ABSTRACT

We, Human Beings are very social creatures and communication is the most important aspect of our daily lives that we use to convey information, ideas and thoughts with each other. With the advent of digital electronics, a faster and more efficient communication is now possible. But this merit comes with an uncertainty of the received data being error-free. The communication channel in which the data gets transmitted in, is very much likely or always has a chance of inducing an error in the data being transmitted. This is because the channel can be affected by various factors such as noise, interference, attenuation, distortion, crosstalk, and jitter. These factors can cause errors in the message being transmitted. The goal of this project is not just transmitting and receiving the data, but to transmit the data in a way that, the person receiving this data at the receiving end is able to confirm its reliability. The aim is to bring about this reliable data transmission using an error detection technique known as Cyclic Redundancy Check or CRC for short. To sum it all up we will be using CRCs to detect any degree of error in a message signal with the ultimate goal of making digital communication always reliable.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF FLOWCHARTS

CHAPTER 1
INTRODUCTION

1.1 MOTIVATION

We were motivated to select an error detection-based project because we could apply the knowledge and skills we had acquired in our previous coursework and gain a deeper understanding of a fundamental concept that is essential to our future careers. We were eager to tackle a challenge that would help us acquire critical skills and knowledge, and were convinced that error detection was a field where we could make a meaningful contribution. Our motivation stemmed from our ambition to develop and learn as professionals, and we viewed this project as the ideal chance to accomplish our objectives.

1.2 PROBLEM STATEMENT

Communication is the most important aspect of our lives that we use on a daily basis to exchange information, ideas and thoughts with each other. With the advent of digital electronics, a faster and more efficient communication is now possible. However, this merit comes with an uncertainty in the received data being reliable or in other words being error free, as the communication channel in which the data gets transmitted in, is very much likely or always has a chance of inducing an error in the data being transmitted.

1.3 OBJECTIVE OF PROJECT

The objective of this project is not just transmitting and receiving the data, but to transmit the data in a way that, the person receiving this data at the receiving end is able to confirm its reliability. The aim is to bring about this reliable data transmission using an error detection technique known as Cyclic Redundancy Check or CRC for short which is useful in detecting any degree of error. The ultimate goal of this project is to make digital communication always reliable.

1.4 LITERATURE SURVEY

All error-detection and correction schemes add some redundancy (i.e., some extra data) to a message, which receivers can use to check consistency of the delivered message and to recover data that has been determined to be corrupted[1]. Error detection and correction schemes can be either systematic or non-systematic. In a systematic scheme, the transmitter sends the original (error-free) data and attaches a fixed number of check bits (or parity data), which are derived from the data bits by some encoding algorithm[2]. If error detection is required, a receiver can simply apply the same algorithm to the received data bits and compare its output with the received check bits; if the values do not match, an error has occurred at some point during the transmission. If error correction is required, a receiver can apply the decoding algorithm to the received data bits and the received check bits to recover the original error-free data. In a system that uses a non-systematic code, the original message is transformed into an encoded message carrying the same information and that has at least as many bits as the original message. Good error control performance requires the scheme to be selected based on the characteristics of the communication channel.

1.5 DESIGN APPROACH

Firmware Design & Development

- Requirement Gathering
- Software Design Document
- Firmware Design and Development
- Coding

Testing

- Module Testing (Hardware modules)
- Unit & Integration testing of the software
- System Testing

CHAPTER 2
HARDWARE DESIGN

2.1 BLOCK DIAGRAM

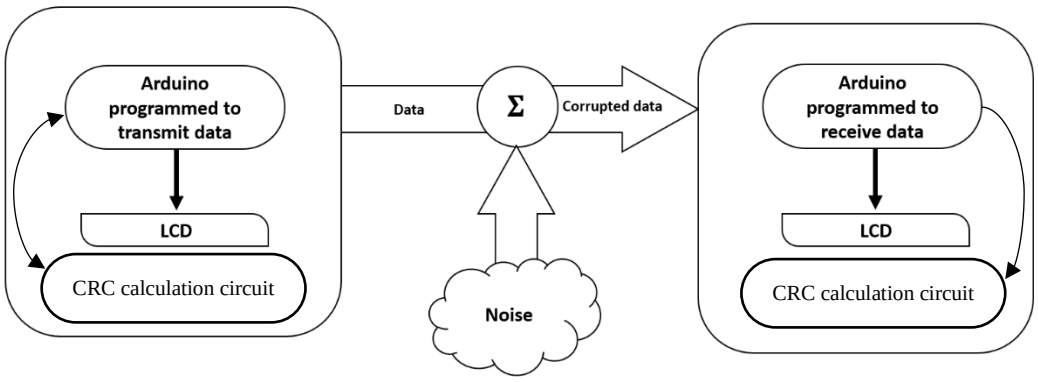

Figure 2.1: Block diagram

Fig 2.1. Shows the overall design consists of three important parts, i.e., Transmitter, Receiver and Channel. The transmitter is programmed to send the data along with the error detecting redundant code [3]. The receiver is programmed to receive and display the data with added circuitry of error detecting if found any. The channel consists of three lines, one for data being transmitted, another for common ground and the third is the clock to synchronise both sides. The channel consists of a break switch to induce some errors in the message being sent [4].

