TUGAS AKHIR – TE 141599

PERANCANGAN PENGGERAK ELEKTRIK DAN KONTROLER *FUZZY*-PI UNTUK PENGATURAN KECEPATAN MOTOR DC *BRUSHLESS*

Marika Ayu Putri Ramadhani
NRP 2213105055

Dosen Pembimbing
Ir. Rusdhianto Effendie A K, M.T.

JURUSAN TEKNIK ELEKTRO
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember
Surabaya 2015

FINAL PROJECT – **TE 141599**

DESIGN ELECTRONIC DRIVE AND FUZZY-PI CONTROLLER FOR SPEED OF MOTOR BRUSHLESS DC

Marika Ayu Putri Ramadhani
NRP 2213105055

Advisor
Ir. Rusdhianto Effendi A K, M.T.

DEPARTMENT OF ELECTRICAL ENGINEERING
Faculty of Industrial Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2015

PERANCANGAN PENGGERAK ELEKTRIK DAN KONTROLER *FUZZY*-PI UNTUK PENGATURAN MOTOR DC *BRUSHLESS*

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Sarjana Teknik
Pada
Bidang Studi Teknik Sistem Pengaturan
Jurusan Teknik Elektro
Institut Teknologi Sepuluh Nopember

Menyetujui :

Dosen Pembimbing

Ir. Rusdhianto Effendie A K, M.T.
NIP. 195704241985021001

SURABAYA
JULI, 2015

PERANCANGAN PENGGERAK ELEKTRIK DAN KONTROLER *FUZZY*-PI UNTUK PENGATURAN MOTOR DC *BRUSHLESS*

Nama : **Marika Ayu Putri Ramadhani**
Dosen Pembimbing 1 : **Ir. Rusdhianto Effendi A.K, M.T.**

ABSTRAK

Motor *Brushless Direct Current* (BLDC) membutuhkan suatu alat untuk menggerakan dan mengendalikan putaran motor BLDC yang biasa disebut dengan penggerak daya atau *driver* motor BLDC agar motor BLDC dapat dikendalikan secara akurat.

Sistem kontroler *fuzzy*-PI digunakan untuk mengatur kecepatan motor BLDC dan menjaga kestabilan putarannya. Kontroler ini memiliki parameter-parameter pengontrol yaitu *gain integral error*, *gain error*, *control offset*, dan *gain control*. Pada metode ini nilai parameter tersebut ditentukan berdasarkan *tunning* sesuai dengan hasil yang diperlukan.

Hasil implementasi pada motor BLDC menunjukkan bahwa kontroler *fuzzy*-PI mampu mendekati nilai kecepatan referensi. Pada saat kondisi beban minimal memiliki spesifikasi respon dengan nilai *time constant* = 2.248 detik, *settling time* $(\pm 5\%) = 6.744\, detik$, *rise time* $(5\% - 95\%) = 6.619\, detik$, *delay time* $= 1.558\, detik$ dan *error steady state* = 0.001%.

Kata kunci : *Driver*, Kontroler *Fuzzy*-PI, Motor BLDC.

PERANCANGAN PENGGERAK ELEKTRIK DAN KONTROLER *FUZZY*-PI UNTUK PENGATURAN MOTOR DC *BRUSHLESS*

Nama : **Marika Ayu Putri Ramadhani**
Dosen Pembimbing 1 : **Ir. Rusdhianto Effendi A.K, M.T.**

ABSTRAK

Motor *Brushless Direct Current* (BLDC) membutuhkan suatu alat untuk menggerakan dan mengendalikan putaran motor BLDC yang biasa disebut dengan penggerak daya atau *driver* motor BLDC agar motor BLDC dapat dikendalikan secara akurat.

Sistem kontroler *fuzzy*-PI digunakan untuk mengatur kecepatan motor BLDC dan menjaga kestabilan putarannya. Kontroler ini memiliki parameter-parameter pengontrol yaitu *gain integral error*, *gain error*, *control offset*, dan *gain control*. Pada metode ini nilai parameter tersebut ditentukan berdasarkan *tunning* sesuai dengan hasil yang diperlukan.

Hasil implementasi pada motor BLDC menunjukkan bahwa kontroler *fuzzy*-PI mampu mendekati nilai kecepatan referensi. Pada saat kondisi beban minimal memiliki spesifikasi respon dengan nilai *time constant* = 2.248 detik, *settling time* $(\pm 5\%) = 6.744\, detik$, *rise time* $(5\% - 95\%) = 6.619\, detik$, *delay time* $= 1.558\, detik$ dan *error steady state* = 0.001%.

Kata kunci : *Driver*, Kontroler *Fuzzy*-PI, Motor BLDC.

Halaman ini sengaja dikosongkan

DESIGN ELECTRONIC DRIVE AND FUZZY-PI CONTROLLER FOR SPEED OF MOTOR BRUSHLESS DC

Name **: Marika Ayu Putri Ramadhani**
***Advisor* 1** **: Ir. Rusdhianto Effendi A.K, M.T.**

ABSTRACT

Motor Brushless Direct Current (BLDC) need a tool to move and to control the BLDC motor rotation which is usually called the driving power or BLDC motor driver that BLDC motors can be controlled accurately.

System of fuzzy-PI controller is used to control the speed of a BLDC motors and maintain stable rotation. This controller has a controller parameters that integral gain error, gain error, control offset and gain control. In this method the value of the parameter is determined by tunning in accordance with the required result.

The results of the implementation of the BLDC motor show that the fuzzy-PI controller is able to approach the reference speed value. At the time of minimal load conditions have response specification with value of time constant = 2248 seconds, settling time (± 5%) = 6,744 seconds, rise time (5% -95%) = 6619 seconds, delay time = 1,558 seconds and error steady state = 0.001%.

Keywords : *Driver, Fuzzy-PI Controller, BLDC Motors.*

Halaman ini sengaja dikosongkan

KATA PENGANTAR

Dengan mengucapkan puji syukur kehadirat Allah SWT yang telah melimpahkan rahmat, hidayah serta karunia-Nya sehingga penulis dapat menyelesaikan tugas akhir dengan judul :

"PERANCANGAN PENGGERAK ELEKTRIK DAN KONTROLER *FUZZY*-PI UNTUK PENGATURAN KECEPATAN MOTOR DC *BRUSHLESS*"

Tugas akhir ini merupakan sebagian syarat untuk menyelesaikan mata kuliah dan memperoleh nilai pada tugas akhir.

Dengan selesainya tugas akhir ini penulis menyampaikan terima kasih sebesar-besarnya kepada:

1. Ibu Tri Setyawati selaku orang tua dan kakak kandung penulis, Agung Yuana Putra sekeluarga atas limpahan doa, kasih sayang, dukungan dan dorongan baik berupa moril atau materil bagi penulis.
2. Bapak Ir. Rusdhianto Effendie A K, M.T. selaku dosen pembimbing.
3. Rekan kerja untuk Tugas Akhir, Suwondo Saputra dan Kurniawan Khoiruddin yang telah banyak membantu penulis dalam penyelesaian Tugas Akhir ini.
4. Semua pihak yang telah banyak membantu untuk menyelesaikan tugas akhir ini yang tidak dapat penulis sebutkan satu persatu.

Penulis menyadari masih banyak kekurangan dalam tugas akhir ini. Kritik dan saran untuk perbaikan tugas ini sangat diperlukan. Akhir kata semoga tugas ini dapat bermanfaat bagi kita semua.

Surabaya, Juli 2015

Penulis

Halaman ini sengaja dikosongkan

DAFTAR ISI

Halaman ini sengaja dikosongkan

DAFTAR TABEL

Halaman ini sengaja dikosongkan

DAFTAR GAMBAR

DAFTAR RIWAYAT HIDUP

Marika Ayu Putri Ramadhani , lahir di Surabaya, Jawa Timur pada tanggal 1 Maret 1993. Setelah lulus dari SMA Trimurti Surabaya tahun 2010, penulis melanjutkan studi di Diploma 3 Bidang Studi *Computer Control* jurusan Teknik Elektro Institut Teknologi Sepuluh Nopember (ITS) Surabaya dan lulus tahun 2013. Kemudian melanjutkan kuliah dengan mengambil program Sarjana Lintas Jalur di Institut Teknologi Sepuluh Nopember (ITS) Surabaya dengan mengambil Bidang Studi Teknik Sistem Pengaturan, Jurusan Teknik Elektro. Pada bulan Juni 2015, penulis mengikuti seminar dan ujian tugas akhir di Bidang Studi Teknik Sistem Pengaturan, Jurusan Teknik Elektro, ITS Surabaya sebagai salah satu persyaratan untuk memperoleh gelar Sarjana Teknik Elektro.

Halaman ini sengaja dikosongkan

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Motor DC adalah sebuah motor yang membutuhkan tegangan dc untuk menjalankannya. Secara umum motor DC dibagi atas 2 macam, yaitu : motor DC dengan sikat (*Brushed* DC *Motor*) dan motor DC tanpa sikat (*Brushless* DC *Motor*). Jenis *Brushed* DC motor memerlukan perawatan pada sikatnya serta banyak terjadi rugi tegangan pada sikat. Sehingga pada era sekarang ini motor DC dikembangkan tanpa menggunakan sikat yang dikenal dengan motor BLDC *(Brushless Direct Current Motor)*. Motor ini dipilih karena efisiensi yang tinggi, suaranya halus, ukuran kompak, keandalan yang tinggi dan biaya perawatan yang rendah. Motor BLDC telah mendominasi banyak aplikasi seperti peralatan rumah, peralatan teknologi informasi, industri, transportasi, *aerospace*, peralatan pertahanan, alat listrik, dan peralatan medis laboratorium berbagai bidang. Keuntungan yang motor BLDC berikan kepada setiap aplikasi yang digunakan, terutama pada industri sangat besar. Penggunaan motor ini dapat menghemat biaya dan waktu pada hampir semua industri.[1]

Perubahan motor *Brushed* DC oleh motor BLDC menjadi penyebab diperlukannya cara kontrol yang berbeda untuk komutasi fase arus dari motor BLDC. Rangkaian kontrol pengganti kumutator pada motor BLDC ini disebut dengan *power electronic* untuk mencatu daya ke kumparan stator . Rangkaian kontrol terdiri atas 6 buah MOSFET yang digunakan untuk *switching* tegangan. Pada penerapannya motor BLDC masih terdapat banyak kesalahan antara kecepatan referensi dan kecepatan *feedback*. Berdasarkan hal itu maka diperlukan suatu kontroler untuk memperbaiki kinerja dari motor BLDC.[1]

Kontroler konvensional seperti *Proporsional Integral* (PI) dan kontroler *Proporsional Integral Derivative* (PID) masih banyak digunakan karena memiliki struktur sederhana, tingkat keandalan tinggi dan mudah untuk mencapai kondisi sistem yang dibutuhkan. Namun motor BLDC memiliki multi-variabel, sistem non-linear dan dengan mudah dapat dipengaruhi oleh variasi parameter serta gangguan. Untuk mengatasi masalah ini kontroler *fuzzy*-PI digunakan untuk mengontrol kecepatan dari motor BLDC.[2]

Penerapan teknik kecerdasan buatan seperti *Fuzzy Logic Control* (FLC) telah ditemukan sebagai kontroler yang cocok untuk sebagian besar sistem nonlinear yang kompleks dimana sistem tersebut memiliki pemodelan matematika yang tidak pasti. Kelebihan FLC adalah karena kontroler untuk sistem apapun dapat dikembangkan tanpa persyaratan untuk model matematika dari suatu sistem. Dengan cara ini, efisiensi dan keandalan *drive* akan meningkat. Sehingga pada tugas akhir ini akan dibuat rangkaian penggerak elektrik dan kontroler *fuzzy*-PI untuk motor BLDC.

1.2 Permasalahan

Permasalahan yang dibahas dalam tugas akhir ini adalah perancangan penggerak elektrik dan pengaturan kecepatan motor BLDC dengan kontroler *fuzzy*-PI untuk memperbaiki respon kecepatan dari motor BLDC sehingga kecepatan motor sesuai dengan yang diperlukan serta bagaimana menentukan aturan dasar kontroler *fuzzy* untuk kontrol kecepatan motor BLDC. untuk meningkatkan performansi serta kestabilan sistem.

1.3 Tujuan

Tujuan dari tugas akhir ini adalah untuk merancang penggerak elektrik untuk motor BLDC dan merancang kontroler *fuzzy*-PI untuk mengatur kecepatan motor BLDC sehingga dapat memperbaiki respon kecepatan dari motor BLDC saat terjadi perubahan beban dan sesuai dengan respon model yang diinginkan.

1.4 Metodologi

Metodologi yang dilakukan dalam pengerjaan tugas akhir ini terdiri dari beberapa tahap, yaitu :

a. Studi Literatur

   Studi literatur dilakukan untuk memperoleh informasi mengenai motor BLDC melalui buku teks, jurnal, artikel, internet dan lain-lain.

b. Pemodelan Sistem

   Pada tahap ini dibuat pemodelan motor BLDC berdasarkan pengambilan data dari motor BLDC. Setelah mendapatkan data yang dibutuhkan, akan dilakukan identifikasi dari motor BLDC untuk mendapatkan model matematika yang akan digunakan untuk mendesain kontroler sesuai dengan yang diharapkan.

c. Desain Kontroler

Pada tahap ini dibuat struktur kontroler *fuzzy*-PI untuk pengaturan kecepatan motor BLDC.

d. Simulasi

Pemodelan motor BLDC dan hasil desain kontroler disimulasikan dengan menggunakan perangkat lunak MATLAB.

e. Penyusunan Buku Tugas Akhir

Penyusunan buku Tugas Akhir meliputi pendahuluan, teori penunjang, perancangan sistem, simulasi dan analisa sistem serta penutup.

1.5 Sistematika

Pada Tugas Akhir ini sistematika penulisan dibagi menjadi lima bab, yaitu :

BAB 1 PENDAHULUAN

Bab ini meliputi latar belakang, permasalahan, tujuan, metodologi, sistematika penulisan dan relevansi pembahasan tugas akhir ini.

BAB 2 TEORI PENUNJANG

Bab ini membahas tinjauan pustaka yang membantu penelitian, diantaranya konsep motor BLDC, teori identifikasi sistem, teori penggerak elektronik meliputi *inverter*, sensor *rotary encoder*, dan arduino serta teori kontroler *fuzzy*-PI.

BAB 3 PERANCANGAN SISTEM

Pada bab ini dijelaskan mengenai perancangan model dinamik motor BLDC serta perancangan kontroler *fuzzy*-PI.

BAB 4 PENGUJIAN DAN ANALISA SISTEM

Bab ini memuat hasil simulasi kontroler pada sistem dan analisa sistem.

BAB 5 PENUTUP

Bab ini berisi kesimpulan dan saran dari hasil pembahasan yang telah diperoleh.

1.6 Relevansi

Hasil yang diperoleh dari tugas akhir ini diharapkan dapat memberikan gambaran mengenai perancangan penggerak elektrik untuk motor BLDC serta penerapan kontroler *fuzzy*-PI pada *plant* dengan menggunakan simulasi model *simulink* MATLAB untuk pengaturan kecepatan motor BLDC.

BAB 2
TEORI PENUNJANG

Bab ini membahas mengenai tinjauan pustaka yang berisi landasan teori di dalam pembuatan Tugas Akhir. Beberapa landasan teori yang digunakan antara lain mengenai motor BLDC, teori identifikasi sistem, teori penggerak elektronik meliputi *inverter*, sensor *encoder* serta teori kontroler *Fuzzy*-PI.

2.1 Motor BLDC (*Brushless Direct Current Motor*) [3]

Motor DC banyak digunakan pada aplikasi dunia industri memiliki konstruksi yang menyebabkan timbulnya beberapa kendala pada penggunaannya. Kendala utama dari suatu motor DC adalah pengunaan kumparan rotor dan pemakaian komutator sikat atau sikat arang untuk menghubungkan kumparan rotor dengan unit *servo drive*. Kendala demikian mencakup penggantian atau penyetelan sikat arang, bunga api karena komutasi, keterbatasan arus dan tegangan, inersia rotor yang tinggi, dan satu lintasan panjang disipasi panas yang dibangkitkan rotor. Namun kelemahan pada motor DC dapat dihilangkan dengan penggantian pemakaian motor DC dengan motor BLDC.

Motor BLDC merupakan motor listrik sinkron AC tiga fasa. Motor ini dapat dikendalikan dengan metode *six-step* maupun metode PWM sinusoidal. Dibandingkan dengan motor DC, motor BLDC memiliki biaya perawatan yang lebih rendah dan kecepatan yang lebih tinggi akibat tidak digunakannya sikat. Dibandingkan dengan motor induksi, motor BLDC memiliki efisiensi yang lebih tinggi karena rotor dan torsi awal yang lebih tinggi karena rotor terbuat dari magnet permanen. Walaupun memiliki kelebihan dibandingkan dengan motor DC dan motor induksi, pengendalian motor BLDC jauh lebih rumit untuk kecepatan dan torsi yang konstan karena tidak adanya sikat yang menunjang proses komutasi dan harga motor BLDC jauh lebih mahal.

Secara umum motor BLDC terdiri dari dua bagian, yakni, *rotor*, bagian yang bergerak, yang terbuat dari permanen magnet dan *stator,* bagian yang tidak bergerak, yang terbuat dari kumparan 3 fasa. Walaupun merupakan motor listrik *synchronous* AC 3 fasa, motor ini tetap disebut dengan motor BLDC karena pada implementasinya motor BLDC menggunakan sumber DC sebagai sumber energi utama yang kemudian diubah menjadi tegangan AC dengan menggunakan inverter 3

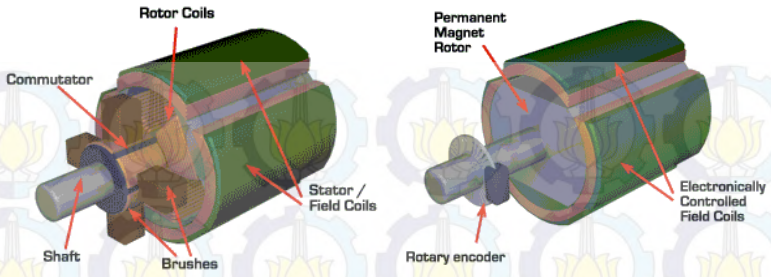
fasa. Tujuan dari pemberian tegangan AC 3 fasa pada stator BLDC adalah menciptakan medan magnet putar stator untuk menarik magnet rotor.

Pada dasarnya, motor BLDC adalah motor DC yang dihilangkan sikat dan komutatornya. Penghilangan komutator dan kumparan rotor akan mengurangi inersia motor dan mampu menaikkan kecepatan rotor. Motor BLDC memiliki rotor berupa suatu magnet permanen, kumparan kawat yang terletak pada stator, dan satu rangkaian elektronik pengganti komutator atau sikat arang. Rangkaian komutasi elektronik mengeliminasi penggantian sikat arang sehingga kumparan dapat dilewati arus dengan tegangan yang lebih tinggi.

Magnet permanen yang digunakan pada rotor memiliki 2 jenis, yaitu :

1. *Rare-earth magnet*
   Magnet ini terbuat dari *samarium cobalt* dan *neodymium iron boron* dan memiliki sifat magnet yang bagus dan banyak tersedia di pasaran, tetapi berharga mahal dan diproduksi secara terbatas.
2. *Ceramic magnets (ferrite)*
   Magnet keramik berharga murah dan dapat digunakan dengan mudah, tetapi magnet ini memiliki sifat magnet yang rendah.

Motor BLDC dengan jenis *rare earth permanen magnet* memiliki inersia rotor yang paling rendah dan ukuran terkecil pada suatu rating torsi tertentu. Stator motor BLDC memiliki kumparan tiga fasa dan menggunakan satu penguat arus PWM untuk mensuplai satu arus sinusoidal tiga fasa pada ketiga kumparan stator. Tipe motor penggerak yang lain menggunakan satu rangkaian pengatur yang lebih sederhana untuk menghasilkan satu arus gelombang persegi tiga fasa, tetapi operasi motor demikian tidak sehalus seperti yang menggunakan penggerak sinusoidal. Gambar 2.1 menunjukkan konstruksi dari motor DC dan motor BLDC.



Gambar 2.1 (a) Konstruksi Motor DC (b) Konstruksi motor BLDC

Gambar 2.1 menunjukkan perbedaan antara motor DC dengan motor BLDC. Pada Gambar 2.1 (a), motor DC menggunakan komutator untuk membalik fasa rotor, sedangkan Gambar 2.1 (b), motor BLDC memiliki magnet permanen pada rotornya dan menggunakan *rotary encoder* untuk membalik fasa rotor secara elektrik. Motor BLDC cenderung lebih efisien jika dibandingkan dengan motor DC karena tidak memiliki sikat dan komutator.

Oleh karena tidak adanya sikat pada motor BLDC, untuk menentukan *timing* komutasi yang tepat pada motor ini sehingga didapatkan torsi dan kecepatan yang konstan, diperlukan 3 buah sensor *Hall* dan atau *encoder*. Pada sensor *Hall, timing* komutasi ditentukan dengan cara mendeteksi medan magnet *rotor* dengan menggunakan 3 buah sensor *hall* untuk mendapatkan 6 kombinasi *timing* yang berbeda, sedangkan pada *encoder, timing* ditentukan dengan cara menghitung jumlah *pole* (kutub) yang ada pada *encoder*.

Pada umumnya *encoder* lebih banyak digunakan pada motor BLDC komersial karena *encoder* cenderung mampu menentukan *timing* komutasi lebih presisi dibandingkan dengan menggunakan sensor *hall*. Hal ini terjadi karena pada *encoder,* kode komutasi telah ditetapkan secara *fixed* berdasarkan banyak *pole* dari motor dan kode inilah yang digunakan untuk menentukan *timing* komutasi. Namun karena kode komutasi *encoder* ditetapkan secara *fixed* berdasarkan banyak *pole* motor, suatu *encoder* untuk suatu motor tidak dapat digunakan untuk motor dengan jumlah *pole* yang berbeda. Hal ini berbeda dengan sensor *hall*. Apabila terjadi perubahan *pole* rotor pada motor, posisi sensor *hall* dapat diubah dengan mudah. Hanya saja kelemahan dari sensor *hall* adalah posisi sensor *hall* tidak tepat akan terjadi kesalahan dalam

penentuan *timing* komutasi atau bahkan tidak didapatkan 6 kombinasi *timing* yang berbeda.

2.1.1 Cara Kerja Motor BLDC

Motor BLDC ini dapat bekerja ketika stator yang terbuat dari kumparan diberikan arus 3 fasa. Akibat arus yang melewati kumparan pada stator timbul medan magnet (B):

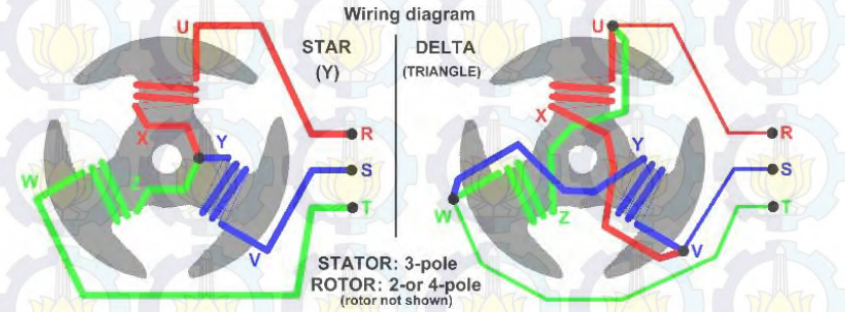
$$B = \frac{\mu N l}{2i} \tag{2.1}$$

Dimana N merupakan jumlah lilitan, i merupakan arus, l merupakan panjang lilitan, dan μ adalah permeabilitas bahan.

Karena arus yang diberikan berupa arus AC fasa, nilai medan magnet dan polarisasi setiap kumparan akan berubah – ubah setiap saat. Akibat yang ditimbulkan dari adanya perubahan polarisasi tersebut dan besar medan magnet tiap kumparan adalah terjadinya medan putar magnet dengan kecepatan N_s :

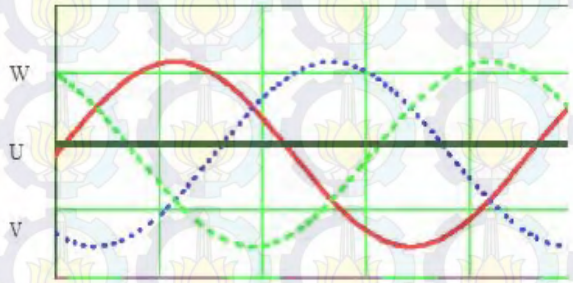
$$N_s = \frac{120f}{p} \tag{2.2}$$

Dimana f merupakan frekuensi tegangan input dinyatakan dalam Hz per satuan detik, p merupakan jumlah kutub (*pole)* pada rotor dan 120° didapat dalam 1 putaran (360°) per 3 fasa motor.



Gambar 2.2 Medan Magnet Putar Stator dan Perputaran Rotor

Berdasarkan gambar 2.2, medan putar magnet stator timbul akibat adanya perubahan polaritas pada stator U, V, dan W. Perubahan polaritas ini terjadi akibat adanya arus yang mengalir pada stator berupa arus AC yang memiliki polaritas yang berubah-ubah.



Gambar 2.3 Tegangan Stator Motor BLDC

Berdasarkan Gambar 2.3, ketika stator U diberikan tegangan negatif maka akan timbul medan magnet dengan polaritas negatif sedangkan V dan W yang diberikan tegangan positif akan memiliki polaritas positif. Akibat adanya perbedaan polaritas antara medan magnet kumparan stator dan magnet rotor, sisi positif magnet rotor akan berputar mendekati medan magnet stator U, sedangkan sisi negatifnya akan berputar mengikuti medan magnet stator V dan W. Akibat tegangan yang digunakan berupa tegangan AC sinusoidal, medan magnet stator U, V, dan W akan berubah – ubah polaritasnya dan besarnya mengikuti perubahan tegangan sinusoidal AC. Ketika U dan V memiliki medan magnet negatif akibat mendapatkan tegangan negatif dan W memiliki medan magnet positif akibat tegangan positif, magnet permanen rotor akan berputar menuju ke polaritas yang bersesuaian yakni bagian negatif akan berputar menuju medan magnet stator W dan sebaliknya bagian positif akan berputar menuju medan magnet stator U dan V. Selanjutnya ketika V memiliki medan magnet negatif dan U serta W memiliki medan magnet positif, bagian positif magnet permanen akan berputar menuju V dan bagian negatif akan menuju U dari kumparan W. Karena tegangan AC sinusoidal yang digunakan berlangsung secara kontinu, proses perubahan polaritas tegangan pada stator ini akan terjadi secara terus menerus sehingga menciptakan medan putar magnet stator dan magnet permanen rotor akan berputar

mengikuti medan putar magnet stator ini. Hal inilah yang menyebabkan rotor pada motor BLDC dapat berputar.

2.2 Prinsip Dasar *Inverter* [4]

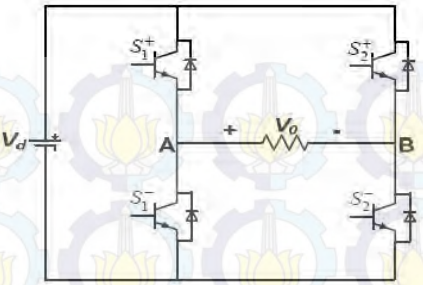
Konverter DC-AC atau biasa disebut *inverter* adalah suatu alat elektronik yang berfungsi untuk menghasilkan keluaran AC sinusoidal dari masukan DC dimana magnitudo dan frekuensinya dapat diatur. *Inverter* biasanya banyak digunakan pada kendali mesin AC dan UPS (*Uninterruptible Power Supplies*).

Dilihat dari jenis masukannya, *inverter* dibagi menjadi dua macam yaitu VSI (*Voltage Source Inverter*) dimana masukannya adalah sumber tegangan DC dan CSI (*Current Source Inverter*) dimana masukannya adalah sumber arus DC. Pada prakteknya, *inverter* yang lebih sering digunakan adalah VSI sedangkan CSI penggunaanya terbatas pada kontrol motor AC dengan daya yang sangat besar.

Salah satu teori dari *inverter* adalah teori *full bridge converter*. *Full bridge converter* adalah rangkaian teori dasar yang digunakan untuk mengubah DC ke AC. *Full bridge converter* mempunyai pasangan saklar (S_1,S_2) dan (S_3,S_4). Keluaran AC didapatkan dari masukan DC dengan membuka dan menutup saklar-saklar pada urutan yang tepat. Tegangan keluaran Vo bisa berupa +Vdc, -Vdc, atau nol, tergantung pada saklar yang mana tertutup.

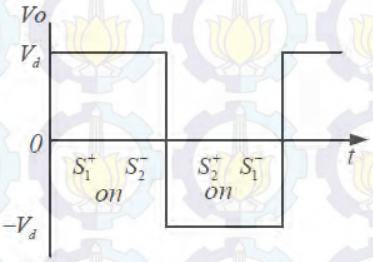
Gambar 2.4 merupakan gambar dari salah satu topologi *inverter bridge*, yaitu *full-bridge* dengan sumber DC yang digunakan adalah sumber tegangan.

Pada dasarnya, untuk menghasilkan keluaran AC sinusoidal, *inverter* bekerja dengan mengatur penyaklaran masukan sumber DC. Dalam satu lengan, transistor yang boleh menyala hanya satu karena apabila dua transistor dalam satu lengan menyala maka sumber tegangan DC akan terhubung singkat. Dengan demikian pada saat S_1^+ menyala maka S_1^- akan mati. Hal yang sama terjadi pada S_2^+ dan S_2^-.



Gambar 2.4 Topologi *Inverter Full-Bridge*

Pada saat S_1^+ dan S_2^- menyala, beban akan merasakan tegangan $V_d\,(V_o = V_d)$. Pada saat S_2^+ dan S_1^- menyala maka beban akan merasakan tegangan $V_d\,(V_o = -V_d)$. Bentuk sinyal tegangan keluaran dari gambar 2.4 adalah sebagai berikut :



Gambar 2.5 Bentuk Tegangan Keluaran *Inverter*

Nilai rms tegangan keluaran dapat dicari dengan rumus :

$$V_o = \left(\frac{2}{T_o}\int_0^{\frac{T_0}{2}} V^2 dt\right)^{\frac{1}{2}} = V_s \qquad (2.3)$$

Keluaran *inverter* dengan penyaklaran seperti diatas adalah gelombang persegi. Gelombang seperti ini memiliki kandungan harmonisa yang besar. Pada umumnya keluaran *inverter* yang diinginkan adalah bentuk gelombang sinus murni karena gelombang sinus murni tidak mengandung harmonisa. Untuk mendapatkan bentuk gelombang sinusoidal maka teknik penyaklaran transistor harus diatur.

Salah satu teknik yang paling umum digunakan dalam mengatur penyaklaran transistor untuk *inverter* adalah PWM (*Pulse Width Modulation*).

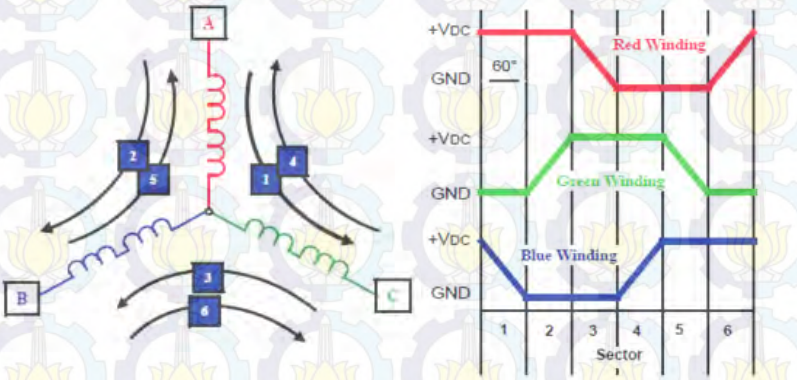
2.3 Metode Pengendalian Motor BLDC[5]

Terdapat beberapa metode yang dapat digunakan dalam pengendalian motor BLDC, diantaranya adalah metode *six-step* dan metode sinyal PWM.

2.3.1 Metode *Six-Step*

Metode Six-Step adalah metode yang paling sering digunakan dalam pengendalian BLDC. Hal ini disebabkan karena metode ini sederhana sehingga mudah diimplementasikan. Hanya saja metode ini memiliki kelemahan yaitu arus RMS (Root Mean Square) yang tinggi. Ini dapat terjadi karena PWM yang digunakan dalam metode ini merupakan PWM square dengan frekuensi tertentu sehingga menciptakan gelombang AC yang bebentuk *trapezoid* atau *square*. Akibat dari gelombang yang beebentuk square atau trapezoid adalah timbulnya gelombang harmonik. Gelombang harmonik inilah yang mengakibatkan motor berputar.

Setiap langkah atau sector adalah ekuivalen dengan 60 derajat elektrikal. 6 sektor menjadi 360 derajat elektrikal atau satu putaran elektrikal.



Gambar 2.6 Komutasi Six-Step

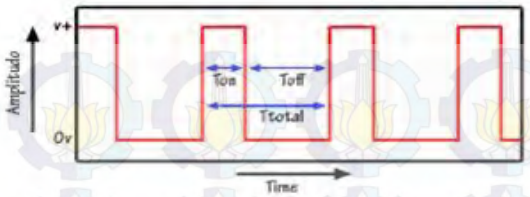
Tanda panah pada kumparan menunjukan arah di mana arus mengalir melalui kumparan-kumparan motor setiap langkah pada Six-Step. Sedangkan untuk urutan langkah komutasi adalah sebagai berikut:

1. Kumparan A diberi tegangan positif, kumparan B tidak diberi tegangan dan kumparan C diberi tegangan negatif.
2. Kumparan A diberi tegangan positif, kumparan B diberi tegangan negatif dan kumparan C tidak diberi tegangan.
3. Kumparan A tidak diberi tegangan, kumparan B diberi tegangan negatif dan kumparan C diberi tegangan positif.
4. Kumparan A diberi tegangan negatif, kumparan B tidak diberi tegangan, dan kumparan C diberi tegangan positif.
5. Kumparan A diberi tegangan negatif, kumparan B diberi tegangan positif, dan kumparan C tidak diberi tegangan.
6. Kumparan A tidak diberi tegangan, kumparan B diberi tegangan positif, dan kumparan C diberi tegangan negatif.

