NEC Protocol Implementation using the SLG47011
SLG47011

This application note describes how to configure the SLG47011 to act as control logic for infrared signal transmitters using the NEC IR communication protocol.

Complete design files for this project can be found in the References section.

Contents

Terms and Definitions **2**

1. Introduction **2**

2. Circuit Connection **2**

3. NEC Infrared Transmission Protocol **3**

4. NEC Protocol Design **5**

5. NEC Protocol Control Logic Demonstration Board **7**

6. Test Results **10**

7. Conclusion **14**

8. Revision History **15**

References

For related documents and software, please visit:

AnalogPAK | Renesas

Download our free Go Configure Software Hub [1] to open the design file [2] and view the proposed circuit design. Use the AnalogPAK development tools [3] to freeze the design into your own customized IC in a matter of minutes. Renesas Electronics provides a complete library of application notes [4] featuring design examples, as well as explanations of features and blocks within the Renesas IC.

[1] GreenPAK Go Configure Software Hub, Software Download and User Guide, Renesas Electronics

[2] AN-CM-418 NEC Protocol Implementation using the SLG47011 AnalogPAK Design File, Renesas Electronics

[3] GreenPAK Development Tools, GreenPAK Development Tools Webpage, Renesas Electronics

[4] GreenPAK Application Notes, GreenPAK Application Notes Webpage, Renesas Electronics

Author: Oleksandr Lykhanskyi

Originally published at Electronic Design

Terms and Definitions

CH	Channel
LSB	Least Significant Bit
LUT	Look-up Table
MSB	Most Significant Bit
MF	Multifunctional
NEC	Nippon Electric Company
NMOS	N-channel Metal-Oxide-Semiconductor
OD	Open-drain
IR	Infrared

1. Introduction

The NEC protocol is one of the most widely used IR communication protocols. It utilizes pulse distance encoding for data transmission.

This application note describes an implementation of the NEC protocol control logic using the Renesas SLG47011, which has the necessary resources to support this functionality.

In this design the NEC protocol command can be programmed according to customer needs using the Memory Table Configurator. The design supports up to 9 different commands which can then be implemented on an IR remote controller.

2. Circuit Connection

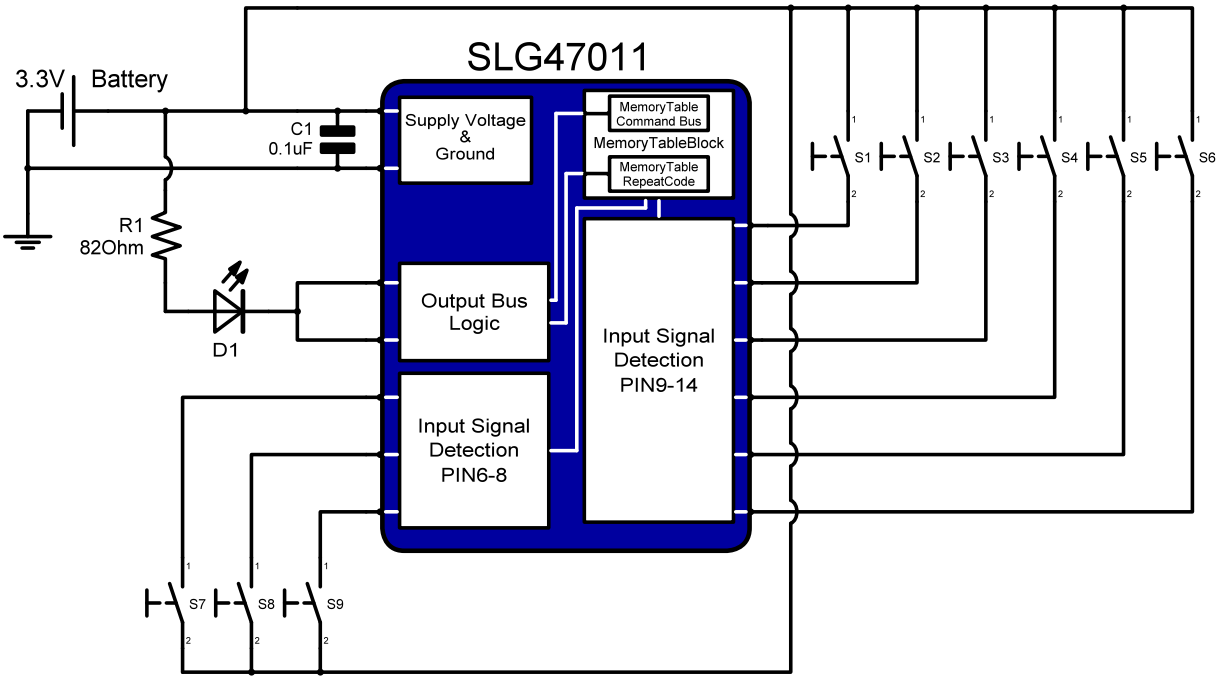

Figure 1. Circuit Connection

3. NEC Infrared Transmission Protocol

Since the NEC protocol utilizes pulse distance encoding, it's important to remember how logical bits are transmitted in order to program a custom message.

- Logical '0' – a 562.5 µs pulse burst followed by a 562.5 µs space. Total transmit time – 1.125 ms.
- Logical '1' – a 562.5 µs pulse burst followed by a 1.6875 ms space (3 x 562.5 µs intervals). Total transmit time – 2.25 ms.