2.2 ARDUINO NANO

The Arduino Nano is an open-source breadboard-friendly microcontroller board based on the Microchip ATmega328P microcontroller (MCU) and developed by Arduino.cc and initially released in 2008. It offers the same connectivity and specs of the Arduino Uno board in a smaller form factor. Fig 2. shows the Arduino Nano is equipped with 30 male I/O headers, in a DIP-30-like configuration, which can be programmed using the Arduino Software Integrated Development Environment (IDE), which is common to all Arduino boards and running both online and offline.

2.2.1 TECHNICAL SPECIFICATIONS OF ARDUINO NANO

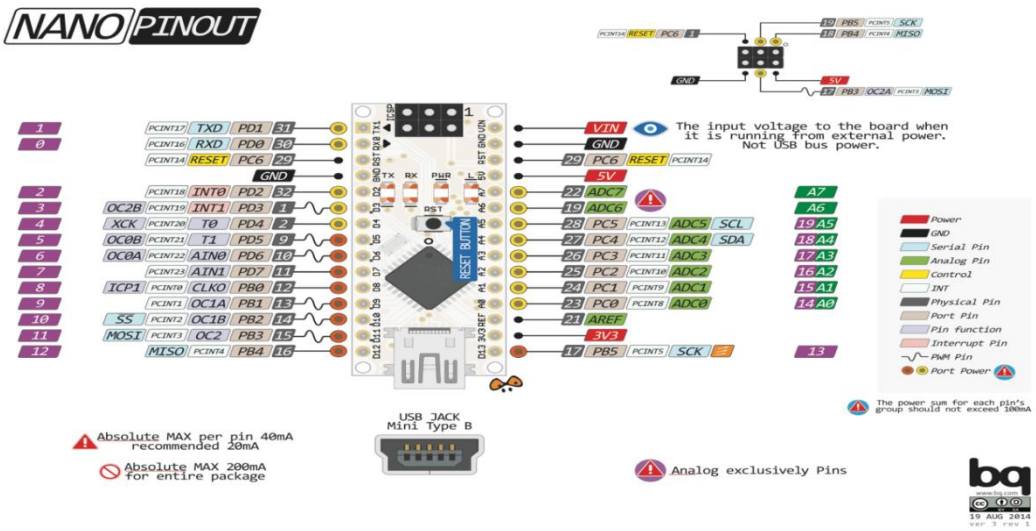

Figure 2.2.1: Arduino nano

Microcontroller:	Microchip ATmega328P[4]
Operating voltage:	5 volts
Input voltage:	5 to 20 volts
Digital I/O pins:	14 (6 optional PWM outputs)
Analog input pins:	8
DC per I/O pin:	40 mA
DC for 3.3 V pin:	50 mA
Flash memory:	32 KB, of which 2 KB is used by bootloader
SRAM:	2 KB
EEPROM:	1 KB
Clock speed:	16 MHz
Length:	45 mm
Width:	18 mm
Mass:	7 g
USB:	Mini-USB Type-B [5]
ICSP Header:	Yes
DC Power Jack:	No

