RISC-V FPGA Implementation

Milestone 2

Abdallah Gabara

Marc Boutros

Mohamed Abdelfattah

Muhammad Azzazy

Requirements

Create a RISCV32I FPGA Implementation that supports all forty-seven user-level instructions as described in the specifications except ECALL, EBREAK, FENCE, FENCE.I, and CSR instructions. The implementation should be single cycle with two seperate memories for instructions and data. Data memory should be byte addressable. All supported instructions should be tested and proven working properly.

Modules

- ControlUnit.v:

Outputs the control signals controlling different modules in the data path to fit the instruction being executed.

- ALUControl.v:

Responsible of outputting the selection line for the ALU based on the instruction in the execution phase and the ALUOp signals from the Control Unit.

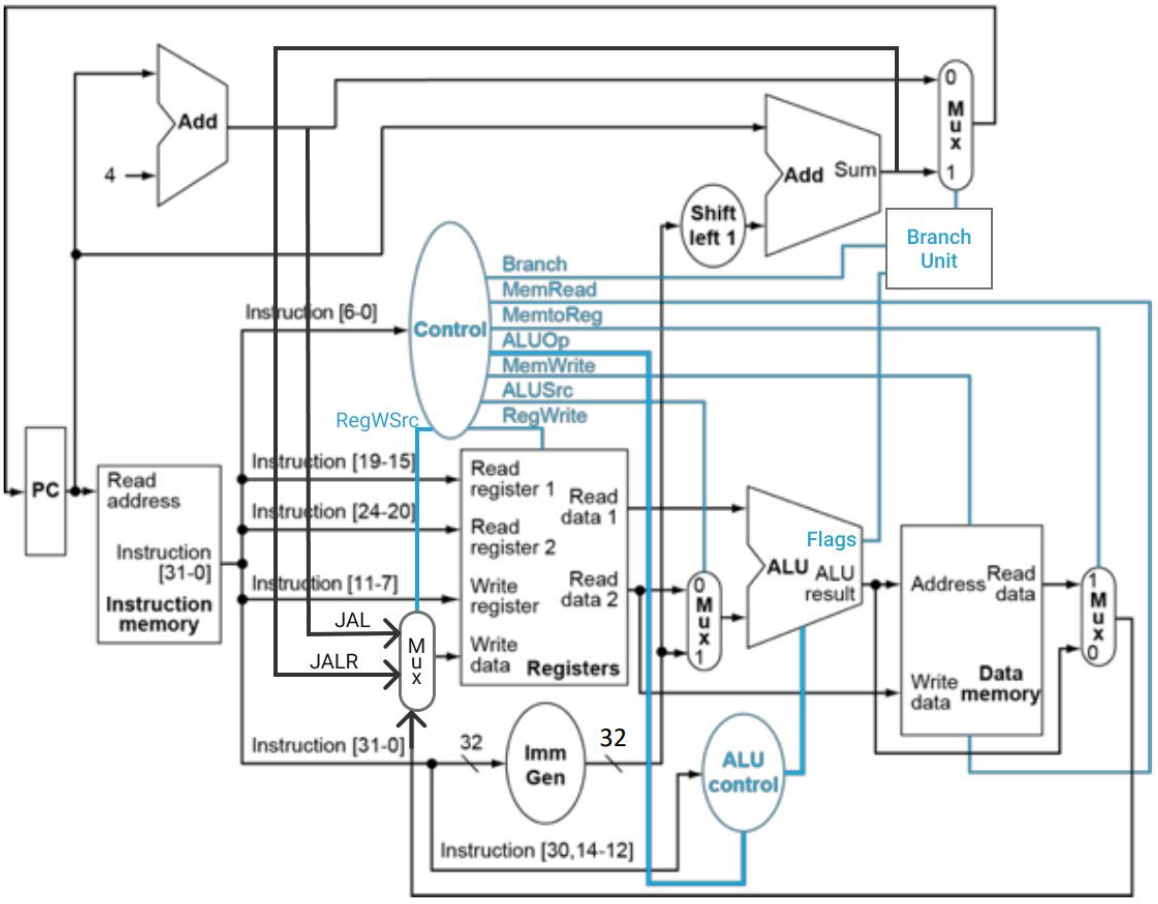
- DataMem.v:

A byte addressable memory divided in bytes. In the reading mode, the memory outputs a word based on 3 signals it receives from the Control Unit (by, half, unsign). In the case of, byte or half reading the memory outputs the required byte or half with sign extension or not depending on the unsign signal. Otherwise, the memory outputs a word conventionally.

- branch_unit.v:

A specialized unit that receives the flags of the ALU and determines whether to go for branching or not provided that the instruction in execution is a jump or branch.

Block diagram



RISCV32I Single cycle implementation

Work procedure

The pipelined datapath with the limited instruction set used in the lab had its pipeline registers removed and rewired as a single cycle datapath. First, multiple components and multiplexers were added to the top module to accomodate for the new connections required to support the full instruction set. An example of the additional implemented paths is multiplexing the rs1 field from the instruction word with zeros to implement the LUI instruction where the shifted immediate is added to register x0 and stored in rd. Similar multiplexing strategies were used to implement the AUIPC instruction were the value written to register rd is calculated using the branching adder; similarly the value written to rd by the instructions JAL and JALR comes from the adder that increments the program counter. Then, the control unit and alu control unit modules were modified and an additional unit, namely, the branching unit was added to support all kinds of branches concisely, that is, dedicating the top modules to rewiring the modules only. Later, the byte addressable little endian data memory was implemented.

Testing strategy

- Come up with different suitable assembly instructions to test each supported functionality.
- Assemble the instructions into binary
- Write the binaries into the Instruction Memory
- Run the simulation
- Inspect and validate the output in the simulation
- Multiple test cases were run on the fpga.