PROTOTIPE *SMART HOUSE* DENGAN PROTOCOL *LOCAL INTERCONNECT NETWORK*

Ponang Wiratmoko[1], Hartanto K. Wardana[2], Darmawan Utomo[2]

1. Program Studi Teknik Elektro, Fakultas Teknik – UKSW
2. Program Studi Sistem Komputer, Fakultas Teknik – UKSW

Intisari

LIN digunakan dalam jaringan *Electronic Control Unit* berkecepatan rendah sebagai pengendali *sensor* dan *actuator* yang sederhana dan tidak *hard realtime* seperti *power window, sunroof,* lampu dan pengatur posisi kursi. Sistem pengendali suhu, lampu, pintu, dan detector alaram dalam *smart house* biasanya tidak memerlukan tanggapan kurang dari 1 detik. Oleh karena itu pada makalah ini diusulkan penerapan protocol LIN pada *smart house*.

Protokol LIN diaplikasikan dalam sistem maket *smart hous*e dengan *node master* berupa aplikasi pada komputer berbasis .NetFramework dan tiga *node slave* menggunakan mikrokontroler AVR ATmega. *Slave* A berfungsi sebagai pengendali suhu ruangan, *slave* B berfungsi sebagai pengendali pencahayaan ruangan dan *slave* C berfungsi sebagai pengendali jendela, pengunci pintu dan sistem peringatan maket. Ketiga *node slave* terhubung dengan *node master* yang memiliki kemampuan untuk memantau status, melakukan pengaturan mode kerja dan merekam aktivitas bus terkait *event* yang terjadi dalam jaringan.

Dari hasil percobaan, *Node master* memiliki *time base* optimal bernilai 47 ms dengan *baudrate* sebesar 19200 bps. Modul pengendali suhu memiliki kemampuan untuk menurunkan suhu dan menyesuaikan suhu ruangan berdasarkan nilai suhu tetapan. Pengaturan suhu dan pencahayaan ruangan serta pengendalian jendela, pengunci pintu dan sistem peringatan dapat dilakukan melalui aplikasi panel pengontrol *node master*.

Kata Kunci: LIN, smart house, rumah pintar

1. Latar Belakang

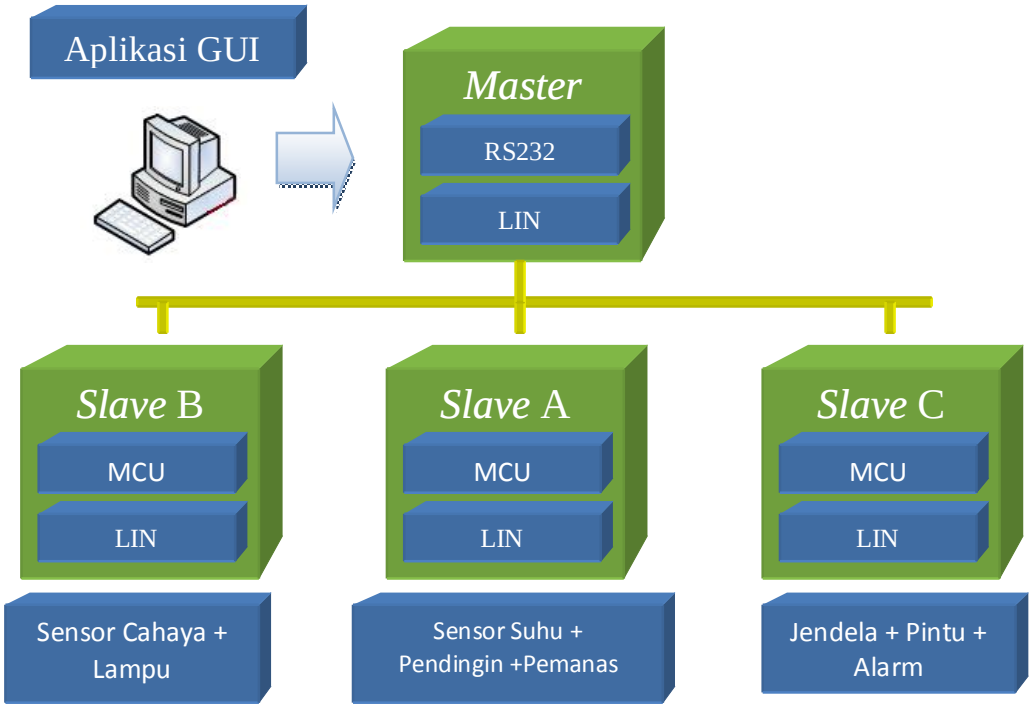
Komunikasi data saat ini menjadi hal yang sangat penting dalam dunia sistem kontrol elektronik. Hal ini tidak lepas arsitektur sistem kontrol dengan pemecahan-pemecahan bagian

unit kontrol untuk membentuk suatu *functional unit*. Setiap *functional unit* ini sering direalisasikan dengan sebuah mikrokontroler sebagai pengontrol *device* dan pengolah data. Dalam dunia otomotif *functional unit* ini lazim disebut *Electronic Control Unit* (ECU). Pembentukan *functional unit* ini dimaksudkan untuk mengurangi beban kerja dari mikroprosesor sehingga sistem diharapkan lebih responsif dan *realtime*.

