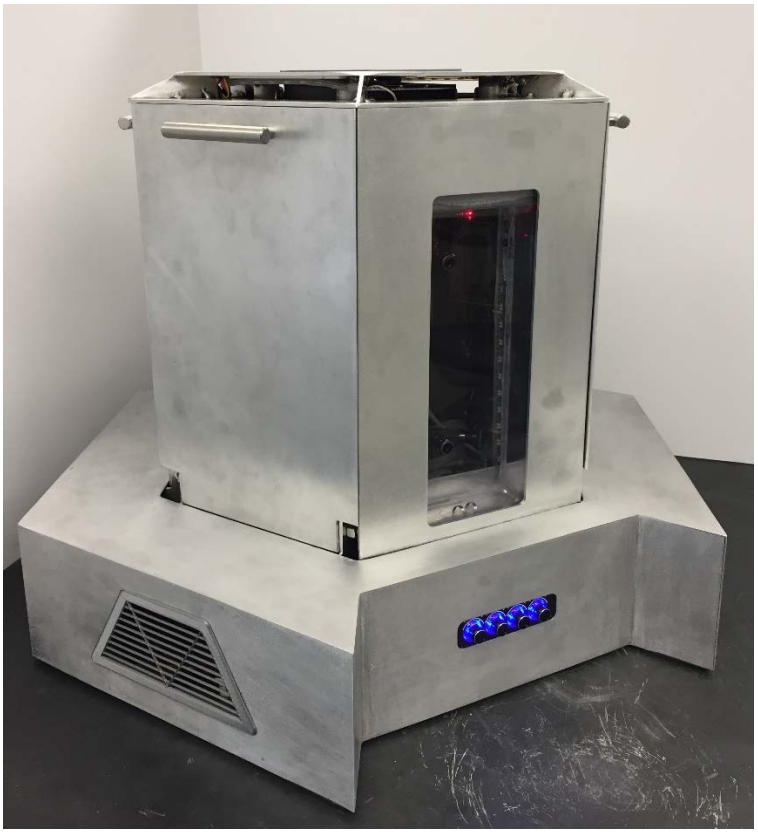
Crayowulf Computer Cluster
Design Report



Team 8
Philip Holmes, Peter Oostema, Noah Pirrotta, and Benjamin Kastner
Engr 339/340 Senior Design Project
Calvin College
10 May, 2018

Executive Summary

Team Crayowulf created a multi-computer cluster (mini-Beowulf cluster) using five Nvidia Jetson TX2 computers. A Beowulf cluster is a cluster built from commercial-off-the-shelf (COTS) parts and open-source software [1]. This project was commissioned by Professor Joel Adams and will be used for research, teaching, and recruitment. The cluster runs Ubuntu Linux operating system and utilizes parallel computing software. One application written for the system is post quantum encryption. A case was designed and fabricated to house the computer cluster, additional electrical components, and all associated wiring. The case resembles the Cray-1 supercomputer from 1973, as requested by Professor Adams. This case provides mobility for the cluster and allows for easy maintenance via hinged side panels. A liquid cooling loop was designed to expel heat from the central processing units (CPUs) and graphics processing units (GPUs) of the Jetsons. The reservoir and water blocks that sit on the chips were custom fabricated for this system, while the pump, radiator, and tubing were purchased. The implementation of a water cooling loop led to a 40% decrease in average CPU core temperature above ambient and a 30% decrease in average GPU core temperature above ambient.

Table of Contents

Table of Figures

Table of Tables

1 Introduction

1.1 Project Background

Professor Joel Adams from the Calvin College Computer Science Department initially developed the idea for this project. He built a mini-Beowulf cluster called Microwulf in 2007 with Tim Brom. Their design did not include a proper case. Microwulf consisted of four computer central processing units (CPUs). Professor Adams wanted to update the project to include newer hardware, as well as a case for the system. Thus, project Crayowulf was born.

1.2 Team Description

Team Crayowulf consists of Ben Kastner, Philip Holmes, Peter Oostema, and Noah Pirrotta, pictured below. The name “Crayowulf” is derived from a combination of Cray, a supercomputing company, and mini-Beowulf cluster.

Ben Kastner Philip Holmes Peter Oostema Noah Pirrotta

1.2.1 Philip Holmes

Philip Holmes is an Engineering major with a concentration in Mechanical Engineering and Physics minor. He has gained both industry experience, through his internship at Crenlo, and research experience, through his Summer Undergraduate Research Fellowship at the Mayo Clinic. He is interested in pursuing Biomedical Engineering research.

For this project, Philip focused on designing the case. He used his courses in machine design and dynamics to accomplish this task.

1.2.2 Noah Pirrotta

Noah Pirrotta is an Engineering major with a concentration in Mechanical Engineering and a Mathematics minor. In the summers of 2016 and 2017, Noah worked for Medallion Instrumentation Systems in Spring Lake, Michigan as an engineering intern. He was an electrical engineering intern in 2016 and a mechanical engineering intern in 2017. After he graduates, Noah will continue working for Medallion as a full-time Mechanical Engineer. Noah has a deep interest in computer technology and has been working with computers for years. He also has some experience with 3D printing.

As part of Team Crayowulf, Noah worked to create the cooling system for the computer cluster. He used CAD software to model the water block for liquid cooling the Nvidia Jetsons. He then fabricated the water block components using a CNC Milling machine. He also designed and printed the 3D printed components of the system.

1.2.3 Peter Oostema

Peter Oostema is an Electrical/Computer Engineering, Computer Science, and Mathematics major. In the summer of 2016, Peter interned at the University of Kentucky working in the advanced networking lab. Here he worked on the VIP Lanes project building a test topology for the application of software defined networking and improving the web monitor for the service. In the summer of 2017, Peter was at Carnegie Mellon in Franz Franchetti's research group, where he implemented the Method of Four Russians for small finite field matrix multiplication. The project also focused on writing the machine instructions to maximize performance.

On this project Peter implemented Supersingular Elliptic Curve Isogeny Cryptography on the computer cluster. He also designed and programed custom electrical components, such as the LED lighting.

1.2.4 Benjamin Kastner

Benjamin Kastner is a Computer Science major and Business minor. He is completing his senior project a year early to facilitate an early graduation date. His interests are in computer networking and system administration. He has taken courses in system administration and operating systems and is currently taking a computer networking course.

In this project Ben configured the software that runs the computer cluster. He then used system benchmarks to compare the cluster to other supercomputers.

2 Project Management

2.1 Team Organization

The team reported to both their engineering advisor, Professor Mark Michmerhuizen, and computer science advisor, Professor Joel Adams. Meetings with Professor Michmerhuizen were not held regularly outside of check-ups during scheduled ENGR 339/340 class days. Meetings with Professor Adams were held on every week besides when Professor Adams was unavailable. The team was advised by their industrial consultant, Eric Walstra, one time during each semester.

The team was split into two groups. This was because the project required both computer science and electrical engineering tasks and mechanical engineering tasks. The computer science and electrical group focused on programming the Nvidia Jetsons and handling other auxiliary electrical equipment. The mechanical engineer group, or computer case and cooling group, focused on the design and creation of the case and cooling system.

2.1.1 Computer Case and Cooling Team

Philip and Noah were delegated to the case and cooling group because of their mechanical engineering majors. They used what they had learned in their machine design & machine dynamics courses to design the case and used what they had learned in their thermodynamics, thermal fluids, and heat transfer courses to design the cooling system.

Specifically, Philip's primary task was to design and fabricate the computer case, while Noah's primary task was to design and fabricate the cooling system. However, there was a fair amount of collaboration between the two.

2.1.2 Computer Software Team

Peter and Ben were delegated to the computer software group because of their computer science majors. Peter also has an electrical engineering major. They used what they had learned in their computer science courses to write the code and make the configurations that allow the Jetsons to communicate with each other. Peter also implemented Supersingular Elliptic Curve Isogeny Cryptography into the cluster. Peter's electrical engineering courses aided him in designing the power distribution system and LED lighting. Ben configured the network settings on the nodes to allow them to communicate, as well as install necessary software packages. Ben's system administration course, operating systems course, and networking course equipped him to perform these tasks.

2.2 Schedule

At the beginning of the project, a plan of action was derived in the form of a Gantt chart. The initial Gantt chart can be found in **Appendix B** as **Figure B63.** The team kept on track with the

schedule for the first semester and all engineering deadlines for deliverables were met. Computer science deadlines were also met, and the Nvidia Jetsons were delivered on schedule.

During the second semester, there were multiple factors that interrupted the progression of this project. The time to compile and debug assembly code were unknown at the beginning and ended up causing delays during the second semester. Similarly, the time needed to fabricate the case and cooling system took longer than expected and delayed progress. As these slowdowns arose, the remaining time was evaluated, and new schedules were drafted. The most recent and relevant schedule can be found in **Appendix B** as **Figure B64.**

2.3 Budget

There were two sources of funding for this project: the Calvin College Computer Science Department and the Calvin College Engineering Department. At the start of the project, the team met with Professor Adams to discuss the budget allowed by the Computer Science Department. Professor Adams and Ben corresponded with Nvidia about purchasing the Nvidia Jetson TX2 units. Nvidia donated one system and charged an educational price for the other four. The Computer Science Department also provided the power supply, network switch, and the appropriate connection cables. For a budget outline on Computer Science funds, please consult **Appendix B**.

The Engineering Department provided funds for the components necessary for the case and cooling system. Most of the case, as well as the water blocks and reservoir for the cooling system, were made primarily using materials found in the Calvin Engineering workshop. However, several additional components for the cooling system, and a small portion of components for the case, were purchased using the funds provided. For these components, a budget composed of individual component pricing was proposed by the engineering students to Professor Michmerhuizen. He approved the initial budget; whenever a team member ordered components, the order cost was deducted from this approved budget. Each member sent in order forms to Professor Michmerhuizen to obtain the parts required for the project under his jurisdiction. No budget issues arose, as many of the parts originally proposed had decreases in price. Thus, a surplus of budget allowed the team to add even more features than expected, such as an Arduino Uno as an LED controller. For the proposed budget and list of project expenses of Engineering funds, consult **Appendix B.**

2.4 Method of Approach

As mentioned above, different teams were selected based on the required individual projects. Philip led the case design, Noah led the cooling system design, Ben led the node configuration, and Peter led the cryptography and electrical design. Noah was selected to lead the three engineers in communication with the computer science faculty, as well as with their engineering advisor.

Ben and Peter were the team members primarily involved with research due to the nature of their roles. Ben heavily researched information on how to configure each of the Nvidia Jetsons to run a mini-cluster setup. He also did research regarding the various software packages used on the Nvidia Jetsons. Peter researched information regarding cryptography as well as electrical design of the system.

3 Research

3.1 Post Quantum Cryptography

Post quantum cryptography includes any cryptographic scheme that cannot be broken by a quantum computer running a polynomial time algorithm. Much of today's cryptography is not quantum cryptography, many are based on the difficulty of factoring numbers or computing a discrete logarithm. Solving these problems takes years due to the sizes of numbers used, but when quantum computers get large enough they will be able to be solved asymptotically faster. So, there is now a push to create cryptographic schemes that cannot be easily broken by a quantum computer attack.

3.1.1 Supersingular Elliptic Curve Isogeny Cryptography

Supersingular Elliptic Curve Isogeny Cryptography uses a shared secret key generation named Diffie-Hellman (later discussed in section 4.1.2). To start the system the parties communicating agree to work from a common supersingular elliptic curve. [2] A supersingular elliptic curve is an elliptic curve with an unusually large number of endomorphisms (also called isogenies). To illustrate this, one can imagine two parties, Alice and Bob, pick two random integers and find their own secret isogeny from them. They then send the resulting elliptic curve from the isogeny as their public key to one another. Taking the elliptic curve from each other, they apply their isogeny again to arrive at the same secret elliptic curve. The curve’s unique j-invariant function is used for encryption and decryption between the two.

3.1.2 Diffie-Hellman Key Exchange

Diffie-Hellman key exchange is a method of establishing a shared secret over public communication. The two parties involved, as above Alice and Bob, start off with a publicly known base. They both generate their own private key and modify the base with it. They then send the combination to each other in a public channel. Alice and Bob next modify what they receive from each other with their private key. At the end, they have the same shared secret that can be used to encrypt and decrypt messages between the two. The process is shown in **Figure 1,** where each operation is represented by a color.

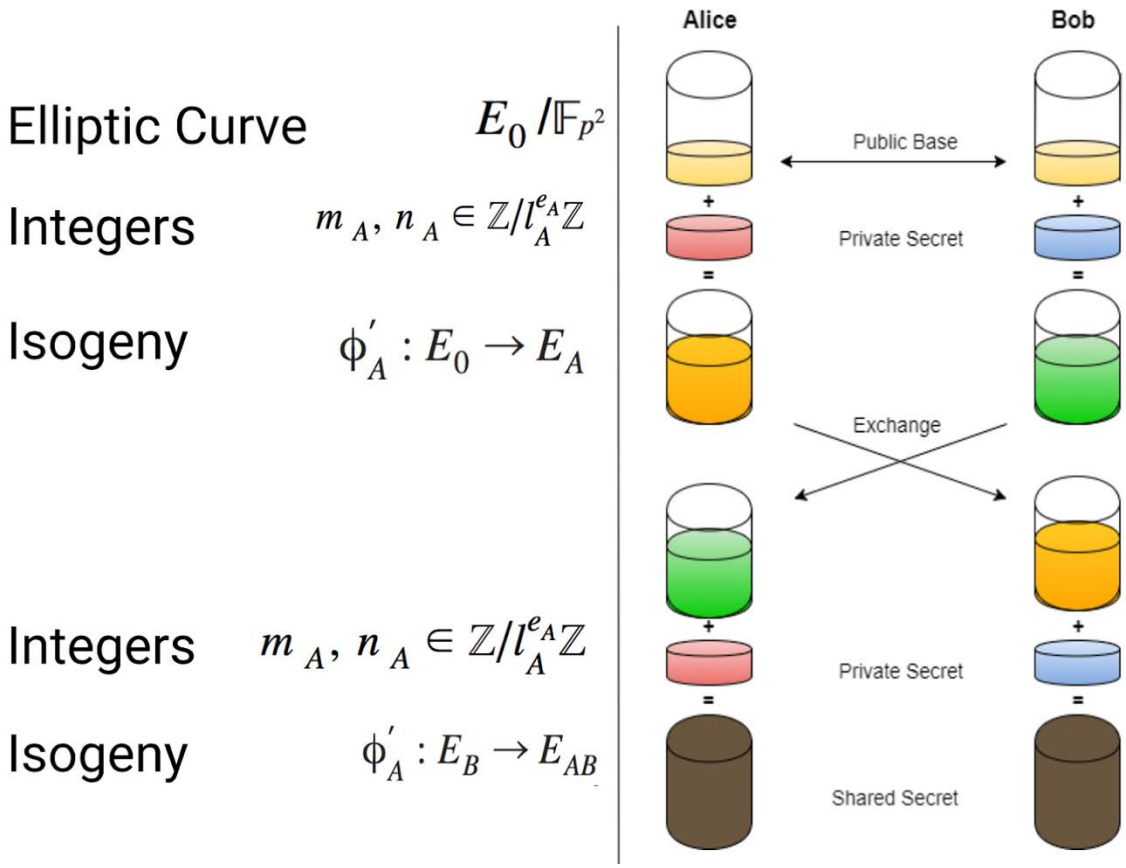


Figure 1. Diffie-Hellman Shared Secret Steps and Relationships to the Cryptography Scheme

3.2 Nvidia Jetson TX2 CPU

The Jetson TX2 has a SoC, Tegra X2 that contains the CPU which has two ARM processors interconnected to work together and a Pascal GPU ("NVIDIA Jetson TX2..."). The processors and their connections are shown in **Figure 2**.

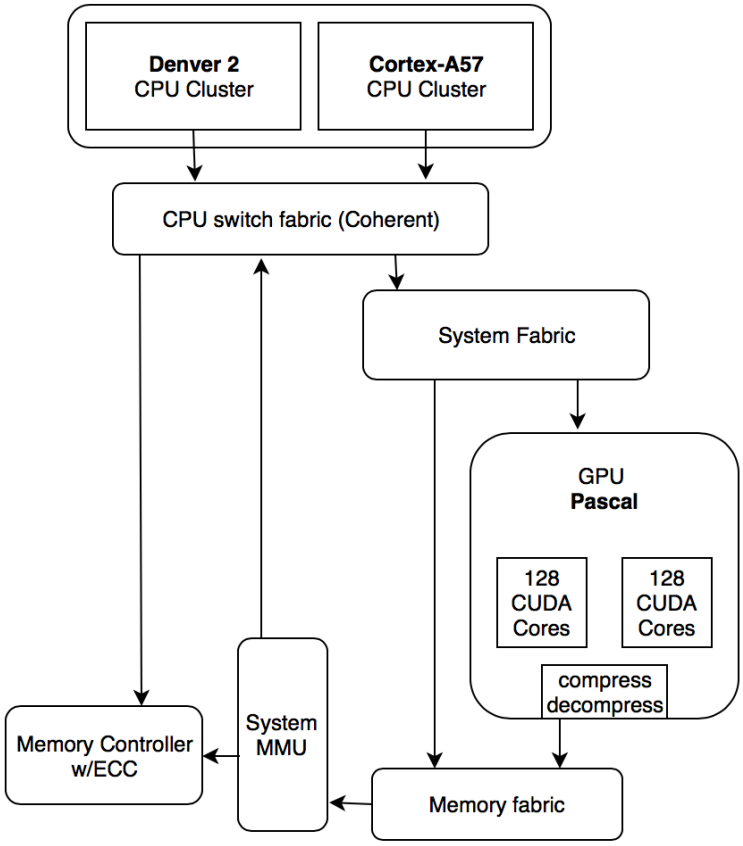


Figure 2. Tegra X2 Block Diagram Simplified from Nvidia [3]

3.2.1 Denver2

Part of the CPU on the Jetson TX2 is a dual core ARM processor running at a maximum of 2.0GHz. This chip was designed by Nvidia as their project Denver to test their GPUs. The Tegra X2 contains the second-generation version called Denver2. This processor is better used for heavy single threaded tasks. It makes extensive use of out-of-order execution to improve performance by better utilizing the pipeline architecture.

Denver2 uses a 7-stage superscalar pipeline. The 7-way pipeline needs to be understood as running two instructions that depend on the last and may need to wait several cycles before the result from the first is available to the second. Superscalar processors can run two independent instruction in the same clock cycle. To fully utilize this pipeline, programs should line up many independent instructions to maximize performance.

Another important aspect of the processor is the behavior of its caches. Denver2 has two levels of caches. The first level is split into instructions and data, while the second is unified. Full information on the caches is contained in **Table 1**. For programming, working within the size limits of the caches will give great performance gains when fewer reads from main memory are needed.

Table 1. Denver2 Caches

Level	Data or Instruction	Associativity	Size	Other
L1 TLB	Instruction	4-way	128 entry	4kB page sizes
L1 TLB	Data	8-way	256 entry	4k, 64kB page sizes
L1 TLB cache	Unified	8-way	2048 entry	
L1	Instruction	4-way	128kB	Per core
L1	Data	4-way	64kB	Per core
L2	Unified	16-way	2MB	Shared

3.2.2 ARM Cortex-A57

The second part of the Tegra X2 CPU is a four core ARM Cortex-A57. This chip has a smaller CPI (clocks per instruction) than the Denver2, but runs the same instruction architecture. Having more, slower cores means it is better used for lighter workloads or parallel processing.

The Cortex-A57 uses a 15-stage pipeline that is not superscalar. It has support for 8-wide out-of-order execution to offset the longer structure of the pipeline. Like Denver2, programs need to be able to feed the processor many independent instructions to make full use of it.

Caches in this chip are similar to Denver2, with two levels: L1 separated into data and instructions for each core and L2 unified storage for all cores. The L1 caches have another condition to pay attention to as well. The L1 instruction cache can read in 128 bits at a time. The L2 data cache can queue up to six 64 bit reads while running more instructions that do not depend on the data. The Cortex-A57 Caches can be seen in **Table 2**.

Table 2. Cortex-A57 Caches

Level	Data or Instruction	Associativity	Size	Other
L1 TLB	Instruction	Full	48 entry	4k, 64k, 1MB page sizes
L1 TLB	Data	Full	32	4k, 64k, 1MB page sizes
L2 TLB	Unified	4-way	1024 entry	Per core
L1	Instruction		48kB	Per core
L1	Data		32kB	Per core
L2	Unified		2MB	Shared

3.2.3 Instruction Set Architecture

Both the Denver2 and the ARM Cortex on the Tegra X2 fully implement the ARMv8-A instruction set architecture. ARMv8-A uses 32, and 64-bit instructions. The instruction set has all the normal arithmetic to be expected: add, multiply, divide, shift, and bitwise operation. It also includes support for ARM's NEON vector instructions. [4] These instructions implement all the same arithmetic but using 128-bit data. The Denver2 and Cortex-A57 both include a cryptographic engine that includes a vectorized multiplication instruction that stores the full 256-bit precision result from two 128 bit floating point numbers. This may be useful when working with more than 700 bits long integers needed in supersingular elliptic curve isogeny cryptography.

3.3 Nvidia Jetson TX2 GPU

The Tegra X2 on the Jetson TX2 has a 256 core Nvidia Pascal GPU. These cores are full CUDA cores that can be used by the API with the same name or other GPGPU interfaces. The GPU has a maximum theoretical output of 750 GFLOPs at single precision. The chip also includes 8 GB of DDR4 memory with a bandwidth of 58.4 GB/s. [5]

3.4 Case and Cooling Research

The case rear I/O required dimensions from online sources for AutoDesk Inventor modeling. HDMI dimensions were found from Data Pro [6]. The dimensions for the ethernet extender purchased used in the system were detailed on Amazon [7].

Various sources were used for the theoretical thermal analysis of the cooling system. The thermal conductivity for 6061-T6 aluminum was obtained from Aerospace Specification Metals Incorporated [8]. The thermal conductivity for thermal paste was found from Gaming Factors [9]. The methods for calculating the hydraulic diameter, Reynold's number, and for identifying turbulent flow were found from the Cengel book listed on uio.no [10]. The method for calculating the convective heat transfer coefficient for water flow through the water cooling blocks was obtained from R. Shankar Subramanian's article [11]. Information regarding NTU effectiveness calculations for the radiator was found using *Fundamentals of Thermal-Fluid Sciences* [12]. An estimate of the effectiveness of the radiator was made using *Fundamentals of Heat Exchanger Design* [13]. Additionally, information regarding the difference between deionized water and distilled water was found from Thought Co. [14]. According to Thought Co., deionized water can react with air and become acidic. Thus, distilled water was used for the cooling system.

4 Design

4.1 Requirements

To ensure a successful preliminary design phase for this project, requirements needed to be clearly defined and understood. There were requirements for both the mechanical and software sides of this system. All system requirements are listed in this section.

4.1.1 Case and Cooling Requirements

- The case needs to be appropriately sized – able to fit in a 30 in by 30 in by 30 in box
- The cooling system must keep the Nvidia Jetsons at or below 60°C
- The case needs to be designed in a way that makes the Nvidia Jetson boards accessible for easy maintenance (e.g. change liquid)
- The case must incorporate 3D printed parts where appropriate
- The case must have structural integrity and be transportable
- The case must be well organized so that it may be used in a teaching environment
- The cooling system must be able to be used to teach the impact that CPU/GPU temperature has on processing speed

4.1.2 Cluster and Software Requirements

- The cluster must consist of 5 Nvidia Jetsons connected together
- The system must use a single computer PSU to power the system
- CUDA, MPI, and NFS must be installed
- The nodes must communicate in a master-worker relationship
- The worker nodes must be able to mount shared storage via NFS
- The nodes must utilize multiple network interfaces efficiently

4.2 Case Design

The case is an integral component of this project. The computer science department commissioned a case to be made so that the system is more interesting and approachable. A computer project similar to the Crayowulf system was completed back in 2007 with four boards linked and talking to each other. Unfortunately, this system became known as the “spaghetti monster”, rather than its official name, “Microwulf”, as there was not a proper case to display it in. A picture of this system can be seen below as **Figure 3.**

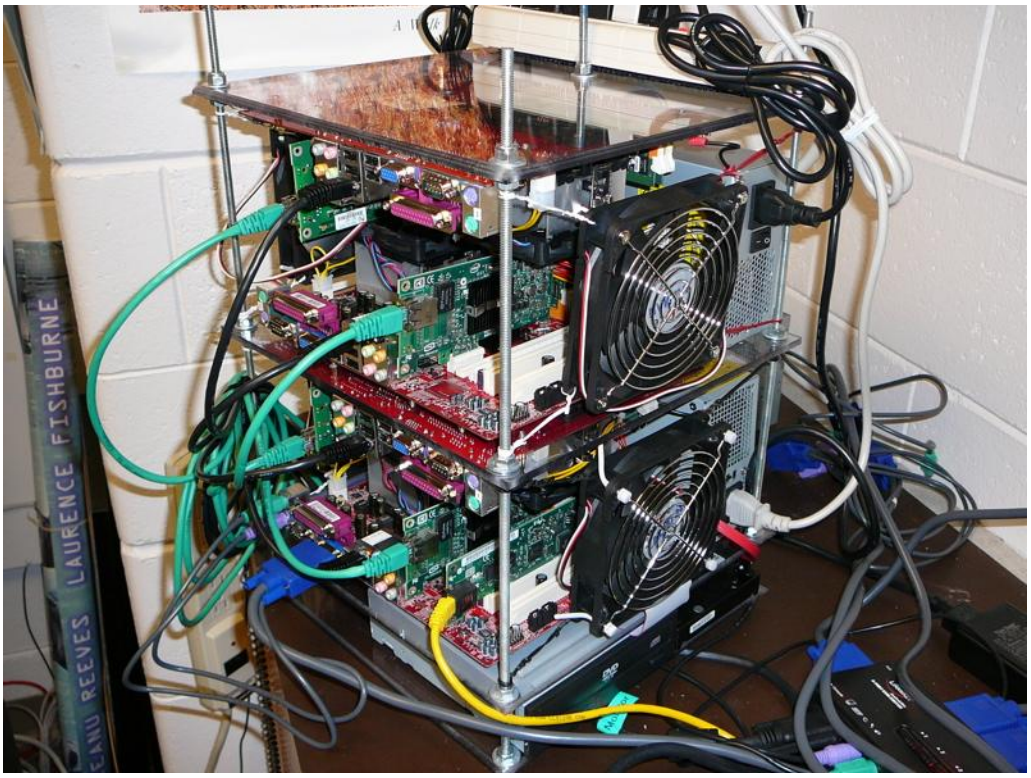


Figure 3. Microwulf “Spaghetti Monster” System [15]

The messy appearance of the Microwulf system necessitated the need for a proper case for the Crayowulf system.

4.2.1 Case Design Criteria

During the design phase of the case, requirements specific to the case and the Christian engineering design norms needed to be kept in consideration.

4.2.1.1 Case Requirements

- The case needs to be appropriately sized - not too small as to impact air flow and appear cluttered on the inside; not too big as to waste materials and appear bulky
- The case needs to be designed in a way that makes the Nvidia Jetson boards accessible for easy maintenance (e.g. change liquid)
- The case must incorporate 3D printed parts where appropriate
- The case must have structural integrity and be transportable
- The case must be well organized so that it may be used in a teaching environment

4.2.1.2 Transparency

A major aspect of this project revolved around open access to the resources used and created. The CAD drawings for the case are readily available for distribution to other universities and to secondary schools.

4.2.1.3 Trust

The design for the case makes this system safe through the structural integrity of the materials used, the storage of electrical components, and the avoidance of sharp edges and objects that are apparent in some computer case designs.

4.2.1.4 Humility

The mechanical engineering team had little experience with machining parts going into this project and ran into adversities in the case fabrication process. When this occurred, they sought the help of Phil Jasperse, who has a high level of experience with machining and tooling. The case team was willing to receive all the help they could get on this aspect of the project.

4.2.2 Case Design Alternatives

For the case, the initial idea proposed by Professor Adams was to have a completely 3D printed case. This would allow for the case design to be easily distributed to other universities and institutions to be recreated. This design alternative was rejected due to price and limitation on 3D printing. The size on the case is simply too large to be completely 3D printed and remain within the budget. Another area of debate on the case design was how the mechanical opening mechanism would be implemented. The initial idea was to have a gear system linked to an electric motor that would open the panels simultaneously. The second option explored was to have a manual linkage system that is counterweighted with a spring retention system. The latter option would be less expensive due to the absence of gears and electric motors. It would also be easier to maintain. The third option which was investigated was a case design which has all of the panels open individually with no linkage. This option would be the simplest and most achievable design for the case.

4.2.3 Case Design Decisions

The selected case design sports a multi-material body, consisting of both metal and 3D printed components. The external paneling and internal skeleton are made entirely of metal, while specific components, such as vent covers, are 3D printed. The alternative of having a linkage opening mechanism was initially chosen over the gear and motor alternative and the independent panel alternative. A handle, pulled by the operator, would control the opening and closing of all side panels. Initial CAD models for this alternative can be seen in **Figure 4** and **Figure 5**.

Figure 4. Preliminary CAD Model of System: Closed

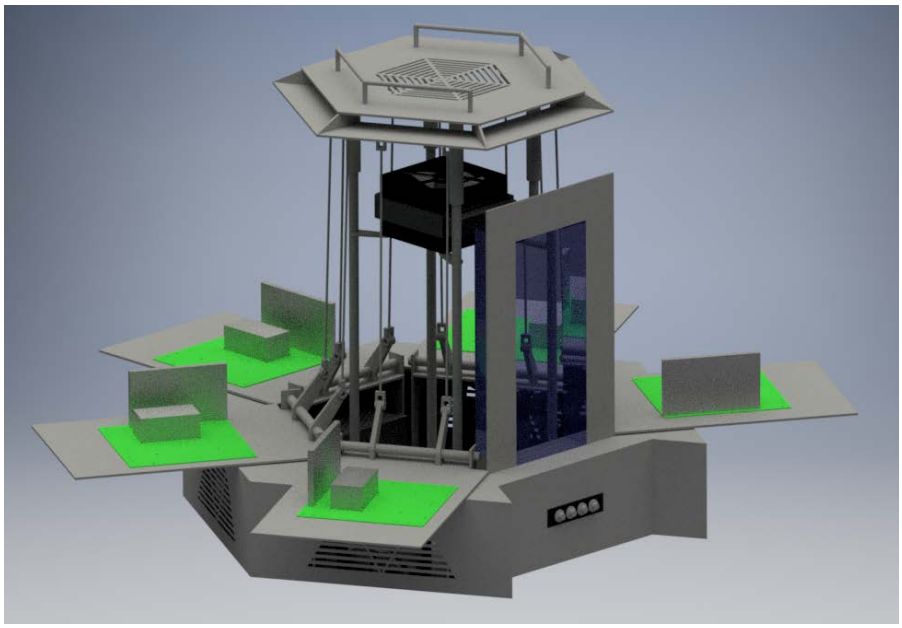


Figure 5. Preliminary CAD Model of System: Open [16]

After consulting Phil Jasperse about the feasibility of fabricating the linkage design, it was decided that the simpler independent panel design would move forward as the final design for the case. A CAD model of this design alternative can be found below as **Figure 6** and **Figure 7.**

Figure 6. Final CAD Model of System: Closed

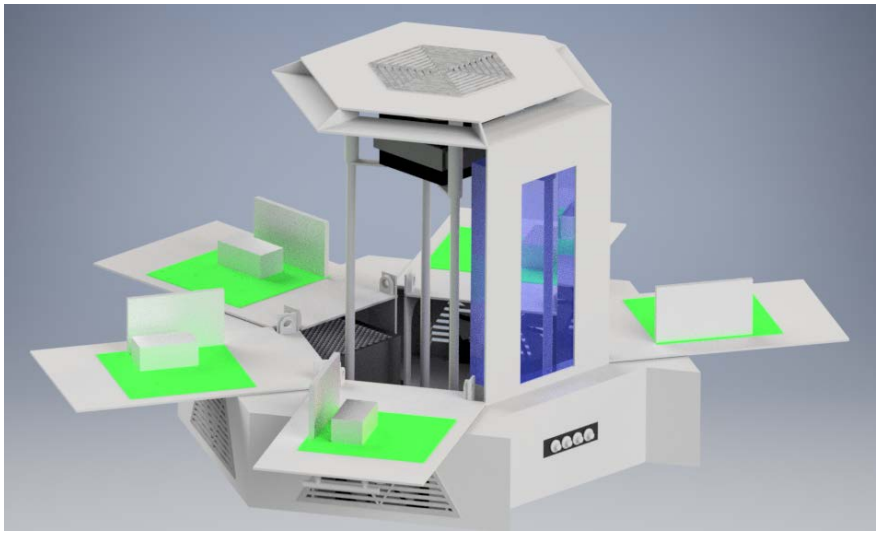


Figure 7. Final CAD Model of System: Open

As can be seen in **Figure 7**, the final system is fully accessible. This allows it to be easily maintained and displayed. The system works well in an educational setting, as students are able to examine the system inside and out. The independent panel opening mechanism also allows the system to be opened more easily when in a constricted server room, as the user will not need to find the large amount of space needed to open all the panels at once.

