

FunBox Classic (FBC)



**Senior Design I - Project Documentation
April 30, 2015**

Group 14

**Stephen Caskey
Nick Johnson**

**Anna Iskender
Kyle McCleary**

Contents

1. Executive Summary.....	1
2. Project Description.....	2
2.1 Project Motivation	2
2.2 Goals and Objectives	2
2.3 Requirement Specifications	3
2.4 Standards and Constraints	4
2.4.1 Standards	4
2.4.2 Constraints	6
3. Research Related to Project.....	11
3.1 Existing Similar Projects and Designs	11
3.1.1 Instructables How to Make a Portable Game System by 1up	11
3.1.2 Adafruit PiGRRL	12
3.1.3 The eNcade.....	13
3.2 Hardware Research	14
3.2.1 System Processor	14
3.2.2 Screen	19
3.2.3 Microcontrollers	25
3.2.4 Communication Technologies	29
3.2.5 LEDs.....	41
3.2.6 Solar Paneling	46
3.2.7 Audio	52
3.2.8 Power System	59
3.2.9 Portable Console Design.....	75
3.3 Software Research	84
3.3.1 Base Operating System	84
3.3.2 Software Frontend.....	86
3.3.3 Software Backend	87
3.3.4 Operating System Modifications.....	90
4. Hardware Design.....	92
4.1 Screen Setup	92
4.1.1 Backlight Controller	93
4.2 Audio.....	94
4.3 Power System.....	97

4.3.1	Wall Charging Circuit Design	97
4.3.2	DC-to-DC Converter Circuit Design	98
4.3.3	Power Supply Design	99
4.3.4	Combining the Power Supply and Charge Circuit	101
4.3.5	Solar Panel Charge Controller Circuit	103
4.3.6	LED Battery Charge Indicator	103
4.3.7	Switching Battery Chargers	107
5.	Prototype Construction	109
5.1	Hardware Acquisition	109
5.2	Hardware Overview	110
5.3	Hardware Integration	111
5.3.1	PCB	112
5.4	Software Overview	113
5.4.1	Software Acquisition	114
5.4.2	Software Integration	115
6.	Prototype Testing	115
6.1	Hardware Testing.....	115
6.1.1	Raspberry Pi 2.....	115
6.1.2	Screen	116
6.1.3	Wall Charge Module	116
6.1.4	Battery	117
6.1.5	Solar Charge Module	118
6.1.6	Power Supply Module	119
6.1.7	Battery Indicator Module	120
6.1.8	Backlight Controller Module	120
6.1.9	Bluetooth Module	120
6.1.10	Controller Module	120
6.1.11	Speakers	120
6.1.12	Audio Module	121
6.2	Software Testing	121
6.2.1	Emulator Tests	121
6.3	Final Integrated System Tests	126
6.3.1	Integration Tests.....	126
6.3.2	Final System Tests.....	127

7. Administrative Content	129
7.1 Milestones	129
7.2 Workload Distribution	130
7.3 Budget and Finances	131
8. Conclusion	133
Appendices	134
Appendix A – Copyright Permissions	134

1. Executive Summary

The following documentation will outline, in full detail, the processes, procedures, and means by which we will complete our project: The FunBox Classic. It will list and describe the goals, objectives, specifications, research, designs, testing procedures and administrative content of the project. The final FunBox Classic prototype will bring back memories of a simpler time, when games did not have gigabytes of resources and content to work through. It will act as a simple, but pleasing, way to play classic Nintendo games. The project will not attempt to break new ground, but it will endeavor to make it as simple as possible for any user, even without advanced technical knowledge, to just pick up and play, wherever they might be.

The end goal of our project is to produce a functioning prototype. This will conform to a variety of specifications. The device will operate at a resolution of 640 x 480 on a 3.5-inch screen. The buttons for the internal and external controllers will be placed at optimum positions, ensuring proper ergonomics. The device will fit comfortably in the hands and weigh little, allowing for hours of gameplay without fatigue. It will be able to connect with a wide variety of external controllers via Bluetooth, including one custom made for the system, allowing for multiplayer on the device. The device will charge using a standard Micro USB port, albeit with a wall charger instead of through a computer. Note that it will still charge through a computer, but much more slowly. The device will also charge through solar panels on the back of the case. This will allow for a little more power to get a user through the day. The casing will be custom designed and 3D printed, ensuring the tightest fit for the internal components.

There were many ways to meet the requirements set out in the requirements, but it was decided to use trusted, proven components and technologies, to maximize efficiency and minimize wasted time and money. A Raspberry Pi 2 was selected as the core of the device, allowing the FBC to run a full Linux operating system for maximum software compatibility. Bluetooth 4.0 LE was selected to ensure the fastest speeds at the lowest power cost. Composite video was chosen to output to the screen of the device, which helped to ensure compatibility with the Raspberry Pi without adding on the additional hardware costs of HDMI. The PCB and buttons from a SNES controller were reused for the internal controller, ensuring that classic responsiveness and feel. The FBC will have a hardware battery indicator on its casing, allowing the user to be notified when the battery is running low and needs to be recharged. Hardware volume control will also be on the case, allowing the user to control the volume level even when the device is turned off or unresponsive. By combining these technologies together, the FunBox Classic is sure to delight and satisfy gamers of all ages.

2. Project Description

The FunBox Classic (FBC) is a portable game console that emulates five Nintendo systems: GameBoy (GB), GameBoy Color (GBC), GameBoy Advance (GBA), Nintendo (NES), and Super Nintendo (SNES). The main workhorse of the FBC is a Raspberry Pi 2 (RP2) which runs the operating system and software needed to support the emulators. Games can be loaded to the FBC by uploading them directly to the SD card in the Raspberry Pi 2 or using a portable storage device in the USB port on the case. A screen is attached to the RP2 and will display the emulated games when the device is turned on. The FBC also contains speakers and a headphone jack to allow users to choose how they want to hear their games. A game controller is attached to the RP2 directly and seated internally in the case for a single user. If multiplayer is desired the FBC also supports Bluetooth controllers. The FBC will operate away from a wall plug for extended periods of time due to the internal rechargeable battery. The solar panels that charge the device when it is not plugged in extend the life of the FBC even further.

2.1 Project Motivation

Our motivation for building the FunBox Classic was mostly due to personal desires. We wanted to build a device that would allow us to play the old classic games we loved on the go in a single all-inclusive portable console that is comfortable to hold and use. We also wanted the device to last long enough to allow us to actually be able to enjoy the game on a road trip or flight. Additionally, a device such as this would allow each group member to obtain design experience in various categories. The combination of electrical circuit design components, hardware/software interaction, and software design components was perfect for our group, as well as the ability to add or remove components as needed or desired.

2.2 Goals and Objectives

Our main goal is to accurately recreate the feeling of playing our old favorite games, while also allowing the player to move around freely while doing so. Ideally, the FBC will replace all of a user's old Nintendo consoles while also adding new features to them.

There are a variety of goals for the FBC that are necessary to the operation of the system. The FBC will have a battery system that will quickly charge a battery from a standard USB connection. The FBC will have an on-off switch that will control the power from the battery to the FBC components. The battery system will also keep the FBC powered for extended periods of time. The FBC will be able to emulate GB, GBC, GBA, NES, and SNES games at their native speeds. Games will be uploaded to the FBC through the USB port on it. The emulated games will be displayed on a screen housed in the case of the FBC and attached

to the RP2. An audio system attached to the RP2 will play the sound from the games via speakers or headphones. A controller to operate the FBC will be attached to the Raspberry Pi and will be comfortable to use. The case will house all the components of the FBC safely. The case will be sturdy enough to survive a drop and will not allow the circuitry in the FBC to short or disconnect.

Additionally, there are a variety of supplementary goals for the FBC that are important to us. The FBC will have built-in Bluetooth to allow users to connect their Bluetooth controllers and keyboards to the FBC for multiplayer or if they would prefer to use that controller instead of the internal controller. The USB port on the FBC will be allow users to attach a USB controller to the FBC for the same reasoning as a Bluetooth controller. The FBC will also have solar panels that will extend the battery life of the device. The FBC will have a button that, when pressed, will illuminate an LED battery charge indicator. The FBC will also have LED indicators for charging, completed charging, battery/charge error, low battery, and power on.

2.3 Requirement Specifications

This list of requirement specifications will describe the minimum goals and objectives for our device in a more precise manner. Following the list we will explain our reasoning for our minimum requirement specifications.

- Screen Size of at least 3.5” but less than 6”
- Display rate of at least 50 FPS
- Bluetooth 4.0 LE or higher
- Flash Memory of at least 16 GB
- 3.5mm Headphone Jack
- 2 Speakers of at least 1 Watt at 8 Ohms
- Charging Voltage of 5V
- Operating Voltage of 5V
- Maximum system current draw of 700 mA
- Solar power charge current of at least 100 mA
- Battery of at least 2100 mAh

We chose the size of the screen to be between 3.5” and 6” for 3 main reasons. The first reason was to maintain the portability of the device. The larger the screen the less portable the FBC will become. The second reason was to keep the power draw of the system low. A larger screen meant shorter battery life. The third reason was to make the games look better. A smaller screen will result in less stretching and will not distort the low resolution games of our consoles. We chose the display rate to be at least 50 FPS to conform to both NTSC and PAL standards.

Bluetooth 4.0 LE was our minimum requirement due to our device being battery powered. The newer Bluetooth LE devices draw significantly less current and will allow for a longer lasting battery life.

The flash memory of the device needs to be at least 16 GB due to it needing to be able to contain an operating system, systems software, emulators, and games. While 16 GB may not be necessary, we want to allow users to hold all of their games on the device.

A 3.5mm headphone jack is required because it is the standard size for headphones and we want the user to be able to plug in any of their own headphones into the FBC. The speakers need to be at least 8 Ohm 1 Watt speakers to ensure that their output through the case is audible and clear.

The charging voltage needs to be 5 volts so that the device may be plugged directly into a standard USB port, which operates at 5 volts. The user needs to be able to plug the device into their phone charger or computer to charge it. The power supply of the system needs to supply an operating voltage of 5 volts because the Raspberry Pi 2 is normally powered by USB.

The current draw of the system should not exceed 700 mA because the device aims to be low power and long lasting. We settled on 700 mA as the cap because the only things drawing significant power in the device are the Raspberry Pi 2 and the screen. The RP2 should not exceed 400 mA and the screen should not exceed 200 mA. A 700 mA current draw maximum compensates for the 600 mA of the RP2 and screen and allows 100 mA for the rest of the components combined. The solar power charge current should be at least 100 mA to partially alleviate the current draw of the rest of the system and to charge the battery at an acceptable rate when the device is not powered.

The battery should have a charge capacity of at least 2100 mAh. At the max current draw of 700 mA a 2100 mAh battery would last for three hours without any solar power. We want the device to last for at least three hours when not in sunlight.

2.4 Standards and Constraints

There are a variety of standards we will adhere to in the design of our project, and constraints that will affect both the design and use of our project. There are standards for pretty much everything, but we are primarily concerned with the standards related to the most important components of our device. Likewise, there are a vast amount of constraints that will have some effect on our project, but we will focus on the ones that have a significant impact.

2.4.1 Standards

Engineering standards specify properties and technical requirements that must be met by systems that implement them. This ensures that minimum guidelines, such as safety, performance, reliability, testability, and interaction with other complying equipment, are met. The following are standards that are implemented either by our project or components in our project.

2.4.1.1 Bluetooth 4.0.

Bluetooth 4.0 is a version of the Bluetooth wireless technology standard for exchanging data over short distances. It uses UHF radio waves in the ISM band to accomplish this. Bluetooth 4.0 uses half the power of Bluetooth 3.0, which makes it crucial for the developing mobile market. It was formerly maintained by IEEE, but they abandoned support. It is now maintained by the Bluetooth Special Interest Group.

2.4.1.2 NTSC

NTSC is the analog television system used mainly in the Americas. It can be transported in a variety of ways, such as coaxial cable or composite video cable. It was developed and standardized by the United States Department of Defense, as seen in document # SMPTE-170M: Television - Composite Analog Video Signal – NTSC for Studio Applications.

2.4.1.3 USB 2.0

USB 2.0 is the second main revision of the Universal Serial Bus standard. This increased theoretical speeds to 480 Mbit/s, 40 times faster than the 1.x standard. It has four pins: Vcc, Ground, Data+, and Data-. The specification was developed and maintained by the International Electrotechnical Commission (IEC) in document # IEC 62680-1: Universal serial bus interfaces for data and power – Part 1: Universal serial bus specification, revision 2.0.

2.4.1.4 Micro USB Revision 1.01

Micro-USB revision 1.01 was a new, smaller connection designed for USB 2.0. Micro-USB adds an ID pin to the previously existing four pins of USB. This allows for USB on-the-go technology, which turns a client device into a host. The specification was developed and maintained by the International Electrotechnical Commission (IEC) in document # IEC 62680-2: Universal serial bus interfaces for data and power - Part 2: Universal serial bus - Micro-USB cables and connectors specification, revision 1.01.

2.4.1.5 FAT32

FAT32 is a variation on the 1970s file system File Allocation Table, or FAT. The 32 stands for the size of the entries in the filetable which are each 32 bits. This

increased the maximum volume size to 2TB, 1000 times more than the previous FAT16 (these both assume 512 byte sectors). It was developed, standardized, and maintained by Microsoft in the Hardware White Paper: "Microsoft Extensible Firmware Initiative FAT32 File System Specification. FAT: General Overview of On-Disk Format. Version 1.03."

2.4.2 Constraints

Design and construct considerations for the FBC must also take into accounts of the likelihood of realistic developmental impact constraints that will need to be addressed. Discussion of potential, if not already existing, implications will help identify legitimate concerns, as well as benign factors that do not pose actual issues in the execution of the project.

2.4.2.1 Economic Constraints

Economic and financial factors will be the most heavily focused constraints to anticipate moving forward with the project cycle. Given the many various components that comprise both the hardware and software aspects of the FBC, hurdles to consider include individual acquisition of building parts and materials, and the availability based on quantity need. One specific constraint would be the necessitation of individual components that do not require mass ordering. Certain FBC features utilize unique IC's and microcontrollers that need no more than one or two centralized components for implementations. Ordering such parts in singular quantity can conflict with manufacturer requirements that dictate quantity bulk order, and potentially limiting the availability to particular models. Another aspect to consider is the financial limitations of ordering components on the individual level as opposed to mass ordering. Bulk purchases generally tend to be at a better pricing figure than simply ordering one component, and having that component alone shipped for manufacturing. An evident challenge is that many of the FBC controllers and IC's are unique from each other and cannot be substituted as a common factor piece. Therefore, the need for many different singular parts arises with the dilemma of specialized orders.

Additionally, an unforeseen economic constraint that was faced in the developmental phase of the FBC was in the form of reshaping in the technological consumer market. In particular, the standard for commercially sold electronic components, Radio Shack Corp., was forced into bankruptcy, thus resulting in multiple store locations closing. This factored into our ability to readily obtain necessary components for design and construction of the FBC, as well as generated refocus onto additional reliance in online ordering conventions.

Adding to ascertaining of parts, and their availability, another constraint in the economic sense is the means to build the FBC. Self-reliance on sectional building would require additional resources, such as soldering stations and specialized tools, like small-scale mounting instruments. Another option would be

resourcing private venues that professionally construct the designated circuit boards needed, which would be another unanticipated fee. This must be factored into additional costs that do not directly go into the FBC, but must be used for achieving its realization.

2.4.2.2 Environmental Constraints

Availability of resources and surround requirements are to be taken into account when dealing with the FBC environmental constraints. Primarily, the conditions of building the device are to be considered first, with access to the proper work environment being essential. For testing and building of parts, we have a home-based work-bench that features amenities ideal for electrical-mechanical construction. However, the limitations facing our work-space include fundamentally scaled utilities that do not necessarily parallel to those of standardized research labs. The Senior Design lab located on UCF is another option that could provide a desired environment for project advancement, but would also require a constant transportation of parts, many of which are on the micro scale, to and from said location. Additionally, the Senior Design lab also has limited space and readily accessible utilities that can compromise work efficiency. Given that so many of the FBC building components are only a few millimeters in surface area, fabrication conditions should be favorable to an abundance of lighting and sizable work-surface area.

The secondary environmental constraint deals with the functionality of the FBC within its surroundings. This constraint directly refers to the operation of the FBC's battery charging solar panel circuit. Essentially, this component harnesses available sunlight into usable electrical energy to charge the internal source lithium polymer battery. For this to occur, operation and temporary storage of the device should be in a location accessible to direct sunlight. This can be a challenge, in particular, for in-use operation, as the solar cells capturing sunlight are located on the back of the FBC, opposite facing from the TFT LCD screen. Due to this design specification, the solar cells are not likely to be directly exposed to environmental sunlight while the FBC is in use. However, the solar cells are capable of operation in indirect lighting, and full lighting can be achieved should the FBC not be in user operation, and is left, screen-side down, for additional battery charging. It is noted that the solar charging panel is the only natural-environment dependent aspect to the project.

2.4.2.3 Social Constraints

The identification of social constraints pertaining to the FBC project can be met with a subjective paradigm, considering the views society can hold on the roles of videogames in the social environment, let alone the advancement of a previously outdated console. The influencing factor of videogames as a household common component make the further advancement, such as the FBC, questionable under what society approves. In particular, social views that videogames can be a

debilitating, and even destructive, factor to human behavior has been an argument used for most of the history of gaming consoles. And while indeed the discussion zeros in on the actual games used for the consoles, the FBC provides a vehicle by means to potentially deliver such disapproved games in the form of a portable handheld device. This brings up the social constraint of whether it is marketable to make readily available a device that can yield more potentially violent gaming platforms.

Additionally, the other social constraint to be discussed is the social reaction to recreating a console that was very popular at its release point in the early 1990's, and giving it new life under modern amenities. Society, to a scale, may have objections to hosting a compilation of emulated games on a private, unique gaming device. Additionally, modeling a device after such an iconic console can make it a tall task to live up to certain reviews if compared to its professionally manufactured predecessor. Although social constraints do not have a direct, immediate constriction to the design and building of the FBC, it sets conscious limitations going forward as to the overall design consideration, as well as final product reception.

2.4.2.4 Health and Safety Constraints

Concern for conditions of operation and safety quality of the product bring are considerations behind the health and safety constraints. While the FBC is a light, portable, low-powered device, potential hazards are to be assumed worth addressing. An immediate concern would be associated with the safeguards behind the device, and the minimization of malfunction risks. The FBC is powered by a lithium polymer battery, rated at 4.2 V, and the overall design of the FBC compacts all components, including the battery, into a close-quarters case. Lithium batteries, in of themselves, can be susceptible to catastrophic failures as a result from overcharging, with fires and explosions in extreme cases. Although the battery we are using comes with overcharge protection, and the FBC features IC's to address the potential for battery overcharge, there is a small, but significant, risk of device failure that can pose as a safety hazard. This is especially relevant given that the FBC poses a dual-option charging system, and should a battery indication system fail, overcharge can be a realistic concern. Additionally, with the battery in close proximity to the rest of the hardware components, self-destructive behavior will likely damage the device as a whole, and even possibly a user while in operation. Such occurrences are unlikely, to say the least, but a disclaimer to the internal power source must also be brought to light.

Another safety constraint comes with the FBC construction process. The device is to be built utilizing multiple electronic and mechanical means for project realization, and a majority of work being conducted by amateur craftsmanship. Unfamiliarity with production tools, such as soldering guns, can lead to burn or shock injuries, as well as other physical injuries from hardware connections and case building. Proper safety procedures and protection apparel will need to be

observed to avoid such risks. Furthermore, the device will be attempted to be assembled in a simulated professional manner, but will not be factory-grade work. Such results can mean potentially exposed electrical components that can pose isolated shock risks. Consideration for working on the device when connected to a power source should be taken to minimize risk of electrocution.

A health concern with the FBC comes from device operation by the user, especially the environment that pertains to the given situation. The console uses a 3.5" TFT LCD screen for visual transmission output of the video games being emulated for use. Although suitable for a portable device, the size limits the user to a small focus point for potentially extended periods of time. This may lead to the user bringing the illuminated screen into closer proximity to their eyes for better visual prowess. This can potentially lead to eye strain problems, and in isolated cases, myopia, that may require attention. Additionally, device operations in a darkened environment can lead to headaches and eye fatigue.

2.4.2.5 Manufacturability Constraints

The building process, along with the means to build of the project, yield manufacturability constraints to be aware of. A specific constraint encompasses the acquisition of parts and components for the FBC. Nearly all of the electrical components are diverse in origin, and require multiple shipping orders from various companies to achieve the materials needs. Also, bulk ordering is unnecessary, due to demand for mostly singular components. This makes the purchasing of parts a constraint to the production process, as locating the necessary parts can be tedious and diverse. Another manufacturability constraint to observe comes from the actual building of the device. The ideal design for the FBC specifies a centralized device that features components surface mounted to the printed circuit board. The two problems with this come from actual achievement of surface mount soldering, and actual restrictions given tight spacing. Surface mounting components, which are only a few millimeters in dimensions, makes for tedious and limited manufacturing, even with the use of specialized low-scale and magnification tools. The actual components and models to be surface mounted are also constrained to what can be achieved under design specifications. Certain parts cannot be directly surface mounted to the location they need to be in due to either internal temperature specifications, or surrounding component conflicts. This limits the manufacturing process by hand, producing the need for production solutions that are achievable on the small, individual scale, given the FBC's manufacturing is done on the private, preliminary platform.

2.4.2.6 Sustainability Constraints

Energy sustainability, both within the device system and the surrounding environment, provides constraints for consideration. Primary restrictions for device energy allocation constitute as battery supply throughout the system, as well as both means of charging the battery, from a plug-in wall charger, and from

attached solar panels. Parts sustainability must also be addressed, with the prospect of continual availability of components and specified models crucial for intended production. Power sustainability within the device is first observed, with the lithium polymer battery supply throughout the FBC being scrutinized for sustaining the necessary power to run all components. Various features of the console, from LCD screen display, to audio speaker output, and auxiliary controller connection all require diverse powering demands that must be met from the source battery. Sustainable power must be achieved equivalently for optimized performance, with proper voltage regulation and DC-to-DC conversion observed as needed.

Additionally, power sustainability constraints include means of charging the FBC's source battery. Two means of charging, the wall charger and the solar panel, are able to provide additional power to the system, but must be done so in a sustainable manner. The wall charger must utilize the given available power from whatever plug-in source while only supplying as much as needed to the FBC. Excessive power draw from the charging source will be wasteful and unnecessary, as well as potentially hazardous. Thus, the wall charger needs to draw current and voltage at an acceptable rating and have means to prevent over-draw. The solar panel charging circuit utilizes renewable energy for additional battery power, and must actually sustain a measureable amount of energy for significant charge. The sustainability constraint with the solar panel contribution comes from the concern of necessary photonic sources needed to supply the desired current for the system.

Sustainability of parts used within the project is also a constraint to recognize. Preliminary planning of parts to be used for the FBC can be compromised should the components not be sustainably produced or marketed. To date, there is no current concern that specific model components are being produced at an unsustainable rate that would produce foreseeable concern in the future. However, constraints in market demand and availability bring to question the sustainable course of anticipation for achieving of parts at a later date for production.

3. Research Related to Project

Making old video game consoles portable is something that hobbyists and video game enthusiasts have done for years. Researching existing projects that are similar to our project will allow us to get a better understanding of what exactly we need to do to make our system work, as well as warn us of potential issues that may arise during the design of our project. Additionally, research into various hardware components will allow us to select the components that best fit the needs of our device. Choosing appropriate components for our device is imperative in order to ensure low power consumption, reactive controls, effective charging and solar charging, and accurate rendering of games. Research into software for the FBC will allow us to determine the best way to emulate games and interact with user input.

3.1 Existing Similar Projects and Designs

There are quite a few projects that are similar to our device. Some of the projects choose to transplant an old console's circuit board into a portable device and natively play its games, while other projects choose to use computers to emulate multiple consoles and store the games on the device itself. We chose to examine three similar projects. Of these, one uses a disassembled console's circuit board and two use microcomputers to emulate a variety of systems. The two latter projects more closely relate to our device, but the first project contains some useful information as well. Sadly, we could find no projects similar to ours that supported Bluetooth or had additional battery charging via solar panels.

3.1.1 Instructables How to Make a Portable Game System by 1up

This project, shown in Figure 3.1.1.1, contains a lot of valuable information related to building our project. It explains the main differences between a few common types of batteries and how to choose between them. This project also shows how to take apart a screen and alter it to consume less power if necessary. Information on how to make a custom controller for a NES console and also how to use the PCB from inside a NES controller. The tutorial for this project also contains a lot of basic wiring and soldering advice that will prove quite useful in the construction of our project. It also explains a variety of ways to design and construct a case. This information is invaluable to us because none of us have any experience with case design.

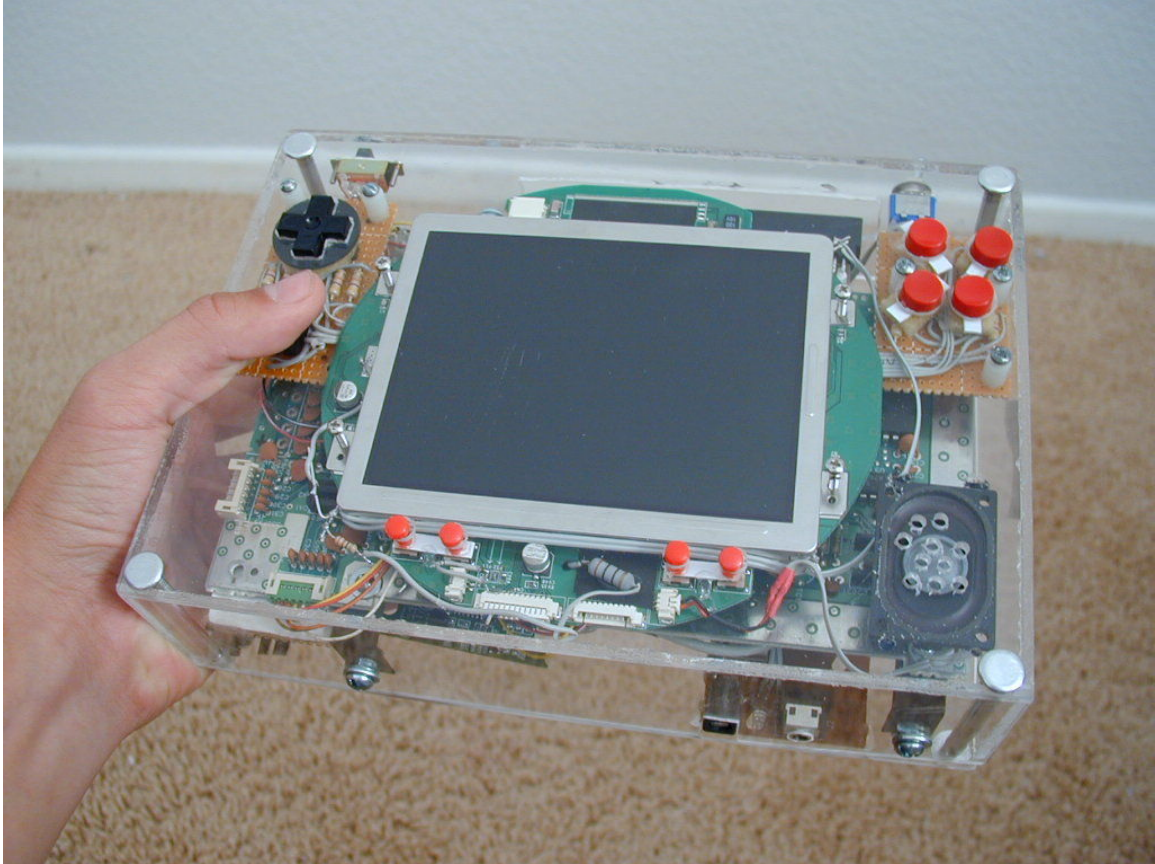


Figure 3.1.1.1 – Portable NES by 1up (Permission Pending)

3.1.2 Adafruit PiGRRRL

The PiGRRRL, shown in Figure 3.1.2.1, is a project that is very similar to the device we aim to make. It effectively emulates the consoles that we want to emulate. The PiGRRRL also has a small screen (2.8”) and operates using batteries. The device also charges via USB. It is lightweight and portable and has a controller built into the case. While the PiGRRRL does use multiple premade circuits that we will not use, such as a charger and DC-to-DC converter combo circuit, it does help to give us a basic understanding of what kind of components we will need. The PiGRRRL also has downloadable files for 3d printing. Examining these files will give us a much better understanding of how 3d modeling and 3d printing work. This project also gives us quite a bit of information on how to setup emulators on a Raspberry Pi. This information could prove to be invaluable. This project also gives even further insight into modifying old console controllers and using them for other purposes.

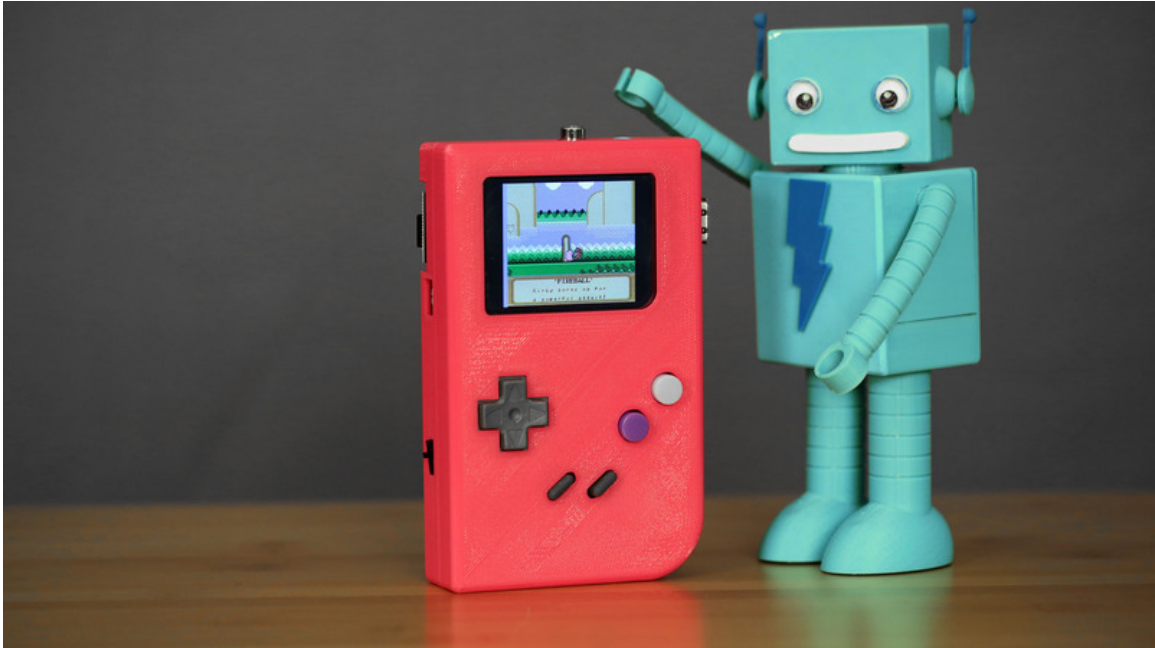


Figure 3.1.2.1 – Adafruit PiGRRL

3.1.3 The eNcade

The eNcade, shown in Figure 3.1.3.1, is another portable console that uses the Raspberry Pi. It has already raised \$6629 on Kickstarter, which shows that there is at least a small market of people interested in a project like this. The eNcade is also very similar to the device we would like to make. Like our device, it is battery powered and has a custom case, onboard controller, built in screen, and a USB port. The software of the eNcade allows users to connect and play games with each other online. The eNcade, like the PiGRRL, uses many pre-built circuits instead of designing circuits around components, but they are connected in a very similar fashion as the PiGRRL. This was a great discovery for us because it shows how we would need to connect the components in our device and what kind of circuits we would need to design.



Figure 3.1.3.1 – The eNcade (Permission Pending)

3.2 Hardware Research

Choosing the hardware that will make up the device is extremely important to the overall design and construction of the project. The components that make up the project will dictate how they interconnect and the circuits that need to be built around them. The research in this section will compare different hardware components and choose the best parts for our project.

3.2.1 System Processor

When deciding what would be the core of our project, we took many factors into consideration including: cost, resources available, size, extensibility, and developer support.

We mainly compared three types of devices: Microcontrollers, FPGAs, and micro-computers such as the Raspberry Pi and BeagleBone Black.

3.2.1.1 Microcontrollers

Table 3.2.1.1 shows a comparison of the different microcontrollers that we considered. The cost was a very appealing factor. Very cheap and very easy to get ahold of, microcontrollers gave us a lot of leeway when budgeting. For the size, the microcontroller boards were, of course, small. This meant that we could fit it in a smaller form factor case, leaving more room for other sub components. As far as extensibility went, it depended on the microcontroller board. Overall, there were some boards with a multitude of expansion capabilities and others with very little.

We felt confident we could find an extensible enough board to suit our needs. Developer support on microcontrollers varied, depending on the popularity of the

chip. No one, however, had tried anything similar to what we were doing on a microcontroller, and this was a large negative.

Processor speed maxed out at 500 MHz, enough to run the emulation code, but not quickly, due to the extra load of interpreting every line of assembly code and translating it to the new architecture. Further, the RAM maxed out around 256 KB, not enough to emulate the 256 KB of the GBA, or the 128KB of the SNES, due to emulator and microcode overhead, as well as the Video RAM requirements of each system.

The processors themselves maxed out at 32-bit, which worked, but that was the absolute minimum a processor could have to process the emulation code. This is due to the fact that the Game Boy Advance had a 32-bit processor, thus a 16-bit or 8-bit processor would not have been able to run its code.

Furthermore, and perhaps more pressingly, the microcontrollers could only hold up to around 256KB of program code maximum, and that was on the higher performance chips. This is definitely too small for even the smallest of our emulators (weighing in at 2.1 MB), let alone the crazy task of fitting an operating system capable of managing these emulators and their filesystems into that small space. This, we believe, is the reason that no one had tried what we were doing before: it was unfeasible with the resources available. For this reason, we were forced to no longer consider microcontrollers.

Company	Model Number	Price	Core Size	Memory	FLASH	Clock Speed
Atmel	ATSAMA5D36A	\$17.53	32-bit	128 KB	160 KB	536 MHz
Atmel	ATSAM4N8AA-AURTR-ND	\$6.08	32-bit	64KB	512KB	100 MHz
Microchip	PIC32MZ2048ECM144-I/PL	\$16.03	32-bit	512KB	2MB	200 MHz
Texas Instruments	MSP430F5522IZQE	\$7.04	16-bit	10KB	32KB	25 MHz
Texas Instruments	TMX320F28377SPTPT	\$32.98	32-bit	82KB	1MB	200 MHz

Table 3.2.1.1 - Microcontroller Comparison

3.2.1.2 FPGAs

The cost was much higher, making it less appealing in that category. Size depended on the capability of the development board, so it wasn't really an issue. Extensibility was much better. Not only were there various expansion ports on the board, but just by the nature of an FPGA, the software capabilities were greatly expanded. This was a highly desirable property. Resources available

were also greater than the microcontroller, so we thought we had a winner this time. Sadly, when we got to developer support, we were forced to abandon this idea. Another group of graduating students had tried this project in the past and it was left unfinished due to “unsolvable complexity” in the project. The FPGA just could not adequately support the strenuous requirements (often 10 times more than original specifications) of emulation, and so they hung what was equivalent to a “Abandon all Faith” sign in front of their project, as a warning to others to not go down that path.

We would have to find some way of programming in all the various processors of the consoles, a task which was impossible given time constraints. As evidence, it took a student a year to fully implement just the NES processor (a relatively simple 8-bit Ricoh 2A03 chip) for his master’s thesis. As we had five systems, with five different CPUs (and supporting systems such as GPUs), we realized that this path would be unfeasible given the time allotted for the project. As a result, we had to abandon the idea of an FPGA.

3.2.1.3 Micro-Computers

Table 3.2.1.2 compares essential features between the two micro-computers that we primarily considered. The cost was relatively decent, costing between \$35-\$55, midway between the others. The size was fairly consistent as well, with the larger Raspberry Pi still only about the size of a credit card, albeit a very bulky one. Extensibility is massive, with a 40 pin GPIO header on the Raspberry Pi 2 and 65 pins on the BeagleBone Black. Developer support for both options were fantastic, with tons of projects similar to ours already fully completed and documented. Resources available were also large, with a quad core 1 GHz processor and 512 MB of RAM on the Raspberry Pi 2 and a 1GHz processor, 512 MB of RAM, and 4 GB of flash storage on the BeagleBone Black.

	CPU Type	Clock Speed	Cores	RAM	Onboard Flash	Native Audio
Raspberry Pi 2	ARMv7	900 MHz	4	1 GB	N/A	3.5mm or HDMI
BeagleBone Black	ARM Cortex-A8	1 GHz	1	512 MB	4GB	HDMI Only
	Native Video	USB Headers	Avg Power Draw	GPIO Pins	Cost	
Raspberry Pi 2	Composite or HDMI	4	650 mA @ 5V	40	\$39.99	
BeagleBone Black	HDMI Only	N/A	460 mA @ 5V	65	\$45	

Table 3.2.1.2 – Essential Features Comparison Chart

Both of these systems would work for our project, unlike the other options. Both were reasonably priced, with comparable features. We ended up choosing the Raspberry Pi 2 for the following reasons.

1. Parallelism is incredibly important in emulation. All graphics processing can be run in parallel, as well as a lot of the assembly interpretation and subsequent code optimization. For this reason, having the 4 cores of the Raspberry Pi 2 was a must, compared to the single core of the BeagleBone Black.
2. While the 512MB of RAM of the BeagleBone was sufficient, we felt more comfortable with the 1GB of RAM of the Raspberry Pi 2. It made us feel confident that we will not run out of memory recreating these games.
3. Due to reasons of extensibility, the BeagleBone only having HDMI was a minus. If we decided to use it, we would be forced into using an HDMI monitor, plug and all. This would raise our costs and lower our options. Having composite as a fallback, especially given the naturally low resolutions and original sizes of these games, was key.
4. For audio, having HDMI only killed it. Not being able to use audio without splitting the HDMI signal would be a huge complication and inconvenience, greatly limiting our options for audio playback. We decided that the Raspberry Pi 2, with its easily replaceable audio jack, was the right choice for the job.
5. The lack of USB headers on the BeagleBone was a huge detriment. Even though we had planned to chop off the tall headers of the board regardless, we still planned to make use of USB connections via soldering wires, as transferring files to the system via USB would be essential to add the games and other bits of software at user convenience.
6. The Raspberry Pi 2 already had the excellent RetroPie backend, with EmulationStation frontend, ported to it perfectly. This would save us time and effort that could be focused on the hardware end of the project. Further, the software already looked very nice and works very well in our desired input configuration.

3.2.1.4 Raspberry Pi 2 Usage

We will be using the following features of the Raspberry Pi 2:

- Composite Video
- 3.5mm Audio
- USB
- GPIO Header
- Micro SD reader

Composite Video

Composite video will be used to output to a screen, as we have determined that this will be sufficient for our needs. We decided that HDMI would complicate

things and add cost, and that input solutions from the GPIO headers would be incomplete without driver overhauls. These are explained in greater detail further in the paper.

3.5mm Audio

The jack will be used for outputting the audio from the various games. We decided against using a custom DSP solution, as it would add complication without much benefit.

USB

The USB headers will be used to attach external USB ports not soldered to the board. The current USB headers are too tall, and we would rather reduce the footprint of the board in the center of the casing. We will use this external header to either load games onto the device using a USB drive, or to debug the system with an external keyboard.

Micro SD

The reader will be used to read the MicroSD card (64GB) that holds our operating system, our emulators, and all games.

GPIO

The header will be used to connect the following devices:

- Screen: The screen will be powered by the 5V Power on GPIO Pin 2 and connected to the ground on GPIO Pin 6.
- Bluetooth: The Bluetooth module will be connected to Pins 8 and 10 on the header, which are for UART Tx and Rx
- Supporting board: Specifically Bluetooth, Internal Game Controller, and Audio Controller will be powered by the 5V power on GPIO Pin 4 and connected to the ground on GPIO Pin 6.
- The internal game controller will be connected to pins 1, 6, 7, 19, and 23.

These can be seen in Table 3.2.1.1 below.

GPIO Pin #	Function	Used By
1	3.3V Power	Controller
2	5V Power	Screen
4	5V Power	Peripheral PCB
6	Ground	Shared Ground
7	GPCLK0	Controller Data
8	TxD	Bluetooth Tx
10	RxD	Bluetooth Rx
18	GPIO	Bluetooth Wake
19	MOSI	Controller Clock
23	SCLK	Controller Latch

Table 3.2.1.1 – GPIO Header Map

3.2.2 Screen

In choosing a proper screen, we considered a variety of options on five different categories: connection type, resolution, size, touchscreen, and power consumption.

3.2.2.1 Connection Type

We had five different options: VGA, DSI, HDMI, SDI, and composite.

