ISSN: 2085-6989

PENGENDALIAN KECEPATAN MOTOR DC DENGAN METODA *FUZZY LOGIC* BERBASIS MIKROKONTROLLER

Oleh
Nasrul
Staf Pengajar Teknik Elektro Politeknik Negeri Padang

ABSTRACT

The Technology and information development involve production process in industries using microcontroller as a brain in control process. The number of control process with microcontroller using Fuzzy Logic method to get the function as is needed. Motors DC are used in some equipment as a driver, not only in small scale but also in huge scale. It used in low or high speed too. The way of controlled chosen depend on the function of DC motor movement. The another method is Pulse Width Modulation (PWM). This is an effective method to controlled DC motor. This method produces square pulses which have specific comparison between high pulse and low pulse. It is usual scale from 0% to 100%. In this research, both Fuzzy Logic method and Pulse Width Modulation (PWM) method base of microcontroller ATMega 8535, both are integrated to control lthe DC motor speed.

Keyword: *Microcontroller, Pulse Width Modulation (PWM), Fuzzy Logic, DC Motor*

PENDAHULUAN

Dengan perkembangan teknologi dan informasi sebagian besar proses produksi pada industri-industri menggunakan mikrokontroller sebagai otak dalam proses kontrolnya. Proses kontrol pada mikrokontrollernya sudah menggunakan *Fuzzy logic* agar sistim yang digunakan dapat berfungsi sesuai dengan yang diinginkan. Demikian juga dengan motor DC banyak digunakan sebagai penggerak dalam berbagai peralatan, baik kecil maupun besar, lambat maupun cepat. Ia juga banyak dipakai karena dapat disesuaikan untuk secara ideal menerima pulsa digital untuk kendali kecepatan. Pemilihan cara pengendalian akan tergantung dari kebutuhan terhadap gerakan motor DC itu sendiri. Metode *Pulse Width Modulation* (PWM) adalah metode yang cukup efektif untuk mengendalikan kecepatan motor DC. PWM ini bekerja dengan cara membuat gelombang persegi yang memiliki perbandingan pulsa *high* terhadap pulsa *low* yang telah tertentu. Perbandingan pulsa *high* terhadap *low* ini akan menentukan jumlah daya yang diberikan ke motor DC.

Mikrokontroller sebagai sebuah "*one chip solution*" pada dasarnya adalah rangkaian terintregrasi (*Integrated Circuit*-IC) yang telah mengandung secara lengkap berbagai komponen pembentuk sebuah mikrokomputer. Berbeda dengan penggunaan mikroprosesor yang masih memerlukan komponen luar tambahan seperti RAM, ROM, *Timer*, dan sebagainya untuk sistem mikrokontroller, tambahan komponen diatas secara praktis hampir tidak dibutuhkan lagi. Hal ini disebabkan semua komponen penting tersebut telah ada bersama dengan sistem prosesor ke dalam IC tunggal mikrokontroller bersangkutan. Dewasa ini generasi AVR (*Alf and Vegard's Rich* prosessor), para desainer sistem elektronika telah diberi suatu teknologi yang memiliki kapabilitas yang amat maju, tetapi

dengan biaya ekonomis yang cukup minimal seperti mikrokontroller ATMega 8535.

Berdasarkan latar belakang ini maka dalam penelitian ini digunakan mikrokontroler ATMega 8535 sebagai pengendali kecepatan motor DC dengan menggunakan metode *Pulse Width Modulation* (PWM) yang diintegrasikan dengan *fuzzy logic*.

Perumusan Masalah

Bagaimana mengendalikan kecepatan motor DC dengan mengintegrasikan metode *Pulse Width Modulation* (PWM) dengan metode *fuzzy logic* berbasis mikrokontroller ATMega 8535?

Tujuan Penelitian

1. Mempelajari dan mengaplikasikan teknologi mikrokontroler ATMega 8535 untuk megendalikan kecepatan motor DC.
2. Mempelajari dan mengaplikasikan metode *Pulse Width Modulation* (PWM) untuk megendalikan kecepatan motor DC.
3. Mempelajari dan mengaplikasikan metode *fuzzy logic* untuk megendalikan kecepatan motor DC.
4. Mengintegrasikan metode *Pulse Width Modulation* (PWM) dengan metode *fuzzy logic* untuk mengendalikan kecepatan motor DC.

Manfaat Penelitian

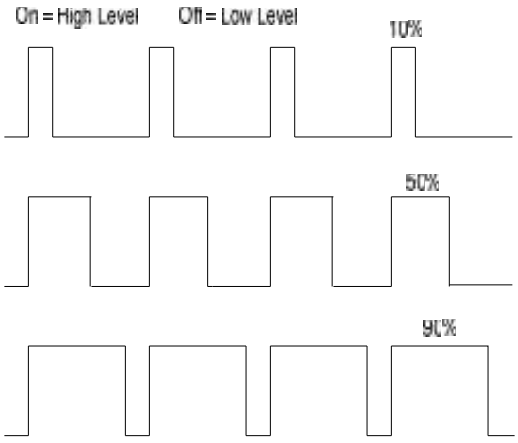
Dengan penelitian ini kita dapat mengaplikasikan metode *Pulse Width Modulation (PWM)* dan metode *Fuzzy Logic* untuk mengendalikan kecepatan motor DC berbasis mikrokontroller ATMega 8535 dalam bidang teknologi informasi dan komunikasi.

Tinjauan Teoritik
Motor DC

Motor DC banyak digunakan sebagai penggerak dalam berbagai peralatan, baik kecil maupun besar, lambat maupun cepat. Motor DC juga banyak dipakai karena dapat disesuaikan untuk secara ideal menerima pulsa digital untuk kendali kecepatan.

