UNIVERSITY OF TEHRAN

Electrical and Computer Engineering Department

Core-Based Embedded System Design
ECE 160 - Spring 1403-1404
Computer Assignment 3 – Memory Interfaces in an Embedded System
Due Date: Tir 15

Description:

A minimal embedded system comprises a processor, an instruction memory, a data memory, and an I/O interface. In this assignment, you will be familiar with the following memory interfaces:

- SPI interface for Flash memory
- Integrated SRAM memory interface

You will build the system step-by-step across the following parts, simulate each part in ModelSim or SystemC (upon your choice), and analyze the results.

Part A: Instruction Memory

In this step, you will replace the simple instruction memory with an SPI Flash memory. A virtual Flash is provided for the simulations that models the Flash memory. The memory contents are stored in a text file (*'mem.mem'*). Suppose that the Flash memory is filled before. As the Flash memory has an SPI interface, you need to add an SPI controller to access the Flash memory contents. The SPI controller HDL code is also provided. This controller can only read data from the memory.

- **Memory Mapping:** Specify address ranges for instruction and data memories like the previous Computer Assignment (CA2).
- **Building System Hardware:** Integrate the processor core with the instruction and data memory modules. Instruction memory is the virtual Flash and it is connected to the system via an SPI controller. Use a simple memory for data memory like CA2.
- **Building System Software:** Write a C/C++ program to compute the factorial of n=10. Compile the code using the provided *Makefile* in CA2.
- **Evaluation:** Simulate the system in ModelSim/SystemC and verify correct memory access and factorial computation.

Part B: Data Memory

Replace the simple data memory with an IP-core of SRAM whose model is provided for simulation. M31HDSP200GB180W_4096X8X1CM16 is a 4KB SRAM memory that its transactions are applied at the positive edge of the clock. This IP core has active low chip select (CEN) and write enable (WEN) pins. The IP core has no ready pin, so you must provide this signal for the processor.

- **Building System Hardware:** Replace the previous data memory with the integrated SRAM.
- **Building System Software:** Software remains unchanged.
- **Evaluation:** Simulate and verify the system.

Part C: Peripheral Interface

Design and add an SPI peripheral interface to the system (IO). It is connected to an off-chip slave SPI input/output.

- **Memory Mapping:** Specify address ranges for memory-mapped registers (input and output registers).
- **Building System Hardware:** Design a complete SPI controller and connect it to the system. Off-chip SPI module generates an interrupt when it has a new input for the system. Connect this interrupt signal to the processor. By receiving an interrupt, the processor initiates a read transaction and as a response the SPI controller receives the new input serially. After computation is completed, the SPI controller will transmit the result serially to the slave initiated by a write transaction from the processor.
- **Building System Software:** Modify the factorial code to read input (n) from SPI receiver port and write the result to the SPI transmitter port.
- **Evaluation:** Simulate and verify system functionality.

Part D: Program Interface (Bonus)

Add a one-bit mode input to the system. This input determines system mode which may be functional (0) or programming (1). If the system is set in the programming mode, instructions will be written to the Flash memory through SPI peripheral.

- **Building System Hardware:** You must add the following:
  - Add the mode input to the system. You can use a multiplexer to implement switching between functional and programming modes.
  - Add write capability to the virtual Flash memory (using Flash memory datasheet).
  - Add write capability to the SPI controller of Flash memory (using its datasheet).
- **Building System Software:** Use the previous software.
- **Evaluation:** Evaluate the entire system in both functional and programming modes.

Deliverables

- System Schematics:
  - Diagrams for each part (e.g., Visio or hand-drawn).
- Hardware Design Codes:
  - All HDL/SystemC codes.
- Software Design Codes:
  - All C/C++ codes.
- Simulation Results:
  - Screenshots verifying correct functionality.
- Put all codes (Verilog/VHDL/SystemC, C/C++) and the report file (including screenshots) in a folder and zip it. Name your zipped file using the following format:

  'YourName_YourFamily_Last5digitOfYourStudentID_CA#3'