CMPE 110 - Fall 2016

Jishen Zhao

Homework 2

Seongwoo Choi

Student ID: 1368039

Scho29

Question 1. Datapath (16 Points)

In this problem, you will modify the single-cycle datapath we built up in class to implement JAL instruction. The MIPS jump and link instruction, JAL, is used to support procedure calls by jumping to jump address (similar to j) and saving the address of the following instruction PC+4 in register $ra ($31), i.e.,

JAL: R[31] ← PC + 4

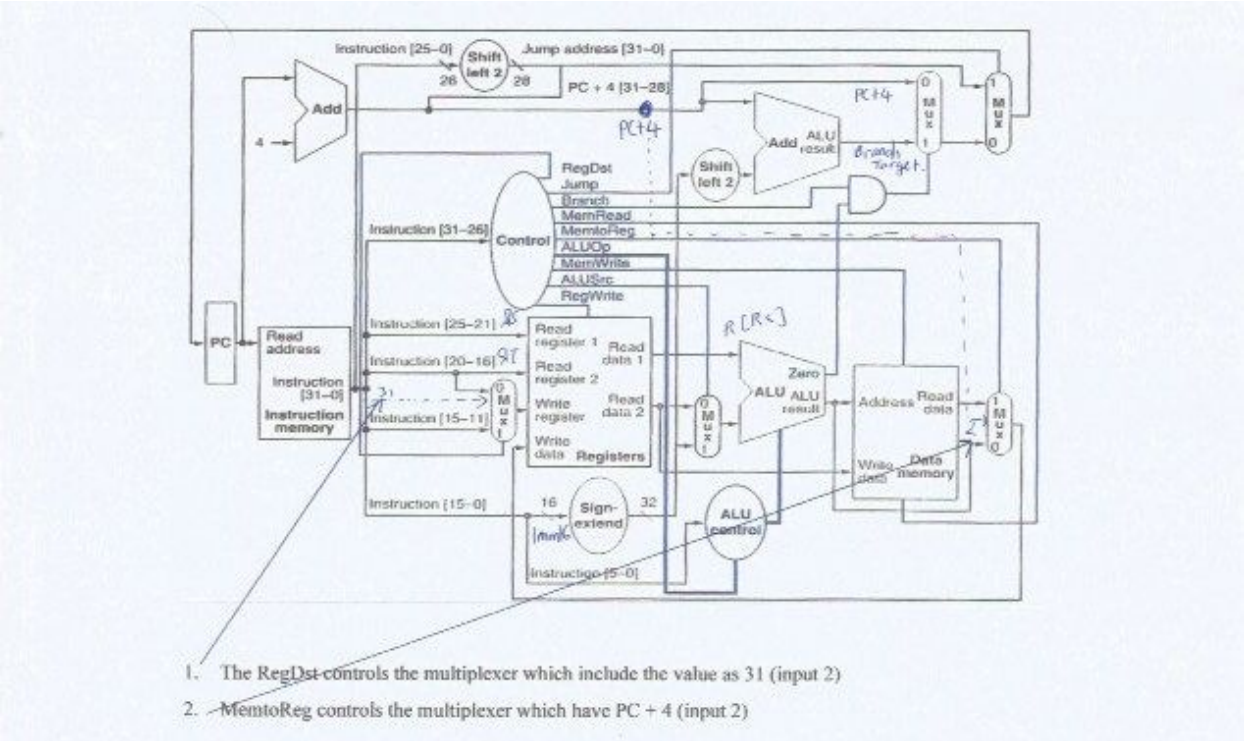
PC ← Jump Address

JAL uses the j instruction format:

JAL Address:

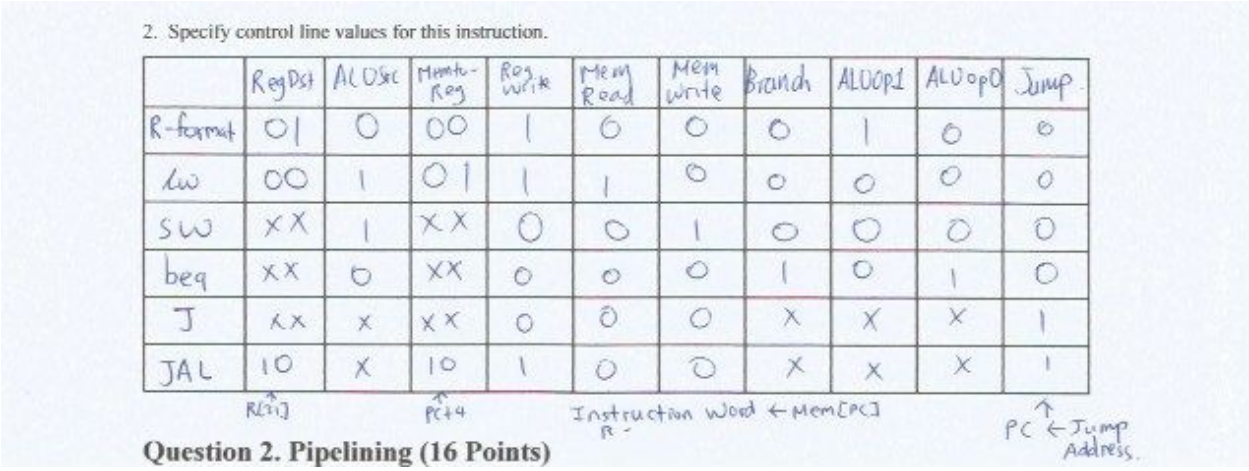
op (6 bits) | Target address (26 bits)

