BAB 2

LANDASAN TEORI

2.1 Microcontroller AT89C52

Komputer secara umum memiliki 3 komponen utama, yakni :

- Central Processing Unit (CPU)
- Memori untuk program dan data
- Sistem Input dan Output (I/O)

CPU mengontrol semua alur informasi antara semua komponen yang berada dalam komputer. CPU juga memproses data dengan melakukan perhitungan digital. Kebanyakan semua proses dilakukan oleh Arithmetic Logic Unit (ALU) yang berada di dalam CPU tersebut.

Microcontroller adalah sebuah komputer yang lengkap yang di produksi ke dalam sebuah chip. Subsistem *I/O* dan memori yang terkandung di dalam sebuah microcontroller sudah dirancang sedemikian rupa agar dapat diterapkan pada hardware dan fungsi-fungsi kontrol pada aplikasi.

Fasilitas-fasilitas yang terdapat pada microcontroller adalah seperti port serial, port input dan output paralel, timer, counter, interrupt control, Analog to digital converter, random access memory, dan read only memory.

Microcontroller AT89C52 merupakan adalah sebuah mikrokomputer 8 bit yang berdaya rendah namun mempunyai kinerja yang tinggi dilengkapi dengan 8K bytes Flash programmable and erasable read only memory (PEROM). Alat ini

dibuat dengan menggunakan teknologi high density nonvolatile memory ATMEL dan kompatibel dengan standar industri 80C51 dan 80C52 pada set instruksi dan susunan pinnya. On-chip Flash tersebut dapat di program ulang sebanyak 1000 kali. Dengan menggabungkan 8-bit CPU dengan Flash pada sebuah chip monolitik, AT89C52 merupakan sebuah mikrokomputer handal yang menyediakan fleksibilitas yang tinggi serta biaya yang efektif untuk berbagai aplikasi pengendalian.

AT89C52 menyediakan keunggulan-keunggulan sebagai berikut :

- Kompatibel dengan produk-produk MCS-51
- memori Flash sebesar 8K bytes dengan kemampuan baca tulis ulang sebanyak 1000 kali.
- Fully Static Operation : 0 Hz – 24 MHz
- Tiga level pengunci an memori progam
- RAM sebesar 256 bytes.
- 32 buah saluran I/O
- 3 buah 16 bit timer atau counter
- arsitektur intterupt 2 level 6 vektor
- Port serial full duplex
- On chip oscillator
- Clock osilator internal

Sebagai tambahan, AT89C52 dirancang dengan logika static untuk operasi dibawah frekuensi nol dan mendukung 2 software pilihan untuk mode power saving. Mode yang pertama yaitu Idle Mode yang akan menghentikan CPU

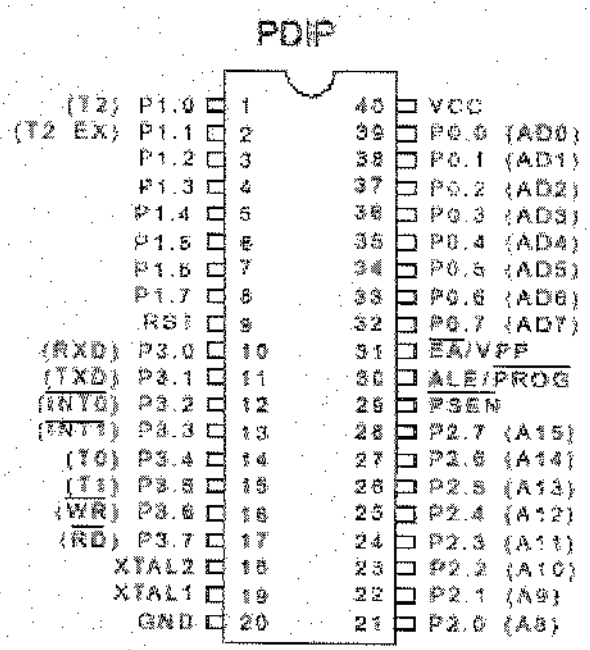
sementara membiarkan RAM, timer/counter, port serial, dan system interrupt tetap berjalan. Mode kedua adalah Power Down Mode yang akan menyimpan isi RAM tetapi menghentikan osilator, serta mematikan semua fungsi-fungsi chip yang lainnya sampai reset hardware berikutnya.

Perbedaan AT89C51 dan AT89C52 dapat di lihat pada tabel dibawah ini.

Tabel 2.1 Perbedaan AT89C51 dan AT89C52

	89C51	89C52
1	Memory Rom 8kb	Memory Rom lebih besar yaitu 8kb
2	Ram 2 x 128kb	Ram 3 x 128kb
3	Memiliki timer 0 dan timer 1	Memiliki timer 0, timer 1, dan timer 2
4	Eksternal interupt 0 dan 1	Eksternal interupt 0, 1, dan 2

2.1.1 Konfigurasi Pin AT89C52



Gambar 2.1 Microcontroller AT89C52

2.1.2 Penjelasan Pin pada AT89C52 :

- VCC

  Pin ini dihubungkan pada Tegangan berupa 5 volt.

- GND

  Pin ini dihubungkan ke Ground.

- Port 0

Port 0 adalah port I/O 8-bit dua arah drain terbuka. Sebagai port output, setiap pin dapat memasukkan 8 buah input TTL. Ketika nilai 1 dituliskan pada pin port 0 maka pin tersebut dapat digunakan sebagai input dengan impedansi tinggi.

Port 0 dapat di rancang sebagai bus address/data multipleks yang low-order selama mengakses memori data dan program eksternal. Pada mode ini, P0 mempunyai internal pullup.

Port 0 menerima kode bytes selama pemrograman Flash dan mengeluarkan kode bytes selama verifikasi program. Eksternal pullup diperlukan selama proses verifikasi program.

- Port 1

Port 1 adalah port I/O 8-bit dua arah dengan internal pullup. Bufer output port 1 dapat menarik/memberikan 4 buah input TTL. Ketika nilai 1 dituliskan pada pin port 1, pin itu akan di pull secara kuat oleh internal pullup dan dapat digunakan sebagai input. Sebagai tambahan, P1.0 dan P1.1 dapat dikonfigurasikan sebagai timer/counter sesuai tabel ini :

Tabel 2.2 Pin Port P1

Port Pin	Alternate Functions
P1.0	T2 (external count input to Timer/Counter 2), clock-out
P1.1	T2EX (Timer/Counter 2 capture/reload trigger and direction control)

- Port 2

Port 2 adalah port I/O 8-bit dua arah dengan internal pullup. Bufer output port 2 dapat menarik/memberikan 4 buah input TTL. Ketika nilai 1 dituliskan pada pin port 2, pin itu akan di pull secara kuat oleh internal pullup dan dapat digunakan sebagai input.

