CS325 Project Report: Low Power Processor Designs & Performance

Team #4: Kelcee Gabbard, Cassandra Priddy, Daria Casey, and Devon Wilkening

Abstract

This project aimed to compare the performance of a low power processor system, the Raspberry Pi 4 Model B, to a standard/high power processor system. Our team conducted performance tests of five parameters: response time, CPU usage, memory usage, disk I/O speed, and power consumption. Our testing methodology emphasized pushing the limits of the lower power system to best gauge its limitations and measuring the higher power systems' performance under the same conditions. The results consistently followed predictions that the higher power processor system would have significantly higher performance rates than the low power processor system and deeper analysis confirmed that the processor itself proved to be the most impactful on CPU usage performance in our test data.

Introduction

Computer processor design has long been a delicate balancing act between maximizing computational capabilities and minimizing power consumption. Due to this, there exists a multitude of system designs which prioritize either side of this equation. This research project seeks to compare the performance of a low power processor design, one that prioritizes energy efficiency and reduced power consumption, to a higher/standard power processor design, which emphasizes computational capabilities and overall performance with a higher power consumption in comparison. The two devices we used for comparison are a Raspberry Pi 4 Model B as the low-power processor system and a Linux desktop featuring an Intel i9 10900k CPU as the higher-power processor. Five testing parameters were explored: response time, CPU usage, memory usage, disk I/O speed, and power consumption. Our methodology focused on stress testing both systems under the same conditions. This study intends to provide insight on the overall difference in performance between the two systems, as well as a more nuanced analysis of each set of test results.

Related Work

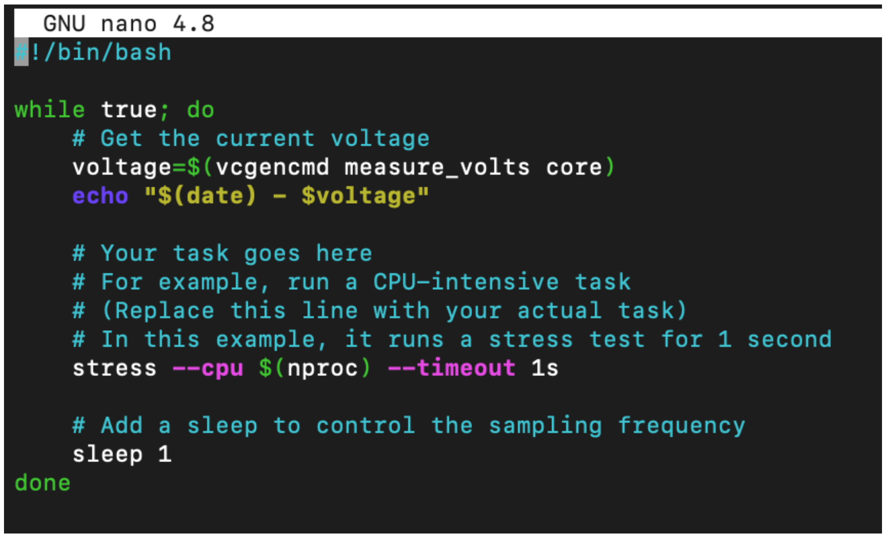
Previous works studied the tradeoffs of architecture and reduced power usage to discover where design is most efficient to create the best processor possible with the information gathered. In "Low-Power Processor Design," Ricardo E Gonzalez explores how design tradeoffs affect the power and performance of general processors, considering how performance should increase with the same amount of power, or normalize with less power [1]. To aid in testing design, "Energy Efficient Computing: A Comparison of Raspberry Pi with Modern Devices [2]," was the most helpful. In this paper, the authors prove that the Raspberry Pi can perform similar tasks to a modern device like a computer with less power making it more energy-efficient [2]. In a lecture given by Simon Segars, he explains why power consumption is an important design criterion for a microprocessor and explains how to use power metrics accurately to reflect the design of new microprocessor technologies [3]. In "ESL Solutions for Low Power Design," the paper writes about a framework the authors developed to employ power-saving techniques and low-power technique exploration that can be integrated into circuit design [4]. In a dissertation written for the 1998 international symposium on low-power electronics and design, J. P. Brennan, A. Dean,

S. Kenyon, and S. Ventrone write about their findings regarding low-power microprocessors for battery-powered devices and present their innovative power-reduction techniques [5]. In another lecture, Ruchir Puri et al., presents on the relationship between high performance and low power in the era of Big Data and shows designs that have succeeded in both areas [6]. Finally, from the International Journal of Advanced Networking Applications, "A Review on Low Power Testing Techniques," reviews and questions industry-standard testing techniques to see if they truly are a good measure of low power testing that can be reflected in circuitry design [7]. Each of these aided in research for the experiment presented here by allowing us to acknowledge whether our initial thoughts were shared and supported, and how to develop our performance tests to truly reflect how power consumption affects performance quality.

