

Penerapan Aljabar Boolean pada *Smart Home Devices*

Erdianti Wiga Putri Andini - 13522053¹

Program Studi Teknik Informatika

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia

¹13522053@std.stei.itb.ac.id

Abstract— *The smart home sector has become a crucial area in home automation, extending its influence to smart cities, manufacturing, and societies. Beyond the confines of human living spaces, Smart Home Systems (SHS) have garnered attention, with conventional systems hampered by wired controls, high installation costs, and limited scalability. Wireless sensor network-based SHS presents a promising solution to these challenges. This study delves into the application of Boolean algebra in shaping the logic governing various operational facets of smart home devices. Boolean algebra enables the formulation of logic rules dictating device responses to diverse home conditions, fostering automation, energy optimization, heightened security, device interconnectivity, scheduled operations, and adaptability to user needs or environmental changes. The study underscores the transformative potential of Boolean algebra in programming smart home devices for tailored responses. This programmability extends to automatic actions, energy-efficient practices, security enhancements, and dynamic adjustments based on evolving conditions. The research's implications loom large over the landscape of smart home technology, offering insights into how Boolean algebra serves as a potent tool for system design and management. Its impact resonates across domains such as energy efficiency, security fortification, and adaptability, thereby contributing significantly to the broader discourse on smart home technology and its potential to enhance users' quality of life.*

Keywords—*Smart Home Devices, Aljabar Boolean, Sensor, Automation.*

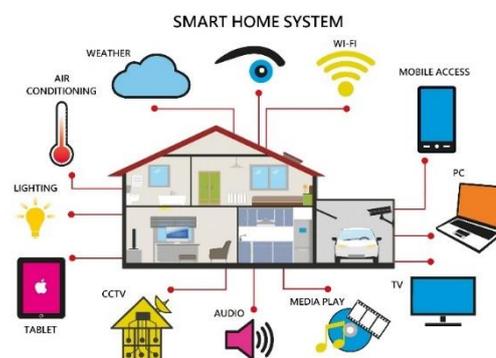
I. PENDAHULUAN

Sektor *smart home* telah menjadi salah satu bidang yang sangat menarik dalam domain otomatisasi dan manajemen rumah. Penggunaan istilah *smart home* tidak terbatas pada tempat tinggal manusia. Sebaliknya, istilah ini mencakup berbagai implikasi teknologis yang lebih luas, termasuk konsep hunian atau kehidupan yang cerdas [1]. Integrasi teknologi yang dikendalikan oleh komputer, seperti *smart cities*, *smart manufacturing*, dan *smart societies*, memperluas konsep *smart home* di luar lingkup tempat tinggal manusia. Dengan fokus utama pada kesejahteraan umat manusia, *smart technology* menjadi penggerak utama bagi ide-ide inovatif seperti sistem rumah pintar (Smart Home System - SHS). Berkembangnya *smart products and services* telah menyebabkan meningkatnya keterhubungan perangkat dan berbagi informasi, yang memengaruhi perkembangan cepat teknologi rumah pintar secara global [2].

Smart home merupakan pemanfaatan teknologi komputer, teknologi kontrol, teknologi tampilan gambar, dan teknologi komunikasi yang saling terhubung melalui jaringan fasilitas-

fasilitas beragam untuk memenuhi kebutuhan otomatisasi seluruh sistem guna memberikan kontrol dan manajemen yang lebih praktis [3]. Implementasi *smart home* tradisional umumnya mengontrol fasilitas menggunakan kabel, yang sulit untuk dilepaskan dari berbagai kendala kabel, biaya instalasinya tinggi, dan skalabilitas sistem yang kurang baik. Sistem *smart home* berbasis teknologi jaringan *wireless* sensor tidak hanya dapat menghilangkan keterbatasan kabel dan mengurangi biaya instalasi, tetapi juga secara signifikan meningkatkan skalabilitas sistem.

Sebenarnya pemanfaatan *smart home* ini dapat melalui berbagai cara seperti, *logic rules*, *artificial neural networks*, *fuzzy neural network*, *support vector machines*, *Bayesian networks*, dll [4]. Namun, cara-cara tersebut terlalu rumit akibat kurangnya lingkungan yang memadai. Terdapat alternatif cara yaitu menggunakan aljabar Boolean. Penggunaan aljabar Boolean dalam konteks ini membantu dalam merancang, mengoptimalkan, dan mengendalikan logika yang mendasari berbagai aspek operasional *smart home*. Aljabar Boolean digunakan untuk mendefinisikan dan mengimplementasikan logika pemrograman pada *smart home devices*. Hal ini melibatkan pengembangan aturan-aturan logika yang mengatur bagaimana perangkat bereaksi terhadap berbagai situasi atau kondisi di rumah, seperti suhu, keamanan, pencahayaan, dan lainnya.



Gambar 1. Macam-macam Smart Home Devices

Sumber: <http://visioforce.com/images/smart-home-01.jpg>

Dengan menggunakan aljabar Boolean, *smart home devices* dapat diprogram untuk merespons kondisi tertentu dan melakukan tindakan otomatis. Contohnya, ketika suhu ruangan mencapai nilai tertentu, sistem pemanas atau pendingin udara dapat diaktifkan atau dinonaktifkan secara otomatis. Aljabar Boolean juga membantu dalam merancang logika untuk

mengoptimalkan penggunaan energi pada *smart home devices*. Pengguna dapat mengonfigurasi *smart home devices* untuk mati atau hidup secara otomatis berdasarkan ketersediaan energi, keberadaan penghuni, atau kondisi cuaca. Efisiensi energi dapat dikelola pula menggunakan aljabar Boolean dengan mematikan perangkat secara otomatis ketika tidak digunakan sehingga dapat menghemat penggunaan energi. Untuk alasan keamanan, aljabar Boolean dapat membuat logika keamanan yang canggih, teknologi ini dapat menjadi sensor yang dapat mendeteksi aktivitas mencurigakan di suatu ruangan dengan mengaktifkan alarm atau mengirim notifikasi kepada pemilik. *Smart home devices* sering kali terdiri dari beberapa perangkat yang saling terhubung. Aljabar Boolean memungkinkan pengembangan logika interkoneksi yang kompleks, memungkinkan perangkat berkomunikasi satu sama lain untuk meningkatkan efisiensi dan fungsionalitas. Aljabar Boolean juga memungkinkan pengaturan penjadwalan otomatis perangkat, seperti penerangan atau sistem irigasi. Pengguna dapat membuat aturan berdasarkan waktu tertentu atau kondisi tertentu untuk mengotomatiskan berbagai tindakan. Alasan terakhir penggunaan aljabar Boolean pada *smart home devices* adalah aljabar ini memungkinkan *smart home devices* untuk bersifat adaptif dan fleksibel. Logika dapat diubah atau diperbarui dengan mudah untuk menyesuaikan perubahan kebutuhan pengguna atau kondisi lingkungan.

