Problem 1: Unlike MIPS, PA-RISC 2.0 has a post-increment load and a load using scaled-index addressing. The code fragments below are from the solution to Problem 2 in the midterm exam the fragments show several MIPS instructions under “Combine” and a new instruction under “Into.” For each “Into” instruction show the closest equivalent PA-RISC instructions and show the coding of the PA-RISC instruction. (See the references page for information on PA-RISC 2.0)

(The term offset used in the PA-RISC manual is equivalent to the term effective address used in class, and is not to be confused with offset as used in this class. Assume that the s field and cc fields in the PA-RISC format are zero.)

Show all the fields in the format, including their names and their values.

Combine:
\[
\text{lbu } $t1, 0($t0) \\
\text{addi } $t0, $t0, 1
\]
Into:
\[
\text{lbu.ai } $t1, 0($t0)+ \quad \# \text{Post increment load.}
\]
Combine:
\[
\text{sll } $t1, $t1, 2 \\
\text{add } $t3, $a1, $t1 \\
\text{lw } $t4, 0($t3)
\]
Into:
\[
\text{lw.si } $t4, (0($t3),$t1) \quad \# \text{Scaled index addressing.}
\]
**Problem 2:** The code fragment below runs on the implementation illustrated below.

(a) Show a pipeline execution diagram for the code fragment on the implementation up to the second fetch of the `sub` instruction; assume the branch will be taken.

(b) Show the value of the labeled wires (A, B, and C) at each cycle in which a value can be determined.

For maximum pedagogical benefit please pay close attention to the following:

- As always, look for dependencies.
- Pay attention to the RAW hazard between `sub` and `sw` and the RAW hazard between `andi` and `bne`.
- Make sure that `add` is fetched in the right time in the second iteration.
- Base timing **on the implementation diagram**, not on rules inferred from past solutions.

```
LOOP:
    add r1, r2, r3
    sub r3, r1, r4
    sw r3, 0(r5)
    andi r6, r3, 0x7
    bne r6, $0, LOOP
    addi r2, r2, 0x8
```
**Problem 3:** Consider the implementation from the previous problem, repeated below.

(a) There is a subtle reason why the implementation cannot execute a `jr` instruction. What is it? Modify the hardware to correct the problem.

(b) There are two reasons why it cannot execute a `jalr` instruction, one given in the previous part and a second reason. What is it? Modify the hardware to correct the problem.