CPE 315 Computer Architecture
Lab 4
Daniel Dohnalek
Jasmine Patel

Data Machine 1 (Daniel):

	O0	O1	O2	O3	Unroll 2	Unroll 4	Unroll 8
Average CPI	1.27900149	1.25987025	1.25925082	1.27623751	1.30356324	1.32706709	1.33875932
Cycles	1416739204	1395782853	1395110307	1413926202	1375858379	1365861781	1360121660
Instructions	1107691587	1107878256	1107889122	1107886412	1055459629	1029233402	1015956886
Branch Misses	6762498	6779064	6787290	6722839	6750874	6793277	6805512
Runtime (measured)	1.826077	1.8083	1.7967	1.845	1.76115	1.76682	1.72133130
Runtime (equation)	1.28677493	1.26774101	1.26713015	1.28421998	1.24964430	1.24056474	1.23535119

Fraction of time executing matadd with O1: 88.24%
Fraction of time executing matadd with loop unrolling 2: 77.78%
Fraction of time executing matadd with loop unrolling 4: 79.49%
Fraction of time executing matadd with loop unrolling 8: 81.85%

Data Machine 2 (Jasmine):

	O0	O1	O2	O3	Unroll 2	Unroll 4	Unroll 8
Average CPI	1.2994282	1.31210687	1.29727756	1.29784957	1.33212353	1.34258876	1.36253099
Cycles	1385064545	1398571416	1382773933	1383379664	1378212458	1367812903	1377312246
Instructions	1065904310	1065896748	1065890445	1065909139	1034598307	1018478771	1010854854
Branch Misses	6736831	6748504	6754686	6729875	6,795,396	6796894	6722488
Runtime (measured)	1.74571113	1.79762691	1.81875224	1.78969501	1.77698610	1.81098491	1.78932576
Runtime (equation)	1.25800595	1.27027376	1.25592546	1.25647562	1.25178242	1.24233690	1.25096500

Fraction of time executing matadd with O1: 76.32%
Fraction of time executing matadd with loop unrolling 2: 84.85%
Fraction of time executing matadd with loop unrolling 4: 77.78%
Fraction of time executing matadd with loop unrolling 8: 77.14%

E-Value calculations:

Amdahl's Law
E = runtime(non-unrolled) / runtime (unrolled)

1.79762691 / 1.77698610 ≈ 1.01 (unrolled 2)
1.79762691 / 1.81098491 ≈ 0.99 (unrolled 4)
1.79762691 / 1.78932576 ≈ 1.00 (unrolled 8)

The matrix size is 1024 x 1024. By using Amdahl's Law, we can tell that the unrolling difference is very small. Even though it seems like we are making it much faster, the only thing we are reducing is the number of comparisons. The number of adds, lds, and strs remain the same. Therefore, there is not a significant difference in runtimes of unrolled and rolled loops.

Observations

Compiling the same assembly code with different optimization levels didn't appear to significantly impact the number of cycles, instruction count, or runtime of the resulting program. All optimization levels (O0 through O3) showed practically identical performance numbers. However, manually loop unrolling did have an impact on the number of instructions executed by the computer. The greater the loop unrolling factor, the fewer instructions were executed. Despite this, there was little impact on the runtime of the program.

The measured runtime was significantly higher than the runtime calculated by the performance equation. This is likely because the performance equations don't take into account the assorted flags in our makefile and the overhead of counting clock cycles and collecting performance statistics. For example, the -g flag decreases runtime for the purpose of debugging.