Abdallah Nawras
Project#4
Vectorized Array Multiplication/Reduction using SSE

Machine
The program is executed on an Intel i7 powered CPU having two physical cores and 4 threads peaking at 2.4GHz.

Populating Arrays
Arrays are populated using nested for loops with floating operations up-to **length** of array.

Array is filled with values as following,

```
for(int i = 0; i< length; i++)
{
	a[i] = (float)length/(i+1);
	b[i] = (float)a[i]/(i+1);
}
```

which generates values like below,

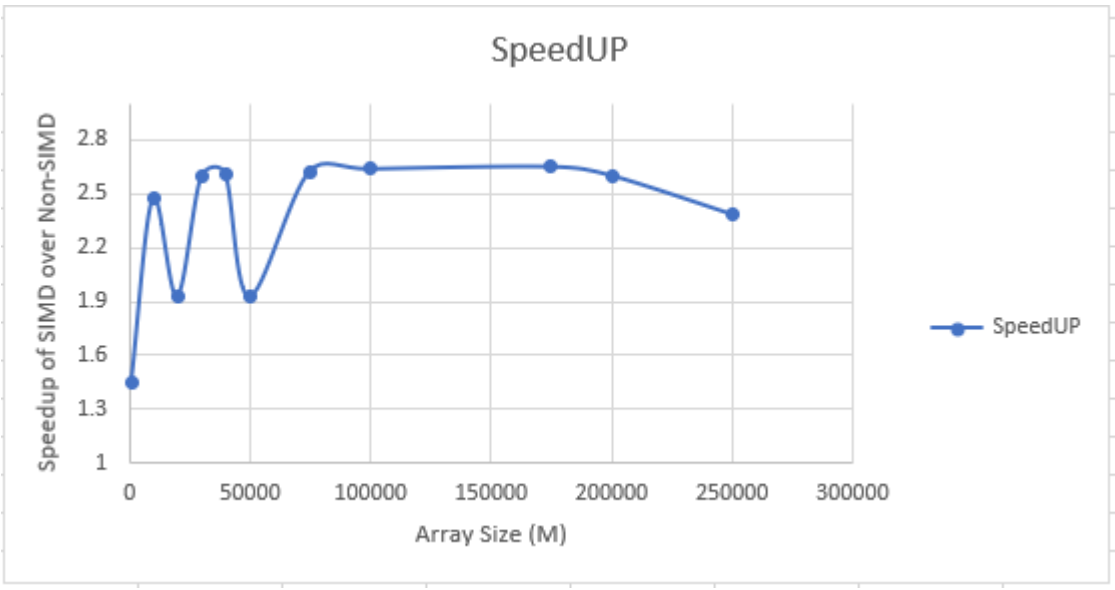
```
a[0] = 100.000000, b[0] = 100.000000
a[10] = 9.090909, b[10] = 0.826446
a[25] = 3.846154, b[25] = 0.147929
a[50] = 1.960784, b[50] = 0.038447
```

