Brown University
School of Engineering
EN164 Design of Computing Systems
Professor Sherief Reda
HW 04. 150 points. Due Date: Tuesday, April 17 2012 in class.

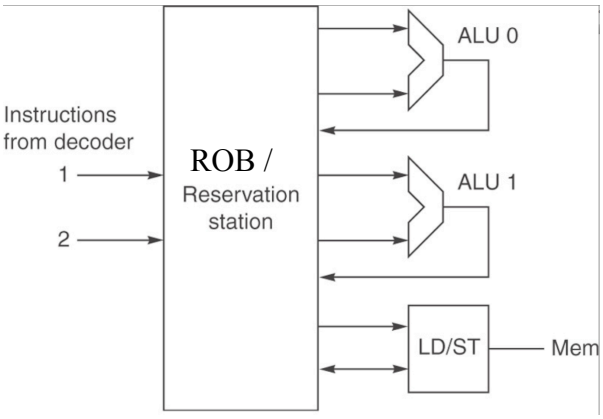
BROWN

1. (40 points – from final 2011) Consider the execution of a loop in a *single-cycle* VLIW processor. Assume that any combination of instruction types can be executed in the same cycle. Note that such assumption only removes resource constraints, but data and control dependencies must be still handled correctly. Assume that register reads occur before register writes in the same cycle.

```
loop: lw   $1, 40($6)
      add  $5, $5, $1
      sw   $1, 20($5)
      addi $6, $6, 4
      addi $5, $5, -4
      beq  $5, $0, loop
```

a. [20 points] Unroll this loop once and schedule it for a 2-issue static VLIW processor. Assume that the loop always executes an even number of iterations. Hint: for correct scheduling, you need to identify all potential data and control hazards. Use register renaming whenever possible to eliminate false dependencies.
b. [10 points] What is the speed-up of using your scheduled code instead of the original code? Assume that the loop will be executed a very large number of times.

2. (50 points) This exercise is about dynamic scheduling using renaming and reservation stations. Assume a microarchitecture as shown in the next figure. Assume that ALUs can do all arithmetic ops (MULTD, DIVD, ADDD, ADDI, SUB) and branches and that reservation station (RS) can dispatch at most one operation from the ROB to each functional unit per cycle (one to each ALU plus one memory op to the LD/ST unit). Assume the EX latency of the instructions are as follows: LD 4, SD 2, integer ADD/SUB 1, BRANCHES 2, ADDD 3, MULTD 5, DIVD 11.



Consider the following code sequence. ADDD, MULT, DIVD are operations on double precision floating points.

```
LD          F2, 0(Rx)
MULTD       F2, F0, F2
DIVD        F8, F2, F0
LD          F4, 0(Ry)
ADDD        F4, F0, F4
ADDD        F10, F8, F2
SD          F4, 0(Ry)
ADDI        Rx, Rx, 8
ADDI        Ry, Ry, 8
SUB         R20, R4, Rx
```

a. (10 points) Suppose all of the instructions of the code are presented in ROB, with no renaming having been done. Highlight any instructions in the code where register renaming would improve performance. Hint: Look for RAW and WAW hazards.

b. (20 points) Suppose the register-renamed version of the code from part (a) is resident in the ROB. Show how the RS should dispatch these instructions out-of-order, clock by clock, to obtain optimal performance on this code. Assume that results must be written into the ROB before they are available for use; i.e., no bypassing. How many clock cycles does the code sequence take?

c. (20 points) In reality the whole instruction sequence of interest is not usually present in the ROB. Instead, various events clear the ROB and as a new code sequence streams in from the decoder, the RS must choose to dispatch what it has. Suppose that the ROB is empty. In cycle 0 the first two register-renamed instructions of this sequence appear in the ROB. Assume it takes 1 clock cycle to dispatch any op. Further, assume that the front end (decoder, register/renamer) will continue to supply two new instructions per clock cycle. Show the cycle-by-cycle order of dispatch of the RS. How many clock cycles does this code sequence require now?