II. LANDASAN TEORI

A. Definisi Aljabar Boolean

Aljabar Boolean, yang ditemukan oleh matematikawan terkemuka George Boole pada tahun 1854 [6], membentuk dasar matematika dari desain logika. Pada tahun 1937, Claude Shannon merinci desain logika dari aljabar Boolean dalam makalah berjudul "A symbolic Analysis of Relay and Switching Circuits," yang meraih penghargaan (Balabanian dan Carlson, 2001; Camara, 2010) [5]. Aljabar Boolean merupakan suatu metode yang digunakan untuk menyederhanakan persamaan dan fungsi Boolean sebelum diterapkan pada perancangan gerbang logika. Dasar logika untuk operasi matematika pada angka biner '0' yang menandakan *false* dan '1' yang menandakan *true*. Fungsi Boolean dan operasinya diilustrasikan melalui diagram yang dikenal sebagai gerbang logika, sebuah konsep yang terungkap dalam penelitian oleh Muller pada tahun 1954 [5].

Aljabar Boolean terutama digunakan untuk menyederhanakan dan menganalisis ekspresi Boolean yang kompleks. Aljabar Boolean terdiri dari tiga operator dasar yaitu AND, OR, dan NOT. Operasi dalam aljabar Boolean direpresentasikan oleh "." untuk AND dan "+" untuk OR. Operasi dapat dilakukan pada variabel yang direpresentasikan menggunakan huruf kapital seperti 'A', 'B', dan sebagainya [7].

Misalkan B adalah himpunan yang didefinisikan pada dua operator biner, $+$ dan \cdot , dan sebuah operator uner, $'$. Misalkan 0 dan 1 adalah dua elemen yang berbeda dari B . Maka, tupel

$$\langle B, +, \cdot, ', 0, 1 \rangle$$

disebut aljabar Boolean jika untuk setiap $a, b, c \in B$ berlaku aksioma berikut [6].

1. Identitas

Aksioma yang menyatakan bahwa hasil operasi pada elemen akan menghasilkan elemen itu sendiri.

$$(i) a + 0 = a$$

$$(ii) a \cdot 1 = a$$

2. Komutatif

Aksioma yang menyatakan bahwa urutan operan ($+$ atau \cdot) tidak mempengaruhi hasil dari operasi alias hasil tetap sama.

$$(i) a + b = b + a$$

$$(ii) a \cdot b = b \cdot a$$

3. Distributif

Aksioma yang menggambarkan hubungan antara operasi AND dan OR terhadap satu sama lain.

$$(i) a \cdot (b + c) = (a \cdot b) + (a \cdot c)$$

$$(ii) a + (b \cdot c) = (a + b) \cdot (a + c)$$

4. Komplemen

Aksioma yang menyatakan bila suatu elemen dioperasikan dengan elemennya sendiri, akan menghasilkan elemen identitas. Untuk setiap $a \in B$ terdapat elemen unik $a' \in B$, sehingga

$$(i) a + a' = 1$$

$$(ii) a \cdot a' = 0$$

B. Operasi Aljabar Boolean Dua Nilai

Eksprei Boolean terbentuk dari elemen B dan/atau variabel yang dapat digabungkan menggunakan operator $+$, \cdot , dan $'$. Penggunaan ekspresi ini juga digunakan untuk membuktikan aksioma-aksioma yang telah dijelaskan pada subbab sebelumnya. Aljabar Boolean sering kali menggunakan model 2-nilai, di mana terdapat himpunan B yang terdiri dari dua elemen, yaitu 0 dan 1, bersama dengan operator biner yaitu $+$ dan \cdot , serta operator uner yaitu $'$. Pembuktian aksioma ini akan dilakukan pada aljabar Boolean dua nilai [6].

1. Kaidah untuk operator biner " \cdot "

Tabel 1. Kaidah Operator Biner \cdot

a	b	$a \cdot b$
0	0	0
0	1	0
1	0	0
1	1	1

Sumber:

[https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/11-Aljabar-Boolean-\(2023\)-bagian1.pdf](https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/11-Aljabar-Boolean-(2023)-bagian1.pdf)

2. Kaidah untuk operator biner “ + ”

Tabel 2. Kaidah Operator Biner +

a	b	a + b
0	0	0
0	1	1
1	0	1
1	1	1

Sumber:

[https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/11-Aljabar-Boolean-\(2023\)-bagian1.pdf](https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/11-Aljabar-Boolean-(2023)-bagian1.pdf)

3. Kaidah untuk operator uner “ ’ ”

Tabel 3. Kaidah Operator Uner ’

a	a’
0	1
1	0

Sumber:

[https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/11-Aljabar-Boolean-\(2023\)-bagian1.pdf](https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/11-Aljabar-Boolean-(2023)-bagian1.pdf)

C. Hukum-hukum Aljabar Boolean

Berdasarkan operator-operator yang tersedia, ditemukan beberapa hukum yang berlaku pada aljabar Boolean yang dijabarkan di bawah ini [6].

- Hukum Identitas
 - $a + 0 = a$
 - $a \cdot 1 = a$
- Hukum Idempoten
 - $a + a = a$
 - $a \cdot a = a$
- Hukum Komplemen
 - $a + a' = 1$
 - $aa' = 0$
- Hukum Dominansi
 - $a \cdot 0 = 0$
 - $a + 1 = 1$
- Hukum Invulusi
 - $(a')' = a$
- Hukum Penyerapan
 - $a + ab = a$
 - $a(a + b) = a$
- Hukum Komutatif
 - $a + b = b + a$
 - $ab = ba$
- Hukum Asosiatif
 - $a + (b + c) = (a + b) + c$
 - $a \cdot (b \cdot c) = (a \cdot b) \cdot c$
- Hukum Distributif
 - $a + (b \cdot c) = (a + b) \cdot (a + c)$
 - $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$

10. Hukum 0 / 1

- $0' = 1$
- $1' = 0$

11. Hukum De Morgan

- $(a + b)' = a' \cdot b'$
- $(a \cdot b)' = a' + b'$

Truth table dari Hukum De Morgan yang pertama tertera sebagai berikut.

Tabel 4. Truth Table De Morgan (i)

A	B	A'	B'	(A + B)'	A.B'
0	0	1	1	1	1
0	1	1	0	0	0
1	0	0	1	0	0
1	1	0	0	0	0

Sumber: <https://testbook.com/maths/boolean-algebra>

Sedangkan truth table dari Hukum De Morgan yang kedua tertera sebagai berikut.

