Rancang Bangun Sistem SCADA Berbasis Raspberry Pi

Putra Arisandy[1], Sisdarmanto Adinandra[2], Medilla Kusriyanto[3]

Teknik Elektro, Fakultas Teknologi Industri
Universitas Islam Indonesia
Jl Kaliurang KM 14,5 Yogyakarta, Indonesia
putra.elco@gmail.com

Abstrak— Sistem SCADA berbasis Raspberry Pi untuk menggantikan fungsi komputer cukup efektif untuk meminimalkan biaya pengadaan dan biaya operasional. Selain itu perancangan sistem SCADA berbasis *open source* juga dapat menggantikan *software* SCADA yang memiliki lisensi berbayar. Pada penelitian ini sistem SCADA pada Raspberry Pi digunakan untuk mengakusisi data dari tiga buah *slave* mikrokontroler dengan komunikasi RS485. Satu buah *slave* juga dihubungkan dengan PLC LG Master K200S dengan komunikasi RS232C untuk mengakusisi data proses produksi pada *plant* konveyor. Kedua komunikasi RS485 dan RS232C dilakukan dengan menggunakan metode *request* dan *response*. Dimana master melakukan *request* terhadap *slave* yang dituju, kemudian *slave* membalas sesuai dengan *request* yang diterima. Untuk melakukan komunikasi dengan seluruh *slave* menggunakan komunikasi RS485, protokol komunikasi dibangun dengan format *header*, alamat, data, dan *tail* agar dapat melakukan proses *reading/writing* menuju alamat *slave* yang tepat. Komunikasi dengan PLC menggunakan komunikasi RS232C menggunakan format protokol yang telah ditentukan pada *datasheet* PLC LG Master K *series*. Segala sistem yang menyangkut *hardware* dan *software* pada sistem ini telah bekerja sesuai dengan tujuan penelitian. Metode *request* dan *response* menggunakan protokol RS485 dan RS232C dapat dieksekusi sesuai dengan alamat *slave* yang dituju. Untuk melakukan komunikasi dengan seluruh *slave* menggunakan komunikasi RS485, waktu sampling yang digunakan sebesar 333 *ms* per *slave* dengan tingkat *error* data saat melakukan komunikasi sebesar 22,66 % yang disebabkan oleh *noise*.

Kata kumci—Raspberry Pi SCADA, Master Slave RS485, LG Master K Arduino

I. PENDAHULUAN

Kemampuan perangkat – perangkat elektronik seperti komputer dan *smartphone* yang dapat dieksplorasi secara *open source* memungkinkan *user* untuk ikut serta dalam proses pengembangan. Pada dunia industri maupun *home automation* perangkat seperti *smartphone* dan komputer banyak digunakan dalam sistem kontrol dan monitoring.

Dalam dunia automasi, sistem kontrol dan monitoring berbasis *Graphical User Interface* (GUI) disebut dengan *Supervisory Control And Data Acquisition* (SCADA). SCADA dapat diartikan sebagai pengumpul data dan sistem pengendali yang ditampilkan dalam bentuk antarmuka untuk memudahkan proses monitoring. Perancangan dan pengoperasian SCADA tidak hanya terbatas pada *personal computer* (PC) dan *smartphone*. SCADA juga dapat dijalankan pada sebuah perangkat *mini* PC Raspberry Pi.

Penggunaan Raspberry Pi sebagai pengganti PC dapat memangkas biaya pengadaan dan operasional yang harus dikeluarkan. Selain itu sistem SCADA berbasis *open source* juga dapat menggantikan fungsi *software* SCADA yang memiliki lisensi berbayar.

Pada penelitian ini, Raspberry Pi digunakan sebagai *master* proses kontrol instalasi penerangan, monitoring suhu, dan monitoring proses produksi pada *plant* konveyor. Raspberry Pi akan dihubungkan dengan 3 *slave* mikrokontroler menggunakan komunikasi RS485. Salah satu *slave* akan terhubung dengan *Programmable Logic Controller* (PLC) melalui komunikasi RS232C.