***Pulse Width Modulation* (PWM)**

Metode *Pulse Width Modulation* (PWM) adalah metode yang cukup efektif untuk mengendalikan kecepatan motor DC. PWM ini bekerja dengan cara membuat gelombang persegi yang memiliki perbandingan pulsa *high* terhadap pulsa *low* yang telah tertentu, biasanya diskalakan dari 0 hingga 100%. Gelombang persegi ini memiliki frekuensi tetap (biasanya max 10 KHz) namun lebar pulsa *high* dan *low* dalam 1 periode yang akan diatur. Perbandingan pulsa *high* terhadap *low* ini akan menentukan jumlah daya yang diberikan ke motor DC. Pada gambar kita dapat melihat bagaimana pengendalian dengan Pulse Width Modulation (PWM).



Gambar 1. Pengendalian dengan Pulse Width Modulation (PWM)

Mikrokontroler AVR ATMega 8535
Konsep Dasar AVR

Secara histories mikrokontroller seri AVR pertama kali diperkenalkan ke pasaran sekitar tahun 1997 oleh perusahaan Atmel. Berdasarkan arsitekturnya, AVR merupakan mikrokontroller dengan lebar bus data 8

bit dimana semua instruksi dikemas dalam kode 16-bit (16-*bit word*) dan sebagian besar instruksi dieksekusi dalam 1 siklus *clock*. Frekuensi kerja mikrokontroller AVR ini pada dasarnya sama dengan frekuensi *oscillator*. Dengan instruksi yang sangat variatif serta jumlah *register* serbaguna sebanyak 32 buah yang semuanya terhubung secara langsung ke ALU (*Arithmetic Logic Unit*), kecepatan operasi mikrokontroller AVR ini dapat mencapai 16 MIPS (enam belas juta instruksi per detik) yang merupakan sebuah kecepatan yang sangat tinggi untuk ukuran mikrokontroller 8 bit yang ada di pasaran sampai saat ini.

Untuk penyimpanan data, mikrokontroller AVR menyediakan dua jenis memori yang berbeda yaitu EEPROM (*Electrically Erasable Programmable Read Only Memory*) dan SRAM (*Static Random Access memory*). EEPROM umumnya digunakan untuk menyimpan data-data program yang bersifat permanen, sedangkan SRAM digunakan untuk menyimpan data variabel yang dimungkinkan berubah setiap saatnya.

Fuzzy Logic

Teori *Fuzzy* mulai dikembangkan pertama kali oleh Profesor Lotfi Zadeh pada ahun 1965 di University of Carlifornia, Berkeley. Teori *Fuzzy* dikembangkan untuk mengatasi teori *"Two Valued Logic"* (logika dua nilai) yang lazim dikenal sebagai logika *Boolean* yang membagi suatu keadaan hanya membagi dua kemungkinan saja misalnya hitam atau putih, benar atau salah dan tidak memberi kemungkinan nilai yang lain. Teori *"Two Valued Logic"* yang kemudian diimplementasikan dalam metode pemecahan masalah ternyata sangat efektif dan berhasil sebatas permasalahan yang dapat dideskripsikan secara tepat kuantitasnya.

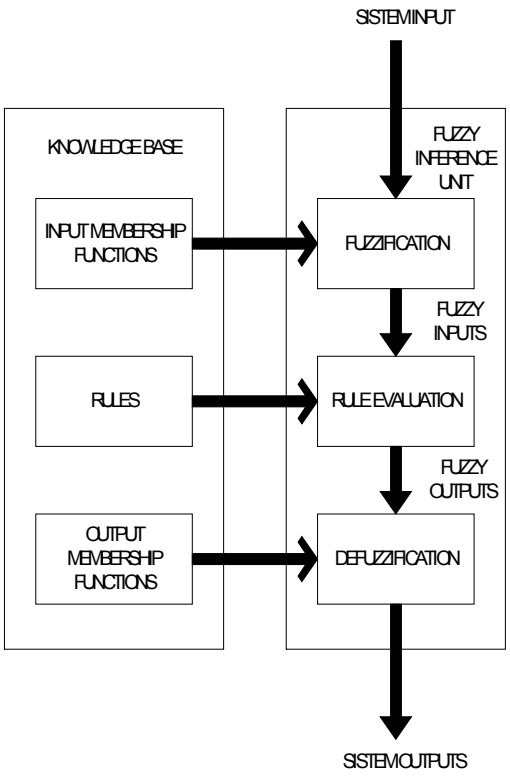
PetraFuz

PetraFuz adalah perangkat lunak yang digunakan untuk mendesain sistim berbasis *Fuzzy Logic*. Dengan adanya perangkat lunak ini, data *fuzzy* seperti *membership function* dan *rules* dapat dibuat dengan mudah.

Fuzzy Routine Engine

Fuzzy Routine Engine adalah suatu program kecil yang merupakan inti dari suatu sistim *fuzzy* menggunakan mikrokontroler.

Secara umum, struktur ini dibagi menjadi 2 blok yaitu: *fuzzy inference kernel* dan struktur data pada *knowlage base* seperti yang tampak pada gambar 2 Blok *fuzzy inference kernel* dibagi menjadi tiga bagian yaitu *fuzzification, rule evolution dan defuzzification*. Blok *knowlage base* berisi struktur data yang menunjang blok pertama, pembuatannya dibantu dengan *PetraFuz*.



