Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK) DOI: 10.25126/jtiik.202295684
Vol. 9, No. 2, April 2022, hlm. 435-442 p-ISSN: 2355-7699
Akreditasi KEMENRISTEKDIKTI, No. 36/E/KPT/2019 e-ISSN: 2528-6579

PENERAPAN MEKANISME KOMUNIKASI MULTI-HOP PADA BLUETOOTH LOW ENERGY

Agung Setia Budi[*1], Muhammad Hanif Azhar Efendi[2], Adhitya Bhawiyuga[3], Mochammad Hannats Hanafi Ichsan[4]

[1,2,3,4]Universitas Brawijaya, Malang
Email: [1]agungsetiabudi@ub.ac.id, [2]hanifazharef@student.ub.ac.id, [3]bhawiyuga@ub.ac.id, [4]hanas.hanafi@ub.ac.id
[*]Penulis Korespondensi

(Naskah masuk: 15 Oktober 2021, diterima untuk diterbitkan: 21 Februari 2022)

Abstrak

Seiring dengan berjalannya waktu, perkembangan teknologi akan semakin pesat, begitu pun teknologi dalam bidang *wireless communication* atau komunikasi tanpa kabel. Salah satu teknologi *wireless* yang sudah ada cukup lama adalah *Bluetooth*, adapun pengembangan dari teknologi *Bluetooth* adalah *Bluetooth Low Energy* (BLE). Tujuan dikembangkannya BLE adalah agar sumber daya yang digunakan saat proses transmisi data lebih rendah dari *Bluetooth* yang sudah ada sebelumnya. Adapun terdapat beberapa batasan yang ada pada *Bluetooth Low Energy* (BLE), salah satunya adalah hanya dapat berkomunikasi satu dengan yang lainnya dalam jarak yang dekat dikarenakan adanya keterbatasan sumber daya. Berdasarkan dari permasalahan tersebut, pada penelitian ini dirancang mekanisme komunikasi *multi-hop* pada BLE untuk mengatasi permasalahan jarak yang terbatas. Pada penelitian ini, topologi yang digunakan adalah topologi *tree*, dan *hardware* yang digunakan adalah ESP32. Hasil yang didapat menunjukkan mekanisme komunikasi *multi-hop* pada BLE yang dirancang dapat diterapkan dengan sukses. Hasil pengujian menunjukkan bahwa pengiriman data dari sebuah *Sensor Node* ke *Sink Node* yang melewati dua *Relay Node* (3-hop) membutuhkan waktu rata-rata 1846,4 *ms*.

Kata kunci: *Bluetooth Low Energy* (BLE), *Multi-hop*, *wireless*, Komunikasi, *ESP32*

THE IMPLEMENTATION OF MULTI-HOP COMMUNICATION MECHANISM ON BLUETOOTH LOW ENERGY

Abstract

Technology has been being developed very fast in the last couple years. One of that technology is wireless communication. There are so many wirelees communication technologies arround us nowadays, such as Wi-Fi, LoRa, Zigbee,and Bluetooth. The recent update of Bluetooth technology is Bluetooth Low Energy (BLE). The purpose of developing BLE is to decrease the energy used during the data transmission process. But there are some limitations in BLE. One of them is that we can only communicate with each other in a short distance due to resources limitation. Based on these problem, in this research we want to implement the multi-hop communication mechanism on BLE to overcome short distance communication problem. In this research, we use tree topology and ESP32 as the hardware. The results of this research shows that the mechanism of multi-hop communication on BLE can be applied successfully. The experiment result shows that the transmissionof data from a Sensor Node to a Sink Node through two Relay Nodes (3-hops) needs the average time of 1846,4 ms.

Keywords: *Bluetooth Low Energy* (BLE), *Multi-hop*, *Wireless*, *Communication*, *ESP32*

1. PENDAHULUAN

Seiring dengan berjalannya waktu, perkembangan teknologi akan semakin pesat, begitu pun teknologi dalam bidang *wireless* atau komunikasi tanpa kabel. Salah satu dari teknologi *wireless* yang sudah ada cukup lama adalah *Bluetooth*, adapun pengembangan dari teknologi *Bluetooth* adalah *Bluetooth Low Energy*. Teknologi *Bluetooth Low Energy* (BLE) awal mulanya diberi nama "Wibree", kemudian seiring berjalannya waktu pada saat proses pengembangan, akhirnya pada 2010 teknologi tersebut dimasukkan ke dalam standar utama *Bluetooth* sebagai fitur pengembangan *Bluetooth* lebih lanjut ketika *Bluetooth Core*

