KONTROL KECEPATAN PEMUTAR GERABAH DENGAN KONTROL LOGIKA *FUZZY* TAKAGI-SUGENO-KANG BERBASIS MIKROKONTROLER ATMEGA 8535

SKRIPSI

JURUSAN TEKNIK ELEKTRO

Diajukan Untuk Memenuhi Persyaratan
Memperoleh Gelar Sarjana Teknik

Disusun oleh:

RAHMA NUR AMALIA

NIM. 0810630084

KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN

UNIVERSITAS BRAWIJAYA

FAKULTAS TEKNIK

MALANG

2012

LEMBAR PERSETUJUAN

KONTROL KECEPATAN PEMUTAR GERABAH DENGAN KONTROL LOGIKA *FUZZY* TAKAGI-SUGENO-KANG BERBASIS MIKROKONTROLER ATMEGA 8535

SKRIPSI

KONSENTRASI TEKNIK KONTROL

Diajukan untuk memenuhi persyaratan
memperoleh gelar Sarjana Teknik

Disusun oleh:
RAHMA NUR AMALIA
NIM. 0810630084

Telah diperiksa dan disetujui oleh :

Dosen Pembimbing I	Dosen Pembimbing II
M.Aziz Muslim, ST., MT., Ph.D	**Erni Yudaningtyas, ST., MT.,Dr**
NIP. 19741203 200012 1 001	**NIP. 19650913 199002 2 001**

LEMBAR PENGESAHAN

KONTROL KECEPATAN PEMUTAR GERABAH DENGAN KONTROL LOGIKA *FUZZY* TAKAGI-SUGENO-KANG BERBASIS ATMEGA 8535

SKRIPSI

JURUSAN TEKNIK ELEKTRO

Diajukan Untuk Memenuhi Persyaratan
Memperoleh Gelar Sarjana Teknik

Disusun oleh:
RAHMA NUR AMALIA
NIM. 0810630084
Skripsi ini telah diuji dan dinyatakan lulus pada
Tanggal

DOSEN PENGUJI

Purwanto, Ir., MT	**Fitriana Suhartati, ST., MT**
NIP.19540424 198601 1 001	**NIP.19741017 199802 2 001**

Goegoes Dwi N, ST., MT
NIP.19711013 200604 1 001

Mengetahui
Ketua Jurusan Teknik Elektro

DR. Ir. Sholeh Hadi Pramono, MS.
NIP. 19580728 198701 1 001

KATA PENGANTAR

Puji syukur kehadirat Allah SWT atas berkat rahmat dan karunia-Nya dan shalawat serta salam penulis sampaikan kepada junjungan Nabi Muhammad SAW sehingga skripsi ini dapat diselesaikan. Dalam menyelesaikan skripsi ini penulis banyak memperoleh bantuan dari berbagai pihak, baik langsung maupun tidak langsung. Oleh karena itu, sudah selayaknya bila pada kesempatan ini penulis mengucapkan terima kasih.

Dalam penyusunan skripsi ini, ada beberapa pihak yang telah memberikan bantuan dan dukungan sehingga skripsiini selesai, maka sangat perlu sekali dalam kesempatan ini penulis menyampaikan ucapan terima kasih, kepada :

1. Bapak M.Aziz Muslim, ST., MT., Ph.D, selaku pembimbing kesatu, berkat beliau yang dengan tekun dan telaten penuh kesabaran telah membimbing sehingga skripsi ini selesai sesuai harapan semua pihak.
2. Ibu Erni Yudaningtyas, ST., MT.,Dr, selaku pembimbing kedua sekaligus Kepala Laboratorium Sistem Kontrol, berkat bimbingan dan arahan yang beliau berikan sehingga skripsi ini dapat terselesaikan.
3. DR. Ir. Sholeh Hadi Pramono, MS.selaku Ketua Jurusan Teknik Elektro, yang telah memberikan banyak masukan kepada penulis demi kelancaran skripsi ini.
4. Bapak dan Ibu dosen yang telah memberikan ilmu dan pengetahuan kepada penulis selama menempuh pendidikan di Teknik Elektro Universitas Brawijaya Malang.
5. Ibuk dan Abah yang telah banyak berkorban dan memberi motivasi dalam penyelesaian skripsi ini.
6. Kedua kakakku, Mas Sovie dan Mas Ibnu, serta Adek ku tercinta, Dek Farisa yang selalu memberi semangat dan keceriaan.
7. Mas Bima untuk ajaran, semangat, dan kesabarannya dalam membantu penulis demi kelancaran skripsi ini.
8. Teman-teman saya, Mas Aldo,Mas Firman, Cepol, Kharis,Anas, Wahyu, Maho, Alvita, Ceri, Riesta, Ninin, Rissa yang selalu membantu dalam pengerjaan skripsi ini.
9. Segenap anggota asisten Lab. Sistem Kontrol, Maho, Rio, Arif, Seif, dan adek-adek 2009, Darmo, Gojin, Salmi yang selalu memberi semangat dan dukungan untuk kelancaran skripsi ini.

10. Segenap anggota tim robot, Taufik, Bagus, Irfan dan dkk yang selalu ada untuk memberikan bantuan kepada penulis.

Akhirnya penulis senantiasa mengharapkan kritik dan saran dari berbagai pihak demi kesempurnaan skripsi ini. Semoga skripsi ini bermanfaat bagi semua pihak.

Malang, Agustus 2012

Penulis,

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

Abstrak

Kontrol logika fuzzy merupakan alternatif sistem kendali modern yang mudah karena tidak perlu dicari model matematis dari suatu sistem, tetapi tetap efektif karena memiliki respon sistem yang stabil. Alat pembuat gerabah yang direncanakan diputar oleh Motor arus searah atau direct current (DC) memang merupakan hal baru bagi para pengrajin, dikarenakan mereka terbiasa menggunakan pemutar manual, yaitu dengan bantuan kaki, namun dalam skripsi ini, penulis menemukan solusi untuk para pemula yang ingin belajar membuat gerabah, yaitu menggunakan alat ini, mereka bisa dimudahkan dalam belajar membuat gerabah, dikarenakan untuk pemula cukup sulit untuk menggunakan pemutar manual. Alat pembuat gerabahini dikontrol oleh mikrokontroler ATMEGA 8535 dan metode pengaturan yang digunakan adalah kontrol logika fuzzy Takagi-Sugeno-Kang.

Logika fuzzy yang dirancang memiliki 2 input (Err kecepatan&ΔErr kecepatan) dan 1 output(PWM). Masing-masing membership function memiliki 5 label. Disini digunakan 25 fuzzy if-then rule. Sedangkan proses kontrol logika fuzzy terdiri dari fuzzifikasi, evaluasi rule dan yang terakhir defuzzifikasi dengan metode weighted average. Penggerak motor (driver motor) menggunakan sistem Pulse Width Modulation (PWM). Input setting point dilakukan dengan menekan tombol keypad yang berupa data digital, yang kemudian dikonversi ke tegangan olehmikrokontroler ATMEGA 8535. Keluaran dari mikrokontroller berupa Pulse Width Modulation (PWM) menjadi masukan untuk penggerak motor (driver motor), dimana kecepatan keluaran dari motor akan dideteksi oleh sensor rotary encoder. Setelah itu akan terjadi pengulangan proses, sampai kecepatan memenuhi nilai set point.

Pengujian respon sistem dilakukan terhadap variasi setting point, variasi beban dan variasi rule. Dari data-data yang diperoleh menunjukkan bahwa respon sistem cukup baik dalam mengejar nilai setting point dalam berbagai variasi yaitu setting point, beban dan rule. Dari penelitian ini, didapatkan respon sistem untuk setpoint 33 Rpm nilai Ess sebesar 2,27% - 3,39%, nilai T_s ketika beban 1 kg sebesar 85. Respon sistem untuk setpoint 52 Rpm dengan nilai Ess sebesar 0,96% - 4,55%, T_s ketika beban 1 kg sebesar 34. Respon sistem untuk setpoint 65 Rpm nilai Ess sebesar 0,96% - 3,84%, T_s ketika beban 1 kg sebesar 55. Respon sistem untuk setpoint 82 Rpm nilai Ess sebesar 0,6% - 1,67%, T_s ketika beban 1 kg sebesar 40s. Respon sistem untuk setpoint 95 Rpm nilai Ess sebesar 1,31% , T_s ketika beban 1 kg sebesar 49s.

Kata- Kata Kunci: Fuzzy Takagi- Sugeno- Kang, Mikrokontroler ATMEGA 8535, Motor arus searah atau direct current (DC)

BABI

PENDAHULUAN

1.1 LatarBelakang

Telah menjadi fenomena bahwakebutuhan manusia akan hal- hal yang bersifat dekoratif sangat banyak digemari, baik di kalangan menengah sampai menengah ke atas. Salah satu contohnya adalah dekoratif interior rumah yang kini mulai menjadi perhatian.Gerabah adalah salah satu jenis hiasan yang digemari penyuka desain interior. Bermacam bentuk bisa dihasilkan dari gerabah, mulai dari pot, patung, guci hingga pajangan–pajangan kecil.

Sampai saat ini penggunaan tenaga manusia sebagai sumber gerak merupakan hal lumrah yang dilakukan oleh para pengrajin untuk membuat gerabah. Selain itu jika dilihat dari tuntutan usaha untuk menghasilkan produk dalam jumlah yang banyak untuk memenuhi pesanan, para pengrajin masih nyaman menggunakan mesin manual. Tetapi untuk para pemula atau untuk anak- anak yang ingin belajar membuat gerabah, mereka sulit untuk menggunakan mesin manual, yang menggunakan kaki sebagai pemutar. Maka dari itu, penulis mencoba untuk melakukan analisis dan membuat terobosan baru tentang mesin pemutar gerabah yang nantinya diharapkan akan berguna untuk mempermudah proses pembelajaran pembuatan gerabah untuk pemula dan anak- anak.

Berdasarkan pada skripsi sebelumnya (Khasbullah, 2011) yang mencoba untuk memecahkan permasalahan tersebut yaitu dengan menggunakan kontroler Proporsional Integral Diferensial (PID), tetapi respon yang dicapai masih lambat.

Dalam skripsi ini, digunakan kontrol logika *Fuzzy,* karena kontrol logika *Fuzzy* merupakan alternatif sistem kendali modern yang mudah, yaitu tidak perlu mencari model matematis dari suatu sistem, tetapi tetap efektif karena memiliki respon sistem yang stabil. Kontrol logika *Fuzzy*juga merupakan salah satu sistem kontrol yang *redundant* atau *fault tolerant* yang artinya masih dapat bekerja dengan adanya pengurangan beberapa rule, maupun jika terjadi kesalahan-kesalahan kecil dalam pemrogramannya (Resmana, 1999).

Perlu diketahui bahwa kelebihan dari kontrol logika *fuzzy* Takagi- Sugeno-Kang dibandingkan dengan menggunakankontrol logika *fuzzy* metode Mamdani adalah dapat menghemat memori mikrokontroler dan lebih cepat di proses oleh mikrokontroler, dikarenakan fungsi keanggotaan keluarannya berupa konstanta atau fungsi linier,

sehingga kontrol logika *fuzzy* Takagi- Sugeno-Kang lebih ringkas dan efisien dalam hal komputasi.

1.2 Permasalahan

Mengacu pada permasalahan yang diuraikan pada latar belakang, maka rumusan masalah dapat ditekankan pada bagaimana kontrol logika *fuzzy* Takagi-Sugeno-Kang digunakan sebagai pengontrol kecepatan pada alat pemutar gerbabah dengan menggunakan mikrokontroler ATMEGA 8535.

1.3 PembatasanMasalah

Untuk menekankan pada objek pembahasan yang ada maka penelitian ini diberikan batasan masalah sebagai berikut:

1. Mekanika alat putar gerabahdi penelitian ini tidak dibahas secara mendalam.
2. Pembahasan ditekankan pada proses pengendalian kecepatan motor yang ditentukan, kinerja driver dan rangkaian elektronika tidak dibahas mendalam.

1.4 Tujuan

Menciptakan sistem yang dapat fleksibel, dimana kecepatan putaran motor arus searah atau *direct current* (DC) sebagai penggerak alat putar gerabah saat terjadi gangguan berupa perubahan berat beban material, kecepatannya dapat diatur dan tetap konstan. Kinerja dari alat pemutar gerabah ini diharapkan dapat maksimal.

1.5 SistematikaPembahasan

Penulisanlaporantugasakhirinidibagimenjadibeberapabab,diantaranya sebagaiberikut:

BAB I Pendahuluan

Memuat latar belakang, rumusan masalah, tujuan, batasan masalah, dan sistematika pembahasan.

BAB II Teori Penunjang

Membahas teori-teori yang mendukung dalam perencanaan dan pembuatan alat.

BAB III Metodologi

Berisi tentang metode penelitian dan perencanaan alat serta pengujian.

BAB IV Perencanaan dan Pembuatan Alat

Membahas analisa spesifikasi perancangan dan perealisasian alat.

BAB V Pengujian Alat

Memuat hasil pengujian terhadap alat yang telah direalisasikan.

BAB VI Kesimpulan dan Saran

Memuat kesimpulan dan saran-saran.

BAB II
TINJAUAN PUSTAKA

Untuk memudahkan dalam memahami cara kerja rangkaian maupun dasar-dasar perencanaan dari sistem yang dibuat, maka perlu adanya penjelasan dan uraian mengenai teori penunjang yang digunakan dalam penulisan skripsi ini. Teori penunjang yang akan dijelaskan dalam bab ini adalah:

- Pengertian Gerabah
- Motor arus searah atau *direct current* (DC)
- Mikrokontroler ATMEGA 8535
- Kontrol Logika *Fuzzy* Takagi-Sugeno-Kang
- Sensor *Rotary Encoder*
- *Pulse Width Modulation*(PWM)

2.1 Teori Dasar Gerabah

2.1.1 Pengertian Gerabah

Gerabah adalah bagian dari keramik yang dilihat berdasarkan tingkat kualitas bahannya. Namun masyarakat ada mengartikan terpisah antara gerabah dan keramik. Ada pendapat gerabah bukan termasuk keramik, karena benda-benda keramik adalah benda-benda pecah belah permukaannya halus dan mengkilap seperti porselin dalam wujud vas bunga, guci, tegel lantai dan lain-lain. Sedangkan gerabah adalah barang-barang dari tanah liat dalam wujud seperti periuk, belanga, tempat air, dll. Untuk memperjelas hal tersebut dapat ditinjau dari beberapa sumber berikut ini.

Menurut The Concise Colombia Encyclopedia, Copyright ã 1995, kata ‘keramik’ berasal dari Bahasa Yunani (Greek) ‘keramikos’ menunjuk pada pengertian gerabah; ‘keramos’ menunjuk pada pengertian tanah liat . ‘Keramikos’ terbuat dari mineral non metal, yaitu tanah lihat yang dibentuk, kemudian secara permanen menjadi keras setelah melalui proses pembakaran pada suhu tinggi. Usia keramik tertua dikenal dari zaman Paleolitikum 27.000 tahun lalu. Sedangkan menurut Malcolm G. McLaren dalam Encyclopedia Americana 1996 disebutkan keramik adalah suatu istilah yang sejak semula diterapkan pada karya yang terbuat dari tanah liat alami dan telah melalui perlakukan pemanasan pada suhu tinggi.

Beberapa teori lain tentang ditemukannya keramik pertama kali, salah satunya terkenal dengan 'teori keranjang'. Teori ini menyebutkan pada zaman prasejarah, keranjang anyaman digunakan orang untuk menyimpan bahan makanan. Agar tak bocor keranjang tersebut dilapisi dengan tanah liat di bagian dalamnya. Setelah tak terpakai keranjang dibuang keperapian. Kemudian keranjang itu musnah tetapi tanah liatnya yang berbentuk wadah itu ternyata menjadi keras. Teori ini dihubungkan dengan ditemukannya keramik prasejarah, bentuk dan motif hiasnya di bagian luar berupa relief cap tangan keranjang (Rosyidi, 2011)

Dari teori keranjang dan teori lainnya dapat dimengerti bahwa benda-benda keras dari tanah liat dari awal ditemukan sudah dinamakan benda keramik, walaupun sifatnya masih sangat sederhana seperti halnya gerabah dewasa ini. Pengertian ini menunjukkan bahwa gerabah adalah salah satu bagian dari benda-benda keramik.

2.1.2 Proses Pembuatan Gerabah

Proses pembuatan gerabah yang dilakukan pengrajin gerabah di Indonesia pada umumnya masih dilakukan secara tradisional, berikut merupakan proses pembuatan gerabah:

1. Pengambilan tanah liat

   Tanah liat diambil dengan cara menggali secara langsung ke dalam tanah yang mengandung banyak tanah liat yang baik. Tanah liat yang baik berwarna merah coklat atau putih kecoklatan. Tanah liat yang telah digali kemudian dikumpulkan pada suatu tempat untuk proses selanjutnya.

2. Persiapan tanah liat

   Tanah liat yang telah terkumpul disiram air hingga basah merata kemudian didiamkan selama satu hingga dua hari. Setelah itu, kemudian tanah liat digiling agar lebih rekat dan liat. Ada dua cara penggilingan yaitu secara manual dan mekanis. Penggilingan manual dilakukan dengan cara menginjak-injak tanah liat hingga menjadi ulet dan halus. Sedangkan secara mekanis dengan menggunakan mesin giling. Hasil terbaik akan dihasilkan dengan menggunakan proses giling manual.

3. Proses pembentukan

   Setelah melewati proses penggilingan, maka tanah liat siap dibentuk sesuai dengan keinginan. Aneka bentuk dan desain dapat dihasilkan dari tanah liat. Seberapa banyak tanah liat dan berapa lama waktu yang diperlukan tergantung

pada seberapa besar gerabah yang akan dihasilkan, bentuk dan desainnya. Perajin gerabah akan menggunakan untuk kedua tangan membentuk tanah liat dan kedua kaki untuk memutar alat pemutar. Kesamaan gerak dan konsentrasi sangat diperlukan untuk dapat melakukannya. Alat-alat yang digunakan yaitu alat pemutar, alat pemukul, batu bulat, kain kecil. Air juga sangat diperlukan untuk membentuk gerabah dengan baik.

4. Penjemuran

   Setelah bentuk akhir telah terbentuk, maka diteruskan dengan penjemuran. Sebelum dijemur di bawah terik matahari, gerabah yang sudah agak mengeras dihaluskan dengan air dan kain kecil lalu dibatik dengan batu api. Setelah itu baru dijemur hingga benar-benar kering. Lamanya waktu penjemuran disesuaikan dengan cuaca dan panas matahari.

5. Pembakaran

   Setalah gerabah menjadi keras dan benar-benar kering, kemudian banyak gerabah dikumpulkan dalam suatu tempat atau tungku pembakaran. Gerabah-gerabah tersebut kemudian dibakar selama beberapa jam hingga benar-benar keras. Proses ini dilakukan agar gerabah benar-benar keras dan tidak mudah pecah. Bahan bakar yang digunakan untuk proses pembakaran adalah jerami kering, daun kelapa kering ataupun kayu bakar.

6. Penyempurnaan.

   Dalam proses penyempurnaan, gerabah jadi dapat dicat dengan cat khusus atau diglasir sehingga terlihat indah dan menarik sehingga bernilai jual tinggi.

2.2 Teori Dasar Mikrokontroler ATMEGA 8535

2.2.1 Pengenalan Mikrokontroler ATMEGA 8535

Mikrokontroler AVR memiliki arsitektur RISC 8 Bit, sehingga semua instruksi dikemas dalam kode 16-bit (16-bits word) dan sebagian besar instruksi dieksekusi dalam satu siklus instruksi clock. Dan ini sangat membedakan sekali dengan instruksi MCS-51 (Berarsitektur CISC) yang membutuhkan siklus 12 clock.

