SMART TABLE

EEL 4915: Senior Design II

Spring 2017

Christopher Rodrigue
Department of Electrical Engineering
University of Central Florida
rodrigue@knights.ucf.edu

Ryan Mulvaney
Department of Electrical Engineering
University of Central Florida
rmulvaney@knights.ucf.edu

Jonathan Lundstrom
Department of Computer Engineering
University of Central Florida
jlundstrom@knights.ucf.edu

Phillip Murphy
Department of Computer Engineering
University of Central Florida
phillipm91@knights.ucf.edu

Contents

List of Figures

List of Tables

1 Executive Summary

The purpose of this project is to design a table that will have a moderately sized matrix of individual cells on its top that would act as pixels. Each cell would consist of an RGB LED to produce any given color value, which would likely be achieved through pulse-width modulation. Each cell would also have an LED driver that would act as a slave to a main microcontroller on another board. Located in the specifications below is a detailed list of functionality to be included in the project. Timekeeping, environmental sensing, weather information from the internet, and playing various games such as "Snake" are expected features.

This device will not be designed with inherent market value in mind. The device will have relatively a cheap production cost, however, the primary motivation for designing the product is simply as an academic exercise in an effort to improve the group members' understanding of integration across various subsystems. This is a proof-of-concept design that has the potential to extend into the market in future revisions if a practical concept is prepared.

One form of interaction will be through an external application that will be developed to stream data to the table and provide an interactive interface for the user, and possibly physical tactile feedback in the form of buttons embedded on the table itself. This will provide a higher entertainment value for the end user, and as such, is something that is not directly quantifiable.

The device will be supplied mains electricity by plugging directly into the wall. This voltage will be transformed, stepped down, rectified, and regulated to acceptable values for the processing units and RGB light generating units. The physical space of the power supply on the board will be at a relative minimum with heat sinking, if required, taking up an appreciable fraction of space.

The total size of the table will be relatively large, based solely on the scope of the display. For any resolution that we require, we will need a decent amount of physical space to display the information. There will be a minimum of 256 individual pixelated cells on the device. Each cell will include an RGB LED and controlled by an LED driver in relative proximity. The dimensions of the pixelated matrix should be at minimum 8 by 16. Overall, the device will have a robust feature set.

2 Product Description

A smart table can be a new way for people to receive up-to-date information that they have become accustomed to in an ever-changing world. Instead of turning on the television, reading the newspaper, or tapping on a smartphone, an individual can conveniently look at the surface of their table.

This section contains:

- A discussion of the motivations and influences of the members responsible for this project
- The goals and objectives anticipated during the initial design process of the device
- A detailed list and explanation of the requirements and specifications for the device being developed

2.1 Motivation

Tables are relatively unappreciated and underwhelming pieces of furniture, but they are of critical importance to modern humans. Innovation has occurred on tabletops and similar surfaces for thousands of years, from the great early discoveries in math and science, up until this present moment. In fact, this document is likely being read with the assistance of a table, or its close relative, the desk.

The smart table is intended to be a new step in the direction of IoT–the Internet of Things. While many home devices have started to enter the realm of IoT, such as thermostats, light bulbs, washing machines, and other appliances, furniture is an unpopular application for IoT. The smart table is intended to take a conventionally static piece of furniture–the table–and transform it into a dynamic device that can provide much more utility to a user, beyond merely supporting physical objects on its surface. With a couple hundred simple electronic components, such as RGB LEDs, an ordinary table can come to life with information. At that point, it is no longer an object with a solitary function. It becomes a much more exciting and interactive device with boundless applications waiting to be programmed by the user.

As a display device, the smart table can show the user useful information that can marginally improve their quality of life through convenience as well as entertainment. Although using discrete LEDs rather than a flat-panel display creates a lower resolution, the smart table can be seen as appealing to both young and old crowds who have a passion for retro electronics. Information such as the time, date, and daily weather could be viewed in a single glance. On the other hand, RGB LEDs can make colors

truly pop; the smart table could display vivid and entrancing patterns of color or pixel art.

The main motivation for designing and building a smart table is to make a statement: that the applications of electronics can not and should not be limited. Although electronics remain critical in modern transportation, medicine, construction, productivity, communication, and more, we would like to see the application of electronics expand to objects that have been overlooked, such as furniture and clothing. We hope that we can live to see the day where every fiber and fabric of our world is connected and empowered by electronics devices.

2.2 Goals and Objectives

The main goals for this project is to design a build an interesting and unique conversation piece that is both aesthetically pleasing while also being technologically complex. As such, both the technical and non-technical parts of the project shall be integrated closely to reinforce one-another in a synergistic way.

Hardware The hardware shall consist of the main logic board, power system, LED array, and the physical interface/mounting system required for the project. All the parts shall be integrated into a singular package for potentially easier deployment.

Software The software shall consist of concise modules for control and data processing of the various features included in the final implementation. The user will have partial communication with the unit through the included wireless capabilities of the system.

Control The smart table will implement a microcontroller to act as the central processing unit to interface with the communication system and physical outputs. As such, the microcontroller will also be used for bulk data processing when required in the control system.

Communications The smart table will implement a wireless communication system to handle any data input required by the user and for implementing any external calls required for the software interface.

Power Supply The power system for this project will be a custom implementation to adequately handle the power requirements as demanded by the system. As the load is partially dynamic, the system will be designed with that idea in mind.

2.3 Requirements and Specifications

- Physical Table
  - Should be no larger than 4 ft long by 8 ft wide by 4 ft tall

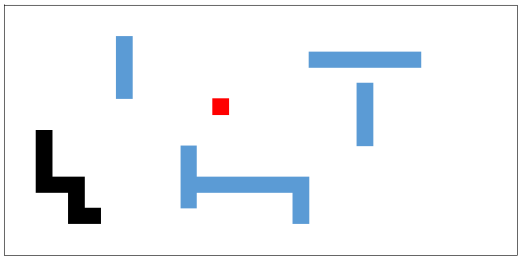
  - Should be structurally sound, holding at least 25 lbs of total weight
  - Constructed primarily out of wood, with a plexiglass covering the cells
- Single microcontroller to run the device
- A minimum 256 individual pixels will be located on the display
- Several auxiliary devices/functions will be included
  - Phone charging port(s)
  - Temperature sensing
  - Bluetooth connectivity
- Software functions
  - Various animations for startup, loading, and other transitions
  - Timekeeping
    - Clock, timers, and alarms
  - Reading from ambient pressure/temperature sensors
  - Play games
  - Fetch and display weather data from external device (see Figure 1)
  - Minimum 15 Hz refresh rate
- 24 VAC transform input to board
- Production cost under $1,000
  - Majority of the price is board fabrication (surface mount devices are relatively cheap)

2.3.1 Examples of Runtime Output

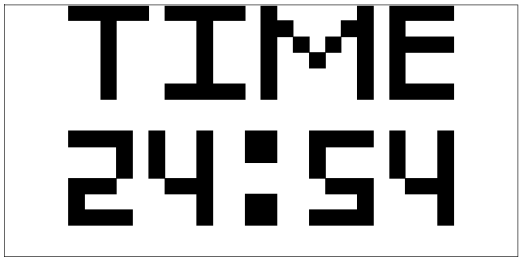
Figure 1 shows examples of potential information displayed while the table is executing code and pushing information to the LED matrix. Figure 1b is an example of the device outputting time to the display. Figure 1a is an example of an instance of Snake being played on the display. Figure 1c is an example of weather reading by the device–the current weather will fetch from an internet service and analyzed. A pre-programmed animation will be shown on the smart table, depending on the weather. In this case, the weather outside is partially cloudy.

2.4 House of Quality

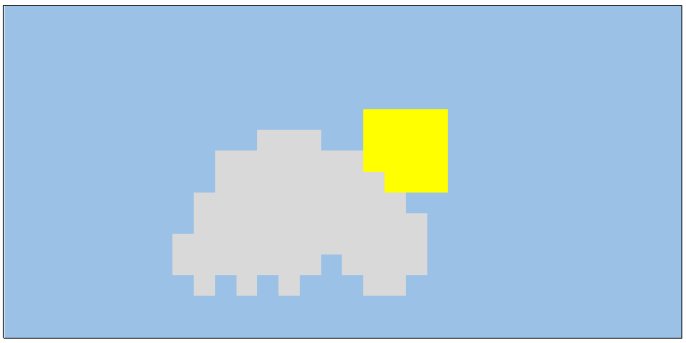
↑ = Positive correlation
↑↑ = Strong positive correlation
↓ = Negative correlation
↓↓ = Strong negative correlation
+ = Positive polarity , positive effect on project quality
- = Negative Polarity , negative effect on project quality



(a) Snake Game



(b) Display Time



(c) Display Weather

Figure 1: Operational Vision

			Engineering					
			Structural Integrity	Dimensions	Power Mode	Heat Generation	Refresh Rate	Cost
			(+)	(-)	(+)	(-)	(+)	(-)
Market	Interactivity	(+)	↑				↑↑	↓
	Functionality	(+)	↑			↓	↑	↓
	Style	(+)		↓			↑↑	↓
	Ease of Use	(+)		↓				↓
	Resolution	(+)	↑	↓↓	↓	↓↓	↓	↓↓
	Power Efficiency	(+)			↑↑	↑	↓	↓
	Cost	(-)	↓	↓↓	↓↓	↓	↓↓	↓↓
			25 lbs	8ft x 4ft			15 Hz min	$1000 max

Table 1: House of Quality

As shown in Table 1, the market parameters are qualitative in nature and are focused on the elements of the project that usually influence an individual to purchase a device. Interactivity is a measure of how engaging the device is, and style is a measure of the furniture's appearance. Ease of use is an important market parameter because it defines the overall accessibility of the device from a non-technical standpoint. Using the device should feel intuitive and simple while maintaining functionality, which is yet another parameter taken into consideration when purchasing a new device. For

the more informed consumer, more technical market parameters are considered when purchasing a new display. There is a trend towards displays with high power efficiency, high resolution, and low cost.

The engineering parameters of the project are quantitative in nature. The structural integrity of the table is an obvious inclusion to this list, and addresses how much weight the table can support since that is the principle function of the device. Relatedly, the dimension of the table is another engineering attribute that has a significant effect on the total cost of the project. Electrical parameters relevant to any display technology are refresh rates, heat generation, and power modes. Ideally, the device will have a comfortable refresh rate, generate an insubstantial amount of heat, and include power modes that will minimize overall power usage and the cost of maintaining the device.

3 Design Constraints and Standards

Every engineering endeavor has certain limitations, collectively referred to as constraints. They may be financial, technological, or legal constraints, but nonetheless they exist and must be dealt with accordingly. Successful designers observe these constraints and work according to various standards to create a successful product.

The following section contains:

- A list of design constraints pertaining to the development of the smart table.
- Various standards that apply to the smart table.

3.1 Standards and Other Safety Concerns

In this section, various electrical and software standards are examined. Also found here is a discussion to the relevance to each standard to our application. Along with various industry standards, also found in this section is a discussion on general safety concerns and procedures to ensure the safety of the group mates at all sections of design and manufacturing.

3.1.1 Soldering Standards

The National Aeronautics and Space Administration (NASA) has a soldering standard, according to a document titled "NASA TECHNICAL STANDARD: SOLDERED ELECTRICAL CONNECTIONS". In this document, they discuss several very important key points of proper soldering techniques. Several diagrams are included, which will be reproduced here in order to describe exactly how solder should look once distributed on the pad. NASA's document includes several sections for through-hole technology, as well as various other surface mount soldering techniques such as round lead termination and other package styles. In this application, we will be dealing with either gull wing packages or simple chip components such as 803 package resistors and capacitors. In order to understand soldering standards, it is important to understand the specific language used to describe several facets of soldering. NASA defines a "fillet" as a smooth concave buildup of material between two surfaces, for example, a fillet of solder between a conductor and a solder pad or terminal. With this definition, please refer to the following figure.

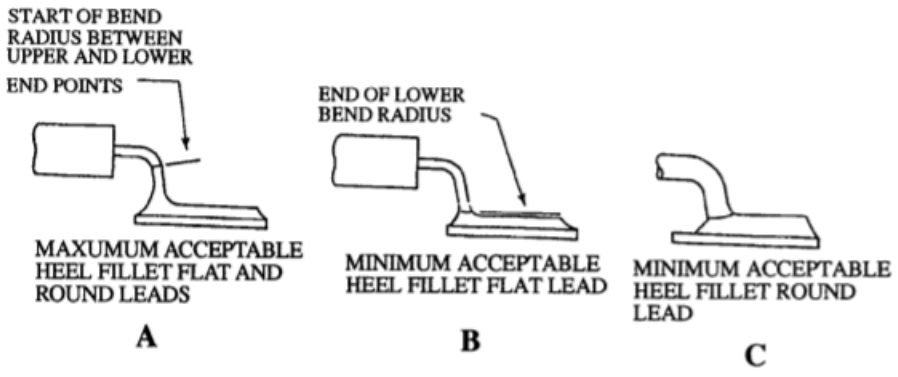


Figure 2: Proper soldering method [Tec] *Permission submitted to reproduce*.

In the figure, Example A is defined as the maximum acceptable heel fillet flat and round leads. If any more solder is added, the heel fillet (the fillet located behind the bend of the gull wing pin of the package) would be destroyed. The concave nature of the fillet must be retained. Example B is a similar situation to Example A, but there is almost too little solder. Any less solder would compromise the integrity of the fillet. Example C is similar to Example B, but with a different style of gull wing that are round rather than flat.

Along with information about what solder should look like on the board (with regards to a gull wing package), NASA provides several pointers on proper hand soldering of printed circuit boards. They specify that pressurized air should never be used to cool solder after working on a solder joint. The solder should always be left to cool under room temperature conditions only. This is because cooling the solder too rapidly could lead to one of several problems, including but not limited to, fragile solder joints or bad fillet connections. In regards to solder coverage, the molten solder should be flowed around the conductor and over the pad; please give reference to the figure above to decide how much solder should be used in order to properly affix the package to the board. Importantly, there should be no motion between conductors and the pad during soldering or while solder is cooling. This rule is in place for the same reason as above: the fillets could be destroyed and the joints themselves could be compromised if not allowed to set properly.

After the action of soldering to the printed circuit board has been completed, it is then important to inspect the board to ensure that some measure of quality has been maintained. Visual inspection of all solder connections is important to ensure that the board will function as expected. Where visual inspection is not possible, other nondestructive means such as x-ray or fiberscope optics shall be used. This is in cases where the pads are located under parts such as microcontrollers or communication modules. We will not be using any package where typical visual inspection would not be sufficient. Therefore, no complex methods of inspection will be necessary, but for more

robust applications it is important to understand that such options for inspection exist. Once the solder paste has been set and completely cooled, it is important to perform a continuity test with a multimeter. The multimeter probes should be probed on the pins of the devices, to ensure that each of the pins are connected through the solder to the board, then through the board back through the solder to the pins of the other connected devices.

For another point of reference, "Circuit Technology Center" is an online resource detailing the proper procedure for soldering components to a board. As previously mentioned, the gull wing package of hardware is the most common type seen on finished products. As a result, an analysis of the gull wing from this site will be considered. Please refer to the figure and table below.

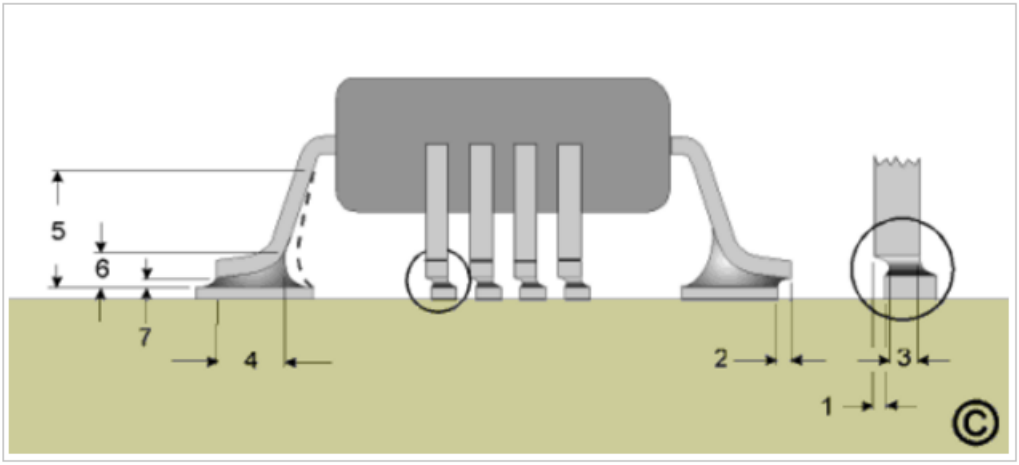


Figure 3: Proper soldering method [Tec] *Permission submitted to reproduce*.

This is a more in-depth analysis of the particulars the solder application, giving particular exammples of both proper and improper solder paste distribution. Circuit Technology Center also has a brief description of soldering techniques, similar to NASA. An important point that Circuit Technology Center makes is the desire to avoid resoldering. Care should be taken to avoid the need for resoldering, as the quality of the resoldered connection may be worse than the original connection. The addition of flux will almost always be necessary when resoldering a connection on the board, as simple reheating typically does not produce the same quality connection as would be expected from a fresh solder connection. Adding a small amount of flux before touching the soldering iron to the lead solder allows the solder to settle properly and gives a better chance that the solder will flow properly and that proper fillets will be formed, as labeled in the above figures.

3.1.2 Lead Solder Safety

We intend on using pb63sn37 solder, which means that the component parts of the actual solder are made up of 63 percent lead and 37 percent tin. The danger of lead

Feature	Dim	Class 1	Class 2	Class 3
Maximum Lead Side Overhang	1	No more than 50% of the lead width or 0.50 mm (.020") whichever is less.	No more than 50% of the lead width or 0.50 mm (.020") whichever is less.	No more than 25% of the lead width or 0.50 mm (.020") whichever is less.
Maximum Toe End Overhang	2	Toe overhang is acceptable as long as it does not violate electrical clearance.	Toe overhang is acceptable as long as it does not violate electrical clearance.	Toe overhang is acceptable as long as it does not violate electrical clearance.
Minimum End Joint Width	3	50% of the lead width.	50% of the lead width.	75% of the lead width.
Minimum Side Joint Length	4	Equal to the lead width or 0.50 mm (.020") whichever is less.	Equal to the lead width or 0.50 mm (.020") whichever is less.	Equal to the lead width or 0.50 mm (.020") whichever is less.
Maximum Heel Fillet Height	5	No maximum, but solder must not touch the component package body.	No maximum, but solder must not touch the component package body.	No maximum, but solder must not touch the component package body.
Minimum Heel Fillet Height	6	Evidence of proper wetting.	Equal to the solder thickness plus 50% of lead thickness.	Equal to the solder thickness plus 100% of lead thickness.
Minimum Heel Fillet Thickness	7	Evidence of proper wetting.	Evidence of proper wetting.	Evidence of proper wetting.

Figure 4: Description of above figure [Tec] *Permission submitted to reproduce*.

and lead byproducts are well known in modern science, and as a result careful concern must be taken when working with lead solder. Lead (Pb) is a known neurotoxin, and can pose several significant chronic health effects. Potential issues include but are not limited to reproductive problems, digestive problems, memory and concentration problems, muscle and joint pain, and more. However, as long as solder is handled properly, there is minimal hazard to any person working with the solder.

First of all, it must be noted that inherent in soldering is the danger of burns. For our application and with the particular solder we intend on using, the setting of the soldering iron will be around 750 degrees Fahrenheit. This, of course, can cause fairly serious burns if not respected. As a result, any time a team mate is handling the soldering iron it is important to be very careful, and only use the soldering iron for a short time before returning the iron to the cradle. While a majority of the production will be done using a pick and place machine and an oven, there are still sections of the build that will require hand-soldering parts. This includes prototyping, where parts will be attached to a breadboard through soldered wired onto the pins. Also, in the case that the finished

product has an issue, hand soldering will be implemented to resolve it.

As mentioned in the introduction to this section, the potential health issues of lead have been well documented in the scientific community, and as such there is a decision to make in terms of what type of solder we intend on using for this project. There are several important considerations to be analyzed before making a final decision on solder use. First and foremost, the health aspect should be the main concern in any development procedure. However, in most applications, it is fairly difficult for typical lead solder to pose any threat to humans. It takes a very large amount of ingested lead solder to pose a threat. Another consideration to be made when choosing a particular solder is RoHS compliance. RoHS is discussed in detail later, however, a brief overview will be discussed here.

Basically, a device that is in compliance with RoHS has restricted certain elements and compounds in order to meet foreign export necessities and other health standards. In our case, RoHS is not of major concern because we do not intend on exporting this product internationally at any point, nor do we plan on selling the product. This could eventually change, and at that point the type of solder should be re-examined. For our immediate needs, however, the 63 percent lead-based solder is acceptable. This solder is much more friendly to use in all applications. It is typically cheaper and more forgiving of temperature fluctuations in the oven. It is also easier to solder in general; experience has shown that non-lead based solders are very picky to temperature changes and require much finer adjustments to flow properly to the pad. Also, they appear almost grainy when soldering and do not look quite as good when completed. In general, if there is no other restraint, it is typically a good idea to use lead-based solder.

In terms of lead solder safety, Carrnegie Mellon University has a document on general safety precautions when working with lead. Of primary concern for CMU is the solder fumes produced by the melting of solder. When the product is in the oven, there is no concern for these fumes as there is an internal fan and ventilation system. However, when hand-soldering and repairing parts, the fumes can become an issue. CMU specifies that working in a well ventilated area, or working under a local exhaust ventilation while making every effort to avoid inhaling the fumes should be exercised at all times while working with lead-based solder. It is also important to wear proper clothing while soldering of any time, which includes typical lab attire such as closed-toed shoes and eye protection.

It is important to understand that while soldering with lead, the fumes are typically more likely produced by flux containing rosin. Flux an alcohol base that allows the solder to flow more evenly and cleanly to any pin or pad on the board. As mentioned previously, it is very important to work in a well ventilated area when soldering. The UCF Texas Instruments Innovation Lab is a good example of a proper soldering environment, where the location is relatively open with good ventilation.

Clearly, it is very important to not touch the tip of the soldering iron while in operation,

as the iron is usually very hot (>750 F for our application). Also, with regards to the iron, ensure that the iron is unplugged and off when leaving the station. Leaving a soldering iron on and heated while unattended is a major fire concern, as the tip could easily fall off the cradle and onto any flammable surface, quickly heating it and causing a fire.

To protect against as many avenues of lead exposure as possible, there are several preventative steps to take. Eye protection is recommended for soldering applications, as solder has a tendency to "spit," or disperse smaller pellets of molten solder into the air. Keeping these out of the users eye is of utmost importance. After completing any work with solder, one should always immediately wash their hands in order to rid them of any lead deposits left from working with the solder. Typical skin-on-lead contact is of no primary health concern, however, if lead deposits exist on the hands, it is possible to accidentally ingest it through other avenues. For example, lead can deposited onto food if it is not removed from the hands after soldering.

When soldering, it is of utmost importance to ensure that the circuit being worked on is not live. There are several reasons for this, ranging from the safety of the parts on the board to the protection against electrocution to the user. If using a live circuit, if there is a fine pitch device that you are trying to solder or two pads otherwise close to each other, it is entirely possible to short the two pads together through an accidental solder bridge. This happens rarely when using a stencil and solder paste, however, when performed by hand-soldering methods, it is more likely to occur. If two pins are accidentally shorted through solder bridge or touching the two pads or pins together with the solder tip (which is electrically conductive), any number of problems could occur. Of primary concern, as previously mentioned, is the safety of the user. If working with high voltage applications, the user could become part of the circuit if not properly grounded, and as a result, may suffer an electric shock. Problems also arise with part destruction, where shorting two pins together could destroy individual ICs on the board, or destroy the power supply. Some soldering irons have a ground pin that can be affixed to the circuit or to earth-ground in order to prevent such problems from occurring, but this is not an all-encompassing solution. The easiest way to prevent these issues, and the safest protocol in general, is to ensure that the power has been removed from the circuit before working on it.

Lead soldering waste, like any other lead byproduct, is considered a hazardous material. Therefore, proper disposal methods for such waste must be followed to ensure that the lead is not handled improperly at a landfill, potentially causing issues with groundwater sources. Lead solder must be disposed of in a metal container, and labeled as hazardous waste. Storage of solder must also be properly handled, with a proper label for the dross (a type of container). Approved labels are provided by UCF EHS if desired. Such labels should describe the potential for danger, clearly stating "contains lead" or some variation thereof. Used solder sponges and contaminated rags must be disposed of in the same way as standard solder, as it contains lead deposits. Follow the above procedure to dispose of these components.

Any further concerns regarding the handle, disposal, or other facets of lead-based products,can be directed to the Environmental Health and Safety (EHS) at UCF, phone number (407) 823-6300.

3.1.3 RoHS Compliance

RoHS, pronounced roh-haws, stands for the Restriction of Hazardous Substances Directive. This is a short hand of the directive on the restriction of the use of certain hazardous substances in electrical and electronic equipment, and was adopted in February 2003 by the European Union. Typically, people associate RoHS primarily with a restraint on lead, and that is the case in most electrical engineering applications. However, RoHS documentation details ten substances that are to be limited: lead, mercury, cadmium, hexavalent chromium, polybromated biphenyls, polybromated diphenyl ether, bis(2-ethylhexyl) phthalate, butyl benzyl phthalate, dibutyl phthalate, and diisobutyl phthalate. RoHS is closely associated with the Waste Electrical and Electronic Equipment Directive, otherwise known as WEEE. A complete discussion on WEEE will not be included here, however, it is important to understand that WEEE sets collection, recycling, and recovery standards for various electronic goods. It is intended to remove various toxic components from electronic parts, such as lead. RoHS does not require specific markings in order to specify that the product is RoHS compliant. However, if shipping to various locations abroad, it is required that you follow the RoHS restrictions.

Our main concern out of the list of 10 restricted substances on the RoHS list is clearly lead, should we decide to be RoHS-compliant. Lead is commonly found in solder and solder paste, and is considered widely to be the standard for solder. RoHS specifies that a concentration of <1000 parts per million is to be maintained over the board to ensure compliance. However, because we are not selling overseas, it is not likely that we will be working towards RoHS-compliance. The common alternative for lead solder is a 98 percent tin solder, which is very unruly and produces fewer acceptable solder joints. It is, however, important to mention the standard for future reference, if the product wereever intended to be sent overseas. It would be important to change the manufacturing procedure to ensure RoHS-compliance.

3.1.4 C Language Standards

As a note, there is not an ISO standard C programming/coding style or standard defined. As such, there are two notable general purpose coding styles that can be observed and followed. Notably the AT&T's Indian Hill Labs [var] coding style and NASA's [NAS] C coding style documentation. Both of these can be pulled from for interesting and useful coding standards to help in the implementation of larger programs and to keep the clutter and code explosion down when building such projects.

AT&T's Indian Hill labs C coding style reveals some decent styles, while not as important with modern styles and compilers, are still useful in reducing clutter. As such, files

longer than 1000 lines are actively discouraged due to handling issues arising from the sheer length of such a file. Additionally, the file shall not have columns longer than 79 lines as some editors cannot handle and/or display greater widths properly. The file structure is specified:

- Prologue: A description of what is included in the file and the purpose of each object.
- Header Files: The include and reason for include, commented out.
- Defines + Typedefs: Assuming these apply to the whole, should be included next.
- Globals: This includes global defines and data.
- Functions: Last in the list, should come in some sort of meaningful order.

Additionally useful are the inclusion of definitions for comments. The comments should describe what is happening, how it is being done, what parameters mean, which globals are used and which are modified, and any restrictions or bugs, which is explicit and clear in the definition. This alone improves the readability of code and the transfer of information for another person working on the same code. Proper tabbing and spacing of comments should be followed in this standard, and are essentially defined as the same alignment as what the comments are commenting upon. If commenting upon a line and if residing upon the same line, the comments should be right-justified to the same amount.

Declarations should be also taken care of in a careful manner to reduce possible errors later on and to increase code readability. As such, when defining a pointer, the individual variables themselves should have the * assigned to them instead of the typedef. This improves readability and avoids the issue of only one variable becoming a pointer variable. Unrelated declarations of variables should be on different lines, even if they are the same type. Additionally, the empty initializer should never be used and structure initializations should be fully enclosed by proper parenthesized braces.

Whitespace should be used generously and, as such, there should be at a minimum two blank line between the end of one function and the comments for the next. This vastly improves readability when scanning a source document. Additionally, there are definitions for styling simple and more complex statements, as such simple statements should only have one statement per line excluding those that are closely related. Compound statements, defined as those included in a list of statements enclosed in a brace, should remain consistent to the predefined local standard. The specification of local standard is due to the fact that C's calls and definitions of various features and functions can changed between the hardware and the compiler. Thus, a universal standard is nearly impossible for coding in C. Operators, such as the binary comma operator

and the ? ternary operator, should be avoided in nesting if possible due to the confusion they can cause in ease-of-readability of the source code. Additionally, unary operators shouldn't be divided from their singular operand. Generally, all binary operators (excluding a few) should be separated from their operands by spaces to improve readability.

Naming conventions, though arguable per project, have some predefined styling conventions used in the AT&T Lab C style documentation.

- Names should not have leading or trailing underscores as these should be reserved for system purposes, if private identifiers are needed they should have leading letters for identification.
- #Defines should be in pure CAPS.
- Enum are capitalized or in all CAPS.
- Function, typedef, and variable names, additionally struct, union, and enum tags should be in lower case.
- Macro functions are in CAPS, though some lower case macro names are acceptable if they behave like a function call
- Avoid names that only are differentiable due to the leading case, as these are near indistinguishable
- Additionally avoid name that look like each other, i.e. 1 i I look similar

Additionally not mentioned in the main list, is to try and avoid names that might conflict with various standard library names, thus avoiding issues in which different systems including more library code than expect and also allowing for the easier extension of your program in the future as per normal operating in code design.

Numerical constant definitions should not be hard-coded in-line if possible as this is bad practice for the most part and should be avoided. As such the #Defines feature should be used in this case, thus also allowing the easier modification and portability of the code in future iterations and testing. Macro parameters and operator-precedence problems can occur when parameters are not properly enclosed inside parentheses. This is hard to avoid and thus it is good practice when possible to write macros that evaluate input parameters once, though this is not always possible. Additionally, when using macros, globals should be avoided since local declarations may be hidden, leading to undefined or undesirable behavior.

Conditional compilation is another area of focus to help with machine dependencies, debugging, and setting certain options at compile-time. By use of the ifdef command,

more portable code can be generated with less hassle and simpler debugging. Enum style definitions are also specified and defined; if enums are used, the first enum constant should have a non-zero value or the first constant should indicate an error. Thus, initialized values can be caught in this way, reducing debug time. Also included in the definitions are further readings on portability and machine-specific coding which is of little relevance to this project. The focus was also on AT&T's coding style instead of NASA's due to its higher relevance.

4 Research and Background Information

This section is meant to examine various ways that the product specifications and requirements can be achieved, through a careful comparison of available hardware and software protocols and components.

This section contains:

- A quick look at products that are similar to the smart table in their appearance or function.
- Competitive technologies and protocols which could be implemented for the smart table.
- Considerations and comparisons of various devices that could meet the needs of the subsystems of the smart table.

4.1 Similar Projects

The following section contains expansions in the research of pre-existing and/or similar projects that architectural styles, ideas, or influences could be pulled from regarding the design of the smart table.

4.1.1 RGB LED Coffee Table

This project demonstrates strongly the end product's idea for the smart table. Key points to look at for the smart table are: its visual appearance, how the LEDs will be driven, how the LED data will be generated, and the physical design. The visual output of this project is quite impressive due to the implementation of various algorithms to generate images. The implementation of Perlin noise leads to very well done images that are both visually and technically complex. This technique useful in procedural generation of images and thus this is an interesting usage of it. The LEDs are driven using the TLC5940 LED drivers from Texas Instruments which is a good choice, as they support each 16 channels, have PWM control, and constant current control. The project uses 8 of these ICs, each driving 16 LEDs and then sent data from the main microcontroller to control the table lighting system. The data the microcontroller is sending comes from an outside computer which processes the stream data and sends it to the microcontroller via a serial interface. This allows the table to have access to much higher-end software packages and another processor to offload the computationally expensive fast fourier transform the table is using. The final consideration of the table is its physical design. The dividers built by the team are a solid solution to the issue of light bleeding into other cells, and the addition of making the interior partially reflective helps with the issues of the cells being physically large. The main takeaways from this

project is the physical table design and the algorithms chosen for image generation. [Lab11]

4.1.2 Clock Shaper

This project is an interesting take on the solution to driving WS2811/WS2812 controlled RGB LEDs. The issue the WS2811/WS2812 pose is the communication protocol the chips use for digital control. The protocol can being roughly thought of as PWM to send digital data, in which two-thirds of the time, high represents a one, while one-third of the time represents a zero, and then a longer amount of time low unlatches the shift register it uses to keep track of the input data. The WS2811/12 ICs can be thought of as a shift register mated to an LED constant current drivers mated to a signal-shaping chip that activates once the shift register is full, allowing the ICs to retransmit data and thus be daisy-chained together. What the clock shaper project aims to address is the computational overhead of constantly generating the pseudo PWM signals required to drive the WS2811 chips, which a lower end microcontroller could account for the vast majority of its available cycles. Thus, this project uses dedicated hardware that is commonly available on most microcontrollers to offload the cost of the signal generation from the main processor to hardware that runs separately of the main unit. SPI modules essentially have data shoved onto them, then the hardware itself sends out the data serially without further intervention besides the initial data pushed to it. The issue with this approach is that the data itself is standard high-low formatting. The author of this project uses another set of external hardware to apply signal shaping to the output of the SPI module. The author essentially uses a simple RC circuit with a transistor to generate the required PWM signals with acceptable margins of error in the signal. This results in massively reduced overhead for driving the LED driver ICs, which is a key consideration in using these ICs. [wik14]

4.1.3 Improved STM32 WS2812B Library

The author of this article discusses multiple methods of driving the WS2812B chips which require custom code to communicate with due to their pseudo-PWM signal requirements. The implementation that the author pursues is essentially using timer-driven interrupts to force the direct memory access (DMA) module to trigger fast data transfers to the general purpose input output pins (GPIO). By using this method, the total number of lines able to be driven using this method is 16, each line carrying a maximum of 1024 LEDs, thus resulting in a total of 16,384 RGB LEDs able to be driven by a single microcontroller. Therefore, it is easy to see the possible uses of this method, though the implementation is hardly easy or the most efficient. The issues posed by this method is the jitter on the output caused by using interrupts to drive the code, which falls within safety margins as the interrupts are short enough. The next issue is the probably of overhead, as a read-modify-write command to change individual bits is a costly operation which the author sidesteps as the hardware provides atomic access to bits thus resulting in a single operation to change a bit. This optimization alone is why this method is viable in driving the LEDs chips, as the interrupts and context switching

add overhead on their own. The end result is a method viable for re-purposing in the final Smart Table product. [hub16]

4.2 Microcontroller Considerations

This section goes into detail in the microcontrollers that were evaluated during the selection phase of the project. Multiple controllers were selected and tested due to varying requirements at the time during the initial planning phases which were constantly shifting in response to capabilities of the controllers themselves and/or possible issues that would arise during stretch goal capabilities being added onto the project.

4.2.1 MSP430G2553

Table 2: MSP430G2553 Basic Specifications [T.Ia]

Specification	Value	Summary of Interest
Main Clock Speed	16MHZ	Basic modern speed, average
Primary Storage	512B	Will have to hyper optimize access
Secondary Storage	16KB	Basic amount needed for semi complex code
Communication Module	USCI	Standard communication (SPI) support
Clock System	Watch Crystal, Internal Programmable, External Source	Allows finer control over clock speed matching

The MSP430G2553 microcontroller is part of Texas Instrument's value line series using the MSP430 core. In particular this is their most powerful offering, that at the time of this paper being written, is available and contains the capability of being slotted into the basic launchpad for the value line. As such, a device fits into a main requirement of the group, which is to bring down the total cost of the project. It was the first microcontroller to come under consideration. Also, all group members have have previous experience with TI's MSP430 cores, and as such, it was a massive incentive to continue the project's focus with this line. The main features that come into consideration for this part are: low overall power draw, a simple instruction set, an internal watch crystal, the ability to use external clocks, a max stable clock speed of 16 MHz, dual timer modules each with three capture/compare registers, multiple I/O pins, a USCI module which includes UART, SPI, I2C capabilities, an on-board ADC, and finally, easy programming access with a JTAG interface.

As this is an MSP430 based microcontroller, low power is the main advertisement and draw of these devices, as their overall power draw is exceptionally low with a draw in the milliwatt range when in power saving mode. This requires careful consideration from

the programmer to implement properly, as the device has to be configured properly to go into low power mode and exit this mode properly as the clock sources for certain devices can become unstable and cause undefined behavior. Despite this additional cost to the programmer, the trade-off of exceptional low power performance is something to pursue in embedded device design. To help aid in reducing the load on the programmer, the MSP430 line has excellent documentation and a well-defined instruction set which massively aids in software development for the microcontroller. Additionally, the programmer, if using the TI-provided header files, has easy access to defined macros and functions to help control the microcontroller more effectively without having to resort to writing in pure assembly language. Also provided is sample source code for general usage.

The clock system for the MSP430 line is in general a very robust system. In this line within the overarching family, the clock system allows for multiple sources to be used and/or configured in the aggregate of instances which a clock is used as an input. In this particular device, a main system clock is used to generate the clock signals used throughout the device. The clock speeds of this device are configurable using presets, to set it certain predefined speeds which are proven stable by Texas Instruments. The presets alow the programmer to set the up to an impressive 16 MHz, impressive for such a low power device with its capabilities. To boot this device comes with the ubiquitous 32kHz watch crystal, a long standing standard used across a multitude of devices due to its ease of production, stability, and most importantly the ability to be cleanly divisible by two. Additionally this crystal is the core to the low power mode, if you still need peripherals to run without the main clock running. The most interesting feature of this clock system, in terms of potential, is its inclusion of the ability to use external clocks for clocking its systems. Therefore if a specific speed was needed, a certain precision, or power draw requirement was needed to be met, an external piece of hardware could be used to provide in this instance; this leads to an incredible amount of freedom and customization of the hardware for the needs of the customer and/or company using this device.

The timer modules built into this device are exceptionally useful for a multitude of tasks. They can be used for a variety of tasks such as implementing PWM without a computational costly method of direct processor intervention. The module itself runs off a configurable clock source, then contains multiple registers inside it to track the number of clock cycles counting and triggers events depending on the configuration of the timer module itself. Additionally, each register can trigger an interrupt event in which a temporary halt to the main code is caused, or even a wake up from low power mode. This ability is core to efficient CPU cycle usage which is a hard line issue that must be considered in such low power devices. The modes also offered by the module for triggering are highly customizable and offer interesting methods of control depending on the application. For example, the following code was developed as a proof of concept for using the timer module on an MSP430 device to generate PWM code for controlling a servo, a very useful and realistic device to interface with in a common scenario.

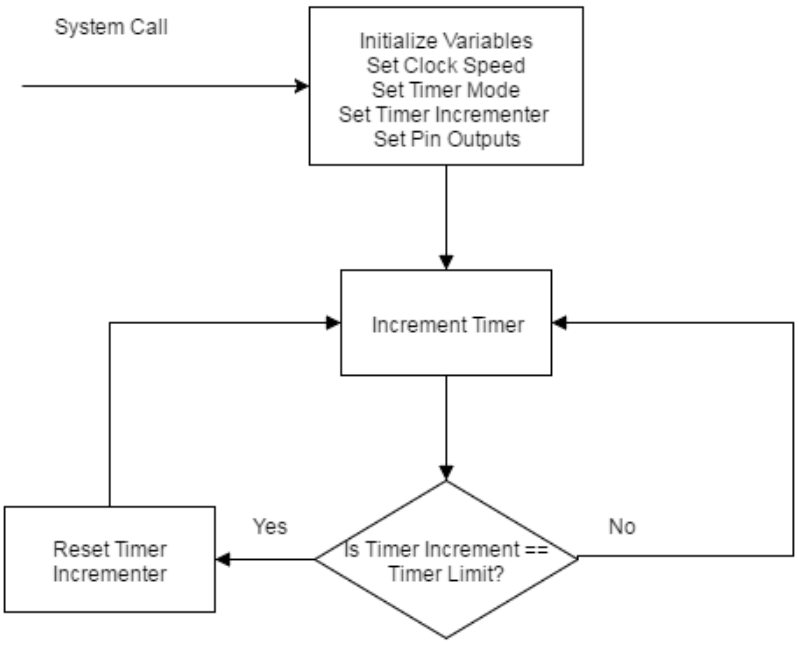


Figure 5: Flowchart of basic PWM control

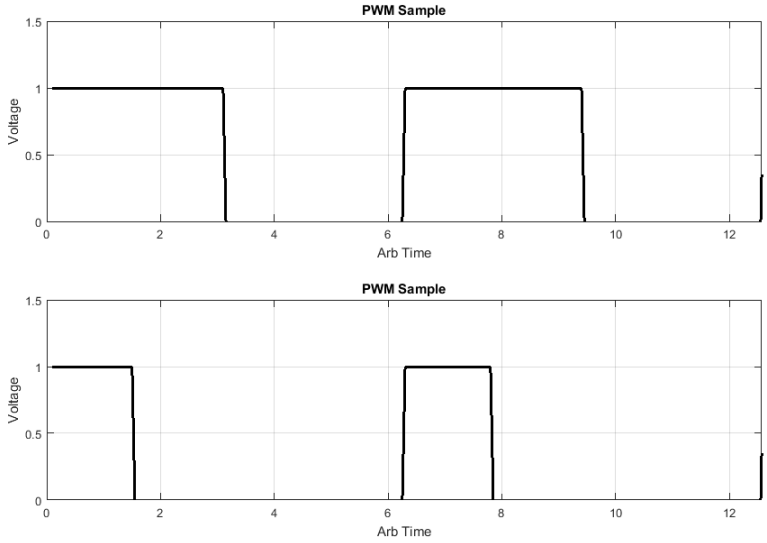


Figure 6: PWM example, demonstrating duty cycles generated from a timer module

As demonstrated in 5 and 6, the implementation of PWM is relatively straightforward due to the usage of a timer module, and computationally inexpensive.

The Universal Serial Communication Interface (USCI) module that is included within the device, is an exceedingly useful hardware add-on that allows for easier low level communication protocols to be implemented in hardware rather than in software thus additionally saving on costly clock cycles which can be redirected onto other tasks. This module in particular gives access to the ability to offload UART and SPI primarily

with partial support for I2C. Both main protocols are costly to implement in software, especially UART due to its very nature of a asynchronous communications protocol. The core reasoning behind the requirement of the USCI module is due to the need for the Smart Table project's further integration with a wireless communication module. As such, modules tend to use one of these protocols in the data transmission between it and another device. Considerations also have to be taken, as USCI is a generic term within TI documentation and does not indicate its true functions, i.e. most other devices in the value series only support SPI and I2C in their USCI module.

