POTD-Problem Based Alarm Clock

By

Sherry Wu

Shirley Xu

Charlene Zheng

Final Report for ECE 445, Senior Design, Fall 2019

TA: Jack Li

11 December 2019

Team 16

Abstract

This project is a problem-based alarm clock that aims to help people wake up in the morning. In addition to the traditional beeping sound, the alarm clock also displays 5 multiple choice questions on the screen that need to be answered correctly by the users in order to be turned off. The alarm clock can be connected with a mobile app via Bluetooth so that users can load their own questions and setup time to wake up. The questions can be of any subject and are typed manually. The flexibility of questions gives users opportunities to review certain things as they like. The app also records the time it takes for them to wake up each morning and the result of their quiz.

Contents

1. Introduction

1.1 Objective

Getting up on time has always been a problem for many people. The very basic alarm that only makes clangorous sound often fails to wake people up, and often lead to people turning it off half-awake and falling back to sleep, which may cause a delay to their days. Setting alarm clocks with cell phones also has certain drawbacks. Often, the cell phone battery dies in the middle of the night, and thus the alarm fails.

To address this issue, we propose a problem-based alarm clock that requires users to answer several multiple choice questions to turn it off. The alarm clock is connected with a mobile app via Bluetooth so that users can load their own questions and set alarm time. The questions can be of any subject and are typed manually. The flexibility of questions gives users opportunities to review certain things as they like. The app also displays the time it takes for them to wake up each morning and the result of their quiz.

1.2 Background

The process of thinking can stimulate people's brains and help people fully wake up, not only physically, but also mentally, according to Peter Balyta, President of Education Technology at Texas Instruments [1].

While waking up in the morning can be hard for a lot of people, there are many applications that attend to address this issue with problem-based approach in the market, such as the Mathe Alarm. It has both Android and iOS app that is available for free. Using this app, users are able to choose the number of simple algebraic questions they want to answer in order to turn off the alarm. Users can also pick the difficulty level of the math problems. However, it does not support custom questions and only have algebraic ones.

Moreover, in the current market, there is no physical alarm that has the similar functionalities. While some people use phone apps as alarm, many people still prefer to use physical alarms. According to a survey from market research firm YouGov in 2011 [2], while 48% of respondents aged 16-34 said they used their phones as alarms, another 38 percent said they use either clock radios or alarm clocks.

1.3 Visual Aid

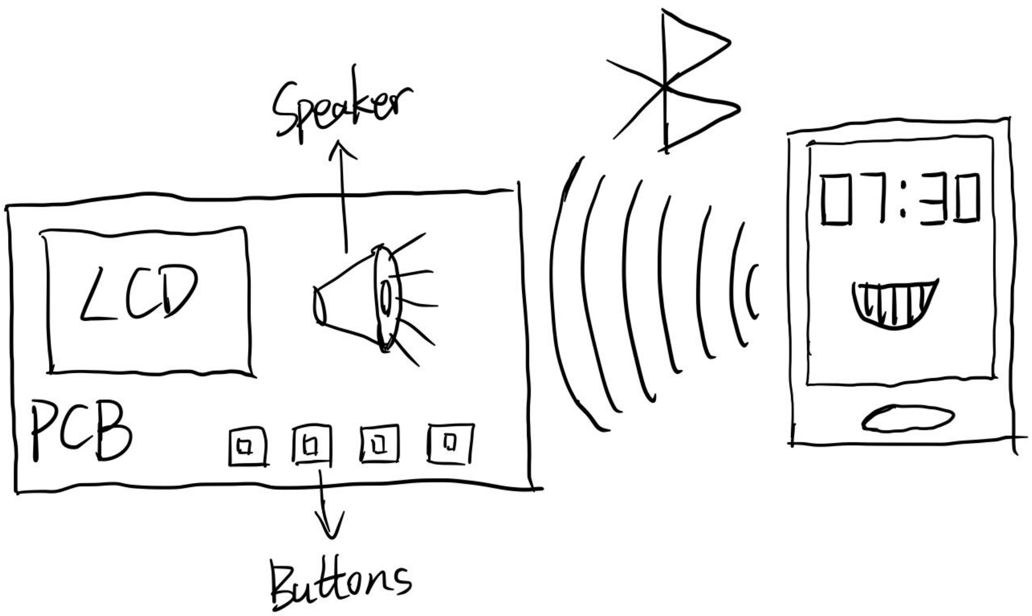

Figure 1. Visual Aid

1.4 High-level Requirements

- The alarm device must be able to function as an actual alarm clock: display time, beep at designated time and turn off the alarm when all questions are correctly answered by the user.
- The alarm device must be able to receive and store questions from the Android application. At alarm time, it should display questions, interact with users and record their input data.
- The alarm device must be able to communicate with the Android app via Bluetooth. The Android application should allow the users to set alarm time and input questions; the alarm device should send recorded data back to the app per the request of the users.

