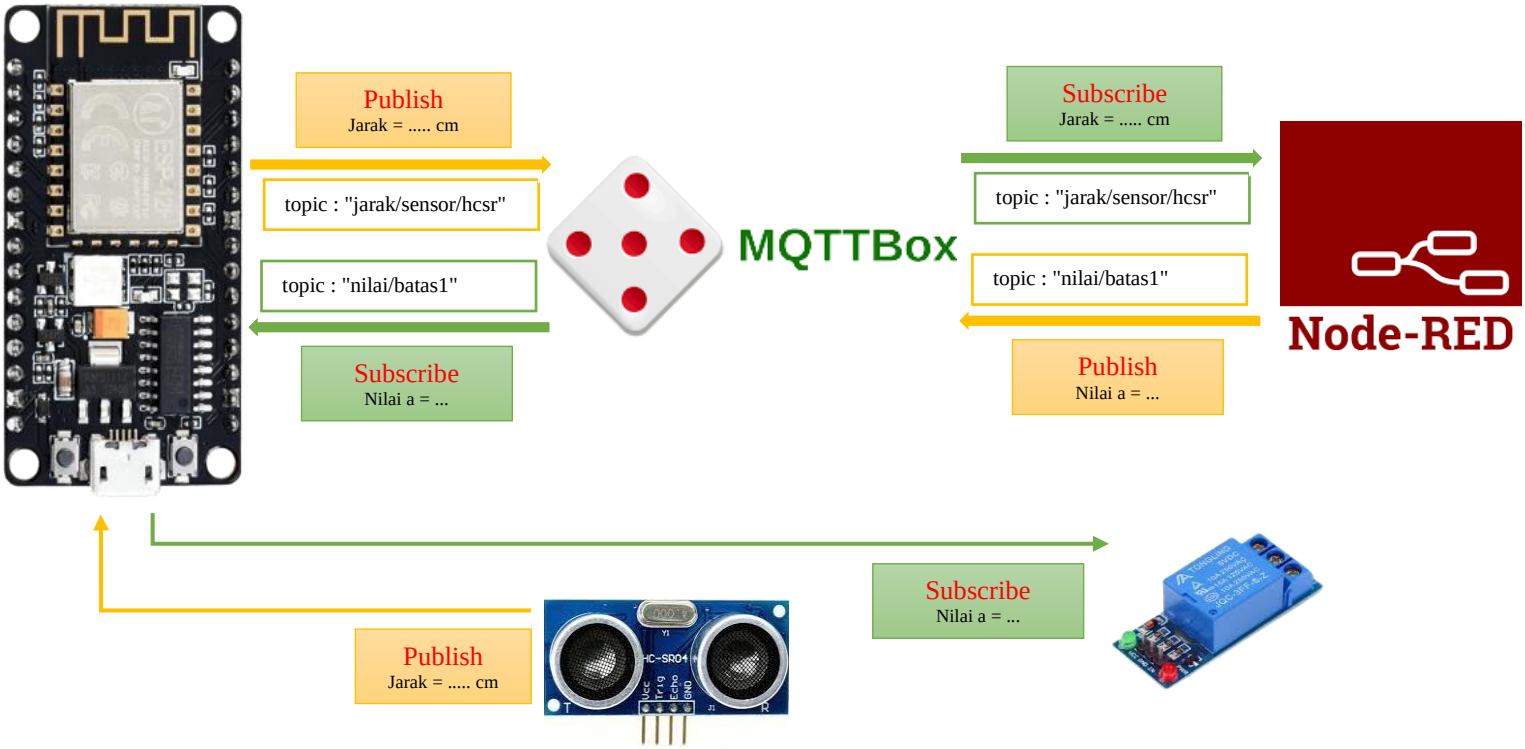
SENSOR HCSR-04 PENGENDALI RELAY

By : TEGUH MUSTOFA/20202227



Keterangan :

Jika jarak kurang dari “a” maka relay mati dan jika jarak lebih dari “a” Relay hidup

Nilai “a” didapat dari inputan node red yang kemudian disimpan di EEPROM

Cara kerja:

Pada awalnya sensor HCSR-04 mkurengu jarak, kemudian lewat node-mcu nilai tersebut dikirim ke MQTTBox, setelah niai terupload maka nilai diunduh oleh Node-red kemudian ditampilkan di interface nya, disisi lain selain node menampilkan nilai yang diambil dari MQTTBox, Node-red juga menginputkan nilai yang kemudian diupload juga ke MQTTbox, kita sebut saja nilai itu sebagai nilai “a”, lalu apa fungsi nilai tersebut, yakni tak lain sebagai pembanding nilai jarak yang mempengarui mati dan hidupnya relay, sebagai contoh : jika jarak kurang dari “a” maka relay mati.

