For this lab, the objective of the written code was to emulate the built in c function of memset() and memcpy(), and to outperform it. Mine unfortunately was unsuccessful in outperforming, but the results came in to be mildly close. This was achieved through parallelization: dividing the buffer up into 4 pieces and using each core to copy or edit that segment.

The time elapsed was calculated by recording the number of cycles and dividing by the cycle speed, then it was confirmed by comparing with the recorded time in the program. The throughput was then calculated by dividing the buffer size by this time, and also confirmed with the throughput calculation from measured time. The virtual machine used was not able to handle 4GB of data at once, so the large dataset was replaced with a 2GB dataset.

Mine performed the worst in comparison to the builtin on the 4KB dataset, likely due to the overhead in splitting into threads or the cost of retrieving the data in the way that I did. None of my instances of memset or memcpy were able to outperform the builtin counterpart, except for my_memcpy on the 2GB dataset. It seemed to be that my functions got closer to the performance of the builtin as the dataset increased, in which case it makes sense that my_memcpy was able to win. For most however, besides the 4KB buffer trials, my throughput was decently close to that of the built in c functions.

Mode	Buffer size	Latency (cycles/operation)	Cycle speed (GHz)	Time elapsed (microseconds)	Throughput (bytes/sec)
memset	4KB	577	2.8	0.21	19410745234
my_memset	4KB	967067	2.8	345	11581411
memcpy	4KB	782	2.8	0.28	14322250639
my_memcpy	4KB	574420	2.8	205	19497928
memset	4MB	4517750	2.8	1613	2479110177
my_memset	4MB	6692852	2.8	2390	1673427113
memcpy	4MB	8959013	2.8	3200	1250137710
my_memcpy	4MB	14442272	2.8	5158	775501251
memset	2GB	2092461478	2.8	747308	2676273881
my_memset	2GB	2860019705	2.8	1021436	1958028467
memcpy	2GB	3974168653	2.8	1419346	1409099736
my_memcpy	2GB	3779078315	2.8	1349671	1481842802