Design and Assembly of a 4-Layer PCB IoT Alarm Clock

Steven Abrego*
*Department of Electrical Engineering, Stanford University, Stanford, CA, 94305

***Abstract*—This paper discusses the design and assembly of a 4-Layer Printed Circuit Board (PCB) Internet-of-Things (IoT) alarm clock as part of the final project for EE 256: Board Level Design. The PCB was designed in Kicad v8.0, and incorporates an onboard ESP32 module for WiFi connectivity to communicate the current date, time, and weather, as well as an onboard FM radio with tunable frequency and volume. All relevant information is displayed for the user on an OLED display.**

***Index Terms*–PCB Design, FM Radio, ESP32, Proximity Sensor, Ambient Light Sensor, CircuitPython, Mu Editor**

I. Introduction

In today's society, being able to quickly design and assemble printed-circuit boards (PCBs) is critical to the advancement of electronic technologies by creating prototypes and test builds. Throughout EE 256: Board Level Design, I have worked towards creating my very own PCB to learn the design process and test my very own product. This paper describes the design process of an IoT Alarm Clock, including the design steps in Kicad v8.0 such as the schematic editor, layout editor, and finally assembly and testing of the board.

II. Board Schematic

For this project, Kicad v8.0 was used for all board schematics, as well as layout and gerber file generation. This software was chosen due to the extensive documentation for beginners, as well as the easy to use interface and long list of libraries for easier design flow and implementation.

This project has many sub-modules all working in conjunction to create the final product. As a result, a hierarchical schematic system was used in the schematic editor of KiCad for easier understanding. Fig. 1 shows the root schematic page, which includes all labeled subsystem schematics and their corresponding connections. The subsystem schematics include the input power circuitry, ESP32 module, microcontroller and supporting circuitry, OLED display module, flash memory circuitry, proximity sensor, ambient light sensor, FM radio circuitry, and microSD card interface and supporting circuitry. Each of these subsystems plays a crucial role in the operation of the overall system and was implemented to ensure successful operation of the IoT alarm clock.

A. Microcontroller

The microcontroller used in this project was an ATSAMD51J19A, which allowed for easy programming through CircuitPython in Mu Editor, similar to that of the Adafruit Feather M4 Express, which was previously assembled and programmed earlier in the course. This microcontroller was part of the original project requirements, and proved to be more than enough computational power for this project. The documentation and robust data sheet for this microcontroller also allowed for easy implementation into the overall architecture.

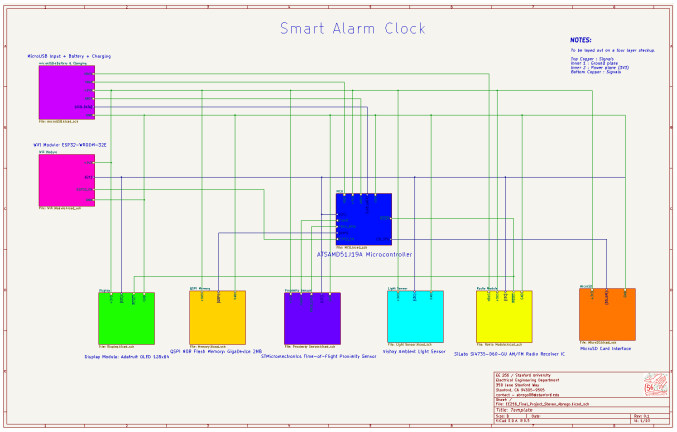

Fig. 1. Root schematic page of all individual systems which comprises the IoT Alarm clock. The subsystems are color-coded as follows: MicroUSB input + Battery + Charging (purple), WiFi module (pink), ATSAMD51J19A Microcontroller (blue), OLED display module (green), QSPI NOR flash memory (mustard), proximity sensor (indigo), light sensor (teal), FM radio (yellow), and microSD card (orange).

B. Input Power

The input power, battery, and charging circuitry was also leveraged from the Adafruit Feather M4 Express, due to the ease of implementation and previous testing with this specific circuitry.

C. ESP32 Wifi Module

For the Wifi Module, an ESP32-DevKit-C was used due to the extensive documentation and preexisting programs for configuring the module. This module communicates with the microcontroller over I^2C, and must be configured as a peripheral instead of a host for correct use. For future revisions, the Adafruit Airlift - ESP32 Wifi breakout board will be used and communicate using SPI to avoid any additional programming for setting the module up as a peripheral, given that extensive amounts of time were spent to achieve this and was not the best use of time.

