AIRHMI
SmartIO Modbus 8R4A4D

Genel Bakış

SmartIO Modbus 8R4A4D, endüstriyel kontrol uygulamaları için tasarlanmış, Modbus protokolü üzerinden kontrol edilebilen, esnek ve güçlü bir giriş/çıkış kontrol kartıdır. Kart, 4 analog giriş, 4 dijital giriş ve 8 röle çıkışı ile çok çeşitli cihazların izlenmesini ve kontrol edilmesini sağlar.

Özellikler

1. **Analog Girişler (4 adet)**:
    - 0-30V arasında giriş sinyallerini algılayabilir.
    - Sensörlerin, potansiyometrelerin veya diğer analog cihazların bağlantısı için uygundur.
2. **Dijital Girişler (4 adet)**:
    - 5V sinyallerle uyumlu.
    - Limit switch, buton veya diğer dijital cihazların bağlantısı için idealdir.
3. **Röle Çıkışlar (8 adet)**:
    - Her röle,5A/250V AC veya 5A/24V DC yük taşıyabilir.
    - Motorlar, lambalar veya diğer yüklerin kontrolü için uygundur.
4. **Modbus RTU Desteği**:
    - RS485 üzerinden haberleşme.
    - 300, 1200, 2400, 4800, 9600, 14400,19200, 38400, 57600, 115200, 230400,460800, 921600 baud hızlarında çalışabilme.
    - Modbus protokolüne uygun tüm cihazlarla uyumluluk.
5. **Çalışma Gerilimi**:
    - Besleme: 24V DC.(12V versiyon için iletişime geçiniz.)
    - Güç tüketimi: Maksimum 5W.

Uygulama Alanları

- Endüstriyel otomasyon sistemleri.
- Akıllı ev uygulamaları.
- HVAC (Isıtma, Soğutma ve Havalandırma) sistemleri.
- Pompa ve motor kontrolü.
- Enerji izleme ve kontrol projeleri.

Donanım Bağlantıları

1. **Giriş ve Çıkış Terminalleri**:
    - Analog girişler: AI1, AI2, AI3, AI4.
    - Dijital girişler: DI1, DI2, DI3, DI4.
    - Röle çıkışları: R1, R2, R3... R8.
2. **Modbus Bağlantısı**:
    - RS485 terminalleri: A(+) ve B(-).
    - Bağlantı şeması: Modbus master cihazına uygun şekilde bağlayın.
3. **Güç Beslemesi**:
    - Kart üzerindeki güç terminallerine doğru polaritede bağlantı yapınız. Güç biriminde korumlar mevcuttur. Ters bağlantıya karşı koruma vardır.

Yazılım ve Konfigürasyon

- **Modbus Adresleme**:
  - Kart, 1-247 arasında bir Modbus cihaz adresine sahiptir.
  - Adres yazılım aracılığıyla ayarlanabilir.
- **Haberleşme Parametreleri**:
  - Standart ayar: 9600 baud, 8N1.
  - Parametreler Modbus üzerinden değiştirilebilir.
- **Kontrol Yazılımı**:
  - Kart, standart Modbus RTU uyumlu herhangi bir SCADA veya HMI yazılımı ile entegre edilebilir. AirHMI ile kullanım için örnek uygulamalar mevcuttur.

LED Göstergeleri

- **Power** : Kartın beslemesinin aktif olduğunu gösterir.
- **Röle LED'leri**: Aktif olan röleler için ışık verir.

Kurulum Talimatları

1. Kartınızı sabit bir yüzeye monte edin.
2. Giriş ve çıkış bağlantılarını şemaya uygun şekilde yapın.
3. Besleme bağlantısını yaparak kartı çalıştırın.
4. Modbus master cihazınızı bağlayarak gerekli konfigürasyonları gerçekleştirin.

Güvenlik Uyarıları

- Yüksek gerilimli cihazlar bağlanırken dikkatli olun.
- Yanlış bağlantılar kartın hasar görmesine neden olabilir.
- Besleme voltajını her zaman belirtilen aralıkta tutun.

Teknik Destek

Herhangi bir sorun veya destek ihtiyacınızda lütfen Teknik Destek Ekibimizle iletişime geçin.

https://tr.airhmi.com/forum

Tek Röleyi Kontrol Etme

Bu komut, bir röleyi kontrol etmek için Modbus RTU protokolü kullanılarak hazırlanmıştır. Komutun detaylarını adım adım açıklayalım:

Komut Yapısı:

01 05 00 00 FF 00 8C 3A

Her byte (sekizli veri birimi) belirli bir anlam taşır. detaylı açıklaması:

Byte	Anlamı	Açıklama
01	Cihaz Adresi	- 0x00: Yayın adresi (broadcast). - 0x01-0xFF: Belirli bir cihaza adresleme yapılır.
05	Komut Kodu	Röle kontrol komutunu belirtir (05 Command).
00 00	Röle Adresi	Kontrol edilen rölenin adresini temsil eder. Adresler 0x0000 ile 0x0008 arasında olabilir.
FF 00	Kontrol Komutu	Röleye uygulanacak işlemi belirtir: - 0xFF00: Röleyi aç (aktif hale getir). - 0x0000: Röleyi kapat (pasif hale getir). - 0x5500: Röleyi ters çevir (mevcut durumunu değiştir).
8C 3A	CRC16	İlk altı byte için hesaplanan 16 bit CRC (Cyclic Redundancy Check).

Komutun Detaylı Açıklaması

1. **Cihaz Adresi (01)**:
   - Bu komut, adresi 0x01 olan cihaza gönderilmiştir.
   - Cihaz adresi, bir Modbus ağında cihazların birbirinden ayrılması için kullanılır.
   - 0x00 adresi kullanıldığında tüm cihazlara aynı anda komut gönderilir (broadcast).
2. **Komut Kodu (05)**:
   - Modbus protokolünde, 0x05 komutu "Write Single Coil" komutudur.
   - Bu komut, bir coil (çıkış) durumunu belirlemek için kullanılır. Burada bir röle kontrolü için uygulanmıştır.
3. **Röle Adresi (00 00)**:
   - Rölenin hangi adrese sahip olduğunu belirtir.
   - Bu örnekte, adres 0x0000 olarak verilmiş ve ilk röle kontrol edilecektir.
   - Birden fazla röle olduğunda adres değerini 0x0001, 0x0002 vb. olarak değiştirebilirsiniz.
4. **Kontrol Komutu (FF 00)**:
   - Röleye uygulanacak kontrol işlemini tanımlar. Seçenekler şunlardır:
     - **0xFF00 (Aç)**: Röle aktif hale getirilir. Kontaklar kapatılır ve elektrik akışı sağlanır.
     - **0x0000 (Kapat)**: Röle pasif hale getirilir. Kontaklar açılır ve elektrik akışı kesilir.
     - **0x5500 (Ters Çevir)**: Rölenin mevcut durumu değiştirilir (Açık ise kapanır, kapalı ise açılır).
5. **CRC16 Checksum (8C 3A)**:
   - Bu değer, ilk 6 byte üzerinden CRC-16 algoritmasıyla hesaplanır.
   - CRC, verinin doğruluğunu kontrol etmek için kullanılır.
   - Hesaplama sırasında 0xFFFF başlangıç değeri ve genelde **polinom: 0xA001** kullanılır.

Örnek Senaryolar

1. **Röle Açma (Activate)**:
   - Komut: 01 05 00 00 FF 00 8C 3A
   - Bu komut, adresi 0x01 olan cihazdaki 0x0000 adresli röleyi aktif hale getirir.
2. **Röle Kapatma (Deactivate)**:
   - Komut: 01 05 00 00 00 00 CD CA
   - Bu komut, adresi 0x01 olan cihazdaki 0x0000 adresli röleyi kapatır.
3. **Röle Durumunu Ters Çevirme (Flip)**:
   - Komut: 01 05 00 00 55 00 DD EA
   - Bu komut, adresi 0x01 olan cihazdaki 0x0000 adresli rölenin mevcut durumunu değiştirir.

