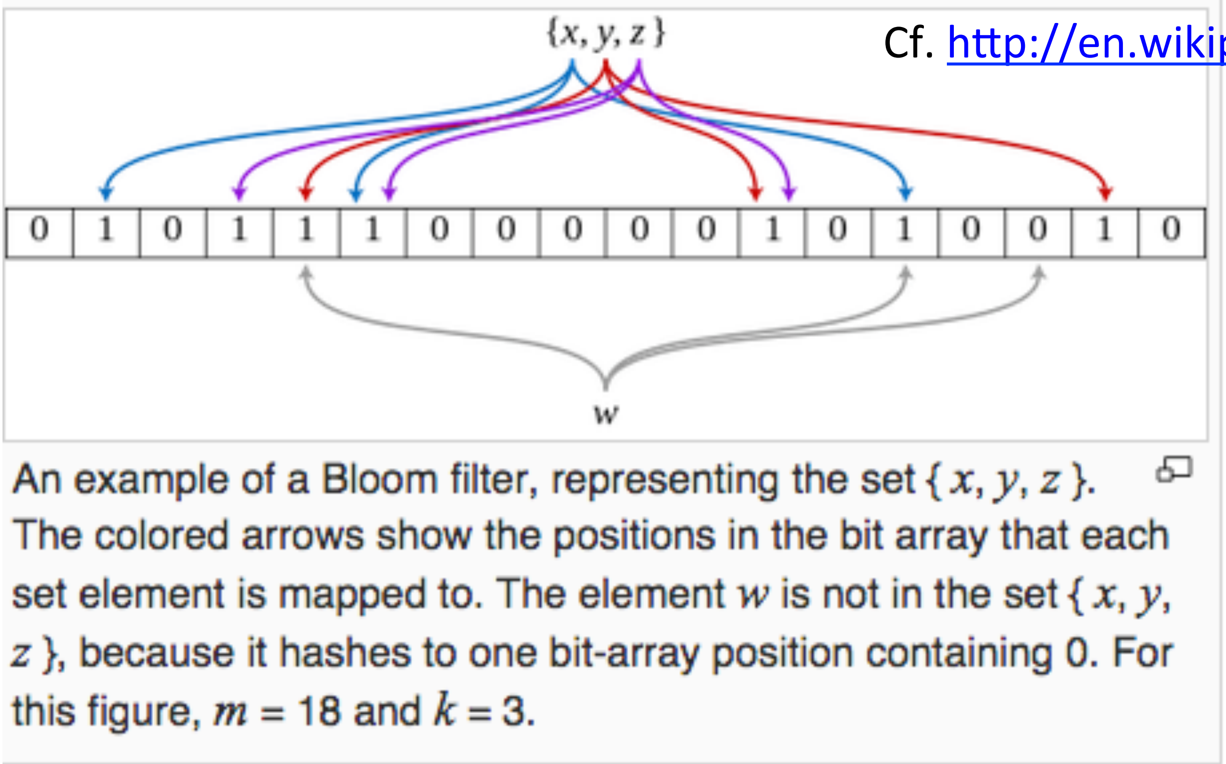
- Resolve limitation of RTM with Software and Hardware hybrid Transactional Memory (TM)
- Due to 'no escape' RTM limitation, trying five TM strategies in sequence.
- No crash, no fault tolerance
  - Higher performance with 5 strategies
    - SpecSW is closer to so called transaction. It consults Contention Manager for commit decision.
  - Simple commit with lock, considering lower impact to concurrently running RTM transaction
- Modification of InvalSTM (software TM by the co-auther Gottschlich) to make it HTM hybrid.

Invyswell Design

- Hardware TM: LWHW(small Tx), BMHW
- Software TM: SpecSW, IrrevocSW (always commit), SglSW (Giant lock)



Bloom filter



An example of a Bloom filter, representing the set { x, y, z }. The colored arrows show the positions in the bit array that each set element is mapped to. The element w is not in the set { x, y, z }, because it hashes to one bit-array position containing 0. For this figure, m = 18 and k = 3.