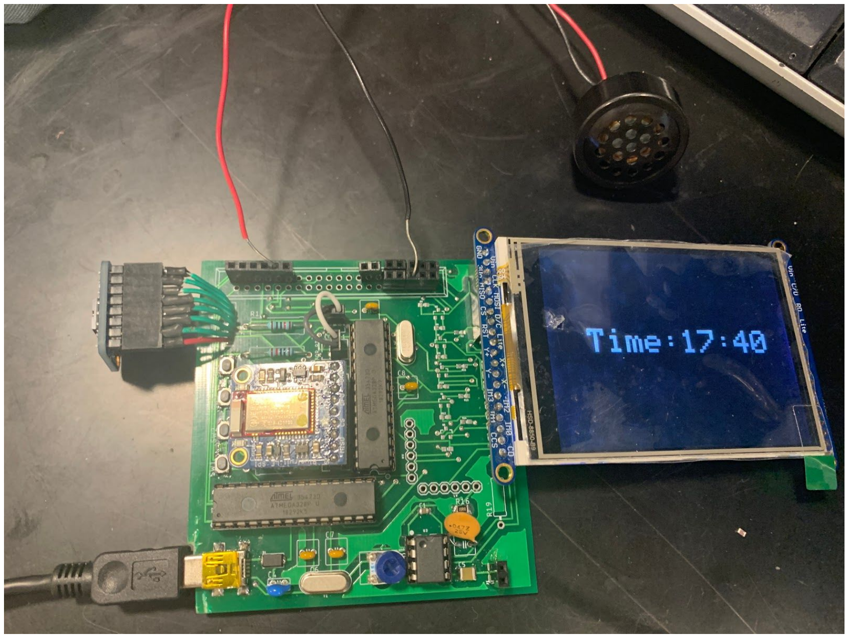

Figure 2. Actual Product

2. Design

The project requires four sections of hardware and one section of software. The hardware contains power supply, control unit, user interface and an alarm. The software is an Android mobile application.

The main power supply is a 5 V-1000 mAh portable power bank, which consistently provides power for all components. The control unit consists of a microcontroller and a Bluetooth module. The user interface includes a microcontroller to control LCD, an LCD module, and physical buttons for users to press. The alarm consists of a speaker and a real time clock module. The software is an Android app that is used to set the alarm time and questions.

Normally, the control unit constantly receives time data from RTC to keep track of the real time in order to check if the current time is the same as the alarm time. Thus, the control unit knows when the speaker should beep. The RTC is connected to LCD microcontroller as well, and the LCD microcontroller can receive time data and directly display it to the screen. When connected to the Bluetooth, the control unit receives question data and alarm time from software and send recorded user data to software. Then the control unit sends question data to the LCD microcontroller, so LCD microcontroller can display questions directly from its memory at the designated time.

At alarm time, the control unit triggers the alarm to go off, signals the user interface to display questions on the LCD module and waits for user input. It records the time it takes the user to answer all the questions correctly and the number of questions they got wrong for the first round. These data are sent to the Android application when the user requests them via the app. If any of the questions are incorrectly answered, the user needs to answer all questions again. After all questions are answered correctly, the control unit turns off the speaker. The user interface displays time again.

The whole system flow is shown in Figure 3.

