Multicore DLX with coherence protocol – Mid Presentation

Project Name: Multicore DLX processor with MESI coherence protocol

Project Number: 3024

Project carried out at: University

Submitted by:

Yarin Koren

Yohai Shiloh

Project instructor approval signature:

Name: _Oren Ganon_

Project Abstract

- Development of a **multicore processor** based on the DLX architecture with MESI(stands for Modified, Exclusive, Shared, Invalid) cache coherence protocol
- **Accelerate computing** using the multiple cores advantage to parallelized the execution of programs by the processor
- Using the **MESI protocol to ensure memory consistency** for all the cores while using a common memory to guarantee the process correctness

Project Abstract (Mid-project)

- Development of a **single core processor** based on the DLX architecture(during the *Advanced Computer Structure Lab*)
- Development of **cache unit** for the processor with full support of its architecture
- **Integrate** the cache unit, **customize** the single core processor and **validate its operation**

Updated Project Requirements

Deliverables and methods

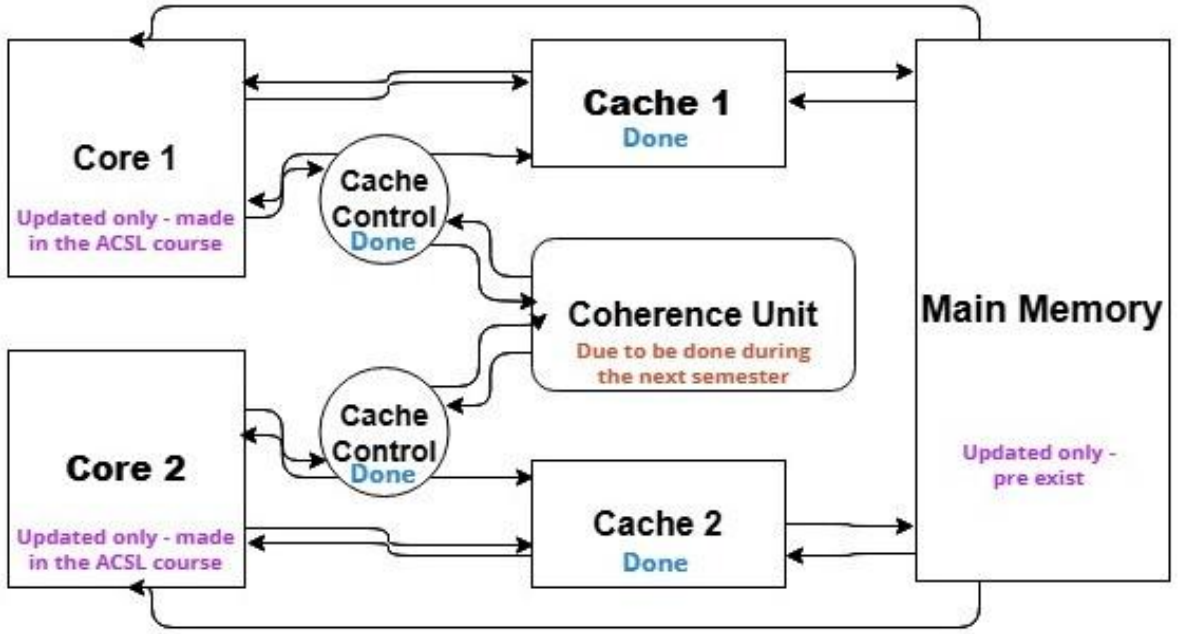
- Development of multicore DLX processor with MESI protocol for cache memory coherence
- Using Verilog and schematic over Xilinx ISE platform to develop and verify the design
- Demonstration of the final design using FPGA board

Updated Project Requirements

Project requirements

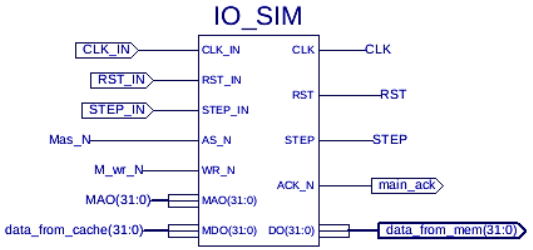
- Full support in the original DLX architecture
- Cache size of 16 memory blocks of 16 words each
- Speedup of 40% in the performance run time (relative to the single core processor)
- Running in at least 60MHz clock frequency