CRC Hesaplama Örneği

İlk 6 byte (01 05 00 00 FF 00) için CRC-16 algoritmasıyla şu adımlar takip edilir:

1. Başlangıç değeri 0xFFFF.
2. Her byte sırayla XOR işlemi uygulanır.
3. **Polinom:** 0xA001 kullanılarak kaydırma işlemleri yapılır.
4. Sonuç: 8C 3A.

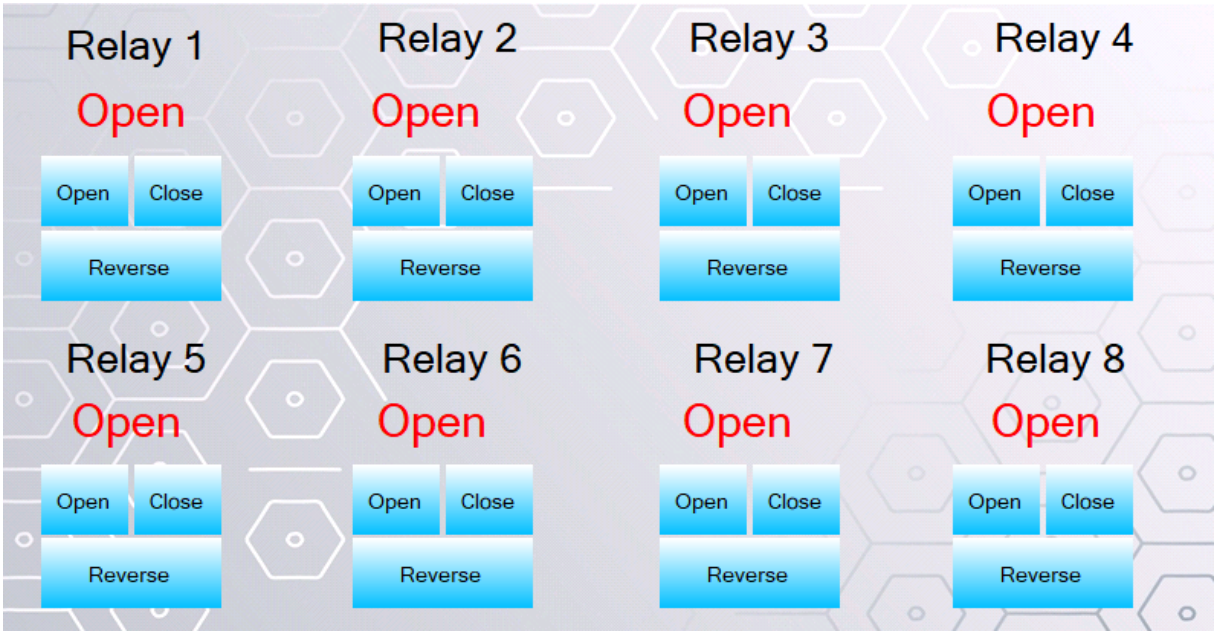
Modbus Çalışma Mantığı

1. Master cihaz, röle kontrol komutunu bu formatta gönderir.
2. Slave cihaz (örneğin, SmartIO), komutu alır ve belirtilen rölede değişikliği gerçekleştirir.
3. Slave cihaz, işlemin tamamlandığını doğrulamak için bir yanıt (ACK) gönderir.

Örnek Komutlar ve Açıklamaları

Komut	Açıklama
01 05 00 00 FF 00 8C 3A	Röle 0'ı AÇ (ON)
01 05 00 00 00 00 CD CA	Röle 0'ı KAPAT (OFF)
01 05 00 01 FF 00 DD FA	Röle 1'i AÇ (ON)
01 05 00 01 00 00 9C 0A	Röle 1'i KAPAT (OFF)
01 05 00 02 FF 00 2D FA	Röle 2'yi AÇ (ON)
01 05 00 02 00 00 6C 0A	Röle 2'yi KAPAT (OFF)
01 05 00 03 FF 00 7C 3A	Röle 3'ü AÇ (ON)
01 05 00 03 00 00 3D CA	Röle 3'ü KAPAT (OFF)
01 05 00 00 55 00 F2 9A	Röle 0'ı TERS ÇEVİR (FLIP)
01 05 00 01 55 00 A3 5A	Röle 1'i TERS ÇEVİR (FLIP)
01 05 00 02 55 00 53 5A	Röle 2'yi TERS ÇEVİR (FLIP)
01 05 00 03 55 00 02 9A	Röle 3'ü TERS ÇEVİR (FLIP)

AirHmi Kod Örneği:



WriteSingleCoil(1, 0x0000, 0x0000) Komutunun Açıklaması

Bu komut, **Modbus RTU** protokolünde bir coil (çıkış) yazma işlemini gerçekleştirir. Komut, röle kontrolü için kullanılır ve aşağıdaki parametrelerle detaylandırılır:

Komut Formatı:

WriteSingleCoil(deviceAddress, coilAddress, value);

Parametre	Değer	Açıklama
deviceAddress	1	Hedef cihazın adresi. Burada **1** adresli cihaza komut gönderiliyor. Modbus RTU'da adres 0x01-0xFF arasında olabilir.
coilAddress	0x0000	Kontrol edilen coil'in (röle) adresi. Burada **0x0000** adresli röle hedefleniyor. Coil adresleri rölelere karşılık gelir (örneğin, Röle 0).
value	0x0000	Coil'e yazılacak değer. Burada coil kapatılıyor (**0x0000: OFF, 0xFF00: ON, 0x5500: Reverse**).

Komutun Anlamı:

- **Cihaz Adresi (1):**
  Komut, Modbus ağında adresi **1** olan cihaza yönlendirilmiştir.
  Eğer birden fazla cihaz bağlıysa, komut yalnızca adresi **1** olan cihaz tarafından işlenir.
- **Coil Adresi (0x0000):**
  Adres **0x0000**, kontrol edilen ilk coil'in (örneğin Röle 0) adresini belirtir.

- **Değer (0x0000):**
  Coil'in durumu aşağıdaki değerlere göre ayarlanır:
  - **0xFF00 (ON):** Coil'i açar (aktif hale getirir).
  - 0x0000 **(OFF):** Coil'i kapatır (pasif hale getirir).
  - **0x5500 (REVERSE):** Coil'i reverse yapar.(Tersine çevirir).

