Question 2 (14 points): Two versions of a function that changes the endianess of the elements of an array stored in memory are shown below. Assume that this function is executed in a processor where the ALU instructions (`slli`, `add`, `srli`, `or`, `addi`) take, on average, one clock cycle; control-flow instructions (`bnez`, `jalr`, `blt`, `jalr`) take, on average, three clock cycles; and memory instructions (`lw`, `sw`, `lbu`, `sb`) take, on average, five clock cycles to execute.

```
241 ChangeEndianess_A:
242     slli t0, a1, 2        # t0 <- 4*n
243     add  t1, a0, t0       # t1 <- &(A[n])
244     bnez a1, nextWord
245     jalr zero, ra, 0
246 nextWord:
247     lw   t2, 0(t0)        # t2 <- AAAAAAAA BBBBBBBB CCCCCCCC DDDDDDDD
248     slli t3, t2, 24       # t3 <- DDDDDDDD 00000000 00000000 00000000
249     srli t4, t2, 8        # t4 <- 00000000 AAAAAAAA BBBBBBBB CCCCCCCC
250     slli t4, t4, 24       # t4 <- CCCCCCCC 00000000 00000000 00000000
251     srli t4, t4, 8        # t4 <- 00000000 CCCCCCCC 00000000 00000000
252     srli t5, t2, 16       # t5 <- 00000000 00000000 AAAAAAAA BBBBBBBB
253     slli t5, t5, 24       # t5 <- BBBBBBBB 00000000 00000000 00000000
254     srli t5, t5, 16       # t5 <- 00000000 00000000 BBBBBBBB 00000000
255     srli t6, t2, 24       # t6 <- 00000000 00000000 00000000 AAAAAAAA
256     or   t3, t3, t4       # t3 <- DDDDDDDD CCCCCCCC 00000000 00000000
257     or   t5, t5, t6       # t5 <- 00000000 00000000 BBBBBBBB AAAAAAAA
258     or   t3, t3, t5       # t3 <- DDDDDDDD CCCCCCCC BBBBBBBB AAAAAAAA
259     sw   t3, 0(t0)        #
260     addi t0, t0, 4        # p++
261     blt  t0, t1, nextWord
262     jalr zero, ra, 0
```

```
265 ChangeEndianess_B:
266     mv   t5, zero         # i <- 0
267     bnez a1, nextWordB
268     jalr zero, ra, 0
269 nextWordB:
270     lbu  t1, 0(t0)        # t1 <- 00000000 00000
271     sb   t1, 3(t0)
272     lbu  t2, 1(t0)        # t1 <- 00000000 00000
273     sb   t2, 2(t0)
274     lbu  t3, 2(t0)        # t1 <- 00000000 00000
275     sb   t3, 1(t0)
276     lbu  t4, 3(t0)        # t1 <- 00000000 00000
277     sb   t4, 0(t0)
278     addi t5, t5, 1
279     blt  t5, a1, nextWordB
280     jalr zero, ra, 0
```

a. (**3 points**) If the value of `a1` is zero, what is the CPI for an execution of `ChangeEndianess_A`?

Only the instructions in lines 242-245 will execute. Thus, the CPI for this execution is:

$$\text{CPI}_{\text{a1=0}} = \frac{1+1+3+3}{4} = 2\frac{\text{clocks}}{\text{instruction}} \tag{1}$$

b. (**3 points**) What is the CPI for an execution of `ChangeEndianess_A` in which `a1` is very large?

If `a1` is very large, then only the instructions inside the loop matter:

$$\text{CPI}_{\text{a1 large}} = \frac{2\times 5+12\times 1+3}{15} = 1.67\frac{\text{clocks}}{\text{instruction}} \tag{2}$$

c. (**4 points**) An execution of `ChangeEndianess_A` in which the value of `a1` is `0x00010000` runs in a 4 GHz RISC-V processor. How long does this execution of `ChangeEndianess_A` takes?

`0x00010000` $= 2^{16} = 2^6 \times 2^{10} = 64 \times 1024$

$$\begin{aligned}\text{Execution Time} &= \frac{64\times 1024\times(2\times 5+12\times 1+3)}{4\times 10^9}\\ &= \frac{64\times 1024\times 25}{4\times 10^9} = 16\times 1024\times 25\times 10^{-9}\\ &= 409600\times 10^{-9} = 409.6\times 10^{-6}\ \text{seconds} = 409.6\ \mu s\end{aligned}$$

d. (**4 points**) An execution of `ChangeEndianess_A` and an execution of `ChangeEndianess_B` receive the same very large value in `a1`. Both functions run on the same computer with the same clock frequency. Which version is faster and by how much?

Assume that the number of iterations in the loop for both executions is K and that the clock frequency for this computer is F

$$\begin{aligned} \text{Time}_A &= (2 \times 5 + 12 \times 1 + 3) \times \frac{K}{F} = 25 \times \frac{K}{F} \\ \text{Time}_B &= (8 \times 5 + 1 + 3) \times \frac{K}{F} = 44 \times \frac{K}{F} \\ \text{Speedup} &= \frac{44}{25} = 1.76 \end{aligned}$$

`ChangeEndianess_A` is 1.76 times faster than `ChangeEndianess_B`.