

Implementasi Master-slave Pada *Embedded System* Menggunakan Komunikasi RS-485

Agusma Wajiansyah^{1*)}, Supriadi²⁾, Noval Ramadhan³⁾, Rizky Sandria⁴⁾, dan Lalu Muhammad Dian Pratama⁵⁾

^{1, 2, 3, 4, 5)} Department of Information Technology, Politeknik Negeri Samarinda, Indonesia
Corresponding Email: ^{*)} agusma.wajiansyah@gmail.com

Abstract – The use of multiprocessor methods in robotics systems has a significant impact on overall robot performance. The Master-slave method is a model of a multiprocessor system where there are several processors that communicate with each other to carry out the robot's overall function. RS-485 can be used as a communication model in the master-slave method. RS-485 is a development of RS-232 which has the ability to communicate with several nodes. In this research, an experiment will be conducted to implement RS-485 to support the master-slave processor communication. Stages of research began with making system design, which includes the design of embedded hardware systems, the design of data communication protocols on RS-485 networks, software design, followed by implementation and testing. The test is carried out to measure the time response of the device to three data transmission models, namely broadcast, addressing slaves without responding and addressing slaves with responses. The test results carried out on three slaves with a communication speed of 9600 bps. Measured response time on broadcast data transmission is 8ms, and address slave without response is 7ms. Whereas delivery by addressing slaves with responses, shows that the measurement method cannot be applied.

Key words : RS-485, Master-Slave processor communication.

I. PENDAHULUAN

Sistem tertanam atau yang lebih dikenal dengan *embedded system* pada decade terakhir telah mendapat perhatian yang sangat signifikan. Ini dikarenakan sistem tertanam sangat flexible untuk diaplikasikan diberbagai bidang. Contoh pengaplikasiannya adalah sebagai perangkat pada bidang instrumentasi, pengendalian, komunikasi, entertainment, robotika dan lain sebagainya. Menurut [1] sistem tertanam adalah kombinasi perangkat keras dan perangkat lunak untuk merespon kejadian di sekitarnya dalam batas yang ditentukan. Sistem tertanam bisa terdiri dari komputer kecil dengan mikroprosesor atau mikrokontroler dan perangkat pendukungnya. Untuk meningkatkan kemampuan komputasi pada *embedded system* dapat digunakan beberapa microprocessor. Teknologi ini dikenal dengan istilah *multiprocessor*.

Penggunaan teknologi *multiprocessor* pada *embedded system* dapat memberikan dampak yang signifikan terhadap proses eksekusi program. Contohnya pada

aplikasi bidang robotika, pada penelitian [2] terukur peningkatan performance sebesar 7.3% jika dibandingkan dengan sistem berbasis single processor. Salah satu metode untuk digunakan pada sistem *multiprocessor* adalah metode Master-slave processor. Beberapa *processor* terhubung melalui kanal komunikasi, salah satu processor bertindak sebagai master dan sisanya sebagai slave. Komunikasi Master-slave processor dapat diterapkan dengan menggunakan media I2C, SPI atau komunikasi multidrop RS-485.

Menurut [3], Komunikasi multidrop RS-485 merupakan salah satu standar komunikasi serial yang dapat menjangkau jarak sepanjang 1.2 Km. untuk berkomunikasi, pada RS-485 menggunakan sepasang media penghantar berbentuk *twisted*. Standar ANSI TIA/EIA-485-A menetapkan kedua penghantar tersebut dengan notasi 'A' dan 'B'. pertukaran informasi pada RS-485 direalisasikan dengan perubahan tegangan pada kedua penghantar tersebut. Jika tegangan pada penghantar 'A' lebih besar dari penghantar 'B' maka dikenal dengan logika *high* dan demikian pula sebaliknya.

Penerapan RS-485 telah dilakukan oleh [4]. Pada penelitian tersebut, ekperimental dilakukan dengan membangun 2 buah perangkat slave dan sebuah perangkat master yang terkoneksi dengan topologi jaringan BUS. Kedua perangkat slave bertujuan untuk melakukan pengukuran temperature lingkungan dan kelembaban udara relative pada titik koordinat tertentu yang didapat dengan bantuan GPS *receiver*. Sedangkan perangkat master bertujuan untuk menampilkan informasi pengukuran. Dari hasil uji yang dilakukan 360 paket data terkirim selama 360 detik tanpa terjadi kesalahan.