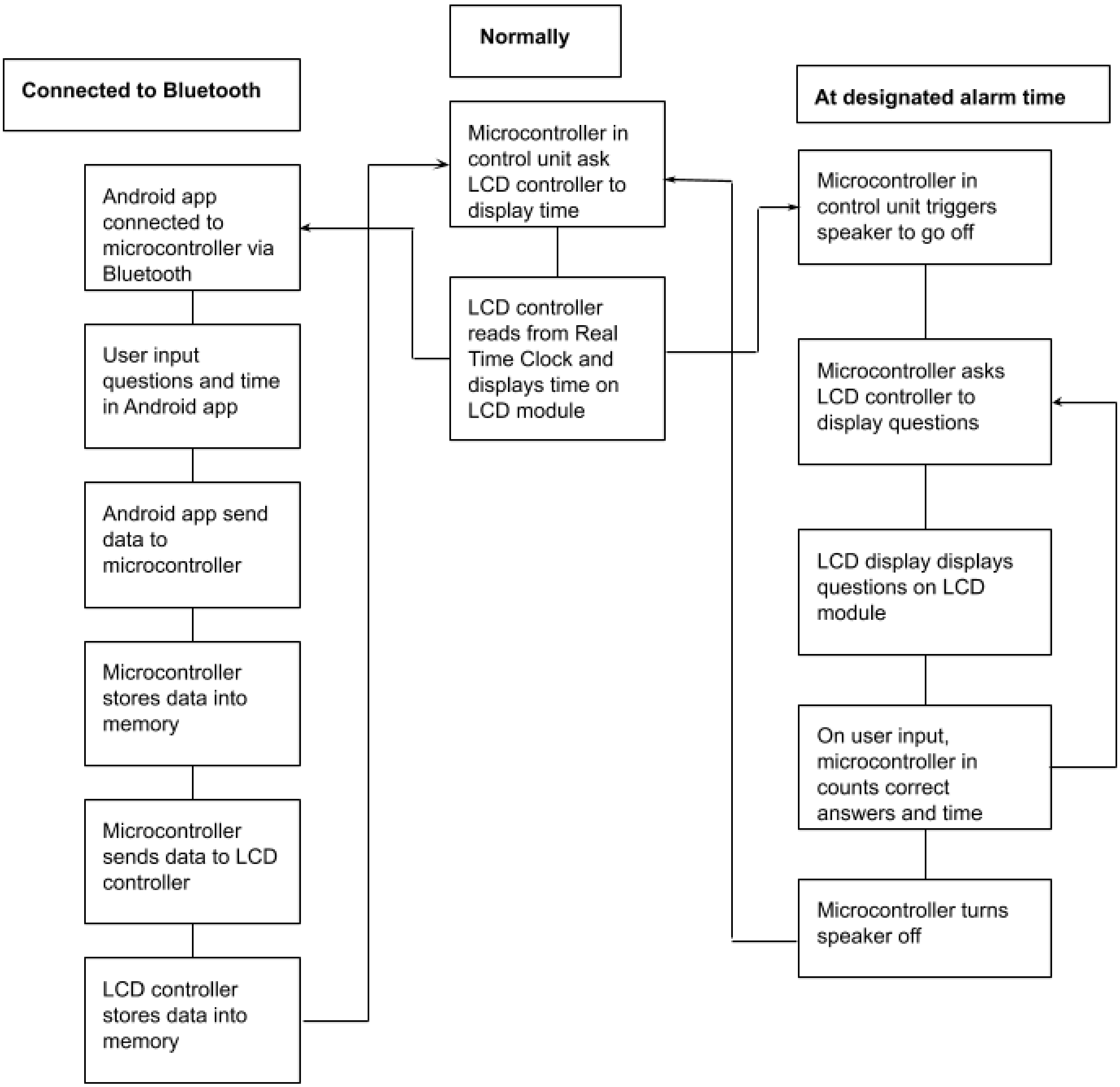

Figure 3. System Flow

2.1 Block Diagram

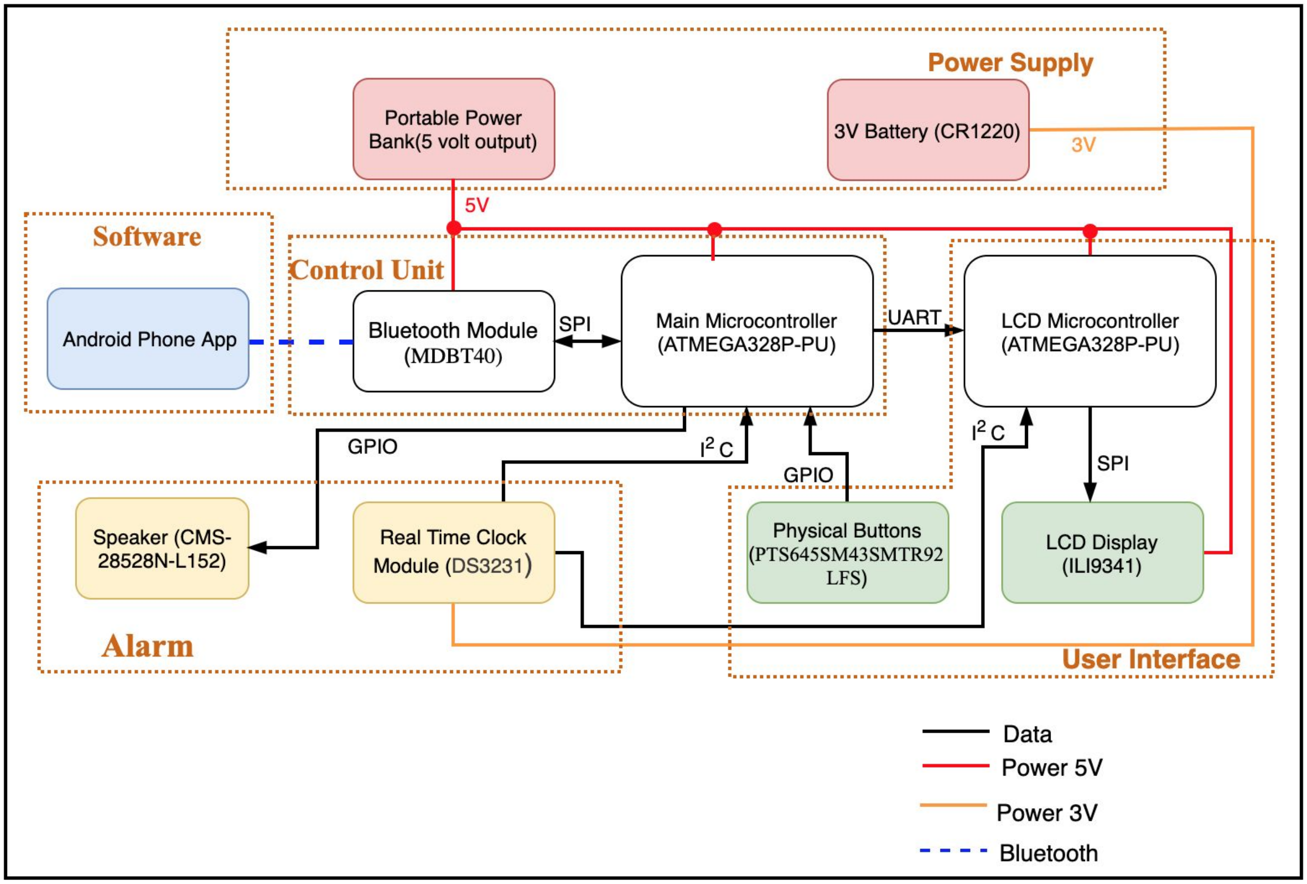

Figure 4. Block Diagram

2.2 Control Unit

The control unit is able to receive data from software (Android Phone App) as input, which provides information of questions and alarm time. The control unit also receives data from physical buttons so as to judge the correctness of user input and record the number of incorrectly answered questions. These data will be sent back to the app through Bluetooth module for user reference. Moreover, control unit communicates with the microcontroller in the user interface to send the questions that needs to be displayed for the day. The control unit is responsible for turning on and off the alarm's ringtone as well.

2.2.1 Main Microcontroller

The microcontroller (Atmega328P) handles input data received from Bluetooth module (MDBT40) via SPI , DS3231 real time clock via I^2C , and physical buttons via GPIO. The microcontroller loads questions to the microcontroller in user interface via UART after the control unit receives question data from software. According to the data received from RTC module, microcontroller determines when to turn on the alarm's speaker; based on the user input, it determines when to turn off the alarm. The number of incorrectly answered questions is stored in the microcontroller and is sent to Android phone app via Bluetooth module. Microcontroller's 32 KB flash memory will be mainly used to store the questions.

Memory Requirement: *Maximum number of questions stored: 30*
Question length constraint: 200 char
Answer length constraints: 25 char
*Integer for time: 10 digit (4 bytes * 10)*
Memory estimated:
*30 * (200 + 4 * 25) byte + (10 * 4) byte +*
some space to store user answer data + several ring-tone music < 32 KB

2.2.2 Bluetooth Module

Bluetooth module Adafruit Bluefruit LE SPI Friend (MDBT40) is used to communicate between Android phone app and the microcontroller in the control unit. The users can load their questions into the microcontroller via Bluetooth module, as well as set the alarm time. The Bluetooth module is also used to send user answer data from microcontroller to the Android phone app when the user's phone is connected to the alarm via Bluetooth.

