# The Memrowave

*A UCF Senior Design Project* *Spring 2015*

***Group 31***

Winston Todd

Jack Gulick

Joseph Serritella

Darren Armstrong

# Table of Contents

# List of Figures

# List of Tables

# 1 Executive Summary

The Memrowave is a UCF engineering Senior Design project for the Fall 2014-Spring 2015 semesters. The main purpose for this project was to make a smarter and more autonomous microwave capable of cooking multiple foods with ease and limited user input. Typical microwaves incorporate the use of tactile buttons and segmented displays to relay information to the user. The user is then required to stand-by and monitor the cooking process. Some of the most common issues with this process are the proper cooking and consistency of the food. The Memrowave has been demonstrated to overcome these issues.

The Memrowave was designed to be a microwave that is more connected and autonomous. An automatic timer was incorporated into the system which makes the user input much simpler and more efficient. Through the use of bar code scanning, cooking configurations based on the code of the food package is sent directly to the Memrowave, allowing it to begin the cooking process once the door is closed. This approach eliminates the hassle of identifying proper cooking times based on different wattage microwave appliances. Along with bar code scanning, a food product database was setup to log all the specific information needed to cook foods. The already established UPC code acts as an index to identify the product.

The connection to the internet was implemented through WIFI, in addition to using this connection for the web database, the Memrowave has the ability to send notifications to the user's phone or mobile device, which allowed the user to roam as the food is being prepared. The main user interface is a LCD touchscreen on the front of the microwave. The user is able to access all the features of the Memrowave, to use it as a regular microwave, or to use the smart features.

Other senior design teams and appliance manufacturers have made efforts to make appliances smarter by creating mobile applications that can serve as a remote control for appliances such as washing machines, and dishwashers to report status changes and when the user's input is needed. When it comes to smart microwaves, most enhances in the industry are done through the ability to having touch-type interfaces and more specific and numerous preset configurations to cook multiple foods. Memrowave was able to add to these enhancements and also be an internet connected device which simplified the use.

The basic components needed to design the Memrowave included the magnetron, internal lights, internal beeper, power switch, turntable motor, and power supply. The more complex components included the processing unit, printed circuit board, camera module, WIFI module, and LCD touchscreen. Divisions between hardware and software form the basis for the design plan:

powering devices, power operation of switches, PCB assembly, and final enclosure assembly are in the hardware side of the design, controlling of the multiple peripherals, constructing the user interface, web database development, and bar code scanning implementation are in the software side of the design. Integration of both the hardware and software was the final step when we were developing the Memrowave.

## 1.1 Realistic Design Constraints

Products designed for use in today's society require many types of constraints which they must adhere to. These project level design constraints include, but are not limited to; Economic, Environmental, Social, Political, Ethical, Health and Safety, Manufacturability, and Sustainability. Chapter 2 discusses the relationships between the Memrowave and these Constraints.

## 1.2 Economic

When fully prototyped and released the Memrowave will be purchasable by anyone with the resources available. The purchaser will only be required to obtain the necessary hardware in the prototype. All software is open source, therefore obtainable for free. This will increases the economic flow into industries related to IC manufacturing and other electrical engineering companies.

## 1.3 Environmental

The Memrowave makes use of pre-purchased microwave ovens, which makes building a Memrowave efficient and easy. Since the microwave used is pre-purchased the environment will see less microwave scraped. Old microwaves can easily be repurposed as Memrowave's saving space in landfills and etc.

## 1.4 Social

When fully prototyped and released the Memrowave will be purchasable by anyone with the resources available. The purchaser will only be required to obtain the necessary hardware in the prototype. All software is open source, therefore obtainable for free. This will help add to the general populations need for more "smart devices" and appliances. The Memrowave serves to teach new users technology because it requires the integration of open source software and the purchased hardware

## 1.5 Political

The Memrowave will have little to no possible effect on the political aspects of the world. The design seeks to only benefit the general user and their need for smart devices.

## 1.6 Ethical

Because the Memrowave's software is open source, therefore obtainable for free. A portion of the general population will see it as an unfair way to avoid paying for software. This may be negative for some users however, we do not agree with this stance. Also since the Memrowave is a "smart device" or appliance and its goal is to make a simpler task simpler. This may be seen as a lazy way out however it is as an ethical project because it is the user's decision to use the Memrowave.

## 1.7 Health and Safety

The Memrowave requires the successful modification of a microwave oven. Microwaves are extremely dangerous because microwave radiation emitted during operation. Microwaves are considered safe because of their ability to trap the RF energy inside of the cooking area. However if energy is leaking out of the cage serious health risks will ensue. When Building the Memrowave the user must be careful and monitor the power of the microwave's magnetron. The user must check for any areas that make leak RF energy. Along with possible RF exposure the user must be alert at all times. The microwave runs at 120V with a lager current to the magnetron's transformer. If shocked a person may be seriously injured. The Memrowave can be extremely dangerous but with carefully consideration and use no health risks or safety issues should be encountered.

## 1.8 Manufacturability

When fully prototyped and released the Memrowave will be purchasable by anyone with the resources available. The purchaser will only be required to obtain the necessary hardware in the prototype. All software is open source, therefore obtainable for free. This provides no need for manufacturing of the product. All manufacturing related issues are directly related to the sources of hardware companies like; Texas Instruments, Panasonic, and Sharp.

## 1.9 Sustainability

Upon the release of the Memrowave all the necessary hardware and software is obtained by the user. The software used is all open source, therefore obtainable for free. The products sustainability is not considered because it is assumed all part are easily obtainable and replaceable. All quality related issues are directly related to the sources of hardware companies like; Texas Instruments, Panasonic, and Sharp.

# 2 Project Description

In this section, we will describe the project in detail, including our objectives, requirements, specifications, and motivation for designing the Memrowave.

## 2.1 Project Motivation

Today's society calls for ever more growing technologies. Consumers demand up-to-date devices that push the envelope of what we can achieve through technologies. These Technologies provide us with revolutionary devices that shape the world, some make the hardest of tasks simple with the push of a button. The ideal to make everyday tasks quicker and more efficient is the main driving force behind our design the Memrowave. The Memrowave seeks to take a common household appliance and create a better more innovative appliance that is smart, fast, and just plain simple for everyone to use. The e Memrowave allowed the user to quickly set up his morning meal or dinner on the go with a quick barcode scan, having the user spend much less time around the kitchen. In order to create this appliance of the future there are certain task to be completed; The Memrowave must learn the cook times for each food, it adapted to the preferences of the user, it alerted the user to completed food cool down times.

In order for the Memrowave to become a "smart: microwave it must first gather the prospective cook times of each food the user may want to consume. There are many possible ways this can be achieved however, we focused on two methods. The first method is to manually store the data containing information such as cook time to the memory of the device. Once the barcode is scanned the Memrowave retrieves the information stored in memory. The second method searches an online database. The database was created as an online database in which the Memrowave accesses through Wi-Fi. Our team can manually store initial products to the online datasheet. This contains the product UPC, cooking times description, power level, and product image. Once the device is scanned the Memrowave will retrieve information via Wi-Fi through the webpage created by our design team. The product information is then used to control the hardware and power supplied. This Online database was created also will have the ability to edit and write to. For example if the barcode cannot be found in the database, it will be saved. The user is prompted to input cooking time, cool down time, power level, and verify correctness. This information is saved and ready for use next time the new barcode is scanned.

The Memrowave allows the user to change the stored cook times previously used. For example, if a meal was deemed to be overcook a user can alter the timing in arbitrary integer increments to achieve the desired taste. This information will then be stored as the new time for later usage. This option will be referenced for a particular item every time that item is processed.

The next task of the Memrowave is allowing the user to not be present during cooking of the product. Utilizing the microwave's ability to access the internet, we create notifications via text messaging. This system pings the user once a product has completed its cook and cool down cycles. This feature keeps the microwave new and innovative compared to standard units. Texting provides the user with the ability to continue on with daily tasks while avoiding waiting for the microwave. The system is particularly affective when a user is out of hearing range of the Memrowave.

Eating provides a person with the energy needed to focus throughout the day however, over eating can be detrimental to a person health. This is where the Memrowave comes in. It can function as a daily calorie counter. Information about a product's nutrition is stored online, which the Memrowave can retrieve and track. This feature isn't supposed to stop the microwave from cooking once it reaches a boundary condition but, it will simply display the amount consumed for the day thus helping the user make a healthy choice.

Lastly, a product can only successful if and only if the mass consumer finds it simple, appealing, and has ease of use. The Memrowave seeks to make this a key aspect of its design by integrating a seamless LCD touch screen as the user interface. The LCD with display all the features mentioned previously such as the barcode scanner, the clock, the calorie counter, and user adjustable controls to change times. When implanted into the Memrowave navigation of the menus will be a breeze, thus link the whole thing together.

The Memrowave seeks to be the next must have house hold appliance. The Memrowave came into design because we wanted to make preparing our meals even simpler. It draws on the ideal that simpler is better and time is important. These goals are achieved by adapting to the preferences of the user, text message alerts to the user of completed food times, a seamless LCD touch screen user interface, and a barcode system that cooks our food for us.

## 2.2 Objectives

There are a number of objectives we will be trying to achieve with the Memrowave. In this section, we will describe each of these objectives in detail.

### 2.2.1 Automatic Timer

The main focus of the Memrowave is to prepare food simpler and faster for the user on the go. Automatic timing cuts out all the user thought on food preparation. The user simply scans the barcode of the product and closes the door. Once closed the user won't have to stress over reading instructions and punching in numbers because the Memrowave does that for him/her. This Automatic timing is what sets our product apart from standard microwaves.

Timing is begun once the Memrowave obtains the cooking information (this is discussed in more detail in a later chapter) and actives the radio frequency (RF) power. Power is kept on until the automatic timer reaches the desired cook time. Timing of products can even be taken a step further by achieving a cool down timer once the first timer reaches zero. This cool down time won't be user operated either thus making it the second automatic timer on the microwave. The function of this automatic timer is cooling of recently cooked products. This information can also be found online for a given product.

Displaying of the automatic timer is accomplished through a 7-segment LED interfaced onto the front of the microwave (this is discussed in more detail in a later chapter). The timer starts at the top of the cook time for the product and count down till zero. Timing is kept by using a microcontroller. The microcontroller will deactivate the RF power once the automatic timer has reached zero. From an overhead view the timer, magnetron, and microcontrollers interact together as depicted in Figure 2.2.1-1.

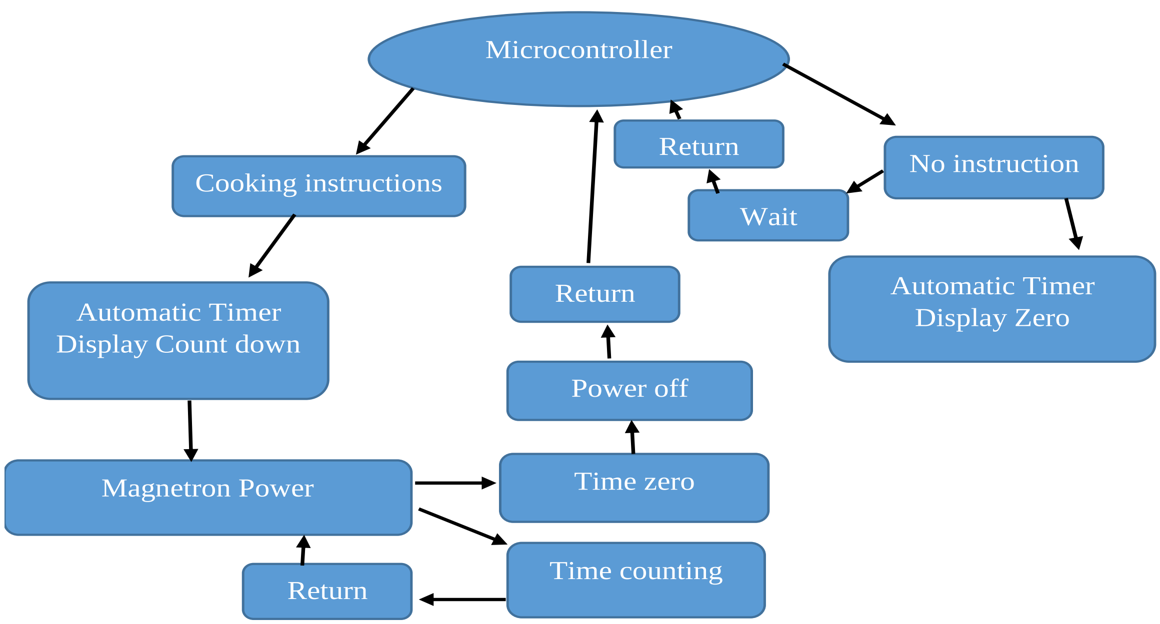


**Figure 2.2.1-1:** Automatic timer flowchart

## 2.2.2 Food Product Database

The Memrowave includes a database of all of the food products that it knows how to cook. This was accomplished using a database that indexes food products based on their UPC, or Universal Product Code. The UPC is convenient because most food products already include a unique UPC that is easily scannable, that is, the barcode.

The food product database stores all of the relevant information for a particular food product, including the UPC, name, description, preparation instructions, cook time, and power level. This information had to be manually entered into the database for each product. Suppose the user scans a food product that is not in the database. The user must manually enter the cook time and power level, but they don't have to do this more than once for a particular product. The Memrowave can remember this information for the next time the user scans this product. This was accomplished by storing the information in the product database.

Sometimes the cook time provided by food manufacturers isn't optimal for a particular microwave or altitude. The food may come out still cold, or it may be a little burned, etc. Because of this, the Memrowave provides the option for the user to modify the cook time of any product. The user's modification is stored in the database and given priority over the manufacturer's information.

#### *2.2.2.1 Cook Times and Power Levels*

In this section we will discuss in detail the process of obtaining cook times and power levels. In order to properly cook product inside a microwave the user must follow the instructions labeled on the package. These instructions are as follow; cooking time, power Level, and cooling down time. While obviously a simple task for all people to complete it is the Memrowave's goal to get rid of this step.

Once stored in the food product database the cook times and power levels will be accessed next time they are needed. In order to access the food product database the user must scan the barcode of the product. The barcode binary is then matched with the information stored in the food product database thus retrieving the cook time, power level, and cool time. The information is then used to have the microcontroller set up and start the automatic time and power the magnetron. Once the cooking cycle is complete the system is powered down and the product is ready to be removed

Proper cooking of many products requires changes in the RF power delivered to the system. It is one of the goals of the Memrowave to regulate this in order to best cook foods. Once a product's information is stored in the food product database we can refer to a given product's power requirement. This requirement will be passed to the microcontroller which will then be feed through a system to regulate and control the power output of the microwave. This control system will be built on a Printed Circuit Board (PCB) which is discussed in more detail in a later chapter. This system can be modeled as shown in Figure 2.2.2.1-1.

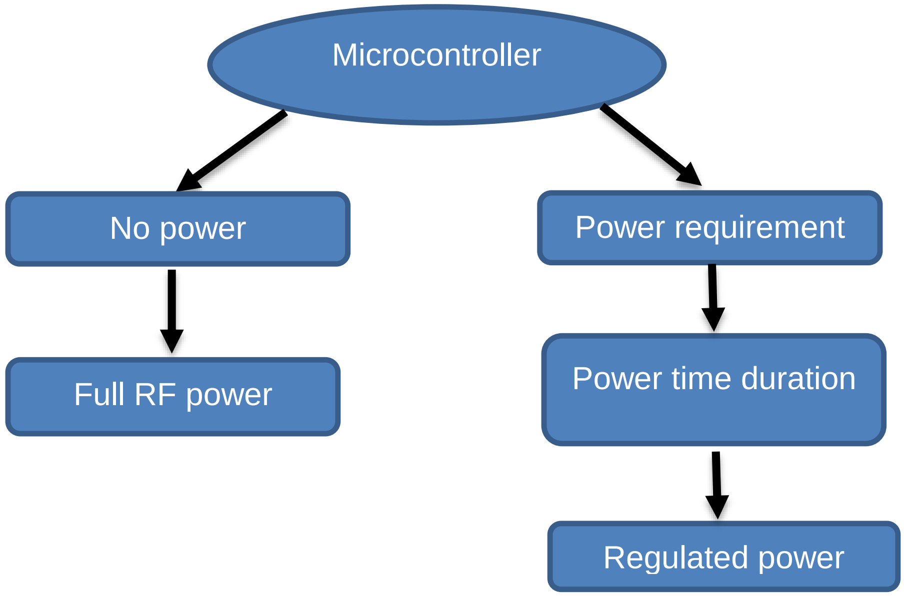


**Figure 2.2.2.1-1:** Power level flowchart

### *2.2.2.2 Preparation Instructions*

Operation instructions describe the step by step process for the user to utilize the Memrowave while Preparation Instructions refer to microwave recipes the Memrowave will store for the user.

Operation instructions for the Memrowave are simple and very quick to follow; here is a step by step process for Operation. Frist the user begins by holding up a products barcode to the camera on the Memrowave. The LCD screen displays the image in real time to aid in the placing of the barcode. Once the Memrowave scanned the barcode a time is displayed and the user is prompted to close the door of the microwave. Once closed the LCD screen will display start and the cooking process begins. This period of time will allow the user to leave the microwave and carry on with other tasks during the waiting period. When the cooking is finished, the user will receive a text message informing that the time is up. This text is sent through the Memrowave using Wi-Fi. When the user opens the microwave door the LCD will display a question asking if the user would like to add or subtract 5 seconds from the cook time. If yes the user can adjust the time to his/her desire. Finally the microwave will return into the stand-by position until the user returns to cook more. Overall operation must contain a minimum amount of steps to make the Memrowave as simple as possible to use with much less time spent in front of it than a standard microwave.

If a product isn't currently saved the Memrowave will proceed as follows. The user is notified from the LCD that the product isn't saved and requires internet access to obtain the information. Once the information is retrieved the user is notified and the cooking process will begin as normal. However, if there is no information about the cooking instructions the user is prompted to store the information personally.

## 2.3 Project Requirements and Specifications

In the previous section, we discussed our objectives for the Memrowave project. In this section, we will outline the specifications that must be met to achieve our objectives for the project.

### 2.3.1 Automatic Timer and Power Level

After a user scans a product's barcode, the Memrowave must automatically set the timer and power level, provided the Memrowave has access to this information. If, for whatever reason, the Memrowave cannot determine the appropriate cook time or power level, it will ask the user to enter this information; however, this should only occur once, because the Memrowave stores this information for the next time the product is scanned.

### 2.3.2 Local Product Database

The Memrowave can function without access to an internet connection. This was achieved by including a local product database. This database stores all of the relevant information for a particular food product, including the UPC, name, description, preparation instructions, cook time, and power level.

When a user scans a food item using the barcode scanner on the Memrowave, it will first search the local database for an entry with a matching UPC. If the UPC is not found in the local database, the Memrowave will then attempt to connect to the internet and retrieve this information from the master product database, which was made available through a web API. If the product cannot be found in the master database or if the Memrowave does not have internet access, then the user will be asked to manually enter the cook time of the product. This manual entry will be saved in the local product database for later use. A flowchart of this process is shown in Figure 2.3.2-1.

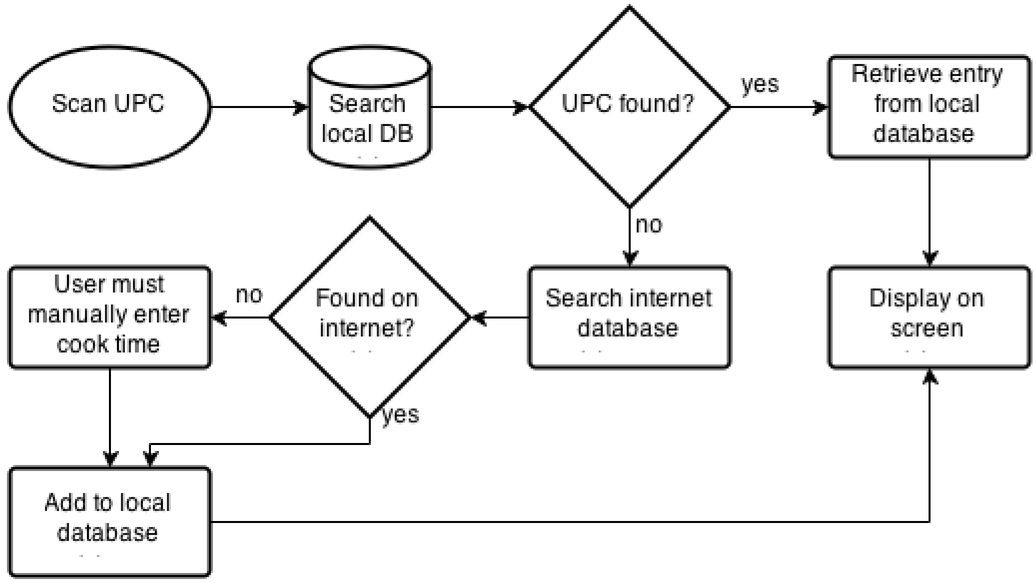


**Figure 2.3.2-1:** Database flow chart

### 2.3.3 Touch Screen Interface

We wanted the Memrowave to feel like a real smart device, so it had to have a touchscreen interface. The touchscreen is large enough to read without straining, and it is large enough for the user to easily interact with the software; however, it is still small enough to fit into the control panel of the microwave. An LCD in the range of 4” to 7” diagonally achieved the size requirements.

The LCD is backlit, so it can be seen clearly in low light, and the brightness is adjustable by the user. Both the display interface and the touch interface are compatible with the ARM microcontroller we chose to run the Memrowave.

The software interface has a clean, ‘smart’ look. It is easy to use, and the control elements are large enough to select easily on the touchscreen. It is quick and responsive, providing feedback whenever it is loading or processing so that it never appears to be frozen.

The interface has a well-defined, intuitive layout. The software provides an interface for every available feature; the user doesn’t have to navigate any external piece of software. All of the Memrowave’s settings are accessible from one central location within the software. The settings are stored on non-volatile storage, so that they do not reset after a power-cycle.

### 2.3.4 Barcode and QR Code Reader

The Memrowave uses a camera to scan product barcodes. This was accomplished using the ZBar barcode scanning library. The barcode scanning is relatively quick, and accurate. It is able to read mostly any barcode, regardless of any differences in colors, packaging, etc.

### 2.3.5 SMS Notifications

With a normal microwave, the user must wait within earshot of the microwave to hear when it is finished. With the Memrowave, we enable the user to know exactly when the microwave finished cooking, even out of earshot. This was achieved this by the implementation of an SMS notification feature.

The SMS notification feature allows the Memrowave to send text messages to the user’s cell-phone. These messages kept the user informed about the state of the Memrowave as it cooks the food. For instance, if the user is defrosting some food, the Memrowave sends an SMS notification instructing the user to flip over the food to defrost the other side.

This feature is used to notify the user when their food has finished cooking, when the food’s cool down time has been reached, when to stir the food, etc. Basically, this feature is used to send any message from the Memrowave to the user.

### 2.3.6 Manual Operation

Sometimes a user may not want to use the smart features of the Memrowave, or the smart features may not work well for certain types of cooking and recipes. For cases such as these, standard manual operation features are necessary, so the Memrowave provides these features.

The manual operation features include basic cook time selection, start and stop buttons, power level selection, and any of the basic features you might find on a standard microwave. These features are described in detail in section 3.1.3.

# 3 Research

Before designing any systems of the Memrowave, we had to research a number of things pertaining to the project, including similar projects and products and the relevant technologies that we would use to design the Memrowave.

## 3.1 Similar Projects and Products

The Memrowave is not entirely unique, but every product builds off its predecessors. In this section, we will be researching products and projects that are similar to the Memrowave to gain a better understanding of how we will design the Memrowave.

### 3.1.1 Retail and Other Smart Appliances

Most microwaves deemed smart by its manufacturers set these products apart from the rest by adding features to the microwave rather than changing its primary functions. Instead of physical buttons, touch-type buttons are used. Power levels are much more graduated. The microwave is transformed into a device made to cook all types of meals with different preset configurations to make it act as a grill, fermentation device, vegetable steamer, and defroster. Some of the microwaves are also self-cleaning, a feature usually for the high-heat capacity of a conventional oven. Many of these features we want to add to our smart microwave, but we want the Memrowave to also be more connected.

More of these connected features are used in smart refrigerators, washers and dryers, and dishwashers. Whirlpool has implemented in its smart appliances what is called 6th Sense Live Technology which allows the users to manage and control these devices. Energy usage, time left on machines, and the condition of all your appliances are available to you through a phone application. The smart appliances application also makes use of cellular push notifications. Samsung has also implemented touch screens into their refrigerators for temperature and other controls. These types of systems seem to be implemented where the appliance is a stand-alone autonomous machine, giving the user status reports or whether some action is needed by the owner.

Most smart appliances make use of touch-enabled interfaces, WIFI, and mobile device applications. Device which can be left alone for hours or are more autonomous than a microwave benefit from those features, these include washers and dryers, air conditioning units, and water heaters.

Other smart technologies include also include system diagnostics, error detection, system failure reports. These can also implemented with a connected system to keep the user more aware of the condition of their appliances. Whirlpool in their Energy Smart water heater has uses system diagnostics to check for dry fire, upper or lower element condition, upper or lower thermistor condition, water temperature limits, and thermostat functionality. These diagnostics provide error codes which are readable on the LCD screen of the user interface; there is not a mobile solution to accessing such information, which we would like to implement in the Memrowave.