Adapun kelebihan dari mikrontroler adalah sebagai berikut:

1. Penggerak pada mikrontroler menggunakan bahasa pemrograman assembly dengan berpatokan pada kaidah digital dasar sehingga pengoperasian sistem menjadi sangat mudah dikerjakan sesuai dengan logika sistem (bahasa *assembly* ini mudah dimengerti karena menggunakan aplikasi dimana parameter input dan

output langsung bisa diakses tanpa menggunakan banyak perintah). Desain bahasa *assembly* ini tidak menggunakan begitu banyak syarat penulisan bahasa pemrograman seperti huruf besar dan huruf kecil untuk bahasa *assembly*tetap diwajarkan.

2. Mikrokontroler tersusun dalam satu *chip* dimana prosesor, memori, dan I/O terintegrasi menjadi satu kesatuan kontrol sistem sehingga mikrokontroler dapat dikatakan sebagai komputer mini yang dapat bekerja secara inovatif sesuai dengan kebutuhan sistem.
3. Sistem *running* bersifat berdiri sendiri tanpa tergantung dengan komputer sedangkan parameter komputer hanya digunakan untuk *download* perintah instruksi atau program. Langkah-langkah untuk *download* komputer dengan mikrokontroler sangat mudah digunakan karena tidak menggunakan banyak perintah.
4. Pada mikrokontroler tersedia fasilitas tambahan untuk pengembangan memori dan I/O yang disesuaikan dengan kebutuhan sistem.
5. Harga untuk memperoleh alat ini lebih murah dan mudah didapat.

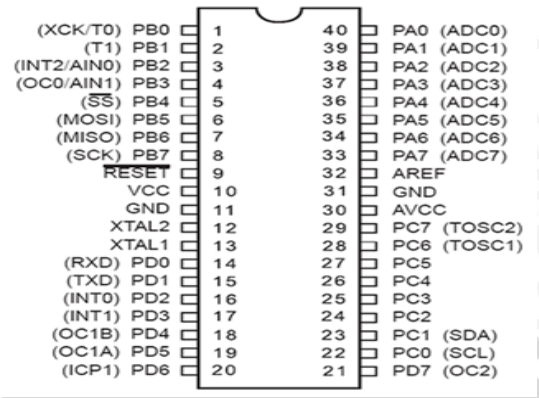
2.2.2 Fitur ATMEGA 8535

Fitur- fitur dalam mikrokontroler ATMEGA 8535 meliputi:

- Sistem *processor* 8 bit berbasis *RISC* dengan kecepatan maksimal 16 MHz.
- Ukuran memori *flash* 8KB, *SRAM* sebesar 512 byte, *EEPROM* sebesar 512 byte.
- ADC internal dengan resolusi 10 bit sebanyak 8 channel
- Port komunikasi serial USART dengan kecepatan maksimal 2.5 Mbps
- Mode *Sleep* untuk penghematan penggunaan daya listrik

2.2.3 Konfigurasi Pin ATMEGA 8535

Konfigurasi Pin mikrokontroler ATMEGA 8535 akan ditampilkan dalam gambar 2.1.

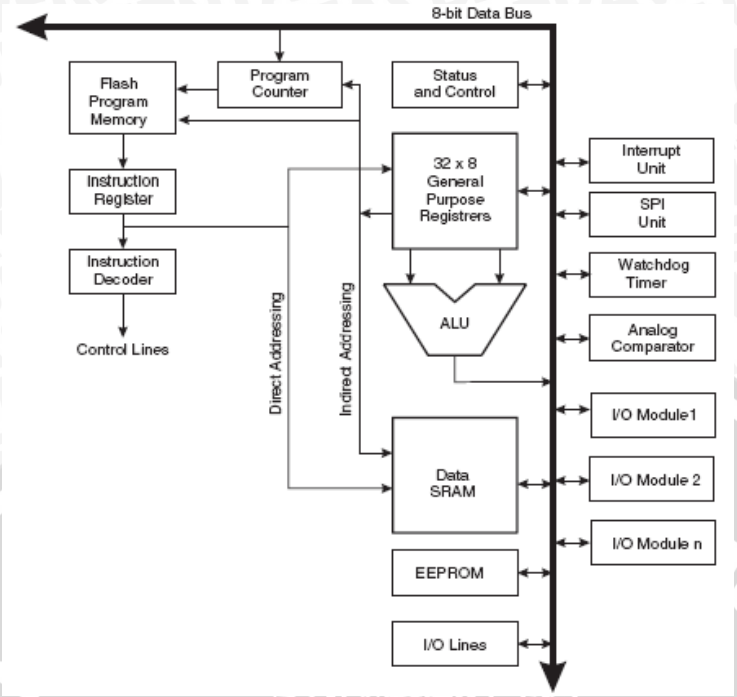


Gambar 2.1 Konfigurasi Pin ATMEGA 8535

Sumber: Irwansetyo, 2010

- VCC merupakan Pin yang berfungsi sebagai pin masukan catudaya
- GND merupakan Pin *Ground*
- Port A (PA0...PA7) merupakan pin I/O dan pin masukan ADC
- Port B (PB0...PB7) merupakan pin I/O dan pin yang mempunyai fungsi khusus yaitu Timer/Counter, komparator Analog dan SPI
- Port C (PC0...PC7) merupakan port I/O dan pin yang mempunyai fungsi khusus, yaitu komparator analog dan Timer Oscillator
- Port D (PD0...PD1) merupakan port I/O dan pin fungsi khusus yaitu komparator analog dan interrupt eksternal serta komunikasi serial
- RESET merupakan pin yang digunakan untuk mereset mikrokontroler
- XTAL1 dan XTAL2 merupakan pin masukan *clock* eksternal
- AVCC merupakan pin masukan untuk tegangan ADC
- AREF merupakan pin masukan tegangan referensi untuk ADC

AVR menjalankan sebuah instruksi tunggal dalam satu siklus dan memiliki struktur I/O yang cukup lengkap sehingga penggunaan komponen eksternal dapat dikurangi. Mikrokontroler AVR didesain menggunakan arsitektur *Harvard*, di mana ruang dan jalur bus bagi memori program dipisahkan dengan memori data. Berikut akan ditampilkan blok diagram Internal ATMEGA 8535, seperti dalam Gambar 2.2.

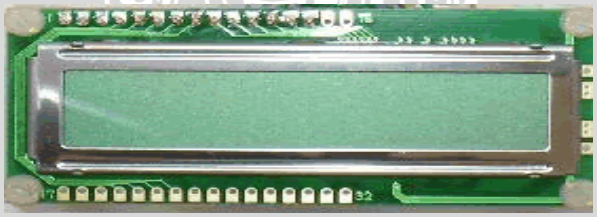


Gambar 2.2 Blok Diagram Internal ATMega 8535

Sumber: Irwansetyo, 2010

2.3 Liquid Crystal Display (LCD)

LCD (*Liquid cristal display*) adalah salah satu komponen elektronika yang berfungsi sebagai tampilan suatu data, baik karakter, huruf ataupun grafik. Jenis LCD yang dipakai pada alat ini adalah LCD M1632, seperti yang tertera dalam Gambar 2.3. LCD terdiri dari dua bagian, yang pertama merupakan panel LCD sebagai media penampil informasi dalam bentuk huruf/ angka dua baris, masing – masing baris bisa menampung 16 huruf/ angka.



Gambar 2.3 LCD Character 2x16

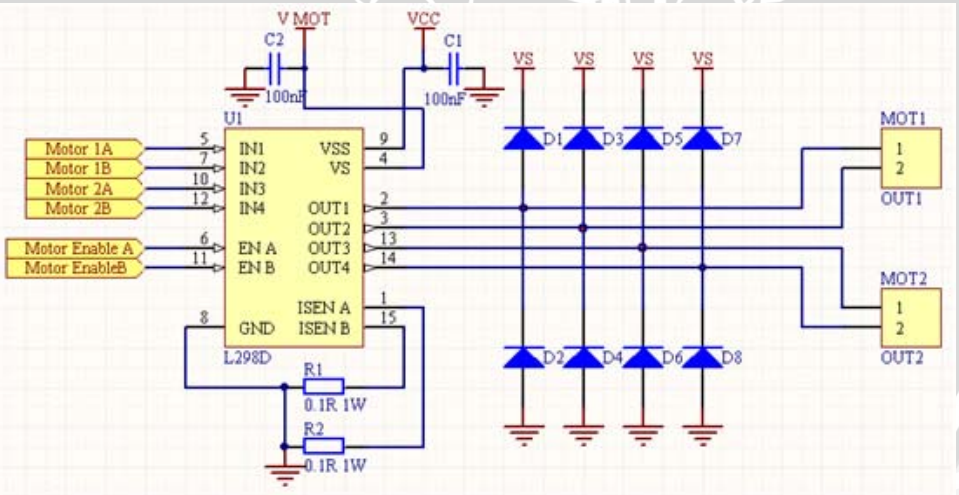
Sumber: Irwansetyo, 2010

Bagian kedua merupakan sebuah sistem yang dibentuk dengan mikrokontroler yang ditempel dibalik pada panel LCD, berfungsi mengatur tampilan LCD. Dengan demikian pemakaian LCD M1632 menjadi sederhana, sistem lain cukup mengirimkan kode – kode ASCII dari informasi yang ditampilkan. Berikut merupakan spesifikasi LCD M1632 **:**

1. Tampilan 16 karakter 2 baris dengan matrik 5 x 7 + kursor.
2. ROM pembangkit karakter 192 jenis.
3. RAM pembangkit karakter 8 jenis (diprogram pemakai).
4. RAM data tampilan 80 x 8 bit (8 karakter).
5. Duty ratio 1/16.
6. RAM data tampilan dan RAM pembangkit karakter dapat dibaca dari unit mikroprosesor.
7. Beberapa fungsi perintah antara lain adalah penghapusan tampilan (*display clear*), posisi kursor awal (*crusor home*), tampilan karakter kedip (*display character blink*), penggeseran kursor (*crusor shift*) dan penggeseran tampilan (*display shift*).
8. Rangkaian otomatis reset saat daya dinyalakan.
9. Catu daya tunggal +5 volt.

2.4 Driver Motor

Driver motor adalah rangkaian listrik yang dapat menjalankan motor dengan menggunakan sinyal kontrol sederhana. Motor membutuhkan aliran arus yang tinggi atau tegangan yang tinggi. Maka dari itu, kita membutuhkan rangkaian listrik driver motor yang merupakan jembatan antara mikrokontroler dan motor. Dalam Gambar 2.4 akan ditampilkan gambar rangkaian driver motor.

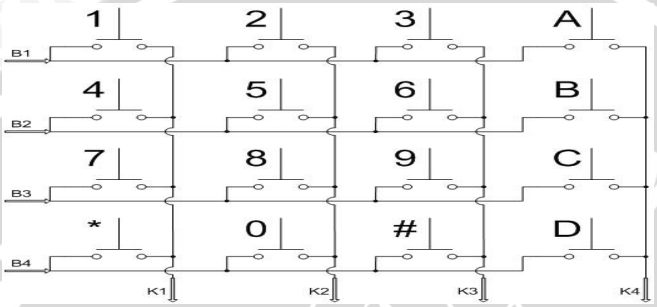


Gambar 2.4 Rangkaian Driver Motor

Sumber: Irwansetyo, 2010

2.5 *Keypad*

Keypad Matriks adalah tombol-tombol yang disusun secara maktriks (baris x kolom) sehingga dapat mengurangi penggunaan pin input. Sebagai contoh seperti yang terdapat dalam Gambar 2.5 , yang terdapat pada gambar *Keypad* Matriks 4×4 cukup menggunakan 8 pin untuk 16 tombol. Hal tersebut dimungkinkan karena rangkaian tombol disusun secara horizontal membentuk baris dan secara vertikal membentuk kolom:



Gambar 2.5 Konstruksi *Keypad* 4 x 4

Sumber: Rosyidi, 2011

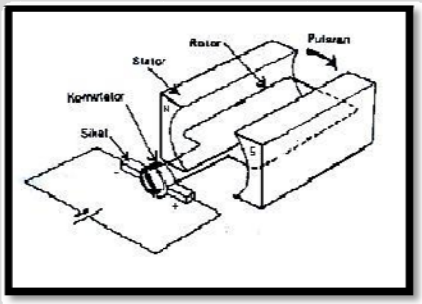
Seperti terlihat dalam Gambar 2.5, apabila saklar '1' ditekan, maka baris 1 dan kolom 1 langsung terhubung ke *ground*. Apabila saklar '2' ditekan, maka baris 1 dan kolom 2 langsung terhubung ke *ground* dan seterusnya.

2.6 Motor DC

Motor DC yang digunakan dalam penelitian adalah motor DC yang menggunakan permanen magnet. Alasan pemilihan motor DC tipe ini adalah kemudahan dalam pengontrolan dengan menggunakan pengaturan tegangan DC. Medan stator motor jenis ini dihasilkan oleh magnet permanen bukan elektromagnet. motor DC mempunyai kurva kecepatan torsi yang linier dalam jangka yang lebar. Penggunaan magnet permanen tidak membutuhkan daya listrik untuk menghasilkan medan stator, sehingga daya dan pendinginan yang diperlukan lebih rendah dibandingkan motor yang menggunakan elektromagnet. Perubahan kecepatan motor dapat diatur dengan cara mengubah-ubah besarnya tegangan DC yang diberikan. Motor DC memiliki beberapa keunggulan, yaitu:

- Bentuknya kompak, ringan dan berdaya kerja tinggi
- Dapat bekerja pada daerah atau tempat yang mempunyai lingkungan ekstrem
- Biaya perawatan mudah

Motor DC hampir sama konstruksinya dengan motor AC, perbedaannya terletak pada sikat dan cincin belah (komutator). Saat siklus pertama, arus mengalir dari kutub positif ke negatif. Aliran arus yang melewati bagian kabel yang berada didekat kutub N magnet akan menimbulkan gaya *Lorentz* ke bawah. Sementara itu aliran arus yang melewati kabel yang berada di dekat kutub S magnet akan menyebabkan gaya Lorentz ke atas. Kedua perpaduan gaya Lorentz tersebut akan menyebabkan kawat berputar. Pada siklus berikutnya terjadi hal yang serupa seperti pada siklus sebelumnya. Apabila arus terus-menerus dialirkan, maka kawat akan berputar secara terus menerus pula. Pada aplikasi sesungguhnya, kawat adalah sebuah rotor yang akan dikopel dengan sebuah as dan akan memutar as tersebut terus menerus seiring perputaran motor. Motor DC ditunjukkan dalam Gambar 2.6.



Gambar 2.6 Motor DC

Sumber: Rosyidi, 2011

2.7 Sensor *Rotary Encoder*

Rotary encoder atau yang dikenal dengan*shaft encoder*adalah perangkat elektronika yang digunakan untuk mengkonversi sudut dari perputaran poros atau roda ke dalam kode digital. Komponen ini biasa digunakan dalam bidang robotika, perangkat komputer dan perangkat elektronik lainnya. *Rotary encoder* dibedakan menjadi dua jenis yakni *rotary encoder* absolut dan relatif.

Rotary encoder absolut merupakan jenis sensor yang mampu menghasilkan kode digital yang unik untuk masing-masing sudut poros. Pada prinsipnya sensor ini tersusun atas plat baja dan kontak. Plat baja dipotong dan disusun dengan pola tertentu kemudian ditempelkan pada suatu poros. Plat baja dan kontak ini secara prinsip kerja menyerupai saklar dalam kondisi ON apabila keduanya saling bersentuhan dan OFF apabila terpisah. Plat baja dan kontak diatur sedemikian rupa sehingga menghasilkan kondisi yang berbeda untuk tiap-tiap sudut poros. Bentuk fisik dari *rotary encoder* absolut ditunjukkan dalam Gambar 2.7.

Gambar 2.7 *Rotary Encoder* Absolut

Sumber : Rosyidi, 2011

Rotary encoder relatif tidak dapat mengukur posisi sudut poros melainkan hanya mengukur perubahan sudut poros terhadap posisi sudut sebelumnya. Rotary ini digunakan ketika metode *rotary* absolut tidak dapat digunakan. Secara prinsip, sistem ini terdiri dari piringan yang dipasang pada poros dan sensor optik. Sistem ini menggunakan metode saklar optik, missal *photodiode*, untuk menghasilkan pulsa listrik yang digunakan sebagai masukan bagi rangkaian kontrol elektronika. Dalam Gambar 2.8 di bawah ini menunjukkan bentuk fisik *rotary encoder* relatif.

Gambar 2.8 *Rotary Encoder* Relatif

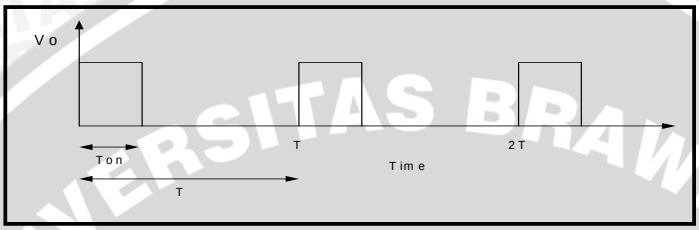
Sumber: Rosyidi, 2011

Salah satu kekurangan *Rotary encoder* relatif adalah tidak dapat menentukan arah putaran poros. Agar dapat mengetahui arah putaran poros, pada sistem harus ditambahkan dua buah sensor optik yang dipasang pada sudut berbeda. Pembacaan arah putaran dan jumlah putaran dilakukan dengan membaca dan membandingkan sinyal keluaran dari kedua buah sensor. Tipe *rotary encoder* ini dikenal dengan *quadrature encoder.*

2.8 *Pulse Width Modulation* (PWM)

Pulse Width Modulation (PWM)digunakan untuk mengatur kecepatan dari motor DC. Dimana kecepatan motor DC tergantung pada besarnya *duty cycle* yang diberikan pada motor DC tersebut.

Pada sinyal PWM, frekuensi sinyal konstan sedangkan *duty cycle* bervariasi dari 0%-100%. Dengan mengatur *duty cycle* akan diperoleh keluaran yang diinginkan. Pada AVR ATMEGA 8535 *duty cycle* ditentukan dari *Output Compare Register 0* (OCR0). Sinyal PWM secara umum dapat dilihat dalam Gambar 2.9.



Gambar 2.9 Sinyal PWM Secara Umum

Sumber: Irwansetyo, 2010

$$Duty\ \ cycle = \frac{T_{on}}{T} x100\% \tag{2.1}$$

$$V_{DC} = Duty\ cycle\ x\ V_{CC} \tag{2.2}$$

Sedangkan frekuensi sinyal dapat ditentukan dengan rumus sebagai berikut:

$$fOCn = \frac{fclk\_I/O}{N.256} \tag{2.3}$$

Timer/counter yang digunakan pada PWM ini yaitu *Timer/Counter0* (8-bit) dengan mode Fast PWM dan *Prescalers factor* (N) yaitu 256.

2.9 Dasar Kontrol Logika *Fuzzy*

2.9.1 Pengertian Logika *Fuzzy*

Sistem *fuzzy* atau logika *fuzzy* adalah salah satu bahasan *soft computing* yang memiliki karakteristik dan keunggulan dalam menangani permasalahan yang bersifat ketidakpastian dan kebenaran parsial. Logika *fuzzy* merupakan pengembangan dari logika boolean yang hanya memiliki nilai *true* (1) atau *false* (0).

2.9.2 Keuntungan menggunakan Logika *Fuzzy*

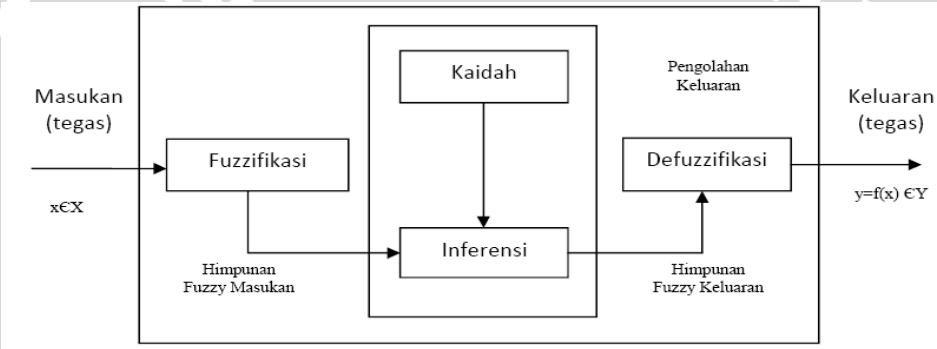
Ada beberapa alasan mengapa menggunakan logika *fuzzy*, diantaranya:

1. Konsep logika *fuzzy* mudah di mengerti. Konsep matematis yang mendasaripenalaran *fuzzy* sangat sederhana dan mudah dimengerti.
2. Logika *fuzzy* sangat fleksibel.

