

علی محمدی سراجی
محمد مهدی کریمی نیک

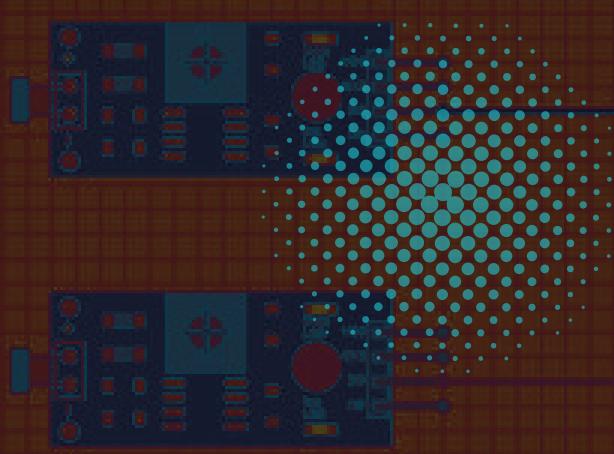
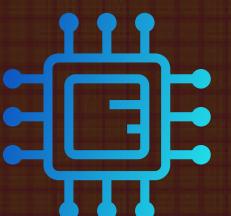
Arduino Parking System

معرفی

مقدمه

بدنه

نتیجه



سیستم پارکینگ هوشمند

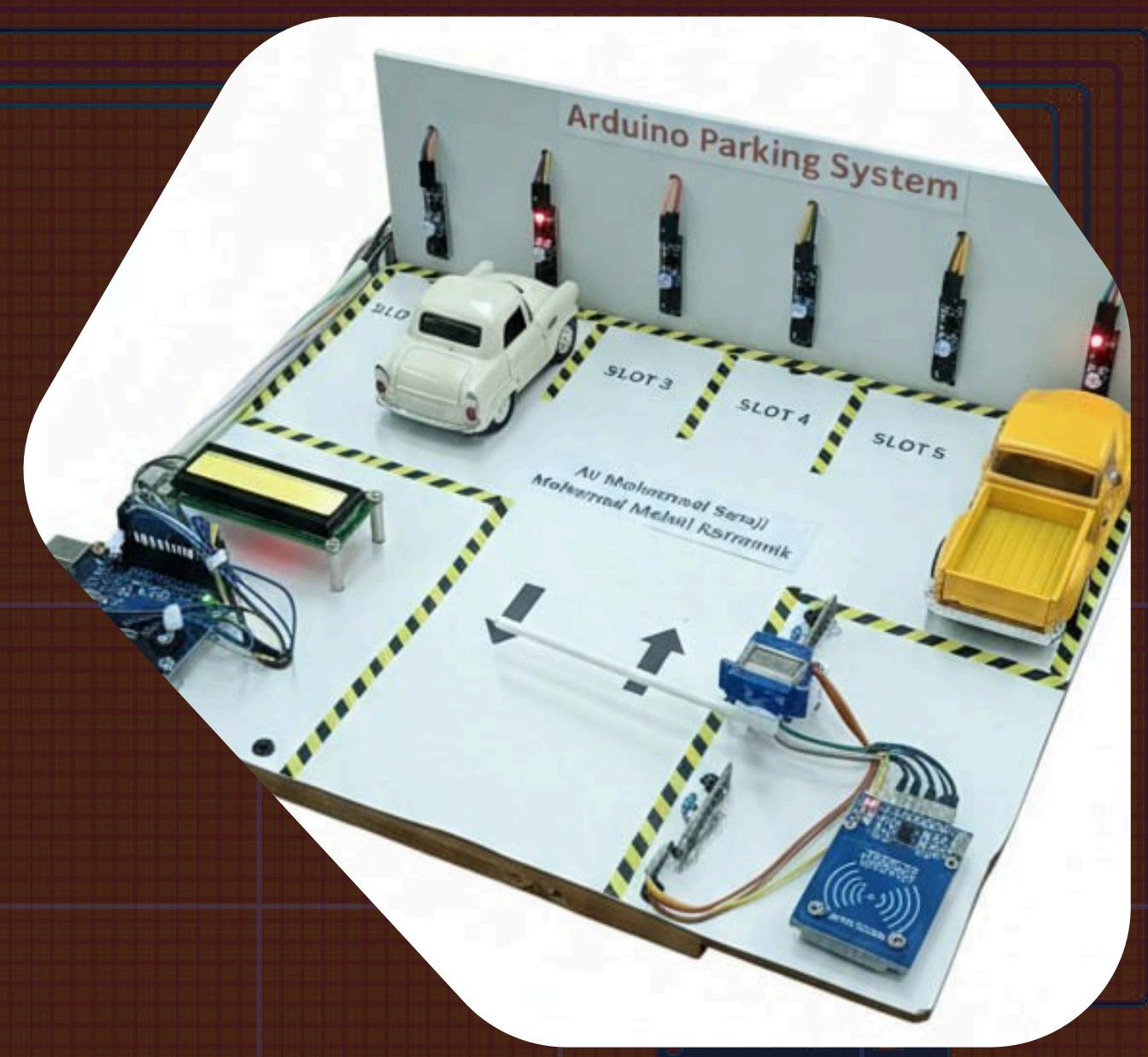
- طراحی سیستم پارکینگ هوشمند با ۶ جای پارک
- تشخیص خودکار حضور خودرو با سنسورها
- باز و بسته شدن درب با سروو موتور
- نمایش وضعیت پارکینگ و ظرفیت خالی روی LCD

