**READ CASES:**

* **If all the Core’s have same address and read is asserted**

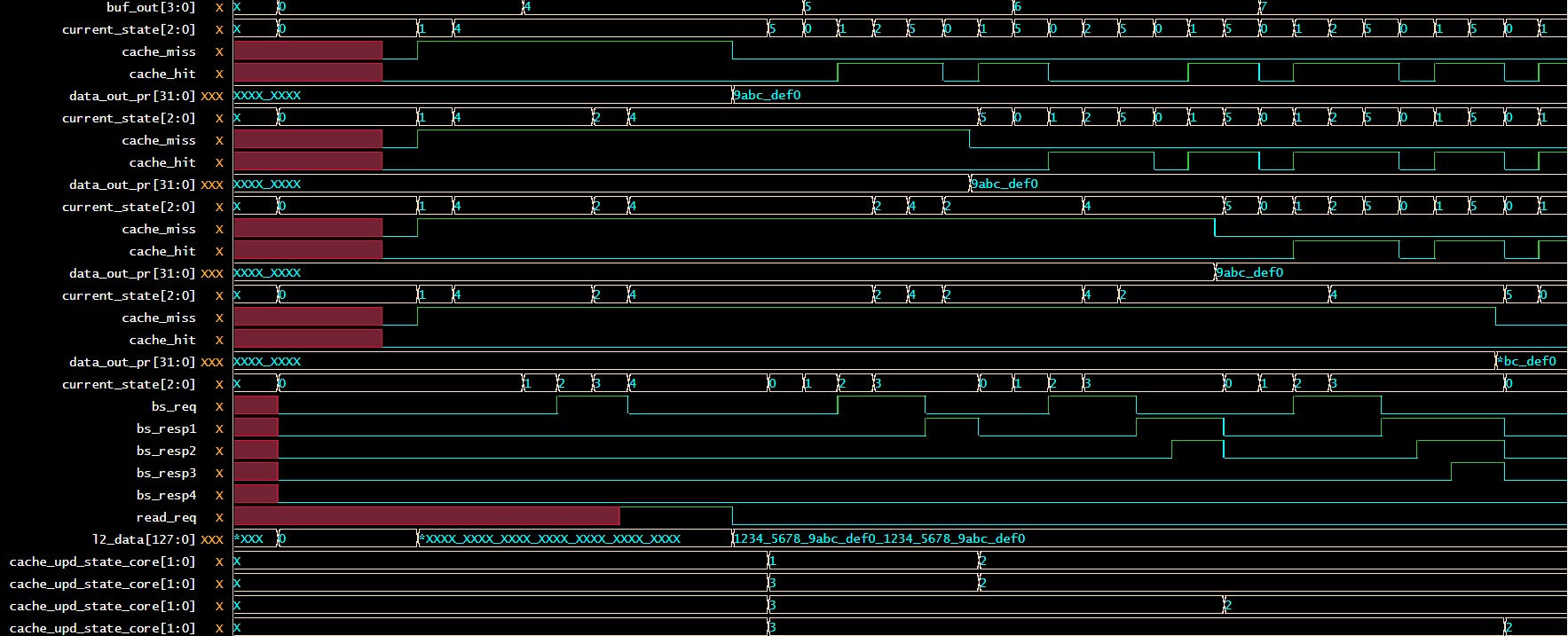
The processor Address given to all Core’s is **32’h11111000**

So, in this scenario Core 0 will not find the requested data at the given address it will be a compulsory cache miss and it will generate a request which will be send to arbiter. As it is given simultaneously so all core will indicate compulsory cache miss and generate a request flag, the arbiter will give grant on priority basis where Core 0 has the highest priority and Core 3 has the lowest priority. After given grant to the Requests, the request flag will be low as grant is given to them and it will be piled up in Fifo. The CCU (Cache Controlling Unit) will send a rd\_en signal through which it will orderly read the requests and work on them. As Core 0 has highest priority, CCU will work on it and send a **L1\_1\_CCU** to L1 Cache Controller of Core 0 to get the address, hit, miss and all credentials. Than it will work on it. As we are seeing a read miss scenario it will do snooping first as all are read miss at same address the response to **bs\_req** signal send by CCU will be 0. So, a **l2\_read\_req** signal will be send to L2 Cache to read data and then data will be sent to Core 0 Cache Controller and its state will be updated to **EXCLUSIVE**. After that for all other Core’s like Core 1,2,3 Cache Controller the data will be given by snooping to them as Core 0 has the data. It will give data to Core 1 and become in **SHARED** state. And Core 0 and Core 1 Cache Controller will send data to Core 2 and for Core 3 all Core’s 0,1,2 will have data as address is same will send data to it and all will be in **SHARED** state.

**Table:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Cores** | **Pr\_Addr[31:0]** | **Hit/miss** | **Request** | **Arbitration** | **Fifo\_in(data\_in)** | **Fifo\_out(buf\_out)** | **State transition** |
| Core\_0 | 32’h11111000 | Miss | 1’b1 | (first) | Written (1) | Read (1) | Exclusive, Shared |
| Core\_1 | 32’h11111000 | Miss | 1’b1 | (second) | Written (2) | Read (2) | Invalid, Shared |
| Core\_2 | 32’h11111000 | Miss | 1’b1 | (third) | Written (3) | Read (3) | Invalid, Shared |
| Core\_3 | 32’h11111000 | Miss | 1’b1 | (last) | Written (3) | Read (4) | Invalid, Shared |

**WAVES:**



**STATES:**

|  |  |
| --- | --- |
| **States** | **Binary Value (In waves)** |
| Modified | 0 |
| Exclusive | 1 |
| Shared | 2 |
| Invalid | 3 |