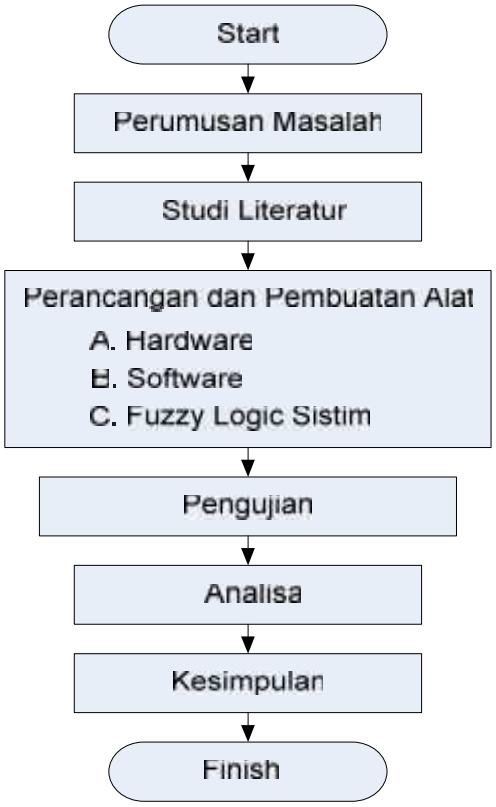
Gambar 2. Diagram Blok Fuzzy Routine Engine

Bagian *fuzzification* mengambil input dari input, lalu dibandingkan dengan input *membership function* yang ada pada *knowlage base* dan menyimpan hasilnya (*fuzzy input*) pada RAM. Setiap label pada *membership function* akan menghasilkan satu *fuzzy input.* Prosedur ini akan menghitung nilai kebenaran tiap input sistim pada setiap *membership function* dari input.

Bagian *rule evolution* memproses data *rules* pada *knowlage base* menggunakan hasil dari bagian *fuzzification (fuzzy input)* dan akan menghasilkan *fuzzy output* yang disimpam pada RAM.

Bagian *defuzzification* menggunakan *fuzzy output* dari bagian *rule evaluation* dan output *membership function* pada *knowledge base* untuk menghasilkan satu nilai output untuk setiap output sistim.

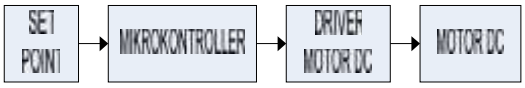
METODOLOGI PENELITIAN



Gambar 3. Metodologi Penelitian

Sistim Hardware

Dari diagram blok pada gambar 3 terlihat bahwa alat yang akan dirancang terdiri dari beberapa bagian:



Gambar 4. Diagram Blok Sistim

Bagian pertama yaitu input (set point) berupa potensiometer yang berfungsi untuk mengatur kecepatan putaran motor DC seperti yang tampak pada gambar. Potensiometer yang digunakan mempunyai tahanan sebesar 0 Ohm sampai dengan 10 Kohm. Besarnya tahanan ini yang akan mepengaruhi besarnya tegangan yang akan diinputkan ke mikrokontroler yaitu sebesar 0V sampai dengan 5 V. Dari 10 Kohm ini dibagi menjadi tiga kondisi untuk mengendalikan kecepatan putaran motor DC dengan keterangan sebagai berikut :

1. Untuk kondisi pertama yaitu sebesar 0 Ohm dengan besarnya tegangan input 0 V. Pada kondisi ini akan memberikan kondisi putaran motor DC tidak bergerak atau diam.
2. Untuk kondisi kedua besarnya tahanan 1 Ohm sampai dengan 5 Kohm dengan besarnya tegangan input 1 V sampai dengan 2,5 V. Pada kondisi ini putaran motor DC akan begerak lambat.
3. Untuk kondisi ketiga besarnya tahanan 5,1 Kohm sampai dengan 10 Kohm. Dengan besarnya tegangan input 2,6 V sampai dengan 5 V. Pada kondisi ini motor DC akan bergerak cepat.

Bagian kedua yaitu sistim minimum mikrokontroler ATMega 8535. Mikrokontroller ini mempuyai 4 buah port I/O yaitu port A, port B, port C dan Port D. Pada mikrokontroller ini sudah

terdapat pengkonversi sinyal analog ke sinyal digital secara internal dengan input yaitu pada port A0 sampai A7. Dimana pada sistim ini port A1 terhubung dengan set point (potensiometer) yang berfungsi untuk mengendalikan kecepatan putaran motor DC. Pada sistim ini port D5 difungsikan sebagai output yang dihubungkan dengan motor DC.

Bagian ketiga yaitu driver motor DC berfungsi untuk menjalankan motor DC dengan menggunakan IC L293D yang dapat menggerakkan motor DC.

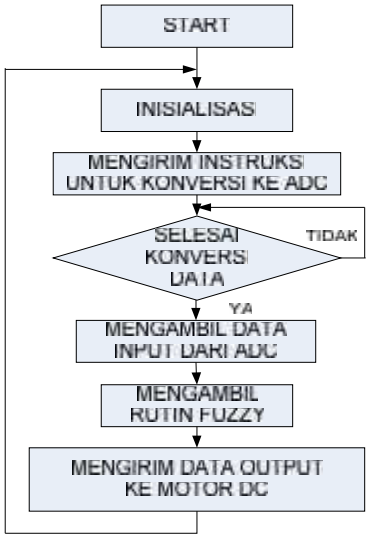
Bagian keempat yaitu motor DC. Motor DC yang digunakan adalah motor DC 12 V.

Sistim Perangkat Lunak

Perangkat lunak yang dirancang untuk sistim ini adalah perangkat lunak untuk menjalan sistim mikrokontroler secara keseluruhan, *software* ini dirancang dengan menggunakan sofware *Bascom* dan software PetraFuz untuk mendesain logika *Fuzzy*.