Open Relay:

```
#include "stk.h"
#include "stdio.h"

int data;

data = WriteSingleCoil(1, 0x0000, 0x0000);
```

Close Relay:

```
#include "stk.h"
#include "stdio.h"

int data;

data = WriteSingleCoil(1, 0x0000, 0xFF00);
```

Reverse Relay:

```
#include "stk.h"
#include "stdio.h"

int data;

data = WriteSingleCoil(1, 0x0000, 0x5500);
```

Tüm Röleleri Kontrol Etme

Bu komut, tüm röleleri kontrol etmek için Modbus RTU protokolü kullanılarak hazırlanmıştır. Tek komut ile tüm röleler açılır, kapatılır yada tersine çevrilir. Komutun detaylarını adım adım açıklayalım:

Komut Yapısı:

01 05 00 FF FF 00 BC 0A

Her byte (sekizli veri birimi) belirli bir anlam taşır. detaylı açıklaması:

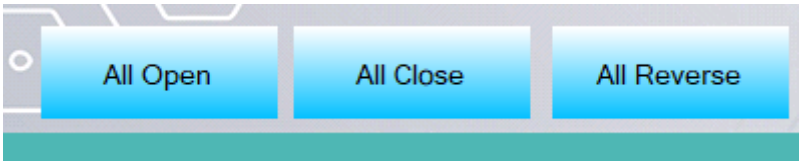
Byte	Anlamı	Açıklama
01	Cihaz Adresi	- 0x00: Yayın adresi (broadcast). - 0x01-0xFF: Belirli bir cihaza adresleme yapılır.
05	Komut Kodu	Röle kontrol komutunu belirtir (05 Command).
00 FF	Röle Adresi	Sabit 0x00FF
FF 00	Kontrol Komutu	Röleye uygulanacak işlemi belirtir: - 0xFF00: Röleyi aç (aktif hale getir). - 0x0000: Röleyi kapat (pasif hale getir). - 0x5500: Röleyi ters çevir (mevcut durumunu değiştir).
BC 0A	CRC16	İlk altı byte için hesaplanan 16 bit CRC (Cyclic Redundancy Check).

* Return cevabı olarak birebir aynısı gelir.

Örnek Senaryolar

1. **Röle Açma (Activate)**:
   - Komut: 01 05 00 FF FF 00 BC 0A
   - Bu komut, tüm röleleri açma işlevi yapar.
2. **Röle Kapatma (Deactivate)**:
   - Komut: 01 05 00 FF 00 00 FD FA
   - Bu komut, tüm röleleri kapatma işlevi yapar.
3. **Röle Durumunu Ters Çevirme (Flip)**:
   - Komut: 01 05 00 FF 55 00 C2 AA
   - Bu komut, tüm röleleri tersine çevirme işlevi yapar.

AirHmi Kod Örneği:



Open All Relay:

```
#include "stk.h"
#include "stdio.h"

int data;

data = WriteSingleCoil(1, 0xFF, 0x0000);
```

Close All Relay:

```
#include "stk.h"
#include "stdio.h"

int data;

data = WriteSingleCoil(1, 0xFF, 0xFF00);
```

Reverse All Relay:

```
#include "stk.h"
#include "stdio.h"

int data;

data = WriteSingleCoil(1, 0xFF, 0x5500);
```

Röle Durumlarını Okuma

Bu komut, **Modbus RTU protokolü** kullanılarak bir cihazdaki rölelerin durumlarını sorgulamak için tasarlanmıştır. Gönderilen komutun ve alınan yanıtın anlamları adım adım açıklanmıştır.

Komut Yapısı:

01 01 00 00 00 08 3D CC

Her byte (sekizli veri birimi) belirli bir anlam taşır. detaylı açıklaması:

Byte	Anlamı	Açıklama
01	Cihaz Adresi	Hedef cihazın adresi. Burada adresi **0x01** olan cihaz hedeflenmiştir.
01	Komut Kodu	Read Coils" komutu. Röle durumlarını kontrol etmek için kullanılır.
00 00	Röle Adresi	Rölelerin sorgulanacağı başlangıç adresi. Bu durumda sabit olarak **0x0000** verilmiştir.
00 08	Kontrol Komutu	Sorgulanan coil (röle) sayısı. Burada toplam **8 röle** sorgulanıyor.
3D CC	CRC16	İlk altı byte için hesaplanan 16 bit CRC (Cyclic Redundancy Check).

Komut Anlamı:

- Bu komut, Modbus ağındaki **1** adresli cihazdaki rölelerin (**coil**'lerin) durumlarını okumayı talep eder.
- **Başlangıç adresi (0x0000):** İlk röleden başlayarak durumlar okunur.
- **Sorgulanan coil sayısı (0x0008):** Toplam 8 röle durumu sorgulanır.

Cihazdan Gelen Yanıt (Return Code):

01 01 01 00 51 88

Byte	Anlamı	Açıklama
01	**Cihaz Adresi**	Yanıt veren cihazın adresi (**0x01**). Gönderilen komutla eşleşir.
01	**Komut Kodu**	"Read Coils" komutu yanıtı. Gönderilen komutun kabul edildiğini belirtir.
01	**Byte Sayısı**	Röle durumlarını belirtmek için dönen byte sayısı. Burada yalnızca **1 byte** döner.
00	**Röle Durumları**	Rölelerin durumunu temsil eden veri.
51 88	**CRC16 Checksum**	Yanıtın hata kontrolü için kullanılan CRC16 değeri.

Röle Durumlarının Anlamı (Byte Data):

- Gelen veri: **00**
  - Bu byte, 8 rölenin açık/kapalı durumlarını bit bazında temsil eder.
  - Her bit bir röleyi ifade eder:
    - **0:** Röle kapalı (OFF).
    - **1:** Röle açık (ON).

  Örnek:

  - Eğer gelen veri **01** olsaydı:
    - Röle 0: **Açık (ON)**.
    - Diğer tüm röleler: **Kapalı (OFF)**.

Örnek Senaryolar:

Gönderilen Komut	Açıklama
01 01 00 00 00 08 3D CC	1 numaralı cihazdaki tüm rölelerin durumlarını oku.

Gelen Yanıt	Açıklama
01 01 01 00 51 88	Cihazdaki tüm röleler kapalı (OFF).
01 01 01 03 41 F8	Röle 0 ve Röle 1 açık (ON), diğer röleler kapalı (OFF).
01 01 01 FF 90 A8	Tüm röleler açık (ON).

AirHmi Kod Örneği:

ReadCoils Komutunun Açıklaması

ReadCoils fonksiyonu, bir Modbus RTU cihazındaki coil (röle) durumlarını okumak için kullanılır.

Komut Formatı:

ReadCoils(deviceAddress, startAddress, count)

Parametre	Değer	Açıklama
deviceAddress	1	Hedef cihazın adresi. Burada **1** adresli cihaza komut gönderiliyor. Modbus RTU'da adres 0x01-0xFF arasında olabilir.
startAddress	0x0000	Okumaya başlanacak coil (röle) adresi. (0x0000 başlangıç adresidir ve ilk röleyi temsil eder.)
value	0x08	Okunacak toplam coil (röle) sayısı. (8 burada toplam 8 coil'in durumlarını okumayı temsil eder.)

Röleleri oku:

```
#include "stk.h"
#include "stdio.h"

int data;

data = ReadCoils(1, 0x0000, 8);

if( data & 0x01 )
   LabelSets("ELabelBox1" , "Close");
else
   LabelSets("ELabelBox1" , "Open");


if( data & 0x02 )
   LabelSets("ELabelBox2" , "Close");
else
   LabelSets("ELabelBox2" , "Open");

if( data & 0x04 )
   LabelSets("ELabelBox3" , "Close");
else
   LabelSets("ELabelBox3" , "Open");
```

```
if( data & 0x08 )
   LabelSets("ELabelBox4" , "Close");
else
   LabelSets("ELabelBox4" , "Open");

if( data & 0x10 )
   LabelSets("ELabelBox5" , "Close");
else
   LabelSets("ELabelBox5" , "Open");

if( data & 0x20 )
   LabelSets("ELabelBox6" , "Close");
else
   LabelSets("ELabelBox6" , "Open");

if( data & 0x40 )
   LabelSets("ELabelBox7" , "Close");
else
   LabelSets("ELabelBox7" , "Open");

if( data & 0x80 )
   LabelSets("ELabelBox8" , "Close");
else
   LabelSets("ELabelBox8" , "Open");
```

Tüm Röleleri Kontrol Etme

Bu komut, Modbus RTU protokolü kullanılarak bir cihazdaki birden fazla coil (röle) durumunu yazmak (açmak/kapatmak) için kullanılır. Bu fonksiyon, birden fazla rölenin durumunu aynı anda değiştirme imkânı sunar.