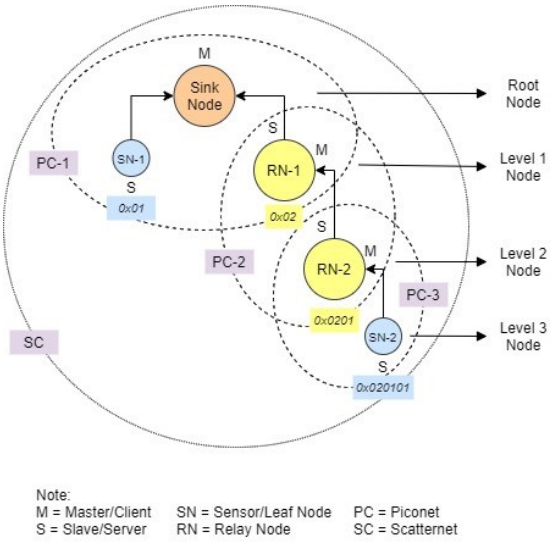
Specification versi 4.0 diumumkan (Zonglin Guo et al., 2015). Tujuan dikembangkannya BLE adalah agar sumber daya yang digunakan saat proses transmisi data lebih rendah dari *Bluetooth* yang sudah ada sebelumnya. Proses penghematan sumber daya yang dilakukan oleh BLE adalah dengan cara terus menerus berada di mode *sleep* dan akan terbangun atau bekerja ketika ada sebuah inisiasi komunikasi (Arief Sukma I. et al., 2018). Transmisi data yang terjadi pada BLE juga dilakukan secara singkat, tidak secara terus menerus mengirimkan data yang akhirnya dapat menyebabkan sumber daya cepat habis (Zonglin Guo et al., 2015). Seiring dengan berkembangnya teknologi, teknologi BLE akhirnya banyak diterapkan dalam berbagai macam bidang, diantarnya adalah dalam bidang olahraga, kesehatan, perangkat cerdas dan lain sebagainya.

Adapun terdapat beberapa batasan yang ada pada BLE, salah satunya adalah hanya dapat berkomunikasi satu dengan yang lainnya dalam jarak yang dekat dikarenakan adanya keterbatasan sumber daya. Maka dari itu diperlukan adanya sebuah cara agar BLE dapat saling berkomunikasi dalam jarak yang cukup jauh ketika berada dalam sebuah jaringan yang cukup luas. Salah satunya adalah dengan cara *multi-hop*, yang artinya meneruskan data yang dikirim melalui beberapa BLE yang ada dalam sebuah jaringan hingga sampai ke tujuan, sehingga kendala jarak yang dekat dapat diatasi. Beberapa penelitian mengenai *multi-hop* pada BLE pernah dilakukan sebelumnya, diantaranya adalah penelitian yang dilakukan oleh Joonkyo mengenai metode penerapan *multi-hop* pada BLE untuk keperluan industri *Internet of Things* (IoT) dengan topologi *tree* (Joonkyo Kim et al., 2015). Selanjutnya dari peneliti Luca yang melakukan penelitian mengenai penerapan *komunikasi multi-hop* pada BLE secara *real time* didalam sebuah jaringan dengan topologi *mesh* (Luca Leonardi et al., 2018). Penelitian yang lainnya juga pernah dilakukan oleh Zonglin yang membahas mengenai *komunikasi multi-hop* pada BLE berbasis *Wireless Sensor Network* yang bekerja secara *on demand* (Zonglin Guo et al., 2015). Kemudian peneliti Manato juga pernah melakukan penelitian mengenai *komunikasi multi-hop* pada BLE yang berbasis *smartphone* untuk mendeteksi longsor (Manato Fujimoto et al., 2016).

Pada beberapa penelitian tersebut kebanyakan penerapan *komunikasi multi-hop* pada BLE dilakukan hanya sebatas metode saja dan belum diterapkan pada sebuah perangkat keras. Oleh karena itu, pada penelitian ini dilakukan penerapan mekanisme komunikasi *multi-hop* pada BLE dengan menggunakan ESP32 sebagai perangkat keras. Kontribusi dari penelitian ini adalah menghasilkan mekanisme komunikasi *multi-hop* pada BLE yang secara spesifik dapat diaplikasikan di ESP32 yang secara default tidak memiliki mekanisme *multi-hop.*

