ECEN 5833

Low Power Embedded Design Techniques

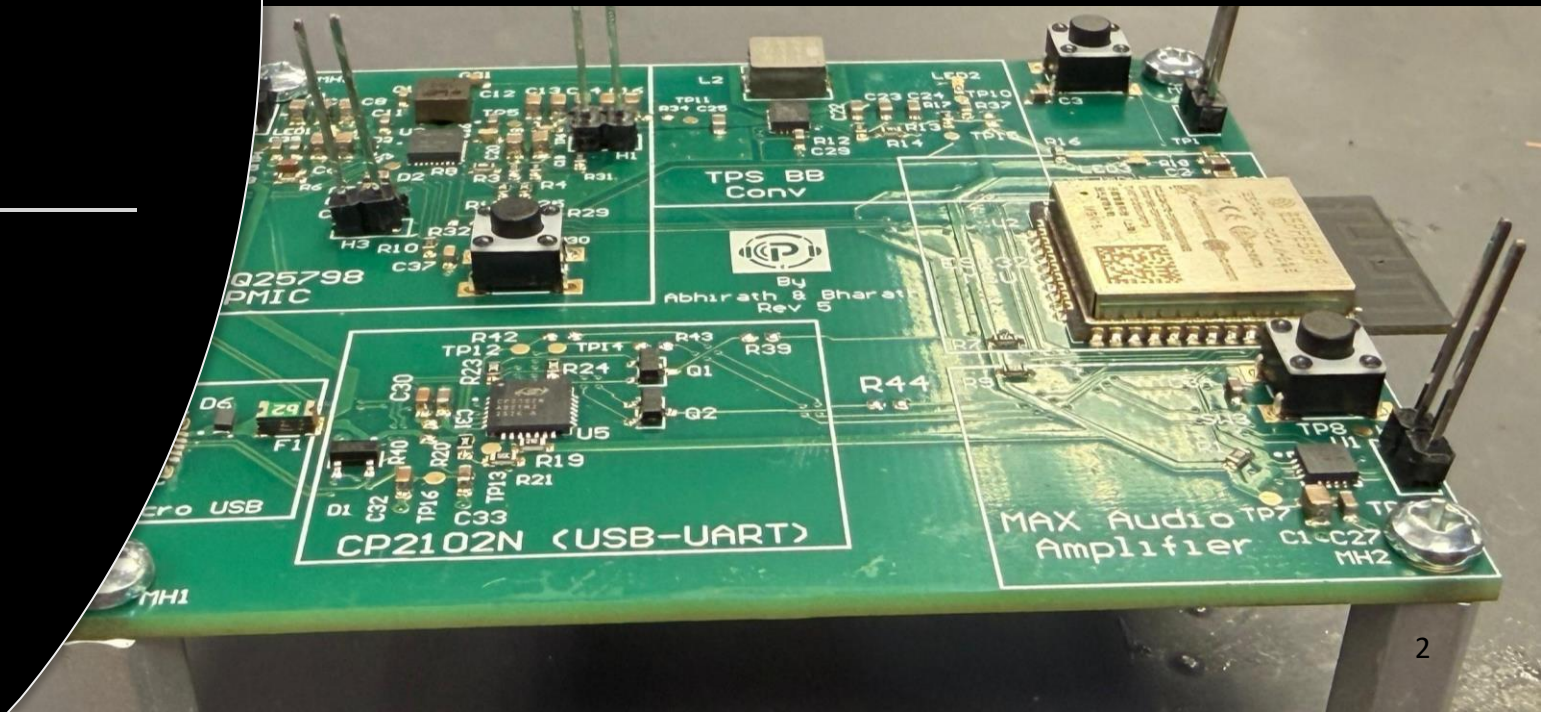
Be Boulder.

