Rancang Bangun Prototipe Embedded System Untuk Kendali Kecepatan Putaran Motor DC

Design and Implementation the Prototype of Embedded System to Control the Rotation Speed of a DC Motor

Agusma Wajiansyah[1], Supriadi[2*]
[1,2] Jurusan Teknologi Informasi, Politeknik Negeri Samarinda
Jl. Cipto Mangun Kusumo, Kec. Samarinda Seberang, Kota Samarinda, Kalimantan Timur 75242
Agusma.wajiansyah@gmail.com[1], supriadi.polnes@gmail.com[2*]

Abstrak – *Motor DC adalah suatu perangkat yang mengubah energi listrik menjadi energi kinetik atau gerakan (motion). gerakan yang dihasilkan dalam bentuk putaran pada rotor dapat diukur dalam satuan putaran per menit (RPM). untuk beberapa applikasi seperti bidang robotic diperlukan pengendalian putaran untuk mendapatkan karakteristik yang spesifik. pengendalian motor DC dapat dilakukan pada processor utama pada sistem robotic, tetapi hal ini akan menambah beban kerja processor sehingga kebutuhan akan sumber daya (memori, kecepatan processor, task scheduling) menjadi besar. pada penelitian ini dilakukan perancangan dan pembuatan embedded system (ES) untuk tujuan kendali kecepatan putaran motor DC. Pada ES disiapkan jalur komuniksi untuk berinteraksi dengan menggunakan jaringan multidrop RS485. tahapan yang dilakuakan pada penelitian ini dimulai dengan menerapkan kendali PID pada ES, tuning kontroller PID dengan pendekatan Experimental-Based Heuristic, dan desain Protokol untuk Machine to machine communication. Hasil penelitian didapat perangkat ES dapat berinteraksi dengan 9 macam perintah sesuai dengan perencanaan. perintah tersebut terdiri dari perintah dengan jawaban dari ES dan perintah tanpa jawaban dari ES, dengan waktu respon rata-rata perintah sebesar 0.32 ms dan 5.7 ms. Dari sisi kendali kecepatan putaran motor DC digunakan metode kendali PID. Dari percobaan yang telah dilakukan didapatkan parameter PID Kp=2.5, Ki=0.1 dan Kd=0.05 dengan error stedy-state sebesar 1%.*

Kata Kunci*: Kendali Kecepatan, PID, RS485, Embedded system.*

Abstract – *A DC motor is a device that converts electrical energy into kinetic or motion energy. the resulting motion in the form of rotation on the rotor can be measured in units of revolutions per minute (RPM). for some applications such as the robotic field, rotation control is needed to obtain specific characteristics. DC motor control can be done on the main processor in the robotic system, but this will increase the processor workload so that the need for resources (memory, processor speed, task scheduling) becomes large. in this study the design and manufacture of embedded systems (ES) was carried out for the purpose of controlling the DC motor rotation speed. In the ES prepared communication lines to interact using the multidrop RS485 network. The stages carried out in this study began by applying PID control to ES, tuning the PID controller with an Experimental-Based Heuristic approach, and designing Protocols for Machine to Machine Communication. The results obtained by the ES device can interact with 9 kinds of commands in accordance with the plan. The command consists of commands with answers from ES and commands without answers from ES, with an average response time of commands of 0.32 ms and 5.7 ms.*

From the control side of the DC motor rotation speed used the PID control method. From the experiments that have been carried out, the parameters PID Kp = 2.5, Ki = 0.1 and Kd = 0.05 with error stedy-state of 1%.

***Keywords**: speed control, PID, RS485, Embedded system.*

1. Pendahuluan

Motor Listrik DC atau Motor DC adalah suatu perangkat yang mengubah energi listrik menjadi energi kinetik atau gerakan (*motion*). Motor DC ini juga dapat disebut sebagai Motor Arus Searah. Seperti namanya, DC Motor memiliki dua terminal dan memerlukan tegangan arus searah atau DC (Direct Current) untuk dapat menggerakannya. Motor DC ini menghasilkan sejumlah putaran per menit atau biasanya dikenal dengan istilah RPM (*Revolutions per minute*) dan dapat dibuat berputar searah jarum jam maupun berlawanan arah jarum jam apabila polaritas listrik yang diberikan pada Motor DC tersebut dibalikan. Kemudahan dalam pengendalian kecepatan dan arah putaran membuat motor DC banyak digunakan untuk applikasi yang memerlukan kecepatan putaran yang luas seperti pada applikasi robotik. Pengendalian pada motor DC meliputi pengendalian kecepatan putaran dan pengendalian arah putaran