Metode ini disebut *six-step* karena mampu menciptakan gelombang *trapezoidal* atau *square* yang menyerupai gelombang sinusoidal, digunakan PWM *square* yang terdiri dari 6 bagian yaitu 2 bagian positif dan 2 bagian negatif, dan 2 bagian *floating*. Masing-masing bagian besarnya 60 derajat gelombang sinusoidal. Kondisi *floating* pada algoritma ini adalah kondisi ketika gelombang sinusoidal bepotongan pada titik 0.

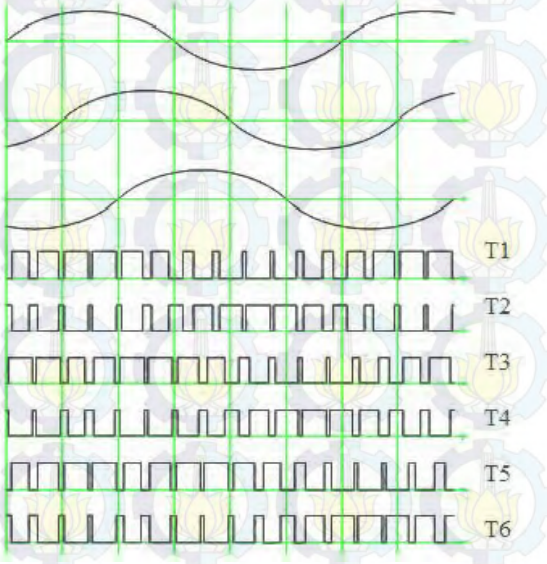
2.3.2 Metode sinyal PWM (*Pulse Width Modulation*)

Pulse width modulation (PWM) secara umum adalah sebuah manipulasi lebar sinyal yang dinyatakan dengan pulsa dalam satu perioda, untuk mendapatkan tegangan rata-rata yang berbeda. Beberapa contoh aplikasi PWM adalah pemodulasian data untuk telekomukasi, pengontrolan daya atau tegangan yang masuk ke beban, regulator tegangan, *audio effect* dan penguatan serta aplikasi lainnya.



Gambar 2.7 Sinyal PWM

Dalam implementasi agar dapat mengendalikan keenam transistor pada driver, sinyal PWM sinusoidal yang didapatkan dibagi menjadi 6 bagian atau step. Masing-masing bagian atau step besarnya 60 derajat. Ini disebabkan karena perbedaan tiap fasa dari sinyal 3 fasa adalah 120 derajat dan tiap 60 derajat terdapat gelombang sinusoidal yang bepotongan dengan nilai 0. Oleh karena itu sinyal PWM harus dibagi menjadi 6 bagian untuk menunjang proses komutasi pada BLDC. Berikut ini implementasi dari algoritma PWM sinussoidal.



Gambar 2.8 Algoritma PWM Sinussoidal

Kecepatan motor BLDC tergantung dari tegangan yang diaplikasikan pada kumparan. Metode PWM digunakan untuk mengendalikan kecepatan motor, sinyal PWM diaplikasikan kesaklar S1 –S6 untuk menetukan rata-rata tegangan pada kumparan.

2.4 Rem Magnetik [6]

Sistem pengereman magnetik menggunakan gaya magnetik untuk memperlambat suatu gerakan, dimana pada umumnya merupakan gerakan poros. Terdapat sebuah piringan berbahan logam non-feromagnetik terpasang dengan poros yang berputar. Piringan tersebut diapit oleh sisi stator berupa sistem lilitan magnetik yang dapat membangkitkan medan magnet dari aliran listrik.

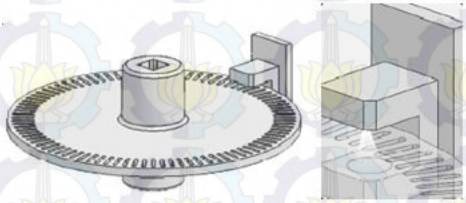
Arus listrik menimbulkan medan magnet pada lilitan dan logam piringan yang memotong medan magnet tersebut akan menimbulkan arus *eddy* pada piringan itu sendiri. Arus *eddy* ini akan menimbulkan medan magnet yang arahnya berlawanan dengan medan magnet sebelumnya, sehingga menghambat gerakan putar dari poros tersebut. Rem magnetik akan bekerja optimal untuk memberikan penurunan kecepatan, tetapi tidak untuk menghentikan gerak suatu objek. Berikut merupakan salah satu contoh dari bentuk fisik dari rem magnetik yang ditunjukkan pada Gambar 2.9.

Gambar 2.9 Bentuk Fisik Rem Magnetik

2.5 Sensor *Rotary Encoder*

Rotary Encoder adalah suatu komponen elektro mekanis yang memiliki fungsi untuk memonitoring posisi *anguler* pada suatu poros yang berputar. Dari perputaran benda tersebut data yang termonitoring

akan diubah ke dalam bentuk data digital oleh *rotary encoder* berupa lebar pulsa kemudian akan dihubungkan ke kontroler (Mikrokontroler/PLC). Berdasarkan data yang didapat berupa posisi *anguler* (sudut) kemudian dapat diolah oleh kontroler sehingga mendapatkan data berupa kecepatan, arah, dan posisi dari perputaran porosnya.



Gambar 2.10 *Optical Shaft Encoder Disk*

Penerapan dari penggunaan *rotary encoder* sering dijumpai pada robot-robot yang membutuhkan kepresisian tinggi dalam hal posisi seperti robot *Mechanum* dan robot *Omni*, selain itu untuk robot berjenis *Differential Drive* (ex : robot tank) juga disarankan menggunakan *rotary encoder* untuk mengatur agar kecepatan putar motor di roda kiri dan kanan bisa sama.

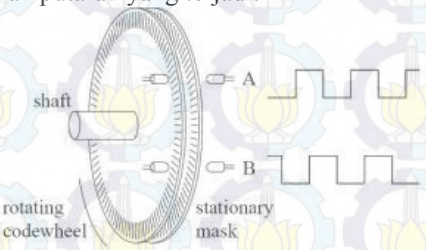
Konstruksi *rotary encoder* berupa piringan tipis yang biasanya di kopel dengan poros yang berputar, umumnya di kopel langsung dengan *shaft* motor. Piringan tipis tersebut terdapat lubang di sepanjang pinggir lingkarannya. Di bagian sisi-sisi piringan terdapat sebuah LED dan *phototransistor* di bagian bersebrangan. Fungsi dari lubang-lubang yang berada di sepanjang pinggir lingkaran tersebut akan menghantarkan cahaya LED ke *phototransistor*, sebaliknya jika cahaya LED tidak menembus lubang piringan maka cahaya akan tertahan. Piringan tersebut akan berputar sesuai dengan kecepatan putaran motor sehingga *phototransistor* akan saturasi ketika cahaya LED menembus lubang-lubangnya.

Pada saat saturasi *phototransistor* akan menghasilkan pulsa dengan range +0.5V s/d +5V. Semakin banyak lubang yang berada pada piringan tentu saja semakin banyak pulsa yang dihasilkan selama satu putaran, hal tersebut berbanding lurus dengan tingkat akurasi yang dihasilkan oleh *rotary encoder*. Ada 2 jenis *rotary* yang umum beredar di pasaran yaitu *incremental rotary encoder* dan *absolute rotary encoder*.

2.5.1 *Incremental Rotary Encoder*

Tipe *incremental rotary encoder* merupakan tipe *rotary encoder* yang paling sederhana karena hanya dapat mengukur perubahan sudut relatifnya saja. Karena kurangnya akurasi dari *incremental rotary encoder* ini perlu ditambahkan satu lagi sensor optik untuk menentukan arah putaran porosnya. Dua buah sensor optik dipasang pada sudut yang berbeda sehingga arah putaran dapat diketahui, biasanya sering disebut *Channel* A dan *Channel* B (Gambar 2.11).

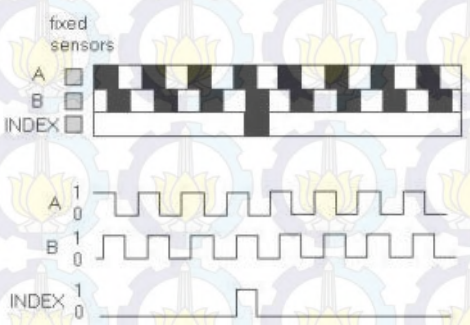
Ketika poros berputar, deretan pulsa akan muncul di masing-masing channel pada frekuensi yang proporsional dengan kecepatan putar sedangkan hubungan fasa antara *channel* A dan B menghasilkan arah putaran. Dengan menghitung jumlah pulsa yang terjadi terhadap resolusi piringan maka putaran dapat diukur. Untuk mengetahui arah putaran, dengan mengetahui *channel* mana yang *leading* terhadap *channel* satunya dapat kita tentukan arah putaran yang terjadi karena kedua *channel* tersebut akan selalu berbeda fasa seperempat putaran (*quadrature signal*). Seringkali terdapat *output channel* ketiga, disebut INDEX, yang menghasilkan satu pulsa per putaran berguna untuk menghitung jumlah putaran yang terjadi.



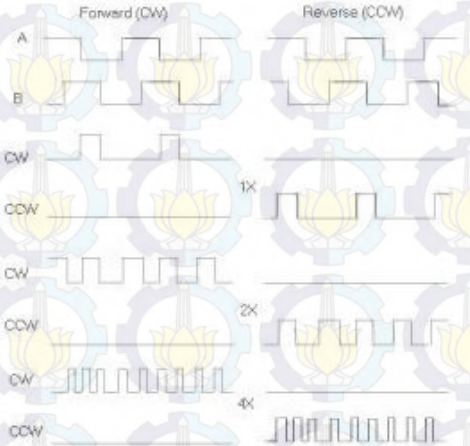
Gambar 2.11 Susunan Piringan untuk *Incremental Encoder*

Contoh pola diagram keluaran dari suatu *incremental encoder* ditunjukkan pada Gambar 2.12. Resolusi keluaran dari sinyal *quadrature* A dan B dapat dibuat beberapa macam, yaitu 1X, 2X dan 4X. Resolusi 1X hanya memberikan pulsa tunggal untuk setiap siklus salah satu sinya A atau B, sedangkan resolusi 4X memberikan pulsa setiap transisi pada kedua sinyal A dan B menjadi empat kali resolusi 1X. Arah putaran dapat ditentukan melalui level salah satu sinyal selama transisi terhadap sinyal yang kedua. Pada contoh resolusi 1X, A = arah bawah dengan B = 1 menunjukkan arah putaran searah jarum

jam, sebaliknya B = arah bawah dengan A = 1 menunjukkan arah berlawanan jarum jam.



Gambar 2.12 Contoh Pola Keluaran *Incremental Encoder*



Gambar 2.13 *Output* dan Arah Putaran pada Resolusi yang Berbeda-Beda

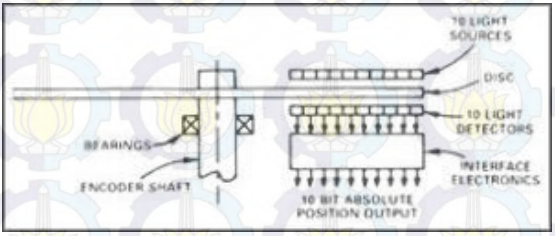
2.5.2 *Absolute Rotary Encoder*

Absolute encoder menggunakan piringan dan sinyal optik yang diatur sedemikian sehingga dapat menghasilkan kode digital untuk menyatakan sejumlah posisi tertentu dari poros yang dihubungkan padanya. Piringan yang digunakan untuk *absolut encoder* tersusun dari segmen-segmen cincin konsentris yang dimulai dari bagian tengah

piringan ke arah tepi luar piringan yang jumlah segmennya selalu dua kali jumlah segmen cincin sebelumnya. Cincin pertama di bagian paling dalam memiliki satu segmen transparan dan satu segmen gelap, cincin kedua memiliki dua segmen transparan dan dua segmen gelap, dan seterusnya hingga cincin terluar. Sebagai contoh apabila *absolut encoder* memiliki 16 cincin konsentris maka cincin terluarnya akan memiliki 32767 segmen. Gambar 2.14 menunjukkan pola cincin pada piringan *absolut encoder* yang memiliki 16 cincin.

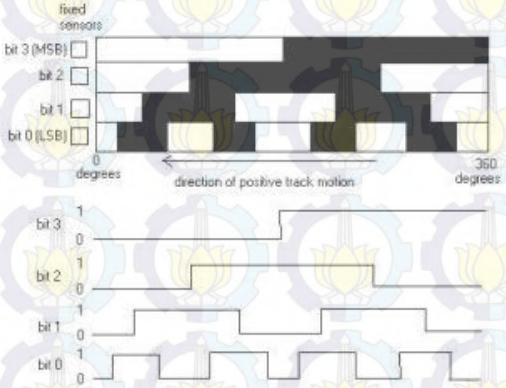
Gambar 2.14 Contoh Susunan Pola 16 Cincin Konsentris pada *Absolut Encoder*

Bilangan yang dihasilkan saat pembacaan nilai terhadap *absolute rotary encoder* ini adalah berupa bilangan biner karena memilki kelipatan dua dari setiap cincinnya. Untuk menghasilkan sistem biner pada susunan cincin maka diperlukan pasangan LED dan *phototransistor* sebanyak jumlah cincin yang ada pada *absolut encoder* tersebut.



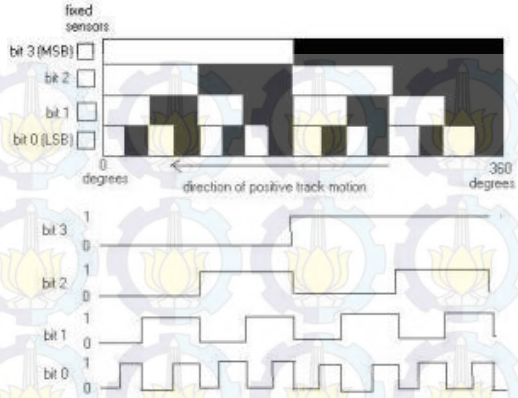
Gambar 2.15 Contoh Susunan Pola 16 Cincin Konsentris pada *Absolut Encoder*

Sistem biner yang untuk menginterpretasi posisi yang diberikan oleh *absolute encoder* dapat menggunakan kode *gray* atau kode biner biasa, tergantung dari pola cincin yang digunakan. Untuk lebih jelas, pada gambar 2.16 contoh *absolut encoder* yang hanya tersusun dari 4 buah cincin untuk membentuk kode 4 bit. Apabila *encoder* ini dihubungkan pada poros, maka *phototransistor* akan mengeluarkan sinyal persegi sesuai dengan susunan cincin yang digunakan. Gambar 5 dan 6 menunjukkan contoh perbedaan diagram keluaran untuk *absolute encoder* tipe *gray code* dan tipe *binary code*.



Gambar 2.16 Contoh Diagram Keluaran *Absolut Encoder* 4-Bit Tipe *Gray Code*

Dengan *absolute encoder* 4-bit ini maka akan didapatkan 16 informasi posisi yang berbeda yang masing-masing dinyatakan dengan kode biner atau kode *gray* tertentu. Tabel 1 menyatakan posisi dan *output* biner yang bersesuaian untuk *absolut encoder* 4-bit. Dengan membaca *output* biner yang dihasilkan maka posisi dari poros yang kita ukur dapat kita ketahui untuk diteruskan ke rangkaian pengendali. Semakin banyak bit yang kita pakai maka posisi yang dapat kita peroleh akan semakin banyak.



Gambar 2.17 Contoh Diagram Keluaran *Absolut Encoder* 4-Bit Tipe *Binary Code*

Tabel 2.1 *Switching Table* dari vektor tegangan *inverter*

Desimal	Rentang Putaran	Kode Biner	Kode *Gray*
0	0 – 22.5	0000	0000
1	22.5 - 45	0001	0001
2	45 – 67.5	0010	0011
3	67.5 - 90	0011	0010
4	90 – 112.5	0100	0110
5	112.5 - 135	0101	0111
6	135 – 157.5	0110	0101
7	157.5 - 180	0111	0100
8	180 – 202.5	1000	1100
9	202.5 - 225	1001	1101
10	225 – 247.5	1010	1111
11	247.5 - 270	1011	1110
12	270 – 292.5	1100	1010
13	292.5 - 315	1101	1011
14	315 – 337.5	1110	1001
15	337.5 - 360	1111	1000

2.6 Identifikasi Sistem [7]

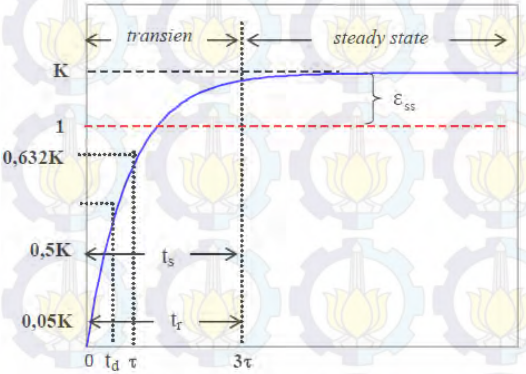
Metode identifikasi merupakan suatu metode yang menggunakan hubungan data masukan dan keluaran yang selanjutnya dilakukan pengujian dan analisa dengan metode pendekatan sehingga dapat ditentukan nilai parameter secara analitik. Beberapa sinyal masukan

yang dapat digunakan untuk mendapatkan respon suatu sistem seperti sinyal *step, ramp, impulse*, dan sinusoidal.

Terdapat dua macam identifikasi sistem yaitu identifikasi statis dan identifikasi dinamis. Untuk identifikasi statis, sinyal yang diberikan berupa sinyal *step* yang konstan sampai sistem mencapai keadaan *steady state*. Sedangkan untuk identifikasi dinamis, sinyal yang digunakan berupa sinyal acak (*random*).

2.6.1 Identifikasi Statis

Suatu sistem disebut system orde satu apabila dilihat secara grafis memiliki bentuk respon seperti yang ditunjukkan pada Gambar 2.18.



Gambar 2.18 Karakteristik Respon Orde Satu

Karakteristik respon sistem orde satu dilihat berdasarkan respon sistem ketika sistem diberi masukan sinyal *step*. Model matematik sistem orde satu dapat dirumuskan dengan:

$$G(s) = \frac{K}{\tau s+1} \tag{2.4}$$

Dimana K menyatakan *gain overall* dan τ menyatakan *time constant* (konstanta waktu). Nilai K merupakan hasil perhitungan dari persamaan 2.5 :

$$K = \frac{Y_{ss}}{X_{ss}} \tag{2.5}$$

Y_{ss} adalah keluaran saat *steady state* dan X_{ss} adalah masukan saat *steady state*. Karakteristik sistem orde satu dibedakan menjadi karakteristik respon transien dan karakteristik respon keadaan tunak atau *steady state*. Karakteristik respon transien pada orde satu terdiri dari :

1. Spesifikasi Teoritis

   Time constant atau konstanta waktu (τ) merupakan waktu yang dibutuhkan respon mulai t = 0 sampai respon mencapai 63,2% dari respon *steady state*. Konstanta waktu menyatakan kecepatan respon sistem.

2. Spesifikasi Praktis

   a. *Settling time* atau waktu tunak (t_s) merupakan waktu yang menyatakan bahwa respon sistem telah masuk pada daerah *steady state*. Jika dihubungkan dengan konstanta waktu τ, maka t_s dapat dirumuskan dengan :

   $$t_s\ (\pm 5\%) \approx 3\tau \quad (2.6)$$
   $$t_s\ (\pm 2\%) \approx 4\tau \quad (2.7)$$
   $$t_s\ (\pm 0.5\%) \approx 5\tau \quad (2.8)$$

   b. *Rise time* atau waktu naik (t_r) merupakan waktu yang menyatakan bahwa respon sistem telah naik dari 5% - 95% atau 10% - 90% dari nilai respon *steady state*.

   $$t_r\ (5\% - 95\%) = \tau\ ln\ 19 \quad (2.9)$$
   $$t_r\ (10\% - 90\%) = \tau\ ln\ 9 \quad (2.10)$$

   c. *Delay time* atau waktu tunda (t_d) merupakan waktu yang dibutuhkan respon mulai t = 0 sampai respon mencapai 50% dari nilai *steady state*. Waktu tunda menyatakan besarnya faktor keterlambatan respon akibat proses *sampling*.

   $$t_d = \tau\ ln\ 2 \quad (2.11)$$

Karakteristik respon *steady state* sistem orde satu diukur berdasarkan kesalahan pada keadaan tunak atau *error steady state* (e_{ss}), yaitu :

$$e_{ss} = Y_{ss} - X_{ss} \quad (2.12)$$

2.6.1.1 Identifikasi Statis dengan Metode Vítečková Orde 1

Metode ini menggunakan pendekatan orde satu dengan kemungkinan adanya waktu tunda. Fungsi alih untuk metode Vitečková Orde 1 ditunjukkan pada Persamaan 2.12.

$$G_{V1}(s) = \frac{K}{\tau_{V1}\, s + 1} e^{-T_{dV1}\, s} \quad (2.13)$$

Dimana T_{dV1} merupakan waktu tunda (*delay time*) dan dapat dicari menggunakan persamaan :

$$T_{dV1} = 1{,}498\ t_{33} - 0{,}498\ t_{70} \quad (2.14)$$

Kemudian τ_{V1} merupakan konstanta waktu dengan persamaan :

$$\tau_{V1} = 1{,}245\ (t_{70} - t_{33}) \quad (2.15)$$

Dimana t_{33} dan t_{70} merupakan waktu saat respon berada pada kodisi 33% dan 70% dari keluaran *steady state*. Apabila T_{dV1} bernilai negatif, maka sistem dianggap tidak memiliki waktu tunda.

2.6.1.2 Identifikasi Statis dengan Metode Vítečková Orde 2

Metode ini menggunakan pendekatan orde dua dengan kemungkinan adanya waktu tunda. Fungsi alih untuk metode Vitečková Orde 2 ditunjukkan pada Persamaan 2.15.

$$G_{V2}(s) = \frac{K}{(\tau_{V2}\, s + 1)^2} e^{-T_{dV2}\, s} \quad (2.16)$$

Dimana T_{dV2} merupakan waktu tunda (*delay time*) dan dapat dicari menggunakan persamaan :

$$T_{dV2} = 1{,}937\ t_{33} - 0{,}937\ t_{70} \quad (2.17)$$

Kemudian τ_{V2} merupakan konstanta waktu dengan persamaan :

$\tau_{V2} = 0{,}794\ (t_{70} - t_{33})$ (2.18)

Dimana t_{33} dan t_{70} merupakan waktu saat respon berada pada kodisi 33% dan 70% dari keluaran *steady state*. Apabila T_{dV2} bernilai negatif, maka sistem dianggap tidak memiliki waktu tunda.

2.6.1.3 Identifikasi Statis dengan Metode Sundaresan & Krishnaswamy

Metode ini menggunakan pendekatan orde satu dengan kemungkinan adanya waktu tunda. Fungsi alih untuk metode Sundaresan & Krishnaswamy ditunjukkan pada Persamaan 2.19.

$$G_{SK}(s) = \frac{K}{\tau_{SK}\, s + 1} e^{-T_{dSK}\, s} \quad (2.19)$$

Dimana T_{dSK} merupakan waktu tunda dan dapat dicari menggunakan persamaan :

$T_{dSK} = 1{,}3\ t_{35,3} - 0{,}29\ t_{85,3}$ (2.20)

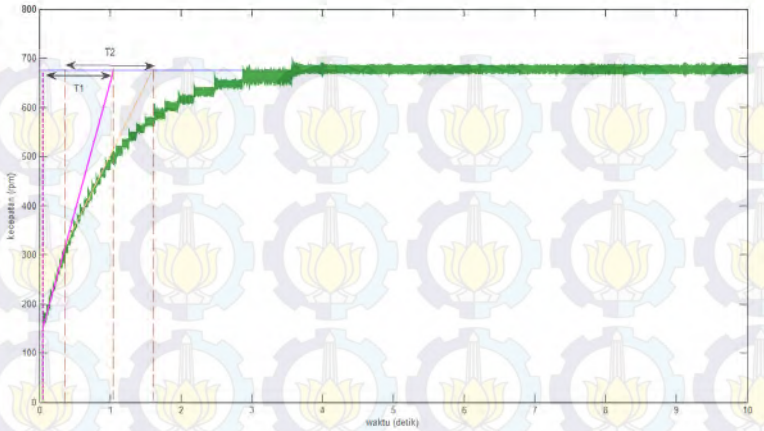
Kemudian τ_{SK} merupakan konstanta waktu dengan persamaan :

$\tau_{SK} = 0{,}67\ (t_{85,3} - t_{35,3})$ (2.21)

Dimana $t_{35,3}$ dan $t_{85,3}$ merupakan waktu saat respon berada pada kodisi 35,3% dan 85,3% dari keluaran *steady state*. Apabila τ_{dSK} bernilai negatif, maka sistem dianggap tidak memiliki waktu tunda.

2.6.1.4 Identifikasi Statis dengan Metode Grafis Terstruktur

Metode ini merupakan secara grafis dengan menarik garis singgung yang memotong respon plant.



Gambar 2.19 Metode Grafik Terstruktur

T_1 dan T_2 merupakan waktu yang digunakan untuk menentukan akar-akar karakteristik dari sistem. Langkah – langkah untuk mendapatkan T_1 dan T_2 adalah sebagai berikut ;

1. Tarik garis singgung dari awal respon sampai garis tersebut mempunyai kemiringan yang sama dengan respon. Garis akan berakhir pada Xss.
2. Tarik garis lurus dari awal repon sampai Xss.
3. Selisih antara garis tersebut merupakan nilai T_1.
4. Tarik lagi garis yang menyinggung dari akhir garis singgung yang pertama sampai menuju Xss.
5. Ulangi lagi langkah 2 pada mulai dari awal garis 2 berpotongan dengan respon.
6. Nilai T_2 adalah selisih 2 garis tersebut.
7. Setelah nilai T_1 dan T_2 didapatkan maka bentuk didapatkan fungsi transfer sebagai berikut :

$$G\ (s) = \frac{1/t1}{s+1/t1} + \frac{1/t2}{s+1/t2} \qquad (2.22)$$

2.7 Validasi Model

Identifikasi parameter sistem didapatkan data untuk mendapatkan model matematik *plant*. Tujuan validasi model secara umum adalah untuk membuktikan bahwa model yang diidentifikasi memenuhi persyaratan permodelan menurut kriteria tujuan (*objective*) dari aproksimasi permodelan yang baik. Model matematika tersebut perlu diuji validasi untuk mengetahui kesamaan dengan *plant* dalam kondisi nyata. Terdapat beberapa metode validasi model, salah satunya yaitu *Root Mean Square Error* seperti uji validasi model untuk mengetahui perbandingan dengan data yang didapat dengan model matematika melalui perhitungan yang disimulasikan.

RMSE adalah pengukuran akurasi pada nilai deret waktu secara statistik, khususnya *trend*. Persamaan (2.23) menyajikan formulasi dasar dari RMSE.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} {e_i}^2} \qquad i = 1,2,3,4\ldots n \tag{2.23}$$

Dimana e adalah persentase kesalahan data hasil dari permodelan terhadap hasil pengukuran seperti yang dirumuskan pada Persamaan (2.24).

$$e_i = A_i - M_i \quad i = 1,2,3,4,..n \tag{2.24}$$

Keterangan :

n : jumlah data
i : urutan data
A : nilai data hasil pengukuran
M : nilai data hasil permodelan

RMSE adalah komponen terpadu dari sebuah model statistika seperti regresi.Hal ini menjadikan RMSE sebagai sebuah pengukuran alami yang digunakan dalam berbagai evaluasi terhadap perkiraan kesalahan yang menggunakan metode statistika khususnya regresi. Secara umum, tidak ada kriteria mutlak dari sebuah data untuk dianggap bagus. Keuntungan yang dimiliki RMSE adalah kesamaan skala yang dimiliki oleh data hasil pengukuran dengan data hasil permodelan. Dengan demikian, RMSE dapat merepresentasikan ukuran dari kesalahan rata-rata. Nilai mutlak dari penghitungan ini dijumlahkan untuk setiap titik yang dilengkapi atau perkiraan dalam domain waktu

dan dibagi kembali dengan jumlah n pengambilan data. Hal ini membuat kesalahan diukur dalam persen sehingga kesalahan sebuah deret waktu dapat dibandingkan di tingkat yang berbeda. Selain mudah dihitung, RMSE memiliki aplikasi yang luas baik dalam bidang penaksiran maupun statistika.

2.8 Teori Kontroler Logika *Fuzzy* [8]

Sebelum konsep logika *fuzzy* diperkenalkan, orang telah mengenal konsep yang disebut logika klasik yang membagi sifat parameter menjadi dua hal yang berlawanan secara tegas, seperti benar atau salah, 0 atau 1. Konsep ini ternyata mempunyai kekurangan dalam penerapannya di kehidupan nyata, karena manusia lebih mengenal konsep linguistik yang menyatakan sesuatu dengan tidak eksak atau samar. Konsep logika *fuzzy* mengubah konsep logika klasik menjadi konsep yang memetakan suatu variabel pada kemungkinan yang tidak eksak sehingga didapatkan sistem linguistik dan permasalahan yang tidak pasti atau tidak presisi serta permasalahan probabilitas.

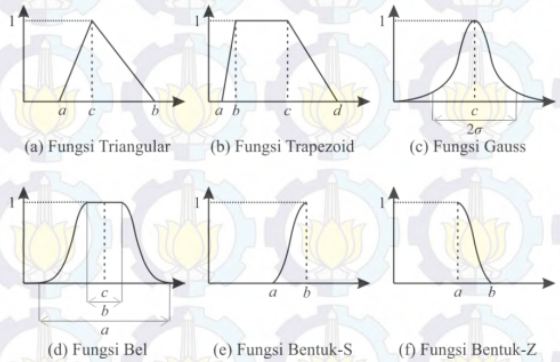
Konsep logika *fuzzy* berawal dari teori *fuzzy set* sebagai salah satu pendekatan untuk menyelesaikan permasalahan yang tidak memiliki ketentuan yang pasti. Teori tersebut dikembangkan oleh Lotfi Zadeh pada tahun 1965 di University of California – Barkeley. Dalam jurnalnya, Zadeh memperkenalkan konsep teori himpunan baru yang dinamakan *fuzzy* set. Konsep logika *fuzzy* menggantikan konsep "benar-salah" dari logika *boolean* menjadi derajat tingkat kebenaran. Teori *fuzzy* menyatakan keanggotaan suatu objek ke dalam fungsi derajat keanggotaan (*membership function*). Hal tersebut memungkinkan keanggotaan suatu objek dapat dinyatakan pada semua bilangan *riil* antara 0 sampai 1. Oleh karena itu, konsep *fuzzy* tersebut sesuai dengan pola pikir manusia yang cenderung menilai suatu objek secara samar.

2.8.1 Himpunan *Fuzzy* (*Fuzzy Set*)

Himpunan *fuzzy* merupakan suatu himpunan yang beranggotakan sejumlah istilah dalam pengertian bahasa yang menyatakan level kualitatif dari semesta pembicaraan. Misalnya pengukuran kecepatan putaran motor dapat diterjemahkan ke dalam beberapa istilah bahasa yang menyatakan level kualitatif dari kecepatan putaran motor. Sehingga apabila semesta pembicaraan berupa kecepatan putaran motor, maka dapat dibuat suatu himpunan *fuzzy* yaitu "Sangat Lambat", "Lambat, "Sedang", "Cepat", "Sangat Cepat".

2.8.2 Fungsi Keanggotaan (*Membership Function*)

Fungsi keanggotaan merupakan suatu fungsi yang didefinisikan untuk suatu anggota himpunan *fuzzy* yang menggambarkan derajat kebenaran suatu kejadian dalam semesta pembicaraan. Fungsi keanggotaan dinyatakan dalam tingkat keanggotaan dengan nilai antara 0 s/d 1. Beberapa fungsi keanggotaan yang biasa digunakan untuk merepresentasikan nilai keanggotan dari suatu himpunan *fuzzy* diantaranya seperti Gambar 2.20 yang menunjukkan bentuk-bentuk dari fungsi keanggotaan tersebut.



