PERANCANGAN SISTEM LENGAN ROBOT TIGA DERAJAD KEBEBASAN BERBASIS *FUZZY LOGIC CONTROLLER*

TUGAS AKHIR

Disusun Oleh :

BENEDICTUS INDRAJAYA
NRP : 2292 100 013

JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2000

PERANCANGAN SISTEM LENGAN ROBOT TIGA DERAJAD KEBEBASAN BERBASIS *FUZZY LOGIC CONTROLLER*

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Sarjana Teknik Elektro
Pada
Bidang Studi Elektronika
Jurusan Teknik Elektro
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember
Surabaya

Mengetahui / Menyetujui

Dosen Pembimbing I	Dosen Pembimbing II
Ir. Iskandar Zulkarnain	Moch. Rivai, ST., MT.
NIP. 130 520 755	NIP. 132 088 341

SURABAYA
Pebruari, 2000

ABSTRAK

ABSTRAK

Pemanfaatan robot dalam dunia industri sangat menunjang proses produksi dengan kecepatan dan kualitas kontrol yang baik. Dalam proses desain kontroler untuk robot secara matematis membutuhkan perhitungan yang rumit dan kemungkinan kesalahan dalam pemodelan secara matematis sistem.

Pada pengontrolan dengan mikrokontroler fuzzy ditawarkan desain sistem yang mudah dan intuitif dengan hasil yang memuaskan. Pada tugas akhir ini diterapkan mikrokontroler fuzzy NLX220 untuk mengontrol gerakan lengan robot tiga derajad kebebasan, yang diharapkan dapat menyederhanakan pemodelan matematis untuk sistem kontrol gerakannya.

Sistem yang dirancang terdiri dari mekanis, elektronis dan perangkat lunak logika fuzzy serta perangkat lunak pada PC. Untuk mekanis, jenis robot yang dirancang adalah robot sistem *elbow* dengan tiga sumbu rotasi. Bagian elektronis terdiri dari rangkaian-rangkaian: modul NLX220, sensor posisi, driver motor dc, penghasil error dan delay error, serta ADC-DAC yang diinterfacekan pada LPT1 PC. Proses penyusunan aturan-aturan fuzzy untuk kontrol sistem tersebut dikembangkan dari sistem kontrol proporsional-derivativ (PD) konvensional, digunakan untuk mengontrol posisi sudut pada tiga buah persambungan/*joint*. Perangkat lunak pada PC digunakan untuk menyimpan titik-titik lintasan yang akan dilewati lengan robot, sekaligus untuk menampilkan posisi sudut masing-masing *joint*. Dari pengujian dengan memberi nilai set posisi pada masing-masing *joint*, kemudian diukur posisi aktualnya, didapat error maksimum sekitar 20%.

KATA PENGANTAR

KATA PENGANTAR

Dengan mengucap syukur ke hadirat Tuhan Yang Maha Esa yang telah melimpahkan rahmat-Nya, sehingga penulis dapat menyelesaikan tugas akhir yang berjudul :

PERANCANGAN SISTEM LENGAN ROBOT TIGA DERAJAD KEBEBASAN BERBASIS FUZZY LOGIC CONTROLLER

Tugas akhir ini merupakan salah satu syarat untuk menyelesaikan studi S1 pada Jurusan Teknik Elektro Fakultas Teknologi Industri Institut Teknologi Sepuluh Nopember Surabaya. Tugas akhir ini memiliki beban 4 SKS (Satuan Kredit Semester).

Dalam penyusunan tugas akhir ini, penulis berdasar pada teori-teori yang diperoleh selama kuliah maupun literatur-literatur penunjang, hasil pengamatan, dan penelitian selama perencanaan dan pembuatan alat, bimbingan dari para dosen pembimbing serta sumbangan pemikiran dan saran dari semua pihak yang menunjang terselesaikannya tugas akhir ini.

Untuk itu penulis ingin mengucapkan terima kasih kepada :

- Bpk. Ir. Iskandar Zulkarnain selaku dosen pembimbing dalam tugas akhir ini, yang banyak memberikan masukan.
- Bpk. Moch Rivai, ST, MT selaku dosen pembimbing dalam tugas akhir ini, yang banyak memberikan masukan, nasihat dan semangat.

- Bpk. Ir. Soetikno sebagai koordinator bidang studi Elektronika di Jurusan Teknik Elektro ITS.
- Ayah, Ibu dan saudara-saudara yang telah membantu, memberi dorongan, bimbingan serta doa kepada penulis.
- Rekan-rekan bidang studi Elektronika yang telah memberikan sumbangan pemikiran dan saran selama menyelesaikan tugas akhir ini.
- Fransisca H, yang telah banyak memberi dorongan moral, pengertian dan bantuanya sehingga mempercepat terselesaikannya tugas akhir ini.
- Semua pihak yang turut mendukung dalam menyelesaikan tugas akhir ini.

Sebagai penutup, penulis menyadari bahwa tugas akhir ini mempunyai banyak kekurangan. Kritik dan saran dari semua pihak sangat penulis harapkan. Akhir kata, semoga tugas akhir ini bermanfaat bagi para pembaca umumnya dan mahasiswa Teknik Elektro pada khususnya.

Surabaya, Februari 2000

Penulis

DAFTAR ISI

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR TABEL

BAB I

PEDAHULUAN

BAB I
PENDAHULUAN

1.1. LATAR BELAKANG

Sistem kontrol untuk robot untuk n derajad kebebasan selama ini biasanya menggunakan kontrol proporsional-derivativ (PD) untuk motor penggerak yang sederhana atau proposional-integral-derivativ (PID) untuk motor penggerak yang kompleks (misal motor AC).

Kelemahan dalam metode ini adalah sulitnya memodelkan sistem secara matematis karena banyak parameter yang harus dipertimbangkan. Kelemahan yang kedua adalah sulitnya perancangan model sistem kontrolnya secara maematis yang selama ini dengan menggunakan sistem persamaan kinematik/dinamik (dengan sistem matrik), dimana hal ini mempunyai kekurangan dalam hal lamanya waktu perancangan dan pembuatan kontroler, serta membutuhkan seorang yang ahli dalam desain matematis sistem kontrolnya.

Logika fuzzy yang ditemukan sejak pertengahan 1960-an oleh Profesor Lofti Zadeh, ternyata mampu menjawab ketidakmampuan teknologi digital dalam mengenali perubahan parameter yang tidak jelas dan menyederhanakan model matematis sistem yang akan dikontrol. Dengan penelitian lanjutan dari Bart Kosko dan lain-lain, teknologi fuzzy dikembangkan untuk penerapan kontrol.

1.2. PERMASALAHAN

Studi yang akan dilakukan dalam Tugas Akhir ini akan diarahkan pada masalah-masalah mengenai:

- Memperbaiki kelemahan dalam pembuatan sistem kontrol untuk sistem lengan robot tiga derajad kebebasan, yakni masalah rumitnya perancangan model matematis dengan menerapkan kontroler logika fuzzy.
- Penerapan logika fuzzy dalam kontrol posisi motor dc penggerak lengan robot, yang dikembangkan dari sistem kontroler proporsional-derivativ (PD).
- Penerapan kontroler berbasis *fuzzy logic* NLX220 untuk perancangan sistem lengan robot tiga derajad kebebasan.

1.3. TUJUAN

Tujuan dari Tugas Akhir ini adalah sebagai berikut:

- Mempelajari teori penunjang sistem robotik tiga derajad kebebasan yang diaplikasikan pada lengan robot .
- Mempelajari kontroler berbasis *fuzzy logic* NLX220.
- Perancangan dan pembuatan sistem lengan robot tiga derajad kebebasan berbasis kontroler logika fuzzy NLX220.
- Melakukan uji coba secara *hardware* dan *software*.
- Mengambil kesimpulan mengenai keandalan sistem yang dibuat.

1.4. METODOLOGI

Untuk mencapai tujuan diatas, maka langkah-langkah yang akan dilakukan:

⇒ Studi literatur/kepustakaan:

Literatur/kepustakaan yang dipelajari adalah tentang dasar teori tentang sistem robotik, konsep logika fuzzy, teori tentang motor dc, sistem akuisisi data dan teori tentang rangkaian analog.

⇒ Perancangan sistem:

Sistem yang dirancang terdiri dari perangkat keras, yaitu perangkat keras mekanis dan elektronis, serta perancangan perangkat lunak, dalam hal ini perangkat lunak untuk NLX220 dan perangkat lunak dalam PC.

⇒ Pembuatan sistem

Sistem yang telah dirancang direalisasikan dalam bentuk sistem yang nyata.

⇒ Pengujian sistem:

Pengujian sistem dilakukan perbagian modul perangkat keras dan perangkat lunak, kemudian dilakukan pengujian sistem secara keseluruhan.

⇒ Penyusunan buku laporan Tugas Akhir.

BAB II

TEORI PENUNJANG

BAB II

TEORI PENUNJANG

2.1 LENGAN ROBOT TIGA DERAJAD KEBEBASAN

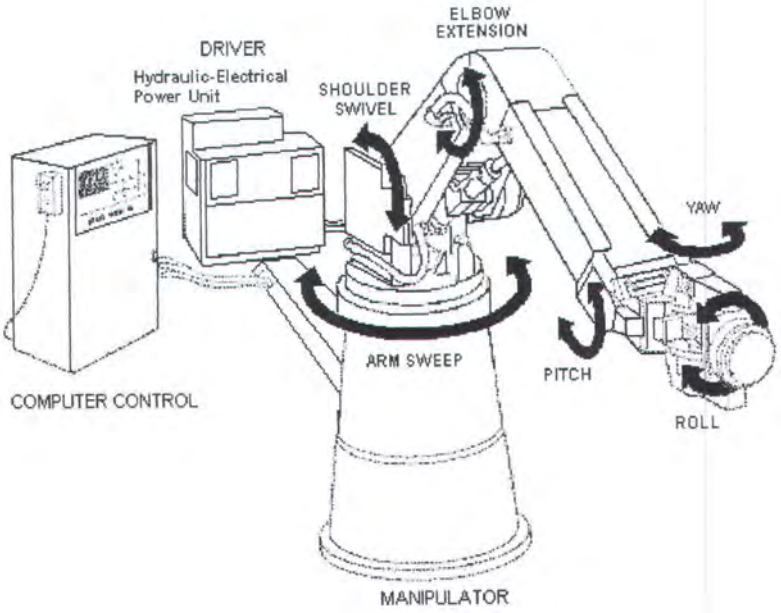
Pada sub bab ini akan dibahas tentang dasar teori yang menunjang dalam perancangan robot tiga derajad kebebasan sistem rotasi / *articulate*. Mula-mula akan dibahas bagian-bagian robot, klasifikasi robot, kemudian trayektori gerakan robot.

2.1.1 Pendahuluan

Istilah robot berasal dari bahasa Czech "Robota" yang berarti "pekerja". Istilah ini menjadi populer ketika pada tahun 1921 seorang penulis naskah drama bernama Karel Capek menerbitkan tulisan berjudul *Rossum's Universal Robot*. Robot dalam tulisan Karel ini dibuat mirip dengan manusia hanya saja tidak memiliki perasaan dan mampu bekerja dua kali lebih kuat daripada manusia. Robot-robot industri yang banyak dikenal tidaklah seperti manusia (secara fisik), tetapi memiliki fungsi kerja seperti manusia. Konsep robot industri ini dipatenkan di Amerika Serikat pada tahun 1954 oleh G.C. Devol (*US Patent* No. 29882377)

Sistem robot modern terdiri oleh paling tidak tiga bagian seperti tampak pada gambar 2.1. Bagian-bagian tersebut antara lain:

1. Manipulator, yaitu struktur mekanis yang ada.



Gambar 2.1. Bagian-bagian Utama Robot

Secara umum susunan manipulator terdiri dari *main frame* dan *wrist* (pergelangan) dengan *tool* pada ujungnya. *Tool* ini dapat berupa alat las, penyemprot cat atau *gripper* dengan berbagai fungsi sesuai tugas kerja dari robot. Alat-alat ini menempel pada ujung lengan robot, sehingga sering disebut *end effector*. *Main frame* sering disebut dengan *arm* (lengan) dan umumnya terdiri dari rangkaian hubungan mekanis (*mechanical links*) yang dihubungkan oleh *joint* dengan *link* yang lain. Fungsi *joint* adalah untuk mengontrol gerakan antar *link*. *Joint* berfungsi untuk menggerakkan *gripper* disebut dengan *wrist*[1] . Pada umumnya sebuah *wrist* memiliki *joint* yang mengatur gerakan *roll*, *pitch* dan *yaw* seperti pada gambar 2.1 diatas.

[1] Yoram Korem, ROBOTIC FOR ENGINEER, (Singapore : MCGraw-Hill, 1987), p.12.

2. *Driver* untuk menggerakkan *joint* manipulator.

   Driver yang dipakai untuk menggerakkan *joint* berfungsi untuk menempatkan *end effector* pada posisi yang diinginkan sesuai dengan tugasnya. Gerakan *end effector* didapatkan dengan mengatur posisi dan kecepatan dari sumbu-sumbu gerakan robot. Sumbu gerakan ini merupakan derajat kebebasan dari robot. Sehingga robot dengan tiga derajat kebebasan memerlukan tiga buah pengontrol sumbu gerakan. Kombinasi dari ketiga sumbu tersebut akan memberikan posisi dari *end effector*.

   Sumbu gerakan dari robot bisa berupa sumbu putar (*rotary*) atau linier. Sumbu putar umumnya memakai *driver* motor-motor elektris sedangkan sumbu linier bisa menggunakan motor-motor hidrolis. Pada umumnya sumbu putar lebih mudah diwujudkan daripada sumbu linier tetapi akurasi sumbu putar lebih buruk.

3. Komputer sebagai kontroler dan penyimpan program tugasnya.

   Untuk menggerakkan *driver* agar mampu berjalan pada koordinat yang diinginkan diperlukan suatu pengontrol agar mampu memperhitungkan posisi yang tepat. Untuk ini umumnya dipakai suatu komputer sebagai kontrolernya. Fungsi komputer disamping sebagai kontroler juga untuk menyimpan program-program untuk menjalankan tugas dari robot tersebut.

2.1.2 *End Effector*

End effector menjadi bagian paling penting dari suatu robot karena *end effector* adalah alat dari robot untuk menyelesaikan tugasnya. *End effector* sendiri

dapat dibagi menjadi dua yaitu *gripper* dan *tool*. *Gripper* adalah *end effector* yang memegang (*to grip*) bagian dari alat operasi. Umumnya dipakai untuk tugas *point to point* (PTP) seperti mengatur mesin. Sedangkan *tool* adalah alat untuk melakukan tugas kerjanya. Robot akan meletakkan *tool* pada daerah kerja (*working area*). Jika memakai *tool*, maka trayektori perlu diperhitungkan dengan lebih baik.

2.1.3 Klasifikasi Sistem Robot

Ada banyak cara dalam melakukan klasifikasi dari robot. Sistem dari robot dapat dibagi menjadi tiga tipe dengan membaginya berdasarkan beberapa klasifikasi berikut ini.

2.1.3.1 Tipe sistem gerakannya

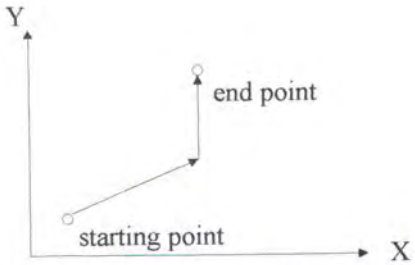
Ada dua tipe sistem gerakan yaitu yang dikenal dengan sistem *Point to Point* dan sistem *Continuous Path*. Sistem ini ditentukan berdasarkan perubahan posisi *end effector* dari robot. Pembagiannya dapat dijelaskan sebagai berikut ini.

1. Tipe *Point to Point* (PTP)

   Pada sistem ini robot bergerak dari satu titik (*starting point*) menuju titik akhir (*end point*) dengan melalui titik-titik antara. Secara umum operasi PTP dapat dirumuskan sebagai : robot bergerak pada beberapa lokasi yang telah ditentukan dan kemudian gerakan akan terhenti. Yang bergerak disini adalah *end effectornya*. Pada sumbu kartesian gerakan *point to point* dapat digambarkan seperti gambar 2.2[2] .

---

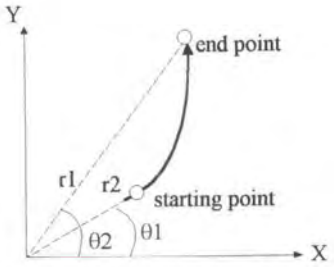
[2] Ibid, p. 5.



Gambar 2.2 Trayektori pada sistem PTP kartesian

2. Tipe *Continuous Path* (CP)

Pada sistem ini *tool* menjalankan tugasnya selama sumbu sedang bergerak, seperti misalnya pada robot untuk mengelas. Tugas robot dalam pengelasan adalah membawa alat las sepanjang jalur yang telah diprogramkan. Pada sistem ini semua sumbu bergerak secara bersama-sama dengan kecepatan yang berbeda. Kecepatan ini dikendalikan oleh komputer pengontrol agar sesuai dengan trayektori yang telah ditentukan. Contoh trayektorinya adalah seperti gambar 2.3[3]. Sisten CP umumnya dipakai untuk lengan robot yang *end effector*nya adalah *tool* yang menjalankan tugasnya pada suatu jalur tertentu. Pada kondisi tertentu sistem ini memerlukan gerakan sumbu geraknya untuk bergerak secara simultan.



Gambar 2.3 *Trayektori* pada sistem CP

[3] Ibid, p. 25.

2.1.3.2 Tipe *Control Loop*

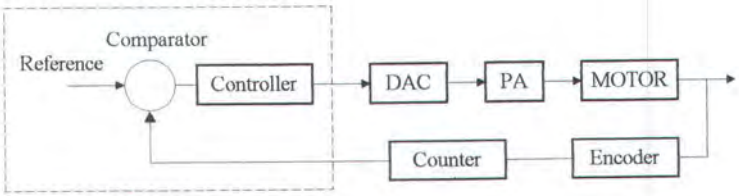
Ada dua buah sistem pengontrolan yang dikenal, yaitu :

1. Sistem *Open Loop*

Sistem *open loop* ini tidak memerlukan inputan dari hasil gerakan *joint*. Jadi hasil perhitungan dari komputer pengontrol akan langsung diberikan pada *driver* untuk menggerakkan joint pada posisi tertentu. Hasil gerakan *joint* ini tidak perlu di-*feedback*-kan lagi ke komputer.

2. Sistem *Closed Loop*

Berbeda dengan sistem *open loop*, maka sistem *closed loop* memerlukan umpan balik dari hasil gerakan motornya (umumnya dipakai motor DC atau motor servo). Blok diagram dari Struktur robot dengan tiga *joint open loop* dapat dilihat pada gambar 2.4.sistem *closed loop*[4].



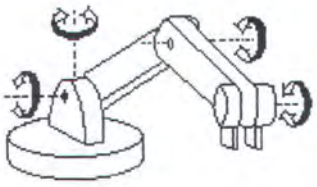
Gambar 2.4 Diagram Blok Sistem *Closed Loop*

2.1.3.3 Struktur Manipulator

Secara stuktur manipulator dapat dibagi menjadi empat sistem berdasarkan sistem koordinat dari *main frame*, yaitu:

1. Sistem *Cartesian* (tiga sumbu linier), dimana semua gerakannya linier/prismatik.

2. Sistem *Cylindrical* (dua sumbu linier dan satu sumbu putar), dimana mempunyai satu gerakan rotasi dan dua gerakan linier/prismatik.
3. Sistem *Spherical* (satu sumbu linier dan dua sumbu putar), gerakan yang dimiliki adalah dua gerakan rotasi dan satu gerakan prismatik.
4. Sistem *Articulated/jointed* (tiga sumbu putar), dimana semua gerakanya rotasi.



Gambar 2.5 Struktur Manipulator *Articulated* Robot[5]

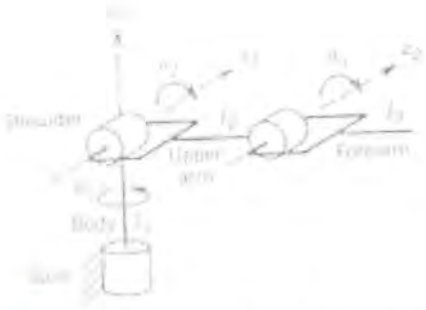
Pada tugas akhir ini akan dibuat robot dengan tipe *articulated* sehingga pembahasan akan lebih ditekankan pada jenis ini.

Articulated robot terdiri dari tiga anggota yang dihubungkan dengan dua buah *revoulute joint* dan ditempatkan pada suatu dasar yang bisa berputar seperti contoh pada gambar 2.6[6] . Bentuk pengaturannya memang menyerupai lengan manusia, sehingga sering disebut dengan lengan robot atau *elbow manipulator*.

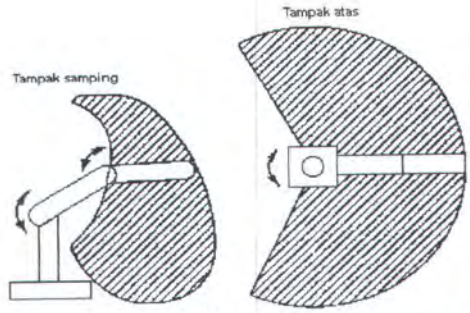
Karena *articulated* robot memiliki tiga buah sumbu putar maka resolusi spasialnya bergantung pada posisi lengan. Akibatnya akurasi robot jenis ini

---

[4] Ibid, p. 33.
[5] K, S, FV, R, C, Gozales, C, S, G, Lee, Robotics: Cotrol, Sensing, Vision, adnd Intelligence, McGraww-Hill, 1987, p. 3.
[6] Spong, Mark W dan Vidyasagar, M, Robot Dynamic and Control, John Wiley & Sons, Inc, 1989, p. 11.



Gambar 2.6 Struktur *Elbow Manipulator*



Gambar 2.7 Daerah Kerja Manipulator *Elbow*

kurang karena kesalahan posisi (*error position*) akan terakumulasi pada ujung lengan. Tetapi robot ini memiliki keuntungan kecepatan yang tinggi dan fleksibilitas yang baik. Dengan struktur *elbow* seperti ini maka daerah kerja dari robot ini adalah seperti pada gambar 2.7[7]. Dari sini akan dibuat robot berjenis *articulated* yang memiliki 3 derajat kebebasan (memiliki 3 *joint*) memakai *closed loop control* dan memakai sistem *Continuous Path.*

2.1.4 Trayektori Gerakan Robot (*Robotic Motion Trajectories*)

Dalam melakukan tugasnya, *end effector* bergerak dari suatu posisi dan orientasi tertentu menuju ke posisi dan orientasi tujuan secara spesifik. Pergerakan *end effctor* ini melewati suatu titik-titik lintasan yang dapat digambarkan dalam koordinat kartesian. Kumpulan titik lintasan yang memberikan jalur translasi dan rotasi dari *end effector* dalam fungsi waktu ini yang disebut dengan trayektori. Jalur *end effector* yang diwujudkan dalam koordinat kartesian akan cukup sulit untuk diwujudkan dalam *joint space.* Keduanya dapat dihubungkan dengan persamaan kinematik. *Forward Kinematic*

digunakan untuk mengubah dari *joint space* ke koordinat kartesian, sedangkan *Inverse Kinematic* digunakan untuk mengubah dari koordinat kartesian ke *joint space*.

Kinematika robot adalah analisa secara geometri, dari gerakan manipulator terhadap titik koordinat referensi dengan tidak memperhatikan gaya/momen yang menggerakan. Dalam analisa kinematika juga dipelajari cara mencari parameter *joint* dan *link*, perhitungan posisi dan orientasi manipulator, apabila diketahui posisi setiap sumbu gerak atau sambungan (*joint*) manipulator .

Informasi suatu urutan transformasi dapat disimpan di memori komputer induk dengan tiga cara sebagai berikut[8] :

- Cara pertama, Robot dituntun secara manual melalui trayektori yang menghubungkan berbagai trasformasi.
- Cara yang kedua adalah mendesain dan mengontrol *motion trajectory* secara manual dari *master control*.
- Cara yang ketiga adalah memberikan data numerik yang menunjukkan berbagai transformasi pada *software* yang menginstruksikan robot untuk bergerak sesuai urutannya.

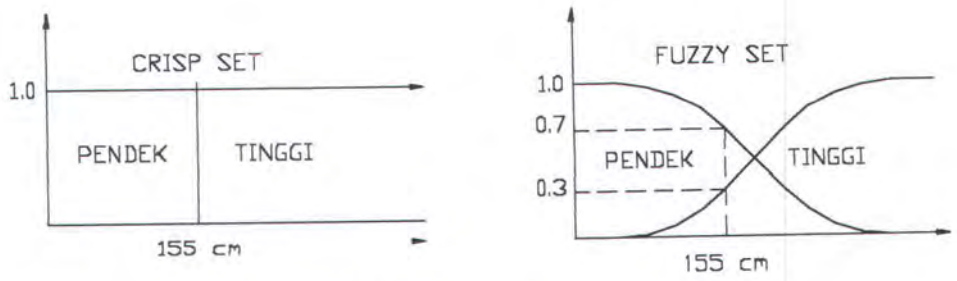
---

[7] Ibid, p. 11.
[8] Mohsen Shaninpoor, A ROBOT ENGINEERING TEXTBOOK, (New York: Harper &Row, 1987), p. 186.

2.2 TEORI LOGIKA FUZZY

Pengembangan logika fuzzy dilakukan untuk menjawab permasalahan *two valued logic* yaitu logika yang hanya mengenal suatu keadaan dalam 2 kemungkinan, seperti benar atau salah. Teori *two valued logic* terbukti efektif untuk memecahkan masalah dengan syarat permasalahannya dapat dideskripsikan kuantitasnya secara tepat.

Sebagai contoh adalah menentukan apakah orang dengan tinggi 160 cm termasuk tinggi, sedang atau pendek. Apabila menggunakan metode *two valued logic* maka pemecahannya adalah memberi range dibawah 155 cm sebagi pendek dan diatas 155 cm sebagai tinggi, kesulitan yang pertama adalah menentukan batas pendek dan tinggi secara kuantitatif, karena setiap orang mempunyai nilai yang berbeda. Kesulitan yang kedua adalah menentukan akan dimasukkan kemanakah nilai 155 cm tersebut (karena tepat berada pada perbatasan). Logika fuzzy membagi suatu keadaan dalam interval [0,1] yang secara intuitif dapat dinyatakan dalam contoh diatas dengan mengelompokkan orang sebagai pendek, agak pendek, sedang, agak tinggi, dan tinggi. Jika logika fuzzy ini diterapkan pada permasalahan tinggi badan diatas maka dapat dikatakan bahwa rang dengan tinggi badan 155 cm mempunyai nila kebenaran 0.7 pendek dan 0.3 tinggi. Hal ini akan memberikan kesimpulan bahwa orang tersebut cenderung pendek. Untuk lebih jelas, permasalahan tersebut dapat dilihat pada gambar 2.8.