2.3 Power Supply

1. The power supply for the whole device is mainly a 5 V-1000 mAh portable power bank. It is connected to the device via USB.
2. The only separate battery power (button battery 3 V CR1220) used is for real time clock.

2.4 User Interface

The user interface is used to receive time and question data from control unit, and outputs user answer data to the control unit. The user interface includes a microcontroller (ATMEGA328P), used solely to control the LCD module via SPI, an LCD (ILI9341) module, used to display questions, and four physical buttons for users to press to answer multiple choice questions.

2.4.1 LCD Microcontroller

The microcontroller (ATMEGA328P) (referred later as the LCD controller) is solely used to control the LCD module via SPI. Normally, it constantly receives data from the RTC to display real time on the LCD module. When the control unit is connected to the Bluetooth, the LCD controller receives questions from the control unit's microcontroller and stores in its memory. When the alarm goes off, the microcontroller from the control unit tells the LCD controller which problem to display.

2.4.2 Physical Buttons

Users use four buttons (PTS810 SJM 250 SMTR LFS) to answer the multiple choice questions. They would send data to the microcontroller in the control unit via GPIO.

2.4.3 LCD Module

The LCD (ILI9341) module is used to display time and questions. It is directly controlled by a separate microcontroller (referred later as the LCD controller) via SPI. The microcontroller in the control unit tells the LCD controller what to display and the LCD controller controls the LCD module to display them accordingly.

2.5 Alarm

2.5.1 Speaker

The purpose of CMS-28528N-L152 speaker is to produce audio output to wake people up. It is controlled by the main microcontroller via GPIO.

2.5.2 Real Time Clock Module

The DS3231 Real Time Clock module is used to keep track of time and send its time data to the microcontroller in the control and the LCD controller unit via I^2C , The RTC is also connected to a pull-up resistor in order to avoid messy data when the data bus is not used. DS3231 has its own battery-backup input for continuous timekeeping.

2.6 Software

2.6.1 Android Application

An auxiliary Android app communicates with the control unit via Bluetooth for users to type in their customized questions as well as set the alarm time. When it is connected to the alarm, users can add questions and set time. The software also displays the time it took the users to get up as well as the number of correctly answered questions.

