Esp-12F 1 Kanallı Röle Modülü 250V/DC30V

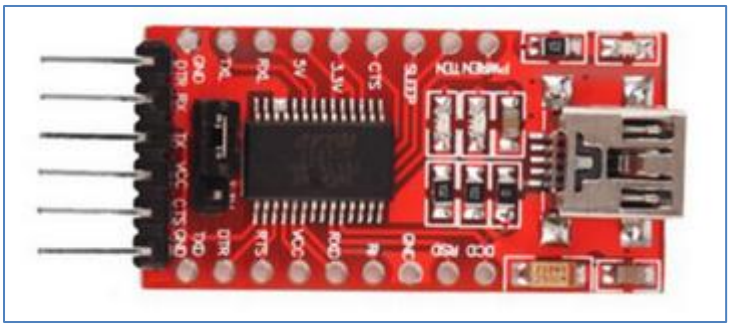
Stok Kodu: 10791

- **Ürün, ESP8266_MOD modülü, 5-28V güç (besleme) girişi ve 1 adet kumanda edilebilir röleden oluşmaktadır.**
- **ESP8266_MOD modülü üzerine uygun programı yükleyerek WI-FI üzerinden uzaktan kumanda edilebilir röle devresi kurmak mümkündür.**
- **Röleler ile 10A – 220V AC veya 10A – 30V DC akım anahtarlanabilir.**

Dikkat: *220V ile röle kontaklarına anahtarlama amaçlı bağlantı yapıldığında cihazın üzerinde 220V açık bir şekilde olacağı için **hayati tehlike** vardır.*
Gereken önlemler alınmalı, cihaz uygun bir kutuya izoleli şekilde konulmalı, kullanıcıları uyaran etiketler cihaz üzerinde direktiflere uygun şekilde yer almalıdır.
Cihaza sadece yetkili elektrik tesisatçısı tarafından müdahale edilmelidir.

- **Ürünün üzerinde herhangi bir program bulunmamaktadır.**
- **Örnek olarak verilen 1 röle kontrol eden WEB sunucu yazılımını cihaza yüklemeniz için gerekli adımlar aşağıda aktarılmıştır.**

- **Ürün üzerinde USB driver bulunmadığı için edineceğiniz FT232 Modül (Şekil – 1) veya benzeri ile programı yüklemeniz gerekmektedir.**



Şekil: 1

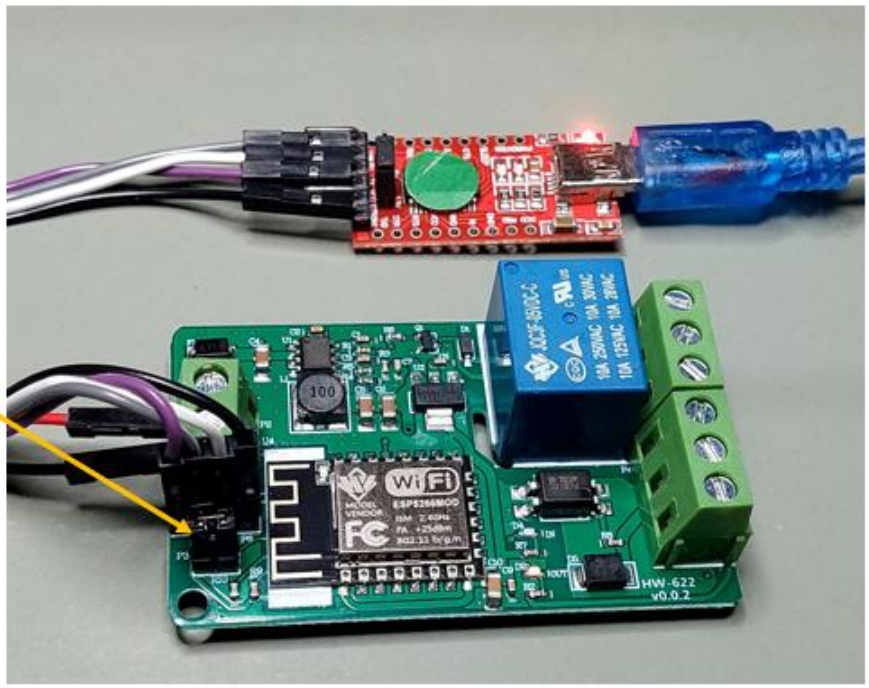
- **Ek-1'de dökümü yer alan programı Arduino IDE'de açın. (Sitemizde yer alıyor)**

- **Programın içindeki WI-FI SSID ve WIFI Şifre parametrelerini kendi WIFI Network ID'niz (SSID) ve şifreniz ile değiştirin. (Örnekte, SSID ve Şifre "DIRENC" olarak geçmektedir.**
- **ESP8266_MOD GPIO 4 numaralı pini röleyi süren transistöre bağlıdır. Yazılımda bu atama yapılmıştır.**

Önemli:
FT-232 modülü üzerindeki voltaj seçim jumper'ı 3.3V moduna alınır.