Gambar 2.8 Fungsi keanggotan himpunan

Dalam logika fuzzy terdapat istilah-istilah sebagai berikut :

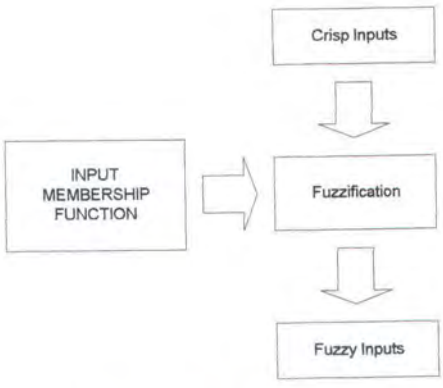
1. Fungsi keanggotaan *(membership function)* adalah fungsi yang memetakan masukan nyata *(chrisp input)* dengan domainnya pada derajat keanggotaan.
2. Skala keanggotaan / Derajat keanggotaan *(degree of membership)* merupakan skala dimana nilai nyata masukan setara dengan fungsi keanggotaannya. Skala ini bernilai antara 0 sampai 1.
3. Nilai masukan *(chrisp input)* adalah nilai masukan yang bernilai skalar, tertentu dan tunggal. Untuk sistem kontrol nilai ini dapat berupa masukan dari *tranducer*. Misalnya kecepatan kendaraan 100 km/jam.
4. Label adalah nama yang merupakan penggambaran dari fungsi keanggotaan.
5. Domain adalah lebar total dari fungsi keanggotaan yang merupakan jangkauan dari suatu konsep angka-angka tertentu, dimana fungsi keanggotaan itu dipetakan.
6. Nilai tengah merupakan suatu nilai dari fungsi keanggotaan dimana mempunyai nilai kebenaran sempurna (1 atau 0).

2.2.1 Proses Logika Fuzzy

Dalam pemecahan masalah menggunakan logika fuzzy diperlukan tiga tahap proses yaitu:

a. Proses *fuzzification* yaitu mengubah variabel input yang berupa variabel *crisp* (besaran nyata berupa variabel yang berorientasi numerik) ke dalam variabel fuzzy.
b. Proses evaluasi aturan (*rule evaluation*) yaitu mencari nilai aksi (*action*) dengan memberikan bobot pada setiap aturan yang diberikan.
c. Proses *defuzzification* yaitu mengubah variabel fuzzy yang terbentuk dari proses evaluasi aturan menjadi variabel *crisp*.

Proses *fuzzification* adalah rangkaian usaha untuk mentransformasi variabel input yang mulanya bersifat numerik (*crisp inputs*) ke variabel fuzzy (*fuzzy inputs*). Transformasi ini dipengaruhi oleh fungsi keanggotaan yang digunakan.



Gambar 2.9 Proses *Fuzzification*

Proses evaluasi aturan (*rule evaluation*) adalah rangkaian usaha untuk menentukan nilai aksi sebagai tanggapan atas setiap input atau kombinasi input yang diberikan dengan memberi bobot pada masing-masing aturan yang telah ditetapkan. Di dalam proses ini terdapat dua komponen utama yaitu himpunan aturan (*rule sets*) dan metode evaluasi aturan.

Himpunan aturan (*rule sets*) adalah semua aturan yang diperlukan untuk menentukan tanggapan terhadap input atau kombinasi input yang diberikan. Aturan ini bersifat linguistik dan mempunyai bentuk "jika ..... maka ....." (*If ..... then .....*). Bentuk aturan ini diciptakan berdasarkan keinginan untuk :

1. Menyediakan cara yang mudah bagi para ahli untuk mengekspresikan pengetahuan dan pengalaman mereka.
2. Menyediakan cara yang mudah bagi para desainer untuk menyusun dan memprogram aturan fuzzy.
3. Mengurangi biaya desain.

Bentuk umum dari aturan fuzzy adalah : jika x_1 adalah A_{k1} dan x_2 adalah A_{k2} atau x_3 adalah A_{k3} ...maka y_1 adalah B_{k1}; dimana x_1, x_2, dan x_3 adalah input kejadian 1 (*antecendent*1), kejadian 2, dan kejadian 3; A_{k1}, A_{k2}, dan A_{k3} adalah himpunan fuzzy yang berkorelasi dengan kejadian; y_1 adalah output kejadian dan B_{k1} adalah himpunan fuzzy yang berkorelasi dengan output.

Metode evalusi aturan adalah metode yang digunakan dalam mengevaluasi aturan yang telah ditetapkan. Ada beberapa metode evaluasi aturan yang sering dipakai seperti *Mini rule* (Mamdani), *Product rule* (Larsen), *Max-Min rule* (Zadeh), *Arithmatic rule* (Zadeh) dan *Boolean*. Metode yang digunakan dalam

tugas akhir ini adalah metode *Max-Min rule* sesuai dengan metode yang dipakai oleh mikrokontroler Fuzzy NLX 220. Konsep dari metode ini adalah mencari nilai minimum pada setiap *rule*, kemudian mencari nilai maksimum dari himpunan aturan yang berkorelasi dengan satu kejadian output sehingga dapat ditentukan nilai aksi yang seharusnya, demikian diulangi untuk setiap kejadian output. Nilai minimum pada setiap *rule* menggambarkan derajat ke-fuzzy-an aturan tersebut, sedangkan nilai maksimum dari nilai minimum himpunan aturan yang berkorelasi dengan satu kejadian output menggambarkan kejadian yang paling dapat "dipercaya" karena mempunyai derajat ke-fuzzy-an paling tinggi sehingga aturan yang mempunyai derajat ke-fuzzy-an paling tinggi diambil sebagai aturan yang paling dapat dipercaya (*the winning rule*).

Proses *defuzzification* adalah tahap terakhir proses logika fuzzy yang berfungsi untuk mengubah output yang berupa output fuzzy menjadi output *crisp*. Ada beberapa metode *defuzzification*, yaitu : *Center of Gravity* (COG), *Fuzzy sigleton*, *Accumulate*, dan *Immediate*. Dalam tugas akhir ini metode yang digunakan adalah metode *accumulate* dan *immediate* sesuai dengan spesifikasi NLX220 yang digunakan.

Metode *accumulate* pada output berarti nilai output sama dengan nilai aksi aturan yang menang ditambah dengan nilai output sebelumnya sehingga metode ini dapat digunakan sebagai pendekatan proses integrasi. Sedangkan metode *immediate* berarti nilai output sama dengan nilai aksi aturan yang menang.

2.2.2 Sistem Kontrol Dengan Logika Fuzzy

Tujuan utama suatu sistem kontrol adalah menghasilkan suatu keluaran yang dikehendaki untuk setiap masukan yang diberikan. Rangkaian usaha yang dilakukan untuk mengolah masukan menjadi keluaran yang dikehendaki disebut dengan proses kontrol. Secara konvensional dikenal beberapa proses kontrol yaitu metoda *look-up table*, metode pemodelan secara matematis untuk mencari fungsi transfer antara masukan dan keluaran, dan metode dengan menggunakan logika fuzzy.

Metoda kontrol berdasarkan logika fuzzy dikembangkan untuk mengatasi kelemahan-kelemahan pada metoda *look-up table* dan metode pemodelan secara matematis. Logika fuzzy menawarkan pemecahan masalah yang intuitif, dan disesuaikan dengan cara berfikir manusia. Penggunaan teknologi fuzzy dalam rekayasa proses dan sistem informasi akan menghasilkan alat-alat yang handal, tahan, luwes, dan lebih canggih dibandingkan dengan alat-alat digital biasa. Hal ini akan memproduksi sistem pengambil keputusan, sistem kontrol otomatis yang akan membawa kepada mesin yang mempunyai daya pikir *(intelegent machine)*.

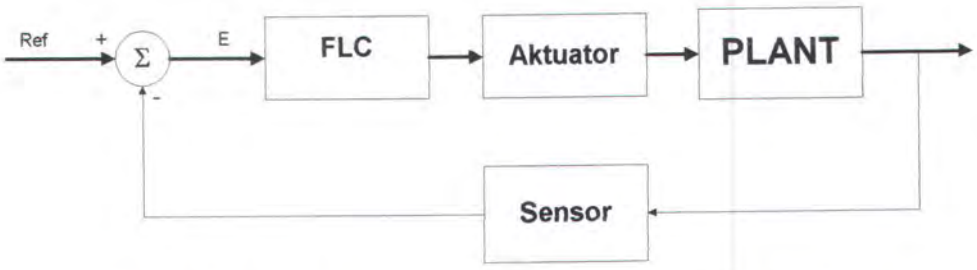
Keuntungan lebih lanjut dari penerapan logika fuzzy dalam kontrol akan mempunyai keuntungan-keuntungan sebagai berikut :

- Kemudahan bagi pemakai yang lebih baik.
- Kemampuan menyesuaikan diri yang lebih baik.
- Kemampuan untuk diagnosa sendiri.
- Kinerja yang lebih baik dengan konsumsi daya yang lebih rendah.

Struktur dasar penggunaan logika fuzzy untuk suatu sistem kontrol, ditunjukkan seperti seperti gambar 2.10.

Sistem kontrol otomatik pada umumnya terdiri dari empat bagian utama : sensor, kontroler, aktuator, dan *plant*. Sensor berfungsi sebagai pengambil data perilaku dari sistem. Aktuator memberikan daya untuk menggerakkan peralatan yang dikontrol (*plant*) agar mencapai suatu harga yang diinginkan.

Kontroler berfungsi memberikan perintah ke aktuator sesuai aksi kontrol menurut besarnya deviasi (*error*) yaitu selisih antara referensi dan output yang terukur oleh sensor. Sistem kontrol yang digunakan adalah sistem kontrol berumpan balik (lup tertutup) karena sistem ini cenderung menjaga hubungan yang telah ditentukan antara keluaran dan masukan acuan dengan membandingkannya dan menggunakan selisihnya sebagai alat pengontrolan, secara umum sistem kontrol berumpan balik mempunyai keunggulan dibanding sistem kontrol tanpa umpan balik (lup terbuka).



Gambar 2.10 Blok Diagram Sistem Kontrol Logika Fuzzy

2.3 *FUZZY MICROCONTROLLER* NLX220

Pada sub bab ini akan dibahas tentang kontroler fuzzy NLX220, yaitu: keistimewaan, fungsi pin-pin, dam cara penggunaan.

2.3.1 Keistimewaan NLX220[9]

Kontroler ini merupakan kontroler Logika Fuzzy (KLF) *stand-alone*, *single chip*, fleksibel, bekerja dengan eksternal ROM (EEPROM) atau *One time Program* (OTP), memiliki empat input analog 8-bit dan empat output analog 8-bit, 28 pin, menggunakan enam tipe *membership function*, 111 *variable fuzzy*, dan 50 *rule*. Kontroler ini dapat dipakai untuk *Power and Batery management*, pengendalian motor, pengendalian pemanas, pengendalian mobil dan kontrol proses industri.

2.3.2 Deskripsi

NLX220 memiliki unjuk kerja yang tinggi, dan merupakan kontroler logika fuzzy *stand alone*. Perangkat ini dapat melakukan operasi logika fuzzy secara langsung dalam *hardware*. NLX220 adalah kontroler yang memiliki keunggulan dalam hal mudah pemakaian, unjuk kerja, terencana dan *robust* atau kuat terhadap kondisi lingkungan operasi yang kasar. Perangkat ini berisi empat input analog 8-bit dan *generator clock internal*. NLX220 memakai daya yang kecil saat operasi normal dan mempunyai mode *power down* sehingga mengurangi daya dengan faktor 10. Metodologi fuzzy menggunakan penjelasan

bahasa yang relatif mirip dengan cara berpikir manusia, sehingga sederhana dalam pemakaiannya. Logika fuzzy dapat digunakan untuk menambah kemampuan pada berbagai produk. Logika fuzzy dapat memperbaiki unjuk kerja dan menaikkan efisiensi.

Memori menyimpan fungsi keanggotaan fuzzy dan parameter-parameter *rule*. Organisasi memorinya fleksibel dan efisien mengadaptasi terhadap berbagai keperluan aplikasi. Perangkat tersebut dapat menyimpan 111 variabel fuzzy, yang diorganisasikan menjadi *rule* seperti yang dikehendaki. Perangkat ini memiliki enam tipe fungsi keanggotaan yang berbeda untuk keperluan aplikasi. Fungsi keanggotaannya memiliki kemiringan yang konstan dan hanya perlu spesifikasi tipe, lebar atau *width* dan *center*. NLX220 memiliki fungsi keanggotaan *floating*. *Center* dan *width* dari fungsi keanggotaan dapat dibuat "mengambang" atau bervariasi secara dinamis. Fungsi keanggotaan *floating* dapat digunakan untuk mengukur *derivative*, membangun *timer* atau untuk mendrift sensor.

Ada dua metode defuzzifikasi yaitu *immediate* dan *accumulate*. Mode immediate akan menghasilkan nilai output dengan nilai yang spesifik. Mode *accumulate* menambahkan suatu nilai pada output sebelumnya.

Informasi aplikasi dimasukkan secara mudah menggunakan sistem *software INSIGHT*™ yang dapat dijalankan pada *Windows*™. Diperlukan sedikit pengetahuan tentang logika fuzzy untuk dapat menggunakan perangkat atau sistem tersebut.

---

[9] ---, Stand Alone Fuzzy Logic Controllers NLX220, Adaptive Logic, 1994.

2.3.3 Fungsi Pin-pin dari NLX220

- **Bagian Input:**

Reset

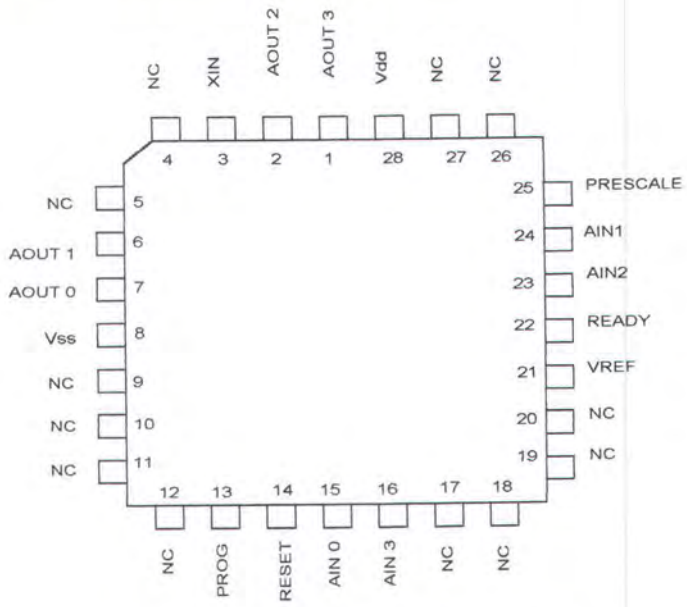
Sebuah signal aktif *low* akan menginisialisasi perangkat ini. **RESET** harus tetap aktif untuk paling sedikit delapan siklus *clock* untuk menjamin operasi yang benar. **RESET** dapat di*drive* oleh *power up delayed circuit.*

AIN(3:0)(Analog Input Data).

Data analog dirubah secara internal menjadi data digital 8-bit. Input-input yang tak terpakai harus dihubungkan ke *ground.*

Xin (*Clock Input*)

Dapat didrive oleh *clock external* atau oleh kristal. Sedang *lead* yang tidak dipakai dihubungkan ke *ground.*



Gambar 2.11 NLX220P dengan 28 pin

Prog

Pin ini digunakan untuk pemrograman NLX220P. Pin ini tidak digunakan pada NLX 220 sehingga pin ini dalam operasinya harus dihubungkan ke *ground.*

Prescale

Level logika satu menyebabkan perangkat ke mode *prescale* sedang *zero* menyebabkan operasi normal.. Pin ini dapat di*ground* jika mode *prescale* tidak dipakai atau dapat juga dihubungkan ke pin READY untuk pemakaian seterusnya. Mode juga dapat diminta selama operasi melalui logika eksternal. Setelah RESET tidak dipertahankan, pin *prescale* harus diberi logika rendah untuk selama paling sedikit empat siklus *clock.* Cara operasi *prescale* dijelaskan pada bab akhir dokumen ini.

- **Bagian Output:**

Aout(3:) Analog Output Data.

Data digital delapan bit dirubah secara internal menjadi sebuah level analog.

Ready

Setelah reset, pin ini akan menunjukkan bahwa NLX220 akan memulai mensampel dan memroses data. Pin ini sebaiknya tidak dihubungkan atau disambung ke PRESCALE selama operasi.

Vref

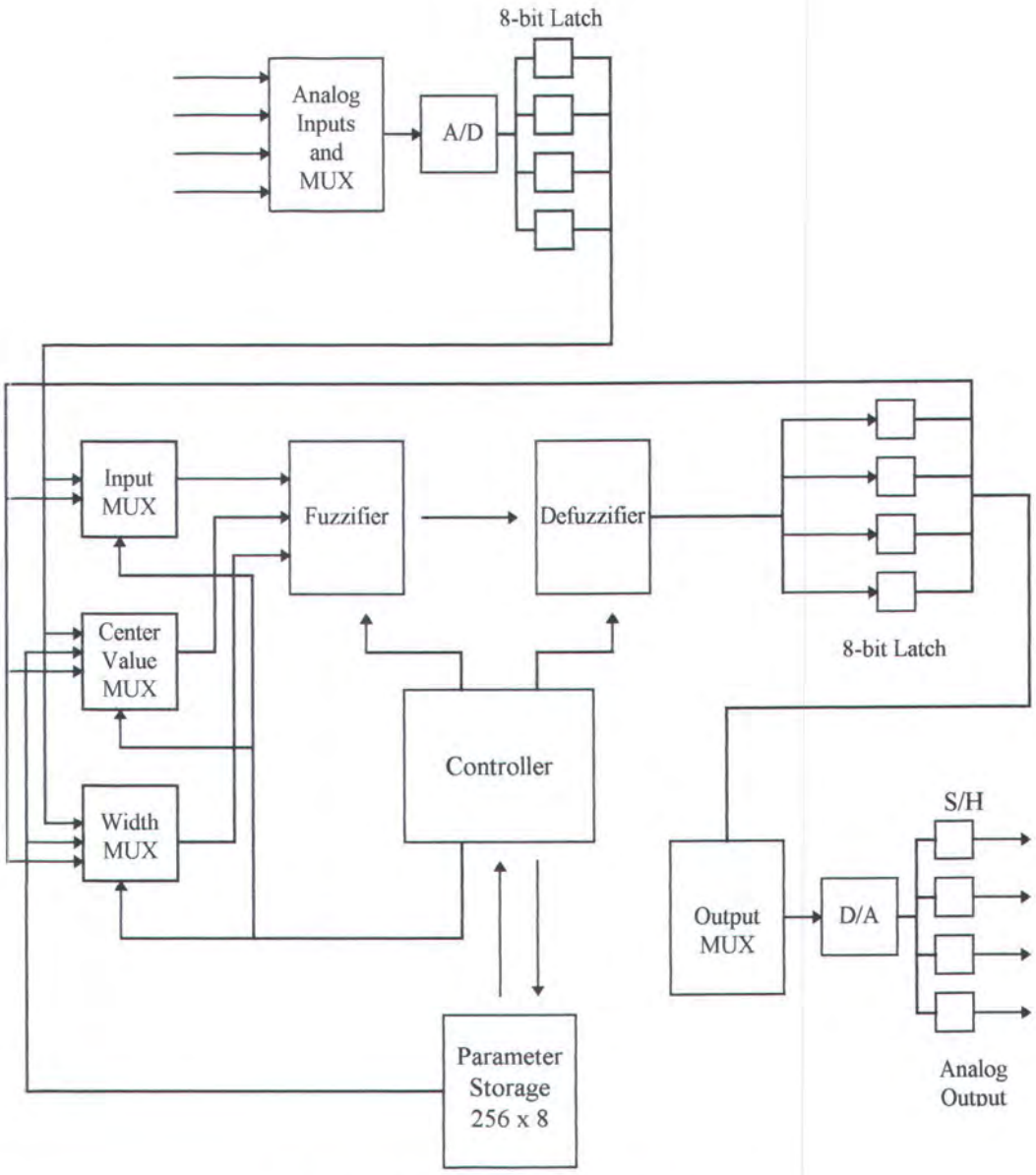
Filter tegangan *reference internal*, harus dihubungkan ke *ground* melalui kapasitor 0,1 μF.

2.3.4 Arsitektur Perangkat

Diagram NLX220 seperti ditunjukkan pada gambar 2.12 dibawah. Elemen-elemen utama adalah *Fuzzifier*, *Defuzzifier* dan Kontroler. *Fuzzifier* merubah data input menjadi data fuzzy. *Fuzzifier* yang berhubungan dengan kontroler adalah mengevaluasi data fuzzy melalui himpunan *rule* yang didefinisikan oleh pemakai untuk menjelaskan bagaimana sistem dikendalikan. Ketika *rule-rule* telah dievaluasi, *Defuzzifier* menghasilkan suatu nilai aksi output yang diperlukan.

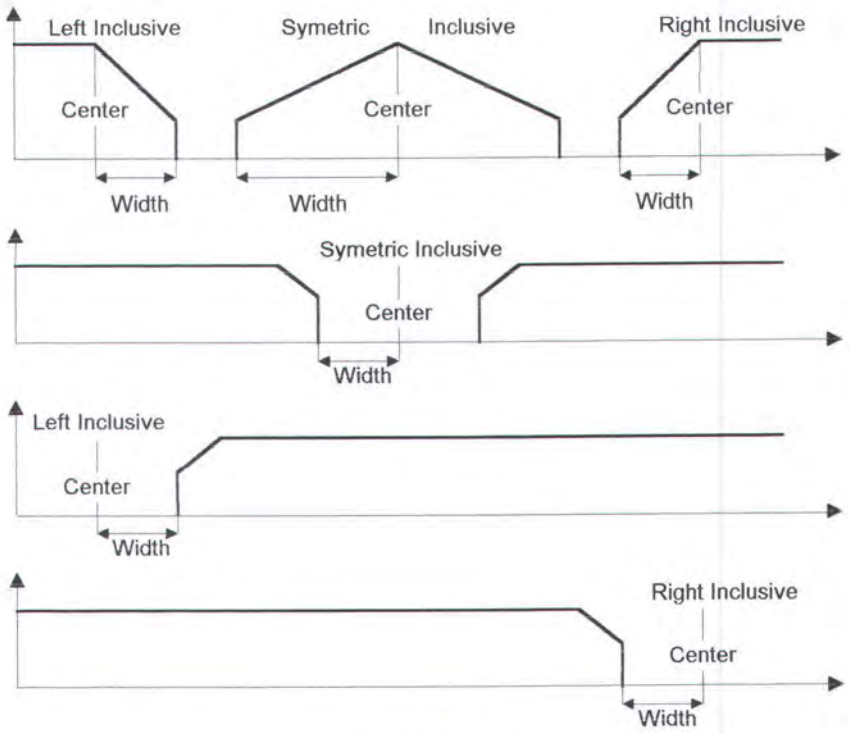
2.3.5 Fungsi Keanggotaan (*Membership Function*)

Fungsi keanggotaan digunakan untuk membagi-bagi input, sehingga input akan memiliki suatu range yang tertentu sesuai dengan apa yang diinginkan. Setiap input akan dibandingkan untuk mengetahui letaknya di dalam range yang telah didefinisikan. Input yang masuk akan memiliki suatu nilai fuzzy dimana semakin besar nilainya menunjukkan bahwa input tersebut merupakan anggota dari *membership function* itu, dan sebaliknya semakin kecil nilainya maka input tersebut bukan merupakan anggota dari range yang didefinisikan.



Gambar 2.12 Diagram Blok NLX220

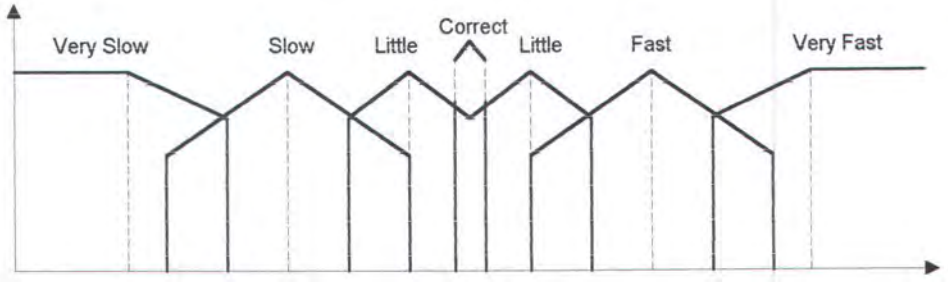
NLX220 memiliki enam fungsi keanggotaan yang berbeda dengan kemiringan yang konstan seperti gambar 2.13. Fungsi keanggotaan tersebut terdiri dari fungsi *Left inclusive, Symetrical inclusive* dan *Right Inclusive* serta fungsi-fungsi kebalikannya yaitu *Left exclusive, Symetrical exclusive* dan *Right exclusive*.



Gambar 2.13 Tipe Fungsi Keanggotaan

Dalam aplikasinya fungsi keangggotaan tersebut perlu diberi nama, tipe dan juga nilai *center* dan *width*. Perlu hati-hati dalam pemilihan fungsi keanggotaan didalam menyederhanakan berbagai macam model. Sebagai contoh, fungsi keanggotaan tunggal *Right* atau *Left inclusive* digunakan untuk meliput range nilai yang besar pada ujung-ujung daerah input.

Untuk pengontrolan yang teliti pada titik operasi yang diinginkan dapat dibuat fungsi keanggotaan *inclusive* simetrik yang sempit. Contoh aplikasi pengendalian motor, yang memerlukan ketelitian. Sebuah contoh dari campuran berbagai tipe yang berbeda lebar fungsi keanggotaannya digunakan untuk memo-nitor kecepatan motor seperti gambar 2.14.



