Design and Implementation of a Hardware-Based Wireless Communication System with Error Detection and Correction

Ufuk Can YİĞİT

20050211021
Ankara Yıldırım Beyazıt University
Department of Electrical and Electronics Engineering
Ankara , Turkey
20050211021@aybu.edu.tr

Abstract

In this project, it is aimed to implement wireless distance measurement and error correction algorithm using 2 esp32 microcontrollers. The distance value received by the transmitter with HC-SR04 is separated into 2 parts as full and decimal parts and coded with hamming (15,11) coding method. The received data is transmitted to the receiver side with ESP-NOW protocol. On the receiver side, random bit errors are applied to this data and errors are detected and corrected with hamming analysis method. Hamming; detects 2 bit errors, corrects 1 bit errors. This theory has worked exactly 100% in our system. In the serial monitor; incoming original distance, coded distance, erroneous distance, resolved distance, how many bit errors were corrected, final distance data are seen for both packets. This project is an example of ensuring data integrity during wireless communication and detecting and correcting errors that occur.

Introduction

In wireless communication systems, due to environmental interference and signal weakening, data transmitted may be corrupted. These errors pose a major problem in terms of sensor reliability, especially in embedded systems. Error detection and correction methods are used to eliminate or minimize this problem. One of these methods, hamming (15,11), is a cheap and practical method capable of correcting single-bit errors and detecting 2-bit errors.

In this project, an esp32-based wireless communication system was designed, the distance data received from the hc-sr04 sensor by the transmitter was transferred to the receiver via the ESP-NOW protocol, artificial errors were added to this data on the receiver side, and it was decoded with hamming decode to correct 1-bit errors and detect 2-bit errors. Thus, the reliability of the system against data corruption that may occur during wireless communication was tested.

The main purpose of the project is to demonstrate how the hamming code works in practice and to concretely demonstrate the error correction application in embedded systems such as esp32.

Methodology

In this project, a wireless data communication system was designed using 2 esp32 and 1 hc-sr04 sensors. The hc-sr04 sensor was used on the transmitter side. The data received here was obtained in float form. This data was divided into 2 parts because the hamming(15,11) code can only encode 11-bit data.

•Data[0]: Integer part (for example 32)
•Data[1]: Decimal part (for example 0.74 --> 74)

Two separate hamming (15,11) were applied to this data and two 15-bit error-protected data were obtained. This data was sent to the receiver side with the ESP-NOW protocol.

A random 1 or 2 bit error was added to the data received on the receiver side due to simulation. Then, hamming decode was applied to both data packets. As a result, 1-bit errors were detected and corrected, 2-bit errors could only be detected. The 2 decoded data pieces were recombined to obtain a float value. Then, the original distance, coded distance, erroneous distance, decoded distance, final distance (after decoding) were printed on the serial monitor to be presented to the user.

This methodology exemplifies the practical use of the hamming code in communication systems.

System design and flow diagram

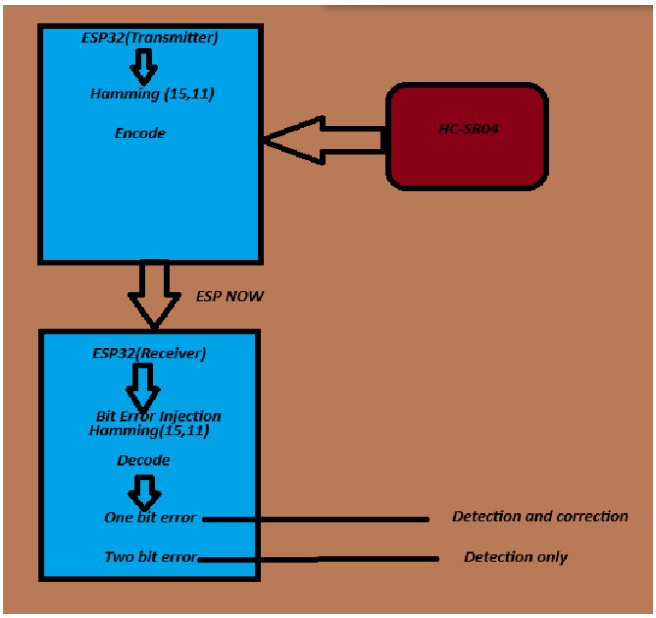

Figure1

The system flow diagram is shown in Figure 1.

