Implementasi Multi-Mikrokontroller pada Mobile Robot sebagai Pengendali Kecepatan dengan Kendali PID

Agus Sehatman Saragih [1)]
[1)]Universitas Palangka Raya
Email : assaragih@gmail.com [1)]

Abstract

Mobile robot navigation system with wall follower method as the motion reference, using ultrasonic sensors in tracking rooms and avoid obstacles / hindrances. The number of ultrasonic sensors are used, can facilitate the robot navigation. PID control is used to control the motor speed mobile robot. Rotary encoder is a sensor that can be used to calculate the rotational speed of the actuator (dc motors).

Microcontroller as the main control has the workload to perform distance calculations (ultrasonic sensor) at the same speed calculation actuators / motors (rotary encoder). The calculation process in real-time and continuous. Application of multi-microcontroller applied to reduce the workload of the main microcontroller / Master. Slave microcontroller is used for the implementation of PID control.

Motor rotational speed by measuring the duty cycle motors pulse are less effective, it can be seen from the values of speed on each trial setting point analog PWM. Motor speed conversion can use another method such as a frequency-to-voltage conversion and use a voltage analog to digital converter (ADC) to get the present value (PV), so the data 8-bit digital-to-analog converter (DAC) directly compared with the 8-bit data analog to digital converter (ADC).

Key Words: mobile robot, PID control, multi-mikrocontroller

1. Pendahuluan

1.1 Latar Belakang

Pengembangan teknik otomasi pergerakan *mobile robot* untuk dapat beroperasi di dunia nyata sudah menjadi bahan penelitian bagi pengembangan *mobile robot* di dunia saat ini.

Sistem navigasi *mobile robot* dengan metode *wall follower* sebagai acuan gerak *mobile robot* dengan menggunakan sensor ultrasonik dalam menelusuri ruangan demi ruangan dan menghindari rintangan/ halangan. Jarak robot dengan halangan/rintangan akan menjadi acuan kontroler untuk melakukan gerakan yang dipresentasikan dengan gerakan aktuator/penggerak. Agar pergerakan robot sesuai dengan *setting* kontroler, maka dibutuhkan pergerakan yang kebal akan gangguan. Gangguan yang dapat terjadi misalnya adanya tanjakan maupun jalur yang yang tidak rata yang menyebabkan beban actuator /penggerak semakin besar. Kendali *PID* digunakan untuk mengontrol kecepatan motor *mobile robot* untuk mempertahankan kecepatannya.

Tujuan utama dari pengembangan robot adalah struktur modular, yang harus memungkinkan penggunaan berbagai interpolasi dan kontrol algoritma, tetapi juga penggunaan sensor eksternal yang mampu dan mencapai waktu sampling di kisaran ms [**1**]. Mikrokontroller sebagai pusat kendali memiliki beban kerja untuk melakukan perhitungan jarak (*ultrasonic sensor*) sekaligus perhitungan kecepatan aktuator/motor (*rotary encoder*). Proses perhitungan berlangsung secara *real-time* dan secara terus menerus. Penerapan multi-prosesor untuk meningkatkan respon jelajah robot sehingga semua sensor dapat selalu bekerja tanpa membebani mikrokontroller utama[2]. Implementasi logika fuzi pada mobil robot dengan menerapkan multi-mikrokontroller menggunakan 7 buah sensor ultrasonik[3].

Pada makalah ini akan digunakan 5 buah sensor ultrasonik sebagai acuan

gerak, dan menerapkan multi-mikrokontroller pada kontrol kecepatan aktuator/motor DC dengan kendali PID.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan di atas, maka dapat dirumuskan beberapa permasalahan sebagai berikut :

1. Bagaimana merancang bangun perangkat keras multi-mikrokontoller *mobile robot* agar dapat menerapkan pengaturan gerak dengan kendali *PID*.
2. Bagaimana merancang dan menerapkan program kendali *PID* berbasis mikrokontroler sebagai pengontrol kecepatan motor.

1.3 Tujuan dan Manfaat

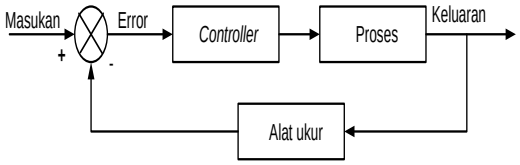
Tujuan penulisan ini adalah untuk Merancang bangun ulang perangkat keras multi-mikrokontoller *mobile robot* agar dapat menerapkan pengaturan gerak dengan kontrol *PID*.

.

1.4 Landasan Teori

1.4.1. Sistem Kendali Loop Tertutup (*close loop*)

Sistem kontrol *loop* tertutup adalah sistem kontrol yang sinyal keluarannya mempunyai pengaruh langsung pada aksi pengontrolan. Jadi sistem kontrol *loop* tertutup adalah sistem kontrol berumpan balik. Sinyal kesalahan penggerak, yang merupakan selisih antara sinyal masukan dan sinyal umpan balik (yang dapat berupa sinyal keluaran atau suatu fungsi sinyal keluaran dan turunannya), diumpankan ke kontroler untuk memperkecil kesalahan dan membuat agar keluaran sistem mendekati harga yang diinginkan, dengan kata lain, istilah *loop* tertutup berarti menggunakan aksi umpan balik untuk memperkecil kesalahan sistem. Gambar 2.1 menunjukan hubungan keluar masuk dari sistem kontrol *loop* tertutup.

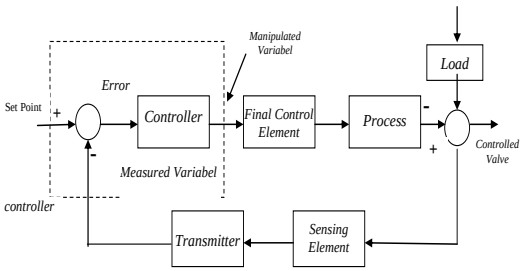


Sumber: Gunterus, Frans.

Gambar 1. Blok diagram sistem kendali *loop* tertutup.

Sistem kendali *loop* tertutup berupaya untuk mempertahankan keluaran sistem agar sama atau hampir sama dengan masukan acuan meskipun terjadi gangguan pada sistem. Jika terjadi perubahan pada nilai masukan refrensi dan keluaran sehingga keluaran akan kembali sesuai dengan masukan referensi.

1.4.2. Elemen Sistem Kendali



Sumber: Gunterus, Frans.

Gambar 2. Diagram kotak sistem pengendalian otomatis.

Keterangan [4]:

1. Proses (*process*) adalah tatanan peralatan yang mempunyai suatu fungsi tertentu (Gunterus, Frans., 1997:1-15);
2. *Controlled Variable* adalah besaran atau *variable* yang dikendalikan. Besaran pada diagram kotak juga disebut keluaran proses atau proses *variable* (Gunterus, Frans., 1997:1-15);
3. *Manipulated Variable* adalah masukan dari suatu proses yang dapat dimanipulasi atau diubah-ubah besarnya agar proses *variable* atau *controlled variable* besarnya sama dengan *set point* (Gunterus, Frans., 1997:1-15);
4. *Sensing Element* adalah bagian paling ujung suatu sistem pengukuran (*measuring system*). Bagian ini juga

disebut sensor atau *primary element* (Gunterus, Frans., 1997:1-15);

5. *Transmitter* adalah alat untuk membaca sinyal *sensing element* dan mengubahnnya menjadi sinyal yang dapat dimengerti oleh *controller* (Gunterus, Frans. 1997:1-15);
6. *Measured Variable* adalah sinyal yang keluar dari *transmitter* (Gunterus, Frans., 1997:1-15);
7. *Set Point* adalah besar proses variabel yang dikehendaki (Gunterus, Frans., 1997:1-15);
8. *Error* adalah selisih antara *set point* dikurangi *measured variable* (Gunterus, Frans., 1997:1-15);
9. *Controller* adalah elemen yang mengerjakan tiga dari empat tahap langkah pengendalian, yaitu membdaningkan, menghitung dan mengeluarkan sebuah perintah (Gunterus, Frans., 1997:1-15);
10. *Final Control Element* adalah bagian akhir dari instrumentasi system pengendalian (Gunterus, Frans., 1997:1-15);

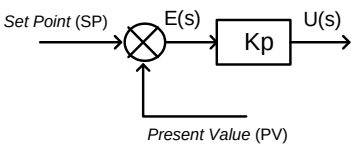
A. Pengendali Proportional (P)

Kontroller Proportional memiliki keluaran yang sebanding/*proportional* dengan besarnya sinyal kesalahan [5]. Secara lebih sederhana di wujudkan dalam :

$$u(t) = K_p.e(t) \text{.....(1)}$$

$$\frac{U(s)}{E(s)} = K_p \text{.....(2)}$$

K_p = Konstanta proporsional.