Proses komunikasi antar kontroler membutuhkan sebuah protokol komunikasi agar proses pertukaran data dapat diterima sesuai dengan alamat yang dituju. Protokol komunikasi RS485 dirancang sendiri oleh peneliti, Sedangkan komunikasi RS232C dengan PLC menggunakan protokol yang sudah ditetapkan oleh *vendor* pembuat sesuai dengan *datasheet* PLC tersebut.

II. TINJAUAN PUSTAKA

A. Komunikasi Serial/UART Raspberry Pi

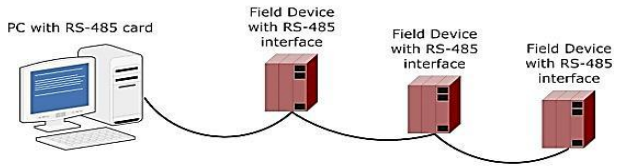
Raspberry Pi 3 memilki 2 buah *built-in Universal Asynchronous Receiver – Tranmitter* (UART) yaitu PL011 dan *mini* UART. Untuk melakukan komunikasi UART menggunakan pin GPIO pada Raspberry Pi 3, perangkat yang digunakan adalah *mini* UART.

Komunikasi UART pada Raspberry Pi bekerja pada tegangan 3.3V sehingga butuh penanganan ekstra saat dihubungkan dengan RS48 *Converter*. Sebuah *adapter* penyetara tegangan harus digunakan untuk menjembatani kedua protokol apabila keduanya memiliki *level* tegangan yang berbeda [1].

B. Komunikasi RS485

Komunikasi RS485 merupakan komunikasi yang menggunakan dua buah kabel untuk menghubungkan antar perangkat. RS485 banyak digunakan untuk menangani

komunikasi jarak jauh mencapai 1220 meter tanpa menggunakan *repeater*. Kelebihan lain dari komunikasi RS485 adalah dapat menghubungkan lebih dari 32 perangkat komunikasi [2]. Karena kemampuan komunikasi jarak jauh dan kemampuan untuk melakukan komunikasi *multi-device* tersebut, RS485 menjadi solusi untuk menutupi kekurangan yang dimiliki oleh komunikasi RS232. Hubungan antar perangkat menggunakan komunikasi RS485 seperti ditunjukkan Gambar 1.



Gambar 1 Contoh Komunikasi RS485

Komunikasi RS485 tidak mengatur standar protokol yang harus digunakan. Karena itu *user* bebas dalam menentukan protokol yang akan digunakan untuk komunikasi antar perangkat. Kebebasan dalam penentuan protokol ini juga membuat RS485 menjadi dasar utama terbentuknya komunikasi yang sering digunakan pada dunia industri seperti Profibus, Interbus, dll. Dalam dunia automasi sering dijumpai perangkat – perangkat seperti RS232 *to* RS485 *Conventer*, RS485 *Smart Switch*, RS485 *Repeater* yang banyak digunakan dalam dunia PLC, SCADA, dan *Remote Terminal Unit* (RTU). Semua perangkat tersebut dibangun menggunakan teknologi RS485.

C. Komunikasi RS232C Master K200S

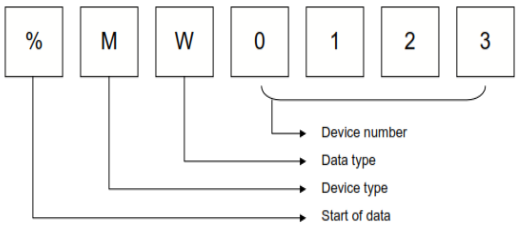
Master K200S tipe A dan C (K3P-07AS dan K3P-07CS) memiliki fitur pendukung untuk melakukan komunikasi dengan *external device* tanpa harus menggunakan modul Cnet I/F [3]. Meskipun tidak memiliki seluruh fungsi yang dapat dilakukan oleh modul Cnet I/F, fitur tersebut sangat berguna bagi *user* yang ingin melakukan pengembangan sistem yang murah berbasis jaringan RS232C.