### 3.1.2 Previous Senior Design Projects

A project similar to what we plan on implementing was created developed by a Senior Design group in the fall 2013 - spring 2014 semesters. The project is called N.O.M.S., the Nutritional Object-identifying Microwave System. It “takes the idea of a traditional microwave and advances the technology to be on par with technology today.” This project for a smart microwave incorporates many of the “smart” ideas we wish to incorporate into our project. N.O.M.S. incorporates the use of a camera as a QR code reader to scan barcodes that contain the information for cooking times and power levels. A touch screen on the microwave is used as the main interface. The LCD touch screen was used to display the time, for UPC code scanning, and proving the status of the product while cooking. For storing the information necessary for cooking configurations, they used the caching system for short term storage then if it was not stored in the cache then a web search would be executed. WIFI was used to provide an internet connection to the microwave, which was configurable through PC or mobile software.

Comparing the N.O.M.S. project to Memrowave, the idea of a more connected appliance is the ultimate goal while the implementation and overall features have many similarities and differences. Both will use a touchscreen as the main user interface for accessing all the features, a camera will be used to collect bar code information on cooking configurations, but different systems for reading and saving cooking configurations are being used, N.O.M.S. used the bar code as a ID tag to go and search a local and/or web database for the cooking configuration, with the Memrowave, the entire cooking information configuration will be contained within the scannable code. The use of WIFI will be mainly the same, as a connection to the internet for Memrowave’s web database. In terms of other features, the N.OM.S. project added an advertising scheme on the LCD screen that shows advertising from manufacturers, consumer specific coupons, other forms of entertainment that would be shown as the food was being cooked, with the Memrowave, phone or mobile device notifications through either Bluetooth, WIFI, or SMS mobile messaging is a feature being considered.

### 3.1.3 Current Microwave Interfaces and Functionality

Before beginning the design for the smart microwave interface, it is important to recognize that not all functionality will be able to be met by the smart features. Many food items that a consumer may want to prepare in a microwave, for example baked potatoes and restaurant leftovers, will not come with a barcode. Therefore manual features and controls must be provided for the smart microwave in order for it to provide a complete solution. To understand what features and controls exist on current microwaves, two existing microwaves were examined and their features were evaluated for inclusion in the features set of the smart microwave.

#### *3.1.3.1 Microwaves Examined*

The first microwave evaluated was a Whirlpool over-the-range microwave hood combination (model MH1150MXT-2). This microwave was inexpensive enough that it was used in new home construction, but has more than just basic features. Physical features include a size of 1.5 cubic feet, 1000 watts cooking power, a turntable, and a cooking rack. The microwave includes many quick control features such as Popcorn, Baked Potato, Pizza, Defrost, and Reheat. Figure 3.1.3.1-1 shows the Whirlpool control panel.

The second microwave used for evaluation was a Panasonic Genius Inverter microwave (model NN-H965X ABH). This microwave is a mid- to upper-range countertop microwave. Physical features include 2.2 cubic feet interior, 1250 watts cooking power, and a temperature sensor that can be utilized during cooking or reheating food. The microwave includes quick control features such as Popcorn, Defrost, Keep Warm, Sensor Reheat, and Sensor Cook which covers nine commonly microwaved foods. The Panasonic control panel is shown in Figure 3.1.3.1-2.


**Figure 3.1.3.1-1:** Whirlpool control panel


**Figure 3.1.3.1-2:** Panasonic control panel

### *3.1.3.2 Basic Microwave Functionality*

The basic control functionality of a microwave is as follows:

1. Select the amount of time to cook the food
2. Optionally, select the power level to cook the food at
3. Start the microwave

As simple as this basic functionality is, the microwaves being examined vary in their implementations. The Whirlpool has separate Cook Time and Cook Power buttons while the Panasonic does not have a distinct cook time button. Setting cook time on the Whirlpool microwave can be accomplished in two ways. The first method is to select the Cook Time button followed by the appropriate numbers on the control panel number pad. The second method assumes that no other buttons have been pressed, i.e. the microwave control panel is in its idle state. With this method, pressing the numbers on the control panel number pad is all that is required. In both cases the power level is displayed as a small number to the right of the input cooking time. Setting the power level starts with pressing the Cook Power button, but diverges to two methods after that point. The user can either keep pressing the Cook Power button to reduce the power in steps or the user can enter the power directly using the control panel number pad. The user can also do both or neither. The Whirlpool microwave basic functionality is shown in Figure 3.1.3.2-1.

The Panasonic microwave takes a narrower approach. There is no cook time button. Number keypad presses always results in a change in the cook time. This can occur because at no time is the control pad number pad used to enter the cook power. Cook power is only adjusted by repeatedly pressing the Power Level button. The result of this design decision is both good and bad. The good result is there is less opportunity for a mistake such as pressing a number pad button that unintentionally changes the power level. The negative consequence is there is less flexibility in how the user inputs the cooking parameters. The Panasonic microwave basic functionality is shown in Figure 3.1.3.2-2.

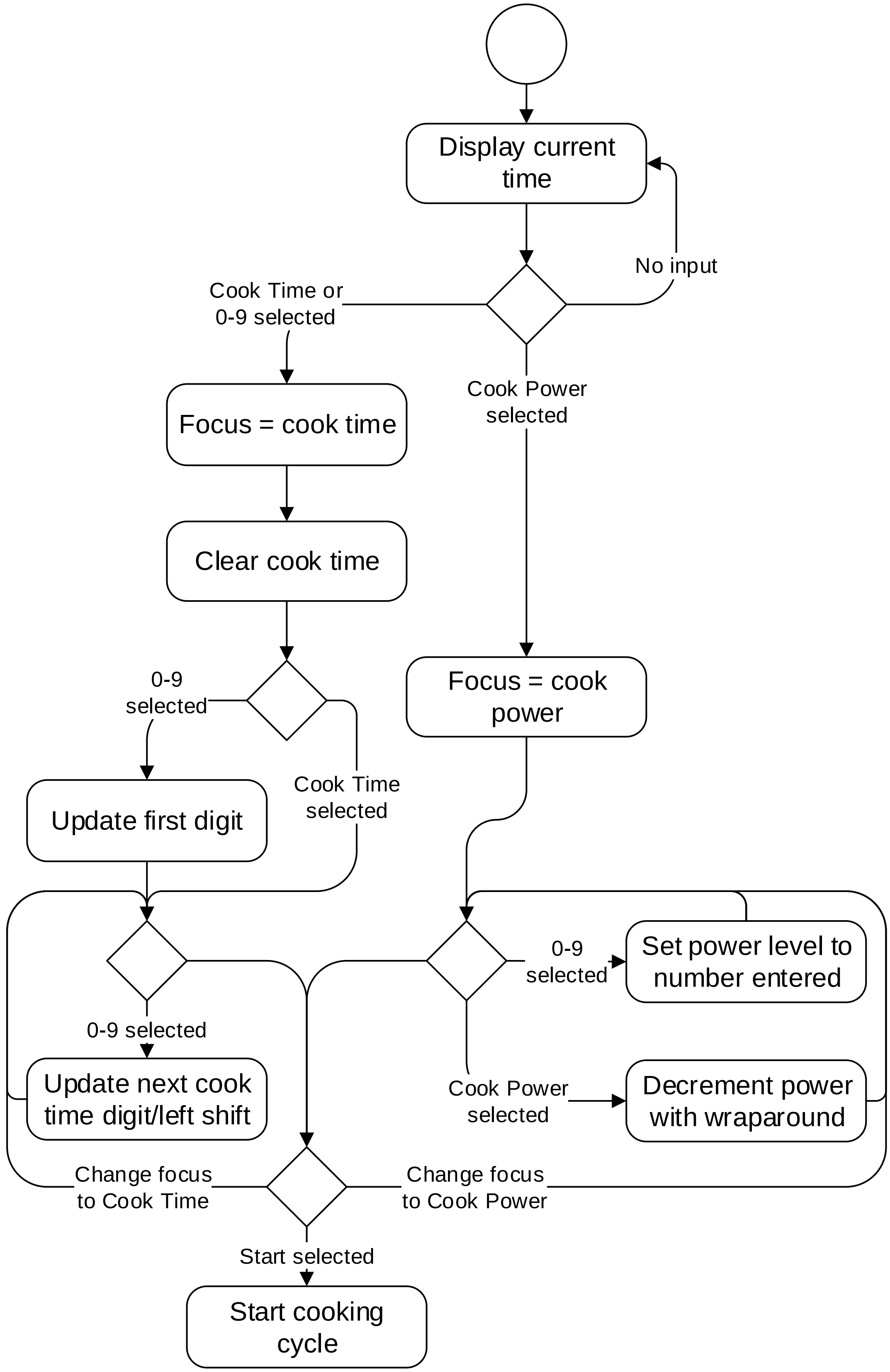


**Figure 3.1.3.2-1:** Whirlpool basic control

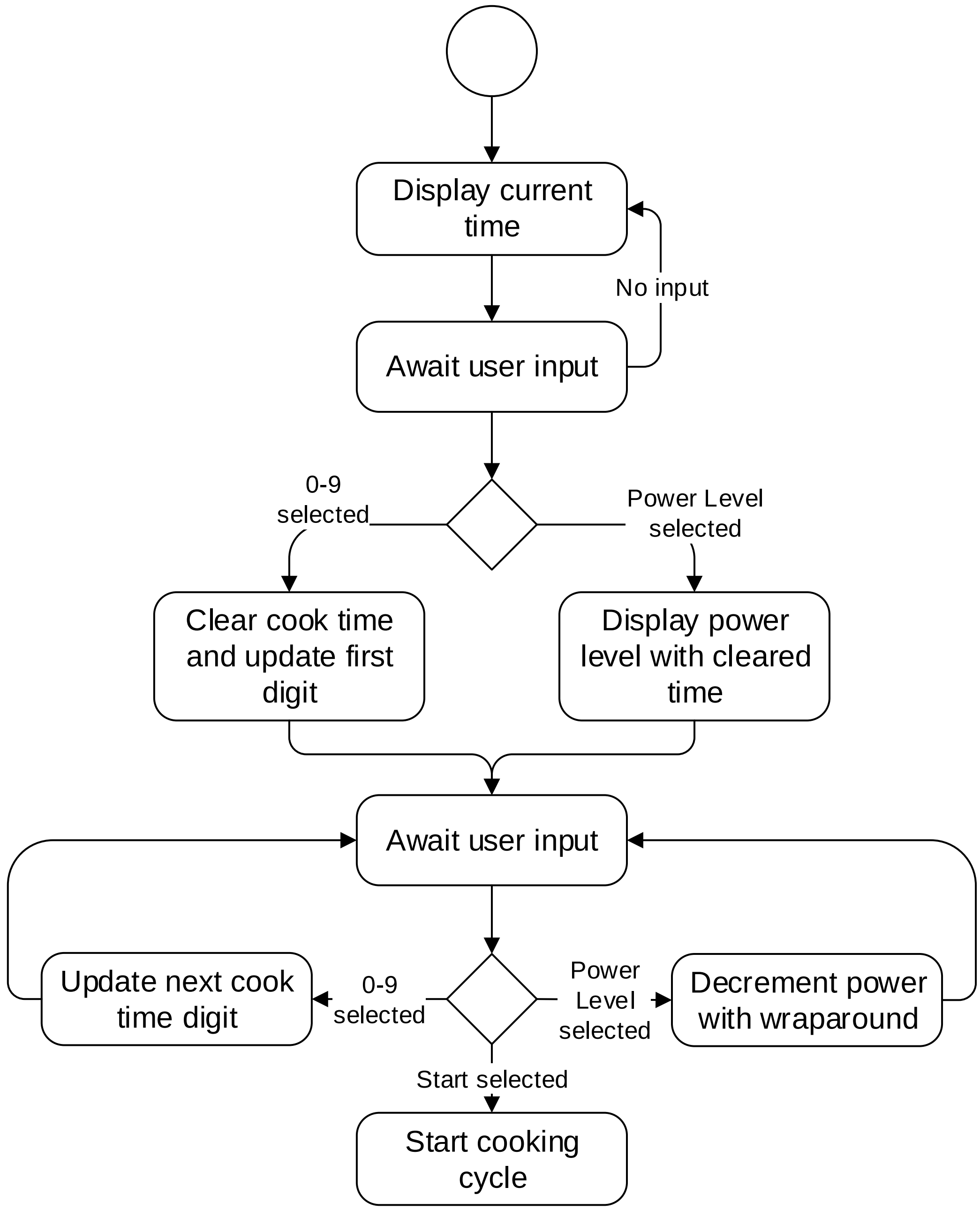


**Figure 3.1.3.2-2:** Panasonic basic control

### *3.1.3.3 Advanced Microwave Functionality*

With the lessons learned while examining basic functionality in mind, the functional differences and similarities of the more advanced control features were compared and evaluated for possible inclusion in the smart microwave interface. The first subset of features shown in Table 3.1.3.3-1 are from the Whirlpool microwave.

| Dedicated button | Power (%) | # of Key Presses | Preset Options | Time (mm:ss) |
|---|---|---|---|---|
| Beverage | 70 | 1 | 1 cup | 1:25 |
| | | 2 | 2 cups | 2:45 |
| Frozen Entrees | 100 | 1 | 10 oz | 7:45 |
| | | 2 | 20 oz | 12:30 |
| Frozen Pizza | 70 | 1 | 1 slice | 2:00 |
| | | 2 | 2 slices | 3:40 |
| | | 3 | 3 slices | 5:30 |
| Potatoes | 100 | 1 | 1 pcs | 7:00 |
| | | 2 | 2 pcs | 9:00 |
| | | 3 | 3 pcs | 12:00 |
| | | 4 | 4 pcs | 15:00 |
| Popcorn | 100 | 1 | 3.50 oz | 2:10 |
| | | 2 | 3.00 oz | 2:05 |
| | | 3 | 1.75 oz | 1:30 |

**Table 3.1.3.3-1:** Whirlpool dedicated button preset functions

These features have dedicated buttons and are designed for quick access to preparing specific foods. The user simply presses a dedicated button multiple times to cycle through the number of servings supported. Some of these foods will likely be handled by the automated feature of the smart microwave, but not all of these foods can be handled by reading a barcode.

The Panasonic microwave only has a single dedicated button to prepare a specific food:  popcorn. While the Panasonic microwave is considered a better featured model than the Whirlpool microwave, it appears to be a deliberate design decision to bury most of the advanced features behind a menu system in order to present a cleaner panel. The popcorn preset options are shown in Table 3.1.3.3-2. Note that there is no standard time since this advanced function relies on the built-in sensor to determine the length of time to apply cook power.

| Dedicated button | Power (%) | # of Key Presses | Preset Options |
|---|---|---|---|
| Popcorn | 100 | 1 | 3.50 oz |
| | | 2 | 3.00 oz |
| | | 3 | 1.75 oz |

**Table 3.1.3.3-2:** Panasonic dedicated button preset functions

Dedicated buttons are convenient, but it is not possible to provide a dedicated button for every preset function that a manufacturer wants to include in their microwave. This leads to the next set of advanced functions which are the general food preparation presets. These functions are usually accessed by first pressing a single master button multiple times to cycle through the various types of foods that have identified presets. In the case of the Whirlpool microwave there are two ways to select the preset function; cycling through presets using multiple presses of the Cook button and also pressing a number on the keypad to jump directly to the preset. The second option requires that the user has memorized which number selects the desired preset function. This is too onerous considering there are only five Cook preset functions, but it seems unlikely that a user will memorize an option unless it is used regularly. After the user selects the type of food to prepare, a quantity is entered using the keypad. The Cook button preset functions are shown in Table 3.1.3.3-3. The number in the Type column indicates the number of times to press the Cook button again or the number to press on the keypad after the Cook button is pressed just once.

| *Cook* button | Type | Power (%) | Servings Supported | Time (mm:ss) |
|---|---|---|---|---|
| Rice | 1 | 100 | 0.5 oz | 21:30 |
| | | | 1.0 oz | 23:40 |
| | | | 1.5 oz | 26:00 |
| | | | 2.0 oz | 27:40 |
| Fresh Vegetables | 2 | 100 | 1 cup | 2:00 |
| | | | 2 cups | 3:15 |
| | | | 3 cups | 5:00 |
| | | | 4 cups | 7:30 |
| Frozen Vegetables | 3 | 100 | 1 cup | 3:00 |
| | | | 2 cups | 5:15 |
| | | | 3 cups | 7:00 |
| | | | 4 cups | 9:15 |
| Canned Vegetables | 4 | 100 | 1 cup | 2:30 |
| | | | 2 cups | 4:30 |
| | | | 3 cups | 6:15 |
| | | | 4 cups | 8:15 |
| Bacon | 5 | 100 | 1 slice | 1:05 |
| | | | 2 slices | 1:45 |
| | | | 3 slices | 2:10 |
| | | | 4 slices | 2:45 |
| | | | 5 slices | 3:30 |
| | | | 6 slices | 4:15 |

**Table 3.1.3.3-3:** Whirlpool Cook button preset functions

The Panasonic microwave has a Sensor Cook button that takes the simpler approach of only cycling through the supported preset functions. If a user frequently prepared pasta using this microwave, it would likely be annoying to have to press the Sensor Cook button nine times every time. All preset functions also rely on the built-in sensor to determine the cook time. Utilizing the sensor removes the need to indicate the quantity of food that the user places in the microwave. The preset functions are shown in Table 3.1.3.3-4.

Note the Whirlpool's Cook function requires a bit more time using the control panel number pad. After Cook is pressed, the desired food is selected using the number pad, or Cook may be pressed repeatedly to cycle through all five of the available options. Finally, the quantity of food is entered using the number pad. Designing a feature that requires that many steps, given the limited display of this microwave, will likely result in reduced use of that feature.

Only some items such as Frozen Entree, Frozen Pizza, Canned Vegetables, and Popcorn stand out as candidates for the automated side of the smart microwave. Other items such as Oatmeal and Bacon may be candidates for automation depending on the form they take. The remaining items will need to be evaluated individually for inclusion in the manual side of the control software developed for this project. Functions that are shared by both microwaves are particularly good candidates for inclusion into the Memrowave interface since these are likely expected by many consumers. The challenge will be to include this functionality without resorting to a selection menu system that is too deep.

| Sensor Cook button | # of Key Presses | Power (%) |
|---|---|---|
| Oatmeal | 1 | 80 |
| Breakfast Sausage | 2 | 60 |
| Omelet | 3 | 60 |
| Frozen Entrees | 4 | 60 |
| Frozen Pizza | 5 | 80 |
| Potatoes | 6 | 80 |
| Fresh Vegetables | 7 | 80 |
| Frozen Vegetables | 8 | 80 |
| Pasta | 9 | 80 |

**Table 3.1.3.3-4:** Panasonic Sensor Cook button preset functions

### *3.1.3.4 Reheat Functionality*

A reheat function is provided by each microwave. The Whirlpool's reheat function shown in Table 3.1.3.4-1 utilizes steps similar to those that were used with the Cook button. The user first presses the Reheat button, followed by a number from 1 to 4 to select the food type, and finally a number indicating the number of servings being reheated. Once again the limited display of the microwave makes these functions a bit more difficult to use and therefore too easy to just ignore. Oddly, the final option, Dinner Plate, requires the user to input the quantity of 1 even though that is the only option available. Perhaps this is an idiosyncrasy of this model and larger models that use the same software have the option for more than one serving.

The Panasonic leverages its sensor for this function. The user simply presses the Sensor Reheat button and then Start. The time and power settings are completely controlled by the microwave. There are no other options to set. This is the kind of simple functionality that is the goal of the smart microwave just extended to all types of foods and food preparation requirements. Reheating food is a vital function of any microwave and a robust reheat feature needs to be included in the smart microwave's software.

| *Reheat* button | Type | Power (%) | Servings | Time (mm:ss) |
| --- | --- | --- | --- | --- |
| Soup/Sauce | 1 | 70 | 1 | 2:15 |
| | | | 2 | 4:15 |
| | | | 3 | 6:00 |
| | | | 4 | 7:20 |
| Casserole | 2 | 70 | 1 | 3:45 |
| | | | 2 | 6:30 |
| | | | 3 | 9:00 |
| | | | 4 | 11:30 |
| Baked Goods | 3 | 70 | 1 | 0:10 |
| | | | 2 | 0:15 |
| | | | 3 | 0:30 |
| | | | 4 | 0:40 |
| | | | 5 | 0:50 |
| | | | 6 | 1:00 |
| Dinner Plate | 4 | 70 | 1 | 5:15 |

**Table 3.1.3.4-1:** Whirlpool Reheat function

#### *3.1.3.5 Defrost Functionality*

The last major microwave feature is the defrost function. The Panasonic cannot use its sensor for this function since that sensor relies on the generation of steam from the food being prepared. Both microwaves rely on the user to enter the weight of the food to defrost, but the Whirlpool further subdivides the foods into three categories: meat, poultry, and fish. The Panasonic makes no distinction and relies solely on weight. Using the defrost feature on the Panasonic microwave requires that the user press the Inverter Turbo Defrost button followed by the appropriate weight using the control panel number pad. No other options are available. For the Whirlpool microwave, the user first presses the Defrost button. The user can repeatedly press the Defrost button to cycle through the three options - Meat, Poultry, and Fish - or press the corresponding number on the number pad. The user then enters the appropriate weight using the number pad. Defrost functionality is important for any microwave. It is simple enough that software should be written to mimic the functionality that comes with the microwave that will be modified for this project.

### *3.1.3.6 Miscellaneous Microwave Functionality*

There are several miscellaneous features that are common to both microwaves although implementations vary. These features are:

- Keep warm: Uses low/intermittent power to keep cooked food warm until it is removed from the microwave

- Timer (hold): After cooking, a timer will continue counting down and notify the user the food is ready after the timer expires.

- More/less: Allows the user to make minor adjustments as needed based on past experience/use of the microwave and specific product. This feature appears on other kitchen appliances such as single serve coffee makers and so should be fairly familiar to many users.

- Quick (add) minute: Used for short and/or quick microwave jobs such as warming a cup of coffee or slightly reheating a meal that has sat out on the table too long.

#### 3.1.3.6.1 Keep Warm

The keep warm feature on the Whirlpool microwave is an all or nothing proposition. Pressing the Warm Hold button turns the microwave on without further user input. The microwave continues to run, but the magnetron only comes on for a few seconds at a time. The Warm Hold can be used in conjunction with regular cooking by pressing the Warm Hold button after the regular cook time and power selections have been made and before Start is pressed.

The Panasonic microwave's Keep Warm feature allows the user to specify the warming time. The microwave uses a low power setting to keep the cooked food warm. The Keep Warm feature can be used in conjunction with regular cooking times by pressing the Keep Warm feature after cook time and power selections have been made and before Start is pressed. The warming time is set after pressing the Keep Warm button.

#### 3.1.3.6.2 Timer (Hold)

The timer feature on the Whirlpool microwave is not what a typical user might expect. It is an independent timer that cannot be incorporated into a food preparation sequence. Since this is an independent timer, use of the microwave is not hindered by a timer counting down. The design decision to make the Whirlpool microwave's timer an independent timer was likely due to its location.

As an over-the-range microwave, it is likely that users will want to utilize the timer for more than just preparing food in the microwave. If a user wanted to cook food in the microwave for 20 minutes followed by a 5 minute hold, the only way to accomplish this would be to start the timer at 25 minutes followed immediately by starting the 20 minutes cooking process.

The Panasonic microwave's timer feature is not an independent timer and can be utilized as part of the cooking process. Using the same example used for the Whirlpool, a user would set the 20 minute cooking time then press the Timer button and set its separate time for 5 minutes. The downside is that the microwave cannot be used while the timer is running. A solution that provides both an independent countdown timer like the Whirlpool microwave and a hold timer than can be inserted as part of the food preparation sequence like the Panasonic microwave is preferable.

#### 3.1.3.6.3 More/Less

The Whirlpool implements this feature without using a separate button and does so in an unintuitive way by using the Cook Power button rather than the Cook Time button. After pressing an automatic cooking option, for example Popcorn, the Cook Power button can be pressed to change the cooking time to more, less, or back to normal. This is likely a very often missed and unused feature of this microwave.

The Panasonic microwave has a separate More/Less button that allows the user to make slight adjustments to the cooking times based on past experience. For example, if the previous bag of popcorn prepared in the microwave was still furiously popping when the microwave stopped, the user may choose to add just a little more time the next time popcorn is prepared. The time change that the More/Less button has on cooking times varies depending on the selection. Popcorn can be +/-10 or 20 seconds while other selections such as Sensor Reheat may adjust by +/-10%.

#### 3.1.3.6.4 Quick (Add) Minute

The Whirlpool microwave has an Add Minute which functions by adding a minute of time for each button press and, if the microwave is already not running, will also start the microwave cooking cycle. Additional minutes can then be added as desired while the microwave is running.

The Panasonic microwave implements this feature using the Quick Minute button. Each press of this button increments the cook time by a full minute. The cook time can be increased whether the microwave is running or has not yet started cooking. Pressing the Quick Minute button does not start the microwave.

## 3.2 Relevant Technologies

The Memrowave is a conjunction of many different technologies into one ‘smart’ device. These technologies include magnetron control, ARM microprocessors, computer vision, display technologies, etc. In the following sections, we will go into detail about each relevant technology we researched while designing the Memrowave.