1. Add to the provided datapath the necessary data and control signals to implement the JAL instruction. Draw and label all components and wires very clearly (give control signals meaningful names; if selecting a subset of bits from many, specify exactly which bits are selected; and so on). Justify the need for the modifications, if any.



2. Specify control line values for this instruction.

MemtoReg and RegDst are now 2 bits,



R[31] <- PC + 4

PC<-Jump Address

Question 2. Pipelining (16 Points)

Given the following code: xor r0, r0, r0

```
addiu     r1, r0, 10
          j L1
loop:     lw r3, 0(r2)
          mul r4, r3, r3
          mul r3, r3, r1
          addiu r0, r0, 1
          div r3, r4, r3
          sw r3, 0(r2)
          addiu r2, r2, 4
L1:       bne r0, r1, -8
```

Calculate the number of cycles it takes to execute one iteration of the loop (including the bne instruction) on the following models:

2.1 A non-pipelined machine

Cycles	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	24	26	27	28	29	30	31	32
xor r0, r0, r0	F	D	E	M	W																											
addiu r1, r0, 10		F	D	D	D	E	M	W																								
j L1			F	F	F	D	E	M	W																							
loop: lw r3, 0(r2)						F	D	E	M	W																						
mul r4, r3, r3							F	D	E	E	E	E	M	W																		
mul r3, r3, r1								F	D	D	D	D	D	D	E	E	E	E	M	W												
addiu r0, r0, 1									F	F	F	F	F	F	D	D	D	D	D	D	E	M	W									
div r3, r4, r3															F	F	F	F	F	F	D	E	E	E	E	M	W					
sw r3, 0(r2)																					F	F	F	F	F	D	D	E	M	W		
addiu r2, r2, 4																										F	F	D	E	M	W	
L1: bne r0, r1, -8																												F	D	E	M	W

Number of cycles for a non-pipelined machine = 32 cycles.

2.2 A pipelined machine with scoreboarding and five adders and five multipliers without data forwarding

2.2

Cycles	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	24	26	27	28	29	30	31	32	33	34
xor r0, r0, r0	F	D	E	M	W																													
addu r1, r0, 10		F	D	E	E	E	E	M	W																									
↑L1			F	D	-	-	-	E	M	W																								
loop: lw r3, 0(r2)								F	D	E	M	W																						
mul r4, r3, r3									F	D	E	E	E	E	E	E	-	M	W															
mul r3, r3, r1									F	D	-	-	-	-	-	-	E	E	M	W														
addu r0, r0, 1										F	D	-	-	-	-	-	E	E	E	E	M	W												
div r3, r4, r3											F	F	F	F	F	F	D	E	E	E	E	M	W											
sw r3, 0(r2)																	F	F	F	F	F	F	D	D	E	M	W							
addu r2, r2, 4																							F	D	-	E	E	E	E	M	W			
L1: bne r0, r1, -8																								F	D	-	-	-	-	E	M	W		

Number of cycles = 31 cycles.

2.3 A pipelined machine with scoreboarding and five adders and five multipliers with data forwarding.

Cycles	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	24	26	27	28	29	30	31	32	33	34
xor r0, r0, r0	F	D	E	M	W																													
addu r1, r0, 10		F	D	E	E	E	E	M	W																									
↑L1			F	F	F	D	E	M	W																									
loop: lw r3, 0(r2)						F	D	E	M	W																								
mul r4, r3, r3							F	D	E	E	E	E	E	E	M	W																		
mul r3, r3, r1								F	D	E	E	E	E	E	E	M	W																	
addu r0, r0, 1									F	-	-	-	-	-	D	E	E	E	E	M	W												
div r3, r4, r3															F	F	F	F	F	F	D	E	E	E	E	M	W							
sw r3, 0(r2)																					F	F	F	F	F	D	D	E	M	W				
addu r2, r2, 4																										F	D	E	E	E	E	M	W	
L1: bne r0, r1, -8																											F	D	-	-	-	E	M	W