Untuk melakukan komunikasi dengan *external device* menggunakan RS232C, terdapat standar protokol yang harus dipatuhi sesuai dengan datasheet PLC LG Master K200S. Kesalahan dalam mengaplikasikan protokol tersebut akan membuat proses penulisan dan pembacaan tidak akan dieksekusi oleh PLC. Secara umum standar protokol RS232C yang digunakan oleh Master K200S adalah seperti ditunjukan Gambar 2.

Header (ENQ)	Station No	Instruction	Instruction type	Data	Tail (EOT)	Frame check (BCC)

Gambar 2 Struktur *frame* protokol LG Master K200S

Instruksi yang digunakan dalam *frame* data dapat dilihat pada Gambar 3. Bagian ini memuat alamat register yang akan diakses.



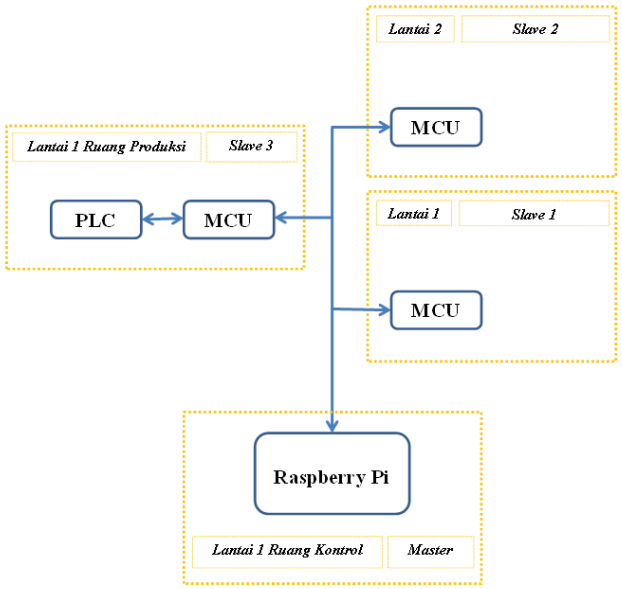
Gambar 3 Struktur *frame* data

Keterangan :

- *Start of data* : Simbol "%" sebagai awal *frame* data.
- *Device type* : *Register* yang akan dieksekusi.
- *Data type* : Tipe data bit (X) atau word (W).
- *Device number* : Alamat *register* yang akan dieksekusi.

III. METODE PENELITIAN

Diagram blok sistem yang digunakan dalam penelitian ini dapat dilihat pada gambar 4.



Gambar 4 Diagram blok sistem

Pada sistem ini Raspberry Pi berperan sebagai *master* untuk memantau dan mengatur seluruh proses yang ada pada *plant* antara lain : mengontrol instalasi penerangan, *monitoring* suhu ruangan, *monitoring* proses produksi pada *plant* konveyor, dan mengontrol proses produksi pada *plant* konveyor. Raspberry Pi terhubung dengan dua mikrokontroler ATMega16 dan satu Arduino Mega menggunakan komunikasi RS485.

Mikrokontroler ATMega16 digunakan untuk memantau suhu pada 8 ruangan menggunakan sensor LM35 dan menghidup matikan lampu pada 8 ruangan. Arduino Mega berperan untuk menerima perintah dari *master* melalui komunikasi RS485 dan meneruskannya menuju PLC LG Master K200S menggunakan komunikasi RS232C. PLC LG

Master K200S berperan untuk *monitoring* dan mengendalikan proses produksi pada *plant* konveyor.

A. *Level Shifter dan RS485 Master*

Untuk bisa melakukan komunikasi RS485 dengan ketiga *slave* mikrokontroler, dibutuhkan penyetara tegangan (*level shifter*) dan RS485 *converter*. Hubungan dari ketiga perangkat ini dapat dilihat pada Gambar 5.

