LAPORAN PRAKTIKUM

ELEKTRONIKA MESIN LISTRIK DAN TEKNIK KENDALI

"*Cloud Arduino IoT*"

Dosen Pengampu: Irfan Bahiuddin, ST, M.Phil., Ph.D.

Disusun Oleh:

Kelompok 4

Chaesar Syaefuddin (19/441195/SV/16547)

Yeyen Karunia (19/441215/SV/16567)

Kelas: ARM 2

DEPARTEMEN TEKNIK MESIN

SEKOLAH VOKASI

UNIVERSITAS GADJAH MADA

YOGYAKARTA

2022

BAB I. DESKRIPSI KASUS

Pada kasus ini bertujuan untuk mengidentifikasi cara mentransfer data ke Platform IoT Arduino Cloud IoT serta bagaimana melakukan perintah kepada aktuator yang berada jauh dengan tempat kita namun masih ada internet pada tempat tersebut. Pada project kali ini topik utamanya adalah control lampu dan pembacaan nilai suhu dan kelembaban menggunakan sensor DHT11 melalui Platform IoT – Arduino cloud IoT.

BAB II. KOMPONEN YANG DIGUNAKAN

2.1 Mikrokontroller ESP32

ESP 32 adalah mikrokontroler yang dikenalkan oleh Espressif System merupakan penerus dari mikrokontroler ESP8266. Pada mikrokontroler ini sudah tersedia modul WiFi dalam chip sehingga sangat mendukung untuk membuat sistem aplikasi *Internet of Things*. Terlihat pada gambar dibawah merupakan pin out dari ESP32. Pin tersebut dapat dijadikan input atau output untuk menyalakan LCD, lampu, bahkan untuk menggerakan motor DC. Pada mikrokontroler ini sudah tersedia modul wifi dan bluetooth sehingga sangat mendukung untuk membuat sistem aplikasi Internet of Things. Memiliki 18 ADC (*Analog Digital Converter*), 2 DAC, 16 PWM, 10 Sensor sentuh, 2 jalur antarmuka UART,pin antarmuka I2C, I2S, dan SPI.

Gambar 1. Mikrokontroller Esp32

ESP32 menggunakan prosesor dual core yang berjalan di instruksi Xtensa LX16 [3], ESP32 memiliki spesifikasi seperti yang ditampilkan pada tabel 1.

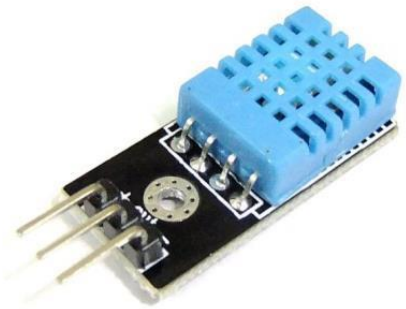
Tabel 1. Spesifikasi Mikrokontroller Esp32

Atribut	**Detail**
CPU	Tensilica Xtensa LX6 32bit Dual-Core di 160/240MHz
SRAM	520KB
FLASH	2MB (max. 64MB)
Tegangan	2.2V sampai 3.6V
Arus Kerja	Rata-rata 80mA
Dapat diprogram	Ya (C, C++, Python, Lua, dll)
Opern Source	Ya
Konektivitas	
Wi-Fi	802.11 b/g/n
Bluetooth	4.2BR/EDR + BLE
UART	3
I/O	
GPIO	32
SPI	4
I2C	2
PWM	8
ADC	18 (12- bit)
DAC	2 (8-bit)

2.2 Sensor DHT11

Sensor DHT11 merupakan sensor dengan kalibrasi sinyal digital yang mampu memberikan informasi suhu dan kelembaban. Sensor ini tergolong komponen yang memiliki tingkat stabilitas yang sangat baik. Produk dengan kualitas terbaik, respon pembacaan yang cepat, dan kemampan anti-interference, dengan harga yang terjangkau. DHT11 memiliki fitur kalibrasi yang sangat akurat. Koefisien kalibrasi ini disimpan dalam OTP program memory, sehingga ketika internal sensor mendeteksi sesuatu suhu atau kelembaban, maka module ini membaca koefisien sensor tersebut. Ukurannya yang kecil, dengan transmisi sinyal