2. METODE KOMUNIKASI *MULTI-HOP* PADA BLE

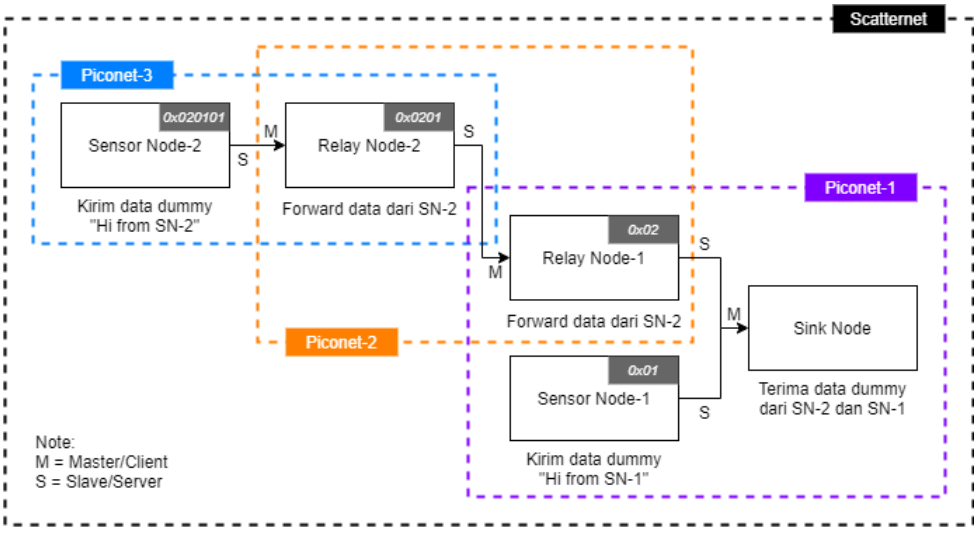
Dalam penelitian ini, topologi yang digunakan ditunjukkan dalam Gambar 1. Topologi ini dipilih karena proses *forwarding* data dalam topologi ini dapat dilihat lebih jelas dan intuitif. Dengan demikian pengembangan metode dan penjelasan mekanisme komunikasi *multi-hop* pada BLE bisa lebih mudah.



Gambar 1. Perancangan Keseluruhan Sistem

Dalam Gambar 1, seluruh *node* yang ada dalam jaringan topologi *tree* dinamakan *Scatternet.* Jaringan ini akan dipecah menjadi beberapa bagian kecil yang bernama *Piconet*. Untuk proses perancangan komunikasi *multi-hop*, awal mulanya akan dilakukan pengalamatan pada setiap *node*, kemudian karena proses komunikasi yang dilakukan adalah satu arah dari *level node* terendah ke *level node* tertinggi maka setiap *node* akan diberi alamat kecuali *sink node* yang berada di *level node* tertinggi. Pemberian alamat yang berupa ID dilakukan untuk memudahkan dalam mengetahui asal data yang dikirimkan. Kemudian pada tiap ujung *sensor* atau *leaf node* data akan pertama kali dikirimkan dan diteruskan menuju *sink node* secara *multi-hop* melalui beberapa *relay node*.

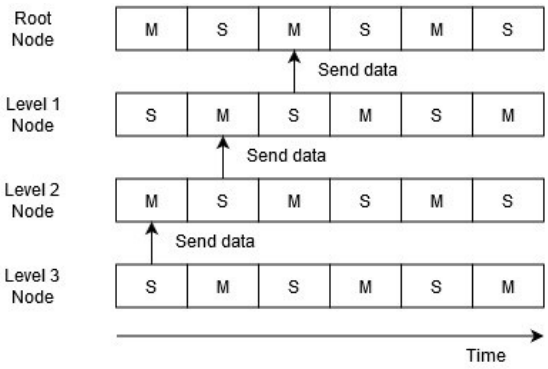
Selanjutnya untuk detail proses perancangan kerja sistemnya dapat dilihat Gambar 2. Tahap pertama akan dimulai dari *level node* yang paling bawah yaitu *level* 3, yakni SN-2 akan mengirmkan data ke RN-2. Selanjutnya tahap kedua, RN-2 akan meneruskan data yang telah diterima dari SN-2 ke RN-1. Tahap ketiga, SN-1 akan mengirimkan data ke *sink node*, setelah menerima data dari SN-1 maka *sink node* akan menerima data dari RN-1 yang berisi data SN-2. Sehingga, nantinya data dari masing-masing *sensor node* yakni SN-1 dan SN-2 akan sama-sama diterima oleh *sink node* yang dikirimkan secara *multi-hop* melalui beberapa *relay node*.