4.2.4 Case Features and Fabrication

The case was fabricated almost entirely out of aluminum. Though some 3D printed components as well as some steel components were created. The completed case, without any electrical hardware installed, can be seen in **Figure 8**, below.

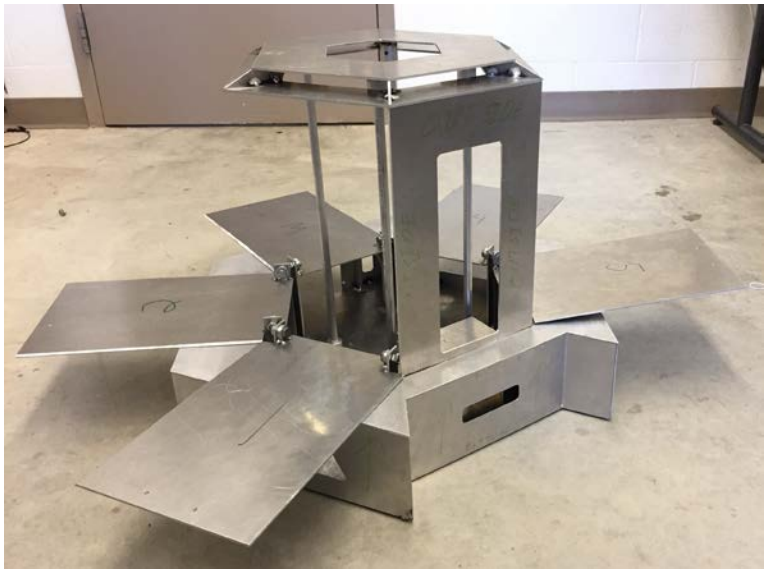


Figure 8. Fabricated Case

The case has many features that are critical to its function and they will be listed and described in the following sub-sections.

4.2.4.1 Case Basement

The lower half of the case is called the basement and can be seen in **Figure 9**.

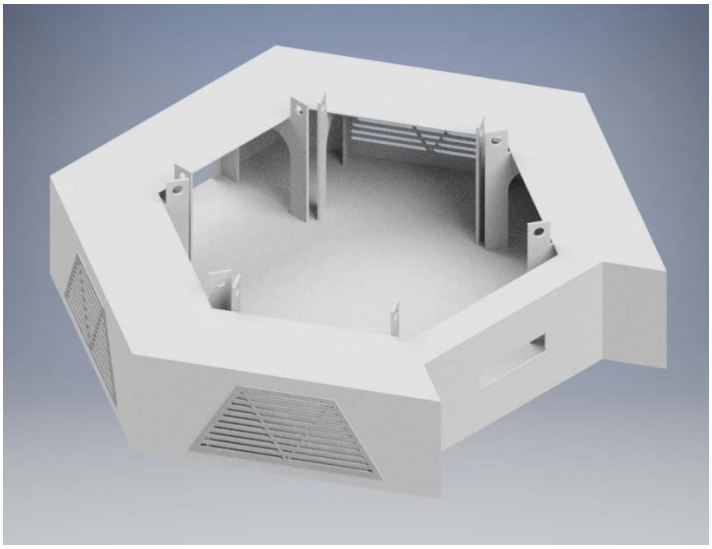


Figure 9. CAD Model of Basement

Its function is to house the power supply, network switch, and Arduino Uno. It is also used to hold fans which pull fresh air into the system. The basement has enough room to effectively hide cables, solving the “spaghetti monster” problem mentioned earlier. The base plate of the system is a flat sheet of 1/8-inch aluminum. A picture of the original sheet stock and cutting lines can be seen in **Figure 10**, below.

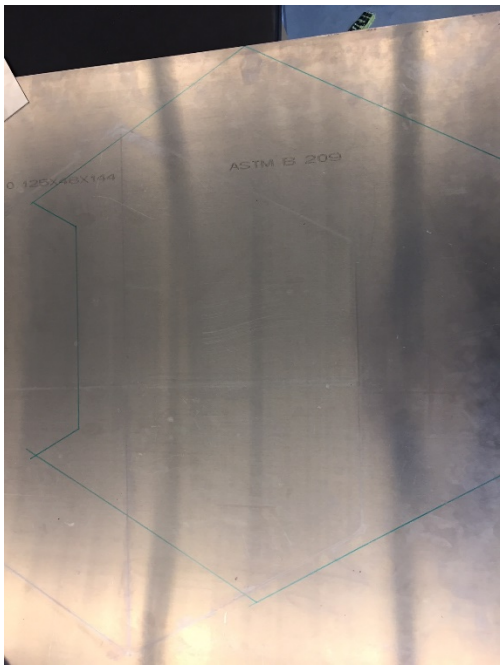


Figure 10. Base Panel Sheet Spock and Cutting Lines

The base was sheared and hand sawed into its characteristic hexagonal shape. The sides and top of the basement are made from 1/16-inch aluminum. The sides consist of 3 separate segments welded to each other and to the base and top. The sides are 2 pieces which had their trapezoidal holes milled out and middle sections bent, as seen in **Figure 11**.

Figure 11. Basement Side Panel

The trapezoidal holes were made to allow fan mounting, and they are covered by 3D printed vent covers, as seen in **Figure 12.** Currently, there are fans in two of the four vent holes. Two fans are sufficient for the current system, but there is room for more fans if the system gets upgraded with new hardware in the future.

Figure 12. Basement Side Panel Vent Holes

The front had a hole milled out to accommodate the fan controller nobs and was bent on each end, as seen in **Figure 13.**

Figure 13. Basement Front Panel

The back panel was the only side piece that was not bent. It had a square milled out for the power supply and another square was also milled out for the rear IO panel containing 2 USB ports, 1 HDMI port, and an ethernet jack. The back panel of the basement can be seen in **Figure 14**.

Figure 14. Basement Back Panel

The top panel, also 1/16-inch aluminum, was sheared and hand sawed into shape and can be seen in **Figure 15**, below.

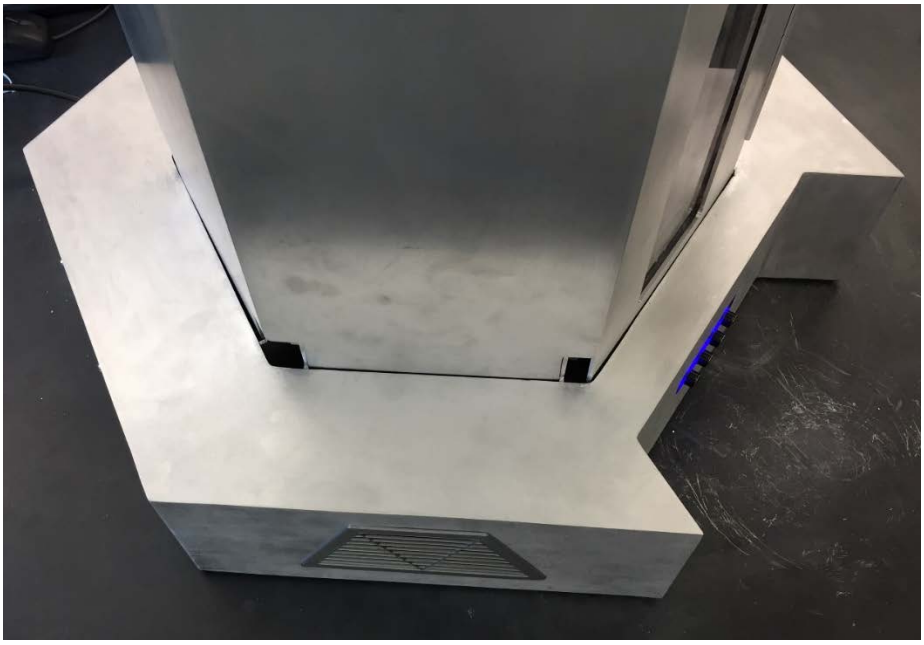


Figure 15. Basement Top Panel

The h-shape brackets, depicted in **Figure 16**, hold up the side and top panels of the basement and give the frame rigidity. They also allow cables to run throughout the basement and act as hinges for the folding side panels which are discussed in the next sub-section. These pieces were milled from a 1/8-inch aluminum sheet.

Figure 16. Basement Brackets

The components of the basement were tack welded together in order to reduce temperature induced warping and melt-through. Larger welds were then added to structurally-critical components once the metal had cooled.

4.2.4.2 Case Side Panels

The case side panels were designed to hold the Nvidia Jetsons and allow them to be easily accessible. The CAD model of the panel is seen below in **Figure 17**.

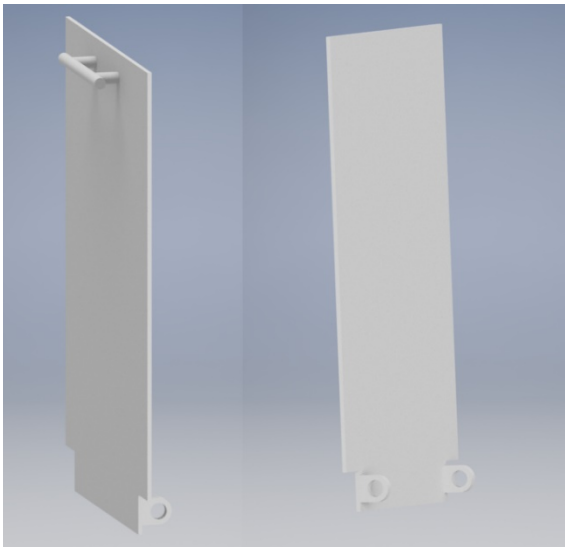


Figure 17. CAD Model of Side Panel

The body of the side panel was sheared from a 1/8-inch aluminum sheet and then milled for precision as well as removing the bottom two corners. The edges of the inside of each side

panel were sanded down to allow the panels to sit next to each other tightly without interference. A sanded corner can be seen in **Figure 18**.

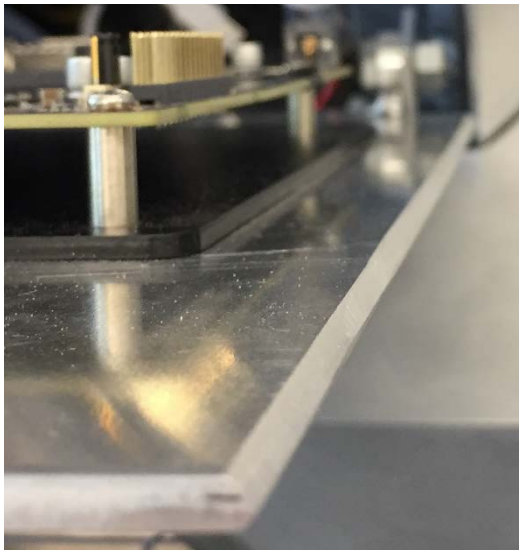


Figure 18. Sanded Corner of a Side Panel

The hinge loops, depicted in **Figure 19**, below, were milled from a piece of ¼-inch aluminum. These allow the panels to pivot.

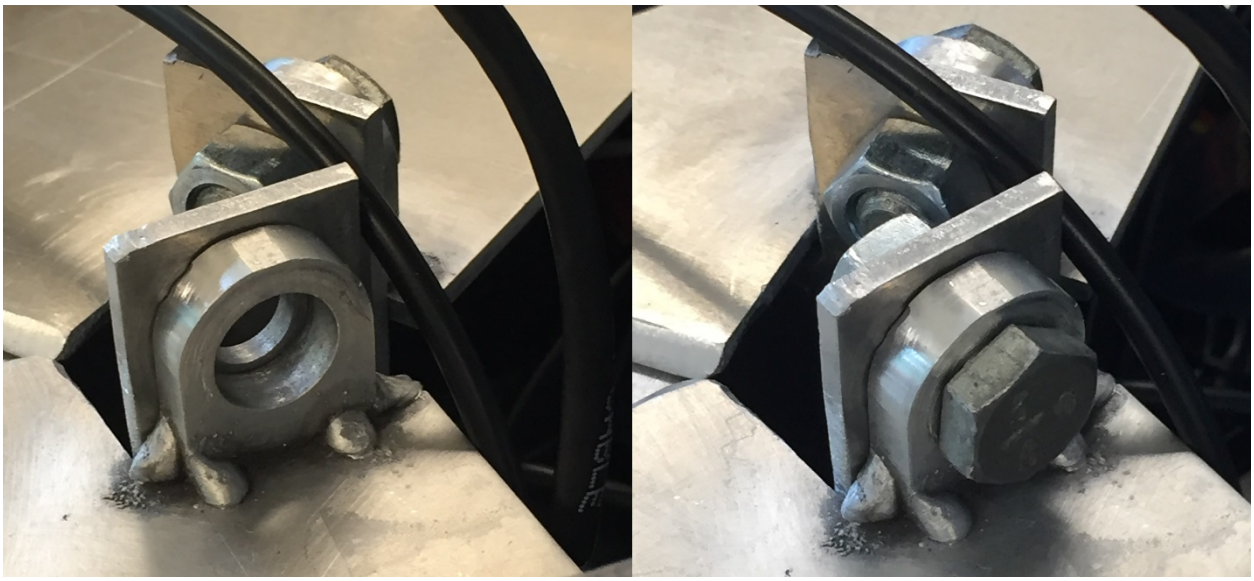


Figure 19. Side Panel Hinge Loops

The hinges pivot about a steel bushing that is held in place by a nut and bolt. The bushing was made in a lathe from a hollow ½-inch tube of steel. The bushing and bolt mechanism can be seen in **Figure 20**.

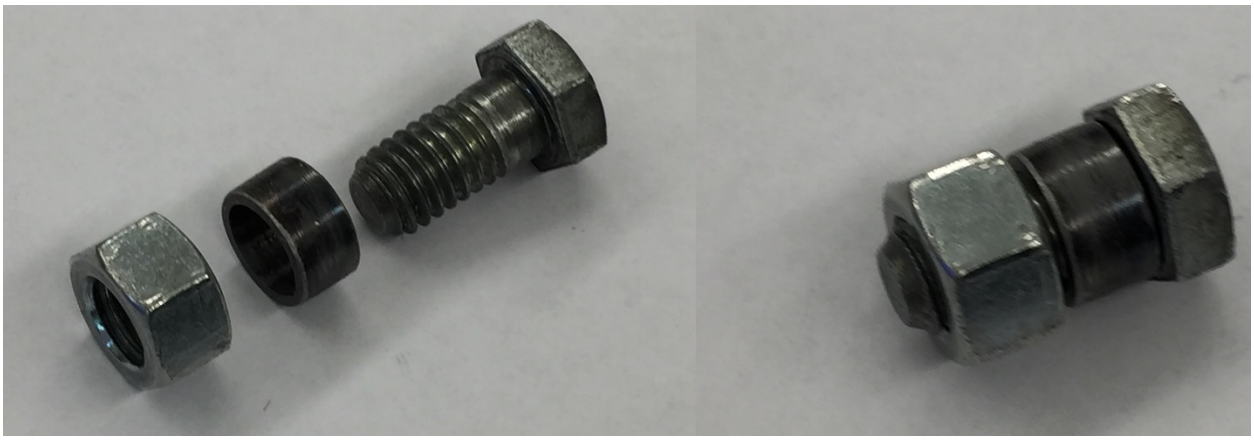


Figure 20. Bolt and Bushing assembly

Two holes were drilled at the top of the panel. These two holes hold both the handle and magnetic contacts for the panel, as seen in **Figure 21**.

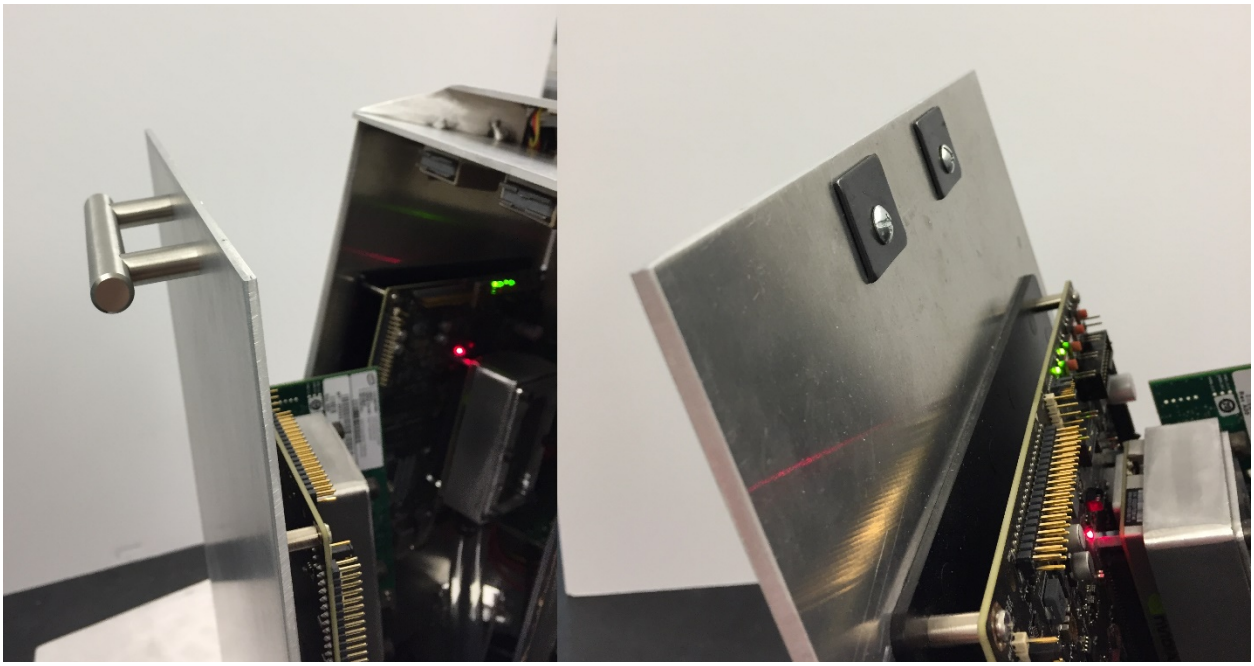


Figure 21. Side Panel Handle and Magnet Contact

Standard cabinetry magnets were chosen as the holding mechanism for this system. They are easy to use and are not strong enough to interfere with the electrical operations of the Jetsons. The magnet itself is mounted on the case top via 2-part epoxy and is seen in **Figure 22**.

Figure 22. Magnets on Top of Case

The Nvidia Jetsons were mounted using the standoff board included with the individual developer kits. These black boards were adhered to the side panels using 2-part epoxy. The finished product can be seen in **Figure 23**.

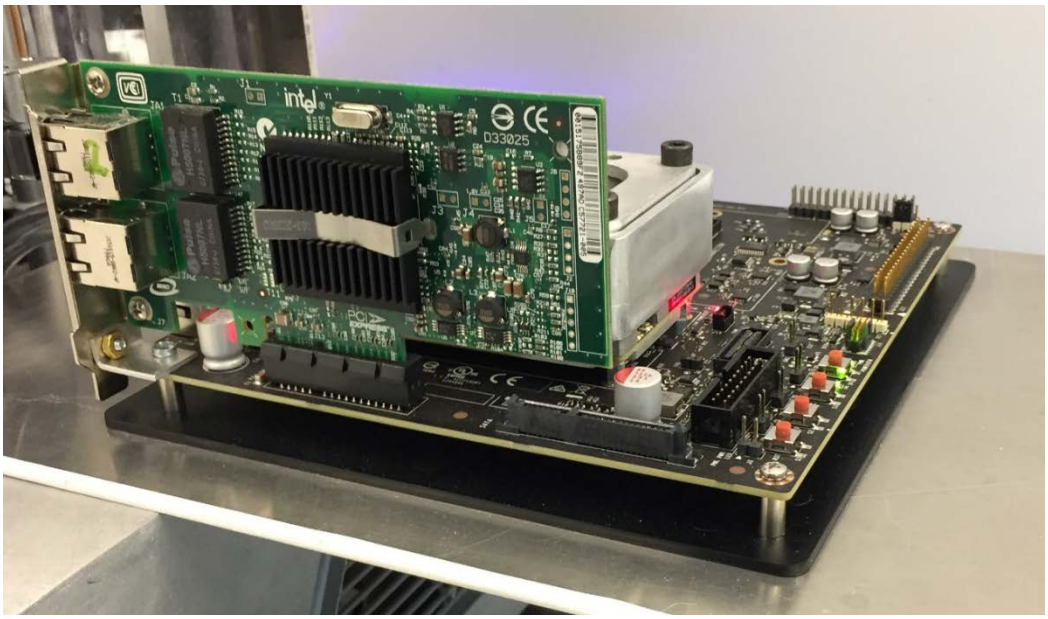


Figure 23. Nvidia Jetson Board Mounting

Small L-brackets were made from 1/16-inch aluminum with holes drilled to secure the networking cards to the assembly. One of these L-bracket is pictured in **Figure 24**, below.

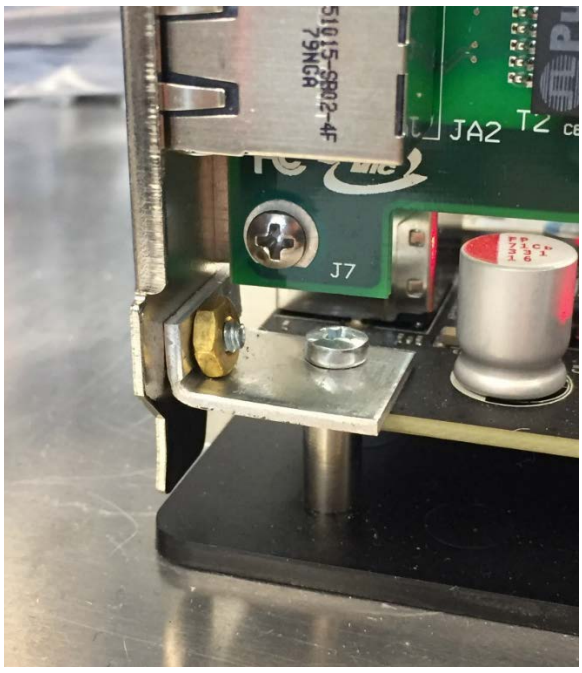


Figure 24. Network Switch Securing Bracket

4.2.4.3 Case Front Panel Assembly

The front panel assembly has multiple purposes. It is designed to hold the pump, the fan controller, and the reservoir. It is also designed to allow the user to look into the system through the reservoir. The CAD model for the front panel assembly can be seen in **Figure 25**, below.

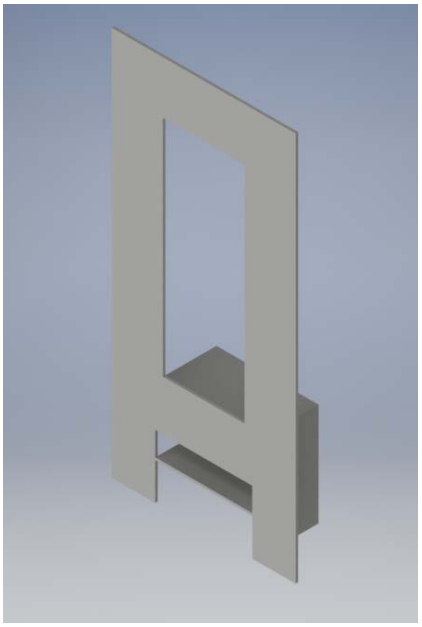


Figure 25. Front Panel Assembly CAD Model

The front panel assembly is made from 1/8-inch aluminum, with holding brackets made from 1/16-inch aluminum. The reservoir is held in place by two brackets, as seen in **Figure 26.** One

bracket extends all the way around the reservoir and keeps it from moving vertically. The other sits low on the back of the reservoir and keeps it from moving horizontally. They are fastened to front panel assembly via nuts and bolts.

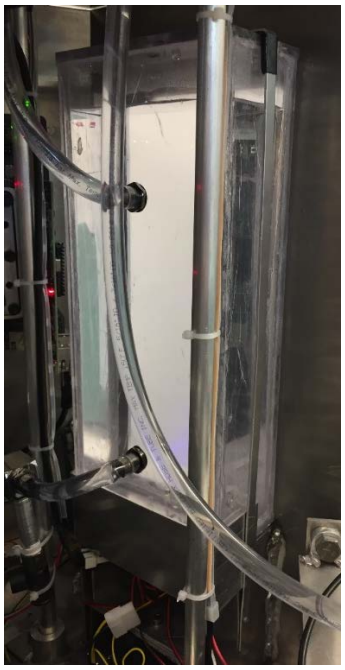


Figure 26. Reservoir brackets

The fan controller is held in place in a sleigh-style bracket, while the pump is held in place by a bracket that hangs down and surrounds the pump. The fan controller and pump brackets can be seen in **Figure 27**, below.

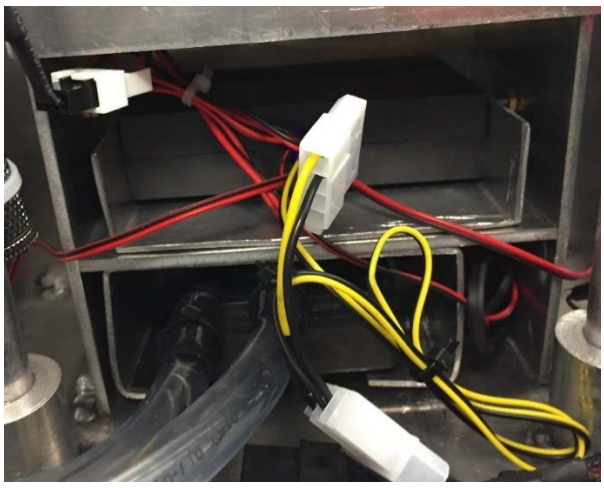


Figure 27. Fan Controller and Pump Brackets

As seen in **Figure 28**, the front panel has been milled out to allow the user to see into the system.

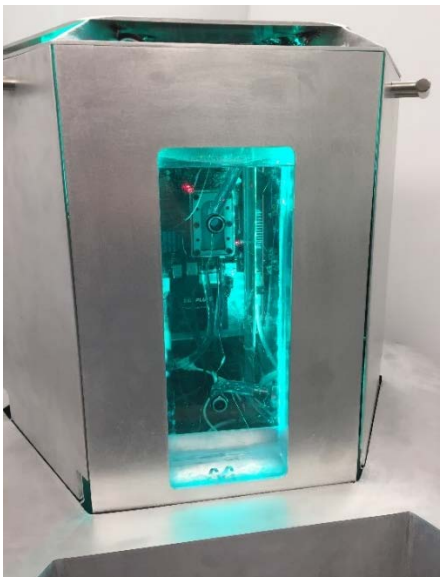


Figure 28. Front Panel Looking into the System

The bottom of the front panel assembly was sturdily welded to the base plate of the case.

4.2.4.4 Case Inner Skeleton

The cases inner skeleton consists of four center posts, adjustable mounts on the base and the top of the case, as well as holding brackets for the network switch, radiator, and fan. The four center posts can be seen in **Figure 29.**

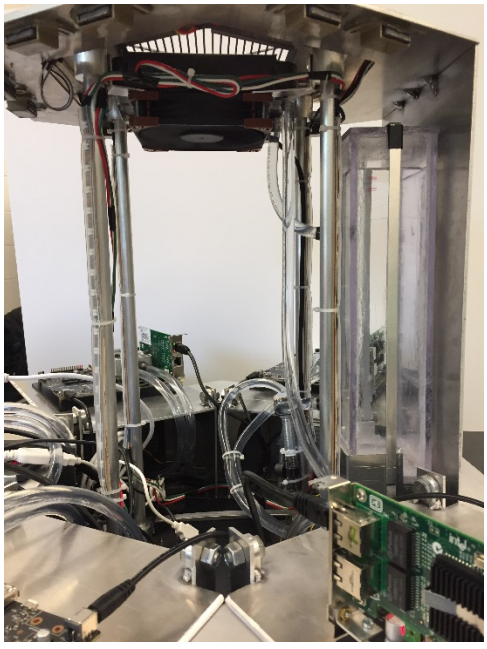


Figure 29. Case Center Posts

These center posts provide the case with rigidity and allow it to be carried from both the top and bottom. The four posts are ½-inch aluminum and are held in place by mounts at both the top and bottom of the case. One of the base mounting points can be seen in **Figure 30.**

Figure 30. Base Mounting Point for Center Post

These base mounting points were machined on the lathe and mill and have set screws which allow for the adjustment of the case top. The set screws are also the mounting points for the network switch retention bracket and the fan and radiator holding bracket. The radiator is held in place with two retention brackets that keep it from moving, as seen in **Figure 31.**

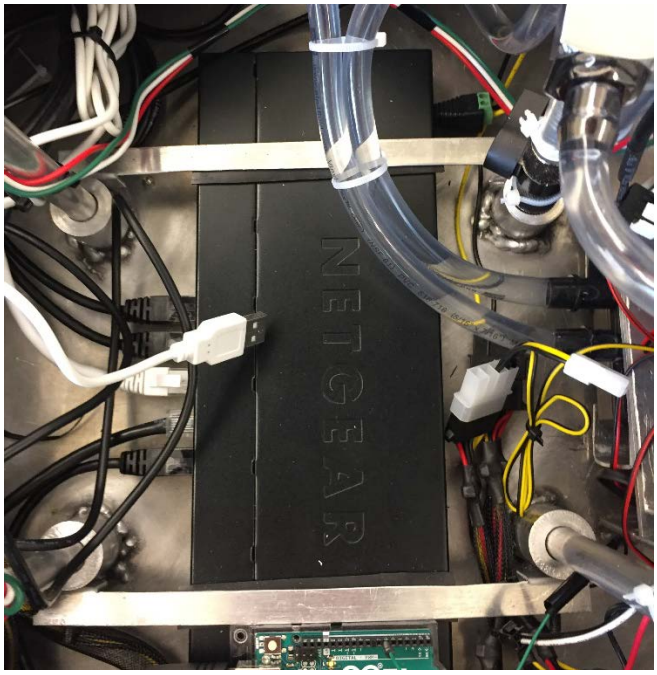


Figure 31. Network Switch Retention Brackets

The fan and radiator have L-shaped brackets that have holes for fasteners. The fan and radiator sandwich the bracket on either side. A picture of one of these brackets can be seen below in **Figure 32.**

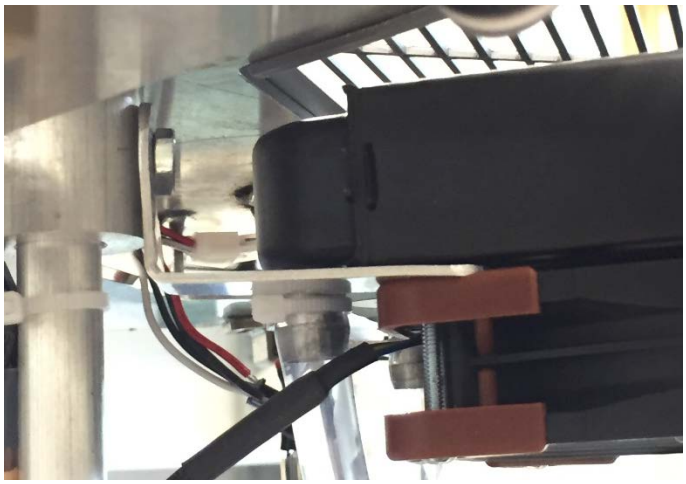


Figure 32. Fan and Radiator Bracket

4.2.4.5 Case Top

The top of the case (CAD model depicted in **Figure 33**, below) was designed to contain LEDs above each node to display temperature and usage information, as well as to let air flow through the radiator and out the top of the case.