Port 2 memancarkan byte alamat nilai high selama memanggil dari memori program eksternaldan selama mengakses ke memori data yang menggunakan pengalamatan 16 bit (MOVX @DPTR). Pada aplikasi ini, port 2 menggunakan pullup internal yang kuat ketika memancarkan nilai 1 selama mengakses ke memori data eksternal yang menggunakan pengalamatan 8 bit (MOVX @ RI), port 2 memancarkan isi dari P2 Special Function Register. Port 2 juga menerima bit-bit alamat nilai high dan beberapa sinyal kontrol selama proses verifikasi dan pemograman Flash.

- Port 3

Port 3 adalah port I/O 8-bit dua arah dengan internal pullup. Bufer output port 3 dapat menarik/memberikan 4 buah input TTL. Ketika nilai 1 dituliskan pada pin port 3, pin itu akan di pull secara kuat oleh internal pullup dan dapat digunakan sebagai input.

Port 3 juga menyediakan fungsi-fungsi dari beberapa kelebihan spesialnya yang beragam seperti pada AT89C51, dapat di lihat pada tabel di bawah ini :

Tabel 2.3 Pin Port P3

Port Pin	Alternate Functions
P3.0	RXD (serial input port)
P3.1	TXD (serial output port)
P3.2	$\overline{INT0}$ (external interrupt 0)
P3.3	$\overline{INT1}$ (external interrupt 1)
P3.4	T0 (timer 0 external input)
P3.5	T1 (timer 1 external input)
P3.6	$\overline{WR}$ (external data memory write strobe)
P3.7	$\overline{RD}$ (external data memory read strobe)

- RST

Input untuk reset. Nilai high pada mesin ini untuk 2 siklus mesin sementara osilator sedang melakukan reset pada alat.

- ALE/PROG

Address Latch Enable adalah pulsa output untuk menahan byte low pada alamat selama mengakses memori eksternal. Pin ini juga merupakan input pulsa program (PROG) selama pemrograman Flash. Pada operasi normal, ALE dipancarkan pada rata-rata konstan 1/6 dari frekuensi osilator dan dapat digunakan untuk tujuan keperluan timing/clocking eksternal.

- PSEN

Program Store Enable adalah untuk memberi pulsa ke memori luar didalam proses pengambilan data dari ROM/EPROM.

- EA/Vpp

External Access Enable, EA harus dihubungkan dengan GND dengan tujuan untuk membolehkan alat memanggil kode dari lokasi memori program eksternal berawal di 0000H sampai FFFFH.

EA harus dihubungkan dengan VCC untuk pengeksekusian program internal. Pin ini juga menerima tegangan programming enable 12 volt (Vpp) selama pemrograman Flash ketika 12 volt programming dipilih.

- XTAL1

Input ke penguat osilator pembalik dan input ke rangkaian operasi clock internal.

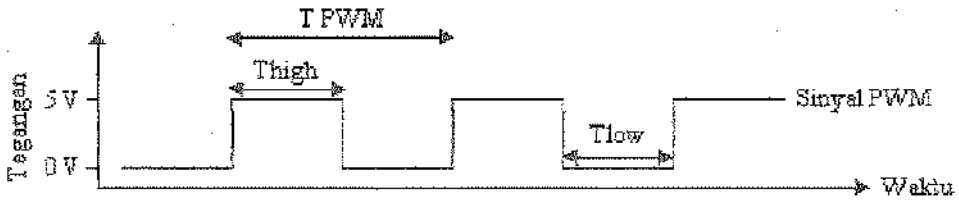
- XTAL2

Output dari penguat oslilator pembalik.

2.2 PWM (Pulse Width Modulation)

Pada dasarnya PWM adalah proses pengaturan kecepatan secara digital yang digunakan pada motor DC. Nilai tegangan yang diberikan pada motor DC tidaklah berubah (konstan), yang diatur adalah rasio waktu pemberian tegangan kepada motor tersebut. Maksudnya, memberikan pulsa-pulsa yang mempunyai lebar waktu on dan lebar waktu off membuat motor DC berputar lebih cepat. Waktu periode terjadi apabila siklus Ton dan Toff terjadi pada frekuensi yang sama pada kecepatan yang berbeda.

Dibawah ini adalah timing diagram PWM :



gambar 2.2 Timing Diagram PWM

2.3 MOTOR DC

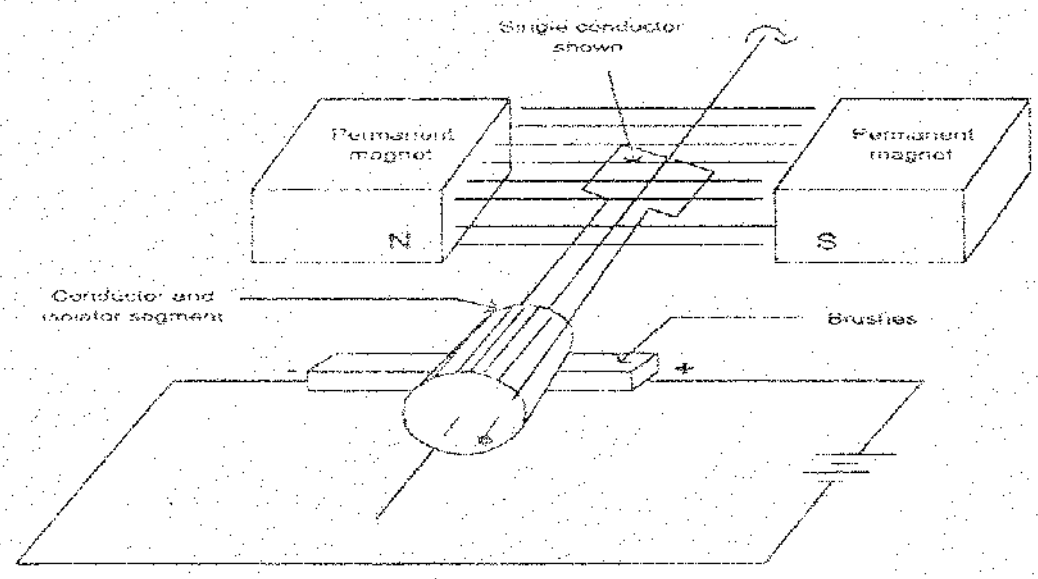
Motor DC adalah alat yang mengubah pulsa listrik menjadi gerak, mempunyai prinsip dasar yang sama dengan motor stepper namun gerakannya bersifat continue atau berkelanjutan. Motor DC dibagi menjadi 2 jenis yaitu ; Motor DC dengan sikat (mekanis komutasi), yaitu motor yang memiliki sikat karbon berfungsi sebagai pengubah arus pada kumparan sedemikian rupa sehingga arah tenaga putaran motor akan selalu sama. Motor DC tanpa sikat , menggunakan semi konduktor untuk merubah maupun membalik arus sehingga layaknya pulsa yang menggerakkan motor tersebut. Biasa digunakan pada sistem servo, karena mempunyai efisiensi tinggi, umur pemakaian lama, tingkat kebisingan suara listrik rendah, karena putarannya halus seperti *stepper* namun putarannya terus menerus tanpa adanya step.