Sumber: Ogata; Teknik Kontrol Automatik.

Gambar 3. Diagram blok proporsional.

Dalam menggunakan pengendali proporsional harus memperhatikan ketentuan-ketentuan sebagai berikut:

1. Kalau nilai Kp kecil, maka kontroler proporsional hanya mampu melakukan koreksi kesalahan yang kecil , sehingga akan menghasilkan respon sistem yang lambat.
2. Kalau nilai Kp dinaikkan, maka respon sistem menunjukkan semakin cepat mencapai keadaan mantapnya. Namun jika nilai Kp diperbesar sehingga mencapai harga yang berlebihan dan akan mengakibatkan sistem bekerja tidak stabil atau respon system akan berisolasi.

B. Pengendali Integral (I).

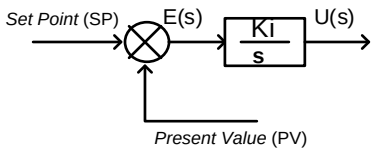
Ciri-ciri pengendali[5], yaitu:

1. Pengendali integral berfungsi menghasilkan respon sistem yang memiliki kesalahan keadaan mantap nol.
2. Pengendali Integral memiliki karakteristik seperti halnya sebuah integral. Keluaran kontroler sangat dipengaruhi oleh perubahan yang sebanding dengan nilai kesalahan.

Persamaan yang menggambarkan pengendali Integral, yaitu:

$$u(t) = K_i \int_0^t e(t)dt \text{.....(3)}$$

$$\frac{U(s)}{E(s)} = \frac{K_i}{s} \text{.....(4)}$$



Sumber: Ogata; Teknik Kontrol Automatik.

Gambar 4. Diagram blok pengendali integral.

Pengendali integral memiliki beberapa karakteristik sebagai berikut:

1. Keluaran kontroler membutuhkan selang waktu tertentu, sehingga kontroler integral cenderung memperlambat respon.
2. Ketika sinyal berharga nol, keluaran controller akan bertahan pada nilai sebelumnya.
3. Jika sinyal kesalahan (*error*) tidak berharga nol maka keluaran akan menunjukkan kenaikan atau

penurunan yang dipengaruhi oleh besarnya kesalahan dan nilai K_i yang berharga besar akan mempercepat hilangnya *offset*. Tetapi semakin besar nilai konstanta K_i akan mengakibatkan peningkatan osilasi dari sinyal keluaran kontroler.

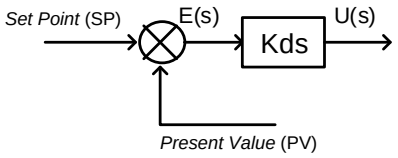
C. Pengendali Diferensial

Keluaran pengendali differensial memiliki sifat seperti halnya suatu operasi derivatif. Perubahan yang mendadak pada masukan kontroler akan menyebabkan perubahan yang besar dan cepat [5]. Persamaan pengendali diferensial, yaitu:

$$u(t) = K_d \frac{de}{dt} \text{ .....(5)}$$

$$\frac{U(s)}{E(s)} = K_d s \text{ .....(6)}$$

Diagram blok yang meng-gambarkan hubungan antara sinyal kesalahan dengan keluaran kontroler.



Sumber : Ogata; Teknik Kontrol Automatik.

Gambar 5. Diagram blok pengendali diferensial

Karakteristik pengendali diferensial adalah sebagai berikut[5]:

1. Kontroler ini dapat menghasilkan keluaran bila tidak ada perubahan pada masukannya (berupa sinyal kesalahan).
2. Jika sinyal kesalahan berubah terhadap waktu, maka keluaran yang dihasilkan kontroler tergantung pada nilai Kd dan laju perubahan sinyal kesalahan.
3. Kontroler diferensial mempunyai karakter untuk mendahului sehingga mampu menghasilkan koreksi yang signifikan sebelum pembangkit kesalahan menjadi besar.

Kontroler PID *(Proportional-Integral-Derivative)* merupakan kombinasi dari tiga jenis kontroler. Jika masing-masing dari ketiga jenis kontroler tersebut berdiri sendiri, maka hasil yang dicapai kurang bagus. Sebab, masing-masing memiliki kelebihan dan kelemahan sendiri-sendiri. Dikombinasikannya ketiga jenis kontroler tersebut menjadi satu sistem kontrol tunggal, diharapkan mampu memberikan kontribusi dari kelebihan masing-masing.

Kontrol proporsional adalah suatu penguat linier yang dapat diatur penguatan-nya. Hubungan antara keluaran kontroler m(t) dan sinyal kesalahan e(t) adalah;

$$m(t) = K_p\, e(t) \quad (2.1)$$

dengan: K_p adalah penguatan proporsional;

$m(t)$ adalah keluaran kontrol
$e(t)$ adalah sinyal kesalahan

Kontrol proporsional integral adalah merupakan perubahan dari keluaran kontrol integral m(t), berubah dengan fungsi waktu yang sebanding dengan sinyal kesalahan. Hubungan antara keluaran kontroler m (t) dan sinyal kesalahan e(t) adalah;

$$m(t) = K_p\, e(t) + \frac{K_p}{T_i} \int_0^t e(t)dt \quad (2.2)$$

dengan: K_p adalah gain proporsional
T_i adalah waktu integral.

Tetapan waktu integral T_i mengatur aksi kontrol integral, sedangkan K_p mem-perkuat bagian proporsional maupun bagian integral dari aksi kontrol. Kebalikan dari tetapan waktu integral T_i disebut laju reset. Laju reset adalah banyaknya pengulangan bagian proporsional dari aksi pengontrolan per detik. Kontrol proporsional derivatif didefinisikan;

$$m(t) = K_p\, e(t) + K_p T_d \frac{de(t)}{dt} \quad (2.3)$$

dengan: K_p adalah gain proporsional
T_d adalah tetapan waktu derivatif

Kontrol derivatif, sering disebut kontrol laju *(rate control)*, karena besar

keluaran kontroler sebanding dengan laju perubahan sinyal kesalahan. Tetapan waktu turunan T_d adalah selang waktu bertambah majunya respon kontrol proporsional yang disebabkan oleh aksi laju *(rate action)*.

Kontroler proporsional integral derivatif (PID) adalah gabungan kontrol proporsional, kontrol integral, dan kontrol turunan. Gabungan kontrol ini mempunyai keunggulan dalam memperbaiki kesalahan sinyal dibandingkan dengan masing-masing dari tiga kontrol tersebut. Persamaan kontroler PID dapat diberikan sebagai berikut:

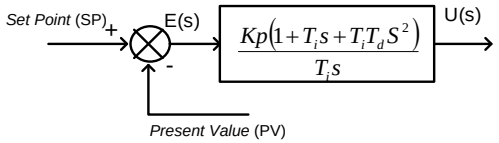
$$m(t) = K_p e(t) + \frac{K_p}{T_i}\int_0^t e(t)dt + K_p T_d \frac{de(t)}{dt} \quad (2.4)$$

Untuk memenuhi sistem yang diinginkan, maka ketiga parameter PID harus ditetapkan secara optimal. Ada beberapa metode untuk menentukan parameter tersebut di antaranya adalah metode coba-coba *(cut dan try method)*, metode Ziegler-Nichols dan metode tanggapan tangga *(step respons)*.
Didefinisikan dengan persamaan:

$$u(t) = K_p.e(t) + K_p.T_d.\frac{de}{dt} + \frac{K_p}{T_i}\int_0^t e(t).dt \quad ....(2.5)$$

$$\frac{U(s)}{E(s)} = K_p\left(1 + T_d s + \frac{1}{T_d s}\right) .....(2.6)$$

Diagram blok pengendali PID ditunjukkan oleh gambar 1.6.



Sumber: Ogata; Teknik Kontrol Automatik.

Gambar 6. Diagram blok pengendali PID.

