UNIVERSITAS INDONESIA

DISAIN DAN IMPLEMENTASI PENGENDALI FUZZY BERBASIS DIAGRAM LADDER PLC MITSUBISHI Q02HCPU PADA SISTEM MOTOR INDUKSI

SKRIPSI

Diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik

SYARIF JAMALUDDIN

0906603064

HALAMAN PERNYATAAN ORISINALITAS

Saya menyatakan dengan sesungguhnya bahwa Skripsi dengan judul:

DISAIN DAN IMPLEMENTASI PENGENDALI FUZZY BERBASIS DIAGRAM LADDER PLC MITSUBISHI Q02HCPU PADA SISTEM MOTOR INDUKSI

yang dibuat untuk melengkapi sebagian persyaratan menjadi Sarjana Teknik pada Program Studi Teknik Elektro Departemen Teknik Elektro Fakultas Teknik Universitas Indonesia, sejauh yang saya ketahui bukan merupakan tiruan atau duplikasi dari Tugas Akhir yang sudah dipublikasikan dan atau pernah dipakai untuk mendapatkan gelar kesarjanaan di lingkungan Universitas Indonesia maupun di Perguruan Tinggi atau Instansi manapun, kecuali bagian yang sumber informasinya dicantumkan sebagaimana mestinya.

Depok, Januari 2012

Syarif Jamaluddin

NPM. 0906603064

HALAMAN PENGESAHAN

Skripsi ini diajukan oleh:

Nama	: Syarif Jamaluddin
NPM	: 0906603064
Program Studi	: Teknik Elektro
Judul	: Disain dan Implementasi Pengendali Fuzzy Berbasis Diagram Ladder PLC Mitsubishi Q02HCPU Pada Sistem Motor Induksi

Telah berhasil dipertahankan di hadapan Dewan Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana Teknik pada Program Studi Teknik Elektro, Fakultas Teknik, Universitas Indonesia

DEWAN PENGUJI

Pembimbing: Ir. Aries Subiantoro M.Sc. (.....…….)

Penguji 1 : Prof. Dr.Eng. Drs. Benyamin Kusumoputro M.Eng. (....….....….)

Penguji 2 :Dr. Ir. Abdul Halim M.Eng. (……..)

Ditetapkan di : Depok

Tanggal : Januari 2012

KATA PENGANTAR

Puji syukur saya panjatkan kepada Allah SWT, karena atas berkat dan rahmat-Nya, saya dapat menyelesaikan skripsi ini. Penulisan skripsi ini dilakukan dalam rangka persyaratan tahap awal penyelesaian skripsi. Saya menyadari bahwa, tanpa bantuan dan bimbingan dari berbagai pihak, pada penyusunan skripsi ini, sangatlah sulit bagi saya untuk menyelesaikan skripsi ini. Oleh karena itu, saya mengucapkan terim kasih kepada:

(1) Ir. Aries Subiantoro, M.Sc. selaku dosen pembimbing yang telah menyediakan waktu, tenaga, dan pikiran untuk mengarahkan saya dalam penyusunan skripsi ini;
(2) Dr. -Ing. Ir. Johan Prasetio selaku direktur teknik PT. Autoteknindo Sumber Makmur yang telah mempermudah saya dalam melakukan percobaan serta memperoleh data yang saya perlukan;
(3) Orang tua dan keluarga saya yang telah memberikan bantuan dukungan material dan moril;
(4) Shelly Eka Pratiwi, Aminullah Yasin, Gan Gan Nugraha serta sahabat-sahabat yang tidak bisa saya sebutkan satu-persatu yang telah banyak membantu saya dalam menyelesaikan skripsi ini.

Akhir kata, saya berharap Allah SWT berkenan membalas segala kebaikan semua pihak yang telah membantu. Semoga skripsi ini membawa manfaat bagi pengembangan ilmu dan bisa dikembangkan di masa yang akan datang.

Depok, 18 Januari 2012

Penulis

.

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI SKRIPSI UNTUK KEPENTINGAN AKADEMIS

Sebagai civitas akademik Universitas Indonesia, saya yang bertanda tangan di bawah ini:

Nama	:	Syarif Jamaluddin
NPM	:	0906603064
Program Studi	:	Teknik Elektro
Departemen	:	Teknik Elektro
Fakultas	:	Teknik
Jenis karya	:	Skripsi

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Indonesia **Hak Bebas Royalti Noneksklusif (*Non-exclusive Royalty-Free Right*)** atas karya ilmiah saya yang berjudul :

Disain dan Implementasi Pengendali Fuzzy Berbasis Diagram Ladder PLC Mitsubishi Q02HCPU Pada Sistem Motor Induksi

Beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Noneksklusif ini Universitas Indonesia berhak menyimpan, mengalihmedia/formatkan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan mempublikasikan skripsi saya tanpa meminta izin dari saya selama tetap mencantumkan nama saya sebagai penulis / pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Depok
Pada tanggal : 18 Januari 2012

Yang Menyatakan

(Syarif Jamaluddin)

ABSTRAK

Nama : Syarif Jamaluddin
Program Studi : Teknik Elektro
Judul : Disain dan Implementasi Pengendali Fuzzy Berbasis Diagram Ladder PLC Mitsubishi Q02HCPU Pada Sistem Motor Induksi

Skripsi ini membahas mengenai disain dan implementasi pengendali fuzzy pada sistem motor induksi menggunakan PLC Mitsubishi Q02HCPU, meliputi perancangan konfigurasi hardware sistem secara keseluruhan, pemrograman fuzzy logic menggunakan diagram ladder, serta perancangan sistem monitoring unjuk kerja fuzzy. Sistem ini memanfaatkan pulsa pembacaan rotary encoder yang terkopel pada pulley motor induksi sebagai feedback kecepatan real.

System fuzzy yang dibuat memiliki dua fungsi keanggotaan masukan (Error & DError) sebagai masukannya serta satu fungsi keanggotaan keluaran (DV) sebagai keluarannya. Metode inferensi yang digunakan adalah tipe max-min Mamdani. Pada proses defuzzifikasi, system dibuat dengan menggunakan dua metode sebagai pembanding, yaitu: middle of maxima dan weighted average. Untuk melakukan fungsi pengawasan, data-data input dan output berupa set point (SP), process value (PV), dan keluaran analog (DV) ditampilkan ke dalam grafik historikal dengan menggunakan software HMI Mitsubishi GT-Designer3.

Berdasarkan nilai rata-rata parameter unjuk kerja yang diperoleh dari grafik respon transien perubahan speed pada siklus Error positive dan negative, respon serta stabilitas yang dihasilkan pada percobaan menggunakan metode defuzzifikasi weighted average relatif lebih baik daripada middle of maxima.

Kata kunci: Fuzzy logic, PLC, Rotary encoder

ABSTRACT

Name : Syarif Jamaluddin
Study Program : Electrical Engineering
Title : Design and Implementation of Fuzzy Controller Based on Ladder Diagram Mitsubishi PLC Q02HCPU In Induction Motor System

This paper discusses the design and implementation of fuzzy controllers on the induction motor system using Mitsubishi PLCs Q02HCPU, including designing the hardware configuration of the overall system, programming of fuzzy logic using ladder diagrams, as well as the performance design of fuzzy monitoring systems. These systems utilize pulse which read out from rotary encoder are coupled to the pulley of induction motor as real velocity feedback.

Fuzzy systems are made to have two membership functions input (Error & DError) as input and one membership function output (DV) as the output. Max-min Mamdani type is used as inference method. In defuzzification process, the system is made by using two methods as a comparison, namely: middle of maxima and the weighted average. To perform a supervisory function the data input and output, such as: set point (SP), process value (PV), and analog output (DV) is shown on the historical graph by using the software HMI Mitsubishi GT-Designer3.

Based on the average value of the performance parameters obtained from the response transient graph in speed change on positive and negative cycle Error, response and stability generated in experiments using the defuzzification method of weighted average relatively better than the middle of maxima.

Key words: Fuzzy Logic, PLC, Rotary encoder

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Sebagian besar proses operasional mesin-mesin produksi di dunia industri menggunakan motor induksi arus bolak-balik (ac) sebagai aktuator nya. Tipe *squirrel cage* atau motor induksi sangkar tupai merupakan yang paling banyak digunakan karena harganya yang relatif murah, mudah didapatkan, serta tidak memerlukan perawatan khusus. Masalah yang timbul umumnya, *user* kesulitan untuk mendapatkan kecepatan putar yang konstan (*set point*) pada sistem yang dibutuhkan kepresisian tinggi hal ini disebabkan oleh pengaruh perubahan beban, frekuensi jala-jala, serta tegangan masukan. Untuk menjaga kestabilan kecepatan putar diperlukan suatu sistem umpan balik yang dapat memonitor kecepatan rotor secara *real*.

Fuzzy logic sebagai sistem kontrol alternatif modern dapat diterapkan untuk mengatasi permasalahan yang bersifat non linier. Dengan memanfaatkan sistem pembacaan pulsa menggunakan *rotary encoder* yang terkopel dengan *pulley* motor, informasi kecepatan aktual motor digunakan sebagai masukan *fuzzy logic* untuk memproses keluaran yang sesuai dengan *set point* yang diinginkan.

Penggunaan PLC *(Programmable Logic Controller)* telah dipercaya sebagai pengendali utama di dalam dunia industri karena telah terbukti kehandalan dan fleksibilitasnya. Ada tiga hal yang harus dipenuhi oleh sebuah piranti pengendali untuk dapat menerapkan *fuzzy logic*, yaitu: Memiliki kapasitas pemrograman yang besar, dapat bekerja didalam format bilangan real, dan memiliki kemampuan pembacaan program dengan cepat *(scanning time)*. PLC Mitsubishi tipe terbaru merupakan salah satu *brand* yang dianggap dapat memenuhi ketiga kriteria tersebut. Berdasarkan permasalahan tersebut maka skripsi ini diberi judul: "Disain dan Implementasi Pengendali *Fuzzy* Berbasis Diagram *Ladder* PLC Mitsubishi Q02HCPU Pada Sistem Motor Induksi".

1.2 Tujuan Penulisan

Adapun tujuan penulisan skripsi ini adalah:

1. Merancang system pengendali kecepatan putar motor induksi berbasis *fuzzy logic* dengan menggunakan PLC (*Programmable Logic Controller*).
2. Mendapatkan respon keluaran berupa perubahan kecepatan putar motor induksi yang ditampilkan kedalam bentuk grafik historikal.
3. Mendapatkan grafik respon transien dari perubahan kecepatan putar motor induksi
4. Mendapatkan hasil perbandingan respon keluaran dari dua metode defuzzifikasi (*weighted average* & *middle of maxima*).

1.3 Batasan Masalah

Berdasarkan latar belakang yang telah dipaparkan di atas maka penulisan skripsi ini dibatasi pada perancangan system *fuzzy logic* kedalam PLC yang dibuat dengan menggunakan bahasa pemrograman ladder, konfigurasi dari piranti-piranti pendukung, serta perancangan HMI *(Human Machine Interface)* untuk mendapatkan kinerja optimal dari system secara keseluruhan.

1.4 Metodologi Penulisan

Beberapa metode yang digunakan untuk menyusun skripsi ini adalah:

1. Metode Kepustakaan

   Yaitu pengumpulan data-data referensi yang berhubungan dengan penyusunan skripsi ini.
2. Metode Konsultasi dan Diskusi

   Yaitu berkonsultasi serta mendiskusikan langsung dengan dosen pembimbing dan pihak lain yang berkompeten di bidangnya.
3. Metode Observasi

   Yaitu meninjau informasi yang berhubungan dengan kegiatan pembuatan skripsi secara langsung.
4. Metode Pengujian

   Yaitu melakukan pengujian terhadap system yang ada untuk mendapatkan data-data hasil percobaan untuk di analisa.

1.5 Sistematika Penulisan

Untuk mempermudah penulisan maka sistematika yang digunakan dalam penulisan skripsi ini dibagi kedalam beberapa bab agar pembahasan yang diberikan mudah dipahami dan sistematis. Pada Bab 1 adalah pendahuluan yang berisikan latar belakang, tujuan penulisan, batasan masalah, metodologi penulisan, dan sistematika penulisan. Pada Bab 2 adalah teori dasar yang menerangkan tentang teori dasar yang digunakan, baik secara umum maupun khusus yang menunjang pembuatan skripsi ini. Pada Bab 3 adalah perancangan dari system *fuzzy logic*, perancangan system *monitoring* pada HMI, dan konfigurasi *hardware* secara global. Pada Bab 4 adalah percobaan dan analisa data yang menjelaskan analisis data dari hasil percobaan, berkaitan dengan tujuan yang ingin dicapai. Pada Bab 5 adalah penutup yang berisikan kesimpulan yang didapat dari pembahasan skripsi ini.

BAB 2
TEORI DASAR

2.1 *Fuzzy Logic*

Pada pertengahan 1960, Prof. Lotfi Zadeh dari universitas California di Barkeley menemukan bahwa hukum benar atau salah dari logika *Boolean* tidak memperhitungkan beragam kondisi nyata. Untuk menghitung gradasi yang tak terbatas jumlahnya antara benar atau salah, Zadeh mengembangkan ide penggolongan *set* yang ia namakan *set fuzzy*. Tidak seperti logika *Boolean*, logika *fuzzy* mempunyai banyak nilai dan membaginya kedalam derajat keanggotaan dan derajat kebenaran. Hal ini telah dibuktikan oleh Bart Kosko bahwa logika *Boolean* adalah kasus khusus dari logika *fuzzy* [3].

2.1.1 *Set Fuzzy*

Sebelum *set fuzzy* ditemukan oleh Lotfi Zadeh, kita hanya mengenal set klasik. *Set* klasik didefinisikan sebagai batasan *crisp* yang tidak mengenal ketidakpastian, dikenal juga sebagai *set crisp*. Untuk *set crisp* A dan B didalam semesta (X) maka operasi yang dapat digunakan:

- *Union*: $A \cup B = \{x \mid x \in A \text{ or } x \in B\}$
- *Intersection*: $A \cap B = \{x \mid x \in A \text{ and } x \in B\}$
- *Complement*: $\overline{A} = \{x \mid x \notin A, x \in X\}$
- *Difference*: $A|B = \{x \mid x \in A, x \notin B\}$

Untuk properti dari *set* klasik yang memiliki kesamaan dengan *set fuzzy*, didefinisikan sebagai berikut:

- *Comutativity*: $A \cup B = B \cup A$

  $A \cap B = B \cap A$
- *Associativity*: $A \cup (B \cup C) = (A \cup B) \cup C$

  $A \cap (B \cap C) = (A \cap B) \cap C$
- *Distributivity*: $A \cup (B \cap C) = (A \cup B) \cap (B \cup C)$

  $A \cap (B \cup C) = (A \cap B) \cup (B \cap C)$

- *Idempotency*: $A \cup A = A$

  $A \cap A = A$
- *Identity*: $A \cup \Phi = A$

  $A \cap X = A$

  $A \cap \Phi = \Phi$

  $A \cup X = X$
- *Transitivity*: jika $A \subseteq B \subseteq C$ maka $A \subseteq C$
- *Involution*: $\acute{A} = A$
- Hukum *De Morgans*: $A \cap B = \overline{A} \cup B$

  $A \cup B = \overline{A} \cap B$

Notasi Φ didefinisikan sebagai semesta kosong.

Set fuzzy sendiri merupakan suatu himpunan yang memuat elemen-elemen keanggotaan dengan factor kepastian / bobot *fuzzy* yang bervariasi antara interval 0 hingga 1. *Set fuzzy* A di dalam semesta (X) dapat didefinisikan dengan notasi sebagai berikut:

$\mu_A(x)$ dimana $0 < \mu_A(x) < 1$

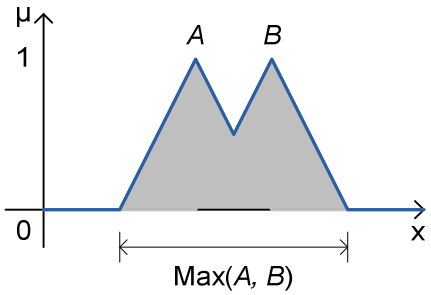
Set fuzzy hanya mengenali tiga jenis operasi:

- *Union*: $\mu_{A \cup B}(x) = \mu_A(x) \vee \mu_B(x) = \max\{\mu_A(x), \mu_B(x)\}$
- *Intersection*: $\mu_{A \cap B}(x) = \mu_A(x) \wedge \mu_B(x) = \min\{\mu_A(x), \mu_B(x)\}$
- *Complement*: $\mu_{|A}(x) = 1 - \mu_A(x)$

1. *Fuzzy Disjunction* (OR)

Adalah sebuah irisan dari dua subset didalam *sets fuzzy* dimana factor kepastian dari *fuzzy* sendiri diperoleh dengan mengambil nilai terbesar diantara keduanya. Notasi di definisikan sebagai berikut:

$$\mu_{A \cup B}(x) = \mu_A(x) \vee \mu_B(x) = \max\{\mu_A(x), \mu_B(x)\}$$



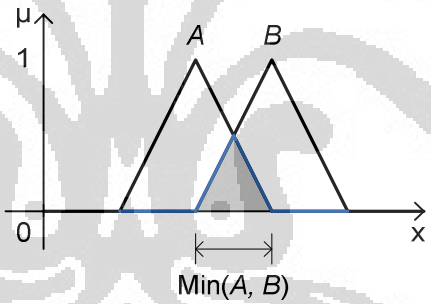
Gambar 2.1 *Fuzzy union*

Gambar 2.1 menggambarkan proses irisan dari dua fungsi keanggotaan *input / output* (A) dan (B) didalam semesta *fuzzy* (x).

2. *Fuzzy Conjunction* (AND)

Adalah sebuah gabungan dari dua subset di dalam *set fuzzy* dimana factor kepastian dari *fuzzy* sendiri diperoleh dengan mengambil nilai terkecil diantara keduanya. Notasi didefinisikan sebagai berikut:

$$\mu_{A \cap B}(x) = \mu_A(x) \wedge \mu_B(x) = \min\{\mu_A(x), \mu_B(x)\}$$



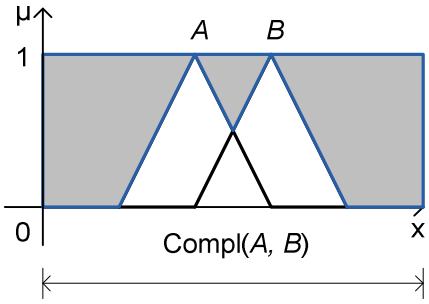
Gambar 2.2 *Fuzzy intersection*

Gambar 2.2 menggambarkan proses gabungan dari dua fungsi keanggotaan *input / output* (A) dan (B) didalam semesta *fuzzy* (x).

3. *Fuzzy Negation*

Adalah sebuah komplemen dari subset di dalam *set fuzzy* dimana factor kepastian dari *fuzzy* sendiri diperoleh melalui pengurangan antara nilai maksimum dengan nilai *fuzzy* saat itu. Notasi didefinisikan sebagai berikut:

$$\mu_{|A}(x) = 1 - \mu_A(x)$$



Gambar 2.3 *Fuzzy negation*

Gambar 2.3 menggambarkan proses komplemen dari dua fungsi keanggotaan *input / output* (A) dan (B) didalam semesta *fuzzy* (x).

2.1.2 Fuzzifikasi dan Fungsi Keanggotaan

Fuzzifikasi adalah proses transformasi dari masukan *crisp* ke dalam masukan *fuzzy*. Langkah pertama dalam fuzzyfikasi adalah menentukan label-label *fuzzy* pada daerah batasan *crisp* dari setiap masukan *crisp*. Contoh, untuk Error kita dapat menentukan daerah label ke dalam tiga label, yaitu: *Negative*, *Zero*, *Positive*. Fungsi keanggotaan dinyatakan untuk memberi arti numeric pada tiap label. Setiap fungsi keanggotaan mengidentifikasi daerah nilai masukan yang berhubungaan dengan label.

1. **Fungsi Keanggotaan *Trapezoid***

   Ditentukan oleh empat parameter {a, b, c, d} sebagai berikut:

$$\textit{Trapezoid}\ (x: a, b, c, d) = \begin{cases} 0 & ; x < a \\ (x - a)/(b - a) & ; a \le x < b \\ 1 & ; b \le x < c \\ (d - x)/(d - c) & ; c \le x < d \\ 0 & ; x \ge d \end{cases}$$

2. **Fungsi Keanggotaan *Gaussian***

   Ditentukan oleh dua parameter {m, σ} sebagai berikut:

$$\textit{Gaussian}\ (x: m, \sigma) = \exp\left(-\frac{(x-m)^2}{\sigma^2}\right)$$

3. Fungsi Keanggotaan *Bell-Shaped*

Ditentukan oleh tiga parameter {a, b, c} sebagai berikut:

$$Bell\ (x: a, b, c) = \frac{1}{1+\left|\frac{x-c}{a}\right|^{2b}}$$

4. Fungsi Keanggotaan *Sigmoidal-Shaped*

Ditentukan oleh dua parameter {a, c} sebagai berikut:

$$Sig\ (x: a, c) = \frac{1}{1+e^{-a(x-c)}}$$

2.1.3 Inferensi dan Basis Aturan

Proses untuk menghasilkan keluaran *fuzzy* yang didapat dari hubungan antara fungsi keanggotaan input dengan output berdasarkan aturan-aturan *fuzzy*. Struktur aturan-aturan dasar *fuzzy* terdiri dari dua komponen, yaitu:

IF <Antecedent> THEN <Consequent>

Pernyataan *IF* yang merupakan bagian dari Antecedent, dan pernyataan *THEN* yang merupakan bagian dari *Consequent. Antecedent* menggambarkan sebuah kondisi sedangkan consequent menggambarkan sebuah kesimpulan[1].

1. Aturan Dasar Zadeh-Mamdani

Beberapa tipe dari aturan-aturan *fuzzy* telah digunakan pada aplikasi system kendali modern saat ini. Salah satu aturan yang populer adalah aturan *fuzzy* yang dikembangkan oleh Zadeh-Mamdani, yaitu:

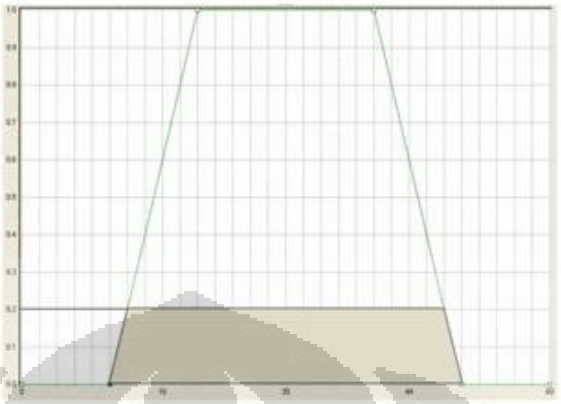
IF x_k is A_k, THEN y_k is B_k $\quad$ k = 1, 2, 3,…

x dan y adalah variable fuzzy yang mendefinisikan semesta elemen sedangkan A dan B adalah fungsi keanggotaan di dalam *sets fuzzy*[1].

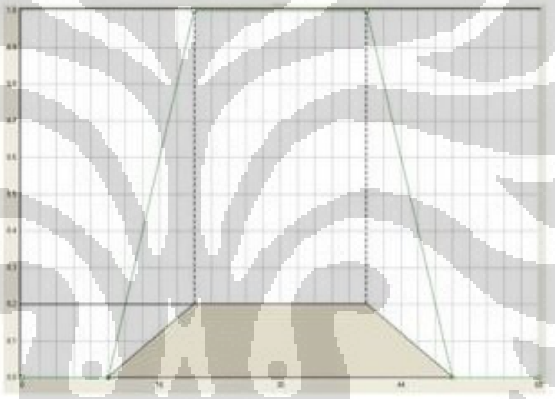
2. Metode Inferensi *Max-Min* Mamdani

Ada dua cara inferensi yang bisa digunakan pada model Mamdani, *clipping (α-cut)* atau *scalling*. Metode yang paling umum digunakan adalah *clipping* karena mudah diimplementasikan dan bila diagregasikan

dengan fungsi lain akan menghasilkan bentuk yang mudah di defuzzifikasikan.



(a)



(b)

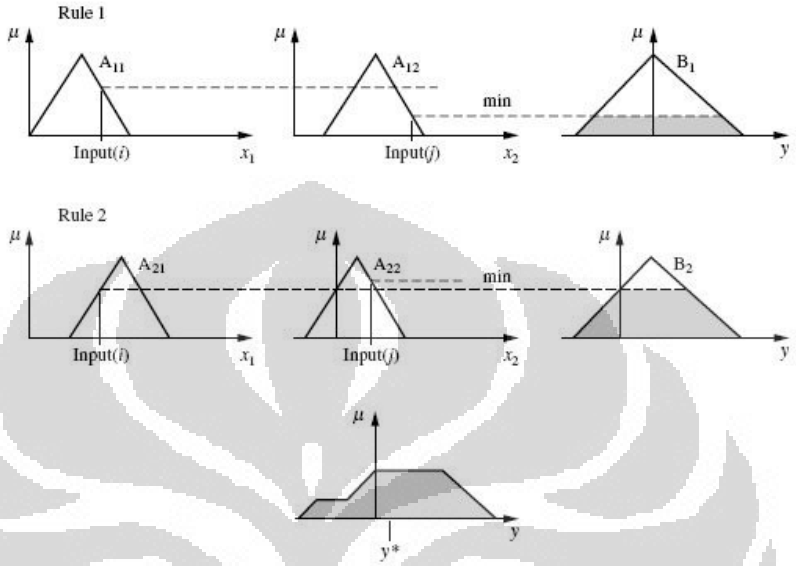
Gambar 2.4 Pola inferensi
(a). Gambar *Clipping*
(b). Gambar *Scalling*

Gambar 2.4(a) menggambarkan pola inferensi dari proses *clipping* dengan langsung memotong bagian dari subset pada suatu fungsi keanggotaan *input* tanpa merubah bentuk. Gambar 2.4(b) menggambarkan pola inferensi dari proses *scalling* dengan mengubah bentuk pola dari subset pada fungsi keanggotaan *input* ke dalam bentuk yg sama dengan bentuk aslinya dengan ukuran lebih kecil. Bentuk model inferensi Mamdani untuk dua masukan *crisp*, sebagai berikut:

$$\mu_B^k (y) = \max[\min[\mu_{A1}^k (\text{input (i)}), \mu_{A2}^k (\text{input (j)})]]$$

k = 1, 2, 3,…

agar lebih mudah untuk dipahami maka proses inferensi Mamdani dibuat kedalam bentuk grafik.



Gambar 2.5 Grafik metode inferensi Mamdani (*max-min*) dengan masukan *crisp*

Gambar 2.5 menjelaskan proses inferensi dengan menggunakan metode *max-min* Mamdani dari dua aturan fuzzy yang terdiri dari masing-masing dua *subset* pada fungsi keanggotaan *input* (i) dan (j). Pada Gambar 2.5 (A_{11}) dan (A_{21}) merupakan *subset-subset* dari fungsi keanggotaan *input* (i) sedangkan (A_{12}) dan (A_{22}) merupakan *subset-subset* dari fungsi keanggotaan *input* (j), (B_1) merupakan hasil inferensi minimum dari *subset* (A_{11}) dan (A_{12}), (B_2) merupakan hasil inferensi minimum dari *subset* (A_{21}) dan (A_{22}), (y) merupakan hasil inferensi maksimum dari (B_1) dan (B_2).

2.1.4 Defuzzifikasi

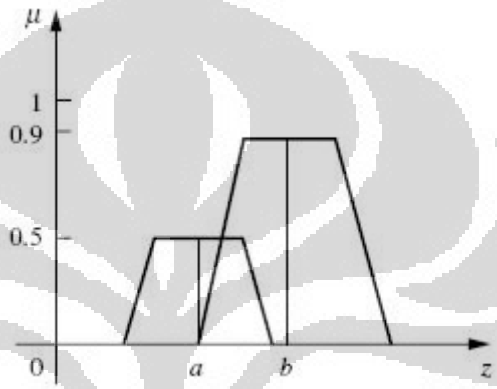
Proses pengubahan dari bentuk *fuzzy* ke dalam *crisp*. Sebelum defuzzyfikasi, yang harus dilakukan pertama kali adalah proses komposisi, yaitu agregasi hasil *clipping* dari semua aturan *fuzzy* sehingga didapatkan satu keluaran *fuzzy* tunggal. Ada beberapa metode yang bisa digunakan pada proses defuzzifikasi antara lain, *weighted average, middle of maxima,* dll.

1. Metode *Weighted Average*

Merupakan metode yang paling banyak di aplikasikan karena lebih mudah dalam komputasi dengan menggunakan microprocessor/microcontroller. Pendekatan matematis dari metode *weighted average* adalah sebagai berikut:

$$z^* = \frac{\sum \mu_c(z) * z}{\sum \mu_c(z)}$$

$$z^* = \frac{a(0.5) + b(0.9)}{0.5 + 0.9}$$



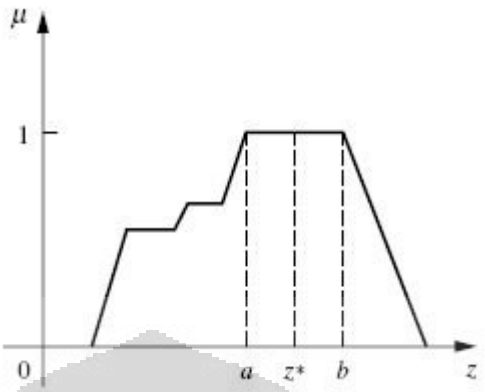
Gambar 2.6 Metode *weighted average*

Gambar 2.6 menjelaskan pembentukan nilai *crisp* dengan menggunakan metode defuzzifikasi *weighted average,* yaitu dengan mencari bobot *fuzzy* hasil inferensi dari masing-masing *subset* pada fungsi keanggotaan *output*, serta mencari nilai tengah dari garis perpotongan bobot fuzzy dari masing-masing subset pada fungsi keanggotaan *output*. Nilai *crisp* pada metode defuzzifikasi weighted average sendiri diperoleh dari jumlah perkalian antara bobot *fuzzy* hasil inferensi dengan nilai *crisp* dari masing-masing *subset* pada fungsi keanggotaan *output* dibagi dengan jumlah dari bobot *fuzzy* hasil inferensi dari masing-masing *subset* pada fungsi keanggotaan *output*.

2. Metode *Middle of Maxima (Mean-Max)*

Merupakan metode yang cukup mudah digunakan karena hanya mencari nilai tengah dari masing-masing nilai *crisp* pada fungsi keanggotaan keluaran. Pendekatan matematis dari metode *middle of maxima* adalah sebagai berikut:

$$z^* = \frac{a+b}{2}$$



Gambar 2.7 Metode *middle of maxima*

Gambar 2.7 menjelaskan pembentukan nilai *crisp* dengan menggunakan metode defuzzifikasi *middle of maxima,* yaitu dengan mencari bobot *fuzzy* hasil inferensi dari masing-masing *subset* pada fungsi keanggotaan *output*, serta mencari nilai tengah dari garis perpotongan bobot fuzzy dari masing-masing subset pada fungsi keanggotaan *output*. Nilai *crisp* pada metode defuzzifikasi weighted average sendiri diperoleh dari jumlah perkalian antara bobot *fuzzy* hasil inferensi dengan nilai *crisp* dari masing-masing *subset* pada fungsi keanggotaan *output* dibagi dengan banyaknya *subset* yang aktif pada fungsi keanggotaan *output*.

2.2 Karakteristik Motor Arus Bolak-Balik (AC)

2.2.1 Keuntungan dan Kerugian Motor Induksi

Motor induksi, merupakan motor yang memiliki konstruksi yang baik, harganya lebih murah dan mudah dalam pengaturan kecepatannya, stabil ketika berbeban dan mempunyai efisiensi tinggi. Mesin induksi adalah mesin *ac* yang paling banyak digunakan dalam industri dengan skala besar maupun kecil, dan dalam rumah tangga. Alasannya adalah bahwa karakteristiknya hampir sesusai dengan kebutuhan dunia industri, pada umumnya dalam kaitannya dengan harga, kesempurnaan, pemeliharaan, dan kestabilan kecepatan. Mesin induksi (asinkron) ini pada umumnya hanya memiliki satu suplai tenaga yang mengeksitasi belitan stator. Belitan rotornya tidak terhubung langsung dengan sumber tenaga listrik,

melainkan belitan ini dieksitasi oleh induksi dari perubahan medan magnetik yang disebabkan oleh arus pada belitan stator.

Hampir semua motor *ac* yang digunakan adalah motor induksi, terutama motor induksi tiga fasa yang paling banyak dipakai di perindustrian. Motor induksi tiga fasa sangat banyak dipakai sebagai penggerak di perindustrian karena memiliki banyak keuntungan, tetapi juga memiliki beberapa kelemahan.

Keuntungan motor induksi tiga fasa :

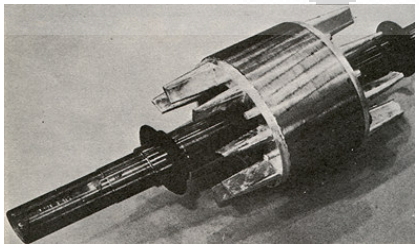
1. Motor induksi tiga fasa sangat sederhana dan kuat.
2. Biayanya murah dan dapat diandalkan.
3. Motor induksi tiga fasa memiliki efisiensi yang tinggi pada kondisi kerja normal.
4. Perawatannya mudah.

Kerugian motor induksi tiga fasa :

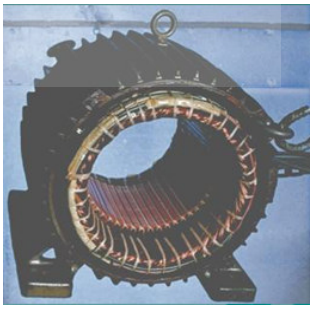
1. Kecepatannya tidak bisa bervariasi tanpa merubah efisiensi.
2. Kecepatannya tergantung beban.
3. Pada torsi start memiliki kekurangan.

2.2.2 Konstruksi Motor Induksi Tiga Fasa

Motor induksi adalah motor ac yang paling banyak dipergunakan, karena konstruksinya yang kuat dan karakteristik kerjanya yang baik. Secara umum motor induksi terdiri dari *rotor* dan *stator*. *Rotor* merupakan bagian yang bergerak, sedangkan stator bagian yang diam. Diantara *stator* dengan *rotor* ada celah udara yang jaraknya sangat kecil. Konstruksi motor induksi dapat dilihat pada Gambar 2.8.



(a)



(b)

Gambar 2.8 Konstruksi motor induksi

(a). Gambar *Rotor* (b). Gambar S*tator*

Komponen *stator* adalah bagian terluar dari motor yang merupakan bagian yang diam dan mengalirkan arus fasa. *Stator* terdiri atas tumpukan laminasi inti yang memiliki alur yang menjadi tempat kumparan dililitkan yang berbentuk silindris.

2.2.3 Prinsip Kerja Motor Induksi Tiga Fasa

Motor induksi adalah peralatan pengubah energi listrik ke bentuk energi mekanik. Pengubahan energi ini bergantung pada keberadaan fenomena alami magnetik, medan listrik, gaya mekanis dan gerak. Jika pada belitan *stator* diberi tegangan tiga fasa, maka pada belitan *stator* akan mengalir arus tiga fasa, arus ini menghasilkan medan magnet yang berputar dengan kecepatan sinkron (n_s). Medan magnet ini akan memotong belitan *rotor*, sehingga pada belitan *rotor* akan diinduksikan tegangan seperti halnya tegangan yang diinduksikan dalam lilitan sekunder transformator oleh fluksi yang dihasilkan arus pada belitan primer. Rangkaian *rotor* merupakan rangkaian tertutup, baik melalui cincin ujung atau tahanan luar. Tegangan induksi pada *rotor* akan menghasilkan arus yang mengalir pada belitan *rotor*. Arus yang mengalir pada belitan *rotor* berada dalam medan magnet yang dihasilkan stator, sehingga pada belitan *rotor* akan dihasilkan gaya (F). Gaya ini akan menghasilkan torsi (τ) dan jika torsi yang dihasilkan lebih besar dari torsi beban, maka *rotor* akan berputar dengan kecepatan yang searah dengan medan putar *stator*. Kecepatan sinkron dari medan putar (n) yang dihasilkan oleh motor induksi bergantung pada banyaknya jumlah kutub (p)dan besarnya frekuensi (f) yang mengalir melewati *stator*.

$$n_s = \frac{120\,x\,fs}{p}\ (rpm)$$

Keterangan:

n_s : Kecepatan sinkron (rpm)

fs : Frekuensi stator (Hz)

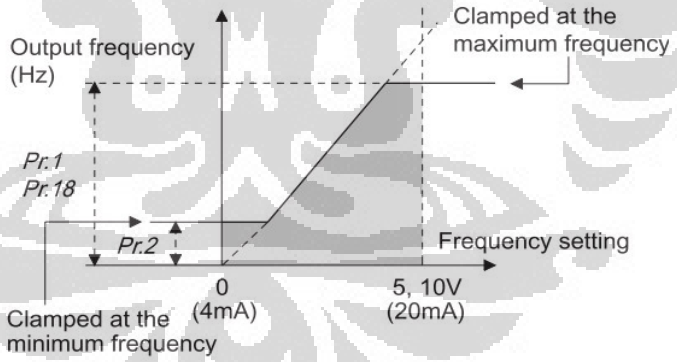
p : Banyaknya kutub

2.3 Pengenalan *Inverter / Variable Frequency Drive*

Inverter adalah sebuah alat yang mengubah listrik arus searah (DC) menjadi arus bolak – balik (AC). Dengan menggunakan transformator, switching, serta rangkaian kontrol tegangan dan frekuensi yang dapat diatur sesuai dengan kebutuhan. Dikarenakan hasil yang didapatkan berupa tegangan atau frekuensi yang dapat diatur, maka inverter dapat diaplikasikan sebagai pengatur kecepatan rotasi sebuah motor listrik AC. Aplikasi ini dikenal dengan sebutan *Variable Frequency Drive* (VFD) atau *Inverter Drive.*

Agar inverter dapat bekerja seusai dengan deskripsi yang diinginkan oleh *user* maka dibutuhkan *parameter setting*. *Parameter setting* berfungsi sebagai batasan operasional pengaman inverter serta mode operasional inverter. Beberapa parameter, diantaranya: Batasan frekuensi maximum, Batasan frekuensi minimum, Mode pengoperasian (*network, parameter unit*, eksternal), dll.

1. ***Maximum Frequency (Parameter No. 1)*** : Batasan nilai frekuensi maksimum untuk motor, misal setting parameter 1 = 60 Hz, maka motor dapat berputar pada kecepatan maksimum 60 Hz.

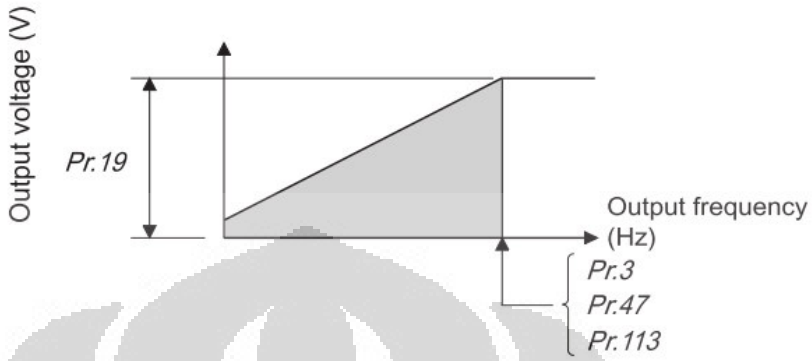


Gambar 2.9 *Maximum/minimum frequency (parameter no. 1)*

Gambar 2.9 menggambarkan batasan frekuensi minimum, maksimum yang dapat dilayani oleh *inverter* serta hubungan antara parameter no. 1, 2, dan 18.

2. ***Minimum Frequency (Parameter No. 2)*** : Batasan nilai frekuensi minimum untuk motor.

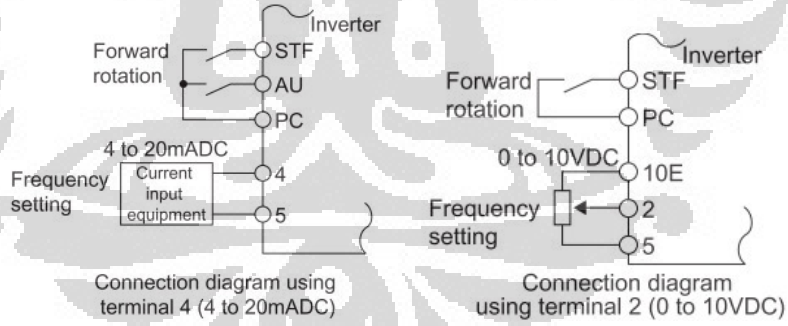
3. ***Base Frequency (Parameter No. 3)*** : Batasan tegangan kerja motor ketika frekuensi maksimum, misal parameter 3 = 60 Hz m maka ketika motor pada kecepatan 60 Hz tegangan pada motor = 380 V.



Gambar 2.10 *Base frequency (parameter no. 3)*

Gambar 2.10 menggambarkan batasan frekuensi dasar yang digunakan sebagai referensi oleh *inverter* serta hubungan antara parameter no. 3, 19, 47, dan 113.

4. ***Analog Input Selection (Parameter No. 73)*** : Mode Pengoperasian eksternal untuk memilih jenis sinyal analog yang akan digunakan, analog tegangan (0 – 10 Vdc) atau analog arus (4 – 20 mA).



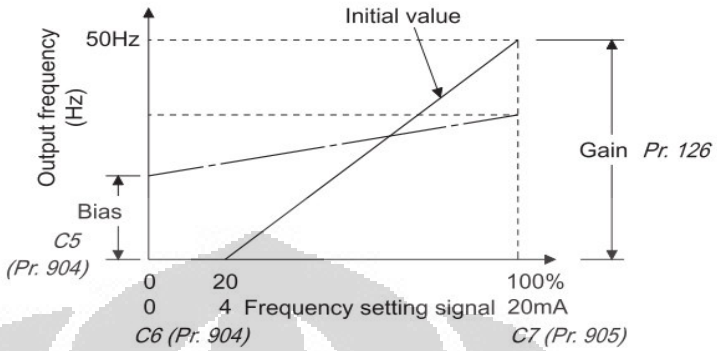
Gambar 2.11 *Analog input selection (parameter no. 73)*

(a). Gambar *Analog* arus (4-20 mA)

(b). Gambar *Analog* tegangan (0-10 Vdc)

Gambar 2.11(a) menggambarkan diagram pengkawatan *inverter* bila *input* eksternal yang digunakan adalah sinyal *analog* berupa arus (4 – 20 mA), Gambar 2.11(b) menggambarkan diagram pengkawatan *inverter* bila *input* yang digunakan adalah sinyal *analog* berupa tegangan (0 – 10 Vdc).

5. ***Frequency Set V Bias / Gain (Parameter No. 903 to 906)*** : Batasan frekuensi minimum dan maximum untuk sinyal analog (4-20 mA) atau (0-10 VDC).



Gambar 2.12 *Frequency set V bias/gain (parameter no. 903 to 906)*

Gambar 2.12 menggambarkan batasan kontrol *bias* dan *gain* frekuensi *inverter* untuk *input* berupa sinyal *analog* serta hubungan antara parameter no. 126, 904, dan 905.

2.4 PLC (*Programmable Logic Controller*)

Secara definisi, PLC (*Programmable Logic Controller*) adalah suatu alat yang dapat diprogram secara logic dan berfungsi untuk mengontrol bermacam macam mesin melalui unit input dan output. Struktur PLC sendiri terdiri dari beberapa bagian, yaitu :

a. Antarmuka / *interface input* **:** Berfungsi untuk menerima sinyal input yang berasal dari luar seperti sensor, *push button*, dll. Sinyal biasanya berupa tegangan DC maupun tegangan AC.

b. Antarmuka / *interface output* **:** Berfungsi sebagai sinyal keluaran dari hasil proses CPU. Interface output terdiri dari beberapa type seperti *Relay, Transistor, dan Triac.*

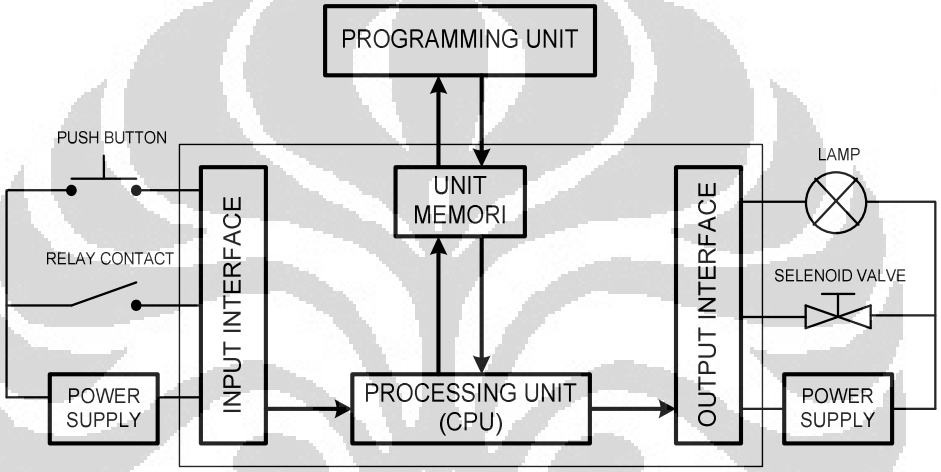
c. *Processing unit* (CPU) **:** CPU dapat dianggap sebagai otak dari PLC yang Berfungsi sebagai pemroses sinyal input sesuai program pada unit memori dan hasilnya dikeluarkan melalui interface output.

d. Unit memori **:** Unit memori berfungsi sebagai tempat untuk menyimpan program. Beberapa tipe memori yang tersedia antara lain *RAM, ROM, EPROM, dan EEPROM.*

e. *Power Supply* **:** Sebagai tegangan sumber untuk menjalankan atau mengoperasikan PLC. Tegangan sumber untuk PLC sendiri bisa memilih tegangan AC dengan rating 100VAC - 240 VAC dan tegangan DC yaitu 24 VDC

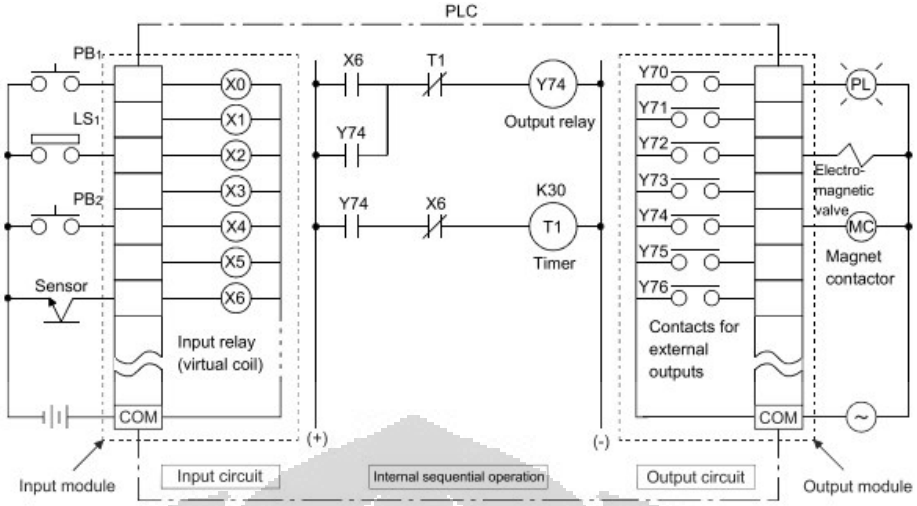
2.4.1 Prinsip Kerja PLC

Dari penjelasan sebelumnya bahwa prinsip kerja dari PLC dapat memenuhi kriteria struktur diatas. Namun dari blok dasar sistem kontrol, prinsip kerja PLC dapat dikembangkan menjadi sebuah struktur seperti gambar berikut.