3. Logika *fuzzy* memiliki toleransi terhadap data-data yang tidak tepat.
4. Logika *fuzzy* mampu memodelkan fungsi-fungsi nonlinier yang sangat kompleks.
5. Logika *fuzzy* dapat membangun dan mengaplikasikan pengalaman-pengalaman para pakar secara langsung tanpa harus melalui proses pelatihan.
6. Logika *fuzzy* dapat bekerjasama dengan teknik-teknik kendali secara konvensional.
7. Logika *fuzzy* didasarkan pada bahasa alami.

2.9.3 Sistem Logika *Fuzzy*

Secara umum sistem logika *fuzzy* dapat digambarkan seperti dalam Gambar 2.10 berikut.

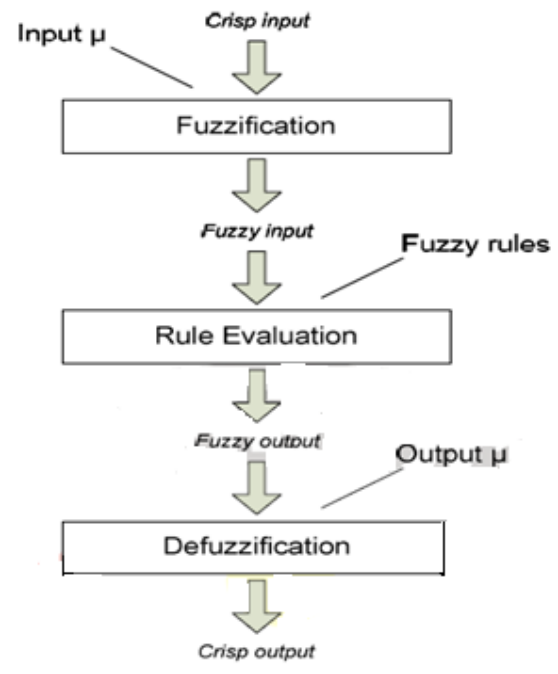


Gambar 2.10 Sistem Logika *Fuzzy*

Sumber: Fajar, 2011

Secara umum, proses yang terjadi pada sistem *fuzzy* terdiri atas 3 tahap, seperti yang tertera dalam Gambar 2.11:

1. *Fuzzification* (fuzzifikasi) adalah suatu fase di mana terjadi pengubahan masukan-masukan yang nilai kebenarannya bersifat pasti (*crisp input*) ke dalam bentuk *fuzzy input* berupa nilai linguistik, yang semantiknya ditentukan berdasarkan fungsi keanggotaan tertentu. Nilai linguistik adalah penamaan suatu grup yang mewakili keadaan atau kondisi tertentu dengan menggunakan bahasa alami.
2. *Rule evaluation* (inferensi) atau biasa disebut fase inferensi merupakan suatu fase penalaran menggunakan *fuzzy input* dan *fuzzy rules* yang telah ditentukan, sehingga diperoleh *fuzzy output*.
3. *Defuzzification* (penegasan) adalah suatu fase yang berfungsi mengubah *fuzzy output* menjadi *crisp value* berdasarkan fungsi keanggotaan yang telah ditentukan.



Gambar 2.11 Diagram Blok Sistem Berbasis Aturan *Fuzzy*

Sumber: Fajar, 2011

2.9.4 Metode penalaran Logika *Fuzzy*

Dalam *fuzzy* terdapat metode penalaran monoton sebagai teknik implikasi *fuzzy*. Jika daerah *fuzzy* direalisasikan dengan implikasi sederhana sebagai berikut:

IF *x* is A THEN *y* is B

Transfer fungsi:

y=*f*((*x*,A)B)

Maka sistem *fuzzy* dapat berjalan tanpa harus melalui komposisi dan dekomposisi *fuzzy*. Nilai *output* dapat diestimasi secara langsung dari nilai keanggotaan yang berhubungan dengan antisedennya.

Penalaran ini sudah jarang digunakan. Tiap – tiap aturan (proposisi) pada basis pengetahuan *fuzzy* akan berhubungan dengan suatu relasi *fuzzy*. Bentuk umum dari fungsi implikasi adalah:

IF *x* is A THEN *y* is B

Dengan x dan y adalah skalar dan A dan B adalah himpunan *fuzzy*. Proposisi IF adalah antiseden dan THEN disebut konsekuen. Proposisi dapat diperluas dengan menggunakan operator *fuzzy* seperti: IF (*x1* is A1) o(*x2* is A2) o (*x3* is A3) … o(*xn* is An) THEN is B Dengan o adalah operator (missal OR atau AND).

2.9.5 *Rule Evaluation* (Fase Inferensi) Menggunakan Metode Takagi-Sugeno-Kang

Sistem inferensi *fuzzy* TSK merupakan salah satu metode inferensi *fuzzy*, selain metode inferensi *fuzzy* mamdani. Sistem inferensi *fuzzy* TSK pertama kali diperkenalkan oleh Michio Sugeno pada tahun 1985, dan kemudian dikembangkan oleh Takagi dan Kang. Pada inferensi *fuzzy* Mamdani, output dari aturan *fuzzy* merupakan himpunan *fuzzy*. Berbeda dengan metode inferensi *fuzzy* Mamdani, output dari aturan *fuzzy* pada inferensi *fuzzy* TSK tidak berupa himpunan *fuzzy*, melainkan sebuah singleton.

Fuzzy singleton adalah himpunan *fuzzy* dengan keanggotan berupa sebuah nilai tunggal atau sebuah persamaan linear. Jika X dan Y merupakan himpunan semesta, maka format dari aturan *fuzzy* TSK adalah sebagai berikut:

IF *x* adalah A
AND *y* adalah B
THEN *z* adalah *f(x,y)*

Pada aturan *fuzzy* TSK di atas x, y dan z adalah variabel linguistik, sedangkan A dan B adalah himpunan *fuzzy* pada x dan y. Adapun f(x,y) adalah fungsi matematik.

2.9.6 Model Sistem Inferensi *Fuzzy* TSK

Sistem Inferensi *fuzzy* TSK memiliki dua bentuk model tergantung pada singletonnya. Kedua model sistem inferensi *fuzzy* TSK tersebut adalah sebagai berikut:

- Orde Nol

  Jika f(x,y)= konstanta, maka bentuk tersebut disebut sebagai model sistem inferensi *fuzzy* TSK orde nol dan dinyatakan sebagai berikut:

  IF *x* adalah *A*
  AND *y* adalah *B*
  THEN *z* adalah *k*

  IF ($X_1$is A_1) AND ($X_2$is A_2) AND… AND($X_n$is A_n) THEN z = k

  Dengan Ai adalah himpunan *fuzzy* ke-i sebagai anteseden dan k adalah suatu konstanta (tegas) sebagai konsekuen.

- Orde Satu

  Secara umum bentuk model *fuzzy* Sugeno Orde-Satu adalah:

  IF *x* adalah *A*
  AND *y* adalah *B*
  THEN *z*adalah $P_1*x_1+ P_2*x_2$+q

IF ($X_1$is A_1) AND… AND($X_n$is A_n) THEN z = $P_1* X_1+ \ldots + P_2* X_2+ q$

Dengan A*i* adalah himpunan *fuzzy* ke-i sebagai anteseden dan p *i* adalah suatu konstanta tegas) ke-i dan q juga merupakan konstanta dalam konsekuen.

2.9.7 Penentuan Output Fuzzy TSK

Dalam menentukan output, menggunakan *Fuzzy* Takagi- Sugeno- Kang, maka diterapkanlah metode *Weighted Average*. Dimana metode ini mengambil nilai rata-rata dengan menggunakan pembobotan berupa derajat keanggotaan. Sehingga y* didefinisikan sebagai:

$$y^{*}=\sum \frac{\mu(y)y}{\mu(y)} \qquad (2.4)$$

di mana y adalah nilai crisp dan μ(y) adalah derajat keanggotan dari nilai crisp y.

BAB III
METODOLOGI PENELITIAN

3.1 Perancangan Alat

Pada tahap perancangan alat, dibuat suatu blok diagram fungsional dari rangkaian yang direncanakan. Perancangan mekanik dilakukan dilakukan sesuai desain untuk mempermudah pembuatan. Perancangan rangkaian dilakukan pada tiap-tiap blok untuk mempermudah perancangan serta penentuan nilai komponen yang digunakan. Perancangan perangkat lunak dilakukan dengan membuat diagram alir untuk program utama dan sub program. Secara garis besar perancangan alat dilakukan dalam tahap berikut:

1. Penentuan spesifikasi alat.
2. Pembuatan diagram blok sistem keseluruhan.
3. Perancangan mekanik
4. Perancangan perangkat elektrik yang terdiri atas rangkaian mikrokontroller, rangkaian sensor.
5. Perancangan perangkat lunak *software* pengontrolan pada mikrokontroller.

3.2 Pembuatan Alat

Pembuatan alat meliputi pembuatan perangkat mekanik sebagai pemutar gerabah, perangkat elektrik sebagai komponen utama serta pembuatan perangkat lunak sebagai komponen pendukung pengujian.

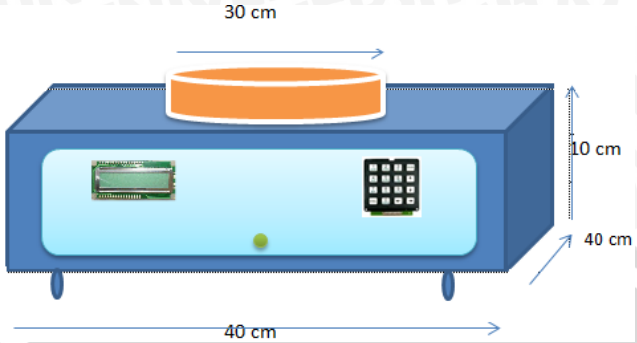
3.2.1 Pembuatan Pemutar Gerabah

Pembuatan pemutar gerabah dilakukan dengan pembuatan desain pemutar gerabah terlebih dahulu kemudian dilanjutkan dengan pembuatan kerangka pemutar gerabah dan penyusunan Motor arus searah atau *direct current* (DC) pada pemutar gerabah.

3.2.2 Pembuatan Perangkat Keras

Pembuatan alat dengan menggunakan komponen elektronika yang telah direncanakan. Pembuatan alat untuk perangkat keras juga meliputi pembuatan driver motor, pembuatan rangkaian mikrokontroller, pembuatan rangkaian sensor, dan

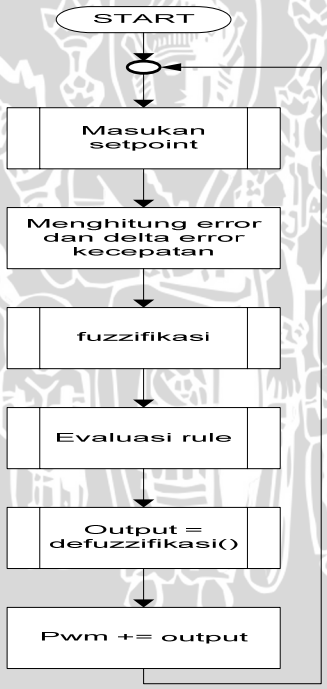
pengkabelan semuanya. Dalam Gambar 3.1 akan ditampilkan gambar rancangan perangkat keras dari pemutar gerabah.



Gambar 3.1 Desain Rancangan Perangkat Keras

3.2.3 Pembuatan Perangkat Lunak

Untuk pembuatan software dilakukan dengan pembuatan *flowchart* terlebih dahulu kemudian dibuat programnya.



Gambar 3.2 Gambar *Flowchart* Sistem Perangkat Lunak

3.3 Pengujian Alat

Untuk mengetahui apakah alat yang telah dibuat sesuai dengan yang direncanakan maka dilakukan pengujian rangkaian. Pengujian yang dilakukan terbagi dua, yaitu :

1) Pengujian perangkat keras

Perangkat keras yang telah dibuat tahap demi tahap akan diuji satu persatu sesuai blok diagram. Pengujian perangkat keras ini meliputi pengujian kestabilan putaran Motor arus searah atau *direct current* (DC) pada konveyor dari berbagai gangguan yang ada.

2) Pengujian secara keseluruhan

Pengujian secara keseluruhan dilakukan dengan menghubungkan tiap perangkat keras sesuai dengan blok diagram dan menjalankan perangkat lunaknya.

3.4 Pengambilan Kesimpulan dan Saran

Tahap terakhir adalah pengambilan kesimpulan dan saran. Kesimpulan diperoleh dari hasil pengujian alat dan kesesuaiannya dengan teori yang telah dipelajari. Saran diberikan untuk memperbaiki kesalahan, dan kemungkinan pengembangan alat agar lebih baik untuk penelitian selanjutnya.

.

.

BABIV
PERANCANGAN DAN PEMBUATAN ALAT

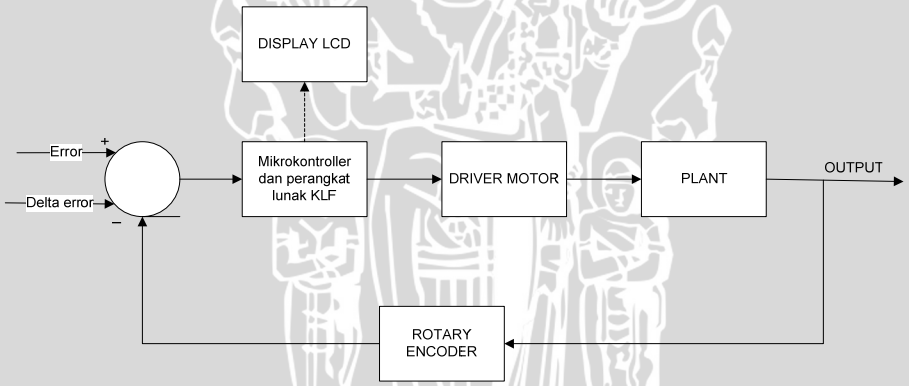
Perancangan alat ini dilakukan secara bertahap dalam bentuk blok, sehingga akan memudahkan dalam analisis pada setiap bloknya maupun secara keseluruhan.Perancangan ini terdiri dari:

- Perancangan sistem.
- Perancangan perangkat keras terdiri atas sensor *rotary encoder*, *driver*motor, dan rangkaian mikrokontroller.
- Perancangan bentuk mekanik pemutar gerabah.
- Perancangan sistem logika *fuzzy Takagi-Sugeno-Kang* serta implementasinya pada mikrokontroller.

4.1. Perancangan Sistem

4.1.1 Blok Diagram Sistem

Blok diagram sistem yang dirancang ditunjukkan dalam Gambar 4.1.



Gambar 4.1 Gambar Blok Diagram Sistem (perancangan)

Pada sistem mekanik pemutar gerabah ini menggunakan satu motor DC sebagai penggeraknya. Motor ini dihubungkan oleh sebuah poros yang berfungsi sebagai aktuator. Karena berfungsi sebagai aktuator, maka diharapkan piringan pemutar gerabah dapat berputar seiring dengan berputarnya motor DC sebagai penggerak. Pada alat ini dipasang sebuah sensor *rotaryencoder* yang berfungsi menghitung kecepatan putaran motor yang juga merupakan kecepatan gerakan pemutar gerabah. Sensor ini terhubung dengan Mikrokontroller ATMEGA 8535 .

Mikrokontroller ATMEGA 8535 berfungsi sebagai perangkat lunak yang membaca dan menghitung kecepatan dari putaran motor. Mikrokontroller menghitung besarnya Error dan Δ Error, dimana Error dan Δ Error ini akan menjadi *crisp* input bagi kontrol logika *fuzzy*. Setelah itu, dilakukan proses fuzzifikasi sehingga dihasilkan derajat keanggotaan masing-masing nilai masukan. Kemudian proses *rule evaluation*, dimana nilai masukan yang telah mempunyai derajat keanggotaan dimasukkan dalam kaidah atur yang direncanakan. Selanjutnya digunakan metode MAX-MIN pada proses rule evaluation, karena menggunakan kontrol logika *fuzzy* Takagi-Sugeno-Kang, jadi tidak memerlukan proses defuzzikasi. Sehingga langsung dilakukan perhitungan agregasi hasil dengan metode *Weighted Average.*

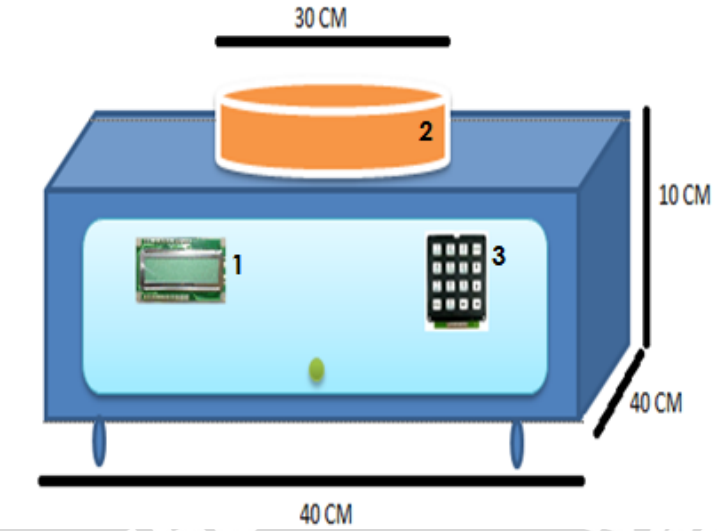
Mikrokontroller ini berhubungan dengan driver motor yang berfungsi sebagai pengkondisi sinyal dari sinyal kontrol mikrokontroller menjadi sinyal untuk menggerakkan aktuatornya yaitu motor DC. Driver motor ini berfungsi mengeksekusi hasil perhitungan dari mikrokontroller.

4.2. Perancangan Perangkat Keras (*hardware*)

Berdasarkan diagram blok perancangan alat yang telah disusun, perancangan perangkat keras meliputi perancangan mekanik, sensor *rotary encoder*dan *driver* motor serta penggunaan modul ATMega 8535. Di bawah ini adalah penjelasan masing-masing rangkaian penyusun keseluruhan alat.

4.2.1. Spesifikasi Alat

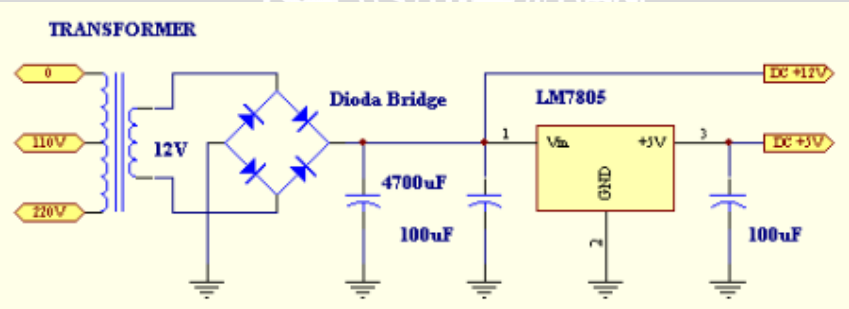
Rangka mekanik berbahan dasar kayu dengan panjang 40 cm dan lebar 40 cm. Motor penggerak alat putar berjumlah 1 buah dan di hubungkan langsung dengan bagian poros alumunium yang berfungsi memutar piringan pemutar yang berdiameter 30 cm. Sensor *rotary encoder* dipasang pada poros yang terhubung motor. Sensor terpasang lurus langsung dengan motor, agar kecepatan motor langsung terdeteksi oleh sensor. Bentuk mekanik alat pemutar gerabah dapat dilihat dalam Gambar 4.2.



