Performance Measurement of Personal Computer

Anubhab Majumdar
Department of Computer Science
North Carolina State University
Email: amajumd@ncsu.edu

Arun Jaganathan
Department of Computer Science
North Carolina State University
Email: ajagana@ncsu.edu

***Abstract*—In this report we describe a set of experiments performed to benchmark the performance of a typical personal computer. The benchmarking is not a performance estimation based on hardware specification - we have measured the different overheads levied by the OS and determined the hardware constrains to gain a true perspective of the system performance. The report details out the experiments and analyzes the results to draw conclusions about the performance of the system being tested.**

I. Introduction

Specifications of a computer system does not always convey it's true performance. The hardware and operating system introduces various overheads which constrains performance. The measure of these overheads are important as they help benchmark the true performance of any system. This knowledge is important for developers as their application runs atop the operating system and heavily uses the services provided by it; thus any bottlenecks in the OS will translate into their applications and degrade performance. Also, the OS determines the baseline "responsiveness" the user expects from the system and applications should not be far from this baseline to ensure smooth customer experience.

A. Goals

Our primary goal was to benchmark the CPU, OS services and memory in details and analyze the results to draw conclusions about their performance. The CPU and OS services experiments are designed and implemented by Anubhab Majumdar. Arun Jaganathan designed the experiments to test memory components and implementation was shared by both the authors.

B. Language

We used trusty C language to design and implement the experiments. The code was compiled with **Apple LLVM version 7.3.0 (clang-703.0.31)** with no optimizations because we wanted the assembly code to be in order of our original program.

C. Duration

We worked on this project for around 70 hours spanning over 3 weeks. This includes determining the deliverables, reading relevant research papers, designing the experiments, coding the experiments, data consolidation, analysis of the data and drafting this report.

II. Machine Description

We have tested one of our personal computer, a MacBook Air. Following are the details of the machine:

1) **Model Name**: MacBook Air
2) **Model Identifier**: MacBookAir7,2
3) **Processor Name**: Intel Core i5
4) **Processor Speed**: 1.6 GHz
5) **Number of Processors**: 1
6) **Total Number of Cores**: 2
7) **L2 Cache**: 256 KB (per core)
8) **L3 Cache**: 3 MB
9) **Memory**: 8 GB
10) **Memory Type**: DDR3
11) **Memory Speed**: 1600 MHz
12) **Memory bus speed**: 1066 MHz
13) **Link Speed**: 5.0 GT/s
14) **Link Width**: x4
15) **Storage**: 128 GB
16) **Medium Type**: Solid State Drive
17) **Operating System**: MacOS Sierra (Version 10.12)

III. Experiments

The experiments are divided into two broad categories:

- CPU, scheduling and OS services experiments
- Memory experiments

Each of these categories are aggregation of small experiments that measure various overheads associated with OS or constraints of hardware. The following subsections describes the methodology, presents the findings and draw inference about it.

Before we dive into explanation about the experiments, we would like to explain the units of measurement. We have measured the operations in term of **cycles** and **time**. CPU cycles were measured using the C function **rdtsc** (Read Time Stamp Counter) to read the CPU cycles before and after any operation and the difference is assumed as the number of cycles consumed by the operation. We should also mention that before using rdtsc, we have used the C function **cpuid** to prevent any out of order execution by CPU. Similarly, time was measured in microseconds by using the **gettimeofday** function before and after an operation and the difference is considered as the time taken to perform the task.

A. *CPU, scheduling and OS services experiments*

The experiments are conducted to measure 5 key overheads. Their names and descriptions are listed the subsections below.

1) Measurement Overhead: We start our benchmarking experiments by measuring overhead to perform 2 basic tasks - reading from memory and looping through multiple iterations of a task.

a) Methodology: For reading time, we allocated a character array of size 10,000 bytes and initialized 'a' in all the locations. The experiment was to measure, in terms of cycles and time, the cost of reading these 10,000 characters. At first, the cost of cpuid is measured using rdtsc function. The program is executed 100 times separately using bash script and the readings the noted. Next, the file read operation of 100,000 characters from the array is performed. The timestamp counter values are measured just before and just after read operation. The difference is noted in a file. This, again, is executed separately 100 times and the results are noted. The mean and median are calculated on the results and noted.

In a similar fashion, the cost is measured in units of time as well. We replaced rdtsc with gettimeofday function and calculated the difference of time measured in microseconds. The results are noted in sheet "Reading Time" in measurements.xls file.

b) Results: The RAM speed of the machine is 1600 MHz (as specified in System Information). Thus to perform 1 read operation it should take 0.6 ns. For reading 10,000 characters it should take 6.25 μs. We estimated it would take thrice the time. We considered page fault, cache miss, TLB miss for overhead and hardware pre-fetching as advantage.

Base Hardware Performance	RAM: 1600 MHz
Estimated Overhead	Thrice of 1 read
Predicted Performance	18.75 μs
Measured Performance	21.65 μs

From the above table we could see our predicted read time is quite close to measured time. The difference could be because of our estimate of page faults handling time. We believe the results are quite accurate as we have measured, to the best of out ability, only the read time and subtracted the for loop time from the results.

2) Loop Overhead: In this experiment we had to "report the overhead of using a loop to measure many iterations of an operation".

a) Methodology: This experiment is quite similar to the previous one. We wrote a code which contains a for-loop "looping" 10,000 times. We measured the cost, in terms of both cycle and time, twice - once by commenting the for-loop part and once by keeping the for-loop.

b) Results: The CPU speed is 1.6 GHz, i.e., it takes 0.6 ns to run one CPU cycle. The assembly code replacing the for-loop had 4 instructions. We estimated one cycle is required for each instruction.

Base Hardware Performance	CPU: 1.6 GHz
Estimated Overhead	No OS overhead
Predicted Performance	24 μs
Measured Performance	24.33 μs

From the above table we could see our predicted read time is quite close to measured time. We believe the results are quite accurate as we have measured, to the best of out ability, only the for-loop time and subtracted any extra time we measured along with the for-loop (like _cpuid function cost).

IV. Conclusion

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

Appendix A
Proof of the First Zonklar Equation

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

Acknowledgment

The authors would like to thank...

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.