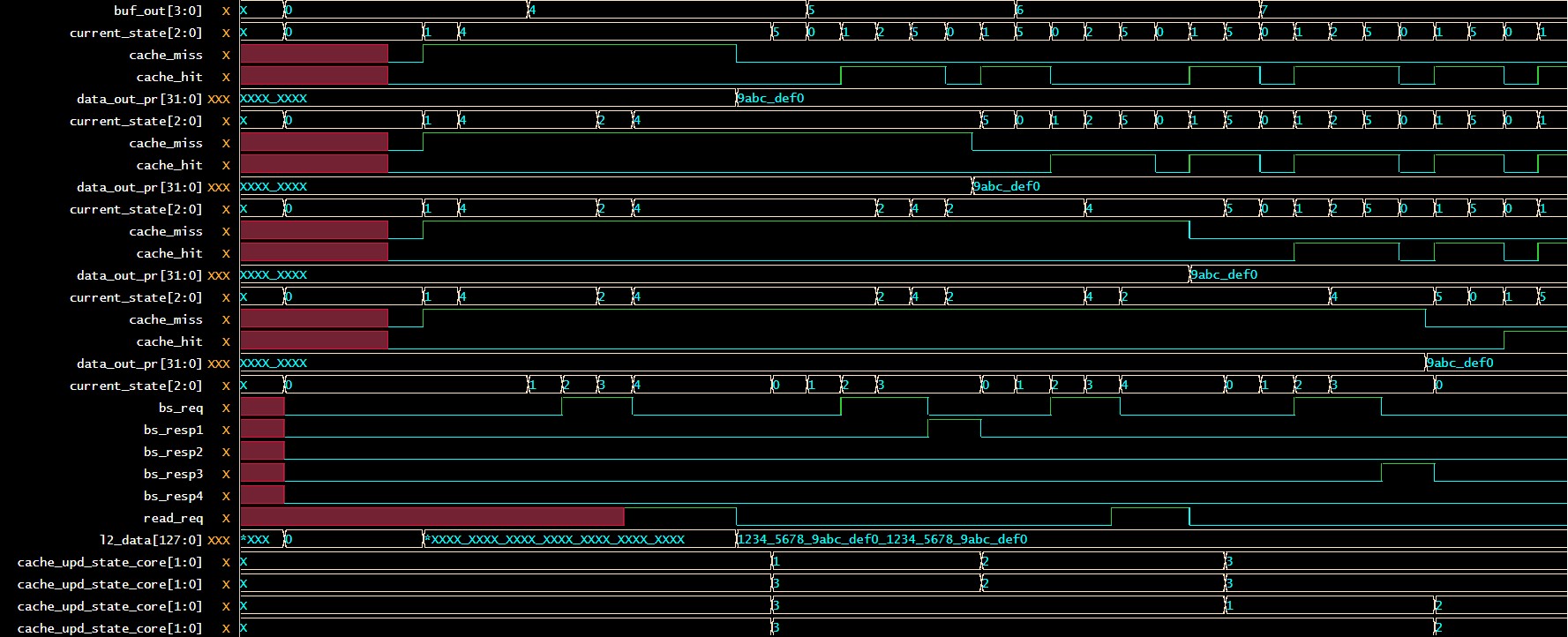
* **If Core 0 and 1 have same address, Core 2 and 3 have same address and read is asserted**

Core 0 and Core 1 get the same address while Core 2 and Core 3 get the same address. When read is asserted, Core 0 gets compulsory miss so request signal is asserted which gets to fifo after arbitration, the arbiter decides on priority bases and queue in all requests in fifo. While Core 0 being processed first, its request after being granted is handled by CCU, and through FSM (finite state machine) it goes through snooping and since no data has yet been loaded it moves to read from l2 state and fetch the data which is then passed on to data\_out\_pr. The other cores working simultaneously gets their grant and requested data is then snooped i.e. core 1 gets the data through snooping from core 0. However, note that Core 2 will have to repeat the same cycle as Core 0 since no same address is found and then Core 3 gets its data through snooping from Core 2.

**TABLE:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Cores** | **Pr\_Addr[31:0]** | **Hit/miss** | **Request** | **Arbitration** | **Fifo\_in(data\_in)** | **Fifo\_out(buf\_out)** | **State transition** |
| Core\_0 | 32’h11111000 | Miss | 1’b1 | (first) | Written (1) | Read (1) | Exclusive, Shared |
| Core\_1 | 32’h11111000 | Miss | 1’b1 | (second) | Written (2) | Read (2) | Invalid, Shared |
| Core\_2 | 32’h11110000 | Miss | 1’b1 | (third) | Written (3) | Read (3) | Exclusive, Shared |
| Core\_3 | 32’h11110000 | Miss | 1’b1 | (last) | Written (4) | Read (4) | Invalid, Shared |