Gambar 4.2 Gambar Rancangan Mekanik (1.) LCD 2.) Piringan Pemutar 3.) Keypad)

4.2.2. Rangkaian Catu Daya Sistem

Sistem mikrokontroler membutuhkan catu daya sebesar 5 volt agar dapat bekerja. Catu daya 5 volt sudah memenuhi tegangan kerja mikrokontroler ATMEGA 8535 yang berkisar antara 4,5 volt sampai 5,5 volt berdasarkan *datasheet*.Rangkaian catu daya ini menggunakan *Fixed Output Regulator* sesuai pada datasheet LM78XX. Regulator yang digunakan adalah jenis LM7805 yang memiliki tegangan keluaran minimal 4,8 volt dan maksimal 5,2 volt berdasarkan datasheetnya. Karena tegangan catu yang dibutuhkan adalah 5 volt dan tegangan keluaran regulator adalah 4.9 volt, maka tegangan untuk mikrokontroller sudah tercukupi. Dalam Gambar 4.3 berikut akan ditampilkan rangkaian catu dayanya.

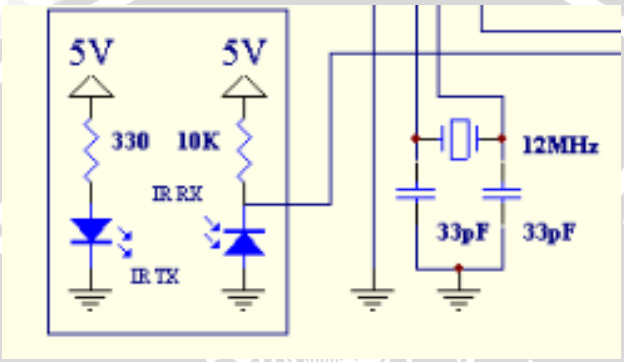


Gambar 4.3 Rangkaian Catu Daya Sistem (perancangan)

Tegangan sumber yang digunakan dalam rangkaian catu daya adalah 24 volt. Nilai keluaran regulator langsung berhubungan dengan mikrokontroller, rangkaian sensor, dan juga rangkaian driver.

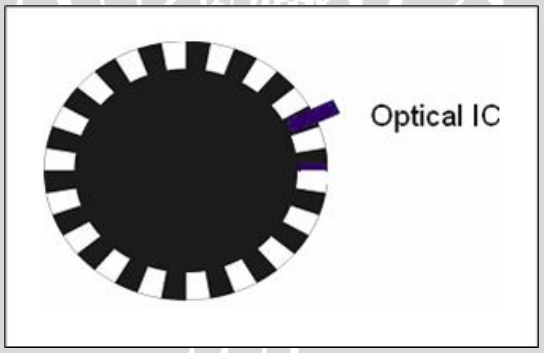
4.2.3. Sensor Rotary Encoder

Sensor putaran (*rotary encoder*) yang digunakan adalah sensor putaran berbasiskan optik menggunakan komponen *Optical IC* dan piringan berpola. Sensor rotari diletakkan pada sebuah roda yang dihubungkan dengan sebuah pegas dan tidak dipengaruhi oleh roda pergerakan. Komponen ini merupakan kombinasi LED inframerah sebagai pemancar cahaya, regulator tegangan, penguat, dan *phototransistor* NPN sebagai penerima cahaya. Dalam Gambar 4.4 menunjukkan rangkaian sensor *rotary encoder*.



Gambar 4.4Rangkaian Sensor *Rotary Encoder* (perancangan)

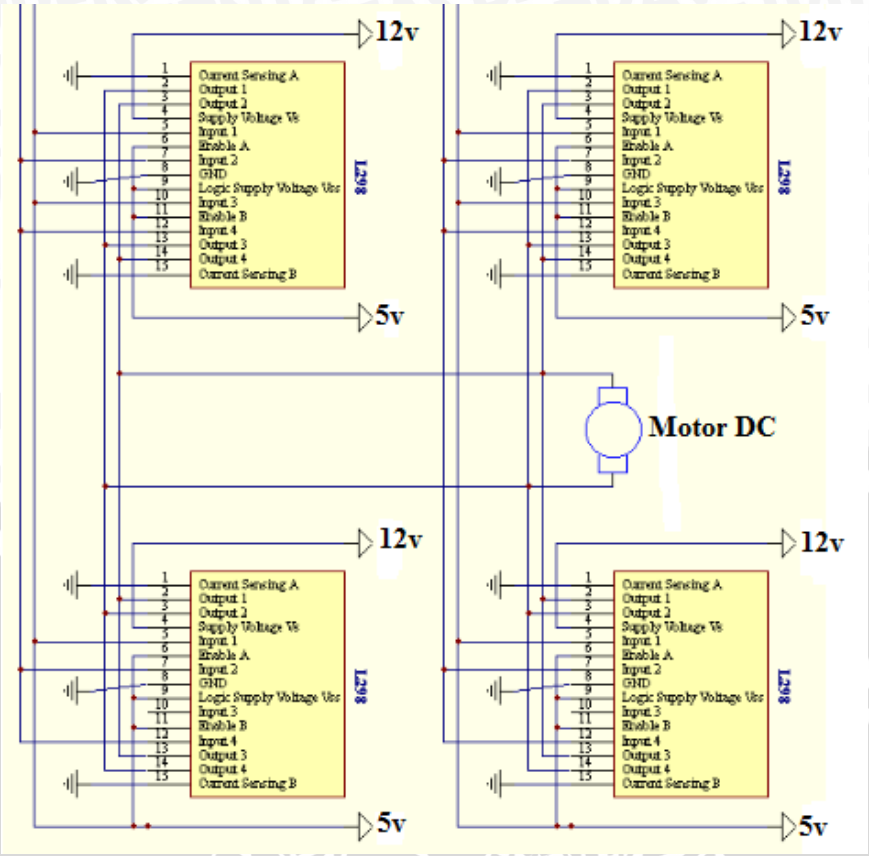
Resolusi sensor yang digunakan pada perancangan ini sebesar 60 pulsa untuk setiap putaran penuh. Sifat sinyal *incremental* didapatkan dengan membuat pola berlubang pada sisi luar piringan berpola seperti ditunjukkan dalam Gambar 4.5.



Gambar 4.5 Letak *Optical IC* Pada Piringan Berpola

4.2.4. Rangkaian Driver Motor

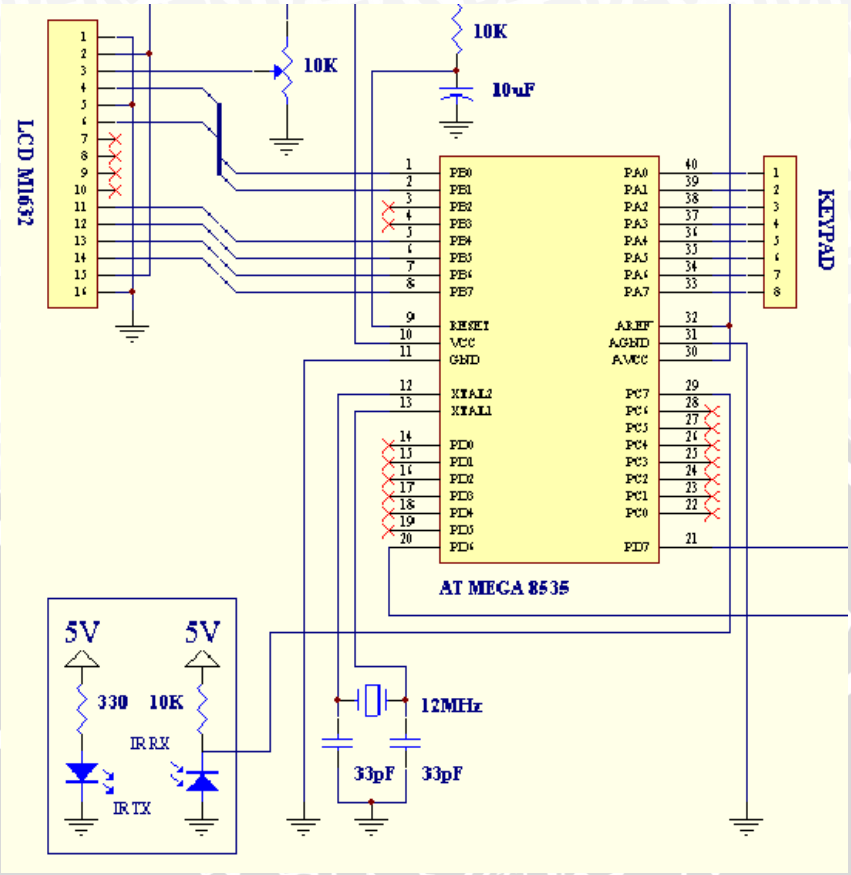
Driver pada rangkaian ini menggunakan *riley* untuk logika *ON-OFF* nya dan untuk mengontrol putaran motor digunakan E-MOSFET kanal N dengan masukan berupa sinyal PWM dari mikrokontroler. Rangkaian *driver* motor akan ditunjukkan dalam Gambar 4.6.



Gambar 4.6 Rangkaian Driver Motor (perancangan)

4.2.5. Rangkaian Mikrokontroler

Pada alat pemutar gerabah ini digunakan mikrokontroler ATMEGA 8535 sebagai pengolah utama sebagai pengolah data dari sensor *rotary encoder*, pengendali arah dan kecepatan motor serta melakukan proses logika *fuzzy*. Konfigurasi kaki I/O dari mikrokontroler ATMEGA 8535 ditunjukkan dalam Gambar 4.7.



Gambar 4.7 Konfigurasi Kaki I/O dari Mikrokontroler ATMEGA 8535(Perancangan)

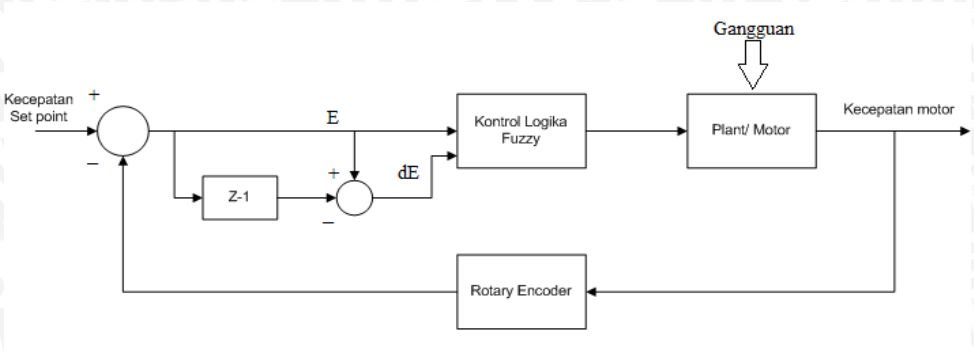
Mikrokontroler ATMEGA 8535 memiliki kaki sebanyak 40 pin, 32 diantaranya dapat diprogram menjadi masukan atau keluaran.

Tabel 4.1 Fungsi Pin Mikrokontroler

No	Pin	Fungsi
1	PA0- PA7	Jalur masukan untuk *keypad*
2	PB0	Jalur keluaran LCD
3	PB1	Jalur keluaran LCD
4	PB4	Jalur keluaran LCD
5	PB5	Jalur keluaran LCD
6	PB6	Jalur keluaran LCD
7	PB7	Jalur keluaran LCD
8	PC7	Jalur masukan *Rotary Encoder*
9	PD6	Jalur keluaran motor
10	PD7	Jalur keluaran motor

4.3. Perancangan Kontrol Logika Fuzzy

Blok diagram kontrol logika *fuzzy* ditunjukkan dalam Gambar 4.8



Gambar 4.8 Blok Diagram Kontrol Logika *Fuzzy* (perancangan)

4.3.1. Variabel Masukan dan Keluaran

Sistem kontrol logika *fuzzy* yang dikembangkan dalam penelitian ini mempunyai dua *crispinput* yaitu *error*kecepatan dan delta *error* kecepatan serta satu crisp *output* yaitu perubahan tegangan. *Error* dan Δ*error*, didefinisikan dengan perumusan sebagai berikut:

Error= SP-PV……4.1

SP = Set point (Nilai Rpm yang diinginkan)

PV = Present Value pada waktu= Rpm (Nilai aktual)

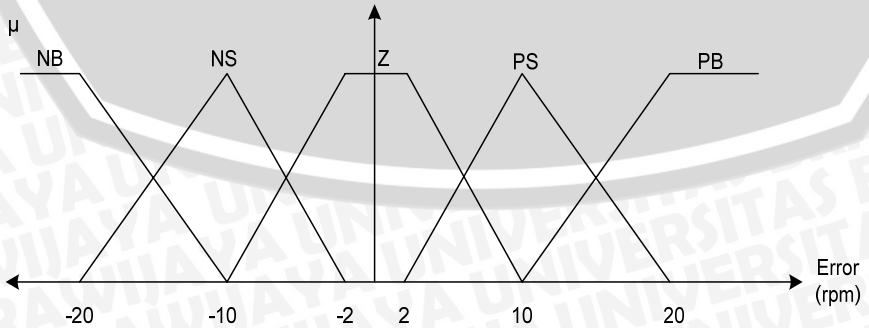
ΔError = Error(n)-Error(n-1)…….4.2

Error(n) = Error pada waktu= n

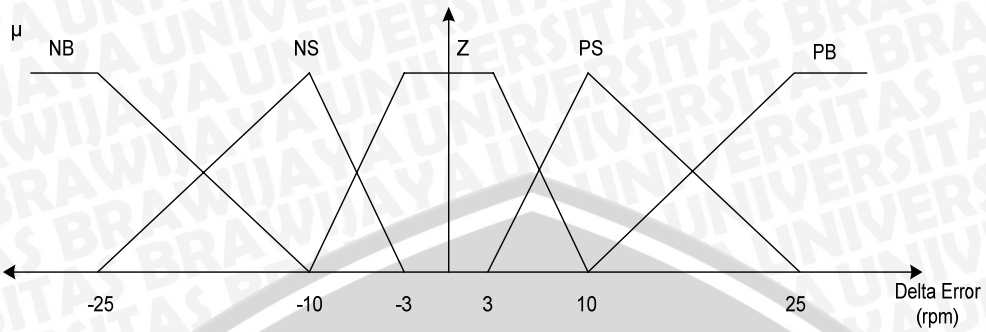
Error(n-1) = Error pada waktu = n-1

4.3.2 Fungsi Keanggotaan Masukan

Fungsi keanggotaan masukan error kecepatan dan delta error kecepatan terdiri atas lima label yaitu Negative Big (NB), Negative Small (NS), Zero (Z), Positive Small (PS), Positive Big (PB). Data masukan dari fungsi keanggotaan didapat dari *keypad* yang dibandingkan dengan hasil keluaran sensor *rotary encoder.*



Gambar 4.9 Fungsi Keanggotaan Masukan Error (perancangan)



Gambar 4.10 Fungsi Keanggotaan Masukan Delta Error (perancangan)

4.3.3 Perancangan Aturan Fuzzy

Tabel 4.2 Aturan Fuzzy

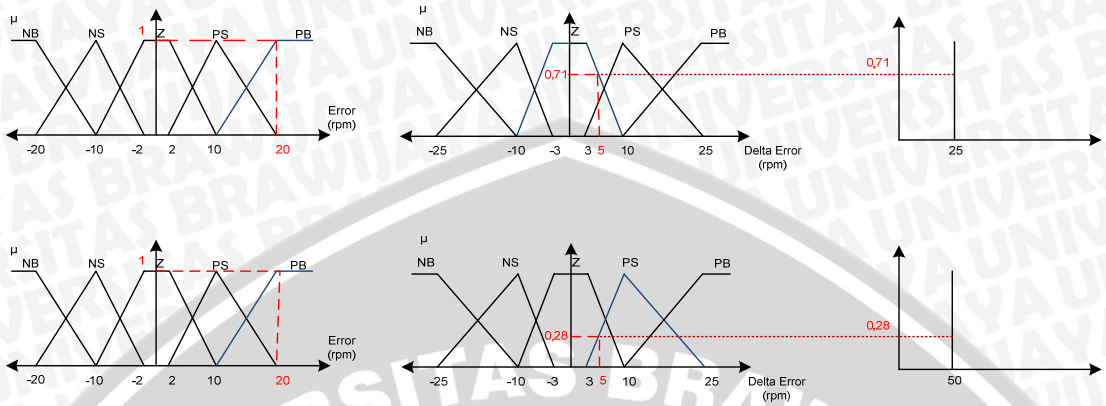
ΔError \ Error	NB	NS	Z	PS	PB
NB	-30	-30	-10	-10	0
NS	-30	-10	-10	0	10
Z	-10	-10	0	10	10
PS	-10	0	10	10	30
PB	0	10	10	30	30

Keterangan: NB=*Negative Big*
NS= *Negative Small*
Z = *Zero*
PS = *Positive Small*
PB = *Positive Big*

4.3.4 Metode Inferensi Max-Min

Untuk mengetahui metode ini pada sistem, dimisalkan visualisasi proses pengolahan masukan ketika sensor*rotary encoder* membaca posisi kecepatan datanya

13 rpm, kemudian data *set point* sebesar 33 rpm, dan misalkan $Error_{(t-1)}$ sebesar 15 rpm, maka dengan metode Max-Min ditetapkan ditunjukkan dalam Gambar4.11.



Besarnya Error (t-1)

Error = SP – PV
= 33 – 13
= 20

Error = SP – PV
= 33 - 18
= 15

Besarnya Delta Error

Delta Error = Error (t) – Error (t-1)
= 20 – 15
= 5

Gambar 4.11 Proses Inferensi Max-Min

4.3.5 Defuzzikasi

Defuzzifikasi adalah proses untuk mengubah keluaran*fuzzy* menjadi keluaran *crisp*. Hasil defuzzifikasi inilah yang digunakan untuk mengatur besarnya rasio kecepatan pada masing-masing motor DC. Metode defuzzikasi yang digunakan adalah *weighted average,* dengan rumusan sebagai berikut:

$$WA = \sum \frac{\mu(y).y}{\mu(y)} \quad \text{.....4.3}$$

Keterangan: y = nilai *crisp*

$\mu(y)$= derajat keanggotaan dari nilai *crisp* y

*2.1.1.1.1 4.4 Perancangan Perangkat Lunak (*Software*)*

Flowchart perancangan perangkat lunak ditunjukkan dalam Gambar 4.12 berikut:

Gambar 4.12*Flowchart* Perangkat Lunak Sistem (perancangan)

BAB V
PENGUJIAN DAN ANALISIS SISTEM

Pengujian dan analisis dilakukan untuk mengetahui apakah sistem telah bekerja sesuai perancangan yang telah dilakukan. Pengujian dilakukan per blok sistem kemudian secara keseluruhan. Pengujian per blok dilakukan untuk mempermudah analisis apabila alat tidak bekerja sesuai dengan perencanaan. Adapun pengujian yang dilakukan sebagai berikut:

- Pengujian sensor *rotary encoder*
- Pengujian rangkaian motor
- Pengujian perangkat lunak (kontrol logika *fuzzy*)
- Pengujian keseluruhan sistem
  - Pengujian pengontrolan mesin pemutar gerabah tanpa beban
  - Pengujian pengontrolan mesin pemutar gerabah dengan beban yang berbeda- beda.

5.1 Pengujian Sensor *Rotary Encoder*

Pengujian ini bertujuan untuk mengetahui level tegangan sinyal listrik oleh rangkaian *optocoupler* dan untuk mengetahui keberhasilan rangkaian sensor *rotary encoder* dalam mendeteksi putaran. Pengujian respon rangkaian sensor *rotary encoder*menggunakan *oscilloscope* untuk melihat karakteristik perubahan tingkat logika dari rangkaian sensor. Motor arus searah atau *direct current* (DC) terhubung dengan piringan berlubang (32 lubang) diberi sumber 12 V agar berputar.