Gambar 2.13 Blok diagram dasar PLC [4]

Dari Gambar 2.13 ada tiga bagian penting dari prinsip kerja PLC, ketiga bagian tersebut dipengaruhi faktor eksternal seperti push button mewakili bagian dari sebuah *input interface*, *solenoid valve* mewakili bagian dari *output interface,* sedangkan untuk programming unit atau biasa disebut dengan CPU dipengaruhi oleh media pemrograman, hal ini bertujuan agar PLC tersebut sesuai dengan deskripsi kerja yang diinginkan. Proses keseluruhan ketika CPU sudah diprogram dapat dilihat pada gambar berikut.



Gambar 2.14 Konfigurasi lengkap PLC [4]

Pada Gambar 2.14, sinyal *input* diwakili oleh dua buah *push button* (X0, X4), *limit switch* (X2), dan *sensor* (X6). Sedangkan untuk sinyal *output* diwakili oleh *pilot lamp* (Y70), *electromagnetic valve* (Y72), dan *electromagnetic contactor* (Y74). Penjelasan program ladder diatas yaitu ketika sensor (X6) aktif, *electromagnetic contactor* (Y74) akan menyala dan *Timer* no. 1 akan menghitung selama 30 ms, saat *timer* sudah mencapai 30 ms akan menon-aktifkan *electromagnetic contactor.*

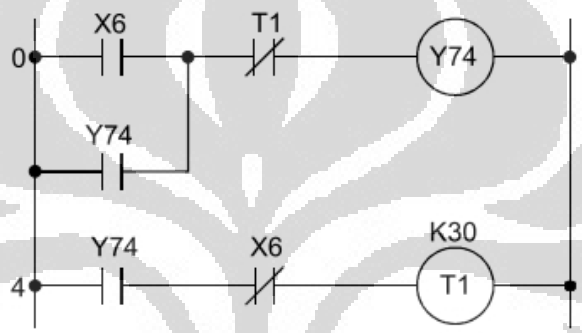
2.4.2 Pemrograman PLC Mitsubishi GX-Developer

Pemrograman PLC pada tipe PLC Mitsubishi *Q series* ***d***apat dilakukan dengan berbagai macam bahasa pemrograman seperti *Instruction List, Sequencial Function Chart, Ladder Diagram,* dan *Function Block Diagram*,. Namun untuk kawasan asia khususnya di Indonesia lebih populer dengan menggunakan bahasa pemrograman *Ladder Diagram.* Untuk sistem yang akan dibuat, penulis menggunakan tipe pemrograman bahasa *Ladder Diagram* untuk membentuk sistem *fuzzy*. Keuntungan dengan menggunakan bahasa pemrograman Ladder Diagram diantaranya :

1. Cepat dan mudah dimengerti karena mirip dengan sistem logika rangkaian konvensional.
2. Mudah dalam penggambaran rangkaian control sekalipun sistem tersebut cukup rumit.

3. Sangat fleksibel, karena semua fungsi serta instruksi program seperti instruksi fungsi logika, fungsi aritmatika, fungsi timer, fungsi counter dan fungsi – fungsi lainnya dapat dibuat dengan sistem ladder diagram, sehingga dapat diedit dan dipindahkan ke *ladder diagram* yang lainnya.
4. Dapat dengan mudah memonitor dan mengevaluasi kerja dari rangkaian kontrol pada saat dieksekusi maupun pada saat *troubleshooting*.

Gambar 2.15, merupakan contoh pemrograman dengan menggunakan bahasa program *ladder* untuk jenis PLC Mitsubishi *Q series*.



Gambar 2.15 Contoh bahasa pemrograman *ladder diagram* [4]

Keterangan :

- X6 = *Input*
- T1 K30 = *Timer* no. 1 dengan konstanta 30 ms
- Y74 = *Output*

2.4.3 Pengalamatan PLC Mitsubishi *Q Series*

Untuk memulai merancang suatu program, penting mengetahui modul apa saja yang akan digunakan serta alamat modul yang terpasang pada PLC Mitsubishi. Hal tersebut berguna agar program yang dibuat dapat berjalan dengan baik. Pengalamatan pada PLC *Q series* menggunakan sistem bilangan Hexadesimal, yaitu sistem bilangan yang mempunyai jumlah bilangan sebanyak 16, dimulai dari 0, 1,2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. Dan berikut cara pengalamatan PLC Q series :

1. Pengalamatan berdasarkan bilangan Hexadesimal.
2. Pengalamatan dimulai dari *slot modul* yang terdekat dengan CPU (slot 0).

3. Tipe pengalamatan bersifat kontinyu, dimana alamat yang sudah digunakan tidak bisa digunakan lagi.
4. Untuk *extension base modul*, alamatnya mengikuti alamat terakhir pada *main base modul.*
5. Modul kosong di alamati sebanyak 16 point.

2.4.4 Pengenalan Divais Dan Instruksi PLC Mitsubishi *Q Series*

Pada semua bahasa pemrograman PLC dengan brand apapun pasti terdapat notasi untuk mengindentifikasi sinyal – sinyal yang diproses dalam PLC.dimana notasi antar brand pasti memiliki perbedaan, hal ini mungkin bertujuan sebagai identitas dari brand masing – masing. Setiap notasi pada PLC Q series yang digunakan mempunyai fungsi dan jenis program yang berbeda tergantung deskripsi yang dibutuhkan. Berikut beberapa notasi pada PLC Q series:

1. ***Input*** **(X)**

   Untuk interface sinyal input pada PLC. Bentuk dari divais ini bisa berupa NC (*Normally Close*) dan NO (*Normally Open*). Untuk inisialisasi *input* menggunakan bilangan Hexadesimal, contoh X0, X1, X2,…, X9, XA,…, XF. Contoh penggunaan *input* (X) pada pemrograman dengan menggunakan diagram *ladder* dapat dilihat pada Gambar 2.16.

Gambar 2.16 Contoh intruksi *input* [4]

Keterangan :

- X0 = *input* NO (*Normally Open*)
- X1 = *input* NC (*Normally Close*)
- Y10 = *output*

Deskripsi kerja :

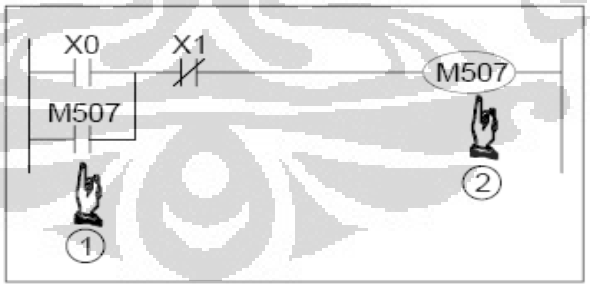
- X0 ditekan = Y10 “On”
- X1 ditekan = Y10 “Off”

2. *Output* (Y)

Mewakili kondisi sinyal *output* yang terhubung pada PLC yang memiliki simbol notasi Y. Bentuk dari divais ini yaitu *output coil*, NO, dan NC. Inisialisai pada divais *output* sama seperti divais *input* yaitu dengan menggunakan sistem bilangan Hexadesimal. Dan contoh programnya sama seperti contoh intruksi *input* pada Gambar 2.16.

3. *Internal Relay* (M)

Fungsi divais *internal relay* adalah sebagai *relay* bantu pada PLC. Bentuk dari divais ini yaitu. *output coil,* NO, dan NC. Sedangkan untuk tipe *internal relay* yang menggunakan notasi SM yaitu: tipe *special relay* untuk berbagai macam kebutuhan dan untuk bentuknya hanya terdiri dari NO dan NC. Sedangkan alamat yang digunakan adalah bilangan desimal. Berikut contoh dari aplikasi *internal relay*. Contoh penggunaan *internal relay* (M) pada pemrograman dengan menggunakan diagram *ladder* dapat dilihat pada Gambar 2.17.



Gambar 2.17 Contoh aplikasi *internal relay* [4]

Keterangan :

- X0 = *Input* “On”
- X1 = *Input* “Off”
- M507 = *Internal Relay*
-

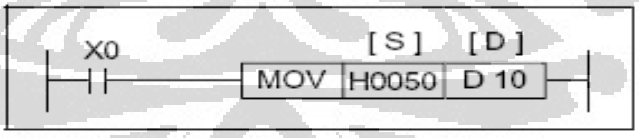
Deskripsi Kerja :

- X0 ditekan = M507 “On”
- X0 dilepas, M507 masih “On” (M507 penguncian / *self locking relay*)
- X1 ditekan = M507 “Off” (kembali ke awal)

4. Konstanta Desimal dan Hexadesimal (K , H)

PLC Mitsubishi mempunyai 2 jenis sistem bilangan yaitu Desimal dengan menggunakan notasi K dan Hexadesimal dengan menggunakan notasi H. Untuk aplikasi penggunaan kontanta desimal maupun hexadesimal terdapat salah satunya pada timer. Dimana nilai konstanta dibutuhkan sebagai nilai seting aktif *timer* (“On” *delay*). Bentuk inisialisasi konstanta desimal yaitu Kn dimana nilai n untuk menentukan besarnya nilai K (Gambar 2.18(a)), sedangkan untuk konstanta hexadesimal yaitu Hn dimana nilai n untuk menentukan besarnya nilai H (Gambar 2.18(b)).

(a)



(b)

Gambar 2.18 Aplikasi konstanta didalam pemrograman *ladder*

(a). Gambar Konstanta desimal

(b). Gambar Konstanta heksadesimal

Keterangan (Gambar 2.18(a)):

- X0 = *input*
- T20 K123 = *Timer* no. 20 dengan nilai konstanta desimal 123 ms

Keterangan (gambar 2.18(b)):

- X0 = *input*
- MOV H0050 D10 = Instruksi memindahkan nilai konstanta Hexadesimal 50 ke alamat data D10

5. Data *Register* (D)

Divais data *register* berfungsi sebagai tempat penyimpan data dengan konstanta desimal maupun hexadesimal. Alamat penggunaan data register berdasarkan bilangan desimal, yaitu: D0, D1, D2, ..., D9, D10, D11, dan seterusnya. Sedangkan bentuk nilai data *register* ini untuk 1 *word* (1 nilai D) yaitu: 16 bit (-32768 s.d 32767 atau 0 s.d FFFF). Untuk data register *double word*, yaitu: 32 bit (-2147483648 s.d 2147483647 atau 0 s.d FFFFFFFF).

6. *Timer* (T)

Divais *timer* berfungsi untuk menunda aktif ("On" *delay*) atau menunda mati ("Off" delay) suatu *relay* ataupun *output.* Bentuk dari timer yaitu: *output coil*, dan kontaknya dapat berupa NO dan NC. Alamat penggunaan *timer* berdasarkan bilangan desimal. *Timer* mempunyai besaran waktu K1 untuk 1/10 detik atau 10 ms, sehingga kita dapat mensetting nilai besaran tersebut dengan faktor pengali dari konstanta K. Selain konstanta ketetapan waktu, *timer* juga dapat menggunakan data secara *indirect*. Contoh penggunaan instruksi *timer* pada pemrograman dengan menggunakan diagram *ladder* dapat dilihat pada Gambar 2.19.

Gambar 2.19 Instruksi *timer* [4]

Pada Gambar 2.19 untuk line 1 (rung 1), T0 (*coil*) adalah alamat *timer* no. 0, sedangkan untuk K10 merupakan nilai konstanta 10 ms. Untuk rung 5, T0 merupakan kontak (NO / NC).

Keterangan :

- X0 = *input*
- T0 = *timer* no. 0 dengan konstanta waktu 10 ms
- Y10 = *output*

Deskripsi :

- X0 ditekan = *timer* no.0 menghitung waktu selama 10 ms
- T0 (10 ms) = *output* Y10 “On”

7. *Increment* (INC / INCP) dan *Decrement* (DEC / DECP)

Instruksi INC dan DEC berfungsi sebagai perintah untuk menambah (INC) atau mengurangi (DEC) 1 pada suatu nilai data (word). Bentuk dari *increment* dan *decrement* yaitu: instruksi. Contoh penggunaan instruksi INC/DEC pada pemrograman dengan menggunakan diagram *ladder* dapat dilihat pada Gambar 2.20.

Gambar 2.20 Instruksi INCP/DECP [4]

Pada rung nomor satu Gambar 2.20, sinyal X0 ketika “On” akan menambah nilai data *register* D0 dari 0 menjadi 1, dan jika X0 diaktifkan untuk kedua kalinya maka nilai D0 akan bertambah menjadi 2, dan begitu seterusnya. Untuk DECP, nilai data D0 akan berkurang 1 ketika sinyal X1 aktif dan Untuk tipe data ini yaitu 16 bit (-32768 s.d 32767).

Keterangan :

- X0 = *input*
- X1 = *input*
- INCP = instruksi menaikkan nilai data (D0, D1, D2,...)
- DECP = instruksi menurunkan nilai data (D0, D1, D2,...)

8. Transfer Data (MOV)

Instruksi MOV berfungsi sebagai perintah untuk untuk memindahkan data atau nilai konstanta ke alamat data tujuan. Bentuk dari perintah MOV yaitu: instruksi. Contoh penggunaan instruksi MOV pada pemrograman dengan menggunakan diagram *ladder* dapat dilihat pada Gambar 2.21.

Gambar 2.21 Instruksi MOV [4]

Keterangan :

- X0 = *input*
- MOV K10 D0 = instruksi memindahkan nilai konstanta 10 ke alamat data D0.
- MOV D0 D1 = instruksi memindahkan nilai data D0 ke alamat data D1.

9. Konversi Nilai Integer ke Float (FLT)

Instruksi FLT berfungsi sebagai perintah untuk mengkonversi nilai dari format *integer* ke dalam format *float*. Bentuk dari perintah FLT yaitu: instruksi. Contoh penggunaan instruksi FLT pada pemrograman dengan menggunakan diagram *ladder* dapat dilihat pada Gambar 2.22.

Gambar 2.22 Instruksi konversi nilai format *integer* ke *float* [4]

Keterangan :

- X0 = input
- FLT D0 D1 = instruksi konversi nilai format *integer* pada alamat data D0 ke format *float* pada alamat data D1.

10. Konversi Nilai Float ke Integer (INT)

Instruksi INT merupakan kebalikan dari instruksi FLT, berfungsi sebagai perintah untuk konversi nilai dari fomat *float* ke dalam format *integer*. Bentuk dari perintah FLT, yaitu: instruksi. Contoh penggunaan instruksi INT pada pemrograman dengan menggunakan diagram *ladder* dapat dilihat pada Gambar 2.23.

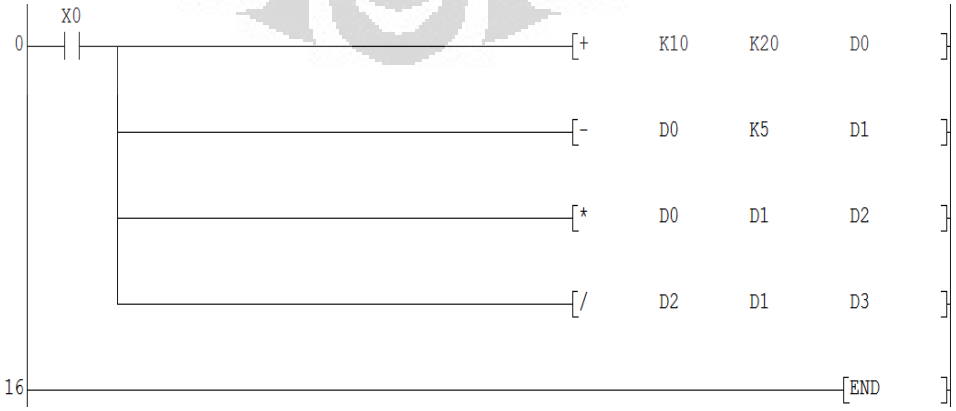
Gambar 2.23 Instruksi konversi nilai format *float* ke *integer* [4]

Keterangan :

- X0 = *input*
- INT D0 D1 = instruksi konversi nilai format *float* pada alamat data D0 ke nilai format *integer* pada alamat data D1.

11. Instruksi Aritmatika (* , / , - , +)

Instruksi aritmatika berfungsi sebagai perintah untuk mengkalkulasi (* , / , + , -) suatu nilai dalam bentuk data. Untuk inisialisasi nilai yang akan di kalkulasi berupa konstanta dan bisa juga data *register*, dimana data *register* tersebut sudah mempunyai nilai. Bentuk dari perintah fungsi aritmatika ini berupa instruksi. Contoh penggunaan instruksi aritmatika pada pemrograman dengan menggunakan diagram *ladder* dapat dilihat pada Gambar 2.24.



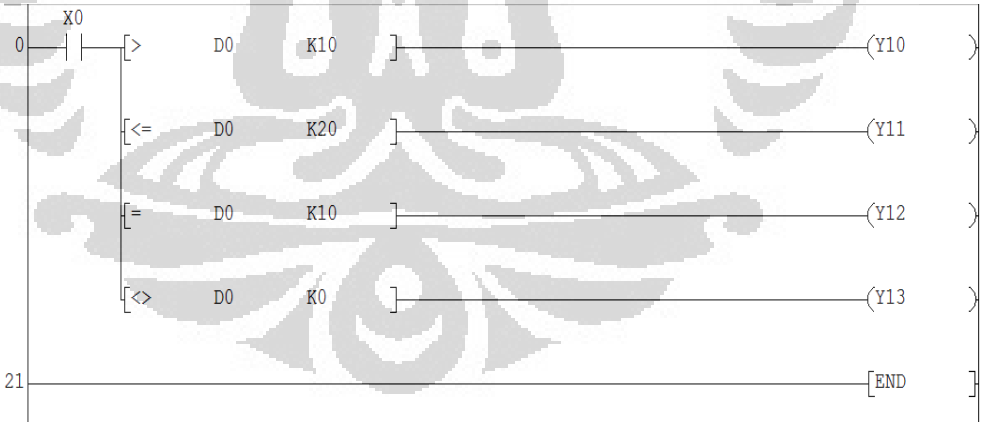
Gambar 2.24 Instruksi aritmatika (*, /, +, -) [4]

Keterangan :

- X0 = *input*
- [+ K10 K20 D0] = menjumlahkan nilai desimal 10 dengan desimal 20, dan hasilnya disimpan ke dalam alamat data D0.
- [- D0 K5 D1] = mengurangi nilai pada alamat D0 dengan nilai desimal 5, dan hasilnya disimpan ke dalam alamat data D1.
- [* D0 D1 D2] = mengalikan nilai pada alamat D0 dengan nilai pada alamat D1, dan hasilnya disimpan ke dalam alamat data D2.
- [/ D2 D1 D3] = membagi nilai pada alamat D2 dengan nilai pada alamat D1, dan hasilnya disimpan ke dalam alamat data D3.

12. Instruksi Perbandingan (< , > , = , <= , >= , < >)

Instruksi perbandingan berfungsi sebagai syarat/kondisi untuk membandingkan suatu nilai ataupun data baik secara *direct* maupun *indirect*. Inisialisasi fungsi dari perbandingan ini dapat berupa nilai konstanta ataupun data register. Fungsi perbandingan berbentuk instruksi. Contoh penggunaan instruksi perbandingan pada pemrograman dengan menggunakan diagram *ladder* dapat dilihat pada Gambar 2.25.



Gambar 2.25 Instruksi perbandingan (< , > , = , <= , >= , < >)

Keterangan :

- X0 = *input*
- Y10, Y11, Y12, Y13 = *output*
- [> D0 K10] = jika nilai pada alamat D0 lebih dari 10 maka *output* Y10 aktif.

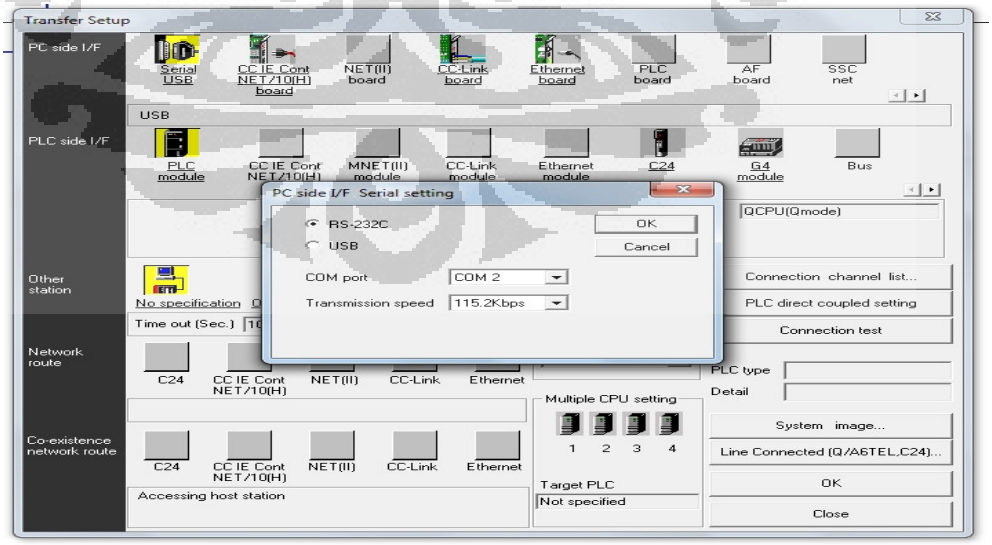
- [<= D0 K20] = jika nilai pada alamat D0 kurang dari sama dengan 20 maka *output* Y11 aktif.
- [= D0 K10] = jika nilai pada alamat D0 sama dengan 10 maka *output* Y12 aktif.
- [< > D0 K0] = jika nilai pada alamat D0 tidak sama dengan 0 maka *output* Y13 aktif.

2.4.5 Transfer Data PLC Mitsubishi *Q Series*

Setelah program selesai dibuat maka program tersebut harus dimasukkan ke dalam memori PLC, hal ini dibutuhkan suatu transfer data dari PC ke memori PLC. Dalam proses transfer data dibutuhkan beberapa komponen untuk melakukan perintah ini seperti kabel transfer data. Untuk tipe PLC yang digunakan pada penulisan ini yaitu PLC Q02H, dimana PLC tersebut mempunyai 2 pilihan kabel komunikasi untuk transfer data, seperti QC30R2 dan USB 2.0. Dan berikut ada 3 tipe transfer data, yaitu :

1. ***Transfer Set Up***

   Transfer set up merupakan suatu metoda untuk melakukan check komunikasi antara PLC dan PC, dan juga pemilihan kabel komunikasi yang digunakan. Dimana hal ini sangat penting sebagai langkah pertama untuk proses *upload* dan *download*.

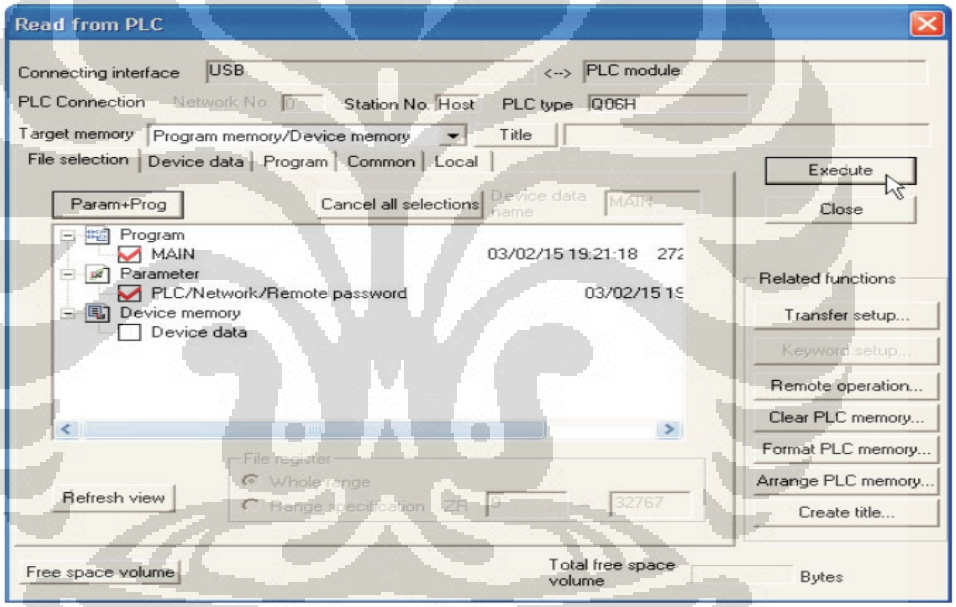


Gambar 2.26 Proses *transfer set up*

Gambar 2.26 menunjukkan pemilihan tipe kabel yang akan digunakan sebagai media komunikasi, yaitu: RS-232C (kabel QC30-R2) atau USB. Jika kabel QC30-R2 yang digunakan maka *user* harus memilih port komunikasi serial (COM 0-40) dan juga kecepatan transmisi data (*baudrate).*

2. *Read*

Read atau bisa disebut dengan *upload,* dimana suatu perintah untuk membaca program PLC ke PC. Data yang ditarik dapat berupa program dan parameter, namun juga bisa *comment* jika PLC tersebut mimiliki *comment.*

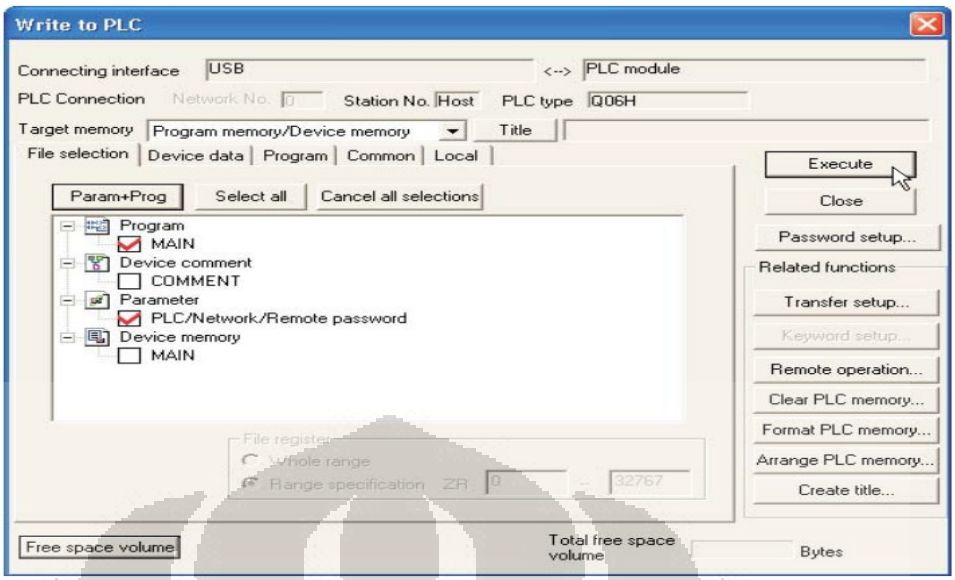


Gambar 2.27 Proses membaca program (*upload*) dari PLC ke PC

Berdasarkan Gambar 2.27, data program yang ditarik antara lain program dan parameter yang diberi cek list merah. Ketika sudah selesai dengan pemilihan data yang akan dibaca, maka langkah selanjutnya yaitu menekan tombol ***Execute***.

3. *Write*

Write atau biasa disebut dengan *download*, dimana suatu perintah untuk menulis program dari PC ke PLC. Pada gambar 2.46, program yang ditransfer ke PLC yaitu program dan parameter yang diberi cek list merah.



Gambar 2.28 Proses memindahkan program (*download*) dari PC ke PLC

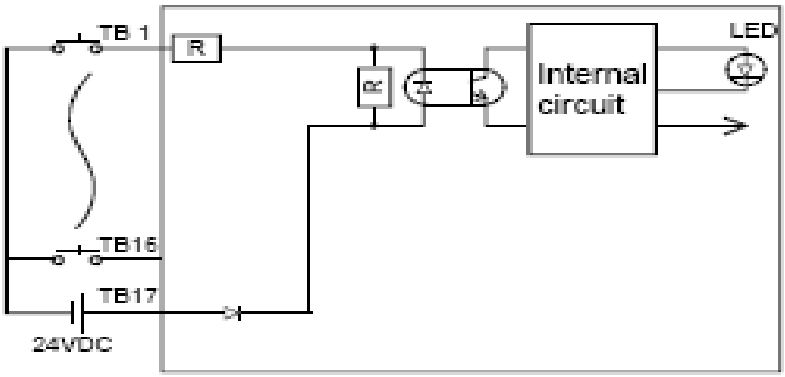
Berdasarkan Gambar 2.28, proses perpindahan program dari PC ke PLC tidak hanya sebatas program dan parameter, namun jika ada tambahan berupa comment maka kolom untuk tulisan COMMENT dapat diberikan cek list lalu kemudian di eksekusi.

2.4.6 Tipe *Wiring Input/Output* PLC

Terdapat dua jenis tipe wiring yang umum digunakan di dalam PLC Mitsubishi, yaitu: *sink* dan *source*. Perbedaan dari kedua tipe *wiring* tersebut adalah polaritas yang masuk ke dalam *common* dari *interface input* ataupun *output*.

1. ***Sink Type***

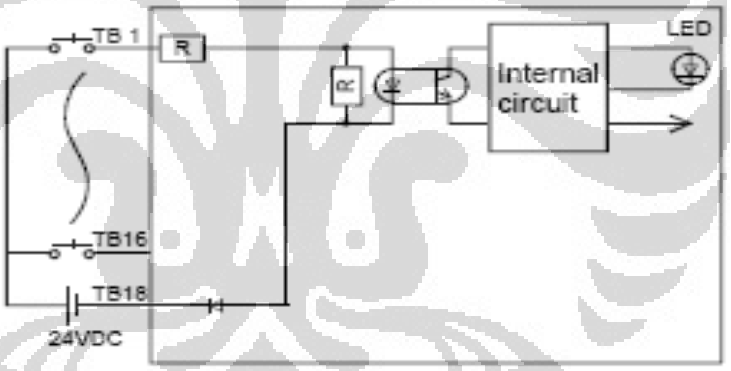
   Pada tipe *sink*, sinyal kabel *input* mendapat tegangan negatif atau 0V (TB1 – TB16), dan *common* mendapat tegangan 24V (TB17) , biasanya tipe *wiring* ini digunakan jika divais *input* yang dipakai berjenis NPN. Contoh *wiring* untuk *sink type* dapat dilihat pada Gambar 2.29.



Gambar 2.29 Tipe *wiring sink, common* (+) [4]

2. Source Type

Pada tipe *source*, sinyal kabel *input* mendapat tegangan positif atau 24 V (TB1 – TB16), dan common mendapat tegangan negatif atau 0V (TB18) , biasanya tipe wiring ini digunakan jika device input yang dipakai berjenis PNP. Contoh *wiring* untuk *source type* dapat dilihat pada Gambar 2.30.

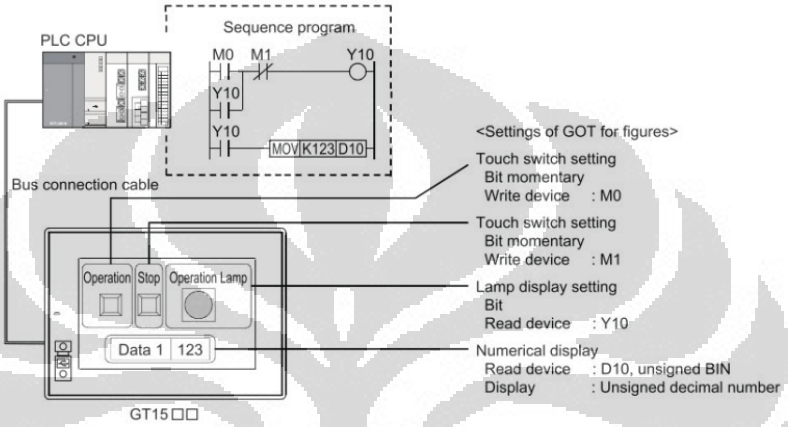


Gambar 2.30 Tipe *wiring source*, c*ommon* (-) [4]

2.5 SoftGOT1000 HMI (*Human Machine Interface*)

Human Machine Interface merupakan piranti yang berfungsi sebagai media komunikasi antara *operator* dengan mesin, dalam hal ini PLC. SoftGOT1000 merupakan modul HMI lisensi Mitsubishi Electric non *touchscreen*. Cara penggunaannya ialah dengan meng*install software* SoftGOT1000 ke dalam *pc desktop* ataupun *laptop*. Untuk operasional object SoftGOT1000 sama persis dengan model GOT (*Graphic Operation Terminal*) *touchscreen*, hanya saja diperlukan *dongle key* berbentuk USB *flashdisk* agar bisa *runtime infinite*. Penggunaan tanpa *dongle key* hanya bisa berjalan selama 2 jam.

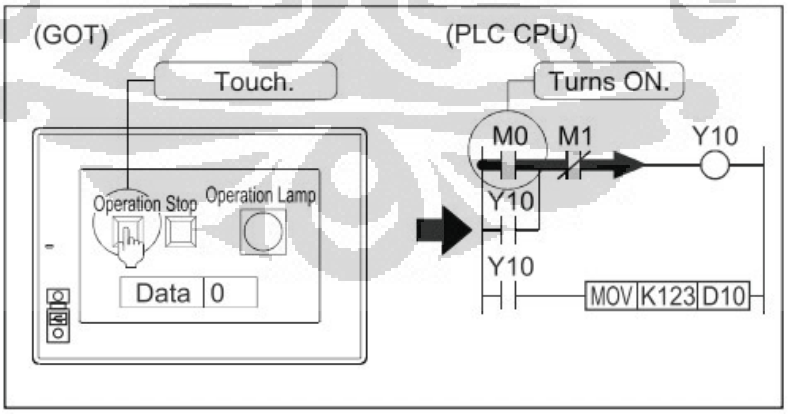
Semua object pada SoftGOT1000 / GOT bekerja dengan cara memonitor perubahan status/nilai data dari divais-divais PLC dalam bentuk *bit* ataupun *register* kemudian ditampilkan kedalam bentuk object lampu, *push button*, *numerical input/diplay*, *level*, *comment*, grafik, dll. *Software* yag digunakan untuk mendisain tampilan HMI SoftGOT1000 / GOT adalah lisensi Mitsubishi Electric dengan tipe GT-Designer3 v1.10L (SW1DNC-GTWK3-E).



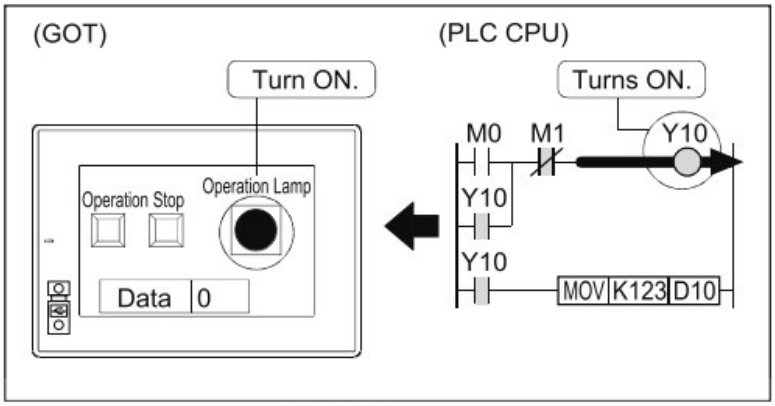
Gambar 2.31 Konfigurasi HMI GOT sistem *Bus*

Berdasarkan Gambar 2.31 komunikasi modul HMI dengan modul kontroller menggunakan media koneksi kabel BUS CPU yang terhubung dengan adaptor komunikasi BUS pada modul HMI.

Cara kerja HMI GOT dapat dilihat pada gambar berikut.



(a)



(b)

Gambar 2.32 Operasional GOT

(a). Gambar Sebelum eksekusi

(b). Gambar Setelah eksekusi

Pada Gambar 2.32(a), ketika *operation push button* ditekan dari HMI maka pada program *ladder*, M0 akan aktif. Dan pada Gambar 2.32(b) dapat dilihat bahwa *operation lamp* pada HMI akan "On", sedangkan pada program ladder, Y10 "On". Tampilan pada HMI tidak semata hanya bentuk *switch* dan *lamp,* tapi juga terdapat beberapa *object* seperti grafik, *level*, *alarm history, numerical display, numerical input, date display, panel meter,* dan masih banyak *object* lainnya.

2.5.1 Sistem Konfigurasi HMI SoftGOT1000

Pada awal mendisain tampilan HMI maka terlebih dahulu mengetahui konfigurasi sistem HMI, agar sistem tersebut dapat berjalan sesuai deskripsi. Yang dimaksud sistem konfigurasi HMI SoftGOT1000 adalah keseluruhan sistem yang terintegrasi dengan HMI SoftGOT1000, yaitu: *system setting* dan *communication setting*.

1. ***System Setting***

   System setting adalah parameter untuk menentukan ukuran resolusi screen dari tampilan SoftGOT1000, biasanya disesuaikan dengan ukuran *screen pc desktop* ataupun *laptop*. Sebagai contoh: untuk screen laptop 14" *wide*, maka ukuran resolusi *screen* dari SoftGOT1000 adalah 1366 x 768

pixel. *User* juga mempunyai pilihan untuk warna yang terdiri dari 16 warna, 256 warna, 65536 warna.

2. Communication Setting

Communication setting adalah pemilihan jenis kontroller yang akan digunakan sebagai acuan pengendali dari SoftGOT1000 tersebut. Untuk pemilihan kontroller ini tidak harus dengan *brand* Mitsubishi, kompatibel juga dengan *brand* yang berbeda, seperti: Omron, Siemens, Allan Bradley, Yokogawa, dll. Setelah jenis kontroller maka langkah selanjutnya dari *communication setting* adalah menentukan tipe *Interface* untuk komunikasi yang akan digunakan, yaitu: Ethernet, USB, RS232, dan RS422/485.

2.5.2 Parameter Disain *Software* GT-Designer3 v1.10L

Untuk mendisain tampilan pada SoftGOT1000, maka perlu mengetahui parameter-parameter yang berhubungan dengan disain tampilan SoftGOT1000 yang telah disediakan oleh software GT-Designer3 v1.10L. Untuk tampilan yang ada pada GT-Designer v1.10L diantanya adalah :

1. Object

Pilihan *object* pada *software* GT-Designer3 v1.10L beragam, antara lain: jenis saklar *push button*, lampu indikator, *level display, numerical display, numerical input, alarm history,* grafik, dll.

2. Common

Common berfungsi untuk menampilkan fungsi spesial dari objek. Contoh: jika ingin menampilkan *Historical Time Graph* maka sebelum itu perlu dibuat *setting parameter logging* yang terdapat pada *common*.

3. Figure

Untuk membuat garis, lingkaran ataupun bentuk lainnya maka figure dapat digunakan. Selain itu, import gambar dapat dilakukan dalam

bentuk file: *.bmp, *.jpg, *.jpeg, dan *.jpe. Untuk menambahkan text pada object dapat dilakukan dengan menggunakan fungsi text. Fungsi text memiliki kompatibilitas dengan tipe-tipe font yang terdapat pada microsoft word.

2.5.3 Transfer Data *Software* GT-Designer3 v1.10L

Fungsi transfer data pada *sofware* GT-Designer3 v1.10L adalah untuk memindahkan disain tampilan GT-Designer3 v1.10L ke dalam *runtime* SoftGOT1000**.** Berikut bagian penting dari transfer data :

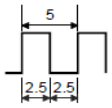
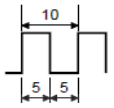
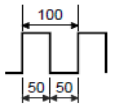
1. **Tes komunikasi**: sebagai pengechekan antara pc dengan kabel komunikasi
2. **Operating System**: sebagai langkah awal operating system untuk GOT agar bisa GOT tersebut menampilkan objek gambar yang telah dibuat di pc
3. **Project Download to SoftGOT1000:** sebagai transfer data, dimana design gambar yang sudah dibuat di pc akan dipindahkan ke GOT
4. **Project Upload to Computer**: sama sperti di atas hanya saja tujuan yang berbeda, dimana data gambar yang ada di GOT dipindahkan ke pc.

2.6 *Interface Module*

2.6.1 *High Speed Counter (QD62)*

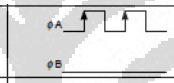
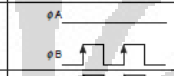
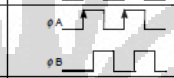
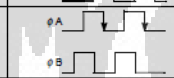
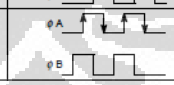
High speed counter modul merupakan suatu modul yang dapat membaca perubahan masukan digital dengan cara *increment* atau *decrement* dalam frekuensi yang tinggi. Frekuensi maksimum masukan yang dapat dibaca mencapai 200 *kilo pulse per seconds*. Pembacaan *input* yang berbentuk pulsa memiliki range antara -2147483648 hingga 2147483647. Spesifikasi dari *high speed counter module* dapat dilihat pada tabel dibawah ini:

Tabel 2.1 Spesifikasi modul *high speed counter (QD62)*

Item \ Model name		QD62		
Counting speed switch settings ＊1		200 k (100 k to 200 kPPS)	100 k (10 k to 100 kPPS)	10 k (10 kPPS or less)
I/O occupied points		16 points (I/O assignment: Intelligent 16 points)		
Number of channels		2 channels		
Count input signal	Phase	1-phase input, 2-phase input		
	Signal level (ϕA, ϕB)	5/12/24 V DC 2 to 5 mA		
Counter	Counting speed (max) ＊2	200 kPPS	100 kPPS	10 kPPS
	Counting range	32-bit signed binary values (–2147483648 to 2147483647)		
	Model	UP/DOWN Preset counter + Ring counter function		
	Minimum count pulse width (μs) (Duty ratio 50 %)	 (Min. phase differential for 2-phase input: 1.25 μs)	 (Min. phase differential for 2-phase input: 2.5 μs)	 (Min. phase differential for 2-phase input: 25 μs)

Berdasarkan Tabel 2.1, modul *high speed counter (QD62)* memiliki dua *channel* pembacaan pulsa, memiliki tiga variasi pembacaan pulsa (200 kpps, 100 kpps, dan 10 kpps), dan level tegangan dari penerimaan sinyal adalah 5/12/24 Vdc.

Tabel 2.2 Metode *input* pulsa

CW/CCW	For addition count		Count at ϕA rise (↑) ϕB is OFF
	For subtraction count		ϕA is OFF Count at ϕB rise (↑)
2-phase multiple of 1	For addition count		Count at ϕA rise (↑) when ϕB is OFF
	For subtraction count		Count at ϕA fall (↓) when ϕB is OFF
2-phase multiple of 2	For addition count		Count at ϕA rise (↑) when ϕB is OFF Count at ϕA fall (↓) when ϕB is ON
	For subtraction count		Count at ϕA rise (↑) when ϕB is ON Count at ϕA fall (↓) when ϕB is OFF

Berdasarkan Tabel 2.2 metode pembacaan pulsa pada modul *high speed counter (QD62)* dibagi kedalam tiga bagian (CCW, 2-*Phase multiple* 1, dan 2-*Phase multiple* 2).

2.6.2 *Analog Output (Q62DA)*

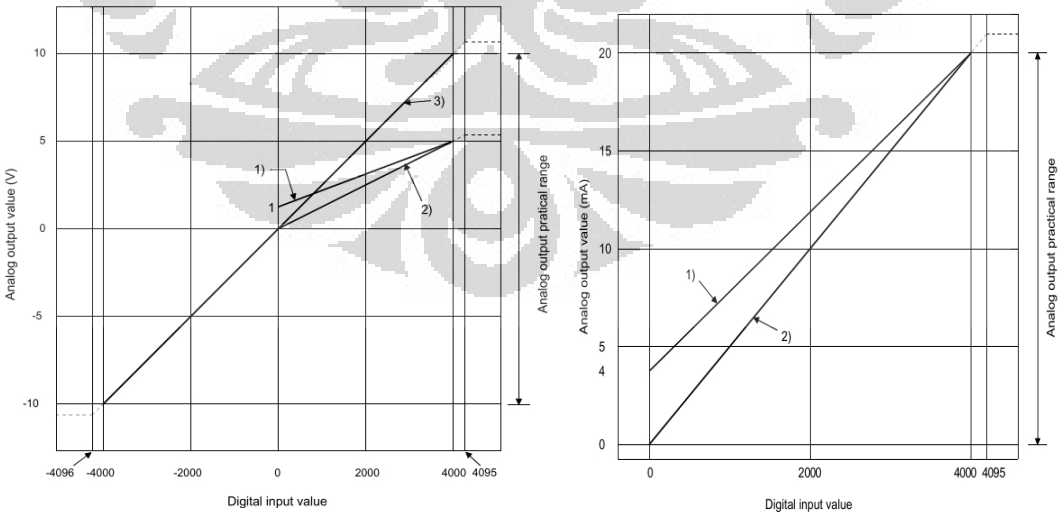
Analog output merupakan suatu modul yang dapat merubah sinyal *output* yang berupa *digital* menjadi besaran tegangan atau arus (*Digital Analog Converter*). Biasanya modul ini digunakan untuk mengatur *linear controlled valve, heater, variable speed drive*, dan lain – lain. Didalam penggunaan modul *analog output,* perlunya diperhatikan yaitu range tegangan/arus, resolusi, dan

scaning time. Sehingga modul ini bisa digunakan sesuai dengan spesifikasi *output* yang akan dikendalikan.

Tabel 2.3 Spesifikasi modul *analog output (Q62DA)*

Item \ Model name		Q62DAN	Q64DAN	Q68DAVN	Q68DAIN
Number of analog output points		2 points (2 channels)	4 points (4 channels)	8 points (8 channels)	
Digital input		16-bit signed binary (normal resolution mode: -4096 to 4095, high resolution mode: -12288 to 12287, -16384 to 16383)			
Analog output	Voltage	–10 to 10 V DC (External load resistance value: 1 kΩ to 1MΩ)			—
	Current	0 to 20 mA DC (External load resistance value: 0Ω to 600Ω)		—	0 to 20 mA DC (External load resistance value: 0Ω to 600Ω)
I/O characteristics, Maximum resolution		Analog output range	Normal resolution mode: Digital input value	Normal resolution mode: Maximum resolution	High resolution mode: Digital input value / Maximum resolution
	Voltage	0 to 5V	0 to 4000	1.25 mV	0 to 12000 / 0.416 mV
		1 to 5V		1.0 mV	0.333 mV
		-10 to 10V	-4000 to 4000	2.5 mV	-16000 to 16000 / 0.625 mV
		User range setting		0.75 mV	-12000 to 12000 / 0.333 mV
	Current	0 to 20 mA	0 to 4000	5 μA	0 to 12000 / 1.66 μA
		4 to 20 mA		4 μA	1.33 μA
		User range setting	-4000 to 4000	1.5 μA	-12000 to 12000 / 0.83 μA
Accuracy (Accuracy in respect to maximum analog output value)	Ambient temperature 25 ± 5 °C	Within ± 0.1 % (Voltage: ±10 mV, Current: ± 20 μA)			
	Ambient temperature 0 to 55 °C	Within ± 0.3 % (Voltage: ± 30 mV, Current: ± 60 μA)			
Conversion speed		80 μs/channel			
Absolute maximum output	Voltage	± 12 V			—
	Current	21 mA		—	21 mA

Berdasarkan Tabel 2.3, modul *digital to analog (Q62AD)* memiliki dua *channel* konversi dari nilai *digital* ke besaran *analog*. Tipe konversi dibagi dua yaitu berupa arus dan tegangan. Untuk arus untuk nilai digital (0 – 4000) dapat dikonversi menjadi (0 – 20 mA) atau (4 – 20 mA). Untuk tegangan nilai digital (0 – 4000) dapat dikonversi menjadi (0 – 5 Vdc) atau (1 – 5 Vdc) sedangkan untuk nilai digital (-4000 – 4000) dapat dikonversi menjadi (-10 – 10 Vdc) semua pilihan tipe konversi dapat diatur melalui *setting* parameter.



