Jason Gallagher

Steven Kravitsky

**SSE Assignment 1**

This assignment was parallelizing Gaussian Elimination using SSE. SSE is part of the x86 instruction set that allows parallelization of multiple data points through a single instruction. Each SSE register can hold up to 16 bytes of data, or four 4-byte floating point numbers. This could in theory speed the process up by 4x. In out implementation of the code we are iterating through all of the rows of the matrix in one big for loop. We check to see if the element we are working on is 16-byte aligned so that we can perform our SSE parallelization, this happens in both the division and elimination step. An example of how we are doing this for the division step can be seen in figure 1. The data can then be loaded into the SSE registers in order to perform the division step. If the number of columns is not divisible by 4 we have to move back and go through the elements one at a time. This can be seen in figure 2. The same checking as figure 1 is done in the elimination step. The data is then put into the SSE registers to be multiplied and subtracted in order to perform the step. This can be seen in figure 3.

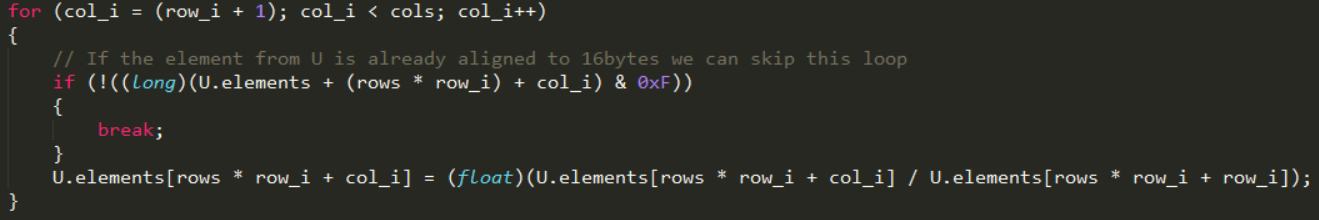
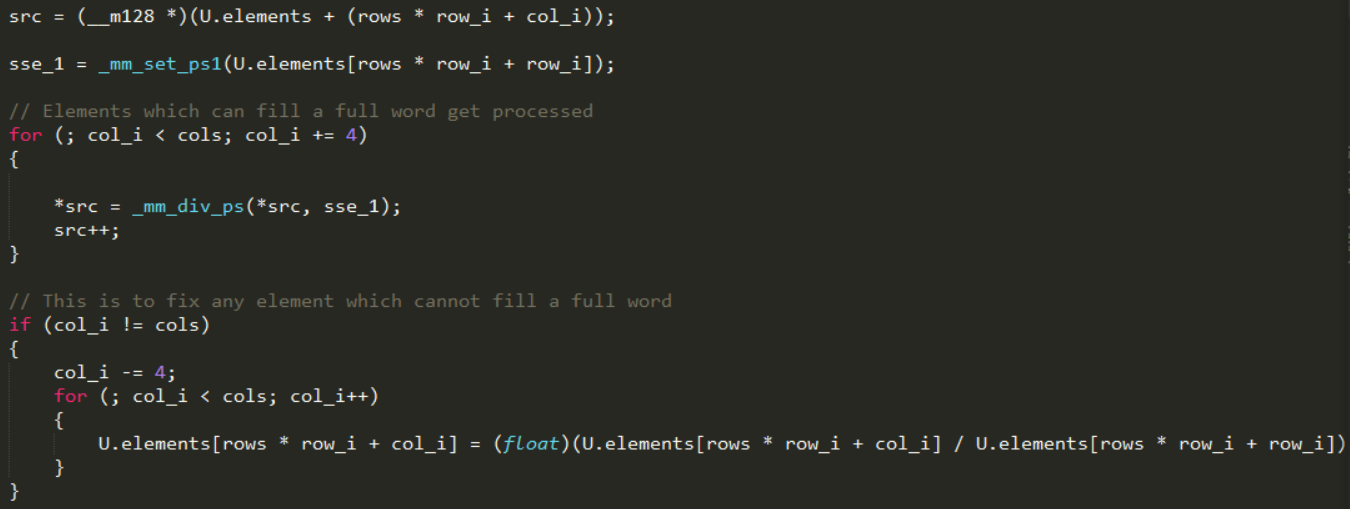


Figure 1: Checking to see if the element is 16-byte aligned

 Figure 2: Loading to the SSE registers, also checking to see if there are more columns than are divisible by 4

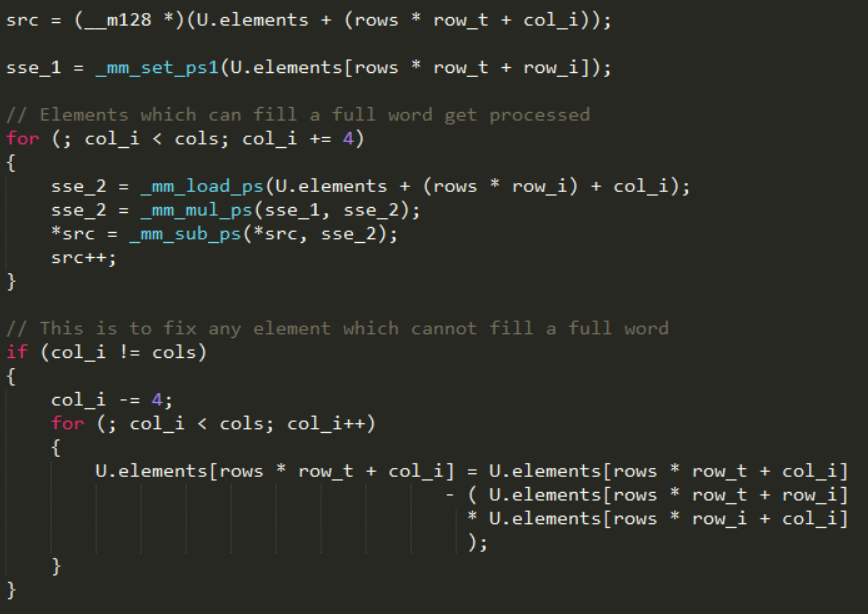


Figure 3: Loading to the SSE registers, also checking to see if there are more columns than are divisible by 4

As far as the performance gained from the SSE implementation we saw an improvement of about 3.07x on average as compared with the serial implementation. The biggest speedup we achieved was 3.13 times and the slowest 3.03x. The timing results for each of the ten trials can be found in figure 4 and figure 5.

Figure 4: Timing results graph

|  |  |  |  |
| --- | --- | --- | --- |
| Trial | Serial | SSE | Speedup |
| 1 | 3.83 | 1.22 | 3.139344 |
| 2 | 3.86 | 1.27 | 3.03937 |
| 3 | 3.94 | 1.27 | 3.102362 |
| 4 | 3.88 | 1.27 | 3.055118 |
| 5 | 3.91 | 1.26 | 3.103175 |
| 6 | 3.87 | 1.27 | 3.047244 |
| 7 | 3.91 | 1.28 | 3.054688 |
| 8 | 3.88 | 1.26 | 3.079365 |
| 9 | 3.92 | 1.28 | 3.0625 |
| 10 | 3.88 | 1.27 | 3.055118 |
| Average | 3.888 | 1.265 | 3.073828 |

Figure 5: Timing results