Ease of prototyping code is an additional draw to this particular device. As the IC fits within the socket provided by the value line launchpad and is supported, the MSP430G2553 is immediately available for prototype use. Then further down the road, in the final production phase, the inclusion of onboard JTAG is a low cost addition to the board and allows for on the fly code tweaking if required in the late stages. While considering all the advantages this chip offers, it was finally discarded due to the issue of memory. The amount of onboard memory is only 16KB and offers 512B of ram; while this is a massive amount of resources for most projects, the Smart Table has the problem of requiring a large amount of data that must be readily available for smooth operation. Additionally more is needed for further possible upgrades at a later date. The cost of swapping a microcontroller at a later date is much higher than the inital cost of using a more expensive microcontroller. Thus this chip was passed on after intensive investigation period. [T.Ia]

4.2.2 MSP430FR6989

The MSP430FR6989 microcontroller is part of the FRAM line of microcontrollers that TI offers. As with the msp430 based microcontrollers, the key points are low power, and quick response times. This particular device was chosen for possible implementation due to the amount of non-volatile memory it had on offering, a large 128KB of storage integrated into the chip. This amount would give plenty of working room for programming, as the minimum storage for a singular frame costs 13KB in the 512 LED implementation that was being discussed at the time. Therefore these chips were further investigated for feasibility of integration into the final product design. Additionally offered by the microcontroller that further made it unique are; a 32-bit hardware multiplier (MPY), three-channel internal direct memory access (DMA), a total of five timer modules with multiple capture/compare registers, 128-bit AES security encryption and decryption processor which is quite useful from random procedural generation, and a enhanced serial communication interface module (eUSCI).

This device's main selling point is its inclusion of FRAM. As such with any newer technology there are a list of pros and cons with the usage of it that must be especially investigated due to its fledgling status. Some of the advantages it provides are: data reliability, low power, speed, unified memory. Additionally due to it being a part of the msp430 line and TI's support, code can be easily ported over to the newer setup with only minor effort. Data reliability is an extremely interesting point, due to the struc-

Table 3: MSP430FR6989 Basic Specifications [T.lb]

Specification	Value	Summary of Interest
Main Clock Speed	16MHZ	Basic modern speed, average
Primary Storage	2KB	SRAM, then the FRAM additionally acts as Primary
Secondary Storage	128KB	FRAM storage, offers potentially higher READ-/WRITE times
Communication Module	eUSCI	Improved USCI module, essentially adds finer control
Clock System	Watch Crystal, Internal Programmable	Standard clock control, watch crystals are always useful
Additional Modules	Hardware Multiplier, DMA	Multiplication and memory access are computationally expensive, will help offload cost

ture of the cells, the data is almost completely unaffected by stray electric fields and radiation and on-top of that the cells have far greater read-write endurance than flash or EEPROM. As a consequence to the technology, the cells are also low power, without the requirement of charge pumps and/or higher voltages of comparable technology. Speed also is improved in this technology, with a far superior write time versus flash and EEPROM, and an additional compressed write + read/verify ability. Another interesting point of this technology and device integration is its unified memory interface, in which both the standard nonvolatile and volatile memory separation doesn't exist and instead they are one and the same. This leads to an even easier time when designing code and as such the programmer only has to keep in mind overall memory usage instead of two different sections and their usage. As a final addition to top off these bonuses for using a TI FRAM device is the ease of integration into an existing project; C code is completely portable from another msp device with only minor work needed. Even with the entirely different memory structure, the programmer doesn't have to bother with worrying about the differences to re-implement their code. Though of course there are things to pay for with such a technology, such as higher cost of production and lower memory density than comparable technologies.

The direct memory access module also comes integrated cleanly into this package. It provides a high speed and low cost method for moving predefined chunks of data with various methods. The most commonly used purpose of this module is for automatic pulling and moving data between peripherals with a minimum of cpu cost. As

this module runs in parallel to the main CPU, it doesn't pull any of the limited cycles for processing. The only considerations the programmer must give to the device is the standard race condition issue in which concurrent access to set resources happens in the wrong order. Another cycle saving hardware module is the hardware multipler; following is a basic implementation in pseudocode of a multiplication operation, multiplication is not an intrinsic low level operation in many microcontrollers.

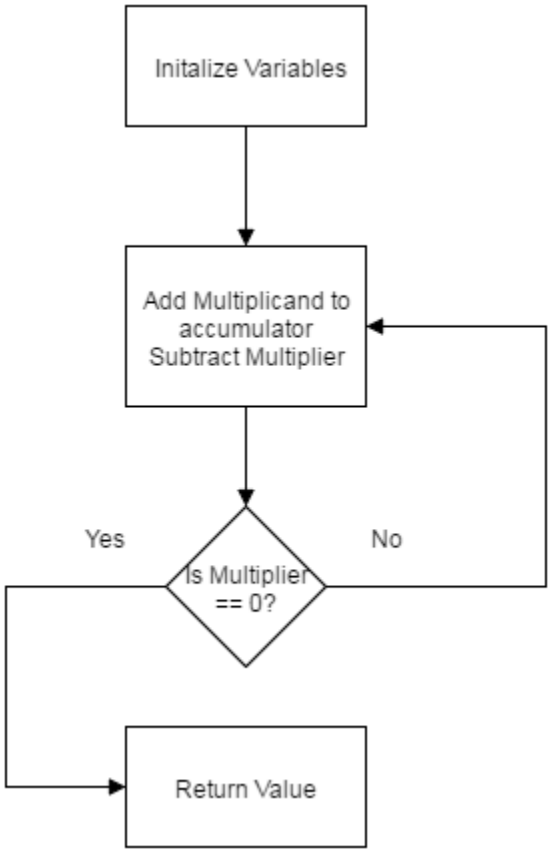


Figure 7: Flow of multiplication in software, in asm

As it is easy to surmise from 7, multiplication is a costly operation, and takes up copious amounts of CPU time when implemented in software. This hardware implementation offered by the MPY allows for an hardware offloading and thus increases speeds up to 17 times versus the software implementation. The module is accessed by predefined macros and is relatively easy to implement. A main draw to using the MSP430FR6989 in relation to this particular module is the possibility of implementing computationally expensive FFT which is a stretch goal for the Smart Table.

The timer module has previously been expanded upon but the main difference with this devices implementation of the module, is its inclusion of five modules with multiple registers for each. This opens the possibility of multiple parallel lines being driven with proper hardware generated PWM without resorting to bit banging. This is a core issue in the Smart Table project, as display driving is an issue requiring copious optimizations.

The security module included with the device also adds a useful ability, of generating an rng number with hardware which provides an optimization for procedural image generation to be also used in the display driving. Finally the eUSCI module is provided with this device. This offers to the project higher bit rates, better interrupt control, and better I2C implementation; all of which are potentially needed depending on the display control style taken and optimizations needed for driving.

Overall this device offered a multitude of improvements over the value line device previously chosen. The additional hardware modules especially offered significant clock optimizations needed for driving. Ultimately this device was passed up due to its hardware limitations imposed upon the FRAM memory. The memory controller is limited to a max speed of 8MHz and as the project requires copious amounts of read/writes this limitation would essentially cause the entire microcontroller to only be able to send display data and never process and of the procedural content or even access peripherals. Thus the memory controller was the main cause of abandonment. [T.Ib]

4.2.3 MSP430F6659

Table 4: MSP430F6659 Basic Specifications [T.Ic]

Specification	Value	Summary of Interest
Main Clock Speed	20MHZ	Max base clock, is a adequate speed
Primary Storage	64KB	A large amount to work with, will potentially help quite abit with buffer streaming
Secondary Storage	512KB	Allows a large potential in look-ups and more advanced code to speed up code
Communication Module	3 USCI, USB	Adds a large improvement to communication control and offloading
Clock System	Watch Crystal, Internal Programmable, External Crystal, FLL	More advanced clock system, adds in the ability for over-clocking by usage of external crystals and/or FLL usage
Additional Modules	Hardware Multiplier, DMA, DAC	Options to help offload computations, and potentially useful digital to analog conversion software

The MSP430F6659 is another device that was considered underneath the MSP430

line. The main considerations with this device were mostly performance related, due to the failure of the earlier devices. The main objects of interest were its slightly different clock system, the three USCI modules, the amount of flash plus ram available, and the inclusion of the MPY, DMA, and various other useful hardware modules.

The clock system capabilities were of key interest to system design. The clock capabilities are on par with previous mentioned microcontrollers, but the main focus on how to truly unlock the microcontroller's capability. The frequency locked loop used in the clock control is key to overclocking the system. Overclocking is fundamentally just running the system at a higher clock, this technique would have hopefully sidestepped an issue that was ran into earlier in testing. The overhead for sending data with the display system at the time was a massive percentage of the total CPU time available. Thus the idea for overclocking the system was investigated and deemed feasible with the implementation of the F6659 microcontroller. By adjusting the FLL outside the stable bounds it was predicted to be able to obtain a 50 percent increase in clock speed over the base max stable. Additionally to add to the total temptation of swapping to this device was its massive 512KB flash and 64KB ram which was a jump forward from the MSP430FR6989. This would open up a multitude of possible implementations of the software as more room was available to act as buffer. On-top of these previous notable improvements was the inclusion of multiple USCI modules which also opened a multitude of additional implementation ideas and also solved the issue of only having one bus for multiple devices using different protocols. And finally, all the previous mentioned hardware modules are included in the package, thus keeping possible implementations and optimizations freely available to squeeze out the maximum amount of efficient cycle usage.

In conclusion to this device consideration, the device was potentially powerful enough to handle the requested feature set for the Smart Table. Though, the main issues were the proposed instability of going outside of the given stable frequencies and thus relying on silicon lottery to keep the project flowing. Additionally even with this speed increase, some of the proposed software modules would have pushed the total cycle costs to the limit of whats available. Therefore this microcontroller is listed as a backup device for development if any events happen that pause production in the final microcontroller choice. [T.Ic]

4.2.4 MSP432P401r

A deviation from the previous generations of low power microcontrollers offered by TI, the MSP432P401r is a distinctly different device on offer. This microcontroller is an ARM based design, a strong change from the previous MSP430 generations. This new line of microcontrollers offers overall a higher base clock speed, more memory, and various new modules to accelerate production. This device offers to the developer improved floating point calculations with the integration or a floating point unit, clock speeds up to 48MHz, 256KB of flash, 64KB of ram, mappable ports (PMAP), and improved power management.

Table 5: MSP432P401 Basic Specifications [T.Id]

Specification	Value	Summary of Interest
Main Clock Speed	48MHZ	A large increase in clock for a low-power device
Primary Storage	32KB	Enough space to keep the code and smaller buffer safely inside
Secondary Storage	128KB	A large amount of room to work with for code
Communication Module	4 eUSCI	Copious options for communication control and offloading of work
Clock System	Watch Crystal, Internal Programmable, External Crystal	Decently complex and flexible system to hook into, more than adequate
Additional Modules	FPU, DMA, ADC, DAC, Voltage Supervisor, RNG Module	Complex array of hardware to hook into for optimizations

The inclusion of the floating point unit and combination of a higher clock speed makes this device a tempting choice, as FFT is a feature that is sought after for the project. Additionally this allows more wiggle room within the software implementation, as this allows for the construction of code that is partially modular and allows for the input of modules later to expand the feature set of the Smart Table. Another feature that shows promise in prototyping is the PMAP module, which allows for the developer to move certain pins and their functions to other pins. While immediately this may not seem like a massive improvement, it allow for easier isolation of certain pins; for example if a pin is experiencing interference from a neighboring pin's signal, the functionality of one pin can simple be moved to test the reasons for the inference, thus saving time in the debugging stage. Power management is also further improved in this device, a software controlled interface is provided to help with supply concerns, therefore allowing for the possibility of designing a system to handle brownouts while retaining data.

Overall this microcontroller proved tempting for further research and implementation. The main concerns were the new architectural changes to be accounted for, the cost/benefit of swapping to such a device with only a minor baseline improvement in offered speeds, and availability. The final decision was that the cost of implementation was higher than the expected return on investment and additionally the availability of the device was in question in time for the prototyping stage was another stumbling block; therefore this device was passed up for further research into available devices. [T.Id]

Table 6: ATmega2560 Basic Specifications [Atm]

Specification	Value	Summary of Interest
Main Clock Speed	16MHZ	Moderate speed for low-power microcontroller
Primary Storage	8KB	Will have to optimize streaming heavily
Secondary Storage	256KB	Will be used for large amounts of hardcoded lookups

4.2.5 ATmega2560

The ATmega2560 offers a variety of interesting alternatives to the MSP430 for use in implementation. It offers a rough parallel to the MSP430g2553 in terms of available features, speed, and available memory for usage. Additionally, the main components needed for the Smart Table implementation are available on the device, including pulse width modulation channels, and SPI communication modules. As such there is little discernible difference between the two microcontrollers in terms of pure specifications; the main difference comes from the available libraries to use as reference material, and the large amount of community support for the devices which is far superior to the TI offerings, thus making the device a "user-friendly" option for implementation.

A quick glance at the available libraries reveals a plethora of software with easy to understand interactions. Immediately noticeable and prepackaged with the specialized IDE for programming these devices are libraries for; EEPROM reading/writing, Ethernet communications, Servo control, LCD interfacing, and even a wifi library. That's only a small sampling of the prepackaged libraries, there's many more easily available, and even more code that easily found for just about everything possible when using one of this microcontrollers. As the community for these devices is massive, almost every possible issue you could run into, someone else already has and has posted the solution thus further making this a tempting device as community support is a massive bonus when learning how to use and develop on a new platform.

Finally though, this device was passed on for a few reasons. Despite the enticing community support, the device itself isn't that powerful even with its decently optimized architecture. Additional the decision was made to use primarily Texas Instruments devices for the bulk of the control work due to a semi-familiar environment and stronger knowledge of the low level interactions needed to optimize the code generation. Overall, the device was promising but passed up to the need for a higher power device and something inside the TI line-up. [Atm]

Table 7: TMS320F28379D Basic Specifications [T.Ie]

Specification	Value	Summary of Interest
Main Clock Speed	200MHZ	Massively powerful, will allow the additional features that are highly computationally expensive
Primary Storage	164KB	A copiously sized working space for the expensive code implementation
Secondary Storage	1024KB	This allows for a large amount of optimizations in code, with the ability to hardcode in look-up tables
Communication Module	4 USB, uPP, CAN, SPI, SCI/UART, I2C	Covers every possible issues pertaining to communications
Clock System	Unified System	High accuracy and moderately flexible system
Additional Modules	FPU, TMU, CLA, ADC, DAC	Large amount of hardware for offloading processes

4.2.6 TMS320F28379D

This microcontroller's basic specifications of 200MHZ clock speed, 1024KB of flash storage, and 164KB SRAM already set it apart from the competition. In addition to the base speeds it must be noted that this device has the D designation implying that the device is of the dual-core type. As the device is a dual core, each core runs independently from the other at the full rated speeds (200MHZ) and contains each an independent memory and ram allocation with a shared buffer between the two for communications. This hardware design leads to the implementation of a real-time operating system with partial threading support which will be highly useful for the Smart Table implementation as the feature-set is a computationally expensive and costly set of features to implement on a single microcontroller. Additionally the microcontroller includes many other features which make it a very enticing microcontroller for implementation as the core for the project.

On-board this microcontroller are a number of useful and interesting hardware modules for various uses. The easiest and mostly immediately recognizable modules to examine are the communication interfaces that are included in the package. As this microcontroller is a much more advanced device compared to more conventional ones, the communications implemented are additionally more advanced; as such the included communication modules are capable of handling USB 2.0, uPP, CAN, SPI, McBSPs, SCI/UART, and I2C which essentially covers every possible basic communication pro-

tocol needed. The more uncommon modules included on this unit include; control law accelerators, floating point units, trigonometric math unit, and a Viterbi math unit. Each one of this modules are exceedingly useful and interesting in the offered functionality. The control law accelerators are a different take on offloading floating point instructions, this units are essentially floating point units with independent instruction pipelines and datapaths which run concurrently and parallel to the main CPU while additionally being clocked at the same speed and controlled by interrupts. This offers the ability to offload a number of instructions without any sort of cycle stealing and without any intervention of the main CPU which is exceedingly useful in any peripherals that need heavy processing but the additional main CPU load would be undesirable behavior. The TMS320F28379D additionally comes with a directly mapped floating point unit additionally to the CLAs. This allows further refinement in computation speeds depending on the application. The Trigonometric math unit (TMU) and the Viterbi/complex math unit (VCU-II) are another two very useful modules integrated with direct access from software. The TMU offers higher performance in applications in which odd signals need to be processed such as phase and magnitude in power applications or FFT algorithms. The Viterbi unit additionally expands the signal processing capabilities by allowing for the implementation of the Viterbi butterfly algorithm which is an useful process in modern digital communications for error estimations at the least. Additionally the instruction sets it adds allow for further FFT refinement which is an implementation that is included in the Smart Table stretch goals. The microcontroller also does offer the standard ADC and PWM controls and various other analog related capabilities as such is required in a normal microcontroller design.

The main concern with this microcontroller implementation is race conditions resulting from the implementation of threading to control the access to each individual core and to guarantee the proper distribution of clock cycles without having to resort to imprecise clock counting or other methods used in flow control. Another issue arising with the implementation of this microcontroller is the sheer number of pins to be dealt with in the board design, though this is a more minor detail, it still affects final implementation of the device. Overall those concerns are far outweighed by the benefits this microcontroller brings to the table, as the sheer amount of power required for clean implementations of the features to be included in the Smart Table are quite costly and the design team would prefer not offloading the processing and control to another device such as a computer or microcomputer like the raspberry pi. As such this microcontroller hits the requirements set out by the internal design team for implementation of all possible features and stretch features in a small package. [T.Ie]

4.3 Serial Communication Technologies

Wired communications characterize any technology or protocol used to establish communication between components within the hardware design of the product, such as data transfer between a microcontroller and LEDs. Serial communications involve transmitting a single bits of data through a channel sequentially, as opposed to parallel

communications, which involve sending multiple bits of data over a number of channels simultaneously.

Several standardized serial communication methods exist that define methods for sending and receiving data synchronously or asynchronously, bit-by-bit. Synchronous serial communication methods such as I²C and SPI could be useful for interfacing the microcontroller with other hardware subsystems such the Bluetooth module or the LED matrix, whereas asynchronous methods such as UART are useful for transmitting and receiving data between the smart table and external devices. It would be ideal to incorporate hardware with native functionality for these communication methods, but bit-banging, a method to support a communication protocol within the software, is another option that could provide flexibility and a greater hardware selection.

Due to the nature of the smart table, which requires hundreds of LEDs to have a good resolution, using a well-established serial communication protocol is absolutely critical. Many microcontrollers do not have a large number of pins, so a functioning the smart table would be impossible with a serial communication protocol due to the sheer number of wires that would be required for an LED matrix. The smart table needs a way to communicate with and drive the RGB LEDs rapidly, which is heavily dependent on the protocol used and the hardware capabilities of the selected microcontroller. The importance of these communication protocols to a project incorporating such a large number of connections becomes more apparent when discussing buses, which effectively allow for fewer pins to be utilized on the microcontroller.

4.3.1 I²C

I²C, also called Inter-Integrated Circuit, is a serial communication protocol originally designed in 1982 by Philips Semiconductor, now known as NXP Semiconductors. I²C has undergone a number of substantial revisions since its inception, beginning with Version 1 and arriving at the current Version 6. These versions have been mainly concerned with increasing the speed of signal transmission through the offering of high-frequency modes for the data and clock signals. As of 2006, the protocol is available for use by the general public, royalty-free and without the need for a license, although NXP charges customers to obtain slave addresses. I²C remains a popular protocol and is supported by a large number of devices. It is usually used at low speeds and for communication between integrated circuits.

I²C uses a serial bus to facilitate communication between devices, allowing multiple masters to communicate with multiple slaves. The protocol is primarily intended for use over short distances, such as communication within the same printed circuit board. Conventional I²C has both a 10 Kbps slow mode, a 100 Kpbs standard mode, a 400 Kbps fast mode, and a 3.4 Mbps high speed mode. An I²C bus utilizes 7-bit addresses and is comprised of two signals, known as the two-wire interface (TWI):

- Serial data (SDA): The data signal. This is the transmission line through which data is bidirectionally transmitted between masters and slaves.
- Serial clock (SCL): The clock signal. The clock signal is created by the master of the bus, however, slaves may command the clock to a lower frequency for extra processing time or when the master is sending too much data.

The serial data and serial clock lines use pull-up resistors, with 4.7 $k\Omega$ resistors being common starting values. The protocol identifies connected devices and classifies them into either master or slave roles. Each role defines the behavior of the device and what it is permitted to do at a certain time. The masters and slaves are distinguished as nodes:

- Master node: The node that sets the clock signal and begins transmitting data to the slaves.
- Slave node: The node that acknowledges the clock and receives data transmitted from the master.

The two-wire bus supports multiple master nodes and can accommodate a staggering 1008 slave nodes. The masters and slaves may even swap roles between signal transmissions. However, the multi-master bus does not permit masters to communicate with other masters simultaneously. There are four modes of transmission that characterize the master-slave nodal behavior:

- Master transmission: The master node transmits data to a slave device.
- Slave transmission: The slave node transmits data to a master device.
- Master reception: The master node receives data from a slave device.
- Slave reception: The slave node receives data from a master device.

I²C signal transmission is governed by a specific bit manipulation scheme. Messages, denoted in all capital letters, are encoded in the signal such as START, STOP, and ACK (acknowledge). Data is sent in one-byte segments. When the master begins transmitting, it uses a START bit along with the desired slave's 7-bit ADDRESS. The last bit determines if data is being read or written. When a slave receives this message, it transmits an ACK bit, and the master continues to operate in either the transmission or reception mode depending if the starting message encoded the last bit with a (0) for write, or (1) for read. The slave then continues in the appropriate mode; if the master is transmitting, the slave must be receiving, and if the master is receiving, the slave must

be transmitting. When either the master or the slave is receiving data, they send an ACK bit after each byte that is received. When the signal transmission is completed, the master send a stops bit to end the transmission or begins a new transmission with another START bit.

As seen in 8, the microcontroller master node is connected to the SDL and the SDA along with three slave nodes: a microcontroller, a digital-to-analog converter, and an analog-to-digital converter. Pull-up resistors are used to connect the two bus lines to the power supply. It is common for peripheral devices such as these to all be included within the same microcontroller, but none-the-less they serve as good examples. What distinguishes I²C from SPI or UART is that the SDA and SCA bus lines are open-drain. The implication of this is that they can only assert a signal low, rather than high. For this reason, I²C is considered to be active-low.

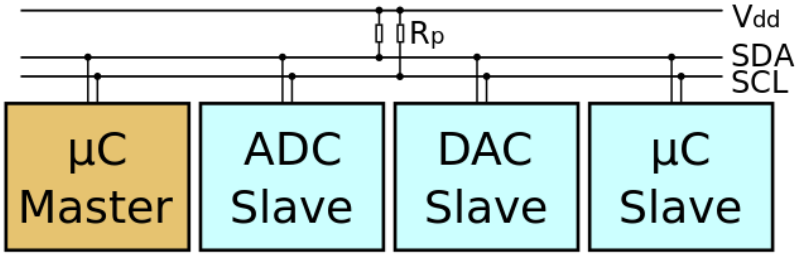


Figure 8: I²C Bus. *The use of this image is governed by the GFDL.* [Bur16b]

The yellow regions in Figure 9 shows that by default, the signal wires are set high using the pull-up resistor. This hardware configuration allows all devices to safely read or write to the SDA and SCL lines at appropriate times. Master devices will wait until a STOP bit has been asserted before trying to write over the bus lines, and slave devices will read the bus lines to check whether the address following the START bit matches. Non-matching slaves will wait until a STOP bit is detected before reading again. Because of the active-low/open-drain format, masters can also utilize a feature known as clock stretching in the event that slaves are unable to keep up with the speed of the master. The advantage to this unique implementation is that the lines are highly resistant to signal conflicts, such as simultaneous high and low assertions from different devices, which could result in corruption of the signal or damaged hardware.

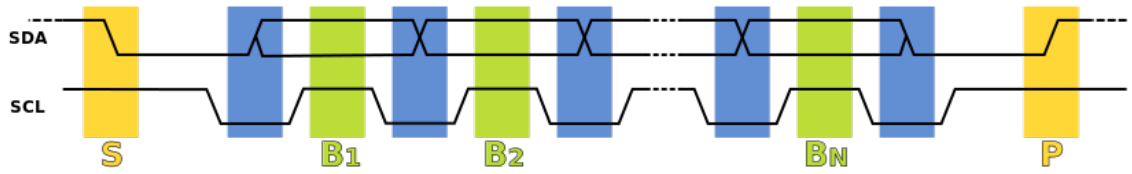


Figure 9: I²C Timing Diagram. *This image has been released to the Public Domain.* [Flo16]

In the case of the smart table, I²C could be used as the interface which would allow the

main microcontroller to serve as the master. Since 3.3 volts and 5 volts are common voltages used for the protocol, I^2C wood be an excellent choice for the smart table's microcontroller and wireless communication module. As the master, the microcontroller would control the exceptionally large LED matrix, and the large number of LEDs in the matrix would be divided into slave groups. This method would allow the use of only a few input pins on the microcontroller, which would be highly manageable. Each slave could have an individual slave address and be assigned assigned to a specific software function, such as displaying time, date, or weather.

4.3.2 SPI

The Serial Peripheral Interface Bus, better known as SPI, is a method of serial communication created by Motorola. It is specified for use over small distances and has been considered a standard embedded system communication method for some time due to its simplicity and versatility. SPI is synchronous, which means that the clock signal is used to synchronize data and transmission. Common modern technologies that utilize SPI are LCD screens and SD cards. SPI is extremely simple at its core. Devices that utilize SPI transmit and receive data at the same time, also referred to as full-duplex mode. Like I2C, SPI utilizes the master and slave method.

SPI buses contain four important wires:

- Slave Select: The Slave Select (SS) is the signal that the master uses to choose the slave in which to engage in communication
- Serial Clock: The Serial Clock (SCLK) is a clock signal that synchronizes all SPI signals, generated by the master.
- Master In Slave Out: The Master In Slave Out (MISO) is a unidirectional data signal that goes from the slave output pin to the master input pin.
- Master Out Slave In: The Master Out Slave In (MOSI) is a unidirectional data signal that goes from the master output pin to the slave input pin.

The SPI protocol is also called a four-wire interface (FWI), which is a higher number than most other serial communication protocols, namely I^2C. The SPI protocol uses a single master but supports multiple slaves. At the start of transmission, the master will generate an SCLK signal at a mutual frequency. This frequency will typically be compatible with the selected slave device. In order to communicate with the slave, the master will then pulls the respective slave select line low and then wait if necessary. Then, the data is transmitting in full duplex mode, with the the master sending data over the MOSI line and the slave sending data over the MISO line. If multiple slaves devices are present, each slave requires its own SS line in order to be addressed by the master.

Figure displays the circular transmission of data between SPI shift registers of the master and slave. During signal transmission, 8-bits of data from a shift registers in the master is swapped with data from a shift register in the slave. The most significant bit from either the master of slave is replaced with the least significant bit from its counterpart. This process is continued for as many clock cycles as it takes until until the shift registers have completely swapped values. After that, the shift registers are reset to prepare for another transmission. When transmission is complete, the slave is released from the SS line.

The signal transmission is all done with respect to the SCLK. With these options, there are four potential modes that communication can occur for most microcontrollers:

- Mode 0: CPOL $= 0$, CPHA $= 0$. The SCLK is active high and data is captured on the rising edge.
- Mode 1: CPOL $= 1$, CPHA $= 0$. The SCLK is active high and data is captured on the falling edge.
- Mode 2: CPOL $= 0$, CPHA $= 1$. The SCLK is active low and data is captured on the rising edge.
- Mode 3: CPOL $= 1$, CPHA $= 1$. The SCLK is active low and data is captured on the falling edge.

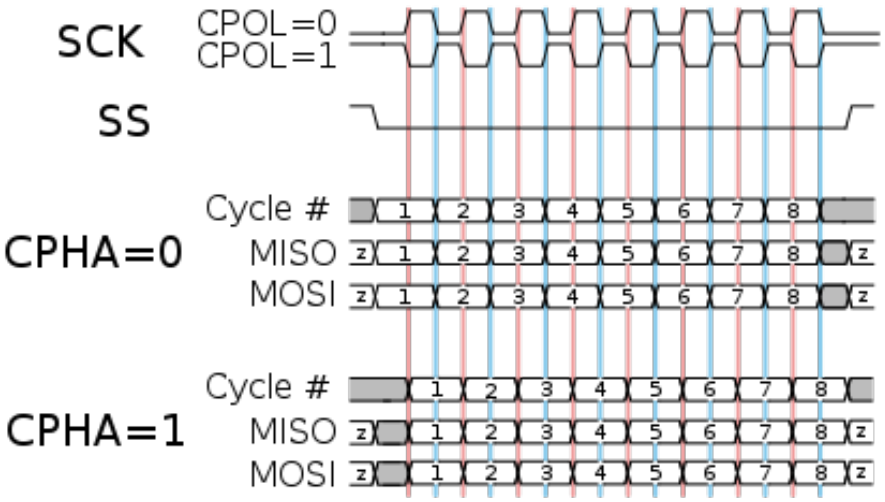


Figure 10: SPI Timing Diagram. *The use of this image is governed by the GFDL.* [Jor16]

Figure 10 shows an accurate timing diagram of SPI signal transmission. The timing of the SCLK signal are characterized by two options: the clock phase (CPHA) and the

clock polarity (CPOL). The clock polarity is responsible for the clock signal inversion or noninversion–(0) for active high or (1) for active low, whereas the clock phase is responsible for shifting the sampling edge clock signal–(0) for the rising edge or (1) for the falling edge.

As previously mentioned, the SPI protocol can be used with a single slave or with muliple slaves. When used with multiples slaves, there are two approaches:

- Independent Slave Select: Each slave has is own select line leading from the master.
- Daisy-Chained Slave Select: There is a single slave select line that branches to each slave.

As seen in Figure 11, the independent slave select method involves each slave to have its own select line leading from the master. This is the most common method employed for SPI communication. In order to accomplish this, the active slave select must be pulled low while the others are kept high, with a pull-up resistor utilized between the select line and power supply to prevent cross-talk. One MOSI line from the master is branched to each slave. The MISO lines of the slaves are also tied together, however, the active low configuration prevents the other slaves from trying to communicate with the master at the same time. A disadvantage of this method is that it can utilize too many pins on the microcontroller for the select lines.

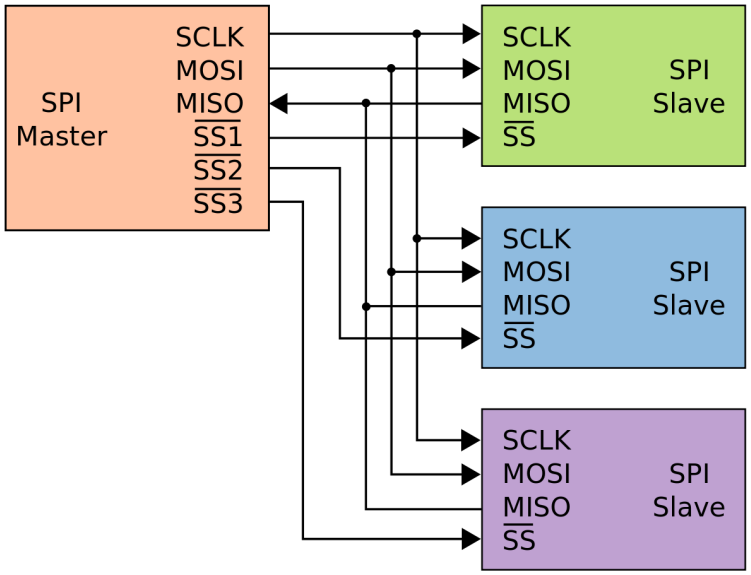


Figure 11: SPI Bus with Individual Slaves. *The use of this image is governed by the GFDL*. [Bur16c]

Figure 12 displays the daisy-chain configuration, which involves each slave sharing a single select line that branch off from the master. In this method, each slave is connected together. The MOSI line from the master is connected to the MOSI line of the first slave. The MISO line from that slave is directly connected to the MOSI line of the next slave, until the MISO of the last slave is connect to the MISO of the master. This configuration essentially acts as a circular buffer, with the same data transferred from the shift register of one device to the next on each clock cycle. The advantage to this method is that fewer pins can be utilized due to the use of a single slave select, as well as support for interfaces such as JTAG.

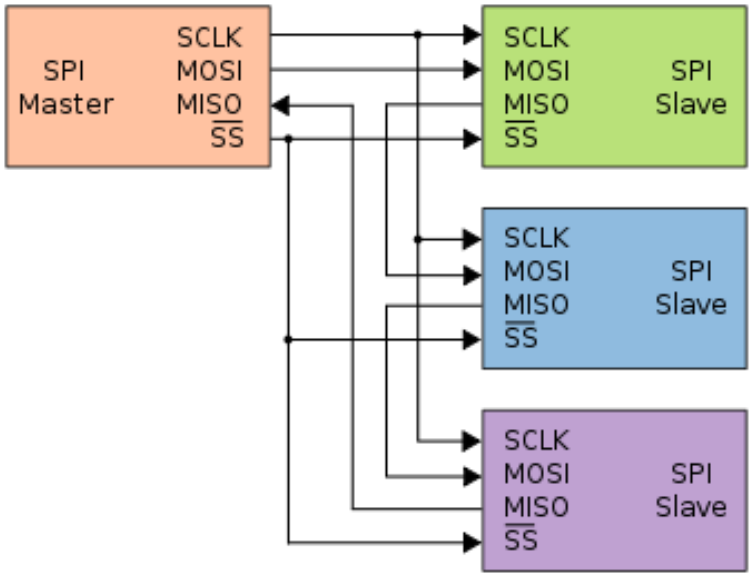


Figure 12: SPI Bus with Daisy-Chained Slaves. *The use of this image is governed by the GFDL*. [Bur16a]

Overall, SPI is much simpler and faster than I2C due to its full duplex design and can support speeds in excess of 10 Mbps, it requires more signal wires than most protocols and can utilize a large number of slave select lines if used in the standard configuration. There are limitations such as the fact the slaves cannot explicitly communicate to each other, and that only a single master is supported using the protocol. There is no form of slave acknowledgement or error checking functionality. As a result, I2C is generally considered to be the smarter and more robust protocol between the two.

4.3.3 UART

A UART, short for universal asynchronous receiver/transmitter, is a hardware device that effectively enables serial communication. It is an interface used and supported by many microcontrollers as an interface between parallel and serial connections. At its core, one side of the UART has a parallel bus with multiple data lines, and the other has

two simples serial lines: the receiver (RX) and the transmitter (TX). It is implemented by various well-known communication standards such as RS-232.

The UART is responsible for:

- Converting the data from the microcontroller received through the parallel bus into serial data sent through the TX serial line.
- Converting data received through the RX serial line into data sent over the parallel bus lines understood by the microcontroller.
- Checks the parity of transmission by inserting a parity on output signals and removing the parity bit on input signals
- Handles interrupts from peripheral devices.

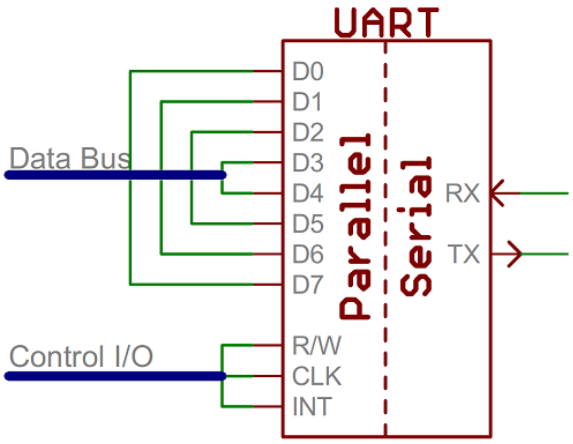


Figure 13: UART Interface. *The use of this image is governed by the CC BY-SA 4.0.* [Lina]

As seen in Figure 13, serial and parallel data is converted through the use of shift registers within the UART. Since UART supports simplex, half-duplex, and full-duplex operation, this determines the number of shift registers. For full-duplex operation, two shift registers are required for both transmitted and received data. Bytes of data from the microcontroller are passed through the shift register and transmitted through the serial lines by-by-bit and vice versa. Data manipulated by some UART interfaces may be stored in buffer on a first-in-first-out (FIFO) manner. Data may be added to the buffer by the microcontroller and transmitted in a single burst, or stored in the buffer by the serial RX until the microcontroller can obtain the data.

UART reception is achieved using a clock signal that is a multiple of the baud rate (bits per second), usually 8. When receiving a signal, the RX line looks for the starting bit by checking if the bit is received in half or more of the clock signal time multiple. If the signal is accepted, it is sampled, passed through the shift register, and received by the

microcontroller. The UART sets a flag that the data has been received and initiates an interrupt so that the data can be utilized by the microcontroller The clock of the UART is data dependent and synchronized with every new signal reception.

Figure 14: UART Timing Diagram. *The use of this image is governed by the CC BY-SA 4.0*. [Linb]

Figure 14 shows how bits are handled through the UART interface. When the microcontroller begins sending data through the shift register, a start bit, parity bit, and stop bit are created by the UART and the data is sent through the serial TX line. Similarly, the UART sets a flag that signals to the microcontroller that it is busy, indicating that the shift register or buffer contains data.

4.4 Wireless Communication Technologies

Wireless communications characterize any technology used to establish external communication with the product, such as accessing the product from a smartphone or computer interface over a local or personal area network. The smart table requires some mechanism for it to communicate externally and wirelessly.

Wireless communication would allow the smart table to integrate with other commonly used devices and have a minimalistic appearance, without depending on long and unwieldy ethernet cables, which could cause individuals to trip and fall. Wireless communication could also eliminate the need for hardware control mechanisms such as touchscreens, since the device's interface could be provided over the network in the form of a graphical user interface (GUI) to interact with it. The smart table is intended to operate within a typical indoor environment, such as a living room or bedroom, which may contain moderate levels of electromagnetic interference (EMI) presented by other common consumer products such as cellular phones, microwave ovens, laptops, routers, radios, and more. As such, a robust communication protocol that handles noise must be selected.

A number of different solutions exist to provide wireless functionality to the smart table in a compact manner. Radio-frequency (RF) circuit design is widely considered to be difficult, especially for novices, due to their sensitivity to noise and the tight specifications required to operate on a specific frequency band. The manufacture of RF circuitry can be time intensive, since small defects can negatively impact performance and cause a device to perform outside of its established frequency specifications. A considerable amount of testing is required for RF devices to ensure that they are safe, since leaky RF devices can pose a threat to individuals with medical devices by inter-

fering with their proper operation. The legal restrictions on radiation are established by the Federal Communications Commission (FCC) and may carry with them heavy fines or imprisonment.

As a result of these difficulties, embedded RF modules exist on the market that have been designed so that they can be easily used within any product that needs wireless communication. There are RF modules that incorporate a number of different wavelengths and frequencies on the electromagnetic spectrum, some of which will be discussed. These devices typically contain all of the necessary hardware in compliance with protocols relevant to the respective underlying technology. They may have internal or external antennas that have been optimized to minimize signal attenuation as much as possible.

The use of a module to incorporate RF communication is suitable for the smart table because it can save a significant amount on the development time, since it would also have a greater success rate. As previously mentioned, there is a low chance that an amateur-designed radio circuit will work due to the advanced nature of RF design. Antenna design in an of itself is a graduate undertaking and would require more time and resources to accomplish. In addition, such a module would be cheaper to implement wireless connectivity for the smart table than a custom solution. A compact embedded module with an integrated antenna would ideally have a wide compatibility with a broad number of devices, such as laptops, cellphones, and computers. The smart table could be conveniently operated with one of these devices in a conspicuous manner without wires. Three wireless communication technologies that immediately come to mind for the purposes of the smart table are infrared, Bluetooth, and Wi-Fi. The advantages and disadvantages of these different types of communication will be discussed in further detail.

4.4.1 Infrared

Consumer Infrared (IR) refers to a number of different standards that utilize light within the nonvisible infrared spectrum to facilitate wireless communication. It is a proposed communication mechanism due to the affordable nature of IR transmitters and receivers. Devices that typically employ IR include television as well as remote controls that operate a wide range of devices. Although there are a number of different implementations, IR transmitters typically operate by sending modulated IR signals encoded with a command to an IR receiver via an IR LED. The IR receiver uses an IR photosensor to detect the signal which is then demodulated and interpreted, which differs by device. The Infrared Data Association (IrDA) is an organization that manages the protocols and standards for products that utilize IR.

Consumer IR has some limitations that may not be well suited to the smart table. Due to the nature of IR light, is unable to penetrate most opaque physical substrates such as walls. As a consequence, the transmitter requires a direct line-of-sight (LOS) to the receiver. This limitation may be a cause for concern in a commercial space or

dwelling that may contain many obstacles that would impede transmission, such as individuals or furniture. The smart table would require an omnidirectional IR receiver that would need to be placed in an unobtrusive manner, so that it could be operated without impediment.

Some modern smartphones conveniently contain IR blasters, which can emulate infrared remote controls. This would facilitate the creation of a mobile application to control the smart table using the smartphone's IR blaster. However, most smartphones do not possess IR blasters, so the smart table would require an external remote control device to serve as the IR transmitter to maintain its accessibility. External hardware such as a remote control is nonideal because it could become lost or separated from the smart table, or break, rendering the smart table uncontrollable. In addition, it would increase the development time of the project due to the separate remote control PCB design and manufacture.

4.4.2 Wi-Fi

Wi-Fi is a brand developed by the nonprofit organization known as the Wi-Fi Alliance, which encompasses the technology that lets devices connect to a Wireless Local Area Network (WLAN) to communicate with each other. It has emerged as a dominant wireless communication standard present in many electronic devices on the market, and due to its ubiquity, it is widely supported. Wi-Fi devices are certified by the Wi-Fi Alliance for compatibility and usage with other Wi-Fi devices. Products with Wi-Fi are based on the IEEE 802.11 standard, a series of protocols that describes the characteristics of the physical layer and media access control sublayer of a wireless network. Together, the physical and media access control layers form the backbone of the 802.11 protocol.

The media access control (MAC) sublayer of the 802.11 is contained within the second lowest layer of the Open Systems Interconnection (OSI) model, a top-level representation of a telecommunication system composed of seven different abstraction layers. The MAC sublayer adds methods to address devices and access channels, which allows multiple devices to connect to a shared network. It has support for both full and half-duplex communication and also provides several additional functions such as error checking. Due to the MAC sublayer, devices connected to a network are assigned unique MAC addresses which allow them to reliably send and receive data packets. A good analogy is the way that homes with addresses may send and receive mail to other homes. The MAC sublayer essentially connects the physical layer with the the logical link control (LLC) sublayer, which is another layer that allows high-level logical frameworks such as the Internet Protocol (IP) to interoperate with each other over networks.