University of Colorado Boulder

Team-4 (Pulse)

A Solar Bluetooth Speaker

Abhirath Koushik, Bharath Varma



Presentation Roadmap

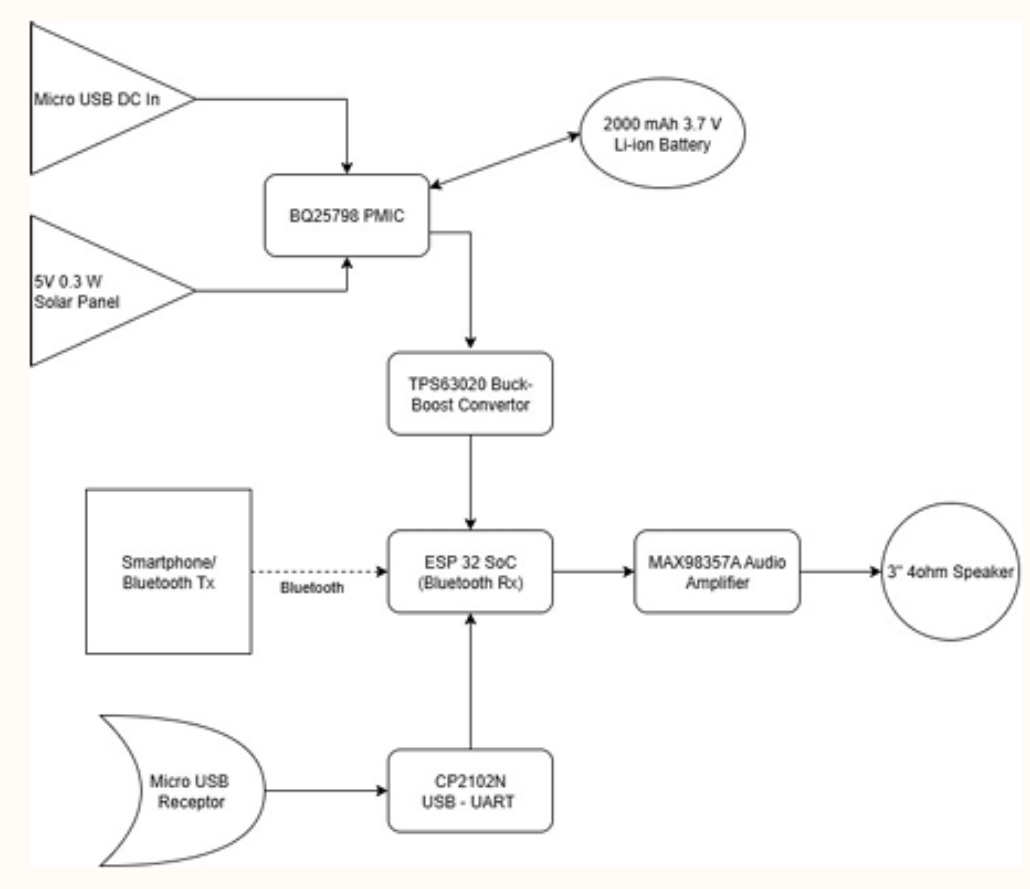
- Project Description & Use Cases
- System Block Diagram (each IC in detail)
- Hardware Simulations
- Software Design
- Low Power Mode Timeline
- Energy Consumption
- Hardware Issued Resolved
- Lessons Learned
- Summary

Project Description & Use Cases

- Our project is to build a Portable Wireless Speaker, integrated with Solar Charging and Bluetooth Audio Streaming.

- This is a self-sufficient Bluetooth Speaker that is easy to carry and eco-friendly, with the ability to stream music real-time. It offers the convenience of a portable speaker with the added benefit of solar energy.