Untuk menghubungkan setiap ECU maka dibuatlah sistem jaringan dengan tujuan setiap sistem dapat berkomunikasi sehingga setiap *functional unit* dapat bekerja secara sinkron. Pembentukan sistem jaringan juga dimaksudkan untuk menghilangkan kerumitan pengakabelan pada *body* aplikasi otomotif. Untuk komunikasi antar ECU saat ini banyak terdapat berbagai macam jenis protokol antara lain CAN, FlexRay,D2B, MOST dan LIN.

LIN digunakan dalam jaringan ECU berkecepatan rendah untuk pengendali *device* baik *sensor* maupun *actuator* yang sederhana dan tidak *hard realtime* seperti *power window, sunroof,* lampu, pengatur posisi kursi, penampil level oli dan bahan bakar. Alasan digunakannya LIN karena protokol yang ada (CAN) dirasa terlalu mahal walaupun protokol tersebut mempunyai kecepatan yang tinggi dan sangat responsif. Tetapi pada aplikasi sederhana unjuk kerja dari CAN tidak dibutuhkan dan berdampak pada biaya produksi yang tinggi dan tidak efisien. Oleh karenanya diciptakanlah LIN untuk mengakomodasi hal tersebut.

1.1. Gambaran Perancangan Sistem



Gambar 1. Blok Diagram Sistem Smart house dengan LIN.

Sistem adalah sebuah maket *smart hous*e dengan pengendali mikro untuk mengatur kerja dari setiap modul pengendalinya (Gambar 1). Sistem terdiri dari empat buah *node* (satu *master* dan tiga *slave*) yang mempunyai fungsi yang berbeda-beda dan semuanya akan terhubung dalam jaringan yang menggunakan protokol LIN sebagai alat komunikasi.

1.1.1. Perangkat Keras

Perangkat keras yang akan dirancang adalah sebagai berikut:

- 3 buah modul *electronic control unit* berbasis IC AVR ATmega8535 dan ATmega8 yang masing-masing modul akan merepresentasikan sebuah *node*. Setiap modul memiliki IC *tranceiver* LIN MCP 2021.
- 1 buah *converter* level tegangan logika RS232 ke level tegangan logika LIN sebagai *interface* Personal Computer (PC) dengan jaringan LIN.

- 1 buah maket smart house dengan 3 buah *node* LIN yang terpasang didalamnya, adapun modul yang terdapat pada *smart house* antara lain modul pengendali suhu, modul pencahayaan ruang dan modul pengontrol pintu dan jendela.

1.1.2. Perangkat Lunak

Perangkat lunak yang akan dirancang terbagi menjadi dua bagian besar yaitu perangkat lunak mikrokontroler pada *node slave* dan perangkat lunak *node master* pada PC. Perangkat lunak pada *node slave* berfungsi untuk :

- Melakukan komunikasi dengan *master node* baik menerima atau mengirim data.
- Mengendalikan kerja dari pendingin dan pemanas ruangan serta membaca suhu pada *node slave* pengendali suhu ruangan
- Mengendalikan aktuator pembuka tutup jendela serta memantau keadaaan pengunci pintu dan jendela pada *node slave* pengendali pintu dan jendela.
- Mengaktifkan alarm peringatan pada *node slave* pengendali pintu dan jendela jika kondisi memenuhi

Perangkat lunak pada *node master* PC berfungsi:

- Melakukan *setting* pada modul *slave*.
- Merekam aktifitas *bus*
- Memberi informasi status kerja dari modul *slave*

1.2. Spesifikasi Alat

Pada alat yang telah dirancang dan direalisasikan memiliki spesifikasi sebagai berikut:

1. Ketiga *node slave* terhubung dengan LIN *Bus* dengan *node master* dan dapat saling berkomunikasi secara *master slave*.
2. Protokol yang digunakan dalam sistem sesuai dengan spesifikasi LIN 2.1
3. Modul pengontrol suhu mempunyai kemampuan untuk mengatur suhu ruangan dalam interval 23°C -34°C dan tingkat ketelitian ±1°C.
4. Modul pengontrol cahaya ruangan dapat berjalan otomatis sesuai intensitas cahaya yang ditentukan dari pengguna atau secara manual menggunakan saklar. Kategori intensitas cahaya akan dibagi menjadi empat yaitu terang, agak terang, agak gelap dan gelap.

5. Alaram pada *slave* berbunyi jika kunci jendela atau pintu terbuka pada status terkunci.
6. Jendela dapat dibuka tutup melalui modul *slave* A dengan tombol buka dan tombol tutup.
7. *Master* dapat mengontrol semua modul pada *slave* dengan aplikasi GUI dan menampilkan status dari setiap modul *slave*.
8. *Master* memiliki kemampuan untuk melakukan *setting* pada setiap modul dalam *node slave*.