The physical layer (PHY) of the OSI model is the lowest layer, which is comprised of the hardware involved in the direct manipulation of the bitstream. The physical layer provides functionality to modulate transmission signals via the Physical Coding Sublayer

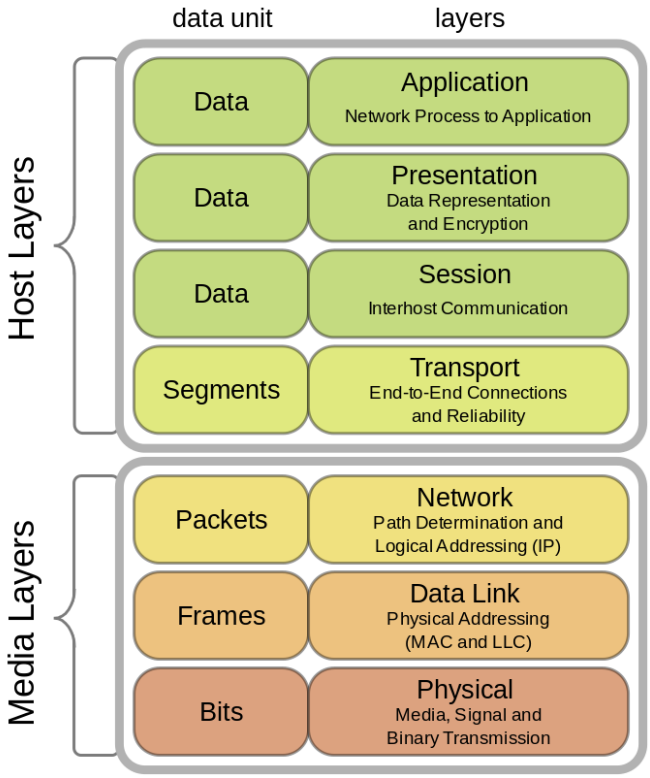


Figure 15: OSI Model. The use of this image is governed by the GFDL. [Off16]

(PCS). It also provides functionality to connect the MAC layer to the physical transmission medium, such as an antenna or cable, via the Physical Media Dependent (PMD) sublayer.

The 802.11 protocol has undergone many lettered revisions and permits wireless communication over The industrial, scientific, and medical (ISM) frequency bands, which are exempt from FCC licensing. The two main frequency bands that 802.11 family devices chiefly operate on are the 2.4 GHz and 5 GHz ISM bands. The 2.4 GHz band is less heavily regulated, and devices within this frequency spectrum are sometimes prone to inference from other devices using the frequency, such as microwave ovens. The 802.11 family uses a number of half-duplex modulation techniques, and 802.11 compliant devices can serve as either hubs or terminals to transmit and receive data on a network.

Wireless routers are 802.11 compliant devices that can serve as a wireless access points (WAP) for other devices to connect over the 2.4 GHz or 5 GHz frequency bands separately or simultaneously. Networked devices (commonly called nodes) that connect to wireless routers utilize transceivers known as wireless network interface controllers (WNIC) to send and receive data through the network in either of two configurations: infrastructure or ad-hoc. The infrastructure configuration allows nodes to connect

to the access point, whereas the ad-hoc configuration allows multiple nodes to connect to each other without a WAP.

Although Wi-Fi is a very mature platform that has existed in some form since 1997, it may not be explicitly necessary for the smart table. The smart table would be dependent the wireless network, which may not be accessible in all households. The use of a wireless network also introduces the problem of network security. A number of different wireless encryption methods exist for networks, such as Wired Equivalent Privacy (WEP) and Wi-Fi Protected Access 1/2 (WPA1/WPA2). Most consumer wireless networks today utilize either the WEP or non-enterprise WPA2 standards, which allow devices on the same network to view and access each other.

Due to security vulnerabilities in certain Wi-Fi networks, an adversary could gain access to a network and take advantage of devices on them. Networks with weak passwords could be easily accessed by an outsider to gain access to the smart table and exploit it. The malware Mirai is just one recent example of a virus that has been designed to infect consumer Wi-Fi routers with weak passphrases. Once compromised, the malware takes control of IoT devices on the network, such as DVRs, televisions, or even the smart table and uses them in a large-scale botnet. The devices are commanded to flood packets to a target network, called a distributed denial-of-service (DDoS) attack, which effectively takes them offline for the duration of the attack.

Another downside of utilizing Wi-Fi is that wireless networks today are already saturated by a myriad of other Wi-Fi capable devices. Because bandwidth on a wireless network is finite, additional devices could negatively impact the quality of service on the network. Network performance measures, such as the upload and download speeds, are sensitive to the number of devices simultaneously using the network. In an age of increasing internet usage for multimedia downloading and streaming, it would be ideal to avoid this situation entirely by utilizing another equally capable wireless communication technology.

4.4.3 Bluetooth

Bluetooth is another branded technology that allows devices to wirelessly communicate across short distances using ultra high frequency (UHF) radio waves in the 2.4 GHz ISM frequency band. Bluetooth was formerly standardized as the IEEE 802.15.1, but has since been maintained by the Bluetooth Special Interest Group (SIG). This standard facilitates the creation of low-powered and portable wireless personal area networks (PAN), which let devices communicate with each other in an ad-hoc manner (although uplinked communication of devices in an infrastructural manner is possible).

Bluetooth is distinguished from Wi-Fi such that it operates on much lower distances and is not focused on providing communication to devices through an access point. Instead, Bluetooth emphasizing direct communication between devices through a master-and-slave communication scheme and portability as a wire-replacement technology.

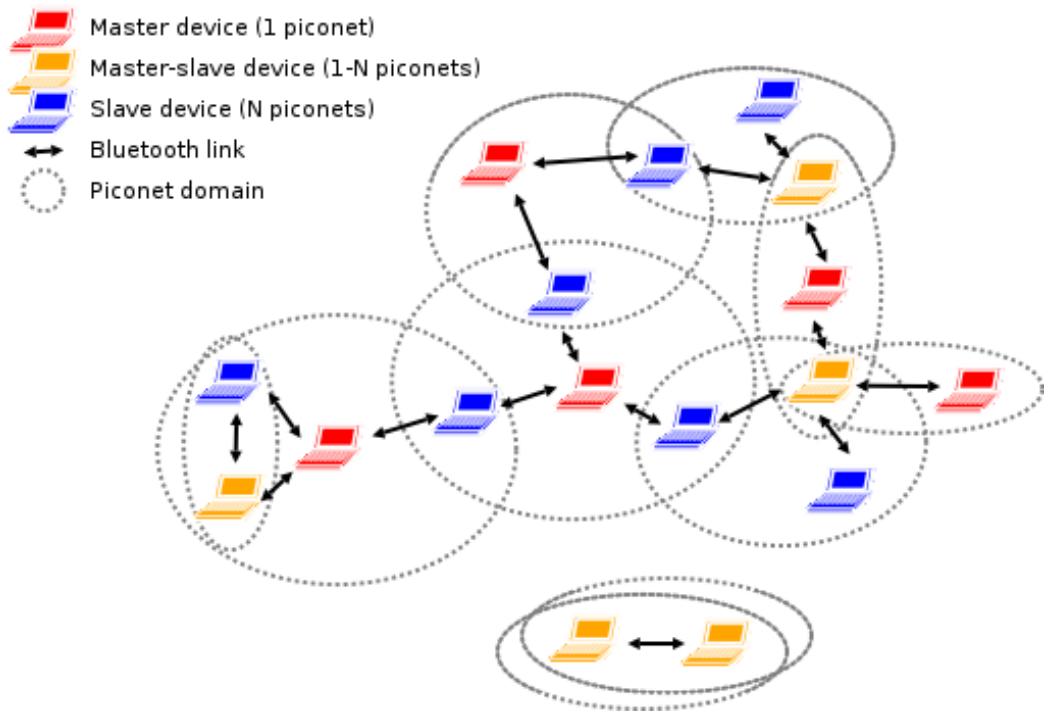


Figure 16: Bluetooth network topology.The use of this image is governed by the GFDL. [Bla16]

Connected Bluetooth devices may swap roles; for example, a car may initiate a connection to a phone as a master, but may then become a slave while operating. Masters can communicate with slaves, however, slaves can not communicate with each other. A Bluetooth network is called a piconet, whereas a scatternet is a higher level topology that consists of multiple piconets. Devices in the scatternet may operate as a master in one piconet, while simultaneously operating as a slave in another piconet. The clocks of the slave devices are synchronized with the clock of the master device. In even time slots, data packets are transmitted by the master and received by the slave, whereas in odd time slots, the reciprocal is true. Bluetooth connections boast a theoretical distance of 100 meters, depending on the Bluetooth version.

Classic Bluetooth utilizes a modulation technique called Frequency Hopping Spread Spectrum (FHSS) to transmit signals. FHSS is accomplished by rapidly "hopping" a signal between channels. The signal is split into several packets, which are hopped 800 times per second over 79 different 1 MHz channels, which may be adaptively selected by avoiding the channels at heavily trafficked frequencies. In Classic Bluetooth, a master may communicate with up to seven slaves, the maximum of which may differ between devices. Classic Bluetooth has a maximum throughput ranging from 0.7 to 2.1 Mbps.

Bluetooth Low Energy (BLE), also known as Bluetooth Smart, is a newer variation of

Bluetooth that uses a smaller number of channels (40) with larger spacing between channels (2 MHz) in its modulation scheme. It does not define a maximum number of slave devices that a master may have. BLE devices typically do not engage in continuous transmission and have a maximum throughput of only 0.27 Mbps. Both of these features lend themselves to the low energy functionality of this Bluetooth variation. The need for BLE is evident based on the growing number of portable devices on the market, which prioritize low energy consumption and long battery life. As such, BLE may not be explicitly necessary in non-battery operated devices, such as those connected to mains electricity.

Table 8: Wireless Technology Comparison. [Meo]

Specification	**Serial Infrared (SIR)**	**Bluetooth 2.1**	**802.11n (Wi-Fi)**
Frequency	850 - 900 nm (wavelength)	2.4 GHz ISM Band	2.4 GHz/5 GHz ISM Bands
Maximum Range	1 m	100 m	70 m
Maximum Data Rate	115 Kbps	3 Mbps	54-600 Mbps
Power Consumption	20-50 mA	5-15 mA	50-110 mA
Nodes Supported	1	7	32

Bluetooth is considered to be the ideal wireless technology to implement in the smart table. In comparison to infrared and Wi-Fi, Bluetooth has excellent range and suitable data rate characteristics as shown in Table 8. There are several different Bluetooth versions in existence, each presented as a revision of the previous version with added features or modified performance specifications. Legacy Bluetooth versions, such as version 2.1, are still commonly used in electronic designs for reasons such as cost, features, and performance. A number of different chips readily available for use in the smart table were examined for suitability based on such factors.

4.5 Bluetooth Module Considerations

There are a number of off the shelf Bluetooth modules on the market that can be quickly and easily integrated into the smart table to add wireless communication functionality. Here, we will discuss a couple of popular modules and select one that is most suited to the project.

4.5.1 RN-42 Bluetooth Module

The RN-42 is a Bluetooth module originally designed and manufactured by Roving Networks, although it has since been acquired by Microchip Technology, Inc. This chip is extremely compact and is suitable for anyone who wants to add wireless communication to their projects. A flexible chip, the RN-42 is compatible with the widely utilized

UART and SPI protocols. It has an antenna trace integrated into the PCB, which is suitable for individuals who lack the time and extensive knowledge to design functional antennas.

The RN-42 is fully certified by the FCC and is a mature and qualified Bluetooth product. It is capable of delivering data at sustained speeds of 1.5 Mbps and burst speeds of up to 3 Mbps in host controller interface modes over a distance of 10 meters. The chip typically retails for approximately 18 dollars USD. Table 9 shows a list of the module's specifications, availble for comparison with the HC-05 module's specification as seen in Table 10.

Table 9: RN-42 Basic Specifications [Tec16]

Specification	**Value**
Class	2
Version	2.1 + EDR
Frequency	2.4 GHz
Sensitivity	-80 dBm
Supply Voltage	3.3 V
SPP Data Rate	240 Kbps (slave), 300 kbps (master)
HCI Data Rate	1.5 (sustained), 3 Mbps (burst)
Transmit Current Draw	30 mA
Power Output	4 dBm
Interfaces Supported	UART, SPI
Modulation Type	GFSK, FHSS

As seen in Table 9, is important to highlight that this chip utilizes Bluetooth 2.1, which is a significant advantage. This version includes the SPP protocol, otherwise known as the Simple Pairing Protocol. This protocol greatly speeds up and simplifies device pairing by reducing the overall number of step required for devices to become connected. It also adds security through the use of pin codes and the ability for devices to inquire information about each other and filter connections before initialization. It also greatly reduces the power consumption required during the discovery and pairing procedures. The SSP is a highly important feature, which is why Bluetooth 2.1 or higher is desired for the project.

Other features of the RN-42 include:

- An integrated Bluetooth stack that does not require a host processor.
- 128-bit encryption for secure connections.
- Low power modes accessible to the programmer.
- Automatic device pairing and discovery for instant access to Bluetooth-ready devices such as Android phones and iPhones.

- Automatic error correction to guarantee the integrity of transmitted data.
- A local and wireless UART configurations with access to both both SPP and HCI modes.
- A USB interface with access to HCI mode.
- Support for human interface device profiles such as controllers, mice, and keyboards.

The main disadvantage to using the RN-42 chip is that lack of support for higher Bluetooth version. Many modern devices utilize Bluetooth 4.0+, which offers numerous performance and security improvements. This is a major drawback, because consumers tend to gravitate towards the latest technological standards. However, this disadvantage can be mitigated because most modern devices are backwards compatible with Bluetooth 2.1 or lower. In addition, by using an older, more mature version of Bluetooth, the developers have a greater understanding of the vulnerabilities and problems associated with it, since most of them have already been observed. It is also a much simpler version of Bluetooth to develop with, since later Bluetooth versions have higher hardware requirements.

4.5.2 HC-05 Bluetooth Module

The HC-05 Bluetooth module is an extremely popular chip that is widely used due to its affordability. For just several dollars, this tiny chip offers an inexpensive solution to hobbyists who need wireless capability in their projects. This chip, like the RN-42, is very tiny and has nearly identical specifications. However, there are some significant differences that must be taken into consideration.

Table 10: HC-05 Basic Specifications [Stu16]

Specification	Value
Class	2
Version	2.0 + EDR
Frequency	2.4 GHz
Sensitivity	-80 dBm
Supply Voltage	3.3 V
Data Rate	3 Mbps
Power Output	4 dBm
Interfaces Supported	UART, USB, I2C
Modulation Type	AFH

As seen in Table 10, the HC-05 module is nearly identical with exception to the AFH modulation type, added support for USB interfacing, and lack of support for the SPI protocol. The advantages of using the HC-05 chip is that it is cheaper than the RN-42

chip by approximately 10 to 15 USD, however, this does not necessarily make it more desirable since the difference is negligible with respect the the overall budget allocated to the development of the smart table.

Both modules utilize optimized integrated antennas which greatly reduce development time and guarantee functionality from a physical standpoint. Using external antennas would result in greater costs and code overhead to establish functionality. The power requirements for these Bluetooth chips are also expected similar, but entirely marginal since the smart table will be connected to a mains voltage line and does not require strict power saving functionality.

A significant disadvantage of using the HC-05 chip is that it lacks Bluetooth 2.1. Since Bluetooth 2.0 lacks the Simple Pairing Protocol, using HC-05 would be frustrating and inconvenient. This aggravating factor alone makes the RN-42 a superior device to use for the wireless communications of the smart table. In addition, there is very little documentation available for the HC-05 chip, which would negatively impact development time.

4.6 Display Implementations

This section explores possible display setups and their associated pros and cons of usage. Additionally explored are solutions and implementation ideas for each discussed method.

Table 11: A Basic Overview of the Types of Displays Considered

Type	**Pro**	**Con**
Multiplexing	Conceptual ease, speed	Cost, size, physical lines
SPI Driven Boards	Conceptual ease, speed, availability	limited device number, cost, physical lines
I2C Driven Boards	Flexibility in protocol	limited device, support, cost due to lack of availability
Custom Boards	Fixes previous issues in other methods	cost
WS2812	Conceptual ease, cost, availability	computational overhead

4.6.1 Multiplexing

Muxing or multiplexing is method which can be used to reduce the total number of lines required to control multiple devices. The method is essentially combining multiple lines into singular lines, i.e. with two lines you can realistically control four lines, four control lines leads into sixteen, and so on. The basic implementation of this idea is achievable through careful usage of a switching network, thus obtaining the amount of control compression needed. Modern discreet packages for such devices are easily obtainable and accessed within a relatively low price bracket, thus implementation cost is low for this method. Though there are considerations that must be made when using this method, such as driving multiple leds at the same time that are connected on the same row or column; depending on implementation only certain arrangements of LEDs can be driving at the same time. Therefore a scanning method akin to the line scanning used in CRT TVs must be implemented in the software controlling the LEDs. As such this also implies that the continually driving and scanning of LEDs must happen to keep the visuals displayed which increases total driver overhead and also increases the complexity of control code. Also the scanning speed has to be taken in consideration, as the array must be driven fast enough such that the refresh rate isn't visible to the eye and such gives visible line flickering which would negatively affect the overall design. The splitting of data and optimization of such data structures and their access cost also must be closely examined due to the nature of having to consider what is essentially a segmented output array to drive, with restrictions given to each piece.

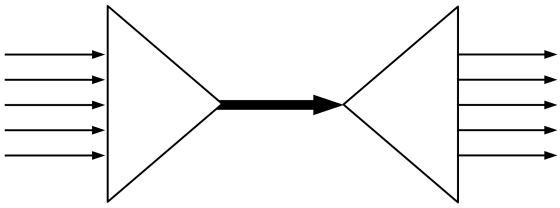


Figure 17: Basic concept of muxing into a demuxer, image under CC[Ano16]

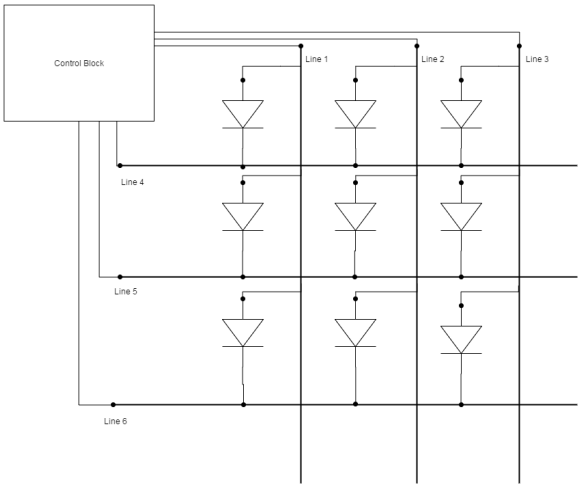


Figure 18: LED Multiplexing Concept, demonstration of using 6 lines to control 9 LEDs

In consideration of driving RGB LEDs and in relation to the Smart Table and using the assumption of driving a 16 x 32 array, the total line cost would come to 48 lines to fully control each individual LED; this assumes a singular color to drive and thus does not represent the total cost for driving and RGB array which increases line cost to 112

lines depending on implementation. Also realistically this is only the cost assuming the microcontroller is directly driving the LEDs which is not possible due to the massive power load this number of LEDs costs; therefore a solution must be considered in which the LEDs are multiplex and yet do not draw current from the microcontroller. The immediate thought is to drive the lines using mosfets and a separate power supply, as this would give an immediate solution to the issue and is intuitively easy to understand. The shortcoming of this method become apparent when the line cost is considered and the power that the mosfets would have to burn using this implementation; therefore a different method to achieve multiplexing is considered.

TI provides multiple signage driving ICs, some of which are designed to drive multiplexed RGB LEDs arrays as the Smart Table is implementing. For consideration the TLC5940 IC was researched as this is a common device to use in this sort of implementation. The advantages it provides is reducing the pin requirements from the control unit, while integrating constant current driving capabilities for the LEDs into one unit. This unit additionally is driven by using a serial interface which is relatively easy to implement and also additional takes processing load off the main unit and thus saving expensive CPU cycles. As such by using a device such as this, additional circuit work can be simplified and additionally the control drivers to be implemented on the microcontroller can be simplified. The main issue with this driver is the amount required, the number required would take up a large amount of physical PCB space and thus drive up costs by a large margin, thus a better solution must be found. Consequently the TLC5958 LED Driver was selected as a replacement part. This IC supports up to 48 channels, and additionally supports 32-Multiplexing natively. This vastly reduces part count and solves the issue of current sinking the lines dynamically. Additionally the device is driven again with a serial interface and has a sort of buffered memory setup allowing faster streaming to offsite storage by the microcontroller which is useful in the limited resource environment that is being used. As such this device vastly improves the potential design by reducing part cost, space compression, and code compression.

Ultimately the core issue with multiplexing a large device such as the Smart Table comes into play with the physical connections. The table itself is physically large, and as thus requires hand production. The production of a multiplexed setup would take at a base level, four lines per wire, massive line runs in which propagation of power issues comes into play, and a sizable preset array to hold each LED in place. The issue additionally with this, is the possible errors in production which would be costly in terms of time to fix and thus makes this method infeasible. To add onto the issues, is the locking the array into a set size is also an issue, as final physical designs can be changed. The cost of testing a prototype is also high, the cost of building a small test setup is quite high due to the cost of coding a one-off design and then the physical board production. Therefore, this style of driving a large LED array did not pass planning phases.

4.6.2 SPI Controlled RGB LEDs

RGB LEDs with SPI driven controllers was investigated for possible implementation early into the project. As these sort of devices can be found commonly, at an affordable price, and can come pre-assembled for easier implementation. These can be found regularly in either dedicated individual boards or in a strip form. Additionally as such, SPI capable hardware is commonly found integrated into microcontrollers as a module, giving this possible control solution a strong weighting to be investigated. There were two main driver ICs examined for this style implementation, both commonly come on a strip interface. The WS2801 and LPD8806 we both looked into for further research. The idea of individual modules were quickly passed over due to runaway costs caused by the physical size of the array to be constructed.

The WS2801 and LPD8806 both essentially accomplish the same idea, which is to take in data from a serial line, and then convert that into driving signals for the LEDs while additionally providing current control circuitry. The basic implementation of control for both ICs is to; select the line to push data over, push data, drop line and move onto next line to drive. Therefore as the Smart Table is planned to implement a 16 x 32 array, sixteen lines would be used overall. This gives a total number of selector lines of sixteen, data lines an addition two or three depending on implementation, therefore worst case a total of nineteen lines to drive the total array of LEDs. Therefore on the microcontroller itself, this would require nineteen pins to properly control this number of strips without implementing a hardware multiplexer to reduce lines. As such this solution is viable from the start in terms of control, as the number of pins is reasonable with the available microcontrollers. Another benefit to this implementation is the data throughput, SPI has a higher data transfer rate and given the added bonus of the driving being done by hardware, maximum clock cycles can be dedicated to the generation of the next image frame. Optimizations due to the functionality of the IC is also possible; for example the LPD8806 supports image holding, or basically the fact the developer does not have to immediately send a refresh frame is there isn't a change that was computed. This allows also for far faster code given intelligent software design. Both ICs are good choices for implementation with some available documentation on both also being available.

SPI driven control was passed on due to a few reasons. The cost of strips, while not incredibly high, is a stumbling point considering the lack of control over the physical design of strips and the output of the LEDs also. Additionally as with strips, the design of the Smart Table must be made with this in mind and thus is a design around the strips instead of the other way around, a severe limitation if an unforeseeable issue crops up during development. Therefore, while a good idea for multiple reasons, ultimately the idea of using SPI controlled LEDs was passed on.

4.6.3 I2C Enabled RGB LEDs

I2C addressed LEDs offer some improvements over SPI control. The most notable is the further reduction of pin count and the ability to immediately address a single LED or section depending on implementation. Two main systems were looked into for I2C implementation, one using a product similar to the BlinkM, or by creating a system based on the PCA9685 IC or similar product. Additionally, another main concern in the physical design of the array is signal degradation due to line length, as the lines would be noticeably long and therefore might have issues caused by outside interference or simple loss of power. As SPI is highly clock dependent and the total signal propagation length is affected directly by clock speed and thus controls the max data throughput, sidestepping this issue was seen as a possible area of improvement.

The main cause of dropping support for this control method is the availability of prefabricated solutions and cost of any already prefabricated ones. Additionally due to the nature of I2C, the most addresses that are accessible is 127 with the modules available; therefore this necessitates the creation of subsections of LEDs to control which would further drive production costs up. This issue could be sidestepped by using a custom protocol mixing SPI's line selection and I2Cs device addressing, but this would drive software complexity up and cost would not drop. Therefore, pure I2C driven devices were passed up.

4.6.4 Custom Boards

A number of custom display implementations were considered in the design phase of the Smart Table. They break down roughly into specialized implementations of the aforementioned display considerations. Some rough ideas considered were an SPI driven microboards in which the display would be made into individual boards with microcontroller upon each one that would act as slaves to a main processing unit. This was decided against due to the sheer number of lines required to control this many devices. Another idea was to take the microboard idea and use I2C this was also voted against due to the 7-bit limitations of most communication modules thus limiting the effective space too much and therefore we would not enough the ability to control the full spectrum of devices. A workaround was proposed for this, which essentially was a combination of SPI's line select and I2C's communication protocol, but this was scrapped due to implementation issues and control. The basic multiplexing and such was also investigated but dropped as the cost overruns would have been even more massive comparative to the original microboard ideas. There were a few other ideas using discreet hardware components investigated but were also scrapped due to cost overruns.

4.6.5 WS2811/WS2812

The WS2811/ WS2812 based RGB LED driver boards offer the most promising features. With a low cost, relatively straightforward control system, they offer an appealing

option in the display design process. The control system for these devices is pretty straightforward as show in 19. As such conceptually the advantages can be seen

Figure 19: Basic High Level Overview of the WS2812 IC

clearly. These devices only require a singular input line for data, and act as a repeater once their internal register is filled. As such this allows for exceedingly single digital control of an virtually unlimited number of LED in a serial control line; the main limiter being the issue of signal propagation delays and the generation of the proper signals in a timed fashion to keep up with the internal requirements of the circuits. As such the main concerns with this implementation are the issues arising from the refresh rate requirements, and power draw. The power draw is a lower concern as the rows can have independent power rails for load balancing issues. The refresh rate necessitates a large amount of constant data generation, a certain rate of data sending, a certain type of control signal additionally. The total computation cost this places upon the system is non-negligible and as such the system must be designed with this in mind and compensation built for it. Though despite this, the ease of operation, and cost makes this the top design pick for implementation in the Smart Table.

4.7 Power Technologies

In order to supply the power for the board, mains power will be the overall input which will be rectified and regulated to some DC values. In this section, the various methods of rectification will be discussed and the positives and negatives of various rectifiers will be investigated. In order for the device to remain reliable and rugged, a proper choice should be made for rectification. If any step of the power supply catastrophically fails, the entire device will be compromised In terms of reliability, please reference paragraph 5.1.7, where a more in depth discussion regarding reliability can be found.

In our application, it is important to keep the necessary current supply to the device in mind. The power consumed by the LED matrix will be significant, and as a result the rectifiers will be working fairly hard in order to supply the LED matrix, the main microcontroller, and the Bluetooth device. Because of this, heat dissipation will become an important consideration in this application.

4.7.1 Linear Regulators

Linear regulators are typically more expensive from a heat dissipation perspective. In our application, we will have a decent amount of current flowing through the regulators as discussed above. As a result, linear regulators tend to heat up, and will require

significant heat sinking on the board. Heat sinking in our case simply will include having a copper plane on both sides of the board connected with vias which will allow the heat to dissipate rather than destroy the rectifier.

Though they are more expensive from a heat dissipation perspective, they are much cheaper in terms of price. Because linear regulators are relatively simple, they are typically cheaper and more simple to produce. This is a consideration towards their application in a final product, however even with the slight savings in cost, we are only using 3 of them in the board in total, and with modern advances in technology both linear regulators and switching regulators are far under a dollar typically. As a result, cost efficiency is not a huge consideration when making a decision between the two parts.

Linear Regulators are also far less efficient than switching regulators. That being said, they do have a slight advantage in a very specific case. If the input and desired output voltage decreases to be very close to each other, the linear regulator becomes more efficient. In fact, typically the efficiency reaches 95+ percent efficient in this case. This characteristic means that the regulator is very efficient in a very niche situation. However, a simple analysis of our design shows that this niche application is not included in the product. The voltage difference between input and output required to observe such high efficiencies is typically much less than one volt. So, one could consider the design implementation that we have included. There are three voltage changes to consider. Immediately following the rectifier, there is a 12V output regulator. The input to this regulator is typically around 18-24V, depending on the particular rectifier. As a result, the difference between input and output over this regulator is anywhere from 6 to 12 volts, which is far to high for a linear regulator to be an efficient solution. Immediately following the 12V regulator is the other two rectifiers. This 12VDC output will be the shared source for the other two regulators. So, the 5V regulator will have a difference in voltage of 7 volts, and the 3.3 volt regulator will have a difference of 8.7 volts. Clearly, neither of these voltage differences is small enough to satisfy the niche case required for an efficient linear regulator. As a result, we will probably be implementing a switching regulator, which will be discussed in the next section.

4.7.2 Switching regulators

Switching regulators are most likely the solution we will implement in the final design in terms of regulating various voltages that we need for DC supplies to the microcontroller, the Bluetooth module, and the LED matrix. As briefly mentioned above, there are three places in the design of the product that require us to have regulators, and we intend on simplifying the bill of materials by using a variable switching regulator. This allows us to buy three of the same regulator rather than three separate parts all with fixed output values. In order to set the variable regulators to the proper value, typically a resistor matrix is used to voltage divide some reference voltage. So then having separate values for resistors allows us to have the same regulator part, and have three separate outputs of our desired values in order to run the board.

Switching regulators have several benefits over the linear regulators discussed above. They are high efficiency in all cases, not just one particular niche case when the input and output voltages are similar in value. Also, the heat production of these parts is very low in comparison. However, this does not become a problem in either case simply because our goal is to design the board in such a way that heat sinking would be present in all locations of regulators, and as a result for our power consumption the heat sinks would be sufficient along with the internal heat sink included with the regulator to dissipate the heat enough so that the temperature would not damage the device, or any other devices int he vicinity of the part.

Though there are several benefits to using switching regulators, there are some issues that arise. However, as mentioned, the problems with linear regulators are far more concerning than the problems associated with the problems tied to switching regulators. Switching regulators are typically more complex to build and manufacture than linear regulators. As a result, they are typically more expensive than a linear regulator of the same output voltage. However, the price difference is not significant enough to warrant any concern. The physical size of the package to go on the board is also typically larger dimensionally, but in the same way that price is not of large concern, board space is also not particularly limited as the mounting of the board has a large amount of space to go. As a result, the concern about dimensions is almost irrelevant. There is one major concern that needs to be taken into account using switching regulators over linear regulators. That is that noise and ripple could be implemented dependent on the switching rate of the switching regulator. This can be neutralized by 10 pico farad capacitors located a the output of the regulator. It is also important to keep the switching regulator output as close to the next input as possible, as this will remove any chance for the trace to work as an antenna and introduce further noise. In the case where the 5V source is supplying the LED matrix, it is typicallly not possible to get the trace to be as short as we desire, so it is expected that ripple and noise will be included through that line. However, we can avoid that through capacitors as mentioned, and it is also important to remember that it is not likely that a small amount of ripple should affect the overall functionality of the product. It will be important to test the product after completion, which is discussed in section 5, in order to ensure that this assumption made about ripple is correct.

4.8 Software Tools

Team collaboration is key for any project, and the tools chosen can greatly reduce possible pain points. The tool selected allow for a large range of interactions and collaborations on the common components. Their is a focus on accountability, flexibility, and simplicity.

4.8.1 Communication

Communication is a key element with any task involving more then one person. Having reliable platforms for messaging, assigning tasks, and sharing information enables accountability and maneuverability during all phases of development.

Slack The tool chosen for general communication is Slack. This tool is a team based web chat service offered by Slack Technologies, Inc. Once a team is created users have to be invited or have an organization specific e-mail. Users are given permissions to create public/private groups, join public groups, notify other users, add new emojis, send direct message to other users, and request to add new slack integrations. Administrators have all permissions users have plus the ability to invite new people, delete user posts, disable accounts, and add/remove slack integrations. This platform promotes users to organically build groups for different tasks, and separate work from play. File-transfers are also included for free with a 5 GB storage limit simplifying the feedback process for quick questions. The supplied integration system gives developers the ability to add notifications for external services, or expose commands to the users. For example integrations with GitHub to see commit notifications, Trello to see task changes, and Google Calender to see new events and notification for upcoming events. All these features supported the decision to use Slack for our day to day communication. [ST16]

Trello Project task planning, assigning, and scheduling can be another paint point for team projects. This can especially be true when people are working in different locations and a physical Scrum/Kanban board can not be used. Trello by Trello, Inc. offers a virtual task board and grants a high level of flexibility with how a team can interact with the boards. Given this amount of flexibility allows the team to decide if a Scrum approach should be take, Kanban, or a combination based on team dynamics. Each person creates their own Trello account, and one user will create a new team. The person who created the new team will need to invite the other members and grant them access to the team boards. All users can create new boards and view existing boards. Boards are made up of lists that can be given any names, e.g. To-do, In Progress, Ready for QA, Blocked, Finished. Each list can have a collection of cards and each card has a short description. The cards can also have people, deadlines, tags, task lists, attachments, comments, and card history. To move a card to another list you can simply drag the card over, and you can always edit cards. Given the characteristic of Trello it filled the project management gap. [Tre16]

Google Drive For more general record keeping Google Drive and Google Docs, by Google, Inc. are used. This project has had the need to evaluate various LEDs, microcontrollers, power regulators, and communication modules. The need to centrally record purchases to monitor budget, parts, sources, and shipping status. Google Sheets granted a clean way to organize all transactions. Taking and storing meeting notes, for future reference, individual accountability, being able to justify time spent on various parts, and tracking problems/solutions. Google Docs made it easy to create

new files for each formal meeting. During research datasheets are found and collected for the components that are evaluated, along with forms, templates, and instructional documentation. These Google products unify the general storage and management for small or slow changing documents. [Goo16]

GitHub For centralized source code GitHub by GitHub, Inc is used. GitHub is a Git hosting provider that offers a centralized place to sync Git branches. GitHub also has a student pack which includes free private repositories while a user is a student.[Git16b] Each repository shows commits, contributers, source, and offers some basic reports. One of the reports is a contributions graph showing how many commits, and a general idea of how much a member has contributed to the project. Being centrally hosted, and free grants this project a powerful tool for source code management. [Git16a]

4.8.2 Development

Taking a project from vision to reality is made simpler with the right tools. Having to build individual parts and combining them into a final product requires that the development cycle promotes collaboration. These ideologies help to determine the tools that will be used to bring this project to completion.

Code Composer Studio Early on the decision to use a Texas Instruments microcontroller was made, and this dictated the choice of integrated development environment (IDE). Code Composer Studios by Texas Instruments is a fork of eclipse by The Eclipse Foundation. Code Composer Studios is purpose tuned to develop and debug applications for Texas Instruments microcontrollers and embedded processors. It includes libraries to simplify the development for supported devices; such as URAT configuration, system clocks, timers, interrupts, and process scheduling. A compiler optimized specifically for these embedded platforms is also included, although restrictions are placed in terms of maximum compiled size, and a soft limit for the industries it can be used for. If these restrictions are hit a precompiled version of GCC is included, and can be used without the size restriction. For Code Composer Studios the free version is meant to be used with the Launchpad series of development boards. This is limitation will reduce the range of microcontrollers that can be considered without purchasing additional licenses. Using the manufactures suggested development environment should reduce the problems that can be experienced, and create a simpler path for porting. [TI16]

Git Working with source code can be a temperamental processes. Allowing multiple people to work on one code base can be challenge. Loosing changes to people overwriting files, having objects renamed, and accounting for who made what change is a nightmare. Modern tools have been created to try and alleviate these problems. One of these tools is Git; a version control system released under the MIT license. Git aims to simplify the head aches of working with a team on development tasks by supplying a solid set of tools to merge, comment, view revisions/blame, and can integrate with other services to further simplify the sharing process. Git operates primarily with branches,

every time a central repository is cloned a new branch is created to represent the remote copy locally. As changes are made they must be committed to the local branch. When the new feature is complete it can be pushed back to the centralized repository. If the central repository has changes that are not present on the local system, they must be pulled to the local machine and merged in before being pushed. [Com16]

Putty Working with serial based communications having a serial client is a must. The development boards used have a USB to Serial interface. The external Bluetooth module uses UART for communication with the microcontroller. This UART signal can be monitored with the development boards USB to Serial interface, a DSA (Digital Signal Analyzer), or a Bus Pirate. Only the Digital Signal Analyzer is a standalone system, but they typically cost $3,000 for an entry level model. The USB to Serial Interface and Bus Priates are either included or relatively cheap solutions. The down side with them is that you need to have software that can access the COM device that they create. This is where Putty comes in, Putty is a terminal emulator that supports multiple communication protocols for example RAW TCP, Telnet, SSH, and Serial. Putty allows you to specify different options for a Serial connection like specify baud rate, data bits, stop bits, parity, and flow control. Putty supplies a clean interface for interacting with Serial devices, and is our preferred terminal emulator. [Tec16] [Tat16]

Eagle PCB design is a required component of this project, and finding a program that is not to expensive and capable can be challenge. Eagle by AutoDesk is a PCB Schematic, layout, and autorouter editor. Students with an EDU e-mail are eligible to receive a free non-commercial Premium license a $820 value per person. [Cad16]

4.8.3 Documentation

Creating documentation with a team and maintaining a unified feel can be hard with out someone working on integrations. Steps were taken early in the project life cycle to reduce the need of manual integration. Early on it was decided to standardize the document folder structure, general writing style, and formating. The documentation is also stored in a GitHub repository, which grants us the ability to audit work done, and it helps with merging changes.

LaTeX Typical document writing is very much what you see is what you get (WYSIWYG), but LaTeX takes a very different approach to document creation. LaTeX has been described as a programming language for documentation writing. The general idea is that you write what you want to say, but you don't worry about formating it as you write. The visual aspect is defined with a set of directives at the beginning of the document. These directives can import plug-ins to grant new functionality, change how sections are formated, and dictate where supporting files will be stored. Document writing is very straight forward, you put **SectionType{Name}** at the beginning to represent different sections. Then you can just write your document like you normally would without thinking about indenting paragraphs, double spacing, or image positioning. Images are added using a figure directive, and you have the option to attach a label for

references, captions, and preferred location. When you want to reference an image you add an autoref directive to your writing and an automatic link will be created with the correct name. LaTeX simply takes all the tedious work of keeping figure numbering correct, and automates it. The Citation system is just as simple. All these features greatly helped drive the decision to use LaTeX. [Zan16]

4.9 Table Implementations

This section explores the issues with the physical table design and the problems inherent in differing scenarios. Then further elaborates upon the other considerations that must be taken due to the project requirements themselves.

4.9.1 Basic Table Design

The Smart Table project is essentially an interactive coffee table interface. As such it follows the expected size constraints of a normal sized table, i.e. dimensions smaller that 4x8 feet while being shorter than 3 feet in height. The additional requirements required for the table design is the integration of dividers for the LEDs and on-top the dividers a diffuser is required for even distribution of light due to LED's inbuilt dispersion pattern. Thus the base design can be cobbled together using a normal glass topped coffee table, the dividers made of possibly MDF board, and a top diffuser made of glass with a coating applied to it or frosted Plexiglas. All parts are easily sourced from local suppliers thus bringing total cost down.

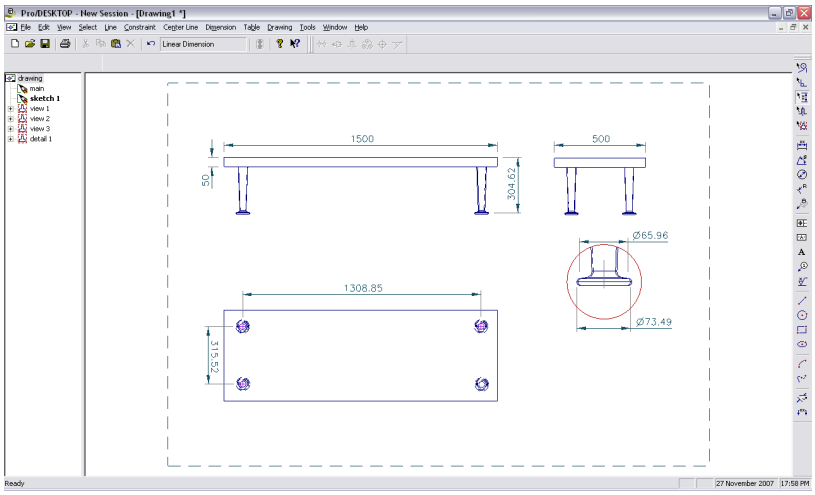


Figure 20: Standard Coffee Table Dimensions, image under CC [Wik]

4.9.2 Flat-Pack Table Design

Flat-pack, ready to assemble, or knock-down furniture is the stylistic choice for the final table design. As such the final product will be laser cut for aesthetic reasons and assembly grounds. Figures 21 and 22 demonstrate the advantages to using a laser cutter as the main production device for the table construction. As laser cutting gives a markedly distinctive edging to the finished pieces and if a white wood is using the final construction is even more distinctive. Also demonstrated in the image set is the other edged sword of laser cutting pieces; the issue posed is the table must be constructed with only flat pieces thus making design and construction into a puzzle. This consequence though is also the main reason for the final decision in production. This stylistic construct is uncommon in normal affairs due to the cost of production and thus allows for a unique element to be added into the final product. As the cost of a laser cutter, time taken, and total wood cost is high for this style, products built using this idea are exceedingly unique and is required for this products final success.

