Intercom System

Săsărman Darius Eric

Tătar Marius Tudor

Project Purpose

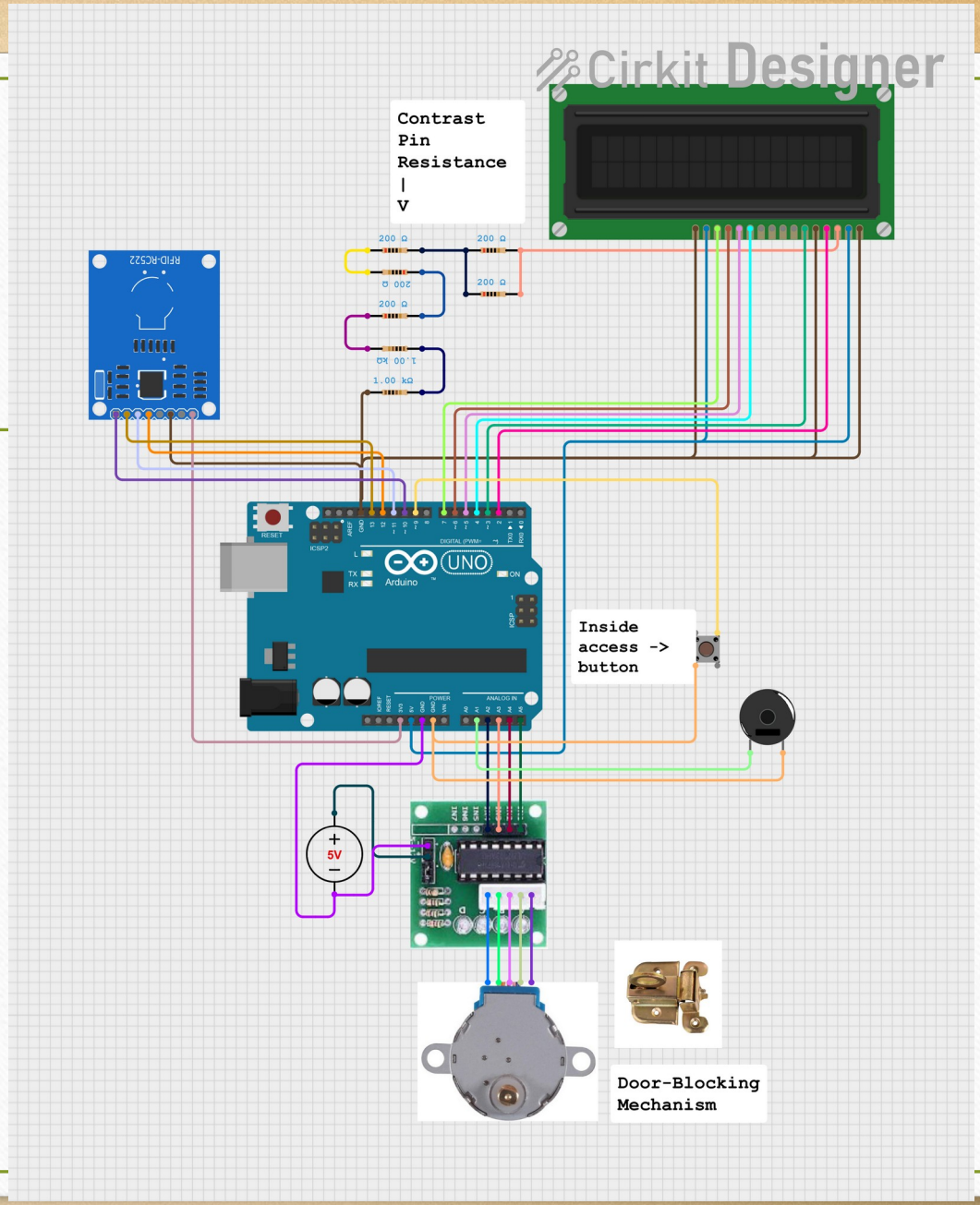
This project is an Arduino-based intercom security system designed to protect against unauthorized access. It alerts the user whenever the door is opened or when suspicious activity is detected. By combining RFID authentication, alert mechanisms, and intentional delays, the system helps prevent brute-force access attempts.

Visual Connection Representation

For designing this project we used an online-available design software available called Cirkit Designer.

In the following slides, we will proceed with explaining what each component does and how all components are connected.

