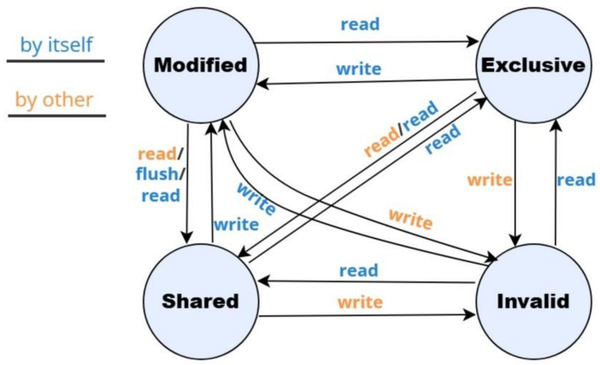
Multicore DLX processor with coherence protocol

Project Number: 24-1-1-3024
Names: Yohai Shiloh and Yarin Koren
Advisor: Oren Ganon

The Iby and Aladar Fleischman Faculty of Engineering
Tel Aviv University

Introduction

This project presents an extension to the basic DLX architecture, by using some of the most common methods to improve CPU performance – adding a cache memory for faster memory access and expanding the single-core CPU to a multicore CPU for parallel computing, while maintaining memory consistency by using a coherence protocol.
The MESI coherence protocol ensures data consistency across caches in a multicore system and prevents the possible usage of invalid data by other cores. It assigns to each data block in the cache a state – Modified, Exclusive, Shared or Invalid.
The possible transitions between the states:



To validate the design and demonstrate its correctness, the project was also loaded into an FPGA chip, such as the one shown below, for physical implementation of the design

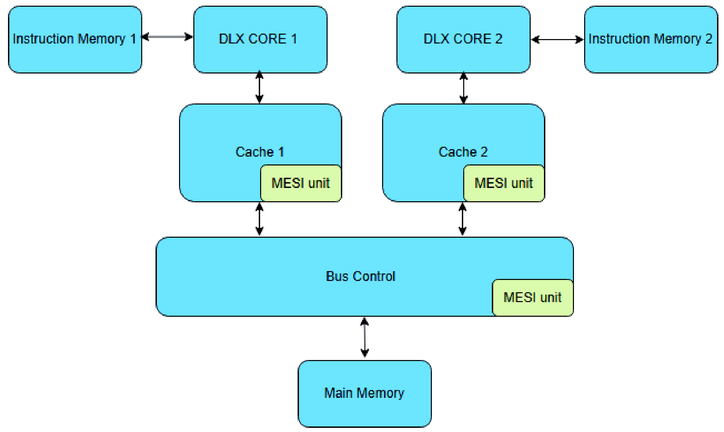
Motivation

As the demand for faster and more power-efficient processors continues to grow, the limitations of hardware and materials used in single-core designs have become more apparent. To address these challenges, the industry increasingly turns to creative solutions — with parallelism being one of the most effective.

Implementation

The project took the basic DLX processor, which was designed in the ASCL lab, and developed its architecture into a multicore processor with cache memory units, including the following components and improvements:

1. Dual DLX independent cores for parallel execution.
2. An instructions-memory unit for each core to reduce bottleneck overload over the bus.
3. A cache memory unit to each core, for faster data access
4. Implementation of the MESI protocol.
5. A bus control unit for correct data routing and memory access scheduling



Functional Demonstration

A set of test programs was designed and executed to verify the correct operation of the multicore system and the MESI cache coherence protocol. Results show consistent data sharing and proper state transitions across caches. For example, the results of a 'bubble sorting' program on a 32-word array are shown below.

Before:

Memory Dump

ADDRESS	VALUES							
0x00000000	0x000000a2	0x000000c4	0x000000e9	0x000000b3	0x00000071	0x00000087	0x000000f7	0x00000034
0x00000008	0x00000035	0x00000079	0x00000096	0x000000ea	0x000000d1	0x00000075	0x0000000b	0x000000e0
0x00000010	0x000000dd	0x0000005b	0x0000004f	0x000000ec	0x000000de	0x00000081	0x000000c9	0x000000af
0x00000018	0x000000d9	0x000000c3	0x00000062	0x0000009a	0x000000a5	0x0000001a	0x000000b8	0x00000044

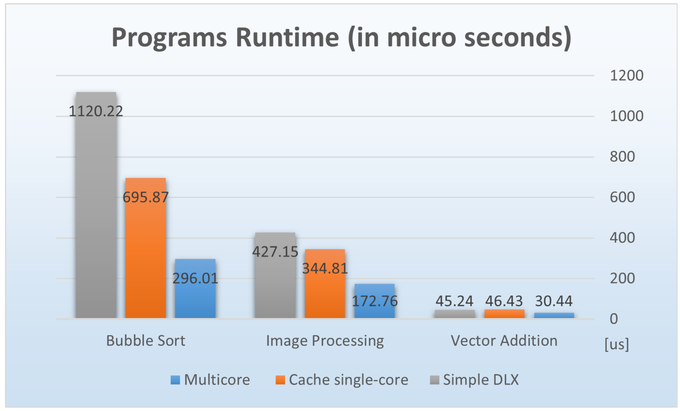
After:

Memory Dump

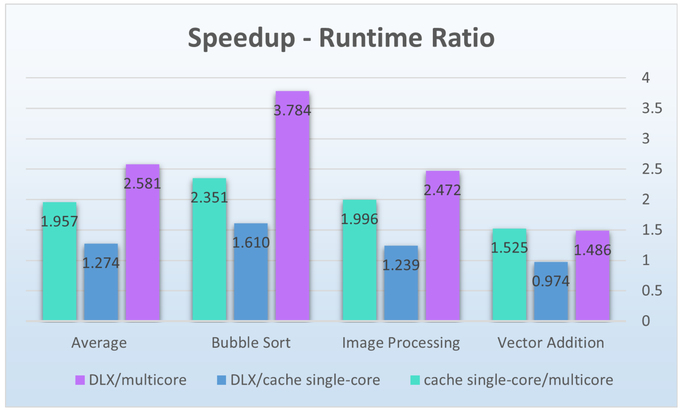
ADDRESS	VALUES							
0x00000030	0x0000000b	0x0000001a	0x00000034	0x00000035	0x00000044	0x0000004f	0x0000005b	0x00000062
0x00000038	0x00000071	0x00000075	0x00000079	0x00000081	0x00000087	0x00000096	0x0000009a	0x000000a2
0x00000040	0x000000a5	0x000000af	0x000000b3	0x000000b8	0x000000c3	0x000000c4	0x000000c9	0x000000d1
0x00000048	0x000000d9	0x000000dd	0x000000de	0x000000e0	0x000000e9	0x000000ea	0x000000ec	0x000000f7

Results

When the multicore processor was tested with representative assembly programs (the results of three of them are shown below), all demonstrated substantial runtime improvements compared to single-core execution. These results confirm the efficiency of parallel execution and the effectiveness of our MESI-based cache coherence mechanism, alongside other improvements that were implemented in the project.



Conclusions



To conclude, our multicore architecture achieved the highest performance across all tests. The benefits were particularly evident when processing large data sets and performing intensive computations on closely located memory addresses, reaching a maximum speedup of ×3.78 and an average speedup of ×2.58.