Computer Architecture
CSCE 3301-01
Fall 2019
MS3 Report
November 11th, 2019
For
Dr. Cherif Salama

Background:

Project: femtoRV32

Author(s): Abdelhakim Badawy - abdelhakimbadawy@aucegypt.edu

Marwan Eid - marwanadel99@aucegypt.edu

Mohammed Abuelwafa - mohammedabuelwafa@aucegypt.edu

Description: Verilog modules that constructs the pipelined Verilog module that simulates a processor with single ported memory instead of dual memory architecture with basic test cases. The module supports 47 instructions of RV32I processor.

Technical Description:

- We used the Verilog modules we implemented in the lab for the single cycle datapath supporting only seven instructions, modified and added some aspects to support all the 47 instructions.
- Then we added the pipeline registers and replaced the instruction memory and the data memory with a single-ported memory that has the instructions and the data. The last instruction is supposed to be an EBREAK instruction for it to stop the PC from being incremented and load the data afterwards.
- We modified the memory to be byte-addressable and of size 4 GB.
- We determine the type of the instruction in case of load or store instructions using the func3 bits of the instruction.
- We generated a slow clock signal that has a frequency equal to half of the normal clock using a 1-bit counter.
- The memory reads instruction (i.e. memory address equals the PC) at the positive edge of the slow clock memory and reads data at the negative edge of the slow clock (i.e. memory address is equal to the value computed at the ALU).
- The inputs of the ALU are selected by two 3x1 multiplexers: one before each input. The first ALU input is selected from (readdata1 ID/EX, MEM/WB MUX output and the EX/MEM ALU result). The second input is chosen from (the output of a 2x1 multiplexer whose inputs are the readdata2

and the immediate output, EX/MEM ALU result, and MEM/WB MUX result).

- The carry flag, zero flag, overflow flag, and sign flag generated by the ALU, along with the current instruction's opcode and func3, and the branch signal, are inputs to the branching control unit which in turn generates a signal (pc_sel) which is an input to the control unit; this signal is an input to a module. All these signals are passed to the pipeline registers according to the block diagram below.
- The instruction needed to be implemented as a NOP instruction are implemented so by setting all the control signals to 0.
- EBREAK instruction is determined by checking its opcode, func3, and the 20^{th} bit of the instruction and is implemented as a halting instruction by setting all control signals to 0 except for the branch signal which is set to 1 and is an input to a module that generates the selection line for the multiplexer whose output is the next program counter; additionally, this module has the branch signal and the instruction as inputs. The multiplexer has four inputs: the output of the adder of the program counter and the hard coded value 4, the output of the adder of the program counter and the immediate generator output, the value of the next program counter in case of a JALR instruction, and the same program counter. Accordingly, we can choose the next program counter based on the current instruction, the branch signal, and the pc_sel signal.
- The value of the next program counter in case of a JALR instruction is determined by ANDing the ALU result with -2, (32'b111…110) which thus sets the least significant bit to zero, since jumps are allowed to only even addresses.

Block Diagram:

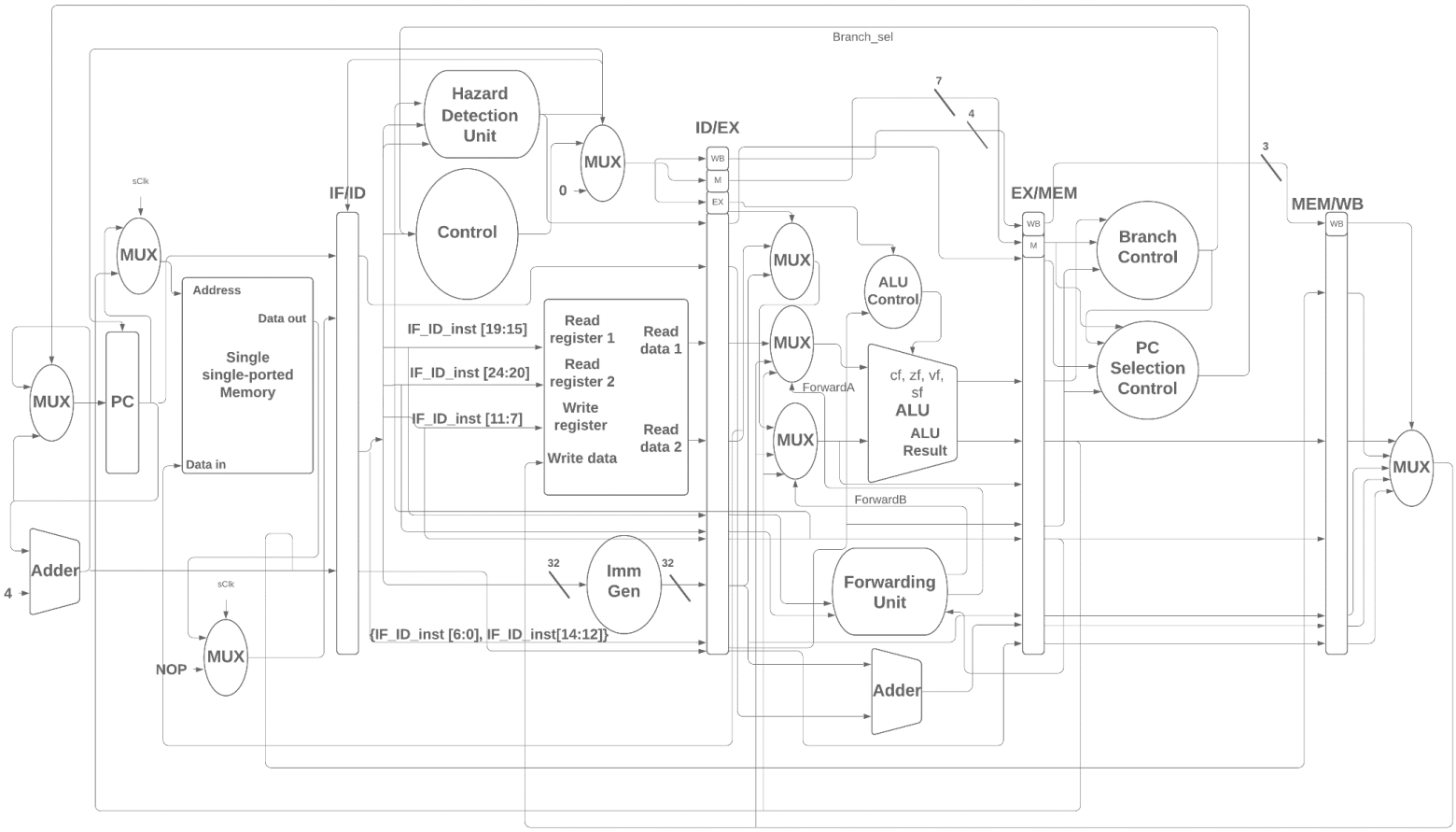

Figure 1

Schematics:

The schematic is attached in the submission folder.