Algoritma Kendali *Elevator Group* Menggunakan Metode *Car Nearest* Berbasis Raspberry Pico

Nafisah Mardhiyyah[1], Dwi Septiyanto[2], Nanang Mulyono[3]

[1,2,3]*Jurusan Teknik Elektro, Politeknik Negeri Bandung, Bandung 40012*
[1]*E-mail :* nafisah.mardhiyyah.toi20*@polban.ac.id*
[2]*E-mail :* dwi.septianto *@polban.ac.id*
[3]*E-mail :* nanang.mulyono *@polban.ac.id*

ABSTRAK

Penggunaan *lift group* sangat penting terutama pada gedung-gedung tinggi dengan banyaknya jumlah permintaan. Tanpa kendali yang efektif, waktu tunggu dan kepadatan di dalam *lift* dapat meningkat yang menyebabkan ketidaknyamanan bagi pengguna. Oleh karena itu, perlu adanya kendali yang tepat pada *lift.* Pada penelitian ini, algoritma kendali diaplikasikan pada miniatur *lift group* yang terdiri dari dua kabin *lift, lift* A dan *lift* B. Masing-masing *lift* beroperasi pada lima lantai yang digerakkan oleh motor AC 3 fasa. Tujuan dari kendali *lift group* ini adalah untuk mengatur *lift* yang akan ditugaskan untuk memenuhi panggilan dan permintaan sehingga pergerakan *lift* lebih optimal, efisien, dan mengurangi waktu tunggu. Kendali yang dibuat menggunakan metode *car nearest.* Posisi *lift,* arah panggilan pendaratan, dan arah *lift* adalah tiga faktor utama yang digunakan dalam metode *car nearest* ini. FS merupakan suatu nilai numerik dengan mempertimbangkan tiga faktor utama tadi. *Lift* dengan nilai FS tertinggi akan ditugaskan untuk memenuhi permintaan dan panggilan yang ada. Diharapkan dengan adanya kendali pada *lift group* ini dapat meningkatkan efisiensi dan mempersingkat waktu tunggu yang dihabiskan oleh pengguna**.** Hasil dari algoritma yang dijalankan adalah permintaan akan dipenuhi oleh *lift* dengan jarak terdekat dan searah dengan pergerakan permintaan dengan nilai FS tertinggi.

Kata Kunci
Gedung, miniatur, lift group, car nearest, Raspberry pico

ABSTRACT

The use of lift groups is crucial, especially in tall buildings with high demand. Without effective control, waiting times and crowding in elevators can increase, causing inconvenience and reducing the efficiency of vertical transportation in these buildings. This control algorithm is applied to a miniature lift group consisting of two lift cabins, Lift A and Lift B. Each lift operates on five floors, driven by a 3-phase AC motor. The purpose of this lift group control is to assign lifts to fulfill calls and requests optimally and efficiently, thereby reducing waiting time. Controls are implemented using the nearest car method. The position of the lift, the direction of the landing call, and the direction of the lift are the three main factors in this method. FS is a numerical value that considers these three factors. The lift with the highest FS value will be assigned to fulfill the requests and calls. This lift group control aims to increase efficiency and shorten the waiting time for users. The algorithm ensures that the request is fulfilled by the elevator with the closest distance and moving in the same direction as the request, based on the highest FS value.

Keywords
Building, miniature, lift group, car nearest, Raspberry pico

1. PENDAHULUAN

Kendali pada *lift group* diperlukan untuk mengatur mobilisasi pergerakan *lift* di dalam suatu bangunan. *Lift* menjadi bagian vital untuk gedung-gedung tinggi dengan lantai banyak. Namun, dengan banyaknya pengguna pada suatu gedung, satu *lift* tidak dapat menampung seluruh arus penumpang sehingga perlu dipasang beberapa atau lebih *lift* yang disebut *lift group*. Kendali pada *lift group* menjadi semakin penting untuk

