Solution to

Computer Architecture HW #4

Due: Tuesday, Sept. 29 (5 PM in ITT 305 mailbox or under my office door, ITT 313)

1. In the following program segment, identify RAW (read-after-write), WAR (write-after-read), and WAW (write-after-write) dependencies.

```
I 1.   DIV   R8, R2, R1
I 2.   MUL   R6, R4, R8
I 3.   ADD   R2, R6, R7
I 4.   SUB   R8, R2, R4
```

RAW
R8 on I1 and I2
R6 on I2 and I3
R2 on I3 and I4

WAR
R2 on I1 and I3
R8 on I2 and I4

WAW
R8 on I1 and I4

2. On the next page, use Tomasula's algorithm assuming that a LD always takes 4 cycles to execute, ADDD takes 1 cycle to execute, and MULD takes 3 cycles to execute.

a) Trace the order of execution similar to what we did in class using numbers with circles around them to indicate when events happen. (Assume that the LD instruction gets the value 1.1 from memory to load into F4)

b) Rewrite the code segment to show the register renamings, i.e., rewrite the instructions with reservation station numbers (e.g., RS_8) replacing the floating point registers (e.g., F2).

Recall our assumption about the order of what happens in a clock cycle:

i) The next instruction from the Instruction Unit gets sent to the appropriate reservation station (if one is available)

ii) Any instruction in a reservation that has both of its operands can be issued if the corresponding functional unit is available (i.e., not being used by another instruction).

iii) A function unit that completes can send its result on the Common Data Bus (CDB). The result with be tagged with the reservation station number initiating the operation. All reservation stations and registers waiting to use the result will update their operands simultaneously. If multiple functional units complete in a clock cycle, assume only the "oldest" instruction will get to send it result.

Assuming single Adder