(a) (b)

Gambar 2.33 Karakteristik modul *analog output*

(a) Gambar Koversi arus (b) Gambar Konversi tegangan

Gambar 2.33(a) dan 2.33(b) menggambarkan karakteristik konversi *modul digital to analog (Q62DA)* berdasarkan tipe konversi arus atau tegangan. Berdasarkan Gambar 2.33(a) karakteristik konversi *digital to analog* untuk *output* tegangan dibagi tiga, yaitu:

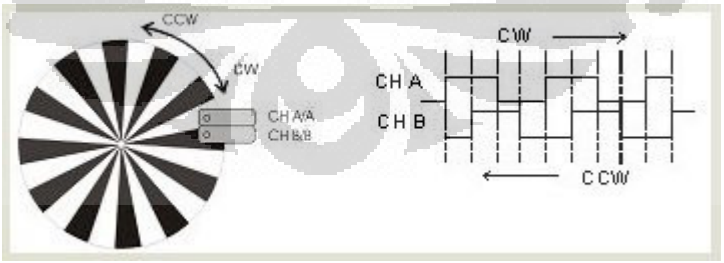
1. *Input* nilai digital (0 – 4000), *output* konversi (1 – 5 Vdc)
2. *Input* nilai digital (0 – 4000), *output* konversi (0 – 5 Vdc)
3. *Input* nilai digital (-4000 – 4000), *output* konversi (-10 – 10 Vdc)

Berdasarkan Gambar 2.33(b) karakteristik konversi *digital to analog* untuk *output* arus dibagi dua, yaitu:

1. *Input* nilai digital (0 – 4000), *output* konversi (4 – 20 mA)
2. *Input* nilai digital (0 – 4000), *output* konversi (0 – 20 mA)

2.7 Sensor Kecepatan (*Rotary Encoder*)

Sensor yang digunakan untuk memonitor kecepatan putaran motor induksi pada penulisan ini adalah *rotary encoder*. *Rotary encoder* digunakan untuk mengubah gerakan liniar atau putaran menjadi sinyal digital, dimana sensor putaran memonitor gerakan putar dari suatu alat. Sensor ini biasanya terdiri dari 2 lapis jenis penyandi, yaitu; Pertama, Penyandi rotari tambahan (yang mentransmisikan jumlah tertentu dari pulsa untuk masing-masing putaran) yang akan membangkitkan gelombang kotak pada objek yang diputar. Secara umum prinsip kerja *rotary encoder* dapat diilustrasikan seperti pada Gambar 2.34.



Gambar 2.34 Prinsip kerja *rotary encoder*

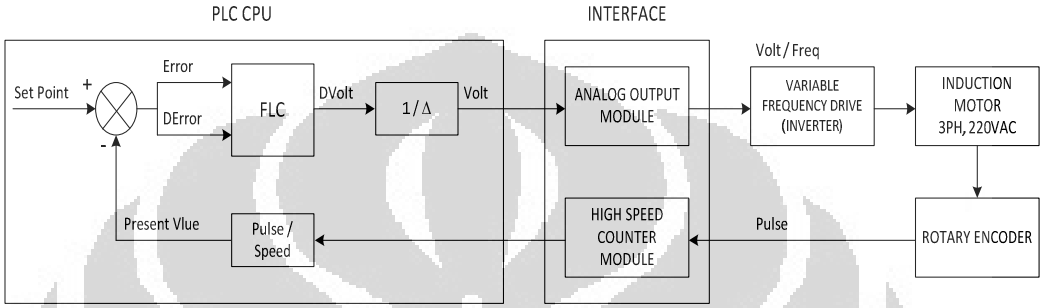
Berdasarkan ilustrasi pada Gambar 2.34, dua buah sensor optis (chanel A/*A* dan chanel B/*B*) pendeteksi “hitam dan putih” digunakan digunakan sebagai acuan untuk menentukan arah gerakan. Searah jarum jam (*clock-wise*. CW) atau

berlawanan arah jarum jam (*counter clock-wise*. CCW). Sedangkan jumlah pulsa (baik A atau B) dapat dihitung (menggunakan prinsip *counter*) sebagai banyak langkah yang ditempuh.

BAB 3
PERANCANGAN SYSTEM

3.1 Perancangan System *Fuzzy*

Blok diagram didalam perancangan sistem *fuzzy* pada penulisan ini secara umum adalah sebagai berikut.



Gambar 3.1 Blok diagram sistem *fuzzy*

Berdasarkan Gambar 3.1, pemrograman pada PLC CPU meliputi pemrograman *set point*, *input* Error & DError, pengendali *fuzzy*, *output crisp* hasil pemrosesan DV menjadi V, serta pemrograman *process value* (PV) hasil pemrosesan *pulsa* menjadi *speed*. Pada Gambar 3.1 modul *interface* dibagi menjadi dua, yaitu: modul *analog output* dan modul *high speed counter*. Pada Gambar 3.1, tegangan keluaran pada modul *analog output* diubah kedalam bentuk frekuensi menggunakan modul *variable frequency drive (inverter)* kemudian digunakan untuk mengubah-ubah kecepatan putar motor induksi (AC) tiga fasa, pulsa hasil pembacaan modul sensor *rotary encoder* diteruskan ke modul *high speed counter* untuk diterjemahkan kedalam bentuk nilai digital.

3.1.1 Fuzzifikasi

Fuzzifikasi adalah metode yang digunakan untuk mengubah nilai masukan *crisp* pada sistem *fuzzy* menjadi nilai *fuzzy*. Perancangan system *fuzzy* pada penulisan kali ini memiliki dua fungsi keanggotaan masukan (Error & DError). Nilai Error pada sistem diperoleh melalui persamaan dibawah ini.

$$Error = SP - PV$$

Keterangan:

- SP (*Set Point*) : Nilai *speed* yang diinginkan.
- PV (*Process Value*) : *Speed* aktual motor yang diperoleh dari pembacaan sensor kecepatan.

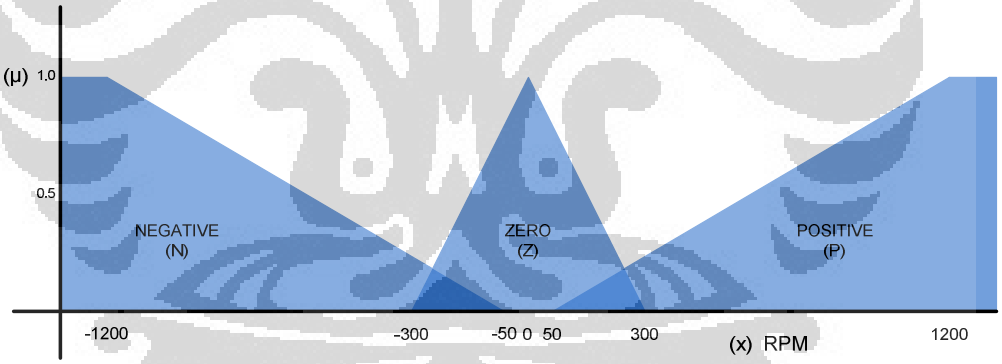
Sedangkan untuk nilai ΔError pada sistem diperoleh melalui persamaan berikut ini.

$$DError = Error(n) - Error(n-1)$$

Keterangan:

- Error(n) : Error saat ini.
- Error(n-1) : Error sebelumnya.

Fungsi keanggotaan masukan Error / DError memiliki model yang sama dibagi kedalam tiga *subset* (*negative, zero, positive*).

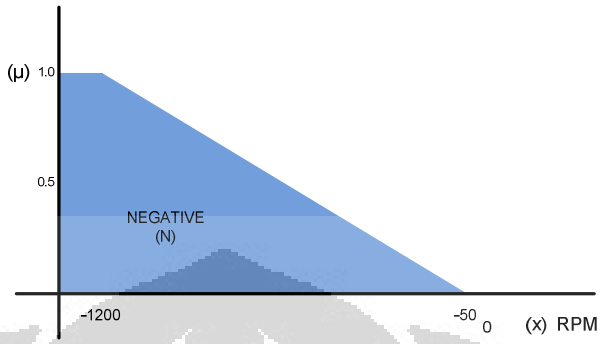


Gambar 3.2 Fungsi keanggotaan masukan Error / DError

Pada Gambar 3.2, *subset negative* berbentuk *trapezoid* dengan range *input* antara -1500 hingga -50 rpm serta *core fuzzy* antara -1500 hingga -1200 rpm. *Subset zero* berbentuk segitiga sama kaki dengan range *input* antara -300 hingga 300 rpm serta *core fuzzy* adalah 0 rpm. *Subset positive* berbentuk *trapezoid* dengan range *input* antara 50 hingga 1500 rpm serta *core fuzzy* antara 1200 hingga 1500 rpm.

1. *Subset Negative* (N) Fungsi Keanggotaan Masukan Error / DError

Model dari *subset negative* pada fungsi keanggotaan masukan Error / DError dapat dilihat pada Gambar 3.3.



Gambar 3.3 *Subset negative* (N) FK *input* Error / DError

Pada Gambar 3.3, *subset negative* (N) memiliki range kerja antara -1500 hingga -50 rpm serta *core fuzzy* antara -1500 hingga -1200 rpm, bentuk yang dianggap dapat mewakili adalah *trapezoid* dengan notasi sebagai berikut:

$\mu_N(x) = 1,$ $x <= -1200$ (rpm)

$(x + 50) / -1150,$ $-1200 < x <= -50$ (rpm)

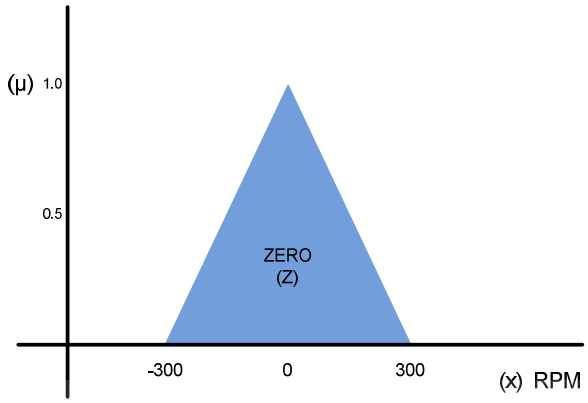
$0,$ $x > -50$ (rpm).

Keterangan:

- $\mu_N(x)$: Bobot *fuzzy* dari *subset negative* pada FK *input* Error / DError
- x : Nilai masukan Error / DError dalam bentuk *speed* (rpm)

2. *Subset Zero* (Z) Fungsi Keanggotaan Masukan Error

Model dari *subset negative* pada fungsi keanggotaan masukan Error / DError dapat dilihat pada Gambar 3.4.



Gambar 3.4 *Subset zero* (Z) FK *input* Error / DError

Pada Gambar 3.4, *Subset zero* (Z) memiliki range kerja antara -300 hingga 300 rpm serta *core fuzzy* adalah 0 rpm, bentuk yang dianggap dapat mewakili adalah segitiga sama kaki dengan notasi sebagai berikut:

$\mu_Z(x) = 0$, $x <= -300$ (rpm)

$(x + 300) / 300$, $-300 < x <= 0$ (rpm)

$(x - 300) / -300$, $0 < x <= 300$ (rpm)

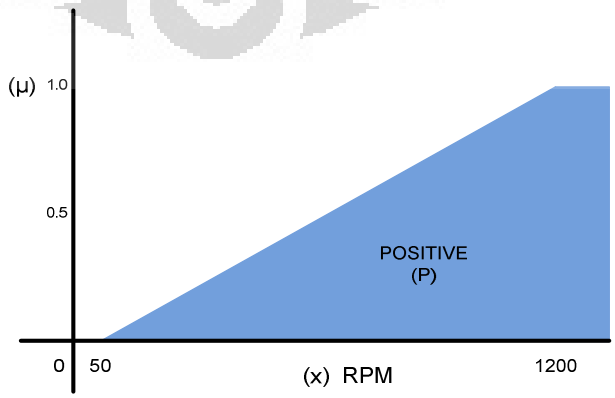
0, $x > 300$ (rpm).

Keterangan:

- $\mu_Z(x)$: Bobot *fuzzy* dari *subset zero* pada FK *input* Error / DError
- x : Nilai masukan Error / DError dalam bentuk *speed* (rpm)

3. *Subset Positive* (P) Fungsi Keanggotaan Masukan Error

Model dari *subset negative* pada fungsi keanggotaan masukan Error / DError dapat dilihat pada Gambar 3.5.



Gambar 3.5 *Subset positive* (P) FK *input* Error / DError

Pada Gambar 3.5, *subset positive* (P) memiliki range kerja antara 50 hingga 1500 rpm serta *core fuzzy* antara 1200 hingga 1500 rpm, bentuk yang dianggap dapat mewakili adalah *trapezoid* dengan notasi sebagai berikut:

$\mu_P(x) = 0,$ $x <= 50$ (rpm)

$(x - 50) / 1150,$ $50 < x <= 1200$ (rpm)

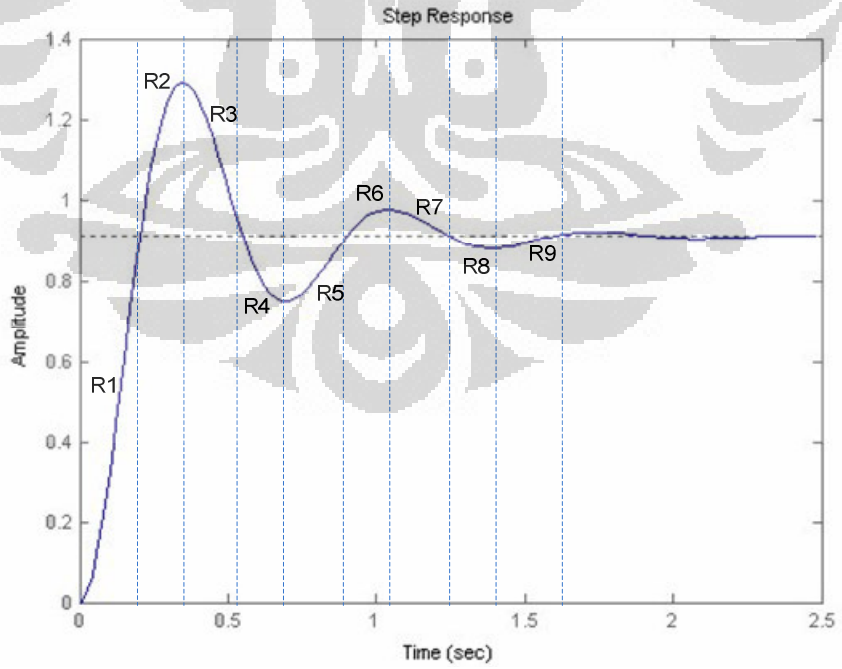
$1,$ $x > 1200$ (rpm).

Keterangan:

- $\mu_P(x)$: Bobot *fuzzy* dari *subset positive* pada FK *input* Error / DError
- x : Nilai masukan Error dalam bentuk *speed* (rpm)

3.1.2 Basis Aturan

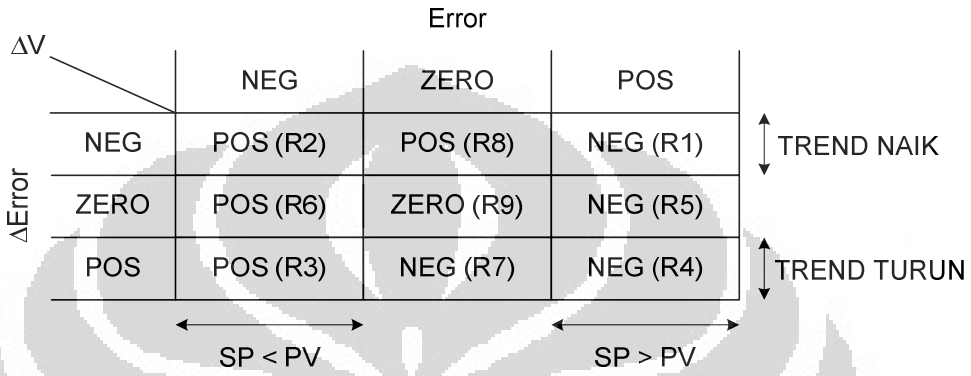
Basis aturan pada sistem yang dibuat memiliki sembilan aturan yang digunakan sebagai dasar dalam mengambil keputusan. Acuan yang digunakan dalam menentukan kesembilan basis aturan *fuzzy* dapat dilihat pada Gambar 3.6.



Gambar 3.6 Acuan dalam pembentukan basis aturan *fuzzy*

Pada Gambar 3.6, pembentukan basis aturan *fuzzy* didasarkan pada perbaikan dari grafik respon transien motor induksi. Plot perbaikan pada Gambar 3.6 dipecah kedalam sembilan area perbaikan dimulai dari R1 hingga R9. Komposisi dari kesembilan aturan tersebut dapat dilihat pada tabel relasi dibawah ini:

Tabel 3.1 Relasi *Fuzzy*



Tabel 3.1 menjelaskan komposisi basis aturan *fuzzy* dari hubungan antara fungsi keanggotaan *input Error, DError*, serta fungsi keanggotaan *output* DV berdasarkan plot perbaikan respon transien motor induksi dari R1 hingga R9. Penjabaran dari kesembilan aturan tersebut didalam bentuk ekspresi matemastis adalah sebagai berikut:

- R1 : JIKA Error adalah *POS (P)* & DError adalah *NEG (N)* MAKA DV adalah *NEG (N)*
- R2 : JIKA Error adalah *NEG (N)* & DError adalah *NEG (N)* MAKA DV adalah *POS (P)*
- R3 : JIKA Error adalah *NEG (N)* & DError adalah *POS (P)* MAKA DV adalah *POS (P)*
- R4 : JIKA Error adalah *POS (P)* & DError adalah *POS (P)* MAKA DV adalah *NEG (N)*
- R5 : JIKA Error adalah *POS (P)* & DError adalah *ZERO (Z)* MAKA DV adalah *NEG (N)*
- R6 : JIKA Error adalah *NEG (N)* & DError adalah *ZERO (Z)* MAKA DV adalah *POS (P)*

- R7 : JIKA Error adalah *ZERO (Z)* & DError adalah *POS (P)* MAKA DV adalah *NEG (N)*
- R8 : JIKA Error adalah *ZERO (Z)* & DError adalah *NEG (N)* MAKA DV adalah *POS (P)*
- R9 : JIKA Error adalah *ZERO (Z)* & DError adalah *ZERO (Z)* MAKA DV adalah *ZERO (Z)*

3.1.3 Inferensi

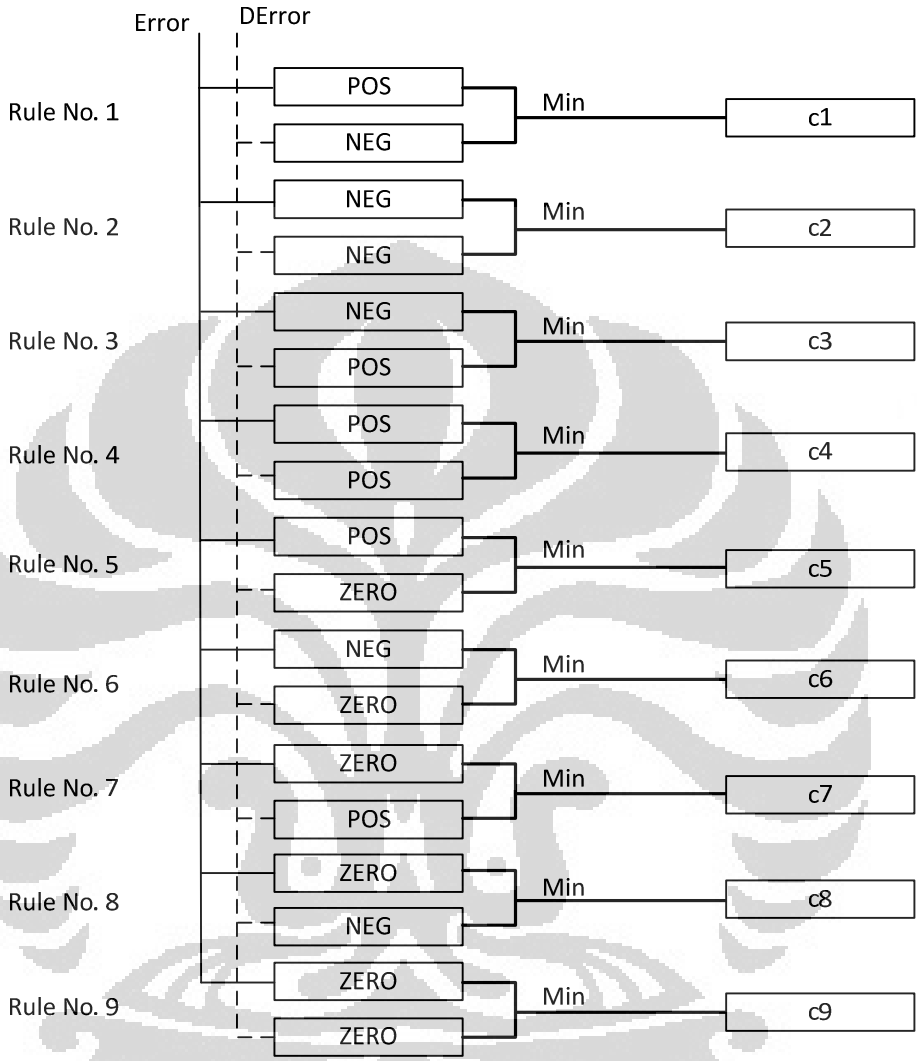
Penjabaran inferensi kedalam bentuk notasi matematis dari kelima aturan dasar fuzzy adalah sebagai berikut:

1. $\mu_P(x1)$ & $\mu_N(x2) \Rightarrow \mu_N(y)$
2. $\mu_N(x1)$ & $\mu_N(x2) \Rightarrow \mu_P(y)$
3. $\mu_N(x1)$ & $\mu_P(x2) \Rightarrow \mu_P(y)$
4. $\mu_P(x1)$ & $\mu_P(x2) \Rightarrow \mu_N(y)$
5. $\mu_P(x1)$ & $\mu_Z(x2) \Rightarrow \mu_N(y)$
6. $\mu_N(x1)$ & $\mu_Z(x2) \Rightarrow \mu_P(y)$
7. $\mu_Z(x1)$ & $\mu_P(x2) \Rightarrow \mu_N(y)$
8. $\mu_Z(x1)$ & $\mu_N(x2) \Rightarrow \mu_P(y)$
9. $\mu_Z(x1)$ & $\mu_Z(x2) \Rightarrow \mu_Z(y)$

Keterangan:

- $\mu_N(x1)$: bobot *fuzzy* pada *subset negative* FK Error
- $\mu_Z(x1)$: bobot *fuzzy* pada *subset zero* FK Error
- $\mu_P(x1)$: bobot *fuzzy* pada *subset positive* FK Error
- $\mu_N(x2)$: bobot *fuzzy* pada *subset negative* FK DError
- $\mu_Z(x2)$: bobot *fuzzy* pada *subset zero* FK DError
- $\mu_P(x2)$: bobot *fuzzy* pada *subset poaitive* FK DError
- $\mu_N(y)$: bobot *fuzzy* pada *subset negative* FK DV
- $\mu_Z(y)$: bobot *fuzzy* pada *subset zero* FK DV
- $\mu_P(y)$: bobot *fuzzy* pada *subset negative* FK DV

Dengan menggunakan aturan Zadeh-Mamdani, proses yang pertama kali dilakukan adalah mencari nilai *minimum* dari basis aturan *fuzzy*. Hubungan nilai *minimum* didalam basis aturan *fuzzy* dapat dilihat melalui Gambar 3.7.



Gambar 3.7 Hubungan *minimum* inferensi Mamdani

Pada Gambar 3.7, bobot *fuzzy* hasil inferensi minimum dari kesembilan aturan fuzzy (R1 – R9) disimpan ke dalam variabel c1 hingga c9. Penjabaran kedalam bentuk notasi matematis untuk mencari nilai *minimum* berdasarkan basis aturan *fuzzy* adalah sebagai berikut:

1. $\mu_{c1} = \text{Min}\{\mu_P(x1), \mu_N(x2)\}$

   $\mu_{c1} = \mu_P(x1), \qquad \mu_P(x1) < \mu_N(x2)$

   $\mu_{c1} = \mu_N(x2), \qquad \mu_N(x2) < \mu_P(x1)$

2. $\mu_{c2} = \text{Min}\{\mu_N(x1), \mu_N(x2)\}$

   $\mu_{c2} = \mu_N(x1), \quad \mu_N(x1) < \mu_N(x2)$

   $\mu_{c2} = \mu_N(x2), \quad \mu_N(x2) < \mu_N(x1)$

3. $\mu_{c3} = \text{Min}\{\mu_N(x1), \mu_P(x2)\}$

   $\mu_{c3} = \mu_N(x1), \quad \mu_N(x1) < \mu_P(x2)$

   $\mu_{c3} = \mu_P(x2), \quad \mu_P(x2) < \mu_N(x1)$

4. $\mu_{c4} = \text{Min}\{\mu_P(x1), \mu_P(x2)\}$

   $\mu_{c4} = \mu_P(x1), \quad \mu_P(x1) < \mu_P(x2)$

   $\mu_{c4} = \mu_P(x2), \quad \mu_P(x2) < \mu_P(x1)$

5. $\mu_{c5} = \text{Min}\{\mu_P(x1), \mu_Z(x2)\}$

   $\mu_{c5} = \mu_P(x1), \quad \mu_P(x1) < \mu_Z(x2)$

   $\mu_{c5} = \mu_Z(x2), \quad \mu_Z(x2) < \mu_P(x1)$

6. $\mu_{c6} = \text{Min}\{\mu_N(x1), \mu_Z(x2)\}$

   $\mu_{c6} = \mu_N(x1), \quad \mu_N(x1) < \mu_Z(x2)$

   $\mu_{c6} = \mu_Z(x2), \quad \mu_Z(x2) < \mu_N(x1)$

7. $\mu_{c7} = \text{Min}\{\mu_Z(x1), \mu_P(x2)\}$

   $\mu_{c7} = \mu_Z(x1), \quad \mu_Z(x1) < \mu_P(x2)$

   $\mu_{c7} = \mu_P(x2), \quad \mu_P(x2) < \mu_Z(x1)$

8. $\mu_{c8} = \text{Min}\{\mu_Z(x1), \mu_N(x2)\}$

   $\mu_{c8} = \mu_Z(x1), \quad \mu_Z(x1) < \mu_N(x2)$

   $\mu_{c8} = \mu_N(x2), \quad \mu_N(x2) < \mu_Z(x1)$

9. $\mu_{c9} = \text{Min}\{\mu_Z(x1), \mu_Z(x2)\}$

   $\mu_{c9} = \mu_Z(x1), \quad \mu_Z(x1) < \mu_Z(x2)$

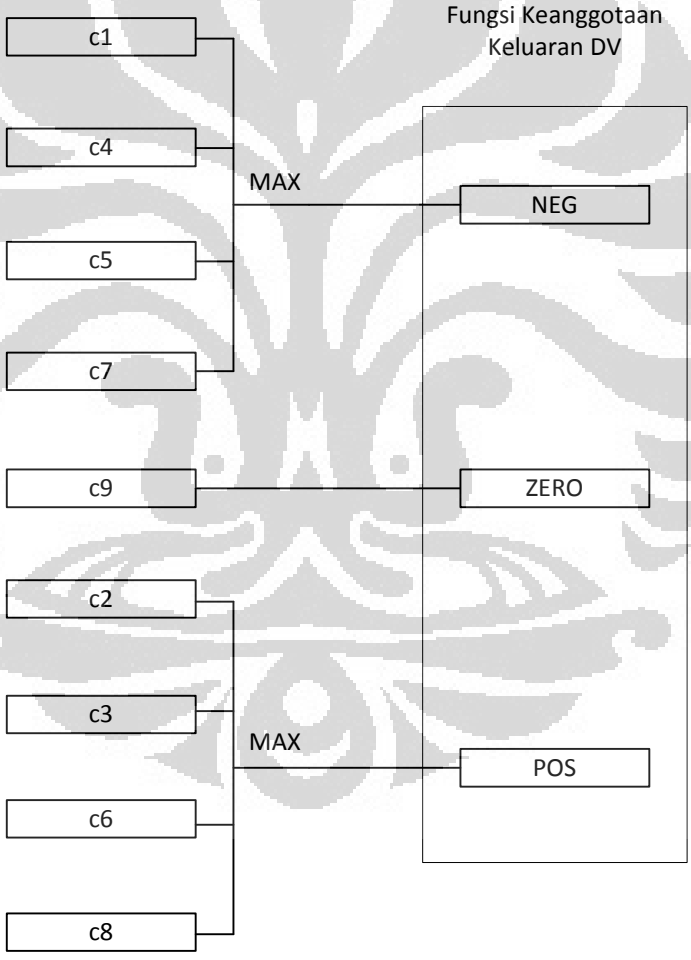
   $\mu_{c9} = \mu_Z(x2), \quad \mu_Z(x2) < \mu_Z(x1)$

Keterangan:

- μ_{c1} : bobot *fuzzy* hasil inferensi *minimum* rule no. 1
- μ_{c2} : bobot *fuzzy* hasil inferensi *minimum* rule no. 2
- μ_{c3} : bobot *fuzzy* hasil inferensi *minimum* rule no. 3
- μ_{c4} : bobot *fuzzy* hasil inferensi *minimum* rule no. 4
- μ_{c5} : bobot *fuzzy* hasil inferensi *minimum* rule no. 5
- μ_{c6} : bobot *fuzzy* hasil inferensi *minimum* rule no. 6

- μ_{c7} : bobot *fuzzy* hasil inferensi *minimum* rule no. 7
- μ_{c8} : bobot *fuzzy* hasil inferensi *minimum* rule no. 8
- μ_{c9} : bobot *fuzzy* hasil inferensi *minimum* rule no. 9

Setelah inferensi *minimum* diperoleh, selanjutnya adalah mencari komposisi untuk menghasilkan keluaran tunggal yang memiliki korelasi dengan fungsi keanggotaan keluaran DV berdasarkan basis aturan *fuzzy*. Metode yang digunakan di dalam proses tersebut adalah mencari nilai maksimum dari nilai-nilai *minimum* yang telah diperoleh. Hubungan nilai *maximum* didalam basis aturan *fuzzy* dapat dilihat melalui gambar di bawah ini:



Gambar 3.8 Hubungan *maximum* inferensi Mamdani

Pada Gambar 3.8, komposisi dari *subset negative* pada fungsi keanggotaan *output* DV terdiri dari variabel (c1, c4, c5, c7), untuk *input fuzzy* sendiri diperoleh dari nilai *maximum* dari keempat variabel tersebut. Pada Gambar 3.8, komposisi

dari *subset positive* pada fungsi keanggotaan *output* DV terdiri dari variabel (c2, c3, c6, c8), untuk *input fuzzy* sendiri diperoleh dari nilai *maximum* dari keempat variabel tersebut. Pada Gambar 3.8, *input fuzzy* untuk *subset zero* pada fungsi keanggotaan *output* DV berasal dari variabel c9. Penjabaran hubungan inferensi *maximum* kedalam bentuk notasi matematis adalah sebagai berikut:

1. $\mu_N(y) = \text{Max}\{\mu_{c1}, \mu_{c4}, \mu_{c5}, \mu_{c7})\}$

   $\mu_N(y) = \mu_{c1}, \quad \mu_{c1} > \mu_{c4} > \mu_{c5} > \mu_{c7}$

   $\mu_N(y) = \mu_{c4}, \quad \mu_{c4} > \mu_{c1} > \mu_{c5} > \mu_{c7}$

   $\mu_N(y) = \mu_{c5}, \quad \mu_{c5} > \mu_{c4} > \mu_{c1} > \mu_{c7}$

   $\mu_N(y) = \mu_{c7}, \quad \mu_{c7} > \mu_{c5} > \mu_{c4} > \mu_{c1}$

2. $\mu_P(y) = \text{Max}\{\mu_{c2}, \mu_{c3}, \mu_{c6}, \mu_{c8})\}$

   $\mu_N(y) = \mu_{c2}, \quad \mu_{c2} > \mu_{c3} > \mu_{c6} > \mu_{c8}$

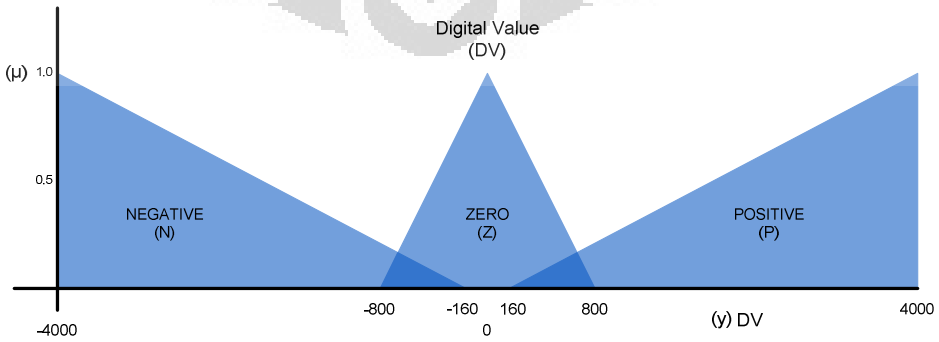
   $\mu_N(y) = \mu_{c3}, \quad \mu_{c3} > \mu_{c2} > \mu_{c6} > \mu_{c8}$

   $\mu_N(y) = \mu_{c6}, \quad \mu_{c6} > \mu_{c3} > \mu_{c2} > \mu_{c8}$

   $\mu_N(y) = \mu_{c8}, \quad \mu_{c8} > \mu_{c6} > \mu_{c3} > \mu_{c2}$

3.1.4 Fungi Keanggotaan Keluaran DV

Fungsi keanggotaan keluaran DV merepresentasikan perubahan tegangan analog yang terhubung sebagai masukan ke modul *inverter*. Agar frekuensi keluaran yang dihasilkan oleh *inverter* stabil dan sesuai dengan perubahan Error & ΔError maka fungsi keanggotaan keluaran DV dibagi kedalam tiga *subset (negative, zero, positive)*. Besaran yang diinginkan dari fungsi keanggotaan keluaran DV adalah *crisp* sehingga memiliki masukan berupa nilai *fuzzy* yang didapatkan dari proses inferensi.

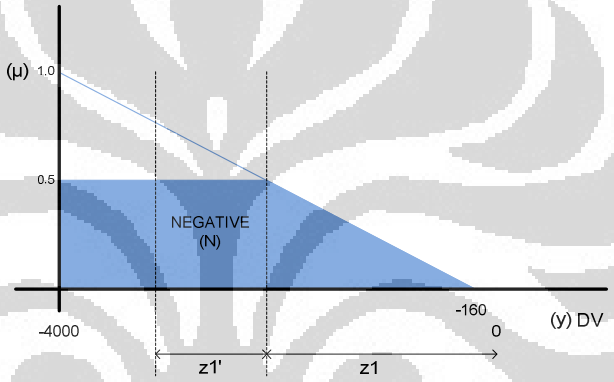


Gambar 3.9 Fungsi keanggotaan keluaran DV

Pada Gambar 3.9, *subset negative* berbentuk segitiga siku-siku dengan range *input* antara -4000 hingga -160 nilai digital serta *core fuzzy* adalah -4000 nilai digital. *Subset zero* berbentuk segitiga sama kaki dengan range *input* antara -800 hingga 800 nilai digital serta *core fuzzy* adalah 0 nilai digital. *Subset positive* berbentuk segitiga siku-siku dengan range *input* antara 160 hingga 4000 nilai digital serta *core fuzzy* adalah 4000 nilai digital.

1. *Subset Negative* (N) Fungsi Keanggotaan Keluaran DV

Model dari *subset negative* pada fungsi keanggotaan keluaran DV dapat dilihat pada Gambar 3.10.



Gambar 3.10 *Subset negative* (N) FK *output* DV

Pada Gambar 3.10, *subset negative* (N) memiliki range kerja antara -4000 sampai -160 nilai digital dengan masukan nilai fuzzy antara 0 hingga 1. Bentuk yang dianggap dapat mewakili adalah segitiga siku-siku dengan notasi sebagai berikut:

$$z1 = -(\mu_N(y) * 3840) - 160$$

$$z1' = (-4000 - z1) / 2$$

$$z_N = z1 + z1'$$

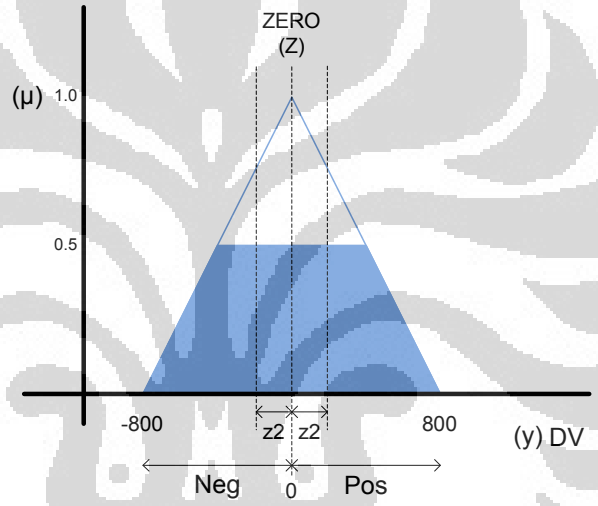
Keterangan:

- $\mu_N(y)$: Bobot *fuzzy* hasil inferensi *maximum* untuk *subset negative* pada FK *output* DV

- z1 : Nilai *crisp* dari titik perpotongan pada *subset negative* FK *ouput* DV
- z1' : Nilai tengah *(crisp)* dari garis perpotongan pada *subset negative* FK *output* DV
- z_N : Nilai *crisp* dari *subset negative* pada FK *output* DV

2. *Subset Zero* (Z) Fungsi Keanggotaan Keluaran DV

Model dari *subset zero* pada fungsi keanggotaan keluaran DV dapat dilihat pada Gambar 3.11.



Gambar 3.11 *Subset zero* (Z) FK *output* DV

Pada Gambar 3.11, *zubset zero* (Z) memiliki range kerja antara -800 sampai 800 nilai digital dengan masukan nilai *fuzzy* antara 0 hingga 1. Bentuk yang dianggap dapat mewakili adalah segitiga sama kaki dengan notasi sebagai berikut:

- Jika sinyal negatif

$$z_Z = ((\mu_Z(y) * 800) - 800) / 2$$

- Jika sinyal positif

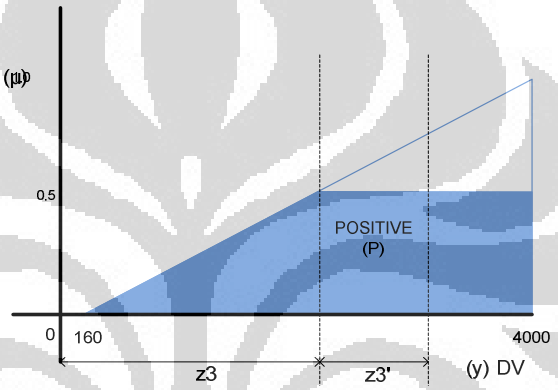
$$z_Z = - ((\mu_Z(y) * 800) + 800) / 2$$

Keterangan:

- $\mu_Z(y)$: Bobot *fuzzy* hasil inferensi *maximum* untuk *subset zero* pada FK *output* DV
- z_Z : Nilai *crisp* dari *subset zero* pada FK *output* DV

3. *Subset Positive* (P) Fungsi Keanggotaan Keluaran DV

Model dari *subset positive* pada fungsi keanggotaan keluaran DV dapat dilihat pada Gambar 3.10.



Gambar 3.12 *Subset positive* (P) FK *output* DV

Pada Gambar 3.12, *subset positive* (P) pada fungsi keanggotaan keluaran DV memiliki range kerja antara 160 sampai 4000 nilai digital dengan masukan nilai *fuzzy* antara 0 hingga 1. Bentuk yang dianggap dapat mewakili adalah segitiga siku-siku dengan notasi sebagai berikut:

$$z3 = (\mu_P(y) * 3840) + 160$$

$$z3' = (4000 - z3) / 2$$

$$z_P = z3 + z3'$$

Keterangan:

- $\mu_P(y)$: Bobot *fuzzy* hasil inferensi *maximum* untuk *subset positive* pada FK *output* DV
- $z3$: Nilai *crisp* dari titik perpotongan pada *subset positive* FK

ouput DV

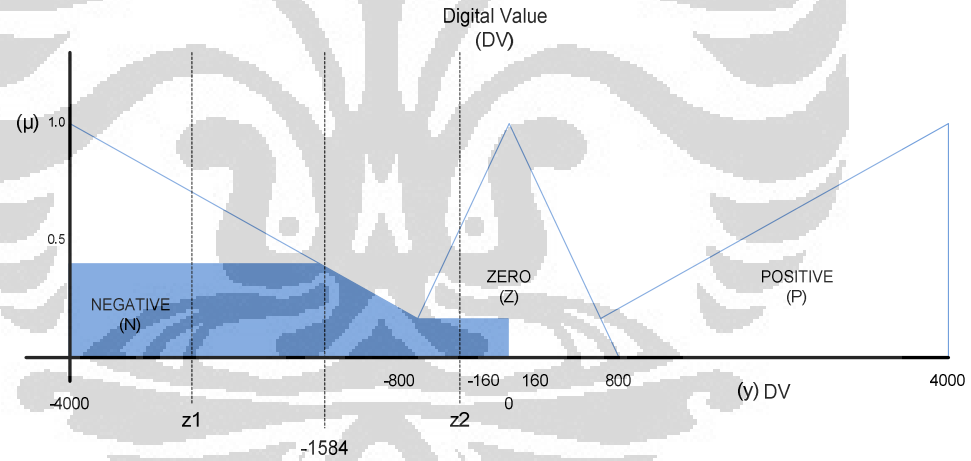
- $z3'$: Nilai tengah *(crisp)* dari garis perpotongan pada *subset positive* FK *output* DV
- z_P : Nilai *crisp* dari *subset positive* pada FK *output* DV

3.1.5 Defuzzifikasi

1. Metode Nilai Tengah (*Middle of Maxima*)

Metode nilai tengah adalah proses untuk mencari nilai keluaran tunggal yang tegas diperoleh dengan menjumlahkan nilai tegas dari *subset* fungsi keanggotaan keluaran frekuensi dibagi dengan banyaknya *subset* yang mengeluarkan nilai tersebut. Pendekatan matematis dari metode *middle of maxima* adalah sebagai berikut:

$$z = \frac{z1 + z2 + z3 + \ldots + zn}{n}$$



Gambar 3.13 Contoh perhitungan matematis metode *middle of maxima*

Berdasarkan Gambar 3.13, nilai defuzzifikasi menggunakan metode *middle of maxima* diperoleh dengan menjumlahkan nilai *crisp* yang aktif dari *subset negative* (z1) dan *subset zero* (z2) dibagi dengan banyaknya *subset-subset* yang aktif (2 *subset*), nilai defuzzifikasi yang dihasilkan adalah -1584 nilai digital. Perhitungan lebih detil adalah sebagai berikut.

$$z = \frac{z1 + z2}{2}$$

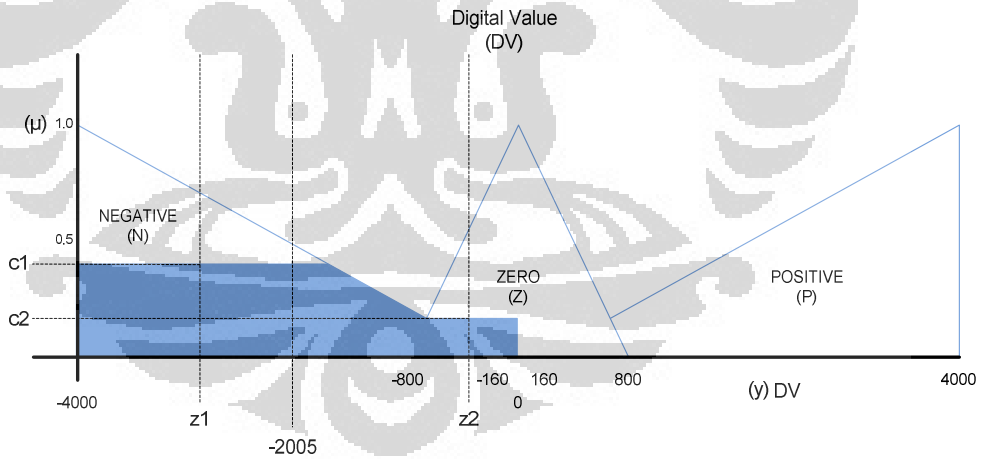
$$z = \frac{(-2848)+(-320)}{2}$$

$$z = -1584$$

2. Metode Bobot Rata-Rata (*Weighted Average*)

Metode bobot rata-rata adalah sebuah proses untuk mencari nilai keluaran tunggal yang tegas diperoleh dengan menjumlahkan perkalian antara masing-masing nilai *fuzzy* dari subset fungsi keanggotaan keluaran dengan nilai tegas subset dibagi dengan jumlah nilai-nilai *fuzzy*. Pendekatan matematis dari metode *weighted average* adalah sebagai berikut:

$$z = \frac{(c1 * z1) + (c2 * z2) + (c3 * z3) + (cn * zn)}{(c1 + c2 + c3 + cn)}$$



Gambar 3.14 Contoh perhitungan matematis metode *weighted average*

Berdasarkan Gambar 3.14, nilai defuzzifikasi menggunakan metode *weighted average* diperoleh dengan menjumlahkan perkalian antara nilai *crisp* dari *subset negative* (z1) dengan bobot *fuzzy* hasil inferensi pada *subset negative* (c1) dan nilai *crisp* dari *subset zero* (z2) dengn bobot *fuzzy* hasil inferensi pada *subset zero* (c2) dibagi dengan

jumlah dari bobot *fuzzy* pada *subset negative* (c1) dan *subset zero* (c2), nilai defuzzifikasi yang dihasilkan adalah -2005 nilai digital. Perhitungan lebih detil adalah sebagai berikut.

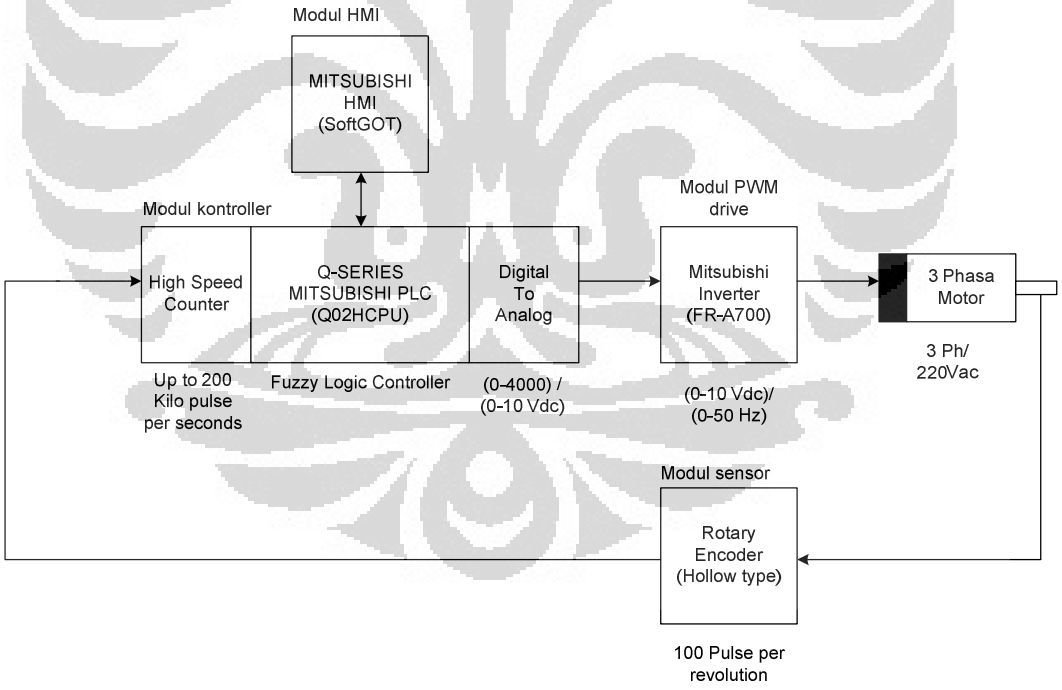
$$z = \frac{(c1 * z1) + (c2 * z2)}{(c1 + c2)}$$

$$z = \frac{(0.4 * (-2848)) + (0.2 * (-320))}{(0.4 + 0.2)}$$

$$z = -2005$$

3.2 Sistem Konfigurasi

Blok diagram sistem pengatur kecepatan putar motor induksi secara keseluruhan adalah sebagai berikut.