VGA is one of the simplest options, with a great deal of support and analog video, so we don't have to worry about digital signal processing. Unfortunately, connecting to VGA would also add a lot of bulk, and since the Raspberry Pi 2 doesn't have a VGA port, we opted against this technology.

DSI is a technology which would allow us to directly connect a ribbon cable to the Raspberry Pi 2 without worrying about the bulk of larger connections that need to be soldered, like HDMI or VGA. It provides a more direct connection to the screen, but also comes at the cost of being far more delicate and easy to break, such as if the ribbon cable tears. We initially considered this technology due to the positives outweighing the cons, however we later learned that the Raspberry Pi does not fully support this technology natively. Thus we ended up not considering it. A more detailed comparison is shown in Table 3.2.2.1 below.

Screen	Nokia	iPhone 3GS	iPhone 4

	N8		
Resolution	360X640	480x320	960x640
Cost	~\$35	~\$20	~\$30
Cost for additional parts, such as driver boards	~\$25	Not Found	Not Found
Conclusion	Costly	Unconnectible	Unconnectible

Table 3.2.2.1 – DSI Screen Comparison

HDMI is a nearly universal technology these days, combining video, audio, and even Ethernet in some cases, into a single cable. As it is a purely digital signal, we would not have to worry about any digital to analog conversion and the signal loss that would allow. We realized that, although one of the screens seemed like it passed muster, that simply plugging in an HDMI screen would be no fun at all. Further, it would be a little bit large for a portable game system at 5” and we could find no HDMI displays at lower resolutions. If we had wanted to, control chips were available to convert other standards for smaller screens to HDMI, but this was not a cost we felt was warranted by the size of the screen and the resolution that we needed. A more detailed comparison is shown in Table 3.2.2.2 below.

Screen	Pixel Qi 10"	HDMI 4 Pi 7"	HDMI 4 Pi 5"
Resolution	1024 x 600	1280 x 800	800 x 480
Cost	\$179.95	\$114.95	\$64.95
Conclusion	Costly	Costly	Good

Table 3.2.2.2 – HDMI Screen Comparison

After HDMI, we moved to SPI, a relatively well supported technology which the Raspberry Pi happened to support out of the box. We compared a number of different screens and finally decided on one to get. A more detailed comparison is shown in Table 3.2.2.3 below.

Screen	5" TFT LCD	3.5" TFT LCD	2.8" TFT LCD
Resolution	800 x 480	320 x 480	240 x 320
Cost	\$29.95	\$39.95	\$29.95
Cost for additional parts, such as driver boards	\$34.95	N/A	N/A
Conclusion	Costly	Good	Too Small

Table 3.2.2.3 – SPI Screen Comparison

Unfortunately, upon hooking it up, the jumper pad for the SPI header fell off. We solved this by routing the cable directly to a 3.3V source. Yet this did not solve our essential problem, namely displaying anything other than white light on the screen. We discovered that SPI was not quite as supported as we thought, as the built in driver only allowed SPI output and did not, as we thought, directly facilitate displaying the O/S screen. We would have to code separate drivers for the Raspberry Pi to transfer the display data. We decided against this, as it would cost us even more already dwindling time.

Finally, we were left with composite. An older technology, it only supports NTSC or PAL video, clearly not HD. Thankfully, this fit our purposes exactly, as all the emulated consoles originally output NTSC or PAL video streams. Further, the maximum resolution of 480i (for NTSC) was, for most applications, fairly small, but perfectly suitable for our games, which maxed out as 512 x 448. A more detailed comparison is shown in Table 3.2.2.4 below.

Screen	3.5" NTSC/PAL	3.5" TFT LCD	4.3" TFT LCD
Resolution	800 x 480	640 x 480	480 x 272
Cost	\$44.95	\$17.57	\$16.85
Conclusion	Costly	Good	Poor Aspect Ratio

Table 3.2.2.4 – Composite Screen Comparison

3.2.2.2 Resolution

We considered the need for the games from each console to still look crisp, without being too letterboxed. The largest required resolution was 512x448, which is only required for certain SNES games, and those that have it can be compressed. The smallest required resolution was 160 x 144 for the Game Boy and Game Boy Color. Based on numerous quality tests with each system and many games, we determined that any distortion of aspect ratio due to stretching or compression was negligible at 480x320 when considering the size of the screen. Thus, we opted for a screen with 480x320 resolution. The scaling comparison of the different resolutions is shown below in Figure 3.2.2.1.

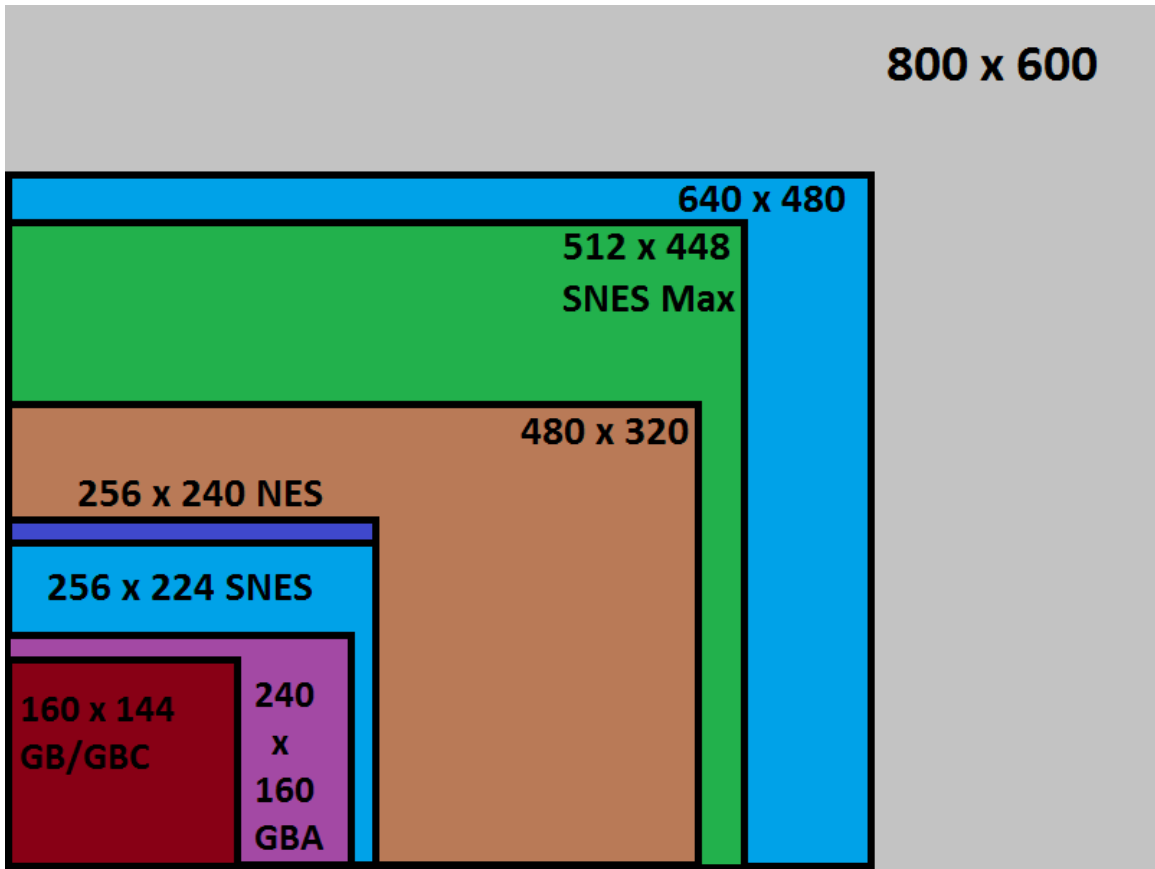


Figure 3.2.2.1 - Screen Resolution Comparison

3.2.2.3 Screen Size

We considered a variety of options, but settled between three major options: <5", 5", and 7".

A <5" screen would be ideal for the resolution chosen. Much larger would produce too much stretching or aspect ratio distortion, leading to poor quality gaming. In addition, the smaller screen size provides for lower power consumption.

A 5" screen is just a little too big, and the resolutions on all compatible screens we could find exceeded our desired resolution. In addition, the additional power draw would reduce our battery life.

A 7" screen is, quite frankly, massive. Once we came to terms with just how unwieldy a 7" screen would be in a portable, not to mention the massive power draw and cost of the display, we quickly abandoned that idea.

For reasons of power consumption, desired resolution, and desired cost, we ended up deciding on a screen that was less than 5" diagonally.

3.2.2.4 Touchscreen

We considered both types of touch screens: resistive and capacitive vs. not having a touchscreen at all.

Resistive touchscreens would have the benefit of allowing us a finer degree of control, using a thinner stylus over a much thicker finger. This would preclude the need to redesign the interface to accommodate the larger finger. In addition, the technology is cheaper in general. However, the downside would be the need for a stylus, which might not be ideal for our needs.

Capacitive touchscreens would allow us to obtain a more responsive touchscreen, with no pressure really needed. In addition, the pointing device, one's finger, would always be at hand. Unless of course one has lost all their fingers, in which case there are probably bigger issues on hand, like how to use the game console at all. Also replacement fingers. The downside, of course, is the necessity to redesign the interface to have more easily pressed (read: bigger) UI elements, and the greater cost.

Finally, not having a touchscreen would limit us to controller and other HID input. This would, however, not require us to include a stylus or increase the size of UI elements. Also, it would cut down on cost.

We researched the feasibility of implementing a touchscreen to our project. The touchscreen's input would require 4 dedicated input pins: two for the flat plane across the screen, and two for the depth of pressure with which we touched (resistive only). As we were going with resistive for cost reasons, we decided that this would use too many of our valuable GPIO pins. Furthermore, screens that we found using touchscreens and fitting our other requirements had custom, extended ribbon cables, which did not fit non custom driver boards.

As a result, we decided to go with a screen that did not have touchscreen capabilities.

3.2.2.5 Power Consumption

We ideally wanted a screen that drew ≤ 500 mA, with a backlight. This, of course, limited our size options as discussed above. Further, it made sure that we chose an efficient screen, and preferably one with an adjustable backlight, or that we made our own adjustable backlight.

For all reasons listed above, we went with a 3.5" TFT-LCD screen ripped out of a car rearview monitor. It provided the best bang for our buck, so to speak. It was the correct resolution, size, and met the power requirements, as well as being able to be directly soldered to the board. It was only \$20, a marked savings compared to screens designed for the Raspberry Pi 2. Additionally, it gives us

the opportunity to create a backlight controller using a microcontroller and to have more soldering experience.

3.2.2.6 Backlight Controller

We plan to implement a twofold system:

First, the backlight will be controlled by a hardware switch linked to the Raspberry Pi 2's GPIO ports, which will determine the level of PWM coming from other GPIO pins connected to the backlight control. This will allow the user to control the level of backlight to save power or to make the screen brighter when needed. We believe this will add significant battery life when desired, at a minor cost to usability.

Second, before the input for the backlight is even touched by the Raspberry Pi 2, it will be fed through a photo-resistor, which will automatically increase or decrease the relative backlight level to match the surrounding light. In this way, we can ensure power savings while not sacrificing much visibility.

These simple changes will ensure a better user experience at a minimal cost to develop and, as a result, we will want a screen with a backlight to control and, preferably, one that can be controlled directly with a PWM pulse, and not just an onscreen menu.

3.2.2.7 Final Choice

We decided to go with the 3.5" TFT LCD that we will rip out of a pre-existing car backup display. This screen meets the resolution requirements, ensuring that it will not stretch or compress the image to the point where the games look "off". It meets the size requirements both in image quality and weight of the system. It meets the standard requirement, being a composite screen. It does not contain a touchscreen, reducing complication. The power consumption, even with backlight at full, is less than 500 mA, ensuring long battery life. The final screen is shown below in Figure 3.2.2.2.



Figure 3.2.2.2 – Final Screen Choice (Permission Pending)

3.2.3 Microcontrollers

We started by comparing three microcontrollers: The Atmel ATtiny13, the Texas Instruments MSP430G2230, and the Microchip PIC12F1501.

3.2.3.1 Atmel ATtiny13

It is a simple, 8 pin chip with a 20MHz clock speed and 1KB of flash memory to program (as well as 64B of slower EEPROM and 64B of SRAM). It has two PWM channels and 32 general purpose registers. The version we are looking at takes 2.7 V to 5.5 V Vcc input and draws just 240 uA/MHz of oscillation. For our purposes, this will come out to around 3 mA, given our Vcc will be 3.3V at 8MHz. This can be seen in the Supply Current curve in Figure 3.2.3.1

The Atmel chip can be programmed in a variety of ways, including SPI and through a dedicated adapter/programmer. Code for the chip is written in C using the easy-to-use, and free, Atmel Studio. This also works with any Atmel AVR device, making it convenient to stay in the Atmel family when choosing chips.

Additionally, we have prior experience in programming ATtiny chips, which will make adoption time non-existent. We have also found related, though not identical, schematics to what we are trying to accomplish, assuring us that a backlight controller will be feasible using the ATtiny13. A programmer for the device can be obtained for only \$22, which is a reasonable initial cost.

It costs \$2.48 per chip, a little on the high side, but still perfectly feasible, even if we had to buy a few.

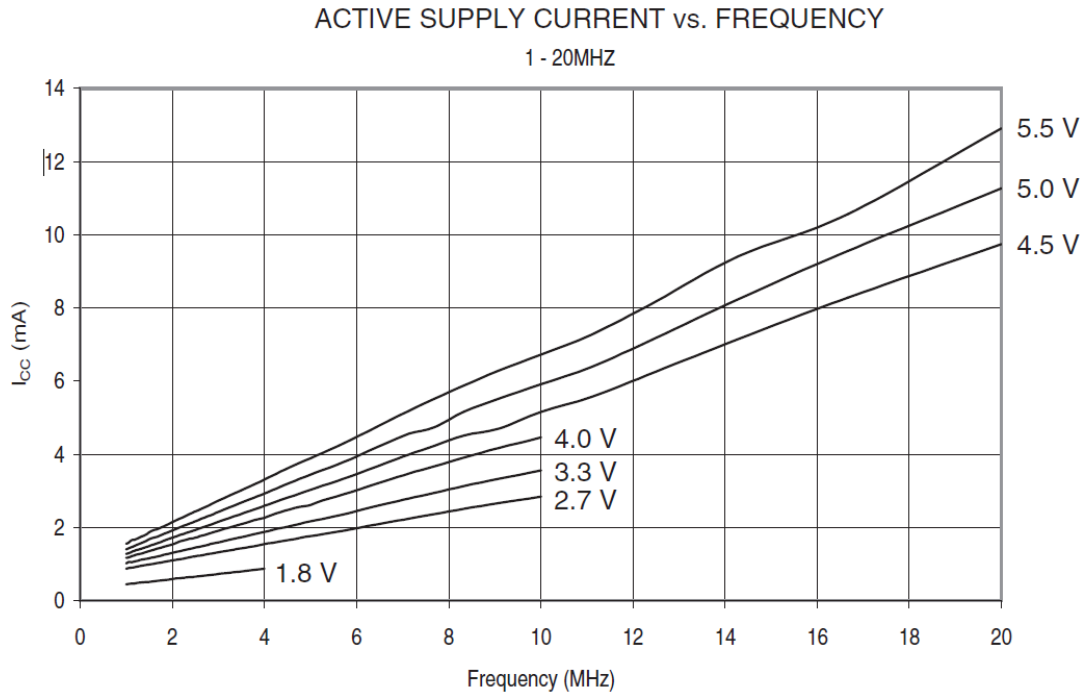


Figure 3.2.3.1 – ATtiny13 Supply Current (Permission Pending)

3.2.3.2 Texas Instruments MSP430G2230

The chip is a simple, 20-pin device with a 16MHz clock cycle and 2KB of flash memory to program (as well as 256B of slower EEPROM and 128B of RAM). It has two PWM channels and 16 general purpose registers. The version we are looking at takes 1.8 V to 3.6 V Vcc input and draws just 220 uA at 2.2V. For our purposes, this will come out to around 2.2 mA, given our Vcc will be 3.3V at 8MHz. This can be seen in the Supply Current curve in Figure 7.1.2. We can see the supply current shown in Figure 3.2.3.2 below.

The TI chip can be programmed in a variety of ways, including SPI and through a dedicated adapter/programmer. Code for the chip is written in C using the free Code Composer. This also works with any TI Embedded device, thanks to TI created libraries, making it easy to stay in the TI family of chips.

Additionally, we have prior experience in programming the MSP430, thanks to Embedded Systems, which will make it very easy to pick up coding where we left off. Code Composer is also still installed, saving a tiny amount of time installing and configuring the IDE. The FET programmer, however, is \$119, representing a significant initial cost hurdle to overcome if we do not build our own device.

It costs \$2.37 per chip, a tad bit expensive, but still perfectly feasible, even if we had to buy a few.

Typical Characteristics – Active Mode Supply Current (Into V_{CC})

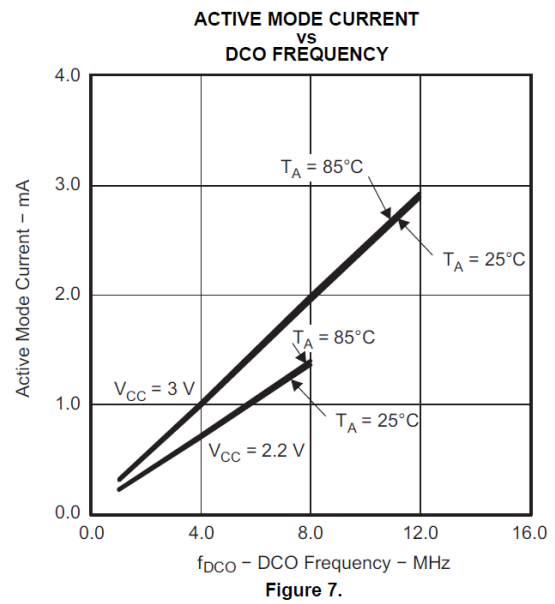
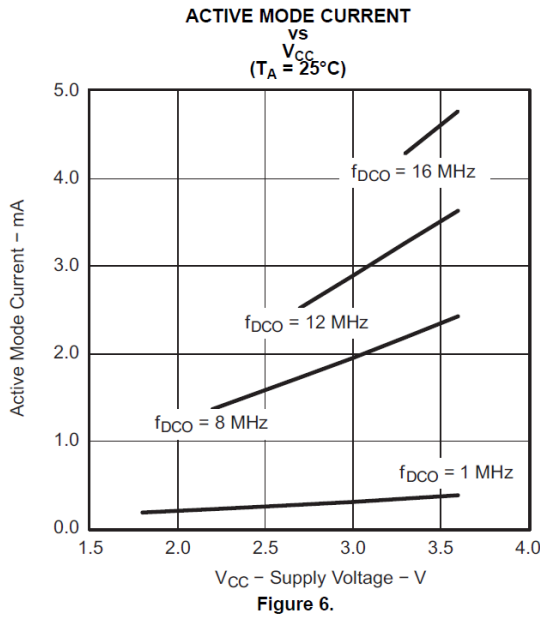


Figure 3.2.2.2 – MSP430G2230 Supply Current (Permission Pending)

3.2.3.3 Microchip PIC12LF1501

The MCU is a simple, 8 pin chip with a 5MHz clock speed and 1KB of flash memory to program (as well as 128B of slower HEF and 64B of RAM). It has four PWM channels and 12 general purpose registers. The version we are looking at takes 1.8 V to 3.6 V V_{CC} input and draws just 30 uA/MHz of oscillation @ 1.8V. For our purposes, this will come out to around .5 mA, given our V_{CC} will be 5V at 8MHz. This can be seen in the Supply Current curve in Figure 3.2.3.1.

The Microchip chip can be programmed through a dedicated adapter/programmer. Code for the chip is written in C using MPLAB. This software disables certain features after 60 days of use, so it might not be ideal for our purpose. This also works with any Microchip embedded device, making it convenient to stay in the Microchip family when choosing chips.

We have no prior experience programming Microchip PIC chips, and thus the learning curve for the various libraries and the IDE might not be desired. Additionally, the programmer required for the device is \$45, which makes the initial costs much higher than other options.

It costs \$.89 per chip, which is a very reasonable cost to adopt, and it slightly offsets the initial high investment of the programmer.

FIGURE 28-9: I_{DD} TYPICAL, EXTERNAL CLOCK (ECH), HIGH-POWER MODE, PIC12LF1501 ONLY

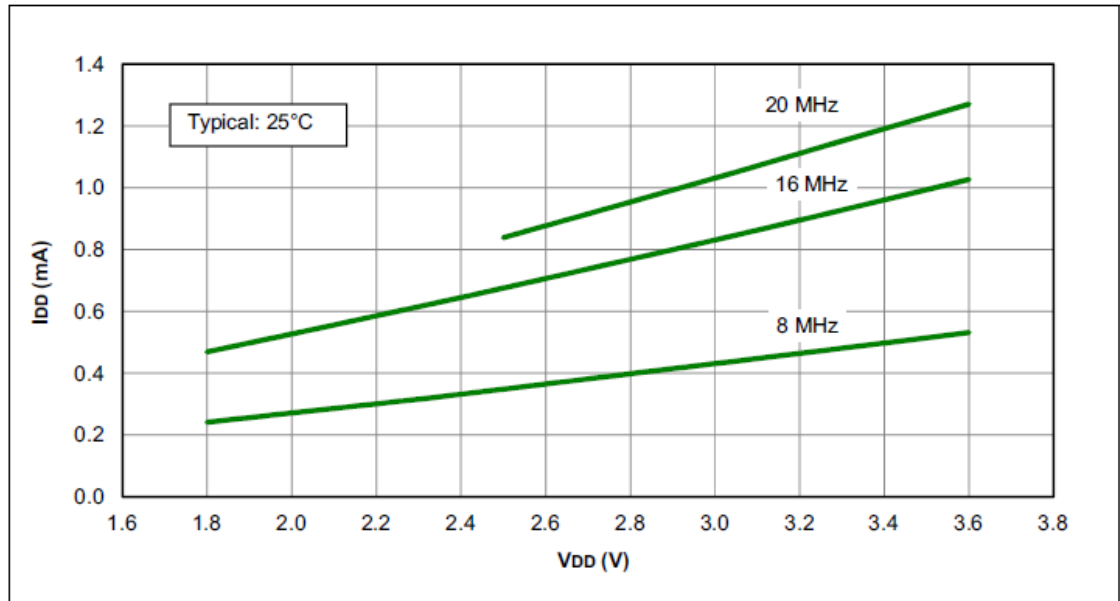


Figure 3.2.3.1 – PIC12LF1501 Supply Current (Permission Pending)

3.2.3.4 Backlight Controller MCU Selection

Based on the information obtained, we decided to go with the Atmel ATtiny13. While the PIC and TI chips did have lower current draw, we decided this was negligible when compared to four other factors.

First, the ATtiny13 had a cheaper programmer that could be used for our surface mount final chips as well as our DIP testing chips.

Second, we considered using an ATmega as well, which would mean we would not have to buy another programmer or setup additional software.

Third, the Atmel chip was most popular among hobbyists. This led to much greater community support and, therefore, many more base schematics we could draw from. This would also give us a greater network for support, in case anything went wrong.

Fourth, the Atmel chip supported up to 5.5 V, giving us confidence that the chip would continue to operate if there were any sudden spikes in voltage, for instance if the power subsystem failed.

Based on these benefits, the Atmel ATtiny13 was the clear choice for the backlight controller.

3.2.4 Communication Technologies

Fun Box Classic will have a multiplayer capability. Two people can play at the same time on one console. This can be done in two ways. The first way is when two users have two players inside the game playing cooperatively against each other.

The other way is when the screen is divided into two parts and two users play the game side-by-side of each other. In order to connect a second user to the console, we will have external controllers that will communicate wirelessly with the console. We can implement wireless connectivity via two types of technologies – Wi-Fi and Bluetooth.

3.2.4.1 Available Wireless Technologies

Both Bluetooth and Wi-Fi deliver good performance, security and functionality that are necessary for secure local wireless transaction communication. However, there are several key differences that have to be addressed for making a decision on which technology we should use in our design.

Bluetooth versus Wi-Fi

Wi-Fi provides higher range of wireless connection. It can cover up to 100 meters where Bluetooth based wireless connections generally designed to cover up to 30 meters. The range is varied based on the class of radio used in implementation. Bluetooth technology has three classes of range. A Class 3 radio has a range of about 1 meter or 3 feet; a Class 2 radios has a range of 10 meters or 33 feet; a Class 1 radios has a range of 100 meters or 300 feet.

If were to use our console as the only media to display a game, we would choose a range of the Class 3 radios. However, we would like the users to be able to connect the Fun Box Classic console to the big display of their choice such as TV display. For this purpose, we'll need to use a bigger range of Bluetooth. Hence, a Class 2 radios will be an ideal choice.

The data transfer rate is not a big concern for our device. We rather have to make sure that our software implementation works good that the control commands over Bluetooth are processed in a way that there is no delay. We will need to take into consideration minimizing the processing delay. Bluetooth offers the security such as 16-digit PIN authentication, frequency hopping and data encryption. Wi-Fi security is higher than the security for Bluetooth. Wi-Fi is protected by different encryption protocols such as WPA (Wi-Fi Protected Access) and WEP (Wired Equivalent Privacy). Bluetooth security is pretty much limited to a key matching, but it is still very secure. Bluetooth devices “cannot be addressed by unauthorized Bluetooth devices because they do not “listen to” incoming messages from any other Bluetooth devices and are not configured to

be “discoverable”.” (2) Bluetooth terminals are protected by 16-digit PIN codes. In order to make a Bluetooth device discoverable and connectible, a hacker will have to first guess a Bluetooth code, which is generated through the Diffie-Hellman key agreement protocol. According to the protocol, the code is 16-byte or 128-bit long and different for each base station, which means there can be over 3×10^{37} different possible PINs.

Frequency hopping for Wi-Fi based networks is within 2.4, 3.6 and 5 GHz. Frequency hopping for Bluetooth is within a 2.4 GHz spectrum.

Deciding to Use Bluetooth

Wi-Fi enables a very fast connection, very big range from the base station, and very high level of security. Bluetooth wireless communications is a simpler technology. It can easily replace the cables that connect devices and, at the same, it provides relatively high levels of security.

Bluetooth is convenient to use when the information need to be transferred between two or more devices that are near each other and the speed is not an issue. Bluetooth technology also requires low power, low cost, and it is ubiquitous.

3.2.4.2 Candidate Bluetooth Module

Choosing the type of Bluetooth chip that will be suitable for our device will depend on several factors such as low power consumption, a range of the wireless single, and compatibility with other devices. We are going to use the Bluetooth chip to interface it with Raspberry pi processor. After conducting some research, we found three Bluetooth chips that would be the most suitable for our design. We have to decide between BR-LE4.0-S2N that designed by Blue Radios, PAN1026 – by Panasonic and RN4020 – by Microchip modules.

BR-LE4.0-S2N Module

BR-LE4.0-S2N, part of Bluetooth version 4.0, is a low energy wireless technology module. The size of the chip is 11.8 x 12.6 x 1.9 mm. To learn the cost, we need to call to Digi-Key. But approximately it is \$13 per chip if buying 10 chips minimum.

It covers over 150 meter or 500 ft distance with integrated antenna. It can be externally controlled via simple ASCII AT commands over the UART or programmed with custom applications embedded in the module.

BR-LE4.0-S2N has very low power consumption – 27mA 0dB TX, RX down to 19.6mA, .9uA sleep w/timer, and 0.4uA deep sleep. It is compatible with TI TPS62730 step down converter, which can extend battery life by up to 20%. It

has 10 milliseconds connect time and low data latency. It supports software adjustable transmit power from short to long-range applications (Class1, 2 & 3).

The chip can handle between 2.4 and 3.6 Volts of power supply voltage. The recommended setting is 3.0Vdc. And it should be receiving less than 10mV pick-to-pick noise. Maximum voltage VDD on any pin is 0.3 V. Current consumption is 24 mA.

PAN1026 Module

PAN1026 part uses Bluetooth version 4.0. It has a dual mode – place-and-play RF module. Among its features are Wi-Fi coexistence and high-speed interfaces: USB 2.0 UART up to 4.3 Mbps. The size of the chip is 15.6mm x 8.7mm x 1.9mm, and it is fully shielded to improve immunity. A Digi-Key part number is P16771CT-ND. The cost is \$15.3 per 1 chip.

The PAN1026 is a short-range Class 2 Bluetooth dual-mode module. The module is compatible with iOS and Android devices, wireless sensors, and can be used as a cable replacement. The embedded serial port profile frees application resources while the command set API creates a simple but flexible firmware interface. PCB layouts are simplified using available Gerber files and minimized with Panasonic's tiny footprint technology.

PAN1026 is a low energy module designed to create low data rate networks using a minimum amount of power. It provides an ultra-fast connection time of 3 ms. It can handle a single Vcc supply between 1.7 and 3.6 Volts.

RN4020 Module

RN4020 uses Bluetooth Version 4.1. It integrates RF, a baseband controller, and command API processor, making it a complete Bluetooth Low Energy Solution. The size of the chip is 11.5 x 19.5 x 2.5mm. A Digi-Key part number is RN4020-V/RM-ND. The cost is \$10.61 per one module.

The RN4020 has a built-in high performance PCB antenna optimally tuned for long range, typically over 100 meters.

The small form factor, surface mount module has the complete Bluetooth stack on-board and is controlled via simple ASCII commands over the UART interface. The RN4020 can be remote controlled by another module over a secure connection and can be updated via the UART interface or over-the-air.

RN4020 is a low energy module for designers who want to easily add low power wireless capability to their products. It can handle a single operating voltage in the range between 3 to 3.6 Volts. Tx Power Consumption is 16 mA, Rx Power Consumption is 16 mA.

3.2.4.3 RN4020 Module Attributes

This modulus uses the latest Bluetooth 4.1 version versus 4.0 that is offered for other two modules. Newer Bluetooth version has better performance and more features.

Cost

The cost of this module is a great standout point. The RN4020 is the cheapest Bluetooth module among its competitors. There is no requirement to buy some specific minimum amount of RN4020 modules. Digi-Key Electronics has 83 available modules in stock that can be purchased at any time, and the price for a single unit is only \$10.61.

The price for a single unit for PAN1026 module is \$15.3. BR-LE4.0-S2N chip is only available for purchase when buying several units. Minimum 10 units have to be purchased with \$13 per each unit.

Size

The size of RN4020 is 11.5 x 19.5 x 2.5 mm in WxLxH and weight is 1.2 grams. This size seem to be pretty reasonable comparing to two other sizes of 11.8 x 12.6 x 1.9 mm for BR-LE4.0-S2N and 8.7 mm x 15.6 mm x 1.9 mm for PAN1026 modules. The widths of the module are a little smaller than the width of the BR-LE4.0-S2N chip. The length of RN4020 is a little bigger than length of the other two modules. The height is 0.6 mm taller than other two modules.

Power Consumption

Microchip's RN4020 Bluetooth Low Energy Module provides a highly integrated solution for delivering low power Bluetooth 4.1 solutions. The advanced command interface offers rapid time to market. The RN4020 module complies with Bluetooth specification version 4.1. It integrates RF, a baseband controller, and command API processor, making it a complete Bluetooth Low Energy Solution.

RN4020 has the best low energy power consumption among three modules. RN4020 can handle a supply voltage in the range between 1.8 and 3.6 Volts DC. We need for our design an input voltage up to 3.3 Volts, which perfectly coincides with the modules voltage range.

A working current depends on profiles but typically 12 mA. A standby current is less than 0.5 mA. Current consumption for dormant mode is less than 700 nA, deep sleep < 5 uA, idle < 1.5 mA, Tx/Rx active is 16 mA at 0 dBm. (3)

3.2.4.4 RN4020 Interface

Hardware Interface

The primary communication interface between the Raspberry Pi MCU and the RN4020 Bluetooth Module shall consist of a Universal Asynchronous Receiver/Transmitter (UART) bus. The UART allows asynchronous serial communication between the RN4020 peripheral module and the Raspberry Pi MCU.

The RN4020 Bluetooth Module has the capability to be commanded by the application software via the recommended and additionally provisioned hardware control lines. The additional provisioned hardware control lines will be used to extend the utility of the RN4020 during development and/or for the final consumer product. The interface between a microcontroller and the RN4020 is shown in Figure 3.2.4.1 below.

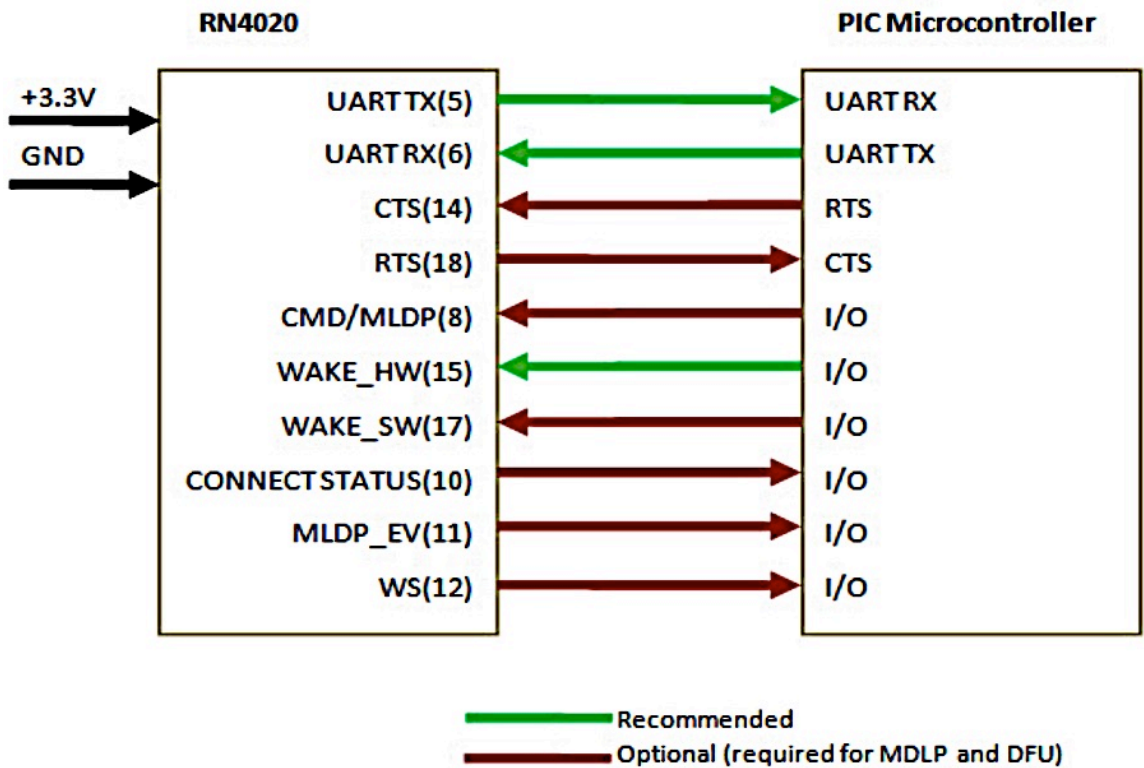


Figure 3.2.4.1 – Interface Descriptions (Permission Pending)

Software Interface

The RN4020 Bluetooth Module utilizes the ASCII command Application Programming Interface (API) defined in the RN4020 Bluetooth Low Energy Module User's Guide. This document will be the primary source of information regarding the RN4020 software command interface.

3.2.4.5 RN4020 Requirements:

Hardware Requirements

The Main Board shall provide a voltage regulator circuit to regulate the operating voltage of the RN4020 Bluetooth Module.

The RN4020 Bluetooth Module voltage regulator shall reset the RN4020 module when the voltage is outside of the recommended operating range.

The RN4020 Bluetooth Module voltage regulator circuit shall provide a $+3.3 \pm 5\%$ DC voltage source.

The Main Board shall provide a serial communication interface between the RN4020 Bluetooth Module and the Raspberry Pi MCU.

The Main Board shall provide the control lines for hardware flow-control between the RN4020 Bluetooth Module and the Raspberry Pi MCU.

The Figure 3.2.4.2 shows the module's pin-out. In this section, are going to specify the use of physical pins, and how they are connected on the circuit board. For our design, we do not need to use all the pins. Below is the description and discussion of pins that we will need to use.

We will definitely need to use the generally required pins such as GND and VDD. The hardware interface requires the use of UART. UART_TX and UART_RX will be used to send or receive data that is coming from evaluation board.

The RTS and CTS pins are needed for hardware flow-control in order to provision for possible application software needs. These pins allow the use of Request to Send and Clear to Send signals to synchronize communication between the RN4020 processor and the Raspberry Pi MCU.

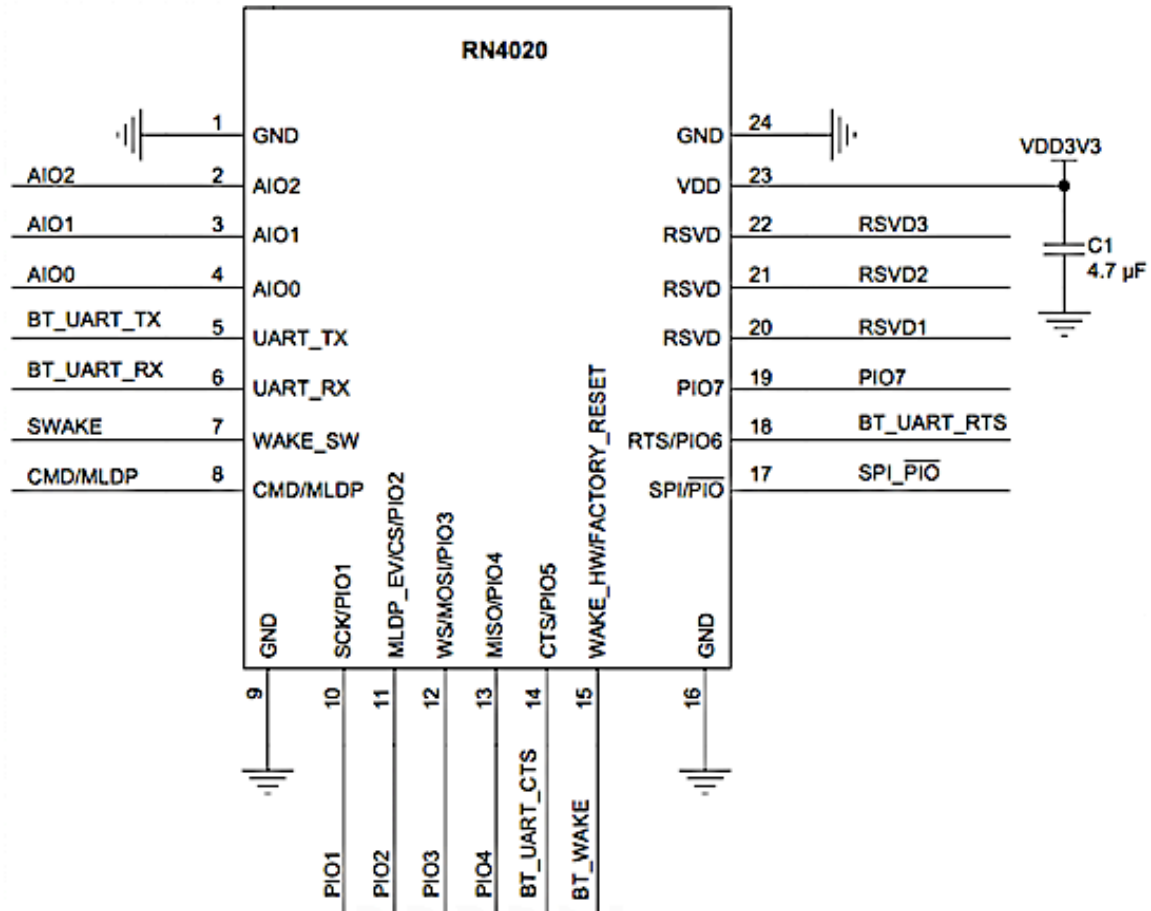


Figure 3.2.4.2 - RN4020 Pin Diagram (Permission Pending)

WAKE_SW – Deep Sleep Wake; active-high to wake module from Deep Sleep. Function: Input; weak pull down
CTS PIO [5] - Reserved for CTS if hardware flow control is on the UART. Function: CTS (input)
WAKE_HW - Hardware wake from dormant state. Function: Active-high; internal pull down.

Figure 3.4.2.3 illustrates the RN4020 pins that will be used. Unused pins will be left in a default configuration or grounded per the module specification. The rationale for not utilizing all the available pins is due to unneeded advanced hardware debug functions and manufacturer firmware update modes that will not be used during the lifecycle of this project.

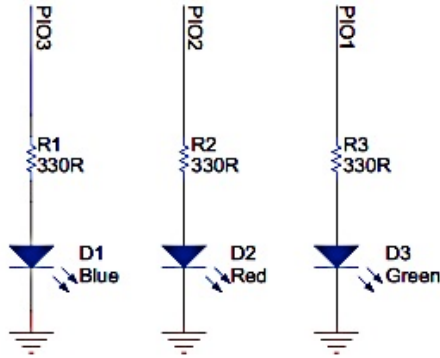


Figure 3.2.4.3 - Status LEDs (Permission Pending)

Software Requirements

In this section, we are going to expand on how the pins will be used and for what reasons.

