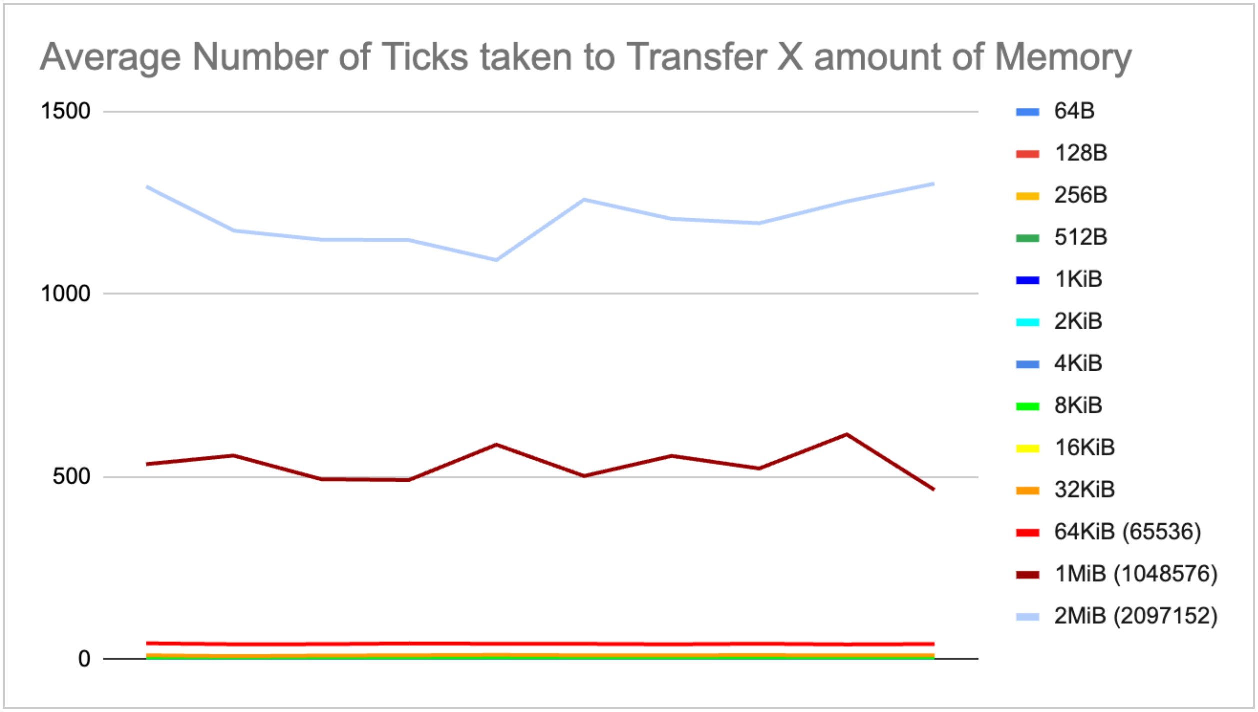
Somya Mehta
Anna Nefedenkova

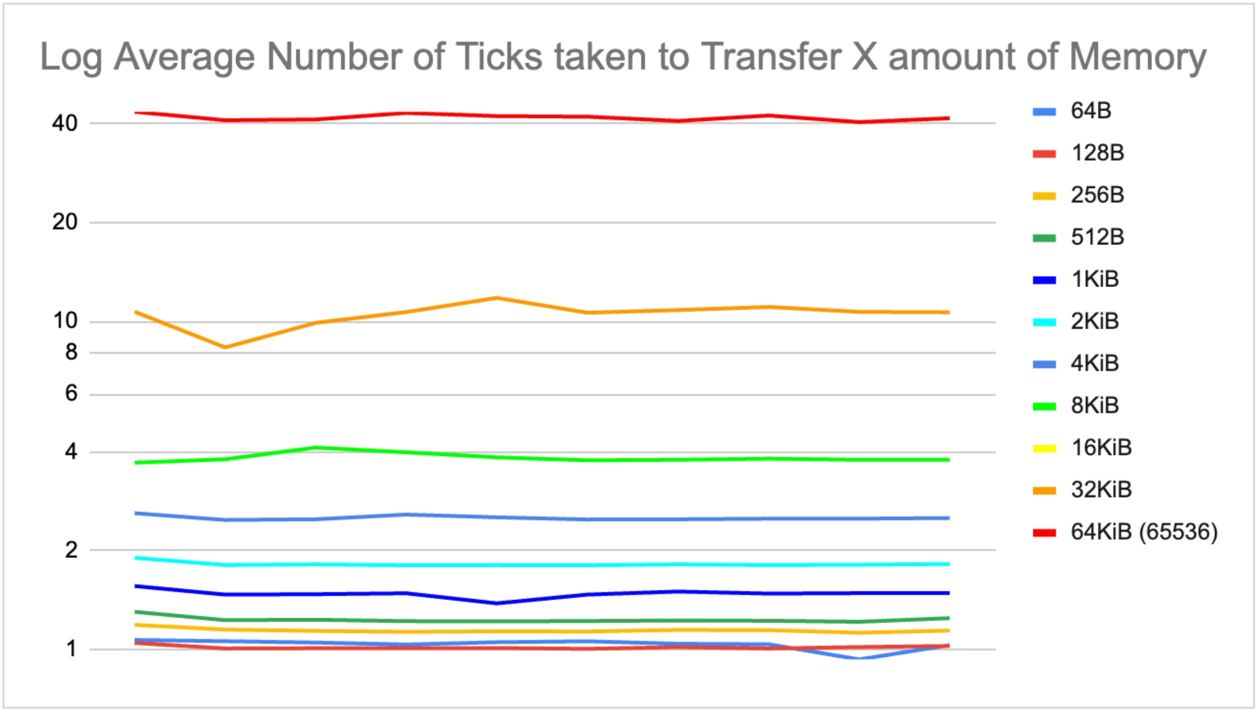
Work Log of meetings:
Monday 11:35 am - 1 pm
Tuesday Office Hours
Wednesday 11:35 - 3 p.m. including office hours
Thursday Office Hours
Sunday 7 pm - 8 pm
Monday Office hours
Tuesday Office hours

