CACHE COHERENCE SIMULATOR USING CONTECH

RACHEL YANOVSKY AND ASHWIN RAGHAVACHARI

WHAT WE DID

- Simulated multiprocessor snooping-based system
  - L1 4-way associative LRU cache
- MESI cache coherence protocol
- Split transaction bus
- Supports lazy optimistic transactional memory

USING CONTECH

- Tool that provides memory traces in the form of task graph
- Our simulator supports the use of barriers and basic locks
- Our simulator reads in any taskgraph

PROCESS THE MEMORY REQUEST

- Check if it is a read or a write
  - For example, a mem copy would be a read and a write
- Separate into 8 byte blocks
- Decide whether a message needs to be sent on the bus
  - Depends on what state it is in

SEND MESSAGE ON BUS

- Each core sends it current memory request to the split bus
- The bus arbitrator uses round robin to deal with bus contention
  - However memory response always gets priority
- Bus responds with an ACK when acknowledged
  - NACK sent if request table is filled, and processor must retry
- Pending requests stay in request table until fulfilled
  - Current processor continues while pending
- All other caches see this message and act accordingly based on MESI protocol

UPDATE CACHE

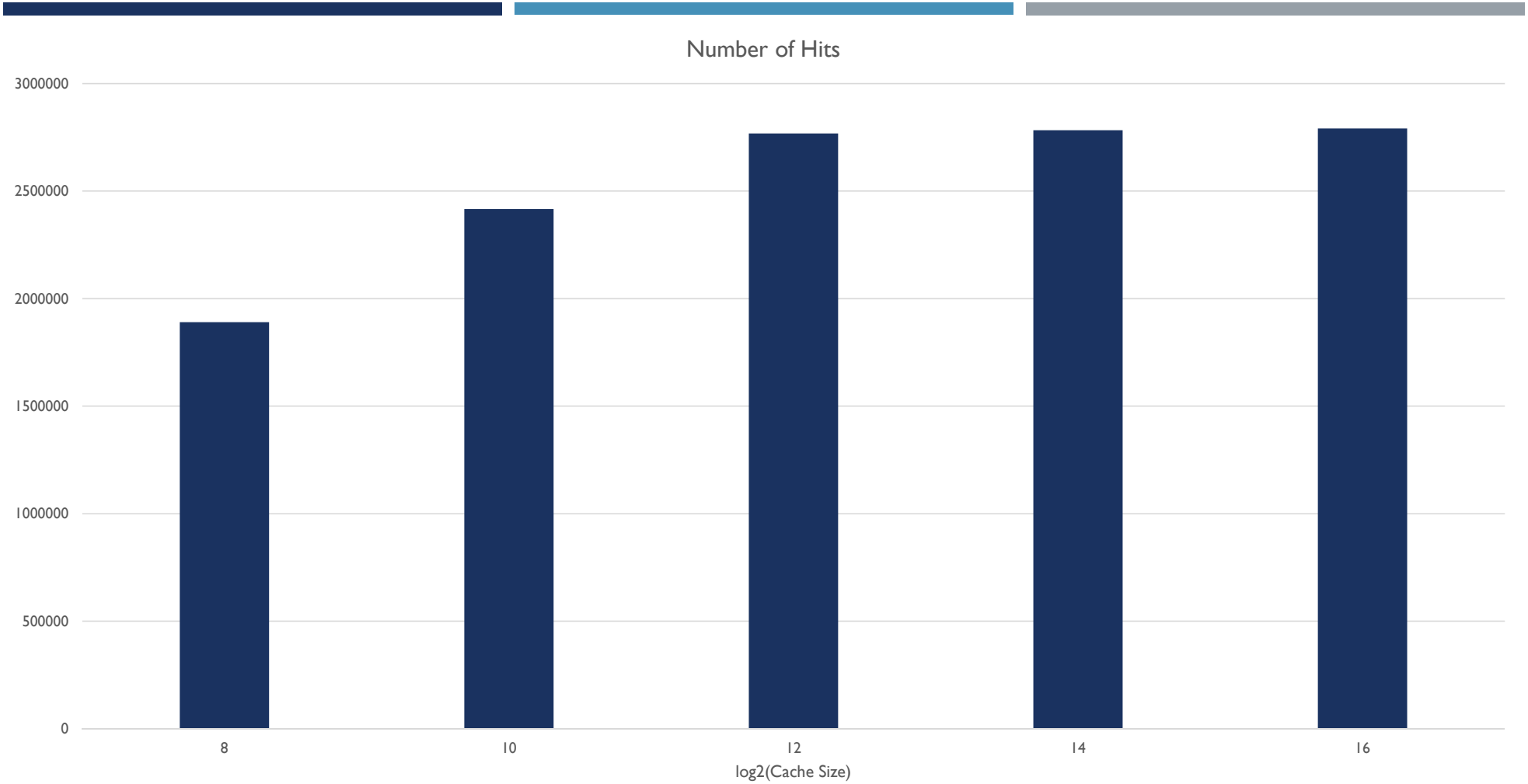
- At this point the processor can finally update its cache
- Puts the value in the correct cache line and sets the status (MESI)
- If an eviction had to occur it would flush to memory over the bus