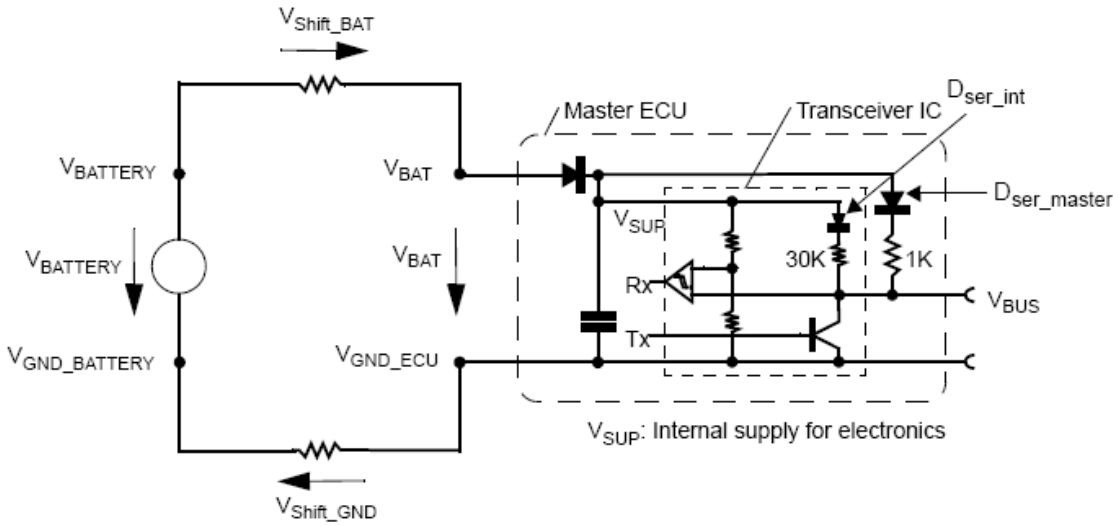
2. Kajian Pustaka

2.1. Protokol LIN [2]

Protokol LIN dibangun di atas protokol UART yang berarti semua *message* yang dikirim dalam *bus* di *encoding* sesuai protokol UART. Ini dapat menurunkan biaya produksi dari segi perancangan IC karena hanya diperlukan sebuah *tranceiver* untuk konversi tegangan saja sehingga tidak memerlukan perancangan IC untuk mengatur kerja protokol dan selanjutnya protokol cukup diatur melalui perangkat lunak. Hal ini didukung dengan hampir semua mikrokontroler telah mempunyai fasilitas UART.

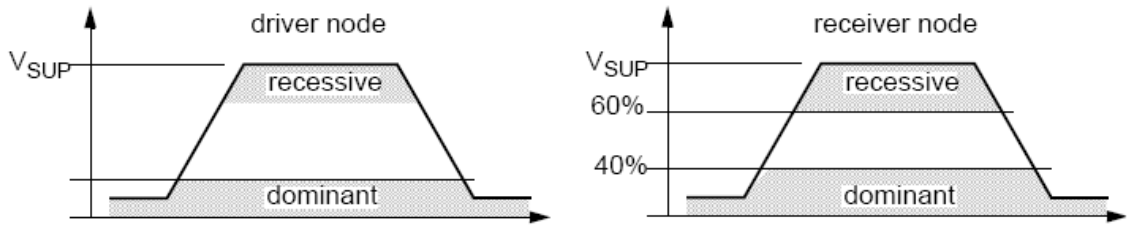
Sebagai jalur data LIN diimplementasikan menggunakan *single wire* sebagai penghubung antar *node*, oleh karenanya LIN sangat praktis dari sisi pengkabelan. Tapi disisi lain menyebabkan nilai yang lebih tinggi pada *Electromagnetic Emission* (EME) dibandingkan dengan *twisted pair* seperti halnya CAN atau MOST. Untuk menjaga EME tetap rendah maka *slew rate* dari *signal* dan juga *bandwith*nya harus dikontrol. Berdasarkan alasan diatas maka data transfer pada *bus* tidak dapat melebihi *baudrate* sebesar 20Kbs/s. Jaringan LIN mempunyai panjang maksimal 40 meter.

Bus line driver LIN berdasar dari standar ISO 9141. Terdiri dari bidirectional *bus line* LIN yang tersambung dengan setiap *driver/receiver* pada masing-masing *node* dengan terminasi sebuah resistor dan dioda ke tegangan supply positif V_{BAT} (Gambar 2). Dioda berfungsi untuk mencegah tidak terkontrolnya *power supply* ECU oleh *bus* dalam kasus *loss of ground*.



Gambar 2. LIN *bus driver line* [1, h.115]

Terdapat dua tingkat tegangan logika LIN yaitu *dominant* dan *recessive*. *Dominant* mewakili logika 0 sedangkan *recessive* mewakili logika 1. Level tegangan logika LIN untuk *node* pengirim dan penerima adalah berbeda. Dalam *node* pengirim kondisi *dominant* terjadi saat tegangan pada *bus* LIN dibawah 20% dari tegangan masukan *transceiver*, sedangkan dalam *node* penerima kondisi *dominant* terjadi saat tegangan *bus* LIN dibawah 40% tegangan masukan *transceiver*. Sementara kondisi *recessive* pada *node* pengirim terjadi saat tegangan *bus* LIN diatas 80% tegangan *transceiver* sedangkan pada *node* penerima terjadi saat tegangan *bus* LIN diatas 60% tegangan masukan *transceiver*. Kondisi tingkat logika LIN dapat dilihat pada Gambar 3. Detail tentang struktur frame LIN dapat dilihat pada acuan [2].