Secara umum motor terdiri dari bagian-bagian :

- *Armature* : menghubungkan sumber energi listrik dengan motor. *Armature* biasanya berbentuk silinder, yang terdiri dari beberapa lilitan konduktor. Armature yang berputar biasanya disebut rotor.

- *Magnet* : *fluks medan magnet* dari magnet akan dipotong oleh rotor. Bila magnet ini berupa lilitan maka arus yang akan menghasilkan medan magnet biasanya sama dengan arus untuk *armature* / *rotor*. Magnet ini biasanya memiliki posisi yang tetap sehingga biasa disebut *stator*.
- *Brushes* / sikat : merupakan penghubung antara *armature* dengan sumber tegangan.
- *Commutator* : menghubungkan lilitan pada *armature* dan *brushes*.

Komponen dasar dari sebuah motor DC sederhana digambarkan dibawah ini :



Gambar 2.3 Komponen Motor DC

Dari gambar di atas, sebuah *armature* dengan satu atau lebih lilitan berbentuk sayap yang saling berpisah dan berputar. Setiap lilitan berujung pada sebuah cincin terpisah (*Commutator*), dimana energi (listrik) dialirkan menuju commutator melalui sikat (*brushes*). Diantara *commutator* terdapat *isolator*, sehingga cincin ini berlaku sebagai saklar *Double-Pole Double-Throw*. Pada saat

armature berputar, *commutator* akan men-*switch* arus terus menerus, sehingga medan magnet *armature* akan bernilai tetap. Putaran armature timbul karena medan magnet armature melawan medan elektromagnetik tetap yang disebut *field.* Pada motor dengan magnet tetap (*Permanent Magnet*), field ditimbulkan oleh magnet tetap.

2.4 Liquid Crystal Display (LCD)

LCD merupakan suatu komponen *opto electronic* yang berfungsi sebagai alat penampil elektronik yang mirip dengan monitor dan diaktifkan dengan molekul kristal cair (*liquid crystal*) yang merupakan unsur utamanya. Kristal seharusnya berbentuk padat tetapi Padat dan cair merupakan dua sifat benda yang berbeda. Molekul-molekul benda padat tersebar secara teratur dan posisinya tidak berubah-ubah, sedangkan molekul-molekul zat cair letak dan posisinya tidak teratur karena dapat bergerak acak ke segala arah. Pada tahun 1888 seorang ahli botani, Friedrich Reinitzer, menemukan fase yang berada di tengah-tengah antara fase padat dan cair. Fase ini memiliki sifat-sifat padat dan cair secara bersama-sama. Molekul- molekulnya memiliki arah yang sama seperti sifat padat, tetapi molekul-molekul itu dapat bergerak bebas seperti pada cairan. Fase kristal cair ini berada lebih dekat dengan fase cair karena dengan sedikit penambahan temperatur (pemanasan), fasenya langsung berubah menjadi cair. Sifat ini menunjukkan sensitivitas yang tinggi terhadap temperatur. Sifat inilah yang menjadi dasar utama pemanfaatan kristal cair dalam teknologi.

Pada umumnya LCD memiliki bidang datar, karena tampilan pada layar LCD memiliki batas sudut pandang yang optimum untuk dapat dilihat dengan jelas.

Jenis LCD yang dipakai memiliki jumlah kombinasi 2 baris dan 16 karakter. LCD ini terkenal dengan format 14 pin tapi dengan tambahan 2 pin untuk back light.

Tabel 2.4 Fungsi masing-masing PIN

PIN	Simbol	Deskripsi
1	0V	Power Supply
2	+5V	
3	Vee	LCD contrast control input
4	RS	Pengaturan Input register
5	R/W	Read/Write
6	E	Enable signal
7-14	D0-D7	8 bit bi-directional µp data bus
15	Lamp-	Back light supply, max. 90mA
16	Lamp+	

2.5 SENSOR

Sensor yang dipakai adalah sensor encoder atau sebagai umpan balik. Sensor Penyandi (Encoder) digunakan untuk mengubah gerakan linear atau

putaran menjadi sinyal digital, dimana sensor putaran memonitor gerakan putar dari suatu alat. Sensor ini biasanya terdiri dari 2 lapis jenis penyandi, yaitu;

1. Penyandi rotari tambahan / *Incremental encoder* (yang mentransmisikan jumlah tertentu dari pulsa untuk masing-masing putaran) yang akan membangkitkan gelombang kotak pada objek yang diputar.
2. Penyandi absolut / *Encoder Absolute* (yang memperlengkapi kode binary tertentu untuk masing-masing posisi sudut) mempunyai cara kerja sang sama dengan perkecualian, lebih banyak atau lebih rapat pulsa gelombang kotak yang dihasilkan sehingga membentuk suatu pengkodean dalam susunan tertentu.

