SEMINAR NASIONAL TEKNOKA

Vol. 6, 2021
ISSN No. 2502-8782

Perancangan Sistem Otomasi Rumah Tinggal Berbasis Node MCU ESP32

Reza Purnama, Emilia Roza, & Rosalina
Program Studi Teknik Elektro, Fakultas Teknik
Universitas Muhammadiyah Prof. Dr. Hamka
Jl.Tanah Merdeka No. 6, Kp. Rambutan, Ps. Rebo, Jakarta Timur
Telp. (021) 87782739, Fax. (021)8400941
E-mail: zarezapurnama@gmail.com

Abstrak

Sistem otomasi rumah merupakan salah satu penerapan konsep Internet of Things (IoT) yang dapat mengintegrasi dan mengontrol berbagai peralatan pada rumah dengan tujuan meningkatkan kemudahan dan kenyamanan penghuni rumah. Penelitian ini bertujuan untuk merancang sebuah sistem rumah cerdas yang dapat mengendalikan perangkat elektronik yaitu lampu dan kipas. Pada penelitian ini menggunakan beberapa komponen yaitu NodeMCU ESP32 sebagai mikrokontroller, sensor DHT11 sebagai sensor pendeteksi suhu ruangan, sensor LDR sebagai pendeteksi tingkat intensitas cahaya, dan aplikasi Telegram Messenger sebagai perantara komunikasi antara sistem dan juga pengguna. Terdapat dua mode yang terdapat pada sistem rumah cerdas ini, yaitu mode manual dan otomatis. Pada mode manual, pengguna memiliki kendali penuh atas semua proses kontrol dan pemantauan sistem, sedangkan pada mode otomatis segala proses kontrol dilakukan oleh sistem secara otomatis berdasarkan kondisi yang terbaca oleh masing-masing sensor. Proses pengujian pada penelitian ini dilakukan dengan cara mengirimkan perintah kepada sistem melalui aplikasi Telegram Messenger dari jarak 1kilometer hingga 5 kilometer, lalu membandingkan konsumsi daya listrik pada saat active mode dan deep sleep mode. Hasil pengujian menunjukkan bahwa proses pengendalian dan pemantauan sistem rumah cerdas berhasil dilakukan hingga jarak 5kilometer dengan tingkat keberhasilan 100%, rata-rata delay waktu pada mode manual 1,195 detik dan mode otomatis 1.687 detik, lalu rata-rata konsumsi daya pada active mode yaitu 0.808Watt dan deep sleep mode 0.0172 Watt.

Kata kunci*: Otomasi, Internet of Things, NodeMCU ESP32, Telegram Messenger*

Abstract

This experiment aimed to design an intelligent home system to control electronic devices called lights and fans. The automation home system is an application of the Internet of Things (IoT) concept that can integrate and control appliances in the home to improve the conveniences and comforts of the householder. This experiment uses NODEMCU ESP32 as a microcontroller, DHT11 sensors as room temperature sensors, LDR sensors as light-intensity detectors, and a Telegram Messenger application as communications between systems and users. This innovative home system has two modes: manual and automatic mode. The user has complete control over all processes and monitoring systems in manual mode. In contrast, on automatic modes, the system automatically controls all processes based on conditions readable by each sensor. Testing in the experiment was done by sending commands to systems through telegram messenger application from 1 km to 5 km and comparing the consumption of electricity for active mode and deep sleep mode. The result shows that we control and monitor innovative home systems up to 5 km at a 100% success rate, an average time delay on manual mode 1.195 seconds, and an automatic mode of 1,687 seconds. The average power consumption during active mode is 0.808 Watt and during deep sleep mode is 0.0172 Watt.

Keywords*: Automation, Internet of Things, NodeMCU ESP32, Telegram Messenger*

Dipresentasikan pada Tanggal 27 November 2021

1 PENDAHULUAN

Internet of Things (IoT) merupakan suatu ruang lingkup dimana seluruh perangkat yang ada di sekitar kita terhubung ke internet dan dapat berinteraksi yang menyebabkan seluruh informasi dapat diakses dan dikendalikan secara langsung tanpa terkendala jarak dan waktu. Dengan kemudahan yang didapatkan tersebut, maka tentu kita juga dapat menerapkan teknologi *IoT* pada rumah kita, yang kemudian dapat dikenal sebagai sistem rumah cerdas atau *smart home system*. Sistem rumah cerdas adalah rumah yang telah dilengkapi dengan sistem yang dapat mengintegrasi dan mengontrol berbagai peralatan pada rumah dengan tujuan meningkatkan kemudahan dan kenyamanan penghuni rumah. Sensor gerak, sensor kontak, dan perangkat pengendali lainnya merupakan komponen utama dari setiap sistem rumah cerdas [1]. Dengan memanfaatkan teknologi saat ini yang membuat segalanya terhubung, memungkinkan kendali penuh atas rumah kita dari mana saja [2].

Sistem rumah cerdas pada penelitian ini menggunakan NodeMCU ESP32 sebagai modul mikrokontroller dengan komunikasi nirkabel melalui *platform Telegram Messenger*. NodeMCU ESP32 memiliki beberapa *power mode*, salah satu nya adalah *deep sleep mode* yang menyebabkan konsumsi daya menjadi lebih hemat dibandingkan saat *active mode*. Pada sistem ini, akan digunakan sensor LDR untuk mengetahui tingkat intensitas cahaya, serta sensor DHT11 untuk mengetahui suhu ruangan. Peralatan rumah yang akan dikendalikan dan dipantau adalah kipas dan dua buah lampu. Sistem ini akan bekerja dengan dua pilihan mode, yaitu mode otomatis dan mode manual. Pada mode otomatis, proses *control* dan *monitoring* akan dilakukan otomatis berdasarkan kondisi yang dibaca oleh sensor. Lalu pada mode manual, pengguna memiliki kendali penuh atas proses *control* dan *monitoring* sistem.

