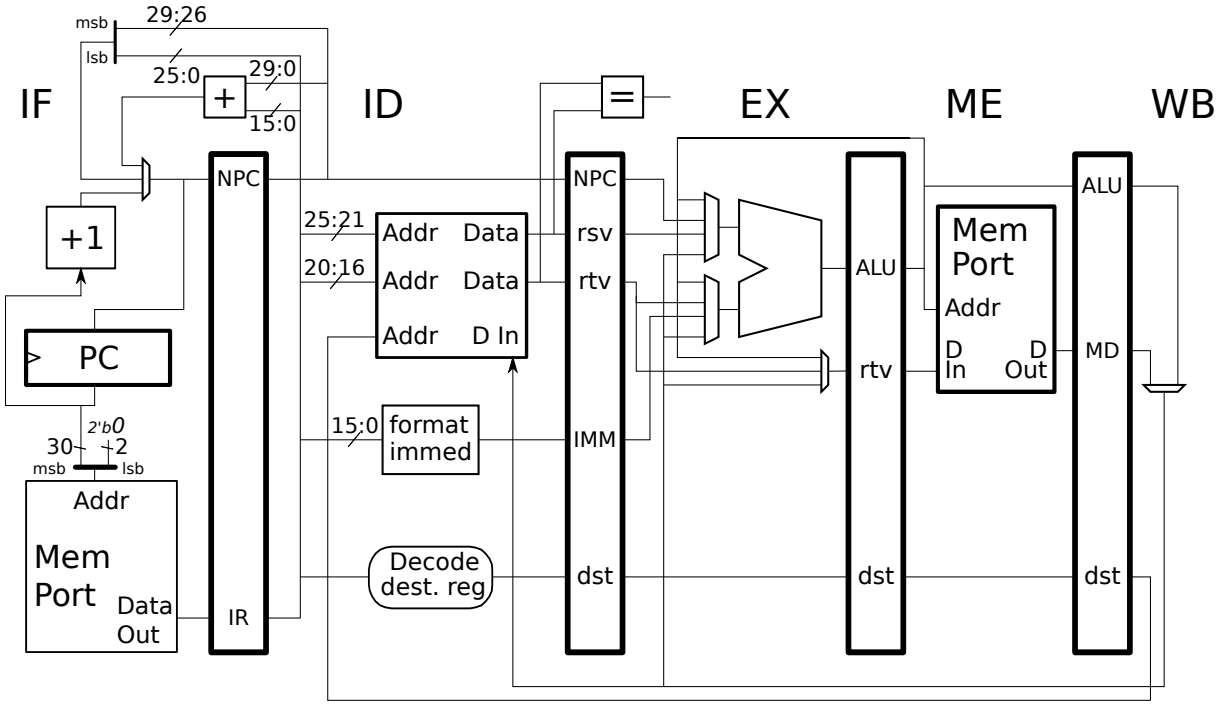
LSU EE 4720 **Homework 4** Solution **Due: 22 February 2019**

Problem 1: The three loops below (probably on the next page) copy an area of memory starting at the address in `r2` to an area of memory starting at the address in `r3`. The number of bytes to copy is in `r5`.



Solution appears on the next page.

(*a*) Show a pipeline execution diagram for each loop on the illustrated implementation.

