ECE 429 / 629 Homework #6

1. Suppose we have a classic RISC five-stage integer pipeline with the MIPS FP pipeline as described in Section A.5 of the text. Assume all memory access take 1 clock cycle, and that a register read/write in the same clock cycle "forwards" through the register file. Assume that branches resolve in the ID stage. Show the timing diagram (using F,D,X,M,W) and compute the total number of cycles to execute the program below:

```
Loop:     L.D       F0, 0(R2)
          L.D       F4, 0(R3)
          MUL.D     F0, F0, F4
          ADD.D     F2, F0, F2
          ADDUI     R2, R2, 8
          ADDUI     R3, R3, 8
          SUBU      R5, R4, R2
          BNEZ      R5, Loop
```

   a. without any forwarding hardware (including forwarding to the branch equality component in the ID stage); branches are handled by flushing the pipeline
   b. with normal forwarding hardware; branches are handled by predicting as not taken

   The initial value of R4 is R2 + 792.

2. Using the MIPS computer in Figure 6.65 (handout), determine the values of all the control lines, as well as the value of the EX/MEM pipeline register, for the first seven clock cycles of the program below. Use the diagram provided on the next page of this homework.

```
Loop:     LW   R1, 0 (R4)
          ADD  R2, R1, R1
          SW   R2, 0 (R4)
          ADDI R3, R3, -1
          BNE  R3, R0, Loop
```

   The initial value for R3 is 99, and the initial value for the memory location to which R4 points is 3. Be sure to clearly indicate the number of bits in each control line, as well as the number of bits in each division of the pipeline register. Use hexadecimal for the register, X for "don't care", and leave blank any cells in the table for which there is not enough information.

Problem #1 adapted from Hennessy and Patterson, *Computer Architecture: A Quantitative Approach*, 3rd edition, Morgan Kaufmann, 2003. (Question A.2)

		cycle 1	cycle 2	cycle 3	cycle 4	cycle 5	cycle 6	cycle 7
Control	PCSrc							
	Branch							
	IF.Flush							
	EX.Flush							
	ID.Flush							
Hazard Detection Unit	IF/IDWrite							
	PCWrite							
	ID.Flush2							
Forwarding Unit	ForwardA							
	ForwardB							
	ALUSrc							
	ALUOp							
	RegDst							
	MemRead							
	MemWrite							
	MemtoReg							
EX/MEM register	M							
	ALUOut							
	data							
	regdst							

Note: The control lines above are grouped according to the component responsible for setting them. The lines are listed above in clockwise order around each component, starting at the bottom of the component. Similarly, the control lines emanating from the pipeline registers, as well as the divisions of these registers, are written in a top-down manner according to the figure. This unambiguous convention should make it possible to determine uniquely all control lines and register divisions, even though some are not labeled in the figure.

Also, assume that a value of zero selects the top incoming line of each Mux, with the lower lines selected by increasing values from top-to-bottom. (As an example, for a Mux with 3 inputs: 0 selects the top line, 1 selects the middle line, and 2 selects the bottom line.)