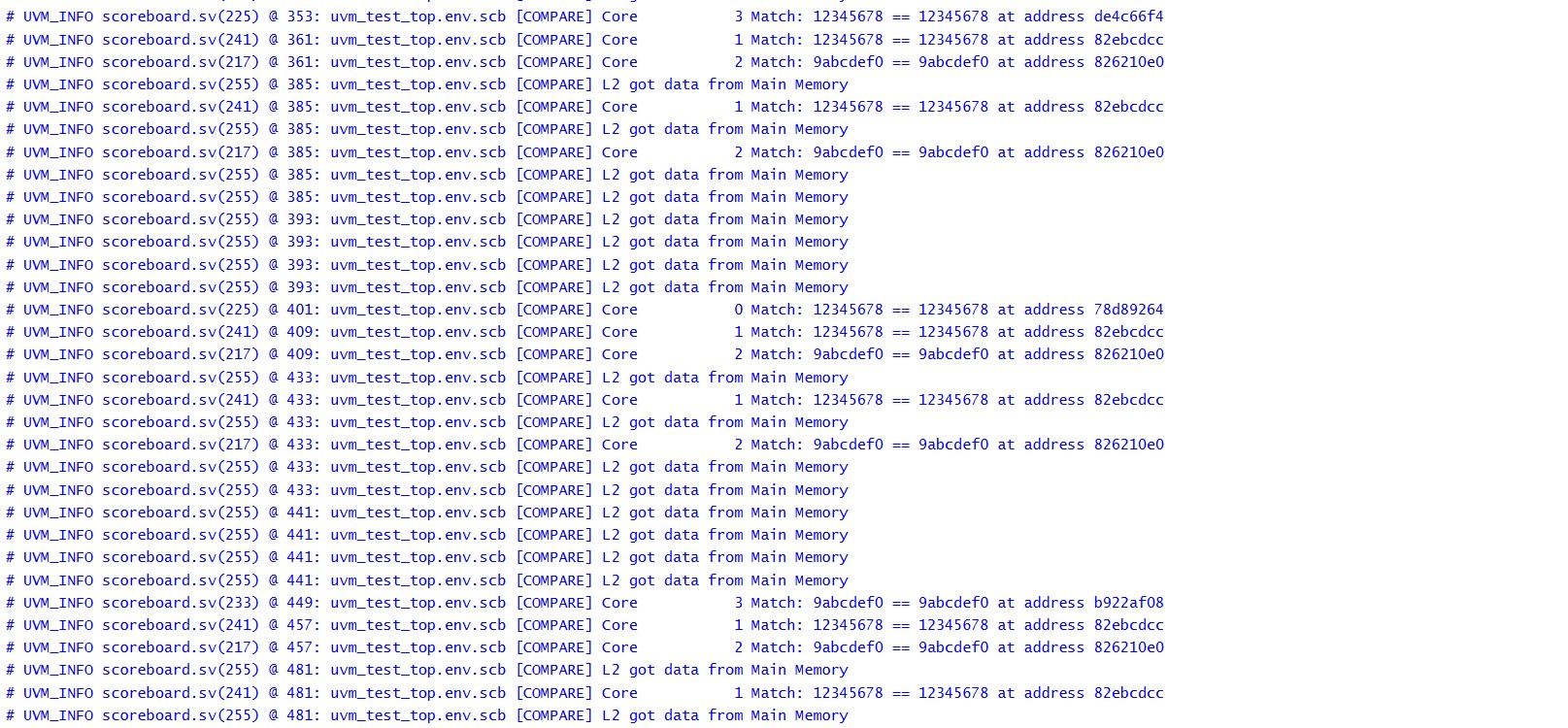
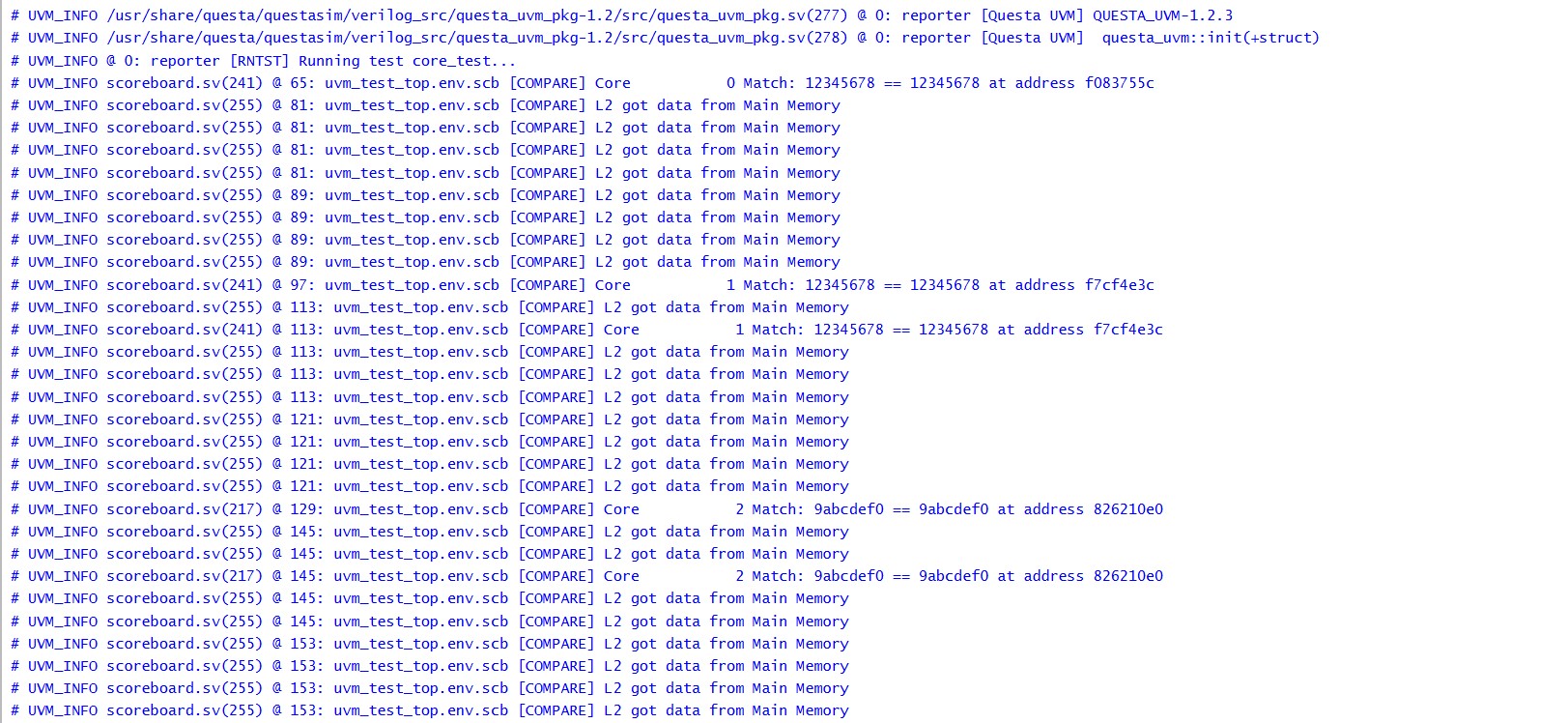
**WAVES:**