Status LED Pins

We have three status LED pins available to us. We need to decide for what purposes we may need to use LEDs. We probably will want to show one LED to the user. Then, we'll need to have a power LED that will indicate that the device is powered. And then, we will need to have some sort of network LED that will indicate that the Bluetooth-active connection is going on. For Bluetooth connection, we may choose to blink LED or not.

Three status pins that can be set as LEDs or for other purposes:

- 1) CONNECTION LED (Green LED)/ PIO1/ SCK pin:

This pin can have three configurations that are defined by software. It can be used for general purpose, diagnostics or as a connection LED. For connection LED, default state is output: active-high indicates the module is connected to a remote device. Active-high indicates a disconnected state. For general purpose, the pin can be configured as PIO1 via software command. The pin should be configured as SCK or diagnostics and factory calibration if pin 17 is asserted. In our case, we may choose to use this pin as a connection LED.

- 2) MLDP_EV (Red LED)/ PIO2/ CS

The pin can be used for general purpose, diagnostics or as the LED indicating MLDP data event. For MLDP_EV, default function is output used for MLDP data event indicator (Red LED). Active-high indicates MLDP data received or UART console data-pending. Low level indicates no events. Event only triggered in CMD mode, when CMD/MLDP (pin 8) is high. For general

purpose, the pin can be configured as PIO2 via “|>” and “|<” commands. The pin should be configured as CS for diagnostics and factory calibration if pin 17 is asserted.

3) WS (Blue LED)/ PIO3/ MOSI

The pin can be used for general purpose, diagnostics or as the LED indicating activity. For LED, default function is an output used for activity indicator. High level indicates module is awake and active. Low level indicates module is in a Sleep state. For general purpose, the pin can be configured as PIO3 via “|>” and “|<” commands. The pin should be configured as MOSI for diagnostics and factory calibration if pin 17 is asserted.

So, the Bluetooth Module gives us three options. The connection LED is our first option. It will be useful when the device is connected to a remote device. This will toggle between high and low, depending, if the connections are active or not. The second pin MLDP_EV gives us an event. Any time the data is received over UART, which means the device sends data to the Bluetooth, it blinks or toggles high and low. It will be useful during the development stage because it will show us that there is a device connected. The third LED is an activity indicator, and is really only used for engineering purposes. We won't need to use it.

Using LEDs during the development stage will help us visually confirm what our hardware is doing. For this reason, we will need to have some visible LED or LEDs that will show that there is some sort of connection or activity going on in the network. It may be a combination of green PIO1 and red PIO2 LEDs since the first LED is simply shows when the module is connected. We may want to have some LED blinking when there are things happening. The first LED is not going to blink because it is going to stay on all the time showing us that there is something connected. So, if we want to create a better Bluetooth status LED, we can use combinational logic from both LED pins, PIO1 and PIO2, to create the desired LED signal output.

Data Transfer Pins

CTS/ PIO5 pin – reserved for CTS (clear to send) if hardware flow control is on the UART. The pin can be configured as CTS (input) or PIO5.

PIO6 – reserved for RTS (request to send) if hardware flow control on UART. Configurable as PIO6 if hardware flow control disable. The pin can be configured as RTS (output) or PIO6.

There are two data transfer pins CTS (input) and RTS (output) that are used to control the data flow on hardware. These pins allow two devices to talk to each other and control when the data should be sent and at what rate. It's a nice plus to have these pins if we will decide to do something like that. Realistically, the

RTS and CTS pins are used in situations when the receiver and transmitter need to communicate without the risk of losing data under specific conditions. Conditions such as system scheduling, timing, and high transfer rates can cause data loss. UART Flow Control makes use of the RTS and CTS signals to allow devices to communicate under these conditions. Even if an immediate need for flow control is not needed, it is a good practice to create a design margin for future applications.

Control Pins

WAKE_HW pin allows the RN4020 module to be signaled to exit a dormant state.

WAKE_SW pin allows the RN4020 module to be signaled to wake from a Deep Sleep state. The use of these pins allows the RN4020 module's state to be controlled by a software implementation.

3.2.4.6 RN4020 PCB Layout

Dimensions

Figures 3.2.4.4 and 3.2.4.5 illustrate the RN4020 Bluetooth Module dimensions. It is important to account for these dimensions into the overall Main Board layout and care design.

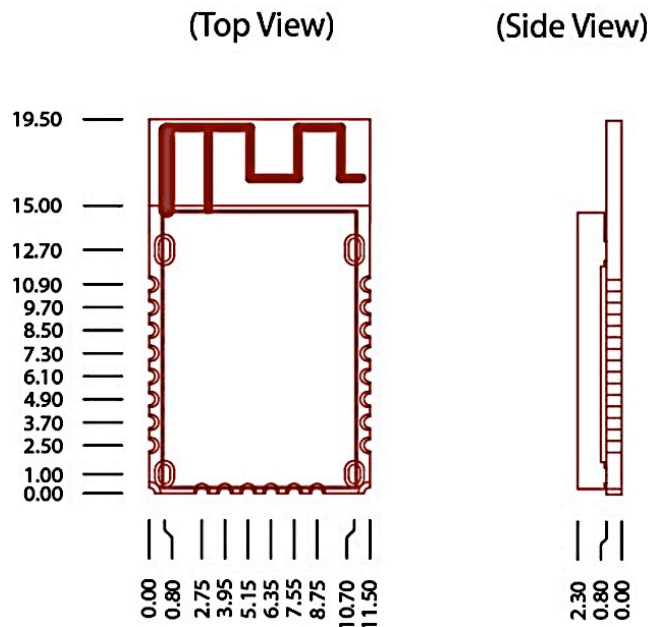


Figure 3.2.4.4 – Top View and Side View of Dimensions (Permission Pending)

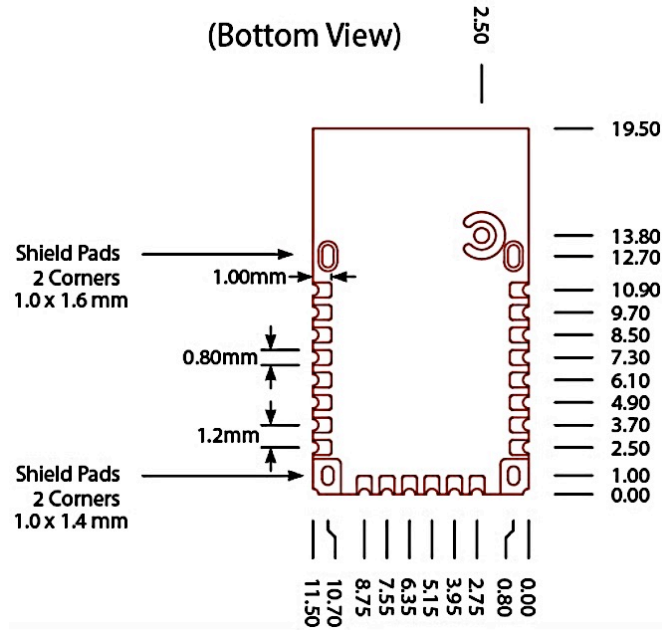


Figure 3.2.4.5 – Bottom View of Dimensions (Permission Pending)

Mounting

Figure 3.2.4.6 shows the recommended mounting details. For optimal radio performance, the RN4020 module's antenna end should protrude at least 31 mm beyond any metal enclosure. The PCB antenna is fabricated on the top copper layer and covered in solder mask. The layers below the antenna do not have copper trace. It is recommended for module to be mounted on the edge of the host PCB. It is permitted for PCB material to be below the antenna structure of the module as long as no copper traces or planes are on the host PCB in that area.

Figure 3.2.4.7 shows example of good, bad, and acceptable positioning of the RN4020 on the host PCB. When laying out the carrier board for the RN4020 module, the areas under the antenna, RF text point (semi-circular pad) and shielding connections should not have surface traces or ground planes.

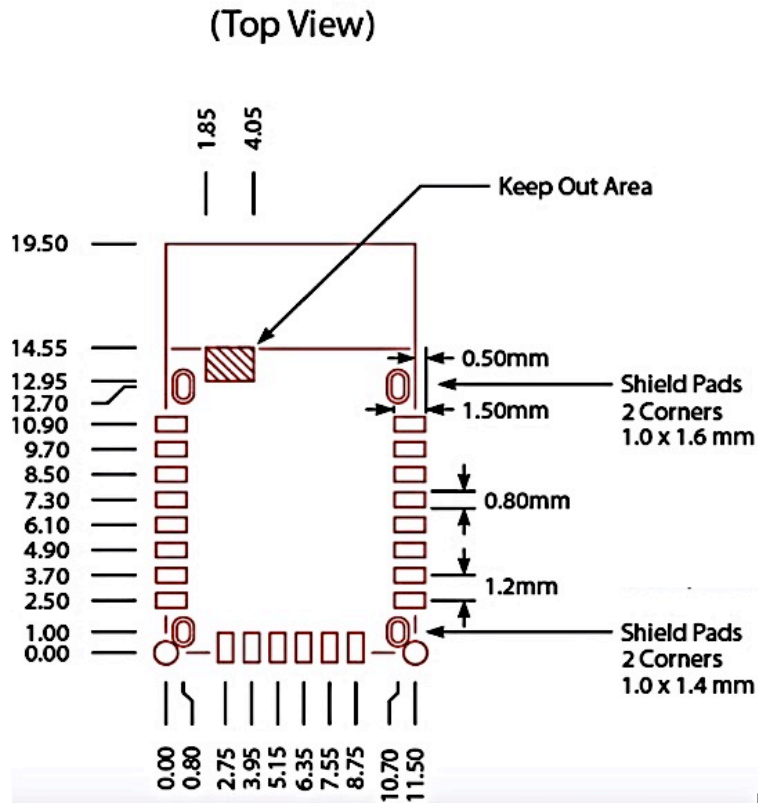


Figure 3.2.4.6 – Recommended PCB Footprint (Permission Pending)

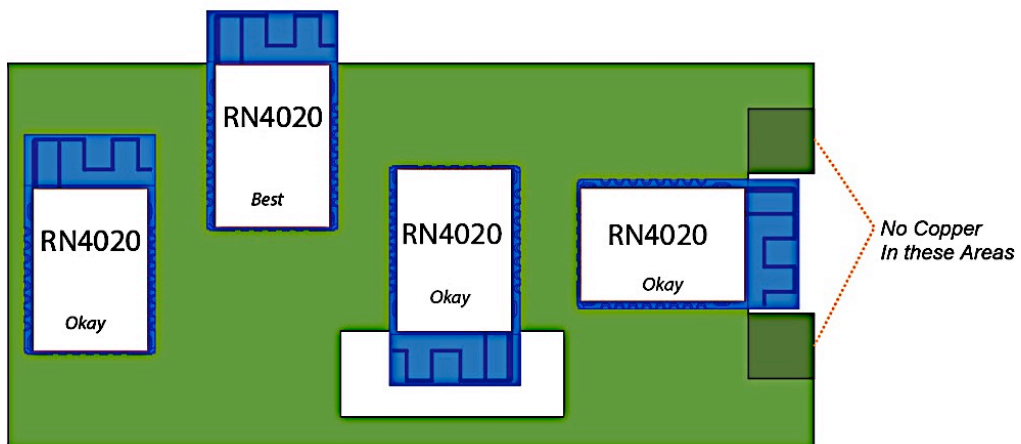


Figure 3.2.4.7 – RN4020 Host PCB Example Layout (Permission Pending)

Soldering

The RN4020 wireless module was assembled using standard lead-free reflow profile IPC/JEDEC J-STD- 020. The module can be soldered to the host PCB using standard leaded and lead-free solder reflow profiles.

To avoid damaging the module, the following recommendations are given:

- Microchip Technology Application Note: “AN233 Solder Reflow Recommendation” (DS00233) provides solder reflow recommendations
- Do not exceed peak temperature (T_p) of 250 C
- Refer to the solder paste data sheet for specific reflow profile recommendations
- Use no-clean flux solder paste
- Do not wash as moisture can be trapped under the shield
- Use only one flow. If the PCB requires multiple flows, apply the module on the final flow

3.2.5 LEDs

During operation of the FBC, it is ideal for a form of a power status indicator light to be available to the user for feedback on the remaining battery life. Considering the need for a visual representation to be compartmentalized for a hand-held, portable device, the use of light emitting diodes (LEDs) are a sufficient choice to indicate the battery charge status. LEDs lend themselves to low power consumption, color representation versatility, and ideal surface-space minimization. The FBC offers two LED-based battery status indicators, one which directly indicates the battery charge status with an RGB LED, and the other which will utilize surface mounted miniature LEDs to indicate a more precise measure of remaining battery life.

3.2.5.1 LED Light Color Properties

The LED is to always be considered as a current-powered pn-junction diode, with an individually identified voltage contribution when illuminated (in the ON status). The current flowing through the LED, the forward current, determines the amount of light, or brightness, emitted, with the maximum forward current distinguishing the limit of current that can pass through the diode without catastrophic failure. Additionally, the LED contains a forward voltage to be considered in relation to the power supply. Maximum forward voltage and current values are specified for the LED used, with common value ranges expected within certain colored LEDs, such as red or green diodes, to be lower than that of blue LEDs, which require higher voltages. Figure 3.2.5.1 below compares the forward voltage and forward current of a red LED and a blue LED. We notice that the red LED, which has a forward voltage of 2V, has a much lower tolerance of current in excess of its forward rating. The blue LED, meanwhile, can withstand values higher than its forward current rating, failing around 3.8V. However, even though the blue LED can function past its forward current rating, doing so can damage the diode and shorten its performance life significantly.

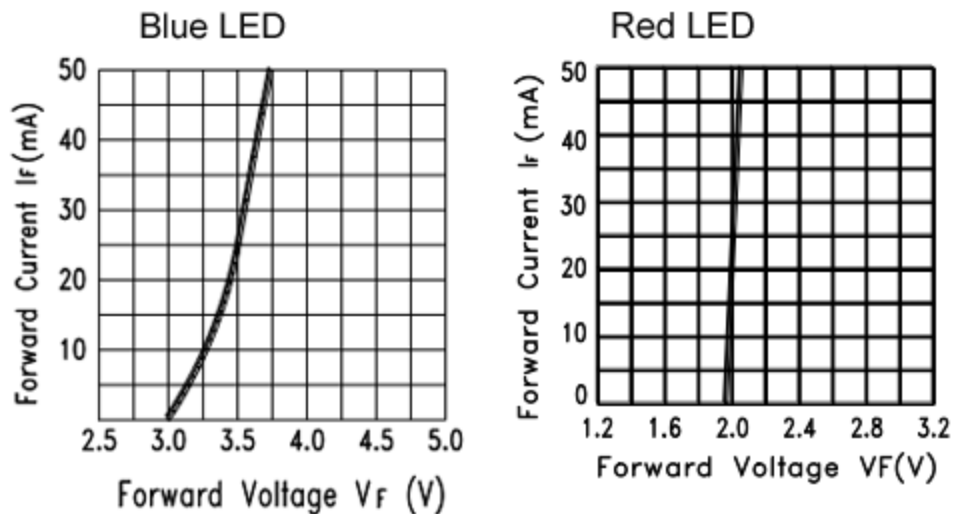


Figure 3.2.5.1 Blue and Red LED I-V Characteristics

It is always important to not overload the LED with too much current that would potentially burn out the diode. A simple solution to restrict current flow would be to design a two-part voltage regulator, consisting of a current-limiting resistor in series with a Zener diode and the LED. The resistor value will be chosen to ensure that the current flowing through the diode will not exceed the maximum forward current value, and the Zener diode, with contribution from the LED forward voltage, will limit the voltage from the power supply across the resistor. This series circuit is ideal for its simplicity and effectiveness. A disadvantage is that it lacks efficiency, as the resistor releases heat as it limits current. However, due to the LED circuit reading directly from the DC battery power source, at a low maximum voltage of 4.2 volts, the resistor/Zener series regulator will be sufficient.

3.2.5.2 Red, Green, Blue (RGB) LEDs

The RGB LED serves as three separate colored LEDs in one 'bulb' display. Within the device are three separate diodes, one each for the red, green, and blue LED. The RGB LED consists of four pins, one anode, or negative, pin for each of the three diodes, and a cathode, or positive, pin commonly shared by all three diodes. Even though the RGB LED joins three separately colored LEDs under one bulb, each one behaves as a single LED, and follows the forward voltage and current restrictions unique to that particular LED. Likewise, the same voltage regulator circuit design can be applied to control each of the three LED colors separately, and in combination to generate many various color outputs.

The RGB LED is ideal for color combinations and sequencing illumination signals, with a wide range of combinations available for output via incorporating the three different LED colors together. Additionally, forward current increase or reduction can dim or brighten the LED for a desired dominant color, and can be used to transition from one diode in the RGB LED turning off to the other turning on, assuming the change in current is not largely immediate in value extremes. Figure 3.2.5.2 below demonstrates the color spectrum wavelengths achieved for maximum intensity, with overlapping defining the capability to combine two separate colors for a third spectrum option. In the FBC application, each of the three available diodes in the RGB LED will be used for a specific status, however, and the transition from each diode within the RGB LED will be the focus.

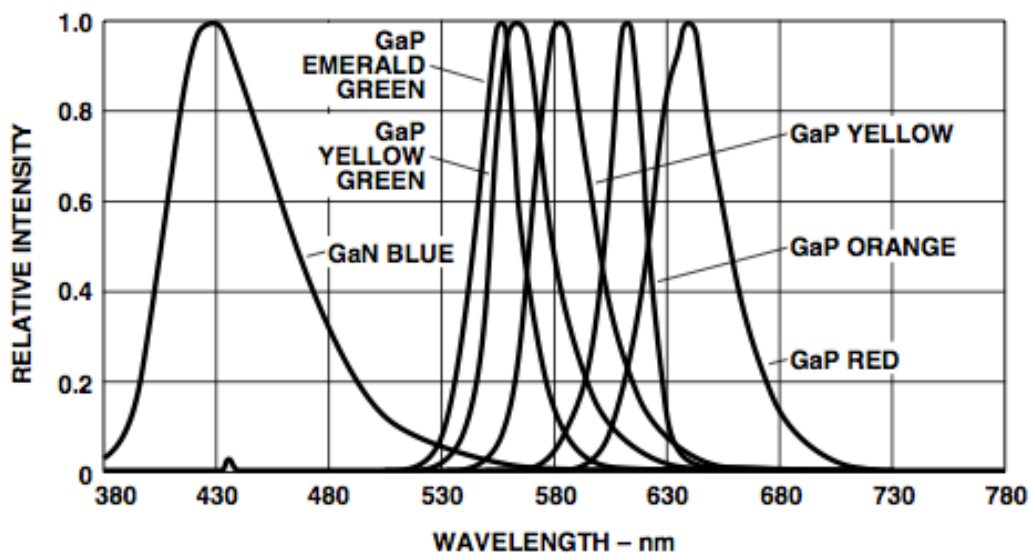


Figure 3.2.5.2 - RGB LED Wavelength Spectrum (Permission Pending)

3.2.5.3 Surface Mount LED (SMD LED)

Surface mounted LEDs (SMD LEDs) are, by comparison to single LEDs, much shorter in height from the surface board, with the average SMD measuring 1.6 millimeters tall, compared to a standard green LED that measures 8.6 millimeters tall. In design, SMDs are designated to be soldered directly to the surface, such that the anode and cathode regions are not evident with pins, but are instead represented as a mounted platform at opposite ends of the SMD. This height reduction makes the SMD ideal for designs with limited spacing constraints, as is the case with a handheld portable device as the FBC. With a square, flat design, SMDs also boasts a significantly increased viewing angle, which is the angle of brightness away from the viewing center of the LED. The SMD viewing angle is typically 100 to 140 degrees, compared to the 30 to 60 degrees viewing angle of a single green standard LED, making the SMD ideal for mounted displays that

can give off a wider range of light across a surface. Figure 3.2.5.3 below shows the viewing angle range standard for LED displays.

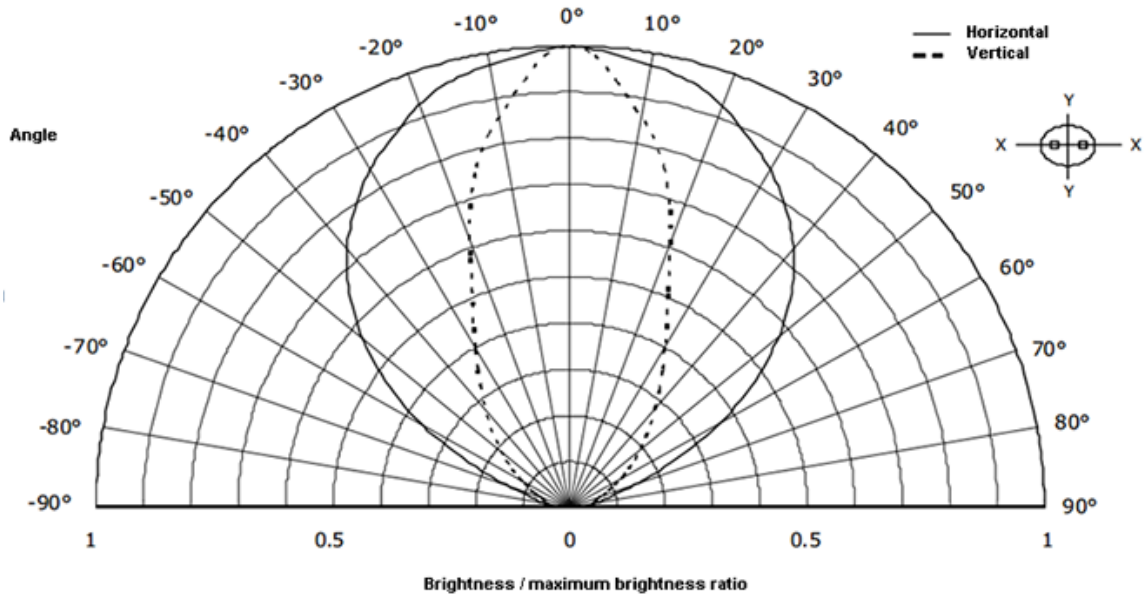


Figure 3.2.5.3 - Viewing Angle Spectrum for LED (Permission Pending)

Selecting the ideal SMD LED for the indicator circuit revolves around taking in to consideration the acceptable forward current and LED-Zener voltage summation contribution. The standard forward current for LEDs is 20 mA, with a lower current simply reducing the luminosity of the LED, and a higher current potentially shortening the operational life of the diode, and even destroying it. Wanting to reduce the current draw from the power source while still attaining measurable current through the microcontroller, we must consider the voltage contribution that will attribute the current along the current-limiting resistor. We are looking for a SMD LED with a lower forward voltage to factor into the Zener contribution, while meeting the ideal forward current and viewing angle advantages found in SMD LEDs. Also, the size of the SMD LED will be considered for space availability of the FBC design layout. Below, in Table 3.2.5.1, are three comparable green SMD LEDs for consideration of the battery power indicator circuit. We chose a green LED indication, as green lighting has been a universally common indicator of sufficient status for most electronics.

Model	Forward Current	Forward Voltage	Viewing Angle	Size Dimensions
LG L29K	20 mA	1.7 V	160°	1.3 mm x 0.8 mm
APA2106MGC	20 mA	2.1 V	120°	2.1 mm x 0.6 mm
LG R971	25 mA	2.2 V	160°	2.1 mm x 1.35 mm

Table 3.2.5.1 SMD LED Model Comparison

Looking at the three green SMD LED models, we first compare the forward current. The LG R971 is the only one that breaks the standard of 20 mA, having a higher tolerance of 25 mA. The advantage to this is that the LED can withstand a higher current draw, allowing for more flexibility in terms of current response from the power system. However, the higher forward current means that the LG R971 requires higher than normal current to achieve maximum brightness, and we want to be mindful of the allocation of current throughout the device for battery supply purposes.

Looking at forward voltage, the LG L29K has a much lower than average rating of 1.7 V. This can allow for lower Zener diode voltage contributions in relation to the current-limiting resistor, and better manage the allowable forward current through the LED. When considering viewing angle, the APA2106MGC is the least practical, as that it has an angle of 120°, much lower than the typical 160°. Since improved viewing angle is, in nearly every sense, preferred, we can eliminate the APA2106MGC from consideration for our design. With size constraints, the LG R971 width, by box design, takes up much more space than other LED models. However, with only 3 SMD LEDs being employed in relation to their side visibility of the case design, this constraint is flexible to overcome.

With all considerations, we choose the LG R971 model over the LG L29K. This decision is primarily based on the LG R971's higher current tolerance, which, for SMD LEDs, is of the highest importance for both turning on the LED while ensuring it withstanding potential current-overdraw. Furthermore, the LG R971 does not require maximum brightness, and lends itself to more flexibility with acceptable current to adequately illuminate for battery status indication. Figure 3.2.5.4 displays the dimensions of the LG R971, both in millimeters and inches, to showcase its desired minimal surface area for design implications.

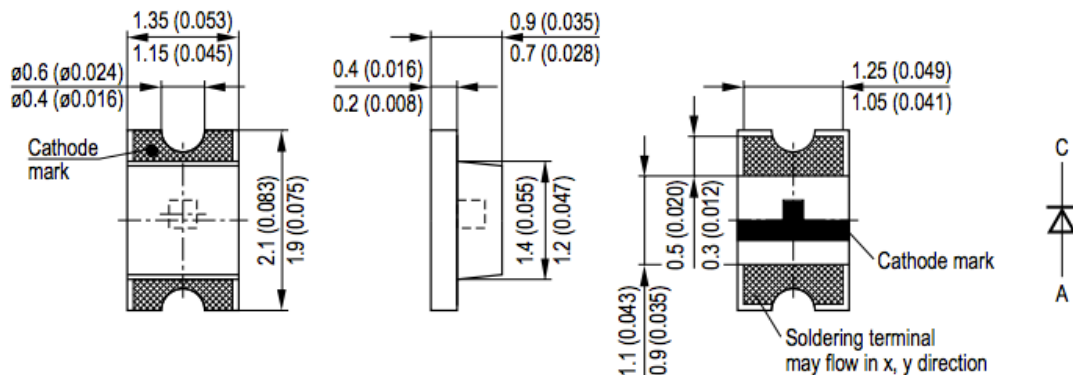


Figure 3.2.5.4 - LG R971 Dimensions (Permission Pending)

3.2.5.4 LED Challenges

The most significant restraint to most LED technology is the forwarding current through the circuit. Since LEDs are under the diode family, they are operated at specified current levels, with a maximum forwarding current defining their limit. At maximum rated current, the LED will be at its peak luminosity, but will also be prone to quick component degradation, and a shortened functional life span. Current exceeding the forward current, past a maximum value, can destroy the diode, and permanently burn out the LED, ruining it for further operation. While LEDs can operate below their forwarding current value, the luminosity will be lessened, and the output can be dim. One of the considerations for any LED circuit is being able to supply an acceptable forward current through the LED. Unfortunately, the standard for forwarding current is 20 mA, which can be proportionally small compared to the current output throughout the device. Realistically, we can implement current-limiting resistors at the LED inputs, which can appropriate current. Although economically speaking, LEDs and resistors are not a financial concern in numbers, size constraints for the FBC limit the amount of additional components added to regulate current.

Another concern for working with LEDs, and SMD LEDs, in particular, is the small workable surface factor. The SMD LEDs we are working with are only 2.1 mm long and 1.35mm wide. For actual implementation on the designated work board, these size constraints require specialized small-scale tools to solder and test the components. This is a necessary constraint, however, as the FBC is meant to be a limited-sized handheld device. For building the SMD LED battery indicator circuit, investment and knowledge in small-scale soldering is to be desired.

3.2.6 Solar Paneling

A specification for the design includes a means to recharge the power supply battery via exterior solar paneling that effectively extends the power supply while meeting the constraints of the power output. While means to charge battery power supplies via solar cells is simple, panel dimensions, quantity, and charge characteristics must be considered. Additionally, the output voltage from the solar module contributions must not overcharge the initial power source, to which voltage regulation application must be applied. In effect, the solar panel charging contribution design will be factored by the necessary standards of available solar modules that meet the constraints of the hardware design. When applying additional charge to the power supply battery via solar paneling contribution, the effectiveness of the resulting solar output must be considered. Solar cell efficiency is most commonly measured in energy conversion efficiency, which yields the percentage of solar photons converted into actual contributing electrical power, and is largely considered for application.

3.2.6.1 Efficiency Versus Cost

The energy conversion efficiency rating of the solar module is an important value to consider when applying a solar-fueled charging contribution, but a higher rating product does not mean that it is commercially realistic, nor the most feasible for the project. The highest recorded efficiency for a solar cell has been 44.7%, with similarly rated cell efficiencies existing, but these cells are not practical in the consumer market, and are often constrained to laboratory research, due to being made from more costly multi-junction sub-cells that require sunlight magnifications. More realistically, crystalline silicon photovoltaic cells are the prevalent product for the solar power market, with monocrystalline, polycrystalline, and thin film silicon panels being the dominating technology. The efficiency ratings of these three materials vary, as does their cost per cell/panel.

Monocrystalline Cells

Monocrystalline silicon solar cells are comprised of single-crystal silicon material with an average efficiency rating of 20%. The appeal to monocrystalline cells is that they are the most commercially efficient panels in circulation, and are capable of performing better in indirect sunlight, such as overcast weather conditions, compared to the polycrystalline and thin film counterparts. The drawback to monocrystalline solar cells is the cost in exchange for better efficiency, and the fragility of the panels themselves.

Polycrystalline Cells

Polycrystalline solar cells are the most commonly, commercially produced solar cells, with a correspondingly lower price and efficiency than monocrystalline. The average efficiency rating for polycrystalline cells is 15%, a value in-step with most widely available solar paneling products. Polycrystalline cells are ideally preferred for small-scale projects with a limited budget, but, like monocrystalline cells, they are fragile in application, and the lower efficiency make them a factor for usefulness in the power contribution.

Thin Film Panels

Thin film solar panels boast the lowest cost per watt in terms of all three materials, but also record the lowest efficiency rating, varying on the specific material used. Thin film panels are commonly made from three types of material. Amorphous Silicon panels have a 8% efficiency rating and have a low manufacturing cost, making it a competitive product. It also can be produced in specified surface dimensions, which is better for design layout. Cadmium Telluride (CdTe) panels are significant in that the efficiency ratings can be as high as 15%, matching the efficiency of crystalline silicon, while maintaining a significantly lower production cost. However, most commercial efficiency ratings

for CdTe are closer to 10%, and a primary concern for development is that cadmium is extremely toxic, and CdTe film panels can potentially be an environmental concern. Copper Indium Gallium DiSelenide (CIGS) film panels are the next growing material for thin film solar panels, with experimental efficiencies reaching as high as 20%, while most commercial ratings hover around 11%. CIGS panels use considerably less cadmium levels, and have greater heat resistances, but struggle to be price competitive compared to amorphous silicon and CdTe films.

3.2.6.2 Solar Cell Selection

Effectively for FBC, the solar paneling design serves primarily as a means to recharge the power source battery, or, alternatively, extend the available power time to the device while in play. Therefore, the solar panel contribution is not specifically designed as the primary source of battery charge, as assumed by the external power charge plug-in, but rather as the auxiliary contribution to an extended alternative. With this in consideration, for supplying additional charge to the Lithium Polymer battery, we look at four separate models of solar cell technology, two monocrystalline cells, one polycrystalline cell, and one amorphous silicon thin film, as shown in Table 3.2.6.1.

Model	Output Current	Efficiency	Dimensions	Cost (single quantity)
IXYS SLMD481H08L	200 mA	22%	89 x 55 mm	\$31.91
IXYS SLMD121H8L	50 mA	22%	86 x 14 mm	\$10.23
P-MAXX- Series Polycrystalline	400 mA	14.8%	80.43 x 19 mm	\$1.99
Panasonic – BSG AM-1417CA	13.5 μ A	Varies	35 x 13.9 mm	\$2.40

Table 3.2.6.1 – Solar Panel Comparisons

By comparison, the monocrystalline cells yield substantially higher efficiency ratings, and significantly higher costs. The thin-film Panasonic-BSG outputs a marginally lower current, not suitable for charging the FBC lithium polymer battery. The P-MAXX polycrystalline cell supplies the largest output current, which is ideal for the time taken to charge the source battery to full capacity, and costs considerably less. However, its limited efficiency raises concerns for implementation on successfully charging the battery in a limited forecast environment, making it unsuitable for an effective handheld design.

Between the two monocrystalline panels, the case size factors into the importance given to the solar cell dimensions. Based on the design constraints, the FBC is designated to have an exterior case area range similar to that of the internal Raspberry Pi2 and the LCD Composite screen of 85 x 56 millimeters (mm), giving approximately the equal amount of space available for the solar cells to be paneled on the back of the case. The SLMD481H08L has a dimension that would closely match the surface area of the case, allowing for the need of only one cell module, with a favorable output current of 200 mA. Likewise, the SLMD121H8L monocrystalline solar cell can match the surface area and 200mA current by paneling four cells lengthwise, and connecting them in parallel, which would sum of the four 50 mA to the near same output current of the single SLMD481H08L.

The cost discrepancy is that four of the SLMD121H8L would be significantly more expensive, but the tradeoff is the benefit of replaceable panels. Should one of the SLMD121H8L get damaged, it can be replaced, while the other three remain functional and can be kept. With the larger, single cell SLMD481H08L, if the panel were to be damaged, the entire solar contribution would need to be replaced. Additionally, the four separate SLMD121H8L panels give way for more maneuverability with the positioning on the device, a flexible attribute to consider.

After reviewing the variety of solar paneling options, we conclude that the monocrystalline design would best serve as the choice technology for the FBC, given its higher efficiency rating in relation to the contribution to extending the power charge of the source lithium polymer battery. Although a lower efficiency rating would be more feasible, a larger surface area would be needed to make up the same electrical charging energy, and for a small, handheld device, such as the FBC, that is not a practical option. In terms of comparing cost and size, we conclude that the IXYS SLMD121H8L solar cell would be the best option for the case design, as well as meeting the specifications for sufficiently charging the device. Below, in Figure 3.2.6.1, is the model part SLMD121H8L that we will use.

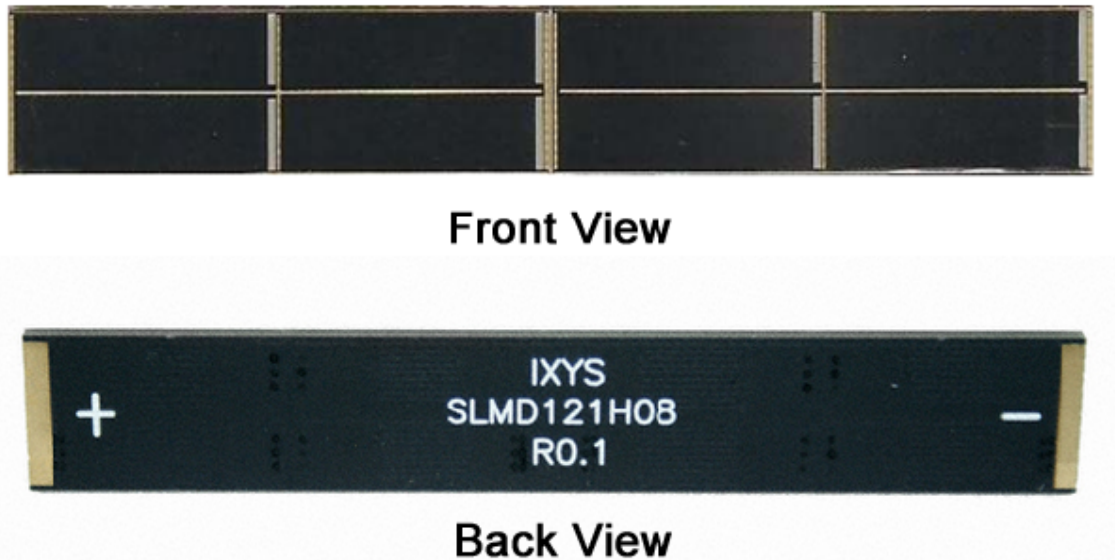


Figure 3.2.6.1 - SLMD121H8L Solar Cell

Additionally, in Figure 3.2.6.2, we have the solar cell size dimensions for perspective of exterior case design.

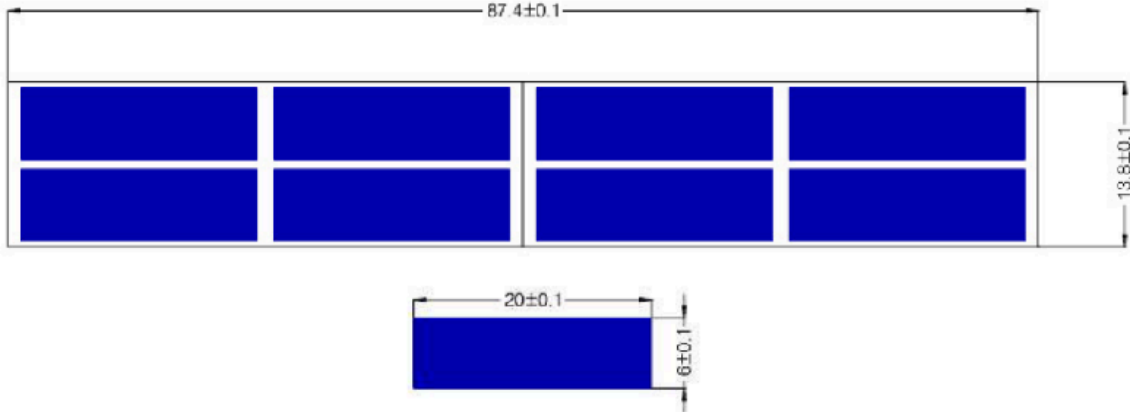


Figure 3.2.6.2 – SLMD121H8L Solar Cell Dimensions

3.2.6.3 Solar Overcharge Protection

One concern to address with the addition of the solar panel circuit to the battery power source is to regulate the voltage discrepancies and avoid current leakage. When the solar panel charger is not being utilized, or in direct sunlight, we do not want reverse current contribution from the battery to flow back into the solar cells. The solar circuit essentially acts as a secondary charger to the FBC, and will use a similar charge protection IC for charging the battery. Additionally, we need to

monitor the solar cell's input voltage contribution to manage supplied charge current, otherwise the solar charger's voltage will collapse when it drops from maximum open-circuit levels. Another concern is that charging our lithium polymer source battery requires constant current and constant voltage from the charger, to avoid battery damage. The solar panel has widely variable voltage and current supply, and a charging IC must be needed. For consideration, we consider three charging IC models, both designated for lithium battery charging, and ideal by means of solar energy.

LT3652

The LT3652 is a 12-pin lithium battery-charging controller, with charge switch protection embedded in it. The charge controller can take a minimum charge voltage of 4.92V and a maximum of 32V, although this is unnecessarily high, given our low power system. The LT3652 produces constant voltage and current to the charging destination, and has a built-in output charge detection to terminate input voltage when charging is complete. This charge controller is ideal for the very use of charging a lithium battery via solar cells, but is intended for a much larger contribution scale.

bq24210

The bq24210 is a 10-pin charge controller, with an input charge voltage range of 3.5V to 18V. Similarly, it allows for a constant voltage and current to be supplied to the charging battery, and specifically caters to lithium ion battery charging circuits. The bq24210 also offers battery charge level detections to signal the controller's status, thus indicating if the battery currently needs to be charged at all, if already at maximum voltage ratings. This model, as with the LT3652, is specifically designed for solar circuits, being ideal for ensuring protection by limiting leakage current through the input pins.

SPV1040

The SPV1040 is an 8-pin, low power charge controller, with input voltage ranging from 0.3V to 5.5V. It provides constant voltage and current to the charging battery, and includes protection from discharge at low voltage levels. One drawback is that this controller is low-powered, and is designed for smaller battery supply contributions, creating concern for conflicting shutdown conditions while charging.

3.2.6.4 Solar Technology Constraints

One of the significantly contributing constraints to renewable energy technology, especially solar energy, is efficiency. Even higher-grade commercial monocrystalline silicon solar cells have typically an efficiency rating of approximately only 20%. While efficiency ratings can certainly reach much

higher, up to about 44%, such values are only reasonable in carefully controlled environments, typically in research laboratories. However, the means to achieve such ratings also include much more costly multi-lens technology for solar magnification, and are not yet practical for public manufacturing. Although lower efficiency ratings are acceptable, and much more cost-efficient, the trade-off is the time required to make up for such energy losses. A 22% efficiency solar cell will take twice as long to produce the same amount of usable electrical energy as would a 44% solar cell. Thus, even cheaper and less efficient polycrystalline cells will take even longer. The constraint is factoring in how long it can take to produce the desired electrical energy to meet product specifications.