Gambar 3. Tingkat Tegangan Logika *Bus* pada LIN [1, h.116]

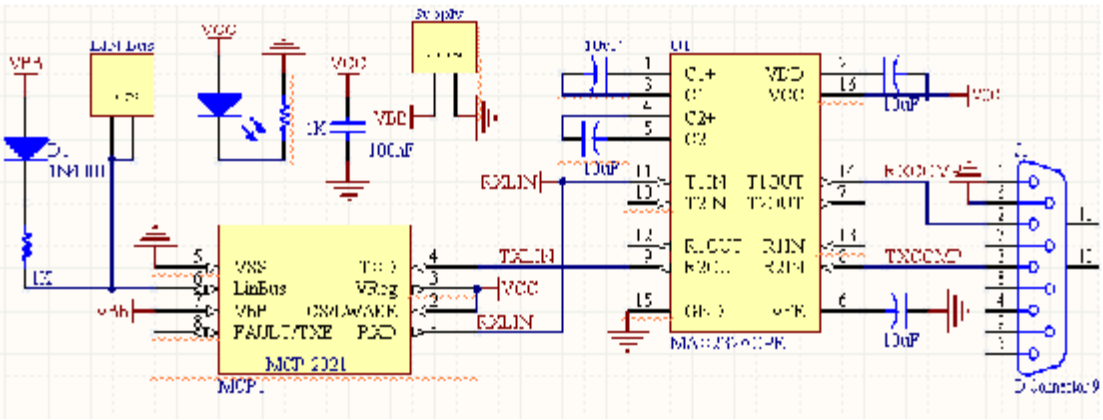
IC MCP202X menyediakan antarmuka fisik antara mikrokontroler dan *half-duplex* LIN *bus*. Hal ini dimaksudkan untuk otomotif dan industri dengan *bus* serial kecepatan sampai 20 Kbaud. MCP202X menyediakan *half-duplex*, dua arah komunikasi antarmuka antara

mikrokontroler dan jaringan serial *bus*. Perangkat ini akan mentranslasikan logika level tegangan CMOS / TTL ke logika level tegangan LIN dan sebaliknya.

3. Perancangan

3.1. Perangkat Keras *Node Master*

Node master adalah sebuah aplikasi GUI pada komputer yang dibuat dengan menggunakan bahasa pemorgramman C# dan terhubung dengan jaringan LIN melalui UART. Pada *node master* hanya terdapat satu modul elektronik yaitu *converter* tingkat tegangan logika RS232 menjadi tingkat tegangan logika LIN sesuai dengan skema *master node* pada Gambar 4. Rangkaian ini menggunakan IC MAX232 yang berfungsi menghubungkan antara modul UART dari komputer yang mempunyai keluaran data dengan level tegangan RS232 dengan jaringan level tegangan LIN yang menggunakan IC transceiver LIN MCP2021 yang mempunyai masukan data dengan level tegangan TTL.



Gambar 4. Skema Rangkaian Perangkat Keras *Master Node*

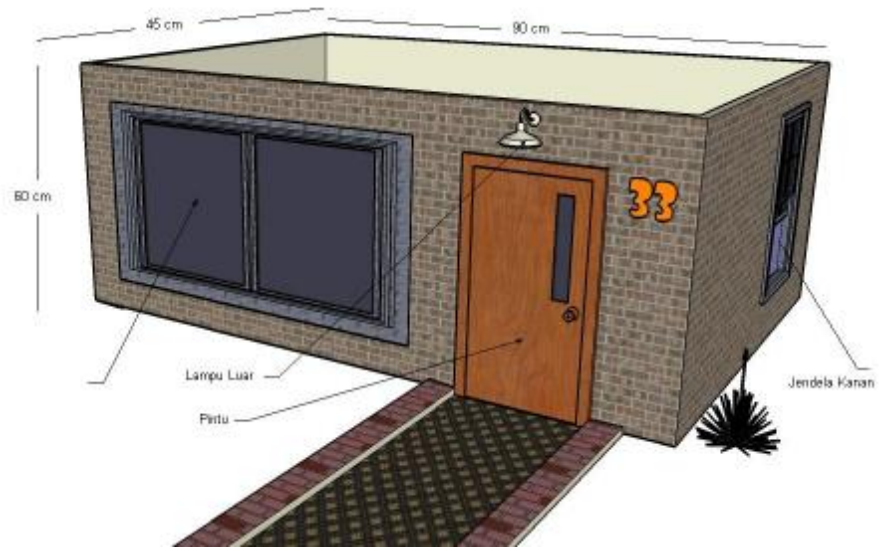
3.2. Perancangan Hardware *Slave* A

Slave A merupakan *node* yang menangani pengendalian suhu ruangan. Dalam *node* ini terpasang sebuah modul mikrokontroler ATmega8535, sensor suhu LM35, modul pemanas menggunakan elemen pemanas dan kipas AC dengan sumber tegangan 220V AC, modul pendingin menggunakan kipas DC dengan sumber dingin (*dry ice*), dan LCD 16x2 sebagai penampil data suhu ruangan.