Number of cycles =34

2.4 A pipelined machine with scoreboarding and one adder and one multiplier without data forwarding

Instructions/cycles	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29
MUL R3, R1, R2	F	D	E1	E2	E3	E4	E5	W1	W2																				
ADD R5, R4, R3		F	/	/	/	/	/	/	/	D	E1	E2	W1	W2															
ADD R6, R4, R1										F	/	D	E1	E2	W1	W2													
MUL R7, R8, R9												F	/	D	E1	E2	E3	E4	E5	W1	W2								
ADD R4, R3, R7														F	D	E1	E2	W1	W2										
MUL R10, R5, R6															F	D	E1	E2	E3	E4	E5	W1	W2						

2.5 A pipelined machine with scoreboarding and one adder and one multiplier with data forwarding

Instructions/cycles	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
MUL R3, R1, R2	F	D	E1	E2	E3	E4	E5	W															
ADD R5, R4, R3		F	/	/	/	/	D	E1	E2	W1	W2												
ADD R6, R4, R1							F	/	D	E1	E2	W1	W2										
MUL R7, R8, R9									F	D	E1	E2	E3	E4	E5	W1	W2						
ADD R4, R3, R7										F	/	/	/	/	D	E1	E2	W1	W2				
MUL R10, R5, R6															F	D	E1	E2	E3	E4	E5	W1	W2

Note: For all machine models, use the basic instruction cycle as follows:

- Fetch (one clock cycle)
- Decode (one clock cycle)
- Execute (MUL takes 6, ADD takes 4 clock cycles, other instructions 1 clock cycle). The multiplier and the adder are NOT pipelined.
- Memory access (one clock cycle)
- Write-back (one clock cycle)

Do not forget to list any assumptions you make about the pipeline structure (e.g., how is data forwarding done between pipeline stages).

Question 3. Branch prediction (16 Points)

Assume a machine with a 7-stage pipeline. Assume that branches are resolved in the sixth stage. Assume that 30% of instructions are branches.

3.1 How many instructions of wasted work are there per branch mis-prediction on this machine?
There are five instructions.

3.2
Assume 1000 instructions are on the correct path of a program and assume a branch predictor accuracy of 10%. How many instructions are fetched on this machine? (Please show your work for full credit.)
We need to use the formula for the given problem. Branch instructions contain 30%. So 0.3 for branch instruction. We can assume the current path instruction is M, the number of instructions in incorrect path is M(0.3)(1-B)*5 = 1.5*M(1-B). Here the branch predictor accuracy is B. Therefore, we can calculate the fetched instructions (FI) using the below formula.

That is FI = correct path instructions + Incorrect path instructions
= M + 1.5*M(1-B)
= 2.5 * M - 1.5 * M * B
FI = M (2.5 - 1.5 B)
In the given question, N = 1000 and B = 10% = 0.1
Therefore, the FI = 1000 + 1000 (1 - 0.1)
= 1000 + 1000 (0.9)
= 1000 + 900

So FI = 1900 instructions

3.3 Let's say we modified the machine so that it used dual path execution (where an equal number of instructions are fetched from each of the two branch paths). Assume branches are resolved before new branches are fetched. Write how many instructions would be fetched in this case, as a function of N. (Please show your work for full credit.)
Here also we can consider the numbers of instructions in correct path is N, number of instructions in the incorrect path is N (0.3) 5. So the FI is equal to the correct path instructions plus incorrect path instructions.

That is FI = N + 1.5 * N (1-0.7) * 5
= N + 1.5 * N * 0.3 * 5
= N + 2.25 * N
Finally, we can get it the FI 3.25 * N.

3.4
Now let's say that the machine combines branch prediction and dual path execution in the following way:

A branch confidence estimator, like we discussed in class, is used to gauge how confident the machine is of the prediction made for a branch. When confidence in a prediction is high, the branch predictor's prediction is used to fetch the next instruction; When confidence in a prediction is low, dual path execution is used instead.

Assume that the confidence estimator estimates a fraction C of the branch predictions have high confidence, and that the probability that the confidence estimator is wrong in its high confidence estimation is M.

Write how many instructions would be fetched in this case, as a function of N, A, C, and M. (Please show your work for full credit.)