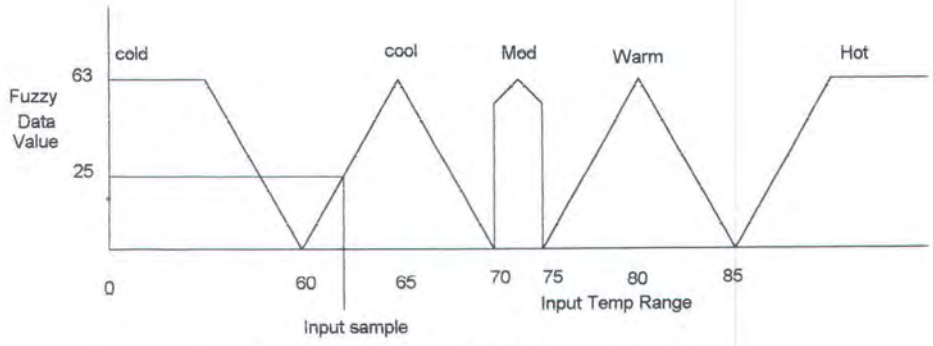
Gambar 2.14 Fungsi Keanggotaan Kecepatan

2.3.6. Variable Fuzzy

Variable fuzzy adalah pernyataan bahasa yang menggambarkan hubungan antara nilai input terhadap fungsi keanggotaan yang meliputi seluruh sumbu. Variabel fuzzy mereferensi sebuah fungsi keanggotaan dan sebuah nilai variabel input. Sebagai contoh sebuah variabel fuzzy sebagai berkut:

If Temperatur is Cool

Dalam contoh ini, 'Temperature' menunjukkan sebuah input dan 'Cool' adalah sebuah fungsi keanggotaan. Proses ini dilakukan oleh *Fuzzifier*. Hasilnya adalah sebuah nilai fuzzy yang menggambarkan derajat keanggotaan dari input terhadap fungsi keanggotaan. Nilai fuzzy adalah sebuah angka yang mempunyai range dari



Gambar 2.15 Fuzifikasi Temperatur Input

0 sampai 63 khususnya untuk NLX220. Gambar 2.15 merupakan contoh evaluasi variabel fuzzy, dimana terdapat suatu input kemudian nilai fuzzy-nya dilihat pada sumbu vertikal.

2.3.7 *Rule*

Rule terdiri dari satu atau lebih variabel fuzzy dan sebuah nilai aksi output. *Rule* digunakan untuk menggambarkan bagaimana kontroler harus bereaksi jika terjadi perubahan data input. Dalam sebuah contoh dibawah, kedua *Rule* berisi dua variabel fuzzy. *Rule* dimasukkan ke *software INSIGHT* dengan format sebagai berikut:

Output -5 If Velocity is Fast and Acceleration is Posistive

Output +5 If Velocity is Little- Slow and Acceleration is Zero.

Pada *rule* pertama, variabel fuzzy pertama adalah 'Velocity is Fast' dan variabel fuzzy kedua adalah 'Acceleration is Positive.' Aksi '+5' atau '-5' adalah nilai angka yang dapat dipakai pada output untuk memperlambat atau mempercepat motor. Dalam contoh dibawah, tanda + digunakan untuk menunjukkan bahwa output tersebut dapat bertambah atau berkurang jika menggunakan mode *accumulate.*

2.3.8 Evaluasi *Rule*

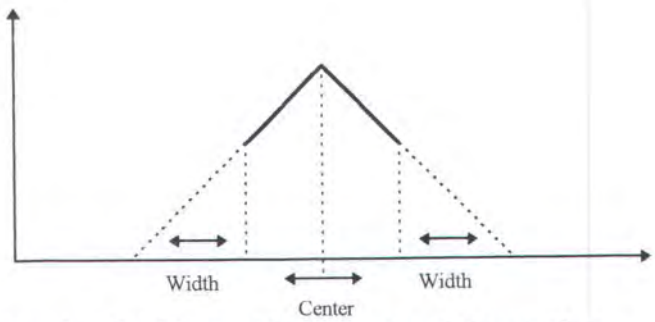
Ada beberapa metode untuk mengevaluasi *Rule* logika fuzzy. NLX220 mengevaluasi *Rule* menggunakan teknik dua langkah *MAX of MIN.* Langkah

pertama (MIN), semua nilai untuk variabel fuzzy dalam *Rule* dibandingkan dan nilai terendah akan mewakili nilai *Rule* tersebut. Pada langkah kedua (MAX), nilai satu *Rule* dibandingkan denga *Rule* yang lain dan *Rule* dengan nilai tertinggi yang menjadi pemenang dan nilai aksinya akan dikeluarkan pada outputnya.

Cara atau metode penentuan fungsi keanggotaan, variabel fuzzy dan *Rule-Rule* yang didefinisikan dan diorganisasi bergantung pada keperluan . Sifat fisik sistem yang dikendalikan harus benar-benar dimengerti sebelum memasukkan model fuzzy. Namun dengan banyaknya pengalaman maka memasukkan sebuah model akan lebih mudah.

2.3.9 *Floating Membership Function*

Rancangan unik dari NLX220 adalah fungsi keanggotaan *Floating* (mengambang). Seperti ditunjukkan gambar 2.16 di bawah ini, fungsi keanggotaan mengambang mempunyai nilai *Center* dan *Width* yang bervariasi secara dinamik. Fungsi keanggotaan mengambang nilai c*enter* atau *width*-nya berasal dari suatu input atau output.



Gambar 2.16 Fungsi Keanggotaan Mengambang

Suatu fungsi keanggotaan dapat dipilih sebagai *floating* pada saat memasukkan rancangan. Fungsi keanggotaan mengambang akan berubah nilai *center* dan nilai *width*nya sesuai dengan data dari nilai input atau outputnya. Sebagai contoh, dua variabel fuzzy dengan fungsi keanggotaannya yang didefinisikan secara konvensionil dengan menggunakan *Rule* sebagai berikut:

In1is small (0,25, Symetrical Inclusive)

In2 is small (0.25, Symetrical Inclusive)

dimana angka pertama adalah nol menunjukkan nilai *Center* dan yang kedua ,25 adalah nilai *Width*. Dua variabel tersebut dapat digabungkan menjadi sebuah *Rule* dengan menggunakan fungsi *floating* sebagai berikut:

IN1 is small_difference(IN2,25,Symetrical Inclusive)

Output +1 If IN1 is small_difference

Dimana variabel fuzzy 'IN1 is small' membandingkan input IN1 terhadap fungsi keanggotaan konvensionil 'small'. Fungsi keanggotaan mengambang dapat memberi penjelasan yang sama namun lebih ringkas.

Dalam variabel fuzzy, fungsi keanggotaan *center small-difference* didefinisikan oleh nilai IN2 yang disimpan dalam *latch input*. Dalam Fuzzifikasi, sebuah input dikurangkan dari *Center* dan hasilnya dirubah untuk mengukur nilai absolut seberapa dekat nilai tersebut terhadap *center*. Jika menggunakan sebuah fungsi keanggotaan *Floating Center* maka akan dikurangkan satu input dari yang lainnya. Fungsi keanggotaan *mengambang* memungkinkan menggunakan variabel fuzzy secara langsung mengukur perbedaan antara dua input. Teknik ini dapat digunakan untuk mengkalibrasi perubahan sebuah sensor. Nilai sensor yang

tetap dibandingkan terhadap himpunan tegangan. *Rule-Rule* kalibrasi mengecek derajat ketidaksetaraan dan menyimpan nilai koreksi di sebuah *latch output*. Jika inputnya dalam keadaan kalibrasi, *center* tersebut akan sesuai dan nilai koreksinya nol. Ketidaksetaraan yang besar akan menyimpan nilai koreksi yang besar. Koreksi digunakan untuk mengatur fungsi keanggotaan *Floating Center* dalam *Rule-Rule* yang memproses data. Fungsi keanggotaan *Floating* dapat digabungkan atau dikombinasi dengan nilai output aksi *Floating* untuk memperoleh derivatif dari sebuah nilai input. *Rule* dapat mereferensi sebuah input sehingga akan langsung dilewatkan ke output. Pada sampling input berikutnya, nilai *latch output* memilih nilai *Center*, yang mengurangkan nilai input sebelumnya dari nilai yang sekarang (*current value*). Perbedaan atau selisih itu dibagi-bagi melalui interval sampling sehingga merupakan nilai derivatif. Sebagai contoh menggunakan sebuah nilai aksi/input yang akan mengukur percepatan motor. Sebuah *Rule* yang menyimpan sebuah input ke *latch output* dapat ditulis sebagai berikut:

VALUE_T0 = IN1 If IN1 is MUST_WIN (0,0 Right Inclusive)

Rule akan menyimpan IN1 pada output. Fungsi keanggotaan MUST_WIN adalah tipe *Right Exclusive* yang dimulai dari nol sedemikian rupa dengan tidak memandang nilai IN1, *Rule* tersebut harus menang (*win*) dan nilai IN1 disimpan di *output latch*. *Rule* kedua menghitung derivatif dan mengatur output yang menggerakkan motor.

ACCELL + If IN1 is value_T1 (*value*-T0,25 *Symetrically inclusive*)

Rule ini menentukan apakah nilai input T1 berada tidak lebih dari 25 diukur dari nilai T0. Dalam aplikasi nyata, ada fungsi keanggotaan lain yang menentukan polaritas derivatif dan *Rule-Rule* lain untuk mengatasi pengaturan dengan variasi yang lebih besar. Contoh diatas adalah fungsi keanggotaan *floating* yang langsung (*strightfoward*). Dalam aplikasi nyata, fungsi keanggotaan *floating* dipakai secara intensif untuk menghemat memori karena hanya menggunakan beberapa variabel fuzzy dan *Rule-Rule* mendeteksi perbedaan antara input-input dibanding yang dilakukan fungsi konvensionil.

2.3.10 Operasi Perangkat

Pemrosesan data meliputi beberapa langkah. Pertama, data analog yang disampel dirubah menjadi digital dan ditahan (*latched*). Berikutnya *Fuzzifier* membandingkan isi *input latch* dengan variabel fuzzy untuk menemukan nilai variabel fuzzy. *Fuzzifier* juga melakukan kalkulasi MAX-of-MIN untuk menentukan *Rule* yang menang. Akhirnya, *Defuzzifier* menentukan nilai aksi *Rule* pemenang dan menahannya untuk mengkonversi menjadi output analog atau *feedback internal*.

2.3.10.1 *Fuzzifier*

Fuzzifier membandingkan data output yang ditahan dengan fungsi keanggotaan untuk menghitung nilai variabel fuzzy. Ketika hitungan MIN telah dilakukan terhadap semua variabel fuzzy dalam sebuah *Rule*, nilai yang mewakili *Rule* disimpan.. Ketika hitungan MAX telah dilakukan terhadap semua *Rule* yang

mereferensi sebuah output maka nilai aksi *Rule* pemenang dilewatkan ke *Defuzzier*.

2.3.10.2 Pembaharuan Data Output Yang Di*latch*

Rule-Rule untuk setiap output dievaluasi dan ditentukan pemenangnya. Setiap output dapat mengggunakan satu atau beberapa buah *Rule*. Ketika *Rule* atau sekelompok *Rule* mempengaruhi output yang telah dievaluasi dan *Rule* berikutnya dievaluasi untuk output yang lain, maka kompiler secara otomatis menyisipkan kode untuk *Rule* terakhir (Last *Rule*) menyebabkan output yang di*latch* diperbaharui oleh nilai aksi *Rule* pemenang. Data yang telah di*latch* dapat segera dipakai sebagai *internal feedback*.

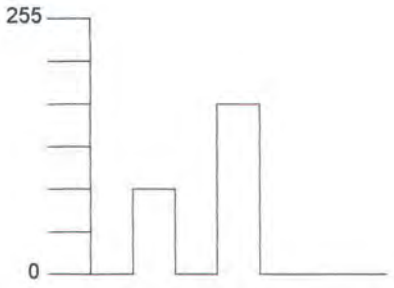
Sebuah data *latch* output kemudian dapat diperbaharui (di*update*) selama masih ada sekelompok *Rule* yang terpisah yang menggunakan output tersebut. Seperti yang telah disinggung sebelumnya, sampling input berlangsung kontinyu. Nilai output analog juga di *update* secara kontinyu. Selama siklus pemrosesan, variabel fuzzy dapat menggunakan sebuah sampel data dari siklus sampel sebelumnya atau dari siklus yang sedang berlangsung (*current*) bergantung dimana letak siklus sampling relatif terhadap siklus pemrosesan. Akan ada lebih dari sekelompok *Rule* yang mengggunakan input dan output yang sama, kemudian nilai output dapat berubah lebih dari sekali selama siklus pemrosesan berdasarkan data input yang berbeda.

2.3.10.3 *Defuzzifier*

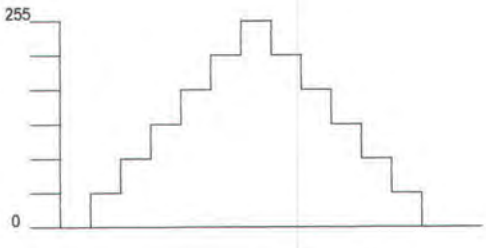
Pada bagian ini data output yang bernilai fuzzy diubah menjadi data *crips*. Nilai aksi *Rule* pemenang dan mode data dimasukkan ke blok *Defuzzifier*. Data digital dari *Defuzzifier* di *latch* dan dirubah menjadi analog untuk men*drive* output atau di inputkan kembali secara internal (*looped back internally*). Jika semua kelompok *Rule* menghasilkan sebuah output yang bernilai nol (*zero*), maka output tidak akan berubah nilainya. Jika lebih dari satu *Rule* menghasilkan suatu nilai bukan nol yang sama, maka *Rule* yang pertama masuk yang akan menang dan aksinya akan menentukan ouput.

Defuzzifikasi menyebabkan nilai aksi *Rule* pemenang menggerakkan sebuah output. Ada dua metode defuzzifikasi, *Immediate* dan *Accumalate*. Dua mode tersebut terlihat pada gambar 2.17 dan 2.18 dalam menyeleksi *Rule*. Fungsi mode *Immediate* seperti suatu susunan tabel, dimana nilai aksinya tertentu untuk sebuah *Rule*. Defuzzifikasi *Immediate* sangat berguna jika nilai output harus absolut.

Mode Akumulasi menambah atau mengurangi output yang ada oleh nilai aksi *Rule* pemenang. Output adalah fungsi aksi yang sedang berlangsung



Gambar 2.17 Defuzzifikasi *Immediate*.



Gambar 2.18 Defuzzifikasi *Accumulate*.

ditambah atau dikurangi output sebelumnya. Defuzzifikasi *Accumulate* dapat digunakan untuk perubahan halus pada output ketika sistem dikendalikan dekat titik operasi yang diinginkan. Hal ini juga berguna untuk fungsi-fungsi waktu.

2.3.11 Organisasi Memori

NLX 220 berisi 256 *byte* memori untuk aplikasi penyimpanan parameter. 32 *byte* terakhir menyimpan nilai fungsi keanggotaan tetap (*fixed member function*) *Center* dan *Width*. Sisanya 224 *byte* diorganisasikan sebagai satu atau lebih *Rule* dengan satu atau lebih variabel fuzzy per *Rule*. Setiap *Rule* memerlukan dua *byte*, plus dua *byte* tambahan untuk setiap variabel fuzzy dalam *Rule* itu. Sebuah *Rule* berisi lima variabel fuzzy untuk contoh akan menggunakan 12 *byte*. Memori diorganisasikan menjadi tiga seksi yang didefinisikan sebagai penyimpan variabel fuzzy/*Rule*, penyimpan *Center* dan penyimpan *Width*.

Tabel 2.1 Organisasi Memori

Alamat Desimal	Alamat Hexa	Fungsi
0	00	*Rule*
-----	------	------
223	DF	*Rule*
224	ED	Center
-----	-------	--------
239	EF	Center
240	FO	Width
-----	------	--------
255	FF	Width

2.3.11.1 *Rule* Dan Penyimpan Variable Fuzzy

Rule diorganisasikan sebagai sebuah atau lebih kelompok variabel-variabel fuzzy. Setiap variabel fuzzy terdiri dari dua *byte*, seperti dijelaskan dalam

tabel 2.2 dan 2.3. *Byte* pertama disimpan pada alamat genap dan yang kedua pada alamat ganjil. *Byte-byte* tersebut dibagi-bagi menjadi bidang-bidang yang mengendalikan bagaimana data diproses.

Tabel 2.2 Organisasi *Byte* Genap (*Command byte*) dalam NLX220

7	6	5	4	3	2	1	0
WF	CF	I/O CONT	I/O SELECT		TYPE 2-7		
AF	MODE				TYPE 1		
AF	MODE		OUTPUT SELECT		TYPE 0		
Type		210					
		000	Last term of Last *Rule* of given output				
		001	Last Term of Current *Rule*				
		010	MF, Symmetrical Inclusive				
		011	MF, Symmetrical Exclusive				
		100	MF, Left Inclusive				
		101	MF, Left Exclusive				
		110	MF, Right Inclusive				
		111	MF, Right Exclusive				
I/O Select		43					
		00	I/O Port 0 as input				
		01	I/O Port 1 as input				
		10	I/O Port 2 as input				
		11	I/O Port 3 as input				
I/O Control		5					
		0	Select from inputs				
		1	Select from outputs				
Mode		6					
		0	Immediate, Output equals Action				
		1	Accumulate, Output equals current output plus two's complement action (-128 to 128)				
AF		7					
		0	Select Action from select byte (fixed)				
		1	Select Action from I/O via select byte (float)				
Output Select		43					
		00	ACTION from current *RULE* set to Output 0				
		01	ACTION from current *RULE* set to Output 1				
		10	ACTION from current *RULE* set to Output 2				
		11	ACTION from current *RULE* set to Output 3				
CF		6					
		0	Select Center from Memory via Select byte (fixed)				
		1	Select Center from I/O via select byte (float)				
WF		7					
		0	Select Width from Memory via Select byte (fixed)				
		1	Select Width from I/O via Select byte (fixed)				

Tiga bit LSB untuk *byte* genap mendefinisikan tipe fungsi keanggotaan atau apakah variabel fuzzy sebelumnya adalah *Rule* terakhir atau variabel fuzzy terakhir dari *Rule* terakhir yang mereferensi output. Ketika bagian *least*

significant memilih tipe fungsi keanggotaan, lima bit-bit MSB dibagi-bagi menjadi tiga bagian bit yang memilih sumber satu input dari empat pin input atau *latch output*. Dua sisa bit-bit MSB mendefinisikan baik itu *Center* dan *Width* dari fungsi keanggotaan *floating* atau *fixed*. Tipe kode signal-signal variabel fuzzy terakhir (001) adalah variabel fuzzy terakhir dari *Rule* yang telah diproses. Ketika hal ini terjadi, hanya dua bit-bit MSB dari lima bagian bit yang digunakan. Bit-bit MSB memilih apakah nilai aksi yang berasal dari lokasi *memory fixed* atau dari *latch* I/O. Bit-bit MSB berikutnya memilih mode output apakah *Immediate* atau *Accumulate*. Kode (000) menunjukkan variabel fuzzy terakhir dari *Rule* terakhir. Dua bit-bit MSB digunakan seperti dijelaskan pada paragraf diatas. Suatu tambahan , dua bit diatas bagian pemilihan tipe digunakan untuk memilih output.

Byte kedua selalu terjadi pada alamat ganjil dan berisi alamat *Center* dan alamat *Width* bagian indeks jika *byte* sebelumnya dipilih sebagai tipe fungsi keanggotaan dan nilai *Center* atau *Width Fixed*. Jika salah satu baik itu *Center* atau *Width* dipilih sebagai *floating*, maka mereka mengambil *byte* ganjil digunakan untuk memilih input atau output. Ketika tipe *byte* pertama adalah variabel fuzzy terakhir atau variabel fuzzy terakhir dari *Rule* terakhir dan aksinya adalah *fixed*, maka *byte* kedua mengandung nilai aksi. Jika aksinya adalah *floating*, maka *byte* ganjil memilih input atau output yang memberikan nilai aksi.

Tabel 2.3 Organisasi *Byte* Ganjil (*Select Byte*) dalam NLX220

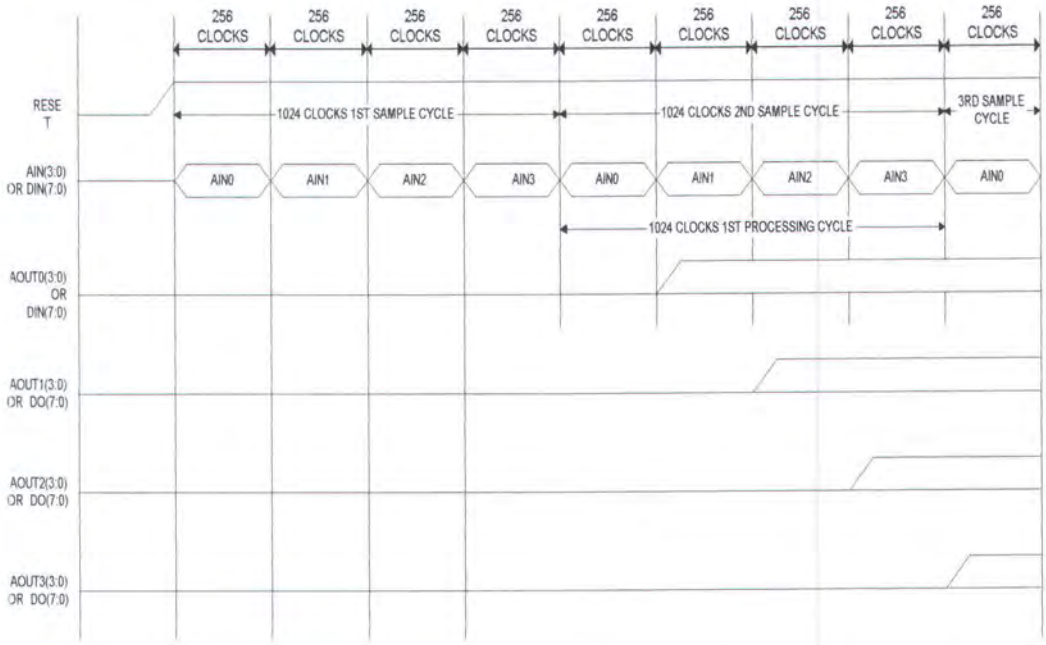
7	6	5	4	3	2	1	0	
CENTER SELECT				WIDTH SELECT				TYPE = 2 - 7 CF or WF = 0 (FIXED)
	I/O CONT	I/O SELECT CENTER			I/O CONT	I/O SELECT WIDTH		TYPE = 2 - 7 CF or WF = 1 (FLOAT)
ACTION								TYPE = 0 - 1 AF = 0 (FIXED)
					I/O CONT	I/O SELECT ACTION		TYPE = 0 - 1 AF = 1 (FLOAT)

Width Select	(3:0)	Used as Address index (E0-EF) for Fixed 6-bit WIDTH value when type = 2 - 7 and WF = 0
Center Select	(7:4)	Used as Address index (F0-FF) for Fixed 8-bit CENTER value when type = 2 - 7 and CF = 0
I/O Select Width	10	
	00	I/O Port 0 as Width (Type = 2 - 7 and WF = 1)
	01	I/O Port 1 as Width (Type = 2 - 7 and WF = 1)
	10	I/O Port 2 as Width (Type = 2 - 7 and WF = 1)
	11	I/O Port 3 as Width (Type = 2 - 7 and WF = 1)
I/O Control	2	
	0	Select from Inputs (Type = 2 - 7 and WF = 1)
	1	Select from Outputs (Type = 2 - 7 and WF = 1)
I/O Select Center	54	
	00	I/O Port 0 as Input (Type = 2 - 7 and CF = 1)
	01	I/O Port 1 as Input (Type = 2 - 7 and CF = 1)
	10	I/O Port 2 as Input (Type = 2 - 7 and CF = 1)
	11	I/O Port 3 as Input (Type = 2 - 7 and CF = 1)
I/ O Control	6	
	0	Select from Inputs (Type = 2 - 7 and CF = 1)
	1	Select from Outputs (Type = 2 - 7 and CF = 1)
ACTION	7:0	8-bit action value to be applied to an output due to winning Last Term of a *Rule* (Type = 1) or Last Term of last *Rule* of given output (Type = 0), and AF = 0 (Fixed)
I/O Select Action	10	
	00	I/O Port 0 as Action (Type = 1 - 0 and AF = 1)
	01	I/O Port 1 as Action (Type = 1 - 0 and AF = 1)
	10	I/O Port 2 as Action (Type = 1 - 0 and AF = 1)
	11	I/O Port 3 as Action (Type = 1 - 0 and AF = 1)
I/O Control	2	
	0	Select from Inputs (Type = 1 - 0 and AF = 1)
	1	Select from Outputs (Type = 1 - 0 and AF = 1)

2.3.12 *Timing* (pewaktuan)

Gambar 2.19 menunjukkan pewaktuan NLX 220. Ada tiga blok untuk pewaktuan meliputi pemultiplekan konverter A/D input, kontroler fuzzy dan pemultiplekan konverter D/A output. Kecepatan pemrosessan adalah fungsi kecepatan *clock* dan banyaknya *clock* (1024) yang diperlukan untuk

penyamplingan data secara lengkap dan siklus pemrosesan. Kecepatan maksimum *clock* adalah 10 Mhz dan minimum 1 MHz.



Gambar 2.19 Pewaktuan I/O

2.3.13 Implementasi Alogaritma Proporsional Derivatif Pada NLX220

Respon aksi kontrol kontroler PD konvensional adalah penjumlahan antara komponen proporsional dan komponen derivatif. Komponen proporsional konvensional adalah aksi kontrol proporsional terhadap nilai *error* (selisih antara nilai yang diharapkan dengan nilai aktual), sedangkan komponen derivatif adalah aksi kontrol sebanding dengan laju perubahan sinyal *error* ($\Delta error/\Delta$waktu). Aksi kontrol PD dalam persamaan dinyatakan sebagai[10] :

[10] Ogata Kashuhiko, TEKNIK KONTROL AUTOMATIK, (Jakarta: Erlangga, 1993), p.158.

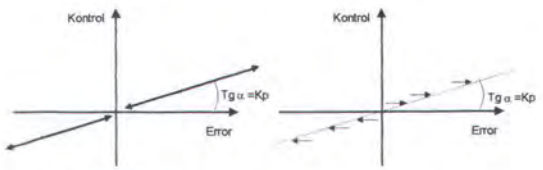
$$m(t) = K_P e(t) + K_P T_d \frac{de(t)}{dt} \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (2.1)$$

dimana m(t)= Aksi kontrol PD

K_P = adalah konstanta kepekaan proporsional, dapat diatur

K_d = adalah konstanta kepekaan turunan, dapat diatur.