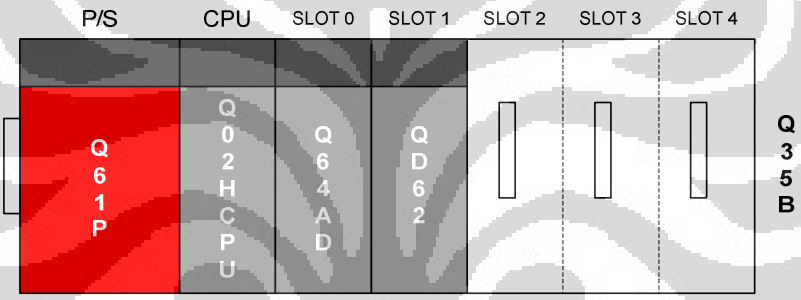
Gambar 3.15 Sistem konfigurasi secara umum

Pada Gambar 3.15, secara umum sistem yang dibuat dibagi menjadi lima bagian, yang pertama adalah modul HMI berupa *screen monitoring fuzzy* yang dijalankan dengan *software* SoftGOT1000 berbasis *windows,* yang kedua adalah

modul kontroller yang terbagi kedalam tiga bagian (modul *high speed counter* dengan spesifikasi pembacaan pulsa hingga 200 kpps, modul CPU Q02HCPU, dan modul *digital to analog* dengan spesifikasi konversi nilai digital 0 – 4000 diubah menjadi tegangan 0 – 10 Vdc), yang ketiga adalah modul *inverter* dengan spesifikasi konversi tegangan 0 – 10 Vdc diubah menjadi frekuensi 0 – 50 Hz, yang keempat modul sensor kecepatan *rotary encoder* dengan spesifikasi resolusi 100 pulse/rev, yang kelima adalah motor induksi tiga fasa 220 ac.

3.2.1 Modul *Controller*

Modul *controller* merupakan pusat pemrosesan dari algoritma-algoritma *fuzzy*, berbentuk PLC *modular*. Konfigurasi modul *controller* adalah sebagai berikut:



Gambar 3.16 Konfigurasi modul *controller*

Pada Gambar 3.16. komponen pada modul *controller* terdiri dari lima modul, yaitu: modul CPU, modul *high speed counter*, modul *analog output*, modul *power supply*, dan modul *base plate*. Spesifikasi dari modul-modul yang digunakan adalah sebagai berikut:

1. **CPU (Q02HCPU)**: kapasitas program (28 kstep), kecepatan pemrosesan (0.034 µ sec), tipe memori (standar ROM).

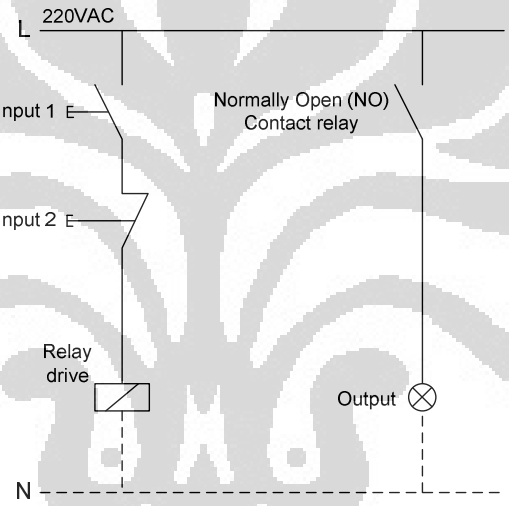
2. ***Power supply* (Q61P)**: masukan *power supply* (100 – 240 VAC), keluaran 5 VDC (6A).

3. ***Base plate* (Q35B)**: slot *power supply* (1), slot CPU (1), slot untuk modul *input, output*, dan *intelligent* modul (5) dimulai dari slot 0 – 4.

4. ***High speed counter*** **(QD62)**: total *channel* (2 ch), range pembacaan pulsa secara inc/dec antara -2147483648 hingga 2147483647, frekuensi maksimum berbentuk pulsa yang dapat dibaca sebesar 200 *kilo pulse per seconds.*

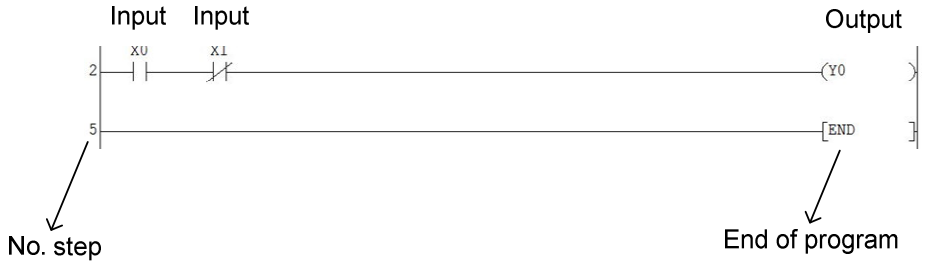
5. ***Analog output*** **(Q62DA)**: total *channel analog* (2 ch), range konversi dari digital ke analog adalah 0 – 10 VDC linier dengan nilai digital 0 – 4000.

Bahasa pemrograman yang digunakan adalah *step ladder*. *Step ladder* merupakan adopsi dari sistem logika kontak relay konvensional.



Gambar 3.17 Sistem kontak konvensional

Pada Gambar 3.17, ketika *input* 1 aktif *(closed ciruit)* maka arus mengalir menuju *relay* dan mengaktifkan kontak "NO" menjadi "NC" pada *relay* dan menyebabkan arus mengalir ke *output*, ketika *input* 2 aktif *(open circuit)* maka arus yang mengalir ke *relay* terputus dan kontak "NC" berubah kembali menjadi "NO", hal ini menyebabkan arus yang mengalir ke *output* menjadi terputus pula.

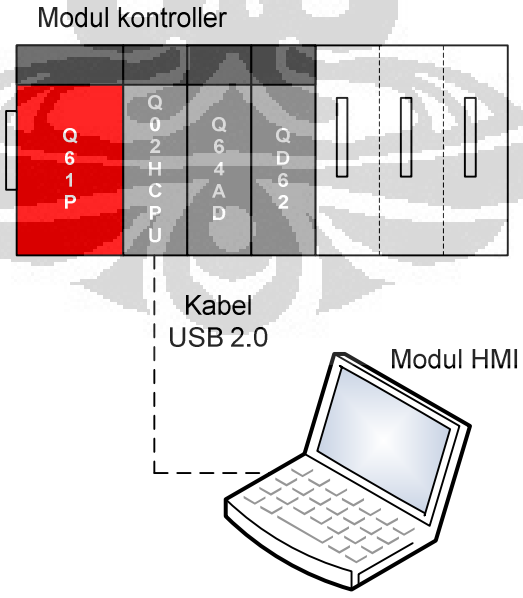


Gambar 3.18 Bahasa pemrograman *ladder*

Pada Gambar 3.18, ketika *input* "X0" aktif maka seolah-olah ada aliran arus dari rung sebelah kiri menuju ke *output* "Y0" pada rung sebelah kanan, ketika *input* "X1" aktif maka seolah-olah aliran arus dari rung sebelah kiri terputus menuju *output* "Y0" pada rung sebelah kanan. Logika pembacaan diagram *ladder* pada Gambar 3.18 memiliki kemiripan dengan sistem logika kontak konvensional. *Software* yang digunakan untuk pemrograman PLC adalah *sofware* integrasi berbasis windows tipe GX-Developer versi 8.95Z.

3.2.2 Modul HMI (*Human Machine Interface*)

Modul HMI merupakan pusat untuk melakukan fungsi pengawasan serta pengoperasian sistem. Komunikasi yang digunakan pada modul HMI adalah sebagi berikut.



Gambar 3.19 Konfigurasi komunikasi tipe *direct*

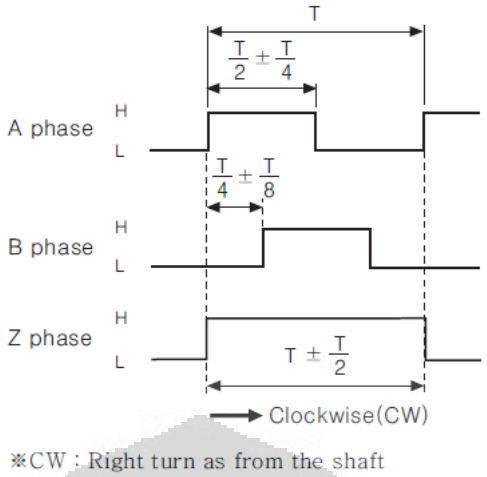
Pada Gambar 3.19, jenis komunikasi antara modul HMI dengan modul kontroller adalah tipe *direct*, dimana modul HMI terhubung langsung ke *port* pemrograman CPU dengan menggunakan kabel USB 2.0 sebagai medianya. *Software* yang digunakan untuk desain modul HMI adalah *sofware* berbasis *windows* tipe GT-Designer3 versi 1.10L *Software* untuk menjalankan fungsi pengawasan dan pengoperasian secara real time adalah *software* berbasis *windows* tipe GT-SoftGOT1000 versi 1.10L.

3.2.3 Modul Sensor Kecepatan

Modul sensor yang digunakan untuk memonitor kecepatan putaran dari motor induksi adalah *rotary encoder* tipe E40H12-100-3-N-24 model *hollow* yang dibuat oleh Autonic.

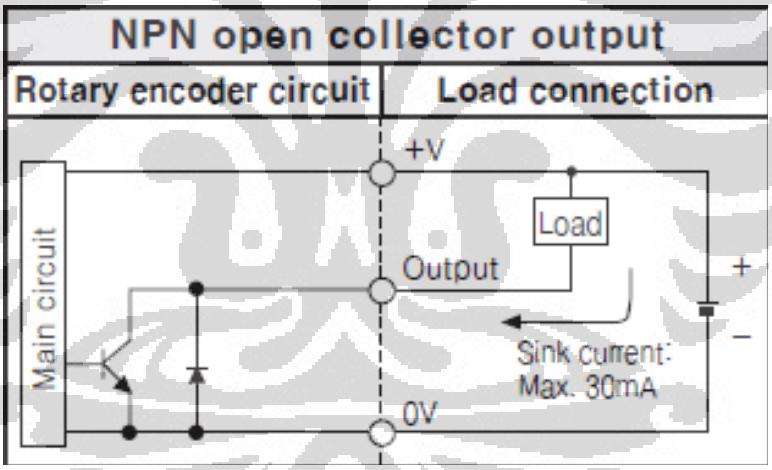
Gambar 3.20 *Rotary encoder*

Gambar 3.20 merupakan model dari *rotary encoder* yang digunakan pada sistem. *Rotary encoder* tipe ini memiliki keluaran ϕA, ϕB, dan ϕZ berbentuk transistor tipe NPN, format metode pulsa keluarannya adalah CW/CCW. ϕA dan ϕB berfungsi sebagai pulsa keluaran dari hasil pembacaan putaran yang dibuat sinkron terhadap *pulley* motor induksi sedangkan ϕZ berfungsi sebagai penyandi posisi nol dari *shaft encoder* sendiri. Spesifikasi dari tegangan input (12 -24 Vdc). Resolusi yang dihasilkan adalah 100 pulse/revolution.



Gambar 3.21 Format metode pulsa keluaran CW/CCW

Pada Gambar 3.21, beda phase antara A-phase dengan B-phase adalah T/2, dimana T adalah periode pulse (sec), Z-phase tidak digunakan pada sistem yang dibuat. *Wiring control single line* dari modul sensor kecepatan yang menggunakan *rotary encoder* dapat dilihat pada Gambar 3.22



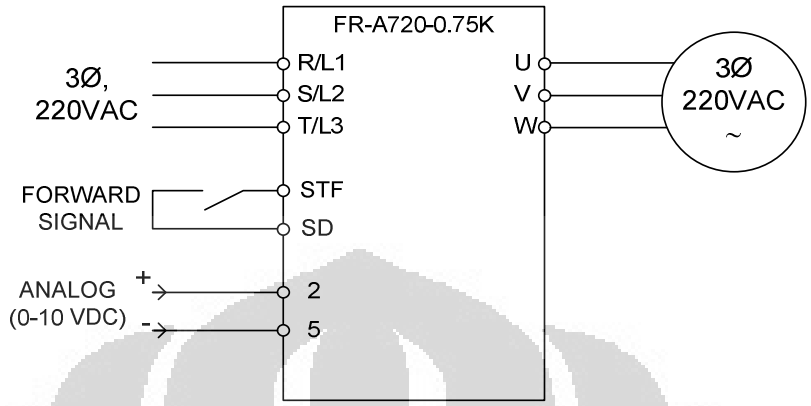
Gambar 3.22 Diagram *wiring control rotary encoder* tipe NPN

Pada Gambar 3.22, *logic* keluaran phase dari *rotary encoder* adalah 0V dengan arus maksimum yang dihasilkan adalah 30 mA.

3.2.4 Modul PWM *Drive*

Modul PWM *drive* terdiri dari modul *inverter* serta motor ac induksi tiga fasa. Modul *inverter* yang digunakan adalah *inverter* yang dibuat oleh Mitsubishi Electric dengan tipe FR-A720-0.75K. Kapasitas *modul inverter* yang digunakan adalah 0.75 KW, memiliki tegangan sumber tipe ac tiga fasa 220V. Metode yang

digunakan untuk pengendalian motor adalah dengan mengatur lebar pulsa tegangan ac hasil rekayasa (*Pulse Width Modulation*). Diagram Wiring control modul *inverter* dapat dilihat pada gambar berikut.



Gambar 3.23 Diagram *wiring control* modul *inverter*

Pada Gambar 3.23, terminal (R/L1, S/L2, T/L3) adalah terminal untuk masukan tiga fasa 220 Vac, terminal (U, V, W) adalah terimnal keluaran yang terhubung ke motor induksi tiga fasa 220 Vac, terminal STF dan SD adalah terminal yang menandakan arah putaran dari motor induksi *forward/clock wise,* terminal 2 dan 5 adalah terminal masukan sinyal eksternal berupa tegangan 0 – 10 Vdc. Penyesuaian settingan parameter *inverter* dengan sistem *fuzzy* yang dibuat dapat dilihat pada tabel berikut.

Tabel 3.2 *Setting* parameter *inverter*

NO.	PARAMETER NO.	NAME	SETTING	DESCRIPTION
1	1	Maximum Frequency	50 Hz	Maximum frequency which can be serve by inverter
2	2	Minimum Frequency	0 Hz	Minimum frequency which can be serve by inverter
3	3	Base Frequency	50 Hz	Base frequency motor
4	73	Analog Input Selection	0	0 - 10 VDC

5	79	Operation Mode Selection	2	External mode
6	125	Terminal 2 Frequency Seting Gain Frequency	50 Hz	Gain frequency output inverter at 10 VDC

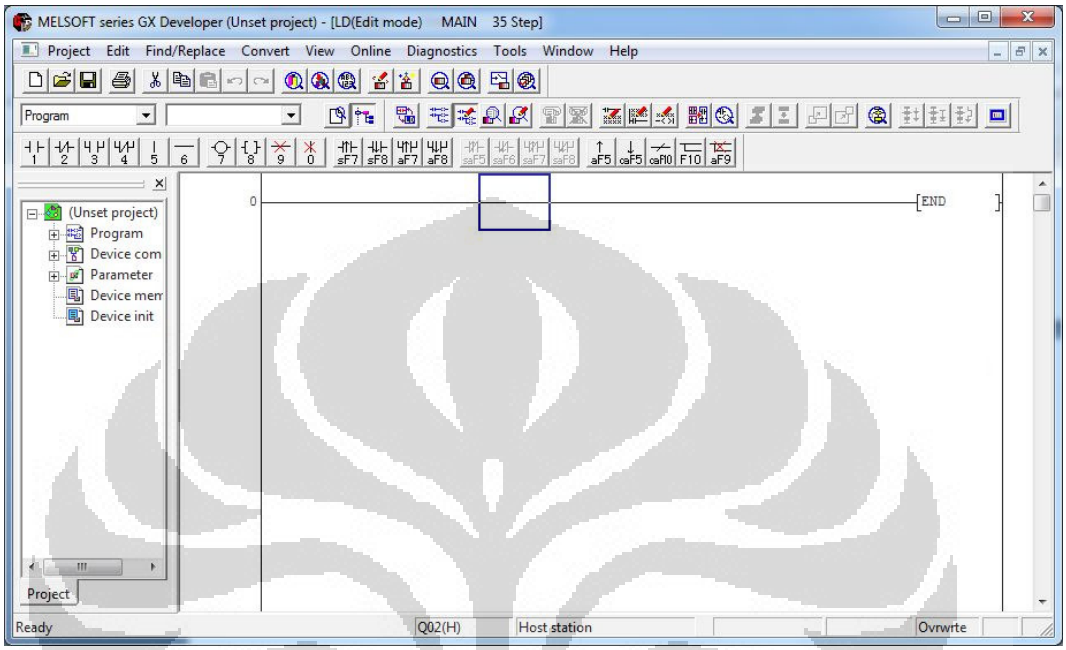
Pada Tabel 3.2, *setting* frekuensi maksimum yang dihasilkan *inverter* berdasarkan parameter no. 1 adalah 50 Hz, *setting* frekuensi minimum yang dihasilkan *inverter* berdasarkan parameter no. 2 adalah 0 Hz, *setting* frekuensi dasar dari motor induksi berdasarkan parameter no. 3 adalah 50 Hz, *setting* pemilihan sinyal masukan *analog* berdasarkan parameter no. 73 adalah 0 (0 – 10 Vdc), *setting* pemilihan mode operasional *inverter* berdasarkan parameter no. 79 adalah 2 (mode eksternal), dan *setting* frekuensi puncak pada saat sinyal masukan *analog* 10 Vdc berdasarkan parameter no. 125 adalah 50 Hz.

Motor yang digunakan pada penulisan kali ini adalah motor induksi tiga fasa dengan kapasitas daya 0.2 kW buatan TECO. Memiliki tegangan input 220VAC dengan frekuensi kerja normal 50 Hz, serta tipe pengasutan Y. Model motor induksi tiga fasa yang digunakan di dalam sistem dapat dilihat pada Gambar 3.24.

Gambar 3.24 Motor induksi ac tiga fasa kapasitas 0.2 kW

3.3 Perancangan Pemrograman PLC

Pemrograman PLC dirancang dan ditulis dengan menggunakan bahasa pemrograman *ladder, software* yang digunakan adalah *software* integrasi PLC Mitsubishi berbasis *windows* tipe GX-Developer versi 8.95Z.



Gambar 3.25 *Preview* GX-Developer

Berdasarkan Gambar 3.25, pada baris pertama adalah *standard tool*, pada baris kedua adalah program *selection tool*, pada baris ketiga adalah *edit & programming tool*, pada kolom sebelah kiri adalah *list project (pogram, parameter, comment)*, dan pada kolom sebelah kanan adalah area penulisan program. Komponen yang digunakan untuk mengedit program diantaranya:

- LD / AND digunakan sebagai kontak masukan seri *normally open* (NO).
- LDI / ANI digunakan sebagai kontak masukan seri *normally closed* (NC).
- OR digunakan sebagai kontak masukan parallel *normally open* (NO).
- ORI digunakan sebagai kontak masukan parallel *normally closed* (NC).

-  Vertical Line digunakan untuk membuat garis bantu vertikal sepanjang satu tab.
-  Horizontal Line digunakan untuk membuat garis bantu horisontal sepanjang satu tab.
-  OUT digunakan sebagai kontak keluaran yang di *drive* oleh kontak masukan sepanjang garis *ladder*.
-  Instruction digunakan untuk menuliskan instruksi khusus di dalam program.
-  Delete Horizontal Line digunakan untuk menghapus garis bantu horisontal sepanjang satu tab.
-  Delete Vertical Line digunakan untuk menghapus garis bantu vertikal sepanjang satu tab.

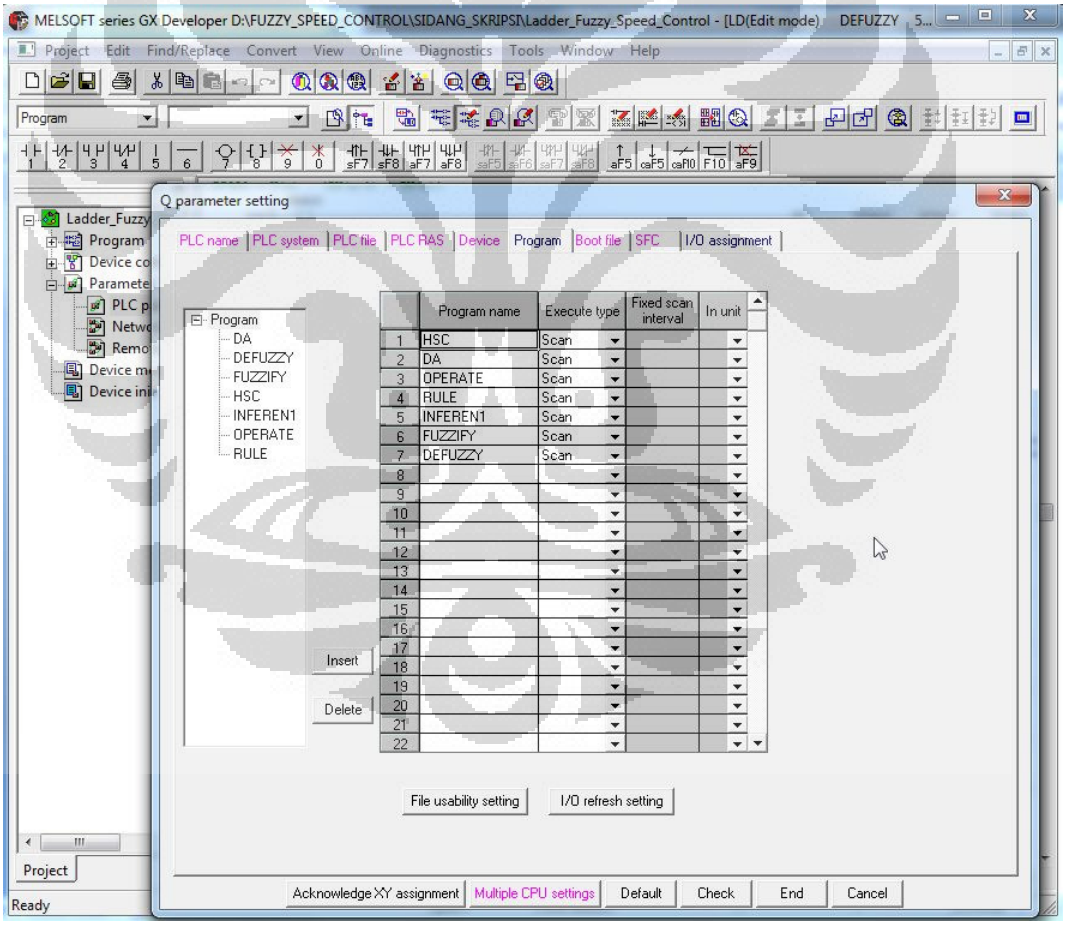
Beberapa fungsi *keyboard* yang membantu di dalam proses pemrogaman, antara lain:

- **SHIFT + INS** berfungsi untuk menyisipkan baris sebanyak satu tab.
- **CTRL + INS** berfungsi untuk menambahkan kolom sebanyak satu tab.
- **SHIFT + DEL** berfungsi untuk menghapus baris sebanyak satu tab.
- **CTRL + DEL** berfungsi untuk menghapus kolom sebanyak satu tab.
- **SHIFT + F2** berfungsi untuk mengubah mode software kedalam mode edit program.
- **SHIFT + F3** berfungsi untuk mengubah mode software kedalam mode online.
- **F4** berfungsi untuk mengkonversi program kedalam mode penulisan CPU PLC setelah proses edit.
- **SHIFT + F4** berfungsi untuk mengkonversi serta menulis program sekaligus ke dalam CPU saat mode online.

Untuk mempermudah dalam melakukan pemrograman PLC, maka program dipecah menjadi enam file program, yaitu:

1. Pemrograman *high speed counter*
2. Pemrograman *analog output*
3. Pemrograman masukan Error & DError
4. Pemrograman fuzzifikasi
5. Pemrograman basis aturan
6. Pemrograman inferensi & keluaran
7. Pemrograman defuzzifikasi

Pengaturan pola eksekusi program dari keenam file tersebut dapat dilakukan pada parameter setting PLC di dalam *software* GX-Developer.

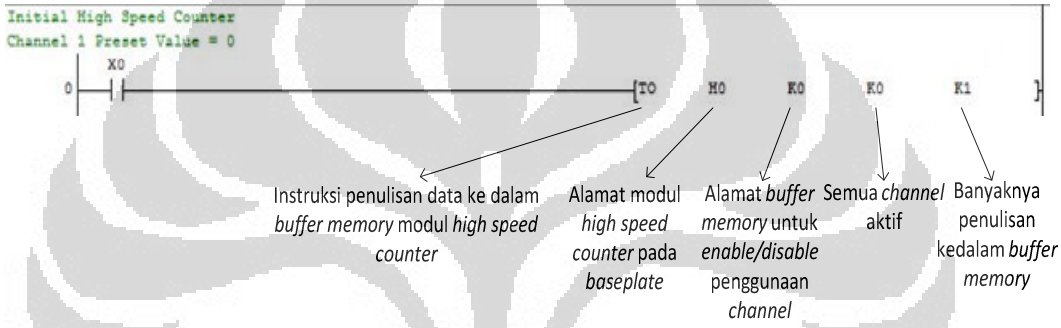


Gambar 3.26 Setting parameter *file* program PLC

Pada Gambar 3.26, file-file program yang akan di scan berdasarkan setting parameter PLC dibagi kedalam tujuh bagian, yaitu: HSC, DA, OPERATE, RULE, INFEREN1, FUZZYFY, DEFUZZYFY.

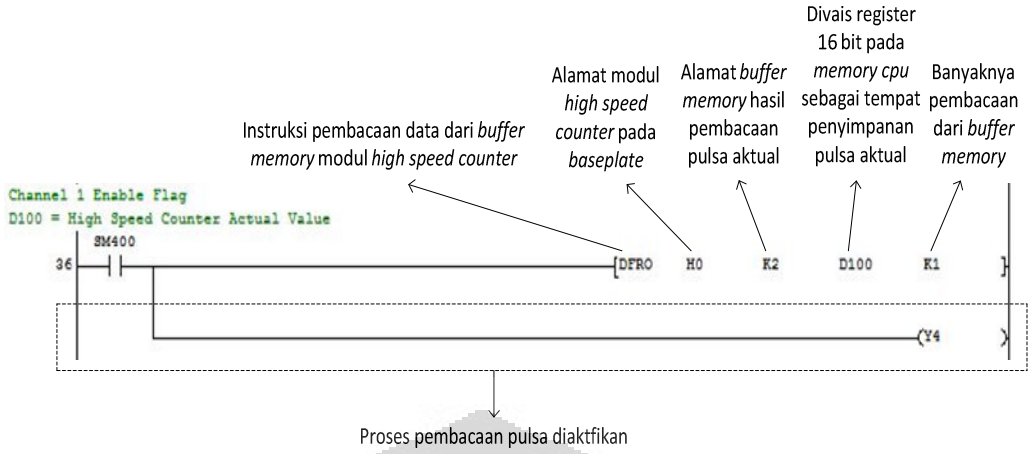
3.3.1 Pemrograman Modul *High Speed Counter*

Pemrograman modul *high speed counter* dibagi kedalam dua tahap, tahap pertama adalah inisialisasi spesifikasi penggunaan kanal pada modul *high speed counter* agar proses pembacaan pulsa keluaran *rotary encoder* dapat dilakukan.



Gambar 3.27 Program inisialisasi penggunaan kanal modul *high speed counter*

Pada Gambar 3.27, parameter penulisan ke *buffer* memori modul *high speed counter* meliputi: alamat modul *high speed counter* adalah H0 (slot pertama pada *base plate*), alamat *buffer* memori yang dituju adalah K0 *(enable/disable channel),* kode instruksi adalah K0 (semua *channel enable*), dan banyak penulisan *buffer* memori adalah K1 (1 *buffer* memori). Tahap kedua adalah *auto refresh*, dimana nilai hasil pembacaan pulsa aktual disimpan didalam divais *register* 16 bit pada memori CPU.

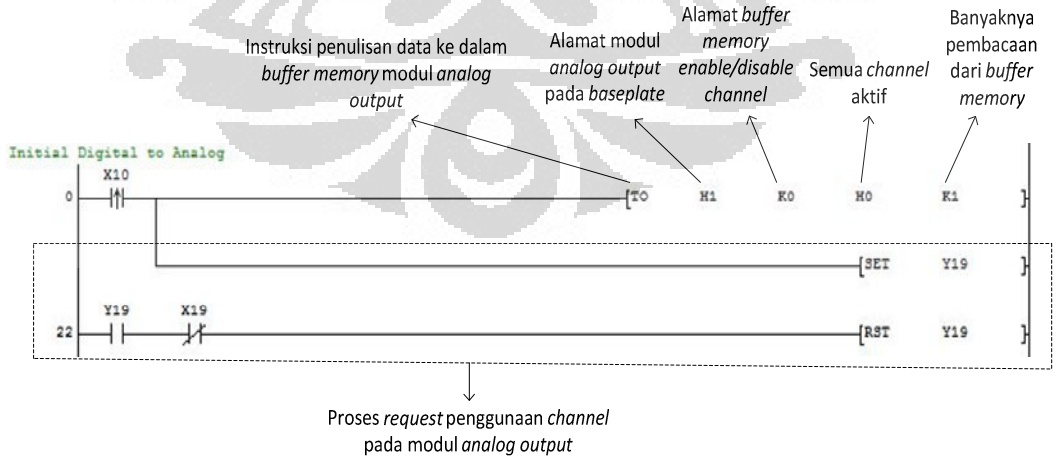


Gambar 3.28 Program *auto refresh* modul *high speed counter*

Pada Gambar 3.28, parameter pembacaan dari *buffer* memori modul *high speed counter* meliputi: alamat modul *high speed counter* adalah H0 (slot pertama pada *base plate*), alamat buffer memori yang dituju adalah K2 *(channel 1 present value), register* 16 bit untuk penyimpanan nilai hasil pembacaan pulsa adalah (**D100)**, dan banyak penulisan buffer memori adalah K1 (1 *buffer* memori).

3.3.2 Pemrograman Modul *Analog Output*

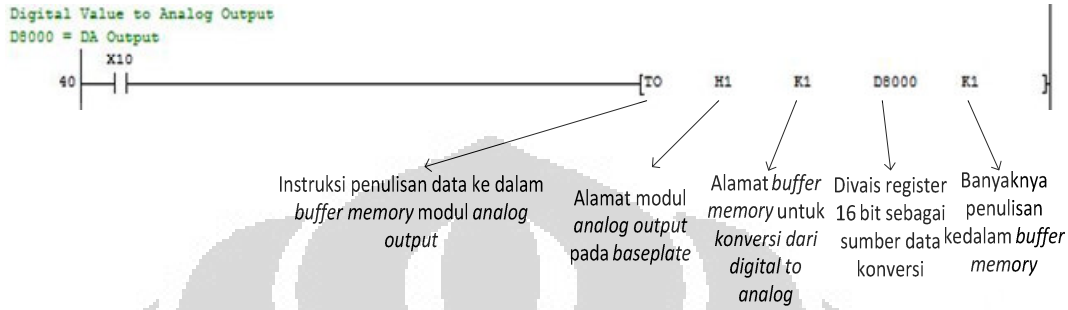
Pemrograman modul *analog output* dibagi kedalam dua tahap, tahap pertama adalah inisialisasi spesifikasi penggunaan kanal pada modul *analog output* agar proses konversi dapat dilakukan.



Gambar 3.29 Program inisialisasi penggunaan kanal *analog output*

Pada Gambar 3.29, parameter penulisan ke *buffer* memori modul *digital to analog* meliputi: alamat modul *digital to analog* adalah H1 (slot kedua pada *base*

plate), alamat *buffer* memori yang dituju adalah K0 *(enable/disable channel),* kode instruksi adalah H0 (semua *channel enable*), dan banyak penulisan *buffer* memori adalah K1 (1 *buffer* memori). Tahap kedua adalah *auto refresh*, dimana nilai digital yang disimpan di dalam *register* 16 bit dikonversi kedalam besaran analog oleh modul *analog output.*



Gambar 3.30 Program *auto refresh analog output*

Pada Gambar 3.30, parameter penulisan ke *buffer* memori modul *digital to analog* meliputi: alamat modul *digital to analog* adalah H1 (slot kedua pada *base plate*), alamat *buffer* memori yang dituju adalah K1 *(channel 1 conversion*), *register* 16 bit untuk konversi nilai digital ke besaran *analog* adalah (**D8000)**, dan banyak penulisan *buffer* memori adalah K1 (1 *buffer* memori). Program untuk mengaktifkan proses konversi dari nilai digital kedalam besaran *analog* pada kanal no. 1 adalah sebagai berikut.

Gambar 3.31 Proses konversi *digital to analog* kanal no. 1 aktif

Pada Gambar 3.31, divais *output* **(Y11)** digunakan sebagai sinyal aktivasi proses konversi. Nilai digital hasil proses defuzzifikasi (0 - 4000) pada *register* 16 bit (**D8000)** selanjutnya akan dikonversi ke dalam besaran *analog* (0 – 10Vdc).

3.3.3 Pemrograman Masukan Error & Derror

Sebelum data pulsa hasil pembacaan modul *high speed counter* dapat diproses ke dalam pemrograman masukan Error & Derror, terlebih dahulu harus di konversi dari format *signed decimal* ke dalam format *real*.

Gambar 3.32 Konversi pulsa aktual dari format *signed decimal* ke *real*

Pada Gambar 3.32, *register* 16 bit (**D100)** di konversi dari format *signed decimal* ke format *real*. Nilai hasil konversi disimpan ke dalam divais *register* 32 bit (**D120)**. Untuk mendukung pemrograman masukan Error & Derror ada beberapa syarat pemrograman yang harus dilakukan, yaitu:

- Pemrograman *Process Value* (PV)
- Pemrograman *Set Point* (SP)

1. **Pemrograman *Process Value* (PV)**

   Nilai *process value* diperoleh dengan cara mengkonversi pulsa hasil pembacaan modul *high speed counter* kedalam format *speed* (rpm). Nilai *speed* dapat diperoleh melalui persamaan berikut.

$$n = \frac{fm \; x \; 60}{res}$$

   Keterangan:

   n = Kecepatan aktual motor (rpm)

   fm = Frekuensi pulsa hasil pembacaan (pulse/sec)

   res = Resolusi dari *rotary encoder* (pulse/revolution)

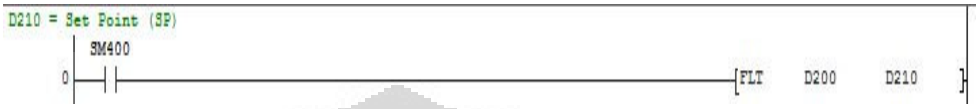
   Pemrograman *process value* dapat dilihat pada gambar dibawah ini.

Gambar 3.33 Pemrograman konversi dari pulsa ke *speed* (rpm)

Pada Gambar 3.33, masukan frekuensi pulsa hasil pembacaan modul *high speed counter* adalah divais *register* 32 bit (**D120**), sedangkan hasil pemrosesan *speed* disimpan ke dalam divais *register* 32 bit (**D160**).

2. Pemrograman *Set Point* (SP)

Nilai *set point* didapatkan dengan cara menginputkan nilai kecepatan yang diinginkan kedalam *register* 16 bit (**D200)** dengan format *integer* melalui modul HMI dengan menggunakan *sofware* SoftGOT1000. Kemudian nilai kecepatan tersebut di konversi kedalam format real seperti pada gambar dibawah ini.



Gambar 3.34 Konversi nilai *set point* dari format *signed decimal* ke *real*

Pada Gambar 3.34, *register* 16 bit (**D200)** di konversi dari format *signed decimal* ke format *real*. Nilai hasil konversi ke dalam format real disimpan kedalam *register* 32 bit (**D210**).

3. Pemrograman Masukan Error

Nilai masukan Error diperoleh dari hasil pengurangan antara nilai *set point* (SP) dengan *present value* (PV). Pemrograman masukan Error dapat dilihat pada gambar berikut.

Gambar 3.35 Pemrograman masukan Error

Pada Gambar 3.35, nilai masukan Error disimpan ke dalam divais *register* 32 bit (**D1000**).

4. Pemrograman Masukan DError

Nilai masukan DError diperoleh dari hasil pengurangan antara nilai Error saat ini dengan Error sebelumnya. Pemrograman masukan DError dapat dilihat pada gambar berikut.

Gambar 3.36 Pemrograman masukan DError

Pada Gambar 3.36, nilai masukan DError disimpan ke dalam divais *register* 32 bit (**D2000**).

3.3.4 Pemrograman Fuzzifikasi

Pada penulisan kali ini, sistem yang dibuat memiliki dua fungsi keanggotaan masukan *fuzzy* (Error & Derror). Pemrograman fuzzifikasi meliputi:

- Pemrograman fungsi keanggotaan masukan Error
- Pemrograman fungsi keanggotaan masukan Derror

3.3.4.1 Pemrograman Fungsi Keanggotaan (FK) Masukan Error

Pemrograman fungsi keanggotaan masukan Error dibagi menjadi tiga subset, yaitu:

- Pemrograman FK masukan Error *subset negative* (N)
- Pemrograman FK masukan Error *subset zero* (Z)
- Pemrograman FK masukan Error *subset positive* (P)

1. **Pemrograman FK Masukan Error *Subset Negative* (N)**

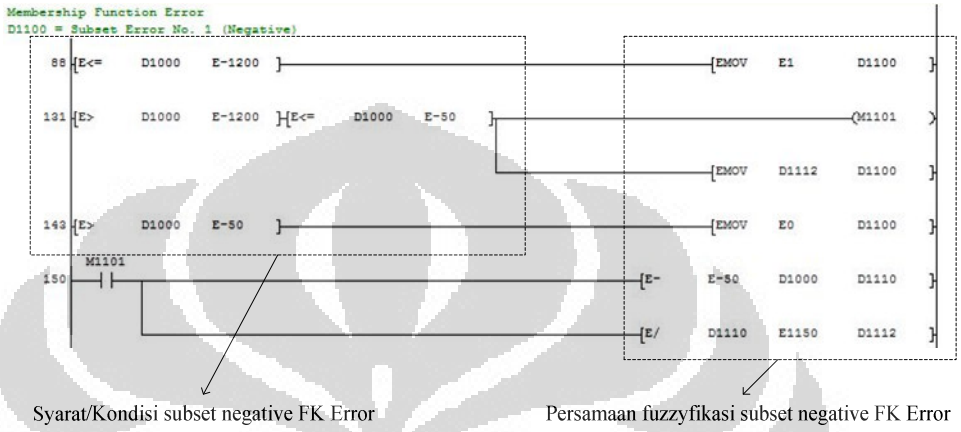
   Pemrograman fungsi keanggotaan masukan Error subset negative harus memenuhi syarat/kondisi dan persamaan fuzzifikasi berikut.

Persamaan	**Syarat/kondisi**
$\mu_N(x1) = 1,$	x1 <= -1200 (rpm)
(x1 + 50) / -1150,	-1200 < x1 <= -50 (rpm)
0,	x1 > -50 (rpm).

Keterangan:

- $\mu_N(x1)$: Nilai/bobot *fuzzy* pada *subset negative* dari FK *input* Error
- x1 : Nilai masukan Error dalam bentuk *speed* (rpm)

Pemrograman *subset negative* dari fungsi keanggotaan *input* Error dapat dilihat pada gambar berikut.



Gambar 3.37 Pemrograman FK masukan Error *subset negative* (N)

Pada Gambar 3.37, nilai fuzzifikasi dari *subset negative* pada fungsi keanggotaan masukan Error disimpan kedalam divais *register* 32 bit (**D1100**).

2. Pemrograman FK Masukan Error *Subset Zero* (Z)

Pemrograman fungsi keanggotaan masukan Error *subset zero* harus memenuhi syarat/kondisi dan persamaan fuzzifikasi berikut.

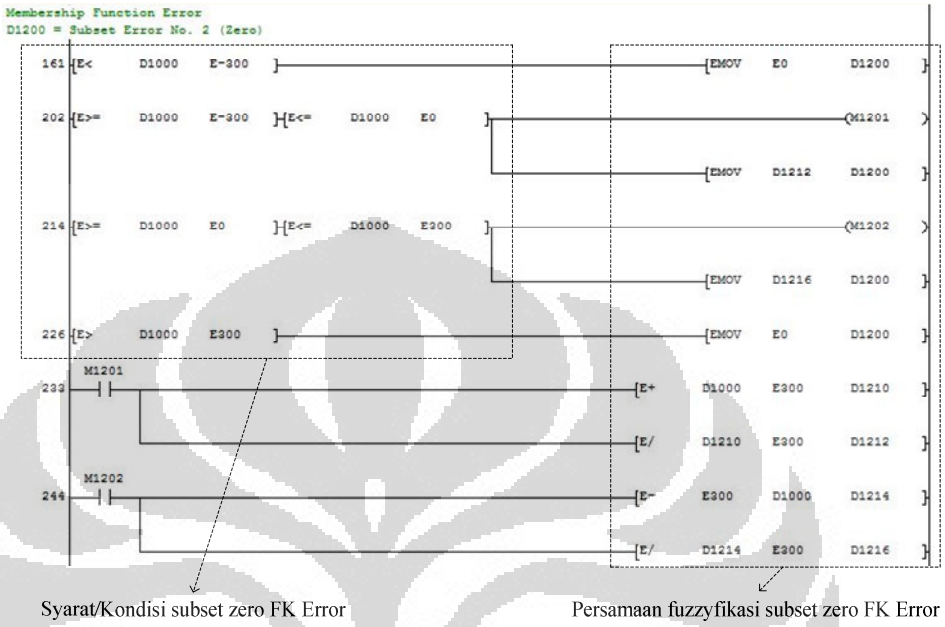
Persamaan	**Syarat/Kondisi**
$\mu_Z(x1) = 0,$	x1 <= -300 (rpm)
(x1 + 300) / 300,	-300 < x1 <= 0 (rpm)
(x1 – 300) / – 300,	0 < x1 <= 300 (rpm)
0,	x1 > 300 (rpm).

Keterangan:

- $\mu_Z(x1)$: Nilai/bobot *fuzzy* pada *subset zero* dari FK *input* Error

- x1 : Nilai masukan Error dalam bentuk *speed* (rpm)

Pemrograman *subset zero* dari fungsi keanggotaan Error dapat dilihat pada gambar berikut.



Gambar 3.38 Pemrograman FK masukan Error *subset zero* (Z)

Pada Gambar 3.38, nilai fuzzifikasi dari *subset negative* pada fungsi keanggotaan masukan Error disimpan kedalam divais *register* 32 bit (**D1200**).

3. Pemrograman FK Masukan Error *Subset Positive* (P)

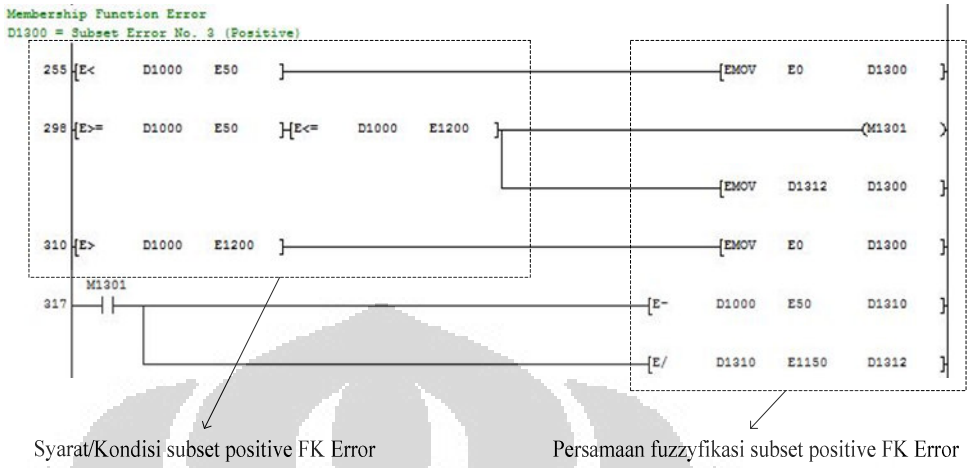
Pemrograman fungsi keanggotaan masukan Error *subset positive* harus memenuhi syarat/kondisi dan persamaan fuzzifikasi berikut.

Persamaan	**Syarat/Kondisi**
$\mu_P(x1) = 0,$	$x1 <= 50$ (rpm)
$(x1 - 50) / 1150,$	$50 < x1 <= 1200$ (rpm)
$1,$	$x1 > 1200$ (rpm).

Keterangan:

- $\mu_P(x1)$: Nilai/bobot *fuzzy* dari *subset positive* pada FK *input* Error
- x1 : Nilai masukan Error dalam bentuk *speed* (rpm)

Pemrograman *subset positive* dari fungsi keanggotaan masukan Error dapat dilihat pada gambar berikut.



Gambar 3.39 Pemrograman FK masukan Error *subset positive* (P)

Pada Gambar 3.39, nilai fuzzifikasi dari *subset positive* pada fungsi keanggotaan masukan Error disimpan kedalam divais *register* 32 bit (**D1300**).

3.3.4.2 Pemrograman Fungsi Keanggotaan (FK) Masukan DError

Pemrograman fungsi keanggotaan masukan DError dibagi menjadi tiga subset, yaitu:

- Pemrograman FK masukan DError *subset negative* (N)
- Pemrograman FK masukan DError *subset zero* (Z)
- Pemrograman FK masukan DError *subset positive* (P)

1. Pemrograman FK Masukan DError *Subset Negative* (N)

Pemrograman fungsi keanggotaan masukan DError *subset negative* harus memenuhi syarat/kondisi dan persamaan fuzzifikasi berikut.

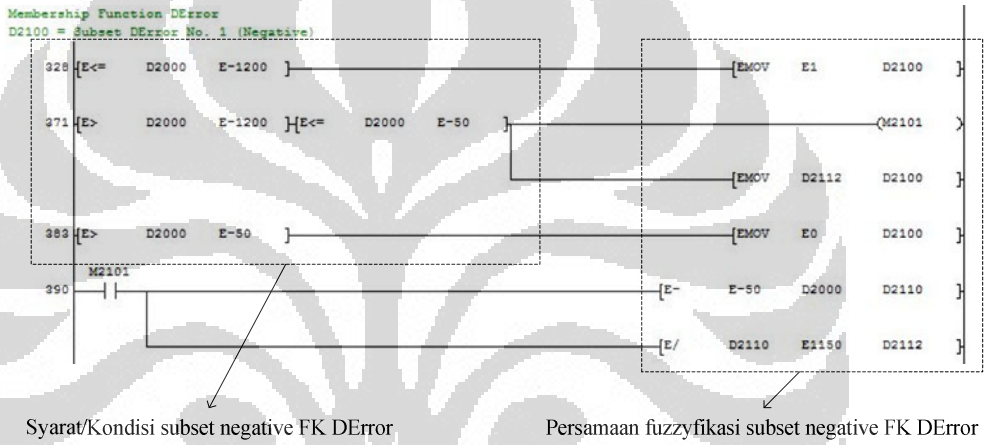
Persamaan	**Syarat/kondisi**
$\mu_N(x2) = 1,$	x2 <= -1200 (rpm)

(x2 + 50) / -1150,	-1200 < x2 <= -50 (rpm)
0,	x2 > -50 (rpm).

