TOP LEVEL DESIGN OF SUN T1 SIMULATOR

Amritam Sarcar

UT El Paso

This document gives the top-level design features that are required to simulate the SUN T1 multiprocessor.

OVERVIEW OF SUN T1 MULTIPROCESSOR:

The Sun T1 multi-core multi-processor was introduced in the year 2005 essentially as a server processor. The cool thing about this processor was that it completely focused on exploiting thread-level parallelism rather than data-level parallelism. However, the purpose of this project is not to focus on exploitation of TLP.

The SUN T1 has 8 cores for each processor. With every core it has a dedicated cache attached called L1 cache. Each of these caches interacts with each other essentially through a global/common bus, which is termed as crossbar switch. They have integrated L2 caches that are associated to each memory bank. The number of L2 caches is four and to each of these L2 caches, a directory is associated.

It is evident that since SUN T1 is a multiprocessor CPU, it requires some kind of coherence protocol to ensure the cardinal properties of coherency. To support this, it uses directory-based protocols.

OVERVIEW OF DIRECTORY-BASED PROTOCOL:

The directory-based protocol is an example of a cache-coherence protocol. It is used to maintain cache coherency across caches. It is mainly used in distributed memory architecture. The most important principle of this kind of protocol is: *communicate with the main/home memory for cache consistency when required.* The following state-diagram explains them in more detail:

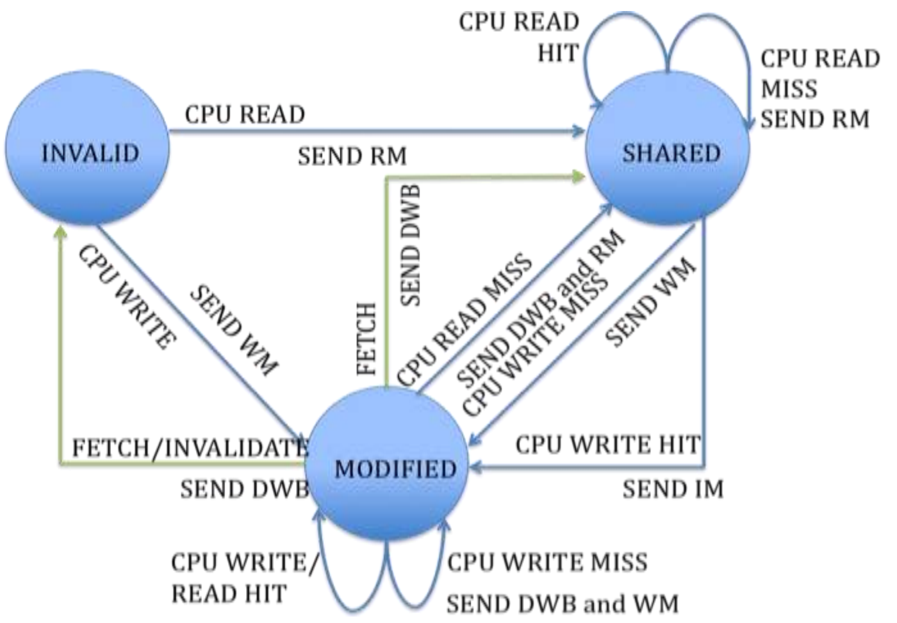


Fig.1. Describes the transitions within individual caches. The transitions shown in blue are the ones that originate in the cache, whereas the ones in green are transitions due to messages coming from the home directory. The abbreviations RM, WM, DWB and IM are read miss, write miss, data write back and invalidate message respectively.