beberapa alasan utama. Seiring dengan pembangunan gedung-gedung tinggi yang meningkat, permintaan layanan *lift* pun semakin meningkat signifikan. Tanpa kendali yang efektif, waktu tunggu dan kepadatan di dalam *lift* dapat meningkat yang menyebabkan ketidaknyamanan bagi pengguna dan menurunkan efisiensi transportasi vertikal di gedung-gedung tersebut. Selain itu, dengan adanya kendali pada *lift group,* pengoperasian pada *lift* dapat dioptimalkan untuk mengurangi konsumsi energi. Pengaturan rute yang tepat dapat mengurangi pemakaian listrik secara signifikan, serta memberikan manfaat baik secara ekonomi maupun lingkungan.

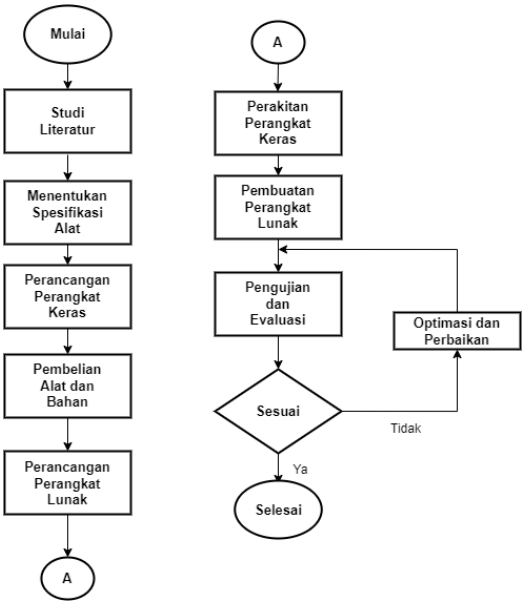
Beberapa penelitian telah dilakukan untuk menemukan logika kontrol yang tepat untuk diaplikasikan pada *lift group* ini. Salah satunya adalah penelitian yang dilakukan oleh [1] dengan judul "Renovasi Sistem Pengendalian Miniatur *Lift* Group Berbasis PLC Di Laboratorium Instalasi". Pada penelitian tersebut dirancang miniatur *lift group* yang terdiri dari dua *lift*, yaitu *lift* A dan *lift* B dengan lima buah lantai. Panggilan akan dipenuhi berdasarkan yang searah dengan *lift* yang sedang beroperasi. Terdapat juga penelitian untuk mengontrol pergerakan *lift group* menggunakan metode *fuzzy* [2]. Pada penelitian ini dirancang kontrol pergerkan *lift group* yang mengatur dua *lift* dengan tujuh lantai. Algoritma kontrol ini diaplikasikan pada CX-PROGRAMMER. Hasil pengujian menunjukkan bahwa terjadi peningkatan pelayanan pada *lift* dengan berkurangnya rata-rata waktu tunggu yang dihabiskan oleh pengguna. Sementara itu, penelitian yang dilakukan oleh [3] dengan judul *Group control lift dispatching system based on S7-1200 PLC* melakukan perbandingan antara algoritma *partition dispatching*, algoritma *lift group control* umum, dan pengujian *lift* tradisional. Hasil pengujian membuktikan bahwa *lift group control* umum mengangkut lebih banyak penumpang dibandingkan dengan *lift* tradisional. *Lift group control* dengan menggunakan algoritma *partition dispatching* memiliki kapasitas lebih besar dan dapat meningkatkan efisiensi daripada algoritma *lift group control* yang normal. Namun, algoritma *partition dispatching* ini cocok digunakan pada *lift group* dengan jumlah *lift* lebih dari lima. Sementara itu, penelitian yang dilakukan oleh [4] merancang algoritma pengiriman *lift* untuk memprioritaskan perjalanan yang searah. Begitupun [5] juga melakukan penelitian yang sama, namun menambhkan *input* berupa jumlah sekelompok orang yang menunggu *lift* yang dipantau menggunakan kamera digital. Dari penelitian yang dilakukan, terbukti bahwa algoritma yang diusulkan dapat menghemat energi hingga 20%. Namun, untuk mengimplementasikan teori ini diperlukan pengetahuan yang kuat mengenaij *Bayesian Network.*