Gambar 2. Diagram proses pengiriman data dari *Sensor Node* ke *Sink Node*

Pada proses meneruskan data, nantinya *relay node* akan melakukan proses perpindahan *role* sesuai dengan kondisi saat proses pertukaran data berjalan dalam masing-masing area *Piconet*. Untuk ilustrasi perpindahan *role* yang dimaksud dapat dilihat dalam Gambar 3.

Berdasarkan Gambar 3 dapat disimpulkan bahwa ketika dalam area *Piconet-3*, *role* dari RN-2 adalah *master* atau *client* ketika menerima data dari SN-2 yang memiliki *role* sebagai *slave* atau *server*. Kemudian dalam area *Piconet-2*, *role* dari RN-2 yang sebelumnya adalah *client* akan berganti menjadi *server* untuk mengirimkan data ke RN-1 yang memiliki *role* sebagai *client*Dan terkahir pada area *Piconet-1*, *role* dari RN-1 yang sebelumnya adalah *client* akan berganti menjadi *server* untuk mengirimkan data ke *sink node* yang memiliki *role* sebagai *client*.

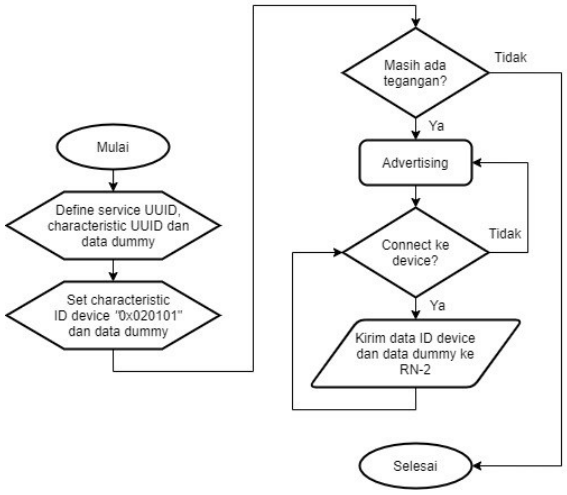


Gambar 3. Perpindahan *Role* yang Dilakukan oleh *Relay Node*

2.1 Pertukaran Data pada *Piconet-3*

Dalam area *Piconet-3* terdapat dua *node* yaitu SN-2 dan RN-2. Pada tahap perancangan area *Piconet-3* akan dibagi menjadi dua bagian, yaitu bagian pengiriman dan penerimaan data. Perancangan pengiriman data yang akan dilakukan oleh SN-2 ke RN-2 dapat dilihat dalam Gambar 5.

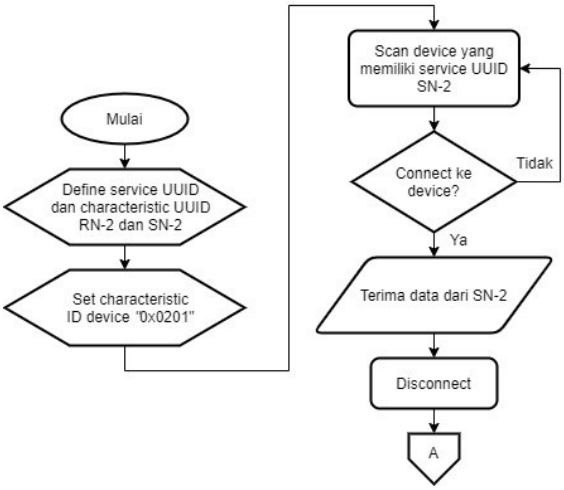
Dalam Gambar 4 dapat dilihat bahwa tahap pertama dari proses pengiriman data oleh SN-2 ke RN-2 adalah dengan inisialisasi *service* UUID, *characteristic* UUID dan data *dummy*. Selanjutnya memberikan nilai pada masing-masing *characteristic* untuk ID *device* yang bernilai "0x020101" dan data *dummy* yang nantinya akan dikirimkan ke RN-2. Setelah tahap persiapan sudah selesai, maka SN-2 akan melakukan *advertising* karena memiliki *role* sebagai *server* untuk mengirimkan data. Kemudian SN-2 akan selalu mengecek apakah telah terhubung ke *device* atau belum, dan ketika terhubung dengan RN-2, maka proses pengiriman data dapat berjalan. Data yang dikirimkan adalah ID *device* SN-2 dan data *dummy*.



