CMPEN 331 – Computer Organization and Design, Lab 5

This lab introduces the idea of the pipelining technique for building a fast CPU. The students will obtain experience with the design implementation and testing of the first four stages (Instruction Fetch, Instruction Decode, Instruction Execute, Memory) of the five-stage pipelined CPU using the Xilinx design package for FPGAs. It is assumed that students are familiar with the operation of the Xilinx design package for Field Programmable Gate Arrays (FPGAs) through the Xilinix tutorial available in the class website.

1. Pipelining

As described in lab 4

2. Circuits of the Instruction Fetch Stage

As described in lab 4

3. Circuits of the Instruction Decode Stage

As described in lab 4

4. Circuits of the Execu

As described in lab 5

5. Circuits of the Memory Access Stage

As described in lab 5

6. Circuits of the Write Back Stage

Referring to Figure 1, in the fifth cycle the first instruction entered the WB stage. The memory data is selected and will be written into the register file at the end of the cycle. All the control signal have a prefix “w”. The second instruction entered the MEM stage; the third instruction entered the EXE stage; the fourth instruction is being decoded in the ID stage; and the fifth instruction is being fetched in the IF stage. All the six pipeline registers are updated at the end of the cycle (the destination register is considered as the six pipeline register). Then the first instruction is committed. In each of the forth coming clock cycles, an instruction will be commited and a new instruction will enter the pipeline. We use the structure shown in Figure 1 as a baseline for the design of our pipelined CPU.