Perangkat Lunak Sistim Mikrokontroler ATMega 8535

Seperti yang telah dijelaskan diatas perangkat lunak ini dirancang dengan menggunakan *Bascom*.



Gambar 5. Flowchart pengontrol kecepatan putaran motor DC

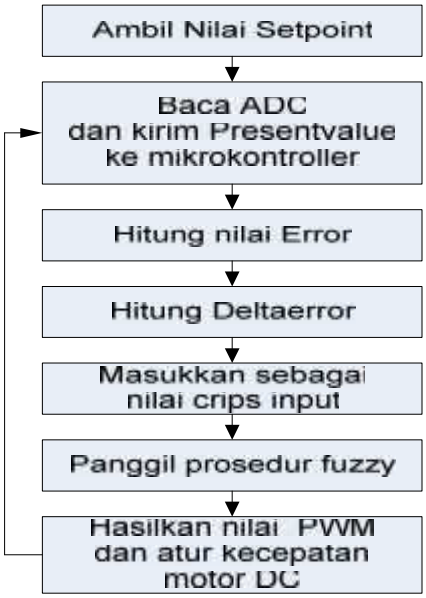
Dari gambar 5. *flowchart* yang tampak pada gambar dapat kita jelaskan sebagai berikut :

Pertama-tama mikrokontroller akan menginisialisasi input potensiometer pada port A1 dan output berupa motor DC pada port D5. Kemudian mikrokontroller akan memberikan innstruksi untuk membaca data analog dari input yang kemudian mengkonversikannya ke data digital. Setelah itu mikrokontroller akan memanggil rutin *fuzzy logic* yang telah dibuat dengan bantuan PetraFuz. Output dari fuzzy logic ini yang nantinya berfungsi untuk mengontrol kecepatan putaran motor DC dan output dari *fuzzy logic* dimasukan sebagai nilai *Pulse Width Modulation* (PWM) untuk mengendalikan kecepatan putaran motor DC.

PetraFuz

PetraFuz adalah perangkat lunak yang digunakan untuk mendesain sistim berbasis *Fuzzy Logic*. Dengan adanya perangkat lunak ini, data *fuzzy* seperti *membership function* dan *rules* dapat dibuat dengan mudah.

Fuzzy Routine Engine seperti pada gambar adalah suatu program kecil yang merupakan inti dari suatu sistim *fuzzy* menggunakan mikrokontroler. Secara umum, struktur ini dibagi menjadi 2 blok yaitu: *fuzzy inference kernel* dan struktur data pada *knowlage base*. Blok *fuzzy inference kernel* dibagi menjadi tiga bagian yaitu *fuzzification, rule evolution dan defuzzification*. Blok *knowlage base* berisi struktur data yang menunjang blok pertama, pembuatannya dibantu dengan PetraFuz.



Gambar 6. Flowchart fuzzy logic dengan PetraFuz

Dari *flowchart* pada gambar 6. diatas dapat kita jelaskan sebagai berikut:

Bagian *fuzzification* mengambil input dari set point lalu dibandingkan dengan input *membership function* yang ada pada *knowlage base* dan menyimpan hasilnya (*fuzzy input*) pada RAM. Setiap label pada *membership function* akan menghasilkan satu *fuzzy input.*

Bagian *rule evolution* memproses data *rules* pada *knowlage base* menggunakan hasil dari bagian *fuzzification (fuzzy input)* dan akan menghasilkan *fuzzy output* yang disimpam pada RAM.

Fungsi keanggotaan input yang direncanakan untuk input *error* dan *deltaerror* (kesalahan dan perubahan kesalahan) terdiri dari 5 label yaitu NB (*Negativ Big*), NS (*Negativ Small*), Z (*Zero*), PS (*Positiv Small*) dan PB (*Positiv Big*).
Dimana :

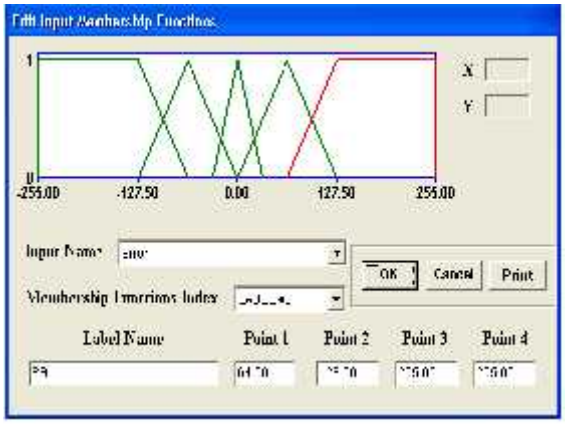
Error (n) = setpoint (n) – pointvalue (n)
Deltaerror (n) = error (n) – error (n-1)

Fungsi keanggotaan output yang direncanakan terdiri dari 5 label yaitu NB (*Negativ Big*), NS (*Negativ Small*), Z (*Zero*), PS (*Positiv Small*) dan PB (*Positiv Big*). Output

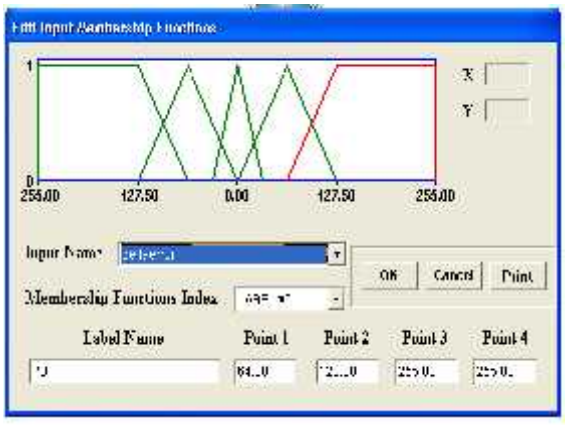
Output = Output(n) – Output(n-1)

Bagian *defuzzification* menggunakan *fuzzy output* dari bagian *rule evaluation* dan output *membership function* pada *knowlage base* untuk menghasilkan satu nilai output untuk setiap output sistim sebagai nilai *Pulse Width Modulation* (PWM) untuk mengendalikan kecepatan putaran motor DC.

