Researcher Guide For The H.aHKBox

Adam Stammer under the direction of Dr. Carl Ferkinhoff

(Dated: May 30, 2020)

Abstract

This document is intended to help guide future researchers in their research and development of the Hardware.astronomy: Housekeeping Box. I began working on this project in the summer of 2019, under the direction of Dr. Carl Ferkinhoff, at Winona State University. This guide summarizes some of the thought processes and plans that went into this project, the tools and softwares used in its development, and gives tips on mistakes made and lessons learned.

I. OVERVIEW OF PROJECT

A. History

Carl Ferkinhoff designed and built the ZEUS Radio Spectrometer while getting his PhD in Astronomy (2014). The device itself uses non visible light frequencies to take 'pictures' of the sky. It's tempting to consider it a telescope, but technically spectrometer is a more apt title. After learning so much with the ZEUS, Carl began designing its predecessor. With the success of the ZEUS, he was able to acquire funding for the ZEUS2. The spectrometer itself is roughly a cylinder about 3' by 4', though I haven't seen it in person so I'm just guessing on the size. Specifics of the history on this remain fuzzy to me, which is why I won't go into more detail in this section. It's fun to know what led to you working on the project, but understanding this is not crucial to success in future development. I strongly suggest you ask Carl about it! I'm sure he'd love to explain it all far better than I can. You can find more information on the background of Dr. Ferkinhoff, ZEUS(2), and his other projects at the following links. `https://sites.google.com/site/carlferkinhoff/welcome-1`
`http://hosting.astro.cornell.edu/~spifiweb/zeus2/index.html`

B. Background: Electromagnetic Interference

It is known that everything gives off various forms of electromagnetic radiation. Some of that light falls under the spectrum that ZEUS2 captures. This means that anything that ZEUS2 looks at can interfere with the astronomical sources that ZEUS2 is concerned with. The air between ZEUS2 and the sky for example. I like to imagine taking a thermal image when it is really hot out. The air itself will make the image fuzzier and less informative, as hot physical objects just blend in with the air in the picture. Only the most hot things will actually show in the picture because they need to overcome the light given off by the air. This is one reason that astronomical observers, like spectrometers, are placed as high in altitude as conveniently possible, like on the top of a mountain. It decreases interference from the air, clouds, and the earth itself. Some of these devices even make it out into space, like the Hubble Telescope. ZEUS2 stays on the ground but not anywhere near Winona, MN. As such, I've never seen the device in person. For a tangible demonstration of the light spectrum, I found `https://viewspace.org/interactives/unveiling_invisible_`

universe/electromagnetic_spectrum/forms_of_light to be a big help. I think an Optics class would cover this kind of thing in depth, but as I have not taken Optics, I can't say for certain.

ZEUS2 is also an object of matter, meaning that it too gives off electromagnetic radiation. Imagine putting on a headlamp that was turned back towards yourself. It's hard to see past the bright light that you can't get away from. Thankfully this electromagnetic radiation is a result of inherent heat in the given objects. The colder they get, the less radiation they give off, and not just in infrared. This means that lifting these observation devices higher decrease interference not only because there is less stuff between the observer and the observed, but also because what is between is colder.

C. ZEUS2 Cool 4 U

By simply cooling down ZEUS2 itself, we can limit how much it interferes with the radiation it is trying to observe. Just cold isn't good enough though. We need it really really cold. The cooler itself functions on the same principles a refrigerator cooler. Things get colder when they decompress, and warmer when they compress. So, just compress something, move the heat away (like a fan), then decompress it, and you've now cooled down that something. A refrigeration unit, will use a coolant, liquid or gas, that's usually inert so as to not cause accidental chemical reactions (Helium for example). Use the process described above to cool said coolant down, and the use the coolant to cool down something else.