Here we can assume that the number of instructions in the correct paths is N.
In the case of lack of confidence present in the incorrect path instructions is N (0.3) (1-C) 5.
That is 1.5 * N (1-C).
The number of instructions in the estimation of high confidence is N (0.3) * C * M * 5 = 1.5* N * C * M.
Therefore, the fetched instruction (FI) is calculated using the below formula.
FI = the number of instructions in correct path and lack of confidence present in the incorrect and also number of instructions in the estimation of high confidence.

= N + 1.5 * N (1 - C) + 1.5 * N * C * M
= N + 1.5 * N - 1.5*N*C + 1.5 * N * C * M
= 2.5 * N - 1.5 * N * C + 1.5 * N * C * M
= N (2.5 - 1.5 * C + 1.5 * C * M)

= N (2.5 + 1.5 * C(M-1))

Question 4. Out of order execution (16 points)

In this problem, we will give you the state of the Register Alias Table (RAT) and Reservation Stations (RS) for an out of order execution engine. Your job is to determine the original sequence of the following five instructions in program order.

```
ADD R3, R10, R3
MUL R1, R1, R10
MUL R11, R7, R9
ADD R1, R2, R3
ADD R5, R1, R11
ADD R7, R2, R6
```

The out-of-order machine in this problem behaves as follows:

- The frontend of the machine has a one-cycle fetch stage and a one-cycle decode stage. The machine can fetch one instruction per cycle, and can decode one instruction per cycle.
- The machine dispatches one instruction per cycle into the reservation stations, in program order. Dispatch occurs during the decode stage.
- An instruction always allocates the first reservation station that is available (in top-to-bottom order) at the required functional unit.
- When a value is captured (at a reservation station) or written back (to a register) in this machine, the old tag was previously at that location is not cleared; only the valid bit is set.
- When an in a reservation station finishes executing, the reservation station is cleared.
- Both the adder and multiplier are fully pipelined. Add instructions take 4 cycles. Multiply instructions take 6 cycles.
- When an instruction completes execution, it broadcasts its result, and dependent instructions can begin execution in the next cycle if they have all operands available.
- When multiple instructions are ready to execute at a functional unit, the oldest ready instruction is chosen.

Initially, the machine is empty. Six instructions then are fetched, decoded, and dispatched into reservation stations, before any instruction executes. Then, one instruction completes execution. Here is the state of the machine at this point, after the single instruction completes:

4.1
Give the six instructions that have been dispatched into the machine, in program order. The source registers for the first instruction can be specified in either order. Give instructions in the following format: “opcode destination ⇐ source1, source2.”

```
ADD R1, R2, R3
MULT R1, R10, R1
ADD R7, R2, R6
MULT R11, R7, R9
ADD R5, R1, R11
ADD R3, R10, R3
```

4.2
Now assume that the machine flushes all instructions out of the pipeline and restarts execution from the first instruction in the sequence above. Show the full pipeline timing diagram below for the sequence of six instructions that you determined above, from the fetch of the first instruction to the writeback of the last instruction. Assume that the machine stops fetching instructions after the sixth instruction. Use “F” for fetch, “D” for decode, “E1,” “E2,” “E3,” “E5,” “E4,” and “E6” to signify the first, second, third and fourth cycles of execution for an instruction (as required by the type of instruction), and “W” to signify writeback. You may or may not need all columns shown.

INSTRUCTION	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
ADD R1, R2, R3	F	D	E1	E2	E3	E4	W										
MULT R1, R10, R1		F	D	/	/	/	E1	E2	E3	E4	E5	E6	W				
ADD R7, R2, R6			F	D	E1	E2	E3	E4	/	/	/	/	/	W			
MULT R11, R7, R9				F	D	/	/	/	E1	E2	E3	E4	E5	E6	W		
ADD R5, R1, R11					F	D	/	/	/	/	/	E1	E2	E3	E4	W	
ADD R3, R10, R3						F	D	E1	E2	E3	E4	/	/	/	/	/	W

4.3 Finally, show the state of the RAT and reservation stations after 9 cycles in the blank figures below.

Reg	V	Tag	Value
R1	0	X	10
R2	1		2
R3	0	D	3
R4	1		4
R5	1	C	5
R6	0		6
R7	0	B	7
R8	1		8
R9	1		9
R10	1		10
R11	0	Y	11

ADD:

	Src 1			Src 2		
	V	Tag	Value	V	Tag	Value
A	0	--	2	1	--	3 —
B	1	--	2	0	--	6
C	0	X	5	0	Y	72
D	0	--	5	1	--	3

MULT:

	Src 1			Src 2		
	V	Tag	Value	V	Tag	Value
X	1	--	10	0	A	5
Y	1	B	8	0	--	9
Z	0	--	10	0	C	10
T						