Keterangan:

- $\mu_N(x2)$: Nilai/bobot *fuzzy* dari *subset negative* pada FK *input* DError
- x2 : Nilai masukan DError dalam bentuk *speed* (rpm)

Pemrograman subset negative dari fungsi keanggotaan DError dapat dilihat pada gambar berikut.



Gambar 3.40 Pemrograman FK masukan DError *subset negative* (N)

Pada Gambar 3.40, nilai fuzzifikasi dari *subset negative* pada fungsi keanggotaan masukan DError disimpan kedalam divais *register* 32 bit (**D2100**).

2. Pemrograman FK Masukan DError *Subset Zero* (Z)

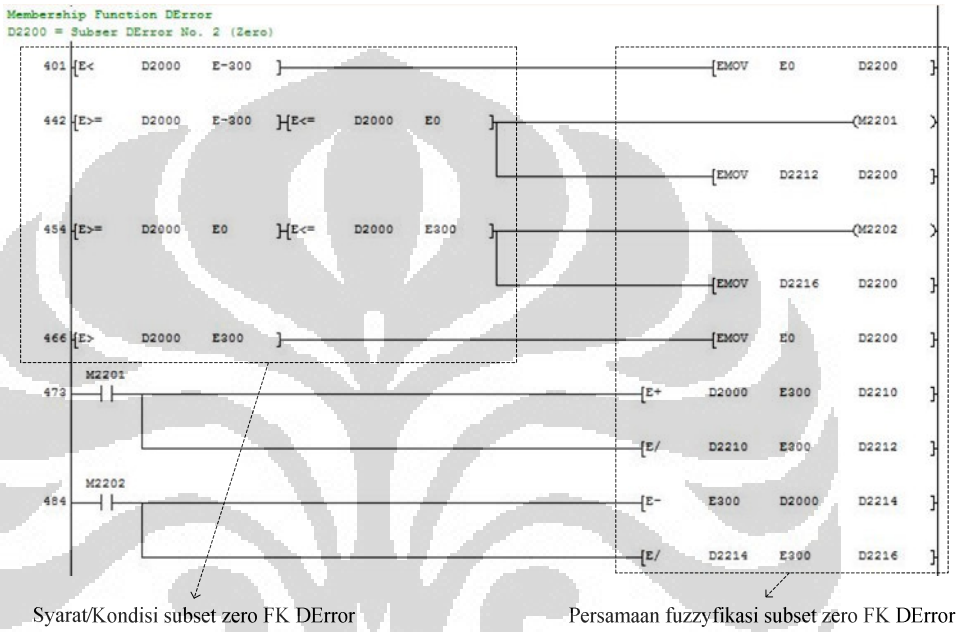
Pemrograman fungsi keanggotaan masukan DError *subset zero* harus memenuhi syarat/kondisi dan persamaan fuzzifikasi berikut.

Persamaan	**Syarat/Kondisi**
$\mu_Z(x2) = 0$,	x2 <= -300 (rpm)
(x2 + 300) / 300,	-300 < x2 <= 0 (rpm)
(x2 – 300) / – 300,	0 < x2 <= 300 (rpm)
0,	x2 > 300 (rpm).

Keterangan:

- $\mu_Z(x2)$: Nilai/bobot *fuzzy* dari *subset zero* pada FK *input* DError
- x2 : Nilai masukan DError dalam bentuk *speed* (rpm)

Pemrograman *subset zero* dari fungsi keanggotaan DError dapat dilihat pada gambar berikut.



Gambar 3.41 Pemrograman FK masukan DError *subset zero* (Z)

Pada Gambar 3.41, nilai fuzzifikasi dari *subset negative* pada fungsi keanggotaan masukan DError disimpan kedalam divais *register* 32 bit (**D2200**).

3. Pemrograman FK Masukan DError *Subset Positive* (P)

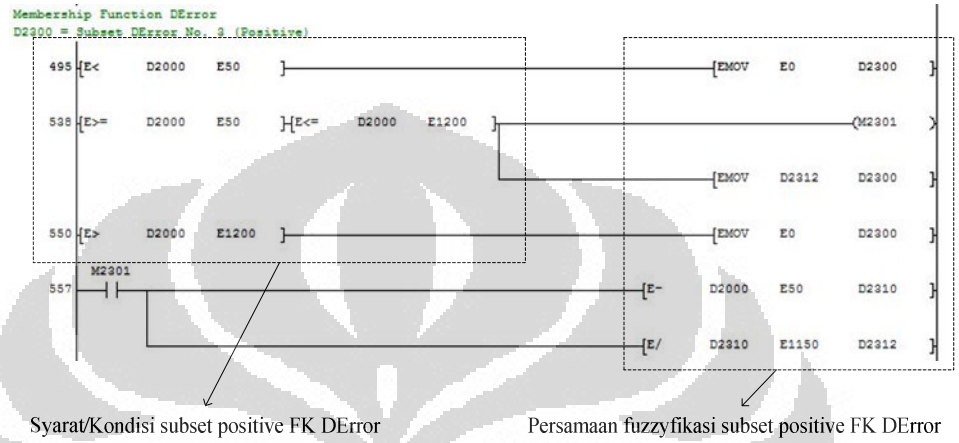
Pemrograman fungsi keanggotaan masukan DError *subset positive* harus memenuhi syarat/kondisi dan persamaan fuzzifikasi berikut.

Persamaan	**Syarat/Kondisi**
$\mu_P(x2) = 0,$	x2 <= 50 (rpm)
(x2 – 50) / 1150,	50 < x2 <= 1200 (rpm)
1,	x2 > 1200 (rpm).

Keterangan:

- $\mu_P(x2)$: Nilai/bobot *fuzzy* pada *subset positive* dari FK DError
- x2 : Nilai masukan DError dalam bentuk *speed* (rpm)

Pemrograman *subset positive* dari fungsi keanggotaan masukan DError dapat dilihat pada gambar berikut.

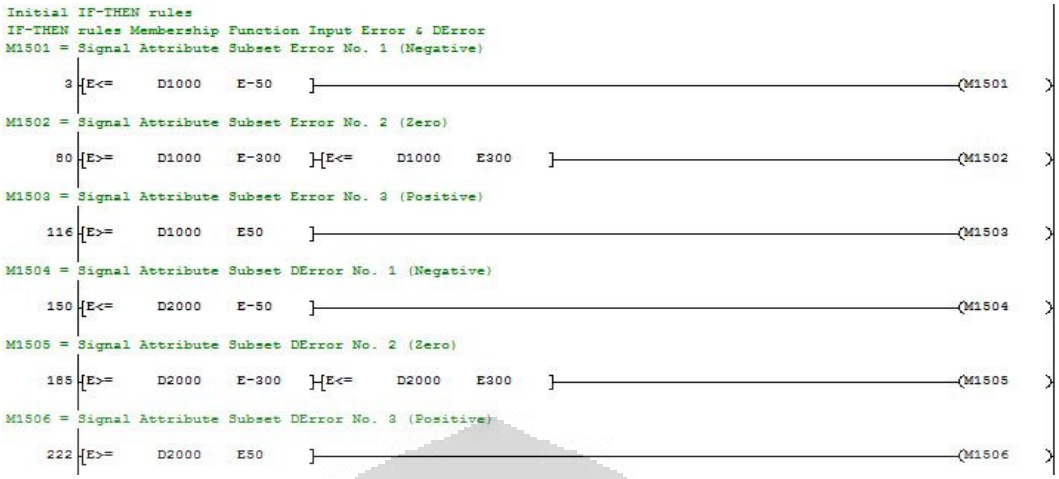


Gambar 3.42 Pemrograman FK masukan DError *subset positive* (P)

Pada Gambar 3.42, nilai fuzzifikasi dari *subset positive* pada fungsi keanggotaan masukan DError disimpan kedalam divais *register* 32 bit (**D2300**).

3.3.5 Pemrograman Basis Aturan *Fuzzy*

Pemrograman basis aturan *fuzzy* pada sistem ini, memiliki sembilan aturan yang digunakan sebagai referensi dalam penentuan *subset-subset* dari fungsi keanggotaan keluaran sebelum dilakukan proses inferensi. Sinyal-sinyal didalam pemrograman basis aturan, ditandai oleh sebuah divais *internal relay* (**M**) yang berbentuk bit. Pemrograman inisialisasi dari *subset-subset* fungsi keanggotaan masukan Error & Derror dapat dilihat pada gambar berikut.



Gambar 3.43 Inisialisasi *subset* FK Error & Derror

Berdasarkan Gambar 3.43, pada fungsi keanggotaan masukan Error: *subset negative* ditandai oleh divais (**M1501**), *subset zero* ditandai oleh divais (**M1502**), dan *subset positive* ditandai oleh divais (**M1503**). Pada fungsi keanggotaan masukan DError: *subset negative* ditandai oleh divais (**M1504**), *subset zero* ditandai oleh divais (**M1505**), dan *subset positive* ditandai oleh divais (**M1506**).

1. **Pemrograman Aturan *Fuzzy* No. 1**

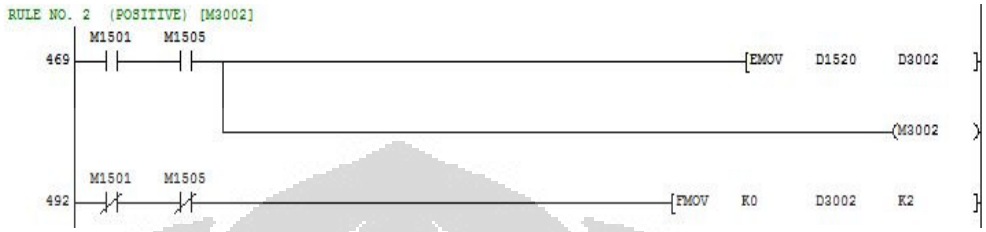
   Pemrograman aturan *fuzzy* no. 1 mengacu kepada Tabel 3.1 memiliki pernyataan matematis, yaitu: JIKA Error adalah *POS (P)* & DError adalah *NEG (N)* MAKA DV adalah *NEG (N)*. Pemrograman aturan *fuzzy* no. 1 dapat dilihat pada gambar berikut.

Gambar 3.44 Pemrograman aturan *fuzzy* no. 1

Pada Gambar 3.44, sinyal aktivasi untuk aturan *fuzzy* no. 1 ditandai oleh divais berbentuk bit (**M3000**) dan divais *register* 32 bit (**D3000**) yang digunakan sebagai transfer nilai *fuzzy*.

2. Pemrograman Aturan *Fuzzy* No. 2

Pemrograman aturan *fuzzy* no. 2 mengacu kepada Tabel 3.1 memiliki pernyataan matematis, yaitu: JIKA Error adalah *NEG (N)* & DError adalah *NEG (N)* MAKA DV adalah *POS (P).* Pemrograman aturan *fuzzy* no. 2 dapat dilihat pada gambar berikut.



Gambar 3.45 Pemrograman aturan *fuzzy* no. 2

Pada Gambar 3.45, sinyal aktivasi untuk aturan *fuzzy* no. 2 ditandai oleh divais berbentuk bit (**M3002**) dan divais *register* 32 bit (**D3002**) yang digunakan sebagai transfer nilai *fuzzy*.

3. Pemrograman Aturan *Fuzzy* No. 3

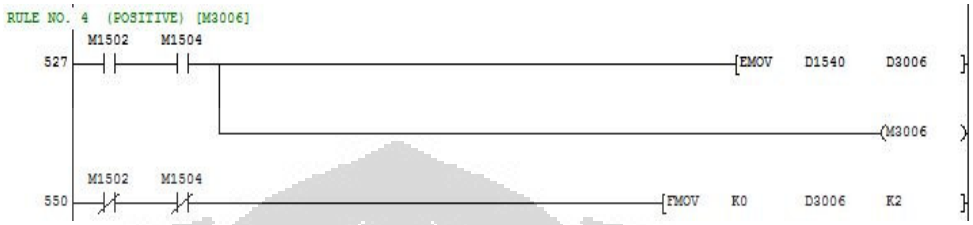
Pemrograman aturan *fuzzy* no. 3 mengacu kepada Tabel 3.1 memiliki pernyataan matematis, yaitu: JIKA Error adalah *NEG (N)* & DError adalah *POS (P)* MAKA DV adalah *POS (P).* Pemrograman aturan *fuzzy* no. 3 dapat dilihat pada gambar berikut.

Gambar 3.46 Pemrograman aturan *fuzzy* no. 3

Pada Gambar 3.46, sinyal aktivasi untuk aturan *fuzzy* no. 3 ditandai oleh divais berbentuk bit (**M3004**) dan divais *register* 32 bit (**D3004**) yang digunakan sebagai transfer nilai *fuzzy*.

4. Pemrograman Aturan *Fuzzy* No. 4

Pemrograman aturan *fuzzy* no. 4 mengacu kepada Tabel 3.1 memiliki pernyataan matematis, yaitu: JIKA Error adalah *POS (P)* & DError adalah *POS (P)* MAKA DV adalah *NEG (N).* Pemrograman aturan *fuzzy* no. 4 dapat dilihat pada gambar berikut.



Gambar 3.47 Pemrograman aturan *fuzzy* no. 4

Pada Gambar 3.47, sinyal aktivasi untuk aturan *fuzzy* no. 4 ditandai oleh divais berbentuk bit (**M3006**) dan divais *register* 32 bit (**D3006**) yang digunakan sebagai transfer nilai *fuzzy*.

5. Pemrograman Aturan *Fuzzy* No. 5

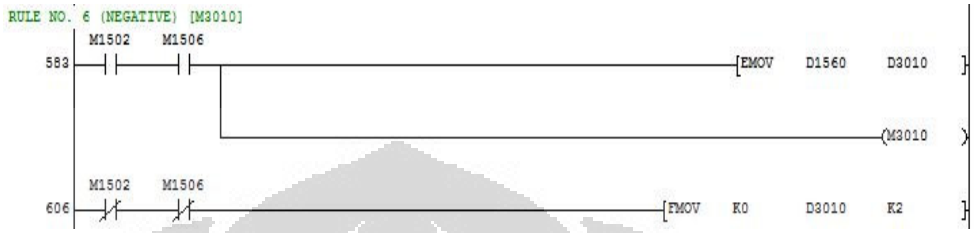
Pemrograman aturan *fuzzy* no. 5 mengacu kepada Tabel 3.1 memiliki pernyataan matematis, yaitu: JIKA Error adalah *POS (P)* & DError adalah *ZERO (Z)* MAKA DV adalah *NEG (N).* Pemrograman aturan *fuzzy* no. 5 dapat dilihat pada gambar berikut.

Gambar 3.48 Pemrograman aturan *fuzzy* no. 5

Pada Gambar 3.48, sinyal aktivasi untuk aturan *fuzzy* no. 5 ditandai oleh divais berbentuk bit (**M3008**) dan divais *register* 32 bit (**D3008**) yang digunakan sebagai transfer nilai *fuzzy*.

6. Pemrograman Aturan *Fuzzy* No. 6

Pemrograman aturan *fuzzy* no. 6 mengacu kepada Tabel 3.1 memiliki pernyataan matematis, yaitu: JIKA Error adalah *NEG (N)* & DError adalah *ZERO (Z)* MAKA DV adalah *POS (P).* Pemrograman aturan *fuzzy* no. 6 dapat dilihat pada gambar berikut.



Gambar 3.49 Pemrograman aturan *fuzzy* no. 6

Pada Gambar 3.49, sinyal aktivasi untuk aturan *fuzzy* no. 6 ditandai oleh divais berbentuk bit (**M3010**) dan divais *register* 32 bit (**D3010**) yang digunakan sebagai transfer nilai *fuzzy*.

7. Pemrograman Aturan *Fuzzy* No. 7

Pemrograman aturan *fuzzy* no. 7 mengacu kepada Tabel 3.1 memiliki pernyataan matematis, yaitu: JIKA Error adalah *ZERO (Z)* & DError adalah *POS (P)* MAKA DV adalah *POS (P).* Pemrograman aturan *fuzzy* no. 7 dapat dilihat pada gambar berikut.

Gambar 3.50 Pemrograman aturan *fuzzy* no. 7

Pada Gambar 3.50, sinyal aktivasi untuk aturan *fuzzy* no. 7 ditandai oleh divais berbentuk bit (**M3012**) dan divais *register* 32 bit (**D3012**) yang digunakan sebagai transfer nilai *fuzzy*.

8. Pemrograman Aturan *Fuzzy* No. 8

Pemrograman aturan *fuzzy* no. 8 mengacu kepada Tabel 3.1 memiliki pernyataan matematis, yaitu: JIKA Error adalah *ZERO (Z)* & DError adalah *NEG (N)* MAKA DV adalah *POS (P).* Pemrograman aturan *fuzzy* no. 8 dapat dilihat pada gambar berikut.

Gambar 3.51 Pemrograman aturan *fuzzy* no. 8

Pada Gambar 3.51, sinyal aktivasi untuk aturan *fuzzy* no. 8 ditandai oleh divais berbentuk bit (**M3014**) dan divais *register* 32 bit (**D3014**) yang digunakan sebagai transfer nilai *fuzzy*.

9. Pemrograman Aturan *Fuzzy* No. 9

Pemrograman aturan *fuzzy* no. 9 mengacu kepada Tabel 3.1 memiliki pernyataan matematis, yaitu: JIKA Error adalah *ZERO (Z)* & DError adalah *ZERO (Z)* MAKA DV adalah *ZERO (Z).* Pemrograman aturan *fuzzy* no. 9 dapat dilihat pada gambar berikut.

Gambar 3.52 Pemrograman aturan *fuzzy* no. 9

Pada Gambar 3.52, sinyal aktivasi untuk aturan *fuzzy* no. 9 ditandai oleh divais berbentuk bit (**M3016**) dan divais *register* 32 bit (**D3016**) yang digunakan sebagai transfer nilai *fuzzy*.

3.3.6 Pemrograman Inferensi & Fungsi Keanggotaan Keluaran

Pemrograman inferensi bertujuan untuk mencari nilai *minimum* dari korelasi dua fungsi keanggotaan masukan *fuzzy* (Error & DError) serta mencari nilai *maximum* dari korelasi antara hasil masing-masing nilai *minimum* terhadap fungsi keanggotaan keluaran *fuzzy* DV berdasarkan basis aturan *fuzzy*. Sedangkan pemrograman fungsi keanggotaan keluaran bertujuan untuk mengubah nilai-nilai *fuzzy* dari proses inferensi kedalam bentuk keluaran tunggal yang tegas sesuai dengan *plot* dari *subset-subset* fungsi keanggotaan keluaran *fuzzy* DV.

3.3.6.1 Pemrograman Inferensi

1. **Pemrograman *Minimum* Aturan *Fuzzy* No. 1 (c1)**

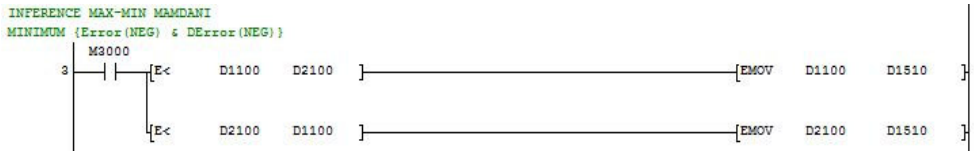
   Nilai *minimum* dari aturan *fuzzy* no. 1 (c1) mengacu kepada Gambar 3.7, diperoleh dari korelasi nilai-nilai *fuzzy* antara *subset positive* (P) FK masukan Error dan *subset negative* (N) FK masukan DError. Pemrograman *minimum* dari aturan *fuzzy* no. 1 dapat dilihat pada gambar berikut.

Gambar 3.53 Pemrograman *minimum* aturan *fuzzy* no. 1 (c1)

Pada Gambar 3.53, nilai *minimum* hasil proses inferensi aturan *fuzzy* no. 1 disimpan kedalam divais *register* 32 bit (**D1510**).

2. **Pemrograman *Minimum* Aturan *Fuzzy* No. 2 (c2)**

   Nilai *minimum* dari aturan *fuzzy* no. 2 (c2) mengacu kepada Gambar 3.7, diperoleh dari korelasi nilai-nilai *fuzzy* antara *subset negative* (N) FK masukan Error dan *subset negative* (N) FK masukan DError. Pemrograman *minimum* dari aturan *fuzzy* no. 2 dapat dilihat pada gambar berikut.



Gambar 3.54 Pemrograman *minimum* aturan *fuzzy* no. 2 (c2)

Pada Gambar 3.54, nilai *minimum* hasil proses inferensi aturan *fuzzy* no. 2 disimpan kedalam divais *register* 32 bit (**D1520**).

3. Pemrograman *Minimum* Aturan *Fuzzy* No. 3 (c3)

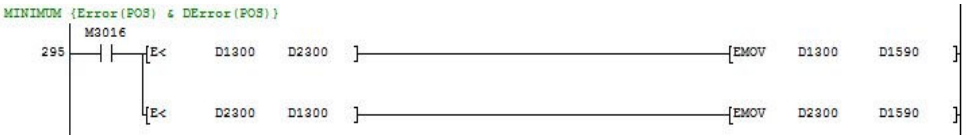
Nilai *minimum* dari aturan *fuzzy* no. 3 (c3) mengacu kepada Gambar 3.7, diperoleh dari korelasi nilai-nilai *fuzzy* antara *subset negative* (N) FK masukan Error dan *subset positive* (P) FK masukan DError. Pemrograman *minimum* dari aturan *fuzzy* no. 3 dapat dilihat pada gambar berikut.

Gambar 3.55 Pemrograman *minimum* aturan *fuzzy* no. 3 (c3)

Pada Gambar 3.55, nilai *minimum* hasil proses inferensi aturan *fuzzy* no. 3 disimpan kedalam divais *register* 32 bit (**D1530**).

4. Pemrograman *Minimum* Aturan *Fuzzy* No. 4 (c4)

Nilai *minimum* dari aturan *fuzzy* no. 4 (c4) mengacu kepada Gambar 3.7, diperoleh dari korelasi nilai-nilai *fuzzy* antara *subset positive* (P) FK masukan Error dan *subset positive* (P) FK masukan DError. Pemrograman minimum dari aturan *fuzzy* no. 4 dapat dilihat pada gambar berikut.



Gambar 3.56 Pemrograman *minimum* aturan *fuzzy* no. 4 (c4)

Pada Gambar 3.56, nilai *minimum* hasil proses inferensi aturan *fuzzy* no. 4 disimpan kedalam divais *register* 32 bit (**D1540**).

5. Pemrograman *Minimum* Aturan *Fuzzy* No. 5 (c5)

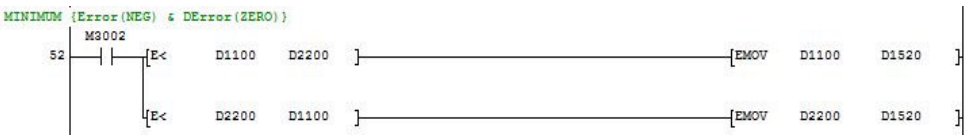
Nilai *minimum* dari aturan *fuzzy* no. 5 (c5) mengacu kepada Gambar 3.7, diperoleh dari korelasi nilai-nilai *fuzzy* antara *subset positive* (P) FK masukan Error dan *subset zero* (Z) FK masukan DError. Pemrograman *minimum* dari aturan *fuzzy* no. 5 dapat dilihat pada gambar berikut.

Gambar 3.57 Pemrograman *minimum* aturan *fuzzy* no. 5 (c5)

Pada Gambar 3.57, nilai *minimum* hasil proses inferensi aturan *fuzzy* no. 5 disimpan kedalam divais *register* 32 bit (**D1550**).

6. Pemrograman *Minimum* Aturan *Fuzzy* No. 6 (c6)

Nilai *minimum* dari aturan *fuzzy* no. 6 (c6) mengacu kepada Gambar 3.7, diperoleh dari korelasi nilai-nilai *fuzzy* antara *subset negative* (N) FK masukan Error dan *subset zero* (Z) FK masukan DError. Pemrograman *minimum* dari aturan *fuzzy* no. 6 dapat dilihat pada gambar berikut.



Gambar 3.58 Pemrograman *minimum* aturan *fuzzy* no. 6 (c6)

Pada Gambar 3.58, nilai *minimum* hasil proses inferensi aturan fuzzy no. 6 disimpan kedalam divais *register* 32 bit (**D1560**).

7. Pemrograman *Minimum* Aturan *Fuzzy* No. 7 (c7)

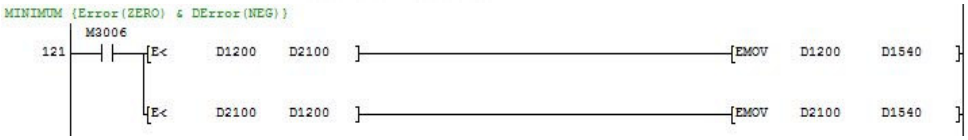
Nilai *minimum* dari aturan *fuzzy* no. 7 (c7) mengacu kepada Gambar 3.7, diperoleh dari korelasi nilai-nilai *fuzzy* antara *subset zero* (Z) FK masukan Error dan *subset positive* (P) FK masukan DError. Pemrograman *minimum* dari aturan *fuzzy* no. 7 dapat dilihat pada gambar berikut.

Gambar 3.59 Pemrograman *minimum* aturan *fuzzy* no. 7 (c7)

Pada Gambar 3.59, nilai *minimum* hasil proses inferensi aturan fuzzy no. 7 disimpan kedalam divais *register* 32 bit (**D1570**).

8. Pemrograman *Minimum* Aturan *Fuzzy* No. 8 (c8)

Nilai minimum dari aturan fuzzy no. 8 (c8) mengacu kepada Gambar 3.7, diperoleh dari korelasi nilai-nilai *fuzzy* antara *subset zero* (Z) FK masukan Error dan *subset negative* (N) FK masukan DError. Pemrograman minimum dari aturan fuzzy no. 8 dapat dilihat pada gambar berikut.



Gambar 3.60 Pemrograman *minimum* aturan *fuzzy* no. 8 (c8)

Pada Gambar 3.60, nilai *minimum* hasil proses inferensi aturan *fuzzy* no. 8 disimpan kedalam divais *register* 32 bit (**D1580**).

9. Pemrograman *Minimum* Aturan *Fuzzy* No. 9 (c9)

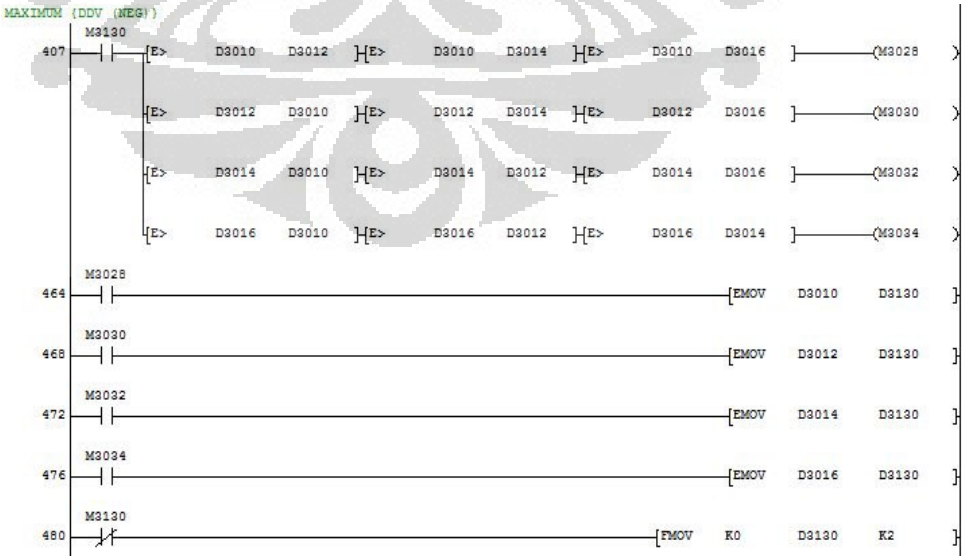
Nilai *minimum* dari aturan *fuzzy* no. 9 (c9) mengacu kepada Gambar 3.7, diperoleh dari korelasi nilai-nilai *fuzzy* antara *subset zero* (Z) FK masukan Error dan *subset zero* (Z) FK masukan DError. Pemrograman *minimum* dari aturan *fuzzy* no. 9 dapat dilihat pada gambar berikut.

Gambar 3.61 Pemrograman *minimum* aturan *fuzzy* no. 9

Pada Gambar 3.61, nilai *minimum* hasil proses inferensi aturan *fuzzy* no. 9 disimpan kedalam divais *register* 32 bit (**D1590**).

10. Pemrograman *Maximum* Aturan *Fuzzy* No. 1 (c1, c4, c5, c7)

Nilai *maximum* no. 1 (c1, c4, c5, c7) mengacu kepada Gambar 3.8, diperoleh dari korelasi hasil nilai-nilai *minimum* yang memiliki keluaran sama terhadap FK keluaran *fuzzy* DV berdasarkan basis aturan *fuzzy* (**c1, c4, c5, c7**). Pemrograman aturan *fuzzy* no. 1 dapat dilihat pada gambar berikut.

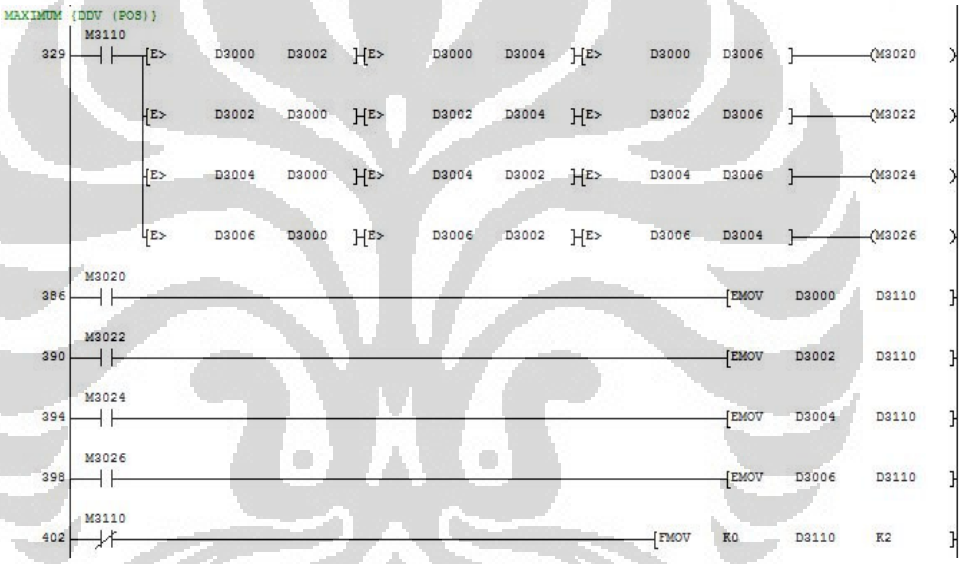


Gambar 3.62 Pemrograman *maximum* aturan *fuzzy* no. 1 (c1, c4, c5, c7)

Pada Gambar 3.62, nilai *maximum* hasil proses inferensi (**c1, c4, c5, c7**) disimpan ke dalam divais *register* 32 bit (**D3110**).

11. Pemrograman *Maximum* Aturan *Fuzzy* No. 2 (c2, c3, c6, c8)

Nilai *maximum* no. 2 (c2, c3, c6, c8) mengacu kepada Gambar 3.8, diperoleh dari korelasi hasil nilai-nilai *minimum* yang memiliki keluaran sama terhadap FK keluaran *fuzzy* DV berdasarkan basis aturan *fuzzy* (**c2, c3, c6, c8**). Pemrograman aturan *fuzzy* no. 2 dapat dilihat pada gambar berikut.



Gambar 3.63 Permrograman *maximum* aturan *fuzzy* no. 2 (c2, c3, c6, c8)

Pada Gambar 3.63, nilai *maximum* hasil proses inferensi (**c2, c3, c6, c8**) disimpan ke dalam divais *register* 32 bit (**D3130**).

3.3.6.2 Pemrograman Fungsi Keanggotaan (FK) Keluaran DV

Pemrograman fungsi keanggotaan keluaran DV dibagi menjadi tiga subset, yaitu:

- Pemrograman FK keluaran DV *subset negative* (N)
- Pemrograman FK keluaran DV *subset zero* (Z)

- Pemrograman FK keluaran DV *subset positive* (P)

1. Pemrograman FK Keluaran DV *Subset Negative* (N)

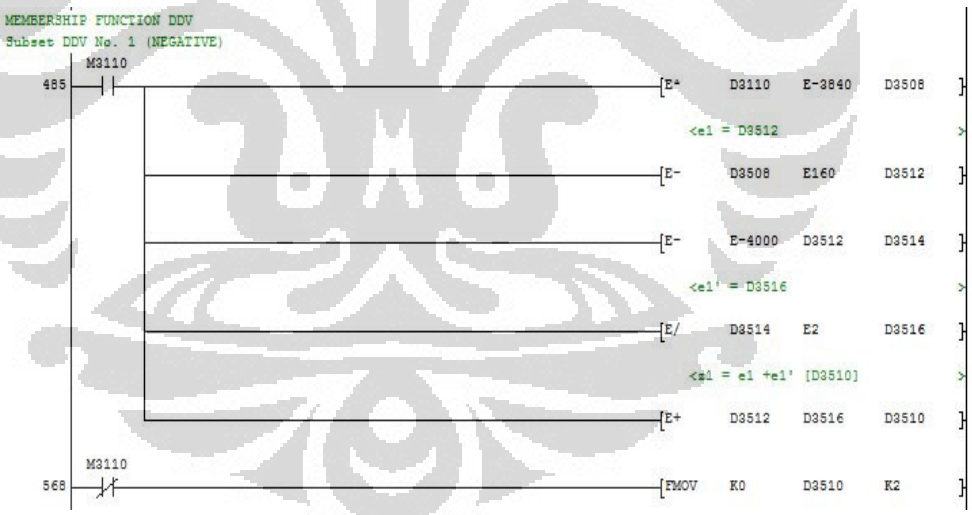
Masukan pada pemrograman fungsi keanggotaan keluaran DV *subset negative* (N) berasal dari inferensi *maximum* aturan *fuzzy* (**c1, c4, c5, c7**) mengacu kepada Gambar 3.8. Persamaan matematis dari fungsi keanggotaan keluaran DV *subset negative* adalah sebagai berikut.

$$z1 = -(\mu_N(y) * 3840) - 160$$

$$z1' = (-4000 - z1) / 2$$

$$z_N = z1 + z1'$$

Pemrograman *subset negative* pada fungsi keanggotaan keluaran DV dapat dilihat pada gambar berikut.



Gambar 3.64 Pemrograman FK keluaran DV *subset negative* (N)

Pada gambar 3.64, nilai tegas dari *subset negative* fungsi keanggotaan keluaran DV disimpan kedalam divais *register* 32 bit (**D3510**).

2. Pemrograman FK Keluaran DV *Subset Zero* (Z)

Masukan pada pemrograman fungsi keanggotaan keluaran DV *subset zero* (Z) berasal dari inferensi *minimum* aturan *fuzzy* **(c9)** mengacu kepada Gambar 3.8. Persamaan matematis dari fungsi keanggotaan keluaran DV *subset zero* adalah sebagai berikut.

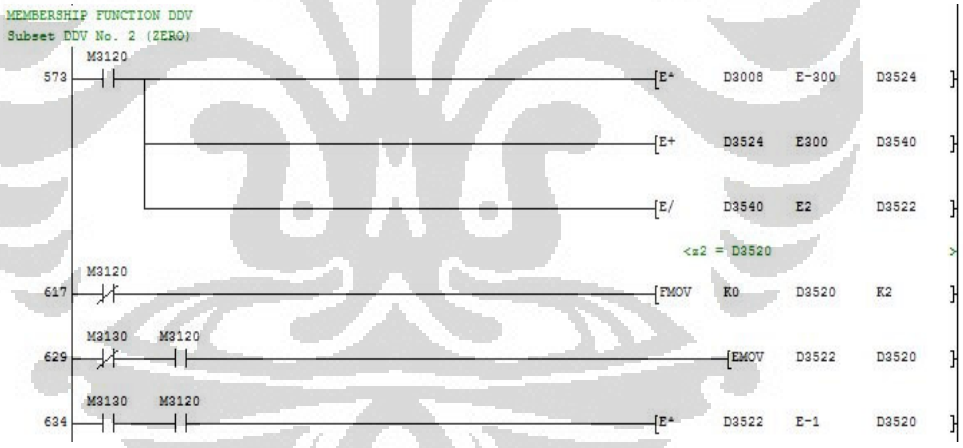
1. Jika sinyal negatif

$$z_Z = ((\mu_Z(y) * 800) - 800) / 2$$

2. Jika sinyal positif

$$z_Z = - ((\mu_Z(Y) * 800) + 800) / 2$$

Pemrograman *subset zero* pada fungsi keanggotaan keluaran DV dapat dilihat pada gambar berikut.



Gambar 3.65 Pemrograman FK keluaran DV *subset zero* (Z)

Pada Gambar 3.65, nilai tegas dari *subset zero* fungsi keanggotaan keluaran DV disimpan kedalam divais *register* 32 bit (**D3520**).

3. Pemrograman FK Keluaran DV *Subset Positive* (P)

Masukan pada pemrograman fungsi keanggotaan keluaran DV *subset positive* (P) berasal dari inferensi *maximum* aturan *fuzzy* (**c6, c7, c8,**

c9) mengacu kepada Gambar 3.8. Persamaan matematis dari fungsi keanggotan keluaran DV *subset positive* adalah sebagai berikut.

$$z3 = (\mu_P(y) * 3840) + 160$$

$$z3' = (4000 - z3) / 2$$

$$z_P = z3 + z3'$$

Pemrograman *subset positive* pada fungsi keanggotaan keluaran DV dapat dilihat pada gambar berikut.

Gambar 3.66 Pemrograman FK keluaran DV *subset positive* (P)

Pada Gambar 3.66, nilai tegas dari *subset positive* fungsi keanggotaan keluaran DV disimpan kedalam divais *register* 32 bit (**D3530**).

3.3.7 Pemrograman Defuzzifikasi

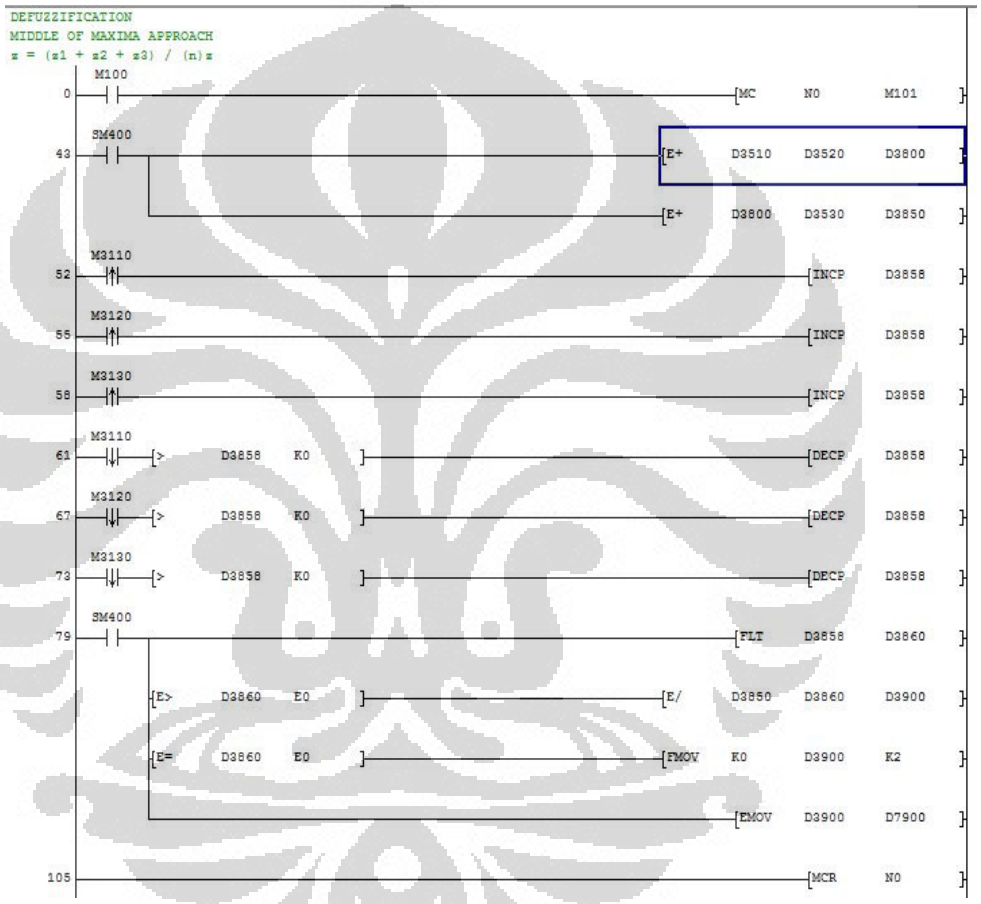
Pada penulisan kali ini, sistem yang dibuat menggunakan dua metode defuzzyfikasi (*weighted average* & *middle of maxima*) sebagai acuan perbandingan respon *speed* motor.

1. Pemrograman Metode Defuzzyfikasi *Middle of Maxima*

Pemrograman metode defuzzyfikasi *middle of maxima* harus memenuhi persamaan matematis berikut.

$$z = \frac{Z1 + Z2 + Z3 + \cdots + Zn}{(1+2+3+\cdots+n)}$$

pemrograman metode defuzzyfikasi *middle of maxima* dapat dilihat pada gambar berikut.



Gambar 3.67 Pemrograman metode defuzzifikasi *middle of maxima*

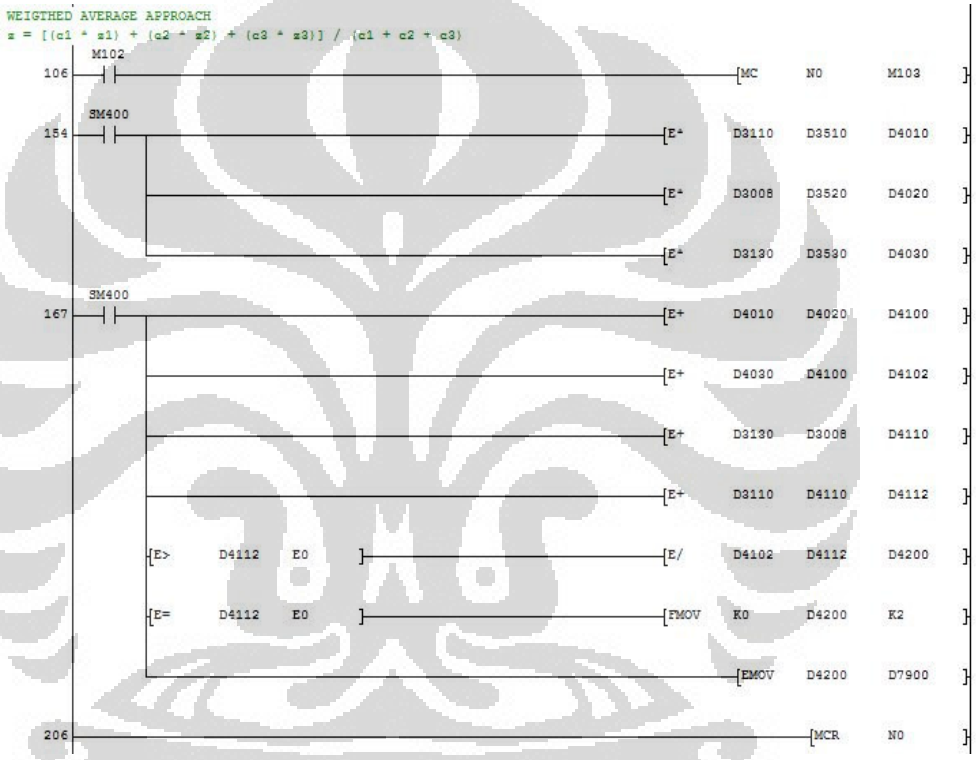
Pada Gambar 3.67, nilai hasil proses defuzzifikasi dengan menggunakan metode *middle of maxima* disimpan kedalam divais *register* 32 bit (**D3900**).

2. Pemrograman Metode Defuzzyfikasi *Weighted Average*

Pemrograman metode defuzzyfikasi *weighted average* harus memenuhi persamaan matematis berikut.

$$z = \frac{(c1 * z1) + (c2 * z2) + (c3 * z3) + (cn * zn)}{(c1 + c2 + c3 + cn)}$$

Pemrograman metode defuzzyfikasi *weigthed average* dapat dilihat pada gambar berikut.



Gambar 3.68 Pemrograman metode defuzzifikasi *weigthed average*

Pada Gambar 3.68, nilai hasil proses defuzzifikasi dengan metode *weighted average* disimpan kedalam divais *register* 32 bit (**D4200**).

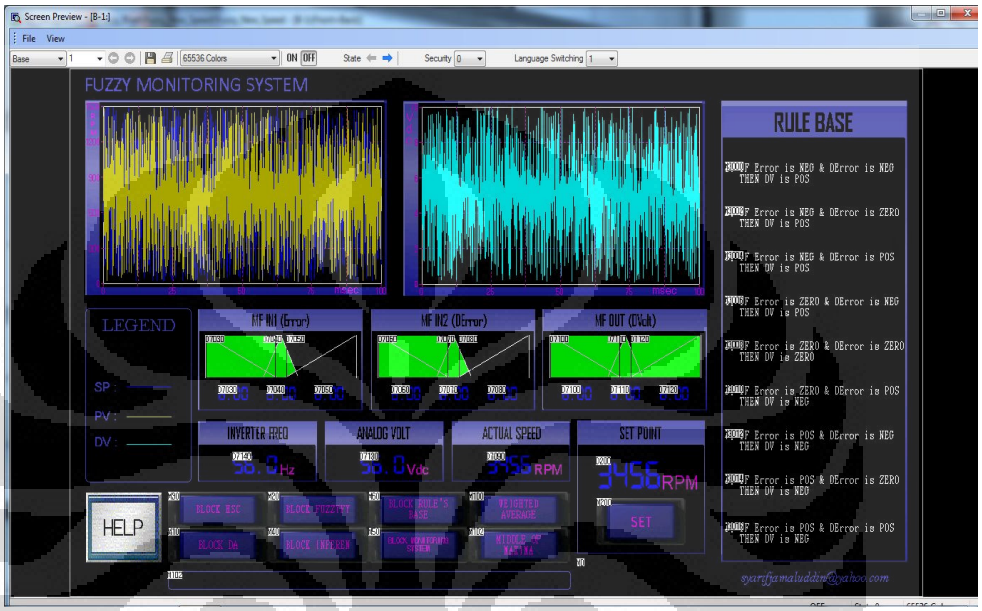
3.4 Perancangan *Screen Monitoring Fuzzy*

Perancangan *screen monitoring fuzzy* menggunakan *software* HMI (*Human Machine Interface*) lisensi Mitsubishi Electric tipe GT-Designer3 v1.10L. untuk menjalankan *screen monitoring fuzzy* secara *real time*, *software* yang

digunakan adalah *software runtime* lisensi Mitsubishi Electric tipe GT-SoftGOT1000 v1.10L. *Preview* dari GT-Designer3 v1.10L dapat dilihat pada gambar berikut.