Penerapan RS-485 juga telah dilakukan oleh [5]. Pada penelitian tersebut digunakan protocol MODBUS yang secara hirarki berjalan diatas RS-485. Komunikasi RS-485 MODBUS digunakan untuk menghubungkan raspberry-pi dan *inverter* pada sistem pembangkit tenaga surya, dan selanjutnya sistem pembangkit ini terkoneksi kembali ke internet dengan menggunakan teknologi *Internet Of Think* (IOT). Dengan terbentuknya komunikasi tersebut maka dapat dilakukan proses pengaturan dan monitoring sistem pembangkit tenaga surya tersebut. Penelitian yang berkaitan dengan pengaplikasian MODBUS pada jaringan komunikasi RS-485 juga dapat ditemukan pada [6][7]

Pada penelitian [8] menggunakan RS-485 untuk membangun smart classroom. Desain arsitektur melibatkan beberapa teknologi baru antara lain Internet Of Think (IoT). Komunikasi RS-485 digunakan untuk

mengendalikan perangkat HAVAC pada *classrooms* dan juga untuk mengendalikan perangkat penerangan. Dengan adanya desain ini diharapkan akan memudahkan proses belajar mengajar dan administrasinya.

Pada penelitian [9] bertujuan untuk pemantauan bahan berbahaya pada angkutan transportasi. Penggunaan RS-485 sebagai jaringan sensor yang digunakan untuk membaca kondisi sekitar bahan berbahaya yang diangkut pada kendaraan. Selain sensor, posisi kendaraan juga diakuisisi dengan menggunakan GPS (*Global Positioning System*). Semua parameter dari sensor dan posisi kendaraan kemudian dikirim ke pusat pemantauan melalui jaringan internet. pemantauan bahan berbahaya dilakukan secara *real-time*, sehingga dapat membuat sistem tranfortasi untuk bahan berbahaya menjadi lebih efektif dan transparan dan dapat mengurangi resiko terjadi kecelakaan.

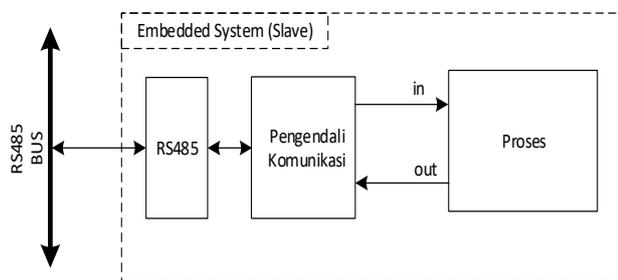
Pada penelitian ini diterapkan protocol komunikasi untuk jaringan komunikasi multidrop RS-485 pada system master-slave multiprosesor. Berbeda dengan penelitian-penelitian yang telah dijelaskan, pada penelitian ini akan dibahas protocol komunikasi pada jaringan RS-485 yang digunakan. Ekperimental dilakukan dengan membangun 3 perangkat *embedded system* yang difungsikan sebagai prosesor slave. Ketiga perangkat *embedded system* tersebut ditanamkan program protocol komunikasi dan beberapa *task* sederhana pada program utamanya. Kemudian akan diukur *respons time* dari setiap *processor slave* terhadap tiga model pengiriman paket data yaitu secara *broadcast*, *addressing slave* dan *addressing slave* dengan *response*.

II. METODOLOGI

Pelaksanaan penelitian mengikuti langkah – langkah yaitu : desain sistem, implementasi dan pengujian. Desain sistem terdiri dari desain perangkat keras, desain protocol komunikasi untuk jaringan multidrop RS-485 dan desain perangkat lunak.

A. Blok diagram sistem

Desain sistem pada penelitian ini ditunjukkan pada Gambar 1.



