Computer Organization & Assembly Language

Final Exam – 2016/1/7

Dept. of Engineering Science, National Cheng Kung University

1. Answer *True* or *False* to the following statements. (24%)

(A) When representing a negative number, the MSB is used as the sign bit.

(B) When performing the binary division operation for computers, it may not be necessary to subtract the Divisor from the Remainder in all iterations.

(C) In computer arithmetic, dividend and remainder must have the same signs when performing the signed division.

(D) Since computer arithmetic is with limited precision, computation results obtained from computers are usually erroneous and suspicious.

(E) The register is considered as a type of combinational elements.

(F) Designing the control part of a processor is straightforward since it can be easily built on the basis of existing datapath.

(G) The critical path in the single-cycle design is referred to as the `lw` instruction.

(H) Reordering code is always possible to avoid pipeline stalls.

2. Perform the following operations of computer arithmetic.

(A) Subtract 0x001F from 0xFF17, interpreting each as 2's complement 16-bit integer. Express the answer both in hex and decimal. (6%)

(B) Assuming single precision IEEE 754 format, what decimal number is represent by this word: 10111110100100000000000000000000 ? (4%)

3. Assume that A=1/3 and B=3. Answer the following questions.

(A) Using the IEEE 754 floating-point format, write down the bit pattern that would represent A (in single precision). Can you represent A exactly? (4%)

(B) What do you get if you add A to itself B times? What is A × B? Are they the same? What should they be? Finally, provide some brief comment on this. (10%)

4. Given the following latencies for the blocks in our datapath, what is the corresponding clock rate in the single-cycle design and the pipelined design, respectively? (4%)

Instruction access: 2 ns
Register read: 1 ns
ALU operation: 2 ns
Data memory access: 2 ns
Register write-back: 1 ns

5. Structural, data and control hazards typically required a processor pipeline to stall. For each of the following optimization techniques, state which pipeline hazard(s) it addresses. Note that some optimization techniques may address more than one hazard. (12%)

(A) branch prediction

(B) instruction scheduling

(C) forwarding

(D) increasing availability of functional units (i.e., ALUs, adders, ... etc)