Pada applikasi robotic, pengendalian motor DC dapat dilakukan pada processor utama pada sistem robotic, tetapi hal ini akan menambah beban kerja processor sehingga kebutuhan akan sumber daya (memori, kecepatan processor, *task scheduling*) menjadi besar. Salah satu metode yang dapat digunakan adalah dengan menambahkan beberapa processor yang akan menjalankan fungsi-fungsi khusus seperti pengendalian motor DC. Processor tersebut dikoneksikan secara master-slave, dimana processor master akan bertindak sebagai processor utama. Processor utama hanya menjalankan sistem kecerdasan robot yang akan memberikan output pada processor kendali motor DC agar didapat gerakan yang dinginkan. dengan menggunakan metode ini, didapati pada sistem robotic akan menggunakan beberapa perangkat embedded system untuk menunjang sistem robot secara keseluruhan. Seperti pada penelitian [1] penggunaan metode ini memberikan dampak yang cukup signifikan terhadap proses eksekusi program, terukur ferformance meningkat sebesar 7,3 %. Komunikasi *Master-slave processor* dapat diterapkan dengan menggunakan media I2C, SPI atau komunikasi *multidrop* RS485.

Penerapan RS-485 telah dilakukan oleh [2]. 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 penukuran 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. Berbagai jenis penerapan RS485 juga dapat dilihat pada [3]–[6]

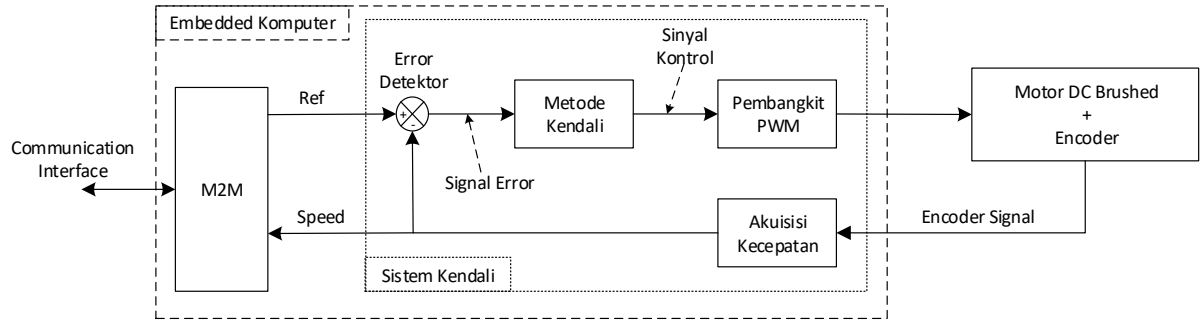
Menurut [7], *Embedded system* (ES) merupakan perangkat komputer yang dirancang khusus untuk applikasi tertentu. Sebuah ES memiliki kebutuhan tertentu dan melakukan tugas yang telah diset sebelumnya. Contoh aplikasinya antara lain adalah instrumentasi medik, *process control*, *automated vehicles control*, dan perangkat komunikasi. Penerapan kontrol PID (*proposinal-integral-Derivativei)* pada ES untuk tujuan kendali sering dijumpai seperti pada [8], [9] yang digunakan sebagai kendali robotic. Berbagai metode dapat diterapkan untuk dapat mengoptimalkan controller PID melalui penentuan nilai parameter PID agar dihasilkan *output* yang memiliki *error* sekecil mungkin.

Pada penelitian ini dilakukan perancangan dan pembuatan perangkat ES untuk mengendalikan Motor DC yang dilengkapi dengan media untuk berinteraksi antar perangkat berbasiskan komunikasi RS485. penelitian ini diharapkan dapat menunjang sistem robot yang akan dikembangkan setelahnya.

ISBN: 978-602-60581-1-9

2. Metode Penelitian

Block diagram sistem yang akan dirancang dan dibangun dalam penelitian ini ditunjukan pada Gambar 1. Kontroller Motor DC (MDC) terdiri dari sistem kendali dan perangkat *Machine to Machine* (M2M) *communication*. Sistem kendali bertugas untuk mempertahankan kecepatan MDC pada referensi (*Ref*) yang diinginkan, sedangkan M2M bertujuan sebagai media interaksi antara perangkat.