Berdasarkan *datasheet*, *Optical IC* GP1A25LC produksi *SHARP Corporation* masih dapat membaca putaran dengan frekuensi maksimum 16000 Hz. Dengan menggunakan nilai frekuensi maksimum tersebut dapat dihitung kecepatan putaran motor maksimum yang dapat dideteksi oleh rangkaian sensor rotari menggunakan persamaan berikut.

$$\omega_{maks} = \frac{F_{maks}}{32} \qquad (5.1)$$

$$\omega_{maks} = \frac{16000 Hz}{32}$$

$$\omega_{maks} = 500 rpm$$

Berdasarkan persamaan 5.1 diketahui bahwa kecepatan putaran motor maksimum yang dapat dideteksi oleh sensor rotari adalah 5000 rpm.

5.1.1 Peralatan Pengujian

Dalam pengujian sensor *rotary encoder* ini digunakan beberapa peralatan, diantaranya:

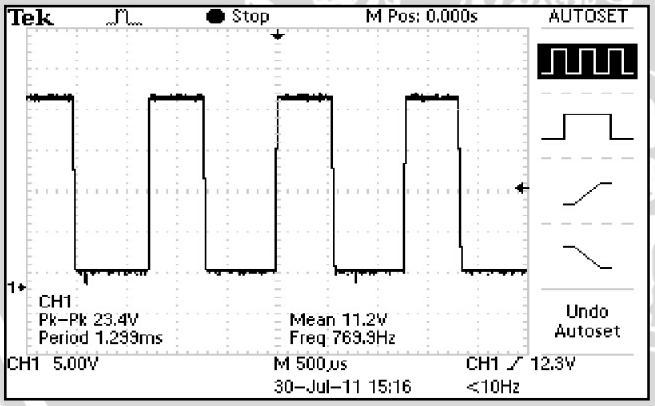
1. Motor arus searah atau *direct current* (DC)
2. Osiloskop dan *probe* osiloskop
3. Rangkaian pengkondisi sinyal *Rotary Encoder*
4. Sensor *Rotary Encoder*

5.1.2 Prosedur Pengujian

Prosedur pengujian dilakukan dengan menghubungkan *probe osciloscope* pada rangkaian pengkondisi sinyal rotari. *Channel* 1 pada osiloskop dihubungkan dengan pin output LM741 dan *channel* 2 pada osiloskop dihubungkan dengan *ground* rangkaian. Selanjutnya motor DC yang terhubung dengan sensor *Rotary Encoder* diberi tegangan maksimal agar berputar.

5.1.3 Hasil Pengujian

Setelah melakukan prosedur pengujian, maka pada osiloskop akan menunjukkan tampilan seperti dalam Gambar 5.1.



Gambar 5.1 Hasil pengujian *Rotary Encoder* menggunakan osiloskop

Berdasarkan Gambar 5.1 didapatkan *rise time* dan *fall time* pulsa saat kecepatan motor maksimal:

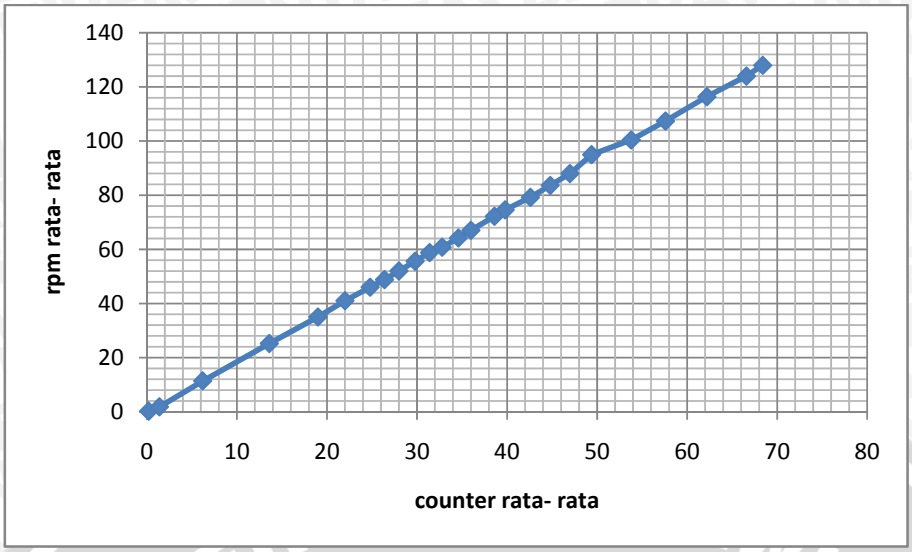
Peak to peak = 23,4V
Period = 1,299ms
Frequency = 769,9Hz

Berikut akan ditampilkan pula hasil pengujian sensor *rotary encoder* dengan nilai *Counter* dan rpm (*radiant per minute*) yang ditunjukkan pada tabel dibawah ini.

2.1.1.1.2 Tabel 5.1 Hasil Pengujian Counter *dan Rpm*

rpm rata- rata	*counter* rata- rata
0,2	0,2
1,8	1,4
11,4	6,2
25,2	13,6
35	19
41	22
46	24,8
48,8	26,4
52	28
55,6	29,8
58,8	31,4
60,8	32,8
64,2	34,6
67	36
72,2	38,6
74,6	39,8
79,2	42,6
83,6	44,8
88	47

rpm rata- rata	*counter* rata- rata
95	49,4
100,4	53,8
107,4	57,6
116,4	62,2
124	66,6
128	68,4



Gambar 5.2 Grafik Hubungan Antara Rpm dan Counter

5.2 Pengujian Motor dan Driver Motor

Tujuan dari pengujian ini adalah untuk mengetahui *output* dari *driver* motor apabila diberi *input* yang berbeda-beda.

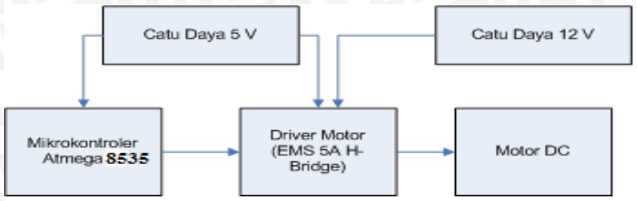
5.2.1 Peralatan yang digunakan

Dalam pengujian sensor *rotary encoder* ini digunakan beberapa peralatan, diantaranya:

1. Minimum sistem mikrokontroler ATMega 8535
2. *Driver* motor
3. Motor DC
4. Catu daya 5V dan 12V

5.2.2 Langkah pengujian

1. Merangkai peralatan seperti pada gambar
2. Mengaktifkan catu daya 5 volt dan 12 volt
3. Menghitung besar PWM dan tegangan dari mikrokontroller
4. Menghitung besar tegangan keluaran dari driver motor yang diperlukan untuk motor
5. Mencatat pergerakan motor DC
6. Mencatat nilai Rpm yang dihasilkan oleh motor



2.1.1.1.3 Gambar 5.3 Diagram Blok Pengujian Driver *Motor DC*

5.2.3 Hasil Pengujian

Hasil pengujian kecepatan motor dengan nilai *Pulse Width Modulation* (PWM) yang ditentukan ditunjukkan pada tabel5.2.

2.1.1.1.4 Tabel 5.2 Hasil Pengujian Pulse Width Modulation *(PWM)*

PWM	V MK (Volt)	V Motor (Volt)	Rpm
10	0.215	0.28	0.2
20	0.410	0.61	1.8
30	0.410	0.61	11.4
40	0.795	1.29	25.2
50	0.984	1.63	35
60	1.176	2.08	41
70	1.351	2.76	46

PWM	V MK (Volt)	V Motor (Volt)	Rpm
80	1.542	2.87	48.8
90	1.752	3.45	52
100	1.944	4.60	55.6
110	2.136	5.50	58.8
120	2.329	6.20	60.8
130	2.522	6.70	64.2
140	2.715	7.40	67
150	2.907	7.77	72.2
160	3.102	8.20	74.6
170	3.292	8.54	79.2
180	3.488	8.94	83.6
190	3.660	9.20	88
200	3.870	9.53	90.8

210	4.050	9.90	95
220	4.250	9.95	100.4
230	4.420	10.10	107.4
240	4.600	10.38	116.4
250	4.780	10.54	124
255	4.900	10.67	128

5.3 Pengujian Sistem Secara Keseluruhan

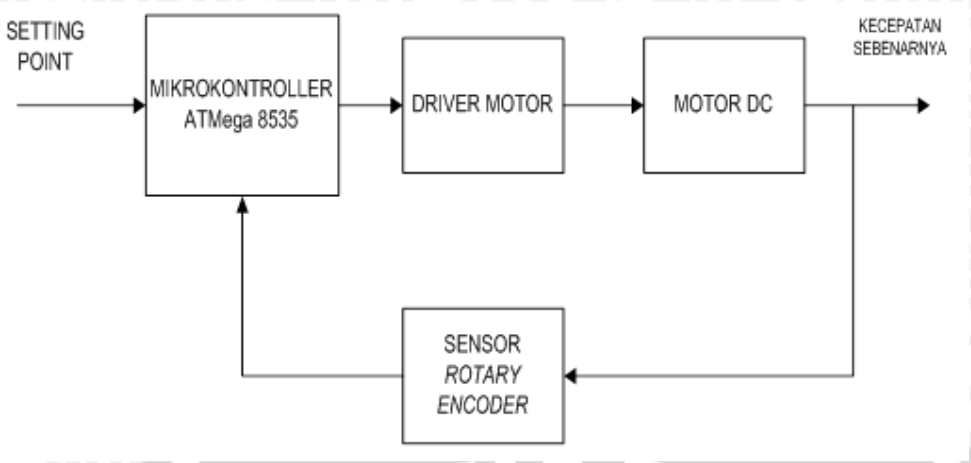
Pengujian ini bertujuan untuk mengetahui kemampuan kerja dari perangkat lunak (Kontrol Logika *Fuzzy*) dan perangkat keras saat diintegrasikan secara bersama- sama. Pengujian dilakukan dengan merangkaikan seperti pada gambar 5.8, kemudian mengaktifkan semua alat yang telah dirakit, setelah itu mengamati kerja dari sistem tersebut. Hasil keluaran dari *crisp output* diukur dan dicatat.

5.3.1 Peralatan Pengujian

1. Sensor *Rotary Encoder*
2. Beban 1 kg, 1,5 kg, dan 2 kg.
3. Minimum sistem mikrokontroller ATMEGA 8535
4. Rangkaian Driver Motor
5. Motor DC
6. LCD

5.3.2 Prosedur Pengujian

Pada pengujian sistem secara keseluruhan, langkah pertama yang harus dilakukan adalah merangkai peralatan seperti pada gambar 5.8. Kemudian kita memberi masukan setpoint dengan nilai masing- masing 33 rpm, 52 rpm, 65 rpm,82 rpm dan 95 rpm. Setelah itu kita akan melihat respon dari masing- masing setpoint ketika tanpa beban atau dengan beban 1 kg, 1,5 kg dan 2 kg, apakah sudah memenuhi *setpoint* atau belum.

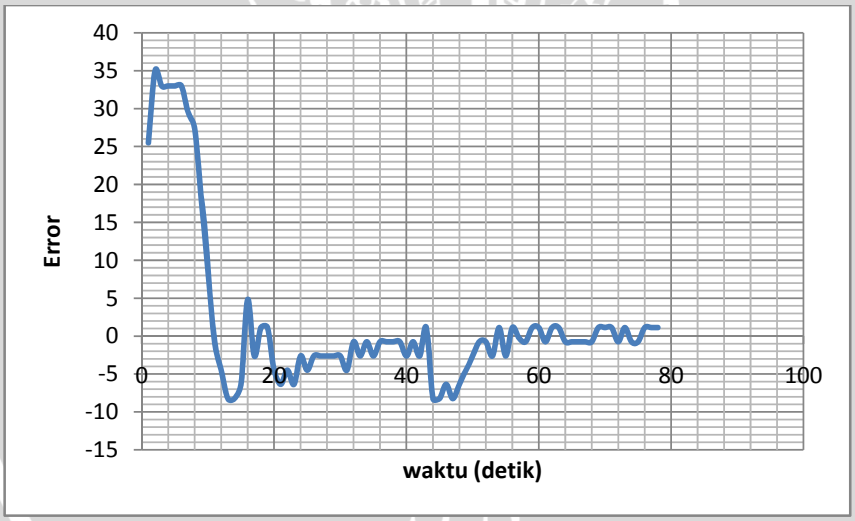


Gambar 5.4 Blok Diagram Pengujian Sistem Secara Keseluruhan (perancangan)

5.3.2.1 Pengujian Masing- masing*Setpoint*

a. Untuk *setpoint* 33 rpm

- Tanpa beban



Gambar 5.5Grafik Keluaran Error Terhadap Waktu dengan *Setpoint* 0-33 Rpm Tanpa Beban

Dari gambar 5.5 didapatkan nilai *time settling* atau t_s dan *error steady state*, berikut akan dijelaskan pengertian *time settling* atau t_s dan *error steady state*, beserta perhitungannya.

- *Time settling* atau t_s adalah waktu yang diperlukan kurva respon untuk mencapai dan menetap dalam daerah sekitar harga akhir yang nilainya ditentukan dengan presentase mutlak dari harga akhirnya (biasanya 5 % atau 2 %). T_s yang didapatkan dari pengujian ini adalah :

T_s= waktu awal di posisi acuan - waktu akhir di posisi tujuan

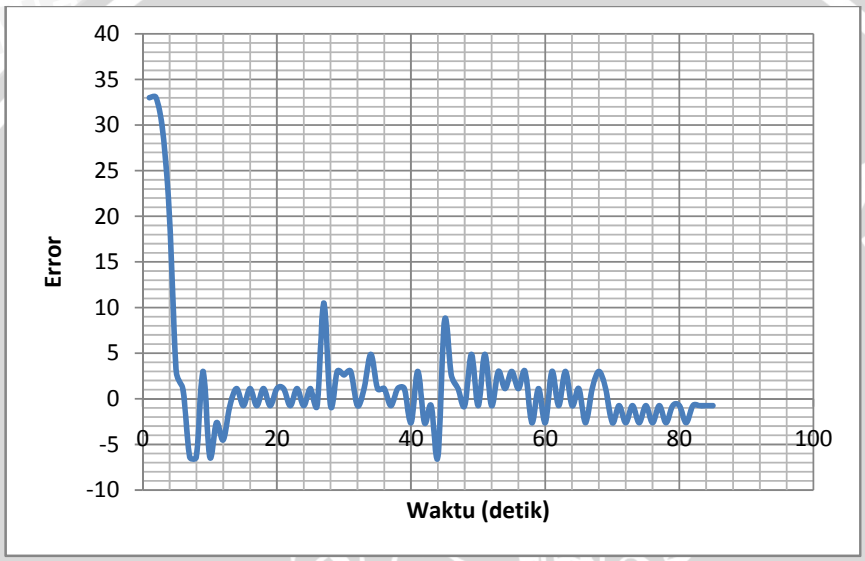
T_s=78-0

T_s= 78s

- *Error steady state* adalah selisih antara nilai keluaran dengan nilai masukan pada saat kondisi *steady state*. Ess yang didapatkan dari pengujian ini adalah:

$$ess = \frac{1{,}12}{33} \text{x } 100\ \% = 3{,}39\ \%$$

- Dengan beban 1 kg



Gambar 5.6Grafik Keluaran Error Terhadap Waktu dengan *Setpoint* 0-33 Rpm Beban 1 Kg

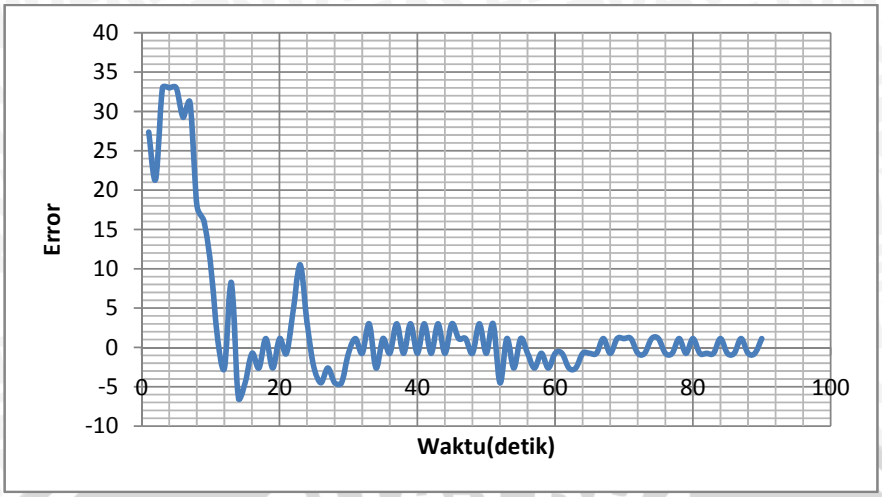
Dalam Gambar 5.6didapatkan nilai- nilai ts dan ess sebagai berikut.

t_s= 85-0

t_s= 85 s

- $ess = \frac{0{,}75}{33} \text{x } 100\ \% = 2{,}27\%$

- Dengan beban 1,5 kg



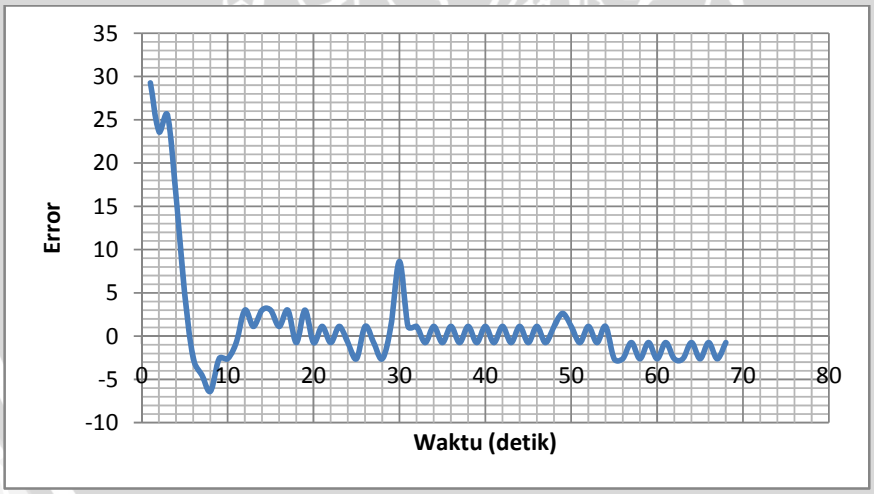
Gambar 5.7Grafik Keluaran Error Terhadap Waktu dengan *Setpoint* 0-33 Rpm Beban 1,5 Kg

Dalam Gambar 5.7didapatkan nilai- nilai ts dan ess sebagai berikut.

- t_s= 95-0

  t_s= 95 s
- $ess = \frac{0,75}{33} \text{x } 100\ \% = 2,27\%$

- Dengan beban 2 kg



Gambar 5.8Grafik Keluaran Error Terhadap Waktu dengan *Setpoint* 0-33 Rpm Beban 2 Kg

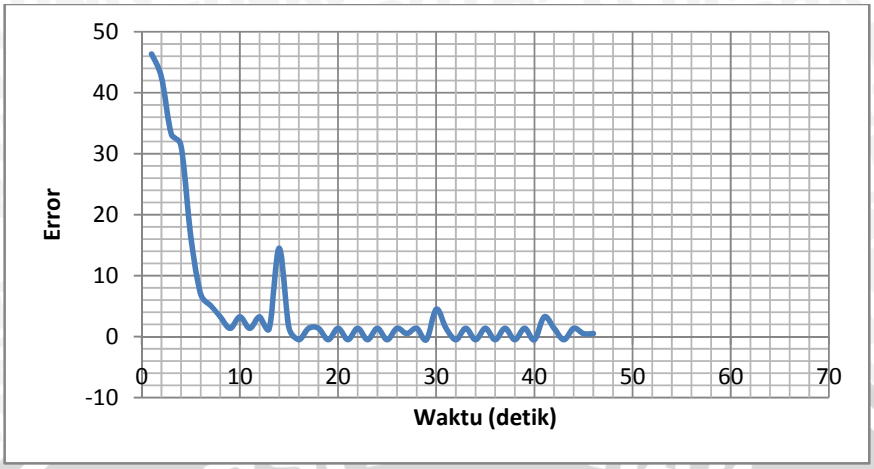
Dalam Gambar 5.8didapatkan nilai- nilai ts dan ess sebagai berikut.