Methodology

In our experiment, Nginx was installed on a Raspberry Pi 4 and an Ubuntu desktop computer. On the Raspberry Pi, we ran the Nginx server, configured it, and accessed the server from another computer using the Raspberry Pi's IP address. From there, we installed a stress testing tool, ApacheBench, on both the Ubuntu desktop computer and the Raspberry Pi, and ran a command to send 10,000 requests to the Pi to check that everything was installed and configured correctly first. After testing the Pi, we tested the server on the Pi and desktop at the same time and ran the command to perform a different number of requests for the different devices. We compared the handling of 10,000, 100,000, 500,000, and 1 million requests with 10 as the number of requests on both the Pi and desktop and gathered the data into a table to create a graph of the response times. To test CPU and memory usage, htop was installed to collect that data. To stress test CPU usage, we used the command: "ab -n 1000 -c 10 http://localhost/."

We ran the same command to stress test memory usage but changed the number of connections from 10 to 1000. To test disk I/O on both systems, we installed iotop and ran the apache2-utils command to collect the necessary data. Upon trying to test for actual power usage, we were unable to find software that measures power usage accurately and we did not have access to a power meter to carry out this portion of the test. We attempted to write a script to output voltage, but because there is no way to measure current, we were unable to convert volts to watts using this method. The script is below:



```
GNU nano 4.8
#!/bin/bash

while true; do
    # Get the current voltage
    voltage=$(vcgencmd measure_volts core)
    echo "$(date) - $voltage"

    # Your task goes here
    # For example, run a CPU-intensive task
    # (Replace this line with your actual task)
    # In this example, it runs a stress test for 1 second
    stress --cpu $(nproc) --timeout 1s

    # Add a sleep to control the sampling frequency
    sleep 1
done
```

Once again, this method did not work so we were unable to measure power for the Raspberry Pi and Ubuntu desktop.

Experimental Results and Analysis

As stated in the proposal, we devised tests to measure website response time, CPU usage, memory usage, and disk I/O to quantify and analyze differences in performance between the low-power processor design of the Raspberry Pi 4 Model B and a higher power system, which we used a standard Linux desktop with ubuntu 20.4. Specifications of each device are as follows:

Hardware Specs	CPU	RAM (GB)
Raspberry Pi 4 Model B	Quad core Cortex-A72 (ARM v8)	2
Linux Desktop	Intel i9 10900k	16

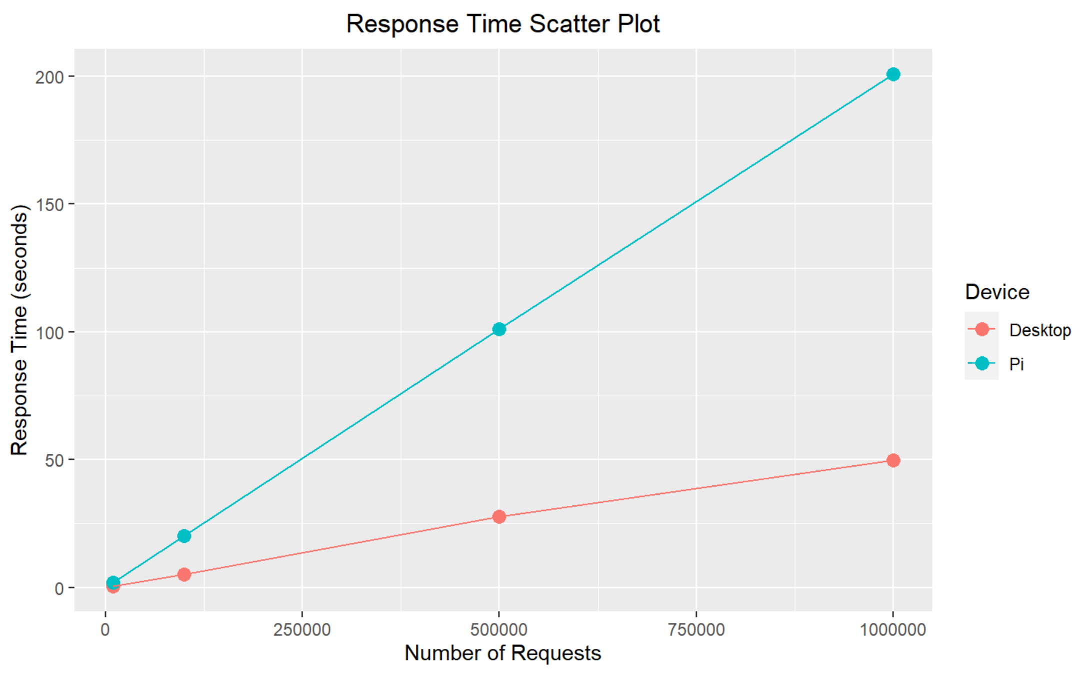
Throughout the following tests, these devices will be referred to as “Pi” and “Desktop” for convenience purposes.