Solution appears below.

```
# Loop A -- SOLUTION Part a
add r4, r3, r5      IF ID EX ME WB
LOOP: # Cycle       0  1  2  3  4  5  6  7  8  9  10 11 12 13 14  -- 1st Iter
 lb r1, 0(r2)          IF ID EX ME WB
 sb r1, 0(r3)             IF ID -> EX ME WB
 addi r2, r2, 1              IF -> ID EX ME WB
 addi r3, r3, 1                    IF ID EX ME WB
 bne r3, r4, LOOP                     IF ID ----> EX ME WB
 nop                                     IF ----> ID EX ME WB
LOOP: # Cycle       0  1  2  3  4  5  6  7  8  9  10 11 12 13 14  -- 2nd Iter
 lb r1, 0(r2)                                     IF ID EX ME WB


# Loop B -- SOLUTION Part a
add r4, r3, r5      IF ID EX ME WB
LOOP: # Cycle       0  1  2  3  4  5  6  7  8  9  10 11 12 13 14  -- 1st Iter
 lw r1, 0(r2)          IF ID EX ME WB
 sw r1, 0(r3)             IF ID -> EX ME WB
 addi r2, r2, 4              IF -> ID EX ME WB
 addi r3, r3, 4                    IF ID EX ME WB
 bne r3, r4, LOOP                     IF ID ----> EX ME WB
 nop                                     IF ----> ID EX ME WB
LOOP: # Cycle       0  1  2  3  4  5  6  7  8  9  10 11 12 13 14  -- 2nd Iter
 lw r1, 0(r2)                                     IF ID EX ME WB


# Loop C -- SOLUTION Part a
add r4, r3, r5     IF ID EX ME WB
addi r4, r4, -8       IF ID EX ME WB
LOOP: # Cycle      0  1  2  3  4  5  6  7  8  9  10 11 12 13  -- 1st Iter
 lw r1, 0(r2)          IF ID EX ME WB
 lw r10, 4(r2)            IF ID EX ME WB
 sw r1, 0(r3)                IF ID EX ME WB
 sw r10, 4(r3)                  IF ID EX ME WB
 addi r2, r2, 8                    IF ID EX ME WB
 bne r3, r4, LOOP                     IF ID EX ME WB
 addi r3, r3, 8                          IF ID EX ME WB
LOOP: # Cycle      0  1  2  3  4  5  6  7  8  9  10 11 12 13  -- 2nd Iter
 lw r1, 0(r2)                               IF ID EX ME WB
```

(*b*) Compute the rate that each loop copies data in units of bytes per cycle. Base this on your execution diagrams.

Based on the time that the first instruction of the loop, `lb`, is in `IF` one iteration of Loop A takes $10 - 1 = 9$ cycles. Each iteration copies one byte, so Loop A copies at a rate of $\frac{1}{9} \approx 0.111$ bytes per cycle.

An iteration of Loop B also takes 9 cycles, but it copies four bytes and so Loop B copies at a rate of $\frac{4}{9} \approx 0.444$ bytes per cycle, four times faster!

An iteration of Loop C takes $9 - 2 = 7$ cycles and copies 8 bytes, so its rate is $\frac{8}{7} \approx 1.143$ bytes per cycle, more than twice as fast as Loop B!

(*c*) Loop A has a wasted delay slot and should suffer stalls. Schedule the code (re-arrange instructions) to fill the delay slot and minimize the number of stalls. Feel free to change instructions and to add new ones, though the loop should still copy one byte per iteration and should copy the data as described above.

The solution appears below. The delay slot was filled by the store instruction, which uses a negative offset because `r3` is incremented before it executes. The increment of `r3` is put a the beginning so that its value will be ready when the `bne` is in `ID`.

```
# Loop A -- SOLUTION Part c
add r4, r3, r5     IF ID EX ME WB
LOOP: # Cycle      0  1  2  3  4  5  6  7  8  9
 addi r3, r3, 1       IF ID EX ME WB
 lb r1, 0(r2)            IF ID EX ME WB
 addi r2, r2, 1             IF ID EX ME WB
 bne r3, r4, LOOP              IF ID EX ME WB
 sb r1, -1(r3)                    IF ID EX ME WB
# Cycle            0  1  2  3  4  5  6  7  8  9
```

(*d*) Loop A can be safely substituted for Loop C. That is, if a program calls Loop C then that call can be changed to a call of Loop A or B and the program will still work correctly. However, if a program calls Loop A, substituting B will not work. Explain why and show sample values for `r2`, `r3`, and `r5` for which this is true.

The number of bytes copied by Loop B is a multiple of 4 and the starting address too must be a multiple of 4. Loop A has no such restrictions. Values that will work for A but not B are `r2=0x1001`, `r3=0x2002`, `r5=3`.

(*e*) If a program calls Loop B substituting C will not work. Explain why and show sample values for `r2`, `r3`, and `r5` for which this is true.

The number of values copied by Loop C must be a multiple of 8. Sample register values that work for B but not C are `r2=0x1000`, `r3=0x2000`, and `r5=4`.