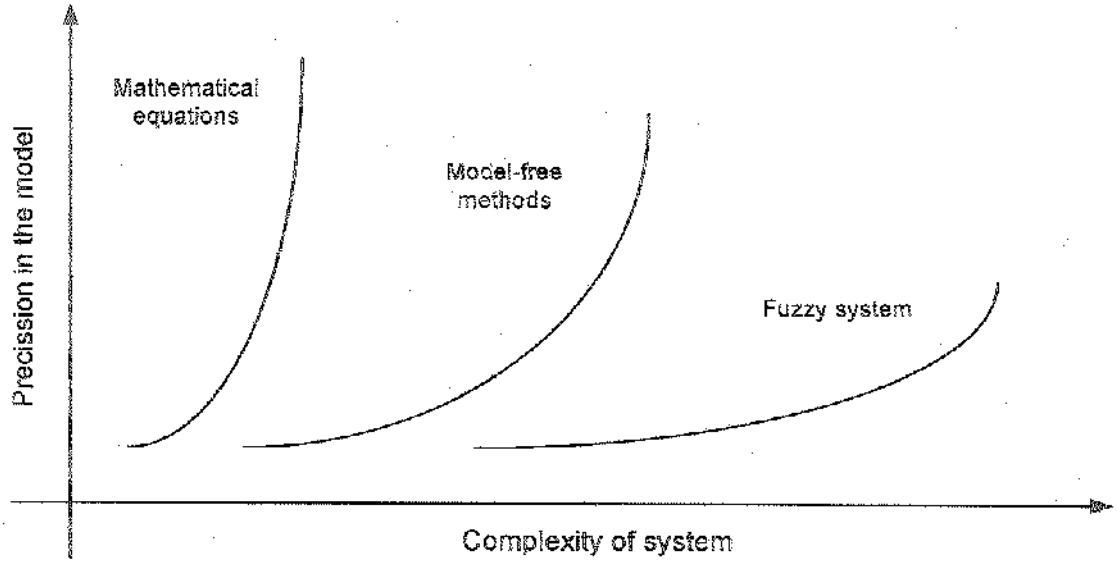
2.6 Logika Fuzzy

Fuzzy Logic pertama kali diperkenalkan oleh Dr. Lotfi A. Zadeh professor Computer Science University of California di Berkeley pada tahun 1965 dan berhasil di aplikasikan dalam bidang control oleh E.H. Mamdani. Sejak itu aplikasi dari *fuzzy logic* berkembang pesat. Kadang untuk masalah-masalah yang rumit tidak cukup hanya berbasis pada logika False(0) Atau True(1) , masalah yang berhubungan dengan pemikiran manusia misalnya. Fuzzy Logic menggunakan interval antara 0 sampai dengan 1.

Fuzzy Logic pada dasarnya merupakan logika bernilai banyak (*Multivalued logic*) yang dapat mendefinisikan nilai diantara keadaan yang biasa kita kenal seperti ya – tidak, hitam – putih, benar – salah dan nol – satu. *Fuzzy Logic* menirukan cara manusia mengambil keputusan dengan kemampuannya

bekerja dari data yang samar atau tidak rinci dan menemukan penyelesaian yang tepat.

Fuzzy Logic berangkat dari kenyataan bahwa dunia nyata adalah sangat kompleks. Kompleksitas ini muncul dari ketidakpastian dalam bentuk ke tidak telitian (*Imprecision*) informasi. Mengapa komputer yang dibuat manusia tidak mampu menangani persoalan yang kompleks dan tidak presisi ini sedangkan manusia bisa ?. Jawabnya adalah manusia mempunyai kemampuan untuk menalar (*Reasoning*) dengan baik yaitu kemampuan yang komputer tidak mempunyainya.



Gambar 2.4 Perbandingan Kompleksitas dan Keakuratan dari Metode yang digunakan.

Pada gambar diatas dapat dilihat perbandingan antara kompleksitas atau kerumitan suatu sistem dengan presisi dari model yang digunakan. Dapat dilihat bila pada suatu sistem dengan kompleksitas kecil, maka persamaan matematik dapat digunakan dan ketelitian yang dihasilkan menjadi sangat berguna dalam

pemodelan sistem. Bila kompleksitas bertambah tetapi data-data spesifikasi sistem tersedia maka metode pemodelan seperti *neural network* dapat digunakan untuk meminimalkan ketidakpastian melalui tindakan pembelajaran oleh sistem berdasarkan pola dan data yang tersedia.

Akan tetapi jika suatu sistem memiliki tingkat kompleksitas tinggi dimana hanya sedikit data yang tersedia dan informasi yang tersedia sangat tidak pasti (*ambigu*) dan tidak presisi. Persamaan matematik menjadi sulit untuk diciptakan dan digunakan, dalam hal ini *fuzzy logic* menjadi salah satu alternatif penyelesaiannya. *Fuzzy Logic* merupakan alternatif cara berpikir yang dapat memodelkan kekompleksan suatu sistem dengan menggunakan pengetahuan dan pengalaman yang kita punyai. Metode *fuzzy logic* menggunakan sekelompok aturan-aturan yang dibuat berdasarkan analogi rasional manusia untuk mengambil keputusan saat keadaan-keadaan tertentu terjadi daripada sebuah persamaan matematika yang rumit. Penalaran *fuzzy* akan menyediakan jalan untuk memahami kinerja dari sistem dengan cara menilai input dan output sistem dari hasil pengamatan. Definisi metode *fuzzy* secara lengkap adalah : suatu metode kontrol yang pengambilan keputusannya didasarkan pada aturan-aturan yang dibuat berdasarkan analogi rasional manusia.

Dalam penggunaan *Fuzzy Logic*, yang perlu diperhatikan adalah *truth values* (nilai kebenaran) dan *membership values* (nilai keanggotaan yang menghubungkan suatu elemen pada suatu himpunan dengan tingkat keanggotannya), yang nilainya berkisar antara 0.0 sampai 1.0. Dimana nilai 0.0 menunjukkan kesalahan mutlak dan 1.0 menunjukkan kebenaran mutlak.

2.6.1 Konsep Logika Fuzzy

Sebelum membahas logika fuzzy ada baiknya kita mengenal Fuzzy Set. Fuzzy sets adalah pengembangan yang lebih lanjut dari konsep matematika dari sebuah set. Sets pertama kali dipelajari secara formal oleh Ahli matematika dari Jerman Georg Cantor(1845-1918).Teori yang diungkapkannya mendapat banyak tentangan pada zamannya, sedangkan sekarang kebanyakan matematikawan percaya bahwa sangat mungkin untuk mengekspresikan sebagian besar masalah matematika ke Teori Set .

Untuk Engineer Control , *Fuzzy logic* dan *Fuzzy Relations* adalah hal yang paling penting untuk memahami bagaimana Aturan Fuzzy bekerja.

2.6.1.1 Konvensional Sets

Sets adalah beberapa koleksi dari objek-objek yang dapat diperlakukan sebagai suatu kesatuan. Georg Cantor mendeskripsikan sebuah set berdasarkan anggota-anggotanya, seperti sebuah item dari *universe* apakah anggota atau bukan. Istilah Set , koleksi, dan Class adalah sama seperti istilah *item, element, member*. Hampir semua yang dianggap set dapat diterima sebagai set dalam pengertian matematika.

Contoh :

a. Set dari bilangan integer positif yang kurang dari 4. merupakan sebuah Finite set dengan 4 anggota (0,1,2,3).

b. Set dari Dinosaurus hidup di basement museum Inggris. Set ini tidak mempunyai anggota dan dinamakan Set kosong (*empty set*).

c. Set dari hasil pengukuran yang lebih besar dari 10 Volt. Walaupun set ini infinite masih mungkin untuk menentukan apakah pengukuran yang diberikan merupakan sebuah anggota ataupun bukan.

Sebuah set dapat ditentukan berdasarkan anggota-angotanya dan anggota-anggotanya memberi sifat ke set tersebut sepenuhnya. List dari member A={0,1,2,3} menyatakan Sebuah Finite Set. Tidak bisa melist semua elemen dari sebuah infinite set, kita harus menyatakan beberapa properti yang menggambarkan elemen-elemen dari set, contohnya predikat x > 10. Set tersebut didefinisikan oleh elemen-elemen dari universe yang membuat predikat menjadi True. Jadi ada 2 cara mendeskripsikan sebuah set yaitu secara eksplisit di dalam sebuah list atau secara implisit dengan sebuah predikat.