Gambar 1. Blok diagram modul slave

Setiap perangkat slave terhubung kedalam jaringan komunikasi membentuk topologi jaringan Bus. Pada jaringan ini hanya terdapat perangkat dengan 2 fungsi utama yaitu *master* dan *slave*. Pada penelitian ini hanya digunakan satu buah *master* dan 3 buah *slave*. Pada sistem master-slave *multiprocessor*, *master* merupakan

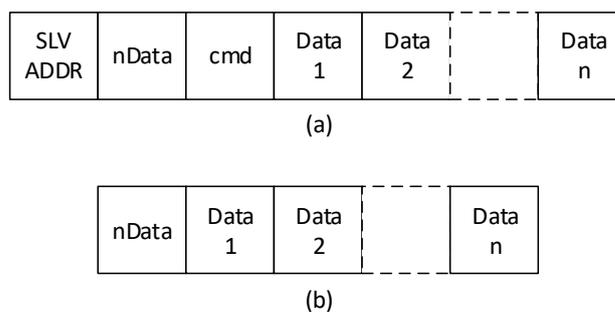
processor aktif yang memulai untuk berkomunikasi sedangkan *slave* merupakan processor pasif yang hanya menerima komunikasi dari *master*.

Pada Gambar 1, blok Proses merupakan fungsi utama dari perangkat slave. Contoh dari Blok proses dapat berupa sistem kendali Motor DC, sistem Akuisisi data sensor dan lain-lain. Setiap proses memiliki parameter input dan output. kedua parameter inilah yang akan diatur atau dimonitor melalui bus RS-485 dengan perantara blok pengendali komunikasi.

Fungsi utama dari blok pengendali komunikasi adalah menyediakan protocol dalam proses pengiriman dan penerimaan data antara perangkat master dan slave, pada jaringan RS-485.

B. Paket data

Setiap data yang dikirim dari perangkat master ke slave atau sebaliknya, akan diencapsulasi hingga menjadi paket data. Paket data yang digunakan terdiri dari dua jenis yaitu paket data dari *master* ke *slave* (*Request*) dan paket data dari *slave* ke *master* (*Response*). Gambar 2 merupakan visualisasi dari kedua jenis paket data tersebut.



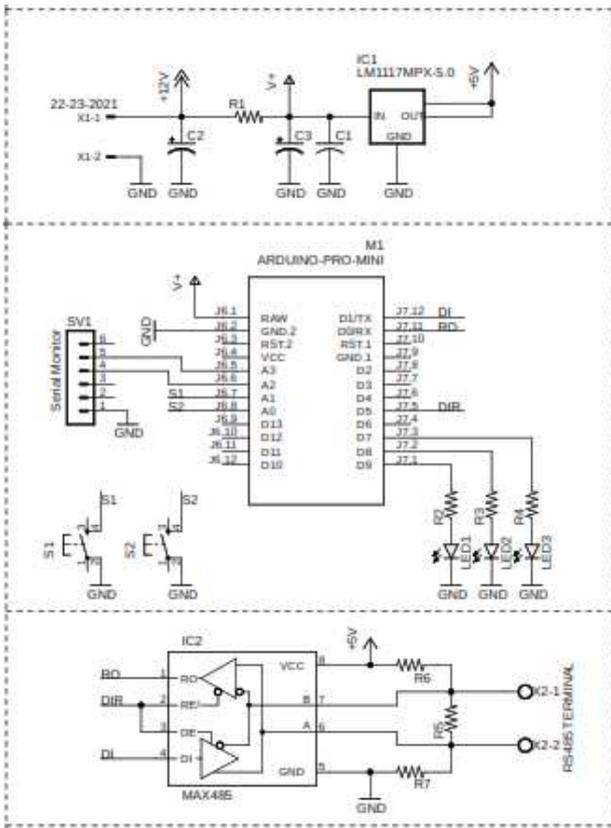
Gambar 2. Paket Data (a) Request, (b) Response

Pada paket *request*, *SLV ADDR* merupakan data pertama yang dikirim. Berisi alamat *slave* yang dituju. *SLV ADDR* memiliki panjang data sebesar 1 byte, pemberian alamat dimulai dari 01H sampai dengan FFH. Alamat 00H digunakan untuk keperluan *broadcast*. Data berikutnya adalah 'nData' yang memberikan informasi jumlah data terkirim.