Gambar 4. *Flowchart* Pengiriman Data SN-2 ke RN-2

Untuk proses penerimaan data yang akan dilakukan oleh RN-2 dari SN-2 dapat dilihat dalam Gambar 5. Dari gambar ini dapat dilihat bahwa tahap pertama dari proses penerimaan data oleh RN-2 dari SN-2 adalah dengan inisialisasi *service* UUID dan *characteristic* UUID RN-2 dan SN-2. Selanjutnya memberikan nilai *characteristic* untuk

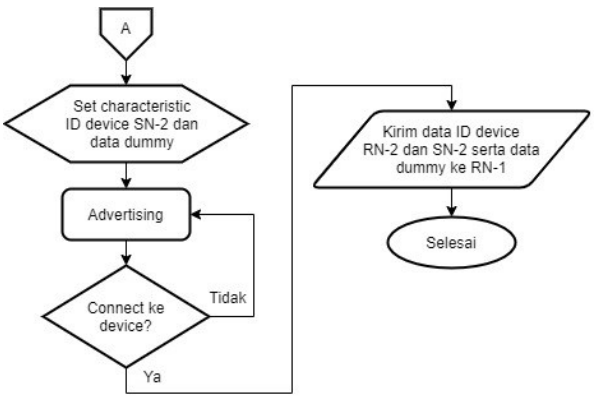
ID *device* yang bernilai "0x0201". Setelah tahap persiapan sudah selesai, maka RN-2 akan melakukan *scanning* karena memiliki *role* sebagai *client* untuk menerima data. Proses *scanning* akan mengecek apakah *device* yang sedang melakukan *advertising* memiliki *service* UUID yang sama dengan SN-2. Jika terdapat *service* UUID yang sama dengan SN-2, maka RN-2 akan mencoba untuk terhubung dengan SN-2. Kemudian ketika RN-2 sudah terhubung dengan SN-2, maka proses penerimaan data dapat berjalan. Data yang diterima adalah ID *device* SN-2 dan data *dummy*. Setelah data diterima, maka RN-2 akan mencoba untuk berhenti terhubung dengan SN-2. Proses pengiriman data kemudian akan berlanjut ke bagian Piconet-2. Oleh karena itu *flowchart* dalam Gambar 5 tidak diakhiri dengan "selesai".



Gambar 5. *Flowchart* Penerimaan Data RN-2 dari SN-2

2.2 Pertukaran Data pada *Piconet-2*

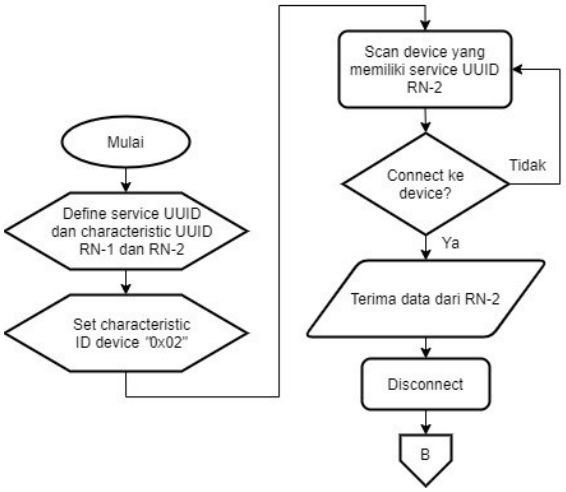
Dalam area *Piconet-2* terdapat dua *node* yaitu RN-2 dan RN-1. Pada tahap perancangan area *Piconet-2* akan dibagi menjadi dua bagian, yaitu bagian pengiriman dan penerimaan data. Perancangan pengiriman data yang akan dilakukan oleh RN-2 ke RN-1 dapat dilihat pada Gambar 6.



Gambar 6. *Flowchart* Pengiriman Data RN-2 ke RN-1

Dari Gambar 6 diatas, ketika proses penerimaan data sudah selesai dilakukan oleh RN-2 dari SN-2, maka proses selanjutnya adalah meneruskan data yang telah diterima dari SN-2 menuju RN-1. Tahap pertama ketika RN-2 sudah berhenti terhubung dengan SN-2 adalah memberikan nilai pada masing-masing *characteristic* untuk ID *device* SN-2 dan data *dummy* yang telah diterima dari SN-2 sebelumnya dan nantinya akan dikirimkan ke RN-1. Setelah tahap persiapan sudah selesai, maka RN-2 yang sebelumnya memiliki *role client* akan berganti menjadi *role server* dan akan melakukan *advertising* untuk mengirimkan data. Kemudian RN-2 akan selalu mengecek apakah telah terhubung ke *device* atau belum, dan ketika terhubung dengan RN-1, maka proses pengiriman data dapat berjalan. Data yang dikirimkan adalah ID *device* RN-2 dan SN-2 serta data *dummy* yang sebelumnya telah diterima dari SN-2.

