

به نام خدا



درس مدارهای واسط

استاد:

دکتر امین فصحتی

پروژه سه

نویسندگان:

یوسف سدیدي | ۴۰۱۱۷۰۵۹۷

مهرشاد دهقانی | ۴۰۱۱۰۵۹۱۲

امیرعلی شیخی | ۴۰۱۱۰۶۱۵۸

کیهان مسعودی | ۴۰۱۱۰۶۵۰۹

زمستان ۱۴۰۳

فهرست مطالب

مقدمه:	۴
بررسی پروتکل‌ها:	۴
۱. پروتکل I2C	۴
معرفی:	۴
ویژگی‌ها:	۴
نحوه‌ی اتصال دو برد آردوینو:	۴
مراحل پیاده‌سازی:	۵
۲. پروتکل SPI	۶
معرفی:	۶
ویژگی‌ها:	۶
نحوه‌ی اتصال دو برد آردوینو:	۶
مراحل پیاده‌سازی:	۷
۳. پروتکل UART	۷
معرفی:	۷
ویژگی‌ها:	۸
نحوه‌ی اتصال دو برد آردوینو:	۸
مراحل پیاده‌سازی:	۸
۴. پروتکل Wi-Fi	۹
معرفی:	۹
ویژگی‌ها:	۹
نحوه‌ی اتصال دو برد آردوینو:	۱۰
مراحل پیاده‌سازی:	۱۰
۵. پروتکل CAN	۱۱

۱۱	معرفی:
۱۱	ویژگی‌ها:
۱۲	نحوه‌ی اتصال دو برد آردوینو:
۱۲	مراحل پیاده‌سازی:
۱۳	۶. پروتکل One-Wire
۱۳	معرفی:
۱۳	ویژگی‌ها:
۱۴	نحوه‌ی اتصال دو برد آردوینو:
۱۴	مراحل پیاده‌سازی:
۱۵	مقایسه پروتکل‌ها:
۱۵	Encryption & Decryption:
۱۵	اندازه‌گیری زمان:
۱۶	مقایسه زمان‌ها:
۱۶	نتیجه‌گیری:

مقایسه عملکرد چند واسط ارتباطی در سیستم‌های نهفته

مقدمه:

در این پروژه، قصد داریم عملکرد چندین واسط ارتباطی را برای انتقال داده‌ها بین دو برد آردوینو مقایسه کنیم. هستند. در ادامه، هر پروتکل CAN, One-Wire, Wi-Fi, UART, SPI, I2C این واسط‌ها شامل کدهای مربوط به صورت جداگانه توضیح داده شده و نحوه‌ی اتصال دو برد آردوینو به یکدیگر شرح داده می‌شود. به تمام موارد به پیوست این فایل موجود است.

بررسی پروتکل‌ها:

۱. پروتکل I2C

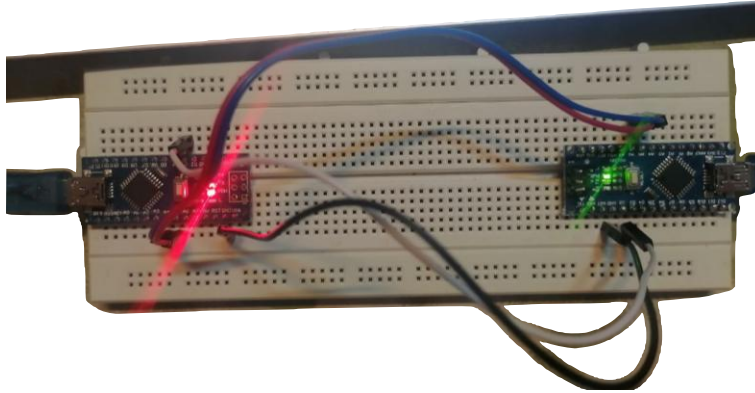
معرفی:

یک روش ارتباطی سریال دو سیمه است که برای انتقال داده I2C (Inter-Integrated Circuit) پروتکل و نمایشگرها مورد استفاده قرار می‌گیرد. این EEPROM بین میکروکنترلرها و قطعات جانبی مانند سنسورها، است که برای همگام‌سازی انتقال داده به کار (SCL) و یک خط کلاک (SDA) پروتکل دارای یک خط داده دارای یک آدرس منحصر به فرد است که امکان ارتباط چندین دستگاه I2C می‌روند. هر دستگاه متصل به باس کنترل ارتباط را Master است که در آن، دستگاه Master-Slave دارای حالت‌های I2C را فراهم می‌کند در اختیار دارد. این پروتکل می‌تواند در نرخ‌های انتقال ۱۰۰ کیلوبیت، ۴۰۰ کیلوبیت، ۱ مگابیت و ۳.۴ مگابیت بر ثانیه عمل کند. سادگی سیم‌کشی، هزینه کم و مصرف توان پایین از مزایای این پروتکل هستند. ویژگی‌ها:

- ارتباط چندگانه (Multi-Master, Multi-Slave)
- استفاده از فقط دو سیم
- قابلیت آدرس‌دهی ۱۲۷ دستگاه روی یک باس
- پشتیبانی از سرعت‌های ۱۰۰ کیلوبیت، ۴۰۰ کیلوبیت و ۳.۴ مگابیت بر ثانیه

نحوه‌ی اتصال دو برد آردوینو:

- SDA (A۴) به SDA
- SCL (A۵) به SCL
- GND به GND



شکل ۱ برد نهایی I2C

مراحل پیاده‌سازی:

۱. تنظیم برد Master برای ارسال داده

۲. تنظیم برد Slave برای دریافت داده

۳. ارسال و دریافت داده با استفاده از توابع Wire.h

I2C-sender | Arduino IDE 2.1.0

```

37
38 digitalWrite(PULSE_PIN, HIGH);
39 delayMicroseconds(100);
40 digitalWrite(PULSE_PIN, LOW);
41
42 int encryptedLength = encryptMessage((byte*)message, strlen(message), encrypted);
43
44 Serial.println("Sending encrypted message in 16-byte chunks...");
45
46 for (int i = 0; i < 128; i += 16) {
47     Wire.beginTransmission(8);
48     Wire.write(&encrypted[i], 16);
49     Wire.endTransmission();
50 }
51
52 Serial.println("Data sent via I2C.");
53 delay(4000);
54 }
55
56 int encryptMessage(byte* input, int inputLength, byte* output) {
57     byte local_iv[16];

```

Output Serial Monitor x

Message (Enter to send message to 'Arduino Nano' on 'COM3') New Line 9600 baud

Message (Enter to send message to 'Arduino Nano' on 'COM3')

Sending encrypted message in 16-byte chunks...

Data sent via I2C.

Sending encrypted message in 16-byte chunks...

Data sent via I2C.

Sending encrypted message in 16-byte chunks...

Data sent via I2C.

Sending encrypted message in 16-byte chunks...

Data sent via I2C.

Sending encrypted message in 16-byte chunks...

Data sent via I2C.

Sending encrypted message in 16-byte chunks...

Data sent via I2C.

I2C-receiver | Arduino IDE 2.1.0

```

47
48 void receiveEvent(int howMany) {
49
50     while (Wire.available()) {
51         char c = Wire.read();
52         receivedEncrypted[receivedIndex++] = c;
53     }
54
55     if (receivedIndex == 128) {
56
57         byte decrypted[156];
58         int decryptedLength = decryptMessage(receivedEncrypted, receivedIndex, decrypted);
59
60         unsigned long receiveTime = micros();
61         unsigned long transmissionTime = receiveTime - pulseTime;
62         Serial.print("Transmission Time: ");
63         Serial.println(transmissionTime);
64
65         if (decryptedLength > 0) {
66             Serial.print("Decrypted message: ");
67             Serial.println((char*)decrypted);

```

Output Serial Monitor x

Message (Enter to send message to 'Arduino Nano' on 'COM4') New Line 9600 baud

Message (Enter to send message to 'Arduino Nano' on 'COM4')

Transmission Time: 21736

Decrypted message: This is a 128-character message. It needs to be exactly 128 characters

Transmission Time: 21744

Decrypted message: This is a 128-character message. It needs to be exactly 128 characters

Transmission Time: 22768

Decrypted message: This is a 128-character message. It needs to be exactly 128 characters

Transmission Time: 21736

Decrypted message: This is a 128-character message. It needs to be exactly 128 characters

Transmission Time: 21740

Decrypted message: This is a 128-character message. It needs to be exactly 128 characters

شکل ۲ تصاویر خروجی I2C

۲. پروتکل SPI

معرفی:

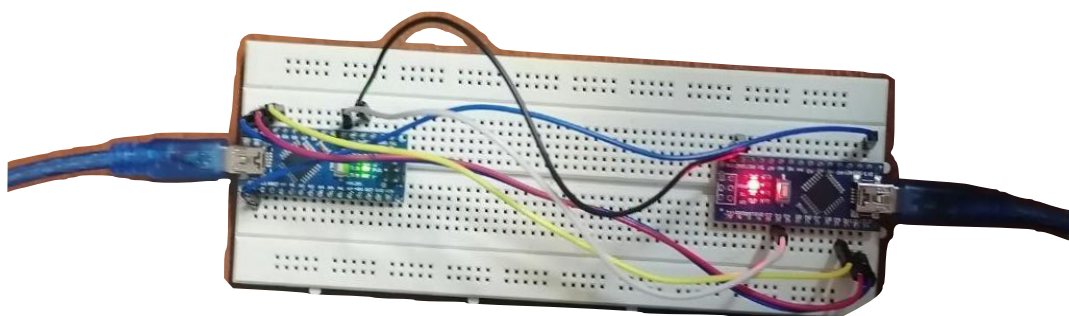
پروتکل SPI (Serial Peripheral Interface) یک روش ارتباطی سریال پرسرعت است که معمولاً برای ارتباط بین میکروکنترلرها و دستگاه‌های جانبی مانند کارت‌های حافظه، نمایشگرهای LCD و مبدل‌های ADC/DAC به کار می‌رود. این پروتکل از چهار خط MISO، MOSI (Master Out Slave In)، SS (Slave Select) و SCK (Serial Clock) برای انتقال داده استفاده می‌کند. SPI نسبت به I2C سرعت بالاتری دارد اما تعداد سیم‌های بیشتری نیاز دارد. این پروتکل معمولاً در مواردی که نیاز به انتقال سریع داده است، مانند پردازش تصویر و ذخیره‌سازی اطلاعات، استفاده می‌شود. ارتباط در SPI به صورت Full Duplex است، به این معنی که داده می‌تواند به‌طور هم‌زمان ارسال و دریافت شود.