PROGRAM

```
#include <Arduino.h>
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include <NewPing.h> //Library untuk HC-SR04
#include <EEPROM.h>  // library untuk EEPROM

#define TRIGGER_PIN  D5  //Pin Trigger HC-SR04 pada NodeMCU
#define ECHO_PIN     D6  //Pin Echo HC-SR04 pada NodeMCU
#define MAX_DISTANCE 500 //Maksimum Pembacaan Jarak (cm)
#define RELAY D4         //Definisi pin relay
#define EEPROM_SIZE 256  //Ukuran size pada EEPROM
int a,b,c;               // variabel untuk menyiman nilai inputan dari node-red
long jarak;              //variabel untuk menyimpan hasil pembacaan sensor jarak
//MQTT Topic
const char* MQTT_JARAK = "jarak/sensor/hcsr";   // topic untuk menghubungkan ke node-red, sebagai pemyimpan nilai jarak (publish)
const char* SUB_NILAI1 = "nilai/batas1";        // topic untuk menghubungkan ke node-red, sebagai pemyimpan nilai batas 1 (subcribe)
const char* SUB_NILAI2 = "nilai/batas2";        // topic untuk menghubungkan ke node-red, sebagai pemyimpan nilai batas 2 (subcribe)
const char* SUB_NILAI3 = "nilai/batas3";        // topic untuk menghubungkan ke node-red, sebagai pemyimpan nilai batas 3 (subcribe)
const char* ssid = "Plezz9";                    // untuk mengubungkan ke wifi dengan SSID "Plezz9"
const char* password = "adgjm1922";                     // untuk mengubungkan ke wifi dengan PW "adgjm1922"
const char* MQTT_SERVER = "broker.mqtt-dashboard.com";      // mqtt sebagi server
unsigned long startMillis=0;                    //variabel menyimpan milidetik terakhir dari loop
//unsigned long currentMillisPublishTemp;


WiFiClient espClient;
PubSubClient client(espClient);
NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);         //Setup Pin HC-SR04 dan Jarak Pembacaan dengan Fungsi Library


void setup_wifi() {
```

```
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  randomSeed(micros());

  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());

}

void HCSR_config(){
 long jarak = sonar.ping_cm();
    unsigned long currentMillis = millis();          // variabel mengambil waktu yang berjalan dan menyimpan di variabel milis_sekarang
  if (currentMillis - startMillis >= 3000) {          //setiap milis_sekarang - hitungan_milis yang mencapai nilai lebih besar atau sama dengan 2000

     //Mencetak Hasil Pembacaan pada Serial Monitor
      Serial.println("Monitoring Jarak");
      Serial.print("Jarak: ");
      Serial.print(jarak);
      Serial.println(" cm");
      delay(1000);
 client.publish(MQTT_JARAK, String(jarak).c_str());
```

```
if (jarak >= (EEPROM.read(1)))                    // nilai EEPROM sebagai batas jarak
       {
         digitalWrite(D4, HIGH);                  //perintah untuk mematikan Relay
         Serial.println("RELAY MATI");             //menampilakan kondisi relay
       }
if (jarak < (EEPROM.read(1)))                     // nilai EEPROM sebagai batas jarak
       {
         digitalWrite(D4, LOW);                   //perintah untuk menghidupkann Relay
         Serial.println("RELAY HIDUP");            //menampilakan kondisi relay
       }
  }
}
void callback(char *topic, byte *message, unsigned int length)
{
  Serial.print("Message arrived in topic: ");
  Serial.println(topic);
  Serial.print("Message:");
  String payload;

 for (int i = 0; i < length; i++)
  {
    Serial.print((char)message[i]);
    Serial.println();
    if ((char)message[i] != "")
      payload += (char)message[i];
  }
  if (String(topic) == "nilai/batas1")
  {
