LAMPIRAN

Gambar 1 Tampilan IFTT Untuk Perintah Mematikan Kipas Angin Pada Google Assistant

Gambar 2 Tampilan IFTT Untuk Perintah Menghidupkan Lampu Pertama Pada Google Assistant

Gambar 3 Tampilan IFTT Untuk Perintah Mematikan Lampu Pertama Pada Google Assistant

Gambar 4 Tampilan IFTT Untuk Perintah Menghidupkan Lampu Kedua Pada Google Assistant

Gambar 5 Tampilan IFTT Untuk Perintah Mematikan Lampu Kedua Pada Google Assistant

Gambar 6 Tampilan IFTT Untuk Perintah Menghidupkan Semua Perangkat Pada Google Assistant

Gambar 7 Tampilan IFTT Untuk Perintah Mematikan Lampu Pada Google Assistant
Gambar 8 Hasil Pengukuran Konektifitas Google Assistant Terhadap Lampu Pertama ON

Gambar 9 Hasil Pengukuran Konektifitas Google Assistant Terhadap Lampu Pertama OFF

Gambar 10 Hasil Pengukuran Konektifitas Google Assistant Terhadap Lampu Kedua ON

Gambar 11 Hasil Pengukuran Konektifitas Google Assistant Terhadap Lampu Kedua OFF

Gambar 12 Hasil Pengukuran Konektifitas Google Assistant Terhadap Kipas Angin ON

Gambar 13 Hasil Pengukuran Konektifitas Terhadap Google Assistant Terhadap Kipas Angin OFF

Gambar 14 Hasil Pengukuran Konektifitas Terhadap Google Assistant Terhadap Semua Perangkat OFF
Coding
#include <ESP8266WiFi.h>
#include "Adafruit_MQTT.h"
#include "Adafruit_MQTT_Client.h"

#define WIFI_SSID "Lahacienda"
#define WIFI_PASS "12345678"

#define MQTT_SERV "io.adafruit.com"
#define MQTT_PORT 1883
#define MQTT_NAME "Zenobia"
#define MQTT_PASS "700fbb89556e42d7a65f31edc0b0c362"

int kps = D2;
int lamp1 = D3;
int lamp2 = D4;

WiFiClient client;
Adafruit_MQTT_Client mqtt(&client, MQTT_SERV, MQTT_PORT,
MQTT_NAME, MQTT_PASS);

Adafruit_MQTT_Subscribe Kipas = Adafruit_MQTT_Subscribe(&mqtt,
MQTT_NAME "/f/Kipas");
Adafruit_MQTT_Publish StatusKipas = Adafruit_MQTT_Publish(&mqtt,
MQTT_NAME "/f/StatusKipas");

Adafruit_MQTT_Subscribe Lampu1 = Adafruit_MQTT_Subscribe(&mqtt,
MQTT_NAME "/f/Lampu1");
Adafruit_MQTT_Publish StatusLampu1 = Adafruit_MQTT_Publish(&mqtt,
MQTT_NAME "/f/StatusLampu1");
void setup()
{
    Serial.begin(9600);

    //Connect to WiFi
    Serial.print("\n\nConnecting Wifi>");
    WiFi.begin(WIFI_SSID, WIFI_PASS);

    while (WiFi.status() != WL_CONNECTED)
    {
        Serial.print(">");
        delay(50);
    }

    Serial.println("OK!");

    //Subscribe to the onoff topic
    mqtt.subscribe(&Kipas);
    mqtt.subscribe(&Lampu1);
    mqtt.subscribe(&Lampu2);
}
mqtt.subscribe(&KontrolONOFF);

pinMode(kps, OUTPUT);
pinMode(lamp1, OUTPUT);
pinMode(lamp2, OUTPUT);
digitalWrite(kps,HIGH);
digitalWrite(lamp1,HIGH);
digitalWrite(lamp2,HIGH);
}

void loop()
{
    //Connect/Reconnect to MQTT
    MQTT_connect();