Gambar 2.20 Bentuk-Bentuk *Membership Function*

Persamaan dari masing-masing *membership function* dapat dinyatakan seperti Persamaan 2.25 - 2.28.

$$Segitiga : \mu_A(x) = \begin{cases} 0 & x \le a \\ \dfrac{x-a}{b-a} & a \le x \le b \\ \dfrac{c-x}{c-b} & b \le x \le c \\ 0 & x \ge c \end{cases} \tag{2.25}$$

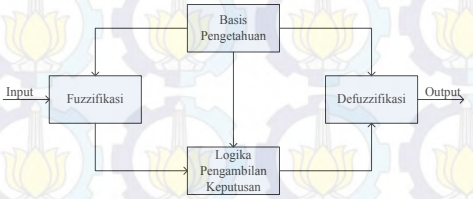
$$Bell\text{-}shaped : \mu_A(x) = exp\left[-\left(\frac{u_i - \bar{x}}{\sigma} \right)^2 \right] \tag{2.26}$$

$$Trapesium : \mu_A(x) = \begin{cases} 0 & x \le a \\ \dfrac{x-a}{b-a} & a \le x \le b \\ 1 & b \le x \le c \\ \dfrac{c-x}{c-b} & x \ge c \end{cases} \quad (2.27)$$

$$Singleton : \mu_A(x) = 1, \quad (2.28)$$

2.8.3 Struktur Dasar Logika *Fuzzy*

Kontroler logika *fuzzy* merupakan suatu kontroler yang proses perhitungan sinyal kontrolnya melalui operasi himpunan *fuzzy* meliputi proses *fuzzifikasi*, relasi *fuzzy*, inferensi *fuzzy* serta *defuzzifikasi*. Pada dasarnya struktur logika *fuzzy* tampak seperti Gambar 2.21 berikut:



Gambar 2.21 Struktur Logika *Fuzzy*

Gambar 2.21 menunjukkan struktur logika *fuzzy*. Fungsi dari bagian-bagian di atas adalah sebagai berikut:

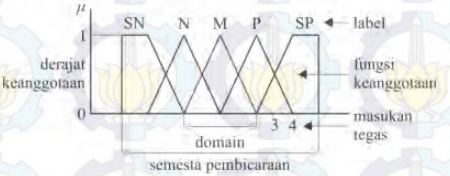
1. *Fuzzifikasi* berfungsi untuk mentransformasikan sinyal *input* yang bersifat *crisp* (bukan *fuzzy*) ke himpunan *fuzzy* dengan menggunakan operator *fuzzifikasi*.
2. Basis pengetahuan berisi basis data dan aturan dasar yang mendefinisikan himpunan *fuzzy* atas daerah-daerah *input* dan *output* dan menyusunnya dalam perangkat aturan kontrol.
3. Logika pengambilan keputusan merupakan inti dari logika *fuzzy* yang mempunyai kemampuan seperti manusia dalam mengambil keputusan. Aksi atur *fuzzy* disimpulkan dengan menggunakan implikasi *fuzzy* dan mekanisme inferensi *fuzzy*.

4. Defuzzifikasi berfungsi untuk mentransformasikan kesimpulan tentang aksi atur yang bersifat *fuzzy* menjadi sinyal sebenarnya yang bersifat *crisp* dengan menggunakan operator fuzzifikasi.

2.8.4 Fuzzifikasi

Fuzzifikasi adalah proses pemetaan *input* dan *output* sistem agar sesuai dengan himpunan *fuzzy*. Pemetaan digunakan dengan cara yang disebut fungsi keanggotaan (*membership function*). Ada banyak metode yang digunakan untuk proses fuzzifikasi seperti yang telah dijelaskan sebelumnya, tetapi bentuk *triangular* dan *trapezoid* yang paling banyak digunakan dalam pembentukan fungsi keanggotaan pada logika *fuzzy*. Hal ini dikarenakan metode bentuk *triangular* dan *trapezoid* lebih mudah diimplementasikan pada kontroler.

Struktur fungsi keanggotaan *fuzzy* dapat dilihat pada Gambar 2.22. Derajat keanggotaan adalah derajat dari masukkan tegas pada sebuah fungsi keanggotaan, memiliki nilai 0 s/d 1. Fungsi keanggotaan mendefinisikan nilai *fuzzy* dengan melakukan pemetaan nilai tegas berdasarkan daerahnya untuk diasosiasikan dengan derajat keanggotaan.



Gambar 2.22 Struktur fungsi keanggotaan *fuzzy*

Masukkan tegas pada umumnya merupakan hasil pengukuran parameter eksternal dari sistem kontrol. Label merupakan deskripsi nama untuk menunjukkan suatu fungsi keanggotaan *fuzzy* yang memiliki domain (lebar fungsi keanggotaan *fuzzy*) tertentu. Semesta pembicaraan memiliki jarak yang mencakup seluruh masukkan tegas yang mungkin ada. Bentuk fungsi keanggotaan harus mewaliki variabel masukkan tegas, namun bentuk yang digunakan dibatasi oleh kemampuan *tool* dalam melakukan perhitungan. Bentuk fungsi yang rumit membutuhkan persamaan fungsi yang lebih kompleks.

2.8.5 Aturan Dasar *Fuzzy*

Aturan dasar *fuzzy* adalah kaidah dasar yang berisi aturan-aturan secara linguistik yang menunjukkan kepakaran terhadap *plant*. Penentuan aturan dasar yang dipakai dalam mengontrol suatu *plant* dapat melalui metode heuristik maupun deterministik. Metode heuristik didasarkan pada pengetahuan terhadap *plant* dan perilaku dari *plant* yang akan dikontrol. Sedangkan metode deterministik diperoleh melalui identifikasi struktur dan parameter dari aturan kontrol. Banyak cara menunjukkan suatu kepakaran ke dalam aturan, format yang paling umum adalah sebagai berikut :

1. Format Aturan *IF-THEN*

   Pemetaan format aturan *IF-THEN* direpresentasikan dalam Persamaan 2.29.

   IF premis THEN conclusion (2.29)

   Dimana premis berupa input kontroler dan conclusion berupa output kontroler. Dengan demikian dari kepakaran dapat diambil kesimpulan, apabila pernyataannya lebih dari satu maka dapat digunakan logika "*AND*" atau "*OR*".

   Contoh penggunaan aturan *IF-THEN* sebagai berikut :

   IF error is N THEN output is NB

   IF error is Z THEN output is Zero

   IF error is P THEN output is PB

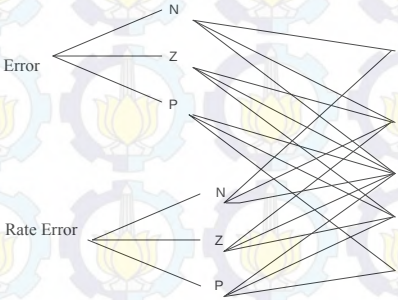
   Jumlah basis aturan dari suatu sistem *fuzzy* ditentukan dari jumlah variabel pada input dan jumlah *membership function* pada variabel masukkan, dirumuskan dalam persamaan 2.30.

$$\prod_{i=1}^{n} N_i = N_1 \times N_2 \times .... \times N_n \qquad (2.30)$$

   Dimana N_i merupakan jumlah *membership function* pada variabel input i. Sebagai contoh apabila variabel input pertama memiliki dua *membership function* dan variabel input kedua memiliki dua *membership function*, maka jumlah basis aturan adalah 2x2 = 4 aturan.

2. Format Hubungan

Pada dasarnya sama dengan aturan *IF-THEN* hanya saja tampilannya lebih sederhana karena menggunakan hubungan garis. Contoh dari penggunaan format hubungan dapat dilihat pada Gambar 2.23.



Gambar 2.23 Aturan Dasar *Fuzzy* Format Hubungan

Gambar 2.7 menjelaskan NB adalah *Negative Big*, NM adalah *Negative Medium*, Z adalah *Zero*, PM adalah *Positive Medium*, dan PB adalah *Positive Big* seperti pada Tabel 2.2.

3. Format *Tabular*

Tabel 2.2 menunjukkan format tabular lebih sederhana daripada format hubungan, variabel linguistik berada pada sisi luar dari tabel, sedangkan sisi dalam berisi dari keputusannya.

Tabel 2.2 Format Tabular

Error/Rate Error	Neg	Zero	Pos
Neg	NB	NM	Zero
Zero	NM	Zero	PM
Pos	Zero	PM	PB

2.8.6 Logika Pengambilan Keputusan

Inferensi *fuzzy* adalah suatu proses formulasi pemetaan *input* terhadap *output* dengan menggunakan logika *fuzzy*. Proses dari inferensi

fuzzy melibatkan fungsi keanggotaan operator logika *fuzzy* dan aturan *if-then*.

Terdapat dua metode inferensi yang paling dikenal, yaitu metode inferensi *Mamdani* dan metode *Takagi-Sugeno*. Metode inferensi *Mamdani* menggunakan fungsi keanggotaan *fuzzy* pada bagian *output*nya. Sehingga setelah proses aturan telah diterapkan, terdapat himpunan *fuzzy* yang harus didefuzzifikasi. Umumnya proses defuzzifikasi berlangsung lebih lambat akibat proses komputasi pada *output*nya.

Metode *Takagi-Sugeno* menggunakan fungsi keanggotaan *output* yang linier atau berupa konstanta. Sedangkan dua bagian pada proses inferensi yaitu fuzzifikasi dan penerapan operator *fuzzy* sama dengan metode inferensi *Mamdani*.

Bentuk aturan *fuzzy Takagi-Sugeno* memiliki bentuk sebagai berikut :

$$\text{If input1} = x \text{ and input2} = y, \text{ then output is } z = ax + by + c \qquad (2.31)$$

Perbedaan dari kedua metode ini terletak pada pengambilan kesimpulan logika *fuzzy*. Pada metode Mamdani, kesimpulan logika *fuzzy* berupa derajat keanggotaan sehingga dalam menyimpulkan suatu logika *fuzzy* dibutuhkan proses *defuzzifikasi*. Sedangkan pada tipe Takagi Sugeno, kesimpulan logika *fuzzy* berupa suatu persamaan sehingga tidak diperlukan proses *defuzzifikasi*. Kelebihan pada logika *fuzzy* tipe Mamdani lebih sederhana, akan tetapi diperlukan kemampuan untuk mengetahui karakteristik *plant* untuk menentukan batasan keluaran kontroler. Pada tipe Takagi-Sugeno tidak diperlukan pengetahuan mengenai karakteristik dari *plant* akan tetapi diperlukan perhitungan yang lebih rumit untuk persamaan pada bagian konsekuen.

2.8.7 Defuzzifikasi

Defuzzifikasi adalah proses yang digunakan untuk mengubah kembali variabel *fuzzy* menjadi variabel nyata, atau dengan kata lain aksi pengaturan *fuzzy* yang masih berupa himpunan, diubah menjadi nilai nyata yang berupa nilai tunggal. Banyak metode yang dapat digunakan untuk proses defuzzifikasi, diantaranya adalah metode harga rata-rata maksimum atau *Mean of Maximum* (MOM) dan metode titik pusat luasan atau *Center of Area* (COA). Berikut merupakan persamaan dari masing – masing metode :

Mean of Maximum (MOM) :

$$U_0 = \arg . \sum_{j=1}^{J} \{\max[\mu_u(u_j(T))]\} J^{-1} \; \forall u \in U(T) \qquad (2.32)$$

Center of Area (COA) :

$$U_0 = \frac{\sum_{k=1}^{m} u_k(T) \bullet \mu_u(u_k(T))}{\sum_{k=1}^{m} \mu_k(u_k(T))} \; \forall u \in U(T) \qquad (2.33)$$

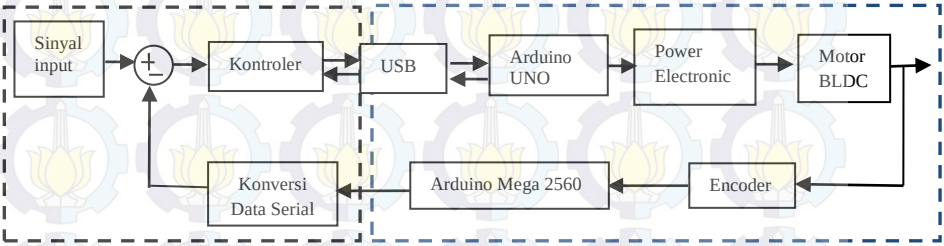
Halaman ini sengaja dikosongkan

BAB 3
PERANCANGAN SISTEM

Pada bab ini membahas mengenai perancangan keseluruhan sistem mulai dari perancangan sistem bagian *software*, sistem bagian *hardware*, identifikasi motor BLDC, dan perancangan kontroler *Fuzzy*-PI.

3.1 Diagram Blok Sistem

Gambar 3.1 menunjukkan diagram blok keseluruhan sistem dari perancangan penggerak elektrik dan pengaturan kecepatan motor BLDC.



Gambar 3.1 Diagram blok sistem

Keterangan :

: Sistem bagian *software* (PC dan *software* MATLAB)

: Sistem bagian *hardware*

Pada diagram blok menunjukkan bahwa sistem terdiri dari beberapa bagian atau subsistem. Bagian sinyal *input* merupakan bagian untuk memasukkan nilai input yang diinginkan. Blok kontroler berisi struktur kontroler *fuzzy*-PI.

Blok USB merupakan bagian dari PC yang berfungsi untuk menghubungkan ke arduino. PC disini juga berfungsi sebagai HMI (*Human Machine Interface*) yang digunakan untuk mengetahui respon keluaran sistem, membuat kontroler dan juga digunakan untuk pemberian nilai sinyal *input* yang diinginkan.

Blok *power electronic* sebagai aktuator yang digunakan untuk pemberian aksi dalam pengaturan kecepatan motor BLDC yang telah

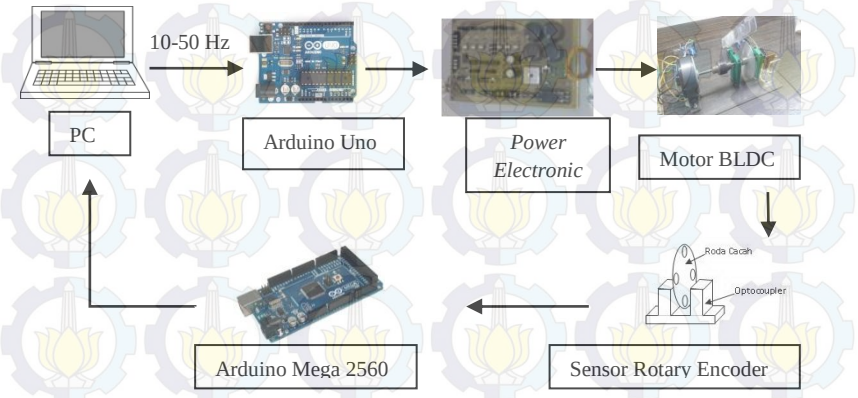
dibuat oleh kontroler. Arduino berfungsi sebagai pembangkit frekuensi fasa, PWM, dan pembacaan data dari sensor *encoder*.

Sensor *rotary encoder* berfungsi sebagai *feedback* berupa sensor kecepatan yang digunakan untuk mengetahui nilai *error* yang terjadi dalam sistem kontrol terhadap *set point*. Nilai *error* yang akan digunakan oleh kontroler untuk memberikan aksi kontrol hingga nilai *error* semakin kecil.

Sehingga secara keseluruhan sistem ini bertujuan untuk membuat rangkaian *power electronic* sebagai komutasi elektrik yang berfungsi menggerakkan motor BLDC dan menghasilkan kecepatan motor yang diinginkan. Prinsip kerja dari sistem ini yaitu rangkaian *power electronic* digunakan untuk menyuplai tegangan AC tiga fasa ke motor BLDC yang akan dioperasikan pada kecepatan tertentu sesuai dengan sinyal *input*. Sistem pengaturan digunakan untuk mengatur kecepatan motor supaya mendekati dengan sinyal *input*.

3.2 Perancangan Perangkat Keras (*Hardware)*

Untuk perancangan perangkat keras (*hardware*) dari sistem terdiri dari beberapa konponen yaitu arduino, rangkaian *power electronic*, rangkaian mekanik motor BLDC, rangkaian sensor *rotary encoder*, rem magnetik, dan PC. Arduino yang digunakan dalam sistem ini adalah arduino UNO dan arduino MEGA 2560. Komponen dan alur dari perangkat keras sistem secara spesifik ditunjukkan pada Gambar 3.2.



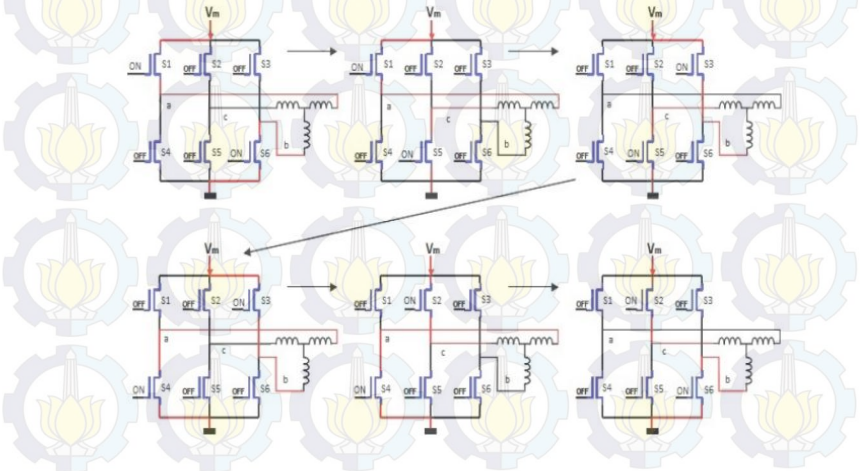
Gambar 3.2 Perancangan Perangkat Keras (*Hardware*)

Nilai sinyal *input* diberikan melalui PC dan selanjutnya dikirim ke arduino melalui *serial port* yang akan memberikan keluaran berupa frekuensi dan PWM ke rangkaian *power electronic*. *Power electronic* akan mengolah frekuensi tegangan masukan dari arduino menjadi frekuensi 10-50 Hz tiga fasa dan PWM dengan *duty cycle* 80% dengan frekuensi 4 KHz.

Sensor *rotary encoder* digunakan untuk mengukur kecepatan motor BLDC yang berputar diantara *optocoupler* tipe U. Keluaran dari sensor berupa pulsa dan dihubungkan ke arduino mega. Hasil pembacaan kecepatan dari *rotary encoder* digunakan untuk mengetahui nilai *error* yang terjadi dengan nilai sinyal *input* yang akan menjadi paremeter untuk mengubah sinyal kontrol.

3.2.1 Perancangan Rangkaian *Power Electronic*

Motor BLDC membutuhkan enam langkah komutasi yang dilakukan secara kontinyu untuk berputar. *Driver* 3 fasa terdiri dari 6 buah saklar yang akan memberikan tegangan positif (sinyal *high*) dan tegangan 0V (sinyal *low*) secara bergantian. Dimana MOSFET (*The metal–oxide–semiconductor field-effect transistor*) akan digunakan sebagai saklar dengan IC pembagi fasa 74HC175 yang akan mengendalikan fase yang masuk *gate* MOSFET tersebut.



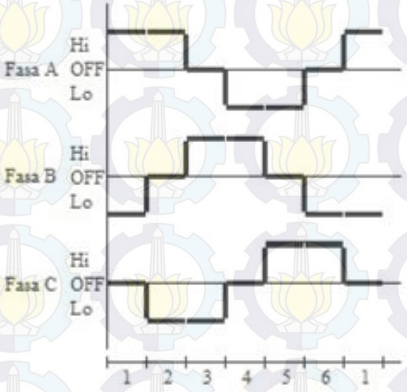
Gambar 3.3 Urutan Pensaklaran pada *Stator*

Sesuai gambar diatas, maka terdapat urutan penyaklaran pada *stator* yang dapat dilihat pada tabel berikut :

Tabel 3.1 Urutan Pengaturan Saklar motor BLDC

Urutan ke-	Saklar Aktif		Fasa A	Fasa B	Fasa C
1	S1	S6	High	Low	Off
2	S1	S5	High	Off	Low
3	S3	S5	Off	High	Low
4	S3	S4	Low	High	Off
5	S2	S4	Low	Off	High
6	S2	S6	Off	Low	High

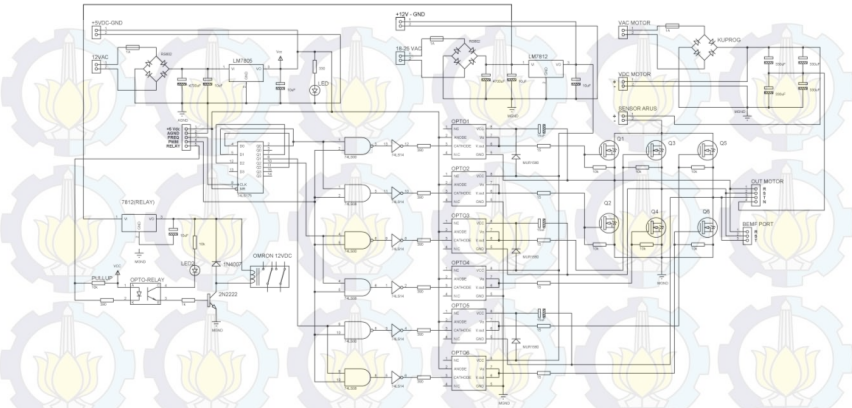
Kemudian berikut merupakan gambar hasil bentuk gelombang dari penyaklaran pada stator motor BLDC :



Gambar 3.4 Bentuk Gelombang Hasil Penyaklaran pada *Stator*

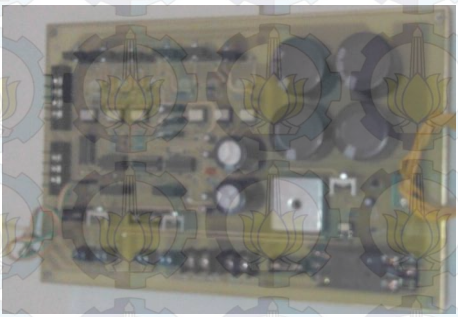
Pada rangkaian *power electronic* terdapat beberapa bagian yang saling melengkapi agar secara keseluruhan rangkaian *power electronic* dapat bekerja dengan baik diantaranya adalah rangkaian *power supply* , pengolah sinyal fasa dan PWM, rangkaian *optocoupler*, dan bagian *switching*.

Berikut merupakan skema dari rangkaian *power electronic* :



Gambar 3.5 Skema Rangkaian *Power Electronic*

Sedangkan bentuk fisik untuk rangkaian *power electronic* terlihat seperti pada gambar berikut :



Gambar 3.6 Bentuk Fisik Rangkaian *Power Electronic*

Rangkaian *power supply* pada gambar diatas menggunakan tegangan sebesar 12 VAC, 18 VAC, dan 64 VAC yang nantinya akan diubah menjadi tegangan DC sebesar 5 VDC, 12 VDC, dan 90 VDC.

3.2.2 Perancangan Mekanik *Plant*

Motor BLDC yang digunakan dalam Tugas Akhir ini adalah motor BLDC ARW31S8P30AM buatan Malaysia dengan spesifikasi seperti pada Gambar 3.7.

Gambar 3.7 Spesifikasi motor BLDC

Berikut ini merupakan gambar dari rancangan konstruksi *plant* motor BLDC :

Gambar 3.8 Kontruksi *Plant* Motor BLDC

Pada gambar dapat dilihat bahwa *shaft* dari motor BLDC diberi piringan alumunium yang akan menjadi beban rem magnetik ketika motor berputar. Beban rem magnetik tersebut digunakan untuk mempengaruhi sistem kerja dari motor BLDC. Pengaruh yang ditimbulkan oleh rem magnetik berupa penurunan kecepatan dari motor BLDC.

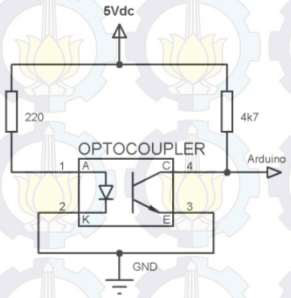
Shaft motor BLDC terhubung dengan *shaft* yang terdapat pada piringan alumunium. Penghubung antar *shaft* menggunakan *coupling*. *Coupling* berfungsi untuk menggabungkan dua *shaft* dan untuk meredam getaran yang ditimbulkan antar *shaft* saat motor berputar.

Rem magnetik pada *plant* terdiri dari dua buah magnet permanen. Magnet diletakkan pada dudukan yang didesain agar bisa diubah posisinya saat motor berputar. Pada ujung *shaft* diletakkan piringan dari sensor *rotary encoder*.

3.2.3 Perancangan Sensor *Rotary Encoder*

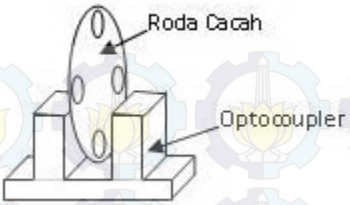
Pada Tugas Akhir ini sensor kecepatan yang digunakan adalah piringan atau roda cacah dan sebuah *optocoupler* tipe U. Piringan *rotary encoder* berputar diantara celah pada *optocoupler*. Optocoupler merupakan komponen *optoisolator* yang memiliki karakteristik penerima (*phototransistor*) akan mengalami perubahan logika 0 ke 1 atau sebaliknya bila terjadi perubahan intensitas cahaya yang dipancarkan oleh pemancar (LED infra merah) untuk penerima.

Phototransistor merupakan jenis transistor yang peka terhadap cahaya infra merah. Piringan akan ditempatkan di tengah dari *optocoupler* tipe U yang berfungsi untuk mempengaruhi intensitas cahaya yang diberikan oleh LED pada *optocoupler* ke *phototransistor* yang akan memberikan perubahan level logika sesuai dengan putaran piringan. Berikut merupakan skema dari rangkaian sensor *rotary encoder* :



Gambar 3.9 Skema Rangkaian Sensor *Rotary Encoder*

Konstruksi sensor *rotary encoder* dapat dilihat pada Gambar 3.10. Dimana pada konstruksi sensor tersebut terdiri dari roda cacah dan *optocoupler* dengan cara kerja telah dijelaskan pada bab sebelumnya.



Gambar 3.10 Konstruksi Sensor *Rotary Encoder* dengan *Optocoupler*

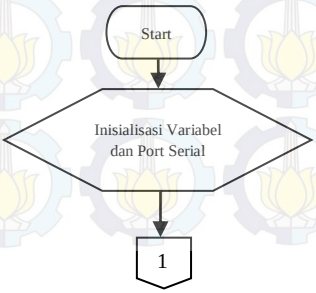
3.3 Perancangan Perangkat Lunak (*Software)*

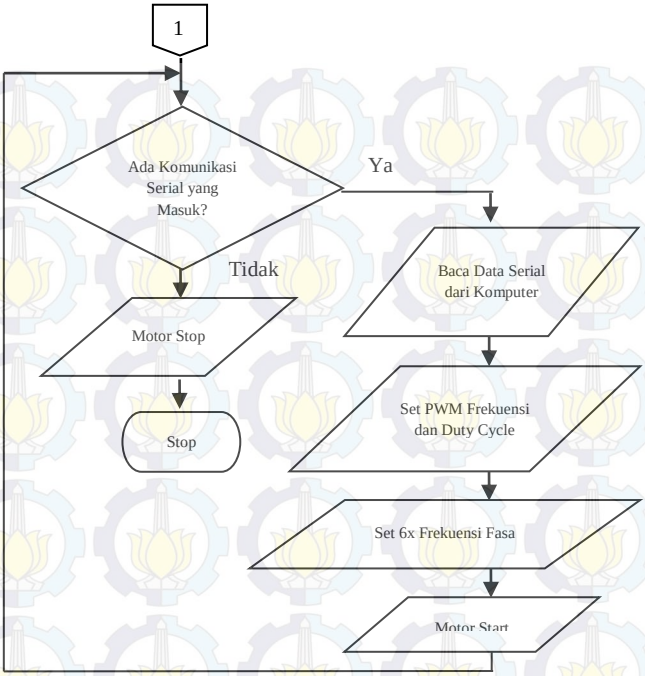
Pada Tugas Akhir ini terdapat perangkat lunak (*software*) yang digunakan dalam perancangan sistem ini diantaranya adalah arduino, dimana arduino ini berfungsi untuk membuat program pembangkit frekuensi dan program membaca kecepatan motor BLDC.

3.3.1 Pemrograman Pembangkit Frekuensi dan PWM

Untuk program pembangkit frekuensi dan PWM ini digunakan arduino UNO dimana perintah untuk membangkitkan frekuensi dilakukan oleh PC melalui *software* MATLAB.

Frekuensi yang dibangkitkan oleh arduino merupakan 6x frekuensi fasa. Hal ini diperlukan karena spesifikasi *hardware* pada *driver inverter* menggunakan rangkaian pembagi fasa. PWM berfungsi untuk memperhalus putaran motor. Syarat frekuensi PWM minimal 4 Khz, hal ini untuk mencegah terjadinya pembacaan PWM di dalam frekuensi fasa, apabila PWM ini masih terbaca oleh rangkaian *switching* maka akan mengakibatkan motor tidak akan berputar. Duty cycle frekuensi fasa harus sebesar 50%, sedangkan duty cycle untuk PWM minimal sebesar 60%. Berikut merupakan *flowchart* dari program pembangkit frekuensi dan PWM :

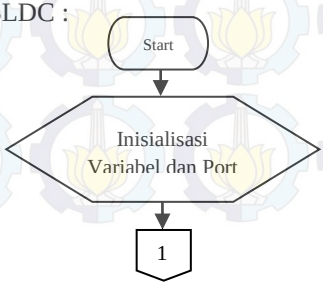


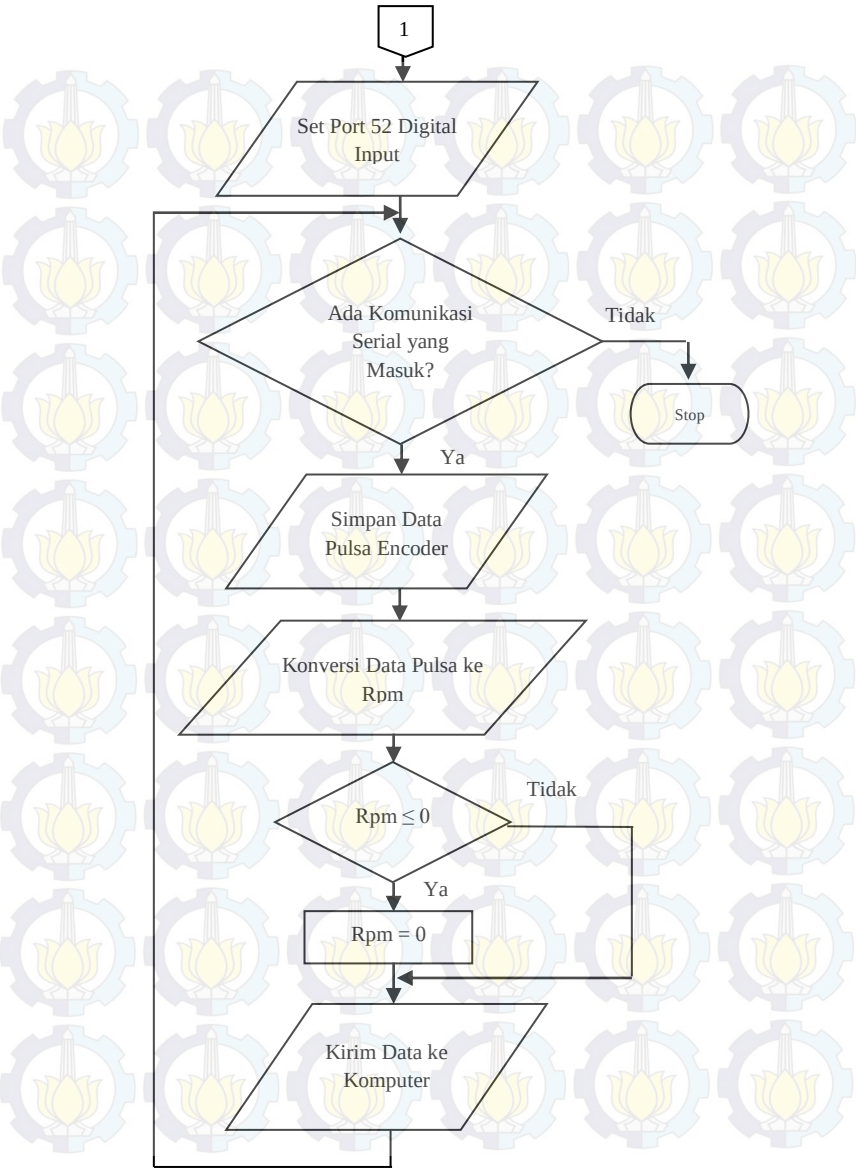


Gambar 3.11 *Flowchart* Program Pembangkit PWM dan Frekuensi

3.3.2 Pemrograman Membaca Kecepatan Motor BLDC

Untuk program pembangkit frekuensi dan PWM ini digunakan arduino Mega 2560 dimana proses pembacaan data kecepatan motor BLDC dengan satuan rpm akan dikirim ke PC melalui *software* MATLAB. Berikut merupakan *flowchart* dari program membaca kecepatan motor BLDC :

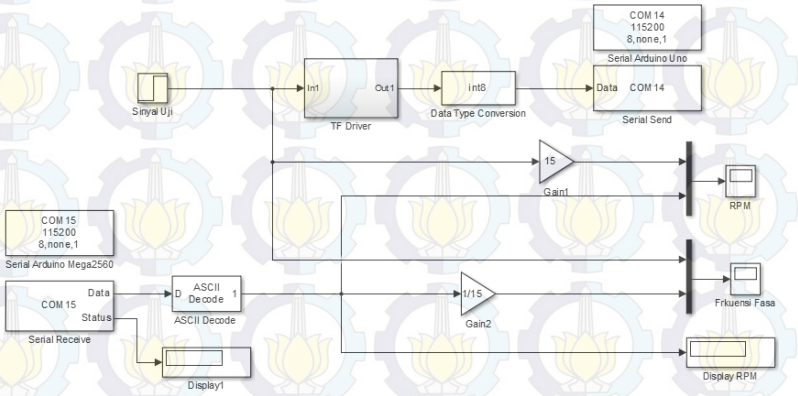




Gambar 3.12 *Flowchart* Program Membaca Kecepatan Motor BLDC

3.3.3 Pemrograman Metode Identifikasi

Perancangan metode identifikasi menggunakan *software* MATLAB R2014a, perancangan ini bertujuan untuk mengidentifikasi *plant* sebelum dilakukan implementasi pengaturan kecepatan pada *plant*. Koneksi antara MATLAB dengan arduino menggunakan blok diagram serial sebagai pengirim dan penerima data.