D. OLED Display

For the display, the Adafruit Monochrome 128x64 OLED display was used due to the extensive libraries used in CircuitPython for configuring the module. The display communicates with the microcontroller over I^2C, and includes libraries for text to display all necessary information.

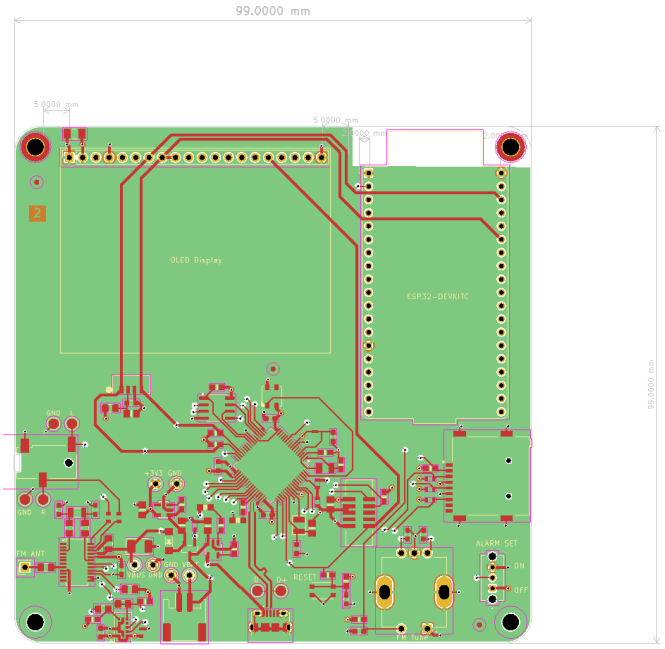

Fig. 2. KiCad board layout including second and third layer plane fills. The second layer (green) is a GND layer and the third layer (orange), is a +3V3 layer. Layer 4 (not shown) is an additional signal layer for routing additional traces.

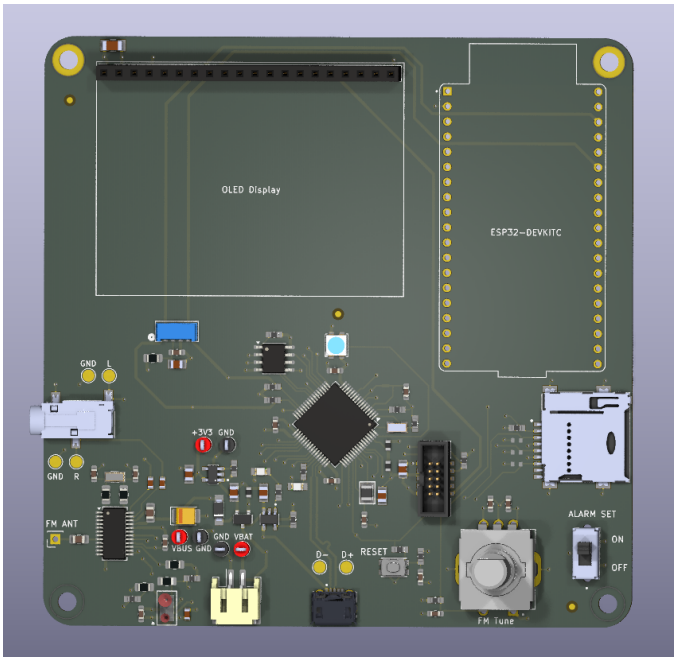

Fig. 3. 3D Rendering of finalized board including 3D models of parts. This also includes test points and 0 Ω resistors for easier testing and verification during the board bring up phase.

E. Flash Memory

An additional 2 MB of flash memory was used for additional programs, should the size of code needed for operation exceeded the storage of the ATSAMD51J19A. The flash memory communicated to the microcontroller over QSPI. The 2 MB of flash memory is an IC from GigaDevice Semiconductor.

F. Proximity Sensor

A proximity sensor from STMicroelectronics was incorporated to meet the project requirement of having multiple sensors on the board. This proximity sensor is responsible for enabling/disabling the FM radio, should the user simply choose to use this feature instead of displaying the time. This sensor communicates with the microcontroller over I^2C.

G. Ambient Light Sensor

An ambient light sensor from Vishay Semiconductor was incorporated also to meet the project requirement of having multiple sensors on the board. This ambient light sensor is responsible for dimming a Neopixel LED depending on the brightness of the room, which can be enabled by the user if the radio knob is pushed down for more than 5 seconds. This feature was intended to be used as a night light, should the user wish to have a light source at their bedside for better visibility at night. This feature was included simply to enhance the functionality of the alarm clock, making good use of the project requirements. This sensor communicates with the microcontroller over I^2C.