    //Read from our subscription queue until we run out, or
    //wait up to 5 seconds for subscription to update
    Adafruit_MQTT_Subscribe * subscription;
    while ((subscription = mqtt.readSubscription(5000)))
    {
        if (subscription == &Kipas){
            Serial.print("Lamp Red: ");
            Serial.println((char*) Kipas.lastread);
            if (!strcmp((char*) Kipas.lastread, "ON")){
                digitalWrite(kps, LOW);
                StatusKipas.publish("ON");
            }
            else if (!strcmp((char*) Kipas.lastread, "OFF")){
                digitalWrite(kps, HIGH);
                StatusKipas.publish("OFF");
            }
        }
    }
}
else{
    StatusKipas.publish("ERROR");
}

if (subscription == &Lampu1){
    Serial.print("Lamp Yellow: ");
    Serial.println((char*) Lampu1.lastread);
    if (!strcmp((char*) Lampu1.lastread, "ON")){
        digitalWrite(lamp1, LOW);
        StatusLampu1.publish("ON");
    } else if (!strcmp((char*) Lampu1.lastread, "OFF")){
        digitalWrite(lamp1, HIGH);
        StatusLampu1.publish("OFF");
    } else{
        StatusLampu1.publish("ERROR");
    }
}

if (subscription == &Lampu2){
    Serial.print("Lamp Red: ");
    Serial.println((char*) Lampu2.lastread);
    if (!strcmp((char*) Lampu2.lastread, "ON")){
        digitalWrite(lamp2, LOW);
        StatusLampu2.publish("ON");
    } else if (!strcmp((char*) Lampu2.lastread, "OFF")){
        digitalWrite(lamp2, HIGH);
        StatusLampu2.publish("OFF");
    } else{
        StatusLampu2.publish("ERROR");
    }
if (subscription == &KontrolONOFF){
    Serial.print("Kondisi Semua Lampu : ");
    Serial.println((char*) KontrolONOFF.lastread);
    if (!strcmp((char*) KontrolONOFF.lastread, "ON")){
        digitalWrite(kps, LOW);
        digitalWrite(lamp1, LOW);
        digitalWrite(lamp2, LOW);
        StatusONOFF.publish("ON");
    }
    else if (!strcmp((char*) KontrolONOFF.lastread, "OFF")){
        digitalWrite(kps, HIGH);
        digitalWrite(lamp1, HIGH);
        digitalWrite(lamp2, HIGH);
        StatusONOFF.publish("OFF");
    }
    else{
        StatusONOFF.publish("ERROR");
    }
}

void MQTT_connect()
{
    // Stop if already connected
    if (mqtt.connected() && mqtt.ping())
    {
    
    }
// mqtt.disconnect();
return;
}

int8_t ret;

mqtt.disconnect();

Serial.print("Connecting to MQTT... ");
uint8_t retries = 3;
while ((ret = mqtt.connect()) != 0) // connect will return 0 for connected
{
    Serial.println(mqtt.connectErrorString(ret));
    Serial.println("Retrying MQTT connection in 5 seconds...");
    mqtt.disconnect();
    delay(5000); // wait 5 seconds
    retries--;
    if (retries == 0)
    {
        ESP.reset();
    }
}
Serial.println("MQTT Connected!");
4 Channel 5V Optical Isolated Relay Module

This is a LOW Level 5V 4-channel relay interface board, and each channel needs a 15-20mA driver current. It can be used to control various appliances and equipment with large current. It is equipped with high-current relays that work under AC250V 10A or DC30V 10A. It has a standard interface that can be controlled directly by microcontroller. This module is optically isolated from high voltage side for safety requirement and also prevent ground loop when interface to microcontroller.

Brief Data:

- Relay Maximum output: DC 30V/10A, AC 250V/10A.
- 4 Channel Relay Module with Opto-coupler. LOW Level Trigger expansion board, which is compatible with Arduino control board.
- Standard interface that can be controlled directly by microcontroller (8051, AVR, *PIC, DSP, ARM, ARM, MSP430, TTL logic).
- Relay of high quality low noise relays SPDT. A common terminal, a normally open, one normally closed terminal.
- Opto-Coupler isolation, for high voltage safety and prevent ground loop with microcontroller.
VCC and RY-VCC are also the power supply of the relay module. When you need to drive a large power load, you can take the jumper cap off and connect an extra power to RY-VCC to supply the relay; connect VCC to 5V of the MCU board to supply input signals.