Table 1: Arduino Nano specifications

2.3 LCD DISPLAY

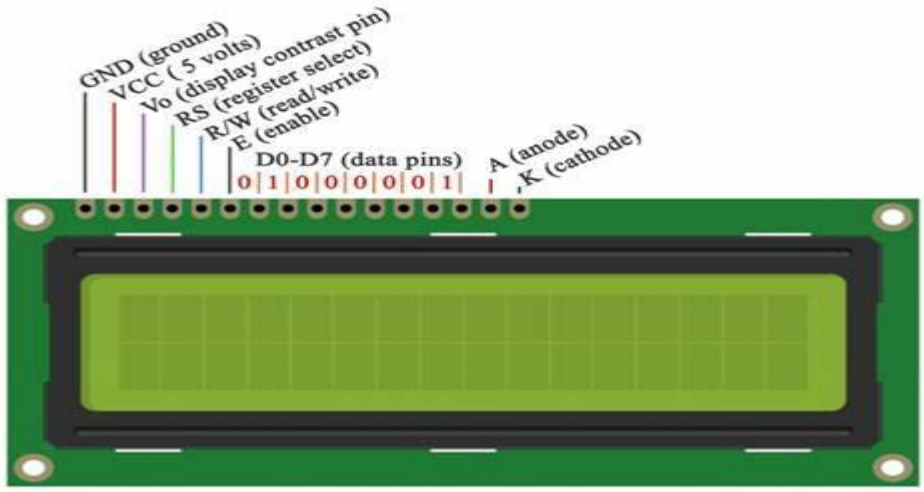

Figure 2.3.1: LCD display

2.3.1 LCD DISPLAY DETAILS

Fig 2.3.1. Shows the 16x2 Alphanumeric LCD can show 16 Columns and 2 Rows therefore a total of (16x2) 32 characters can be displayed. Each character can either be an alphabet or number or even a custom character. This particular LCD gas a green backlight, you can also get a Blue Backlight LCD to make your projects stand our and visually appealing, apart from the backlight colour both the LCD have the same specifications hence they can share the same circuit and code. If your projects require more characters to be displayed you can check the 20x4 Graphical LCD which has 20 Columns and 4 Rows and hence can display up to 80 characters.

2.3.2 SPECIFICATIONS OF 16X2 LCD DISPLAY

- Operating Voltage: 4.7V to 5.3V
- Operating Current 1mA (without backlight)
- Can display (16x2) 32 Alphanumeric Characters
- Custom Characters Support
- Works in both 8-bit and 4-bit Mode

2.4 74HC273 OCTAL D-TYPE FLIP-FLOP

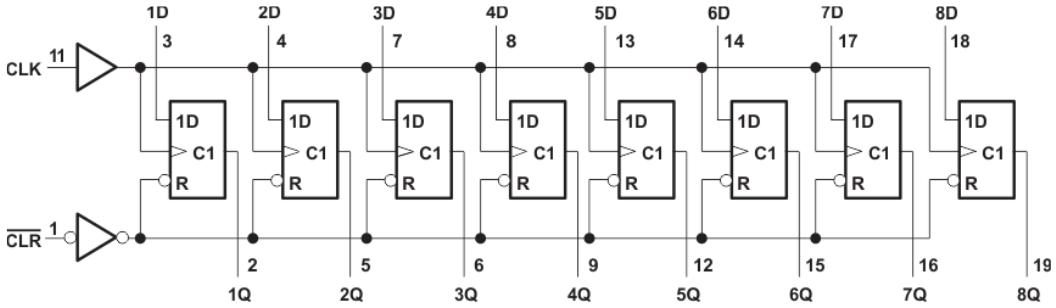

Figure 2.4: 74HC273 pin diagram

Fig 2.4. Is 74HC273 is an octal D-type flip-flop with reset. It has eight edge-triggered D-type flip-flops with individual D inputs and Q outputs. The common buffered clock (CLK) input loads all flip-flops simultaneously when the clock enable (CE) is low. The clear (CLR) input is active low and clears all flip-flops when it is held low. The 74HC273 is useful for applications where the true output only is required and the clock and master reset are common to all storage elements.

2.5 74HC86 QUAD 2-INPUT XOR

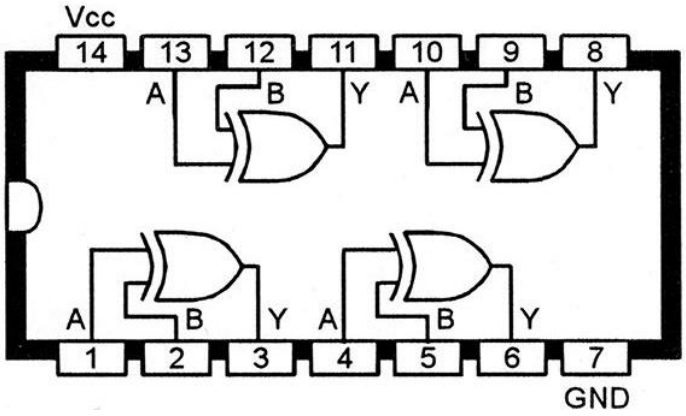

Figure 2.5: 74HC86 pin diagram

Fig 2.5. Shows that device contains four independent 2-input XOR gates. Each gate performs the Boolean function Y = A $\oplus$ B in positive logic

2.6 CIRCUIT DIAGRAMS

The circuit diagram consists of three main parts, namely the transmitter, receiver and the channel. Both the transmitter and the receiver have been provided with a CRC calculation circuit to verify the reliability of the received data shown in fig 6 .