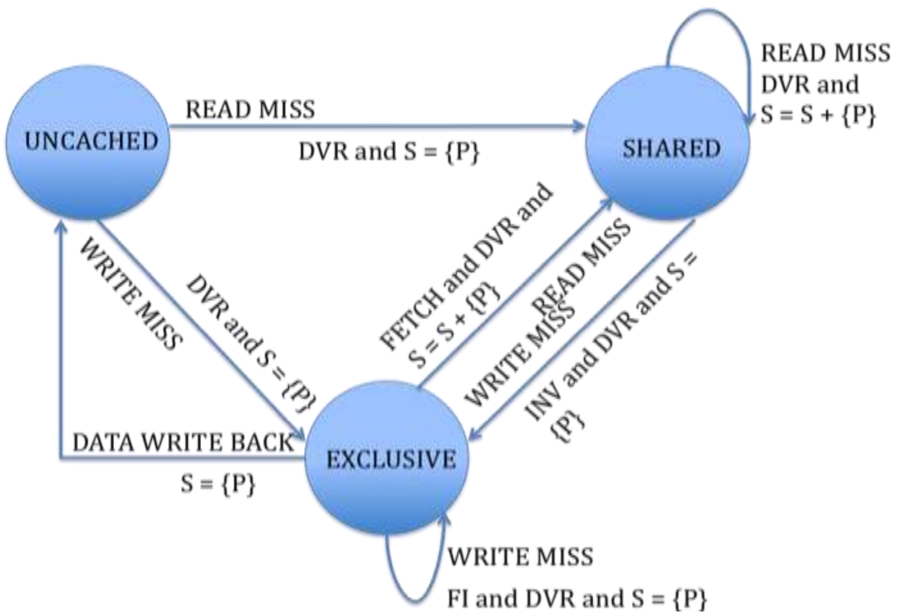


Fig.2. Describes the transitions within a directory. The abbreviations DVR, INV and FI are data value reply, invalidate and fetch/invalidate message respectively.

ARCHITECTURAL OVERVIEW OF SUN T1: (*from the implementation point of view*)

Before we actually simulate the working of the different components, it is required to design each of the components and understand their interactions between them.