The design is available online at: https://app.cirkitdesigner.com/project/88a5f0a9-2c33-4f38-bbb4-efba90fe0c04



Components used

Arduino Uno Board

This board serves as the central control unit for our project. All other components are connected to it.

Though limited in terms of pin count and computation, we chose this board as online support is widely available.

Technical Specifications:
Microcontroller: ATmega328P
Board type: Microcontroller development board
Operating voltage: 5V
Input voltage (recommended): 7–12V
Input voltage (limits): 6–20V

Input / Output
Digital I/O pins: 14
- 6 support **PWM** (~)

Analog input pins: 6 (A0–A5)
DC current per I/O pin: 20 mA
DC current for 3.3V pin: 50 mA

Piezo-electric buzzer

Pin	Arduino Connection	Mentions
Pin 1	A1	Alternative Voltage source
Pin 2	GND	-

This buzzer serves as the main alarm notifier when an intrusion is attempted or an entrance is executed.

Working principle:

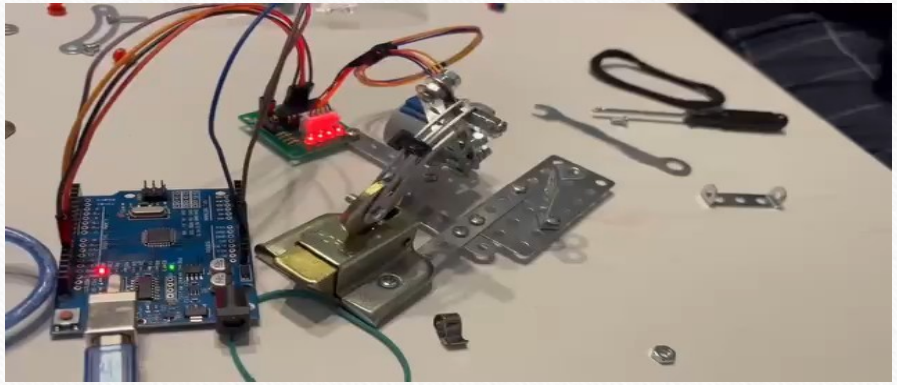
- The buzzer is an **electromagnetic actuator** that contains a **coil (inductor)** inside.
- When the Arduino sends a **digital signal (PLAY / ON)**, an alternative electric (0-5V) **current flows through the coil**.
- The energized coil creates a **magnetic field**, which moves a **metal diaphragm**.
- This vibration of the diaphragm produces **sound**.
- When the signal stops, the magnetic field disappears and the sound stops.

LCD 1602 Display

General Features
Type: Character LCD Display
Display Format: 16 × 2 characters
Controller: HD44780-compatible
Interface: 4-bit / 8-bit parallel
Operating Voltage: 5 V
Character Matrix: 5 × 8 dots
Backlight: LED (White)
Used for checking the internal State.

Pin	Arduino Connection	Mentions
VSS	GND	Voltage source for the control unit
VDD	5V	Voltage source for the control unit
VO	→ 2.7k OHMs connection → GND	Used for character contrast with background
RW	GND	Always writing
E	D3	-
DB0,DB1,DB2, DB3	-	Not used in 4-bit mode
DB4	D4	-
DB5	D5	-
DB7	D7	-
A	5V	Voltage source for backlight
B	GND	Voltage source for backlight

Stepper Motor : 28BYJ-48 with custom mechanical locking device



Pin	Arduino Pin	Justification
INT1	A5	Pins INT1-4 are used to control the motor’s position
INT2	A4	-
INT3	A3	-
INT4	A2	-
V+,V-	-	Connected to external voltage source that shares a common ground with the arduino board.

28BYJ-48 Stepper Motor
Type: Unipolar Stepper Motor
Step Angle: 5.625° / 64 (≈ 0.088° after gearbox)
Gear Ratio: 1:64
Operating Voltage: 5 V DC
Phases: 4
Coils: 5-wire configuration

ULN2003 Driver Module
Type: Darlington Transistor Array Driver
Channels: 7 (4 used for 28BYJ-48 control)
Input Logic Level: TTL / CMOS compatible
Output Type: Open Collector
Max Output Current: up to 500 mA per channel
Integrated Flyback Diodes: Yes

Inside Unlock Button

Pin	Arduino Connection	Mentions
Pin 2	D9	It is an input that uses the internal pull-up resistor.
Pin 3	GND	-

