



# **Project Report**

## **STM32 SoundBox**

**Submitted By,**

Vishesh Yadav(2021UEC2600)

Electronics and Communication Engineering

# Contents

<b>1</b>	<b>Synopsis</b>	<b>1</b>
<b>2</b>	<b>Introduction</b>	<b>2</b>
<b>3</b>	<b>Schematic Diagram</b>	<b>3</b>
<b>4</b>	<b>PWM Audio</b>	<b>4</b>
4.1	Audio playback . . . . .	4
4.2	STM32F103C8T6 PWM description . . . . .	5
4.3	Low-pass Filter . . . . .	6
<b>5</b>	<b>SPI Flash Memory</b>	<b>7</b>
5.1	Introduction . . . . .	7
5.2	Pinout . . . . .	7
5.3	STM32 SPI Configuration . . . . .	8
5.4	Hardware Configuration . . . . .	9
<b>6</b>	<b>ESP8266-01 Wifi module</b>	<b>10</b>
6.1	Hardware configuration . . . . .	10
6.2	Software configuration . . . . .	11
<b>7</b>	<b>HC-05 Bluetooth module</b>	<b>12</b>
7.1	Hardware configuration . . . . .	12
7.2	Software configuration . . . . .	13
<b>8</b>	<b>ILI9341 TFT display</b>	<b>14</b>
8.1	Introduction . . . . .	14
8.2	Circuit Diagram . . . . .	15
<b>9</b>	<b>Working</b>	<b>16</b>
<b>10</b>	<b>Bill of Material</b>	<b>17</b>
<b>11</b>	<b>End Result</b>	<b>18</b>

# 1. Synopsis

This project presents a audio playback system designed around the STM32 microcontroller. Leveraging Pulse Width Modulation (PWM) technology, the system meticulously generates audio signals, subsequently amplified by a dedicated PAM audio amplifier to ensure clear sound output.

To elevate beyond the conventional controls, the system incorporates Wi-Fi and Bluetooth modules, enabling data reception from a user's phone. This empowers users with the ability to remotely control.

Furthermore, a TFT display is integrated to provide visual feedback and enhance user interaction. This display can potentially showcase a multitude of information, such as temperature, AQI(Air quality index) and the amount received.

## 2. Introduction

This project brings you a sound box. Built using an STM32 microcontroller, this uses PWM (Pulse Width Modulation) to make sounds. A PAM amplifier then boosts the sound.

The software development focuses on three key aspects:

- Data Reception and Processing: The system's software receives transaction data through either Wi-Fi or Bluetooth and processes it to identify crucial information like transaction type and amount.
- Audio Cue Generation: Based on the processed data, the software triggers the PWM module to generate pre-programmed audio cues corresponding to different transaction types. For instance, a unique chime might signify a successful transaction, while a different sound could indicate a failed transaction.