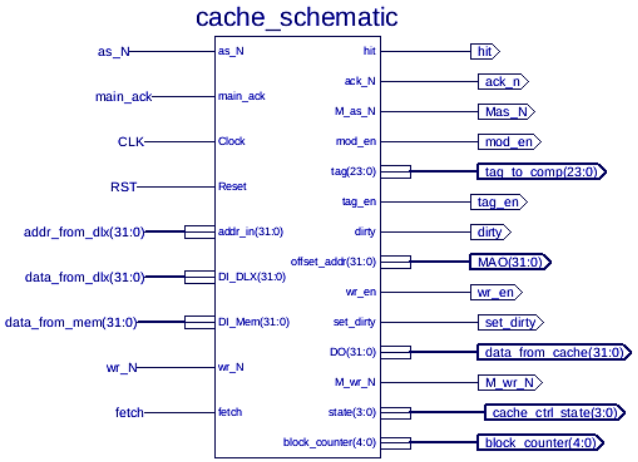
Updated and elaborated block diagram

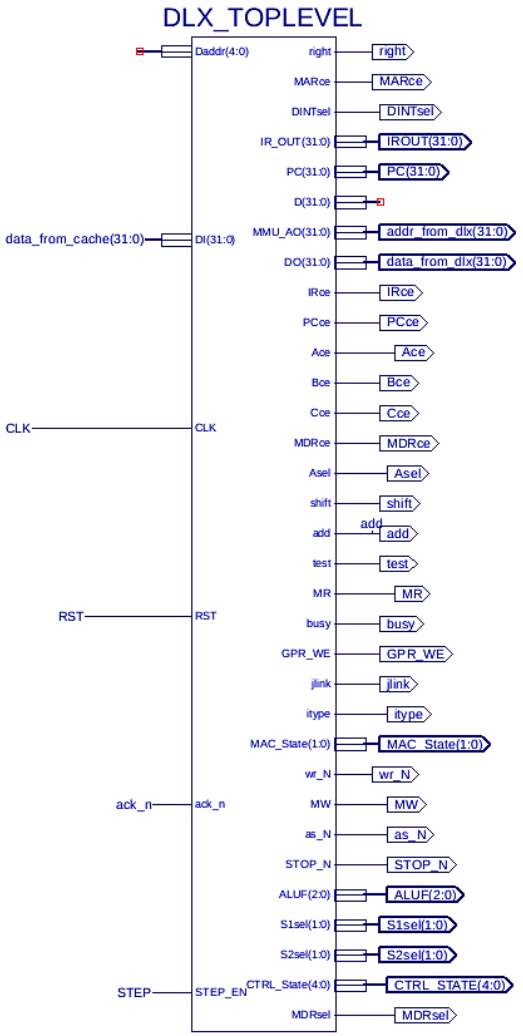


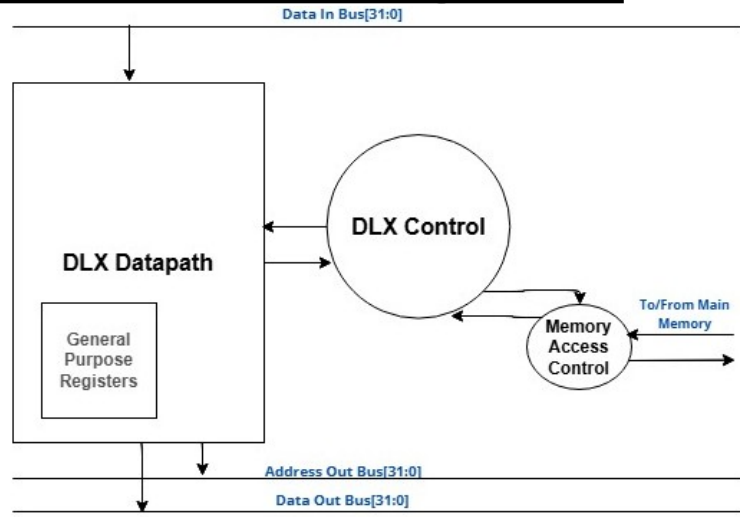
- All the new blocks – cache and cache control was implemented in Verilog language on Xilinx platform
- The core unit already designed and implemented during the *'Advanced Computer Structure Lab course'* and was only connected to the new blocks (which designed to support the original core without modifications)
- The cores, caches and cache control units are identical for both processors and therefore considered as done
- The Main Memory was existed and only its content was updated, using the DLX architecture assembly code

