Performance and Optimization of Graph Algorithms on x86 and ARM Based Processors

Benjamin Swann, Jesus Rivera III, Tomas Vasquez

Texas State University- CS3339 Computer Architecture

10/13/2021

J.Rivera: j_r1470@txstate.edu

B.Swann: bts74@txstate.edu

T.Vasquez: t_v39@txstate.edu

Table of Contents

Abstract

The performance of AMD x86_64, Intel x86_64, ARM 32 bit, and ARM 64 were tested using a graph sorting algorithm. This program was compiled locally on each machine using -O0 and -O3 optimization. Both optimization versions of the program were used to test the performance of each machine. The results recorded from each test were: power usage(watts), real time, user time, system time, instruction count, and cycles.

Performance and Optimization of Graph Algorithms on x86 and ARM Based Processors

Introduction

This project's goal is to decern the difference in time and power-time efficiency for a given program between different systems. The systems have a range of architectures, clock speeds, cores, and amounts of ram. The time a program takes to execute is important, but how much power it used is also important. If a system is marginally faster than another, but uses 100 times as much power, then that system is not much of an upgrade for running a program.

Performance can also be improved by optimizing a program during compiling. As such, another metric that will be considered is the performance when compiled with O3 compared to the default compilation.

The original scope of this test also included running the same test, but using older versions of GCC to preform compiling, to see if updated versions of the compiler would result in any performance gain, this was dropped however to accommodate the power measurements.

Questions/Goals

- Performance of program using different machines
- Performance using various levels of optimization during compiling.
- Runtime vs power consumptions of each device etc...

Predictions

Jesus Rivera III- I believe that the biggest impact on performance will be the clock speed of a given machine. I do not think that the amount of Ram nor the type of Architecture will have too big of an impact on the run time or power usage.

Benjamin Swann- Because performance is calculated with the formula: CPU time = Instructions * CPI * Clock Cycle Time and CPI = cycles per instruction, I also think that the

processor speed will have the biggest impact on the performance due to the number of instructions staying constant.

Devices

Spec	Intel Laptop	AMD DESKTOP	PI 3	PI 4
Architecture	X86_64	x86 64bit	ARM 32bit	ARM 64bit
Processor Speed	3400MHz	4617.33 MHz	600MHz	1500MHz
Operating System	Ubuntu 20.04.1	Ubuntu 20.04.1	Ubuntu Server 5.11.0	Ubuntu Server 5.11.0
G++ Version	9.3.0	9.3.0	10.3	10.3
RAM	8GB	32GB	1GB	8GB
Cores	2	6	4	4
CPU	Intel I7 6600i	Ryzen 5 3600	BCM2835	BCN2835

Table 1: Systems used in Testing

Performance Tools

Performance tools utilized were the “perf” command to get the runtime values of the program, and the “powertop” suite (Intel, n.d.) to get the power usage of program. Powertop is an Intel open source developed tool used to log and monitor the power usage of programs and running task in a Linux environment.

Benchmarked Code

The program used was created by T. Vasquez (Vasquez, n.d.). The program gets passed an adjacency matrix that is either an undirected, weighted, or a directed unweighted graph. It then performs a Breadth First Traversal and Depth First Traversals. It adds vertices, deletes them, and adds an edge between two distinct points. If such a point already exists, it does not place one. If an edge gets removed, it will disconnect the graph. We have accounted for that in the program by counting and listing them at end of program.

Procedure

1. Benchmark code downloaded to machine
2. Benchmark code compiled locally using O0 and O3
3. Powertop installed on machine (sudo apt-get install powertop)
4. Run powertop
5. Run program using perf to record time, cycles, and instructions in a separate terminal
6. Log power used by program from powertop

Numerical Data

Pi3

Metric	O0 Optimization	O3 Optimization
Power	1.53W	1.44W
Real	8:46	4:22
User	2.47s	2.28s
System	8:41	4:04
Instructions	270799200553	3597057310
Cycles	313506599059	5369980243

Table 2:PI3 Results

Pi4

Redirected output is routing program output to a file instead of to the console.