Komut Yapısı:

01 0F 00 00 00 08 01 FF BE D5

Her byte (sekizli veri birimi) belirli bir anlam taşır. detaylı açıklaması:

Byte	Anlamı	Açıklama
01	Cihaz Adresi	Hedef cihazın adresi. Burada adresi **0x01** olan cihaz hedeflenmiştir.
0F	Komut Kodu	"Write Multiple Coils" komutu. Birden fazla rölenin durumunu değiştirmek için kullanılır.
00 00	Röle Başlangıç Adresi	Rölelerin yazılmaya başlanacağı adres. Sabit olarak **0x0000** (ilk röle).
00 08	Röle Sayısı	Yazılacak toplam röle sayısı. Sabit olarak **8 röle** kontrol ediliyor.
01	Byte Sayısı	Röle durumlarını temsil eden toplam byte sayısı. Burada **1 byte** olarak tanımlanmış.
FF	Röle Durumları	Rölelerin açık/kapalı durumlarını bit bazında temsil eder. **FF = 11111111 (Tüm röleler açık).**
BE D5	CRC16	Komutun hata kontrolü için kullanılan CRC16 değeri.

Cihazdan Gelen Yanıt (Response):

01 0F 00 00 00 08 94 0B

Byte	Anlamı	Açıklama
01	**Cihaz Adresi**	Yanıt veren cihazın adresi (**1**).
0F	**Komut Kodu**	Gönderilen komutun bir yanıtı olduğunu belirtir (**Write Multiple Coils** komut yanıtı).
00 00	**Başlangıç Adresi**	Rölelerin yazılmaya başlanacağı adres. Sabit olarak **0x0000** (ilk röle).
00 08	**Röle Sayısı**	Yazılacak toplam röle sayısı. Sabit olarak **8 röle**.
94 0B	**CRC16 Checksum**	Yanıtın hata kontrolü için kullanılan CRC16 değeri.

Bu yanıt, komutun doğru bir şekilde işlendiğini ve cihazın talep edilen röle durumlarını ayarladığını belirtir.

Röle Durumlarının Anlamı (Bit Bazlı Analiz)

Gönderilen Byte (FF):

- **FF** (Binary: 11111111)
  - Bit0: Röle 0 (Açık)
  - Bit1: Röle 1 (Açık)
  - Bit2: Röle 2 (Açık)
  - Bit3: Röle 3 (Açık)
  - Bit4: Röle 4 (Açık)
  - Bit5: Röle 5 (Açık)
  - Bit6: Röle 6 (Açık)
  - Bit7: Röle 7 (Açık)

Örnek Komutlar ve Açıklamaları

1. **Tüm Röleleri Aç (All Relays ON):**

   01 0F 00 00 00 08 01 FF BE D5

   - **FF (Binary: 11111111):** Tüm röleler açık.

2. **Tüm Röleleri Kapat (All Relays OFF):**

   01 0F 00 00 00 08 01 00 FE 95

   - **00 (Binary: 00000000):** Tüm röleler kapalı.

3. **Röle 0 ve Röle 1 Açık, Diğerleri Kapalı:**

   01 0F 00 00 00 08 01 03 BE 94

   - **03 (Binary: 00000011):**
     - Röle 0: Açık (ON).
     - Röle 1: Açık (ON).
     - Röle 2-7: Kapalı (OFF).

4. **Röle 0 Kapalı, Röle 1 ve Röle 2 Açık, Diğerleri Kapalı:**

   01 0F 00 00 00 08 01 06 3E 14

   - **06 (Binary: 00000110):**
     - Röle 0: Kapalı (OFF).
     - Röle 1: Açık (ON).
     - Röle 2: Açık (ON).
     - Röle 3-7: Kapalı (OFF).

Özet

Tüm röleleri kontrol etme komutu:

- Çoklu rölenin durumlarını tek bir komutla ayarlamak için tasarlanmıştır.
- Gönderilen durumlar bit bazında kontrol edilerek rölelerin açık (ON) veya kapalı (OFF) durumu belirlenir.
- Bu komut, endüstriyel otomasyon ve enerji yönetimi projelerinde toplu röle kontrolü için ideal bir çözümdür.

AirHmi Kod Örneği:

ReadCoils Komutunun Açıklaması

WriteMultipleCoils fonksiyonu, Modbus RTU protokolü kullanılarak bir cihazdaki birden fazla coil (röle) durumunu yazmak (açmak/kapatmak) için kullanılır. Bu fonksiyon, birden fazla rölenin durumunu aynı anda değiştirme imkânı sunar.

Komut Formatı:

WriteMultipleCoils(deviceAddress, startAddress, value)

Parametre	Açıklama
deviceAddress	Hedef cihazın Modbus adresi. Bu, sorgulanan cihazın adresini temsil eder. (1 burada cihaz adresidir.)
startAddress	Rölelerin (coils) yazılmaya başlanacağı adres. (0x0000 başlangıç adresidir ve ilk röleyi temsil eder.)
value	Rölelerin açık/kapalı durumlarını bit bazında temsil eden veri. (0x55, 8 bitlik bir veridir.)

Örnek Kod:

```
#include "stk.h"
#include "stdio.h"

int data;

data = WriteMultipleCoils(1, 0x0000, 0x55);
```

Analog Girişleri Okuma

Bu komut, Modbus RTU protokolü kullanılarak bir cihazdaki analog giriş değerlerini okumak için tasarlanmıştır. Aşağıda gönderilen komut ve cihazdan gelen yanıtın anlamı detaylıca açıklanmıştır.

Komut Yapısı:

01 03 40 00 00 01 90 1B

Her byte (sekizli veri birimi) belirli bir anlam taşır. detaylı açıklaması:

Byte	Anlamı	Açıklama
01	Cihaz Adresi	Hedef cihazın adresi. Burada adresi **0x01** olan cihaz hedeflenmiştir.
01	Komut Kodu	**Read Holding Registers (0x03)** komutu. Analog giriş değerlerini okumak için kullanılır.
40 00	Başlangıç Register Adresi	Analog girişlerin (ADC) değerlerinin saklandığı register adresi. **0x4000** başlangıç adresidir.
00 01	Register Sayısı	Sorgulanan coil (röle) sayısı. Burada toplam **8 röle** sorgulanıyor.
90 1B	CRC16	İlk altı byte için hesaplanan 16 bit CRC (Cyclic Redundancy Check).

Komutun Anlamı:

- **01 (Cihaz Adresi):** Bu komut adresi **1** olan cihazdaki analog girişleri okumayı hedefler.
- **03 (Komut Kodu):** Analog girişlerin değerlerini okumak için kullanılan standart Modbus komutudur.
- **40 00 (Başlangıç Register Adresi):** Analog giriş değerlerini tutan register adresini belirtir.
- **00 01 (Register Sayısı):** Toplamda **1 register** okunacak (1 analog giriş değeri).
- **90 1B (CRC16):** Komutun doğruluğunu kontrol etmek için kullanılan CRC16 değeridir.

Cihazdan Gelen Yanıt (Response):

01 03 02 00 01 79 84

Byte	Anlamı	Açıklama
01	**Cihaz Adresi**	Yanıt veren cihazın adresi. Burada **0x01** adresli cihaz yanıt vermiştir.
03	**Komut Kodu**	Gönderilen **Read Holding Registers** komutunun yanıtı.
02	**Byte Sayısı**	Dönen veri uzunluğu (2 byte). Analog girişin değerini temsil eder.
00 01	**Analog Giriş Değeri**	Okunan analog girişin ham değeri: **0x0001**. Bu değer işlenmelidir (örneğin bölünerek gerçek değere çevrilir).
79 84	**CRC16 Checksum**	Yanıtın hata kontrolü için kullanılan CRC16 değeri.