- t_s= 68-0

  t_s= 68 s
- $ess = \frac{0,75}{33} \text{x } 100\ \% = 2,27\%$

b. Untuk *setpoint*52rpm

- Tanpa beban



Gambar 5.9Grafik Keluaran Error Terhadap Waktu dengan *Setpoint* 0-52 Rpm Tanpa Beban

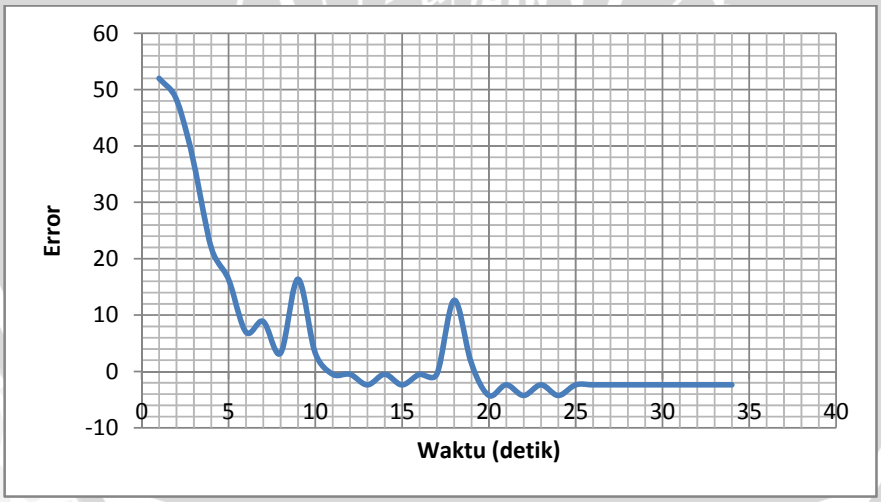
Dalam Gambar 5.9didapatkan nilai- nilai ts dan ess sebagai berikut.

- t_s= 68-0

  t_s= 68 s

- $ess = \frac{0,5}{52} \text{x } 100\ \% = 0,96\%$

- Dengan beban 1 kg



Gambar 5.10Grafik Keluaran Error Terhadap Waktu dengan *Setpoint* 0-52 Rpm Beban 1 Kg

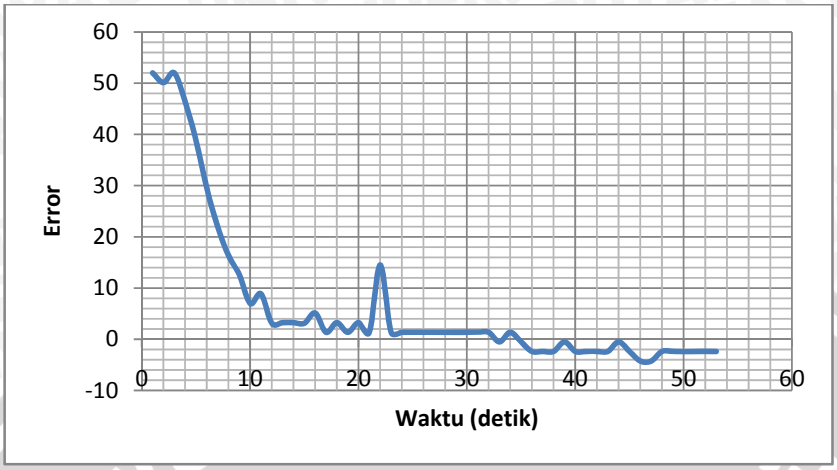
Dalam Gambar 5.10didapatkan nilai- nilai ts dan ess sebagai berikut.

- t_s= 34-0

  t_s= 34 s

- $ess = \frac{2,37}{52} x\ 100\ \% = 4,55\%$

- Dengan beban 1,5 kg



Gambar 5.11Grafik Keluaran Error Terhadap Waktu dengan *Setpoint* 0-52 Rpm Beban 1,5 Kg

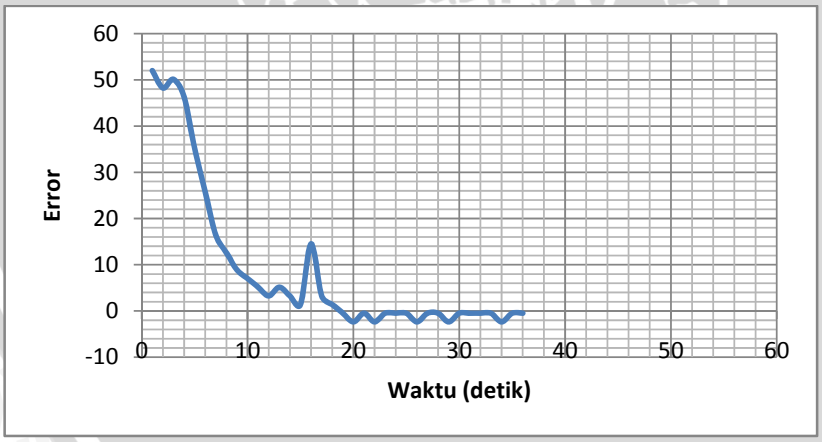
Dalam Gambar 5.11didapatkan nilai- nilai ts dan ess sebagai berikut.

- t_s= 53-0

  t_s= 53 s

- $ess = \frac{2,37}{52} x\ 100\ \% = 4,55\%$

- Dengan beban 2 kg



Gambar 5.12Grafik Keluaran Error Terhadap Waktu dengan *Setpoint* 0-52 Rpm Beban 2 Kg

Dalam Gambar 5.12didapatkan nilai- nilai ts dan ess sebagai berikut.

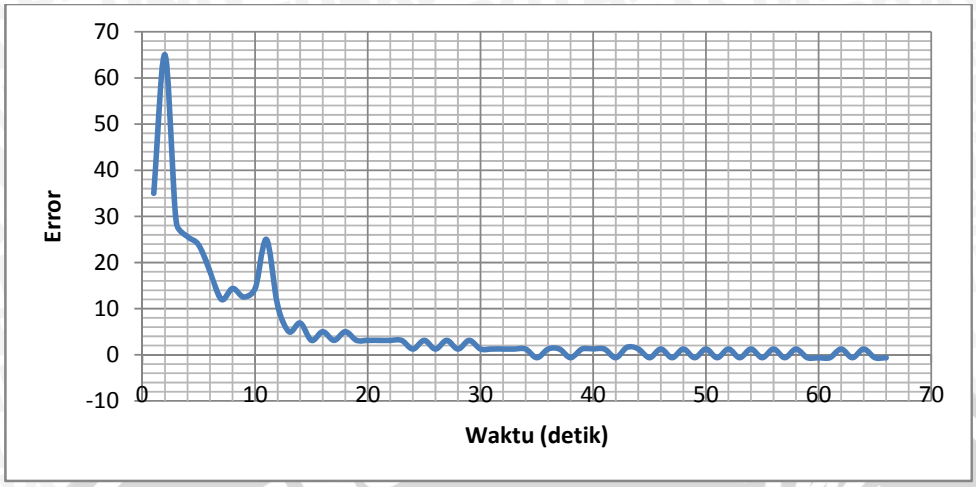
- t_s= 36-0

  t_s= 36 s

- $ess = \frac{0,5}{52} x\ 100\ \% = 0,96\%$

c. Untuk *setpoint*65rpm

- Tanpa beban



Gambar 5.13Grafik Keluaran Error Terhadap Waktu dengan *Setpoint* 0-65 Rpm Tanpa Beban

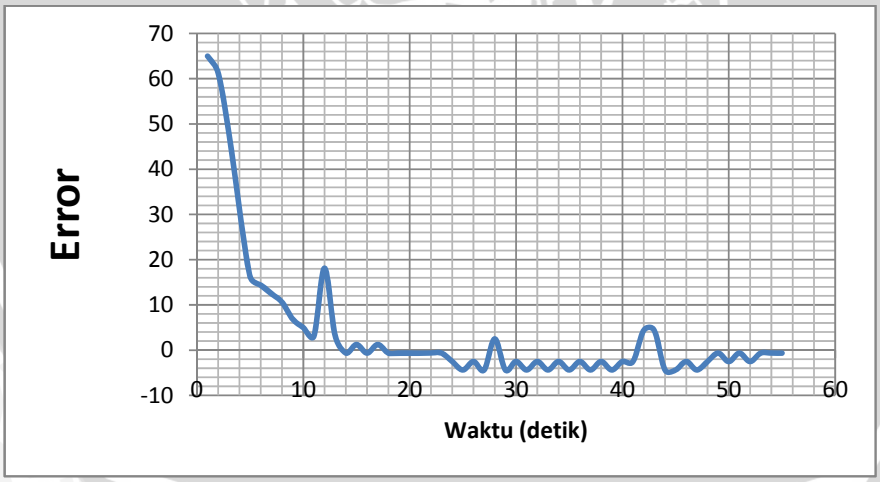
Dalam Gambar 5.13didapatkan nilai- nilai ts dan ess sebagai berikut.

- t_s= 67-0

  t_s= 67 s

- $ess = \frac{0,63}{65} \text{x } 100\ \% = 0,96\%$

- Dengan beban 1 kg



Gambar 5.14Grafik Keluaran Error Terhadap Waktu dengan *Setpoint* 0-65 Rpm Beban 1 Kg

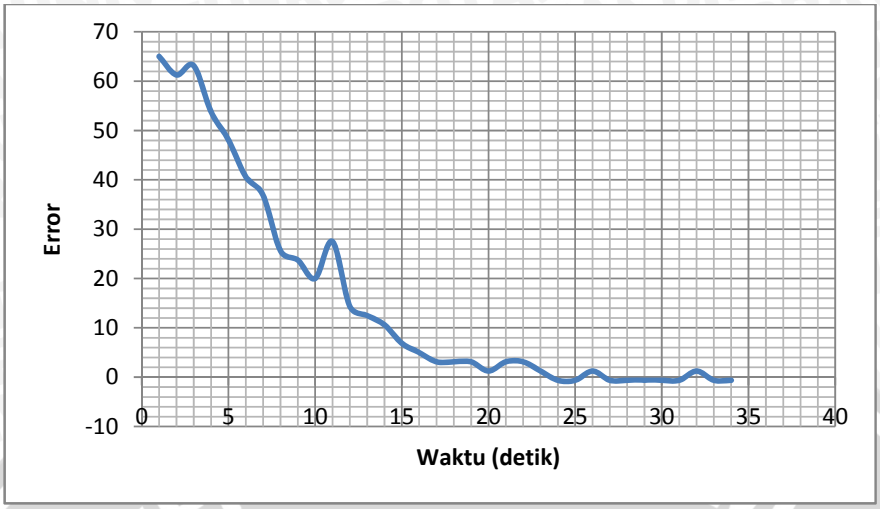
Dalam Gambar 5.14didapatkan nilai- nilai ts dan ess sebagai berikut.

- t_s= 55-0

  t_s= 55 s

- $ess = \frac{0,63}{65} \text{x } 100\ \% = 0,96\%$

- Dengan beban 1,5 kg



Gambar 5.15Grafik Keluaran Error Terhadap Waktu dengan *Setpoint* 0-65 Rpm Beban 1,5 Kg

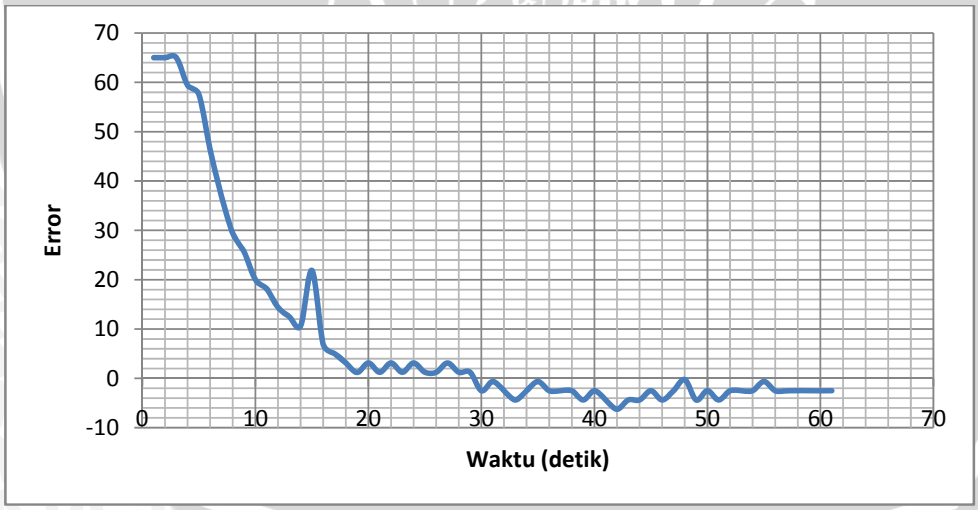
Dalam Gambar 5.15didapatkan nilai- nilai ts dan ess sebagai berikut.

- t_s= 34-0

  t_s= 34 s

- $ess = \frac{0,63}{65} \text{x } 100\ \% = 0,96\%$

- Dengan beban 2 kg



Gambar 5.16 Grafik Keluaran Error Terhadap Waktu dengan *Setpoint* 0-65 Rpm Beban 2 Kg

Dalam Gambar 5.16didapatkan nilai- nilai ts dan ess sebagai berikut.

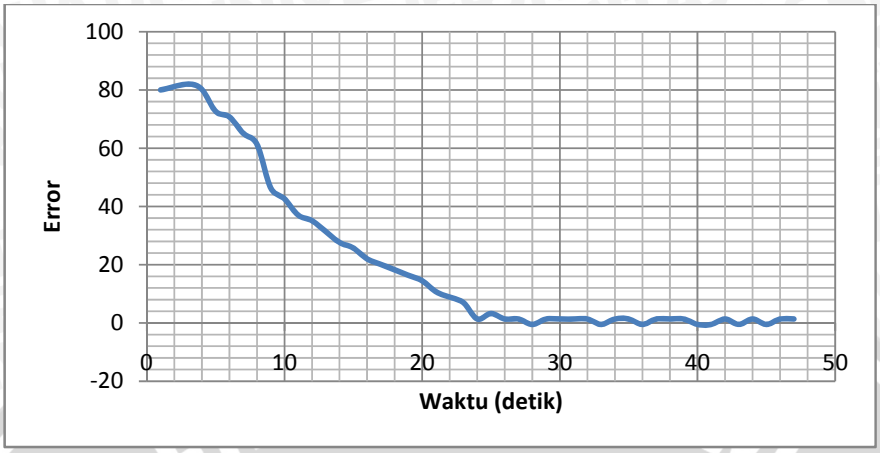
- t_s= 61-0

  t_s= 61 s

- $ess = \frac{2,5}{65} \text{x } 100\ \% = 3,84\%$

d. Untuk *setpoint*82rpm

- Dengan tanpa beban



Gambar 5.17 Grafik Keluaran Error Terhadap Waktu dengan *Setpoint* 0-82 Rpm Tanpa Beban

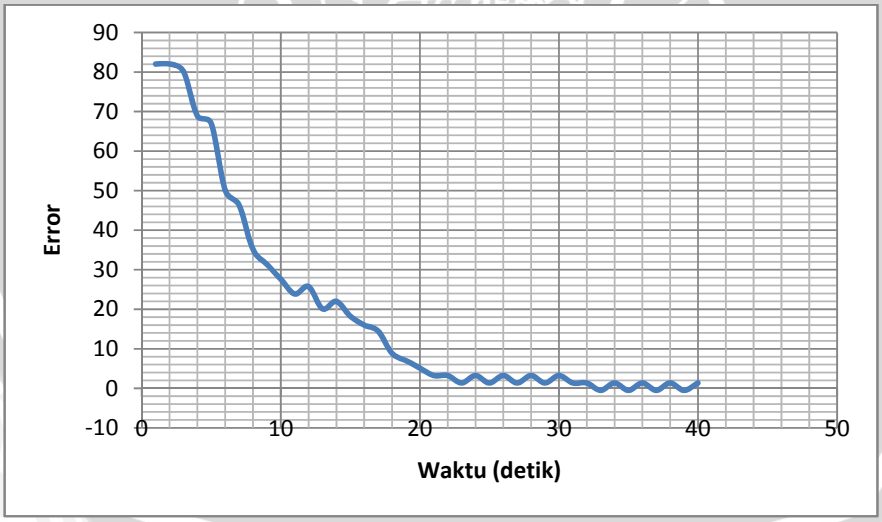
Dalam Gambar 5.17didapatkan nilai- nilai ts dan ess sebagai berikut.

- t_s= 47-0

  t_s= 47 s

- $ess = \frac{1,37}{82} \text{x } 100\ \% = 1,67\%$

- Dengan beban 1 kg



Gambar 5.18 Grafik Keluaran Error Terhadap Waktu dengan *Setpoint* 0-82 Rpm Beban 1 Kg

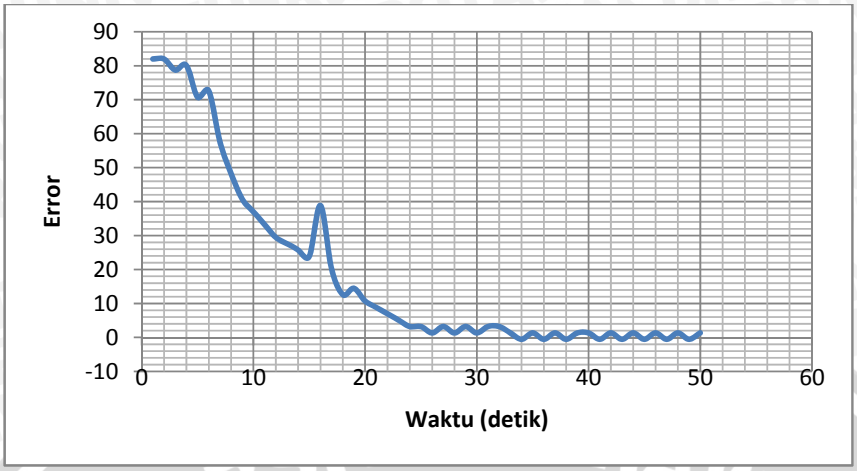
Dalam Gambar 5.18didapatkan nilai- nilai ts dan ess sebagai berikut.

- t_s= 40-0

  t_s= 40 s

- $ess = \frac{1,37}{82} \text{x } 100\ \% = 1,67\%$

- Dengan beban 1,5 kg



Gambar 5.19 Grafik Keluaran Error Terhadap Waktu dengan *Setpoint* 0-82 Rpm Beban 1,5 Kg

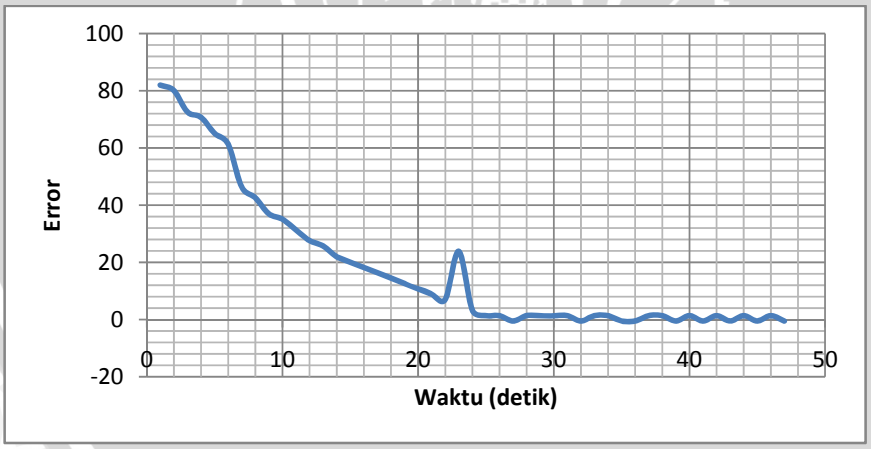
Dalam Gambar 5.19didapatkan nilai- nilai ts dan ess sebagai berikut.

