Implementation of a Cache Controller using a HDL

1. Project Overview

This project requires students to **design, implement, and test** a **cache controller** for a simple computer system using a hardware description language (HDL). The project will be handled by the same teams that implemented the ALU. The cache controller will be built around a finite state machine (FSM) to manage the operations and state transitions effectively. This exercise aims to deepen students' understanding of cache memory management, FSM-based design, and HDL programming, which are critical in computer engineering and architecture.

2. Learning Objectives

Students will achieve the following by completing this project:

- Develop a comprehensive understanding of cache memory mechanisms and their impact on system performance.
- Gain practical experience in designing a hardware module based on FSM principles.
- Implement and simulate a set-associative cache controller using an HDL.
- Evaluate the performance and efficiency of the cache system through rigorous testing.

3. Project Description

3.1 Cache Controller Specifications

The cache controller to be designed must adhere to the following specifications:

- Cache Type: 4-way set associative
- Cache Size: 32 KB
- Block Size: 64 bytes
- Word Size: 4 bytes
- Number of Sets: 128 sets
- Associativity Level: 4-way
- Replacement Policy: Least Recently Used (LRU)
- Write Policy: Write back with write allocate

3.2 FSM-Based Design

The cache controller should be implemented as a finite state machine with clearly defined states including, but not limited to:

- IDLE: Waiting for a new request.
- READ HIT: Accessing data for a read request found in the cache.
- READ MISS: Handling read requests when data is not in the cache.
- WRITE HIT: Managing write operations when data is found in the cache.
- WRITE MISS: Processing write operations when data is not found in the cache.
- EVICT: Evicting data from the cache according to the LRU policy.

Each state must handle the transitions based on the cache operation required and update the cache accordingly.

3.3 Hardware Description Language

Students may select one of the following HDLs:

- VHDL
- Verilog
- SystemVerilog

3.4 Simulation and Testing Tools

Appropriate tools for simulation and validation must be used, such as:

- ModelSim for wave simulation

4. Deliverables

4.1 Detailed Design Specification

This document must include:

- Detailed FSM diagram showing all states and transitions
- Functional block diagram of the cache controller
- Description and justification of the chosen HDL

4.2 Implementation

The complete HDL code for the cache controller, including:

- Modular HDL code for each component of the cache controller
- In-line comments explaining critical sections of the code

4.3 Test Bench and Simulation Results

A detailed test plan and the corresponding simulation results must be presented, illustrating:

- Various test cases covering all possible states and transitions
- Waveforms and state transitions observed during simulations
- Performance metrics such as hit rate and access time

4.4 Final Report 500-1500 words

The project culminates in a comprehensive report that covers:

- Overview of the design and implementation process
- Technical challenges encountered and solutions implemented
- Analysis of the performance data collected during simulations

5. Evaluation Criteria

Projects will be evaluated based on:

- Technical complexity and correctness of the FSM design
- Functionality and robustness of the HDL implementation
- Completeness and depth of the testing process
- Quality of documentation

6. Timeline

The project spans 4 weeks, and is to be handed în by uploading it to the virtual campus assignment named Proiect Cache Controller, the due date being 31.05.2024, 24:00.

Any questions will be discussed during the laboratory.