A food freezer can run off of a single refrigeration unit, but that's not cold enough for ZEUS2. You would have to compress your coolant down way way too much to get it cold enough for ZEUS2. To make this process more economical, you can do it in stages. Compress, Cool, Decompress, Repeat. These are usually called Multistage Refrigeration Systems and can be implemented differently. Some have multiple compressors, one for each stage, while others just loop the coolant through a single compressor multiple times. Sometimes I like to use an oversimplified explanation for this cooler by saying it's a freezer inside of a freezer inside of a freezer.

As you can imagine, this cooling process takes time. Plug in a new refrigerator in your kitchen and it takes hours to get down to the proper temperature. Astronomical Observa-

tion sites tend to be rented out to various groups, making observation time precious and expensive. If the temperature of ZEUS2 is wrong, data can't be collected. ZEUS2 is also calibrated such that collected data is dependent on a very small window of acceptable temperatures. This means that ZEUS2 must be kept cold but not too cold, or data collection must cease until those ideal conditions are met. This is why ZEUS2 needs housekeeping equipment to control the cooler.

D. The Problem

ZEUS2 already has equipment that accomplishes this task. It's expensive, relatively old, proprietary, and specific to tasks that fall somewhat outside the scope of ZEUS2. The equipment works, but it's far from perfect. What happens when the equipment breaks? Originally, you'd have to send the equipment into the manufacturer for repair. This takes a lot of time and costs quite a bit. To make matters worse, what do you do when the manufacturer stops supporting your equipment, or goes out of business all together? You can try to fix the device yourself, but the designs are proprietary so you'll have to do a good bit of reverse engineering, and you risk breaking the equipment worse. This also takes even more time. You can buy an entirely new device that will serve the same purpose, but you'll eventually run into the same problems with that equipment, and it's really expensive.

Along with this problem, the existing equipment was not designed for ZEUS2. It isn't even one piece equipment, but multiple separate units working together. It functions well enough for the task at hand but it's not the most efficient about it. The equipment takes up a lot of space for features that don't even apply to ZEUS2, and some features rely on workaround and hacks that are inconvenient. A result of this inefficiency is that sometimes ZEUS2 is overcooled, and operators have to wait for the device to warm back up to temp. This can waste precious observation time.

II. THE SOLUTION

This is where the H.aHKBox comes in. The goal is design a single unit that can replace the existing equipment, that can not only be self repaired easily, but also leaves room for future additions. It can function not only as a the temperature monitor, but also as the

cooler controller, among other things. It is a modular systems such that every functional unit can be developed as a plug-n-play circuit board. I like to think of this like plugging a Graphics Card into the Motherboard of a desktop computer. These cards can communicate together as necessary, get power from the Motherboard, and each have their own external connections that allow for hookups to external sensors and devices. This project is Open Source, to aid in future repairs, and increase implementation and design. The project is broken into 4 physical design units listed in the subsections below.

A. 4-wire AC Bridge

Unknown Daughtercard.

B. 4-wire and 2-wire DC Bridge

Unknown Daughtercard.

C. PID Controller

This is a daughtercard to function as a PID (Proportional-Integral-Derivative) Controller. This is a common control unit for systems that need to accurately and precisely respond to incoming data in a small amount of time. The Cruise Control of a car is an example of this. It takes in the measured the speed of the vehicle, compares this input with the user input desired speed, and outputs in a way that puts mechanical response on the engine.

This particular PID Controller Card will function as the temperature controller. It will read in from temperature sensors connected to ZEUS2, compare those temperatures to the desired values, and output to the cooler such that the proper temperatures are maintained.

D. Chassis

This is foundation of the unit. It consists of the case itself, power supplies, cooling fan, touch screen, motherboard, micro-controllers (Arduino Uno, Raspberry Pi), and port panel. The chassis does not perform any functions without cards installed. It's not a perfect analogy, but you can imagine a desktop computer without a cpu or gpu. It is the foundation

for the device, but it is relatively useless alone. This is where all official design time has been invested so far.

E. P-Card

Since the daughtercards need to be designed and tested but cost a small fortune to manufacture, I designed a prototyping card. It can plug into the motherboard, and uses a female connector along with banana jacks to allow for prototyping circuits on the connected breadboard.