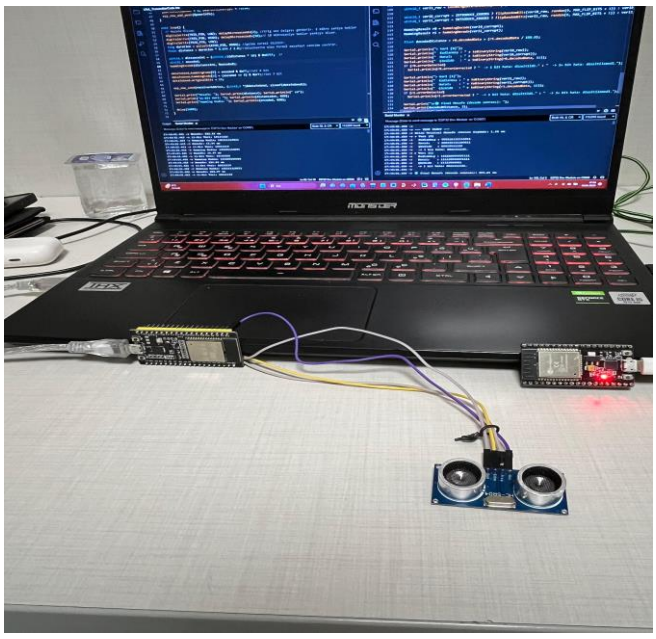

Figure2

The hardware design of the system is also shown in figure 2.

Tests and Results

We tested our system by uploading system codes via the Arduino IDE program..

The transmitter and receiver esp32 codes are given in the appendix link.

Transmitter Code :

This code displays the value received from the HC-SR04 distance sensor:

•It makes it error-proof by coding it with the Hamming (15,11) algorithm.

•It sends it to the receiver ESP32 wirelessly with ESP-NOW.

•It shows the distance, 11-bit data and Hamming code on the serial monitör

Receiver Code :

•Receives the encoded code (15,11) coming from the transmitter

•Applies random bit flip

Performs Hamming analysis (decode)

•If there is an error;

-Detects and corrects 1 bit error

-Detects only 2 bit error

•Recreates the original data

•Writes all the details to the serial monitör

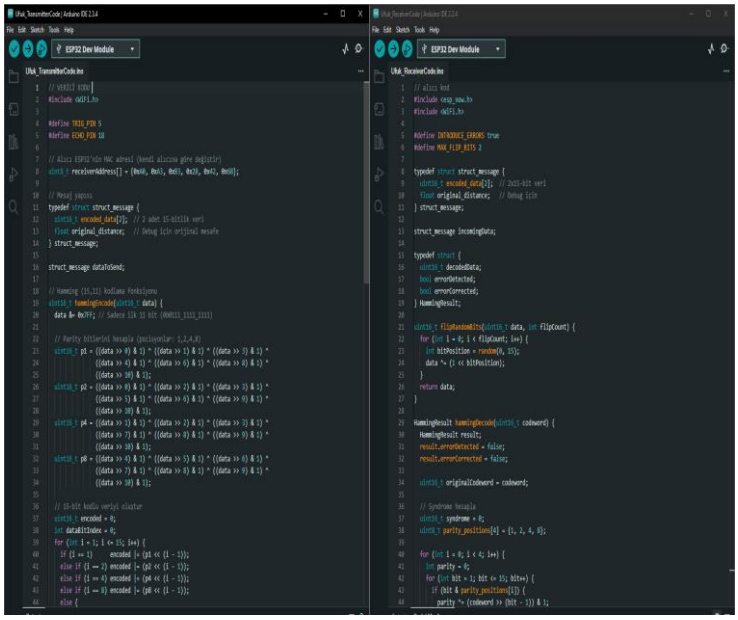

Figure3

Test :

```
18:27:16.156 -> Mesafe: 5.49 cm gönderildi.
18:27:16.156 -> Gönderim durumu: Başarılı
```