2.6.1 TRANSMITTER AND RECEIVER

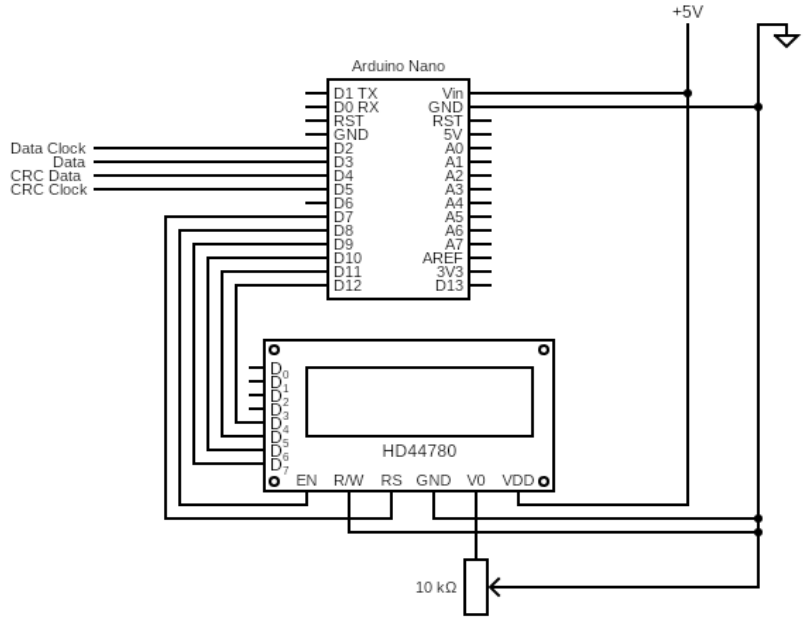

Figure 2.6.1: Transmitter and Receiver

2.6.2 CHANNEL

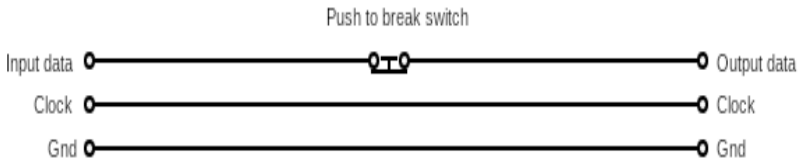

Figure 2.6.2: Channel

The channel is provided with a push to break switch that, upon pushing breaks the circuit which lets us induce error into the message to verify the working of our error detection circuit shown in fig 2.6.2

2.6.3 CRC CIRCUIT

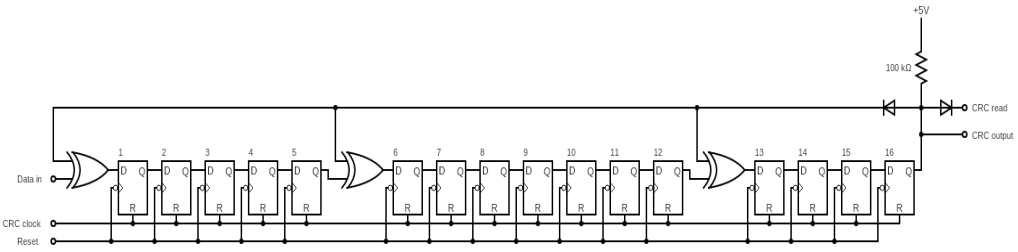

Figure 2.6.3: CRC circuit

- Fig 2.6.3. Shows CRC or Cyclic Redundancy Check is a method of detecting accidental changes/errors in the communication channel. CRC uses Generator Polynomial which is available on both sender and receiver side[5].
- An example generator polynomial is of the form like $x^3 + x + 1$. This generator polynomial represents key 1011. Another example is $x^2 + 1$ that represents key 101.
- n: Number of bits in data to be sent from sender side.
- k: Number of bits in the key obtained from generator polynomial[6].
- Sender Side (Generation of Encoded Data from Data and Generator Polynomial (or Key)): The binary data is first augmented by adding k-1 zeros in the end of the data Use modulo-2 binary division to divide binary data by the key and store remainder of division. Append the remainder at the end of the data to form the encoded data and send the same
- Receiver Side (Check if there are errors introduced in transmission) Perform modulo-2 division again and if the remainder is 0, then there are no errors.
- The key used for this project is famously known as CRC Modem which is a 16-bit CRC key and is of the form $x^{16} + x^{12} + x^5 + 1$.
- This generator polynomial can be expressed in hexadecimal in the form 0xd175.
- This generator polynomial has a hamming distance of 4.
- This hamming distance is valid for message length lower than 32751, this means that any 4-bit errors in this bit range can be detected with 100% certainty.
- The way this operates is that, the CRC circuit consists of D type flip-flops and XOR gates which simulate the division of the message signal by the generator polynomial.
- The output of this division can be visualised as a reminder[7].
- Once the crc of the original message is generated, it is read out from the d flip-flops by shifting it with the help of 16 zeroes and is transmitted by the transmitter along with the original message[8].
- This means that we are adding the original binary message and the reminder.
- The receiver end also has a similar crc calculation circuit with the generator polynomial known[9].
- When the receiver calculates CRC, it finds that the polynomial is completely divisible by the key if there is no error occurred in the message and if any of the flip-flops is non-zero then it means there is some error in the message.