Pengendali PID menggabungkan kelebihan pengendali P, I dan D, yaitu sebagai berikut:

- Pengendali P : untuk memperbaiki respon transien.
- Pengendali I : untuk menghilangkan *error steady state*.
- Pengendali D : memberikan efek redaman.

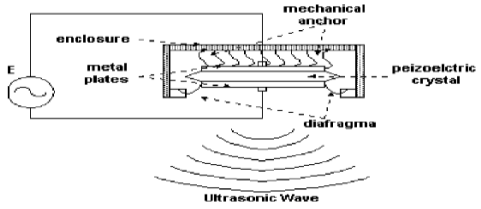
Sumber: (Gunterus, Frans. 1997:1-15).

1.4.3 Sensor Ultrasonik

Sensor ultrasonik adalah sensor yang bekerja berdasarkan prinsip pantulan gelombang suara dan digunakan untuk mendeteksi keberadaan suatu objek tertentu di depannya, frekuensi kerjanya pada daerah di atas gelombang suara dari 40 KHz hingga 400 KHz.

Sensor ultrasonik terdiri dari dari dua unit, yaitu unit pemancar dan unit penerima. Struktur unit pemancar dan penerima sangatlah sederhana, sebuah kristal *piezoelectric* dihubungkan dengan mekanik jangkar dan hanya dihubungkan dengan diafragma penggetar. Tegangan bolak-balik yang memiliki frekuensi kerja 40 KHz – 400 KHz diberikan pada plat logam. Struktur atom dari kristal *piezoelectric* akan berkontraksi (mengikat), mengembang atau menyusut terhadap polaritas tegangan yang diberikan, dan ini disebut dengan efek *piezoelectric*.

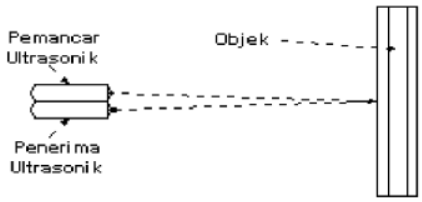
Kontraksi yang terjadi diteruskan ke diafragma penggetar sehingga terjadi gelombang ultrasonik yang dipancarkan ke udara (tempat sekitarnya), dan pantulan gelombang ultrasonik akan terjadi bila ada objek tertentu, dan pantulan gelombang ultrasonik akan diterima kembali oleh oleh unit sensor penerima. Selanjutnya unit sensor penerima akan menyebabkan diafragma penggetar akan bergetar dan efek *piezoelectric* menghasilkan sebuah tegangan bolak-balik dengan frekuensi yang sama. Untuk lebih jelas tentang prinsip kerja dari sensor ultrasonik dapat dilihat pada Gambar 1.7 berikut.



Gambar 7. Prinsip kerja sensor ultrasonik

Besar amplitudo sinyal elekrik yang dihasilkan unit sensor penerima tergantung

dari jauh dekatnya objek yang dideteksi serta kualitas dari sensor pemancar dan sensor penerima. Proses *sensing* yuang dilakukan pada sensor ini menggunakan metode pantulan untuk menghitung jarak antara sensor dengan objek sasaran. Jarak antara sensor tersebut dihitung dengan cara mengalikan setengah waktu yang digunakan oleh sinyal ultrasonik dalam perjalanannya dari rangkaian Tx sampai diterima oleh rangkaian Rx, dengan kecepatan rambat dari sinyal ultrasonik tersebut pada media rambat yang digunakannya, yaitu udara. Prinsip pantulan dari sensor ulrasonik ini dapat dilihat pada Gambar 1.8.



Gambar 8. Prinsip pemantulan gelombang ultrasonik

Waktu dihitung ketika pemancar aktif dan sampai ada masukan dari rangkaian penerima dan bila pada melebihi batas waktu tertentu rangkaian penerima tidak ada sinyal masukan maka dianggap tidak ada halangan di depannya.

1.4.4 Mikrokontroller AVR ATMega 16

AVR merupakan seri mikrokontroler CMOS 8-bit buatan Atmel, berbasis arsitektur RISC (*Reduced Instruction Set Computer*). Hampir semua instruksi dieksekusi dalam satu siklus *clock*. AVR mempunyai 32 register *general-purpose*, *timer/counter* fleksibel dengan mode *compare, interrupt internal* dan *eksternal*, serial UART, *programmable Watchdog Timer, dan mode power saving*. Mempunyai ADC dan PWM internal. AVR juga mempunyai *In-System Programmable Flash on-chip* yang mengijinkan memori program untuk diprogram ulang dalam sistem menggunakan hubungan serial SPI. Atmega16 adalah mikrokontroler CMOS 8-bit daya-rendah berbasis arsitektur RISC yang ditingkatkan.

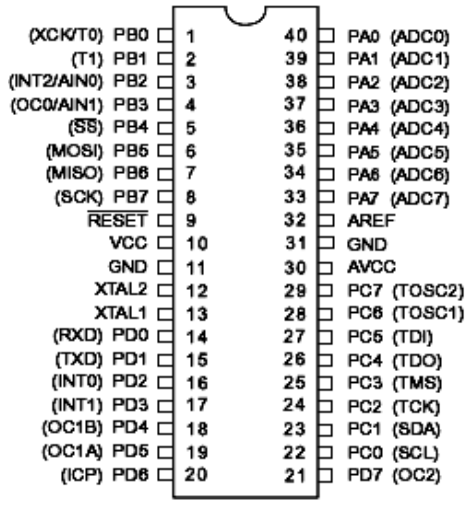
Kebanyakan instruksi dikerjakan pada satu siklus clock, Atmega16 mempunyai *throughput* mendekati 1 MIPS per MHz membuat disainer sistem untuk mengoptimasi komsumsi daya versus kecepatan proses.

Beberapa keistimewaan dari AVR ATmega16 antara lain:

1. Fitur AVR ATmega 16
   - 130 *Powerful Instructions – Most Single Clock Cycle Execution*
   - 32 x 8 *General Purpose Fully Static Operation*
   - Mencapai 16 MIPS *Throughput* pada 16 MHz
   - *On-chip 2-cycle Multiplier*
2. Data dan *Non-volatile Program Memory*
   - 8K Bytes flash *In-System Self-Programmable*
   - *Optional Boot Code Section dengan Independent Lock Bits*
   - 512 Bytes EEPROM
   - 512 Bytes Internal SRAM
   - Pemprograman pengunci untuk keamanan software
3. Fitur Peripheral
   - Dua Timer/Counters 8-bit dengan *Separate Prescalers* dan Mode pembanding
   - Dua Timer/Counters 8-bit dengan *Separate Prescalers* dan Mode pembanding
   - Satu Timer/Counter 16-bit dengan *Separate Prescaler*, Mode pembanding, dan Capture Mode (penangkap trigger)
   - *Real Time Counter* dengan *Separate Oscillator*
   - Empat saluran PWM
   - 8-channel, 10-bit ADC
   - *Byte-oriented Two-wire Serial Interface*
   - *Programmable* Serial USART

4. Fitur khusus Mikrokontroler
   - Rangkaian Power-on Reset dan deteksi *Brown-out* yang dapat diprogram
   - Kalibrasi Oscillator RC internal
   - Sumber interrupsi External dan Internal
   - Enam Mode Sleep: *Idle*, *ADC Noise Reduction, Power-save, Power-down, Stdanby dan Extended Stdanby*
5. I/O dan Kemasan
   - 32 *Programmable* saluran I/O
   - 40-pin PDIP, 44-lead TQFP, 44-lead PLCC, dan 44-pad MLF
6. Tegangan kerja
   - 2,7 – 5,5V untuk Atmega16L
   - 4,5 – 5,5V untuk Atmega16

Pin-pin pada ATmega16 dengan kemasan 40-pin DIP (*dual in-line package*) ditunjukkan oleh Gambar 1.9. Guna memaksimalkan performa dan paralel, AVR menggunakan arsitektur Harvard (dengan memori dan bus terpisah untuk program dan data).