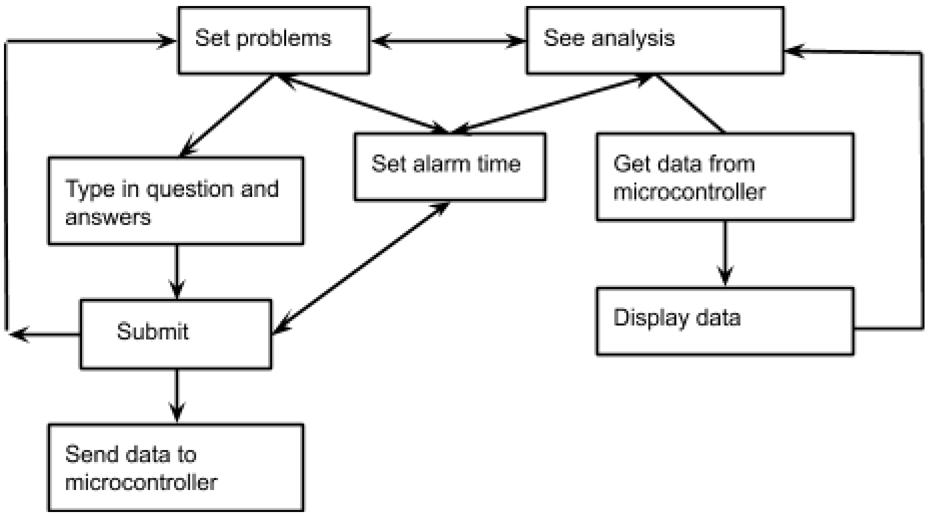

Figure 5. Software Flow

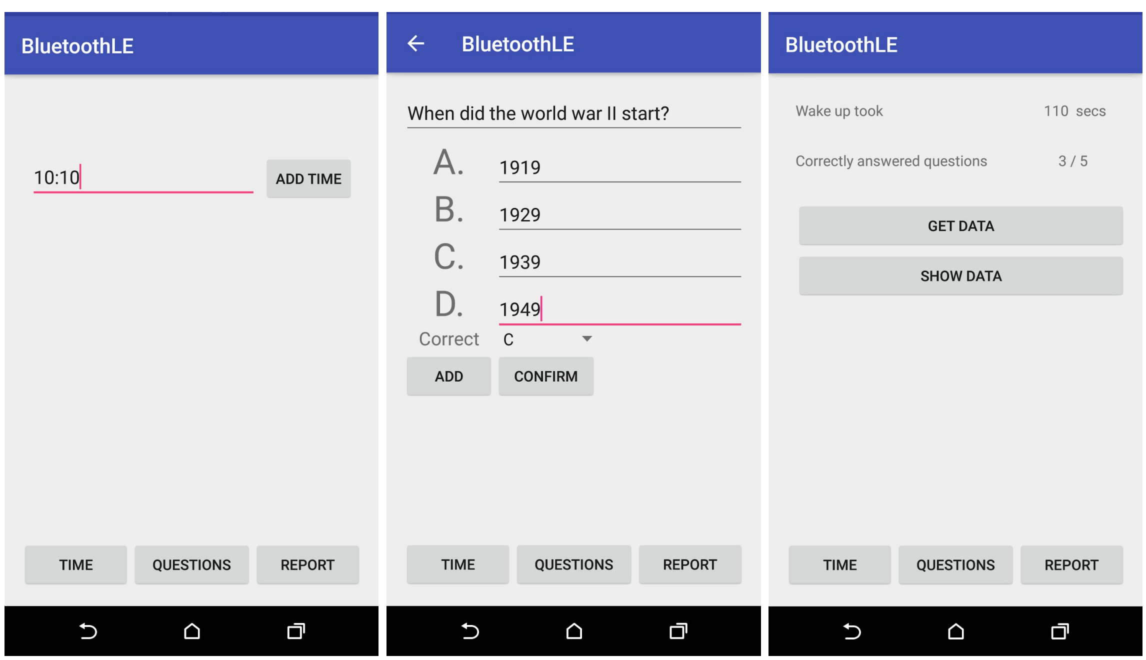

Figure 6. Android Application User Interface

2.7 Schematic

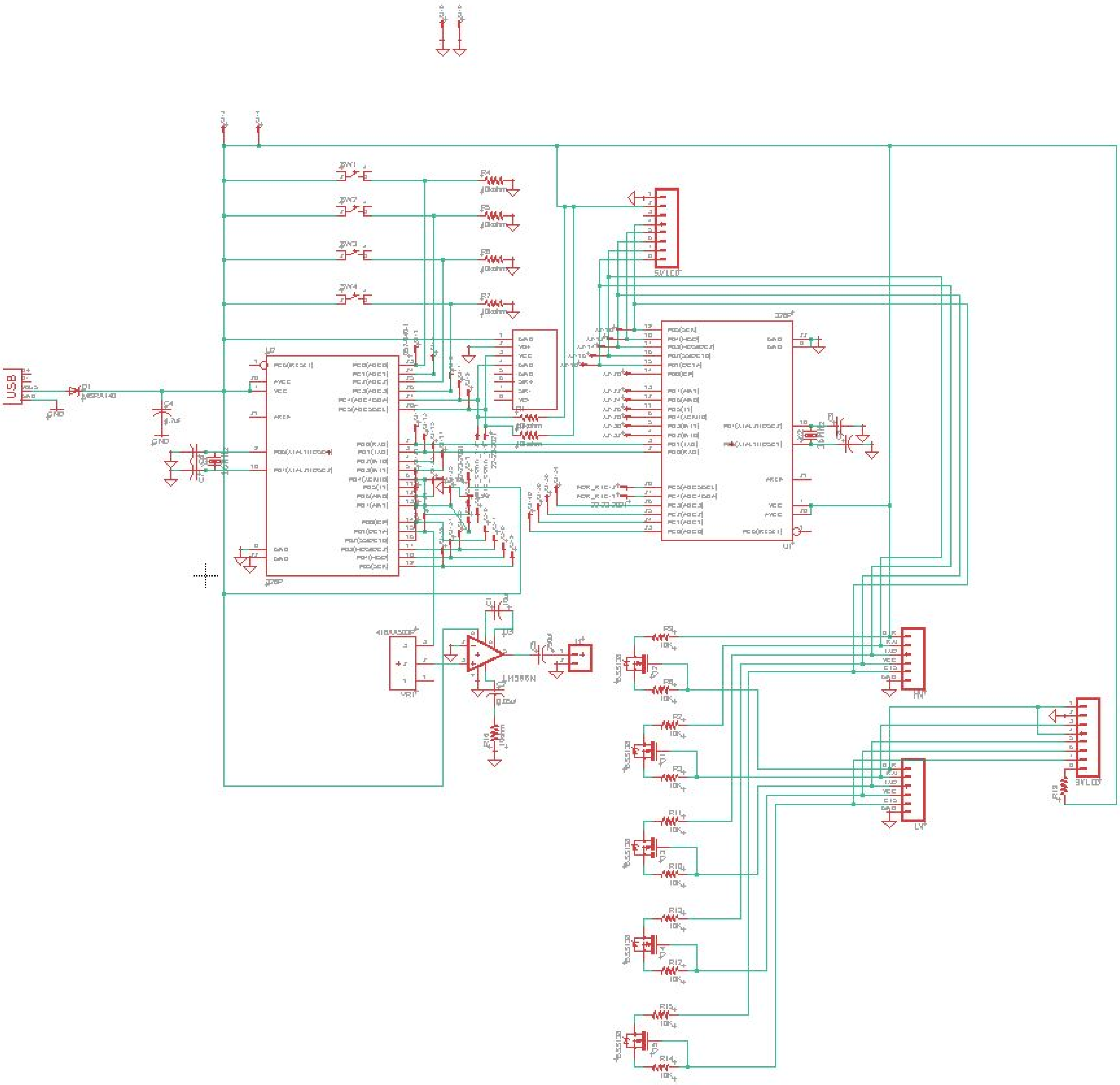

Figure 7. Schematic

3. Design Verification

3.1 Control Unit Verification

3.1.1 Main Microcontroller

To test the Main Microcontroller works:

1. Press the buttons to control the LED on the microcontroller.
2. Write a test program to have the microcontroller trigger the speaker to beep.
3. Write a test program to have the microcontroller turn on LED based on RTC input.
4. a. Write a test program to have the microcontroller turn on LED based on Bluetooth module input.
   b. Write a test program to have the Bluetooth module turn on LED based on microcontroller input.
5. Write a test program to have the microcontroller send data to LCD microcontroller (Figure 8 and Figure 9).