Tabel 5. Truth Table De Morgan (ii)

A	B	A'	B'	(A.B)'	A' + B'
0	0	1	1	1	1
0	1	1	0	1	1
1	0	0	1	1	1
1	1	0	0	0	0

Sumber: <https://testbook.com/maths/boolean-algebra>

D. Fungsi Boolean

Fungsi Boolean adalah suatu fungsi matematis yang menerima satu atau lebih input biner dan menghasilkan output biner berdasarkan aturan logika Boolean. Input dan output dari fungsi ini hanya dapat mengambil dua nilai, yaitu 0 atau 1, yang mewakili nilai kebenaran (false atau true). Fungsi Boolean umumnya digunakan dalam konteks logika digital, aljabar Boolean, dan perancangan rangkaian logika. Fungsi Boolean dapat diekspresikan melalui operasi-operasi dasar aljabar Boolean, seperti AND, OR, dan NOT. Beberapa contoh fungsi Boolean dasar melibatkan satu atau lebih variabel Boolean, misalnya fungsi dengan variable n dapat dinyatakan dalam $f(x_1, x_2, \dots, x_n)$ dengan masing-masing variabel bernilai 1 atau 0. Contoh dari fungsi Boolean antara lain:

- $f(x) = x$
- $f(x, y) = x' \cdot y'$
- $f(x, y, z) = x + y'z$
- $f(x, y, z) = xy(x + y + x'y'z')$
- $f(x, y) = (x + y)'$

Fungsi Boolean ini memiliki dua bentuk standar (bentuk kanonik). Pertama, sebagai penjumlahan dari hasil kali (*sum-of-product* atau SOP) kedua sebagai perkalian dari hasil jumlah (*product-of-sum* atau POS) [6]. Terdapat dua jenis suku pada fungsi Boolean yaitu minterm dan maxterm. Minterm adalah suku (term) di dalam ekspresi

boolean mengandung literal yang lengkap dalam bentuk hasil kali, sedangkan maxterm adalah suku (term) di dalam ekspresi boolean mengandung literal yang lengkap dalam bentuk hasil jumlah.

Bentuk fungsi SOP terdiri dari gabungan beberapa minterm, sedangkan fungsi POS terdiri dari beberapa maxterm. Contoh bentuk fungsi SOP adalah $f(x, y, z) = x'y'z + xy'z' + xyz$. Contoh fungsi POS adalah $g(x, y, z) = (x + y + z)(x + y' + z)(x + y' + z')$.

Minterm dan maxterm dapat dibentuk dari status tiap peubah. Untuk minterm, setiap peubah yang bernilai 0 dinyatakan dalam bentuk komplemen, sedangkan peubah yang bernilai 1 dinyatakan tanpa komplemen, berlaku sebaliknya untuk maxterm. Lebih jelasnya tertera pada tabel di bawah ini.

Tabel 6. Cara Membentuk Minterm dan Maxterm dengan Dua Peubah

x	y	Minterm		Maxterm	
		Suku	Lambang	Suku	Lambang
0	0	$x'y'$	m_0	$x + y$	M_0
0	1	$x'y$	m_1	$x + y'$	M_1
1	0	xy'	m_2	$x' + y$	M_2
1	1	xy	m_3	$x' + y'$	M_3

Sumber:

[https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/11-Aljabar-Boolean-\(2023\)-bagian1.pdf](https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/11-Aljabar-Boolean-(2023)-bagian1.pdf)

Tabel 7. Cara Membentuk Minterm dan Maxterm dengan Tiga Peubah

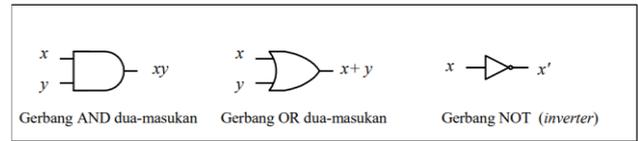
x	y	z	Minterm		Maxterm	
			Suku	Lambang	Suku	Lambang
0	0	0	$x'y'z'$	m_0	$x + y + z$	M_0
0	0	1	$x'y'z$	m_1	$x + y + z'$	M_1
0	1	0	$x'y z'$	m_2	$x + y' + z$	M_2
0	1	1	$x'y z$	m_3	$x + y' + z'$	M_3
1	0	0	$x y' z'$	m_4	$x' + y + z$	M_4
1	0	1	$x y' z$	m_5	$x' + y + z'$	M_5
1	1	0	$x y z'$	m_6	$x' + y' + z$	M_6
1	1	1	$x y z$	m_7	$x' + y' + z'$	M_7

Sumber:

[https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/11-Aljabar-Boolean-\(2023\)-bagian1.pdf](https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/11-Aljabar-Boolean-(2023)-bagian1.pdf)

E. Rangkaian Logika

Fungsi Boolean dapat direpresentasikan dalam bentuk rangkaian logika. Rangkaian logika ini merujuk pada susunan dan interkoneksi dari gerbang logika, yang menggunakan prinsip-prinsip aljabar Boolean untuk menghasilkan keluaran berdasarkan nilai-nilai input. Rangkaian ini adalah implementasi fisik dari operasi-operasi aljabar Boolean, dan umumnya digunakan dalam desain sirkuit digital. Terdapat tiga gerbang logika dasar yang membentuk rangkaian logika dalam aljabar Boolean yaitu gerbang AND, gerbang OR, dan gerbang NOT.



Gambar 2. Gerbang Logika Dasar

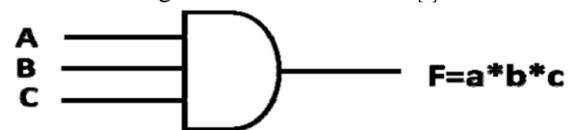
Sumber:

[https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/11-Aljabar-Boolean-\(2023\)-bagian1.pdf](https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/11-Aljabar-Boolean-(2023)-bagian1.pdf)

Penjelasan lebih lanjut dan contoh penggunaan gerbang tertera berikut ini.

1. Gerbang AND

Gerbang AND memiliki dua atau lebih sinyal masukan tetapi hanya satu sinyal keluaran. Pada Gerbang AND, semua masukan harus bernilai *true* baru akan menghasilkan keluaran *true* [8].



Gambar 3. Aplikasi Gerbang AND pada Tiga Peubah

Sumber:

https://www.academia.edu/8846853/Logic_Gates_and_Boolean_Algebra

Truth table dari gambar tersebut tertera di bawah ini.

Tabel 8. Truth Table Gerbang AND Tiga Peubah

A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

2. Gerbang OR

Pada gerbang OR, jika salah satu sinyal masukan *true*, maka keluaran akan *true*. Keluaran hanya menjadi *false* ketika semua masukan adalah *false* [8].



Gambar 4. Aplikasi Gerbang OR pada Tiga Peubah

Sumber:

https://www.academia.edu/8846853/Logic_Gates_and_Boolean_Algebra

Truth table dari gambar tersebut tertera di bawah ini.