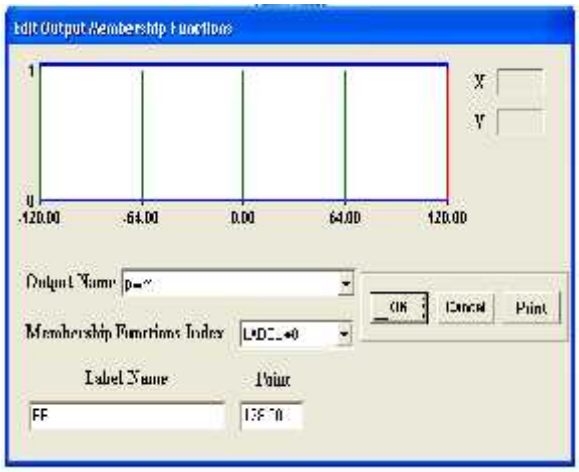
Adapun fungsi keanggotaaan membership function sistim yang dirancang dengan menggunakan PetraFuz pada gambar 7.



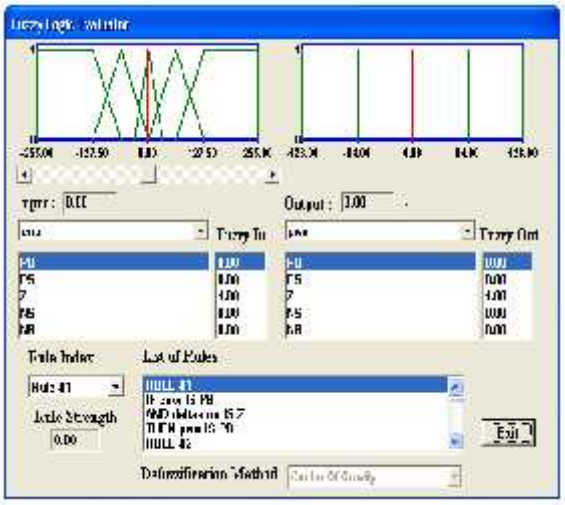
(a) *Membership Function untuk Error*



(b) Membership Function untuk Delta Error



(c) Membership Function untuk Output



(d) Fuzzy Logic Evaluator

Gambar 7(a)(b)(c)(d) Membership Function Input-Output PetraFuzz

Adapun rule yang dirancangkan sebanyak 15 rule seperti yang tampak pada table 1,2,3 dan 4 dalam hal ini digunakan ***If.....AND.....Then.....***

Tabel 1. Rule-rule *Fuzzy Control* yang dirancang

Rule No.	Input		Output (Result)
	Error	DeltaError	
1	PositivBig (PB)	Zero (Z)	PositivBig (PB)
2	Zero (Z)	NegativBig (NB)	NegativBig (NB)
3	NegativBig (NB)	Zero (Z)	NegativBig (NB)
4	Zero (Z)	PositivBig (PB)	PositivBig (PB)
5	PositivSmall (PS)	Zero (Z)	PositivSmall (PS)
6	Zero (Z)	NegativSmall (NS)	NegativSmall (NS)
7	NegativSmall (NS)	Zero (Z)	NegativSmall (NS)
8	Zero (Z)	PositivSmall (PS)	PositivSmall (PS)
9	Zero (Z)	Zero (Z)	Zero (Z)
10	NegativSmall (NS)	NegativSmall (NS)	NegativSmall (NS)
11	PositivSmall (PS)	PositivSmall (PS)	PositivSmall (PS)
12	NegativBig (NB)	NegativBig (NB)	NegativBig (NB)
13	NegativBig (NB)	NegativSmall (NS)	NegativBig (NB)
14	PositivBig (PB)	PositivBig (PB)	PositivBig (PB)
15	PositivBig (PB)	PositivSmall (PS)	PositivBig (PB)

Tabel 2. Membership FunctionUntuk input Error

Crisp Input Error	Numerical Range
PositivBig (PB)	64 - 255
PositivSmall (PS)	0 - 128
Zero (Z)	-32 - 32
NegativSmall (NS)	-128 - 0
NegativBig (NB)	-64 - 255

Tabel 3. Membership FunctionUntuk input Delta Error

Crisp Input Delta Error	Numerical Range
PositivBig (PB)	64 - 255
PositivSmall (PS)	0 - 128
Zero (Z)	-32 - 32
NegativSmall (NS)	-128 - 0
NegativBig (NB)	-64 - 255

Tabel 4. Membership FunctionUntuk Output

Crisp Output	Numerical Range
PositivBig (PB)	128
PositivSmall (PS)	64
Zero (Z)	0
NegativSmall (NS)	-64
NegativBig (NB)	-128