Analog Giriş Değerinin Hesaplanması

Analog giriş değerleri ham ADC (Analog-Dijital Çevirici) değerleridir. Bu değer, gerçek analog giriş voltajını temsil etmesi için bir ölçekleme faktörü ile işlenir. Bu cihazda, ölçekleme faktörü **100** olarak belirtilmiştir.

Örnek:

- Gelen veri: 00 01 (Decimal: 1)
- Gerçek değer hesaplaması: Okunan Değer / 100
- Sonuç: Analog girişin gerçek değeri **0.01** birimdir (Voltaj).

Örnek Komutlar ve Yanıtlar

1. **Analog Giriş Değeri Okuma:**

   Gönderilen Komut: 01 03 40 00 00 01 90 1B
   Cihaz Yanıtı: 01 03 02 00 64 79 C4

   - Okunan değer: 00 64 (Decimal: 100)
   - Gerçek değer: 100/100=1.0 (örneğin, 1.0 Volt).

2. **Düşük Bir Analog Giriş Değeri:**

Gönderilen Komut: 01 03 40 00 00 01 90 1B
Cihaz Yanıtı: 01 03 02 00 0A 79 C9

- Okunan değer: 00 0A (Decimal: 10)
- Gerçek değer: 10/100=0.1 (örneğin, 0.1 Volt).

Kullanım Alanları

1. **Gerilim Ölçümü:**
   - Analog sensörlerden gelen gerilim değerlerini okumak için kullanılır.
2. **Sensör Entegrasyonu:**
   - Akım, sıcaklık veya basınç sensörlerinin analog çıkışları bu girişlerle ölçülebilir.
3. **Endüstriyel Kontrol:**
   - Analog sinyalleri dijital verilere dönüştürerek işleme alınmasını sağlar.

AirHmi Kod Örneği:

ReadHoldingRegisters Komutunun Açıklaması

ReadHoldingRegisters fonksiyonu, bir Modbus RTU cihazındaki holding register (tutucu kayıt) verilerini okumak için kullanılır. Bu kod, 1 numaralı cihazdan adresi 0x0000 olan register'dan başlayarak 4 adet holding register'ı okuyup, bu değerleri Registers dizisine depolar.

Komut Formatı:

ReadHoldingRegisters(int deviceAddress, int startAddress, int count, short* buffer);

Parametre	Açıklama
deviceAddress	Modbus cihazının adresi. Bu adres, sorgulanan cihazı ağ üzerindeki diğer cihazlardan ayırır.
startAddress	Okumaya başlanacak register adresi. Genellikle cihazdaki bir veri bloğunun başlangıç adresini belirtir.
count	Okunacak toplam register sayısı. Her bir register 16 bit (2 byte) uzunluğundadır.
buffer	Okunan verilerin depolanacağı bir dizi (pointer). Bu dizide her bir register değeri saklanır.

Örnek Kod:

```
#include "stk.h"
#include "stdio.h"

char buffer[200];

short Registers[4];

ReadHoldingRegisters(1, 0x0000, 4 , Registers);

double adc1 = Registers[0] / 100.0;
double adc2 = Registers[1] / 100.0;
double adc3 = Registers[2] / 100.0;
double adc4 = Registers[3] / 100.0;


sprintf(buffer,"%.02f",adc1);

LabelSets("ELabelBox9" , buffer );

sprintf(buffer,"%.02f",adc2);

LabelSets("ELabelBox10" , buffer );

sprintf(buffer,"%.02f",adc3);

LabelSets("ELabelBox11" , buffer );

sprintf(buffer,"%.02f",adc4);

LabelSets("ELabelBox12" , buffer );
```

Dijital Girişleri Okuma

Bu komut, Modbus RTU protokolü kullanılarak cihazdaki dijital girişlerin durumlarını kontrol etmek için tasarlanmıştır. Aşağıda, gönderilen komutun ve cihazdan alınan yanıtın detaylı bir çözümlemesi bulunmaktadır.

Komut Yapısı:

01 01 00 00 00 04 3D C9

Her byte (sekizli veri birimi) belirli bir anlam taşır. detaylı açıklaması:

Byte	Anlamı	Açıklama
01	Cihaz Adresi	Hedef cihazın adresi. Burada adresi **0x01** olan cihaz hedeflenmiştir.
01	Komut Kodu	Read Coils" komutu. Röle durumlarını kontrol etmek için kullanılır.
00 01	Başlangıç Adresi	Okumaya başlanacak dijital girişlerin adresi. Burada adres **0x0001** olarak verilmiştir.
00 04	Kontrol Komutu	Sorgulanan toplam dijital giriş sayısı. Bu örnekte **4 dijital giriş** sorgulanıyor.
3D C9	CRC16	İlk altı byte için hesaplanan 16 bit CRC (Cyclic Redundancy Check).

Komutun Anlamı

- **Cihaz Adresi (01):** Komut, adresi 1 olan cihazdan dijital girişlerin durumlarını sorgular.
- **Komut Kodu (01):** Dijital girişlerin durumlarını kontrol etmek için kullanılan standart Modbus komutudur.
- **Başlangıç Adresi (00 01):** İlk dijital girişin adresi **0x0001**'dir.
- **Dijital Giriş Sayısı (00 04):** Toplamda **4 dijital giriş** sorgulanacaktır.
- **CRC16 Checksum (3D C9):** Komutun doğru bir şekilde işlenebilmesi için eklenen hata kontrol kodudur.

Cihazdan Gelen Yanıt (Response)

01 01 01 0F 8C 35

Byte	Anlamı	Açıklama
01	**Cihaz Adresi**	Yanıt veren cihazın adresi (**0x01**).
01	**Komut Kodu**	Gönderilen **Read Coils** komutunun yanıtı.
01	**Veri Byte Sayısı**	Dijital giriş durumlarını içeren byte sayısı. Burada **1 byte** veri döner.
0F	**Dijital Giriş Durumları**	Dijital girişlerin durumunu bit bazında temsil eder.
8C 35	**CRC16 Checksum**	Yanıtın doğruluğunu kontrol etmek için kullanılan CRC16 değeri.

Dijital Giriş Durumlarının Anlamı

Dönen Veri (0F):

- Dijital girişlerin durumları **bit bazında** temsil edilir.
- Her bit, bir dijital girişin açık/kapalı (1/0) durumunu belirtir.

Bit	Dijital Giriş	Durum
Bit0	**Dijital Giriş 1**	Açık (**1**)
Bit1	**Dijital Giriş 2**	Açık (**1**)
Bit2	**Dijital Giriş 3**	Açık (**1**)
Bit3	**Dijital Giriş 4**	Açık (**1**)

Örnek Komutlar ve Yanıtlar

1. **Tüm Dijital Girişler Açık:**
   - **Gönderilen Komut:** 01 01 00 01 00 04 3D C9
   - **Yanıt:** 01 01 01 0F 8C 35
   - **Açıklama:**
     Dönen veri **0F** (Binary: 00001111):
     - **Bit0-3:** Tüm dijital girişler açık (ON).
2. **İlk İki Dijital Giriş Açık, Diğerleri Kapalı:**
   - **Gönderilen Komut:** 01 01 00 01 00 04 3D C9
   - **Yanıt:** 01 01 01 03 41 89
   - **Açıklama:**
     Dönen veri **03** (Binary: 00000011):
     - **Bit0:** Açık (ON).
     - **Bit1:** Açık (ON).
     - **Bit2-3:** Kapalı (OFF).
3. **Tüm Dijital Girişler Kapalı:**

- **Gönderilen Komut:** 01 01 00 01 00 04 3D C9
- **Yanıt:** 01 01 01 00 51 88
- **Açıklama:**
  Dönen veri **00** (Binary: 00000000), tüm dijital girişlerin kapalı (OFF) olduğunu belirtir.