### 3.2.1 Magnetron Control

All of our subsystems are brought through a centralized system controller. This controller will handle tasks such as interfacing with the food product database, communicating with user interfaces, managing Wi-Fi access, and most importantly the magnetron. Controlling of the magnetron is vital to the Memrowave. This section will discuss the possible ways of controlling a microwave’s magnetron and how to best incorporate a system to control the magnetron. Figure 3.2.1-1 shows a diagram of the system collaboration.

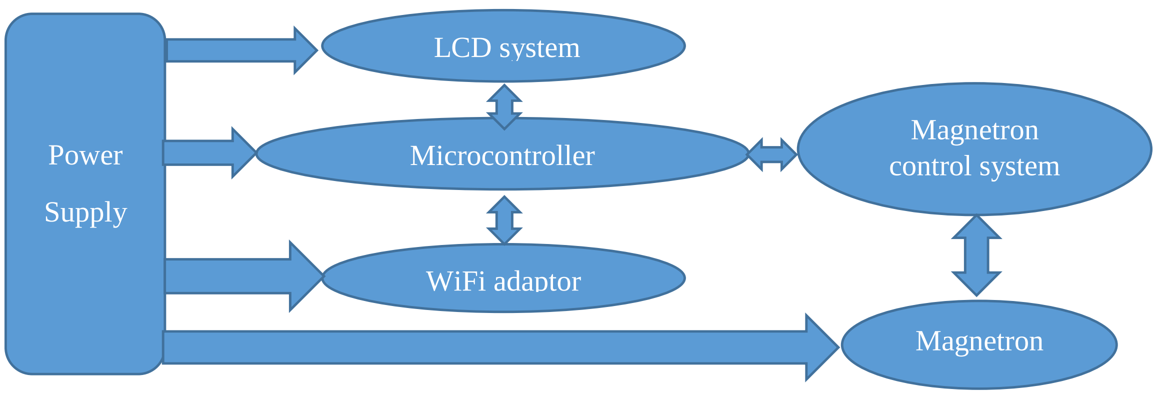


**Figure 3.2.1-1:** System control diagram

The magnetron is essentially an oscillator that produces a magnetic field and simultaneously an electric field. Average frequency ranges generated from the magnetron are around 0.5 GHz to 30 GHz however a specific magnetron will work at a sole frequency. Magnetrons are used in many applications from military to consumer. They can be used as a sputtering system or a cooking system. The Magnetron is made up of a cathode, anode, resonant cavities, permanent magnet, and filament. The anode servers as the positive source in the system while the cathode servers as the negative. The Anode is usually made from a copper block. The filament leads are used to keep structure order. The resonance cavities of the magnetron are what determine the center frequency of the magnetron. With all these elements working simultaneously electric and magnetic fields can be controlled and created by the magnetron interacting with electrons. In order to operate the magnetron current must flow through the filament and the cathode. Electron flow through the filament will transport electrons between the anode and cathode. Current is generated from this interaction; with a presence of current a magnetic field is formed. The permanent magnet will alter the flow of current causing a circular path of flow. This path is a function of magnetic field and electron mobility. Oscillations are created from the resonance cavities in the device. This will oscillate the charges thus cooking the food stored in the path of the generated RF power.

It was determined that salvaging the magnetron system from the original purchased microwave yields an effective outcome to our design. The next step was to review the possible ways of controlling the Memrowave's magnetron. If the design only required the magnetron to run at max power the system would only require a signal to turn on and off the magnetron. However to best meet our design we require the Memrowave to cook off the exact instructions of the product being considered. For example if the product requires a cooking time of 10 minutes for a 1000 Watts microwave at three fourths power. When scanned the Memrowave prepares the instructions and begins to cook at 750 Watts, the only way to achieve this is to regulate the power into the magnetron. In order to regulate our power we must first decide on what type of signal modulation we will choose. An inverter microwave was deemed too difficult for the project therefore, we opted to control power by varying the time on and off for the magnetron. Each power level corresponded to a give cycle time, for example 50% was 26 seconds on and 26 seconds off.

### 3.2.2 ARM Microprocessors

The Memrowave's interface is handled by an ARM microprocessor. ARM microprocessors are the most widely used microprocessors in the world of mobile handheld devices. They are reduced instruction set computing (RISC) processors that use fewer transistors than typical desktop computing processors resulting in lower cost and lower power requirements which is desirable for the Memrowave. Features and processing power allow ARM processors to run operating systems such as embedded Linux and Android which increases functionality and usability in applications such as the Memrowave. The prevalence of ARM processors in the marketplace should allow for multiple choices in software and connecting hardware during the design phase.

### 3.2.3 Computer Vision for Barcodes and QR Codes

Since we've decided to implement the barcode and QR code reader using a camera, we needed a computer vision solution to read them. Luckily, there is an open source, multi-format barcode image processing library available for

Android, called Zbar. This library supports UPC-A, UPC-E, QR Codes, and a few other barcode formats.

## 3.2.4 Output Display Technologies

The Memrowave's primary interface displays the software interface including pictures of food products, bar codes, and QR codes. The output display can be handled by a number of technologies. It is possible at the start to conclude that since the device will reside within the Memrowave it should be a direct display and not a projected display. Direct display technologies include emissive technologies such as organic light emitting diode as well as non-emissive technologies such as liquid crystal display and electronic ink. Since it is desirable to have the Memrowave display color photos of food products, e-ink is eliminated. That leaves two possible displays: OLED and LCD. The final; product incorporated an LCD resistive touch screen.

### *3.2.4.1 Organic Light Emitting Diode Display Overview*

Most emissive display technologies don't have the capability to display a satisfactory interface for the Memrowave due to lack of colors or pixel density. This is not the case for OLED displays which have recently seen increasing use in many applications such as mobile devices and televisions. As an emissive display technology, OLED displays don't require external/backlighting in order to be viewed. Lack of backlighting usually results in a thinner display and more energy efficiency. The viewing angle for OLED displays is extremely wide so that a Memrowave user could be standing to the side of the device and still be able to clearly view the display. Response time is outstanding in the sub-millisecond range. Contrast and color display capability are both excellent which would enhance the display of food product photos used by the Memrowave interface. OLED displays also have excellent black levels which would make display of the bar and QR codes look sharper. Some OLED negatives include variations in the aging of the different colors Unfortunately OLED displays tend to be more expensive and availability is limited at this time.

### *3.2.4.2 Liquid Crystal Display Overview*

The liquid crystal display is a mature technology that is currently prevalent in the smart device market as well as other markets. As a non-emissive technology, LCD must utilize an external light source in order to view the display. This backlighting does add some additional thickness and weight to the hardware, but this shouldn't be a problem since the Memrowave is not a handheld device. The backlight also reduces black level representation and can cause colors to look a bit more washed out. Many backlit screens can also be hard to see in bright lights, and kitchens are usually brightly lit, so visibility could be an issue. Some clarity issues can be countered by using thin-film transistor (TFT) technology, an active-matrix technology, which provides a brighter and sharper display and would be desirable at small display sizes such as used in the Memrowave. Another problem with LCD visibility can be a somewhat limited viewing angle. This can be highly dependent on the quality of the LCD. The number of colors that can be represented is quite good and should be more than adequate to display the pictures used by the Memrowave. Response time is fair and should be acceptable since the Memrowave will not be utilizing moving images like a television

#### *3.2.4.3 Output Display Technologies Summary*

OLED displays have more favorable features such as better visibility in bright light and wider viewing angles than LCD displays. OLEDS are also thinner since they do not require a backlight. Unfortunately OLED displays are currently have limited availability and are higher priced than LCD displays. Overall price is a factor for the Memrowave therefore a liquid crystal display will be chosen for the project.

### 3.2.5 Input Touch Screen Technologies

Primary input for the Memrowave will be accomplished with a touch screen device. These input devices are normally integrated with a display, such as a liquid crystal display, but the technology is distinct from the display technology that it may be used with. It is even possible to add a touch screen input device to a display that does not have integrated touch input. The two major touch screen technologies that could be used for the Memrowave are resistive and capacitive.

#### *3.2.5.1 Resistive Touch Screen Overview*

Resistive touch screens consist of two thin layers of material, usually transparent, that are separated by a thin gap so they are not touching. The inward facing surfaces of both layers are electrically resistive. The concept is that when a user presses on the top layer using a finger or stylus, the two layers make contact and the location of that contact can be determined. This design is an advantage since the location of the press can be determined regardless of the item pressing the layers together. A finger in an oven mitt or covered in flour is just as capable of being detected as a clean finger or a stylus. This would be desirable for the Memrowave since it is a kitchen appliance. Resistive touch screens are also fairly inexpensive technology and cost is a factor for this project. Disadvantages of resistive touch screens are that the two additional layers of material reduces the contrast of whatever display they are placed over and they can easily be damaged by sharp objects.

#### *3.2.5.2 Capacitive Touch Screen Overview*

Capacitive touch screens consist of a hard insulating layer such a glass with a transparent electrically conductive coating. The electrostatic field of the touch screen distorts when another electrical conductor such as a human finger comes in contact with the surface coating. This distortion can be processed to determine the location of the touch including the ability to sense multiple simultaneous touches. Since the surface is not a soft, flexible layer it is more resistant to damage by sharp objects. Contrast of an underlying display is not negatively impacted. However, since the technology requires a touch from an electrically conductive source, it would not detect a touch from a finger in an oven mitt and might not detect a touch by a finger covered with flour. Despite their prevalence in the market, capacitive touch screens tend to be more expensive than resistive touch screens.

### 3.2.6 Operating Systems

We have two options for the operating system of the Memrowave. We could design a custom operating system, or we can modify an existing operating system to suit our purposes. We decided to utilize android as the OS.

Designing a custom operating system would most likely be too complex, because of all of the advanced features of the Memrowave. It includes a screen, so we'd presumably need to implement some kind of display driver. It includes internet access, so we would need to implement some kind of network interface driver. It has a touchscreen, so we'd need to implement some kind of driver for that. With the availability of open source operating systems designing a custom OS would really just add unnecessary work.

This leads to the selection of a pre-existing open-source operating system. The most obvious choice would be some Linux-based operating system, because of its ubiquitous support for ARM microcontrollers. In fact, there are a number of Linux-based operating system options available for the Beaglebone Black, including Ubuntu, Angstrom, and Android.

The main advantage of Ubuntu is its large support base. The Ubuntu software repositories include mostly everything we would need to design the software for the Memrowave. The Angstrom distribution is a relatively new distribution developed by a smaller team, so it generally cannot match the level of support Ubuntu offers; however, the Angstrom distribution was created with embedded systems in mind, so it will most likely be more lightweight than Ubuntu and optimized for embedded use.

The problem with using Ubuntu or Angstrom is that Linux is generally lacking in support for touch screen interfaces. Most graphical Linux interfaces are based on window systems, which generally do not work very well for touch screens. This means we would have to either develop our own touchscreen GUI framework or configure some third party GUI library before we could even begin to implement our user interface. Generally, programming the user interface without such a framework in place is not a good idea. While it definitely could be done, it would make it much more difficult to keep a consistent style and flow throughout the entire user interface. This is where Android really shines.

Android was created specifically for touch screen devices, so it includes all of the necessary libraries to implement a touch screen user interface with consistent styling and control flow. It also includes a network stack, local database support, and mostly every other feature we need in an operating system. If we were to use either the Ubuntu or Angstrom distributions, we would need to carefully select all of the different libraries that we would need to write all of the software for the system, and try to fit them together into a working product. With Android, this is not necessary, because it already includes all of the software libraries necessary to accomplish what we want. This is why we've chosen Android for the Memrowave's operating system.

## 3.2.7 Wi-Fi Interface

The Memrowave is intended to be a connected kitchen appliance. The primary use of this interface is to retrieve product records from a central database located on the Internet. This activity will only be necessary for scanned items that are not already in the local database. Given this small requirement, it is not necessary to have a high bandwidth solution. While it is possible to connect the Memrowave to a network using the Beaglebone Blacks' built-in Ethernet port, it is highly unlikely that most houses will have a twisted-pair network port located in the kitchen. Therefore, connectivity is achieved by using Wi-Fi. Wi-Fi is a widely used and supported wireless technology that connects many different computing devices to a local network.

### *3.2.7.1 IEEE 802.11*

The standard associated with Wi-Fi is IEEE 802.11. This standard has been updated numerous times over the years with the specific version designated by the suffix, e.g. 802.11n. This suffix is used to distinguish Wi-Fi device support and capabilities. The more common versions in use today are shown in Table 3.2.7.1-1. While there are other published versions, some earlier and some more recent, they are not commonly available and there i no effort to support them. Bandwidth values are not included in the table. Bandwidth is not of vital importance to the Memrowave since the actual amount of data being exchanged should be small.

| 802.11*x* | Freq (GHz) | Range (ft) |
|---|---|---|
| b | 2.4 | 115 |
| g | 2.4 | 125 |
| n | 2.4 | 230 |
| | 5 | 230 |
| ac | 5 | 115 |

**Table 3.2.7.1-1:** Common Wi-Fi standards

While 802.11b and 802.11g are older standards dating back to 1999 and 2003 respectively, they still see some use in low cost consumer products and in older equipment that has not been replaced due to cost. For example, it is quite likely that some Internet service providers may still supply wireless access points that only support the 802.11b/g standards. IEEE 802.11n is unique among the version shown because it supports both 2.4GHz and 5GHz frequencies. This capability is called dual band support. It also has double the usable range of the other versions, even the newer 802.11ac. The latest standard that is seeing increasing deployment, 802.11ac, has the capability for higher bandwidth compared to the earlier versions, but bandwidth is not a major consideration for this project. Fortunately, 11ac devices normally also support 11n Wi-Fi at a minimum.

#### *3.2.7.2 Wi-Fi Interference*

One complaint often heard regarding microwaves is that they interfere with wireless communications, including Wi-Fi connections. The Memrowave will not be utilizing the Wi-Fi connection during cooking operations. The only time that Internet communication is needed is during the scan and identify process. Once a product record has been received from the online database and the settings are sent to the Memrowave's control board (by the user pressing Start), the user interface including the Wi-Fi being placed on standby. The Memrowave control board will handle all microwave operations including time display and cancel via opening the cabinet door. Therefore, microwave interference with Wi-Fi connectivity is not an issue.

#### *3.2.7.3 Wi-Fi Interface Summary*

After reviewing the options, the best range of Wi-Fi support for the Memrowave is 802.11g/n. This gives a good range of access point support, provides adequate bandwidth, and provides the option of dual band via 11n. Having legacy support of 11g is important to remove a roadblock to possible adoption in the marketplace. Older devices support both 11b and 11g therefore there is no explicit requirement to support 11b. The biggest reason to include 11n support is because of the increased range which is important since kitchen distance to a wireless access point, which may be at the opposite end of many houses, is likely not considered by many consumers. Dual band support is welcome, but not required since interference is not a concern. 11ac, while new and high bandwidth, just adds cost without any real benefit given the Memrowave's low bandwidth needs. The final consideration is the Wi-Fi adapter support in the version of Android used.

# 4 Project Design Details

In this section, we will begin to describe all of the details of the design process of the Memrowave, starting with our initial design approaches, continuing on to the hardware, and finishing with the software.

## 4.1 Initial Design Approaches

The Memrowave's main function is to integrate a high powered processing unit with a printed circuit board of our own design. A high power processor allowed for more accurate barcode scanning to determine what product is ready for use. If we chose to use a smaller embedded processor we predicted that the overall design will be hindered. However we first did look into perusing a single embedded processor to be the main unit of the Memrowave. The main advantage of this route is form factor. Using one embedded system allows for only one overall printed circuit board design. Compared to using a BeagleBone Black integration which uses three separate layers as depicted in Figure 4.1-1.

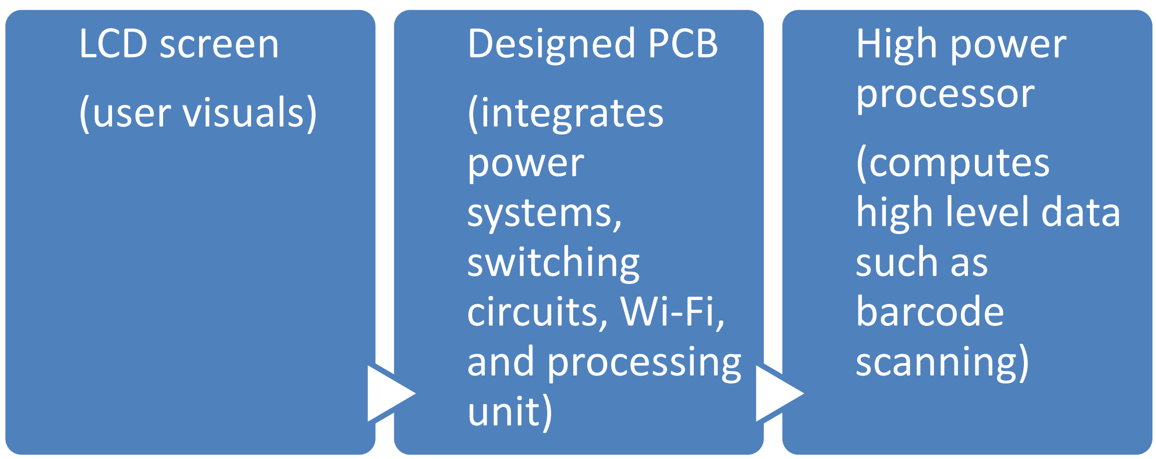


**Figure 4.1-1:** Possible integration of designs

However, the single benefit of using only one printed circuit board with the main processor integrated to it was not enough to merit use. The main disadvantage was shown to be its slow speeds when scanning barcodes. Therefore the Memrowave design team used a high powered processor

Using this method requires our team to design the PCB that functions as the integration of all subsystems. The PCB integrates with the power supply, LCD screen, Wi-Fi, and the processing unit we use. This PCB also functions as a sandwich layer between our subsystems, forcing our team to make the PCB which connects them from the middle.

Overall our design meets the requirements to create a new innovative microwave to help make daily life easier. This was accomplished through unique integration of high level architecture and sub-systems.

## 4.2 Hardware

In this section, we will detail the design process of the Memrowave's hardware. While designing the hardware, we be sure to meet all of our objectives and specifications outlined previously.

### 4.2.1 Power Supply

The power supply for the microwave had to be able to power the microwave as well as the additional electronics and components added to the system. This includes the LCD display, processor unit, PCB circuitry, and WIFI adapter. The typical microwave has a power specification between 850-1800 watts of total power, we were able for our microwave to fall into these guidelines. The power supply provided AC power to the magnetron, turntable, fans, and lights. DC power was provided to the processing unit, camera, and LCD screen. Since the magnetron typically uses 4000 V at 300mA when at full power, a HV transformer circuit must implemented. This can be done through using a printed circuit board with a voltage handling in the proper range since we want to keep a space-saving form-factor for a microwave. However, considerations for developing this HV circuit were dropped because of the dangers of direct arc over and corona production. When voltage potentials exceed the dielectric's ability to withstand it, arc over occurs. Corona discharge is a failure mechanism which causes degradation of the insulation system, essentially a precursor to direct arc over occurrence.

A typical camera module which we could use as the Memrowave's QR code reader and scanner input voltages range from 6-20 volts DC and requires 50 milliamps typically. The camera can be wired directly to the PCB power supply board with current limiting or through or processing unit. Since the camera is used infrequently and under certain conditions, power saving opportunities are available. This would be ideal and easier to implement if power can be supplied through the processing unit. The camera can be implemented under the control of the processing unit only to be turned on or active while food products are about to be scanned or if the user comes into close proximity to the microwave, the microwave can be activated out of a power save mode for less power consumption.

Powering a 4-7 inch LCD touchscreen will require a range of 4.0-6.0 volts DC and a minimum of 500 milliamps. Based on the processing unit we use for the Memrowave, a power will either need be power through PCB circuitry or the processing unit. Limits on the voltages the processor can supply will be the deciding factor in this decision but use of the processing unit as the power delivery system directly would make power saving much easier, due to the LCD touchscreen being another component that is used intermittently and also be adjusted for brightness, power savings would be possible.

A possible wireless network processor for creating WIFI connections will require 3-5 volts DC and a max current of 100 milliamps. Due to the necessary voltages and the close integration needed between the processing unit and the Wi-Fi module, power will most likely be provided through the processor. Our ideal Wi-Fi module will fit into these power specifications while providing the best integration and WIFI range and speed possible in our budget

Common parts of the typical microwaves such as the internal lights, beeper, and turntable will be powered through the AC outlet voltages and specific current loadings. Since each of these internal parts work intermittently, the task for switch them on and off will be done through the processing unit or other means, such as opening the microwave door to turn on the internal light.

Since both AC and DC power is needed within the Memrowave, a way for converting AC to DC is needed. An AC DC inverter will be used to convert the 120V AC domestic outlet to 12V-20V DC. The inverter must also be able to meet our total DC current needs as well. To acquire the multiple voltages needed for the DC devices, voltage regulation will used based on the specific DC devices that we will chose, ideally a maximum of three different voltage regulators should be sufficient. Another possible way of obtaining the dried voltages is to build the voltage dividers as well on the PCB, this option would save space as well as money and more accurate voltage could be produced. The use of voltage regulator components brings the advantage of a more robust system, a current loading limits can observed much easier based on the device specifications and with voltage regulators a more precision DC voltage can be.

Below in Table 4.2.1-1 is a working summary of the main parts and component of the Memrowave including details on required voltages and currents.

| Components | Voltage(V) | Current(mA) |
|---|---|---|
| Magnetron | 4000 AC | 300 |
| Processing Unit | 5-12 DC | 30 |
| LCD Touchscreen | 4-6DC | 500 |
| Camera | 6-20 DC | 50 |
| WIFI Module | 3-5 DC | 100 |
| Internal Light | 120DC | 200 |

| | | |
|---|---|---|
| Beeper | 120 DC | 20 |
| Turntable Motor | 21 AC | 150 |

**Table 4.2.1-1:** Microwave component power specifications

In Figure 4.2.1-7 is a power flow chart for all the electrical parts and components of the Memrowave.

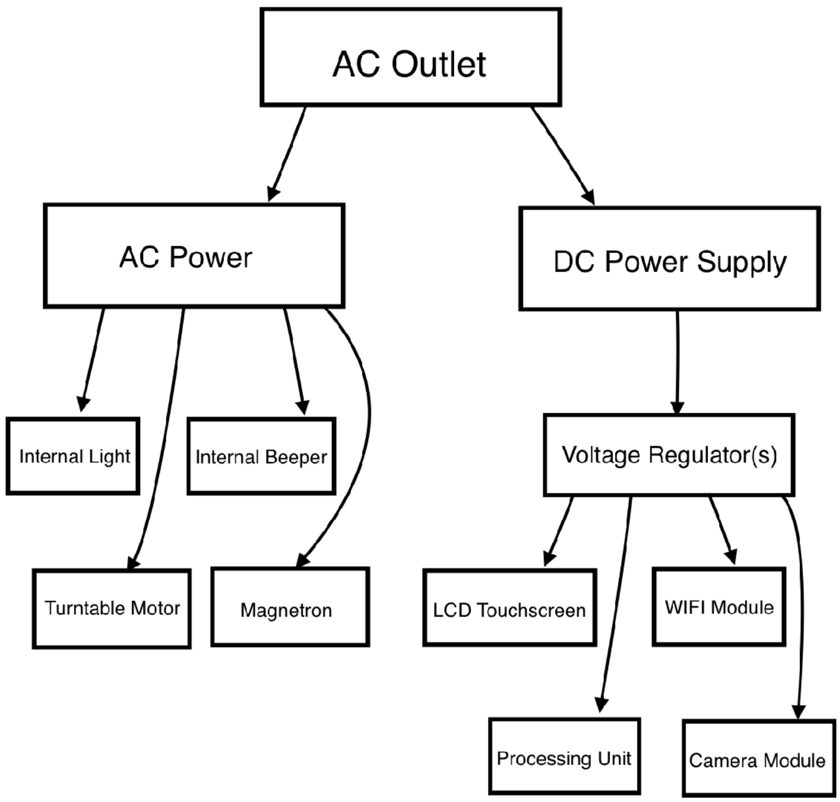


**Figure 4.2.1-2:** Power Flow Chart

## 4.2.2 Magnetron Controller

The controller was used for the Memrowave was the MSP430 created by Texas Instruments. The reasoning behind using the MSP430 to control the power delivered to the Magnetron was its cheap cost to obtain (Starting at $9.99 according to Texas Instrument's website), ease of use, availability, already used for other features in the Memrowave, and ability to program Pulse width modulation functionality. However a PWM was not used in the final design.

The MSP430 was acquired from ordering directly from the Texas Instruments website and use of available pre-purchase models. The Memrowave used mostly pre-purchased models from previous usage. Placement of the controller system was curial to the design of the Memrowave. Our PCB was form fitted enough for proper placement in the microwave. The PCB housed the controller for the magnetron, the power supply connections, and connections to the BeagleBone black. The board was mounted behind the original microwaves interface. The area provided little interaction from pre-existing hardware. The area also houses a large free space to fit components as seen in multiple models such as Kenmore, GE, and etc. Once gutted the controller was mounted in the desired location and sealed upon completion. We can create a flow chart of the controller's layout as follows in Figure 4.2.2-1.