CHAPTER 3

SOFTWARE DESIGN

3.1 HARDWARE RESOURCE ALLOCATION

Arduino Nano Pin Number	Description
D12	D4 pin of LCD
D11	D5 pin of LCD
D10	D6 pin of LCD
D9	D7 pin of LCD
D8	Enable pin of LCD
D7	Register select pin of LCD
D5	CRC clock output
D4	CRC data input
D3	Data pin
D2	Clock pin

Figure 3.1: Hardware resource allocation

3.2 PROGRAMMING LANGUAGE

- C/C++

3.3 DEVELOPMENT TOOLS

- Arduino SDK 1.6.9

3.4 ARDUINO IDE

Arduino software is needed to program Arduino boards and must be downloaded from the Arduino website and installed on a computer. This software is known as the Arduino IDE (Integrated Development Environment). Drivers must be installed in order to be able to program an Arduino from the Arduino IDE. After this mandatory procedure is done, code is written in C programming language and the code is dumped on to the Arduino Nano microcontroller.

3.5 FLOWCHART

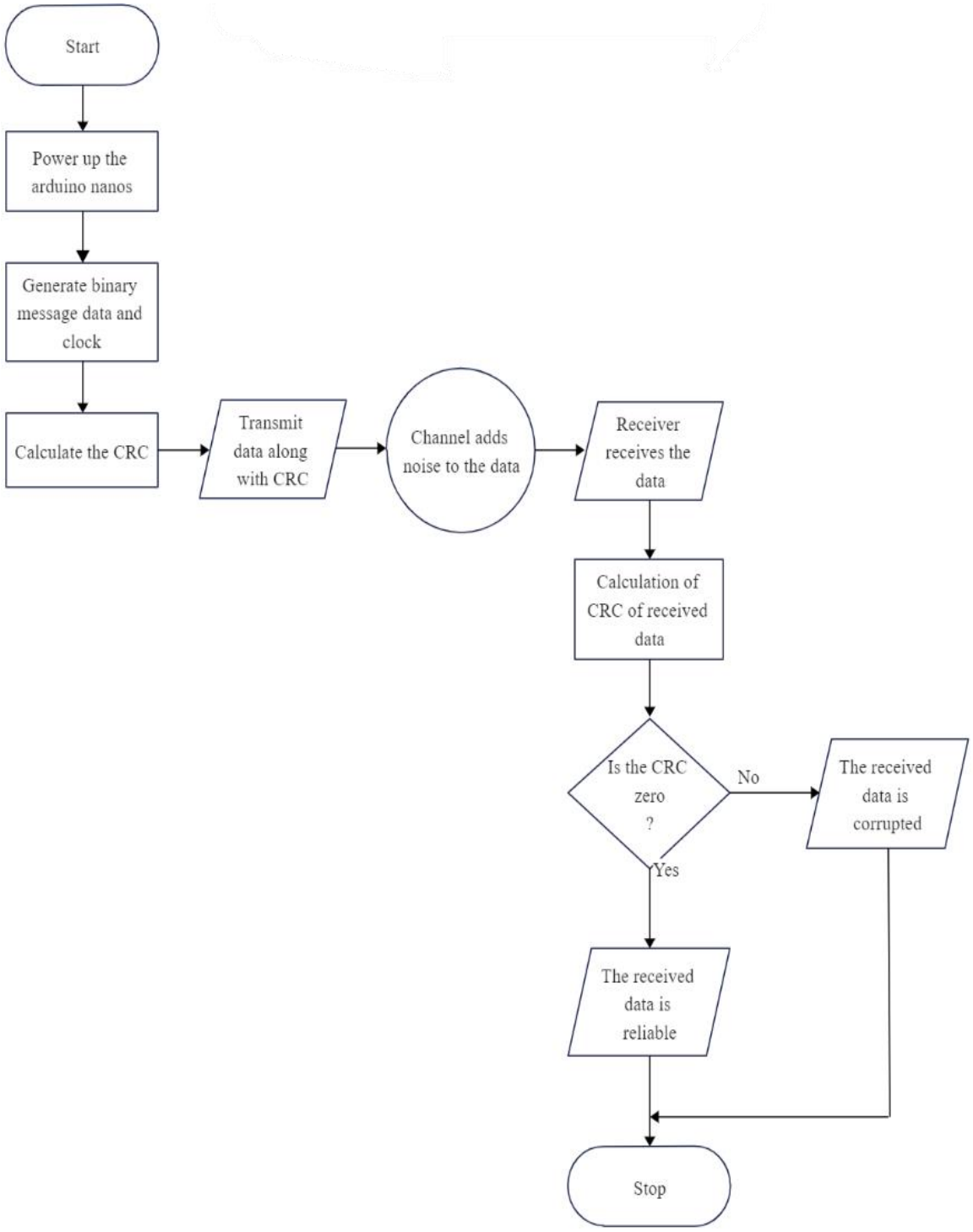

Figure 3.5: System processing flowchart

CHAPTER 4
RESULTS

4.1 MODULE TESTING

4.1.1 POWER TESTING

Test Scenario	Expected Result	Actual Result
Connect the SMPS adaptor to power supply and measure the output voltages by using digital multimeter.	Digital voltmeter should show the output as +5VDC.	Output obtained on Digital voltmeter is +4.93 VDC.

