Assignment 0 – Chen-Yu Chang U93093024

Part 1.

1a.

* CPU: Intel(R) Core(TM) i7-9700 CPU @ 3.00GHz
* Operating Frequency: 799.987 MHz
* 8 Cores

1b.

* L1d Cache: 32K

L1i Cache: 32K

L2 Cache: 256K

L3 Cache: 12288K

* Microarchitecture: Coffee Lake-S
* There are 8 cores and 8 siblings, so there is one virtual processor per core.
* Max Memory Bandwidth: 41.6GB/s

Part 2.

2a.

The method to determine the accuracy of the timer is to do repetitive operation many times. Using the average time of every run, we can decrease the number of cycles and compare the average. When numbers diverge, it is reaching its limit of accuracy. For resolution, we can find that it should be within milliseconds range.

2b.

There are problems for RDTSC-based method. When using multiple new CPU chips, there will be some difference on time counters on different CPUs. With different frequency depending on the load, it displays a non-exact time elapsed. Therefore, when measuring time, we will get different time on distinct CPUs. However, the timers can still be useful. The time counter can be synchronized before RDTSC is run. Also, we can keep track on the instructions run and eliminate those instructions skewed. Overall, we can still know the approximate.

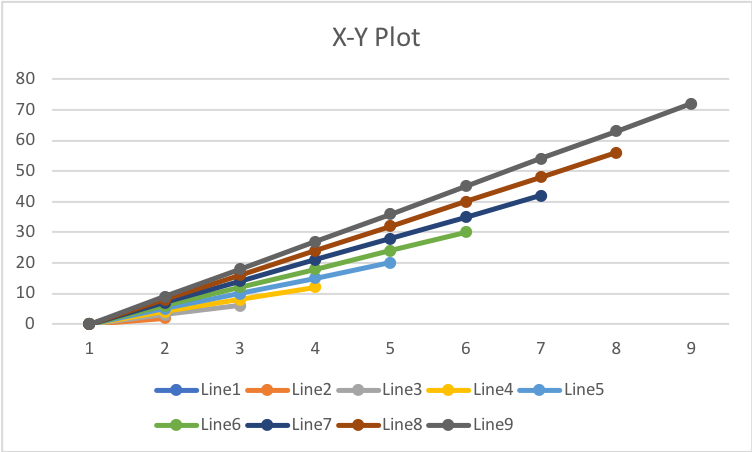
2c.

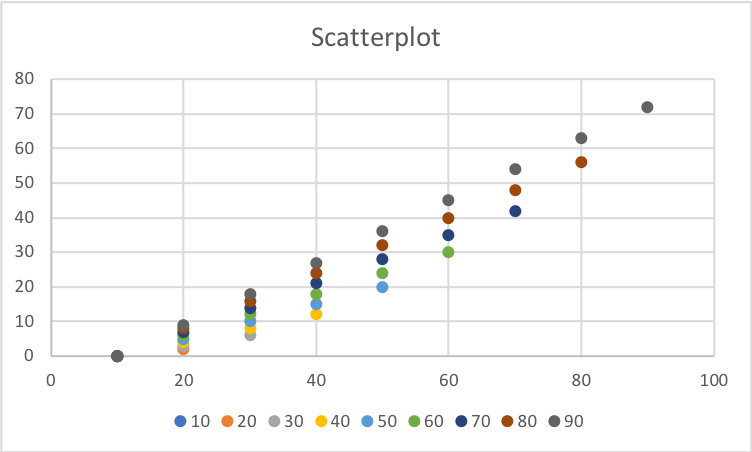
* We do not have to change for gettimeofday
* For RDTSC, we will change the clock rate (CLK\_RATE) to 3.0e9
* For times, we will change the typical second ticks of 100 to a system configuration (sysconf(\_SC\_CLK\_TCK))