Updated and elaborated block diagram









On the top – the DLX core block diagram(preexist)

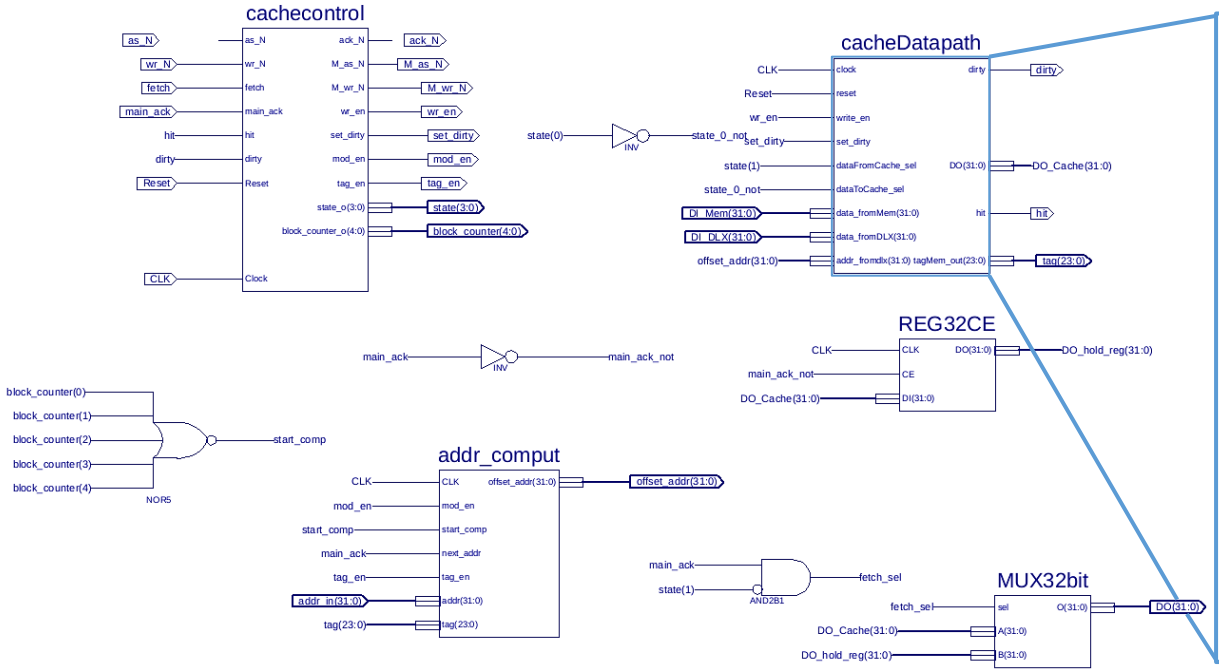
On the left – the updated single core processor(with cache) blocks diagram and connectivity