Gambar 3.13 Blok Diagram Simulink Perancangan Metode Identifikasi

3.4 Identifikasi Sistem

Identifikasi sistem diperlukan untuk mendapatkan model matematika dari motor BLDC. Pada Tugas Akhir ini, dilakukan identifikasi statis dengan melihat keluaran respon kecepatan motor terhadap referensi yang diberikan. Sinyal uji *step* diberikan melalui arduino dengan memberi nilai sinyal *input* sebesar 45 Hz atau 675 rpm. Hasil respon *plant* akan ditampilkan di simulink melalui blok diagram *Serial Receive*. Data kecepatan disimpan dalam bentuk *workspace* dengan *time sampling* sebesar 0.001 detik.

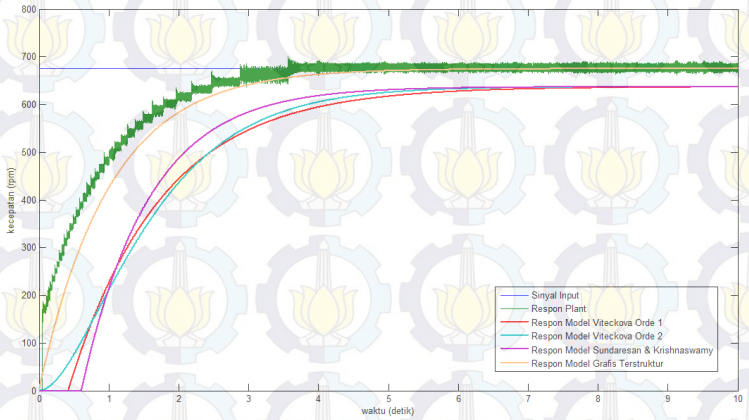
3.4.1 Metode Identifikasi Sistem

Pada Tugas Akhir ini digunakan empat metode identifikasi sistem yaitu metode Viteckova 1st *Order*, metode Viteckova 2nd *Order*, metode Sundaresan – Krishnaswamy dan metode Grafis Terstruktur. Dari keempat metode tersebut akan dicari RMSE yang paling kecil dan mendekati dengan model plant motor BLDC. Berikut merupakan table hasil dari identifikasi sistem dengan beberapa metode identifikasi :

Tabel 3.2 Validasi Model Matematika motor BLDC

No.	Metode	Model Matematika	RMSE
1.	Viteckova 1st *Order*	$\frac{0.9436}{1.3242s+1}\cdot e^{-0.4189s}$	128.221
2.	Viteckova 2nd *Order*	$\frac{0.9436}{0.713s^2+1.689s+1}$	125.978
3.	Sundaresan & Krishnaswamy	$\frac{0.9436}{0.963s+1}\cdot e^{-0.6081s}$	123.895
4.	Grafis Terstruktur	$\frac{1.003(s+1)}{1.007s^2+2.007s+1}$	33.692

Dari keempat hasil identifikasi tersebut dipilih model matematika dengan nilai RMSE paling kecil. Berikut merupakan hasil simulasi dari hasil identifikasi sistem :

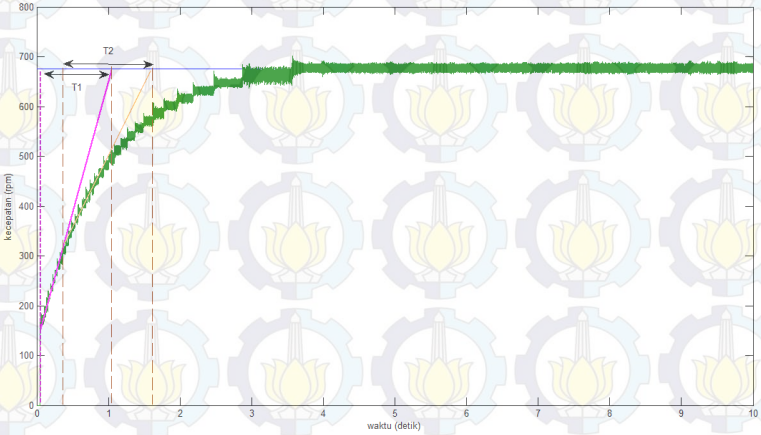


Gambar 3.14 Grafik Hasil Identifikasi

Metode Grafis Terstruktur dipilih karena memiliki validasi nilai RMSE yang terkecil. Hasil respon dari pendekatan Grafis Terstruktur berupa respon orde 2 tanpa *delay*. Dari hasil grafik respon pada MATLAB didapatkan bahwa pendekatan model matematika dengan metode Grafis Terstruktur yang paling mendekati respon *plant* yang sebenarnya.

Perhitungan dalam menentukan pendekatan model matematika dengan metode Grafis Terstruktur adalah sebagai berikut:

1. Plot hasil respon plant
2. Tarik garis singgung yang memotong kurva respon plant, kemudian hitung τ_1 dan τ_2, seperti gambar 3.14



Gambar 3.15 Penarikan Garis Singgung pada Metode Grafis

3. Penurunan rumus model matematika seperti di bawah ini :

$$G(s) = K\left(\frac{1/(1.007)}{s+1.007} + \frac{1/(1)}{s+1}\right) \quad (3.1)$$

Mula-mulai gain K dianggap bernilai 1

4. Menentukan nilai gain (K)
Gain (K) bisa didapat apabila setelah diberi sinyal uji *unit step* sesuai dengan *set point dan* melihat hasil respon melalui *scoope*. Setelah melihat hasil respon, maka nilai dari K didapat melalui rumus :

$$K = Yss\left(\frac{Set\ Point}{Yss}\right) = 1.003\left(\frac{675}{1.003}\right) = 7.522 \quad (3.2)$$

5. Fungsi alih keseluruhan

$$G(s) = 7.522\left(\frac{2.007s + 2}{1.007s^2 + 2.007s + 1}\right)$$

$$= 1.003\left(\frac{s+1}{1.007s^2+2.007s+1}\right) \quad (3.3)$$

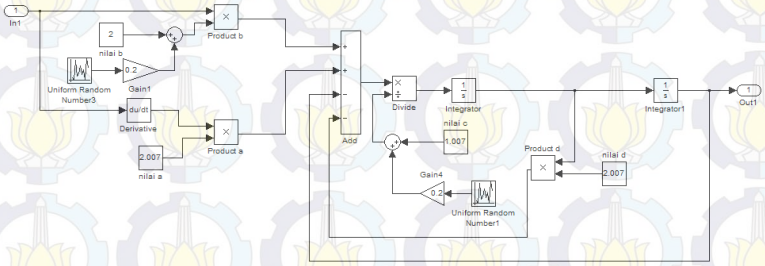
3.4.2 Metode Simulasi Pembebanan Sistem

Untuk simulasi pembebanan sistem menggunakan *plant* parameter bervariasi. Dimana ketika melakukan identifikasi sistem akan mendapatkan model matematika dengan bentuk seperti berikut :

$$G(s) = \frac{as+b}{cs^2+ds+1} \quad (3.4)$$

Dengan didapatkannya model matematika tersebut, maka *plant* parameter bervariasi dibuat dengan mengubah parameter b dan c (asumsi nilai b= ±0.2 dan c = ±0.20) sehingga didapatkan tiga kemungkinan nilai dari masing – masing parameter. Maka secara keseluruhan akan terdapat sembilan kemungkinan dari *plant* parameter bervariasi, dimana hal ini nantinya akan disimulasikan seolah-olah sistem telah mengalami perubahan beban.

Dibuatnya *plant* parameter bervariasi ini bertujuan untuk mengetahui respon kontroler jika *plant* mengalami perubahan parameter. Berikut merupakan blok diagram dari *plant* parameter bervariasi yang disimulasikan pada *software* MATLAB :



Gambar 3.16 Blok Diagram dari Sub Sistem *Plant* Parameter Bervariasi

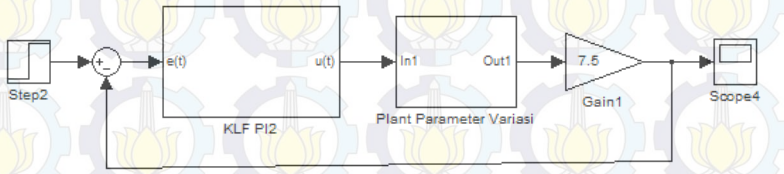
Blok diagram tersebut merupakan penurunan dari persamaan 3.4 seperti berikut ini :

$$\frac{Y(s)}{U(s)} = \frac{as+b}{cs^2+ds+1}$$

$$cs^2Y(s) + dsY(s) + Y(s) = asU(s) + bU(s)$$

$$c\ddot{y} + d\dot{y} + y = a\dot{u} + bu$$

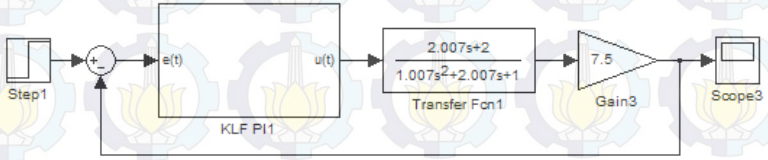
$$\ddot{y} = \frac{1}{c}(a\dot{u} + bu - d\dot{y} - y) \quad (3.5)$$



Gambar 3.17 Blok Diagram dari *Plant* Parameter Bervariasi

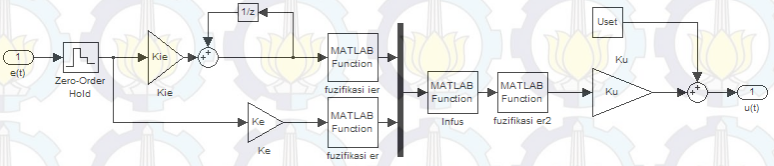
3.5 Perancangan Kontroler *Fuzzy*-PI

Pada Tugas Akhir ini digunakan kontroler *fuzzy*-PI untuk mengatur kecepatan motor BLDC. Kontroler *fuzzy*-PI yang digunakan adalah kontroler *fuzzy*-PI dengan *rule base* PI. Berikut merupakan blok diagram kontroler *fuzzy*-PI :



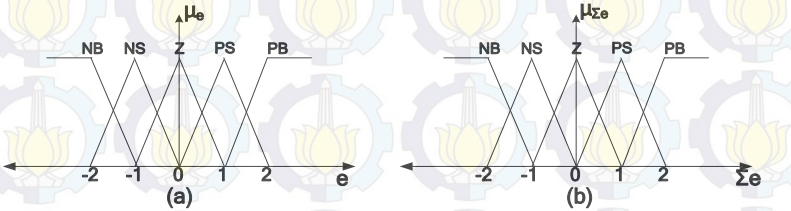
Gambar 3.18 Blok Diagram Kontroler *Fuzzy*-PI

Di dalam blok kontroler *fuzzy*-PI terdapat sub sistem dari kontroler *fuzzy*-PI seperti yang terlihat pada gambar berikut :



Gambar 3.19 Blok Diagram Sub Sistem Kontroler *Fuzzy*-PI

Untuk fungsi keanggotan yang digunakan pada proses *fuzzifikasi* dari *error* dan integral *error* yaitu fungsi segitiga dengan 5 himpunan pendukung untuk *error* dan integral *error*. Gambar 3.20 menunjukkan fungsi keanggotaan dari *error* dan integral *error*.



Gambar 3.20 Fungsi Keanggotan : (a) *error ;* (b) integral *error*

Karena kontroler yang digunakan *fuzzy*-PI adalah dengan *rule base* PI maka tabel *rule base* terlihat pada tabel berikut ini :

Tabel 3.3 Tabel Basis Aturan Mack Vicar Wheelan

	$u_{\sum e}1$	$u_{\sum e}2$	$u_{\sum e}3$	$u_{\sum e}4$	$u_{\sum e}5$
ue1	1	1	2	2	3
ue2	1	2	2	3	4
ue3	2	2	3	4	4
ue4	2	3	4	4	5
ue5	3	4	4	5	5

BAB 4
HASIL SIMULASI DAN PENGUJIAN SISTEM

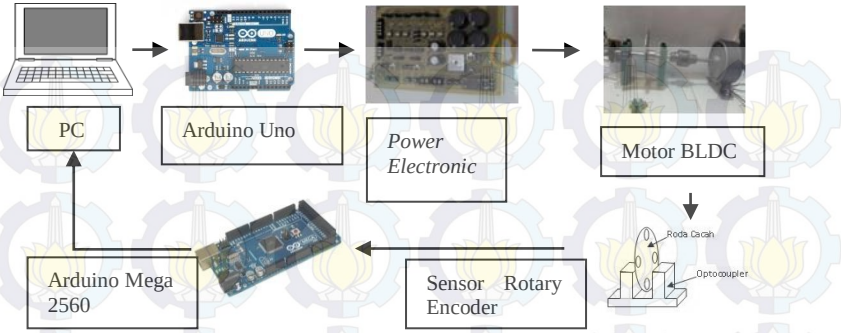
Pada bab ini menjelaskan tentang hasil simulasi dan pengujian dari perancangan sistem yang telah dilakukan dan dijelaskan pada bab sebelumnya. Bab ini terdiri dari pengujian kontroler yang telah diimplementasikan dengan menggunakan *software* MATLAB. Pengujian ini dilakukan untuk mengetahui respon keluaran dari sistem dan juga untuk mengetahui proses kerja dari kontroler yang didesain.

4.1 Gambaran Singkat Pengujian Sistem

Pengujian sistem *hardware* dilakukan terhadap respon kecepatan motor BLDC serta pengujian terhadap arduino sebagai pembangkit frekuensi fasa. Pengujian sistem *software* berupa hasil pengujian respon kontroler *fuzzy*-PI terhadap model *plant* motor BLDC dengan beban rem magnetic secara bervariasi. Kecepatan referensi motor BLDC bernilai konstan kemudian dilakukan pengujian terhadap kontroler *fuzzy*-PI dengan tujuan untuk mengetahui bahwa kontroler dapat mengatasi perubahan respon pada model *plant* motor BLDC karena perubahan parameter pada kondisi beban minimal, nominal dan maksimal.

4.2 Pengujian Kecepatan Motor BLDC

Pengujian ini dilakukan untuk mengetahui dan mengkalibrasi kecepatan hasil dari pembacaan sensor *rotary encoder* motor BLDC dengan kecepatan hasil pembacaan tachometer. Pengujian dilakukan dengan menghubungkan motor BLDC dengan sumber tiga fasa dari rangkaian *power electronic*. Piringan sensor *rotary encoder* dipasang pada *shaft* motor yang nantinya piringan akan dibaca oleh *optocoupler* tipe U. Pulsa hasil pembacaan *rotary encoder* dikirim ke arduino Mega 2560. Data yang didapat kemudian diolah dan hasilnya ditampilkan pada *display* yang terdapat pada PC. Tachometer digunakan untuk melihat perbandingan kecepatan dari piringan dengan hasil pengolah arduino. Mekanisme pengujian kecepatan BLDCM dapat dilihat pada Gambar 4.1 berikut :



Gambar 4.1 Mekanisme Pengujian Kecepatan Motor BLDC

Hasil pengujian untuk kecepatan motor BLDC dapat dilihat pada Tabel 4.1.

Tabel 4.1 Hasil Pengujian Kecepatan Motor BLDC

Frekuensi Fasa (Hz)	Kecepatan (Rpm)			
	Tachometer	Pengukuran 1	Pengukuran 2	Pengukuran 3
10	155,5	155	157	158
15	218,5	217	217	221
20	295	294	296	298
25	372,5	370	371,5	374
30	447	447	451	442
35	524,5	523	525	527
40	600	598	599	603
45	673	671,4	678	670

Tabel 4.1 Hasil Pengujian Kecepatan Motor BLDC (Lanjutan)

Frekuensi Fasa (Hz)	Kecepatan (Rpm)			
	Tachometer	Pengukuran 1	Pengukuran 2	Pengukuran 3
50	748.5	745	749	750
55	826	820	826	835

Untuk mencari presentase *error* rata-rata digunakan rumus sebagai berikut :

$$error = \frac{(pembacaan\ sensor - Pembacaan\ Tachometer)}{Pembacaan\ Tachometer}\ x\ 100\% \quad (4.1)$$

Sehingga didapatkan *error* rata-rata dari semua pembacaan sebesar 0.109%. Berdasarkan data yang didapatkan pada tabel, hasil pembacaan kecepatan motor dengan *tachometer* dan perhitungan memiliki selisih yang kecil. Hal tersebut menunjukkan bahwa pengujian untuk motor BLDC telah berhasil dan sesuai.

4.3 Pengujian Kontroler Arduino

Pengujian arduino menggunakan *software* Matlab R2014a. Sinyal uji berupa frekuensi fasa dengan rentang 10-55 Hz. Arduino akan membangkitkan 6x frekuensi fasa sesuai dengan nilai rentang yang diberikan oleh PC. Pengukuran frekuensi dari arduino diukur menggunakan osiloskop digital.

Tabel 4.2 Hasil Pengujian Pembangkit Frekuensi Fasa Arduino

Frekuensi Fasa (Hz)	Perhitungan 6x Frekuensi Fasa (Hz)	Pembangkit Frekuensi Arduino (Hz)	Error (%)
10	60	56	-6.66667
20	120	119	-0.83333

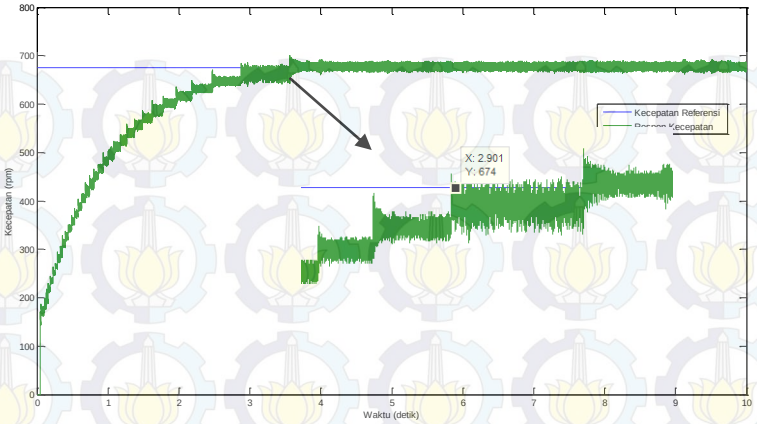
Tabel 4.2 Hasil Pengujian Pembangkit Frekuensi Fasa Arduino (Lanjutan)

Frekuensi Fasa (Hz)	Perhitungan 6x Frekuensi Fasa (Hz)	Pembangkit Frekuensi Arduino (Hz)	Error (%)
30	180	181	0.555556
40	240	241	0.416667
43	260	262	0.769231
50	300	301	0.333333
55	330	333	0.909091

Berdasarkan data yang didapatkan pada tabel, hasil pengujian pembangkit frekuensi fasa arduino dan perhitungan memiliki selisih yang kecil. Hal tersebut menunjukkan bahwa pengujian untuk kontroler arduino telah berhasil dan sesuai.

4.4 Simulasi Motor BLDC

Simulasi ini dilakukan untuk melihat respon kecepatan dari motor BLDC dengan kecepatan referensi konstan seperti pada Gambar 4.1. Dari grafik hasil respon kecepatan ini dapat diperoleh model matematik dari motor BLDC melalui identifikasi sistem yang telah dijelaskan pada bab sebelumnya.

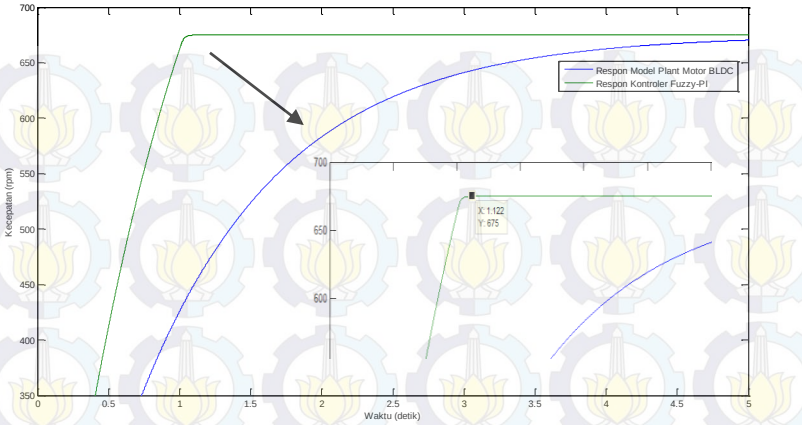


Gambar 4.2 Respon Kecepatan Motor BLDC

Dari hasil grafik diatas, dapat diketahui bahwa pada saat t = 2.901 detik respon kecepatan *plant* model baru mencapai nilai yang kurang lebih sama dengan kecepatan referensi (waktu *steady state*).

4.5 Simulasi dan Pengujian dengan Kontroler *Fuzzy*-PI

Pada subbab ini akan menunjukkan hasil simulasi model *plant* motor BLDC dengan kontroler *fuzzy*-PI pada *plant* motor BLDC yang telah dirancang. Nilai parameter kontroler *fuzzy*-PI diperoleh dari hasil *tunning* dengan nilai *gain integral error* (Kie) = 0.001, *gain error* (Ke) = 0.1, *control offset* (Uset) = 20 dan *gain control* (Ku) = 25.

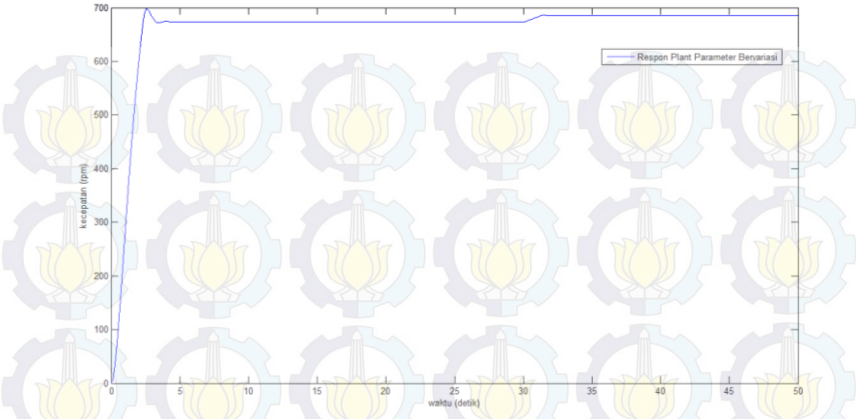


Gambar 4.3 Respon Kontroler *Fuzzy*-PI

Dengan spesifikasi respon sebagai berikut $\tau = 0.707$ detik, $t_s(\pm 5\%) = 2.121\ detik,\quad t_r\ (5\% - 95\%) = 2.081\ detik,\quad t_d = 0.49\ detik$ dan e_{ss} = - 0.03%. Dalam simulasi tersebut, hasil yang didapatkan adalah kontroler dapat bekerja dengan baik mengikuti nilai referensi yang telah diberikan, pada saat t = 1.122 detik.

4.6 Simulasi *Plant* Parameter Bervariasi

Tahap ini dilakukan untuk mengetahui dan menguji proses kerja dari kontroler yang telah didesain. Pengujian dilakukan dengan Simulink yang terdapat dalam *software* Matlab. Pada Tugas Akhir ini dilakukan pengujian untuk kontroler *fuzzy*-PI. Metode pembebanan diberikan dalam selang waktu 50 detik. Hal ini untuk mempermudah melihat respon kontroler dengan baik.



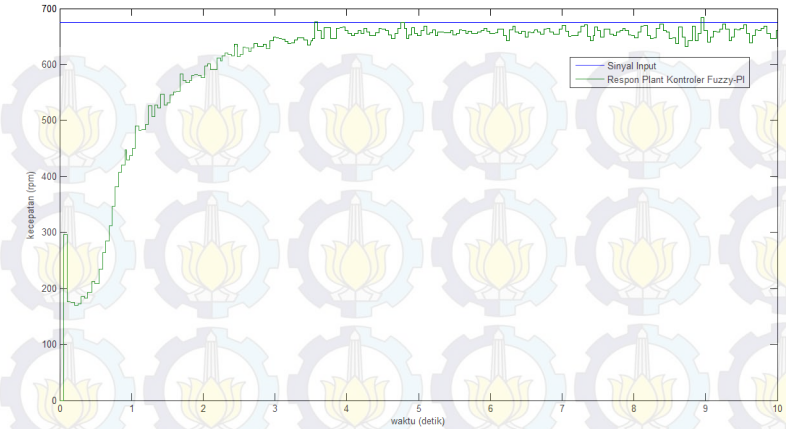
Gambar 4.4 Respon *Plant* Parameter Bervariasi

Saat simulasi untuk model *plant* parameter bervariasi, kontroler *fuzzy*-PI mengalami *overshoot* tetapi dapat kembali sesuai dengan nilai referensi yang telah diberikan sebelumnya.

4.7 Implementasi dan Pengujian dengan Kontroler *Fuzzy*-PI

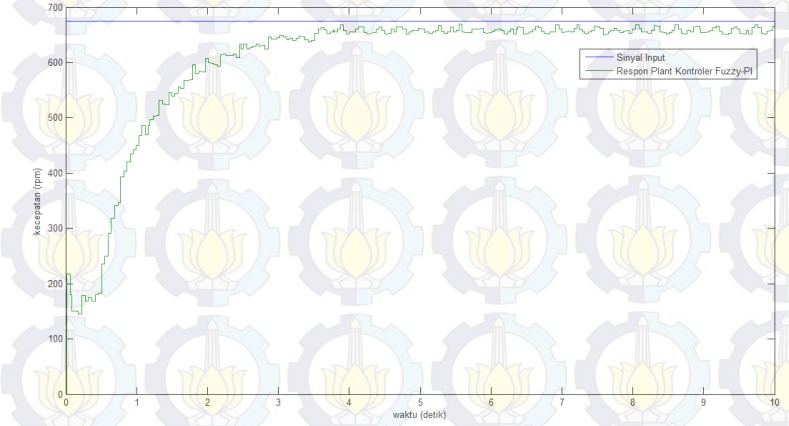
Pada implementasi kontroler *fuzzy*-PI menggunakan beban rem magnetik. Untuk kondisi beban minimal, piringan rotor tidak diberi beban rem magnetik. Untuk kondisi beban nominal, piringan rotor diberi beban rem magnetik sebesar ½ luas rem magnetik. Untuk kondisi beban maksimal, piringan rotor diberi beban magnetik sepenuhnya dari luas penampang rem magnetik.

Nilai parameter kontroler *fuzzy*-PI diperoleh dari hasil *tunning* dengan nilai *gain integral error* (Kie) = 0.001, *gain error* (Ke) = 0.1, *control offset* (Uset) = 12 dan *gain control* (Ku) = 25.



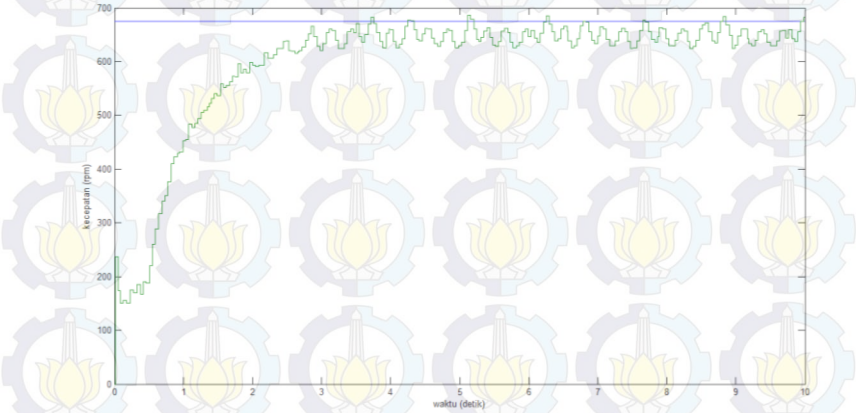
Gambar 4.5 Respon Kontroler *Fuzzy*-PI dengan Beban Minimal

Dari hasil respon tersebut terlihat bahwa kecepatan motor BLDC sedikit menurun dari kecepatan referensi tetapi tetap dapat mengikuti kecepatan referensi. Spesifikasi respon tersebut diantaranya $\tau = 2.248$ detik, $t_s(\pm 5\%) = 6.744\ detik$, $t_r\ (5\% - 95\%) = 6.619\ detik$, $t_d = 1.558\ detik$ dan $e_{ss} = 0.001\%$.



Gambar 4.6 Respon Kontroler *Fuzzy*-PI dengan Beban Nominal

Dari hasil respon tersebut terlihat bahwa kecepatan motor BLDC tetap dapat mengikuti kecepatan referensi. Spesifikasi respon tersebut diantaranya τ = 2.455 detik, $t_s(\pm 5\%) = 7.367\ detik$, $t_r\ (5\% - 95\%) = 7.228\ detik$, $t_d = 1.701\ detik$ dan e_{ss} = -0.08%.



Gambar 4.7 Respon Kontroler *Fuzzy*-PI dengan Beban Maksimal

Dari hasil respon tersebut terlihat bahwa kecepatan motor BLDC tetap dapat mengikuti kecepatan referensi meskipun kecepatan motor BLDC menurun. Spesifikasi respon tersebut diantaranya τ = 2.226 detik, $t_s(\pm 5\%) = 6.679\ detik$, $t_r\ (5\% - 95\%) = 6.554\ detik$, $t_d = 1.542\ detik$ dan e_{ss} = -0.09%.

Halaman ini sengaja dikosongkan

DAFTAR PUSTAKA

[1] N. Parhizkar, M. Shafiei, and M. Bahrami Kouhshahi " *Direct Torque Control of Brushless DC Motor Drives with Reduced Starting Current Using Fuzzy Logic Controller*", *IEEE Uncertainty Reasoning and Knowledge Engineering (URKE), 2011 International Conference on*, Vol. 1, No.2 August 2011.

[2] M. V. Ramesh, J. Amarnath, and S. Kamakshaiah "*Speed Control of Brushless DC Motor by Using Fuzzy Logic Pi Controller", ARPN Journal of Engineering and Applied Sciences*, Vol.6, No.9, September 2011.

[3] Hidayat, Alfin. *Cascade Fuzzy Sliding Mode Control-PID untuk Pengaturan Posisi Pada Brushless DC Motor*. Thesis Elektro – ITS. 2012.

[4] Muhammad H. Rashid, Ph.D., *"POWER ELECTRONICS HANDBOOK DEVICES, CIRCUITS, AND APPLICATIONS Third Edition"*, University of West Florida, U.S.A, Ch. 34, 2011.

[5] Loe, Yohan. *Kontroler Motor BLDC Menggunakan Microchip*. Skripsi Sistem Komputer – Universitas Binus. 2014.

[6] Candra Wardianto, Ovi. *Kontrol Fuzzy Adaptif Gain Scheduling Untuk Pengaturan Motor Induksi 3 Fasa*. Tugas Akhir Elektro-ITS. 2013.

[7] Putri Suryaditya, Nindita. *Pengaturan Proses Face Miling pada Mesin Computer Numerikal Control (CNC) dengan Kontroler Fuzzy-PID*. Tugas Akhir Elektro-ITS. 2013.

[8] Fitria Fauzy, Rizky. *Desain Kontroler Pid Fuzzy Untuk Pengendalian Tekanan dan Level Oksigen Gas Buang Pada Boiler*. Tugas Akhir Elektro-ITS. 2012.

Halaman ini sengaja dikosongkan

BAB 5
PENUTUP

5.1 Kesimpulan

Dari hasil desain diperoleh bahwa dengan menggunakan kontroler *fuzzy*-PI pada pengaturan kecepatan motor BLDC, dapat diambil kesimpulan sebagai berikut :

a. Pada kondisi beban minimal, implementasi kontroler *fuzzy*-PI mampu mendekati nilai kecepatan referensi dengan nilai τ = 2.248 detik, $t_s(\pm 5\%) = 6.744\ detik$, $t_r\ (5\% - 95\%) = 6.619\ detik$, $t_d = 1.558\ detik$ dan e_{ss} = 0.001%.

b. Pada kondisi beban nominal, implementasi kontroler fuzzy-PI mampu mendekati nilai kecepatan referensi dengan nilai τ = 2.455 detik, $t_s(\pm 5\%) = 7.367\ detik$, $t_r\ (5\% - 95\%) = 7.228\ detik$, $t_d = 1.701\ detik$ dan e_{ss} = -0.08%.

c. Pada kondisi beban maksimal, implementasi kontroler *fuzzy*-PI mampu mendekati nilai kecepatan referensi dan hasil respon mendekati respon beban nominal dengan nilai τ = 2.226 detik, $t_s(\pm 5\%) = 6.679 detik$, $t_r\ (5\% - 95\%) = 6.554\ detik$, $t_d = 1.542\ detik$ dan e_{ss} = -0.09%.

5.2 Saran

Untuk pengembangan penelitian, penulis menyarankan untuk menggunakan metode kontroler lain yang dapat memberi respon lebih baik sehingga dapat mencapai nilai kecepatan referensi dengan lebih akurat pada respon keluaran motor BLDC.

