EE 361L Fall 2018
Pipelined LEGLite

November 13, 2018

1 Introduction

LEGLite-Pipe is pipelined version of LEGLite-Single, as described in Homework 6. It has five pipeline stages just as the pipelined LEGv8 in Chapter 4 in the textbook as shown in Figure 1. Note that all sequential circuit components (e.g., PC, data memory) are synchronized with the positive clock edge; except the register file, which is synchronized with the negative clock edge.

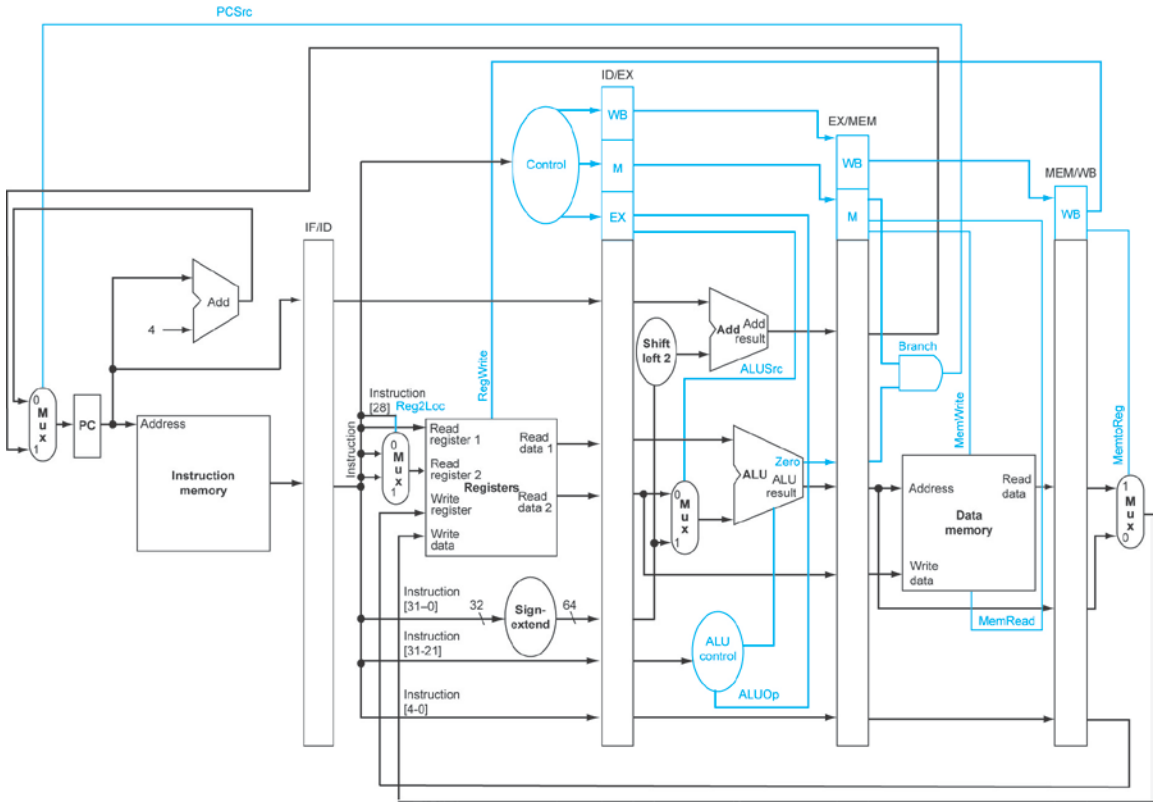

Figure 1. Five-stage Pipeline LEGv8 (from textbook).

The lab assignment is to design and implement LEGLite-Pipe on an FPGA.

Since we are short on time, we will simplify the design of LEGLite-Pipe:

- It uses parts from LEGLite-Single (homework 6) including Parts.V, IM1.V, and IM2.V.
- It only executes one instruction at a time. In particular, after an instruction is issued into the pipeline, the subsequent instructions are stalled for 3 clock cycles. In other words, there are three "bubbles" following each instruction. Thus, each instruction takes 4 clock cycles. (Comment: Due to its simplicity, the computer does not utilize its pipeline architecture, unfortunately.)

The following is an example of how the computer works. Suppose the computer is executing the following program fragment in instruction memory.

