Parallel Processing Project: Max Bandwidth

Aman Hogan-Bailey

University of Texas at Arlington

CSE-5351: Parallel Processing

Contents

Theoretical Maximum Bandwidth (Part 1)

The `lscpu` command indicates that the UTA cluster given has an Intel® Xeon® Processor E5-2680 v4. According to Intel's ARK website, the theoretical maximum bandwidth for this processor is 76.8 GB/s (or 76,800 MB/s). This value, however, warrants verification based on the system's configuration.

The `dmidecode` command indicates the memory is DDR4-2400, with four memory channels and an 8-byte bus width. Given this configuration, the theoretical bandwidth for a single processor is as follows:

$$Bandwidth_P = TransferRate * BusWidth * MemChannel$$

$$Bandwidth_P = 2400\frac{MT}{s} * 8 * \ 4 = 76{,}800\frac{MB}{s}$$

$$Bandwidth_P = 76{,}800\frac{MB}{s} = 76.8\frac{GB}{s}$$

This value confirms the Intel ARK website's reported bandwidth. Furthermore, the system has two processors, each independently accessing four memory channels. Assuming optimal utilization of all channels, the total theoretical memory bandwidth for the entire system is:

$$Bandwidth_T = Bandwidth_P * NumOfProcessors$$

$$Bandwidth_T = 76{,}800\frac{MB}{s} * 2 = 153600\frac{MB}{s}$$

This figure represents the maximum data transfer rate the system could theoretically achieve under ideal conditions.

Max Bandwidth Testing

My experiments used a size of 100,000,000 or (100 x 1000 x 1000) for the vector since neither my laptop nor the UTA cluster could process 1 billion values in a reasonable time. All results were compiled using level three GCC optimization. Below are the results for parts 2-4.

Max Bandwidth Testing Boilerplate Code (Part 2)

The first experiment used the boilerplate code in class. The write bandwidth was 1,873 MB/s (1.2% max bandwidth), and the read bandwidth was 5,548 MB/s (3.6% max bandwidth). The results are in the first cell of *Table 1* and *Table 2*.

Write Bandwidth Experiments (Part 3)

The experiments indicate that using *Non-Temporal Writes + Setting Memory to Zero Before Timing* yields the highest write bandwidth, achieving a peak bandwidth of 55,364 MB/s (36% max bandwidth) at 6 threads. For the *No Optimization*, the best result was also at 6 threads, with a bandwidth of 22934 MB/s. *Setting Memory to Zero Before Timing* had a maximum bandwidth of 24,990 MB/S at 4 threads. The maximum bandwidth for writes occurred between four and six threads for all optimization types over several experiments (Table 1).

Across all optimization types, the write bandwidth more than doubled going from one thread to six threads. Similarly, for all optimization types, the bandwidth decreased after six threads. The data gathered shows evidence to suggest that *Non-Temporal Writes + Setting Memory to Zero Before Timing yields* the highest bandwidth and that the highest bandwidth occurs between 4 to 6 threads (Figure 1).

Read Bandwidth Experiments (Part 4)

The experiments indicate that using an *Unroll Loop Size* of 1 and a thread size of 1 always yields the worst performance at around 5,548 MB/s. Doubling the *Unroll Loop Size* increases the bandwidth at a far less rate than doubling the *Number of Threads*. The highest bandwidths are achieved at an *Unroll Loop Size* greater than one and a *Number of Threads* over 8. The highest bandwidth was with a *Number of Threads* of 16 and an *Unroll Loop Size* of 16 at 46,183 MB/s (30% max bandwidth) (Table 2).

The data suggests that the thread count affects bandwidth more than unroll loop size. The data also suggests that *Number of Threads* of 16 and higher will achieve the highest read bandwidth (Figure 2).

Conclusions

In conclusion, this project assessed various optimizations to determine their effectiveness in achieving max memory bandwidth for both read and write operations on the UTA cluster. The theoretical maximum memory bandwidth is 153,600 MB/s. However, practical experiments revealed that performance falls significantly short of this ideal.

For write operations, the highest bandwidth *used Non-Temporal Writes + Setting Memory to Zero Before Timing*, reaching 55,364 MB/s, or 36% of the theoretical maximum, with six threads. This optimization consistently outperformed others between 4 to 6 threads. Additionally, across all optimization types, the maximum write bandwidth was observed to generally be achieved between 4 and 6 threads.

In read operations, the best performance was with 16 threads and an unroll loop size of 16, achieving a peak bandwidth of 46,183 MB/s, approximately 30% of the theoretical maximum. The experiments suggest that thread count has a more substantial impact on both read and write bandwidths than other factors such as unroll loop size. Specifically, the thread count was critical in optimizing bandwidth, with higher thread counts generally yielding better results for read operations.

Overall, while the tested optimizations, particularly "Non-Temporal Writes + Setting Memory to Zero Before Timing," significantly improved memory bandwidth from 1% to 36% utilization, the results also highlight the inherent challenges in fully leveraging the system's theoretical potential. Further refinement of these and other optimization strategies will be essential in closing the gap between theoretical and practical performance in high-performance computing environments.