Pendekatan bentuk proporsional konvensional dalam bentuk fuzzy dapat dilihat pada gambar 2.20. Kemiringan/*sloop* dari garis hubungan antara nilai *error* aksi kontrol adalah adalah konstanta kepekaan proporsional (gambar 2.20.a). Pada pendekatan dalam bentuk fuzzy, pertambahan aksi kontrol sebanding dengan pertambahan *error* (gambar 2.20.b). Jika *error* bertambah besar dan positif, maka aksi kontrol harus diperbesar positif pula, agar kondisi *plant* bergerak mendekati nilai set. Demikian pula sebaliknya, jika *error* bertambah besar negatif maka aksi kontrol harus dikoreksi membesar negatif agar *plant* begerak mendekati nilai set.



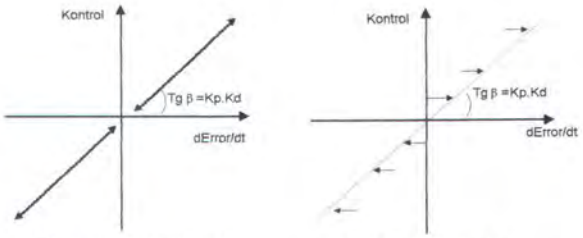
(a) Kontroler Proporsional dalam sistemkontrol analog

(b) Pendekatan Kontroler Proporsional Dalam Sistem Kontrol Logika Fuzzy

Gambar 2.20 Perbandingan Kontroler Proporsional Konvensional dengan Fuzzy

Pendekatan bentuk derivatif dari sistem analog konvensional dapat dilihat pada gambar 2.21. Pendekatannya mirip dengan pendekatan pada bentuk

proporsional, yaitu dengan menambahkan konstanta aksi pada varibel kontrol fuzzy yang berhubungan dengan selisih *error* terhadap *error* sebelumnya. *Error*



(a) Kontroler Derivativ pada Sistem Kontrol Analog

(b) Pendekatan Kontroler Derivativ dalam Sistem Kontrol Logika Fuzzy

Gambar 2.21 Perbandingan Kontroler Derivatif Konvensional dengan Fuzzy

pada saat [t+1] dikurangi *error* pada saat [t]. Hal ini dapat dilakukan dengan membuat fungsi keanggotaan *error* yang mempunyai *center floating* terhadap tundaan sinyal *error* (*error* lalu/pada saat [t]).

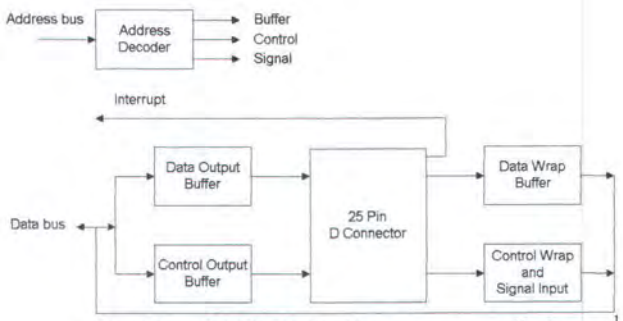
Kemudian kedua unsur diatas dijumlahkan sehingga terbentuk suatu sistem kontrol pendekatan PD menggunakan logika fuzzy. Keuntungan metode PD menggunakan logika fuzzy selain desainnya mudah dan cepat, juga sifat penentuan nilai aksi berlaku lokal pada bagian komponen tertentu saja sehingga dapat mengatasi permasalahan yang non linier. Beda pada metode konvensional, jika mengubah besarnya konstanta, maka seluruh karakteristik sistem berubah. Kesulitan yang dialami jika menggunakan variabel fuzzy adalah menentukan batasan *error* (kecil, besar, sedang , dll), untuk mengatasi hal ini diperlukan pengalaman yang cukup dari perancang mengenai respon *plant*.

2.4 *PARALLEL PORT* LPT1

Port paralel disebut juga *Adapter Parallel*, *Adapter* adalah peralatan yang dapat menghubungkan komputer dengan unit di luar komputer. Biasanya setiap PC memiliki sebuah port paralel, tetapi bisa ditambahkan menjadi LPT1 dan LPT2.

Pada umumnya port paralel LPT dihubungkan dengan printer. Kecuali digunakan untuk mengontrol kerja printer, penggunaan port paralel sebagai sarana komunikasi dan proses transfer data (input/output) hampir terabaikan. Hal ini disebabkan karena pada dasarnya jalur data yang ada pada port paralel LPT hanya berfungsi sebagai jalur data yang bersifat satu arah (*unidirectional*), yaitu sebagai jalur data output. Blok diagram dari port paralel adalah seperti gambar 2.22.

Pada tugas akhir ini, port paralel printer LPT1 akan dimanfaatkan sebagai *supervisor*, yaitu untuk komunikasi menampilkan data hasil proses kontroler fuzzy pada IBM-PC sekaligus IBM-PC digunakan untuk memori tempat menyimpan informasi titik-titik lintasan yang harus dilalui oleh lengan robot.



Gambar 2.22 Blok diagram *parallel port*[11]

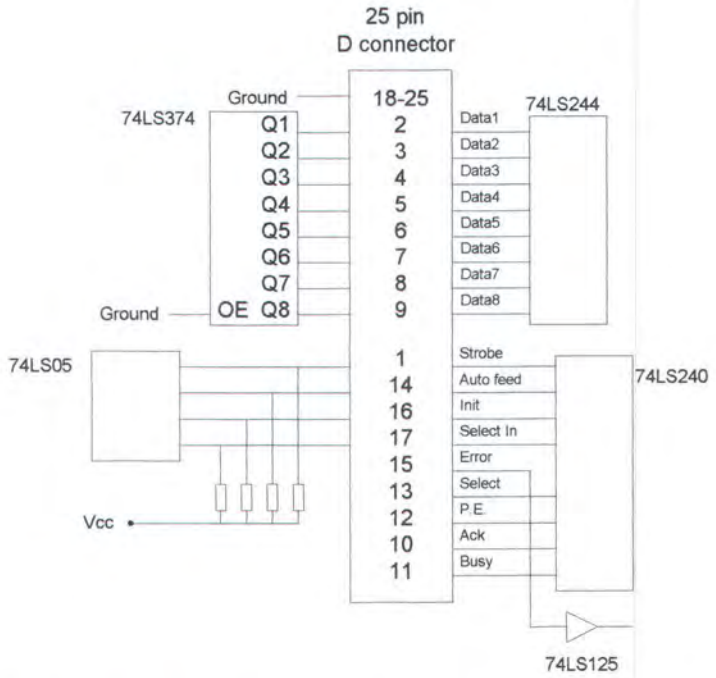
[11] ---, "IBM PC AT Technical Reference", IBM, 1984, p. 20

2.4.1 Konfigurasi Pin

Port paralel printer LPT yang terdapat pada komputer IBM PC, terdiri dari sebuah konektor DB-25 pin dengan 17 jalur sinyal yang terbagi atas 3 kelompok jalur sinyal dan 8 jalur yang terhubung dengan *ground*. Adapun 17 jalur sinyal yang terdapat pada port paralel LPT dan terhubung melalui konektor DB-25 tersebut adalah :

1. Jalur sinyal kontrol (4 jalur)
2. Jalur sinyal status (5 jalur)
3. Jalur sinyal data (8 jalur)

Untuk lebih jelasnya, dapat dilihat fungsi dari masing-masing pin seperti gambar 2.23



Gambar 2.23 Sinyal input dan output dari *parallel port*[12]

[12] Ibid, p. 25-27

Jalur kontrol digunakan sebagai jalur kontrol *interface* dan proses *handshaking* sinyal dari PC ke printer. Jalur status digunakan untuk sinyal *handshake* dan sebagai indikator status, seperti kertas habis *(paper-end)* dan sibuk *(busy)*. Jalur data (D0 - D7) digunakan untuk memindahkan data dari PC ke printer.

2.4.2. Alamat Memori

Pada saat kita menghubungkan komputer IBM PC dengan suatu peralatan luar (*peripheral*) melalui port paralel LPT, maka pertama kali kita harus mengetahui alamat memori yang digunakan oleh port paralel LPT. Ada 3 macam alamat memori *(base address)* yang digunakan oleh port printer pada komputer yaitu[13] :

1. 956 atau 3BCh, yaitu alamat port printer dengan MBA Card
2. 888 atau 378h, yaitu alamat port printer dengan CGA Card
3. 632 atau 278h, yaitu alamat port printer 3ed

Sedangkan pendefinisian alamat *(base address)* pada pin-pin port paralel LPT dapat dilihat pada tabel 2.4. Alamat-alamat yang terlihat pada tabel 2.4, kesemuanya harus didefinisikan dan dikontrol secara *software* pada saat kita akan mengakses pin yang bersangkutan. Agar lebih jelas, misalnya digunakan *base address* 378h, maka penggunaan memori ini akan mengatur fungsi-fungsi sebagaimana seperti gambar 2.23 dapat dijelaskan sebagai berikut:

---

[13] Paul Bergsman. "Microcomputer Journal", January 1995, p. 30

Tabel 2.4 Alamat Memori Pin Pada *Port Printer*

Pin	Alamat Memori (Base Address)
1	D0, Base + 2, Bit 1
2	D0, Base Address, Bit 1
3	D1, Base Address, Bit 2
4	D2, Base Address, Bit 3
5	D3, Base Address, Bit 4
6	D4, Base Address, Bit 5
7	D5, Base Address, Bit 6
8	D6, Base Address, Bit 7
9	D7, Base Address, Bit 8
10	D6, Base + 1, Bit 7
11	D7, Base + 1, Bit 8
12	D5, Base + 1, Bit 6
13	D4, Base + 1, Bit 5
14	D1, Base + 2, Bit 2
15	D3, Base + 1, Bit 4
16	D2, Base + 2, Bit 3
17	D3, Base + 2, Bit 4

- *Port 378h* berupa IC 74LS378 (output) dan IC 74LS244 (input). IC 74LS374 mampu memberi logika '1' (source) sampai 2.6 mA dan logika '0' (*sink*) sampai 24 mA. Port ini sebenarnya bisa sebagai input maupun output, tetapi karena pin OE disambung permanen ke *ground*, maka port ini tidak boleh sebagai input. Port hanya boleh mengirim sinyal ke luar. Karena ada IC 74LS244, port bisa dibaca oleh oleh komputer. Dalam hal ini komputer membaca output 74LS374, jadi komputer membaca dirinya sendiri.
- *Port 379h* hanya bertugas sebagai input. Normalnya port ini membaca status printer, yaitu sinyal *Error*, SLCT, *Paper End*, *Acknowledge* dan *Busy*. Jadi hanya data bit ke 3 sampai bit ke 7 saja yang dipakai.
- *Port 37Ah* berupa IC 74LS05 (*output open collector*) dan IC 74LS240 (input), dimana output 74LS05 berhubungan dengan input 74LS240. IC 74LS05 mampu memberi logika *Low* sampai 7 mA (dengan R *pull-up*

sebesar 4.7KΩ[14]) . Karena output dari 74LS05 bersifat *open collector*, maka port ini dapat bertindak sebagai input maupun output. Agar tidak merusak sinyal dari luar, terlebih dahulu dikirim data 0 ke alamat ini dan diterima sebagai data 1 sehingga ketika diberikan data luar 0 atau 1 maka data tersebut mengikuti data luar. Khusus untuk bit 2 memakai dua buah *not open collector*, sehingga dikeluarkan dengan data 1. Jadi untuk menjadikan port ini sebagai input terlebih dahulu dikeluarkan data 0000 0100 b atau 4h ke port ini.

2.5 MOTOR ARUS SEARAH (DC)

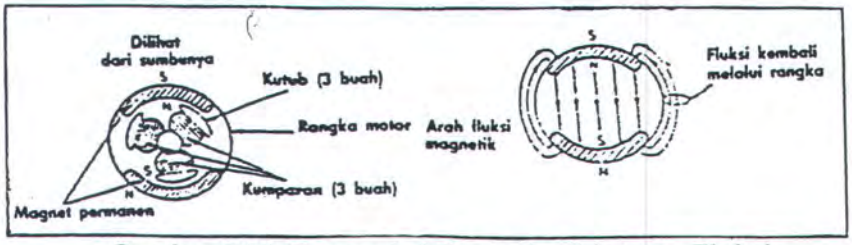
Dalam bidang industri dan rumah tangga motor listrik arus searah (DC) digunakan secara luas, mulai dari motor mikro yang sangat kecil sampai motor berukuran ribuan daya kuda.

2.5.1 Prinsip kerja

Meskipun penerapanya sangat luas, semua motor DC pada prinsipnya adalah segulung kawat yang dialiri arus listrik dan ditempatkan dalam suatu medan magnet. Akibatnya gulungan kawat ini akan mengalami suatu gaya yang sebanding dengan kekuatan medan magnetnya. Arah gaya membentuk sudut siku terhadap arus dan arah medan magnet. Arah gaya ini akan terbalik jika arah arus atau medan magnetnya dibalik, sedangkan arah gaya tidak akan berubah bila arah arus dan medan magnet keduanya dibalikkan.

---

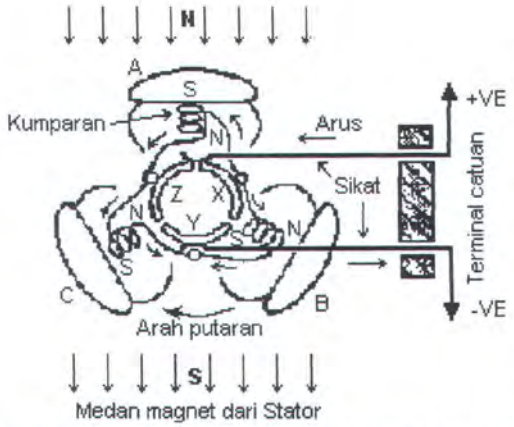
[14] ---, opcit, hal. 26



Gambar 2.24 Konstruksi Rotor dan Lintasan Fluksi

Dilihat dari penghasil medan magnetnya motor DC dibagi menjadi motor elektromagnet (medan magnet dihasilkan oleh kumparan) dan motor magnet permanen. Pada tugas akhi ini digunakan motor DC dengan magnet permanen (PM, *permanent magnet*). Di dalam motor PM medan magnet dihasilkan oleh satu atau beberapa magnet permanen. Magnet-magnet ini digenggam oleh penggenggam besi, baja, atau rangka motor itu sendiri. Magnet ini merupakan bagian motor yang diam, dan disebut stator. Kawat yang mengalirkan arus listrik digulung pada bagian motor yang berputar dan bagian yang berputar ini disebut rotator.

Susunan rotor yang paling sederhana, biasanya digunakan pada motor-motor yang kecil dan murah, rotor dibuat menjadi tiga buah kutub kumparan yang dibuat dari logam berlapis. Gambar 2.24 memperlihatkan konstruksi motor kecil dan lintasan fluksi magnetnya. Medan magnet dihasilkan oleh dua buah magnet serbuk besi yang dibentuk mengikuti bagian dalam rangka motor yang dibuat dari plat besi. Kedua magnet ini bersama-sama mengimbaskan medan magnet, sehingga dihasilkan medan magnet yang kuat pada tengah motor. Fluksi magnet menempuh suatu lintasan tertutup melalui rangka motor seperti pada gambar.



Gambar 2.25 Prinsip Kerja Motor DC PM

Mengalirnya arus di dalam suatu kumparan akan menimbulkan gaya yang menggerakkan rotor.

Arus dialirkan ke kumparan rotor melalui dua buah sikat (*brush*) yang sekaligus merupakan kontak dengan cincin penghantar pada rotor. Cincin ini, yang disebut komutator, dibagi menjadi tiga bagian yang disusun berdekatan. Kedua sikat akan memindahkan arus dari satu kumparan ke kumparan lainnya sesuai dengan putaran motor. Pada gambar 2.25 diperlihatkan cara kerja motor.

2.5.2 Karakteristik

Gambar 2.26 adalah kurva karakteristik sebuah motor kecil 1 Watt. Kurva tersebut memperlihatkan torsi (T) dengan tegangan tetap dalam hubungannya dengan nilai-nilai : kecepatan (N), efisiensi ($\cap$), daya (P) dan arus (I).

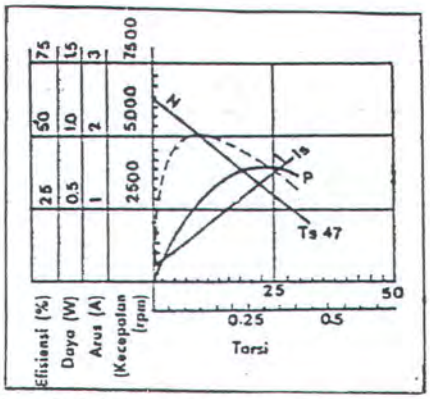
Hubungan antara arus dan torsi membentuk sebuah garis lurus. Tanpa menghiraukan tegangan yang digunakan atau kecepatan putarannya, perbandingan T/I adalah tetap untuk setiap motor. Ini adalah fungsi dari jumlah lilitan motor,

kekuatan magnet, jenis dan jumlah besi pada rotor dan stator, serta celah udara di antara rotor dan stator. Tentu saja masih ada faktor lain yang membuat nilai T/I ini tetap.

Hal kedua yang dapat diperoleh dari kurva ini adalah hubungan antara kecepatan dengan torsi. Kurva ini juga merupakan garis lurus yang mmperlihatkan hubungan antara kecepatan tanpa beban (ketika T = 0) dengan torsi diam (T_S), dimana motor dalam keadaan diam.

Untuk menjelaskan karakteristik tersebut, pertama-tama dilihat pengaruh tegangan yang digunakan dan resistansi rotor. Rangkaian pengganti suatu motor dapat digambarkan seperti gambar 2.27.a Terlihat kumparan rotor terhubung seri dengan sebuah resistor yang menggambarkan resistansi efektif kumparan yang terukur pada terminal-terminalnya. Tegangan yang diberikan ke motor adalah jumlah dari tegangan pada rotor (V_A) dan jumlah tegangan jatuh pada resistansi kumparan (V_R).

Tegangan V_A disebut juga tegangan GGL balik. Gaya gerak listrik (GGL)

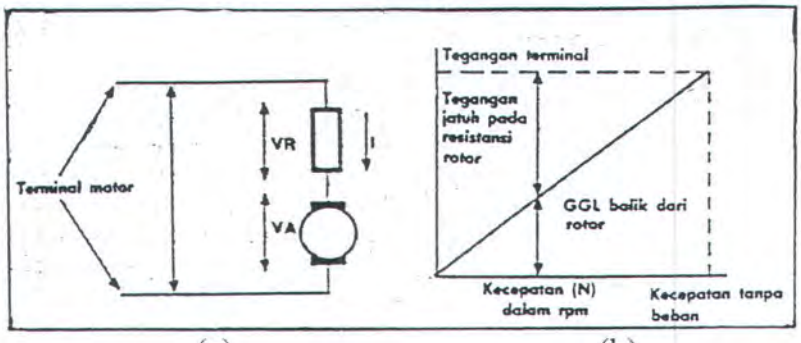


Gambar 2.26 Karakteristik Motor 1 watt

balik ini adalah tegangan yang dibangkitkan pada kumparan rotor pada saat rotor berputar sesuai dengan hukum elektromagnetik. Tegangan yang dibangkitkan ini berlawanan arah dengan tegangan yang diberikan, dan besarnya tergantung pada kecepatan putar rotor (N). Pada setiap tegangan V_A dan V_R harus sama dengan tegangan yang diberikan, yaitu V.

Pada saat awal, ketika motor dalam keadaan diam, kecepatan motor adalah nol, sehingga V_A dan V_R juga sama dengan nol. Karena motor tidak berputar, maka arus pada saat ini disebut sebagai arus diam I_S yang menimbulkan torsi diam T_S. Dengan menganggap bahwa motor dapat berputar tanpa hambatan, maka motor akan mengalami percepatan dan tegangan V_A akan bertambah sebanding dengan kecepatannya, juga V_R yang merupakan selisih antara V_A dan V akan mulai berkurang. Arus I yang dinyatakan sebagai V_R/R akan semakin mengecil juga. Proses ini akan terus belangsung sampai dicapai suatu titik stabil dimana torsi motor sebanding dengan keperluan beban.

Bagi motor yang berputar bebas tanpa beban beban yang ada hanya



(a) (b)

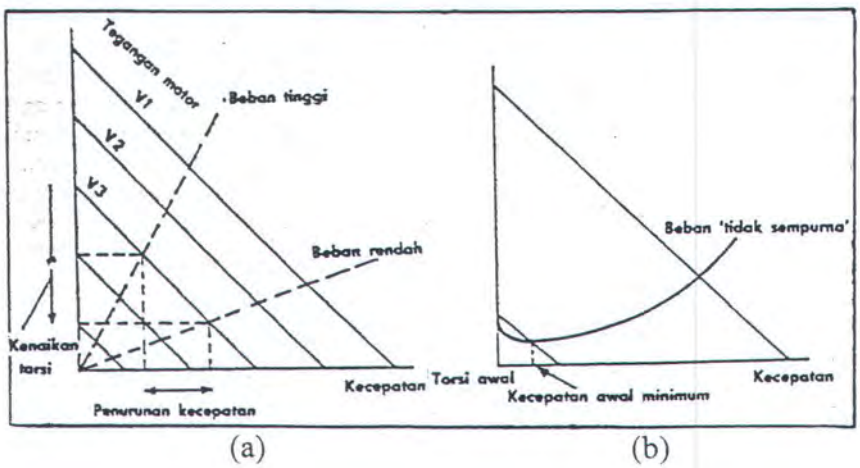
Gambar 2.27.a Rangkaian Pengganti Motor
Gambar 2.27.b Hubungan Tegangan Jatuh Terhadap Kecepatan

gesekan pada bantalan dan tahanan udara, sehingga V_A harganya mendekati harga V, sehingga nilai V_R, I dan T akan kecil . Hal ini ditunjukan pada gambar 2.27.b. Tegangan pada terminal motor adalah jumlah antara V_A dan V_R dimana V_A berbanding lurus dengan kecepatan. Oleh karena itu arus motor I dan juga torsinya T akan berbanding lurus dengan V_R.

2.5.3 Pengaturan Kecepatan Motor DC

Sistem pengaturan motor yang paling sederhana hanya menggunakan resistor seri yang diseri pada motor. Pada pengaturan ini mempunyai kelemahan bahwa pada pengaturan torsi yang lebih rendah, kecepatan motor berubah-ubah dengan kasar, sesuai dengan perubahan beban. Metode pengaturan seperti ini memiliki banyak keterbatasan dan selain kesederhanaannya, keuntungan yang diberikan hampir tidak ada.

Pengaturan kecepatan berikutnya adalah pengaturan tegangan variabel.



(a) (b)

Gambar 2.28.a Kurva Untuk Beban Sempurna
Gambar 2.28.b Kurva Untuk Beban Tidak Sempurna

Gambar 2.28.a memperlihatkan kurva performa motor untuk beberapa harga tegangan. V1 adalah tegangan terminal standar. Pada tegangan yang lebih rendah, torsi diam dan kecepatan tanpa beban menurun. Jika kedua garis beban motor digambarkan, maka perbaikan performa akan segera terlihat ketika torsi dinaikkan disertai dengan sedikit penurunan kecepatan. Gejala ini hampir mendekati karakteristik ideal, dimana kenaikan torsi yang besar seharusnya tidak mengubah kecepatan.

Terdapat dua keuntungan lain dari metoda pengaturan tegangan ini. Pertama, kecepatan motor memiliki hubungan linear dengan tegangan yang diberikan. Hal ini sangat berguna jika motor digunakan sebagai sistem penggerak sederhana. Keuntungan kedua adalah performa pada kecepatan rendahnya yang lebih baik (lihat gambar 2.28.b). Di samping itu, kecepatan awal minimum bagi motor lebih kecil.

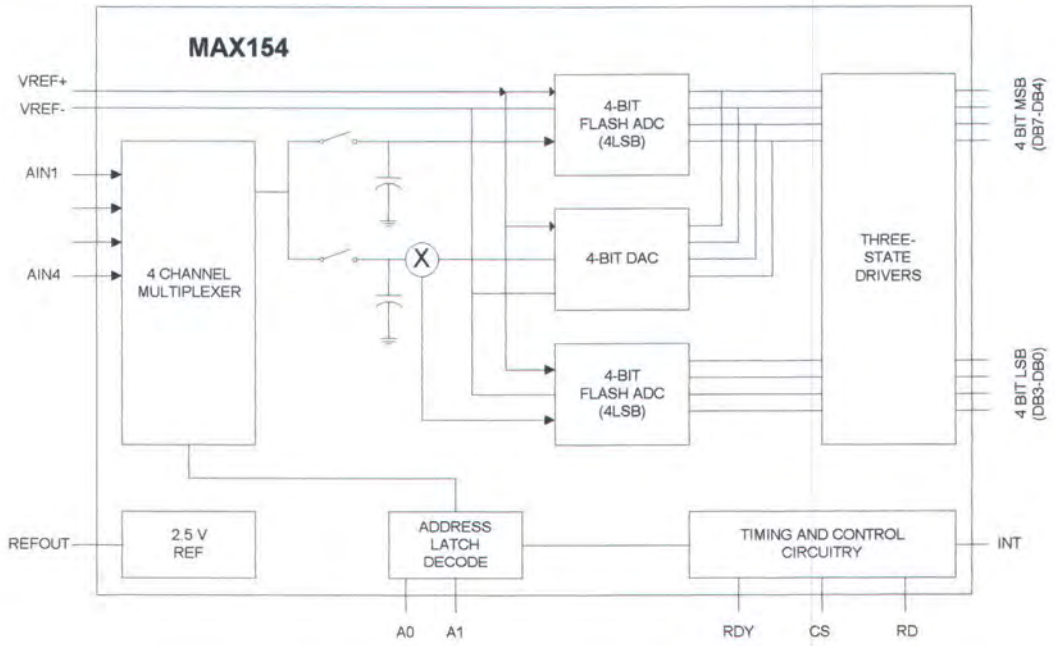
Metode ini mempunyai kelemahan, yaitu pada kecepatan rendah, tegangan yang diberikan ke motor hanya sedikit. Hal ini dapat menjadi masalah jika hubungan antara sikat dengan komutator kotor, berkarat, atau aus. Pada beberapa keadaan, motor tidak dapat berputar akibat kontak yang jelek ini. Hal ini kontras sekali dengan metoda pengaturan resistor seri, dimana tegangan catu akan selalu muncul seluruhnya meskipun pada kecepatan rendah.