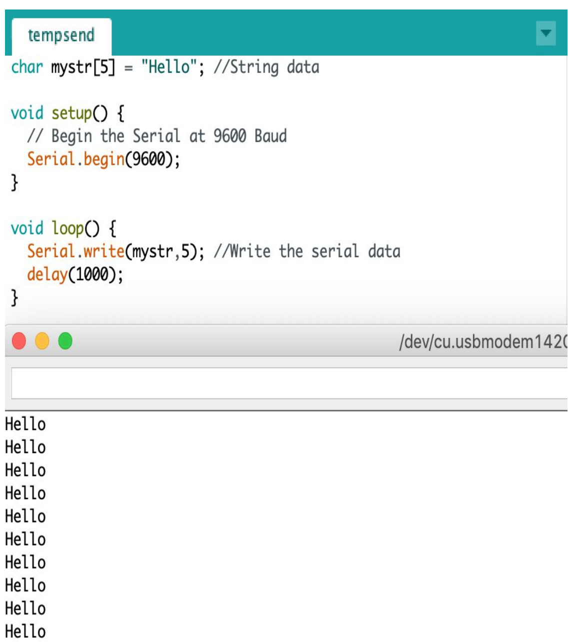

Figure 8. Main Microcontroller and Monitor **Figure 9. LCD Microcontroller Code Monitor**

During the test, there were some challenges (Figure 10):

1. When it is the alarm time, the Main Microcontroller needs to make the speaker beep, as well as read data from buttons. Because the microcontroller can only do one thing at a

time, there is a loop that the Main Microcontroller needs to trigger the speaker first and then read data from buttons and repeat the process. The loop goes so fast that when the button is pressed, the Main Microcontroller is making the speaker beep and will miss data from buttons. To solve the problem, the frequency of triggering the speaker is decreased every 1,000 loop. But the buttons still need to be pressed for a second to make sure that the data is received by the Main Microcontroller.

2. There was data lost during SPI transmission between Bluetooth and Main Microcontrooler, and UART transmission between Main Microcontroller and LCD Microcontroller. When sending a very long string, there were some characters missing in the middle of the string. To solve the problem, firstly the frequency of both microcontrollers are decreased to 1200 baud/sec. Secondly, the frequency of both microcontrollers reading data from Real Time Clock is decreased to every 10,000 loop.

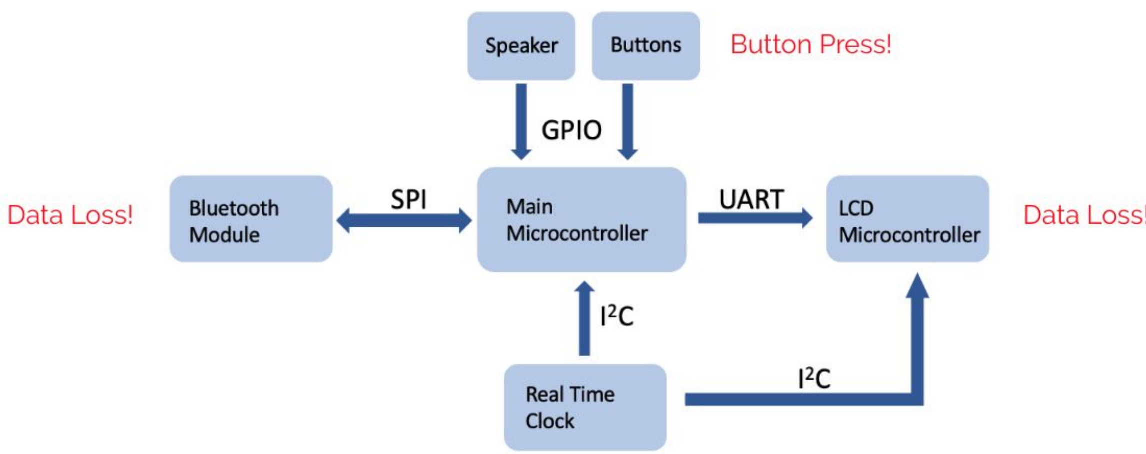

Figure 10. Challenges When Integrating The System

It was verified that the Main Microcontroller was correctly connected and works as expected.

3.1.2 Bluetooth Module

To test the Bluetooth Module works:

1. Write an Android application to control the LED to light up on the microcontroller.
2. Write a test program to have the microcontroller turn on LED based on Bluetooth module input.
3. Use the Android app to connect to the Bluetooth module, input and send time and questions strings to the microcontroller and print them on the console.
4. Use the microcontroller to send strings to the Android app and have the app print them on the screen.

It was verified that the Bluetooth Module was correctly connected and was able to send data between the Android app to the microcontroller.

3.2 Power Supply Verification

To test the portable power bank and button battery works:

1. Measure the output voltage of portable power bank using an oscilloscope, make sure that the output voltage stays within 2% of 5 V.
2. Measure the output voltage of button battery using a Voltmeter, make sure that the output voltage stays within 1% of 3 V.

It was verified that the portable power bank does provide the right voltage (4.97 V), and the button battery does provide a right voltage (2.98 V).

Before the test, we find out that our original design of using two 9 V Alkaline batteries is not able to support our alarm for a long time:

Two 9 V battery (Duracell) = 2 * 580 mAh = 1160 mAh

Backlight Current for LCD ≃ 80 mAh (according to datasheet)

Duration = 1160 mAh / 80 mAh ≃ 14.5 hours

So we changed our design to use a 10000 mAh 5 V portable power bank, which the power bank, the duration of the alarm has improved:

Duration = 10000 mAh / 80 mAh ≃ 125 hours ≃ 5.3 days

3.3 User Interface Verification

3.3.1 LCD Microcontroller

To test the LCD Microcontroller works:

1. Write a test program for the LCD microcontroller to display some text on the LCD display.
2. Write a program to have the microcontroller in the control unit control the LED on the LCD microcontroller.

Figure 11. LCD Displays Text Received From LCD Controller

It was verified that the LCD Microcontroller was correctly connected and works as expected.

3.3.2 Physical Buttons

To test the buttons work:

1. Write a test program to have Main Microcontroller read data from buttons. Make sure that the data correctly represents whether the button is pressed.

It was verified that the Main Microcontroller can read data from buttons and determine whether the button is pressed.