System Block Diagram



CP2102

- Converts USB communication from Micro-USB Receptor to UART Serial interface for programming the ESP32
- Provides the Serial Interface for debugging and display I2C info and Music playback statuses during program execution
- Ensures auto-programming of the ESP32 through DTR and RTS control signals

BQ PMIC

- Manages power from Multiple Sources (USB and Solar) to charge and discharge the 2000 mAh Li-ion battery, and power the entire system
- Uses the MPPT algorithm to maximize the power extraction from the Solar Panels
- Supports multiple charging phases (Fast Charge, Trickle Charge) that is automatically selected based on the charging levels and is also configurable through I2C.
- We confirmed that the 2000 mAh battery is both charging and discharging correctly when connected to the BQ PMIC.
- Programmed the ESP32 to retrieve vital information from the BQ PMIC via I2C
- Voltages at BAT, SYS and VAC pins can be transmitted through I2C to give us the Battery Level, Charging Status and Solar Input voltages Battery Level and Charging Status to the ESP32
- Even without any programming, we used a 3ohm at PROG Pin to configure the BQ for a Single Cell (1S) battery configuration with a default charging of 4.2V

TPS Buck-Boost

- The TPS IC acts as a High-Efficiency DC-DC Switching Regulator. Its primary role is to ensure stable voltage supply for the Power Rail.
- We designed it based on the simulations from TI Webench and Application notes.
- Provides stable 3.3V to the input power rails of both ESP32 and MAX audio amplifier.
- During our energy measurements, we confirmed a stable 3.258 V using the Test Points placed on the board.

ESP32 MCU

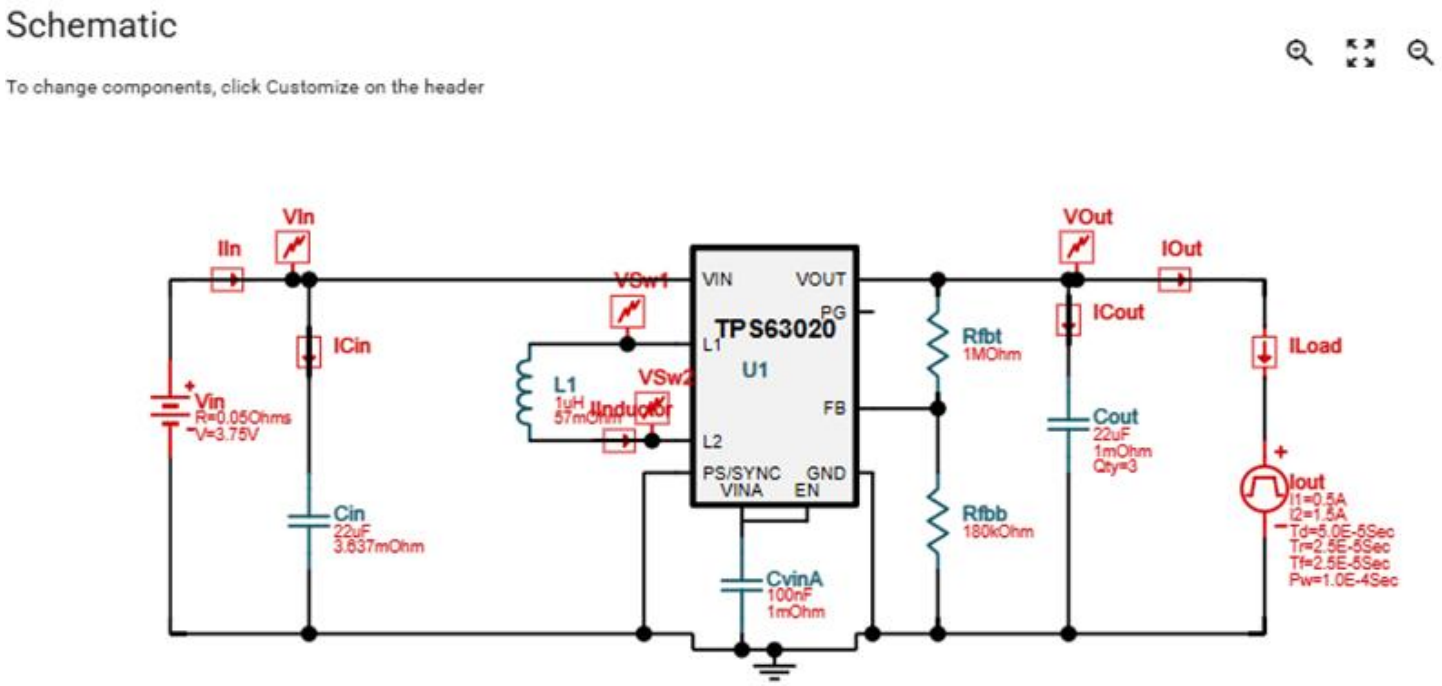
- Central MCU that controls all system operations
- Receives audio data from a phone via Bluetooth Classic with A2DP profile
- Processes digital audio and send it to MAX Audio Amplifier through I2S (BCLK, LRCLK and DIN Signals)
- Manages the Power Modes (Active and Deep Sleep) to minimize energy consumption
- Retrieves Charging and Voltage info from the BQ PMIC through I2C
- Manages the GPIO Push Buttons that provides Play, Vol+- controls for the Speaker