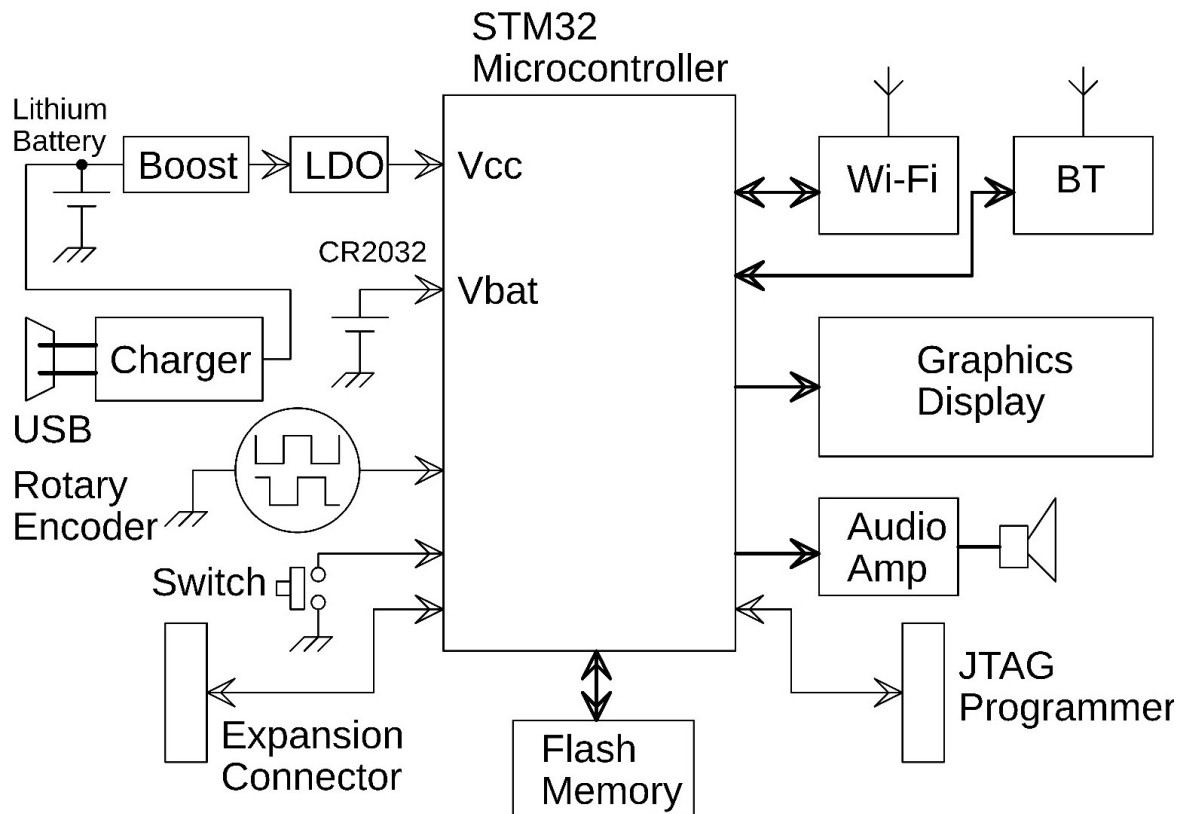
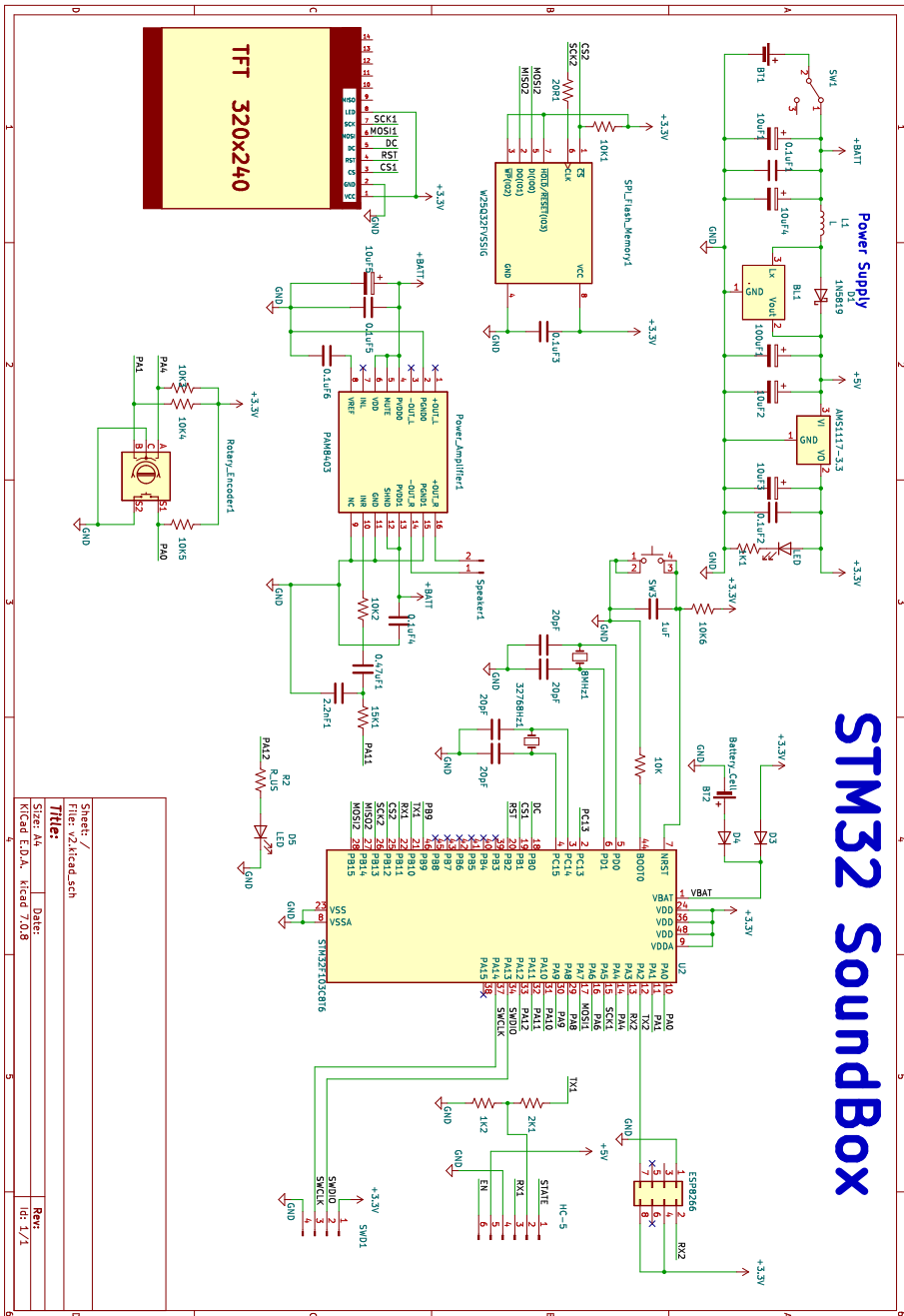


Figure 1: STM32 Soundbox Block Diagram

### 3. Schematic Diagram



# 4. PWM Audio

## 4.1 Audio playback

Audio playback requires two dedicated timers:

- Timer used as system timer, SysTimer: generates an interrupt at a programable rate.
- Timer used in PWM mode . This timer should be able to produce high frequency PWM signal.

The .WAV file data can be stored in a SPI-based Flash , the internal Flash memory does not have sufficient capacity.