2 LANDASAN TEORI

2.1 *Internet of Things*

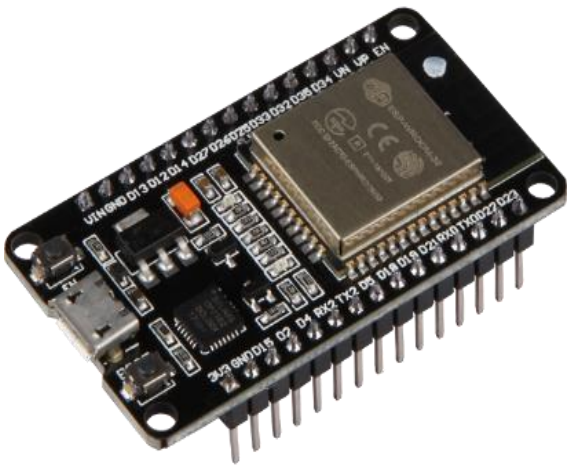
Internet of Things (IoT) adalah sebuah bidang menarik yang bertujuan untuk membuat semua perangkat yang terdapat di sekitar kita terhubung ke internet dan dapat berinteraksi dengan kita, maupun dengan yang lainnya [3], menyatakan bahwa dengan terhubungnya berbagai perangkat dengan internet, maka proses distribusi data dapat dilakukan secara terus-menerus, serta proses kontrol dapat dilakukan dari manapun.

Pada dasarnya, prinsip kerja IoT yaitu melakukan interaksi antar perangkat dalam proses pengiriman dan penerimaan data yang telah diolah secara otomatis tanpa bantuan manusia. Sistem IoT umumnya terdiri atas sensor-sensor, *embedded system*, komunikasi nirkabel, penyimpanan data via internet, sistem perangkat lunak, dan tentunya perangkat *end user* (mobile). Input dapat berupa sensor-sensor yang relevan dengan kebutuhan sistem. Dengan menggunakan sensor sebagai input, maka dapat mencegah terjadinya *human error*. Dalam sistem IoT, sensor-sensor tersebut mengirimkan data ke server sentral *(cloud)* via protokol jaringan komunikasi *wireless* seperti *Bluetooth, Zigbee*, dan *Wi-Fi*. Lalu server sentral *(cloud)* mengolah data dan melakukan analisis, kemudian mengirimkan hasil analisis tersebut kepada *end user* [4]. Berikut dijelaskan skema IoT pada Gambar 1.

Gambar 1 *Skema Kerja Internet of Things*

2.2 NodeMCU ESP32

ESP32 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*. Berbeda dengan arduino, yang membutuhkan catu daya sebesar 5volt pada atmega328nya, ESP32 hanya membutuhkan catu daya 3,3 Volt sehingga, jika digunakan catu daya 5 Volt untuk mengaktifkan ESP32, maka akan berpotensi merusak perangkat ESP32 [5]. Berikut adalah fisik dari ESP32 pada Gambar 2.



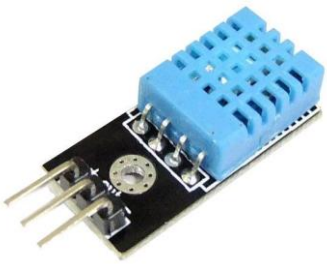
Gambar 2 *NodeMCU ESP32*

2.3 Sensor LDR (*Light Dependent Resistor)*

Gambar 3 *Sensor LDR*

Photoresistor atau yang juga dikenal sebagai LDR (*Light Dependent Resistor*) merupakan komponen resistor variabel yang nilai hambatannya berubah karena faktor intensitas cahaya. Jika intensitas cahaya yang mengenai permukaan sensor cukup terang, nilai hambatannya menjadi rendah, ±1K ohm dan apabila LDR diletakkan pada tempat yang gelap, maka nilai hambatannya menjadi tinggi hingga mencapai 10M ohm [6]. Bentuk fisik dari sensor LDR dapat dilihat pada Gambar 3.

2.4 Sensor DHT-11

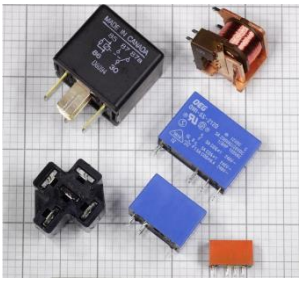


Gambar 4 *Sensor DHT-11*

Sensor DHT-11 yang ditunjukkan pada Gambar 4, merupakan sensor yang dapat mengukur temperatur dan juga kelembaban. Keluaran sensor DHT11 berupa sinyal digital yang sudah terkalibrasi. Jangkauan pengukuran temperatur dari sensor ini adalah 0-50°C dan jangkauan pengukuran kelembaban relatif sebesar 20-90%. Sensor DHT-11 membutuhkan catu daya sebesar 3 sampai 5,5volt DC. Keakuratan untuk kelembaban relatifnya sebesar ±4% dan keakuratan untuk temperatur sebesar ±2°C [7].