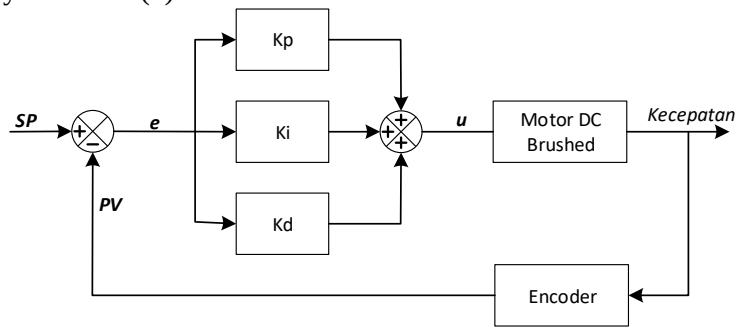


Gambar 1. Block diagram *embeded system* untuk kontroller Motor DC.

2.1. Kendali Kecepatan Motor DC dengan menggunakan Kontroller PID

Block diagram kendali kecepatan Motor DC ditunjukan pada Gambar 1. Salah satu masalah pada implementasi controller PID adalah bagaimana menetukan Konstanta Kontroller (Proposional,Integral, Differensial) dan menerapkannya secara diskrit pada ES.

Gambar 2 memperlihatkan salah satu model kendali PID yang tersusun secara parallel. Set Point (*SP*) merupakan nilai referensi (kecepatan) yang diinginkan terjadi pada output, sinyal error (*e*) merupakan selisih antara *SP* dan nilai actual (*PV*). Sinyal error ini diolah controller PID untuk mendapatkan sinyal kendali (*u*)



Gambar 2. Parallel PID model.

Algoritma controller PID pada Gambar 2 secar matematis dijabarkan pada persamaan (1):

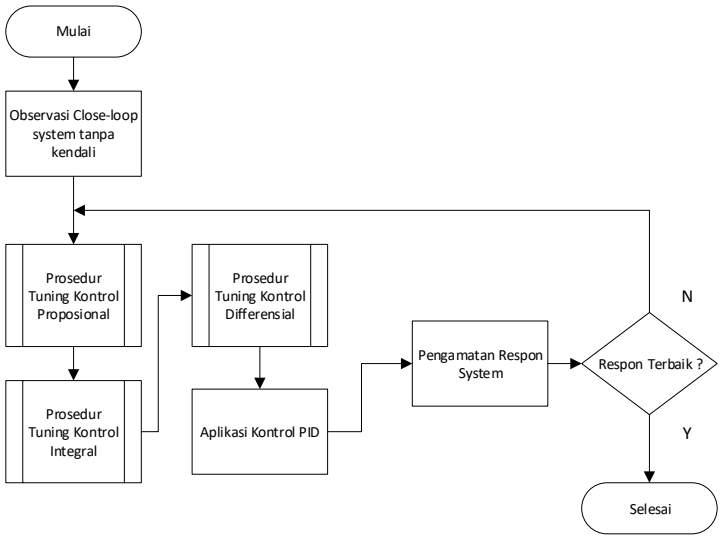
$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{d}{dt} e(t) \quad (1)$$

Pada Persamaan (1) dimana $u(t)$ adalah sinyal output kendali dalam domain waktu, $e(t)$ adalah sinyal error (selisih antara Set Point (*SP*) dan nilai actual (*PV*)) dalam domain waktu, K_p adalah Konstanta controller Proposional, K_i adalah Konstanta controller Integral, $K_d$adalah Konstanta controller Differensial.