Metric	O0	O3	O3 Redirect output	O0Redreicted
Power	1.55W	1.55W	0.5W	0.45W
Real	49.64s	48.9 s	2.258 s	4.87 s
User	0.61s	0.4 s	0.44 s	1 s
System	48.85s	48.34 s	2.05 s	3.87 s
Instructions	1.31194E+11	1.29608E+11	3518539553	3721775058
Cycles	7405660948	72998575860	3660325325	3867289491

Table 3: PI4 Results

AMD

Metric	O0	O3
Power	53.3mW	14.3mW
Real	0.465 s	0.449 s
User	0.064 s	0.077 s
System	0.266 s	0.206 s
Instructions	1137404611	909623495
Cycles	1181741190	976427479

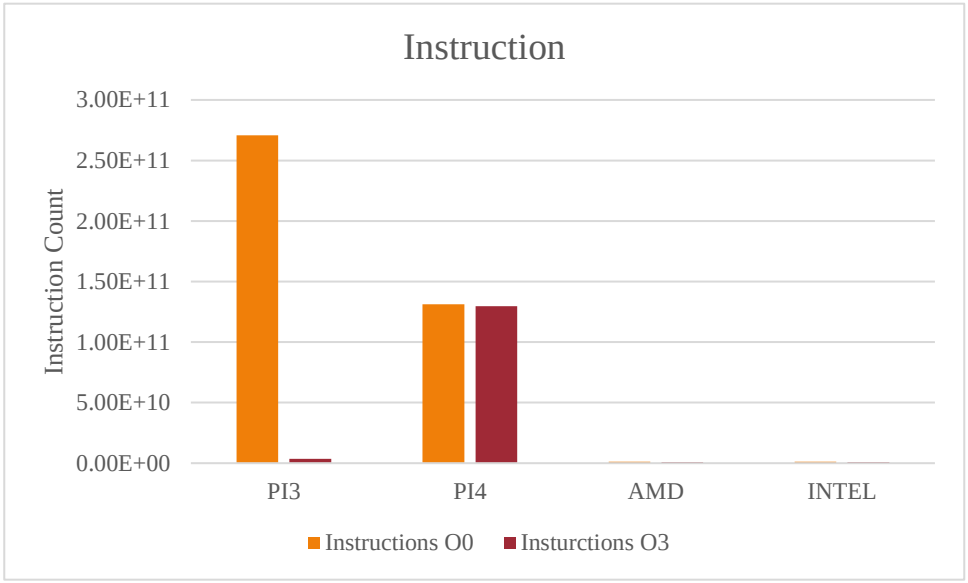
Table 4: AMD Desktop Results

Intel

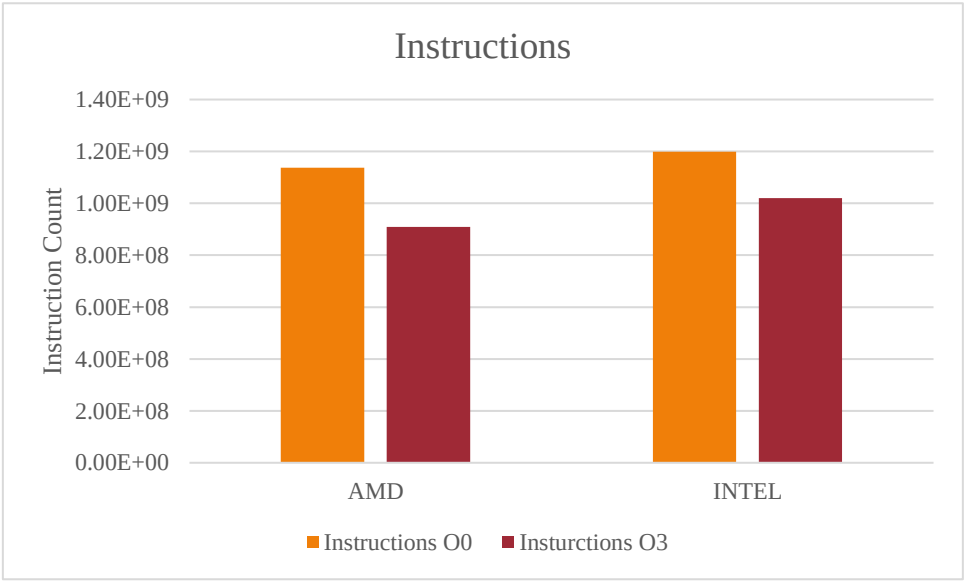
Metric	O0	O3
Power	7.34mW	11.5mW
Real	0.767 s	0.725 s
User	0.191 s	0.201 s
System	0.336 s	0.29 s
Instructions	1199351032	1020535979
Cycles	1623351604	1474991922

