CprE 381 Homework 7

1. **Pipeline Simulation (50 points)**

For each of the modules from Figure 4.51 (page 304 of P&H) that are listed in the table below, specify what the inputs and outputs are in each cycle for the following code. You do not need to specify values for those modules, inputs, or outputs not listed in the table. Manually simulate (i.e., fill out the table) until the **sw** has completed (i.e., left the write-back stage). *[Hint: the table already has the first couple of cycles filled out. If a value depends on an instruction before or after the code below, report it as X.]*

Cycle	Instruction Memory		Register File				ALU				MemtoReg MUX			PCSrc MUX		
	Addr	Instr	Read reg 1	Read data 1	Write reg	Write data	A	B	Op (e.g, add, sub)	ALU result	1	0	s	1	0	s
1	0x00000010	lui …	X	X	X	X	X	X	X	X	X	X	X	0xXXXXXXXX	0x00000014	0
2	0x00000014	addi …	X	X	X	X	X	X	X	X	X	X	X	0xXXXXXXXX	0x00000018	0
3	0x00000018	sub…	0x00	0x00000000	X	X	X	0x00001001	lui	0x10010000	X	X	X	0xXXXXXXXX	0x0000001c	0

[Pipelined MIPS – Simulation Table]

```
# Assume that $a0 = 3, $a1 = 1024, $a2 = 1023, $a3 = -1
# at the start of your manual simulation.
# Assume that lui is supported by the lui operation in
# the ALU and that the value shifted for lui is the B
# input of the ALU (note that this is likely different
# than your project implementation and that's OK).
# The following instructions start at address 0x00000010:
lui  $s0, 0x1001
addi $t0, $zero, 42
sub  $t1, $a0, $a1
xor  $t2, $a2, $a3
ori  $s0, $s0, 0x0040
beq  $t0, $a3, Exit
addi $t4, $zero, 0
sll  $zero, $zero, 0
sw   $t0, 0($s0)
...
Exit: # This label resolves to address 0x00000100.
```

2. **Data Dependencies (30 points)**

Here is a short MIPS assembly language loop. (This is a simpler version of a very common operation in scientific applications.) Assume that we run this code on the 5-stage pipelined datapath.

```
    Frodo:
          lw   $t0, 0($a1)
          add  $t1, $t0, $a2
          lw   $t0, 0($a0)
          add  $t0, $t0, $t1
          sw   $t0, 0($a0)
          sub  $a3, $a3, 1
          addi $a0, $a0, 4
          addi $a1, $a1, 4
          bne  $a3, $0, Frodo
```

Find all data dependencies and RAW hazards in the code. How many stalls do you need to insert to solve RAW hazards?

3. **Exam Question (20 points)**

Develop your own exam question (roughly 10-15 points) from MIPS arithmetic, single-cycle processor design, performance analysis, pipelining, or data hazards. Your question shouldn't simply ask students to recall information but should ask for an application of a concept or require understanding of a concept or need analysis of a processor/application. You must include a correct and complete solution to your question. This question should be your own work and not copied from a book or an old exam. **Post your question and solution to the HW_Help channel for others to use. Please include the screenshot of your question in your HW solutions report to get full credits.**