2.5 Relay

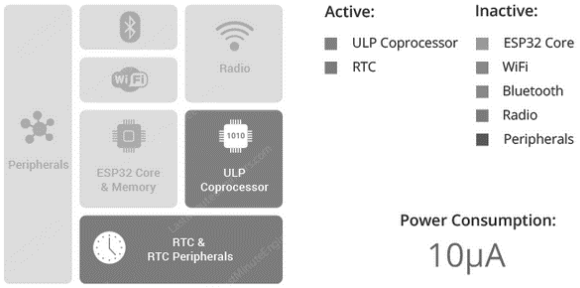
Sebuah relay memungkinkan sebuah sinyal atau gelombang listrik untuk diaktifkan atau dimatikan oleh aliran listrik yang berbeda. Seringkali, sebuah relay menggunakan tegangan rendah atau arus rendah untuk mengontrol tegangan tinggi dan atau arus tinggi. Tegangan atau arus listrik rendah dapat digerakkan oleh saklar yang relatif kecil, ekonomis, dan dapat dibawa ke relai dengan kabel kecil, yang dimana pada titik tersebut relai mengontrol arus yang lebih besar di dekat beban [8]. Bentuk fisik dari *relay* dapat dilihat pada Gambar 5.



Gambar 5 *Relay*

2.6 *Deep-sleep Mode*

Pada keadaan *active mode*, ESP32 membutuhkan sekitar 80mA pada operasi normal dan dapat mencapai 260mA saat melakukan transimisi data melalui WiFi. Ketika sistem yang dirancang menggunakan sumber daya langsung, maka menjadi masalah untuk kebutuhan daya tersebut. Akan tetapi, jika sistem yang dirancang menggunakan sumber daya dari baterai, setiap mA yang keluar tentu harus diperhitungkan. Oleh karena itu, solusi untuk mengurangi konsumsi daya dari ESP32 adalah dengan mengaktifkan *Deep Sleep Mode*.



Gambar 6 *Konsumsi Daya ESP32 saat Deep-sleep Mode*

Berdasarkan Gambar 6, saat *deep sleep mode* aktif maka CPU, sebagian besar RAM, dan semua *peripheral* digital akan dimatikan. Bagian yang tetap aktif dari *chip* adalah RTC *controller*, RTC *peripherals*, dan RTC *memory*. Konsumsi daya yang dibutuhkan pada mode ini sekitar 0.15mA sampai 10uA.

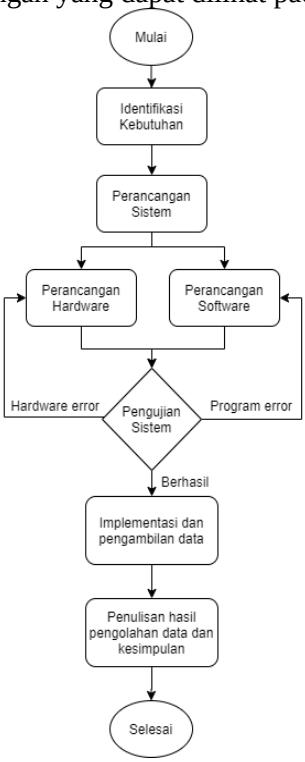
Selama mode ini, CPU utama dimatikan, sementara ULP co-processor akan membangunkan sistem utama, berdasarkan pengukuran sensor yang dilakukan. Data yang terdapat pada memori utama dari *chip* akan terhapus, sehingga tidak dapat diteruskan pada saat sistem diaktifkan kembali.

Untuk mengatasi permasalahan tersebut, maka RTC *memory* tetap aktif dalam mode ini. Jadi selama *mode deep sleep* aktif, data yang tersimpan pada RTC *memory* tidak akan hilang dan dapat dipulihkan kembali pada saat sistem utama diaktifkan.

3 METODE PERANCANGAN

3.1 Alur Perancangan

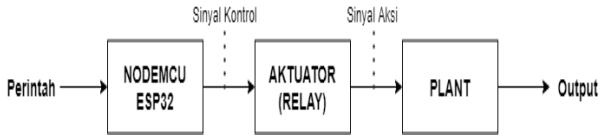
Dalam perancangan sistem rumah cerdas pada penelitian ini terdapat beberapa proses dan langkah-langkah kerja yang dilakukan. Berikut skema alur perancangan yang dapat dilihat pada Gambar 7.



***Gambar 7** Alur Perancangan Sistem*

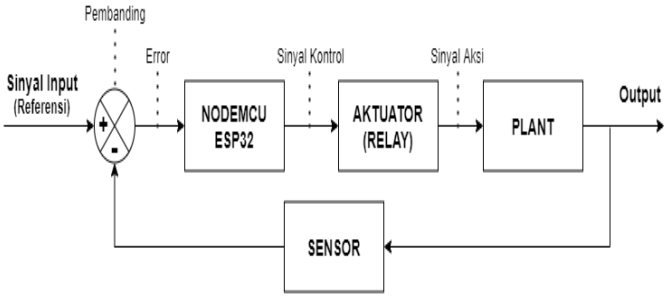
3.2 Blok Diagram Sistem

Sistem rumah cerdas pada penelitian ini memiliki dua mode yaitu mode manual dan mode otomatis. Pada mode manual, *input* berasal dari perintah yang dikirimkan pengguna melalui aplikasi *Telegram Messenger*, perintah tersebut diproses oleh mikrokontroller NodeMCU ESP32 lalu menghasilkan sinyal kontrol yang kemudian dikirim ke relay untuk mengendalikan perangkat elektronik yang dimaksud. Pada mode manual menggunakan skema *open-loop control system*, dimana *output* yang dihasilkan tidak memiliki pengaruh dan tidak dapat dijadikan umpan balik pembanding terhadap *input*. Blok diagram sistem rumah cerdas pada mode manual dapat dilihat pada Gambar 8.