HASIL DAN PEMBAHASAN

Hasil *Rule-rule Fuzzy Control* dalam bentuk database yang dihasilkan dari *software* PetraFuz yang terdapat dalam mikrokontroller berdasarkan tabel adalah sebagai berikut:

```
NUMINP      EQU 02H
NUMOUT      EQU 01H
            .CODE
            ORG 4100H
INPUT_MFS   ;Input Membership Functions
IN0MF       ;error
            DB 0A0H,008H,0FFH,0FFH     ;PB
            DB 080H,008H,0A0H,008H     ;PS
            DB 070H,010H,080H,010H     ;Z
            DB 040H,008H,060H,008H     ;NS
            DB 000H,0FFH,040H,008H     ;NB
            DB 000H,000H,000H,000H     ;
            DB 000H,000H,000H,000H     ;
            DB 000H,000H,000H,000H     ;
IN1MF       ;deltaerror
            DB 0A0H,008H,0FFH,0FFH     ;PB
            DB 080H,008H,0A0H,008H     ;PS
            DB 070H,010H,080H,010H     ;Z
            DB 040H,008H,060H,008H     ;NS
            DB 000H,0FFH,040H,008H     ;NB
            DB 000H,000H,000H,000H     ;
            DB 000H,000H,000H,000H     ;
            DB 000H,000H,000H,000H     ;
               .CODE
               ORG 41A0H
OUTPUT_MFS  ;Output Membership Functions
OUT0MF      ;output
            DB 0D5H                    ;PB
            DB 0AAH                    ;PS
            DB 080H                    ;Z
            DB 055H                    ;NS
            DB 02BH                    ;NB
            DB 000H                    ;
            DB 000H                    ;
            DB 000H                    ;
               .CODE
               ORG 41B8H
RULE_START  DB 000H     ;IF error IS PB
          DB 00AH               ;AND deltaerror IS Z
          DB 080H               ;THEN output IS PB
          DB 002H               ;IF error IS Z
            DB 00CH                 ;AND deltaerror IS NB
          DB 084H               ;THEN output IS NB
```

```
        DB 004H             ;IF error IS NB
        DB 00AH             ;AND deltaerror IS Z
        DB 084H             ;THEN output IS NB
        DB 002H             ;IF error IS Z
          DB 008H               ;AND deltaerror IS PB
        DB 080H             ;THEN output IS PB
        DB 001H             ;IF error IS PS
        DB 00AH             ;AND deltaerror IS Z
        DB 081H             ;THEN output IS PS
        DB 002H             ;IF error IS Z
          DB 00BH               ;AND deltaerror IS NS
        DB 083H             ;THEN output IS NS
        DB 003H             ;IF error IS NS
        DB 00AH             ;AND deltaerror IS Z
        DB 083H             ;THEN output IS NS
        DB 002H             ;IF error IS Z
          DB 009H               ;AND deltaerror IS PS
        DB 081H             ;THEN output IS PS
        DB 002H             ;IF error IS Z
        DB 00AH             ;AND deltaerror IS Z
        DB 082H             ;THEN output IS Z
        DB 003H             ;IF error IS NS
          DB 00BH               ;AND deltaerror IS NS
        DB 083H             ;THEN output IS NS
        DB 001H             ;IF error IS PS
          DB 009H               ;AND deltaerror IS PS
        DB 081H             ;THEN output IS PS
        DB 004H             ;IF error IS NB
          DB 00CH               ;AND deltaerror IS NB
        DB 084H             ;THEN output IS NB
        DB 004H             ;IF error IS NB
          DB 00BH               ;AND deltaerror IS NS
        DB 084H             ;THEN output IS NB
        DB 000H             ;IF error IS PB
          DB 008H               ;AND deltaerror IS PB
        DB 000H             ;AND error IS PB
        DB 080H             ;THEN output IS PB
        DB 000H             ;IF error IS PB
          DB 009H               ;AND deltaerror IS PS
        DB 080H            ;THEN output IS PB
END_OF_RULE DB 0FFH
```

PEMBAHASAN

Hardware

Input (set point) berupa potensiometer yang berfungsi untuk mengatur kecepatan putaran motor DC. Besarnya tahanan ini yang akan mepengaruhi besarnya tegangan yang akan diinputkan ke mikrokontroler yaitu sebesar 0V sampai dengan 5 V. Untuk kondisi sebesar 0 Ohm dengan besarnya tegangan input 0 V memberikan kondisi putaran motor DC tidak bergerak atau diam. Untuk kondisi yang besarnya tahanan 1 Ohm sampai dengan 5 Kohm dengan besarnya tegangan input 1 V sampai dengan 2,5 V kecepatan putaran motor DC akan begerak lambat. Untuk kondisi besarnya tahanan 5,1 Kohm sampai dengan 10 Kohm dengan besarnya tegangan input 2,6 V sampai dengan 5 V kecepatan putaran motor DC akan begerak cepat. Kondisi dari input-input ini dimasukkan pada bagian pengkonveri analog ke digital (ADC) yang terdapat pada mikrokontroller. Dari bagian ADC ini kemudian data yang dihasilkan dimasukan ke bagian *Fuzzy Routine Engine* yang terdapat pada RAM. Output dari *Fuzzy Routine Engine* menghasilkan nilai *Pulse Width Modulation* (PWM) untuk mengendalikan kecepatan putaran motor DC.