1. **Disain *Screen Monitoring Fuzzy***

   Disain *screen monitoring fuzzy* dapat dilihat pada gambar berikut.



Gambar 3.69 Disain *screen monitoring fuzzy*

Keterangan:

- Kolom grafik sebelah kiri: 1. garis berwarna kuning adalah nilai *process value* (PV) yang ditampilkan kedalam bentuk grafik historikal, 2. garis berwarna biru tua adalah nilai *set point* (SP) yang ditampilkan kedalam bentuk grafik historikal
- Kolom grafik sebelah kanan: garis berwarna biru muda adalah nilai *analog output* yang ditampilkan kedalam bentuk grafik historikal
- Blok MF IN1 (Error): tampilan level berwarna hijau serta tampilan *7-segment* berwarna biru adalah nilai dari masing-masing subset pada fungsi keanggotaan *input* Error *(negative, zero, positive)*
- Blok MF IN2 (DError): tampilan level berwarna hijau serta tampilan *7-segment* berwarna biru adalah nilai dari masing-masing

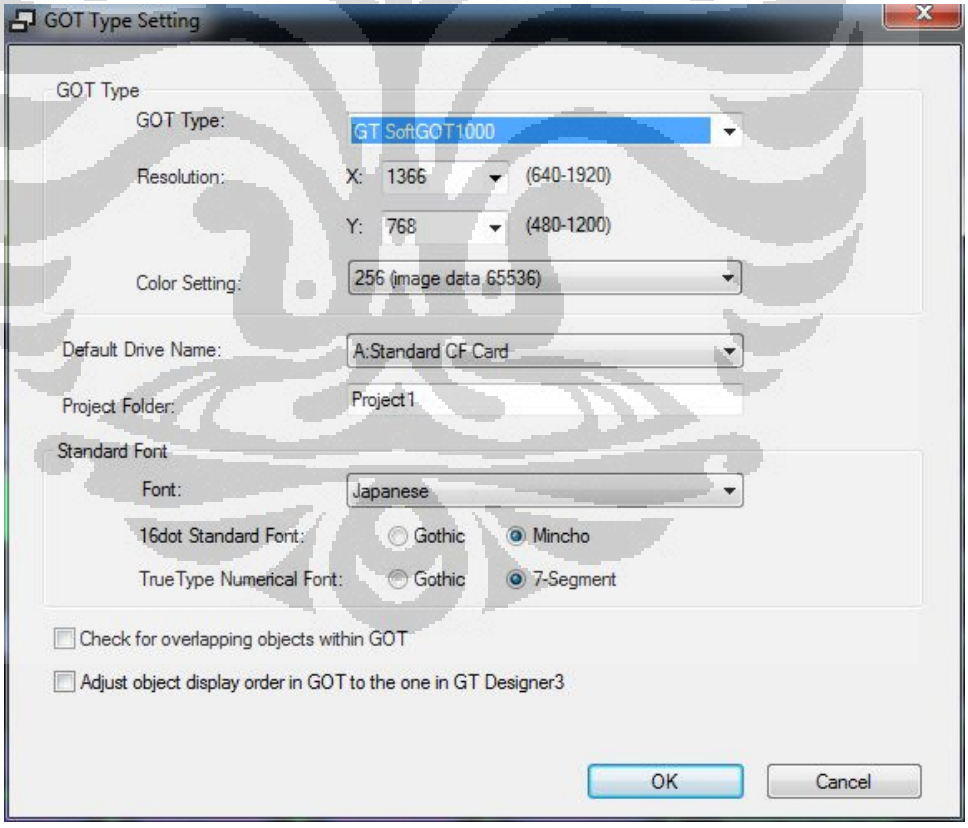
subset pada fungsi keanggotaan *input* DError *(negative, zero, positive)*

- Blok MF OUT (DVolt): tampilan level berwarna hijau serta tampilan *7-segment* berwarna biru adalah nilai dari masing-masing subset pada fungsi keanggotaan *output* DV *(negative, zero, positive)*
- Blok INVERTER FREQ: tampilan *7-segment* berwarna biru adalah nilai dari frekuensi yang dihasilkan oleh *inverter* (Hz)
- Blok ANALOG VOLT: tampilan *7-segment* berwarna biru adalah nilai dari tegangan keluaran yang dihasilkan oleh modul *digital to analog* (Vdc)
- Blok ACTUAL SPEED: tampilan *7-segment* berwarna biru adalah nilai dari *speed* aktual / *process value* (rpm)
- Blok SET POINT: tampilan *7-segment* berwarna biru adalah nilai *set point* yang dimasukkan melalui modul HMI (rpm)
- Blok RULE BASE: sembilan tampilan pernyataan matematis pada blok RULE BASE adalah aturan-aturan *fuzzy* yang digunakan di dalam sistem, berwarna putih bila tidak aktif dan berwarna biru tua bila aktif
- BLOK HSC: *Button switch* untuk mengaktifkan blok program *high speed counter*
- BLOK DA: *Button switch* untuk mengaktifkan blok program *digital to analog*
- BLOK INFEREN1: *Button switch* untuk mengaktifkan blok program inferensi
- BLOK RULE'S BASE: *Button switch* untuk mengaktifkan blok program basis aturan *fuzzy*
- BLOK MONITORING SYSTEM: *Button switch* untuk mengaktifkan blok program sistem *monitoring fuzzy*
- BLOK FUZZYFY: *Button switch* untuk mengaktifkan blok program fuzzifikasi

- BLOK WEIGHTED AVERAGE: *Button switch* untuk mengaktifkan blok program metode defuzzifikasi *weighted average*
- BLOK MIDDLE OF MAXIMA: *Button switch* untuk mengaktifkan blok program metode defuzzifikasi *middle of maxima*
- BLOK HELP: *Button switch* untuk mengaktifkan *screen* layanan bantuan pengoperasian

2. Spesifikasi *Screen Monitoring Fuzzy*

Spesifikasi *screen monitoring fuzzy* dapat dilihat pada gambar berikut.



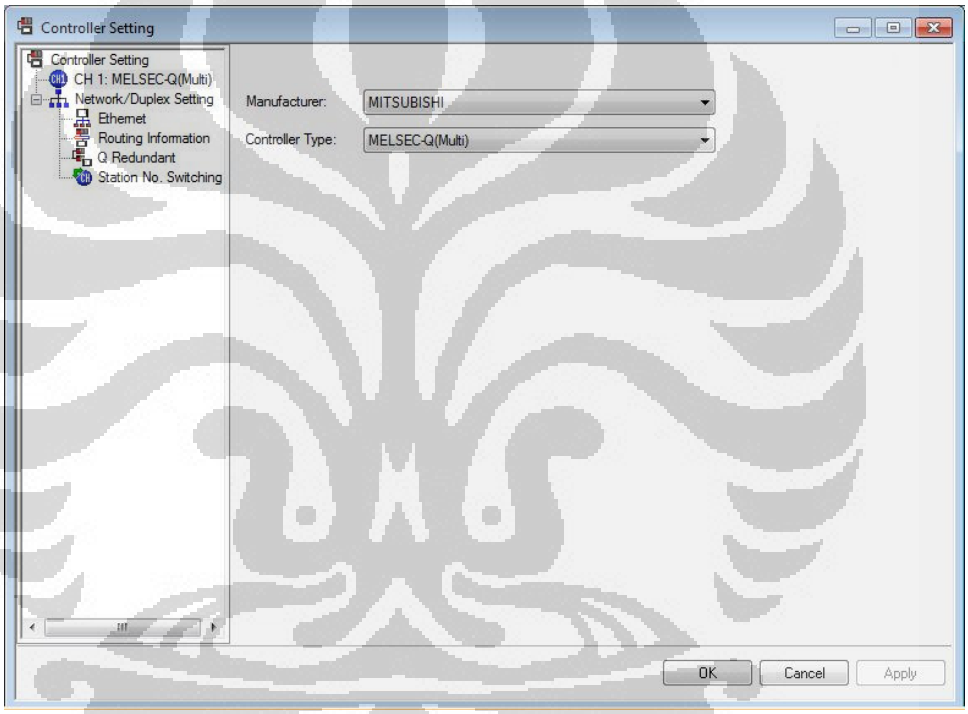
Gambar 3.70 Spesifikasi *screen monitoring fuzzy*

Keterangan:

- Tipe dari Graphic Operation Terminal (GOT): GT SoftGOT1000
- Ukuran dari resolusi screen: 1366 x 768 (laptop 14")
- Setingan warna: 65536
- Standar model huruf: Mincho
- Standar model angka: 7-Segment

3. Spesifikasi Konektivitas Pengendali

Spesifikasi konektivitas SoftGOT1000 dengan CPU pengendali dapat dilihat pada gambar berikut.



Gambar 3.71 Spesifikasi *controller*

Keterangan:

- Tipe CPU pengendali: MELSEC-Q(Multi) / Q-series CPU
- Pabrikasi: Mitsubishi

4. Spesifikasi *Basic Logging* Grafik Historikal

Spesifikasi *basic logging* untuk grafik historikal dapat dilihat pada gambar berikut.

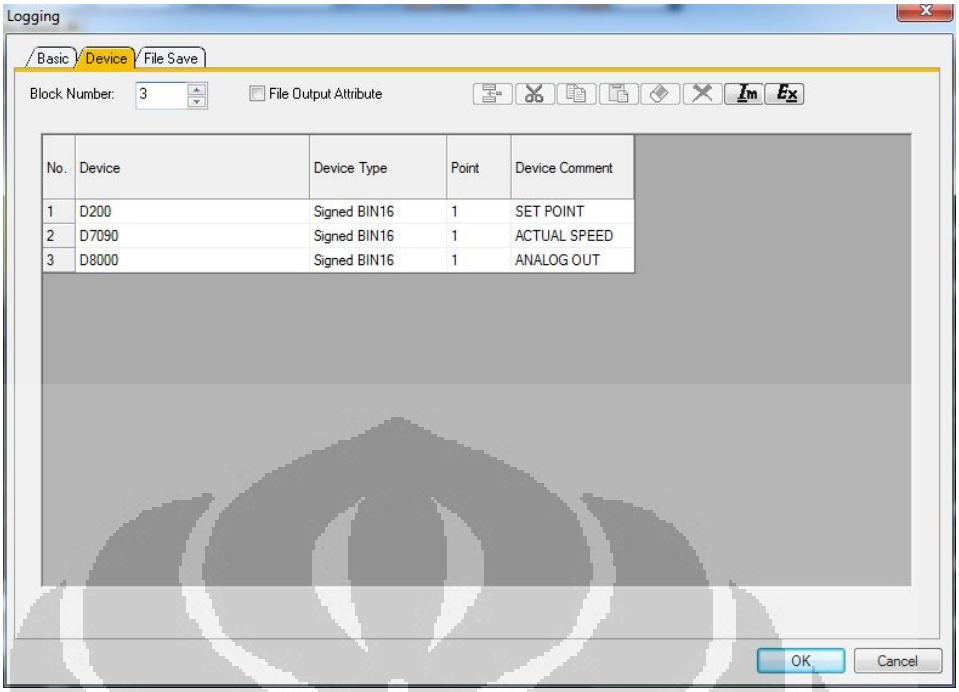
Gambar 3.72 *Basic logging* grafik historikal

Keterangan:

- Identitas logging: 1
- Nama logging: SPEED
- Jumlah file logging: 1
- Jumlah log dalam satu file logging: 1000 kali
- Logging sampling: 1 log x 100 ms

5. Spesifikasi Divais *Logging* Grafik Historikal

Spesifikasi divais *logging* untuk grafik historikal dapat dilihat pada gambar berikut.



Gambar 3.73 Divais *logging* grafik historikal

Keterangan:

- Jumlah divais logging: 3
- **D200** : SET POINT, signed decimal
- **D7090**: ACTUAL SPEED, signed decimal
- **D8000**: ANALOG OUT, signed decimal

6. Daftar Divais Pada *Screen Monitoring Fuzzy*

Daftar divais yang digunakan di dalam *screen monitoring fuzzy* dapat dilihat pada tabel berikut.

Tabel 3.3 Daftar divais *screen monitoring fuzzy*

NO.	DEVICE	MARK	TYPE	FORMAT
1	D200	SET POINT	W	SIGN DEC
2	D1000	ERROR	DW	REAL
3	D2000	DERROR	DW	REAL

4	D2500	DELTA	DW	REAL
5	D1100	MF ERROR (SUBSET NEG)	DW	REAL
6	D1200	MF ERROR (SUBSET ZERO)	DW	REAL
7	D1300	MF ERROR (SUBSET POS)	DW	REAL
8	D2100	MF DERROR (SUBSET NEG)	DW	REAL
9	D2200	MF DERROR (SUBSET ZERO)	DW	REAL
10	D2300	MF DERROR (SUBSET POS)	DW	REAL
11	D160	ACTUAL SPEED	DW	REAL
12	D3130	OUT INFEREN NO 1 (c1)	DW	REAL
13	D3008	RULE NO 5 / OUT INFEREN NO 2 (c2)	DW	REAL
14	D3110	OUT INFEREN NO 3 (c3)	DW	REAL
15	D3510	MF DVOLT (z1)	DW	REAL
16	D3520	MF DVOLT (z2)	DW	REAL
17	D3530	MF DVOLT (z3)	DW	REAL
18	D3000	RULE NO 1	DW	REAL
19	D3002	RULE NO 2	DW	REAL
20	D3004	RULE NO 3	DW	REAL
21	D3006	RULE NO 4	DW	REAL
22	D3010	RULE NO 6	DW	REAL
23	D3012	RULE NO 7	DW	REAL
24	D3014	RULE NO 8	DW	REAL
25	D3016	RULE NO 9	DW	REAL
26	D3900	MIDDLE OF MAXIMA	DW	REAL
27	D4200	WEIGHTED AVERAGE	DW	REAL
28	D8000	ANALOG OUTPUT	W	SIGN DEC
29	M3000	SIGNAL FLAG RULE NO 1	B	(-)
30	M3002	SIGNAL FLAG RULE NO 2	B	(-)
31	M3004	SIGNAL FLAG RULE NO 3	B	(-)

32	M3006	SIGNAL FLAG RULE NO 4	B	(-)
33	M3008	SIGNAL FLAG RULE NO 5	B	(-)
34	M3010	SIGNAL FLAG RULE NO 6	B	(-)
35	M3012	SIGNAL FLAG RULE NO 7	B	(-)
36	M3014	SIGNAL FLAG RULE NO 8	B	(-)
37	M3016	SIGNAL FLAG RULE NO 9	B	(-)

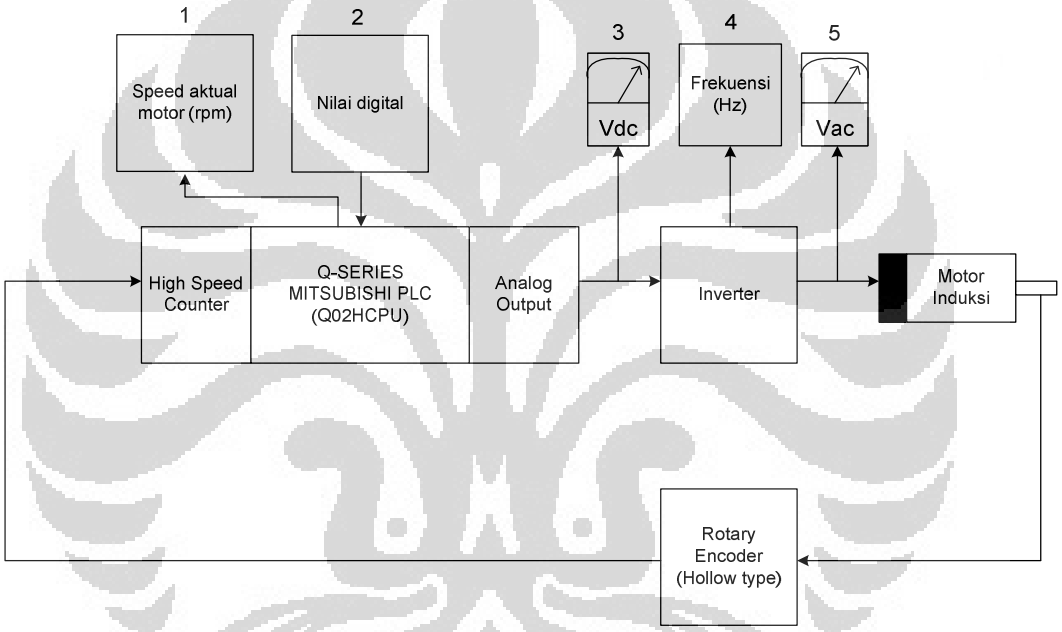
Keterangan:

- W : 16 bit (word)
- DW : 32 bit (double word)
- B : 1 bit
- SIGN DEC : -32767 to 32768
- REAL : $\pm$ 2^{-126} to 2^{128}

BAB 4
PERCOBAAN DAN ANALISIS DATA

4.1 Pengukuran

Sebelum melakukan percobaan secara *realtime* terlebih dahulu melakukan pengukuran untuk memperoleh data mengenai karakteristik dari masing-masing *hardware* yang terintegrasi dengan sistem. Lokasi titik pengukuran dapat dilihat pada gambar berikut.



Gambar 4.1 Lokasi titik pengukuran pada sistem

Keterangan:

1. Pengambilan data *speed* aktual (rpm) melalui *software* GX-Developer v8.95Z.
2. *Input* data nilai digital melalui *software* GX-Developer v8.95Z.
3. Pengambilan data tegangan keluaran *analog* (Vdc) dengan menggunakan multimeter digital tipe HIOKI 3256-50.

4. Pengambilan data frekuensi keluaran *inverter* (Hz) melalui *display unit* tipe DU07 yang terintegrasi dengan *inverter*.
5. Pengambilan data tegangan keluaran *inverter* (Vac) dengan menggunakan multimeter digital tipe HIOKI 3256-50.

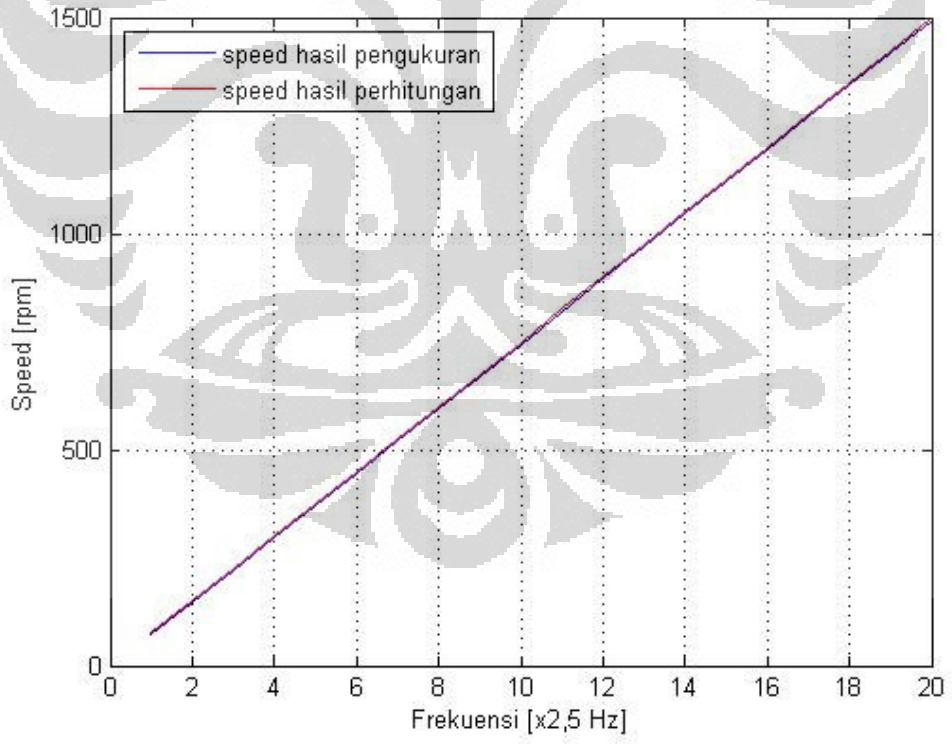
Pengukuran pada sistem *open loop* dilakukan sebanyak 20 data secara linier dengan cara memberikan *input* nilai digital dimulai dari 200 hingga 4000. Data hasil pengukuran pada sistem diatas dapat dilihat pada tabel berikut.

Tabel 4.1 Data hasil pengukuran sistem *fuzzy*

NO.	ANALOG (Vdc)	FREKUENSI INV/ Ω (Hz)	SPEED PENGUKURAN (rpm)	SPEED PERHITUNGAN (rpm)
1	0,5	2,5	72	75
2	1	5	146	150
3	1,5	7,5	222	225
4	2	10	296	300
5	2,5	12,5	372	375
6	3	15	446	450
7	3,5	17,5	521	525
8	4	20	596	600
9	4,5	22,5	670	675
10	5	25	745	750
11	5,5	27,5	820	825
12	6	30	895	900
13	6,5	32,5	970	975
14	7	35	1044	1050
15	7,5	37,5	1119	1125
16	8	40	1194	1200
17	8,5	42,5	1269	1275
18	9	45	1343	1350

19	9,5	47,5	1418	1425
20	10	50	1493	1500

Pada Tabel 4.1, kolom ANALOG (Vdc) merupakan data hasil pengukuran tegangan keluaran modul *digital to analog* dimulai dari 0,5 Vdc dengan kenaikan 0,5 Vdc sebanyak 20 data, kolom FREKUENSI INV (Hz) merupakan data hasil pengukuran frekuensi keluaran modul *inverter* dimulai dari 2,5 Hz dengan kenaikan 2,5 Hz sebanyak 20 data, kolom SPEED PENGUKURAN (rpm) merupakan data hasil pembacaan *speed* aktual hasil pengukuran dimulai dari 72 hingga 1493 rpm sebanyak 20 data, dan kolom SPEED PERHITUNGAN (rpm) merupakan nilai *speed* hasil perhitungan pada percobaan *open loop* yang dimulai dari 75 hingga 1500 rpm sebanyak 20 data. Plot data *speed* hasil pengukuran dan perhitungan kedalam bentuk grafik dapat dilihat pada gambar berikut.



Gambar 4.2 Grafik perbandingan *speed* hasil pengukuran dan *speed* hasil perhitungan pada percobaan *open loop*

Berdasarkan Gambar 4.2, garis berwarna biru pada grafik adalah nilai *speed* hasil pengukuran sedangkan garis berwarna merah adalah nilai *speed* hasil perhitungan. Nilai *speed* hasil perhitungan diperoleh dengan menggunakan persamaan dibawah ini.

$$n_s = \frac{120 x\, fs}{4} \ (rpm)$$

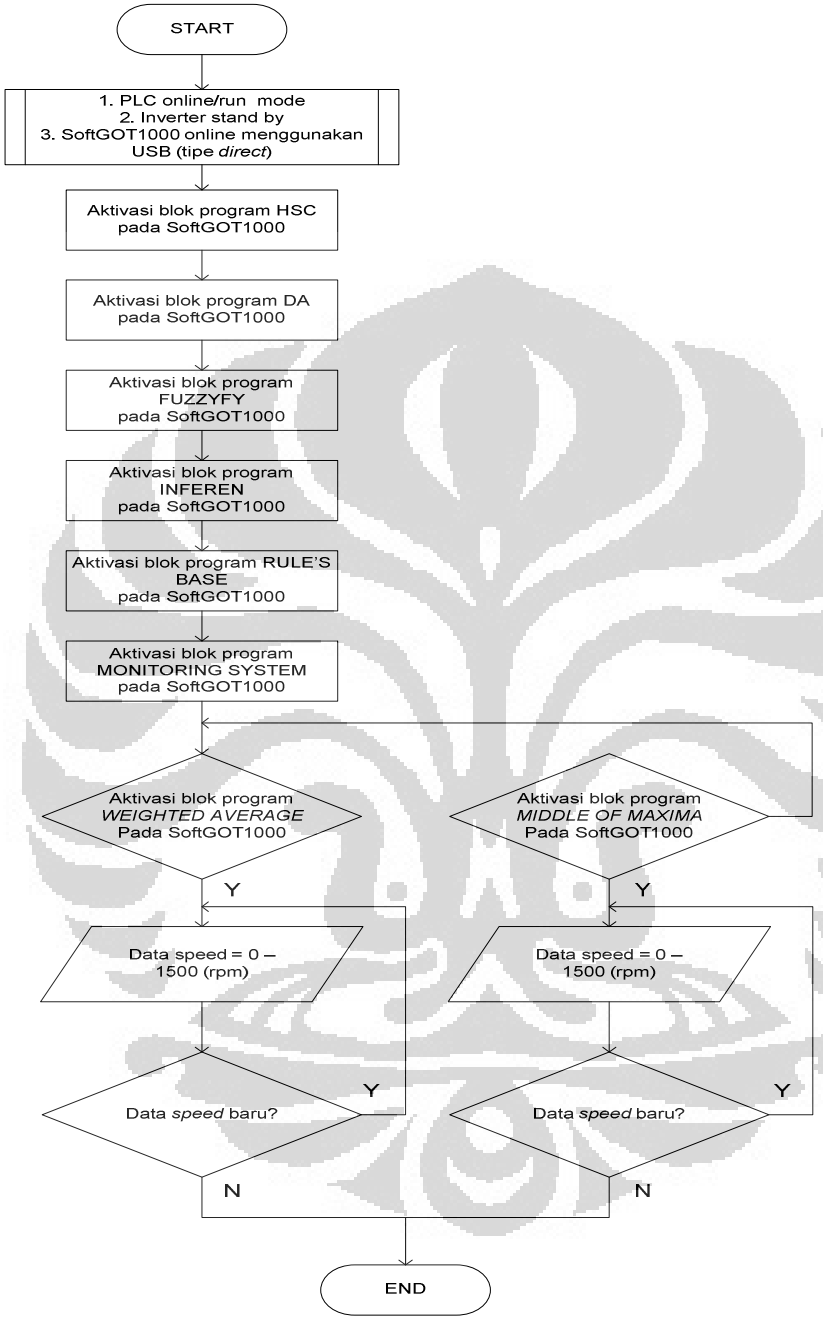
Keterangan:

n_s : *Speed* hasil perhitungan (rpm)

fs : Frekuensi yang dihasilkan oleh inverter (Hz)

4.2 Prosedur Percobaan

4.2.1 *Flowchart* Percobaan



Gambar 4.3 *Flowchart* percobaan

4.2.2 Langkah Percobaan

Sebelum melakukan percobaan, pastikan *wiring* modul *controller*, modul *inverter* sudah terhubung dengan baik dan benar. Penjelasan dari langkah-langkah percobaan berdasarkan *flowchart* pada Gambar 4.3 adalah sebagai berikut.

1. **Persiapan**: posisi modul *controller* dalam keadaan *online/run mode*, modul *inverter* dalam keadaan *standby*, dan posisi SoftGOT1000 dalam keadaan *online* dengan modul *controller* menggunakan kabel USB.

2. **Aktivasi blok program**: aktivasi blok DA, blok HSC, blok FUZZYFY, blok INFEREN, blok RULE'S BASE, dan blok MONITORING SYSTEM.

3. **Aktivasi metode defuzzyfikasi**: pilih salah satu metode defuzzyfikasi yang akan digunakan dengan mengaktivasi blok program WEIGHTED AVERAGE atau MIDDLE OF MAXIMA.

4. **Masukan nilai data *speed***: nilai data *speed* dapat diinput pada kolom SET POINT, besarnya nilai yang dapat diinput dari 0 sampai 1500 (rpm).

4.3 Percobaan Menggunakan Metode Defuzzifikasi *Weighted Average*

Pada percobaan *realtime* dengan menggunakan metode defuzzifikasi *weighted average* dibagi ke dalam dua tahap, yaitu siklus Error *positive* (300, 600, 900, 1200) rpm dan siklus Error negative (900, 600, 300, 0) rpm.

4.3.1 Siklus Error *Positive* (300, 600, 900, 1200) RPM *(Weighted Average)*

Siklus Error *positive* diperoleh jika nilai *set point* (SP) lebih besar daripada nilai *speed* aktual (PV), sedangkan nilai Error sendiri diperoleh dari hasil pengurangan antara *set point* (SP) dengan *speed* aktual (PV).

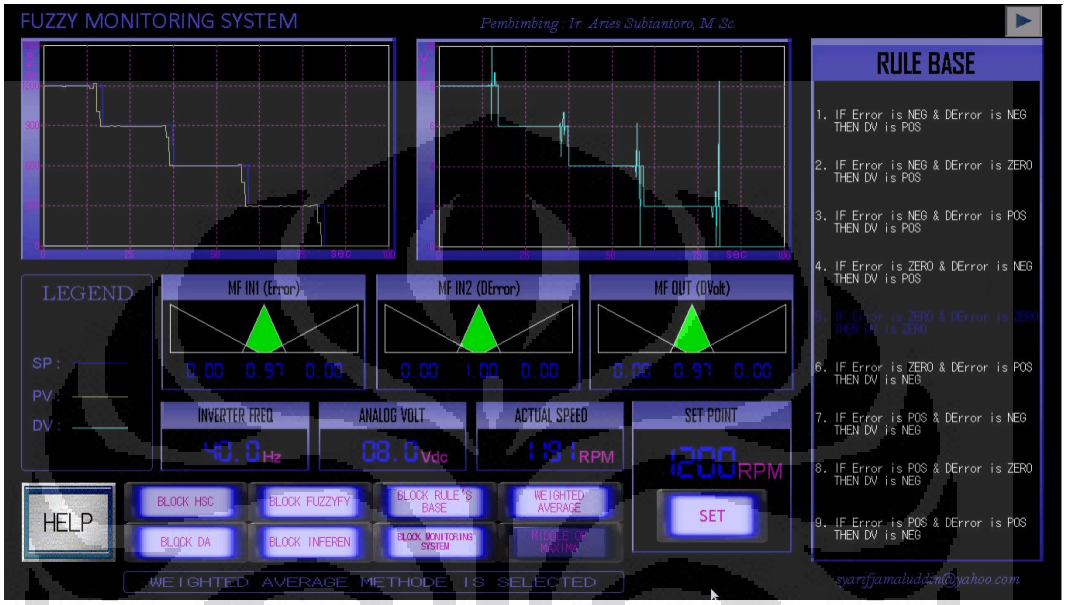
Error = SP – PV

Error *positive* = Error dimana SP > PV

Keterangan:

- SP = Nilai yang diinginkan (*Set Point*)
- PV = Nilai *speed* aktual (*Process Value*)

Data hasil percobaan berupa tampilan *screen* modul HMI pada siklus Error *positive* dengan menggunakan metode defuzzifikasi *weighted average* dapat dilihat pada gambar berikut.



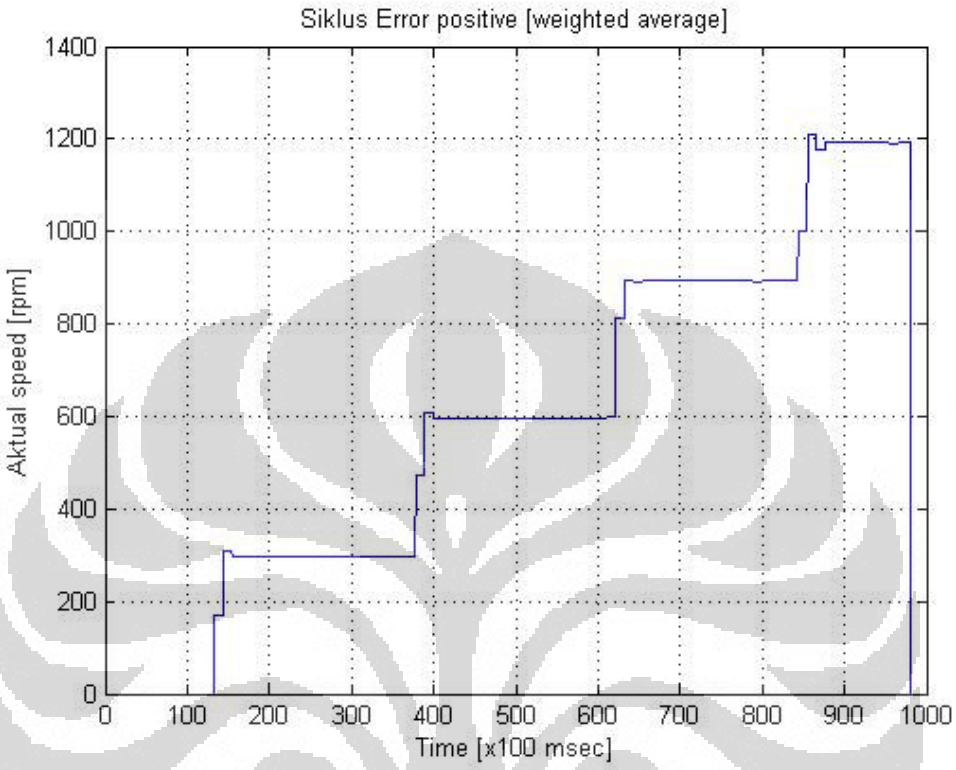
Gambar 4.4 Tampilan modul HMI siklus Error *positive* metode defuzzifikasi *weighted average*

Keterangan:

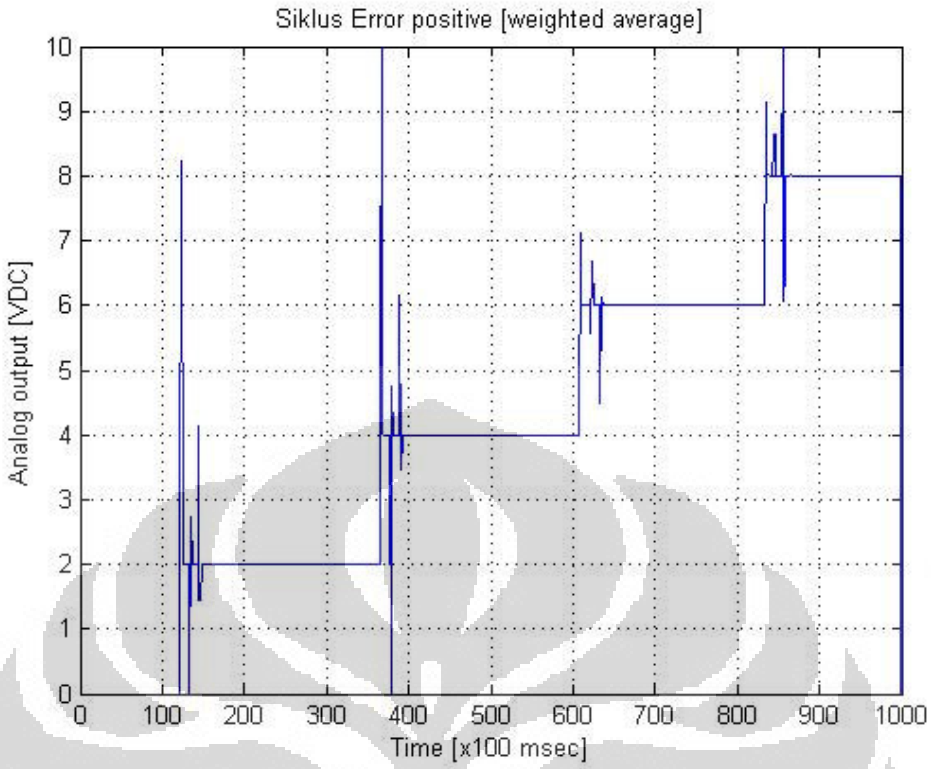
- Kolom grafik sebelah kiri: 1. garis berwarna kuning adalah nilai *process value* (PV) yang ditampilkan kedalam bentuk grafik historikal, 2. garis berwarna biru tua adalah nilai *set point* (SP) yang ditampilkan kedalam bentuk grafik historikal
- Kolom grafik sebelah kanan: garis berwarna biru muda adalah nilai *analog output* yang ditampilkan kedalam bentuk grafik historikal
- Blok MF IN1 (Error): tampilan level berwarna hijau serta tampilan *7-segment* berwarna biru adalah nilai dari masing-masing subset pada fungsi keanggotaan *input* Error *(negative, zero, positive)*
- Blok MF IN2 (DError): tampilan level berwarna hijau serta tampilan *7-segment* berwarna biru adalah nilai dari masing-masing subset pada fungsi keanggotaan *input* DError *(negative, zero, positive)*

- Blok MF OUT (DVolt): tampilan level berwarna hijau serta tampilan *7-segment* berwarna biru adalah nilai dari masing-masing subset pada fungsi keanggotaan *output* DV *(negative, zero, positive)*
- Blok INVERTER FREQ: tampilan *7-segment* berwarna biru adalah nilai dari frekuensi yang dihasilkan oleh *inverter* (Hz)
- Blok ANALOG VOLT: tampilan *7-segment* berwarna biru adalah nilai dari tegangan keluaran yang dihasilkan oleh modul *digital to analog* (Vdc)
- Blok ACTUAL SPEED: tampilan *7-segment* berwarna biru adalah nilai dari *speed* aktual / *process value* (rpm)
- Blok SET POINT: tampilan *7-segment* berwarna biru adalah nilai *set point* yang dimasukkan melalui modul HMI (rpm)
- Blok RULE BASE: sembilan tampilan pernyataan matematis pada blok RULE BASE adalah aturan-aturan *fuzzy* yang digunakan di dalam sistem, berwarna putih bila tidak aktif dan berwarna biru tua bila aktif
- BLOK HSC: *Button switch* untuk mengaktifkan blok program *high speed counter*
- BLOK DA: *Button switch* untuk mengaktifkan blok program *digital to analog*
- BLOK INFEREN1: *Button switch* untuk mengaktifkan blok program inferensi
- BLOK RULE'S BASE: *Button switch* untuk mengaktifkan blok program basis aturan *fuzzy*
- BLOK MONITORING SYSTEM: *Button switch* untuk mengaktifkan blok program sistem *monitoring fuzzy*
- BLOK FUZZYFY: *Button switch* untuk mengaktifkan blok program fuzzifikasi
- BLOK WEIGHTED AVERAGE: *Button switch* untuk mengaktifkan blok program metode defuzzifikasi *weighted average*
- BLOK MIDDLE OF MAXIMA: *Button switch* untuk mengaktifkan blok program metode defuzzifikasi *middle of maxima*

Grafik transien perubahan data *speed* pada siklus Error *positive* dengan menggunakan metode defuzzifikasi *weighted average* dapat dilihat pada gambar berikut.



(a)



(b)

Gambar 4.5 Grafik respon transien perubahan data *speed* siklus Error *positive* metode defuzzifikasi *weighted average*

(a) Gambar *Actual speed*

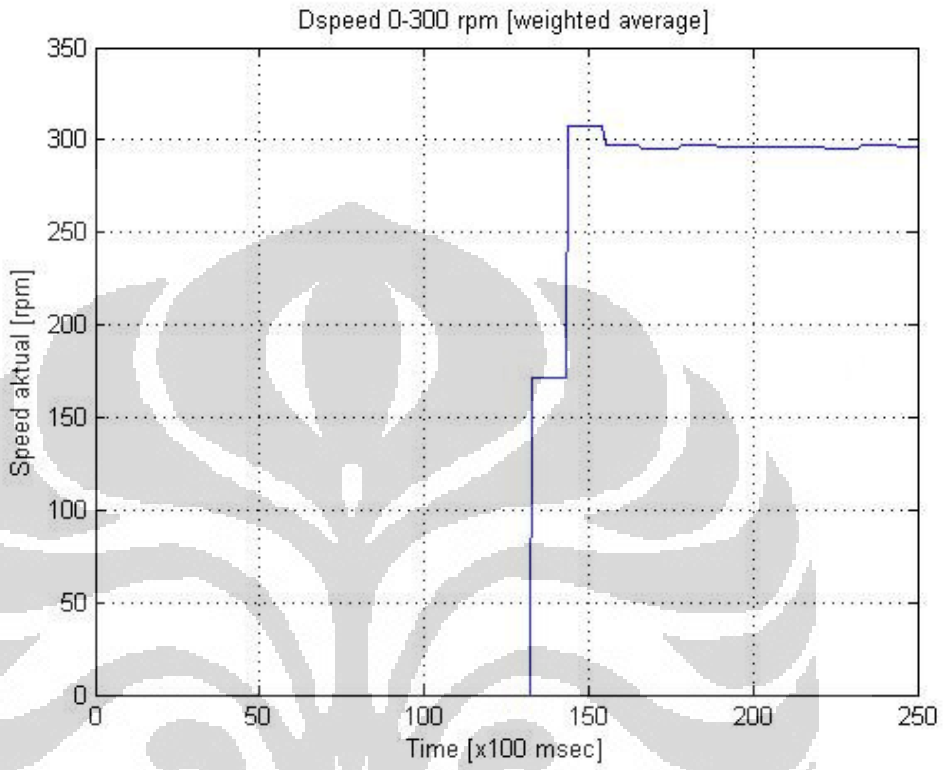
(b) Gambar *Analog output*

Gambar 4.5(a) menggambarkan grafik transien *speed* aktual sedangkan Gambar 4.5(b) menggambarkan grafik transien tegangan keluaran dari modul *digital to analog* dari siklus Error *positive* pada percobaan menggunakan metode defuzzifikasi *weighted average*. Siklus Error *positive* memiliki empat data perubahan *speed* sebagai berikut.

- Perubahan *speed* 0 ke 300 rpm
- Perubahan *speed* 300 ke 600 rpm
- Perubahan *speed* 600 ke 900 rpm
- Perubahan *speed* 900 ke 1200 rpm

1. Perubahan *Speed* 0 Ke 300 RPM (*Weighted Average*)

Data hasil percobaan perubahan *speed* 0 ke 300 rpm dapat dilihat pada gambar berikut.



Gambar 4.6 Grafik respon transien perubahan *speed* 0 ke 300 rpm (*weighted average*)

Gambar 4.6 menggambarkan grafik transien perubahan *speed* 0 ke 300 rpm pada percobaan menggunakan metode defuzzifikasi *weighted average*. Berdasarkan grafik respon transien perubahan *speed* 0 ke 300 rpm metode defuzzifikasi *weighted average* diperoleh data sebagai berikut.

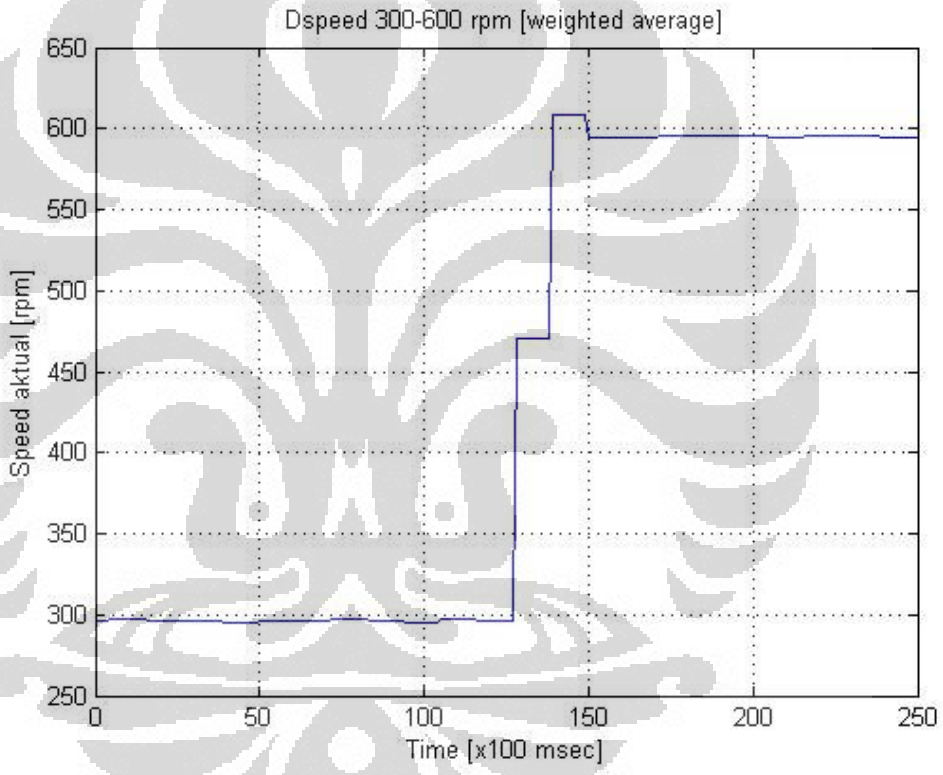
Tabel 4.2 Parameter transien perubahan *speed* 0 ke 300 rpm (*weighted average*)

Settling time (sec)	Overshoot (%)	Undershoot (%)	Error steady state (rpm / %)
2,3	2,67	0	4 / 1,33

Pada Tabel 4.2, parameter unjuk kerja perubahan speed 0 ke 300 rpm pada percobaan menggunakan metode defuzzifikasi *weighted average, settling time* yang diperoleh adalah 2,3 sec, *overshoot* yang diperoleh adalah 2,67 %, dan *error steady state* yang diperoleh adalah 4 rpm / 1,33 %.

2. Perubahan *Speed* 300 Ke 600 RPM (*Weighted Average*)

Data hasil percobaan perubahan *speed* 300 ke 600 rpm dapat dilihat pada gambar berikut.



Gambar 4.7 Grafik respon transien perubahan *speed* 300 ke 600 rpm (*weighted average*)

Gambar 4.7 menggambarkan grafik transien perubahan *speed* 300 ke 600 rpm pada percobaan menggunakan metode defuzzifikasi *weighted average*. Berdasarkan grafik respon transien perubahan *speed* 300 ke 600 rpm metode defuzzifikasi *weighted average* diperoleh data sebagai berikut.

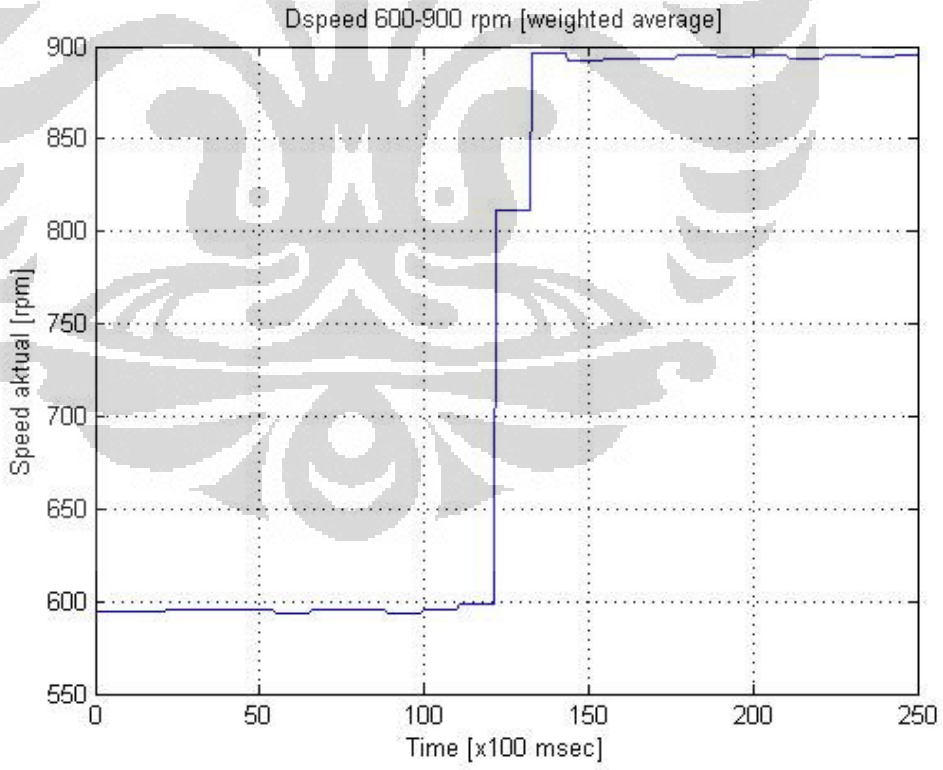
Tabel 4.3 Parameter transien perubahan *speed* 300 ke 600 rpm *(weighted average)*

Settling time (sec)	Overshoot (%)	Undershoot (%)	Error steady state (rpm / %)
2,3	1,33	0	5 / 0,83

Pada Tabel 4.3, parameter unjuk kerja perubahan *speed* 300 ke 600 rpm pada percobaan menggunakan metode defuzzifikasi *weighted average, settling time* yang diperoleh adalah 2,3 sec, *overshoot* yang diperoleh adalah 1,33 %, dan *error steady state* yang diperoleh adalah 5 rpm / 0,83 %.