معرفی

مقدمه

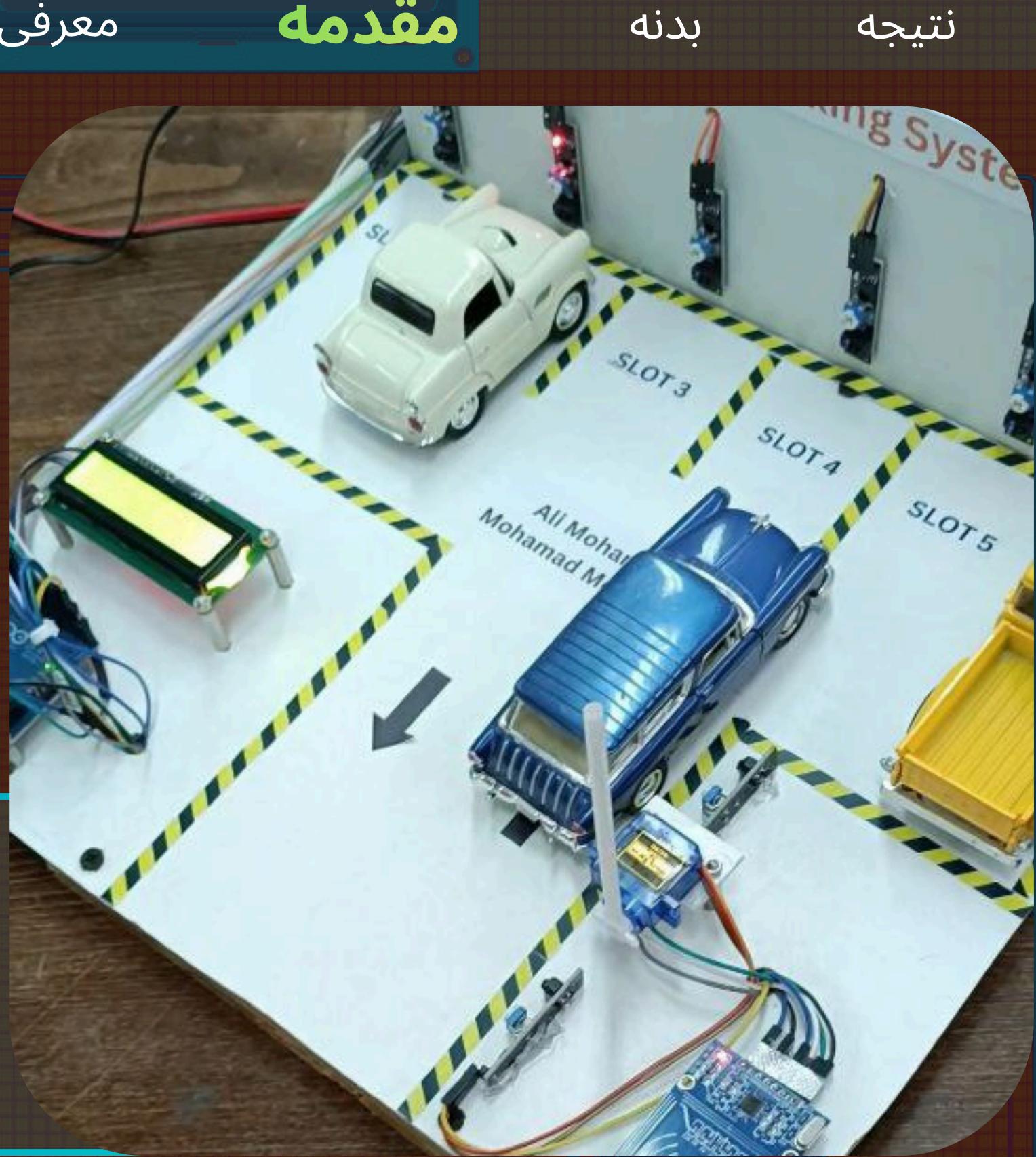
بدنه

نتیجه

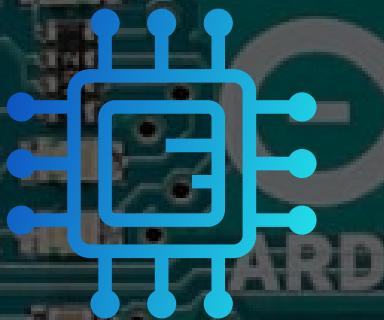


هدف از این پروژه؟

- افزایش دقت و سرعت در مدیریت ظرفیت پارکینگ
- حذف ورود افراد غیرمجاز با استفاده از کارت خوان RFID
- کاهش نیاز به نیروی انسانی در کنترل درب ورودی
- نمایش لحظه‌ای وضعیت جای پارک‌ها