Table 5: INTEL Laptop Results

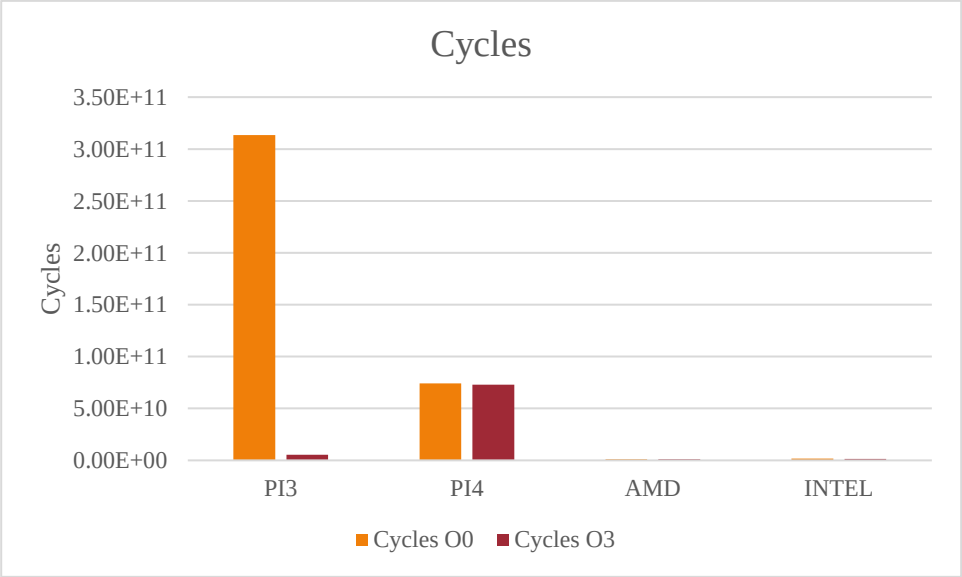
Graphs



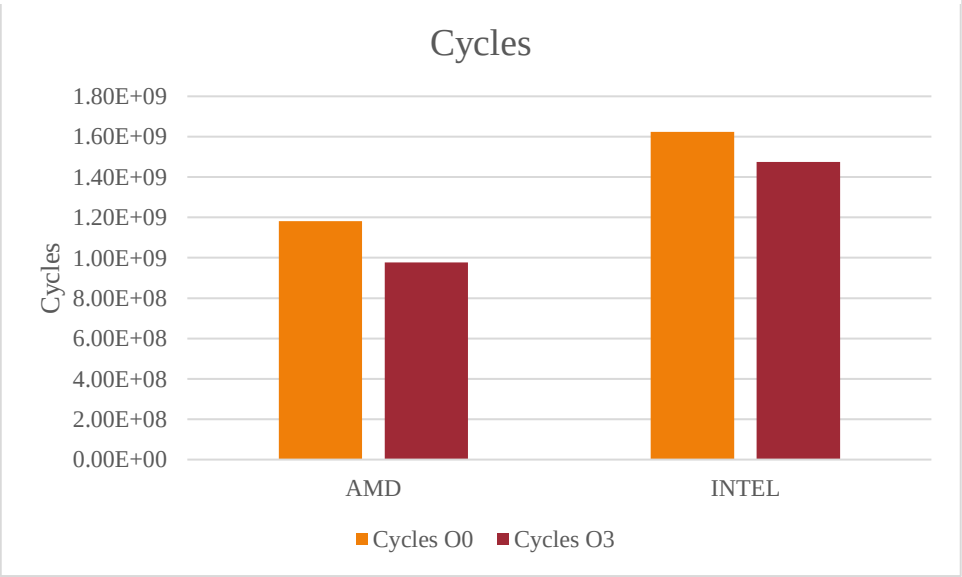
Graph 1: All devices Instruction count



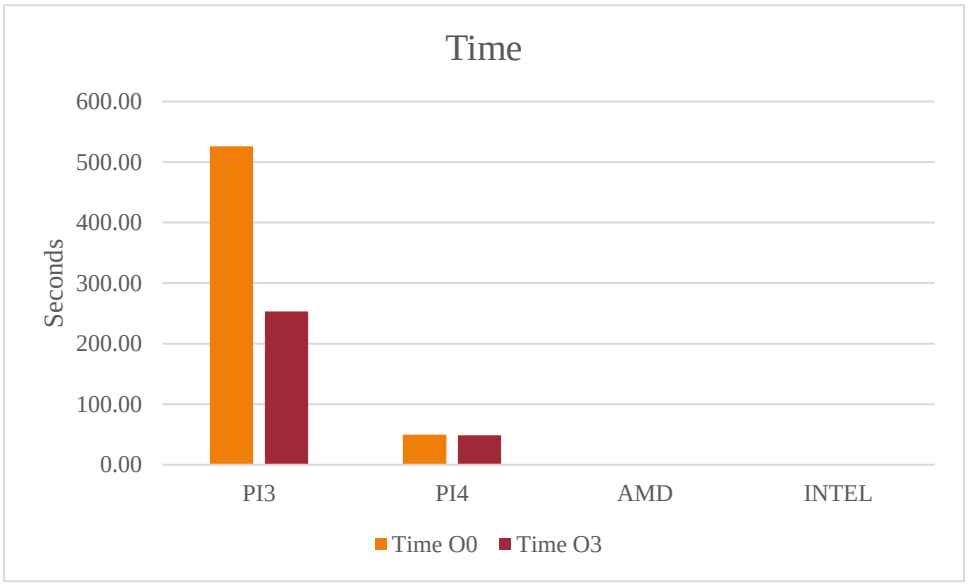
Graph 2: Closer look at AMD and INTEL Instruction count