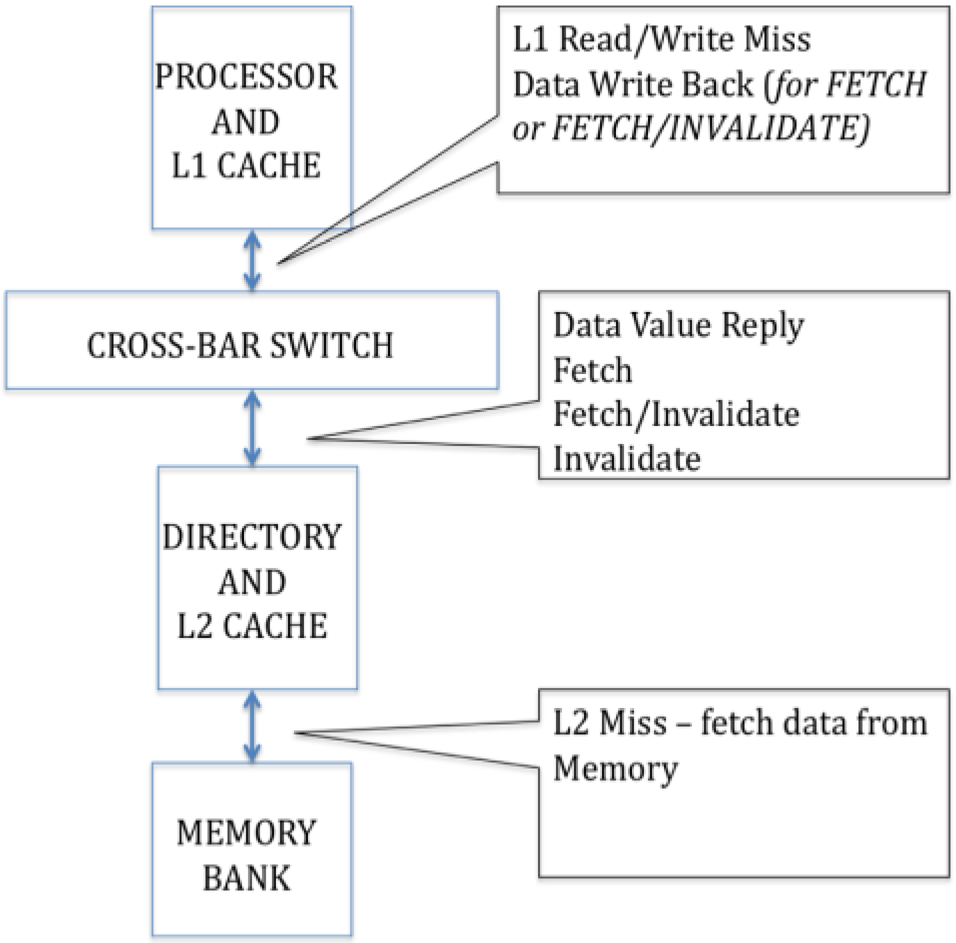


Fig. 3. A schematic diagram showing the main components of the architecture and their communication with each other. All the communications are bi-directional. This diagram shows the type of messages that interact with each component. It essentially depicts the originating place for each of the message.

Diagram 3 gives us the top-level design of the simulator. Though this diagram looks simple, however, when we zoom into each of the components, we realize that a lot of interconnection happens.

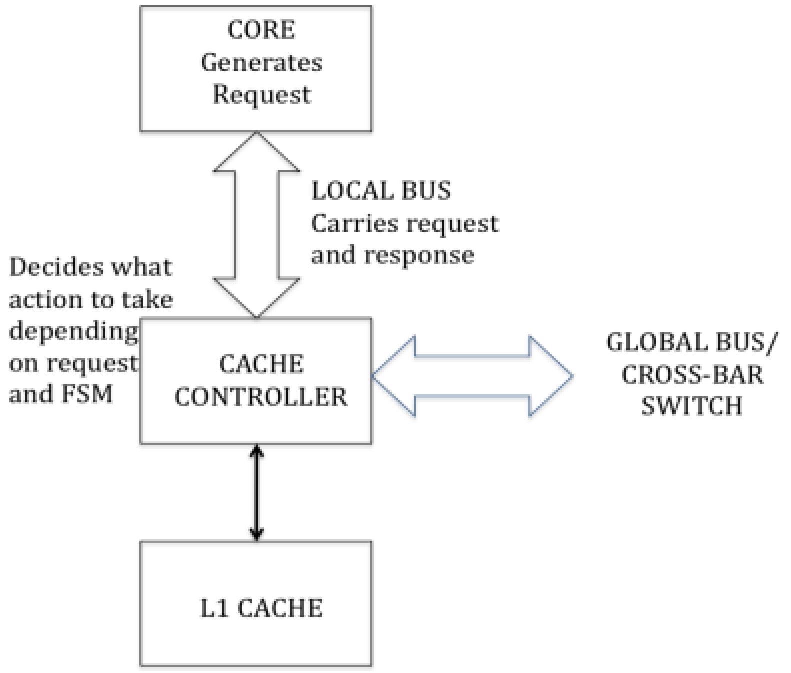


Fig 4.1 The internal diagram of the CORE and CACHE. The components shown here are actually implemented.

