1. Do a brief search on “cache line size”. What are common sizes?

32 – 64 Bytes

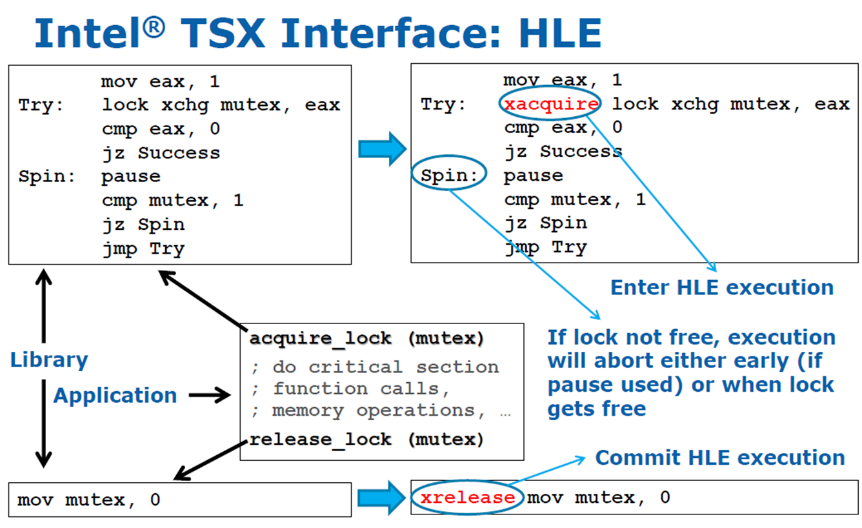
2. How do you get independent data elements (and locks) to be allocated on different cache lines?

Pad extra bytes between the data elements.

3. What is the key idea in the MCS lock as compared to a regular spin lock?

Threads spin on independent, private locks that can be cached without coherency traffic.

Consider the left-hand side of the diagram (i.e., existing code) for acquire\_lock(mutex) for questions 7 to 9.



x86 specifics:

lock – a prefix byte that tells the memory controller to make an instruction’s data memory accesses atomic

xchg – exchange the contents of a register and the contents of a memory location

eax – a CPU register (extended accumulator)

pause – hint to processor to delay instruction execution from this thread; also acts as a memory barrier; acts as a no-op for processors that do not implement SSE2

xacquire and xrelease – act as no-ops for processors that do not implement TSX

(source: Ravi Rajwar and Martin Dixon, IDF 2012)

4. Why is the code pattern called test-and-test-and-set?

cmp is a read (test) while lock xchg is a test-and-set type of atomic read-modify-write instructions.

5. What is the performance benefit of test-and-test-and-set?

Test part of the loop requires no memory traffic.

6. What is the key idea in hardware transactional memory?

Track changes at the cache line level, and if no interfering accesses commit all cache line updates as a single atomic action.