NOTES: If you want complete optical isolation, connect "Vcc" to Arduino +5 volts but do NOT connect Arduino Ground. Remove the Vcc to JD-Vcc jumper. Connect a separate +5 supply to "JD-Vcc" and board Gnd. This will supply power to the transistor drivers and relay coils.

If relay isolation is enough for your application, connect Arduino +5 and Gnd, and leave Vcc to JD-Vcc jumper in place.
It is sometimes possible to use this relay boards with 3.3V signals, if the JD-VCC (Relay Power) is provided from a +5V supply and the VCC to JD-VCC jumper is removed. That 5V relay supply could be totally isolated from the 3.3V device, or have a common ground if opto-isolation is not needed. If used with isolated 3.3V signals, VCC (To the input of the opto-isolator, next to the IN pins) should be connected to the 3.3V device's +3.3V supply.

NOTE: Some Raspberry-Pi users have found that some relays are reliable and others do not actuate sometimes. It may be necessary to change the value of R1 from 1000 ohms to something like 220 ohms, or supply +5V to the VCC connection.

NOTE: The digital inputs from Arduino are Active LOW: The relay actuates and LED lights when the input pin is LOW, and turns off on HIGH.

**Module Layout:**

![宗图](image)

**Operating Principle:**

See the picture below: A is an electromagnet, B armature, C spring, D moving contact, and E fixed contacts. There are two fixed contacts, a normally closed one and a normally open one. When the coil is not energized, the normally open contact is the one that is off, while the normally closed one is the other that is on.
Supply voltage to the coil and some currents will pass through the coil thus generating the electromagnetic effect. So the armature overcomes the tension of the spring and is attracted to the core, thus closing the moving contact of the armature and the normally open (NO) contact or you may say releasing the former and the normally closed (NC) contact. After the coil is de-energized, the electromagnetic force disappears and the armature moves back to the original position, releasing the moving contact and normally closed contact. The closing and releasing of the contacts results in power on and off of the circuit.

**Input:**

VCC : Connected to positive supply voltage (supply power according to relay voltage) GND : Connected to supply ground.

IN1: Signal triggering terminal 1 of relay module
IN2: Signal triggering terminal 2 of relay module
IN3: Signal triggering terminal 3 of relay module
IN4: Signal triggering terminal 4 of relay module

**Output:**

Each module of the relay has one NC (normally close), one NO (normally open) and one COM (Common) terminal. So there are 4 NC, 4 NO and 4 COM of the channel relay in total. NC stands for the normal close port contact and the state without power. NO stands for the normal open port contact and the state with power. COM means the common port. You can choose NC port or NO port according to whether power or not.

**Testing Setup:**

When a low level is supplied to signal terminal of the 4-channel relay, the LED at the output terminal will light up. Otherwise, it will turn off. If a periodic high and low level is supplied to the signal terminal, you can see the LED will cycle between on and off.

For Arduino

1. Step 1:
   Connect the signal terminal IN1, IN2, IN3 & IN4 of 4-channel relay to digital pin 4, 5, 6, 7 of the Arduino Uno or ATMega2560 board, and connect an LED at the output terminal.

IN1> 4; IN2> 5; IN3>6; IN4>7

Step 2:
Upload the sketch "4 Channel Relay Demo " to the Arduino Uno or ATmega2560 board. Then you can see the LED cycle between on and off.

The actual figure is shown below:

Arduino Sketch: 4 Channel Relay Demo

/******************************************
Name: 4 channel_relay
Description: control the 4 channel relay module to ON or OFF
Website: www.handsontec.com
Email: techsupport@handsontec.com
******************************************/

//the relays connect to

int RelayControl1 = 4;    // Digital Arduino Pin used to control the motor
int RelayControl2 = 5;
int RelayControl3 = 6;
int RelayControl4 = 7;

void setup()
{
    Serial.begin(9600);
    pinMode(RelayControl1, OUTPUT);
    pinMode(RelayControl2, OUTPUT);
    pinMode(RelayControl3, OUTPUT);
    pinMode(RelayControl4, OUTPUT);
}

void loop()
{

digitalWrite(RelayControl1, HIGH); // NO1 and COM1 Connected (LED on)
delay(1000);
digitalWrite(RelayControl1, LOW); // NO1 and COM1 disconnected (LED off) delay(1000);
digitalWrite(RelayControl2, HIGH);
delay(1000);
digitalWrite(RelayControl2, LOW);
delay(1000);
digitalWrite(RelayControl3, HIGH);
delay(1000);
digitalWrite(RelayControl3, LOW);
delay(1000);
digitalWrite(RelayControl4, HIGH);
delay(1000);
digitalWrite(RelayControl4, LOW);
delay(1000);
}
**NodeMCU ESP8266 ESP-12E WiFi Development Board**