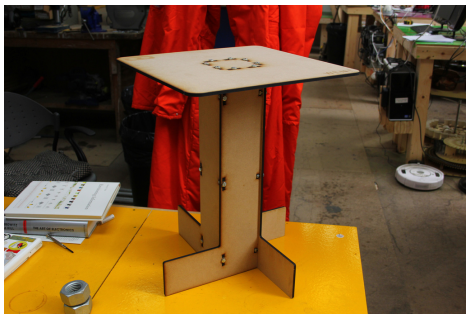


Figure 21: Laser Cut Table, example of fit together furniture, image under CC [oom16]

Figure 22: Laser Cut Table, another example of possible fitting techniques, image under CC [Ale16]

The tables dimensions are determined by the LED array's total size, the LED's light output parameters, and the physical limitations of using wood as the main construction element. Thus concerning the LED's light output considerations, the cells that are to house the LEDs are 1.5 x 1.5 inches and are 1 inch in depth to appropriately output the light within reasonable standards. Thus as the table is to be designed to comprise of an array of 16 x 32 this gives us (n+1) dividers as the cost in one dimension to create the necessary walls for light containment. Therefore, the total number of dividers would be 17 x 33 to have enough partitions to fittingly handle the job of limiting light bleed into neighboring pixels and containing the equipment from outside view and/or natural damage. This gives the total dimensions of the table to be of a bare minimum of 2 x 4 feet excluding the markup cost added by the dividers themselves. Thus as a consideration we have to account for the total size of the dividers and the integration of such into the final design. It is reasonable to assume that if the thinnest easily available wood is 1/10th of an inch and that the final product will not put any undue stress upon the construct, the new size total would become easily 25.7 x 51.3 inches, approximately 2.2 x 4.3 feet. The previously listed numbers all exclude issues with the physical limitations of the material available for usage.

Physical restrictions of wood itself must be addressed as the table itself is a load bearing construct and wood is not a material which acts the same in compression, tension, or along the grain. Due to the choice of using a laser cutting machine for construction, the wood itself is not something we have fine control over in terms of its grain, as we are dictated and controlled by the materials that are available and order-able within reasonable time-frames. Thus the simplest solution is to use plywood which is cheap, easily obtainable, can be found in a white wood such as birch, and solves the grain issue. Plywood is essentially smaller cuts of multiple types of wood or singular types that for each layer of wood the grain is orientated a direction and glued to another layer with its grain rotated 45, 90, or some other degree. This gives the final product a cheaper cost due to wood reuse and lower total quality required in the product; additional this also solves the issue of wood being easier to break along its grain, which once again due to the Smart Table's design is a core issue to be addressed earlier rather than later. This product comes in multiple layers depending on initial material used and thickness.

Noting the earlier commentary on the table construction and design we have to revise the earlier statements concerning total size and cost of materials. The table design chosen for the Smart Table integrates the dividers into the actual table itself and thus is all load bearing and therefore must account for the added stresses of expected load and its own weight. As a consequence this forces the size of the dividers to change into a thickness and material capable to handle such issues. Previously elaboration upon was plywood and thus it is the material to be chosen, in particular three layer birch wood based plywood which is the best material to hit the requirements of structural integrity and aesthetic concerns. The thickness of the material is 1/4th of an inch and therefore using previous definitions gives the table a minimum size of 28.25 x 56.25 inches or rounding approximately 2.5 x 5 feet, and assuming a height of 2 feet and a contiguous area, gives the total amount of wood to be in the range of 400 square feet. Therefore care must be given to optimization of the visual design in relation to the total cost of materials to bring down the total cost of the product without negatively affecting the visual stylistic choice within reason to obtain a viable balance.

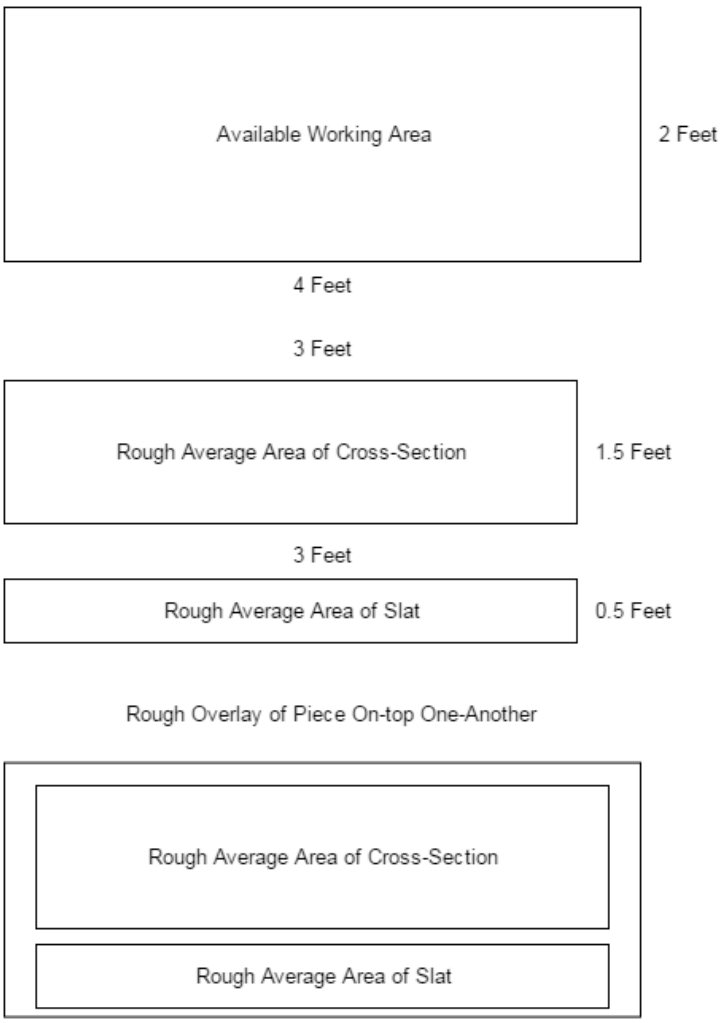


Figure 23: Conceptual Idea of Table from Research Designs

Additionally the construction of such a table necessitates the usage of a laser cutter for production, which is available to the Smart Table design team. As shown in Figure 23 the table design can be worked into the constraints caused by the exclusive usage of a laser cutter for construction of the table.

5 Design

After researching relevant background information and weighing the advantages and disadvantages of several types of components, protocols and implementations for various hardware and software features of the smart table, the design of the hardware and software must be determined with the requirements and specifications in mind. Only the components that best suit the needs of the smart table have been selected, for reasons ranging from affordability and ease of implementation.

This section contains:

- Hardware and software block diagrams that logically depict the function of all hardware and software subsystems and how they work together.
- A detailed look at the specific components selected for the table, such as which specifications are important and how they plan to be used in the smart table.
- Schematics for all of the major subsystems of the smart table, such as the communication subsystem, lighting subsystem, and power subsystem.

5.1 Hardware Design

The hardware design of the smart table table primarily consists of the microcontroller, wireless communications, lighting, power supply, and physical structure of the table itself. Each hardware subsystem must meet the specifications decided earlier. The devices selected during the design process will be examined by their objective, specifications, and functions. Idiosyncrasies of each component will be explained in detail.

5.1.1 Hardware Block Diagram

The hardware block diagram in Figure 24 is a top-level depiction of how devices withing the smart table will interact with each other. The table will run off of 120VAC mains electricity, which will be passed through a transformer. and stepped down to 24VAC. An AC to DC rectifier will convert the voltage to DC at which point it will be stepped down to 12V with a regulator. The LED matrix will receive 5V input using a 5V regulator, and the Bluetooth module and main microcontroller will utilize 3.3V using a regulator. Bluetooth communications will be transmitted to the Bluetooth module from an external device. The Bluetooth module will relay this information to the microcontroller through a UART interface, and the microcontroller will in turn directly manipulate the LED matrix.

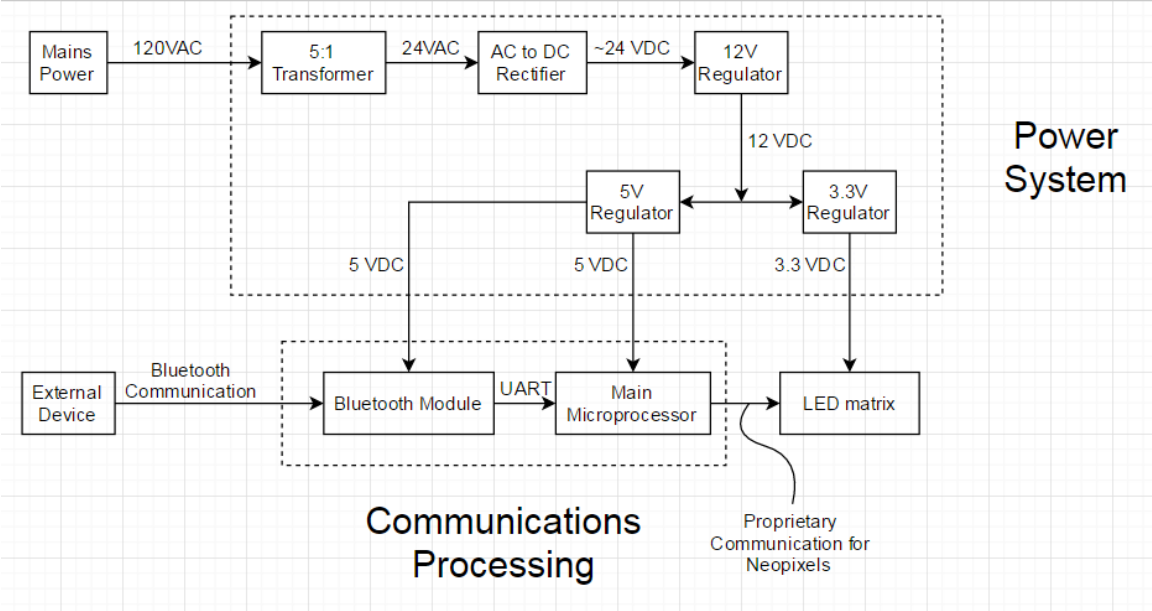


Figure 24: Hardware Block Diagram.

5.1.2 Hardware Design Overview

Following is an explanation of the implementation parts in our project. This list details our use case, but not the specifications of the parts. Please reference 4.2 for a detailed bill of materials with specific details about parts such as price and dimensions. This parts discussion will loosely follow the block diagram above, parts will be discussed in input to output order, where the input is the mains power to the system and the output is the LED matrix.

Transformer This device is intended to run off of mains power, which in the United States is 120V at 60 Hz. In order to rectify this to any usable form, the first step is to step that 120VAC down to 24VAC. This 24VAC will be modified by a switching regulator down the line. The input will be a 120VAC 60 Hz signal, and the output will be 24VAC 60Hz signal.

Part number: FS12-090-C2-B Manufacturer: Triad Magnetics
Price per part: $4.69
Type: Through Hole
Instantaneous quantity available: 2,649

AC to DC Rectifier At this stage, the 24VAC will be rectified to 24VDC by a rectifier. The exact voltage output is not a major concern, as this DC voltage will be immediately regulated down following the rectifier. As long as this voltage output is not lower than the dropout voltage of the 12V regulator, this device will satisfy our specifications.

Part number: MDB6S Manufacturer: Fairchild Semiconductor
Price per part: $0.43
Type: Surface Mount or Through hole
Instantaneous quantity available: 65,291

12v DC regulator This device will take the 24VDC output of the rectifier and regulate it to 12V. This regulator is an intermediate step in lowering the voltage from 24VDC to 5VDC. It is possible to move directly from 24VDC to 5VAC, however this introduces potential for failure as such a step would heat up the regulator to an undesirable factor. Regulators are cheap enough from both a board space perspective and a cost analysis perspective that having an intermediate voltage regulator is not a problem.

Part number:MAX5033CUSA+ Manufacturer: Maxim Integrated
Price per part: $3.68
Type: Surface Mount
Instantaneous quantity available: 2,176

DC regulator This is the second of three regulators, which will provide 5VDC to the microcontroller. The regulator is sourced from the 12VDC from the 12VDC regulator, which is the same source as the 3.3VDC regulator.

Part number:LM2674MX-5.0/NOPB Manufacturer: Texas Instruments
Price per part: $3.28
Type: Surface Mount
Instantaneous quantity available: 15,819

3.3v DC regulator This is the final regulator, which will provide 3.3VDC to the LED matrix. The regulator is sourced from the 12VDC from the 12VDC regulator, which is the same source as the 5VDC regulator.

Part number:LM3940 Manufacturer: Texas Instruments
Price per part: $1.49
Type: Surface Mount
Instantaneous quantity available: 12,652

Microcontroller The microcontroller is a dual core device which will run the code prepared by the software team. This device will drive the LED matrix by providing serial communication down a single line. The microcontroller will also communicate with the Bluetooth module in order to receive input from the user.

Part number:TMS320F28379D Manufacturer: Texas Instruments
Price per part: $32.12644
Type: Surface Mount
Instantaneous quantity available: 15,000

Bluetooth Module In order to receive input from the user, Bluetooth communication from either a phone, tablet, or other device will be implemented. This module will communicate with that foreign device and send user input directly to the microcontroller.

Part number: RN-42 Manufacturer: Microchip Technology
Price per part: $15.27
Type: Surface Mount
Instantaneous quantity available: 3,278

RGB LED 512 of these parts, oriented in a sixteen by eight grid, will be implemented as the devices "screen". They will receive serial communication from the microcontroller, and will be supplied by the output of the 3.3VDC regulator. A non standard serial communication type will be used in order to communicate with the LEDs. The LEDs will all be connected in series, and most of the complexities in terms of making patterns and other such display considerations will be offloaded to the software portion on the microcontroller.

Part number: 1528-1809-ND Manufacturer: Adafruit Industries LLC
Price per part: $7.95
Type: Surface Mount
Instantaneous quantity available: Large amount

Transient Voltage Suppressor The presence of transients from the wall power supply are entirely possible, and are something that the device should be able to protect from. In order to keep any large voltage spikes coming from any source, transient voltage suppressors will be implemented to funnel large voltage spikes directly to ground, keeping the device robust and preventing catastrophic failures.

Part number: SMAJ90A Manufacturer: Micro Commercial Co
Price per part: $0.57
Type: Surface Mount
Instantaneous quantity available: 9,478

Various Passive Components Various Resistors and capacitors will be included throughout the device. The capacitors will mostly be used as bypass capacitors in order to short any AC interference on the lines to ground, in order to protect the microcontrollers. Resistors may be used to set RC time constants.

5.1.3 Microcontroller

The microcontroller is arguably the most important part of the smart table (at least the *smart* part). The microcontroller will have the highest number of tasks of the entire electronic system of the table and process the most code. As the main processor of the smart table, it will directly control the LED matrix as well as interface with the Bluetooth module to receive various remote commands from the user. In this project, a

development board will be used which contains the surface mount microcontroller that will ultimately be utilized in the final project. The development board speeds up initial development and will simplify code debugging.

The main goals for the microcontroller include:

- Send and receive data to and from the Bluetooth module to perform certain functions and report status of the smart table.
- Send signals to the LED matrix to modulate their colors, patterns, and intensity.
- Run subroutines for special LED matrix functions such as displaying time, date, and games.
- Manage the smart table's power modes.

Table 12 shows the specifications of the microcontroller. The microcontroller that was chosen is extremely capable and robust, include a variety of serial communications protocols such as USB, SPI, I2C, uPP, SCI, and McBSP. Although we will ultimately use a custom WS2812 protocol, these features are nice to have for redundancy in the event that a different configuration needs to be implemented to do unforseen circumstances. It is also desirable to have support for a large number of communication protocols in the event that a new feature is going to be added to the smart table that would require one. The dual-core, 32-bit RISC architecture is familiar to the development team and provides enough flexibility and functionality to perform extremely complex calculations require for matrix algebra and manipulation, if utilized. The supply voltage of 3.3 V is compatible with other modules used, such as the Bluetooth module, which means that time and money can be saved on the design of the power system.

Table 12: Specifications of Microcontroller [Ins16]

Specification	**Value**
Architecture	Dual-Core 32-bit RISC
Frequency	200 MHz
RAM	204 KB
ePWM	24 (PWM), 16 (HRPWM)
GPIO	97
ADC	4 (12-bit/16-bit)
SPI	2
I2C	3
USB	1
Supply Voltage	3.3 V

Also shown in Table 12 is the enhanced pulse-width modulation channels are absolutely essentially to control the LED matrix since PWM will allow for adaptive brightness on the LEDs. The intensity of the red, green, and blue LEDs can be individually varied because each LED will have individual duty cycles. This means that a hiugh number of color combinations can be attained, up to 16 million. The PWM feature of the microcontroller will also allow for power savings, since the LEDs will be cycled on and off rather than being powered on continuously.

Table 13: Parts for Microcontroller [Ins16]

Part	Description	Cost
C2000 LaunchPad	C2000 Delfino MCUs F28379D LaunchPad Development Kit	$33.79
TMS320F28379D	Dual-Core Delfino Microcontroller	$0.00

Table 13 shows a list of components that will be used in the development of the smart table. The microcontroller development board, also known as the LaunchPad, is a familiar format that was used during coursework when learning the basics of embedded systems such as microcontrollers. The development board will be used in conjunction with a solderless breadboard to interface with a scaled-down one-dimensional LED matrix and the Bluetooth module, and can easily be programmed and powered via USB. This means that for initial prototype testing and demonstration, a custom power supply is not entirely necessary since the development board includes a powered USB header. This setup allows for easy rewiring and on-the-fly code changes without significant effort.

It could be argued that the smart table has too many GPIO pins for the purposes of the project, however, one must keep in mind the other hardware requirements required to drive hundreds to thousands of pulse width modulated RGB light-emitting diodes. These devices require extremely fast switching, since the human eye can begin to detect flickering and other visual artifacts below a 30 Hz refresh rate, which equate to 30 frames per second. To meet this requirement, a microcontroller capable of operating at high frequencies is needed so that it can maintain a minimum refresh rate under load.

The 200 MHz frequency of the C2000 microcontroller was determined to be sufficient, which gives a lot of leeway when it comes to expanding the LED matrix. The microcontroller not only meets this frequency requirement, but it also has enough RAM to cache frames and manage multiple running programs. The 204 KB of RAM will allow just enough volatile memory to engage in high speed communication via the Bluetooth module and display the various software modules with proper timing. In addition, some software modules will require taxing calculations, such as the video game. Additionally, a potential Fast Fourier Transform demonstration could be used to showcase the power and speed of the C2000 microcontroller.

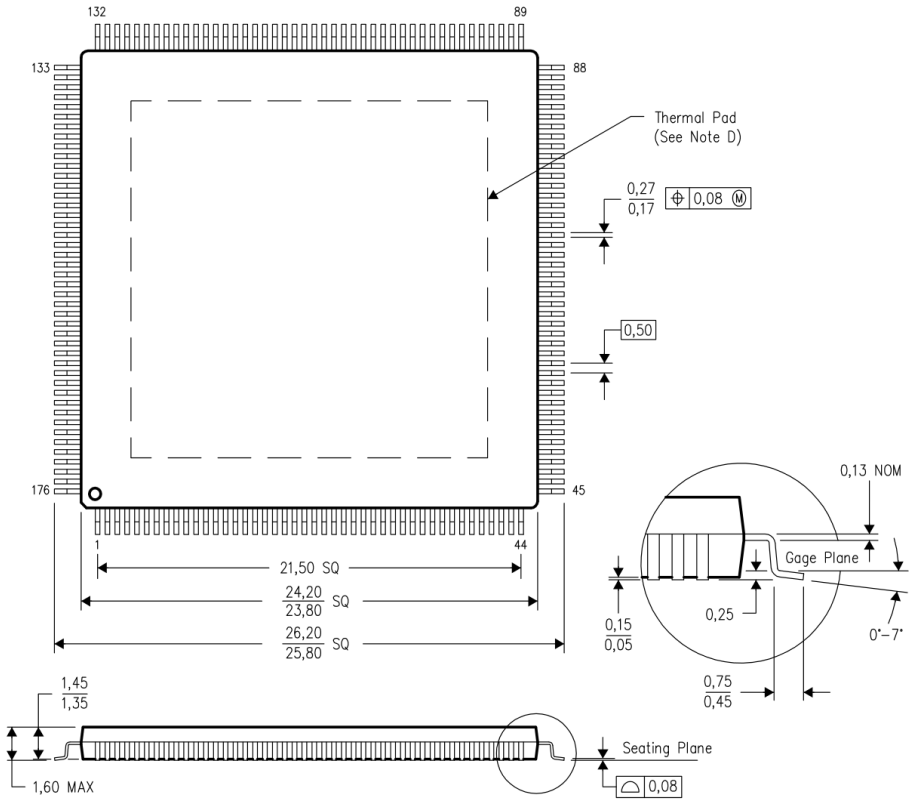


Figure 25: Dimensions of the C2000 Microcontroller. *Permission submitted to reproduce*. [Ins16]

Figure 29 shows the physical dimensions of the microcontroller that will be utilized after the development board prototype. The development board is only intended to be used to develop a working prototype, after which the surface mounted quad-flatpack package, as seen in Figure 29, will be directly soldered to a printed circuit board populated only with the pins necessary for the table to function, along with a flash emulation tool and USB header to program the device. Although quad-flatpack packages are a bit more difficult to work with than dual in-line packages since they are surface mount devices rather than through-hole devices, the microcontroller dimensions are still manageable and can be placed on the board with the help of a third party who can apply solder paste, place it using a pick and place machine, and finally baking them onto the board.

The microcontroller has many redundant features that allow for a high number of inputs and compatibility. Both CPU cores have access to the same features, represented by the yellow (CPU-1) and green (CPU-2) bus lines. More importantly, this microcontroller has many features built in that most budget microcontrollers do not have, such as enhanced PWM channels to create precise waveforms, 12-bit/16-bit analog-to-digital converters, and control law accelerators which doubles the processing power of the chip and enables speedy computation of complex mathematics such as DSP algorithms,

trignonometry and transforms. The JTAG interface is a welcome edition that will enable advanced debugging features provided the proper device programming hardware such as a FET is available.

5.1.4 Bluetooth Module

The Bluetooth module is a critical component that will enable access to the device's core features using wireless communications. With a Bluetooth module, the smart table will be compatible with a staggering number of consumer devices, and will be able to receive commands remotely from the user. The Bluetooth module will essential act as the interpreter between the microcontroller and the user.

The main goals for the Bluetooth module include:

- Receive data such as NIST time, weather updates, and LED settings from an external device through a Bluetooth connection
- Transmit data from the external device to the microcontroller through a UART interface to trigger certain functions and power states
- Maintain compatibility with a wide number of devices using mature Bluetooth 2.1

Table 14 shows the important specifications of the Bluetooth module selected for the smart table. It is important to note that the current draw and power efficiency characteristics were not strong considerations when selecting this module, because the smart table will be plugged directly into the mains electricity and will not need the same level of power optimization that is needed for portable devices. Rather, emphasis was placed on a device that was wide compatibility with existing products, which Bluetooth 2.1 manages to do successfully. Newer standards such as Bluetooth 4.0+ only work with the latest generation of smart devices, which could negatively impact the accessibility of the smart table to its audience.

Table 14: Specifications of Bluetooth Module [Tec16]

Specification	Value
Frequency	2.4 GHz
Supply Voltage	3.3 V
SPP Data Rate	240 Kbps (slave), 300 kbps (master)
HCI Data Rate	1.5 (sustained), 3 Mbps (burst)
Transmit Current Draw	30 mA
Typical Power Output	4 dBm

The fact that the chip uses a 3.3 V supply voltage is important, since the microcontroller also uses the same voltage input. This means that an additional level converter is not

needed to incorporate the module, as the two chips together can be located along the same power bus at a lower cost. A large serial bit stream will be transmitted to the table from an external device such as a smartphone to update information which is displayed on the LED matrix. It has an embedded Bluetooth stack and natively supports SPP profiles.

Table 15 shows why the RN-42 is an extremely popular chip. The tiny module is very incredibly robust and has extensive documentation despite it deceivingly low cost. Although it is not as cheap as certain industrial components, since it is mainly targeted towards hobbyists, the module is still incredibly economical at $18.95. This minimizes the development budget substantially.

Table 15: Parts for Bluetooth Module [Tec16]

Part	Description	Cost
RN-42 Bluetooth Module	Roving Networks Bluetooth 2.1 Class 2 surface mount module with antenna	$18.95

As an off the shelf Bluetooth module, the RN-42 will be readily programmable and easily configurable with the chosen microcontroller over the UART interface. The simplex mode will primarily be utilized for the wireless communications, meaning that data will only be transmitted to the smart table by an external device; there are no intentions to receive data from the table. Radiofrequency serial bitstream will be transmitted to the RN-42 and received by its UART shift register, which will then be transmitted to the microcontroller. The only pins necessary to populate on the chip are the UART_TX, UART_RX, UART_CTS, UART_RTX, GND, and VDD to bias the chip.

The RN-42 has an incredibly small PCB footprint, and has a form factor that is slightly larger than a typical postage stamp. It measures 13.4 x 25.8 x 2.4 mm in its dimensions, maximizing PCB surface area. The Bluetooth module includes a PCB trace antenna that simplifies implementation. A custom antenna does not need to be designed, since the wireless capability is presumed to work out of the box. No additional components are needed to get the chip to operate wirelessly, however, care must be taken when it comes to PCB placement of the surface mounted component.

The antenna must be not be kept in close proximity to other electronic devices or component, as well as neighboring PCB traces or copper ground planes, to eliminate the potential for electromagnetic interference. The clearance for the antenna and suggested placement method is typical for an RF module. The suggested area of clearance is roughly 31 mm around the antenna, for optimal radio performance. The antenna should not be enshrouded in any sort of metal enclosure, which would act as a Faraday cage and effectively block any incoming or outgoing Bluetooth signals.

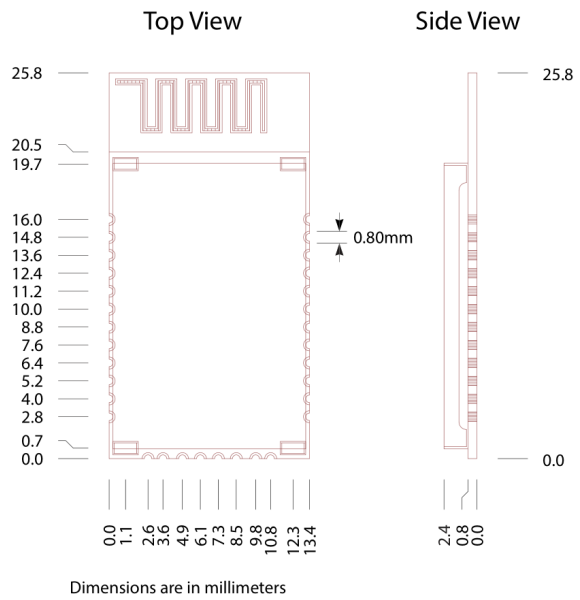


Figure 26: Dimensions of the RN-42 Bluetooth Module. *Permission submitted to reproduce*. [Tec16]

5.1.5 Light-Emitting Diode Matrix

The LED matrix is the central feature of the smart table and will serve as the main display. It is comprised of 512 red, green, and blue LED chips connected in serial. The LED matrix will be arranged in the approximate shape of the smart table. It is intended to be programmable and will display information at the request of the user.

The main goals for the LED matrix include:

- Receiving pulse-width modulated signals from the microcontroller in response to wireless commands.
- Display vivid colors and patterns at a suitable refresh rate.
- Exhibit detailed information such as time, date, and weather.

As seen in Table 16 LED matrix consists of 512 RGB LED chips that utilize a custom protocol called WS2812.The RGB LEDs are integrated into extremely small and circular PCBs in the 5050 form factor. The LED chip includes a built in control circuit containing drivers for signal shaping and amplification. In addition, it contains data latching and an oscillator for precise timing. Constant current ensures that the light produced maintains consistency. They accept a 5 V supply voltage and can transfer data at speeds up to 800 Kbps and consume just 0.3 W. The maximum distance specifies how long the wire connecting LED chips may be before an amplifier must be implemented.

Table 16: Specifications of LED Module [WS16]

Specification	Value
Type	RGB
Color Combinations	16,777,216
Supply Voltage	5 V
Scan Frequency	400 Hz
Power Consumption	0.3 W
Data Rate	800 Kbps
Transmission Delay	300 ns
Maximum Distance	5 m

Table 17 shows that WS2812 LED RGB LED chips are a cost effective solution in the design of the smart table. An important aspect of the LED chips is that they integrate many circuits that are required for typical lighting. Since the smart table utilizes hundred of LEDs, it would be cost prohibitive and time intensive to manually design and fabricate an individual PCB for each LED that contains the proper drivers. The LED chips offer a unique solution that includes all of the required features for the smart table's LED matrix at an affordable price.

Table 17: Parts for LED Matrix [WS16]

Part	Description	Cost	Quantity
LED Chip	WS2812B 5050 RGB LED Chip	$55.20	512

Beyond single data input and data output pins, the chips need only be connected to power and ground. Each because each LED chip has an RGB LED, shown in the center of 27, it is possible to create up to 256 different color combinations. However, since the LEDs are compatible with pulse-width modulation, over 16 million color combinations can be produced per LED chip. This allows extraordinary flexibility when it comes to displaying data on the surface of the smart table. They have impressive visual output, especially considering their cost.

Figure 28 shows three LED chips being cascaded. The WS2812 protocol used by the LED chips is extremely useful because they do not need to be multiplexed. All of the LEDs can simplify be cascaded in series, with one chip's DOUT pin connected to the next chip's DIN pin. The DIN of the first LED receives 24-bit serial data from the microcontroller at up to 800 Kbps, which is then sent to the data latch. The signal is internally passed through the signal reshaping and amplifier circuit, which guarantees that the waveform is not distorted when the signal travels to the next LED chip through the DOUT pin. The presence of the signal reshaping and amplifying circuit means that the signal integrity is not impacted by the number of LED chips cascaded, but rather by the speed of the signal transmission itself.

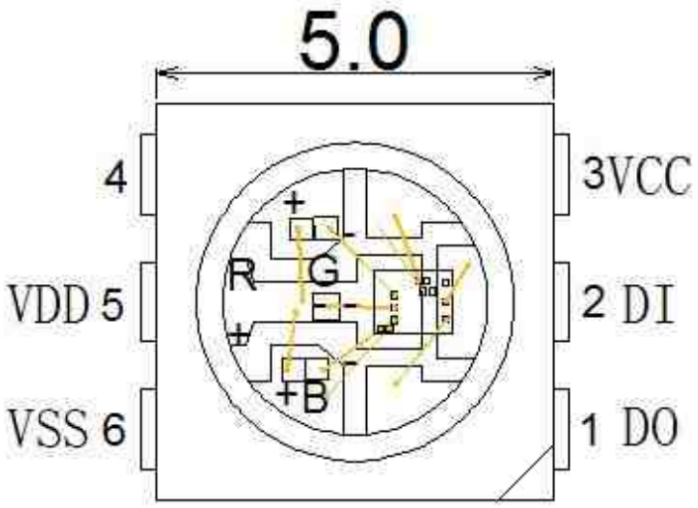


Figure 27: Pinout Diagram of LED Module. *Permission submitted to reproduce*. [WS16]

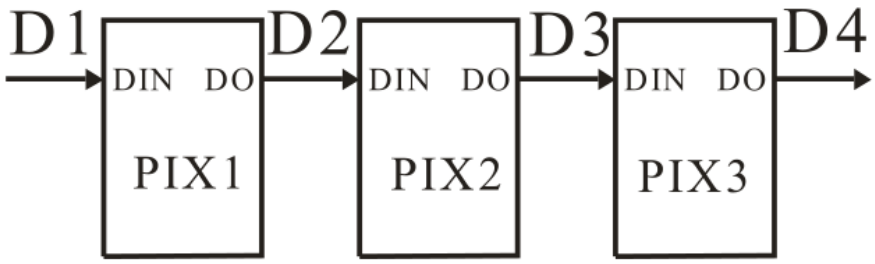


Figure 28: Cascade Diagram of LED Chip. *Permission submitted to reproduce*. [WS16]

The chips themselves are extremely small, a mere 5 mm in diameter as shown in 29. This gives a lot of leeway when spacing the chips out and isolating them from one another in opaque compartments to prevent light bleeding through from neighboring LEDs. Them chip themselves to not need to be mounted to a PCB, and instead will be wired together and reinforced with some sort of strip or ribbon. Since only 512 LEDs are being implemented, an amplifier circuit should not be needed to increase the signal power because the total length of the LED chip strip should not exceed 5 meters.

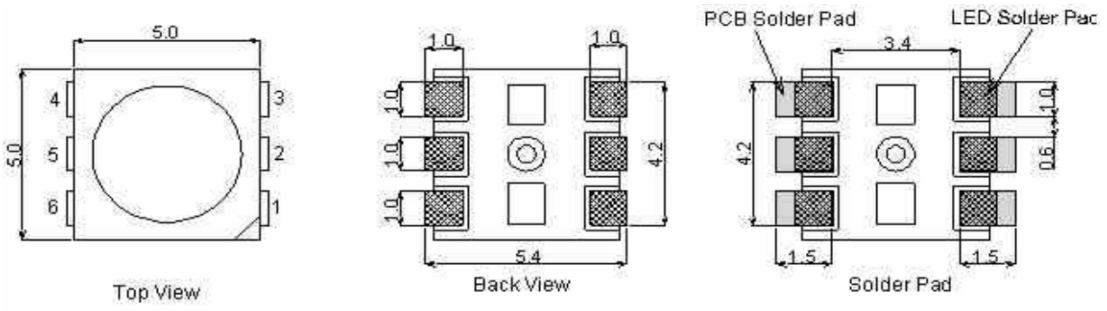


Figure 29: Dimensions of LED Module. *Permission submitted to reproduce*. [WS16]

5.1.6 Table Structure

The previously displayed Figure 30 shows the basic conceptual idea that the table design shall follow. This style was chosen for its distinctive look and ability to be laser cut which simplifies the build process as this device is available for usage. As can be seen in Figure 30 this design is able to be cut-out using a laser and is easily assembled while additionally by the clever usage of cross-connecting slats the design keeps structural soundness. Also notable from the figure as a clever design implementation are the inclusion of stress relief cuts included key areas to help with the stresses cause by wood's natural expansion and contraction.

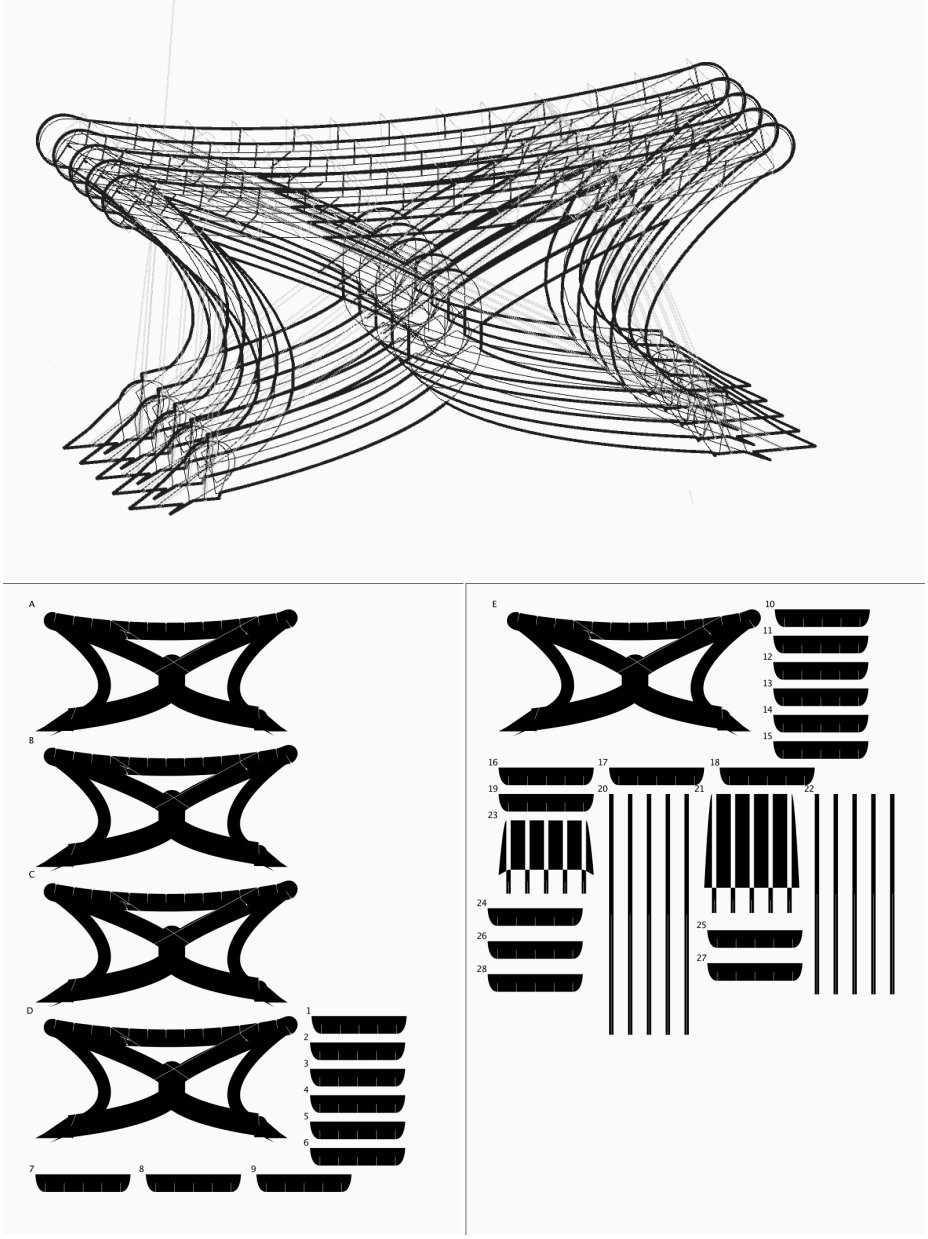


Figure 30: Conceptual Design of Table, Prototype Proof of Concept

As this is a simple proof of concept the main differences between this and the final design are as follows; the number of slats and support structures will mimic the LED display to generate a clever integration of the table itself and the display, the overall shape will be noticeably different and optimized for the least amount of wastage of wood, and finally there will be an included recession upon it for the placement of the diffuser required for the LEDs. The final design will be a more elegant and refined design as such, therefore this figure must be taken with the assumption of the knowledge that it is a proof of concept.

5.1.7 Hardware Schematics

It is important to note that the hardware design has been separated into three separate schematics. This is very important for the readability and organization of the project. If the project schematics are not separated, they would be very difficult to read.

Schematic Creation Procedure In the following section, the procedure that was used to create the final schematic diagrams will be explored. Eagle will be the software used to create the schematics. This software will also be implemented to design the board and produce the GERBERs necessary to produce the final board.

Eagle is a software that is designed to aid in the design of printed circuit boards. This is known as a CAD software, or computer aided design. When designing a board in eagle, the program produces two separate files with .scm (schematic) and .brd (board) extensions. In this paper, the first part of these two extensions will be focused on, the schematic diagram and design. Once the design is finalized, the board file must be prepared and the GERBER files are produced. An in depth discussion on eagle files produced for board production can be found in section 4.

In design, physical parts must be chosen to satisfy the design requirements. Once the parts have been decided on, Eagle libraries must be created in order to use these parts in the schematic file. These libraries are created by referencing the data sheets of the parts we desire to use. Once the libraries are created, the parts can then be used as desired in the schematic. The parts that are using will be listed below, as well as the information provided by DigiKey, a parts supplier. This information includes part number, price, and various other important data points on the parts.

Communication System In this subsection, an in depth discussion of the schematic diagram of the smart table communication system will be discussed. The first thing to notice in the design of this communication section is the level conversion that is produced by the MOSFET circuit in the bottom section of the schematic. This is a very important addition to the circuit for the communication between the portions of the board. The microcontroller and the Bluetooth controller are both communicating on 3.3 volt lines. This 3.3V communication is fine between the Bluetooth module and the microcontroller, because they are sourced from the same 3.3V regulator. As a result, the UART communication will not be affected, and the communication between the two will not damage either device or damage the signal in any way.

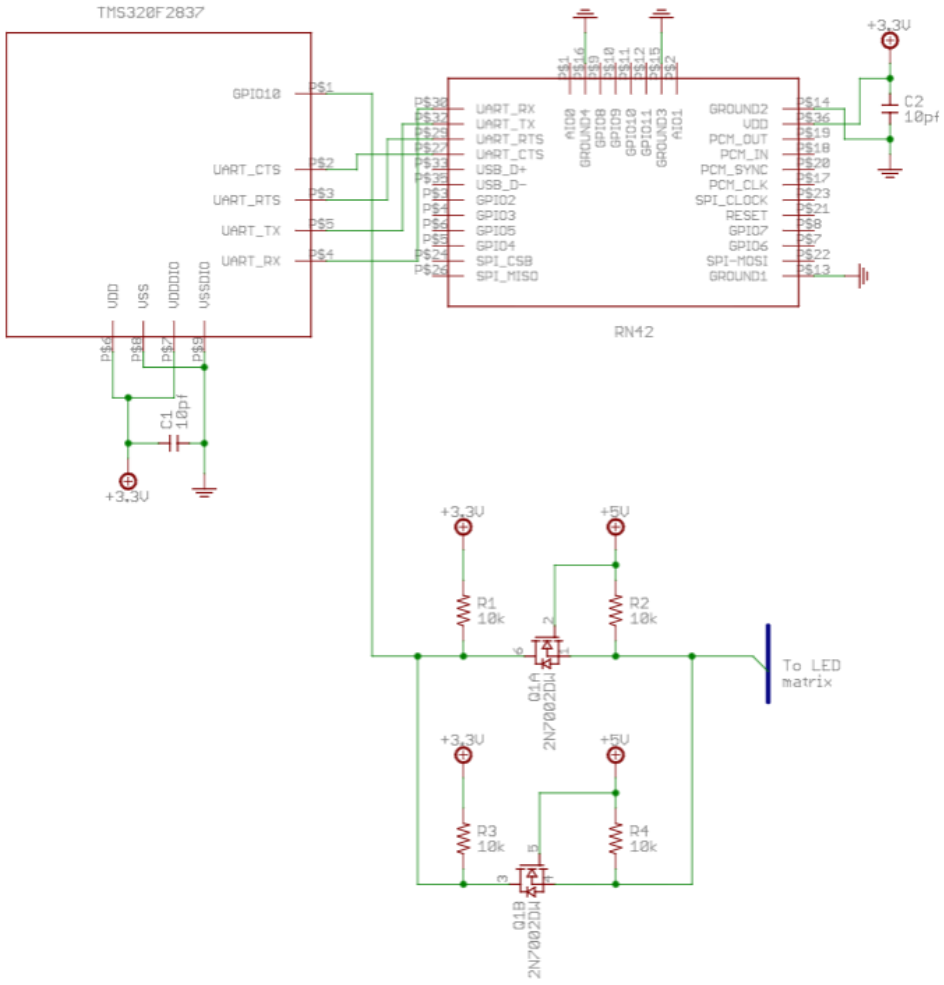


Figure 31: Communication System Schematic

Another thing to notice with regards to this schematic is the microcontroller schematic symbol. This part is a 176 pin microcontroller, which is clearly more pins than this symbol has. Because a large majority of the pins on the microcontroller are not used, they were not included in the schematic symbol for the microcontroller. Only the relevant pins of the micro have been depicted here. These include the UART communication, as well as power and ground, which are separated by a capacitor which serves as a high frequency short in order to filter out any noise from any outside source. Other than that, the schematic is relatively straightforward. The RN45 is displayed in its entirety, as it has a relatively small amount of pins in comparison to the main microcontroller. More detail for these parts can be found below.