H. FM Radio

For the FM Radio, the Si4735-D60-GU IC from Skyworks was used. This particular IC was used due to previous implementation provided, along with supporting preexisting KiCad schematics. An additional 32.768 kHz external oscillator was used alongside the IC, as recommended from the data sheet. This IC communicates with the microcontroller over I^2C.

I. MicroSD Card Interface

To meet project requirements, a push-push style microSD card reader from GCT was implemented to be able to read and write data to a microSD card using SPI.

III. Board Layout

For the board layout, a 4-layer design was chosen for ease of routing signals, as well as easily utilization of GND and +3V3 planes for components to connect to. The 4 layers are as follows (from top to bottom):

1) Signal Layer
2) GND Layer
3) +3V3 Layer
4) Additional Signal Layer

One major design consideration was to place all user-interface components near the edge of the board for ease of use. This includes the 3.5 mm jack for the radio speaker, the battery connector, the microUSB connector, the FM radio encoder knob, the alarm ON/OFF switch, and the microSD card interface. All the components were placed along the edges of the board to avoid having the user accidentally touch another component on the board while trying to interact with

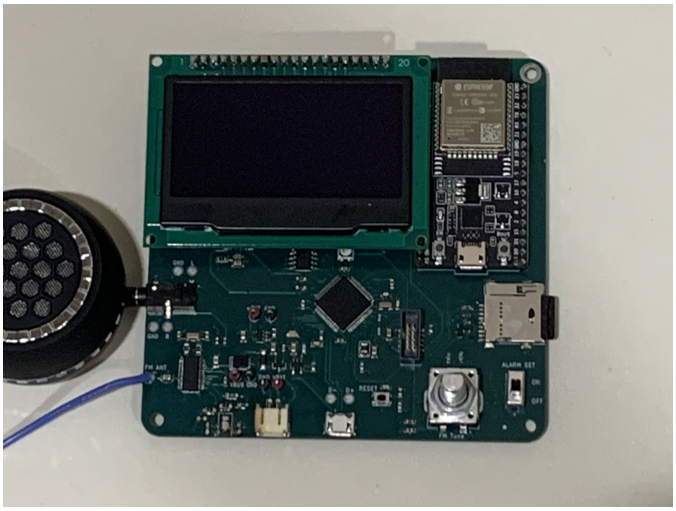

Fig. 4. Finalized assembled board, including all modules as well as attached speaker and FM antenna (left). The antenna is a long blue wire (0.5 meters in length) attached to the edge of the board to receive FM signals.

the board. Fig. 2 shows the finalized board layout and all routing, while Fig. 3 shows a 3D rendering of the board with component models. The 3D model does not include parts for the OLED display or the ESP32 module, but does include their respective connectors (pin adapter for display and holes in board for direct connection of ESP32 module).

IV. Board Assembly & Testing

Once the board gerber files were generated and sent out to be fabricated, a test and board bring up plan was devised to ensure correct functionality of the board. Once the boards arrived, they were inspected for any anomalies, then assembled. After successful placement of parts and reflow, testing was completed to make sure the board was being powered correctly. Here the test points were very helpful in ensuring everything was powered correctly. A test to flash the bootloader onto the microcontroller was conducted, and was a success on the first try. Once the microcontroller was configured correctly, testing of each individual subsystem programmatically was conducted to ensure correct functionality. Once this was completed, programming of the the overall functionality of the board was began. Fig. 4 shows a fully assembled board with all external components attached. The board assembly and testing was my favorite part of the experience as I got to see my board come to life and all my hard work pay off. At this point I could begin to design the functionality of the board from a program perspective and optimize the user experience of this IoT Alarm Clock.

V. Conclusion

To conclude, the design and assembly of a 4-Layer PCB IoT Alarm Clock was completed in Kicad v8.0, assembled and tested by hand, and programmed using CircuitPython in Mu Editor. Future revisions of this design will allow for an on-board speaker with a class-D audio amplifier for a lighter design, and the ESP32 module will be replaced by an Adafruit AirLift ESP32 module for easier programmability. Nonetheless, this current revision was a success as a first revision, proving successful from a board-level and assembly perspective. I truly enjoyed working on this project and I am grateful for all I was able to learn about PCB design in just a single quarter. This class has enabled me with the skills and know-how to design a PCB myself from start to finish, and design whatever I can imagine. PCB design truly is a powerful skill for any Electrical Engineer, and will greatly help me in my career.

VI. Acknowledgements

I would like to thank Steve Clark and Brian Pham for their guidance and mentorship throughout the duration of this project.