Halaman ini sengaja dikosongkan

LAMPIRAN A

A.1 Program *Fuzzifikasi* untuk Kontroler *Fuzzy*-PI

```
function xf=fusi(x)
xf=[0 0 0 0 0]';
if x<-2
    xf(1)=1;
elseif x<-1
    xf(1)=-1-x;
    xf(2)=x-(-2);
elseif x<0
    xf(2)=0-x;
    xf(3)=x-(-1);
elseif x<1
    xf(3)=1-x;
    xf(4)=x-0;
elseif x<2
    xf(4)=2-x;
    xf(5)=x-1;
else
    xf(5)=1;
end
```

A.2 Program *Fuzzy Inference* untuk Kontroler *Fuzzy*-PI

```
function uf=infus(eder)
erf(1)=eder(1);
erf(2)=eder(2);
erf(3)=eder(3);
erf(4)=eder(4);
erf(5)=eder(5);

derf(1)=eder(6);
derf(2)=eder(7);
derf(3)=eder(8);
derf(4)=eder(9);
derf(5)=eder(10);
```

```
uf=[0  0  0  0  0]';

rbf=[1  1  2  2  3
     1  2  2  3  4
     2  2  3  4  4
     2  3  4  4  5
     3  4  4  5  5];
 for i=1:5
     for j=1:5
         k=rbf(i,j);
         % inference rule mamdani
          uf(k)=max( uf(k), min(erf(j),derf(i))
);
         %Larsent arithmatik rule
         % uf(k)=0.5*( uf(k) +
sqrt(erf(j)*derf(i)) );
     end
 end
```

A.3 Program *Defuzzifikasi* untuk Kontroler *Fuzzy*-PI

```
function x=defusi(xf)
atas=-2*xf(1)-1*xf(2)+0*xf(3)+1*xf(4)+2*xf(5);
bawah=xf(1)+xf(2)+xf(3)+xf(4)+xf(5);
x=atas/bawah;
```

LAMPIRAN B

B.1 Program Arduino Pembangkit Frekuensi Fasa dan PWM

```
int bacadata = 0;
//byte bacadata = 0;
int analogValue = 0;  // variable to hold the analog value
char disp2[4];
int pwm = 0;
//char disp3[1];

void setup() {
  // put your setup code here, to run once:
  TCCR1B = TCCR1B & B11111000 | B00000010;
  //pinMode(22, OUTPUT);
  pinMode(12, OUTPUT);    //pin frekuensi 74175
  Serial.begin(115200);
}

void loop() {
  // put your main code here, to run repeatedly:
  //analogValue = analogRead(1);
  //Serial.print('#');
  //sprintf(disp2, "%4d", analogValue);
  //Serial.print(disp2);
  //digitalWrite(22, LOW);
  //range frekuensi 10-100 Hz dengan skala pembacaan 60-600 Hz
  if (Serial.available() > 0) {
    bacadata = (Serial.read());
    //pembangkit frekuensi fasa
    digitalWrite(12, HIGH);
    delayMicroseconds(79000 / bacadata);
    //delayMicroseconds(74000 / bacadata);  //frekuensi 50% good
    digitalWrite(12, LOW);
    delayMicroseconds(79000 / bacadata);
    analogWrite(10, 204);              //output pin pwm dengan duty cycle
80%
  }

  if (Serial.available() < 1) {
```

```
    analogWrite(10, 0);
  }
}
```

B.2 Program Arduino sebagai Pembaca Sensor Kecepatan

```
int duration = 0;
int rpm = 0;
char bufer[4];
int x;

void setup() {
  // put your setup code here, to run once:
  pinMode(52, INPUT); //encoder pin
  Serial.begin(115200);
  //Serial1.begin(115200);
  //delay(10);
}

void loop() {
  // put your main code here, to run repeatedly:
  //duration = pulseIn(52, HIGH, 100000);
  duration = pulseIn(52, HIGH, 100000);
  rpm = (5500000 / 3) / duration;
  if (rpm < 0) {
    rpm = 0;
  }
  if (rpm > 1000) {
    rpm = 0;
  }
  //delay(100);
  sprintf(bufer, "%4d", rpm);
  Serial.print(bufer);
  //delay(10);
}
```

Halaman ini sengaja dikosongkan

DESIGN ELECTRONIC DRIVE AND FUZZY-PI CONTROLLER FOR SPEED OF MOTOR BRUSHLESS DC

Name **: Marika Ayu Putri Ramadhani**
***Advisor* 1** **: Ir. Rusdhianto Effendi A.K, M.T.**

ABSTRACT

Motor Brushless Direct Current (BLDC) need a tool to move and to control the BLDC motor rotation which is usually called the driving power or BLDC motor driver that BLDC motors can be controlled accurately.

System of fuzzy-PI controller is used to control the speed of a BLDC motors and maintain stable rotation. This controller has a controller parameters that integral gain error, gain error, control offset and gain control. In this method the value of the parameter is determined by tunning in accordance with the required result.

The results of the implementation of the BLDC motor show that the fuzzy-PI controller is able to approach the reference speed value. At the time of minimal load conditions have response specification with value of time constant = 2248 seconds, settling time (± 5%) = 6,744 seconds, rise time (5% -95%) = 6619 seconds, delay time = 1,558 seconds and error steady state = 0.001%.

Keywords : *Driver, Fuzzy-PI Controller, BLDC Motors.*

Halaman ini sengaja dikosongkan

KATA PENGANTAR

Dengan mengucapkan puji syukur kehadirat Allah SWT yang telah melimpahkan rahmat, hidayah serta karunia-Nya sehingga penulis dapat menyelesaikan tugas akhir dengan judul :

"PERANCANGAN PENGGERAK ELEKTRIK DAN KONTROLER *FUZZY*-PI UNTUK PENGATURAN KECEPATAN MOTOR DC *BRUSHLESS*"

Tugas akhir ini merupakan sebagian syarat untuk menyelesaikan mata kuliah dan memperoleh nilai pada tugas akhir.

Dengan selesainya tugas akhir ini penulis menyampaikan terima kasih sebesar-besarnya kepada:

1. Ibu Tri Setyawati selaku orang tua dan kakak kandung penulis, Agung Yuana Putra sekeluarga atas limpahan doa, kasih sayang, dukungan dan dorongan baik berupa moril atau materil bagi penulis.
2. Bapak Ir. Rusdhianto Effendie A K, M.T. selaku dosen pembimbing.
3. Rekan kerja untuk Tugas Akhir, Suwondo Saputra dan Kurniawan Khoiruddin yang telah banyak membantu penulis dalam penyelesaian Tugas Akhir ini.
4. Semua pihak yang telah banyak membantu untuk menyelesaikan tugas akhir ini yang tidak dapat penulis sebutkan satu persatu.

Penulis menyadari masih banyak kekurangan dalam tugas akhir ini. Kritik dan saran untuk perbaikan tugas ini sangat diperlukan. Akhir kata semoga tugas ini dapat bermanfaat bagi kita semua.

Surabaya, Juli 2015

Penulis

Halaman ini sengaja dikosongkan

DAFTAR ISI

Halaman ini sengaja dikosongkan

DAFTAR TABEL

Halaman ini sengaja dikosongkan

DAFTAR GAMBAR

DAFTAR RIWAYAT HIDUP

Marika Ayu Putri Ramadhani , lahir di Surabaya, Jawa Timur pada tanggal 1 Maret 1993. Setelah lulus dari SMA Trimurti Surabaya tahun 2010, penulis melanjutkan studi di Diploma 3 Bidang Studi *Computer Control* jurusan Teknik Elektro Institut Teknologi Sepuluh Nopember (ITS) Surabaya dan lulus tahun 2013. Kemudian melanjutkan kuliah dengan mengambil program Sarjana Lintas Jalur di Institut Teknologi Sepuluh Nopember (ITS) Surabaya dengan mengambil Bidang Studi Teknik Sistem Pengaturan, Jurusan Teknik Elektro. Pada bulan Juni 2015, penulis mengikuti seminar dan ujian tugas akhir di Bidang Studi Teknik Sistem Pengaturan, Jurusan Teknik Elektro, ITS Surabaya sebagai salah satu persyaratan untuk memperoleh gelar Sarjana Teknik Elektro.

Halaman ini sengaja dikosongkan

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Motor DC adalah sebuah motor yang membutuhkan tegangan dc untuk menjalankannya. Secara umum motor DC dibagi atas 2 macam, yaitu : motor DC dengan sikat (*Brushed* DC *Motor*) dan motor DC tanpa sikat (*Brushless* DC *Motor*). Jenis *Brushed* DC motor memerlukan perawatan pada sikatnya serta banyak terjadi rugi tegangan pada sikat. Sehingga pada era sekarang ini motor DC dikembangkan tanpa menggunakan sikat yang dikenal dengan motor BLDC *(Brushless Direct Current Motor)*. Motor ini dipilih karena efisiensi yang tinggi, suaranya halus, ukuran kompak, keandalan yang tinggi dan biaya perawatan yang rendah. Motor BLDC telah mendominasi banyak aplikasi seperti peralatan rumah, peralatan teknologi informasi, industri, transportasi, *aerospace*, peralatan pertahanan, alat listrik, dan peralatan medis laboratorium berbagai bidang. Keuntungan yang motor BLDC berikan kepada setiap aplikasi yang digunakan, terutama pada industri sangat besar. Penggunaan motor ini dapat menghemat biaya dan waktu pada hampir semua industri.[1]

Perubahan motor *Brushed* DC oleh motor BLDC menjadi penyebab diperlukannya cara kontrol yang berbeda untuk komutasi fase arus dari motor BLDC. Rangkaian kontrol pengganti kumutator pada motor BLDC ini disebut dengan *power electronic* untuk mencatu daya ke kumparan stator . Rangkaian kontrol terdiri atas 6 buah MOSFET yang digunakan untuk *switching* tegangan. Pada penerapannya motor BLDC masih terdapat banyak kesalahan antara kecepatan referensi dan kecepatan *feedback.* Berdasarkan hal itu maka diperlukan suatu kontroler untuk memperbaiki kinerja dari motor BLDC.[1]

Kontroler konvensional seperti *Proporsional Integral* (PI) dan kontroler *Proporsional Integral Derivative* (PID) masih banyak digunakan karena memiliki struktur sederhana, tingkat keandalan tinggi dan mudah untuk mencapai kondisi sistem yang dibutuhkan. Namun motor BLDC memiliki multi-variabel, sistem non-linear dan dengan mudah dapat dipengaruhi oleh variasi parameter serta gangguan. Untuk mengatasi masalah ini kontroler *fuzzy*-PI digunakan untuk mengontrol kecepatan dari motor BLDC.[2]

Penerapan teknik kecerdasan buatan seperti *Fuzzy Logic Control* (FLC) telah ditemukan sebagai kontroler yang cocok untuk sebagian besar sistem nonlinear yang kompleks dimana sistem tersebut memiliki pemodelan matematika yang tidak pasti. Kelebihan FLC adalah karena kontroler untuk sistem apapun dapat dikembangkan tanpa persyaratan untuk model matematika dari suatu sistem. Dengan cara ini, efisiensi dan keandalan *drive* akan meningkat. Sehingga pada tugas akhir ini akan dibuat rangkaian penggerak elektrik dan kontroler *fuzzy*-PI untuk motor BLDC.

1.2 Permasalahan

Permasalahan yang dibahas dalam tugas akhir ini adalah perancangan penggerak elektrik dan pengaturan kecepatan motor BLDC dengan kontroler *fuzzy*-PI untuk memperbaiki respon kecepatan dari motor BLDC sehingga kecepatan motor sesuai dengan yang diperlukan serta bagaimana menentukan aturan dasar kontroler *fuzzy* untuk kontrol kecepatan motor BLDC. untuk meningkatkan performansi serta kestabilan sistem.

1.3 Tujuan

Tujuan dari tugas akhir ini adalah untuk merancang penggerak elektrik untuk motor BLDC dan merancang kontroler *fuzzy*-PI untuk mengatur kecepatan motor BLDC sehingga dapat memperbaiki respon kecepatan dari motor BLDC saat terjadi perubahan beban dan sesuai dengan respon model yang diinginkan.

1.4 Metodologi

Metodologi yang dilakukan dalam pengerjaan tugas akhir ini terdiri dari beberapa tahap, yaitu :

a. Studi Literatur

   Studi literatur dilakukan untuk memperoleh informasi mengenai motor BLDC melalui buku teks, jurnal, artikel, internet dan lain-lain.

b. Pemodelan Sistem

   Pada tahap ini dibuat pemodelan motor BLDC berdasarkan pengambilan data dari motor BLDC. Setelah mendapatkan data yang dibutuhkan, akan dilakukan identifikasi dari motor BLDC untuk mendapatkan model matematika yang akan digunakan untuk mendesain kontroler sesuai dengan yang diharapkan.

c. Desain Kontroler

Pada tahap ini dibuat struktur kontroler *fuzzy*-PI untuk pengaturan kecepatan motor BLDC.

d. Simulasi

Pemodelan motor BLDC dan hasil desain kontroler disimulasikan dengan menggunakan perangkat lunak MATLAB.

e. Penyusunan Buku Tugas Akhir

Penyusunan buku Tugas Akhir meliputi pendahuluan, teori penunjang, perancangan sistem, simulasi dan analisa sistem serta penutup.

1.5 Sistematika

Pada Tugas Akhir ini sistematika penulisan dibagi menjadi lima bab, yaitu :

BAB 1 PENDAHULUAN

Bab ini meliputi latar belakang, permasalahan, tujuan, metodologi, sistematika penulisan dan relevansi pembahasan tugas akhir ini.

BAB 2 TEORI PENUNJANG

Bab ini membahas tinjauan pustaka yang membantu penelitian, diantaranya konsep motor BLDC, teori identifikasi sistem, teori penggerak elektronik meliputi *inverter*, sensor *rotary encoder*, dan arduino serta teori kontroler *fuzzy*-PI.

BAB 3 PERANCANGAN SISTEM

Pada bab ini dijelaskan mengenai perancangan model dinamik motor BLDC serta perancangan kontroler *fuzzy*-PI.

BAB 4 PENGUJIAN DAN ANALISA SISTEM

Bab ini memuat hasil simulasi kontroler pada sistem dan analisa sistem.

BAB 5 PENUTUP

Bab ini berisi kesimpulan dan saran dari hasil pembahasan yang telah diperoleh.

1.6 Relevansi

Hasil yang diperoleh dari tugas akhir ini diharapkan dapat memberikan gambaran mengenai perancangan penggerak elektrik untuk motor BLDC serta penerapan kontroler *fuzzy*-PI pada *plant* dengan menggunakan simulasi model *simulink* MATLAB untuk pengaturan kecepatan motor BLDC.

BAB 2
TEORI PENUNJANG

Bab ini membahas mengenai tinjauan pustaka yang berisi landasan teori di dalam pembuatan Tugas Akhir. Beberapa landasan teori yang digunakan antara lain mengenai motor BLDC, teori identifikasi sistem, teori penggerak elektronik meliputi *inverter*, sensor *encoder* serta teori kontroler *Fuzzy*-PI.

2.1 Motor BLDC (*Brushless Direct Current Motor*) [3]

Motor DC banyak digunakan pada aplikasi dunia industri memiliki konstruksi yang menyebabkan timbulnya beberapa kendala pada penggunaannya. Kendala utama dari suatu motor DC adalah pengunaan kumparan rotor dan pemakaian komutator sikat atau sikat arang untuk menghubungkan kumparan rotor dengan unit *servo drive*. Kendala demikian mencakup penggantian atau penyetelan sikat arang, bunga api karena komutasi, keterbatasan arus dan tegangan, inersia rotor yang tinggi, dan satu lintasan panjang disipasi panas yang dibangkitkan rotor. Namun kelemahan pada motor DC dapat dihilangkan dengan penggantian pemakaian motor DC dengan motor BLDC.

Motor BLDC merupakan motor listrik sinkron AC tiga fasa. Motor ini dapat dikendalikan dengan metode *six-step* maupun metode PWM sinusoidal. Dibandingkan dengan motor DC, motor BLDC memiliki biaya perawatan yang lebih rendah dan kecepatan yang lebih tinggi akibat tidak digunakannya sikat. Dibandingkan dengan motor induksi, motor BLDC memiliki efisiensi yang lebih tinggi karena rotor dan torsi awal yang lebih tinggi karena rotor terbuat dari magnet permanen. Walaupun memiliki kelebihan dibandingkan dengan motor DC dan motor induksi, pengendalian motor BLDC jauh lebih rumit untuk kecepatan dan torsi yang konstan karena tidak adanya sikat yang menunjang proses komutasi dan harga motor BLDC jauh lebih mahal.

Secara umum motor BLDC terdiri dari dua bagian, yakni, *rotor*, bagian yang bergerak, yang terbuat dari permanen magnet dan *stator,* bagian yang tidak bergerak, yang terbuat dari kumparan 3 fasa. Walaupun merupakan motor listrik *synchronous* AC 3 fasa, motor ini tetap disebut dengan motor BLDC karena pada implementasinya motor BLDC menggunakan sumber DC sebagai sumber energi utama yang kemudian diubah menjadi tegangan AC dengan menggunakan inverter 3

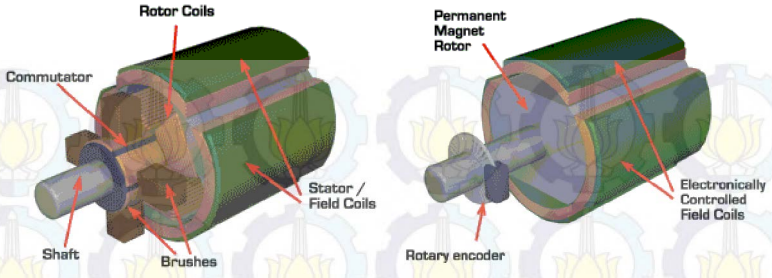
fasa. Tujuan dari pemberian tegangan AC 3 fasa pada stator BLDC adalah menciptakan medan magnet putar stator untuk menarik magnet rotor.

Pada dasarnya, motor BLDC adalah motor DC yang dihilangkan sikat dan komutatornya. Penghilangan komutator dan kumparan rotor akan mengurangi inersia motor dan mampu menaikkan kecepatan rotor. Motor BLDC memiliki rotor berupa suatu magnet permanen, kumparan kawat yang terletak pada stator, dan satu rangkaian elektronik pengganti komutator atau sikat arang. Rangkaian komutasi elektronik mengeliminasi penggantian sikat arang sehingga kumparan dapat dilewati arus dengan tegangan yang lebih tinggi.

Magnet permanen yang digunakan pada rotor memiliki 2 jenis, yaitu :

1. *Rare-earth magnet*
   Magnet ini terbuat dari *samarium cobalt* dan *neodymium iron boron* dan memiliki sifat magnet yang bagus dan banyak tersedia di pasaran, tetapi berharga mahal dan diproduksi secara terbatas.
2. *Ceramic magnets (ferrite)*
   Magnet keramik berharga murah dan dapat digunakan dengan mudah, tetapi magnet ini memiliki sifat magnet yang rendah.

Motor BLDC dengan jenis *rare earth permanen magnet* memiliki inersia rotor yang paling rendah dan ukuran terkecil pada suatu rating torsi tertentu. Stator motor BLDC memiliki kumparan tiga fasa dan menggunakan satu penguat arus PWM untuk mensuplai satu arus sinusoidal tiga fasa pada ketiga kumparan stator. Tipe motor penggerak yang lain menggunakan satu rangkaian pengatur yang lebih sederhana untuk menghasilkan satu arus gelombang persegi tiga fasa, tetapi operasi motor demikian tidak sehalus seperti yang menggunakan penggerak sinusoidal. Gambar 2.1 menunjukkan konstruksi dari motor DC dan motor BLDC.



Gambar 2.1 (a) Konstruksi Motor DC (b) Konstruksi motor BLDC

Gambar 2.1 menunjukkan perbedaan antara motor DC dengan motor BLDC. Pada Gambar 2.1 (a), motor DC menggunakan komutator untuk membalik fasa rotor, sedangkan Gambar 2.1 (b), motor BLDC memiliki magnet permanen pada rotornya dan menggunakan *rotary encoder* untuk membalik fasa rotor secara elektrik. Motor BLDC cenderung lebih efisien jika dibandingkan dengan motor DC karena tidak memiliki sikat dan komutator.

Oleh karena tidak adanya sikat pada motor BLDC, untuk menentukan *timing* komutasi yang tepat pada motor ini sehingga didapatkan torsi dan kecepatan yang konstan, diperlukan 3 buah sensor *Hall* dan atau *encoder.* Pada sensor *Hall, timing* komutasi ditentukan dengan cara mendeteksi medan magnet *rotor* dengan menggunakan 3 buah sensor *hall* untuk mendapatkan 6 kombinasi *timing* yang berbeda, sedangkan pada *encoder, timing* ditentukan dengan cara menghitung jumlah *pole* (kutub) yang ada pada *encoder*.

Pada umumnya *encoder* lebih banyak digunakan pada motor BLDC komersial karena *encoder* cenderung mampu menentukan *timing* komutasi lebih presisi dibandingkan dengan menggunakan sensor *hall.* Hal ini terjadi karena pada *encoder,* kode komutasi telah ditetapkan secara *fixed* berdasarkan banyak *pole* dari motor dan kode inilah yang digunakan untuk menentukan *timing* komutasi. Namun karena kode komutasi *encoder* ditetapkan secara *fixed* berdasarkan banyak *pole* motor, suatu *encoder* untuk suatu motor tidak dapat digunakan untuk motor dengan jumlah *pole* yang berbeda. Hal ini berbeda dengan sensor *hall*. Apabila terjadi perubahan *pole* rotor pada motor, posisi sensor *hall* dapat diubah dengan mudah. Hanya saja kelemahan dari sensor *hall* adalah posisi sensor *hall* tidak tepat akan terjadi kesalahan dalam

penentuan *timing* komutasi atau bahkan tidak didapatkan 6 kombinasi *timing* yang berbeda.

2.1.1 Cara Kerja Motor BLDC

Motor BLDC ini dapat bekerja ketika stator yang terbuat dari kumparan diberikan arus 3 fasa. Akibat arus yang melewati kumparan pada stator timbul medan magnet (B):

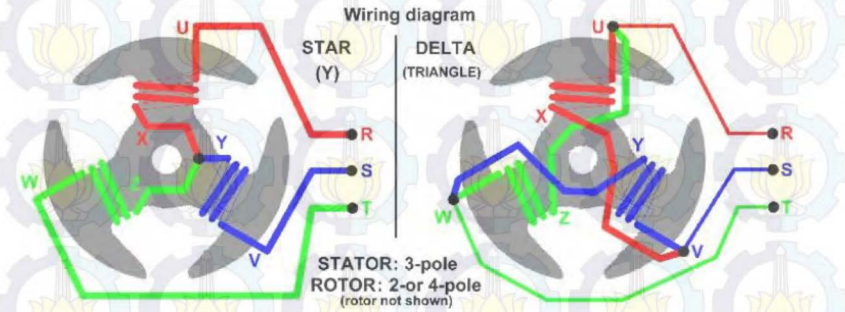
$$B = \frac{\mu N l}{2i} \tag{2.1}$$

Dimana N merupakan jumlah lilitan, i merupakan arus, l merupakan panjang lilitan, dan μ adalah permeabilitas bahan.

Karena arus yang diberikan berupa arus AC fasa, nilai medan magnet dan polarisasi setiap kumparan akan berubah – ubah setiap saat. Akibat yang ditimbulkan dari adanya perubahan polarisasi tersebut dan besar medan magnet tiap kumparan adalah terjadinya medan putar magnet dengan kecepatan N_s :

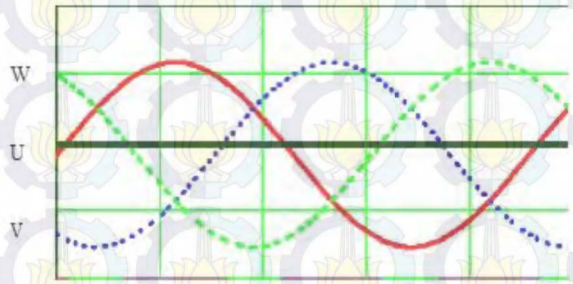
$$N_s = \frac{120f}{p} \tag{2.2}$$

Dimana f merupakan frekuensi tegangan input dinyatakan dalam Hz per satuan detik, p merupakan jumlah kutub (*pole)* pada rotor dan 120° didapat dalam 1 putaran (360°) per 3 fasa motor.



Gambar 2.2 Medan Magnet Putar Stator dan Perputaran Rotor

Berdasarkan gambar 2.2, medan putar magnet stator timbul akibat adanya perubahan polaritas pada stator U, V, dan W. Perubahan polaritas ini terjadi akibat adanya arus yang mengalir pada stator berupa arus AC yang memiliki polaritas yang berubah-ubah.



Gambar 2.3 Tegangan Stator Motor BLDC

Berdasarkan Gambar 2.3, ketika stator U diberikan tegangan negatif maka akan timbul medan magnet dengan polaritas negatif sedangkan V dan W yang diberikan tegangan positif akan memiliki polaritas positif. Akibat adanya perbedaan polaritas antara medan magnet kumparan stator dan magnet rotor, sisi positif magnet rotor akan berputar mendekati medan magnet stator U, sedangkan sisi negatifnya akan berputar mengikuti medan magnet stator V dan W. Akibat tegangan yang digunakan berupa tegangan AC sinusoidal, medan magnet stator U, V, dan W akan berubah – ubah polaritasnya dan besarnya mengikuti perubahan tegangan sinusoidal AC. Ketika U dan V memiliki medan magnet negatif akibat mendapatkan tegangan negatif dan W memiliki medan magnet positif akibat tegangan positif, magnet permanen rotor akan berputar menuju ke polaritas yang bersesuaian yakni bagian negatif akan berputar menuju medan magnet stator W dan sebaliknya bagian positif akan berputar menuju medan magnet stator U dan V. Selanjutnya ketika V memiliki medan magnet negatif dan U serta W memiliki medan magnet positif, bagian positif magnet permanen akan berputar menuju V dan bagian negatif akan menuju U dari kumparan W. Karena tegangan AC sinusoidal yang digunakan berlangsung secara kontinu, proses perubahan polaritas tegangan pada stator ini akan terjadi secara terus menerus sehingga menciptakan medan putar magnet stator dan magnet permanen rotor akan berputar

mengikuti medan putar magnet stator ini. Hal inilah yang menyebabkan rotor pada motor BLDC dapat berputar.

2.2 Prinsip Dasar *Inverter* [4]

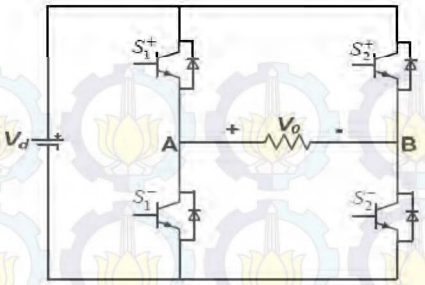
Konverter DC-AC atau biasa disebut *inverter* adalah suatu alat elektronik yang berfungsi untuk menghasilkan keluaran AC sinusoidal dari masukan DC dimana magnitudo dan frekuensinya dapat diatur. *Inverter* biasanya banyak digunakan pada kendali mesin AC dan UPS (*Uninterruptible Power Supplies*).

Dilihat dari jenis masukannya, *inverter* dibagi menjadi dua macam yaitu VSI (*Voltage Source Inverter*) dimana masukannya adalah sumber tegangan DC dan CSI (*Current Source Inverter*) dimana masukannya adalah sumber arus DC. Pada prakteknya, *inverter* yang lebih sering digunakan adalah VSI sedangkan CSI penggunaanya terbatas pada kontrol motor AC dengan daya yang sangat besar.

Salah satu teori dari *inverter* adalah teori *full bridge converter*. *Full bridge converter* adalah rangkaian teori dasar yang digunakan untuk mengubah DC ke AC. *Full bridge converter* mempunyai pasangan saklar (S_1,S_2) dan (S_3,S_4). Keluaran AC didapatkan dari masukan DC dengan membuka dan menutup saklar-saklar pada urutan yang tepat. Tegangan keluaran Vo bisa berupa +Vdc, -Vdc, atau nol, tergantung pada saklar yang mana tertutup.

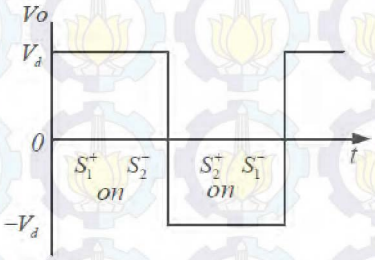
Gambar 2.4 merupakan gambar dari salah satu topologi *inverter bridge*, yaitu *full-bridge* dengan sumber DC yang digunakan adalah sumber tegangan.

Pada dasarnya, untuk menghasilkan keluaran AC sinusoidal, *inverter* bekerja dengan mengatur penyaklaran masukan sumber DC. Dalam satu lengan, transistor yang boleh menyala hanya satu karena apabila dua transistor dalam satu lengan menyala maka sumber tegangan DC akan terhubung singkat. Dengan demikian pada saat S_1^+ menyala maka S_1^- akan mati. Hal yang sama terjadi pada S_2^+ dan S_2^-.



Gambar 2.4 Topologi *Inverter Full-Bridge*

Pada saat S_1^+ dan S_2^- menyala, beban akan merasakan tegangan $V_d\,(V_o = V_d)$. Pada saat S_2^+ dan S_1^- menyala maka beban akan merasakan tegangan $V_d\,(V_o = -V_d)$. Bentuk sinyal tegangan keluaran dari gambar 2.4 adalah sebagai berikut :



Gambar 2.5 Bentuk Tegangan Keluaran *Inverter*

Nilai rms tegangan keluaran dapat dicari dengan rumus :

$$V_o = \left(\frac{2}{T_o}\int_0^{\frac{T_0}{2}} V^2 dt\right)^{\frac{1}{2}} = V_s \qquad (2.3)$$

Keluaran *inverter* dengan penyaklaran seperti diatas adalah gelombang persegi. Gelombang seperti ini memiliki kandungan harmonisa yang besar. Pada umumnya keluaran *inverter* yang diinginkan adalah bentuk gelombang sinus murni karena gelombang sinus murni tidak mengandung harmonisa. Untuk mendapatkan bentuk gelombang sinusoidal maka teknik penyaklaran transistor harus diatur.

Salah satu teknik yang paling umum digunakan dalam mengatur penyaklaran transistor untuk *inverter* adalah PWM (*Pulse Width Modulation*).

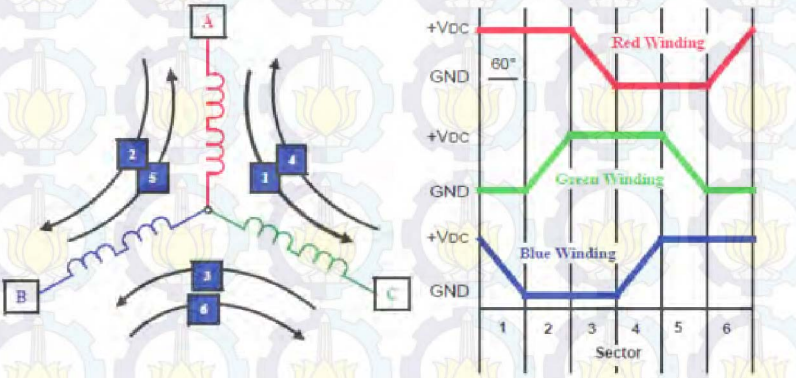
2.3 Metode Pengendalian Motor BLDC[5]

Terdapat beberapa metode yang dapat digunakan dalam pengendalian motor BLDC, diantaranya adalah metode *six-step* dan metode sinyal PWM.

2.3.1 Metode *Six-Step*

Metode Six-Step adalah metode yang paling sering digunakan dalam pengendalian BLDC. Hal ini disebabkan karena metode ini sederhana sehingga mudah diimplementasikan. Hanya saja metode ini memiliki kelemahan yaitu arus RMS (Root Mean Square) yang tinggi. Ini dapat terjadi karena PWM yang digunakan dalam metode ini merupakan PWM square dengan frekuensi tertentu sehingga menciptakan gelombang AC yang bebentuk *trapezoid* atau *square*. Akibat dari gelombang yang beebentuk square atau trapezoid adalah timbulnya gelombang harmonik. Gelombang harmonik inilah yang mengakibatkan motor berputar.

Setiap langkah atau sector adalah ekuivalen dengan 60 derajat elektrikal. 6 sektor menjadi 360 derajat elektrikal atau satu putaran elektrikal.



Gambar 2.6 Komutasi Six-Step

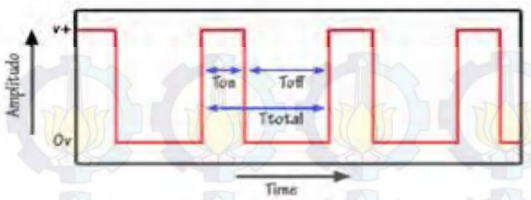
Tanda panah pada kumparan menunjukan arah di mana arus mengalir melalui kumparan-kumparan motor setiap langkah pada Six-Step. Sedangkan untuk urutan langkah komutasi adalah sebagai berikut:

1. Kumparan A diberi tegangan positif, kumparan B tidak diberi tegangan dan kumparan C diberi tegangan negatif.
2. Kumparan A diberi tegangan positif, kumparan B diberi tegangan negatif dan kumparan C tidak diberi tegangan.
3. Kumparan A tidak diberi tegangan, kumparan B diberi tegangan negatif dan kumparan C diberi tegangan positif.
4. Kumparan A diberi tegangan negatif, kumparan B tidak diberi tegangan, dan kumparan C diberi tegangan positif.
5. Kumparan A diberi tegangan negatif, kumparan B diberi tegangan positif, dan kumparan C tidak diberi tegangan.
6. Kumparan A tidak diberi tegangan, kumparan B diberi tegangan positif, dan kumparan C diberi tegangan negatif.

Metode ini disebut *six-step* karena mampu menciptakan gelombang *trapezoidal* atau *square* yang menyerupai gelombang sinusoidal, digunakan PWM *square* yang terdiri dari 6 bagian yaitu 2 bagian positif dan 2 bagian negatif, dan 2 bagian *floating*. Masing-masing bagian besarnya 60 derajat gelombang sinusoidal. Kondisi *floating* pada algoritma ini adalah kondisi ketika gelombang sinusoidal bepotongan pada titik 0.