NodeMCU is an open source IoT platform. It includes firmware which runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module. The term "NodeMCU" by default refers to the firmware rather than the DevKit. The firmware uses the Lua scripting language. It is based on the eLua project, and built on the Espressif Non-OS SDK for ESP8266. It uses many open source projects, such as lua-cjson, and spiffs.

**Features**

Version: DevKit v1.0 Breadboard Friendly

- Light Weight and small size.
- 3.3V operated, can be USB powered.
- Uses wireless protocol 802.11b/g/n. Built-in wireless connectivity capabilities.
- Built-in PCB antenna on the ESP-12E chip.
- Supports Lua (alike node.js) and Arduino C programming language.

**PINOUT DIAGRAM**

NodeMCU ESP8266 v1.0
Source
https://iotbytes.wordpress.com/nodemcu-pinout/

All GPIO runs at 3.3V!!
## Specifications of ESP-12E WiFi Module

<table>
<thead>
<tr>
<th>Wireless Standard</th>
<th>IEEE 802.11 b/g/n</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency Range</td>
<td>2.412 - 2.484 GHz</td>
</tr>
<tr>
<td>Power Transmission</td>
<td></td>
</tr>
<tr>
<td>802.11b : +16 ± 2 dBm (at 11 Mbps)</td>
<td></td>
</tr>
<tr>
<td>802.11g : +14 ± 2 dBm (at 54 Mbps)</td>
<td></td>
</tr>
<tr>
<td>802.11n : +13 ± 2 dBm (at HT20, MCS7)</td>
<td></td>
</tr>
<tr>
<td>802.11b : -93 dBm (at 11 Mbps, CCK)</td>
<td></td>
</tr>
<tr>
<td>Receiving Sensitivity</td>
<td></td>
</tr>
<tr>
<td>802.11g : -85 dBm (at 54 Mbps, OFDM)</td>
<td></td>
</tr>
<tr>
<td>Wireless Form</td>
<td></td>
</tr>
<tr>
<td>802.11n : -82 dBm (at HT20, MCS7)</td>
<td></td>
</tr>
<tr>
<td>IO Capability</td>
<td></td>
</tr>
<tr>
<td>UART, I2C, PWM, GPIO, 1 ADC</td>
<td></td>
</tr>
<tr>
<td>Electrical Characteristic</td>
<td>On-board PCB Antenna</td>
</tr>
<tr>
<td></td>
<td>UART, I2C, PWM, GPIO, 1 ADC</td>
</tr>
<tr>
<td></td>
<td>3.3 V Operated</td>
</tr>
<tr>
<td>Operating Temperature</td>
<td></td>
</tr>
<tr>
<td>15 mA output current per GPIO pin</td>
<td></td>
</tr>
<tr>
<td>12 - 200 mA working current</td>
<td></td>
</tr>
<tr>
<td>Less than 200 uA standby current</td>
<td></td>
</tr>
<tr>
<td>-40 to +125 ºC</td>
<td></td>
</tr>
<tr>
<td>Serial Transmission</td>
<td>110 - 921600 bps, TCP Client 5</td>
</tr>
<tr>
<td>Wireless Network Type</td>
<td>Security Type Encryption Type Firmware Upgrade Network Protocol User Configuration</td>
</tr>
</tbody>
</table>
Local Serial Port, OTA Remote Upgrade IPv4, TCP / UDP / FTP / HTTP, AT + Order Set, Web Android / iOS, Smart Link APP

Disclaimer
Information provided in this document are compilation from various online resources. Einstronic Enterprise does not ensure the completeness, accuracy and reliability of the information and do not own any rights on any registered trademarks involved. Information provided should be intended for references only.