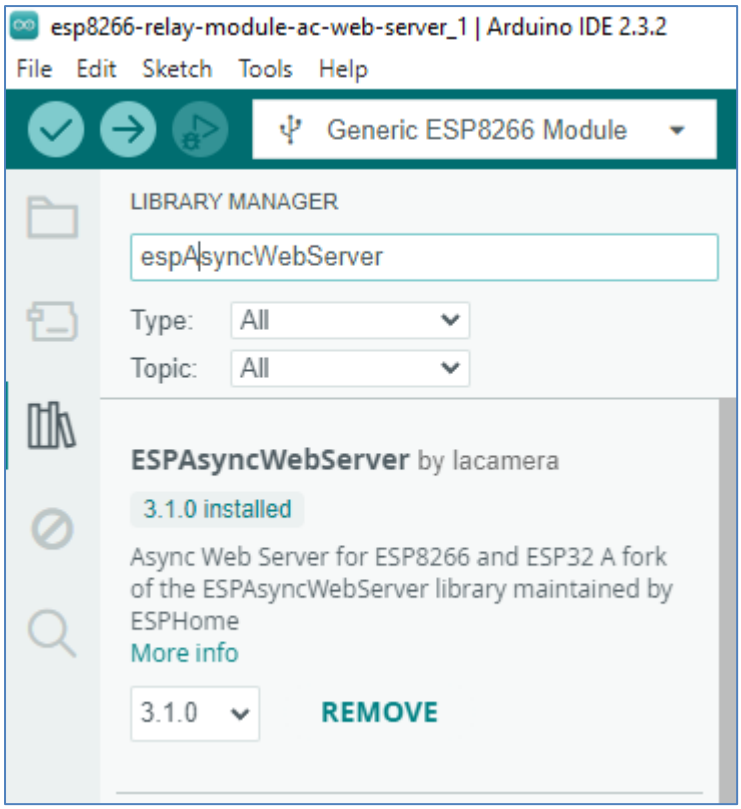
Resimdeki gibi FT-232 dönüştürücü ve Cihazın RX-TX-GND ve Vcc uçları bağlanır. Siyah renkli jumper takılarak program yükleme moduna geçilmesi sağlanır.

Cihaz	FT232.....
RX	TX (Beyaz)
TX	RX (Gri)
GND	GND (Siyah)
Vcc	Vcc (Mor)