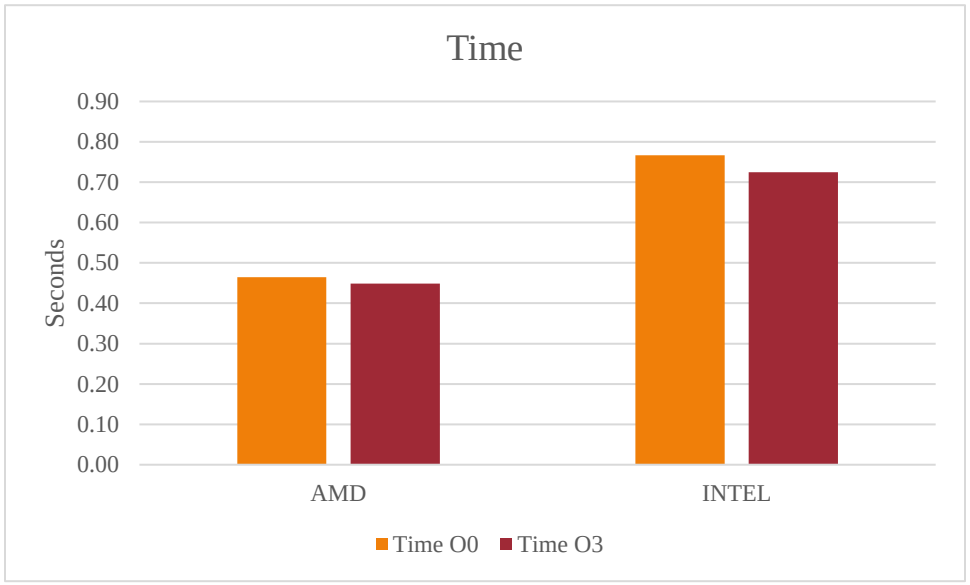
Graph 3: All Devices Cycle count



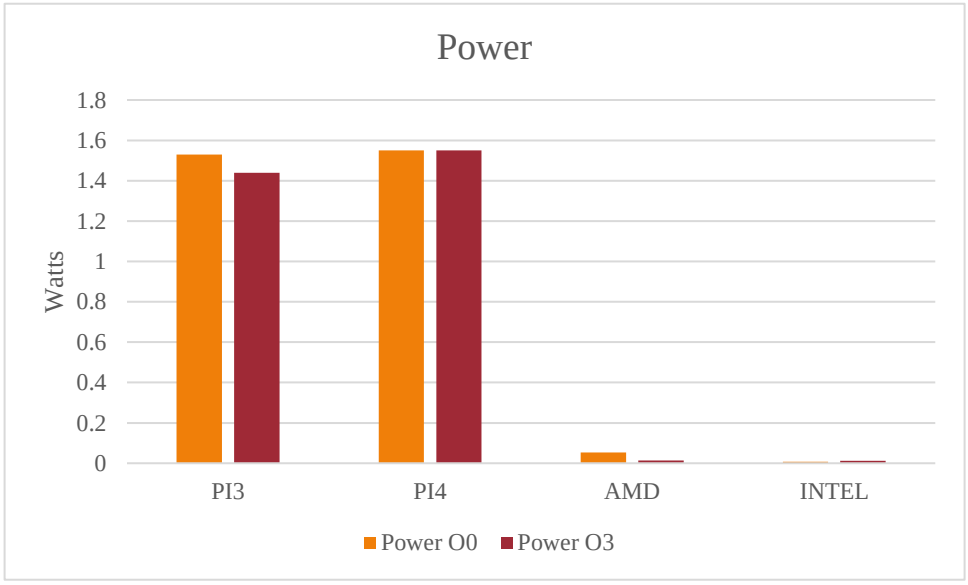
Graph 4: Closer look at AMD and INTEL Cycle count



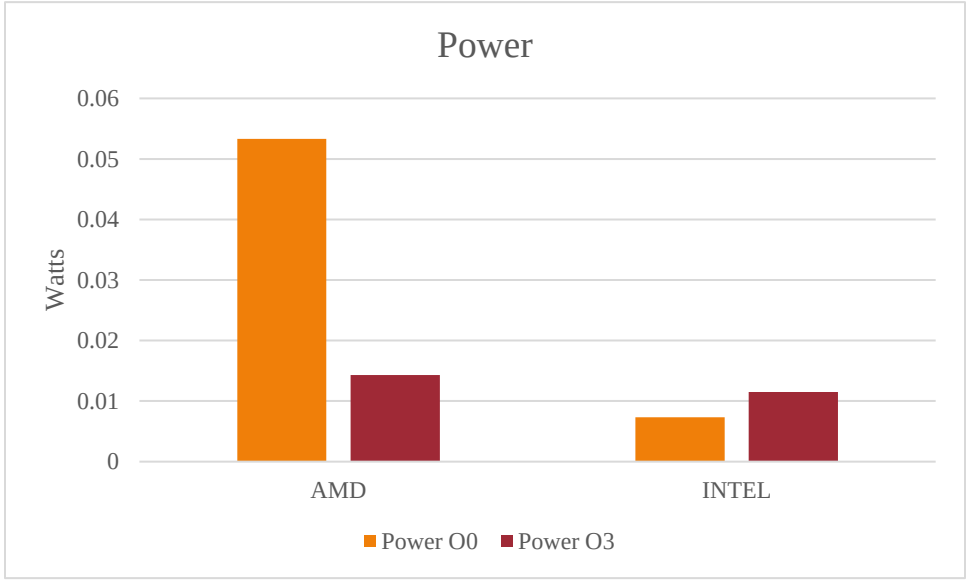
Graph 5: All devices Real time



Graph 6: Closer look at AMD and INTEL Real time



Graph 7:All devices power usage



Graph 8: Closer look at AMD and INTEL power usage

Analysis

O0 vs O3

PI3

The PI3 saw a large improvement using the compile time optimization flags. The program had a speed up of 2.08, the largest speed up due to optimization of all tested systems, a 98.7% reduction in instructions, and a 98.3% reduction in clock cycles. The optimization doubled the performance of the program however the power usage remained nearly the same with a reduction of only 5.9%. Using compile time optimization is very useful in reducing the time to execute the program, but not very effective in reducing the power required by the CPU.