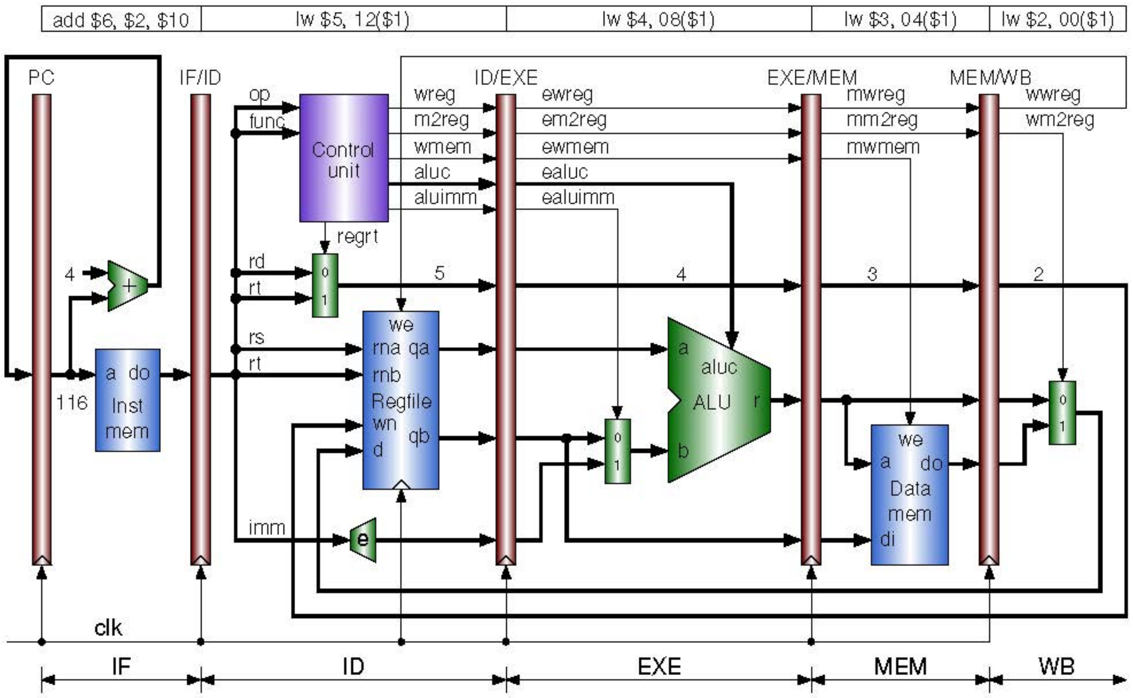

Figure 1 Pipeline write back (WB) stage

7. Table 1 lists the names

Table 1 MIPS general purpose register

Register Name	Register Number	Usage
$zero	0	
$at	1	Reserved for assembler
$v0, $v1	2, 3	Function return values
$a0 - $a3	4 – 7	Function argument values
$t0 - $t7	8 – 15	Temporary (caller saved)
$s0 - $s7	16 – 23	Temporary (callee saved)
$t8, $t9	24, 25	Temporary (caller saved)
$k0, $k1	26, 27	Reserved for OS Kernel
$gp	28	Pointer to Global Area
$sp	29	Stack Pointer
$fp	30	Frame Pointer
$ra	31	Return Address

8. Table 2 lists some MIPS instructions that will be implemented in our CPU

Table 2 MIPS integration instruction

9. Initialize the first 10 words of the Data memory with the f

A00000A
1000001
2000002
30000033
40000044
50000055
60000066
70000077
80000088
90000099

10. Write a Verilog code that implement the following instructions using the design shown in Figure 1. Write a Verilog test bench to verify your code: (You have to show all the signals written into and out from the MEM/WB register and the inputs to the Regfile block in your simulation outputs)

```
instruction                 comment
lw $2, 00($1)         # $2 ← memory[$1+00]; load x[0]
lw $3, 04($1)         # $3 ← memory[$1+04]; load x[1]
lw $4, 08($1)         # $4 ← memory[$1+08]; load x[2]
lw $5, 12($1)         # $5 ← memory[$1+12]; load x[3]
add $6, $2, $10
```

`Assume that the register $1 has the value of 0`

11. Write a report that contains the following:
    a. Your Verilog design code. Use:
        i. Device: XC7Z010- CLG400 -1 or choose any other FPGA type. You can use Arria II if you are using Quartus II software.
    b. Your Verilog® Test Bench design code. Add "`timescale 1ns/1ps" as the first line of your test bench file.
    c. The waveforms resulting from the verification of your design with ModelSim showing all the signals written in and out from the MEM/WB register and the inputs to the Regfile block.
    d. The design schematics from the Xilinx synthesis of your design. Do not use any area constraints.
    e. Snapshot of the I/O Planning and
    f. Snapshot of the floor planning

12. REPORT FORMAT: Free form, but it must be:
    g. One report per student.
    h. Have a cover sheet with identification: Title, Class, Your Name, etc.
    i. Using Microsoft word and it should be uploaded in word format not PDF. If you know LaTex, you should upload the Tex file in addition to the PDF file.
    j. Double spaced

13. You have to upload the whole project design file zipped with the word file.

14. **For students who to** [illegible] **all of the above requirements, you need to design t**

In order to focus our attention o [illegible] re 1 is redrawn by putting the register file on the WB stage where the execution result of an instr [illegible] in Figure 2. The state of the art content can be read correctly in the ID stage after it is written [illegible] Data hazards occur in the code example shown in Figure 3. You need to add the required c [illegible] zard problem and do forwarding if needed.

