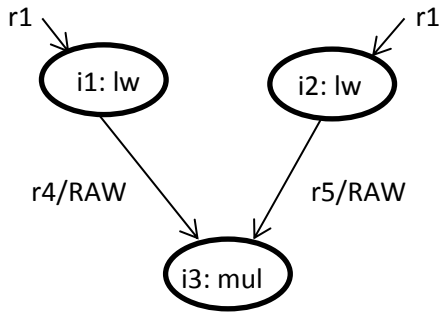
CPSC 3300 – Spring 2017 – Homework 6
Due at class time on Tuesday, April 4

1. For the MIPS instruction sequence below, complete the data dependency diagram. (Destination register is listed first except for sw instruction; sw writes into memory rather than a register.) (35 pts.)

```
i1: lw   r4, 0( r1 )  // reg[4] ← memory[ reg[1] + 0 ]
i2: lw   r5, 4( r1 )  // reg[5] ← memory[ reg[1] + 4 ]
i3: mul  r6, r4, r5   // reg[6] ← reg[4] * reg[5]
i4: sub  r8, r6, r7   // reg[8] ← reg[6] - reg[7]
i5: sw   r8, 8( r1 )  // memory[ reg[1] + 8 ] ← reg[8]
i6: add  r1, r1, r2   // reg[1] ← reg[1] + reg[2]
```



2. Draw the dependency diagram for the following MIPS code (35 pts.)

```
i1: add  r3, r4, r6    // reg[3] ← reg[4] + reg[6]
i2: sub  r5, r3, r2    // reg[5] ← reg[3] - reg[2]
i3: lw   r7, 0( r5 )   // reg[7] ← memory[ reg[5] + 0 ]
i4: add  r5, r5, r8    // reg[5] ← reg[5] + reg[8]
i5: sub  r9, r5, r7    // reg[9] ← reg[5] - reg[7]
```

3. Using the Transmeta Crusoe 4-way VLIW format in the notes (one load/store slot; two ALU slots with the second slot capable of integer, shift, and floating-point/MMX operations; and, one branch or immediate value slot) and the same pipeline stage and memory load timing as the 5-stage pipeline with forwarding, show the VLIW instructions needed for the code sequence in question 2 above. You can rename registers as needed to remove false dependencies and increase parallel execution. Note that the Crusoe has a register scoreboard so that a VLIW instruction of all nops is not required for correct timing. (30 pts.)