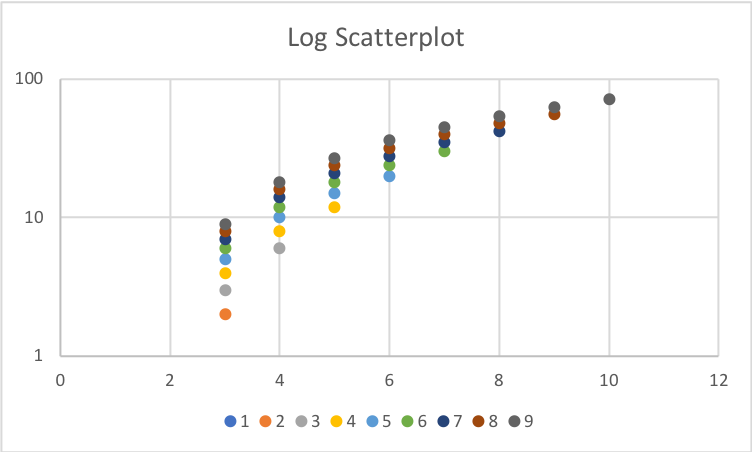
2f.

* The closest I can get is 0.999217708 seconds. The resolution is to its nanoseconds and the standard deviation is about 0.042 seconds.

Part 3.

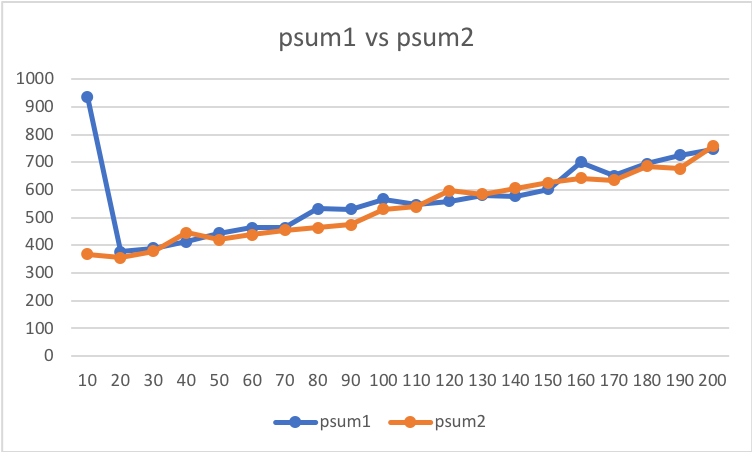






Part 4.

4b.



4c.

A good way to get rid of anomalies could be removing redundant data from database or reducing the need to restructure the database every time new fields are added.

4d.

We can get that the CPE for psum1 is about 6.34 and 4.82 for psum2, which is not the same written in the material. The reason might be the gettime function with non-fixed frequency makes the calculation have some flaws.

Part 5.

5a.

Starting a loop

done

real 0m0.152s

user 0m0.148s

sys 0m0.002s

5b.

Starting a loop

done

real 0m0.029s

user 0m0.026s

sys 0m0.002s

5c.

.file "test\_O\_level.c"

.section .rodata

.LC0:

.string "\n Starting a loop "

.LC1:

.string "\n done "

.text

.globl main

.type main, @function

main:

.LFB0:

.cfi\_startproc

pushq %rbp

.cfi\_def\_cfa\_offset 16

.cfi\_offset 6, -16

movq %rsp, %rbp

.cfi\_def\_cfa\_register 6

subq $32, %rsp

movl %edi, -20(%rbp)

movq %rsi, -32(%rbp)

movq $0, -16(%rbp)

movl $.LC0, %edi

call puts

movq $0, -8(%rbp)

jmp .L2

.L3:

addq $3, -16(%rbp)

addq $1, -8(%rbp)

.L2:

cmpq $100000000, -8(%rbp)

jle .L3

movl $.LC1, %edi

call puts

leave

.cfi\_def\_cfa 7, 8

ret

.cfi\_endproc

.LFE0:

.size main, .-main

.ident "GCC: (GNU) 4.8.5 20150623 (Red Hat 4.8.5-44)"

.section .note.GNU-stack,"",@progbits

5d.