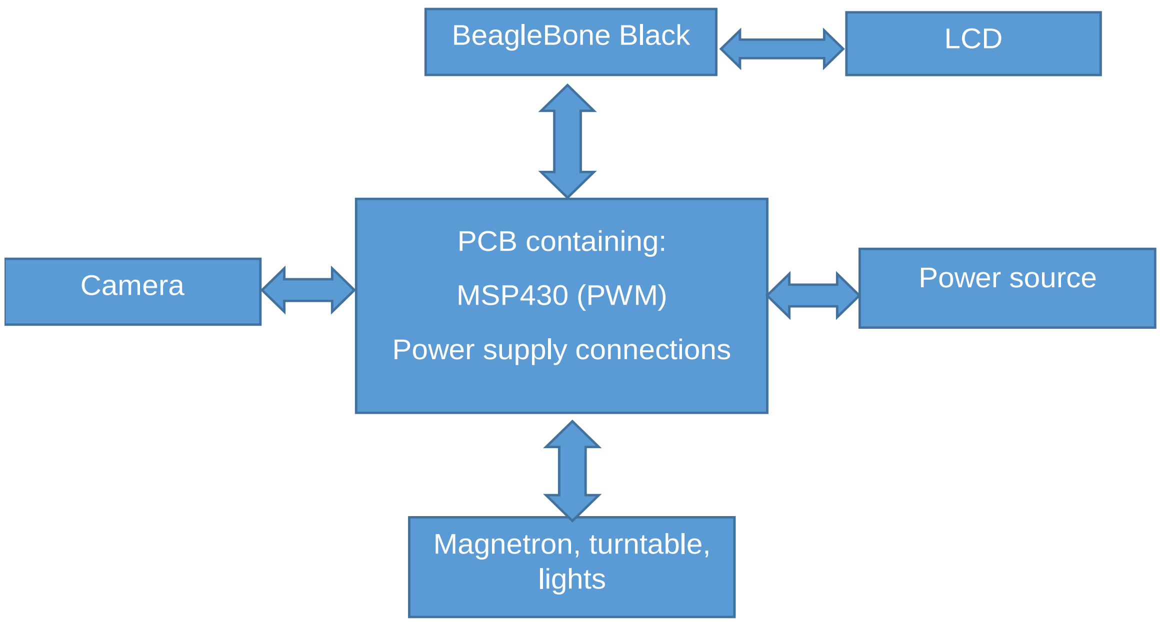


**Figure 4.2.2-1:** Controller placement

As Figure 4.2.2-1 shows the board was sandwiched between the boards together behind the shell of the microwave. Lastly our magnetron controller will ideally be interfaced with other controllers that work with the other subsystems. All controllers will be organized onto the PCB on completion. As stated before form factor was a must in our design since we are limited by space constraints.

## 4.2.3 Miscellaneous Microwave Controller

Along with powering the magnetron we must also be able to interact and control two other important features of the microwave. These are controlling the turntable and powering the interior light during cooking and opening of the main door. An option to simplify our design can be to scratch the turntable and have the food cook stationary. However using the turntable provides a more thoroughly cooked product. The turntable's purpose in the microwave is to help balance and even the cooking of a product. Microwaves cook by launching standing waves into the product and exciting atoms thus heating food. Figure 4.2.3-1 depicts such a standing wave.

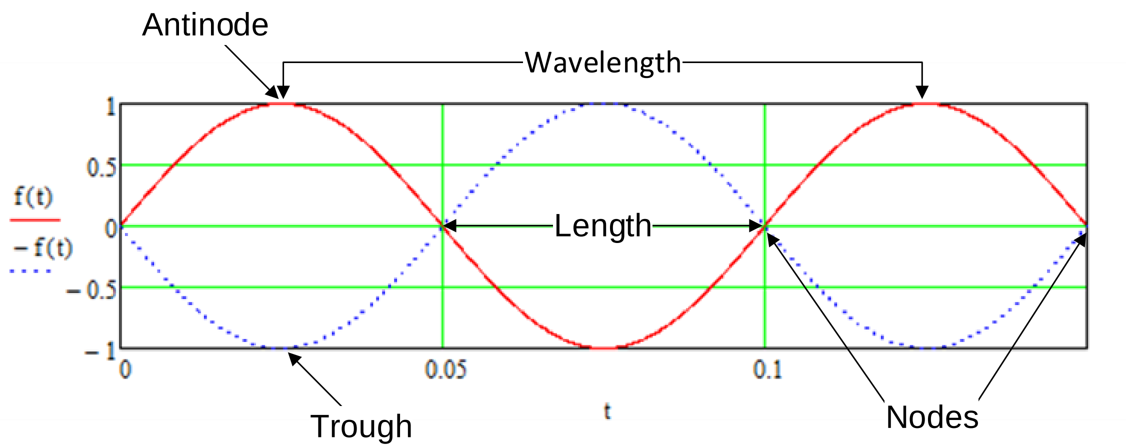


**Figure 4.2.3-1:** Standing wave depiction

For parts of the product that are located in the nodes of the wave the amplitude of energy is at a minimum. Therefore in these locations the product heats slower. Using the turntable provides a solution to this problem because we will be able to rotate and allow different areas to interact with the wave nodes. The goal of the Memrowave is to provide the user with the best possible experience therefore we kept the turntable as part of the final product. This required us to control power to the turntable. Lighting is also another must have feature of the Memrowave. We kept the lighting system that came preinstalled with the original microwave. Since we are gutting the internals of the microwave's systems we needed a way to control power flow to the lights inside the microwave, which calls for another system to be integrated.

When working with the turntable it is important to figure out the method in which we will achieve controlling of it. Since the microwave we purchased already contained a motor for the turntable we reused this motor and connecting it to a controller of our choosing. This option was beneficial because saved on the cost of our design and can saved time. If we decided to remove the original motor we lose production time on removal and installation of a new motor system. Being able to integrate the original motor with the power supply we implemented and the controller we used simplified our design. Using a motor that only relies on one speed is also beneficial because a circuit to control speed does not have to be implemented. For example if our motor needed to be capable of changing speeds and directions, a solution was an H-bridge circuit in combination with a PWM. Without this constraint our system saves on parts and design hours. Since we do not need to add circuitry to change speeds our task is simplified. Therefore using the original motor is a more likely candidate.

Other pieces of hardware that must be powered are the microwave's pilot light. The light needed to be supplied with an input signal from our power supply. It is also imperative that the light is allowed to switch to an on and off state. In Table 4.2.3-1 we provide a true table to show states.

| Condition | Status |
|---|---|
| Microwave door open | Light is on |
| Microwave door closed and not cooking | Light is off |
| Microwave door is closed and currently cooking | Light is on |

**Table 4.2.3-1:** Illumination table

The Memrowave kept the original microwave light. The light was controlled using the MSP430G2553. The next hardware we needed to control was the microwaves fans. The original microwave fan was used in the final Memrowave. Controlling the Memrowave required 4 input/output pins to function. These are listed as; lights, magnetron, buzzer, and the door switch. Because we only needed 5 pins we used the MSP430G2553. This chip was able to control power to the magnetron, turntable motor, lights, and fans. A low level diagram of the layout is shown in Figure 4.2.3-2

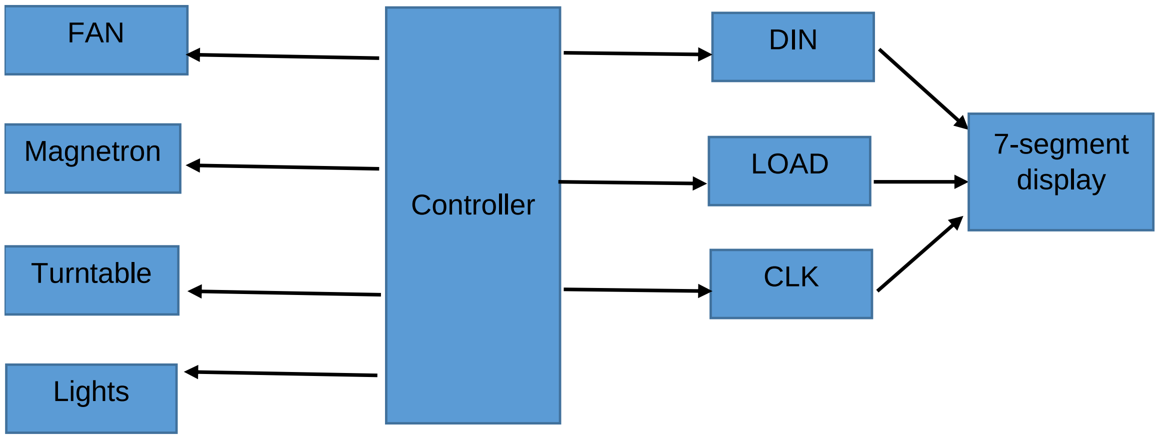


**Figure 4.2.3-2:** Pin Diagram

In order to switch the electrical elements on we created switching circuits. These switching circuits will function in unison during the operation of the microwave and during ideal use. The switching circuits were created using BJTs. Two of the circuits we made; one for the fan motor the turntable motor, the light, and one for the magnetron. Each circuit was placed on a separate pin of the microcontroller. Prototyping of circuitry was conducted on a breadboard. Once prototyping is completed we will then fabricate a printed circuit board which will house the circuits. The circuit was implemented by our design team using purchased passive components, transistors, wires, and solder. For better time management our design team purchased a solder station. This allowed for consist work and reworking of our printed circuit board.

## 4.2.4 ARM microprocessor

In this section, we will discuss the selection of the ARM microprocessor for the Memrowave.

### *4.2.4.1 ARM Microprocessor Specifications*

While most ARM microprocessors will have adequate computing power for the Memrowave, there were two general and fundamental requirements that will be used to determine which ARM microprocessor was used. The first requirement was capability of running Android out of the box, preferably Android 4.x or higher. Android was selected to aid software development such as the user interface and food preparation database. The second requirement was the microprocessor had to be available in an experimenter board that can in turn easily connect to add-on expansion boards similar to Arduino shield add-on boards. This requirement was needed since this was the method, which was used to connect the touch screen display. The board designed and built for the Memrowave was connected like any other expansion board. Using these two requirements immediately eliminated some ARM microprocessors such as the Broadcom BCM2835 used in the popular Raspberry Pi board which doesn't have a supported, stable version of Android available. The Texas Instruments OMAP4460 ARM Cortex-A9 MPCore used on the Pandaboard ES looked very promising at first due to its Android support, graphics capabilities, and built-in camera interface which would be useful for this project. However, the Pandaboard is more of a standalone development platform that doesn't use standard add-on boards. The Pandaboard was also expensive compared to most experimenter boards. The Texas Instruments Sitara AM335x ARM Cortex-A8 Microprocessor used on the Beaglebone Black seemed to be the right fit.

#### *4.2.4.2 TI Sitara AM335x Microprocessor*

The only ARM microprocessor fitting the requirements of Android support and expansion board capability was the Texas Instruments Sitara AM335x ARM Cortex-A8 Microprocessor used on the Beaglebone Black. Table 4.2.4.2-1 shows specifications for the Sitara AM3358 microprocessor. Table 4.2.4.2-2 contains the specifications and features of the Beaglebone Black. Figure 4.2.4.2-1 shows the dimensions of a typical Beaglebone Black expansion board. Note the 46-pin headers on opposite edges of the board. These headers are used both to connect the Memrowave PCB to the Beaglebone as well as provide a pass-through connection to the LCD touch screen display. The set dimensions of the expansion board required several design considerations. First, it was important that the LCD touch screen display selected for the project was a Beaglebone Black expansion board, known as a "cape", or could be modified to fit the Beaglebone Black's headers. The Memrowave PCB was sandwiched between the LCD and the Beaglebone Black using matching 46-pin headers. Second, components and associated connections had to fit within the dimensions of the chosen expansion board profile. This included choosing components that stayed within the height of the headers since the board was sandwiched between the Beaglebone Black and the LCD.

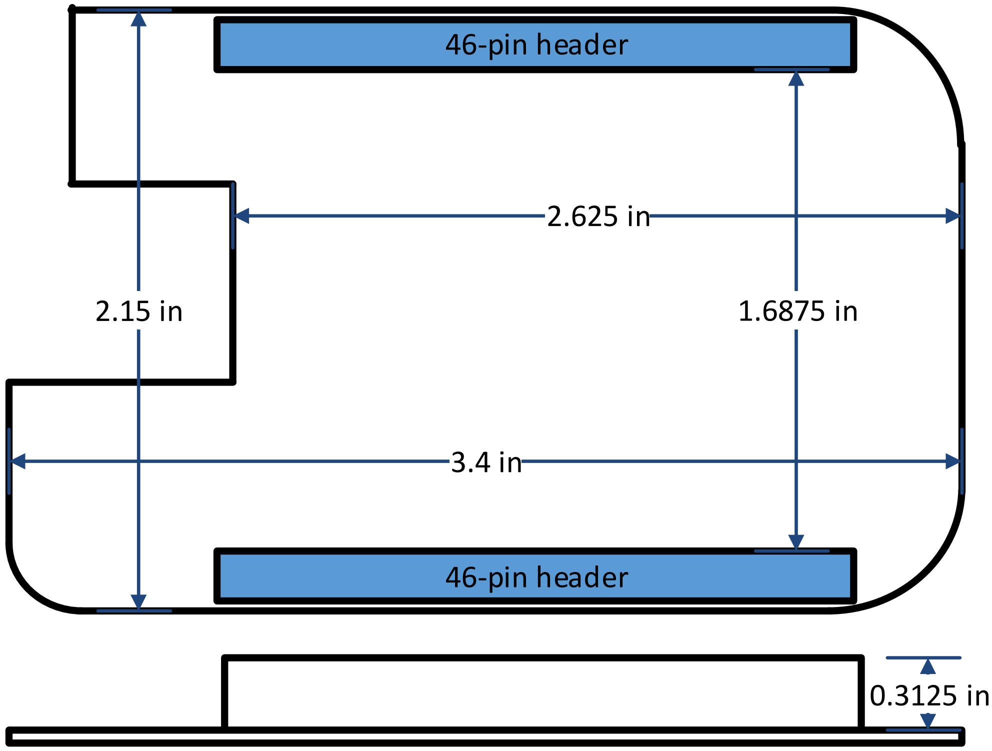


**Figure 4.2.4.2-1:** Beaglebone Black expansion board dimensions

| | |
|---|---|
| ARM CPU | ARM Cortex-A8 |
| ARM MHz (Max.) | 1000 |
| ARM MIPS (Max.) | 2000 |
| On-Chip L1 Cache | 64 KB (ARM Cortex-A8) |
| On-Chip L2 Cache | 256 KB (ARM Cortex-A8) |
| Other On-Chip Memory | 128 KB |
| Graphics Acceleration | 3D |
| Display Options | LCD |
| DRAM | 1 16-bit (GPMC NAND flash NOR Flash SRAM) |
| USB | 2 |
| MMC/SD | 3 |
| UART (SCI) | 6 |
| PWM (Ch) | 3 |

| | |
|---|---|
| Real Time Clock | 1 |
| I2C | 3 |
| SPI | 2 |
| DMA (Ch) | 64-Ch EDMA |
| Standby Power | 7 mW |
| IO Supply (V) | 3.3 |
| Operating Temp Range (C) | -40 to 105 C |

**Table 4.2.4.2-1:** Sitara AM3358 ARM Microprocessor Details

| |
|---|
| 4GB 8-bit eMMC on-board flash storage |
| 512MB DDR3L 800MHZ SDRAM |
| 3D graphics accelerator |
| NEON floating-point accelerator |
| 2x PRU 32-bit microcontrollers |
| SD/MMC Connector for microSD |
| USB host, Ethernet, HDMI |
| 2x 46 pin headers |

**Table 4.2.4.2-2:** Beaglebone Black Details

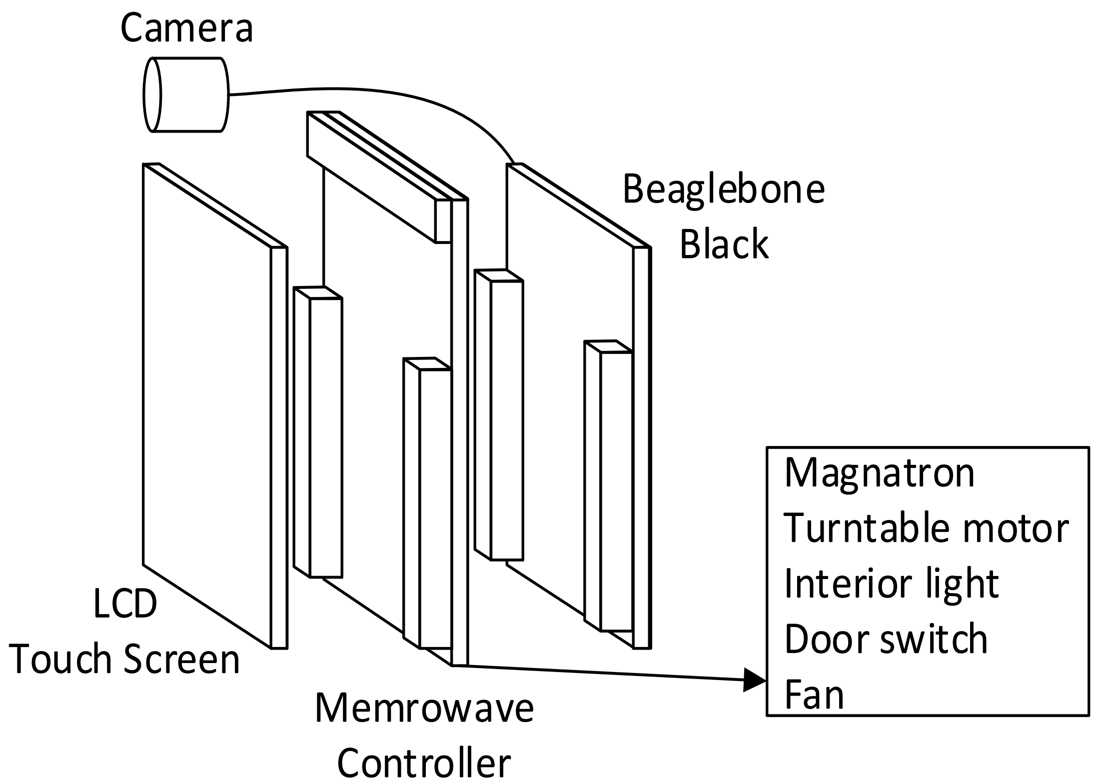


**Figure 4.2.4.2-2:** Simple Memrowave system diagram

## 4.2.5 LCD with Touch Screen Display

In this section, we will discuss the various options for the output display and touch screen hardware.

### *4.2.5.1 Specifications*

The Memrowave's display handled the input and output of the software interface. The display was a thin film transistor (TFT) liquid crystal display (LCD) utilizing an integrated resistive touch screen. The physical size of the display were determined by potential microwave chassis space. This space used was only able to accommodate a 4.3-inch diagonal display with the display in the portrait orientation. The display used was compatible with the Beaglebone Black with Android 4.2.2 running as the operating system. The display receives its power directly from the Beaglebone black.

#### *4.2.5.2 LCD with Touch Screen Display*

After extensive research there does not appear to be much demand for LCD capacitive touch screen displays for the Beaglebone Black at this time. Most displays at this small size use resistive touch screens. The display used was a 4.3-inch display from 4D Systems. See specifications in Table 4.2.5.2-1. Note that the normal display width/height dimensions are shown for the rotated portrait orientation that was used for the Memrowave. Also note that the overall size of the part is larger than the display since the LCD PCB extends beyond the actual display boundaries on all 4 sides.

| | |
|---|---|
| Display size | 4.3-inch |
| Display resolution | 480 x 272 pixels (used in portrait) |
| Aspect ratio | 16:9 approx. (30:17 actual) |
| Touch type | resistive |
| Display width | 53.8 mm |
| Display height | 95 mm |
| Overall width | 80 mm |
| Overall height | 120.4 mm |
| Power | 5V and 3.3V provided by Beaglebone Black |
| OS support | TI Android 4.2<br>Angstrom<br>Debian |

**Table 4.2.5.2-1:** 4DSystems 4DCAPE-43(T) LCD touchscreen cape details

### 4.2.6 Wi-Fi Interface

Internet connectivity gave the Memrowave's software access to an online products database that can be queried for items that are not found in the local database. This keeps the local database as small as possible, a desirable outcome when working with a limited amount of memory. Only the user interface software running on Android required Internet connectivity. This connectivity was provided via a USB Wi-Fi device attached to the Beaglebone Black's USB port. Kitchen appliances tend to remain in use for many years and are not replaced nearly as frequently as smart phones and other smart devices. It is likely that Wi-

Fi standards will change during the typical lifespan of a microwave. Using a USB device has the advantage of being easy to replace or upgrade. The USB adapter used was the Logic Supply UWN200 Compact USB Wi-Fi Adapter with 4" Antenna. Table 4.2.6-1 shows the adapter's specifications.

| | |
|---|---|
| Controller | MediaTek MT7601 (Ralink 7601) Controller |
| Standards | IEEE 802.11b/g/n |
| Frequency | 2.4GHz |
| USB | 1.1, 2.0, 3.0 |
| Linux driver | MT7601U |

**Table 4.2.6-1:** Logic Supply UWN200 specifications

### 4.2.7 Camera

For the camera, we used the Logitech HD C270 USB webcam, shown in Figure 4.2.7-1. It has a 1280 x 720 resolution, which was more than enough to read barcodes and QR-codes reliably.


**Figure 4.2.7-1:** Logitech C270 Webcam

## 4.3 Software

The Memrowave cannot be ‘smart’ without well developed software, so in this section we will be discussing the software design details and considerations for the Memrowave.

### 4.3.1 The Android Operating System

In section 3.2.6, we compared the various operating systems that we could have used for the Memrowave and decided that Android was the best choice. Android support for the BeagleBone Black was provided by the open-source Rowboat project, which enables Android Jelly Bean v4.2.2 to run on TI Sitara ARM Cortex-A processors.

### 4.3.2 Drivers, Libraries, Dependencies

There are a number of drivers that are needed for Android to support the hardware of the Memrowave. A driver was needed for the LCD to work properly, but luckily, this driver was included in the Android distribution we used. The driver for the resistive touch screen was included as well. However, the driver for the camera was not included, so it was obtained separately and installed on the system, as well as the driver for the Wi-Fi interface.

Android included most of the libraries we needed to implement in software of the Memrowave, with a few exceptions. Android didn’t have a native library for reading barcodes or QR-codes using a camera. To handle this, used the open source ZBar multi-format barcode image processing library. This software library is licensed under the Apache License 2.0, which gives us permission to use the software in our project. A link to the full text of the Apache License 2.0 is shown in Appendix A.

### 4.3.3 Microwave Control

The microwave hardware is controlled by a circuit separate from the BeagleBone Black, with its own microcontroller. The microwave control circuit takes care of the timing, power level, magnetron control, fan control, turntable control, beeper, and the light control. The BeagleBone Black communicates with the microwave control circuit via the I2C communication protocol, with the BeagleBone Black configured as the master and the microwave control circuit configured as the slave. In Table 4.3.3-1, we outline all of the features of the microwave control circuit and assign the I2C commands that will be used to invoke these features.

| Command | Description | I2C Command |
|---|---|---|
| set_timer | Clears the timer and sets it to the specified time | 0x00 |
| get_timer | Requests the current state of the cook timer | 0x01 |
| clear | Clears the current time from the timer display | 0x02 |
| cook_start | If the timer is set, this command will start the microwave, which will then proceed to cook the food at the current power level setting. | 0x03 |
| cook_stop | Pauses the timer and stops the microwave from cooking | 0x04 |
| timer_start | Starts the timer without cooking | 0x05 |
| set_power_level | Sets the power level to the specified value | 0x06 |
| add_time | Adds the specified amount of time to the current timer state | 0x07 |
| set_clock | Sets the real-time clock to the specified value | 0x08 |

**Table 4.3.3-1:** Microwave control circuit commands

For commands that are expected to return a value, the I2C master must write the I2C command to the slave device, and then read from the slave device to retrieve the result. Simultaneously, on the slave device, after receiving a command that expects some return value, the slave must write to the master. An activity diagram of the I2C command sequence on the microwave control board is shown in Figure 4.3.3-1.