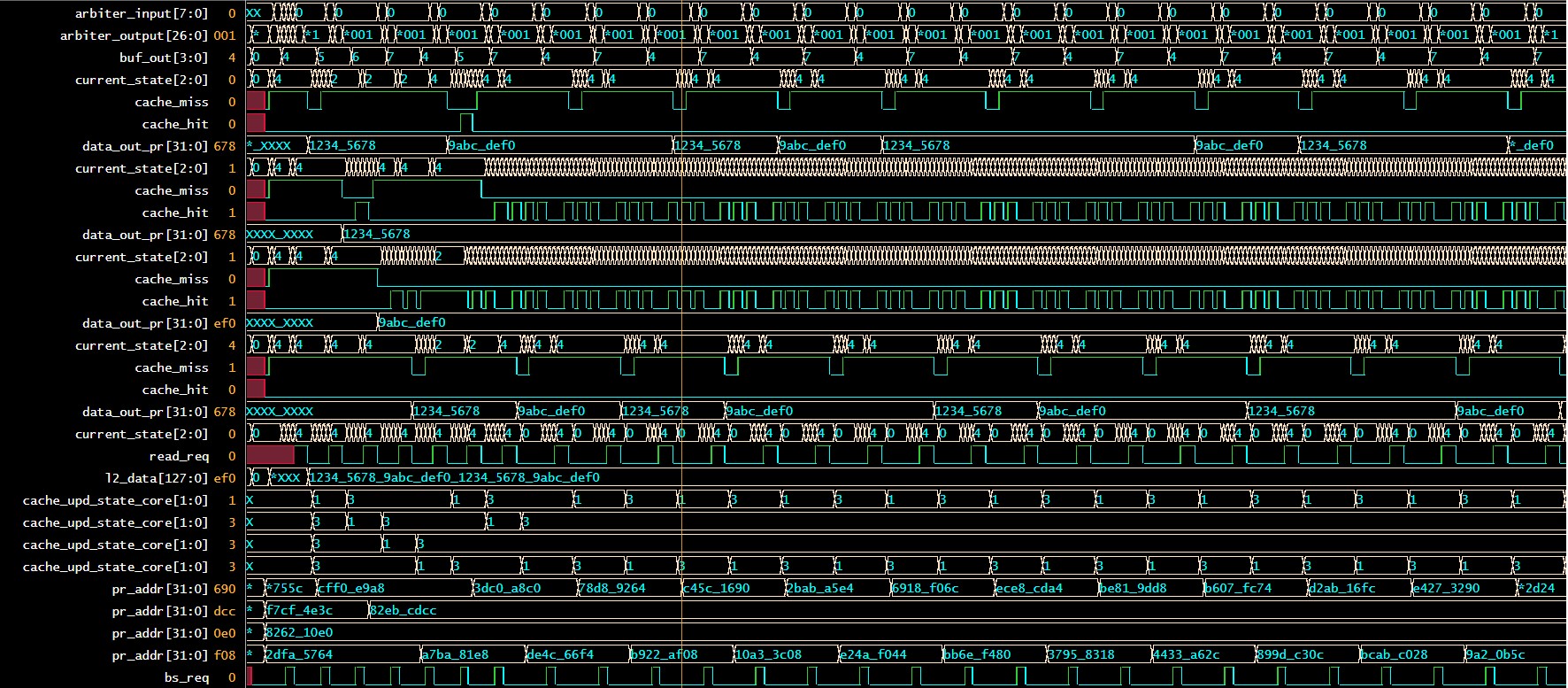
* **If random read cases are given across the cores with random address**

So, we made a UVM scoreboard check in which we initialized main memory of some data and on basis of offset we will compare data if all cases pass and according to offset the data is send to processor matches it will insure our read cases are working perfectly**.**