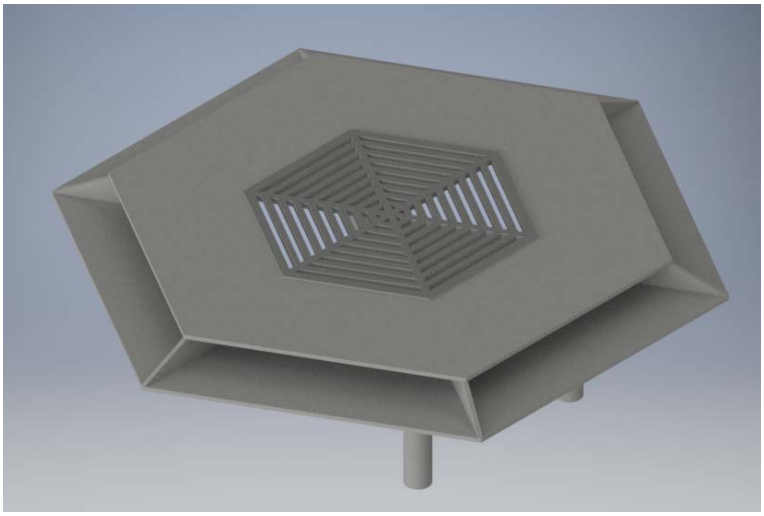


Figure 33. CAD Model of Case Top

The upper and lower layers of the top were sheared and the centers were milled from 1/8-inch aluminum. The angular pieces which separate the upper and lower levels of the lid, seen in **Figure 34**, were also sheared from 1/8-inch aluminum.

Figure 34. Separator Piece between the Two Top Layers

The hexagonal vent hole at the top has a 3D printed vent cover which clips into the top, as depicted in **Figure 35.**

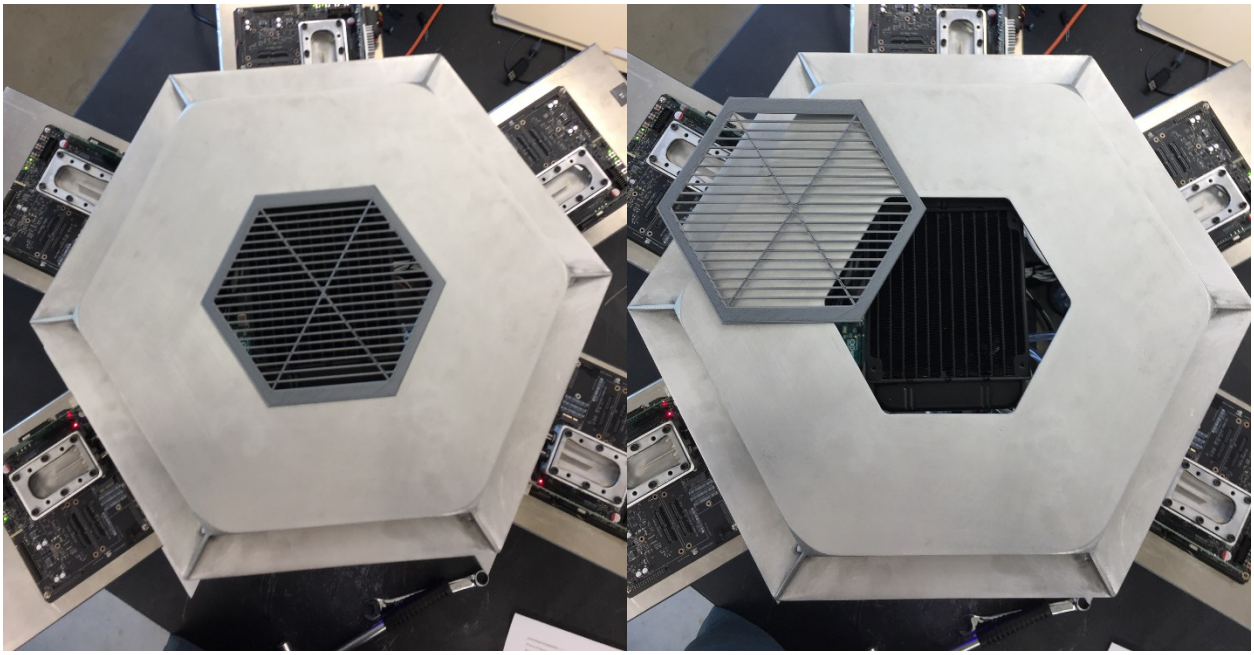


Figure 35. Top Vent Hole with 3D printed Vent Cover

Once all the components were made and assembled, the whole case was sanded to a consistent finish.

4.3 Cooling Design

The electrical energy required by computers transfers into heat inside of the electrical components on the computer boards. The CPU and GPU on a computer use a large portion of this electrical energy and can become very hot in doing so. Overheating of the CPU and GPU can cause the lifetime of these components to decrease. Thus, a cooling system was required to dissipate the heat generated within these components. The more effective the cooling system, the greater the potential increase in life of components.

4.3.1 Cooling Design Criteria

There were not many design requirements for the cooling system, though the requirements were vital to system design. When designing the cooling system, these requirements, as well as the various design norms following a Christian perspective, needed to be kept in mind.

4.3.1.1 Cooling Design Requirements

- The cooling system must keep the Nvidia Jetsons at or below 60°C
- The cooling system must be able to be used to teach the impact that CPU/GPU temperature has on processing speed

4.3.1.2 Transparency

Once again, open access is an integral part of this project. The CAD files for the water block design needed to be available so that the cooling system could be replicated or evaluated by anyone who desired to do so. Along with this, the cooling capacity of the system was not overemphasized.

4.3.1.3 Caring and Trust

Since liquid cooling was implemented, the lifespan of the Nvidia Jetsons will increase due to decreased operating temperatures. Prolonging the life of these units saves the user future costs, such as the cost to replace an Nvidia Jetson. Also, the cooling loop contains a piece of silver coil within the reservoir to deter organic buildup in the system. This was intended to decrease maintenance time and increase operating efficiency. Similarly, the liquid cooling system needed to be robustly made and adequately sealed to prevent any liquid from leaking into the case.

4.3.2 Cooling Design Alternatives

For the cooling system, the question on whether the system should be air cooled or water cooled was raised. Air cooling is an inexpensive and nearly maintenance-free alternative, but is limited to the amount of cooling it can offer. Water cooling is more expensive and requires some maintenance over time but provides significantly higher cooling potential and quieter operation.

Table 3. Decision Matrix for Determining Cooling System

Criteria	Weight	Aluminum Heatsink	Copper Heat Pipes	Water Cooling Loop
Cooling Potential	9	6	8	10
Price	4	10	6	4
Educational Value	8	2	5	7
Aesthetic	7	3	6	9
Lifetime	4	10	9	4
Weighted Scores		171	214	241

4.3.3 Cooling Design Decisions

For the cooling system, a water cooling design was chosen with the aid of a decision matrix (See **Table 3).** Water cooling provides high cooling potential within a smaller volume than air cooling. It also allows for near silent operation and an overall better aesthetic. The water cooling system required water blocks for the 5 Jetson CPU/GPU dies, a pump, plastic tubing and drainage valve, a metal radiator and fan, and a reservoir to hold the cooling fluid. The pump, radiator, fan, drainage valve and tubing were purchased, while the water blocks, drainage valve pipe, and reservoir were fabricated by the mechanical team. Using blocks of aluminum, clear polycarbonate, rubber sheets, and a milling machine, the water blocks were custom fabricated for the Nvidia Jetsons. The reservoir was also fabricated in-house and integrated into the front panel area of the case. The pump was mounted to the base of the case and the radiator and fan combo were mounted onto the upper internal skeleton of the case. The flow of the cooling system can be seen in **Figure 36**, below.

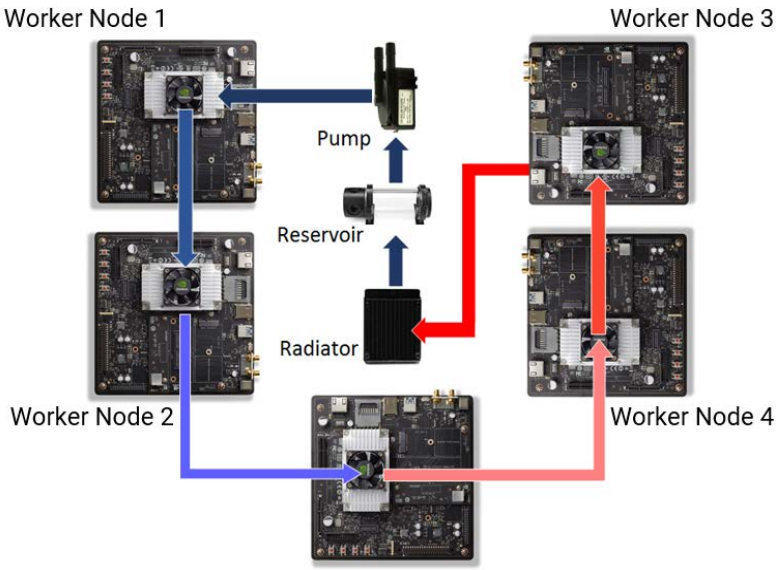


Figure 36. Water Cooling Flow Design [17]

4.3.4 Cooling Loop Fabrication

4.3.4.1 Water Cooling Block Prototyping

The cooling (or water) block was designed using an iterative process of CAD work and 3D printing. After the first cooling block was designed, it was 3D printed. Using the 3D printed prototype, major changes were made. The cooling block size was increased, and a different method of securing the block to the Nvidia Jetson was designed, see **Figure 37** and **Figure 38** to compare the cooling block prototypes. Then, the second-generation prototype was 3D printed, and again changes were made. This time, additional holes were created for fasteners to secure the polycarbonate layer more tightly, and the size of the block was slightly modified. After this block was printed, changes were made once more. Moving to the fourth-generation water block, fastener hole sizes were decreased, and all holes were changed to the same size. Again though, additional changes were made to the fourth generation to provide clearance for the tube fittings and the height of the block was slightly increased. The fifth-generation cooling block was the final result of the prototyping phase, see **Figure 39**.

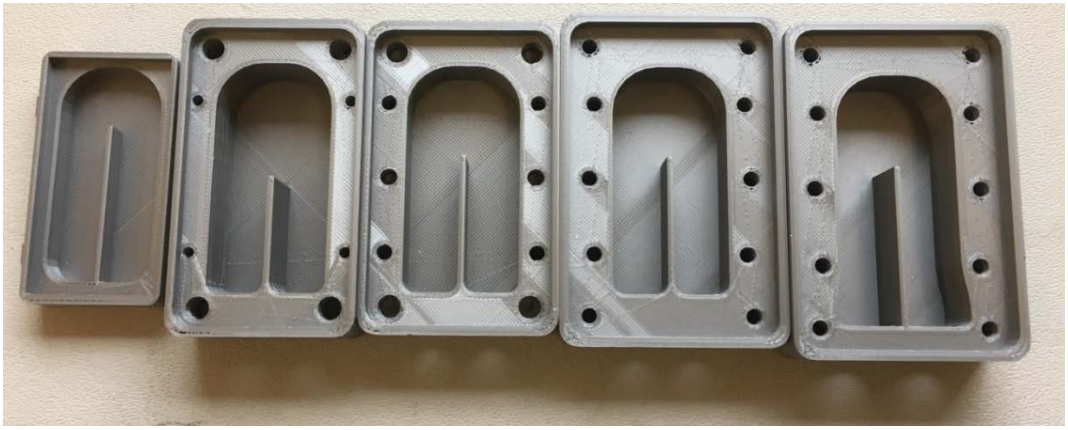


Figure 37. Cooling Block Prototypes Top (Generations L-to-R)

Figure 38. Cooling Block Prototypes Front (Generations L-to-R)

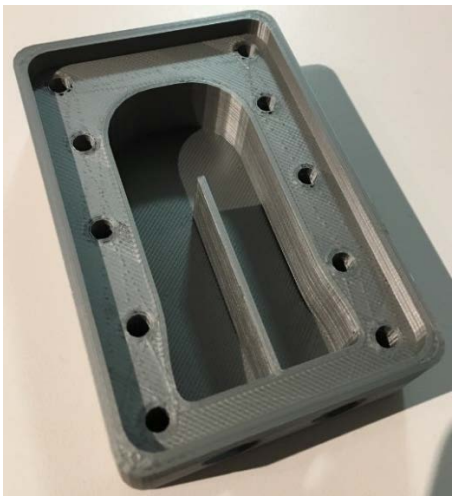


Figure 39. Fifth Generation Cooling Block

4.3.4.2 Water Cooling Block Fabrication

After finalizing the water block design through the 3D printing process, the water block was created out of a 4-inch by 1-inch aluminum bar. Along with the aluminum block, a rubber gasket, a polycarbonate cover, and an aluminum force distributor bracket were all created, which are pictured right to left in **Figure 40**.

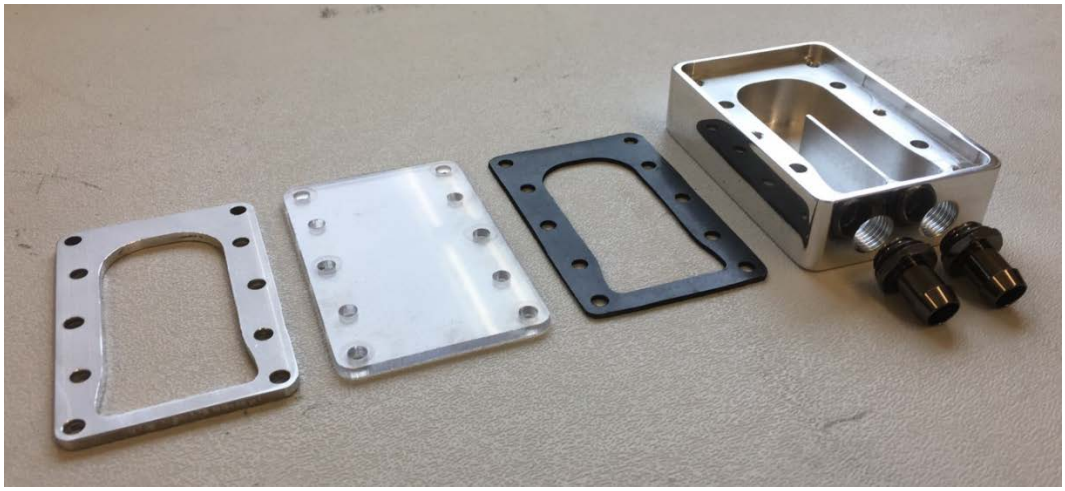


Figure 40. Cooling Block Components

CAM software within AutoDesk Inventor was used to program the milling and drilling operations needed for the HAAS mill to CNC the aluminum bar into the water block design. After the mill was used to create the aluminum block, secondary operations, including sawing, de-burring, grinding, sanding, internal threading, and drilling were necessary to finish the blocks. The fitting holes needed to be drilled and threaded, and the fastener holes needed to be threaded as well. The fasteners used to secure the polycarbonate cover and gasket to the aluminum block were #10-32 screws. The fittings used in the fluid entry and exit on the block were standard PC cooling fittings with G1/4-inch threading. In addition to this, some holes needed to be re-drilled for certain blocks, and programming was updated for the CNC to avoid this on remaining blocks.

Coarse and fine sandpapers were used both on the outside and inside of the water blocks to smooth them out. The water block going on the head node was polished in addition to sanding. A finished aluminum block is shown in **Figure 41.**

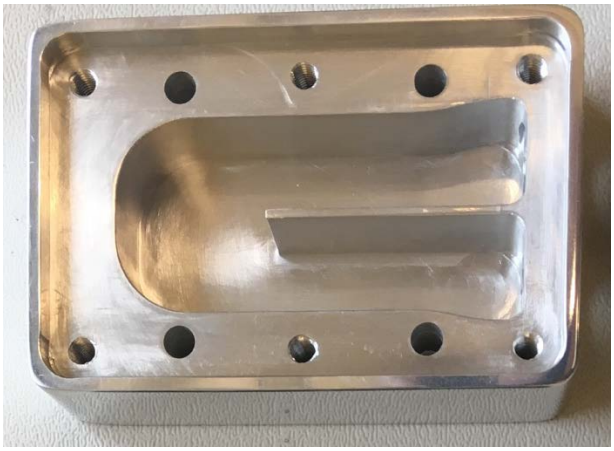


Figure 41. Finished Aluminum Block

The polycarbonate cover was created after the aluminum blocks were made. To design the cover, AutoDesk Inventor was used, and CAM was used to program the HAAS mill for CNC operations. A 1/8-inch-thick polycarbonate sheet was secured to an aluminum fixture so that the mill could precisely cut it. After the first cover was made, the program was adjusted to account for oversizing tolerance issues, as the polycarbonate cover did not fit into the cooling block. After adjusting the program, the remaining covers fit into the cooling blocks well. See **Figure 42** for the polycarbonate cover.

Figure 42. Polycarbonate Cover

After creating the polycarbonate covers, rubber gaskets were made out of 1/16-inch rubber to stop waterflow from exiting between the aluminum and polycarbonate layers. The gaskets were initially attempted by hand, but due to the complexity of the design, they were created using the CNC function on the HAAS mill. Again, AutoDesk Inventor was used to create the gasket in CAD, and CAM was used to program the mill. The rubber sheet was secured to an aluminum fixture using the piece of wood shown in **Figure 43**; this compressed the rubber so that more precise cuts could be made. The mill was then used to cut out the shape of the gasket and drill the holes in the gasket. A hobby knife was used to cut off excess rubber remaining on the

gasket. A spare polycarbonate cover was also used as a stencil to shape the gasket further to account for tolerance oversizing issues. A finished rubber gasket can be seen in **Figure 44**.

Figure 43. Wood Fixture

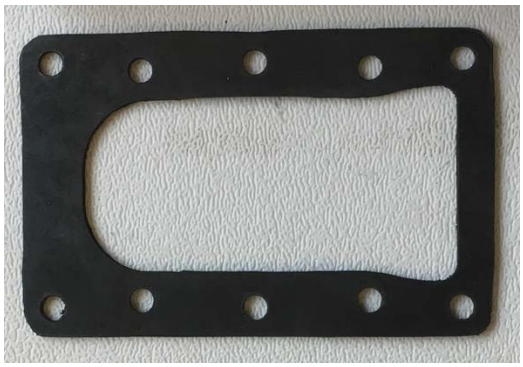


Figure 44. Rubber Gasket

To secure the gasket between the polycarbonate cover and aluminum block, the aluminum force distributor bracket was created. At first, a water block was tested by just securing the polycarbonate with six screws and washers, but the block leaked due to the polycarbonate bowing and not compressing the gasket. Then, the aluminum force distributors were made in AutoDESK Inventor and CAM was used to program the HAAS mill once again. A 3-inch by 3-inch cross-section hollow aluminum bar was secured to the mill and the first bracket was created. However, the mill needed to be reprogrammed due to oversizing from tolerances once again. After milling the new brackets, they were de-burred and sanded with both coarse and fine sandpapers. A finished aluminum force distributor bracket is shown in **Figure 45.**

Figure 45. Aluminum Force Distributor Bracket

After each of the parts needed to create the complete cooling block were fabricated, the gasket was placed in the aluminum block, the polycarbonate cover was placed on top of the gasket, and the aluminum bracket was placed on top of the polycarbonate cover. Six #10-32 screws were used to fasten these three layers to the aluminum block. Plumbing tape was wrapped around the threads of the tube fittings before being screwed into each aluminum water block. A wrench was used to tighten the fittings to ensure a watertight connection. For the loosely fit cooling block without screws and plumbing tape (though not visible anyway), see **Figure 46.**

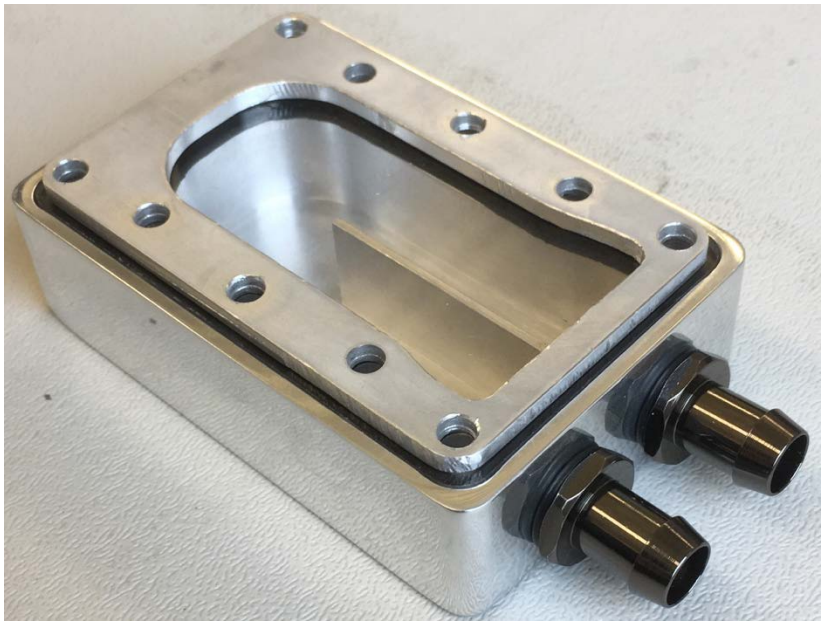


Figure 46. Assembled Water Cooling Block

4.3.4.3 Reservoir Fabrication

The reservoir needed to store water for the system was created using ¼-inch-thick polycarbonate and plastic adhesives. The polycarbonate was cut into six rectangular sections using a table saw. All the sections besides the top were glued together using a thin plastic glue, called Proweld. After this, a thick Loctite adhesive was used on the seams inside and outside of the reservoir. The reservoir was tested and any places where leaks formed were sealed using

the thick Loctite adhesive. Holes were drilled and threaded for the top section and the face of the reservoir with the fluid exit fitting. Later, an additional hole was drilled and threaded to move the fluid entry fitting to the same side as the exit fitting. The top section was placed on the reservoir and was sealed using the Loctite. Plugs were placed in the top holes of the reservoir. Various testing was done using the water blocks and pump along with the reservoir. Additional adhesive was placed where leaks occurred and where leaks did not occur to ensure a watertight reservoir. The finished reservoir can be seen in **Figure 47.**

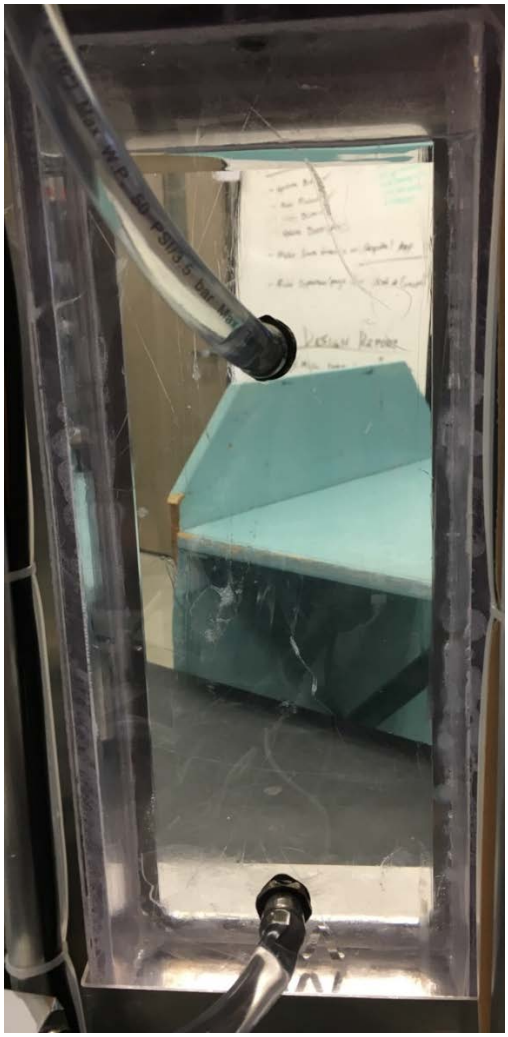


Figure 47. Water Reservoir

4.3.5 Cooling Loop Installation

The cooling components were linked together using clear food-grade tubing with an inner diameter of 5/16-inch. The reservoir was connected to the pump, which was then connected to the drain valve. The drain valve connected to the first water block, the four blocks after that were connected in series, and the fifth block was connected to the radiator in addition to the fourth block. The radiator was then connected to the reservoir to close the cooling loop. The

reservoir was filled with distilled water using a funnel and tubing through the top port. The antimicrobial silver coil was placed in the reservoir prior to plugging the top of the reservoir.

The cooling blocks were secured to the Nvidia Jetsons after installing them into the case and running liquid through the system to ensure no further leaks occurred. The blocks were installed in several steps. First, the stock heat sink and fan were removed from the Nvidia Jetson, as shown in **Figure 48**. The thermal paste on the Nvidia Jetson plate housing the processors, see **Figure 49**, was then wiped off using 91% isopropyl alcohol. The bottom of the water block was wiped with isopropyl alcohol as well. Then, thermal paste was spread on the Nvidia Jetson plate, see **Figure 50**. The block was then placed on the plate and was secured using the screws from the stock heat sink and fan, see **Figure 51**. **Figure 52** shows the head node block secured to its Nvidia Jetson.

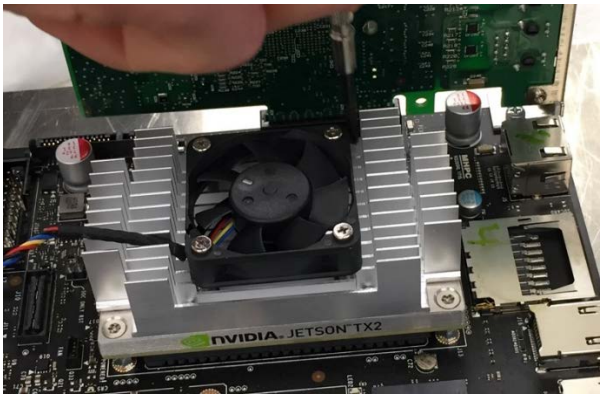


Figure 48. Stock Cooler Removal

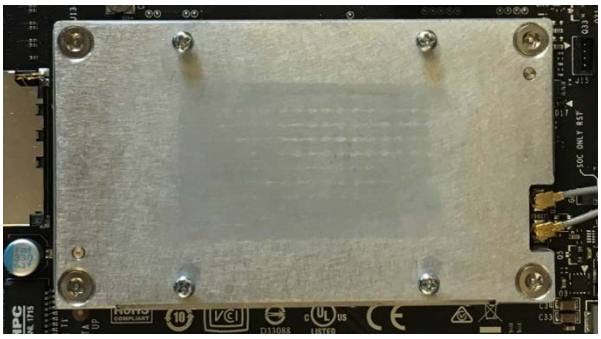


Figure 49. Nvidia Jetson Plate

Figure 50. Thermal Paste Application

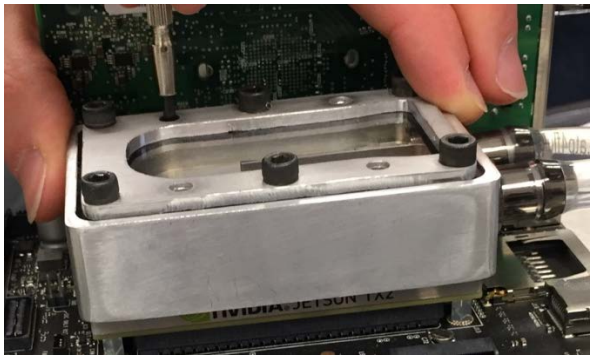


Figure 51. Water Block Attachment to Nvidia Jetson

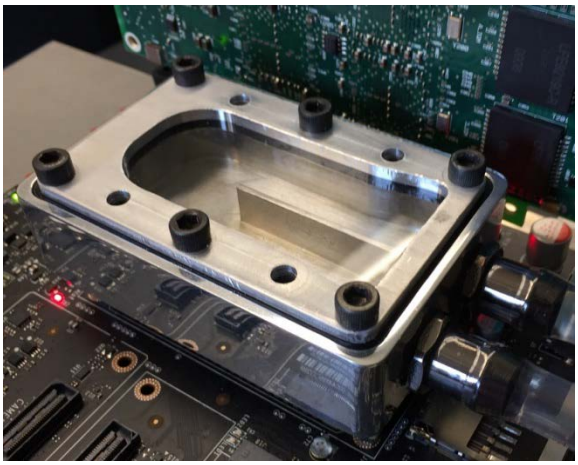


Figure 52. Head Node Cooling Block

4.4 Encrypted Command Shell Design

4.4.1 Encrypted Command Shell Criteria

The command shell needed to keep the system secure and not reveal any information as instructions run on it. As a point of interest, the encryption scheme needed to be post-quantum secure. This security cannot be easily broken by quantum computers. The added security runs fast enough as to not be noticeable. The initial connection takes no longer than two seconds, similar to establishing an SSH connection. Sending commands and receiving output takes no longer than a tenth of a second, close to the reaction speed of humans.

4.4.2 Encrypted Command Shell Design Alternatives

Symmetric key cryptography uses the same randomly generated key for both encryption and decryption. Doing this allows for small key sizes compared to the level of security. However, this is not always usable as the parties communicating still need a way to communicate the private key securely.

