AKS

1. Başla
2. Seri haberleşme ve SPI protokolü başlat.
3. Nextion ekranını başlat ve MCP2515 CAN modülünü resetle, hızı ayarla, normal moda geçir.
4. Butonlar için dijital pinleri giriş modunda tanımla (INPUT\_PULLUP).
5. Nextion ekranındaki bileşenleri (Numaralar, Resimler, Gauge, Progress Bar) oluştur.
6. CAN modülü ile kullanılacak veriler için yapı (struct) ve değişkenleri tanımla.
7. Her bir butonun durumunu kontrol et: HIGH: İlgili durumu bildir (örn. "Butona basınız.") ve Nextion ekranında uygun resmi göster. LOW: İlgili durumu bildir (örn. "Buton basıldı.") ve Nextion ekranında farklı bir resmi göster.
8. MCP2515 modülü üzerinden gelen CAN mesajlarını oku:
9. Mesaj ID'sine göre işlemler yap: Eğer ID 0x101 ise: Batarya sıcaklığını oku, dereceye çevir ve Nextion ekranındaki sıcaklık bileşenine gönder.
10. Eğer ID 0x100 ise: Hücre voltajlarını hesapla: Hücrelerin toplam, minimum ve maksimum voltajlarını belirle. Bu değerleri Nextion ekranına yazdır.
11. Eğer ID 0x102 ise: Akım bilgisini oku, çevir ve ekrana gönder.
12. Eğer ID 0x103 ise: Batarya doluluk oranını (SoC) hesapla hem yüzde hem de Progress Bar ile ekrana göster.
13. 0x201: Harici bir butonun durumunu kontrol et ve ilgili resmi göster.
14. 0x200: Motor hızını hesapla: Negatif hız değerlerini 0 yap, maksimum hız değerini 180 ile sınırla. Hızı Nextion ekranındaki Gauge ve Number bileşenlerine gönder.
15. ID bunlardan biri değilse "Bilinmeyen ID" mesajını yazdır.
16. Bitir.

metin, el yazısı, diyagram, siyah beyaz içeren bir resim

Açıklama otomatik olarak oluşturuldu

BMS Gönderici

1. Başla
2. MCP2515 CAN modülü SPI üzerinden resetle.
3. CAN hız ayarı ve normal mod yapılır.
4. Sıcaklık sensörü için OneWire protokolü başlat.
5. Seri haberleşme başlat. Hata varsa bilgi mesajlarını yazdır.
6. Voltaj için analog pinlerden okuma yap. Analog okunan değerler 0-1023 arasında bir değeri temsil eder. Bu değerler 0-5V aralığına çevir.
7. Toplam voltaj hesapla.
8. Minimum ve maksimum voltaj bul.
9. Toplam voltaj, minimum ve maksimum voltaj ekrana yazdır.
10. Hücre voltajlarını 100 ile çarp tam sayı haline getir. Her bir voltaj 2 byte halinde CAN mesajına yaz.
11. Mesaj ID belirle.
12. Mesaj gönderimi başarılıysa ilgili mesajları ekrana yazdır. Başarısızsa da ilgili mesajı ekrana yazdır.
13. Sıcaklık sensörü okumasını gerçekleştir.
14. Sensör bağlantısı kopmuşsa hata mesajı yazdır ve 28. adıma git.
15. Sıcaklık verisini CAN mesajına dönüştürmek için 100 ile çarparak tam sayı haline getir. 2 byetlık veri olarak CAN mesajına yaz.
16. Mesaj ID tanımla.
17. Mesaj gönderimi başarılıysa sıcaklık değerini ekrana yazdır. Başarısızsa da ilgili hata mesajını ekrana yazdır.
18. Akım için pinden n =1000 kez okuma yap.
19. Okunan değerin ortalaması alınıp akım değerini hesapla. Akım =( (Vcc/2) –(Vcc\* analog değer)) / sensitivity
20. Akım değeri 1000 ile çarparak tam sayı haline getir. Her bir voltaj 2 byte halinde can mesajına yaz.
21. Mesaj ID belirle.
22. Mesaj gönderimi başarılıysa akım değerini ekrana yazdır.
23. SoC hesaplaması için pil voltajına bağlı olarak doluluk oranı hesaplanır. SoC = ((Toplam Voltaj – 12.0) / (16.8 – 12.0)) \* 100
24. Eğer toplam voltaj 12V veya daha azsa SoC= 0, 16.8V veya üzerindeyse SoC= 100 yazdır.
25. SoC değeri 100 ile çarpılarak tam sayı haline getirilir. 2 bytelık veri olarak CAN mesajına yaz.
26. Mesaj gönderimi başarılıysa SoC değerini ekrana yazdır.
27. Her 2 saniyede bir tekrar ederek ekrana yazdır.
28. Bitir.