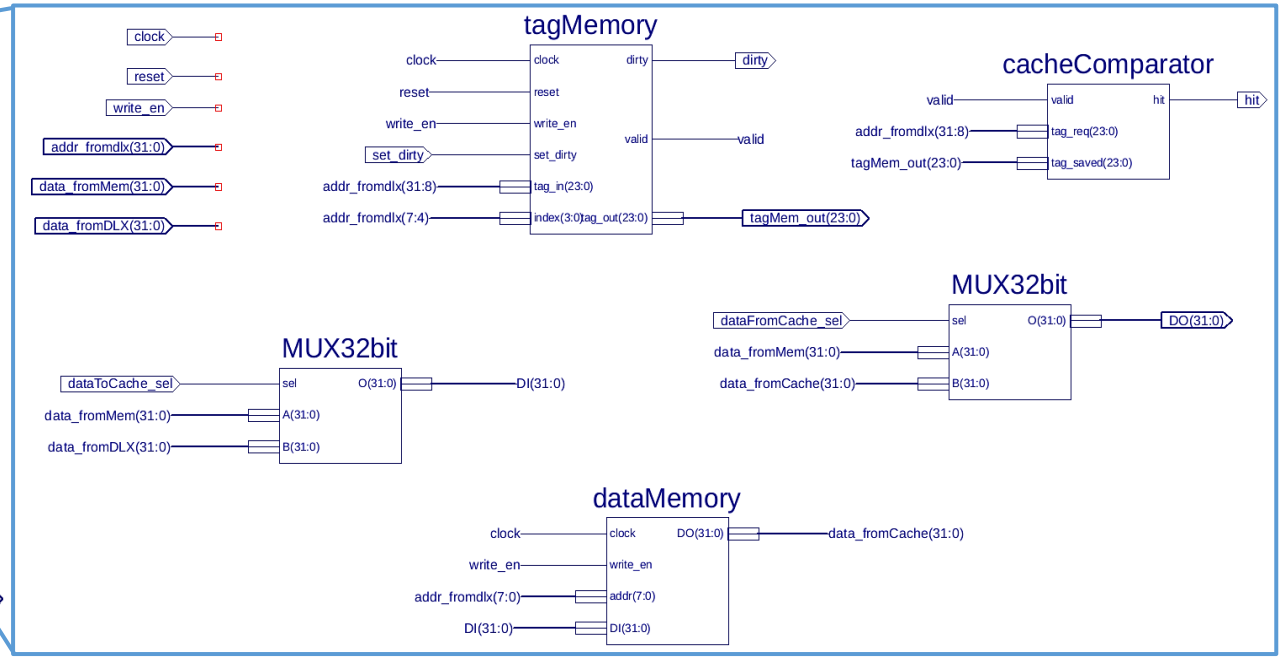
The blocks:

- DLX_TOPLEVEL – single core DLX(from the Advanced Computer Structure Lab)
- Cache_schematic – core's cache
- Isfetchcheck – checking if DLX reads instruction
- IO_SIM – Simulates main memory. Was given in the Advanced Computer Structure Lab

Multicore DLX with coherence protocol – Mid Presentation

Inside the cache





Cache inner structure scheme(left picture)
contains the blocks:

- ‘cachecontrol’ – managing cache operation
- ‘addr_comput’ – calculate address wished to read from/write to
- ‘cacheDatapath’ – contains block’s data and tags. Its full structure detailed on the right side of the slide