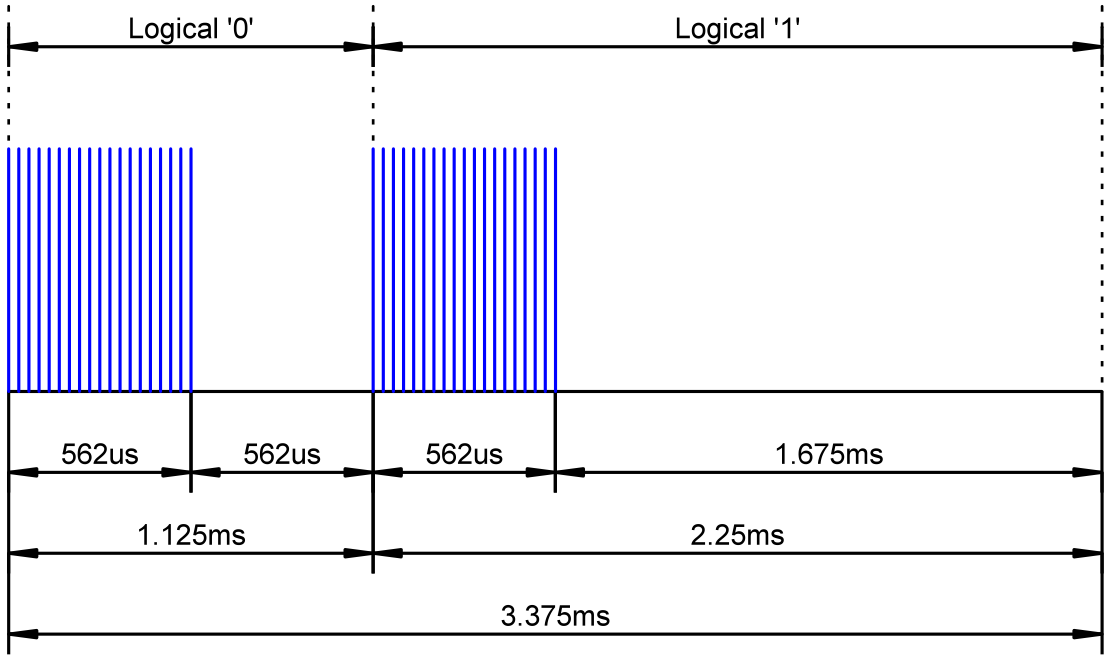

Figure 2. NEC Protocol Timing Diagram

Note: In this design, NMOS OD PINs 4-5 are connected in parallel to increase output signal current. As such, the waveforms will show you an inverted version of the output signal.

According to the NEC protocol specifications a standard carrier frequency of 38.222 kHz is used. In this design a frequency of 38.167 kHz was chosen as the closest match to the specifications.

The structure of the NEC protocol message is defined as follows:

1. 9 ms leading pulse burst (16 times the pulse burst length used for a logical data bit)
2. 4.5 ms space
3. 8-bit address for the receiving device
4. 8-bit logical inverse of the address
5. 8-bit command
6. 8-bit logical inverse of the command
7. Final 562.5 µs pulse burst to signify the end of message transmission.

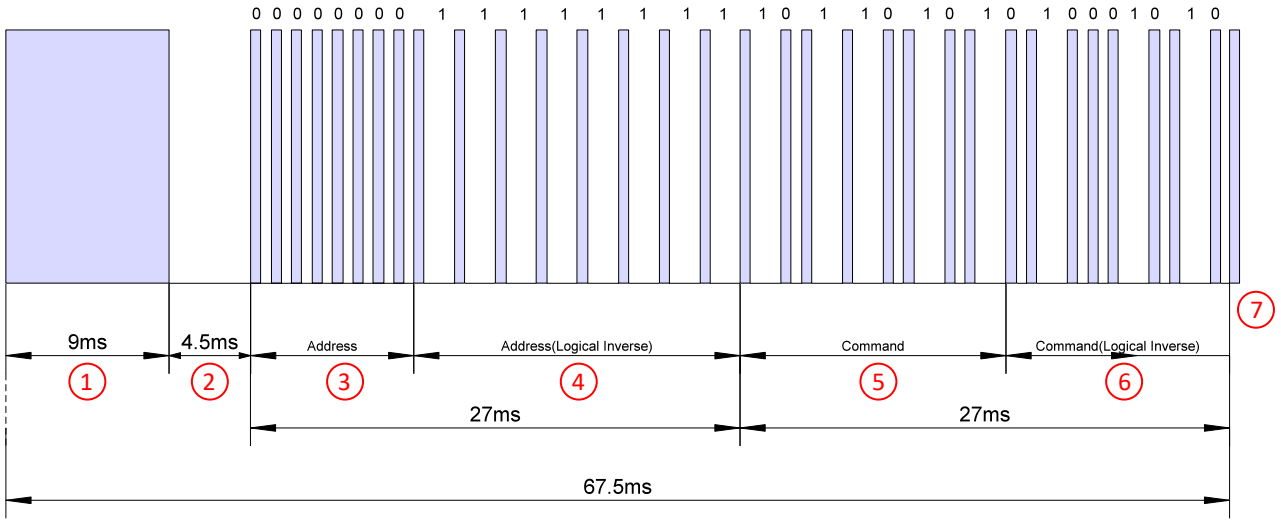

Figure 3. NEC Protocol Message Structure

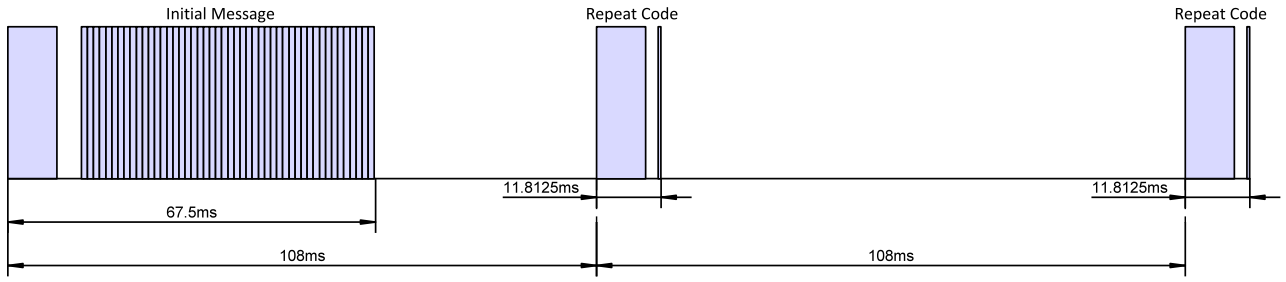

Figure 4. NEC Protocol Message sequence

Note: Although the NEC protocol has a standard message structure, the extended version of it is used quite often. In the extended version of the NEC protocol, to increase the range of possible addresses, instead of including the 8-bit logical inverse of the address the second frame represents a register address. This design is developed with the extended NEC protocol in mind.

