**Assignment 0**

1)

a) CPU - Intel(R) Core(TM) i5-4570S CPU @ 2.90GHz

b) Operating Frequency of Cores – 800MHz

c) No. Of Cores – 4

d) L1d cache: 32KB

L1i cache: 32KB

L2 cache: 256KB

L3 cache: 6144KB

e) Microarchitecture- Haswell Intel 4th Gen, x86\_64

f) There are 4 cores and 4 siblings, so there is one virtual processor per core.

Part 2)

1. Accuracy is determined by the degree to which certain values are close to a correct value or standard. In this case the resolution is 10^-9 seconds. The problems with the RDTSC (Read Time-Stamp Counter- returns the number of clock cycles since last reset) timer are that:

* On older multi-cored processors, the rate could change differently on different cores, as they scaled their clock speeds according to different loads;
* On more recent processors, the rate remains constant while the clock speed changes, so that timings on a lightly-loaded core may seem slower than they are.
* Out-of-order execution may mean that the register isn't read when you think it is.

1. Modifications – I changed CLK\_RATE 2.9 \* 10^9, as this was the maximum operating frequency found in part 1.
2. Resolution- 10^-9 seconds,

Error- The error is calculated by taking the average of 10 values of the time, and then finding the error or deviation of the maximum of the 10 values using the mean. The mean was calculated to be 0.000533922s. The maximum time calculated in the trials was 0.000584384s. The error calculated was hence +- 9.5%. This occurs since there is an increased amount of rounding error if run multiple times.

Variance – Roughly 10 ^-7 seconds

Part 4)

|  |  |  |
| --- | --- | --- |
| 10 | 1647 | 424 |
| 20 | 500 | 441 |
| 30 | 661 | 428 |
| 40 | 467 | 451 |
| 50 | 730 | 477 |
| 60 | 547 | 538 |
| 70 | 559 | 527 |
| 80 | 563 | 560 |
| 90 | 636 | 598 |
| 100 | 689 | 640 |
| 110 | 701 | 631 |
| 120 | 678 | 656 |
| 130 | 1259 | 687 |
| 140 | 754 | 745 |
| 150 | 780 | 754 |
| 160 | 802 | 764 |
| 170 | 818 | 796 |
| 180 | 895 | 854 |
| 190 | 871 | 843 |
| 200 | 916 | 895 |

Part 5)

a)Starting a loop

done

real 0m0.207s

user 0m0.203s

sys 0m0.003s

b) Starting a loop

done

real 0m0.004s

user 0m0.000s

sys 0m0.004s

c) The Loop:

.LFB0:

.cfi\_startproc

pushq %rbp

.cfi\_def\_cfa\_offset 16

.cfi\_offset 6, -16

movq %rsp, %rbp

.cfi\_def\_cfa\_register 6

subq $48, %rsp

movl %edi, -36(%rbp)

movq %rsi, -48(%rbp)

movq $0, -8(%rbp)

movl $.LC0, %edi

call puts

movq $0, -32(%rbp)

jmp .L2

.L3:

addq $3, -8(%rbp)

addq $1, -32(%rbp)

.L2:

cmpq $100000000, -32(%rbp)

jle .L3

movl $.LC1, %edi

call puts

leave

.cfi\_def\_cfa 7, 8

ret

.cfi\_endproc

d) After Optimization-

main:

.LFB11:

.cfi\_startproc

subq $8, %rsp

.cfi\_def\_cfa\_offset 16

movl $.LC0, %edi

call puts

movl $.LC1, %edi

call puts

addq $8, %rsp

.cfi\_def\_cfa\_offset 8

ret

.cfi\_endproc

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

e)

Starting a loop

300000003

done

real 0m0.197s

user 0m0.000s

sys 0m0.004s

The code has been optimized so that there is no steps variable but, rather the final value of steps is printed out as an immediate

Part 6)

b) The payload loops are what I expected- it's doing different arithmetic operations on the different, unique values of arrays, and calculates the time taken to perform them. It does the same operation multiple times and takes the best value as the end result.

c) The Memory BW = 8050.0421 MB/s. This is much less than the maximum bandwidth of the processor which is 25.6 GB/s.

d)

AI = 1.0

Gflops/s = 2.722377

AI = 2.0

Gflops/s = 3.549086

AI = 3.0

Gflops/s =4.983474

AI = 4.0

Gflops/s =5.191539

AI = 5.0

Gflops/s = 5.122549

AI = 6.0

Gflops/s = 5.089410

AI = 7.0

Gflops/s =5.091758

f) It can be seen from the plot above that the GFLOPS/s plateaus roughly after arithmetic intensity of 3. The GFLOPS/s plateaus roughly after this point because there is a limit to the performance of the CPU. The GFLOPS/s is bounded by the maximum memory bandwidth of the CPU, even if arithmetic intensity is increased past a value of 3.

Part 7

1. Not missing many skills
2. 4 hours
3. Trying to get the scatter plot for Part 4