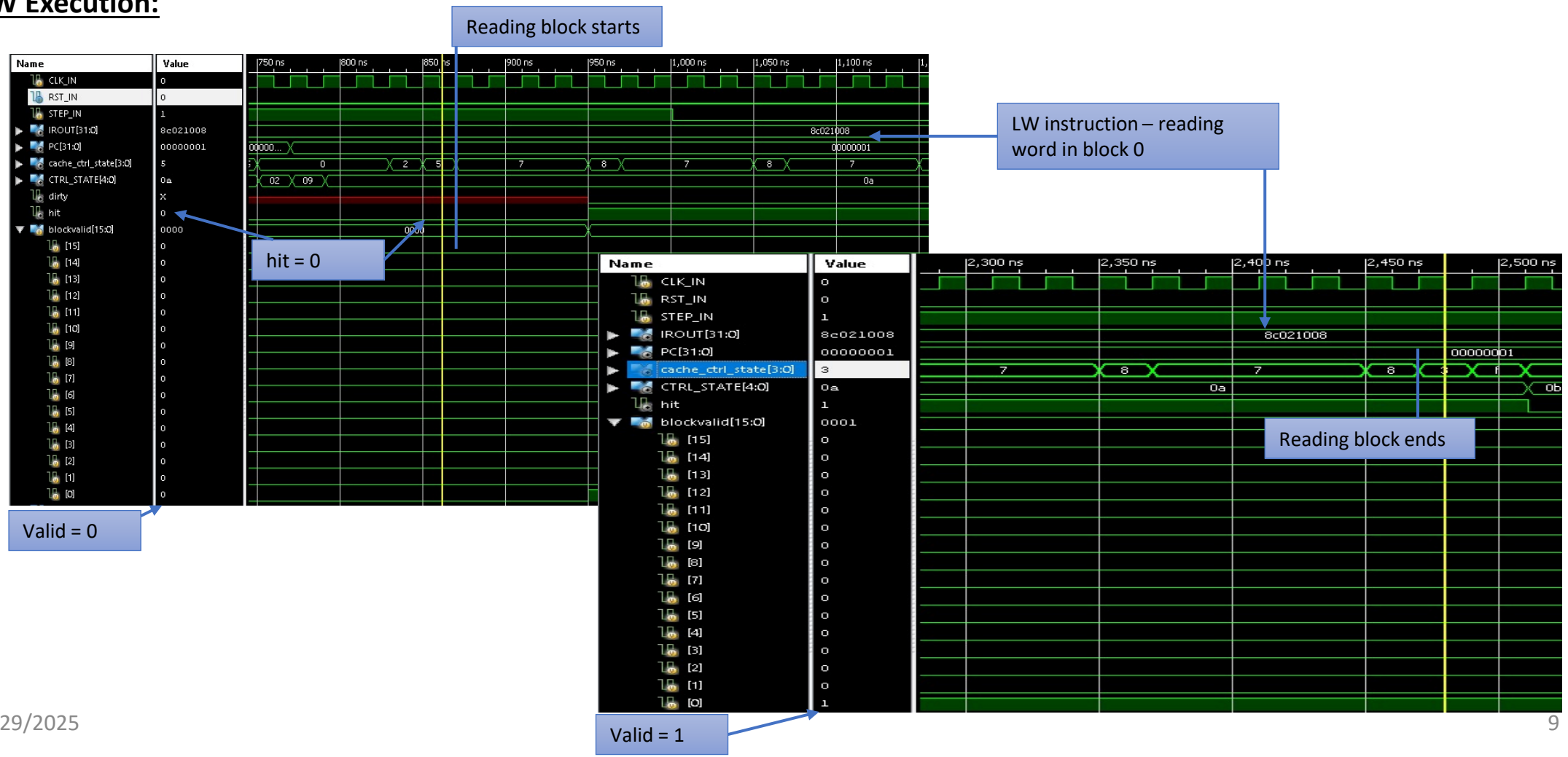
Cache datapath block(right scheme) components:

- ‘tagMemory’ – contains blocks' tags in caches
- ‘dataMemory’ – contains cache data
- ‘cacheComparator’ – checks if we hit or missed the required block

All the blocks was implemented using Verilog HDL over Xilinx’s ISE software, except the ‘cacheDatapath’ block that implemented using the ISE’s schematic tools, the MUX32bit and the REG32CE blocks that was provided to us by the staff of the ACSL course and the basic logic gate, which is included in the ISE built-in library