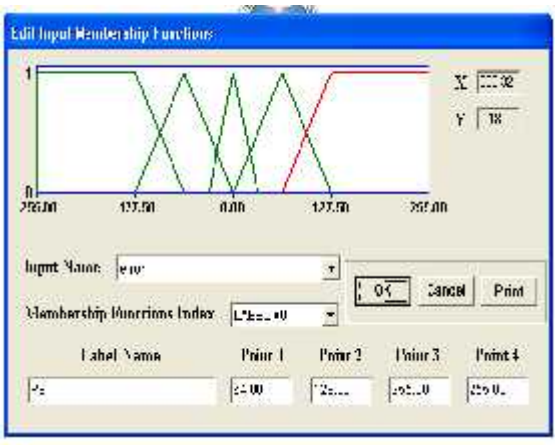
Software

Fuzzy Routine Engine adalah suatu program kecil yang merupakan inti dari suatu sistim *fuzzy* menggunakan mikrokontroler yang dibagi menjadi 2 blok yaitu: *fuzzy inference kernel* dan struktur data pada *knowlage base.* Blok *fuzzy inference kernel* dibagi menjadi tiga bagian yaitu *fuzzification, rule evolution dan defuzzification*. Blok *knowlage base* berisi struktur data yang menunjang blok pertama, pembuatannya dibantu dengan PetraFuz. Bagian *fuzzification* mengambil input dari set point dari potensiometer lalu dibandingkan dengan input *membership function* yang ada pada *knowlage base* dan menyimpan hasilnya (*fuzzy input*) pada RAM. Setiap label pada

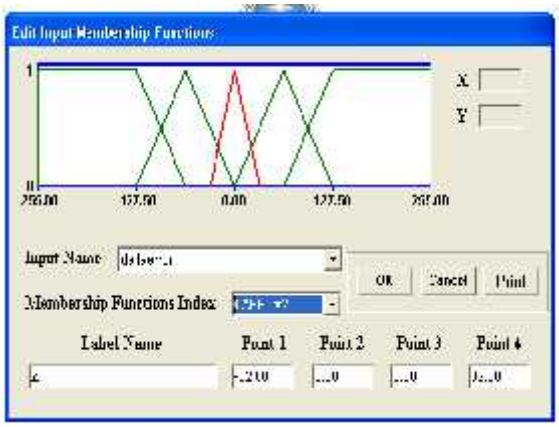
membership function akan menghasilkan satu *fuzzy input.* Fungsi keanggotaan input yang direncanakan untuk input *error* dan *deltaerror* (kesalahan dan perubahan kesalahan) terdiri dari 5 label yaitu NB (*Negativ Big*), NS (*Negativ Small*), Z (*Zero*), PS (*Positiv Small*) dan PB (*Positiv Big*). Bagian *rule evolution* memproses data *rules* pada *knowlage base* menggunakan hasil dari bagian *fuzzification (fuzzy input)* dan akan menghasilkan *fuzzy output* yang disimpam pada RAM. Bagian *defuzzification* menggunakan *fuzzy output* dari bagian *rule evaluation* dan output *membership function* pada *knowlage base* untuk menghasilkan satu nilai output untuk setiap output sistim sebagai nilai *Pulse Width Modulation* (PWM) untuk mengendalikan kecepatan putaran motor DC.

Berikut ini akan dijelaskan proses *fuzzy logic* berdasarkan *membership function* input dan output untuk masing-masing rule.

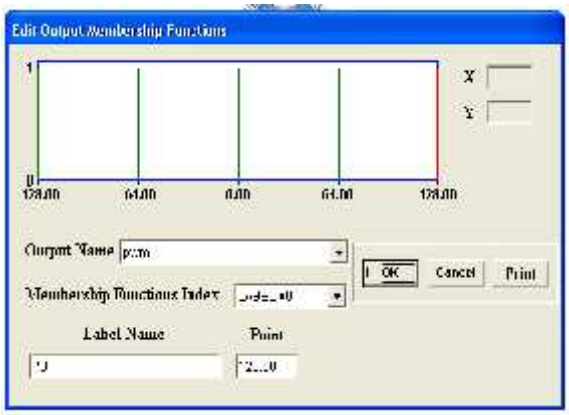
Rule 1; If Error is PositivBig and Delta Error is Zero then output is PositivBig dimana kondisi kecepatan putaran motor DC akan cepat.



(a) Membership Function untuk input Error



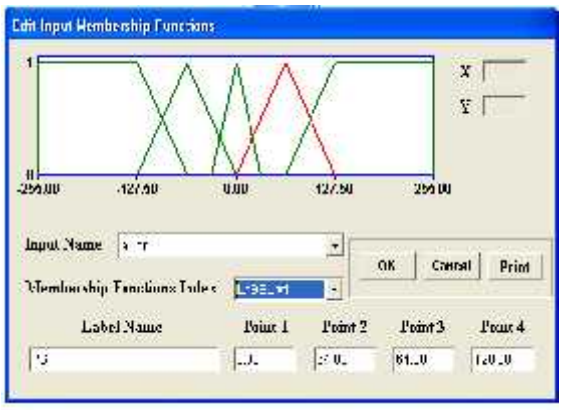
(b) Membership Function untuk input Delta Error