.file "test\_O\_level.c"

.section .rodata.str1.1,"aMS",@progbits,1

.LC0:

.string "\n Starting a loop "

.LC1:

.string "\n done "

.text

.globl main

.type main, @function

main:

.LFB11:

.cfi\_startproc

subq $8, %rsp

.cfi\_def\_cfa\_offset 16

movl $.LC0, %edi

call puts

movl $100000001, %eax

.L3:

subq $1, %rax

jne .L3

movl $.LC1, %edi

call puts

addq $8, %rsp

.cfi\_def\_cfa\_offset 8

ret

.cfi\_endproc

.LFE11:

.size main, .-main

.ident "GCC: (GNU) 4.8.5 20150623 (Red Hat 4.8.5-44)"

.section .note.GNU-stack,"",@progbits

--- There is no ‘steps’ and ‘i’ variables anymore.

5e.

Starting a loop  
steps: 300000003  
  
 done  
  
real: 0.065s  
user:0.041s  
sys: 0.002s

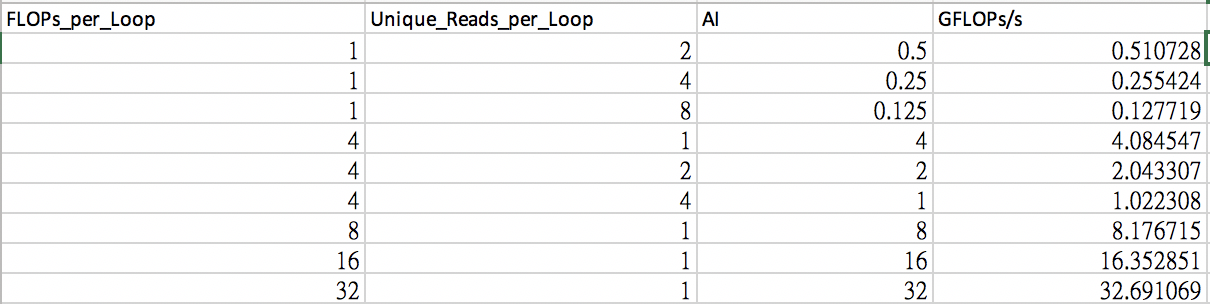
The code was optimized to calculate steps. Movl $300000003, %esi was printed.

Part 6.

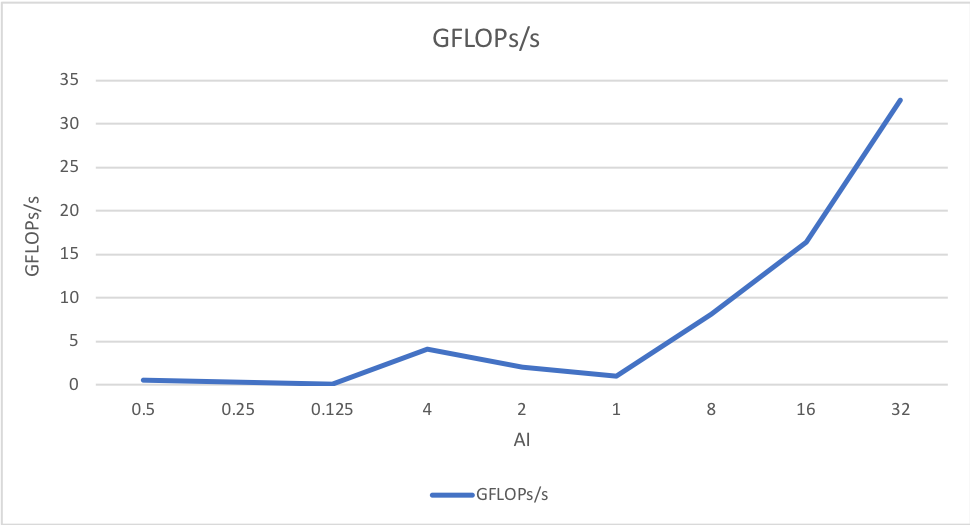
6c.

The memory bandwidth is 11508.9742MB/s. This is less than the maximum bandwidth of the processor (41.6GB/s)

6d.



6e.



6f.

The graph shows that GLOPs/s increases as AI increases. Within the range of [1/8,2]. We can see that it will hit to a wall and decrease a little. Before that, it has a limited bandwidth. However, when AI goes over the range, it still shoots up.

Part 7.

7a. I did not miss any of the parts.

7b. This assignment took me about 8-10 hours in total, separated in several days.

7c. I spent the most time on part 4, but not too much.

7d. Not now.