Slave B dan *Slave* C pada protocol dasarnya sama dengan perbedaan pada yang dikendalikan. Misal untuk *Slave* B dibutuhkan relay AC, saklar manual, dan Sensor Cahaya. Sedangkan pada *Slave* C, driver motor, sensor pintu, alaram, dan pengunci pintu solenoid.

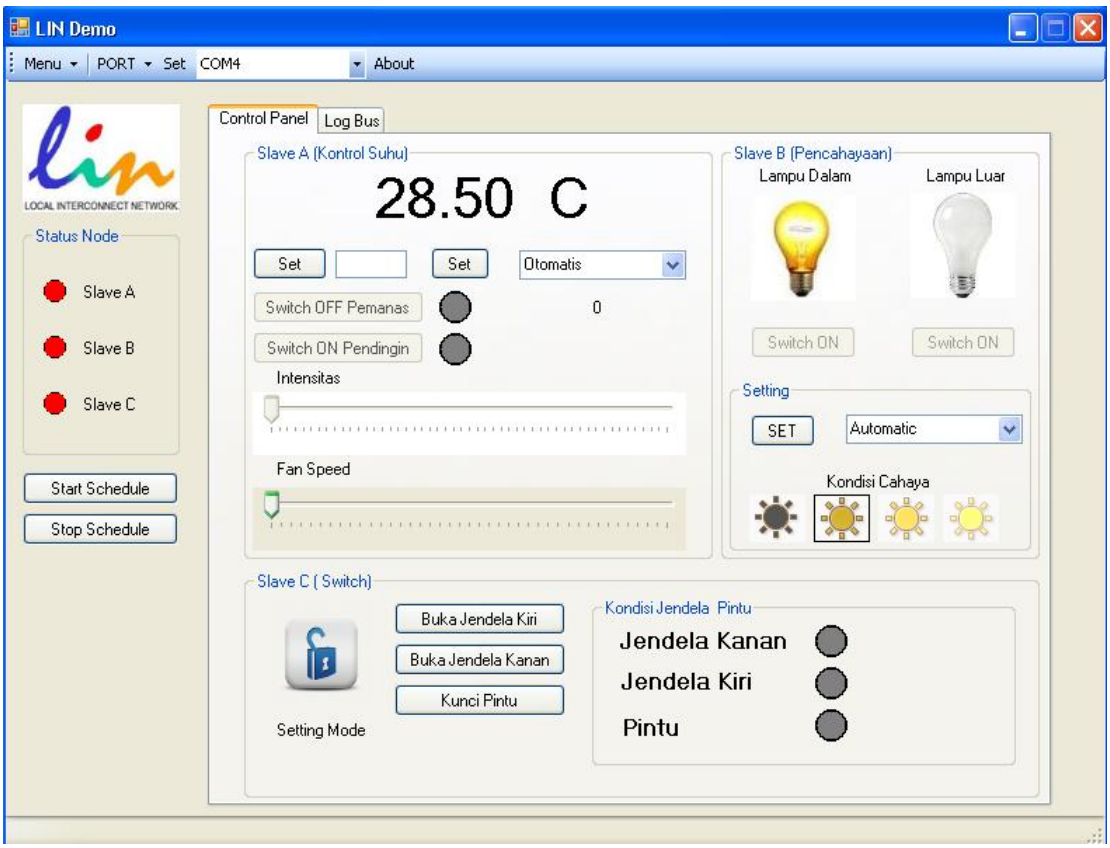
3.3. Maket Smart House

Maket *smart house* berukuran lebar 60 cm, tinggi 45 cm, dan panjang 90 cm. Terdapat satu pintu terletak di depan dengan sebuah pengunci elektronik. Jendela sebanyak dua buah dipasang pada sisi samping kanan dan samping kiri. Peletakan sensor cahaya terdapat pada bagian atap maket.



Gambar 5. Rancangan Maket Sistem

Untuk memudahkan pengguna dibuat antarmuka dengan pengguna seperti pada Gambar 6.