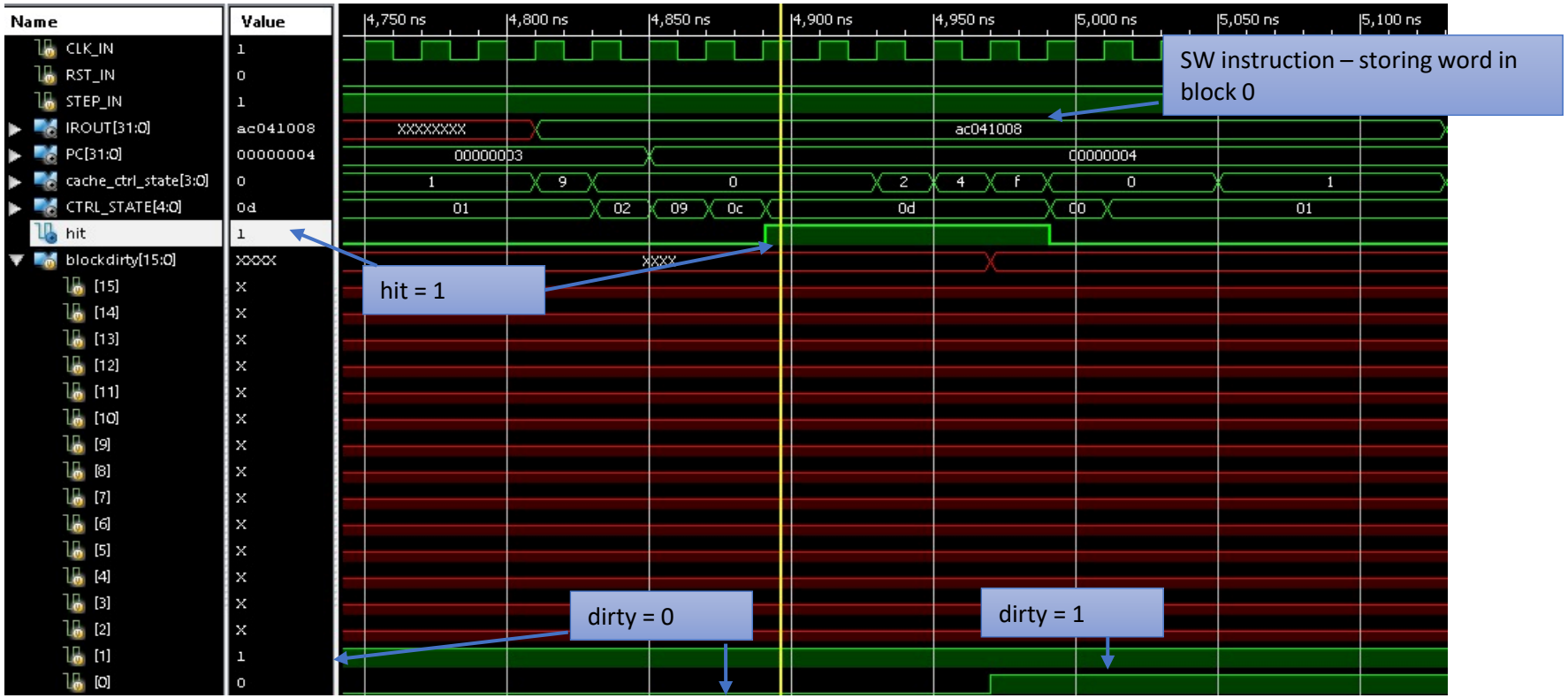
Project products

LW Execution:



Project products - continue

SW Execution:



Project products – continue

Unachieved milestones

- Running the program with the updated design on the FPGA board

quantized requirements

- Full instruction support
- Cache size of 16 blocks of 16 words (new)

Updated schedule(past semester)

Milestone	Planned delivery date	Actual delivery date	Remarks
Researching DLX	8.12.2024	15.12.2024	Revision of the original architecture and evaluate possible changes
Implementing DLX	22.12.2024		After considerations we designed the cache with no need to make changes in the original design
Simulate and verify DLX	1.1.2025		Validation only after changes
Re-architecture the basic DLX, and adding cache memory	12.1.2025	15.1.2025	
Writing benchmark in DLX assembly	18.1.2025	25.1.2025	
Simulating and verify DLX + cache	27.1.2025	6.2.2025	Without running the design on the FPGA board
Progress Presentation Submission	2.2.2025	13.2.2025	

Updated schedule(next semester)

Milestone	Planned delivery date	Actual delivery date	Remarks
Verify DLX with cache	27.1.2025	18.3.2025	Verifying with the FPGA board – debt from past semester
Designing multicore and researching MESI protocol	2.3.2025	23.3.2025	Postponed due to the delay of the singled-core processor verifying
Implementation of the design and the protocol	9.3.2025	30.3.2025	
Performance evaluation	23.3.2025	20.4.2025	
Simulation and verification	20.4.2025	27.4.2025	
Performance analysis	15.5.2025		
Poster Submission and finishing the work	25.5.2025		
Documentation	1.7.2025		
Final deliverables submission	2.7.2025		