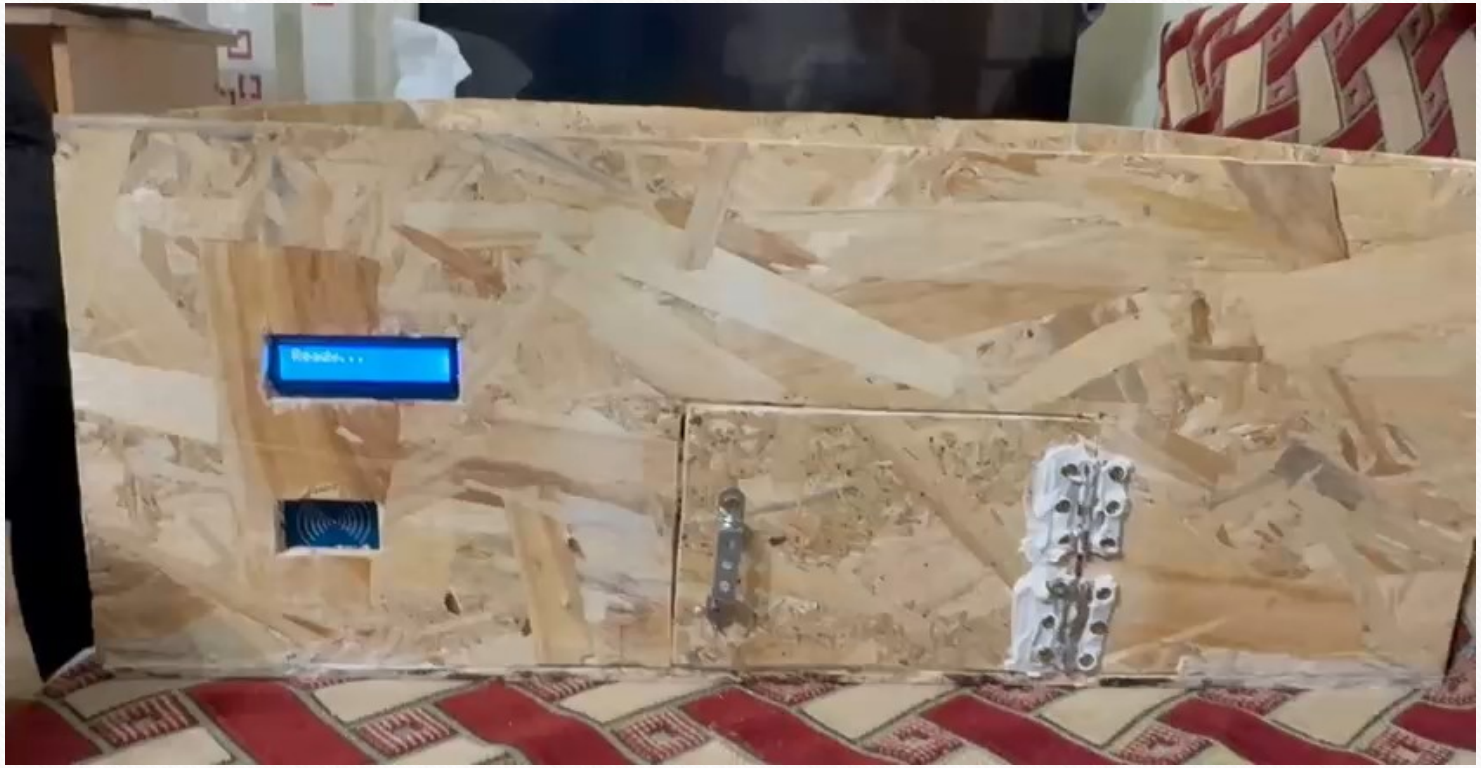
General Description
Type: Momentary push button (tactile switch)
Operation: Normally Open (NO)
Function: Closes the circuit only while pressed
Action type: Momentary (non-latching), Requires debouncing

RFID Authentication Sensor

General Features
Type: RFID / NFC Reader
Operating Frequency: 13.56 MHz
Standard: ISO/IEC 14443A
Chipset: MFRC522
Modulation: ASK (Amplitude Shift Keying)

Pin	Arduino Connection	Mentions
MISO	D12	-
MOSI	D11	-
3.3V	3.3V	-
RST	-	Not used by board : always in reading mode.
SDA	D10	-
GND	GND	-
IRQ	-	Not used by SPI communication.
SCK	D13	-

Physical Representation



The mechanical and physical aspects of this project were an unexpected challenge that we had to overcome.