To playback the audio, the microcontroller reads each 8-bit sample and then outputs

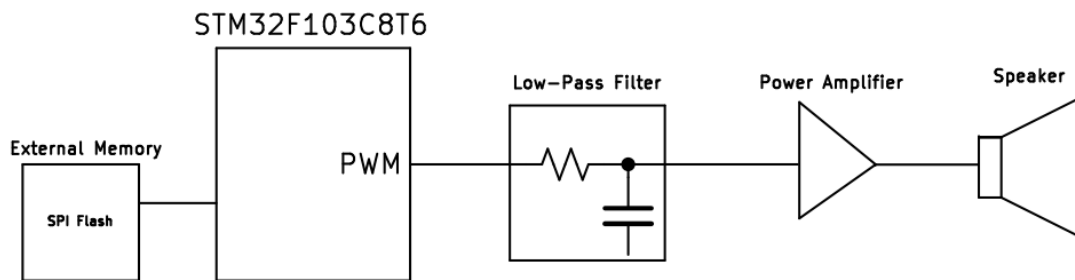


Figure 2: Block Diagram

it to a digital-to-analog converter (DAC) at the desired sample rate. The analog wave produced by the DAC is amplified and fed to a speaker to produce the sound. As the

STM32F103C8T6 MCU doesn't have a on-chip DAC, an alternative method is used to implement 1 channel DAC .Such method is the use of built-in PWM to generate a signal whose pulse width is propotional to amplitude of the sample data.

The PWM signal is then integrated by a low-pass filter to remove high frequency components, leaving only low-frequency content. The out of the low-pass filter provid-eds a reasonable reproduction of the original analog signal.

## 4.2 STM32F103C8T6 PWM description

This section introduces basic concepts of PWM generation through timer of the STM32 MCU.

Pulse Width Modulation consists of a signal of fixed period whose duty cycle is variable. The PWM period and duty cycle are determined by the value of the Prescaler, Auto Reload register (ARR), and Capture/Compare register (CCR) respectively.

The PWM output signal period is fixed by the ARR register which determines the maximum value that the counter can count before starting a new period.

$$\text{Period} = \left( \frac{\text{Timer Clock}}{\text{Desired Frequency}} \right) - 1$$

$$\text{Timer Clock} = \left( \frac{\text{Clock Frequency}}{\text{Prescaler Value}} \right)$$

Taking,

Clock Frequency = 72MHz

Prescaler = 1

Desired Frequency = 281KHz

Substituting above, values in the formula we get Period = 255.

The value of ARR (Auto Reload register) will be 255.

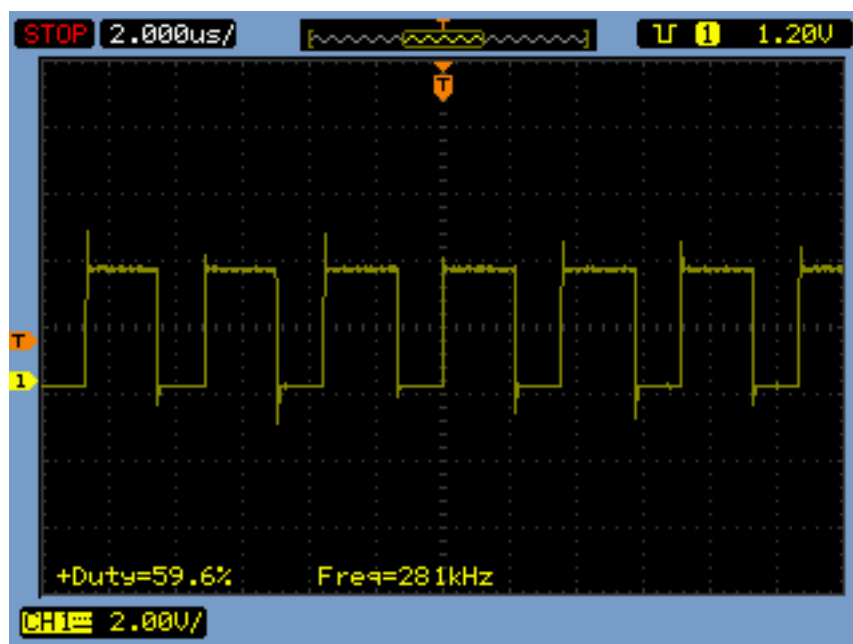


Figure 3: PWM signal with desired frequency

### 4.3 Low-pass Filter

The PWM output signal is then fed to a simple RC low-pass filter to generate an output voltage directly proportional to the average time spent in the HIGH state. When this is averaged over time, we get a reasonable reproduction of the original analog signal.

There are several design considerations involved in the selection of the cut-off frequency of the low-pass filter. Primarily, the filter cut-off frequency must be much lower than the PWM frequency to reduce the noise generated by PWM switching. A PWM frequency of 10 times the cut-off frequency is a generally sufficient.

From a hardware standpoint, a higher PWM frequency is easier to filter, but it corresponds to a smaller reload value and therefore results in lower resolution. By contrast, a smaller PWM frequency is difficult to filter.

The values for R and C are computed with this following formula:

$$f = \frac{1}{2\pi RC}$$

where f is the cut-off frequency of the filter.

Since the cut-off frequency of the low-pass filter is set at 1kHz, the R and C component values are calculated as: R = 15 k $\Omega$ , C = 10nF. The output of the low-pass filter is passed forward to a power amplifier to drive a speaker.