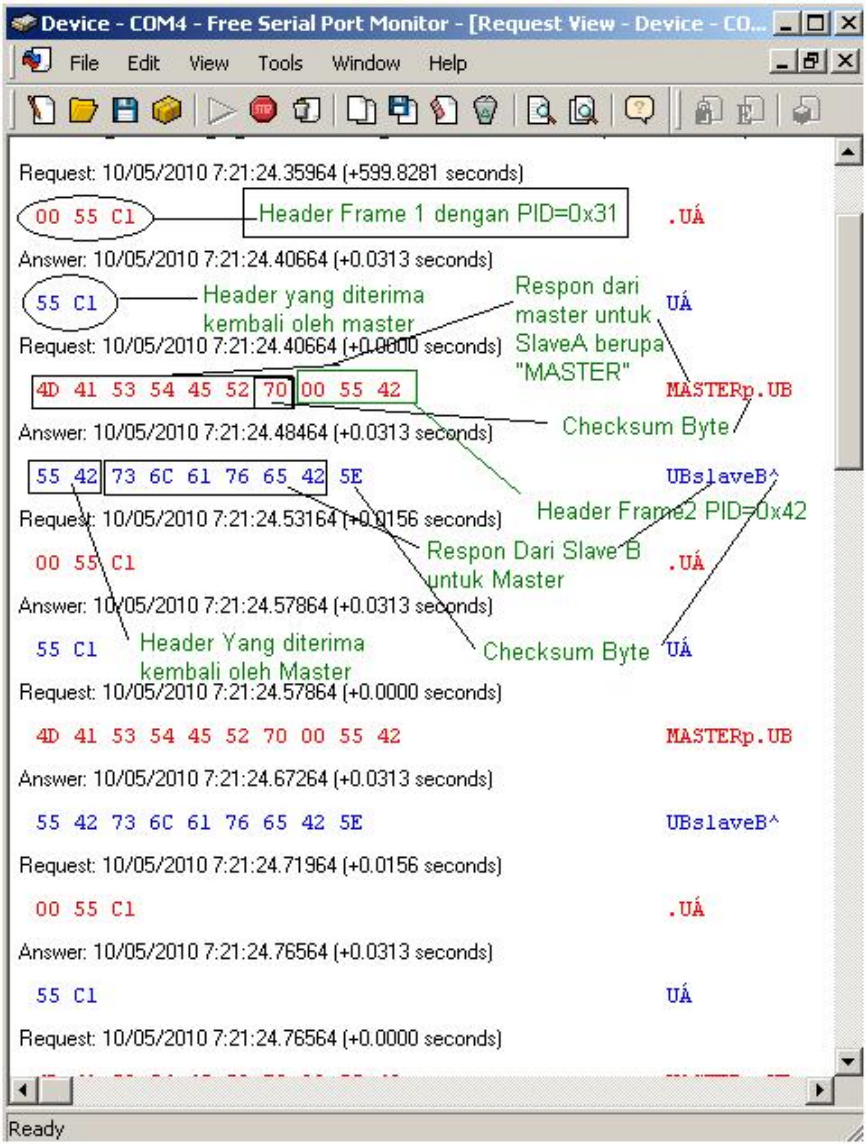


Gambar 6. Tampilan Aplikasi Node Master

4. Pengujian Sistem

Untuk mengukur kinerja dan tingkat keberhasilan sistem tersebut, dilakukan pengujian meliputi pengujian protokol LIN, pengujian modul elektronik disertai modul aktuator pada tiga node slave dan pengujian pengendalian modul pada setiap node slave melalui node master.

Gambar 7 menunjukkan aktivitas bus yang terdeteksi dengan aplikasi Serial Port Monitor. Selama 5 menit percobaan tidak terdapat kesalahan pengiriman maupun kesalahan penerimaan *response*. Pada *slave* A, yang berfungsi menampilkan *response* yang dikirim oleh *master* berupa string “MASTER”, juga tidak terdapat kesalahan. Hal ini ditunjukkan dengan string “MASTER” yang ditampilkan pada LCD tidak mengalami perubahan. Dari hasil pengujian diatas, dapat disimpulkan perancangan komunikasi antar *node* menggunakan protokol LIN telah berhasil.



Gambar 7. Aktivitas Bus Pada Komunikasi Antar Node

4.1. Pengujian Modul Slave A

4.1.1. Pengujian Pemanas *Mode Manual*

Pengujian dilakukan untuk mengetahui perbandingan peningkatan *delay* picu triac pada elemen pemanas dengan tingkat suhu yang dihasilkan oleh pemanas. Pengujian dilakukan dengan mengatur *level* intensitas pemanasan yang dilakukan melalui aplikasi *node master* dengan mengatur *trackbar* intensitas pada panel pengendali *slave* A saat sistem bekerja pada mode pengendalian suhu secara *manual*. Intensitas yang merupakan representasi sudut picu,

terbagi dalam 50 *level* pemanasan. Berdasarkan setiap level pemanasan mewakili *delay* picu sebesar 0.2 ms.

Delay picu diatur dengan pengiriman *header frame* ID 0x10 diikuti pengiriman *response* oleh *master* berupa nilai *level* pemanasan sesuai dengan nilai pada *trackbar*. *Response* dari *header* tersebut akan diterima oleh *slave* A dengan *action* berupa pengubahan nilai *delay* picu triac elemen pemanas. Pengujian setiap level pemanasan dilakukan selama waktu 5 menit dan dipantau kenaikan suhu setiap menitnya.

