Computer Organization & Assembly Language

Final Exam – 2013/1/11

Dept. of Engineering Science, National Cheng Kung University

1. Answer *True* or *False* to the following statements. (30%)

(A) A right shift instruction can replace an integer division by a power of 2 if no underflow occurs.

(B) In computer arithmetic, floating-point addition may not be associative; that is, x+(y+z) != (x+y)+z.

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

(D) Reordering code is a possible way to avoid pipeline stalls.

(E) In an instruction set like the 80x86, where instructions are not of the same length, pipelining is considerably easier.

(F) In a pipelined system, forwarding will eliminate the need of any stalls.

(G) Pipelining increases throughput and reduces individual instruction execution time.

(H) Pipeline bubbles cause loss of performance since that a number of pipeline stages are left empty.

(I) The CPI value of an ideal pipeline processor is the same as that of a single-cycle processor.

(J) Increasing the depth of pipeline (i.e., number of stages) without balancing the cost of each stage may reduce the overall performance.

2. Perform the following operations of computer arithmetic.

(A) Add $3.63_{ten} \times 10^4$ to $6.87_{ten} \times 10^3$, assuming that you have only three significant digits, first with guard and round digits and then without them. (5%)

(B) Assuming single precision IEEE 754 format, what decimal number is represent by this word: 10111110100100000000000000000000 ? (5%)

(C) Show the IEEE 754 binary representation for the floating-point number 20.5_{ten} and $-5/6_{ten}$ in single precision, respectively. (10%)

3. When using the IEEE 754 format, what is the meaning of *overflow* and *underflow*, respectively? (8%)

4. Suppose we want to execute the following code segment on the pipelined CPU of the textbook:

```
add     $2, $5, $4
add     $4, $2, $5
lw      $5, 100($2)
add     $3, $2, $5
```

Suppose there is no hardware supports for the forwarding and stalling, but the register file can write a register at the first half of a cycle and read it at the second half of the cycle.

(A) How many NOPs and at what places that you can add to make the code segment execute correctly on our pipeline? (4%)

(B) Suppose the pipeline stalls, but no forwards, when there is data hazard. How many cycles will the above code segment execute? (3%)

(C) How many cycles will the above code segment execute if a single-cycle CPU is used? (3%)

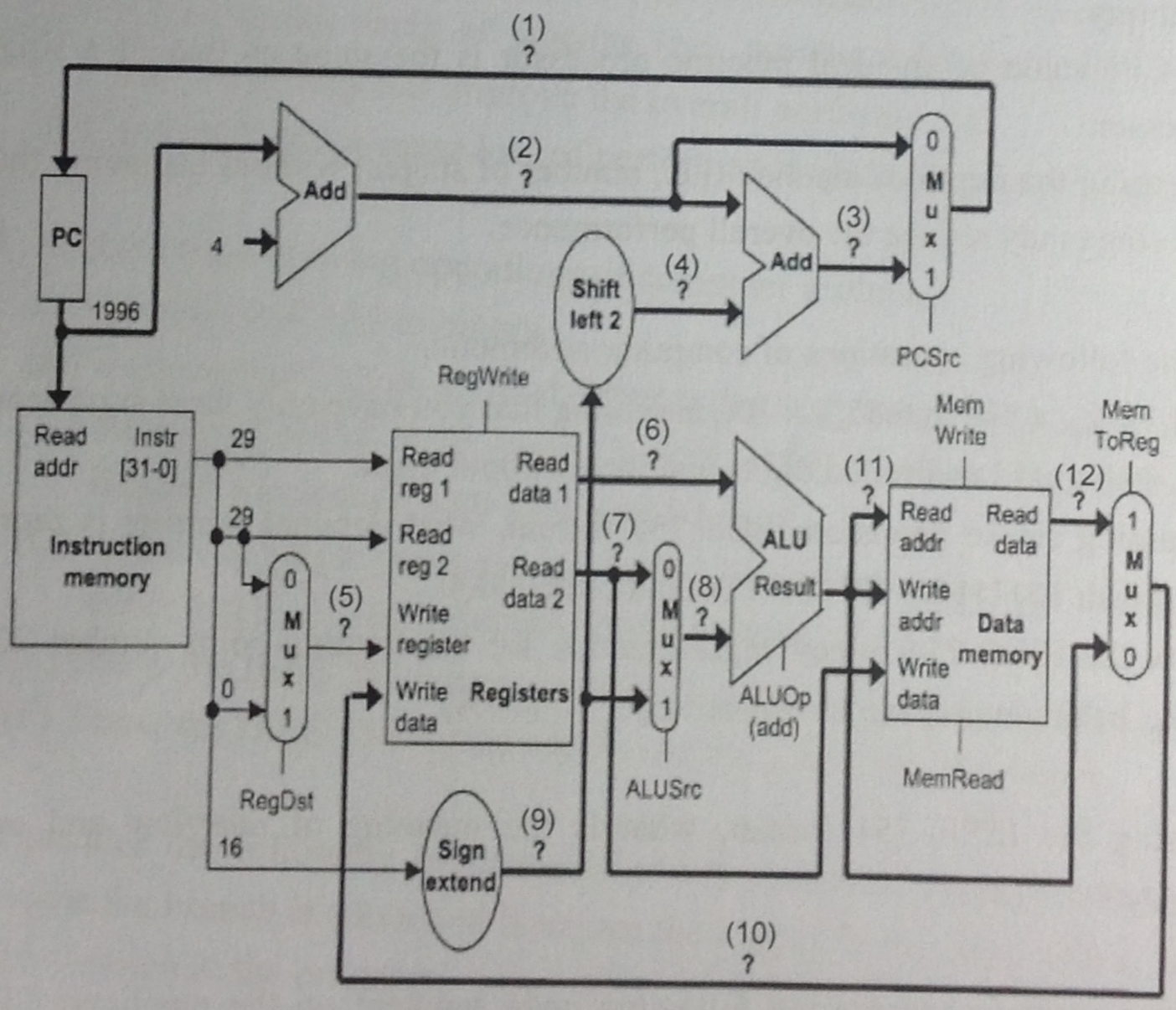
5. Let's say we want to execute the following immediate addition instruction in the single-cycle datapath:

```
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'.



6. 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? (8%)

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