3. Perubahan *Speed* 600 Ke 900 RPM (*Weighted Average*)

Data hasil percobaan perubahan *speed* 600 ke 900 rpm dapat dilihat pada gambar berikut.



Gambar 4.8 Grafik transien perubahan *speed* 600 ke 900 rpm (*weighted average*)

Gambar 4.8 menggambarkan grafik transien perubahan *speed* 600 ke 900 rpm pada percobaan menggunakan metode defuzzifikasi *weighted average*. Berdasarkan grafik respon transien perubahan *speed* 600 ke 900 rpm metode defuzzifikasi *weighted average* diperoleh data sebagai berikut.

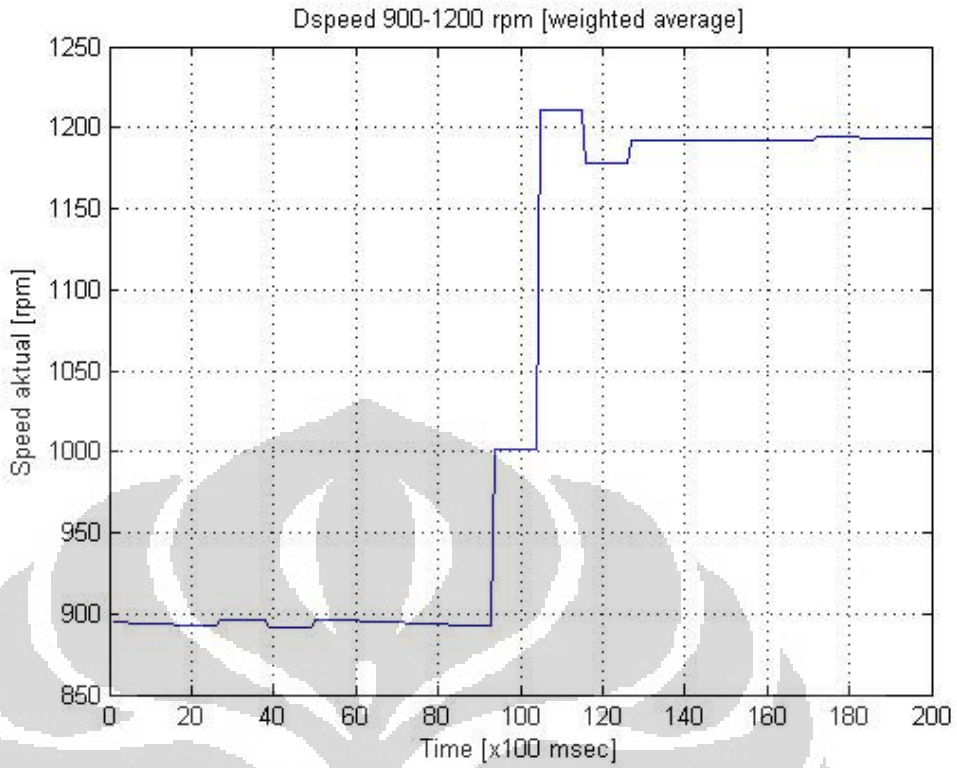
Tabel 4.4 Parameter transien perubahan *speed* 600 ke 900 rpm (*weighted average*)

Settling time (sec)	Overshoot (%)	Undershoot (%)	Error steady state (rpm / %)
2,3	0	0	6 / 0,67

Pada Tabel 4.4, parameter unjuk kerja perubahan *speed* 600 ke 900 rpm pada percobaan menggunakan metode defuzzifikasi *weighted average, settling time* yang diperoleh adalah 2,3 sec dan *error steady state* yang diperoleh adalah 6 rpm / 0,67 %.

4. Perubahan *Speed* 900 Ke 1200 RPM (*Weighted Average*)

Data hasil percobaan perubahan *speed* 900 ke 1200 rpm dapat dilihat pada gambar berikut.



Gambar 4.9 Grafik respon transien perubahan *speed* 900 ke 1200 rpm (*weighted average*)

Gambar 4.9 menggambarkan grafik transien perubahan *speed* 900 ke 1200 rpm pada percobaan menggunakan metode defuzzifikasi *weighted average*. Berdasarkan grafik respon transien perubahan *speed* 900 ke 1200 rpm metode defuzzifikasi *weighted average* diperoleh data sebagai berikut.

Tabel 4.5 Parameter transien perubahan *speed* 900 ke 1200 rpm (*weighted average*)

Settling time (sec)	Overshoot (%)	Undershoot (%)	Error steady state (rpm / %)
3,4	0,92	0	8 / 0,67

Pada Tabel 4.5, parameter unjuk kerja perubahan *speed* 900 ke 1200 rpm pada percobaan menggunakan metode defuzzifikasi *weighted average, settling time* yang diperoleh adalah 3,4 sec, *overshoot* adalah 0,92 %, dan *error steady state* yang diperoleh adalah 8 rpm / 0,67 %.

4.3.2 Siklus Error *Negative* (900, 600, 300, 0) RPM *(Weighted Average)*

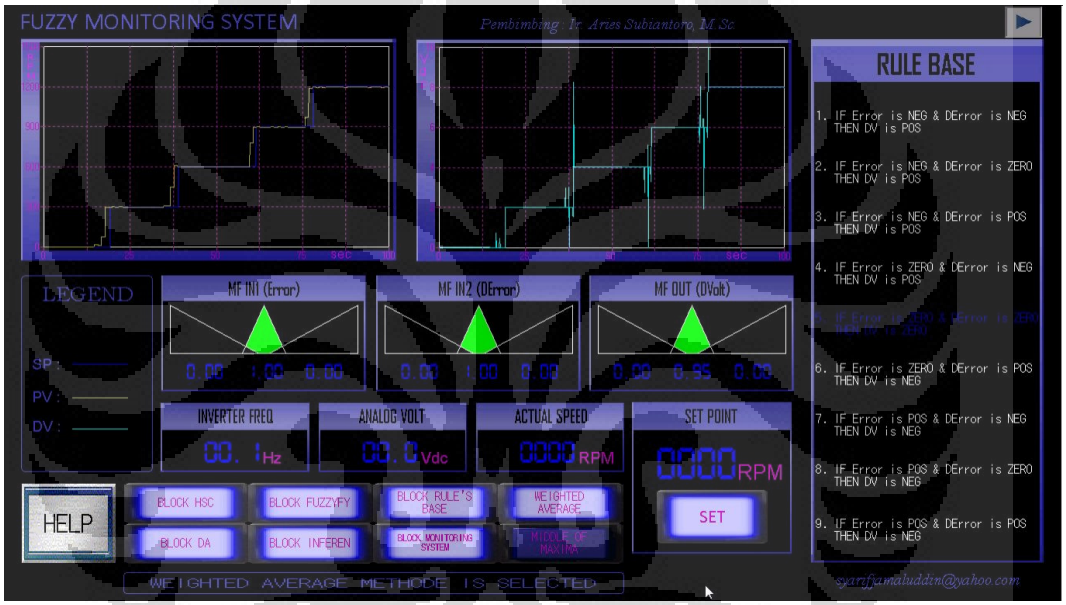
Siklus Error *negative* diperoleh jika nilai *set point* (SP) lebih kecil daripada nilai *speed* aktual (PV), sedangkan nilai Error sendiri diperoleh dari hasil pengurangan antara *set point* (SP) dengan *speed* aktual (PV).

Error = SP – PV

Error *negative* = Error dimana SP < PV

Keterangan:

- SP = Nilai yang diinginkan (*Set Point*)
- PV = Nilai *speed* aktual (*Process Value*)



Gambar 4.10 Tampilan modul HMI siklus Error *negative* metode defuzzifikasi *weighted average*

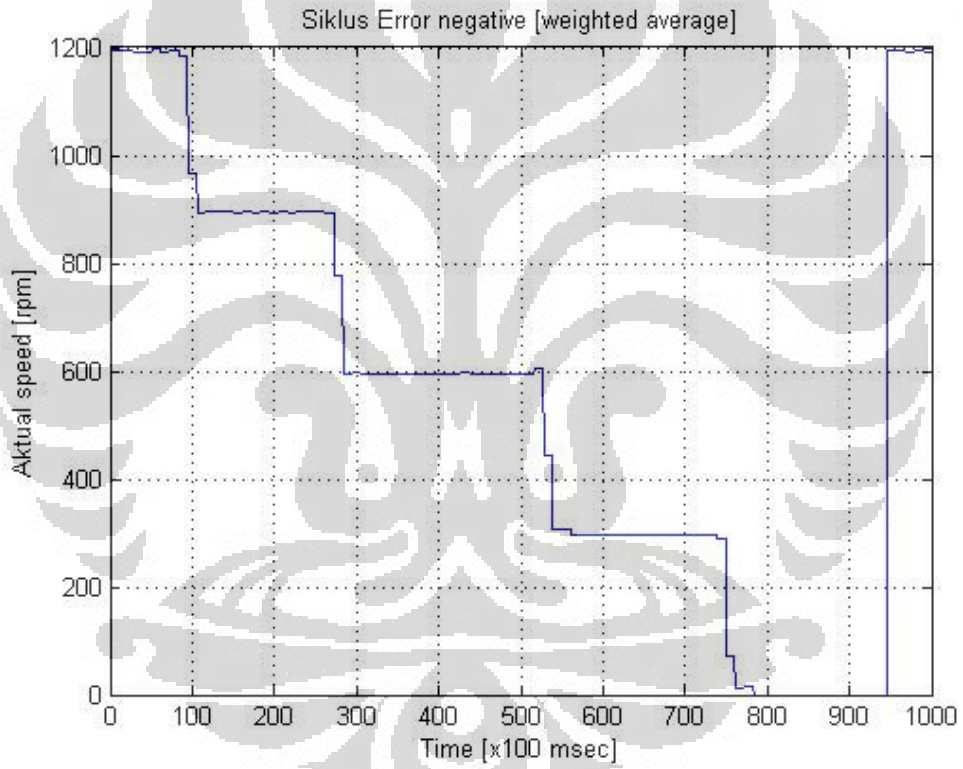
Keterangan:

- Kolom grafik sebelah kiri: 1. garis berwarna kuning adalah nilai *process value* (PV) yang ditampilkan kedalam bentuk grafik historikal, 2. garis berwarna biru tua adalah nilai *set point* (SP) yang ditampilkan kedalam bentuk grafik historikal
- Kolom grafik sebelah kanan: garis berwarna biru muda adalah nilai *analog output* yang ditampilkan kedalam bentuk grafik historikal

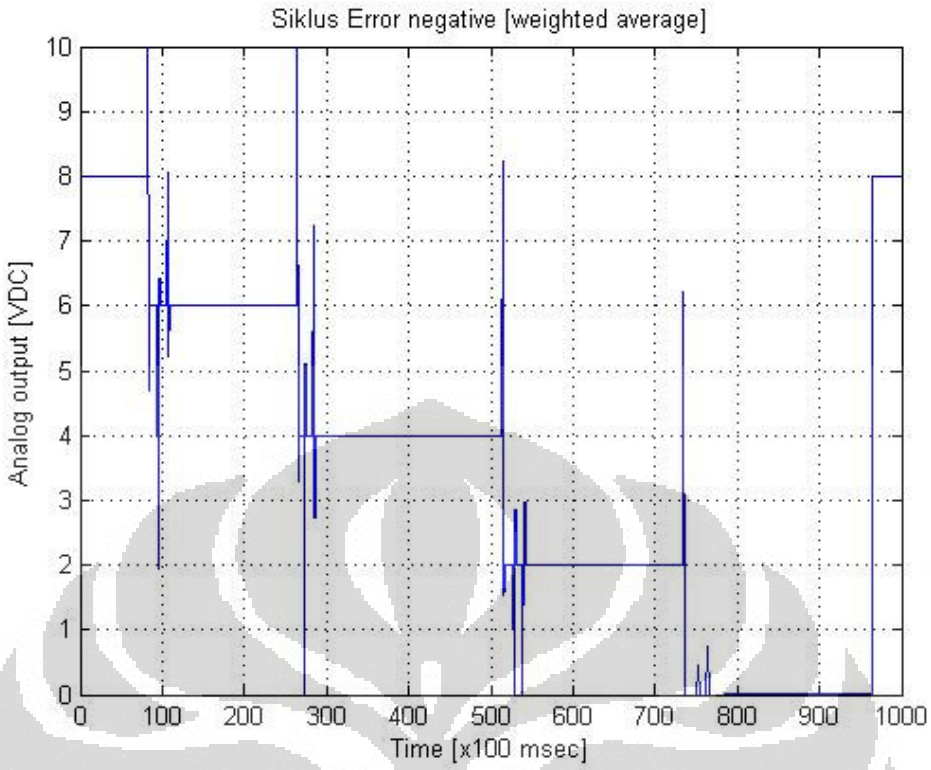
- Blok MF IN1 (Error): tampilan level berwarna hijau serta tampilan *7-segment* berwarna biru adalah nilai dari masing-masing subset pada fungsi keanggotaan *input* Error *(negative, zero, positive)*
- Blok MF IN2 (DError): tampilan level berwarna hijau serta tampilan *7-segment* berwarna biru adalah nilai dari masing-masing subset pada fungsi keanggotaan *input* DError *(negative, zero, positive)*
- Blok MF OUT (DVolt): tampilan level berwarna hijau serta tampilan *7-segment* berwarna biru adalah nilai dari masing-masing subset pada fungsi keanggotaan *output* DV *(negative, zero, positive)*
- Blok INVERTER FREQ: tampilan *7-segment* berwarna biru adalah nilai dari frekuensi yang dihasilkan oleh *inverter* (Hz)
- Blok ANALOG VOLT: tampilan *7-segment* berwarna biru adalah nilai dari tegangan keluaran yang dihasilkan oleh modul *digital to analog* (Vdc)
- Blok ACTUAL SPEED: tampilan *7-segment* berwarna biru adalah nilai dari *speed* aktual / *process value* (rpm)
- Blok SET POINT: tampilan *7-segment* berwarna biru adalah nilai *set point* yang dimasukkan melalui modul HMI (rpm)
- Blok RULE BASE: sembilan tampilan pernyataan matematis pada blok RULE BASE adalah aturan-aturan *fuzzy* yang digunakan di dalam sistem, berwarna putih bila tidak aktif dan berwarna biru tua bila aktif
- BLOK HSC: *Button switch* untuk mengaktifkan blok program *high speed counter*
- BLOK DA: *Button switch* untuk mengaktifkan blok program *digital to analog*
- BLOK INFEREN1: *Button switch* untuk mengaktifkan blok program inferensi
- BLOK RULE'S BASE: *Button switch* untuk mengaktifkan blok program basis aturan *fuzzy*
- BLOK MONITORING SYSTEM: *Button switch* untuk mengaktifkan blok program sistem *monitoring fuzzy*

- BLOK FUZZYFY: *Button switch* untuk mengaktifkan blok program fuzzifikasi
- BLOK WEIGHTED AVERAGE: *Button switch* untuk mengaktifkan blok program metode defuzzifikasi *weighted average*
- BLOK MIDDLE OF MAXIMA: *Button switch* untuk mengaktifkan blok program metode defuzzifikasi *middle of maxima*

Grafik transien perubahan data *speed* pada siklus Error *negative* dengan menggunakan metode defuzzifikasi *weighted average* dapat dilihat pada gambar berikut.



(a)



(b)

Gambar 4.11 Grafik transien perubahan data speed siklus Error *negative* metode defuzzyfikasi *weighted average*

(c) Gambar *Actual speed*

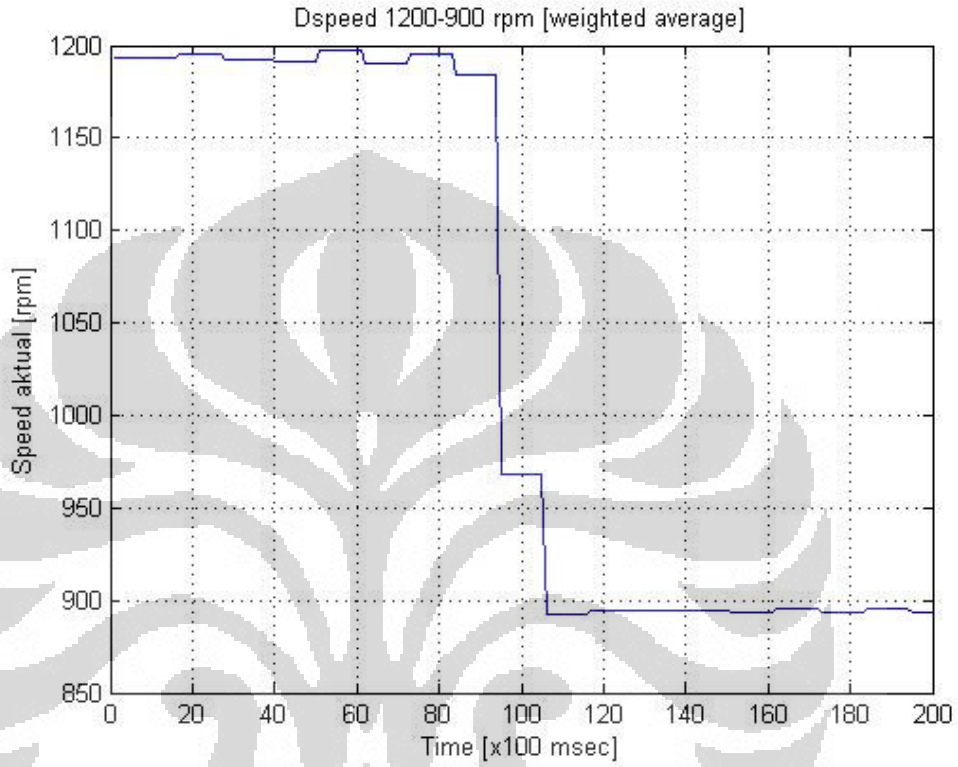
(d) Gambar *Analog output*

Gambar 4.11(a) menggambarkan grafik transien *speed* aktual sedangkan Gambar 4.11(b) menggambarkan grafik transien tegangan keluaran dari modul *digital to analog* dari siklus Error *negative* pada percobaan menggunakan metode defuzzifikasi *weighted average*. Siklus Error *negative* memiliki empat data perubahan speed sebagai berikut.

- Perubahan *speed* 1200 ke 900 rpm
- Perubahan *speed* 900 ke 600 rpm
- Perubahan *speed* 600 ke 300 rpm
- Perubahan *speed* 300 ke 0 rpm

1. Perubahan *Speed* 1200 Ke 900 RPM (*Weighted Average*)

Data hasil percobaan perubahan *speed* dari 1200 ke 900 rpm dapat dilihat pada gambar berikut.



Gambar 4.12 Grafik respon transien perubahan speed 1200 ke 900 rpm (*weighted average*)

Gambar 4.12 menggambarkan grafik transien perubahan *speed* 1200 ke 900 rpm pada percobaan menggunakan metode defuzzifikasi *weighted average*. Berdasarkan grafik respon transien perubahan *speed* 1200 ke 900 rpm metode defuzzifikasi *weighted average* diperoleh data sebagai berikut.

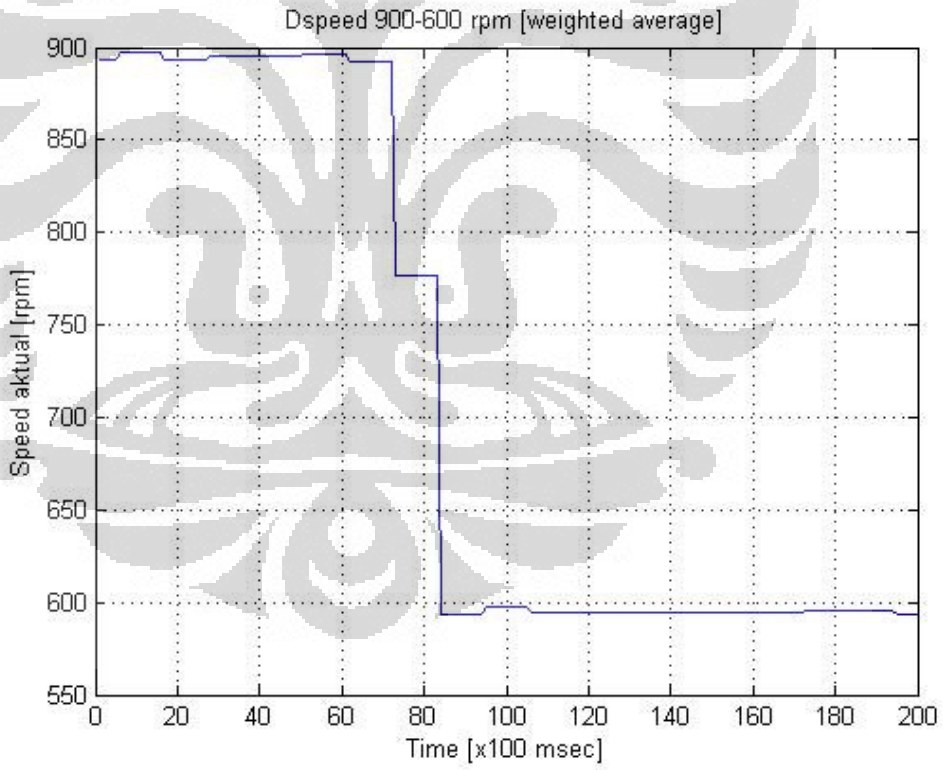
Tabel 4.6 Parameter transien perubahan *speed* 1200 ke 900 rpm (*weighted average*)

Settling time (sec)	Overshoot (%)	Undershoot (%)	Error steady state (rpm / %)
2,3	0,89	0	6 / 0,67

Pada Tabel 4.6, parameter unjuk kerja perubahan *speed* 1200 ke 900 rpm pada percobaan menggunakan metode defuzzifikasi *weighted average, settling time* yang diperoleh adalah 2,3 sec, *overshoot* adalah 0,89 %, dan *error steady state* yang diperoleh adalah 6 rpm / 0,67 %.

2. Perubahan *Speed* 900 Ke 600 RPM (*Weighted Average*)

Data hasil percobaan perubahan *speed* 900 ke 600 rpm dapat dilihat pada gambar berikut.



Gambar 4.13 Grafik respon transien perubahan *speed* 900 ke 600 rpm (*weighted average*)

Gambar 4.13 menggambarkan grafik transien perubahan *speed* 900 ke 600 rpm pada percobaan menggunakan metode defuzzifikasi *weighted average*. Berdasarkan grafik respon transien perubahan *speed* 900 ke 600 rpm metode defuzzifikasi *weighted average* diperoleh data sebagai berikut.

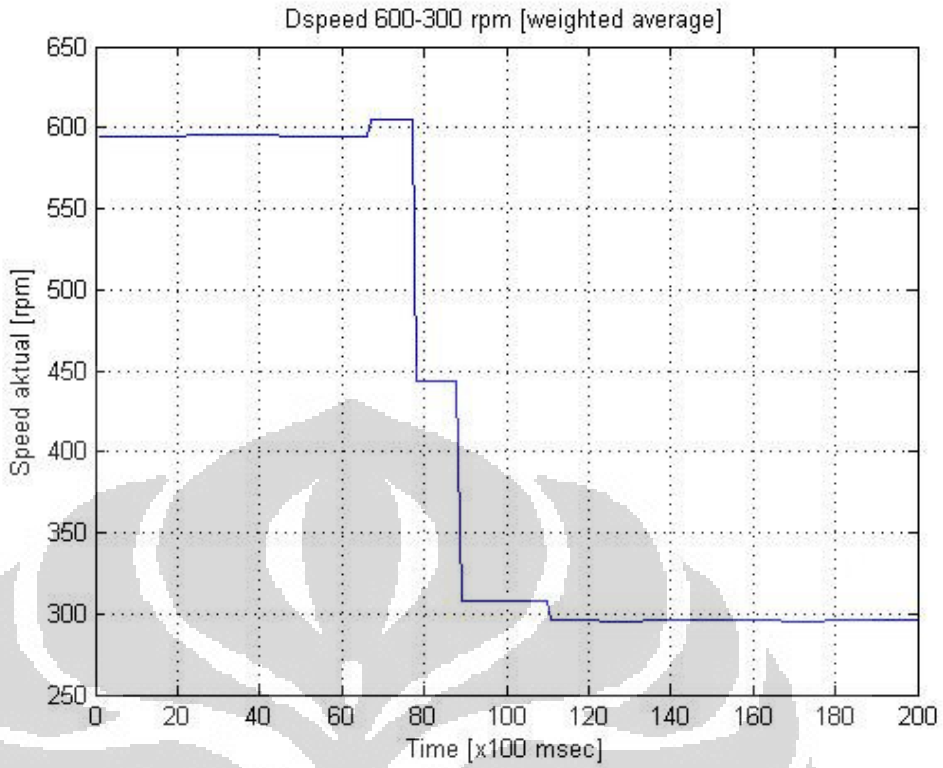
Tabel 4.7 Parameter transien perubahan *speed* 900 ke 600 rpm (*weighted average*)

Settling time (sec)	Overshoot (%)	Undershoot (%)	Error steady state (rpm / %)
2,3	1	0	5 / 0,83

Pada Tabel 4.7, parameter unjuk kerja perubahan *speed* 900 ke 600 rpm pada percobaan menggunakan metode defuzzifikasi *weighted average, settling time* yang diperoleh adalah 2,3 sec, *overshoot* adalah 1 %, dan *error steady state* yang diperoleh adalah 5 rpm / 0,83 %.

3. Perubahan *Speed* 600 Ke 300 RPM (*Weighted Average*)

Data hasil percobaan perubahan *speed* 600 ke 300 rpm dapat dilihat pada gambar berikut.



Gambar 4.14 Grafik respon transien perubahan *speed* 600 ke 300 rpm (*weighted average*)

Gambar 4.14 menggambarkan grafik transien perubahan *speed* 600 ke 300 rpm pada percobaan menggunakan metode defuzzifikasi *weighted average*. Berdasarkan grafik respon transien perubahan *speed* 600 ke 300 rpm metode defuzzifikasi *weghted average* diperoleh data sebagai berikut.

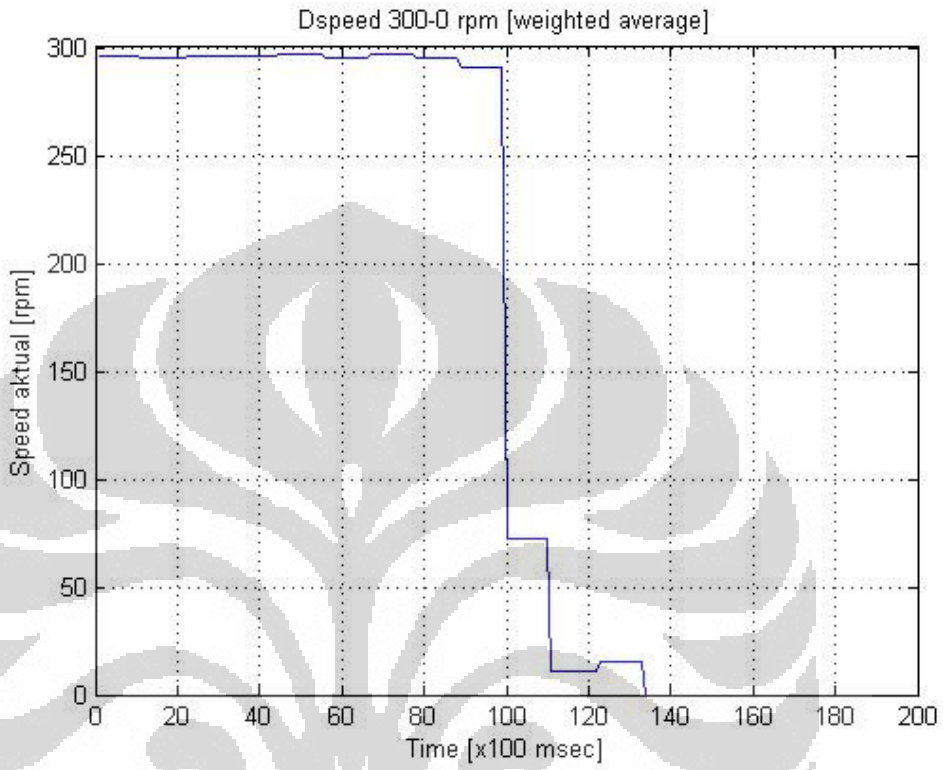
Tabel 4.8 Parameter transien perubahan *speed* 600 ke 300 rpm (*weighted average*)

Settling time (sec)	Overshoot (%)	Undershoot (%)	Error steady state (rpm / %)
4,4	0	0,83	5 / 1,67

Pada Tabel 4.8, parameter unjuk kerja perubahan *speed* 600 ke 300 rpm pada percobaan menggunakan metode defuzzifikasi *weighted average, settling time* yang diperoleh adalah 4,4 sec, *undershoot* adalah 0,83 %, dan *error steady state* yang diperoleh adalah 5 rpm / 1,67 %.

4. Perubahan *Speed* 300 Ke 0 RPM (*Weighted Average*)

Data hasil percobaan perubahan speed 300 ke 0 rpm dapat dilihat pada gambar berikut.



Gambar 4.15 Grafik respon transien perubahan *speed* 300 ke 0 rpm (*weighted average*)

Gambar 4.15 menggambarkan grafik transien perubahan *speed* 300 ke 0 rpm pada percobaan menggunakan metode defuzzifikasi *weighted average*. Berdasarkan grafik respon transien perubahan *speed* 300 ke 0 rpm metode defuzzifikasi *weighted average* diperoleh data sebagai berikut.

Tabel 4.9 Parameter transien perubahan *speed* 300 ke 0 rpm (*weighted average*)

Settling time (sec)	Overshoot (%)	Undershoot (%)	Error steady state (rpm / %)
4,5	0	0	0

Pada Tabel 4.9, parameter unjuk kerja perubahan *speed* 300 ke 0 rpm pada percobaan menggunakan metode defuzzifikasi *weighted average, settling time* yang diperoleh adalah 4,5 sec.

4.4 Percobaan Menggunakan Metode Defuzzifikasi *Middle Of Maxima*

Pada percobaan *realtime* dengan menggunakan metode defuzzifikasi *middle of maxima* dibagi ke dalam dua tahap, yaitu siklus Error *positive* (300, 600, 900, 1200) rpm dan siklus Error negative (900, 600, 300, 0) rpm.

4.4.1 Siklus Error *Positive* (300, 600, 900, 1200) RPM (*Middle Of Maxima*)

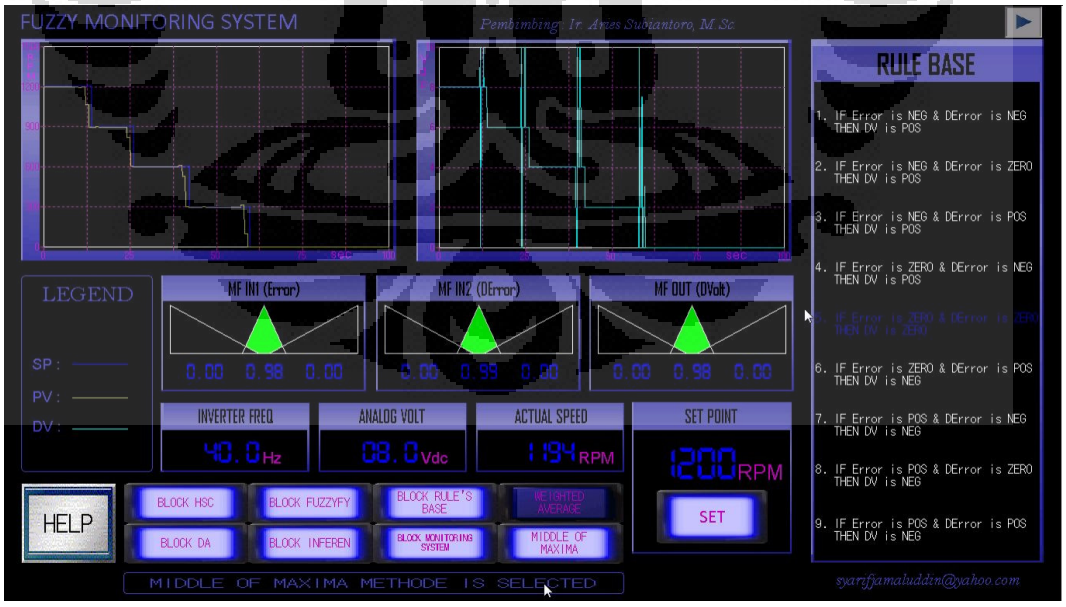
Siklus Error *positive* diperoleh jika nilai *set point* (SP) lebih besar daripada nilai *speed* aktual (PV), sedangkan nilai Error sendiri diperoleh dari hasil pengurangan antara *set point* (SP) dengan *speed* aktual (PV).

Error = SP – PV

Error *positive* = Error dimana SP > PV

Keterangan:

- SP = Nilai yang diinginkan (*Set Point*)
- PV = Nilai *speed* aktual (*Process Value*)



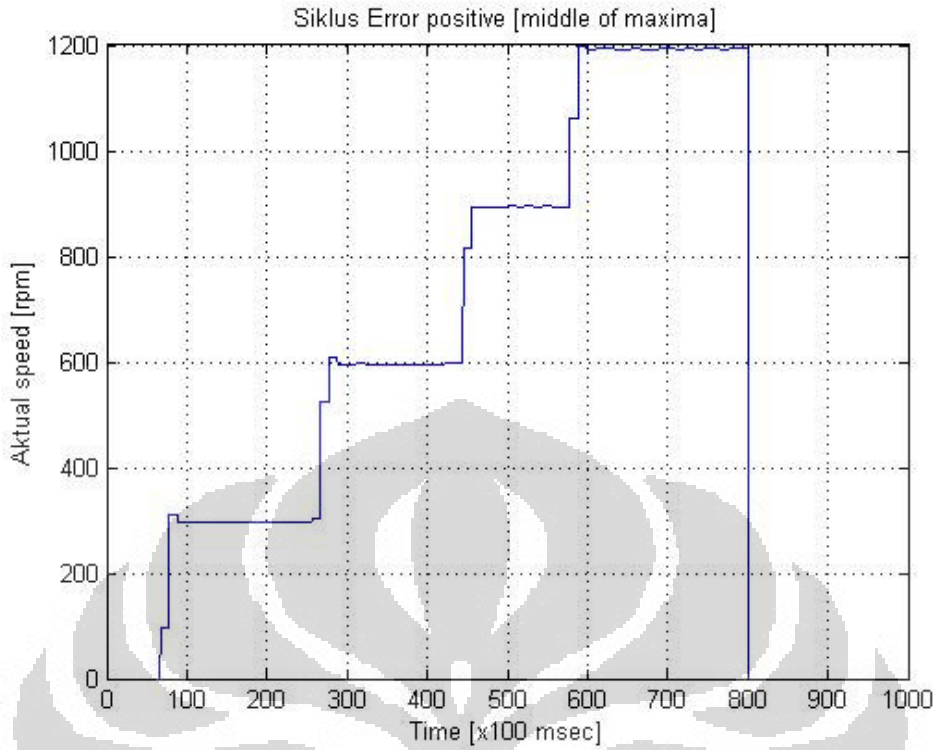
Gambar 4.16 Tampilan modul HMI siklus Error *positive* metode defuzzifikasi *middle of maxima*

Keterangan:

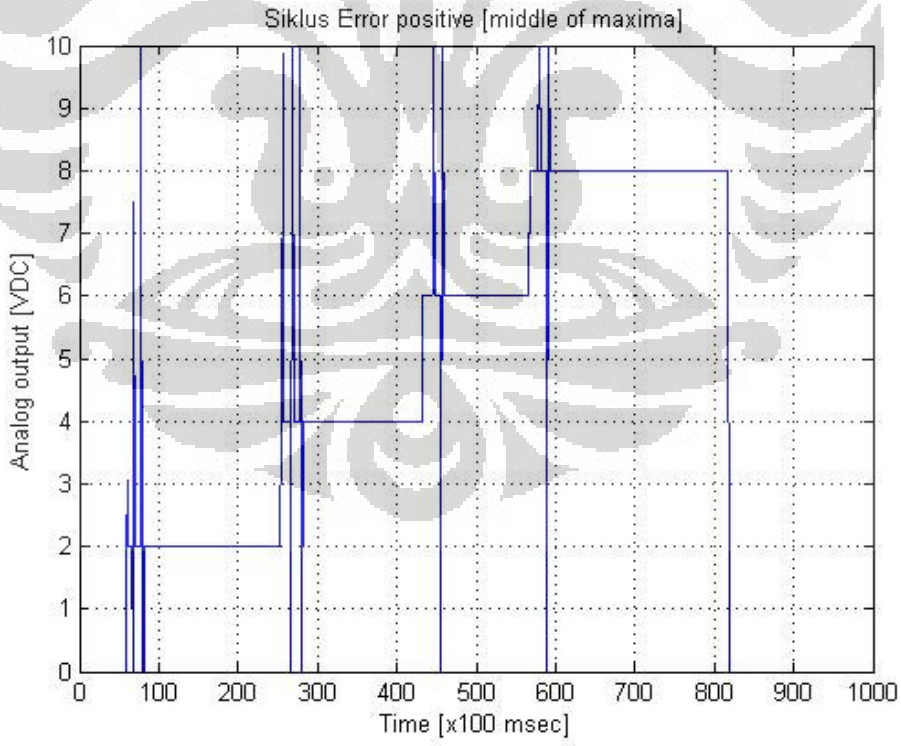
- Kolom grafik sebelah kiri: 1. garis berwarna kuning adalah nilai *process value* (PV) yang ditampilkan kedalam bentuk grafik historikal, 2. garis berwarna biru tua adalah nilai *set point* (SP) yang ditampilkan kedalam bentuk grafik historikal
- Kolom grafik sebelah kanan: garis berwarna biru muda adalah nilai *analog output* yang ditampilkan kedalam bentuk grafik historikal
- Blok MF IN1 (Error): tampilan level berwarna hijau serta tampilan *7-segment* berwarna biru adalah nilai dari masing-masing subset pada fungsi keanggotaan *input* Error *(negative, zero, positive)*
- Blok MF IN2 (DError): tampilan level berwarna hijau serta tampilan *7-segment* berwarna biru adalah nilai dari masing-masing subset pada fungsi keanggotaan *input* DError *(negative, zero, positive)*
- Blok MF OUT (DVolt): tampilan level berwarna hijau serta tampilan *7-segment* berwarna biru adalah nilai dari masing-masing subset pada fungsi keanggotaan *output* DV *(negative, zero, positive)*
- Blok INVERTER FREQ: tampilan *7-segment* berwarna biru adalah nilai dari frekuensi yang dihasilkan oleh *inverter* (Hz)
- Blok ANALOG VOLT: tampilan *7-segment* berwarna biru adalah nilai dari tegangan keluaran yang dihasilkan oleh modul *digital to analog* (Vdc)
- Blok ACTUAL SPEED: tampilan *7-segment* berwarna biru adalah nilai dari *speed* aktual / *process value* (rpm)
- Blok SET POINT: tampilan *7-segment* berwarna biru adalah nilai *set point* yang dimasukkan melalui modul HMI (rpm)
- Blok RULE BASE: sembilan tampilan pernyataan matematis pada blok RULE BASE adalah aturan-aturan *fuzzy* yang digunakan di dalam sistem, berwarna putih bila tidak aktif dan berwarna biru tua bila aktif
- BLOK HSC: *Button switch* untuk mengaktifkan blok program *high speed counter*

- BLOK DA: *Button switch* untuk mengaktifkan blok program *digital to analog*
- BLOK INFEREN1: *Button switch* untuk mengaktifkan blok program inferensi
- BLOK RULE'S BASE: *Button switch* untuk mengaktifkan blok program basis aturan *fuzzy*
- BLOK MONITORING SYSTEM: *Button switch* untuk mengaktifkan blok program sistem *monitoring fuzzy*
- BLOK FUZZYFY: *Button switch* untuk mengaktifkan blok program fuzzifikasi
- BLOK WEIGHTED AVERAGE: *Button switch* untuk mengaktifkan blok program metode defuzzifikasi *weighted average*
- BLOK MIDDLE OF MAXIMA: *Button switch* untuk mengaktifkan blok program metode defuzzifikasi *middle of maxima*

Grafik transien perubahan data *speed* pada siklus Error *positive* dengan menggunakan metode defuzzifikasi *middle of maxima* dapat dilihat pada gambar berikut.



(a)



(b)

Gambar 4.17 Grafik respon transien perubahan data *speed* siklus Error *positive* metode defuzzyfikasi *middle of maxima*

(a) Gambar *Actual speed*

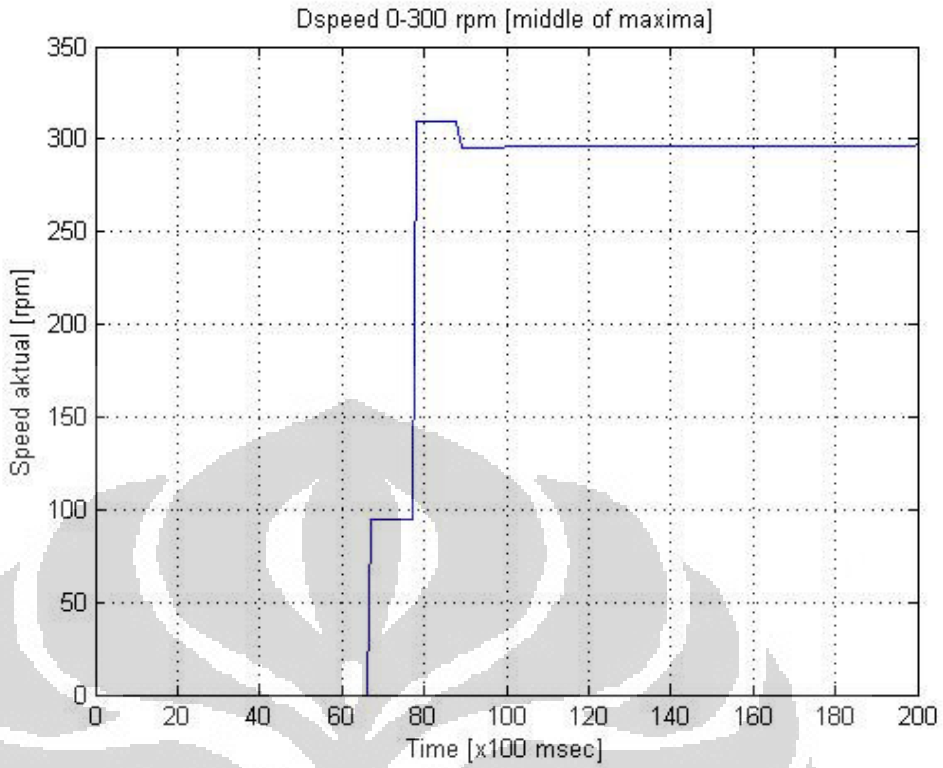
(b) Gambar *Analog output*

Gambar 4.17(a) menggambarkan grafik transien *speed* aktual sedangkan Gambar 4.17(b) menggambarkan grafik transien tegangan keluaran dari modul *digital to analog* dari siklus Error *positive* pada percobaan menggunakan metode defuzzifikasi *middle of maxima*. Siklus Error *positive* memiliki empat data perubahan speed sebagai berikut.

- Perubahan *speed* 0 ke 300 rpm
- Perubahan *speed* 300 ke 600 rpm
- Perubahan *speed* 600 ke 900 rpm
- Perubahan *speed* 900 ke 1200 rpm

1. Perubahan *Speed* 0 Ke 300 RPM (*Middle Of Maxima*)

Data hasil percobaan perubahan speed 0 ke 300 rpm dapat dilihat pada gambar berikut.



Gambar 4.18 Grafik respon transien perubahan *speed* 0 ke 300 rpm (*middle of maxima*)

Gambar 4.18 menggambarkan grafik transien perubahan *speed* 0 ke 300 rpm pada percobaan menggunakan metode defuzzifikasi *middle of maxima*. Berdasarkan grafik respon transien perubahan *speed* 0 ke 300 rpm metode defuzzfikasi *middle of maxima* diperoleh data sebagai berikut.

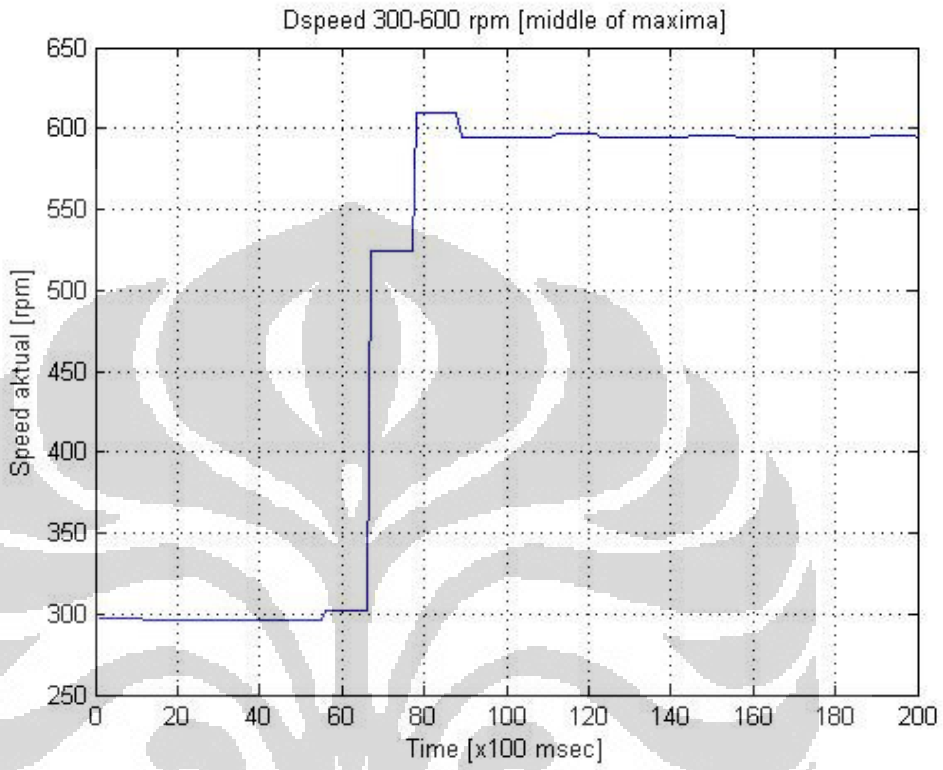
Tabel 4.10 Parameter transien perubahan *speed* 0 ke 300 rpm (*middle of maxima*)

Settling time (sec)	Overshoot (%)	Undershoot (%)	Error steady state (rpm / %)
2,3	3,33	0	5 / 1,67

Pada Tabel 4.10, parameter unjuk kerja perubahan *speed* 0 ke 300 rpm pada percobaan menggunakan metode defuzzifikasi *middle of maxima, settling time* yang diperoleh adalah 2,3 sec, *overshoot* adalah 3,33 %, dan *error steady state* adalah 5 rpm / 1,67 %.

2. Perubahan *Speed* 300 Ke 600 RPM (*Middle Of Maxima*)

Data hasil percobaan perubahan *speed* 300 ke 600 rpm dapat dilihat pada gambar berikut.