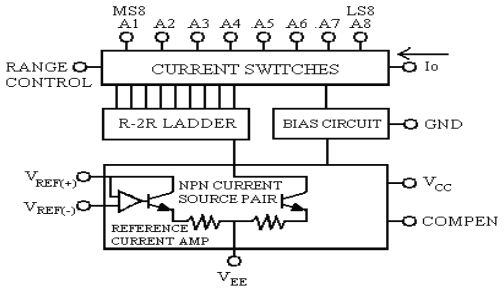


Gambar 9. Pin-pin atmega16 kemasan 40-pin

1.4.5 DAC (*Digital to Analog Converter*)

Digital Analog Convertion adalah suatu rangkaian elektronika yang berfungsi untuk merubah besaran digital menjadi besaran analog. Rangkaian ini diperlukan pada saat rangkaian memberikan keluaran berupa sinyal digital kemudian diubah menjadi sinyal analog. Rangkaian digital ini juga dapat digunakan sebagai alat kontrol yang dapat mengoperasikan parameter-parameter tegangan maupun arus kedalam bentuk analog. Dalam pembahasan skripsi ini jenis IC DAC yang dipakai adalah jenis 0808.

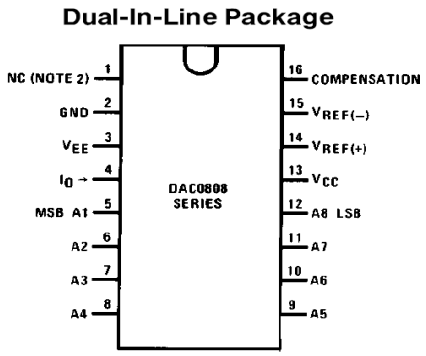
IC DAC 0808 adalah jenis D/A *Convertion* yang mempunyai 8 bit masukan dan dibangun dengan metode konversi rangkaian R-2R *ladder*. Masukan dari data biner IC ini sesuai dengan gerbang (*gate*) TTL, sehingga keluaran dari port standart melalui PIO, IC DAC 0808 ini mempunyai konfigurasi pena dan diagram blok seperti pada Gambar 1.10 dan 1.11 sebagai berikut:



Sumber: *Datasheet*

Gambar 10. Konversi rangkaian r-2r ladder

Pada dasarnya hubungan DAC ke sistem minimum adalah merupakan hubungan yang sederhana, karena D/A *convertion* adalah merupakan peralatan keluaran dimana penulisan data tidak lagi memerlukan kontrol yang khusus.



Sumber: *Datasheet*

Gambar 11. Diagram blok konfigurasi DAC 0808

DAC 0808 disuplai pada Vcc dengan tegangan +10 volt untuk memberikan referensi pada tegangan masukan yang berlevel TTL. Sedangkan mengenai jangkauan tegangan keluaran ditentukan oleh V_{ref} (+) dan V_{ref} (-), dengan persamaan keluarannya adalah:

$V_{out} = V_{ref} x (A1/2 + A2/4 + A3/8 + ...+A8/256$

Kisaran mengenai keluaran DAC 0808 dapat diatur sesuai dengan catu daya tegangan referensi yang telah diberikan. Pedoman keluaran minimum juga dapat kita atur atau diset dengan membandingkan tegangan pada referensi.

2.5 Vexta Motor

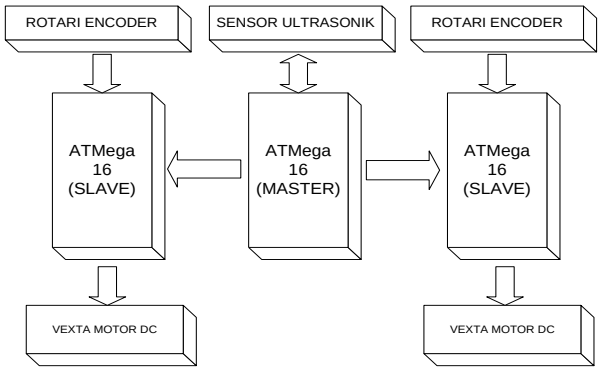
Sumber: Sutojo, dkk. 2010[4]

2. Pembahasan

2.1 Perancangan Perangkat Keras

Secara umum perancangan sistem dari navigasi robot dengan kendali PID terdiri dari masukan (*input*), kontroler dan keluaran (*output*). Masukan (*input*) terdiri beberapa sensor antara lain sensor ultrasonik dan sensor penyandi *(Rotary Encoder)*. Kontroler yang digunakan terdiri dari tiga buah mikrokontroler AVR ATMega 16 yang terdiri dari satu master dan dua slave. Dari sisi keluaran (*output*) terdapat driver motor, pada driver motor dibagi menjadi 2 untuk driver motor utama menggunakan modul dari VEXTA Driver Brushless motor DC dan untuk mengatur kecepatan kita harus menggunakan rangkaian DAC *(Digital to analog converter)* karena keluaran dari mikrokontroller berupa data digital 8 bit (0-255) sedangkan pada driver motor menggunakan input tegangan (analog) 0-5 V.

Diagram Blok sistem yang akan dibuat terlihat dalam gambar 12:



Gambar 12. Diagram blok sistem

2.1.1 Sensor Ultrasonik

Sistem sensor ultrasonik digunakan sebagai masukan dari proses pengontrolan robot terbagi atas dua bagian, yaitu untuk perangkat keras dan lunak (*controller*).

Pin sensor ultrasonik GND, 5V, SIG.

Gambar 13. Sensor ultrasonik

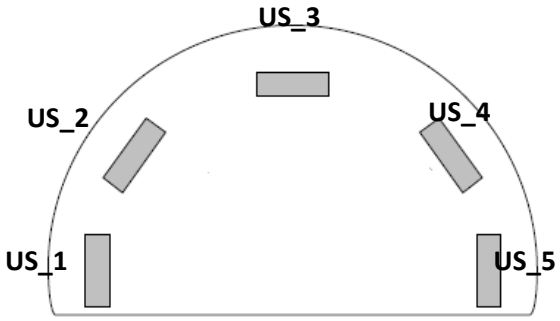
Sensor ultrasonik pada gambar 12 merupakan modul *ping parallax*. Sensor ultrasonik terdiri dari bagian pemancar dan penerima. Sensor *ping parallax* merupakan sensor yang dapat mengukur jarak, pengukuran dilakukan dengan mengirimkan gelombang PWM ultrasonik dengan frekuensi 40 KHz dengan kecepatan 344 m/s (1 cm/29,07 us) kemudian *ping* akan menerima pantulan (*duty cycle*). Waktu pantulan akan dihitung dengan menggunakan fasilitas timer dari mikrokontroler yaitu waktu selama pin SIG dalam kondisi high atau 1. Jarak robot dan halangan dapat diperoleh dengan rumus:

$$Jarak(S) = Kecepa\tan(V) \times \frac{Waktu(t)}{2}$$

$$S\,cm = \frac{1cm}{29,7} \times \frac{t\,us}{2}$$

$$S\,cm = 1cm \times \frac{t\,us}{58,14us}$$

Terdapat 5 pasang sensor ultrasonik dan semuanya terdapat pada bagian depan. Hal ini untuk menjangkau semua halangan yang ada di sekitar robot. Ultrasonik bagian sisi kiri dan kanan bertugas menjaga jarak antar dinding, sedang ultrasonik depan menjaga dari tabrakan. Desain Sensor ditunjukkan pada gambar 2.3.



Gambar 14. Posisi sensor ultrasonik

Sensor ultrasonik ini mengirim data *duty cycle* ke mikrokontroler dengan terus-menerus sehingga semua sensor ultrasonik akan aktif dalam pengiriman data. Untuk itu proses *scanning* sensor ultrasonik terhadap objek dilakukan secara bergantian agar data yang didapat lebih mudah, dan valid.

2.1.2 Sensor Penyandi (*Rotary Encoder*)

Vexta Motor DC dilengkapi dengan sensor penyandi (*rotary encoder*). Sensor ini digunakan untuk mengetahui kecepatan motor yang dipresentasikan ke dalam putaran per detik yang akan menjadi pembanding penentuan *error* kecepatan motor. Sensor penyandi di sini digunakan untuk mendeteksi perpindahan / pergerakan putaran roda robot yang akan menghasilkan pulsa-pulsa yang membentuk frekuensi gelombang persegi. Setiap pulsa yang dihasilkan oleh sensor penyandi dimasukkan ke pin mikrokontroler yang mana mikrokontroler akan menghitung waktu yang dibutuhkan untuk menghasilkan 1 pulsa yang akan dikonversikan dalam bentuk putaran per detik (rps), yang selanjutnya data tersebut dapat diolah oleh mikrokontroler dalam proses kontrol robot.