Kullanım Alanları

1. **Sensör Durumlarının İzlenmesi:**
   - Dijital girişlere bağlı sensörlerin (örneğin, butonlar veya sınır anahtarları) durumlarını izlemek için kullanılır.
2. **Endüstriyel Otomasyon:**
   - Dijital sinyallerin kontrolü ve izlenmesi için idealdir.
3. **Hata Tespiti:**
   - Dijital girişlerin beklenen durumlarla uyumlu olup olmadığını kontrol etmek için.

AirHmi Kod Örneği:

ReadCoils Komutunun Açıklaması

ReadCoils fonksiyonu, bir Modbus RTU cihazındaki coil (dijital input) durumlarını okumak için kullanılır.

Komut Formatı:

ReadCoils(deviceAddress, startAddress, count)

Parametre	Değer	Açıklama
deviceAddress	1	Hedef cihazın adresi. Burada **1** adresli cihaza komut gönderiliyor. Modbus RTU'da adres 0x01-0xFF arasında olabilir.
startAddress	0x0001	Okumaya başlanacak coil (dijital giriş) adresi.
value	0x04	Okunacak toplam coil (röle) sayısı. (4 burada toplam 4 coil'in durumlarını okumayı temsil eder.)

Örnek kod:

```
#include "stk.h"
#include "stdio.h"


int data;

data = ReadCoils(1, 0x0001, 4);


if( data & 0x01 )
   LabelSets("ELabelBox13" , "Close");
else
   LabelSets("ELabelBox13" , "Open");


if( data & 0x02 )
   LabelSets("ELabelBox14" , "Close");
else
   LabelSets("ELabelBox14" , "Open");

if( data & 0x04 )
   LabelSets("ELabelBox15" , "Close");
else
   LabelSets("ELabelBox15" , "Open");

if( data & 0x08 )
   LabelSets("ELabelBox16" , "Close");
else
   LabelSets("ELabelBox16" , "Open");
```

Set Baud Rate Komutu

Bu komut, Modbus RTU protokolü kullanılarak bir cihazın baud rate (haberleşme hızı) değerini değiştirmek için tasarlanmıştır. Aşağıda gönderilen komutun yapısı, anlamı ve örnekleri detaylı bir şekilde açıklanmıştır.

Komut Yapısı:

01 01 00 00 00 04 3D C9

Her byte (sekizli veri birimi) belirli bir anlam taşır. detaylı açıklaması:

Byte	Anlamı	Açıklama
01	Cihaz Adresi	Hedef cihazın adresi. Burada adresi **0x01** olan cihaz hedeflenmiştir.
06	Komut Kodu	**0x06** komutu, bir register'a tek bir değer yazmak için kullanılır.
20 00	Başlangıç Adresi	Baud rate değerinin yazılacağı adres (örneğin, **0x2000**).
00 60	Kontrol Komutu	Sorgulanan toplam dijital giriş sayısı. Bu örnekte **4 dijital giriş** sorgulanıyor.
3D C9	CRC16	İlk altı byte için hesaplanan 16 bit CRC (Cyclic Redundancy Check).

Örnek Komutlar ve Açıklamaları

1. **Baud Rate 4800 Ayarı**

Gönderilen Komut: 00 06 20 00 00 30 83 DB

Açıklama:

Byte	Anlamı	Açıklama
00	Cihaz Adresi	Broadcast (tüm cihazlar) adresi kullanılmıştır.
06	Fonksiyon Kodu	Register'a tek bir değer yazmak için kullanılır.
20 00	Register Adresi	Baud rate ayarı için kullanılan register adresi.
00 30	Baud Rate Değeri	**0x30 = 4800** / 100 baud.
83 DB	CRC16	Komutun hata kontrol kodu.

2. **Baud Rate 9600 Ayarı**

Gönderilen Komut: 00 06 20 00 00 01 42 1B

Açıklama:

Byte	Anlamı	Açıklama
00	Cihaz Adresi	Broadcast (tüm cihazlar) adresi kullanılmıştır.
06	Fonksiyon Kodu	Register'a tek bir değer yazmak için kullanılır.
20 00	Register Adresi	Baud rate ayarı için kullanılan register adresi.
00 60	Baud Rate Değeri	**0x60 = 9600 / 100** baud.
42 1B	CRC16	Komutun hata kontrol kodu.

3. **Baud Rate 115200 Ayarı**

Gönderilen Komut: 00 06 20 00 00 05 43 D8

Açıklama:

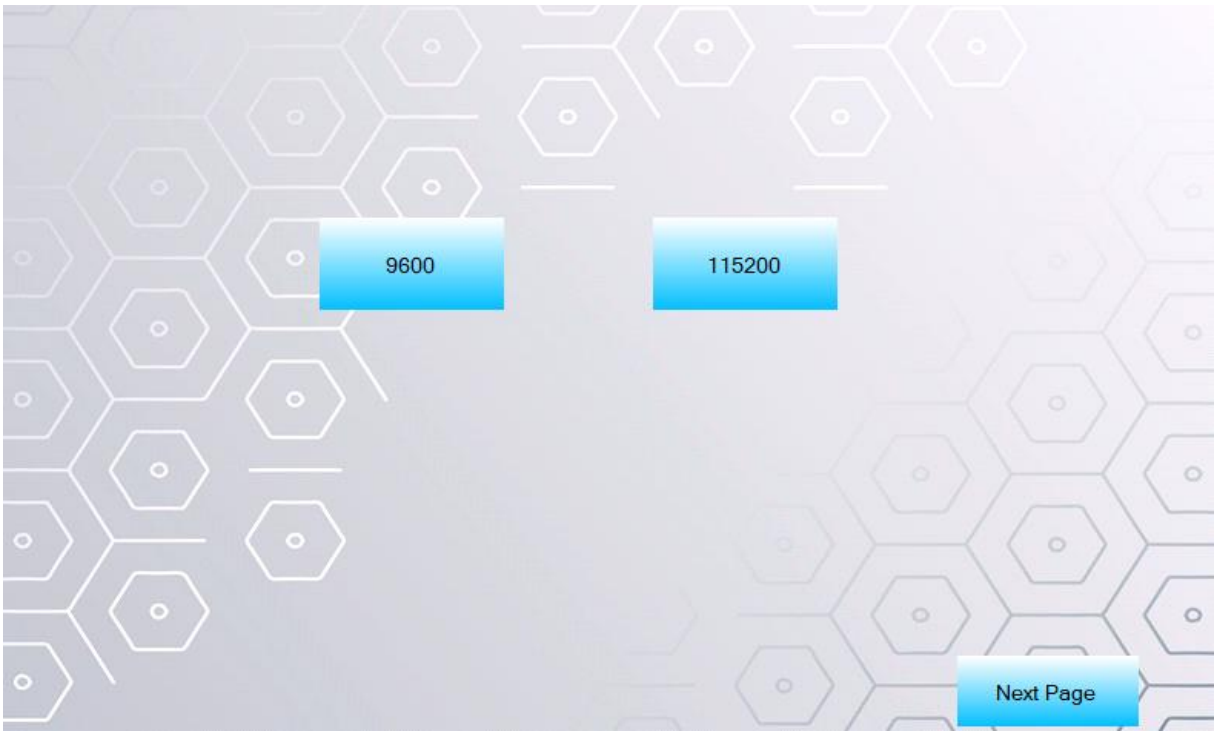
Byte	Anlamı	Açıklama
00	Cihaz Adresi	Broadcast (tüm cihazlar) adresi kullanılmıştır.
06	Fonksiyon Kodu	Register'a tek bir değer yazmak için kullanılır.
20 00	Register Adresi	Baud rate ayarı için kullanılan register adresi.