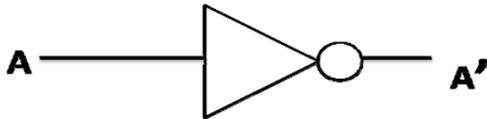
Tabel 9. Truth Table Gerbang OR Tiga Peubah

A	B	C	F
0	0	0	0
0	0	1	1

0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

3. Gerbang NOT

Gerbang NOT juga dikenal sebagai Inverter. Pada Gerbang NOT, hanya terdapat satu sinyal masukan dan satu sinyal keluaran, di mana keadaan keluaran selalu berlawanan dengan keadaan masukan.



Gambar 5. Aplikasi Gerbang NOT

Sumber:

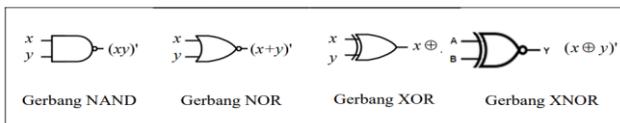
https://www.academia.edu/8846853/Logic_Gates_and_Boolean_Algebra

Truth table dari gambar tersebut tertera di bawah ini.

Tabel 10. Truth Table Gerbang NOT

A	A'
0	1
1	0

Selain gerbang logika dasar, terdapat juga gerbang logika turunan diantaranya adalah gerbang NAND, gerbang NOR, gerbang XOR, dan gerbang XNOR [6]. Penjelasan lebih lanjut tertera di bawah ini.



Gambar 6. Gerbang Logika Lanjutan

Sumber:

[https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/11-Aljabar-Boolean-\(2023\)-bagian1.pdf](https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/11-Aljabar-Boolean-(2023)-bagian1.pdf)

III. PEMBAHASAN

Sistem dari *smart home device* sendiri banyak yang berbasis sensor. Sistem ini merujuk pada jaringan perangkat elektronik yang dilengkapi berbagai jenis sensor untuk mendeteksi dan merespons kondisi tertentu di rumah dan sekitarnya. Sensor-sensor ini berperan penting dalam memberikan informasi yang diperlukan untuk otomatisasi dan kendali perangkat. Sistem ini menggunakan informasi yang diberikan oleh sensor-sensor ini untuk membuat keputusan otomatis atau memberikan pemberitahuan kepada penghuni rumah. Pengguna dapat mengelola dan mengawasi kondisi rumah mereka melalui aplikasi *handphone*, yang memungkinkan mereka mengakses informasi dari sensor-sensor tersebut dan mengendalikan *smart home devices* dengan lebih

efisien. Beberapa sensor yang sering digunakan adalah sensor gerak, sensor suhu, sensor kelembaban, sensor cahaya, sensor warna, sensor suara, sensor asap, serta sensor karbon monoksida. Sensor-sensor ini bekerja sesuai dengan *smart home device* yang dibutuhkan. Tidak semua *smart home device* mencakup sensor yang sama. Pembahasan lebih lanjut mengenai sistem tiap *smart home device* tertera pada subbab-subbab berikut ini.

A. Lampu Otomatis (Smart Lamp)

Pada saklar *smart lamp* di rumah, fungsi gerbang logika dimanfaatkan untuk mengatur penggunaan dan penyalahgunaan lampu berdasarkan kondisi tertentu. Gerbang logika AND dapat digunakan agar lampu hanya otomatis menyala saat sensor gerak mendeteksi pergerakan dan sensor cahaya lingkungan mendeteksi tingkat cahaya yang rendah dengan batas intensitas tertentu [10]. Pendekatan desain yang memanfaatkan gerbang logika ini tidak hanya efisien dalam penggunaan energi, tetapi juga meningkatkan secara keseluruhan pengalaman otomatisasi. Representasi gerbang logika pada *smart lamp* dapat dinyatakan melalui tabel kebenaran dan skema rangkaian. Tabel kebenaran menampilkan hasil keluaran dari gerbang logika untuk semua kemungkinan kombinasi masukan, sementara skema rangkaian menggambarkan hubungan antara berbagai komponen elektronik.

Implementasi gerbang AND beserta komponen cahaya dan juga pergerakan tertera pada gambar di bawah ini. Program yang saya buat menggunakan bahasa C. Program pertama ini berfungsi untuk mengimplementasikan gerbang AND, dimana kedua input harus sama-sama bernilai *true* agar hasilnya pun bernilai *true*, sedangkan bila kombinasi input bernilai lain akan menghasilkan nilai *false*.

```
/* Fungsi gerbang Logika AND */
boolean andGate(boolean input1, boolean input2){
    return (input1 && input2);
}
```

Gambar 7. Implementasi gerbang AND

Sumber: Dokumen Pribadi

Seperti yang sudah dijelaskan pada paragraf sebelumnya, lampu akan menyala bila sensor cahaya lingkungan mendeteksi tingkat cahaya yang rendah. Pada program yang saya buat, saya membuat batas bahwa intensitas cahaya < 300 akan terdeteksi sebagai cahaya rendah sehingga akan menghasilkan nilai *true*.

```
/* Fungsi untuk mengecek tingkat intensitas cahaya */
boolean checkLowLightIntensity(int cahaya){
    return (cahaya < 300);
}
```

Gambar 8. Pengecekan Tingkat Cahaya Rendah/Tinggi

Sumber: Dokumen Pribadi

Program terakhir yang dibuat adalah implementasi kedua program di atas. Dimana keluaran dari program ini adalah keterangan apakah sebuah *smart lamp* sudah waktunya untuk menyala atau tidak. Jika sudah waktunya menyala akan mengeluarkan keterangan "Lampu menyala", sedangkan jika tidak akan mengeluarkan keterangan

“Lampu mati”. Nilai `lowLight` dan gerakan harus `true` untuk membuat `smart lamp` menyala.

```
/* Sistem smart Lamp untuk deteksi cahaya dan gerakan */
void smartLamp(boolean lowLight, boolean gerakan){
    boolean lightOn = andGate(lowLight, gerakan);

    if (lightOn) {
        printf("Lampu menyala");
    }
    else {
        printf("Lampu mati");
    }
}
}
```

Gambar 9. Implementasi Smart Lamp dengan Gerbang AND

Sumber: Dokumen Pribadi

Program utama lampu otomatis ini tertera sebagai berikut.

```
int main () {
    int cahaya;
    boolean gerakan;

    printf("Tingkat intensitas cahaya: ");
    scanf("%d", &cahaya);
    printf("Apakah ada gerakan? 1 jika iya, 0 jika tidak: ");
    scanf("%d", &gerakan);

    boolean lowLight = checkLowLightIntensity(cahaya);
    smartLamp(lowLight, gerakan);
    return 0;
}
```

Gambar 10. Program Utama Smart Lamp

Sumber: Dokumen Pribadi

Contoh dari penggunaan program tersebut akan ditampilkan sebagai berikut.