Kedua metoda pengaturan diatas pada umumnya sudah cukup memadai untuk mengatur motor secara sederhana.

2.6 *ANALOG TO DIGITAL CONVERTER* MAX154

MAX154 adalah konverter dari analog ke digital dengan empat *channel* masukan analog, kecepatan tinggi, dilengkapi dengan pembangkit tegangan referensi dan rangkaian *sample-and-hold* internal. Kecepatan konversi dari ADC ini adalah 2,5 μs yang dapat dicapai karena penggunaan teknik konversi "*half-flash*". Diagram fungsional internal MAX154 dapat dilihat pada gambar 2.29.

Teknik konversi "*half-flash*" menggunakan 2 flash-ADC 4-bit untuk memperoleh hasil 8-bit. Dengan menggunakan 15 buah komparator, 4-bit atas (MSB) dari *flash*-ADC membandingkan tegangan masukan yang tidak diketahui dengan referensi dan menghasilkan empat bit atas. DAC internal menggunakan bit-bit MS untuk membangkitkan sinyal analog dari konversi yang pertama. Tegangan sisa yang mewakili perbedaan antara tegangan masukan yang tidak diketahui dengan tegangan dari DAC kemudian dibandingkan dengan tegangan



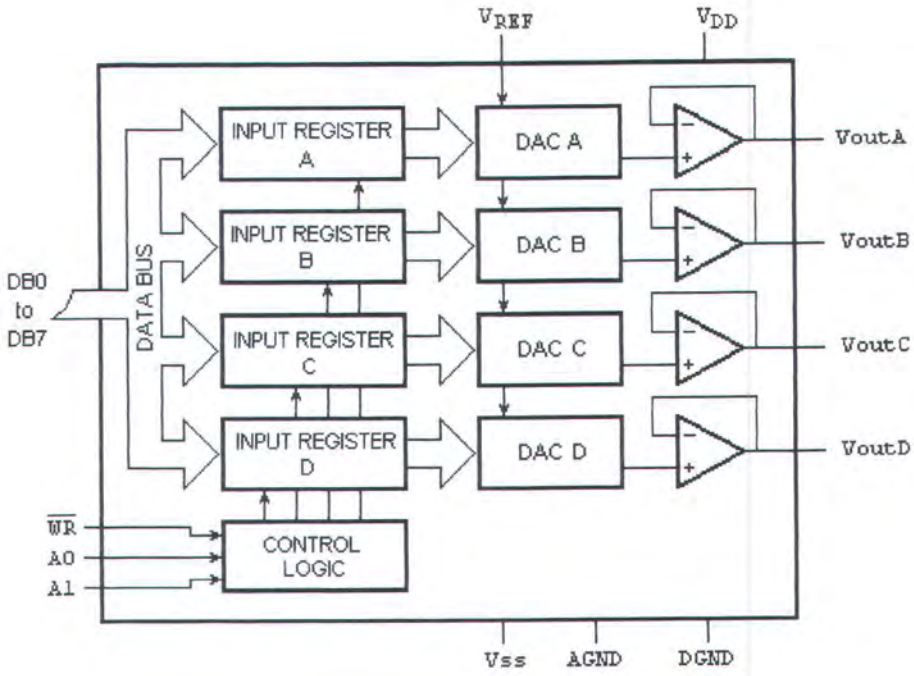
Gambar 2.29 Diagram Fungsional MAX154

referensi melalui 15 pembanding cepat pada bit-bit bawah (LSB) untuk memperoleh hasil 4 bit bawah. Operasi dari MAX154 dimulai dengan jatuhnya sinyal CS dan RD, kemudian komparator masukan membandingkan tegangan masukan dengan referensi selama waktu sekitar 1μs. Setelah siklus pertama hasil dari MS di*latch* ke *buffer* keluaran dan hal ini memulai konversi untuk LS. Sinyal INT berubah menjadi rendah sekitar 600ns kemudian menandakan akhir konversi, dan empat bit bawah di*latch* ke *buffer* keluaran. Data dapat diakses dengan menggunakan sinyal RD dan CS.

Interface digital dengan mikroprosesor dengan MAX154 dipermudah dengan adanya kemampuan ADC ini untuk berfungsi sebagai lokasi memori eksternal atau port I/O tanpa rangkaian logika eksternal. Data keluaran menggunakan rangkaian *buffer three-state*, dan di*latch*. Hal ini mengizinkan hubungan langsung dengan data bus mikroprosesor atau port I/O sistem.

2.7 *DIGITAL TO ANALOG CONVERTER* MX7226

MX7226 adalah konverter digital ke analog delapan bit dengan empat kanal keluaran tegangan, dilengkapi dengan *amplifier* penyangga keluaran dan masukan digital yang mudah bagi sistem *interfacing* prosesor atau TTL/CMOS yang sederhana. Secara internal masukan *logic* disangga dengan dua *buffer*, sehingga semua kanal tegangan output dapat di*update* dengan menggunakan sinyal kontrol secara simultan. Diagram fungsional DAC ini pada gambar 2.30.



Gambar 2.30 Diagram Fungsional MX7226

MX7226 mempunyai empat buah DAC jaringan *R-2R ladder* yang akan mengkonversi 8 bit data digital ke tegangan analog yang ekivalen dan proporsional terhadap tegangan referensi yang dikenakan. Tegangan referensi menentukan ouput skala penuh DAC. Pada MX7226, semua input tegangan referensi (V_{REF}) pada DAC-DAC nya dijadikan satu. Yang perlu diperhatikan Impedansi input minimum pada V_{REF} adalah 2 kΩ, terjadi pada saat input digital 01010101. Nilai maksimumnya tidak berhingga, terjadi saat input digital 00000000. Karena impedansi input pada V_{REF} bergantung pada input kode digital, maka impedansi output sumber tegangan untuk V_{REF} tidak boleh lebih dari 4Ω untuk menjamin linearitas ouput analog. Tegangan referensi yang disarankan adalah antara 2 volt sampai (V_{DD} - 4 volt) untuk menjamin kecukupan konversi dan *buffering* tegangan output.

Tegangan output V_{outA}, V_{outB}, V_{outC} dan V_{outD} adalah sebagi berikut:

$$V_{out} = N_B \times \frac{V_{REF}}{256}$$

dimana N_B adalah nilai numerik input kode biner pada input DAC. Semua output tegangan pada MX7226 dibuffer dengan *unity gain follower* yang presisi dengan harga *slew rate* 3V/μs. Line *address* A0 dan A1 digunakan untuk memilih DAC mana yang akan diberikan input digital. Ketika $\overline{WR}$ *low* register input DAC yang dialamati transparan. Data akan di-*latch* di register tersebut ketika $\overline{WR}$ menjadi *high*.

MX7226 dispesifikasikan untuk beroperasi pada V_{DD} antara +11,4 volt sampai +16,5 volt dan V_{SS} pada 0 volt sampai -5 volt. DAC ini dapat juga dioperasikan pada *singgle supply* (V_{SS} = 0 V), meskipun demikian *zerro code error* dapat dikurangi pada V_{SS} = -5V. Sinyal transien digital atau AC antara pin *analog ground* (AGND) dan *digital ground* (DGND) dapat menyebabkan *noise* pada output analog. Disarankan AGND dan DGND disatukan dan dihubungkan pada *ground* dengan kualitas yang terbaik.

BAB III

PERANCANGAN DAN PEMBUATAN PERANGKAT KERAS DAN PERANGKAT LUNAK

BAB III
PERANCANGAN DAN PEMBUATAN PERANGKAT KERAS DAN PERANGKAT LUNAK

3.1. PERANCANGAN SISTEM

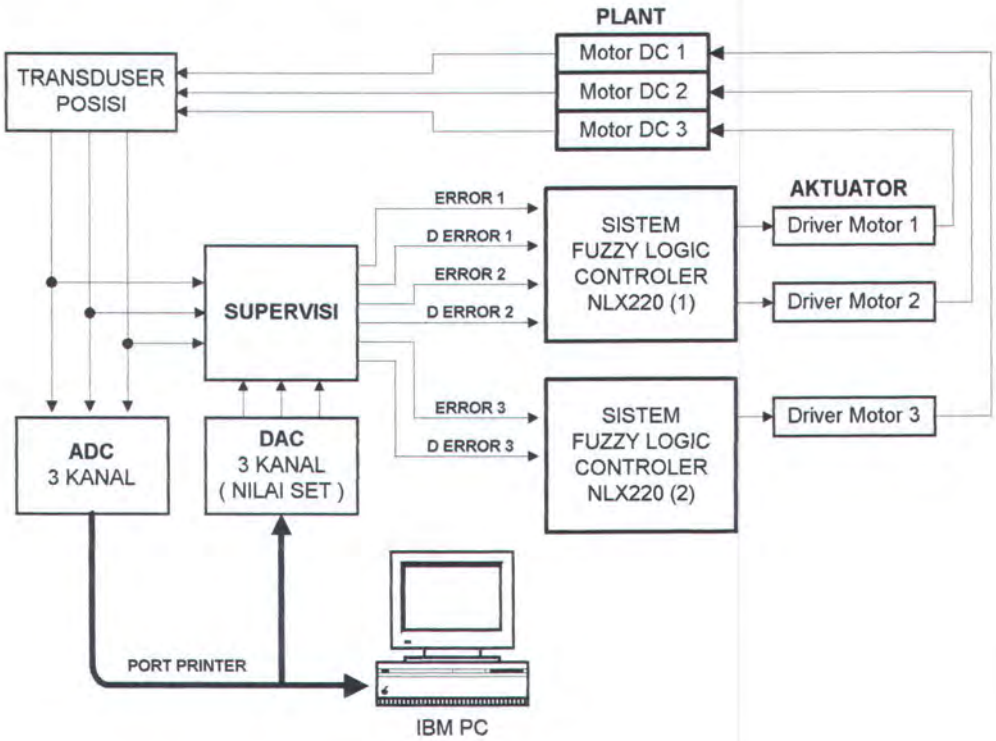
Sistem lengan robot tiga derajad kebebasan yang dibuat akan mempunyai kriteria-kriteria perangkat keras sebagai berikut :

- Mekanik yang dirancang adalah lengan robot dengan struktur *elbow* dengan tiga bagian utama yaitu *base*, *body* dan *upper arm*. Masing-masing bagian digerakan oleh motor DC kecil.
- Untuk menentukan titik-titik yang akan dilewati/dituju oleh *end effector* lengan robot, lengan robot dituntun oleh pengguna, dan setiap titik yang dilewati disimpan dalam memori/file dalam PC.
- Fungsi PC dalam hal ini adalah untuk menyimpan titik-titik yang akan dilewati oleh lengan robot (sebagai memori) sekaligus menampilkan posisi sudut tiga buah *joint* pada lengan robot. *Interface* antara kontroler lengan robot dan PC melalui *Parallel Port* LPT1.
- Dalam menempuh suatu lintasan lengan robot digerakan oleh kontroler menggunakan logika fuzzy (*fuzzy rule based control system*). Kontroler ini menggunakan kontroler fuzzy logic NLX220.

Sedangkan perangkat lunak dari sistem ini akan mempunyai spesifikasi sebagai berikut :

- Perangkat lunak dalam kontroler logika fuzzy, yang merupakan sebuah sistem berbasis aturan fuzzy.
- Perangkat lunak dalam komputer, berfungsi sebagai pemberi nilai yang dikehendaki (*setting point*) pada saat lengan robot dituntun oleh pengguna, sebagai memori untuk menyimpan titik-titik yang akan dilintasi oleh lengan robot dan menampilkan posisi sudut tiga buah *joint* pada lengan robot.

Diagram blok perangkat keras sistem lengan robot tiga derajad kebebasan berbasis kontroler logika fuzzy adalah seperti pada gambar 3.1



Gambar 3.1 Perancangan Blok Diagram Sistem

Secara umum, alat ini mempunyai cara kerja sebagai berikut :

Setelah daya dinyalakan, tranduser mulai menyampling posisi sudut tiga buah *joint* lengan robot. Dari PC diberikan tegangan *setting* awal, dimana tegangan tersebut sebanding dengan nilai posisi sudut awal (posisi *default*). Tegangan *setting* ini diberikan PC melalui DAC yang di*interface*kan pada LPT1 PC.

Rangkaian supervisi memproses sinyal dari tranduser dan tegangan *setting*, sehingga diperoleh tegangan yang merupakan perbedaan antara tegangan dari tranduser dan tegangan *setting*, tegangan ini adalah tegangan *error*.

Selain tegangan *error*, pada rangkaian supervisi terdapat rangkaian yang menunda suatu sinyal sebesar waktu tertentu t, keluaran dari rangkaian ini adalah tegangan *Delay Error* (D Error).

Kedua macam tegangan ini (*Error* dan *Delay Error*) dimasukkan ke dalam prosesor Fuzzy, untuk selanjutnya diolah oleh *rule-rule* fuzzy, yang selanjutnya hasil keluaran proses fuzzy, diumpankan ke aktuator (*driver* motor DC) untuk mengubah-ubah posisi sudut masing-masing *joint* pada lengan robot.

Pada perangkat lunak PC terdapat dua buah mode untuk menjalankan lengan robot pada lintasannya. Mode pertama adalah mode MANUAL, yaitu lengan robot dituntun satu persatu oleh pengguna untuk dijalankan. Pada mode ini dapat disimpan posisi sudut tiga buah *joint* lengan, kemudian disimpan pada memori/*file* di PC. Mode kedua adalah mode RUN, dimana lengan robot berjalan

sendiri dengan di*drive* kontroler fuzzy berdasarkan data posisi sudut yang akan dituju yang ada pada memori/*file* di PC.

Sedangkan fungsi dari masing-masing blok adalah sebagai berikut :

1. Tranduser berfungsi sebagai pengubah besaran posisi sudut tiga buah *joint* lengan robot besaran elektris.
2. Unit supervisi merupakan unit penghasil *sinyal error* dan *delay error* dari sinyal masukan.
3. Unit fuzzy *logic controller* NLX220 berfungsi sebagai pemroses data dengan menggunakan aturan-aturan fuzzy serta mengontrol keluaran men*drive plant*. Kontroler fuzzy akan menggerakan *plant* secara cepat jika posisi yang dituju masih jauh dan memperlambat kecepatan jika posisi sudah dekat. Kontroler akan menghentikan gerakan *plant* jika sudah sampai pada posisi yang dituju dan mempertahankan posisi tersebut.
4. Aktuator adalah unit untuk menggerakan *plant* yang berupa motor DC. Fungsi dari aktuator adalah mem*buffer* sinyal dari kontroler fuzzy agar cukup kuat untuk menggerakan *plant*.
5. ADC berfungsi sebagai pengubah sinyal analog dari tranduser posisi untuk ditampilkan pada layar monitor PC.
6. DAC berfungsi untuk memberikan nilai set pada kontroler fuzzy, nilai set ini sebanding dengan nilai posisi sudut yang harus dituju oleh masing-masing *joint* lengan robot.

3.2 PERANCANGAN PERANGKAT KERAS

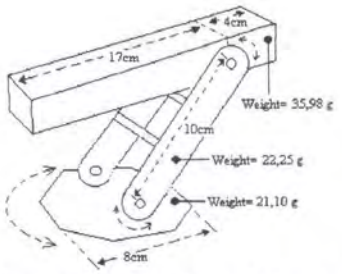
Perancangan perangkat keras terdiri dari perangkat mekanis dan elektronis. Bagian elektronis terdiri dari rangkaian driver motor DC (aktuator), rangkaian sensor posisi, rangkaian supervisi yang terdiri dari penghasil error dan delay error, serta ADC-DAC LPT1.

3.2.1 Mekanis

Mekanis yang dirancang adalah lengan robot tiga derajad kebebasan model *elbow* yang dibuat dari bahan akrilik. Lengan robot terdiri dari tiga bagian, yaitu *base*, *body* dan *arm* yang masing-masing bagian dihubungkan secara langsung dengan motor DC. Rancangan lengan robot ini pada gambar 3.2.

Motor DC yang dipakai adalah jenis *indirect drive*, mempunyai roda gigi reduksi internal yang berfungsi untuk menaikan torsi. Konsekuensi dari roda gigi reduksi ini adalah kecepatan motor menurun, namun cukup memadai untuk menggerakan lengan robot. Jangkah gerakan putar motor ini motor ini juga dibatasi secara internal, mampu bergerak dengan jangkah 180^{o}.

Tegangan catu maksimum motor DC ini adalah 6 volt. Pada pengukuran tanpa beban, didapat bahwa dengan tegangan catu 5 volt didapat arus motor sebesar 57 mA. Nilai ini yang akan dipakai untuk menentukan torsi dan kecepatan gerak *plant* maksimum, dimana untuk mencapai torsi tersebut nilai aksi dari aktuator yang didrive oleh kontroler fuzzy harus mampu menyediakan arus dan tegangan seperti diatas.



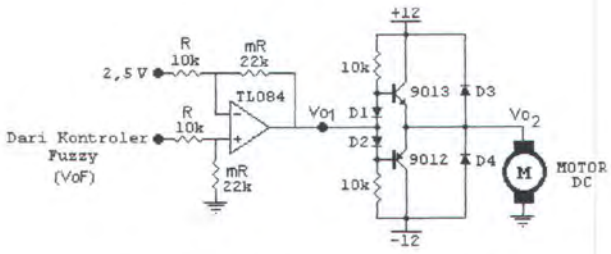
Gambar 3.2 Lengan Robot Tiga Derajad Kebebasan

3.2.2 Aktuator

Aktuator adalah bagian yang menggerakan *plant* dimana aktuator akan mengubah sinyal level tegangan dari kontroler fuzzy untuk menggerakan motor DC pada plant. Aktuator ini harus mampu mengubah sinyal tegangan dari kontroler fuzzy untuk menggerakan motor DC agar berputar ke kiri atau ke kanan dan mampu menyediakan arus yang memadai.

Karena jangkah level sinyal tegangan output dari kontroler fuzzy adalah 0 sampai 5 volt, maka dirancang bahwa aktuator akan menggerakan motor DC ke kiri dengan torsi maksimum pada level sinyal input 0 volt dan akan menggerakkan motor ke kanan dengan torsi maksimum pada level sinyal input 5 volt. Pada posisi level sinyal input 2,5 volt, aktuator harus menghentikan gerakan motor.

Dengan memperhatikan kondisi diatas maka dirancang aktuator yang terdiri dari penguat diferensial yang input tak membaliknya dihubungkan ke keluaran kontroler fuzzy dan input membaliknya dihubungkan pada suatu tegangan referensi 2,5 volt. Pada output penguat diferensial ini dihubungkan dengan amplifier *push pull* untuk menyediakan arus yang cukup untuk men*drive* motor.



Gambar 3.3 *Driver* Motor DC

3.2.2.1 Penguat Diferensial

Rangkaian diferensial dibuat dengan penguatan tegangan sebesar 2,2, hal ini karena diperkirakan rangkaian *push pull* mempunyai penguatan tegangan sebesar 0,9. Pada rangkaian diferensial sederhana ini tegangan outputnya adalah:

$$Vo1 = m\,(E1-E2) = m\,(VoF - 2,5)$$

dimana E1/VoF = keluaran kontroler fuzzy

E2 = tegangan referensi 2,5 volt.

m = penguatan amplifier = mR/R

Dengan memilih resistor R = 10k, dengan penguatan yang diinginkan (m) sebesar 2,2, maka resistor mR = 22k. Sehingga pada saat output kontroler fuzzy = 0 volt maka output penguat diferensial = -5,5 volt dan ini akan menggerakan motor ke kiri dengan torsi maksimum, dan pada saat output kontroler fuzzy = 5 volt maka output penguat diferensial = +5,5 volt, dan motor akan berputar kekanan dengan torsi maksimum. Rangkaian penguat diferensial ini dibuat tiga buah untuk men*driver* tiga buah motor DC pada *joint* lengan robot.

3.2.2.1 Penguat Daya *Push Pull*

Pada Bab II telah dijelaskan cara-cara untuk mengatur kecepatan putar motor DC, dalam tugas akhir ini digunakan cara mengatur tegangan catu pada motor DC. Pada keluaran penguat diferensial diperlukan rangkaian penguat daya untuk memperbesar arus sehingga cukup untuk menggerakan motor. Rangkaian penguat daya yang dipakai pada perencanaan ini merupakan penguat simetris komplementer kelas B (*push pull*) . Penguat ini menggunakan transistor tipe NPN dan PNP yang dihubungkan secara komplemen simetris supaya dapat menggerakkan motor dalam dua arah yang berlawanan. Masing-masing transistor dirangkai dengan konfigurasi *common* kolektor (pada gambar 3.3) sehingga mempunyai penguatan arus A_I besar dan penguatan tegangan A_V mendekati 1.

Dalam perancangan ini diasumsikan bahwa transistor NPN dan PNP yang dipakai benar-benar komplementer, artinya kurva transkonduktansi keduanya adalah serupa dan mempunyai harga βdc yang sama. Transistor yang dipilih adalah 9013 (NPN) dan 9012 (PNP) yang mempunyai IC maksimum 400mA, diperkirakan cukup untuk menyediakan arus yang memadai bagi motor. Penguatan dc (βdc) semua transistor dipilih mendekati sama yaitu rata-rata sebesar 160.

Tegangan supply yang diberikan penguat ini merupakan supply simetris sebesar ±12 volt. Untuk menghilangkan cacat penyeberangan kedua transistor dibias dengan bias dioda D1 dan D2 yang biasa digunakan pada penguat *push pull*. Arus kecil stasioner karena bias ini pada emitor-kolektor dan kolektor-

emitor kedua transisitor ini adalah sama karena kedua transistor terhubung seri. Arus stasioner ini besarnya adalah:

$$Ic = \frac{V_{CC} - V_{EE} - 2V_{BE}}{2R}$$ dimana resistor bias R = 10k

$$Ic = \frac{12 - (-12) - 2(0,7)}{20k} = 1,13 \text{ mA}$$

Arus kecil ini mengangkat titik kerja ke dua transistor sedikit diatas titik sumbat (*cutoff*) untuk mencegah cacat penyeberangan. Jika ada input tegangan positif, akan membuka transistor bagian atas dan menutup bagian bawah, tegangan emitor transistor atas akan mengikuti tegangan basis dan memberikan tegangan posistif ini ke beban (motor). Demikian pula sebaliknya bila mendapat tegangan input negatif.

Penguatan tegangan sinyal pada transistor yang sedang aktif adalah:

$$A = \frac{R_L}{R_L + \text{Re}}$$

dimana RL = Resistansi dalam motor, dari pengukuran = 87Ω

Re = Resistansi dinamik transistor, biasanya diambil 1Ω

$$A = \frac{87}{87+1} = 0,98$$

Penguatan daya pada transistor yang sedang aktif:

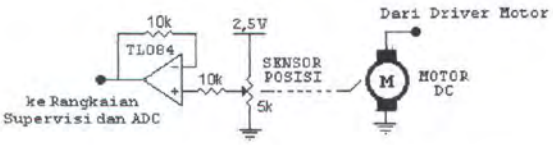
$$G = \beta_{dc} \frac{R_L}{R_L + \text{Re}}$$ dimana βdc = 160

G = 160 x 0,98 = 156,8

Dioda D3 dan D4 yang terpasang paralel terhadap ke dua transistor adalah untuk melindungi transisitor dari *spike* transien lilitan motor.

3.2.3 Tranduser Posisi

Tranduser posisi dalam tugas akhir ini adalah tiga buah resistor variabel potensio linier yang ditempatkan pada masing-masing persambungan lengan robot. Potensio yang dipakai mempunyai jangkah sudut antara 0^{o} sampai 180^{o} (-90^{o} sampai $+90^{o}$). Dengan memberikan tegangan referensi 2,5 volt maka posisi sudut yang dikonversi adalah: -90^{o} setara dengan 0 volt, sampai $+90^{o}$ setara dengan 2,5 volt. Potensio ini dikopelkan secara langsung pada motor DC untuk mengetahui posisi sudut dari motor. Gambar tranduser posisi pada gambar 3.4.



Gambar 3.4 Tranduser Posisi

Keluaran potensio ini dihubungkan pada *foltage folower* menggunakan op-amp TL084 yang berisikan empat buah op-amp. Penggunaan *foltage folower* ini dimaksudkan untuk menjamin sinyal posisi tetap valid karena digunakan kabel yang relatif panjang.

3.2.4 Rangkaian Supervisi

Rangkaian supervisi adalah rangkaian yang dihubungkan dengan masukan

NLX220 dan berfungsi sebagai:

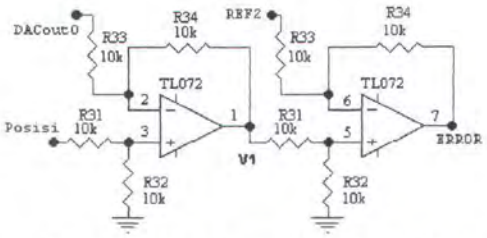
1. Penghasil sinyal *error* yaitu selisih antara nilai tegangan referensi (nilai setting) dari ouput DAC dan sinyal posisi dari pengondisi sinyal untuk keperluan proses.
2. Penghasil sinyal perbedaan *error* (*delay error*) dengan cara menunda sinyal *error* dari penghasil sinyal *error*, sehingga keluaran dari rangkaian ini adalah *delay* dari sinyal masukan.

Nilai tegangan referensi untuk keperluan *setting*, dalam hal ini adalah posisi sudut akhir dari masing-masing motor servo diperoleh dari DAC yang di*interface*kan ke IBM PC melalui port LPT1. Nilai *setting* yang dimasukkan oleh pengguna diteruskan ke DAC untuk memperoleh tegangan yang sesuai dengan nilai *setting* (jangkah antara 0 volt sampai 2,5 volt). Tegangan ini dipakai oleh rangkaian penghasil *error* untuk memperoleh nilai *error* yaitu perbedaan/selisih antara nilai *setting* dengan nilai posisi sebenarnya.