Selain bagaimana alogoritma yang diterapkan, terdapat aspek-aspek lain yang perlu diperhatikan dalam merancang *lift group.* Menurut [6] beberapa sistem yang dibutuhkan untuk membangun pengendalian pada *lift group* adalah *control host system, communication system, signal acquisition system,* dan *electrical control system.* [7] melakukan penelitian dengan mengatur kecepatan motor induksi 3 fasa menggunakan *inverter.* Hasil pengujian menunjukkan bahwa tegangan, arus, daya, faktor daya dan kecepatan motor induksi tiga fasa berbanding lurus dengan frekuensi operasi motor induksi tiga fasa. Hal ini sependapat dengan [8] dimana motor induksi dapat diatur dengan mengubah frekuensi pada *inverter* dengan cara merubah tegangan input pada inverter dengan menggeser tahanan pada potensio meter yang dihubungkan pada *inverter*. Alasan dipilihnya Raspberry Pi Pico sebagai mikrokontroler yang digunakan pada penelitian ini dikarenakan memiliki akses memori yang lebih tinggi daripada ESP32. Dimana akses memori tinggi diperlukan untuk aplikasi yang memiliki respon cepat seperti pada control *lift group* ini [9]. Hal ini sependapat dengan penelitian yang dilakukan oleh [10], bahwa Arduino memiliki stabilitas yang baik untuk pengembangan yang membutuhkan perhitungan kecepatan rendah. Sementara Raspberry Pi Pico, baik digunakan untuk pengembangan yang melibatkan perhitungan kumulatif pada tingkat kecepatan yang tinggi.

Berdasarkan penjelasan di atas, maka dibuatlah kendali *lift group* yang dibuat menggunakan metode *car nearest* untuk menentukan *lift* mana yang akan diperintahkan untuk memenuhi panggilan. Dengan begitu, diharapkan dengan adanya algoritma kontrol ini dapat meningkatkan efisiensi dan mempersingkat waktu tunggu yang dihabiskan oleh pengguna.

2. METODE PENELITIAN

Metode penelitian yang dilakukan dijelaskan pada *flowchart* di bawah ini



Gambar 1. Diagram Alir Pelaksanaan

Setiap tahapan dilakukan secara *sequencial* dimulai dari studi literatur mengenai topik yang dibahas baik dari segi penunjang teori maupun penelitian-penelitian yang telah dilakukan sebelumnya. Hasil dari studi literatur kemudian ditentukan spesifikasi dari sistem yang akan dibuat baik itu dari segi perangkat keras maupun perangkat lunak. Setelah itu, dilakukan perancangan perangkat keras yang terdiri dari pembuatan skematik rangkaian, desain miniatur; dan perancangan perangkat lunak yang terdiri dari pembuatan algoritma program.

Sementara itu, pembuatan perangkat keras terdiri dari pembuatan PCB, *wiring* komponen, dan pembuatan miniatur *lift group*. Pembuatan perangkat lunak terdiri dari pembuatan *script program* sesuai dengan algoritma yang telah ditentukan sebelumnya. Pembuatan *script program* dilakukan di aplikasi thonny dengan menggunakan bahasa micropython. Masing-masing dari perangkat keras maupun perangkat lunak kemudian diintegrasikan dan dirakit sehingga dapat menghasilkan kendali sesuai dengan yang diinginkan.