- ایجاد امکان شبیه‌سازی و تست کامل سیستم قبل از اجرا



Arduino Uno



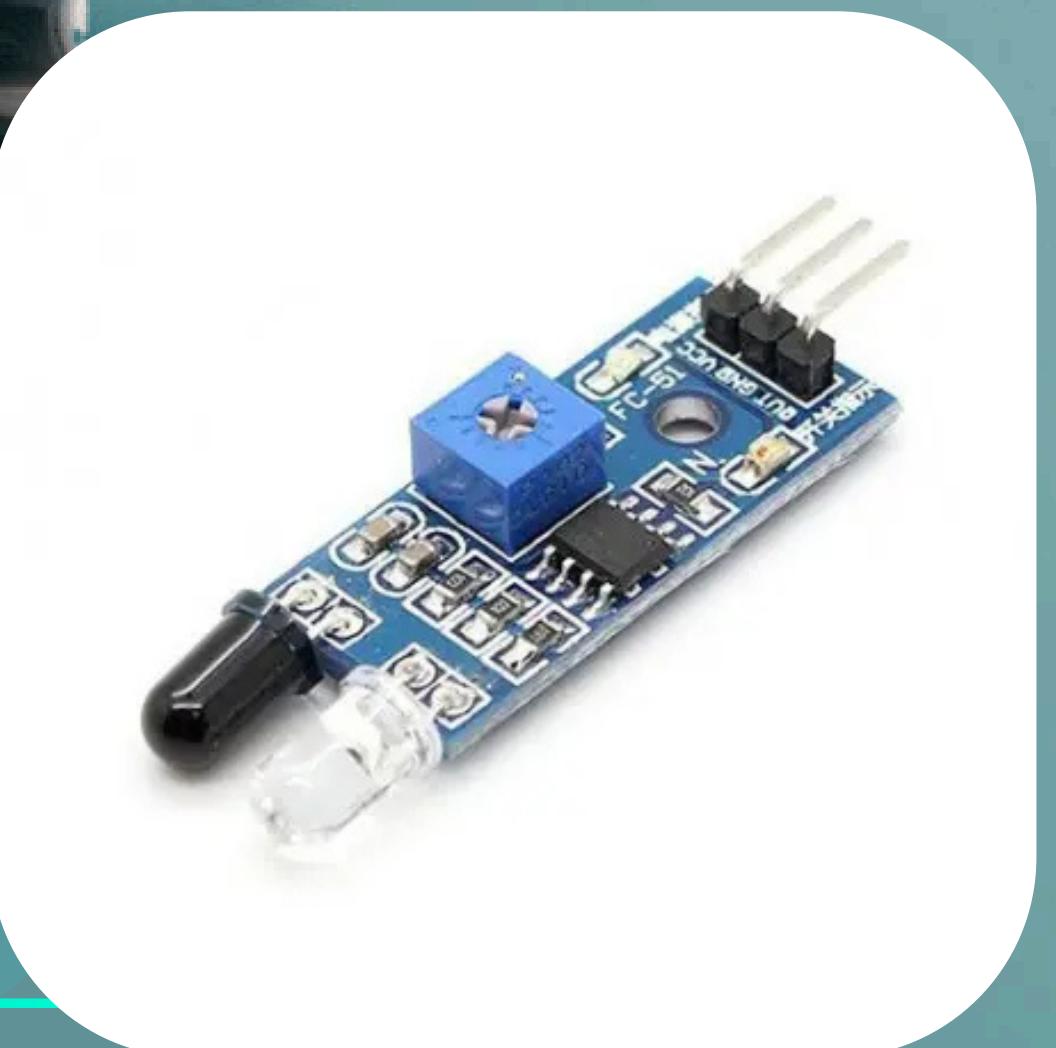
- مبتنی بر میکروکنترلر ATmega328P
- ۱۴ پایه دیجیتال و ۶ ورودی آنالوگ
- پشتیبانی از PWM روی ۶ پایه
- پورت USB برای برنامه‌ریزی و تغذیه
- سازگاری بالا با پروتئوس، سنسورها و ماژول‌ها
- مناسب برای پروژه‌های کنترل، رباتیک و IoT