**Compare Checks:**

****

**Waves:**

****

**WRITE CASES**

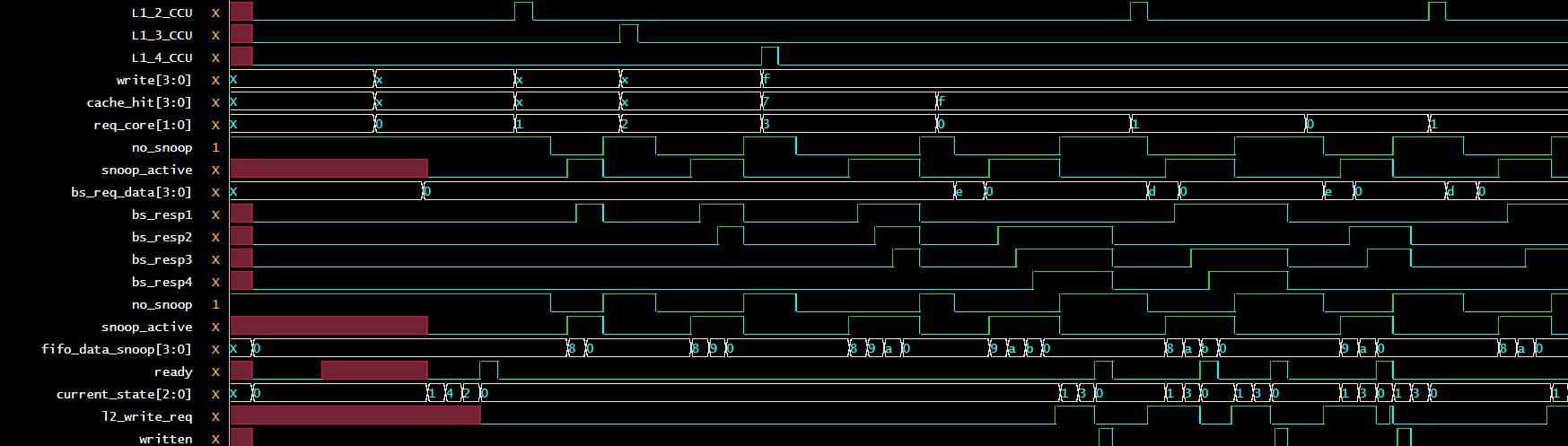
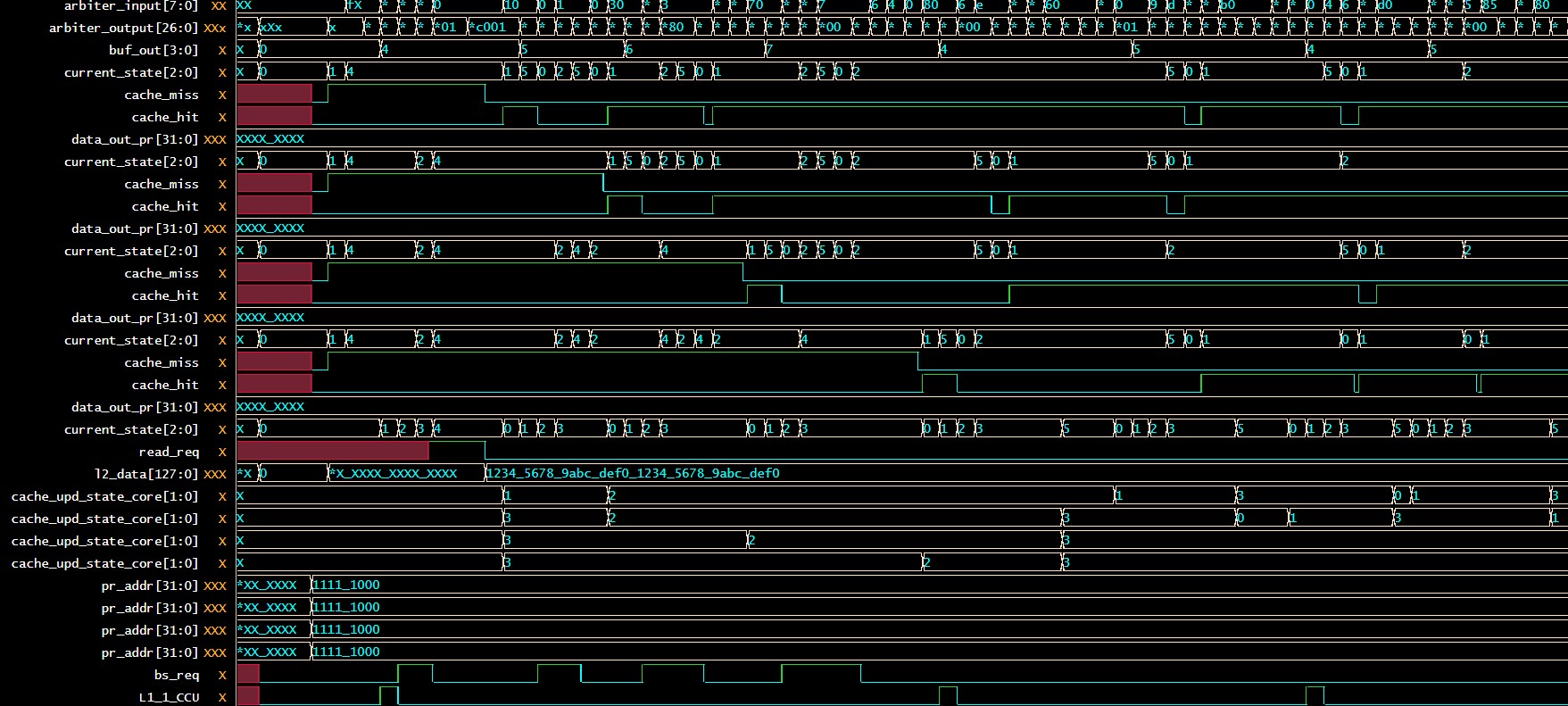
* **If all the Cores have same address and write is asserted**

Write signal is asserted for all four cores with same pr\_addr, requests are generated with core\_0 having most priority on arbitration gets its request granted. There is a compulsory miss since tag doesn’t match, no data in other cores available so finally data is being fetched from L2 and states transition from invalid to exclusive. This happens for other core simultaneously. Second time around for core\_1, there is a miss again so data is taken from all other cores and state change from Exclusive to **SHARED** for core 0 and 1. Then for other cores it happens too. Then it finally gets Hit for core\_0 and state changes to **MODIFIED** while other cores are invalidated. The modified data is written to L2 and state changes from **MODIFIED** TO **EXCLUSIVE**. Then for the next core the hit happens and other cores are invalidated and it continues for other cores.