Lighting System The LED matrix is sourced by a 5V regulator, and is expecting a 5V signal on the communication line, however, the microcontroller and Bluetooth module are both tied to 3.3 V lines. There is a relatively simple solution to this problem, known

as level conversion. The problem of changing DC digital signals from one voltage to another for different voltage integrated circuits is fairly common problem. Using a mosfet in the configeration seen above, with a voltage source on either side of the mosfet (source and drain sides), allows the conversion of digital voltage. This is a bidirectional procedure, so any 3.3V communication coming in from the microcontroller will be changed to 5V. If the LED matrix could talk back, any 5V communication coming back would be converted live to 3.3V so the micro could read it. However. this will not be occurring as the LED matrix is a slave to the microcontroller with no communicating ability. As mentioned, this level conversion happens live, meaning the communication will not be affected at all by the addition of this mosfet.

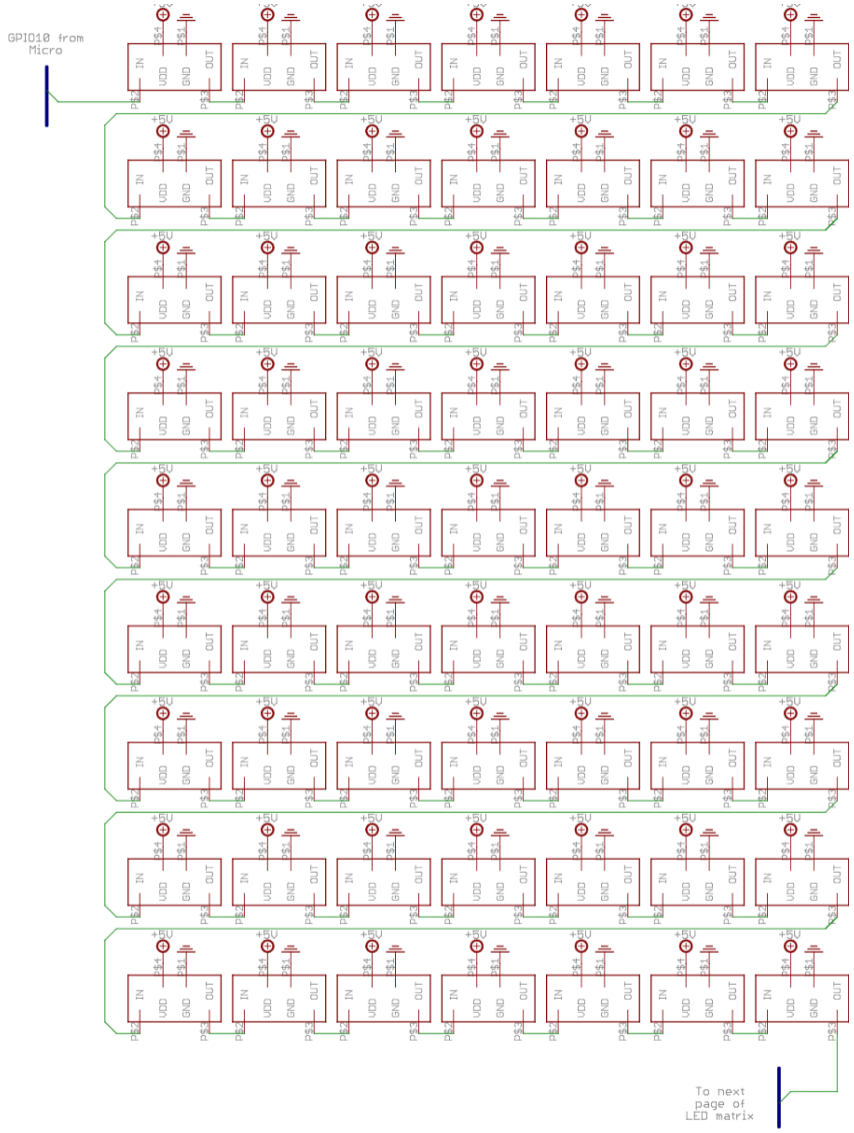


Figure 32: Lighting System Schematic

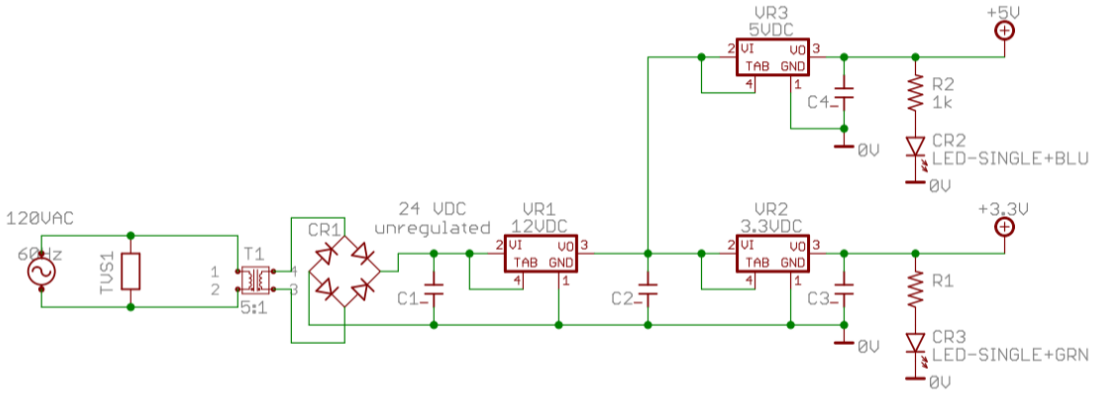


Figure 33: Power System Schematic

Power System The power system on the board has an input of 120VAC and an output of both 5VDC and 3.3VDC, for the microcontrollers and the LEDs respectively. The system is divided into three subsections: transformation, rectification, and regulation.

The transformation portion consists of a 5:1 transformer in order to transform 120VAC to 24VAC. This step prevents a large number of regulators to step the 120V down, if the device simply rectified the 120VAC to 120VDC immediately, it would take a large number of regulators to step that DC voltage down.

The second step is to rectify the 24VAC to DC in order to prepare it for regulation. The output of this rectifier is not entirely relevant, as it will be an input to a switching regulator. As such, the regulator will resolve the DC voltage to some known value. As long as the output to the rectifier is not below 12V (the rated value of the first regulator) + whatever drop out voltage of the regulator is specified (normally 1V) , there will be no problem supplying the regulator circuit.

The third and final step in the power system is the regulation, as mentioned above. There will be three regulators in total, with values of 12V, 5V, and 3.3V. The first regulator immediately following the rectifier will be the 12V regulator. This device will be supplied solely by the output of the rectifier, which will be some DC voltage above 13V. We do not need to know the exact voltage, however it is expected to be around 24-28V, which is more than enough for supplying the 12V regulator. This regulator will then supply the other two regulators, the 5V and the 3.3V. These similarly sourced regulators are the final step in the power supply system on board. The 5V supplies the microcontrollers and the 3.3V supplies the entire LED matrix. It is possible that several

3.3V regulators will be needed in order to supply such a load, but this decision will be made in testing. When two regulators share the same source and the same output, one will act as a load on the other. However, it is possible that this may need to be the final setup if one regulator is not enough to supply the board.

In order to provide a level of reliability to the device, there are several measures of protection inherent in the system. The device is protected in several places by transient voltage suppressors, or TVSs. These devices are commonly used to prevent voltage spikes into an electrical system, and provide a simple and cheap degree of insurance. Typically impliemnted as a Schottky diode oriented in a reverse bias from the expected source of the transient.

Bypass capacitors will also be used in strategic places in order to prevent any AC voltage interference picked up on the power lines from disrupting the communication of the microcontrollers. These capacitors will be relatively small (.01 uF), and will be coupled very closely to the pins of the microcontroller supply pins.

5.2 Software Design

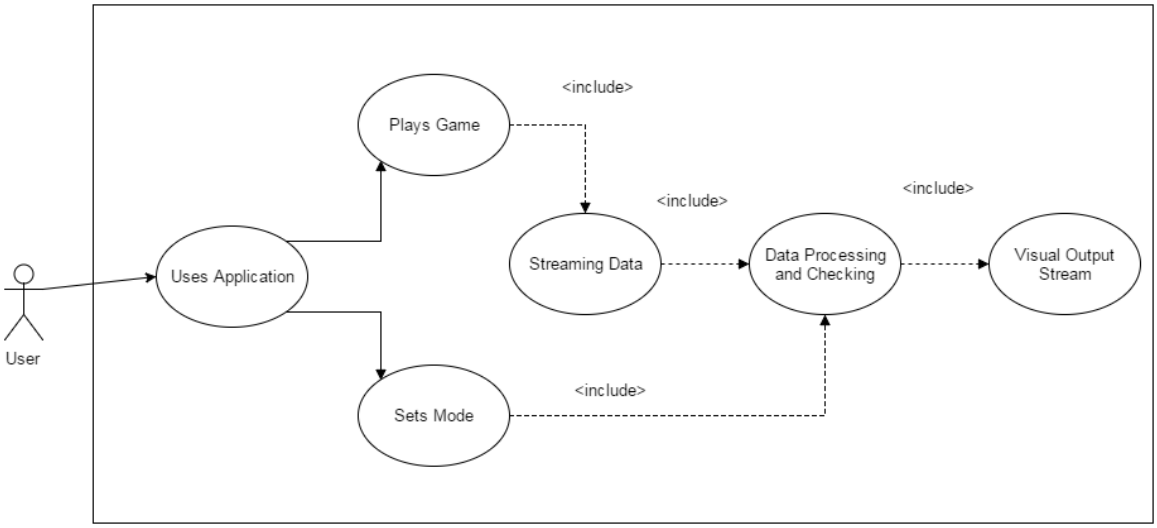


Figure 34: Basic Use Case of Software

Figure 34 is the use case diagram for the basic software interface of the Smart Table. It demonstrates the main to states the external application has and its accompanying effects on the tables control loop.

5.2.1 Software Block Diagram

A object oriented style of distinct modules will be incorporated into the high level design of the software which will ease the issues with transferring code and concurrent access.The high level overview of this concept for the Smart Table is shown in Figure 35. Also additionally the clean distinction between modules can be observed from

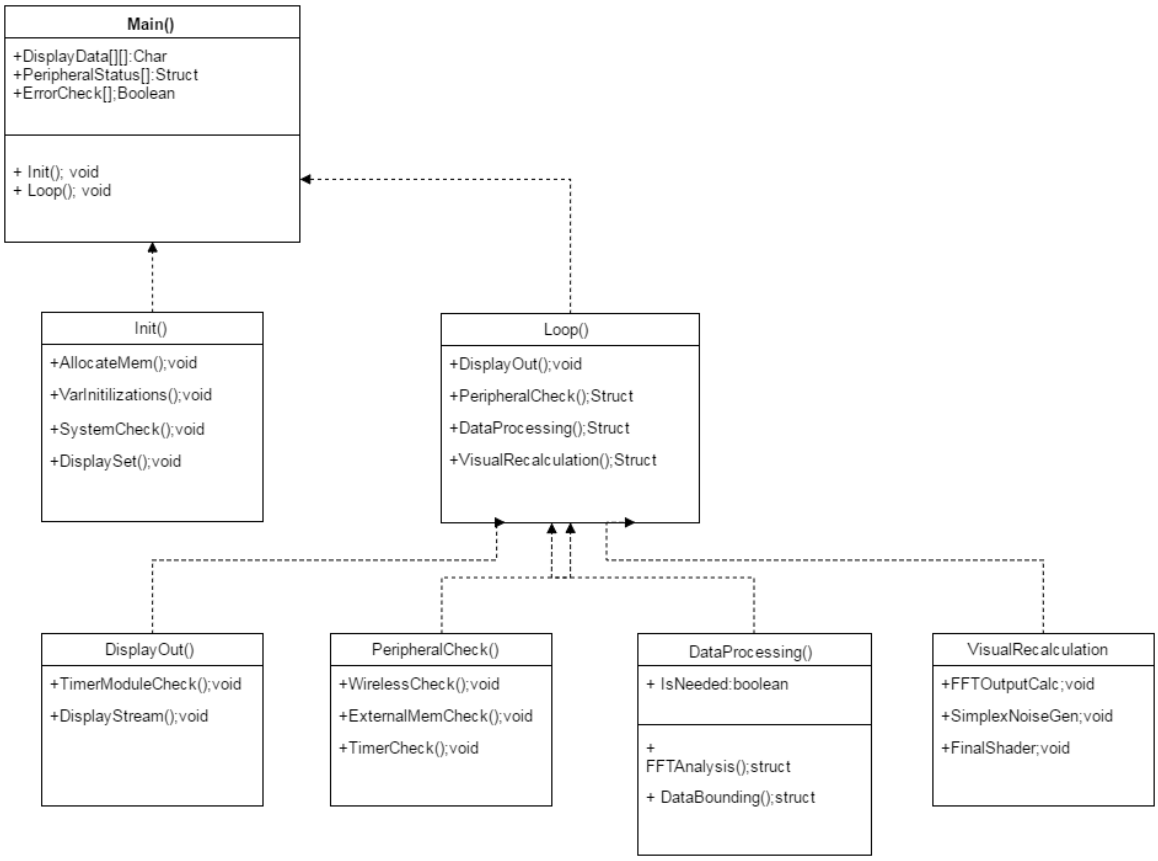


Figure 35: Class Diagram of High Level Design of Software

the high level design thus it can be inferred the project will take advantage of modern project designs arising from multiple headers and multilevel file designs.

5.2.2 Display Output Driving

The display driver is a key point in the software design as it is a timer sensitive and computationally expensive process. The key points of the high level design shown in Figure 36 are the usage of the inbuilt timer modules to control the software flow at key points and to reduce the possibly of timing errors arising from other methods such as cycle counting or offloading the process onto and external module like the SPI communication interface. As such the system shall implement a timing system in which the graphics unit pushes data onto a buffer that additionally is a buffer for the actual output data that is concurrently modified. This will be done with the implementation of interrupts from the timing module and as such is a cycle independent system and is easily modified and expanded upon. The implementation of the dual buffer system allows for seamless transitions and additionally helps with possible race conditions cause by the dual core, dual cache, dual buffer system.

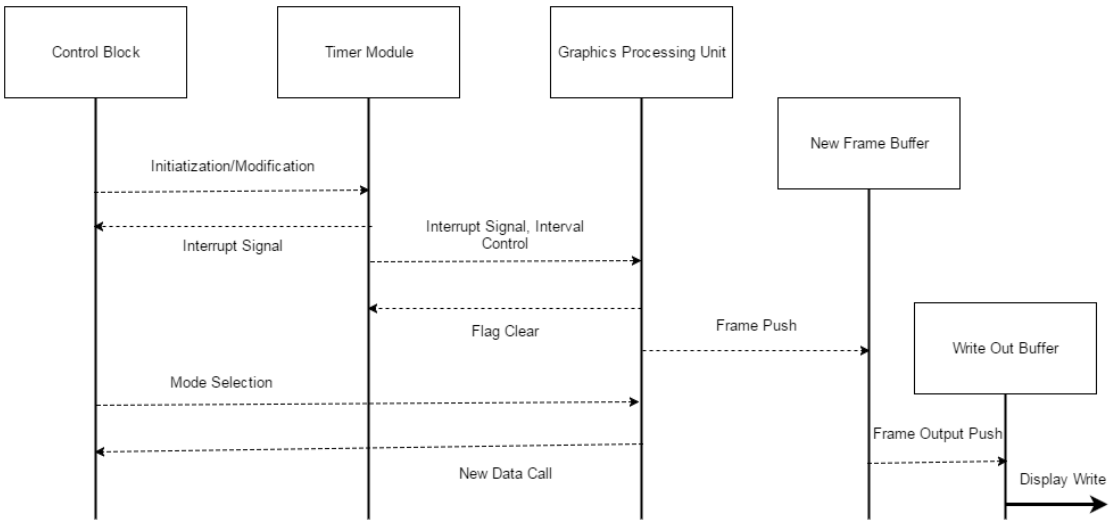


Figure 36: Control Flow of the Graphical Display Module

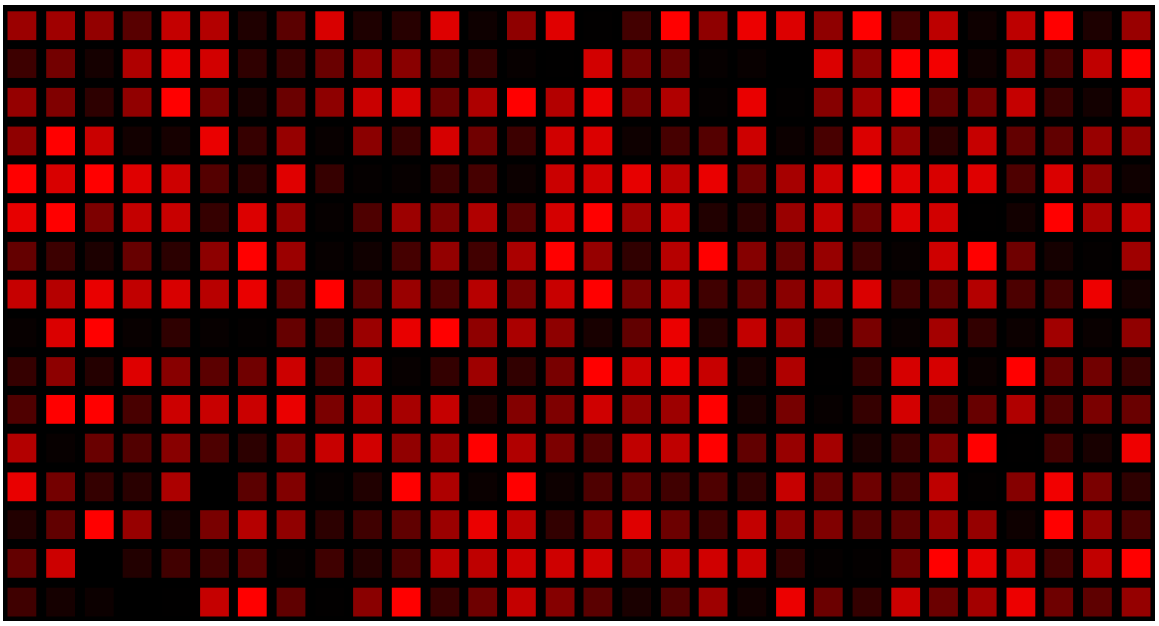


Figure 37: Simulated Output of Smart Table with additional Simple Noise Overlay

5.2.3 Simplex Noise Generation

Implemented within the graphics processing control block is a Simplex Noise shader overlay that is overlaid onto certain algorithms. Simplex noise is a method devising for the generation of n-dimensional noise that is similar to Perlin noise. The main differences between these two methods is the visual look, fewer artifacts, and a lower computational overhead. This method is a procedural function that is useful for adding visual flair to images and a sense of movement depending on implementations. As such this method is vital to the Smart Table implementation as a visual display shader to add additional value to the product. Displayed in Figure 37

5.2.4 Procedural Image Generation

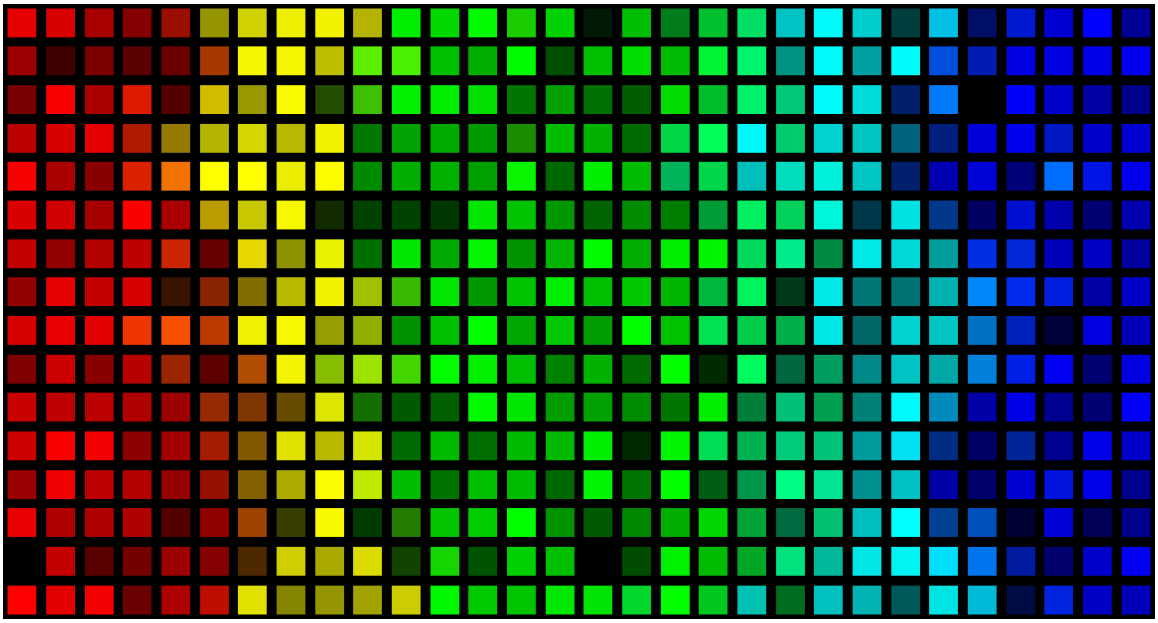


Figure 38: Simulated Output of Smart Table using Procedural Generation

Implemented within the graphics processing control block are methods used for the procedural generation of images. Shown in Figure 38 is a simulated output of the methods used in the image generation upon the Smart Table. The images shall be implemented using procedural methods due to the ease of scaling, memory concerns, and the visual interest that can be easily added by usage of such methods. As such various methods such as the implementation of shaders and blending shall be used in the procedural image generation code blocks and shall be optimized to the highest amount possible due to the computation cost of such methods.

5.2.5 Microcontroller Software Interaction

The information and software flow shown in Figure 39 demonstrates parts of the design that must be considered in implementation of the device as it is a dual-core device which entails a number of interesting technical challenges to overcome. As such the concept of threading in the real-time system must be considered and a cpu scheduling additionally must be implemented for proper system and software control. As such the careful design of shared memory interfaces and consideration of race conditions will be key in the successful design of the finished Smart Table project and the operation in a clean way.

Time and Date The time and date functionally requires the implementation of an external device to deliver the information needed for the function as shown in Figure 40. The implementation of an "on-board" module to obtain the information was considered by the cost and complexity of such a module forced the team to abandon this pursuit.

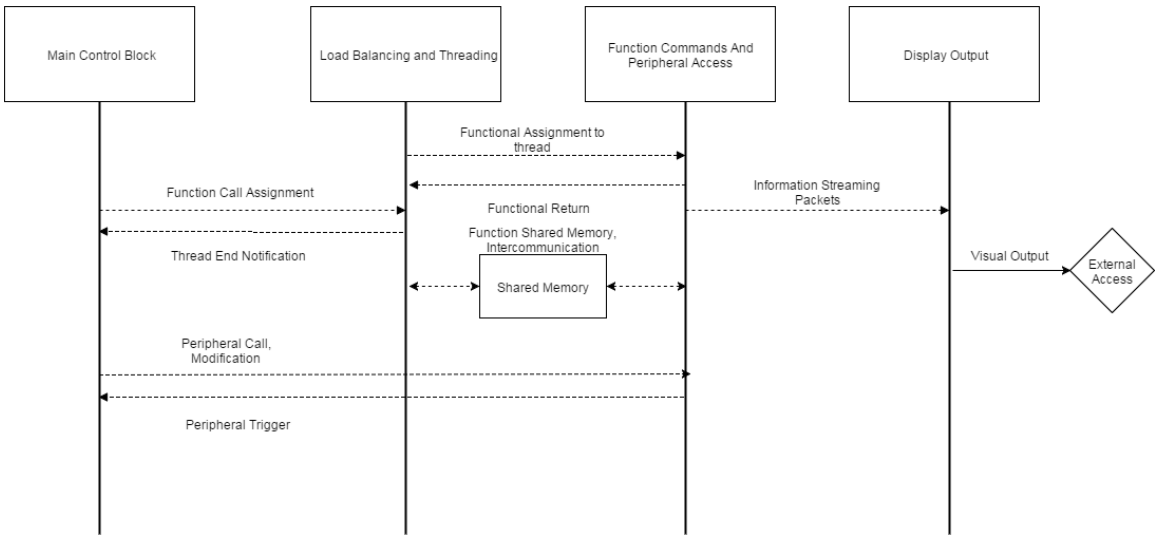


Figure 39: Information Transfer Diagram and Control Between Modules

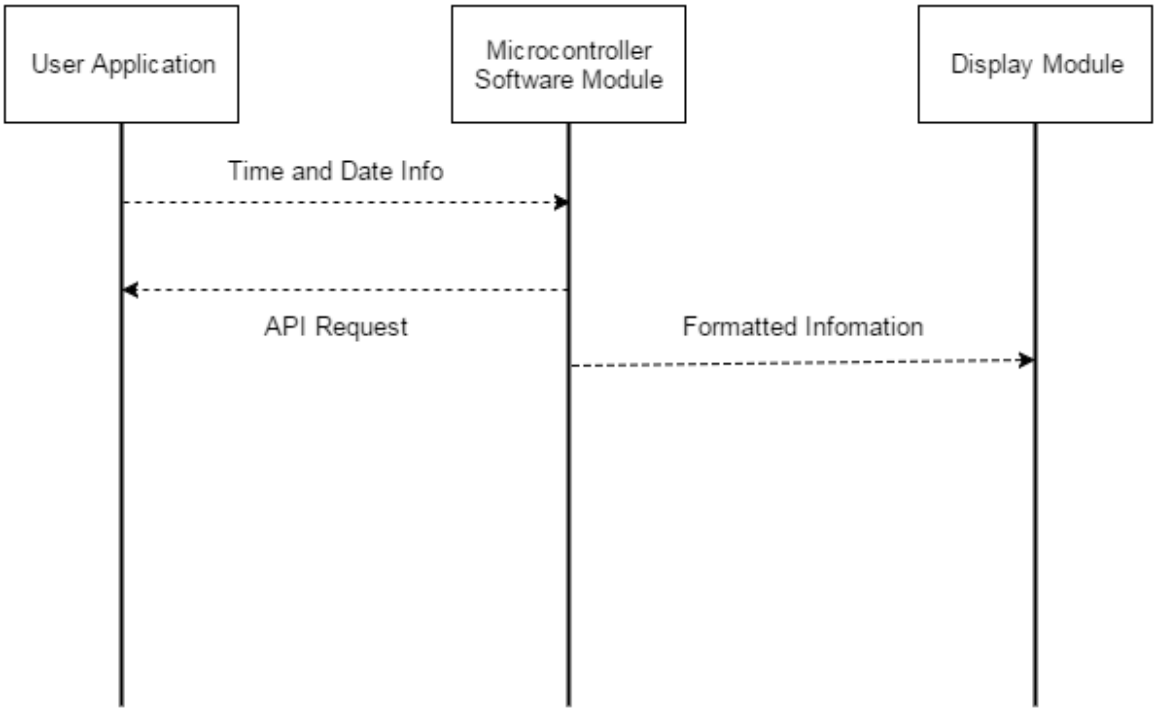


Figure 40: Time and Date Software High Level Design

As such a pseudo client-server architecture for this add-on feature was decided as it could be implemented in the user facing software implementation. The core idea of this implementation is that the user application will pull the local device data of the time and date and send that to the microcontroller and thus once received the Smart Table will process and generate the required visual output image in response to the push request.

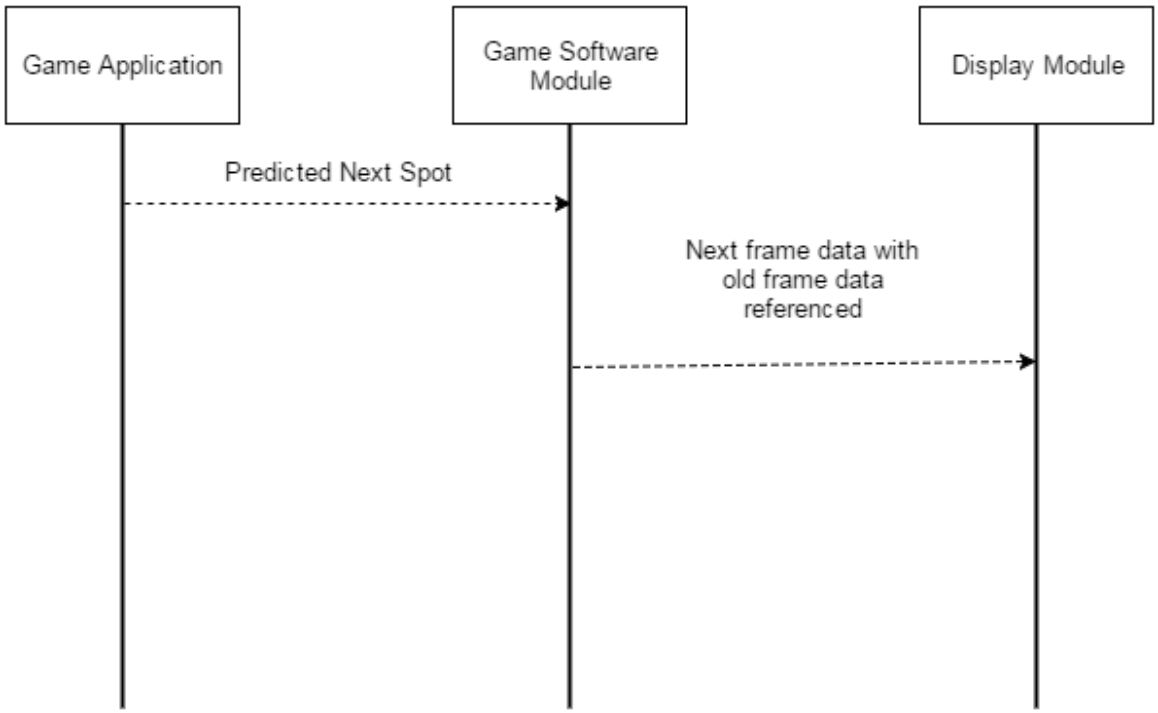


Figure 41: High Level Design for Control of Snake-Like Game

Snake-Like Game The snake like game requires a client server design as well. Due to cost overruns and design constraints of implementing a controller into the table or by using third party hardware, the decision to turn the user's device into a control system was decided upon. As such the user's device will act as a controller and generate the interface upon it for visual feedback on the game, while concurrently the user can also look at the Smart Table as the visual output of the snake game will be mirrored upon it, this is shown in Figure 41. As such the main data that will be sent to the table is position data of the snake and any object data to be rendered, thus keeping total transmission costs down and partially offloading the heavy data generation.

Color Controls The color controls shall essentially act as a shader function on-top the basic pattern functions that will be generated. As such the color controls will be partially implemented on both devices in a synergistic configuration to allow for minimal latency and data transfer as demonstrated in Figure 42. In effect they will essentially be a mode selection option overlapped with the normal operations.

5.2.6 User Interface

The user interface shall be a bare bones application developed with ease of use and optimization in mind as shown in Figure 43. As the total throughput to the microelectronic is low, the optimization of packets must be controlled carefully through the application of dynamic programming ideals.

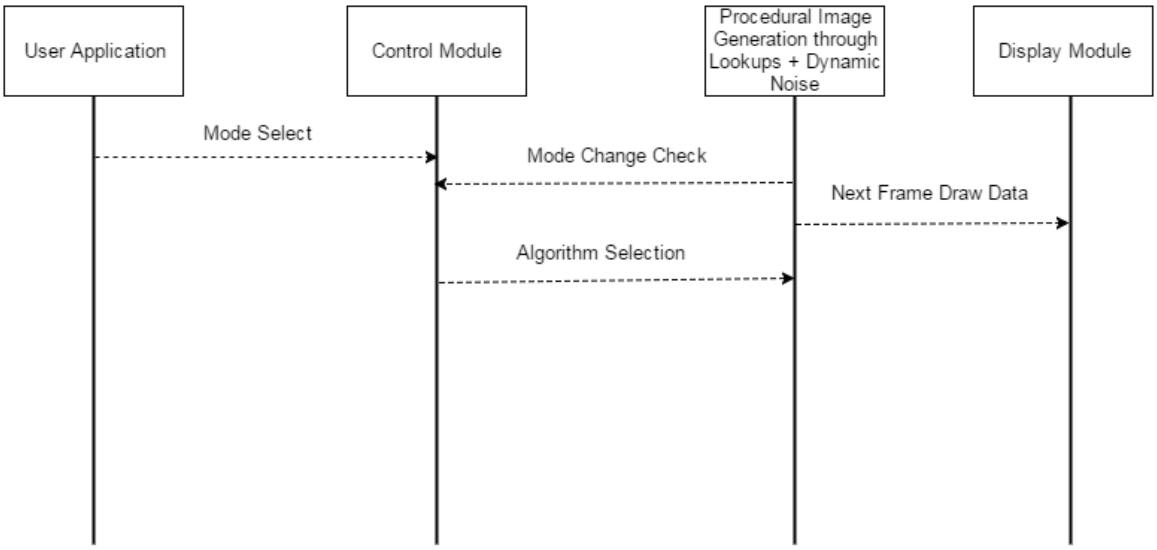


Figure 42: Information Transfer Diagram and Control Between Color Modules

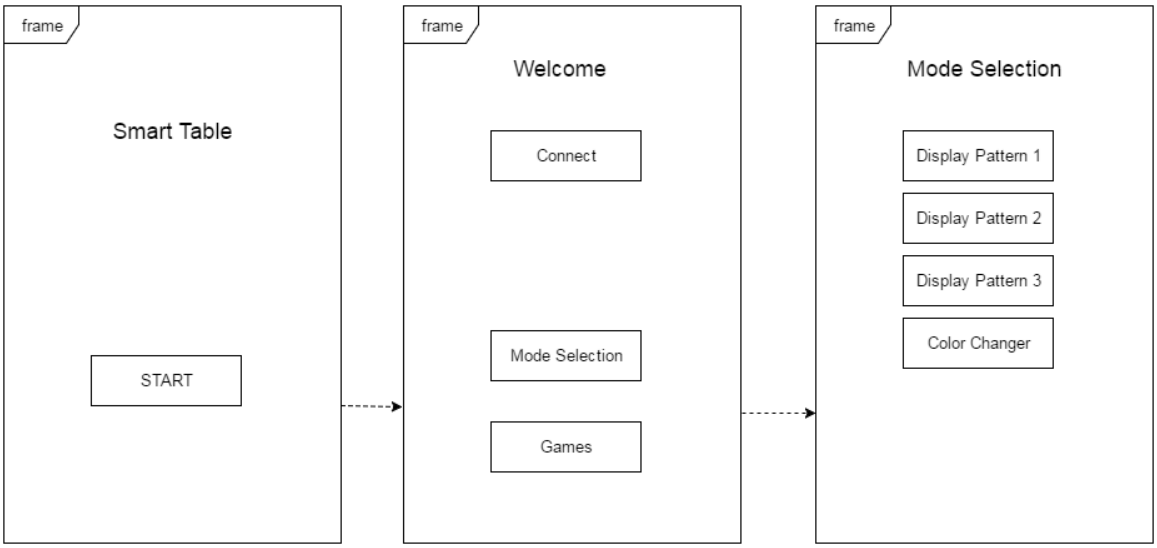


Figure 43: Example of User Interface for User Facing Application

5.3 Design Summary

In essence the design of the total software system will be object oriented optimized design that focus on key ideals of dynamic programming and various streamlining techniques to an nonpareil environment for the user. As such the total computational cost must be kept in mind during the software implementation phase and thus will be carefully calculated to keep costs down and user interaction speeds high.

6 Prototype Fabrication

Now that the major design of the smart table is complete, it will now be evaluated in the bounds of a physical circuit board. The feasibility of the build will be discussed and the particular details of the build will be explored. Particularly, the output of the Eagle software will be explained in detail, and the source of the parts will be detailed.

This section contains:

- Images of a working prototype of the smart table.
- A procedure for designing a printed circuit board using the Eagle layout software
- A discussion of how the PCB will be fabricated after is has been designed
- A detailed bill of materials depicting all of the parts chosen for the smart table including quantities and costs.
- Expectations of the prototype, such as potential issues with both the hardware and software during construction and development.

As shown in Figure 44 the key components of the Smart Table have been purchased and production and testing begun upon the design. In the picture shown the main parts of the microcontroller, RGB LEDs, Bluetooth communications, and the power conversation circuitry are shown which compose the bulk of the table's main functionality. Additionally the rest of the components for the board production themselves have been purchased in addition to the RGB LEDs required for the table's visual output

6.1 Printed Circuit Board

The finished schematic and subsequent board layout (which will be completed within the first few weeks of senior design two, or possibly over the winter break period) will need to be realized in the form of a printed circuit board. This board fabrication is a fairly expensive step in the procedure, as the intention for our group is to have board fabrication done entirely professionally. This is simply due to the fact that any mistake at this stage could be incredibly costly in terms of materials, and more importantly in terms of time consumption. Diagnosing problems with a board are typically incredibly time consuming procedures, and are unacceptable at such a late stage in the project. The plan for fabrication are detailed in this section.

6.1.1 PCB Design

As discussed in section 3, Eagle is the primary vehicle of design, and both the schematic and the board layout will be completed in this software. Once the board layout is complete, Eagle has a built in method to export a certain number of files, called GERBERs,

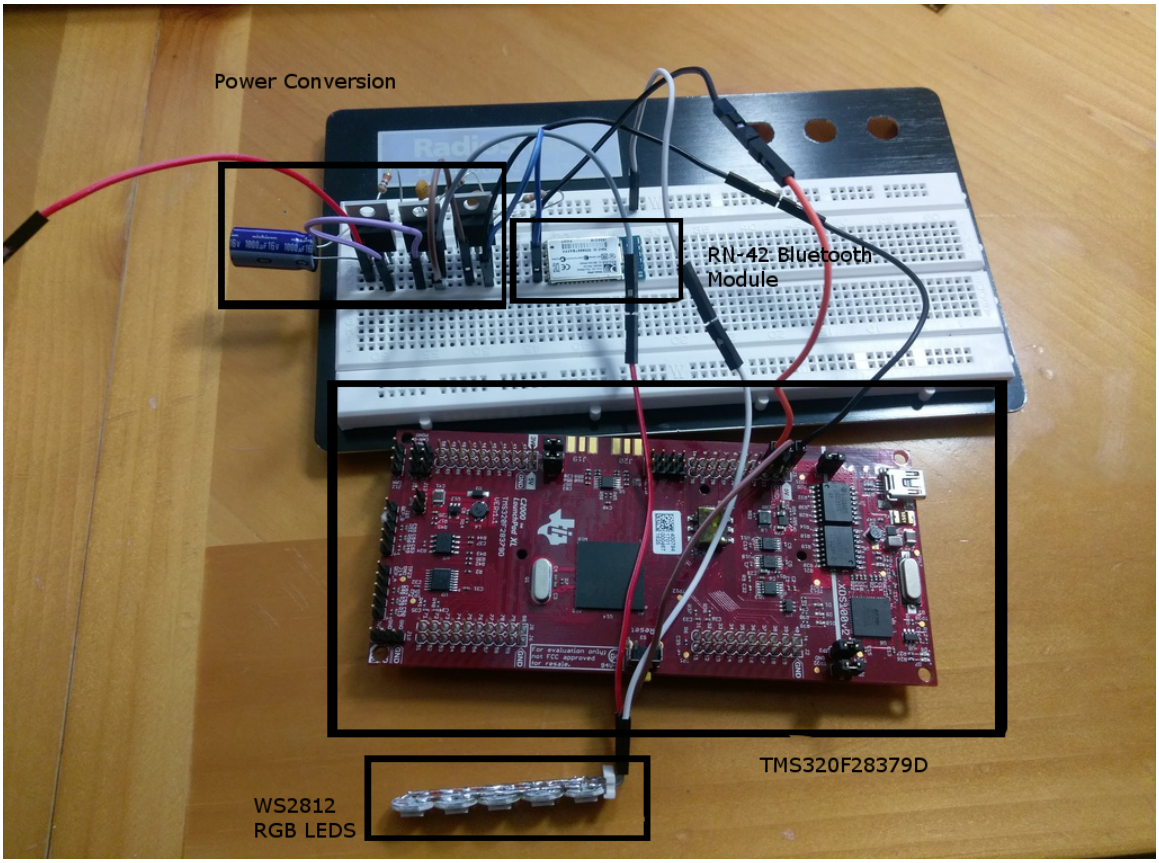


Figure 44: Smart Table Prototype.

which lay out certain specifications for board manufacturers in order for the fabrication of a printed circuit board. These files are in a human readable ASCII format, which simply lay out where a certain feature needs to be. We will use the RS-274X family of files for our project, which is considered the standard to describe any given PCB layer. The specific GERBER files that we plan on using are listed below:

Copper Top and Bottom (.cmp and .sol) These two files lay out the copper features for the board, otherwise known as the traces on the board. These layers correspond to the "top" and "bottom" layers in eagle, and are typically simply used for conductive paths or planes.

Top Soldermask and Bottom Soldermask (.stc and .sts) The soldermask on the top and bottom of the board is a thing layer of polymer applied to the exposed copper of traces on the board, in order to prevent both oxidation and to prevent unintentional shorts due to soldering. This is especially relevant in small pitch devices, such as our microcontroller and the Bluetooth module. In both of these applications, the solder mask is incredibly important to be applied correctly. Typically there is no soldermask on the bottom of the board unless there are parts there, as is our case. The intention at this point is to keep all of the parts on one side of the board for simplicity.

Top Silkscreen and Bottom Silkscreen (.plc and .pls) When the board is created in Eagle, it is important to have labels for each part so that when the bill of materials is created, each part can be linked to the board design. These references are typically called "reference designators", examples of such designators are "R1, R2, C1, C2" for two resistors and two capacitors. These will number off in order until every part on the board has been listed. These also exists so that any individual working on the board has a method with which to eliminate confusion if any parts need to be replaced or re-soldered. The soldermask itself is the ink located on the board, and is typically printed in a color very clearly different than the board color for readability A common convention in board design is white silkscreen on a green board, which is likely to be the design we use.

Drill Legend (.drd) The board will have various drills located throughout in order for various applications. The most common, and the likely applications in our use case, include mounting holes and vias. In order to physically mount the printed circuit board to the table. The vias are small features on the board (typically around .012 inches) that allow a copper trace to jump from the top of the board to the bottom of the board. This is a specific case, in general a board can have any number of layers, including internal layers for extremely complex boards, which our board is not one. However, in these arbitrarily complex boards, a via can connect any coper layer, whether it be a trace or a plane, to any other layer's trace or plane. This is useful simply because board design will typically grow complicated enough that traces will need to duck under each other, or connect to some ground plane on the back of the board. It is imperative that using vias not be overused however, because vias are less reliable than simply having a trace from one point to another, as they are more exposed to corrosion and other such issues. Please reference section 3, design, for a more detailed discussion on this topic.