metin, ekran görüntüsü, tasarım, çizgi içeren bir resim

Açıklama otomatik olarak oluşturuldumetin, siyah beyaz, diyagram, makbuz içeren bir resim

Açıklama otomatik olarak oluşturuldu

Şarj aleti gönderici

1. Başla.
2. Düğme pinini ata.
3. Seri haberleşme ve SPI protokolü başlat.
4. MCP2515 CAN modülünü resetle, hızı ayarla, normal moda geçir.
5. Hata durumu varsa uygun mesajı ekrana yazdır.
6. Başlatma işlemleri başarılıysa: "Başlatıldı" mesajını ekrana yazdır.
7. Düğme durumu okuma işlemleri için değişkenleri ata.
8. Düğme durumunu oku: (HIGH veya LOW)
9. CAN mesajı hazırla: ID ve mesaj uzunluğu belirle. Düğme durumunu mesajın veri alanına yaz.
10. CAN mesajını MCP2515 üzerinden gönder.
11. Gönderim başarılıysa: "Gönderildi" mesajı ve düğme durumu seri porta yazdır. Başarısızsa: "Hata: Mesaj gönderilemedi" mesajını seri porta yazdır.
12. Her döngü arasında 1 saniye bekle ve işlem tekrar et.
13. Bitir.

diyagram, metin, taslak, teknik çizim içeren bir resim

Açıklama otomatik olarak oluşturuldu

İzolasyon gönderici

VOLTAJ ALIMI VE İZOLASYON DİRENCİ GÖNDERİMİ KODU ALGORİTMASI

1. Başla.
2. Seri haberleşme ve SPI protokolü başlat.
3. MCP2515 CAN modülü resetle, hızını ayarla ve normal moda geçir, hata durumunda mesaj ekrana yazdır.
4. Tüm başlatma işlemleri başarılıysa "Başlatıldı" mesajı ekrana yazdır.
5. direnc\_gonderimi değişkeni tanımla.
6. Eğer direnc\_gonderimi true ise izolasyon direnci olarak sabit bir değer tanımla ve bu değeri iki byte’a böl. CAN mesajına ekle. CAN mesajının ID’sini belirle ve CAN üzerinden gönder.
7. Gönderim başarılıysa "Direnç değeri gönderildi" mesajı ve değer ekrana yazdır, aksi takdirde hata mesajı yazdır.
8. Eğer direnc\_gonderimi false ise CAN mesajını oku ve gelen mesajın ID’si belirlenen değer ise voltaj değerini al.
9. Voltaj değerini ekrana yazdır.
10. direnc\_gonderimi tekrar true yap.
11. Her döngü arasında 1 saniye bekle ve işlem tekrar et.
12. Bitir.

metin, diyagram, ekran görüntüsü, paralel içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Telemetri Algoritma**

1. **Başlat**.
2. **SPI İletişimini Başlat**: SPI iletişim protokolü başlatılır.
3. **MCP2515 Modülünü resetle**.
4. **MCP2515 Bitrate ve Çalışma Modunu Ayarla**:

* Bitrate olarak 500 Kbps ve çalışma modu olarak Normal Mod ayarlanır.

5.**Seri İletişimi Başlat**:Seri haberleşme başlatılır ve bir bilgi mesajı yazdırılır: "CAN Alıcı Başladı..."

6.**Ana Döngü Başlat**:Program döngüye girer.

7.**CAN Mesajı Alımını Kontrol Et**:

* MCP2515 modülünden gelen mesaj olup olmadığını kontrol et.
  + Eğer mesaj **yoksa**, döngünün başına dön.
  + Eğer mesaj **varsa**, bir sonraki adıma geç.