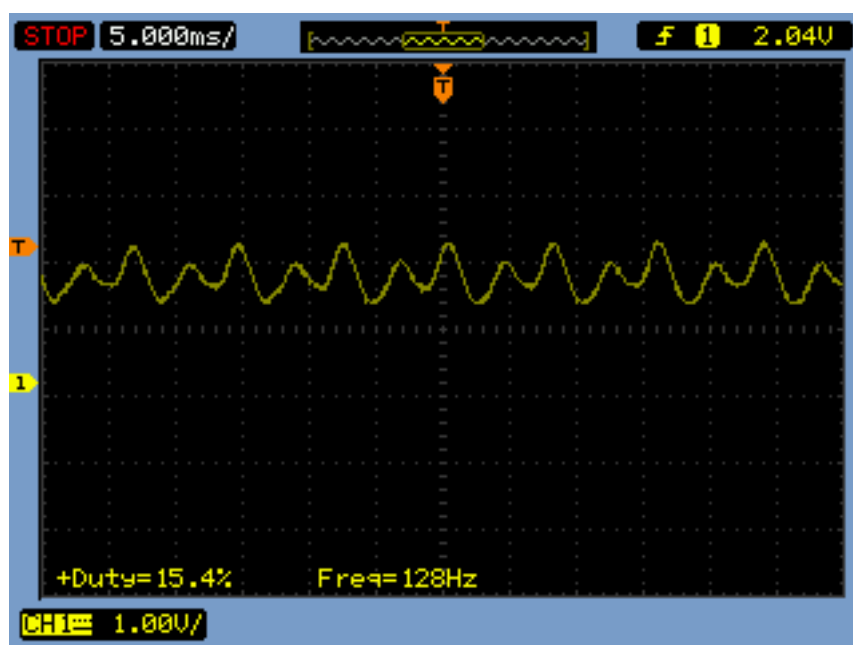


Figure 4: Low-pass filter output



# 5. SPI Flash Memory

## 5.1 Introduction

The W25Q32FV is a 32M-bit serial flash memory with 4KB sectors and support for Dual/Quad I/O. It operates within a voltage range of 2.7V to 3.6V and is available in various package types to cater to different application requirements.

This flash memory is designed to meet industry standards, ensuring reliability and compatibility with existing systems. Winbond's flash memory products are known for their performance, reliability, and adherence to industry standards, making them suitable for a wide range of applications including consumer electronics, industrial control systems, and more.

## 5.2 Pinout

The W25Q32FV flash memory chip comes in various package types with different pin configurations. The pinout includes the following key pins and functions:

- /CS (Chip Select): This is the chip select input and is used to select the chip for communication.
- DO (IO<sub>1</sub>) (Data Output): This is the data output pin for Standard, Dual, or Quad SPI instructions.
- /WP (IO<sub>2</sub>) (Write Protect): This pin is used for write protect input for Standard/Dual SPI and is not available for Quad SPI.
- GND (Ground): This is the ground pin for the power supply.
- VCC (Power Supply): This is the power supply pin.
- /HOLD or /RESET (IO<sub>3</sub>): This pin is used for Hold or Reset input and is not available for Quad SPI.
- DI (IO<sub>0</sub>) (Data Input): This is the data input pin for Standard, Dual, or Quad SPI instructions.
- CLK (Serial Clock): This is the serial clock input for SPI communication.

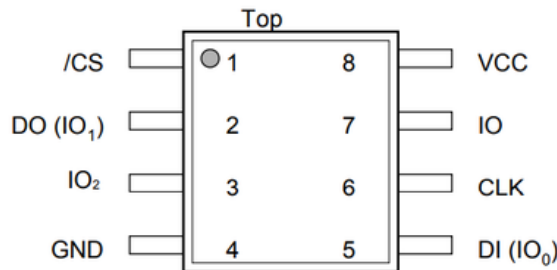


Figure 5: Pinout of W25Q32FV

## 5.3 STM32 SPI Configuration

The Following configuration were done to run the SPI Flash memory with suitable read and write speeds.

### Software Configuration

- Set up the STM32 microcontroller to use the SPI peripheral for communication with the W25QXX chip. This involves configuring the SPI interface, setting the data transfer mode as Full-Duplex Master and sufficient baud rate 9 MBits/s in this case.
- Initialize the GPIO pin that is connected to the chip select (CS) pin of the W25QXX chip as an output pin.

SPI2 Mode and Configuration	
Mode	
Mode	Full-Duplex Master
Hardware NSS Signal	Disable

Configuration	
Reset Configuration	
✓ NVIC Settings	✓ DMA Settings
✓ Parameter Settings	✓ GPIO Settings
Configure the below parameters :	
Search (Ctrl+F) [Icons]	
Basic Parameters	
Frame Format	Motorola
Data Size	8 Bits
First Bit	MSB First
Clock Parameters	
Prescaler (for Baud Rate)	4
Baud Rate	9.0 MBits/s
Clock Polarity (CPOL)	Low
Clock Phase (CPHA)	1 Edge
Advanced Parameters	
CRC Calculation	Disabled
NSS Signal Type	Software

Figure 6: SPI configuration

## 5.4 Hardware Configuration

- Connect the power supply (VCC and GND) of the W25QXX chip to the appropriate power pins on the STM32 microcontroller.
- Connect the SPI communication lines (SCK, MISO, MOSI) of the W25QXX chip to the corresponding SPI pins on the STM32 microcontroller.
- Connect the chip select (CS) pin of the W25QXX chip to a GPIO pin on the STM32 microcontroller.