1. Kasus dimana intensitas cahaya rendah dan terdeteksi pergerakan, maka lampu akan menyala.

```
PS D:\ITB\SEM 3\Matdis\Tugas> ./tes
Tingkat intensitas cahaya: 100
Apakah ada gerakan? 1 jika iya, 0 jika tidak: 1
Lampu menyala
```

Gambar 11. Kasus Lampu Menyala

Sumber: Dokumen Pribadi

Karena intensitas cahayanya < 300 , maka akan bernilai `true` atau ‘1’. Terdeteksi pula adanya pergerakan sehingga bernilai `true` atau ‘1’ pula.

2. Kasus dimana intensitas cahaya rendah dan tidak terdeteksi pergerakan, maka lampu akan tetap mati.

```
PS D:\ITB\SEM 3\Matdis\Tugas> ./tes
Tingkat intensitas cahaya: 100
Apakah ada gerakan? 1 jika iya, 0 jika tidak: 0
Lampu mati
```

Gambar 12. Kasus Lampu Mati (1)

Sumber: Dokumen Pribadi

Karena intensitas cahayanya < 300 , maka akan bernilai `true` atau ‘1’. Namun tidak terdeteksi adanya pergerakan sehingga bernilai `false` atau ‘0’.

3. Kasus dimana intensitas cahaya tinggi dan terdeteksi pergerakan, maka lampu akan tetap mati.

```
PS D:\ITB\SEM 3\Matdis\Tugas> ./tes
Tingkat intensitas cahaya: 600
Apakah ada gerakan? 1 jika iya, 0 jika tidak: 1
Lampu mati
```

Gambar 13. Kasus Lampu Mati (2)

Sumber: Dokumen Pribadi

Karena intensitas cahayanya ≥ 300 , maka akan bernilai `false` atau ‘0’. Terdeteksi adanya pergerakan menyebabkan akan bernilai `true` atau ‘1’.

4. Kasus dimana intensitas cahaya tinggi dan terdeteksi pergerakan, maka lampu akan tetap mati.

```
PS D:\ITB\SEM 3\Matdis\Tugas> ./tes
Tingkat intensitas cahaya: 600
Apakah ada gerakan? 1 jika iya, 0 jika tidak: 0
Lampu mati
```

Gambar 14. Kasus Lampu Mati (3)

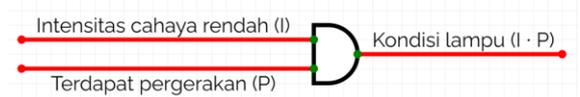
Sumber: Dokumen Pribadi

Karena intensitas cahayanya ≥ 300 , maka akan bernilai `false` atau ‘0’. Tidak terdeteksi adanya pergerakan menyebabkan bernilai `false` atau ‘0’.

Dari keempat kondisi di atas, dapat disimpulkan kondisi-kondisi tersebut dalam *truth table* dan gerbang logika berikut.

Tabel 11. Truth Table Smart Lamp

Apakah intensitas cahaya rendah?	Apakah ada pergerakan?	Kondisi lampu (1 menyala, 0 mati)
1	1	1
1	0	0
0	1	0
0	0	0



Gambar 15. Logic Gate Smart Lamp

Sumber: Dokumen Pribadi

B. Pagar Otomatis (Automatic Gate)

Dalam sistem pagar otomatis, gerbang logika digunakan untuk mengatur akses ke properti. Dengan mengombinasikan berbagai gerbang logika, kita bisa membuat sistem pengaturan yang aman dan efisien. Penerapan gerbang logika dalam sistem pagar otomatis memungkinkan otomatisasi yang mulus dan meningkatkan tingkat keamanan. Kombinasi ini dengan Internet of Things (IoT) dan teknologi sensor memungkinkan pengelolaan data dan pengendalian yang lebih canggih, menjadikan pengalaman smart home menjadi lebih praktis dan aman [10].

Pagar ini dapat terbuka dengan dua kondisi. Bisa dengan kode akses yang benar berupa kartu, dan juga bisa dengan deteksi adanya kendaraan dan pembacaan plat nomor kendaraan yang sesuai dengan plat nomor kendaraan pemilik rumah. Kita bisa memanfaatkan gerbang OR dan AND untuk memastikan bahwa pagar tetap terkunci kecuali jika kondisi yang telah dijelaskan sebelumnya terpenuhi. Model implementasi pagar otomatis beserta persyaratan tersebut tertera pada program berikut.

Pada program pagar otomatis ini membutuhkan kombinasi dua gerbang logika yaitu gerbang AND dan gerbang OR. Implementasi programnya sebagai berikut.

```

/* Fungsi gerbang Logika OR */
boolean orGate(boolean input1, boolean input2) {
    return (input1 || input2);
}

/* Fungsi gerbang Logika AND */
boolean andGate(boolean input1, boolean input2){
    return (input1 && input2);
}

```

Gambar 16. Implementasi Gerbang OR dan Gerbang AND

Sumber: Dokumen Pribadi

Program selanjutnya adalah program untuk mengecek plat nomor kendaraan yang akan masuk. Pengecekan ini dilakukan terhadap list plat kendaraan-kendaraan di rumah tersebut. Apabila plat yang akan masuk ditemukan pada list tersebut, akan bernilai *true*.

```

/* Fungsi untuk mengecek kebenaran plat kendaraan pemilik rumah */
boolean checkLicensePlate(char* inputPlate, char* plateList[], int listSize) {
    boolean plateFound = false;
    int i = 0;
    while (!plateFound && i < listSize) {
        if (strcmp(inputPlate, plateList[i]) == 0) {
            plateFound = true;
        }
        i++;
    }
    return plateFound;
}

```

Gambar 17. Implementasi Checking Plat Nomor Kendaraan

Sumber: Dokumen Pribadi

Program terakhir adalah implementasi pagar otomatis. Program ini akan mengeluarkan *true* jika access code bernilai *true* alias pada kenyataannya adalah pemilik rumah mempunyai kartu akses yang dapat membuka pagar. Program juga akan bernilai *true* jika kendaraan yang berada di depan pagar adalah tipe kendaraan yang diizinkan dan juga platnya sesuai dengan list plat kendaraan yang dimiliki rumah tersebut (dicek melalui program checkLicensePlate). Jika keluaran bernilai *true*, program akan mengeluarkan pesan "Akses diterima, pagar terbuka." sebagai tanda bahwa pagar telah otomatis terbuka. Namun sebaliknya bila keluaran bernilai *false*, program akan mengeluarkan "Akses ditolak, pagar terkunci." sebagai tanda bahwa pagar tidak dapat terbuka.

```

/* Fungsi pagar otomatis dengan gerbang OR dan AND */
void automaticGate(boolean accessCode, boolean authorizedVehicle, boolean licensePlate) {
    boolean checkVehicle = andGate(authorizedVehicle, licensePlate);
    if (orGate(accessCode, checkVehicle)) {
        printf("Akses diterima, pagar terbuka.");
    }
    else {
        printf("Akses ditolak, pagar terkunci.");
    }
}