Sensor penyandi (*rotary encoder*) akan menghasilkan 300 pulsa setiap satu putaran motor pada kecepatan 250 rpm. Frekuensi yang dihasilkan sensor penyandi (*rotary encoder*) di dapatkan dengan persamaan berikut;

250 rpm (rotation per minute) = 4,17 rps (rotation per second)

2.1.3 Perancangan Kontroller ATMega 16

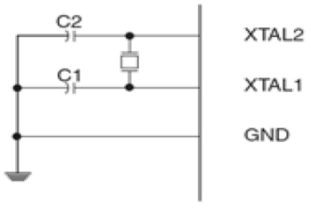
Dalam membuat rangkaian mikrokontroler memerlukan pemahaman mengenai sistem minimum dari mikrokontroler yang akan dirancang itu sendiri. Sistem rangkaian yang dirancang diusahakan menggunakan rangkaian yang seringkas mungkin dan dengan pengkabelan yang baik, karena biasanya rangkaian tersebut bekerja pada frekwensi yang relatif tinggi, sehingga peka terhadap noise dari luar.

AVR ATMega 16 mempunyai rangkaian eksternal yang relatif sedikit dibanding dengan mikrokontoler yang lain. Rangkaian eksternal yang dibutuhkan hanya berupa rangkaian :

- *clock generator CPU*
- Perancangan *Interfacing I/O*

a. Rangkaian *Clock Generator*

Mikrokontroller ATMega 16 memiliki osilator internal (on chip oscillator) yang dapat digunakan sebagai sumber clock bagi CPU. Untuk mengunakan osilator internal diperlukan sebuah kristal antara pin XTAL1 danXTAL2 dan kapasitor ke ground seperti Gambar 14. Untuk kristalnya dapat diginkan frekuensi dari 6 sampai 12 MHz. Sedangkan untuk kapasitor dapat bernilai 27 pF sampai 33 pF. Rangkaian Clock ditunjukkan pada gambar 15.



Gambar 15. Rangkaian oscilator

Mikrokontroler mempunyai 12 periode osilator dalam melakukan satu siklus mesin. Sedangkan dalam suatu program terdapat lebih dari satu siklus mesin. Sehingga dalam mengeksekusi suatu program memakan banyak waktu. Dengan mempercepat siklus mesin, dapat menjadi solusi akan lamanya waktu ekekusi program. Untuk mempercepat kita dapat mengatur waktu mein siklus sebagai berikut:

$$Tcycle = \frac{C.12}{fosc}$$

$$1\mu s = \frac{1.12}{fosc}$$

$$fosc = 12\,MHz$$

b. Perancangan Interfacing I/O

Perancangan Interfacing I/O Rangkaian I/O dari mikrokontroller mempunyai kontrol direksi yang tiap bitnya dapat dikonfigurasikan secara individual, maka dalam pengkonfigurasian I/O yang digunakan ada yang berupa operasi port ada pula yang dikonfigurasi tiap bit I/O.

Berikut ini akan diberikan konfigurasi dari I/O mikrokontroller tiap bit yang ada pada masing-masing port yang terdapat pada mikrokontroler :

A. AVR ATMEGA16 (MASTER)

1. Port A
   Port B digunakan sebagai komunikasi pararel dengan mikrokontroler slave A.
2. Port B
   Port B digunakan sebagai komunikasi pararel dengan mikrokontroler slave B.
3. Port C
   Digunakan untuk mentriger dan membaca data sensor ultrasonik dengan konfigurasi sebagai berikut:
   - Port D.0 sebagai keluaran/masukan ultrasonik depan
   - Port D.1 sebagai keluaran/masukan ultrasonik kanan serong
   - Port D.2 sebagai keluaran/masukan ultrasonik kiri serong
   - Port D.3 sebagai keluaran/masukan ultrasonik kanan
   - Port D.4 sebagai keluaran/masukan ultrasonik kiri
4. Port D
   Port D digunakan sebagai keluaran data dari mikrokontroler ke driver motor, dengan konfigurasi sebagai berikut:
   - Port D.0 sebagai keluaran arah putar motor kanan (CW/CCW)
   - Port D.1 sebagai keluaran *run/brake* motor kanan
   - Port D.2 sebagai keluaran start/stop motor kanan
   - Port D.3 sebagai keluaran putar motor kiri (CW/CCW)
   - Port D.4 sebagai keluaran *run/brake* motor kiri
   - Port D.5 sebagai keluaran start/stop motor kiri
   - Port D.6 sebagai keluaran start/stop komunikasi dengan slave A
   - Port D.7 sebagai keluaran start/stop komunikasi dengan slave B

B. AVR ATMEGA16 (SLAVE)

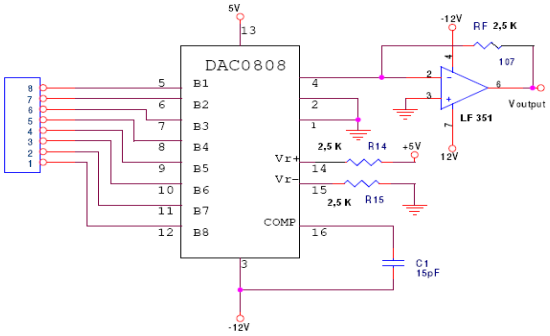
1. Port A
   Semua bit yang ada pada port A dikonfigurasikan komunikasi 8 bit dengan DAC 0808.
2. Port B
   Tidak digunakan.
3. Port C
   Semua Port C digunakan komunikasi pararel dengan mikrokontroler master:
4. Port D
   Port D digunakan sebagai masukan data sensor penyandi (rotary encoder) dan sebagai penanda adanya data yang dikirim mikrokontroler master :

- Port D.0 sebagai masukan sensor penyandi (rotary encoder)
- Port D.1 sebagai keluaran start/stop komunikasi dengan mikrokontroler master

2.1.4 Konverter Digital ke Analog (DAC)

Modul VEXTA motor DC adalah motor DC yang kecepatannya diatur oleh tegangan *analog* 0-5V. Untuk mengubah setiap konfigurasi logika masukan digital kedalam tegangan analog, maka dibutuhkan Konverter *digital* ke *analog*. Jenis IC DAC yang digunakan adalah jenis 0808. IC DAC 0808 memiliki 8 bit input yang *compatible* dengan *gate* TTL sehingga parameter keluaran dapat dihubungkan secara langsung. Jangkah tegangan output dari DAC dapat diatur sesuai dengan catu daya tegangan referensi yang diabaikan.

Proses konversi yang dilakukan DAC pada prinsipnya menggunakan metode resistor pembagi tegangan atau bisa disebut R-2R *ladder*, yang tegangan keluarannya akan memiliki nilai yang presisi. Konverter *digital* ke *analog* terdiri dari DAC 0808 yang berfungsi mengubah data 8 bit digital (0-255) menjadi besaran arus dan rangkaian Operasional Amplifier LF 351 yang akan mengubah arus menjadi tegangan. Rangkaian converter *digital* ke *analog* sepert pada Gambar 16.



Gambar 16. Rangkaian DAC

$$\frac{V_{ref}}{R_{14}} = I_{14} : I_{14} = 2\,mA$$

$$V_{ref} = 5V$$

$$R_{14} = \frac{5V}{2\,mA} = 2{,}5\,K$$

$$R_{14} = \quad R_{15} = 2{,}5\,K$$

$$I_{out(\max)} = K\left(\frac{255}{256}\right)$$

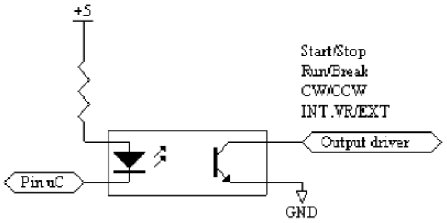
$$K = \frac{V_{ref}}{R_{14}} = 2\,mA$$

$$I_{out(\max)} = 2{,}01\,mA$$

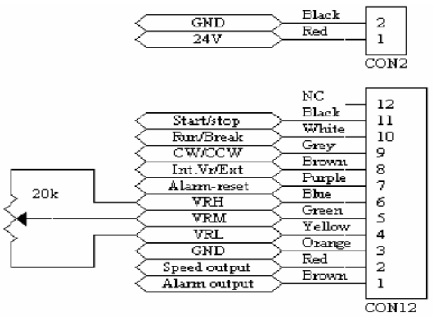
$$R_f = \frac{V_{out(\max)}}{2{,}01\,mA} = 2{,}5\,K$$

2.1.5 Driver Motor DC Vexta Brushless

Driver motor DC Brushless merupakan modul jadi sehingga tinggal mengaktifkan saja. Menghubungkan rangkaian driver motor dengan rangkaian mikrokontroler mengunakan *optocoupler* dimaksudkan agar ground dari rangkaian driver motor dan mikrokontroler terpisah hal ini untuk mencegah adanya arus balik dari motor yang dapat mengakibatkan mikrokontroller dapat ter-reset dan menghilangkan program yang tersimpan dalam mikrokontroler. Pada gambar 17 dan gambar 18 terlihat hubungan antara pin mikrokontroler dengan input ke driver motor.



