Digital Pulse-Coded Laser Authentication System

Contents

Abstract

This project implements a contactless digital authentication system utilizing modulated laser pulses for access control applications. The system employs an 8-bit binary sequence transmitted via a 650nm laser diode and detected by a dedicated laser sensor, demonstrating the practical application of optical physics principles in modern security solutions. The implementation achieves reliable authentication while maintaining cost-effectiveness through the use of readily available components.

1 Introduction

1.1 Background

Traditional authentication systems often rely on physical contact or radio frequency (RF) communication, each presenting their own limitations. Physical contact systems are subject to wear and contamination, while RF systems can be vulnerable to interference and interception. Optical authentication offers an alternative approach that combines the benefits of contactless operation with directed, line-of-sight security.

1.2 Project Objectives

- Design and implement a contactless authentication system using laser-based communication
- Achieve reliable signal transmission and detection under normal indoor lighting conditions
- Optimize the timing parameters for practical usage while maintaining system integrity
- Demonstrate the feasibility of low-cost optical authentication using common components

2 System Architecture

2.1 Hardware Components

2.1.1 Transmission Module

- 650nm laser diode module
- microcontroller
- Power supply: USB connection via Arduino

2.1.2 Reception Module

- Laser sensor module with built-in signal conditioning
- Arduino analog input for signal processing
- Status LEDs (Red and Green) with appropriate current-limiting resistors

2.2 Authentication Protocol

The system implements an 8-bit authentication sequence:

- Binary pattern: [1, 0, 1, 1, 0, 0, 1, 1]
- Optimized timing parameters:
  - Pulse width: 100ms (HIGH state duration)
  - Bit interval: 200ms (total time per bit)
  - Complete sequence duration: ∼1.6 seconds

3 Implementation Details

3.1 Signal Generation

The transmission module generates precisely timed laser pulses using the following parameters:

```
const unsigned long pulseWidth = 100;     // Active pulse duration (ms)
const unsigned long bitInterval = 200;    // Time between pulse starts (ms)
const unsigned long readInterval = 20;    // Sensor sampling interval (ms)
const unsigned long debounceTime = 500;   // Anti-noise delay (ms)
```

3.2 Signal Detection and Processing

The reception system employs several key features to ensure reliable operation:

1. Analog threshold detection
2. Software debouncing
3. Circular buffer implementation for continuous monitoring
4. Pattern matching algorithm for sequence verification

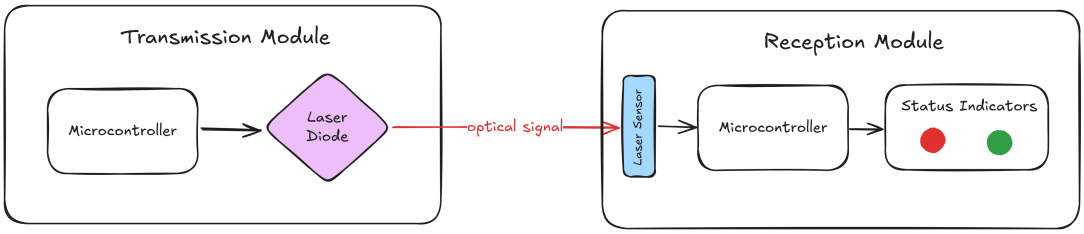

Figure 1: Signal Flow Diagram

4 System Performance

4.1 Timing Optimization

The system's timing parameters were optimized through iterative testing:

Parameter	Original	Optimized	Improvement
Pulse Width	300ms	100ms	66.7% reduction
Bit Interval	500ms	200ms	60% reduction
Total Sequence	4.0s	1.6s	60% reduction
Debounce Time	1000ms	500ms	50% reduction

Table 1: Timing Optimization Results

4.2 Reliability Considerations

The system's reliability depends on several factors:

- Line-of-sight maintenance between transmitter and receiver
- Physical stability of the mounting system
- Power supply stability

5 Future Improvements

Potential enhancements for the system include:

- Implementation of error correction codes
- Support for multiple unique authentication sequences
- Integration of mechanical mounting solutions
- Development of a more compact form factor

6 Conclusion

This project successfully demonstrates a practical, contactless authentication system using modulated laser pulses. By employing readily available components and optimizing timing parameters, the system achieves reliable authentication with a significantly reduced processing time. This work validates the feasibility of low-cost optical authentication for access control and lays a foundation for future advancements in optical security technologies, such as increased security through more complex modulation schemes, integration with existing security infrastructure, and miniaturization for broader applicability.

7 References

1. Arduino LLC. (2024). Arduino Official Documentation.
2. Fritzing. (n.d.). Fritzing: Software for electronics. Retrieved from https://fritzing.org/