Gambar 4.19 Grafik respon transien perubahan *speed* 300 ke 600 rpm (*middle of maxima*)

Gambar 4.19 menggambarkan grafik transien perubahan *speed* 300 ke 600 rpm pada percobaan menggunakan metode defuzzifikasi *middle of maxima*. Berdasarkan grafik respon transien perubahan *speed* 300 ke 600 rpm metode defuzzfikasi *middle of maxima* diperoleh data sebagai berikut.

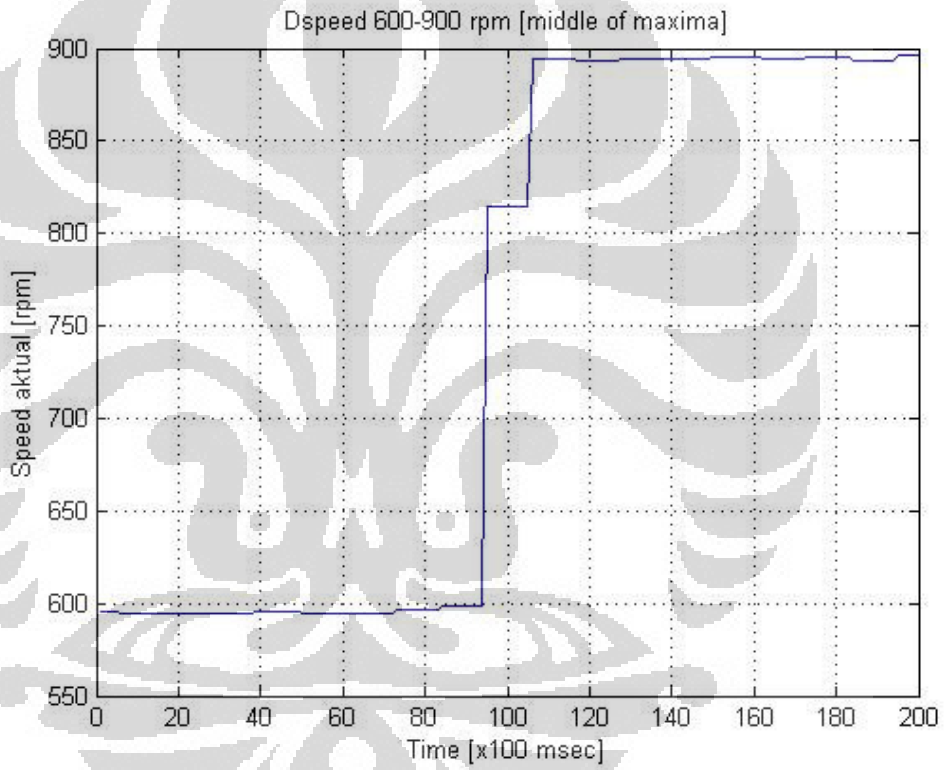
Tabel 4.11 Parameter transien perubahan *speed* 300 ke 600 rpm (*middle of maxima*)

Settling time (sec)	Overshoot (%)	Undershoot (%)	Error steady state (rpm / %)
2,3	1,67	0	5 / 0,83

Pada Tabel 4.11, parameter unjuk kerja perubahan *speed* 300 ke 600 rpm pada percobaan menggunakan metode defuzzifikasi *middle of maxima, settling time* yang diperoleh adalah 2,3 sec, *overshoot* adalah 1,67 %, dan *error steady state* adalah 5 rpm / 0,83 %.

3. Perubahan *Speed* 600 Ke 900 RPM (*Middle Of Maxima*)

Data hasil percobaan perubahan speed 600 ke 900 rpm dapat dilihat pada gambar berikut.



Gambar 4.20 Grafik respon transien perubahan *speed* 600 ke 900 rpm (*middle of maxima*)

Gambar 4.20 menggambarkan grafik transien perubahan *speed* 600 ke 900 rpm pada percobaan menggunakan metode defuzzifikasi *middle of maxima*. Berdasarkan grafik respon transien perubahan *speed* 600 ke 900 rpm metode defuzzifikasi *middle of maxima* diperoleh data sebagai berikut.

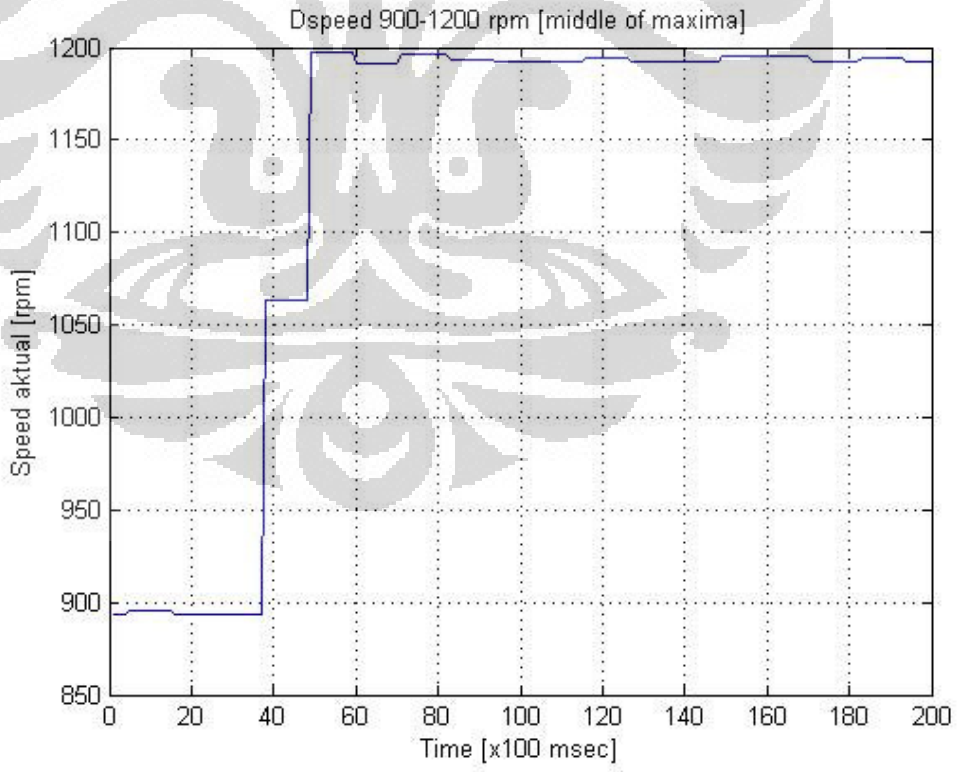
Tabel 4.12 Paramter transien perubahan *speed* 600 ke 900 rpm (*middle of maxima*)

Settling time (sec)	Overshoot (%)	Undershoot (%)	Error steady state (rpm / %)
2,3	0	0	5 / 0,56

Pada Tabel 4.12, parameter unjuk kerja perubahan *speed* 600 ke 900 rpm pada percobaan menggunakan metode defuzzifikasi *middle of maxima, settling time* yang diperoleh adalah 2,3 sec dan *error steady state* adalah 5 rpm / 0,56 %.

4. Perubahan *Speed* 900 Ke 1200 RPM (*Middle Of Maxima*)

Data hasil percobaan perubahan *speed* 900 ke 1200 rpm dapat dilihat pada gambar berikut.



Gambar 4.21 Grafik respon transien perubahan *speed* 900 ke 1200 rpm (*middle of maxima*)

Gambar 4.21 menggambarkan grafik transien perubahan *speed* 900 ke 1200 rpm pada percobaan menggunakan metode defuzzifikasi *middle of maxima*. Berdasarkan grafik respon transien perubahan *speed* 900 ke 1200 rpm metode defuzzifikasi *middle of maxima* diperoleh data sebagai berikut.

Tabel 4.13 Paraeter transien perubahan *speed* 900 ke 1200 rpm (*middle of maxima*)

Settling time (sec)	Overshoot (%)	Undershoot (%)	Error steady state (rpm / %)
3,4	0	0	7 / 0,58

Pada Tabel 4.13, parameter unjuk kerja perubahan *speed* 900 ke 1200 rpm pada percobaan menggunakan metode defuzzifikasi *middle of maxima, settling time* yang diperoleh adalah 3,4 sec dan *error steady state* adalah 7 rpm / 0,58 %.

4.4.2 Siklus Error *Negative* (900, 600, 300, 0) RPM (*Middle Of Maxima*)

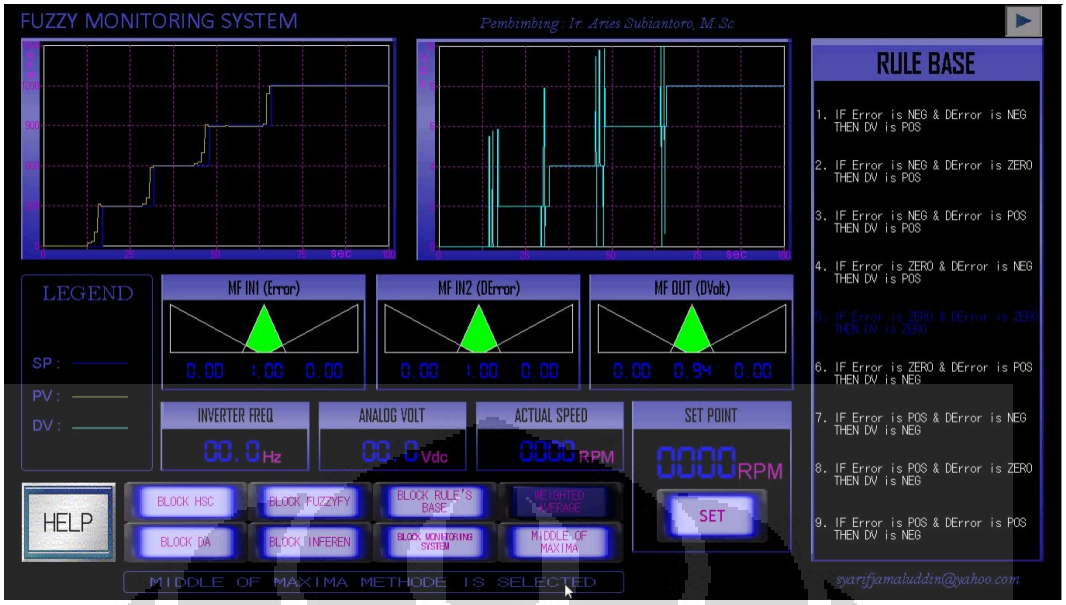
Siklus Error *negative* diperoleh jika nilai *set point* (SP) lebih kecil daripada nilai *speed* aktual (PV), sedangkan nilai Error sendiri diperoleh dari hasil pengurangan antara *set point* (SP) dengan *speed* aktual (PV).

Error = SP – PV

Error *positive* = Error dimana SP < PV

Keterangan:

- SP = Nilai yang diinginkan (*Set Point*)
- PV = Nilai *speed* aktual (*Process Value*)



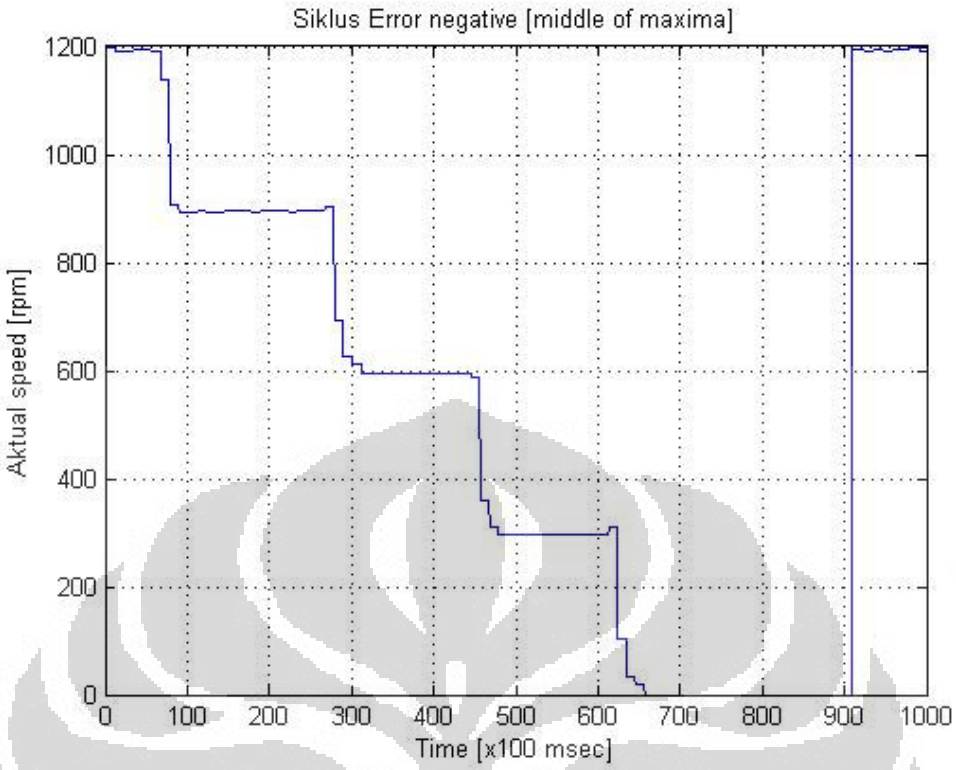
Gambar 4.22 Tampilan modul HMI siklus Error *negative* metode defuzzifikasi *middle of maxima*

Keterangan:

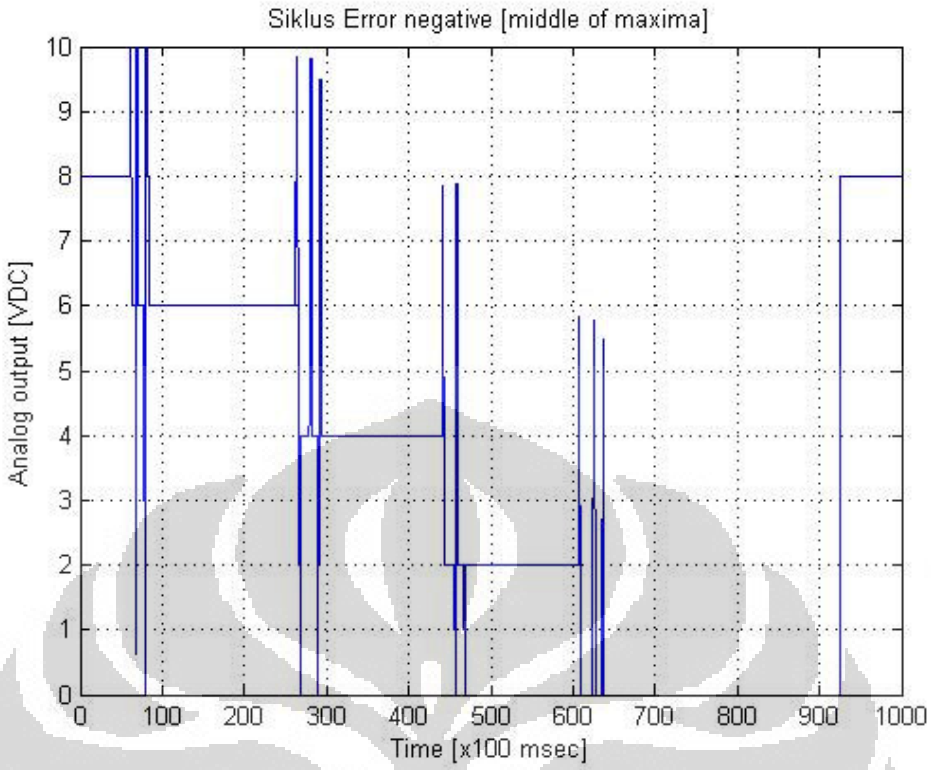
- Kolom grafik sebelah kiri: 1. garis berwarna kuning adalah nilai *process value* (PV) yang ditampilkan kedalam bentuk grafik historikal, 2. garis berwarna biru tua adalah nilai *set point* (SP) yang ditampilkan kedalam bentuk grafik historikal
- Kolom grafik sebelah kanan: garis berwarna biru muda adalah nilai *analog output* yang ditampilkan kedalam bentuk grafik historikal
- Blok MF IN1 (Error): tampilan level berwarna hijau serta tampilan *7-segment* berwarna biru adalah nilai dari masing-masing subset pada fungsi keanggotaan *input* Error *(negative, zero, positive)*
- Blok MF IN2 (DError): tampilan level berwarna hijau serta tampilan *7-segment* berwarna biru adalah nilai dari masing-masing subset pada fungsi keanggotaan *input* DError *(negative, zero, positive)*
- Blok MF OUT (DVolt): tampilan level berwarna hijau serta tampilan *7-segment* berwarna biru adalah nilai dari masing-masing subset pada fungsi keanggotaan *output* DV *(negative, zero, positive)*
- Blok INVERTER FREQ: tampilan *7-segment* berwarna biru adalah nilai dari frekuensi yang dihasilkan oleh *inverter* (Hz)

- Blok ANALOG VOLT: tampilan *7-segment* berwarna biru adalah nilai dari tegangan keluaran yang dihasilkan oleh modul *digital to analog* (Vdc)
- Blok ACTUAL SPEED: tampilan *7-segment* berwarna biru adalah nilai dari *speed* aktual / *process value* (rpm)
- Blok SET POINT: tampilan *7-segment* berwarna biru adalah nilai *set point* yang dimasukkan melalui modul HMI (rpm)
- Blok RULE BASE: sembilan tampilan pernyataan matematis pada blok RULE BASE adalah aturan-aturan *fuzzy* yang digunakan di dalam sistem, berwarna putih bila tidak aktif dan berwarna biru tua bila aktif
- BLOK HSC: *Button switch* untuk mengaktifkan blok program *high speed counter*
- BLOK DA: *Button switch* untuk mengaktifkan blok program *digital to analog*
- BLOK INFEREN1: *Button switch* untuk mengaktifkan blok program inferensi
- BLOK RULE'S BASE: *Button switch* untuk mengaktifkan blok program basis aturan *fuzzy*
- BLOK MONITORING SYSTEM: *Button switch* untuk mengaktifkan blok program sistem *monitoring fuzzy*
- BLOK FUZZYFY: *Button switch* untuk mengaktifkan blok program fuzzifikasi
- BLOK WEIGHTED AVERAGE: *Button switch* untuk mengaktifkan blok program metode defuzzifikasi *weighted average*
- BLOK MIDDLE OF MAXIMA: *Button switch* untuk mengaktifkan blok program metode defuzzifikasi *middle of maxima*

Grafik transien perubahan data *speed* pada siklus Error *negative* dengan menggunakan metode defuzzifikasi *middle of maxima* dapat dilihat pada gambar berikut.



(a)



(b)

Gambar 4.23 Grafik transien perubahan data *speed* siklus Error *negative* metode defuzzyfikasi *middle of maxima*

(c) Gambar *Actual speed*

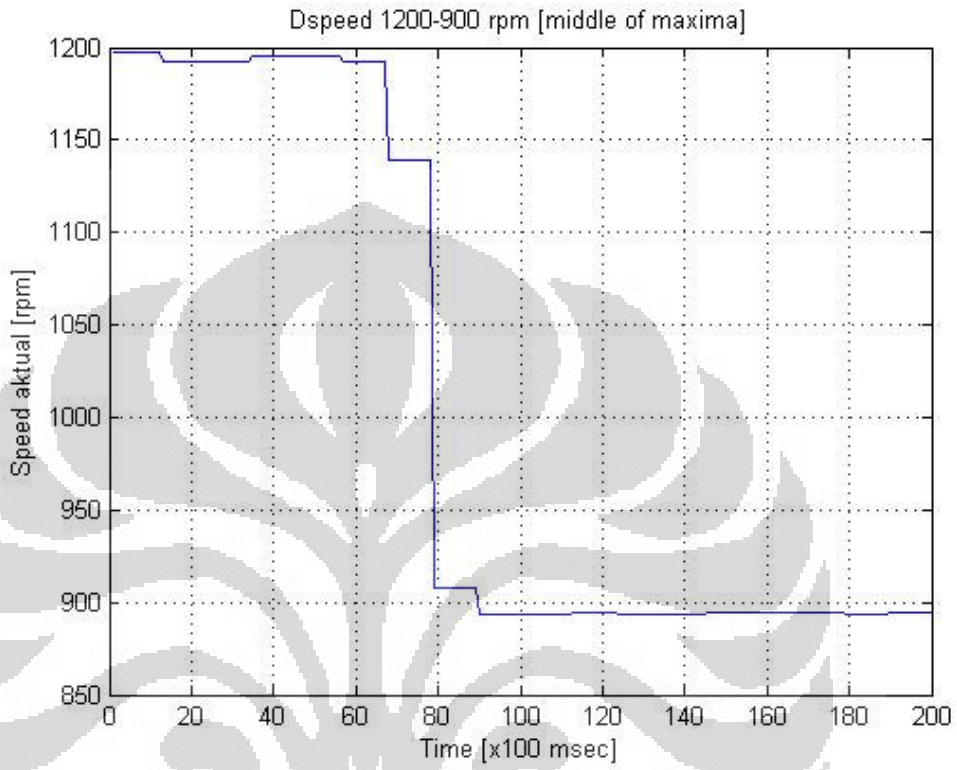
(d) Gambar *Analog output*

Gambar 4.23(a) menggambarkan grafik transien *speed* aktual sedangkan Gambar 4.23(b) menggambarkan grafik transien tegangan keluaran dari modul *digital to analog* dari siklus Error *negative* pada percobaan menggunakan metode defuzzifikasi *middle of maxima*. Siklus Error *negative* memiliki empat data perubahan speed sebagai berikut.

- Perubahan *speed* 1200 ke 900 rpm
- Perubahan *speed* 900 ke 600 rpm
- Perubahan *speed* 600 ke 300 rpm
- Perubahan *speed* 300 ke 0 rpm

1. Perubahan *Speed* 1200 Ke 900 RPM (*Middle Of Maxima*)

Data hasil percobaan perubahan *speed* 1200 ke 900 rpm dapat dilihat pada gambar berikut.



Gambar 4.24 Grafik respon transien perubahan *speed* 1200 ke 900 rpm (*middle of maxima*)

Gambar 4.24 menggambarkan grafik transien perubahan *speed* 1200 ke 900 rpm pada percobaan menggunakan metode defuzzifikasi *middle of maxima*. Berdasarkan grafik respon transien perubahan *speed* 1200 ke 900 rpm metode defuzzifikasi *middle of maxima* diperoleh data sebagai berikut.

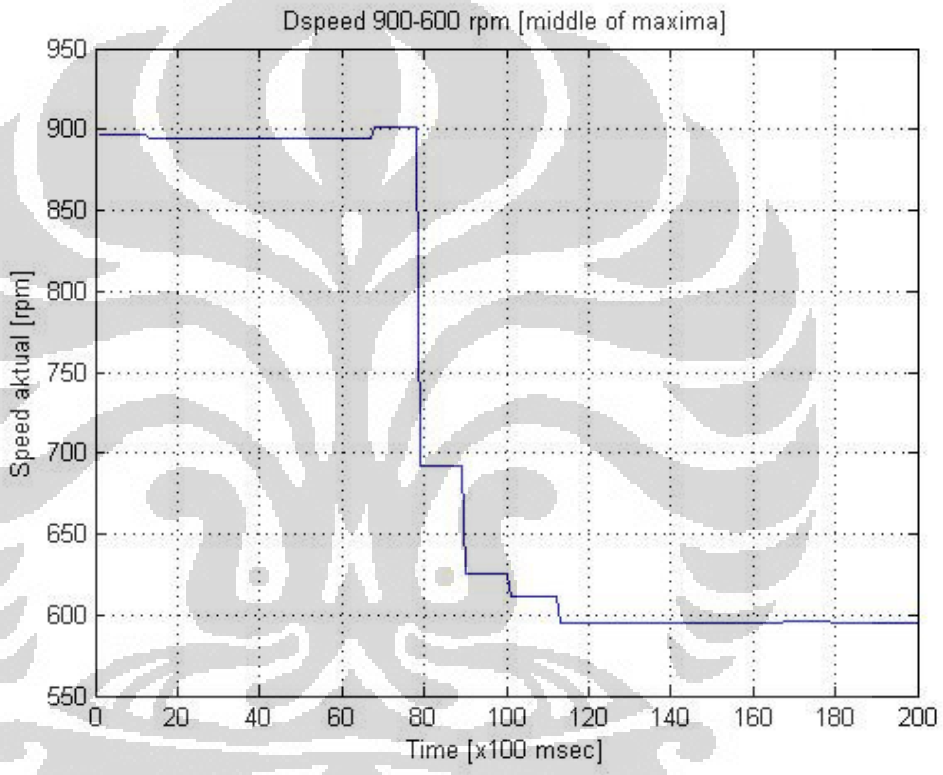
Tabel 4.14 Parameter transien perubahan *speed* 1200 ke 900 rpm (*middle of maxima*)

Settling time (sec)	Overshoot (%)	Undershoot (%)	Error steady state (rpm / %)
2,3	0	0	6 / 0,67

Pada Tabel 4.14, parameter unjuk kerja perubahan *speed* 1200 ke 900 rpm pada percobaan menggunakan metode defuzzifikasi *middle of maxima, settling time* yang diperoleh adalah 2,3 sec dan *error steady state* adalah 6 rpm / 0,67 %.

2. Perubahan *Speed* 900 Ke 600 RPM (*Middle Of Maxima*)

Data hasil percobaan perubahan *speed* 900 ke 600 rpm dapat dilihat pada gambar berikut.



Gambar 4.25 Grafik respon transien perubahan *speed* 900 ke 600 rpm (*middle of maxima*)

Gambar 4.25 menggambarkan grafik transien perubahan *speed* 900 ke 600 rpm pada percobaan menggunakan metode defuzzifikasi *middle of maxima*. Berdasarkan grafik respon transien perubahan *speed* 900 ke 600 rpm metode defuzzifikasi *middle of maxima* diperoleh data sebagai berikut.

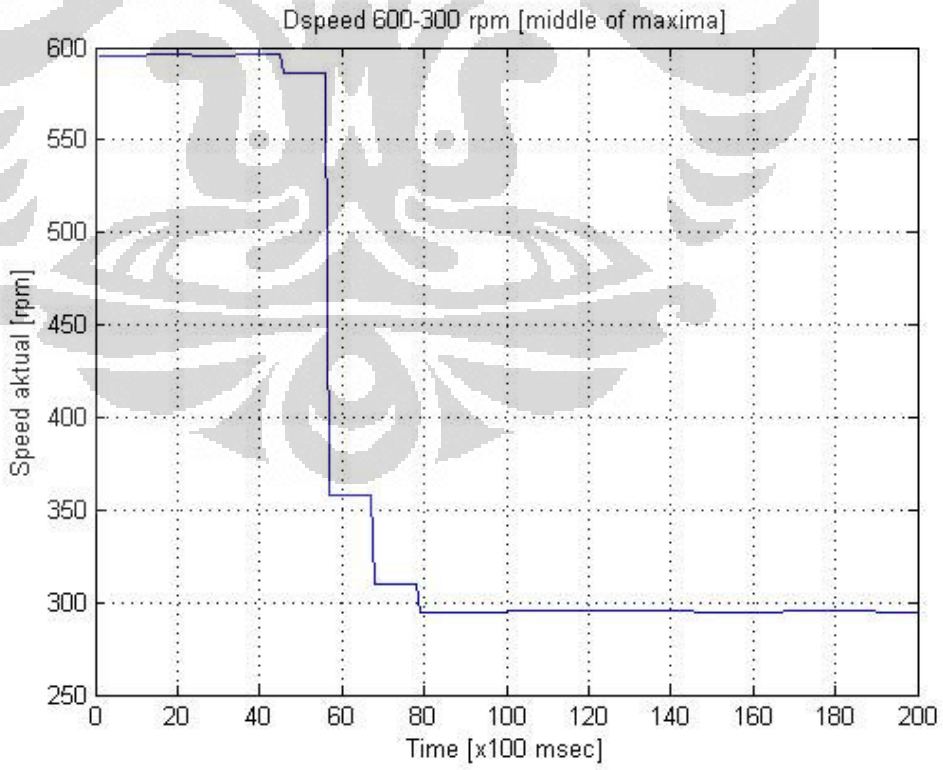
Tabel 4.15 Parameter transien perubahan *speed* 900 ke 600 rpm (*middle of maxima*)

Settling time (sec)	Overshoot (%)	Undershoot (%)	Error steady state (rpm / %)
4,6	0	0,22	5 / 0,83

Pada Tabel 4.15, parameter unjuk kerja perubahan *speed* 900 ke 600 rpm pada percobaan menggunakan metode defuzzifikasi *middle of maxima, settling time* yang diperoleh adalah 4,6 sec, *undershoot* adalah 0,22 %, dan *error steady state* adalah 5 rpm / 0,83 %.

3. Perubahan *Speed* 600 Ke 300 RPM (*Middle Of Maxima*)

Data hasil percobaan perubahan *speed* 600 ke 300 rpm dapat dilihat pada gambar berikut.



Gambar 4.26 Grafik respon transien perubahan *speed* 600 ke 300 rpm (*middle of maxima*)

Gambar 4.26 menggambarkan grafik transien perubahan *speed* 600 ke 300 rpm pada percobaan menggunakan metode defuzzifikasi *middle of maxima*. Berdasarkan grafik respon transien perubahan *speed* 600 ke 300 rpm metode defuzzifikasi *middle of maxima*

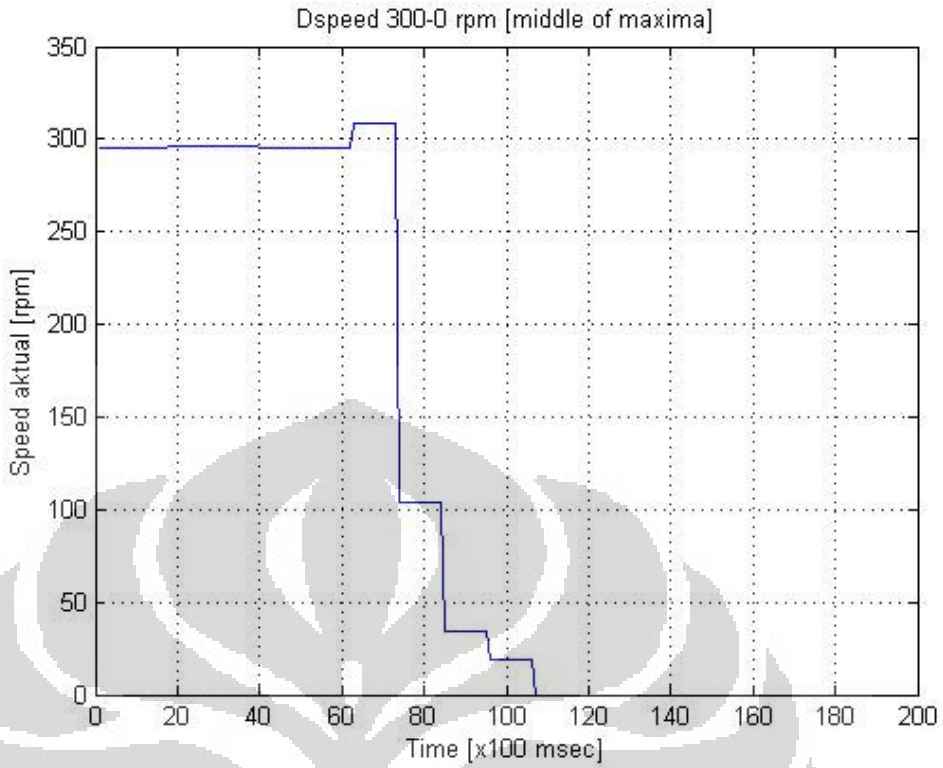
Tabel 4.16 Parameter transien perubahan *speed* 600 ke 300 rpm (*middle of maxima*)

Settling time (sec)	Overshoot (%)	Undershoot (%)	Error steady state (rpm / %)
3,4	0	0	5 / 1,67

Pada Tabel 4.16, parameter unjuk kerja perubahan *speed* 600 ke 300 rpm pada percobaan menggunakan metode defuzzifikasi *middle of maxima, settling time* yang diperoleh adalah 3,4 sec dan *error steady state* adalah 5 rpm / 1,67 %.

4. Perubahan *Speed* 300 Ke 0 RPM (*Middle Of Maxima*)

Data hasil percobaan perubahan *speed* 300 ke 0 rpm dapat dilihat pada gambar berikut.



Gambar 4.27 Grafik respon transien perubahan *speed* 300 ke 0 rpm (*middle of maxima*)

Gambar 4.27 menggambarkan grafik transien perubahan *speed* 600 ke 300 rpm pada percobaan menggunakan metode defuzzifikasi *middle of maxima*. Berdasarkan grafik respon transien perubahan *speed* 300 ke 0 rpm metode defuzzifikasi *middle of maxima*

Tabel 4.17 Parameter transien perubahan *speed* 300 ke 0 rpm (*middle of maxima*)

Settling time (sec)	Overshoot (%)	Undershoot (%)	Error steady state (rpm / %)
4,5	0	3	0

Pada Tabel 4.17, parameter unjuk kerja perubahan *speed* 300 ke 0 rpm pada percobaan menggunakan metode defuzzifikasi *middle of maxima, settling time* yang diperoleh adalah 4,5 sec dan *undershoot* adalah 3 %.

4.5 Analisis Data

Beberapa parameter unjuk kerja yang digunakan sebagai bahan analisis berdasarkan grafik respon transien perubahan *speed* pada siklus Error *positive* dan *negative* adalah sebagai berikut.

- *Error steady state* (e_{ss}) adalah nilai selisih antara nilai *set point* dengan nilai aktual *speed / process value* pada kondisi *steady state*.
- *Overshoot (% OS)* adalah nilai puncak lewatan maksimum pada respon transien, biasanya dinyatakan dalam prosentase selisih nilai *set point* dengan *process value* terhadap nilai *set point* itu sendiri. Besarnya prosentase menunjukkan kestabilan relatif sistem.
- *Undershoot (% US)* adalah nilai lewatan minimum pada respon transien, dinyatakan dalam prosentase selisih nilai awal dengan *process value* terhadapa nilai awal itu sendiri.
- *Settling time* (T_s) adalah besarnya waktu yang diperlukan oleh osilasi teredam (*damped*) transien untuk bertahan pada ±2% nilai akhir.

4.5.1 Rekapitulasi Parameter Unjuk Kerja Pada Percobaan Menggunakan Metode Defuzzifikasi *Weighted Average*

Rekapitulasi parameter unjuk kerja pada percobaan menggunakan metode defuzzifikasi *weighted average* dibagi menjadi dua bagian, percobaan pada siklus Error *positive* dan *negative*.

1. **Siklus Error *positive (weighted average)***

   Rekapitulasi parameter unjuk kerja siklus Error *positive* metode defuzzifikasi *weighted average* dapat dilihat pada tabel berikut.

Tabel 4.18 Rekapitulasi parameter unjuk kerja siklus Error *positive* (*weighted average*)

No.	Δspeed (rpm)	Settling time (sec)	Overshoot (%)	Undershoot (%)	Error steady state (rpm / %)
1	0 ke 300	2,3	2,67	0	4 / 1,33
2	300 ke 600	2,3	1,33	0	5 / 0,83
3	600 ke 900	2,3	0	0	6 / 0,67
4	900 ke 1200	3,4	0,92	0	8 / 0,67

Berdasarkan Tabel 4.18, 1. Perubahan *speed* 0 ke 300 rpm, *settling time* yang diperoleh adalah 2,3 sec, *overshoot* adalah 2,67 %, dan *error steady state* adalah 4 rpm / 1,33 %, 2. Perubahan *speed* 300 ke 600 rpm, *settling time* yang diperoleh adalah 2,3 sec, *overshoot* adalah 1,33 %, dan *error steady state* adalah 5 rpm / 0,83 %, 3. Perubahan *speed* 600 ke 900 rpm, *settling time* yang diperoleh adalah 2,3 sec dan *error steady state* adalah 6 rpm / 0,67, 4. Perubahan *speed* 900 ke 1200 rpm, *settling time* yang diperoleh adalah 3,4 sec, *overshoot* adalah 0,92 %, *error steady state* adalah 8 rpm / 0,67 %. Nilai rata-rata parameter unjuk kerja respon transien dari siklus Error *positive* pada percobaan menggunakan metode defuzzifikasi *weighted average* adalah sebagai berikut.

- Nilai rata-rata *settling time* (T_s) :

  $= \frac{2,3 + 2,3 + 2,3 + 3,4}{4}$

  = 2,575 sec

- Nilai rata-rata *overshoot (% OS)* :

  $= \frac{2,67 + 1,33 + 0,92}{4}$

  = 1,23 %

- Nilai rata-rata *error steady state* (e_{ss}) :

$$= \frac{1,33 + 0,83 + 0,67 + 0,67}{4}$$

$$= 0,875\ \%$$

2. Siklus Error *negative (weighted average)*

Rekapitulasi parameter unjuk kerja siklus Error *negative* metode defuzzifikasi *weighted average* dapat dilihat pada tabel berikut.

Tabel 4.19 Rekapitulasi parameter unjuk kerja siklus Error *negative* (*weighted average*)

No.	Δspeed (rpm)	Settling time (sec)	Overshoot (%)	Undershoot (%)	Error steady state (rpm / %)
1	1200 ke 900	2,3	0,89	0	6 / 0,67
2	900 ke 600	2,3	1	0	5 / 0,83
3	600 ke 300	4,4	0	0,83	5 / 1,67
4	300 ke 0	4,5	0	0	0

Berdasarkan Tabel 4.19, 1. Perubahan *speed* 1200 ke 900 rpm, *settling time* yang diperoleh adalah 2,3 sec, *overshoot* adalah 0,89 %, dan *error steady state* adalah 6 rpm / 0,67 %, 2. Perubahan *speed* 900 ke 600 rpm, *settling time* yang diperoleh adalah 2,3 sec, *overshoot* adalah 1 %, dan *error steady state* adalah 5 rpm / 0,83 %, 3. Perubahan *speed* 600 ke 300 rpm, *settling time* yang diperoleh adalah 4,4 sec, *undershoot* adalah 0,83 %, dan *error steady state* adalah 5 rpm / 1,67, 4. Perubahan *speed* 300 ke 0 rpm, *settling time* yang diperoleh adalah 4,5 sec. Nilai rata-rata parameter unjuk kerja respon transien dari siklus Error *negative* pada

percobaan menggunakan metode defuzzifikasi *weighted average* adalah sebagai berikut.

- Nilai rata-rata *settling time* (T_s) :

  $= \frac{2,3 + 2,3 + 4,4 + 4,5}{4}$

  = 3,375 sec

- Nilai rata-rata *overshoot (% OS)* :

  $= \frac{0,89 + 1}{4}$

  = 0,472 %

- Nilai rata-rata *undershoot (% US)* :

  $= \frac{0,83}{4}$

  = 0,207 %

- Nilai rata-rata *error steady state* (e_{ss}) :

  $= \frac{0,67 + 0,83 + 1,67}{4}$

  = 0,792 %

4.5.2 Rekapitulasi Parameter Unjuk Kerja Pada Percobaan Menggunakan Metode Defuzzifikasi *Middle Of Maxima*

Rekapitulasi parameter unjuk kerja pada percobaan menggunakan metode defuzzifikasi *middle of maxima* dibagi menjadi dua bagian, percobaan pada siklus Error *positive* dan *negative*.

1. **Siklus Error *positive (middle of maxima)***

   Rekapitulasi parameter unjuk kerja siklus Error *positive* metode defuzzifikasi *middle of maxima* dapat dilihat pada tabel berikut.

Tabel 4.20 Rekapitulasi parameter unjuk kerja siklus Error *positive* (*middle of maxima*)

No.	Δspeed (rpm)	Settling time (sec)	Overshoot (%)	Undershoot (%)	Error steady state (rpm / %)
1	0 ke 300	2,3	3,33	0	5 / 1,67
2	300 ke 600	2,3	1,67	0	5 / 0,83
3	600 ke 900	2,3	0	0	5 / 0,56
4	900 ke 1200	3,4	0	0	7 / 0,58

Berdasarkan Tabel 4.20, 1. Perubahan *speed* 0 ke 300 rpm, *settling time* yang diperoleh adalah 2,3 sec, *overshoot* adalah 3,33 %, dan *error steady state* adalah 5 rpm / 1,67 %, 2. Perubahan *speed* 300 ke 600 rpm, *settling time* yang diperoleh adalah 2,3 sec, *overshoot* adalah 1,67 %, dan *error steady state* adalah 5 rpm / 0,83 %, 3. Perubahan *speed* 600 ke 900 rpm, *settling time* yang diperoleh adalah 2,3 sec dan *error steady state* adalah 5 rpm / 0,56, 4. Perubahan *speed* 900 ke 1200 rpm, *settling time* yang diperoleh adalah 3,4 sec dan *error steady state* adalah 7 rpm / 0,58 %. Nilai rata-rata parameter unjuk kerja respon transien dari siklus Error *positive* pada percobaan menggunakan metode defuzzifikasi *middle of maxima* adalah sebagai berikut.

- Nilai rata-rata *settling time* (T_s) :

  $= \frac{2,3 + 2,3 + 2,3 + 3,4}{4}$

  = 2,575 sec

- Nilai rata-rata *overshoot (% OS)* :

  $= \frac{3,33 + 1,67}{4}$

  = 1,25 %

- Nilai rata-rata *error steady state* (e_{ss}) :

$$= \frac{1{,}67 + 0{,}83 + 0{,}56 + 0{,}58}{4}$$

$= 0{,}91\ \%$

2. Siklus Error *negative (middle of maxima)*

Rekapitulasi parameter unjuk kerja siklus Error *negative* metode defuzzifikasi *middle of maxima* dapat dilihat pada tabel berikut.

Tabel 4.21 Rekapitulasi parameter unjuk kerja siklus Error *negative* (*middle of maxima*)

No.	Δspeed (rpm)	Settling time (sec)	Overshoot (%)	Undershoot (%)	Error steady state (rpm / %)
1	1200 ke 900	2,3	0	0	6 / 0,67
2	900 ke 600	4,6	0	0,22	5 / 0,83
3	600 ke 300	3,4	0	0	5 / 1,67
4	300 ke 0	4,5	0	3	0

Berdasarkan Tabel 4.21, 1. Perubahan *speed* 1200 ke 900 rpm, *settling time* yang diperoleh adalah 2,3 sec dan *error steady state* adalah 6 rpm / 0,67 %, 2. Perubahan *speed* 900 ke 600 rpm, *settling time* yang diperoleh adalah 4,6 sec, *undershoot* adalah 0,22 %, dan *error steady state* adalah 5 rpm / 0,83 %, 3. Perubahan *speed* 600 ke 300 rpm, *settling time* yang diperoleh adalah 3,4 sec dan *error steady state* adalah 5 rpm / 1,67, 4. Perubahan *speed* 300 ke 0 rpm, *settling time* yang diperoleh adalah 4,5 sec dan *undershoo* adalah 3 %. Nilai rata-rata parameter unjuk kerja respon transien dari siklus Error *negative* pada percobaan menggunakan metode defuzzifikasi *middle of maxima* adalah sebagai berikut.

- Nilai rata-rata *settling time* (T_s) :

  $= \frac{2,3 + 4,6 + 3,4 + 4,5}{4}$

  = 3,7 sec

- Nilai rata-rata *undershoot (% US)* :

  $= \frac{0,22 + 3}{4}$

  = 0,805 %

- Nilai rata-rata *error steady state* (e_{ss}) :

  $= \frac{0,67 + 0,83 + 1,67}{4}$

  = 0,792 %

BAB 5
KESIMPULAN

Dari hasil percobaan yang telah dilakukan dapat diambil kesimpulan bahwa:

1. Berdasarkan data hasil pengukuran, diketahui bahwa karakteristik dari masing-masing *hardware* yang digunakan adalah linier.
2. Module *controller*, modul *inverter*, dan modul sensor kecepatan dapat bekerja dengan baik.
3. *Speed* aktual (PV) hasil pembacaan modul sensor kecepatan, *set point* (SP), dan tegangan keluaran dari modul *analog output* (DV) sudah berhasil ditampilkan kedalam bentuk grafik historikal secara *realtime* melalui SoftGOT1000.
4. Data log *speed* aktual (PV) dan *analog output* (DV) sudah berhasil ditampilkan kembali kedalam bentuk grafik respon transien perubahan *speed*.
5. Berdasarkan parameter unjuk kerja yang didapatkan dari grafik respon transien pada siklus Error *positive,* rata-rata *settling time* dengan menggunakan metode defuzzifikasi *weighted average* sama dengan *middle of maxima* adalah 2,575 sec, rata-rata *overshoot* dengan menggunakan metode defuzzifikasi *weighted average* adalah 1,23% sedangkan dengan menggunakan metode *middle of maxima* adalah 1,25%, rata-rata *error steady state* dengan menggunakan metode defuzzifikasi *weighted average* adalah 0,875% sedangkan menggunakan metode *middle of maxima* adalah 0,91%.
6. Berdasarkan parameter unjuk kerja yang didapatkan dari grafik respon transien pada siklus Error *negative,* rata-rata *settling time* dengan menggunakan metode defuzzifikasi *weighted average* adalah 3,375 sec sedangkan dengan menggunakan metode *middle of maxima* adalah 3,7 sec, rata-rata *overshoot* dengan menggunakan metode defuzzifikasi *weighted average* adalah 0,472% sedangkan dengan menggunakan metode *middle of maxima* adalah 0, rata-rata *undershoot* dengan menggunakan metode

defuzzifikasi *weighted average* adalah 0,207% sedangkan dengan menggunakan metode *middle of maxima* adalah 0,805%, rata-rata *error steady state* dengan menggunakan metode defuzzifikasi *weighted average* sama dengan *middle of maxima* adalah 0,792%.

7. Berdasarkan hasil perbandingan rata-rata parameter unjuk kerja yang diperoleh dari grafik respon transien, respon dan stabilitas yang dihasilkan oleh metode defuzzifikasi *weighted average* relatif lebih baik dari metode *middle of maxima.*

REFERENSI

[1] Banks, Walter & Gordon, Hayward. (2002). *Fuzzy Logic in Embeddeed Microcomputer and Control Systems.*Canada: Byte Craft Ltd.

[2] Ibrahim, Ahmad M. (2003). *Fuzzy Logic for Embedded Systems Applications.*USA: Elsevier Science.

[3] Klir, George J & Bo, Yuan. (1995). *Fuzzy Sets and Fuzzy Logics: Theory and Applications.* New Jersey: Prentice-Hall Inc.

[4] Mitsubishi. (2006). *Analog-Digital Converter Module User's Manual.* Mitsubishi Electric.

[5] Mitsubishi. (2005). *FATEC Inverter Beginner Course*. Mitsubishi Electric.

[6] Mitsubishi. (2006). *Graphic Operation Terminal Training Manual*. Mitsubishi Electric.

[7] Mitsubishi. (2009). *GT-SoftGOT1000 Operating Manual*. Mitsubishi Electric.

[8] Mitsubishi. (2004). *High Performance Model QCPU (Q MODE) User's Manual (Hardware Design, Maintenance and Inspection).* Mitsubishi Electric.

[9] Mitsubishi. (2003). *High-Speed Counter Module User's Manual.* Mitsubishi Electric

[10] Mitsubishi. (2005). *Inverter FR-A700 Instruction Manual (Applied).* Mitsubishi Electric.

[11] Mitsubishi. (2004). *QCPU (Q MODE) / QnACPU Programming Manual (Common Instruction).* Mitsubishi Electric.

[12] Ogata, Katsuhiko. (1992). *Modern Control Engineering.* Minnesot: Prentice-Hall Inc.

[13] Pitowarno, Endra. (2006). *Desain, Kontrol, dan Kecerdasan Buatan*. Yogyakarta: Andi Offset.

[14] Ross, Timothy J. (2004). *Fuzzy Logic With Engineering Applications* (2nd ed.). USA: Jhon Wiley & Sons Ltd.