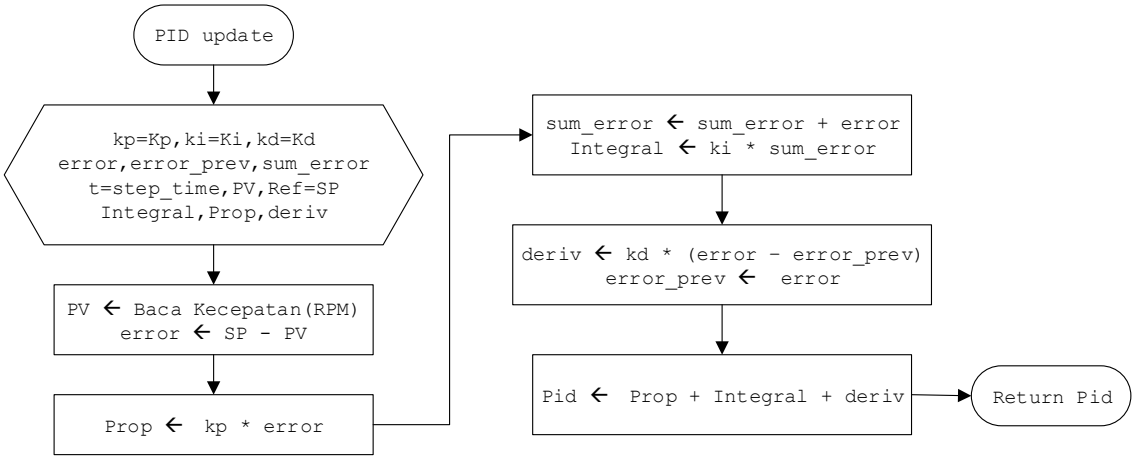
ISBN: 978-602-60581-1-9

2.2. Tuning PID controller dengan pendekatan *Experimental-Based Heuristic*

Pendekatan *Experimental-based Heuristic* (EBH) adalah pendekatan untuk pemecahan masalah dengan menggunakan metode praktis yang cukup signifikan untuk mencapai tujuan. Dengan kata lain, *Heuristic* adalah aturan praktis di mana proses penyelesaiannya bergantung pada aturan intuitif atau empiris [10]. Pada penelitian ini, pendekatan EBH digunakan untuk menemukan kombinasi dari parameter PID hingga ditemukan kombinasi yang cocok untuk Motor DC yang digunakan. Indicator ini dilihat dari respon sistem yang mendekati referensi yang dikehendaki.



Gambar 3. Tahapan pendekatan experimental-based heuristic yang digunakan.



Gambar 4. Flowchart algoritma PID.

Tahap awal pendekatan EBH dilakukan dengan melakukan observasi untuk mendapatkan karakteristik close-loop tanpa kendali dari MDC. Berikutnya dilakukan observasi karakteristik close-loop dengan kendali Proposional dari MDC, hal ini perlu dilakuakan berulang dengan penambahan nilai konstanta proposional (Kp) secara bertahap sehingga didapat sinyal error (e) *stedy-state* yang sekecil mungkin tetapi tidak membuat respon output berosilasi. Nilai Kp yang besar memungkinkan terjadi overshoot. Tahap selanjutnya menambahkan control Integral (Ki) secara bertahap sehingga didapatkan sinyal error (e) *stedy-state* yang lebih kecil dibanding percobaan hanya dengan kontrol Proposional saja dan mengurangi nilai overshoot yang mungkin terjadi. Hal yang sama juga dilakukan dengan control Differensial, penambahan konstnta kontrol

ISBN: 978-602-60581-1-9

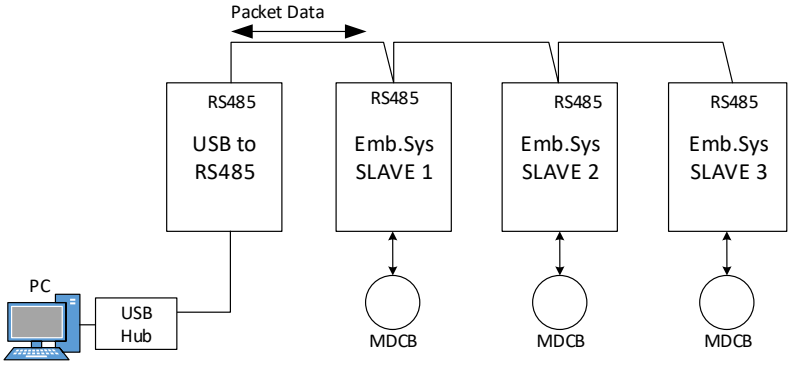
Diferrensial akan membuat respon sistem semakin *responsive.* Penambahan ini dilakukan hingga didapat respon sistem yang dapat diterima. Pendekatan EBH yang digunakan pada penelitian ini ditunjukan pada Gambar 3. Gambar 4 merupakan implementasi dari algoritma Kontrol PID.

Untuk mengetahui kecepatan putaran poros MDC, terlebih dahulu harus diketahui frekuensi yang dibangkitkan sensor encoder dan konstanta yang disebut cycle-per-revolution (CPR). Kecepatan putaran MDC dapat dihitung dengan persamaan (2). Dimana *n* adalah rotasi-per-menit (rpm), f adalah frekuensi sinyal encoder (Hz) dan *cpr* adalah *cycle-per-revolution*.