***Gambar 8** Blok Diagram Mode Manual (Open-loop system)*

Pada mode otomatis, *input* dapat berasal dari intensitas cahaya atau suhu ruangan dengan nilai tertentu sebagai *set point*. Nilai dari *input* akan diproses oleh NodeMCU ESP32 lalu menghasilkan sinyal kontrol yang kemudian dikirim ke relay untuk mengendalikan perangkat elektronik seperti kipas atau lampu. Pada mode otomatis menggunakan skema *close-loop control system*, dimana *output* yang dihasilkan akan digunakan sebagai pembanding atau selisih terhadap nilai *set point*. Blok diagram sistem rumah cerdas pada mode otomatis dapat dilihat pada Gambar 9.

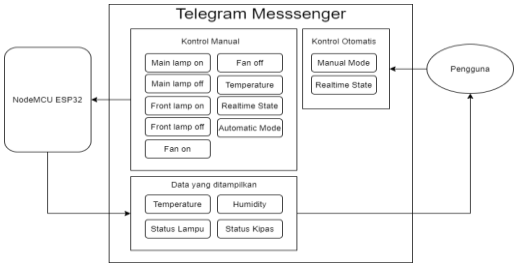


***Gambar 9** Blok Diagram Mode Otomatis (Close-loop system)*

3.3 Blok Diagram Software

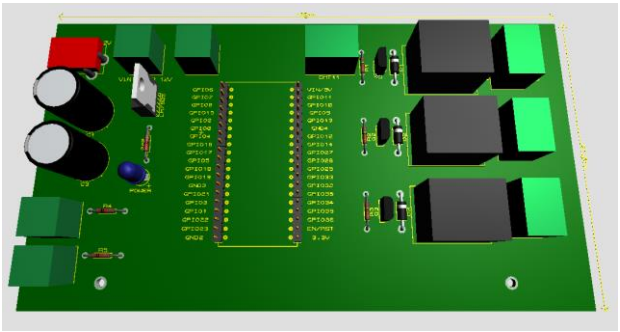
Dalam merancang program tentu harus diawali dengan perencanaan tentang bagaiamana program akan bekerja secara baik sebagai penghubung antara pengguna dan perangkat keras sistem. Berdasarkan blok diagram pada Gambar 10, sistem rumah cerdas yang akan dibangun pada penelitian ini memiliki mode manual dan mode otomatis.

Pada mode manual, pengguna dapat memberikan perintah kepada NodeMCU ESP32 sebagai pusat kendali sistem melalui tombol-tombol virtual yang akan muncul pada aplikasi Telegram Messenger. Setelah proses kendali selesai, maka NodeMCU ESP32 akan memberikan *feedback* kepada pengguna berupa data atau status yang dikirimkan ke Telegram Messenger. Sedangkan pada mode otomatis, sistem akan bekerja otomatis untuk mengendalikan lampu dan kipas berdasarkan kondisi yang dibaca oleh sensor-sensor.



Gambar 10 *Blok Diagram Software*

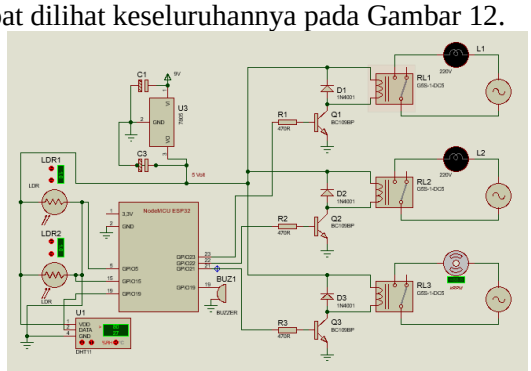
3.4 Perancangan Hardware



Gambar 11 *Visualisasi 3D Perancangan Perangkat Keras*

Perangkat keras yang digunakan pada penelitian ini terdiri atas beberapa komponen yang dirancang serta dihubungkan satu sama lain sehingga menjadi sebuah rangkaian elektronika yang utuh dari sistem rumah cerdas. Berdasarkan visualisasi 3D pada Gambar 11, terdapat beberapa bagian dari perancangan hardware ini, diantaranya rangkaian *power supply*, rangkaian input, dan rangkaian output.