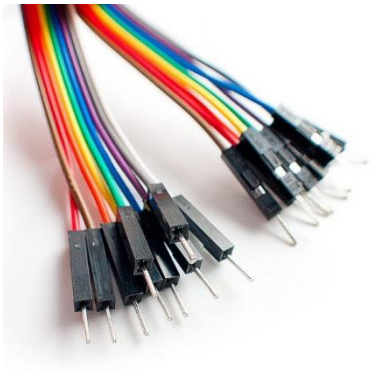
hingga 20 meter, membuat produk ini cocok digunakan untuk banyak aplikasi-aplikasi pengukuran suhu dan kelembaban.



Gambar 2. Sensor DHT11

2.3 Kabel Jumper

Kabel jumper adalah kabel yang di pergunakan untuk menghubungkan satu komponen dengan komponen lain ataupun menghubungkan jalur rangkaian yang terputus pada *breadboard*. Umumnya, kabel jumper digunakan pada *breadboard* atau alat prototyping lainnya agar lebih mudah untuk mengutak-atik rangkaian. Konektor yang ada pada ujung kabel terdiri atas dua jenis yaitu konektor jantan (*male connector*) dan konektor betina (*female connector*). Prinsip kerja kabel jumper yaitu menghantarkan arus listrik dari satu komponen ke komponen lainnya yang saling terhubung.

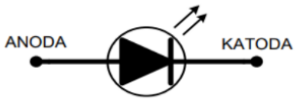


Gambar 3. Kabel Jumper

2.4 LED

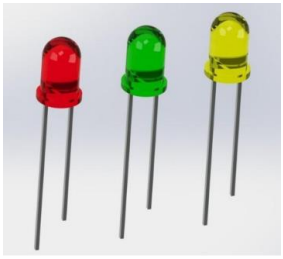
Light Emiting Diode (LED) adalah komponen yang dapat memancarkan cahaya. Struktur LED sama dengan dioda. Untuk mendapatkan pancaran cahaya pada semikonduktor, dopping yang dipakai adalah gallium, arsenic, dan phosporus. Jenis dopping yang berbeda akan menghasilkan warna cahaya yang berbeda.

Bentuk LED bermacam-macam, ada yang bulat, persegi empat dan lonjong. Simbol LED terlihat pada gambar 3.



Gambar 4. Simbol LED

LED memiliki kaki 2 buah seperti dengan dioda yaitu kaki anoda dan kaki katoda. Pada gambar diatas kaki anoda memiliki ciri fisik lebih panjang dari kaki katoda pada saat masih baru, kemudian kaki katoda pada LED ditandai dengan bagian body yang dipapas rata. Pemasangan LED agar dapat menyala adalah dengan memberikan tegangan bias maju yaitu dengan memberikan tegangan positif ke kaki anoda dan tegangan negatif ke kaki katoda. Konsep pembatas arus pada dioda adalah dengan memasangkan resistor secara seri pada salah satu kaki LED.

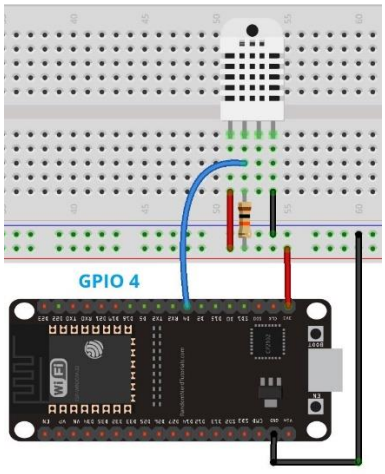


Gambar 5. LED

BAB III. RANGKAIAN PROGRAM

3.1 Rangkaian Esp32 dan Sensor DHT11

Pada kasus ini, rangkaian mikrokontroller Esp32 dan sensor DHT11 akan disusun dengan skema rangkaian seperti dibawah ini.



