CS 407 : Applied Formal Methods
Project Proposal
Formal Verification of MESI Cache Coherency

Luke Marzen

October 26, 2023

This project is a solo effort by *Luke Marzen*, team *Cache Me Outside*. In modern multi-processor systems, each processor has its own cache used to minimize data access times (Figure 1). To maintain a *coherent*, up-to-date, view of shared memory a cache coherency protocol constrains the permitted transactions. One of the most common of these protocols is called MESI. MESI was originally proposed by (Papamarcos and Patel 1984) and is named after the four states that a cache-line can be assigned: *Modified*, *Exclusive*, *Shared*, and *Invalid*. The MESI protocol can be defined by a finite-automata shown in Figure 2. Each private cache is connected to a *caching-agent* (i.e. a processor) as well as a bus which monitors all other cache transactions via *Snooping*. Formally, we can define coherency with the following LTL specification (Harrison 2010),

$$\Box\left[\begin{array}{l}\forall i.\ \big(\text{Cache}(i) \in \{\texttt{Modified}, \texttt{Exclusive}\}\big) \\ \qquad \rightarrow \forall j.\ \big(\neg(j = i) \rightarrow\ \text{Cache}(j) = \texttt{Invalid}\big)\end{array}\right].$$

The primary objective of this project is to verify the MESI Cache Coherency Protocol through *Model Checking*. To this end, SPIN will be used to explore the state space and validate that the coherency property is not violated. There are a few additional considerations when creating the model. MESI is modular in the sense that it can be extended beyond the state space of just two caching-agents, with the practical limits being communication

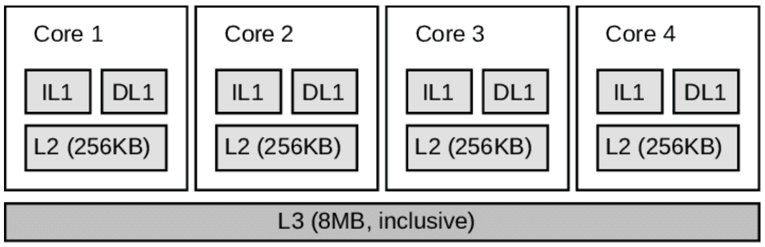

Figure 1: Cache architecture of the Intel Core i7 4790 processor. (Nakamoto 2018)

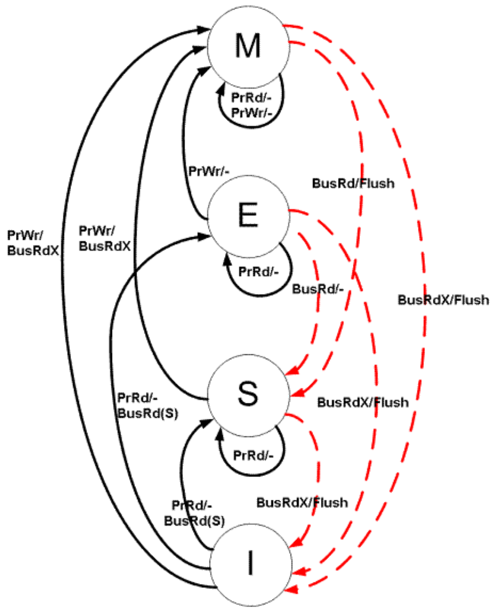

Figure 2: State transition diagram for Illinois MESI protocol. Transitions: Processor initiated transactions (black), Bus initiated transaction (red). (Culler, Singh, and Gupta 1998)

overhead and physical footprint of the processor. To verify as many configurations as possible, the number of caching agents N will be incremented until the state space becomes too large to model in SPIN. While the size of the caches could vary across caching-agents, in practice it is more common for caches at the same level to be symmetrical. This simplifies creating the model in SPIN and limits the number of configurations.

The planned timeline is shown in Table 1. Changes will be tracked with git. Work will be done on a development branch, before being later merged into main. The git repository will be structured as follows,

```
applied-formal-methods-final-project-cache-me-outside
|--reports
|--models/*.pml
|--results
```

TABLE 1 Timeline

Week 10	Submit Proposal
Week 11	Begin implementing models in SPIN.
Week 12	Prepare for midterm project report/presentation
Week 13	Complete model implementations in SPIN.
Thanksgiving Break	–
Week 14	Analyze results
Week 15	Prepare for project presentation

References

Culler, David, Jaswinder Pal Singh, and Anoop Gupta. 1998. *Parallel Computer Architecture: A Hardware/Software Approach.* San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. ISBN: 9780080573076.

Harrison, John. 2010. "Formal Methods at Intel — An Overview." Intel Corporation. Presentation at the Second NASA Formal Methods Symposium, NASA HQ, Washington DC, 14th April 2010, 09:00–10:00. Accessed October 25, 2023. https://shemesh.larc.nasa.gov/NFM2010/talks/harrison.pdf.

Nakamoto, Aoi. 2018. "W-Shield: Protection against Cryptocurrency Wallet Credential Stealing." In *Workshop on Security and Privacy in E-Commerce 2018,* 71–107. Yokohama, Japan, June.

Papamarcos, Mark S., and Janak H. Patel. 1984. "A Low-Overhead Coherence Solution for Multiprocessors with Private Cache Memories." In *Proceedings of the 11th Annual International Symposium on Computer Architecture,* 348–354. ISCA '84. New York, NY, USA: Association for Computing Machinery. ISBN: 0818605383. https://doi.org/10.1145/800015.808204. https://doi.org/10.1145/800015.808204.