Response Time Tests

The first set of tests run were on the response times of the web server, these tests served as a baseline comparison of the two systems before further testing. To run these tests, we used the previously initialized Nginx server, access to which was established via using another computer and the Raspberry Pi’s IP address. Stress tests of both systems were conducted using the “ab -n 10000 -c 10 http://localhost/” command, with “10000” referring to the number of requests, which were incremented to 100,000, 500,000, and 1,000,000 request in sequential tests. The number of connects, “10” in the command, were kept constant as changing them did not product relevant or consistent data for the purposes of this experiment and increased risk of confounding factors affecting data validity. Raw data is as seen below:

Time Taken for Tests		
Number of Requests	Desktop	Pi
10000	0.51	2.029
100000	4.991	20.162
500000	27.865	101.241
1000000	49.831	200.773

To further clarify results and data we used both excel and RStudio—an IDE for R, a programming language geared toward statistical computations and data analysis—for data manipulation and visualization. By using RStudio's visualization capabilities, we were able to gain a deeper understanding of patterns and trends present in our data. This is apparent with the generation of our Response Time Scatter Plot and Average Response Time Comparison bar graph below.



As is visible in the scatter plot above, there is a positive relationship between the response time in seconds and the number of requests made, this is to be expected as when the volume of requests increases it heightens demand for resources such as processing power, memory, or network bandwidth on the system. This also explains why the Raspberry Pi's response time increases at a markedly higher rate than the Desktop response time as it has less resources to leverage with a lower power processor, less memory, etc.

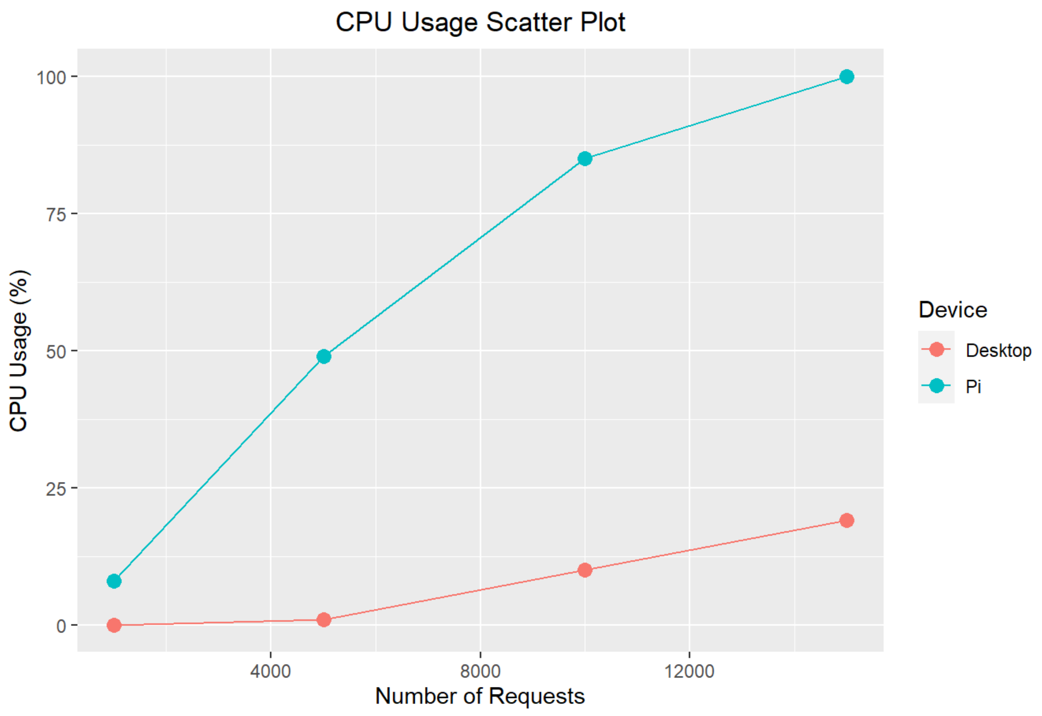
To summarize the response time tests results, desktop tests had a much narrower range throughout, capping out at approximately 50 seconds, and averaging at 10.8 seconds overall, while the Pi capped at approximately 200 seconds, roughly four times longer, with an average of 81.1 seconds response time across tests. On average, the Raspberry Pi's response time was 3.92 times higher than that of the Desktop's.