Address	Instruction
0	ADD X1,X2,X3
2	ANDI X4,X5,#9
4	CBZ X4,Loop
6	ADDI X5,X3,X1

The following shows the instructions being processed in the computer. Note that bubbles are introduced in the ID stage. The example assumes that the branch CBZ X4,Loop is taken to address 0. The example shows, per clock cycle, the PC value and the contents per stage of the pipeline. It shows the controller state, which is assumed to be a 2-bit value. The controller is essentially a 2-bit up-counter.

Controller State	**PC**	**IF**	**ID**	**EX**	**MEM**	**WB**
0	0	**ADD X1,X2,X3**	Bubble	???	???	???
1	0	ADD X1,X2,X3	**ADD X1,X2,X3**	Bubble	???	???
2	0	ADD X1,X2,X3	Bubble	**ADD X1,X2,X3**	Bubble	???
3	0	ADD X1,X2,X3	Bubble	Bubble	**ADD X1,X2,X3**	Bubble
0	2	**ANDI X4,X5,#9**	Bubble	Bubble	Bubble	**ADD X1,X2,X3**
1	2	ANDI X4,X5,#9	**ANDI X4,X5,#9**	Bubble	Bubble	Bubble
2	2	ANDI X4,X5,#9	Bubble	**ANDI X4,X5,#9**	Bubble	Bubble
3	2	ANDI X4,X5,#9	Bubble	Bubble	**ANDI X4,X5,#9**	Bubble
0	4	**CBZ X4,Loop**	Bubble	Bubble	Bubble	**ANDI X4,X5,#9**
1	4	CBZ X4,Loop	**CBZ X4,Loop**	Bubble	Bubble	Bubble
2	4	CBZ X4,Loop	Bubble	**CBZ X4,Loop**	Bubble	Bubble
3	4	CBZ X4,Loop	Bubble	Bubble	**CBZ X4,Loop**	Bubble
0	0	**ADD X1,X2,X3**	Bubble	Bubble	Bubble	**CBZ X4,Loop**
1	0	ADD X1,X2,X3	**ADD X1,X2,X3**	Bubble	Bubble	Bubble
Etc	Etc	Etc	Etc	Etc	Etc	Etc

As stated earlier, the controller is a 2-bit up-counter. In states 0, 2, and 3, it inserts a bubble at the ID stage; but in state 1, it allows the instruction to proceed down the pipeline. In states 0, 1, and 2, it disables the PC; but in state 3, it updates the PC.

2 Assignment

There are three parts to this assignment which is organized into three subprojects. You are required to write an individual report, which will be graded for writing style. It must have at least 4 pages of text.

Subproject 1: Implement LEGLite-Pipe in System verilog so that it can run the program in IM1.V from Homework 6. Thus, it must be able to execute “ADD”, “ADDI”, and “CBZ”. Use the System verilog files from Homework 6. Also use the attached files in a folder Subproject 1:

- LEGLitePipe.V: This is a verilog file for the LEGLite-Pipe. It’s very incomplete and may be buggy.
- testbenchLEGLitePipe.V: This is the testbench for the processor.

You have to design your own controller, and PC logic.

Before writing any System verilog code,

- Draw a block diagram of your computer. It should be similar to Figure 1 but for LEGLite-Pipe.
- Understand completely how the computer should operate.

Demonstrate it to your TA.

Subproject 2: Modify the LEGLite-Pipe in verilog so that it can also execute LD and ST. It should be able to run the program in IM2.V from Homework 6.

There is a folder for Subproject 2, which has a testbench: testbenchSubproject2. Recall that program IM2.V will constantly check switch 0 of the I/O to check if it's 0 or 1. If it's 0 then the 7-segment display is set to display "0". If the switch is 1 then the 7-segment display is set to display "2".

Demonstrate it to your TA.

Subproject 3: Configure the LEGLite-Pipe from Subproject 2 into the FPGA in the Basys3 board.
You must create a System verilog module FPGADevice.V that has components:

- instruction memory
- data memory and I/O
- CPU.

There is a folder for Subproject 3 which shows what this System verilog module should look like. It may be buggy. There is also a testbench for it. Once you have the module synthesized, connect its input port to a sliding switch of the Basys board, and its output port to a 7 segment display. Then you can download the bit file into the FPGA. Demonstrate it to your TA.

3. Grading

Total points for this lab is 40

- Writing Style: 20 points for the written report. No revisions.
- LEGLite-Pipe Implementation: 20 points depends on how far the student gets
    - Subproject 1 = 14 points
    - Subprojects 1 and 2 = 16 points
    - Subprojects 1, 2, and 3 = 20 points