6.1.2 PCB Fabrication

Contracting Once the aforementioned GERBER files have been produced by Eagle, two separate foreign entities need to be contracted. In order to produce the final printed circuit board, a stencil will be created to properly distribute the solder paste onto the board, and the actual board itself will need to be created by an outside source as discussed above. The board itself will likely be outsourced to a company called Alberta Printed Circuits, or APC. We expect the price of this procedure to cost around $100 or less. Preparing the board through such a professional channel will streamline this procedure, which is a major benefit towards using this method.

Stenciling The stencil is also a relatively simple procedure, however this is a majority of the cost for the project in the long term. The total price of a professionally created stencil is around $220. The stencil simply is a thin metal sheet, which has various small apertures for the solder paste to run through in another step in the manufacturing of the board, discussed later in this section. The thin sheet is framed in a thick metal box, which is used to stabilize the entire structure once fixed in the solder distribution

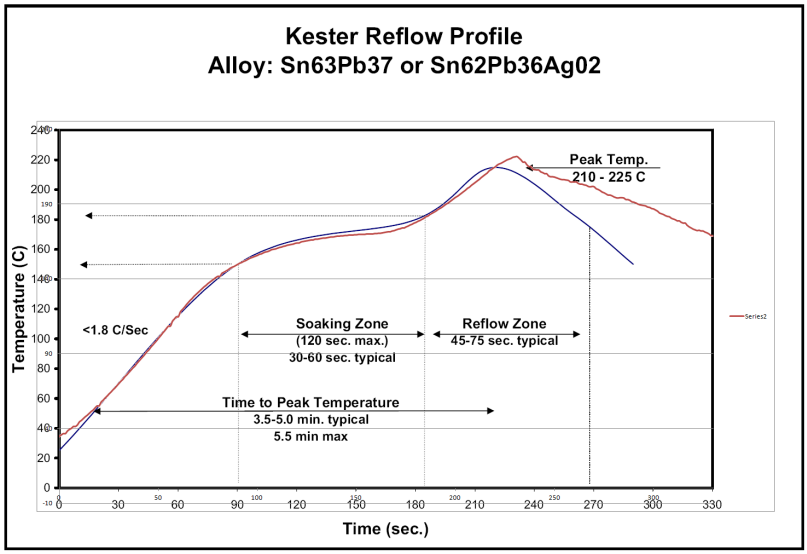
machine. The total cost is typically material cost in this application, though keeping the tolerances of the apertures is also a cost consideration, as with our microcontroller and Bluetooth modules that have small pitch, the cost for keeping the apertures tight to tolerance goes up. These two steps, stencil and printed circuit board manufacturing, are both performed off site by professional companies. Once these steps have been completed, the results of these steps will be brought to the company that Ryan Mulvaney works for, called I-Con. This company has allowed the use of their solder paste distribution machine, their pick and place machine, and their oven. The three stages will be discussed briefly below.

Solder Pasting The Solder paste distribution machine basically takes the stencil produced by the stencil entity, locks it into a fixed position over the printed circuit board produced by the other company, and proceeds to apply solder paste in a wiping motion across the stencil. The machine takes two passes in order to ensure that the paste was applied properly to the board. This is the board manufacturing step that is least likely to have any failure as it is simply placing solder paste on a board.

Pick and Place Machine The second section is the pick and place machine. This machine is the most complicated of the production line that we will use, and as such has the highest likelihood of failure. This machine is connected directly to the previous machine, and the time delay between solder paste distribution and part placement is very short. This is to prevent the solder paste from hardening on its own, as the alcohol inside the paste evaporates In order to operate the machine, a .CSV must be produced from the information generated by eagle. The .csv will be produced from the generated bill of materials, because this excel file has the x and y coordinates of the parts on the board. It also has the rotational information of the parts. When the program is written to populate the board via the pick and place machine, the nozzles that actually pick the parts need to know exactly how far along to move (the x and y coordinates), and how far to rotate (rotation), both of which are gathered from the .csv.

The parts themselves are typically sourced from a reel, which is then prepared and placed into the machine on several feeders. Once all of the information is produced and loaded into the machines computer, and the feeders have been populated properly, the machine can begin the production process. The machine has 4 nozzles, which are connected to a vacuum device. These use suction to pick up any part provided by the feeders, which move the parts forward as they are placed. The suction of the nozzle picks up the part, then rotates to whatever rotation provided by the .csv, then the next nozzle moves to the next part and picks up the part. After 4 parts are placed, the nozzles place the 4 parts from left to right in their place on the board. In order to properly place the new set of parts without the potential for knocking other parts that are already placed on the board, parts are typically programmed to be placed in order of shortest to largest. This also limits the potential of the nozzles clipping parts as the nozzles move about the board.

Baking The last step is the oven, which also follows immediately after the previous machine in order to prevent any problems with solder paste hardening. If the paste is not cooked at a proper heat before allowing it to settle, issues can arise with proper seating or conductivity. The oven is very long, and the parts simply ride a dolly down the oven to be heated. The oven is not heated exactly the same throughout. As the graph below shows in order for typical solder paste (sn63pb37, discussed in section 1.4) to be properly reflowed, a standard for temperature must be met as the board moves through the oven.



As you can see, the board must not be too hot for too long, and typically keeps a temperature of mid 100 degrees Celsius throughout the heating procedure until the very end of the process. Our production line does not have a cooling procedure, and the boards are simply allowed to air cool as they come out of the oven. This is, for our uses, a perfectly acceptable way of cooling the boards after heating. However, if this were for any robust application such as medical or aerospace, it may be required for a refrigeration step after the heating to ensure proper seating and conductivity.

After the circuit board has completed the air cooling cycle, the board is complete at this time. The board has been populated and soldered, and assuming there are no errors in any part of the procedure after visual and some electrical inspection (some continuity testing can be done here with any multimeter), the board should be prepared to be plugged into the mains power from the wall on the input side, and the LED matrix on the output side. The full testing procedure that is expected to be implemented can be found in the next section, section 5.

6.2 Prototype Expectations

The build procedure is detailed in detail above in section 4.1.2 Board manufacturing considerations. In this section, the potential for failure at several steps will be detailed, and any of these failures will be coupled with a contingency plan in order to diminish the impact that the failure will have on the overall time line already set for the project. As the entire project is on a particularly tight schedule, it is important that any an all issues be planned for and a plan written up to avoid them.

6.2.1 Potential Hardware Issues

The most common issues when working on a project of this scope are typically time delays from parts. If a particular part is needed for any application, and it is not available for some number of weeks, this causes a bottle neck in the production of the project and causes serious issues when it comes to producing the completed package on time. In an extremely time limited project such as this one, time is considered more valuable than money. This issue can be avoided by ensuring that no procedure on the board can be accomplished only by a single part. Currently, it does not appear that this is the case with our build, as all the parts are very generic and can have many substitutions that can be made.

Another issue of similar origin is ordering a part that is non operational, or ordering a part that is then destroyed through improper soldering or heating, or through unintentional use causing a catastrophic malfunction. This second type of issue is typically solved by ordering several more parts than are neccessary, then eating this cost as a time insurance. The value of the extra parts far outweighs the potential for providing a late project in Senior Design II.

When designing the board, careful attention to detail must be taken in order to provide a correct design to the board manufacturer and the stencil manufacturer. As mentioned above in section 4.1.2, the stencil and board manufacturing cost is by far the most expensive component. This expense is seen in both time and monetary value. In eagle, it is very important to ensure beyond a shadow of a doubt that every single trace, plane, and pad on the board is impeccably placed so that the physical printed circuit board is perfectly produced by the various companies on the first try. The companies that produce the board and stencil will only do exactly as we ask, so if we ask them to do the wrong thing, we will have to eat the time and monetary costs, which will be significant.

Catastrophic failures were mentioned before in this section, but will be expanded here. Typically, we do not expect a well designed project to fail, but even with proper design, a lack of discipline when producing the final product can lead to a failure that costs time to resolve. There are two steps to prevent a failure that destroys parts, or even entire boards forcing a significant amount of time and money to be spent in order to

resolve. Firstly, as discussed in the above paragraph, impeccable attention to detail is necessary for preventing any problems in the final board production. Experience is key here, as several mistakes that have been made between the various participants in this group lend an insight towards what issues may be seen in design.

One of the more common issues one can expect in design is an incorrect ranking of nested planes. If you have two planes (such as ground and Vcc) that are nested within each other, careful consideration needs to be taken to their rank. Rank is defined as plane priority when the board is manufactured. If a nested plane has a smaller rank than the plane that is surrounding it, the plane will be swallowed by the larger board. in our example above with a Vcc plane and a Ground plane, you would short the power rail to the ground rail and destroy the board and potentially your supply. Typically Eagle is very good at catching many of the other accidental shorting errors by using a design rule check, so such errors will not be discussed here. However, the ranked planes issue is not caught by eagle, so careful attention must be paid to this potential issue. In a similar vein, if any particular part is mislabeled in the library, a catastrophic issue can be introduced (a brief discussion on libraries can be found in section 3 - design). If, for example, the library for the microchip mislabels the ground pin as the Vcc pin, you can quite easily destroy the chip.

A more common catastrophic failure has to do not with design malfunctions, but rather incorrect handling of a perfectly valid board. When soldering parts to the board, especially with small pitch devices such as the microcontroller or the Bluetooth module, it is possible to accidentally short two of the pins or pads together. Even with a steady hand, a microscope and years of experience, it is still very easy to accidentally make an undetectable mistake where a connection is made where it should not be made. This is not always catastrophic, for example you could just disable some functionality of the device or disrupt communication by adding noise. However, as discussed above, shorting power to ground is almost always a catastrophic failure. These issues with soldering can be eliminated by using the pick and place technique combined with an oven, which provides machine accuracy rather than a human hand which has far more potential for error. More insidious and difficult to catch problems can easily arise in testing however, which are normally very difficult to detect. For example, static electricity is a huge issue when working with active semi conductor devices. In Florida, the issue of static electricity is not a huge problem, because the humidity in the air prevents it from being so prevalent. That being said, certain steps can be taken to completely eliminate the possibility for this phenomenon to occur, such as wearing anti static bracelets that are properly connected to a ground source when handling any printed circuit board.

Another issue that can happen with regards to mishandling the board include working with stranded wire near the board. If a stranded wire manages to be stripped and then cut near the board, very small diameter conductive wire filaments can fall onto the board causing shorts between small pitch devices. This can be resolved typically by blowing compressed air onto the board, blowing away any bits of wire without damaging the device. However, if this is not caught before powering on the device, failures can occur.

A last thought about catastrophic failures with regards to mishandling has to do with the final mounted location of the board. This may seem like an obvious consideration, but it is worth discussing here. If the mounting location is conductive (which we do not expect it to be, but this may change), it is important to properly isolate the board from the conductive mounting surface so that you do not short wildly across the board. This can be done with tape or other rubber insets. As the saying goes an ounce of prevention is worth a pound of cure, so steps will be taken to avoid these issues. As long as we do our part in regards to proper handling of the board, none of the issues discussed above should arise.

6.2.2 Potential Software Issues

As with any software based projects there are many potential issues that must be taken into account and more-so due to the software being implemented on a device with highly limited resources and debugging capabilities. As such there are a few key elements to be designed for and checked against, being as show in Figure 45 the issues of buffer overflow causing undefined behavior. Buffer overflow is an issue that arises

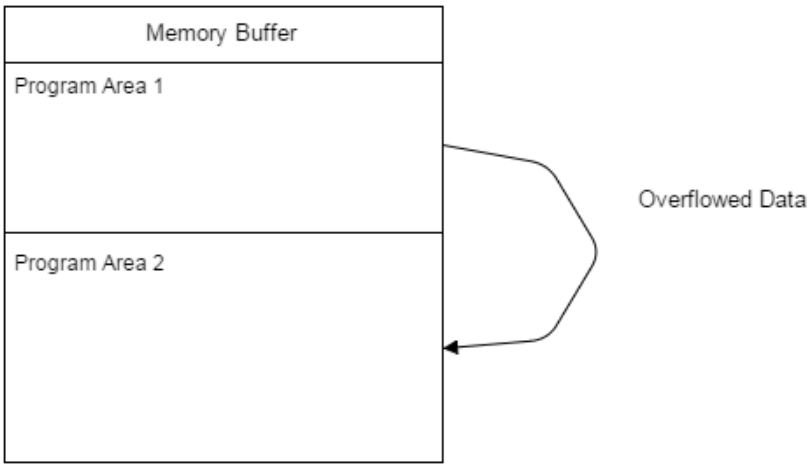


Figure 45: Demonstration of Undefined Behavior due to Buffer Overflow

from the the lack of additional space for a program to expand into or the lack of buffer safeguards for when an (i.e) integer hits it's maximum limit and thus overflows into the neighboring address. As such both of these issues cause extremely unpredictable results and behavior and can cause a number seeming unrelated issues crop up, thus making this near impossible to debug for. Therefore, due to this issues, care must be given early on to mitigate any such problems and to avoid the possibilities entirely by adding in safeguards into the code. Additionally race conditions must be planned for an accounted for, these are when concurrent access to the same device or address happens. While initially this does not seem odd, but the issue is easy to understand when a read/write cycle is observed. Assume two operations are accessing the same address of memory, one has a read command, the other a write; as such the read command is dependent upon if the write command executes before or after in the information it

retrieves. Therefore if the read command's calling function is critically dependent upon the return information the desired behavior is almost most assuredly going to be completely irrevocably incorrect. There are a number of other possible issues that can arise from the compiler and various systems being used on, but the issues shall be mitigated by the production of the code being on one standardize system to help avoid this issue.

6.2.3 Prototype Constraints

There are various economic and social constraints to be attended to in regards to this project. In order to perform several of the functions we desire in terms of physical production of the product, we require a stencil. The stencil is a fairly expensive part, typically around $200. As a result, even though this price has been included into the budget for the product, it may be desirable to hand solder as many of the parts as possible. This being said, the fine pitch devices will likely restrict the ability to hand solder the parts onto the board. Hand soldering such parts could lead to accidental shorts which could damage the board in the worst case, or cause unreliable behavior in the best case. The size of the board is not unlimited as well, for monetary reasons as well. Although technically from a physical perspective, the board could be as big as the bottom of the table, this is limited by the price of such a board from the board manufacturer. That being said, it is not relevant to have an arbitrarily large board because long thin traces, such as the ones on an arbitrarily large board, would act as antennas. We are using bypass capacitors to handle shorting of AC interference patterns, however it is not desirable in our communication lines to have any interference. As a result, we do want to keep the board as small as possible with the shortest trace length for each connection.

From a social standpoint, the board is currently only going to be implemented as a one player solution for several reasons, discussed below. It is important to understand that from a technology aspect, there is no limit on having several users access the device at the same time. The code that we are implementing allows for an arbitrary amount of foreign connections to the device. However, the limit comes when it is considered that the board is only 16 x 8 pixels. This severely limits the amount of space allowed for several people to be playing at once. There would be too much overlap, and as a result the usability of the product would not be optimal. It is possible to overcome this social limitation in two ways, only one of which is practical for our use. The first, non practical solution, is to add more LED pixels to the project. As a result of this implementation, the resolution would be increased and there would be more space for more people to be included in the experience. The second solution is a software solution. This is to come up with a more clever way of applying our 512 pixels to allow multiple people to play at once. For example, a way of allowing two players to interact in a limited space is to include a version of snake where two players could compete for the food, and if they contact each other they would lose the game. Several solutions to the multiple player problem likely exist, and it is an issue that we would like to at least attempt to tackle in the implementation of the software of our product.

6.3 Parts Acquisition and Bill of Materials

Parts in Table 18 includes the electrical engineering aspect of the build, including the physical printed circuit board and the components on the board. In order to procure parts for both the final design, parts will be sourced from mostly Digikey and Mouser, as the parts from these distributer are expected to be high quality and low potential for failure. That being said, the parts will still be ordered in larger quantities than necessary in order to prevent a single bad part setting back production by potentially several weeks. In order to ensure that the parts are operational when received from the source, testing procedures detailed in section 5 will be implemented on the individual parts.

Table 18: Bill of Materials

Part	**Part Number**	**Cost Per Part**	**Number in Design**	**Total Cost for parts**
Microcontroller	TMS320F	32.12	1	32.12
Transformer	237-1577-ND	4.69	1	4.69
AC-DC Rectifier	MDB6SFSCT-ND	0.43	1	0.43
12VDC Regulator	MAX5033CUS	3.68	1	3.68
5VDC Regulator	LM2674MX-5.0	3.28	1	3.28
3.3V Regulator	LM3940IMPX-3.3	1.49	1	1.49
Bluetooth Module	RN42	15.27	1	15.27
Neopixel	1528-1809-ND	0.50	512	256.00
TVS	SMAJ9.0A	0.57	1	0.57
Resistors	Various	Various	<15	Low
Capacitor	Various	Various	<15	Low

Total Price for components: $317.53

7 Testing

This section will outline various test procedures intended to ensure that the device functions entirely as expected. The test procedures will be discussed in detail, and expectations for said tests will be laid out.

This section includes:

- Hardware test requirement overviews for various devices within the smart table.
- Performance test characteristics of the main microcontroller.
- Procedures and environments for testing the Bluetooth module, power supply, and LED matrix.
- Simulated and physical software test specifications.

7.1 Hardware Testing

Several components on the device as a whole require specific consideration before compiling all of the parts into a completed project. The microcontroller and Bluetooth module have a large code base, which can mostly be tested in real time by the development of a testbed. That being said, the final code will need to be tested on the physical chip to ensure that the device is executing the code as expected.

In the hardware domain, several components are to be considered for testing. Of primary concern is the power supply circuit, of which the design has been discussed in section 3.1.7 above. Several components are implemented that will not be tested as to their incredible likelihood that they will every come into play, and damage could be done to the components downstream of the power supply by pushing the system to the outer limits. Such components include the Transient voltage suppressor, which will only kick on after a very large voltage is present, which we certainly do not expect to encounter under standard runtime. On top of this, it is expected that such a device has been tested by the manufacturer of the IC, and as such testing this device is outside of our domain. A fuse is another example of a device which is outside of the domain of our testing procedure, as we expect such a device to function direct from the manufacturer. However, there are some tests that we need to perform on the power system. Reference section 5.2 below for a detailed explanation on this testing procedure.

Typically, you can assume that every part is working properly when coming from the factory. However, it is our duty to ensure that this is the case before employing the part into out final design. In this section, the test design for the individual parts will be examined. If the parts individually work, then it is assumed that once combined properly on the board that the parts will not fail. However, it is important to check the

final design before combining each of the tested parts. If the individual parts work properly, and the total design does not work, the design is typically the issue in some manner or another. It is important to mention that in some of the testing procedures, such as the fuse testing, that it will be assumed that if one part in a batch is operational, than the entire batch is operational.

When testing the compiled hardware, it is important to remember that after the board is compiled with several working parts, it is assumed to be working. The quickest way to test the hardware is to plug the final design into the mains power, then use a multimeter to test the various nodes that are mentioned in the circuit diagrams. Namely, such nodes as 12Vreg, 5Vreg, 3.3Vreg, ect. For the hardware side, there is no complicated AC signals that need to be processed as the first step is to get the AC wall signal into a DC waveform. Therefore, no oscilloscope will be needed to ensure that the hardware works, however the software testing will require an oscilloscope in testing to ensure that the proper serial communications are being transmitted.

7.1.1 Hardware Testing Overview

Fuse The fuse is a simple part, an as such is a relatively easy part to test. This is an important part to ensure the functionality of before implementing into a design, as a failure here could destroy the functionality of our product, or worse cause a fire causing other damage. Considering the product is likely to be housed in a wooden enclosure, it is of utmost importance to verify the part works. Any increase in current due to a problem down stream from this part will be caught and shut down by the fuse. For our use case, a simple test can be implemented here. In the lab, we will simply hook the fuse up to a voltage source and a small resistor of known value (10 ohms is a good value to use). Then, as the voltage is increased, it is expected that the fuse should trip at around >50VDC for our test case. This 50V is the value that the current of the test circuit exceeds 5A, which is the fuse value we plan on using.

Transformer The test procedure for ensuring that the transformer is operational is as follows. We are using a 5:1 AC to AC transformer. Therefore, whatever input is received to the input should be transformed down to 20 percent of the original value. As can be plainly seen, the test procedure for this part is very simple and can be done in any lab with a function generator. Any AC sinusoidal voltage input by an external source will be measured at the output, and if the voltage is not 20 percent smaller than the input, the device is not working. Most importantly for this project, we should ensure that the transformer can handle 24 Volts as we expect, otherwise our power system will not function properly. An in depth examination of the testing we expect to perform on the power system itself can be found in the next section, 5.2.

Regulators Each of the Regulators will be tested in the same manner, as they are all simply switching regulators with different output values. Every regulator input has a DC value, and we know the expected input to each (18VDC input to 12V regulator, 12V input for the 5V and 3.3V regulators). In order to test the regulators, we will perform a

similar test to the transformer except we will be using DC voltage as input and measuring DC output Voltage. For each switching regulator, tested values will be compared against expected values, and parts will be given a pass or fail rating based on how they perform.

Microcontroller and Bluetooth Module These two components will not be as vigorously tested from a purely hardware point of view unless a real problem is noticed by the computer engineers. We will be assuming that the component is operational, and the testing will be performed on the software side to ensure that the serial and UART communication is performing as expected. An expansion of this testing can be explored in the Microcontroller module testing section, 5.3.

RGB LEDs The RGB LED has several components that we can test from a hardware and software component. We expect to use Adafruit's specialty RGB LED, known as the NeoPixel. As discussed in the previous paragraph, section 5.3 will include a detailed explanation on the software and communication testing that will be performed on these LEDs. The hardware side of the testing will be relatively straightforward, where the LEDs will be powered, and simply ensuring that each LED is operational. That includes placing the 5V source on the Vcc and grounding the LED, and ensuring that the LEDs are in fact operational.

Various Passive Components These components can be tested simply by using a multimeter. The multimeter has components for both resistors and capacitors, and some even include capability to check inductors. Because of the simplicity of these parts, they are typically assumed to be operational out of box, however we can still check the components to ensure that we received the proper parts from the distributor.

7.1.2 Microcontroller Testing

Current Test Characteristics The performance characteristics of the chosen microcontroller are exceptional. As seen in Figure 46, there is an approximately linear relationship between the current draw and the microcontrollers clock frequency. This means that the chip is highly efficient, even under high loads. This is very significant, since the large number of LEDs in the smart table tax the microcontroller exceptionally during subroutines that utilize every LED in the matrix. This characteristic is also reassuring because it probably means that the matrix array can expanded without fear of too much current being pulled by the microcontroller, since it has an upper clock frequency of approximately 200 MHz where it pulls approximately 0.5 A. The only limitations to worry about would be performance, since additional LEDs would require sufficient RAM.

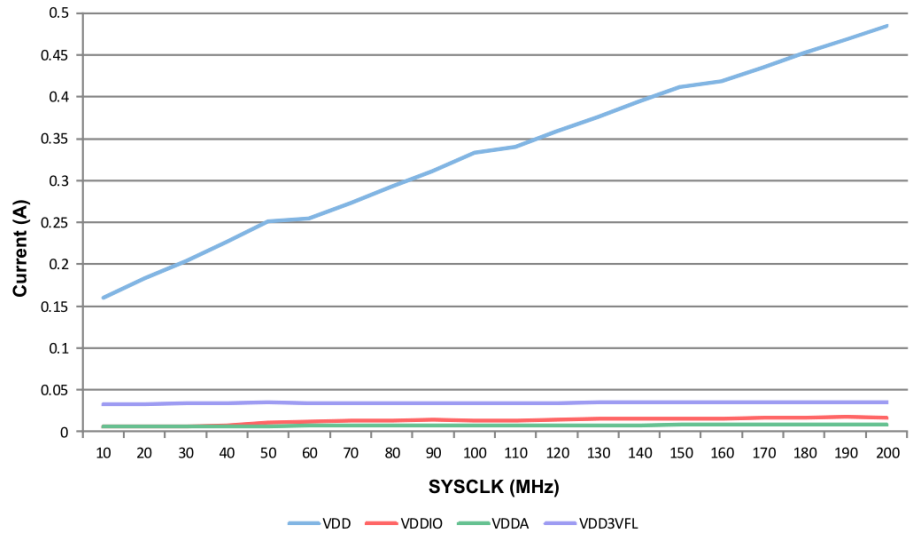


Figure 46: Current Characteristics of the Microcontroller. *Permission submitted to reproduce*. [Ins16]

Power Test Characteristics Similarly, the power characteristics of the microcontroller are also exceptional. As seen in Figure 47, the microcontroller consumes under one watt at the maximum clock frequency. The relationship is linear, rounding out at approximately 0.6 watts at 1 MHz. Even at maximum clock speeds, the power dissipation of the microcontroller is marginal. The is significant because there is no fear in damaging the microcontroller under heavy stress, as long as it is not overclocked beyond the factory clocks and outside the desirable performance characteristics. A more detailed explanation can be found when discussing the nonlinear temperature characteristics observed in Figure 48.

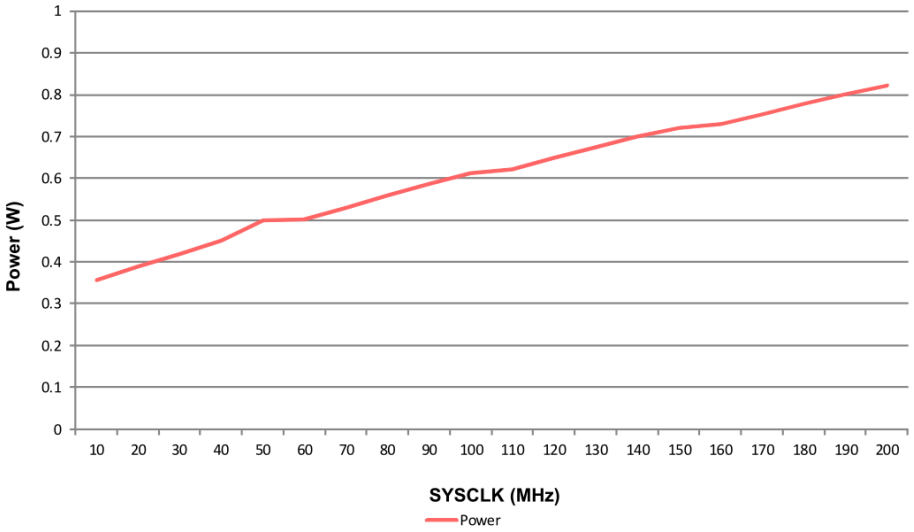


Figure 47: Power Characteristics of the Microcontroller. *Permission submitted to reproduce*. [Ins16]

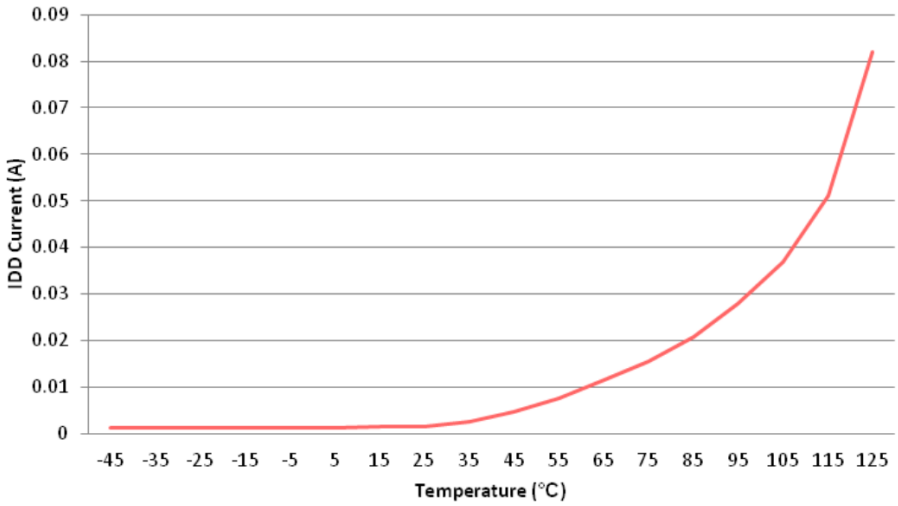


Figure 48: Temperature Characteristics of the Microcontroller. *Permission submitted to reproduce*. [Ins16]

Temperature Test Characteristics In Figure 48, it is evident that the temperature curve shows an exponential relationship with the current draw. This is typical of most processors as thermal regulation has consistently been a bottleneck in computing, because very high voltages are required to maintain stability at higher clock frequencies. The current shown shown on the vertical axis is a result of applied voltage, since current and voltage demonstrate a linear relationship. Each clock cycle, the gate capacitances of transistors are charged with energy, which can be represented by Equation 1

$$E = \frac{1}{2}CV^2 \tag{1}$$

Where the capacitance is C, the applied voltage is V, and the energy stored is E. When incorporating the clock frequency, f, Equation 1 becomes

$$P = fE = \frac{1}{2}CV^2f \tag{2}$$

Equation 2 explains the relationship between the clock frequency and power dissipation is linear, which can be observed in Figure 47, and the quadratic relationship between the clock frequency and the voltage. To achieve higher clock frequencies, exponentially higher voltages must be used. This is why the lowest voltage should be chosen at a reasonably stable and fast clock frequency. A higher number of cores in the processor results in larger capacitance, which creates a linear relationship between power and capacitance. This explains why it is more efficient to utilize multiple cores. Due to the relationship between voltage, current, and capacitance

$$\frac{dV}{dt} = \frac{I}{C} \tag{3}$$

It can be deduced from Equation 3 that as the current increases, transistors are charged to their turn-on voltage at a faster rate. These high voltages are responsible for the excess heat production produced at higher clock frequencies, caused by transistors switching at rapid speeds.

7.1.3 Bluetooth Module Testing

The RN-42 can be programmed and tested easily with a PC that has Bluetooth access. Once the module is properly biased with 3.3 volts, it can be paired to the PC. It is critical for the ground pins to be connected as well.

Terminal Environment A terminal environment such as Tera Term can be used to transmit and receive serial data to and from the RN-42 for testing purposes. The terminal environment must be configured for UART communication. The proper settings must be selected in Tera Term in order for the devices to communicate properly over UART. It is important to download the program and open it. In the **Setup** menu, the user must select **Serial Port** and enter the appropriate port in which the module is currently connected. Figure shows the appropriate settings for communicating with the RN-42, after which **OK** must be selected. The user is then free to enter a series of test characters to see if there is a response from the module. Figure 49 shows the required serial port settings to begin communicating with the RN-42 in TeraTerm.

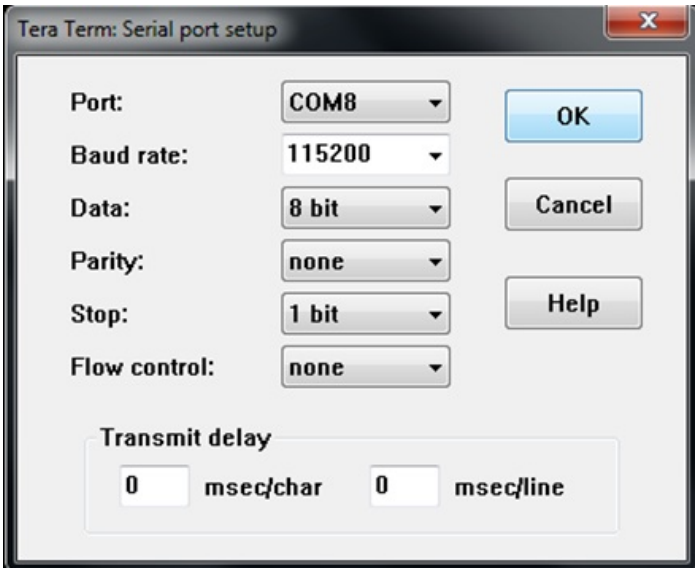


Figure 49: Tera Term Serial Port Settings. *Permission submitted to reproduce*. [Digb]

UART Commands UART commands can be sent through Tera Term to the RN-42 after it has been connected. UART commands can be divided into four categories:

- Get: This command is used to read the UART configuration
- Set: This command is used to set the UART configuration.
- Action/Change: This command is used to change settings or trigger actions.
- GPIO: This command is used to control GPIO pins.

The RN-42 needs to be in command mode to use the above commands, which is achieved by transmitting $$$ to the module in Tera Term, which will be acknowledged with CMD to confirm that the mode has been activated.Figure 50 shows an example of terminal output. Once in command mode, the tester may perform any number of test-related actions such as connecting to devices, transmitting data, altering the GIPO pins and restoring the default settings of the device.

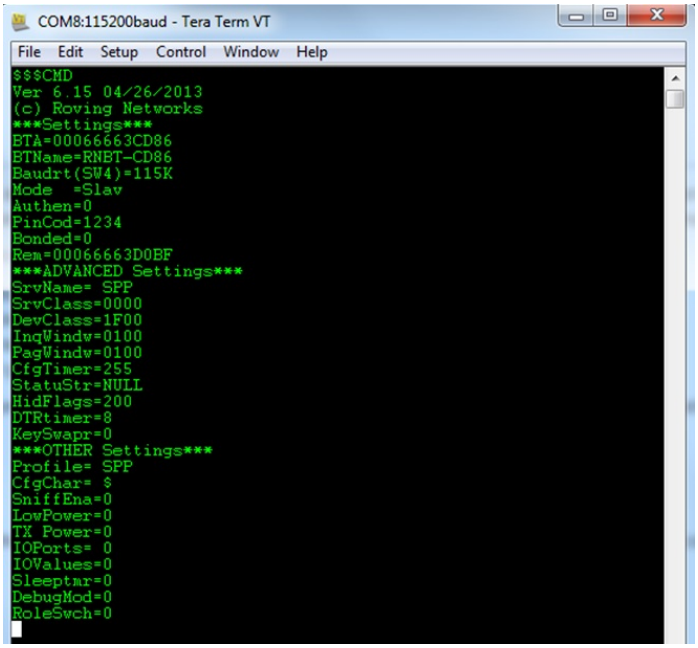


Figure 50: UART Commands in Tera Term. *Permission submitted to reproduce*. [Digc]

7.1.4 LED Matrix Testing

Qualitative Testing The appearance of a single frame on any display is actually an illusion to the naked eye. This is because the LEDs flicker extremely fast as a function of the duty cycle parameter of the pulse-width modulation technique. The duty cycle

effectively defines the brightness of the frame, and by association, the intensity of the colors shown.

Since the LEDs of the smart table are connected in serial, one would assume that there would be a measurable delay when attemping to render a single frame. However, the data signal transmission delay time is roughly 300 nanoseconds, which is far too short of an interval for a human being to detect. As a result, a signal pulse can activate the exact number of LEDs required for a given frame, even when the LEDs are cascaded together.

Quantitative Testing It should be noted that the terms refresh rate, frame rate, frames per second, and Hertz are all interchangeable and refer to the same measurable quantity. Since the distance between the LEDs will be constant once the final matrix is configured, the only parameters to test will be the lower and upper frequency limits and brightness values of the LED matrix.

A refresh rate of at least 15 Hz is desired, however, 60 Hz would make the smart table competitive with many commercial displays. It is said that some humans can detect flickering when the refresh rate dips below 60 Hz. The refresh rate could be measured by a tester with an oscilloscope by probing the PWM GPIO pin connected to the LED matrix. The refresh rate would simply correspond to the measured frequency. Brightness values of the LED matrix are the result of the duty cycle of the PWM signal produced by the microcontroller. This signal would appear as a rectangular waveform, and can also be observed with the help of an oscilloscope.

7.1.5 Power Supply Testing

In order to decide how much current is required to run the LED matrix, we simply take the current draw of an individual LED pixel, then multiply it by 512 to decide what the power supply will need to supply the system. In this case, the power consumption from the other various parts are negligible. This includes the microcontroller, the Bluetooth module, and the various other components on the board. The max current draw from the individual LED pixels is about 60 mA. It is not specified, but this max current draw is likely in the case of all 3 LEDs sustained on, which produces a white color.

In order to ensure that the limits of the system are not pushed, careful consideration is given in the code and in the application of the hardware to ensure that as few LEDs are pulling the max of 60 mA at any given time. For example, we do not want to have the entire screen all displaying white at any time. This is not just a good idea from a power consumption perspective, but also from a design perspective. Such an arrangement of pixels would be jarring, and we want the product to be as aesthetically pleasing as possible. The only situation that would draw such a large amount of current is the application of various animations.

It is expected that the applications that we implement to designate various system

processes, like loading or booting up, will potentially be using large amounts of system current. With that observation, it is important to note that this situation will occur for a fairly short amount of time when compared to the total runtime of the product, in other words there will be brief periods of large power draw. The power system should be able to handle the sustained white on all 512 LED case for a relatively long period of time, in order to ensure that at no time the device will become damaged by over consumption of power. However, it is impossible to sustain such a current draw for an indefinite amount of time, and a solution to this will be discussed below.

$$60\ mA * 512 = 30.72\ A \tag{4}$$

The equation above is a reflection of the absolute maximum current draw for the system. It is important to understand that, through proper design in the software, we should never come close to such a wild maximum current draw. However, we should still design the hardware to be able to handle this fringe case. We are expecting to be operating at, for an absolute maximum situation, around 33 percent of this max. This leaves us at a power system that can handle 10A of current supply at any given time, which is still a very high current draw for such a system. This also means that large heat sinks will need to be included into the power system, but this is a discussion that has already been visited in the design section, section 3.

In order to test that the power system can handle such a power draw, we will need to prototype the system on a breadboard. The intent is to have a software build set so that we can run various test animations in order to check the current draw of the system, and determine whether the power system can handle the current draws. The 10A that we expect to be operating at is a bit dangerously high to be performing tests with multimeter, as most have a 500 mA fuse and a 5-10 A fuse. As a result, there are several ways to bypass this and test the current in a less invasive method. The two safest methods are to use a current probe attached to an oscilloscope, or to connect a 1 ohm resistor directly following the output of the 5V regulator in series with the power rail of the LED matrix circuit. This way, we can simply measure the voltage over this resistor, and following Ohms law

$$V = IR \tag{5}$$

Therefore,

$$I = \frac{V}{R} \tag{6}$$

we can determine the current in Amps is equal to the voltage across the resistor. The

microcontroller that we are implementing has various analog to digital converters, which we will be taking advantage of to display the instantaneous current to the display. This is one of the various subroutines that we plan on complimenting, which are discussed in the software discussions above in section 3.

No matter the method of testing the current through the system, as mentioned above we desire to keep the nominal current no higher than 10 A. In order to stress test the power supply, we will figure out experimentally what series of LEDs provide a current of around 10 A. The exact value of LEDs will change with respect to the current draw of the various passive components and the micro and the Bluetooth module, however we can estimate the number of LEDs we will expect to turn on at full power with the following equation.

$$N = \frac{10\ A}{60\ mA} \tag{7}$$

$$N = 167\ LEDs \tag{8}$$

So, the system should be able to handle 167 LEDs turned to white color, and the rest turned off, indefinitely as this would be the nominal 10 A that we desire to handle at any given time. It is important to keep in mind that this 167 LED number discussed is with reference to an LED which has been configured to be white, which is the maximum current draw case for any individual LED. If the LED, for example, is displaying a pure red color, then we would expect the current draw of that specific LED to be about 20 mA. This is because the LED itself is an RGB LED, which means that the color is produced from one or more of the three LEDs on board, pulse width modulated to produce a specific color. In the case of white, which is our maximum current draw case of 60 mA, all three LEDs are on. However, for the red case, only the red LED would be on, which is 1/3 of the total LEDs. So, the power draw of the individual LEDs at any given time are likely far below the 60 mA max discussed above. For an approximation, we can assume that around half of this 60 mA current draw, 30 mA, would be the nominal value for any given situation. We can now introduce a new term for our system, nominal maximum current draw. If we know that the current draw of any given LED will be around half of the max of 60 mA at any time, we can perform the following math:

$$30\ mA * 512 = 15\ A \tag{9}$$

This is the expected nominal maximum value, in other words this is the absolute largest value that we reasonably expect to see during the operation of the product. Therefore, by the expectations of the power system analysis, without any detailed practical physical analysis, we expect to be operating at around less than 10 A at any time nominally throughout operation, with spikes of up to 15 A depending on what specifically the system is pushing to the screen. It is possible hypothetically to push all the way up to 30 A

as we saw in the analysis above, however due to intelligent software design we should not see anywhere near this value.

As discussed briefly above, power supply stress testing will be completed experimentally on the device prototype running a test program that will simulate the 10A draw with various 15-20A spikes in order to simulate potential large draws from animations and other potential draws. At this point, the testing will simply be running life cycle style testing on the components, basically running a cycle of current drawing procedures to test the device. After running the test procedure for some time, at this point we expect the power system stress test to last for about a day or so. Keep in mind that for the device to be successful, it only has to operate for 15 consecutive minutes for the board when it comes time to grade the project. However, designing a product to only work for 15 minutes is a very poor idea. That being said, one day of consecutive operation under stressful conditions should be more than enough to prove that the device will function as expected for as long as necessary

7.2 Software Testing

Software testing is necessary to verify that individual software components work as expected, and that unexpected results don't occur after making changes. Unlike hardware software can easily be tested before being applied to physical hardware. Functions can be designed with portable in mind and called from an emulation layer to be tested on a computer before being transfered to the physical microcotroller.

Due to the flexibility of software testing can be performed at multiple stages. Although each stage has it's own complexities, and limitations the earlier you can catch the problem the more convenient your debugging options are.

7.2.1 Software Testing Overview

Simulated This stage offers the easiest debugging options. A simulated environment is best at detecting logic errors in an application. The ability to execute code in a very controlled manner, and retrieve the results in an equally controllable way greatly increases the likely hood of catching a problem early. This stage will allow us to leverage existing testing frameworks, and debugging tools to help us detect the possibility of memory leaks, stack or buffer overflows, and use after free bugs.

The downside of simulating the application is that you can't very easily verify communication with external modules without creating additional simulation layers for them. An example of this would be simulating input, the physical version will be using Bluetooth. The communication protocol for the Bluetooth module is a serial protocol, and will be using the hardware acceleration available on the microcontroller. With us creating our own simulation suite we will most likely replace the input portion with another input function that will access the computers keyboard. The reason for this is that at this

stage we want to focus on core code functionality. Another limitation we will have is screen simulation. The hardware counter part will be using a bit bang based protocol, but simulating that on a computer is very time consuming. Instead we will create a drop in test module that will act like the display, but not have the communication portion.

Although these downsides mean we can't test everything we are given the benefit of having portable code. Meaning that development can continue without a physical version, and you can see the results immediately. This also gives us a clean cutoff of system interactions and opens up the possibility of running our code on other hardware platforms without having to rewrite the entire code base. Instead we can write a new input module, display module, or communication module.

Physical By this time we will be writing our code to the microcontroller, and test with the physical hardware. This stage we lose many debugging options, and must switch to a very different set of tools and methods. The main tests that would be performed at this stage are integration tests.