F. Knowledge "Requirements"

Astronomy is cool! Carl knows it and he'll do everything to help you understand it too! Don't ever shy away from learning something new, astronomy related or not. That said, this project tends to fall under mechanical, electrical, and software engineering, with significant standing in control theory. I mean this with lots of love: this project is essentially a glorified thermostat. To do it right you're going to need to understand the basics of how astronomy is done, but you don't need an astronomy degree. You'll learn a lot more than you need to, and not just in astronomy, which is really cool! Yes, you are here to work on this project, but you're also here to learn and have fun! If something peaks your interest and it will help you accomplish a task, spend some time to learn more about it.

III. WHAT WE'VE ACCOMPLISHED

A. Case

The case to be used was picked out quite a while ago, but I don't remember exactly which one. There should be documentation in OSF regarding the case selection. The 3D renders are where I'd start looking.

B. Power Supplies

After a lot research and consideration regarding the desired capabilities of the H.aHKBox, we decided to have two separate power supplies: One 24V 350W, One 5V 110W. The goals here is that each card would run around 10W max, but one or two cards could run at a higher power. Just had the idea of adding a ammeter to each supply which could report to the screen.

C. Touch Screen

We chose a touchscreen designed specifically for the Raspberry Pi. It allows for the Raspberry Pi to be mounted to the back of it which helps efficiently use the space inside the case. It's 7" and uses HDMI and USB to connect to the Pi.

D. Motherboard

With 10 card slots, 3 temperature sensors, arduino hookup, 7-segment display output, and connectors for data and power, the electrical design for this board is well under way.

E. P-Card

The design for this card is finished, but translating that into electrical designs that can be send to a manufacturer have proven difficult. I'm very close to being finished with it though.

F. Microcontrollers

We did choose the Arduino Uno and Raspberry Pi Model 3 B+ to act as the brains of the Motherboard. They're well documented and have the communication features I sought.

IV. WHAT STILL NEEDS TO BE DONE

A. Card Design

The Bridge Cards mentioned above have not been started on as far as I'm aware. Designs for the Motherboard are fleshed out enough at this point that someone could begin working on those cards.

B. P-Card

The P-card design is so close to being done. Hopefully within the first week or two work this summer we'll be able to order the cards from a manufacturer.

C. Motherboard

Designs of the Motherboard still need to be finished. The limited space has already proven to be a challenge, but it shouldn't be too big of a problem. Running traces for this board is going to be a real challenge, but it should be fun too.

D. Programming

Once the Motherboard designs are finished the microcontrollers can be programmed. I've been looking forward to this since I started working on this project. It'll be a fun balancing act of low level communication protocols.

E. Port Panel

External devices need to be able to connect to the H.aHKBox and this panel will act as a the connection converts so that wires are able to plug in. This is probably the last thing that will happen since its design depends on not only all the other designs but also on what external devices will be used and what connectors they use.

V. COMMON TOOLS

The tools used in this project are to some degree personal preference. If it works, that's good enough. That said, some things make a lot more sense than alternatives even if you have to learn a new program. This project is a fantastic opportunity to learn new softwares, especially since the university gives students access to professional softwares that are desirable to add to your resume. One of my coworkers last year learned SolidWorks from scratch and is now comfortable enough to add it to her resume. I myself knew very little about digital circuit design, but now I'm relatively fluent with CircuitMaker. Below I'll list the software I used and what I used it for, but you can always make a case for something else.

A. CircuitMaker

I spent most of my time in this software. It is touchy and has a lot of bugs, but it's free which is nice. it's also quite similar to the professional software Altium which is used by many companies. I'm sure the knowledge and experience I've gained on this software would carry over to Altium quite well. My biggest suggestion here is to use a mouse, not a trackpad, whenever possible. The zoom functions don't work properly with a trackpad, so I pretty much always had a USB mouse with me, or used my Desktop PC. Also, this software is not that lightweight. It's not easy to run and the more complex your project, the slower everything goes. All the more reason why I used my Desktop whenever possible. The WSU issued laptops can do it, but only barely.