3.4 LCD Verification

To test the LCD work:

1. Write a test program to have LCD Microcontroller send data to LCD. The characters should be correctly displayed on LCD screen.

It was verified that the LCD Microcontroller was correctly connected and works as expected.

3.5 Alarm Verification

3.5.1 Speaker

To test the speaker works:

1. Use main microcontroller to input sound data to speaker and then send signal to check if the speaker can beep when signal is sent.

It was verified that the speaker can be controlled by the main microcontroller to beep.

3.5.2 Real Time Clock

To test the RTC works:

1. Connect RTC to Arduino to check if the current time of RTC is the same as real time after setting initial time for RTC.
2. Let main microcontroller send the time data computed by RTC to LCD controller to be printed to LCD screen.
3. Let LCD controller directly printed time data computed by RTC to LCD screen.

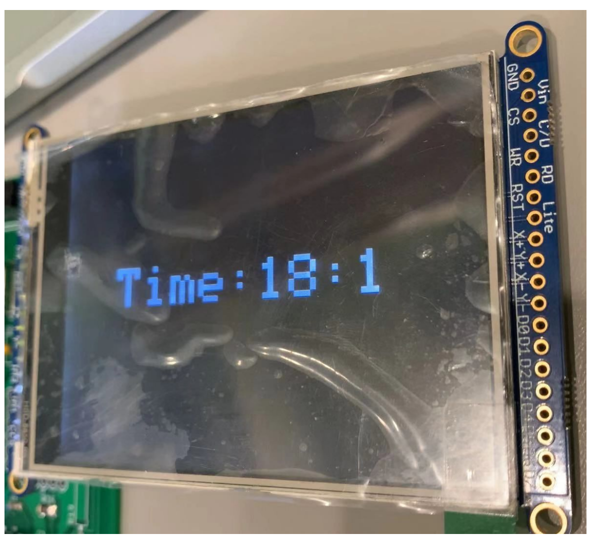

Figure 12. LCD Displays Correct Time with RTC Data

It was verified that the RTC does keep track of the real time, and it is able to connect both microcontroller via I^2C.

3.6 Software Verification

To test the functionality of the Android application:

1. Use the Android app to connect to the Bluetooth module, input and send time and questions strings to the microcontroller and print them on the console (Figure 13).
2. Use the microcontroller to send strings to the Android app and have the app print them on the screen.

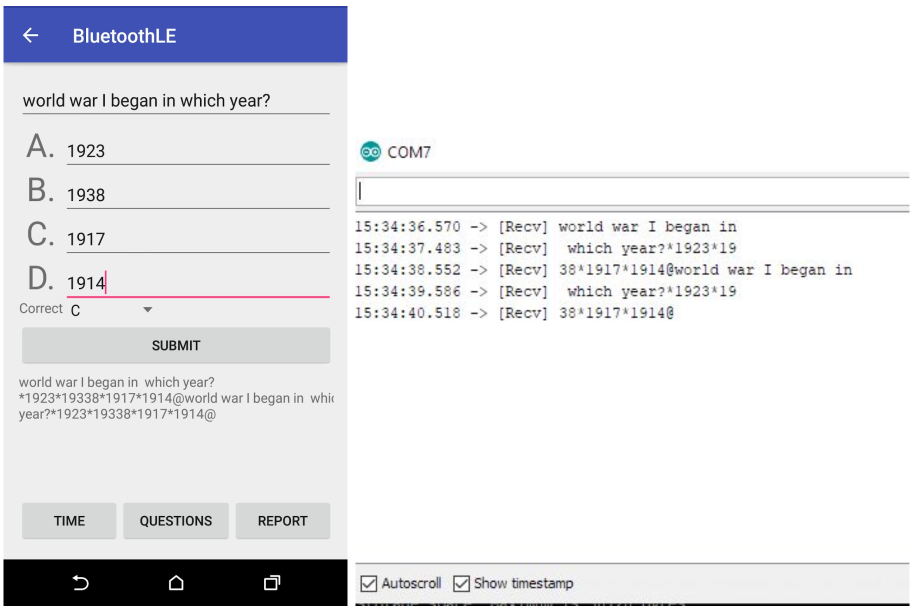

Figure 13. Android Application Sending Data to Microcontroller

During the test, it was found that the Bluetooth Module cannot send more than 20 bytes at a time and therefore the strings being sent to the microcontroller needs to be sent in substrings of length of 20 bytes.

It was verified that the software was able to send and receive strings to and from the microcontroller.

4. Costs

4.1 Parts

Part	Cost(prototype)
10000 mAh Portable Power Bank (Amazon; Anker)	$27.99
3 volt Button Battery (Digikey; CR1220)	$0.95
Microcontroller * 2 (Digikey; ATMEGA328P-PU)	$3.92
Physical Buttons * 4 (Digikey; PTS810 SJM 250 SMTR LFS)	$1.24
Bluetooth Module (Amazon; MDBT40)	$19.48
LCD Module (Adafruit; ILI9341)	$29.95
Speakers (Digikey; CMS-28528N-L152)	$3.59
Real Time Clock (Digikey; 1528-1598-ND)	$13.95
Total	$101.07

4.2 Labor

Salary per hour is estimated to be $20/hour. Total hours to complete the project per person is estimated to be 165 hours (15 h/week * 11 weeks). The total labor cost is estimated to be:

3 person * $20/hour * 165 hour/person * 2.5 = $24,750

5. Conclusion

5.1 Accomplishments

The entire alarm device as well as the Android application work as intended. The Android application allows the users to connect to the alarm device and set alarm time and questions they would like to answer. The alarm device is able to function as an alarm clock, displaying time throughout the day and receive user input through the Android application and effectively ring and display questions for users to answer at designated time. The alarm device interacts with the users and records the time it takes them to turn the device off and the number of correctly answered questions, which are sent to the Android application per the request of the user.