      a = payload.toInt();              //merubah nilai inputan dari node-red ke bentuk integer kemudian disimpan
variable "a"
      Serial.print("Nilai A = ");
      Serial.println(a);                //menampilkan nilai "a"
      EEPROM.write(1 , a);              //memasukkan Nilai "a" kedalam EEPROM dengan Address (1)
      EEPROM.commit();                  //perintah untuk mengeksekusi penginputan nilai
```

```
  }
  if (String(topic) == "nilai/batas2")
  {
     b = payload.toInt();              //merubah nilai inputan dari node-red ke bentuk integer kemudian disimpan variable "b"
     Serial.print("Nilai B = ");
     Serial.println(b);               //menampilkan nilai "b"
     EEPROM.write(2 , b);             //memasukkan Nilai "b" kedalam EEPROM dengan Address (2)
     EEPROM.commit();                 //perintah untuk mengeksekusi penginputan nilai
  }
  if (String(topic) == "nilai/batas3")
  {
     c = payload.toInt();              //merubah nilai inputan dari node-red ke bentuk integer kemudian disimpan variable "C"
     Serial.print("Nilai C = ");
     Serial.println(c);               //menampilkan nilai "C"
     EEPROM.write(4 , c);             //memasukkan Nilai "C" kedalam EEPROM dengan Address (4)
     EEPROM.commit();                 //perintah untuk mengeksekusi penginputan nilai
  }
}


void reconnect()
{
  // Loop until we're reconnected
  while (!client.connected())
  {
 Serial.print("Attempting MQTT connection...");
  String clientId = "belajarmqtt";
    clientId += String(random(0xffff), HEX);
    //Attempt to connect
    if (client.connect("espClient"))
    {
      Serial.println("connected");
      client.subscribe("nilai/batas1");     //perintah untuk mendapatkan nilai 1 dari node-red ketika connect
      client.subscribe("nilai/batas2");     //perintah untuk mendapatkan nilai 2 dari node-red ketika connect
```

```
      client.subscribe("nilai/batas3");     //perintah untuk mendapatkan nilai 3 dari node-red ketika connect
    }
    else
    {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      delay(5000);
    }
  }
}

void setup(){
 Serial.begin(115200);              //jenis srial monitor
 EEPROM.begin(EEPROM_SIZE);         //memaggil fungsi EEPROM
 pinMode(TRIGGER_PIN, OUTPUT);      //mendeklarasikan pin  sebagai OUTPUT
 pinMode(ECHO_PIN, INPUT);          //mendeklarasikan pin  sebagai INPUT
 pinMode(D4, OUTPUT);               //mendeklarasikan pin  sebagai OUTPUT
 EEPROM.read(1);                    //menginputkan nilai ke EEPROM dengan ADDRES (1);
 EEPROM.read(2);                    //menginputkan nilai ke EEPROM dengan ADDRES (2);
 EEPROM.read(4);                    //menginputkan nilai ke EEPROM dengan ADDRES (4);
 setup_wifi();
  client.setCallback(callback);         //memanggil fungsi callback
  client.setServer(MQTT_SERVER, 1883);  //mendeklarasikan mqtt server dan portnya
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
  WiFi.setAutoReconnect(true);
  WiFi.persistent(true);

}

void loop(){
  HCSR_config();                     //memanggil funtcion HCSR_config
```

```
      Serial.print("Nilai A = ");
      Serial.println(b);
      Serial.print("Nilai B = ");
      Serial.println(b);
      Serial.print("Nilai C = ");
      Serial.println(c);
  Serial.print("Nilai EEPROM A");
  Serial.println(EEPROM.read(1));
  Serial.print("Nilai EEPROM B");
  Serial.println(EEPROM.read(2));
  Serial.print("Nilai EEPROM C");
  Serial.println(EEPROM.read(4));
    if (!client.connected())
    {
      reconnect();
    }
    client.loop();
}
```