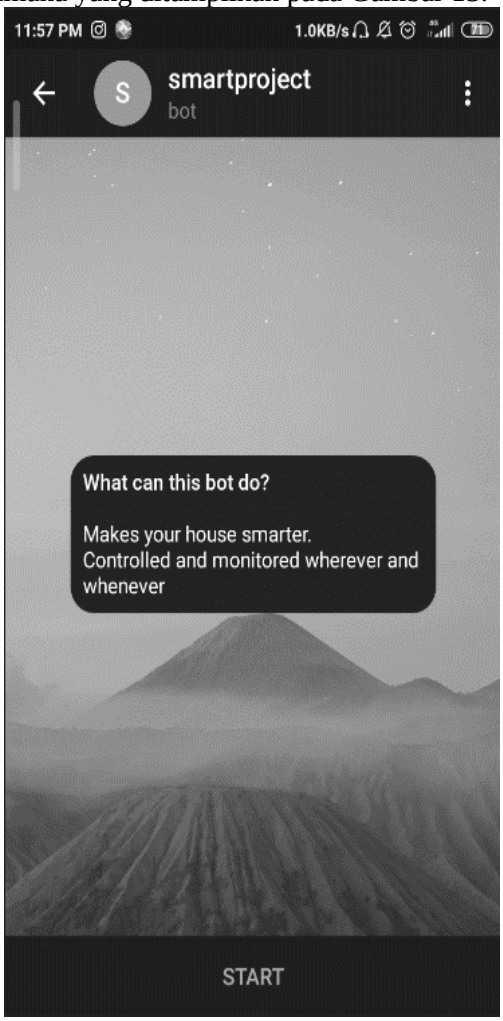
NodeMCU ESP32 berperan sebagai pengumpul data dari sensor dan kemudian mengolah data tersebut untuk memberikan *trigger* kepada relay. Untuk dapat beroperasi NodeMCU ESP32 membutuhkan tegangan sebesar 5 Volt yang didapatkan dari tegangan *output* IC regulator LM7805. Semua sensor dan *relay* yang masing-masing berperan sebagai *input* dan *output*, dihubungkan ke pin GPIO pada NodeMCU ESP32. Pada lampu dan kipas masing-masing terhubung ke sumber tegangan AC. Rangkaian dan pengkabelan dapat dilihat keseluruhannya pada Gambar 12.



Gambar 12 *Rangkaian dan Pengkabelan Hardware*

3.5 Perancangan *User Interface*

Pada sistem rumah cerdas ini menggunakan aplikasi Telegram Messenger yang merupakan aplikasi pengolah pesan. Namun, sebelum aplikasi tersebut dapat digunakan sebagai media komuikasi antara pengguna dan sistem, maka perlu dirancang sebuah *user interface* didalamnya, sebagaimana yang ditampilkan pada Gambar 13.

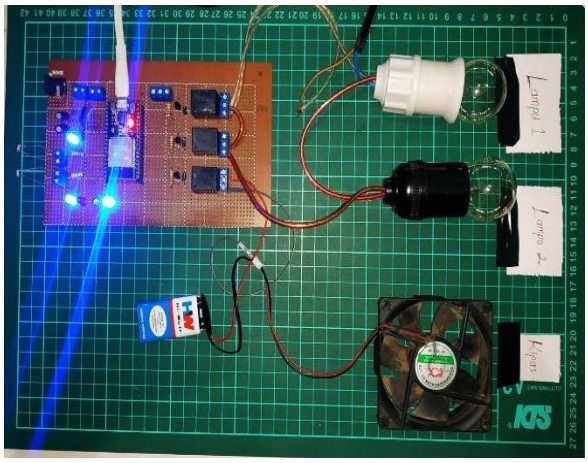


Gambar 13 *Tampilan Antarmuka Telegram Messenger*

4 HASIL DAN PEMBAHASAN

4.1 Hasil Perancangan Hardware

Pada Gambar 14 merupakan hasil dari perancangan perangkat keras untuk sistem rumah cerdas. Seluruh komponen pada perangkat keras ini disusun di atas sebuah PCB bolong dengan panjang 16 cm dan lebar 9 cm. Perangkat keras ini terdiri atas beberapa bagian, yaitu bagian *input*, proses, dan *output*. Pada bagian *input* terdiri dari rangkaian *power supply*, sensor LDR, dan DHT11. Pada bagian proses menggunakan mikrokontroller NodeMCU ESP32. Pada bagian *output* terdiri dari rangkaian lampu indikator, serta rangkaian *relay* untuk mengendalikan lampu dan kipas.

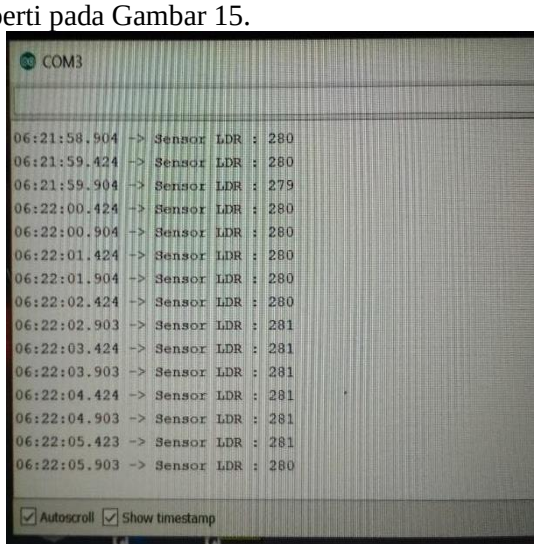


Gambar 14 *Hasil Perancangan Hardware*

4.2 Pengujian Sensor LDR

Pada penelitian ini menggunakan 2 buah sensor LDR untuk mengetahui intensitas cahaya pada lingkungan sekitar. Masing-masing sensor dihubungkan ke kanal *input* analog 12bit dari NodeMCU ESP32, sehingga rentang data yang dihasilkan dari sensor LDR yaitu 0 hingga 4095, yang kemudian dikonversikan ke rentang 0 hingga 500 untuk memudahkan pembacaan data dan kalibrasi sensor.

Sebelum pengujian dari masing-masing sensor, dilakukan kalibrasi terhadap lingkungan sekitar seperti pada Gambar 15.



