On the following sheets, demonstrate that the final pipelined DLX implementation with delayed branches, forwarding, etc. does in each clock step when it executes the program of PH chapter 3 problem 1 as modified below through the last clock step used by the second execution of a LW instruction.

\[
\begin{align*}
\text{LW} & \quad \text{R1,0(R2)} \\
\text{loop:} & \\
\text{ADDI} & \quad \text{R1,R1,#1} \\
\text{SW} & \quad \text{R1,0(R2)} \\
\text{ADDI} & \quad \text{R2,R2,#4} \\
\text{SUB} & \quad \text{R4,R3,R2} \\
\text{BNZ} & \quad \text{R4,loop} \\
\text{LW} & \quad \text{R1,0(R2)}
\end{align*}
\]

Notice that the LW instruction was replicated into the delay slot of the BNZ instruction.

Make reasonable assumptions about the numeric values in the registers (such as PC=0 or PC=0x1000) so you can write numeric data into the registers on the following sheets. The first state you show should have this initial PC value in PC and all the other pipeline registers with UNDEFINED values.

Show in each register box the new value which is stored in it after the last clock tick. Thus, the second state should show PC containing 4 or 0x1004, and IF/ID/IR containing LW R1,0(R2). (You do not have to write the numeric values of instructions in the IR.)

Whenever a data memory or register value changes, write an explanation of the change (address or register number and the new value) in the data memory or register file box for the snapshot after the clock tick when the change takes effect.
Memory (cache and its controller)

PC

Instruction Memory (cache and its controller)

IR

NPC

Register File (31x32)

4

Add

 IF/ID

ID/EX

EX/MEM

MEM/WB

ALUOutput

ALUOutput

ALUOutput

Data Memory (cache etc)

LMD

LMD

LMD