Question 1:

In order to complete the following coding part we referred to the following source (https://developer.arm.com/documentation/den0024/a/The-,A64-instruction-set/Memory-access-instructions/Memory-barrier-and-fence-instructions) to replace the clflush and mfence function for the AArch64 architecture. A thorough research was accomplished and the following link above was the best result found. Other alternatives are commented out in the code section. The main issue with building the code for this assignment was debugging it. The reason our team went with the chosen instruction was the reliability of it cleaning and validating the cache. We have generated our results, by running 10 times the 1,000,000 iterations “for” loop and took the average of the ticks in each 10 cycles. Generating more data allowed us to look at a bigger number of iterations and determine whether we have any outliers.

It can be observed that after 2KiB and 8KiB of memory DRAM operations, the number of ticks increases significantly with the memory size being accessed. Before 2KiB, all performances were within the range of 100 nanoseconds, or 1 tick. There are no significant outliers. It can be observed that the consistency of ticks it takes to complete one cycle (1,000,000 iterations) declines with greater memory size. The following observation mentioned above can be explained by the architecture of the DRAM. Not only is it expected that dealing with larger chunks of memory is much more time-consuming, but we know that accessing multiple rows is much less efficient as opposed to when the data is located in one row. We also know that there will be multiple pre-precharge operations, as we have many more rows that need to be accessed. A longer tRP can delay the pre-charging of a row after a read or write operation, slowing down subsequent memory accesses. A longer tRRD can increase the time required to access different rows in the same bank, affecting memory access efficiency. A greater memory size with many more operations taking place causes the cycles to vary in the overall number of ticks taken to complete them.





We have included a second table, where only the lower range of memory transfers is included for a better representation.

Question 2:

M1 Chips use the LPDDR4X at 4267MHz speed, however, we were only able to find the following closest datasheet to our machine on the manufacturer's website (Hynix: https://product.skhynix.com/support/downloads/techs.go), therefore we will compare the specs with the DDR4-3200 16Gb SDRAM:

Values: (the ones in parentheses, were additionally googled)
tRCD (Row to Column Delay): 13.75 ns. (14-20 cycles)
tRP (Row Precharge Time): 13.75 ns. (14-20 cycles)
tCAS/tCL (Column Address Strobe latency): 22 tCK (16-22 cycles)
tRC (Row Cycle Time): 45.75 ns. (42-60 cycles)
tRAS (Row Address Strobe): 32 ns. (42-60 cycles)

Values not included in the datasheet (were googled separately):
tRRD (Row to Row Delay): 4-6 cycles.
tWR (Write Recovery Time): 10-15 cycles.
tWTR (Write to Read Delay): 2-6 cycles.
tCWD (Column Write Delay): 4-6 cycles.
tRTP (Read to Precharge): 4-6 cycles.
tCCD (Column-to-Column Delay): 2-4 cycles.
tBURST (Data transfer time): 8 cycles.

As it is very difficult to calculate each parameter of the DRAM separately, we are giving an overall description of the following parameters and how they fit into the observations in question 1. In question 1, we were able to see how long it takes for our computer to transfer data from one place in the DRAM to another. The above timing parameters all contribute to the amount of time it would take for the data to transfer.

tRCD: to access data within the DRAM, the memory controller must activate a specific row. This, in turn, creates a delay of tRCD clock cycles before the memory controller can specify a column address. This therefore contributes to the overall time it takes to transfer data within the DRAM, specifically within one row of the DRAM.

tRP: After accessing data within a specific row, the memory controller needs to precharge that row before it can activate another row. tRP is the delay in clock cycles between when the memory controller does a precharge and when it is ready to accept a new row activation. Smaller tRP allows for more efficient use of memory bandwidth.

tCAS: the number of clock cycles it takes for the memory to access and provide data from the column address after the row has been activated.

tCL: delay in clock cycles between the memory controller specifying the column address for data access and providing the requested data.

tRC: the time it takes to activate a row, access data within the row, precharge, and close the row before being able to access the row again.

tRAS: delay between when a specific row in memory is activated and when it can be precharged.

While our values from the datasheet are not a perfect match to the machine we are running our code on, we know that the speed of our DRAM is 4267 MHz, which is very high. We know that the speed at which the memory operates and the number of cycles in a second is very high. Therefore, we can expect the operation to occur very quickly.

Question 3:
The Mac M1 chip computer that our team is working on has a “closed-row” row buffer policy. After running the operation for the second time, the time taken in ticks did not change throughout all memory size iterations.