A Appendix A: Circuit Diagram

A.1 Circuit Schematics

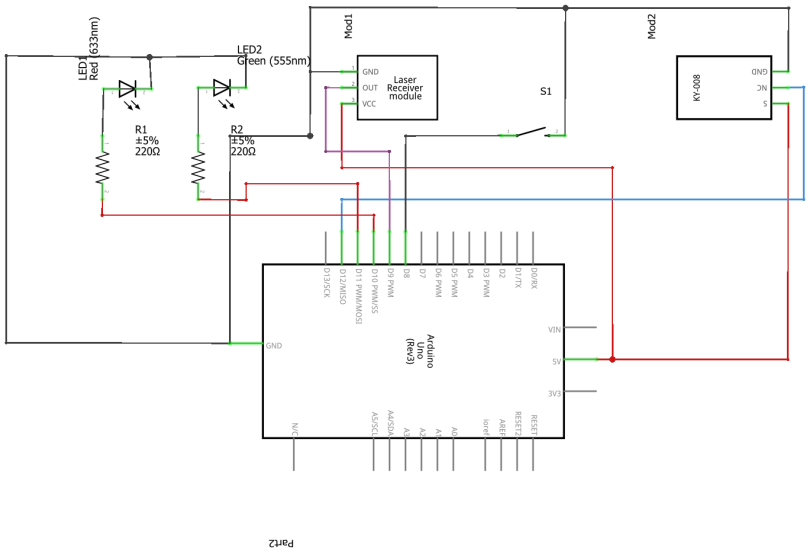

Figure 2: Circuit Schematics

A.2 Circuit Connections

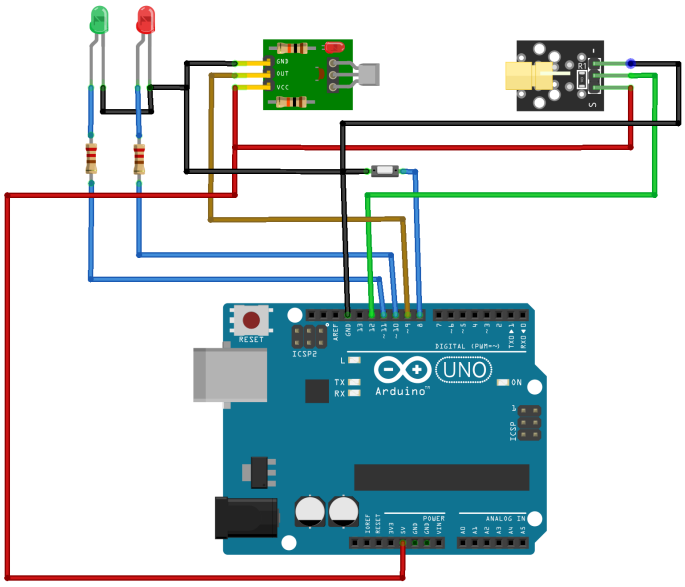

Figure 3: Circuit Connections

B Appendix B: Source Code

The complete Arduino source code implementation[1]:

```
const int laserPin = 12;      // laser diode
const int greenLedPin = 11;   // unlock indicator led
const int redLedPin = 10;     // lock indicator led
const int buttonPin = 8;      // push button pin
const int receiverPin = 9;    // laser receiver

const int inputSequence[] = { 1, 0, 1, 1, 0, 0, 1, 1 };
const int expectedSequence[] = { 1, 0, 1, 1, 0, 0, 1, 1 };
const int dataLength = sizeof(inputSequence) / sizeof(inputSequence[0]);
int receivedBuffer[24];  // circular buffer
int bufferIndex = 0;

// timing constants
const unsigned long pulseWidth = 100;
const unsigned long bitInterval = 200;
const unsigned long readInterval = 20;
const unsigned long debounceTime = 50;

// state variables
unsigned long previousMillis = 0;
unsigned long lastReadTime = 0;
unsigned long lastButtonDebounce = 0;
int dataIndex = 0;
bool transmitting = false;
bool isUnlocked = false;
int lastButtonState = HIGH;
int buttonState = HIGH;

void setup() {
  Serial.begin(9600);
  pinMode(laserPin, OUTPUT);
  pinMode(greenLedPin, OUTPUT);
  pinMode(redLedPin, OUTPUT);
  pinMode(buttonPin, INPUT_PULLUP);
  pinMode(receiverPin, INPUT);

  digitalWrite(laserPin, LOW);
  digitalWrite(greenLedPin, LOW);
  digitalWrite(redLedPin, HIGH);
}

void startNewTransmission() {
  transmitting = true;
  dataIndex = 0;
```

```
  Serial.println("\n=== Starting transmission ===");
  previousMillis = millis();
}

void handleButton() {
  int reading = digitalRead(buttonPin);

  if (reading != lastButtonState) {
    lastButtonDebounce = millis();
  }

  if ((millis() - lastButtonDebounce) > debounceTime) {
    if (reading != buttonState) {
      buttonState = reading;
      if (buttonState == LOW && !transmitting) {
        startNewTransmission();
      }
    }
  }
  lastButtonState = reading;
}

void checkReceivedSequence() {
  receivedBuffer[bufferIndex] = digitalRead(receiverPin);
  bufferIndex = (bufferIndex + 1) % 24;

  for (int start = 0; start <= 24 - dataLength; start++) {
    bool matched = true;
    for (int i = 0; i < dataLength; i++) {
      if (receivedBuffer[(start + i) % 24] != expectedSequence[i]) {
        matched = false;
        break;
      }
    }
    if (matched) {
      isUnlocked = !isUnlocked;
      digitalWrite(greenLedPin, isUnlocked);
      digitalWrite(redLedPin, !isUnlocked);
      Serial.println(isUnlocked ? "UNLOCKED" : "LOCKED");
      memset(receivedBuffer, 0, sizeof(receivedBuffer));
      bufferIndex = 0;
      break;
    }
  }
}

void loop() {
```

```
  unsigned long currentMillis = millis();

  handleButton();

  // transmit
  if (transmitting) {
    if (dataIndex < dataLength) {
      unsigned long timeSinceBitStart = currentMillis - previousMillis;

      digitalWrite(
        laserPin,
        timeSinceBitStart < pulseWidth &&
        inputSequence[dataIndex] == 1 ? HIGH : LOW
      );

      if (timeSinceBitStart >= bitInterval) {
        previousMillis = currentMillis;
        dataIndex++;
      }
    } else {
      transmitting = false;
      digitalWrite(laserPin, LOW);
    }
  }

  // receive
  if (currentMillis - lastReadTime >= readInterval) {
    lastReadTime = currentMillis;
    checkReceivedSequence();
  }
}
```

[1]This code represents the theoretical implementation of the system.