Dari hasil pengujian didapat bahwa untuk menaikkan suhu ruangan, delay picu maksimal yang diberikan sebesar 7,6 ms atau saat *trackbar* intensitas pemanasan diatur sebesar 12. Jika delay picu yang diberikan melebihi 7,6 ms, pemanas tidak mampu untuk menaikkan suhu ruangan. Hal ini terjadi karena daya diberikan kepada elemen ketika delay picu melebihi 7,6 ms tidak cukup untuk disebarkan keseluruh ruangan. Dengan daya maksimal yaitu saat tidak terdapat *delay* pemicuan triac (*level trackbar* diatur sebesar 50), rata-rata kenaikan suhu setiap menitnya adalah 0,96 ℃/menit. Dari hasil pengujian juga dapat dilihat bahwa perubahan suhu per menit selalu meningkat.

4.1.2. Pengujian Pendingin Ruangan

Pengujian dilakukan untuk mengetahui perbandingan besar *duty cycle* isyarat PWM yang dihasilkan oleh *slave* sebagai pengendali kecepatan kipas pendingin dengan tingkat perubahan suhu yang dihasilkan. Pengujian dilakukan dengan mengatur *level* pendinginan melalui aplikasi *node master*. *Level* pendinginan diatur melalui *trackbar* intensitas saat *slave* A bekerja pada mode pengendalian suhu secara *manual* dan kondisi pendingin aktif.

Intensitas pendinginan *slave* A diatur oleh *node master* dengan mengirimkan *header frame* ID 0x12 diikuti *response* dengan panjang 1 *byte* berupa nilai *level* pendinginan sesuai dengan nilai *trackbar* intensitas. *Response* akan diterima oleh *slave* A untuk mengubah nilai *register* OCR0. Nilai *register* OCR0 akan mengubah *duty cycle* isyarat PWM yang dihasilkan. Sesuai dengan perancangan pada bab III kenaikan setiap *level* intensitas mewakili kenaikan 5 nilai OCR0.

Pengujian setiap level pendinginan dilakukan selama 5 menit dan dipantau suhu keluaran setiap menitnya untuk mengetahui tingkat perubahan suhu yang dihasilkan. Berdasarkan hasil pengujian, kemampuan modul pendingin untuk menurunkan suhu dimulai saat level pendinginan

diatur pada nilai 22, yaitu saat *register* OCR0 bernilai 110. Dengan nilai OCR0 sama dengan 110 maka tegangan masukan kipas adalah: (110/255)* 16V=6,9 Volt.

Dari hasil pengujian juga dapat dilihat bahwa laju maksimal penurunan suhu per menit terjadi saat kipas pendingin mendapatkan tegangan masukan maksimal yaitu saat OCR0 bernilai 250 dengan laju 0,74 °C/menit. Pada proses pendinginan dengan level 1 dan level 5 didapat hasil penurunan suhu adalah negatif. Hal tersebut dikarenakan kipas pendingin berputar lambat sehingga kenaikan atau penurunan suhu dalam ruangan bukan merupakan hasil pengaruh dari kipas pendingin.

4.1.3. Pengujian Modul Pengendali Suhu Ruangan Ototomatis

Pengujian modul pengendalian suhu otomatis bertujuan untuk mengetahui kemampuan modul untuk menyesuaikan suhu ruangan sesuai tetapan yang diinginkan. Pengujian dilakukan dengan mengatur tetapan suhu melalui *node master* dengan nilai tetapan suhu sesuai dengan spesifikasi yang direalisasikan yaitu suhu 20 °C sampai 34 °C. Pengujian dilakukan selama 10 menit setiap variasi tetapan suhu dengan pemantauan setiap menit.

Untuk mengatur suhu yang diinginkan, *master* akan mengirim *header frame* dengan ID 0x0E dan *response* adalah tetapan suhu yang diinginkan saat *slave* A bekerja pada mode pengendalian suhu secara otomatis.Setelah *response* diterima oleh *slave* maka modul pemanas dan pendingin akan otomatis bekerja menyesuaikan tetapan suhu yang diinginkan.

Berdasarkan hasil pengujian, suhu keluaran minimal yang dapat diturunkan oleh modul pengendali suhu adalah sebesar 22,5 °C. Suhu tersebut dicapai dengan titik suhu awal adalah 28,5 °C dan dicapai dalam waktu 10 menit. Sementara kemampuan modul pengendali suhu untuk mencapai suhu maksimal adalah 34 °C. Pada pengujian dicapai dalam waktu 10 menit saat titik suhu awal adalah 22,5 °C.

4.2. Pengujian Modul Slave B

4.2.1. Pengujian Sensor Intensitas Cahaya

Pengujian sensor intensitas cahaya bertujuan untuk mengetahui perbandingan kondisi cahaya dengan rentang tegangan keluaran sensor. Tabel 1 menunjukkan nilai tegangan keluaran sensor pada waktu tertentu.

