Question 5 (25 points): The main competitor to *TinyProc Inc.* is *Sneak Inc.* and they are trying to gain market share with their `S75` processor. `S75` uses the same Instruction Set Architecture as the TP500 and it appears to be an improved copy of the TP500. *TinyProc Inc.* has managed to gain access to an `S75`. The most important application in this market is `ControLux`, an application that controls the dash board of a luxury car. The compiler developed by *TinyProc Inc.* is *TinyComp* and the code generated by *TinyComp* can run both on TP500 and on `S75`. *Sneak Inc.* have developed their own compiler called *SneakComp* which generates code that can also run on both the TP500 and the `S75`. A performance comparison between TP500 and `S75` reveals that `S75` is twice as fast as TP500 when running `ControLux`. Management is very worried and has asked you, as the Industrial Internship Program (IIP) student with a placement at *TinyProc Inc.*, to investigate what *Sneak Inc.* has done to deliver this performance gain. Here is the information that you have to work with:

- `ControLux` compiled with *SneakComp* runs twice as fast in `S75` compared with `ControLux` compiled with *TinyComp* running on TP500.
- `S75` runs at 750 MHz while TP500 runs at 500 MHz.
- The instructions in the ISA of TP500 can be divided into three classes according to the number of clocks that each instruction needs to execute: ALU instructions, load/store instructions and branch instructions, When `ControLux` is compiled with *TinyComp*, its instruction mix is 40% ALU, 25% load/stores, and 35% branches.
- In TP500, the ALU instructions require 1 clock cycle to execute, the load/store instructions require 5 clock cycles, and the branches require 3 cycles.
- You have been able to measure the number of clocks per instruction for ALU and branch instructions, but not for load/store instructions, in the `S75`. On average it takes 1 clock cycle to execute an ALU instruction and 3 clock cycles to execute a branch in the S75. To measure the number of clock cycles per load/store instruction in the `S75` you run the `ControLux` code generated by the *TinyComp* on the `S75`. You find that the `S75` is 2.025 times faster than the `TP500` when executing this code.

1. (**6 points**) What is the CPI of `ControLux` compiled with *TinyComp* running on TP500?

2. (**7 points**) What is the CPI of `ControLux` compiled with *TinyComp* when running in the `S75`?

3. (**7 points**) What is the average number of clock cycles required to execute load/store instructions when executing the `ControLux` code generated with *TinyComp* on the `S75`?

4. (**5 points**) *TinyProc Inc.* has managed to obtain an executable for `ControLux` that was generated by *SneakComp*. Using hardware event counters you managed to determine that in their code both the number of branch and the number of load/store instructions were reduced by 20% in comparison with the code gerated with *TinyComp*. What is the percentage change in the number of ALU instructions executed by this version of `ControLux` in comparison with the code generated with *TinyComp*?