Debugging with physical hardware forces you to use manufacturing debugging utilities, in this case Code Composer Studios includes the debugger. Along with physical devices like oscilloscopes and digital logic analyses. Hardware, especially communication protocols are known to be time sensitive, and debugging at this stage can actually introduce issues. Their exists a few ways to avoid these timing based failures, the most reliable one being to make sure all debugging actions are passive. Another method is to make sure all external components can function without continuous interactions, and you do not break in the middle of a software driven communication.

As the build process progresses this type of debugging can become harder. In early iterations when all development is done with development boards and breadboards debugging circuity is normally included and accessible over a USB interface. Once your able to move on from this early stage to a custom PCB you must include a programming and debugging interface. Modern microcontrollers can offer interfaces like JTAG, SPI, or their own proprietary in circuit programming protocols. Once the protocol and pins are isolated the decision of how will you connect has to be determined. The option of having an on board JTAG to USB interface is an option, but it increases the per unit cost. The other option is to expose the pins and use an external debugger/programmer. The cost of external devices normally increases due to extra circuit protection, and the programmer being designed to work with a larger variety of microcontrollers. This project relies on the JTAG interface being exposed, and using an external TI programmer/development device. This grants the best compatibility with the chosen processor, and development environment.

In addition to the digital debugging interfaces occasionally you need to inspect the signals between components. This project has two digital devices that the microcontroller communicates with and they are the RN-42 Bluetooth module, and the LEDs. In the case that a communication issues arises between these two parts an oscilloscope or

even better digital logic analyses can help identify the cause. So test points must be added and somewhat easy to access too validate these signals. On a bread board this is easily done with some alligator clips and a wire. Working with a PCB is a different experience, you aren't guaranteed a method of attaching alligator clips, soldering a wire can damage the board if done repeatedly, and having to hold a prob to a fine pitched device is not something you want to do with the increased chance of shorting something.

7.2.2 Simulated Testing

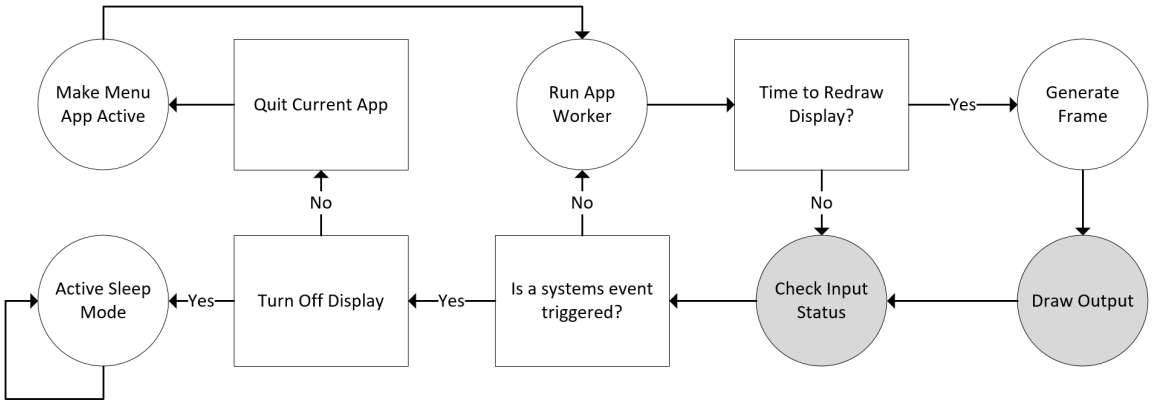


Figure 51: Program State Diagram

This project will be written in C specifically for the chosen microcontroller. In addition to this the code will be designed in a manner where it's possible to call the functions from a standard GCC compiled console application. The purpose for this is to write unit tests for the core modules, and to create a human intractable simulator. To achieve this the program will be broken into modules. For example an input module will get executed on a fixed schedule and update a standard struct containing the current input status. As shown in Figure 51 the coding flow follows a very rhythmic pattern making it easy to separate components into modules. By taking the approach that the code will be running as a state machine verifying inputs, checking if a system override is necessary, and if the display is updated. This structure allows us to standardize the function header for things like "Run App Worker" and "Check Input Status" making it possible to create a simulated testing layer. This also means that anything that would require a platform specific call will perform that call through the screen update or input detection stage granting the desired portability.

Once the code can run on a computer, the actual unit tests need to be written. The idea is that all code will be developed with the concept of test driven development. Test driven development is the ideology that unit tests are created before any code is written. Historically unit tests are something developers have actively tried to avoid. Microsoft used to even have an entire class of developers who would only test and write unit tests. Not testing creates more work in the long run, and we have decided to heed this warning of the past.

The task to determine what needs a unit test, and what to test for is an important task. For example a snake application you want to verify that the snake can move in all directions, and has a game over when it collides with an occupied space. How do you test for the collision? Should the unit test play an entire game? Maybe preset the variables and test specific cases, like the snake is about to run into the wall does it move across the board and does the game end when it hits the wall. How do you test the generation of new pieces for the snake to pickup? This project has determined that individual states will be tested. This decreases the complexity of the tests and increases the likely hood of them working correctly after code refactoring. Since the display can have multiple layers all tests will be done with individual modules, so that a full screen game cannot have an overlapping image. For screens that would be partially shown the remaining pixels will be left black emulating a current running application.

As additional features are added with their own unit tests, the code is checked each time it's compiled to ensure no corruption unintended errors occur. If an error is found due to a code change the developer must resolve the issue immediately before it can pushed back into the mainstream branch. If the code is not resolved the branch will be considered broken, and the person will be notified that they have broken the build. This extra level of annoyance is to ensure that coding standards are met, and that other developers are not trying to develop against broken code.

In the case that standard library calls differ between the GCC environment and the Code Composer Studios environment **#ifdef** will be used to either create an alias or remap the call to the appropriate function. Due to the fact that very few exotic data structures will be used their is a very small expectation of problems. The unit tests can also verify memory footprints, and estimate CPU requirements if to much inefficient code is written.

7.2.3 Physical Testing

The testing of the physical product is a harder undertaking. Once the debug ports are available the option to step through the code from a computer is an option. Although it's possible it's not always the easiest way to locate the problem. For example if you are trying to verify the Bluetooth module is working as intended, but your device isn't connecting. It is possible to try and check the serial interface for what the module is returning with a digital logic analyzer, but this can be slow and cumbersome as communications protocols get more complicated or less standardized. The LEDs used operate using a non standard communication protocol. They use a form of pulse width modulation to pass the data similar to a digital servo. In these cases the use of a oscilloscope is more appealing to assist in verifying the pulse widths and width consistency.

This project will primarily rely on manual testing of the hardware. Below is the general flow that will be followed to perform a full system test.

The general flow will be disassociate any device currently paired with the Bluetooth

module. Turn on the table, and a basic screen animation should be displayed. This will display a simple RGB pattern with a shifting effect to verify that data pins haven't accidentally been shorted and that each LED is properly displaying. If any LEDs do not light up or break the pattern a visual inspection of the LED will need to be done, and replaced if faulty.

Then a device needs to be paired with the table. If the device fails to validate make sure the Bluetooth module is powered on, and has a good connection to the microcontroller. If this doesn't resolve the issue inspect the communication between the Bluetooth controller and the microcontroller verifying that they are operating at the same baud rates. If this appears normal the problem is likely a software bug so connect the debugging interface to a computer and step through the pairing process code.

After the interfacing hardware is verified to be functional we can proceed with a systems test. The first action is to check that the time correctly synced from the Bluetooth device, if this failed attach the in circuit debugger and verify that the time code is correctly trying to poll the Bluetooth device. The next application to test is the weather application, it should match the data available on *weather.com* based on the Bluetooth's device current location. If this doesn't match attempt to refresh the weather from the mobile application. If the weather still isn't correctly reflected attach the in circuit debugger and verify that the appropriate code is being called. Then perform similar tests for all the remaining applications. For example snake doesn't rely on data from the internet therefore you must only test the game logic and Bluetooth input interface. Assuming your able to start the game it should run just like the simulated version.

The tests performed for each application should be similar to the type of testing done with the simulated version. The tester should attempt changing applications multiple times, and in different orders. The tester should verify that all apps function as intended. For example in snake the players snake should get longer for pellet the snake eats, and the game should have a clear game over when the player hits a wall or it's self. The tester should be able to pause the current application at anytime. The tester should also be able to return to the main menu by click a return or start button on the Bluetooth device.

On the main menu the user should have the option to scroll through all the available options, and each selected option should show an animated demo in the remaining portion of the display. These little demos should make it clear what the current selection is, and it must be possible to enter each application. It must also be possible to close each application. During the entire interaction the frame rate should never drop below 30 frames per second.

Given that all tests are successful the build is considered valid, and will should be tagged as a functional commit. These steps are crucial to maintain functional code. Due to difficult of testing with hardware most logic issues should be caught in the simulated testing stage with Unit Tests. The main purpose of this testing stage is to verify

successful integration, and execution of hardware specific code. These testing steps should grant a positive user experience, and a positive development experience.

8 Adminstrative Content

Ever engineering and design effort requires careful administrative planning to be successful. As such, certain administrative tasks must be drafted and fulfilled to facilitate the timely creation of the smart table. Each group member maintains full administrative responsibility and is expected to serve as an administrator to the project. All information is subject to change at any time baring extenuating circumstances preventing the timely updating of information containing within.

The section contains:

- Information pertaining to the division of labor between the project members.
- High-level overviews of the base hardware and software subsystems.
- Projected milestone and a timeline for the project.
- Financial contributions of each group member to the overall cost of the project.

8.1 Division of Labor

Tasks have been divided among group members based on hardware and software subsystems. The tasks that have been delegated are color coded for visibility. As can be seen in 52, some group members share the same task and are expected to work together to fulfill it. These delegations are not binding, since each group member has decided to personally take part in the total development of the smart table.

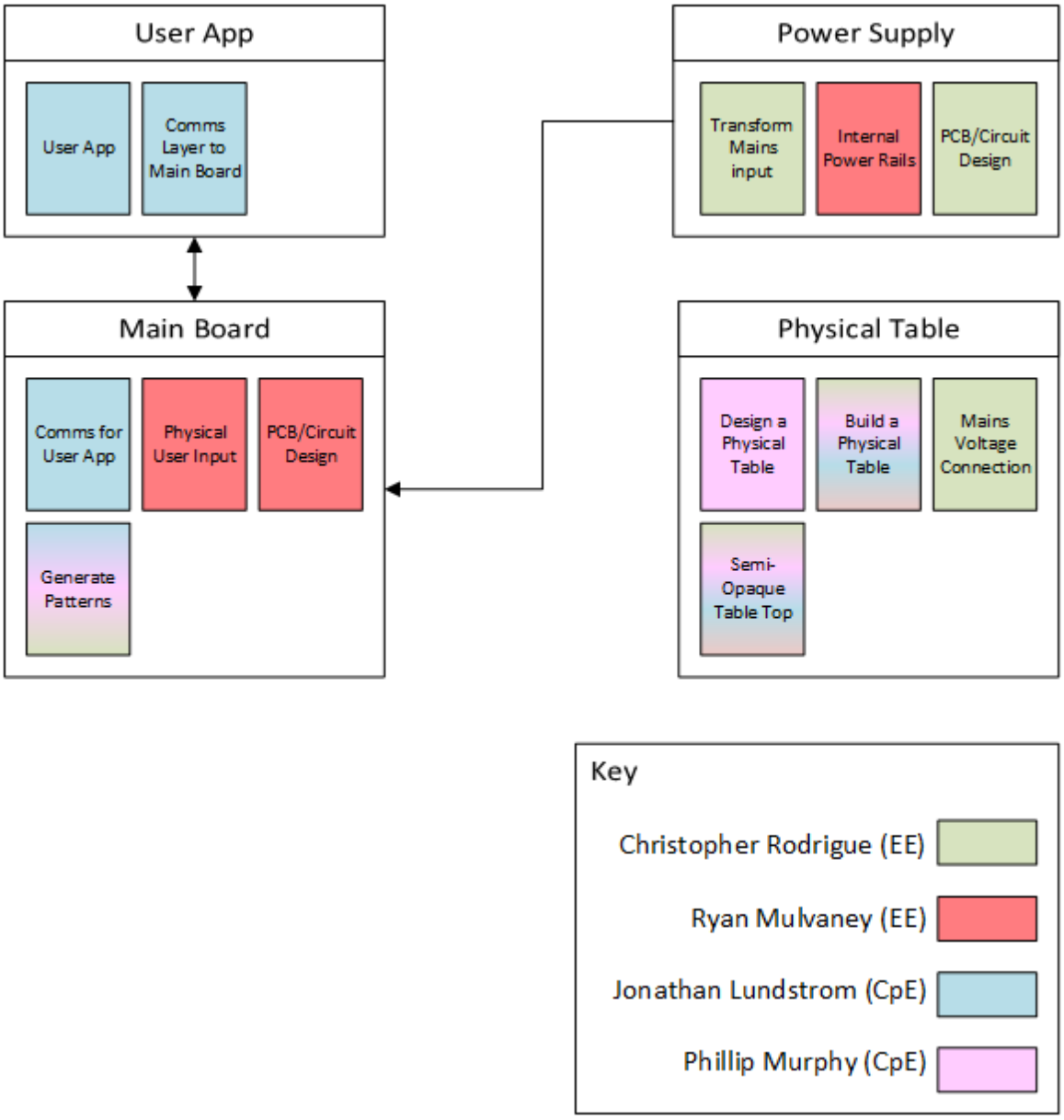


Figure 52: Block Flow Diagram

8.2 Project Milestones

Project milestones have been characterized using Gantt charts. These tools allow for visual clarification of project goals, which encourages timeliness and organization when it comes to meeting the proper internal deadlines. Once all initial milestones are met, the design team will attempt to create a series of additional milestones to incorporate a list of stretch goals for the project.

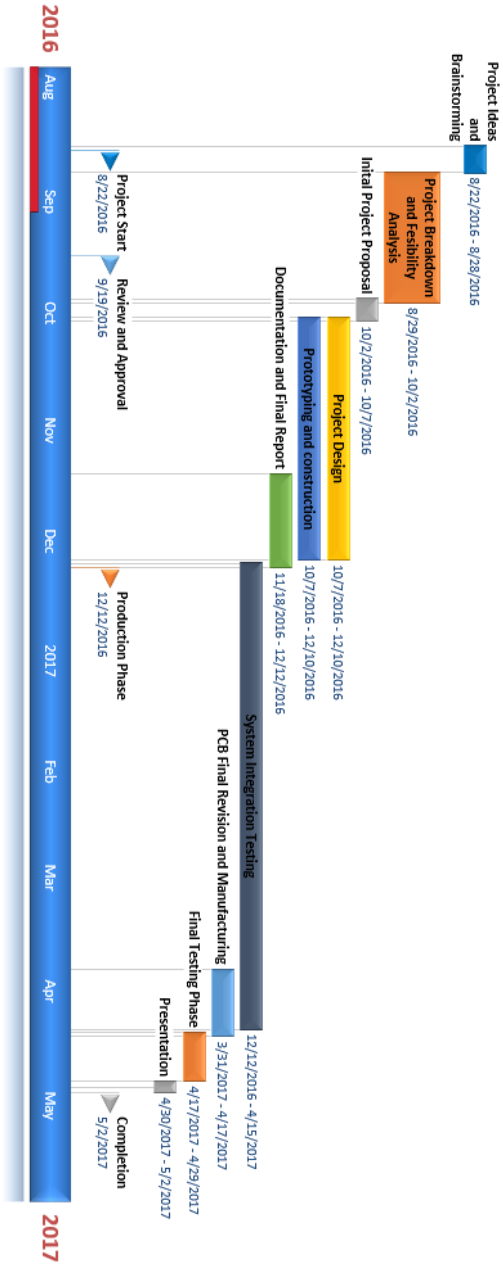


Figure 53: Block Flow Diagram

8.3 Budget and Finance

	Est Amount	Est Quantity	Est Total
Indiv Contribution			
Christopher Rodrigue	$1,000	1	$1,000
Ryan Mulvaney	$1,000	1	$1,000
Jonathan Lundstrom	$2,000	1	$2,000
Phillip Murphy	$0	1	$0
Subtotal			$4,000
Parts			
Prototype Boards	$20	12	($240)
Main Microcontrollers	$15	1 3	($45)
RGB LED	$0.10	512-1024	($110)
PCB Boards	$300	1	($300)
Capacitor/Resistor	$0.01	5,000	($50)
Table Cost	$250	1 2	($500)
Subtotal			($1,245)
Total			$2,755

Table 19: Budget Allocation

This project will not be backed by any sponsors; as such, each group member will pledge the dollar value that they are willing to contribute to the overall cost of the project, which can be seen in Table 19. The tabularized values represent maximum individual contributions to the project. It is anticipated that the overall cost of the project will not exceed $2,000.00 USD. The values depicted are non-binding estimates.

It is expected that the majority of the cost of the project will be spent on PCB manufacturing. Miscellaneous SMD components, such as the RGB LEDs, LED drivers, resistors, and capacitors can be purchased in bulk quantities for relatively low-dollar values. It is noteworthy that the cost of the components is directly proportional to their quality, so this fact must also be taken into consideration when conducting a cost-benefit analysis of hardware assets. Software assets are expected to have zero-dollar value. This is due to the fact that the project will primarily incorporate free or open source software.

8.4 Stretch Goals

A number of stretch goals are envisioned for the smart table beyond the precommitment specifications. If the precommitment specifications are met, a number of exciting features would be desirable to integrate within the table to showcase its potential as the evolutionary successor to the standard table. Features anticipated are ones that would make the smart table more appealing to diverse audiences, such as children and adults. Currently, the development team of the table feels that its current feature

set might only appeal to the interests of young adults, so a number of stretch goals have been drafted that could serve as audience-inclusive additions for the final design.

Activity Tracker Integration The first desirable inclusion would be a software feature that enable the table to display information from an activity tracker, such as Fitbit. Activity trackers are typically wearable devices that can monitor and record biological paramters such as heartbeat, steps taken, distance traveled, calories burned, and more. These wearable devices are often combined with user apps to put the data in the hands of the wearer, who can then use it to make informed decisions and take actions to imporve their health and lifestyle by promoting exercise and dietary responsibility. Fitness trackers are relatively affordable and are a growing electronic trend among both young and old adults. The devices are generally affordable and many health insurance companies even offer discounts for wearing them and sharing the data, which also subsidizes the cost of the devices altogether.

Since activity trackers are so popular, it behooves the design team to incorporate functionality that could display information from them on the smart table. Many commercial activity trackers have developer APIs which provide access to the functions on the wearable device by a third party. Although the table wouldn't be involved in measuring, recording, or analyzing this data, it would still be useful to have another, relatively larger location to view the information quickly and efficiently, such as the table. Through a wireless communication protocol such as Bluetooth, this information could be requested and received by a mobile user application and transmitted to the LED display matrix on the table.

The design team think that this idea has several merits. Since many people, particularly adults, exercise in their own home using instructional videos, often in the comfort of their own living room, the smart table would provide a large and comfortable way for them to view all of the information within the same space. Rather than having to hold and unlock their smartphone, open a mobile app, and swipe through several screens to view information, the user could simply glance at their smart table and view all of the information relevant to their workout on the same surface, customized to their specific needs. A couple watching a movie, both wearing their activity trackers, could view their heartbeats in real-time during suspenseful jump scares or fast-paced action scenes without even turning their head.

Analog Controller Support Additional features that were hypothesized for the smart table include analog controller support for the precommitted video game feature. These controllers are used as input devices to perform various actions within video games, such as controlling the movement of virtual avatars. Many children are still used to playing console video games with analog controllers, and console game systems have more or less always used some form of analog input device, ranging from the iconic Super Nintendo Entertainment System (SNES) to the latest consoles released by Sony and Microsoft. Video games are an ever-growing market, and incorporating analog input is a great way to pay homage and increase the appeal of the mart table to younger

audiences.

Support for these could be easily implemented by purchasing an off the shelf controllers, such as a SNES controller. The wires and connector to the controller could be removed and replaced with a USB cable and male USB Type B connector. A small PCB within the smart table with a female USB Type B header and supportive circuitry could connect the controller to the appropriate microcontroller pins. The microcontroller would contain code that maps the SNES controller button actuation to the LEDs used for the game. Multiple player functionality could be also incorporated for this feature. The design team feels that playing a game using physical controllers, instead of a wireless app interface, would be a more engaging and enjoyable experience, especially for younger audiences.

9 Conclusion

After the completion of this document, a convention was held between the group members in order to assess the feasibility of what we have discussed in the document above. Throughout the group, there is a consensus that the work that we have provided here is of our own work, and also should be considered adequate for the assigned task. Overall, the work was our own original work, and any location where work was pulled from a foreign location, proper references have been made. Please reference section 9, references, for any external sources that were referenced during the production of this document.

Not only has the paper here been prepared originally by the four group members, it is also important to note that the plans provided indicate a reasonably successful project. This means that we expect that, within a reasonable amount of stress and difficulty, we will be able to manage a successful build of the aforementioned product within the time period of one semester. Granted, a large amount of work has been put in this semester, including large amounts of planning regarding parts accumulation, schematic design, and a small amount of board design. Please reference section 4 for most of the design considerations already considered.

The group believes that, through the use of I-con, and through the budget granted and given by the various group members, we should have no problem at all with the physical build. The software design may prove to be the most difficult part, and difficult in this sense does not actually necessarily mean difficult in terms of learning new material to apply to the project as a whole. In actuality, a majority of what is needed for this project is simply knowledge that the two software engineers already are privy to. The difficulty may result from the simple scale of the project. Keep in mind that, through the project is expected to be successfully completed within the time frame given (1 semester), this does not necessarily make it an easy task to complete. A large code base will have to be developed for the success of this product to be operated correctly. Software design, in terms of feasibility and flow charts, as well as other important relevant discussions, can be found in section 4 above in the software design sections.

It is also important to keep in mind the potential difficulty of creating a table from scratch, which is still an option for the group to decide the feasibility of. Should the decision to be made to go with this method, there is both a larger cost associated with this decision in terms of time and actual monetary value. That being said, it is important to remember that we plan on attempting the TI competition mentioned at the beginning of the semester. As a result, it is possible that having a better aesthetic for the project would provide an edge that would allow us to win the competition and potentially have the project pay for itself.

The group has considered several issues that could arise during several portions of the build. The first section for the potential for failure has actually already been assessed

and evaluated throughout the previous 120 pages, and that is the design section. The design has already been mostly complete in terms of hardware, but there are several sections of design that have yet to be approached. Not only this, but these sections of design will likely be left until most of the board has been put together and software begins to seriously be written. For example, there is to be a film on the table on top where the pixels will sit under and project their colors onto. This is likely to be a foggy section of plexiglass, but the final decisions on what type of plexiglass and other considerations such as width, will need to be tested live once the final product is almost done. This should not provide a large amount of project delay, but it is something to consider in the grand scheme of things. Not only should this particular detail be left until the end, the actual table will be decided rather early on into next semester, but still will not be finalized until next semester or possibly the time between semesters.

The next section to discuss is parts procurement. We are also lucky in several regards with respect to this portion of the product development. Firstly, in the same vein as the design, much of this section has been successfully completed. A large number of parts, including the main microcontroller, the Bluetooth module, and the individual pixel LEDs (neopixels), have already been procured. Not only have they been procured, but they have also been tested and ensured that they are the parts that we actually want to use. This is exceptional, because this allows us to simply populate the board once the design is finalized and the boards come in from the board houses. That being said, this portion of the build cannot be simply ignored. There are still several components on the board that cannot be tested in the final design, such as various resistor values. While we expect values such as 200k to be sufficient for pull up resistors, and for the values on the actual neopixels to be sufficient in terms of brightness, there is still the potential for change there. This does present an issue if a problem is found, as this would require a new part order and about a week delay. This is why it is important to have the complete, populated board in our hands as fast as possible so that software design can get underway. Capacitors for bypassing any AC interference are also subject to change, but these changes are relatively easy to make.

As a side note to any mention of parts procurement, another benefit of working at I-con is that various parts are already sourced by this company. In other words, in an emergency situation, it is extremely easy to simply grab several parts from the workplace and throw them into the product as needed. This has already been discussed with the workplace, and has been verified by Ryan Mulvaney's boss. Seeing as any part that would be taken from this location would likely cost fractions of cents, there is no large concern for this. This provides an unbelievable amount of flexibility in terms of final design choices, and prevents a large amount of problems from happening when the time crunch is occurring later in the second semester. We expect to avoid a large amount of pain and misery by taking advantage of this.

Another production consideration to worry about is the board and stencil procurement. This will be handled by an outside company (likely tropical stencil for the stencil and Alberta printed circuits for the printed circuit board), however they do take around a

week to get back with the finished products once you submit a design with the gerber files. An in depth discussion on the gerber files we expect to be producing to these companies can be found in section 4.1.2. Now, these considerations in terms of a time consideration will not be relevant unless there is a serious problem in terms of design. The design has already been finished and checked, so it is not likely that this issue will arise to cost us any time delay. However, it is important to not gloss over any situation that could arise, and have a contingency plan for everything so that any situation that does arise, however unlikely, can be dealt with accordingly.

Now, as mentioned we have a finished design that will produce gerber files to produce to these foreign companies. If, for some reason, should the companies produce incorrect boards, at that point there is a time loss. They will fix the problem free of charge in terms of cost, but they cannot role back the clock and any time lost fixing the problem is lost. This is incredibly unlikely, and has never happened in my experience. However, it is a potential problem. This is going to be avoided by handling the board procurement over winter break. This is an excellent solution for two reasons. First of all, if there is a problem it will be handled before the semester even starts in theory. The second, and better reason that this is a good solution is that, in the more likely case that the board comes back alright, we can begin building the final product immediately, and potentially even before the second semester begins. Overall, this is likely over planning in terms of what is practically going to cause us issues, but it is better to be over prepared than under prepared.

The next section to concern ourselves with is the physical build of the board. This has been briefly considered in section 4.1.2, however the practical issues that can be seen are going to be seen here. The board design in terms of layout has been carefully considered from an electrical standpoint. However, several problems can arise when actually building the board. There are a number of issues that can arise from building a board through a pick and place, followed by a reflow oven. Such issues are not immediately noticeable from a simple schematic diagram or eagle layout .brd file. This is where practical experience comes in handy. However, even with practical experience, it is possible to miss some problem and have to have a board design at worst or simply hand solder the parts at best. It is likely that, due to the simplicity of the board, we would never have to experience the problems discussed below, however they will be laid out as to prepare for their potential. The primary issue when picking and placing a board with a pick and place machine comes from differing heights of parts. If the board is being placed, and one part is placed incorrectly, which means that a larger part is placed before a smaller part, there can be a conflict. This problem is typically avoided by intelligently designing the program that is physically placing the parts onto the board.

Our group will be working with the I-con production team to ensure that the board is intelligently handled in terms of how the board is placed. This intelligent program will avoid a large number of problems in this realm. This will also ensure that the quickest and most efficient pick and place procedure will occur, causing less issues in the long

run. In a similar manner to the previously discussed problem, there is also a problem that can arise in a reflow oven, called shadowing. This is a process where a large part can overshadow (where the name comes from) a smaller part if they are close together in the reflow oven. This large part prevents the smaller part from receiving the proper heat through the reflow procedure, and can cause problems with the solder not flowing properly. This problem is expanded when using non standard solder, such as lead free solder. As a result, our solder will be lead based so that we have as few problems in the build as possible. The solution to prevent shadowing is to build the board in such a way that there are no very small parts that are surrounded by very large parts. We do not expect to have any abnormally large parts on the board, but it is important to understand why some issues can arise so that in the future they can be avoided. As previously said, neither of these issues are likely to be seen in our particular build, however understanding the issues is very important so that if any problem arises from the build it can be dealt with quickly and efficiently.

Once the board is built, various testing procedures such as the ones discussed in section 5 will be implemented to ensure the correct functionality of the board. This is likely to be successful, and if it's not the board can be tested through various methods to ensure that the problem is resolved and the issues are dealt with. Then, any issues from this point are software related. The software challenge will be, comparatively speaking, not particularly difficult. The software engineers on the team have dealt with significantly harder problems before, and as a result it is not likely to see any huge issues with the way that we approach the software. However, it is important to understand that any time code is written for a project, especially when dealing with a new type of hardware like we are with the neopixels, that problems can arise.

Issues likely to crop up from the hardware point of view are issues with communicating with the pixels and communicating with the Bluetooth module. There is a proprietary method of communication that has to be dealt with when handling the pixels. This will be handled by th e microcontroller, which is a software hurdle that will need to be overcome. This provides the potential for a holdup in the design of the board. If the software proves to be more difficult than expected, this could be the most time consuming portion of the build. Or, hopefully, it could take five minutes to understand and prepare, and as a result this would be one less problem to attend to when handling the project.

The Bluetooth module presents its own series of problems. Two major ones are the physical communication between the Bluetooth module and the main microcontroller, and the other problem is the fact that a new interface will have to be built to handle talking to the board through Bluetooth. This will likely be a web aplication of some sort, but no matter what the final realization is, any number of problems can crop up from the design of this section. Building web applications can provide it's own set of issues, and those will be dealt with in time. Bluetooth communication is very standard, and as such should not be complicated when it comes time to produce the final code.

10 Senior Design II Final Implementation

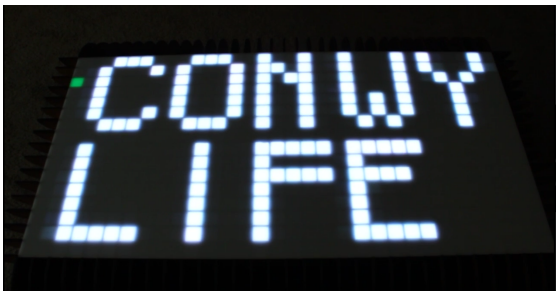
This section will outline the final design and post-prototype table assembly. Although the majority of research and design considerations were followed, a limited number of technical design deviations (TDDs) were taken to guarantee the proper and timely operation of the smart table. The TDDs taken were carefully weighted with respect to necessity, cost, and senior adviser approval.

This section includes:

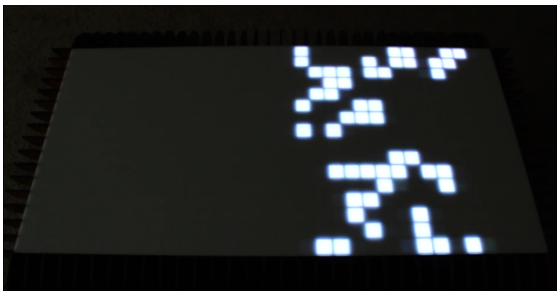
- Images of the final implementation of the software, table assembly, user application, and printed circuit board
- Developer commentary and discussion concerning the technical design deviations taken
- Improvements or additions that could be implemented in the future

10.1 Game and Demo Application Software

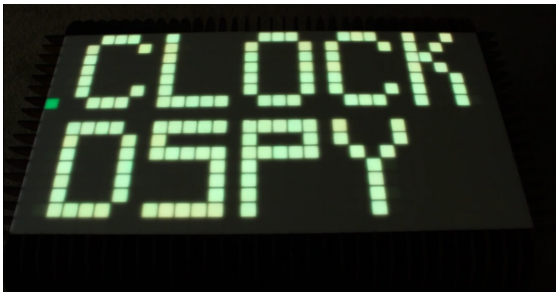
This section will give an overview of various aspects of the software and software development phase.



(a) Conway's Game of Life menu entry.

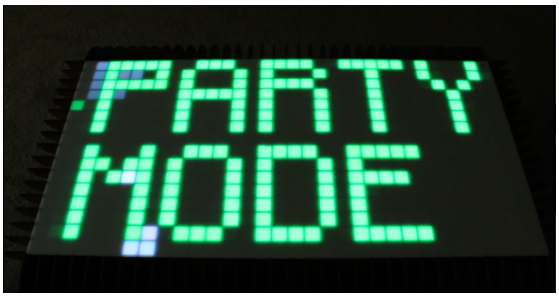


(b) Conway's Game of Life application.

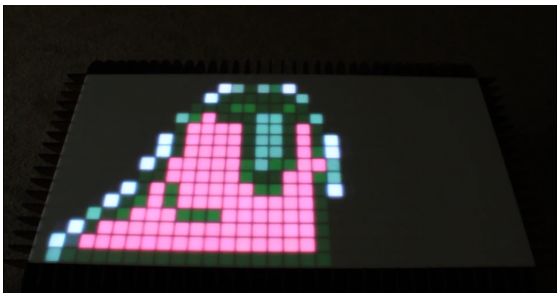


(c) Clock Display menu entry.

(d) Clock Display application.



(e) Party Mode menu entry.



(f) Party Mode application.

Figure 54: Smart Table Demo Applications.

10.1.1 Technical Design Deviations

Originally slated for the smart table was a very limited number of games and applications, namely time, date, weather, and "Snake." The original intent of the table was for it to be a digital information platform. However, the developers discovered that the multicolor display left a lot of room for creativity, which invited for vibrant and colorful applications such as arcade-style video games and animations. Thus, the vision of the smart table transformed into that of a game and multimedia table.

10.1.2 Emulator

To speed up software development an emulator was written that takes the project's code as an input. As shown in Figure 55 the emulator outputs the various functions of the written software for the microcontroller and additionally takes in keyboard inputs for

the substitute bluetooth controls.

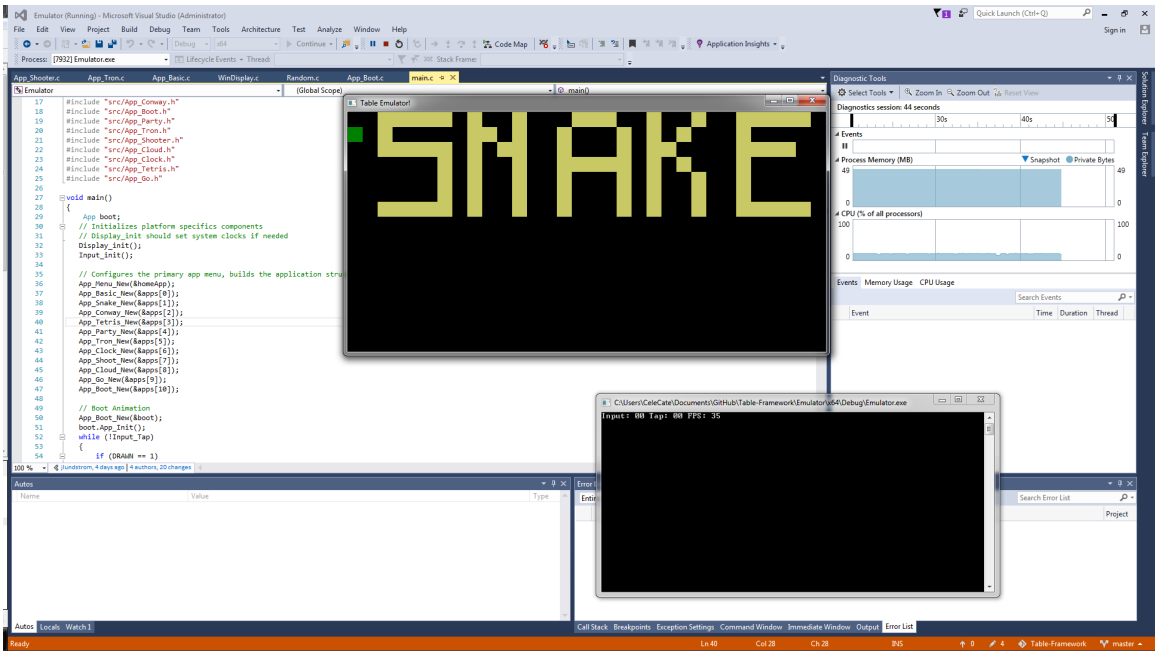


Figure 55: Screen-capture of the software emulation.

This tool ended being invaluable for rapid expansion of the app codebase and functional checks of various lower level function calls.

10.1.3 Demo Applications

The final software implementation of the smart table included a host of games and demo applications. As shown in Figure 54 the visual appearance of the selected demo applications: Conway's Game of Life, a clock, and "party mode." The left column of images shows the menu entry of the demos, and the right column of the image shows the runtime behavior of the selected demo. The demo applications were included to emphasize the hardware and software capabilities of the smart table without requiring significant user input.

Conway's Game of Life This is a simple zero-player game which only requires the inital seeding to "play". As the app is designed, there are presets able to be used or the user and define the seed manually with the controller.

Clock Display This application takes in time data from the external android controller and displays the time.

Party Mode This application is mostly a proof of concept of using a gif and displaying it. The gif is split into separate bitmaps and is parsed externally before being included

into the microcontroller. This possibly could be expanded to on chip parsing if SD card support was added.

Cloud Display This application uses a noise generation application to generate its output. The user can control the functions hue, "zoom factor", and base brightness level. The function being used is a turbulence function that is used to procedurally draw clouds.

10.1.4 Game Applications

Among the other applications included on the smart table are a variety of single-player and two-player games. These games were designed with user interactivity and engagement in mind, and make heavy use of user input. They include Snake, Block Drop, Tron, Go, and Space Invaders, and a detailed discussion of each game can be found below.

Snake This is the standard "Snake" game, in which the player controls a snake that grows after eating food. The food is placed in a random spot and the player dies if the snake intersects itself or crosses into a wall. This game is only single player but is expandable to two player.

Block Drop This is a "Tetris" clone and supports two players. Implemented is scoring keeping, next block, and adding in a second player in the middle of the game.

Tron This game is essentially the Tron lightcycle racing and is strictly two player. The objective of this game is to force the other player to impact into a wall or the trail your "lightcycle" leaves behind it.

Go This game is also known as Chinese checkers. The board implemented is the standard 13x13 training board size. The game does support checking for valid moves and gives visual clues for the player's turn. This game additionally is strictly two-player.

Space Invaders This is a "proof of concept" game, it is roughly a clone of "Space invaders with a hint of Galaga". This game demonstrates the possibly of using pixel art, various animations, complex layered drawing, and complex hit detection. This is a strictly single player game, and shows the potential of larger scaled games being implemented on the framework.

10.1.5 Future Improvements

Future design iterations includes: optimizing the downtime caused by the "waiting for frame to draw" by using double buffering, offloading expensive computations such as noise to the second core, switching the bluetooth polling over to interrupt driven, and cleaning the code in the apps as some are "proof of concept". As such this implementation has a number of shortcuts taken due to time constraints.

10.2 Table Assembly and Wiring

This section will give an overview of the final design, issues, and possible design improvements in the next iteration. For reference in Figure 56 is the completed design to keep in mind during the following text.

Figure 56: Final table assembly.

10.2.1 Technical Design Deviations

The table manufacturing and design were essentially extensions of the earlier research and build work done. As a consequence of the chosen LED height (distance between the LED and the diffuser) that was decided upon during the testing phase, the dividers additionally had reflective material applied on the interior to increase the brightness of each "pixel" as shown in Figure 58. The final result of this additional step, vastly increased the final brightness of the pixel.

10.2.2 Table Manufacturing Process

A quick recap of the base design materials is; made out of 7/32 3-layer birch plywood, laser cut, and diffuser on-top. Additional the requirement of being structurally sound. Overall the main design process followed (find a real-life reference) -> (find an acceptable way to integrate the LED's into the design) -> (revise/repeat), the end result after

multiple iterations is shown in Figure 57. This design was done using Sketchup, and the overall design was driven by the concept of integrating the LED dividers into the table itself, thus resulting in this design. After the model was completed, it was further split up into separate "jobs" and thus was all laser cut with some additional pieces added in as a buffer for future accidents.

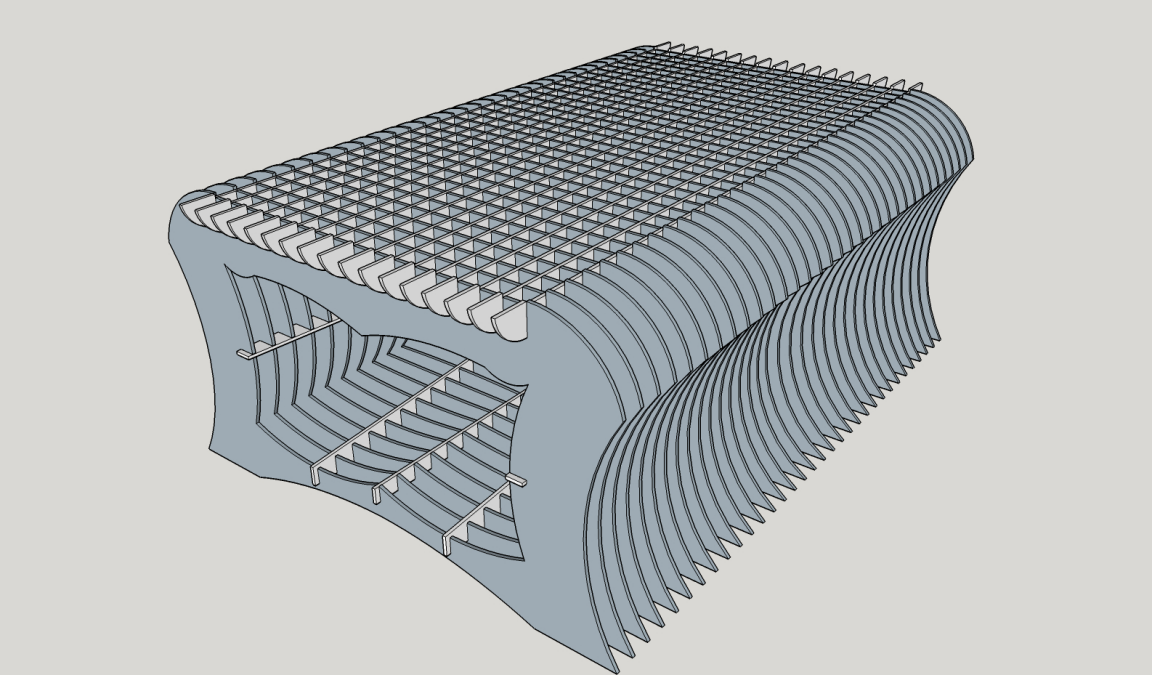


Figure 57: Final table design render.

After all these steps were completed (design, laser, staining, sanding, reflective material application) the final assembly process was started. As the table was partially designed as a jigsaw puzzle, it required three separate people to assemble properly and a large amount of time. The overall assembly flow was (main slats)+(bottom slats)+(placeholder top slats) ->(top slats) ->(LED Backboard) ->(main endpieces) ->(side slats). Thus the table itself was completed.