CPU Usage Tests

A Linux command-line utility, “htop”, was installed to conduct CPU and Memory tests as it allows for a real-time view of the system’s performance data. The command used for CPU usage tests was “ab -n 1000 -c 10 http://localhost/.” Raw test results are as seen below:

CPU Usage		
Number of Requests	Desktop	Pi
1000	0%	8%
5000	1%	49%
10000	10%	85%
15000	19%	100%

RStudio was again used to further analyze the data and for visual analysis. Below is a scatter plot of the results. As is apparent in the scatter plot, the number of requests to the web server and CPU usage have a positive relationship, most notably regarding the Pi because it has more limited resources, thus, is under more strain as the requests increase and the relationship is exaggerated compared to the desktop.



The number of requests parameters were altered from the previous test because the Pi maxed out at 100 percent CPU usage at just 15,000 requests, at which point Desktop had only reached 19 percent of its CPU usage capacity. On average, the Raspberry Pi’s percentage CPU usage was 17.7 times higher than that of the Desktop’s CPU usage.

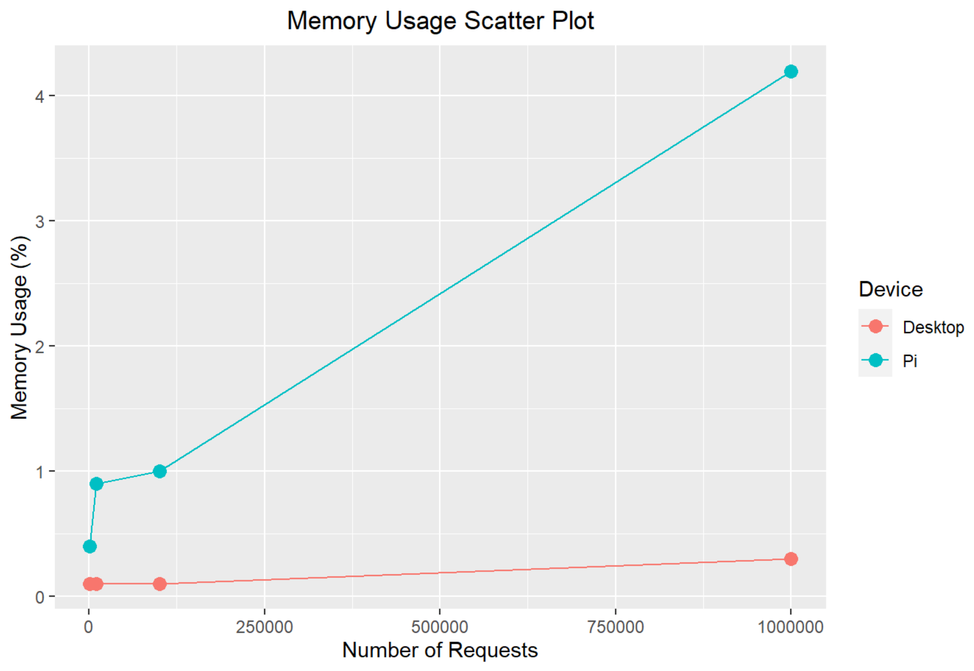
Memory Usage Tests

Similarly to the CPU tests, htop was used for memory usage testing. Connections were increased from 10 to 1000 for the memory test to improve data by testing more connections concurrently which is more likely to show limitations of memory in a system than only sending 10 requests at a time which doesn't put as much strain on the memory. Raw data for memory tests is as shown below.