8.**Mesaj ID'sini Kontrol Et**:

* Gelen mesajın ID'si 0x202 mi?
  + **Hayır**: Mesaj ID'si 0x202 değilse döngünün başına dön.
  + Evet: Mesaj ID'si 0x202 ise bir sonraki adıma geç.

**9.** **Veriyi Ayrıştır**:

* Mesajın verileri ayrıştırılır:
  + Hız (1. bayt)
  + SoC (2. bayt)
  + Sıcaklık (3. bayt)
  + Kalan Enerji (4. ve 5. bayt birleşimi)

10. **Veriyi Yazdır**:

* Ayrıştırılan veriler Seri Monitör'e yazdırılır: Hız, SoC, sıcaklık, kalan enerji

11.**Döngüye Geri Dön**:

* Yeni mesajları kontrol etmek için döngünün başına dön.

12. **Bitir** (Program döngüden çıkmaz, sonsuz döngü devam eder).

**Telemetri – Aks Arası Haberleşme İsterleri**

**1. Veri Toplama ve Aktarım İsterleri**

* **Zorunlu Veri Türleri:**
  + Araç hızı (km/h)
  + Batarya paketi sıcaklığı (°C)
  + Toplam batarya gerilimi (V)
  + Kalan enerji miktarı (Wh)
* **Veri Doğrulama:**
  + Toplanan verilerin algoritmalarla kontrol edilerek doğruluğunun onaylanması.
* **Zaman Damgası:**
  + Verilerin araç çalışmaya başladığı andan itibaren milisaniye bazında damgalanması.
  + Kayıtlar arasındaki zaman farkının **5 saniyeyi geçmemesi**.

**2. Veri Güvenliği ve Kayıp Yönetimi**

* **Sinyal Kesintisi Yönetimi:**
  + Sinyal kesildiğinde en az **60 saniyelik veri paketi** geçici bellekte tutulmalı.
  + Bağlantı yeniden sağlandığında bu veriler izleme merkezine tekrar gönderilmelidir.
* **Kayıt Dosyası Yönetimi:**
  + Her araç başlatıldığında eşsiz bir kayıt dosyası oluşturulması.
  + Kayıt dosyalarının üzerine yazma hatalarının önlenmesi.

**3. Haberleşme İsterleri**

* **Aktarım Menzili:**
  + Haberleşme için kullanılan kartların en az **1000 m menzile** sahip olması.
* **Bağlantı Kararlılığı:**
  + Veri aktarımının kesintisiz ve güvenilir bir şekilde sağlanması.

**4. Sistemin Genel İsterleri**

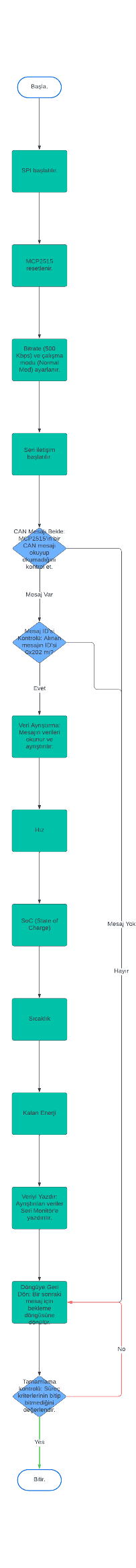
* **Geriye Dönük Uyumluluk:**
  + Sistemin geçmiş verileri saklayarak gerektiğinde yeniden işleyebilmesi.
* **Gecikme Toleransı:**
  + Haberleşme sırasında minimum gecikme (latency) ile veri iletimi yapılması.
* **Enerji Verimliliği:**
  + Telemetri sisteminin düşük güç tüketimiyle çalışabilmesi.

**5. Donanım ve Yazılım Uyumluluğu**

* **Donanım Gereksinimleri:**
  + Yüksek dayanıklılığa sahip kartların kullanımı (çevresel etkilere dayanıklı).
* **Yazılım Gereksinimleri:**
  + Verilerin sıkıştırılması ve paketlenmesi için etkili algoritmaların kullanılması.