Untuk proses perancangan penerimaan data yang akan dilakukan oleh RN-1 dari RN-2 dapat dilihat pada Gambar 7.



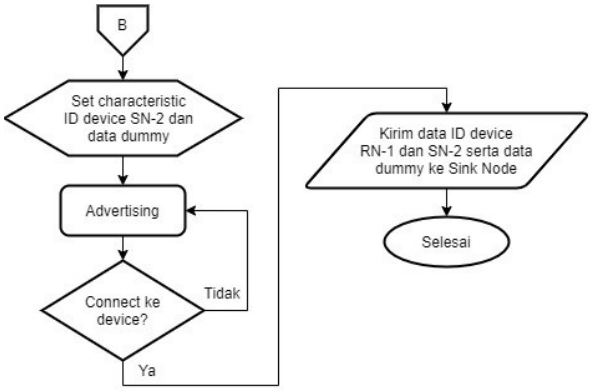
Gambar 7. *Flowchart* Penerimaan Data RN-1 dari RN-2

Dari Gambar 8 diatas, dapat dilihat bahwa tahap pertama dari proses penerimaan data oleh RN-1 dari RN-2 adalah dengan inisialisasi *service* UUID dan *characteristic* UUID RN-1 dan RN-2. Selanjutnya memberikan nilai *characteristic* untuk ID *device* yang bernilai "0x02". Setelah tahap persiapan sudah selesai, maka RN-1 akan melakukan *scanning* karena memiliki *role* sebagai *client* untuk menerima data. Proses *scanning* akan mengecek apakah *device* yang sedang melakukan *advertising* memiliki *service* UUID yang sama dengan RN-2. Jika terdapat *service* UUID yang sama dengan RN-2, maka RN-1 akan mencoba untuk terhubung dengan RN-2. Kemudian ketika RN-1 sudah terhubung dengan RN-2, maka proses penerimaan data dapat berjalan. Data yang diterima adalah ID *device* RN-2 dan SN-2 serta data *dummy* yang sebelumnya telah dikirimkan dari SN-2 dan diteruskan oleh RN-2.

Setelah data diterima, maka RN-1 akan mencoba untuk berhenti terhubung dengan RN-2.

2.3 Pertukaran Data pada *Piconet-1*

Dalam area *Piconet-1* terdapat tiga *node* yaitu *Sink Node*, SN-1 dan RN-1. Pada jaringan ini bagian yang paling penting adalah pengiriman data dariRN-1 ke *Sink Node* yang merupakan kelanjutan dari proses pengiriman data sebelumnya. Alur pengiriman data dalam jaringan ini ditunjukkan dalam Gambar 8.

