►Solution◄

Question 1: (20 points)

The main competitor of *TinyProc Inc.* is *Sneak Inc.* The latter is trying to increase their market share by offering their latest processor `S75`. Despite using the same Instruction Set Architecture, `S75` is an improvement over `TP500` from *TinyProc Inc.* The most important application in this market is `ControLux`, an application that controls the dashboard of a luxury car. After getting access to `S75`, *TinyProc Inc.* was able to design *TinyComp*, a compiler that generates code that runs on both `TP500` and `S75`. Trying to preserve their market share, *Sneak Inc.* developed their own compiler *SneakComp*, which also generates code that runs on both processors. An independent performance comparison between `TP500` and `S75` reveals that `S75` is twice as fast as `TP500` when running `ControLux`. The *TinyProc Inc.* management is panicking and has asked you, as an 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 cycles 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`, an ALU instruction requires 1 cycle to execute, a load/store instruction requires 5 cycles, and a branch requires 3 cycles.
- In `S75`, you were able to measure the number of cycles per instruction for ALU and branch instructions, but not for load/store instructions. On average, it takes 1 cycle to execute an ALU instruction and 3 cycles to execute a branch. To measure the number of cycles per load/store instruction, you ran the `ControLux` code generated by *TinyComp* on the `S75`. You found out that `S75` is 2.025 times faster than `TP500` when executing this code.

a. (5 points) What is the CPI of `ControLux`, compiled with *TinyComp*, when it runs on `TP500`?

Solution:

$$\begin{aligned} CPI_{\texttt{TP500}} &= 0.4 \times 1 + 0.25 \times 5 + 0.35 \times 3 \\ &= 0.4 + 1.25 + 1.05 \\ &= 2.7 \text{ cycles per instruction} \end{aligned}$$

b. (5 points) What is the CPI of `ControLux`, compiled with *TinyComp*, when it runs on `S75`?

Solution: The code is the same for both executions, therefore the number of instructions executed, N, is also the same.

$$\begin{aligned} Time_{\texttt{TP500}} &= 2.025 \times Time_{\texttt{S75}} \\ \frac{N \times \text{CPI}_{\texttt{TP500}}}{Freq_{\texttt{TP500}}} &= 2.025 \times \frac{N \times \text{CPI}_{\texttt{S75}}}{Freq_{\texttt{S75}}} \\ \frac{2.7}{500\ MHz} &= 2.025 \times \frac{\text{CPI}_{\texttt{S75}}}{750\ MHz} \\ \text{CPI}_{\texttt{S75}} &= \frac{750 \times 2.7}{500 \times 2.025} \\ \text{CPI}_{\texttt{S75}} &= 2 \text{ cycles per instruction} \end{aligned}$$

c. (5 points) What is the average number of cycles required to execute load/store instructions when executing the `ControLux` code generated with *TinyComp* on the `S75`?

Solution:

$$\begin{aligned} \text{CPI}_{\texttt{S75}} &= 0.4 \times 1 + 0.25 \times \text{CPI}_{load/store} + 0.35 \times 3 \\ 2 &= 0.4 + 0.25 \times \text{CPI}_{load/store} + 1.05 \\ \text{CPI}_{load/store} &= \frac{2.0 - 1.45}{0.25} \\ \text{CPI}_{load/store} &= 2.2 \text{ cycles per instruction} \end{aligned}$$

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

Solution: If the code generated by *TinyComp* executes 1000 instructions, then it executes 400 ALU instructions, 250 loads/stores, and 350 branches. Therefore, the code generated by *SneakComp* executes 200 loads/stores and 280 branches.

$$Time_{\texttt{TP500}} = \frac{I_{\texttt{TP500}} \times \text{CPI}_{\texttt{TP500}}}{Freq_{\texttt{TP500}}}$$

$$Time_{\texttt{TP500}} = \frac{1,000 \times 2.7}{500\ MHz}$$
$$Time_{\texttt{S75}} = \frac{I_{\texttt{S75}} \times \text{CPI}_{\texttt{S75}}}{Freq_{\texttt{S75}}}$$
$$Time_{\texttt{S75}} = \frac{I_{\texttt{S75}} \times \text{CPI}_{\texttt{S75}}}{750\ MHz}$$
$$Time_{\texttt{TP500}} = 2 \times Time_{\texttt{S75}}$$
$$\frac{1000 \times 2.7}{500\ MHz} = 2 \times \frac{I_{\texttt{S75}} \times \text{CPI}_{\texttt{S75}}}{750\ MHz}$$

$$I_{\texttt{S75}} \times \text{CPI}_{\texttt{S75}} = \frac{1,000 \times 2.7 \times 1.5}{2}$$
$$I_{\texttt{S75}} \times \text{CPI}_{\texttt{S75}} = 2,025\ cycles$$

$$A = \text{number of ALU instructions executed in } \texttt{S75}$$
$$A \times 1 + 200 \times 2.2 + 280 \times 3 = 2,025$$
$$A + 1,280 = 2,025$$
$$A = 745\ instructions$$

$$Percentage\ change = \frac{745 - 400}{400} \times 100 = 86.25\%$$

Another way to arrive at the same answer:

$$Time_{\texttt{S75}} = \frac{I_{\texttt{S75}} \times \text{CPI}_{\texttt{S75}}}{Freq_{\texttt{S75}}}$$
$$Time_{\texttt{S75}} = \frac{(480 + A) \times \frac{A+1,280}{480+A}}{750\ MHz}$$
$$Time_{\texttt{S75}} = \frac{A + 1,280}{750\ MHz}$$
$$Time_{\texttt{TP500}} = 2 \times Time_{\texttt{S75}}$$
$$\frac{1000 \times 2.7}{500\ MHz} = 2 \times \frac{A + 1,280}{750\ MHz}$$
$$\frac{1000 \times 2.7 \times 1.5}{2} = A + 1,280$$
$$A + 1,280 = 2025$$
$$A = 745\ instructions$$