Design of Multi-Threaded Implementations:

(1) The first implementation uses one thread for each chunk of the input array, and each thread computes a portion of the saxpy operation. The second implementation uses multiple threads, and each thread computes a portion of the saxpy operation on multiple chunks.

(2)

Threads	Input Size	V1	V2
4	10^4	0.000510s	0.000469s
8	10^4	0.001366s	0.000927s
16	10^4	0.001556s	0.001310s
4	10^6	0.002648s	0.002933s
8	10^6	0.004679s	0.005803s
16	10^6	0.007432s	0.010590s
4	10^8	0.323773s	0.494746s
8	10^8	0.216009s	0.503037s
16	10^8	0.271725s	1.151922s

Chunking outperforms the serial version in for each different input sizes and threads, but striding gets outperformed by the serial version only for 10^8 input size and 16 threads. In general, larger input sizes may benefit more from multi-threaded implementations, as there is more potential for parallelism.

From the table, we can notice the disparity of the performance with the two versions we implemented. Striding out performs chunking for most input sizes and threads because it better allocates the resource but chunking can outperform striding in very large input sizes(10^8) because it can minimize the overhead associated with creating and synchronizing threads.