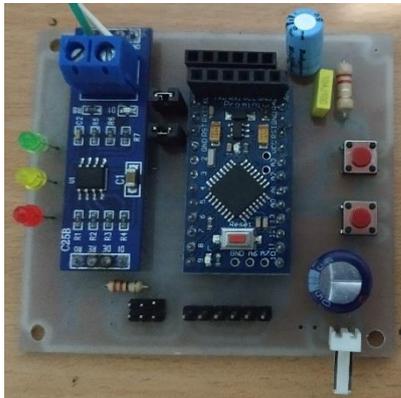
Pada bagian 'cmd' dari paket data *request*, memberikan informasi perintah yang harus dilakukan oleh *slave*. Lebar data 'cmd' sebesar 1 byte, sehingga dapat dibentuk sebanyak 256 perintah. Bagian dari paket data berikutnya adalah 'Data 1', 'Data 2' sampai dengan 'Data n' merupakan parameter dari perintah. Setiap perintah dapat memiliki parameter atau tidak, tergantung dari keperluannya. Jumlah data yang dikirim maksimal sebanyak 256 data, hal ini berlaku jika 'nData' berukuran 1 Byte.

Pada paket data *Response* hanya terdiri dari 'nData' dan 'Data 1', 'Data 2' sampai dengan 'Data n' yang memiliki arti sama pada paket data *request*.

Data dikirim secara serial dengan menggunakan USART (*Universal Synchronous and Asynchronous serial Receiver and Transmitter*) yang telah tersedia pada mikrokontroler. Frame data pada USART ditunjukkan pada Gambar 3. Setiap frame terdiri dari *Star bit* (*St*), 9 bit



Gambar 6. Desain rangkaian Slave



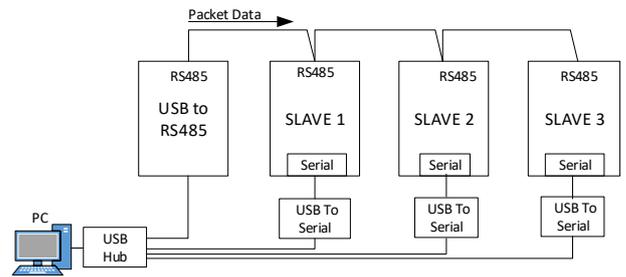
Gambar 7. tampilan Modul Slave

Pada penelitian ini dilakukan 3 jenis pengujian, yaitu pengujian dengan *broadcast*, *addressing slave* dan *addressing* dengan respon data. Pengujian tersebut menggunakan blok diagram rangkaian yang ditunjukkan pada Gambar 8. Antara *USB to RS-485* ke Slave 1, Slave 1 ke Slave 2 dan Slave 2 ke Slave 3 terhubung menggunakan kabel penghantar dengan panjang 2m, sedangkan kecepatan transmisi data yang digunakan sebesar 9600bps. Untuk tujuan pengujian setiap modul *slave* ditambahkan sebuah port komunikasi untuk tujuan mengirimkan paket data yang telah diterima pada jalur bus RS-485. Personal Computer (PC) digunakan sebagai perangkat *master*, dan juga sebagai perangkat monitor yang akan menerima data dari modul *slave*. Data yang dimonitor ini merupakan paket data yang diterima modul

slave pada jalur komunikasi RS-485. Untuk itu pada PC digunakan 4 buah Port Komunikasi Serial, yang terdiri dari 1 port untuk mengirim/menerima data dari bus RS-485, dan 3 buah port untuk menerima data dari masing modul slave. Pengiriman data monitor dilakukan pada saat *slave* dalam kondisi *idle*.