Memory		
Number of Requests	Desktop	Pi
1000	0.1%	0.4%
10000	0.1%	0.9%
100000	0.1%	1.0%
1000000	0.3%	4.2%

Figure 8

Below is a scatter plot to make the trend of the data more visibly apparent. In addition, some calculations have been made to further quantify the proportional difference between the results for the memory usage tests. On average, the Pi's memory usage for the same number of requests is 9.25 times higher than the memory usage of the desktop device.



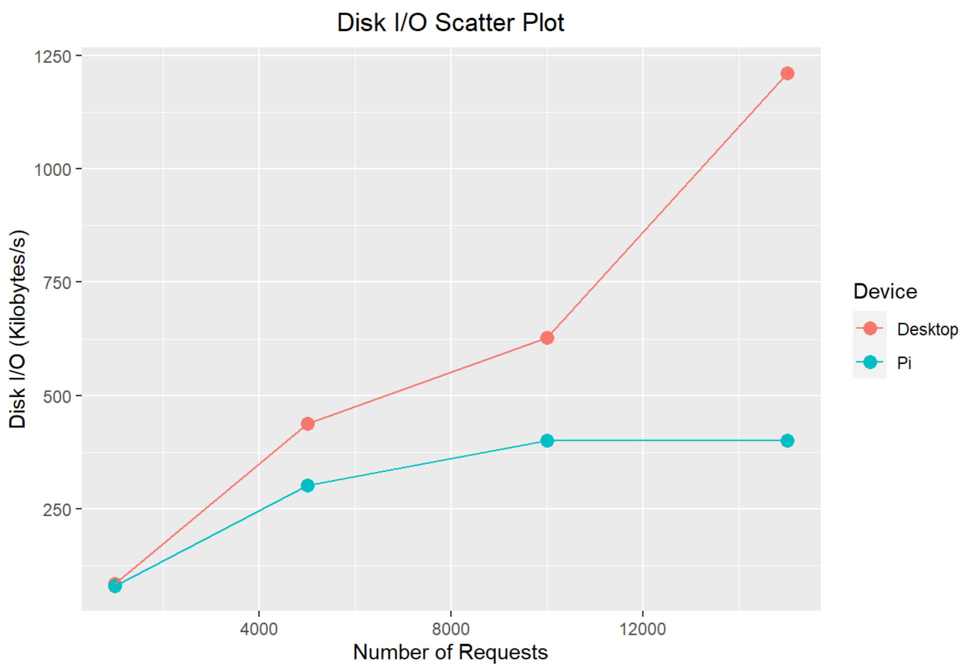
Predictably, there is a positive relationship between memory usage percentage and number of requests which is steeper for the Pi than the desktop device. This can be primarily attributed to the significant differences in their hardware specifications as the discrepancy in hardware capabilities directly impacts the limitations of the device, hence the Pi's memory usage climbs at a much faster rate while the desktop is essentially idling.

Disk I/O Tests

Disk I/O serves as a representation of efficiency of interactions between the processor itself and the storage; this can be a bottle neck in system performance, especially in those with low power processors. Thus, disk I/O was an essential component in comparing device performance. Tests were conducted via use of the iotop tool for visualization and apache2-utils for running the tests script itself. Raw data is as follows:

Disk I/O		
Number of Requests	Desktop (K/s)	Pi (K/s)
1,000	84.87	80
5,000	437	302
10,000	627	400
15,000	1212	400.93

As is apparent there is a positive trend between disk I/O speed and the number of results. In this series of tests, contrasting the previous tests parameters, the desktop device takes the lead. This is the case because unlike memory usage, CPU usage, or response time, in the case of disk I/O, a higher number is better because as it indicates a more efficient transfer of data.



On average, the desktop device's disk I/O speed was 1.775 times faster than the Pi's. The Pi's disk I/O speed plateaued at approximately 400 kilobytes/second while the desktop device's did not reach its limit within our test range.

Power Tests

Our team was unable to obtain reliable and relevant power measurements of both devices in a way that would allow us to effectively compare power usage. However, we can reasonably hypothesize that power usage tests would have been the point at which the Pi's performance exceeded that of the desktop, as the pi is a characteristically lower-power system and the higher performance capabilities of the desktop are at the expense of power efficiency and consumption.

