EC605: Computer Engineering Fundamentals — Fall 2019
Lab 5: Single Cycle ARM Datapath

Goals

- Gain experience with digital design and Verilog.
- Gain understanding of computer organization via hands-on implementation of an ARM datapath.
- Gain knowledge on building complex, multi-module structures by mapping out parts in block diagrams.

Overview

In this lab you will design a single-cycle ARM-like datapath, which supports variation on a sub-set of the LEGv8 instructions. The structure of the datapath should follow the one described in the textbook and lectures. The movk instruction implementation is not specified in the lecture, and you may implement the datapath to support it as you see fit.

You must completely follow the definition in this handout for the design, especially the specific opcode, which is not necessarily the same as the actual ARM standard.

The simplified ARM instruction format is as follows:

R-format:

opcode	Rm	shamt	Rn	Rd
31 21	20 16	15 10	9 5	4 0

D-format:

opcode	DT address	Rn	Rt
31 21	20 10	9 5	4 0

B-format:

opcode	BR address
31 21	20 0

CB-format:

opcode	COND BR address	Rt
31 21	20 5	4 0

IM-format:

opcode	MOV immediate	Rd
31 21	20 5	4 0

Following is the simplified subset of instructions to be supported:

Instruction	**Opcode**
BRANCH	11'h0A0
AND	11'h450
ADD	11'h458
ORR	11'h550
CBNZ	11'h5A0
SUB	11'h658
MOVK	11'h794
STUR	11'h7C0
LDUR	11'h7C2

Task 1: Datapath Components

1. **Instruction Memory**
   The instruction memory, with sample instructions is provided in *IMem.v*. Note that the instruction memory is read-only. Simulate the instruction memory module and make sure that it operates correctly.

   Note: In the standard datapath, the PC counter is incremented by 4. For the laboratory, increment the PC by 1 instead of 4.

2. **Register File**
   The register file module, *Register_File.v* is provided. Simulate the register file module and make sure that it operates correctly.

3. **ALU**
   Implement and test a 64-bit ALU, supporting the following operations:

Opcode	Operation
1	AND
2	OR
3	NOT
4	MOVA: Out <= A
5	MOVB: Out <= B
6	Add
7	Subtract

8	Any operation that you may need for implementation

The ALU also has a Zero output flag.

4. **Data Memory**
   Implement and test a data memory module. This module can be similar in structure to the register file. A good size would be twice the size of the register file.

5. **Control logic**
   Implement and test a unified control unit, which combines the functionality of the central control unit and the ALU control unit. The input should be an instruction, and the outputs are the control inputs to all the datapath elements.

Task 2: Integrated Datapath

1. In a new project, implement and test a working datapath that instantiates and connects all the datapath components.
2. Modify the content of the instruction memory to include a complete test assembly program, and demonstrate the workings of your datapath with a waveform.

First Milestone Deliverables

Completed modules and working testbenches for the following portions of the datapath, and submit Verilog files of the modules and testbenches:

- Register file – read and write operations
- Data memory – read and write operations
- ALU – all operations and zero flag
- Incrementing PC counter and branch selection control

Second Milestone Deliverables

Using the modules form task 1, draw a complete block diagram of the single cycle data path with the necessary wires. Make sure to name your wires with names that allow quick recognition of its role instead of letter labels, such as A, B, C… etc. Submit the diagram as a PDF.

Final Deliverables

- Submit your Verilog code and testbench for each datapath component and the integrated datapath. Also submit a pdf with a waveform and a description of the tests done for each component, and a description of your implementation of the movz instruction, and any other design choices that you made. In your pdf, also include your block diagram from milestone 2.
- Sign-up to demo your design to a TA. Come prepared to answer questions on your design.