3.2.4.1 Rangkaian Penghasil *Error*

Rangkaian penghasil *error* dibangun dari dua buah penguat diferensial berpenguatan 1 (gambar 3.5). Pada penguat diferensial pertama, masukan membalik dikenakan pada ouput DAC (pemberi nilai set) dan masukan tak membalik dikenakan pada output sensor posisi (nilai aktual posisi). Keluaran dari rangkaian ini adalah selisih dari output DAC dikurangi output pengondisi sinyal, atau dinyatakan sebagai persamaan:

V1 = tegangan DACout - tegangan Posisi



Gambar 3.5 Rangkaian Penghasil Error

Untuk menjaga sinyal (V1) ini tidak keluar dari jangkauan masukan prosesor fuzzy (karena bisa positif atau negatif), maka sinyal *error* tersebut di-*offset* dengan menggunakan rangkaian penguat diferensial kedua yang berpenguatan 1. Masukan membalik dari penguat ini diberi tegangan referensi konstan sebesar -2.5 volt(REF2). Dalam persamaan:

$$\text{Error} = V1 - (-2{,}5) = (\text{DACout} - \text{posisi}) + 2{,}5$$

Jadi jika selisih antara nilai set dan aktual = 0 volt maka setara sinyal *error* = 2,5 volt pada input kontroler fuzzy, *error* posif maksimum adalah 5 volt dan *error* negatif maksimum adalah 0 volt.

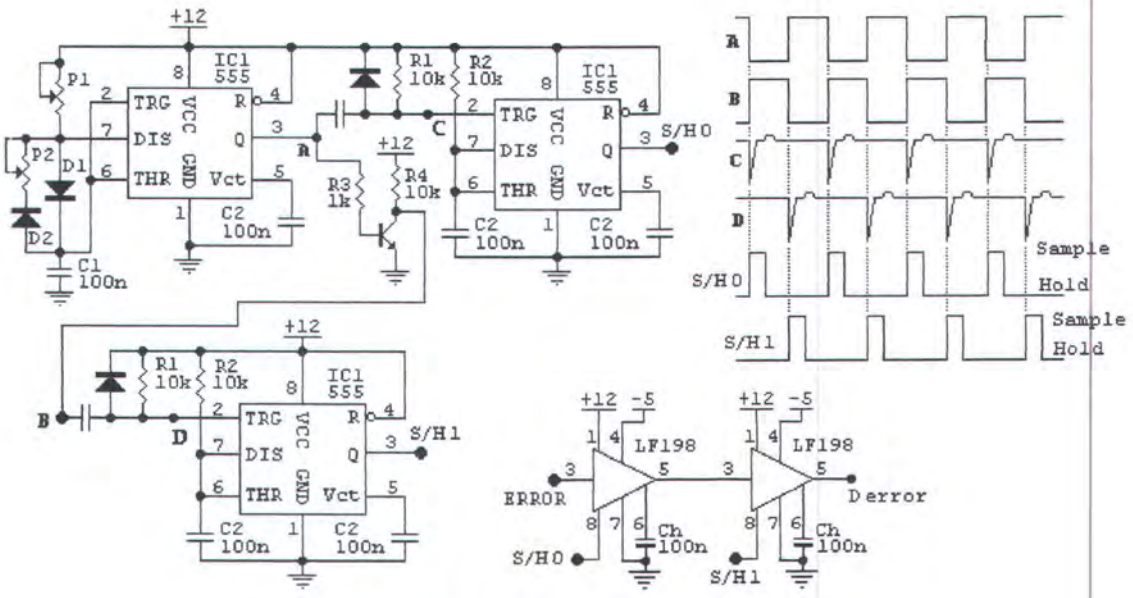
3.2.4.2 Rangkaian Penghasil *Delay Error*

Untuk menghasilkan sinyal *delay error* diperlukan suatu rangkaian yang dapat menunda suatu sinyal sebesar waktu t. Rangkaian penghasil *delay error* dibangun dari dua buah tingkat IC *sample and hold* LF198 yang dikendalikan oleh sebuah buah *one shut multivibrator* IC555 untuk setiap tingkatnya. Dua buah *one shut multivibrator* ini dikendalikan oleh sebuah *multivibrator astabil* IC555 yang mempunyai *duty cicle* 50% dan sebuah inverter dari transistor. *One shut multivibrator* yang pertama di*triger* oleh transisi turun output *multivibrator*

astabil. Sedangkan yang kedua di*triger* oleh transisi turun output *multivibrator astabil* yang dibalik oleh inverter transistor. Tundaan waktu selama t detik dapat diatur dengan mengatur periode *multivibrator astabil*.

Rangkaian *multivibrator astabil* (gambar 3.6) dirancang agar mempunyai *duty cicle* 50%, dimaksudkan agar periode sampel untuk setiap tingkat *sample and hold* sama, yaitu 50% dari periode total (waktu tundaan sinyal t). Hal ini dilakukan dengan memasang dua buah dioda yang dipasang seri pada potensio P1 dan P2. Dengan mengatur agar P1=P2 maka waktu *charge* (arus pengisian lewat P1 dan D1) dan *discharge* (arus pengosongan lewat P2 dan D2) kapasitor C1 akan sama dan ini akan membuat ouput IC555 mepunyai waktu *high* dan *low* yang sama. Rangkaian *multivibrator astabil* ini dirancang mempunyai periode maksimum 500 mdetik yang diperkirakan cukup untuk memberi tundaan sinyal *error*. Waktu tundaan ini juga mempertimbangkan *output drop rate* dari *sample and hold* jika dipakai kapasitor *hold* 100 nF maka *drop rate*-nya 10^{-4} volt/detik (pada *data sheet*).

Rangkaian *one shut multivibrator* dirancang hanya akan bekerja jika disulut oleh transisi turun sinyal *triger*. Hal ini dilakukan dengan memasang kapasitor kopling, resistor *pull-up* dan dioda untuk memangkas *spike* positif yang tidak diinginkan. Ouput rangkaian ini digunakan oleh LF198 untuk menyampling sinyal pada saat pulsa *high*, dan menggenggam level tegangan yang disampling pada saat pulsa *low*. Lebar pulsa *high* yang digunakan untuk sampling dibuat kira-kira 1 mdetik. Pemilihan waktu sampling ini mempertimbangkan *acquisition time* (waktu yang dibutuhkan agar output valid) dari *sample and hold* jika dipakai



Gambar 3.6 Rangkaian *Delay Error*

kapasitor *hold* 100 nF adalah 20 μdetik. Dua buah rangkaian *one shut multivibrator* ini bekerja secara bergantian dengan selisih waktu setengah dari perioda *multivibrator astabil*.

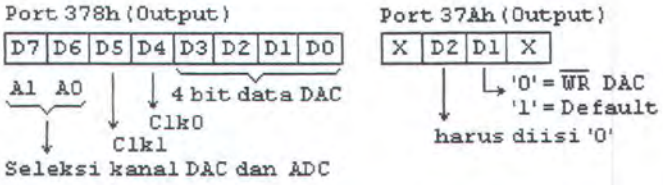
Alasan digunakannya dua buah tingkat *sample and hold* adalah agar output rangkaian penghasil *delay error* ini benar-benar merupakan tundaan dari sinyal input. Jika dipakai hanya satu tingkat, maka pada saat mode sampling maka output *sample and hold* sama dengan input, jadi ouputnya bukan benar-benar tundaan sinyal input. Sinyal *high* pada pin 8 akan menyebabkan mode sampling pada IC LF198, sedangkan pada saat *low* sinyal yang disampling akan ditahan (*mode hold*) sampai ada input *high* lagi. Rangkaian lengkap dan diagram waktunya ada pada gambar 3.6.

3.2.5 DAC Pemberi Nilai Set

Pada sistem yang direncanakan, posisi akhir dari masing-masing persambungan lengan dimasukan oleh pengguna. Nilai ini dalah nilai set atau nilai referensi yang akan dibandingkan oleh rangkaian supervisi dengan nilai posisi yang sebenarnya untuk memperoleh sinyal *error*. Nilai set dimasukan pengguna melalui PC dan akan diubah ke tegangan analog oleh DAC yang di-*interface*kan lewat port LPT1 komputer.

Karana yang dikendalikan adalah tiga buah *plant*, maka dibutuhkan tiga buah kanal DAC. DAC yang dipilih adalah MX7226, DAC 8-bit 4 kanal (CMOS Quad 8-Bit DAC) dari MAXIM. DAC ini mempunyai keunggulan antara lain: 4 kanal tegangan output yang ter-*buffer*, tidak membutuhkan komponen eksternal untuk penyesuaian output, input digital yang ter-*buffer* dengan *latch* untuk masing-masing kanal, kompatibel TTL/CMOS dan dapat dioperasikan dengan catu daya tunggal atau dual.

DAC ini dikendalikan oleh port LPT1 komputer yaitu oleh port 378h (output 8-bit) dan port 37Ah (output) bit ke 1. Port 378h dan port 37Ah di-*buffer* oleh IC 74LS245 serta 74LS125 untuk menjaga level digital agar valid untuk mengkompensasi panjangnya kabel printer.



Gambar 3.7 Port LPT1 yang Dilibatkan dalam *Write* DAC

Data yang diberikan dibagi mejadi 4 bit data bawah dan 4 bit atas (bit ke 0-3 port 378h) secara berurutan dan ditahan oleh dua buah IC 74LS175 untuk memperoleh 8 bit data.

Sisa bit port 378h digunakan untuk kontrol, yaitu bit ke 4 untuk *clock* 74LS174 (Clk0) yang menahan 4-bit data bawah, bit ke 5 untuk untuk *clock* 74LS174 (Clk1) yang menahan 4-bit data atas, bit ke 6 dan 7 digunakan untuk seleksi kanal DAC (A0 dan A1).

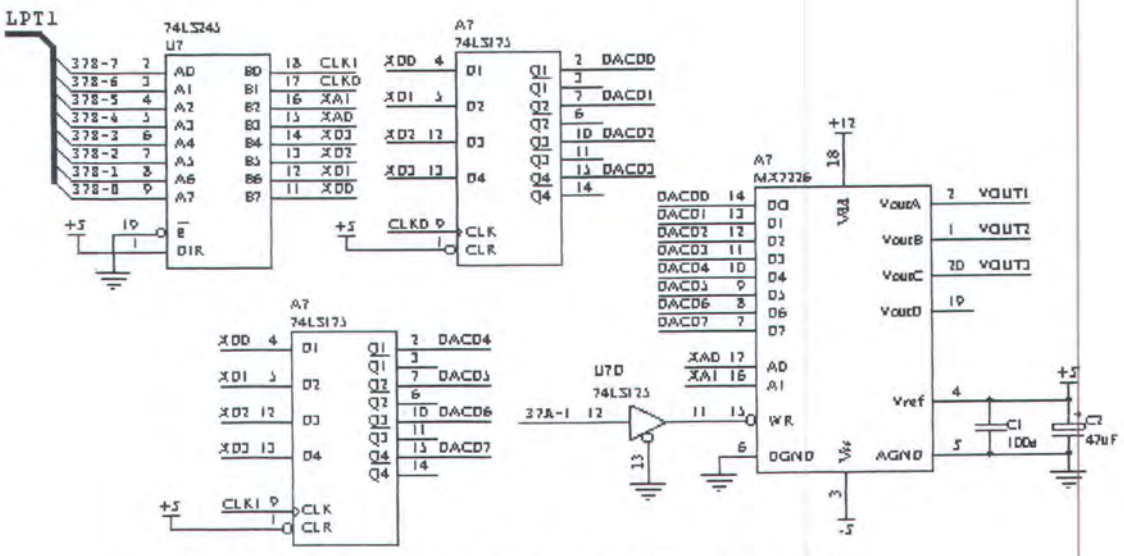
Untuk *write* DAC (*latch* data digital ke *buffer* kanal yang terseleksi) digunakan port 37Ah bit ke 1. Secara lengkap isi data port LPT1 yang dilibatkan dalam operasi *write* DAC pada gambar 3.7, rangkaian lengkap pada gambar 3.8.

Cara kerja rangkaian untuk mengeluarkan 8-bit data digital ke salah satu kanal adalah sebagai berikut:

- port 37Ah diisi xxxx x01xb (x=dont' care) ini adalah nilai *default.*
- D7-D0 port 378h secara berurutan diisi oleh: seleksi kanal DAC (2-bit), 0, 0, 4 bit data bawah.
- D7-D0 port 378h diisi oleh: seleksi kanal DAC , 1, 0, 4 bit data bawah.
- D7-D0 port 378h diisi oleh: seleksi kanal DAC , 0, 0, 4 bit data bawah.
- D7-D0 port 378h diisi oleh: seleksi kanal DAC , 0, 0, 4 bit data atas.
- D7-D0 port 378h diisi oleh: seleksi kanal DAC , 0, 1, 4 bit data atas.
- D7-D0 port 378h diisi oleh: seleksi kanal DAC , 0, 0, 4 bit data atas.
- port 37Ah diisi xxxx x00xb (*write* DAC).
- port 37Ah diisi xxxx x01xb (default).

Urutan diatas adalah satu siklus pengeluaran data digital ke salah satu kanal DAC yang akan diubah ke tegangan analog yang bersesuaian.

Pada pin referensi DAC diberikan tegangan referensi sebesar +5 volt, sehingga untuk setiap step kenaikan 1 LSB data digital akan diperoleh step kenaikan tegangan analog sebesar $5/2^8 =$ 19,53 mV. Catu daya yang digunakan adalah +12 volt dan -5 volt, hal ini dimaksudkan agar semua jangkah output analog dapat terkover (0-5 volt) tanpa ada tegangan yang terpangkas (terutama untuk data digital 00h maka output harus 0 volt atau tidak ada tegangan offset). Untuk memberi nilai set pada tiga buah plant maka hanya dibutuhkan 3 buah kanal dari 4 kanal yang tersedia. Output dari DAC ini dikenakan pada input rangakaian penghasil *error* untuk dibandingkan dengan sinyal posisi.



Gambar 3.8 Rangkaian DAC 8-Bit 4 Kanal Port LPT1

3.2.6 *Analog to Digital Converter* (ADC)

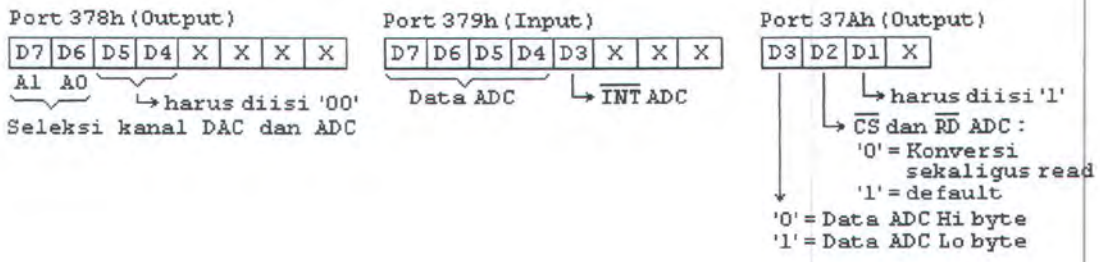
Pada sistem yang direncanakan, lengan robot mampu mencapai titik-titik posisi yang didefinisikan oleh pengguna sehingga dapat membentuk/menempuh trayektori/lintasan tertentu. Untuk memasukan nilai titik-titik posisi yang harus dilintasi oleh lengan, lengan robot digerakan secara manual oleh pengguna untuk mencapai suatu titik, kemudian nilai posisi pada titik ini diakuisisi oleh ADC kemudian disimpan di memori PC (*mode setting*) . Pada saat menjalankan lengan robot (*mode run*) DAC memberikan nilai set posisi berdasarkan data-data posisi yang diambil oleh ADC, disamping itu PC juga memonitor melaui ADC apakah posisi masing-masing lengan sudah sama dengan titik akhir yang dituju. Jika sudah sama, nilai set pada DAC oleh PC diganti dengan data berikutnya untuk bergerak ke posisi berikutnya pula. Demikian seterusnya hingga semua data-data titik posisi sudah dilewati semua. Dengan demikian *end effector* (ujung paling akhir lengan robot) dapat membentuk lintasan yang diiginkan oleh pengguna. Jadi fungsi rangkaian ADC adalah untuk memasukan nilai posisi ke memori PC (pada saat *mode set*) dan memonitor posisi masing-masing lengan (pada saat *mode run*).

Karena yang akan diakuisisi tiga buah plant maka diperlukan ADC yang mempunyai lebih dari satu kanal. Modul ADC dalam tugas akhir ini menggunakan MAX154 (8-bit ADC 4 kanal) yang hanya memerlukan komponen luar yang minimal, disebabkan ADC ini sudah mempunyai rangkaian multiplekser dan rangkaian *track-and-hold internal*. Selain itu sanggup beroperasi dengan catu +5V, dan tidak memerlukan rangkaian *clock eksternal*.

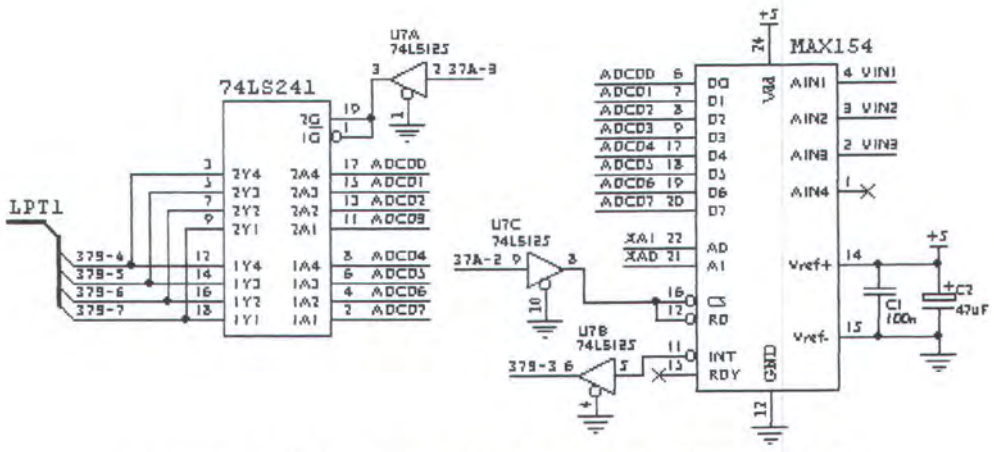
Rangkaian ADC diperlihatkan pada gambar 3.10 dalam rangkaian tersebut ADC dikontrol oleh port LPT1 antara lain:

- Port 37Ah bit ke 2 (ouput) dipakai untuk kontrol *start conversion* sekaligus membaca data hasil konversi sebelumnya (pin /CS dan /RD dijadikan satu).
- Port 379h bit ke 3 (input) digunakan untuk *pooling* sinyal /INT.
- Port 379h bit 4 sampai 7 (4 bit input) dipakai sebagai jalur data secara 4 bit.
- Port 37Ah bit ke 3 (ouput) dipakai untuk kontrol seleksi 4 bit data ADC yang akan dimasukan ke port 379h. IC 74LS241 akan meloloskan 4-bit data bawah ADC jika output port 37Ah bit-1 bernilai '1' dan akan meloloskan 4-bit data atas ADC jika output port 37Ah bit-1 bernilai '0'.
- Untuk seleksi kanal A1-A0 dipakai port 378h bit ke 4 dan 5 (sama dengan seleksi kanal DAC diatas).

Secara lengkap isi data port LPT1 yang dilibatkan dalam operasi ADC pada gambar 3.9, rangkaian lengkap rangkaian ADC pada gambar 3.10. Pada pin referensi ADC diberikan tegangan referensi sebesar +5 volt, sehingga untuk setiap step kenaikan 1 LSB data digital setara dengan tegangan analog sebesar $5/2^8$ = 19,53 mV.



Gambar 3.9 Port LPT1 yang dilibatkan dalam operasi ADC



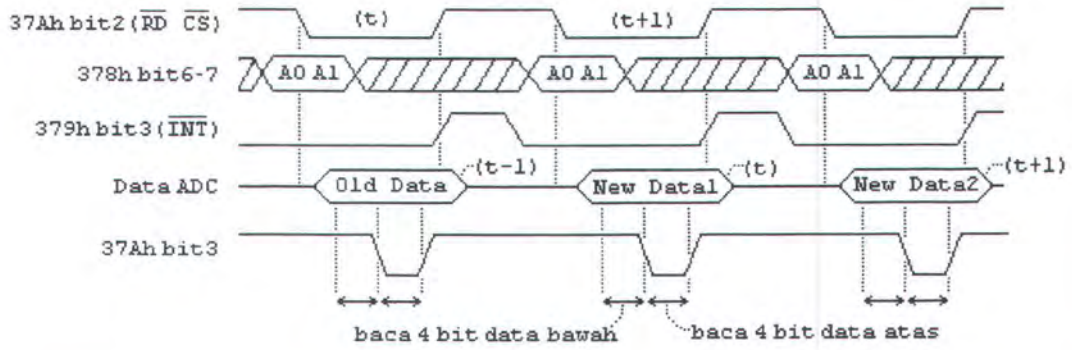
Gambar 3.10 Rangkaian ADC 8-Bit 4 Kanal Port LPT1

Cara kerja Rangkain ADC adalah sebagai berikut:

- Port 37Ah disi nilai *default*: xxxx111xb (x= dont' care)
- Seleksi kanal dikeluarkan pada port 378h bit 7-6 (A1A0), misalkan kanal 1 = '0000xxxxb'.
- Port 37Ah diisi: xxxx101xb (/CS dan /RD dibuat *low*), ini akan memberikan perintah *start conversion* pada ADC (t) dan sekaligus mengakibatkan data dari hasil konversi sebelumnya (t-1)keluar pada pin data ADC (D7-D0).
- Port 379h dibaca, hasil pembacaan ini adalah bit 4 data bawah (pada D7-D4) hasil konversi ADC sebelumnya (t-1).
- Port 37Ah diisi: xxxx001xb (/CS dan /RD masih *low*) untuk memberikan seleksi pada 74LS241 agar meloloskan 4 bit data atas ADC.
- Port 379h dibaca, hasil pembacaan ini adalah bit 4 data atas (pada D7-D4) hasil konversi ADC sebelumnya (t-1).
- Port 37Ah diisi: xxxx111xb (nilai *default*) atau /CS dan /RD dibuat *high*. Sinyal pada pin /INT pada ADC akan *high* seiring dengan transisi *high* /CS.

- Port 379h dibaca untuk *pooling* sinyal /INT (pada D3). Jika /INT masih *high*, maka konversi (t) belum selesai, jika sudah *low* maka konversi (t) sudah selesai. Hasil data konversi (t) dibaca dengan mengulangi siklus diatas.

Diagram waktu siklus konversi dan pembacaan data hasil konversi melaui LPT1 pada gambar 3.11.



Gambar 3.11 Siklus konversi dan Pembacaan Data Hasil Konversi Melalui LPT1

3.3 PERANCANGAN PERANGKAT LUNAK

Perangkat lunak yang dirancang terdiri dari perangkat lunak untuk kontroler fuzzy dan perangkat lunak untuk PC.

3.3.1 Perangkat Lunak Pada Kontroler Logika Fuzzy

Perangkat lunak pada kontroler fuzzy berupa *rule set* atau himpunan aturan-aturan yang berfungsi mengatur posisi sudut masing-masing persambungan/*joint* berdasarkan nilai set dari DAC.

Seperti dijelaskan dalam dasar teori, Dalam perancangan *rule-rule* fuzzy untuk kontroler dikembangkan berdasarkan respon kontroler PD *(proporsional-derivative*) konvensional dan untuk unjuk kerja maksimal dicapai dengan metode coba-coba.

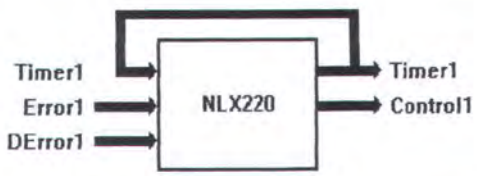
Langkah pertama dalah membuat model fuzzy dengan menentukan I/O yang diperlukan. NLX220 mempunyai empat port input sinyal analog dan empat port output sinyal analog. Parameter yang diperlukan untuk satu kontroler motor, untuk input adalah:

1. *Timer* (*loop back*), digunakan untuk mengatur kecepatan proses.
2. *Error*, selisih antara nilai set dan nilai aktual.
3. *DError*, tundaan/*delay* dari sinyal *error*.

Untuk ouput adalah:

1. *Timer*, yang di*loopback*kan untuk mengatur kecepatan proses.
2. *Control*, adalah keluaran aksi kontrol yang diberikan ke motor.

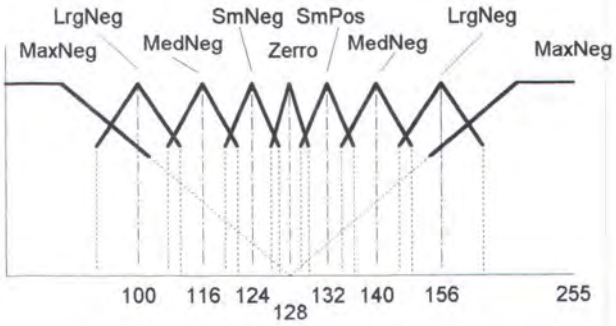
Gambar 3.12 menunjukan hubungan I/O untuk kontoler satu motor. Karena yang akan dikontrol adalah tiga buah motor maka diperlukan tiga buah sistem yang sama, sehingga dibutuhkan dua buah NLX220.