- t_s= 50-0

  t_s= 50 s

- $ess = \frac{1,37}{82} \text{x } 100\ \% = 1,67\%$

- Dengan beban 2 kg



Gambar 5.20 Grafik Keluaran Error Terhadap Waktu dengan *Setpoint* 0-82 Rpm Beban 2 Kg

Dalam Gambar 5.20didapatkan nilai- nilai ts dan ess sebgai berikut.

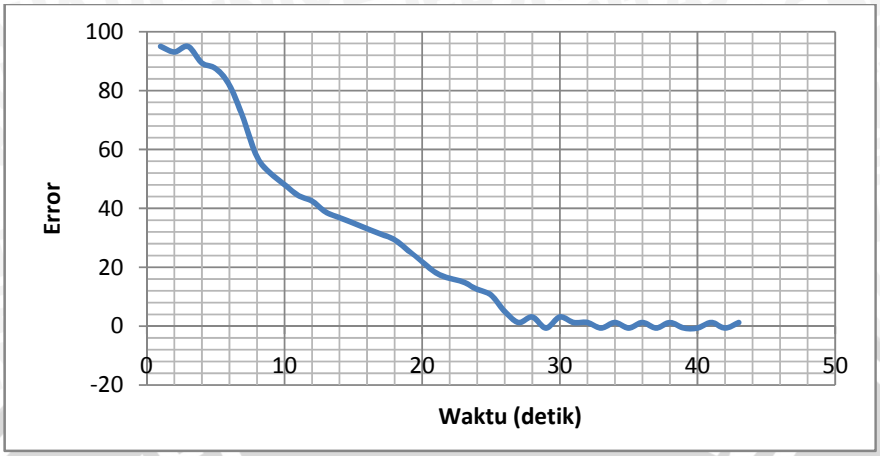
- t_s= 47-0

  t_s= 47 s

- $ess = \frac{0,5}{82} \text{x } 100\ \% = 0,6\%$

e. Untuk *setpoint*95rpm

- Tanpa beban



Gambar 5.21 Grafik Keluaran Error Terhadap Waktu dengan *Setpoint* 0-95 Rpm Tanpa Beban

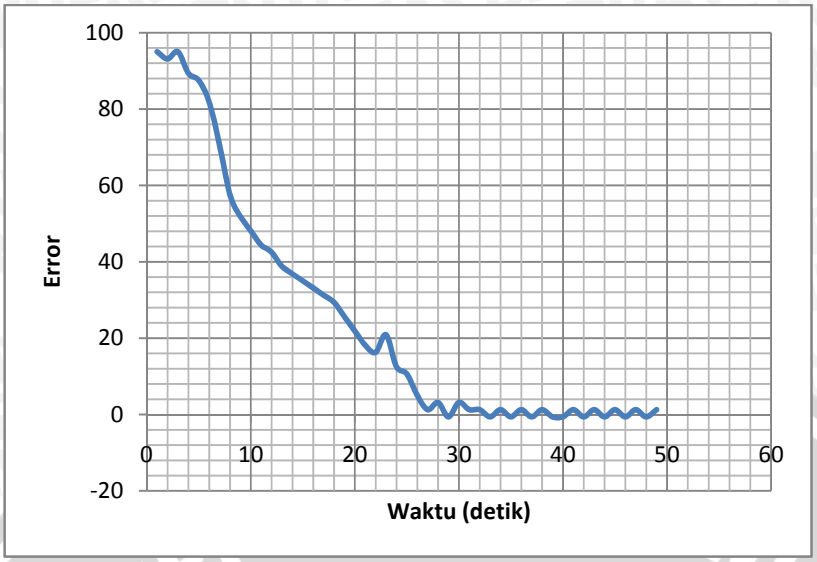
Dalam Gambar 5.21didapatkan nilai- nilai ts dan ess sebagai berikut.

- t_s= 43-0

  t_s= 43 s

- $ess = \frac{1,25}{95} \text{x } 100\ \% = 1,31\%$

- Dengan beban 1 kg



Gambar 5.22 Grafik Keluaran Error Terhadap Waktu dengan *Setpoint* 0-95 Rpm Beban 1 Kg

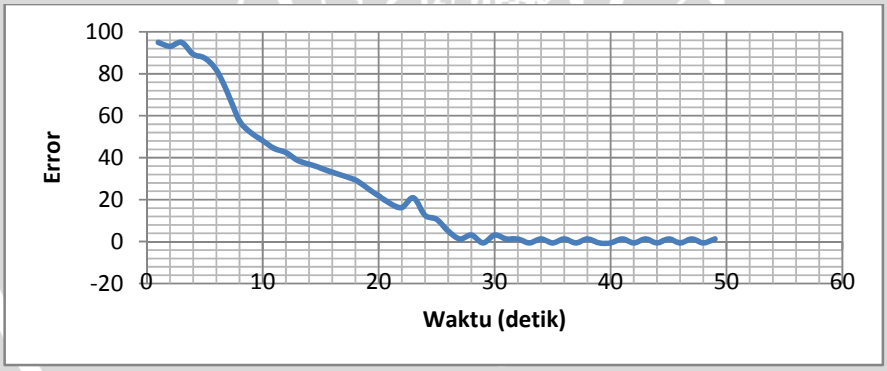
Dari gambar 5.22didapatkan nilai- nilai ts dan ess sebgai berikut.

- t_s= 49-0

  t_s= 49 s

- $ess = \frac{1,25}{95} \text{x } 100\ \% = 1,31\%$

- Dengan beban 1,5 kg



Gambar 5.23 Grafik Keluaran Error Terhadap Waktu dengan *Setpoint* 0-95 Rpm Beban 1,5 Kg

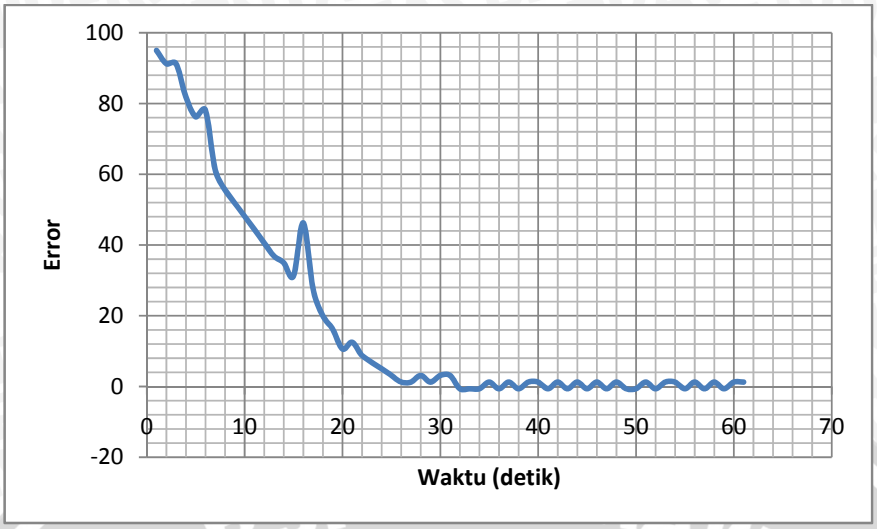
Dalam Gambar 5.23didapatkan nilai- nilai ts dan ess sebagai berikut.

- t_s= 49-0

  t_s= 49 s

- $ess = \frac{1,25}{95} \text{x } 100\ \% = 1,31\%$

- Dengan beban 2 kg



Gambar 5.24 Grafik Keluaran Error Terhadap Waktu dengan *Setpoint* 0-95Rpm Beban 2Kg

Dalam Gambar 5.24 didapatkan nilai- nilai ts dan ess sebagai berikut.

- t_s= 61-0

  t_s= 61 s

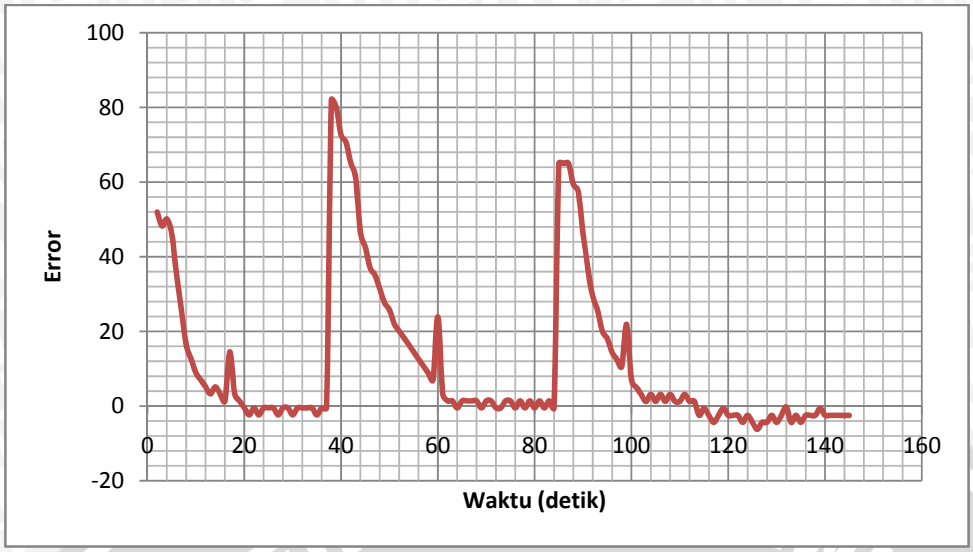
- $ess = \frac{1,25}{95} \text{x } 100\ \% = 1,31\%$

Tabel 5.3 Nilai Ts dan Ess pada Masing- Masing *Setpoint*

NO.	*SETPOINT (rpm)*	*BEBAN*							
		TANPA BEBAN		*1 KG*		*1,5 KG*		*2 KG*	
		t_s *(second)*	*ess (%)*	t_s *(second)*	*ess (%)*	t_s *(second)*	*ess (%)*	t_s *(second)*	*ess (%)*
1.	*0-33*	*78*	*3,39*	*85*	*2,27*	*95*	*2,27*	*68*	*2,27*
2.	*0-52*	*68*	*0,96*	*34*	*4,55*	*53*	*4,55*	*36*	*0,96*
3.	*0-65*	*67*	*0,96*	*55*	*0,96*	*34*	*0,96*	*61*	*3,84*
4.	*0-82*	*47*	*1,67*	*40*	*1,67*	*50*	*1,67*	*47*	*0,6*
5.	*0-95*	*43*	*1,31*	*49*	*1,31*	*49*	*1,31*	*61*	*1,31*

5.3.2.2 Pengujian Keseluruhan

Pada pengujian ini, model pemutar gerabah diberi beban sebesar 2 kg, yang kemudian dapat kita lihat respon dari masing- masing *setpoint* berupa kecepatan yang didapat dari menekan tombol *keypad*, yaitu *setpoint*pertama sebesar 52 rpm dan *setpoint*kedua sebesar 65 rpm, *setpoint*ketiga sebesar 82 rpm, maka respon sistem dengan kontrol logika*fuzzy*ditunjukkan dalam Gambar 5.25.



Gambar 5.25 Grafik Respon Keluaran Sistem dengan *Setpoint* yang Berbeda- Beda

5.3.2.3 Pengujian Tanpa Kontroller

a. Tujuan

Untuk mengetahui perbedaan respon sistem saat tidak ada kontroler dan saat ada kontroler

b. Peralatan yang digunakan

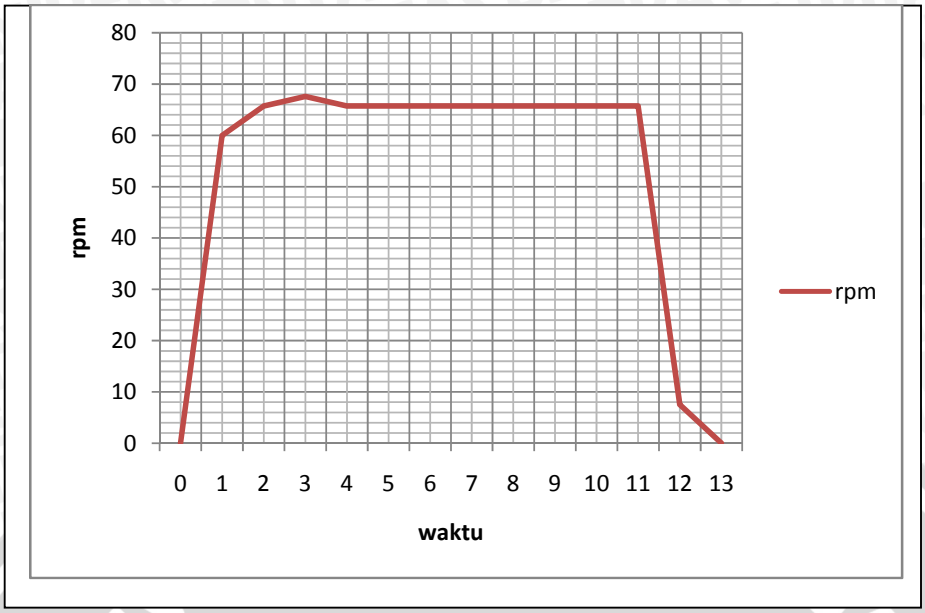
- Model pemutar gerabah lengkap dengan sensor *rotary encoder*, motor DC, *keypad*.
- Catu daya 12 volt
- Sistem mikrokontroler ATMEGA 8535
- *Driver* motor menggunakan modul EMS 5A H-Bridge

c. Langkah Pengujian

Pengujian alat ini dilakukan dengan cara menjalankan sistem pemutar gerabah dengan cara memasukkan nilai salah satu *set point*dengan tanpa kontroler yaitu 65 rpm, kemudian sistem akan berjalan sampai 10 s lalu dimatikan. Kemudian dengan *set point* yang sama dan perlakuan yang sama,pemutar gerabah dijalankan dengan kontroler.

d. Hasil Pengujian

Grafik respon tegangan masukan motor tanpa kontroler dapat dilihat dalam Gambar 5.26.



Gambar 5.26 Respon Tegangan MasukanMotor dengan *Set Point*65 rpm Tanpa Kontroler

Ts= 3,5 s

Tr= 2 s

Tf= 1,8 s

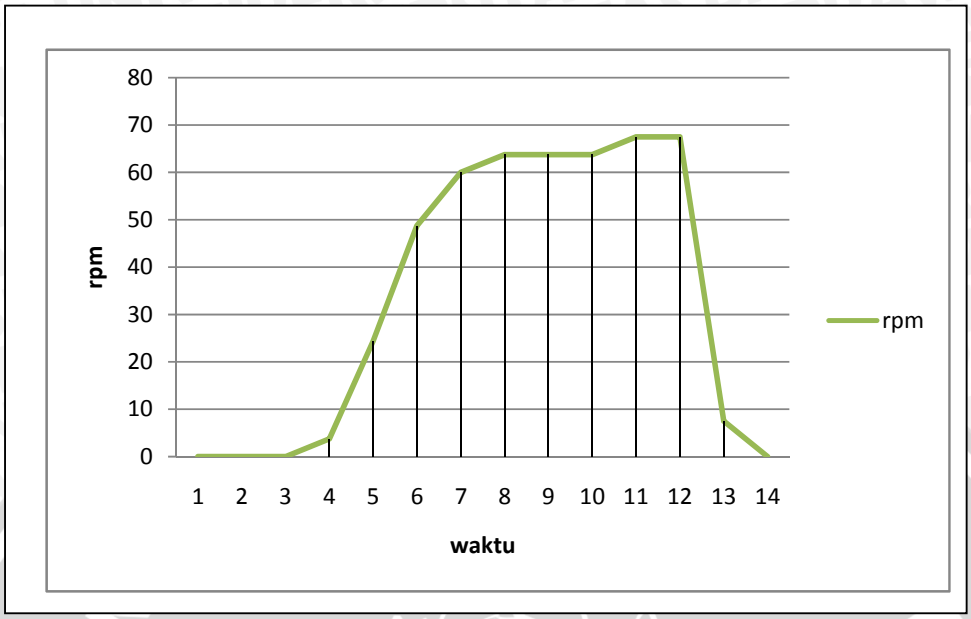
$$\tau = \frac{Tr+Tf}{2}$$

$$\tau = \frac{2+1,8}{2}$$

$$= 1,9 \text{ s}$$

$$\frac{1}{\tau s+1} = \frac{1}{1,9s+1}$$

Grafik respon sistem dengan kontroler dapat dilihat dalam Gambar 5.27



Gambar 5.27 Respon Tegangan Masukan Motor dengan Set Point 65 rpm Dengan Kontroler

BAB VI
KESIMPULAN DAN SARAN

6.1 Kesimpulan

Berdasarkan hasil pengujian tiap blok dan pengujian sistem secara keseluruhan yang telah dilakukan dalam Bab V, dapat disimpulkan beberapa hal sebagai berikut:

Hasil pengujian terhadap aplikasi kontrol logika *fuzzy* Takagi-Sugeno-Kang menunjukkan bahwa kontrol logika *fuzzy* Takagi-Sugeno-Kang dapat digunakan sebagai kontrol kecepatan pada alat pemutar gerabah.

NO.	***SETPOINT (rpm)***	***BEBAN***							
		TANPA BEBAN		***1 KG***		***1,5 KG***		***2 KG***	
		t_s ***(second)***	*ess (%)*	t_s ***(second)***	*ess (%)*	t_s ***(second)***	*ess (%)*	t_s ***(second)***	*ess (%)*
1.	***0-33***	***78***	***3,39***	***85***	***2,27***	***95***	***2,27***	***68***	***2,27***
2.	***0-52***	***68***	***0,96***	***34***	***4,55***	***53***	***4,55***	***36***	***0,96***
3.	***0-65***	***67***	***0,96***	***55***	***0,96***	***34***	***0,96***	***61***	***3,84***
4.	***0-82***	***47***	***1,67***	***40***	***1,67***	***50***	***1,67***	***47***	***0,6***
5.	***0-95***	***43***	***1,31***	***49***	***1,31***	***49***	***1,31***	***61***	***1,31***

6.2 Saran

Beberapa hal yang direkomendasikan untuk pengembangan lebih lanjut adalah:

1. Penggunaan sensor *rotary encoder* dengan jumlah lubang yang lebih banyak agar kesalahan pengukuran akan semakin kecil dan pengukuran kecepatan lebih tepat lagi.
2. Pembuatan mekanik yang baik dan lebih presisi akan membuat alat pemutar gerabah semakin stabil dalam pergerakan dan kecepatannya.
3. Penambahan sensor berat pada alat pemutar gerabah akan memudahkan dalam pengambilan data untuk pengembangan penelitian selanjutnya.

DAFTAR PUSTAKA

ATMEL. 2007. *ATMEGA8535,8-bit AVR microcontroller with 16 kbytes in system programable flash*

Fajar. 2007. *Aplikasi Logika Fuzzy dalam Optimisasi Produksi Barang Menggunakan Metode Mamdani dan Metode Sugeno*

Gerabah. http://jurnal.isi-dps.ac.id/index.php/artikel/article/view/491(25Juni2012)

Hidayat, Rosidi. *" Proses Pembuatan Roda Gigi Payung Pemutar Gerabah",*

Irwansetyo.*" Mengenal dan Memahami Prinsip Dasar Mikrokontroler AVR ATMEGA8535 untuk Simulasi dan Pemodelan Sistem Aplikasi"*

Jamsihidi, M.1993. "*Fuzzy Logic and Control*", New Jersey: Prentice-Hall.