3.2 Langkah-Langkah Pengerjaan

1. Mempersiapkan material yang diperlukan
2. Melakukan Setup berupa wiring
3. Menghubungkan ESP32 ke laptop menggunakan Kabel USB
4. Membuat akun pada platform Arduino IoT
5. Melakukan install pada Arduino IoT Cloud
6. Mendefinisikan perangkat ESP32
7. Membuat variabel-variabel yang dibutuhkan seperti temperature dan humidity
8. Membuat tampilan dashboard Internet of Things
9. Kemudian, memasukan kode program yang sesuai dengan variabel-variabel yang telah didefinisikan sebelumnya.
10. Masukkan Secret Key dan SSID wifi + password pada opsi network
11. Langkah terakhir yaitu Compiling kode, memastikan bahwa esp terdeteksi, dan jika sudah terkoneksi maka bisa klik Upload sampai terdapat notifikasi Succes Done Uploading

3.2 Kode Program

```
/*
Sketch generated by the Arduino IoT Cloud Thing "Untitled"
https://create.arduino.cc/cloud/things/6a5605af-6d4f-43d7-8c67-a1eb23eab066
Arduino IoT Cloud Variables description
The following variables are automatically generated and updated when changes are made to the Thing
String msg;
CloudTemperatureSensor temperature;
CloudRelativeHumidity humidity;
bool led_switch;
Variables which are marked as READ/WRITE in the Cloud Thing will also have
functions
which are called when their values are changed from the Dashboard.
```

```
These functions are generated with the Thing and added at the end of this
sketch.
*/
#include "thingProperties.h"
#include "DHT.h"
#define DHTpin 4
#define DHTTYPE DHT11
DHT dht(DHTpin,DHTTYPE);
int LED = 13;
void setup() {
pinMode(LED, OUTPUT);
digitalWrite(LED, LOW);
// Initialize serial and wait for port to open:
Serial.begin(9600);
// This delay gives the chance to wait for a Serial Monitor without blocking if
none is found
delay(1500);
// Defined in thingProperties.h
initProperties();
// Connect to Arduino IoT Cloud
ArduinoCloud.begin(ArduinoIoTPreferredConnection);
/*
The following function allows you to obtain more information
related to the state of network and IoT Cloud connection and errors
the higher number the more granular information you'll get.
The default is 0 (only errors).
Maximum is 4
*/
setDebugMessageLevel(2);
ArduinoCloud.printDebugInfo();
}
void loop() {
```

```
ArduinoCloud.update();
// Your code here
dht_sensor_getdata();
CloudRelativeHumidity humidity;
String msg;
CloudTemperatureSensor temperature;
}
/*
Since LedSwitch is READ_WRITE variable, onLedSwitchChange() is
executed every time a new value is received from IoT Cloud.
*/
void onLedSwitchChange() {
// Add your code here to act upon LedSwitch change
if(led_switch){
digitalWrite(LED, LOW);
}
else{
digitalWrite(LED, HIGH);
}
}
/*
Since Humidity is READ_WRITE variable, onHumidityChange() is
executed every time a new value is received from IoT Cloud.
*/
void onHumidityChange() {
// Add your code here to act upon Humidity change
}
/*
Since Msg is READ_WRITE variable, onMsgChange() is
executed every time a new value is received from IoT Cloud.
*/
void onMsgChange() {
```

```
// Add your code here to act upon Msg change
}
void dht_sensor_getdata()
{
float hm = dht.readHumidity();
Serial.print("Humidity ");
Serial.println(hm);
float temp = dht.readTemperature();
Serial.print("Temperature ");
Serial.println(temp);
humidity = hm;
temperature = temp;
msg="Temperature = " + String (temperature)+" Humidity = " +
String(humidity);
}
```

3.3 Hasil Simulasi