Gambar 3.12 I/O pada NLX220 untuk Kontroler 1 Motor

Langkah ke dua adalah menentukan fungsi keanggotaan dari masing-masing parameter input yang digunakan. Fungsi keanggotaan untuk error yang mana diperlukan untuk bagian respon proporsional adalah seperti gambar 3.13, dan penulisan pada *insight* adalah sbb:

1. Error is Zerro (128, 2, symmetric inclusive)
2. Error is SmNeg (124, 4, symmetric inclusive)
3. Error is SmPos (132, 4, symmetric inclusive)
4. Error is MedNeg (116, 6, symmetric inclusive)
5. Error is MedPos (140, 6, symmetric inclusive)
6. Error is LrgNeg (100, 12, symmetric inclusive)
7. Error is LrgPos (156, 12, symmetric inclusive)
8. Error is MaxNeg (128, 24, right exclusive)
9. Error is MaxPos (128, 24, left exclusive)

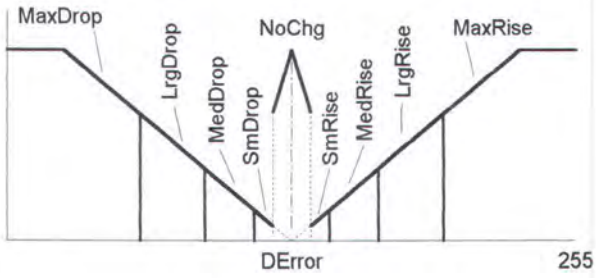


Gambar 3.13 Fungsi Keanggotaan *Error*

Fungsi keanggotaan untuk perubahan *error* yang diperlukan untuk bagian respon derivatif adalah seperti gambar 3.14. Perubahan *error* dihitung berdasarkan error yang lalu (*DError*), hal ini dapat dihitung secara internal dengan membuat fungsi keanggotaan *error* dengan *center* yang *floating* terhadap *DError* dan penulisan pada *insight* adalah sbb:

1. Error is NoChg (DError, 2, symmetric inclusive)

2. Error is SmDrop (DError, 4, right exclusive)
3. Error is SmRise (DError, 4, left exclusive)
4. Error is MedDrop (DError, 6, right exclusive)
5. Error is MedRise (DError, 6, symmetric inclusive)
6. Error is LrgDrop (DError, 12, right exclusive)
7. Error is LrgRise (DError, 12, symmetric inclusive)
8. Error is MaxDrop (DError, 24, right exclusive)
9. Error is MaxRise (DError, 24, left exclusive)



Gambar 3.14 Fungsi Keanggotaan Perubahan *Error*

Fungsi keanggotaan untuk *Timer* yang diperlukan untuk menentukan kecepatan respon sistem kontrol. Kontroler tidak akan mengubah kondisi kontrol bila *timer* belum mencapai maksimum. Fungsi keanggotaan *Timer* adalah sbb:

1. Timer is Zerro (0, 1, symmetric inclusive)
2. Timer is NotMax (249, 0, left inclusive)
3. Timer is Max (250, 0, right inclusive)

Langkah ke tiga adalah menentukan aturan-aturan (*rules*) untuk menentukan aksi kontrol. Dengan mempertimbangkan persamaan 2.1 dan gambar

2.20 serta gambar 2.21 pada Bab II, maka aksi koreksi terhadap nilai kontrol untuk bagian proporsional dan derivatif adalah sbb:

// Start awal:

1. If Timer is Zerro then Control = 128
2. If Timer is NotMax then Control + 0

// Bagian proporsional:

3. If Error is Zerro then Control = 128
4. If Error is MaxPos then Control = 255
5. If Error is MaxNeg then Control = 0
6. If Error is LrgPos then Control = 233
7. If Error is LrgNeg then Control = 23
8. If Error is MedPos then Control = 173
9. If Error is MedNeg then Control = 83
10. If Error is SmdPos then Control = 143
11. If Error is SmNeg then Control = 113

// Rule pemisah:

12. If Timer is NotMax then Control + 0

// Bagian Derivativ:

13. If Error is NoChg then Control + 0
14. If Error is MaxRise then Control + 40
15. If Error is MaxDrop then Control - 40
16. If Error is LrgRise then Control + 20
17. If Error is LrgDRop then Control - 20
18. If Error is MedRise then Control + 6
19. If Error is MedDrop then Control - 6
20. If Error is SmdRise then Control + 2
21. If Error is SmDrop then Control - 2

// Pengatur kecepatan proses:

22. If Timer is Max then Timer = 1
23. If Timer is NotMax then Timer + 20

Variabel untuk umpan balik adalah *Error* yang menentukan nilai aksi pada register output control. Pada rule 1, pada saat awal (timer =0*) register output control* diberi nilai 128, setara dengan 2,5 volt (motor tidak diberi aksi apa-apa). Ini untuk memperkecil transien sistem pada saat start awal. *Rule* 2 dan 3, jika

timer tidak maksimum (*NotMax*), register control tidak di*update*, control di*update* hanya pada saat *timer* maksimum. Hal ini digunakan untuk mengatur kecepatan proses dengan mengatur penambahan *timer* (*rule* 23). *Rule* 12 adalah *rule* pemisah untuk memisahkan bagian proporsional dengan derivatif. Jika tidak dipisah, NLX220 akan memenangkan salah satu *rule* dari kelompok/urutan yang dianggap sejenis (dalam hal ini sama-sama menggunakan variabel *error* pada bagian proporsional dan dervatifnya). Untuk itu perlu dibuat *rule* lain utuk memisahkan ke dua kelompok dengan satu *rule* yang tidak akan berpengaruh pada kontrol. Pada bagian proporsional (*rule* 3-11) koreksi = 0 jika *error* = 0 (*Zerro*) dan akan membesar positif jika *error* juga membesar positif, demikian sebaliknya. Pada bagian derivatif (*rule* 13-21) koreksi = 0 jika tidak ada perubahan pada *error* (NoChg) dan akan membesar positif jika *error* naik positif, demikian juga sebaliknya.

3.3.2 Perencanaan Perangkat Lunak Pada PC

Perangkat lunak pada komputer mempunyai tugas utama sebagai penampil data dari proses fuzzy dan memori penyimpan titik-titik posisi sudut yang harus ditempuh oleh lintasan lengan robot. Dalam pembuatannya dipakai perangkat lunak *Borland Delphi* yang sangat memudahkan pemrograman yang dilakukan dalam lingkungan *Windows*.

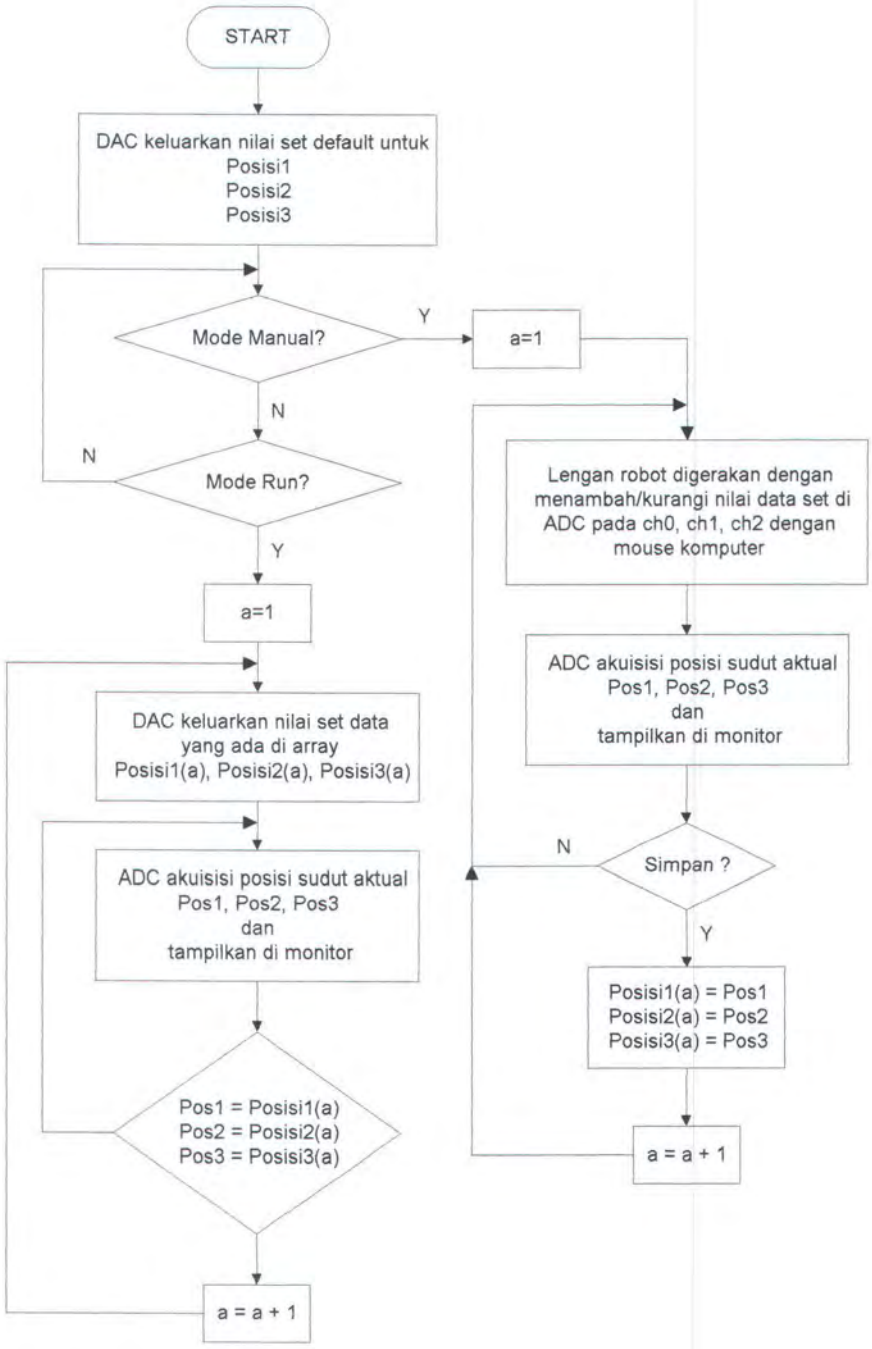
Perangkat lunak ini akan mempunyai fungsi-fungsi sebagai berikut:

- Menampilkan proses yang terjadi dalam plant, dalam hal ini adalah posisi sudut masing-masing *joint* lengan.
- Menyimpan titik-titik nilai set yang dimasukan pengguna dengan menuntun lengan robot (pada saat *mode manual*) dan menentukan nilai set secara otomatis (pada saat *mode run*) dengan berdasarkan informasi nilai titik-titik set yang telah diinputkan oleh pengguna.

Setelah program dimulai maka pertama kali yang dilakukan oleh PC adalah mengeluarkan nilai set posisi sudut *default* masing-masing lengan. Pengguna dapat memasukan titik-titik posisi sudut dengan menuntun robot lewat *mouse* PC. Pada setiap titik yang harus dilewati, nilai ini disimpan dalam *array* data di memori PC. Pada saat robot dijalankan dalam *mode run*, program pada PC akan memberikan nilai set posisi secara berurutan berdasarkan data yang ada pada array. Pada saat proses fuzzy menyelesaikan data pertama, PC akan mengganti nilai set data berikutnya, demikian seterusnya sampai data di*array* selesai ditempuh oleh lengan robot.

Pada mode manual, proses pengambilan data posisi sudut dilakukan oleh ADC MAX154 yang di*interface*kan lewat port LPT1 PC, Sedangkan pada *mode run*, proses pemberian *setting* dilakukan oleh DAC MX7226 yang juga lewat port LPT1 PC. Diagram alir dari perangkat lunak ini pada gambar 3.15.

Hasi konversi dari ADC ditampilkan dalam bentuk posisi sudut, dimana 128 x 1 LSB (setengah dari skala penuh, karena referensi pada sensor posisi adalah 2,5 volt) pada ADC mewakili 180°. Listing dari proses pengambilan dan



Gambar 3.15 Diagram Alir Program Pada PC

penampilan data dari ADC MAX154 lewat LPT1 PC dengan menggunakan *Borland Delphi* adalah sebagai berikut :

```
procedure TForm1.Timer1Timer(Sender: TObject);
begin
   CH_ADC:=0;                                        {akuisisi chanel 0}
   ADC_in;
   V_ADCTam:=V_ADC*(5/256)*(180/128);
   Edit4.Text:=FloatToStrF(V_ADCTam,ffFixed,5,2);
   CH_ADC:=$40;                                      {akuisisi chanel 1}
   ADC_in;
   V_ADCTam:= V_ADC*(5/256)*(180/128);
   Edit5.Text:=FloatToStrF(V_ADCTam,ffFixed,5,2);
   CH_ADC:=$80;                                      {akuisisi chanel 2}
   ADC_in;
   V_ADCTam:= V_ADC*(5/256)*(180/128);
   Edit6.Text:=FloatToStrF(V_ADCTam,ffFixed,5,2);
end;

Procedure ADC_in;
begin
   asm
     MOV AL,CH_ADC
     MOV DX,378h
     OUT DX,AL              {siapkan address}
     MOV AL,1000b
     MOV DX,37Ah
     OUT DX,AL              {Start Conversi & READ chanel}
     MOV CX,1000
@test1:
     DEC CX
     CMP CX,0
     JE  @lanjut            {jika INT tetap tidak valid pada 1000x looping , langsung keluar}
     MOV DX,379h
     IN  AL,DX
     AND AL,8h
     CMP AL,0               {test INT ADC apakah sudah 0}
     JNE @test1             {jika belum baca lagi}
     MOV DX,379h
     IN  AL,DX              {READ low byte}
     AND AL,0F0h
     ROR AL,4
     XOR AL,1000B
     MOV BL,AL
```

```
    MOV AL,0
    MOV DX,37Ah
    OUT DX,AL
    MOV DX,379h
    IN  AL,dx              {READ hi byte}
    AND AL,0F0h
    XOR AL,80h
    OR  BL,AL
    MOV V_ADC,BL
    JMP @Selesai
@lanjut:
    MOV AL,0
    MOV V_ADC,AL
@Selesai:                  {berikan kondisi defauld}
    MOV AL,1100b
    MOV DX,37Ah
    OUT DX,AL
    MOV AL,0
    MOV DX,378h
    OUT DX,AL
  end;
end;
```

Nilai set yang diberikan oleh DAC berdasarkan data posisi sudut, dimana jangkah dari 1 LSB sampai setengah sekala penuh DAC(128 LSB) mewakili 0^{o} sampai 180^{o}. *Listing* dari proses pengeluaran data pada DAC MX7226 lewat LPT1 PC dengan menggunakan *Borland Delphi* adalah sebagai berikut :

```
Procedure OutDAC;
begin
  asm
    MOV AL,VDAC            {kirim low byte}
    AND AL,0Fh
    OR  AL,CHDAC
    MOV DX,378h
    OUT DX,AL
```

```
    OR  AL,00010000b          {clock low byte}
    OUT DX,AL                 {latch di low byte}
    AND AL,11101111b
    OUT DX,AL
    MOV AL,VDAC {kirim hi byte}
    ROR AL,4
    AND AL,0Fh
    OR  AL,CHDAC
    MOV DX,378h
    OUT DX,AL
    OR  AL,00100000b
    OUT DX,AL     {latch di hi byte}
    AND AL,11011111b
    OUT DX,AL
    MOV AL,1110b
    MOV DX,37Ah
    OUT DX,AL     {WR DAC}
    MOV AL,0Ch
    MOV DX,37Ah
    OUT DX,AL
    MOV AL,0
    MOV DX,378h
    OUT DX,AL
  end;
end;
```

BAB IV

PENGUJIAN DAN PENGUKURAN

BAB IV

PENGUJIAN DAN PENGUKURAN

4.1 PENGUJIAN DAN PENGUKURAN

Sebelum tugas akhir ini dijalankan sebagai sebuah sistem, maka perlu dilaksanakan pengujian untuk tiap bagian sistemnya dan selanjutnya dilakukan kalibrasi dan pengukuran terhadap alat yang direncanakan.

Pengujian yang dilakukan meliputi pengujian:

- Driver motor DC
- Sensor posisi
- Rangkaian penghasil *error*
- Rangkaian penghasil *delay error*
- Rangkaian ADC dan DAC yang di*interface*kan ke LPT1 PC.
- Simulasi perangkat lunak fuzzy

Pengujian *driver* motor DC dilakukan dengan memberikan masukan tegangan antara 0 volt sampai 5 volt dengan motor DC dalam keadaan terpasang. Kemudian diukur keluaran diferensial amplifier yang menggerakan amplifier *push pul* dengan volt meter. Keluaran amplifier *push pull* juga diukur (tegangan jepit motor) dengan voltmeter.

Pengujian sensor posisi dilakukan dengan memposisikan motor pada posisi sudut tertentu kemudian keluaran *foltage follower* yang menyangga keluaran potensio diukur dengan voltmeter.

Pengujian rangkaian penghasil *error* dilakukan dengan memberikan input tegangan antara 0 sampai 2,5 volt pada rangkaian (masukan dari sensor posisi) dan memberikan tegangan antara 0 sampai 5 volt pada masukan nilai set, kemudian output dari rangkaian ini diukur dengan voltmeter.

Kalibrasi rangkaian penghasil *delay error* dilakukan dengan menset dua buah potensio pada bagian rangkaian *astable multivibrator* agar mendapatkan sinyal keluaran dengan periode 40ms, *duty cycle* 50%. Keluaran kedua *monostable mutivibrator* juga dukur apakah masing-masing mempunyai periode *high* selebar 1 ms (*sample*) dan periode *low* (*hold*) sebesar 19 ms. Pengujian rangkaian *delay error* secara keseluruhan dengan cara memberikan masukan *function generator* gigi gergaji, kemudian output dilihat pada osiloskop.

Rangkaian ADC dan DAC yang di*interface*kan pada LPT1 PC diuji dengan menghubungkan setiap *chanel* keluaran DAC ke masukan ADC. Dibuat program sederhana pada komputer untuk mengeluarkan suatu nilai data pada masing-masing keluaran DAC. Keluaran DAC diukur dengan voltmeter dan hasil konversi pada ADC ditampilkan pada monitor PC.

4.2 PENGUKURAN

4.2.1 Pengukuran Rangkaian *Driver* Motor DC

Pengukuran pada *driver* motor DC diperoleh hasil sebagai berikut:

Tabel 4.1 Pengukuran *Driver* Motor DC

No	Tegangan Input (V)	Keluaran Diverensial Amplifier (V)			Keluaran Push Pull Amplifier (V)		
		M1	M2	M3	M1	M2	M3
1	0	-5,57	-5,53	-5,55	-5,23	-5,15	-5,37
2	1,25	-2,77	-2,78	-2,77	-2,63	-2,66	-2,65
3	2,50	0	0	-0,01	0	0	0
4	3,75	2,80	2,75	2,77	2,63	2,64	2,64
5	4,94	5,42	5,38	5,35	5,27	5,21	5,20

M1, M2 dan M3 adalah *driver* untuk motor 1, motor 2 dan motor 3. Dari hasil pengukuran diatas terlihat bahwa rangkaian cukup memadai untuk menggerakan motor dengan input antara 0 sampai sedikit dibawah 2,5 volt motor berputar ke kiri dan untuk sedikit diatas 2,5 sampai 5 volt motor berputar kekanan. Pada input 2,5 volt motor akan berhenti berputar.

4.2.2 Pengukuran Sensor Posisi Motor DC

Hasil pengukuran pada rangkaian sensor posisi sudut motor DC diperoleh data sebagai berikut:

Tabel 4.2 Pengukuran Sensor Posisi Motor DC

No	Posisi Sudut (^{0})	Keluaran modul (V)			*Desired value* (V)	Error (%)		
		M1	M2	M3		M1	M2	M3
1	0	0,336	0,257	0,332	0,000	-	-	-
2	45	0,648	0,635	0,631	0,625	3,68	1,60	0,96
3	90	1,275	1,235	1,260	1,250	2,00	2,00	0,80
4	135	1,852	1,860	1,865	1,875	1,22	0,80	0,53
5	180	2,280	2,220	2,240	2,500	8,80	11,20	10,40

M1, M2 dan M3 adalah sensor posisi pada motor 1, 2 dan 3. Penyimpangan nilai yang diukur dan yang diharapkan terjadi karena pada saat posisi potensio maksimum (baik ke kiri maupun ke kanan) resistensinya tidak 0 atau maksimum (ada *offset* resistensi).

4.2.3 Pengukuran Rangkaian Penghasil *Error*

Hasil pengukuran pada rangkaian penghasil *error* diperoleh data sebagai berikut:

Tabel 4.3 Pengukuran Rangkaian Penghasil *Error*

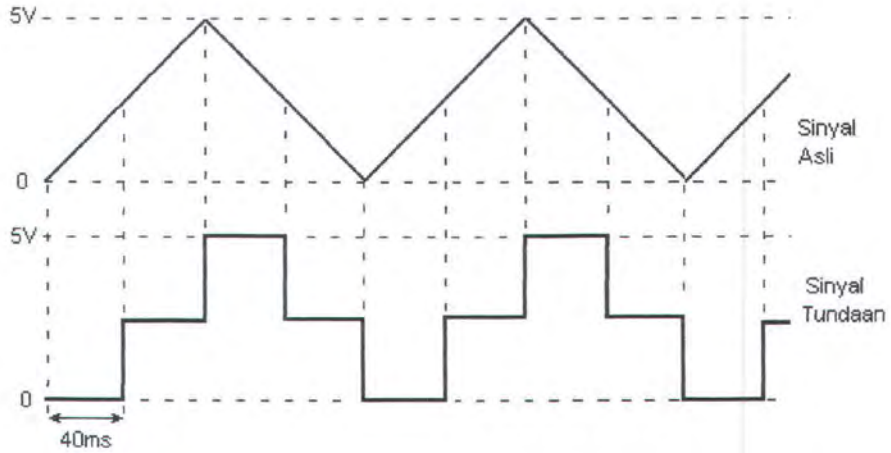
No	Masukan Modul Error (V)		Keluaran Modul Error (V)			*Desired value* (V)	Error (%)		
	Set	Posisi	M1	M2	M3		M1	M2	M3
1	0,000	2,500	0,005	0,008	0,006	0,000	-	-	-
2	0,625	1,875	1,261	1,263	1,260	1,250	0,88	1,04	0,80
3	1,250	1,250	2,522	2,525	2,521	2,500	0,88	1,00	0,84
4	1,875	0,625	3,768	3,772	3,768	3,750	0,48	0,59	0,48
5	2,500	0,000	5,010	5,010	5,010	5,000	0,20	0,20	0,20

M1, M2 dan M3 adalah keluaran modul untuk motor 1, 2 dan 3.

4.2.4 Pengukuran Rangkaian Penghasil *Delay Error*

Kalibrasi dilakukan seperti yang dijelaskan diatas, didapat sinyal output *astable multivibrator* dengan periode 40 ms dengan *duty cycle* 50%. Untuk keperluan tersebut diset P1 harus sama dengan P2. Keluaran masing-masing *monostable multivibrator* yang menggerakan dua tingkat *sample and hold* periode *high*nya mendekati 1 ms.

Pengujian dengan sinyal generator gigi gergaji amplitudo 5 volt dan peride 160 ms(frekuensi: 6,25 Hz) didapat sinyal output modul *delay error* seperti gambar 4.2.



Gambar 4.1 Sinyal Output Modul *Delay Error* dengan Input Sinyal Gigi Gergaji

4.2.5 Pengukuran Modul ADC-DAC LPT1

Pengukuran pada modul ADC-DAC LPT1 didapat data sebagai berikut:

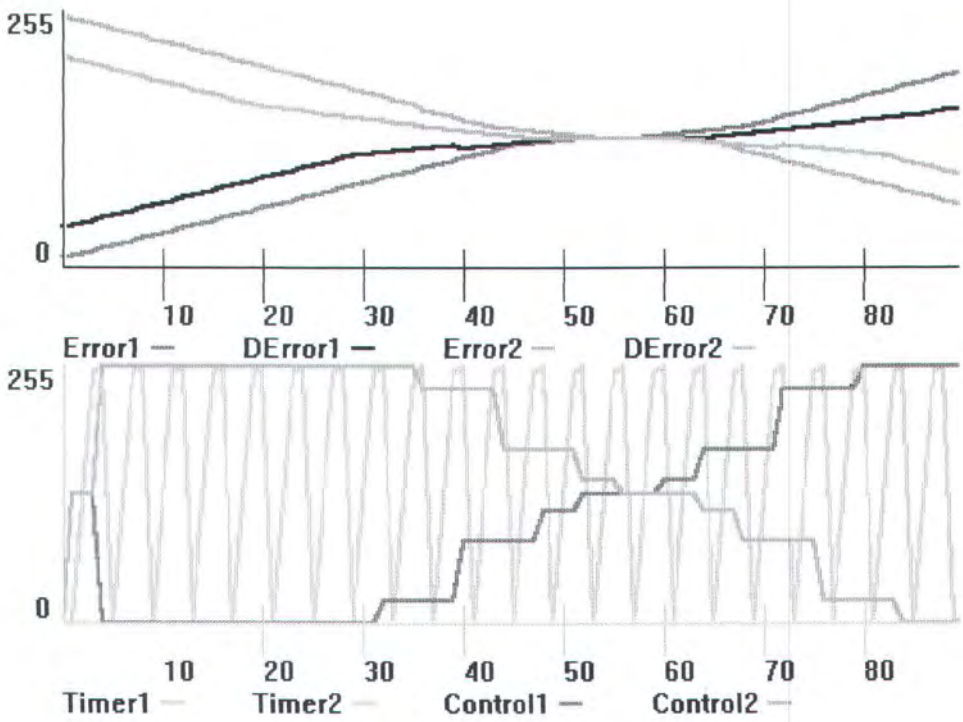
Tabel 4.4 Pengukuran Modul ADC-DAC

No	Tampilan pada Monitor untuk ADC dan DAC (mV)	Tegangan terukur pada output DAC / input ADC (mV)			Error pada tampilan DAC dan ADC (%) (error konversi)		
		CH0	CH1	CH2	CH0	CH1	CH2
1	0,000	9,5	9,5	10,3	100	100	100
2	39,063	50,6	50,7	51,4	22,8	23,0	24,0
3	214,840	231,0	231,0	232,0	7,0	7,0	7,4
4	410,160	429,0	430,0	431,0	4,4	4,6	4,8
5	605,470	626,0	628,0	629,0	3,3	3,6	3,7
6	800,780	827,0	828,0	829,0	3,2	3,3	3,4
7	1015,600	1045,0	1045,0	1046,0	2,8	2,8	2,9
8	1269,500	1302,0	1302,0	1304,0	2,5	2,5	2,6

No	Tampilan pada Monitor untuk ADC dan DAC (mV)	Tegangan terukur pada output DAC / input ADC (mV)			Error pada tampilan DAC dan ADC (%) (error konversi)		
		CH0	CH1	CH2	CH0	CH1	CH2
9	1406,300	1440,0	1441,0	1443,0	2,3	2,4	2,5
10	1503,900	1542,0	1541,0	1543,0	2,5	2,4	2,5
11	1718,800	1759,0	1759,0	1760,0	2,3	2,3	2,3
12	2011,700	2050,0	2050,0	2050,0	1,9	1,9	1,9
13	2304,700	2350,0	2350,0	2350,0	1,9	1,9	1,9
14	2402,300	2450,0	2450,0	2450,0	1,9	1,9	1,9
15	2500,000	2540,0	2540,0	2540,0	1,6	1,6	1,6

4.2.6 Hasil Simulasi Perangkat Lunak Fuzzy

Hasil Simulasi menggunakan perangkat lunak INSIGHT, kurva aksi kontrol terhadap *Error* dan *DError*:



Gambar 4.3 Grafik Simulasi *Rule* Fuzzy

4.2.7 Pengujian Gerakan Robot

Pengukuran dilakukan dengan memberi nilai set, kemudian posisi sudut berhenti masing-masing persambungan diukur, dibandingkan dengan nilai set. Untuk memudahkan pengukuran, semua satuan input dan output dalam mili volt (diukur pada ouput DAC dan sensor posisi). Data hasil pengukuran sebagai berikut:

Tabel 4.5 Pengujian Gerakan robot

Input Posisi (mV)			Posisi Berhenti (mV)			Error (%)		
M1	M2	M3	M1	M2	M3	M1	M2	M3
146,48	292,97	2412,10	117,19	312,50	2441,5	19,99	6,67	1,22
312,50	507,81	2275,40	273,44	517,58	2255,60	12,45	1,92	0,87
498,05	712,89	2080,10	468,75	722,66	2100,60	5,88	13,59	0,99
751,95	908,20	1875,00	722,66	917,97	1855,20	3,89	1,08	1,06
1064,50	1142,60	1699,20	1054,70	1152,30	1679,50	1,20	0,85	1,16
1250,00	1337,90	1425,80	1240,20	1357,50	1410,10	0,78	1,47	1,10
1748,00	1416,00	1123,00	1757,80	1425,80	1133,80	0,56	1,25	0,96
2031,30	1503.90	966,80	2041,00	1494,10	986,33	0,47	0,65	2,02
2197,30	1621,10	468,75	2207,00	1611,30	488,20	0,44	0,60	4,15
2382,80	1816,40	97,66	2392,60	1806,60	117,19	0,43	0,54	19,99

Pada pengujian gerakan robot untuk menempuh lintasan tertentu, didapat bahwa semakin banyak titik-titik posisi yang dimasukan dalam kumpulan nilai set pada memori PC, gerakan robot akan semakin halus.