Gambar 5 Diagram blok master

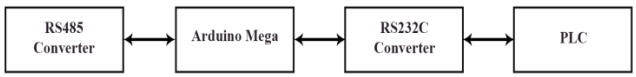
B. *RS485 Slave 1 dan 2*

Berbeda dari master, pada *slave* 1 dan *slave* 2 yang menggunakan mikrokontroler ATMega16 dapat langsung terhubung dengan RS485 *converter* tanpa harus menggunakan *level shifte*r. Hal tersebut dikarenakan level tegangan I/O dan *serial* pada ATMega16 dan level tegangan pada RS485 *converter* sama – sama bekerja pada tegangan 5V. Diagram blok rangkaian *slave* 1 dan 2 dapat dilihat pada Gambar 6.

Gambar 6 Diagram blok *slave* 1 dan 2

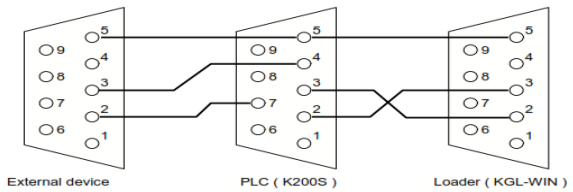
C. *RS485 dan RS232C Slave 3*

Fungsi utama *slave* 3 pada sistem ini adalah untuk melakukan akusisi data *register* PLC dan meneruskannya saat terdapat *request* dari *master*. Seperti halnya sebuah *Random Access Memory* (RAM) pada komputer, dengan cara ini proses pembacaan *register* PLC menjadi lebih cepat karena data telah diolah dan disimpan di memori *slave* 3. Tujuannya adalah untuk mendapatkan hasil pembacaan *register* secepat mungkin tanpa harus menunggu satu periode *sampling* dari *master*. Diagram blok sistem *slave* 3 dapat dilihat pada Gambar 7.



Gambar 7 Diagram blok *slave* 3

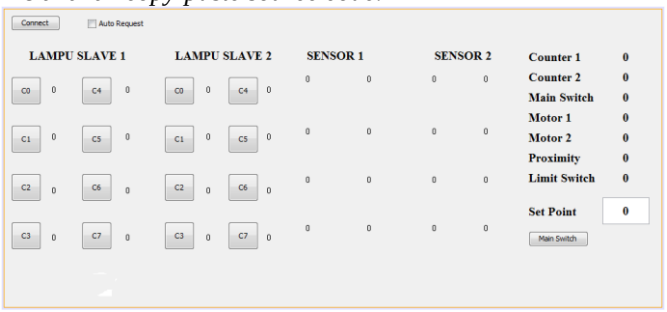
Konfigurasi kabel yang digunakan pada RS232C yang digunakan untuk melakukan komunikasi dengan *external device* dapat dilihat pada Gambar 8.



Gambar 8 Konfigurasi kabel RS232C dengan *external device*

D. *Graphical User Interface (GUI)*

Proses pembuatan GUI dibangun dengan JAVA SWING berbasis pemrograman JAVA JDK/JRE 8 menggunakan *software* Netbeans IDE 8.1. Tujuan utama digunakannya bahasa pemrograman JAVA adalah karena fleksibitlitas, dimana pemrograman dapat dilakukan melalui komputer dan di-*run* langsung menggunakan Raspberry Pi tanpa harus melakukan *copy-paste source code.*



Gambar 9 Tampilan GUI

E. *Reading dan Writing Komunikasi RS485*

Protokol komunikasi RS485 dirancang sendiri oleh peneliti untuk memudahkan proses parsing dan splitting data. Format protokol komunikasi RS485 dapat dilihat pada Gambar 10.

Header ($)	Alamat (01 s/d 03)	Instruksi	Tail (#)

Gambar 10 Frame *reading* komunikasi RS485

Pada proses writing terdapat dua bentuk format. Gambar 11 menunjukan frame writing menuju *slave* 1 dan 2. Dan Gambar 12 menunjukkan frame writing menuju *slave* 3.

