Building a Binary Transmission Device with STM32F401RE

Ishan Leung

September 2024

Abstract

This article describes the design, construction, and operation of a device that transmits a binary sequence of the form `111XXXXXXXX000` using an STM32F401RE microcontroller, a LoRa radio module, buttons, and an LCD for user interaction. The article covers the necessary hardware setup, mathematical background using Laplace and Fourier transforms, encoding and decoding techniques, circuit diagrams, and the source code for both encoding and decoding the sequence.

Contents

1 Introduction

This project aims to build a device capable of transmitting a binary sequence of the form `111XXXXXXXX000`, where the 8-bit user-defined segment can be altered using buttons. The device utilizes the STM32F401RE microcontroller, a LoRa radio module for transmission, and an LCD to display the current sequence.

2 Materials

- STM32F401RE microcontroller
- Breadboard
- LCD (16x2)
- LoRa radio module (e.g., SX1276)
- 3 push buttons (for '1', '0', and 'Send')
- Slide switch (Send/Receive mode)
- Resistors and capacitors for circuit setup
- Jumper wires
- Power supply (USB or battery pack)

3 Mathematical Background

3.1 Transmission Formula and Encoding with Example Encoding $\mathbf{211}_{10}(111XXXXXXXX000, XXXXXXXX = 11010011_2)$

The device encodes a sequence of bits into a time-domain signal that is modulated onto a carrier frequency (915 MHz). The transmission signal $S(f)$ is defined by the following formula:

$$S(f) = \frac{1}{2}\left[X_{total}(f - f_c) + X_{total}(f + f_c)\right] \tag{1}$$

where:

- $f_c = 915 \times 10^6\,\text{Hz}$ is the carrier frequency.
- $X_{total}(f)$ is the Fourier transform of the bit sequence, represented as:

$$X_{total}(f) = \sum_{n=0}^{13} a_n \cdot T \cdot \text{sinc}(fT) \cdot e^{-j2\pi f n T} \tag{2}$$

- a_n is the amplitude of each bit (1 for '1', 0 for '0').
- $T = 1\,\text{ms}$ (or $T = 1 \times 10^{-3}\,\text{s}$) is the duration of each bit.

3.1.1 Step 1: Fourier Transform of Each Bit

For each bit n, calculate $X_n(f)$ using the detailed formula:

- For Bit 0 ($n = 0$, $a_0 = 1$):

$$X_0(f) = 1 \cdot T \cdot \text{sinc}(fT) \cdot e^{-j2\pi f \cdot 0 \cdot T} = T \cdot \text{sinc}(fT) \tag{3}$$

- For Bit 1 ($n = 1$, $a_1 = 1$):

$$X_1(f) = 1 \cdot T \cdot \text{sinc}(fT) \cdot e^{-j2\pi f \cdot 1 \cdot T} \tag{4}$$

- For Bit 2 ($n = 2$, $a_2 = 1$):

$$X_2(f) = 1 \cdot T \cdot \text{sinc}(fT) \cdot e^{-j2\pi f \cdot 2 \cdot T} \tag{5}$$

- For Bit 3 ($n = 3$, $a_3 = 1$):
$$X_3(f) = 1 \cdot T \cdot \text{sinc}(fT) \cdot e^{-j2\pi f \cdot 3 \cdot T} \tag{6}$$
- For Bit 4 ($n = 4$, $a_4 = 1$):
$$X_4(f) = 1 \cdot T \cdot \text{sinc}(fT) \cdot e^{-j2\pi f \cdot 4 \cdot T} \tag{7}$$
- For Bit 5 ($n = 5$, $a_5 = 0$):
$$X_5(f) = 0 \cdot T \cdot \text{sinc}(fT) \cdot e^{-j2\pi f \cdot 5 \cdot T} = 0 \tag{8}$$
- For Bit 6 ($n = 6$, $a_6 = 1$):
$$X_6(f) = 1 \cdot T \cdot \text{sinc}(fT) \cdot e^{-j2\pi f \cdot 6 \cdot T} \tag{9}$$
- For Bit 7 ($n = 7$, $a_7 = 0$):
$$X_7(f) = 0 \cdot T \cdot \text{sinc}(fT) \cdot e^{-j2\pi f \cdot 7 \cdot T} = 0 \tag{10}$$
- For Bit 8 ($n = 8$, $a_8 = 0$):
$$X_8(f) = 0 \cdot T \cdot \text{sinc}(fT) \cdot e^{-j2\pi f \cdot 8 \cdot T} = 0 \tag{11}$$
- For Bit 9 ($n = 9$, $a_9 = 1$):
$$X_9(f) = 1 \cdot T \cdot \text{sinc}(fT) \cdot e^{-j2\pi f \cdot 9 \cdot T} \tag{12}$$
- For Bit 10 ($n = 10$, $a_{10} = 1$):
$$X_{10}(f) = 1 \cdot T \cdot \text{sinc}(fT) \cdot e^{-j2\pi f \cdot 10 \cdot T} \tag{13}$$
- For Bit 11 ($n = 11$, $a_{11} = 0$):
$$X_{11}(f) = 0 \cdot T \cdot \text{sinc}(fT) \cdot e^{-j2\pi f \cdot 11 \cdot T} = 0 \tag{14}$$
- For Bit 12 ($n = 12$, $a_{12} = 0$):
$$X_{12}(f) = 0 \cdot T \cdot \text{sinc}(fT) \cdot e^{-j2\pi f \cdot 12 \cdot T} = 0 \tag{15}$$
- For Bit 13 ($n = 13$, $a_{13} = 0$):
$$X_{13}(f) = 0 \cdot T \cdot \text{sinc}(fT) \cdot e^{-j2\pi f \cdot 13 \cdot T} = 0 \tag{16}$$