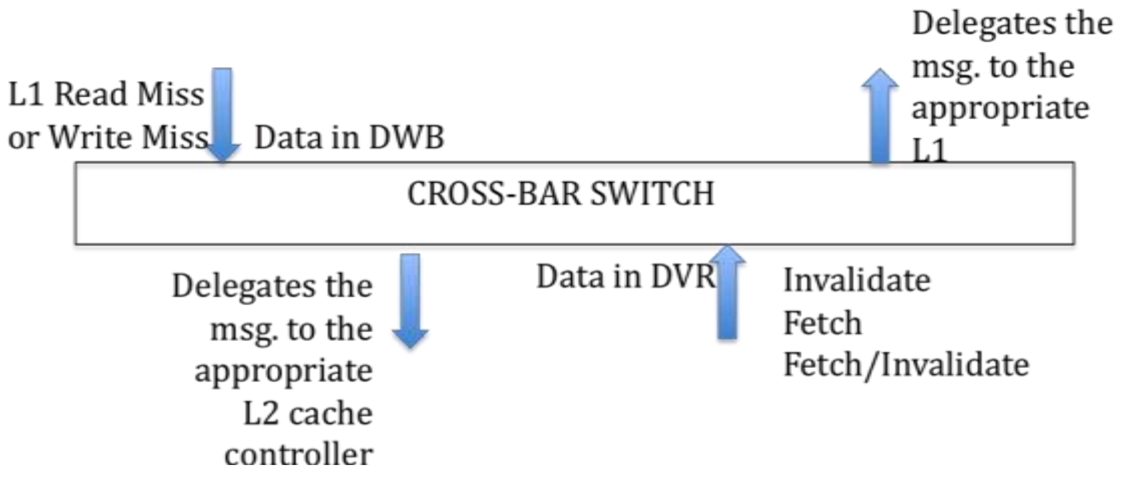


Fig.4.2 The crossbar switch plays an important role of delegating messages between L1 cache controllers and Directories. This is important because the implementation takes care of the fact that L1 cache controller should not know which is the home directory.