MAX Audio Amplifier

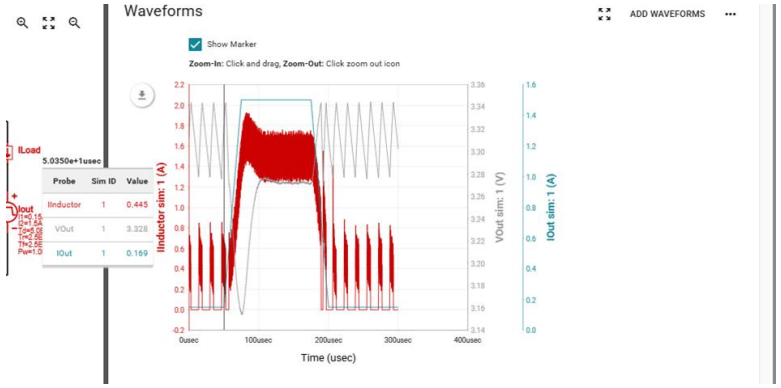
- This is a Digital Class-D audio amplifier that converts I2S audio data into amplified speaker output
- It receives the digital PCM audio (BCLK, LRCLK, DIN) from ESP32 and converts to Analog Amplified Output.
- MAX provides a configurable Gain which can be selected through GAIN Pin. We connected to VDD providing 6dB
- We configured SDMODE Pin to (L+R)/2 mode (with 634k ohm) which ensures the complete music is audible in a Mono speaker
- MAX also supports Active and Shutdown states to minimize energy consumption through SDMODE Pin

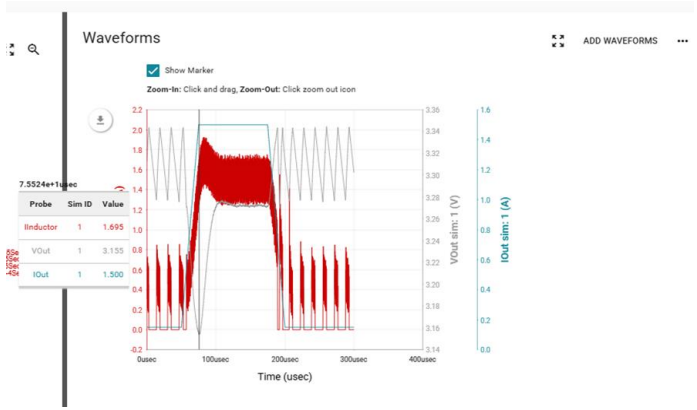
Hardware Simulation

- Our TPS63020 Buck Boost Converter outputs the load power rail. The output of the TPS63020 Buck-Boost Converter powers the ESP32 MCU and a MAX98357A amplifier. So, we will be using Webench to figure out the optimal capacitance value by simulating a load transient simulation.



