Problem 6: Hyperblock (20 pts)

As described in class, Hyperblock scheduling uses predication support to replace unbiased branches with predicates, which enables larger code blocks.

A) [2 pts] In one sentence, in terms of code optimizations, explain what benefit does larger scheduling code blocks provide?

Larger scheduling code blocks enable greater flexibility for instruction scheduling.

One optimization that can be applied to Hyperblock is **Instruction Promotion**. Instruction Promotion hoists the operation from a predicated instruction and replaces the original predicated instruction with a conditional move. With Instruction Promotion, operations can be scheduled and issued before their corresponding predicates are determined. Below shows an example of Instruction Promotion.

Before:

```
         cmplt B6,B7-> B0
[B0]     ld MEM1-> A5
[!B0]    ld MEM2-> A5
         nop 4
         addi A5,8 -> A8
```

After Instruction Promotion:

```
         ld MEM1-> A5
         ld MEM2-> A6
         cmplt B6,B7-> B0
[!B0]    mv A6-> A5
         nop 4
         addi A5,8 -> A8
```

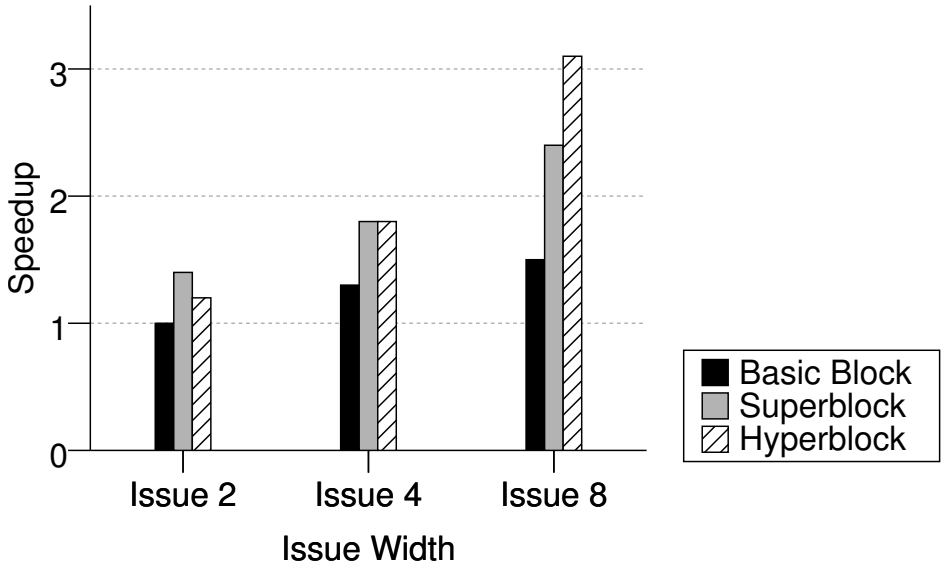
Assume we run this code on a processor that supports predication but can only issue a predicated instruction after its corresponding predicate has been resolved.

B) [4 pts] For the example above, can Instruction Promotion ever improve system performance? Why or why not?

Yes it can. With Instruction Promotion, the program can hide some of the load latency.

C) [4 pts] For the example above, can Instruction Promotion ever degrade system performance? Why or why not?

Yes it can. Instruction Promotion: (1) introduces extra instructions and (2) can increase register pressure. [Note that extra instructions may not always increase register pressure]



D) [10 pts] The graph above shows the performance comparison of a program optimized using Hyperblock and Superblock respectively with different issue widths. With all other factors being equal, as the figure shows, when the issue width is low, Superblock provides higher speedup than Hyperblock. However, when the issue width is high, Hyperblock provides higher speedup than Superblock. Explain why this can happen?

A wider issue width can tolerate the wasted instructions in a hyperblock, but does not benefit the superblock (all else being equal).

A more detailed explanation: Hyperblock uses predication which increases the total number of instructions to execute. When the issue width is low, executing extra predicated instructions requires extra work, which slows down the processor as all resources of the processor has already been fully utilized. When the issue width is high, however, Hyperblock provides a greater number of independent instructions from the multiple paths of control to fill the available processor resources. As Hyperblock also enables larger code blocks for better optimization for unbiased branches, Hyperblock provides better speedup.