**Table**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Cores** | **Pr\_Addr[31:0]** | **Hit/miss** | **Request** | **State** | **Hit/miss** | **State** | **Hit/miss** | **State** | **Hit/miss** | **State** | **Hit/miss** | **State** |
| Core\_0 | 32’h11111000 | Miss | 1’b1 | E, S | hit | M, E | miss | E, I | miss | I | miss | I |
| Core\_1 | 32’h11111000 | Miss | 1’b1 | I, S | miss | I | hit | M, E | miss | E, I | miss | I |
| Core\_2 | 32’h11111000 | Miss | 1’b1 | I, S | miss | I | miss | I | hit | M, E | miss | E, I |
| Core\_3 | 32’h11111000 | Miss | 1’b1 | I, S | miss | I | miss | I | miss | I | hit | M, E |

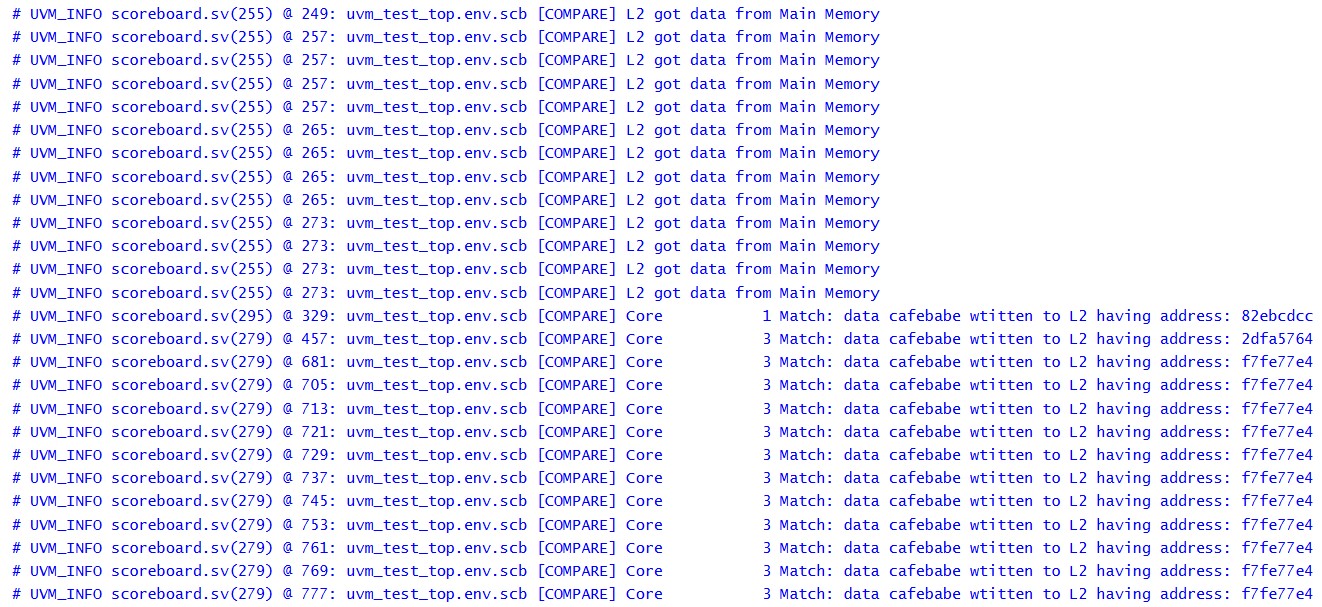
**WAVES:**

****

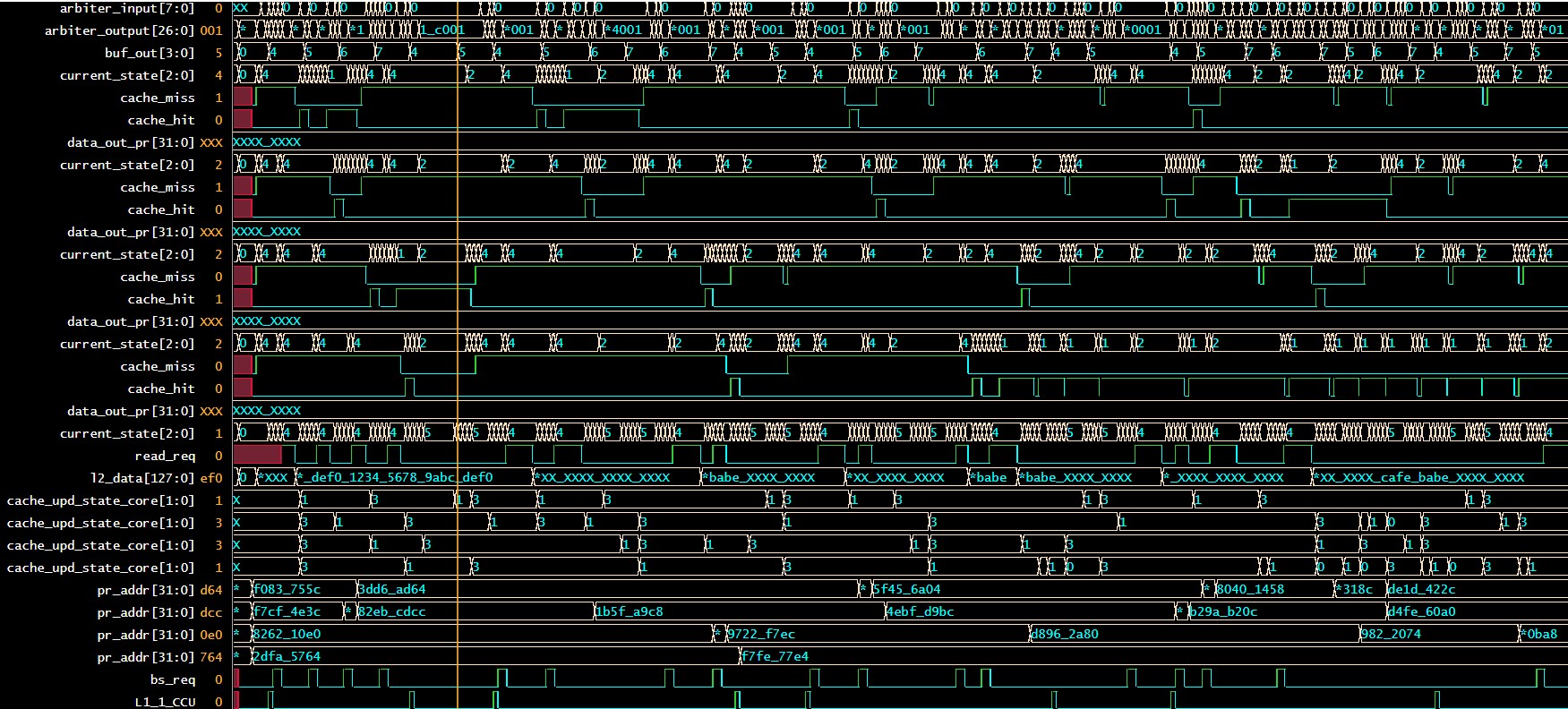
* **If random write cases are given across the cores with random address**

