ECE 30
Introduction to Computer Engineering

Study Problems, Set #7
Spring 2014

1. Assuming the instruction mix shown below for a *multi-cycle* implementation, what is the average CPI? If the clock period of an alternative *single-cycle* implementation is equal to the time it takes to execute the longest instruction, how much faster is the *multi-cycle* implementation?

Instruction	Frequency	No. of clock cycles
arithmetic / logical	40%	4
lw	30%	5
sw	15%	4
beq	10%	3
j	5%	3

Solution:

$\mathsf{CPI}_{\text{multi}} = 0.4 \times 4 + 0.3 \times 5 + 0.15 \times 4 + 0.1 \times 3 + 0.05 \times 3 = 4.15.$

Let the clock period of the multi-cycle implementation be T. Then, the clock period of the single-cycle implementation is $5T$ because `lw` (which is the instruction that takes the maximum number of clock cycles to execute) takes 5 clock cycles. Let I be the number of instructions.

$\mathsf{ExecutionTime}_{\text{single}} = \mathsf{CPI}_{\text{single}} \times I \times 5T = 5IT.$
$\mathsf{ExecutionTime}_{\text{multi}} = \mathsf{CPI}_{\text{multi}} \times I \times T = 4.15IT.$

Speed Up $= 5/4.15 = 1.2.$

2. Briefly describe two benefits of a multi-cycle datapath implementation over a single-cycle implementation.

   Solution:

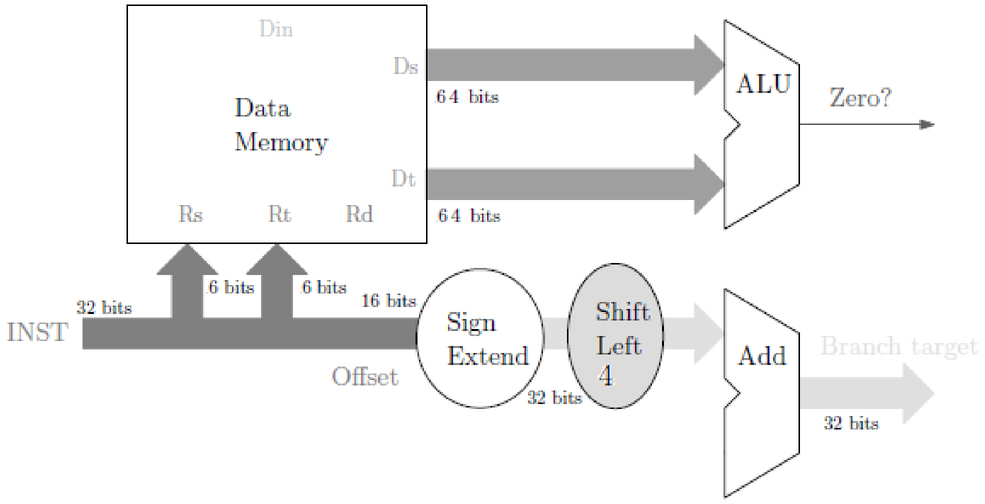
   (a) Resource Sharing:

   We can use the same resource in multiple cycles. For example, a single ALU can be used for computing the new PC value as well as for data computation, by scheduling these operations in different clock cycles. This avoids the need to provide two separate adders for PC computation in a multi-cycle implementation (which was required for a single-cycle implementation since all adders were used simultaneously in the same single clock cycle that executed an instruction).

   (b) Time Saving on Short Instructions:

   A single-cycle implementation requires the clock period to be as long as the longest instruction, i.e., the instruction that causes the longest delay to complete its execution. This results in the datapath "doing nothing" (called "idle time") for a significant fraction of the clock period for short, efficient instructions which take only a fraction of a clock cycle to execute. On the other hand, a multi-cycle implementation spreads the execution of a single instruction over multiple clock cycles, by breaking each instruction into different functional steps and executing each step in one clock cycle. So, efficient instructions will take fewer clock cycles and more complex instructions will take more clock cycles. Once a short, efficient instruction has executed its few steps in a few clock cycles, the first step of the next instruction can start execution immediately after, in the next clock cycle, thus minimizing the "idle time" of the datapath.

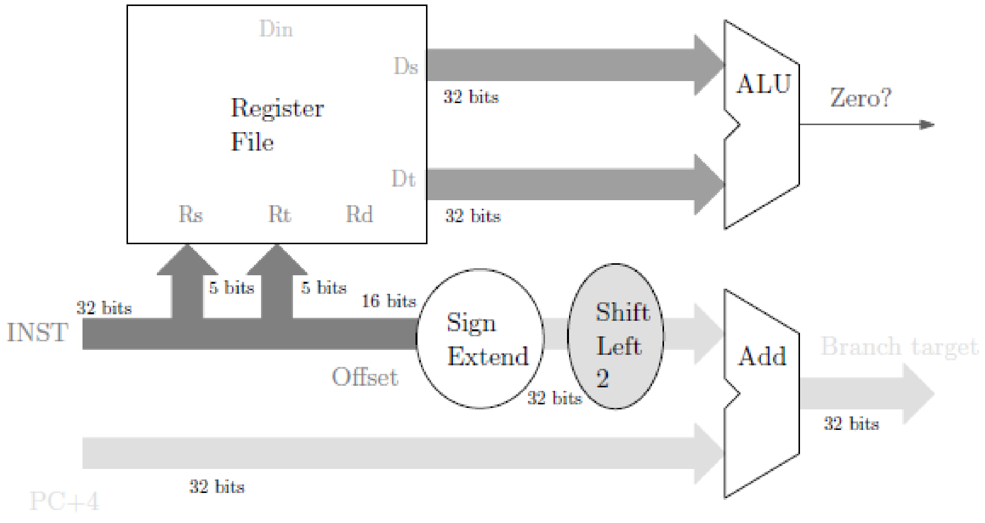
3. The diagram below shows the datapath for a single-cycle implementation of the `beq` instruction. Find the errors and the missing items/connections.



Incorrect and Incomplete Datapath for beq

Solution:

(a) The block on the left hand side is "Register File" instead of "Data Memory".
(b) The inputs Rs and Rt to the Register File are 5 bits wide instead of 6 bits.
(c) The ALU inputs are 32 bits wide instead of 64 bits.
(d) The offset is "Shifted Left" over 2 rather than 4 positions.
(e) The adder in the bottom right is missing a 2nd input, i.e., "PC + 4 (32 bits)".



Correct Datapath for beq