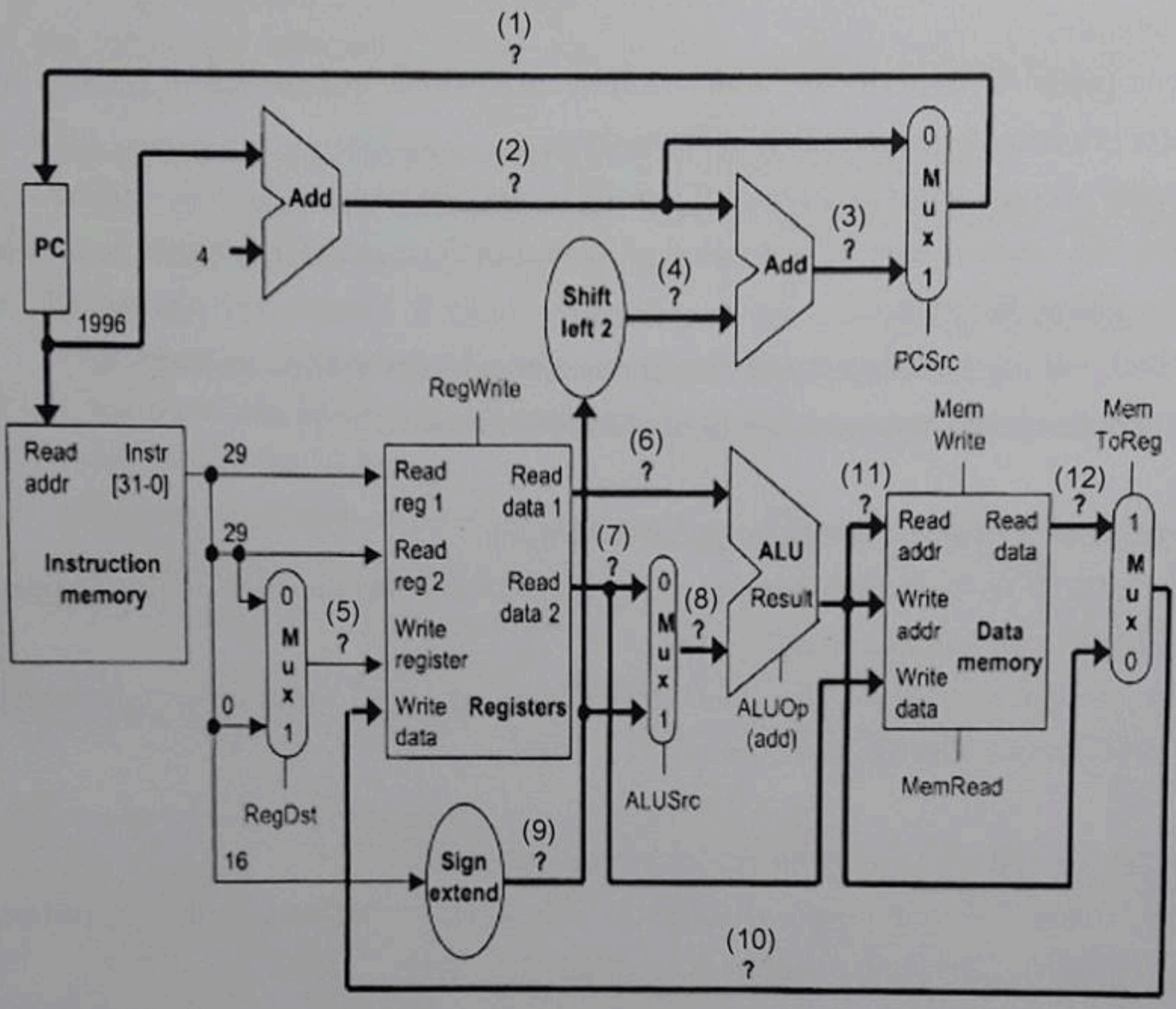
6. Let's say we want to execute the following immediate addition instruction in the single-cycle datapath:

```
addi $29, $29, 16
```

The single-cycle datapath diagram below shows the execution of this instruction. Several of the datapath values are filled in already. You are to provide values for the twelve remaining signals in the diagram, which are marked with a '?' symbol. (24%)

Note that you should:

- Show decimal values.
- Assume register $29 initially contains the number 129.
- If a value cannot be determined, mark it as 'X'.



7. Consider a MIPS machine with a 5-stage pipeline with a cycle time of 10ns. Assume that you are executing a program where a fraction *f* of all instructions immediately follow a load upon which they are dependent. Also, there is no other data hazard or control hazard.

(A) With forwarding enabled, what is the total execution time for *N* instructions (in terms of *f*)? (4%)

(B) Consider a scenario where the MEM stage, along with its pipeline registers, needs 12ns. There are now two options: *add another MEM stage so that there are MEM1 and MEM2 stages* or *increase the cycle time to 12ns so that the MEM stage fits within the new cycle time and the number of pipeline stages remain unaffected.* For a program mix with the above characteristics, when is the first option better than the second. Your answer should be based on the value of *f*. (You may ignore the first few cycles to fill the pipeline.) (4%)

(C) Embedded processors have two different memory regions - a faster scratchpad memory and a slower normal memory. Assume that in the 6-stage machine (with MEM1 and MEM2 stages), there is a region of memory that is faster and for which the correct value is obtained at the end of the MEM1 stage itself while the rest of the memory needs both MEM1 and MEM2 stages. For the sake of simplicity, assume that there are two load instructions *lw.fast* and *lw.slow* that indicate which memory region is accessed. If 40% of the fraction *f* mentioned above get their value from the fast memory, how does the answer to the previous question change? (4%)