One solution would be to simply apply more solar cells in the paneling circuit, with connections in parallel or series designed as needed for achievable voltage or current levels. However, cost and space can be debilitating factors. Depending on the application, surface area for solar paneling can be limited. In the case of the FBC, being a portable handheld device, the surface space for additionally solar cells is extremely limited. Thin film solar cells, which can wrap around the device and are more flexible to design, are a realistic solution. However, this technology is the most limited in output efficiency, and can also be potentially harmful to the environment, pending on the material used. Additionally, designing a panel with more solar cells can be costly, with the marginal benefit of added power production needing to outweigh the marginal cost of purchasing more cells. Such cost considerations were factored into choosing the SLMD121H8L cells for the auxiliary charging circuit.

3.2.7 Audio

The cornerstone attribute of the FBC is to recapture retro video games in a modern sense, and a hallmark trait of the Super Nintendo Entertainment System (SNES) is the vintage gaming music. In original hardware production, gaming systems, such as the SNES, used audio sound chips that were separate from the main system, and relied on five audio wave channels (Pulse Wave1, Pulse Wave2, Triangular, Noise, and Delta Modulated Samples) to accomplish the precise tempo, pitch, and other effects for the soundtrack composition. Given that FBC will be supporting SNES games, it is important to compare the original SNES audio to the FBC's Emulation Station for accessing the audio sample codes from the game ROM. The SNES audio system was the Nintendo S-SMP, which operated at 2.048 MHz and performed from an 8-bit CPU core, the Sony SPC700. The S-DSP is credited for mixing and creating the actual sounds and audio, yielding the unique and iconic music samples based off of all five audio channels. By comparison, the FBC software, Emulation Station, accesses the same audio samples from the game ROM, and transmits them via the 3.5mm audio jack to the system stereo speakers. These emulated files arguably lack the distinct modulated waves and noise compositions achieved by the S-DSP, as well as prove to have an audio speed discrepancy from the emulated SPC700 on Emulation Station. However, efforts to incorporate the original S-SMP sound chip

to the FBC’s microcontroller proved excessively challenging, and overstepped the allotted size constraints of the casing. Therefore, it was decided to work past the potential audio shortcomings of the FBC emulation in favor of reduced size and power.

3.2.7.1 Stereo Output

The primary choice for audio output from the FBC is decided between monophonic or stereophonic speaker design. The Raspberry Pi 2 core audio output lends itself to two audio channel pins, allowing the potential for either speaker design. Monophonic audio is generated through a single channel output, and only needs one audio speaker to transmit sound, a design classic with original handheld consoles, such as the Nintendo Gameboy Color. Using both audio inputs for two separate speakers, stereophonic output can be achieved. Stereophonic sound allows more depth and surround-sound output, generating a more perspective approach to the system user. Therefore, the stereophonic two-speaker design will be used.

Amplifiers

Standard audio output speaker sizes in portable handheld consoles range 22 mm to 29 mm in diameter, and have an output impedance of 8 ohms. Given this output impedance, the speakers will need an amplifier circuit to make them functional with the Raspberry Pi 2’s line level high impedance. The TDA2822M 8-pin amplifier offers dual-speaker stereo capabilities while limiting current drawn from the device and keeping a power voltage within the 5volt supply. Comparatively, TS4984 amplifier offers a 16-pin multi-stereo option, and comes with a much larger supply current. LM4880, on the other hand, is another 8-pin stereo capable amplifier, but with lower required current draw than the TS4984. In Table 3.2.7.1, we compare each of the three audio amplifier models for consideration.

Model	Supply Current	Supply Voltage	Power Output
TDA2822M	6 mA	1.8 V to 15 V	300 mW
TS4984	7.4 mA	2.2 V to 5.5 V	1 W
LM4880	3.6 mA	2.7 V to 5.5 V	250 mW

Table 3.2.7.1 – Amplifier Comparison

Given the desire for a low current draw from the power supply contribution, we chose the LM4880 model for our audio output amplifier. The LM4880 uses the minimum 8 pins to use two magnetic speakers for the stereo output, while handling the 8-ohm impedance load. The supply voltage also allows for direct power input from the system power supply. Additionally, we chose the LM4880 based on compatibility review with multiple controller platforms, including the core Raspberry Pi 2. To note, we will require the use of two LM4880 audio

amplifiers for implementation, with one being used for external speaker output, and another one for direct audio transmission from the Raspberry Pi2 to auxiliary plug-in headphones.

3.2.7.2 Volume Control

Audio control levels need to be established for user application in selecting the desired volume output of the device, or to completely mute the audio all together. The most common method for device volume control would be through the use of a voltage-dividing, resistive-setting analog potentiometer. When the user slides the potentiometer bar or wheel, the level of resistance is set to drive the system's voltage to ground for muted audio, or to the maximum rated voltage for allowable amplitude levels.

Linear Versus Logarithmic Control

Setting the correct resistance for the desired audio levels is achieved by moving the analog potentiometer bar or knob in the direction indicated of higher, or lower resistance. However, the concept of changing audio at acceptable rates factors in the amplitude level change as detected in decibels by the human ear. The two most common potentiometers for volume control are linear and logarithmic rated potentiometers. Linear potentiometers change the output audio response on a linear scale, with the change of resistance proportional to the extent of which the potentiometer has been moved. Logarithmic potentiometers change output audio response on a logarithmic scale, with an initially steady change in volume levels increasing at a faster rate near the approaching power limit for maximum value.

For user consideration, the logarithmic potentiometer is the preferred design based on the human ear reception. The human ear is sensitive to amplitude changes on the logarithmic scale, and audio signals are better responded to gradual changes, instead of linear levels. However, many available potentiometers, especially models common in handheld devices, are defaulted as linearly changing controllers. The easiest way around this is the addition of a series resistor to divide the rate of voltage change. Figure 3.2.7.1 below compares the linear and logarithmic potentiometer levels, as well as the presence of the series resistor to toggle the output rate.

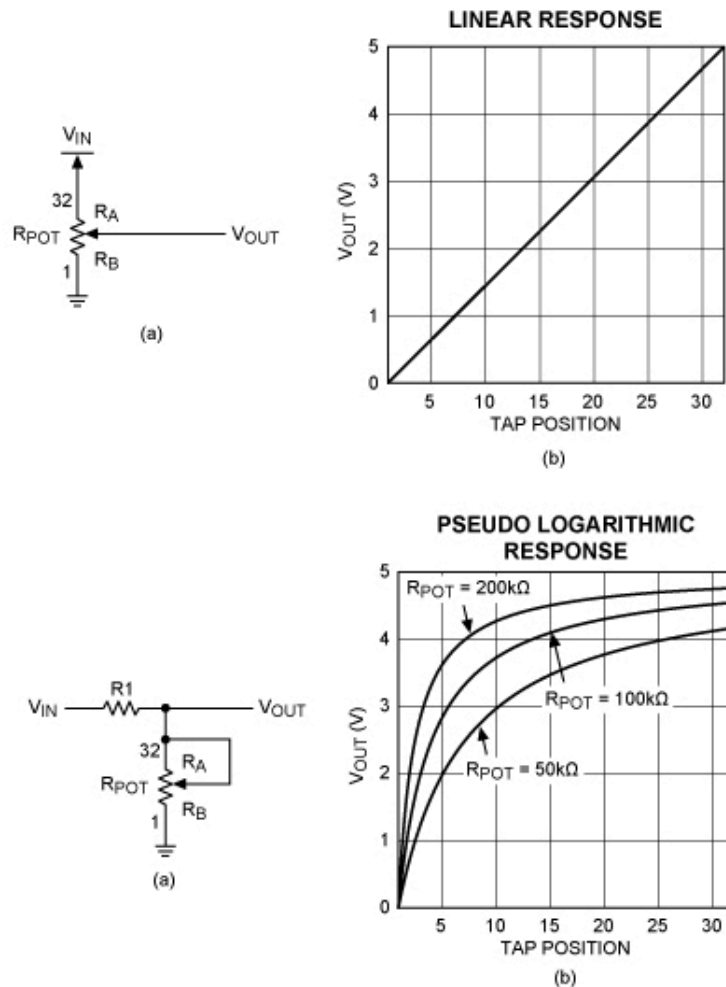


Figure 3.2.7.1 - Linear and Logarithmic Potentiometer Responses (Permission Pending)

Analog Potentiometers

The most common, and simpler, potentiometer design found in popular handheld gaming devices is the analog models. Analog potentiometers are three-pin electro-mechanical devices that can adjust the output amplitude signals by increasing or decreasing their internal resistance to permit anywhere between maximum allowable voltage to zero voltage output. This change in resistance is directly commanded by the user, and is usually executed by means of an external slide bar, a thumbwheel, or a rotary knob. Typically, analog potentiometers alone function as linear audio output devices, and need an added resistor voltage divider in series to produce a more logarithmic change in volume, as discussed above. The clearest advantage for analog potentiometers is that they, unlike digital, require no additional voltage source to operate; they simply

regulate the output voltage to the speakers at the user's settings. Additionally, analog models are useful in that the resistance set by the user is maintained, even when the device is powered down, and then powered back on again. However, analog potentiometers are limited in precision, and over larger mechanical aspects to factor into case design restraints.

Digital Potentiometers

As opposed to analog potentiometers, which are directly adjusted by user interface, digital potentiometers can be used in an integrated circuit for better precision. Digital potentiometers are programmable microcontroller-interfaced components that feature an internal mechanical potentiometer that can be adjusted by input pin setting signals, such as a push-button command. Using multi-resistor ladder integrated circuits, as shown in Figure 3.2.7.2, the digital potentiometer uses a wiper switch connection to corresponding resistance levels to control the output current to the stereo system. The user push-button controls act as the mechanical turn knob or wheel, with each button input sending a status signal to the next available wiper register. When the wiper register is set to the corresponding "close" signal, the switch to the matching resistor closes, increasing the resistance that the potentiometer give out.

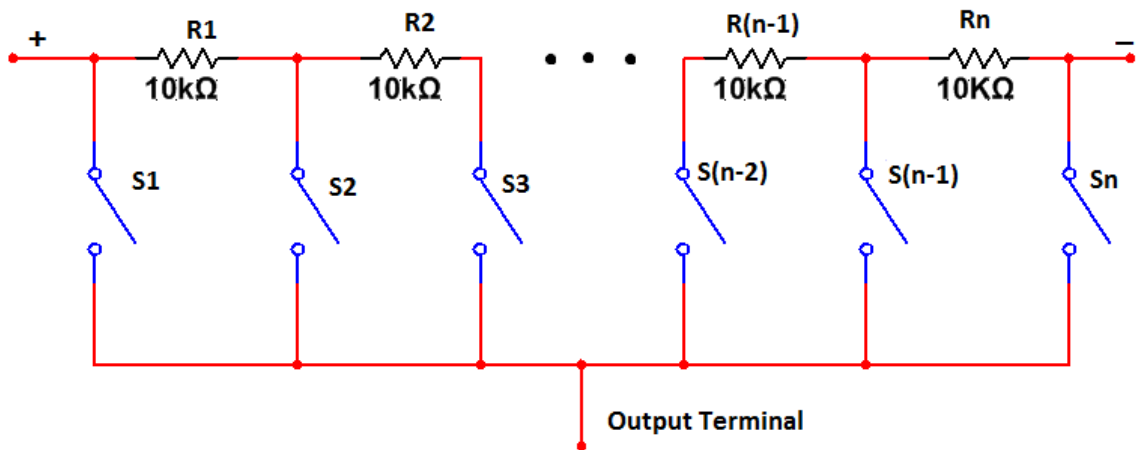


Figure 3.2.7.2 - Resistor Ladder Digital Potentiometer Circuit

Initially, this ladder resistance stacking only offers a linear response, which, as discussed above, is undesirable. However, many recent digital potentiometers, such as the Microchip AN1080, can be programmed to have a logarithmic characteristic by simultaneously closing multiple wipers in the ladder circuit, which will change the audio output for a more gradual response. This creates a digital to analog converter (DAC) feature that can smooth out the audio output. This is demonstrated in Figure 3.2.7.3, where we see that as the slider bar is moved from lowest position (maximum resistance) to highest position (minimum resistance), the output voltage increases on a linear rate. However, the decibel

level of audio output has a logarithmic curve at the same slide bar directions. Digital potentiometers can offer better precision and can be programmed to scale in correspondence to the desired change of volume output. One challenge with the digital versions, however, is that when the system is powered down, the potentiometer does not save the last input resistance value on it, and usually defaults to a median resistance. Additionally, the digital models typically require an input voltage to operate, and likely a DC-to-DC converter for the output voltage from the connecting amplifier circuit.

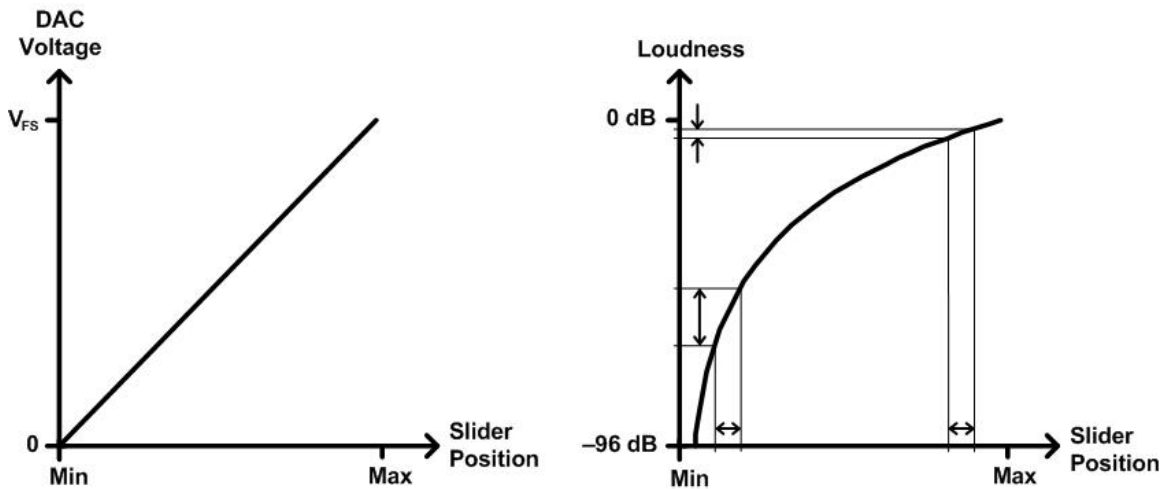


Figure 3.2.7.3 - Digital Potentiometer Using DAC to Create Logarithmic Output

Potentiometer Model Selection

Potentiometer model selection came down to deciding between linear or logarithmic, analog or digital, and finally the model designs available. Clearly, the logarithmic design will be used for the FBC, as linear volume control will have too harsh of an effect on users, and will significantly lower the gaming experience quality. Both analog and digital potentiometer designs are initially set to linear controls, with the analog potentiometer acting as a logarithmic through the use of a series resistor, and the digital model being digitally configured to adjust resistance intervals in a logarithmic simulation.

Analog and digital potentiometers can both be appropriately functional for the FBC, with the advantages and disadvantages considered within the design constraints. The analog design can be accomplished by direct user interface, and can maintain the designated volume setting, even through power phases. Additionally, with the analog mechanical volume control used, the user can visibly see the potentiometer setting and know what output volume to expect. Digital potentiometers offer better audio precision, and can be logarithmically programmed to change volume at a custom ideal rate. The lack of significant mechanical components also makes the digital design a more likely candidate for size-restricted projects. Both designs also have drawbacks, with the analog

taking up significant space, and the digital being power dependent. When deciding between the two, we considered the power consumption and size availability factors of the FBC. As with many handheld, portable gaming devices, audio level precision is not a significant concern, as long as the output transitions on a logarithmic scale. For those considerations, we choose the more traditional analog potentiometer design.

The three most commonly used mechanical potentiometer designs are the slide bar, thumbwheel, and rotary models. The slide bar design adjusts the applied resistance by the user sliding an externally exposed bar linearly up and down a track. Visually, the user can use the position of the slide bar in relation to the maximum and minimum ranges to expect the current volume output. The thumbwheel model functions via a semi-exposed disk that can be rotated at an angle generally between 270° and 300°, and is generally numbered from “0” through “10” on the outer perimeter. Each number represents the volume level of the device, with “0” equating to a ‘muted’ output, and “10” meaning maximum volume at the highest output voltage allowed. Despite the linear increase numerically, the number rating represents sound levels on the logarithmic scale. The rotary potentiometer adjusts volume via a protruding knob that, like the thumbwheel, can be rotated along a 270°-300° angle. The rotary design is discounted due to its functionality requiring a significant knob protrusion. The volume control mechanism to the FBC is specified to be operated with a single finger from the user, and the turn-knob design of a rotary potentiometer requires more effort, making it the more inconvenient model.

Between the slide bar and thumbwheel designs, we choose the thumbwheel. The slide bar covers more exterior surface area due to its slide track, and can be prone to more unintended motion by the user when handling the device. Additionally, the slide bar can very easily go from minimum, muted output to maximum volume in a single swipe, which can be detrimental if accidentally moved. The thumbwheel, on the other hand, can be more carefully adjusted, is less prone to significant unintended volume change, and requires less linear spacing for operation. The standard impedance rating for potentiometers in low-voltage small devices is typically 10 Kilo-ohms, and the ideal thumbwheel diameter for a handheld design is approximately 9-12 mm. Variations between company models does not significantly alter the intended functionality for the audio caption, and based on current availability, we choose the Bourns 3352T-103LF-ND thumbwheel potentiometer. The model and dimensions are shown here, in Figure 3.2.7.4. Included in the diagram is also a demonstration of the rotation of the thumbwheel in relation to the internal resistance wiper. Starting with the maximum 10 Kilo-ohm resistance, at the thumbwheel positioned at the furthest counterclockwise angle, the volume is “muted”. As the thumbwheel is turned in the clockwise direction, the wiper resistance is decreased as angle rotation increases. When at the highest possible clockwise setting, the wiper resistance is zero, and maximum voltage is outputted through the audio speaker or jack systems.

3352T

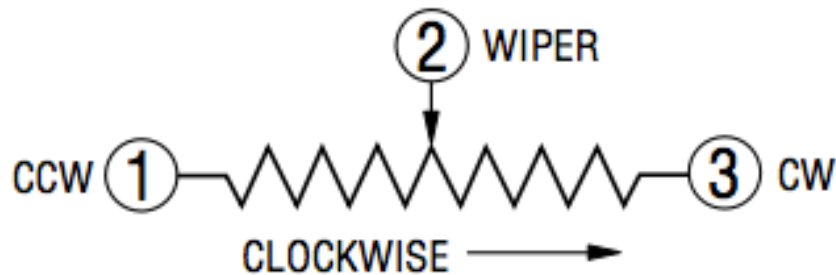
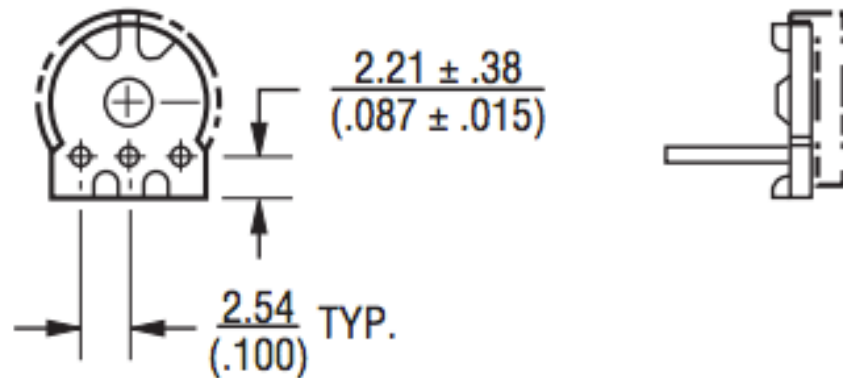


Figure 3.2.7.4 - 3352T Potentiometer Diagram

3.2.8 Power System

Our power system, shown in Figure 3.2.8.1, is one of the most important parts of our console. The power system needs to adequately apply power to the different components, while also supporting prolonged device usage away from a wall outlet. One of our requirements was that the battery in our system would be able to charge from a standard USB wall adapter or computer outlet. The 5 volts from the USB will be put through a charging circuit specific to our battery that will ensure our battery safely charges. The battery will also be able to receive a small amount of charge via the solar panels attached to the case. The workhorse of our system is the Raspberry Pi 2 which runs off of 5 volts through USB. That means our power subsystem needs to output 5 volts. Since none of the common types of batteries will produce a constant 5-volt output we need to regulate the output. The output of the battery will be put through a DC-to-DC conversion circuit and regulated to the 5 volts needed by the Raspberry Pi 2. The Raspberry Pi 2 has a regulated 5-volt output built and a regulated 3.3 volt output built in. These outputs will be used to power the various components of the mainboard that will interact with the Raspberry Pi 2. The first thing we need to do is choose a battery to

determine how we can charge it and what kind of DC-to-DC conversion circuit we need to make.

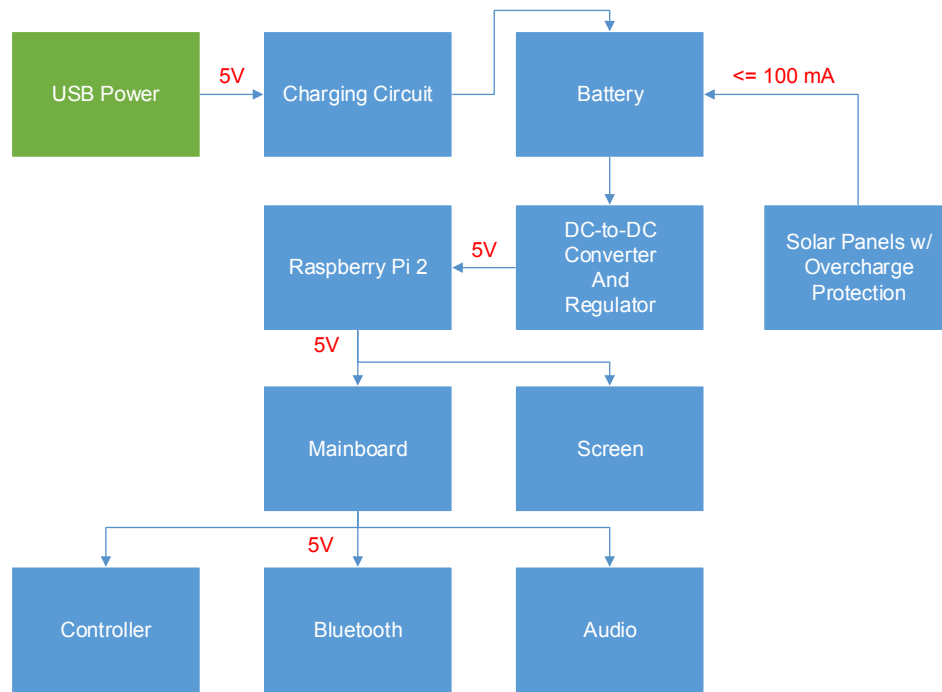


Figure 3.2.8.1: Power Subsystem

3.2.8.1 Battery Selection

Due to our system being a handheld game console, we determined that power to our system would be supplied by a battery in order to make the system portable. Choosing an appropriate battery for a portable system is very important. We compared the positives and negatives of five different types of batteries before settling on the type of battery we are going to use. The five types of batteries we considered were standard alkaline batteries, nickel-cadmium (NiCd) batteries, nickel metal hydride (NiMH) batteries, lithium-ion (Li-ion) batteries, and lithium polymer (LiPo) batteries. We compared the batteries focusing mainly on size, weight, capacity, and longevity to try and keep the system small and running for extended periods of time, and to have the battery last for a lot of charge cycles. Other battery attributes that factored into our decision were output voltage, charging methods, safety, and environmental impact.

Alkaline

We decided against using alkaline batteries pretty early on so not a lot of research was done for them. During discussion we determined that the cost of constantly replacing alkaline batteries was prohibitive. Another factor that played into our decision not to use alkaline batteries is that they take up more space

than other batteries while having a lower capacity and output voltage than what we were looking for. Ultimately we decided that alkaline batteries were not the appropriate battery for our device.

Nickel-Cadmium (NiCd)

Nickel-cadmium batteries were one of the first types of rechargeable batteries. The term nickel-cadmium comes from the elements that make up NiCd batteries, nickel oxide hydroxide (NiOOH) and cadmium (Cd). While they were very widely used in the past they have seen a drastic decrease in use in recent years due to the emergence of lithium-ion and nickel metal hydride batteries. We decided NiCd batteries were worth looking into as they are cheap and still used in some devices and are very heavily used in the RC community.

First we took a look at the primary factors of comparison. NiCd batteries are available in standard sizes as well as battery packs with multiple battery cells. NiCd batteries are also heavier than other batteries. Ideally we wanted a smaller battery, but this was not enough of a deterrent for us not to take a closer look at NiCd batteries. The largest capacity AA sized NiCd battery I could find was 1000 mAh. This is significantly lower than some of the other battery options that were considered. We would need five AA NiCd batteries to come close to reach the capacity of a 5000 mAh lithium-ion battery (which is about the size of two AA batteries) or lithium polymer battery. NiCd batteries last for quite a few charge cycles. While this would normally be a good thing, NiCd batteries also suffer from something called a memory effect. This means that NiCd batteries will hold less of a charge over time if not completely discharged each time. This is a huge negative for us due to the fact that we would not want our game system to have to die before it can be charged, and also because we will have a solar power unit that will constantly be charging the battery.

The nominal output voltage of a single NiCd battery cell is 1.2 volts. The actual voltage can range anywhere from 1.35 volts to 1.1 volts depending on where the battery is in the discharge process. NiCd batteries have a decently flat discharge curve so the vast majority of the time they operate at around 1.2 volts. While this is certainly useful in many situations, our battery output voltage will be fed through a dc-to-dc converter and regulated to supply the needed voltage for our various components. It actually might be slightly detrimental to have such a flat discharge curve because of the battery level indicator on our device. Figure 3.2.8.2 shows the discharge curve of a single NiCd battery cell. The figure shows that between 90% and 5% battery remaining there is only a 100 millivolt change in voltage.

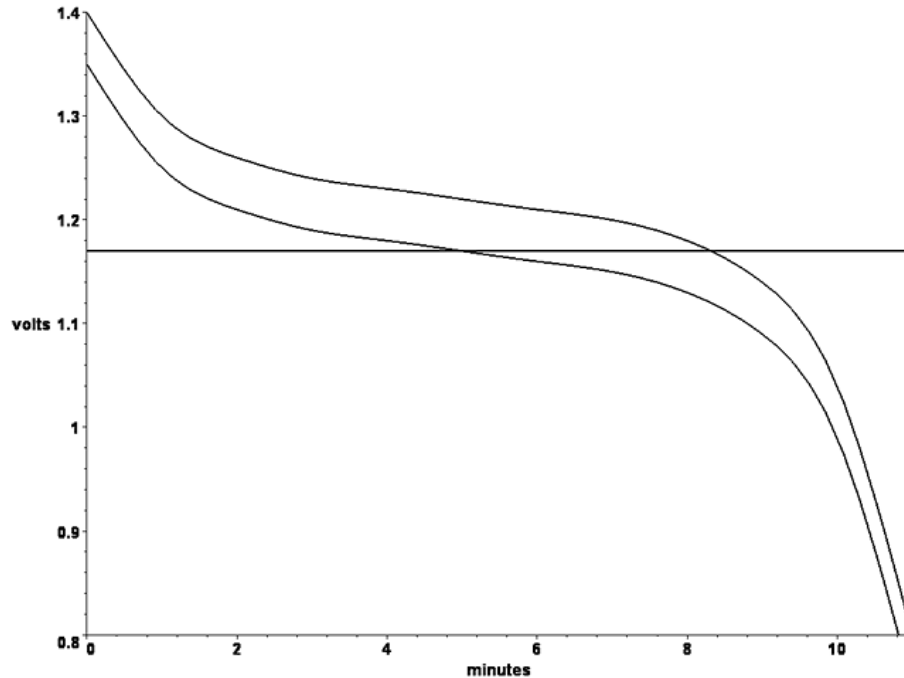


Figure 3.2.8.2: Discharge Curve of a Normal Nickel-Cadmium Battery and One Suffering from the Memory Effect (Permission Pending)

Charging a NiCd battery consists of supplying it with a steady current of 10% of its capacity for fourteen to sixteen hours. The voltage supplied by the charger varies with temperature, but not by much. Charging efficiency is never 100% and batteries tend to have a higher capacity than listed so charging longer than ten hours is required. Overcharging is a possible issue with NiCd batteries because they do not change in voltage by much, so a charger will have a harder time determining when a NiCd battery has reached full charge based on the battery's voltage. Another potential downside to NiCd batteries is the self-discharge rate. NiCd batteries will discharge 10-20% per month when not under any load. While not a massive detriment to us, we would like to be able to pick up the console after not using it for a few months and still have some charge remaining in the battery.

NiCd batteries are both safe and durable. They can be fully discharged safely unlike other battery types. NiCd batteries do contain cadmium, though, which is a toxic metal. NiCd batteries require special disposal and have a negative impact on the environment if not disposed of properly. While we do not worry about having to change out the battery anytime in the near future, we would prefer to have a battery in our device that does not have a material that is hazardous to the environment.

Nickel Metal Hydride (NiMH)

Nickel metal hydride batteries are significantly newer than NiCd batteries and have been improved upon significantly in relatively recent years. Like NiCd batteries they contain nickel oxide hydroxide (NiOOH), but in place of cadmium it has an alloy that joins with a hydrogen atom during the charging process to form a metal hydride (MH). NiMH batteries have replaced NiCd batteries in many applications. We decided to look into NiMH batteries because they are very similar to NiCd batteries, but they tend to have more charge capacity while being just as cheap if not cheaper.

Like NiCd batteries, NiMH batteries are available in standard sizes and battery packs with multiple cells. NiMH batteries are lighter than their NiCd counterparts while having a larger capacity. The largest capacity AA sized NiMH battery we found was 2600 mAh. This was a significant improvement over the capacity of the NiCd batteries we saw. We would only need two of those batteries to reach the same 5000 mAh threshold we set for NiCd batteries which makes it roughly similar in size to lithium-ion batteries. Both of those attributes were huge positives towards the efforts to make our console very portable. There are downsides to using multiple NiMH batteries simultaneously, though. The different cells may discharge at different rates, so charging and discharging them can become more complex. Testing each cell individually is a pretty big hassle. The amount of charge cycles that a NiMH battery lasts seems to fluctuate quite a bit based on the current drawn. They seem to last for several hundred charge cycles with normal use, which suits our needs. NiMH batteries also do not suffer from the memory affect that plagues NiCd batteries.

Again, like NiCd batteries the nominal output voltage of a single NiMH battery cell is 1.2 volts. The actual voltage can range from around 1.4 volts to 1 volt depending on where the battery is in the discharge process. The discharge curve of NiMH batteries is very similar to NiCd batteries. This means that the vast majority of the time NiMH cells operate at around 1.2 volts. The same positives and negatives apply here as they did for the NiCd battery. The flat discharge curve is a detriment to our battery level indicator and the near constant voltage is unnecessary for our device components. Figure 3.2.8.3 shows both the charge and discharge curve of a single NiMH battery cell. The curve is incredibly similar to the discharge curve of a NiCd battery. Many of the websites I visited while gathering information said the discharge curves of NiCd and NiMH batteries were basically interchangeable.

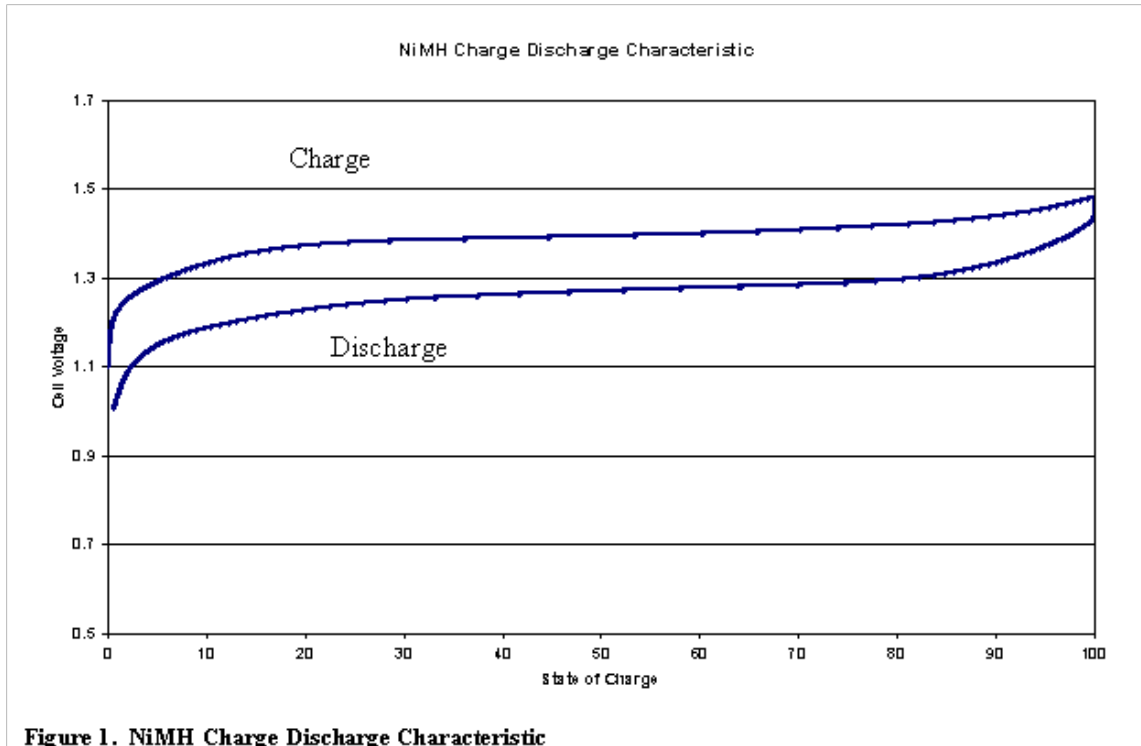


Figure 3.2.8.3: Charge and Discharge Curve of a Nickel Metal Hydride Battery (Permission Pending)

Charging a NiMH battery is often similar to charging a NiCd battery. The same 10% of its capacity for around fourteen hours is supplied to the battery. The voltage supplied is between 1.4 and 1.6 volts. Similarly to NiCd batteries, NiMH batteries need to worry about overcharging because of the constant current supplied to the battery by the charger. NiMH batteries have a very large self-discharge rate in comparison to other batteries. NiMH batteries self-discharge at about 30% a month when not under any load. This by itself is not terrible, but what makes it particularly bad for our implementation is that the self-discharge is not linear. The battery can self-discharge up to 20% of its capacity in the first day alone. This is not ideal for a portable console that aims to be taken on long drives and flights.

NiMH batteries are also safe and durable. They are used as batteries for many power tools much in the same manner as NiCd batteries. NiMH batteries are also very environmentally friendly. They do not contain the cadmium that makes NiCd batteries so toxic. For this reason and a few others nickel metal hydride batteries are a solid choice. If we had to choose between NiCd batteries or NiMH batteries we would choose NiMH, but we still considered two other types of batteries.

Lithium-ion (Li-ion)

Lithium-ion batteries did not see commercial use until the 1990s. They are some of the newest batteries available and they are more expensive than the previously discussed types of batteries. Li-ion batteries get their name from the lithium ions that move from the anode to the cathode while the battery is discharging and from the cathode to the anode when the battery is charging. The advent of Li-ion batteries has had a huge effect on portable electronics. Li-ion batteries are used widely in consumer electronics and are smaller than both NiCd and NiMH batteries. Most modern phones and portable devices use Li-ion batteries.

Li-ion batteries are available in all the standard sizes, but also come in different sizes and shapes. The most notable shapes are cylinders and flat/rectangular like a cell phone battery. They are lighter than both NiCd and NiMH batteries, and have a similar capacity to NiMH batteries. The largest capacity I could find in AA size was 800 mAh, but we would be using a slightly larger sized Li-ion battery that can hold up to a 3400 mAh charge on a single cell. The lightweight and high capacity properties of the li-ion batteries were a huge selling point for us. The amount of charge cycles that a li-ion battery lasts is similar to a NiMH battery, generally in the range of 300 full cycles. Li-ion cells have no memory effect whatsoever.

The nominal output voltage of a single Li-ion cell is 3.7 volts. The actual voltage output of a Li-ion battery ranges from around 4.2 volts when charging should be stopped down to 2.5 volts when discharge should be absolutely stopped. Ideally you should stop discharge around 3 volts. A Li-ion battery should never be discharged below a certain voltage. When Li-ion batteries drop below or raise above certain voltage thresholds irreversible chemical reactions take place that ruin the battery and can cause fires and explosions. The discharge curve of li-ion batteries is different from the nearly flat curve of NiCd and NiMH batteries. It will be easier for us to design a battery level indicator around a battery with more fluctuation in its voltage. The voltage is going to be regulated so the fluctuations in voltage from the Li-ion battery are a non-issue. Figure 3.2.8.4 shows the discharge curve for a single Li-ion cell at a constant current of 20% of its charge capacity and the discharge curve of a single NiCd or NiMH cell at 20% of its charge capacity. Clearly the voltage in the Li-ion discharge curve fluctuates far more than the voltage in the nickel oxide hydroxide based batteries we have discussed.

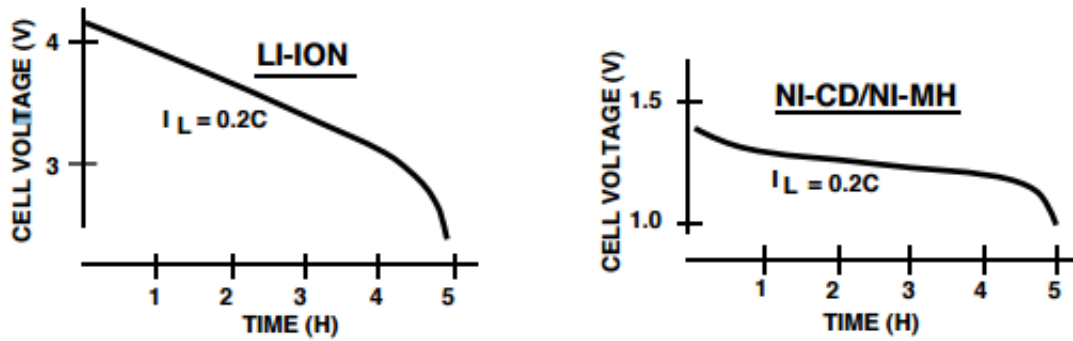


FIGURE 3. CELL DISCHARGE CURVE

Figure 3.2.8.4: Discharge Curve of a Li-ion Cell VS Discharge Curve of a NiCd/NiMH Cell (Permission Pending)

Charging a Li-ion battery is more complex than charging a NiCd or NiMH battery. Instead of just supplying a constant current the whole time, a constant voltage is also applied towards the end of the charging process. A constant current is supplied to the Li-ion battery based on the Li-ion battery and the charger that is used. The voltage output of the charger will increase as the charge in the cell increases and the current remains constant. When the maximum charge voltage (generally around 4.2 volts) is reached, a constant voltage equal to the maximum charge voltage is applied to the Li-ion battery. The current supplied by the charger will decrease until reaching a set value when the charger will stop charging the battery. This process is faster than the “trickle charging” method of the nickel batteries. As stated before, overcharging a Li-ion battery is dangerous and can cause explosions, so strictly following the maximum and minimum voltage values and the maximum current of a Li-ion battery when charging is very important. Li-ion batteries, unlike their nickel based counterparts, have a very low self-discharge rate. Generally Li-ion batteries only self-discharge at a rate of around 2% a month. Such a small self-discharge rate is ideal for a console that you might forget about for a few months and then want to play immediately. A Li-ion battery should also not be allowed to self-discharge below a safe voltage.

Li-ion batteries are less safe and durable than NiCd and NiMH batteries. While they are used in a lot of consumer electronics, they are not generally used in heavy duty applications. While safety is a concern with Li-ion batteries, when handled properly it is easy to use Li-ion batteries safely. Some Li-ion battery cells come with a circuit to shut off the battery when it approaches being overcharged or over-discharged. These batteries are known as protected batteries. On top of that Li-ion chargers are made with both constant voltages and constant currents that will detect when to switch from constant current to constant voltage and when to stop supplying the constant voltage. A lot of Li-ion chargers actually allow you to set the constant voltage supplied by the charger for when you are using multiple cells to supply a higher voltage. Like NiMH batteries, Li-ion

batteries are negligibly toxic. While NiMH batteries are a solid choice for a lot of applications, we ultimately agreed that the faster charging, lighter weight, and more fluctuation in voltage of a Li-ion battery better suited our needs. Li-ion is the current front-runner but there is still one more type of battery to consider in lithium polymer.

Lithium Polymer (LiPo)

Lithium polymer batteries are very similar to Li-ion batteries, but they come in a soft pouch instead of the hard cylinder or rectangle shape of a standard Li-ion battery. The reason they are able to be held in a soft pouch is because instead of using the liquid electrolyte that a standard Li-ion battery uses, they instead use a solid polymer electrolyte. These batteries are commonly referred to as lithium polymer batteries because of the lithium ions and the polymer electrolyte.