3.1.2 Step 2: Summing All Contributions

The total frequency representation $X_{total}(f)$ is the sum of all individual bit contributions:

$$X_{total}(f) = X_0(f) + X_1(f) + X_2(f) + X_3(f) + X_4(f) + X_6(f) + X_9(f) + X_{10}(f) \tag{17}$$

Bits 5, 7, 8, 11, 12, and 13 contribute 0 because they are zero-valued bits.

3.1.3 Step 3: Modulating Onto Carrier

To shift the total signal around the carrier frequency f_c, the modulated signal $S(f)$ is calculated as:

$$S(f) = \frac{1}{2}\left[X_{total}(f - f_c) + X_{total}(f + f_c)\right] \tag{18}$$

3.2 Decoding Formula and Filtering

After receiving the modulated signal, it is demodulated and passed through a Low-Pass Filter (LPF) to isolate the baseband signal. The RC filter function is:

$$H(s) = \frac{1}{1 + sRC} \tag{19}$$

where:

- $s = j\omega$ is the complex frequency variable.
- $R = 796\,\Omega$.
- $C = 0.1\,\mu\text{F} = 0.1 \times 10^{-6}\,\text{F}$.

Laplace Transform Detailed Calculation

3.2.1 Step 1: Calculating Cutoff Frequency

$$f_{cu} = \frac{1}{2\pi RC} = \frac{1}{2\pi \times 796 \times 0.1 \times 10^{-6}} \approx 2\,\text{kHz} \tag{20}$$

3.2.2 Step 2: Applying the Filter to the Demodulated Signal

Given the demodulated signal:

$$R_{demod}(s) = \frac{1}{2}X(s) + \frac{1}{2}X(s - 4j\pi f_{cu}) \tag{21}$$

Filtering it:

$$V_{out}(s) = H(s) \cdot R_{demod}(s) = \frac{1}{1 + sRC} \cdot \left(\frac{1}{2}X(s) + \frac{1}{2}X(s - 4j\pi f_{cu})\right) \tag{22}$$

Since $s = 4j\pi f_{cu}$ is much higher than the cutoff frequency, the high-frequency term is attenuated, leaving:

$$V_{out}(s) \approx \frac{1}{2}X(s) \tag{23}$$

4 Decoding the Received Signal

4.1 Mathematical Decoding Process

After receiving and filtering the signal, the remaining baseband signal $v_{out}(t)$ is sampled to determine each bit's value.

4.1.1 Step 1: Sampling and Bit Detection

The filtered output $v_{out}(t)$ is sampled at intervals corresponding to each bit's duration T. The sampled values are:

- **Sample at $t = 0$**:
$$v_{out}(0) = \frac{1}{2} \cdot 1 = 0.5 \tag{24}$$
- **Sample at $t = T$**:
$$v_{out}(T) = \frac{1}{2} \cdot 1 = 0.5 \tag{25}$$
- **Sample at $t = 2T$**:
$$v_{out}(2T) = \frac{1}{2} \cdot 1 = 0.5 \tag{26}$$
- **Sample at $t = 3T$**:
$$v_{out}(3T) = \frac{1}{2} \cdot 1 = 0.5 \tag{27}$$
- **Sample at $t = 4T$**:
$$v_{out}(4T) = \frac{1}{2} \cdot 1 = 0.5 \tag{28}$$
- **Sample at $t = 5T$**:
$$v_{out}(5T) = \frac{1}{2} \cdot 0 = 0 \tag{29}$$
- **Sample at $t = 6T$**:
$$v_{out}(6T) = \frac{1}{2} \cdot 1 = 0.5 \tag{30}$$
- **Sample at $t = 7T$**:
$$v_{out}(7T) = \frac{1}{2} \cdot 0 = 0 \tag{31}$$