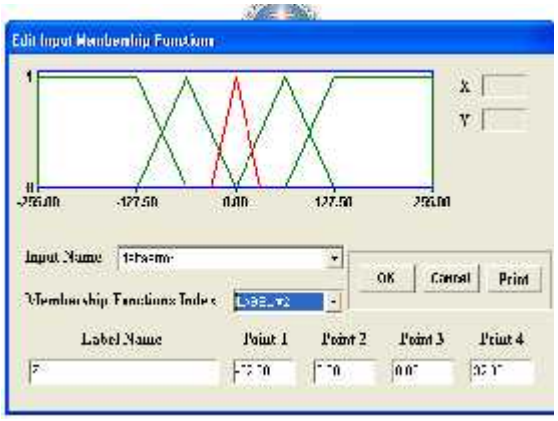
(c) Membership Function untuk output

Gambar 8(a)(b)(c) Membership Function untuk input dan output Rule 1

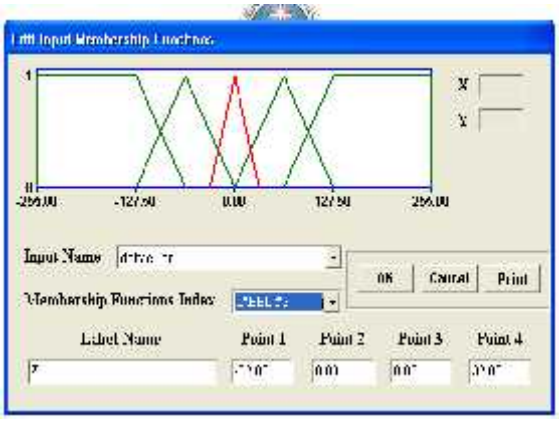
Rule 5; If Error is PositivSmall and Delta Error is Zero then output is PositivSmall dimana kondisi kecepatan putaran motor DC akan lambat.



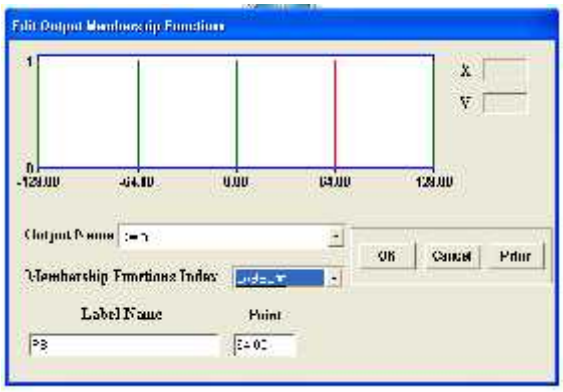
(*a) Membership Function untuk input Error*



(b) Membership Function untuk input Delta Error

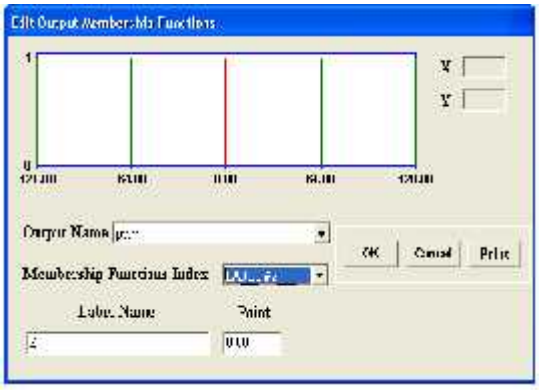


(b) Membership Function untuk input Delta Error



(c) Membership Function untuk output

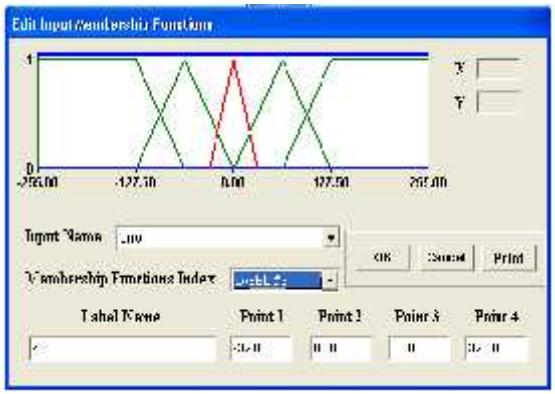
Gambar 9(a)(b)(c) Membership Function untuk input dan output Rule 5



(c) Membership Function untuk output

Gambar 10(a)(b)(c) Membership Function untuk input dan output Rule 9

Rule 9; If Error is Zero and Delta Error is Zero then output is Zero dimana kondisi kecepatan putaran motor DC akan diam atau tidak bergerak.



(a) Membership Function untuk input Error

Dari hasil pengujian simulasi *fuzzy logic* dengan mengggunakan PetraFuz dapat kita lihat bahwa desain *Membership Function* Sistim dan Rule Fuzzy Control yang dirancang menghasilkan output yang sesuai dengan yang kita inginkan. Output dari fuzzy logic ini yang nantinya berfungsi untuk mengontrol kecepatan putaran motor DC dan output dari *fuzzy logic* dimasukan sebagai nilai *Pulse Width Modulation* (PWM) untuk mengendalikan kecepatan motor DC.

KESIMPULAN

1. Dengan metode *Pulse Width Modulation* (PWM) kita dapat mengendalikan kecepatan motor DC.
2. Dengan metode *fuzzy logic* kita mengendalikan kecepatan motor DC.
3. Dengan mengintegrasikan metode *Pulse Width Modulation* (PWM) dengan metode *fuzzy logic* kita dapat mengendalikan kecepatan putaran motor DC.
4. Dengan mengunakan PetraFuz desain *Fuzzy Logic* berbasis mikrokontroller dapat dirancang dan dibuat dengan mudah

DAFTAR PUSTAKA

Ibrahim, Ahmad M.(2003).”Fuzzy Logic for Embedded System Application.” New York: Elsevier Science.

Kazuo Tanaka, Hua O. Wang (2001).”Fuzzy Control systems Design and Analysis.” New York: Jhon Wiley and Son Inc.

Ross, Timothy J. (1997).”Fuzzy Logic With Engineering Applications.” New York: McGraw-Hill Int.

Turner, Rufus P., Rutherford, Brinton L. (1993). ”133 Rangkaian Elektronika.” Jakarta: Elex Media Komputindo.

Wiyono, Didik (2007).”Panduan Praktis Mikrokontroller Keluarga AVR Menggunakan DT-Combo AVR-51 Starter Kit dan DT-Combo AVR Exercise Kit.” Surabaya: Innovative Electronics.