Conclusions and Future Work

In summary, we used a variety of parameters to quantify and compare performance of a low power processor system, the Raspberry Pi 4 Model B, and a standard Linux desktop system with an Intel i9 10900k processor. Our findings confirmed our initial predictions that the higher power system would have a better measure of performance across all tests. Throughout all our testing the desktop device never reached its limit has been the case in all previous tests. This is because the desktop's system capabilities completely eclipse that of the Raspberry Pi 4's and our testing methods focused on showing performance changes for both devices as the workload increased. The test with the highest proportional disparity in performance between the two was the CPU usage test, which the Pi had on average 17.69 times higher CPU usage to the desktop on, whereas on the response time test the Pi on average took only 3.9 times longer, on the memory test the pi's memory usage percentage was on average 9.25 times higher, and for the disk I/O test the Pi on average had speeds 0.65 times slower than the desktop, or in other words the desktop was on average 1.775 times faster. This simplified overview of test results makes the discrepancy of performance between the two devices very clear. In addition, the especially high difference between the results in the CPU test explicitly illustrate our findings that the power of the processor directly affects the overall performance of the system because the processor is intrinsically linked to device performance.

Future work may involve comparing performance through more complex daily usage as opposed to tests specifically designed to put stress on different system components. This would provide a more in-depth analysis of how the systems compare with typical usage. Beyond this, our tests could be improved by finding a better method to measure and compare power usage.

In conclusion, our comprehensive testing has an anticipated trend of the higher power system performing better in all tests, with the caveat that it likely would have been less efficient than the Pi in a test of power usage.

References

[1] R. E. Gonzalez, "Low-Power Process Design," dissertation, Departments of Electrical Engineering and Computer Science, Stanford, CA, 1997

[2] Shah, Munam & Anwaar, Waqas. (2015). Energy Efficient Computing: A Comparison of Raspberry PI with Modern Devices. International Journal of Computer and Information Technology. 4. 410-413.

[3] S. Segars, "Low Power Design Techniques for Microprocessors," in *ISSCC*, Nov. 19, 2023

[4] S. Kaiser, I. Materic, and R. Saade, "ESL Solutions for Low Power Design," thesis, 2008

[5] J. P. Brennan, A. Dean, S. Kenyon, and S. Ventrone, "Low Power Methodology and Design Techniques for Processor Design," thesis, 1998

[6] R. Puri et al., "Bridging High Performance and Low Power in the era of Big Data and Heterogeneous Computing," Nov. 19, 2023

[7] S. Jadeja, P. Mathuria, and J. Popat, "A Review on Low Power Testing Techniques," dissertation

Appendix

To avoid confusion of responsibilities, our group decided on a general outline of task delegation during the introductory phases of the project. Task assignment was based on individual availability, current working knowledge/strengths and contextual understanding of the project, and preference. Individual contributions are as follows, written by the corresponding team member:

> Daria Casey set up the Raspberry Pi and helped run the 5 performance tests on both machines. The raw data from the tests, as well as generated graphs, were recorded by her in an excel sheet that was shared with the group. She also created a documentation log to ensure team members were informed of how the tests were completed. Lastly, the presentation was created by her, and she completed half of the slides.
>
> Devon Wilkening helped set up and run the 5 performance tests on the two machines. He aided in recording the raw data and performed initial data analysis as the tests were being completed. Half of the presentation slides were completed by him, including verification that our findings were accurate to those in our sourced related works. He also served as the primary spokesperson during team meetings to effectively communicate project progress and goals.
>
> Cassandra Priddy's primary contributions were writing the Abstract, Introduction, Results, and Conclusion sections of the report. This necessitated the analyzation and refinement of raw data collected from experimentation by other team members, which was accomplished primarily with the use of RStudio. In addition, Cassandra was present to observe several testing sessions led by other team members and aided in proofreading the report and presentation and provided graphs and data analysis used in each.

Kelcee Gabbard’s contributions included helping run the 5 performance tests and data observation and analysis. She also helped with documentation of each testing strategy and data to assist the other group members. Initial contribution was made with researching methodology and previous works related to the topic. Gabbard was also responsible for the Introduction along with Priddy, Related Works, Methodology, and References for this report.