Hardware Simulation – Dynamic Load Simulation



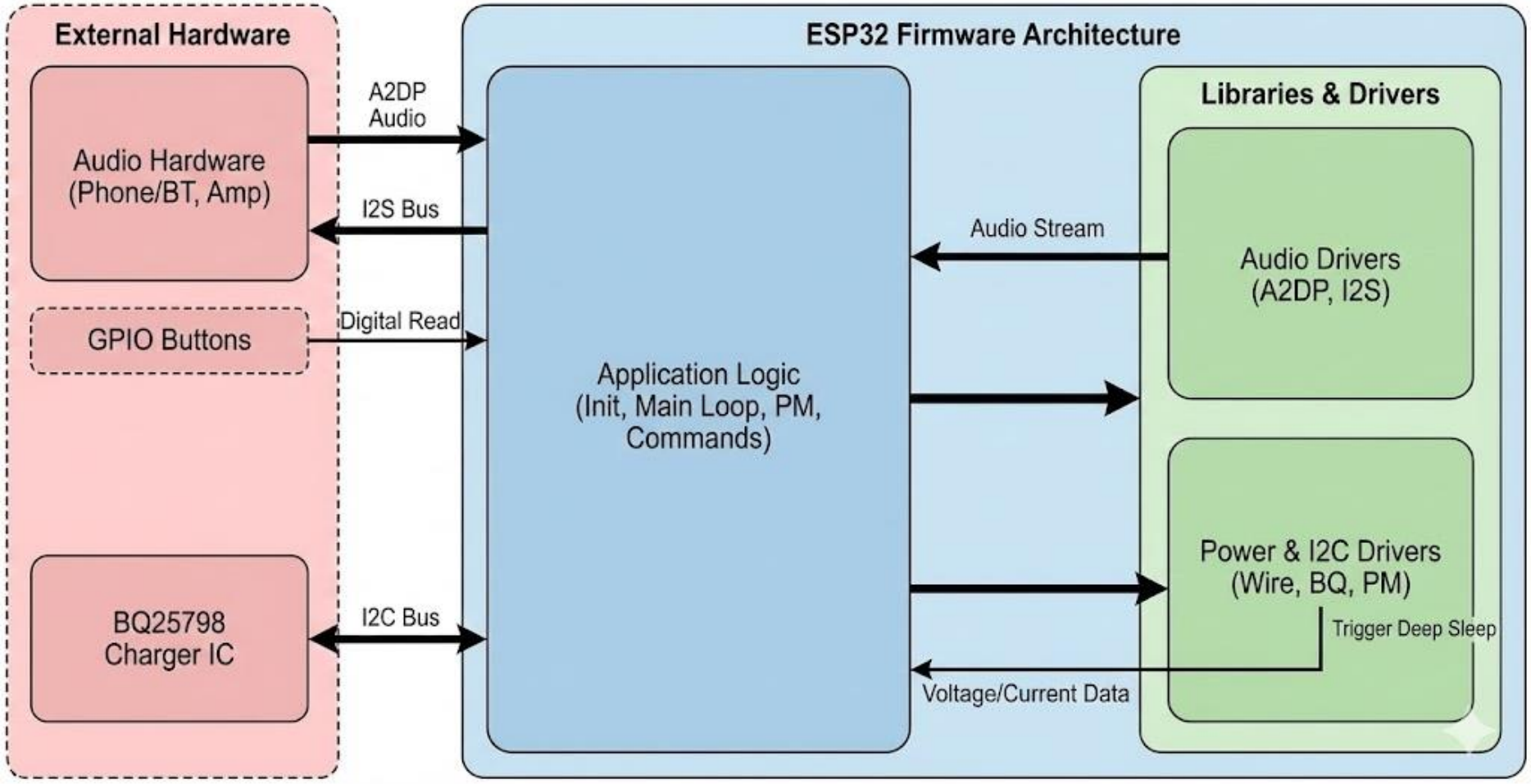


- At Vin = 5V, right before the VOut drops, VOut = 3.328

At Vin = 2.5V, at the point where VOut is the lowest. VOut = 3.155

- There is a 173 mV drop. The minimum voltage is within the working voltage of ESP32 and MAX98357A. So, there is no requirement to add a bulk. There is a 173 mV drop. The minimum voltage is within the working voltage of ESP32 and MAX98357A. So, there is no requirement to add a bulk capacitance.
- Also, as confirmed from our board measurements, our voltage rail stays in the range supported by both MAX and ESP32.