4.1.2 ARDUINO BOARD TESTING

POWER ON LED TEST

Test Scenario	Expected Result	Actual Result
Apply +5VDC power to Arduino board.	Power ON LED should glow.	Power ON LED was glowing.

ARDUINO pin testing

Test Scenario	Expected Result	Actual Result
Apply +5VDC power to Arduino board.	LEDs should glow by writing a LEDs blinking program.	LEDs were glowing.

4.1.3 LCD SCREEN TESTING

Test Scenario	Expected Result	Actual Result
Apply power to LCD and run a standard test code to print “Hello!” on the screen using Arduino.	LCD should turn on and display the message “Hello!”.	The LCD screen successfully powered up and displayed the “Hello!” message.

4.2 SYSTEM TESTING

4.2.1 TRANSMITTER TESTING

While transmitting the message “I’m errorless :)”, the CRC circuit should calculate the CRC to be $(1101\ 0000\ 0100\ 1011)_b$ and this should be visible on the LCD as well as the output of D Flip-Flops.

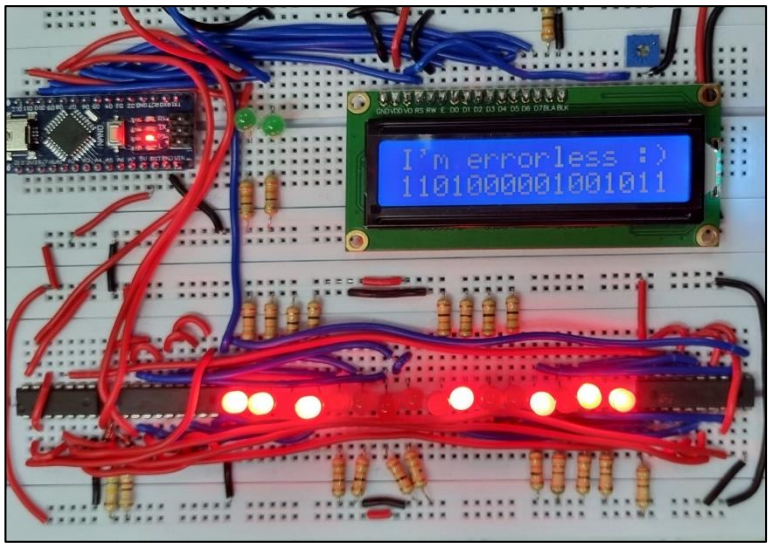

Figure 4.2.1: Transmitter output

4.2.2 RECEIVER TESTING

When receiving the message “I’m errorless :)” along with the correct CRC and given that no error is induced in the channel, the CRC calculation circuit should display the CRC to be zero on the output of the D Flip-Flops.