Header ($)	Alamat (01 / 02)	Instruksi (W)	Port (B)	Bit (0 s/d 7)	Value (0/1)	Tail (#)

Gambar 11 *Frame writing* menuju *slave* 1 dan 2

Header ($)	Alamat (03)	Instruksi (W)	Register (M / D)	Value	Tail (#)

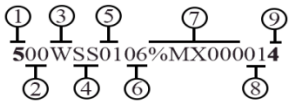
Gambar 12 *Frame writing* menuju *slave* 3

F. *Reading dan Writing Komunikasi RS232C*

Untuk melakukan reading register – register PLC yang digunakan pada sistem ini hanya membutuhkan 1 baris frame seperti ditunjukkan Gambar 13.

Gambar 13 *Frame* untuk membaca 6 *register* PLC

Register yang dapat ditulis pada penelitian ini hanya terdapat 2 buah register yaitu register M000 dan register D000. Gambar 14 menunjukkan contoh penulisan menuju register M000 bit 0.



Gambar 14 Frame untuk menulis register M000 bit 0

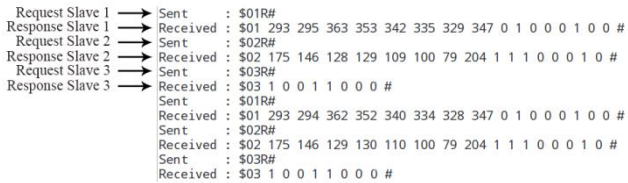
Keterangan :

1. *Header*
2. *Station number*
3. *Main instruction*
4. *Instruction type*
5. *Number of block*
6. *Length of device definition*
7. *Device definition*
8. Data
9. *Tail*

IV. HASIL DAN ANALISIS

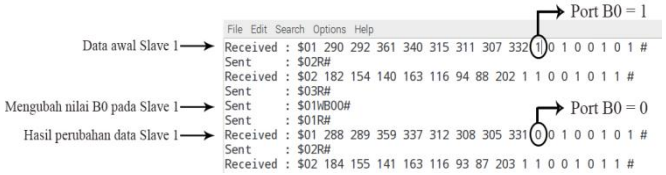
A. *Pengujian dan Analisa Komunikasi Raspberry Pi dengan Seluruh Slave*

Dari hasil pengujian yang dilakukan terhadap komunikasi RS485 antara *master* dengan seluruh *slave*, format data protokol RS485 yang dirancang oleh penulis dapat bekerja seperti yang diinginkan. Semua proses *writing/reading* data untuk *slave* tertentu juga berhasil dieksekusi sesuai dengan alamat *slave* yang dituju. Gambar 15 menunjukkan proses *reading* data dari seluruh *slave* bekerja dengan semestinya.



Gambar 15 Monitoring komunikasi RS485

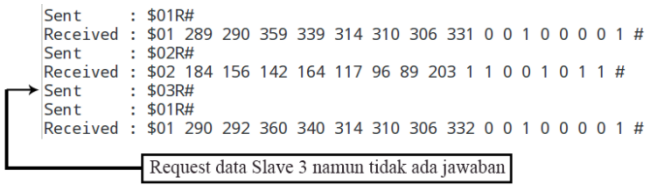
Gambar 16 menunjukkan proses untuk mengubah nilai *port* B0 pada *slave* 1 yang pada awalnya bernilai 1 dan diubah menjadi 0.



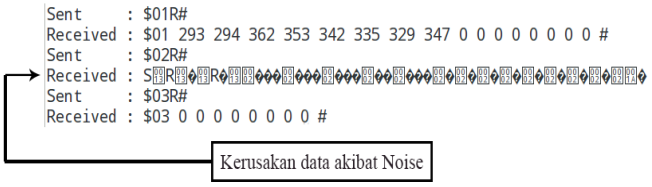
Gambar 16 Proses *writing port* B0 *slave* 1

Selama proses pengujian komunikasi RS485, seringkali ditemukan kerusakan data saat *slave* membalas perintah dari *master* dan begitu pula sebaliknya. Salah satu penyebab kerusakan data adalah pengaruh dari *noise* yang disebabkan oleh kualitas komponen, tingkat kerapian dalam proses penyolderan, dan tingkat kerapian *wiring*. Pengaruh dari *noise* menyebabkan format protokol data tidak lagi sesuai dengan format protokol yang telah ditetapkan. Saat hal tersebut terjadi maka data yang diterima tidak akan diproses, sehingga tampilan pada SCADA tidak akan di-*update* (menggunakan data yang sebelumnya).