2.6.1.2 Fuzzy Sets

Menurut Zadeh yang merupakan pencipta fuzzy logic Kebanyakan set lebih dari sebuah standar either-or untuk anggotanya. Sebagai contoh Set dari Orang-orang muda. Bayi dengan umur 1 tahun akan menjadi anggota set tersebut sedangkan orang dengan umur 100 tahun bukanlah anggota set tersebut. Tetapi bagaimana dengan orang yang berumur 20,30,40. Contoh lainnya adalah ramalan cuaca mengenai temperatur tinggi, angin kencang, hari yang cerah. Pada kasus lain sebuah standar kelihatan bukan fuzzy, tetapi sebenarnya Fuzzy antara lain : Batas kecepatan 60 km/jam, jam check out dari hotel jam 12, umur bapak-bapak

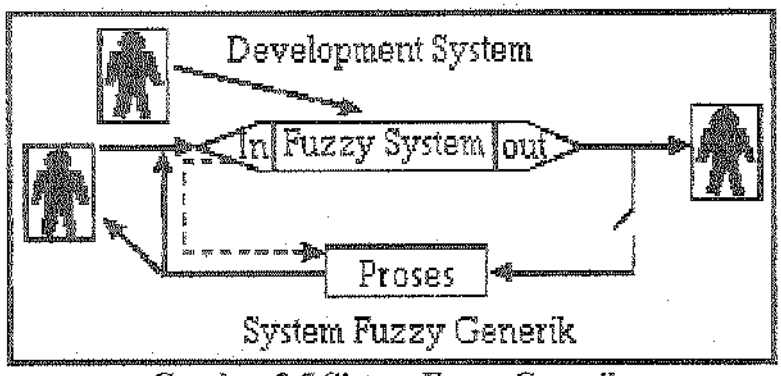
50 tahun. Professor Zadeh mengajukan sebuah grade keanggotaan dimana transisi dari keanggotaan(*membership*) menjadi Non anggota terjadi secara bertahap bukan secara tiba-tiba. Tingkatan keanggotaan untuk semua anggotanya inilah yang menggambarkan sebuah Fuzzy Set. Grade keanggotaan secara normal berkisar diantara nilai 0 sampai 1 yang sering disimbolkan dengan tanda μ. Makin tinggi nilainya makin tinggi pula tingkat keanggotannya.

Operasi di Fuzzy Set ada 3 antara lain : *Intersection*, *Union*, dan *Complement*.

Seperti telah dikatakan sebelumnya dalam menggunakan fuzzy logic, ada beberapa hal pokok yang perlu diperhatikan yaitu *truth values* dan *membership values*, *fuzzy sets*, *membership degree* dan *membership function*. Himpunan fuzzy adalah semua penggambaran tentang keadaan suatu masalah. *Membership degree* dan *membership function* adalah termasuk dalam himpunan *fuzzy*. Membership *function* memberi hasil yang disebut membership values. Pada masalah yang lebih luas, akan ditemukan sub himpunan *fuzzy* yang satu akan berpotongan dengan sub himpunan *fuzzy* yang lain dan himpunan fuzzy akan lebih dari satu himpunan. Hal ini menyebabkan perlunya suatu operasi himpunan fuzzy, yang berguna untuk menentukan tingkat derajat keanggotaan.

Berikut penjelasan beberapa jenis teori fuzzy :

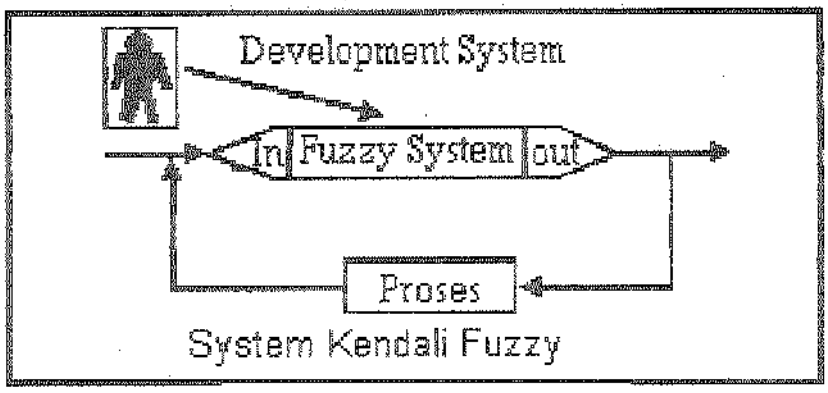
a. Sistem *Fuzzy* Secara Umum



Gambar 2.5 Sistem Fuzzy Generik

Sistem *fuzzy* secara umum dapat dilihat pada Gambar diatas. Pada gambar tersebut terdapat blok proses, sistem *fuzzy*, dan sistem pengembangan (*development system*). Pihak *developer* diletakkan paling atas pada gambar ini. Selain itu, terdapat dua operator, yaitu seorang yang bertanggung jawab atas masukan untuk sistem *fuzzy* dan keluaran dari proses, dan seorang lagi bertugas membawa masukan ke dalam proses dan menentukan keluaran dari sistem *fuzzy*. Operator ini sebenarnya tidak mesti seorang operator manusia, biasanya sistem *fuzzy* atau *non-fuzzy* yang berfungsi mengantarkan masukan atau keluaran sinyal proses. Dari gambar ini dapat diturunkan beberapa sistem sistem *fuzzy*, seperti pengendali *fuzzy*, klasifikator *fuzzy*, dan sistem pendiagnosaan *fuzzy*.

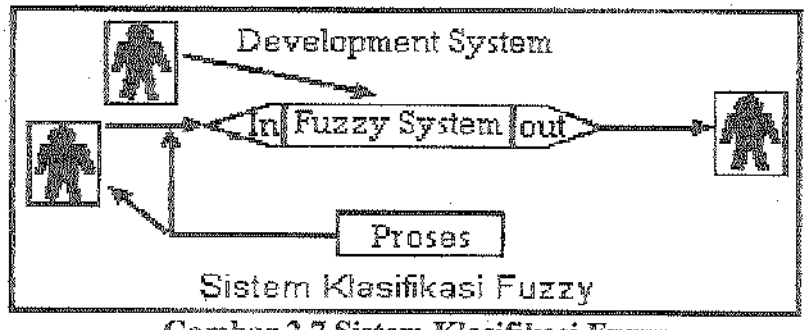
b. **Sistem Kendali *Fuzzy***



Gambar 2.6 Sistem Kendali Fuzzy

Sebuah kendali *fuzzy* yang digambarkan pada Gambar diatas merupakan suatu sistem lingkar tertutup, di mana tidak terdapat operator yang menjadi bagian dari sistem lingkar kendali (*control loop*). Contoh dari sistem kendali ini adalah *vacuum cleaner*. Sistem pada alat ini mengatur daya motor penghisap tergantung pada banyaknya debu di lantai atau karpet. Contoh lain dari sistem kendali *fuzzy* adalah optimisasi torsi dalam sistem anti slip yang digunakan kereta listrik dan sistem kereta bawah tanah. Masukan sistem kendali berupa kecepatan kereta dan koefisien resistansi rel.