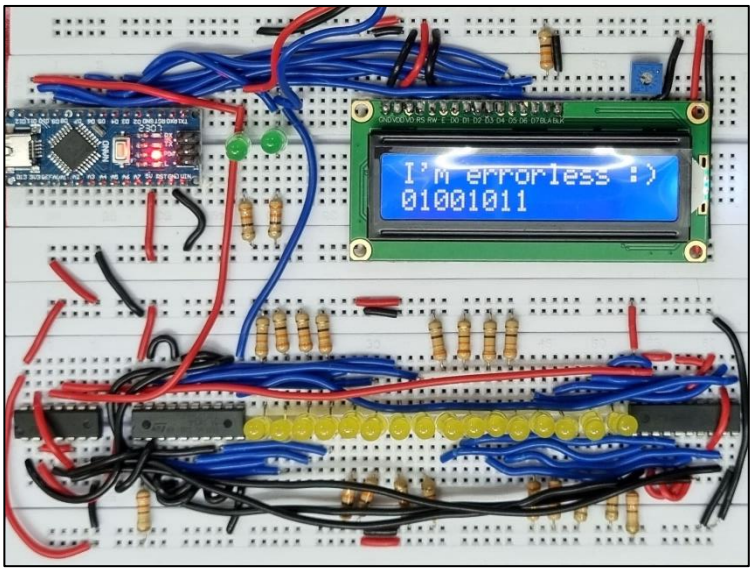

Figure 4.2.2: Receiver output

CHAPTER 5

CONCLUSION AND FUTURE SCOPE

5.1 CONCLUSION

- The conclusion of data transmission and reception with error detection is that it provides a reliable mechanism for ensuring the integrity of data being transferred between systems or devices. By incorporating error detection techniques, such as checksums or cyclic redundancy checks (CRC), it becomes possible to detect and potentially correct errors that may occur during the transmission process.
- When data is transmitted from a sender to a receiver, it is important to verify that the received data is identical to the transmitted data. Errors can occur due to various factors, such as noise, interference, or transmission issues. Error detection techniques involve adding extra bits or information to the transmitted data, which can be used to check for errors upon reception.
- Upon receiving the data, the receiver performs the error detection algorithm using the additional information. If the calculated result matches the received error detection code, it indicates that the data has been received correctly. If there is a mismatch, it signifies that errors may have occurred during transmission, and further action can be taken, such as requesting retransmission or notifying the sender.
- The use of error detection techniques helps to ensure data integrity, reducing the chances of undetected errors and enabling reliable communication between systems. It is particularly crucial in applications where accuracy and reliability are essential, such as telecommunications, networking, file transfers, and digital storage systems.
- In conclusion, incorporating error detection mechanisms in data transmission and reception processes provides a valuable means of verifying data integrity and increasing the reliability of communication systems.

5.2 FUTURE SCOPE

- Satellite Communication and Deep Space Missions: For satellite communication and deep space missions, where transmission distances are vast and signal degradation is significant, error detection and correction will play a crucial role. Future systems will focus on advanced error correction techniques to ensure reliable communication over long distances.
- Intelligent Error Recovery Mechanisms: Future systems may incorporate intelligent error recovery mechanisms that adapt to the specific characteristics of the transmission medium. Machine learning algorithms can be employed to analyse patterns of errors and dynamically adjust the error recovery strategies, improving overall reliability and performance.
- Quantum Error Correction: With the emergence of quantum computing and quantum communication, new error detection and correction techniques will be required. Quantum error correction is an active area of research, aiming to protect quantum states from decoherence and noise. Advancements in this field will play a crucial role in future quantum communication systems.
- Increased Speed and Bandwidth: With the growing demand for data-intensive applications such as high-definition video streaming, virtual reality, and Internet of Things (IoT) devices, there will be a need for faster data transmission and higher bandwidth. Future technologies may focus on improving data rates and expanding the capacity of communication networks.

CHAPTER 6

REFERENCES

[1] Ilievska N (2019) Towards the Fixed Length Redundancy Code. 2019 27th Telecommunications (TELFOR) https://doi.org/10.1109/teor48224.2019.8971113 Zeh A, Bezzateev S (2012) Describing a cyclic code by another cyclic code. 2012 IEEE International Symposium on Information Theory Proceedings .

[2] Fujimoto K, (2018) Checksum and Cyclic Redundancy Check Mechanism. Encyclopedia of Database Systems, 423–424 https://doi.org/10.1007/978-1-4614-8265-9_1474

[3] Турдиев О, А Хомоненко А, Д Гофман, М,В. (2022) Comparison of probable code number PNC and cyclic redundancy code CRC. Vestnik of Russian New University. Series «Complex Systems: Models, Analysis, Management», 4, 119–131 https://doi.org/10.18137/rnu.v9187.21.04.p.119

[4] Bhure R D, Manjunathachari K (2020) Quad Code Sequence Generation Using Cyclic Redundancy Technique to Enhance Target Detection in Doppler Rader. Journal of Electrical Engineering & Technology, 15(6), 2745–2757 https://doi.org/10.1007/s42835-020-00519-1

[5] Zhilin I, Kreschuk A (2016) Generalized concatenated code constructions with low overhead for optical channels and NAND-flash memory. 2016 XV International Symposium Problems of Redundancy in Information and Control (REDUNDANCY) https://doi.org/10.1109/red.2016.7779358

[6] Yacoub EB (2023) Trapping and Absorbing Set Enumerators for Protograph-Based Generalized LowDensity Parity-Check Code Ensembles. 2023 XVIII International Symposium Problems of

[7] Redundancy in Information and Control Systems (REDUNDANCY) https://doi.org/10.1109/redundancy59964.2023.10330177

[8] Deundyak VM, Kosolapov Yu V (2019) On the strength of asymmetric code cryptosystems based on the merging of generating matrices of linear codes. 2019 XVI International Symposium “Problems of Redundancy in Information and Control https://doi.org/10.1109/redundancy48165.2019.9003319

[9] sais M, Rafalia N, Abouchabaka J (2023) DNA technology for big data storage and error detection solutions Hamming code vs Cyclic Redundancy Check (CRC) E3S Web of Conferences, 412, 01090 https://doi.org/10.1051/e3sconf/202341201090