5.2 Uncertainties

Due to memory restrictions, the alarm device is not able to store too many questions. The alarm device does not work reliably if the user input too many questions. Also, due to the size of the LCD display, the alarm device is not able to display the entire question if the user input a very long question.

However, the amount of questions it can store is sufficient for the users to work with for a while. Also, the alarm cannot be turned off unless all questions are answered correctly, which effectively helps the users wake up.

5.3 Ethical Considerations

The Android app we implemented requires the use to turn on location services to work with Bluetooth module. To access the hardware identifications of nearby external devices via Bluetooth (and Wi-Fi) scans, the app must have the ACCESS_FINE_LCATION or ACCESS_COARSE_LOCATION permissions[3]. We should be aware of the potential ethical issues from gaining that access and should not be using the location access to track or monitor users in any way.

Our application should align with the IEEE Code of Ethics in section 7, "To accept responsibility.."[4] there could be potential risks that result from having access to private user data.

5.4 Future Work

There are several possible improvements to the device that we built:

a. Expand the capacity of the device using external memory
b. Develop IOS version of the mobile app
c. Support different ringtones and let the users specify via the mobile app
d. Support multiple alarm times
e. Have a better physical wrapping

References

[1] A. Hess, “10 highly successful people who wake up before 6 a.m.,” *CNBC*, 17-May-2018. [online]. Available: https://www.cnbc.com/2018/05/17/10-highly-successful-people-who-wake-up-before-6-a-m.html. [Accessed 14 Sep. 2019].

[2] M. Floyd, “Almost 60% of 16-34 year olds use a phone as their primary timepiece,” *YouGov*, 05-May-2011. [Online]. Available: https://today.yougov.com/topics/lifestyle/articles-reports/2011/05/05/brother-do-you-have-time. [Accessed: 14 Sep. 2019].

[3] Access to Hardware Identifier [Online]. Available: https://developer.android.com/about/versions/marshmallow/android-6.0-changes.html?#behavior-hardware-id [Accessed: Oct. 27, 2019].

[4] Ieee.org, "IEEE IEEE Code of Ethics", IEEE policies section 7, 2016. [Online]. Available: http://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: Oct. 27, 2019].

Appendix A Requirement and Verification Table

Table 1: R&V for Microcontroller in Control Unit

Requirements	**Verification**
1. Able to get time from RTC module. 2. Able to get data from physical buttons. 3. Able to send data to the microcontroller in user interface. 4. Able to send data to Bluetooth module and get data from Bluetooth module.	1. Write a test program to have the microcontroller turn on LED based on RTC input. 2. Write a test program to have the microcontroller turn on LED based on button input. 3. Press the buttons to control the LED on the microcontroller in user interface. 4. a. Write a test program to have the microcontroller turn on LED based on Bluetooth module input. b. Write a test program to have the Bluetooth module turn on LED based on microcontroller input.

Table 2: R&V for Bluetooth Module

Requirements	**Verification**
1. Able to connect to Android app. 2. Able to receive data from Android app. 3. Able to send data to Android app. 4. Able to send data to microcontroller and able to receive data from microcontroller.	1. Write an Android application to connect LED on microcontroller. 2. Write an Android application to control the LED to light up on the microcontroller. 3. Press buttons and display output on the Android app. 4. a. Write a test program to have the microcontroller turn on LED based on Bluetooth module input. b. Write a test program to have the Bluetooth module turn on LED based on microcontroller input.

Table 3: R&V for Power Supply

Requirements	**Verification**
1. Provides 5 V +/- 2% from the portable battery bank. 2. Provides 3 V +/- 1% from the button battery.	1. Measure the output voltage using an oscilloscope, make sure that the output voltage stays within 2% of 5 V 2. Measure the output voltage using a Voltmeter, make sure that the output voltage stays within 1% of 3 V.

Table 4: R&V for Microcontroller in User Interface

Requirements	**Verification**
1. Able to control the LCD display. 2. Able to receive data from the microcoller in the control unit.	1. Write a program for the LCD controller. to display some text on the LCD display 2. Write a program to have the. microcontroller in the control unit control the LED on the LCD controller.

Table 5: R&V for Physical Buttons

Requirements	**Verification**
1. Button pressed can be detected by the microcontroller over GPIO. 2. Buttons can be easily pressed.	1. Measure the output signal using an oscilloscope, make sure that the output correctly represents whether the button is pressed. 2. Press Button and ensure that it can be done.

Table 6: R&V for LCD display

Requirements	**Verification**
1. LCD Display is able to receive data over SPI.	1. Connect the LCD Display to the SPI port of the microcontroller and send random characters. The characters should be correctly displayed on LCD.

Table 7: R&V for speaker

Requirements	**Verification**
1. Speaker is able to receive data over GPIO and produce audio output when signal is sent.	1. Connect speaker with microcontroller, use microcontroller to input data to speaker and send signal, check if the speaker can produce correct music when signal is sent.

Table 8: R&V for Real Time Clock

Requirements	**Verification**
1. RTC is able to keep track of real time. 2. RTC is able to transmit over I^2C to microcontroller.	1. Current time of RTC is the same as real time after setting initial time for RTC. 2. Time computed by RTC can be received by microcontroller and displayed on LCD.

Table 9: R&V for Software

Requirements	**Verification**
1. Android app is able to connect to Bluetooth. 2. User is able to use the Android app to send data to control unit. 3. Android app is able to receive data sent from the control unit.	1. Write an Android application to connect LED on microcontroller. 2. Write an Android application to control LED to light up on microcontroller. 3. a. Write a test program to have the microcontroller turn on LED based on Bluetooth module input. b. Write a test program to have the Bluetooth module turn on LED based on microcontroller input.