c. **Sistem Klasifikasi Fuzzy**

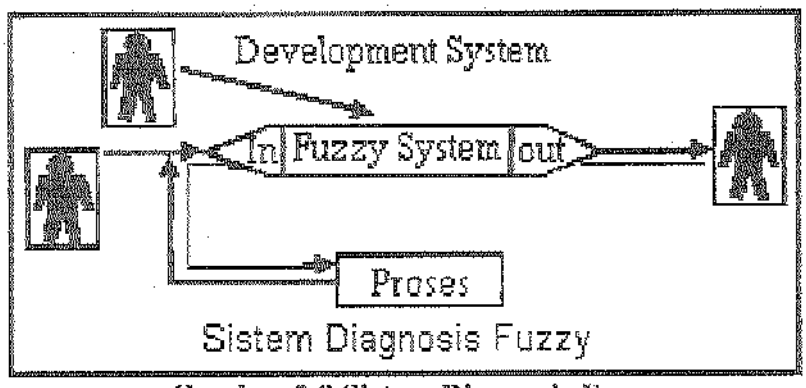


Gambar 2.7 Sistem Klasifikasi Fuzzy

Pada sistem klasifikasi *fuzzy* diatas tidak terdapat *loop* tertutup. Sistem ini hanya menerima masukan dan keluaran dari proses untuk selanjutnya memberikan informasi berupa kondisi (*state*) dari proses tadi. Informasi kondisi ini dapat digunakan untuk mengendalikan sistem atau memberikan tanggung jawab kendali kepada operator. Secara matematis, sistem klasifikasi lebih dekat pada teori himpunan daripada teori fungsi. Pada sistem ini, sifat kesamaan (*Vagueness*) sering ditemui pada opini pakar dan jarang menggunakan model relasi *fuzzy*.

Contoh dari sistem klasifikasi *fuzzy* adalah mesin cuci fuzzy. Beberapa variabel/parameter mesin cuci ditentukan berdasarkan jumlah dan jenis pakaian. Keluaran atau informasi dari sistem klasifikasi ini digunakan untuk menentukan jenis *spin-dry* serta lembut atau kasar gesekan pakaian yang optimal. Contoh ke dua dari sistem *fuzzy* ini adalah sistem transmisi otomatik *fuzzy*. Sistem ini menggunakan beberapa sensor yang ditaruh pada sistem ABS, sistem *power steering*, sistem kendali motor, dan bagian penting lainnya. Selama kendaraan berjalan, sistem ini akan terus memantau dan menilai kondisi mobil tersebut, seperti beban kendaraan, kondisi mobil pada saat melewati jalan yang menanjak atau menurun dan kondisi-kondisi lainnya. Pada Gambar diatas, gambar operator manusia pada kiri dan kanan sistem klasifikasi *fuzzy*, biasanya merupakan suatu sistem khusus yang bertugas memberikan informasi yang diperlukan untuk kemudian di proses.

d. Sistem Diagnosis *Fuzzy*



Gambar 2.8 Sistem Diagnosis Fuzzy

Pada sistem diagnosis *fuzzy* gambar diatas peranan manusia/operator lebih dominan. Pengiriman data dilaksanakan oleh operator ke dalam sistem, ketika sistem memerlukan data tambahan. Selain itu operator dapat meminta atau menanyakan informasi dari sistem diagnosis berupa hasil konklusi diagnosis atau prosedur detail hasil diagnosis oleh sistem. Dari sifat sistem ini, sistem diagnosis *fuzzy* dapat digolongkan pada sistem pakar *fuzzy*. Sistem pakar *fuzzy* adalah sistem pakar yang menggunakan notasi *fuzzy* pada aturan-aturan dan proses *inferensi* (*logika keputusan*). Salah satu kelebihan sistem pakar *fuzzy* dibandingkan sistem pakar konvensional adalah jumlah aturan lebih sedikit, sehingga sistem lebih transparan untuk dianalisa. Kekurangannya adalah kehandalan sistem sangat tergantung pada baik-buruknya proses pengumpulan aturan seperti prosedur pertanyaan dan komponen-komponen kuisioner, serta sering terjadi kesulitan untuk menyimpulkan suatu pernyataan tertentu oleh operator.

Bidang aplikasi sistem diagnosis ini biasanya suatu proses yang besar dan kompleks, sehingga sangat sulit dianalisa menggunakan algoritma eksak dan dimodelkan dengan model matematika biasa. Pada permulaan persiapan sistem,

jumlah aturan yang digunakan ini biasanya sangat banyak. Namun pada tahap akhir, jumlah aturan akan lebih sedikit dan mudah dibaca. Ini merupakan sifat sistem pakar *fuzzy*, seperti yang dikatakan oleh *Prof. Zadeh*, bahwa sistem pakar *fuzzy* akan menggunakan aturan-aturan yang lebih sedikit dibandingkan sistem pakar konvensional sehingga mudah dibaca dan membantu menghindarkan inkonsistensi dan inkomplit sistem pengendali. Yang perlu diperhatikan pada sistem diagnostik ini adalah, tidak berlakunya proses defuzzifikasi, karena sistem ini hanya menghasilkan sifat keluaran berupa aproksimasi linguistik yang merupakan suatu pernyataan atau jawaban yang mudah dipahami oleh operator.

2.6.2 Fuzzy Logic Controller

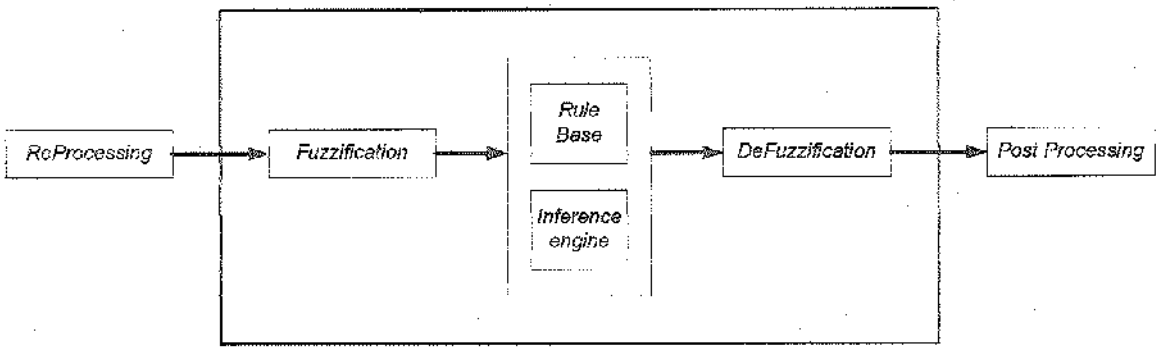
Fuzzy dipergunakan sebagai pengendali motor dc Karena *fuzzy logic* mempunyai kinerja yang baik dalam mengatasi sistem tidak linear dengan data input yang sedikit dan persamaan matematis yang diperlukan pun tidak rumit atau sederhana jika dibandingkan dengan metode-metode lainnya seperti *Proportional-Integral-Derivative* (PID), dan *Model Reference Adaptive Control* (MRAC).