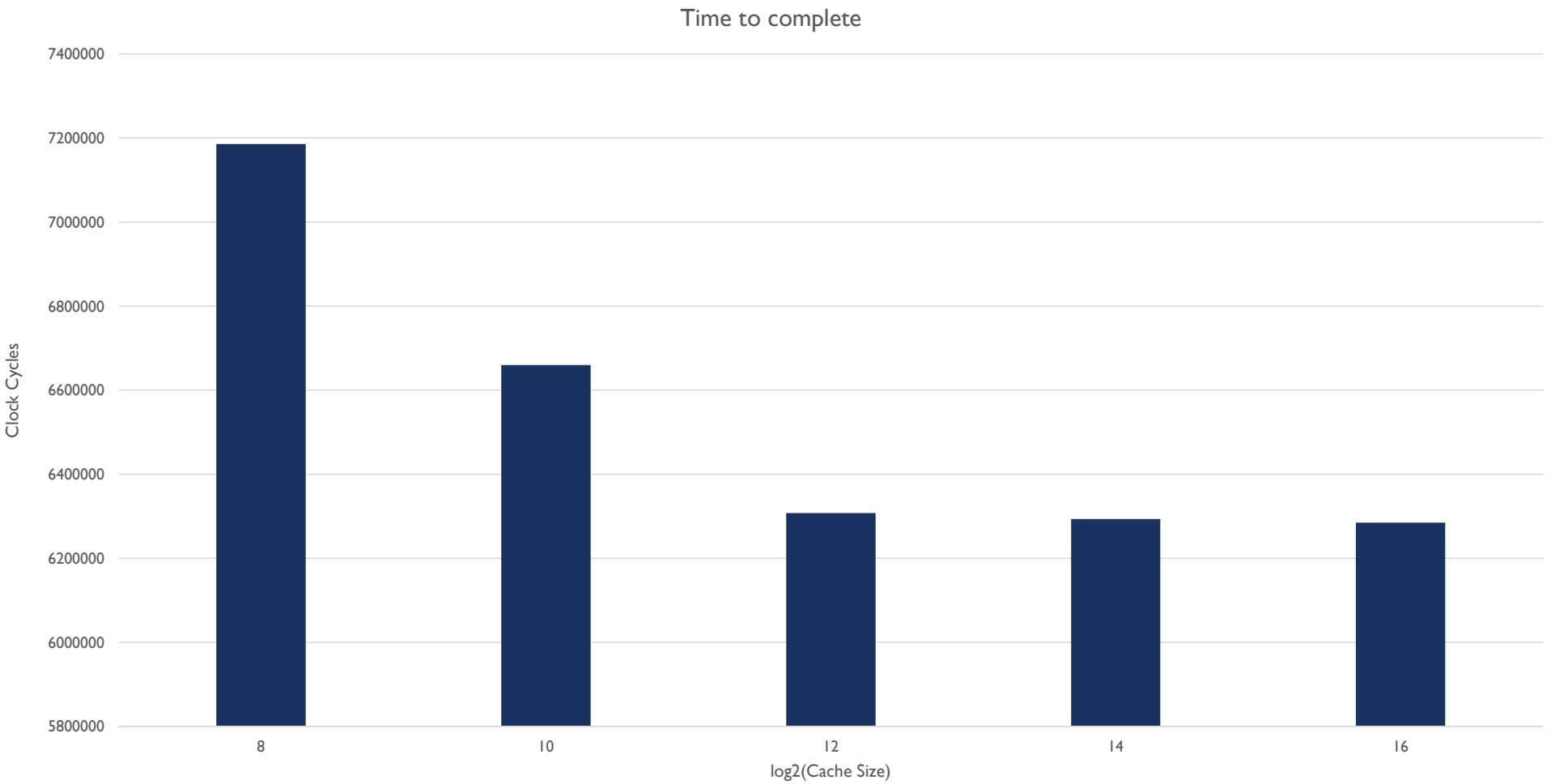
TRANSACTIONAL MEMORY

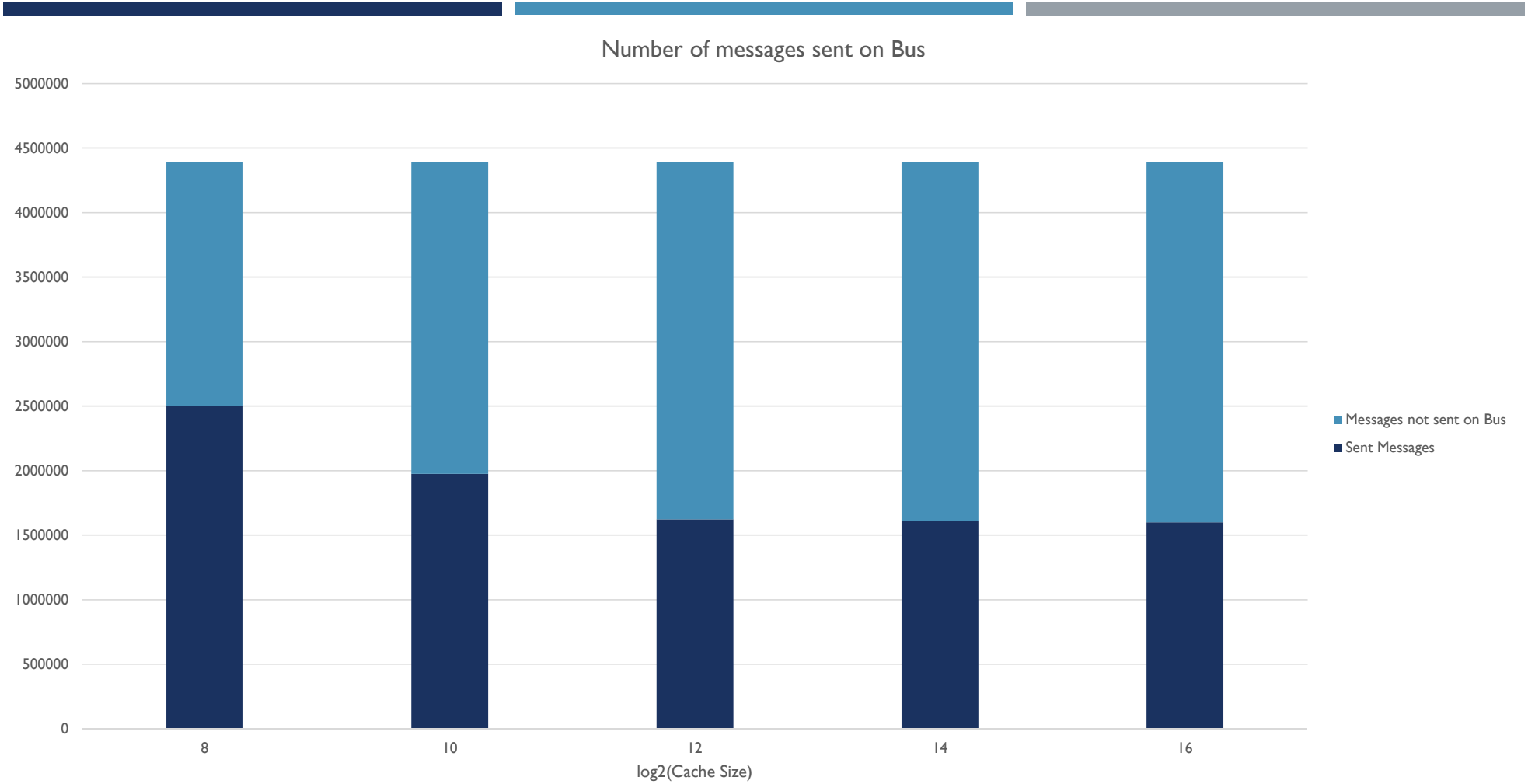
- We created our own trace – mimicking the bank example from class
  - Different cores try to access the same memory address
  - All of the memory addresses in our example where on the same cache line!
- Each core saves the read and write data in a log
- Upon a commit, send all the requests on the bus
- The bus will broadcast a message and the other cores will have to restart accordingly