Lattice based cryptography generates keys with the lattices used in machine learning. Since the problem is not well understood there is no well-known way to invert the operation. The downside of this scheme is that keys are several kilobytes long, which is a burden when trying to send keys quickly.

Multivariate cryptography is based on the difficulty of solving multivariable equations. Unfortunately, the theory on this system is not fully developed, and the keys are over 100 kB in length.

Hash based quantum secure cryptography is based on complex hashing functions. The drawback of this scheme is there is a limited number of keys available for encryption.

4.4.3 Encrypted Command Shell Design Decisions

The end goal of the work in cryptography was to use it to secure the head node's connection to the internet. The initial prototype encrypts using sockets in a C application for the actual communication.

Development on supersingular elliptic curve isogeny cryptography started from Microsoft's implementation. A simplified map of the functions used for generating a key is shown in **Figure 53**, and identifies which functions may be easily sped up. Much of the computation relies on functions that have been written in assembly. Using vectorized instructions would theoretically sped up the computation, however they cannot be efficiently used for large integer operations and so Microsoft's use of 64-bit instructions is the fastest known method as they require access to a carry flag. In the end, Microsoft's unmodified solution was used for the encrypted shell.

To make use of the cryptographic key, the data being sent is ciphered with it. The chosen cipher for this implementation was RC5. RC5 allows for a variable length key, which is required to use the over-700-bytes-long key. Using this, both parties use their established private secret to communicate over a public network.

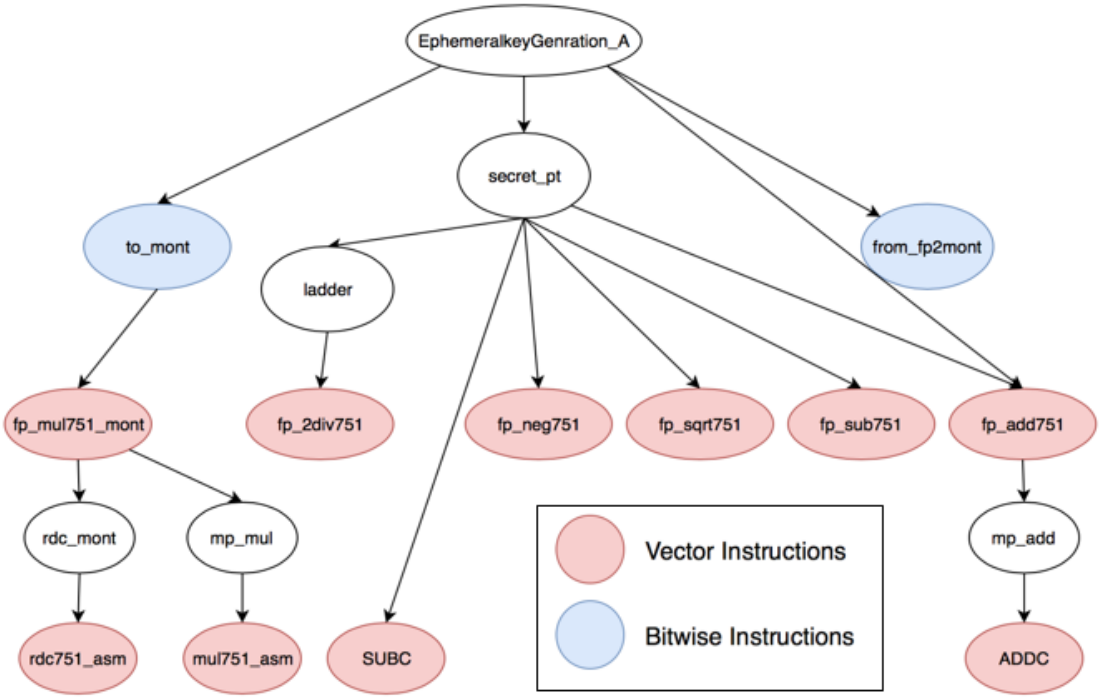


Figure 53. Key Generation Function Call Tree and Suggested Speedups [18]

4.5 Auto Power-On Design

4.5.1 Auto Power On Criteria

To avoid reaching into the case and pressing the separate power buttons on each board, a solution for powering on the cluster when power is supplied was required.

4.5.2 Auto Power on Design Alternatives

A software solution was ideal as it would add no additional cost, however turning on from a network connection is not supported by the Jetsons. So a hardware solution was required. Nvidia includes documentation on hardware solutions available on their website. The documentation states that the Nvidia Jetson TX2 includes hardware implementing this behavior by connecting one pin to ground. [19] However, the documentation itself and several official Nvidia forum posts give conflicting reports on whether this is enough. For another plan, Nvidia has a diagram of a circuit that can be connected over the terminals of the power switch to turn on the boards several seconds after power being supplied. This was the solution chosen. In testing it was found to not function as expected.

4.5.3 Auto Power on Design Decisions

The auto power on circuit is shown in **Figure 54**. It includes two MOSFETs, two diodes and several capacitors and resistors. This makes the circuits inexpensive to build. The circuit can

also be constructed on a single layer making it possible to fabricate at Calvin College. The circuit was simulated, and it was verified to provide a high voltage signal over the power switch after three seconds. The simulated output can be seen in **Figure 55**.

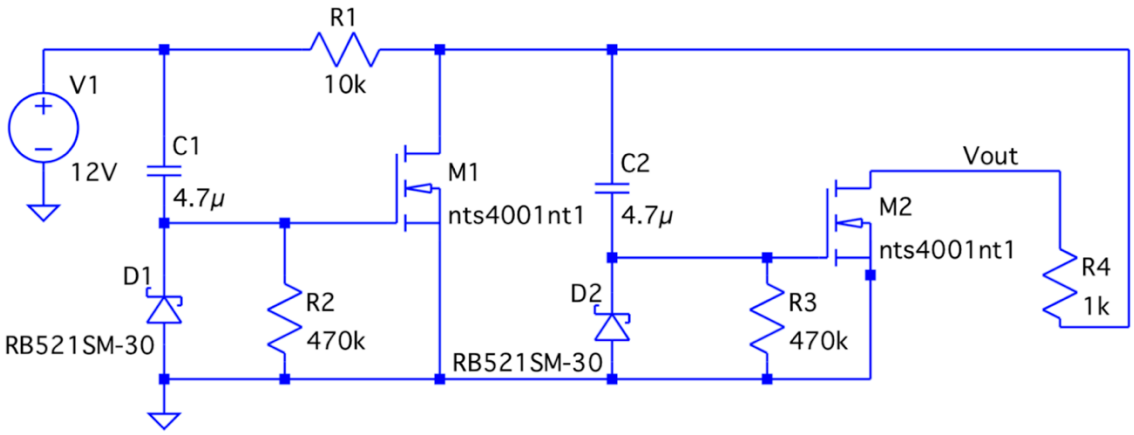


Figure 54. Power Delay Circuit

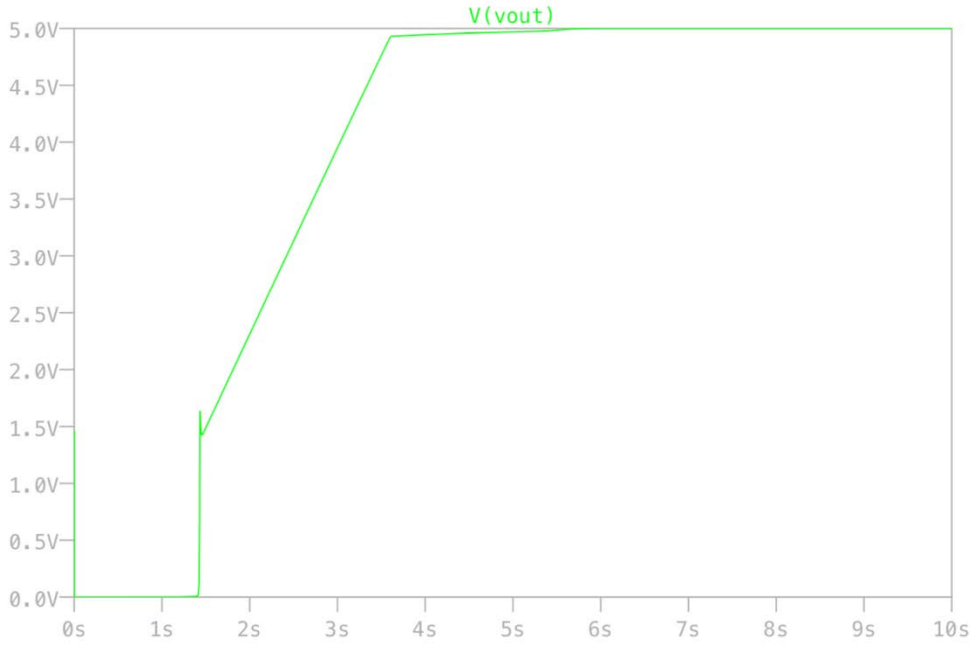


Figure 55. Transient Response of Power Delay Circuit

To print out the board a separate file is needed and is shown in **Figure 56**. The board layout is identical to the similar circuit except it is reflected to allow the MOSFETs pins to be in the correct orientation.

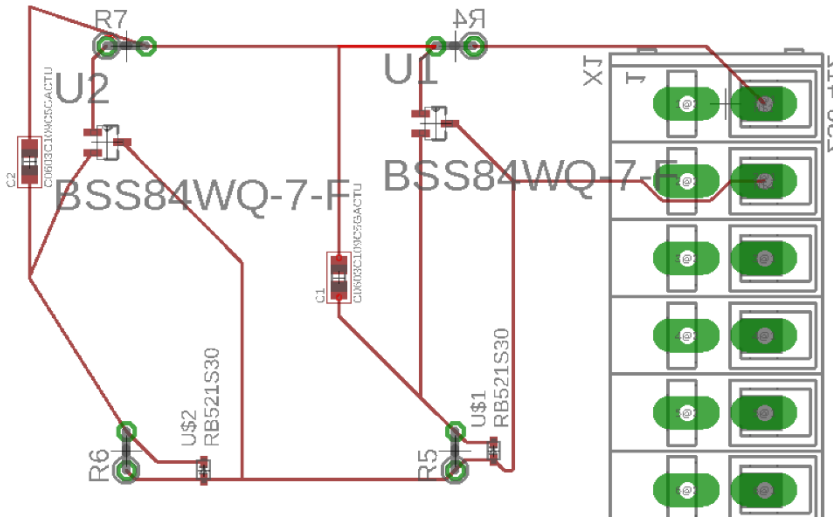


Figure 56. Board Layout for Fabrication

The fabricated circuit can be seen below as **Figure 57**.

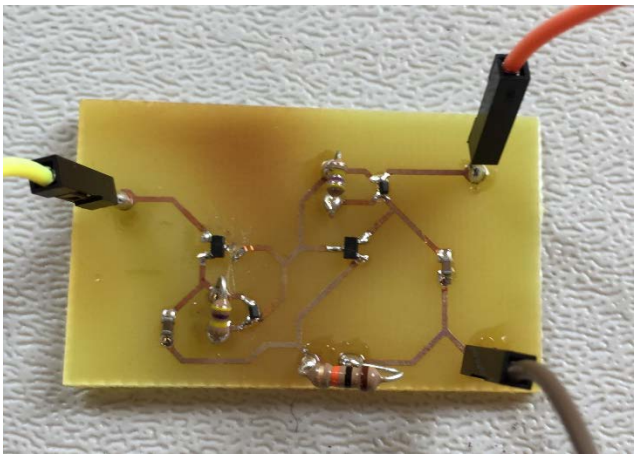


Figure 57. Fabricated Power-on Circuit

4.6 Lighting Design

4.6.1 Lighting Criteria

The lighting system needs to display temperature and processor usage information for the overall system and per board.

4.6.2 Lighting Design Alternatives

Many computer component companies produce programmable RGB strips. They all offer some amount of programmability, but they are limited in features, and only programmable through

their proprietary GUI. Using an Arduino gives the full programmability of a C program, while also being slightly less expensive.

After deciding on an Arduino, one of the three RGB strips made for them had to be decided on. DotStar produces single color LED strips; this limits the amount of information displayable. RGB plus produces analog RGB LEDs for Arduinos. They come at $20 per meter and have separate drivers for red, green, blue, and white levels. Neopixel has a RGB LED strip that is digitally controlled; they are $25 per meter, and require one connection directly to the Arduino. The size of our system calls for two meters of LEDs and so would cost $40 for analog LEDs and $50 for digital ones. The digital strips were selected as the ease to setup and fewer connections needed in the case was worth the extra cost.

4.6.3 Lighting Design Decisions

An Arduino was selected as it gives the highest level of programmability at a lower cost than a commercial product. To send data to the Neopixel RGB strip the data lines needs to be connected to a GPIO pin on the Arduino. A resistor needs to be on the data line to prevent damage to the LEDs from spikes on the line while turning on. To power the LEDs, they are connected to a 5V line on the system's PSU. A decoupling capacitor is added over the 5V connector. A diagram of this system can be seen in **Figure 58**.

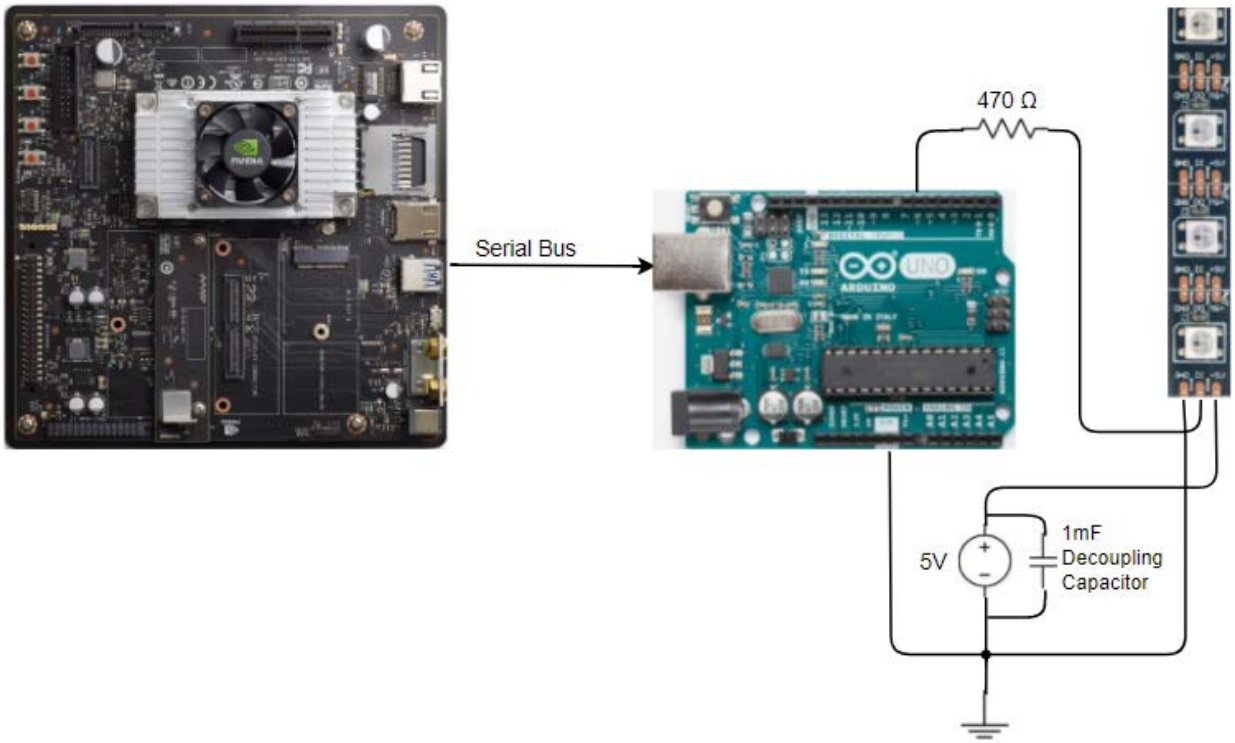


Figure 58. Lighting Design Diagram [20] [21]

For software, one Jetson must gather all the system temperatures, and processor usages and send the information to the Arduino. After the Arduino has received the data, it selects the

information to display and formats it for the LEDs. To display temperature, LEDs associated with each board can follow a gradient from blue to red as the processors' temperature changes from 30°C to 70°C. Processor usage can be displayed for every CPU core of every Jetson, as the number of processors is equal to half of the Neopixels on each meter. More usage simply translates to a higher intensity of white light. A similar approach can display the usage of the board's GPUs displaying higher intensity for more usage. Both can also be displayed at the same time as the CPUs can take up the first half of the LEDs assigned to each board, and the GPU can be displayed on the other half. To separate the two processors, blue will represent the ARM processor as that is the color used by their company, and green for the Nvidia GPU.

The ideal position for the LED strips for displaying information would be on the outside of each panel to maximize visibility. However, this takes away from there being any internal lighting in the case. Outside lighting would involve routing wires outside the case raising issues with the Cray design of the system. As a compromise, we attached one meter of lights to the top ventilation so that it is visible outside of the case and illuminates the inside. The other meter of lights was split up on the support posts of the case.

4.7 Beowulf Cluster Design

4.7.1 Beowulf Cluster Criteria

The computer cluster was designed to match the specifications laid out by the computer science department.

4.7.1.1 Requirements

- The cluster must consist of 5 Nvidia Jetsons connected together
- The system must use a single computer PSU to power the system
- CUDA, MPI, and NFS must be installed
- The nodes must communicate in a master-worker relationship
- The worker nodes must be able to mount shared storage via NFS
- The nodes must utilize multiple network interfaces efficiently

4.7.2 Beowulf Cluster Design Alternatives

4.7.2.1 Operating systems

It does not seem wise to adapt an alternative operating system to the Jetsons. The default Linux based operating system is likely the best choice, as Windows would entail unnecessary complications.

4.7.2.2 Jetson Model

The Jetson TX2 is the newest model [22]. Its predecessors are the Jetson TX1 [22] and, before that, the TK1 [23]

4.7.2.3 Shared storage medium

The shared storage medium could either be a USB flash drive, SATA hard drive, or SD card attached to the head node. More can be stored on a USB flash drive, but the SD card will keep the system less cluttered on the inside. The SD card would also be more securely attached which will be relevant in a moving case.

4.7.2.4 Operating system storage

The operating system could be stored on an SD card instead of the on-board eMMC. The worker nodes could run their operating system by network booting, which appears to be possible on the Jetson TX1 [24] although this might have to be done via a PCIe network card [25]. The SD Card allows for expandability which may be necessary for future application of the system.

It is also possible to clone the image from the eMMC on one node and flash that image to the other nodes to avoid repeating many configuration steps [26] [27]. It is also possible to switch to SD cards later by copying the image from the eMMC to the SD card, at least on the Jetson TX1 [28].

4.7.2.5 Networking

Professor Joel Adams mentioned in personal communication that the limiting factor of Microwulf was the speed of the network interconnects (Gigabit ethernet). It was assumed that the Jetson CPUs are more powerful than the CPUs on Microwulf (they are several years newer), which meant that Gigabit ethernet would be even more limiting, although, as the team researched near the end of the project, the clock speed of the Jetson CPU is 2Ghz [29] while the clock speed of the AMD Athlon 64 X2 3800+ AM2 CPUs used by Microwulf [30] is also 2Ghz [31]. Granted, the Jetson's do have a discrete GPU, while Microwulf does not. Other researchers, using a TX1, suggested that networking would become a bottleneck [32]. Other networking technologies (e.g. InfiniBand or 10 Gigabit Ethernet) do exist.

Chris Wieringa mentioned in personal communication that channel bonding enables multiple interfaces to share the same IP address and therefore increase throughput. He informed the team that various modes of channel bonding exist, some of which can be used with unmanaged ("dumb") switches (e.g. mode 6 channel bonding) and others which require managed ("smart") switches. Channel bonding requires the physical installation of additional network interfaces on the Jetsons, since each Jetson has only one Ethernet port.

4.7.2.6 Remote Login Protocols

SSH (secure shell) is commonly used for remote logins but suffers from overhead due to the encryption necessary for security. RSH (remote shell) performs the same function as SSH but performs little authentication or encryption. This results in a speed increase, as Professor Adams mentioned in personal communication. SSH and RSH are not mutually exclusive – both can be operational at the same time.

4.7.3 Beowulf Cluster Design Decisions

Configuration and setup files are available in the Crayowulf GitHub organization[1]. Paths to the mentioned configuration files on GitHub are given in footnotes.

4.7.3.1 Operating System

The cluster runs a modified [33] version of Ubuntu for its operating system called Linux For Tegra, provided by Nvidia. Linux for Tegra R28.1 [34] was the current version when the project began. It corresponds to Jetpack 3.1 [35] (Jetpack is the program used for flashing the boards and installing software such as CUDA). During the project, Linux for Tegra R28.2 was released, but R28.1 was used in an attempt to be consistent. The factory image currently available for the Jetsons is R28.1 [36], but Jetpack 3.2 will flash the OS with R28.2 if the OS option is selected [37]. Anyone reproducing the project is advised to use the most current version of Linux for Tegra and Jetpack. This will likely require the user to select the option to flash the OS using Jetpack. During the setup process, backup images were made of the system following the instructions from [27].

4.7.3.2 Jetson Model

The Jetson TX2 was chosen because it is the newest model, with the most modern components.

4.7.3.3 Operating System Storage

The operating system was installed on the eMMC chip on each node. Network booting was not used because of the added complexity. Using the eMMC chip meant that SD cards did not have to be purchased, resulting in a small cost savings.

4.7.3.4 Networking

4.7.3.4.1 Channel bonding

Other networking technologies were not chosen because they were prohibitively expensive. Worker nodes have a 2-port PCIe expansion card installed and the head node has a 4-port card. The carrier board that the Jetson module is installed on has a 4-lane PCIe connector [38].

[1] https://github.com/crayowulf

The head node uses the onboard interface to communicate with the internet and use the 4-port expansion card for communication with the worker nodes.

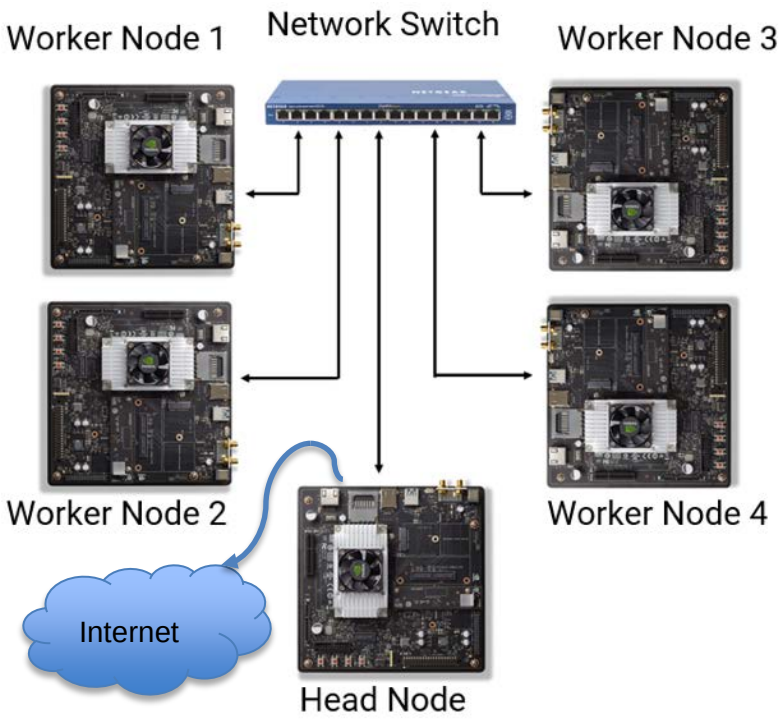


Figure 59. Node Configuration

A 16-port unmanaged network switch was purchased, which was intended to be used with mode 6 channel bonding. An unmanaged switch was chosen due to cost considerations, although in retrospect more research could have been done. According to [39], the driver for the network card must have the `set_mac_address` function for mode 6 bonding to work. Research was performed on all the networking drivers in the kernel source for Linux for Tegra [34] and the `set_mac_address` function was found in several of them, so it was assumed that bonding support would be available. Christopher Wieringa recommended purchasing Intel network cards because he had had better experiences with them than the StarTech or Realtek cards the team was considering, and that they had good Linux support. He was also able to inform the team that 9000k was a standard size for Jumbo frames.

Unfortunately, mode 6 channel bonding requires the bonding kernel module to function. Bonding was attempted, following the instructions from [39] The bonding kernel module was not precompiled on the version of Ubuntu installed on the boards, which, after some effort to compile it, resulted in the channel bonding feature being de-prioritized in favor of completing the rest of the setup tasks. The PCIe expansion cards are still being used, although only one port per card is in use. It is possible that channel bonding could be fully implemented by a future student (see Section 7)

As part of the troubleshooting process for enabling an Arduino to communicate with the Jetsons, the CONFIG_USB_FTDI_ELAN [40] and CONFIG_USB_SERIAL_CH341 [41] modules were

compiled on worker03 using the instructions and code from [42]. The bonding module was also compiled at that time on worker01 and worker03 using instructions and code from [42]. The files used for this are in the `buildJetsonTX2Kernel` repository in the Crayowulf GitHub organization. It made sense to compile the bonding module if other modules were also being compiled.

4.7.3.4.2 DHCP

Each worker node acquires an IP address at boot via DHCP. Christopher Wieringa recommended using DHCP as opposed to simply using `/etc/hosts`. DHCP was set up and configured with help from the following sources: [43] [44] [45] [46] [47] [48] [49] [50] [51] [52] [53] [54]. It was necessary to modify the files `dhcpd.conf`[2] and `isc-dhcp-server`.[3] The IP addresses are assigned by the MAC addresses of the interfaces on the PCIe expansion cards. This guarantees that each node will always get the same IP address, which is useful for troubleshooting and is necessary for the DNS setup. The IP addresses inside the cluster are in the form 192.168.2.x where `x` is a number in the range 1-5. The range 192.168.2.150-192.168.2.200 is dynamically assigned, so any device connected to the switch will receive an IP address. This is useful for troubleshooting, e.g. connecting a laptop, as Chris Wieringa mentioned in personal communication.

4.7.3.4.3 IP Masquerading

The head node is the only node in the cluster connected to the internet on a regular basis. It connects to the internet via the onboard port. The other nodes are connected to the internet temporarily during setup (See Appendix C). Various sources were useful during the setup process [43] [55] [56]. It was necessary to modify `rc.local`[4] to enable IP masquerading.

4.7.3.4.4 Firewall

The head node runs a set of iptables firewall rules to prevent outside access. The interface eth0 is the onboard port (not the PCIe card) and is the only port on the Jetson that ever connects to the outside world (Figure 7). The iptables rules[5] disallow any connections to the cluster that are not initiated by the cluster, disallow any SSH traffic unless it originates from Calvin's campus, and finally blocks all other traffic on eth0. These rules should not affect internet access in a new location, provided the new location does not have other measures (e.g. extra firewall rules at a school) that block internet access. They may need to be modifed to allow SSH connections from other IP ranges if the cluster is demonstrated at a different location, such as a high school. All traffic on the loopback adapter is allowed and all traffic on the wireless adapter is blocked. Iptables was set up with help from [56] [57] [58] [59] [60] [61] [62] [63] [64] [65] [66] [52] [67] [68].

[2] files/head/dhcpd.conf
[3] files/head/isc-dhcp-server
[4] files/head/rc.local
[5] files/head/iptables.rules

Initially a pair of scripts was used to ensure iptables rules persisted across reboots [69], but Christopher Wieringa recommended (in personal communication) using iptables-persistent [70]. He was also able to advise that there was not much risk of iptables conflicting with the Ubuntu Network Manager, despite a line in the Ubuntu documentation warning of that possibility [57].

4.7.3.4.5 DNS

To enable the worker nodes to look each other up without requiring all their hostnames to be in the `/etc/hosts`[6] file (Note: the head node is placed in `/etc/hosts` because the worker nodes do not seem able to access DNS during the boot process), the head node runs a Dnsmasq server as recommended by Christopher Wieringa in personal communication. This allows the user to specify a hostname such as `worker03` rather than typing `192.168.2.4`. Dnsmasq listens only on `eth1` (The first interface on the 4-port card on the head node). See `dnsmasq.conf`[7] for more information.

Dnsmasq looks at `/etc/hosts`[8] on the head node for the mappings from IP address to hostname. This is why it is necessary that the same IP address is always assigned to the same host by DHCP. The head node is named `headNode`. The worker nodes are named `worker01` – `worker04`. The head node can be thought of as `worker00`, but it must be referred to as `headNode`, as that is its hostname.

An issue that required a fair amount of troubleshooting was the failure of the `dnsmasq` service to start upon boot. This was resolved by adding a line to `rc.local`[9] following the example of a solution to a similar problem involving a different service [71]. Dnsmasq was set up with help from [72] [73] [74] [75] [76] [77] [78] [79].

4.7.3.4.6 NTP

The head node runs an NTP (Network Time Protocol) server to ensure that the clocks on all the nodes stay in sync, as recommended by Christopher Wieringa in personal communication. The worker nodes all contact the server for the proper time. The time zone is set to `America/Detroit`. NTP was set up with help from the following sources: [80] [81] [82] [83] [84] [85] [86] [87] [88] [89] and personal communication with Christopher Wieringa. NTP was first enabled using the broadcast options in `/etc/ntp.conf`[10] but was later switched to a more secure method restricting access to only the cluster subnet using the restrict option on the head node[11] and using the server option on the worker nodes[12] to tell them which server to get the current time from.

---

[6] files/worker/hosts
[7] files/head/dnsmasq.conf
[8] files/head/hosts
[9] files/head/rc.local
[10] files/head/ntp.conf
[11] files/head/ntp.conf
[12] files/worker/ntp.conf

4.7.3.5 Remote Login Protocols

SSH is set up to use public key authentication – a password is not enough, as Christoper Wieringa advised in personal communication. See `sshd_config`.[13] SSH can also be used within the cluster. Public key authentication is required for the configuration management tool, Ansible, to function. SSH was set up with help from the following sources: [90] [91] [92] [93] [94] [95] [96] [97] [98] [99] [100] [101] [102] [103] [104] [105] [106] [107] and personal communication with Chris Wieringa, who also recommended disabling SSH root login to force users to use sudo if they need root access, as well as providing other assistance. A new user will need to use a flash drive to place his or her public key on the cluster in `~/.ssh/authorized_keys`, or a member of the `sudo` group can perform this action.

Remote shell, or RSH, can also be used within the cluster but not from outside due to the requirement of public key authentication on SSH. RSH was recommended by Professor Adams in personal communication because using it for computations avoids the time penalty from the encryption and decryption used by SSH. The interior of the cluster is considered a secure environment, so encrypted communications are not as important. A `.rhosts`[14] file must be placed in each user's home directory to enable RSH for that user. The `.rhosts` file must have 644 permissions according to the manual page for `rhosts` [108]. RSH is not available from the outside of the cluster – this was tested by creating a brand-new user account (one without SSH access to the cluster) on a workstation and attempting to RSH into the cluster. RSH was set up with help from [109] [110] [111] [112] [113] [114] [115] [116], and personal communication with Christopher Wieringa.