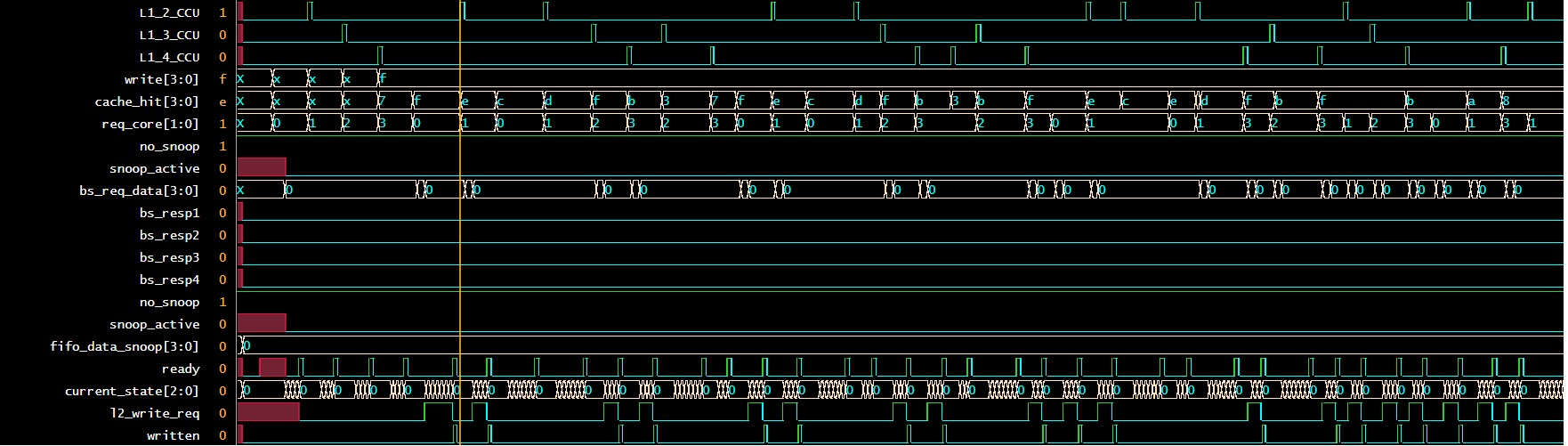
So, we made a UVM scoreboard check in which we initialized pr\_data which is **32’hCAFEBABE** and on basis of offset we will compare data if all cases pass and according to offset the data is written to L2 or not.

**Compare Checks:**

****

**WAVES:**

****

****

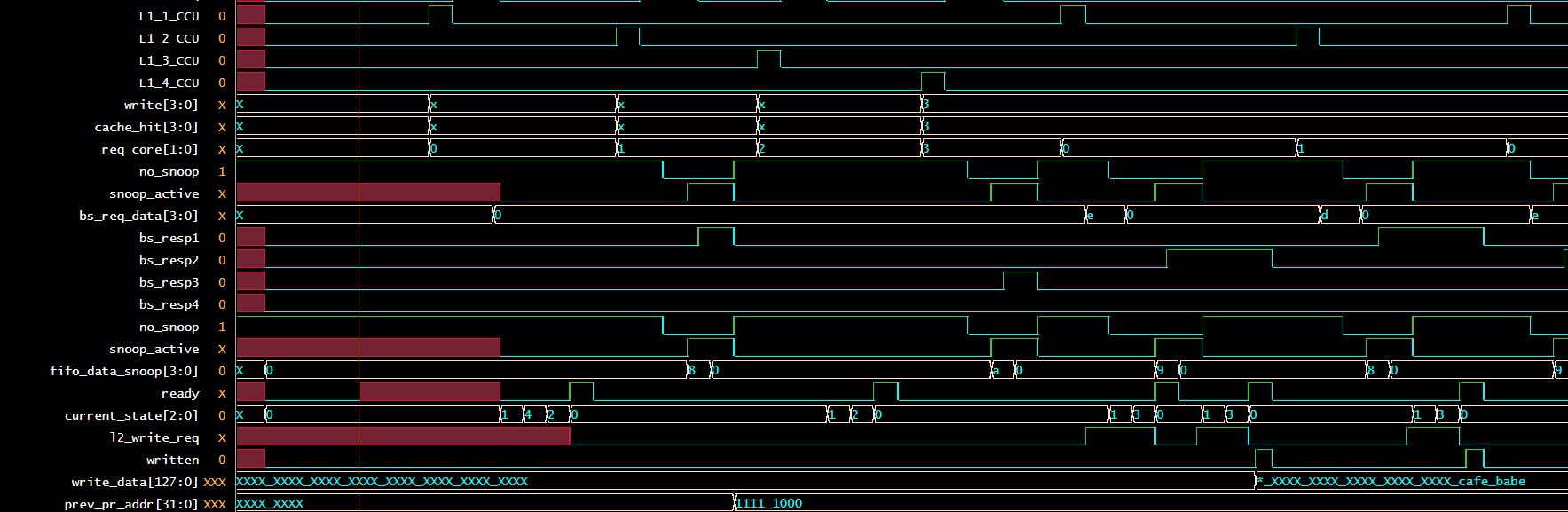
* **If Core 0 and 1 are given write with same address and Core 2 and 3 are given read with same address**