4. NEC Protocol Design

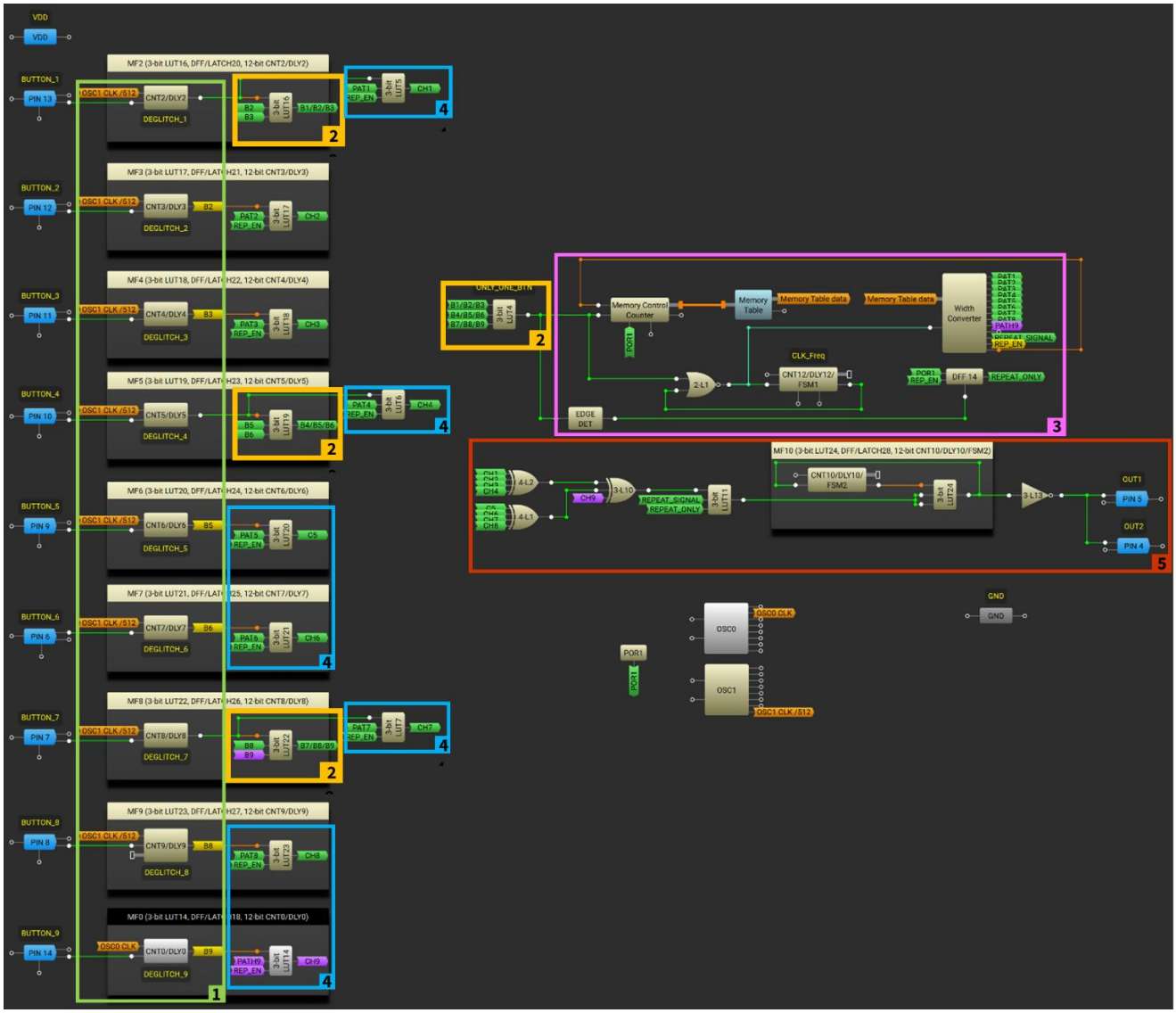

Figure 5. NEC Protocol GreenPAK Design

This design consists of five main phases:

1. Detection of the input signals. To ensure proper functionality of the design, each input signal is connected to a CNT/DLY block that serves as a deglitch. The deglitch delay time is configured to be 40 ms on a rising edge of the input signal.
2. During the second phase, the design checks whether only one input signal is at a high level. In this scenario, 3-bit LUT4 switches from its idle-state - high to low, which resets the Memory Control Counter.
3. The third phase of the design is responsible for the NEC protocol control logic. The Memory Table block paired with the Width Converter (in 12 to 12 mode) is used to manually set the desired commands (including RepeatCode).
   a. OUT0-OUT8 contain nine possible messages.
   b. OUT10-OUT11 contain the RepeatCode and the RepeatEnable signals respectively (RepeatEnable is a custom signal which switches from idle-state - low to high level - 40 ms after the end of a message).
   c. The Width Converter block utilizes an internal clock signal with a period which corresponds to 1.778 kHz to ensure that the length of each bit is 562.5 µs.

d. The DFF block is used as a one-bit memory cell. The DFF block resets every time a single input signal switches to a high-level and will pass the RepeatCode signal to the output bus only when the input signal is kept high for more than 108 ms.

Note: While setting up custom messages, it is important to remember the main operating principles of the Memory Table block. The Memory Table block can hold and send out up to 4096 12-bit words. Each word requires two bytes to hold, with the last four bits left unused. In the Memory Table's Manual Editor mode you can see that each 12-bit word is stored in parallel between twelve outputs (OUT0(LSB) - OUT11(MSB)) so that each OUT is responsible for holding one of the 12 bits of that word.

For example: To receive a message 10011100 (LSB) on OUT0, the following steps should be taken:

1. Split the desired message into two chunks.
2. Swap them.
3. Mirror each of those chunks horizontally.
4. Write down the hex value of the resulting combination in the Memory Table.

Alternatively, the Manual Editor can be used to set up the 1st through 8th word manually.