4.7.3.6 NIS

NIS (Network Information Services) is running on the cluster to manage the users, as recommended by Christopher Wieringa in personal communication. Each node has two accounts pre-created during the flashing process, named `nvidia` and `ubuntu`. All other accounts are managed by NIS. The NIS server on the head node stores the user and group information and propagates it to all the worker nodes. The nodes are all part of the same NIS domain (`crayowulf.nis`). It should be noted that even though a NIS-managed user may be in a group the `/etc/groups` file on a worker node will not reflect that. NIS was set up with assistance from: [71] [117] [118] [119] [120] [121] [122] [123] [124] [125] [126] [127] [128] [129] [130] [131].

Although logging in to a NIS-managed user account via command prompt (switch to another virtual terminal e.g. Ctrl+Alt+F6) or remotely (SSH or RSH) functions properly, logging in via the GUI does not function properly on the worker nodes. One is still ablet to log into a GUI on the head node. Switching to a virtual terminal shows an error stating that "server "headNode" not responding, still trying", which according to [132] indicates a NIS-related issue. Because this

[13] files/head/sshd_config
[14] files/common/.rhosts.

issue does not affect the critical functionality of the cluster, it was deemed low priority. Distributed computations can still be run. It is still possible to log in to the `ubuntu` account on each node.

Currently, it is necessary to install NIS manually because it prompts for the NIS domain (`crayowulf.nis`) that it should join. According to Christopher Wieringa, there should be a way to automate the installation, perhaps by placing a file on the node before installing NIS, but this feature was deemed low priority.

4.7.3.7 Shared Storage & NFS

Shared storage will be on a Solid-State Drive that is attached to the head node. An SSD is fast and easy to implement. Unfortunately, the SATA controller on the Jetson is limited to SATA Gen 2 speeds [29], even though SATA Gen 3 drives are available (the SSD currently installed on the cluster is SATA Gen 3, see Appendix B). The fastest SD card when the project started had a read speed of about 300MB/s [133] [134], about the same speed as a SATA Gen 2 SSD [135]. It was also necessary to be careful which brand of SD card was purchased, as problems have encountered with some brands or models [28]. However, that was a new product while the SSD was a relatively mature product.

The only faster option than a Gen 2 SSD was an enthusiast-grade USB flash drive [136] that required the host to have specific capabilities (UASP support) to achieve the advertised speeds [137]. The chances of not reaching the advertised speeds of the flash drive seemed unacceptably high based on the team's research. It seemed safest to use a Solid-State Drive instead, even if performance is not as good as the flash drive.

The home directories for the NIS-managed users reside on the SSD. Each home directory is of the format `/home2/<username>`. Note that there is no user actually named `bin`, it is only another directory used for executable files that all the nodes need access to. The same goes for `/home2/shared_install_files`. See section 7 for a better method. `/home2` was used instead of `/home` because of the two accounts pre-created when flashing. According to personal communication with Christopher Wieringa, it is impossible to have home directories for NIS managed users and non-NIS managed users in the same location. Fortunately, as Christopher Wieringa suggested, it is possible to move the users who already have their home directories in `/home` to `/home2` by editing `/etc/passwd` and re-running the make command on NIS (`sudo make -C /var/yp/`) on the head node. When adding new users, one must use the `--home` option to the `adduser` command the specifies the home directory.

An IP address to hostname mapping was placed in /etc/hosts on all the worker nodes because the nodes did not appear to be able to access DNS during the bootup process. Success was finally achieved by adding the automount option in `/etc/fstab`. NFS was set up with help from the following sources: [80] [126] [138] [139] [140] [141] [142] [143] [144] [145] [146] [147].

4.7.3.8 OpenMPI

OpenMPI (the software used to run parallel computations [15]) is included if you use Jetpack to install all the available packages, which I recommend. Elizabeth Shoop pointed this out in personal communication. However, it did not come with CUDA support, (CUDA support is turned off by default, according to the configure script accompanying OpenMPI [148]) which made it necessary to recompile OpenMPI so that distributed computations could take advantage of the GPU.

This was done by obtaining the source for the `openmpi` Debian package from the Ubuntu repositories and rebuilding the package. To do this, the `--with-cuda` option was added to the `override_dh_auto_configure` section of the `openmpi-1.10.*/Debian/rules` file. The version number was increased to `104` and the `apt-hold` command was used to hold the packages so that they are not overridden by an upgrade later. The packages have been uploaded to the `binaries` repository in the Crayowulf GitHub organization. The entry `/usr/lib/openmpi/lib` was added to the `LD_LIBRARY_PATH` variable and the entry `/usr/lib/openmpi` was added to the `PATH` variable as mentioned by [148]. It is not necessary for the CPU-only HPL benchmark to run (see section 4.7.3.8 for more information on the HPL benchmark).

OpenMPI jobs can be started by running `mpirun --hostfile /etc/openmpi/openmpi-default-hostfile -np 30 <nameOfExecutable>` in a terminal window. OpenMPI was configured to use RSH for inter-node communication, as recommended by Professor Joel Adams.[15] It was necessary to add `/usr/local/cuda-8.0/bin` to the `PATH` variable `~/.bashrc`.[16] to allow `make` to find the `nvcc` compiler. OpenMPI was rebuilt and repackaged with assistance from [149] [150] [151] [152] [153] [154] [155, 156] [148] [157] [158] [159] [160] [161] [162] [163] [164] [165] [166] [167] [168] and personal communication with Christopher Wieringa.

4.7.3.10 HPL (Linpack) Benchmark

The entry `/usr/lib/openblas-base` was added to the variable `LD_LIBRARY_PATH` in the ~/`.bashrc`[17] file because `openblas` is a dependency for HPL.

Much effort was put into compiling the CUDA-enabled HPL benchmark. A build was obtained, but the benchmark would not run. Jonathan Bentz from NVIDIA informed the team via personal communication that `cudaHostRegister`, used by the HPL benchmark, is not supported on the ARM architecture. See section 5.1 for HPL results from the CPU-only version.

[15] files/common/openmpi-mca-params.conf
[16] files/common/.bashrc
[17] files/common/.bashrc

The following sources were helpful in successfully compiling and using the HPL Benchmark (CPU version): [169] [170] [171] and personal communication with Professor Joel Adams.

4.7.3.11 NAMD

NAMD [172] is a molecular simulation program. NAMD was developed by the Theoretical Biophysics Group in the Beckman Institute for Advanced Science and Technology at the University of Illinois at Urbana-Champaign. Team Crayowulf uses NAMD with VMD (see next section) to demo Crayowulf. Because the NAMD team does not provide precompiled binaries for the ARM architecture, it was necessary to compile NAMD from source code, requiring a great deal of additional preparation time. The `notes.txt` file accompanying NAMD [172] contains build instructions for NAMD, so only Jetson-specific notes are included here.

Charm++ [173] must be built before building NAMD. The first attempt at building Charm++ involved researching the way to include MPI support. A build was obtained, however, upon attempting to compile NAMD with CUDA support, an error message appeared stating that "`CUDA builds require non-MPI SMP or multicore Charm++ arch for reasonable performance.`" Because CUDA support was desired for this project, that attempt was abandoned.

A CUDA-enabled version of NAMD was attempted. It was failing with the CUDA-related error "`too many resources requested for launch`". The team was not able to resolve the error despite extensive research. It may be worth noting that the recommended settings for number of worker threads in NAMD is `#ofCoresPerNode/#ofGPUsPerNode – 1` [174]. This means that on a cluster with 1 GPU per node, using that equation will involve division by zero. Further research is necessary.

For the Production version of NAMD (used for successful demonstrations), the TCL and FFTW libraries from the NAMD website [175] were used, specifically the `fftw-linux-arm64.tar.gz` and `tcl8.5.9-linux-arm64-threaded.tar.gz` files. These files were extracted and the resulting folders were renamed to `~/namd-buildProduction/NAMD_2.12_Source/fftw` and `~/namd-buildProduction/NAMD_2.12_Source/tcl`, respectively. Charm++ for the build of NAMD currently in use was compiled with the command `./build charm++ net-linux-arm7 -j4 --with-production`. NAMD was configured with the command `./config Linux-ARM64-g++ --charm-arch net-linux-arm7`. This is a non-CUDA build of charm that uses the CPU only. The `namd2` binary is currently located in `/home2/ben/namd-buildProduction/NAMD_2.12_Source/Linux-ARM64-g++`.

NAMD defaults to using SSH [176], but as previously mentioned, RSH was desired. Both options mentioned by Jim Phillips [176] were tried without success, so SSH is being used instead.

University of Illinois at Urbana-Champaign provides a helpful tutorial on the basics of NAMD, along with input/configuration files for the simulations [177].

Helpful sources regarding NAMD: [178] [179] [180] [181].

4.7.3.12 VMD

VMD is a molecular visualization program, which can be used with NAMD [182] [183]. VMD was developed by the Theoretical and Computational Biophysics Group in the Beckman Institute for Advanced Science and Technology at the University of Illinois at Urbana-Champaign. VMD can be used to display a visual representation of the molecules that are being simulated by NAMD. It was necessary to build VMD from source as precompiled binaries are not available for the ARM architecture. It was possible to build VMD with CUDA support.

An MPI build of VMD was attempted and achieved but because it was not quite clear what parts of VMD were using MPI, that effort was abandoned.

4.7.3.12.1 Building VMD from Source

First, one must build the VMD plugins. The `tcl` directory was copied from the NAMD directory into the VMD `lib` directory (the top level of the VMD build directory contains `lib`, `plugins`, `vmd-1.9.3` and `vmd-1.9.3.src.tar.gz`).

Note that these libraries include Tcl version 8.5 but version 8.6 was specified in the `vmd-1.9.3/configure`[18] file. Also note that the Tcl8.6 package is installed on the nodes. This might be the cause of VMD's inability to load several packages during startup, since the VMD build instructions say that the Tcl version used for all components of VMD should be the same or errors may occur at runtime [182]. Errors regarding missing packages do appear at runtime, but do not appear to affect VMD's ability to connect to a running NAMD simulation.

The file `plugins/Make-arch`[19] was modified to edit the compiler flags and options (and include the netcdf library, but this might not be necessary [184]) VMD was compiled with the command `make LINUXCARMA TCLINC=-I/home2/ben/vmd-buildProductionCUDA_noMPI/lib/tcl/include TCLLIB=-F/home2/ben/vmd-buildProductionCUDA_noMPI/lib/tcl/`.

Note: do not use the `-j` option with `make` here, this caused a build failure. The final destination of the plugins was set using the `PLUGINDIR` environment variable. In this case, the command `export PLUGINDIR=/home2/ben/vmd-buildProductionCUDA_noMPI/vmd-1.9.3/plugins` was used and checked using `echo $PLUGINDIR`. Then the command `make distrib` was used to move the compiled plugins to the vmd-1.9.3 directory.

[18] files/vmdFiles/configure
[19] files/vmdFiles/Make-arch

In the `vmd-1.9.3/Makefile`,[20] the `linux.carma.cuda.opengl` target was created, based on the other targets but with several features removed (they gave errors during the build process are were deemed unnecessary), such as VRPN (the Virutal Realiity Peripheral Network) [185], ACTC (faster triangle consolidation) [186] [187], and LIBSBALL (spaceball input device, see the configure script associated with [182]).

The file `vmd-1.9.3/configure`[21] was edited to change the `install_name` and `install_dir` values, Tcl and tk library locations and libraries to include, CUDA settings, and compiler options. The CUDA settings were copied from the `LINUX` target to the `LINUXCARMA` target.

VMD was compiled with

```
make linux.carma.cuda.opengl
cd src
make veryclean
make -j4
make install
```

The command `make install` will install VMD in the directory specified by the config file. This directory must be in the PATH variable, which is why `/home2/bin/bin` is in `.bashrc`.[22] The VMD startup script is located in `/home2/bin/bin` and the rest of the VMD files in `/home2/bin/vmd`. VMD can be run by typing `vmd-cuda` in a terminal because that is the specified install name. The entry `/usr/local/cuda-8.0/targets/aarch64-linux/lib/` was placed in `LD_LIBRARY_PATH` in `~/.bashrc`[23] so that VMD could find the required CUDA libraries.

A build of the OptiX Ray Tracing Engine [188] was attempted, but it is unfortunately not available on the ARM platform [189].

4.7.3.12.2 Running an IMD Simulation with VMD

To run an IMD (Interactive Molecular Dynamics) simulation with VMD and NAMD, do the following:

- Launch VMD by typing `vmd-cuda` in a terminal. Output will appear about packages not being loaded, this will not affect this simulation, although it is probably not supposed to appear – see section 4.7.3.9.1 (Building VMD).
- Go to `File > New Molecule`. click `browse`, then browse to `/home2/ben/namd-tutorial-files/1-1-build/example-output/ubq_ws.pdb`. Click `OK`, then click `Load`. The molecule should appear in the VMD window.

[20] files/vmdFiles/Makefile
[21] files/vmdFiles/configure
[22] files/common/.bashrc
[23] files/common/.bashrc

- Go to `Graphics > Representations` and set the drawing method to `CPK`. The molecule should become a ball and stick model.
- Go to `Extensions > Simulation > IMD Connect (NAMD)`. Enter `headNode` for the hostname and `2030` for the port. DO NOT click `connect` yet.
- Open a new terminal window and change directory to `/home2/ben/namd-buildProduction/NAMD_2.12_Source/Linux-ARM64-g++` and start NAMD with the following command to run the simulation on all the nodes: `./charmrun ++nodelist /home2/ben/nodelist ++p 30 ./namd2 ~/namd-tutorial-files/1-2-sphere/ubq_ws_eq.conf`
- To run the simulation only on the headNode, use `./charmrun ++local ++p 6 ./namd2 ~/namd-tutorial-files/1-2-sphere/ubq_ws_eq.conf`
- Optional: These commands may be prefaced with the `time` command to measure the total elapsed time.
- Wait until the line about waiting for Interactive simulation appears, then click `connect` in the IMD window. The simulation will run, and the molecules will begin to move.

This IMD tutorial was very helpful: [190]. Other helpful sources, such as resolving build errors or learning how to use VMD: [191] [186] [192] [193] [194] [195] [196] [177] [197] [198].

4.7.3.13 Ansible

Ansible is the configuration management tool used on the cluster. Christopher Wieringa recommended a configuration management tool in personal communication and suggested Ansible would be a good choice. A configuration management tool allows for repeatable system setup processes (called playbooks) to be run with little effort. A playbook can also be re-run to implement a change after its initial run. Ansible is installed on the head node and SSH-es into the worker nodes when run. Christopher Wieringa recommended installing Ansible on the head node in personal communication because it removes any reliance on another computer – otherwise, another computer with Ansible installed would be necessary to manage the cluster. Ansible is agentless [199], meaning it does not need to be installed on any of the worker nodes. It is only installed on the head node. This makes it very easy on system resources.

The Ansible playbooks (programs) are stored in the `ansible` git repository in the `crayowulf` GitHub organization. The system configuration files that Ansible places in their proper locations in the cluster are stored in the `files` repository. These repositories are cloned to `/home/nvidia` on the head node, meaning they are accessed locally at `/home/nvidia/ansible` and `/home/nvidia/files`. Ansible makes it much easier for another party to reproduce this particular software configuration.

4.7.3.13.1 Runing Ansible

Ansible is run from the `nvidia` account (The nvidia and ubuntu accounts are created when you flash the Jetson). Ansible looks for its files in `/home/nvidia/files`, although the playbooks

may be edited to change this. The inventory records used by Ansible are stored in the `/home/nvidia/ansible/inventory` file.

Note: `-K` means to ask for the sudo password. Replace `<playbookname>` with the name of the desired playbook.

To run a playbook on the head node, the `-c local` option must be used, e.g. `ansible-playbook -K -c local -i inventory <playbookname>.yml.`

To run on the worker nodes, use `ansible-playbook -K -i inventory <playbookname>.yml`

The `mpi.yml` and `vmd.yml` playbooks are run with the following command to limit their runs to the worker nodes: `ansible-playbook -K -i inventory <playbookname>.yml --limit "workerNodes"`

`ansible-playbook -K -c local -i inventory <playbookname>.yml --limit "headNode”` is used to run `mpi.yml` or `vmd.yml` on the head node.

`--limit "worker02"` limits the playbook to running only on worker02 instead of the rest of the nodes. Replace `worker02` with the hostname of any node.

4.7.3.13.2 Playbook Information

`headNodeSetup.yml` installs packages and places configuration files on the headNode for many of the features previously discussed. `workerNodeSetup.yml` does the same for the worker nodes.

`mpi.yml` installs the custom compiled OpenMPI packages and can be run on either the head or the worker nodes. The packages reside in `/home2/shared_install_files/openmpi`.

`vmd.yml` installs packages necessary for building VMD.

`arduinoWorkers.yml` installs packages necessary for Arduino development. This includes the Arduino IDE, programs for gathering system information such as temperature (to be fed to the program controlling the LEDs) and packages necessary for building the kernel module to connect to the Arduino. See the Arduino section for an explanation.

The following sources were useful in configuring Ansible and writing playbooks: [200] [199] [201] [202] [203] [204] [205] [206] [207] [208] [209] [210] [211] [212] [213] [214] [215] [216] [217] Ansible would sometimes crash on the “Gathering facts” stage. Because none of the information captured in that stage was used, the `gather_facts` option was set to “no” in all

playbooks [218] (apparently “False” works too [219], although this has not been verified on the cluster).

4.7.3.14 Performance Tuning

NVIDIA provides a script called `jetson_clocks.sh` in the `/home/nvidia` directory on each Jetson. According to its usage information, when run without arguments, it sets the frequencies of the CPU, GPU and EMC to their maximum. A line executing this file was added to `rc.local`[24] on each node because it was found that the settings did not persist across a reboot. Due to time restrictions, not much performance turning was accomplished.

4.7.3.15 Other System Configuration Notes

The wireless network adapters were disabled using assistance from this forum thread: [220].

Miscellaneous helpful sources: [221] [222] [223] [224] [225] [226] [227] [228] [229] [230] [231] [232] [233] [234] [235].

[24] files/head/rc.local & files/worker/rc.local

5 Integration and Testing

5.1 System Benchmarks

To run HPL (CPU-only), change directory to `/home2/ben/hpl-2.2V2/bin/aarch64` and run the command `mpirun xhpl`. A score of 10.58 Gflops was obtained with the CPU-only HPL benchmark. The `HPL.dat` file used is available in the files repository.[25] This is less than the score obtained by Microwulf (over 26 Gflops) [30], which is a much lower score than the team hoped for. That said, Christopher Wieringa informed the team in personal communication that HPL scores are highly dependent on the implementation of blas in use. The version of blas in use, called `openblas`, is a generic version which means that it may not be optimized for the ARM architecture.

5.2 Cryptography Implementation Testing

Adding the encryption scheme into the command shell involved running the encryption setup while the server and client are setting up. From there the RC5 cipher needed to be layered on all communication between the computers. Testing for functionality involved running the command shell and checking that the correct instructions are run and the correct output it returned.

A list directory command was sent ciphered to the server. The server ran the correct command and returned the correct text back to the client. The specific times of these operations were not found but take less time than is noticeable, which was the requirement of the system.

5.3 Water Cooling Loop Testing

5.3.1 Cooling Loop Leakage Testing

To test the water cooling system, the water blocks, pump, and reservoir were linked together with tubing and the reservoir was filled with water. The pump was turned on and water flowed through the system along with entrapped air. After initially running the system, any leaks that occurred were patched. The system was run for over 48 hours to ensure no more leaks occurred. However, air stayed in the system due to the position of the reservoir entry flow since the reservoir was not completely filled with water. Thus, the position of entry was changed from the top of the reservoir to where it is now, see **Figure 47**. With this adjustment, since the entry flow was below the water level, no air bubbles were introduced to the system, and any air initially trapped in the components was released after several hours. Additional water testing occurred once the cooling system was installed in the case before electronic components were installed. The cooling system was confirmed to have no leaks before implementing electronic components.

[25] files/common/HPL.dat

Once the system was completely constructed, there were multiple final tests run. The cluster was benchmarked so that it can be compared to other supercomputers. An analysis of the effectiveness of the cooling system was performed to ensure that the performance requirements were met. The cooling system meets the 60°C maximum CPU/GPU die specification.

5.3.2 Theoretical Thermal Analysis

To provide an estimate for the temperatures throughout the water in the cooling loop as well as the Nvidia Jetson processors, a theoretical thermal analysis was performed. The thermal analysis assumed various parameters, all located in **Appendix F**. The assumption that all the heat generated by the Nvidia Jetsons was transferred completely to the water in the water blocks was made. It was also assumed that this heat transfer rate, 75 W total, was dissipated by the radiator at a steady rate. The radiator effectiveness was assumed to be 40%. The heat transfer between the Nvidia Jetson processors and the water flowing through the system required the calculation of various thermal resistances. Conductive resistances through the aluminum Nvidia Jetson plate and water block were calculated based on nominal values. The thermal resistance of the water was found by estimating the convective heat transfer coefficient for water moving through a rectangular cross-section. Heat transfer within the water was analyzed using mass flow rate of the water, specific heat capacity, the heat generation assumption, and one temperature from either the inlet or outlet of the water block. The heat transfer analysis of the radiator required the NTU-Effectiveness method and was used to find the temperature of the exit stream from the fifth water block. By finding this temperature, the other temperatures of the system were found using the methods described.

The resulting temperatures for both water and processing units are shown in **Figure 60**. Here, the estimated Nvidia Jetson processing temperatures differ from experimental values, though they are not off by a large magnitude. Differences in results were based on the assumptions made in the theoretical thermal analysis and how these assumptions differed from reality. For instance, the Nvidia Jetson GPUs were hotter than the CPUs in the experimental results, but the theoretical model did not account for this. The temperature variations may have resulted from differing power draws of each processing unit. Also, the processing units were not evenly spread across the Nvidia Jetson plate. Additional factors could have played a role in the temperature differences as well. Nonetheless, the theoretical model was useful in showing that only a slight temperature gradient occurred in the water stream between the first and fifth water blocks.

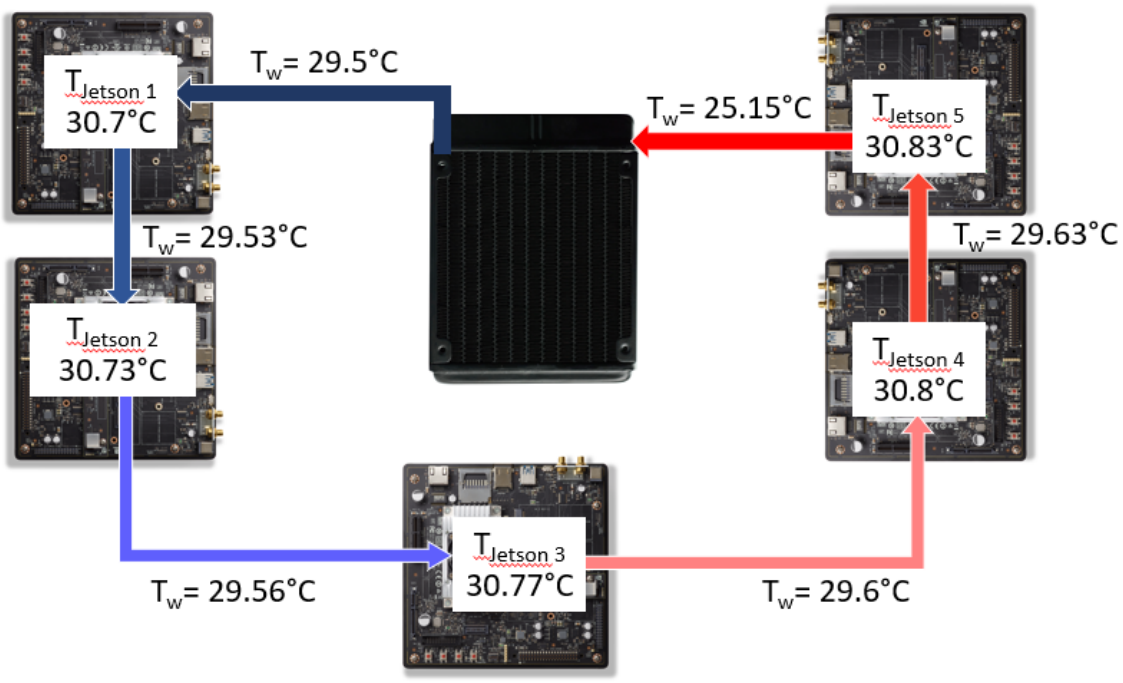


Figure 60. Theoretical Steady State Temperatures

5.3.3 Experimental Thermal Analysis

Once the cooling loop was installed into the system, thermal tests could be performed. The first thermal testing was performed with the stock air coolers on the Nvidia. The case was closed, and all fans were set to their highest setting. The clock rates of the CPUs on all the Jetsons were set to 100%, and the high-performance linpack (HPL) benchmark was used to utilize the CPUs. The GPUs were also being utilized, but not at 100%. The temperature of the testing room was 25°C. Below, **Figure** 61 shows the average CPU and GPU temperatures during the stock cooler test.

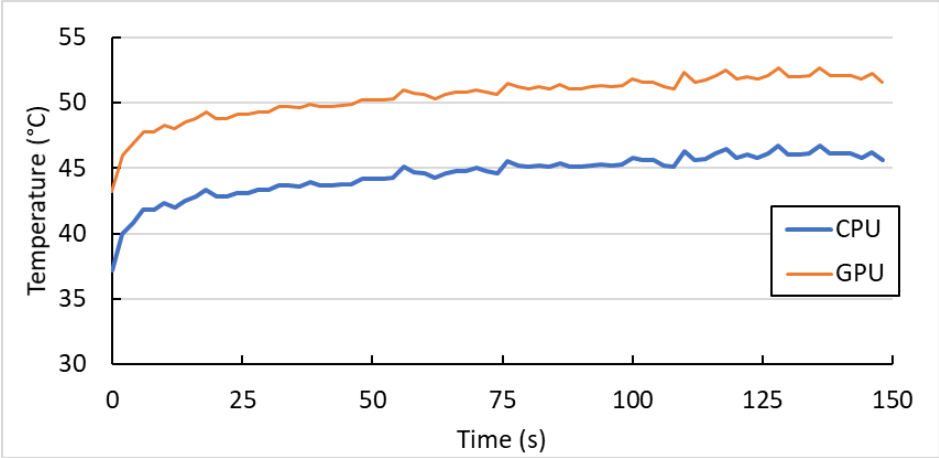


Figure 61**.** Stock Cooler Thermal Test

The temperatures of the processing units ramped up quickly, but then reached an average stable temperature of 45.9°C for the CPU and 51.9°C for the GPU.

The water blocks were installed onto the Jetsons and the same thermal tests were performed. Below, **Figure 62** shows the average CPU and GPU temperatures during the water cooling test.

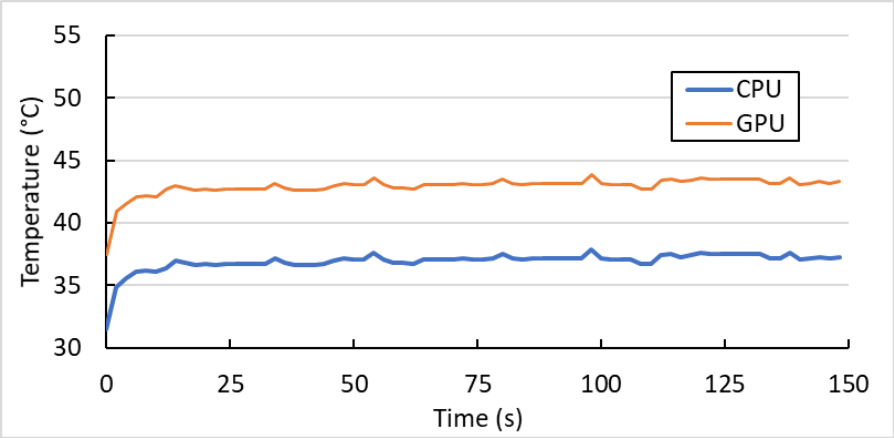


Figure 62**.** Water Block Thermal Test

The temperatures of the processing units ramped up quickly, but then reached an average stable temperature of 37.3°C for the CPU and 43.3°C for the GPU. When considering the ambient temperature of the room, the water cooling system yields a 40% decrease in CPU temperatures coming from air cooling, and a 30% decrease in GPU temperatures. Note from the figures that a stable temperature was reached much quicker when using the water blocks than when using the stock coolers. Also note that the idle temperatures for the CPUs and GPUs were approximately 9°C cooler when using the water blocks than when using the stock coolers.

5.4 Electrical Testing

Five boards were constructed instead of six, as one of the extra MOSFETs was lost to the floor. All components were successfully soldered to the board and tested on a Jetson. The boards did not function as intended. When connected to a voltage source the high output expected was only a few millivolts, much lower than needed. Examining the circuit found that the model provided for the MOSFET had an incorrect pinout. So, the circuits would not work for providing a power-on signal when active.

As an alternative, two contacts on the Jetson TX2 were connected together. The difficulty in this plan came from the two points being bare contacts on either side of a resistor a few thousands of an inch long. This, combined with the nearby components, made it a poor decision to solder these connections together. So instead a tiny wire connecting the contacts was attached to the board. Applying this solution to all boards worked and they are all now booting when power is supplied.

6 Business Plan

6.1 Executive Summary

Our company is a computer case and cooling developer, manufacturer, and distributor. We strive to stay at the front of computer case and cooling technologies. We target customers from the everyday PC user to the PC enthusiast. The custom PC market is thriving as of late and our company will claim a slice of that pie.

6.2 Mission Statement

Our mission is to develop products that are affordable, durable, and cutting-edge. We strive to provide products that we are proud of in order to ensure that our customers receive the highest quality of goods.

6.3 Business Description

Our company develops, manufactures, and sells high quality computer cases and computer cooling solutions. Our customers consist of the everyday consumer, the researcher, and the PC enthusiast. We strive to stay at the front of computer case and cooling technologies.

6.4 SWOT Analysis

Strengths:

- Designing cases for multi-computer builds
- Designing liquid cooling devices
- Having been computer enthusiasts for years

Weaknesses:

- Current computer case and cooling companies have years of experience and have established certain methods/techniques for designing and testing products. Our company lacks experience and capital, which are both necessary assets.

Opportunities:

- Our company can research new computer case and liquid cooling technologies for use with computer hardware approaching deployment in the near future. As a new company, we can specialize in catering products for this future hardware.

Threats:

- There are many competitors on the market who produce either computer cases, liquid cooling systems, or both. Since we are a small company that focuses on both cases and cooling, it makes it more difficult to lead the industry since we must divide our resources.

6.5 Industry Environment/Background

The computer case and cooling industry has exploded in the past 10 years with an increase in compatibility of components. This increase in compatibility has made it easier for consumers to build their own computers. While computer components such as CPUs, GPUs, and memory degrade in value by vast amounts over time, computer cases and cooling solutions retain their value for longer. The demands for new innovative computer cases and cooling solutions will increase as the unslakable thirst of modern-day computer enthusiasts becomes even greater.

6.6 Competitor Analysis

Competitors consist of companies such as Corsair, Antec, EK, Noctua, Fractal Design, and many others. There are a large variety of case producers, and many cooling companies that have products on the market right now. Some have been in the case and cooling markets for years. Our competitors have capital already; we do not. However, our company allocates a large amount of resources to researching and developing our technologies. Also, we do not have any company owners above us, so we can explore methods of creating as we see fit rather than needing to limit our creativity to the bounds of others.