**6. Hata Tespiti ve Çözümleme**

* **Hata Algılama:**
  + Veri doğrulama sırasında tespit edilen hataların loglanması.
  + Hataların yeniden iletim için işaretlenmesi.
* **Durum İzleme:**
  + Telemetri sisteminin performans ve hata durumlarının anlık olarak izlenebilmesi.



**Sensör Algoritma:**

1. **Başlat**: Program döngüsü (loop) başlatılır.
2. **CAN Mesajı Kontrolü**: MCP2515 modülü üzerinden yeni bir mesajın varlığı kontrol edilir:
   * + **Eğer mesaj yoksa:**1 saniye beklenir ve döngü yeniden başlatılır.
     + **Eğer mesaj varsa:** Mesajın can\_id değeri okunur.
3. **Mesaj ID Kontrolü ve İşleme**
   * can\_id değerine göre mesajın içeriği işlenir:
     + **Eğer can\_id 0x123 ise:**
       - İzolasyon direnci hesaplanır:
         * Voltaj değeri (data[0] << 8 | data[1]) işlemiyle alınır.
         * Voltaj değeri 100'e bölünerek ölçeklenir.
         * Sonuç seri porta yazdırılır.
     + **Eğer can\_id 0x101 ise:**
       - Sıcaklık hesaplanır:
         * (data[0] << 8 | data[1]) / 100.0 formülüyle sıcaklık hesaplanır.
         * Sıcaklık seri porta yazdırılır.
     + **Eğer can\_id 0x100 ise:**
       - Voltaj hesaplanır:
         * (data[0] << 8 | data[1]) / 100.0 formülüyle voltaj hesaplanır.
         * Voltaj seri porta yazdırılır.
     + **Eğer can\_id 0x102 ise:**
       - Akım hesaplanır:
         * (data[0] << 8 | data[1]) \* 0.001 formülüyle akım hesaplanır.
         * Akım seri porta yazdırılır.
     + **Eğer can\_id 0x103 ise:**
       - Şarj durumu (SoC) hesaplanır:
         * (data[0] << 8 | data[1]) / 100.0 formülüyle SoC hesaplanır.
         * SoC seri porta yazdırılır.
     + **Eğer can\_id 0x201 ise:**
       - Şarj durumu belirlenir:
         * Şarj durumu bayrağı (data[0]) kontrol edilir.
         * Sonuç seri porta yazdırılır.
     + **Eğer can\_id 0x200 ise:**
       - Motor hızı hesaplanır:
         * (data[0] << 8 | data[1]) işlemiyle hız değeri alınır.
         * Hız seri porta yazdırılır.
     + **Diğer ID'ler:**
       - "Bilinmeyen ID" mesajı yazdırılır.
4. **Sonlandırma** İşlem tamamlandığında 1 saniye beklenir ve döngü yeniden başlatılır.

|  |  |
| --- | --- |
| Pin Numarası | Bağlantı/Buton |
| 24 | Şarj |
| 26 | Far |
| 28 | Silecek |
| 30 | Flaşör |
| 32 | Sağ Sinyal |
| 34 | Sol Sinyal |
| 36 | Fren Stop |

metin, diyagram, ekran görüntüsü, çizgi içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Motor Aks Akış Diyagramının Algoritması:**

1. **Başlat**: Program çalışmaya başlar.
2. **Setup İşlemleri**
   * Seri iletişim, SPI, MCP2515 ve DS1302 gibi gerekli donanım ve protokol ayarları yapılır.
   * Sensör pini giriş olarak ayarlanır ve kesme (interrupt) fonksiyonu atanır.
3. **Ana Döngü (Loop):**
   * Her döngüde geçen süre kontrol edilir.
   * Eğer 1 saniye geçtiyse:  
     a. Hız hesaplanır.  
     b. RTC'den (Gerçek Zaman Saati) saat, dakika ve saniye bilgisi alınır.  
     c. CAN mesajı oluşturulur ve gönderilir.  
     d. Hız ve zaman bilgisi seri monitöre yazdırılır.
4. **Kontrol Et:** Döngüye tekrar başlanır (Başlangıca dön).
5. **Bitiş:** Program sonlandırılır (gerektiğinde).

diyagram, taslak içeren bir resim

Açıklama otomatik olarak oluşturuldu