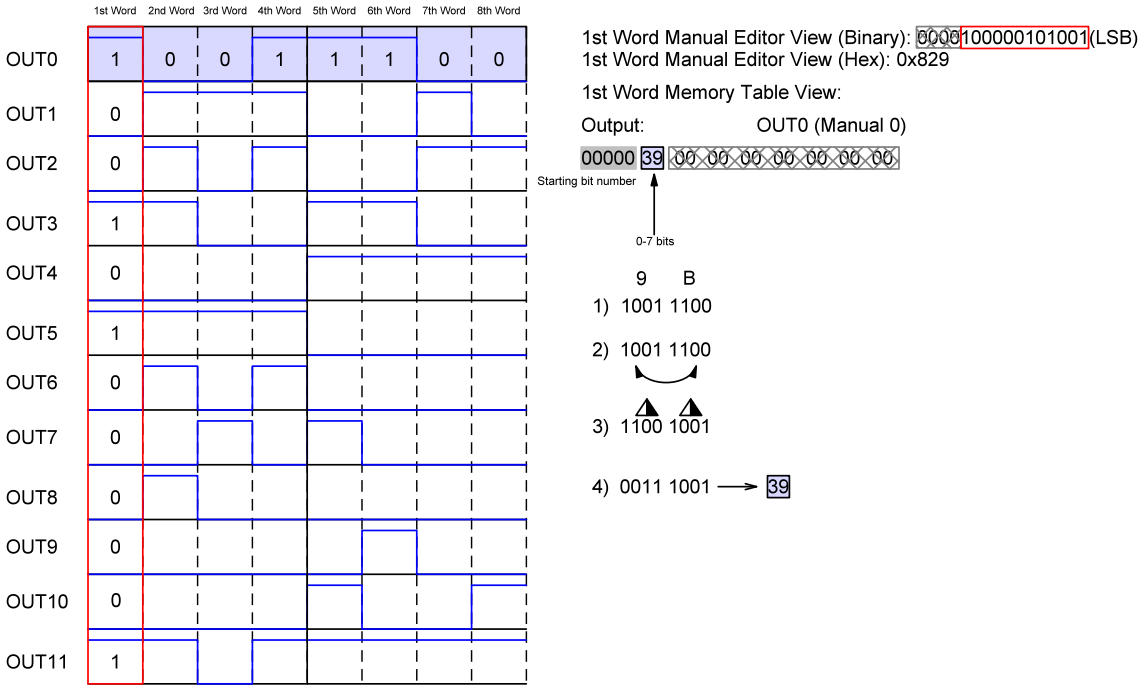

Figure 6. Using the Memory Table to setup custom messages

4. The fourth phase of the design serves as a 3-bit Multiplexer for each message on OUT0-OUT8 which allows the command to pass through only when the RepeatEnable signal is low.
5. The fifth phase of the design serves as the main BUS for all signals. Afterwards, the passing signal is modulated with a 38.222 kHz frequency generated by the internal clock (MF10 block).

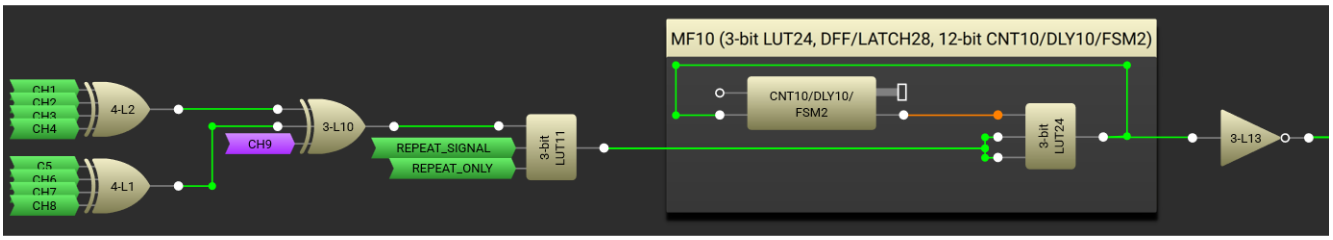

Figure 7. Last phase of the NEC Protocol design

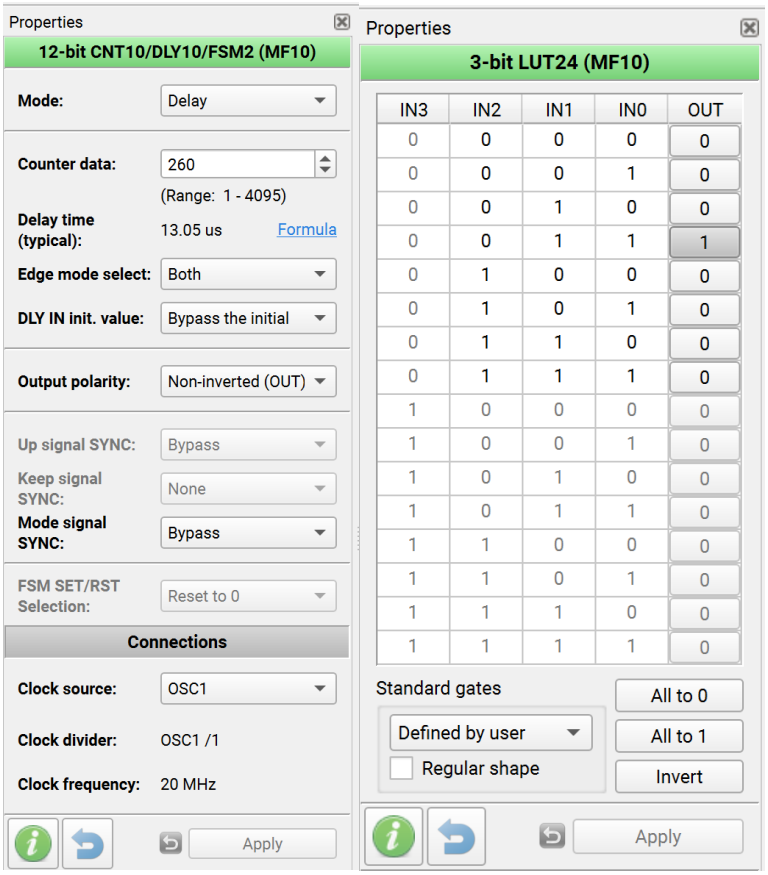

Figure 8. Configuration settings for the MF10 block

Note: As mentioned previously the output signal is passed through an inverter to drive the IR LED through the OD NMOS.

5. NEC Protocol Control Logic Demonstration Board

A demonstration board was created to test the viability of the design.

To decode the NEC protocol, an ArduinoUNO along with Keyestudio IR Receiver were used. Figure 10 shows these devices along with the connections used.