6.7 Market Analysis/Plan

We will market our products through internet advertisement, as those who are viewing the internet are using a computer. Review samples of cases and cooling solutions will be sent to social media figures, such as Linus Sebastian from Linus Tech Tips, who will make our products known to the public. Our product line will accommodate the everyday consumer, the Ph.D. researcher, and the PC enthusiast.

7 Future Work

Improvements that could be made to the cluster and shortcomings that could be rectified include:

- It is currently impossible to log into a NIS-managed account on the worker nodes. By switching to another virtual terminal (e.g. pressing Ctrl+Alt+F6) one can see an error message about the head node not responding. Research indicates that this error is NIS related [132], but the issue was deemed low priority as it did not affect SSH or RSH logins. The `ubuntu` account on all the nodes via the GUI can still be accessed.
- It would be best to have created a partition on the SSD (shared via NFS) that is separate from the home directories. It could be named `/share`. This would look much more professional than just storing shared executable files in `/home2/bin` or storing shared installer files in `/home2/shared_install_files`. Note that there is no user named `bin` or `shared_install_files`, those are ordinary directories.
- Giving more descriptive names to the GitHub repositories would look more professional (`files` is rather vague).
- Modifying CUDA-enabled HPL to run on the ARM platform.
- Building and troubleshooting a CUDA-aware build of NAMD. This is theoretically possible as a build was obtained, but errors occurred at runtime. NAMD currently runs on the CPUs only.
- Write a script that shuts down all the worker nodes when run. Currently, each worker node must shut down by logging in via SSH or RSH and executing `sudo shutdown -h now`.
- Finish enabling the bonding module on all the nodes and implement channel bonding.
- Ensure Jumbo frames are properly enabled and are being used by benchmarks and other computations.
- Move the NAMD binary to a more permanent and available location. The notes.txt file accompanying NAMD [175] states that the `make release` command will aid in this process.

Water cooling improvements could also be made. The current water cooling loop allows the system to be used in a teaching environment because of its eye-catching nature and its technical implications. Temperature probes could be placed in the future along the length of the piping, and students could see how much heat is being dispersed by the water block.

8 Conclusion

The case was created completely in-house using mostly aluminum sheet metal, though it also included 3D printed and steel parts. The case was modeled after the Cray-1 supercomputer. It incorporated a mechanism for easily accessing the internals of the computer and liquid cooling systems. Though tolerancing issues arose from cutting and welding, these obstacles were overcome and the case was completed. The liquid cooling system was created using a pump, drainage valve, radiator, fans, reservoir, and water cooling blocks, which were placed on the Nvidia Jetsons. The water blocks and reservoir were custom built. The water blocks were made using aluminum, rubber, and polycarbonate, while the reservoir was constructed using polycarbonate and plastic adhesives. The water cooling loop outperformed the air cooling devices which came on the Nvidia Jetson TX2 developer kits. LEDs were integrated into the system to display information for CPU and GPU temperatures, as well as to use different lighting schemes. Overall, the system performed beyond expectations.

Through this design process, it was found that it is possible to successfully construct a Beowulf cluster using NVIDA Jetsons, although ARM support for various scientific applications may be limited. Anyone reproducing this project is advised to consider their use case carefully and verify that any software necessary will function properly on ARM. This cluster should provide a valuable teaching and recruitment tool for the Computer Science department.

9 Resources and Acknowledgements

Professor Michmerhuizen was our engineering senior design advisor.

Over the course of the semester, there have been individuals and entities that have been extremely helpful and will continue to be helpful in this project.

NVIDIA Corporation donated one of the Jetsons and allowed the team to purchase the others at the educational price.

Jonathan Bentz from NVIDIA Corporation helped explain why the CUDA-enabled version of the HPL benchmark would not run.

Phil Jasperse is the head of the Calvin College engineering machine shop and has so far been helpful in training the members of the team in metal working. He is extremely friendly and will indubitably be a valuable asset during the fabrication of the system.

Eric Walstra is the industrial consultant for the project. He has met with the team and given them helpful advice and direction. A future meeting with him is already in the works.

Professor Joel Adams came up with the original idea for this project and he has been very involved in its development. The team has been meeting with him on a weekly basis for the sake of keeping on track. He will continue to meet with the team for the duration of the project.

Chris Wieringa and Chuck Holwerda have aided the team in sourcing the components of the system such as wiring and the Nvidia Jetsons themselves. Chris has also been a valuable source of input on the code being written and on system configuration.

Microsoft has written the fastest, freely-available implementation of supersingular isogeny cryptography. The low-level operations are written at the lowest level possible, so they will be sped up using custom instructions implemented by the Jetson TX2.

An Nvidia Developer forum was used to find CAD files for the Nvidia Jetson TX2 developer kit. These files were used for dimensioning the Nvidia Jetson for other CAD work. The site may continue to provide necessary information as the project progresses.

10 References/Bibliography

[1] R. G. Brown, "What's a Beowulf?," 24 May 2004. [Online]. Available: http://webhome.phy.duke.edu/~rgb/Beowulf/beowulf_book/beowulf_book/node9.html. [Accessed 10 May 2018].

[2] C. P. L. a. M. N. Costello, ""Efficient algorithms for supersingular isogeny Diffie-Hellman."," 2016.

[3] "Tegra Block Diagram," [Online]. Available: https://devblogs.nvidia.com/parallelforall/wp-content/uploads/2017/03/Tegra_Parker_Block_Diagram-1.png.

[4] "ARM® Compiler armasm User Guide: 12.68 VMULL," ARM, 2016. [Online]. Available: infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.dui0473m/dom1361289967170.html.. [Accessed 10 December 2017].

[5] ""NVIDIA Jetson TX2 System-on-Module Pascal GPU + ARMv8 + 8GB LPDDR4 + 32GB eMMC + WLAN/BT."," NVIDIA Corporation, 2017. [Online]. Available: developer2.download.nvidia.com /assets/embedded/downloads/secure/tx2/Jetson%20TX2%20Module%20Data%20Sheet/Jetson_TX2_Module_DataSheet_v.

[6] Data Pro, "DataPro," n.d.. [Online]. Available: https://www.datapro.net/products/hdmi-panel-mount-coupler-f-f.html. [Accessed 10 May 2018].

[7] Amazon, "Amazon," 11 December 2014. [Online]. Available: https://www.amazon.com/UGREEN-Ethernet-Extension-Shielded-Connector/dp/B00QV0K2DI/ref=sr_1_4?s=electronics&ie=UTF8&qid=1525973549&sr=1-4&keywords=ethernet%2Bcable%2Bextender&th=1. [Accessed 10 May 2018].

[8] Aerospace Specification Metals Inc., "ASM Aerospace Specification Metals Inc.," n.d.. [Online]. Available: http://asm.matweb.com/search/SpecificMaterial.asp?bassnum=ma6061t6. [Accessed 10 May 2018].

[9] Gaming Factors, "Best Thermal Paste 2018 - Tested by Experts," 2018. [Online]. Available: https://www.gamingfactors.com/best-thermal-paste/. [Accessed 10 May 2018].

[10] uni.no. [Online]. Available: https://www.uio.no/studier/emner/matnat/math/MEK4450/h11/undervisningsmateriale/modul-5/Pipeflow_intro.pdf. [Accessed 10 May 2018].

[11] R. S. Subramanian, "Heat transfer in Flow Through Conduits," n.d.. [Online]. Available: http://web2.clarkson.edu/projects/subramanian/ch330/notes/Heat%20Transfer%20in%20Flow%20Through%20Conduits.pdf. [Accessed 10 May 2018].

[12] Y. Cengel, J. Cimbala and R. Turner, Fundamentals of Thermal-Fluid Sciences, New York: McGraw-Hill Companies, Inc., 2008.

[13] R. K. a. D. P. S. Shah, Fundamentals of Heat Exchanger Design, J. Wiley, 2003.

[14] A. M. Helmenstine, "The Difference Between Distilled and Deionized Water," 12 January

2018. [Online]. Available: https://www.thoughtco.com/distilled-versus-deionized-water-609435. [Accessed 10 May 2018].

[15] J. Adams, "Microwulf Computer Cluster," [Online]. Available: https://www.calvin.edu/~adams/research/microwulf/.

[16] "Nvidia Jetson CAD Files," [Online]. Available: https://devtalk.nvidia.com/default/topic/998129/jetson-tx2/links-to-jetson-tx2-resources/.

[17] "JETSON TX2 DEVELOPER KIT UK Store Page," [Online]. Available: https://www.nvidia.co.uk/buy/embedded/jetson-tx2-dev-kit/.

[18] Microsoft, "SIDH v2.0 (C Edition)," 21 April 2017. [Online]. Available: https://github.com/dconnolly/msr-sidh.

[19] "Nvidia Jetson Additional File Downloads," [Online]. Available: https://developer.nvidia.com/embedded/downloads.

[20] "Arduino Uno Store Page," [Online]. Available: https://store.arduino.cc/usa/arduino-uno-rev3.

[21] "LED Store Page," [Online]. Available: https://www.pololu.com/product/2550.

[22] NVIDIA Corporation, "NVIDIA JETSON," NVIDIA Corporation, n.d.. [Online]. Available: https://www.nvidia.com/en-us/autonomous-machines/embedded-systems-dev-kits-modules/. [Accessed 09 May 2018].

[23] NVIDIA Corporation, "JETSON TK1," NVIDIA Corporation, n.d.. [Online]. Available: http://www.nvidia.com/object/jetson-tk1-embedded-dev-kit.html. [Accessed 09 May 2018].

[24] fchk, JerryChang and linuxdev, "Booting TX1 via network - I don't get it.," 15 August 2017. [Online]. Available: https://devtalk.nvidia.com/default/topic/1022628/booting-tx1-via-network-i-don-t-get-it-/. [Accessed 09 May 2018].

[25] teg5926 and kayccc, "Ethernet Interface," 29 November 2016. [Online]. Available: https://devtalk.nvidia.com/default/topic/979056/?comment=5029282. [Accessed 09 May 2018].

[26] anon., "Jetson/TX2 Cloning," 20 April 2017. [Online]. Available: https://elinux.org/index.php?title=Jetson/TX2_Cloning&oldid=440946. [Accessed 09 May 2018].

[27] NVIDIA Corporation, "NVIDIA Tegra Linux Driver Package Development Guide," 13 July 2017. [Online]. Available: https://developer.nvidia.com/embedded/dlc/l4t-documentation-28-1. [Accessed 07 May 2018].

[28] kangalow, "Run Jetson TX1 from SD Card," 26 January 2017. [Online]. Available: http://www.jetsonhacks.com/2017/01/26/run-jetson-tx1-sd-card/. [Accessed 09 May 2018].

[29] NVIDIA Corporation, "Jetson TX2-TX2i Module Data Sheet," 08 March 2018. [Online]. Available: http://developer.nvidia.com/embedded/dlc/jetson-tx2-tx2i-module-data-sheet. [Accessed 28 April 2018].

[30] J. Adams, "Microwulf: A Personal, Portable Beowulf Cluster," Calvin College Computer Science Department, 01 August 2007. [Online]. Available: http://www.calvin.edu/~adams/research/microwulf/. [Accessed 09 May 2018].

[31] Advanced Micro Devices, "AMD NPT Family 0Fh Desktop Processor Power and Thermal Data Sheet," February 2007. [Online]. Available: https://support.amd.com/TechDocs/33954.pdf#search=AMD%20Athlon%2064%20X2%203800%2B%20. [Accessed 09 May 2018].

[32] G. Volkema and G. Khanna, "Scientific Computing Using Consumer Video- Gaming Hardware Devices," Center for Scientific Computation & Visualization Research.

[33] linuxdev, "Do I have to flash Jetson TX1 by using JetPack?," 02 March 2016. [Online]. Available: https://devtalk.nvidia.com/default/topic/920710/jetson-tx1/do-i-have-to-flash-jetson-tx1-by-using-jetpack-/post/4823375/#4823375. [Accessed 07 May 2018].

[34] NVIDIA Corporation, "Linux for Tegra R28.1," [Online]. Available: https://developer.nvidia.com/embedded/linux-tegra-r281. [Accessed 28 April 2018].

[35] NVIDIA Corporation, "JetPack 3.1 Release Notes," [Online]. Available: https://developer.nvidia.com/embedded/jetpack-3_1. [Accessed 28 April 2018].

[36] N. Corporation, "Jetson Download Center," NVIDIA, [Online]. Available: https://developer.nvidia.com/embedded/downloads. [Accessed 28 April 2018].

[37] NVIDIA Corporation, "Release Notes," [Online]. Available: https://docs.nvidia.com/jetpack-l4t/#developertools/mobile/jetpack/l4t/3.2/jetpack_l4t_release_notes.htm. [Accessed 28 April 2018].

[38] NVIDIA Corporation, "NVIDIA Jetson TX1/TX2 Developer Kit Carrier Board," 26 September 2017. [Online]. Available: http://developer.nvidia.com/embedded/dlc/jetson-tx1-tx2-developer-kit-carrier-board-spec-20170615. [Accessed 10 May 2018].

[39] C. Schroder, "Boost Reliability with Ethernet Bonding and Linux," 27 August 2007. [Online]. Available: http://www.enterprisenetworkingplanet.com/netsysm/article.php/3696561/Boost-Reliability-with-Ethernet-Bonding-and-Linux.htm. [Accessed 09 May 2018].

[40] anon., "Jetson/Tutorials/Program An Arduino," 09 January 2015. [Online]. Available: https://elinux.org/Jetson/Tutorials/Program_An_Arduino. [Accessed 06 May 2018].

[41] AutoCar, "Can not upload Arduino Nano Program," 14 April 2018. [Online]. Available: https://devtalk.nvidia.com/default/topic/1032264/jetson-tx2/can-not-upload-arduino-nano-program/post/5251715/#5251715. [Accessed 06 May 2018].

[42] kangalow, "Build Kernel and ttyACM Module – NVIDIA Jetson TX2," 31 July 2017. [Online]. Available: http://www.jetsonhacks.com/2017/07/31/build-kernel-ttyacm-module-nvidia-jetson-tx2/. [Accessed 30 April 2018].

[43] T. Brom, "Microwulf Software and Network Configuration Notes," Calvin College, 16 May 2008. [Online]. Available: http://www.calvin.edu/~adams/research/microwulf/sys/microwulf_notes.pdf. [Accessed 26 April 2018].

[44] "Dynamic Host Configuration Protocol (DHCP)," [Online]. Available: https://help.ubuntu.com/lts/serverguide/dhcp.html. [Accessed 27 April 2018].

[45] "isc-dhcp-server," 14 April 2015. [Online]. Available: https://help.ubuntu.com/community/isc-dhcp-server. [Accessed 27 April 2018].

[46] deadlydog, "Why does Git treat this text file as a binary file?," 16 January 2014. [Online]. Available: https://stackoverflow.com/a/21170043. [Accessed 27 April 2018].

[47] Canonical Ltd, "dhcp-options - Dynamic Host Configuration Protocol options," Canonical Ltd, 2010. [Online]. Available: http://manpages.ubuntu.com/manpages/trusty/man5/dhcp-options.5.html. [Accessed 27 April 2018].

[48] Apurv, "check status of services and daemons," 16 November 2014. [Online]. Available: https://askubuntu.com/a/551272. [Accessed 27 April 2018].

[49] blade4, "Re: How to View The Services Running?," 20 August 2011. [Online]. Available: https://ubuntuforums.org/showthread.php?t=1829276&p=11170336#post11170336. [Accessed 27 April 2018].

[50] win32sux, "ping through a specific interface," 11 Februrary 2006. [Online]. Available: https://www.linuxquestions.org/questions/linux-networking-3/ping-through-a-specific-interface-414246/#post2098734. [Accessed 27 April 2018].

[51] A. Froehlich, "Understanding DHCP Fundamentals," 30 August 2016. [Online]. Available: https://www.networkcomputing.com/unified-communications/understanding-dhcp-fundamentals/1073432460. [Accessed 27 April 2018].

[52] itsbruce, "Move iptables rule (w/o removing and adding)," 24 July 2014. [Online]. Available: https://unix.stackexchange.com/a/146368. [Accessed 27 April 2018].

[53] anon., "IP Addressing," [Online]. Available: https://help.ubuntu.com/lts/serverguide/network-configuration.html.en#ip-addressing. [Accessed 28 April 2018].

[54] anon., "traceroute(8) - Linux man page," n.d.. [Online]. Available: https://linux.die.net/man/8/traceroute. [Accessed 30 April 2018].

[55] "IP Masquerading," [Online]. Available: https://help.ubuntu.com/lts/serverguide/firewall.html#ip-masquerading. [Accessed 26 April 2018].

[56] Celada, "What is the right iptables rule to allow apt-get to download programs?," 29 September 2012. [Online]. Available: https://serverfault.com/a/433304 . [Accessed 26 April 2018].

[57] "IptablesHowTo," 10 September 2017. [Online]. Available: https://help.ubuntu.com/community/IptablesHowTo. [Accessed 28 April 2018].

[58] leucos, "How to remove this iptables rule?," 13 January 2015. [Online]. Available: https://serverfault.com/a/659142. [Accessed 28 April 2018].

[59] "CLI (command line interface)," 24 December 2017. [Online]. Available: https://fedoraproject.org/wiki/How_to_edit_iptables_rules#CLI_.28command_line_interface.29. [Accessed 28 April 2018].

[60] "NetworkManager2016," 29 September 2016. [Online]. Available: https://help.ubuntu.com/community/NetworkManager. [Accessed 28 April 2018].

[61] vurp0, "What is the difference between “systemctl start” and “systemctl enable”?," 14 February 2016. [Online]. Available: https://askubuntu.com/a/733510. [Accessed 28 April 2018].

[62] F. Marier, "Using iptables with NetworkManager," 6 April 2018. [Online]. Available: https://feeding.cloud.geek.nz/posts/using-iptables-with-network-manager/. [Accessed 28 April 2018].

[63] jdthood, "How can I completely remove NetworkManager?," 31 January 2013. [Online]. Available: https://askubuntu.com/a/249972. [Accessed 28 April 2018].

[64] Computer Hope, "Linux ifup, ifdown, and ifquery command," 29 December 2017. [Online]. Available: https://www.computerhope.com/unix/ifup.htm. [Accessed 28 April 2018].

[65] R. User, "Is it enough adding iptables rules without restart?," 13 March 2011. [Online]. Available: https://serverfault.com/a/246942. [Accessed 28 April 2018].

[66] Jonnyuser, "Is it enough adding iptables rules without restart?," 13 March 2011. [Online]. Available: https://serverfault.com/a/246842. [Accessed 28 April 2018].

[67] HermanAB, "Re: how to reset iptables .," 15 January 2010. [Online]. Available: https://ubuntuforums.org/showthread.php?t=1381516&p=8668050#post8668050. [Accessed 28 April 2018].

[68] L. Noodén, "-F -X and --policy," 15 January 2010. [Online]. Available: https://ubuntuforums.org/showthread.php?t=1381516&p=8668856#post8668856. [Accessed 28 April 2018].

[69] "Solution #2 /etc/network/if-pre-up.d and ../if-post-down.d," 10 September 2017. [Online]. Available: https://help.ubuntu.com/community/IptablesHowTo#Solution_.232_.2Fetc.2Fnetwork.2Fif-pre-up.d_and_...2Fif-post-down.d. [Accessed 28 April 2018].

[70] "Solution #3 iptables-persistent," 10 September 2017. [Online]. Available: https://help.ubuntu.com/community/IptablesHowTo#Solution_.233_iptables-persistent. [Accessed 28 April 2018].

[71] C. A, "How to install a cluster with NIS and NFS in Ubuntu 16.04," 15 May 2017. [Online]. Available: https://ilearnedhowto.wordpress.com/2017/05/15/how-to-install-a-cluster-with-nis-and-nfs-in-ubuntu-16-04/. [Accessed 27 April 2018].

[72] "Dnsmasq," 15 December 2011. [Online]. Available: https://help.ubuntu.com/community/Dnsmasq. [Accessed 28 April 2018].

[73] Canonical Ltd, "dnsmasq - A lightweight DHCP and caching DNS server.," [Online]. Available: http://manpages.ubuntu.com/manpages/artful/man8/dnsmasq.8.html. [Accessed 28 April 2018].

[74] L. Madsen, "Assign unique hostname to dhcp client with dnsmasq," 23 July 2012. [Online]. Available: https://leifmadsen.wordpress.com/2012/07/23/assign-unique-hostname-to-dhcp-client-with-dnsmasq/. [Accessed 28 April 2018].

[75] "Local Caching using NetworkManager," 21 January 2016. [Online]. Available: https://wiki.debian.org/HowTo/dnsmasq#Local_Caching_using_NetworkManager. [Accessed 28 April 2018].

[76] Oli, "How do I install dig?," 7 February 2011. [Online]. Available: https://askubuntu.com/a/25100. [Accessed 28 April 2018].

[77] Meilin, "How to Flush/Clean DNS cache in Ubuntu 12.04 Precise," 201-. [Online].

Available: http://ubuntuguide.net/flush-clean-dns-cache-ubuntu-12-04-precise. [Accessed 28 April 2018].

[78] Canonical Ltd, "hosts - static table lookup for hostnames," 2010?. [Online]. Available: http://manpages.ubuntu.com/manpages/trusty/man5/hosts.5.html. [Accessed 28 April 2018].

[79] jdthood, "How do I fix DNS resolving which doesn't work after upgrading to Ubuntu 13.10 (Saucy)," 31 October 2013. [Online]. Available: https://askubuntu.com/a/368935. [Accessed 28 April 2018].

[80] anon., "Building a Nvidia Jetson TK1 Cluster," 17 August 2017. [Online]. Available: http://selkie-macalester.org/csinparallel/modules/RosieCluster/build/html/. [Accessed 08 May 2018].

[81] anon., "Using the Command Line (unattended)," 13 June 2016. [Online]. Available: https://help.ubuntu.com/community/UbuntuTime#Using_the_Command_Line_.28unattended.29. [Accessed 28 April 2018].

[82] J. Heller, "Ansible Switch Basics - Time Zone and NTP," ca. 2016. [Online]. Available: https://getsatisfaction.cumulusnetworks.com/cumulus/topics/ansible-switch-basics-time-zone-and-ntp. [Accessed 28 April 2018].

[83] AxelK, "6.5. ntpd access restrictions," 16 March 2015. [Online]. Available: http://support.ntp.org/bin/view/Support/AccessRestrictions. [Accessed 28 April 2018].

[84] anon., "timezone - Configure timezone setting," 26 April 2018. [Online]. Available: http://docs.ansible.com/ansible/latest/modules/timezone_module.html. [Accessed 28 April 2018].

[85] J. Hess, "dpkg-reconfigure - reconfigure an already installed package," 08 November 2015. [Online]. Available: http://manpages.ubuntu.com/manpages/xenial/man8/dpkg-reconfigure.8.html. [Accessed 28 April 2018].

[86] G. Newell, "How To Display The Date And Time Using Linux Command Line," 19 April 2018. [Online]. Available: https://www.lifewire.com/display-date-time-using-linux-command-line-4032698. [Accessed 28 April 2018].

[87] anon., "Troubleshooting," 13 July 2016. [Online]. Available: https://help.ubuntu.com/community/UbuntuTime#Troubleshooting. [Accessed 28 April 2018].

[88] anon., "What is NTP Server Stratum?," [Online]. Available: http://www.worldtimesolutions.com/support/ntp/What_is_NTP_server_stratum.html. [Accessed 28 April 2018].

[89] anon., "14.6. Basic NTP configuration," [Online]. Available: https://www.tldp.org/LDP/sag/html/basic-ntp-config.html. [Accessed 28 April 2018].

[90] Indiana University Knowledge Base, "Set up SSH public-key authentication to connect to a remote system," [Online]. Available: https://kb.iu.edu/d/aews#linux. [Accessed 28 April 2018].

[91] G. Rutenberg, "ssh-keygen Tutorial – Generating RSA and DSA keys," 5 October 2007. [Online]. Available: https://www.guyrutenberg.com/2007/10/05/ssh-keygen-tutorial-

generating-rsa-and-dsa-keys/. [Accessed 28 April 2018].

[92] "OpenSSH 7.0 disables ssh-dss keys by default," 13 August 2015. [Online]. Available: https://www.gentoo.org/support/news-items/2015-08-13-openssh-weak-keys.html. [Accessed 28 April 2018].

[93] Iowan, "Re: Location of Known_hosts file?," 8 February 2014. [Online]. Available: https://ubuntuforums.org/showthread.php?t=2204411&p=12923531#post12923531. [Accessed 27 April 2018].

[94] theTuxRacer, "Is it possible to remove a particular host key from SSH's known_hosts file?," 10 January 2011. [Online]. Available: https://askubuntu.com/q/20865. [Accessed 27 April 2018].

[95] Adi, "How long is a 2048-bit RSA key? [closed]," 23 June 2013. [Online]. Available: https://security.stackexchange.com/questions/37907/how-long-is-a-2048-bit-rsa-key#comment60051_37907. [Accessed 27 April 2018].

[96] RainDoctor, "How can I tell how many bits my ssh key is?," 09 May 2010. [Online]. Available: https://superuser.com/a/139311. [Accessed 27 April 2018].

[97] S. Josefsson, "Why I don’t Use 2048 or 4096 RSA Key Sizes," 03 November 2016. [Online]. Available: https://blog.josefsson.org/2016/11/03/why-i-dont-use-2048-or-4096-rsa-key-sizes/. [Accessed 27 April 2018].

[98] G. van Dijk, "Upgrade your SSH keys!," 23 September 2016. [Online]. Available: https://blog.g3rt.nl/upgrade-your-ssh-keys.html. [Accessed 27 April 2018].

[99] Jarrod, "Linux How To: Join Two Files – Append One File To Another," 13 July 2016. [Online]. Available: https://www.rootusers.com/linux-how-to-join-two-files-append-one-file-to-another/. [Accessed 27 April 2018].

[100] JulianTosh, "ssh without password for /home shared over NFS," 14 May 2009. [Online]. Available: https://www.linuxquestions.org/questions/linux-server-73/ssh-without-password-for-home-shared-over-nfs-725864/#post3540385 . [Accessed 28 April 2018].

[101] gms02004, "Correct SSH Directory permissions," 27 February 2017. [Online]. Available: https://linux.uits.uconn.edu/2017/02/27/correct-ssh-directory-permissions/. [Accessed 28 April 2018].

[102] "Troubleshooting," 30 July 2015. [Online]. Available: https://help.ubuntu.com/community/SSH/OpenSSH/Keys#Troubleshooting. [Accessed 28 April 2018].

[103] "chmod Shortcuts," [Online]. Available: http://catcode.com/teachmod/shortcuts.html. [Accessed 28 April 2018].

[104] guntbert, "SSH Permission denied (publickey)," 23 June 2013. [Online]. Available: https://askubuntu.com/questions/311558/ssh-permission-denied-publickey#comment393419_311558. [Accessed 28 April 2018].

[105] B. Voigt, "SSH: The authenticity of host <host> can't be established," 06 May 2012. [Online]. Available: https://superuser.com/a/421084. [Accessed 28 April 2018].

[106] Tgr, "Why am I still getting a password prompt with ssh with public key authentication?," 13 August 2017. [Online]. Available: https://unix.stackexchange.com/a/55481. [Accessed

30 April 2018].

[107] anon., "SSH/OpenSSH/Configuring," 18 November 2017. [Online]. Available: https://help.ubuntu.com/community/SSH/OpenSSH/Configuring. [Accessed 10 May 2018].

[108] Canonical Ltd., "rhosts — allow rlogin, rsh and rcp connections to the local machine without a password.," ca. 2016. [Online]. Available: http://manpages.ubuntu.com/manpages/xenial/en/man5/rhosts.5.html. [Accessed 28 April 2018].

[109] IBM, "Allowing users to run the remote shell program (rsh) without a password (AIX® Linux)," [Online]. Available: https://www.ibm.com/support/knowledgecenter/SSZJPZ_11.7.0/com.ibm.swg.im.iis.productization.iisinfsv.install.doc/topics/wsisinst_config_pe_rsh_no_pw.html. [Accessed 27 April 2018].

[110] Vishnu, "How to use passwordless rsh?," 11 October 2012. [Online]. Available: https://askubuntu.com/a/198929. [Accessed 17 April 2018].

[111] C. Wierenga, "Remote Access - Linux," [Online]. Available: https://cs.calvin.edu/sysadmin/remoteaccess-linux.php. [Accessed 27 April 2018].

[112] Veritas Technologies LLC, "Enabling rsh for Linux," [Online]. Available: https://sort.veritas.com/public/documents/sfha/5.1sp1/linux/productguides/html/dmp_install/apcs04.htm. [Accessed 27 April 2018].

[113] Mind Chasers Inc, "Configure Netfilter / iptables on Linux to block unwanted network traffic," 18 October 2012. [Online]. Available: https://www.mindchasers.com/dev/netfilter. [Accessed 28 April 2018].

[114] Oli, "Does “old” rsh exist in Ubuntu?," 28 August 2013. [Online]. Available: https://askubuntu.com/a/338319. [Accessed 28 April 2018].

[115] "rsh(1) - Linux man page," 15 August 199. [Online]. Available: https://linux.die.net/man/1/rsh. [Accessed 28 April 2018].

[116] anon., "Securing OpenSSH," 18 November 2017. [Online]. Available: https://wiki.centos.org/HowTos/Network/SecuringSSH#head-9c01429983dccbf74ade8674815980dc6434d3ba. [Accessed 10 May 2018].

[117] jlinkels, "[SOLVED] cannot add nis users to groups on client machine," 14 January 2010. [Online]. Available: https://www.linuxquestions.org/questions/linux-networking-3/cannot-add-nis-users-to-groups-on-client-machine-782062/#post3826187. [Accessed 28 April 2018].

[118] anon., "SettingUpNISHowTo," 10 September 2017. [Online]. Available: https://help.ubuntu.com/community/SettingUpNISHowTo. [Accessed 28 April 2018].

[119] V. Gite, "UNIX / Linux Command To Check Existing Groups and Users," 6 April 2015. [Online]. Available: https://www.cyberciti.biz/faq/linux-check-existing-groups-users/. [Accessed 28 April 2018].

[120] T. Kukuk, "Homepage of the Linux NIS/NIS+ Projects," 22 January 2012. [Online]. Available: http://www.linux-nis.org/. [Accessed 28 April 2018].

[121] T. Kukuk, "The Linux NIS(YP)/NYS/NIS+ HOWTO," July 2013. [Online]. Available:

http://www.linux-nis.org/nis-howto/HOWTO/. [Accessed 28 April 2018].

[122] anon., "Configuring NIS Services in Linux," 200?. [Online]. Available: http://tutorials.section6.net/home/configuring-nis-services-in-linux. [Accessed 28 April 2018].

[123] anon., "nsswitch.conf(5) - Linux man page," 10 May 2013. [Online]. Available: https://linux.die.net/man/5/nsswitch.conf. [Accessed 27 April 2018].

[124] V. Gite, "Linux List All Users In The System," 2 January 2018. [Online]. Available: https://www.cyberciti.biz/faq/linux-list-users-command/. [Accessed 28 April 2018].

[125] Oracle Corporation and/or its affiliates, "NIS Binding Problems," ca. 2010. [Online]. Available: https://docs.oracle.com/cd/E19683-01/806-4077/6jd6blbe4/index.html. [Accessed 28 April 2018].