(TCRT5000) سنسور مادون قرمز



- تشخیص حضور خودرو با ارسال و دریافت نور IR
- دارای مازول فرستنده و گیرنده
- خروجی دیجیتال (۱/۰) قابل اتصال مستقیم به آردوینو
- مناسب برای فواصل کوتاه
- حساسیت قابل تنظیم با پتانسیومتر



سرو و موتور (Servo Motor - SG90)



- چرخش کنترل شده بین 0 تا 180 درجه
- دقت بالا و پاسخ سریع
- کنترل توسط سیگنال PWM آردوینو
- مناسب برای باز و بسته کردن درب
- ابعاد کوچک و مصرف توان کم

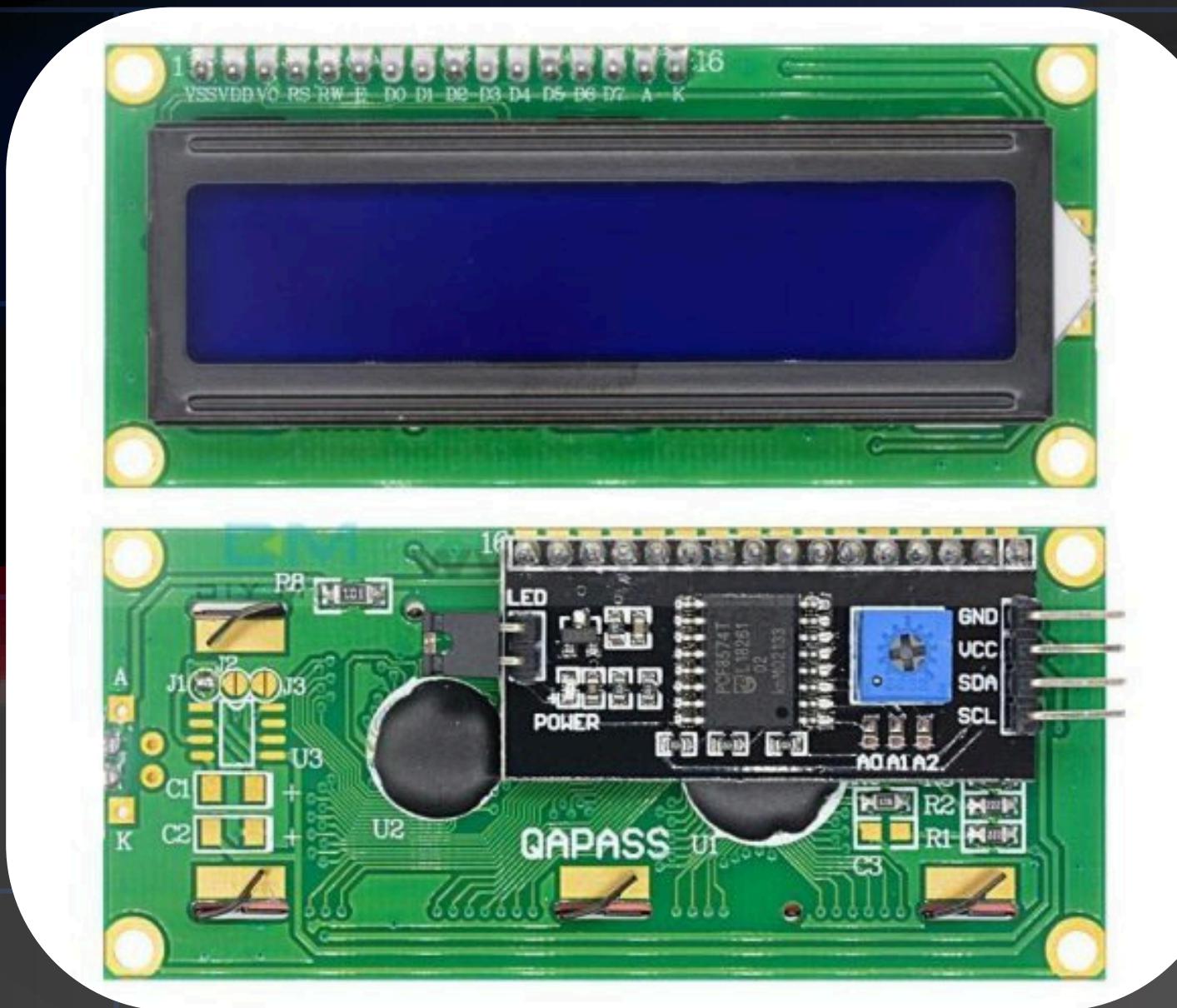
I2C + LCD 16×2

معرفی

مقدمه

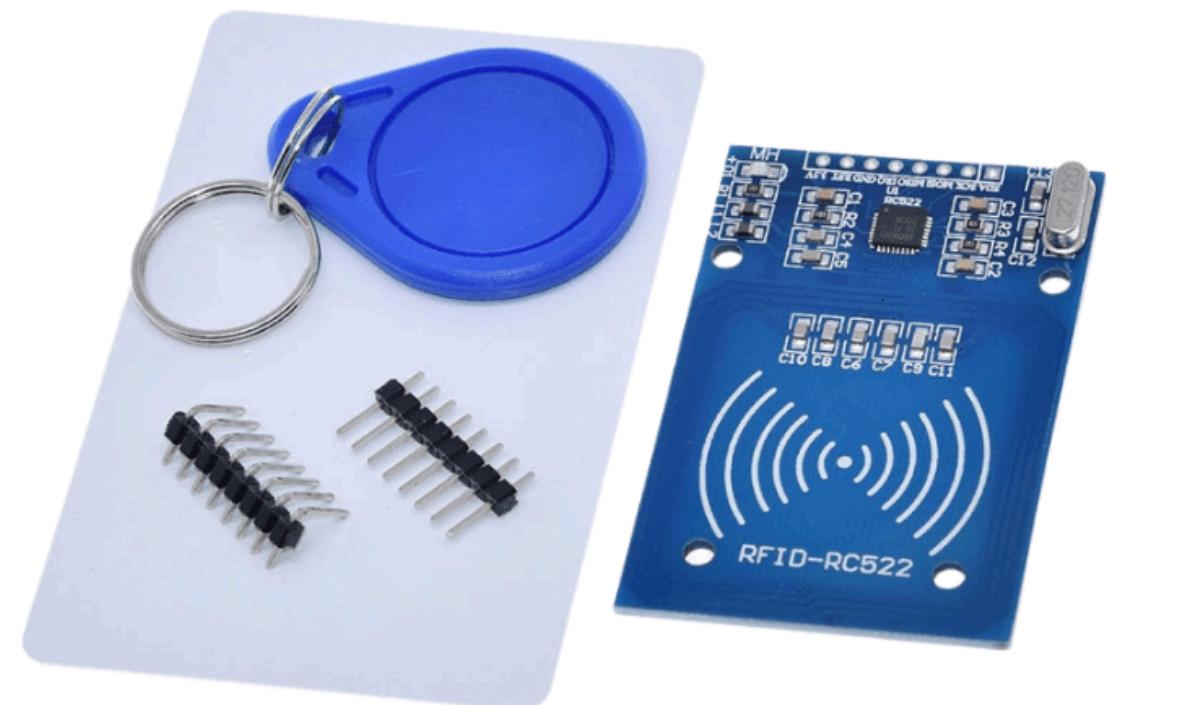
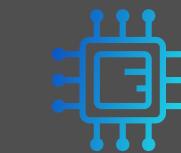
بدنه

