EI338 Computer System Engineering Homework 1

Zhou Litao 518030910407 F1803016

September 24, 2020, Fall Semester

Exercise 1

1) CPU clock rate is 2 MHz,and program takes 40 million cycles to execute,please calculate CPU time.
2) CPU clock rate is 100 MHz,and program takes 40 million cycles to execute,please calculate CPU time.
3) CPU clock rate is 2 MHz,and program takes 80 million cycles to execute,please calculate CPU time.

Solution.

$$\begin{aligned}\text{CPU Time} &= \frac{\text{Seconds}}{\text{Program}} = \frac{\text{Cycles}}{\text{Program}} \times \frac{\text{Seconds}}{\text{Cycle}} \\ &= \frac{\text{Cycles}}{\text{Clock Rate}}\end{aligned} \tag{1}$$

1. CPU Time $= \frac{40 \text{ million}}{2 \text{ million}} = 20s$
2. CPU Time $= \frac{40 \text{ million}}{100 \text{ million}} = 0.4s$
3. CPU Time $= \frac{80 \text{ million}}{2 \text{ million}} = 40s$

□

Exercise 2 Suppose that there are three components in a computer system which can be improved (such as IO, CPU, memory), and the acceleration ratio of the three components is: $S_1 = 15, S_2 = 10, S_3 = 5$ (S_i represents the acceleration ratio of the $i-th$ component).If the proportion of the components that can be improved is:$F_1 = 30\%, F_2 = 20\%, F_3 = 10\%$ .Please calculate $S_{overall}$.

Solution.

$$\begin{aligned} S_{overall_1} &= \frac{1}{(1-F_1)+\frac{F_1}{S_1}} = \frac{1}{(1-0.3)+\frac{0.3}{15}} = \frac{25}{18} \\ S_{overall_2} &= \frac{1}{(1-F_2)+\frac{F_2}{S_2}} = \frac{1}{(1-0.2)+\frac{0.2}{10}} = \frac{50}{41} \\ S_{overall_3} &= \frac{1}{(1-F_3)+\frac{F_3}{S_3}} = \frac{1}{(1-0.1)+\frac{0.1}{5}} = \frac{25}{23} \\ \Longrightarrow \ S_{overall} &= S_{overall_1} \times S_{overall_2} \times S_{overall_3} = \frac{25}{18} \cdot \frac{50}{41} \cdot \frac{25}{23} = 1.841 \end{aligned} \tag{2}$$

□

Exercise 3

1) Suppose CPU Clock Speed is 1000MHz. A benchmark has 1000 instructions:300 instructions are loads/stores (each take 2 clock cycles),400 instructions are sub (each takes 1 cycle),300 instructions are square root (each takes 40 cycles).Please calculate the CPI and the CPU time for the benchmark.

2) Suppose CPU Clock Speed is 1000MHz. And two different programs are running on the computer.Both of them need three types of instructions:I_1, I_2, I_3.I_1 needs one cycle,I_2 needs two cycles,I_3 needs three cycles. Program one includes 10 millions I_1, 5 millions I_2, 1 millions I_3. Program two includes 5 millions I_1, 2 millions I_2, 2 millions I_3.Please calculate the CPU time and CPI of the two programs.

Solution.

1.

$$\text{CPI} = \frac{300 \times 2 + 400 + 300 \times 40}{1000} = 12.7 \tag{3}$$

$$\text{CPU time} = \frac{12.7 \times 1000}{1,000,000,000} = 1.27 \times 10^{-5}s \tag{4}$$

2. For program 1,

$$\text{CPI} = \frac{10m + 5m \times 2 + 1m \times 3}{10m + 5m + 1m} = 1.4375 \tag{5}$$

$$\text{CPU time} = \frac{1.4375 \times 16m}{1,000m} = 0.023s \tag{6}$$

For program 2,

$$\text{CPI} = \frac{5m + 2m \times 2 + 2m \times 3}{5m + 2m + 2m} = 1.6667 \tag{7}$$

$$\text{CPU time} = \frac{10m}{1,000m} = 0.01s \tag{8}$$

□

Exercise 4 You can choose any of the technologies introduced in the compiler/operating system/architecture,but the technologies should be related to the impact on the overall performance of the system. Based on the technologies you chosen, read some related paper.And write a summary(such as brief introduction of the technologies, your thoughts, ideas, conclusions and so on.At least 100 words).

Solution.

The impact of *multithreading* on computer performance Multithreading is the ability of a processor to provide multiple threads of execution concurrently, supported by the operating system. Multithreading allows multiple threads to share the functional units of a single processor in an overlapping fashion, i.e. multithreading shares most of the processor core among a set of threads, duplicating only private state, such as the registers and program counter. There are various kinds of multithreading. For example, *Fine-grained multithreading* switches between threads on each clock, while *Coarse-grained multithreading* only switches on costly stalls, such as level two or three cache misses. The most common implementation of multithreading is called *Simultaneous multithreading*, where techniques such as register renaming and dynamic scheduling allow multiple instructions from independent threads to be executed without regard to the dependences among them, in order to exploit the superscalar ability of the processor.

The ultimate goal of multithreading is to improve the throughput of the whole system. From my perspective, these three techniques shows an increasing pattern in complexity, and SMT exploits the thread-level parallelism to the most. My another observation is that different techniques may serve for particular use. For example, coarse-grained multithreading works better on multiple threads that are executing different tasks. In fact, for a single-core processor with no superscalar, SMT will make no sense. Therefore, it can also be concluded that multithreading techniques are advancing as the rest of the system evloves. □