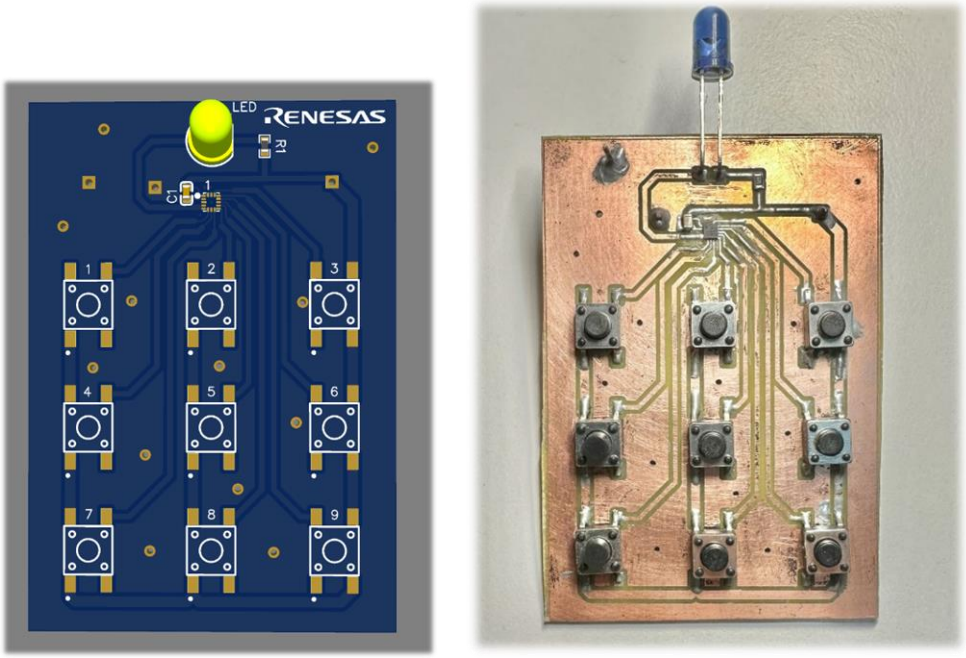

Figure 9. Test Demo Board

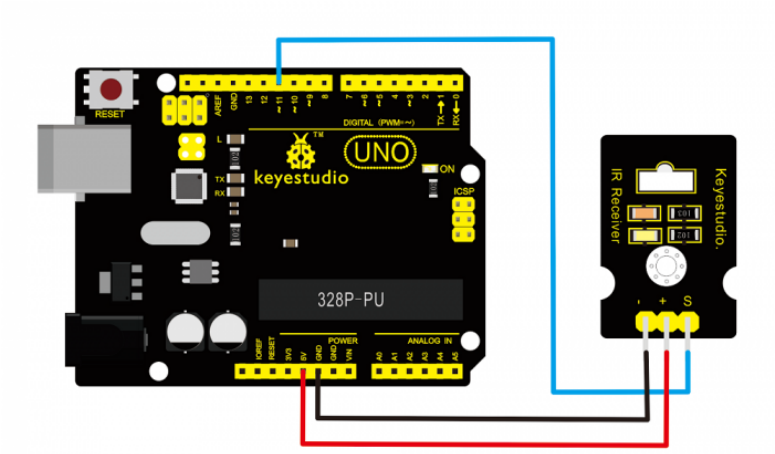

Figure 10. ArduinoUNO and Keyestudio IR Receiver connection schematic.

The following code was implemented on the Arduino Board.

Code:

```
#include <IRremote.h>

// You may change PIN number according to your needs.
const int IR_RECEIVE_PIN = 11;

void setup() {
  Serial.begin(9600);

  IrReceiver.begin(IR_RECEIVE_PIN, ENABLE_LED_FEEDBACK);
}
void loop() {

  if (IrReceiver.decode()) {

    Serial.println("========================================");
    Serial.print("Protocol: ");
    Serial.println(getProtocolString(IrReceiver.decodedIRData.protocol));

    Serial.print("Data: 0x");
    Serial.println(IrReceiver.decodedIRData.decodedRawData, HEX);

    if (IrReceiver.decodedIRData.flags & IRDATA_FLAGS_IS_REPEAT) {
      Serial.println("Repeat Code (Key is kept pressed)");
    }
    Serial.println("========================================");
    Serial.println();
    IrReceiver.resume();
  }
}
```

The following results were obtained:

```
========================================
Protocol: NEC
Data: 0x6F905583
========================================
========================================
Protocol: NEC
Data: 0x0
Repeat Code (Key is kept pressed)
========================================
```

The “Data” fully matches the programmed command through the MemoryTable block (LSB first).

6. Test Results

The design was tested in hardware and the following waveform results were obtained:

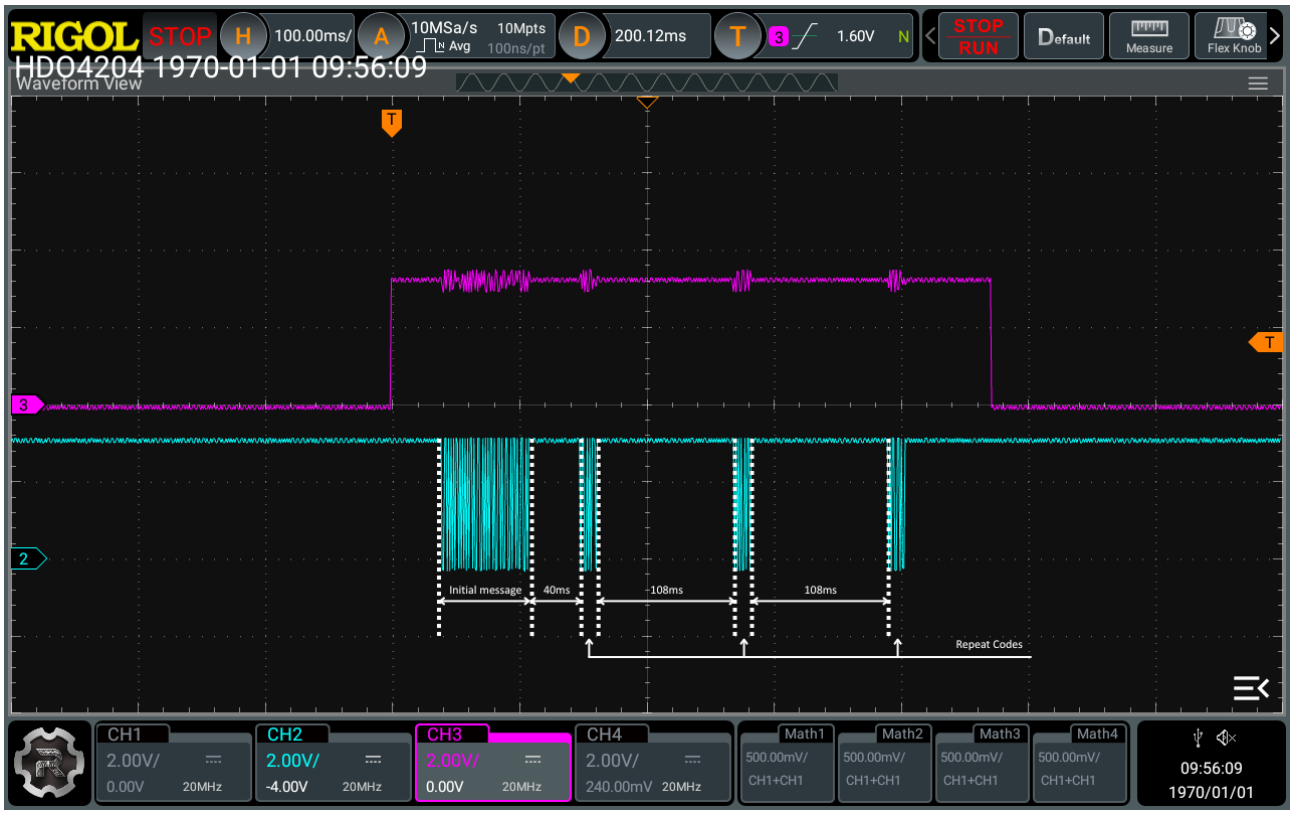

Figure 11. Modulated Output Signal

CH3 (pink) – active-high input signal **CH2 (blue) –** modulated output signal

The waveform on the Figure 1 demonstrates modulated output signal covering the initial message and repeat code obtained on the demonstration board.

