CSCI 2400: Computer Systems
Recitation Exercise 4
(Dated: November 3, 2014)

1. Look at the assembly instructions below. These instructions will be pipelined to speed up the code execution. Unfortunately the code cannot be pipelined as is due to data dependencies. Add 'NOP' instruction in the code below at the appropriate locations to ensure that pipelining can be accomplished.

```
mov $10, %eax
add $2, %eax
mov $4, %ebx
mov $5, %ecx
add $1, %ebx
add $1, %ecx
add %ecx, %eax
add %ebx, %eax
```

2. Consider the example of an unpipelined processor shown in Figure 1. We assume that the combinational logic requires 300 ps (picoseconds) to process any computation and the loading of the results in the register requires 20 ps. The maximum rate at which we could operate in this system, also called as the throughput, is given by the following formula:

$$Throughput = \frac{1 \text{ instruction}}{(20 + 300) \text{ picosecond}} \frac{1000 \text{ picosecond}}{1 \text{ nanosecond}} \approx 3.12 \text{ GIPS} \ , \tag{1}$$

where GIPS stands for giga-instructions per second. The total time required to perform a single instruction from beginning to end is known as the latency. In this system, the latency is 320 ps. Thus, *Throughput* (in GIPS) = $\frac{1}{latency \text{ (in picoseconds)}} \times 1000$. Also note that each instruction I1, I2 and I3 thus requires *latency* ps to process completely.

Suppose we divide the computation performed by our system in three stages A, B and C (see Fig. 2), where each requires $300/3 = 100$ ps. Then we could put pipeline registers between the stages so that each instruction moves through the system in three steps, requiring three complete clock cycles from beginning to end. In this system, we could cycle the clocks every $100 + 20 = 120$ picoseconds, giving a throughput of around 8.33 GIPS. Since processing a single instruction requires 3 clock cycles, the latency of this pipeline is $3 \times 120 = 360$ ps.

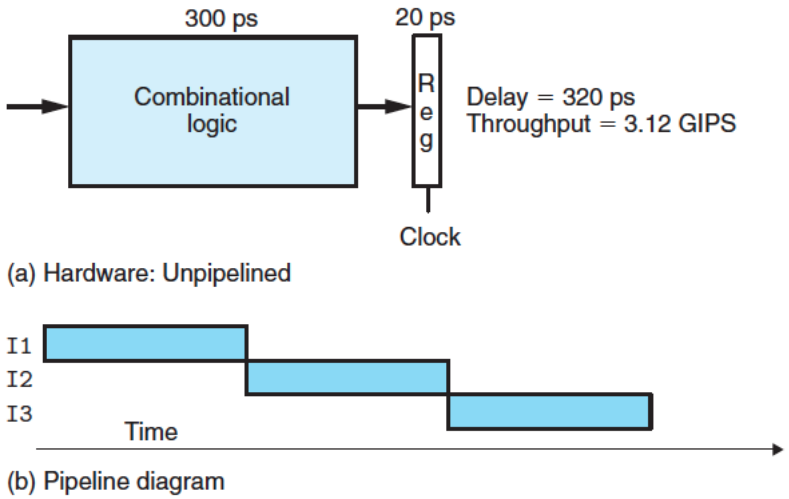

FIG. 1: Shows an unpipelined system. (Figure taken from Bryan & O'Hallaron textbook.)

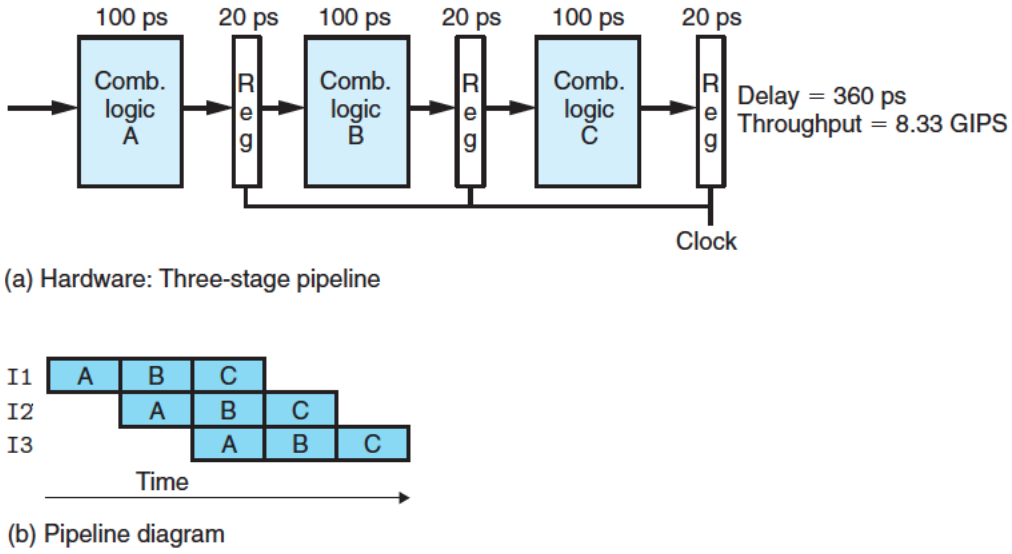

FIG. 2: Shows a three-stage pipelined system. (Figure taken from Bryan & O'Hallaron textbook.)

We have increased the throughput of the system by a factor of $8.33/3.12 = 2.67$ at the expense of some added hardware and a slight increase in the latency ($360/320 = 1.12$).

(a) Considering the unpipelined system of Figure 1, calculate the throughput and latency for a 6-stage pipeline. Assume that each stage of the combinational logic requires the same computation time and that the overall computation time (in this case 300 ps) gets distributed equally for all the 6 stages. Also, assume the overhead for each register to be 20 ps.

(b) Consider an unpipelined system requiring the total computational time for the combinational logic to be C ps and the overhead for each register to be R ps. Derive the formulae for throughput and latency for after converting the unpipelined system into N-stage pipeline. Assume that each stage of the combinational logic requires the same computation time and that the overall computation (in this case C) gets distributed equally for all the N stages.

(c) For the N-stage pipelined system of part 2b, derive the condition for N such that the delay caused **solely** by the overall register overhead exceeds the total computational time required by the N stages of the combinational logic.

3. Solve problem 5.15 from Chapter 5 in the CSAPP textbook.