LiPo batteries are available in various different pouch sizes. They are lighter than their Li-ion counterparts, but in turn they suffer from a worse energy density. The lighter weight of LiPo batteries make them perfect for use in radio controlled models, but at the cost of a firm structure. LiPo batteries are in danger of having pressure put on the pouch that increases capacity retention. The difference in weight will be barely noticeable in a handheld gaming console. Having to make sure that the LiPo battery does not have any pressure put on it could make designing the interior layout of the case more difficult.

The voltage characteristics of a LiPo battery are the same as a Li-ion battery, and the batteries are also charged in the same manner. They also have the same amount of charge cycles as a Li-ion battery. They are, in general, more expensive than a Li-ion battery of equal charge capacity, but the difference in cost is not a deal breaker.

The LiPo battery is another solid choice, and one that potentially suits our needs. While we really like how lightweight LiPo batteries are, we do not think that such a minor difference in weight will have a huge impact on the portability of our system, yet it is still a benefit when designing a handheld device. We do, however, think that being required to compensate for a sensitive battery pack that cannot withstand pressure will make determining the interior layout of the case significantly harder, but at the same time the wide variety of sizes and shapes that LiPo pouches come in could make case layout easier. LiPo batteries can be as thin as a credit card which is a huge selling point. When it comes down to a direct comparison between Li-ion and LiPo the choice is still not clear, so both of these types of batteries will be compared when choosing a specific battery.

Choosing a Battery

To pick a specific battery we first had to decide how much voltage we would need to supply to the circuit. In order to keep our cost and battery size lower we decided that the standard 3.7 volts provided by a Li-ion or LiPo battery cell was fine, and that we would increase the voltage to whatever we needed it to be with a DC to DC converter. From there it could just be regulated down to whatever voltage different components in the circuit need.

After we determined that 3.7 volts would suffice, we had to figure out what charge capacity we would need for our device to run for a fair amount of time. The Raspberry Pi 2 will draw anywhere from 300 mA to 500 mA of current, but we can safely assume that for our case the Raspberry Pi 2 will draw no more than 300 mA on average. The 3.5-inch screen that we selected says that it will draw around 150 mA. To be safe I will assume that it will draw 200 mA. The other components of the device will not add up to anywhere over 200 mA, so assuming that 700 mA of current will be drawn on average is a fair estimation, if not too high. This means for a battery life of 3 hours, not including the charge provided by the solar panels, we would need a battery with a capacity of 2100 mAh. Our selections will be limited to batteries at or above 2100 mAh.

The next thing to check when choosing a battery is physical size, weight, and shape. Upon browsing through many different batteries we came to an important choice between two Li-ion batteries and one LiPo battery. Table 3.2.8.1 shows a comparison of the specifications between the two different batteries. The difference in physical size of the two Tenergy Li-ion batteries is simply that the 5200 mAh battery is the same size as two of the 2200 batteries put next to each other with a wrapper around them. The LiPo battery is wider than the Li-ion batteries, but it is also less than half as thick. The LiPo battery is also the lightest battery, though the 2200 mAh Li-ion battery is very similar in weight. The 5200 weighs about twice as much as the other two batteries. All three batteries contain a protection circuit, which is ideal as additional protection against overcharging and overdischarging. The 2200 mAh Li-ion battery is the only cylindrical battery, the others being rectangular. We determined that a rectangular battery was more ideal because the screen and Raspberry Pi 2 are both rectangular so the battery could just be placed under those components. The price difference is less than ten dollars between the three batteries.

	Tenergy Li-Ion 18650 Battery Module 5200	Tenergy Li-Ion 18650 Battery Module 2200	Adafruit Lithium Ion Polymer Battery
Capacity (mAh)	5200	2200	2500
Size (mm)	66 x 37 x 19	69 x 19	65 x 51 x 8
Weight (g)	96	54	52
Protection Circuit	Yes	Yes	Yes
Shape	Rectangular	Cylindrical	Rectangular
Price	\$19.99	\$10.99	\$14.95

Table 3.2.8.1: Comparison of Battery Choices

The LiPo battery fit our needs the best. It has the rectangular shape that we determined was a better fit than a cylindrical shape. Even though it is wider than the other batteries it is still not as wide as the screen or Raspberry Pi 2. It is also lighter than the other batteries and right between them in terms of cost. It has a capacity of 2500 mAh which is enough for our needs. While it would have been nice to have a battery with a large capacity like the 5200 mAh battery, it was decided that the thin and lightweight features of the LiPo battery made it the best fit for the FunBox Classic.

3.2.8.2 Charger

Due to the selection of a LiPo battery, we need to choose a charging IC chip to match our needs. There are a large amount of things to compare between the different chips. Even though different chips have many different features there are a few features that matter more to us than others. We need the chip to make sure the battery safely charges without overcharging. The charging circuit needs to take in 5 volts from a standard USB connection, either to a wall or a computer, and charge the battery using the constant current followed by constant voltage method. The constant current should be 500 mA and the constant voltage should be 4.2 volts. There are some additional peripherals that it would be nice for the chip to have but that are not necessary. These peripherals are mostly just additional safety features. There are quite a few different LiPo charging ICs, so choosing the one that best fits the FBC will prove to be challenging. After choosing a chip we will need to design a charging circuit using the IC to take in a voltage input and charge the battery. We also would like to include two LEDs to show when the battery is charging and when it is fully charged.

Charging IC Selection

After comparing many Li-ion charger chips based on the specifications we set, we narrowed our selection down to three different ICs. The ICs that we are choosing from are the LTC1734, LTC4056, and the MCP73831. We will now examine the specific chips in more detail.

The LTC1734, shown in Figure 3.2.8.5 in a sample 300mA constant current circuit, has a constant current anywhere from 200mA to 700mA programmable by attaching a resistor to the PROG pin. It is available in both a 4.1 and 4.2-volt model, the latter being the relevant one to our project. The IC enters sleep mode when no power is connected to it and draws negligible current from the battery. The PROG pin also has a constant 1.5-volt output during the constant current portion of charging, which then drops down to very low voltage levels as charging nears completion and the current through the resistor reaches the cutoff level. The chip also has a built in temperature sensor that protects the circuit from

getting too hot, and limits the output current to prevent damaging the battery or IC. The LTC1734 is 1 mm thick and has 6 pins.

300mA Li-Ion Battery Charger

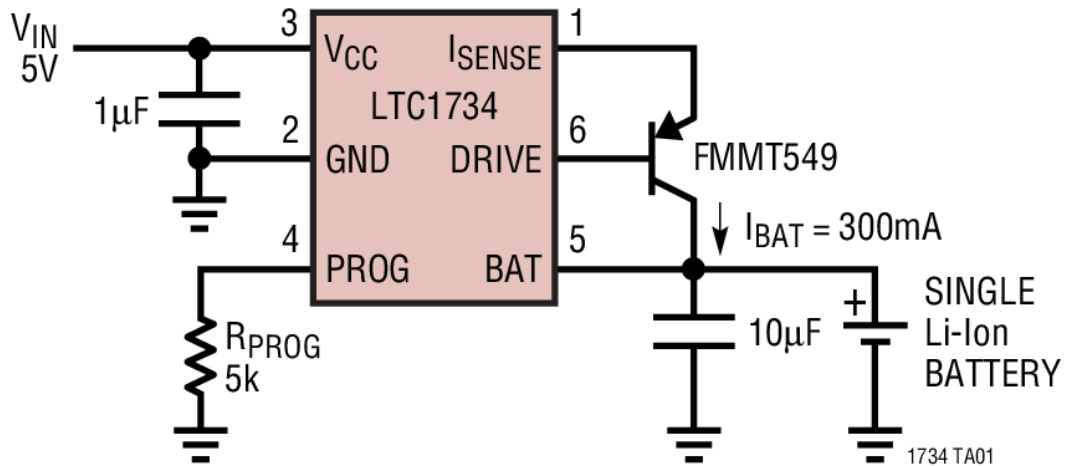


Figure 3.2.8.5: Sample 300 mA LTC1734 Circuit (Permission Pending)

The LTC4056, shown in a 700 mA constant current charging circuit in Figure 3.2.8.6, is made by the same company as the LTC1734 and shares some similarities. Per modern Li-ion standards the constant voltage output is 4.2 volts. The constant current can reach up to 1400 mA using a resistor attached to the PROG pin on the IC. The constant current is equal to the current through the PROG resistor multiplied by nine hundred fifteen. The PROG pin outputs a reference voltage of 1 volt for resistor selection. The current through the PROG resistor can be tested in a similar fashion to the LTC1734 to shut down the circuit if desired. The CHRG and TIMER/SHDN pins are used to program a shutdown timer for the circuit to protect against overcharging during the constant voltage step of the charging process. The LTC4056 has the same temperature and current protection as the LTC1734.

The MCP73831, shown in a sample 500 mA charging circuit in Figure 3.2.8.7, has only 5 pins, which is less than the number of pins in both of the LTC charging ICs. The charging current is anywhere from 15 to 500 mA. The output current is also decided by the resistor attached to the PROG pin. The MCP73831, like the other ICs, has current and temperature protection. The V_{DD} pin takes a voltage, V_{in} , anywhere from 3.75 to 6 volts. V_{DD} needs to be attached to at least a 4.7 microfarad bypass capacitor. V_{BAT} connects to the battery and outputs the constant current and voltage needed by the battery. It also needs to be connected to at least a 4.7 microfarad bypass capacitor. The STAT pin is an

output that connects to a LED to show the status of the battery or to a resistor to connect to a microcontroller. The V_{SS} pin connects to ground. The charge cycle ends if the current drops below 5 percent of the charge current.

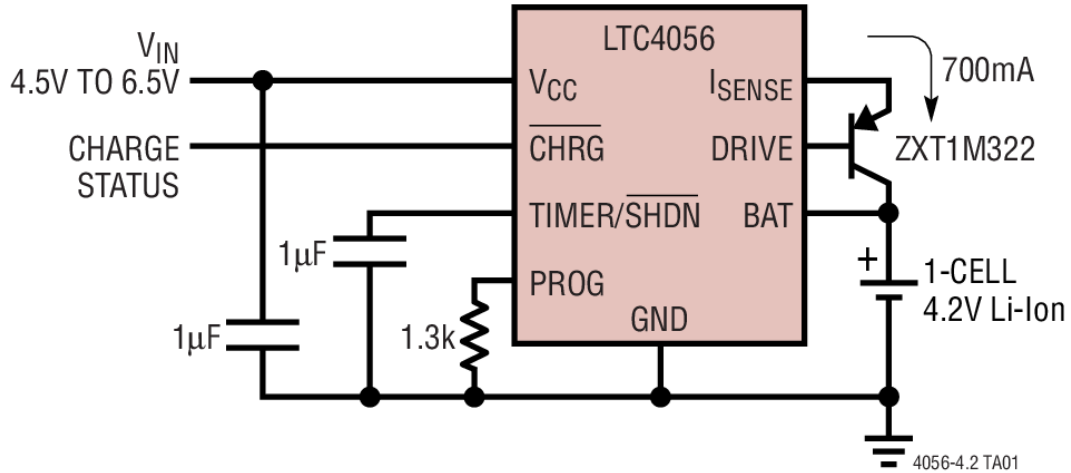


Figure 3.2.8.6: Sample 700 mA LTC4056 Circuit (Permission Pending)

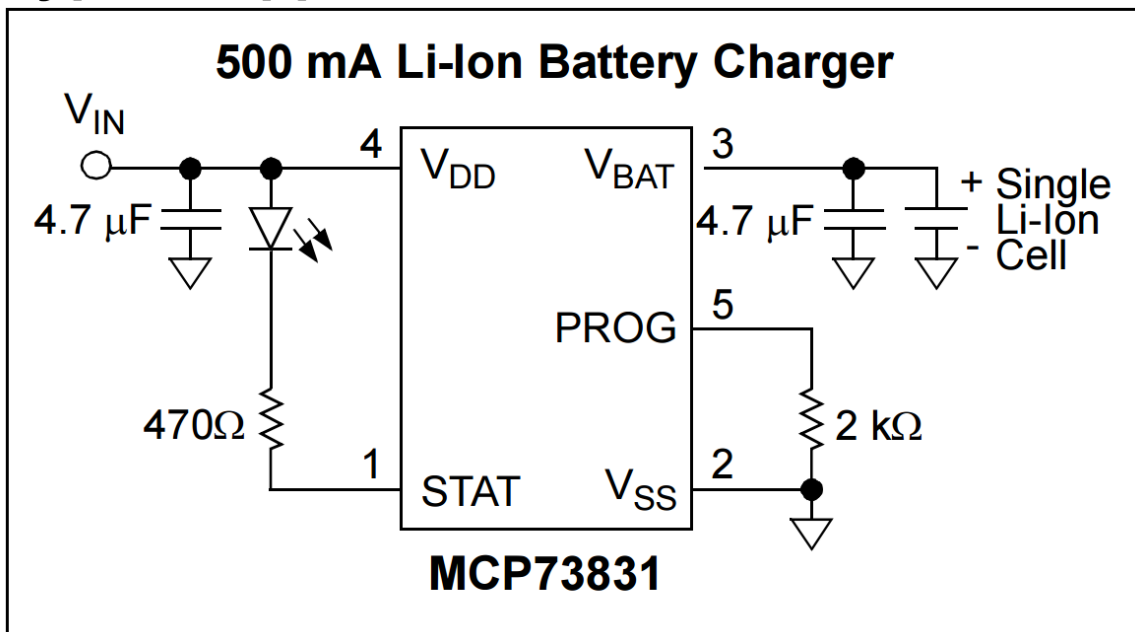


Figure 3.2.8.7: Sample 500 mA MCP73831 Circuit (Permission Pending)

Comparing the datasheets of the three different chips made choosing the MCP73831 a pretty clear choice. The MCP73831 offers the most out of the chips compared to the complexity of including it in a circuit. The timer feature on the LTC4056 was deemed unnecessary. The MCP73831 will also allow us to hook

up an LED to show when the battery is charging and fully charged. The max charging current of 500 mA is perfect for us. Ultimately the MCP73831 charger IC offers everything we need and will be the easiest to implement in our charger circuit. At 56 cents per unit the MCP73831 is also well within our budget.

3.2.8.3 DC-to-DC Converter and Regulator

Now that we have a circuit to charge the battery we need to make a circuit to power our device using the battery. The output of the battery ranges from 3 to 4.2 volts and the battery is able to output up to 2500 mA of current, which is far more than our device will draw. Since we want to turn 3 – 4.2 volts into a regulated 5 volts we need to find a Step-Up (also sometimes called Boost) DC-to-DC converter chip. Just like with the charging IC there are a multitude of Step-UP DC-to-DC converter chips available. Thankfully we have a few requirements for our converter chip that will help narrow down the selection. The chip needs to be able to output 5 volts. Whether it only outputs 5 volts or has a variable voltage output does not make a huge difference to us, but a variable voltage source will allow us to make the voltage slightly over 5 volts if we need to. The converter also needs to be able to output enough current to power our system. While our system will probably run under 700 mA, without having been able to test the actual current draw of the system we want to be safe and only look at converters that support close to or more than 1 amp of current output. We also would like to limit our search to synchronous converters due to their smaller size and higher efficiency than their nonsynchronous counterparts. Removing the need for an external power diode is also nice. Another thing we need the converter to have is a dedicated enable or shutdown pin. This pin will connect to the battery and ground through a simple switch. This switch will act as the ON/OFF switch of the device. A power efficiency of over 90% at 1 amp is also desired. A lot of the converters we came across the power efficiency seemed to fall off significantly around 1 amp. The two DC-to-DC converters we compared that were most suitable to our device were the PAM2401 and the TPS61030

The PAM2401, shown in Figure 3.2.8.8 in a sample 5 volt out circuit, can take an input voltage anywhere from 0.9 volts to 4.75 volts. Our LiPo battery will always be between these ranges when operating safely. The PAM2401 supports currents up to 3 amps which is well above our 1-amp requirement. The chip also has a dedicated enable pin. The output voltage can range anywhere from 2.5 volts to 5 volts. While we currently need 5 volts, if it turns out we need slightly higher voltage due to the addition of a diode or longer cables than we would be forced to get a new IC. The power efficiency of the system, shown in Figure 3.2.8.9, drops to around 90% for our operating voltage range. This fits our requirements. The PAM2401 is definitely an okay choice, but does not leave any room for future changes to circuits.

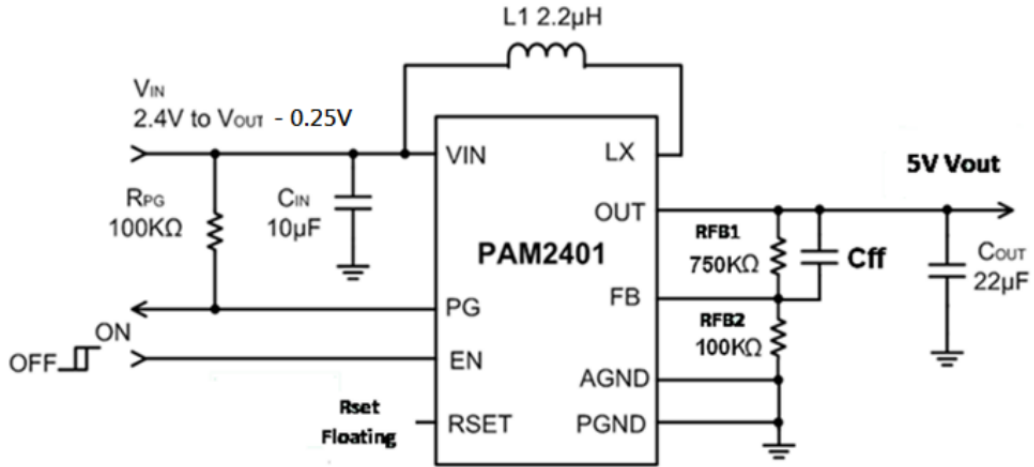


Figure 3.2.8.8: PAM2401 5 Volt Output Circuit (Permission Pending)

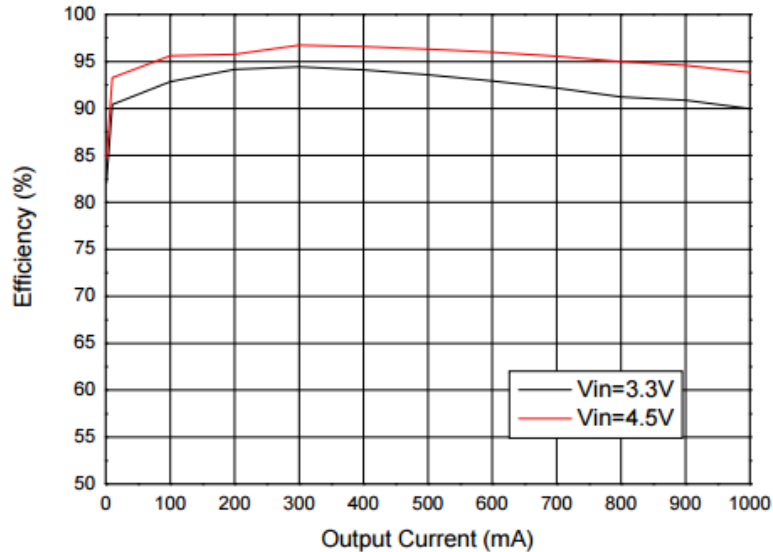


Figure 3.2.8.9: PAM2401 Efficiency Curve (Permission Pending)

The TPS61030, shown in Figure 3.2.8.10 in a sample 5 volt out circuit, can take an input voltage anywhere from 1.8 to 5.5 volts. The LiPo battery will always be well within this range. The TPS61030 outputs currents up to 1 amp at 5 volts. This perfectly matches our needs. Like the previous IC, this chip also has a dedicated enable pin. The output voltage of the chip can go up to 5.5 volts. This gives us a little headroom in case we need to increase the voltage a bit to compensate for changes in the circuit. The power efficiency of the system, shown in Figure 3.2.8.11, is about 95% at 3 volts input voltage and 1-amp output current. This efficiency will always meet our requirement of above 90% because our battery will normally be operating at a higher input voltage with a lower output current which will only increase the efficiency even further. The TPS61030 is an

excellent choice. It fits all of our requirements and has good efficiency which will lead to longer battery life.

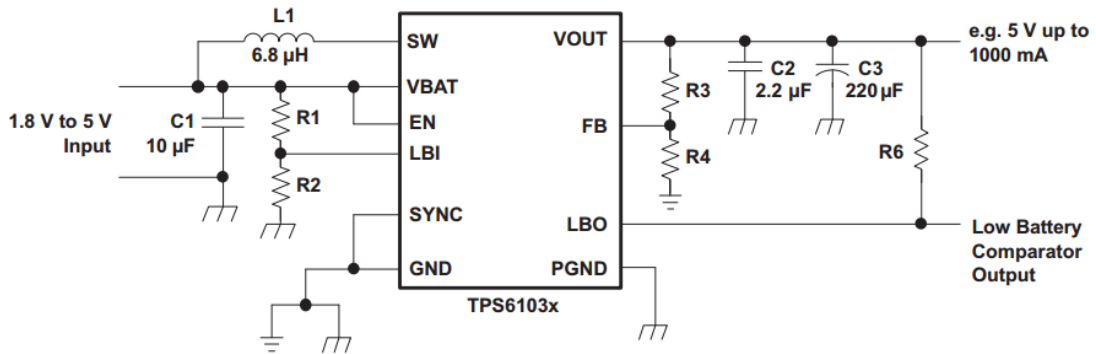


Figure 3.2.8.10: TPS61030 5 Volt Output Circuit (Permission Pending)

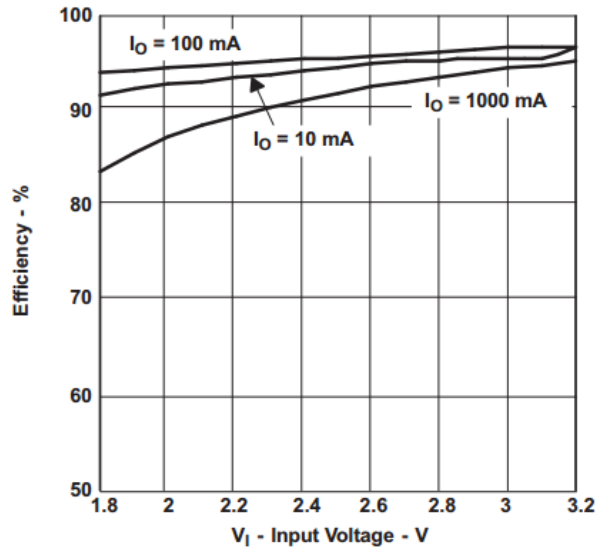


Figure 3.2.8.11: TPS61030 Efficiency Curve (Permission Pending)

Both the PAM2401 and the TPS61030 fit our specifications, but the TPS61030 definitely matched what we need more closely. The 3-amp max output current of the PAM2401 is unnecessary for our applications and the efficiency is not as good as the TPS61030. The slightly over 5 volt maximum output that the TPS61030 is potentially useful whereas we would be stuck with 5 volts maximum using the PAM2401. The TPS61030 meets our needs perfectly while giving us just enough headroom. It is also perfect for our portable system because the power efficiency is so high. The LBO (low battery comparator output) pin will allow us to have an LED indicate when the device has low battery. The cost of the chip is only \$3.15 which is definitely an acceptable amount.

3.2.9 Portable Console Design

There are two lines of portable handheld Nintendo consoles that we considered researching for design of our product. They are Nintendo DS and Game Boy lines. Below, we can see two examples in Figure 3.2.9.1: a unibody design on the left, and a clamshell design on the right.

3.2.9.1 The Nintendo DS and 3DS Line



Figure 3.2.9.1 - Dual Display Consoles

In Nintendo DS, consoles have two displays. One of them is LCD screen working and another is a touchscreen. Both screens have a clamshell design, and located either on a one-piece case or a case with a folding top. Two screens add a lot of weight to the console; so, we have to be very careful if we decide to pursue this design.

Nintendo 3DS is an autostereoscopic device, which means it projects stereoscopic 3D effects without any additional accessories such as 3Dglasses. This design of a console looks very appealing. It would enhance user-experience playing the games on it. However, 3D

A console has built-in microphone and supports wireless connectivity. Wireless connectivity (Wi-Fi) provides the multiplayer capability. Nintendo created its own Nintendo Wi-Fi Connection service that allows multiple users directly connect to each other via Wi-Fi within a short distance without connecting to local wireless network. Display sizes vary between 3.12 and 4.88 inches. Displays have 5 levels of brightness. The weight varies between 214 and 336 grams.

Input controls on Nintendo DS devices include: D-pad, circle pad, C-stick, A/B/X/Y, L/R, ZL/ZR, and START/SELECT buttons, home button, touchscreen, microphone, camera, wireless switch, motion sensor, and gyro sensor.

Hence, a unibody case looks very big and uncomfortable. The case with a folding upper part looks good, but it will be very hard to 3D-print it in the lab. We want to

create a custom product that should not be dependent on the Nintendo Wi-Fi Connection.

3.2.9.2 The Game Boy Line

Game Boy line introduces a single display consoles. Game Boy Pocket has electroluminescent backlight for low-light conditions. It can have up to five brightness levels. Weight varies between 80 and 142 grams.

Input controls on Nintendo DS devices include: D-pad, circle pad, C-stick, A/B, L/R, and START/SELECT buttons, home button, touchscreen, microphone, camera, wireless switch, motion sensor, and gyro sensor.

The first console in Figure 3.2.9.2 has a general rectangular shape. This shape does not look very convenient to use. If we decide to use it, we would like to modify and add some elongation to sides. It would be more comfortable for a user to hold this device while playing the games. The console on Figure 3.2.9.3 has a clamshell design similar to Nintendo DS.



Figure 3.2.9.2 – Single-Display Rectangular Console



Figure 3.2.9.3 – Single-Display Clamshell Console

A Game Boy line offers nice designs that look less bulky, lighter in weight and provide the same game-playing capabilities. On the first image, we like the location of D-pad, A/B, and L/R buttons. The second image has elongated sides, which make the device easier and more comfortable to hold. Also elongated sides provide extra space that we could use for the speaker, for example. The third design looks nice but we think it won't be very comfortable to hold it in hands.

3.2.9.3 Research on the Case Design

The design of the case will depend on such characteristics as the shape of the case, dual- or single-display layout, the location of the buttons, and the size of inside components.

Dual versus Single Display

Two-display design looks more appealing toward providing a better user-experience. However, since we will build our device manually in a regular lab conditions, we will most likely have an extra weight on it versus professional Nintendo line devices. Having two screens on our console will make our device very heavy and bulky. So, we decided to go with single-display console option.

Clamshell versus Rectangular Design

The rectangular design looks less complicated to manufacture. However, it will be more challenging to design the layout of inside components. We have circuit boards, speakers, battery and other components. If we choose this case, we'll have to decide on how compactly put inside parts together so they will not short-circuit.

We plan to install solar panels on our console as an additional source of keeping the battery charged for longer time. If we choose the clamshell design, we can locate solar panels on the back of the folding top. A user can receive solar light this way almost continuously. The solar panels will keep the battery alive for many hours. If we choose the rectangular shape for the device, solar panels will be located on the backside.

This design won't provide continuous charging of the battery since the user can't control the change of the angle of holding the device. However, a user can always flip the device so it can directly face the sunlight and leave it for some period of time until the device won't be charged to the desired level.

Solar panels are relatively big in size and will take a lot of space on the console. We need to choose a shape of the device, on which solar panels won't look bulky but will be a nice complement to our elegant device. In clamshell design, solar panels will be elegantly suited on the backside of the upper part and we won't need to take up on extra length to locate the other parts that go into the device. We'll also have a plenty of room in the lower part, and we can comfortably locate there all our circuit boards.

If we choose the rectangular shape for device, we'll have to locate solar panels on the backside of the console. It can make our device look too thick. We'll have to apply extra thinking on how to locate our circuit boards, speakers and other

parts to make the device look elegant. So far two options are somewhat similar in the level of manufacturing difficulty.

Next thing we need to consider is manual manufacturing and 3D printing of the device. We think we may run into an unsolvable problem when we'll be ready to 3D-print our case. We think 3D printing two parts separately and then attaching them together is the level of manufacturing difficulty that is far beyond from our capabilities.

We will have to take a lot of risk if we decide to pursue the clamshell design because if we won't be able to attach the two halves, our entire design will fail, and we'll have to redesign the case completely. So, to be on the safe side, we decided to choose a rectangular shape device with few modifications. We would like our design to be similar to the design shown in Figure 3.2.9.4. This model has very clean look. We would like to elongate the sides a little bit more to deliver a better user-experience.



Figure 3.2.9.4 - Retro Bit Portable Handheld Console V2.0
Reprinted with the permission of Retro Bit Games

3.2.9.4 Designing the Front-Facing Buttons

We would like to implement the lean design technique for our console. When the user's attention is shifted from one location to another, there is a cost associated with time or effort. If buttons are not easily accessible, the user may feel frustrated and even lose interest in playing on the console with such buttons. The design of the buttons should minimize this inconvenience by providing the buttons located to the nearest possible position.

We want to design our console with minimum amount of buttons, but without reducing the user-experience. For front-facing buttons we need to have a start, select and game controlling buttons. Usually, Nintendo consoles have the following buttons: The D-pad, an analog stick, and start, select and four A/B/X/Y buttons. We decided not to use an analog stick on our console. We can see an example of an analog stick and a D-pad in Figure 3.2.9.5.



Figure 3.2.9.5 - Analog stick on the top and D-pad on the bottom

Front-facing buttons that will be used:

- D-pad
- 4 A/B/X/Y buttons
- Start button
- Select button

In Figure 3.2.9.6, a Nintendo Wii U Pro Controller is shown. We would like to reuse the circuit board from this controller that contains D-pad, select and four A/B/X/Y buttons.



Figure 3.2.9.6 - Nintendo Wii U Pro Controller

3.2.9.5 Basic Design Philosophy

The mechanical design of the Fun Box Classic portable gaming console stemmed from two basic requirements:

- Create a cost-effective device that meets all applicable environmental requirements.
- Utilize boards outlines and interconnections of all the parts in order to fit them inside the case.

Research into recent designs of the portable consoles led us to select the mechanical design philosophy of a Retro Bit portable handheld console shown in Figure 3.2.9.4 and Nintendo Wii U Pro Controller shown in Figure 3.2.9.6.

Figure 3.2.9.7 represents the rough design of Fun Box Classic console. The button assignments are given in Table 3.2.9.1.

(Front View)

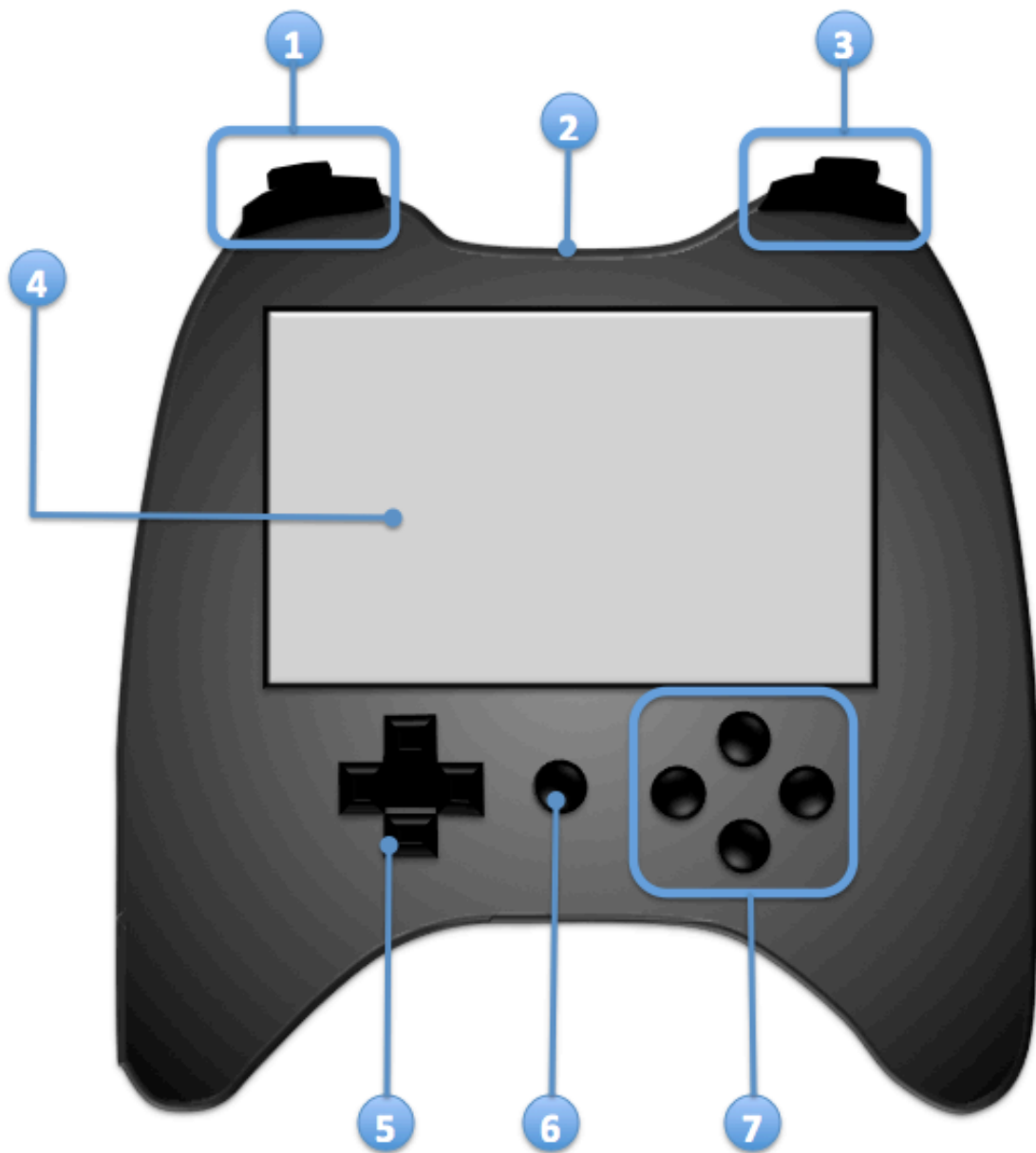


Figure 3.2.9.7 – Case Design of Fun Box Classic Console

Number	Button Name
1	L & ZL Buttons
2	USB Port
3	R & ZR Buttons
4	LCD Display
5	Control D-Pad
6	Power Button
7	A/B/X/Y buttons

Table 3.2.9.1 Front View System Components

Advantages include:

- The opportunity to use highly cost-effective castings common across the range for several of the major components, such as the front panel, back pane and two shoulder buttons.
- Well proven structural integrity
- Effective space utilization

3.2.9.6 Parts Placement

To make this type of case, we first have to measure all the parts to find out the width and height of them. Then, we will stack all the parts and find the depth of the console.

We will measure all our parts in mm units since these are the most common units used in specifications for the parts that we bought.

Front Panel

The width and height of the display are respectively 105 x 65 mm. The width and height of the circuit board with the front buttons are 127 x 50.8 mm. Then we consider the dimensions of elongated legs that will be similar to the ones in Nintendo Wii U Pro Controller. That gives a portable console with the approximate dimensions of width 158.5 across at its widest point and about 160.5 mm tall.

The visual representation of the front panel is shown in the Figure 3.2.9.8.

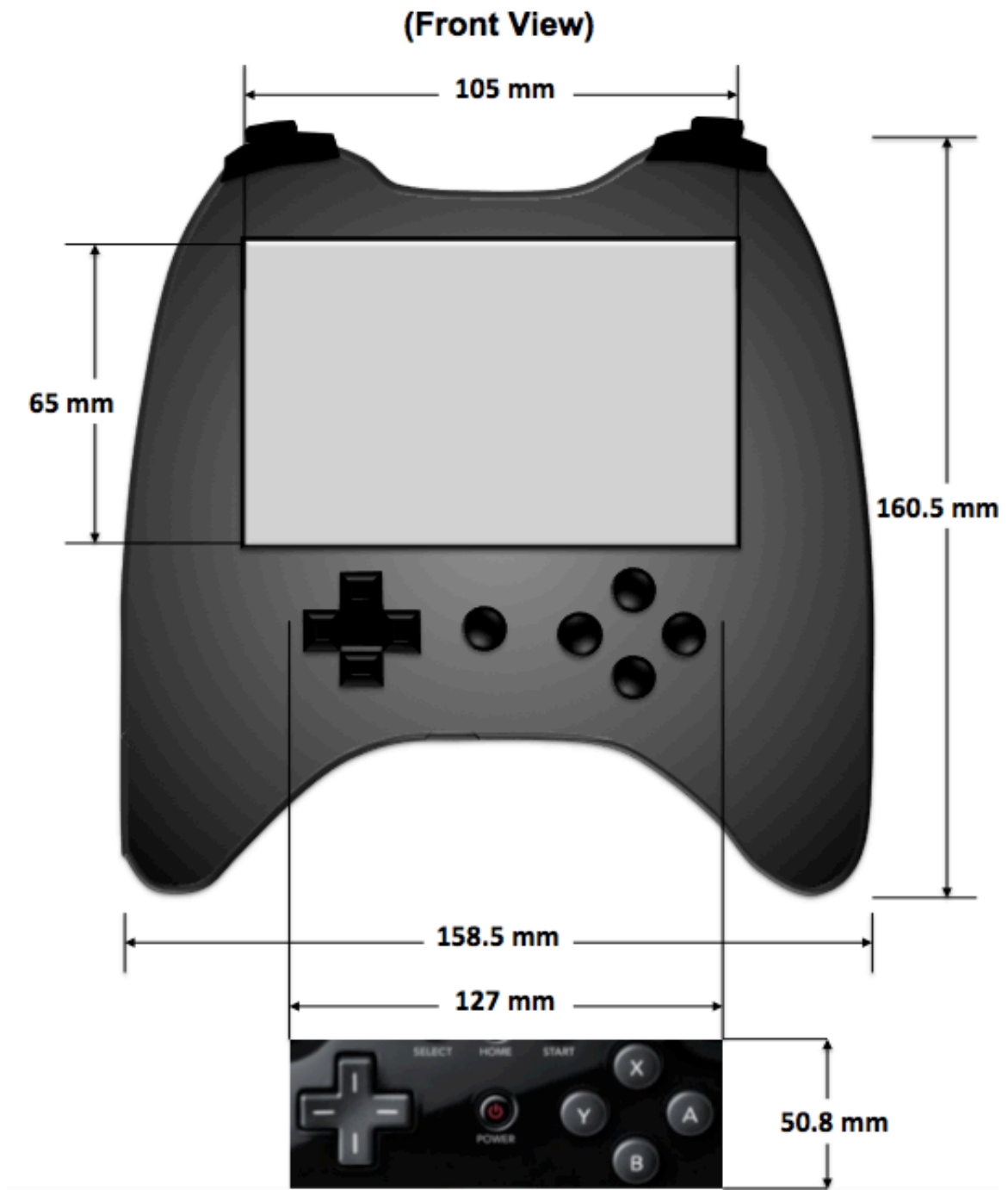


Figure 3.2.9.8 - Front Panel

Back Panel

A physical characteristic of the backside of the case is to allow the users to remove the battery through an accessible lid that can be unscrewed. We will use rechargeable lithium ion polymer battery that has 65 x 51 mm in width x height. Analyzing the lid sizes of existing consoles and considering the size of the lithium ion battery, we will design the lid with dimensions about 65 x 127 mm. We will also implement two solar panels. There are also two solar panels located in the backside. The size of each solar battery is 86 x 14 mm in width x height. The approximate design of the back panel is shown in the Figure 3.2.9.9.