ویژگی‌ها:

- استفاده از ۴ سیم (MISO، MOSI، SCK، SS)
- پشتیبانی از سرعت‌های بسیار بالا (تا چندین مگابیت بر ثانیه)
- ساختار Master-Slave ساده

نحوه‌ی اتصال دو برد آردوینو:

- MOSI به MOSI (D۱۱)
- MISO به MISO (D۱۲)
- SCK به SCK (D۱۳)
- SS (CS) به SS (D۱۰)
- GND به GND



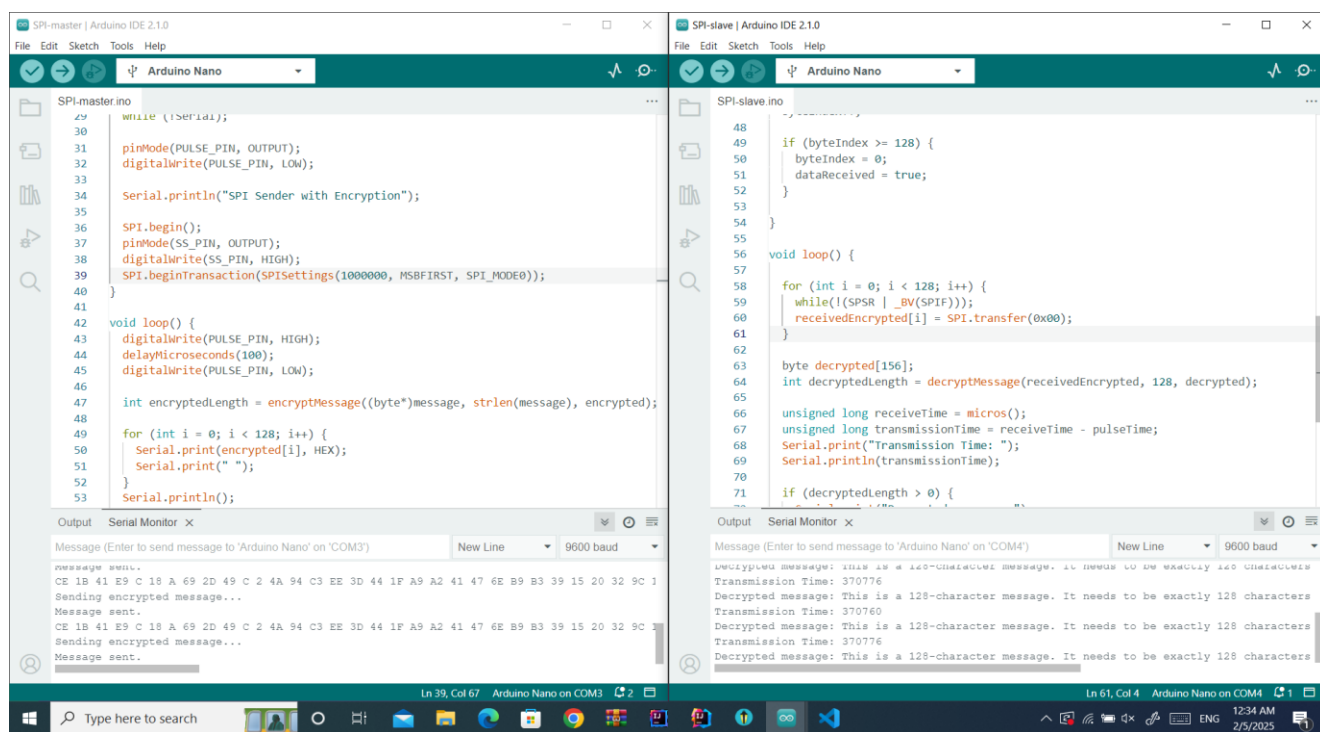
شکل ۳ برد نهایی SPI

مراحل پیاده‌سازی:

1. تنظیم Master برای ارسال داده

2. تنظیم Slave برای دریافت داده

3. تبادل داده با استفاده از کتابخانه SPI.h



شکل ۴ تصاویر خروجی کد SPI

۳. پروتکل UART

معرفی:

پروتکل (UART (Universal Asynchronous Receiver-Transmitter یکی از قدیمی‌ترین روش‌های ارتباطی سریال است که در آن داده‌ها بدون نیاز به کلاک مشترک بین دو دستگاه ارسال و دریافت می‌شوند. در این روش، دستگاه‌های فرستنده و گیرنده باید روی یک نرخ انتقال داده مشخص تنظیم شوند. UART در ارتباطات سریال پایه، ماژول‌های GPS، بلوتوث و ماژول‌های مخابراتی به کار می‌رود. این پروتکل

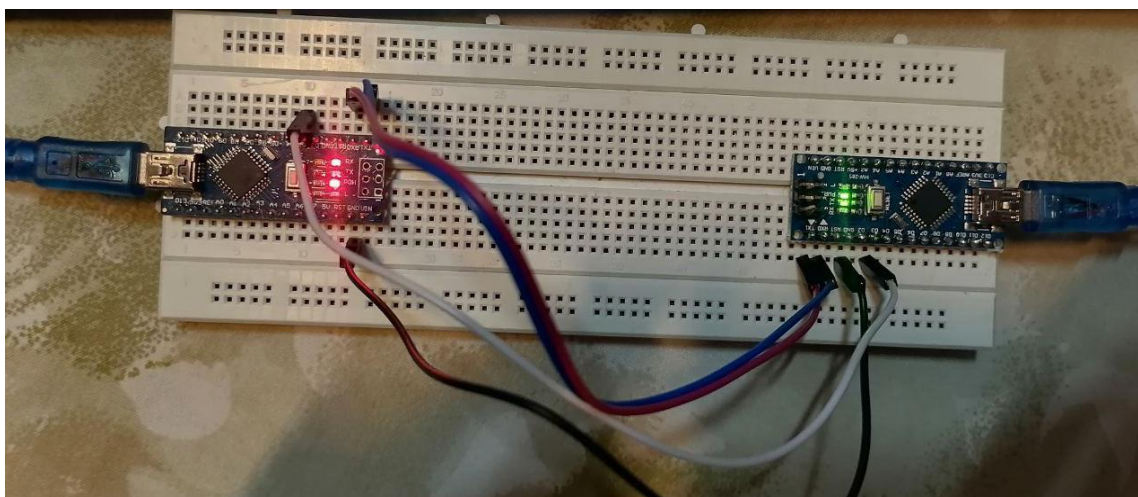
دارای دو خط اصلی TX (Transmit) و RX (Receive) است که داده را ارسال و دریافت می‌کنند. در مقایسه با I2C و SPI، پروتکل UART ساده‌تر است اما نیاز به تنظیم دقیق نرخ انتقال داده دارد.