2.1 Metode *Car Nearest*

Algoritma pengendalian *Car Nearest* (NC) dikenal sebagai algoritma dasar dalam sistem kendali *lift group*. Posisi *lift*, arah panggilan pendaratan, dan arah *lift* adalah tiga faktor utama yang digunakan dalam algoritma ini. FS (*Figure of Suitability*) merupakan suatu nilai numerik yang digunakan untuk pemilihan *lift* berdasarkan permintaan pengguna. *Lift* yang memiliki nilai FS paling besar , maka akan dipilih untuk memenuhi panggilan yang ada [10]. Ketika ada lebih dari satu *lift* dengan nilai FS yang sama, *lift* terdekat akan ditugaskan untuk melayani penumpang. Nilai FS ditentukan dengan aturan sebagai berikut:

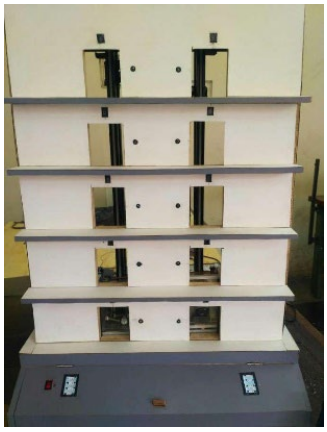
- Aturan 1: Jika *lift* bergerak menuju tempat pendaratan panggilan dan searah dengan panggilan pendaratan itu FS = (N+2)-d.
- Aturan 2: Jika *lift* bergerak menuju tempat pendaratan panggilan tetapi berlawanan arah dengan panggilan pendaratan FS = (N+1)-d.
- Aturan 3: Jika *lift* tidak bergerak (*IDLE*) maka FS = (N+1)-d.
- Aturan 4: Jika *lift* bergerak menjauhi landasan maka FS = 1.

Dimana : d adalah jarak antara titik pendaratan dan titik pendaratan posisi lift saat ini (dalam satuan lantai).

N adalah jumlah lantai di dalam gedung.

2.2 Konstruksi dan Spesifikasi Miniatur *Lift Group*

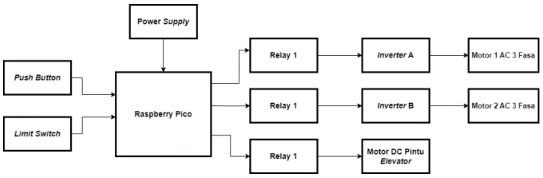
Pada sistem ini, rangka terbuat dari kayu konsul dan aluminium profile. Desain *lift group* ini berukuran 630 mm x 715 mm. Konstruksi *real* dari miniatur *lift group* dapat dilihat pada gambar di bawah ini:



Gambar 2. Konstruksi *Lift Group*

Input berupa tombol panggilan *lift* akan diproses oleh kontroler Raspberry Pico. Menggunakan algoritma kendali yang telah dirancang, kontroler akan menentukan dan memilih *lift* untuk memenuhi panggilan. Setelah itu, kontroler akan

memerintahkan *inverter* untuk menjalankan motor AC agar dapat menaik turunkan *lift cabin*. Kontroler juga akan memerintahkan motor DC untuk membuka dan menutup pintu *lift cabin* maupun pintu lantai ketika terdapat panggilan. Hal ini sesuai dengan blok diagram di bawah ini:



Gambar 3. Blok Diagram Sistem Kendali *Lift Group*

Sementara itu, spesifikasi alat dijelaskan pada tabel di bawah ini:

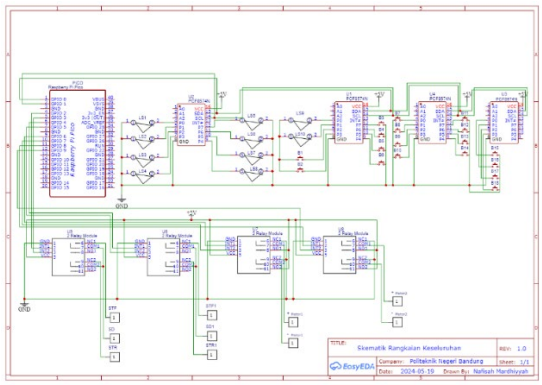
Tabel 1. Spesifikasi Sistem Kendali Lift Group