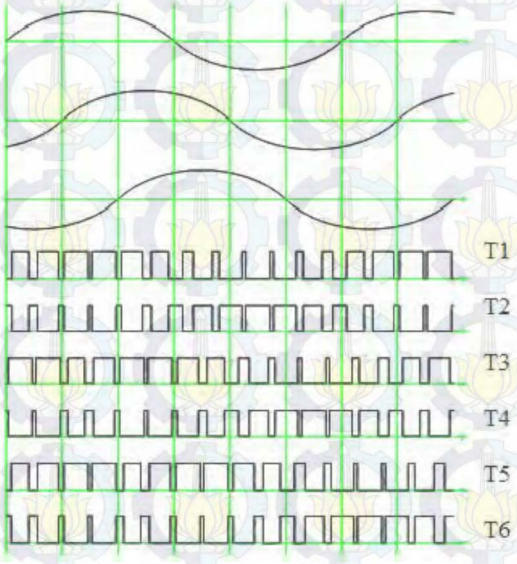
2.3.2 Metode sinyal PWM (*Pulse Width Modulation*)

Pulse width modulation (PWM) secara umum adalah sebuah manipulasi lebar sinyal yang dinyatakan dengan pulsa dalam satu perioda, untuk mendapatkan tegangan rata-rata yang berbeda. Beberapa contoh aplikasi PWM adalah pemodulasian data untuk telekomukasi, pengontrolan daya atau tegangan yang masuk ke beban, regulator tegangan, *audio effect* dan penguatan serta aplikasi lainnya.



Gambar 2.7 Sinyal PWM

Dalam implementasi agar dapat mengendalikan keenam transistor pada driver, sinyal PWM sinusoidal yang didapatkan dibagi menjadi 6 bagian atau step. Masing-masing bagian atau step besarnya 60 derajat. Ini disebabkan karena perbedaan tiap fasa dari sinyal 3 fasa adalah 120 derajat dan tiap 60 derajat terdapat gelombang sinusoidal yang bepotongan dengan nilai 0. Oleh karena itu sinyal PWM harus dibagi menjadi 6 bagian untuk menunjang proses komutasi pada BLDC. Berikut ini implementasi dari algoritma PWM sinussoidal.



Gambar 2.8 Algoritma PWM Sinussoidal

Kecepatan motor BLDC tergantung dari tegangan yang diaplikasikan pada kumparan. Metode PWM digunakan untuk mengendalikan kecepatan motor, sinyal PWM diaplikasikan kesaklar S1 –S6 untuk menetukan rata-rata tegangan pada kumparan.

2.4 Rem Magnetik [6]

Sistem pengereman magnetik menggunakan gaya magnetik untuk memperlambat suatu gerakan, dimana pada umumnya merupakan gerakan poros. Terdapat sebuah piringan berbahan logam non-feromagnetik terpasang dengan poros yang berputar. Piringan tersebut diapit oleh sisi stator berupa sistem lilitan magnetik yang dapat membangkitkan medan magnet dari aliran listrik.

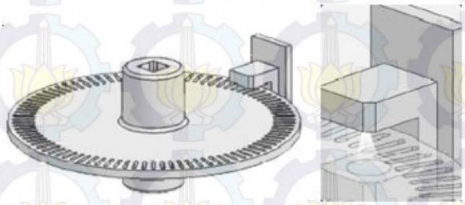
Arus listrik menimbulkan medan magnet pada lilitan dan logam piringan yang memotong medan magnet tersebut akan menimbulkan arus *eddy* pada piringan itu sendiri. Arus *eddy* ini akan menimbulkan medan magnet yang arahnya berlawanan dengan medan magnet sebelumnya, sehingga menghambat gerakan putar dari poros tersebut. Rem magnetik akan bekerja optimal untuk memberikan penurunan kecepatan, tetapi tidak untuk menghentikan gerak suatu objek. Berikut merupakan salah satu contoh dari bentuk fisik dari rem magnetik yang ditunjukkan pada Gambar 2.9.

Gambar 2.9 Bentuk Fisik Rem Magnetik

2.5 Sensor *Rotary Encoder*

Rotary Encoder adalah suatu komponen elektro mekanis yang memiliki fungsi untuk memonitoring posisi *anguler* pada suatu poros yang berputar. Dari perputaran benda tersebut data yang termonitoring

akan diubah ke dalam bentuk data digital oleh *rotary encoder* berupa lebar pulsa kemudian akan dihubungkan ke kontroler (Mikrokontroler/PLC). Berdasarkan data yang didapat berupa posisi *anguler* (sudut) kemudian dapat diolah oleh kontroler sehingga mendapatkan data berupa kecepatan, arah, dan posisi dari perputaran porosnya.



Gambar 2.10 *Optical Shaft Encoder Disk*

Penerapan dari penggunaan *rotary encoder* sering dijumpai pada robot-robot yang membutuhkan kepresisian tinggi dalam hal posisi seperti robot *Mechanum* dan robot *Omni*, selain itu untuk robot berjenis *Differential Drive* (ex : robot tank) juga disarankan menggunakan *rotary encoder* untuk mengatur agar kecepatan putar motor di roda kiri dan kanan bisa sama.

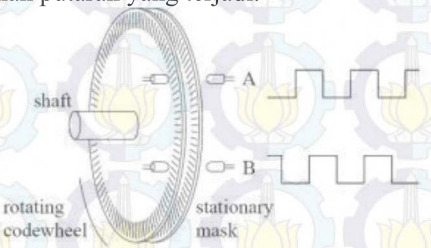
Konstruksi *rotary encoder* berupa piringan tipis yang biasanya di kopel dengan poros yang berputar, umumnya di kopel langsung dengan *shaft* motor. Piringan tipis tersebut terdapat lubang di sepanjang pinggir lingkarannya. Di bagian sisi-sisi piringan terdapat sebuah LED dan *phototransistor* di bagian bersebrangan. Fungsi dari lubang-lubang yang berada di sepanjang pinggir lingkaran tersebut akan menghantarkan cahaya LED ke *phototransistor*, sebaliknya jika cahaya LED tidak menembus lubang piringan maka cahaya akan tertahan. Piringan tersebut akan berputar sesuai dengan kecepatan putaran motor sehingga *phototransistor* akan saturasi ketika cahaya LED menembus lubang-lubangnya.

Pada saat saturasi *phototransistor* akan menghasilkan pulsa dengan range +0.5V s/d +5V. Semakin banyak lubang yang berada pada piringan tentu saja semakin banyak pulsa yang dihasilkan selama satu putaran, hal tersebut berbanding lurus dengan tingkat akurasi yang dihasilkan oleh *rotary encoder*. Ada 2 jenis *rotary* yang umum beredar di pasaran yaitu *incremental rotary encoder* dan *absolute rotary encoder*.

2.5.1 *Incremental Rotary Encoder*

Tipe *incremental rotary encoder* merupakan tipe *rotary encoder* yang paling sederhana karena hanya dapat mengukur perubahan sudut relatifnya saja. Karena kurangnya akurasi dari *incremental rotary encoder* ini perlu ditambahkan satu lagi sensor optik untuk menentukan arah putaran porosnya. Dua buah sensor optik dipasang pada sudut yang berbeda sehingga arah putaran dapat diketahui, biasanya sering disebut *Channel* A dan *Channel* B (Gambar 2.11).

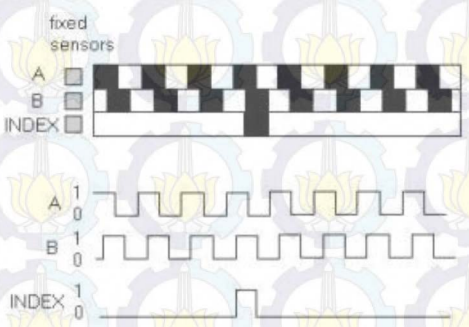
Ketika poros berputar, deretan pulsa akan muncul di masing-masing channel pada frekuensi yang proporsional dengan kecepatan putar sedangkan hubungan fasa antara *channel* A dan B menghasilkan arah putaran. Dengan menghitung jumlah pulsa yang terjadi terhadap resolusi piringan maka putaran dapat diukur. Untuk mengetahui arah putaran, dengan mengetahui *channel* mana yang *leading* terhadap *channel* satunya dapat kita tentukan arah putaran yang terjadi karena kedua *channel* tersebut akan selalu berbeda fasa seperempat putaran (*quadrature signal*). Seringkali terdapat *output channel* ketiga, disebut INDEX, yang menghasilkan satu pulsa per putaran berguna untuk menghitung jumlah putaran yang terjadi.



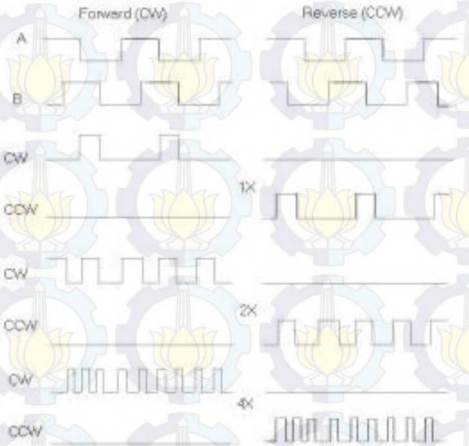
Gambar 2.11 Susunan Piringan untuk *Incremental Encoder*

Contoh pola diagram keluaran dari suatu *incremental encoder* ditunjukkan pada Gambar 2.12. Resolusi keluaran dari sinyal *quadrature* A dan B dapat dibuat beberapa macam, yaitu 1X, 2X dan 4X. Resolusi 1X hanya memberikan pulsa tunggal untuk setiap siklus salah satu sinya A atau B, sedangkan resolusi 4X memberikan pulsa setiap transisi pada kedua sinyal A dan B menjadi empat kali resolusi 1X. Arah putaran dapat ditentukan melalui level salah satu sinyal selama transisi terhadap sinyal yang kedua. Pada contoh resolusi 1X, A = arah bawah dengan B = 1 menunjukkan arah putaran searah jarum

jam, sebaliknya B = arah bawah dengan A = 1 menunjukkan arah berlawanan jarum jam.



Gambar 2.12 Contoh Pola Keluaran *Incremental Encoder*



Gambar 2.13 *Output* dan Arah Putaran pada Resolusi yang Berbeda-Beda

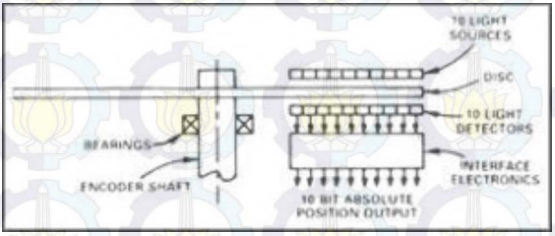
2.5.2 *Absolute Rotary Encoder*

Absolute encoder menggunakan piringan dan sinyal optik yang diatur sedemikian sehingga dapat menghasilkan kode digital untuk menyatakan sejumlah posisi tertentu dari poros yang dihubungkan padanya. Piringan yang digunakan untuk *absolut encoder* tersusun dari segmen-segmen cincin konsentris yang dimulai dari bagian tengah

piringan ke arah tepi luar piringan yang jumlah segmennya selalu dua kali jumlah segmen cincin sebelumnya. Cincin pertama di bagian paling dalam memiliki satu segmen transparan dan satu segmen gelap, cincin kedua memiliki dua segmen transparan dan dua segmen gelap, dan seterusnya hingga cincin terluar. Sebagai contoh apabila *absolut encoder* memiliki 16 cincin konsentris maka cincin terluarnya akan memiliki 32767 segmen. Gambar 2.14 menunjukkan pola cincin pada piringan *absolut encoder* yang memiliki 16 cincin.

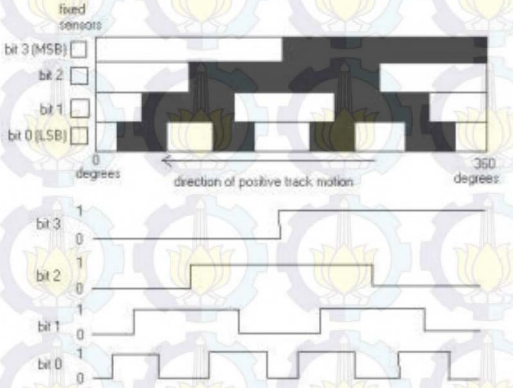
Gambar 2.14 Contoh Susunan Pola 16 Cincin Konsentris pada *Absolut Encoder*

Bilangan yang dihasilkan saat pembacaan nilai terhadap *absolute rotary encoder* ini adalah berupa bilangan biner karena memilki kelipatan dua dari setiap cincinnya. Untuk menghasilkan sistem biner pada susunan cincin maka diperlukan pasangan LED dan *phototransistor* sebanyak jumlah cincin yang ada pada *absolut encoder* tersebut.



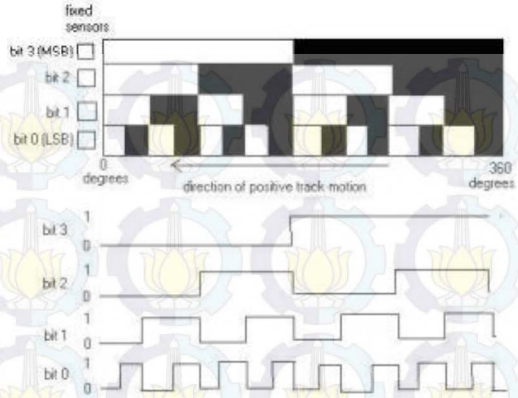
Gambar 2.15 Contoh Susunan Pola 16 Cincin Konsentris pada *Absolut Encoder*

Sistem biner yang untuk menginterpretasi posisi yang diberikan oleh *absolute encoder* dapat menggunakan kode *gray* atau kode biner biasa, tergantung dari pola cincin yang digunakan. Untuk lebih jelas, pada gambar 2.16 contoh *absolut encoder* yang hanya tersusun dari 4 buah cincin untuk membentuk kode 4 bit. Apabila *encoder* ini dihubungkan pada poros, maka *phototransistor* akan mengeluarkan sinyal persegi sesuai dengan susunan cincin yang digunakan. Gambar 5 dan 6 menunjukkan contoh perbedaan diagram keluaran untuk *absolute encoder* tipe *gray code* dan tipe *binary code*.



Gambar 2.16 Contoh Diagram Keluaran *Absolut Encoder* 4-Bit Tipe *Gray Code*

Dengan *absolute encoder* 4-bit ini maka akan didapatkan 16 informasi posisi yang berbeda yang masing-masing dinyatakan dengan kode biner atau kode *gray* tertentu. Tabel 1 menyatakan posisi dan *output* biner yang bersesuaian untuk *absolut encoder* 4-bit. Dengan membaca *output* biner yang dihasilkan maka posisi dari poros yang kita ukur dapat kita ketahui untuk diteruskan ke rangkaian pengendali. Semakin banyak bit yang kita pakai maka posisi yang dapat kita peroleh akan semakin banyak.



Gambar 2.17 Contoh Diagram Keluaran *Absolut Encoder* 4-Bit Tipe *Binary Code*

Tabel 2.1 *Switching Table* dari vektor tegangan *inverter*

Desimal	Rentang Putaran	Kode Biner	Kode *Gray*
0	0 – 22.5	0000	0000
1	22.5 - 45	0001	0001
2	45 – 67.5	0010	0011
3	67.5 - 90	0011	0010
4	90 – 112.5	0100	0110
5	112.5 - 135	0101	0111
6	135 – 157.5	0110	0101
7	157.5 - 180	0111	0100
8	180 – 202.5	1000	1100
9	202.5 - 225	1001	1101
10	225 – 247.5	1010	1111
11	247.5 - 270	1011	1110
12	270 – 292.5	1100	1010
13	292.5 - 315	1101	1011
14	315 – 337.5	1110	1001
15	337.5 - 360	1111	1000

2.6 Identifikasi Sistem [7]

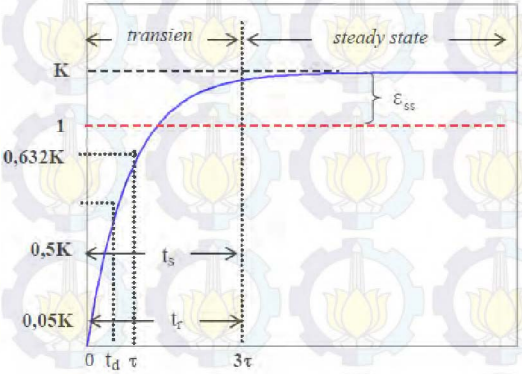
Metode identifikasi merupakan suatu metode yang menggunakan hubungan data masukan dan keluaran yang selanjutnya dilakukan pengujian dan analisa dengan metode pendekatan sehingga dapat ditentukan nilai parameter secara analitik. Beberapa sinyal masukan

yang dapat digunakan untuk mendapatkan respon suatu sistem seperti sinyal *step, ramp, impulse*, dan sinusoidal.

Terdapat dua macam identifikasi sistem yaitu identifikasi statis dan identifikasi dinamis. Untuk identifikasi statis, sinyal yang diberikan berupa sinyal *step* yang konstan sampai sistem mencapai keadaan *steady state*. Sedangkan untuk identifikasi dinamis, sinyal yang digunakan berupa sinyal acak (*random*).

2.6.1 Identifikasi Statis

Suatu sistem disebut system orde satu apabila dilihat secara grafis memiliki bentuk respon seperti yang ditunjukkan pada Gambar 2.18.



Gambar 2.18 Karakteristik Respon Orde Satu

Karakteristik respon sistem orde satu dilihat berdasarkan respon sistem ketika sistem diberi masukan sinyal *step.* Model matematik sistem orde satu dapat dirumuskan dengan:

$$G(s) = \frac{K}{\tau s+1} \tag{2.4}$$

Dimana K menyatakan *gain overall* dan τ menyatakan *time constant* (konstanta waktu). Nilai K merupakan hasil perhitungan dari persamaan 2.5 :

$$K = \frac{Y_{ss}}{X_{ss}} \tag{2.5}$$

Y_{ss} adalah keluaran saat *steady state* dan X_{ss} adalah masukan saat *steady state*. Karakteristik sistem orde satu dibedakan menjadi karakteristik respon transien dan karakteristik respon keadaan tunak atau *steady state*. Karakteristik respon transien pada orde satu terdiri dari :

1. Spesifikasi Teoritis

   Time constant atau konstanta waktu (τ) merupakan waktu yang dibutuhkan respon mulai t = 0 sampai respon mencapai 63,2% dari respon *steady state*. Konstanta waktu menyatakan kecepatan respon sistem.

2. Spesifikasi Praktis

   a. *Settling time* atau waktu tunak (t_s) merupakan waktu yang menyatakan bahwa respon sistem telah masuk pada daerah *steady state*. Jika dihubungkan dengan konstanta waktu τ, maka t_s dapat dirumuskan dengan :

   $$t_s\ (\pm 5\%) \approx 3\tau \quad (2.6)$$
   $$t_s\ (\pm 2\%) \approx 4\tau \quad (2.7)$$
   $$t_s\ (\pm 0.5\%) \approx 5\tau \quad (2.8)$$

   b. *Rise time* atau waktu naik (t_r) merupakan waktu yang menyatakan bahwa respon sistem telah naik dari 5% - 95% atau 10% - 90% dari nilai respon *steady state*.

   $$t_r\ (5\% - 95\%) = \tau\ ln\ 19 \quad (2.9)$$
   $$t_r\ (10\% - 90\%) = \tau\ ln\ 9 \quad (2.10)$$

   c. *Delay time* atau waktu tunda (t_d) merupakan waktu yang dibutuhkan respon mulai t = 0 sampai respon mencapai 50% dari nilai *steady state*. Waktu tunda menyatakan besarnya faktor keterlambatan respon akibat proses *sampling*.

   $$t_d = \tau\ ln\ 2 \quad (2.11)$$

Karakteristik respon *steady state* sistem orde satu diukur berdasarkan kesalahan pada keadaan tunak atau *error steady state* (e_{ss}), yaitu :

$$e_{ss} = Y_{ss} - X_{ss} \quad (2.12)$$

2.6.1.1 Identifikasi Statis dengan Metode Vítečková Orde 1

Metode ini menggunakan pendekatan orde satu dengan kemungkinan adanya waktu tunda. Fungsi alih untuk metode Vitečková Orde 1 ditunjukkan pada Persamaan 2.12.

$$G_{V1}(s) = \frac{K}{\tau_{V1}\, s + 1} e^{-T_{dV1}\, s} \quad (2.13)$$

Dimana T_{dV1} merupakan waktu tunda (*delay time*) dan dapat dicari menggunakan persamaan :

$$T_{dV1} = 1{,}498\ t_{33} - 0{,}498\ t_{70} \quad (2.14)$$

Kemudian τ_{V1} merupakan konstanta waktu dengan persamaan :

$$\tau_{V1} = 1{,}245\ (t_{70} - t_{33}) \quad (2.15)$$

Dimana t_{33} dan t_{70} merupakan waktu saat respon berada pada kodisi 33% dan 70% dari keluaran *steady state*. Apabila T_{dV1} bernilai negatif, maka sistem dianggap tidak memiliki waktu tunda.

2.6.1.2 Identifikasi Statis dengan Metode Vítečková Orde 2

Metode ini menggunakan pendekatan orde dua dengan kemungkinan adanya waktu tunda. Fungsi alih untuk metode Vitečková Orde 2 ditunjukkan pada Persamaan 2.15.

$$G_{V2}(s) = \frac{K}{(\tau_{V2}\, s + 1)^2} e^{-T_{dV2}\, s} \quad (2.16)$$

Dimana T_{dV2} merupakan waktu tunda (*delay time*) dan dapat dicari menggunakan persamaan :

$$T_{dV2} = 1{,}937\ t_{33} - 0{,}937\ t_{70} \quad (2.17)$$

Kemudian τ_{V2} merupakan konstanta waktu dengan persamaan :

$\tau_{V2} = 0{,}794\ (t_{70} - t_{33})$ (2.18)

Dimana t_{33} dan t_{70} merupakan waktu saat respon berada pada kodisi 33% dan 70% dari keluaran *steady state*. Apabila T_{dV2} bernilai negatif, maka sistem dianggap tidak memiliki waktu tunda.

2.6.1.3 Identifikasi Statis dengan Metode Sundaresan & Krishnaswamy

Metode ini menggunakan pendekatan orde satu dengan kemungkinan adanya waktu tunda. Fungsi alih untuk metode Sundaresan & Krishnaswamy ditunjukkan pada Persamaan 2.19.

$$G_{SK}(s) = \frac{K}{\tau_{SK}\, s + 1} e^{-T_{dSK}\, s} \quad (2.19)$$

Dimana T_{dSK} merupakan waktu tunda dan dapat dicari menggunakan persamaan :

$T_{dSK} = 1{,}3\ t_{35,3} - 0{,}29\ t_{85,3}$ (2.20)

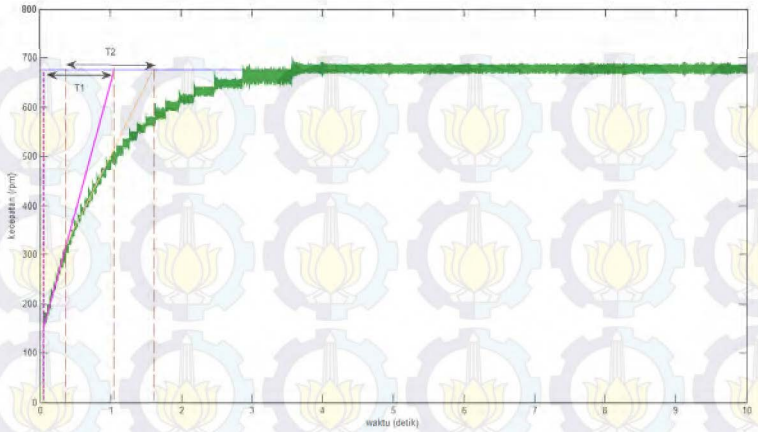
Kemudian τ_{SK} merupakan konstanta waktu dengan persamaan :

$\tau_{SK} = 0{,}67\ (t_{85,3} - t_{35,3})$ (2.21)

Dimana $t_{35,3}$ dan $t_{85,3}$ merupakan waktu saat respon berada pada kodisi 35,3% dan 85,3% dari keluaran *steady state*. Apabila τ_{dSK} bernilai negatif, maka sistem dianggap tidak memiliki waktu tunda.

2.6.1.4 Identifikasi Statis dengan Metode Grafis Terstruktur

Metode ini merupakan secara grafis dengan menarik garis singgung yang memotong respon plant.



Gambar 2.19 Metode Grafik Terstruktur

T_1 dan T_2 merupakan waktu yang digunakan untuk menentukan akar-akar karakteristik dari sistem. Langkah – langkah untuk mendapatkan T_1 dan T_2 adalah sebagai berikut ;

1. Tarik garis singgung dari awal respon sampai garis tersebut mempunyai kemiringan yang sama dengan respon. Garis akan berakhir pada Xss.
2. Tarik garis lurus dari awal repon sampai Xss.
3. Selisih antara garis tersebut merupakan nilai T_1.
4. Tarik lagi garis yang menyinggung dari akhir garis singgung yang pertama sampai menuju Xss.
5. Ulangi lagi langkah 2 pada mulai dari awal garis 2 berpotongan dengan respon.
6. Nilai T_2 adalah selisih 2 garis tersebut.
7. Setelah nilai T_1 dan T_2 didapatkan maka bentuk didapatkan fungsi transfer sebagai berikut :

$$G\ (s) = \frac{1/t1}{s+1/t1} + \frac{1/t2}{s+1/t2} \qquad (2.22)$$

2.7 Validasi Model

Identifikasi parameter sistem didapatkan data untuk mendapatkan model matematik *plant.* Tujuan validasi model secara umum adalah untuk membuktikan bahwa model yang diidentifikasi memenuhi persyaratan permodelan menurut kriteria tujuan (*objective*) dari aproksimasi permodelan yang baik. Model matematika tersebut perlu diuji validasi untuk mengetahui kesamaan dengan *plant* dalam kondisi nyata. Terdapat beberapa metode validasi model, salah satunya yaitu *Root Mean Square Error* seperti uji validasi model untuk mengetahui perbandingan dengan data yang didapat dengan model matematika melalui perhitungan yang disimulasikan.

RMSE adalah pengukuran akurasi pada nilai deret waktu secara statistik, khususnya *trend*. Persamaan (2.23) menyajikan formulasi dasar dari RMSE.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} {e_i}^2} \qquad i = 1,2,3,4\ldots n \tag{2.23}$$

Dimana e adalah persentase kesalahan data hasil dari permodelan terhadap hasil pengukuran seperti yang dirumuskan pada Persamaan (2.24).

$$e_i = A_i - M_i \quad i = 1,2,3,4,..n \tag{2.24}$$

Keterangan :

n : jumlah data
i : urutan data
A : nilai data hasil pengukuran
M : nilai data hasil permodelan

RMSE adalah komponen terpadu dari sebuah model statistika seperti regresi.Hal ini menjadikan RMSE sebagai sebuah pengukuran alami yang digunakan dalam berbagai evaluasi terhadap perkiraan kesalahan yang menggunakan metode statistika khususnya regresi. Secara umum, tidak ada kriteria mutlak dari sebuah data untuk dianggap bagus. Keuntungan yang dimiliki RMSE adalah kesamaan skala yang dimiliki oleh data hasil pengukuran dengan data hasil permodelan. Dengan demikian, RMSE dapat merepresentasikan ukuran dari kesalahan rata-rata. Nilai mutlak dari penghitungan ini dijumlahkan untuk setiap titik yang dilengkapi atau perkiraan dalam domain waktu

dan dibagi kembali dengan jumlah n pengambilan data. Hal ini membuat kesalahan diukur dalam persen sehingga kesalahan sebuah deret waktu dapat dibandingkan di tingkat yang berbeda. Selain mudah dihitung, RMSE memiliki aplikasi yang luas baik dalam bidang penaksiran maupun statistika.

2.8 Teori Kontroler Logika *Fuzzy* [8]

Sebelum konsep logika *fuzzy* diperkenalkan, orang telah mengenal konsep yang disebut logika klasik yang membagi sifat parameter menjadi dua hal yang berlawanan secara tegas, seperti benar atau salah, 0 atau 1. Konsep ini ternyata mempunyai kekurangan dalam penerapannya di kehidupan nyata, karena manusia lebih mengenal konsep linguistik yang menyatakan sesuatu dengan tidak eksak atau samar. Konsep logika *fuzzy* mengubah konsep logika klasik menjadi konsep yang memetakan suatu variabel pada kemungkinan yang tidak eksak sehingga didapatkan sistem linguistik dan permasalahan yang tidak pasti atau tidak presisi serta permasalahan probabilitas.

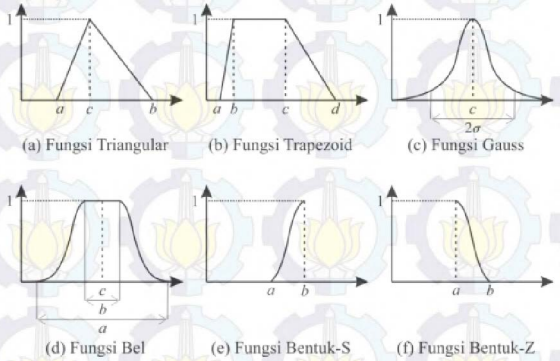
Konsep logika *fuzzy* berawal dari teori *fuzzy set* sebagai salah satu pendekatan untuk menyelesaikan permasalahan yang tidak memiliki ketentuan yang pasti. Teori tersebut dikembangkan oleh Lotfi Zadeh pada tahun 1965 di University of California – Barkeley. Dalam jurnalnya, Zadeh memperkenalkan konsep teori himpunan baru yang dinamakan *fuzzy* set. Konsep logika *fuzzy* menggantikan konsep "benar-salah" dari logika *boolean* menjadi derajat tingkat kebenaran. Teori *fuzzy* menyatakan keanggotaan suatu objek ke dalam fungsi derajat keanggotaan (*membership function*). Hal tersebut memungkinkan keanggotaan suatu objek dapat dinyatakan pada semua bilangan *riil* antara 0 sampai 1. Oleh karena itu, konsep *fuzzy* tersebut sesuai dengan pola pikir manusia yang cenderung menilai suatu objek secara samar.

2.8.1 Himpunan *Fuzzy* (*Fuzzy Set*)

Himpunan *fuzzy* merupakan suatu himpunan yang beranggotakan sejumlah istilah dalam pengertian bahasa yang menyatakan level kualitatif dari semesta pembicaraan. Misalnya pengukuran kecepatan putaran motor dapat diterjemahkan ke dalam beberapa istilah bahasa yang menyatakan level kualitatif dari kecepatan putaran motor. Sehingga apabila semesta pembicaraan berupa kecepatan putaran motor, maka dapat dibuat suatu himpunan *fuzzy* yaitu "Sangat Lambat", "Lambat, "Sedang", "Cepat", "Sangat Cepat".

2.8.2 Fungsi Keanggotaan (*Membership Function*)

Fungsi keanggotaan merupakan suatu fungsi yang didefinisikan untuk suatu anggota himpunan *fuzzy* yang menggambarkan derajat kebenaran suatu kejadian dalam semesta pembicaraan. Fungsi keanggotaan dinyatakan dalam tingkat keanggotaan dengan nilai antara 0 s/d 1. Beberapa fungsi keanggotaan yang biasa digunakan untuk merepresentasikan nilai keanggotan dari suatu himpunan *fuzzy* diantaranya seperti Gambar 2.20 yang menunjukkan bentuk-bentuk dari fungsi keanggotaan tersebut.



Gambar 2.20 Bentuk-Bentuk *Membership Function*

Persamaan dari masing-masing *membership function* dapat dinyatakan seperti Persamaan 2.25 - 2.28.

$$Segitiga : \mu_A(x) = \begin{cases} 0 & x \le a \\ \dfrac{x-a}{b-a} & a \le x \le b \\ \dfrac{c-x}{c-b} & b \le x \le c \\ 0 & x \ge c \end{cases} \tag{2.25}$$

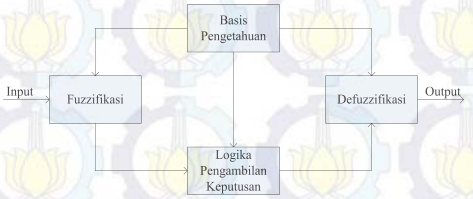
$$Bell\text{-}shaped : \mu_A(x) = exp\left[-\left(\frac{u_i - \bar{x}}{\sigma} \right)^2 \right] \tag{2.26}$$

$$Trapesium : \mu_A(x) = \begin{cases} 0 & x \le a \\ \frac{x-a}{b-a} & a \le x \le b \\ 1 & b \le x \le c \\ \frac{c-x}{c-b} & x \ge c \end{cases} \quad (2.27)$$

$$Singleton : \mu_A(x) = 1, \quad (2.28)$$

2.8.3 Struktur Dasar Logika *Fuzzy*

Kontroler logika *fuzzy* merupakan suatu kontroler yang proses perhitungan sinyal kontrolnya melalui operasi himpunan *fuzzy* meliputi proses *fuzzifikasi*, relasi *fuzzy*, inferensi *fuzzy* serta *defuzzifikasi*. Pada dasarnya struktur logika *fuzzy* tampak seperti Gambar 2.21 berikut:



Gambar 2.21 Struktur Logika *Fuzzy*

Gambar 2.21 menunjukkan struktur logika *fuzzy*. Fungsi dari bagian-bagian di atas adalah sebagai berikut:

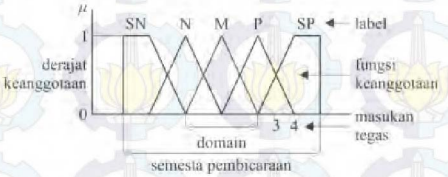
1. *Fuzzifikasi* berfungsi untuk mentransformasikan sinyal *input* yang bersifat *crisp* (bukan *fuzzy*) ke himpunan *fuzzy* dengan menggunakan operator *fuzzifikasi*.
2. Basis pengetahuan berisi basis data dan aturan dasar yang mendefinisikan himpunan *fuzzy* atas daerah-daerah *input* dan *output* dan menyusunnya dalam perangkat aturan kontrol.
3. Logika pengambilan keputusan merupakan inti dari logika *fuzzy* yang mempunyai kemampuan seperti manusia dalam mengambil keputusan. Aksi atur *fuzzy* disimpulkan dengan menggunakan implikasi *fuzzy* dan mekanisme inferensi *fuzzy*.