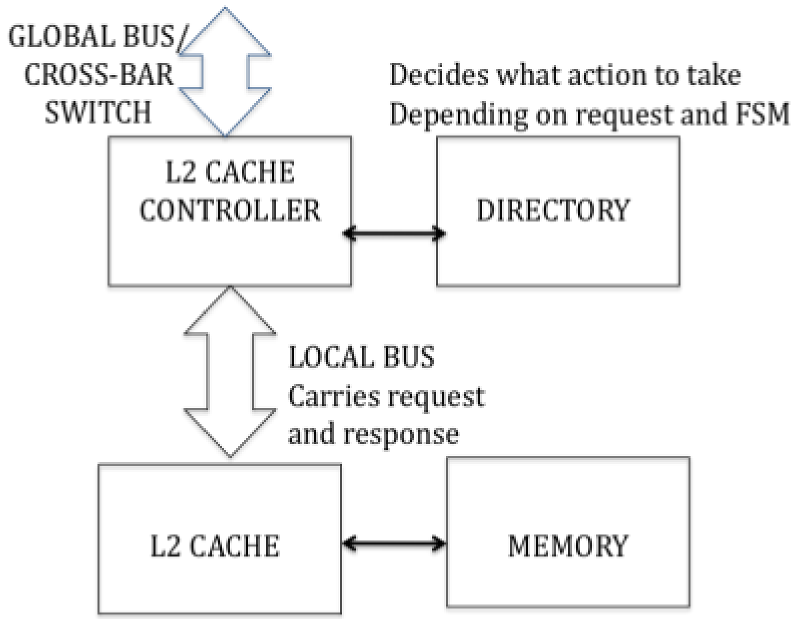
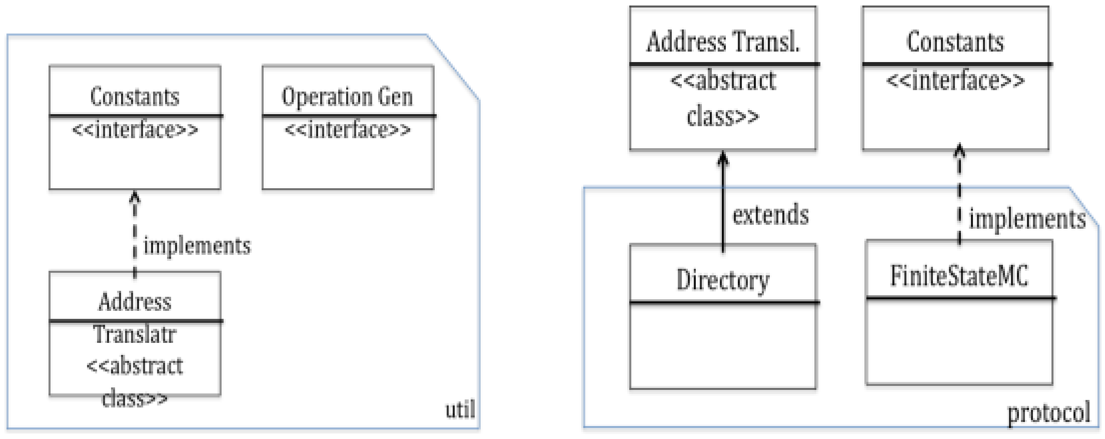
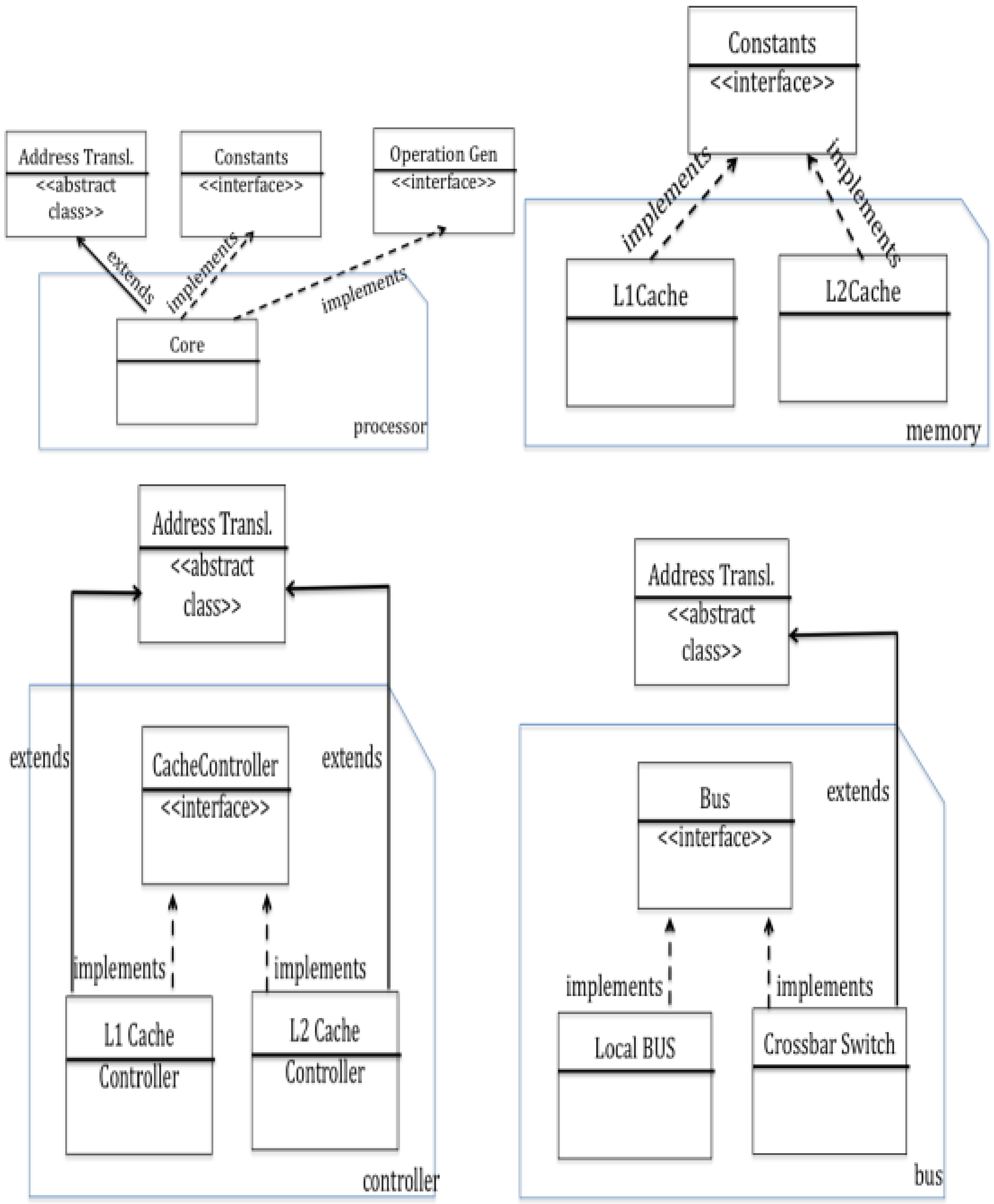


Fig.4.3. The inner components of the L2 Cache and the Directory. On receiving message from the crossbar switch, it delegates to the directory to know what actions are to be taken. As per the action, L2 cache controller fetches data from the L2 Cache or again sends messages to the Global bus.