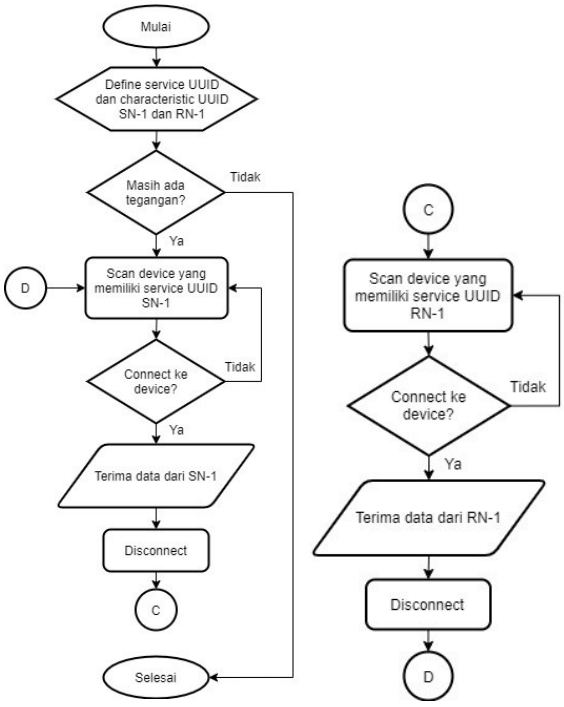


Gambar 8. *Flowchart* Pengiriman Data RN-1 ke *Sink Node*

Dari Gambar 10 diatas, ketika proses penerimaan data sudah selesai dilakukan oleh RN-1 dari RN-2, maka proses selanjutnya adalah meneruskan data yang telah diterima dari RN-2 menuju *Sink Node*. Tahap pertama ketika RN-1 sudah berhenti terhubung dengan RN-2 adalah memberikan nilai pada masing-masing *characteristic* untuk ID *device* SN-2 dan data *dummy* yang telah diterima dari RN-2 sebelumnya dan nantinya akan dikirimkan ke *Sink Node*. Setelah tahap persiapan sudah selesai, maka RN-1 yang sebelumnya memiliki *role client* akan berganti menjadi *role server* dan akan melakukan *advertising* untuk mengirimkan data. Kemudian RN-1 akan selalu mengecek apakah telah terhubung ke *device* atau belum, dan ketika terhubung dengan *Sink Node*, maka proses pengiriman data dapat berjalan. Data yang dikirimkan adalah ID *device* RN-1 dan SN-2 serta data *dummy* yang sebelumnya telah diterima dari RN-2.

Untuk proses penerimaan data yang akan dilakukan oleh *Sink Node* dari SN-1 dan RN-1 dapat dilihat pada Gambar 9. Dari gambar ini dapat dilihat bahwa tahap pertama dari proses penerimaan data oleh *Sink Node* dari SN-1 dan RN-1 adalah dengan inisialisasi *service* UUID dan *characteristic* UUID SN-1 dan RN-1. Karena ini merupakan *Sink Node* yang dimana data akan terakhir diterima maka tidak perlu memberikan ID *device*. Setelah tahap persiapan sudah selesai, maka *Sink Node* akan melakukan *scanning* karena memiliki *role* sebagai *client* untuk menerima data. Proses *scanning* yang pertama akan mengecek apakah *device* yang sedang melakukan *advertising* memiliki *service* UUID yang sama dengan SN-1. Jika terdapat *service* UUID yang sama dengan SN-1, maka *Sink Node* akan mencoba untuk terhubung dengan SN-1. Kemudian ketika *Sink Node* sudah terhubung dengan SN-1, maka proses penerimaan data dapat berjalan. Data yang diterima adalah ID *device* SN-1 serta data *dummy*. Setelah data diterima, maka *Sink Node* akan mencoba untuk berhenti terhubung dengan SN-1.

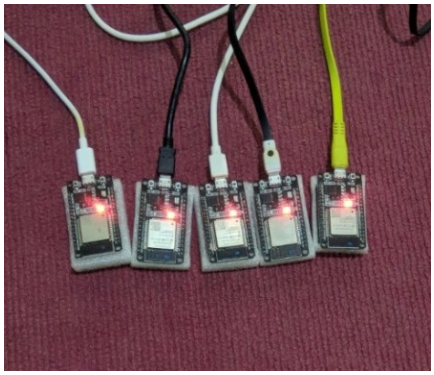
Ketika sudah berhenti terhubung dengan SN-1, maka *Sink Node* akan melakukan proses *scanning* yang kedua dengan mengecek apakah *device* yang sedang melakukan *advertising* memiliki *service* UUID yang sama dengan RN-1. Jika terdapat *service* UUID yang sama dengan RN-1, maka *Sink Node* akan mencoba untuk terhubung dengan RN-1. Kemudian ketika *Sink Node* sudah terhubung dengan RN-1, maka proses penerimaan data dapat berjalan. Data yang diterima adalah ID *device* RN-1 dan SN-2 serta data *dummy* yang sebelumnya telah dikirimkan dari SN-2 dan diteruskan oleh RN-2 dan RN-1. Setelah data diterima, maka *Sink Node* akan mencoba untuk berhenti terhubung dengan RN-1.



Gambar 9. *Flowchart* Penerimaan Data *Sink Node* dari SN-1 dan RN-1

2.4 Implementasi *Node*

Setiap *node* yang ada di pada diagram blok dalam Gambar 2 diwakili oleh sebuah ESP32. Sehingga keseluruhan jaringan terdiri dari komponen ESP32 sebanyak 5 buah yang langsung terhubung ke laptop sebagai sumber tegangannya. Untuk foto hasil dari proses implementasinya dapat dilihat dalam Gambar 10.



Gambar 10. Implementasi Arsitektur Setiap *Node*

3. HASIL PENGUJIAN

Pengujian pada sistem dilakukan untuk mengetahui kinerja secara fungsional dan non-fungsional. Pengujian fungsional dilakukan untuk mengetahui bahwa sistem telah berjalan sesuai dengan kebutuhan fungsional yang diharapkan, yaitu *Sensor Node* dapat mengirim data ke *Sink Node* melalui dua *Relay Node.* Hasil dari 10 kali pengujian ini didapatkan 100% keberhasilan *Sensor Node* dapat mengirim data ke *Sink Node* melalui dua *Relay Node*.