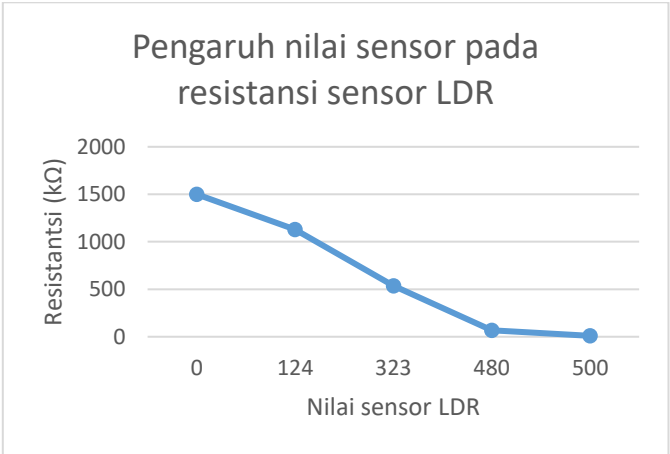
Gambar 15 *Nilai Sensor LDR saat Pagi Hari*

Setelah dilakukan kalibrasi didapatkan nilai *threshold* pada beberapa kondisi, yaitu 0 hingga 299 untuk kondisi gelap dan 300 hingga 500 untuk kondisi terang. Penentuan nilai ini disesuaikan dengan kebutuhan *user*. Nilai *threshold* dan hasil kalibrasi sensor masing-masing dapat dilihat pada Tabel 1.

Tabel 1 *Hasil pengukuran sensor LDR pada waktu tertentu*

Pengujian	Waktu	Kondisi	Nilai Pada Sensor
1	05.00	Gelap	0
2	06.21	Gelap	280
3	09.25	Terang	465
4	12.50	Terang	500
5	16.31	Terang	436
6	19.09	Gelap	0

Sensor LDR pada dasarnya merupakan komponen resistor variabel yang nilai resistansinya dapat berubah-ubah karena faktor intensitas cahaya. Pada Gambar 16 diperlihatkan grafik dari pengaruh nilai sensor LDR yang didapatkan berdasarkan intensitas cahaya di lingkungan sekitar terhadap nilai resistansi sensor LDR. Jika intensitas cahaya yang diterima sensor LDR cukup tinggi, maka nilai resistansi dari sensor LDR menjadi rendah sekitar 10k ohm, sedangkan jika intensitas cahaya yang diterima sensor LDR rendah, maka nilai resistansi dari sensor LDR menjadi tinggi hingga mencapai 1.5M ohm.



Gambar 16 *Pengaruh Intensitas Cahaya terhadap Resistansi Sensor LDR*

4.3 Pengujian Sensor DHT-11

Pada penelitian ini digunakan 1 buah sensor DHT11 untuk mengetahui temperatur dan juga kelembaban dari lingkungan sekitar. Hasil dari pembacaaan kondisi oleh sensor DHT11 digunakan untuk menghidupkan atau mematikan kipas.

Proses pertama dari pengujian sensor DHT11 adalah mencari nilai *threshold* untuk mengetahui ambang batas suhu. Nilai *threshold* yang didapatkan yaitu pada suhu 25°C [9]. Suhu yang terbaca pada sensor DHT11 dan pengaruhnya dalam proses pengendalian kipas dapat dilihat pada Tabel 2.

Tabel 2 *Pengaruh suhu pada sensor DHT-11 terhadap kontrol kipas angin*

Pengujian	Suhu Pada Sensor (dalam °C)	Kondisi Ruangan	Kondisi Kipas
1	23.20	Normal	Mati
2	24.40	Normal	Mati
3	25.80	Panas	Hidup
4	24.20	Normal	Mati
5	26.30	Panas	Hidup

4.4 Pengujian Kontrol pada Jarak Tertentu

Pada penelitian ini digunakan aplikasi Telegram Messenger sebagai perantara komunikasi antara *user* dan perangkat keras. Proses kontrol sistem rumah cerdas menggunakan Telegram Messenger dapat dilakukan baik dari jarak dekat maupun jarak jauh. Pengujian kontrol sistem rumah cerdas dilakukan pada jarak 1 km hingga 5km di berbagai lokasi yang dijelaskan pada Tabel 3.

Tabel 3 *Lokasi pengujian kontrol*

Lokasi Rumah	Lokasi Pengujian	Jarak (dalam km)
Komplek Hankam Mabes TNI, Jatimakmur, Pondok Gede, Bekasi	Kantor PLN Jatimakmur	1
	Perumahan Kologad	2
	Plaza Pondok Gede	3
	Kantor Kecamatan Pondok Gede	4
	Jalan Gamprit Jatiwaringin	5

Proses pengujian ini bertujuan untuk mencari *delay* waktu yang terjadi pada saat *user* mengirimkan perintah hingga perangkat keras merespon perintah tersebut.