We wanted to achieve a visual resemblance to the actual object, found at the entrance of any secured building

Control behavior and Code

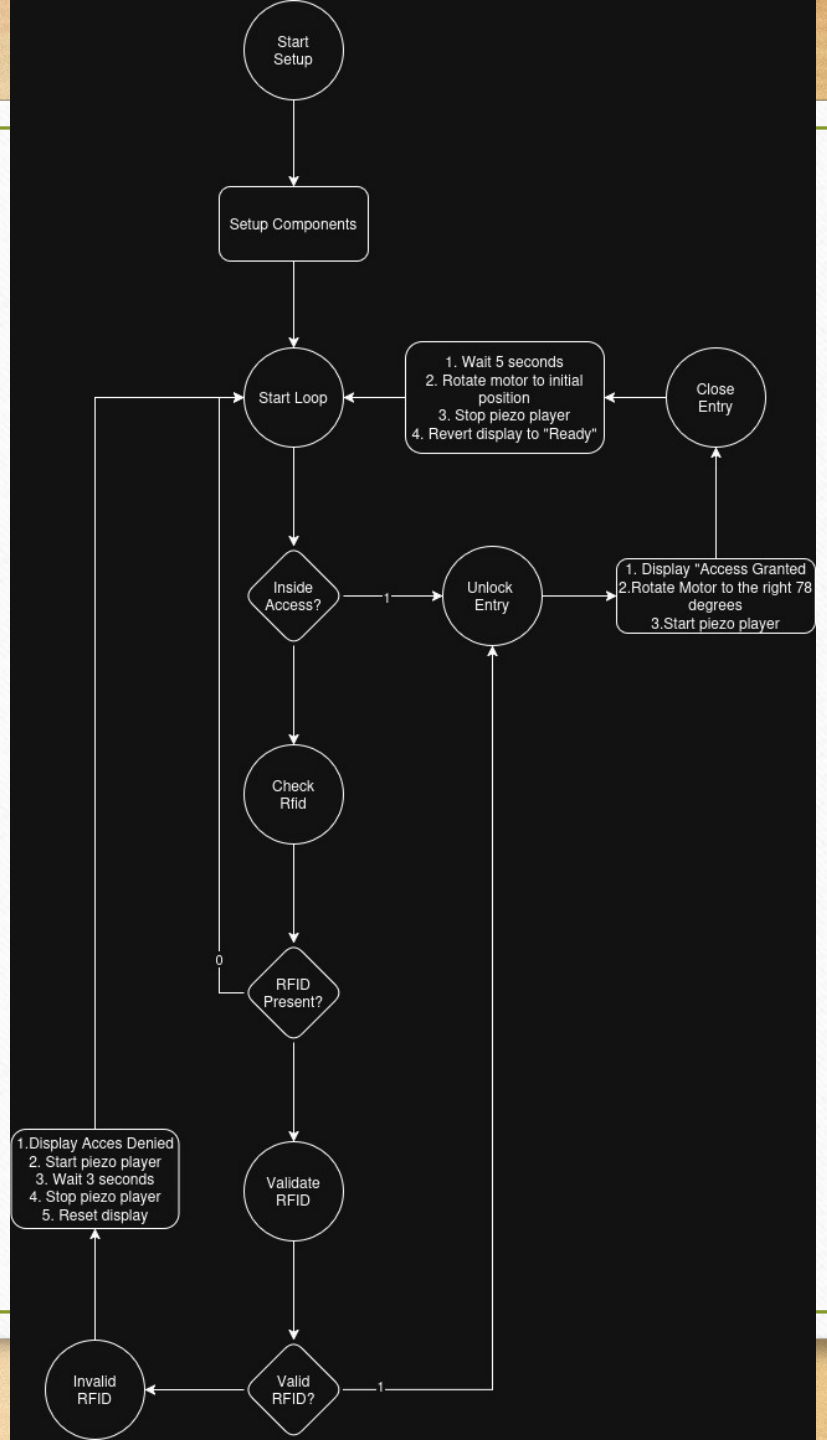
Control behavior

As any Arduino UNO Board does, the control behavior can be broken into two main sections : “Setup” and “Loop”.