A interesting thing of note that we found during testing, but not recorded in data, was outputting the console output directly to a text file instead. Doing this saw the real time of the program reduced to under a minute and might make for future study the time and power requirements for outputting text to the consol.

PI 4

The PI4 saw a very insignificant improvement using the compile time optimization flags. The program had a speed up of 1.015, the smallest speed up due to optimization of all tested systems, a 1.2% reduction in instructions, and a 1.4% reduction in clock cycles. The optimization did not affect the power utilization.

Additional Test, Console Output Redirect

We preformed 2 additional tests on this system. The previously mentioned redirecting of console output to a text file. These two tests do not have bearing on the original goal of this research, but we are including the analysis on them anyways as

they are of interest and have a significant improvement in performance and power usage. Data for these tests are included in table 3 as *O0Redirected output* and *O3Redirected output*.

The two additional test preformed were using the same criteria for the previous test but having the output of the program be loaded into a text file instead of being displayed on the console. This resulted in significant increase in performance. Table 6 clearly shows that by redirecting the output of the program to a text file, power usage and time to run are significantly reduced, and that by using optimization and redirection power usage and run time can be reduced by an absurd amount.

Output Redirected Analysis								
	Power	**%Reduced**		**Power**	**%Reduced**		**Power**	**%Reduced**
O0	1.55	71.0%	**O3**	1.55	67.7%	**O0Redirected**	0.45	-11.1%
O0Redirected	0.45		**O3Redirected**	0.5		**O3Redirected**	0.5	
	Time			**Time**			**Time**	
O0	49.664	89.9%	**O3**	48.9	95.4%	**O0Redirected**	5	54.8%
O0Redirected	5		**O3Redirected**	2.258		**O3Redirected**	2.258	
	Instruct			**Instruct**			**Instruct**	
O0	1.312E+11	97.2%	**O3**	1.3E+11	97.3%	**O0Redirected**	3.72E+09	5.5%
O0Redirected	3.722E+09		**O3Redirected**	3.52E+09		**O3Redirected**	3.52E+09	
	Cycles			**Cycles**			**Cycles**	
O0	7.406E+10	94.8%	**O3**	7.3E+10	95.0%	**O0Redirected**	3.87E+09	5.4%
O0Redirected	3.867E+09		**O3Redirected**	3.66E+09		**O3Redirected**	3.66E+09	

Table 6: Additional Test Results

Possible Source of Error

The results of the optimization of the program on the PI4 might not have been done correctly as the results show the results of O0 and O3 being nearly identical while all other systems had some form of improvement due to optimization. Another indicator that there may have been a mistake while testing is the fact that the redirected test had a significant reduction in instruction and cycle count, and the redirection is done at run time and not compile time. The fact that the O3 doesn't

have a significant reduction in those areas indicates that the O3 test might have accidently been run using the O0 flag.

AMD Desktop

The AMD Desktop saw a small improvement using the compile time optimization flags. The program had a speed up of 1.036, a 20.0% reduction in instructions, and a 17.4% reduction in clock cycles. The optimization did increase the performance of the program but only by a very small amount and in ways not related to time. However, the power usage was greatly improved by reducing the amount of power used by 73.2%. On the AMD system, the optimization does not seem to be very effective at reducing the time to run the program, but it is useful at reducing the amount of power the program needs to be run.

INTEL Laptop

The Intel laptop saw a small improvement using the compile time optimization flags. The program had a speed up of 1.058, a 14.9% reduction in instructions, and a 9.1% reduction in clock cycles. The optimization did increase the performance of the program but only by a very small amount, and not really in anyway related to time. The optimization did also seemed to increase the power usage by 56.7%. Due to not really having any effective change in performance, and increasing the power consumption, optimization is not very useful on the Intel system (within scope of this project).