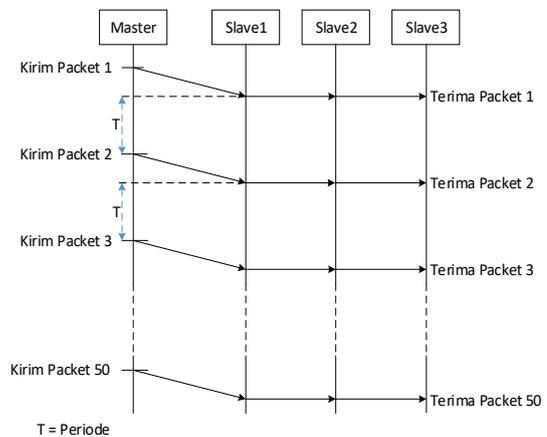
A. Pengujian dengan *broadcast*

Pengujian ini bertujuan untuk melihat kinerja perangkat slave dalam menerima paket data. Paket data dikirimkan dengan periode pengiriman antar paket data sebesar 10ms, 8ms, 6ms, 4ms, 2ms. Sebuah paket data pada pengujian ini memiliki panjang sebesar 11 Byte dengan format sesuai dengan Gambar 2(a) dimana 8 Byte digunakan sebagai parameter perintah.



Gambar 8. Blok Diagram Rangkaian Pengujian Komunikasi RS-485

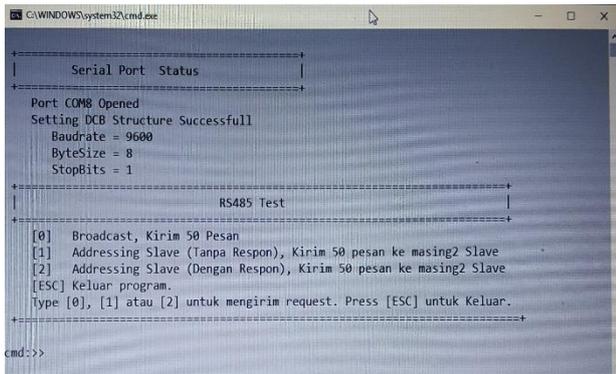
Gambar 9 menunjukkan model pengiriman data yang digunakan. Pada pengujian ini dikirimkan sebanyak 50 paket data *broadcast*



Gambar 9. Model pengiriman paket data pada pengujian broadcast

Pada pengujian ini akan hitung jumlah data yang hilang saat proses transmisi terhadap perubahan periode pengiriman. Pengiriman paket data ke jalur komunikasi RS-485 dilakukan dengan Aplikasi yang dibuat menggunakan pemrograman C pada PC, Gambar 10 Menunjukkan Tampilan aplikasi tersebut. Hasil pengukuran ini ditunjukkan pada table 1. Dari percobaan ini dapat dilihat bahwa slave dapat menerima paket data dari master dengan baik dengan interval waktu pengiriman paket lebih besar dari 8ms. Jika antar paket

data yang dikirim lebih kecil dari nilai ini maka resiko kehilangan data saat pengiriman makin besar pula.



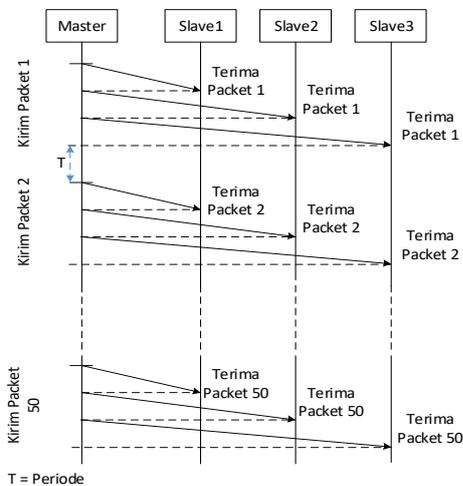
Gambar 10. Tampilan program Aplikasi pada PC untuk mengirim data ke jalur komunikasi RS-485

Tabel 1. Hasil pengukuran jumlah data yang diterima modul slave saat pengiriman paket data secara Broadcast

Periode Pengiriman	Jumlah paket data diterima		
	Salve 1	Salve 2	Salve 3
10ms	100%	100%	100%
9ms	100%	100%	100%
8ms	100%	100%	100%
7ms	25.6%	25.6%	25.6%
6ms	0%	0%	0%
5ms	0%	0.3%	0.3%
4ms	0%	0%	0%