No	Spesifikasi	Keterangan
Konsul		
1	Utilitas Bangunan	*Building Automation System* (BAS)
2	Bentuk Miniatur	Persegi panjang
3	Ukuran Miniatur	Panjang 63cm x lebar 43cm x tinggi 84,5cm
4	Bahan Miniatur	Aluminium profile, besi siku, hollow, dan papan kayu
Perangkat Lunak		
5	*Operating System* PC	Windows 10
6	*Controller Software*	Thonny
7	*Communication Software*	USB Serial
Perangkat Keras		
8	Motor AC	Tegangan 380 V Arus Nominal 0,08 A Daya 15 W
9	*Variable Speed Drive*	Daya 400 kW Tegangan Masukan 380 V Tegangan Keluaran Beban 380 V
10	Raspberry Pico	Jumlah I/O 26 pin: I/O Digital 23 pin I/O Analog 3 pin
Panel Miniatur *Lift Group*		
11	*Input* Digital	32 *Input* Digital (tombol *start/stop,* 10 *limit switch,* 10 tombol pilihan, dan 10 tombol panggilan)
12	*Output* Digital	8 *Output* Digital (4 pin motor DC, 4 pin motor AC)

2.3 Perancangan dan Pembuatan Perangkat Keras

2.3.1 Perancangan Perangkat Keras

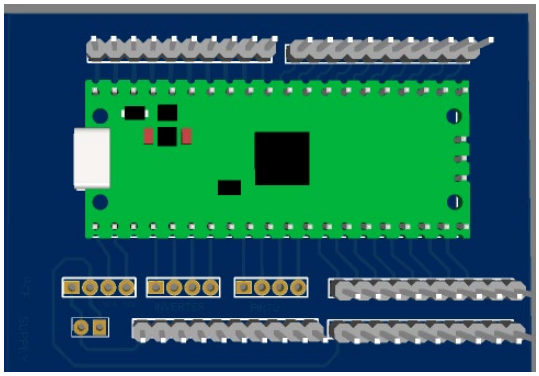
Skematik rangkaian ditunjukkan pada gambar di bawah ini, dimana *limit switch* dan *push button* dihubungkan ke *expansion module* PCF8574. Sementara untuk 2 relay yang terhubung ke motor DC dan 2 relay yang terhubung ke motor AC dihubungkan langsung ke *raspberry pico.*



Gambar 4. Diagram Pengkabelan Antar Komponen

2.3.2 Pembuatan Perangkat Keras

Setelah masing-masing perangkat berhasil dilakukan *wiring,* selanjutnya dilakukan pembuatan PCB untuk penempatan komponen yang digunakan. Pembuatan desain PCB dilakukan di aplikasi easyeda. PCB ini digunakan untuk penempatan raspberry pico sehingga lebih mudah untuk dilakukan *wiring* pada komponen lainnya.