ویژگی‌ها:

- استفاده از دو سیم TX و RX
- عدم نیاز به سیگنال کلاک
- نرخ تبادل داده قابل تنظیم (۹۶۰۰، ۵۷۶۰۰، ۱۱۵۲۰۰ بیت بر ثانیه)

نحوه‌ی اتصال دو برد آردوینو:

- از TX برد اول به RX برد دوم
- از RX برد اول به TX برد دوم
- GND به GND



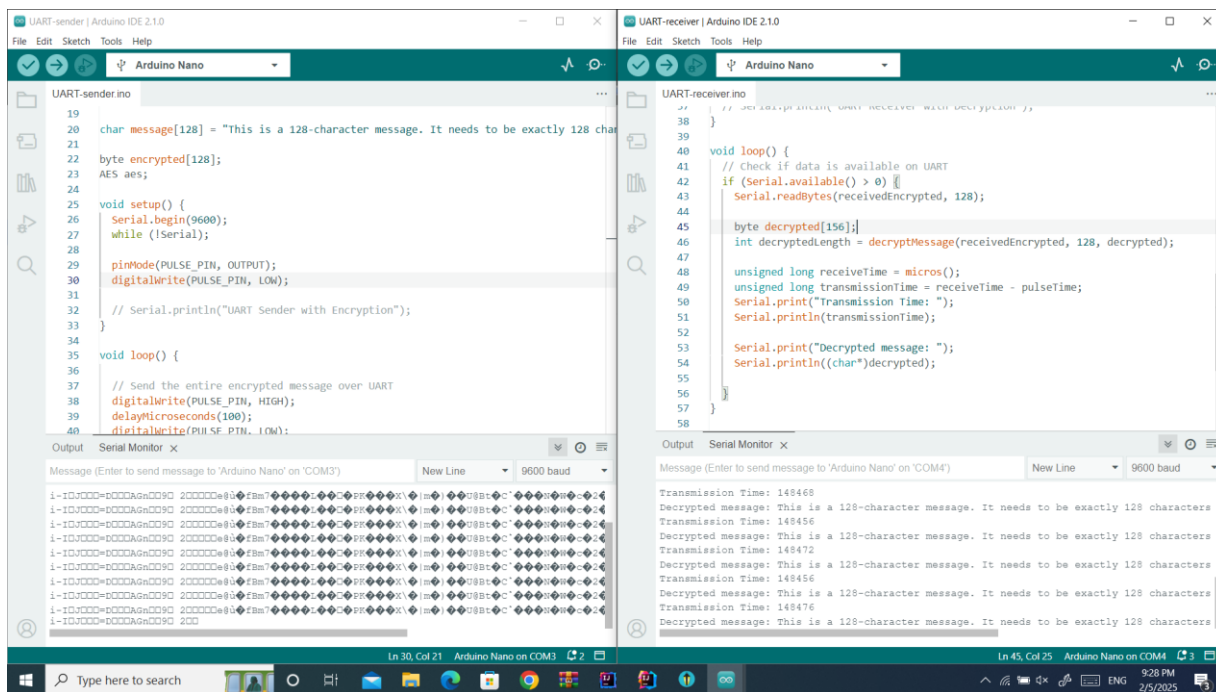
شکل ۵. برد نهایی UART

مراحل پیاده‌سازی:

1. تنظیم سرعت و مقدار داده در دو برد

2. ارسال و دریافت داده با استفاده از `Serial.begin()`

3. نمایش داده‌ها در Serial Monitor



شکل ۶ تصاویر خروجی کد UART

۴. پروتکل Wi-Fi

معرفی:

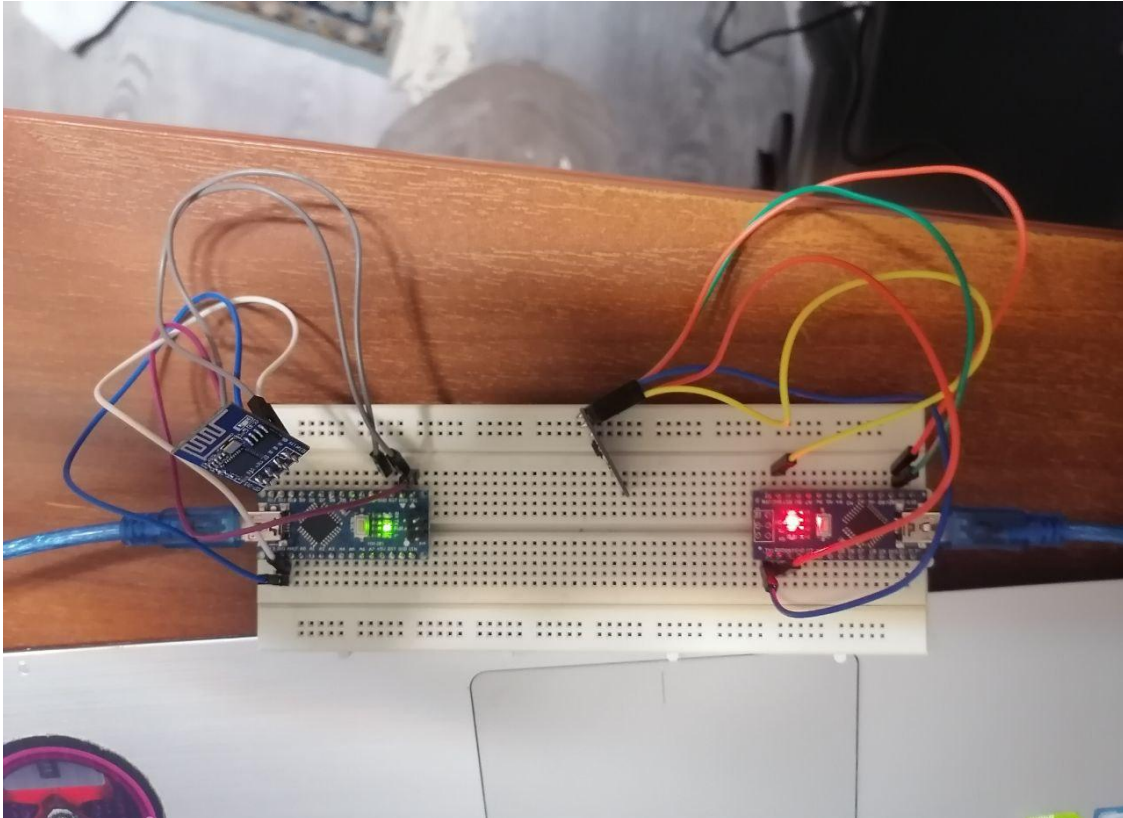
Wi-Fi یک روش ارتباط بی‌سیم است که از طریق امواج رادیویی داده‌ها را ارسال و دریافت می‌کند. در این پروژه، از ماژول ESP8266 برای اتصال بردهای آردوینو به شبکه استفاده می‌شود. این پروتکل در اینترنت اشیا (IoT) و سیستم‌های کنترل از راه دور کاربرد دارد. Wi-Fi امکان ارسال داده به بردهای دیگر و همچنین اتصال به سرورهای اینترنتی را فراهم می‌کند. این پروتکل از استانداردهای ۲.۴ گیگاهرتز و ۵ گیگاهرتز پشتیبانی می‌کند و بسته به شرایط محیطی برد قابل توجهی دارد.

ویژگی‌ها:

- انتقال داده بی‌سیم با برد زیاد
- نرخ انتقال داده بالا (تا چند صد مگابیت بر ثانیه)
- امکان ارتباط بردها با شبکه‌های محلی و اینترنت

نحوه‌ی اتصال دو برد آردوینو:

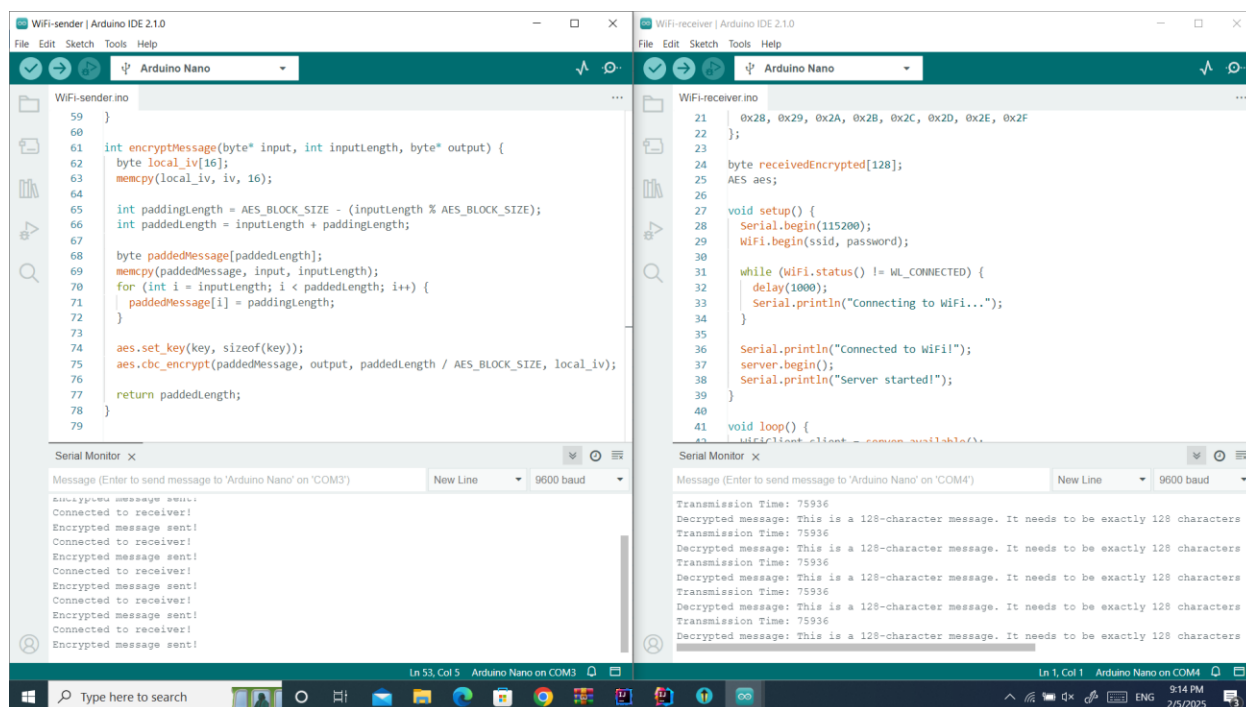
- استفاده از ماژول ESP8266
- برقراری ارتباط یک برد به عنوان سرور و دیگری به عنوان کلاینت



شکل ۷ برد نهایی WIFI

مراحل پیاده‌سازی:

1. تنظیم SSID و Password شبکه
2. راه‌اندازی سرور و کلاینت با استفاده از WiFi.h
3. ارسال و دریافت داده از طریق پروتکل TCP یا UDP



شکل ۸ تصویر خروجی WIFI

۵. پروتکل CAN

معرفی:

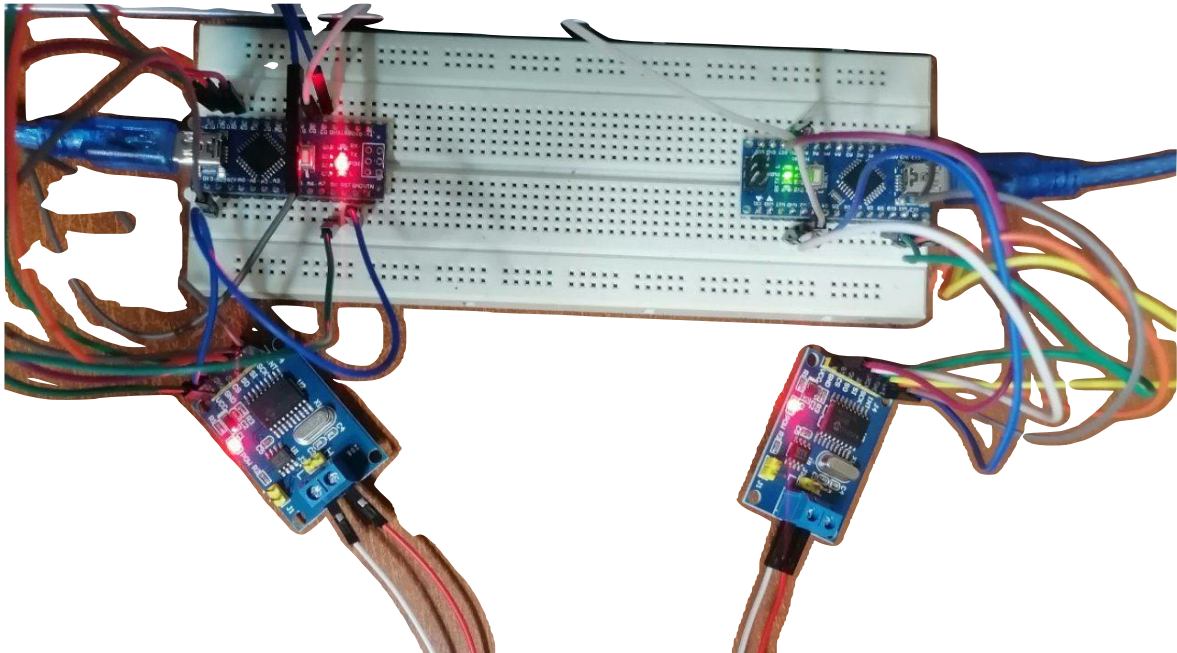
پروتکل CAN (Controller Area Network) یک استاندارد ارتباطی سریال است که در سیستم‌های صنعتی و خودرویی برای تبادل داده بین میکروکنترلرها استفاده می‌شود. این پروتکل به دلیل پایداری بالا، مقاومت در برابر نویز و قابلیت کار در محیط‌های صنعتی پر نویز، کاربرد گسترده‌ای دارد.

ویژگی‌ها:

- ارتباط چند دستگاه بدون نیاز به Master
- مقاوم در برابر نویز
- استفاده از دو سیم CAN_H و CAN_L

نحوه‌ی اتصال دو برد آردوینو:

- استفاده از مژول MCP2515 CAN
- از CAN_H برد اول به CAN_H برد دوم
- از CAN_L برد اول به CAN_L برد دوم
- اتصال پایه های مژول به پین های SPI در آردوینو