Software Design



Software Design

- We have used Arduino IDE to program the ESP32 as it has extensive libraries for A2DP, GPIO that we have used
- The Main Control loop handles the Push button polling
- Bluetooth A2DP audio pipeline is directed towards the I2S driver in the background, ensuring there is no interruption to the music playback
- A State Machine is used to periodically communicate through I2C to the BQ PMIC to monitor Charging and Battery Voltages
- Timers are used to check for Pause state, and trigger Deep Sleep if inactive for more than 2 minutes

```
=== BQ25798 Battery / Charge Status ===
Charge state : Fast charge (CC)
Input source : USB SDP

Voltages:
  VBAT : 4.088 V (89%)
  VSYS : 4.084 V
  VBUS : 4.912 V
  VSOLAR: 1.074 V (VAC2)
Current:
  Charging  : 462 mA
=======================================
```

Low Power Mode Timeline

- **Streaming Mode**

This is the normal mode of operation when Bluetooth, GPIO, I2C and all peripherals are active. MAX amplifier SD Pin is in High State

- **Deep Sleep Mode**

This is the Power Saving Mode when Bluetooth, I2C and Peripherals are stopped. Only a specified RTC GPIO (Pin 17) is used to wake up the ESP from this mode
MAX amplifier SD Pin is in Low mode, shuts down the amplifier and saves power

```
Voltages:
  VBAT : 4.121 V (92%)
  VSYS : 4.162 V
  VBUS : 4.916 V
  VSOLAR: 0.254 V (VAC2)
Current:
  Charging  : 462 mA
=====================================

[PM] Idle for 30009 ms. Sleeping.
[PM] Entering DEEP_SLEEP
[PM] MAX disabled (SD_MODE = LOW)
[PM] Deep sleep starting... press Vol-/Prev button to wake
ets Jul 29 2019 12:21:46

rst:0x5 (DEEPSLEEP_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:1
load:0x3fff0030,len:4980
load:0x40078000,len:16612
load:0x40080400,len:3480
entry 0x400805b4
Woken from deep sleep!
[PM] Woken from deep sleep!
[PM] MAX enabled (SD_MODE = HIGH)
[PM] Initialized, timer running
BQ25798 REG00 value: 0x04
Starting Bluetooth Now!
```

Energy Consumption

- Time to Charge:

When ESP BT connected but no music plays, the battery gets Charged from 88% to 100% in 30 minutes

- Time to Discharge:

Moderate Mode: In 1hr, the battery level reduced from 98% to 89%

Loud Mode: In 1hr, the battery level reduced from 98% to 84%

Measured Power Analysis

We used a shunt resistor of 1.6 Ohm and Regulated Voltage at the Power Rail was 3.258V

Mode	Amplifier Volume	Voltage Diff Across Resistor	Current	Power (P=3.258 * I)
Amplifier Quiet Level	50%	125 mV	I = 125/1.6 = 78.13 mA	254.5 mW
Amplifier Moderate Level	75%	150mV	I = 150/1.6 = 93.75 mA	305.4 mW
Amplifier Loud Level	100%	300mV	I = 300/1.6 = 187.50 mA	610.9 mW
ESP32 Deep Sleep	NA	0.5mV	I = 0.5/1.6 = 0.31 mA	1 mW
ESP32 No Music but BL Active	NA	117mV	I = 117/1.6 = 73.13 mA	238.2 mW
ESP32 Wake Up	NA	46.5mV	I = 46.5/1.6 = 29.06 mA	94.67 mW