APPENDIX A – BILL OF MATERIALS

Sl. No.	Component Name	Qty	Cost
1	Arduino Nano v3.0	2	1,000
2	16x2 (1602) LCD Display	2	300
3	Solderless Breadboard	3	300
4	10k ohm pot	2	100
5	Current limiting resistors	50	75
6	LEDs	36	75
7	74HC273 (Octal D F/F)	4	100
8	74HC86 (Quad 2-input XOR)	2	25
9	Connecting wires	-	300
10	Diode	2	25
Total			2,300

APPENDIX B – CODE

Transmitter:

// Start of code

```
#include<LiquidCrystal.h>

// Transmit rate in bps
#define TX_RATE 5
#define TX_CLOCK 2
#define TX_DATA 3
#define LCD_D4 4
#define LCD_D5 5
#define LCD_D6 6
#define LCD_D7 7
#define LCD_RS 8
#define LCD_EN 9

const char *message = "Hello, world!";

void setup() {
  pinMode(TX_CLOCK, OUTPUT);
  pinMode(TX_DATA, OUTPUT);

  // Initialize the LCD screen
  LiquidCrystal lcd(LCD_RS, LCD_EN, LCD_D4, LCD_D5, LCD_D6, LCD_D7);
  lcd.begin(16, 2);
  lcd.setCursor(0, 0);
  lcd.print(message)
  for (int byte_idx = 0; byte_idx < strlen(message); byte_idx++) {
    char tx_byte = message[byte_idx];

    // Clear the second line of the display
    lcd.noCursor();
    lcd.setCursor(0, 1);
    lcd.print("        ");
    lcd.setCursor(byte_idx, 0);
    lcd.cursor();

    for (int bit_idx = 0; bit_idx < 8; bit_idx++) {
      bool tx_bit = tx_byte & (0x80 >> bit_idx);
      digitalWrite(TX_DATA, tx_bit);
      delay((1000 / TX_RATE) / 2);
```

```
      // Update the LCD
      lcd.noCursor();
      lcd.setCursor(bit_idx, 1);
      lcd.print(tx_bit ? "1" : "0");
      lcd.setCursor(byte_idx, 0);
      lcd.cursor();

      // Pulse clock
      digitalWrite(TX_CLOCK, HIGH);
      delay((1000 / TX_RATE) / 2);
      digitalWrite(TX_CLOCK, LOW);
    }
  }
  digitalWrite(TX_DATA, LOW);
}

void loop() {
  // put your main code here, to run repeatedly:
}
```

// *End of code*

Receiver:

// Start of code

```
#include <LiquidCrystal.h>

// Pin assignments
#define RX_CLOCK 2
#define RX_DATA 3
#define LCD_D4 4
#define LCD_D5 5
#define LCD_D6 6
#define LCD_D7 7
#define LCD_RS 8
#define LCD_EN 9

LiquidCrystal lcd(LCD_RS, LCD_EN, LCD_D4, LCD_D5, LCD_D6, LCD_D7);

char message[16];
volatile byte rx_byte = 0;
volatile int bit_position = 0;
volatile bool update_lcd = true;

void setup()
 pinMode(RX_DATA, INPUT);
 strcpy(message, "");
 lcd.begin(16, 2);
 attachInterrupt(digitalPinToInterrupt(RX_CLOCK), onClockPulse, RISING);
}

void onClockPulse() {
 bool rx_bit = digitalRead(RX_DATA);

 if (bit_position == 8) {
  rx_byte = 0;
  bit_position = 0;
 }

 if (rx_bit) {
  rx_byte |= (0x80 >> bit_position);
 }

 bit_position += 1;

 if (bit_position == 8) {
  strncat(message, (const char *)&rx_byte, 1);
 }
 update_lcd = true;
```

```
}

void loop() {
  if (update_lcd) {
    update_lcd = false;

    lcd.noCursor();
    lcd.setCursor(0, 0);
    lcd.print(message);
    lcd.setCursor(0, 1);

    for (int i = 0; i < 8; i += 1) {
      if (i < bit_position) {
        lcd.print(rx_byte & (0x80 >> i) ? "1" : "0");
      } else {
        lcd.print(" ");
      }

    lcd.setCursor(strlen(message), 0);
    lcd.cursor();
  }
}
```

// *End of code*

APPENDIX C – REFERENCES

1. https://en.wikipedia.org/wiki/Error_detection_and_correction
2. https://en.wikipedia.org/wiki/Parity_bit
3. https://www.youtube.com/@BenEater
4. https://en.wikipedia.org/wiki/Checksum
5. https://en.wikipedia.org/wiki/Cyclic_redundancy_check
6. https://ocw.mit.edu/communication-systems-engineering-spring-2009.pdf
7. http://web.mit.edu/6.02/www/f2010/handouts/lectures/L7.pdf