Gambar 17 menunjukkan *slave* tidak merespon *request* dari *master* akibat kerusakan data yang terjadi saat pengiriman dari *master* menuju *slave*. Sedangkan Gambar 18 menunjukkan kerusakan data akibat *noise* yang terjadi saat pengiriman data dari *slave* menuju *master*.



Gambar 17 *Slave* 3 tidak merespon *request* dari *master*



Gambar 18 Kerusakan data akibat *noise*

Faktor lain yang menyebabkan kerusakan data dalam sistem ini adalah kecepatan *sampling* data dan *timing* pengubahan logika untuk melakukan kegiatan *read/write* pada *pin* $\overline{RE}/DE$ IC MAX485.

Tabel 1 Kerusakan data pada komunikasi RS485

Jumlah *Request* Data	*Sampling Time* (ms)	Jumlah *Error* Pada Percobaan			Rata - Rata	% Rata - Rata
		1	2	3		
300	100	140	168	152	152	50,66 %
	200	98	98	118	98	32,66 %
	300	95	83	76	83	27,66 %
	333	82	61	68	68	22,66 %
	350	59	73	66	66	22 %
	400	55	70	67	67	22,33 %

Pada Tabel 4.1 memperlihatkan kerusakan data yang terjadi dalam komunikasi RS485 yang disebabkan oleh *noise*. Berdasarkan data tersebut, kecepatan *sampling* yang digunakan dalam sistem ini adalah sebesar $333\ ms$ untuk setiap *slave*. Hal tersebut dikarenakan persentase rata – rata antara $333\ ms$, $350\ ms$, dan $400\ ms$ tidak berbeda jauh. Selain itu dengan menggunakan $333\ ms$, total waktu yang digunakan untuk mengakses seluruh *slave* kurang dari 1 detik atau hanya sebesar $999\ ms$.

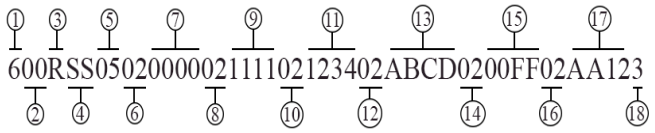
Pin $\overline{RE}/DE$ pada IC MAX485 berfungsi untuk menentukan instruksi yang ingin dilakukan. Jika *pin* $\overline{RE}/DE$ berlogika 0 maka proses yang akan dilakukan adalah proses *reading*. Sedangkan jika *pin* $\overline{RE}/DE$ berlogika 1 maka proses

yang dilakukan adalah proses *writing*. Saat melakukan proses *writing* dibutuhkan waktu jeda antara mengubah *pin* $\overline{RE}/DE$ menjadi 1, mengirim paket data, dan mengembalikan *pin* $\overline{RE}/DE$ menjadi 0. Selama proses pengujian didapatkan waktu jeda terbaik untuk meminimalisir kerusakan data adalah sebesar 10 ms.

B. Pengujian dan Analisa Konverter RS485 – RS232C (Slave3)

Pada pengujian komunikasi antara *slave* 3 dengan PLC LG Master K200S, format protokol komunikasi RS232C yang digunakan harus sesuai dengan format protokol yang telah ditetapkan oleh *vendor* pembuat. Dari hasil pengujian pada *plant* konveyor format protokol yang digunakan untuk proses *reading/writing* sesuai dengan apa yang telah dituliskan di *datasheet* LG Master K200S.

Untuk melakukan eksekusi *reading*, hanya membutuhkan satu baris *frame* untuk mengakses 5 buah *register* seperti yang telah ditunjukkan Gambar 13. Setelah instruksi tersebut dikirimkan ke PLC, maka PLC akan membalas dengan format data seperti ditunjukkan oleh Gambar 20.