Sistem kontrol berbasiskan mikroprosesor digital banyak digunakan dalam proses kontrol suatu sistem. Bila dibandingkan dengan sistem kontrol secara analog, sistem kontrol digital jauh lebih fleksibel dalam responnya. Hal tersebut disebabkan karena banyaknya variasi algoritma yang bisa diterapkan dalam sistem kontrol digital. Sistem kontrol digital mempunyai keunggulan lebih baik untuk menangani ketidak-lincaran dari sistem, perubahan keadaan pada

sistem seperti perubahan beban, supply sistem dan juga perubahan parameter-parameter lainnya.

Algoritma yang digunakan dalam sistem kontrol digital dapat dipilih berdasarkan beberapa metode tertentu seperti *Proportional-Integral-Derivative* (PID), *Model Reference Adaptive Control* (MRAC). Metode kontrol PID cukup efektif untuk menangani sistem dengan respon tetap namun bila sistem menjadi tidak linear maka sistem PID akan mengalami kesulitan. Metode MRAC dapat menangani ketidak-linearan sistem dan lingkungan kontrol yang berubah-ubah dengan cara membandingkan hasil proses output aktual dengan model referensi. Namun metode ini membutuhkan model matematika dari proses yang berlangsung untuk mensimulasikan relasi input-output. Untuk menciptakan model matematika dari sistem tentunya merupakan kesulitan sendiri.

FLC adalah merupakan pengembangan dari teori *fuzzy logic* dan digunakan untuk pengontrolan suatu alat atau sistem. Seperti halnya pola pikir manusia yang membuat keputusan (*making decision*), didasarkan atas aturan-aturan yang ada maka FLC akan bertindak sama. Pola pikir manusia dalam membuat keputusan bila diterapkan pada logika komputer merupakan kalimat if-then, begitu pula pada FLC, FLC akan bekerja berdasarkan aturan yang ada. Contohnya If A then Y, atau If B then Z. Dibawah ini adalah bagan FLC :



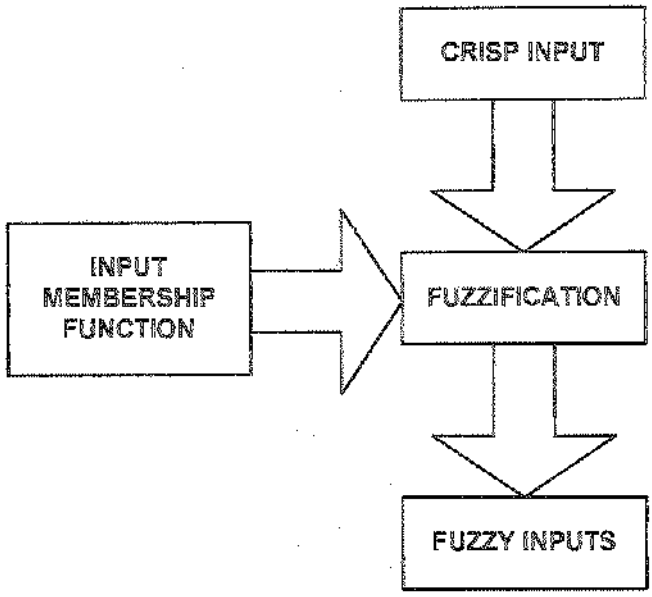
Gambar 2.9 Fuzzy Logic Controller

Aturan pada *fuzzy logic* (*fuzzy rules*) sangatlah penting didalam FLC, untuk menerapkan teori *fuzzy logic*. Oleh sebab itu diperlukan langkah-langkah seperti bagan diatas tersebut yang meliputi :

- *Fuzzification*, yaitu proses penentuan masukan (input) yang akan diubah nilainya oleh *membership function*.
- *Rule Evaluation*, yaitu proses dimana nilai hasil dari *fuzzification* dibandingkan dengan aturan-aturan yang telah dibuat.
- *Defuzzification*, mengubah hasil dari *rule evaluation* menjadi nilai yang dapat dijalankan (*crisp output*).

2.6.2.1 Fuzzifikasi

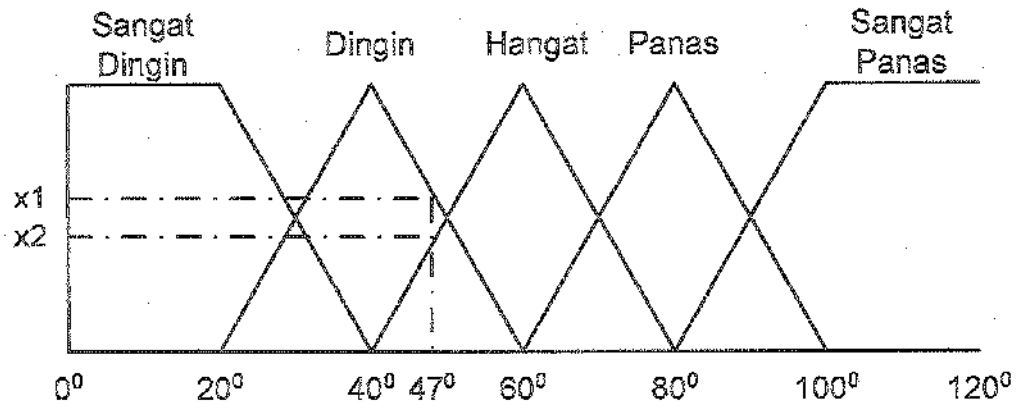
Proses ini berfungsi untuk merubah suatu besaran analog/digital menjadi *fuzzy input*. Secara diagram blok dapat dilihat pada gambar dibawah ini :



Gambar 2.10 Fuzzification

Prosesnya dapat dijelaskan sbb :