Figure4(Transmitter Serial Monitor)

```
18:27:16.156 ->
18:27:16.156 -> --- YENİ PAKET ---
18:27:16.156 -> Gelen Orijinal Mesafe (verici ölçümü): 5.49 cm
18:27:16.156 -> > Veri [0]
18:27:16.156 ->   Kodlanmış : 000000000101101
18:27:16.156 ->   Hatalı    : 000000000101100
18:27:16.200 ->   Çözüldü   : 00000000101
18:27:16.200 ->   -> 1 bit hata: düzeltildi.
18:27:16.200 -> > Veri [1]
18:27:16.200 ->   Kodlanmış : 000001100000100
18:27:16.200 ->   Hatalı    : 000001100000100
18:27:16.200 ->   Çözüldü   : 00000110001
18:27:16.200 ->
18:27:16.200 -> ● Final Mesafe (decode sonrası): 5.49 cm
18:27:18.175 ->
```

Figure5(Receiver Serial Monitor)

A 1 bit error was detected in the first packet sent and this was corrected with the help of hamming code. No error was detected in the second sent data. Although the received data was corrupted, it was corrected correctly and the final distance was calculated correctly.

```
18:27:18.142 -> Mesafe: 4.54 cm gönderildi.
18:27:18.175 -> Gönderim durumu: Başarılı
```

Figure6(Transmitter Serial Monitor)

```
18:27:18.175 -> --- YENİ PAKET ---
18:27:18.175 -> Gelen Orijinal Mesafe (verici ölçümü): 4.54 cm
18:27:18.175 -> > Veri [0]
18:27:18.175 ->   Kodlanmış : 000000000101010
18:27:18.175 ->   Hatalı    : 000000000101010
18:27:18.175 ->   Çözüldü   : 00000000100
18:27:18.175 -> > Veri [1]
18:27:18.175 ->   Kodlanmış : 000001100101110
18:27:18.175 ->   Hatalı    : 000101100101111
18:27:18.175 ->   Çözüldü   : 00110110101
18:27:18.175 ->   -> 1 bit hata: düzeltildi.
18:27:18.175 ->
18:27:18.175 -> ● Final Mesafe (decode sonrası): 8.37 cm
18:27:20.178 ->
```

Figure7(Receiver Serial Monitor)

Since there was no error in the first data sent, the error could not be detected. In the second data sent, a 2-bit error was detected. However, since hamming can only correct 1-bit errors, it detected the error but could not correct it. The final distance could not be calculated correctly. This is an expected situation in hamming.

Conclusion

In this project, it has been successfully tested to detect and correct errors that may occur during wireless communication using ESP32 microcontrollers. The data received by the transmitter ESP32 via HC-SR04 was transferred to the receiver side by applying (15,11) hamming encode. On the receiver side, it was aimed to add artificial error to this data with bit flip and then decode the data and analyze it with hamming(15,11). As seen above, these tests were successfully completed.

As a result of the project, data security was increased with this low-cost and low-resource system; it was shown that error correction codes can be applied effectively on embedded systems. This system can then be combined with other error detection and correction methods and integrated into more complex systems.

Appendix

Source codes :

https://github.com/ufukcanyqt/Hamming-Code-With-ESP32

References :

https://www.geeksforgeeks.org/hamming-code-in-computer-network/

https://www.tutorialspoint.com/error-correcting-codes-hamming-codes

https://howtomechatronics.com/tutorials/arduino/ultrasonic-sensor-hc-sr04/