Exercise 1 What hardware am I using?

Model: Intel i5-7200U Normal clock rate: 2.5 GHz Boost: 3.1GHz 2 Cores 2 Threads per –> 4 CPUS

L1d (data) Cache: 64 KiB / 32 per core (1 KiB = 1024 byte) Lii (instrution) Cache: 64 KiB / 32 per core L2 Cache: 512 KiB / 256 per core L3 Cache: 3 MiB (1 MiB = 1024 KiB)

Theoretical max bandwith: DDR speed (MHz) * Memory Bus width (B) * No. of Channels 2133 * 4 * 2 = 1.7 GB/s

Vectorization: sse, sse2, sse4_1, sse4_2, ssse3 avx, avx2

Exercise 3 Compiler Flags

For explicit results see (results/results.json.

In general: - Flags `-O2` and `-O3` optimize away the Empty-`LoopExperiment` and the `AccumulateExperiment` resulting in a runtime of 0.0s.

For `-O0`: - The Empty-`LoopExperiment` is generally (indepently of flag combintations) faster compared to the other Experiments, althoug sometimes `AccumulateExperiment` is faster, which can be explained by the simplicity of the problem and the fact that the CPU is not 'pinned' to the specific task.

For `-O2` and `-O3`: - The flag `-funroll-loops` increases the runtime of the `FillVectorExperiment` which could indicate that forcing the compiler to unroll the loops is counter prodictive as it 'negates' the optimizations of the `-O2`/`-O3` flags.

For `-ffast-math`: - This flag does not benefit any of our experiments, as we do not have any sorts of math operation that is affected by fast math. (It usually only affects IEEE-754/floats operations)

For `-march=native`: - Negligible differences in runtime for most experiments, probably again, the problem is too simple to benefit from the flag to notice a difference.

Exercise 4 Parallel daxpy Routine

See code daxpy.cc.