B. Pengujian dengan slave addressing

Spesifikasi data yang dikirimkan pada pengujian ini sedikit berbeda dengan pengujian sebelumnya. Perbedaan terletak pada 'SLV ADDR' yang berisi alamat slave tujuan, dan dengan panjang paket data sebesar 11 Byte. Model pengiriman data pada pengujian ini ditunjukkan pada Gambar 11. Paket data dikirim secara sequential dimulai dari paket data untuk slave 1, slave 2 dan slave 3. Periode pengiriman untuk sequential berikutnya dimulai kembali setelah jeda sebesar 10ms, 9ms, 8ms, 7ms, 6ms, 5ms dan 4ms. Proses pengiriman ini diulang 50 kali untuk setiap slave.



Gambar 11. Model pengiriman paket data pada pengujian slave addressing

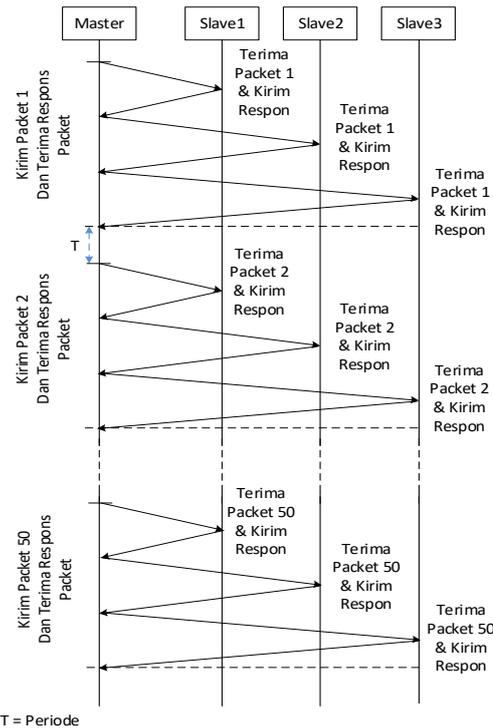
Pada pengujian ini juga diukur jumlah data yang hilang selama proses transmisi. Hasil pengujian ditunjukkan pada table 2. Dari table ini kehilangan data mulai terjadi saat periode pengiriman sebesar 8ms dan semakin kecil periode pengiriman maka resiko kehilangan data akan semakin besar.

Tabel 2. Hasil pengukuran jumlah data yang diterima modul slave saat pengiriman paket data secara Slave Addressing

Periode Pengiriman	Jumlah paket data diterima		
	Salve 1	Salve 2	Salve 3
10ms	100%	100%	100%
9ms	100%	100%	100%
8ms	99.6%	98.8%	99.2%
7ms	98%	98.8%	99.2%
6ms	86.4%	83.2%	88.8%
5ms	7.2%	8.8%	9.2%
4ms	0%	0%	0%

C. Pengujian dengan slave addressing dan respons

Spesifikasi data yang dikirim pada pengujian ini tidak jauh berbeda dengan pengujian sebelumnya. Perbedaan hanya terdapat pada 'cmd' yang berisi perintah yang mengindikasikan master meminta response dari slave. Model pengujian ini ditunjukkan pada Gambar 12.



Gambar 12. Model pengiriman data pada pengujian slave addressing dengan response

Setiap master mengirim paket data ke slave, master akan menunggu data response dari slave tersebut. Setelah data diterima, master kembali mengirimkan paket data ke modul slave berikutnya. Table 3 menunjukkan hasil pengujian dengan periode sebesar 10ms sampai dengan 0ms. Dari table ini tampak bahwa pengujian addressing slave dengan response tidak berpengaruh terhadap perubahan periode pengiriman. hal ini mengindikasikan

bahwa slave dapat dialamati kembali setelah sebelumnya mendapatkan perintah yang sama. Hal ini berbeda dengan saat slave mendapat perintah *broadcast*, dimana saat itu slave memerlukan waktu sebesar 8ms sebelum mendapat perintah *broadcast* kembali.

