HENÜZ DENENMEDİ, HTML DAHİL

Konum bilgisinin hem "ön" hem de "arka" gibi birden fazla değeri aynı anda temsil edebilmesi için, konum bilgisini bir bit maskesi olarak kullanabiliriz. Her bir bit, belirli bir konumu temsil eder. Örneğin:

* 0x01 (1): Ön
* 0x02 (2): Arka
* 0x04 (4): Sağ
* 0x08 (8): Sol
* 0x10 (16): Üst
* 0x20 (32): Alt
* 0x40 (64): Merkez

Bu şekilde, birden fazla konum bilgisini birleştirerek tek bir byte içinde temsil edebiliriz. Örneğin, hem ön hem de arka konumu temsil etmek için 0x01 | 0x02 = 0x03 değerini kullanabiliriz.

**Bit Maskesi Kullanımı:**

Konum bilgisi artık bir bit maskesi olarak temsil ediliyor. Örneğin, 0x01 | 0x02 = 0x03 hem ön hem de arka konumu temsil eder.

decode\_location fonksiyonu, bu bit maskesini çözümleyerek hangi konumların aktif olduğunu belirler.

**Çıktı Formatı:**

Birden fazla konum bilgisi varsa, bunlar virgülle ayrılarak ekrana yazdırılır. Örneğin: BNO1 sensörü (Ön, Arka): Çalışıyor.

**Esneklik:**

Bu yöntem, gelecekte yeni konumlar eklemek istediğinizde kolayca genişletilebilir. Yeni bir konum eklemek için sadece yeni bir bit tanımlamanız yeterlidir.

Eğer sensor\_data şu şekilde tanımlanmışsa:

sensor\_data = {

"BNO1": {"location": 0x01 | 0x02, "error": 0}, # Hem ön hem arka

"BMP1": {"location": 0x02, "error": 0}, # Sadece arka

"AMG1": {"location": 0x03, "error": 1}, # Hata durumu (ön ve arka)

"IR1": {"location": 0x04, "error": 0}, # Sadece sağ

"CAM1": {"location": 0x05, "error": 0} # Ön ve sağ (0x01 | 0x04)

}

Alıcı tarafında şu şekilde bir çıktı alırsınız:

BNO1 sensörü (Ön, Arka): Çalışıyor.

BMP1 sensörü (Arka): Çalışıyor.

Hata: AMG1 sensörü (Ön, Arka) veri göndermiyor.

IR1 sensörü (Sağ): Çalışıyor.

CAM1 sensörü (Ön, Sağ): Çalışıyor.