[126] jpwigan, "NIS Server -Users not getting /home directory," 16 July 2008. [Online]. Available: https://www.linuxquestions.org/questions/linux-server-73/nis-server-users-not-getting-home-directory-656161/#post3216789. [Accessed 28 April 2018].

[127] M. Anicas, "How To Create a Sudo User on Ubuntu [Quickstart]," 28 March 2016. [Online]. Available: https://www.digitalocean.com/community/tutorials/how-to-create-a-sudo-user-on-ubuntu-quickstart. [Accessed 28 April 2018].

[128] Oracle Corporation and/or its affiliates, "Administering NIS Users," ca.2010. [Online]. Available: https://docs.oracle.com/cd/E19253-01/816-4556/anis2-38485/index.html. [Accessed 28 April 2018].

[129] Oracle Corporation and/or its affiliates, "Password Files and Namespace Security," ca. 2010. [Online]. Available: https://docs.oracle.com/cd/E19253-01/816-4556/anis2-32914/index.html. [Accessed 28 April 2018].

[130] whistl, "slow login usually means dns reverse lookup issue," 07 July 2007. [Online]. Available: https://www.linuxquestions.org/questions/linux-networking-3/ssh-login-do_ypcall-clnt_call-rpc-timed-out-566957/#post2814352. [Accessed 28 April 2018].

[131] anon., "Configure NIS Server," 08 March 2011. [Online]. Available: https://www.server-world.info/en/note?os=Debian_6.0&p=nis. [Accessed 02 May 2018].

[132] redhat, "RHEL mount hangs: nfs: server [...] not responding, still trying," 3 May 2017. [Online]. Available: https://access.redhat.com/solutions/28211. [Accessed 28 April 2018].

[133] M. Singleton, "Sony just announced the world’s fastest SD card," 22 February 2017. [Online]. Available: https://www.theverge.com/circuitbreaker/2017/2/22/14701420/sony-sd-card-worlds-fastest-sfg. [Accessed 09 May 2018].

[134] I. Paul, "Sony reveals the 'world's fastest' SD card," 23 Februrary 2017. [Online]. Available: https://www.pcworld.com/article/3173628/storage/sony-reveals-the-worlds-fastest-sd-card.html. [Accessed 09 May 2018].

[135] Western Digital Corporation, "Difference between SATA I, SATA II and SATA III," n.d.. [Online]. Available: https://kb.sandisk.com/app/answers/detail/a_id/8142/~/difference-between-sata-i%2C-sata-ii-and-sata-iii. [Accessed 09 May 2018].

[136] CORSAIR, "Flash Voyager® GTX USB 3.0 256GB Flash Drive," 201-. [Online]. Available: https://www.corsair.com/us/en/Unformatted-Capacity/flash-voyager-gtx-

config/p/CMFVYGTX3B-256GB. [Accessed 28 April 2018].

[137] Anonymous, "Great idea, but limited driver support may question value," 14 May 2015. [Online]. Available: https://www.newegg.com/Product/Product.aspx?item=N82E16820233767. [Accessed 28 April 2018].

[138] anon., "Network File System (NFS)," 200-. [Online]. Available: https://help.ubuntu.com/stable/serverguide/network-file-system.html. [Accessed 28 April 2018].

[139] anon., "Host Variables," 26 April 2018. [Online]. Available: http://docs.ansible.com/ansible/latest/intro_inventory.html#host-variables. [Accessed 28 April 2018].

[140] Cheesemill, "Re: List all mount points and mounted drives," 05 April 2012. [Online]. Available: https://ubuntuforums.org/showthread.php?t=1952999&p=11819837#post11819837. [Accessed 28 April 2018].

[141] anon., "FilePermissions," 10 November 2013. [Online]. Available: https://help.ubuntu.com/community/FilePermissions. [Accessed 28 April 2018].

[142] S. Moon, "9 commands to check hard disk partitions and disk space on Linux," 16 June 2014. [Online]. Available: https://www.binarytides.com/linux-command-check-disk-partitions/. [Accessed 28 April 2018].

[143] J. P. A. Glaubitz, "nfs mount fails during boot because network is not ready," 18 February 2014. [Online]. Available: https://bugzilla.redhat.com/show_bug.cgi?id=1027114#c8. [Accessed 28 April 2018].

[144] Pierz, "How to see log to find a boot problem," 06 September 2016. [Online]. Available: https://askubuntu.com/a/821801. [Accessed 28 April 2018].

[145] A. Dunn, "Bug 1023737 - fstab method of CIFS/NFS mounts will not mount on startup," 18 August 2014. [Online]. Available: https://bugzilla.redhat.com/show_bug.cgi?id=1023737#c5. [Accessed 02 May 2018].

[146] Indiana University, "What is NFS?," 18 July 2017. [Online]. Available: https://kb.iu.edu/d/adux. [Accessed 09 May 2018].

[147] M. T. Jones, "Network file systems and Linux," 10 November 2010. [Online]. Available: https://www.ibm.com/developerworks/library/l-network-filesystems/index.html. [Accessed 09 May 2018].

[148] Software in the Public Interest, "Open MPI: Open Source High Performance Computing," Software in the Public Interest, 28 Feb 2018. [Online]. Available: https://www.open-mpi.org/. [Accessed 30 April 2018].

[149] anon., "synchronize - A wrapper around rsync to make common tasks in your playbooks quick and eas," 26 April 2018. [Online]. Available: http://docs.ansible.com/ansible/latest/modules/synchronize_module.html. [Accessed 28 April 2018].

[150] anon., "copy - Copies files to remote locations," 26 April 2018. [Online]. Available:

docs.ansible.com/ansible/latest/copy_module.html. [Accessed 28 April 2018].

[151] anon, "Passing Variables On The Command Line," 26 April 2018. [Online]. Available: http://docs.ansible.com/ansible/latest/user_guide/playbooks_variables.html#passing-variables-on-the-command-line. [Accessed 28 April 2018].

[152] GreensterRox, "How can I pass variable to ansible playbook in the command line?," 20 February 2017. [Online]. Available: https://stackoverflow.com/a/42347953. [Accessed 28 April 2018].

[153] V. Gite, "How to use Ansible to autoremove unwanted package dependency with apt," 05 November 2017. [Online]. Available: https://www.cyberciti.biz/faq/how-to-use-ansible-to-autoremove-unwanted-package-dependency-with-apt/. [Accessed 28 April 2018].

[154] anon., "Running Ad Hoc Commands," n.d.. [Online]. Available: https://ansible-tips-and-tricks.readthedocs.io/en/latest/ansible/commands/. [Accessed 28 April 2018].

[155] A. Danshin, "How to install MPI in Ubuntu," 01 March 2012. [Online]. Available: https://jetcracker.wordpress.com/2012/03/01/how-to-install-mpi-in-ubuntu/. [Accessed 30 April 2018].

[156] Bill, "Jetson TK1 MPI Cluster," 15 July 2014. [Online]. Available: https://devtalk.nvidia.com/default/topic/761234/jetson-tk1/jetson-tk1-mpi-cluster/post/4261645/#4261645. [Accessed 30 April 2018].

[157] V. Gite, "How to: Recompiling / Rebuild Debian / Ubuntu Linux Binary Source File Packages," 15 January 2008. [Online]. Available: https://www.cyberciti.biz/faq/rebuilding-ubuntu-debian-linux-binary-package/. [Accessed 30 April 2018].

[158] anon., "PinningHowto," 12 February 2018. [Online]. Available: https://help.ubuntu.com/community/PinningHowto. [Accessed 30 April 2018].

[159] anon., "BuildingTutorial," 23 May 2017. [Online]. Available: https://wiki.debian.org/BuildingTutorial. [Accessed 30 April 2018].

[160] packagecloud, "Working with Debian and Ubuntu Source Packages," 13 April 2015. [Online]. Available: https://blog.packagecloud.io/eng/2015/04/13/working-with-deb-ubuntu-source-packages/. [Accessed 30 April 2018].

[161] November Rain, "How to install Open MPI with Cuda-Aware MPI in Linux Debian," 30 September 2015. [Online]. Available: https://stackoverflow.com/q/32856863. [Accessed 30 April 2018].

[162] F. Mitha, 11 October 2015. [Online]. Available: https://unix.stackexchange.com/a/235471. [Accessed 30 April 2018].

[163] M. Vegter, "building Debian package with non-standard options," 11 October 2015. [Online]. Available: https://unix.stackexchange.com/q/235461. [Accessed 30 April 2018].

[164] aneeshep, "Finding out what package a command came from [duplicate]," 18 November 2010. [Online]. Available: https://askubuntu.com/a/13794. [Accessed 30 April 2018].

[165] Karl, " Installed OpenMPI library, but cannot use mpicc command in Linux Ask Question up vote 0 down vote favorite I am following openmpi install file.," 17 December 2012. [Online]. Available: https://stackoverflow.com/q/13909432. [Accessed 30 April 2018].

[166] vii22, "Is the available global memory in TX2 less than TX1?," 14 June 2017. [Online].

Available: https://devtalk.nvidia.com/default/topic/1013916/jetson-tx2/is-the-available-global-memory-in-tx2-less-than-tx1-/post/5167284/#5167284. [Accessed 30 April 2018].

[167] Mecasickle, "NVCC "No command 'nvcc' found"," 25 January 2010. [Online]. Available: https://devtalk.nvidia.com/default/topic/457664/cuda-programming-and-performance/nvcc-quot-no-command-39-nvcc-39-found-quot-/post/3254803/#3254803. [Accessed 30 April 2018].

[168] Ophidian, "How to add a directory to the PATH?," 22 July 2009. [Online]. Available: https://askubuntu.com/a/60219. [Accessed 30 April 2018].

[169] A. Petitet, R. Whaley, J. Dongarra and A. Cleary, "HPL - A Portable Implementation of the High-Performance Linpack Benchmark for Distributed-Memory Computers," 24 February 2016. [Online]. Available: http://www.netlib.org/benchmark/hpl/. [Accessed 30 April 2018].

[170] cstotts, "HPL on Jetson TK1," 19 January 2016. [Online]. Available: https://devtalk.nvidia.com/default/topic/908451/jetson-tk1/hpl-on-jetson-tk1/post/4785149/#4785149. [Accessed 30 April 2018].

[171] advanced clustering technologies, inc., "How do I tune my HPL.dat file?," advanced clustering technologies, inc., n.d.. [Online]. Available: http://www.advancedclustering.com/act_kb/tune-hpl-dat-file/. [Accessed 03 May 2018].

[172] J. C. Phillips, R. Braun, W. Wang, J. Gumbart, E. Tajkhorshid, E. Villa, C. Chipot, R. D. Skeel, L. Kale and K. Schulten, "Scalable molecular dynamics with NAMD," *Journal of Computational Chemistry,* vol. 26, no. 16, pp. 1781-1802, 26 March 2005.

[173] G. Zheng, L. Kale, E. Bohm, A. Bhatele, C. Mei, F. Gioachin, R. Venkataraman, Y. Sun, B. Acun, S. White and J. Galvez, "Charm++," University of Illinois, 199-. [Online]. Available: http://charm.cs.illinois.edu/research/charm. [Accessed 30 April 2018].

[174] NVIDIA Corporation, "GPU-Accelerated NAMD," n.d.. [Online]. Available: https://www.nvidia.com/en-us/data-center/gpu-accelerated-applications/namd/. [Accessed 02 May 2018].

[175] University of Illinois at Urbana Champaign, "NAMD," University of Illinois at Urbana Champaign, 27 March 2018. [Online]. Available: http://www.ks.uiuc.edu/Research/namd/. [Accessed 08 May 2018].

[176] J. Phillips, "Re: 2.11 how to remove charmrun remote-shell options," 20 December 2016. [Online]. Available: http://www.ks.uiuc.edu/Research/namd/mailing_list/namd-l.2015-2016/2637.html. [Accessed 30 April 2018].

[177] University of Illinois at Urbana-Champaign, "Tutorials," n.d.. [Online]. Available: http://www.ks.uiuc.edu/Training/Tutorials/. [Accessed 05 May 2018].

[178] C. Chen, "/bin/csh: bad interpreter: No such file or directory," 02 April 2011. [Online]. Available: http://chunleichen.blogspot.com/2011/04/bincsh-bad-interpreter-no-such-file-or.html. [Accessed 30 April 2018].

[179] anon., "Package: csh (20110502-2.1ubuntu1) [universe]," ca. 2016. [Online]. Available: https://packages.ubuntu.com/xenial/csh. [Accessed 30 April 2018].

[180] mac.ryan, "How can I get gmake?," 21 March 2007. [Online]. Available: https://ubuntuforums.org/showthread.php?t=389841&p=2332040#post2332040.

[Accessed 30 April 2018].

[181] ssam, "How can I get gmake?," 21 March 2007. [Online]. Available: https://ubuntuforums.org/showthread.php?t=389841&p=2332631#post2332631. [Accessed 30 April 2018].

[182] anon., "VMD," Theoretical and Computational Biophysics Group in the Beckman Institute for Advanced Science and Technology at the University of Illinois at Urbana-Champaign, 27 March 2018. [Online]. Available: http://www.ks.uiuc.edu/Research/vmd/. [Accessed 30 April 2018].

[183] W. Humphrey, A. Dalke and K. Schulten, "VMD -Visual Molecular Dynamics," *Molecular Graphics,* vol. 14, pp. 33-38, 1996.

[184] University of Illinois at Urbana-Champaign, "Compiling plugins from source code," 14 April 2018. [Online]. Available: http://www.ks.uiuc.edu/Research/vmd/plugins/doxygen/compiling.html. [Accessed 08 May 2018].

[185] Virtual Reality Peripheral Network, "Virtual Reality Peripheral Network - Official Repo," 02 January 2017. [Online]. Available: https://github.com/vrpn/vrpn/wiki. [Accessed 30 April 2018].

[186] J. Stone, "Re: tc.h problem," 23 January 2013. [Online]. Available: http://www.ks.uiuc.edu/Research/vmd/mailing_list/vmd-l/21321.html. [Accessed 30 April 2018].

[187] B. Grantham, "ACTC - The Triangle Consolidator," n.d.. [Online]. Available: http://plunk.org/~grantham/public/actc/. [Accessed 30 April 2018].

[188] NVIDIA, "NVIDIA® OptiX™ Ray Tracing Engine," n.d.. [Online]. Available: https://developer.nvidia.com/optix. [Accessed 30 April 2018].

[189] dlacewell, "OptiX on TX1," 15 June 2017. [Online]. Available: https://devtalk.nvidia.com/default/topic/1013309/optix/optix-on-tx1/post/5168434/#5168434. [Accessed 30 April 2018].

[190] anon., "How to run Interactive Molecular Dynamics," 08 March 2006. [Online]. Available: http://www.ks.uiuc.edu/Research/vmd/imd/tutorial/. [Accessed 30 April 2018].

[191] A. Ashtikar, "Symmetric multi-processing, multi-threading and synchronisation explained," 13 October 2016. [Online]. Available: https://techtake.info/2016/10/13/symmetric-multi-processing-multi-threading-and-synchronisation-explained/. [Accessed 30 April 2018].

[192] Pablo, "error C1083: Cannot open include file: 'FL/Fl.h': No such file or directory," 27 October 2014. [Online]. Available: https://stackoverflow.com/q/26579955. [Accessed 30 April 2018].

[193] GUIDO, "Missing tk.h and tcl.h files when building VRip," 10 March 2012. [Online]. Available: https://stackoverflow.com/a/9649478. [Accessed 30 April 2018].

[194] Matthieu, "nvcc fatal : Unsupported gpu architecture 'compute_20," 29 September 2017. [Online]. Available: https://askubuntu.com/a/960553. [Accessed 30 April 2018].

[195] G. Hawkins, "How do I reload .bashrc without logging out and back in?," 25 March 2010. [Online]. Available: https://stackoverflow.com/a/2518150. [Accessed 30 April 2018].

[196] All users responding to thread, "Can I get a report of ALL the libraries linked when building my C++ executable (gcc)? (including statically linked)," 201-. [Online]. Available: https://stackoverflow.com/questions/4925012/can-i-get-a-report-of-all-the-libraries-linked-when-building-my-c-executable. [Accessed 30 April 2018].

[197] anon., "Cortex-R4 and Cortex-R4F Technical Reference Manual," 2009. [Online]. Available: http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0363e/ch12s01s02.html. [Accessed 30 April 2018].

[198] R. Crovella, "CUDA: How to use -arch and -code and SM vs COMPUTE," 26 February 2016. [Online]. Available: https://stackoverflow.com/a/35657430. [Accessed 30 April 2018].

[199] anon., "What is Ansible?," [Online]. Available: https://www.ansible.com/resources/videos/quick-start-video. [Accessed 28 April 2018].

[200] anon., "Hey Wait, A YAML Gotcha," 26 April 2018. [Online]. Available: http://docs.ansible.com/ansible/latest/playbooks_variables.html#hey-wait-a-yaml-gotcha . [Accessed 28 April 2018].

[201] anon., "template - Templates a file out to a remote server," 26 April 2018. [Online]. Available: http://docs.ansible.com/ansible/latest/modules/template_module.html. [Accessed 28 April 2018].

[202] R. Gupta, "SSH Error: Permission denied (publickey,password) in Ansible," 26 October 2015. [Online]. Available: https://stackoverflow.com/a/33339064. [Accessed 28 April 2018].

[203] R. Gupta, "SSH Error: Permission denied (publickey,password) in Ansible," 22 October 2015. [Online]. Available: https://stackoverflow.com/q/33280244. [Accessed 28 April 2018].

[204] T. Moses, "Ansible stuck on gathering facts," 26 July 2017. [Online]. Available: https://serverfault.com/a/865294. [Accessed 28 April 2018].

[205] Computer Hope, "Linux diff command," 24 Jaunary 2018. [Online]. Available: https://www.computerhope.com/unix/udiff.htm. [Accessed 28 April 2018].

[206] anon., "Templating (Jinja2)," 26 April 2018. [Online]. Available: http://docs.ansible.com/ansible/latest/user_guide/playbooks_templating.html. [Accessed 28 April 2018].

[207] anon., "apt - Manages apt-packages," 26 April 2018. [Online]. Available: http://docs.ansible.com/ansible/latest/modules/apt_module.html. [Accessed 28 April 2018].

[208] anon., "Intro to Playbooks," 26 April 2018. [Online]. Available: http://docs.ansible.com/ansible/latest/user_guide/playbooks_intro.html. [Accessed 28 April 2018].

[209] anon., "Latest Releases Via Apt (Ubuntu)," 26 April 2018. [Online]. Available: http://docs.ansible.com/ansible/latest/installation_guide/intro_installation.html#latest-releases-via-apt-ubuntu. [Accessed 28 April 2018].

[210] S. Vokes, "Configuring a Laptop with Ansible, Part Two," 22 September 2015. [Online]. Available: https://spin.atomicobject.com/2015/09/22/ansible-config-example/. [Accessed 28 April 2018].

[211] anon., "Connection Plugins," 28 April 2018. [Online]. Available: https://docs.ansible.com/ansible/devel/plugins/connection.html. [Accessed 28 April 2018].

[212] anon., "local - execute on controller," 28 April 2018. [Online]. Available: https://docs.ansible.com/ansible/devel/plugins/connection/local.html. [Accessed 28 April 2018].

[213] anon., "hostname -Manage hostname," 26 April 2018. [Online]. Available: http://docs.ansible.com/ansible/latest/modules/hostname_module.html. [Accessed 28 April 2018].

[214] anon., "Variables," 26 April 2018. [Online]. Available: http://docs.ansible.com/ansible/latest/user_guide/playbooks_variables.html. [Accessed 28 April 2018].

[215] anon., "file - Sets attributes of files," 26 April 2018. [Online]. Available: http://docs.ansible.com/ansible/latest/modules/file_module.html. [Accessed 28 April 2018].

[216] anon, "command - Executes a command on a remote node," 26 April 2018. [Online]. Available: http://docs.ansible.com/ansible/latest/modules/command_module.html. [Accessed 28 April 2018].

[217] S. Vokes, "Configuring a Laptop with Ansible, Part One," 21 September 2015. [Online]. Available: https://spin.atomicobject.com/2015/09/21/ansible-configuration-management-laptop/. [Accessed 02 May 2018].

[218] anon., "Turning Off Facts," 26 April 2018. [Online]. Available: http://docs.ansible.com/ansible/latest/user_guide/playbooks_variables.html#turning-off-facts. [Accessed 28 April 2018].

[219] anon., "Intro to Playbooks," 26 April 2018. [Online]. Available: http://docs.ansible.com/ansible/latest/user_guide/playbooks_intro.html#hosts-and-users. [Accessed 28 April 2018].

[220] Adriano, steeldriver, P. Bianchi and L. Vilnius, "How to disable wireless from command line," 201-. [Online]. Available: https://askubuntu.com/questions/597116/how-to-disable-wireless-from-command-line. [Accessed 10 May 2018].

[221] Radu, "Check If a Particular Service Is Running on Ubuntu," 10 September 2013. [Online]. Available: https://stackoverflow.com/a/18721237. [Accessed 28 April 2018].

[222] Tonie, "TX2 config#3 for USB LANE MAPPING," 07 September 2017. [Online]. Available: https://devtalk.nvidia.com/default/topic/1013740/jetson-tx2/tx2-config-3-for-usb-lane-mapping/post/5208042/#5208042. [Accessed 2018 April 2018].

[223] Tonie, "TX2 config#3 for USB LANE MAPPING," 07 September 2017. [Online]. Available: https://devtalk.nvidia.com/default/topic/1013740/jetson-tx2/tx2-config-3-for-usb-lane-mapping/post/5208046/#5208046. [Accessed 28 April 2018].

[224] A. Saha, "Static and Dynamic Libraries | Set 1," ca. 2013. [Online]. Available: https://www.geeksforgeeks.org/static-vs-dynamic-libraries/. [Accessed 30 April 2018].

[225] dusty_nv, "GPU usage monitoring on TX1," 31 March 2016. [Online]. Available: https://devtalk.nvidia.com/default/topic/928386/jetson-tx1/gpu-usage-monitoring-on-tx1/post/4848792/#4848792. [Accessed 30 April 2018].

[226] caxenie, "GPU usage monitoring on TX1," 31 March 2016. [Online]. Available: https://devtalk.nvidia.com/default/topic/928386/jetson-tx1/gpu-usage-monitoring-on-tx1/post/4848804/#4848804. [Accessed 30 April 2018].

[227] caxenie, "GPU usage monitoring on TX1," 31 March 2016. [Online]. Available: https://devtalk.nvidia.com/default/topic/928386/jetson-tx1/gpu-usage-monitoring-on-tx1/post/4848831/#4848831. [Accessed 30 April 2018].

[228] Zanna, "Vim editor, how can I save a file in other directory," 10 November 2016. [Online]. Available: https://askubuntu.com/a/848005. [Accessed 30 April 2018].

[229] NVIDIA Corporation, "JETSON TX1 AND TX2 DEVELOPER KITS," 20 July 2017. [Online]. Available: https://developer.nvidia.com/embedded/dlc/l4t-28-1-jetson-developer-kit-user-guide. [Accessed 02 May 2018].

[230] Silver Moon , "5 commands to check memory usage on Linux," 26 October 2013. [Online]. Available: https://www.binarytides.com/linux-command-check-memory-usage/. [Accessed 03 May 2018].

[231] N. Long, "How can I make chown work recursively?," 22 March 2011. [Online]. Available: https://superuser.com/q/260925. [Accessed 03 May 2018].

[232] R. W, "Commands To Update / Upgrade All Packages in Ubuntu 12.10," 27 December 2012. [Online]. Available: https://www.liberiangeek.net/2012/12/commands-to-update-upgrade-all-packages-in-ubuntu-12-10/. [Accessed 10 May 2018].

[233] NVIDIA Corporation, "NVIDIA TEGRA LINUX DRIVER PACKAGE QUICK START GUIDE," NVIDIA Corporation, 201-. [Online]. Available: http://developer2.download.nvidia.com/embedded/L4T/r28_Release_v1.0/BSP/l4t_quick_start_guide.txt. [Accessed 10 May 2018].

[234] TakTak, "Cannot determine IP address," 04 January 2016. [Online]. Available: https://devtalk.nvidia.com/default/topic/907971/jetson-tx1/cannot-determine-ip-address/post/4771710/#4771710. [Accessed 10 May 2018].

[235] dusty_nv, "JetPack 2.0 on Jetson TK1 - Installation doesn' t finish," 23 November 2015. [Online]. Available: https://devtalk.nvidia.com/default/topic/898683/jetson-tk1/jetpack-2-0-on-jetson-tk1-installation-doesn-t-finish/post/4736127/#4736127. [Accessed 10 May 2018].

[236] Honey_Patouceul and inuxdev, "Can't create new user that can login to GUI," 10 August 2017. [Online]. Available: https://devtalk.nvidia.com/default/topic/1021929/jetson-tx2/can-t-create-new-user-that-can-login-to-gui/post/5201139/. [Accessed 09 May 2018].

[237] J. Adams, *Personal communication,* 2017.

[238] "8.3.3 Remote commands," 14 August 1996. [Online]. Available: The Wikipedia article seems to imply .rlogin or .rhosts are the same file: https://en.wikipedia.org/wiki/Rlogin#Security . [Accessed 27 April 2018].

[239] anon., "Basics of NAMD," n.d.. [Online]. Available:

http://www.ks.uiuc.edu/Training/Tutorials/namd/namd-tutorial-unix-html/node4.html. [Accessed 05 May 2018].

11 Appendices

Appendix A. Safety Assessment Form

Senior Design Engineers Hazard Assessment Form

Team Number: 8 Date assessment completed:
Team Members: Peter Oostema, Noah Pirrotta, Philip Holmes, Ben Kastner
Faculty Mentor: Professor Mark Michmerhuizen
Contact (email or Text #): np26@students.calvin.edu

Project Summary
Project Crayowulf intends to create a mini-supercomputer system with a mechanical enclosure and liquid cooling system. The mechanical enclosure will be made from metal and 3-D printed materials. The enclosure fabrication will involve cutting, welding, machining, etc.. The liquid cooling system will likewise involve metal parts that need to be machined.

Exposure and Protective Equipment Assessment

	Questions	Sample activities	Resulting hazards
☒Y ☐N	Are there activities that produce flying fragments, objects or particles?	Cutting, Milling, CNC (Machine Shop Tools)	**Impact** from flying metal
☐Y ☒N	Does the layout of the workplace and/or location of co-workers present a potential risk of injury? (congested area, flammable materials nearby)		
☒Y ☐N	Do you observe any sources of objects that might pierce the feet or cut the hands?	Machine Shop Equipment	**Penetration** of nails, screws, sheet metal, saw blade
☐Y ☒N	Are there any sources of rolling or pinching objects that could crush the feet?		
☒Y ☐N	Are there twirling blades or parts that could cut or entrap body parts?	Sawing, drilling	**Laceration** from sharp edges rotating
☐Y ☒N	Is there potential for injury to the head from falling objects, moving objects or electrical conductors?		
☒Y ☐N	Is there actual or potential exposure to chemicals?	Spray Painting	**Chemical** – eye irritation, fumes
☒Y ☐N	Are there any sources of high temperatures that could cause burns, eye injury, ignition of protective equipment or heat stress?	Welding	**Heat**, ie. hot sparks, splash, hot surfaces, sunburn
☒Y ☐N	Are there any sources of harmful dust?	Grinding/sanding	**Harmful dust,** ie.metal dust
☒Y ☐N	Are hands exposed to lacerations or abrasions?	Cutting and sanding	**Laceration**, abrasion
☒Y ☐N	Are there any sources of light radiation?	Welding	**Light (optical) Radiation,** ie. Intense radiation, glare
☒Y ☐N	Are there any electrical hazards?	Electrical wiring, circuitry, power supply	**Electrical hazards** including working with power supply wiring
☐Y ☒N	Is there exposure to biological hazards?		
☐Y ☒N	Are there potential fall hazards?		
☒Y ☐N	Are there excessive noise levels?	Machine Shop Equipment	**Noise exposure**

CHEMICAL HAZARDS

Chemical Type	Present	Task/Job	Affected area (mark all that apply)
Compressed gas	☒Y ☐N	**Shop Compressed gas hose**	☒eyes ☐lungs ☒skin
Carcinogen	☒Y ☐N	**Welding Fumes**	☐eyes ☒lungs ☐skin
Corrosive	☐Y ☒N		☐eyes ☐lungs ☐skin
Cryogenic	☐Y ☒N		☐eyes ☐lungs ☐skin
Flammable/Combustible	☒Y ☐N	**Spray Paint**	☒eyes ☒lungs ☐skin
Highly Toxic	☐Y ☒N		☐eyes ☐lungs ☐skin
Irritant	☒Y ☐N	**Welding Fumes**	☐eyes ☒lungs ☐skin
Pesticide	☐Y ☒N		☐eyes ☐lungs ☐skin
Pyrophoric	☐Y ☒N		☐eyes ☐lungs ☐skin
Sensitizer	☐Y ☒N		☐eyes ☐lungs ☐skin
Water Reactive	☐Y ☒N		☐eyes ☐lungs ☐skin

Please use the chart below to indicate (check) what areas are affected by the identified hazards.

HAZARD SUMMARY

Hazard	Eye/face	Head	Hand	Skin	Foot	Lungs	Hearing
Impact	☒	☐	☒	☒	☐	☐	☐
Penetration	☒	☐	☒	☒	☒	☐	☐
Compression or "roll-over"	☐	☐	☐	☐	☐	☐	☐
Laceration or Abrasion	☐	☐	☒	☒	☐	☐	☐
Chemical	☒	☐	☐	☐	☐	☒	☐
Heat	☒	☐	☒	☐	☐	☐	☐
Harmful dusts	☐	☐	☒	☒	☐	☒	☐
Light (optical) Radiation	☒	☐	☒	☒	☐	☐	☐
Electrical hazards	☐	☐	☒	☒	☐	☐	☐
Disease transmission	☐	☐	☐	☐	☐	☐	☐
Slips, Trips or Falls	☐	☐	☐	☐	☐	☐	☐
Noise	☐	☐	☐	☐	☐	☐	☒

For each of the boxes checked in the hazard summary above, please indicate how your team members will be protected (i.e., type of PPE, isolation of hazard, distance from hazard, etc.).