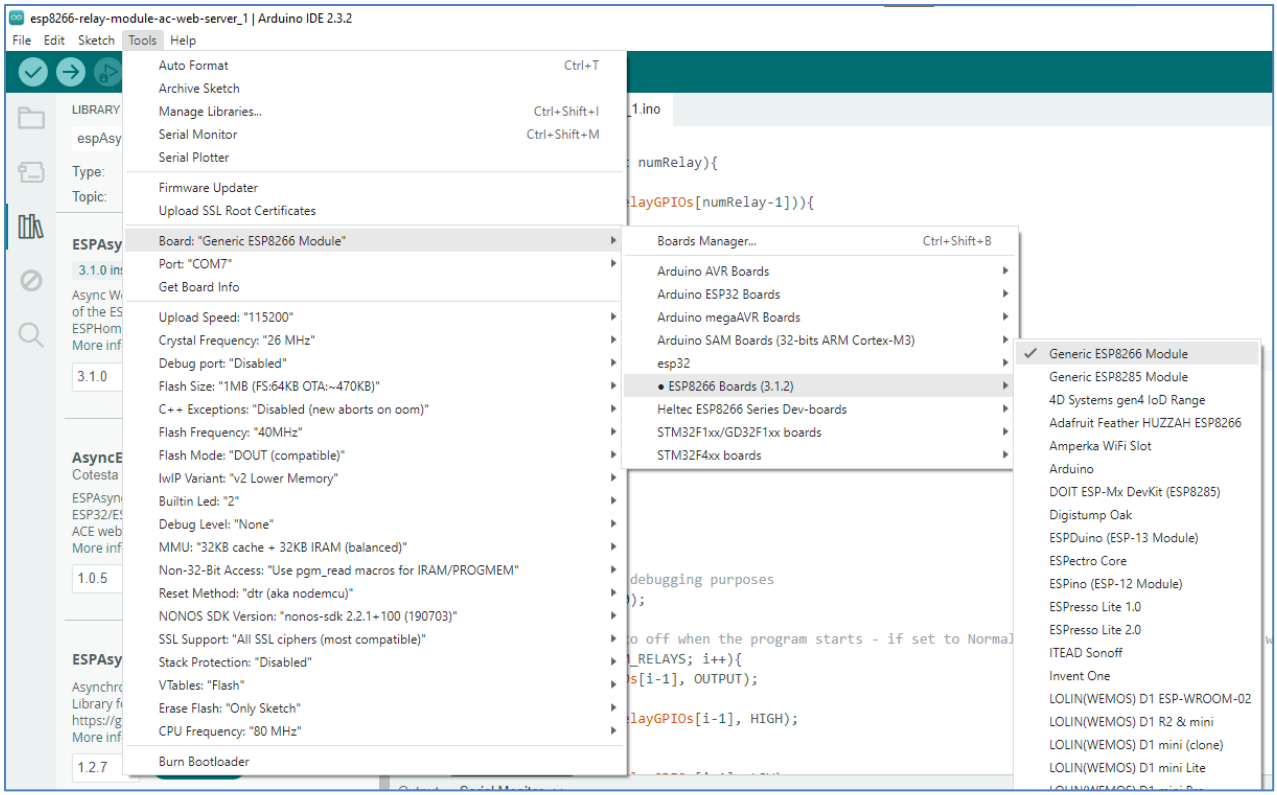


Şekil: 2

- **Şekil 2'de gösterildiği gibi FT232 modül ile ürün arasında bağlantıyı yapın, siyah jumper'ı program yükleme moduna geçiş için takın.**
- **Arduino-IDE'de Tools → "Manage Libraries" – "Kütüphaneleri Yönet" sekmesini açın. "ESPAsyncWebServer" kütüphanesini aratın ve kurun.**



- **Board olarak Generic ESP8266 Module seçeneğini seçin.**



- **Programı derleyin ve yükleyin.**
- **Siyah renkli jumper'ı çıkarın, ürünü reset'leyin. (Enerjisini kesip, tekrar verin)**
- **Arduino IDE – Serial Monitor ekranında programın mesajlarını izleyin.**
- **Ürün SSID ve şifresini girdiğiniz WI-FI hizmet noktasına bağlanıp, IP alacaktır. Bu IP ekranda yazılır. (Şekil 3)**

```
connected with DIRENC.NET, channel 11
dhcp client start...
ip:10.34.35.42,mask:255.255.255.0,gw:10.34.35.1
Connecting to WiFi..
10.34.35.42
```

Şekil: 3

- **Cihazınızdan FT-232 bağlantılarını sökün, cihazı 5V-30V arası bir gerilim ile besleyin.**
- **Yaklaşık 1 dakika cihazın IP adresi almasını bekleyin.**
- **Ekrandan aldığınız IP'yi browser'ınızda (Google-Chrome vb.) hedef sunucu olarak yazarak enter'a basın. (Örneğimizde 10.34.35.42)**
- **Aşağıdaki Şekil 4'de gösterilen formatta bir ekran göreceksiniz.**
- **Tuşları sağa sola kaydırdığınızda röle seslerini duyacaksınız.**
- **Sisteminiz çalışıyordur.**
- **Bu program üzerinde değişiklikler yaparak uygulamayı zenginleştirebilirsiniz.**