$$n = \frac{f}{cpr * 60} \tag{2}$$

2.3. Machine to machine comunication

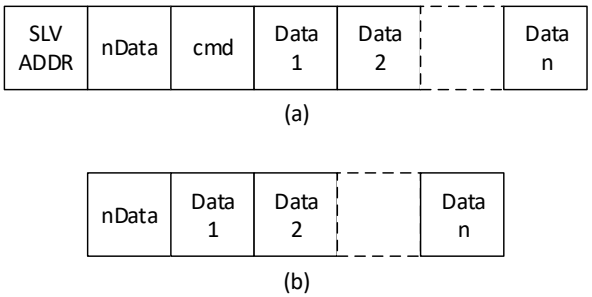
Machine to machine communication (M2MC) merupakan sarana untuk mengakses perangkat kendali MDC. Pada penelitian ini digunakan komunikasi multidrop RS485 dengan model protocol *master-slave*. Untuk menunjang ekperimental digunakan 3 buah perangkat MDC sebagai slave dan sebuah master yang berupa personal computer (PC).



Gambar 5. Ekperimental detail.

2.3.1. Paket Data

Fungsi utama dari M2MC adalah menyediakan protocol dalam proses pengiriman dan penerimaan data. Paket data yang digunakan terdiri dari dua jenis yaitu paket data dari master ke slave (Request) dan paket data dari slave ke master (Resposn). Gambar 6 merupakan visualisasi dari kedua jenis paket data tersebut.



Gambar 6. Paket data (a) Resques, (b) Respons.

ISBN: 978-602-60581-1-9

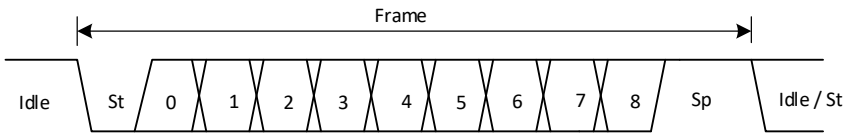
Pada paket request, SLV ADDR merupakan data pertama yang dikirim. Berisi alamat slave yang dituju. SLV ADDR memiliki panjang data sebesar 1 byte, pemberian alamat dimulai dari 01H sampai dengan FFH, dan 00H digunakan sebagai alamat broadcast. Data berikutnya adalah 'nData' yang memberikan informasi jumlah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

Pada byte 'cmd' memberikan informasi perintah yang harus dilakukan oleh slave. Lebar data 'cmd' sebesar 1 byte, sehingga dapat dibentuk sebanyah 256 perintah. Perintah dikelompokan menjadi 2 jenis yaitu perintah tanpa menunggu Respond dimulai dengan data 00H – 7FH dan perintah dengan Respons dimulai dengan nilai 80H - FFH. Dan 'Data 1', 'Data 2' sampai dengan 'Data n' merupakan parameter dari perintah. Setiap perintah dapat memiliki parameter atau tidak, tergantung dari keperluannya. Pada penelitian ini dibentuk 9 perintah yang terdiri dari 6 perintah tanpa respons, dan 3 perintah dengan respons dari slave. Perintah tersebut yaitu : Perintah *Set Kp*, *Set Ki*, *Set Kd* digunakan untuk konfigurasi konstanta PID, Perintah *Set Referensi* untuk memberikan nilai kecepatan Acuan. *Perintah Run* untuk mengaktifkan MDC, *Perintah Stop* untuk mematikan MDC. Perintah *Ping* untuk pengecekan konektifitas antara master dan Slave, *Perintah Baca parameter PID* digunakan melihat nilai parameter PID yang digunakan dan *perintah Baca nilai kecepatan actual* digunakan untuk melihat nilai actual kecepatan. untuk 'nData' berisi Jumlah data yang dikirim maximal sebanyah 8 Byte. Table 1 menunjukan nilai yang digunakan untuk perintah – perintah tersebut.

Tabel 1. Kode perintah pada kendali Motor DC.

Nilai (Hexadecimal)	Fungsi
0x01	Running
0x02	Stop
0x03	Set Kp
0x04	Set Ki
0x05	Set Kd
0x06	Set Referensi (Set Point)
0x81	Cek Konektifitas
0x82	Baca nilai konstanta PID
0x83	Baca nilai output aktual

Pada paket data Respons 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). Frame data pada USART ditunjukan pada Gambar 7. Setiap frame terdiri dari Star bit (St), 9 bit data (0 .. 8) dan Stop bit (Sp). Bit data (0..7) merupakan data yang dikirim (SLV ADDR, nData, cmd, Data 1, …) dan bit ke-8 merupakan flag yang akan bernilai '1' jika yang dirim adalah SLV ADDR dan sebaliknya.