نتیجه



- نمایش وضعیت لحظه‌ای ظرفیت پارکینگ
- قابلیت نمایش متن در دو سطر
- ارتباط ساده با استفاده از پروتکل I2C
- کاهش تعداد پایه‌های اتصال از 6-12 پایه به فقط 2 پایه
- امکان تنظیم روشنایی و کنتراست از طریق پتانسیومتر I2C
- مناسب برای سیستم‌های مانیتورینگ و کنترل

کارت خوان RFID (ماژول RC522)

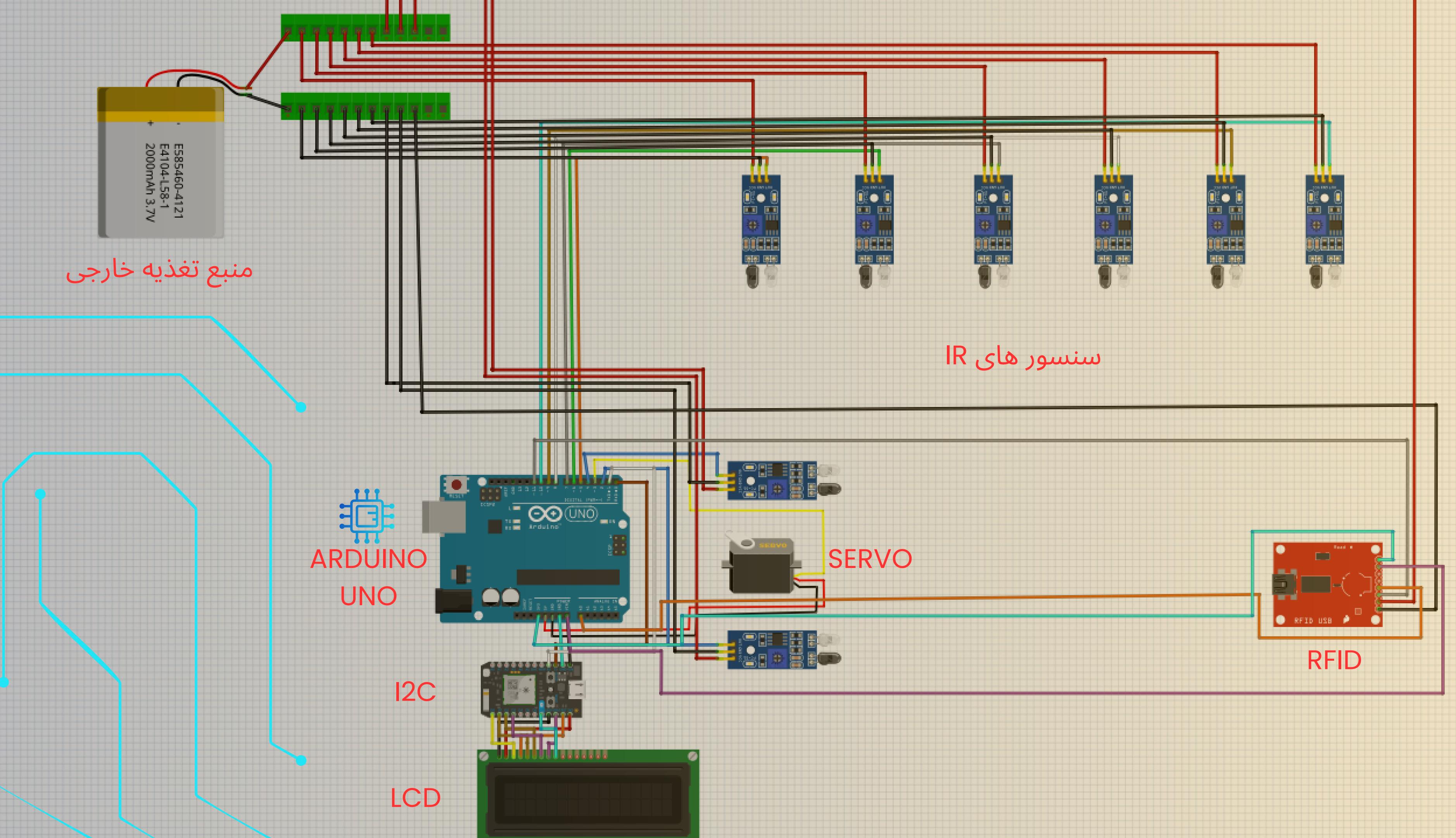


- فرکانس کاری 13.56MHz
- پشتیبانی از کارت‌ها و تگ‌های MIFARE
- ارتباط از طریق رابط SPI با آردوینو
- سرعت بالا در خواندن شناسه کارت
- مناسب برای کنترل دسترسی و احراز هویت
- مصرف توان بسیار کم و قیمت اقتصادی

ساختار کلی سیستم و نوآوری‌ها

- افزودن کارت‌خوان RFID برای کنترل ورود افراد مجاز
- افزایش امنیت و جلوگیری از ورود خودروهای غیرمجاز
- استفاده از ماژول I2C برای کاهش چشمگیر تعداد سیم‌ها
- آزاد شدن پایه‌های بیشتر در آردوینو برای سنسورها و ماژول‌های دیگر
- تنظیم سروو به نحوی که بجای زمان با سنسور کار کند





کد برنامه

```

RFID
if(rfid.PICC_IsNewCardPresent() && rfid.PICC_ReadCardSerial()) {
    lastCard = "";
    for (byte i = 0; i < rfid.uid.size; i++) {
        lastCard += String(rfid.uid.uidByte[i], HEX);
    }
    lastCard.toUpperCase();
    cardOK = true;
    rfid.PICC_HaltA();
}

سن سور ها
void Read_Sensor() {
    S1 = (digitalRead(ir_car1) == 0);