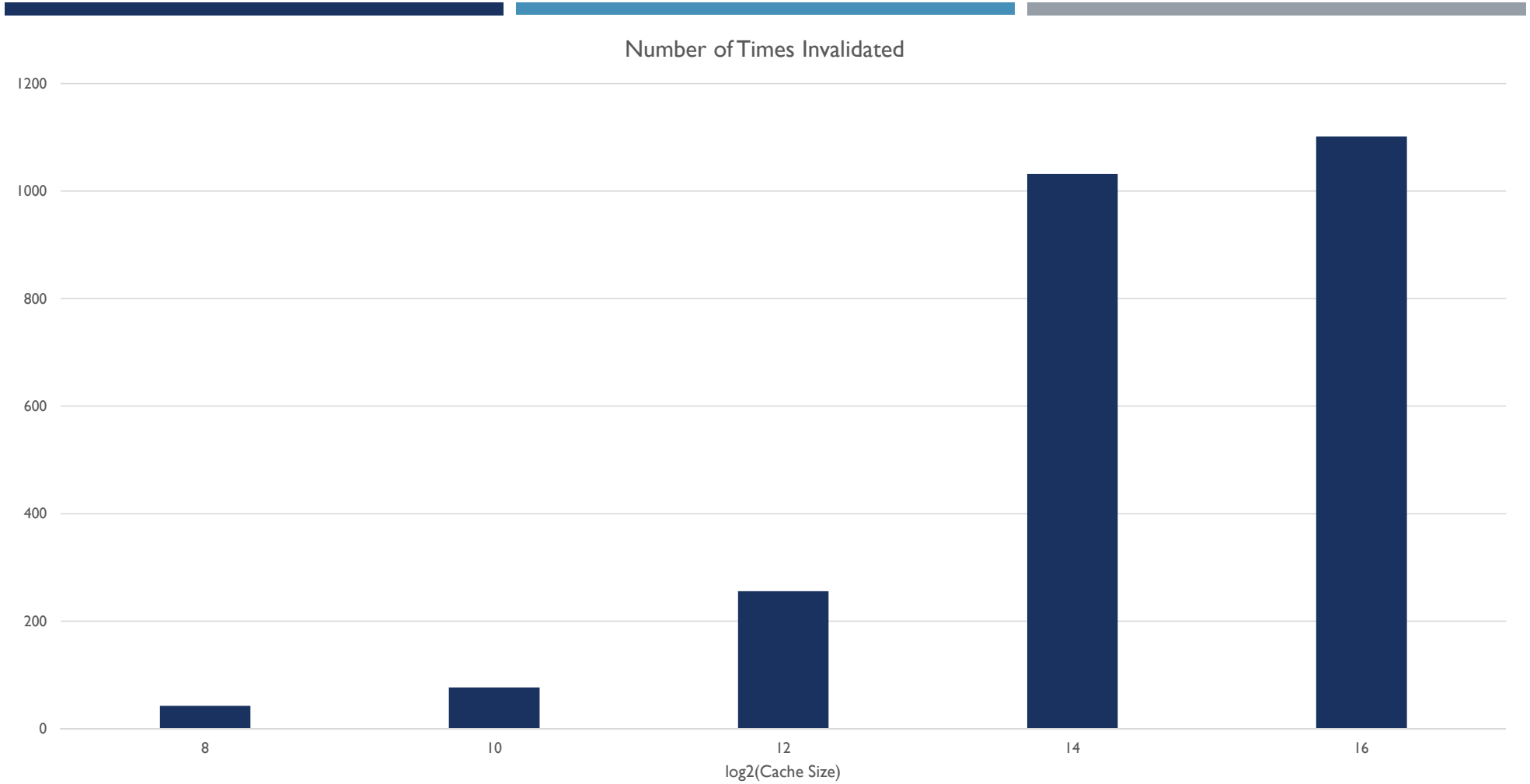
RESULTS

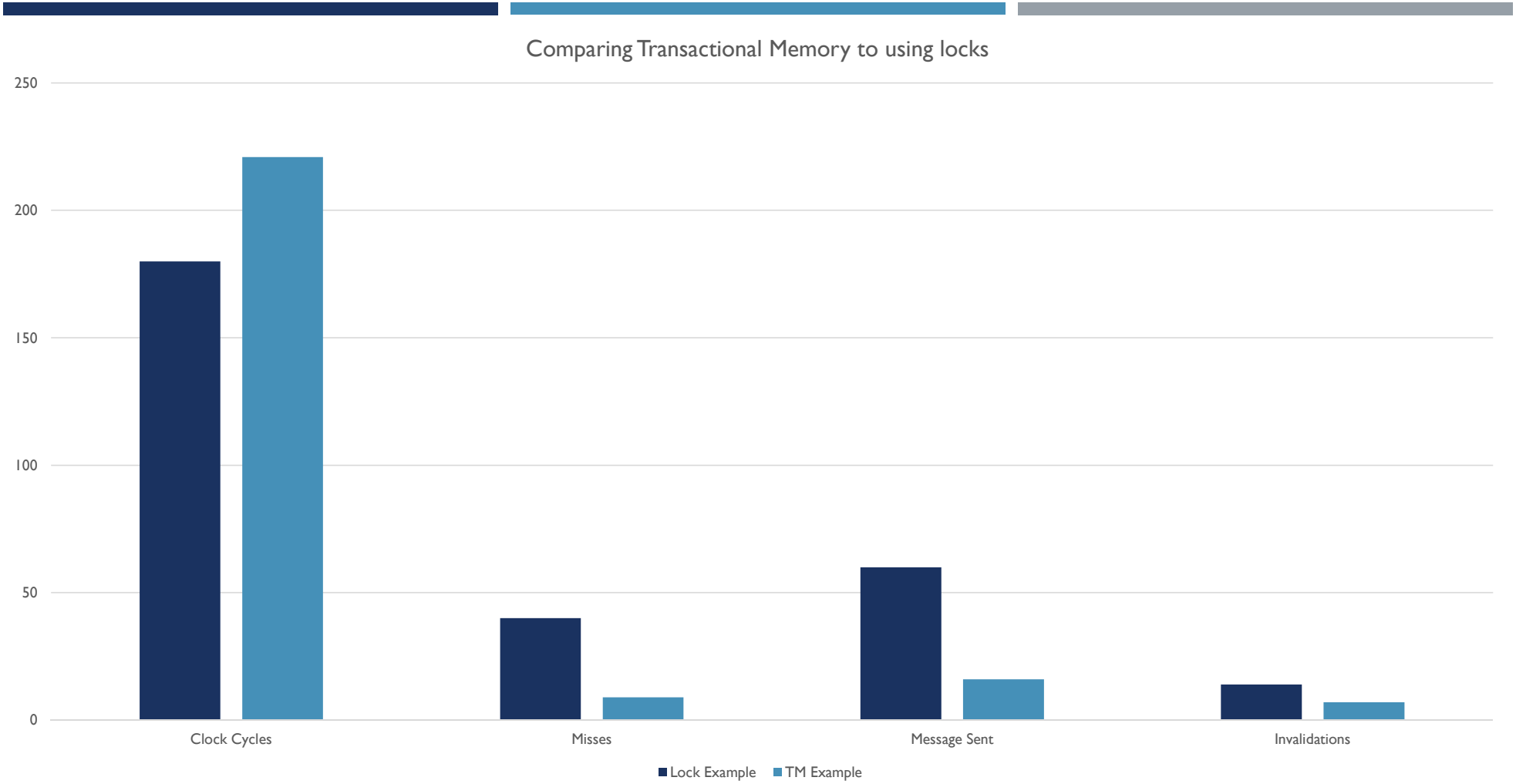
ALL USE THE SAME TASKGRAPH - CHECK OUT OUR WEBSITE FOR THE SAME ANALYSIS WITH OTHER GRAPHS


Number of Hits
3000000
2500000
2000000
1500000
1000000
500000
0
8
10
12
14
16
log2(Cache Size)


Time to complete
7400000
7200000
7000000
6800000
6600000
6400000
6200000
6000000
5800000
Clock Cycles
8
10
12
14
16
log2(Cache Size)


Number of messages sent on Bus
5000000
4500000
4000000
3500000
3000000
2500000
2000000
1500000
1000000
500000
0
8
10
12
14
16
log2(Cache Size)
Messages not sent on Bus
Sent Messages


Number of Times Invalidated
1200
1000
800
600
400
200
0
8
10
12
14
16
log2(Cache Size)


Comparing Transactional Memory to using locks
250
200
150
100
50
0
Clock Cycles
Misses
Message Sent
Invalidations
Lock Example
TM Example

QUESTIONS?