Khasbullah, H. 2011.*Kontrol Kecepatan Pemutar Gerabah dengan Kontroller PID Menggunakan Mikrokontroller ATMega 8535*. (Skripsi). Malang: Universitas Brawijaya.

Kusumadewi, S. 1997. *Neuro Fuzzy Integrasi Sistem Fuzzy dan Jaringan Syaraf*. Jakarta: Erlangga.

Murakami, S. and Maeda, M. 1985. *Automobile Speed Control System Using a Fuzzy Logic Controller,* Amsterdam: North-Holland.

Petrafuz.1999. *Sistem Pengembangan Kendali Fuzzy Logic berbasis Mikrokontroler Keluarga MCS51,* Prosiding Seminar Nasional Penerapan Teknologi Kendali dan Instrumentasi pada Pertanian.Jakarta: BPPT.

Putra, R. 1999. *Implementasi Kendali Fuzzy Logic pada Microcontroller untuk Kendali Putaran Motor DC*. (Skripsi). Surabaya: Institut Teknologi Sepuluh November.

LAMPIRAN I

DAFTAR GAMBAR ALAT

Gambar 1. Gambar Alat Tampak Depan

Gambar 2. Gambar Alat Tampak Samping

LAMPIRAN II

LISTING PROGRAM

```
/*****************************************************
This program was produced by the
CodeWizardAVR V2.05.0 Professional
Automatic Program Generator
© Copyright 1998-2010 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com

Project : Skripsi Rahma
Version :
Date    : 7/11/2012
Author  : Bima , Aldo dan Rahma
Company :
Comments:


Chip type           : ATmega32
Program type        : Application
AVR Core Clock frequency: 12.000000 MHz
Memory model        : Small
External RAM size   : 0
Data Stack size     : 512
*****************************************************/

#include <mega32.h>
#include <stdio.h>
#include <delay.h>
#include <stdlib.h>

// Alphanumeric LCD Module functions
#include <alcd.h>
```

```
// Declare your global variables here
int cnt=0;
float rpms=0;
//int time=0;
char buf[16];
int flag;
unsigned char pwm;
int flagButt=0;
//int min = 0;
int flagMode;
int flagCnt;
int numberEntered;
float bim;

float Ea;
float Eb;
float Ec;
float Ed;
float Ee;
float Ef;

float DEa;
float DEb;
float DEc;
float DEd;
float DEe;
float DEf;

float sl;
float l;
float s;
float c;
float cs;
```

```
float err;
float Derr;

float SV;
float PV;

int i,k;
float x,y;

float uErorNb, uDErorNb;
float uErorN, uDErorN;
float uErorZ, uDErorZ;
float uErorP, uDErorP;
float uErorPb, uDErorPb;
float outU [4];
float outP [4];
float R;

// External Interrupt 0 service routine
interrupt [EXT_INT0] void ext_int0_isr(void)
{
// Place your code here
cnt++;
}

// Timer1 overflow interrupt service routine 1 second
interrupt [TIM1_OVF] void timer1_ovf_isr(void)
{
// Reinitialize Timer1 value
      TCNT1H=0xD23A >> 8;
      TCNT1L=0xD23A & 0xff;
      flag = 1;
```

```
                                                                    flagC
nt++;
    /*
    time ++;
    if (time>60){
        min++;
        time=0;
    }
    */
}

float findMin (float a, float b)
{
  if (a>b){
    return b;
  }
  if (b>=a){
    return a;
  }
}
float findUNb (float x, float a, float b)
{
  if (x<=a){
    return 1.0;
  }
  if (x>a)
  {
    if (x<b){
      return ((x-b)/(a-b));
    }
  }
  if (b<=x){
    return 0.0;
```

```
    }
}
float findUN (float x, float a, float b, float c)
{
   if (x<=a){
      return 0.0;
   }
   if (x>a){
      if(x<b){
         return ((x-a)/(b-a));
      }
   }
   if (x==b){
      return 1.0;
   }
   if (x>b){
      if (x<c){
         return ((x-c)/(b-c));
      }
   }
   if (x>=c){
      return 0.0;
   }
}
float findUZ (float x, float b, float c, float d, float e)
{
   if (x<=b){
      return 0.0;
   }
   if (x>b){
      if (x<c){
         return ((x-b)/(c-b));
      }
   }
```

```
    if (x>=c){
      if (x<= d){
        return 1.0;
      }
    }
    if (x>d){
      if (x<e){
        return ((x-e)/(d-e));
      }
    }
    if (x>=e){
      return 0.0;
    }
}
float findUP (float x, float d, float e, float f){
    if (x<=d){
      return 0.0;
    }
    if (x>d){
      if (x<e){
        return ((x-d)/(e-d));
      }
    }
    if (x==e){
      return 1.0;
    }
    if (x>e){
      if (x<f){
        return ((x-f)/(e-f));
      }
    }
    if (x>=f){
      return 0.0;
    }
```

```
}
float findUPb (float x, float e, float f)
{
  if (x<=e){
     return 0.0;
  }
  if (x>e){
     if (x<f){
       return ((x-e)/(f-e));
     }
  }
  if (x>=f){
     return 1.0;
  }
}
float fuzzy ()
{

////////////////////////////


  uErorNb = 0; uDErorNb = 0;
  uErorN = 0; uDErorN = 0;
  uErorZ = 0; uDErorZ = 0;
  uErorP = 0; uDErorP = 0;
  uErorPb= 0; uDErorPb = 0;

  x = 0; y = 0;

  for (i=0;i<4;i++){
     outU [i] =0.0;
     outP [i] =-1.0;
  }
////////////////////////////////////
```

```
//////////////////////////////////////////////////////
   uErorNb = findUNb (err, Ea, Eb);
   uErorN = findUN(err, Ea, Eb, Ec);
   uErorZ = findUZ (err,Eb, Ec, Ed, Ee);
   uErorP = findUP (err, Ed, Ee, Ef);
   uErorPb = findUPb (err, Ee, Ef);

   uDErorNb = findUNb (Derr, DEa, DEb);
   uDErorN = findUN(Derr,DEa, DEb, DEc);
   uDErorZ = findUZ (Derr,DEb, DEc, DEd, DEe);
   uDErorP = findUP(Derr, DEd, DEe, DEf);
   uDErorPb = findUPb (Derr, DEe, DEf);
   /*
   cout<<uErorNb<<"\t\t"<<uDErorNb<<endl;
   cout<<uErorN<<"\t\t"<<uDErorN<<endl;
   cout<<uErorZ<<"\t\t"<<uDErorZ<<endl;
   cout<<uErorP<<"\t\t"<<uDErorP<<endl;
   cout<<uErorPb<<"\t\t"<<uDErorPb<<endl;
   cout<<"****************************************"<<endl;
   */
//////////////////////////////////////////////////////
   i=0;
   if (uErorNb>0){
     if(uDErorNb>0){
        outU[i]=findMin (uErorNb,uDErorNb);
        outP[i]=sl;
        i++;
     }
     if(uDErorN>0){
        outU[i]=findMin (uErorNb,uDErorN);
        outP[i]=sl;
        i++;
     }
     if(uDErorZ>0){
```

```
        outU[i]=findMin (uErorNb,uDErorZ);
        outP[i]=l;
        i++;
      }
      if(uDErorP>0){
        outU[i]=findMin (uErorNb,uDErorP);
        outP[i]=l;
        i++;
      }
      if(uDErorPb>0){
        outU[i]=findMin (uErorNb,uDErorPb);
        outP[i]= s;
        i++;
      }

  }

  //N
  if (uErorN>0){
      if(uDErorNb>0){
        outU[i]=findMin (uErorN,uDErorNb);
        outP[i]= sl;
        i++;
      }
      if(uDErorN>0){
        outU[i]=findMin (uErorN,uDErorN);
        outP[i]= l;
        i++;
      }
      if(uDErorZ>0){
        outU[i]=findMin (uErorN,uDErorZ);
        outP[i]= l;
        i++;
      }
```

```
    if(uDErorP>0){
       outU[i]=findMin (uErorN,uDErorP);
       outP[i]= s;
       i++;
    }
    if(uDErorPb>0){
       outU[i]=findMin (uErorN,uDErorPb);
       outP[i]= c;
       i++;
    }
}

//Z
if (uErorZ>0){
    if(uDErorNb>0){
       outU[i]=findMin (uErorZ,uDErorNb);
       outP[i]=l;
       i++;
    }
    if(uDErorN>0){
       outU[i]=findMin (uErorZ,uDErorN);
       outP[i]=l;
       i++;
    }
    if(uDErorZ>0){
       outU[i]=findMin (uErorZ,uDErorZ);
       outP[i]=s;
       i++;
    }
    if(uDErorP>0){
       outU[i]=findMin (uErorZ,uDErorP);
       outP[i]=c;
       i++;
    }
```

```
    if(uDErorPb>0){
      outU[i]=findMin (uErorZ,uDErorPb);
      outP[i]= c;
      i++;
    }
  }

  //P
  if (uErorP>0){
    if(uDErorNb>0){
      outU[i]=findMin (uErorP,uDErorNb);
      outP[i]= l;
      i++;
    }
    if(uDErorN>0){
      outU[i]=findMin (uErorP,uDErorN);
      outP[i]= s;
      i++;
    }
    if(uDErorZ>0){
      outU[i]=findMin (uErorP,uDErorZ);
      outP[i]=c;
      i++;
    }
    if(uDErorP>0){
      outU[i]=findMin (uErorP,uDErorP);
      outP[i]=c;
      i++;
    }
    if(uDErorPb>0){
      outU[i]=findMin (uErorP,uDErorPb);
      outP[i]= c;
      i++;
    }
```

```
}

//PB
if (uErorPb>0){
  if(uDErorNb>0){
     outU[i]=findMin (uErorPb,uDErorNb);
     outP[i]=s;
     i++;
  }
  if(uDErorN>0){
     outU[i]=findMin (uErorPb,uDErorN);
     outP[i]=c;
     i++;
  }
  if(uDErorZ>0){
     outU[i]=findMin (uErorPb,uDErorZ);
     outP[i]=c;
     i++;
  }
  if(uDErorP>0){
     outU[i]=findMin (uErorPb,uDErorP);
     outP[i]=c;
     i++;
  }
  if(uDErorPb>0){
     outU[i]=findMin (uErorPb,uDErorPb);
     outP[i]= cs;
     i++;
  }
}
/*
cout<<uErorNb<<"\t\t"<<uDErorNb<<endl;
cout<<uErorN<<"\t\t"<<uDErorN<<endl;
cout<<uErorZ<<"\t\t"<<uDErorZ<<endl;
```

```
    cout<<uErorP<<"\t\t"<<uDErorP<<endl;
    cout<<uErorPb<<"\t\t"<<uDErorPb<<endl;
    cout<<"----------------------------"<<endl;
    for (i=0;i<4;i++){
       cout<<i<<" "<<outU[i]<<"\t"<<outP[i]<<endl;
    }
    cout<<"****************************"<<endl;
    */
    for (k=0;k<4;k++){
       for (i=0;i<4;i++){
          if (outP[k]==outP[i]){
             if (outU[k]>outU[i]){
                outU[i]=0;
             }
          }
       }
    }
    for (i=0;i<4;i++){
       //cout<<i<<" "<<outU[i]<<"\t"<<outP[i]<<endl;
       x+=(outP[i]*outU[i]);
       y+=outU[i];
    }
    //cout<<"########################################"<<endl;
    return x/y;
/*****************************************************/
}
void enterNumber (int x)
{
int y;
   if (numberEntered>(-1)){
      y = numberEntered*10;
      numberEntered = y+x;

      if (numberEntered>95){
```

```
        numberEntered = 95;
      }
    }
    if (numberEntered == (-1)){numberEntered = x;
    }

    if (x==11){
    numberEntered = 0;
    }
    }

    void scanKeypad ()
    {
        int x;
        x=0;
        PORTA = 0b01111111;
        if (PINA.0 == 0){
            //lcd_clear();
            //lcd_puts("bawah");
            /*
            if (pwm>0){
                pwm-=5;
            }
            OCR2=pwm;
            flagButt=0;
            */
            x=1;
        }
        if (PINA.1 == 0){
            //lcd_clear();
            //lcd_puts("3");

    if (flagMode==0){
```

```
                                                                    enter
Number (3);


}
        x=1;
    }
    if (PINA.2 == 0){
        //lcd_clear();
        //lcd_puts("2");
if (flagMode==0){


                                                                    enter
Number (2);


}
        x=1;
    }
    if (PINA.3 == 0){
        //lcd_clear();
        //lcd_puts("1");
if (flagMode==0){


                                                                    enter
Number (1);


}
        x=1;
```

```
        }
        delay_ms(50);
        if (x==0){
          PORTA = 0b10111111;
          if (PINA.0 == 0){
            //lcd_clear();
            //lcd_puts("cor");
if (flagMode==0){
              x=11;

                                                                    enter
Number (x);


}
              x=1;
          }
          if (PINA.1 == 0){
            //lcd_clear();
            //lcd_puts("6");
if (flagMode==0){


                                                                    enter
Number (6);


}
              x=1;
          }
          if (PINA.2 == 0){
            //lcd_clear();
            //lcd_puts("5");
```

```
if (flagMode==0){

                                                    enter
Number (5);

}
      x=1;
    }
    if (PINA.3 == 0){
      //lcd_clear();
      //lcd_puts("4");

if (flagMode==0){

                                                    enter
Number (4);

}
      x=1;
    }
    delay_ms(50);
  }
  if(x==0){
    PORTA = 0b11011111;
    if (PINA.0 == 0){
      //lcd_clear();
      //lcd_puts("men");
      if (flagMode==1){
        flagMode=0;
```

```
            OCR2 = 0;
            numberEntered = -1;
            PV=0;
            SV=0;
            err = 0;
            Derr = 0;
          }
          x=1;
        }
       if (PINA.1 == 0){
          //lcd_clear();
          //lcd_puts("9");

if (flagMode==0){



                                                                    enter
Number (9);



}
          x=1;
       }
       if (PINA.2 == 0){
          //lcd_clear();
          //lcd_puts("8");

if (flagMode==0){


                                                                    enter
Number (8);
```

```
}
            x=1;
          }
          if (PINA.3 == 0){
            //lcd_clear();
            //lcd_puts("7");

if (flagMode==0){

                                                                    enter
Number (7);


}
            x=1;
          }
          delay_ms(50);
      }
      if (x==0){
        PORTA = 0b11101111;
        if (PINA.0 == 0){
          //lcd_clear();
          //lcd_puts("atas");
          /*
          if (pwm<255){
              pwm+=5;
          }
          OCR2=pwm;
          flagButt=0;
          */
          x=1;
        }
        if (PINA.1 == 0){
```

```
            //lcd_clear();
            //lcd_puts("ent");
            SV = numberEntered;
            flagMode = 1;
            x=1;
          }
          if (PINA.2 == 0){
            //lcd_clear();
            //lcd_puts("0");
            if (flagMode==0){


                                                                  enter
Number (0);



}
            x=1;
          }
          if (PINA.3 == 0){
            //lcd_clear();
            //lcd_puts("can");
            x=1;
          }
        }
}
void introLcd(){
lcd_clear();
lcd_gotoxy(0,0);
   sprintf(buf,"masukan RPM : ");
   lcd_puts(buf);
lcd_gotoxy(0,1);
   if (numberEntered==(-1)){
      sprintf(buf,"0");
```

```
        lcd_puts(buf);
    }
    if (numberEntered>(-1)){
        sprintf(buf,"%d",numberEntered);
        lcd_puts(buf);
    }
    lcd_gotoxy(8,1);
sprintf(buf,"(MAX 95)");
    lcd_puts(buf);
}

void tampil()
{
      lcd_clear();
      // Place your code here
      /*
      lcd_gotoxy(0,0);
      sprintf(buf,"cnt = %d",cnt);
      lcd_puts(buf);
      */
      lcd_gotoxy(0,0);
      ftoa(err,2,buf);
      lcd_puts(buf);
      rpms=cnt*1.875;
      //rpms = (int)rpm;
      lcd_gotoxy(12,0);
      sprintf(buf,"%d",pwm);
      lcd_puts(buf);
      /*
      lcd_gotoxy(0,1);
      sprintf(buf,"rpm = ");
      lcd_puts(buf);
      ftoa(rpms,2,buf);
      lcd_puts(buf);
```

```
        */
        lcd_gotoxy(0,1);
        ftoa(bim,2,buf);
        lcd_puts(buf);
        lcd_gotoxy(11,1);
        ftoa(R,2,buf);
        /*
        if(time>9){
            sprintf(buf,"%d.%ds",min,time);
        }
        if (time<10){
            sprintf(buf,"%d.0%ds",min,time);
        } */
        lcd_puts(buf);
        cnt=0;
}


void main(void)
{
// Declare your local variables here

// Input/Output Ports initialization
// Port A initialization
// Func7=Out Func6=Out Func5=Out Func4=Out Func3=In Func2=In Func1=In
Func0=In
// State7=1 State6=1 State5=1 State4=1 State3=T State2=T State1=T State0=T
PORTA=0xF0;
DDRA=0xF0;

// Port B initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In
Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
```

```
PORTB=0x00;
DDRB=0x00;

// Port C initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In
Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTC=0x00;
DDRC=0x00;

// Port D initialization
// Func7=Out Func6=Out Func5=In Func4=In Func3=In Func2=In Func1=In
Func0=In
// State7=0 State6=0 State5=T State4=T State3=T State2=T State1=T State0=T
PORTD=0x40;
DDRD=0xC0;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=0xFF
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: 11.719 kHz
// Mode: Normal top=0xFFFF
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
```

```
// Timer1 Overflow Interrupt: On
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x05;
TCNT1H=0xD2;
TCNT1L=0x3A;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: 11.719 kHz
// Mode: Fast PWM top=0xFF
// OC2 output: Inverted PWM
ASSR=0x00;
TCCR2=0x7F;
TCNT2=0x00;
OCR2=0;
pwm=0;
// External Interrupt(s) initialization
// INT0: On
// INT0 Mode: Falling Edge
// INT1: Off
// INT2: Off
GICR|=0x40;
MCUCR=0x02;
MCUCSR=0x00;
GIFR=0x40;
```

```
// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x04;

// USART initialization
// USART disabled
UCSRB=0x00;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
SFIOR=0x00;

// ADC initialization
// ADC disabled
ADCSRA=0x00;

// SPI initialization
// SPI disabled
SPCR=0x00;

// TWI initialization
// TWI disabled
TWCR=0x00;

// Alphanumeric LCD initialization
// Connections specified in the
// Project|Configure|C Compiler|Libraries|Alphanumeric LCD menu:
// RS - PORTB Bit 2
// RD - PORTC Bit 0
// EN - PORTB Bit 3
// D4 - PORTB Bit 4
// D5 - PORTB Bit 5
```

```
// D6 - PORTB Bit 6
// D7 - PORTB Bit 7
// Characters/line: 16
lcd_init(16);

//output point
sl = -30;
l = -10;
s = 0;
c = 10;
cs = 30;

// Erorr point
Ea = -20;
Eb = -10;
Ec = -2;
Ed = 2;
Ee = 10;
Ef = 20;

// Delta Erorr point
DEa = -25;
DEb = -10;
DEc = -3;
DEd = 3;
DEe = 10;
DEf = 25;

SV = 33;
R = 0;
err = 0;
Derr = 0;

flagMode = 0;
```

```
numberEntered =-1;
flagCnt = 0;
bim = 0 ;
// Global enable interrupts
#asm("sei")

   while (1){
    // Place your code here
    scanKeypad ();
    delay_ms(100);
if (flagMode==0){
introLcd();
            flagCnt = 0;
}
if (flagMode==1){
PV=rpms;
if (flag){
            flag= 0;
            err=SV-PV;
            pwm = OCR2;
            if (flagCnt>1){
               Derr = Derr-err;
               bim = Derr;
               R = fuzzy();
               if((pwm*(-1))<=R){
                  pwm += (unsigned char) R;
               }
               OCR2 = pwm;
               flagCnt=0;
            }
            Derr = err;
            tampil();
}
}
```

}
}

LAMPIRAN III