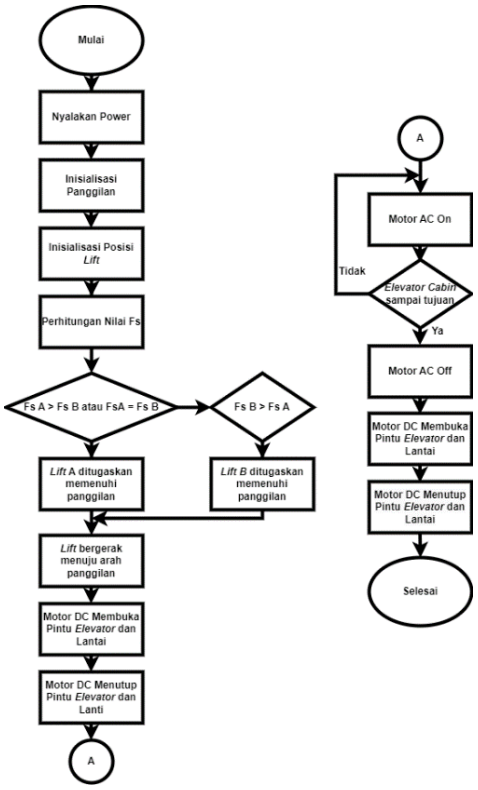


Gambar 5. Design PCB

2.4 Perancangan dan Pembuatan Perangkat Lunak

2.4.1 Perancangan Perangkat Lunak

Pada proses perancangan perangkat lunak, dibuat algoritma metode *car nearest* untuk pengendalian *lift group.* Pada pembuatan *lift group* ini terdiri dari beberapa komponen, yaitu motor AC 3 fasa sebagai penggerak utama, raspberry Pico sebagai kontroler, *limit switch* untuk sensor posisi, dan motor DC sebagai penggerak untuk buka tutup pintu lift cabin. Pada gambar di bawah ini merupakan diagram alir algoritma miniatur *lift group*.

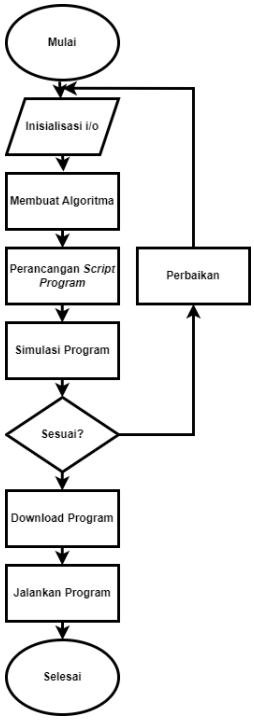


Gambar 6. Diagram Alir Metode *Car Nearest*

Berdasarkan diagram alir di atas, ketika *power* dinyalakan dan *lift* mendapatkan panggilan maka kontroler akan menentukan *lift* yang ditugaskan berdasarkan nilai Fs yang paling besar. Setelah itu, pintu lantai maupun *lift* akan terbuka dan penumpang pun masuk. *Lift cabin* akan digerakkan oleh motor AC sampai tiba di lantai tujuan. Setelah tiba di lantai tujuan, motor AC akan mati dan pintu lantai maupun pintu *lift* akan membuka kembali untuk mempersilahkan penumpang keluar. Setelah itu, pintu akan menutup Kembali dan proses pun selesai. Begitulah *lift* akan memiliki *sequencial* yang sama secara *looping* ketika terdapat panggilan.

2.4.2 Pembuatan Perangkat Lunak

Algoritma program yang telah ditentukan pada perancangan *software* diimplementasikan dengan pembuatan *script* program dengan menggunakan bahasa *microphyton* yang dibuat di aplikasi *thonny.* Pertama-tama dibuat terlebih dahulu *script program* antar masing-masing komponen, baru dilakukan integrasi yang memuat program keseluruhan, Gambar di bawah menjelaskan diagram alir dalam pembuatan *software.*



Gambar 7. Diagram Alir Pembuatan Perangkat Lunak

3. HASIL DAN PEMBAHASAN

3.1 Pengujian Kecepatan Respons

Pengujian kecepatan respon dilakukan untuk menguji berapa waktu yang diperlukan oleh sistem untuk menampilkan *lift* mana yang ditugaskan untuk memenuhi panggilan dari sejak mulai ditekan *push button* permintaan. Waktu diukur menggunakan *stopwatch*.