I also found the tutorials to be outdated as the interface has changed since the tutorials were developed. The Altium forums were extremely helpful in debugging things as most problems I ran into someone else had run into before me. Even still there were some bugs that I just never did figure out. Some went away on their own, others didn't. When in doubt, restart the application. Save your work as often as you can. It's easy to get lost in the flow of work and then have the program crash and you just wasted two hours. I'm not proud but that happened to me more than once.

B. OSF

Open Science Foundation is where we document our work. It provides a centralized location for all documents regarding Hardware.astronomy projects. It has automatic backups, funding for like a hundred years so our data will never get lost to lack of funding, and allows us to open our research up to the public when we're done. You can upload files and type up your own pages with pictures and the like. You can use plain text typing or a simplified version of LaTeX when typing directly in OSF. This is where you're going to find all of the previously done research. It tends to be a little slow to load, which is a pain, but all in all it's not bad. Right now, the OSF page for the H.aHKBox is really unorganized and it's hard to find what you need. I'll probably spend some time really soon cleaning everything up.

You also have a personal page for file uploads and documents typing. I used this a bit for planning the project, like a to-do list for the day/week and solidifying my long term goals with dates. Mostly, though, I wrote daily reports. I'd start the day by typing out a short paragraph about what I wanted to accomplish that day. Then I'd end the day by typing out what I did and how it related to that goal. I wasn't directly tracking my "success" at staying on task, but putting it into words certainly helped me get started each day, and decide what to do the next day.

C. SolidWorks

I didn't have to use SolidWorks that much because I wasn't doing any mechanical design. I'm also not an expert at using the software. I have used Blender, Maya, and 3DS before and I can say with confidence that SolidWorks is easier to use, especially when it comes to mechanical design like this. The tutorials I followed were pretty good, and what few problems I ever had were easily solved with a google search. Again, I suggest a mouse over a trackpad.

D. Git and Github

If you've never used github, this could be a fun excuse to do so. If you have any intention of working in software in the future I strongly suggest you learn how to use it. There are a lot of tutorials out there and a lot of different ways to do it. I prefer using git for my file

hosting over OSF just because it's so much faster for me to do that. I even use github for all of my school stuff like homework. It allows me to easily transition from laptop to desktop and back.

I use the commandline git tool, not the GUI, so I can't help much if you use the GUI. I haven't used git with multiple people on the same project in the better part of a decade so if you are interested in that, it'll be a learning experience for both of us. I know I need to figure out how to do that soon enough anyway. I am currently the owner of the git repository for this project, so if you have any interest in using it too just let me know and I'll look into a better setup for multiple people.

Github does link into OSF really well so if you don't have any interest in git, you can still access all of the files I put there through OSF. I update git multiple times a day but tend to only transition it to OSF itself when I hit major checkpoints. If you need my most up to date version of something that I've been working on very recently, you should probably check git first unless you specifically ask for me to update the OSF version.

E. Whiteboard

I know this may be meaningless with COVID19 keeping everyone inside, but a large whiteboard is so so helpful. I'm sure paper sketching is a fine substitute but after working on this project I decided I'm definitely getting a whiteboard for my room.

F. Coworkers

We all know different stuff to different degrees. We can all lean on each other and it only increases the fun to share knowledge. The only thing I love more than teaching others is learning from others. You need help with something, just ask. If not me someone else. If I don't know the answer I'll likely help you find it.

G. Email

Emails aren't a bad way to communicate, especially if someone is currently unavailable by other means. I check mine regularly and you should do the same.

H. Microsoft Teams

I've never used Teams before. Last summer we used Slack but it wasn't the most cooperative. Hopefully Teams will be a little more seamless.

I. Discord

I use discord for a lot of different things so I pretty much always have it going if I'm at my computer. I'll be making a discord channel specifically for this research group once we get started and it's my preferred method of contact. Instant messages are great if it's a simple question and voice calls are a good backup for more in depth conversations.