Byte	Anlamı	Açıklama
04 80	Baud Rate Değeri	**0x0480 = 115200 / 100** baud.
43 D8	CRC16	Komutun hata kontrol kodu.

Kullanım Alanları

1. **Bağlantı Hızının Ayarlanması:**
   - Cihazın haberleşme hızını (baud rate) değiştirmek için kullanılır.
2. **Cihaz Konfigürasyonu:**
   - Modbus cihazlarının farklı hızlara sahip ağlarda çalışmasını sağlamak için ayar yapılır.
3. **Endüstriyel Otomasyon:**
   - Farklı cihazların aynı iletişim hızında çalışabilmesi için standartlaştırma yapılır.

AirHmi Kod Örneği:



WriteSingleRegister Komutunun Açıklaması

WriteSingleRegister komutu, Modbus RTU veya TCP protokolü kullanılarak bir cihazdaki tek bir register'a (16-bit veri saklama alanı) değer yazmak için kullanılır. Bu komut, cihazın bir parametresini değiştirmek veya bir ayarı güncellemek için kullanılır.

Komut Formatı:

WriteSingleRegister(deviceAddress, startAddress, count)

Parametre	Değer	Açıklama
deviceAddress	0	Broadcast (tüm cihazlar) adresi kullanılmıştır.
Register Address	0x2000	Değerin yazılacağı register'ın adresi.
value	0x04 80	Register'a yazılacak 16-bit değer.

Örnek kod:

```
#include "stk.h"
#include "stdio.h"


#include "stk.h"


int baud    = 9600;
int setBaud = baud / 100;

WriteSingleRegister(0,0x2000,setBaud);



#include "stk.h"


int baud    = 115200;
int setBaud = baud / 100;

WriteSingleRegister(0,0x2000,setBaud);
```

Cihaz Modbus Adresi Değiştirme Komutu

Bu komut, Modbus RTU protokolü kullanılarak bir cihazın Modbus adresini değiştirmek için tasarlanmıştır. Write Single Register (0x06) komutunu kullanır ve cihaz adresi, yazılacak bir register'a yeni bir değer olarak atanır.

Komut Yapısı:

Bytes	Anlamı	Açıklama
00	Cihaz Adresi	Komutun hedeflendiği cihazın Modbus adresi (**0x00** ise broadcast adresidir).
06	06 command	**0x06**, bir register'a tek bir değer yazmak için kullanılır.
40 00	command register	**0x4000**, cihaz adresini değiştirmek için kullanılan register'dır.
00 01	device address	yeni cihaz adresini belirtir (örneğin, **0x0001**).
5C 1B	CRC16	CRC16 checksum of the first 6 bytes of data

Return code: 00 06 40 00 00 01 5C 1B

Komut Cevabı:

Bytes	Anlamı	Açıklama
00	Cihaz Adresi	Komutun hedeflendiği cihazın Modbus adresi (**0x00** ise broadcast adresidir).
06	06 command	**0x06**, bir register'a tek bir değer yazmak için kullanılır.
40 00	command register	**0x4000**, cihaz adresini değiştirmek için kullanılan register'dır.

00 01	device address	yeni cihaz adresini belirtir (örneğin, **0x0001**).
5C 1B	CRC16	CRC16 checksum of the first 6 bytes of data

Gönderilen Komut ve Açıklaması

Örnek 1: Cihaz Adresini 0x01 olarak Ayarlama

Gönderilen komut:

00 06 40 00 00 01 5C 1B

Byte	Anlamı	Açıklama
00	**Cihaz Adresi**	Broadcast adresi (**0x00**), tüm cihazlara aynı anda komut gönderilir.
06	**Fonksiyon Kodu**	**Write Single Register** komutu.
40 00	**Register Adresi**	**0x4000**, cihaz adresini değiştirmek için kullanılan register.
00 01	**Yazılacak Değer**	Yeni cihaz adresi **0x01** olarak ayarlanır.
5C 1B	**CRC16**	Komutun doğruluğunu kontrol eden hata kontrol kodu.

Örnek 2: Cihaz Adresini 0x02 olarak Ayarlama

Gönderilen komut:

00 06 40 00 00 02 1C 1A

Byte	Anlamı	Açıklama
00	**Cihaz Adresi**	Broadcast adresi (**0x00**), tüm cihazlara aynı anda komut gönderilir.
06	**Fonksiyon Kodu**	**Write Single Register** komutu.
40 00	**Register Adresi**	**0x4000**, cihaz adresini değiştirmek için kullanılan register.
00 02	**Yazılacak Değer**	Yeni cihaz adresi **0x02** olarak ayarlanır.
1C 1A	**CRC16**	Komutun doğruluğunu kontrol eden hata kontrol kodu.

Örnek 3: Cihaz Adresini 0x03 olarak Ayarlama

Gönderilen komut:

00 06 40 00 00 03 DD DA

Byte	Anlamı	Açıklama
00	**Cihaz Adresi**	Broadcast adresi (**0x00**), tüm cihazlara aynı anda komut gönderilir.
06	**Fonksiyon Kodu**	**Write Single Register** komutu.
40 00	**Register Adresi**	**0x4000**, cihaz adresini değiştirmek için kullanılan register.
00 03	**Yazılacak Değer**	Yeni cihaz adresi **0x03** olarak ayarlanır.
DD DA	**CRC16**	Komutun doğruluğunu kontrol eden hata kontrol kodu.

Cihazdan Gelen Yanıt (Response)

Cihaz, komutun başarıyla işlendiğini doğrulamak için aynı formatta bir yanıt döner.

Gönderilen Komut	Cihaz Yanıtı
00 06 40 00 00 01 5C 1B	00 06 40 00 00 01 5C 1B
00 06 40 00 00 02 1C 1A	00 06 40 00 00 02 1C 1A
00 06 40 00 00 03 DD DA	00 06 40 00 00 03 DD DA

Yanıt, gönderilen komut ile birebir aynı formatta olur. Bu, cihazın adres değiştirme işlemini başarıyla tamamladığını gösterir.

Kullanım Alanları

1. **Cihaz Adresi Güncelleme:**
   - Modbus ağındaki cihazların adresini değiştirerek, ağda farklı adreslere sahip cihazların çakışmasını önler.
2. **Broadcast Kullanımı:**
   - Broadcast adresi (**0x00**) kullanılarak tüm cihazlara aynı komut gönderilebilir.
3. **Otomasyon Sistemleri:**
   - Endüstriyel uygulamalarda cihazların benzersiz adreslerle konfigüre edilmesi için kullanılır.

Özet

- **Set Device Address Command**, bir cihazın Modbus adresini değiştirmek için kullanılır.
- Kullanıcı, cihaz adresini değiştirmek için **0x4000** register'ına yeni bir değer yazar.
- Cihaz, komutu işledikten sonra aynı komut formatında bir yanıt döner.

AirHmi Kod Örneği:

WriteSingleRegister Komutunun Açıklaması

WriteSingleRegister komutu, Modbus RTU veya TCP protokolü kullanılarak bir cihazdaki tek bir register'a (16-bit veri saklama alanı) değer yazmak için kullanılır. Bu komut, cihazın bir parametresini değiştirmek veya bir ayarı güncellemek için kullanılır.

Komut Formatı:

WriteSingleRegister(deviceAddress, startAddress, count)

Parametre	Değer	Açıklama
deviceAddress	0	Broadcast (tüm cihazlar) adresi kullanılmıştır.
Register Address	0x4000	Değerin yazılacağı register'ın adresi.
value	0x00 02	Modbus yeni adresi 2 olarak belirliyoruz.