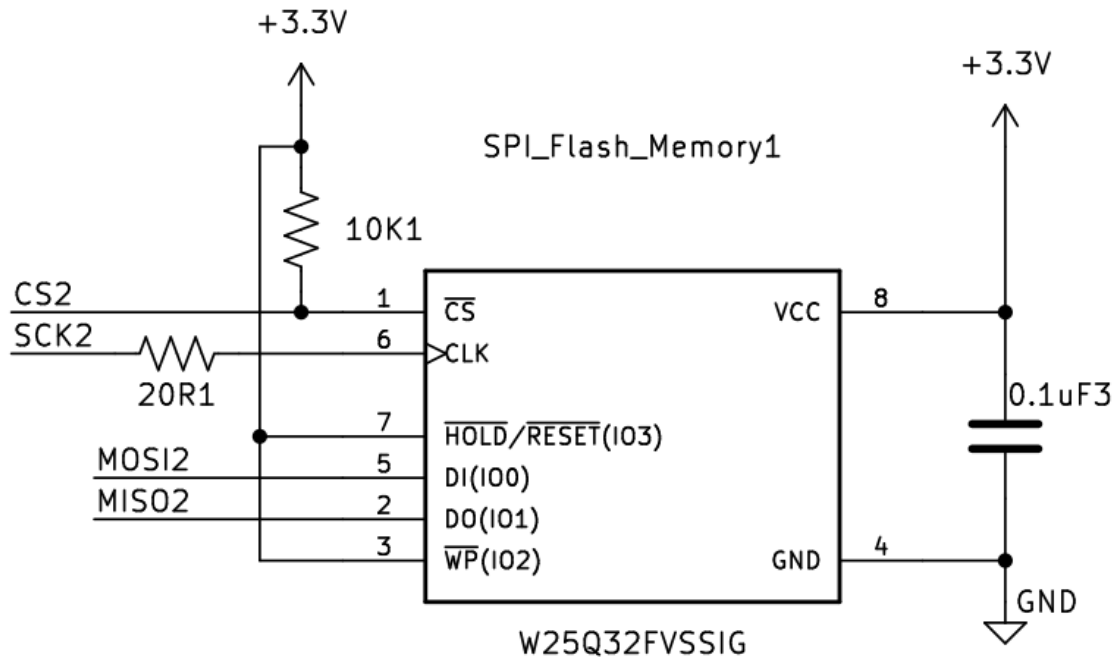


Figure 7: Hardware Connections

Connect the power pins (VCC and GND) of the W25QXX chip to a 3.3V power supply of the STM32 with suitable decoupling capacitor(0.1uF capacitor is used in above circuit).

Connect the SCK (Serial Clock), MISO (Master In Slave Out), and MOSI (Master Out Slave In) pins of the W25QXX chip to the corresponding SPI pins on the STM32 microcontroller. These pins are usually labeled as SCK, MISO, and MOSI on the STM32 microcontroller.

Connect the chip select (CS) pin of the W25QXX chip to a GPIO pin on the STM32 microcontroller. This pin will be used to enable and disable communication with the W25QXX chip.

*\*These connection done are for Standard SPI mode if you want to use any other mode please refer to the data sheet*

## 6. ESP8266-01 Wifi module

### 6.1 Hardware configuration

- Connect the VCC pin of the ESP8266 module to a 3.3V power supply. It's important to note that the ESP8266 operates at 3.3V and is not 5V tolerant, so connecting it to a 5V power supply directly can damage the module.
- Connect the TX (transmit) pin of the ESP8266 module to the RX (receive) pin of the microcontroller .
- Connect the RX (receive) pin of the ESP8266 module to the TX (transmit) pin of the microcontroller .
- ESP8266 modules have additional pins for specific functions, such as reset, enable (EN), and GPIO pins for general-purpose use. The enable pin be pulled up to enable functioning of ESP8266 and reset can be used if required.