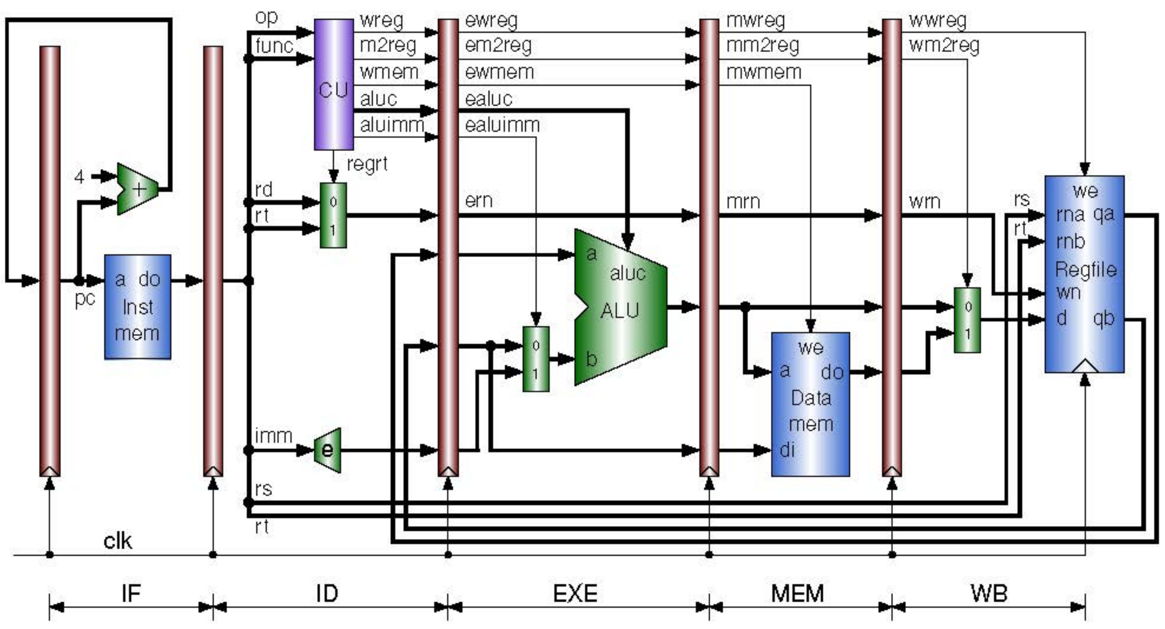

Figure 2 Writing result to the register file in the write back stage

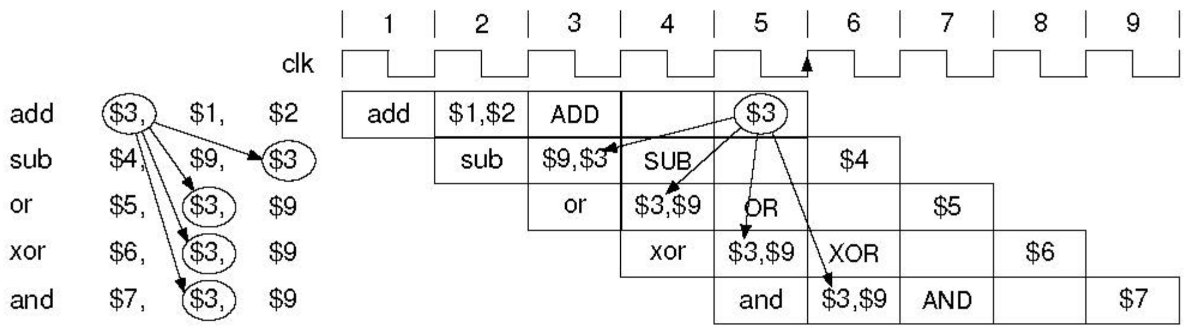

Figure 3 Data hazard examples

15. Initialize the first 11 words of the register (Regfile block) with the following HEX values:

00000000
A00000AA

4000004
5000005
6000006
7000007
80000088
90000099

16. Write a Verilog code that implement the instructions using the design shown in **Figure 3**. Write a Verilog test bench to verify your code: (You have to show qa and qb signals that output from the Regfile block in your simulation outputs)

17. Write a report that contains the following:
    k. Your Verilog design code. Use:
        i. Device: Zyboboard (XC7Z010- -1CLG400C)
    l. Your Verilog® Test Bench design code. Add "`timescale 1ns/1ps" as the first line of your test bench file.
    m. The waveforms resulting from the verification of your design with ModelSim showing all the signals written in and out from the MEM/WB register and the inputs to the Regfile block.
    n. The design schematics from the Xilinx synthesis of your design. Do not use any area constraints.
    o. Snapshot of the I/O Planning and
    p. Snapshot of the floor planning
    q. Generate the bitstream.
    r. The design should be free from errors when synthesized, implemented and generated of bit stream.

18. You have to upload the whole project design file zipped with the word file.