Gambar 17. Koneksi input driver ke mikrokontroler



Gambar 18. Pin-pin diver motor

2.2 Perancangan Perangkat Lunak

Perancangan sistem perangkat lunak pada *mobile* robot dengan kendali PID

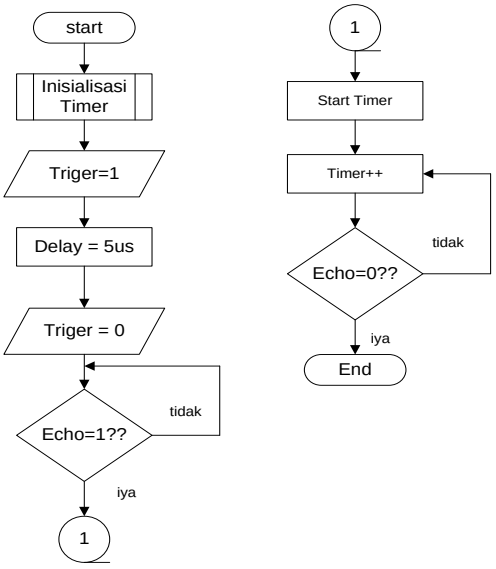
menggunakan bahasa pemprogram C berbasis mikrokontroler AVR ATMEGA 16. Pemprogram yang dilakukan meliputi pemprogramanan sensor ultrasonik sebagai indikator navigasi, pembacaan waktu 1 pulsa rotary encoder sebagai acuan kecepatan actual motor, dan pemprogramanan kecepatan motor dengan kendali PID.

2.2.1 Pemprogramanan Sensor Ultrasonik

Mikrokontroller memberikan sinyal pulsa high pada pin *triger pulse input* dari sensor untuk mengaktifkan sensor ultrasonik. Kemudian mikrokontroler akan menggunakan timer untuk menghitung waktu yang dibutuhkan antara sinyal pengiriman dengan sinyal penerimaan yang jaraknya didapat dari persamaan:

$$m = \frac{t \times v}{2}$$

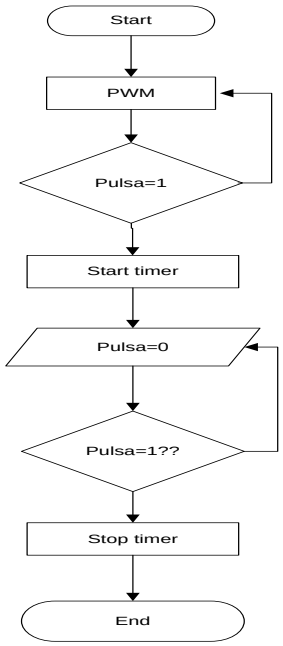
Keterangan:
m = jarak
t = waktu kirim dan terima
v = Cepat Rambat Bunyi (340m/s)
Diagram alir pemprogramanan sensor ultrasonik ditunkukkan pada gambar 19.



Gambar 19. Diagram alir sensor ultrasonik

2.2.2 Pemprogramanan Sensor Penyandi

Sensor Penyandi akan menghasilkan 300 pulsa setiap satu putaran. Mikrokontroler akan menghitung waktu yang dibutuhkan untuk menghasilkan 1 pulsa, yaitu besarnya waktu *off* dan *on*. Diagaram alir program ditunjukkan pada gambar 20.



Gambar 20. Counter waktu 1 pulsa sensor penyandi

Untuk mendapatkan kecepatan motor *rotation per second* (rps) digunakan rumus:

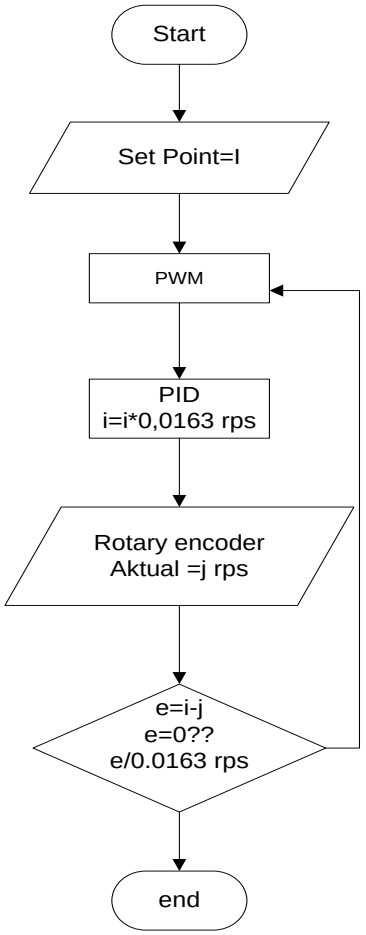
$$v(rps) = \frac{1}{300 \times t},$$

t = waktu yangdibutuhkan untuk 1 pulsa (on-off)'

2.2.3 Pemprogramanan DAC

DAC 0808 akan menghasilkan tegangan 0-5 V dengan bilangan digital 0 – 255 yang akan menghasilkan kecepatan 0 – 250 Rpm. Keluaran yang DAC akan menjadi *setting point* yang dipresentasikan dalam bentuk kecepatan (Rps) yaitu;

250 Rpm (*Rotation Per Minute*) = 4,17 Rps (*Rotation Per Second*)=256 bit
1 bit = 0,163 Rps



Gambar 21. Pemprogramman *Digital to Analog* (DAC)

2.2.4 Pemprogramman Kecepatan Motor dengan Kendali PID

Untuk mengimplementasikan kontrol PID pada mikrokontroler, PID harus dirubah ke dalam persamaan dikrit:

$$V_0 = K_P e + K_i \int e\,dt + K_d \frac{de}{dt} \quad ...(1)$$

$$\frac{dV_0}{dt} = K_p \frac{de}{dt} + K_i \frac{d}{dt}\left(\int e\,dt\right) + K_d \frac{d^e e}{dt^2} \quad ...(2)$$

$$\frac{dV_0}{dt} = K_p \frac{de}{dt} + K_i e + K_d \frac{d}{dt}\left(\frac{de}{dt}\right) \quad ...(3)$$

dikali dengan Ts, sehingga

$$\frac{\Delta V_0}{Ts} = K_p \frac{\Delta e}{Ts} + K_i e + K_d \frac{d}{dt}\left(\frac{de}{dt}\right) \quad ...(4)$$

$$\Delta V_0 = K_p \Delta e + K_i eTs + K_d \Delta\left(\frac{\Delta e}{Ts}\right) \quad ...(5)$$

Dengan;

$\Delta V_o = V_{on} - V_{on-1}$ ...(6)

$\Delta e = e_n - e_{n-1}$ ...(7)

Sehingga persamaanya menjadi:

$$V_o - V_{on-1} = K_p(e_n - e_{n-1}) + Kie_nTs + \frac{K_d}{Ts}(\Delta e_n - \Delta e_{n-1}) \quad ...(8)$$

Pada kondisi akhir, perubahan Δ pada error sebelumnya dapat didistribusikan menjadi:

$\Delta e_n = e_n - e_{n-1}$ ...(9)

$\Delta e_{n-1} = e_{n-1} - e_{n-2}$ ..(10)

Kemudian disubstitusikan ke dalam persamaan, menjadi:

$$V_0 - V_{0n-1} = K_p(e_n - e_{n-1}) + K_i e_n Ts + \frac{K_d}{Ts}\left[(e_n - e_{n-1}) + (e_{n-1} - e_{n-2})\right] \quad (11)$$

$$V_o = V_{on-1} + K_p(e_n - e_{n-1}) + K_i e_n Ts + \frac{K_d}{Ts}(e_n - 2e_{n-1} + e_{n-2}) \quad (12)$$

dengan:

V_0 = keluaran

V_{0n-1} = keluaran sebelumnya

K_p = Konstanta Proporsional

K_i = Konstanta Integral

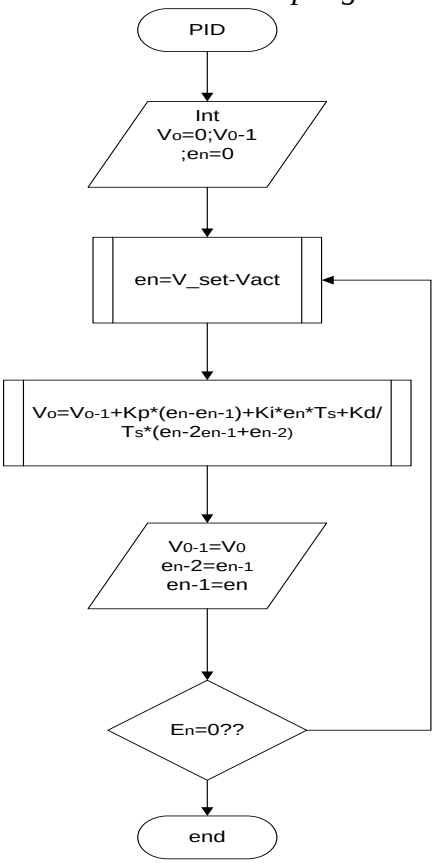
K_d = Konstanta *derivative*

e_n = *Error* sekarang

e_{n-1} = *Error* 1 kali sebelumnya

e_{n-2} = *Error* 2 kali sebelumnya

Ts = *Time Sampling*

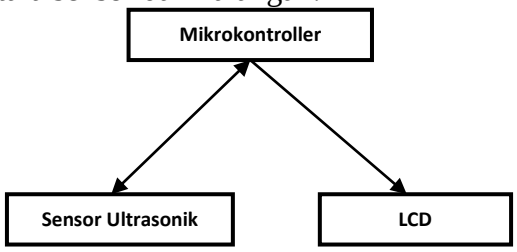


Gambar 22. Pemprogramman kontrol kecepatan motor dengan PID

2.3 Pengujian Sistem

2.3.1 Analisis dan Pengujian Sensor Ultrasonik

Pengujian ini bertujuan untuk mengetahui berapa besarnya waktu yang dihasilkan timer mikrokontroler dan jarak antara sensor dan halangan.



Gambar 23. Rangkaian Pengujian Sensor ultrasonik

Pengukuran sensor dilakukan dengan memberikan halangan yang digeser semakin menjauhi sensor ultrasonik dan mengukur jarak sensor dengan halangan dengan meteran.

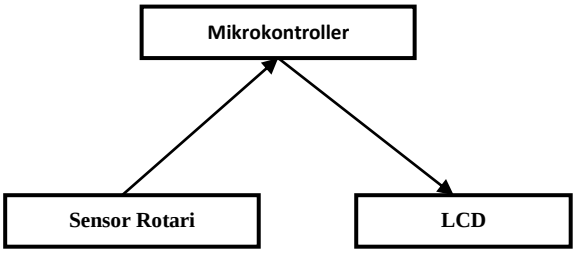
Tabel 1. Data Hasil Pengukuran Sensor ultrasonik

Pengujian	Jarak menggunakan meteran (cm)	½ Waktu Tampilan LCD (us)	jarak (cm)
1	1	92,59	3,1
2	3	92,59	3,1
3	10	300,9	10,35
4	20	601,8	20,70
5	30	879,6	30,25
6	40	1168	40,21
7	50	1458	50,16
8	60	1747	60,11
9	70	2037	70,07
10	80	2326	80,02
11	90	2627	90,37
12	100	2908	100,7
13	Tak terhingga	9016	310,1

Dari Tabel 1 hasil pengukuran di atas dapat diketahui bahwa jarak terkecil yang diperoleh adalah 3,1cm, Sedangkan jarak maksimal yang dapat diukur adalah 310 cm. Perbedaan jarak antara pengukuran dengan meteran dan jarak tampilan LCD dapat disebabkan karena peletakan posisi halangan yang kurang tepat dan bidang halangan yang tidak rata.

2.3.2 Analisis dan Pengujian Sensor Penyandi (*Rotary Encoder* **)**

Pengujian ini bertujuan untuk mengetahui berapa besarnya waktu yang dihasilkan timer mikrokontroler untuk mengcounter 1 pulsa dari sensor rotary encoder.



Gambar 24. Rangkaian Penguji Sensor Penyandi

Pengukuran dilakukan dengan mengukur waktu *duty cycle* sensor penyandi yaitu waktu untuk menghasilkan 1 pulsa. Untuk mengetahui waktu yang digunakan untuk satu putaran adalah:

1 putaran = 300 pulsa

waktu 1 putaran = 300 pulsa * waktu satu pulsa

$$Putaran/s = \frac{1\,Putaran \times 1\,Detik}{Waktu\,1\,putaran}$$

Tabel 2. Pengujian Sensor Penyandi

Pengujian	PWM	waktu (s)	kecepatan (r/s)
1	255	0,215	4.6
2		0,208	4.8
3		0,201	4.9
4		0,194	5.1
5		0,187	5.3
6			
7	128	0,396	2.5
8		0,368	2.7
9		0,347	2.8
10		0,326	3
11		0,312	3.2
12			
13	64	0,715	1.3
14		0,687	1.4
15		0,653	1.5
16		0,569	1.7
17		0,535	1.8

18			
19	32	1,125	0.8
20		1,035	0.9
21		0,986	1
22			
23	16	3,528	0.2
24		2,861	0.3
25		2,413	0.4

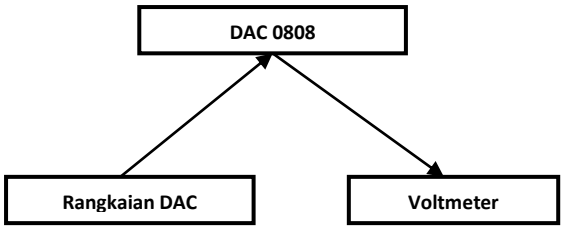
Tabel 2 di atas menunjukkan bahwa semakin besar nilai PWM, maka waktu yang digunakan untuk satu putaran juga semakin kecil dan jumlah putaran yang dicapai dalam satu detik juga semakin besar. Namun data hasil pengujian tidak sesuai, yaitu pada kecepatan maksimal (255) 4,7 rps sedangkan kecepatan maksimal hasil perhitungan 4,17 rps.
maka toleransi hasil kesalahan rata-rata dapat diperoleh;

$$Kesalahan = \left| \frac{Hasil\ Perhitungan - Hasil\ Percobaan}{Hasil\ Perhitungan} \right| \times 100\%$$

$$Kesalahan = \left| \frac{4{,}17 - 4{,}6}{4{,}17} \right| \times 100\%$$

Kesalahan = 10,31 %

2.3.3 Analisis dan Pengujian Rangkaian *Digital to analog* (DAC)



Gambar 25. Rangkaian Penguji DAC

Pengukuran dilakukan dengan memberikan data 8 bit dengan 2^n pada port masukan DAC yang kemudian di ukur besar tegangan keluarannya menggunakan voltmeter.

Tabel 3. Pengujian Rangkaian DAC 0808

Pengujian	PWM	Tegangan (V)
1	2	37,8 mV
2	4	77,8 mV
3	8	156,1 mV
4	16	312,6 mV
5	32	0,625 V
6	64	1,258 V
7	128	2,513 V
8	255	5,01 V

Tabel 3 menunjukkan bahwa semakin besar nilai PWM maka tegangan yang dihasilkan akan semakin besar.

$$V_{outout} = V_{cc} \frac{2n\ input}{256}$$

$$= 5V \frac{255}{256}$$

$$= 4{,}98V$$

maka toleransi hasil kesalahan rata-rata dapat diperoleh::

$$Kesalahan = \left| \frac{Hasil\ Perhitungan - Hasil\ Percobaan}{Hasil\ Perhitungan} \right| \times 100\%$$

$$Kesalahan = \left| \frac{4{,}98 - 5{,}01}{4{,}98} \right| \times 100\%$$

Kesalahan = 0,6 %

Pergeseran pada pengujian ini disebabkan karena penggunaan komponen yang masing–masing memiliki tingkat toleransi yang berbeda sehingga akan mempengaruhi besar tegangan hasil yang diperoleh.