Tabel 2. Pengujian Kecepatan Respon

Input	**Tampilan LCD**	**Waktu Respon**
Permintaan turun lantai 5		82 ms
Permintaan naik lantai 4		63 ms
Permintaan turun lantai 4		65 ms
Permintaan naik lantai 3		63 ms
Permintaan turun lantai 3		64 ms
Permintaan naik lantai 2		81 ms
Permintaan turun lantai 2		60 ms
Permintaan naik lantai 1		59 ms

Dari pengujian di atas, didapatkan bahwa rata-rata waktu yang dibutuhkan oleh sistem untuk menentukan *lift* yang ditugaskan adalah 67,125 ms.

3.2 Pengujian Algoritma Kendali *Lift Group*

Untuk menguji algoritma kendali *lift group,* diasumsikan *lift* pada posisi tertentu dan kemudian dihitung nilai Fs dari masing-masing *lift* (*lift* A dan *lift* B) sehingga dapat diketahui *lift* mana yang akan memenuhi panggilan. Pengujian pertama diasumsikan terdapat permintaan turun dari masing-masing lantai, kecuali lantai 1 dengan posisi *lift* sebagai berikut:

Tabel 3. Posisi Lift Pertama

Lantai	*Lift* A	*Lift* B
5		
4		
3		
2		
1		

Berdasarkan posisi *lift* sebagaimana tabel di atas, maka dapat dihitung nilai Fs sebagai berikut:

Tabel 4. Perhitungan Fs Posisi Pertama Permintaan Turun

Lantai	Perhitungan Fs *Lift* A	Perhitungan Fs *Lift* B	Hasil Perhitungan	Kondisi Real
5	Fs= n+1-d =5+1-4= 2	Fs= n+2-d = 5+2-0=7	Lift B dipilih	Button pressed: Down on floor 5 Nilai fs_A adalah 2 Nilai fs_B adalah 7 Floor 5 assigned to Lift B
4	Fs = n+1-d =5+1-3= 3	Fs= n+2-d = 5+2-1=6	Lift B dipilih	Button pressed: Down on floor 4 Nilai fs_A adalah 3 Nilai fs_B adalah 6 Floor 4 assigned to Lift B
3	Fs= n+1-d = 5+1-2=4	Fs= n+2-d = 5+2-2=5	Lift B dipilih	Button pressed: Down on floor 3 Nilai fs_A adalah 4 Nilai fs_B adalah 5 Floor 3 assigned to Lift B

Lantai	Perhitungan Fs *Lift* A	Perhitungan Fs *Lift* B	Hasil Perhitungan	Kondisi Real
2	Fs= n+1-d = 5+1-1=5	Fs= n+2-d = 5+2-3=4	Lift A dipilih	Button pressed: Down on floor 2 Nilai fs_A adalah 5 Nilai fs_B adalah 4 Floor 2 assigned to Lift A
1				

Pengujian selanjutnya, posisi lift masih sama namun diasumsikan permintaan naik dari masing-masing lantai kecuali lantai 5. Maka didapatkan perhitungan Fs sebagai berikut:

Tabel 5. Perhitungan Fs Posisi Pertama Permintaan Naik

Lantai	Perhitungan Fs *Lift* A	Perhitungan Fs *Lift* B	Hasil Perhitungan	Kondisi Real
5				
4	Fs= n+2-d= 5+2-3=4	Fs= n+1-d = 5+1-1=5	Lift B dipilih	Button pressed: Up on floor 4 Nilai fs_A adalah 4 Nilai fs_B adalah 5 Floor 4 assigned to Lift B
3	Fs= n+2-d = 5+2-2=5	Fs= n+1-d = 5+1-2=4	Lift A dipilih	Button pressed: Up on floor 3 Nilai fs_A adalah 5 Nilai fs_B adalah 4 Floor 3 assigned to Lift A
2	Fs= n+2-d = 5+2-1=6	Fs = n+1-d = 5+1-3= 3	Lift A dipilih	Button pressed: Up on floor 2 Nilai fs_A adalah 6 Nilai fs_B adalah 3 Floor 2 assigned to Lift A
1	Fs= n+2-d = 5+2-0=7	Fs= n+1-d = 5+1-4= 2	Lift A dipilih	Button pressed: Up on floor 1 Nilai fs_A adalah 7 Nilai fs_B adalah 2 Floor 1 assigned to Lift A