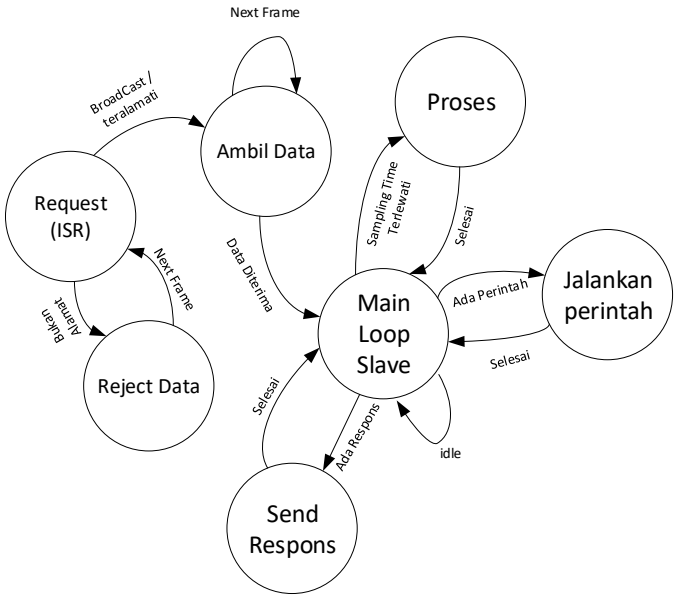


Gambar 7. Frame Data Serial.

ISBN: 978-602-60581-1-9

2.3.2. Rancangan Perangkat Lunak

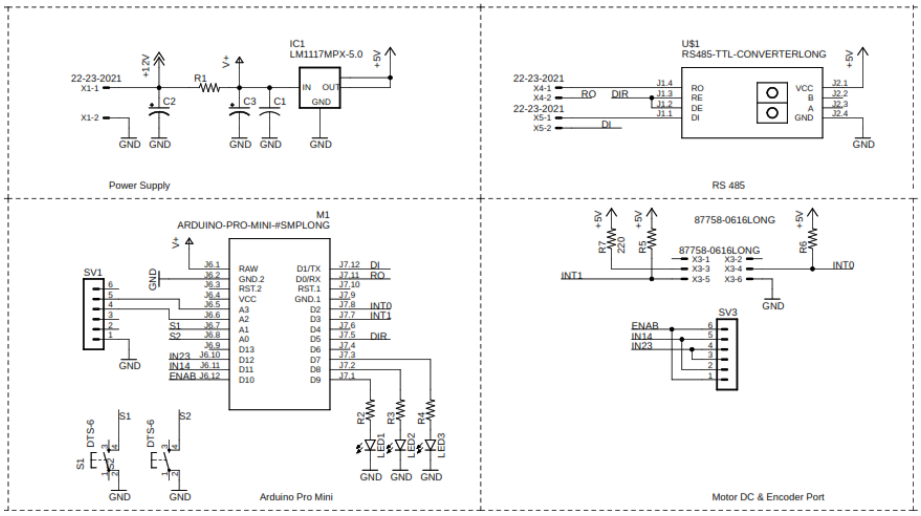
M2MC bertujuan untuk menyediakan protocol komunikasi. Saat proses penerimaan data, Pengendali Komuniksi melakukan pengecekan pada paket data yang datang. Jika alamat yang datang merupakan broadcast atau alamat yang sesuai maka akan dilakukan proses pembacaan data berikutnya. Proses ini ditunjukan pada Gambar 8. Penerimaan data bekerja secara interrupt dan juga menggunakan mode MPC (Multi processor Communication). Mode ini merupakan fitur komunikasi yang banyak dijumpai pada serial USART mikrokontroller. Data dari master hanya akan diterima selama data dikirim secara broadcast atau dengan menyertakan alamat slave yang dituju.



Gambar 8. State diagram perangkat slave.

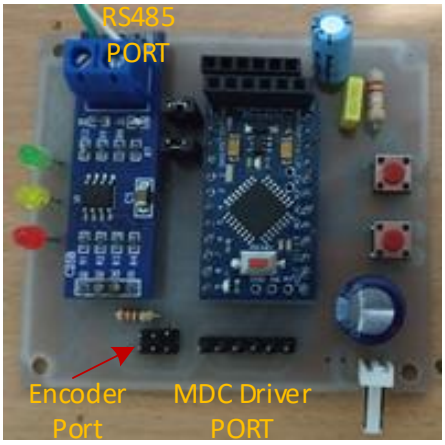
3. Hasil dan Pembahasan

Pada penelitian ini digunakan motor DC coreless dengan type 2342L012CR dari pabrikan Faulhaber. Untuk menggerakan motor DC digunakan driver L298 yang dikonfigurasi secara parallel. Gambar 9 dan Gambar 10 menunjukan Desain dan Prototipe dari Kontroller MDC.