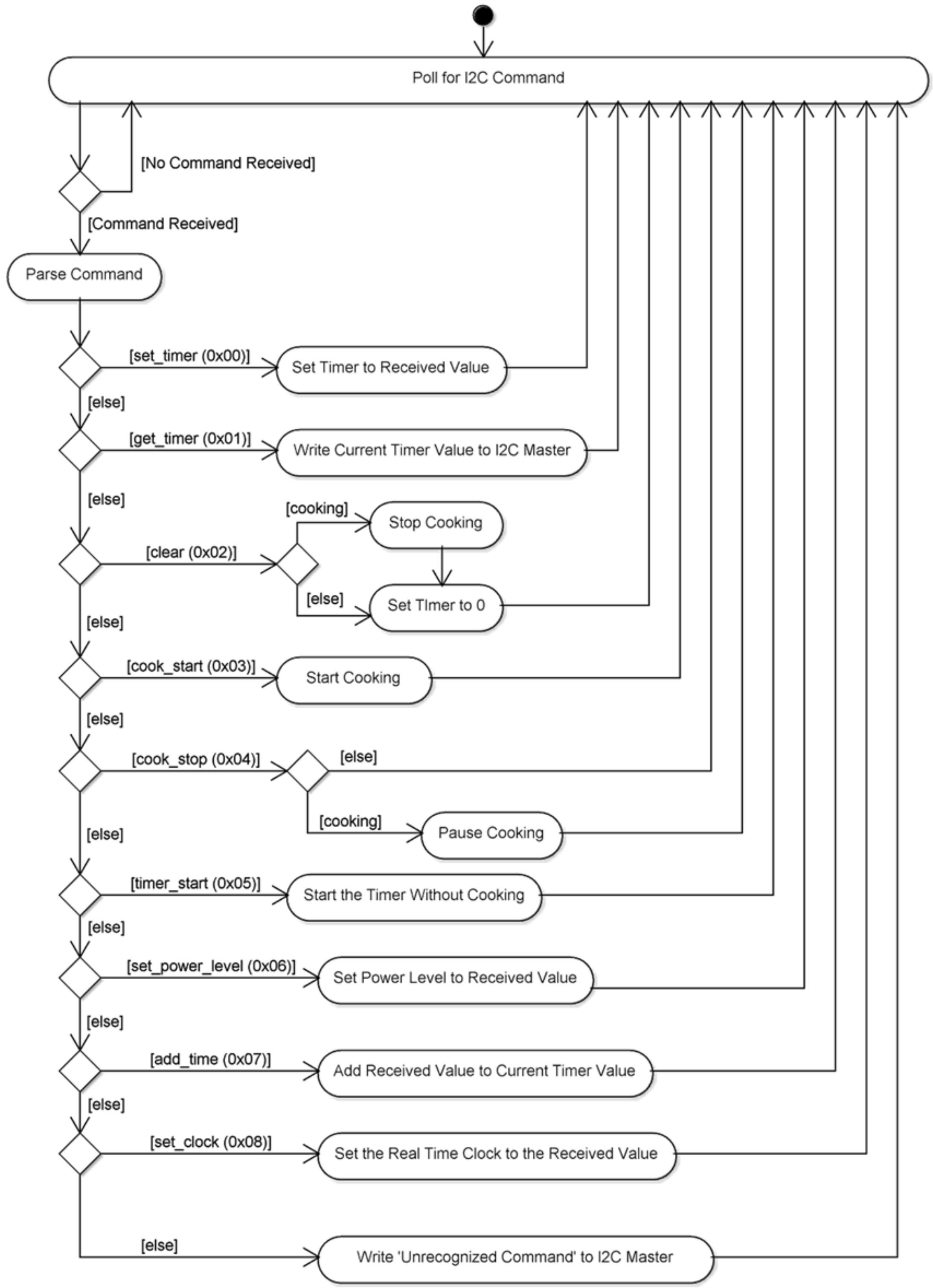


**Figure 4.3.3-1:** Microwave control board I2C command poll sequence

On the Beaglebone Black, we have a single class that allows for the rest of the software to interface with the microwave control board. This class implements all of the I2C communications between the Beaglebone Black and the microwave control board and provide a simple interface for the rest of the software to control the microwave hardware. A UML diagram of this class is shown in Figure 4.3.3-2. The UML diagram also describes an enumeration that contains the possible states of the microwave.

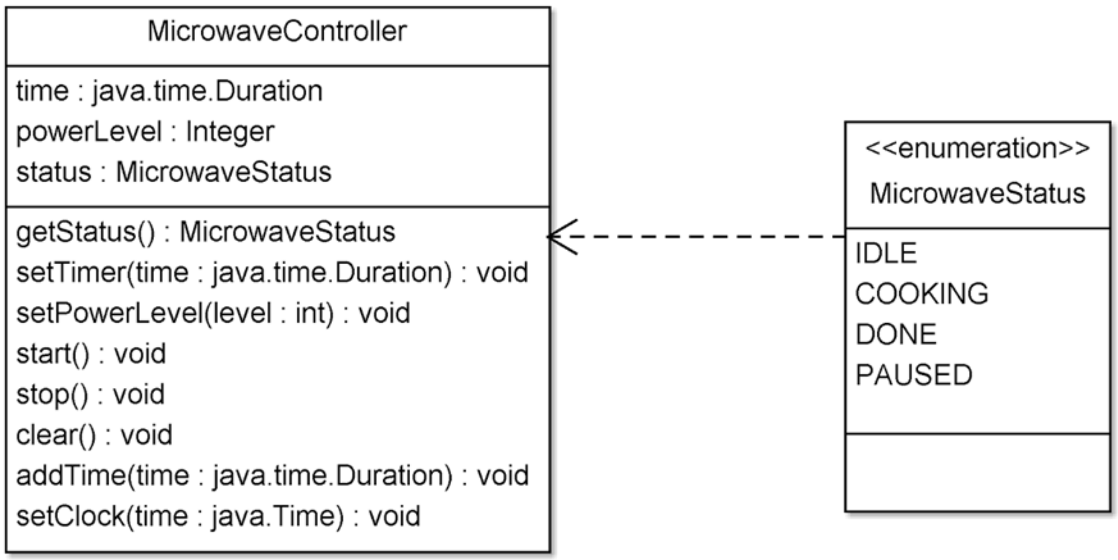


**Figure 4.3.3-2:** MicrowaveController UML Diagram

There is only one microwave control circuit, so we have to ensure that there can only be one instance of the Microwave Controller class. This could be accomplished by making all of the methods static, but this would allow any class in any thread to gain access to the microwave hardware. Another option would be to implement the class in a singleton pattern. Instead of having a public constructor like a normal class, this class stores a single instance of itself as a private static variable. When another class would like to gain access to this instance, it will call a method that returns a reference to this instance, or the method will create the instance if it has not yet been created. Implementing the class in this way allowed for us to control what parts of the software can gain access to the microwave hardware.

## 4.3.4 Barcode and QR Code Software

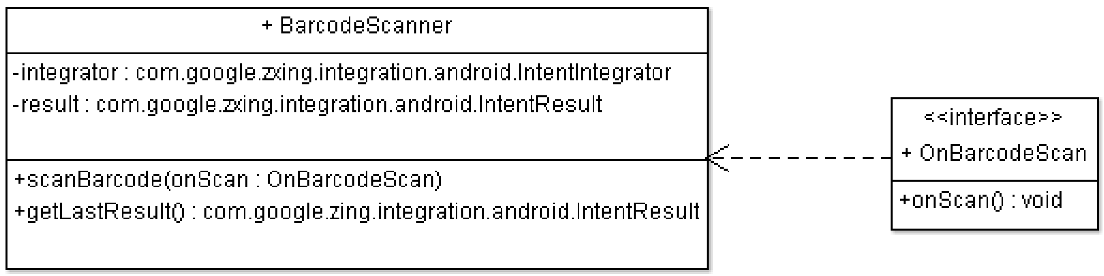


**Figure 4.3.4-1:** BarcodeScanner class diagram

The barcode scanning was done using the BarcodeScanner class, which used the ZBar barcode library to scan barcodes and QR-codes using the Memrowave's camera. A class diagram of the BarcodeScanner class is shown in Figure 4.3.4-1, along with an interface for the OnBarcodeScan callback.

## 4.3.5 Local Product Database

Android uses local SQLite databases to store application data. SQLite is a software library that implements a self-contained, server less SQL database engine.

The structure we've designed for the Memrowave's local product database is shown in Figure 4.3.5-1. The database called *Products* has a table called *Product*. This table has a row for each product. This row will have a unique id, the product's upc, the product's name, etc. The *has_steps* column defines whether or not the product has any steps. If this is false, the specified *power_level* and *cook_time* will be used. If it is set to true, each step will be stored as a row in a unique table in the database. The *steps_id* column in the *Product* table will store the id of the table that contains the steps. Each step has an *id*, *name*, *description*, *cook_time*, and *power_level*.

The figure also shows an example product, oatmeal. For purposes of this example, the oatmeal product will require multiple steps to prepare, so has_steps is set to true, and the cook_time and power_level fields are left blank. The steps_01 table holds all of the steps needed to prepare the oatmeal. The first row of the steps_01 table holds the first step, with an example id of 0001. The name is given as "Step 1", and the description field stores the instructions for the step. The Memrowave will be cooking for this step, as indicated by the provided cook_time and power_level. If there is another step in the steps_01 table, it will be shown on the Memrowave after the first step finishes cooking.

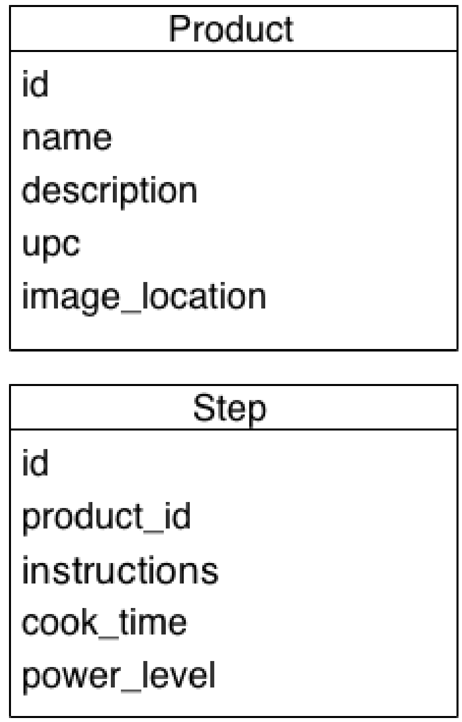


**Figure 4.3.5-1:** Local product database schema

Now, we need to define how this database was implemented in our software. We created a Product class that stores the information about a product and a Step class that holds the information about a single step. If a product has any steps, they will be stored as a list of Step objects. Then we create a singleton class that will provide access to the product database. This class provides methods to search the database and add, remove, and update any records. Figure 4.3.5-2 shows how we will implement these classes.

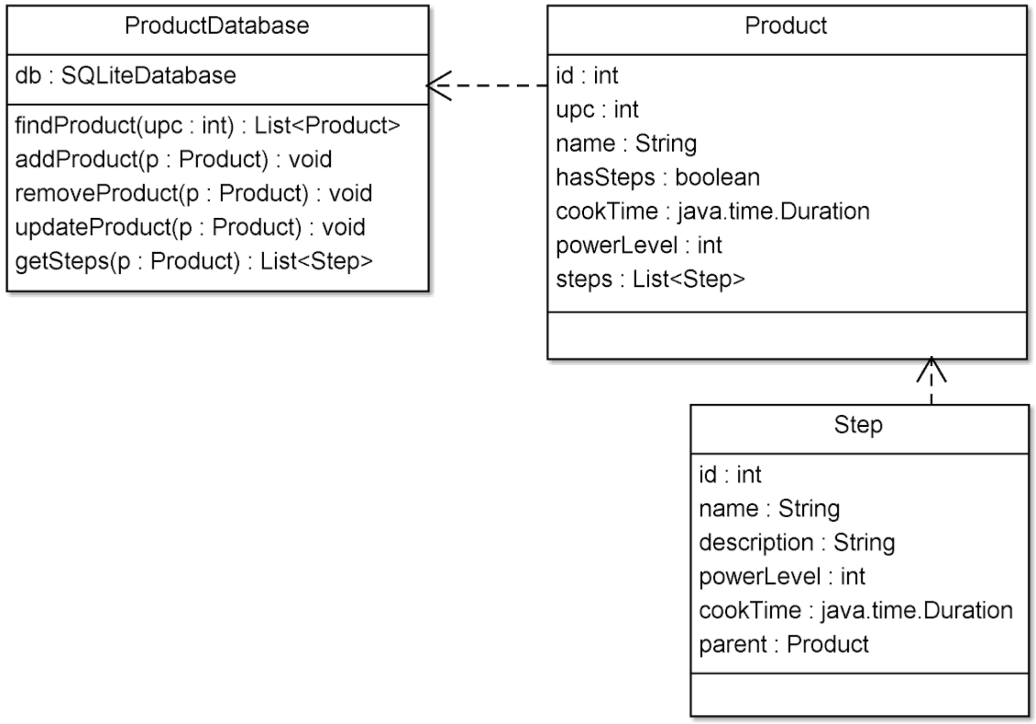


**Figure 4.3.5-2:** Local product database class diagrams

The ProductDatabase class has several methods to allow the Memrowave software to interface with the SQLite database. The findProduct method takes an integer UPC code as an argument and returns a list of products from the database that matches this UPC code. UPC codes are supposed to be unique, so it would make sense for this method to return only a single Product instance, however, a List is used to ensure any duplicate results are shown as well. The addProduct method takes an instance of the Product class as an argument and returns nothing. This class will verify the information in the Product class is formatted properly and then it will add the product to the database. The removeProduct class will remove the specified product from the database. The updateProduct will replace an entry in the database with the specified Product instance, provided that the ids match. Finally, the getSteps will retrieve the list of steps for a particular product from the database and return it.

### 4.3.6 User Interface

The primary means of using the Memrowave is a touch screen interface and not physical buttons like older microwaves. Touch screen interfaces should be reasonably familiar with the majority of users today due to the proliferation of handheld smart devices such as cellular phone and tablets. With this in mind, conventions that are used by modern communication devices, are used for the

Memrowave interface where feasible. Interface design is kept simple while also trying to avoid driving too deeply into menus or requiring too many steps to accomplish a simple task. Breaking conventions used for current microwaves should be avoided unless there is a clear advantage to make a change. For example, there is no need to change the orientation of the number keypad from what is currently used on microwaves to the keypad used for calculators.

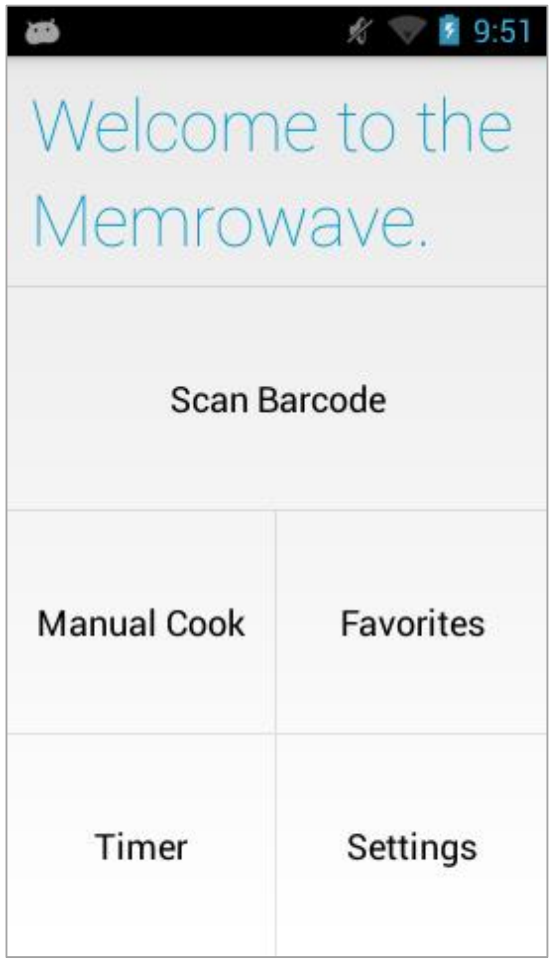


**Figure 4.3.6-1:** Main menu/home screen

The first screen that a user will see is the Memrowave's main menu or home screen. This screen remained simple with a clear focus on what makes the Memrowave unique, that is the ability to scan and automatically prepare food products. Figure 4.3.7-1 is a simple view of the main menu as seen by the user.

From the home screen, the user is given five options: Scan a barcode, use the manual cook feature, view a list of favorite products, start a timer, and change the system settings.


**Figure 4.3.6-2:** Barcode Scan Activity screenshot

The scan barcode button brings up the BarcodeScanActivity shown in Figure 4.3.7-2. This activity uses the camera and the Zbar library to automatically scan and decode any barcode in frame. When a scan is successful, the UPC will be used to search for matching products in the local and web databases.

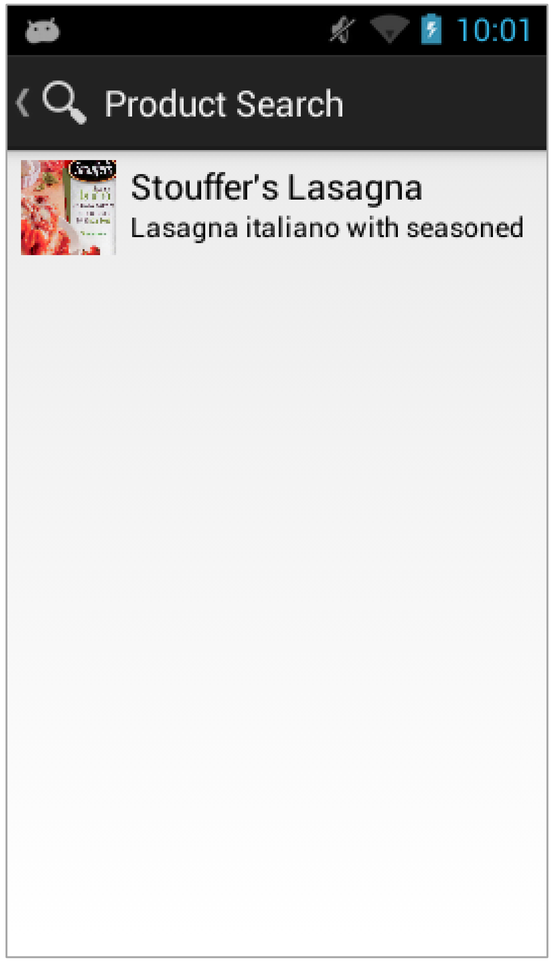


**Figure 4.3.6-3:** Product Search Activity screenshot

The product search screen is shown in Figure 4.3.6-3. This activity was used to implement both the product search and favorites features.


**Figure 4.3.6-4:** Product Description Activity screenshot

From the product search screen, the user may select a product to obtain a more detailed view of the products information. This is accomplished via the ProductDescriptionActivity, shown in Figure 4.3.6-4.

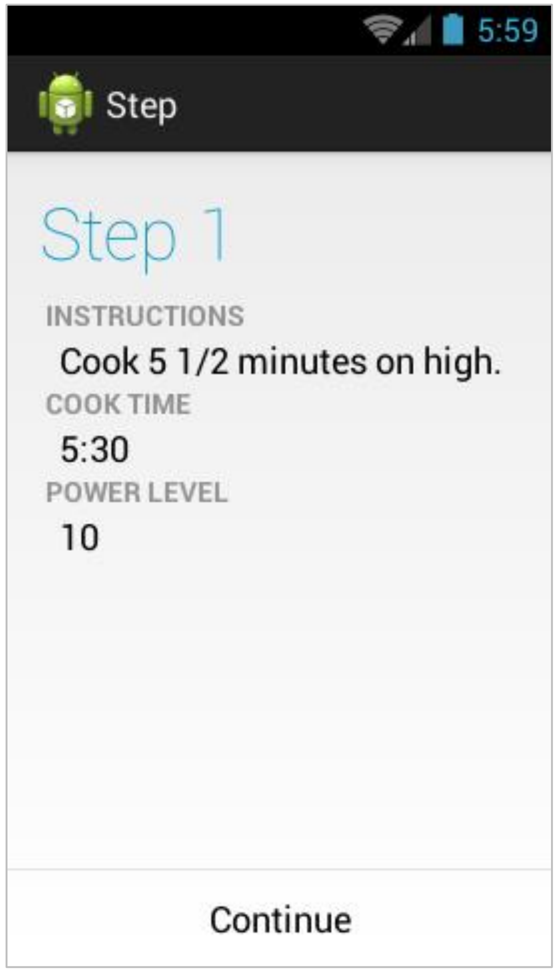


**Figure 4.3.6-5:** Step Description Activity screenshot

The ProductDescriptionActivity shows all of the information about a particular product, including the name, description, image, and any associated cooking steps. The ‘Cook’ button shown on the bottom of the screen will start the StepDescriptionActivity for the first cooking step. A screenshot of this Activity is shown in Figure 4.3.6-5.

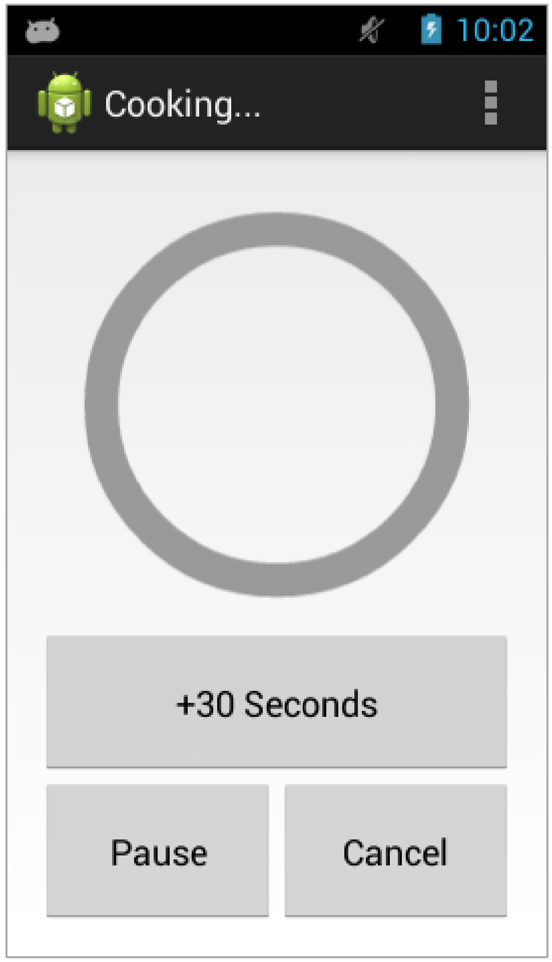


**Figure 4.3.6-6:** Cook Activity screenshot

Pressing the ‘Continue’ button on the StepDescriptionActivity will cause the Memrowave to begin cooking the product at the appropriate power level, for the specified amount of time. A screenshot of the CookActivity that handles the cooking process is shown in Figure 4.3.7-6.

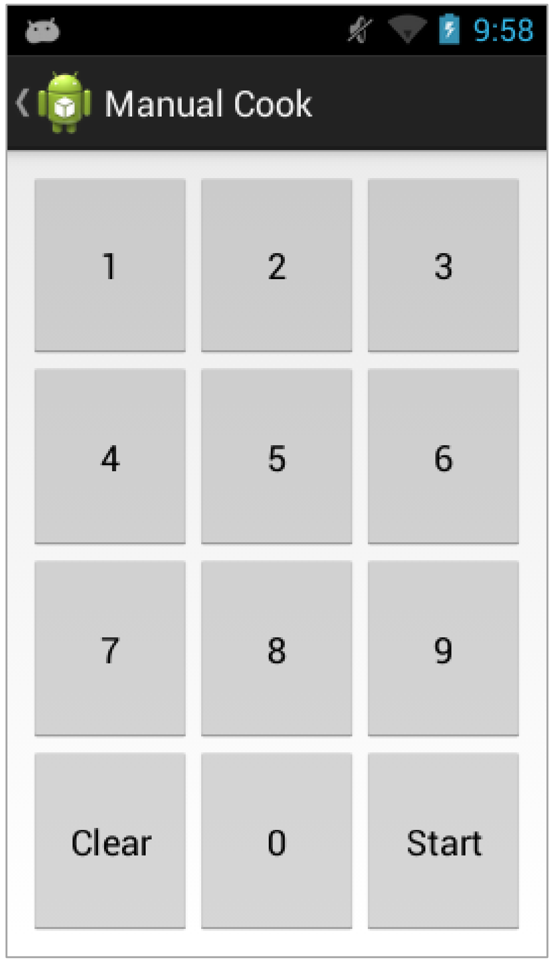


**Figure 4.3.6-7:** Manual Cook Activity screenshot

From the home screen, the user is given the option to operate the microwave manually, like a normal microwave. This functionality was implemented in the ManualCookActivity shown in Figure 4.3.6-7.

After the user enters the desired cook time and presses the start button, the Cook Activity will be started with the appropriate parameters.

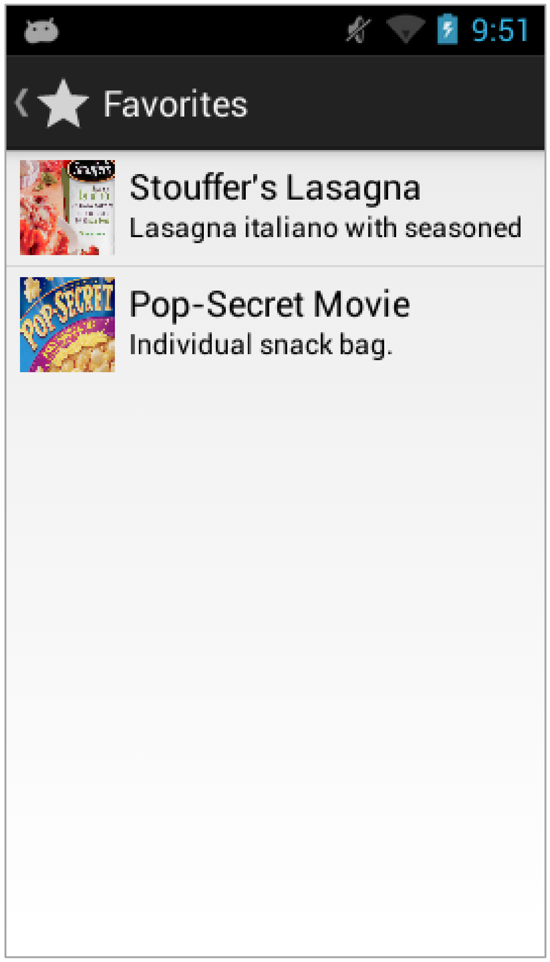


**Figure 4.3.6-8:** Favorites screen

On the home screen, the user is also given an option to view favorite products. The ProductDescriptionActivity shown in Figure 4.3.6-4 has a star icon in the ActionBar that allows the user to favorite a product. When this is done, the product_id is added to the favorites table of the database. The favorites screen retrieves a list of products from the favorites table and displays the appropriate products in a modified instance of the ProductSearchActivity. A screenshot of the favorites screen is shown in Figure 4.3.6-8.