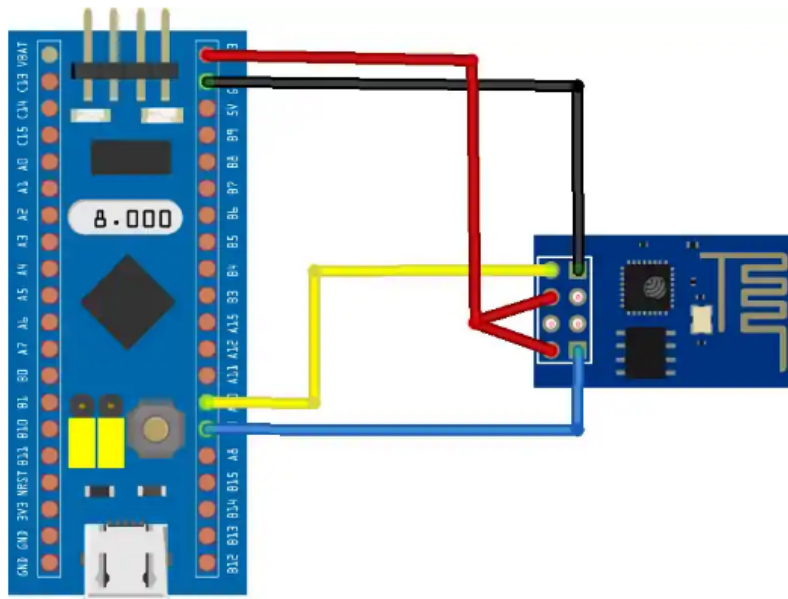
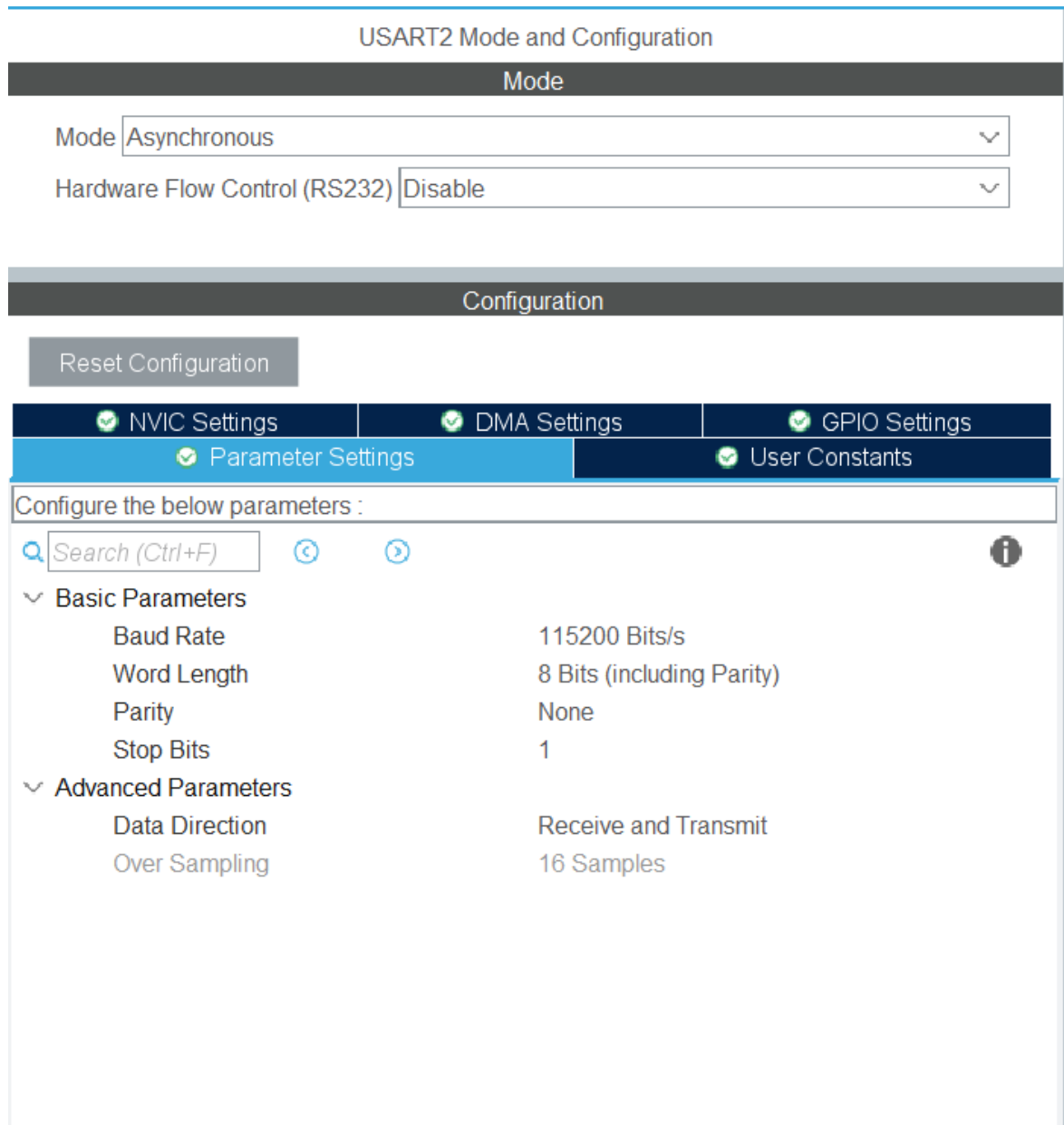


Figure 8: ESP8266 connections with STM32

The module can be used to add Wi-Fi connectivity to a wide range of devices, including microcontrollers, sensors, and other embedded systems. It supports TCP/IP communication, allowing devices to send and receive data over Wi-Fi networks. Additionally, the ESP8266 can function as a web server, host web pages, and communicate with cloud services, making it a powerful tool for building connected and smart devices.

## 6.2 Software configuration

- After connecting the ESP8266 module following the above step, now comes the software part. ESP8266 uses UART to communicate with the STM32. The microcontroller sends AT commands to perform the actions required.
- The baud rate for communication is set to 115200Bits/s.