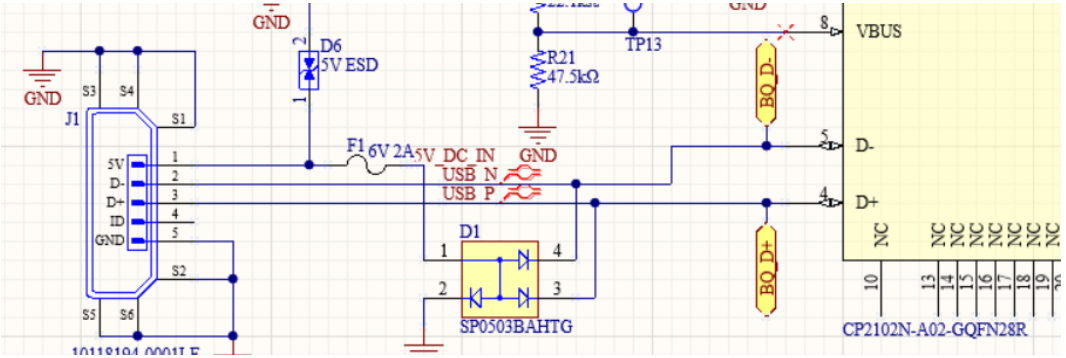
Initial Power Analysis with Measured

Amplifier Mode	**Streaming (2 h)**	**Idle (600 sec)**	**Deep-Sleep (21.831 h)**	**Startup (7.5 sec)**	**Total / Day (In Wh)**	**Total/Day (in mAh)**	**Measured Total/Day**
Quiet (10-30% volume)	3.7V * 270mA * 2h = 1.998 Wh	0.00049 Wh	0.00086 Wh	0.00034 Wh	**2.00**	**540.54**	0.508 Wh
Moderate (30-60% volume)	3.108 Wh	0.00049 Wh	0.00086 Wh	0.00034 Wh	**3.11**	**840.54**	0.610 Wh
Loud (60-100% volume)	5.328 Wh	0.00049 Wh	0.00086 Wh	0.00034 Wh	**5.33**	**1440.54**	1.25 Wh

Hardware Issues Resolved

- Incorrect Diode Connections

During our board bring-up, we noticed that CP2102 was not detected in our computer. In our detailed analysis, we understood that this was due to an Internal connection between the Diode 1(5V Input) and 4,3 terminals (of USB lines). Due to this, the host could not recognize the USB lines.



- Added Push Buttons to ESP GPIO Pins

We added additional GPIO Push Buttons that are directly soldered to the ESP32 IO Pins. These are for Play/Pause, Vol+/Next and Vol-/Prev Track features that can be controlled from the speaker itself.

Lessons Learned

- Initial Battery Sizing was Inadequate – We started with a 1200mAh but our theoretical energy calculations showed that 2000mAh battery was required for loud amplifier mode, if used for multiple hours

- Firmware: BQ PMIC Watchdog Timer needed to be disabled before enabling ADC - Without disabling the watchdog, the PMIC resets I2C communication, causing battery voltage readings to fail and the same voltage was printed each time

- MAX Amplifier Gain and Channel – This is not firmware controlled, depending on our connection of Gain Pin to VDD or GND and the connection of SDMODE pin to an appropriate resistance value, the Gain and Channel were selected

- Understanding Application Circuit Values – In the BQ application circuit, the Prog Pin resistor was mentioned as 6.04k ohm. However, for our 1S Cell configuration, the Prog pin resistor should be 3k ohm. We learnt the importance and skill of reading and understanding the datasheets.

Summary of Goals Achieved

List of Features:

- Solar Panel used as supplementary source for battery charging ✓
- Speaker will work while being charged as well ✓
- Battery Level Indication is shown on the Speaker itself ✗
- LED Indications for Charging, Low Charge, Pairing etc ✓
- Will switch off automatically if there is any inactivity for 2 mins (Standby Mode) ✓

Stretch Goals/Features:

- Play/Pause Button on the Speaker ✓
- Volume Knob/Buttons on the Speaker ✓
- Syncing between Speakers to play the same song ✗

Live Demo and QnA