Tabel 1. Pengujian Sensor Cahaya

Waktu (WIB)	Tegangan Keluaran Sensor (Volt)	Kondisi Cahaya Pengukuran	Kondisi Cahaya Perancangan
07.00	4 - 4,3	Terang	Terang
09.00	4,5 – 4,95	Terang	Terang
12.00	4,9 - 4,95	Terang	Terang
15.00	4,4 - 4,95	Terang	Terang
17.00	1,1 - 1,5	Redup	Redup
18.00	0,2 - 0,5	Redup Sekali	Redup Sekali
20.00	0	Redup Sekali	Redup Sekali

4.2.2. Pengujian Pengendalian Lampu Terkontrol Master

Pengujian ini dilakukan untuk mengetahui apakah modul lampu pijar pada *slave* C dapat berkomunikasi secara benar dengan aplikasi pada *node master*. Pengujian dilakukan dengan menekan tombol untuk mematikan atau menghidupkan masing-masing lampu pada aplikasi *node master* saat *slave* B bekerja dalam mode *master control*. Proses ini dilakukan *master* dengan mengirim *header frame* dengan ID 0x14. *Response* sepanjang 1 *byte* data akan dikirim oleh *master* berupa nilai keadaan yang diinginkan. Kemudian indikator nyala lampu pada *panel* pengontrol akan berubah sesuai kondisi yang dideteksi oleh sensor pada *slave* B.

4.2.3. Pengujian Pengendali Lampu Otomatis

Pengujian bertujuan untuk mengetahui kondisi lampu berdasarkan intensitas cahaya diluar ruangan. Pengujian dilakukan dengan mengatur mode kerja *slave* B menjadi otomatis melalui *combobox.* Pada mode ini, tombol Switch Lampu nonaktif. Setelah penekanan tombol "SET", *master* mengirim *header frame* dengan ID 0x12 diikuti pengiriman *response byte* bernilai 0x04 kepada *slave* B. Setelah menerima *response* tersebut, *slave* B mengubah mode kerja pengendali lampu menjadi mode otomatis.

Indikator kondisi cahaya pada panel pengendali *slave* B berubah sesuai kondisi cahaya yang terdeteksi oleh sensor pada *slave* B. Data kondisi cahaya diperoleh *node master* melalui *frame* dengan ID 0x07. Indikator lampu panel pengendali berubah sesuai kondisi lampu dengan data yang diperoleh melalui *frame* dengan ID 0x08. *Slave* B akan mengatur kondisi lampu nyala/mati sesuai dengan kondisi cahaya pada luar ruangan.

4.3. Pengujian Modul Slave C

4.3.1. Pengujian Modul Driver Motor Pengendali Jendela

Pengujian ini bertujuan untuk mengetahui apakah modul *slave* C dapat berkomunikasi secara benar dengan aplikasi *node master*. Pengujian dilakukan dengan penekanan tombol pada aplikasi node master untuk memberi perintah kepada slave C. Untuk membuka atau menutup jendela digunakan frame dengan ID 0x17 diikuti oleh *response* yang dikirim oleh master berisi 1 byte data yang akan diterima oleh slave C. *Response* yang bernilai 0x01 akan mengubah posisi jendela kanan, sementara dengan response yang bernilai 0x02 akan mengubah posisi jendela kiri.

Pada saat *response* diterima, *slave* C mengubah posisi jendela berkebalikan dari posisi sebelumnya. Posisi jendela akan tertutup jika sebelumnya jendela pada posisi terbuka, sementara posisi jendela akan terbuka jika sebelumnya jendela pada posisi tertutup. Indikator posisi jendela akan berwarna merah jika jendela dalam kondisi menutup dan akan berwarna abu-abu dalam kondisi terbuka.

Untuk membuat posisi pintu terbuka/terkunci *master* berkomunikasi dengan *slave* C melalui *frame* dengan ID 0x18 yang diikuti *response byte*. Pada saat *response* diterima, *slave* C mengubah posisi kunci berkebalikan dari posisi sebelumnya. Pada saat posisi terkunci, *slave* C akan menarik batang *solenoid* sehingga posisi pengunci pintu terbuka. Sementara pada saat posisi terbuka, *slave* C akan mendorong batang *solenoid* sehingga pintu terkunci.

Pada panel *slave* C terdapat *picture box* untuk mengaktifkan dan menonaktifkan sistem pengaman, berupa *buzzer* yang terpasang pada *slave* C. Pada saat status *slave* C terkunci, peringatan pada aplikasi *node master* akan muncul jika jendela dan/atau pintu dalam kondisi terbuka dan *buzzer* pada *slave C* akan menyala.

5. Kesimpulan

Berdasarkan dari pengujian ini, diperoleh beberapa kesimpulan sebagai berikut:

1. Sistem *smart house* berprotokol LIN berhasil direalisasikan pada maket dengan satu *node master* dan tiga *node slave*. LIN yang berhasil dibuat memiliki *time base* optimal bernilai 47 ms dengan *baudrate* sebesar 19200 bps.
2. Pengaturan suhu dan pencahayaan ruangan serta pengendalian jendela, pengunci pintu dan sistem peringatan dapat dilakukan melalui aplikasi panel pengontrol *node master*.

Daftar Pustaka

1. LIN Specification Package Revision 2.1: http://www.lin-subbus.org/index.php?pid=8&lang=en&sid=b49535, 2008
2. LIN Overview Presentation Motorola: http://www.licm.fr/IMG/pdf/LIN.pdf. 2008