USART2 Mode and Configuration

Mode

Mode

Hardware Flow Control (RS232)

Configuration

Reset Configuration

✓ NVIC Settings    ✓ DMA Settings    ✓ GPIO Settings

✓ Parameter Settings    ✓ User Constants

Configure the below parameters :

▼ Basic Parameters

Baud Rate	115200 Bits/s
Word Length	8 Bits (including Parity)
Parity	None
Stop Bits	1

▼ Advanced Parameters

Data Direction	Receive and Transmit
Over Sampling	16 Samples

Figure 9: ESP8266 UART configuration

# 7. HC-05 Bluetooth module

## 7.1 Hardware configuration

- Connect the VCC pin of the HC-05 module to a 5V power supply. Ensure that the voltage level is compatible with the module's specifications(3.6-6V).
- Connect the TX pin of the HC-05 module to the RX pin of the STM32 microcontroller.
- Connect the RX pin of the HC-05 module to the TX pin of the STM32 microcontroller.
- Ensure that the STM32 microcontroller and the HC-05 module share a common ground.

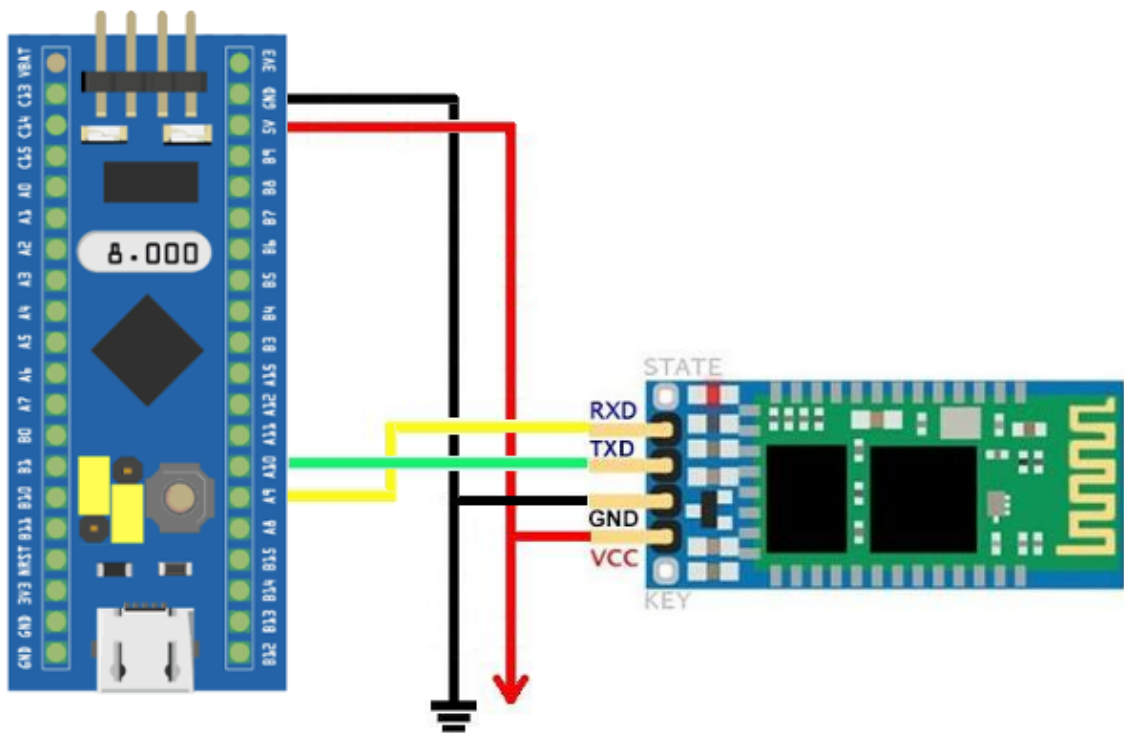


Figure 10: HC-05 bluetooth module with STM32

*\* Make sure to check the voltage output at the TX and RX to make sure the signal does not damage the module.*

## 7.2 Software configuration

- Configure the UART peripheral on the STM32 microcontroller to communicate with the HC-05 module. Set the baud rate, data bits, stop bits, and parity according to the HC-05's specifications(9600 Bits/s).
- Initialize the UART RX interrupt to process the data when sent, rather than waiting for it continuously.

USART1 Mode and Configuration

Mode

Mode

Hardware Flow Control (RS232)

Configuration

☒ NVIC Settings ☒ DMA Settings ☒ GPIO Settings

☒ Parameter Settings ☒ User Constants

Configure the below parameters :

Basic Parameters

Baud Rate	9600 Bits/s
Word Length	8 Bits (including Parity)
Parity	None
Stop Bits	1

Advanced Parameters

Data Direction	Receive and Transmit
Over Sampling	16 Samples

Figure 11: HC-05 configuration

## 8. ILI9341 TFT display

### 8.1 Introduction

The ILI9341 is a popular Thin-Film Transistor (TFT) Liquid Crystal Display (LCD) driver integrated circuit (IC) used in various 2.8-inch display modules. These modules are known for their affordability, ease of use, and good balance of features, making them a popular choice for hobbyists and makers in various projects.

Feature of ILI9341 TFT display :

- **Resolution:** 240 x 320 pixels, offering a decent amount of detail for basic graphics and text displays.
- **Interface:** Typically uses an SPI (Serial Peripheral Interface) for communication with microcontrollers, simplifying connection and control.
- **Size:** 2.8-inch diagonal, making it compact and suitable for portable devices or small projects.
- **Power Supply Voltage (VDD):** While the ILI9341 was originally designed for a 2.8V power supply, it can operate with a wider range of 1.65V to 3.3V. This flexibility allows for easier integration with various microcontrollers and development boards that typically operate at 3.3V.