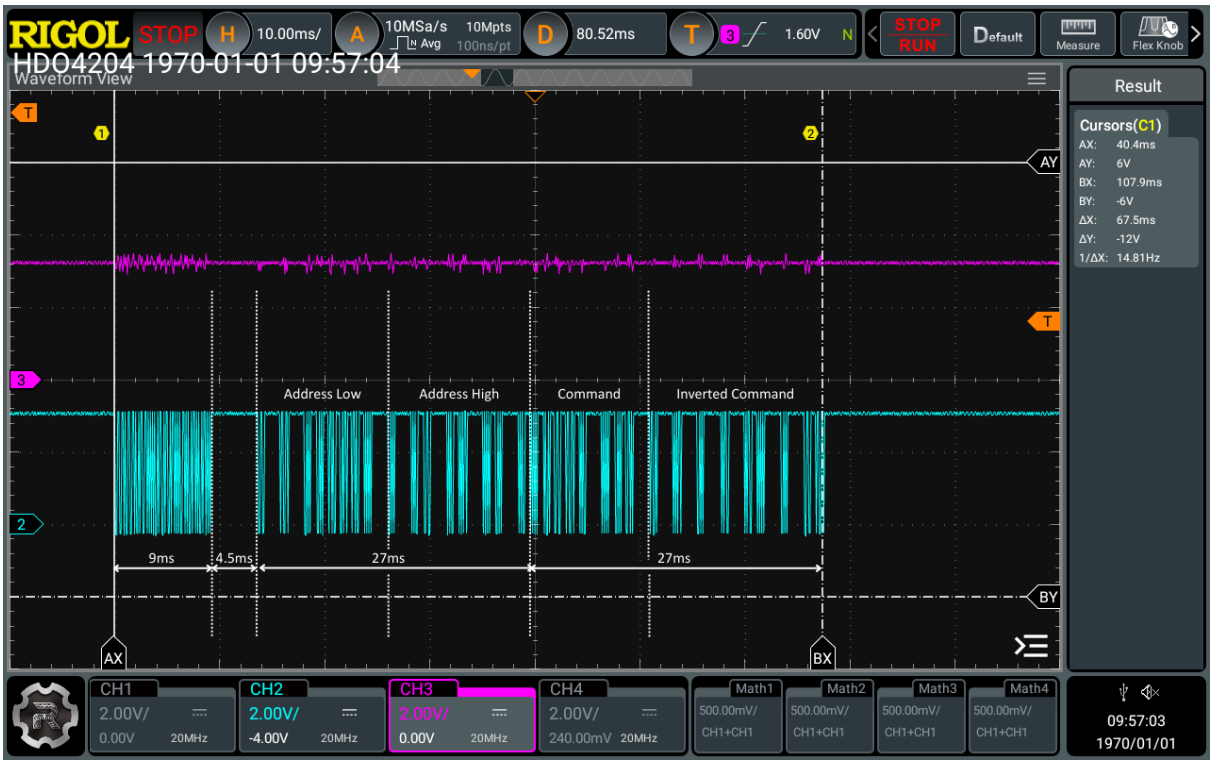

Figure 12. Initial Message

CH3 (pink) – active-high input signal **CH2 (blue) –** modulated output signal

The waveform on the Figure 2 demonstrates the initial message timeframe.

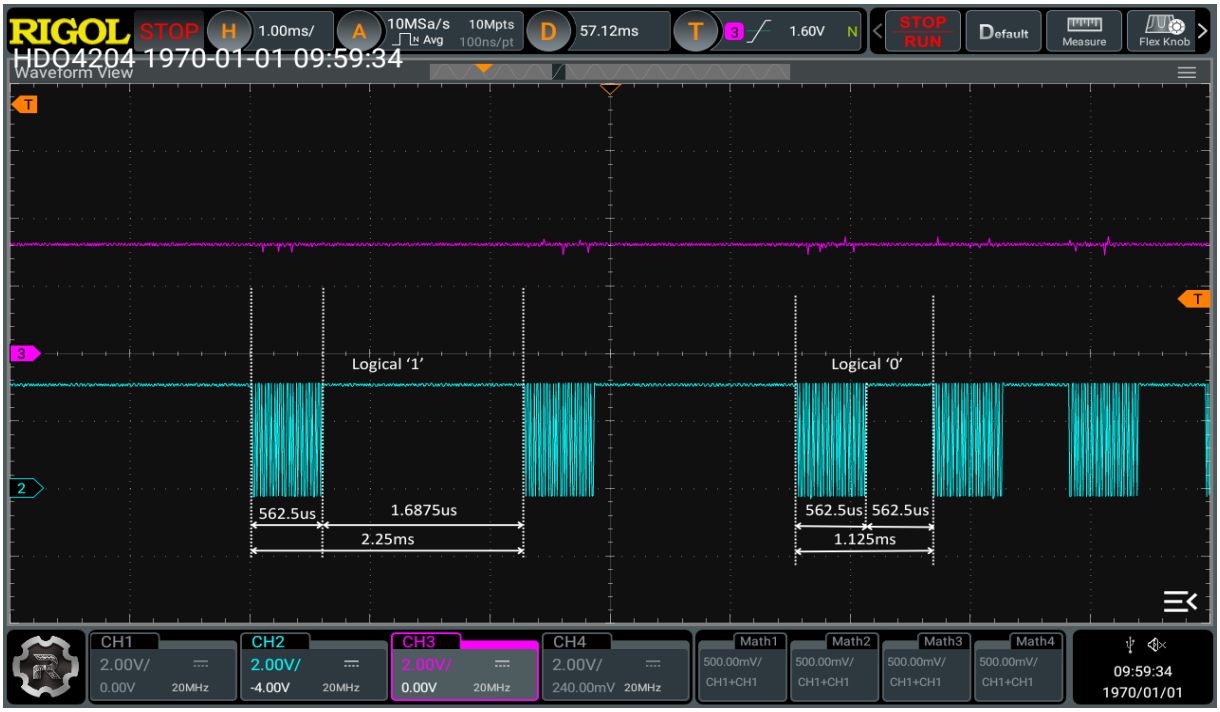

Figure 13. Logical '1' and '0'

CH3 (pink) – active-high input signal **CH2 (blue) –** modulated output signal

This waveform demonstrates logical '1' and '0' timeframes.