Hazard	Protection Measure
Impact	**Safety Glasses, Work Gloves (when appropriate)**
Penetration	**Safety Glasses, Work Gloves (when appropriate)**
Laceration or Abrasion	**Work Gloves if appropriate**
Chemical	**Safety Glasses, Filtration Mask**
Heat	**Welding Mask, Welding Gloves**
Harmful Dusts	**Avoid dusting metal off with hands – used compressed air, Gloves if necessary**
Light Radiation	**Welding Mask, Welding Gloves**
Electrical Hazards	**Disconnect Power when possible, Properly Ground**
Noise	**Hearing protection – Earplugs/Headphones**

Appendix B. Additional Figures and Tables

ID		Task Mode	Task Name	Duration	Start	Finish
0			**Team 8 Gantt Chart**	**171 days?**	**Fri 9/15/17**	**Sat 5/12/18**
1			**Senior Design Deliverables**	**155 days**	**Tue 10/10/17**	**Sat 5/12/18**
2	✓		Work Breakdown Structure	2 days	Tue 10/10/17	Wed 10/11/17
3	✓		Notify Professor Nielsen	3 days	Wed 10/11/17	Fri 10/13/17
4	✓		Budget Proposal	4 days	Wed 10/11/17	Mon 10/16/17
5	✓		Research	5 days	Mon 10/16/17	Fri 10/20/17
6	✓		Website	6 days	Mon 10/23/17	Mon 10/30/17
7	✓		Project Poster	5 days	Mon 10/30/17	Fri 11/3/17
8	✓		PPFS Outline	4 days	Wed 11/1/17	Mon 11/6/17
9	✓		Draft PPFS	6 days	Mon 11/6/17	Mon 11/13/17
10	✓		Safety Assessment	6 days	Mon 11/13/17	Mon 11/20/17
11	✓		Final PPFS	20 days	Tue 11/14/17	Mon 12/11/17
12	✓		Second Presentation	5 days	Mon 11/27/17	Fri 12/1/17
13			Senior Design Night Presentation	7 days	Sat 4/28/18	Sat 5/5/18
14			Senior Design Final Report	10 days	Tue 5/1/18	Sat 5/12/18
15			**CS Deliverables**	**153 days**	**Sat 9/30/17**	**Tue 5/1/18**
16	✓		Proposal	16 days	Sat 9/30/17	Fri 10/20/17
17	✓		CS Pesentation 1	5 days	Wed 11/29/17	Tue 12/5/17
18			CS Pesentation 2	5 days	Wed 4/25/18	Tue 5/1/18
19	✓		Computer Parts Ordered	7 days	Tue 10/24/17	Wed 11/1/17
20			Buffer Zone	21 days	Mon 4/2/18	Mon 4/30/18
21			**Mechanicial Enclosure**	**135 days**	**Tue 10/10/17**	**Sun 4/15/18**
22	✓		CAD Design Draft	27 days	Tue 10/10/17	Wed 11/15/17
23			CAD Design Final	14 days	Mon 1/29/18	Thu 2/15/18
24			Case Fabrication	21 days	Thu 2/15/18	Thu 3/15/18
25			System Assembly	4 days	Thu 3/15/18	Tue 3/20/18
26			System Tuning and Benchmaking	10 days	Tue 3/20/18	Sun 4/1/18
27			**Computer Engineering**	**138 days**	**Sat 10/21/17**	**Tue 5/1/18**
28			Configuring Operation System	4 days	Wed 11/15/17	Mon 11/20/17
29			Configuring Network Hard Drive	8 days	Tue 11/21/17	Thu 11/30/17
30			Wiring	1 day	Fri 12/1/17	Fri 12/1/17
31			API's	22 days	Mon 1/1/18	Tue 1/30/18
32			Research Cryptography Scheme	67 days	Fri 9/15/17	Mon 12/18/17
33			First Implementation	43 days	Mon 1/1/18	Wed 2/28/18
34			Improvements/Parallelize	44 days	Thu 3/1/18	Tue 5/1/18
35			**Cooling System Configuration**	**89 days**	**Tue 10/10/17**	**Fri 2/9/18**
36			Settle On Parts	74 days	Mon 10/23/17	Thu 2/1/18
37			Cooling parts Ordered	1 day	Fri 2/9/18	Fri 2/9/18
38			Water Block & Reservoir Design	43 days	Mon 1/1/18	Wed 2/28/18
39			Water Block & Reservoir Fabrication	12 days	Wed 2/28/18	Thu 3/15/18

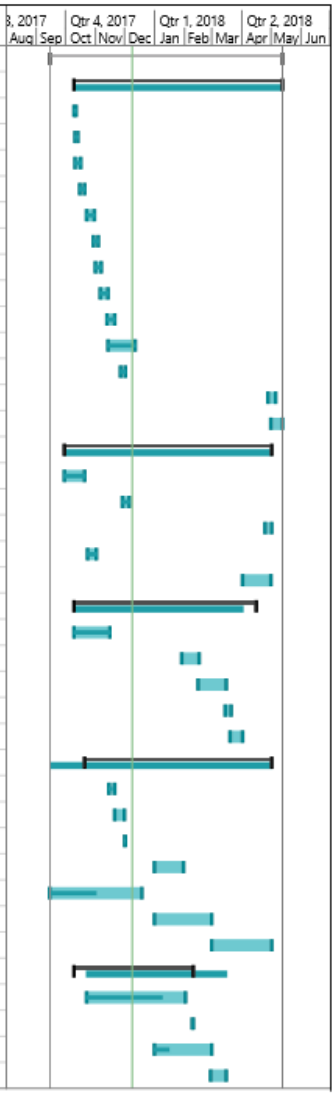


Figure B63. Initial Project Gantt Chart

ID	Task Mode	Task Name	Duration	Start	Finish
0		**Team 8 Gantt Chart**	**171 days?**	**Fri 9/15/17**	**Sat 5/12/18**
1		**Senior Design Deliverables**	**155 days**	**Tue 10/10/17**	**Sat 5/12/18**
2		Work Breakdown Structure	2 days	Tue 10/10/17	Wed 10/11/17
3		Notify Professor Nielsen	3 days	Wed 10/11/17	Fri 10/13/17
4		Budget Proposal	4 days	Wed 10/11/17	Mon 10/16/17
5		Research	5 days	Mon 10/16/17	Fri 10/20/17
6		Website	6 days	Mon 10/23/17	Mon 10/30/17
7		Project Poster	5 days	Mon 10/30/17	Fri 11/3/17
8		PPFS Outline	4 days	Wed 11/1/17	Mon 11/6/17
9		Draft PPFS	6 days	Mon 11/6/17	Mon 11/13/17
10		Safety Assessment	6 days	Mon 11/13/17	Mon 11/20/17
11		Final PPFS	20 days	Tue 11/14/17	Mon 12/11/17
12		Second Presentation	5 days	Mon 11/27/17	Fri 12/1/17
13		Senior Design Night Presentation	7 days	Sat 4/28/18	Sat 5/5/18
14		Senior Design Final Report	8 days	Tue 5/1/18	Thu 5/10/18
15		**CS Deliverables**	**153 days**	**Sat 9/30/17**	**Tue 5/1/18**
16		Proposal	16 days	Sat 9/30/17	Fri 10/20/17
17		Computer Parts Ordered	7 days	Tue 10/24/17	Wed 11/1/17
18		CS Pesentation 1	5 days	Wed 11/29/17	Tue 12/5/17
19		Buffer Zone	21 days	Mon 4/2/18	Mon 4/30/18
20		CS Pesentation 2	5 days	Wed 4/25/18	Tue 5/1/18
21		**Mechaniclal Enclosure**	**149 days**	**Tue 10/10/17**	**Fri 5/4/18**
22		CAD Design Draft	27 days	Tue 10/10/17	Wed 11/15/17
23		CAD Design Final	14 days	Mon 1/29/18	Thu 2/15/18
24		Case Fabrication	52 days	Thu 2/15/18	Fri 4/27/18
25		System Assembly	6 days	Fri 4/27/18	Fri 5/4/18
26		System Tuning and Benchmaking	11 days	Fri 4/20/18	Fri 5/4/18
27		**Cooling System Configuration**	**135 days**	**Mon 10/23/17**	**Fri 4/27/18**
28		Settle On Parts	74 days	Mon 10/23/17	Thu 2/1/18
29		Cooling parts Ordered	1 day	Fri 2/9/18	Fri 2/9/18
30		Water Block & Reservoir Design	43 days	Mon 1/1/18	Wed 2/28/18
31		Water Block & Reservoir Fabrication	43 days	Wed 2/28/18	Fri 4/27/18
32		**Computer Engineering**	**162 days**	**Fri 9/15/17**	**Mon 4/30/18**
33		Research Cryptography Scheme	67 days	Fri 9/15/17	Mon 12/18/17
34		Configuring Operation System	4 days	Wed 11/15/17	Mon 11/20/17
35		Configuring Network Hard Drive	8 days	Tue 11/21/17	Thu 11/30/17
36		Wiring	1 day	Fri 12/1/17	Fri 12/1/17
37		API's	22 days	Mon 1/1/18	Tue 1/30/18
38		First Implementation	66 days	Mon 1/1/18	Sat 3/31/18
39		Improvements/Parallelize	22 days	Sun 4/1/18	Mon 4/30/18

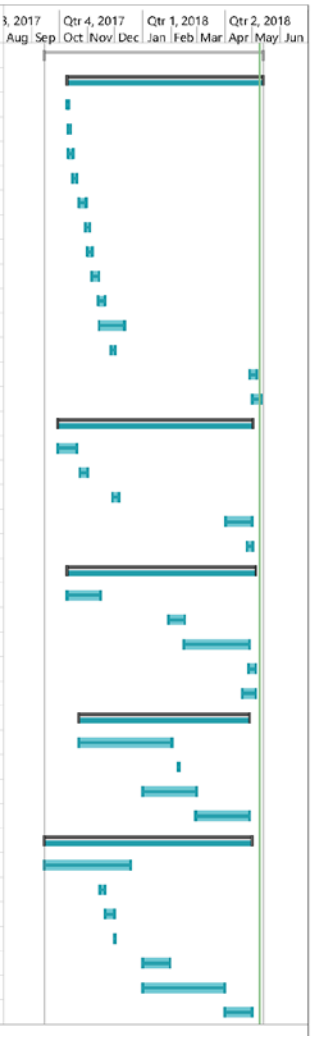


Figure B64. Final Project Gantt Chart

Table B4. Items Charged to Computer Science Project Budget

Item	Quantity	Source	Notes	Final Cost
Jetson TX2 developer kits	5	http://www.nvidia.com/object/jetsontx2-edu-discount.html	Not including fifth Jetson because it was donated.	$1,209.33
Extra 2-port network adapters (PCIe card)	4	https://www.amazon.com/Intel-Corp-EXPI9402PTBLK-1000-PTDualPort/dp/B0084P52JW		$114.93
Extra 4-port network adapter for head node (PCIe card)	1	https://www.amazon.com/EXPI9404PTLBLK-Express-Profile-Server-Adapter/dp/B005DASO3M		$44.00
USB male to female 2 ft extender cables	4		Two are used to connect to the I/O panel on the back of the case. Only two are strictly necessary. Having extra is useful during setup when one is plugging and unplugging peripherals from several Jetsons.	$6.99
SATA power and data cable for hard drive	1	https://www.amazon.com/SMAKN-22-pin-Female-Power-Extension/dp/B00L9R3AKA/ref=sr_1_3?s=electronics&ie=UTF8&qid=1509230431&sr=1-3&keywords=SATA+22+Pin		
SATA 90 degree connector	1	https://www.amazon.com/Ocr-SATA-Female-Degree-Interface/dp/B017XF7ZM0/ref=sr_1_1?s=electronics&ie=UTF8&qid=1509230122&sr=1-1&keywords=sata+right+angle+adapter		
Network switch (unmanaged)	1	https://www.amazon.com/NETGEAR-16-Port-Gigabit-Ethernet-Desktop/dp/B01AX8XHRQ		$77.24

120 Gb SSD	1	https://www.amazon.com/SanDisk-120GB-SDSSDA-120G-G26-Newest-Version/dp/B01F9G414U		
4-Pin Male Molex Connector to 12V DC 5.5mm x 2.1mm Plug 16" LED Power Cable	1	https://www.amazon.com/gp/product/B0121QHR2E/ref=oh_aui_search_detailpage?ie=UTF8&psc=1	Used to power network switch.	$75.83
DC Power 12V 5.5mm x 2.5mm Barrel Male Plug Connector Pigtail	5	https://www.amazon.com/gp/product/B016IGA60S/ref=oh_aui_search_detailpage?ie=UTF8&psc=1	Used to power the Jetsons.	$15.00
HDMI male to female extension cable	1		Used to connect to the I/O panel on the back of the case.	$6.88
Total				$1,550.20

Table B5. Proposed Engineering Budget

	Number of Units	Price Per Unit ($/unit)	Total Price ($)
Pump	1	35.99	35.99
Tubing (Large)	10	1.09	10.90
Radiator	1	16.99	16.99
Fittings	3	8.99	26.97
Rubber Sheet	1	8.56	8.56
Water-Proof Adhesive	1	10.00	10.00
Fan	1	25.95	25.95
Fan controller	1	15.00	15.00
Silver Coil	1	7.00	7.00
Lighting			40.00
3-D Printed Materials			75.00
Additional Case Materials and Opening Mechanism			75.00
Shipping			27.15
Contingency			100.00
Total Cost ($)			474.51

Table B6. Engineering Expenses

		Number of Units	Price Per Unit ($/unit)	Total Price ($)	Shipping ($)	Balance
					Starting Balance:	474.51
Order 1	Pump	1	34.73	34.73	4.74	439.78
	Tubing 10 ft	1	5.49	5.49		434.29
	Radiator	1	16.99	16.99		417.30
	Fittings 4 Pack	3	9.99	29.97		387.33
	Rubber Sheet	1	9.86	9.86		377.47
	Fan	1	25.95	25.95		351.52
	Fan controller	1	12.99	12.99		338.53
	Silver Coil	1	8.49	8.49		330.04
	Zip Ties	1	11.99	11.99		313.31
Order 2	Arduino Uno	1	22.00	22.00	27.30	291.31
	Neopixel 2m LED Strip	1	49.90	49.90		241.41
	MOSFET 30V 12 pack	1	3.97	3.97		237.44
	Diode 30V 12 pack	1	2.20	2.20		235.24
	Resistor 470k ohm 12 pack	1	0.61	0.61		234.63
	Resistor 10k ohm 12 pack	1	0.60	0.60		234.03
	Capacter 4.7UF	1	0.89	0.89		205.84
Order 3	Magnetic Cabinet Catch	1	8.00	8.00	0.00	197.84
	Vinyl Tubing 10 ft	1	4.00	4.00		193.84
	Cabinet Handles	1	12.59	12.59		181.25
	Alphacool 3-way Connector	1	8.95	8.95		172.30
	Alphacool 2-way Ball Valve	1	12.99	12.99		159.31
	Fittings 4 Pack	1	11.98	11.98		147.33
	40P Dupont Cable pack	1	6.99	6.99		140.34
Order 4	Vinyl Tubing 10 ft	1	8.00	8.00	0.00	132.34
	Pressure equilizer Stop Plug	1	7.99	7.99		124.35
Order 5	1.5 ft Ethernet Extention Cable	1	5.69	5.69	0.00	118.66
	Fan Filter 10 Pack	1	6.19	6.19		112.47
	Liquid Cooling Plugs 4 pack	1	10.98	10.98		101.49
Order 6	1.5 ft Cat 6 Ethernet Cable 6 pack	1	8.95	8.95	0.00	92.54

Appendix C. Node Setup

A separate PC running Ubuntu Linux is necessary to use Jetpack. Check the Jetpack documentation to see which version of Ubuntu is recommended for the current version of Jetpack.

The Ansible playbooks assume that they are being run from the `nvidia` account and that the `ansible` repository is at `/home/nvidia/ansible` and the files repository is located at `/home/nvidia/files`.

This guide assumes that the Jetsons have had no software modifications made since being removed from their packaging. If it is desired to use a later version of Linux for Tegra than the board shipped with, first flash the OS using Jetpack, then work through the setup steps (minus the on-screen instructions regarding the installation of the NVIDIA drivers).

A properly grounded antistatic mat and wrist strap are recommended to prevent damage to your components.

Setup steps are provided below for the head and worker nodes. There are many similarities but also some differences.

Head Node

- Install the PCIe network card.
- Connect monitor, keyboard and mouse, but NOT ethernet.
- Boot the node and follow the on-screen instructions to install the NVIDIA drivers, reboot when done.
- Log in and disable WiFI (click the network indicator in the upper right of the desktop and uncheck “Enable WiFI”). (default password for `nvidia` account is `nvidia` and `ubuntu` for the `ubuntu` account)
- Change the passwords on the ubuntu and nvidia accounts.
- Use a flash drive to transfer `files/head/iptables.rules`, `files/head/sshd_config` and SSH public key (generated on your workstation) to the node. Hint: It’s easiest to unplug the keyboard and use your mouse to copy the files to the desktop.
- The SSH public key goes in ~./ssh/authorized_keys
- Load the firewall rules with `sudo iptables-restore <path/to/iptables.rules>`
- Check that the firewall rules were loaded using `sudo iptables -L -v`
- Place `sshd_config` in `/etc/ssh/sshd_config.` You will need to use `sudo` to place `sshd_config.`
- Restart the SSH service: `sudo service ssh restart`.
- Connect the ethernet cable to the on-board port, not the PCIe card.
- Run `sudo apt update && apt dist-upgrade`. This is to update the OS and packages on the node. It will take a long time.

- Edit `/etc/apt/sources.list` and uncomment the lines that enable the universe repository.
- Run `sudo apt update && sudo apt install iptables-persistent`, saving the firewall rules when prompted.
- Reboot
- Edit `/etc/ssh/sshd_config` to allow password authentication (`PasswordAuthentication yes`). This is necessary because Jetpack does not support public key SSH. Later we will undo this change.
- Restart the SSH service: `sudo service ssh restart`
- Use Jetpack to perform a full install on the board (do not flash the OS, as that will overwrite the previous steps).
- Disable password SSH (`PasswordAuthentication no`).
- Restart the SSH service: `sudo service ssh restart`
- Change directory to `/home/nvidia`.
- `sudo apt-get update`
- `sudo apt-add-repository ppa:ansible/ansible`
- `sudo apt-get update`
- `sudo apt-get install ansible`
- `git clone https://github.com/crayowulf/ansible.git`
- `git clone https://github.com/crayowulf/files.git`
- Install the `nis` package with `sudo apt update && apt install nis` (Provide the domain name `crayowulf.nis` when prompted for it).
- Run the `headNodeSetup` playbook to finish the rest of the basic setup tasks: `ansible-playbook -K -c local -i inventory headNodeSetup.yml.`
- Reboot
- Make the mount point for the NFS share: `sudo mkdir /home2`. Note that it would be wise to create two partitions on the drive, perhaps one called `/home2` and another called `/share`. See section 7 of this report for a more detailed explanation. Format the SSD with `sudo mkfs.ext4 /dev/sda1 -L /home2`. It was necessary to adjust the path to the drive on the Jetson, although this step was not recorded. This part of setup is based on the NFS instructions from [80].
- Run the `mpi` and `vmd` playbooks on the head node as shown in the Ansible section of this document to install CUDA-aware OpenMPI and VMD.

Worker Node

- Install the PCIe network card.
- Connect monitor, keyboard and mouse, but NOT ethernet.
- Boot the node and follow the on-screen instructions to install the NVIDIA drivers
- Reboot.
- Log in and disable WIFI (click the network indicator in the upper right of the desktop and uncheck “Enable WIFI”).
- Change the passwords on the ubuntu and nvidia accounts.
- Use a flash drive to transfer `files/worker/iptables.rules`, `files/worker/sshd_config`, and SSH public key (this time generated on `headNode`) to the node. Hint: It’s easiest to unplug the keyboard and use your mouse to

copy the files to the desktop. Note that the firewall rules on the worker nodes are only necessary during the initial setup process, when they are connected directly to the outside world. At all other times, they are connected to the network switch and are protected by the firewall rules
    - The SSH public key is from the head node and goes in `~./ssh/authorized_keys`
    - Load the firewall rules with `sudo iptables-restore <path/to/iptables.rules>`
    - Check that the firewall rules were loaded using `sudo iptables -L -v`
    - Place `sshd_config` in `/etc/ssh/sshd_config` and restart the ssh service using `sudo service ssh restart`. You will need to use `sudo` to place `sshd_config`
- Connect the ethernet cable to the board
- Edit `/etc/ssh/sshd_config` to allow password authentication (`PasswordAuthentication yes`).
- Restart the SSH service: `sudo service ssh restart`
- Use Jetpack to perform the "Install on Target" (do not flash the OS, as that will overwrite the previous steps).
- While the packages install, run `ifconfig` and get the MAC address of eth1. Edit head/isc-dhcp-server and place eth1's MAC address in the appropriate place, depending on which number this worker is (`worker01, worker02, etc.`).
- Disable password SSH (`PasswordAuthentication no`).
- Restart the SSH service: `sudo service ssh restart`
- Shutdown the cluster
- Run the `headNodeSetup.yml` playbook to update all the configuration files on the head node pertaining to adding a worker.
- Disconnect the ethernet cable and connect the node to the network switch.
- Make sure it gets the proper IP address and that it is possible to SSH to it from the head node.
- `sudo apt update`
- `sudo apt dist-upgrade`. This is to update the OS and packages on the node and will take a long time.
- Edit `/etc/apt/sources.list` and uncomment the lines that enable the universe repository.
- Install the `nis` package with `sudo apt update && apt install nis` (Provide the domain name `crayowulf.nis` when prompted for it).
- From the `headNode`, run the `workerNodeSetup.yml` playbook to finish basic setup.
- Reboot.
- Run the `mpi` and `vmd` playbooks as shown in the Ansible section of this document to install CUDA-aware OpenMPI and VMD.
- Repeat for the rest of the worker nodes.

Appendix D. Lighting and Encrypted Shell Readme File

Github:Code ls://github.com/crayowulf/

The lighting directory contains the code run on the Arduino. The program contains several lighting functions and a method for reading from the Serial port with minimal delay. The program is simply loaded onto the Arduino from the Arduino IDE and controlled by a C program.

The directory `writeToArduinoSerial` contains the controlling program for the Arduino lighting system. It contains a makefile so simply running `make` compiles the file. The program reads in the system information from a text file and sends the necessary parts to the Arduino on its Serial port. After executing the program several options are available. Typing 1 gives a rainbow flowing across the LEDs; 2 gives a changing single color flow across the LEDs; 3 displays the temperatures of the system, CPU temps and GPU temps are shown on four LEDs each, each node's temperatures are displayed between each gap on the top of the system and repeated along the supports; 4 displays the node usage information, however there is an error in TEGRASTATS that reads each usage as 0%, so this will not display any useful information; 5 displays a fade between RGB, and CMY; 6 displays an improved fade between all the colors in 5, without going to black in-between.

The directory `siec` includes a client and server for running an SIEC encrypted command shell. To compile run the instruction `make arch=ARM64 CC=gcc`, for other systems replace the architecture and c compiler with those needed there.

The directory `sysInfo` contains scripts for obtaining the temperature and usage information from the cluster. The system will not run RSH instructions from a script so `perBoard.sh` needs to be run on each node. One node then runs `shell.sh` to gather all of the individual information and save it in one file.

Appendix E. Cluster Maintenance

To add a user, use the command `sudo adduser --home /home2/<newuser> <newuser>`. Then run `sudo adduser <newuser> video` to add the user to the video group – otherwise the new user will not be able to log in on any of the nodes. This thread gave the solution and stated that adding the user to the video group gives him or her permission to use the GPU to display the Ubuntu desktop [236]. Note that after adding or modifying a user, it is necessary to run the command `sudo make -C /var/yp/` to propagate the changes to the worker nodes [71].

Appendix F. Theoretical Thermal Analysis EES Code

```
"Team 8 - Crayowulf
Senior Design
Cooling Block Analysis"


"!Assumptions for thermal properties"
T_air_in = 25 [C]                                   "Air Temperature"
P_air = 101.3 [kPa]                                 "Assume standard atmospheric pressure"
epsilon = 0.4

T_w = 30 [C]                                        "Assumed water temperature for properties"

P_w = 50 [psi]                                      "Pressure including atmospheric"
P_w_SI = convert(psi, kPa) * P_w

"!Assumed Areas and Lengths"
A = 0.002 [m^2]
L_1 = 0.00465 [m]                                   "NJ Plate Thickness"
L_2 = 0.00127 [m]                                   "Cooling Block Plate Thickness"
D_tube = (5/16 * 0.0254) [m]
A_tube = pi * D_tube ^ 2 / 4
w_rect = 0.015 [m]
h_rect  = 0.017 [m]
L_char = 0.122 [m]                                  "Approximate Length From Block inlet to exit (down middle)"
L_paste = (0.005 * 0.0254) [m]

"!Thermophysical Properties"
k_al = 177 [W/(m*K)]                                "From http://asm.matweb.com/search/SpecificMateri
al.asp?bassnum=ma6061t6"
k_paste = 8.9 [W/(m*K)]                             "https://www.gamingfactors.com/best-thermal-paste/"
k_w = conductivity(Water,T=T_w,P=P_w_SI)
cp_w = cp(Water,T=T_w,P=P_w_SI)
rho_w = density(Water,T=T_w,P=P_w_SI)
mu_w = viscosity(Water,T=T_w,P=P_w_SI)
Pr_w = prandtl(Water,T=T_w,P=P_w_SI)

rho_air = density(Air,T=T_air_in,P=P_air)
cp_air = cp(Air,T=T_air_in)

"!Assumed Flow Rates and Velocity"
Vol_dot_w = 400 [L/hr]                              "Assumed based on data sheet"
Vol_dot_wSI = convert(L/hr, m^3/s) * Vol_dot_w
m_dot_w = rho_w * Vol_dot_wSI
V_w = Vol_dot_wSI / A_tube
Vol_dot_air = 121.8 [m^3/hr]
Vol_dot_airSI = Vol_dot_air * convert(m^3/hr, m^3/s)


"!Heat Transfer Assumptions"
Q_dot_NJ = 15 [W]                                   "Per unit Power Draw"

"!Convection Heat Xfer Coefficient Calcualtions"
"Information from https://www.uio.no/studier/emner/matnat/math/MEK4450/h11/undervisningsmateriale/modul-5/
Pipeflow_intro.pdf and http://web2.clarkson.edu/projects/subramanian/ch330/notes/Heat%20Transfer%20in
%20Flow%20Through%20Conduits.pdf"
D_h = 2 * w_rect * h_rect / (w_rect + h_rect)       "Hydraulic Diameter of rectangular cross-section"
Re = rho_w * V_w * D_h / mu_w                       "Turbulent if above 4000"                    "Last Check ~5*
10^4"

Nusselt = 0.023 * Re^0.8 * Pr_w^0.4                 "Dittus-Boelter"
h_conv = Nusselt * k_w / D_h
```

```
"!Estimate of Water temperatures"
Q_dot_NJ = m_dot_w * cp_w * (T_2 - T_1)          "CB 1"
Q_dot_NJ  = m_dot_w * cp_w * (T_3 - T_2)         "CB 2"
Q_dot_NJ  = m_dot_w * cp_w * (T_4 - T_3)         "CB 3"
Q_dot_NJ  = m_dot_w * cp_w * (T_5 - T_4)         "CB 4"
Q_dot_NJ  = m_dot_w * cp_w * (T_6 - T_5)         "CB 5"

T_w1 = average(T_1,T_2)
T_w2 = average(T_2,T_3)
T_w3 = average(T_3,T_4)
T_w4 = average(T_4,T_5)
T_w5 = average(T_5,T_6)

"!Heat Transfer for NJ Die"
"Total Thermal Resistance Each NJ"
R_1 = L_1 / (k_al *A)
R_2 = L_paste / (k_paste * A)
R_3 = L_2 / (k_al *A)
R_4 = 1 / (h_conv * A)
R_tot = R_1 + R_2 + R_3 + R_4
Q_dot_NJ = (T_NJ1 - T_w1) / R_tot                "NJ 1"
Q_dot_NJ = (T_NJ2 - T_w2) / R_tot                "NJ 2"
Q_dot_NJ = (T_NJ3 - T_w3) / R_tot                "NJ 3"
Q_dot_NJ = (T_NJ4 - T_w4) / R_tot                "NJ 4"
Q_dot_NJ = (T_NJ5 - T_w5) / R_tot                "NJ 5"



"!Radiator (Heat Exchanger) Heat Transfer"
"Air Analysis"
m_dot_air = Vol_dot_airSI * rho_air
C_dot_air = cp_air * m_dot_air

"Water Analysis"
C_dot_w = cp_w * m_dot_w

"Maximum Heat Transfer"
Q_dot_HEmax = C_dot_air * (T_6 - T_air_in)


"Actual Heat Transfer"
Q_dot_HEact = epsilon * Q_dot_HEmax
Q_dot_HEact = 75 [W]


SOLUTION
Unit Settings: SI C kPa J mass deg
A = 0.002 [m^2]
A_tube = 0.00004948 [m^2]
cp_air = 1005 [J/kg*K]
cp_w = 4182 [J/kg*K]
C_dot_air = 40.23 [W/K]
C_dot_w = 462.7 [W/K]
D_h = 0.01594 [m]
D_tube = 0.007938 [m]
epsilon = 0.4
h_conv = 9055 [W/m^2*K]
h_rect = 0.017 [m]
k_al = 177 [W/(m*K)]
```

k_{paste} = 8.9 [W/(m*K)]
k_w = 0.6031 [W/m*K]
L_1 = 0.00465 [m]
L_2 = 0.00127 [m]
L_{char} = 0.122 [m]
L_{paste} = 0.000127 [m]
μ_w = 0.0007977 [kg/m*s]
$\dot{m}_{air}$ = 0.04005 [kg/s]
$\dot{m}_w$ = 0.1106 [kg/s]
Nusselt = 239.3
Pr_w = 5.532
P_{air} = 101.3 [kPa]
P_w = 50 [psi]
$P_{w,SI}$ = 344.7 [kPa]
$\dot{Q}_{HEact}$ = 75 [W]
$\dot{Q}_{HEmax}$ = 187.5 [W]
$\dot{Q}_{NJ}$ = 15 [W]
Re = 44672
ρ_{air} = 1.184 [kg/m^3]
ρ_w = 995.8 [kg/m^3]
R_1 = 0.01314 [C/W]
R_2 = 0.007135 [C/W]
R_3 = 0.003588 [C/W]
R_4 = 0.05522 [C/W]
R_{tot} = 0.07908 [C/W]
T_1 = 29.5 [C]
T_2 = 29.53 [C]
T_3 = 29.56 [C]
T_4 = 29.6 [C]
T_5 = 29.63 [C]
T_6 = 29.66 [C]
$T_{air,in}$ = 25 [C]
T_{NJ1} = 30.7 [C]
T_{NJ2} = 30.73 [C]
T_{NJ3} = 30.77 [C]
T_{NJ4} = 30.8 [C]
T_{NJ5} = 30.83 [C]
T_w = 30 [C]
T_{w1} = 29.51 [C]
T_{w2} = 29.55 [C]
T_{w3} = 29.58 [C]
T_{w4} = 29.61 [C]
T_{w5} = 29.64 [C]
$\dot{Vol}_{air}$ = 121.8 [m^3/hr]
$\dot{Vol}_{airSI}$ = 0.03383 [m^3/s]
$\dot{Vol}_w$ = 400 [L/hr]
$\dot{Vol}_{wSI}$ = 0.0001111 [m^3/s]
V_w = 2.245 [m/s]
w_{rect} = 0.015 [m]

No unit problems were detected.

KEY VARIABLES

Nvidia Jetson Temperatures

T_{NJ1} = 30.7 [C]
T_{NJ2} = 30.73 [C]
T_{NJ3} = 30.77 [C]
T_{NJ4} = 30.8 [C]
T_{NJ5} = 30.83 [C]

Average Temperature of Water in Water Blocks

T_{w1} = 29.51 [C]
T_{w2} = 29.55 [C]
T_{w3} = 29.58 [C]
T_{w4} = 29.61 [C]
T_{w5} = 29.64 [C]