Gambar 20 Frame protokol balasan PLC

Keterangan :

1. *Header* : 6 heksadesimal
2. *Station number* : *Station number* PLC
3. *Main instruction*: Perintah *reading*
4. *Instruction type* : Membaca *single* bits
5. *Number of block* : *Membaca* 6 buah *register*
6. *Length of data* : Terdapat 2 word pada “0000”
7. *Data* : Nilai *register* M000 bertipe word
8. *Length of data* : Terdapat 2 word pada “1111”
9. *Data* : Nilai *register* P000 bertipe word
10. *Length of data* : Terdapat 2 word pada “1234”
11. *Data* : Nilai *register* P002 bertipe word
12. *Length of data* : Terdapat 2 word pada “ABCD”
13. *Data* : Nilai *register* C000 bertipe word
14. *Length of data* : Terdapat 2 word pada “00FF”
15. *Data* : Nilai *register* C001 bertipe word
16. *Length of data* : Terdapat 2 word pada “AA12”
17. *Data* : Nilai *register* D000 bertipe word
18. *Tail* : 3 heksadesimal

Nilai *header* dan *tail* pada *frame* balasan PLC berbeda dari *header* dan *tail* saat melakukan *request*. Dimana saat melakukan *request*, *header* bernilai 5 heksadesimal dan *tail* bernilai 4 heksadesimal. Sedangkan pada *frame* balasan, *header* bernilai 6 heksadesimal dan *tail* bernilai 3 heksadesimal.

Karena *slave* 3 menangani 2 jenis komunikasi yaitu RS485 dengan *master* dan RS232C dengan PLC, maka *sampling time* sangat berpengaruh agar interupsi tidak saling mengganggu satu sama lain. Dari hasil pengujian yang telah dilakukan dengan mengubah - ubah nilai *sampling time*, didapatkan *sampling time* yang terbaik pada komunikasi RS232C adalah sebesar 100 ms. Pengaruh dari *sampling time* RS232C yang terlalu cepat akan menyebabkan penerimaan data pada komunikasi RS485 terganggu sehingga *slave* 3 tidak merespon *request* dari *master*. Sedangkan *sampling time* RS232C yang terlalu lambat akan menyebabkan pembacaan *register* PLC menjadi lambat atau tidak *real time*.

V. KESIMPULAN

1. Format protokol komunikasi RS485 untuk menangani komunikasi lebih dari 2 perangkat dirancang dengan format *header*, alamat perangkat, data, dan *tail*.
2. Format protokol komunikasi RS232C LG Master K200S sesuai dengan format yang telah dituliskan di *datasheet*.
3. *Sampling time* terbaik yang dibutuhkan Raspberry Pi untuk mengakses seluruh *slave* adalah sebesar 333 ms per *slave*.
4. Kerusakan data komunikasi RS485 disebabkan oleh *noise* dan *sampling time* yang terlalu cepat.
5. Pengubahan logika pada *pin* $\overline{RE}/DE$ yang terlalu cepat dapat menyebabkan kerusakan data.
6. Waktu jeda terbaik dalam pengubahan logika pada *pin* $\overline{RE}/DE$ dalah sebesar 10 ms.

VI. DAFTAR PUSTAKA

[1] Raspberrypi.org. (2018). *The Raspberry Pi UARTs – Raspbery Pi Documentation*. [online] Available at: https://www.raspberrypi.org/documentation/configuration/uart.md [Accessed 12 May 2018].

[2] BB SmartWorx, "A Practical Guide to Using RS-422 and RS-485 Serial Interfaces," *RS-422 and RS-485 Applications Ebook*, Oct. 2010. [Online] Available : http://www.bb-elec.com/Learning-Center/All-White-Papers/Serial/RS-422-and-RS-485-Applications-eBook.aspx.

[3] LG Industrial Systems, "User’s Manual LG Programmable Logic Controller Master-K," Datasheet, Accessed on Feb. 08, 2018. [Online] Available : http://www.ehaegypt.com/uploads/K200_300_1000S_yi2jwqxz.pdf.