Tabel 4 *Delay waktu pengujian pada mode manual*

Perintah	*Delay* Waktu (dalam detik)				
	1 km	2 km	3 km	4 km	5 km
Main Lamp On	0.68	0.59	0.91	2.22	1.78
Front Lamp On	1.58	0.82	1.09	1.35	0.97
Fan On	0.57	1.76	1.10	1.75	1.41
Temperature	1.69	0.54	1.25	1.80	0.90
Realtime State	0.60	0.50	0.68	1.19	2.19
Automatic Mode	0.50	0.50	1.66	2.50	0.90

Berdasarkan Tabel 4 di atas maka didapatkan *delay* waktu yang bervariasi pada masing-masing jarak pengujian. Rata-rata *delay* waktu pada tiap jarak dan secara keseluruhan pada mode manual dapat ditentukan dengan cara dibawah ini:

1. Rata-rata *delay* waktu pada jarak 1 km

$$Rata1 = \frac{waktu\ pada\ setiap\ perintah}{banyaknya\ perintah}$$

$$Rata1 = \frac{0.68 + 1.58 + 0.57 + 1.69 + 0.60 + 0.50}{6} \quad (1)$$

$$Rata1 = 0.93\ detik$$

2. Rata-rata *delay* waktu pada jarak 2 km

$$Rata2 = \frac{waktu\ pada\ setiap\ perintah}{banyaknya\ perintah}$$

$$Rata2 = \frac{0.59 + 0.82 + 1.76 + 0.54 + 0.50 + 0.50}{6}$$

$$Rata2 = 0.785\ detik$$

3. Rata-rata *delay* waktu pada jarak 3 km

$$Rata3 = \frac{waktu\ pada\ setiap\ perintah}{banyaknya\ perintah}$$

$$Rata3 = \frac{0.91 + 1.09 + 1.10 + 1.25 + 0.68 + 1.66}{6}$$

$$Rata3 = 1.11\ detik$$

4. Rata-rata *delay* waktu pada jarak 4 km

$$Rata4 = \frac{waktu\ pada\ setiap\ perintah}{banyaknya\ perintah}$$

$$Rata4 = \frac{2.22 + 1.35 + 1.75 + 1.80 + 1.19 + 2.50}{6}$$

$$Rata4 = 1.80\ detik$$

5. Rata-rata *delay* waktu pada jarak 5 km

$$Rata5 = \frac{waktu\ pada\ setiap\ perintah}{banyaknya\ perintah}$$

$$Rata5 = \frac{1.78 + 0.97 + 1.41 + 0.90 + 2.19 + 0.90}{6}$$

$$Rata5 = 1.35\ detik$$

6. Rata-rata *delay* waktu pada mode manual

$$Rata\ manual = \frac{Rata\ 1 + Rata\ 2 + Rata\ 3 + Rata\ 4 + Rata\ 5}{banyaknya\ percobaan} \quad (2)$$

$$Rata\ manual = 1.195\ detik$$

Tabel 5 *Delay waktu pengujian pada mode otomatis*

Perintah	*Delay* Waktu (dalam detik)				
	1 km	2 km	3 km	4 km	5 km
Realtime State	1.87	2.58	1.43	1.68	1.00
Manual Mode	2.67	1.48	1.15	1.67	1.34

ISSN No. 2502-8782

Berdasarkan Tabel 5 didapatkan *delay* waktu yang bervariasi pada masing-masing jarak pengujian. Rata-rata *delay* waktu pada tiap jarak dan secara keseluruhan pada mode otomatis dapat ditentukan dengan cara di bawah ini :

1. Rata-rata *delay* waktu pada jarak 1 km

$$Rata1 = \frac{waktu\ pada\ setiap\ perintah}{banyaknya\ perintah}$$
$$Rata\ 1 = \frac{1.87 + 2.67}{2}$$
$$Rata\ 1 = 2.27\ detik$$

2. Rata-rata *delay* waktu pada jarak 2 km

$$Rata\ 2 = \frac{waktu\ pada\ setiap\ perintah}{banyaknya\ perintah}$$
$$Rata\ 2 = \frac{2.58 + 1.48}{2}$$
$$Rata\ 2 = 2.03\ detik$$

3. Rata-rata *delay* waktu pada jarak 3 km

$$Rata\ 3 = \frac{waktu\ pada\ setiap\ perintah}{banyaknya\ perintah}$$
$$Rata\ 3 = \frac{1.43 + 1.15}{2}$$
$$Rata\ 3 = 1.29\ detik$$

4. Rata-rata *delay* waktu pada jarak 4 km

$$Rata\ 4 = \frac{waktu\ pada\ setiap\ perintah}{banyaknya\ perintah}$$
$$Rata\ 4 = \frac{1.68 + 1.67}{2}$$
$$Rata\ 4 = 1.675\ detik$$

5. Rata-rata *delay* waktu pada jarak 5 km

$$Rata\ 5 = \frac{waktu\ pada\ setiap\ perintah}{banyaknya\ perintah}$$
$$Rata\ 5 = \frac{1.00 + 1.34}{2}$$
$$Rata\ 5 = 1.17\ detik$$

6. Rata-rata *delay* waktu pada mode otomatis

$$Rata\ otomatis = \frac{Rata\ 1 + Rata\ 2 + Rata\ 3 + Rata\ 4 + Rata\ 5}{banyaknya\ percobaan}$$
$$Rata\ otomatis = \frac{2.27 + 2.03 + 1.29 + 1.675 + 1.17}{5}$$
$$Rata\ otomatis = 1.687\ detik$$

4.5 Pengujian Daya pada Active Mode dan Deep-sleep Mode

Pada sistem rumah cerdas ini, *active mode* digunakan saat sistem aktif dimana segala proses *input* dan *output* dapat dilakukan oleh perangkat keras dan lunak, serta *user* dapat memberikan perintah maupun memantau dari aplikasi *Telegram Messenger*. Daya listrik yang dikonsumsi pada saat *active mode* bervariasi tergantung pada proses yang sedang dilakukan. Pada Tabel 6 diperlihatkan hasil pengujian seberapa besar arus listrik yang mengalir ke NodeMCU ESP32 pada saat *active mode.*