References

None

Tables

Table 1

Write Bandwidth in MB/s

OPTIMIZATION TYPE	THREADS						
-	1	2	4	6	8	16	Max (56)
NO OPTIMIZATION	1873.03	15906.29	24426.44	22332.79	22934.4	21680.34	20831.96
SET MEM TO ZERO	8147.186	15926.02	24990.27	28048.28	24250.28	21757.42	20791.36
NON-TEMPORAL WRITES + SET MEM TO ZERO	18753.98	36113.14	51770.81	55364.78	41067.18	30657.11	27848.67

Table 2

Read Bandwidth in MB/s

UNROLL LOOP SIZE	THREADS						
-	1	2	4	6	8	16	Max (56)
1	5548.587	11823.2	17659.26	21231.76	29564.17	39803.77	38204.66
2	6780.363	11082.23	11759.63	29283.38	34911.61	41957.01	38974.58
4	7112.106	15241.37	22704.32	31341.01	35477.65	45514.73	38771.48
8	7310.7	15547.81	23645.08	32170.81	36292.61	45458.67	38795.57
16	7277.537	15612.33	27032.05	28612.17	35463.62	46183.56	38372.18

Figures

Figure 1.

Figure shows line plot of Bandwidth and Number of threads across various optimizations.

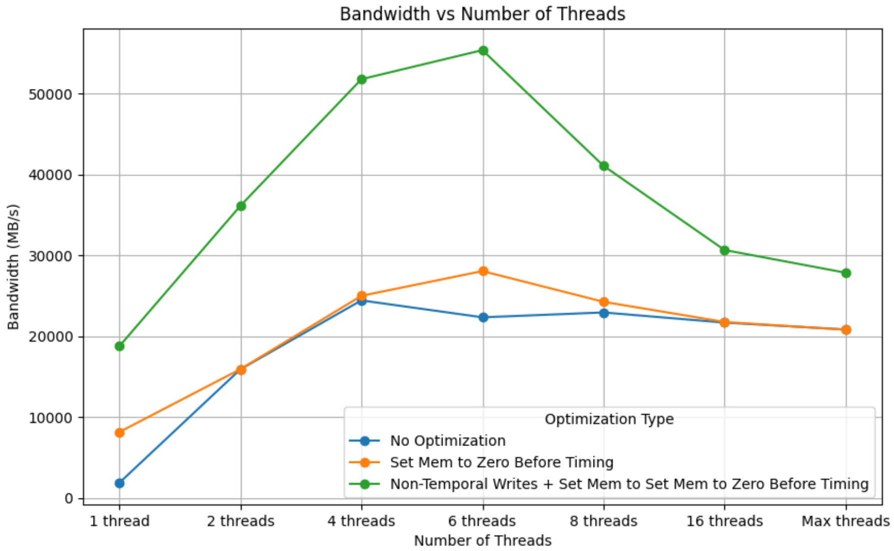
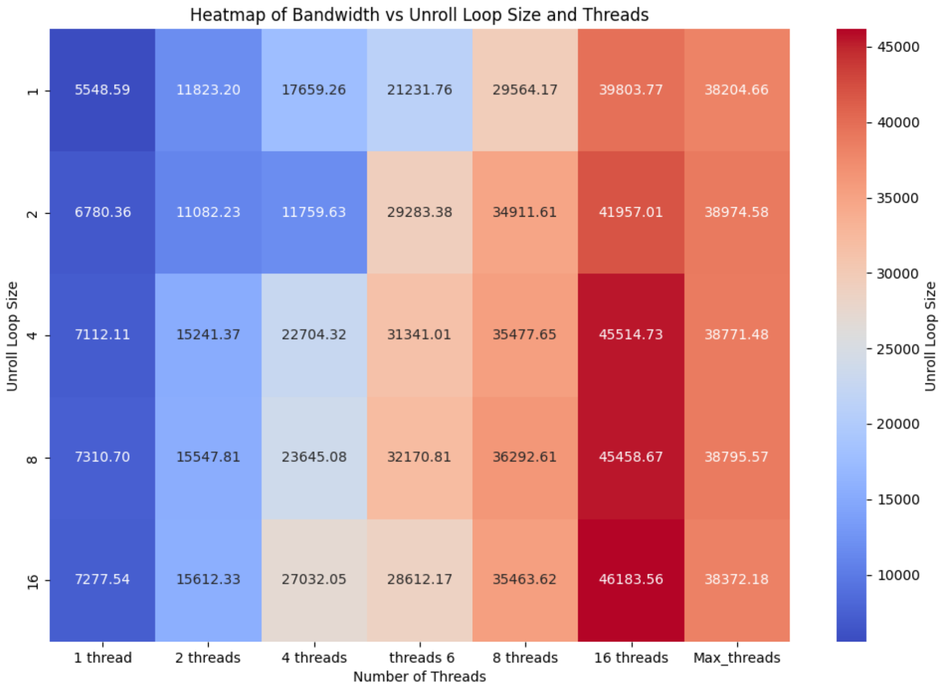


Figure 2.

Figure shows heatmap between Unroll loop size and number of threads.


Heatmap of Bandwidth vs Unroll Loop Size and Threads
Unroll Loop Size
1 thread
2 threads
4 threads
threads 6
8 threads
16 threads
Max_threads
Number of Threads
Unroll Loop Size