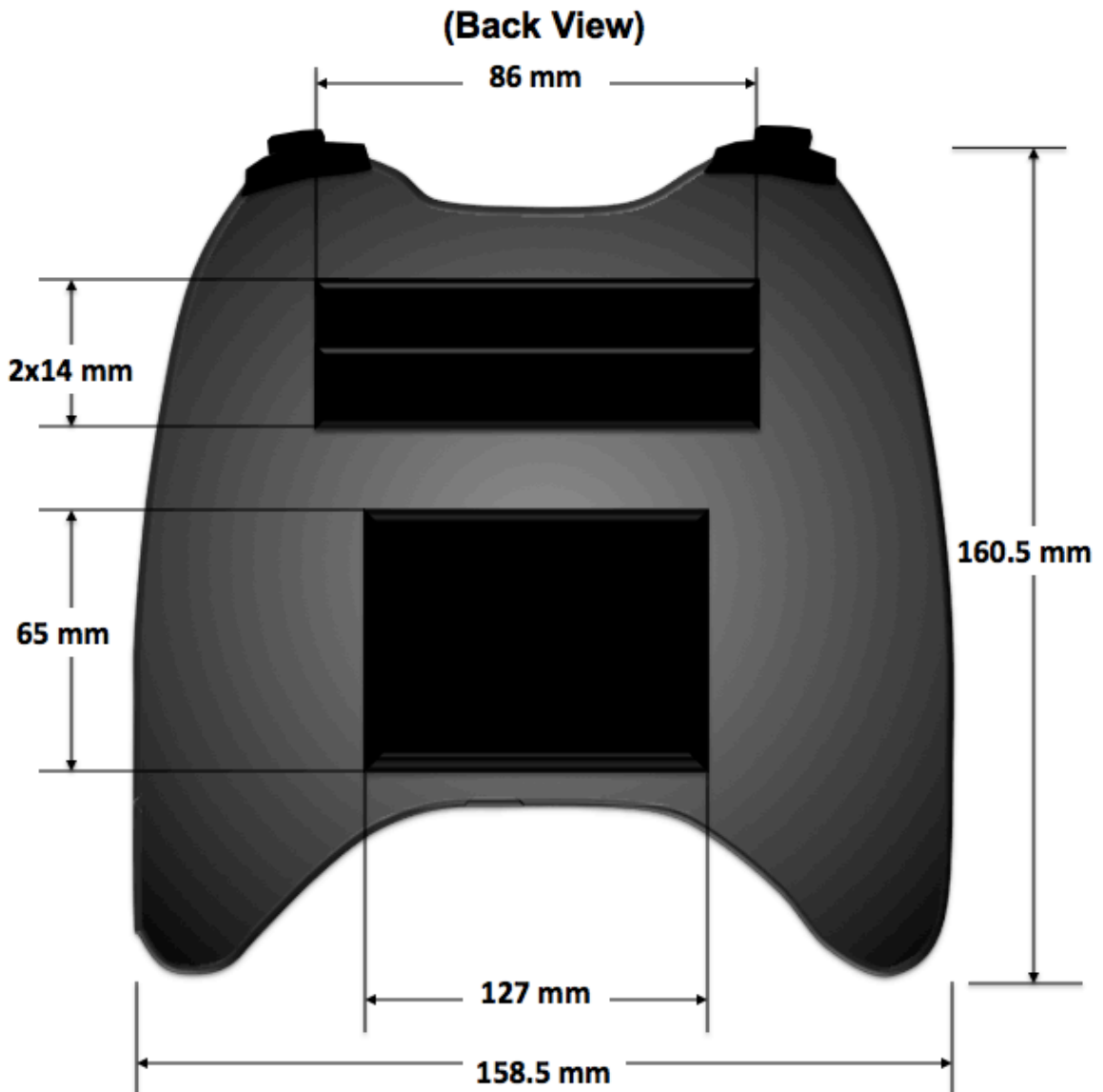


Figure 3.2.9.9 - Back Panel

Side View

The Table 3.2.9.2 has an approximate estimation of the thickness of the future console. This analysis is based on the specifications of some big parts that will go inside the console and also on the specifications of existing products that we compare our product to. We can see what we think a side of the console will look like in Figure 3.2.9.10.

Part	Thickness (mm)	Weight (g)
Solar Panel	2	5
Battery	8	52
Display	15	132
Raspberry Pi	3	50
Other Components	57	240
Total	85	479

Table 3.2.9.2 - Estimate Thicknesses of the Console



Figure 3.2.9.10 - Side View of the Future Console

3.3 Software Research

This section details the research into a base operating system for our device, a software frontend GUI for users to interact with, a software backend to run the games, and potential modifications to the operating system.

3.3.1 Base Operating System

Initially, we considered using a FPGA system and designing from the very lowest level a custom operating system, built from a custom kernel geared towards the FPGA. The advantages were obvious. We would have been able to maximize power efficiency, since no unnecessary OS functions would have been required that would have drained more power, like a task manager. We would have been able to cut down on the disk space used by the OS as, again, no unnecessary space wasters, like advanced GUI elements, would be required. Finally, we would have been able to make sure the OS used all components of the hardware, such as the processor and memory, most optimally. This would have

ensured the maximum possible amount of resources was available to the emulators, for maximum performance for each game.

However, the main disadvantages were too great to overcome. Learning to build an operating system from scratch, including the BIOS, the kernel, libraries, drivers, the GUI, and the user ring, would have taken an inordinate amount of time. From memory management to file system permissions, there was simply too much to do in too little time. In addition, we would have had to port all the emulators to our custom operating system, requiring more time. The operating system or emulator ports alone could have filled up a Computer Science Senior Design project, and thus we opted to abandon our overly ambitious goal. We can see how complex a standard operating system structure is by the flowchart in Figure 3.3.1.1 below.

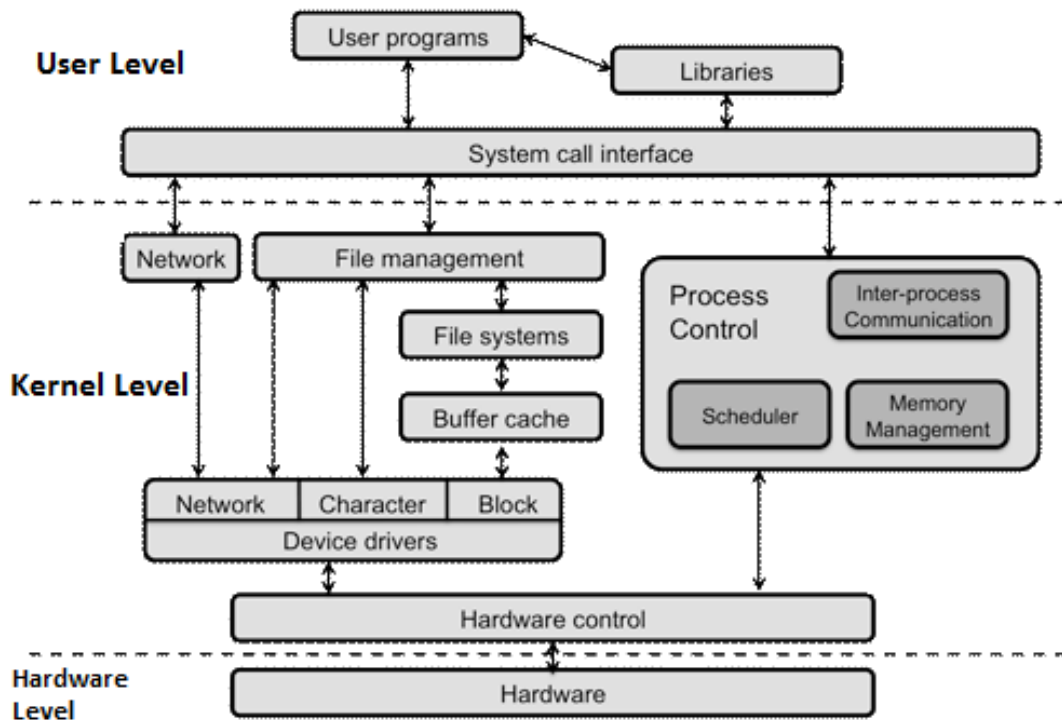


Figure 3.3.1.1 – Full Operating System Flowchart (Permission Pending)

We decided to go with a Linux Kernel, due to its open-source, and therefore freely available, nature as well as the easily modifiable kernel. We thought, initially, about using something easy to use, like Ubuntu, but realized we should probably switch to a lighter distro that would more easily run on a micro-computer, like Raspberry Pi or BeagleBone Black. Looking at commercial derivatives, such as Fedora, and fan-proclaimed “pure” distributions such as Arch Linux, we opted to go with a midline Linux distro, Debian. Specifically, we used the most supported version of Debian with the Raspberry Pi – Raspbian, made by the developers of the Raspberry Pi to work best with their system. A more detailed comparison of Linux distros is shown in Table 3.3.1.1.

Distro	Cost	Default FS	DE	arm support	GUI Installer	Rpi Support	ES Support
Arch-Linux	Free	None	None	Yes - Unofficial	No	Unofficial	Non-native
Debian	Free	ext4	GNOME, etc.	Yes - 32-bit	Yes	Yes	Native Pi
Fedora	Free	ext4	GNOME	Yes - 32-bit	Yes	2nd Party	No
Ubuntu	Free	ext4	Unity on GNOME	Yes	Yes	2nd Party	Non-native

Table 3.3.1.1 – Linux Distro Comparison

As can be seen in the table above, Debian meets all of our system requirements perfectly, and thus becomes the obvious choice.

3.3.2 Software Frontend

We will be using a frontend called EmulationStation, which is built on top of the Raspbian variation of the Debian distro, specifically designed for Raspberry Pi. This will allow us to focus more on the hardware and not be concerned with the already functional software components. In addition to performing all necessary frontend functions, the EmulationStation software is freely available and encourages people to use it in whatever projects they have. The GUI shown to the user is displayed in Figure 3.3.2.1 below.

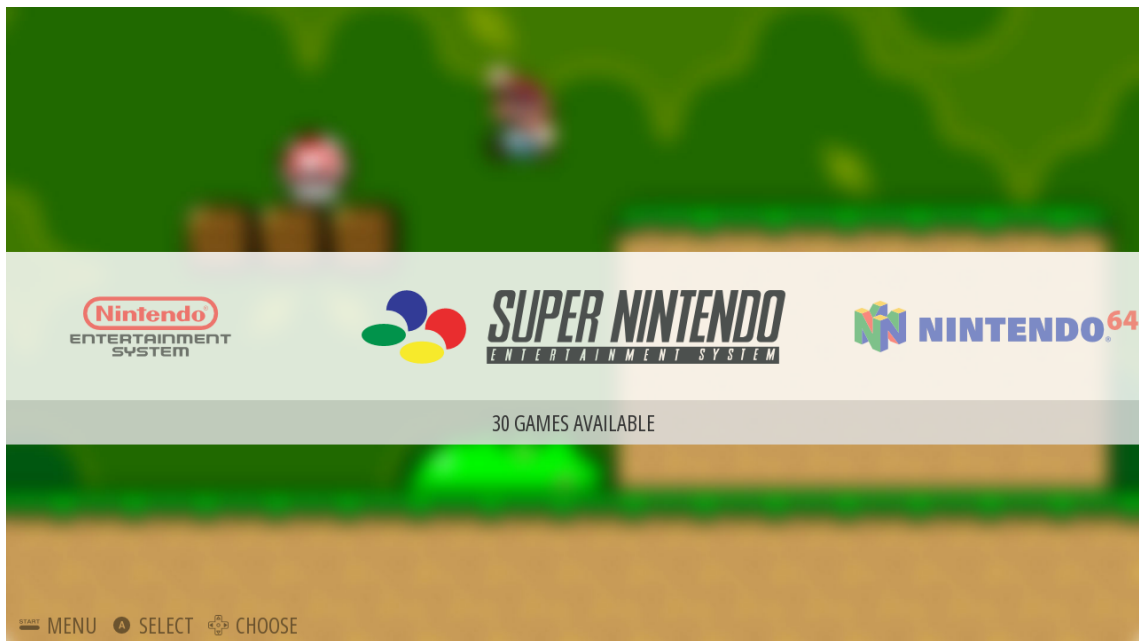


Figure 3.3.2.1 – EmulationStation GUI

We looked into other more fragmented solutions, some using RetroArch like RetroPie does. Each of these solutions, however, would require the user to browse a standard desktop environment like GNOME. This, given the size of the screen and the lack of internal keyboard and mouse controls, would be a non-ideal solution. EmulationStation is the official frontend of a combination solution called RetroPie, which allows us to have all emulators in a central directory, with games, emulators, and settings easily controllable with only a standard game controller. Certain options do require the use of a keyboard to configure, but these will be preconfigured before the final release, making that level of user input unnecessary.

As we have decided to support only the maximum native resolution of our screen, which is 640x480, we will configure the EmulationStation to support only this maximum resolution. This will reduce the needed resources from the Raspberry Pi 2, which will prevent low resource issues, such as blank screens and freezing. Additionally, we will only be enabling support for the Nintendo Entertainment System (NES), the Super Nintendo Entertainment System (SNES), the Game Boy (GB), the Game Boy Color (GBC), and the Game Boy Advance (GBA). These limitations will help to cut down on required resources as well. The BIOSs required to emulate each system will be extracted directly from the physical system, thereby sticking to legal use of the copied material.

Additionally, we will configure EmulationStation to accept not just the internal controller, but our external Bluetooth controller as well. Thanks to a built-in configuration utility, this will prove to be very easy.

3.3.3 Software Backend

ROMs of the test games will be extracted from our own physical copies of each game, using already available flash chips. These will be saved in their appropriate directories on the Micro SD card. To put ROMs on the card, it will be necessary to use a direct USB connection.

All emulator tests were performed using much larger sets of conditions, which we have consolidated to save space in the diagram. For each emulator, we performed a barrage of 47 to 159 tests, depending on the system being emulated. These are tests which the speed-running community uses to evaluate emulators. Since these players are obsessed with shaving off even one more millisecond of time from their playthrough of any given game, they know that they need perfect emulation. As a result, we can be assured that the tests are comprehensive and accurate.

For the NES, we decided between three emulators: Jnes, FCEUX, and NesterJ. Jnes was a very accurate and powerful emulator, but required too many system resources for accurate emulation. NesterJ was a solid emulator, that had been proven to work on portable systems, such as when it was ported to the Nintendo DS through homebrew applications, but ultimately did not suit our needs due to

emulation inaccuracy and having not been updated in quite some time. We opted to go with FCEUmm, a port of FCEUX with better mapper support, which provided the best balance of updates, performance, and system requirements.

The emulators below are the most accurate ones that were precompiled for the Raspberry Pi. The emulators puNES(closed-source) and Nestopia (open-source), when compiled for the Raspberry Pi, might provide better accuracy and performance, and so might be considered as the project progresses for maximum enjoyment. The test results are shown in Table 3.3.3.1.

Test	Jnes	nesterJ	FCEUX
APU (/40)	8	8	18
CPU (/55)	21	23	38
Mapper (/13)	0	1	7
PPU (/42)	6	10	21
Misc(/5)	3	4	5
Demo (/3)	0	0	0
Total (/158)	38	46	89
Grade (%)	24.05%	29.11%	56.33%

Table 3.3.3.1 – NES Emulator Accuracy Comparison

For the SNES, we decided between three emulators as well: ZSNES, a variation of SNES9x called PiSNES, and higan (formerly BSNES). Higan is, bar none, the most accurate emulator on the market. The creator mapped every single pathway of every single chip on the SNES and even expansion chips on the cartridges. For this effort, he was rewarded with 100% emulation accuracy. However, emulating an entire system perfectly requires far more resource than our Raspberry Pi 2 can support, and thus we sadly had to get rid of this option. ZSNES is an excellent emulator, being very accurate and also considerate of resources. Unfortunately, the port for Linux systems was not optimized well, and thus we were forced to remove it from consideration. We ended up going with a port of the popular SNES9x emulator specifically for Raspberry Pi. The test results are shown in Table 3.3.3.2.

As we can see, PiSNES reaches nearly 90% accuracy, without the massive performance drops seen by the higan emulator. As a result, it is the clear choice.

Test	ZSNES	PiSNES	higan
Blargg Tests (/9)	2	4	9
Official Tests (/30)	20	29	30
Cx4 Tests (/8)	8	8	8
SPC7110 Tests (/12)	12	12	12
Total (/59)	42	53	59
Grade (%)	71.19%	89.83%	100.00%

Table 3.3.3.2 – SNES Emulator Accuracy Comparison

For the GB and GBC, we chose between no\$gbm, Visual Boy Advance, and Gambatte. No\$gbm was relatively clunky, experienced slowdown, and has an appalling level of accuracy. Visual Boy Advance supports Game Boy, Game Boy Color, and Game Boy Advance games, making it a great choice to cut down on needed cores for emulation. However, when playing around with it on the portable system, there were some aspects of the user interface we felt did not mesh properly with the system as a whole. In addition, the main team stopped updating the emulator in 2004, and the new team has made some feature additions which we felt were not worth the additional resources required. Finally, its emulation accuracy is still poor. Gambatte, on the other hand, is pre-built with support for the backend system of RetroArch, making it ideal for our idea of a unified system set. It does not support GBA games, but we felt this lacking was made up for by constant updates and integration with the backend. The test results are shown in Table 3.3.3.3.

Test	no\$gmb	VBA	Gambatte
CPU (/12)	9	12	12
Sound (/24)	1	1	24
MEM (/3)	0	2	3
OAM (/8)	2	3	3
Total (/47)	12	18	42
Grade (%)	25.53%	38.30%	89.36%

Table 3.3.3.3 – Game Boy and Game Boy Color Emulator Accuracy

Additionally, we can see that Gambatte achieves nearly 90% accuracy, far more than either of the other tested options. With minimal performance effect and significantly better emulation accuracy, we can safely say that Gambatte is the clear choice.

Finally, for the GBA, we decided between Visual Boy Advance and gpSP. Visual Boy Advance was discarded for the reasons stated above. GpSP, while being originally for the PSP, received a native Raspberry Pi port and seems to work

flawlessly on it. For this reason of ease of use, we chose gpSP. As these were the only two options available and VBA had already been discounted, we felt that testing was superfluous at this point.

3.3.4 Operating System Modifications

Even using a premade operating system, there are several modifications we will have to make to maximize effectiveness on our platform.

First and foremost, we will make sure to maximize battery life. To this end, we will be disabling unnecessary features of the OS, such as the SAMBA share for the emulators, as we will have no network capabilities. Further, we will disable the LAN chip itself, giving us an estimated savings of 40 mA. These changes nearly double the battery life of the system, as the total current drawn when fully stressed will be 260 mA and it will idle at 200 mA. More exact measurements of power savings are shown in Table 3.3.4.1.

100% Load Cores	0	1	2	3	4
Pi2	230	280	320	380	420
Pi2 w/ OS Changes	200	215	230	245	260

Table 3.3.4.1 – Power Saved Due to Ethernet Removal

Secondly, as we are trying to minimize user complexity, auto-USB synchronization will be setup. By this, we mean that upon plugging in a USB drive to the system, it will copy the directory structure and contents of the ROM folder onto the USB. From then on, any changes made to the USB, such as adding or removing a ROM, will also be made on the system once the USB is plugged back in. In this way, the user does not have to navigate the Linux filesystem and worry about messing something up. We will also disable most setup options, so the user does not accidentally, say, delete the core emulators.

Thirdly, we considered overclocking the BCM2836 chip, as this made Nintendo 64 games playable. However, as we decided to not add an analog stick and as most games had large compatibility issues with the conversion to the ARM architecture, we dropped Nintendo 64 support. As a result, the performance benefits of overclocking were no longer necessary, and we were only left with a much larger power footprint. The idea of overclocking the system was then scrapped, leaving us with a more power efficient chip that serves performance purposes.

Finally, we will lock the software volume at 100%, leaving all volume control up to the hardware potentiometer. This will eliminate the potential of a user forgetting what they set the software volume at, and thus minimize user error in that area.

As for non-crucial modifications, we will be adding a splash screen at boot, to add a sense of professionalism.

4. Hardware Design

It goes without saying that the meat of a Senior Design project is, in fact, the design. This is the most important part of the first half of Senior Design. In this section, we will lay out, in detail, the design of the various hardware components that make up the FunBox Classic. Properly laid out, well researched, and well documented design will ensure the least amount of wasted time, effort, and money when it comes time to actually build the project.

4.1 Screen Setup

Our first step will be to take apart the screen and strip the component connectors, making sure not to damage the delicate ribbon cable in the process. Once properly disassembled and stripped, we will move onto the Raspberry Pi. We will first remove the existing composite video/stereo audio hybrid connector, as it is tall and does not fit our needs. This will also leave the audio connections open for our custom audio processing, to output it to headphones or speakers on a switch. We will connect the composite cable to the solder point PP24, marked with orange in Figure 4.1.1 below, which is the Composite Signal Input. The ground will be soldered to solder point PP6, marked in black, which is a ground point.

Next, we prepare the external screen to accept a more reasonable (for our purposes, at least) 5V input as opposed to the 12 V it currently accepts (due to it being a car monitor, this makes sense as cigarette lighters output 12V). This is accomplished with relative ease. In the figure below, you can see the linear regulator chip marked 1 in Figure 4.1.1. If we remove this chip, the circuit automatically accepts the desired 5V again. We then connect the 5V input and the ground to the unused USB header marked 2.

Finally, we de-solder the switches below 2 and link them to the backlight controller. By sending timed pulses, we are able to then control the backlight capabilities of the monitor.

By performing these three relatively simple steps, we have a natively supported screen, with the resolution and quality we need, with full backlight (and even contrast and saturation if needed) control. This also avoids any messy complications due to driver conflicts and any loss in quality due to analog to digital conversion.

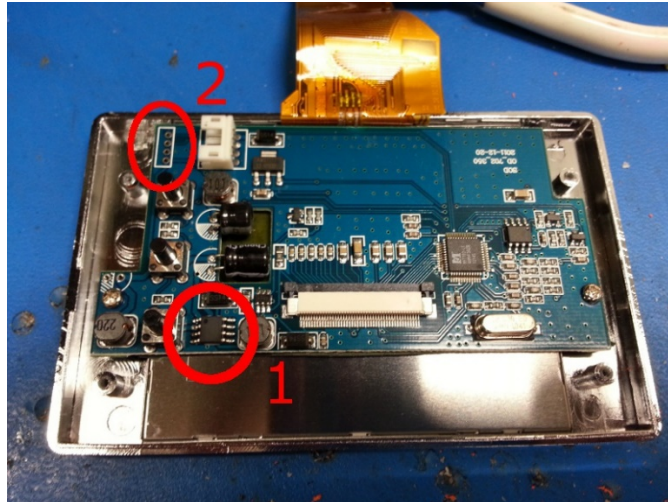


Figure 4.1.1 – How to De-Regulate the Screen (Permission Pending)

4.1.1 Backlight Controller

As the original brightness control switches for the screen would add unnecessary bulk (and complication, due to the menu system) to the design of our system, we have opted for manual control of the brightness. Two options are potentially available to us.

First, the method that would be guaranteed to work, we could have pulses sent to the areas where the switches used to be, in pre-programmed ways, to control the brightness. This method has the benefit of already being known to work, as it's just substituting wired pulses from a microcontroller for the original switches. The downside, of course, is that the onscreen menu overlay would display the volume, negating any software control of that interface. Additionally, it might be slower, as navigating a menu is, by its nature, slower than direct control.

Second, the method that we are currently testing, we could bypass the microcontroller and send whatever pulses it uses to control brightness to the LEDs directly. This method has the benefit of allowing us to use our own GUI for brightness level displays, as well as being faster due to bypassing the onscreen menu overlay. The disadvantage to this method is, of course, that it might not be possible or, more specifically, it might not be possible for us.

Whichever method ends up being the correct one, we plan on using the ATtiny13 from Atmel to control the backlight. This tiny, and cheap, chip is able to do both of the things we might require: either sending pre-programmed pulses by listening for specific input (the user changing brightness) or using PWM to control the brightness directly, also listening for the user's input (if the screen accepts this method). Additionally, with the use of a photoresistor and a switch that the user would control, we can set the brightness to automatically dim or brighten based on the current level of light in the room. This would, naturally, have to be a

system the user could disable at will, as these systems can provide an inadequate (or overlarge) level of light at times. A basic schematic for the backlight controller is shown in Figure 4.1.1.1.

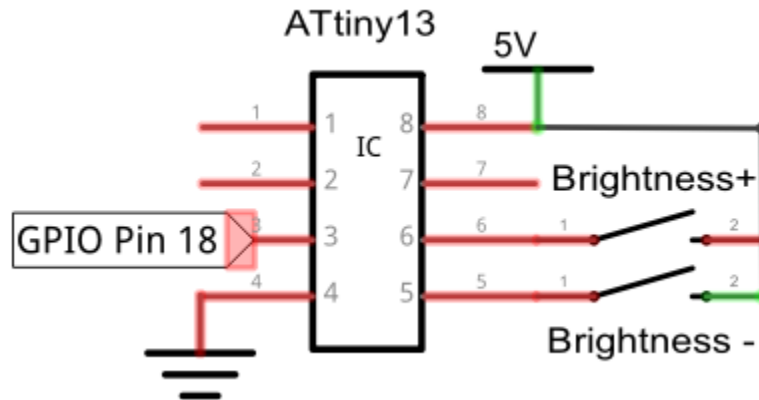


Figure 4.1.1.1 – Basic Backlight Controller Schematic

4.2 Audio

The FBC offers two designated means to output audio from the software: two external stereo speakers that export sound to the surrounding area of operation, and also via an audio jack port that can be paired with external headphones to provide a closed audio system designated only for the headphone operator. A specification for the FBC design is to externally output the device audio via the stereo system speakers by default, but to only supply audio through the audio jack port when a device is plugged to it, thus muting the stereo speakers until the auxiliary listening device has been removed.

4.2.1 Speakers

We will be using two external speakers in the FBC audio stereo system, each positioned on either side of the device to maximize the audio dimension output to the user. When choosing our device speakers, we reference previous handheld gaming models as reference to determine what is an acceptable means of output in terms of frequency and power rating. As previously mentioned, the standard for speaker impedance in handheld devices is 8 ohms. Frequency ranges for the SNES games can rate up to 8 to 12 Kilohertz, so we will want our speakers to be able to attain that maximum frequency output. Power consumption is a main concern, with minimal current draw from the power source being desired, especially given the use of dual-stereo speaker components. Additionally, speaker size in relation to the case design is considered. We are looking for a traditional round speaker design, with a minimum diameter of 22mm, which is the speaker diameter for the classic Gameboy Color, and a maximum diameter of 30

mm. Below, in Table 4.2.1.1, we compare available small-scale output speakers for consideration.

Model	Frequency Range	Power Input	Diameter
102-2502-ND	448 Hz – 7 KHz	0.3 W	20 mm
668-1231-ND	500 Hz – 20 KHz	1 W	28 mm
102-1554-ND	530 Hz – 20 KHz	0.1 W	27 mm

Table 4.2.1.1 - Stereo Speaker Model Specs

Comparatively looking at the selected models, we first consider the frequency range. The 102-2502-ND can attain the lowest available frequency of 448 Hz, but also only can reach 7 KHz. The other two models can reach up to 20 KHz, well within the maximum frequency output range desired. For power consideration, the 668-1231-ND takes the most power, at 1 Watt, much more than the other models, and more than ideal for power consumption. For diameter dimensions, the 102-2502-ND is the most efficient in size, which can be ideal for extreme case design constraints. However, considering all fields together, we choose the 102-1554-ND model. This model requires the smallest input power at 0.1 W, and is in the ideal frequency output range. Factoring in that the FBC will be using two stereo speakers for audio output, this model fulfills more of the desired requirements for the sound system.

4.2.2 Audio Jack Output

The secondary source of audio output is from the implementation of an exterior audio headphone jack. When an auxiliary device, such as a headphone plug, is inserted into the audio jack, a connection is made that sends the audio output, from the amplifier, to the headphone output. Additionally, when the plug connector of the auxiliary device is inserted into the jack, an internal switch is flipped, breaking the connection to the dual external stereo speakers, and only providing audio output to the plug-in. The FBC will be featuring the audio jack primarily for headphone plug-in use, so we will use the standard 3.5-mm jack size for our design. To be noted, the core Raspberry Pi2 for the FBC initially featured a 3.5-mm jack already attached, but we removed that feature from the surface, as it was not needed for our device implementation.

For audio jacks, phone connectors are used as the input to channel audio signals. In particular, stereo headphone jacks comprise of three contacts: the tip, ring, and sleeve (ground). Each of the three connectors directly corresponds with the audio jack when plugged in. There are two types of headphone jacks for device use consideration: open circuit audio jack, and closed circuit audio jack. Open circuit jacks, when unoccupied by a connector, initially serve as a connector between the amplifier and the speakers. When a phone connector is inserted, the tip of the connector also receives audio output, and both the stereo

speakers and the headphones output audio. Closed circuit jacks also serve the initial purpose of connection between the amplifier and speakers. However, a moveable internal switch within the jack port keeps this connection by remaining closed. When a headphone connector is inserted, the switch is flipped open, breaking output connection to the speakers. Now, the only audio output is directed to the plug-in headphones. Figure 4.2.2.1 below exemplifies the headphone connector opening the jack switch.

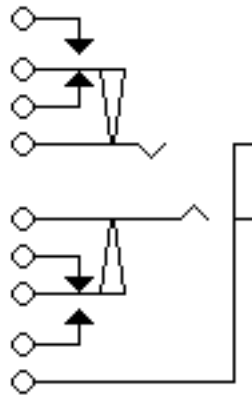


Figure 4.2.2.1 - Closed Circuit Audio I/O (Permission Pending)

When the auxiliary headphone connector is inserted, from the right side of the diagram, into the audio port, two pin connectors are pushed outward. When pushed, the two connectors move the mechanical dials on either side away from the initial arrow contact to the exterior contacts. This breaks the connection, cutting signal away from the speakers, and only applying audio output to the jack connector. Below, in Figure 4.2.2.2, is the audio circuit output in relation to speakers and audio jack, using an NMOS inverter to signal power to the respecting audio outputs. Note, due to schematic limitations in Multisim, only the symbol for a single input, single output audio jack was available. However, we can still simulate the stereo system dual-speaker audio jack by placing two single jacks side by side in series to more closely resemble the audio schematic from Figure 4.2.2.1

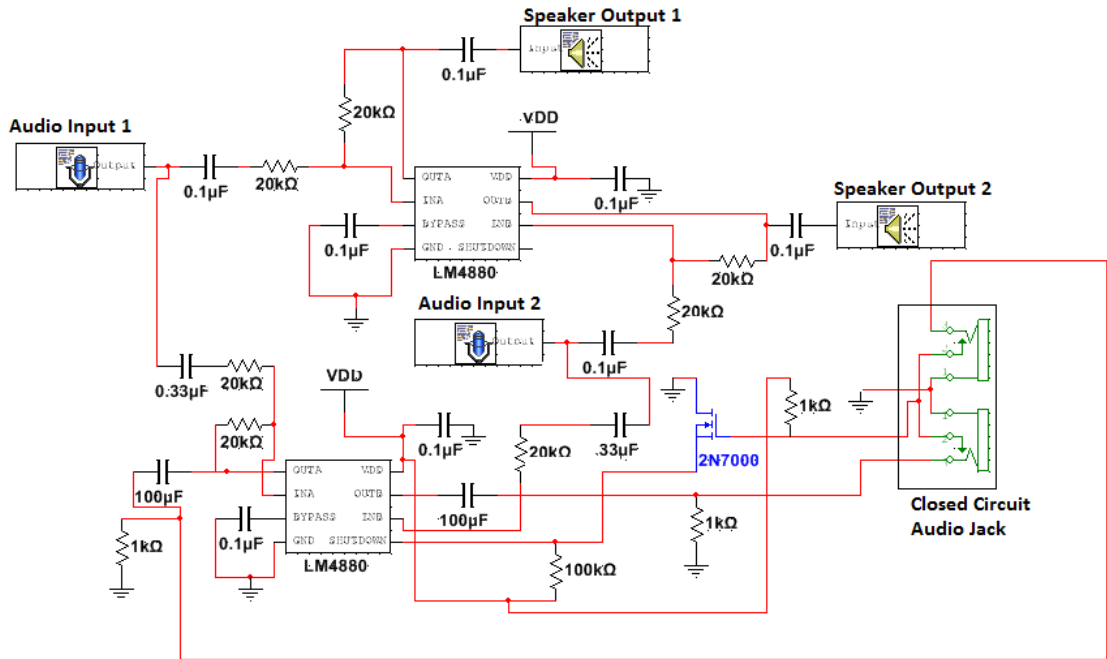


Figure 4.2.2.2 - Closed Circuit Audio Speakers and Audio Jack

4.3 Power System

The power system design includes the design of all the circuits related to the power subsystem. This consists of the battery wall charging circuit, DC-to-DC converter circuit, 5-volt power supply circuit, solar panel charging circuit, and LED battery indicator circuit. Also included in this section is the design of a relay circuit that will switch from solar charging to wall charging when the wall charger is plugged in.

4.3.1 Wall Charging Circuit Design

Figure 7.3.1.3 showed a sample 500 mA LiPo battery charging circuit using the MCP73831. We wanted to make a circuit very similar to this but with multiple LEDs to show the various states of the battery during charging. According to the datasheet, the STAT pin has three possible states: High when the battery is fully charged, Low when the battery is charging, and High Z when the battery is not connected or the MCP73831 enters its shutdown mode. In Figure 7.3.1.3 the diode will light when the battery is charging because the voltage difference between V_{in} and STAT will be positive and the current through the diode will cause light. When the battery is fully charged the STAT pin will be High and no current will flow through the LED. In order to have an indication of when the battery is fully charged we can add a resistor and LED from STAT to ground, as shown in Figure 4.3.1.1, or we can use a red/green LED and attach a resistor from the STAT pin to ground. We determined using a second LED was better because it would allow a third option for lighting both LEDs simultaneously. Three

combinations of LED lighting correspond to the three states of the STAT pin as shown in Table 4.3.1.1. When the STAT pin is in the High state the voltage drop from STAT to ground through the LED resistor will cause a current to flow through the LED lighting it. In the High Z state of the STAT pin both LEDs will light because there is a path from V_{in} to ground through the two LEDs and their resistors which will ignore the high impedance path into the STAT pin.

USB 5V Input

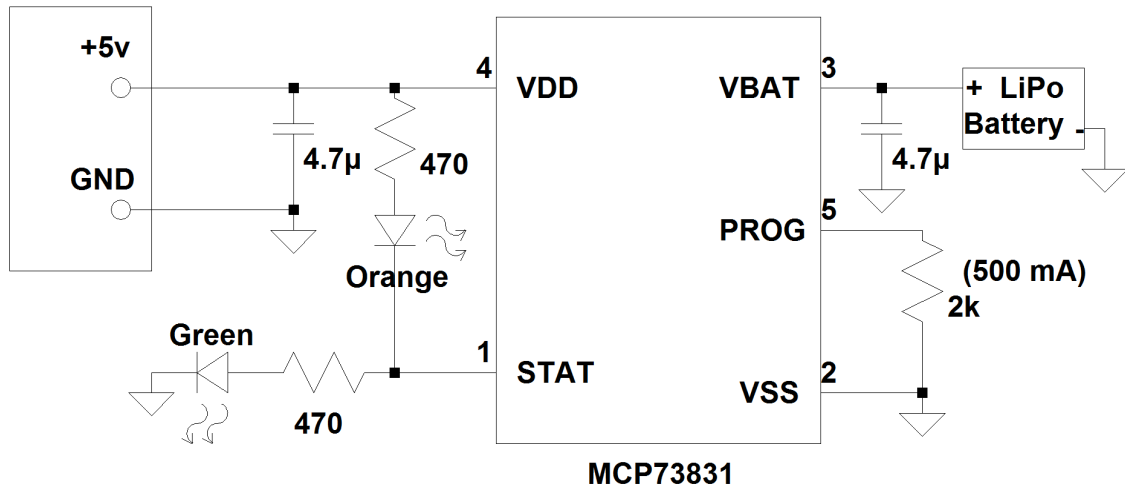


Figure 4.3.1.1 - Schematic of Charging Circuit with Charge State LEDs

Charge State	Orange LED	Green LED
Charging	Yes	No
Finished	No	Yes
Shutdown/No Battery	Yes	Yes

Table 4.3.1.1 - Table of Active LEDs per Charge State

This circuit will safely charge the battery using the standard 5 volts provided by USB 2.0 ports plugged into a wall adapter or a computer, and will show the user the current charging state of the battery. The total cost of the circuit is less than \$3.00. The circuit also does not have many components, which will help keep the PCB as small as possible. With the additional charge protection circuit built into our battery, we feel very confident that our LiPo battery will be able to charge without any major complications. Now that the battery can be charged we need to use it to power the device. First the output of the battery needs to be converted to a higher voltage level and then regulated.

4.3.2 DC-to-DC Converter Circuit Design

Figure 3.2.8.10 showed a sample regulated 5-volt output circuit using the TPS61030. We wanted to make a circuit very similar to this circuit, but with an additional switch to connect and disconnect the battery to the enable pin on the TSP61030. This will act as our power switch. Additionally when the device is turned on and working we want an LED to shine. We also want to use the LBO pin to light an LED when the battery is close to 3 volts to indicate that the battery is low and the user should turn off and/or plug in the device as soon.

All references to components in the following circuit design information will be referring to the circuit in Figure 3.2.8.10 until stated otherwise. The output voltage of the device is determined by the resistors R3 and R4 and the voltage at the FB pin. The TPS61030 datasheet says the resistor R4 should be around 200 k Ω and that the FB pin voltage is typically 500 mV. Using R4 = 200k Ω and VFB = 500 mV we calculated that R3 needs to be 1.8 M Ω . The LBO pin is active low when battery voltage drops below the set level and high otherwise. The set battery voltage cutoff is determined by the resistors R1 and R2 and the onboard LBI threshold voltage. The resistor R2 is supposed to be around 500 k Ω and the LBI voltage threshold is 500 mV. Using R2 = 510 k Ω and VLBI-Threshold = 500 mV and VBAT = 3 V we calculated that R1 should be 2.55 M Ω . Using a 2.7 M Ω resistor instead makes the LBO pin go active low when VBAT = 3.15 volts. This is suitable to our needs. The datasheet says that in typical applications, such as ours, an inductor with inductance 6.7 μ H is recommended. C1 is recommended to be 10 μ F by the datasheet. It also says to put a 100 nF capacitor in parallel with C1 and as close to the chip as possible. The output capacitor is recommended to be 220 μ F. A 2.2 μ F capacitor at the output will be used as a decoupling capacitor. The basic 5-volt output DC-to-DC converter circuit calculated from the datasheet without any additions is shown in Figure 4.3.2.1.

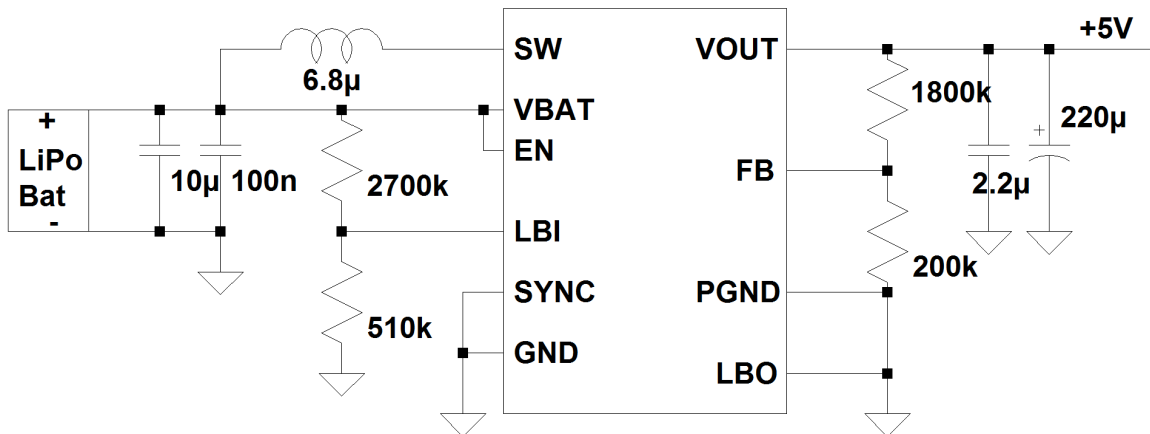


Figure 4.3.2.1 - DC-to-DC Converter and Regulator Circuit

4.3.3 Power Supply Design

Now that the basic circuit design has been handled we can add what we need to it. The first thing we need to add is a power switch that will switch the EN pin from the battery to ground. This is demonstrated in Figure 4.3.3.1. Also shown in Figure 4.3.3.1 is a blue LED attached with a resistor from V_{OUT} to ground as a power indicator. When the device is on the LED will be lit. The last major change is the addition of an LED low battery indicator. The LBO pin outputs V_{BAT} when it is High, so a PNP transistor with the emitter connected to V_{BAT} , the base connected to LBO, and the collector connected to a resistor and the LED. When LBO and the emitter are both V_{BAT} the transistor will be off and no current will flow through the LED. When LBO is low the transistor will turn on and current will flow through the LED. A 6 k Ω resistor will be attached between LBO and the base of the PNP to limit the base current. Figure 4.3.3.1 shows the power supply circuit that connects the battery to the Raspberry Pi 2 with both a power indicator LED and a low battery indicator LED.

