# Report for CS2610 Assignment 4

## Q) How did you measure latency?

We measured the latency using **RDTSCP** along with **CPUID** instructions.
To measure the time at any we instant we use the following code:

```
asm volatile ("CPUID\n\t"
        "RDTSCP\n\t"
        "mov %%edx, %0\n\t"
        "mov %%eax, %1\n\t": "=r" (cycles_high), "=r" (cycles_low)::
        "%rax", "%rbx", "%rcx", "%rdx");
```

Then we use the following code to extract time from given variable:

```
start = ( ((uint64_t)cycles_high << 32) | cycles_low );
```

Now we perform some operation on an element which will be called and then repeat the above process to find end time as well.

```
asm volatile("RDTSCP\n\t"
        "mov %%edx,     %0\n\t"
        "mov %%eax,     %1\n\t"
        "CPUID\n\t": "=r" (cycles_high1), "=r" (cycles_low1)::
        "%rax", "%rbx", "%rcx", "%rdx");

end = ( ((uint64_t)cycles_high1 << 32) | cycles_low1 );
```

Then we measure the latency of accessing the element as (end - start). Reference 1.
To improve the accuracy of the values obtained we repeat the code 100 times to reduce noise in our plots.
We isolated the cpu using **isolcpus** and ran the code on a single core during run to prevent OS scheduled operations.

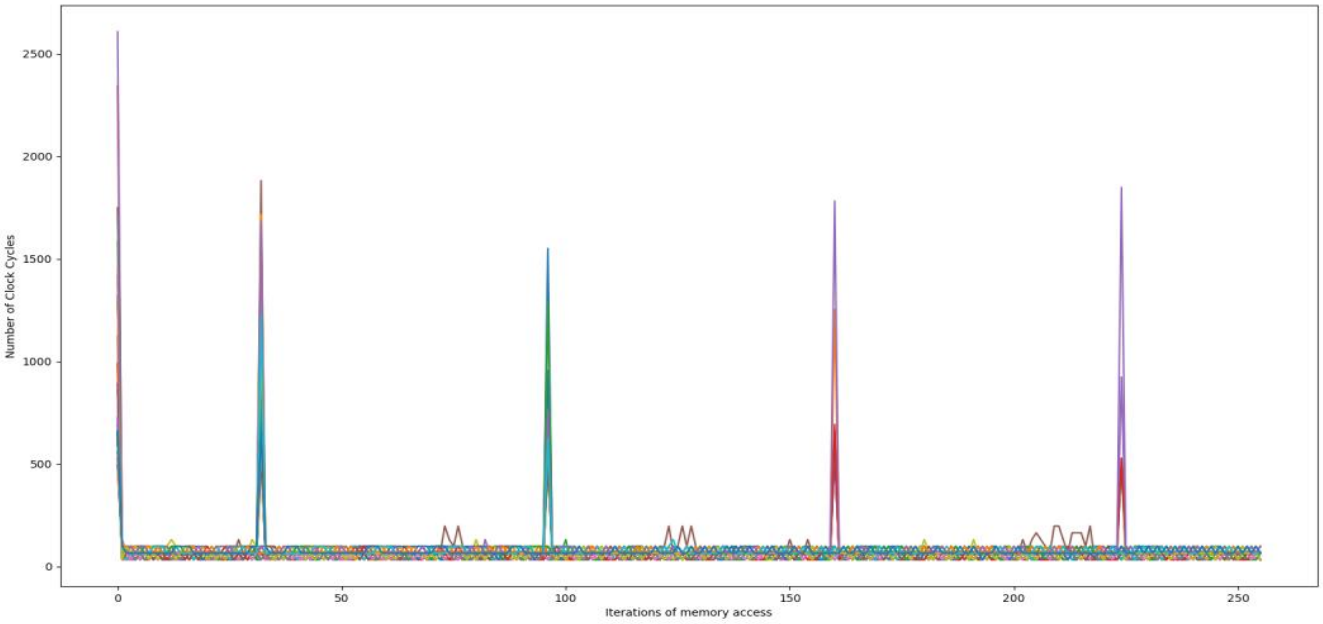
## Q) What are the actual specifications of the L1 cache present in your system?

We used the command **lscpu** command for finding specifications. Reference 2.

```
Caches (sum of all):
L1d:                   192 KiB (6 instances)
L1i:                  192 KiB (6 instances)
L2:                   3 MiB (6 instances)
L3:                   16 MiB (1 instance)
```

## Q)What is the cache block size inferred by reverse engineering?

The inferred cache block size 64B.
On running the c program on an array of characters of size 256,
We infer from the miss times that they spike at certain values of memory accesses.
From the below graph we see that the plot with Number of clock cycles vs iteration of memory accesses, we see that the time spikes at 32, 96, 160, 224



From the difference between the values we obtain the spikes at we can infer that the block size is 64B.

It matches with the actual cache block size of the processor inferred from Reference 3..

## Q)What is the set associativity inferred by reverse engineering?

Using **lscpu** we get L1d cache size is 192KB.
We initialize a array arr of massive size
We access arr+192, arr+2*192, arr +3*192, …..
Expected latency  values are low, low, …. , high, high, ….
Thus the number of consecutive values of low or high(after the first set of lows) would be the set associativity of the cache.
Looking at the value we obtain for latency values:

```
0      204        <-cold miss
1      196        <-low
2      203        <-low
3      204        <-low
4      211        <-high 1
5      214        <-high 2
6      228        <-high 3
7      235        <-high 4
8      237        <-high 5
9      238        <-high 6
10      233       <-high 7
11      211       <-high 8
12      200       <-low
13      193       <-low
14      193       <-low
15      189       <-low
16      188       <-low
17      182       <-low
```

From the values we infer 4 - 11 are high latency values giving a set associativity value of 8.
The value found matches with the set associativity from Reference 3.

**References :**

**1:**
**https://www.intel.com/content/dam/www/public/us/en/documents/white-papers/ia-32-ia-64-benchmark-code-execution-paper.pdf**
**2:**
**https://unix.stackexchange.com/questions/167038/is-there-any-way-to-know-the-size-of-l1-l2-l3-cache-and-ram-in-linux**
**3:**
**https://www.cpu-world.com/CPUs/Zen/AMD-Ryzen%205%20Mobile%205600H.html**

CS20B052 Mohammed Rizan Farooqui
cs20b52@smail.iitm.ac.in

CS20B021 Chathurvedhi Talapaneni
cs20b021@smail.iitm.ac.in