Şekil: 4

Ek-1

```
/*********
Rui Santos
Complete project details at https://RandomNerdTutorials.com/esp8266-relay-module-ac-web-server/
The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.
*********/
// Import required libraries
#include "ESP8266WiFi.h"
#include "ESPAsyncWebServer.h"

// Set to true to define Relay as Normally Open (NO)
#define RELAY_NO    true

// Set number of relays
#define NUM_RELAYS  1

// Assign each GPIO to a relay
int relayGPIOs[NUM_RELAYS] = {4};

// Replace with your network credentials
const char* ssid = "DIRENC";
const char* password = "DIRENC";

const char* PARAM_INPUT_1 = "relay";
const char* PARAM_INPUT_2 = "state";

// Create AsyncWebServer object on port 80
AsyncWebServer server(80);

const char index_html[] PROGMEM = R"rawliteral(
<!DOCTYPE HTML><html>
<head>
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <style>
    html {font-family: Arial; display: inline-block; text-align: center;}
    h2 {font-size: 3.0rem;}
    p {font-size: 3.0rem;}
    body {max-width: 600px; margin:0px auto; padding-bottom: 25px;}
    .switch {position: relative; display: inline-block; width: 120px; height: 68px}
    .switch input {display: none}
    .slider {position: absolute; top: 0; left: 0; right: 0; bottom: 0; background-color: #ccc; border-radius: 34px}
    .slider:before {position: absolute; content: ""; height: 52px; width: 52px; left: 8px; bottom: 8px; background-
color: #fff; -webkit-transition: .4s; transition: .4s; border-radius: 68px}
    input:checked+.slider {background-color: #2196F3}
    input:checked+.slider:before {-webkit-transform: translateX(52px); -ms-transform: translateX(52px); transform:
translateX(52px)}
  </style>
</head>
<body>
  <h2>ESP Web Server</h2>
  %BUTTONPLACEHOLDER%
<script>function toggleCheckbox(element) {
```

```
  var xhr = new XMLHttpRequest();
  if(element.checked){ xhr.open("GET", "/update?relay="+element.id+"&state=1", true); }
  else { xhr.open("GET", "/update?relay="+element.id+"&state=0", true); }
  xhr.send();
}</script>
</body>
</html>
)rawliteral";

// Replaces placeholder with button section in your web page
String processor(const String& var){
  //Serial.println(var);
  if(var == "BUTTONPLACEHOLDER"){
    String buttons ="";
    for(int i=1; i<=NUM_RELAYS; i++){
      String relayStateValue = relayState(i);
      buttons+= "<h4>Relay #" + String(i) + " - GPIO " + relayGPIOs[i-1] + "</h4><label class=\"switch\"><input
type=\"checkbox\" onchange=\"toggleCheckbox(this)\" id=\"" + String(i) + "\" "+ relayStateValue +"><span
class=\"slider\"></span></label>";
    }
    return buttons;
  }
  return String();
}

String relayState(int numRelay){
  if(RELAY_NO){
    if(digitalRead(relayGPIOs[numRelay-1])){
      return "";
    }
    else {
      return "checked";
    }
  }
  else {
    if(digitalRead(relayGPIOs[numRelay-1])){
      return "checked";
    }
    else {
      return "";
    }
  }
  return "";
}

void setup(){
  // Serial port for debugging purposes
  Serial.begin(115200);

  // Set all relays to off when the program starts - if set to Normally Open (NO), the relay is off when you set the
relay to HIGH
  for(int i=1; i<=NUM_RELAYS; i++){
    pinMode(relayGPIOs[i-1], OUTPUT);
    if(RELAY_NO){
```

```
      digitalWrite(relayGPIOs[i-1], HIGH);
    }
    else{
      digitalWrite(relayGPIOs[i-1], LOW);
    }
  }
  // Connect to Wi-Fi
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting to WiFi..");
  }
  // Print ESP8266 Local IP Address
  Serial.println(WiFi.localIP());
  // Route for root / web page
  server.on("/", HTTP_GET, [](AsyncWebServerRequest *request){
    request->send_P(200, "text/html", index_html, processor);
  });
  // Send a GET request to <ESP_IP>/update?relay=<inputMessage>&state=<inputMessage2>
  server.on("/update", HTTP_GET, [] (AsyncWebServerRequest *request) {
    String inputMessage;
    String inputParam;
    String inputMessage2;
    String inputParam2;
    // GET input1 value on <ESP_IP>/update?relay=<inputMessage>
    if (request->hasParam(PARAM_INPUT_1) & request->hasParam(PARAM_INPUT_2)) {
      inputMessage = request->getParam(PARAM_INPUT_1)->value();
      inputParam = PARAM_INPUT_1;
      inputMessage2 = request->getParam(PARAM_INPUT_2)->value();
      inputParam2 = PARAM_INPUT_2;
      if(RELAY_NO){
        Serial.print("NO ");
        digitalWrite(relayGPIOs[inputMessage.toInt()-1], !inputMessage2.toInt());
      }
      else{
        Serial.print("NC ");
        digitalWrite(relayGPIOs[inputMessage.toInt()-1], inputMessage2.toInt());
      }
    }
    else {
      inputMessage = "No message sent";
      inputParam = "none";
    }
    Serial.println(inputMessage + inputMessage2);
    request->send(200, "text/plain", "OK");
  });
  // Start server
  server.begin();
}
  void loop() {

}
```