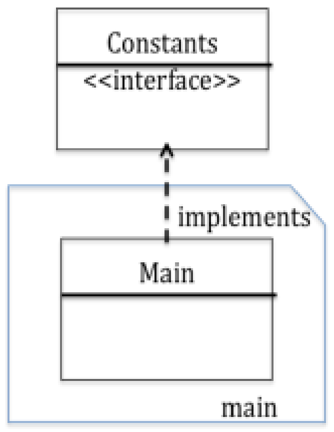
PRESENT STATUS:

The core implementation has been complete, though extensive test cases is required to be generated for being 100% correct. The implementation is being carried out in Java. The class diagram as per date is given below.





Address Transl.
<<abstract class>>
Constants
<<interface>>
Operation Gen
<<interface>>
extends
implements
implements
Core
processor
Constants
<<interface>>
implements
implements
L1Cache
L2Cache
memory
Address Transl.
<<abstract class>>
extends
CacheController
<<interface>>
extends
implements
implements
L1 Cache Controller
L2 Cache Controller
controller
Address Transl.
<<abstract class>>
Bus
<<interface>>
extends
implements
implements
Local BUS
Crossbar Switch
bus



INITIAL RESULTS:

The following output is the initial result on running the same core only 10 times. Just to check whether accessing L1, L2, directory is working.

```
Request generated by coreno :: 0
Request is :: 0x1c9:1
Request recieved by cachecontroller id :: 0 is :: 0x1c9:1
Message recieved by crossbar-switch ::0:RM:457
Transaction READ complete for core :: 0

Request generated by coreno :: 0
Request is :: 0xe9:1
Request recieved by cachecontroller id :: 0 is :: 0xe9:1
Message recieved by crossbar-switch ::0:RM:233
Transaction READ complete for core :: 0

Request generated by coreno :: 0
Request is :: 0x21:1
Request recieved by cachecontroller id :: 0 is :: 0x21:1
Message recieved by crossbar-switch ::0:RM:33
Transaction READ complete for core :: 0

Request generated by coreno :: 0
Request is :: 0x107:1
Request recieved by cachecontroller id :: 0 is :: 0x107:1
Message recieved by crossbar-switch ::0:RM:263
Transaction READ complete for core :: 0

Request generated by coreno :: 0
Request is :: 0xab:1
Request recieved by cachecontroller id :: 0 is :: 0xab:1
Message recieved by crossbar-switch ::0:RM:171
Transaction READ complete for core :: 0

Request generated by coreno :: 0
Request is :: 0x49:1
Request recieved by cachecontroller id :: 0 is :: 0x49:1
Message recieved by crossbar-switch ::0:RM:73
Transaction READ complete for core :: 0

Request generated by coreno :: 0
Request is :: 0x11f:1
Request recieved by cachecontroller id :: 0 is :: 0x11f:1
Message recieved by crossbar-switch ::0:RM:287
Transaction READ complete for core :: 0

Request generated by coreno :: 0
Request is :: 0x19e:1
Request recieved by cachecontroller id :: 0 is :: 0x19e:1
Message recieved by crossbar-switch ::0:RM:414
Transaction READ complete for core :: 0

Request generated by coreno :: 0
Request is :: 0x11d:0:ac1bdbec
Request recieved by cachecontroller id :: 0 is :: 0x11d:0:ac1bdbec
Message recieved by crossbar-switch ::0:WM:285
Transaction WRITE complete for core :: 0

Request generated by coreno :: 0
Request is :: 0xa:1
Request recieved by cachecontroller id :: 0 is :: 0xa:1
Message recieved by crossbar-switch ::0:RM:10
Transaction READ complete for core :: 0
```

FUTURE WORK:

A lot requires to be done starting from testing, actually generating random requests, analyzing and is possible, a visual front end.