به نام خدا

**ایده کلی**

به منظور طراحی و تولید دستگاهی که قابلیت جمع اوری و ارسال برخی داده های مورد نیاز از وسایل نقلیه را دارد، ایده زیر اجرا شده است که در ادامه این مستند به بررسی ان پرداخته خواهد شد.

سیستم مورد نظر از دو بخش نرم افزار و سخت افزار تشکیل شده است. در بخش سخت افزار چهار قسمت اصلی وجود دارد:

1. مبدل OBD2 to UART
2. سنسورهای محیطی (MPU6050 در نسخه فعلی)
3. ماژول کنترلر (ESP32 Wrover در نسخه فعلی)
4. ماژول GSM – GPS (SIM868 در نسخه فعلی)

در بخش نرم افزار نیز کد نوشته شده برای کنترلر دستگاه مورد بررسی قرار خواهد گرفت.

این سیستم اطلاعات مورد نیاز را توسط اینترفیس های موجود جمع اوری کرده و پس از پردازش توسط کنترلر از طریق ماژول GSM به سرور ارسال میکند. در نسخه فعلی اطلاعات زیر قابل دریافت است

1. اطلاعات وسیله نقلیه شامل:
   * دمای موتور – ولتاژ باتری – دور موتور – سرعت – سطح سوخت
2. اطلاعات محیطی شمال:
   * شتاب در سه محور - چرخش در سه محور - دمای محیط
3. اطلاعات مکانی شمال:
   * طول و عرض جغرافیایی

**سخت افزار**

مبدل OBD2 to UART

این ماژول متشکل از سه بخش اصلی تامین تغذیه، کنترلر اصلی و Tranciever است. در بخش کنترلر اصلی از چیپست stn1110 استفاده شده است که وظیفه تبدیل پروتکل های OBD به UART را بر عهده دارد.

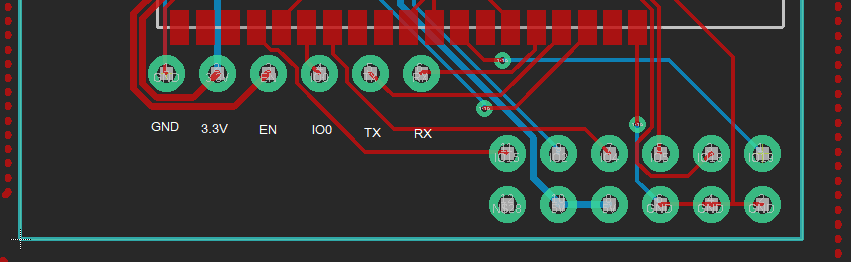
ماژول MPU6050

در آخرین نسخه از طراحی (ng)، این ماژول به صورت on board روی بورد مبدل OBD2 to UART طراحی شده است اما عملکرد صحیح ان مشاهده نشد. به همین دلیل با استفاده از یک کانکتور ماژول MPU6050 به بورد مبدل متصل شده است.

ماژول ESP32

نسخه مورد استفاده در دستگاه ESP32-WROVER است که به شرح زیر قابل برنامه ریزی است:

با استفاده از یک مبدل USB to UART و با سیم بندی زیر میتوان این ماژول را برنامه ریزی کرد



|  |  |
| --- | --- |
| ESP32 Module | CH340 USB to UART Module |
| GND | GND |
| 3.3V | 3.3V |
| TX | RX |
| RX | TX |

پس از انجام سیم بندی فوق، با استفاده از دو پایه EN و IO0 از ماژول ESP32، ماژول را میتوان به دو مود Download و Normal برد. به منظور برنامه ریزی، ماژول بایستی در مد Download قراربگیرد. پروسه برنامه ریزی به ترتیب زیر است:

1- قبل از اتصال مبدل USB to UART به PC، پایه EN را به HIGH کنید

2- مبدل را به PC متصل کنید. پایه IO0 را LOW کنید.

3- پایه EN را قطع و وصل کنید تا ماژول ریستارت شود و در مود Download عمل کند.

4- از طریق IDE آردینو عمل program را انجام دهید

5- پس از اتمام پایه IO0 را قطع و بار دیگر ماژول را ریستارت کنید.

4- ماژول SIM868

یک نسخه از این ماژول طراحی شده است اما به مرحله تولید و تست نرسیده است. در حال حاضر به دلیل حساسیت های طراحی این ماژول، از ماژول اماده استفاده شده است. راهنمای آپدیت فرم ویر این ماژول در مسیر مرتبط در گیت لب موجود است.

**نرم افزار**

در بخش نرم افزار، از IDE آردیونو جهت توسعه استفاده شده است. به منظور توسعه فرمویر برای ماژول ESP32 Wrover لازم است تا از طریق منوی tools ماژول مورد نظر را انتخاب کرد. برای این منظور از طریق منوی Tools->Board، آیتم ESP32 Wrover Kit All Versions را باید انتخاب کرد

لیست کتابخانه های استفاده شده

|  |  |
| --- | --- |
| **کتابخانه** | **مورد استفاده** |
| LiquidCrystal.h | راه اندازی LCD کارکتری جهت دیباگ دستگاه |
| SoftwareSerial.h | راه اندازی پین های نرم افزاری پروتکل سریال |
| Adafruit\_MPU6050.h | راه اندازی ماژول MPU6050 |
| Adafruit\_Sensor.h |
| Wire.h | پروتکل I2C مورد استفاده ماژول MPU6050 |
| ArduinoJson.h | آماده سازی دیتای جمع اوری شده به فرمت JSON |

لیست توابع استفاده شده

|  |  |
| --- | --- |
| **تابع** | **عملکرد** |
| String obdResponsePurify(String input) | حذف کرکترهای اضافه از جواب دریافتی OBD2 |
| String readSerialData(bool keepOK, int device) | خواندن پورت سریال با توجه به نوع ماژول (sim868 یا OBD2) و نگه داشتن کلمه کلیدی OK یا خیر |
| String certainExploder(String input, int number) | جدا کردن بخشی از پاسخ دریافتی. مورد استفاده در جداسازی طول و عرض جغرافیایی از رشته دریافتی ماژول sim868 |
| void logger(int x, int y, int dly, bool needsClear, String msg) | نمایش پیامهای دیباگ روی LCD |
| void mayday() | راه انداز مجدد سیستم در صورت بروز خطا |
| int mpuSetup() | راه اندازی ماژول MPU6050 |
| int simSetup() | راه اندازی ماژول sim868 |
| int obdSetup() | راه اندازی ماژول OBD |
| int httpInit() | راه اندازی HTTP session |
| int httpSend(String outputData) | ارسال درخواست |
| int httpTerm() | بستن HTTP session |
| int fillLocationData() | دریافت اطلاعات مکان از sim868 |
| int fillVechData() | دریافت اطلاعات وسیله نقیله از OBD2 |
| int fillSensorsData() | دریافت اطلاعات از ماژول MPU6050 |
| void setup() | راه اندازی اولیه سیستم |
| void loop() | حلقه اصلی |