- **Sample at $t = 8T$**:
$$v_{out}(8T) = \frac{1}{2} \cdot 0 = 0 \tag{32}$$

- **Sample at $t = 9T$**:
$$v_{out}(9T) = \frac{1}{2} \cdot 1 = 0.5 \tag{33}$$

- **Sample at $t = 10T$**:
$$v_{out}(10T) = \frac{1}{2} \cdot 1 = 0.5 \tag{34}$$

- **Sample at $t = 11T$**:
$$v_{out}(11T) = \frac{1}{2} \cdot 0 = 0 \tag{35}$$

- **Sample at $t = 12T$**:
$$v_{out}(12T) = \frac{1}{2} \cdot 0 = 0 \tag{36}$$

- **Sample at $t = 13T$**:
$$v_{out}(13T) = \frac{1}{2} \cdot 0 = 0 \tag{37}$$

4.1.2 Step 2: Threshold Detection

Each sampled value $v_{out}(nT)$ is compared against a threshold (e.g., 0.25) to determine the bit:

- If $v_{out}(nT) > 0.25$, then the bit is detected as '1'.
- If $v_{out}(nT) \leq 0.25$, then the bit is detected as '0'.

For the given sequence, the detected bits are:

- Bits 0-4: $v_{out} \approx 0.5$, so detected as '1'.
- Bit 5: $v_{out} = 0$, so detected as '0'.
- Bit 6: $v_{out} \approx 0.5$, so detected as '1'.
- Bits 7-8: $v_{out} = 0$, so detected as '0'.
- Bits 9-10: $v_{out} \approx 0.5$, so detected as '1'.
- Bits 11-13: $v_{out} = 0$, so detected as '0'.

4.2 Decoding in Code

The following code demonstrates how the STM32 decodes the received signal based on the slide switch position (Send/Receive mode):

Listing 1: STM32 Decoding Code

```
#include <SPI.h>
#include <RadioLib.h>

// Define LoRa module settings
SX1276 lora = new Module(10, 2, 9, 3);  // NSS, DIO0, RESET, DIO1 pins

void setup() {
    Serial.begin(9600);
    lora.begin(915.0); // Set frequency to 915 MHz
    lora.setSpreadingFactor(7);
    lora.setBandwidth(125.0);
    lora.setCoding
    lora.setCodingRate(5);
```

```
    lora.setOutputPower(17);
}

void loop() {
    String receivedData;
    int state = lora.receive(receivedData); // Receive the transmitted data
    if(state == RADIOLIB_ERR_NONE) {
        Serial.print("Received: ");
        Serial.println(receivedData); // Print the decoded sequence
    }
}
```

4.3 Expected Output at Each Step

- **Step 1:** The received signal is demodulated to bring it back to baseband.
- **Step 2:** The signal is filtered with the LPF, removing high-frequency components.
- **Step 3:** The filtered signal is sampled at each bit duration and compared against a threshold to determine the bit values.
- **Step 4:** The detected bits are combined to reconstruct the original transmitted sequence.

5 Circuit Diagrams

5.1 Transmitter and Receiver Circuit

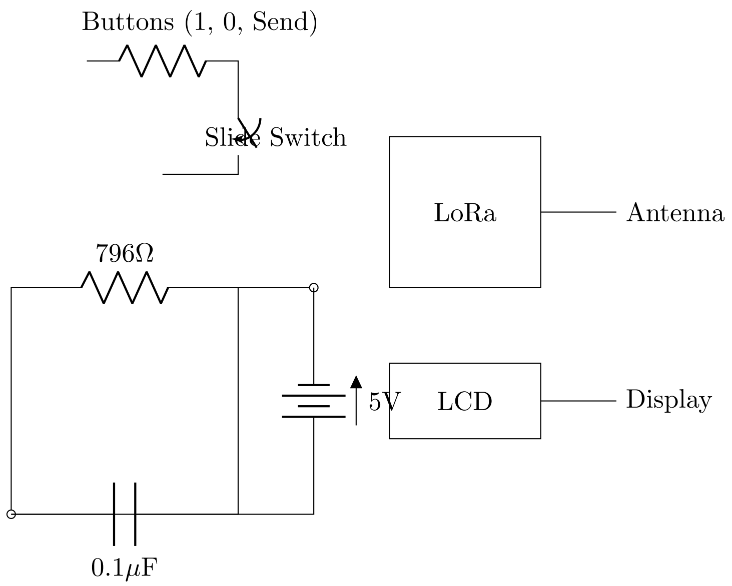

Figure 1: Circuit for Transmitter and Receiver with RC Low-Pass Filter connected to STM32 and LoRa module.