4. KESIMPULAN

Secara keseluruhan, sistem kendali *lift group* telah berhasil dibuat dan dapat disimpulkan bahwa:

1. Kecepatan respon *lift* untuk menjawab panggilan yang ada adalah tidak lebih dari 1s dengan rata-rata kecepatan respon 67,125 ms.
2. Kesesuaian *lift* dalam memenuhi panggilan telah berhasil dilakukan dibuktikan dengan adanya kesamaan antara hasil perhitungan Fs dengan respon yang sebenarnya terjadi.

UCAPAN TERIMA KASIH

Terima kasih kepada Politeknik Negeri Bandung melalui Wakil Direktur Akademik yang telah memberikan bantuan pendanaan TA dengan kelompok pembiayaan AI.

DAFTAR PUSTAKA

[1] G. Yogi, “Renovasi Sistem Pengendalian Miniatur Lift Group Berbasis PLC Di Laboratorium Instalasi”.

[2] M. Primadiansya, “PENGONTROL PERGERAKAN GROUP LIFT MENGGUNAKAN LOGIKA FUZZY BERBASIS PROGRAMMABLE LOGIC CONTROL (PLC)”.

[3] H. Huan, H. Tian, Z. Yan, dan Z. Yao, “Group control elevator dispatching system based on S7-1200 PLC,” dalam *2019 Chinese Automation Congress (CAC)*, Hangzhou, China: IEEE, Nov 2019, hlm. 3853–3857. doi: 10.1109/CAC48633.2019.8996971.

[4] A. Sudaesi, G. W. Wiriasto, dan J. Majapahit, “Rancang Bangun Simulator Pengendalian Lift 6 Lantai Berbasis PLC,” *Jurnal Teknologi Informasi, Komputer dan Aplikasinya (JTIKA)*, vol. 4, no. 1, hlm. 97–106, 2022.

[5] Y. Bapin dan V. Zarikas, “Smart Building’s Elevator with Intelligent Control Algorithm based on Bayesian Networks,” *ijacsa*, vol. 10, no. 2, 2019, doi: 10.14569/IJACSA.2019.0100203.

[6] X. Zhang dan Y. Shang, “Design and Research of Elevator Group Control System Based on PLC,” *J. Phys.: Conf. Ser.*, vol. 1646, no. 1, hlm. 012116, Sep 2020, doi: 10.1088/1742-6596/1646/1/012116.

[7] A. A. G. Basyar, S. Handoko, dan I. Setiawan, “Implementasi Inverter Alvitar 12 Dan Toshiba VF15 Sebagai Pengendalian Kecepatan Pada Motor Induksi 3 Fasa Untuk Aplikasi Sistem Konveyor Terkendali,” *Transient*, vol. 10, no. 1, hlm. 184–189, Mar 2021, doi: 10.14710/transient.v10i1.184-189.

[8] E. S. Nasution dan A. Hasibuan, “Pengaturan Kecepatan Motor Induksi 3 Phasa Dengan Merubah Frekuensi Menggunakan Inverter ALTIVAR 12P,” *SISFO*, vol. 2, no. 1, hlm. 25–34, Nov 2018, doi: 10.29103/sisfo.v2i1.1001.

[9] A. H. Lubis dan S. Aryza, “Penentuan Microcontroller Unit (MCU) Terbaik berdasarkan Pembobotan Objektif,” *SNASTIKOM*, 2022.

[10] M. Thothadri, “An Analysis on Clock Speeds in Raspberry Pi Pico and Arduino Uno Microcontrollers,” *AJETM*, vol. 6, no. 3, hlm. 41, 2021, doi: 10.11648/j.ajetm.20210603.13.