Cf. http://en.wikipedia.org/wiki/Bloom_filter

- False positive matches are possible, but false negatives are not
  1. A strong space advantage: 1% error with ~9.6 bits/element.
     Additional ~4.8 bits/elem. reduces ~1% false-positive rate
  2. O(k) for add items or to check
  3. Can not remove an element

Optimization in specSW

- Avoiding abort of inflight HW transaction
  - Reading lock for inflight transaction list aborts hardware transactions (RTM)
  - Using slotted array to eliminate lock in linked list
- Reducing conflict
  - Changing commit order
    - Lock -> commit -> invalidate -> unlock
    - Recording read location in bloom filter if the read is not in write-set
      (RAW, True dependency: http://en.wikipedia.org/wiki/Data_dependency)

Other optimizations discussed in section 6.

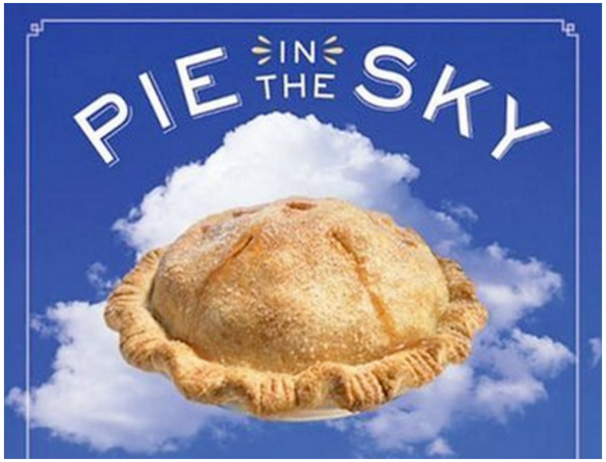
- Fail-Fast mode with priority, read-only transaction

Bug in RTM HW Sandboxing (Sec. 5.2)

- "loop-hole", an unlikely sequence of events in which
  - (1) mutually inconsistent reads cause a spurious memory write,
  - (2) which overwrite an address later used as the target of an indirect jump in that same transaction,
  - (3) thereby causing a jump to a location that happens to contain either an _xend (commit transaction) instruction, or immediate data that looks like one.
  - Executing this instruction without the final commit lock check could prematurely commit an inconsistent set of changes
- Solution:
  - Check the commit_lock before doing an indirect jump (hand code)
  - No indirect jumps: also preventing buffer overflow vulnerability

Conclusion

- Architecture researches sometimes direct to a brute forth or a combination of existing solution.
- They may seem to show presence in evaluation of improvement in benchmarks, instead of proposing a novel algorithm. … 5% Club …
- I think it is due to they do not want to change the important component ‘CPU, or LSI Chip’. Some have been trying it, ending with ‘Pie in the Sky’, except ones eventually introduced in real LSI.



References (except ones inline)

- Haswell microarchitecture: http://en.wikipedia.org/wiki/Haswell_%28microarchitecture%29
- Intel Haswell TSX page: https://software.intel.com/en-us/blogs/2012/02/07/transactional-synchronization-in-haswell
- Haswell instruction set manual: https://software.intel.com/sites/default/files/m/9/2/3/41604

- An Efficient Software Transactional Memory Using Commit-Time Invalidation, Justin E. Gottschlich, et.al: http://justingottschlich.com/content/cgo10_inval.pdf
- An Evaluation of Intel's Restricted Transactional Memory for CPAs: https://kar.kent.ac.uk/36939/1/9780956540973_020.pdf

- Radix Tree: http://en.wikipedia.org/wiki/Radix_tree
- Apaptive Radix Tree: ARTful http://www3.informatik.tu-muenchen.de/~leis/papers/ART.pdf