LSU EE 4720 Homework 5 Due: 4 April 2003

Problem 1: [Easy] Complete pipeline execution diagrams for the following code fragments running on the fully bypassed MIPS implementations with floating point units as described below.

```
# One ADD unit, latency 3, initiation interval 1.
add.d f0, f2, f4
sub.d f6, f0, f8
add.d f8, f10, f12

# One ADD unit, latency 3, initiation interval 2.
add.d f0, f2, f4
sub.d f6, f0, f8
add.d f8, f10, f12

# Two ADD units (A and B), latency 3, initiation interval 4.
add.d f0, f2, f4
sub.d f6, f0, f8
add.d f8, f10, f12
```

Problem 2: [Easy] Choose the latency and initiation interval for the add and multiply functional units so that the two instructions stall to avoid a structural hazard. Show a pipeline execution diagram consistent with them. (The easy way to solve it is to do the PED first, then figure out the latency and initiation interval.)

```
mul.d f0, f2, f4
add.d f6, f8, f10
```

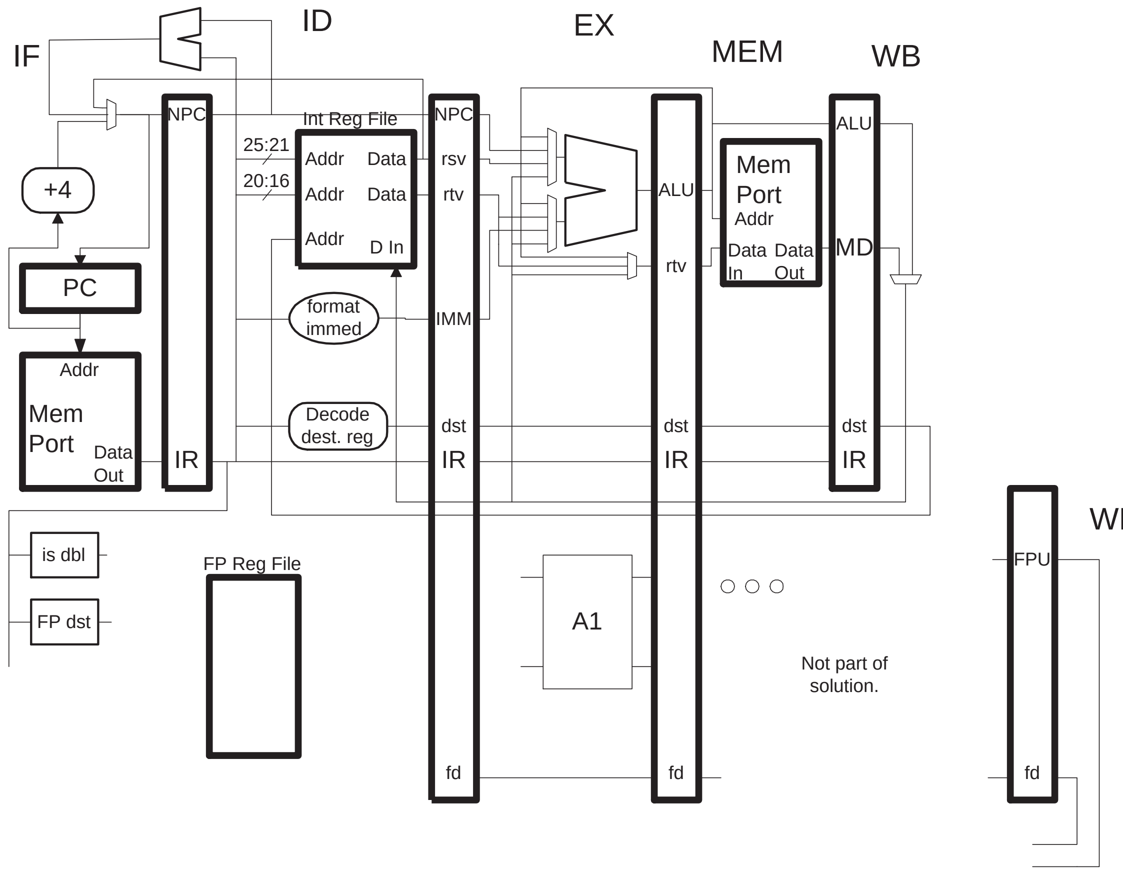
Problem 3: The two PEDs below show execution of MIPS code produce that produces wrong answers. For each explain why and show a PED of correct execution.

```
# PED showing a DESIGN FLAW. (The code runs incorrectly.)
add.s f1, f10, f11  IF ID A1 A2 A3 A4 WF
sub.d f2, f0, f4       IF ID A1 A2 A3 A4 WF

# PED showing a DESIGN FLAW. (The code runs incorrectly.)
mul.d f0, f2, f4    IF ID M1 M2 M3 M4 M5 M6 WF
sub.s f1, f10, f11     IF ID A1 A2 A3 A4 WF
```

Problem 4: As described below design the logic for the floating-point register file in the MIPS implementation below. The FP portion shows only part of add functional unit. Assume that is the only functional unit.

- Describe how the FP register file works. For reference, here is a description of the integer register file: The integer register file has two read ports and a write port. Each read port has a five-bit address input and a 32-bit data output. The write port has a five-bit address input and a 32-bit data input. Reads from zero retrieve 0, writes to zero have no effect.
- The following signals are available: $\boxed{\text{is dbl}}$; if 1 the instruction uses double-precision operands, otherwise single-precision. $\boxed{\text{FP dst}}$: if 1 the instruction writes the floating-point register file, otherwise it does not (possibly because it's not a floating-point instruction).
- Show all connections to the FP register file. Show the number of bits or the bit range for each connection.
- The WF stage provides two signals, FPU (the value to write back) and fd, something generated in ID (as part of the solution). Additional signals can be sent down the pipeline.
- Keep In Mind: The hardware should work for both single and double operands. (That's what makes the problem interesting. If you're confused first solve it assuming only double operands, then attempt the full problem.)
- Make sure the fragments from the previous problem would run correctly.


IF
ID
EX
MEM
WB
WF
+4
PC
Addr
Mem Port
Data Out
NPC
IR
Int Reg File
25:21
20:16
Addr
Addr
Addr
Data
Data
D In
format immed
Decode dest. reg
rsv
rtv
IMM
dst
fd
ALU
rtv
Mem Port
Addr
Data In
Data Out
MD
is dbl
FP dst
FP Reg File
A1
Not part of solution.
FPU