Örnek kod:

```
#include "stk.h"

WriteSingleRegister(0,0x4000,1);  // adresi 1 olarak ayarlama

WriteSingleRegister(0,0x4000,2);  // adresi 2 olarak ayarlama
```

Cihaz Modbus Adresini Okuma Komutu

Bu komut, Modbus RTU protokolü kullanılarak bir cihazın **Modbus adresini** okumak için tasarlanmıştır. Komut, **Read Holding Registers (0x03)** fonksiyonunu kullanır. Aşağıda gönderilen komutun yapısı, cihazdan alınan yanıt ve örnekler detaylı bir şekilde açıklanmıştır.

Bytes	Anlamı	Açıklama
00	Cihaz Adresi	Hedef cihazın Modbus adresi (**0x00** ise broadcast adresidir).
03	Fonksiyon Kodu	read the device address command
40 00	Register Adresi	**0x4000**, cihaz adresini tutan register'dır. 0x8000: Cihaz versiyon numarası
00 01	Register Sayısı	Okunacak register sayısı. Burada sabit olarak **1** register okunur.
90 1B	CRC16	CRC16 checksum of the first 6 bytes of data

Gönderilen Komut ve Açıklaması

Örnek Komut: Cihaz Adresini Okuma

Gönderilen komut:

00 03 40 00 00 01 90 1B

Byte	Anlamı	Açıklama
00	**Cihaz Adresi**	Broadcast adresi (**0x00**), tüm cihazlara aynı anda komut gönderilir.
03	**Fonksiyon Kodu**	**Read Holding Registers** komutu.
40 00	**Register Adresi**	**0x4000**, cihaz adresini tutan register.
00 01	**Register Sayısı**	**1 register** okunacaktır.
90 1B	**CRC16**	Komutun doğruluğunu kontrol eden hata kontrol kodu.

Cihazdan Gelen Yanıt (Response)

Cihaz, okunan cihaz adresi ile bir yanıt döner. Yanıtın formatı şu şekildedir:

Alan	Anlamı	Açıklama
Device Address	**Cihaz Adresi**	Yanıt veren cihazın adresi (örneğin, 0x01).
Function Code	**Fonksiyon Kodu**	Gönderilen **Read Holding Registers** komutunun yanıtı (**0x03**).
Byte Count	**Veri Byte Sayısı**	Dönen verinin toplam byte uzunluğu. Burada **2 byte** döner.
Data	**Cihaz Adresi**	Cihazın mevcut Modbus adresini tutan değer.
CRC16	**Hata Kontrolü (Checksum)**	Yanıtın doğruluğunu kontrol etmek için kullanılan CRC16 değeri.

Örnek Yanıt 1: Adres 0x01

01 03 02 00 01 79 84

Byte	Anlamı	Açıklama
01	**Cihaz Adresi**	Yanıt veren cihazın adresi (**0x01**).
03	**Fonksiyon Kodu**	Gönderilen **Read Holding Registers** komutunun yanıtı.
02	**Veri Byte Sayısı**	Okunan veri 2 byte'tır.
00 01	**Cihaz Adresi**	Cihaz adresi **0x0001**'dir.
79 84	**CRC16**	Yanıtın hata kontrol kodu.

Örnek Yanıt 2: Adres 0x02

02 03 02 00 02 7D 85

Byte	Anlamı	Açıklama
02	**Cihaz Adresi**	Yanıt veren cihazın adresi (**0x02**).
03	**Fonksiyon Kodu**	Gönderilen **Read Holding Registers** komutunun yanıtı.
02	**Veri Byte Sayısı**	Okunan veri 2 byte'tır.
00 02	**Cihaz Adresi**	Cihaz adresi **0x0002**'dir.

Byte	Anlamı	Açıklama
7D 85	**CRC16**	Yanıtın hata kontrol kodu.

Örnek Yanıt 3: Adres 0x03

03 03 02 00 03 81 85

Byte	Anlamı	Açıklama
03	**Cihaz Adresi**	Yanıt veren cihazın adresi (**0x03**).
03	**Fonksiyon Kodu**	Gönderilen **Read Holding Registers** komutunun yanıtı.
02	**Veri Byte Sayısı**	Okunan veri 2 byte'tır.
00 03	**Cihaz Adresi**	Cihaz adresi **0x0003**'dir.
81 85	**CRC16**	Yanıtın hata kontrol kodu.

AirHmi Kod Örneği:

Örnek kod:

```
#include "stk.h"
#include "stdio.h"

char buffer[200];
```

```
short Registers[4];

ReadHoldingRegisters(2, 0x4000, 1 , Registers); // Cihaz adresi sorgulama

sprintf(buffer,"%d",Registers[0]);

LabelSets("ELabelBox19" , buffer );


ReadHoldingRegisters(2, 0x8000, 1 , Registers); // Cihaz versiyon numarası sorgulama

sprintf(buffer,"%d",Registers[0]);

LabelSets("ELabelBox20" , buffer );
```

ADC Örnekleme Değiştirme

Bu komut, Modbus RTU protokolü kullanılarak bir cihazın adc örnekleme değerini değiştirmek için tasarlanmıştır. Aşağıda gönderilen komutun yapısı, anlamı ve örnekleri detaylı bir şekilde açıklanmıştır.

Komut Yapısı:

00 06 10 00 03 e8 8c 65

Her byte (sekizli veri birimi) belirli bir anlam taşır. detaylı açıklaması:

Byte	Anlamı	Açıklama
00	Cihaz Adresi	Komutun hedeflendiği cihazın adresi. Burada broadcast adresi (**0x00**) kullanılmıştır.
06	Komut Kodu	**0x06** komutu, bir register'a tek bir değer yazmak için kullanılır.
10 00	Başlangıç Adresi	değerinin yazılacağı adres (örneğin, **0x1000**).
03 E8	Kontrol Komutu	1000 değerinin hex karşılığı(Örnekleme sayısı)
3D C9	CRC16	İlk altı byte için hesaplanan 16 bit CRC (Cyclic Redundancy Check).

Kullanım Örnekleri

Örnek 1: ADC Örnekleme Sayısını 500 Yapmak

Gönderilen Komut:

00 06 10 00 01 F4 D9 6C

Byte	**Anlamı**	**Açıklama**
00	Broadcast Adresi	Komut tüm cihazlara gönderilir.
06	Komut Kodu	Write Single Register komutu.
10 00	Başlangıç Adresi	ADC örnekleme değerinin saklandığı register adresi.

Byte	Anlamı	Açıklama
01 F4	Yazılacak Değer	**0x01F4**, decimal karşılığı **500**'dür.
D9 6C	CRC16 Checksum	Hata kontrol kodu.

Örnek 2: ADC Örnekleme Sayısını 2000 Yapmak

Gönderilen Komut:

00 06 10 00 07 D0 D5 4E

Byte	Anlamı	Açıklama
00	Broadcast Adresi	Komut tüm cihazlara gönderilir.
06	Komut Kodu	Write Single Register komutu.
10 00	Başlangıç Adresi	ADC örnekleme değerinin saklandığı register adresi.
07 D0	Yazılacak Değer	**0x07D0**, decimal karşılığı **2000**'dir.
D5 4E	CRC16 Checksum	Hata kontrol kodu.

AirHmi Kod Örneği:

Örnek kod:

```
#include "stk.h"
#include "stdio.h"
#include "stdlib.h"

char data[100];
KeypadNum("" , data );

LabelSets("ELabelBox21",data);



int idata = atoi(data);

WriteSingleRegister(0,0x1000,idata);  // Keypadden alınan modbus değeridir.
```