```

Gambar 18. Implementasi Automatic Gate

Sumber: Dokumen Pribadi

Program utama pagar otomatis ini tertera sebagai berikut.

```

int main() {
    boolean accessCode, authorizedVehicle, licensePlate;
    char* daftarPlatRumah[] = {"B1327ATJ", "B1436NZT", "N3010LG", "D1202AN", "A1725AAT"};

    printf("Apakah kendaraan yang akan masuk adalah kendaraan yang diizinkan?");
    printf("\n1 jika iya, 0 jika tidak: ");
    scanf("%d", &authorizedVehicle);

    printf("Kode akses (bila pemilik rumah = 1, bila bukan = 0): ");
    scanf("%d", &accessCode);

    printf("Masukkan plat kendaraan: ");
    char* inputPlate = (char*) malloc(8 * sizeof(char));
    scanf("%s", inputPlate);
    int listSize = ARRAY_LENGTH(daftarPlatRumah);
    licensePlate = checkLicensePlate(inputPlate, daftarPlatRumah, listSize);

    automaticGate(accessCode, authorizedVehicle, licensePlate);
    return 0;
}

```

Gambar 19. Program Utama Automatic Gate

Sumber: Dokumen Pribadi

Contoh dari penggunaan program tersebut akan ditampilkan sebagai berikut dengan contoh list plat kendaraan pada rumah tersebut adalah ["B1327ATJ", "B1436NZT", "N3010LG", "D1202AN", "A1725AAT"].

1. Kasus dimana pengguna memiliki kartu akses, tipe kendaraan adalah tipe yang diizinkan, dan plat nomor kendaraan sesuai dengan list.

```

PS D:\ITB\SEM 3\Matdis\Tugas> ./tes
Apakah kendaraan yang akan masuk adalah kendaraan yang diizinkan?
1 jika iya, 0 jika tidak: 1
Kode akses (bila pemilik rumah = 1, bila bukan = 0): 1
Masukkan plat kendaraan: B1436NZT
Akses diterima, pagar terbuka.

```

Gambar 20. Kasus Pagar Terbuka (1)

Sumber: Dokumen Pribadi

Karena pengguna memiliki kartu akses maka bernilai *true*. Kendaraan merupakan tipe yang diizinkan sehingga bernilai *true*, serta plat nomor kendaraan sesuai dengan list sehingga bernilai *true* juga.

2. Kasus dimana pengguna memiliki kartu akses, tipe kendaraan adalah tipe yang diizinkan, dan plat nomor kendaraan tidak sesuai dengan list.

```

PS D:\ITB\SEM 3\Matdis\Tugas> ./tes
Apakah kendaraan yang akan masuk adalah kendaraan yang diizinkan?
1 jika iya, 0 jika tidak: 1
Kode akses (bila pemilik rumah = 1, bila bukan = 0): 1
Masukkan plat kendaraan: A1627NDJ
Akses diterima, pagar terbuka.

```

Gambar 21. Kasus Pagar Terbuka (2)

Sumber: Dokumen Pribadi

Karena pengguna memiliki kartu akses maka bernilai *true*. Kendaraan merupakan tipe yang diizinkan sehingga bernilai *true*, namun plat nomor kendaraan tidak sesuai dengan list sehingga bernilai *false* juga. Pagar tetap akan terbuka karena pengguna memiliki kartu akses sehingga accessCode bernilai *true*.

3. Kasus dimana pengguna tidak memiliki kartu akses, tipe kendaraan adalah tipe yang diizinkan, dan plat nomor kendaraan sesuai dengan list.

```

PS D:\ITB\SEM 3\Matdis\Tugas> ./tes
Apakah kendaraan yang akan masuk adalah kendaraan yang diizinkan?
1 jika iya, 0 jika tidak: 1
Kode akses (bila pemilik rumah = 1, bila bukan = 0): 0
Masukkan plat kendaraan: N3010LG
Akses diterima, pagar terbuka.

```

Gambar 22. Kasus Pagar Terbuka (3)

Sumber: Dokumen Pribadi

Karena pengguna tidak memiliki kartu akses maka bernilai *false*. Kendaraan merupakan tipe yang

diizinkan sehingga bernilai *true*, serta plat nomor kendaraan sesuai dengan list sehingga bernilai *true* juga.

4. Kasus dimana pengguna memiliki tidak memiliki kartu akses, tipe kendaraan adalah bukan tipe yang diizinkan, dan plat nomor kendaraan tidak sesuai dengan list.

```
PS D:\ITB\SEM 3\Matdis\Tugas> ./tes
Apakah kendaraan yang akan masuk adalah kendaraan yang diizinkan?
1 jika iya, 0 jika tidak: 0
Kode akses (bila pemilik rumah = 1, bila bukan = 0): 0
Masukkan plat kendaraan: D1729NA
Akses ditolak, pagar terkunci.
```

Gambar 23. Kasus Pagar Terkunci

Sumber: Dokumen Pribadi

Karena pengguna tidak memiliki kartu akses maka bernilai *false*. Kendaraan bukan merupakan tipe yang diizinkan sehingga bernilai *false*, serta plat nomor kendaraan tidak sesuai dengan list sehingga bernilai *false* juga. Hal ini mengakibatkan pagar tetap dalam kondisi terkunci.

Dari keempat kondisi di atas, dapat disimpulkan kondisi-kondisi tersebut dalam *truth table* dan gerbang logika berikut.

Tabel 12. Truth Table Automatic Gate

Memiliki kartu akses?	Tipe kendaraan diizinkan?	Plat nomor kendaraan sesuai?	Kondisi pagar (1 terbuka, 0 terkunci)
1	1	1	1
1	1	0	1
0	1	1	1
0	0	0	0



Gambar 24. Logic Gate Smart Gate

Sumber: Dokumen Pribadi

C. Penyiram Otomatis

Penyiram otomatis adalah *smart home device* yang berguna untuk mengotomatisasi irigasi tanaman dengan memanfaatkan sensor kelembaban tanah dan sensor cahaya [11]. Dengan menggunakan teknologi sensor kelembaban tanah, perangkat ini dapat mendeteksi tingkat kelembaban pada tanah, sehingga penyiraman hanya dilakukan ketika tanah berada pada kondisi tidak lembab. Selain itu, sensor cahaya digunakan untuk menyesuaikan jadwal penyiraman berdasarkan intensitas cahaya di sekitar tanaman, memastikan bahwa proses irigasi disesuaikan dengan kondisi pencahayaan yang optimal. Dengan demikian, penyiram otomatis tidak hanya meningkatkan efisiensi penggunaan air, tetapi juga melindungi tanaman dan memberikan solusi irigasi yang cerdas dan ramah lingkungan.

Penyiram otomatis ini dapat menyala dengan kondisi dimana tingkat kelembaban tanah rendah dan intensitas cahaya tinggi. Untuk merealisasikan persyaratan ini dapat memanfaatkan gerbang NOT dan AND untuk memastikan

bahwa penyiram tetap dalam kondisi mati kecuali jika kondisi yang telah dijelaskan sebelumnya terpenuhi. Model implementasi penyiram otomatis beserta persyaratan tersebut tertera pada program berikut.