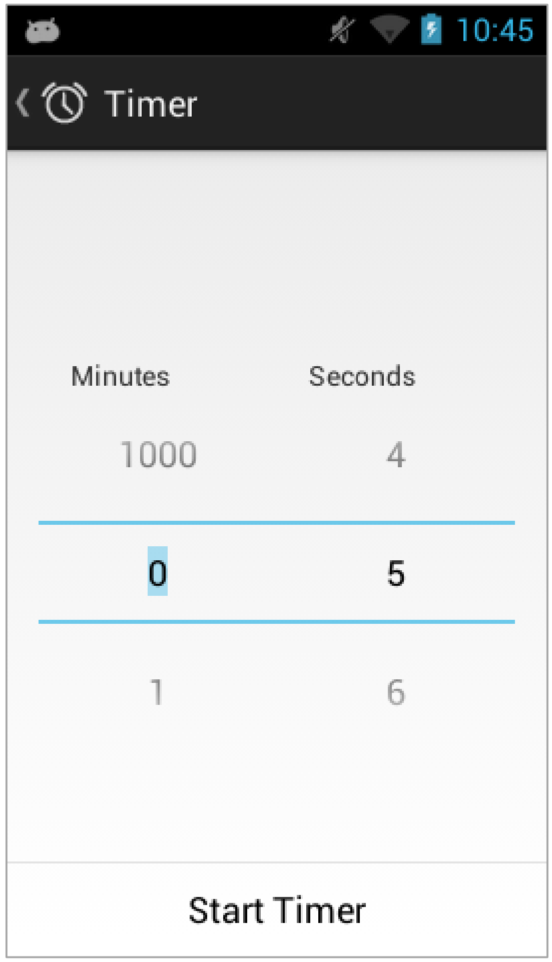


**Figure 4.3.6-9:** Timer Activity screenshot

In addition to allowing the user to view favorite products, the home screen also allows the user to start a timer. The timer functionality reuses the CookActivity to count down using the Memrowave's hardware, but it never actually starts the magnetron or turntable. The amount of time to count down is entered via the TimerActivity shown in Figure 4.3.6-9.

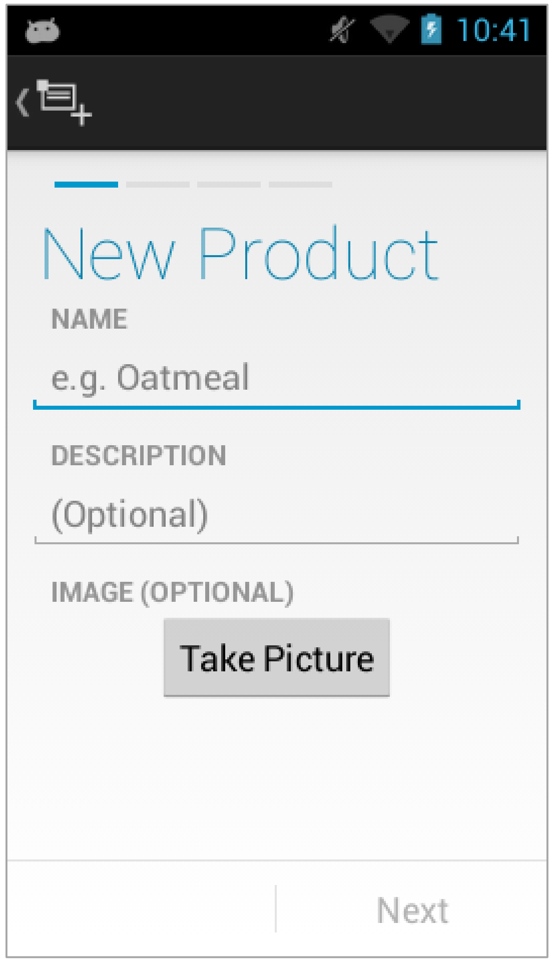


**Figure 4.3.6-10:** Product Wizard Activity screenshot

The ProductWizardActivity is used to enter new products into the local database, and to edit existing products. For a new product, the UPC is automatically entered from the BarcodeScanActivity, but the rest of the information must be manually entered by the user. This information includes the Product's name and description, as well as any cooking steps. The user is given the option to use the Memrowave's built in camera to take a picture of the product they are adding, via the 'Take Picture' button.

### 4.3.7 SMS Notifications

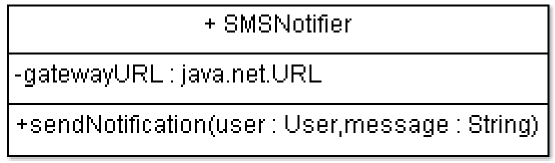


**Figure 4.3.7-1:** SMSNotifier class diagram

Sending SMS messages over the internet requires an SMS gateway server. There are commercial options available, but these usually require a monthly fee and charge for each message sent. After reviewing the various options available to us, we decided to use the Twilio SMS service. Twilio allowed us to send text messages very easily, just by doing standard HTTP requests to their web API. It was also free to user their trial account, with the only downside being that each message sent also included additional text specifying that it was a trail account. However, this was okay for our purposes. Figure 4.3.8-1 specifies the class that was used to implement the SMS notifications.

# 5 Design Summary

In section 4, we discussed the design details of the Memrowave for each subsystem. In this section, we will discuss the design of the overall system that will integrate all of the subsystems.

## 5.1 Software Design Summary

In section 4.3.6, we outlined the software design details of each part of the system. In this section, we will combine all of the subsystems into an overall design summary of the Memrowave's software. This design summary will show how we will implement the main program, which ties all of the separate parts together.

The main entry point to the program acts as an Android 'Launcher'. This brings up the Memrowave's main interface whenever the Home button is pressed on the device's screen, effectively blocking the user form encountering the standard Android interface. Figure 5.1-1 shows class diagrams for several of the Android 'Activities' that make up the program.

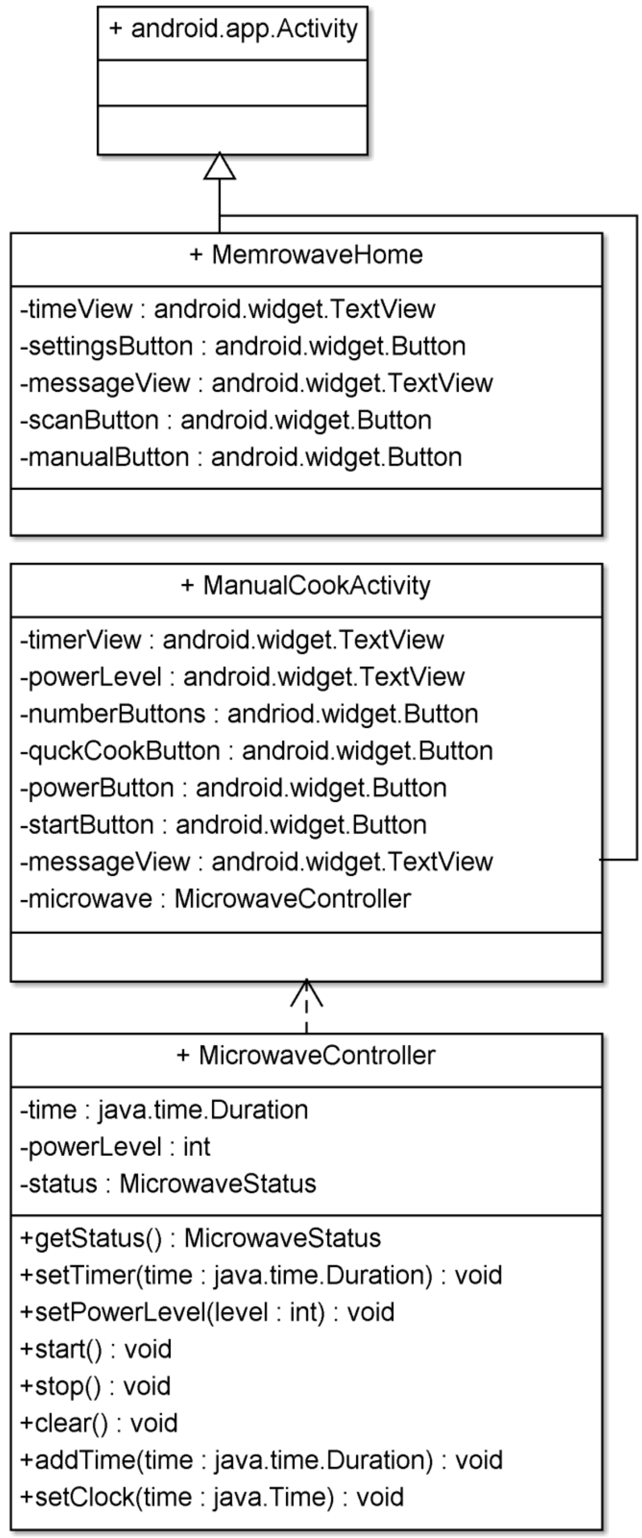


**Figure 5.1-1:** Class diagrams home screen, manual cook screen, and microwave controller class

The MemrowaveHome class acts as the Launcher replacement, so anytime the Home button is pressed, this Activity will be displayed on the screen. The user interface of the Activity was described in detail in section 4.3.7, and a mock-up of the screen was shown in Figure 4.3.7-1. Each element of the mock-up has its corresponding android-specific UI control element as an attribute in the class diagram. Also in Figure 5.1-1 is a class diagram for the ManualCookActivity. This class describes the screen that is displayed when a user selects the 'Manual Cook' option on the main screen. Like the MemrowaveHome class, this class has UI control elements for each of the control elements of the mock-up shown in Figure 4.3.7-3. In addition to these attributes, the ManualCookActivity has a reference to the MicrowaveController class, which allows for it to send commands to the microwave hardware. A class diagram of the MicrowaveController class is shown in Figure 5.1-1 as well. This class will use the I2C interface of the Beaglebone Black to send commands to the microwave control board, as described in detail in section 4.3.3.

Figure 5.1-2 shows three different Android Activity classes, including the BarcodeScanActivity class, ProductCookActivity class, and QuickCookActivity class. The BarcodeScanActivity class is used to implement the barcode scanning screen of the Memrowave. A mock-up of this screen is shown in Figure 4.3.7-6 and Figure 4.3.7-7. The class diagram lists all of the Android-specific UI control elements that will be needed to implement this class, and it also includes a reference to the BarcodeScanner class, which processes the camera images to scan the user's barcode.

After a user scans a barcode and a product match is found, the ProductCookActivity is displayed on the screen. A mock-up of this screen is shown in Figure 4.3.7-10. The ProductCookActivity class will implement the product cook screen. The displayed information about a particular product, including the cook time, power level, name, etc. The ProductCookActivity gives an option to modify the product's database entry to change the cook time and power level. Finally, the ProductCookActivity will have a button that will start the cooking process.

The QuickCookActivity class, which is also shown in Figure 5.1-2, displays a list of food products that may not necessarily have a barcode, and yet can be cooked automatically by the Memrowave, nonetheless. The class has a reference to the QuickCookDatabase class, which will access the local database to retrieve all of the quick cook product options. The list of quick cook items will then be displayed on the android.widget.ListView UI control element. When the user selects an item from this list, the callback method defined in the OnItemClickListener will be executed. This will load the ProductCookActivity with the selected quick cook product.

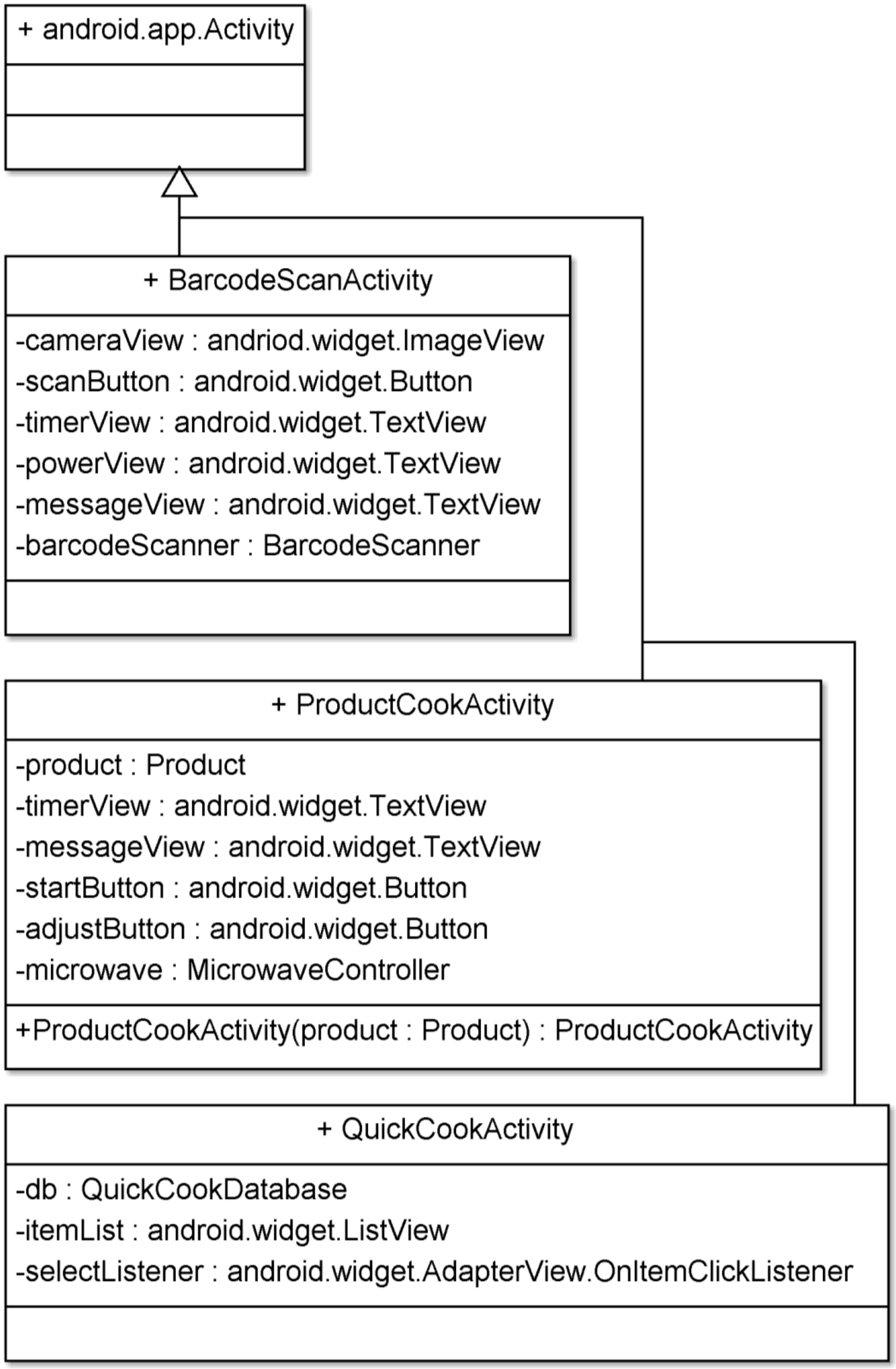


**Figure 5.1-2:** Class diagrams for the barcode scan screen, product cook screen, and quick cook screens

The NewProductActivity class, shown in Figure 5.1-3, will display a screen that allows the user to enter a new product into the local product database. A mock-up of this screen was shown in Figure 4.3.7-15.The class has references to a TextView UI element for the product name, a TimePicker UI element for the cook time selection, a NumberPicker UI element for the power level selection, and a few buttons, including a button to save the new product to the database and a button to scan the barcode of the new product. When the user presses the save button, the information entered in the UI elements will go through a verification process to be sure it's formatted correctly, and then the product will be added to the local product database using the ProductDatabase class reference.

The next class in Figure 5.1-3 is the ProductAdjustmentActivity. This class gives the user the option to modify the cook time and power level of a particular product. A mock-up of this screen is shown in Figure 4.3.7-17. This class is similar to the NewProductActivity class, except the only user-modifiable options will be the cook time and power level.

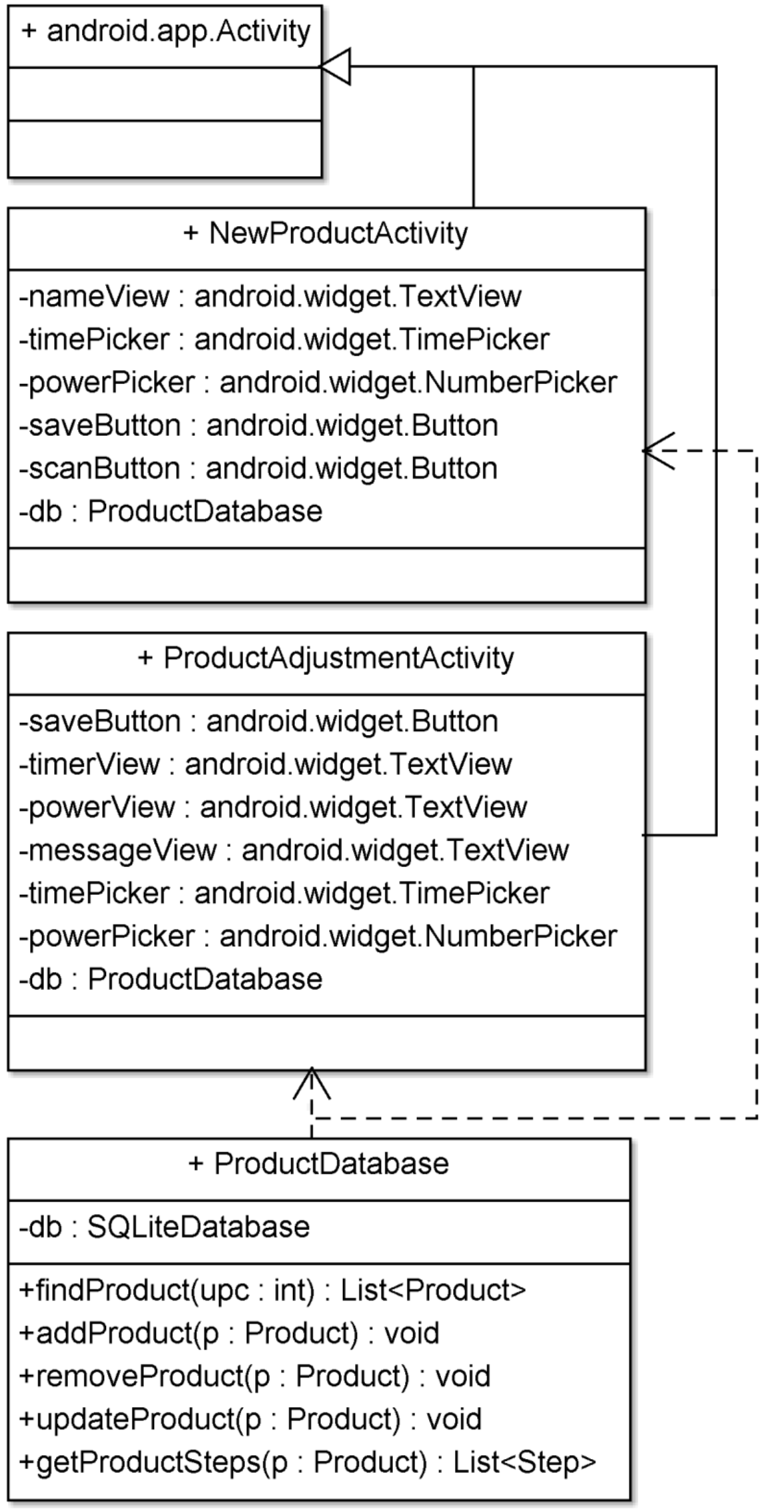


**Figure 5.1-3:** Class diagrams for the new product screen, modify product screen, and the product database

## 5.2 Hardware Design Summary

In this section, provide a high-level overview of the hardware of the Memrowave.

A schematic diagram for the microwave control board is shown in Figure 5.2-1. The schematic has the MSP430 microcontroller that wihch controls the microwave's hardware. The I2C pins on the MSP430 will be connected to the I2C pins of the BeagleBone via the header shown in the schematic. The MAX6958 LED display driver is also shown in the schematic. This part uses 5V logic for the serial communications, while the BeagleBone Black uses 3.3V. To make the two work together, a logic level shifter circuit was used for each of the three serial communication lines. The LED display will be connected to the board via a ribbon cable, so the relevant pins of the MAX6958 were connected to a 16 pin header in the schematic.

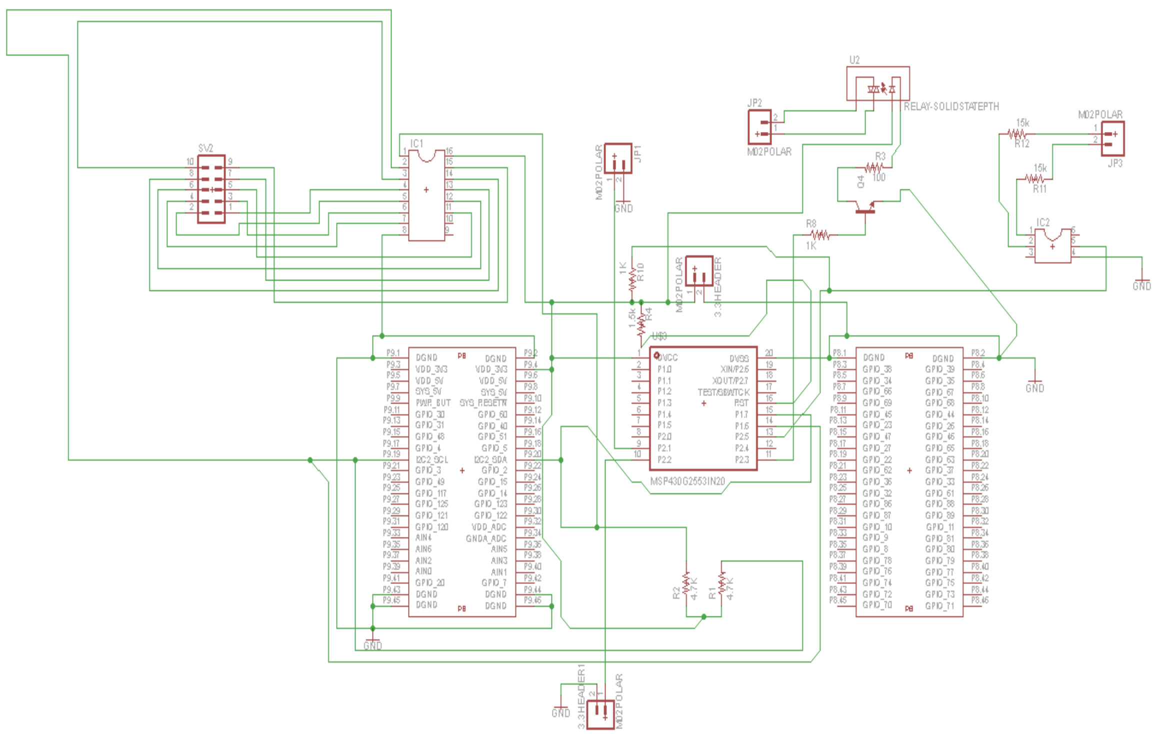


**Figure 5.2-1:** Microwave control board schematic

Figure 5.2-2 shows the PCB design for the schematic in Figure 5.2-1. Design files from SparkFun Electronics' BeagleBone Black prototyping cape were modified to create our own custom board design. SparkFun Electronics made their design files for the prototyping cape available under the CC BY-NC-SA 3.0 license, which gives us permission to use the design files in our project. All of the parts that were not relevant to our project were removed from the design files, and our own circuits were then inserted in their place. We then positioned the parts on the board and routed the appropriate connections.Figure 5.2-3 shows how the circuit from Figures 5.2-1 and 5.2-2 was connected to the various microwave components.