Tabel 3. Hasil pengukuran jumlah data yang diterima modul slave saat pengiriman paket data secara *Slave Addressing* dengan *Response*

Periode Pengiriman	Jumlah paket data yang diterima (%)					
	MS < > SL1		MS < > SL2		MS < > SL3	
10ms	100	100	100	100	100	100
9ms	100	100	100	100	100	100
8ms	100	100	100	100	100	100
7ms	100	100	100	100	100	100
6ms	100	100	100	100	100	100
5ms	100	100	100	100	100	100
4ms	100	100	100	100	100	100
3ms	100	100	100	100	100	100
2ms	100	100	100	100	100	100
1ms	100	100	100	100	100	100
0	100	100	100	100	100	100

IV. KESIMPULAN

Berdasarkan hasil pengujian pada perangkat *master-slave* berbasis komunikasi RS-485, dapat disimpulkan bahwa protokol komunikasi yang ditanamkan pada *embedded system* berbasis mikrokontroler ATmega328, dan dengan kecepatan transmisi data pada jalur komunikasi RS-485 sebesar 9600bps memiliki *response time* sebesar 8ms saat menerima paket data secara *broadcast*. Sedangkan saat paket data diterima secara *addressed*, *response time* menjadi sebesar 7ms. Untuk pengujian pengiriman paket data secara *addressed* dengan *response*, maka didapatkan bahwa periode pengiriman data tidak berpengaruh. Hal ini menunjukkan bahwa setelah selesai mengerjakan perintah dari paket data ini maka modul *slave* dapat menerima paket data berikutnya.

UCAPAN TERIMA KASIH

Terima kasih disampaikan kepada P3M Politeknik Negeri Samarinda yang telah memberikan dukungan dan dana dalam penelitian ini.

DAFTAR PUSTAKA

- [1] H. A. Alshazly and M. Hassaballah, "An Embedded System for a Bluetooth Controlled Mobile Robot Based on the ATmega8535 Microcontroller," *Egypt. Comput. Sci. J.*, vol. 40, no. 1, pp. 61–72, 2016.
- [2] A. Wajiansyah, H. Purwadi, A. Astagani, and S. Supriadi, "Implementation of master-slave method on multiprocessor-based embedded system: case study on mobile robot," *Int. J. Eng. Technol.*, vol. 7, no. 2, pp. 53–56, 2018.
- [3] H. Marais, "Analog Device Application Note (AN-960), RS-485/RS-422 Circuit Implementation Guide," 2008.
- [4] K. S. Wibawa, A. A. K. O. Sudana, and P. W. Buana, "Mikrokontroler Sistem Komunikasi Sensor Jamak Menggunakan Serial Rs-485 Multi Processor Communication," *LONTAR Komput.*, vol. 7, no. 2, pp. 122–131, 2016.
- [5] M. Siva, R. Krishna, K. Dinesh, and N. S. Shanbog, "Low Cost Remote Monitoring of Solar Plant through RS485 Communication," *Int. J. Innov. Technol. Explor. Eng.*, vol. 8, no. 9, pp. 3034–3037, 2019.
- [6] A. S. Ashtekar, B. J. Parvat, and C. B. Kadu, "Application of MODBUS to Communicate the PLC and Lab VIEW for Real Time Process Control," no. 11, pp. 41–45, 2013.
- [7] A. W. Wardhana, "Monitoring and Control System for Building Application Using MODBUS Remote Terminal Unit Protocol With AVR Atmega Family Microcontroller," vol. 6, no. 2, pp. 1–11, 2016.
- [8] L. Huang and J. Su, "A Context Aware Smart Classroom Architecture for Smart Campuses," *Appl. Sci.*, vol. 9, no. 1837, pp. 1–34, 2019.
- [9] Y. Xie, M. Yu, J. Fu, D. Chen, and C. Yang, "A hazmat transportation monitoring system based on Global Positioning System/Beidou Navigation Satellite System and RS485 bus," *Proc. - 2016 9th Int. Congr. Image Signal Process. Biomed. Eng. Informatics, CISP-BMEI 2016*, pp. 1059–1063, 2017.
- [10] Y. Lu, T. Hu, and X. Wang, "Information centric networking media streaming experiment platform design," in *Smart Computing and Communication, Second International Conference, SmartCom 2017*, 2017, vol. 10699 LNCS, pp. 332–343.