The “Setup” function is only called once, at the beginning of the execution. During this phase, the Arduino board executes the following: Stepper Motor settings, pin mode configurations, LCD initialization and securing SPI communication with the RFID reader

The “Loop” function is continually called until the entire execution is halted. During this phase, the Control Unit works as follows :

1. Check if inside acces button is pushed
   if yes => grant access
2. Check if RFID card is present
   if yes and Valid => grant access



Code

```
#include <SPI.h>
 #include <MFRC522.h>
 #include <LiquidCrystal.h>
 #include <Stepper.h>

 /* ====== CONSTANTS ========== */

 const int stepsPerRevolution = 2048;
 const int stepsForDoor = 420;

 /* ======== PINS ======== */

 // RFID
 #define SS_PIN   10
 #define RST_PIN  8
```

```
// LCD
 LiquidCrystal lcd(2, 3, 4, 5, 6, 7);

 // Butoane / LED / Buzzer
 #define BTN_CALL 9
 #define BUZZER   A1

 // Motor
 #define IN1 A5
 #define IN2 A4
 #define IN3 A3
 #define IN4 A2

 /* ======= GLOBAL VARIABLES ====== */

 bool lastButtonState = HIGH;
```

Code

```
  /* ================== COMPONENTS ================== */
  MFRC522 rfid(SS_PIN, MFRC522::UNUSED_PIN);
  Stepper myStepper(stepsPerRevolution, IN1, IN3, IN2, IN4);

/* ================== UID ================== */

byte authorizedUID[4] = {0x73, 0x77, 0xF1, 0x2C};

/* ================== FUNCTIONS ================== */

bool checkUID(byte *uid) {
    for (byte i = 0; i < 4; i++) {
      if (uid[i] != authorizedUID[i])
            return false;
     }
    return true;
}
```

```
void beep(int duration = 200) {
   tone(BUZZER, 3000);
 }
 void accessDenied() {
   lcd.clear();
   lcd.print("Access Denied");
   beep(500);
   delay(2000);
   noTone(BUZZER);
   lcd.clear();
   lcd.print("Ready...");
 }
```

Code

```
void unlockDoor() {
    lcd.clear();
    lcd.print("Access Granted");
    beep(300);

    myStepper.step(-stepsForDoor);
    delay(500);

    delay(5000);

    myStepper.step(stepsForDoor);
    delay(500);
    digitalWrite(IN1, LOW);
    digitalWrite(IN2, LOW);
    digitalWrite(IN3, LOW);
    digitalWrite(IN4, LOW);

    noTone(BUZZER);
    lcd.clear();
    lcd.print("Ready...");
  }
```

Code

```
/* ================== SETUP ================== */

 void setup() {

   myStepper.setSpeed(8);

   pinMode(BTN_CALL, INPUT_PULLUP);
   pinMode(BUZZER, OUTPUT);

   pinMode(A0, INPUT);

   pinMode(IN1, OUTPUT);
   pinMode(IN2, OUTPUT);
   pinMode(IN3, OUTPUT);
   pinMode(IN4, OUTPUT);
```

```
   lcd.begin(16, 2);
   lcd.print("Initializing...");

   pinMode(10, OUTPUT);
   digitalWrite(10, HIGH);
   SPI.begin();
   rfid.PCD_Init();

   delay(1000);
   lcd.clear();
   lcd.print("Ready...");
 }
```

Code

```
/* ================== LOOP ================== */
 void loop() {
   bool currentButtonState = digitalRead(BTN_CALL);

   if (lastButtonState == HIGH && currentButtonState
== LOW) {
     lcd.clear();
     lcd.print("Inside unlock");
     delay(3000);
     unlockDoor();
   }

   lastButtonState = currentButtonState;
```

```
if (rfid.PICC_IsNewCardPresent()
&& rfid.PICC_ReadCardSerial()) {
     lcd.clear();
     lcd.print("Reading card rn");
     delay(3000);
     if (checkUID(rfid.uid.uidByte))
unlockDoor();
     else accessDenied();

     rfid.PICC_HaltA();
     rfid.PCD_StopCrypto1();
     lcd.clear();
     lcd.print("ready");
   }
 }
```

Future Improvements

If we were to improve upon this original design...

The first aspect we would change is the control unit. While ARDUINO UNO is great for an introduction, this project, in it's complete state, requires more pins than there are available.

The second target would strive to achieve would be to add the call reciever component, illustrated in the image shown in this slide. It wasn't added yet, due to power constraints.

The third objective would be to expand this single-input and single-call-reciever system into a multiple-input and multiple-call-reciever system. We could achieve this by multiplexing.

Other secure components could be added as well.

Thank you for your attention