System Vs System

The systems that are of major interest of comparing are the PI 3 vs. PI4, as that will help show the difference in 32 bit systems and 64 bit systems of the same architecture. The PI 4 vs. Intel Laptop, as that compares two 64 bit systems with the same amount of RAM and different Architecture. The AMD Desktop vs. Intel Laptop as that helps show if having a significant more amount of Ram in the same architecture has any large impact on performance.

PI3 vs PI4

The results of testing show that the PI3 by far had the longest execution time and was the most power inefficient. The 600MHz clock speed of the PI3 is 2.5 times less than the clock speed of the PI4's 1500Mhz. This would lead to an expectation that for the same program the PI4 should only be 2.5 times faster than the PI3. This was not the case , the PI4 was in fact 10.6 times faster than the PI3 for O0 and 5.2 times faster for O3. This discrepancy could have a few roots. The two systems have different amounts of RAM, as well as being 32bit system vs a 64bit system.

PI 4 vs INTEL Laptop

The result of testing shows that there is a speed up of 64.7 for O0 test and a speed up of 67.4 for O3 test. The Intel laptop has x86 64 bit architecture while the PI 4 has a ARM 64 bit architecture. Both systems had 8 GB of memory but the Intel has only 2 cores while the PI has 4 cores. The Intel has a clock rate of 3400MHz, 2.27 times as fast as the 1500MHz of the PI 4. This would lead to the idea that the Intel would have a program speed about 2.3 times faster than the PI 4. This is again not the case though.

The speed up of the Intel compared to the PI4 is mostly attributed to the diffrence in architecture. The fact that the Intel has 2 less cores and still had such a significant difference in performance indicates that the number of cores does not have a significant impact on the performance of the program being used as a benchmark. The power usage of the Intel was also significantly less than the PI4. The O0 test had the the Intel use 99.5% less power than the PI and the O3 test use 99.3% less power. Some possible reasons for this drastic diffrence in power usage could be simply due to the physical design of the laptop. In general laptops are designed to use the least amount of power possible for a given task as to maximize battery life, which could be a leading cause to the small power usage of the Intel processor. It is worth noting though that the power measurement is solely that of the CPU, not of the entire system, thus as a whole is its possible that the Intel Laptop could use more power as a complete system when compared to the PI 4.

AMD Desktop vs. INTEL Laptop

The AMD Desktop and Intel laptop had very similar performance even though they are vastly different system both in form factor and design. The AMD desktop was had better performance in both tests, but also used more power in both tests. The AMD has a clock rate of 4614MHz, 1.35 times faster than the 3400MHz of the Intel Laptop. This indicates a expected speedup of about 1.35. The real speedup of the AMD to the Intel was 1.65 for the O0 test and 1.26 for the O3 test. Unlike the other comparisons, the speed up is roughly what was expected. This is believed to be because while the systems might have different CPU manufacturers, the Architecture for both is the same, as well as both being 64-bit systems.

The Intel used 86.2% less power than the AMD for the O0 test and 19.6% less for the O3 test. A note of interest is that though the AMD used substantially more power than the Intel, it still used a vastly less amount than either of the PI's.

ARM vs x86_64

Though the comparisons and test of all the systems, the x86_64 architecture is a much better architecture in terms of power usage and execution time (at least as far as the benchmark program goes). Though the x86 architecture had much better performance and power usage in general, the compile time optimization did not have much effect on either. The ARM architecture, however, did see a massive increase in performance using the compile time optimizations.

Conclusion

The test preformed reviled that a 64-bit system (PI 4) performed much better than a system with the same architecture as a 32-bit system (PI 3), and that ARM systems use more power for the same task than a x86 system. ARM systems see much better results using compile time optimization than x86 systems, which in some instances saw an increase in power consumption. Overall the x86 systems had the best performance and the least power usage for the benchmark program, and was better than the ARM systems in all metrics.

References

Intel. (n.d.). Retrieved from https://github.com/fenrus75/powertop

Vasquez, T. (n.d.). *Graph*. Retrieved from https://github.com/Rhino0311/arch_proj/tree/main/graph