Core 0 and Core 1 get the same address while Core 2 and Core 3 get the same address. When write is asserted, Core 0 gets compulsory miss so request signal is asserted which gets to fifo after arbitration, the arbiter decides on priority bases and queue in all requests in fifo. While Core 0 being processed first, its request after being granted is handled by CCU, and through FSM (finite state machine) it goes through snooping and since no data has yet been loaded it moves to read from l2 state and fetch the data which is then passed on to data\_out\_pr. The other cores working simultaneously gets their grant and requested data is then snooped i.e. core 1 gets the data through snooping from core 0. However, note that Core 2 will have to repeat the same cycle as Core 0 since no same address is found and then Core 3 gets its data through snooping from Core 2. Then Core 0 will have write hit it will generate Request and upon getting grant, CCU will do snooping and invalidate the Core having data on this address in our case it will be Core 1, its data will be invalidated and new data will be written to L2 as it is Write through till L2.

**Table**

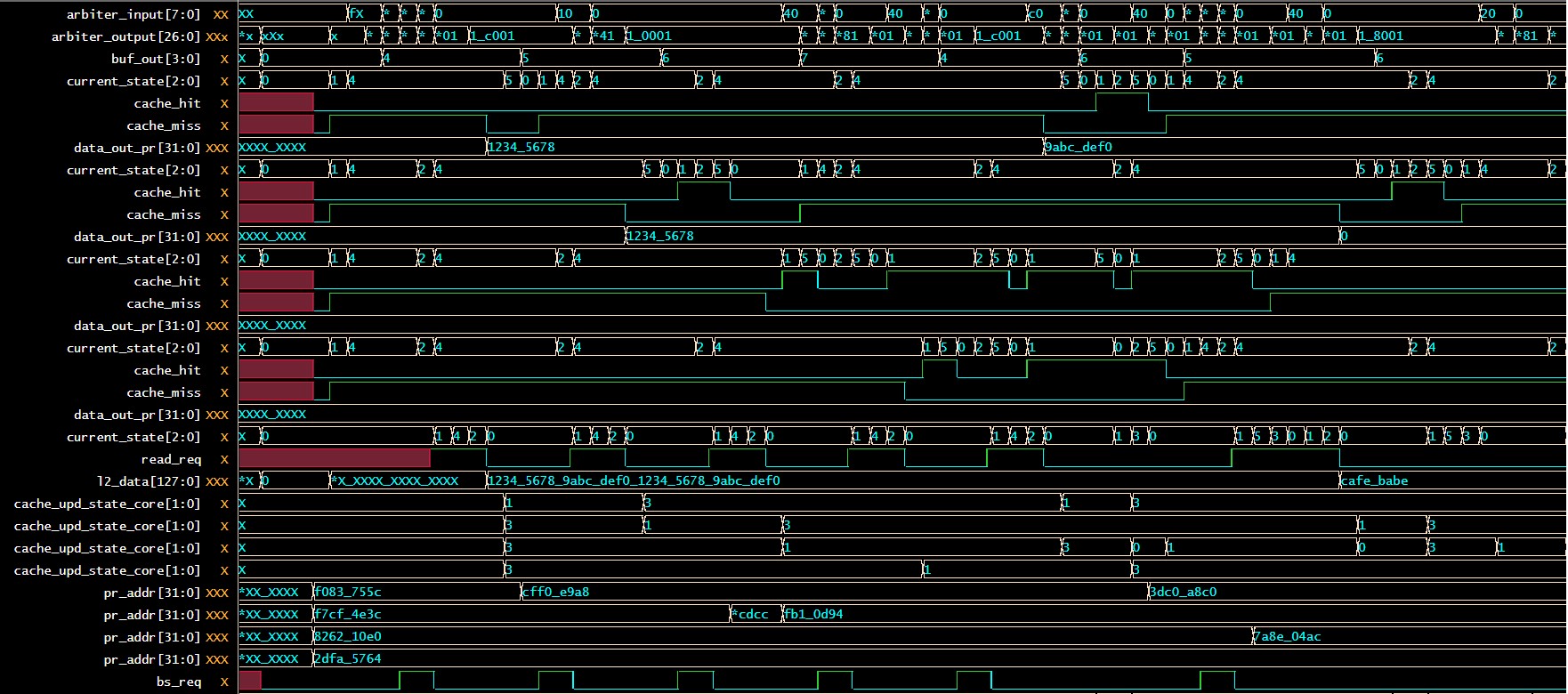
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Cores** | **Pr\_Addr[31:0]** | **Write/Read** | **Hit/miss** | **State** | **Hit/miss** | **State** | **Hit/miss** | **State** |
| Core\_0 | 32’h11111000 | Write | Miss | E, S, I | hit | S, M, E | Miss | E, I |
| Core\_1 | 32’h11111000 | Write | Miss | I, S, I | miss | S, I | Hit | I, M, E |
| Core\_2 | 32’h11110000 | Read | Miss | I, E, S | hit | S | Hit | S |
| Core\_3 | 32’h11110000 | Read | Miss | I, I, S | hit | S | Hit | S |

**WAVES:**

****

* **If Multiple Cores get random read and write requests with random addresses.**

**Waves:**

****