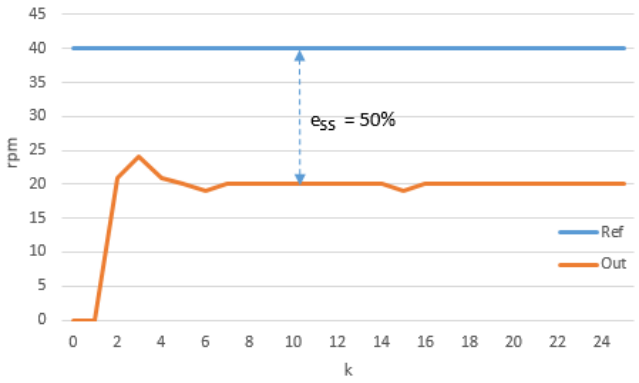
Gambar 9. Desain schematic *embedded system* kontroller Motor DC.

ISBN: 978-602-60581-1-9



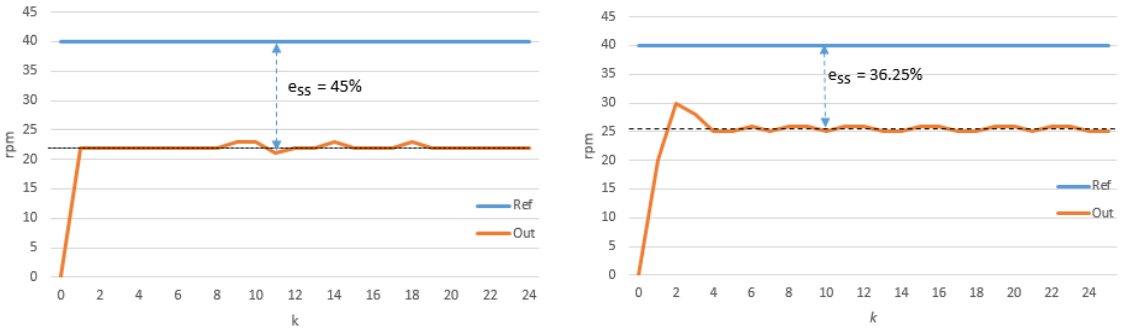
Gambar 10. hasil implementasi *embedded system* kontroller Motor DC.

Gambar 11 menunjukan respon sinyal yang didapat tanpa kendali (Kp=1, Ki=0, Kd=0) (garis berwarna orange), garis berwarna biru adalah siyal referensi. Dari Gambar tersebut dapat dilihat Error Stedy State (e_{ss}) sebesar 50%.



Gambar 11. Respon Close loop Tanpa Kontroller.

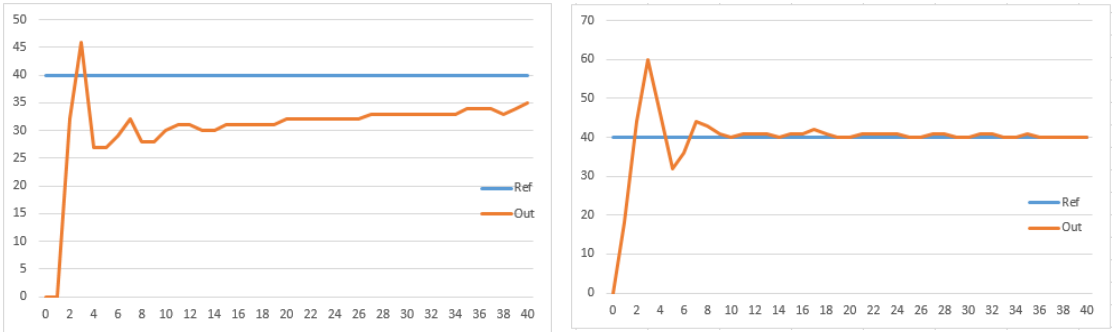
Tuning parameter PID dilakukan dengan menambahkan kendali proposional secara perlahan. Gambar 12 menunukan perubahan e_{ss} menjadi sebesar 45% saat nilai Kp adalah 1.5 dan e_{ss} menjadi sebesar 36.25% saat nilai Kp diubah menjadi 2.5.



Gambar 12. Respon Close Loop dengan controller Proposional (kiri) Kp = 1.5, (kanan) Kp = 2.5.