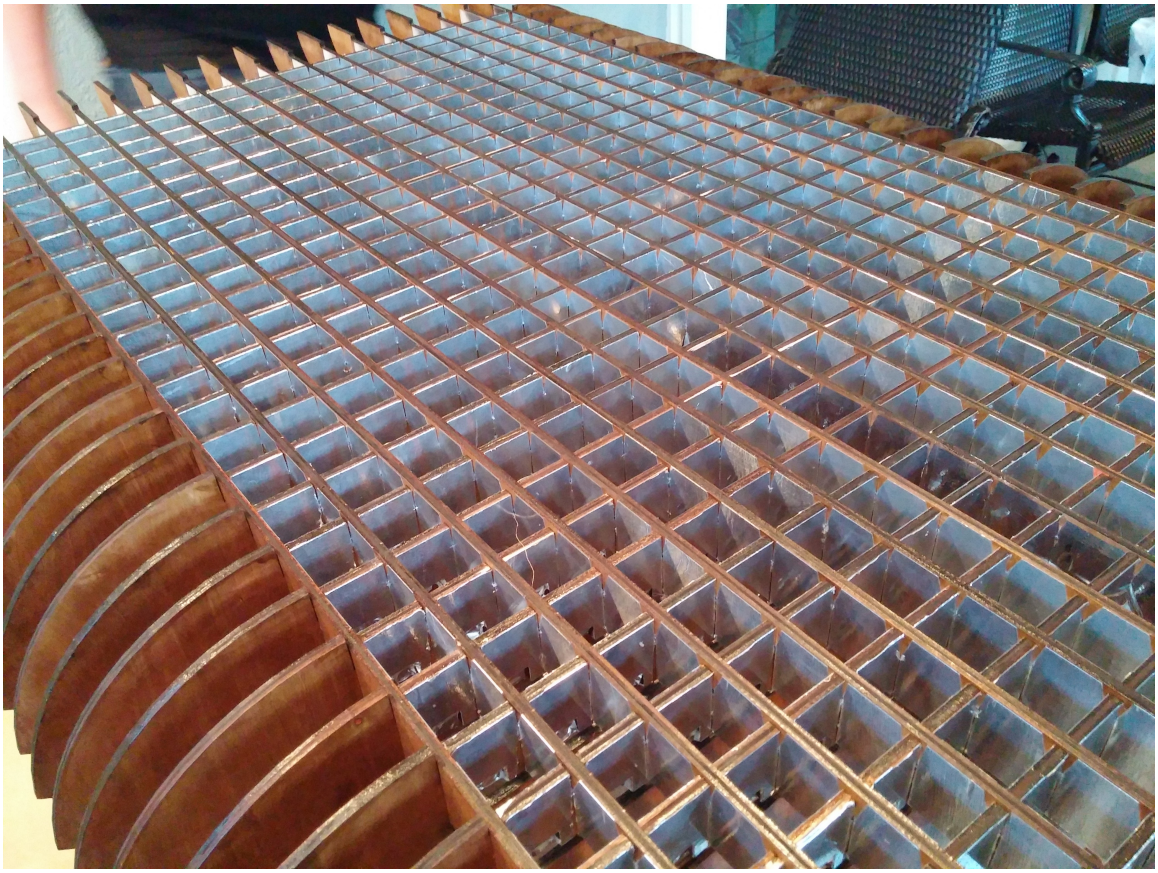


Figure 58: Image of the pixel design showing the interior reflective material.

10.2.3 LED Strips and Wiring

Stated earlier, the table uses WS2812 based LEDs for both cost, and ease of implementation. The final design uses 16 strips of 32 LEDs each with power lines for every 64 LEDs and one continuous data line. The LEDs were attached to a backboard that was inserted into the table as shown in Figure 59.

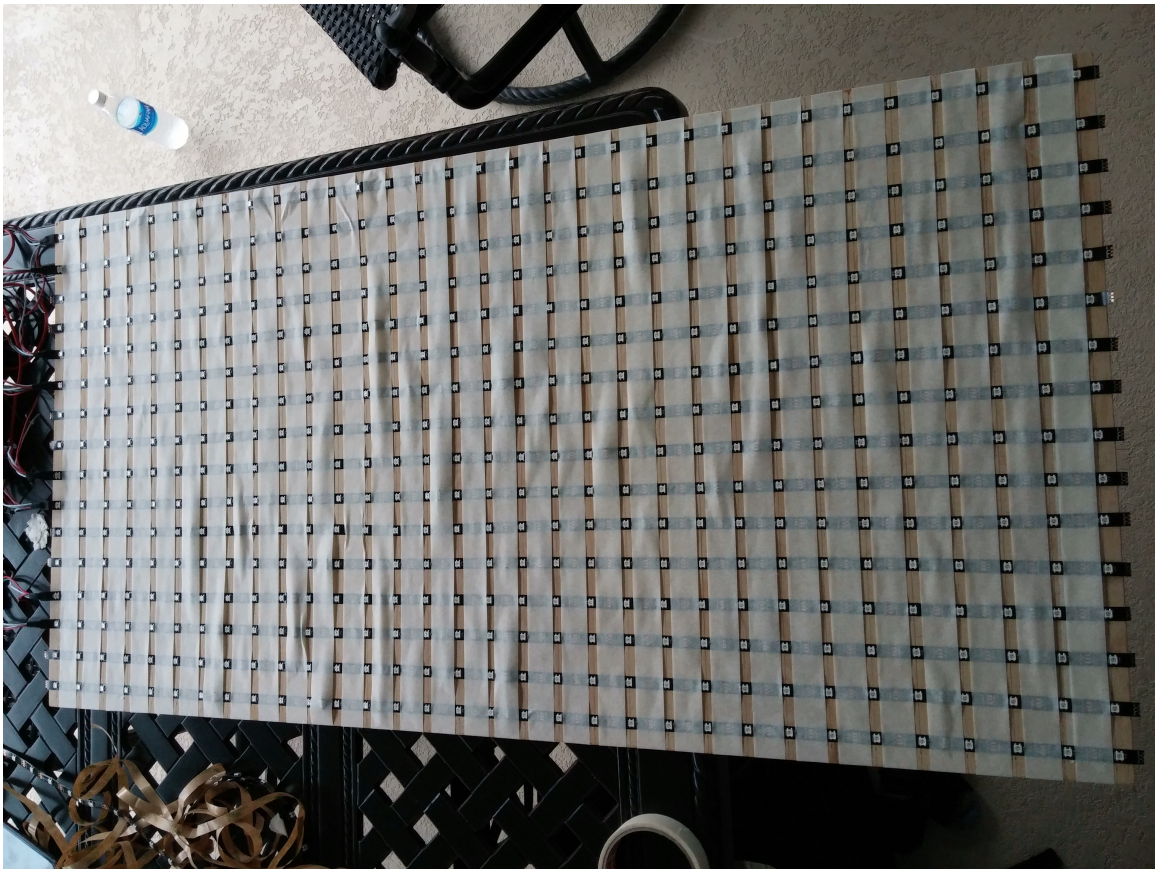


Figure 59: LED backboard inside the table.

Overall, this setup worked out well given the constraints.

10.2.4 Future Improvements

In revision two, more LEDs, less reliance on a laser cutter, and a table design with less "IKEA factor" would be the main points of improvement. More LEDs as the microcontroller can easily handle a larger number and the resolution of the current version is too low for most games. Less reliance on the laser cutter simply due to production costs, as the laser does vastly speed up work, the table design could be simplified to use less time overall and therefore reduces the "IKEA factor."

10.3 Bluetooth and Input

10.3.1 Technical Design Deviations

Bluetooth 2.1 was desired as an ideal method of communication for the smart table due to its ubiquity and compatibility. An important technical design deviation taken was the inclusion of two Bluetooth modules rather than one. The developers decided

to incorporate multiplayer gaming because the general vision of the table shifted from information to entertainment. Since a Bluetooth module can only pair with a single device, two of them are required to support two independent connections. Up to three players are possible because the microcontroller supports up to three independent UART devices through the Texas Instruments Serial Communications Interface (SCI); they are SCI-A, SCI-B, and SCI-C respectively. The first module operates on the SCI-B and the second module operates on the SCI-C. As such, the modules are connected to a total of four GPIO pins on the microcontroller; two of them are multiplexed as inputs, and two are multiplexed as outputs.

10.3.2 User Application Design

There are a couple of approaches to gathering user input: a smartphone user application or a physical remote controller. The developers chose to implement communication in the form of a smartphone application to minimize hardware production costs. The smartphone application was designed for the open-source Android operating system since it is one of the most widely used mobile platforms. The application is intended to serve as a graphical user interface through which the Smart Table can be controlled remotely. The application was written in the Java programming language and allows the user to issue commands using virtual buttons on the touch-screen display of the smartphone. The design lifecycle can be seen in the following image. The most noticeable difference between earlier versions of the app and the final version is the increased button size, as well as the pink and blue color contrast for easier visual differentiation of the buttons.

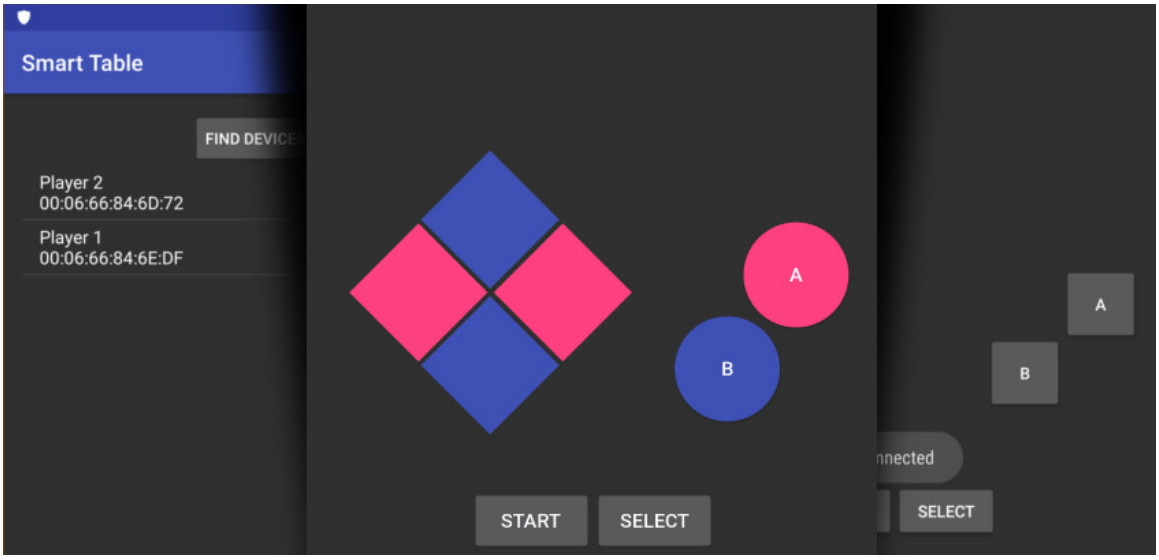


Figure 60: User Application Evolution.

10.3.3 User Application Features

Not pictured are the subtleties of the application. When the virtual buttons are tapped, haptic feedback is sent to the user through the integrated vibration motor of the smartphone. Although haptic feedback is not a replacement for a physical controller, it allows the user to confirm that a virtual button has been pressed without the need for visual feedback. This gives a natural feel to the user, in lieu of a physical remote with real buttons. The application can be used comfortably in either vertical or horizontal orientations, and the button layout resembles that of modern video game controllers to provide the user with familiarity. A defining feature of the user application is that the Android operating system time and date can be synchronized with the microcontroller and subsequently shown on the display.

10.3.4 User Input Overview

An individual simply needs to have the application on their supported Android device to communicate with the smart table. In a typical scenario, a user will download and install the Smart Table Android application. Once the app is opened, the user will connect to one of the two Bluetooth connections, broadcasted as "Player 1" and "Player 2" respectively. Upon tapping a button, the associated command is translated into packets sent over the Bluetooth frequency-hopping spreading spectrum (FHSS) and stored in a data buffer. The data is then echoed to the UART transmit pin of the Bluetooth module. The microcontroller continuously polls the UART receive pin for a command, which activates the various control flow statements in the main microcontroller program.

10.3.5 Future Improvements

In the future, it would be desirable to support multiple mobile platforms such as iOS and Windows Mobile. Physical controller support would also be ideal for those who prefer it. If the developers decided to integrate their product with existing Internet-of-Things devices, such as the Amazon Echo, Philips Hue, or Nest Learning Thermostat, it would be necessary to use Wi-Fi with, or in place of, Bluetooth. Such integrations would make the smart table a more versatile and coveted product.

10.4 Printed Circuit Board and Power Supply

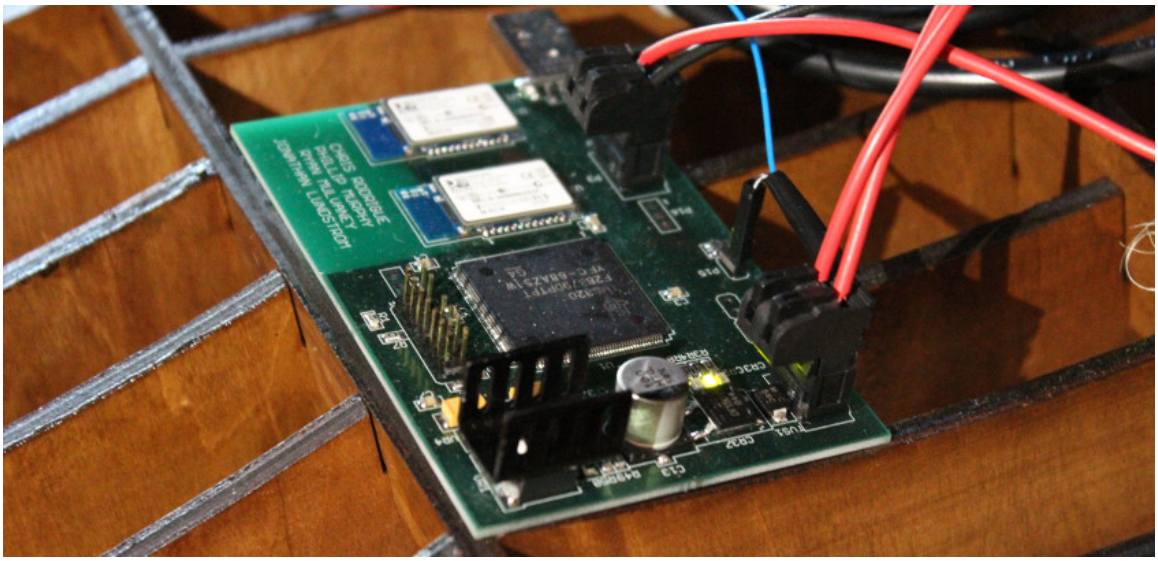


Figure 61: Final Printed Circuit Board.

10.4.1 Technical Design Deviations

Various deviations were made upon completing the initial prototype build early in Senior Design II. Upon completion of the first batch of code and the first PCB, several issues were noted and resolved. The notes of such changes are listed in this section, along with commentary regarding the requisite changes. There were also several connections to the microcontroller that needed to be made that were not made. These connections were prescribed by the data sheet that were missed on the first rev. These changes were not necessary upon further inspection, but were recommended so it was decided to make them in board two. Along with the changes that were made according to the various microcontroller pins, there was also a mistake in the wiring from the Bluetooth module to the microcontroller, which was fixed rather quickly.

The following figure is the modified and final flow chart showing the power and data flow in the overall system.

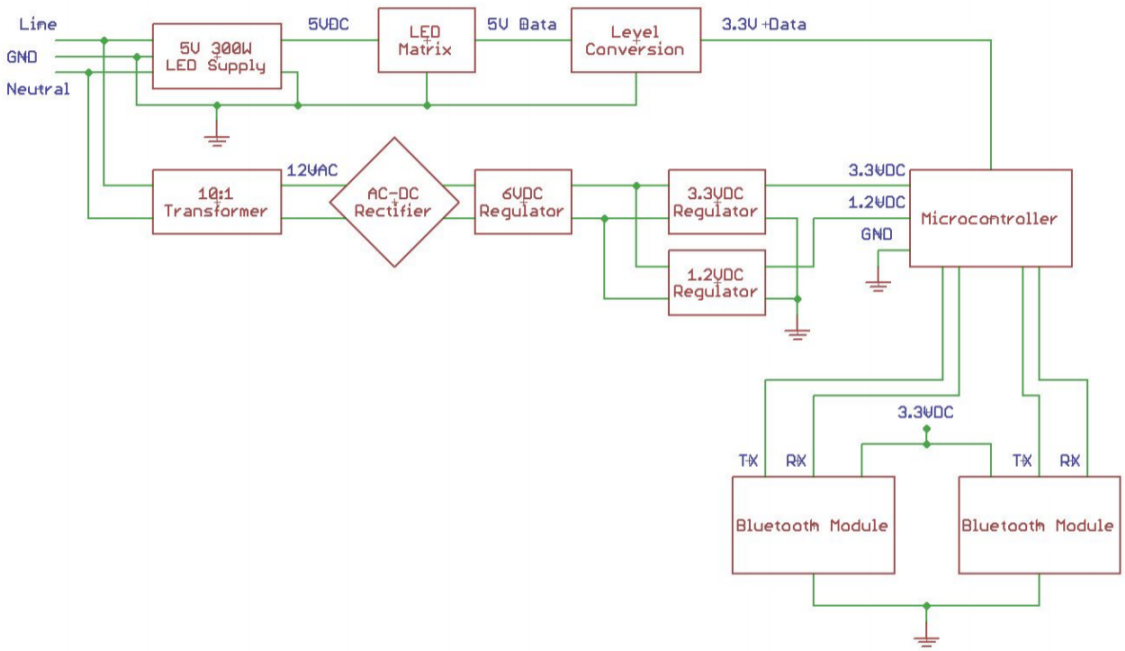


Figure 62: Final Power and Data Flowchart.

10.4.2 Schematic and PCB Design

The schematic is shown as follows, which as discussed above has various deviations, most notably the removal of the various headers that were initially intended to supply the LED matrix, but it was decided after rev 1 to simply have the 5V supply directly feed the matrix. The schematic also includes the new level converter, which is valuable in converting the 3.3V to 5V quickly. This was initially a problem where the standard level converter with MOSFETs was giving us some issues, the TI solution rectified the problem immediately.

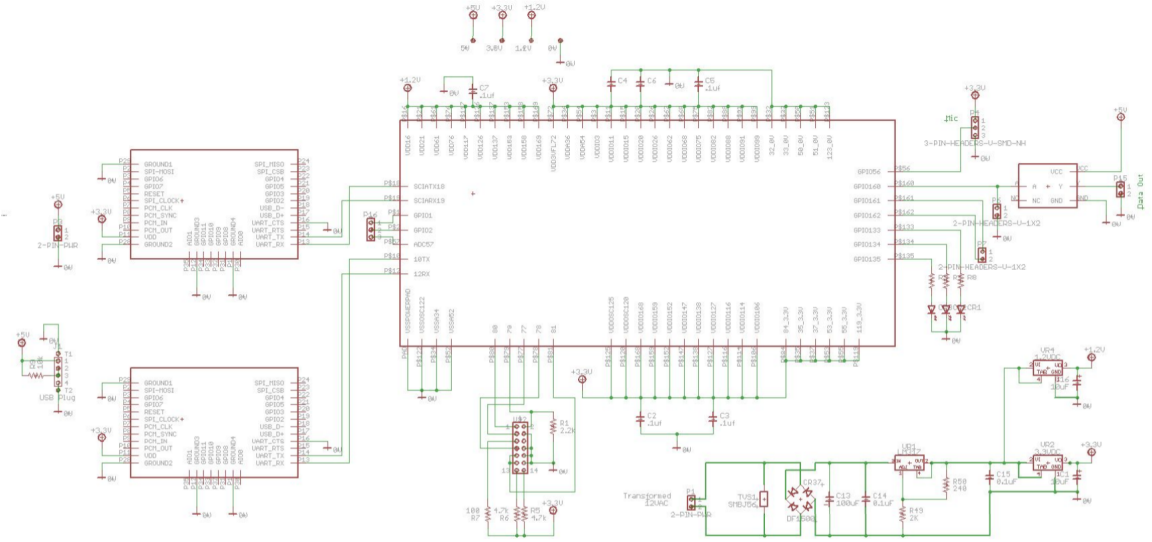


Figure 63: Final Schematic.

Lastly, the board layout reflects the changes in the schematic, the relative size remains the same while adding several features. The second Bluetooth module can be found under the first Bluetooth module, and the headers are removed from the perimeter of the board, making the overall size thinner.

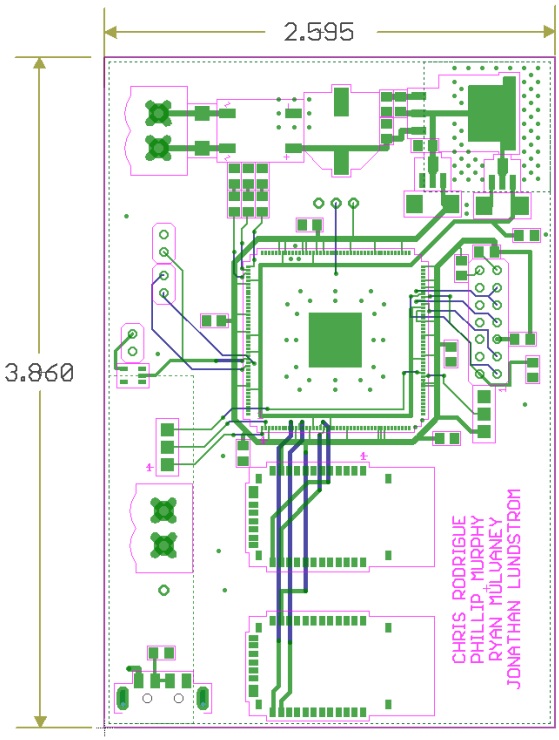


Figure 64: Final Layout.

10.4.3 Power Supply Design

Primarily, the major concern from a hardware standpoint was a problem with heat dissipation. In the first board rev, the decision was naively made to step the DC voltage down after regulation from 28VDC down to 6VDC. This was a bad choice, simply due to pure power dissipation in the regulator. As can be seen in the calculations made in the final presentation given for the conclusion of Senior Design II. The basic power calculations in the old rev lead to a dissipation of 11.4W, which results in a whopping 57 degree Celsius increase in temperature. This was far too hot, and actually led to regulator shutdowns as well as a scathingly hot regulator.

The solution to this was fairly simple. The input to the board, as can be seen from the flow chart in section 5.1.1, was initially slated to be 24VAC, which is wall power run through a 5:1 transformer in order to drop the voltage. However, this results in that massive power dissipation over the regulator as it stresses to bring the voltage down 18V. It is clear that the solution is to simply lower the input voltage to the system, which was achieved by using a 10:1 transformer at the wall rather than a 5:1 transformer. This leads to a much easier voltage step down for the regulator to handle, and as a result the system does not heat up as much and the system does not shut down. Even with the various heat sinking capability we had on the board, there was still a problem until the voltage input to the system was lowered. The 5V input still allows for the USB charging to occur at the bottom left of the board in its upright configuration as shown. The heat sinking provided in the top right of the board under the main regulator, as well as under the main micro, allows for some heat dissipation to occur.

10.4.4 Future Improvements

Throughout the semester, the project progressed very smoothly, and we were lucky to have no major setbacks throughout the semester. This came from very careful planning at each step, where the decisions were made months ahead of time to avoid any conflict or any time loss. At the beginning, it was understood that the most disastrous setbacks were likely to be PCB setbacks, simply because of the time cost associated with a board spin. In the case that a board was damaged or mistakenly wired, a replacement could take up to a week, which is a long time to be idle. Due to careful construction of the board layout, we never had a problem that shut the board down entirely, though the board did have small rewiring changes as discussed previously.

If I were to restart the project at this time, there are a few changes that could be changed in order to rectify several problems that we had as a team. First of all, the 5V power supply is strong enough to supply both the LED matrix as well as the micro and Bluetooth modules. In the current revision, it is set up so that the power supplies are separated and a 5V supply enters the supply separately through a completely different header. This problem could be simplified by just using the 5V header for everything. This also removes the heat problem of the regulator in the system. It actually removes the problem regulator entirely; the 3.3V and 1.2V regulator is more than enough.

Other problems that were addressed through the course of the semester that I would re implement include bypassing the board with the 5V supply for everything but level conversion. The board still needs the 5V for communication, but any power that runs to the LED matrix needs to run direct to the matrix.

10.5 Final Thoughts

Jonathan Lundstrom The last ten months have been packed full of challenges that needed to be overcome. Where we ended we have a relatively solid framework and code base. But, it is far from production ready code. More can be done to detect invalid operational states, loading software could be done over Bluetooth, and applications could be further abstracted so that apps can be compiled separately and loaded. Although none of these tasks were covered in an academic setting; I am confident all these tasks are doable with time and research.

Phillip Murphy Overall, main changes for the next iteration would be to completely redo the concept and massively increase the pixel density in the next iteration, and add an external clock crystal to the microcontroller to increase the clock speed to the max acceptable stable. Additional analog inputs such as an environmental microphone and/or classic component input from older systems would be versatile additions.

Christopher Rodrigue The project was a significant undertaking that tested the developers' abilities to integrate several subsystems together to produce a functional product. Bluetooth has unsurprisingly proven itself to be a reliable technology worthy of inclusion in future projects, and satisfied the wire-replacement needs of the smart table. For the next iteration of a smart table, it would be wise to include another wireless technology such as Wi-Fi Direct to broaden its feature set and integration with other IoT devices, especially if the product goes to market.

Ryan Mulvaney There is potential here for a product that is marketable to the general public. Overall, the project did very well in the senior design showcase, highlighted by our first place finish in the general event. It is interesting to see such a desire for a product, and there is some thought to be given to whether we move the device to market or retire it as a well performing senior design project. Because of the difficulty of manufacturing several parts of the project, including the table itself, there would need to be a large amount of thought given to overcoming several of the current design protocols to make such a move. However, it is entirely possible given some thought to move to market. Such considerations will be made at a later date.

Appendix A References

[Ale16] Ryan Alexander. *Laser Cut Coffee Table*. Oct. 23, 2016. URL: https://www.flickr.com/photos/onecm/7162032880.

[Ano16] The Anome. *Multiplexing Diagram*. Oct. 28, 2016. URL: https://commons.wikimedia.org/wiki/File:Multiplexing_diagram.svg.

[Atm] Atmel. *Product Page*. URL: http://www.atmel.com/Images/Atmel-2549-8-bit-AVR-Microcontroller-ATmega640-1280-1281-2560-2561_datasheet.pdf.

[Bla16] Rob Blanco. *Bluetooth Network Topology*. Oct. 31, 2016. URL: https://commons.wikimedia.org/wiki/File:Bluetooth_network_topology.png.

[Bur16a] Colin Burnett. *Daisy-Chained SPI Bus*. Oct. 31, 2016. URL: https://commons.wikimedia.org/wiki/File:SPI_three_slaves_daisy_chained.svg.

[Bur16b] Colin Burnett. *I2C*. Oct. 31, 2016. URL: https://commons.wikimedia.org/wiki/File:I2C.svg.

[Bur16c] Colin Burnett. *Typical SPI Bus*. Oct. 31, 2016. URL: https://commons.wikimedia.org/wiki/File:SPI_three_slaves.svg.

[Cad16] CadSoft. *Eagle PCB Design*. Oct. 16, 2016. URL: https://cadsoft.io/.

[Com16] Community. *Git*. Oct. 22, 2016. URL: https://git-scm.com/.

[Diga] DigiKey. *Online Catalog*. URL: http://www.digikey.com/catalog/en.

[Digb] DigiKey. *Tera Term Serial Port Settings*. URL: https://eewiki.net/display/Wireless/Getting+Started+with+RN42+Bluetooth+Module.

[Digc] DigiKey. *UART Commands in Ter aTerm*. URL: https://eewiki.net/display/Wireless/Getting+Started+with+RN42+Bluetooth+Module.

[Flo16] Marcin Floryan. *I2C Timing Diagram*. Oct. 31, 2016. URL: https://commons.wikimedia.org/wiki/File:I2C_data_transfer.svg.

[Git16a] Inc. GitHub. *GitHub*. Oct. 16, 2016. URL: https://github.com/.

[Git16b] Inc. GitHub. *Student Developer Pack*. Oct. 22, 2016. URL: https://education.github.com/pack.

[Goo16] Inc. Google. *Google Docs*. Oct. 16, 2016. URL: https://www.google.com/docs/about/.

[hub16] hubmartin. *Improved STM32 WS2812B DMA library*. Oct. 1, 2016. URL: http://www.martinhubacek.cz/arm/improved-stm32-ws2812b-library.

[Ins16] Texas Instruments. *TMS320F2837xD Dual-Core Delfino Microcontroller Data Sheet*. Oct. 16, 2016. URL: http://www.ti.com/lit/ds/symlink/tms320f28379d.pdf.

[Jor16] Jordsan. *SPI Timing Diagram*. Oct. 31, 2016. URL: https://commons.wikimedia.org/wiki/File:SPI_timing_diagram2.svg.

[Lab11] Mark Labbato. *RGB LED Coffee Table*. July 29, 2011. URL: https://e2e.ti.com/group/launchyourdesign/m/msp430microcontrollerprojects/447779.

[Lina] Jim Lindblom. *UART Interface*. URL: https://learn.sparkfun.com/tutorials/serial-communication.

[Linb] Jim Lindblom. *UART Timing Diagram*. URL: https://learn.sparkfun.com/tutorials/serial-communication.

[Meo] Renato Meola. *Comparison of Common Wireless Technologies*. URL: http://www.embedded.com/print/4016205.

[NAS] NASA. *Nasa-C-Style*. URL: http://homepages.inf.ed.ac.uk/dts/pm/Papers/nasa-c-style.pdf.

[Off16] Offnfopt. *OSI Model*. Oct. 31, 2016. URL: https://commons.wikimedia.org/wiki/File:OSI_Model_v1.svg.

[oom16] oomlout. *Laser Cut End Table*. Oct. 23, 2016. URL: https://www.flickr.com/photos/snazzyguy/6940883396.

[ST16] Inc. Slack Technologies. *Slack*. Oct. 16, 2016. URL: https://slack.com/.

[Stu16] ITEAD Studio. *HC-05 Specifications Wiki*. Oct. 16, 2016. URL: https://www.itead.cc/wiki/Serial_Port_Bluetooth_Module_(Master/Slave)_:_HC-05.

[T.Ia] T.I. *Product Page*. URL: http://www.ti.com/product/MSP430G2553.

[T.Ib] T.I. *Product Page*. URL: http://www.ti.com/product/MSP430FR6989.

[T.Ic] T.I. *Product Page*. URL: http://www.ti.com/product/MSP430F6659.

[T.Id] T.I. *Product Page*. URL: http://www.ti.com/product/MSP432P401R.

[T.Ie] T.I. *Product Page*. URL: http://www.ti.com/product/TMS320F28379D.

[Tat16] Simon Tatham. *Putty*. Oct. 16, 2016. URL: http://www.chiark.greenend.org.uk/~sgtatham/putty/.

[Tec] Nasa Technical. *Nasa - Technical Documentation*. URL: https://www.sti.nasa.gov/.

[Tec16] Microchip Technology. *RN-42/RN-42-N Data Sheet*. Oct. 16, 2016. URL: ww1.microchip.com/downloads/en/DeviceDoc/50002328A.pdf.

[TI16] Incorporated Texas Instruments. *Code Composer Studio*. Oct. 16, 2016. URL: http://www.ti.com/tool/ccstudio.

[Tre16] Inc. Trello. *Trello*. Oct. 16, 2016. URL: https://trello.com/home.

[var] various. *Recommended C Style and Coding Standards*. URL: http://www.maultech.com/chrislott/resources/cstyle/indhill-cstyle.html.

[Wik] Wiki. *ProDesktopDrawing*. URL: https://upload.wikimedia.org/wikipedia/en/b/ba/ProDesktop_Drawing.PNG.

[wik14] wiki. *WS2811 Clock Shaper*. Mar. 12, 2014. URL: http://wiki.artifactory.org.au/doku.php?id=projects:ws2811clockshaper.

[WS16] World-Semi. *WS2812 Data Sheet*. Oct. 16, 2016. URL: https://cdn-shop.adafruit.com/datasheets/WS2812.pdf.

[Zan16] Benito van der Zander. *TeXstudio*. Oct. 16, 2016. URL: http://www.texstudio.org/.

Appendix B Permissions

Brian Senese
Field Applications Engineer - Automotive

Nov 5, 12:35 PM

Permission to use figures from paper

Hello Mr. Senese,

I am an electrical engineering student enrolled in senior design at the University of Central Florida. I am writing to request your permission to use some figures from your paper "Implementing Bluetooth Wireless Technology In An Embedded Device" (https://scholar.google.com/scholar?cluster=12632269517449322 4&hl=en&as_sdt=0,10) in a design paper that will not be published.

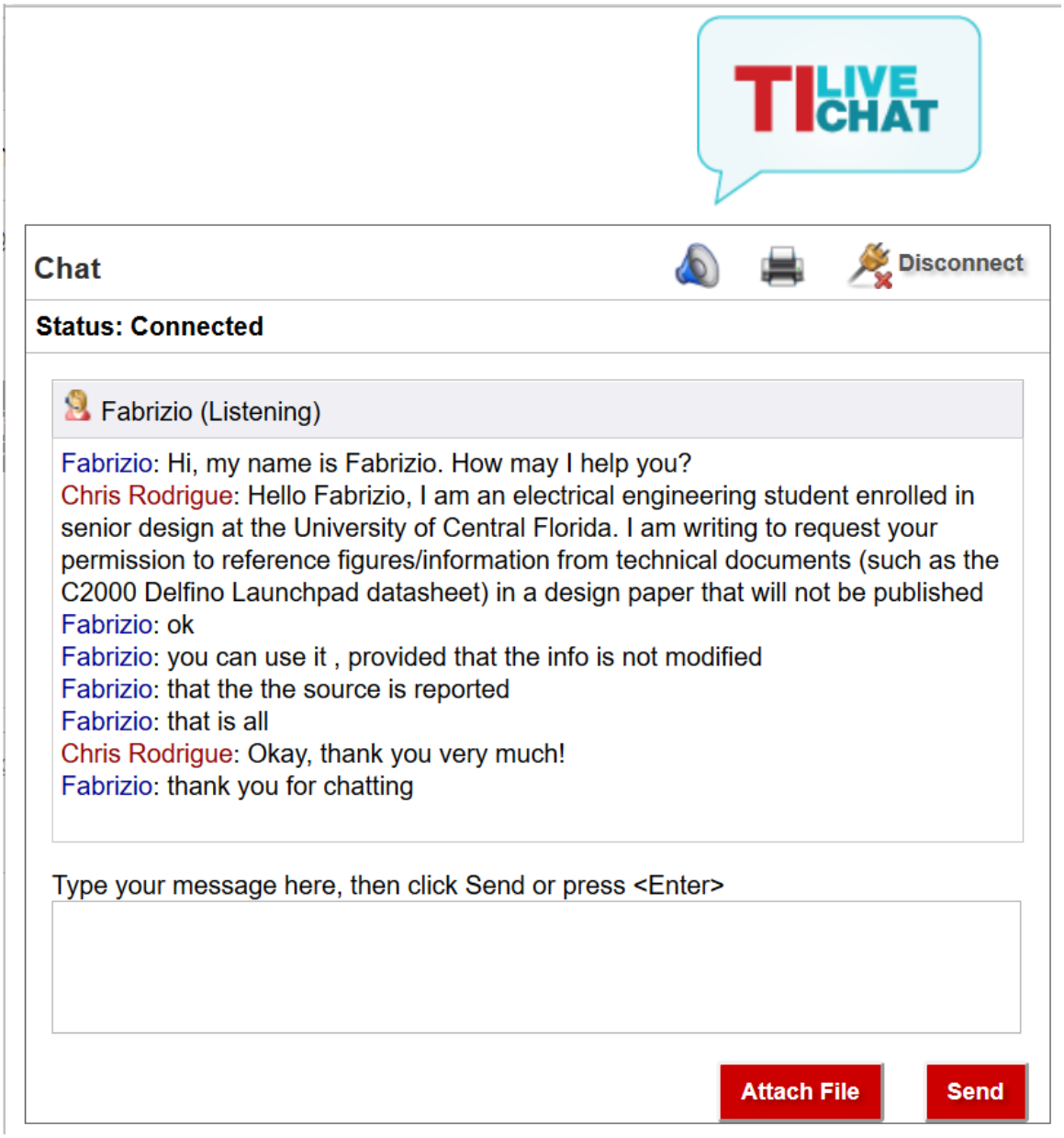
Thank you very much,

Chris Rodrigue
(407) 569-9802
rodrigue@knights.ucf.edu

Chris -
go ahead.... I am surprised that paper is still circulating... it is an oldie but a goodie.

Best Regards,
Brian

9:33 PM


TI LIVE CHAT
Chat
Disconnect
Status: Connected
Fabrizio (Listening)
Fabrizio: Hi, my name is Fabrizio. How may I help you?
Chris Rodrigue: Hello Fabrizio, I am an electrical engineering student enrolled in senior design at the University of Central Florida. I am writing to request your permission to reference figures/information from technical documents (such as the C2000 Delfino Launchpad datasheet) in a design paper that will not be published
Fabrizio: ok
Fabrizio: you can use it , provided that the info is not modified
Fabrizio: that the the source is reported
Fabrizio: that is all
Chris Rodrigue: Okay, thank you very much!
Fabrizio: thank you for chatting
Type your message here, then click Send or press <Enter>
Attach File
Send

Permission to use figures/information from technical documents

Chris Rodrigue <chris.rdrg@gmail.com> 7:53 AM (7 minutes ago)

to docerrors, university

Hello,

I am an electrical engineering student enrolled in senior design at the University of Central Florida. I am writing to request your permission to reference figures/information from technical documents (such as the RN42 Bluetooth module datasheet) in a design paper that will not be published.

Thank you very much,

Chris Rodrigue
(407) 569-9802
rodrigue@knights.ucf.edu

Permission to use figures from Fab Academy

Chris Rodrigue <chris.rdrg@gmail.com> Nov 5

to vt.y

Hello Mr. Vincent,

I am an electrical engineering student enrolled in senior design at the University of Central Florida. I am writing to request your permission to use some figures from your website Fab Academy (such as http://archive.fabacademy.org/archives/2016/fablabshangai/students/96/Week15/one-to-many%20connection.png) in design paper that will not be published.

Thank you very much,

Christopher Rodrigue
407.569.9802 | chris.rdrg@gmail.com

Renato Meola
Director of Marketing

Nov 8, 10:56 AM

Permission to use information from paper

Hello Mr. Meola,

I am an electrical engineering student enrolled in senior design at the University of Central Florida. I am writing to request your permission to use and modify some information with attribution from your paper "Improvements in IR communication push it beyond cell phones and PDAs" (http://www.embedded.com/print/4016205) in a design paper that will not be published.

Thank you very much,

Chris Rodrigue
(407) 569-9802
rodrigue@knights.ucf.edu

Permission to use figures/information from technical documents

Chris Rodrigue <rodrigue@knights.ucf.edu> 8:20 AM (0 minutes ago)
to pr, flavio.stiffan

Hello,

I was referenced to you from a chat support specialist, Eva.

I am an electrical engineering student enrolled in senior design at the University of Central Florida. I am writing to request your permission to reference figures/information from technical documents (such as the Bluetooth to UART interfacing document here: http://www.nxp.com/documents/application_note/AN10307.pdf) in a design paper that will not be published.

Thank you very much,

Chris Rodrigue
(407) 569-9802
rodrigue@knights.ucf.edu

On Fri, Dec 2, 2016 at 8:42 AM, aimee@world-semi.com <aimee@world-semi.com> wrote:

Dear Chris,

Nice to contact with you and thanks for your keen interest in our products.

We are happy to know that you are interested in WS2812 series and herein we give you permission you can use all full list of technical parameters in you design paper.

By the way, may I know that what's you designing?

Should there be anything I can do for you, please feel free to let me know.

Hope everything goes well with you and your business.
Have a Nice Day...!

Thanks & Best Regards!

Tel: 86-755-27809527 ext.6008 | Mob: 86-13926507059

Skype: shaoqinggirl

Backup email: aimeechueng@foxmail.com

Web: http://www.world-semi.com/en

WORLDSEMI CO., LIMITED

From: Chris Rodrigue
Date: 2016-12-02 00:23
To: aimee
Subject: Permission to use figures/information from technical documents

Hello,

I am an electrical engineering student enrolled in senior design at the University of Central Florida. I am writing to request your permission to reference figures/information from technical documents (such as the WS2812 datasheet) in a design paper that will not be published, with proper attribution to Worldsemi.

Thank you very much,

Chris Rodrigue
(407) 569-9802
rodrigue@knights.ucf.edu

Your Name:

Ryan Mulvaney

Your Email Address:

ryan.mulvaney@i-con.com

Subject:

Becoming an Astronaut
Careers at NASA
Doing Business With NASA
Educator Resources
International Space Station
Educational Opportunities
Mars Program
NASA History
NASA Images and Videos
Permission to Use NASA Images
Solar System
Student Resources

Question or Comment:

Good Evening,

My name is Ryan Mulvaney, and I am an electrical engineering student enrolled at the University of Central Florida. I am currently working on a senior design project, and for part of this project I would like to reproduce some pictures from a document on your website. The document can be found here:

https://nepp.nasa.gov/docuploads/06AA01BA-FC7E-4094-AE829CE371A7B05D/NASA-STD-8739.3.pdf

I am writing to request permission to refer to various pictures in the technical document that I am working on.

Please get back to me at this email, ryan.mulvaney@i-con.com or at 407-621-1872.

Thanks for your help,
Ryan Mulvaney

Permission to replicate

Ryan Mulvaney
Thu 12/1, 1:58 PM
kevin.brown@digikey.com

Reply all

Sent Items

Show all 1 attachments (191 KB) Download Save to OneDrive - I-CON Systems Holdings, LLC

MyAnalytics

Good afternoon,

My name is Ryan Mulvaney, and I am an electrical engineering student enrolled at the University of Central Florida. I am currently working on a senior design project, and for part of this project I would like to reproduce some pictures from the “images” section of your website. I am writing to request permission to refer to these pictures in the technical document that I am working on. For example, when searching RN-42, I would like to use the attached picture in my technical document.

Please get back to me at this email, ryan.mulvaney@i-con.com or at 407-621-1872.

Thanks for your help,
Ryan Mulvaney

Jim Lindblom
Electrical Engineer / Content Creator at Sparkfun Electronics

Permission to use information from tutorials/articles

Hello Mr. Lindblom,

I am an electrical engineering student enrolled in senior design at the University of Central Florida. I am writing to request your permission to reference figures/information from some of your SparkFun articles (such as Bluetooth Basics and Serial Communication) in a design paper that will not be published.

Thank you very much,

Chris Rodrigue
(407) 569-9802
rodrigue@knights.ucf.edu

Hi Chris,

Thanks for checking. Feel free to use any of the content in our tutorials, it should all be available under the CC BY-SA 4.0 license (https://creativecommons.org/licenses/by-sa/4.0/).

Cheers,
-Jim

2:43 PM