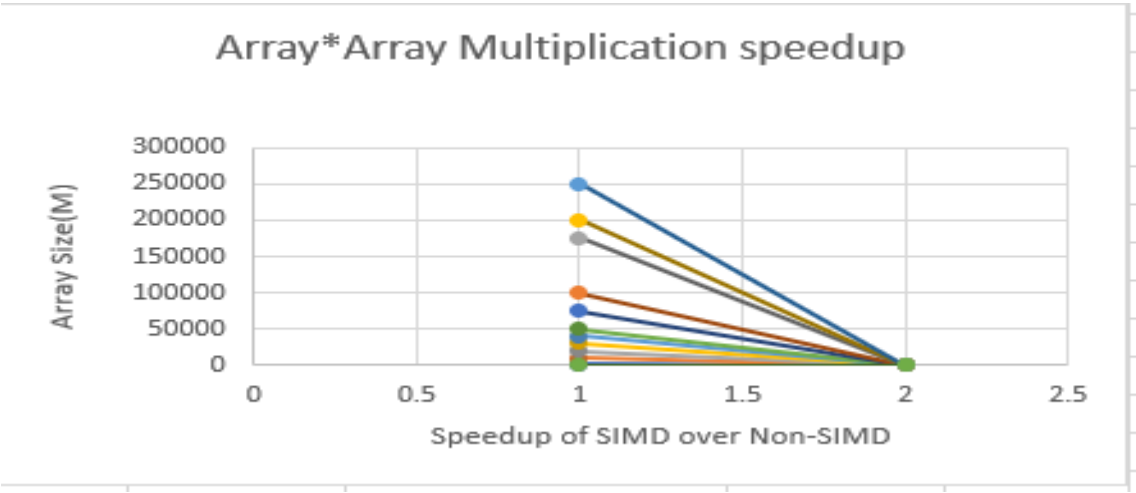
Table of performances

Array Size	Time SIMD	Time non-SIMD	SpeedUP
1000	0.001638	0.004232	1.454142
10000	0. 1.66481	0. 0.417267	2.473754
20000	0. 0.64143	2.551173	1.926828
30000	0. 1.449651	3.779724	2.592601
40000	2.59067	6.661285	2.609908
50000	3.965711	10.504781	1.927488
75000	9.067076	23.682929	2.624269
100000	15.752192	41.874829	2.63715

175000	48.60547	128.960678	2.649529
200000	63.734359	168.690496	2.600725
250000	98.982567	262.911489	2.385564

Graph: SpeedUp versus Array Size





Findings.

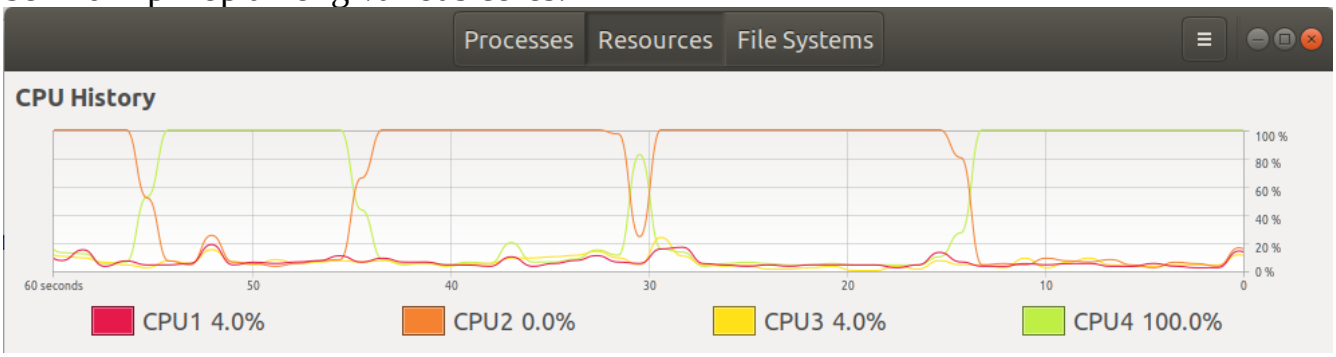
- Data and graph shows the SpeedUPs differences are minor for small array sizes.
- SpeedUPs difference is large for the larger arrays.
- Even though the SpeedUP with SIMD array operations grow with the increase in array size though this growth is not linear. There are some larger array sizes with smaller

SpeedUP values as compared to relatively smaller ones. This is most probably due to context switching cost between CPUs as observed below.
- Smaller non-SIMD speeds are equivalent to the SIMD counterparts most probably due to setup times.
- Likely the CPU registers can accommodate smaller array sizes, thus this smaller setup time makes the small array-data sizes to appear having better speedups.

Further Observations

Core Utilization

An overview of System performance monitor on Linux computer shows CPU switches between various CPU cores / threads during program execution. Most probably an algorithm as a heat mitigation strategy on either hardware or software side is the reason behind flip-flop among various cores.



CPU Temperature

This program can potentially serve as a foundation for CPU benchmark and CPU stress test application due to higher operating temperatures during program execution. Temperatures upto 93 C were (~200 F) noted.