Tabel 6 *Konsumsi arus listrik pada saat active mode*

Keadaan	Arus Listrik (dalam mA)
WiFi Off	140
WiFi On	80
Satu Perintah	140
Dua Perintah	200
Tiga Perintah	260

Hasil pengukuran arus listrik pada saat *active mode* menunjukkan bahwa naik turunnya arus listrik bergantung pada keadaan atau status pada ESP32. Konsumsi daya listrik pada saat *active mode* dapat diketahui dengan cara di bawah ini:

- Rata-rata arus listrik *active mode*

$$Rata - rata = \frac{Jumlah\ arus\ dari\ setiap\ keadaan}{banyaknya\ keadaan} \quad (3)$$

$$Rata - rata = \frac{(140 + 80 + 140 + 200 + 260)\ mA}{5}$$
$$Rata - rata = 164 \text{ mA atau } 0.164 \text{ A}$$

- Tegangan (V) = 4.93 Volt
- Daya (P) saat *active mode* = V x I = 4.93volt x 0.164 A = 0.808 Watt

Selain *active mode*, pada penelitian ini juga menggunakan *deep sleep mode*. *Deep sleep mode* merupakan keadaan dimana hampir semua bagian pada arsitektur ESP32 dinonaktifkan, kecuali RTC *peripherals* dan RTC *memory*. Mode ini digunakan pada saat pengguna sedang tidak membutuhkan proses kontrol akan sistem rumah cerdas, sehingga diharapkan dapat menghemat konsumsi daya listrik. Pada Tabel 7 diperlihatkan hasil pengujian seberapa besar arus listrik yang mengalir ke NodeMCU ESP32 pada saat *deep sleep mode*.

Tabel 7 *Konsumsi listrik pada saat deep sleep mode*

Status	Keadaan	Waktu (dalam menit)	Arus Listrik (dalam mA)
Wifi	Tidak Terkoneksi	45	3.5
		90	3.5
		120	3.5
		150	3.5

Hasil pengukuran arus listrik pada saat *deep sleep mode* menunjukkan angka 3,5mA pada setiap waktu percobaan, sehingga konsumsi daya listrik pada saat *deep sleep mode* dapat diketahui dengan cara di bawah ini:

- Tegangan (V) = 4.93 Volt
- Daya (P) saat *active mode* = V x I = 4.93volt x 0.0035 A = 0.0172 Watt

5 SIMPULAN

Berdasarkan hasil penelitian yang telah didapatkan, maka kesimpulan yang dapat dibuat adalah sebagai berikut:

1. Komunikasi antara pengguna dan perangkat keras dari sistem rumah cerdas menggunakan Telegram Messenger berlangsung dengan lancar hingga jarak 5kilometer dengan tingkat keberhasilan 100%. Pada saat pengguna memberikan perintah maka perintah tersebut akan dieksekusi oleh NodeMCU ESP32 dan setelah berhasil maka pengguna akan menerima pesan *feedback* dari sistem.
2. Berdasarkan pengujian dan perhitungan *delay* waktu pengujian kontrol pada Tabel 4 dan Tabel 5, saat mode manual didapatkan rata – rata 1.195 detik dan saat mode otomatis didapatkan rata – rata 1.687 detik. Rata – rata tersebut merupakan *delay* waktu dari setiap perintah yang pengguna kirimkan baik dari mode manual maupun otomatis. Hasil tersebut telah memenuhi tujuan yaitu *delay* waktu kontrol sistem rumah cerdas ≤ 4 detik.
3. Berdasarkan pengujian dan perhitungan konsumsi daya listrik pada Tabel 6 dan Tabel 7 saat *active mode* dimana seluruh bagian di dalam arsitektur ESP32 aktif didapatkan rata – rata daya sebesar 0.808 Watt, sedangkan saat *deep sleep mode* didapatkan rata-rata konsumsi daya sebesar 0.0172Watt. Hasil tersebut telah memenuhi tujuan yaitu konsumsi daya system rumah cerdas ≤ 1.28Watt.

KEPUSTAKAAN

[1] M. Schwartz, *Arduino Home Automation Projects*. 2014.

[2] C. Batrinu, *ESP8266 home automation projects*. 2017.

[3] M. Schwartz, *Internet of Things with*. Packt Publishing Ltd., 2016.

[4] J. S. Wilkinson and I. D. Pletcher, *Internet of Things (IoT) Technologies, Applications, Challenges, and Solutions*. CRC Press, 2019.

[5] N. Kolban, “Kolban’s Books on ESP8266 and ESP32,” 2016, pp. v–vi.

[6] M. Syahwil, *Panduan Mudah Belajar Arduino Menggunakan So,*. Yogyakarta: CV ANDI OFFSET, 2017.

[7] A. Najmurrokhman, A, Kusnandar, “Prototipe Pengendali Suhu Dan Kelembaban Untuk Cold Storage Menggunakan Mikrokontroler Atmega328 Dan Sensor Dht11,” *J. Teknol. Univ. Muhammadiyah Jakarta*, vol. 10, no. 1, pp. 73–82, 2018.

[8] C. Platt, *Encyclopedia of Electronic Components Volume 1*, 1st ed. Make, 2012.

[9] M. H. Kotta, “Neutral Temperature and Indonesian Human Comfortable Temperature Range (Case Study Research in Office Buildings in Makassar),” *Metropilar - J. Ilm. Fak. Tek.*, vol. 6, no. 1, pp. 23–29, 2008, [Online]. Available: http://ojs.uho.ac.id/index.php/metropilar/article/view/447/287.