Pada program penyiram otomatis ini membutuhkan kombinasi dua gerbang logika yaitu gerbang NOT dan gerbang AND. Implementasi programnya sebagai berikut.

```
/* Fungsi gerbang NOT */
boolean notGate(boolean input) {
    return (!input);
}

/* Fungsi gerbang AND */
boolean andGate(boolean input1, boolean input2){
    return (input1 && input2);
}
```

Gambar 25. Implementasi Gerbang NOT dan Gerbang AND

Sumber: Dokumen Pribadi

Program selanjutnya adalah program untuk mengecek tingkat kelembaban tanah. Bila kelembaban tanah > 0.5 maka akan bernilai *true* yang artinya tanah tersebut dalam kondisi lembab.

```
// Fungsi untuk mendeteksi suatu tanah Lembab atau tidak
boolean soilMoisture(float kelembabanTanah) {
    float batasLembab = 0.5;
    /* Mengembalikan true jika tanah Lembab */
    return (kelembabanTanah > batasLembab);
}
```

Gambar 26. Implementasi Deteksi Kelembaban Tanah

Sumber: Dokumen Pribadi

Program selanjutnya adalah program untuk mengecek intensitas cahaya. Bila intensitas cahaya > 300 maka akan bernilai *true* yang artinya tingkat intensitas cahaya dalam kondisi tinggi.

```
// Fungsi untuk mendeteksi intensitas cahaya tinggi/rendah
boolean lightIntensity(float intensitasCahaya) {
    float batasCahaya = 300;
    /* Mengembalikan true jika intensitas cahaya tinggi yaitu > 300 */
    return (intensitasCahaya > batasCahaya);
}
```

Gambar 27. Implementasi Deteksi Intensitas Cahaya

Sumber: Dokumen Pribadi

Program terakhir adalah implementasi penyiram otomatis. Program ini akan mengeluarkan *true* jika tanah tidak lembab dan intensitas cahaya tinggi. Maka dari itu, hasil deteksi kelembaban tanah sebelumnya harus diimplementasikan pada gerbang NOT untuk mendapat hasil tanah tidak lembab. Lalu akan diimplementasi pada gerbang AND bersama dengan intensitas cahaya tinggi. Bila keluaran bernilai *true*, program akan mengeluarkan "Penyiram otomatis menyala.". Begitu pula sebaliknya bila keluaran bernilai *false*, program akan mengeluarkan "Penyiram otomatis mati."

```
// Fungsi untuk mendeteksi apakah tanah perlu disiram
boolean automaticIrrigation(boolean soilMoisture, boolean lightIntensity) {
    /* Mengembalikan true jika tanah tidak Lembab dan intensitas cahaya tinggi */
    boolean perluDisiram = andGate(notGate(soilMoisture), lightIntensity);
    if (perluDisiram) {
        printf("Penyiram otomatis menyala.");
    }
    else {
        printf("Penyiram otomatis mati.");
    }
}
```

Gambar 28. Implementasi Penyiram Otomatis

Sumber: Dokumen Pribadi

Program utama penyiram otomatis ini tertera sebagai berikut.

```
int main() {
    float kelembabanTanah, intensitasCahaya;
    boolean lembab, intensitas;
    printf("Kelembaban tanah: ");
    scanf("%f", &kelembabanTanah);
    printf("Intensitas cahaya: ");
    scanf("%f", &intensitasCahaya);
    lembab = soilMoisture(kelembabanTanah);
    intensitas = lightIntensity(intensitasCahaya);
    automaticIrrigation(lembab, intensitas);
    return 0;
}
```

Gambar 29. Implementasi Automatic Gate

Sumber: Dokumen Pribadi

Contoh dari penggunaan program tersebut akan ditampilkan sebagai berikut.

1. Kasus dimana tanah tidak lembab dan intensitas cahaya tinggi.

```
PS D:\ITB\SEM 3\Matdis\Tugas> ./tes
Kelembaban tanah: 0.2
Intensitas cahaya: 400
Penyiram otomatis menyala.
```

Gambar 30. Kasus Penyiram Menyala

Sumber: Dokumen Pribadi

Karena tanah tidak lembab maka bernilai *false*. Setelah diimplementasikan pada gerbang NOT, status kelembaban tanah akan bernilai *true*. Intensitas cahaya tinggi maka bernilai *true*. Kombinasi kedua kondisi ini menyebabkan penyiram menyala.

2. Kasus dimana tanah tidak lembab dan intensitas cahaya rendah.

```
PS D:\ITB\SEM 3\Matdis\Tugas> ./tes
Kelembaban tanah: 0.2
Intensitas cahaya: 100
Penyiram otomatis mati.
```

Gambar 31. Kasus Penyiram Mati (1)

Sumber: Dokumen Pribadi

Karena tanah tidak lembab maka bernilai *false*. Setelah diimplementasikan pada gerbang NOT, status kelembaban tanah akan bernilai *true*. Intensitas cahaya rendah maka bernilai *false*. Kombinasi kedua kondisi ini menyebabkan penyiram mati.

3. Kasus dimana tanah lembab dan intensitas cahaya rendah.

```
PS D:\ITB\SEM 3\Matdis\Tugas> ./tes
Kelembaban tanah: 1.2
Intensitas cahaya: 100
Penyiram otomatis mati.
```

Gambar 32. Kasus Penyiram Mati (1)

Sumber: Dokumen Pribadi

Karena tanah lembab maka bernilai *true*. Setelah diimplementasikan pada gerbang NOT, status kelembaban tanah akan bernilai *false*. Intensitas

cahaya rendah maka bernilai *false*. Kombinasi kedua kondisi ini menyebabkan penyiram mati.

4. Kasus dimana tanah lembab dan intensitas cahaya tinggi.

```
PS D:\ITB\SEM 3\Matdis\Tugas> ./tes
Kelembaban tanah: 1.2
Intensitas cahaya: 500
Penyiram otomatis mati.
```

Gambar 33. Kasus Penyiram Mati (1)

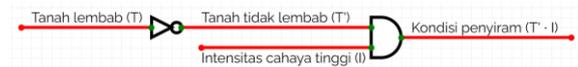
Sumber: Dokumen Pribadi

Karena tanah lembab maka bernilai *true*. Setelah diimplementasikan pada gerbang NOT, status kelembaban tanah akan bernilai *false*. Intensitas cahaya tinggi maka bernilai *true*. Kombinasi kedua kondisi ini menyebabkan penyiram mati.

Dari keempat kondisi di atas, dapat disimpulkan kondisi-kondisi tersebut dalam *truth table* dan gerbang logika berikut.