    S2 = (digitalRead(ir_car2) == 0);
    S3 = (digitalRead(ir_car3) == 0);
    S4 = (digitalRead(ir_car4) == 0);
    S5 = (digitalRead(ir_car5) == 0);
    S6 = (digitalRead(ir_car6) == 0);
}
cpp
Copy code
slot = totalSlots - (S1 + S2 + S3 + S4 + S5 + S6);

```

LCD

```

LiquidCrystal_I2C lcd(0x27,16,2);
cpp
Copy code
lcd.setCursor(0,0);
lcd.print("S1:F");
lcd.setCursor(5,0);
lcd.print("S2:E");
cpp
Copy code
if (slot == 0 && !parkingFullShown) {
    if (digitalRead(ir_back) == 0) {
        lcd.clear();
        lcd.print("Parking Full!");
        parkingFullShown = true;
    }
}

```

سرو

```

if (cardOK && digitalRead(ir_enter)
== 0 && flag1 == 0) {
    if (slot > 0) {
        flag1 = 1;
        myservo.write(90);
    }
}
cpp
Copy code
if (digitalRead(ir_back) == 0) {
    myservo.write(90);
    delay(3000);
    myservo.write(180);
}

```

چالش‌ها

سخت افزاری

- پیچیدگی زیاد سیم‌ها و اتصال‌های متعدد
- مشکل در افت ولتاژ و تداخل بین بخش‌ها
- نیاز به منبع تغذیه جداگانه برای پایداری مدار
- زمان‌برترین بخش پروژه بود

معرفی

مقدمه

بدنه

نتیجه

نرم افزاری

- تداخل ورودی‌های سنسورها با یکدیگر
- مشکل خواندن همزمان دیتا
- نیاز به نوشتن و اصلاح خطبه خط کد
- هماهنگ‌سازی RFID با سایر بخش‌ها

نتیجه‌گیری پروژه