Pengujian non-fungsional dilakukan untuk mengetahui kinerja dari sistem yang sudah menerapkan protokol *komunikasi multi-hop* dengan menggunakan parameter pengujian *waktu pengiriman.* Waktu ini dihitung mulai dari data dikirim oleh *Sensor Node* hingga data diterima oleh *Sink Node.* Data hasil pengujian ditunjukkan dalam Tabel 1. Berdasarkan data dalam Tabel 1, maka dapat diketahui bahwa waktu yang dibutuhkan untuk pengiriman data satu hop kurang dari 650 *ms* dan rata-rata waktu pengiriman data dari *Sensor Node* ke *Sink Node* adalah 1846,4 *ms*.

Tabel 1. Pengujian Delay Pengiriman Data SN-2 ke Sink Node

No.	SN-2 ke RN-2 (ms)	RN-2 ke RN-1 (ms)	RN-1 ke *Sink Node* (ms)	Total (ms)
1	611	612	612	1835
2	624	610	620	1854
3	616	611	621	1848
4	610	617	818	1845
5	622	628	617	1867
6	613	620	621	1854
7	609	610	616	1835
8	625	617	607	1849
9	611	609	621	1841
10	602	611	623	1836
Rata-rata				1846,4

4. KESIMPULAN DAN RENCANA PENGEMBANGAN

Kesimpulan dari penelitian ini adalah untuk meningkatkan jangkauan jarak dari BLE kita bisa memanfaatkan komunikasi *multi-hop.* Dengan model komunikasi ini data dilewatkan melalu node terdekat untuk kemudian diteruskan ke node selanjutnya sampai dengan node tujuan. Berdasarkan hasil pengujian diperoleh bahwa implementasi komunikasi *multi-hop* ini dapan berjalan dengan baikdengan tingkat keberhasilan 100% dari 10 kali pengujian dengan waktu pengiriman satu hop kurang dari 650 *ms* dan rata-rata waktu pengiriman dari *Sensor Node* ke *Sink Node* (3-hop) adalah 1846,4 *ms*.

Adapun rencana pengembangan dari penelitian ini adalah mengoptimasi state yang ada dalam setiap node sehingga waktu pengiriman data dapat dikurangi lagi.

DAFTAR PUSTAKA

C. JUNG, K. KIM, J. SEO, B. N. SILVA & K. HAN. 2017. Topology Configuration and Multihop Routing Protocol for Bluetooth Low Energy Networks, in IEEE Access, vol. 5, pp. 9587-9598, doi: 10.1109/ACCESS.2017.2707556.

INDRAYANA, A., PRIMANANDA, R., & AMRON, K. 2017. Rancang Bangun Sistem Komunikasi Bluetooth Low Energy (BLE) Pada Sistem Pengamatan Tekanan Darah. Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, vol. 2, no. 8, p. 2462-2472. ISSN 2548-964X.

J. KIM, S. KANG & J. PARK. 2015. Bluetooth-based tree topology network for wireless industrial applications. 2015 15th International Conference on Control, Automation and Systems (ICCAS), Busan, pp. 1305-1308, doi: 10.1109/ICCAS.2015.7364839.

L. LEONARDI, G. PATTI & L. LO BELLO, "Multi-Hop Real-Time Communications Over Bluetooth Low Energy Industrial Wireless Mesh Networks," in IEEE Access, vol. 6, pp. 26505-26519, 2018.

M. FUJIMOTO, S. MATSUMOTO, Y. ARAKAWA, H. SUWA & K. YASUMOTO. 2016. Development of BLE-Based Multi-hop Communication System for Detecting Slope Failure Using Smartphones," 2016 45th International Conference on Parallel Processing Workshops (ICPPW), Philadelphia, PA, pp. 16-21.

S. WANG & K. CHIANG.2017. BLE Tree Networks for Sensor Devices in Internet of

Things. 2017 IEEE 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech), pp. 1304-1309, doi: 10.1109/DASC-PICom-DataCom-CyberSciTec.2017.210.

Z. GUO, I. G. HARRIS, L. TSAUR & X. CHEN. 2015. An on-demand scatternet formation and multi-hop routing protocol for BLE-based wireless sensor networks," 2015 IEEE Wireless Communications and Networking Conference (WCNC), New Orleans, LA, pp. 1590-1595.

Halaman ini sengaja dikosongkan