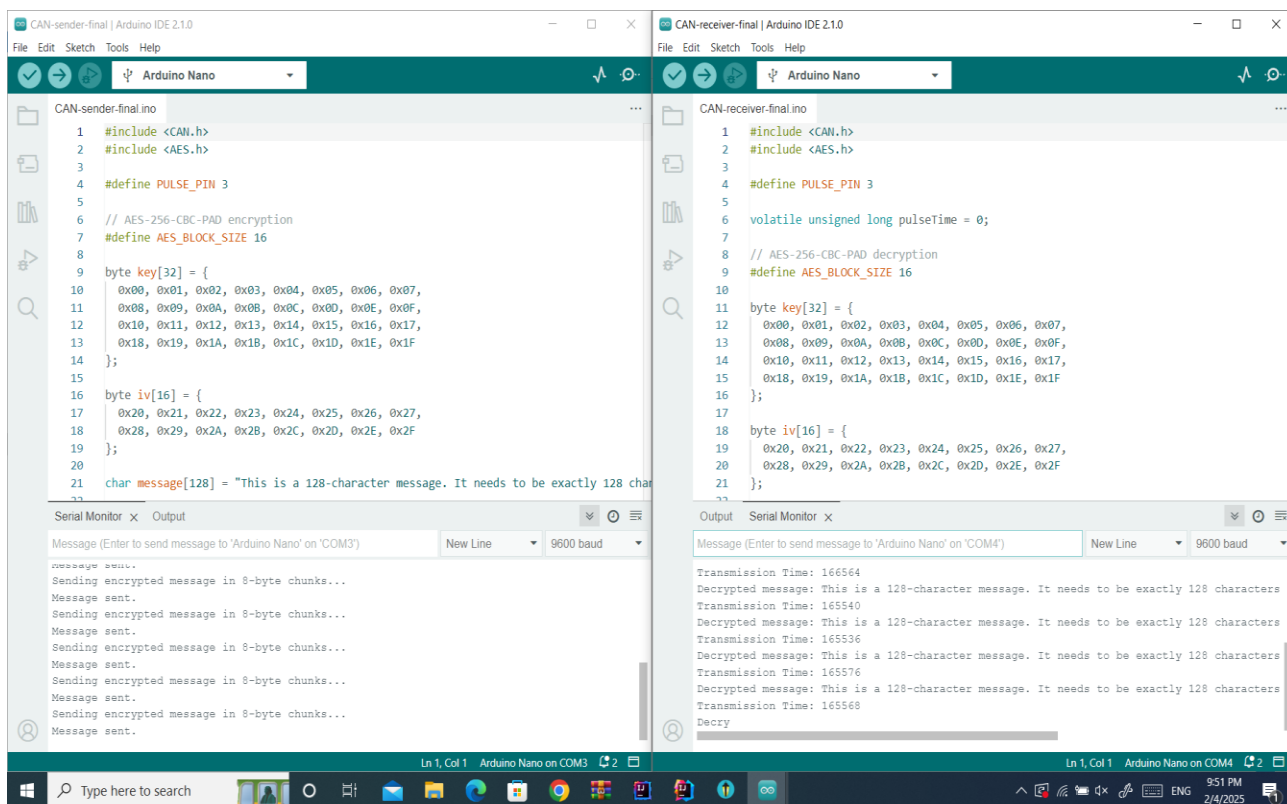


شکل ۹ برد نهایی CAN

مراحل پیاده‌سازی:

1. مقداردهی اولیه مژول CAN
2. ارسال پیام از یک برد و دریافت توسط دیگری

3. پردازش داده‌ها



شکل ۱۰ تصاویر خروجی کد CAN

۶. پروتکل One-Wire

معرفی:

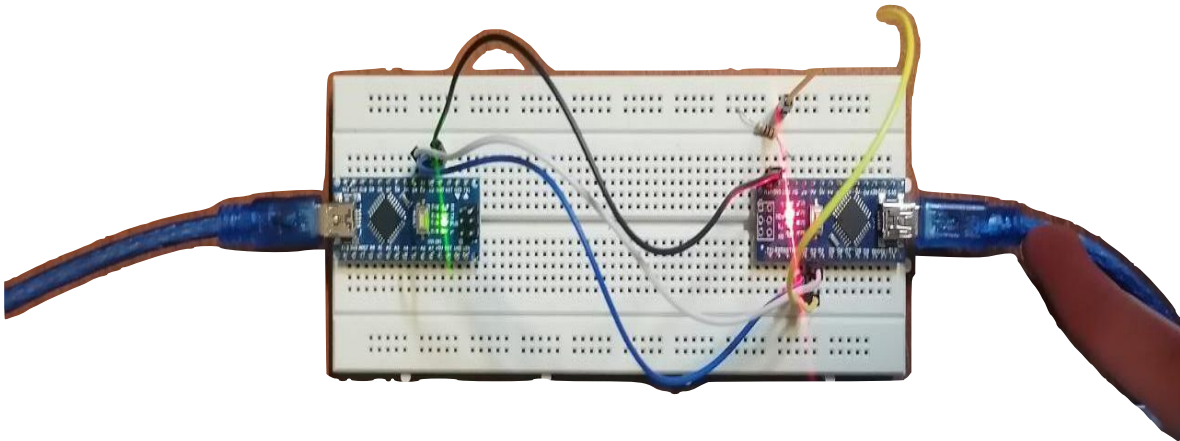
پروتکل One-Wire یک روش ارتباطی سریال ساده و کم‌هزینه است که تنها از یک سیم دیتا به همراه GND استفاده می‌کند. این پروتکل معمولاً در سنسورهای دما مانند DS18B20 و سیستم‌های خواندن RFID به کار می‌رود. One-Wire نیازی به خطوط اضافی مانند کلاک ندارد و به دلیل طراحی ساده، در بسیاری از سنسورها و سیستم‌های نهفته کاربرد دارد.

ویژگی‌ها:

- استفاده از یک سیم برای انتقال داده
- قابلیت اتصال چند دستگاه به یک خط
- سرعت پایین در مقایسه با SPI و I2C

نحوه‌ی اتصال دو برد آردوینو:

- یک سیم دیتا بین دو برد متصل شود
- استفاده از مقاومت پول‌آپ ۴.۷ کیلو اهم



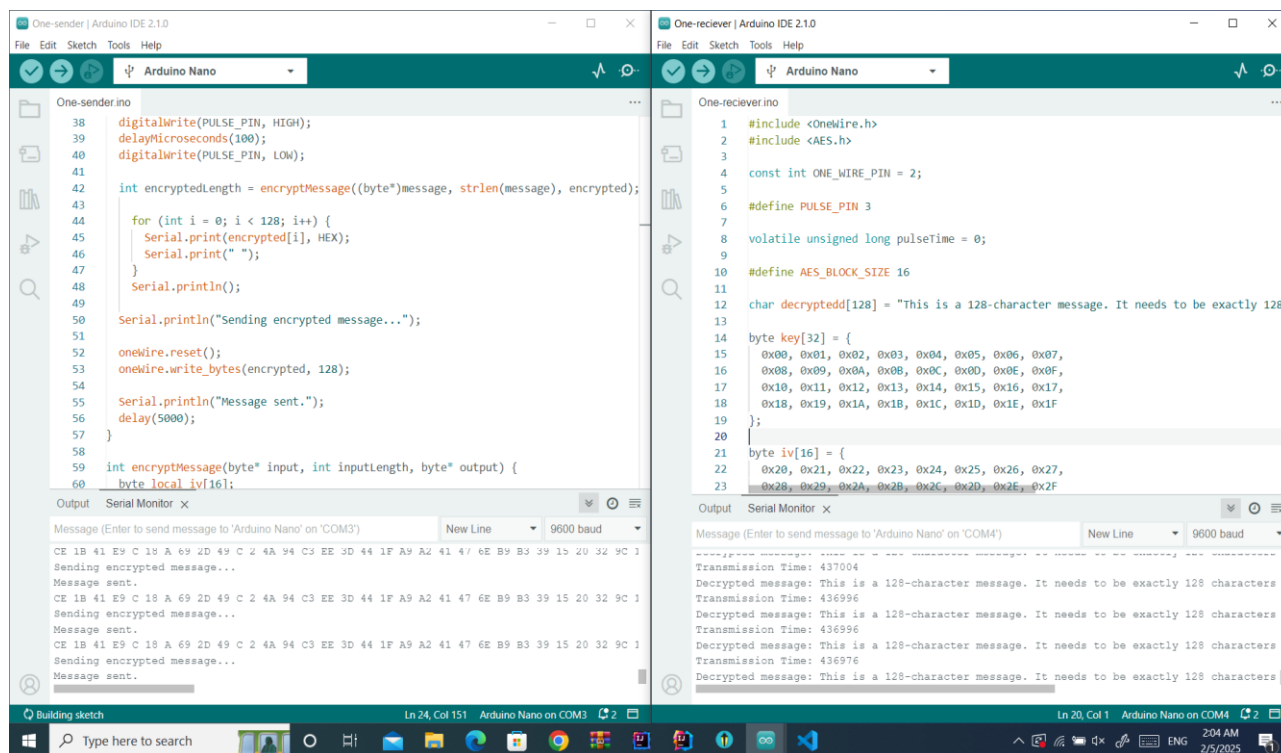
شکل ۱۱ برد نهایی One-Wire

مراحل پیاده‌سازی:

1. مقداردهی اولیه پروتکل با کتابخانه OneWire.h

2. ارسال و دریافت داده از طریق یک خط

3. پردازش اطلاعات دریافت‌شده



شکل ۱۲ تصاویر خروجی کد One-Wire

مقایسه پروتکل‌ها:

Encryption & Decryption:

در تمام از کدها از کتابخانه AESLib و توابع آن برای Encryption و Decryption به صورت مشابهی استفاده شده است. کلیدها نیز برای همه یکسان در نظر گرفته شده است.

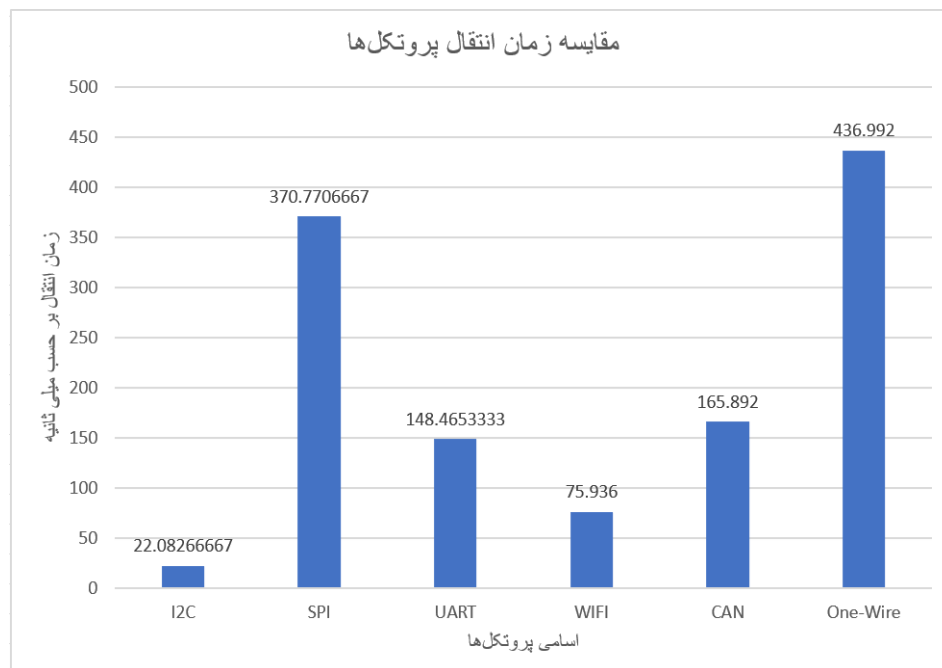
اندازه گیری زمان:

در تمام موارد برای اندازه گیری زمان ارسال از Digital Pin 3 استفاده شده است. به این صورت که فرستنده پیش از شروع کار یک پالس را به واسطه این پین به گیرنده ارسال می‌کند و در آن لحظه گیرنده زمان را ثبت

می‌کند. سپس انتقال پیام انجام می‌شود و در نهایت گیرنده مجدداً زمان را ثبت می‌کند. با محاسبه اختلاف این دو زمان، transitionTime محاسبه می‌شود. (دریافت پالس در گیرنده به صورت event driven پیاده سازی شده است).

مقایسه زمان‌ها:

با توجه به اعداد مشخص شده در تصاویر می‌توان نتیجه گرفت که پروتکل I2C در بین این پروتکل‌ها از سرعت بیشتری برخوردار است. پس از آن WIFI, UART, CAN, SPI, One-Wire قرار دارند. توجه کنید که با توجه به کتابخانه‌های مختلف و توابع و پارامترهای مختلف این زمان‌ها می‌تواند متفاوت باشد. برای مثال در UART با تغییر baudRate برای پورت Serial زمان انتقال پیام تغییر چشمگیری خواهد داشت.



نمودار ۱

نتیجه‌گیری:

در این سند، پروتکل‌های مختلف ارتباطی مورد بررسی قرار گرفتند و نحوه‌ی اتصال دو برد آردوینو برای هر پروتکل شرح داده شد. در ادامه، مقایسه‌ی عملکرد آن‌ها انجام خواهد شد تا مشخص شود کدام روش برای شرایط مختلف بهتر است.