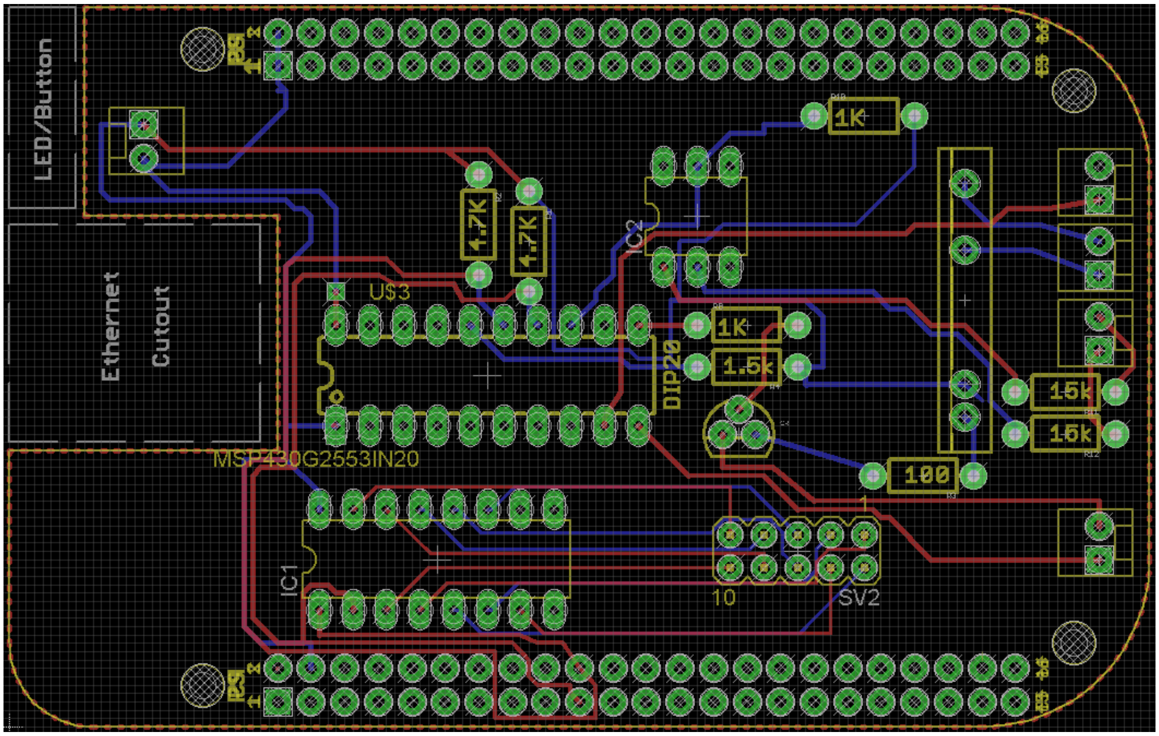


**Figure 5.2-2:** Microwave control circuit ‘Cape’ for the BeagleBone Black

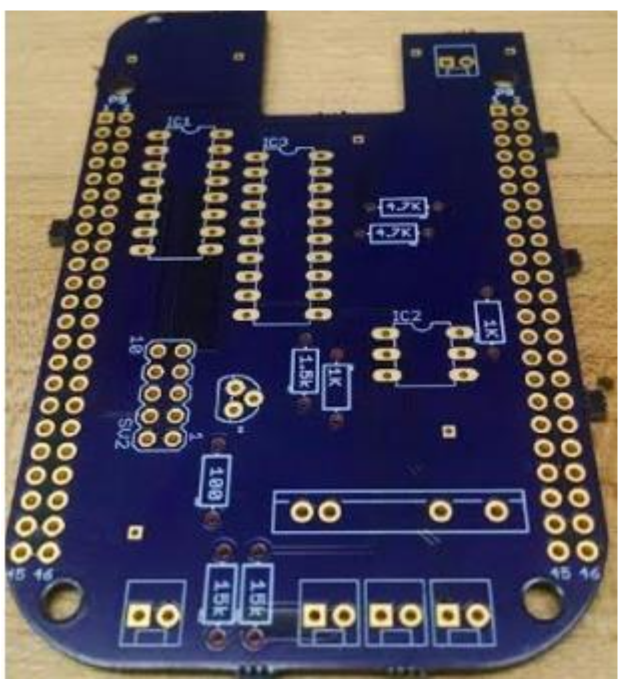

**Figure 5.2-3:** The fabricated PCB

# 6 Project Prototype Construction and Coding

In section 6, we will discuss how the design of the Memrowave was implemented.

## 6.1 Parts Acquisition and BOM

An outline for the specific parts and components for our project has been developed. All the parts were divided into sections and will have detailed information based on the following criteria:

- Division between hardware and software application.
- Relative importance of the specific item.
- Difficulty in acquiring the item.
- Part manufacturers and locations.
- Engineering time frame for receiving the item.

This outline will serve the basis for what and who was responsible for the certain components and parts are required for the Memrowave. The division between software and hardware is obvious due to the need for separate hardware and software design that is necessary and later integrated together. In our project many of the components were necessary such as the power supply and processing unit, while other are of lesser importance like a specific type of mounting bracket. Giving each part or component a certain level of importance helped in setting a certain value for getting the product developing a plan for acquiring it in a timely fashion. Part manufacturers and locations played a part in acquiring certain items. Ideally we got all our parts within the United States in order to secure relatively quick turnaround shipping times and to limit any issues with receiving goods from overseas. An engineering time frame for the each part and component is laid out in order to setup a starting block for each item as it is needed for integration into the final engineering the Memrowave and to keep an up to date schedule throughout the development phase.

The working outline on the next page shows many of the parts and components needed for the Memrowave as well as some of the specifics details about acquiring such items:

| Parts | Division | Priority | Engineering Time Frame | Manufacturers/Retailers | Acquisition Difficulty |
|---|---|---|---|---|---|
| **Processing Unit** | Software | High | Early | Texas Instruments; Sparkfun Electronics; Arduino; Intel | Low |
| **LCD Touch Screen Module** | Software | High | Early | Sparkfun Electronics; Arduino; 4D Systems; Digi-Key | Low |
| **Camera Module** | Software | High | Early | Sparkfun Electronics; Digi-Key; E-Con Systems; Byd IT | Low |
| **WIFI Module** | Software | High | Early | Sparkfun Electronics; Digi-Key; Whiznets; Microchip Technology | Low |
| **AC-DC Converter** | Hardware | High | Middle | Delta Electronics; Mouser; Sankan Electric; CUI Inc. | Low |
| **Voltage Regulator** | Hardware | High | Middle | Delta Electronics; Mouser; Sankan Electric; CUI Inc. | Low |

**Table 6.1-1:** Parts Acquisition

Many of the common place parts in or on a regular microwave such as the turntable and lights were salvaged from the microwave we used. A list of these items are shown below:

- Internal Lights
- Internal Beeper
- Clock Display
- Magnetron
- Power switch
- RF shielding
- Door assemblies

Bill of Materials

The bill of materials includes all items that will need to be purchased in the course of the conceptualizing phase of the Memrowave. Details about certain miscellaneous items will be hard to pin down due to unforeseeable changes and alterations which may need to be done such as additional supports for assembling and mounting parts, wiring details. Due to the nature of some of the components and parts, whether to design a part itself or to buy the part also plays a factor in the bill of material list based on available options, the flexibility, usable, and cost effective must be considered. Below in Table 6.1-2 is a pervious working outline of all the components and parts that need to be purchased with typical prices, as well as money which will be allocated to certain items were a specific monetary amount can’t be given at this time.

| Bill of Materials | | | |
|---|---|---|---|
| Item(s) | Cost Per Item | Quantity | Total |
| Processing Unit | $60 | 1 | $60 |
| WIFI Module | $23 | 1 | $23 |
| Camera Module | $32 | 1 | $32 |
| LCD Touchscreen | $140 | 1 | $140 |
| Power Switch | $17 | 2 | $34 |
| Wire | $30 | 1 | $30 |
| Total Estimated Cost | | | $353 |

**Table 6.1-2:** Rough Bill of Materials Estimate

## 6.2 PCB Vendor and Assembly

For the Memrowave a reliable printed circuit board assembly is needed for the additional components we will be adding to the Memrowave. Since the PCB will be mainly for the power supply and power regulation, the design of the PCB will most likely be of one layer, measuring at most a maximum of 4 inch by 3 inch to keep as compact a design as possible while allowing space to fit all the needed board-level parts. A minimum of three boards was desired for design and testing purposes. In the final stages of the design and assembly another board would be desirable after any changes that need to be made. The options we have in making this PCB is either to do self-fabrication or to use a PCB vendor and/or assembler to do the fabrication and/or assembly for us. Location

- Turn around/Delivery times
- Cost based on our needs
- PCB design software choices
- Assembling capabilities

These criteria set the basis for making the preliminary and final decision on which PCB vendor will be most ideal for this project. The turnaround time will be crucial as we have a limited time frame for testing and assembling the final appliance.

The following outline is a list of the possible vendors we could have used and the details described to make a decision which PCB vendor is ideal.

| Vendor | Location | Turn-Around Time (Days) | Cost | Software Choices | Assembling Capabilities | Other Services |
|---|---|---|---|---|---|---|
| **PCB-Pool** | Vacaville, California | 1-8 | $60 | Eagle CAD; PCAB; Target; Sprint | No | Electrical Testing |
| **OSH Park** | Portland, Oregon | 12 | $60 | Eagle CAD; Gerber CAM files | No | |
| **ExpressPCB** | Santa Barbara, California | 3 | $97 | Proprietary Software | No | |
| **Advanced Circuits** | Tempe, Arizona; Maple Grove, Minnesota; Aurora, Colorado | 5 | $99 | PCB Artist | Yes | Electrical Testing |

**Table 6.2-1:** PCB Vendors

# 6.3 Final Coding Plan

For the final coding plan, we divided the software into sections and assigned each section to a group member. Winston Todd and Jack Gulick are in charge of the software, so the sections will be split between them. Jack was be put in charge of developing the software for the microwave control board described in section 4.3.3. This included developing software to control all of the microwave components in the correct sequence and developing the software to meet the I2C interface specifications listed in section 4.3.3. He also was in charge of programming the user interface described in section 4.3.7. This included programming all of the UI controls for each Activity listed in the class diagrams shown in Figure 5.1-1.

Winston was in charge of implementing the barcode scanner class described in section 4.3.4. This entails interfacing the ZBAR barcode software library with the Memrowave's software. He was also in charge of implementing the software to interface with the microwave control board, as described in the class diagram in Figure 4.3.3-2. Section 4.3.5 describes the local product database and provides class diagrams in Figure 4.3.5-2. Winston was in charge of implementing these classes to meet those specifications. In addition to the local database, Winston was in charge of implementing a master database server.

After implementing those software sections, we will work together to combine all of those subsystems into the final software system. Section 5.1 shows class diagrams of all of the class that need to be implemented. These classes will tie together all of the subsystems into the final product.

# 7 Project Prototype Testing

It is important to create well designed tests for the project to ensure that any bugs in either the hardware or software will be caught. With this in mind, we will proceed to detail the testing of both the Memrowave's hardware and its software.

## 7.1 Hardware Test Environment

In this section we will discuss the testing environment for our prototype's hardware. We will provide information on the testing equipment available to our design group, what equipment will be used in testing, the labs we have access to, and go over possible beneficial test to run. Understanding what we have at our disposal will better help our testing during the prototype design phase. We need to present a clear list of possible equipment that can aid in troubleshooting problems with our design. The first device to consider is a signal generator. The signal generator function as a source for input signals. It allowed for us to test to analyze multiple different input responses such as trigonometric, sync functions, square waves, and much more. The importance of the signal generator in our testing environment was its ability to input these wide range of inputs.

Testing the outputs of the microcontrollers was accomplished by using a DC power supply. Allowing for our design group to testing input/output pins on our controllers. We also needed the DC power supply during testing of our switching circuits. These circuits required biasing power to operate. During testing of these subsystems via bread board the DC power supply is required to bias elements.

The most imported piece of equipment used when testing hardware was the oscilloscope. The oscilloscope was used to measure and record output signals from our sub systems. For example the oscilloscope can be used to characterize noise on subsystems. Figure 7.1-2 depicts the noise from a 5 volt offset. As shown the ripple (Vpp) is measured in decibels.

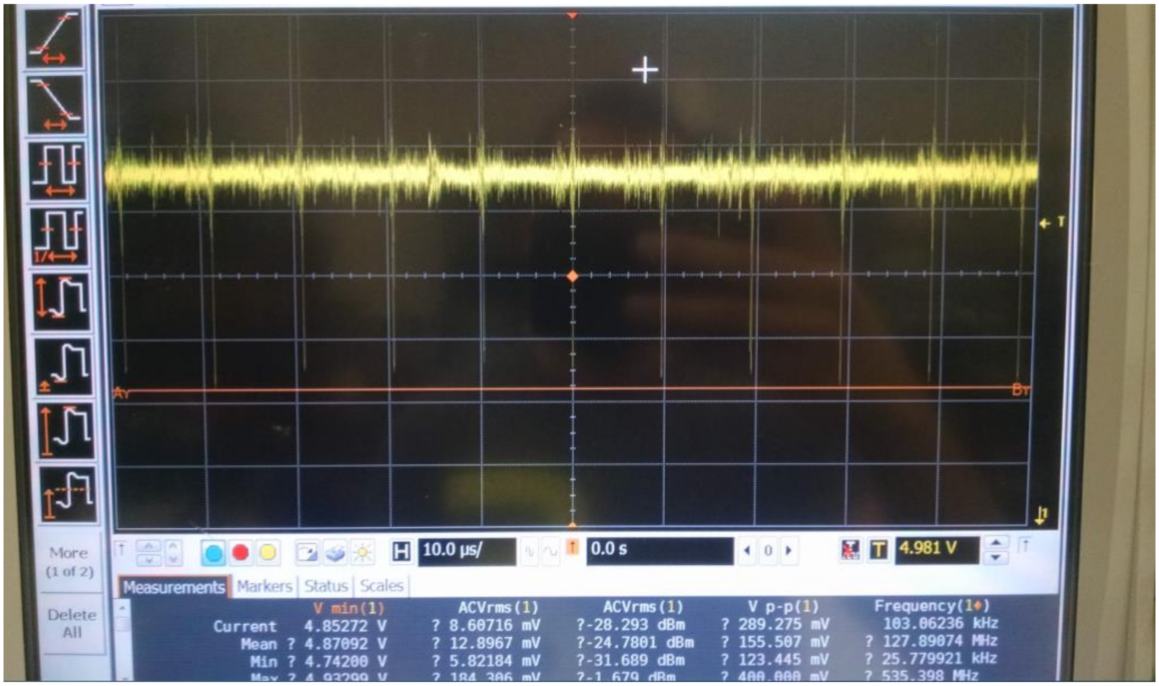


**Figure 7.1-1:** Noise on DC offset

Now that we have a proper list of equipment that was useful for characterizing our design, testing the hardware, and subsystem integration we can discuss briefly our testing locations. These laboratory locations provided the equipment necessary for testing our design. Table 7.1-1 lists the laboratories and the equipment available.

| Laboratory | Equipment |
|---|---|
| Senior Design Lab (Eng 1, Room 456) | Tektronix MSO 4034B Digital Mixed Signal |
| | Oscilloscope, 350 MHz, 4 Channel |
| | Tektronix AFG 3022 Dual Channel Arbitrary Function Generator, 25 MHz |
| | Tektronix DMM 4050 6 ½ Digit Precision Multimeter |
| | Agilent E3630A Triple Output DC Power Supply |
| | Dell Optiplex 960 Computer |
| EECS Smart Lab (HEC, Room 384) | Oscilloscope, 350 MHz, 4 Channel |
| | Tektronix AFG 3022 Dual Channel Arbitrary Function Generator, 25 MHz |
| | Tektronix DMM 4050 6 ½ Digit Precision Multimeter |
| | Agilent E3630A Triple Output DC Power Supply |
| | Dell Optiplex 990 Computer |
| CAAT Lab (Eng 1, Room 234, limited access to members of CAAT) | Spectrum Analyzer |
| | Sweep Generator |
| | Tektronix DPO4104B Oscilloscope |
| | RF power meter |

**Table 7.1-1:** Testing equipment list

A list of the possible test can be shown in Table 7.1-2

| Test | Equipment |
| --- | --- |
| Voltage Regulators variation | Agilent E3630A Triple Output DC Power Supply |
| | Tektronix DPO4104B Oscilloscope |
| MOSFET Switch responses | Oscilloscope, 350 MHz, 4 Channel |
| | Tektronix AFG 3022 Dual Channel Arbitrary Function Generator, 25 MHz |
| | Tektronix DMM 4050 6 ½ Digit Precision Multimeter |
| Filter Responses (If needed) | Spectrum Analyzer |
| | Sweep Generator |
| | Tektronix DPO4104B Oscilloscope |
| Magnetron Power Output | Tektronix AFG 3022 Dual Channel Arbitrary Function Generator, 25 MHz |
| | RF power meter |

**Table 7.1-2:** Testing Overview

## 7.2 Hardware Specific Testing

This section enclosed specific test the design team will conduct during our prototyping stages.

## Test Name:
Characterization of Memrowave’s voltage regulators

**Objective:**
The objective of this experiment is to characterize any voltage regulators used in our system. The voltage regulator will be tested to determine if the output is the desired value. As well as testing to see if the output value is consistently where we would expect it to be. DC output may still vary by a percent and we want to know how often and how far off our output voltages may be.

**Supplies:**

- Tektronix DPO4104B Oscilloscope
- Agilent E3630A Triple Output DC Power Supply
- Tektronix DMM 4050 6 ½ Digit Precision Multimeter

**Preparation:**

1. The voltage regulator is removed from packaging and place on an antistatic pad.
2. The Tektronix DPO4104B Oscilloscope is powered up.
3. The Agilent E3630A Triple Output DC Power Supply is powered up.
4. The Tektronix DMM 4050 6 ½ Digit Precision Multimeter is powered up

**Procedure:**

1. Voltage regulator is bias with the required input voltage Vin
2. The regulator is manually adjusted till the required output voltage Vr is received using the Tektronix DMM 4050 6 ½ Digit Precision Multimeter
3. Vr is inputted to the Tektronix DPO4104B Oscilloscope
4. The Oscilloscope is set to display a histogram of Vr from port 1

**Expected Result:**
The histogram is expected to display a large frequency around Vr. If Vr is highly consistent the voltage regulator will be used. If the histogram is not frequent around Vr we will replace the voltage regulator.

## Test Name:
Characterizing MOSFET switches

**Objective:**
The objective of this experiment is to characterize any MOSFET switches used in our system. The MOSFET switch will be tested to determine if the output is the desired value set. As well as testing to see if the output value is consistently where we would expect it to be. Each MOSFET may vary by a percent and we want to know how often and how much our output voltages vary.

**Supplies:**

- Tektronix MSO 4034B Digital Mixed Signal
- Tektronix AFG 3022 Dual Channel Arbitrary Function Generator, 25 MHz
- Oscilloscope, 350 MHz, 4 Channel
- Tektronix DMM 4050 6 ½ Digit Precision Multimeter
- Agilent E3630A Triple Output DC Power Supply

**Preparation:**
1. The MOSFET and resistors are removed from packaging and place out for assembly
2. The Tektronix AFG 3022 Dual Channel Arbitrary Function Generator is powered up.
3. The Agilent E3630A Triple Output DC Power Supply is powered up.
4. The Tektronix DMM 4050 6 ½ Digit Precision Multimeter is powered up

**Procedure:**
1. The MOSFET switch is constructed on a bread board and hooked up to equipment required to test
2. MOSFET switch is bias with the required input voltage Vin
3. Vout is inputted to the Oscilloscope, 350 MHz, 4 Channel
4. The Oscilloscope is set to display the time plot of the output voltage required

**Expected Result:**
The output is expected to display a switching nature. Vout is expected to be present when the MOSFET receives its required gate voltage to switch on. This will be observed via Oscilloscope.

## Test Name:

Verifying motor controllers active turntable correctly

**Objective:**

The objective of this experiment is to check if the subsystems are integrated correctly. MOSFET switches used in our system as well a DC power supply and the turntable motor. The system will be tested to determine if integration was a success. Each subsystem will be tested in a similar manor.

**Supplies:**

- Tektronix MSO 4034B Digital Mixed Signal
- Tektronix AFG 3022 Dual Channel Arbitrary Function Generator, 25 MHz
- Oscilloscope, 350 MHz, 4 Channel
- Tektronix DMM 4050 6 ½ Digit Precision Multimeter
- Agilent E3630A Triple Output DC Power Supply

**Preparation:**

1. The motor is removed from packaging.
2. The Tektronix AFG 3022 Dual Channel Arbitrary Function Generator is powered up.
3. The Agilent E3630A Triple Output DC Power Supply is powered up.
4. The Tektronix DMM 4050 6 ½ Digit Precision Multimeter is powered up

**Procedure:**

1. The motor is connected to the switching circuit on the bread board.
2. MOSFET switch is bias with the required input voltage Vin
3. The output voltage will be monitored by the multimeter
4. The motor is observed during testing to verify correct outputs.

**Expected Result:**

The Motor is monitor visually by testers, once the MOSFET switching circuit is biased and verified to be correctly switching. The motor is expected to be powered during the changing of our switch. If the motor is nothing powered we can expect an unwanted voltage on the output of the switch.

## Test Name:
Checking for proper webcam integration to BeagleBone.

**Objective:**
The objective of this experiment is to check if the subsystems are integrated correctly. The BeagleBone black will be connected to the webcam and powered to check for a connection. The system will be tested to determine if integration was a success. LCD monitors and Wi-Fi will be tested similarly.

**Supplies:**

- Logitech Webcam C270
- BeagleBone Black Dev board
- LCD screen
- Tektronix DMM 4050 6 ½ Digit Precision Multimeter
- Agilent E3630A Triple Output DC Power Supply

**Preparation:**
1. The Beagle bone placed in testing area.
2. The Board is removed from packaging and placed on antistatic pad.
3. Power supply is set to the required voltage to power the Dev board.
4. The multimeter is used to verify the correct value.

**Procedure:**
1. The BeagleBone and webcam are connected together.
2. The BeagleBone is powered up from the power supply.
3. Expected to receive input from the Logitech C270 webcam.
4. A design member will hold up a barcode to the camera.
5. A design member will verify that the product is in sight on the LCD screen.

**Expected Result:**
The outcome for the test is to verify that all the involved subsystem are communicating with each other after integration. We expect for the webcam to be displayed on the LCD screen.

## Test Name:

Characterization of system noise floor

**Objective:**

The objective of this experiment is to characterize the noise floor of our system. The noise floor will be tested to determine the output power in dB caused by the thermal noise of the environment. Having information on the noise floor of our system will greatly improve our ability to predict outputs.

**Supplies:**

- Tektronix DPO4104B Oscilloscope
- Agilent E3630A Triple Output DC Power Supply
- Tektronix DMM 4050 6 ½ Digit Precision Multimeter

**Preparation:**

1. The system is not connected to any power supply
2. The Tektronix DPO4104B Oscilloscope is powered up.
3. The Agilent E3630A Triple Output DC Power Supply is powered up.
4. The Tektronix DMM 4050 6 ½ Digit Precision Multimeter is powered up
5. Isolation of the system from anything other than thermal noise.

**Procedure:**

1. The System is left off and connected to the DPO4104B Oscilloscope.
2. The Scope is set to measure in dB
3. The output of a given element is inputted to the Tektronix DPO4104B Oscilloscope
4. The value is measured and determined to be the noise floor of the system.

**Expected Result:**

The expectation of the experiment is to determine the noise caused by leaving the system open to energy being coupled in from the environment. This is what we can predict will caused our idealized values to be off by a certain percent. The thermal noise will affect the signal to noise ratio of the system.

## Test Name:

Characterization of energy coupling from magnetron to controllers.

**Objective:**

The objective of this experiment is to characterize the noise energy coupled from our magnetron to our elements such as controllers. The energy coupled to controllers may affect the inputs of circuits thus changing values. This will be tested to determine the output power in dB caused by the energy coupled from the magnetron. Having information on the energy coupling of our system will greatly improve our ability to predict how much precautions we will need to take to shield components from the RF noise from the magnetron.

**Supplies:**

- Tektronix DPO4104B Oscilloscope
- Agilent E3630A Triple Output DC Power Supply
- Tektronix DMM 4050 6 ½ Digit Precision Multimeter

**Preparation:**

1. The system is not connected to any power supply
2. The Magnetron is connected to an isolated power supply
3. The Tektronix DPO4104B Oscilloscope is powered up.
4. The Agilent E3630A Triple Output DC Power Supply is powered up.
5. The Tektronix DMM 4050 6 ½ Digit Precision Multimeter is powered up
6. Apply shielding to elements as necessary.

**Procedure:**

1. The System is left off and connected to the DPO4104B Oscilloscope.
2. The Scope is set to measure in dB
3. The output of a given element is inputted to the Tektronix DPO4104B Oscilloscope
4. Power the Magnetron and observe if energy is coupled in the form of ripple in expected ideal outputs.
5. The value is measured and determined to be the noise floor of the system.

**Expected Result:**

The expectation of the experiment is to determine the noise caused by leaving the system open to energy being coupled in from the environment and the magnetron. This is what we can predict will caused our idealized values to be off by a certain percent and determine if we need to shield our elements. The thermal noise and magnetron will affect the signal to noise ratio of the system.

## 7.3 Software Test Environment

The software testing was completed in two separate environments. We designed unit tests for each class in the software. These tests run on the Beaglebone Black through the Android Unit Testing Framework. These tests ensure that changes to the software did not break any feature of any of the classes. In section 7.3.1, we will be describing the unit tests for each class described in section 5.1.

In addition to unit testing, we designed end-to-end tests that will be used to test the functionality of the entire system. These tests simulated most actions that a user might take, to be sure the software works consistently. The tests described a physical sequence of actions that must be performed on the Memrowave to test the software functionality. These tests were not automated, so the environment wiwas be different than the unit tests. More specifically, the test were performed in the UCF senior design lab and the test procedures will be performed by a member of the group.

## 7.4 Software Specific Testing

In this section, we will outline the various tests we will use to verify the correct functionality of the software. In section 7.4.1, we will outline the unit tests that werel applied to each class in the Memrowave's software. In section 7.4.2, we will outline the end-to-end tests that will be performed by a group member to verify that all of the software subsystems are integrated properly.