Figure 12: ILI9341 TFT Display



## 8.2 Circuit Diagram

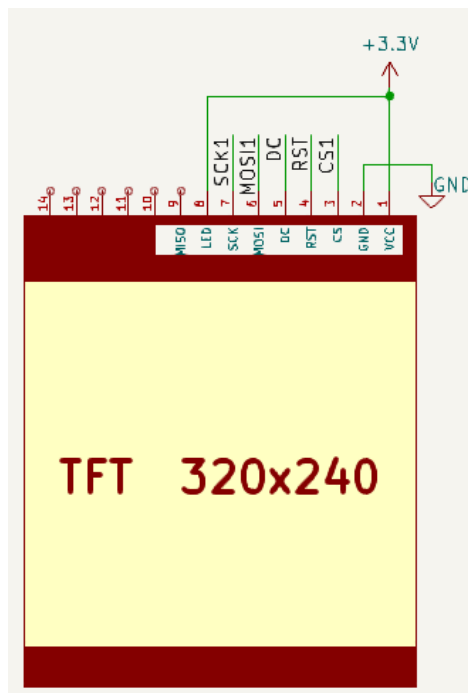


Figure 13: TFT Circuit

The ILI9341 display typically uses a 4-wire SPI (Serial Peripheral Interface) for communication with microcontrollers. Here's a breakdown of the common pinout found on most ILI9341 display modules:

- **Chip Select(CS):** Active low signal to select the ILI9341 for communication.
- **Reset signal(RST):** Active low signal to reset the ILI9341.
- **Data/Command Select(DC):** High for command mode, low for data mode.
- **Master Out Slave In(MOSI):** Sends data from the microcontroller to the ILI9341.
- **Master In, Slave Out(MISO):** Optional, used for SPI read operations (not always implemented).
- **Serial Clock(SCK):** Provides the clock signal for SPI communication.
- **Backlight control(LED):** High for backlight on, low for backlight off (may vary depending on module).

## 9. Working

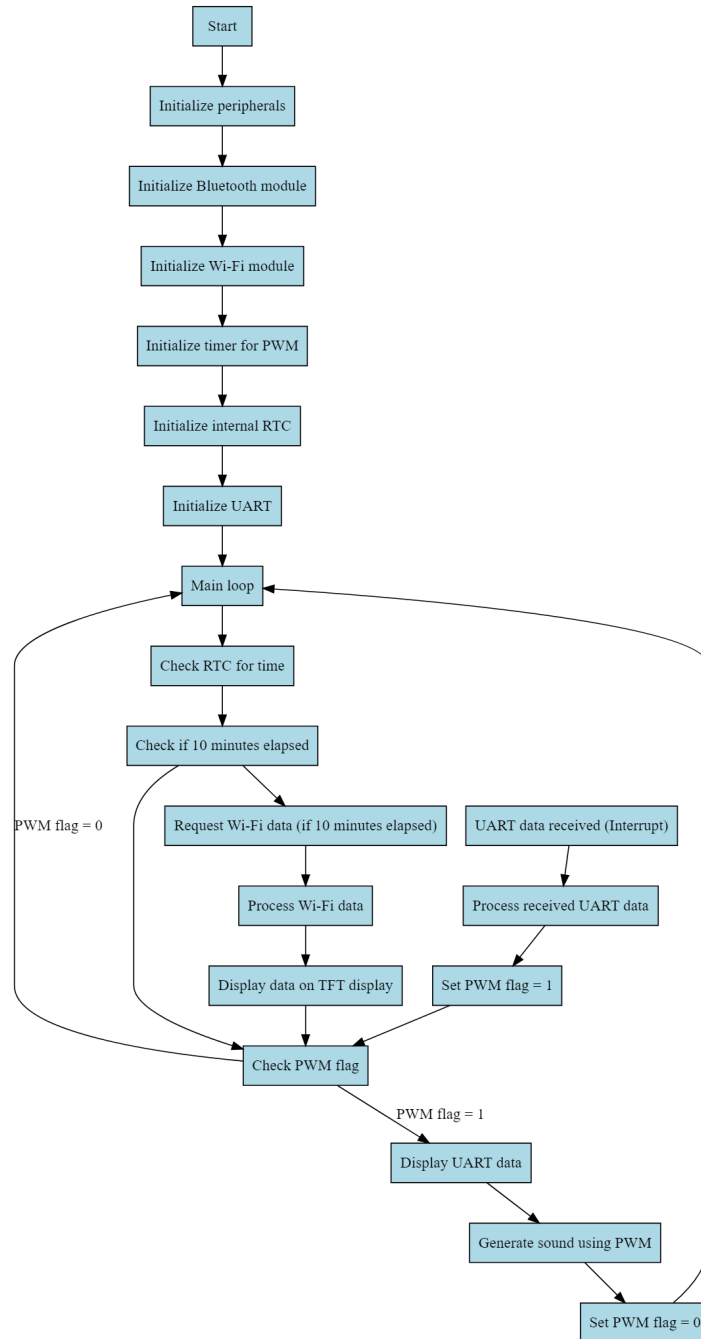


Figure 14: Flowchart of code

## 10. Bill of Material

## 11. End Result