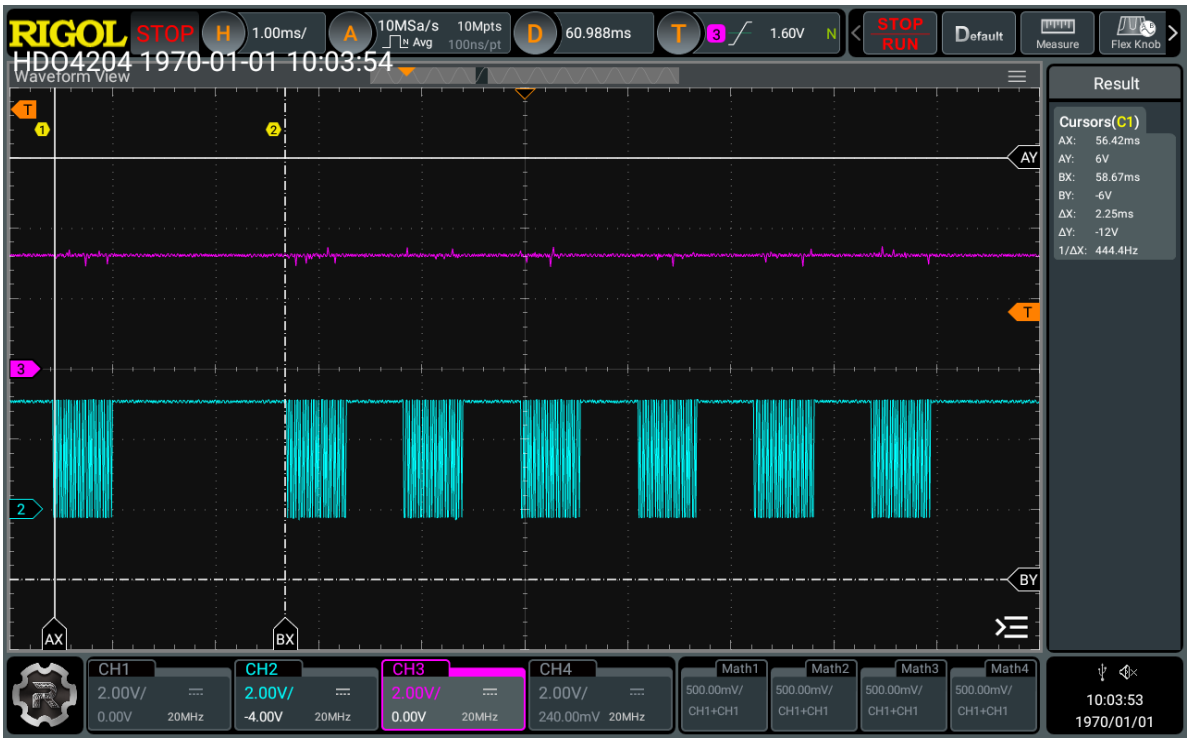

Figure 14. Logical ‘1’ Timeframe

CH3 (pink) – active-high input signal **CH2 (blue) –** modulated output signal

This waveform demonstrates logical ‘1’ timeframe. The measurements were carried out via cursor tool.

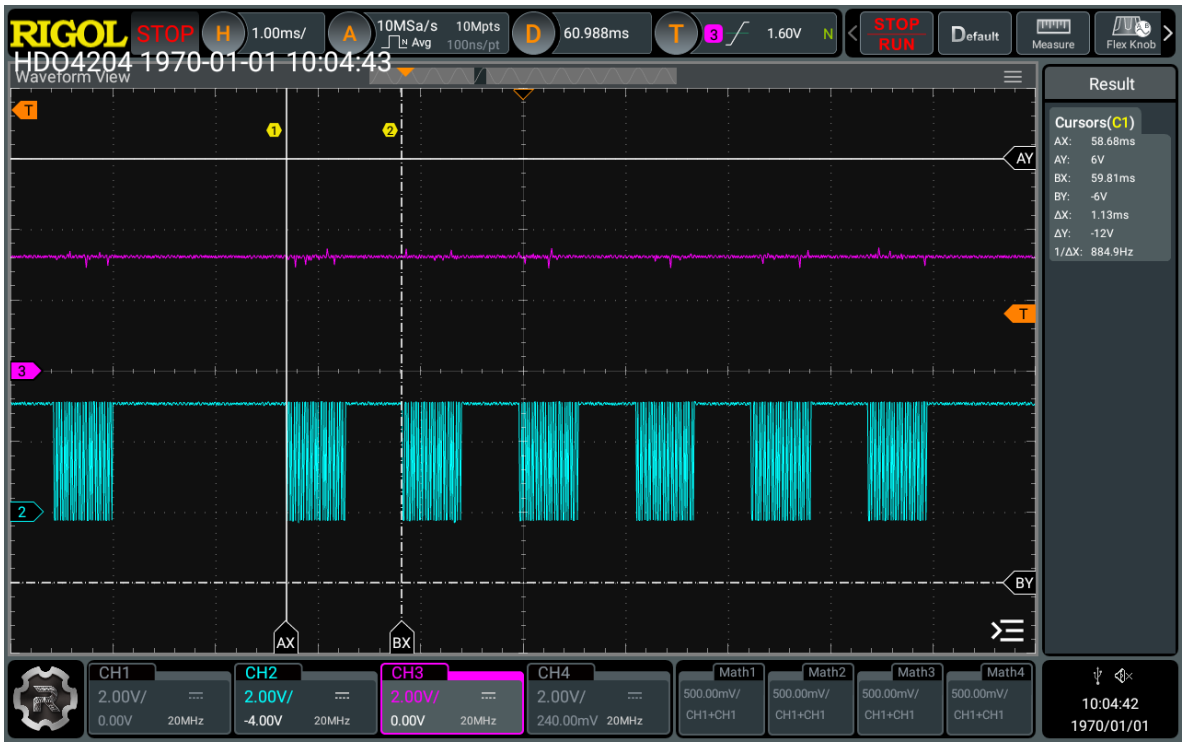

Figure 15. Logical ‘0’ Timeframe

CH3 (pink) – active-high input signal **CH2 (blue) –** modulated output signal

This waveform demonstrates logical ‘0’ timeframe. The measurements were carried out via cursor tool.