### 7.4.1 Unit Tests

Each of the critical classes in the Memrowave's software must be tested to ensure that it is working properly. In the following sections, we will describe the unit tests that will be applied to each class.

#### *The BarcodeScanner Class*

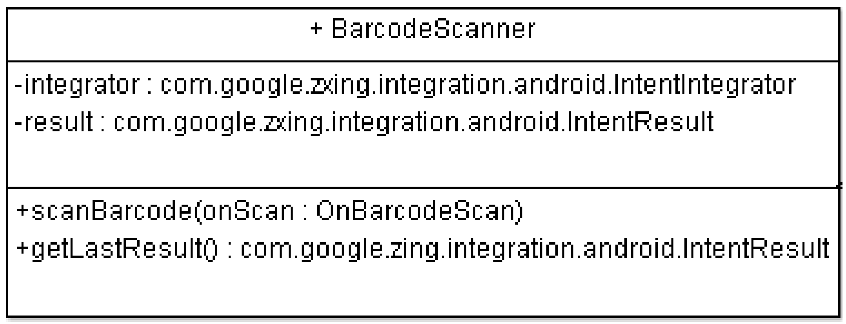


**Figure 7.4.1-1:** The BarcodeScanner class

This test will ensure that the BarcodeScanner class shown in Figure 7.4.1-1. To test the barcode scanner class, we will gather a list of barcode images and their respective UPC codes. The images will be stored in one array and the UPC codes will be stored in another. The test will go through each image in the array, use the BarcodeScanner class to read the barcode in the image, and then check whether or not the scanned UPC matches the expected value. If the class obtains the correct result for all of the barcodes, it passes the test.

#### *The ProductDatabase Class*

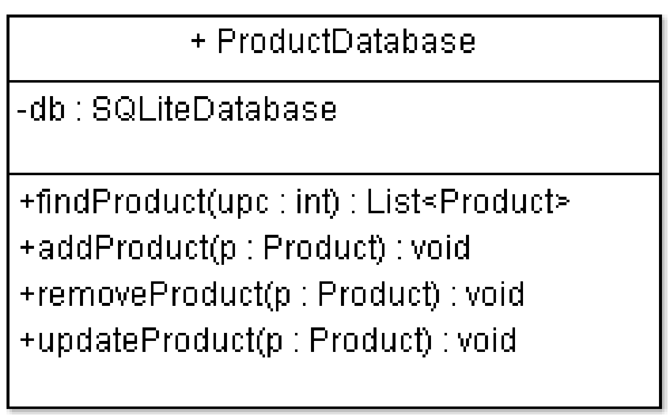


**Figure 7.4.1-2:** The ProductDatabase class

To test the ProductDatabase class shown in Figure 7.4.1-2, we will test each method of the class to ensure that they work as expected. We will supply the ProductDatabase class with a test SQLiteDatabase to do the operations on.

The first method listed in the class diagram shown in Figure 4.3.5-2 is the findProduct method. This method takes an integer UPC code as an argument and returns a list of all the products in the database that have that UPC code. To test this method, we will populate the database with a number of example products and create a corresponding instance of the Product class for each. The test will then call the findProduct method for each of these products and ensure that each Product that is returned by the method matches the expected product.

The second method listed is the addProduct method. This method takes an instance of the Product class as an argument and adds it to the database. To test this method, we will create an example product instance and attempt to add it to the database using the addProduct method. The findProduct method will then be used to retrieve the entry. If the product retrieved from the database matches the product added to the database, then the test is passed successfully.

The next method is the removeProduct method. This method takes an instance of the Product class as an argument and removes this entry from the database, if it exists. To test this method, we will first add a test product to the database using the addProduct method, then we will verify that the product was added using the findProduct method, and finally, we will call the removeProduct method on this product. If a subsequent call to the findProduct method returns no results, then the product will be considered to be removed successfully.

The updateProduct method takes an instance of the Product class as an argument and changes the entry in the database with a matching id. To test this method, we will first add a test product to the database using the addProduct method. We will then modify the original Product instance and call the updateProduct method with the modified product as the argument. We will then call the findProduct method to get the product with the matching id from the database. If the Product instance returned by the findProduct method matches the modified Product instance, then the method works as expected.

### *The UserDatabase Class*

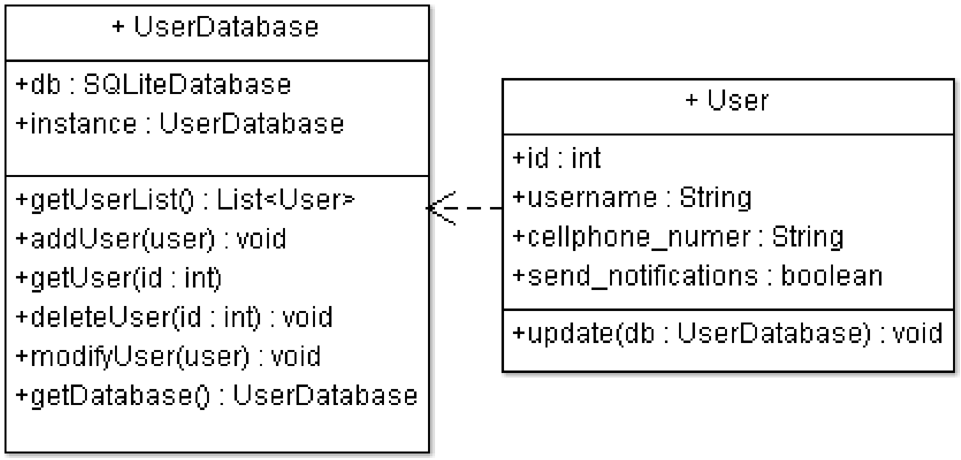


**Figure 7.4.1-3:** The UserDatabase class

The UserDatabase class is shown in Figure 7.4.1-3. To test this class, we will first create an empty SQLite database. After each test, this database will be erased so that any subsequent tests are conducted from a blank state.

The addUser method takes an instance of the User class as an argument and adds this user to the database. To test this method, we will create a number of test users and add them to the database using this method. We will then use SQL operations on the database to verify that the test users were added correctly.

The getUserList method takes no arguments and returns a list of all the users in the database. To test this method, we will first call the method on a blank database and verify that the method returns an empty list. We will then add a number of test users to the database using the addUser method. Finally, we will call the getUserList method and verify that the resulting list of users matches the original list.

The getUser method takes an integer ID as an argument and returns the corresponding user from the database. To test this method, we will add a test user to the database using the addUser method. We will then call the getUser method with the appropriate id as the argument. Finally, we will verify that the user retrieved by the getUser method matches the original test user.

The deleteUser method takes an integer ID as an argument and deletes the corresponding entry from the database. To test this method, we will add a test user to the database using the addUser method, then we will verify that the user was added correctly using the getUser method. We will then call the deleteUser method to delete the user from the database. The delete operation will be verified by calling the getUser method again and verifying that no user is returned.

### *The SMSNotifier Class*

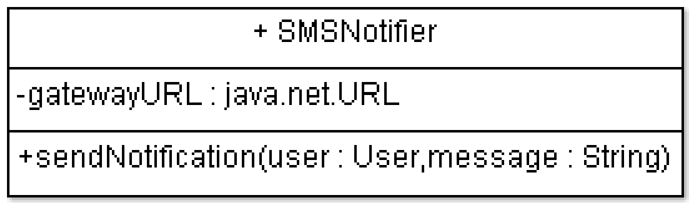


**Figure 7.4.1-4:** The SMSNotifier class

To test the SMSNotifier class shown in Figure 7.4.1-4, we will create a test user with one of the group member's cell phone number. We will then send a test message to this user and verify that the message was sent to the phone properly. We will then disable notifications in the user's settings and attempt to send the message again. In this case, we will verify that the message was not sent.

## 7.4.2 End-to-end Tests

Unit testing was helpful to ensure that all of the subsystems worked properly, but it was difficult to verify that all of the subsystems will work together properly using unit testing. To test the integration of all of the software subsystems, we designed end-to-end tests that will be performed by a group member.

## Test Name:

Main Screen Test

### Objective:

The objective of this test will be to verify that the main screen of the Memrowave's software works correctly.

### Supplies:

- Memrowave

### Preparation:

1. Ensure that the Memrowave hardware is assembled and working properly
2. Press the home button to bring up the main screen of the Memrowave

### Procedure:

1. Press the 'Settings' button and verify that the main settings screen is displayed. Press the back button and verify that the Memrowave returns to the main screen.
2. Press the 'Scan' button and verify that the barcode scanning screen is displayed. Press the back button and verify that the Memrowave returns to the main screen.
3. Press the 'Manual Cook' button and verify that the manual operation screen is displayed. Press the back button and verify that the Memrowave returns to the main screen.
4. Repeat steps 1 through 3, pressing the home button to return to the main screen instead of the back button. Verify that the Memrowave returns to the main screen for each.

## Test Name:

Manual Operation Screen Test

### Objective:

The objective of this test is to verify that the manual operation screen works correctly.

### Supplies:

1. Memrowave

### Preparation:

1. Ensure that the Memrowave hardware is assembled and working properly
2. Press the home button to bring up the main screen of the Memrowave

### Procedure:

1. Press the ‘Manual Cook’ button on the main screen.
2. On the manual cook screen, press the ‘Quick Cook’ button. Verify that the quick cook screen is displayed, and then press the back button. Verify that the manual cook screen is displayed again.
3. Enter a cook time on the number pad. Verify that the entered time is shown on the timer display.
4. Press the ‘Power’ button and verify that the power level menu is shown. Enter a power level and verify that the software returns to the manual cook screen with the appropriate power level shown.
5. Press the clear button and verify that the entered cook time is reset to zero.
6. Enter another cook time and press the start button. Verify that the Memrowave begins to cook and counts down the time properly. When the timer reaches zero, verify that the Memrowave stops cooking and produces the appropriate notifications.

## Test Name:

Product Scan Screen Test

### Objective:

This test is designed to verify that the product scanning screen works correctly.

### Supplies:

1. Memrowave

### Preparation:

1. Ensure that the Memrowave hardware is assembled and working properly.
2. Insert a new test product into the local product database.
3. Press the home button to bring up the main screen of the Memrowave.

### Procedure:

1. Press the ‘Scan’ button on the main screen. Verify that the product scan screen is displayed.
2. Verify that the camera preview is shown on the screen.
3. Hold the barcode for the test product up to the camera so that the entire barcode is shown on the preview. Press the ‘Scan Now’ button and verify that the barcode is scanned properly and that the product details screen is shown.
4. Press the back button and verify that the product scan screen is displayed again.

## Test Name:

Product Cook Test

### Objective:

This test is designed to verify that the Memrowave can scan and cook a product automatically.

### Supplies:

1. Memrowave

### Preparation:

1. Ensure that the Memrowave hardware is assembled and working properly.
2. Insert a new test product into the local product database.
3. Press the home button to bring up the main screen of the Memrowave.

### Procedure:

1. Press the ‘Scan’ button on the main screen. Verify that the product scan screen is displayed.
2. Verify that the camera preview is shown on the screen.
3. Hold the barcode for the test product up to the camera so that the entire barcode is shown on the preview. Press the ‘Scan Now’ button and verify that the barcode is scanned properly and that the product details screen is shown.
4. Verify that the correct cook time and power level are shown for the test product.
5. Put the product into the Memrowave and press the ‘Start’ button. Verify that the Memrowave begins to cook the food. When the timer reaches zero, verify that the Memrowave stops cooking and sends the appropriate notifications.

## Test Name:

Quick Cook Test

### Objective:

This test is designed to verify that the Quick Cook feature works correctly.

### Supplies:

1. Memrowave

### Preparation:

1. Ensure that the Memrowave hardware is assembled and working properly.
2. Verify that the Quick Cook database has been populated with example products.
3. Press the home button to bring up the main screen of the Memrowave.

### Procedure:

1. On the main screen, press the ‘Manual Cook’ button, then press the ‘Quick Cook’ button. Verify that the ‘Quick Cook’ screen is displayed.
2. Verify that the Quick Cook products inserted in the Preparation section are displayed in the product list.
3. Select one of the quick cook products and verify that the product cook screen is displayed with the correct information.
4. Press the ‘Back’ button to return to the Quick Cook screen.
5. Press the ‘Add New’ button. Verify that the ‘New Quick Cook Product’ screen is displayed. Enter the name, cook time, and power level, and press the save button.
6. Press the ‘Home’ button to return to the main screen, and then navigate to the ‘Quick Cook’ screen. Verify that the new Quick Cook item is shown in the list.

# 8 Standards

The products designed is essentially a microwave oven therefore a list of standards must be adhered to. The projects sub-systems can be broken down and listed. Each subsystem contains a variety of standards associated with it. The standards are presented in figure 8-1.

| SUBSYSTEM | STANDARD(S) |
|---|---|
| **MICROWAVE OVEN** | FDA: Sec. 1030.10 Microwave ovens<br><br>Sec. 1010.1 Scope.<br><br>Sec. 1010.2 Certification.<br><br>Sec. 1010.3 Identification.<br>Sec. 1010.4 Variances<br><br>Sec. 1010.5 Exemptions for products intended for United States Government use.<br><br>Sec. 1010.13 Special test procedures.<br><br>Sec. 1010.20 Electronic products intended for export.<br><br>Sec. 1004.1 Manufacturer's obligation to repair, replace, or refund cost of electronic products.<br>Sec. 1004.2 Plans for the repair of electronic products.<br><br>FCC: Applications filed to obtain Special Temporary Authority (STA) in the services described in 47 C.F.R. §1.913(d)<br><br>47 C.F.R. 1.1307(b), 1.1310, 2.1091, 2.1093] |
| **ANDROID** | Standard design UX-B1<br><br>Navigation UX-N1, UX-N2, UX-N3 |

| | |
|---|---|
| | Notifications UX-S1, UX-S2<br><br>General WR-GL, WR-VF, WR-BF<br><br>Packaging WR-PK |
| **POWER** | 1B.3− Planning Standards for Transmission Voltage<br><br>1B.4—Reliability Criteria for System Planning<br><br>1B.7—Lightning and Other Overvoltage Protection<br><br>1C.1—Power Quality Introduction<br><br>1C.2.1—Voltage Level and Range<br><br>1C.3.1—Voltage Balance<br><br>1C.4.1—Harmonic Distortion<br><br>1C.5.1—Voltage Fluctuation and Flicker<br><br>1C.6.1—Voltage Disturbances<br><br>1C.7.1—Stray Voltage<br><br>1C.8.1—Voltage Frequency |
| **ELECTRICAL COMPONENTS** | IEC 60028 Ed. 2.0 b:1925<br><br>International standard of resistance for copper<br><br>Revises IEC 60034-11 Ed. 1.0 b:197 Rotating electrical machines - Part 11: Thermal protection |

**Figure 8-1:** Standards

# 9 Administrative Content

In this section we will discuss the administrative content of the project, including project milestones and budget considerations.

## 9.1 Milestone Discussion

The milestone discussion details how the Memrowave project moved from the initial conceptual stage to the final product stage. A rough outline of what was needed to be done to reach the final end product is shown below.

I. Conceptual Stage
II. Design Stage
III. Part and Component Collection
IV. Initial Assembly and Testing
V. Redesign
VI. Final Assembly and Testing

Within each major section, divisions between software and hardware are outlined. The completion of the first areas was finished with the initial writing of this document. Decisions on specific parts and components have been decided and detailed in their corresponding sections while others have been specified without exact part information. For the part and component collection, all parts were collected based on the importance and how extensive any further development on the item is necessary such as the processing unit and LCD touchscreen which will need continuous development throughout the project to make updates, changes, and to find and fix any bugs. On the hardware side, crucial components such as the magnetron, power supply, and voltage regulators were needed as soon as possible to make sure such parts will fit into to the overall assembly and scheme of the Memrowave. Initial assembly and testing was the most extensive and time consuming sections of the overall project. After some development by both hardware and software sections of the Memrowave, we started the integration of both divisions. Not particularly into an enclosure, but a checking of the overall system's abilities to doing certain functions and the robustness. A list of the specific checks were completed:

- Software functionality
- Power supply working properly
- Electrical component check
- LCD touchscreen functionality
- Processing unit operation
- Switch tests

- Camera operation
- WIFI operation
- Notification operation
- Total power consumption

Further tests were conducted after assembling into the microwave enclosure takes place. After the initial round of testing any changes, errors and delays were addressed. Final design conclusions were drawn in the following categories and will dictate how to progress:

- Successes
- Failures
- Additions
- Withdrawals

Once final design decisions were set in stone, a final assembly and development period for the final product began. At this point both hardware and software have been fully developed and integration issues and fine tuning will be the only issues left to address. Below is an outline of the major milestones.

| Hardware | | | |
|---|---|---|---|
| | **Part Collection** | **Part Testing and Characterization** | **Initial System Development** |
| | PCB Design | Functionality | PCB fabrication and assembly |
| | DC Power Supply | Usablility | Proper operation of powered devices |
| | Voltage Regulators | | Power specifications are within design |
| | Internal Lights | | |
| | Internal Beeper | | |
| | Turnable Motor | | |
| | Magnetron | | |
| | | | |
| **Software** | | | |
| | **Part Collection** | **Part Testing and Characterization** | **Initial System Development** |
| | Processing Unit | Functionality | Integration of Processing Unit and LCD Touch Screen |
| | LCD Touch Screen | Usablility | Integration of Processing Unit and WIFI Module |
| | WIFI Module | | Integration of Processing Unit and Camera Module |
| | Camera Module | | |
| | | | |
| **Hardware /Software** | | | |
| | **Integration Tasks** | **Redesign Evaluation** | **Final Integration** |
| | Proper operation of software part collection under power supply | Successes | Proper operation of any alternative functions or processes |
| | Desired control of internal lights by processing unit | Failures | Proper operation of software part collection under power supply |
| | Desired control of internal beeper by processing | Additions | Desired control of internal lights by processing unit |

| | | | |
|---|---|---|---|
| | unit | | |
| | Desired control of magnetron by processing unit | Withdrawals | Desired control of internal beeper by processing unit |
| | Desired control of turntable motor by processing unit | | Desired control of magnetron by processing unit |
| | Available space to fit components into microwave enclosure | | Desired control of turntable motor by processing unit |
| | | | Available space to fit components into microwave enclosure |
| | | | Cosmetic fine tuning |
| | | | Proper function of the user interface |
| | | | Overall functionality of the Memrowave |

## 9.2 Finance and Budget Discussion

The Memrowave development is only possible with a strong financial backing. The Memrowave design team decided on financing the project fully out of pocket. The reasoning for this is due to the fact that the Memrowave overall cost was deem to be low enough for self-purchase, the design didn't fit the needs of Boeing, and our team did not want to risk signing over parts endorsed by sponsors. The financing was then be handled completely internally by group members; Joseph Serritella, Darren Armstrong, Jack Gulick, and Winston Todd. Finance was split evenly by a divisor of 4. The Benefits of out of pocket financing are; Parts can be ship to a destination of our choosing, Project will be able to remain in our control afterwards, and dealing with sponsor will not be an issue. Because our project is personally financed parts are not required to be shipped to a specific location. The benefit of this is found in personal leisure time. The Memrowave design team was able to acquire parts at any time and have them shipped to a location of our choosing. For example our project will not be limited to being stored with other group design parts. The next big influence on personal financing was we were allowed without question to have our prototype remain in our control. Lastly because we are personally financing the Memrowave we will not have to adhere to a sponsor's requests. The Memrowave's budget will consisted of, LCD screens, Wi-Fi, printed circuit boards, and more along these lines. Our design which was heavily influenced by the microwave we acquired. Our design allowed us to reuse the original parts such as motors and fans, our budget will run lower. The budget calculated in this section will cover replacing motors, the magnetron and others. This will give our budget a worst case scenario.

The budget included uses the medium price listing for cameras, Wi-Fi Chips, development boards, Touch screen panels, Microcontrollers, resistors, transistors, and so on. To save time in broken parts and PCB we will also be ordering from vendors that will supply more than one board upon purchase. The Best PCB manufacturer for this will be OCH Park because they supply 3 copies of the printed circuit board with the purchase.

The Following Table 9.2-1 provides information on our budget.

| | Projected | Actual |
|---|---|---|
| Development Board | $70 | $55 |
| Microcontrollers (x4) | $40 | Free |
| Logitech Webcam C270 | $40 | Free |
| Microwave | $200 | Free |
| Microwave Turntable replacement | $25 | Free |
| Turntable motor replacement | $20 | Free |
| Microwave fan replacement | $30 | Free |
| Fan motor replacement | $30 | Free |
| Transformer DC/AC | $10 | 18.99 |
| Voltage Regulators | $20 | 12.66 |
| Ringer | $5 | Free |
| Lights | $5 | Free |
| Wi-Fi Card | $40 | $13 |
| PCB fab | $60 | $36.55 |
| Magnetron replacement | $60 | Free |
| Power cord | $5 | Free |
| LCD screen | $100 | $66 |
| Discrete Components | $25 | $60 |
| **TOTAL:** | $760 | 268.53 |

**Table 9.2-1:** Budget

Table 9.2-1 provides the layout for the possible budget we will encounter during our design. This budget was calculated to be seven hundred and sixty dollars. The Memrowave was under budget by $529.45. This was accomplished since all the microwave components were salvaged from the obtained microwave.

# Appendix A – Copyright Permissions

“Apache License, Version 2.0.” Internet: http://www.apache.org/licenses/LICENSE-2.0 [12/1/2014]

“GNU GENERAL PUBLIC LICENSE.” Internet: http://www.gnu.org/licenses/gpl-3.0.html [12/1/2014].

‘Texas Instruments Terms of Use.” Internet: http://www.ti.com/corp/docs/legal/termsofuse.shtml [12/1/2014].

Images from Sparkfun Electronics are licensed under the CC BY-NC-SA 3.0 license. Internet: http://creativecommons.org/licenses/by-nc-sa/3.0/ [12/1/2014].

# Appendix B – Datasheets

“MAX6958 8-Digit LED Display Driver.” Internet: https://www.sparkfun.com/datasheets/Components/General/COM-09622-MAX7219-MAX7221.pdf [12/1/2014].

“7-Segment Display - 20mm (Blue)” Internet: http://dlnmh9ip6v2uc.cloudfront.net/datasheets/Components/LED/1LEDBLUCC.pdf [12/1/2014].

“MSP430G2553 Datasheet.” Internet: http://www.ti.com/lit/ds/symlink/msp430g2553.pdf [12/1/2014].

“BeagleBone Black Reference Manual RevC.1.” Internet: https://github.com/CircuitCo/BeagleBone-Black/blob/master/BBB_SRM.pdf?raw=true [12/1/2014].

“TI AM335X Sitara Processor.” Internet: http://www.ti.com/lit/ds/symlink/am3359.pdf [12/1/2014].

# Appendix C – References

“Android API Package Index.” Internet: http://developer.android.com/reference/packages.html [11/28/2014].

“Android API Guides.” Internet: http://developer.android.com/guide/index.html [11/28/2014].

“Play SMS: Free and Open Source SMS Software.” Internet: http://playsms.org [12/2/2014].

“ZXing Barcode Software Library.” Internet: https://github.com/zxing/zxing/ [12/2/2014].

"OLED." Internet: http://en.wikipedia.org/wiki/OLED [12/1/2014].

"Liquid Crystal Display." Internet: http://en.wikipedia.org/wiki/Liquid-crystal_display [12/1/2014].

"Touchscreen." Internet: http://en.wikipedia.org/wiki/Touchscreen [12/1/2014].

"Wi-Fi." Internet: http://en.wikipedia.org/wiki/Wi-Fi [12/1/2014].

"Raspberry Pi Frequent Asked Questions." Internet: http://www.raspberrypi.org/help/faqs/ [12/1/2014].

"Raspberry Pi Wiki." Internet: http://en.wikipedia.org/wiki/Raspberry_Pi [12/1/2014].

"Beagleboard:BeagleBoneBlack Wiki." Internet: http://elinux.org/Beagleboard:BeagleBoneBlack [12/1/2014].

"BeagleBoard." Internet: http://en.wikipedia.org/wiki/BeagleBoard [12/1/2014].

"PandaBoard." Internet: http://www.omappedia.com/wiki/PandaBoard [12/1/2014].

"PandaBoard." Internet: http://en.wikipedia.org/wiki/PandaBoard [12/1/2014].

"PWM type." Internet: http://sinhardware.com/index.php/vrm-articles/113-digital-vs-analog-pwms/74-analog-vs-digital-pwms [12/1/2014].

“Timers.” Internet: http://homepages.ius.edu/RWISMAN/C335/HTML/msp430Timer.HTM [12/1/2014].

“Coding PWM.” Internet: http://www.egr.msu.edu/classes/ece480/capstone/fall10/group04/Dong,%20Roy%20-%20Application%20Note.pdf [12/1/2014].

“Stepper Motors.” Internet: http://www.pcbheaven.com/wikipages/How_Stepper_Motors_Work/ [12/1/2014].

“Brushless motors” Internet: http://electronics.howstuffworks.com/brushless-motor.htm [12/1/2014].

“Brush motors” Internet: http://electronics.howstuffworks.com/brush-motor.htm [12/1/2014].

“Switching circuits” Internet: http://www.electronics-tutorials.ws/transistor/tran_7.html [12/1/2014].

“Brushless motors” Internet: http://electronics.howstuffworks.com/brushless-motor.htm [12/1/2014].