Tabel 13. Truth Table Penyiram Otomatis

Tanah lembab? (T)	T'	Intensitas cahaya tinggi? (I)	Kondisi penyiram (1 menyala, 0 mati)
1	0	1	0
1	0	0	0
0	1	1	1
0	1	0	0



Gambar 34. Logic Gate Smart Gate

Sumber: Dokumen Pribadi

IV. KESIMPULAN

Pemanfaatan aljabar Boolean dalam *smart home* memberikan solusi efisien dan mudah diimplementasikan, dibandingkan dengan metode kompleks seperti logic rules, artificial neural networks, fuzzy neural network, support vector machines, dan Bayesian networks. Aljabar Boolean berperan dalam merancang aturan logika untuk mengatur respons perangkat terhadap kondisi seperti suhu, keamanan, dan pencahayaan. Dengan menggunakan aljabar Boolean, *smart home devices* dapat diprogram untuk merespons situasi khusus, seperti mengatur pemanas berdasarkan suhu. Penerapan aljabar Boolean juga mendukung perancangan logika untuk mengoptimalkan penggunaan energi, mengelola efisiensi energi, dan meningkatkan keamanan dengan mendeteksi kejadian mencurigakan. *Smart home devices* dapat dijadwalkan secara otomatis berdasarkan aturan waktu atau kondisi tertentu sesuai kebutuhan pengguna.

Hasil dari fungsi Boolean untuk tiap jenis *smart home devices* berbeda-beda, tergantung oleh kondisi yang diperlukan oleh perangkat tersebut. Seperti yang sudah dibahas pada bab III, untuk lampu otomatis akan menyala bila hasil fungsi Boolean bernilai 1 atau *true* yaitu di saat intensitas cahaya

rendah dan terdeteksi pergerakan. Lalu untuk pagar otomatis akan terbuka bila hasil fungsi Boolean bernilai 1 atau *true* yaitu di saat pengguna memiliki kartu akses atau kendaraan yang berada di depan pagar termasuk kendaraan yang diizinkan dengan plat nomor yang sesuai. Dan yang terakhir adalah penyiram otomatis akan menyala bila hasil fungsi Boolean bernilai 1 atau *true* yaitu di saat kondisi tanah tidak lembab dan intensitas cahaya tinggi.

V. UCAPAN TERIMA KASIH

Sebagai penulis makalah ini, saya ingin mengungkapkan rasa terima kasih yang tulus kepada semua pihak yang telah memberikan dukungan dan inspirasi selama proses penulisan sehingga saya dapat menyelesaikan makalah yang berjudul “Penerapan Aljabar Boolean pada *Smart Home Devices*” ini dengan baik. Saya ucapkan terima kasih kepada:

1. Ibu Dr. Nur Ulfa Maulidevi, Ibu Dr. Fariska Zakhrativa Ruskanda, dan Bapak Dr. Rinaldi Munir sebagai dosen pengajar mata kuliah IF2120 Matematika Diskrit atas pengajaran materi-materi yang telah dibagikan di kelas Teknik Informatika.
2. Kedua orang tua saya yang selalu memberikan dukungan moral dan doa restu. Kehadiran dan semangat positif dari kedua orang tua saya memberikan semangat lebih untuk menyelesaikan tugas ini dengan baik.
3. Para penulis yang telah menciptakan karya-karya yang menjadi landasan bagi penulisan makalah ini. Referensi dari jurnal dan artikel-artikel telah memberikan wawasan dan kontribusi penting pada pemahaman topik.

Makalah ini tidak mungkin terwujud tanpa kontribusi berharga dari setiap individu yang disebutkan di atas. Semua bantuan dan dukungan yang diberikan telah menjadi pendorong keberhasilan penulisan makalah ini. Sekali lagi, terima kasih banyak atas segala bantuan dan dukungan yang ada. Semoga makalah ini dapat bermanfaat bagi orang lain.

REFERENSI

- [1] D. Debnath, A. H. Siddique, M. Hasan et al., “Smart Electrification of Rural Bangladesh through Smart Grid,” *Sustainable Communication Networks and Application*, vol. 55, 2020
- [2] M. Hasan, P. Biswas, M. T. I. Bilash, and M. A. Z. Dipto, “Smart Home Systems: Overview And Comparative Analysis,” in *Proceedings of the 2018 Fourth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN)*, pp. 264–268, Kolkata, India, November 2018. H. Poor, *An Introduction to Signal Detection and Estimation*. New York: Springer-Verlag, 1985, ch. 4.
- [3] Fang, X., Misra, S., Xue, G., & Yang, D. (2012). Smart grid — the new and improved power grid: a survey. *IEEE Communications Surveys & Tutorials*, 14(4), 944-980. E. H. Miller, “A note on reflector arrays (Periodical style—Accepted for publication),” *IEEE Trans. Antennas Propagat.*, to be published.
- [4] Humayun, Kabir., Hoque, Robiul., & Yang, Sung-Hyun. “Implementation of Boolean Control Network Based Intelligent System in Smart Home”. Seoul, South Korea. 2016.
- [5] Sarma, Subhrajit., Bhuyan, Rama Kanta. “Boolean Algebra and Logic Gates”. Article in *International Journal of Mathematics Trends and Technology*. Maret, 2019.
- [6] Rinaldi Munir. “Aljabar Boolean Bagian 1.” Homepage Rinaldi Munir. Sekolah Teknik Elektro dan Informatika (STEI) ITB. [https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/11-Aljabar-Boolean-\(2023\)-bagian1.pdf](https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/11-Aljabar-Boolean-(2023)-bagian1.pdf) [Diakses 7 Desember 2023]

- [7] Testbook. (2023, May 4). *Boolean Algebra: Definition, Laws, Rules & Theorems with Examples*. Testbook. [Online]. Available: <https://testbook.com/maths/boolean-algebra> [Diakses 8 Desember 2023]
- [8] Raval, Kalyani. “Logic Gates and Boolean Algebra”. *International Journal of Multidisciplinary Research Hub*. Vindhyanager, India. November, 2014.
- [9] Wsk, G. (2023, October 24). *Tabel Kebenaran gerbang logika*. WikiElektronika.com. [Online]. Available: <https://wikielektronika.com/gerbang-logika/> [Diakses 9 Oktober 2023]
- [10] Geeks, L. (2023, October 13). *Logic Gate in Smart home Devices: Enhancing automation and Security - Lambda Geeks*. <https://lambdageeks.com/logic-gate-in-smart-home-devices/> [Diakses 9 Oktober 2023]
- [11] Munandar, M. F. S., Nurlaela, L., & Bangsa, I. A. (2022). Implementasi Penyiraman Otomatis dengan Sensor Gy-302 dan YL-69 pada Alat Penyiram Tanaman. *Jurnal JE-UNISLA: Electronic Control, Telecommunication, Computer Information and Power System*, 7(1), 1. <https://doi.org/10.30736/je-unisla.v7i1.750>

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 10 Desember 2023



Erdianti Wiga Putri Andini
13522053