Suatu besaran analog dimasukkan sebagai input (*crisp input*), lalu input tersebut dimasukkan pada batas *scope/domain* (merupakan suatu batas dari kumpulan input tertentu, misalnya panas antara 60-100^{0}C) sehingga input tersebut dapat dinyatakan dengan label (dingin, hangat, panas) dari *membership function*. *Membership function* ini biasanya dinamakan *membership function input*. Dari *membership function* kita bisa mengetahui berapa *degree of membership function*-nya (memberikan bobot pada suatu input yang telah kita berikan sehingga input tadi dapat dinyatakan dengan suatu nilai, biasanya antara 0,0-1,0). Contoh dari proses *fuzzification* adalah sbb :

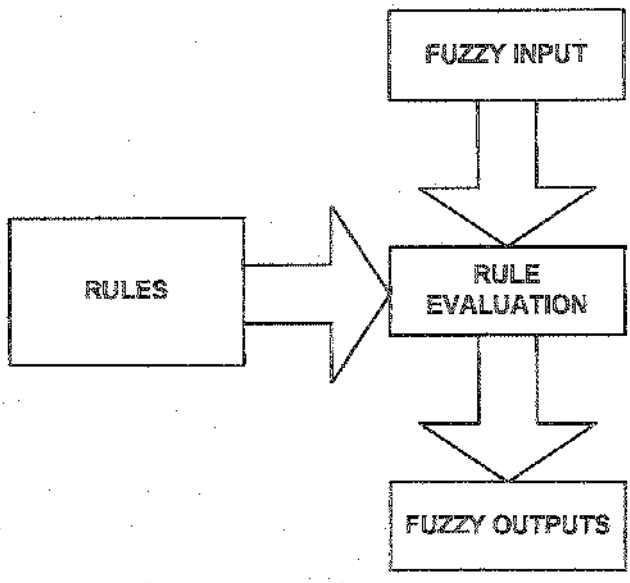


Gambar 2.11 Membership Function

Dari gambar diatas diperoleh crisp input adalah 47^0C. Sehingga didapat 2 fuzzy input yang masing-masing adalah : Dingin (X1) dan Hangat (X2), nilai X1 dan X2 dapat dicari dengan rumus persamaan garis. Yang menentukan system tersebut sensitive atau tidak adalah *membership function* ini. Jika *membership function*-nya banyak maka system akan menjadi sangat sensitive. Yang dimaksud sensitive disini adalah jika inputnya berubah sedikit saja maka system akan cepat merespon dan menghasilkan suatu output lain. Output dari *fuzzification* ini adalah sebuah nilai input *fuzzy*.

2.6.2.2 Rule Evaluation

Proses ini berfungsi untuk mencari suatu nilai *fuzzy output* dari *fuzzy input*. Suatu nilai *fuzzy input* yang berasal dari proses *fuzzification* kemudian dimasukkan kedalam sebuah rule yang telah dibuat untuk dijadikan sebuah *fuzzy output*, dapat digambarkan sbb:



Gambar 2.12 Rule Evaluation

Gambar diatas merupakan bagian utama dari *fuzzy*, karena disinilah sistim akan belajar menjadi pintar atau tidak. Jika system tersebut tidak pintar dalam mengatur *rule* maka system yang akan dikontrol menjadi kacau. Format dari rule adalah sbb:

"If antecedent1 operator antecedent2 then consequent1 operator consequent2"

Contoh :

"If suhu is panas and kelembapan is kering then penyemprot is sangat lama"

Ada beberapa operator yang digunakan dalam *fuzzy* : *And, Or, Not*. Jika operator yang digunakan adalah *And* maka input yang terkecil diambil. Misalnya :

"If suhu is panas (0,15) and kelembapan (0,19) then penyemprot sangat lama"

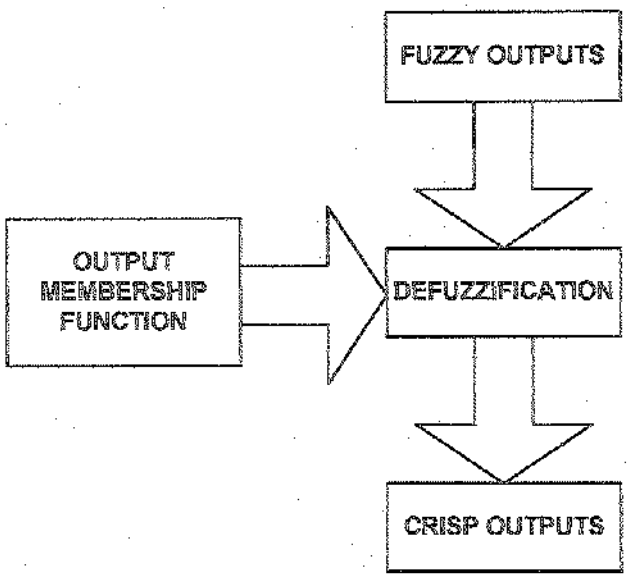
Nilai *fuzzy output* dari pernyataan tersebut adalah 0,15. Nilai 0,15 dan 0,19 dari contoh diatas diambil dari dua membership function input dengan cara menarik garis lurus vertical dari nilai yang diinginkan. Jika operator yang digunakan adalah *Or*, maka *fuzzy outputnya* diambil dari nilai yang terbesar. Jika operator

yang digunakan adalah adalah *operator Not* maka *fuzzy outputnya* adalah kebalikannya. Misalnya *Not* 0,9 maka akan menghasilkan 0,1 dan *Not* 0,8 akan menghasilkan 0,2.

2.6.2.3 Defuzzifikasi

Proses defuzzifikasi merupakan proses untuk memetakan data dari lingkungan fuzzy kembali menjadi data yang bisa diaplikasikan untuk kontrol sistem. Sebagai contoh, dalam memberikan instruksi ke motor DC untuk mempercepat putarannya tidak bisa dilaksanakan dalam lingkup besaran fuzzy seperti "tambah sedikit putaran motor". Kita harus menerjemahkan perintah tersebut kedalam suatu nilai crisp misalnya : tambah *duty cycle* motor sebesar 25%.

Proses defuzzifikasi dimulai dari masukknya fuzzy output kedalam proses kemudian sistem akan mengambil data dari output membership function untuk melihat perintah (dalam bentuk nilai *Center_point)* apa yang harus diaplikasikan untuk setiap fungsi keanggotaan yang aktif dalam proses fuzzifikasi.



Gambar 2.13 Defuzzification

Nilai-nilai tersebut dimasukkan kedalam suatu rumus yang dinamakan *COG (Center Of Gravity)* untuk mendapatkan hasil akhir yang disebut *crisp output*. *Crisp output* adalah suatu nilai digital yang akan dibutuhkan oleh system untuk mengolah data pada system yang telah dirancang. Rumus Center Of Gravity dapat ditulis pada persamaan dibawah ini :

$$Crisp\_Output = \frac{\sum_i (Fuzzy\_Output_i) \times (Point\_Center\_Position_i)}{\sum_i (Fuzzy\_Output_i)}$$