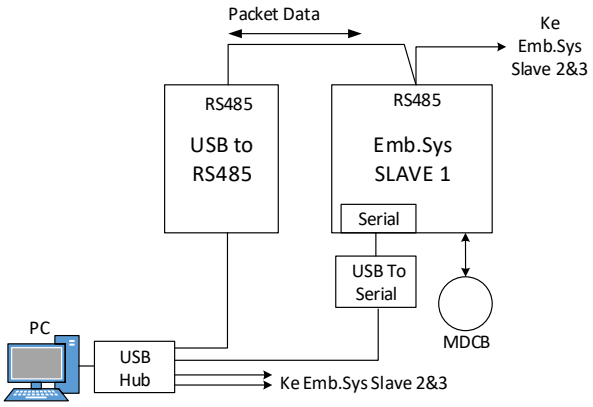
ISBN: 978-602-60581-1-9

Untuk mengurangi nilai e_{ss} ditambahkan Kontrol integral . Gambar 13 (kiri) adalah sinyal respon untuk control PI (Kp=2.5, Ki=0.1) sedangkan Gambar 13 (kanan) untuk sinyal respon dengan control PID (Kp=2.5,Ki=0.1,Kd=0.05).



Gambar 13. Respon Close loop (kiri) dengan controller PI, Kp=2.5;Ki=0.1,
(kanan) dengan control PID, Kp=2.5;Ki=0.1;Kd=0.05.

Pada bagian M2MC dilakukan pengukuran untuk melihat waktu respon setiap perintah yang diberikan. Untuk megukur waktu respon digunakan model yang ditunjukan pada gambar 14.



Gambar 14. Model pengukuran Waktu Respon.

Tabel 1. Waktu Respon Setiap Perintah pada Kendali MDC.

Perintah	Waktu Respon Rata-rata (ms)
Running	0.2
Stop	0.2
Set Kp	0.4
Set Ki	0.4
Set Kd	0.4
Set Referensi (Set Point)	0.3
Cek Konektifitas	2.8
Baca nilai konstanta PID	10
Baca nilai output aktual	4.3

ISBN: 978-602-60581-1-9

4. Kesimpulan

Pada penelitian ini telah diimplementasikan Embedded system untuk kendali kecepatan putaran motor DC. Untuk kendali kecepatan putaran motor DC tersebut digunakan Kontroller PID. Metode Pendekatan *Experimental-based Heuristic* digunakan untuk proses tuning parameter PID. Hasil percobaan didapatkan nilai Kp = 2.5, Ki=0.1, Kd=0.05 dengan Erorr Stedy State sebesar 1%. Untuk berinteraksi dengan Embedded system digunakan komunikasi RS485. Dengan 9 perintah interaksi, 6 diantaranya adalah perintah tanpa jawaban dari embedded system, dan 3 perintah dengan jawaban dari embedded system. Dari hasil percobaan waktu respons rata-rata untuk perintah tanpa jawaban sebesar 0.32 ms dan waktu respon untuk perintah dengan jawaban sebesar 5.7ms.

Ucapan Terima Kasih

Terimakasih kepada Moderen Computing Resert Center, Jurusan Teknologi Informasi untuk semua dukungannya dan kepada Pusat Penelitian dan Pengabdian Kepada Masyarakat Politeknik Negeri Samarinda atas atas dukungan keuangan.

Referensi

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] 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.

[6] L. Huang and J. Su, "A Context Aware Smart Classroom Architecture for Smart Campuses," *Appl. Sci.*, vol. 9, no. 1837, pp. 1–34, 2019.

[7] M. Wolf, *Computers as Components:Principles of Embedded Computing System Design*, Second Edi. Morgan Kaufmann, 2008.

[8] A. Najmurrokhman, I. Irfansyah, and A. Daelami, "Rancang Bangun Auto Balancing Robot Menggunakan Metode Kendali PID Design and Implementation of A Self-Balancing Robot Based on PID Control Method," *TELKA - Telekomun. Elektron. Komputasi dan Kontrol*, vol. 5, no. 1, pp. 15–23, 2019.

[9] H. Miftahul, F. Firdaus, and D. Derisma, "Pengontrolan Kecepatan Mobile Robot Line Follower Dengan Sistem Kendali PID," *TELKA - Telekomun. Elektron. Komputasi dan Kontrol*, vol. 2, no. 2, pp. 150–159, 2016.

[10] L. Magnani, Heuristic Reasoning vol. 16: Springer International Publishing Switzerland, 2015.

ISBN: 978-602-60581-1-9