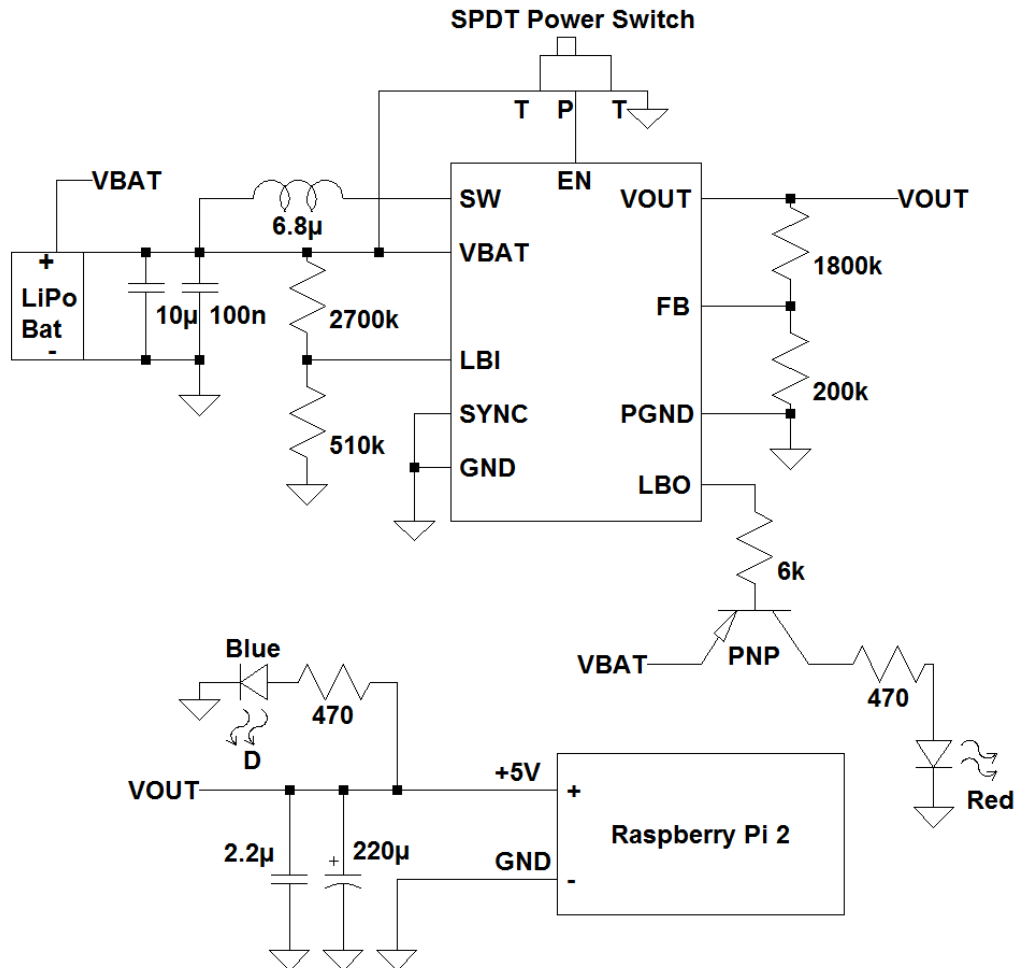


Figure 4.3.3.1 - Power Supply Circuit

4.3.4 Combining the Power Supply and Charge Circuit

The power supply and charge circuit will join at the battery. Some changes to the circuits may be needed due to coming in contact with additional components. The charging circuit has the battery in parallel with a 4.7 microfarad capacitor while the power supply circuit has the battery in parallel with a 10 microfarad capacitor and a 0.1 microfarad capacitor. These capacitors make the 4.7 microfarad capacitor unnecessary and it will be removed when joining the circuits. The charger circuit also has a 4.7 microfarad capacitor on the other side of the IC. This capacitor will be changed to a 10 microfarad capacitor for symmetry. No other changes were needed to join the circuits. The completed battery, charger, and regulated power supply circuit is shown in Figure 4.3.4.1.

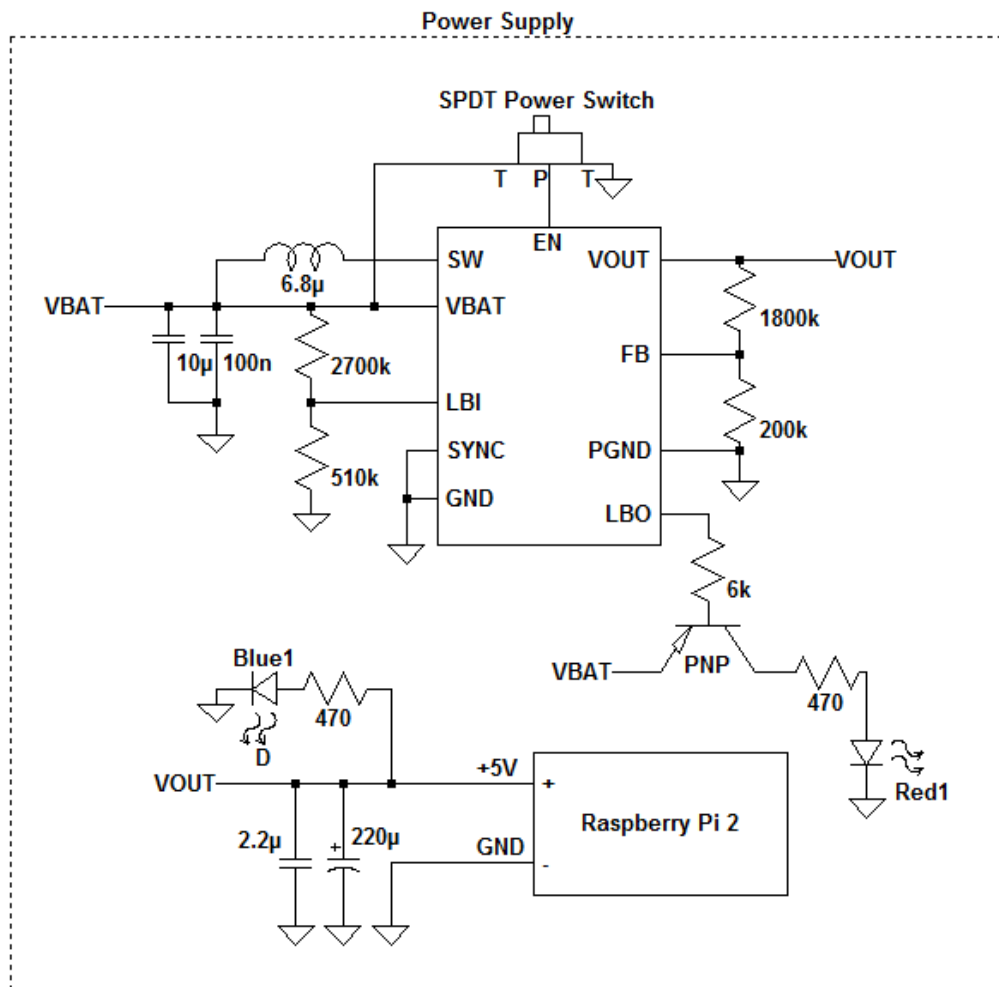
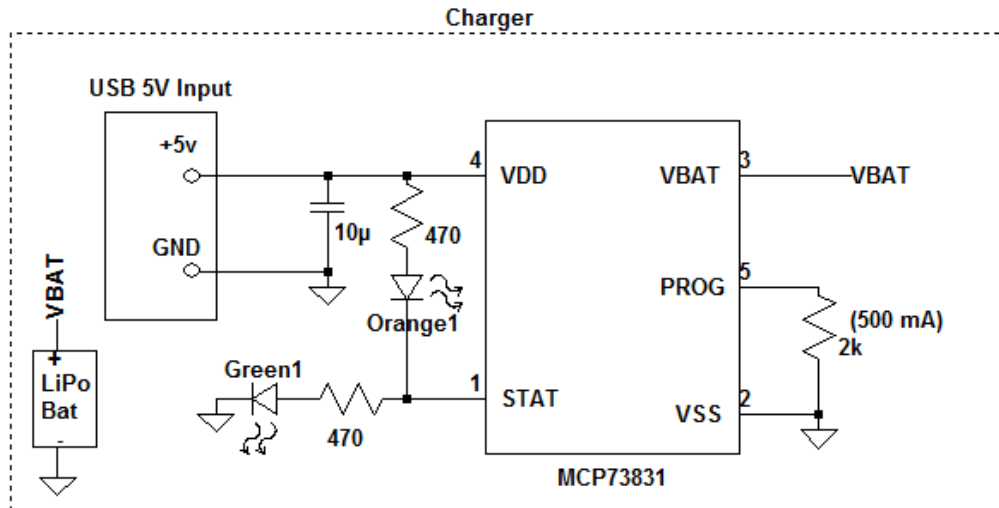


Figure 4.3.4.1 - Complete Power Supply, Charger, and Battery Schematic

4.3.5 Solar Panel Charge Controller Circuit

Comparing all three charge controllers, each one meets the initial concern for constant voltage and current supplied to the lithium polymer battery. The LT3652 is a moderate-powered controller that requires a minimum input voltage of approximately 5V, which is close to the maximum output of the solar circuit. Additionally, the LT3652 has the largest amount of pins, at 12 pins, that are excessive. The SPV1040, on the opposite spectrum, is targeted for lower power inputs, and raises concern for the compatibility of charging our 4.2V source battery. Thus, we choose the bq24210, which has an ideal input voltage range, and included protection from significant current leakage. Additionally, many sample solar charging lithium battery circuits researched have employed the use of the bq24210, making it an ideal component. Figure 4.3.5.1 below shows the solar cell charging circuit, with the bq24210 used in connection with the battery source.

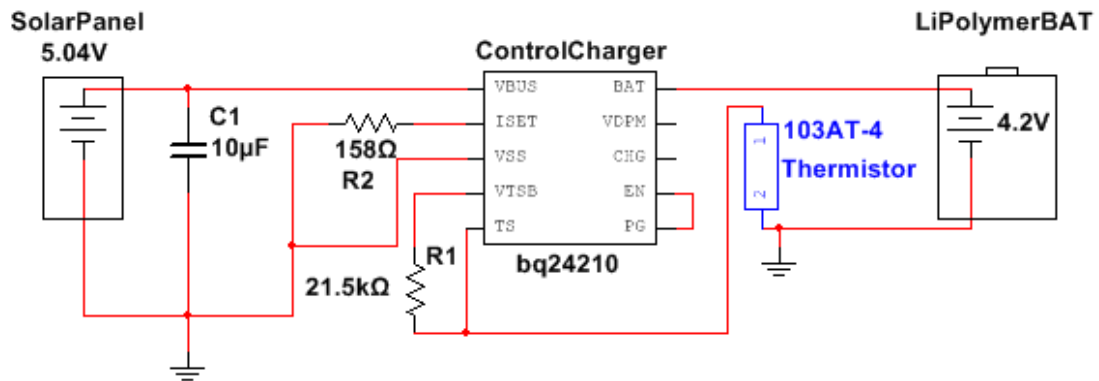


Figure 4.3.5.1 - Battery Control Charger Circuit

For the bq24210, we added a 103AT-4 thermistor close to our charging battery, to take advantage of the thermal protection system. Since the lithium polymer battery does not have a thermistor built in to it, we can add an external one close to the battery. If the battery begins to overheat, the TS pin will trigger the control charger to reduce the voltage and current flow to the battery. Pins VDPM and CHG are designated for charging detection signals, such as an LED connection for battery status. Since we are already using a more detailed battery indicator LED circuit, we will not be using these pins.

4.3.6 LED Battery Charge Indicator

A specification for the FBC was to design a light-based indicator of battery supply levels available to the device. Voltage regulated LED circuits enable a test current to be sent from the battery power source to the designated diodes, with the LED output response indicating the charge status of the battery. The FBC

has two separate LED battery charge indicators. The first indicator will use a RGB LED close to the user's targeted line of vision while operating the device, next to the upper right corner of the resistive touchscreen. This indicator will utilize each of the red, green, and blue LEDs to indicate the current battery charge status while the device is powered on. Additionally, the LED circuit diagram will be managed to have only one of the three diodes on at one time.

When the FBC is powered on, a current will be generated across each corresponding resistor to the LED. The blue LED will illuminate when the battery charge level reads 100%, indicating that the battery is fully charged. Should the user be charging the FBC's battery while the system is in use, the blue LED of the RGB LED will alert the user that the battery is fully charged, and may be removed from the battery charger. The blue LED will turn off at any battery charge below the maximum 4.2V charge of the lithium-ion battery. When the battery is charged between 100% and 25%, the green LED will illuminate, glowing brightest at the higher battery percentage charge, and dimming as the charge percentage reduces. Once the battery charge level reaches below 25%, the green LED will turn off. Simultaneously, when it has been indicated that the battery charge is below 25%, the red LED will illuminate. This final LED will stay on until the battery shuts down at 3V, in which all three diodes of the RGB LED will be powered down, along with the device.

The secondary charge indicator is designed for the user to more precisely identify the remaining voltage in the source battery, without needing to power on the entire device for a status check. On the bottom side paneling of the FBC, a line of 4 green SMDs will be connected in parallel, with the power supply connection delivered via push-button action, located just left of the SMDs on the hardware. While the button is pushed down, a switch bridging the battery power source to the SMD's is closed, sending current across the limiting resistors and to the SMD's. At a battery charge reading ranging between 100% and 70%, all three SMD's will be on. Following, at a charged range of less than 70% to 40%, the rightmost SMD, SMD3, will turn off, indicating that the battery is roughly two thirds charged.

4.3.6.1 LED Microcontroller

The push-button battery indicator LED feature is intended to have the SMD LEDs turn on in relation to the charge status of the source battery, and stay illuminated for the specified time of five seconds. Initial designs would accomplish simply using a push-button feature to allow for a current signal to send to the LED circuit for as long as the button remained pressed, and using Zener diodes and current limiting resistors to control the allowable charge flowing through the LEDs. However, this method does not address the specification of keeping the LEDs lit without continuously pressing the push-button, or, in the design case, keeping the LEDs lit for 5 seconds after the push-button has been pressed, signaling the battery status. Therefore, the implementation of a programmed microcontroller can

be used. Now, with the push-button component directly connecting the power supply battery to the input pin, the corresponding output voltage can be signaled to the three SMD LEDs, and the initial programming will instruct each LED status to remain active for the specified time. In terms of choosing which microcontroller will be suitable to achieve the battery status indicator, we primarily look for meeting the required pins needed and acceptable voltage supply to keep the microcontroller on.

The FBC power source is rated at 5 volts, so a microcontroller within that range would be suitable. We determined that 3 SMD LEDs will be needed for the indicator status: 3 LEDs lit for a battery charge range of 70% to 100%, 2 LEDs lit for a battery charge range of 40% to 70%, and 1 LED lit for a battery charge range of 10% to 40%. Thus, 3 pins from the microcontroller are to be utilized, as well as an additional pin each for the turn-on controller voltage, the push-button battery status signal, and ground. So, the minimum pin number we are looking for in our microcontroller is an 8-pin device. We can look at several models to achieve the design circuit specifications. An 8-pin MSP430G2210 offers over 2 KB of flash memory and 4 general-purpose pins. However, like many in its family, it is an ultra-low power controller that would need the voltage source input to be regulated to a lower value. A similar microcontroller to look at is the ATtiny module. Comparatively, the ATtiny 13 and ATtiny 25 both offer 8 pins and a Vcc input of 5 volts. The ATtiny 25 also offers 2K Bytes of programmable flash memory, which is an excessive amount needed for the battery indicator LED circuit, so the 1K Byte ATtiny 13 serves better to avoid resource overhead.

In addition to monitoring acceptable turn-on voltage to the microcontroller, consideration for the battery test current to be read must also be observed. As previously mentioned, flagging the battery status by the remaining source voltage is an ideal method for turning on the corresponding LEDs. However, with our source lithium polymer battery, voltage change is hardly detectable until the battery has nearly discharged; this will result in all 3 LEDs being lit, but quickly dropping from the just 2 lit, and finally 1 LED lit, until the low power LED turns on, in a very short amount of time. A more accurate reading for the lithium polymer battery is via a test current signal, and can be accomplished by current updating microcontrollers. The microcontroller tests the battery current draw against the battery cell capacity, predicting the remaining charge left in the battery. Battery gauge microcontrollers are able to test current, voltage, and temperature status from the source battery, being an ideal candidate for the battery source indicator circuit. In particular, the TI bq27200 gauge series microcontrollers are lithium battery-specific, and can accurately read the remaining battery voltage via a small test current across a designated flag resistor. This model is ideal, in that it, like the ATtiny series, is 8-pins and draws a small amount of current from the battery for continuous operations. The designated status current can be outputted from the microcontroller's specified pin, and be channeled across the Zener-LED series branches to signal which SMD LED to turn on for indication.

Below, in Table 4.3.6.1, we compare the three discussed microcontrollers for consideration and comparison.

Model	Supply Voltage	Active-Mode Current	Number of Operation Modes
MSP430G2210	1.8V to 3.6V	220 μ A	5
ATtiny13	2.7V to 5.5V	240 μ A	2
bq27200	2.6V to 4.5V	< 90 μ A	5

Table 4.3.6.1 - Battery Status Controller Comparison

From the three considered parameters, we see that the MSP430G2210 consumes the less voltage needed for controller operation, while the ATtiny13 requires the most, above the readily available 5V pin from the power system. For the supply current needed for the controller to be in active, or turn-on, mode, the bq27200 uses a significantly less amount of current, compared to the MSP430G2210 and ATtiny13. This is desirable for channeling current to the controller without worry of drawing excess current through the rest of the gauge circuit. The number of operation modes is considered for use in application of when the battery status is actually being called upon.

The ATtiny13 has only two modes, active mode, and power down mode, which fulfill the basic “on” and “off” conditions for the controller. However, the MSP430G2210 and bq27200 have 5 modes of operation. In addition to active and power down mode, the bq27200 also features a hibernate and data retention modes, which are ideal for saving battery status data when low input voltage is applied to the controller, but still enough to be in a transient stage without yet being powered down. The fifth option, ship mode, can be ignored, as it only pertains to the controller’s status for manufacturing purposes. The MSP430G2210 also has 5 modes of operation, all being different stages of low power. With each level of input power decreasing, the MSP430G2210 has a mode that systematically shuts down particular features, such as certain timers or CPU operations, that may not be immediately essential for constant operations.

Comparatively, we choose the bq27200 for the FBC battery indicator controller. The operational supply voltage is in the ideal range provided from the internal power source, and the low supply current in active mode is extremely useful. Additionally, the bq27200 features a pin specifically for detecting overheating from the battery via a thermistor, which, when flagged, will decrease power to the controller, or shut it down completely.

The battery indicator circuit will be programmed to read a test current from the source lithium polymer battery when the push-button closes the circuit switch connecting the battery to the input pin. The corresponding output current at the pins connecting the 3 LEDs is 40 mA, so current-dividing resistors will be needed

to limit the current down to 20 mA for the maximum forward current allowed by the LED. In this case, 100-ohm SMD 660-SL1TTE101J resistors will be sufficient. Once the test current has been read from the battery, and the corresponding signals turn on the correct LED indicators, the LED(s) should remain illuminated for a 5 second period, after which the controller will reset, and wait for the current reading from the battery when the push-button is pressed again. Figure 4.3.6.1 below demonstrates the SMD LED push-button battery indicator circuit, with current-limiting resistors and voltage-regulating Zener diodes included.

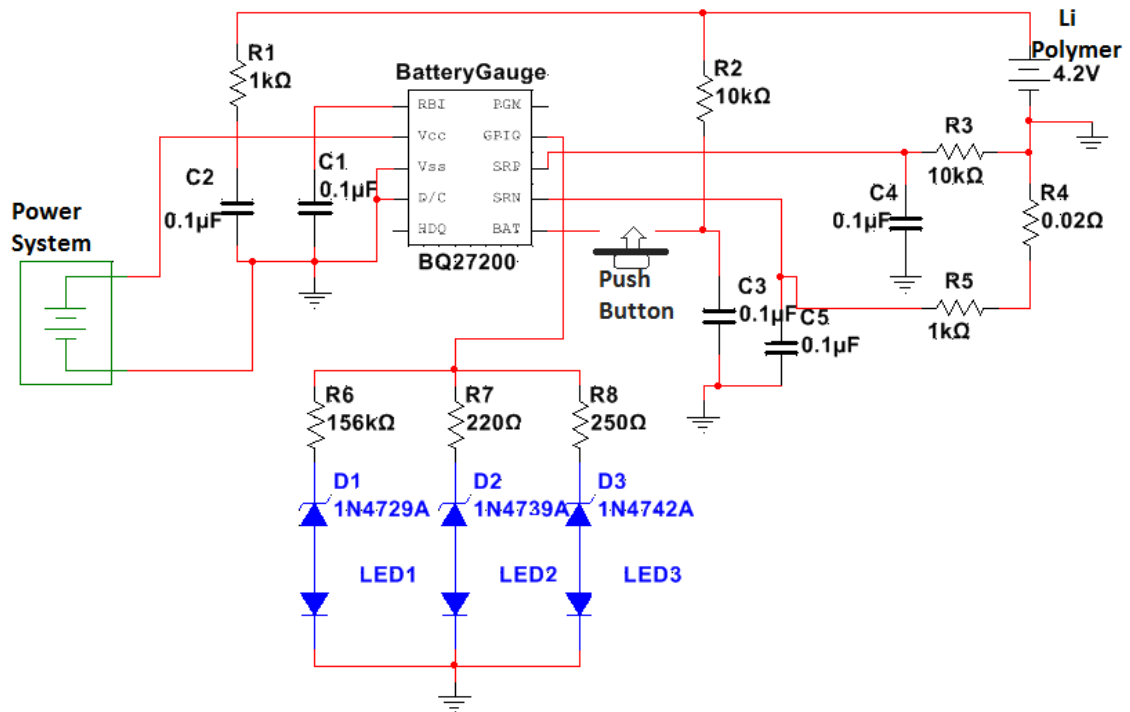


Figure 4.3.6.1 - Battery Status Indicator SMD LED Circuit

4.3.7 Switching Battery Chargers

In this design, we do not want the solar cell charger to be charging the source battery at the same time as the wall-charger. In effect, the solar charger is more of a secondary auxiliary charger, and the wall charger is to take priority when introduced to the system. To accomplish this, a relay circuit will be employed to switch from solar charging to wall charging when the USB charger is inserted to the power supply. For this, the KS2E-M-Series double throw relay will accomplish the task, with a switch voltage at the wall charger's 5-volt value. In functionality, the relay will have its power source, or its coil voltage, at the source input for the wall charger. Additionally, the double throw input pins will be set to both the wall charger circuit and the solar circuit, and the output pin to the lithium polymer battery.

Initially, when the wall charger is not in play, the coil voltage is at zero, and the relay circuit will be “open”, connecting the solar panel charging circuit to the battery source. When the wall charger is introduced to the system, the relay coil will turn on, and the internal switch will “close”, switching from the solar circuit connectivity to a direct connection between the wall-charging circuit and the source battery. Once the wall charger is removed, the relay will power down, and the switch will return to the solar charging circuit. Figure 4.3.7.1 below represents the two charging circuits connected to the relay for connection to the source battery. Initially, the relay connects the solar charger to the battery, but when the wall charger is introduced, the relay’s internal switch flips to break the solar circuit’s connection with the battery, and establishes a connection between the wall charger and the battery.

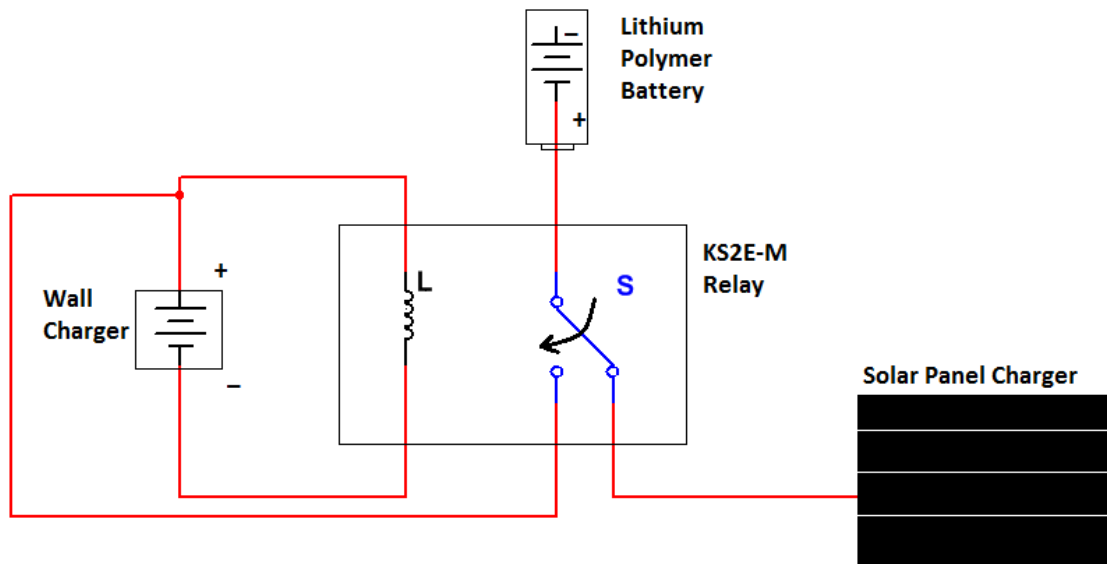


Figure 4.3.7.1 – Relay Switching Diagram

5. Prototype Construction

The prototyping of the project is incredibly important. We will lay out an overall plan for acquisition and integration in order to ensure that the building of the FBC goes as smoothly as possible. This section will first outline a plan for acquiring the hardware needed for the project, followed by a high level description of the integration of the various hardware components. A more in depth plan for combining the hardware components will follow the high level description of hardware integration.

5.1 Hardware Acquisition

Table 5.1.1 details the acquisition status for the hardware components for the project. If a project has not yet been acquired, a plan for acquisition will be determined.

A full list of the hardware components that we will need is outlined in Table 5.1.2. This table lists the various hardware components that we have selected for our project and the amount that we need for our project.

Component	Acquisition Status	Acquisition Plan
Resistors	Not Acquired	Order from DigiKey
Capacitors	Not Acquired	Order from DigiKey
Inductors	Not Acquired	Order from DigiKey
LEDs	Not Acquired	Order from DigiKey
LCD Screen	Acquired	
Raspberry Pi 2	Acquired	
ICs	Not Acquired	Order from DigiKey
Batteries	Not Acquired	Order from Adafruit
Plug Headers	Not Acquired	Order from DigiKey
Transistor	Not Acquired	Order from DigiKey
Buttons/Switch	Not Acquired	Order from DigiKey
PCB	Not Acquired	Order from OSH Park

Table 5.1.1 – Component Acquisition Information

Component	Quantity
Raspberry Pi 2	1
LCD Screen	1
ATTiny	1
ATMega	1
RN4020	2
SLMD121H8L	4
BQ24210	1
105-2502-ND	2
LM4880	2
3352T	1
2500 mAh LiPo Battery	1
500 mAh LiPo Battery	1
MCP73831	2
TPS61030	2
KS2E-M	1
BQ27200	1
MicroUSB Header	1
USB Header	1
Headphone Jack	1
PNP Transistor	1
Push Buttons	3
SPDT Switch	1
103AT-4 Thermistor	1
PCB	1
Resistors	Assorted
Capacitors	Assorted
Inductors	Assorted
LEDs	Assorted

Table 5.1.2 – Component List

5.2 Hardware Overview

In this section we will reduce and integrate the various hardware modules and to the best of our ability. The hardware modules contained in our system are the Bluetooth module, battery, solar charge module, wall charge module, power supply module, controller module, audio module, screen backlight control module, battery indicator module, the screen, and the Raspberry Pi 2. A high level design of the connection of these modules is shown in Figure 5.2.1. This design connects the smaller modules onto a single PCB which will connect with the Raspberry Pi 2, battery, and screen.

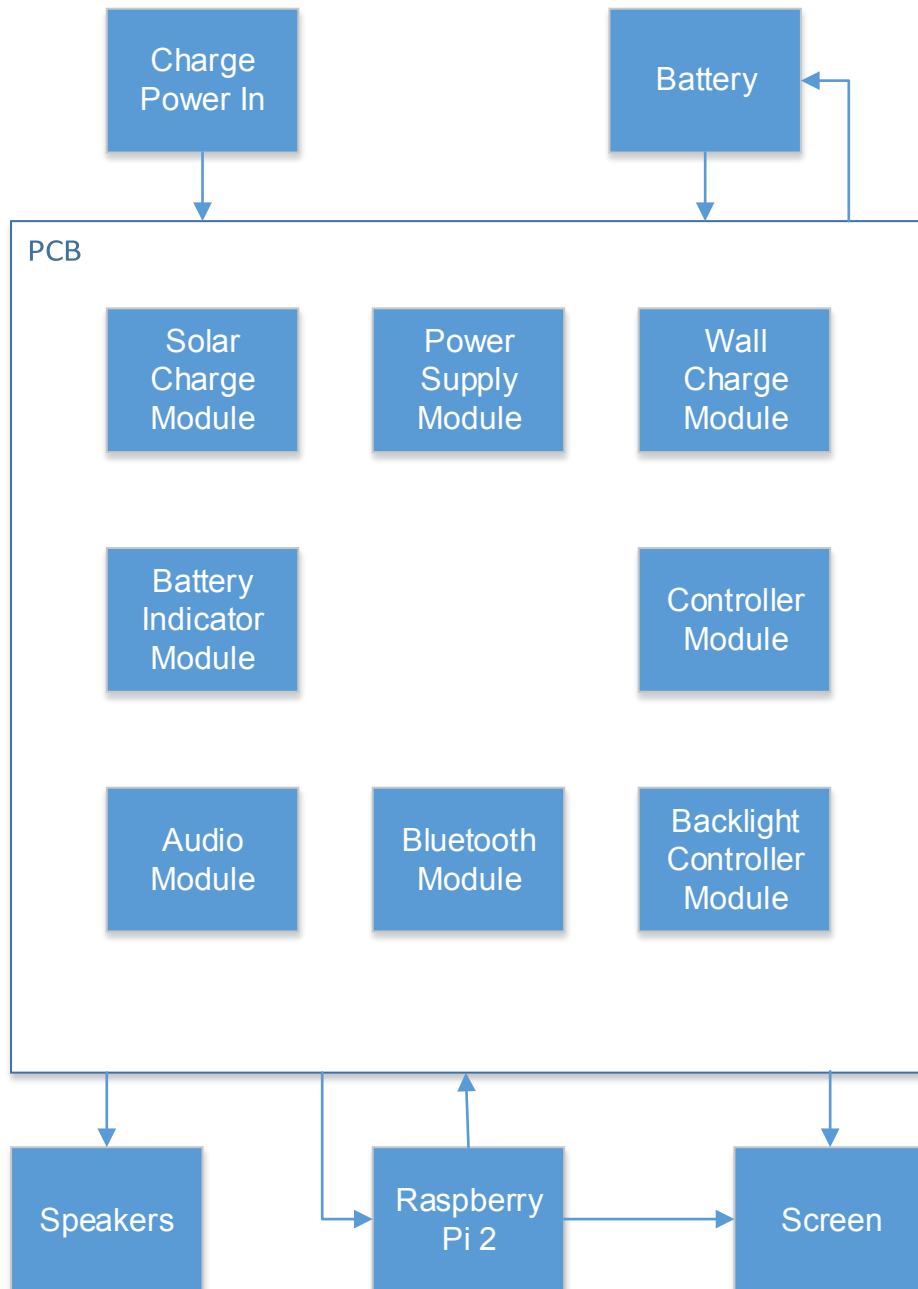


Figure 5.2.1 – High Level Hardware Integration

5.3 Hardware Integration

The actual integration of the hardware will begin with the design and acquisition of the PCB. The PCB will integrate the solar charge module, wall charge module, backlight controller module, power supply module, battery indicator module, controller module, audio module, and Bluetooth module.

5.3.1 PCB

The PCB will connect to the Raspberry Pi 2, the screen, the battery, and external power. The PCB will be small enough to fit in a portable case. Our goal in this section is to determine how we will design the PCB and how we will acquire the PCB that we designed.

5.3.1.1 Design

CADSoft Eagle is the most common software used in PCB design. There are a few other options that we explored, such as Fritzing, but ultimately we settled on making our PCB in the tried and true environment of Eagle. Eagle contains libraries of many components and also allows users to make their own components.

We will give our best efforts to make the PCB as minimalistic as possible in Eagle in order to preserve the portability of our system. Additionally, the PCB should have all necessary connections to the battery, Raspberry Pi 2, and screen easily accessible on the sides of the PCB.

Figure 5.3.1.1 shows the modules on the PCB interconnected and connected to the hardware components not on the PCB.

5.3.1.2 Acquisition

After designing our PCB in Eagle we will need to order the PCB. There are many manufacturers that will make and ship PCBs when sent designs. We decided to order our PCB from OSH Park. OSH Park boards only cost \$5 per square inch for 3 copies of a dual layer PCB. This is a reasonable price and will also leave us with two extra PCBs to work with if the first one becomes damaged. We estimate the board to be 5 square inches or less. This will result in the cost of the PCB being \$25 or less. OSH Park ships within 12 calendar days of ordering, which will allow us to begin testing our PCB in a timely fashion.

5.3.1.3 Mounting

We will tin the leads on the PCB and place the surface mount components on them. We will then use a hot air gun to reflow the solder on the PCB to attach the surface mount components to it.

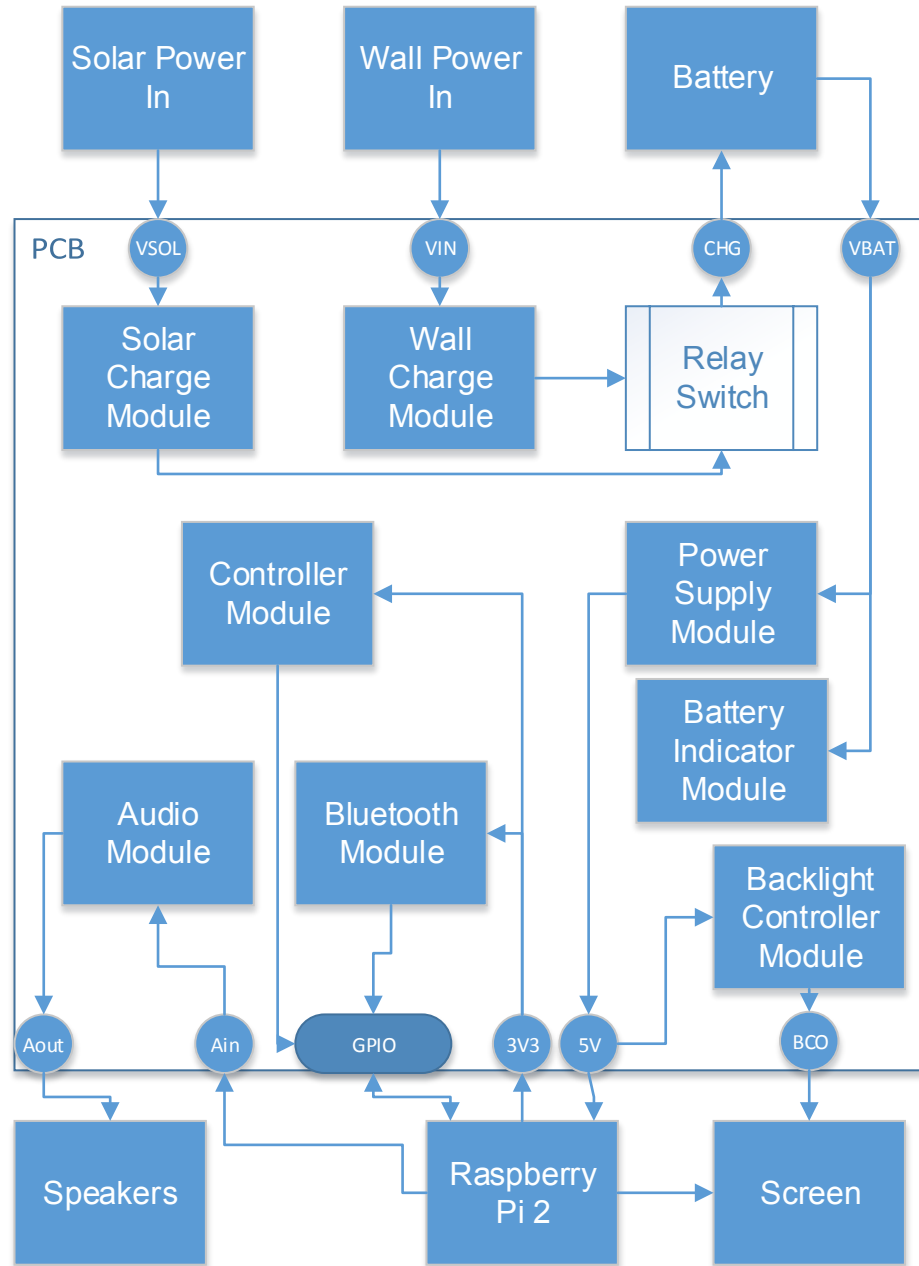


Figure 5.3.1.1 – High Level PCB Design

5.4 Software Overview

As seen in Figure 5.4.1, at boot the Raspberry Pi will automatically load the EmulationStation GUI. From there, user input from the controller, either external or internal, will choose the option desired: either an emulator or the settings application. From there the user can choose to use the emulator to play a game, change settings as desired, or return to the main GUI menu.

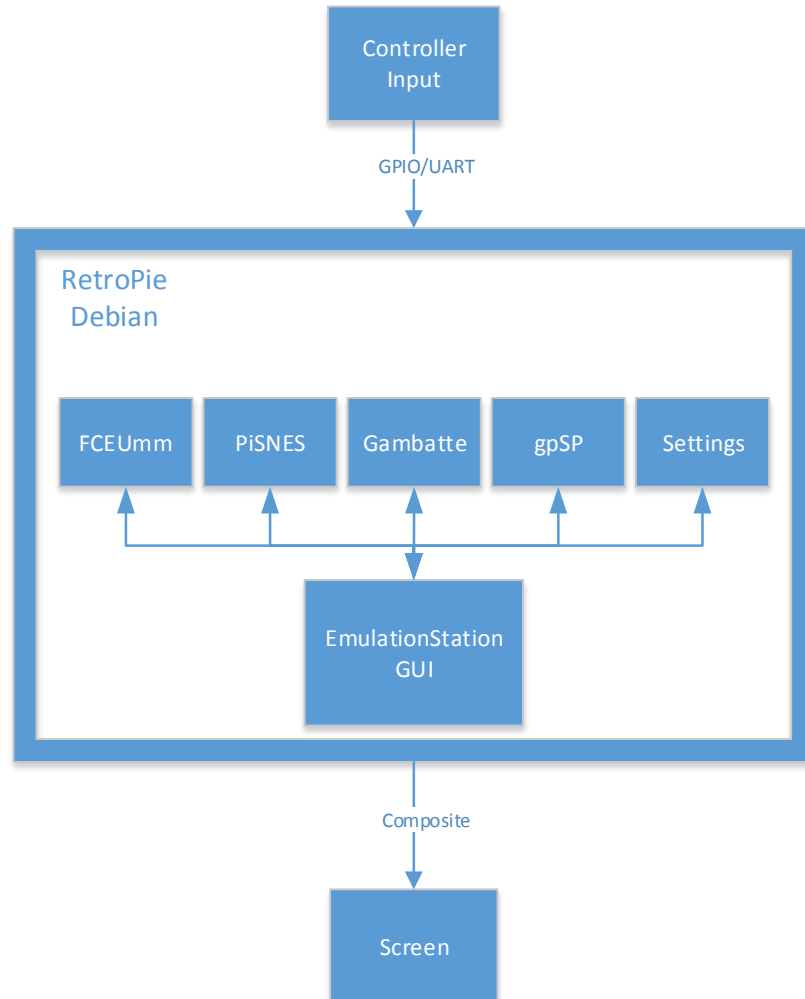


Figure 5.4.1 – Software Flowchart

5.4.1 Software Acquisition

In order to acquire our software, we performed the following steps:

1. Downloaded the latest Raspbian build (Debian Wheezy 3.18) from raspberrypi.org
2. Downloaded the latest RetroPie build (Version 3 BETA 2) from blog.petrockblock.com/retropie/retropie-downloads/
3. Used the installed software to download the emulators.
 - a. Gambatte, PiSNES, gpSP, FCEUmm.
4. Downloaded SNESDev, a controller handler.
5. Acquired game images (ROMs) by using a modchip on each console to rip the games from their original cartridges.
6. Acquired system BIOSs by using a modchip on each console to rip the BIOS from its respective console.

5.4.2 Software Integration

In order to implement the software into our project, we performed the following steps:

1. Wrote the disk image file of Raspbian to our MicroSD.
2. Ran Raspi-Config to expand filesystem and enable overscan, as well as default output to the composite video.
3. Installed RetroPie on our Raspbian build
4. Ran the RetroPie setup script to enable easy download and installation of emulators.
5. Installed emulators.
6. Installed SNESDev.
7. Rebooted the system to load the device stack.
8. Inserted a blank USB drive to copy file structure of ROM folder automatically.
9. Copied ROMs to USB drive and replaced it in the FBC to copy test ROMs to FBC's MicroSD card.
 - a. Non-removable after final assembly, thus this method is necessary
10. Confirmed basic functionality.

6. Prototype Testing

Extensively testing our prototype is exceedingly important to ensure that it works properly during its prolonged use and in standard operating conditions. In order to best determine what parts may be causing issues we have decided to test the device modularly. First we will test each hardware module individually. Then we will test the software. After the hardware modules and software have both been shown to work properly the, they will be added piece by piece to the device to test if it still works. Finally, the final combined prototype will be tested. This testing method will allow us to pinpoint issues as they come along

6.1 Hardware Testing

We will test each hardware module that will be placed on the PCB. We will also test the battery, Raspberry Pi 2, speakers, screen, and charger. All tests will be run in standard operating temperatures and normal lighting. The tests will be performed in such a way as to allow insight into why a particular component may or may not work in order to help our troubleshooting process.