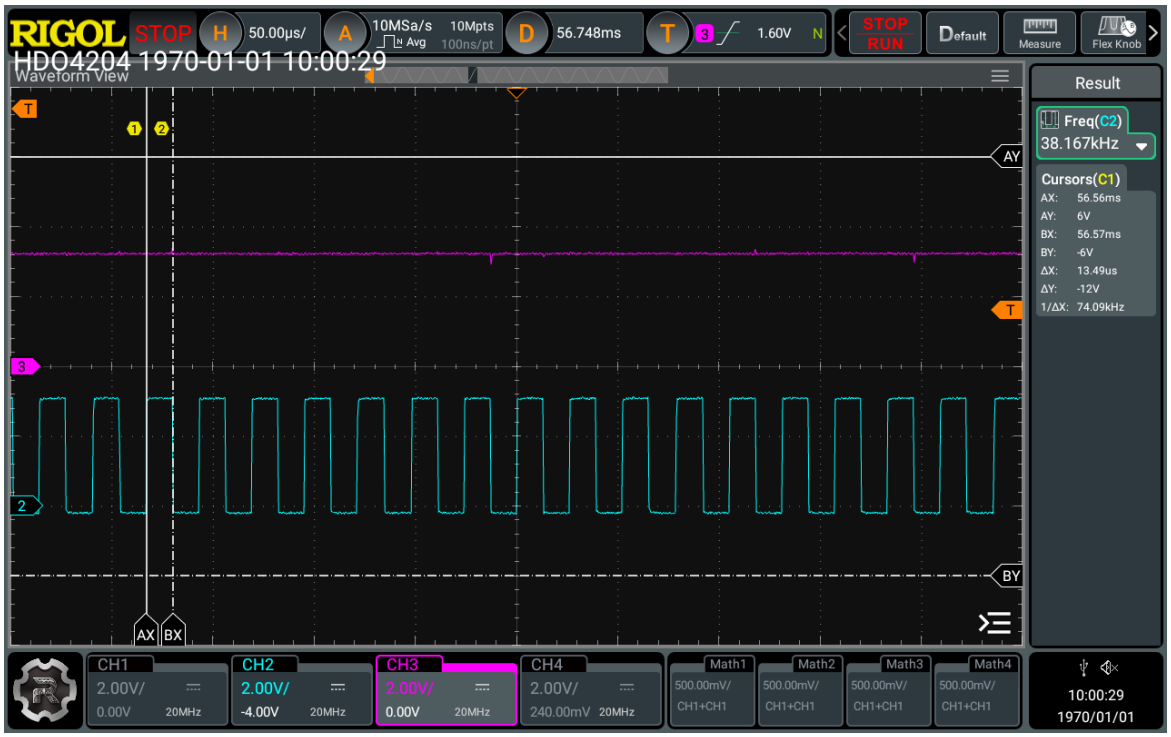

Figure 16. Carrier Frequency

CH3 (pink) – active-high input signal **CH2 (blue) –** modulated output signal

The waveform on Figure 6 demonstrates carrier frequency. The measurements were carried out via cursor and oscilloscope measure tools.

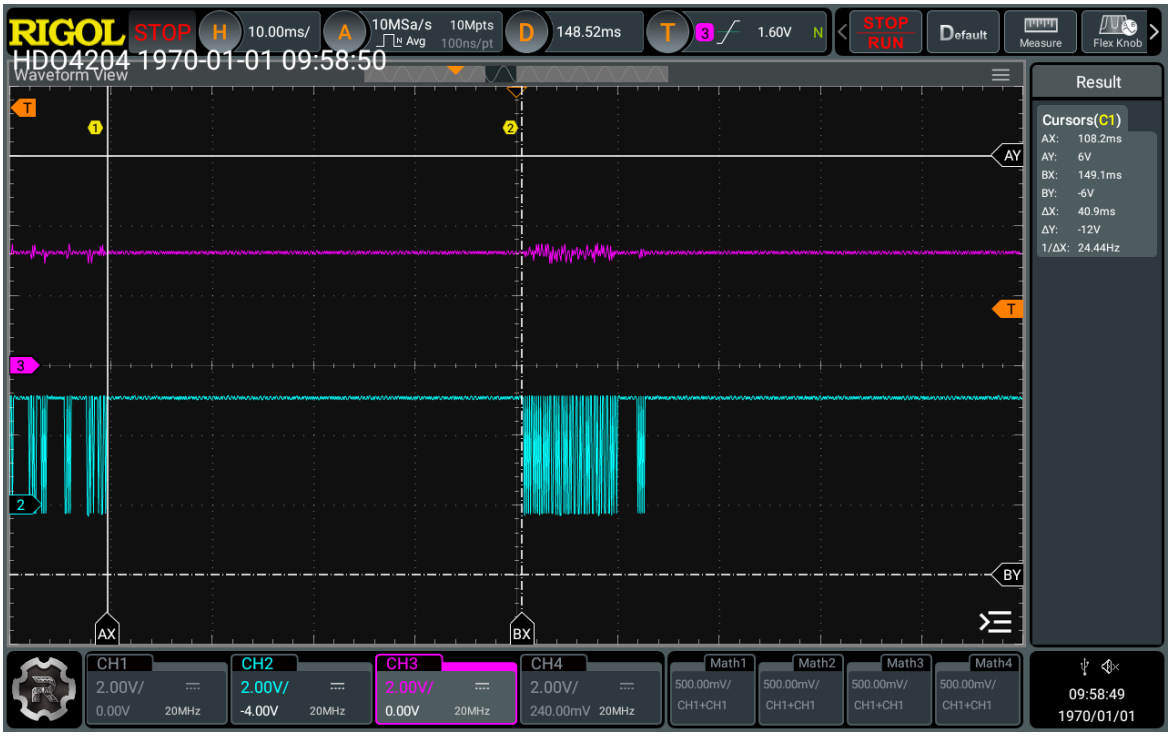

Figure 17. Repeat Code Timeframe

CH3 (pink) – active-high input signal **CH2 (blue) –** modulated output signal

This waveform demonstrates Repeat Code timeframe. The measurements were carried out via cursor tool.

7. Conclusion

From the waveforms, we can conclude that the design operates accordingly to the NEC protocol specifications. The design implemented using the Renesas SLG47011 is proven to be fully capable of transmitting messages to a receiver device via the NEC protocol. Viability of the design was tested on the Arduino UNO platform along with Keyestudio IR Receiver.

8. Revision History

Revision	Date	Description
1.00	Dec 3, 2025	Initial release

IMPORTANT NOTICE AND DISCLAIMER

RENESAS ELECTRONICS CORPORATION AND ITS SUBSIDIARIES (“RENESAS”) PROVIDES TECHNICAL SPECIFICATIONS AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT OF THIRD-PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for developers who are designing with Renesas products. You are solely responsible for (1) selecting the appropriate products for your application, (2) designing, validating, and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. Renesas grants you permission to use these resources only to develop an application that uses Renesas products. Other reproduction or use of these resources is strictly prohibited. No license is granted to any other Renesas intellectual property or to any third-party intellectual property. Renesas disclaims responsibility for, and you will fully indemnify Renesas and its representatives against, any claims, damages, costs, losses, or liabilities arising from your use of these resources. Renesas' products are provided only subject to Renesas' Terms and Conditions of Sale or other applicable terms agreed to in writing. No use of any Renesas resources expands or otherwise alters any applicable warranties or warranty disclaimers for these products.

(Disclaimer Rev.1.01)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Contact Information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit www.renesas.com/contact-us/.

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.