Computer Sciences Department
University of Wisconsin-Madison
CS/ECE 552 – Introduction to Computer Architecture
In-Class Exercise (04/09)

Answers to all questions should be uploaded on Canvas.

1. [1 point] From the textbook (Twist on Check Yourself 4.10): State whether the following techniques or components are associated primarily with a software- or hardware-based approach to exploiting ILP. In some cases, the answer may be both:

    1. Branch prediction
    2. Multiple issue
    3. VLIW
    4. Superscalar
    5. Dynamic scheduling
    6. Out-of-order execution
    7. Speculation
    8. Reorder buffer
    9. Register renaming

    Given that the answer to 4 (superscalar) is "hardware", why is it not also associated with software?

    Virtualization allows it to handle by compiler and architect so user don't have to worry about that.

2. [9 points] Consider the following MIPS assembly program running on our standard **five-stage pipeline with full forwarding and bypassing**:

```
lw  $t0, 0($s2)
and $s2, $t0, $t1
sub $t2, $s0, $s2
lw  $t0, 4($t0)
sub $t0, $t0, $t2
lw  $t2, 8($s0)
or  $s1, $t2, $t0
lw  $t2, 0($t0)
add $t1, $t2, $s1
```

(a) [3 points] Reorder the assembly instructions above such that the program executes with the **minimum number of cycles** on the five-stage pipeline with full forwarding and bypassing. Your reordered code should not change the program's output. Do not add/remove/modify any instructions; **only reordering is allowed**. How many cycles does it take to execute the reordered program (from the IF stage of the first instruction to the WB stage of the last)?

lw $t0, 0($s2)
lw $t0, 4($t0)
and $s2, $t0, $t1
lw $t2, 4($s0)
sub $t2, $s0, $s2
sub $t0, $t0, $t2
lw $t2, 0($t0)
or $s1, $t2, $t0
add $t1, $t2, $s1

1 2 3 4 5 6 7 8 9 10 11 12 13 14
F D X M W
F D X M W
F D X M W
F D X M W
F D X M W
F D X M W
F D X M W
F D X M W

14.

(b) [3 points] Assume the physical register map table and free list are initialized as follows:

Map Table	
Architectural Reg	**Physical Reg**
$t0	P1
$t1	P2
$t2	P3
$s0	P4
$s1	P5
$s2	P6

Free List	P7, P8, P9, P10, P11, P12, P13, P14, P15, P16, P17, P18, P20

Apply register renaming (i.e., replace the architectural registers with physical registers) to the **original assembly program** above (not your reordered version), filling in the blanks below with physical registers. Instructions must be renamed in program order. Pop registers from the free list from left to right. You do not need to push registers back onto the free list.

lw P7, 0(P6)
lw P8, 4(P7)
and P9, P8, P2
lw P10, 4(P8)
sub P11, P8, P10 P4, P6
sub P9, P8, P11
lw P12, 0(P9)
or P13, P12, P9
add. P14, P12, P13

(c) [3 points] Reorder your **renamed** instructions (from 2B) such that the program executes with the **minimum number of cycles** on the five-stage pipeline with full forwarding and bypassing. Your reordered code should not change the program's output. Do not add/remove/modify any instructions; **only reordering is allowed**. How many cycles does it take to execute the reordered, renamed program (from the IF stage of the first instruction to the WB stage of the last)?

```
lw P7, 0(P6)
sub P11, P4, P6
lw P8, 4(P7)
~~and P9 P8, P2.~~
lw P10, 4(P8)
and P9, P8, P2.
~~lw P12, 0(P9)~~ sub P9, P8, P11
lw P12, 0(P9)
or P13, P12, P9
add P14, P12, P13
```

11