6.1.1 Raspberry Pi 2

In order to test the Raspberry Pi 2 we will first simply plug it in and see if it turns on and outputs to a known working screen. If the Raspberry Pi 2 fails to turn on or output to the screen we will know that it needs to be fixed or replaced prior to it being used as the workhorse for our project. Additionally, the RP2 has two built in

test points labeled TP1 and TP2. Measuring the voltage drop from TP1 to TP2 should give a value between 4.75 and 5.25 volts. If the voltage shown is outside of that range the power being provided to the device may be the issue. TP2 can also be used to test the F3 Polyfuse on the RP2. If the voltage from one side of the F3 Polyfuse to TP2 differs from the voltage from the other side of the F3 Polyfuse to TP2 by more than 0.3 volts the polyfuse is most likely broken.

6.1.2 Screen

We will test the screen in two ways. First we will test the screen to make sure that it turns on and wakes upon receiving a video signal. If the screen works we will then attach various devices to the screen and compare the visual on the screen with the visual on a known working screen.

The screen will first be tested using the composite input and power cable that it comes with. The power cable on the screen will be connected to a standard 12-volt power supply. Both composite video inputs will be tested by connecting to the composite video output of a known working Raspberry Pi 2. If the screen turns on and displays the output of the Raspberry Pi 2 it will be considered functional. If not it will be determined faulty and a new screen may need to be acquired.

Next, the screen will be tested using the composite input to connect to a Raspberry Pi 2, a Super Nintendo, and a Nintendo 64. These same devices will be connected to a known working screens composite input and visually compared with the screen for our device. If there are any major discrepancies in the output visuals we must find ways to fix them or get a different screen.

After the screen has been removed from its case and no longer has its composite cables or power cable we will test the device in a different manner. We will salvage the composite video connectors from the old cables of the screen and use them to build an adapter on a breadboard. This way we will be able to connect the adapter to the composite pins on the screen circuit board and test different devices using the adapter. We will also create an adapter for power to the screen in a similar fashion, but due to the bypassing of the regulator on the screen circuit we will supply the screen with 5 volts instead of 12 volts.

6.1.3 Wall Charge Module

The wall charge module will be tested before the battery so that the battery will have a working charger to test with. The wall charge module will be tested by attaching it to a known working mostly drained LiPo battery and a known working USB cable and wall adapter. The charge module will be tested to ensure a few different things. The charge module will first be tested to see if it will charge the battery properly. The charge module will then be tested to ensure that the overcharge protection is working properly.

In order to test if the charge circuit properly charges a battery the first thing we will test is if the circuit outputs the correct constant current during the first part of the charge cycle. If the current output of the chip is not close to 500 mA then something is wrong with the circuit and it needs to be troubleshot. After the current output is determined to be correct the battery will be charged for 30 minutes. The battery will then be tested for a change in voltage across its positive and negative terminals. If the voltage has not increased the charger circuit needs to be fixed. The battery will then be reattached to the charger. The voltage output of the device will be monitored while the battery charges. Once the voltage output reaches the set limit of 4.2 volts the voltage should not increase any further. If the voltage stays at 4.2 volts then the constant current/constant voltage charging portion of the circuit works properly.

In order to test for overcharge protection we will continue the charging of the battery from where we stopped testing to see if the charging portion of the circuit worked properly. We will begin by measuring the output current instead of the voltage. The current should steadily decline until it reaches 5% of the charge current, around 25 mA. If the current continues below 25 mA then the overcharge protection is not working properly and needs to be fixed. If the current drops to 0 from 25 mA then the battery will be removed from the charging circuit. The voltage across the battery will be tested. If it is 4.2 volts the overcharge protection of the charge circuit works. If it is not 4.2 volts the circuit needs to be looked at and possibly reconfigured.

The tests will then be performed again while monitoring the status LEDs to determine if the status LEDs properly indicate their status.

6.1.4 Battery

The testing of the battery will be broken into two sections. First, the battery will be tested to ensure that it charges properly, discharges properly, and retains charge properly. Second, the internal overdischarge protection circuitry of the battery will be tested to ensure user safety in the end product. These tests will determine whether the battery meets our needs or not.

6.1.4.1 Battery Characteristics

The battery characteristics test has 3 parts: charging, discharging, and charge retention. The order of these tests depends on the initial state of the battery when it is prevented for testing. So the very first step is to check the voltage across the battery. If the battery is charged, 4.2 volts, the first test will be charge retention, second discharging, and finally charging. If the battery is initially discharged, ≤ 3 volts, the first test will be charging, then charge retention, and then discharging. If the battery is somewhere in between it will follow the same format as if it were initially discharged.

Testing charging will start by measuring and recording the voltage across the battery. The battery will then be hooked up to a known working charging circuit. The battery will be charged at a rate of 1/5 charge capacity for 30 minute. If the battery voltage has not changed the battery is faulty.

Testing discharging will also start by measuring and recording the voltage across the battery. The battery will then be hooked up to a simple discharge circuit through a resistor to drain the battery at a rate of 1/5 charge capacity for 30 minutes. If the battery voltage has not changed the battery is faulty.

Testing charge retention will yet again start by measuring and recording the voltage across the battery. The battery will sit isolated for 24 hours. If the voltage has changed more than a very small amount then the battery is faulty.

6.1.4.2 Internal Overdischarge Protection Circuit

The internal overdischarge protection circuitry will be tested by draining the battery normally until it reaches a voltage of 3 volts across its terminals. Then the battery will be carefully drained until it reaches 2.75 volts. If the battery continues to drain the internal protection circuit is faulty.

6.1.5 Solar Charge Module

The solar charge module will be tested in a very similar fashion to the wall charge module. A known working mostly drained LiPo battery will be connected. The solar charge module will be tested to determine if it charges the battery correctly. The solar charge module will then be tested to determine if the overcharge protection is working properly.

In order to test if the solar charge circuit properly charges a battery the first thing we will test is if the circuit outputs the correct constant current during the first part of the charge cycle. If the current output of the chip is not close to 200 mA then something is wrong with the circuit and it needs to be troubleshot. After the current output is determined to be correct the battery will be charged for 30 minutes. The battery will then be tested for a change in voltage across its positive and negative terminals. If the voltage has not increased the solar charger circuit needs to be fixed. The battery will then be reattached to the charger. The voltage output of the device will be monitored while the battery charges. Once the voltage output reaches the set limit of 4.2 volts the voltage should not increase any further. If the voltage stays at 4.2 volts then the constant current/constant voltage charging portion of the circuit works properly.

In order to test for overcharge protection we will continue the charging of the battery from where we stopped testing to see if the charging portion of the circuit worked properly. We will begin by measuring the output current instead of the

voltage. The current should steadily decline until it reaches 5% of the charge current, around 10 mA. If the current continues below 10 mA then the overcharge protection is not working properly and needs to be fixed. If the current drops to 0 from 10 mA then the battery will be removed from the charging circuit. The voltage across the battery will be tested. If it is 4.2 volts the overcharge protection of the charge circuit works. If it is not 4.2 volts the circuit needs to be looked at and possibly reconfigured.

6.1.6 Power Supply Module

The power supply module will first be tested to make sure that the module does not output any voltage when the power switch is off. Next it will be tested when on to ensure that the voltage output of the power supply is a regulated 5 volts no matter what the voltage across the battery currently is. The last test of the power supply is to ensure that the low battery LED turns on and that it does so at the correct time.

The power switch simply switches the enable pin on the DC-to-DC converter chip between VBAT and ground. First, without the battery connected, we will use a multimeter to check for continuity between the enable pin and ground when the switch is off and to check for continuity between the enable pin and VBAT when the switch is on. Then the same test will be done in reverse to ensure that the switch is not bridged. The battery will then be connected with the power switch off. The output voltage of the power supply should be 0 volts. If it is not then the power supply module is not working properly. Next the power switch will be turned on. The output voltage of the power supply should be 5 volts. If it is not very close to 5 volts then the power supply module is not working properly.

Now that we have determined the power supply puts out 5 volts, we need to test it with the different voltages the battery will reach in a charge/discharge cycle. The battery will be charged to 4.2 volts and connected to the power supply and the switch will be turned on. If the output voltage is not 5 volts the power supply is malfunctioning. If the output voltage is 5 volts the power supply is working properly. Next the battery will be discharged to 3.3 volts and connected to the power supply. If the output voltage is not 5 volts the power supply is malfunctioning.

The last test is to check that the low battery indicator works properly. The low battery light is supposed to turn on around 3.15 volts. The way to test this is to drain the battery to around 3.2 volts and then hook it up to the power supply and see if the light turns on. If it does it needs to be reconfigured. If it does not, disconnect the battery from the power supply and discharge down to 3.15 volts. Reconnect to the power supply and see if the light turns on. If it does not it is malfunction. If it does then the low battery indicator is working.

6.1.7 Battery Indicator Module

The battery indicator module will be tested by charging and draining a battery while attached to the battery indicator circuit. After the battery has fully charged we will drain the battery at 1/5 capacity for an hour. If the battery indicator does not show around 80% then it is malfunctioning. Likewise, this test will be performed hourly until the battery is drained, subtracting 20% capacity for each hour of the test.

6.1.8 Backlight Controller Module

The backlight controller module will be tested by attaching it to a known working screen and utilizing the buttons to increase and lower the brightness. If the brightness of the screen does not change then the module is malfunctioning. Otherwise, the backlight controller module is functioning as desired.

6.1.9 Bluetooth Module

The Bluetooth module will be tested by connecting it via UART to a known working Raspberry Pi 2. We will then attempt to connect a variety of Bluetooth devices (controllers, keyboards, mice) to the Raspberry Pi 2. If the devices find the Raspberry Pi 2 and vice versa then the Bluetooth module works. Otherwise the Bluetooth module needs to be fixed.

6.1.10 Controller Module

The controller module will be tested by connecting it to the GPIO pins of a known Raspberry Pi 2. First we will make sure that the controller is recognized by the system. Once the controller has been recognized by the device, we need to make sure that each button works as intended. Once all buttons have been tested and configured, a game needs to be played on the Raspberry Pi 2 using the controller to make sure that the response time of the buttons is adequate for the games being played, and that the controller does not suffer from noticeable ghosting.

6.1.11 Speakers

The speakers will be tested by connecting them to a known working amplifier and attempting to play some audio through them. If no sound comes out then the speakers are faulty. Otherwise the quality of this audio will be compared with the quality of audio output from other portable game consoles. If the speakers are of a similar or better quality then they will be considered working. Otherwise they will be considered faulty or inadequate.

6.1.12 Audio Module

The audio module will be tested by connecting it to a known working composite stereo sound output and a known working pair of 8 ohm speakers. If the speakers do not correctly output the composite sound, then the audio module is faulty and needs to be fixed. If the speakers do output the correct sound then the volume control will be tested by moving it up and down to change the audio output volume. If the level of sound does not change then the audio module is faulty and needs to be fixed. If the level of sound does change then the audio module will be considered working.

6.2 Software Testing

We will make use of both premade tools within the system as well as our own senses and knowledge to test all important aspects of the software. All tests will be run with the system in standard operating temperatures with adequate lighting for the screen, to reduce hardware bias.

The premade tools will allow us to test things that we can't judge with sight and sound alone, such as framerate. These will be the most objective tests, as they will rely on quantitative analysis.

Using our own senses will allow us to test things that cannot be judged purely quantitatively, such as artifacting. The metric for these tests will be simpler: if it looks or sounds wrong, it fails.

6.2.1 Emulator Tests

For each emulator, we will be testing three games. The first will be a low end game, the second will mid to high end game, and the third will be a high end game, pushing one or more aspects of the system to its limits.

In this way we can ensure that the entire spectrum of games will be supported, with a few exceptions that, for example, did not follow Nintendo's programming guidelines.

For general testing, each game will be checked for framerate, resolution, graphics artifacting, and audio glitches. Special features of each console, such as the SuperFX chip for the SNES, will also be checked.

Games must reach a minimum of 50 frames per second, which is the requirement for PAL games. NTSC games will be required to output at no less than 59.97 fps. Additionally, all games must display at full resolution (or higher) with no cutoff portions. Graphics must be free of artifacts to the point that the gaming experience is not interrupted. Audio must be free of glitches such that the music feels consistent.

Additionally, we will be testing the basic functions and usability of the GUI for the system, ensuring proper operation at all times.

6.2.1.1 Game Boy

For the Game Boy, the first game we will test will be Pokemon Red Version. Graphically non-taxing as it was, this was a solid game that spawned an empire. The graphics and the audio are immediately recognizable to almost anyone in our generation, making testing for accuracy easy.

Donkey Kong Land will be the second game tested. This game was an attempt by RARE to capture the spirit and style of the SNES Donkey Kong as accurately as possible. With detailed sprites and animation that simulated a 3D experience, this game is technologically demanding and visually pleasing. Additionally, we have experience playing this game, and thus know what to look for when it comes to testing.

Finally, we will look into a little known title called Faceball 2000. This is a port of an Atari multiplayer shooter, which resembles a very low quality version of Doom. With pre-rendered graphics, convincing pseudo-3D, first person view, and the insane support for 16-player deathmatch, this is a game that truly pushes the Game Boy to its limits.

6.2.1.2 Game Boy Color

We start off with a non-taxing, but still well made and excellent game: The Legend of Zelda: Oracle of Ages. This game did not have much in the way of technical requirements, but easily recognizable graphics and iconic sound from the Zelda franchise will make emulation mistakes easy to spot. We'd also like to test the "Game Boy Advance" special features, which are only accessible on a Game Boy Advance, to see if our system can fake that.

Cannon Fodder was a relatively obscure port of an Amiga game which, while most of the game wasn't anything special, had one main feature that set it apart. It had a PC quality full motion video to open the game, and used the largest cartridge size possible to store that. Many emulation systems have trouble emulating this video properly, due to the hardware wizardry required to have it work on a small system like the Game Boy Color. We think this will serve as a good, midrange stress test for our system.

Shantae is the final game we will test for the Game Boy Color. Commonly cited as one of the best games to come out of the GBC's final years, Shantae combines color and animation on par with early GBA games with advanced

lighting and scaling effects, as well as a rich and detailed soundtrack. These make the game perfect for pushing our emulated system to its absolute limits

6.2.1.3 Game Boy Advance

Wario Ware Inc. Mega Microgame\$ starts off our tests. Simple graphics, simple layering, simple audio, this game was maddeningly fun but by no means pushed the hardware to its limits. The gameplay is simple as well, taking place with 5 second long microgames, getting faster and faster as you progress. It will be easy to see if the game is being emulated properly just by measuring game time. If it's not the full five seconds, the game is clearly not being emulated properly and is experiencing slowdown.

Advance Wars 2: Black Hole Rising is a turn-based tactics game. It has simple graphics, but with advanced use of graphics layers as well as an AI that has to consider every possible move on a "board" far larger than a chessboard, the game still manages to tax the system. Two of the biggest things we will be testing with this game is the accuracy of the AI compared to the original system and the speed with which it computes optimal moves.

The Castlevania: Double Pack will be our final test game under the GBA. It combines two complex games, Castlevania: Harmony of Dissonance and Castlevania: Aria of Sorrow into one cartridge, increasing emulation complexity. With fast platforming, rich soundtracks, and gorgeous graphics, these games pushed the GBA to its limits. One thing we will be checking in particular with this title is the ability to properly load each game into memory, without sector overlap. This has been an issue with non-x86 emulation in the past, and we hope to make sure that is not the case here.

6.2.1.4 Nintendo Entertainment System

For the NES, you have to start with Super Mario Bros, one of the most iconic and beloved games of all time. Simple graphics, quick and easy gameplay, and a classic soundtrack, this game won't push the system by any means. It lacks the memory mapper chips of other, later NES games, making emulation easy by comparison. The main thing we will be looking at is if it controls as tightly as the original game, as the graphics and sound should be a non-issue. Additionally, we will check to make sure original glitches, like the Minus World, are present, as these will mean that the game is being emulated perfectly. There's no better way to check imitation than to see if the same bugs are repeated.

The Legend of Zelda comes up next. Fitting a massive world into the space of a normal NES cartridge just wasn't possible, and so this game was one of the first to take advantage of the MMC1 expansion chip. It allowed save games, a rarity

at the time, multi-directional scrolling, and a greater ROM size. As a result, emulation complexity increased substantially. With a rich and varied soundtrack, smooth graphic transitions, and the ability to support many enemies on screen at once, we feel this is a good game to test the mid-range capabilities of the NES. We will mainly be making sure that the memory mapper emulation capability works as intended, as the graphics and sound will most likely not be an issue beyond that.

Kirby's Adventure was a game that truly pushed the limits of the NES. First off, the largest cartridge of the NES, 6Mbit, was designed specifically for this game. Second, it used the MMC3, a more advanced version of the MMC1 which also implemented a scanline based IRQ counter that made layered scrolling easier, and even more ROM in the form of additional banks. This game could not be fully contained even on the largest cartridge available. Additionally, it pushed 8-bit graphics to their limits, with rich color, smooth animations, and even advanced features like parallax scrolling. This game pushed the system to its limits, and we are eager to see if our system can handle this game while continuing to run at full speed and with no issues.

6.2.1.5 Super Nintendo Entertainment System

Unlike the other controls, which either had one or no hardware add-ons, the SNES had eleven different types of so-called "enhancement" chips, ranging from trigonometric calculations to full simulated 3D graphics. The difficulty in emulating these various co-processors led to less accurate emulation for years while figuring out their inner workings. Additionally, the SNES uses advanced graphics rendering techniques, like Mode 7, which are not easy to duplicate. The SNES testing will mainly focus on making sure these capabilities function.

Tools built into EmulationStation and the RetroPie core of the FunBox Classic will allow us to test each emulator for its accuracy and speed in games on the system.

We will use the framerate monitor in EmulationStation, the resolution measurement capabilities of the RetroPie core, and use the logging capabilities of each emulator.

We start with Chrono Trigger, widely considered one of the best games of all time. With crisp graphics, a beloved soundtrack, and an advanced battle system, Chrono Trigger still holds its own against the games of today. However, we placed it in the low-end category due to its lack of any enhancement chips or special features beyond the base SNES gameplay. This is one of the finest, if not the finest, example of what a technically "low-end" game can be and its proper operation on our system is a must.

Star Fox was a shooter, although not your typical one. It was one of the first console games to adopt and take advantage of 3D polygons. This feat, before only accomplished by coin operated arcade machines dedicated to that purpose,

was made possible by the use of the Super FX chip. This graphics co-processor enabled true 3D graphics, albeit very low resolution ones. As a result, the rest of the system was free to handle the non-graphics tasks, and it made for a very fast paced game. The accuracy in emulating the Super FX chip will be our primary test here. Any kind of polygon artifacting or other error will result in a failure for this test.

Finally, we come to easily the most advanced game on the SNES: Star Ocean. Released only in Japan, this game combined tried to fit 48Mbits of data into the largest available SNES cartridge of 32Mbits. This was only made possible through the use of the S-DD1 enhancement chip, which handled on-the-fly decompression of game assets. This chip was so difficult to emulate that, up until very recently, additional graphics packs had to be included with game rips for proper emulation. It used software drivers to overcome the 64Kbit limit of the onboard SPC700 sound chip, swapping bits in and out of the chip to greatly increase the level of audio quality. This is never more evident than in the fact that it was able to support full surround sound in those 64Kbits. In addition, it did its Mode 7 graphics processing using software tricks and not the onboard graphics. This allowed a much greater range of effects than the chip was limited to.

As a result of all of these hardware and software tweaks, this is easily the most difficult game to emulate on our list. We will be extensively checking for errors in graphics processing, such as not fully decompressed graphics, to make sure this game operates properly. In addition to testing all capabilities of the chip, we will be applying the unofficial English translation from DeJap to the ROM. This will ensure that our system can play “unofficial” games in addition to 1:1 rips.

6.2.1.6 EmulationStation GUI

To make the EmulationStation experience enjoyable, we need to make sure that logos and such are clearly visible. To that end, we will be testing visibility and usability.

One of the first things we will need to test is overscan or underscan. This will be a very real issue for the device, as it is using a composite output. As a result, it will be necessary to see if any graphics or text are being cropped and to adjust accordingly.

Additionally, we will need to test readability. EmulationStation outputs at 720p by default, clearly a far larger resolution than our screen supports. Additionally, with a 4:3 aspect ratio on a 3.5” screen, we expect the text to be somewhat small after scaling. We will be checking to make sure that all text, be it for settings or other areas of information, are completely legible and easy to make out for any potential user.

We will, of course, also have to test whether or not the control scheme for our internal controller works properly and fluidly with the GUI. Ideally it should be

seamless, with none of the lag so often associated with pulling someone out of an experience. A lag as little as 0.1 ms is enough to be jarring to a user. As a result, we will be timing the display with internal software timers for testing, ensuring that all transitions take less time between frames than this crucial milestone. Making sure that all button presses are correctly interpreted by the system is also key, as the user should not have to play the game of “guess the button”.

Finally, we will be doing usability testing. We will have an unfamiliar user use the device for the first time and see if there is any portion of the interface they have a hard time either using or figuring out. We will make appropriate changes to the interface pending those results.

6.3 Final Integrated System Tests

These tests will be done as the components of the system are put together. They will ensure that as each component is added the system will continue to work. If a new component is added and the system no longer works we will be able to determine what caused the issue. Afterwards we will test the system as a whole in a variety of ways.

6.3.1 Integration Tests

We will start the overall integration testing by combining the solar charge module and wall charge module by using a relay switch and connecting them to the battery. We will test this by testing the output of the relay switch when the wall charger is not plugged in. If the output voltage is not the same voltage as the output voltage of the solar charge module then the combination is faulty. Otherwise the wall charger will be plugged in. The output of the relay switch will be tested and compared with the output voltage of the wall charge module. If the voltages are not the same then the combination needs to be fixed. Otherwise the combination is working.

Next the power supply module will be connected. We will test this new addition by testing that the output of the power supply is 5 volts when on and 0 volts when off. If this is not the case then the addition of the power supply module caused something to go wrong.

The battery indicator module will be attached next. The newly combined circuit will be tested by checking the battery indicator at various states of charge. If the system still works as intended we will connect the current module to the Raspberry Pi 2. If the Raspberry Pi 2 turns on when the power switch on the power supply is on then this combination is working. If not then there is some issue with the connection between the power supply and the Raspberry Pi 2.

Next we will attach the screen to the RP2. If the screen turns on and displays the RP2 output when the power supply is turned on then the addition of the screen is working properly. Otherwise the connection between the screen and Raspberry Pi 2 is likely faulty. The next thing that will be added is the backlight controller. If the backlight controller is able to control the backlight of the circuit then we can proceed adding components. Otherwise we need to determine what the issue is.

The controller module will be added to the system next. If the system recognizes the controller and it functions properly then we will add the next module. If any issues arise we will troubleshoot why adding the controller module caused an issue.

The next module to integrate is the audio module and speakers. If the audio module and speakers correctly play the sound from the system then we can move on to the last component. Otherwise we need to figure out what is wrong with the connection between the audio module and the RP2.

The last module to get added is the Bluetooth module. If we are able to find and connect to an external Bluetooth device from the system then the Bluetooth is working properly. If not the Bluetooth and Raspberry Pi 2 connection is probably faulty.

6.3.2 Final System Tests

The first way we will test the system is simply by using it to play games. If the console efficiently and effectively plays games and is comfortable to use for an extended period of time then we can move on to the next test. Otherwise we need to figure out why it is not working properly or what we can do to make it more comfortable.

The next test we will do is a charge duration test. We will charge the battery to full. We will then use the console away from sunlight and record how long the battery lasts before the console dies. We will then perform the same test in sunlight to determine how long the solar panels extend our battery life.

The next test we will do is a stress test. While we do not need the system to survive in extreme conditions, it needs to be able to handle a small fall or two. We will drop the system onto a couch from 2 feet above it. We will then attempt to turn the system on and check all the components. If everything still works we will repeat this test a few more times. If not then the structural integrity of the device needs to be revisited.

The last test we will do is a temperature test. We want the device playable in both cold and hot areas, seeing as it is meant to be portable. We will place the device in the fridge for an hour and then attempt to use it. If it works still we will move on to the hot test. If it does not we need to find a way to better insulate the

device. To perform the hot test we will place the device in an oven at a 110 degrees for an hour and then attempt to use it. If it works still the testing of the device is finished. If not then we need to find a way to make the device be able to survive a hot day in Arizona.

7. Administrative Content

7.1 Milestones

Milestones are important for the purpose of sticking to a schedule. The charts below, in Tables 7.1.1, 7.1.2, and 7.1.3 detail the milestones, their expected completion dates, and whether those were met, met late, are in progress, or were not met.

Milestone	Action	Date	Met
Project	Chosen	1/28/2015	Yes
Initial Design	Completed	4/9/2015	Yes
Research	Completed	4/20/2015	Yes
Final Documentation	Completed	4/23/2015	Yes - Late

Figure 7.1.1 - Senior Design 1 Milestones

Milestone	Action	Date	Met
Order Parts	Completed	5/16/2015	In Progress
Controls	Produce Output	5/23/2015	In Progress
PCB	Designed	6/1/2015	In Progress
PCB	Completed	6/8/2015	In Progress
Bluetooth Controller	Completed	6/15/2015	In Progress
Power Subsystem	Completed	6/15/2015	In Progress
Case	Designed	6/5/2015	In Progress
Case	Printed	6/10/2015	In Progress
Testing	Completed	6/30/2015	In Progress
Prototype	Completed	7/8/2015	In Progress
Final Documentation	Completed	7/13/2015	In Progress

Figure 7.1.2 - Senior Design 2 Controller Milestones

Milestone	Action	Date	Met
Order Parts	Completed	5/16/2015	In Progress
Screen	Displays Input	5/23/2015	In Progress
PCB	Designed	6/1/2015	In Progress
PCB	Completed	6/8/2015	In Progress
Backlight Controller	Controls Backlight	6/2/2015	In Progress
Backlight Controller	Uses Photoresistor	6/15/2015	In Progress
Power Subsystem	Regulates Power	6/5/2015	In Progress
Power Subsystem	Completed	6/15/2015	In Progress
Audio Subsystem	Completed	6/15/2015	In Progress
Bluetooth Controller	Completed	6/15/2015	In Progress
Case	Designed	6/5/2015	In Progress
Case	Printed	6/10/2015	In Progress
Controller Subsystem	Completed	6/15/2015	In Progress
Integration Testing	Completed	6/30/2015	In Progress
Prototype	Completed	7/8/2015	In Progress
Final Documentation	Completed	7/13/2015	In Progress

Figure 7.1.3 - Senior Design 2 Main System Milestones

7.2 Workload Distribution

The workload for this project was divided amongst the group members, keeping in mind individual strengths, requests for sections, and time and effort involved. All responsibilities are detailed in Figure 7.2.1. While most parts are insulated and modular, each group member still played a key role in developing the overall system. Roles were decided on by group consensus and parts were purchased according to the need of each individual member. Each group member worked on the sections they were assigned for the paper, and then administrative sections were discussed and written jointly.

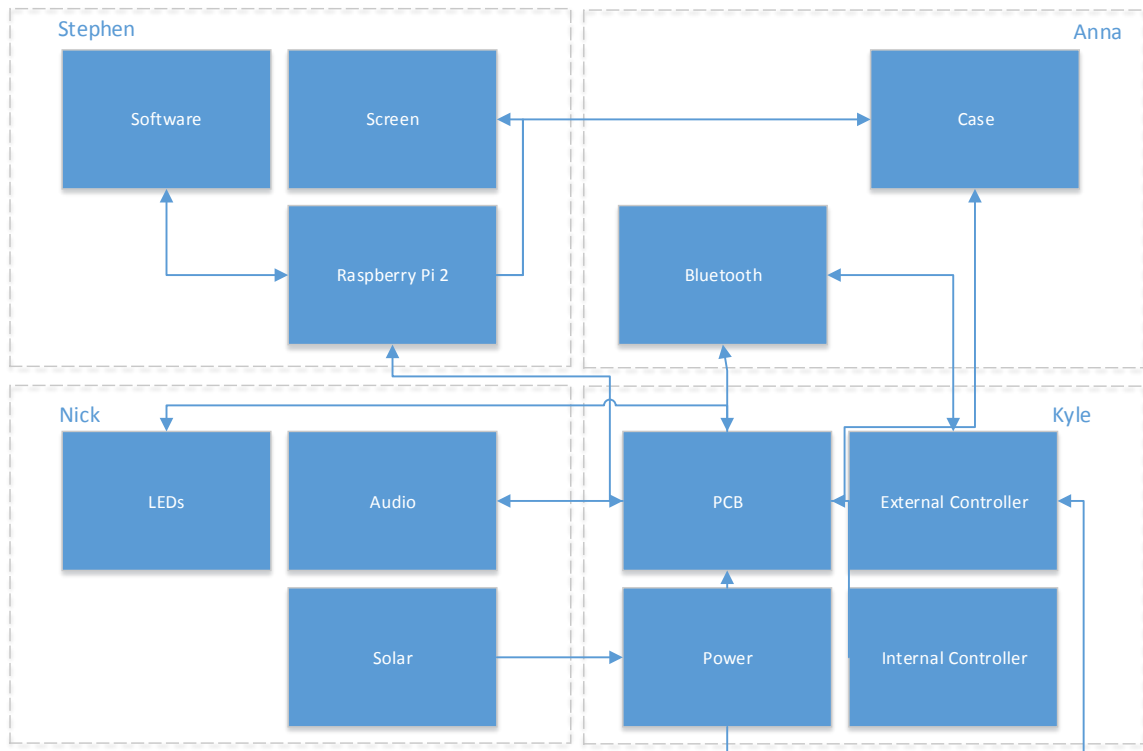


Figure 7.2.1 – Workload Distribution

7.3 Budget and Finances

The FunBox Classic is estimated to cost approximately \$400. This could definitely be cut down if we were to make it again, but initial investments on tools and other materials certainly helped to bring the cost up. We decided not to apply for a sponsorship, as we wanted to keep the device ourselves. We will be splitting all costs equally, as we think that \$100 each is a reasonable price to pay. The budget is laid out in Table 7.3.1.

Item	Cost
Battery	\$15
Bluetooth	\$11
Case	\$20
Extra Parts	\$50
Microcontrollers	\$10
Misc. Components	\$20
PCBs	\$75
Raspberry Pi 2	\$35
Screen	\$20
Speakers	\$10
Tools	\$150
Total	\$416

Table 7.3.1 – Project Budget

8. Conclusion

When the documentation was, at last, completed, the group felt it had a much clearer understanding of what needed to be done to reach the final, functional prototype of the FunBox Classic. Each group member became intimately familiar with their individual component research areas while still not losing sight of the overall project definition. While each member was qualified for the tasks ahead, designing schematics and diagrams was a new endeavor for most, and required the utmost care and attention to succeed. Each step of the design process, each decision made was carefully and completely documented and laid out in this paper. This added accountability and justification for every step, ensuring that the right choices had been made.

What started as an overly ambitious and underestimated, in terms of work, project turned into the FunBox Classic as it stands today. Gone were the lofty goals of a custom operating system and a cartridge slot, replaced with the more reasonable Linux and MicroSD slot. What started as vague ideas and frantically talked about plans began to take shape into individual components, overarching modularity, and the central system that would tie them all together. When all was said and done for, we had our project: A Raspberry Pi 2 connected to a custom support PCB with Bluetooth, custom power regulation, audio splitters, and a controller, all outputting to a composite screen and contained within a custom case. Detailed and exhaustive testing procedures exist to ensure the proper and complete operation of the FunBox Classic, no matter the situation.

This Senior Design documentation pushed the group members into things that had not been taught before. Skills like soldering, PCB design, and proper and extensive schematic design. These were things that either weren't taught or were glossed over in the classroom, which were important to learn due to their essential natures in the industry at large. We learned time-management, although not without a great deal of reticence, practiced and improved technical writing, and made our best attempts at working on communication and group dynamics. The biggest thing learned, however, was how to take a vague concept, tweak it, build on it, and improve it until it stopped being just an idea, turning into a fully fleshed out product design. Throughout the process, group members became intimately familiar with things such as power regulation, Bluetooth, solar power, and how older devices compensated for low system specifications. This project spanned several classes worth of ideas, from the simplest electrical networks all the way up to operational amplifiers, which can be used for almost an astounding number of different things. We are pleased that we have completed our project design document and are ready to get to work building and testing our project, leading up to the construction and presentation of our final, functional prototype.

Appendices

Appendix A – Copyright Permissions



Adafruit Industries <support@adafruit.com>

to me ▾

Totally ok!

On Wed, Apr 15, 2015 at 7:37 PM, Stephen Caskey <stevecask@gmail.com> wrote:

contactname : Stephen Caskey
email address : stevecask@gmail.com
contact us 2 section : press
useragent string : Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.2272.118 Safari/537.36
message text : To Whom It May Concern,

I'm not sure if this is specifically a "Press/media inquiry" but I could not find a more suitable category. I am planning to use the Raspberry Pi 2 along with your HX8357-D Screen for a senior design project. I came across your comparisons for the Raspberry Pi 2 and was wondering I would be able to use the images and data in the articles here:

<https://learn.adafruit.com/embedded-linux-board-comparison/performance>

<https://learn.adafruit.com/introducing-the-raspberry-pi-2-model-b?view=all>

as well as diagrams and data within your datasheets in my senior design paper, with credit properly given of course.

Please let me know if this would be permissible!

Sincerely,

Stephen Caskey
Client IP: 184.88.53.111

Press/Media Inquire

2 messages

Nick Johnson <nick.j8809@gmail.com>

Tue, Apr 28, 2015 at 7:44 PM

To: info-en-c@wikimedia.org

To whom it may concern,

I'm not sure if this is specifically a "Press/media inquiry" but I could not find a more suitable category. I am conducting research for a Senior Design project here at the University of Central Florida. Our report researches audio jack technology, and I was wondering if I could use the jack schematic from your page

[http://en.wikipedia.org/wiki/Phone_connector_\(audio\)](http://en.wikipedia.org/wiki/Phone_connector_(audio))

All credit from the diagram would be credited to Wikipedia in the report.

Please let me know if this would be permissible!

Sincerely,
Nick Johnson

EN-Copyvio <info-en-c@wikipedia.org>

Tue, Apr 28, 2015 at 8:12 PM

To: nick.j8809@gmail.com

Dear Nick Johnson,

In principle, text on Wikipedia is available under CC-BY-SA license, and may be used free of charge for any purpose. Reading more about the license should help explain it in simpler terms: <<https://creativecommons.org/licenses/by-sa/3.0/>>.

Some embedded images and media are under specific licenses, which can be seen upon clicking on the desired image or file. Most images are available under free licenses such as CC-BY-SA, but some copyrighted content (such as book covers) falls under the "fair use" clause. For more information, see the page <<https://en.Wikipedia.org/wiki/Wikipedia:Copyrights>>.

A specific permission for reusing Wikipedia's freely licensed content is not necessary, as long as the re-user observes the license conditions. For most cases, this means:

* An attribution is required, which can simply be a link to the history page of an article or image <https://en.Wikipedia.org/wiki/Help:Tracking_changes#Page_history>. For images, mentioning the creator is a good idea. E.g. for this image <[https://en.wikipedia.org/wiki/Phone_connector_\(audio\)#/media/File:Phone_jack_symbols.png](https://en.wikipedia.org/wiki/Phone_connector_(audio)#/media/File:Phone_jack_symbols.png)>, the creator is "Omegatron" (as can be seen after clicking "View author information"). So, you can mention something like "Image credit: Omegatron, Wikipedia".

* If you modify the content, you must re-release it under a similar free license, which allows others to use the new content freely <<https://en.wikipedia.org/wiki/Share-alike>> ('SA' or 'ShareAlike').

For more information please see: <https://en.Wikipedia.org/wiki/Wikipedia:Copyrights#Reusers.27_rights_and_obligations> or <<https://commons.wikimedia.org/wiki/Commons:Reuse>>.

If you have any questions, you can ask them at the Wikipedia Help Desk: https://en.wikipedia.org/wiki/Wikipedia:Help_desk

Please note: Neither the Wikimedia Foundation, nor the authors of articles on Wikimedia sites, nor the volunteers answering mail to this address provide legal advice. It is your responsibility, if you intend to reuse content from Wikimedia sites, to determine how the licenses of the content that we host apply to your intended uses.

Yours sincerely,
Utkarsh Atmaram

Press/Media Inquire

Nick Johnson <nick.j8809@gmail.com>
To: bourns.marcom@bourns.com

Tue, Apr 28, 2015 at 7:38 PM

To whom it may concern,

I'm not sure if this is specifically a "Press/media inquiry" but I could not find a more suitable category. I am planning to use the Bourns 3352T-103LF-ND potentiometer for a Senior Design project here at the University of Central Florida. I was wondering if I could use dimension diagrams from the 3352T-103LF-ND datasheet here for our report:

<http://www.bourns.com/data/global/PDFs/3352.pdf>
All credit from the diagrams will be credited to Bourns.

Please let me know if this would be permissible!

Sincerely,
Nick Johnson

Press/Media Inquire

Nick Johnson <nick.j8809@gmail.com>
To: sysdev@microsoft.com

Tue, Apr 28, 2015 at 7:34 PM

To whom it may concern,

I'm not sure if this is specifically a "Press/media inquiry" but I could not find a more suitable category. I am conducting research for a Senior Design project here at the University of Central Florida. Our project involves studies in potentiometers, and I was wondering if we could use the DAC potentiometer response diagrams from your company website here:

[https://msdn.microsoft.com/en-us/library/windows/desktop/dd370798\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/dd370798(v=vs.85).aspx)

All credit from diagrams used will be credited to Windows in the report.

Please let me know if this would be permissible!

Sincerely,
Nick Johnson

Press/Media Inquire

Nick Johnson <nick.j8809@gmail.com>
To: sales@maximintegrated.com

Tue, Apr 28, 2015 at 7:28 PM

To whom it may concern,

I'm not sure if this is specifically a "Press/media inquiry" but I could not find a more suitable category. I am researching for a Senior Design Project here at the University of Central Florida. Our project includes research in potentiometers, and I was wondering if I could use the linear and logarithmic response diagrams from your company website here:

<http://www.maximintegrated.com/en/app-notes/index.mvp/id/838>

All credit from the site used will be credited to Maxim Integrated in the report.

Please let me know if this would be permissible!

Sincerely,
Nick Johnson

Press/Media Inquires

Nick Johnson <nick.j8809@gmail.com>
To: stefan.schmidt@osram.com

Tue, Apr 28, 2015 at 1:56 PM

To whom it may concern,

I'm not sure if this is specifically a "Press/media inquiry" but I could not find a more suitable category. I am planning to use the LG R971 LED in a Senior Design project here at the University of Central Florida. I was wondering if I would be able to use diagrams and charts from the LG R971 datasheet:

http://www.osram-os.com/Graphics/XPic9/00078860_0.pdf

With all credit to the used material going to Osram Opto Semiconductors in the report.

Please let me know if this would be permissible!

Sincerely,
Nick Johnson

Press/Media Inquires

Nick Johnson <nick.j8809@gmail.com>
To: matris@matrisled.com

Tue, Apr 28, 2015 at 1:52 PM

To whom it may concern,

I'm not sure if this is specifically a "Press/media inquiry" but I could not find a more suitable category. I am planning a research report for a Senior Design project here at the University of Central Florida. A significant part of the project investigates LED research. I was wondering if I could use the LED Viewing Angle diagram found on your company's page:

http://www.matrisled.com/led_screen_viewing_angle.htm

All credit to the diagram will be given to Matrisled in the report.

Please let me know if this would be permissible!

Sincerely,
Nick Johnson

Press/Media Inquires

Nick Johnson <nick.j8809@gmail.com>
To: press.relations@avagotech.com

Tue, Apr 28, 2015 at 1:33 PM

To Whom It May Concern,

I'm not sure if this is specifically a "Press/media inquiry" but I could not find a more suitable category. I am planning to use the Avago HSMF-A341-xxxxx tri-color led for a Senior Design project here at the University of Central Florida. I am wondering if I would be able to use the supplied diagrams from the HSMF-A341-xxxxx datasheet in our report. All credit to Avago Technologies will be given within the report.

Please let me know if this would be permissible!

Sincerely,
Nick Johnson

Press/Media Inquires

Nick Johnson <nick.j8809@gmail.com>
To: sales@mec-corp.com

Tue, Apr 28, 2015 at 1:23 PM

To Whom It May Concern,

I'm not sure if this is specifically a "Press/media inquiry" but I could not find a more suitable category. I am planning to use the IXYS SLMD121H87 solar cell for a Senior Design Project here at the University of Central Florida. I'm wondering if I would be able to use your images and diagrams from the SLMD121H87 data sheet for the report:
<http://ixapps.ixys.com/DataSheet/20110107-SLMD121H08-DATA-SHEET.pdf>
All credit will be given to the IXYS Corporation within our report as well.

Please let me know if this would be permissible!
Sincerely,
Nick Johnson