4. Defuzzifikasi berfungsi untuk mentransformasikan kesimpulan tentang aksi atur yang bersifat *fuzzy* menjadi sinyal sebenarnya yang bersifat *crisp* dengan menggunakan operator fuzzifikasi.

2.8.4 Fuzzifikasi

Fuzzifikasi adalah proses pemetaan *input* dan *output* sistem agar sesuai dengan himpunan *fuzzy*. Pemetaan digunakan dengan cara yang disebut fungsi keanggotaan (*membership function*). Ada banyak metode yang digunakan untuk proses fuzzifikasi seperti yang telah dijelaskan sebelumnya, tetapi bentuk *triangular* dan *trapezoid* yang paling banyak digunakan dalam pembentukan fungsi keanggotaan pada logika *fuzzy*. Hal ini dikarenakan metode bentuk *triangular* dan *trapezoid* lebih mudah diimplementasikan pada kontroler.

Struktur fungsi keanggotaan *fuzzy* dapat dilihat pada Gambar 2.22. Derajat keanggotaan adalah derajat dari masukkan tegas pada sebuah fungsi keanggotaan, memiliki nilai 0 s/d 1. Fungsi keanggotaan mendefinisikan nilai *fuzzy* dengan melakukan pemetaan nilai tegas berdasarkan daerahnya untuk diasosiasikan dengan derajat keanggotaan.



Gambar 2.22 Struktur fungsi keanggotaan *fuzzy*

Masukkan tegas pada umumnya merupakan hasil pengukuran parameter eksternal dari sistem kontrol. Label merupakan deskripsi nama untuk menunjukkan suatu fungsi keanggotaan *fuzzy* yang memiliki domain (lebar fungsi keanggotaan *fuzzy*) tertentu. Semesta pembicaraan memiliki jarak yang mencakup seluruh masukkan tegas yang mungkin ada. Bentuk fungsi keanggotaan harus mewaliki variabel masukkan tegas, namun bentuk yang digunakan dibatasi oleh kemampuan *tool* dalam melakukan perhitungan. Bentuk fungsi yang rumit membutuhkan persamaan fungsi yang lebih kompleks.

2.8.5 Aturan Dasar *Fuzzy*

Aturan dasar *fuzzy* adalah kaidah dasar yang berisi aturan-aturan secara linguistik yang menunjukkan kepakaran terhadap *plant*. Penentuan aturan dasar yang dipakai dalam mengontrol suatu *plant* dapat melalui metode heuristik maupun deterministik. Metode heuristik didasarkan pada pengetahuan terhadap *plant* dan perilaku dari *plant* yang akan dikontrol. Sedangkan metode deterministik diperoleh melalui identifikasi struktur dan parameter dari aturan kontrol. Banyak cara menunjukkan suatu kepakaran ke dalam aturan, format yang paling umum adalah sebagai berikut :

1. Format Aturan *IF-THEN*

   Pemetaan format aturan *IF-THEN* direpresentasikan dalam Persamaan 2.29.

   IF premis THEN conclusion (2.29)

   Dimana premis berupa input kontroler dan conclusion berupa output kontroler. Dengan demikian dari kepakaran dapat diambil kesimpulan, apabila pernyataannya lebih dari satu maka dapat digunakan logika "*AND*" atau "*OR*".

   Contoh penggunaan aturan *IF-THEN* sebagai berikut :

   IF error is N THEN output is NB

   IF error is Z THEN output is Zero

   IF error is P THEN output is PB

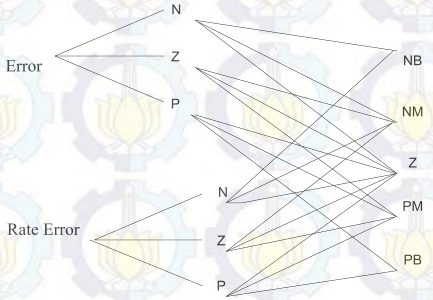
   Jumlah basis aturan dari suatu sistem *fuzzy* ditentukan dari jumlah variabel pada input dan jumlah *membership function* pada variabel masukkan, dirumuskan dalam persamaan 2.30.

$$\prod_{i=1}^{n} N_i = N_1 \times N_2 \times .... \times N_n \qquad (2.30)$$

   Dimana N_i merupakan jumlah *membership function* pada variabel input i. Sebagai contoh apabila variabel input pertama memiliki dua *membership function* dan variabel input kedua memiliki dua *membership function*, maka jumlah basis aturan adalah 2x2 = 4 aturan.

2. Format Hubungan

Pada dasarnya sama dengan aturan *IF-THEN* hanya saja tampilannya lebih sederhana karena menggunakan hubungan garis. Contoh dari penggunaan format hubungan dapat dilihat pada Gambar 2.23.



Gambar 2.23 Aturan Dasar *Fuzzy* Format Hubungan

Gambar 2.7 menjelaskan NB adalah *Negative Big*, NM adalah *Negative Medium*, Z adalah *Zero*, PM adalah *Positive Medium*, dan PB adalah *Positive Big* seperti pada Tabel 2.2.

3. Format *Tabular*

Tabel 2.2 menunjukkan format tabular lebih sederhana daripada format hubungan, variabel linguistik berada pada sisi luar dari tabel, sedangkan sisi dalam berisi dari keputusannya.

Tabel 2.2 Format Tabular

Error/Rate Error	Neg	Zero	Pos
Neg	NB	NM	Zero
Zero	NM	Zero	PM
Pos	Zero	PM	PB

2.8.6 Logika Pengambilan Keputusan

Inferensi *fuzzy* adalah suatu proses formulasi pemetaan *input* terhadap *output* dengan menggunakan logika *fuzzy*. Proses dari inferensi

fuzzy melibatkan fungsi keanggotaan operator logika *fuzzy* dan aturan *if-then*.

Terdapat dua metode inferensi yang paling dikenal, yaitu metode inferensi *Mamdani* dan metode *Takagi-Sugeno*. Metode inferensi *Mamdani* menggunakan fungsi keanggotaan *fuzzy* pada bagian *output*nya. Sehingga setelah proses aturan telah diterapkan, terdapat himpunan *fuzzy* yang harus didefuzzifikasi. Umumnya proses defuzzifikasi berlangsung lebih lambat akibat proses komputasi pada *output*nya.

Metode *Takagi-Sugeno* menggunakan fungsi keanggotaan *output* yang linier atau berupa konstanta. Sedangkan dua bagian pada proses inferensi yaitu fuzzifikasi dan penerapan operator *fuzzy* sama dengan metode inferensi *Mamdani*.

Bentuk aturan *fuzzy Takagi-Sugeno* memiliki bentuk sebagai berikut :

$$\text{If input1} = x \text{ and input2} = y, \text{ then output is } z = ax + by + c \qquad (2.31)$$

Perbedaan dari kedua metode ini terletak pada pengambilan kesimpulan logika *fuzzy*. Pada metode Mamdani, kesimpulan logika *fuzzy* berupa derajat keanggotaan sehingga dalam menyimpulkan suatu logika *fuzzy* dibutuhkan proses *defuzzifikasi*. Sedangkan pada tipe Takagi Sugeno, kesimpulan logika *fuzzy* berupa suatu persamaan sehingga tidak diperlukan proses *defuzzifikasi*. Kelebihan pada logika *fuzzy* tipe Mamdani lebih sederhana, akan tetapi diperlukan kemampuan untuk mengetahui karakteristik *plant* untuk menentukan batasan keluaran kontroler. Pada tipe Takagi-Sugeno tidak diperlukan pengetahuan mengenai karakteristik dari *plant* akan tetapi diperlukan perhitungan yang lebih rumit untuk persamaan pada bagian konsekuen.

2.8.7 Defuzzifikasi

Defuzzifikasi adalah proses yang digunakan untuk mengubah kembali variabel *fuzzy* menjadi variabel nyata, atau dengan kata lain aksi pengaturan *fuzzy* yang masih berupa himpunan, diubah menjadi nilai nyata yang berupa nilai tunggal. Banyak metode yang dapat digunakan untuk proses defuzzifikasi, diantaranya adalah metode harga rata-rata maksimum atau *Mean of Maximum* (MOM) dan metode titik pusat luasan atau *Center of Area* (COA). Berikut merupakan persamaan dari masing – masing metode :

Mean of Maximum (MOM) :

$$U_0 = \arg . \sum_{j=1}^{J} \{\max[\mu_u(u_j(T))]\} J^{-1} \ \forall u \in U(T) \qquad (2.32)$$

Center of Area (COA) :

$$U_0 = \frac{\sum_{k=1}^{m} u_k(T) \bullet \mu_u(u_k(T))}{\sum_{k=1}^{m} \mu_k(u_k(T))} \ \forall u \in U(T) \qquad (2.33)$$

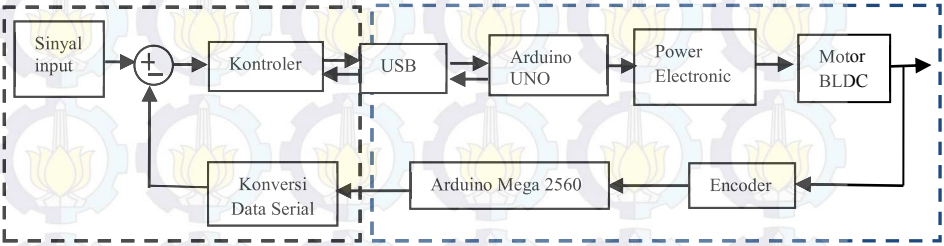
Halaman ini sengaja dikosongkan

BAB 3
PERANCANGAN SISTEM

Pada bab ini membahas mengenai perancangan keseluruhan sistem mulai dari perancangan sistem bagian *software*, sistem bagian *hardware*, identifikasi motor BLDC, dan perancangan kontroler *Fuzzy*-PI.

3.1 Diagram Blok Sistem

Gambar 3.1 menunjukkan diagram blok keseluruhan sistem dari perancangan penggerak elektrik dan pengaturan kecepatan motor BLDC.



Gambar 3.1 Diagram blok sistem

Keterangan :
: Sistem bagian *software* (PC dan *software* MATLAB)
: Sistem bagian *hardware*

Pada diagram blok menunjukkan bahwa sistem terdiri dari beberapa bagian atau subsistem. Bagian sinyal *input* merupakan bagian untuk memasukkan nilai input yang diinginkan. Blok kontroler berisi struktur kontroler *fuzzy*-PI.

Blok USB merupakan bagian dari PC yang berfungsi untuk menghubungkan ke arduino. PC disini juga berfungsi sebagai HMI (*Human Machine Interface*) yang digunakan untuk mengetahui respon keluaran sistem, membuat kontroler dan juga digunakan untuk pemberian nilai sinyal *input* yang diinginkan.

Blok *power electronic* sebagai aktuator yang digunakan untuk pemberian aksi dalam pengaturan kecepatan motor BLDC yang telah

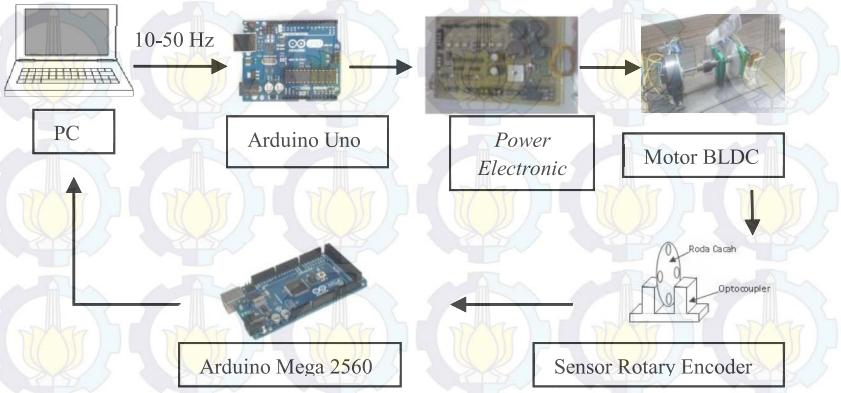
dibuat oleh kontroler. Arduino berfungsi sebagai pembangkit frekuensi fasa, PWM, dan pembacaan data dari sensor *encoder*.

Sensor *rotary encoder* berfungsi sebagai *feedback* berupa sensor kecepatan yang digunakan untuk mengetahui nilai *error* yang terjadi dalam sistem kontrol terhadap *set point*. Nilai *error* yang akan digunakan oleh kontroler untuk memberikan aksi kontrol hingga nilai *error* semakin kecil.

Sehingga secara keseluruhan sistem ini bertujuan untuk membuat rangkaian *power electronic* sebagai komutasi elektrik yang berfungsi menggerakkan motor BLDC dan menghasilkan kecepatan motor yang diinginkan. Prinsip kerja dari sistem ini yaitu rangkaian *power electronic* digunakan untuk menyuplai tegangan AC tiga fasa ke motor BLDC yang akan dioperasikan pada kecepatan tertentu sesuai dengan sinyal *input*. Sistem pengaturan digunakan untuk mengatur kecepatan motor supaya mendekati dengan sinyal *input*.

3.2 Perancangan Perangkat Keras (*Hardware)*

Untuk perancangan perangkat keras (*hardware*) dari sistem terdiri dari beberapa konponen yaitu arduino, rangkaian *power electronic*, rangkaian mekanik motor BLDC, rangkaian sensor *rotary encoder,* rem magnetik, dan PC. Arduino yang digunakan dalam sistem ini adalah arduino UNO dan arduino MEGA 2560. Komponen dan alur dari perangkat keras sistem secara spesifik ditunjukkan pada Gambar 3.2.



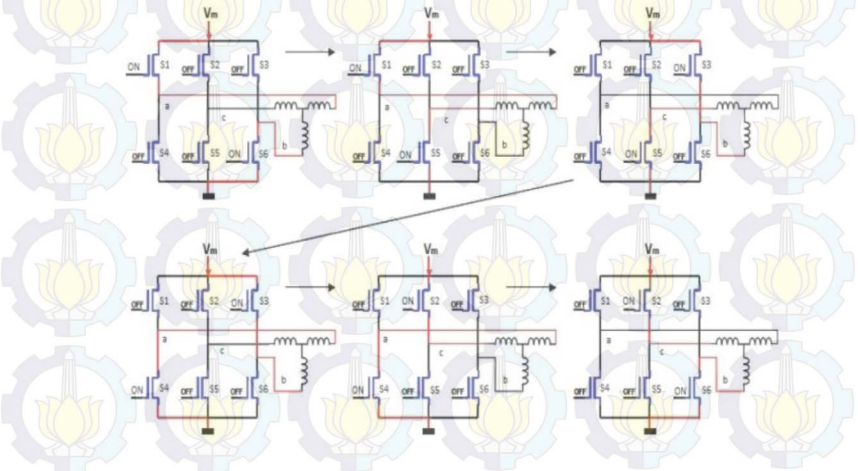
Gambar 3.2 Perancangan Perangkat Keras (*Hardware*)

Nilai sinyal *input* diberikan melalui PC dan selanjutnya dikirim ke arduino melalui *serial port* yang akan memberikan keluaran berupa frekuensi dan PWM ke rangkaian *power electronic*. *Power electronic* akan mengolah frekuensi tegangan masukan dari arduino menjadi frekuensi 10-50 Hz tiga fasa dan PWM dengan *duty cycle* 80% dengan frekuensi 4 KHz.

Sensor *rotary encoder* digunakan untuk mengukur kecepatan motor BLDC yang berputar diantara *optocoupler* tipe U. Keluaran dari sensor berupa pulsa dan dihubungkan ke arduino mega. Hasil pembacaan kecepatan dari *rotary encoder* digunakan untuk mengetahui nilai *error* yang terjadi dengan nilai sinyal *input* yang akan menjadi paremeter untuk mengubah sinyal kontrol.

3.2.1 Perancangan Rangkaian *Power Electronic*

Motor BLDC membutuhkan enam langkah komutasi yang dilakukan secara kontinyu untuk berputar. *Driver* 3 fasa terdiri dari 6 buah saklar yang akan memberikan tegangan positif (sinyal *high*) dan tegangan 0V (sinyal *low*) secara bergantian. Dimana MOSFET (*The metal–oxide–semiconductor field-effect transistor*) akan digunakan sebagai saklar dengan IC pembagi fasa 74HC175 yang akan mengendalikan fase yang masuk *gate* MOSFET tersebut.



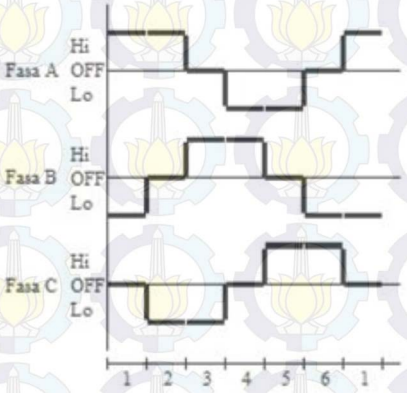
Gambar 3.3 Urutan Pensaklaran pada *Stator*

Sesuai gambar diatas, maka terdapat urutan penyaklaran pada *stator* yang dapat dilihat pada tabel berikut :

Tabel 3.1 Urutan Pengaturan Saklar motor BLDC

Urutan ke-	Saklar Aktif		Fasa A	Fasa B	Fasa C
1	S1	S6	High	Low	Off
2	S1	S5	High	Off	Low
3	S3	S5	Off	High	Low
4	S3	S4	Low	High	Off
5	S2	S4	Low	Off	High
6	S2	S6	Off	Low	High

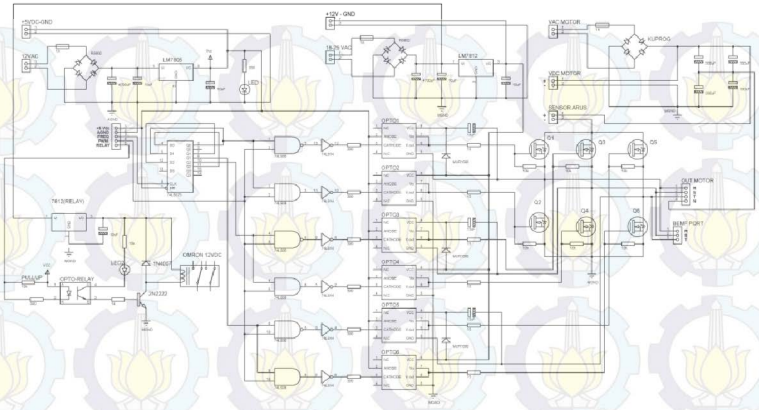
Kemudian berikut merupakan gambar hasil bentuk gelombang dari penyaklaran pada stator motor BLDC :



Gambar 3.4 Bentuk Gelombang Hasil Penyaklaran pada *Stator*

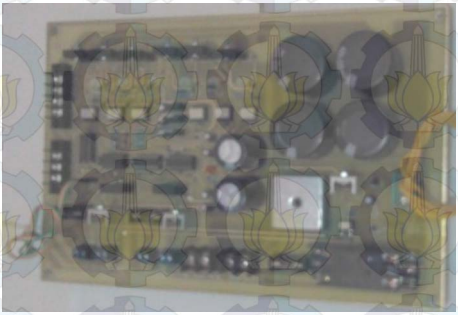
Pada rangkaian *power electronic* terdapat beberapa bagian yang saling melengkapi agar secara keseluruhan rangkaian *power electronic* dapat bekerja dengan baik diantaranya adalah rangkaian *power supply* , pengolah sinyal fasa dan PWM, rangkaian *optocoupler*, dan bagian *switching*.

Berikut merupakan skema dari rangkaian *power electronic* :



Gambar 3.5 Skema Rangkaian *Power Electronic*

Sedangkan bentuk fisik untuk rangkaian *power electronic* terlihat seperti pada gambar berikut :



Gambar 3.6 Bentuk Fisik Rangkaian *Power Electronic*

Rangkaian *power supply* pada gambar diatas menggunakan tegangan sebesar 12 VAC, 18 VAC, dan 64 VAC yang nantinya akan diubah menjadi tegangan DC sebesar 5 VDC, 12 VDC, dan 90 VDC.

3.2.2 Perancangan Mekanik *Plant*

Motor BLDC yang digunakan dalam Tugas Akhir ini adalah motor BLDC ARW31S8P30AM buatan Malaysia dengan spesifikasi seperti pada Gambar 3.7.

Gambar 3.7 Spesifikasi motor BLDC

Berikut ini merupakan gambar dari rancangan konstruksi *plant* motor BLDC :

Gambar 3.8 Kontruksi *Plant* Motor BLDC

Pada gambar dapat dilihat bahwa *shaft* dari motor BLDC diberi piringan alumunium yang akan menjadi beban rem magnetik ketika motor berputar. Beban rem magnetik tersebut digunakan untuk mempengaruhi sistem kerja dari motor BLDC. Pengaruh yang ditimbulkan oleh rem magnetik berupa penurunan kecepatan dari motor BLDC.

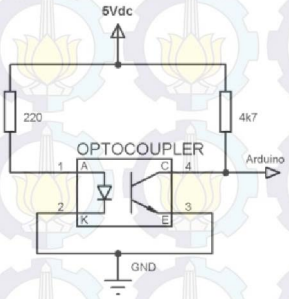
Shaft motor BLDC terhubung dengan *shaft* yang terdapat pada piringan alumunium. Penghubung antar *shaft* menggunakan *coupling*. *Coupling* berfungsi untuk menggabungkan dua *shaft* dan untuk meredam getaran yang ditimbulkan antar *shaft* saat motor berputar.

Rem magnetik pada *plant* terdiri dari dua buah magnet permanen. Magnet diletakkan pada dudukan yang didesain agar bisa diubah posisinya saat motor berputar. Pada ujung *shaft* diletakkan piringan dari sensor *rotary encoder*.

3.2.3 Perancangan Sensor *Rotary Encoder*

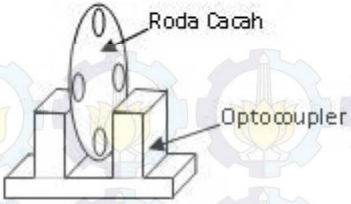
Pada Tugas Akhir ini sensor kecepatan yang digunakan adalah piringan atau roda cacah dan sebuah *optocoupler* tipe U. Piringan *rotary encoder* berputar diantara celah pada *optocoupler*. Optocoupler merupakan komponen *optoisolator* yang memiliki karakteristik penerima (*phototransistor*) akan mengalami perubahan logika 0 ke 1 atau sebaliknya bila terjadi perubahan intensitas cahaya yang dipancarkan oleh pemancar (LED infra merah) untuk penerima.

Phototransistor merupakan jenis transistor yang peka terhadap cahaya infra merah. Piringan akan ditempatkan di tengah dari *optocoupler* tipe U yang berfungsi untuk mempengaruhi intensitas cahaya yang diberikan oleh LED pada *optocoupler* ke *phototransistor* yang akan memberikan perubahan level logika sesuai dengan putaran piringan. Berikut merupakan skema dari rangkaian sensor *rotary encoder* :



Gambar 3.9 Skema Rangkaian Sensor *Rotary Encoder*

Konstruksi sensor *rotary encoder* dapat dilihat pada Gambar 3.10. Dimana pada konstruksi sensor tersebut terdiri dari roda cacah dan *optocoupler* dengan cara kerja telah dijelaskan pada bab sebelumnya.



Gambar 3.10 Konstruksi Sensor *Rotary Encoder* dengan *Optocoupler*

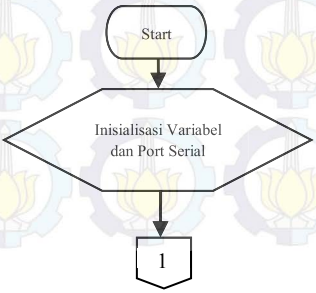
3.3 Perancangan Perangkat Lunak (*Software)*

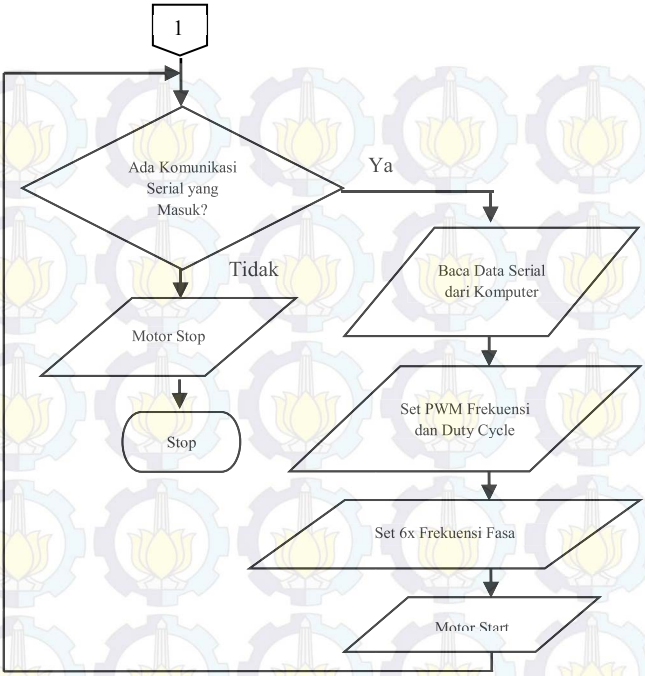
Pada Tugas Akhir ini terdapat perangkat lunak (*software*) yang digunakan dalam perancangan sistem ini diantaranya adalah arduino, dimana arduino ini berfungsi untuk membuat program pembangkit frekuensi dan program membaca kecepatan motor BLDC.

3.3.1 Pemrograman Pembangkit Frekuensi dan PWM

Untuk program pembangkit frekuensi dan PWM ini digunakan arduino UNO dimana perintah untuk membangkitkan frekuensi dilakukan oleh PC melalui *software* MATLAB.

Frekuensi yang dibangkitkan oleh arduino merupakan 6x frekuensi fasa. Hal ini diperlukan karena spesifikasi *hardware* pada *driver inverter* menggunakan rangkaian pembagi fasa. PWM berfungsi untuk memperhalus putaran motor. Syarat frekuensi PWM minimal 4 Khz, hal ini untuk mencegah terjadinya pembacaan PWM di dalam frekuensi fasa, apabila PWM ini masih terbaca oleh rangkaian *switching* maka akan mengakibatkan motor tidak akan berputar. Duty cycle frekuensi fasa harus sebesar 50%, sedangkan duty cycle untuk PWM minimal sebesar 60%. Berikut merupakan *flowchart* dari program pembangkit frekuensi dan PWM :

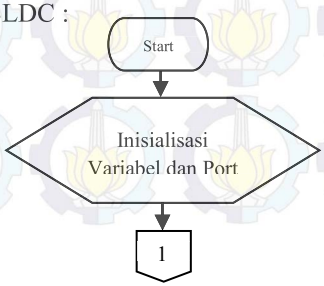


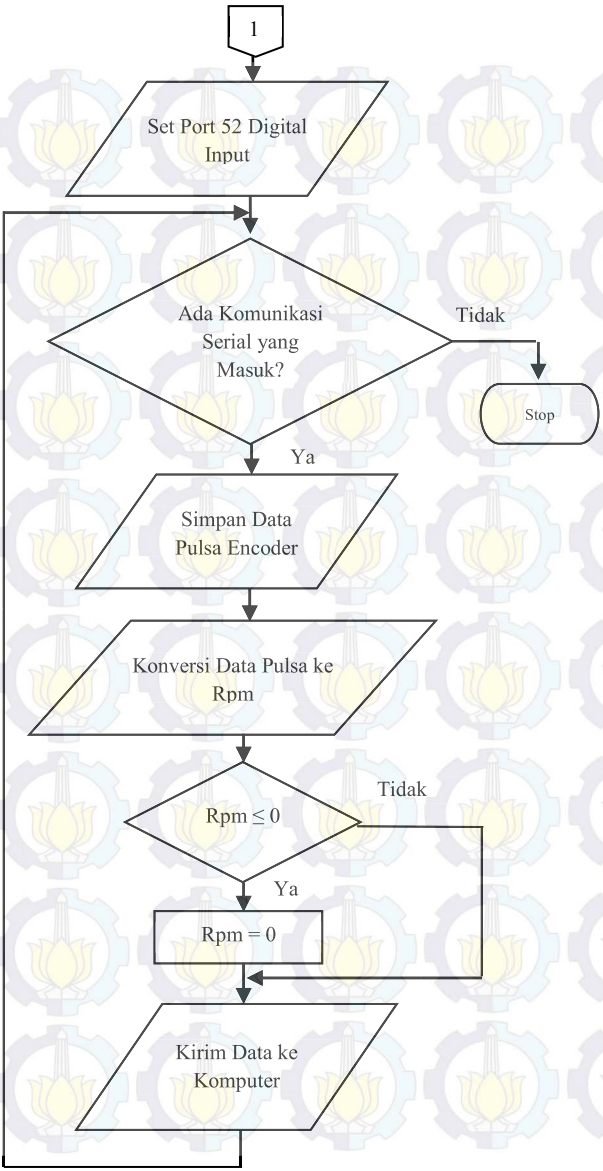


Gambar 3.11 *Flowchart* Program Pembangkit PWM dan Frekuensi

3.3.2 Pemrograman Membaca Kecepatan Motor BLDC

Untuk program pembangkit frekuensi dan PWM ini digunakan arduino Mega 2560 dimana proses pembacaan data kecepatan motor BLDC dengan satuan rpm akan dikirim ke PC melalui *software* MATLAB. Berikut merupakan *flowchart* dari program membaca kecepatan motor BLDC :

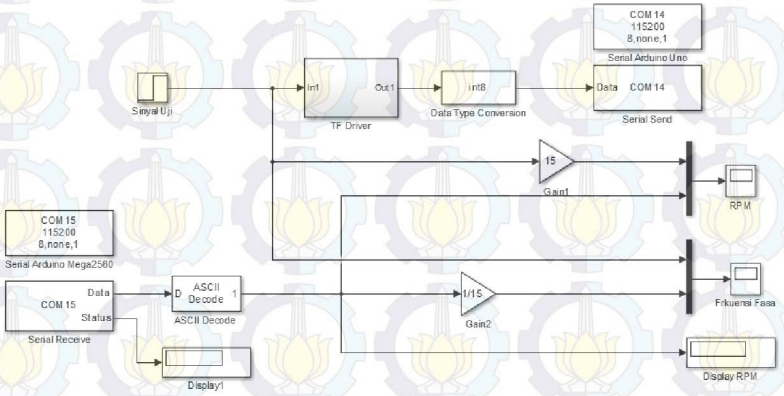




Gambar 3.12 *Flowchart* Program Membaca Kecepatan Motor BLDC

3.3.3 Pemrograman Metode Identifikasi

Perancangan metode identifikasi menggunakan *software* MATLAB R2014a, perancangan ini bertujuan untuk mengidentifikasi *plant* sebelum dilakukan implementasi pengaturan kecepatan pada *plant*. Koneksi antara MATLAB dengan arduino menggunakan blok diagram serial sebagai pengirim dan penerima data.



Gambar 3.13 Blok Diagram Simulink Perancangan Metode Identifikasi

3.4 Identifikasi Sistem

Identifikasi sistem diperlukan untuk mendapatkan model matematika dari motor BLDC. Pada Tugas Akhir ini, dilakukan identifikasi statis dengan melihat keluaran respon kecepatan motor terhadap referensi yang diberikan. Sinyal uji *step* diberikan melalui arduino dengan memberi nilai sinyal *input* sebesar 45 Hz atau 675 rpm. Hasil respon *plant* akan ditampilkan di simulink melalui blok diagram *Serial Receive*. Data kecepatan disimpan dalam bentuk *workspace* dengan *time sampling* sebesar 0.001 detik.

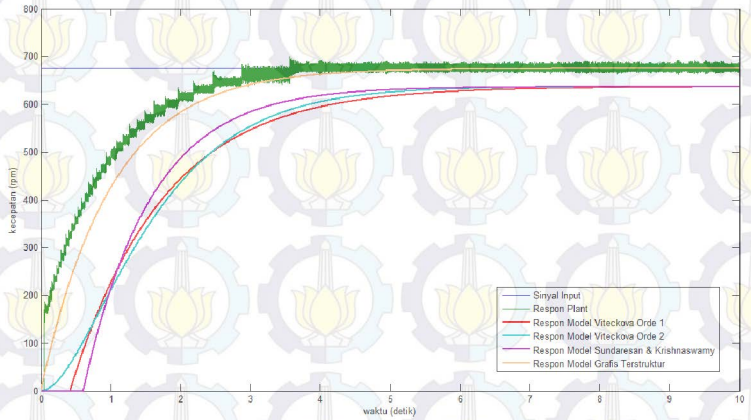
3.4.1 Metode Identifikasi Sistem

Pada Tugas Akhir ini digunakan empat metode identifikasi sistem yaitu metode Viteckova 1st *Order*, metode Viteckova 2nd *Order*, metode Sundaresan – Krishnaswamy dan metode Grafis Terstruktur. Dari keempat metode tersebut akan dicari RMSE yang paling kecil dan mendekati dengan model plant motor BLDC. Berikut merupakan table hasil dari identifikasi sistem dengan beberapa metode identifikasi :

Tabel 3.2 Validasi Model Matematika motor BLDC

No.	Metode	Model Matematika	RMSE
1.	Viteckova 1st *Order*	$\frac{0.9436}{1.3242s+1} \cdot e^{-0.4189s}$	128.221
2.	Viteckova 2nd *Order*	$\frac{0.9436}{0.713s^2+1.689s+1}$	125.978
3.	Sundaresan & Krishnaswamy	$\frac{0.9436}{0.963s+1} \cdot e^{-0.6081s}$	123.895
4.	Grafis Terstruktur	$\frac{1.003(s+1)}{1.007s^2+2.007s+1}$	33.692

Dari keempat hasil identifikasi tersebut dipilih model matematika dengan nilai RMSE paling kecil. Berikut merupakan hasil simulasi dari hasil identifikasi sistem :

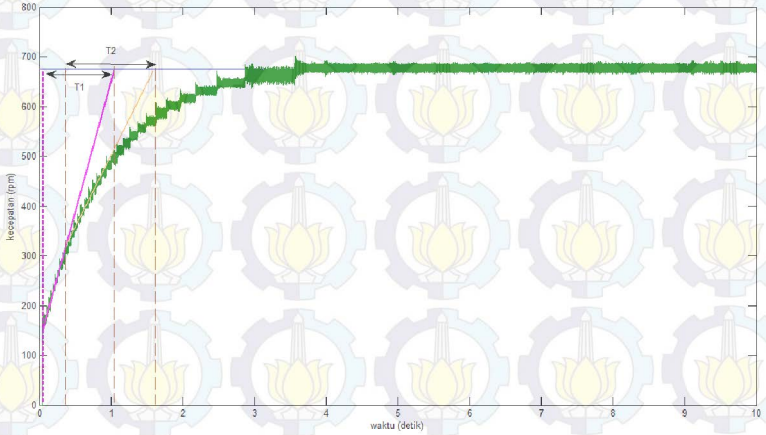


Gambar 3.14 Grafik Hasil Identifikasi

Metode Grafis Terstruktur dipilih karena memiliki validasi nilai RMSE yang terkecil. Hasil respon dari pendekatan Grafis Terstruktur berupa respon orde 2 tanpa *delay*. Dari hasil grafik respon pada MATLAB didapatkan bahwa pendekatan model matematika dengan metode Grafis Terstruktur yang paling mendekati respon *plant* yang sebenarnya.