2.3.4 Analisis dan Pengujian Rangkaian Mikrokontroler dan Komunikasi Pararel antar mikrokontroler.

Hasil Pengujian Komunikasi Pararel antar Mikrokontroler.

a. Langkah – langkah pengujian
   1. Membuat program flip-flop untuk menguji rangkaian mikrokontroler dengan kode program:

(***PROGRAM MASTER***)

```
#include <atmega16.h>
#include <delay.h>
while(1)
{
PORTD.6=1;PORTA=255;delay
_ms(1000);
PORTA=0;delay_ms(1000);POR
TD.6=0;
PORTD.6=0;PORTA=255;delay
_ms(1000);
```

```
PORTA=0;delay_ms(1000);
PORTD.6=0;}
```

(***PROGRAM SLAVE***)

```
#include <atmega16.h>
#include <delay.h>
while(1)
{while (PIND.1==1)
{PORTA=PINC;}
;}
```

2. Mengamati kombinasi biner pada rangkaian LED port A ATMega slave

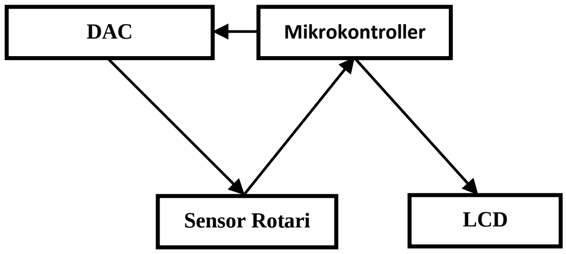
Tabel 4. Hasil Pengujian Komunikasi Paralel Multi-Mikrokontroler

PORTA (MASTER)	PORTA (SLAVE)	Hasil
PORTA.0 = 1	PORTA.0 = 1	Led mati
PORTA.1 = 1	PORTA.1 = 1	Led mati
PORTA.2 = 1	PORTA.2 = 1	Led mati
PORTA.3 = 1	PORTA.3 = 1	Led mati
PORTA.4 = 1	PORTA.4 = 1	Led mati
PORTA.5 = 1	PORTA.5 = 1	Led mati
PORTA.6 = 1	PORTA.6 = 1	Led mati
PORTA.7 = 1	PORTA.7 = 1	Led mati
PORTA.0 = 0	PORTA.0 = 0	Led hidup
PORTA.1 = 0	PORTA.1 = 0	Led hidup
PORTA.2 = 0	PORTA.2 = 0	Led hidup
PORTA.3 = 0	PORTA.3 = 0	Led hidup
PORTA.4 = 0	PORTA.4 = 0	Led hidup
PORTA.5 = 0	PORTA.5 = 0	Led hidup
PORTA.6 = 0	PORTA.6 = 0	Led hidup
PORTA.7 = 0	PORTA.7 = 0	Led hidup
PORTA.0 = 0	PORTA.0 = 0	Led hidup
PORTA.1 = 0	PORTA.1 = 0	Led hidup
PORTA.2 = 0	PORTA.2 = 0	Led hidup
PORTA.3 = 0	PORTA.3 = 0	Led hidup
PORTA.4 = 0	PORTA.4 = 0	Led hidup
PORTA.5 = 0	PORTA.5 = 0	Led hidup
PORTA.6 = 0	PORTA.6 = 0	Led hidup
PORTA.7 = 0	PORTA.7 = 0	Led hidup
PORTA.0 = 0	PORTA.0 = 0	Led hidup
PORTA.1 = 0	PORTA.1 = 0	Led hidup
PORTA.2 = 0	PORTA.2 = 0	Led hidup
PORTA.3 = 0	PORTA.3 = 0	Led hidup
PORTA.4 = 0	PORTA.4 = 0	Led hidup
PORTA.5 = 0	PORTA.5 = 0	Led hidup
PORTA.6 = 0	PORTA.6 = 0	Led hidup
PORTA.7 = 0	PORTA.7 = 0	Led hidup

Dari Table 2.4 dapat disimpulkan bahwa mikrokontroler master sudah dapat berkomunikasi dengan mikrokontroler slave, dapat dilihat ketika data PORTA mikrokontroler master akan sama dengan PORTA mikrokontroler slave.

2.3.4 Analisis dan Pengujian Implementasi PID

Pengujian ini bertujuan untuk mengetahui berapa besarnya waktu yang dihasilkan timer mikrokontroler untuk mengcounter 1 pulsa dari sensor *rotary encoder*.



Gambar 26. Rangkaian Penguji Implementasi PID

Pengukuran dilakukan dengan menampilkan data pengujian *setting point* (SP) PWM analog yang dirubah dalam bentuk kecepata (rps) dengan perbandingan telah ditentukan, kemudian menampilkan kecepatan putar motor (rps) sebagai present value (SP) dengan *sample time* 1 detik dan menampilkan PWM analog hasil dari proses PID digital.

8 bit = 255 = 250 rpm
= 4,17 rps
1 = 4,17/255 rps
= 0,0164 rps

Tabel 5. Pengujian Implementasi PID

Setting point (SP) rps	Present value (PV) rps	PWM
4.17	4.8	221
2,1	2,7	111

Tabel 5 menunjukkan perbedaan *setting point* (rps) dan present value (rps) (SP < PV) menyebabkan PWM analog yang dikeluarkan kontroler semakin kecil, perbedaan hasil pengujian pembacaan besar nilai kecepatan sensor penyandi (*rotary encoder*) dengan besar nilai hasil perhitungan menyebabkan kecepatan maksimal motor tidak dapat dihasilkan.

3. Kesimpulan

Penerapan multi mikrokontroller (master-slave) pada mobil robot dapat bekerja dengan baik. Pembacaan kecepatan putar motor dengan mengukur *duty cycle* pulsa motor kurang efektif, hal tersebut dapat dilihat dari besar nilai kecepatan yang berubah pada setiap

percobaan *setting point* PWM analog. Implementasi PID sebagai pengatur kecepatan motor sudah dapat bekerja, hal ini dapat dilihat pada *setpoint* (SP) PWM analog 5 V (255 desimal/4,17rps) dan pada pembacaan sensor penyandi kecepatan yang terbaca 4,8 rps dan kontroler menghasilkan PWM analog 4,3 V (221 desimal) untuk mencapai *error* sama dengan 0. Implementasi PID pada pengatur kecepatan motor DC pada navigasi *mobile robot* sangat berguna untuk menjaga kecepatan motor kiri dan kanan sesuai dengan *settng point* yang ditentukan.

Masih banyak perhitungan konversi data *setting point* (SP) dan *present value* (PV) dengan mengunakan perbandingan sehingga diharapkan dapat dirancang sistem dengan *setting point* (SP) dan *present value* (PV) berupa tegangan.

Konversi kecepatan motor dapat menggunakan metode lain yaitu menggunakan konversi frekuensi ke tegangan dan menggunakan pembacaan *present value* (PV) dengan menggunakan konverter tegangan analog ke digital (ADC) sehingga data 8 bit konverter digital ke analog (DAC) PWM analog dapat langsung dibandingkan dengan data 8 bit konverter anlog ke digital (ADC).

Daftar Pustaka

[1] Terbuc, M., Hace,A., Jazernik, K., " Open Structure Multipocessor Robot Controller". In , vol 2, pages 998-902.

[2] Widodo N.S., "Penerapan Multi Mikrokontroller pada Model Robot Mobil Berbasis Logika Fuzi". Yogyakarta: Jurnal Teknik Elektro

[3] Wicaksono, H., "Pemprosesan SRF05, CMPS03, TPA81, Sistem Motor Secara MultiProsesor pada Robot KRPAI". In Prosiding Conference on Smart_grenn Technology in Electrical and Infromation System, Bali, 2013. Universitas Ahmad Dahlan.

[4] Gunterus, Frans: Falsafah Dasar: "*Sistem Pengendalian Proses*", jakarta: PT. Elex Media Komputindo, Jakarta, 1994.

[5] Ogata, Katsuhiko: *Teknik Kontrol Automatik* – terjemahan: Ir. Edi Laksono, Erlangga, Jakarta, 1991.