Perhitungan dalam menentukan pendekatan model matematika dengan metode Grafis Terstruktur adalah sebagai berikut:

1. Plot hasil respon plant
2. Tarik garis singgung yang memotong kurva respon plant, kemudian hitung τ_1 dan τ_2, seperti gambar 3.14



Gambar 3.15 Penarikan Garis Singgung pada Metode Grafis

3. Penurunan rumus model matematika seperti di bawah ini :

$$G(s) = K\left(\frac{1/(1.007)}{s+1.007} + \frac{1/(1)}{s+1}\right) \quad (3.1)$$

Mula-mulai gain K dianggap bernilai 1

4. Menentukan nilai gain (K)
Gain (K) bisa didapat apabila setelah diberi sinyal uji *unit step* sesuai dengan *set point dan* melihat hasil respon melalui *scoope*. Setelah melihat hasil respon, maka nilai dari K didapat melalui rumus :

$$K = Yss\left(\frac{Set\ Point}{Yss}\right) = 1.003\left(\frac{675}{1.003}\right) = 7.522 \quad (3.2)$$

5. Fungsi alih keseluruhan

$$G(s) = 7.522\left(\frac{2.007s + 2}{1.007s^2 + 2.007s + 1}\right)$$

$$= 1.003\left(\frac{s+1}{1.007s^2+2.007s+1}\right) \quad (3.3)$$

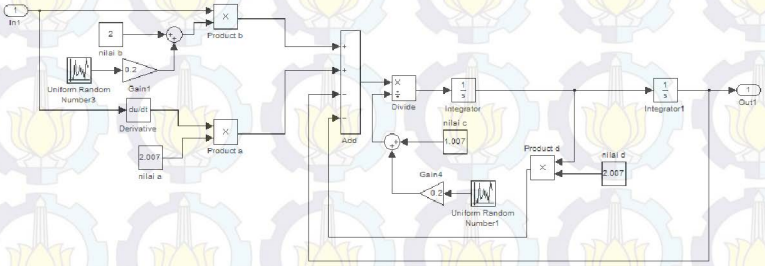
3.4.2 Metode Simulasi Pembebanan Sistem

Untuk simulasi pembebanan sistem menggunakan *plant* parameter bervariasi. Dimana ketika melakukan identifikasi sistem akan mendapatkan model matematika dengan bentuk seperti berikut :

$$G(s) = \frac{as+b}{cs^2+ds+1} \quad (3.4)$$

Dengan didapatkannya model matematika tersebut, maka *plant* parameter bervariasi dibuat dengan mengubah parameter b dan c (asumsi nilai b= ±0.2 dan c = ±0.20) sehingga didapatkan tiga kemungkinan nilai dari masing – masing parameter. Maka secara keseluruhan akan terdapat sembilan kemungkinan dari *plant* parameter bervariasi, dimana hal ini nantinya akan disimulasikan seolah-olah sistem telah mengalami perubahan beban.

Dibuatnya *plant* parameter bervariasi ini bertujuan untuk mengetahui respon kontroler jika *plant* mengalami perubahan parameter. Berikut merupakan blok diagram dari *plant* parameter bervariasi yang disimulasikan pada *software* MATLAB :

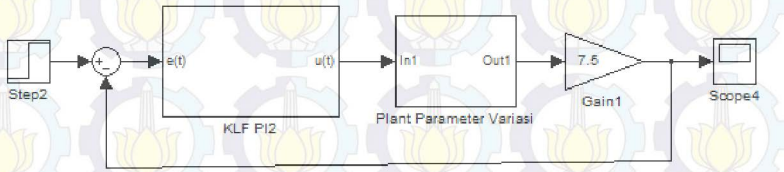


Gambar 3.16 Blok Diagram dari Sub Sistem *Plant* Parameter Bervariasi

Blok diagram tersebut merupakan penurunan dari persamaan 3.4 seperti berikut ini :

$$\frac{Y(s)}{U(s)} = \frac{as+b}{cs^2+ds+1}$$

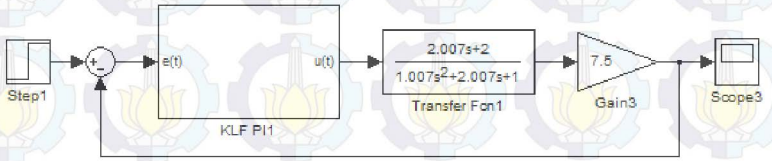
$$cs^2Y(s) + dsY(s) + Y(s) = asU(s) + bU(s)$$
$$c\ddot{y} + d\dot{y} + y = a\dot{u} + bu$$
$$\ddot{y} = \frac{1}{c}(a\dot{u} + bu - d\dot{y} - y) \quad (3.5)$$



Gambar 3.17 Blok Diagram dari *Plant* Parameter Bervariasi

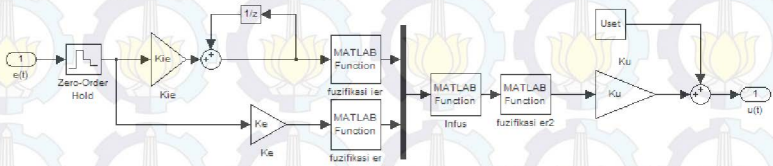
3.5 Perancangan Kontroler *Fuzzy*-PI

Pada Tugas Akhir ini digunakan kontroler *fuzzy*-PI untuk mengatur kecepatan motor BLDC. Kontroler *fuzzy*-PI yang digunakan adalah kontroler *fuzzy*-PI dengan *rule base* PI. Berikut merupakan blok diagram kontroler *fuzzy*-PI :



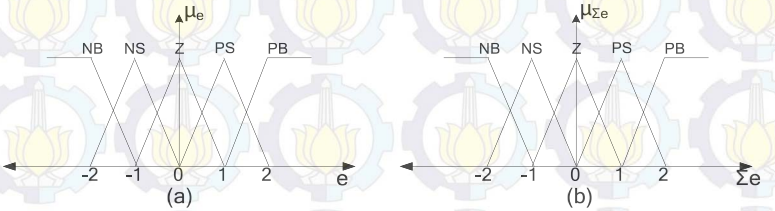
Gambar 3.18 Blok Diagram Kontroler *Fuzzy*-PI

Di dalam blok kontroler *fuzzy*-PI terdapat sub sistem dari kontroler *fuzzy*-PI seperti yang terlihat pada gambar berikut :



Gambar 3.19 Blok Diagram Sub Sistem Kontroler *Fuzzy*-PI

Untuk fungsi keanggotan yang digunakan pada proses *fuzzifikasi* dari *error* dan integral *error* yaitu fungsi segitiga dengan 5 himpunan pendukung untuk *error* dan integral *error*. Gambar 3.20 menunjukkan fungsi keanggotaan dari *error* dan integral *error*.



Gambar 3.20 Fungsi Keanggotan : (a) *error ;* (b) integral *error*

Karena kontroler yang digunakan *fuzzy*-PI adalah dengan *rule base* PI maka tabel *rule base* terlihat pada tabel berikut ini :

Tabel 3.3 Tabel Basis Aturan Mack Vicar Wheelan

	$u_{\sum e}1$	$u_{\sum e}2$	$u_{\sum e}3$	$u_{\sum e}4$	$u_{\sum e}5$
ue1	1	1	2	2	3
ue2	1	2	2	3	4
ue3	2	2	3	4	4
ue4	2	3	4	4	5
ue5	3	4	4	5	5

BAB 4
HASIL SIMULASI DAN PENGUJIAN SISTEM

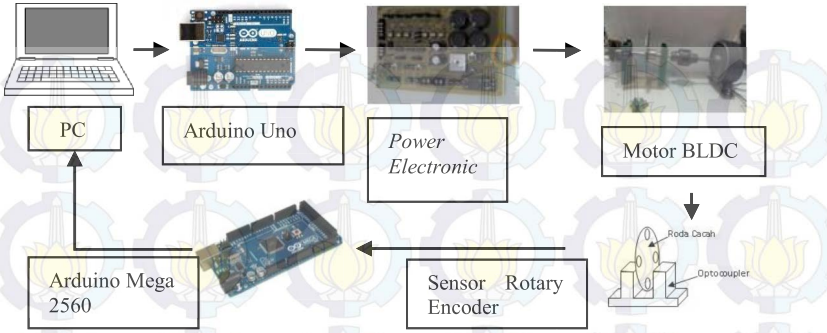
Pada bab ini menjelaskan tentang hasil simulasi dan pengujian dari perancangan sistem yang telah dilakukan dan dijelaskan pada bab sebelumnya. Bab ini terdiri dari pengujian kontroler yang telah diimplementasikan dengan menggunakan *software* MATLAB. Pengujian ini dilakukan untuk mengetahui respon keluaran dari sistem dan juga untuk mengetahui proses kerja dari kontroler yang didesain.

4.1 Gambaran Singkat Pengujian Sistem

Pengujian sistem *hardware* dilakukan terhadap respon kecepatan motor BLDC serta pengujian terhadap arduino sebagai pembangkit frekuensi fasa. Pengujian sistem *software* berupa hasil pengujian respon kontroler *fuzzy*-PI terhadap model *plant* motor BLDC dengan beban rem magnetic secara bervariasi. Kecepatan referensi motor BLDC bernilai konstan kemudian dilakukan pengujian terhadap kontroler *fuzzy*-PI dengan tujuan untuk mengetahui bahwa kontroler dapat mengatasi perubahan respon pada model *plant* motor BLDC karena perubahan parameter pada kondisi beban minimal, nominal dan maksimal.

4.2 Pengujian Kecepatan Motor BLDC

Pengujian ini dilakukan untuk mengetahui dan mengkalibrasi kecepatan hasil dari pembacaan sensor *rotary encoder* motor BLDC dengan kecepatan hasil pembacaan tachometer. Pengujian dilakukan dengan menghubungkan motor BLDC dengan sumber tiga fasa dari rangkaian *power electronic*. Piringan sensor *rotary encoder* dipasang pada *shaft* motor yang nantinya piringan akan dibaca oleh *optocoupler* tipe U. Pulsa hasil pembacaan *rotary encoder* dikirim ke arduino Mega 2560. Data yang didapat kemudian diolah dan hasilnya ditampilkan pada *display* yang terdapat pada PC. Tachometer digunakan untuk melihat perbandingan kecepatan dari piringan dengan hasil pengolah arduino. Mekanisme pengujian kecepatan BLDCM dapat dilihat pada Gambar 4.1 berikut :



Gambar 4.1 Mekanisme Pengujian Kecepatan Motor BLDC

Hasil pengujian untuk kecepatan motor BLDC dapat dilihat pada Tabel 4.1.

Tabel 4.1 Hasil Pengujian Kecepatan Motor BLDC

Frekuensi Fasa (Hz)	Kecepatan (Rpm)			
	Tachometer	Pengukuran 1	Pengukuran 2	Pengukuran 3
10	155,5	155	157	158
15	218,5	217	217	221
20	295	294	296	298
25	372,5	370	371,5	374
30	447	447	451	442
35	524,5	523	525	527
40	600	598	599	603
45	673	671,4	678	670

Tabel 4.1 Hasil Pengujian Kecepatan Motor BLDC (Lanjutan)

Frekuensi Fasa (Hz)	Kecepatan (Rpm)			
	Tachometer	Pengukuran 1	Pengukuran 2	Pengukuran 3
50	748.5	745	749	750
55	826	820	826	835

Untuk mencari presentase *error* rata-rata digunakan rumus sebagai berikut :

$$error = \frac{(pembacaan\ sensor - Pembacaan\ Tachometer)}{Pembacaan\ Tachometer}\ x\ 100\% \quad (4.1)$$

Sehingga didapatkan *error* rata-rata dari semua pembacaan sebesar 0.109%. Berdasarkan data yang didapatkan pada tabel, hasil pembacaan kecepatan motor dengan *tachometer* dan perhitungan memiliki selisih yang kecil. Hal tersebut menunjukkan bahwa pengujian untuk motor BLDC telah berhasil dan sesuai.

4.3 Pengujian Kontroler Arduino

Pengujian arduino menggunakan *software* Matlab R2014a. Sinyal uji berupa frekuensi fasa dengan rentang 10-55 Hz. Arduino akan membangkitkan 6x frekuensi fasa sesuai dengan nilai rentang yang diberikan oleh PC. Pengukuran frekuensi dari arduino diukur menggunakan osiloskop digital.

Tabel 4.2 Hasil Pengujian Pembangkit Frekuensi Fasa Arduino

Frekuensi Fasa (Hz)	Perhitungan 6x Frekuensi Fasa (Hz)	Pembangkit Frekuensi Arduino (Hz)	Error (%)
10	60	56	-6.66667
20	120	119	-0.83333

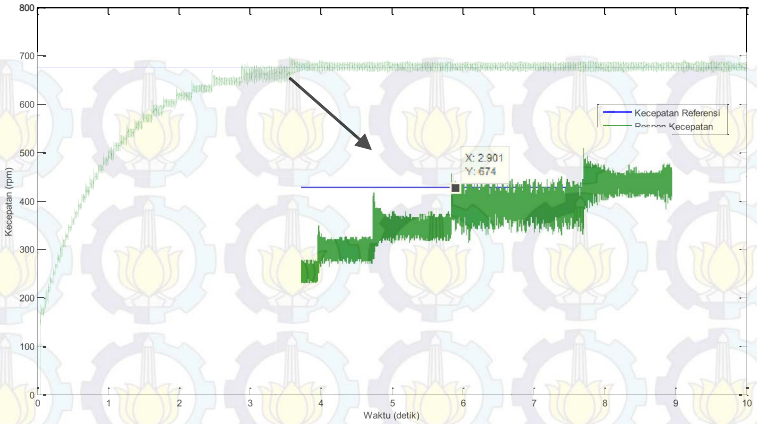
Tabel 4.2 Hasil Pengujian Pembangkit Frekuensi Fasa Arduino (Lanjutan)

Frekuensi Fasa (Hz)	Perhitungan 6x Frekuensi Fasa (Hz)	Pembangkit Frekuensi Arduino (Hz)	Error (%)
30	180	181	0.555556
40	240	241	0.416667
43	260	262	0.769231
50	300	301	0.333333
55	330	333	0.909091

Berdasarkan data yang didapatkan pada tabel, hasil pengujian pembangkit frekuensi fasa arduino dan perhitungan memiliki selisih yang kecil. Hal tersebut menunjukkan bahwa pengujian untuk kontroler arduino telah berhasil dan sesuai.

4.4 Simulasi Motor BLDC

Simulasi ini dilakukan untuk melihat respon kecepatan dari motor BLDC dengan kecepatan referensi konstan seperti pada Gambar 4.1. Dari grafik hasil respon kecepatan ini dapat diperoleh model matematik dari motor BLDC melalui identifikasi sistem yang telah dijelaskan pada bab sebelumnya.

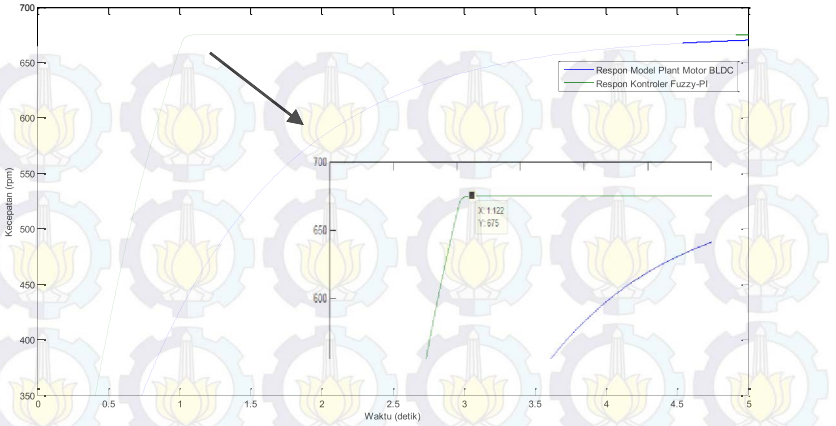


Gambar 4.2 Respon Kecepatan Motor BLDC

Dari hasil grafik diatas, dapat diketahui bahwa pada saat t = 2.901 detik respon kecepatan *plant* model baru mencapai nilai yang kurang lebih sama dengan kecepatan referensi (waktu *steady state*).

4.5 Simulasi dan Pengujian dengan Kontroler *Fuzzy*-PI

Pada subbab ini akan menunjukkan hasil simulasi model *plant* motor BLDC dengan kontroler *fuzzy*-PI pada *plant* motor BLDC yang telah dirancang. Nilai parameter kontroler *fuzzy*-PI diperoleh dari hasil *tunning* dengan nilai *gain integral error* (Kie) = 0.001, *gain error* (Ke) = 0.1, *control offset* (Uset) = 20 dan *gain control* (Ku) = 25.

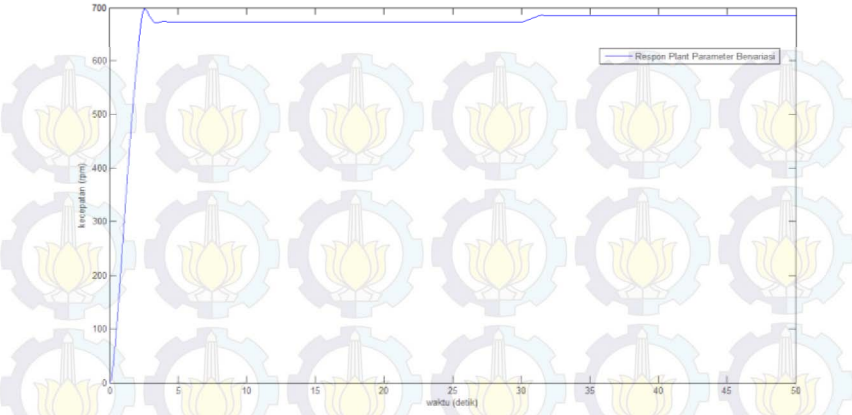


Gambar 4.3 Respon Kontroler *Fuzzy*-PI

Dengan spesifikasi respon sebagai berikut τ = 0.707 detik, $t_s(\pm 5\%) = 2.121\ detik$, $t_r\ (5\% - 95\%) = 2.081\ detik$, $t_d = 0.49\ detik$ dan e_{ss} = - 0.03%. Dalam simulasi tersebut, hasil yang didapatkan adalah kontroler dapat bekerja dengan baik mengikuti nilai referensi yang telah diberikan, pada saat t = 1.122 detik.

4.6 Simulasi *Plant* Parameter Bervariasi

Tahap ini dilakukan untuk mengetahui dan menguji proses kerja dari kontroler yang telah didesain. Pengujian dilakukan dengan Simulink yang terdapat dalam *software* Matlab. Pada Tugas Akhir ini dilakukan pengujian untuk kontroler *fuzzy*-PI. Metode pembebanan diberikan dalam selang waktu 50 detik. Hal ini untuk mempermudah melihat respon kontroler dengan baik.



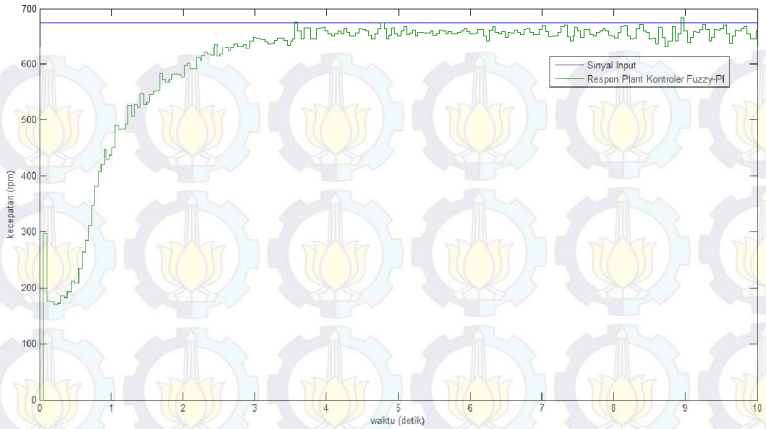
Gambar 4.4 Respon *Plant* Parameter Bervariasi

Saat simulasi untuk model *plant* parameter bervariasi, kontroler *fuzzy*-PI mengalami *overshoot* tetapi dapat kembali sesuai dengan nilai referensi yang telah diberikan sebelumnya.

4.7 Implementasi dan Pengujian dengan Kontroler *Fuzzy*-PI

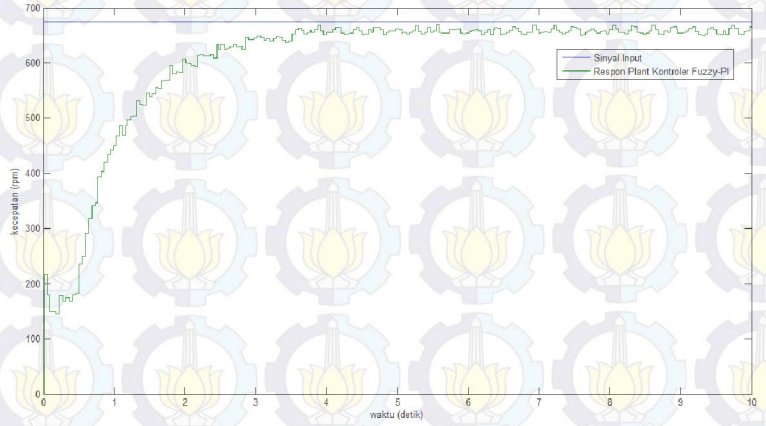
Pada implementasi kontroler *fuzzy*-PI menggunakan beban rem magnetik. Untuk kondisi beban minimal, piringan rotor tidak diberi beban rem magnetik. Untuk kondisi beban nominal, piringan rotor diberi beban rem magnetik sebesar ½ luas rem magnetik. Untuk kondisi beban maksimal, piringan rotor diberi beban magnetik sepenuhnya dari luas penampang rem magnetik.

Nilai parameter kontroler *fuzzy*-PI diperoleh dari hasil *tunning* dengan nilai *gain integral error* (Kie) = 0.001, *gain error* (Ke) = 0.1, *control offset* (Uset) = 12 dan *gain control* (Ku) = 25.



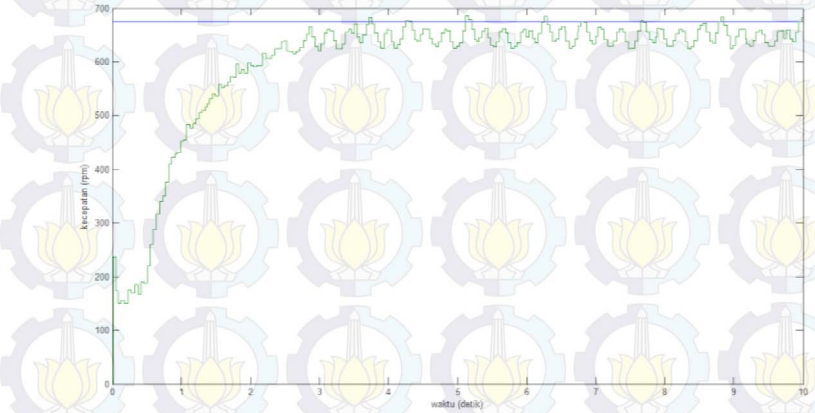
Gambar 4.5 Respon Kontroler *Fuzzy*-PI dengan Beban Minimal

Dari hasil respon tersebut terlihat bahwa kecepatan motor BLDC sedikit menurun dari kecepatan referensi tetapi tetap dapat mengikuti kecepatan referensi. Spesifikasi respon tersebut diantaranya $\tau = 2.248$ detik, $t_s(\pm 5\%) = 6.744\ detik$, $t_r\ (5\% - 95\%) = 6.619\ detik$, $t_d = 1.558\ detik$ dan $e_{ss} = 0.001\%$.



Gambar 4.6 Respon Kontroler *Fuzzy*-PI dengan Beban Nominal

Dari hasil respon tersebut terlihat bahwa kecepatan motor BLDC tetap dapat mengikuti kecepatan referensi. Spesifikasi respon tersebut diantaranya τ = 2.455 detik, $t_s(\pm 5\%) = 7.367\ detik$, $t_r\ (5\% - 95\%) = 7.228\ detik$, $t_d = 1.701\ detik$ dan e_{ss} = -0.08%.



Gambar 4.7 Respon Kontroler *Fuzzy*-PI dengan Beban Maksimal

Dari hasil respon tersebut terlihat bahwa kecepatan motor BLDC tetap dapat mengikuti kecepatan referensi meskipun kecepatan motor BLDC menurun. Spesifikasi respon tersebut diantaranya τ = 2.226 detik, $t_s(\pm 5\%) = 6.679\ detik$, $t_r\ (5\% - 95\%) = 6.554\ detik$, $t_d = 1.542\ detik$ dan e_{ss} = -0.09%.

Halaman ini sengaja dikosongkan

DAFTAR PUSTAKA

[1] N. Parhizkar, M. Shafiei, and M. Bahrami Kouhshahi " *Direct Torque Control of Brushless DC Motor Drives with Reduced Starting Current Using Fuzzy Logic Controller*", *IEEE Uncertainty Reasoning and Knowledge Engineering (URKE), 2011 International Conference on,* Vol. 1, No.2 August 2011.

[2] M. V. Ramesh, J. Amarnath, and S. Kamakshaiah "*Speed Control of Brushless DC Motor by Using Fuzzy Logic Pi Controller"*, *ARPN Journal of Engineering and Applied Sciences,* Vol.6, No.9, September 2011.

[3] Hidayat, Alfin. *Cascade Fuzzy Sliding Mode Control-PID untuk Pengaturan Posisi Pada Brushless DC Motor*. Thesis Elektro – ITS. 2012.

[4] Muhammad H. Rashid, Ph.D., *"POWER ELECTRONICS HANDBOOK DEVICES, CIRCUITS, AND APPLICATIONS Third Edition"*, University of West Florida, U.S.A, Ch. 34, 2011.

[5] Loe, Yohan. *Kontroler Motor BLDC Menggunakan Microchip*. Skripsi Sistem Komputer – Universitas Binus. 2014.

[6] Candra Wardianto, Ovi. *Kontrol Fuzzy Adaptif Gain Scheduling Untuk Pengaturan Motor Induksi 3 Fasa*. Tugas Akhir Elektro-ITS. 2013.

[7] Putri Suryaditya, Nindita. *Pengaturan Proses Face Miling pada Mesin Computer Numerikal Control (CNC) dengan Kontroler Fuzzy-PID*. Tugas Akhir Elektro-ITS. 2013.

[8] Fitria Fauzy, Rizky. *Desain Kontroler Pid Fuzzy Untuk Pengendalian Tekanan dan Level Oksigen Gas Buang Pada Boiler*. Tugas Akhir Elektro-ITS. 2012.

Halaman ini sengaja dikosongkan

BAB 5
PENUTUP

5.1 Kesimpulan

Dari hasil desain diperoleh bahwa dengan menggunakan kontroler *fuzzy*-PI pada pengaturan kecepatan motor BLDC, dapat diambil kesimpulan sebagai berikut :

a. Pada kondisi beban minimal, implementasi kontroler *fuzzy*-PI mampu mendekati nilai kecepatan referensi dengan nilai τ = 2.248 detik, $t_s(\pm 5\%) = 6.744\ detik$, $t_r\ (5\% - 95\%) = 6.619\ detik$, $t_d = 1.558\ detik$ dan e_{ss} = 0.001%.

b. Pada kondisi beban nominal, implementasi kontroler fuzzy-PI mampu mendekati nilai kecepatan referensi dengan nilai τ = 2.455 detik, $t_s(\pm 5\%) = 7.367\ detik$, $t_r\ (5\% - 95\%) = 7.228\ detik$, $t_d = 1.701\ detik$ dan e_{ss} = -0.08%.

c. Pada kondisi beban maksimal, implementasi kontroler *fuzzy*-PI mampu mendekati nilai kecepatan referensi dan hasil respon mendekati respon beban nominal dengan nilai τ = 2.226 detik, $t_s(\pm 5\%) = 6.679 detik$, $t_r\ (5\% - 95\%) = 6.554\ detik$, $t_d = 1.542\ detik$ dan e_{ss} = -0.09%.

5.2 Saran

Untuk pengembangan penelitian, penulis menyarankan untuk menggunakan metode kontroler lain yang dapat memberi respon lebih baik sehingga dapat mencapai nilai kecepatan referensi dengan lebih akurat pada respon keluaran motor BLDC.

Halaman ini sengaja dikosongkan

LAMPIRAN A

A.1 Program *Fuzzifikasi* untuk Kontroler *Fuzzy*-PI

```
function xf=fusi(x)
xf=[0 0 0 0 0]';
if x<-2
    xf(1)=1;
elseif x<-1
    xf(1)=-1-x;
    xf(2)=x-(-2);
elseif x<0
    xf(2)=0-x;
    xf(3)=x-(-1);
elseif x<1
    xf(3)=1-x;
    xf(4)=x-0;
elseif x<2
    xf(4)=2-x;
    xf(5)=x-1;
else
    xf(5)=1;
end
```

A.2 Program *Fuzzy Inference* untuk Kontroler *Fuzzy*-PI

```
function uf=infus(eder)
erf(1)=eder(1);
erf(2)=eder(2);
erf(3)=eder(3);
erf(4)=eder(4);
erf(5)=eder(5);

derf(1)=eder(6);
derf(2)=eder(7);
derf(3)=eder(8);
derf(4)=eder(9);
derf(5)=eder(10);
```

```
uf=[0 0 0 0 0]';

rbf=[1 1 2 2 3
     1 2 2 3 4
     2 2 3 4 4
     2 3 4 4 5
     3 4 4 5 5];
 for i=1:5
     for j=1:5
         k=rbf(i,j);
         % inference rule mamdani
          uf(k)=max( uf(k), min(erf(j),derf(i))
);
         %Larsent arithmatik rule
         % uf(k)=0.5*( uf(k) +
sqrt(erf(j)*derf(i)) );
     end
 end
```

A.3 Program *Defuzzifikasi* untuk Kontroler *Fuzzy*-PI

```
function x=defusi(xf)
atas=-2*xf(1)-1*xf(2)+0*xf(3)+1*xf(4)+2*xf(5);
bawah=xf(1)+xf(2)+xf(3)+xf(4)+xf(5);
x=atas/bawah;
```

LAMPIRAN B

B.1 Program Arduino Pembangkit Frekuensi Fasa dan PWM

```
int bacadata = 0;
//byte bacadata = 0;
int analogValue = 0;  // variable to hold the analog value
char disp2[4];
int pwm = 0;
//char disp3[1];

void setup() {
  // put your setup code here, to run once:
  TCCR1B = TCCR1B & B11111000 | B00000010;
  //pinMode(22, OUTPUT);
  pinMode(12, OUTPUT);    //pin frekuensi 74175
  Serial.begin(115200);
}

void loop() {
  // put your main code here, to run repeatedly:
  //analogValue = analogRead(1);
  //Serial.print('#');
  //sprintf(disp2, "%4d", analogValue);
  //Serial.print(disp2);
  //digitalWrite(22, LOW);
  //range frekuensi 10-100 Hz dengan skala pembacaan 60-600 Hz
  if (Serial.available() > 0) {
    bacadata = (Serial.read());
    //pembangkit frekuensi fasa
    digitalWrite(12, HIGH);
    delayMicroseconds(79000 / bacadata);
    //delayMicroseconds(74000 / bacadata);  //frekuensi 50% good
    digitalWrite(12, LOW);
    delayMicroseconds(79000 / bacadata);
    analogWrite(10, 204);                 //output pin pwm dengan duty cycle
80%
  }

  if (Serial.available() < 1) {
```

```
    analogWrite(10, 0);
  }
}
```

B.2 Program Arduino sebagai Pembaca Sensor Kecepatan

```
int duration = 0;
int rpm = 0;
char bufer[4];
int x;

void setup() {
  // put your setup code here, to run once:
  pinMode(52, INPUT); //encoder pin
  Serial.begin(115200);
  //Serial1.begin(115200);
  //delay(10);
}

void loop() {
  // put your main code here, to run repeatedly:
  //duration = pulseIn(52, HIGH, 100000);
  duration = pulseIn(52, HIGH, 100000);
  rpm = (5500000 / 3) / duration;
  if (rpm < 0) {
    rpm = 0;
  }
  if (rpm > 1000) {
    rpm = 0;
  }
  //delay(100);
  sprintf(bufer, "%4d", rpm);
  Serial.print(bufer);
  //delay(10);
}
```