BAB V

PENUTUP

BAB V

PENUTUP

5.1. KESIMPULAN

Dari hasil perancangan dan pembuatan serta pengujian sistem secara keseluruhan, dapat diambil kesimpulan sebagai berikut:

- Dengan menggunakan sistem yang berbasiskan mikrokontroler fuzzy NLX220, mempunyai keuntungan yaitu proses perancangan kontroler pada sistem robot dapat disederhanakan, tanpa memperhatikan perhitungan matematis yang rumit.
- Kerugian menggunakan sistem yang berbasiskan mikrokontroler fuzzy NLX220 adalah diperlukanya banyak rangkaian pendukung bila pin-pin I/O pada NLX220 tidak cukup untuk mengontrol suatu sistem dengan jumlah *plant* yang banyak, misalnya dalam tugas akhir ini untuk mengontrol posisi tiga buah motor dc.
- Dalam pembuatan sistem kontrol menggunakan logika fuzzy diperlukan pengetahuan yang mendalam tentang bagaimana sistem tersebut berperilaku, sebab dalam pembuatan aturan-aturan fuzzy selalu didasarkan pada pengalaman tentang suatu sistem. Pengalaman yang semakin banyak akan menjadikan kontrol fuzzy menjadi semakin optimal.
- Ketidakpresisian pada posisi yang diinginkan disebabkan beberapa hal, antara lain: sulitnya merealisasikan plant yang sempurna/ideal sesuai dengan standar

robot industri, terjadinya penyimpangan pada setiap modul pendukung kontroler fuzzy NLX220, kelemahan pada NLX220 dimana pada pengujian output yang dinaikan dengan nilai tetap kemudian diturunkan dengan nilai tetap hasilnya tidak sama (linearitas output kurang, kelemahan ADC-DAC internal NLX220).

- Dengan menggunakan sistem mikrokontroler NLX220 ternyata tidak ekonomis untuk sistem yang sederhana, karena membutuhkan banyak rangkaian pendukung.

5.2 SARAN

Saran-saran yang diharapkan dapat berguna untuk pengembangan dan penggunaan lebih lanjut dari tugas akhir ini adalah:

1. Sistem lengan robot yang direncanakan dapat dikembangkan menjadi empat derajad kebebasan dengan menambah variabel gerakan *griper* pada *end effector* robot.
2. Untuk penerapan kontroler fuzzy pada pemasalahan ketidaklinieran sistem robotik akan terlihat pada aplikasi kontrol robot dinamik, dimana beban yang berfariasi pada robot (termasuk bobot dari masing-masing lengan pada berbagai posisi) harus dikompensasi oleh torsi yang sesuai untuk mendapatkan gerakan yang diinginkan. Hal ini dapat dilakukan dengan menambah variabel kecepatan gerakan plant disamping variabel *error* posisi dan perubahan *error*.
3. Untuk keperluan penilitian secara mendalam dan ekonomis terhadap respon kontroler fuzzy disarankan untuk menggunakan model fuzzy simulasi pada

PC, kemudian untuk *interfacing* ke *plant* dapat dilakukan dengan DAC dan ADC.

DAFTAR PUSTAKA

DAFTAR PUSTAKA

Bell, D. A., 1981, *Solid State Pulse Circuits* 2nd, Reston Publishing Company, Inc., A Prentice-Hall Company, Reston, Virginia.

Bergsman, P., 1995, "Microcomputer Journal".

Coughlin, R. F., Driscoll, F. F., Soemitro, H. W. (Ed.), 1992, *Penguat Operasional dan Rangkaian Terpadu Linear* 2nd Edition, Penerbit Erlangga, Jakarta.

Gozales, K. S. FV. R. C., Lee, C. S. G., 1987, *Robotics: Cotrol, Sensing, Vision, and Intelligence*, McGraw-Hill, Singapore.

Jamshidi, M. (Ed.), Vadiee, N. (Ed.), Ross, T. J. (Ed.), 1993, **Fuzzy Logic and Control**, *Software and Hardware Applications*, PTR Prentice Hall, Englewood Cliffs, New Jersey.

Korem, Y., 1987, *Robotic For Engineer*, McGraw-Hill, Singapore.

Ogata, K., Leksono, E. (Ed.), 1993, *Teknik Kontrol Automatik (Sistem Pengaturan)* 1st Edition, Penerbit Erlangga, Jakarta.

Shaninpoor, M., 1987, *A Robot Engineering Textbook*, Harper & Row, New York.

Snyder, W. E., 1985, *Industrial Robots : Computer Interfacing and Control*, Prentice/hall International, Inc., Englewood Cliffs, New Jersey.

Spong, M. W., Vidyasagar, M., 1989, *Robot Dynamics and Control*, John Willey & Sons, Inc., Singapore.

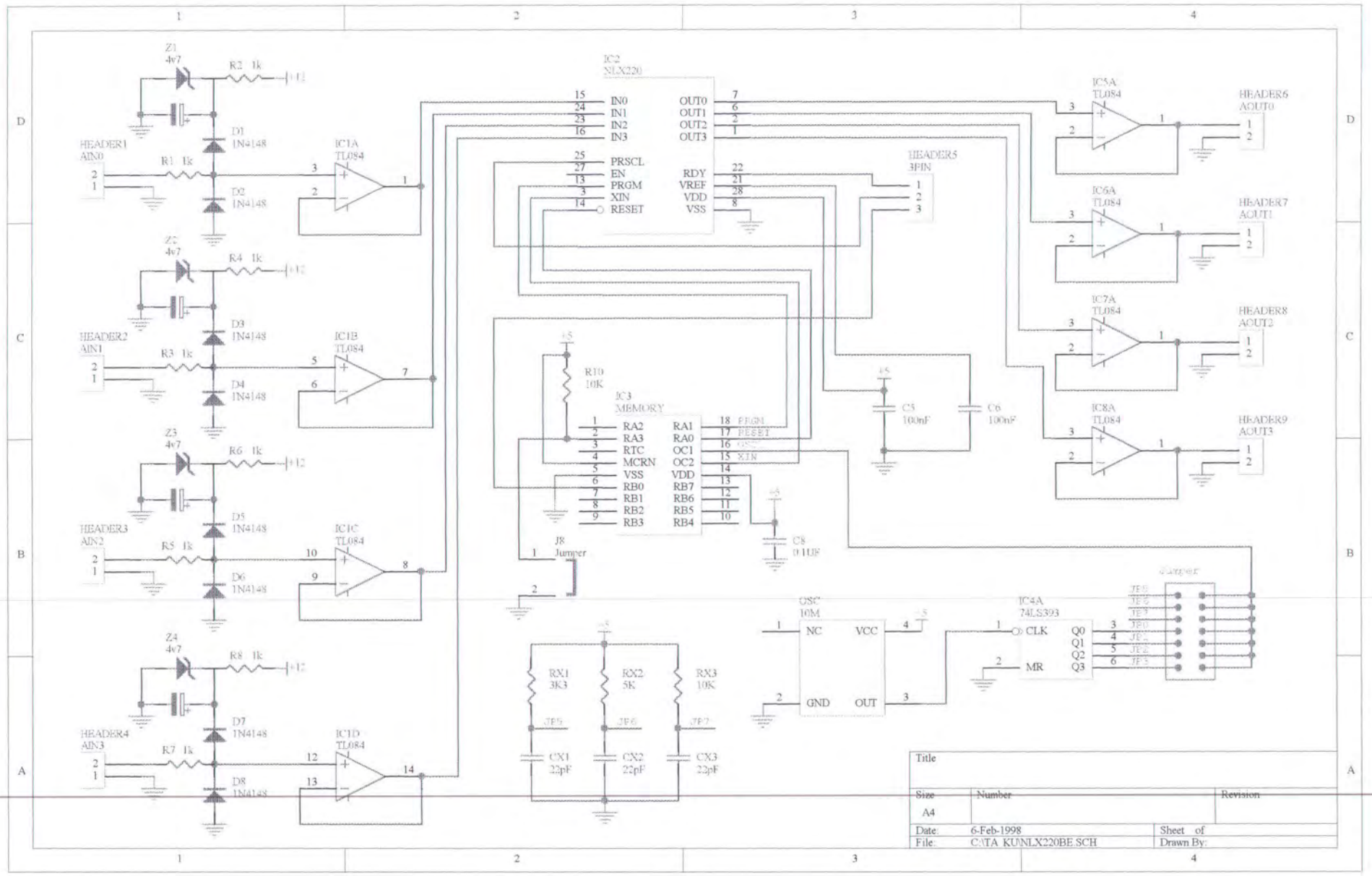
Yan, J., Ryan, M., Power, J., 1994, **Using Fuzzy Logic**, *Towards Intelligent System*, Prentice/hall International, Inc., Englewood Cliffs, New Jersey.

---, 1992, ELEX nomor1 paket 8, PT. Elex Media Komputindo, Kelompok Gramedia, Jakarta.

---, 1984, "IBM PC AT Technical Reference", IBM.

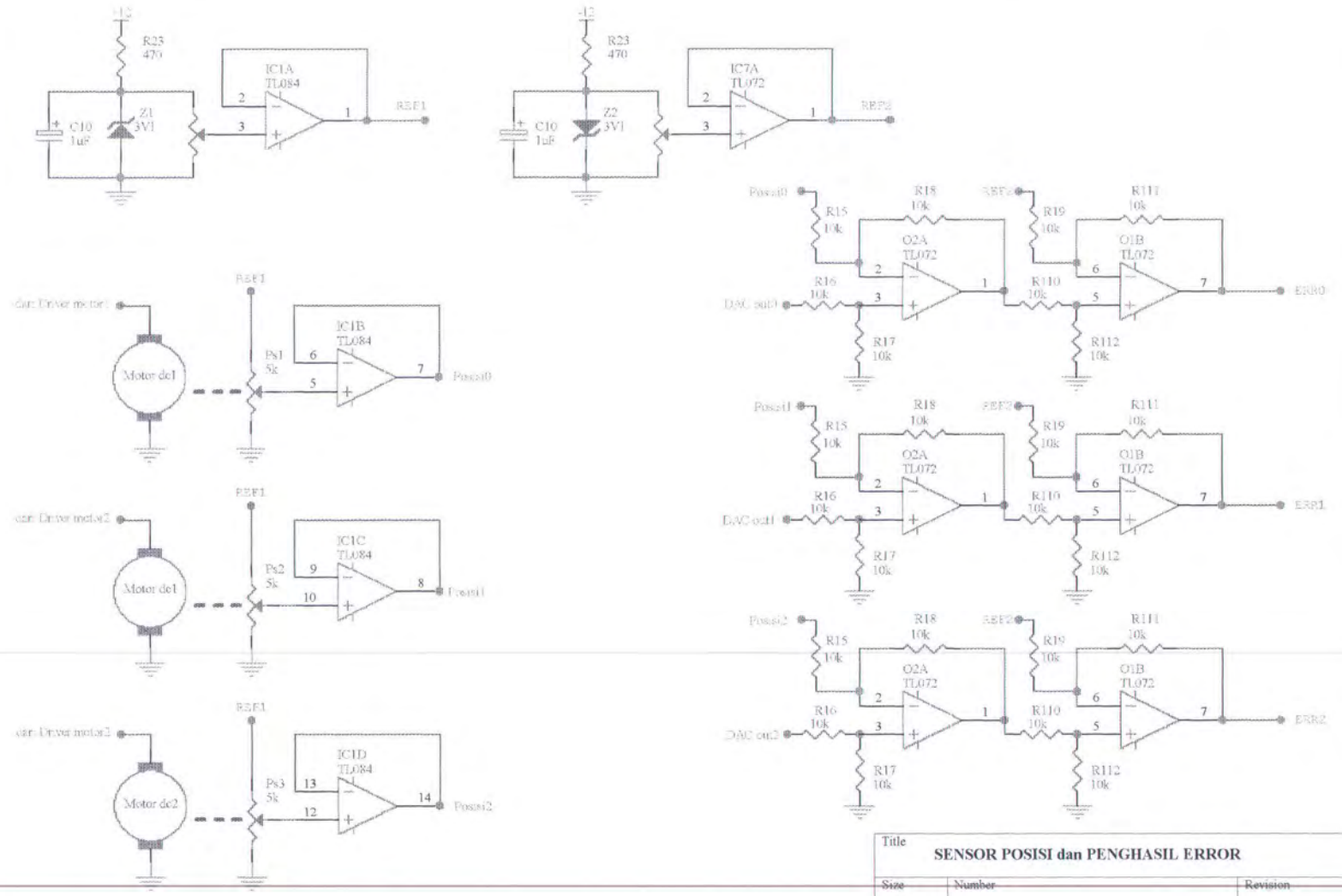
---, 1994, NLX220, Stand Alone Fuzzy Logic Controllers, Adaptive Logic, Inc.

LAMPIRAN

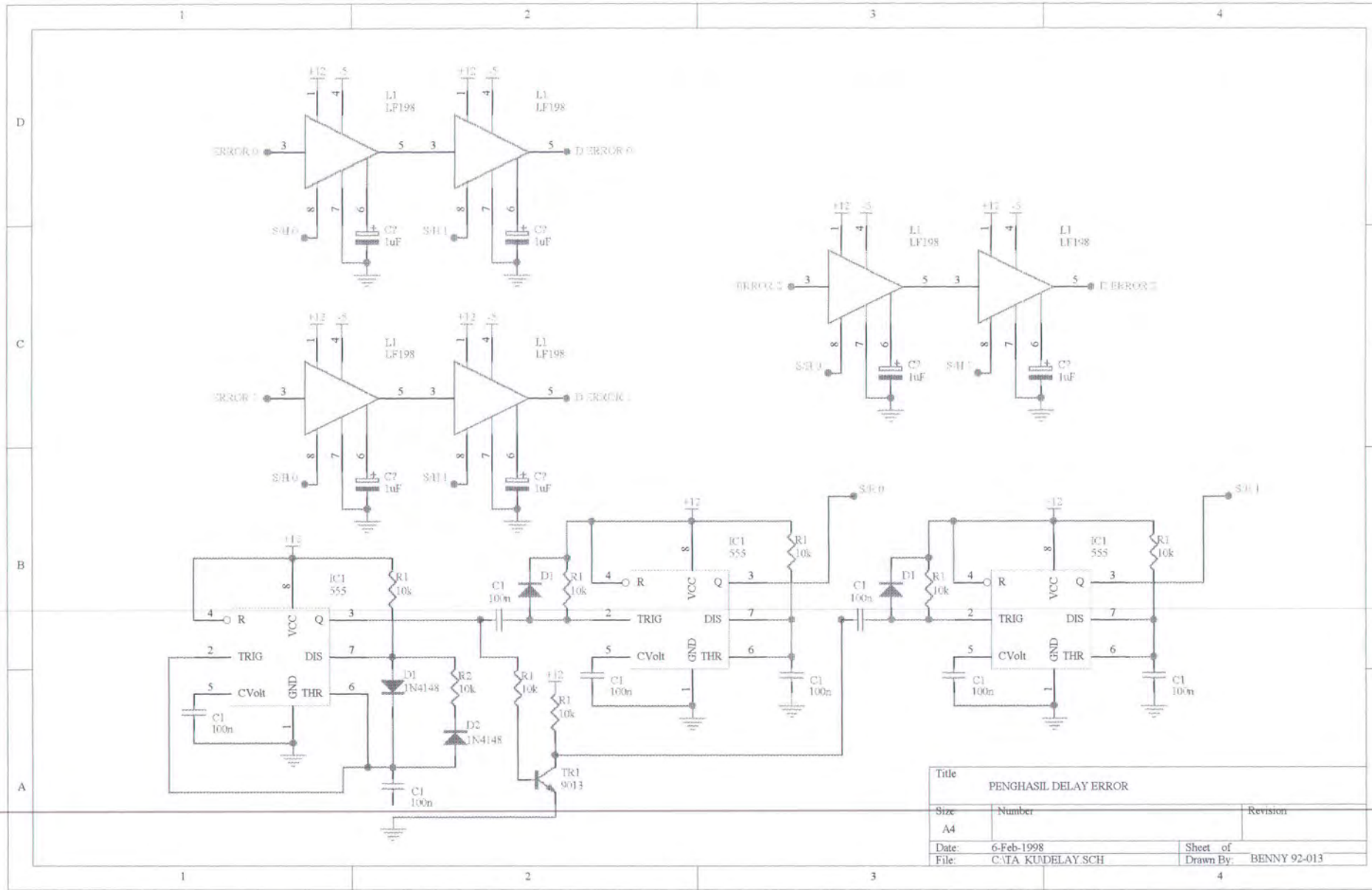


Lampiran 1. Skematik Modul NLX220

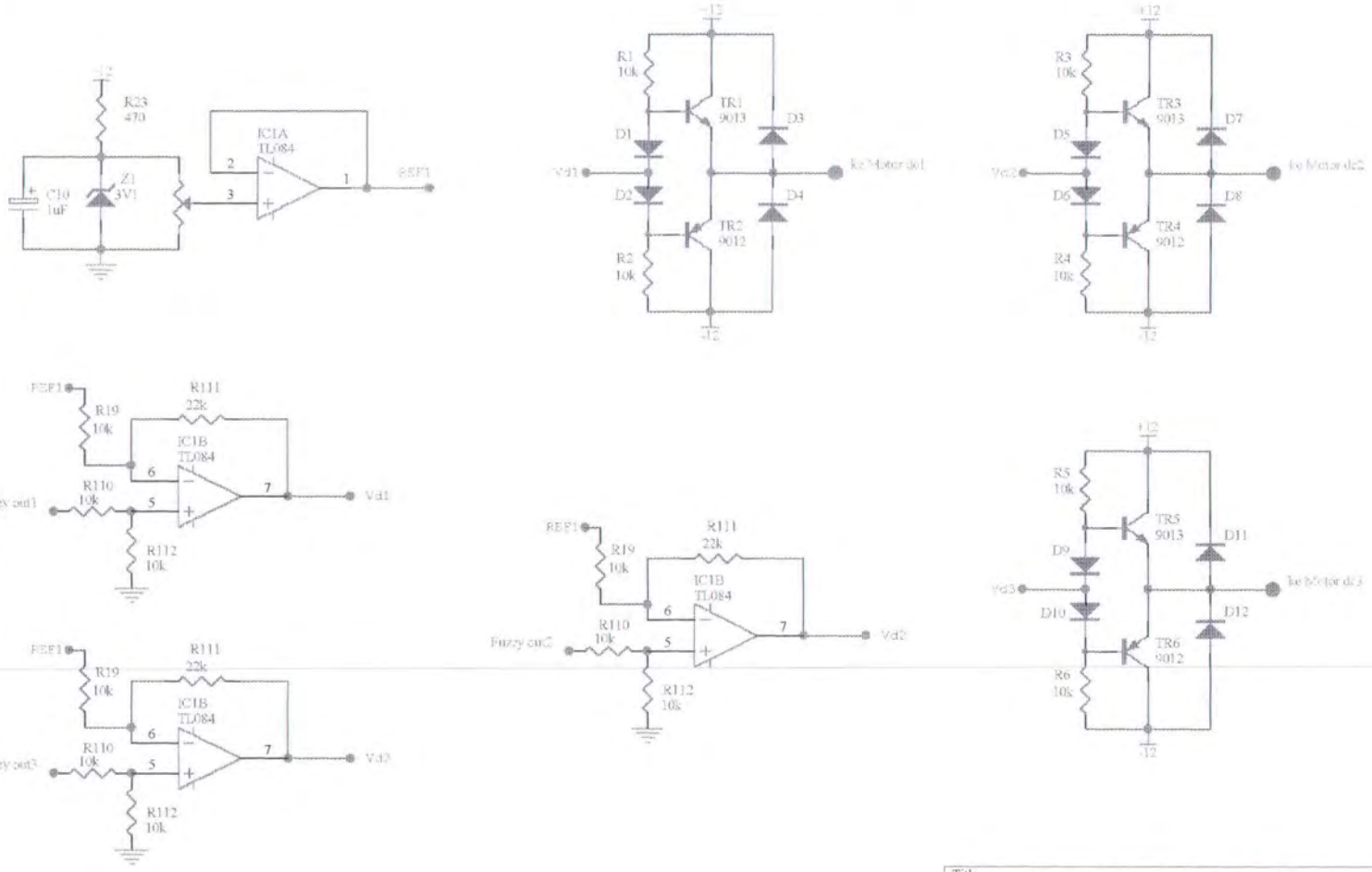
Lampiran 2. Skematik Rangkaian Sensor Posisi Dan Penghasil Error



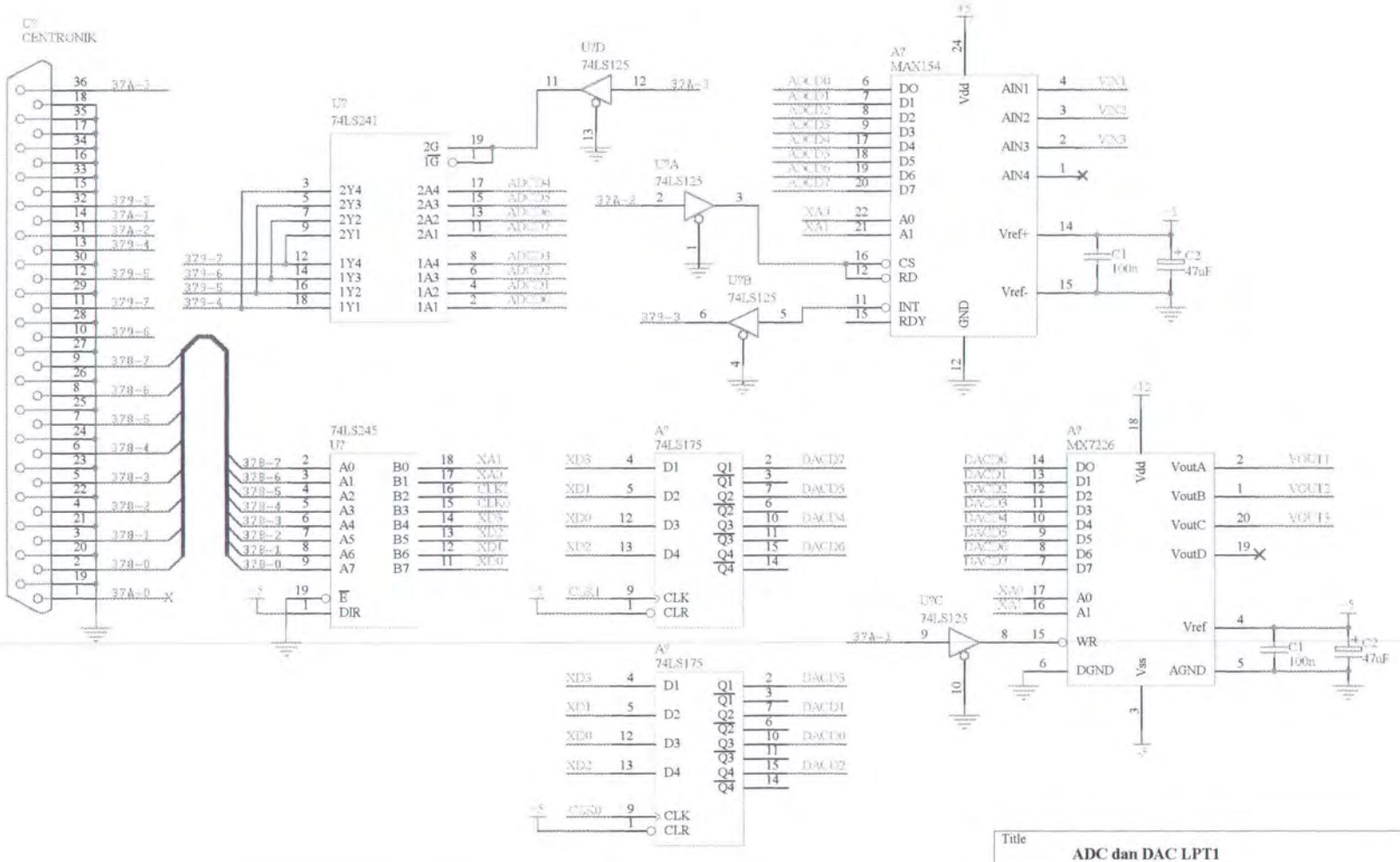
Lampiran 3. Skematik Rangkaian Penghasil Delay Error



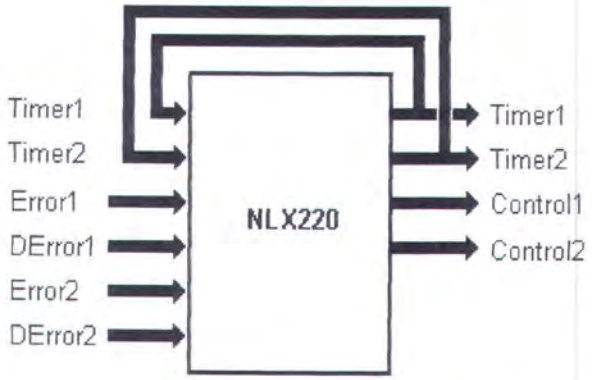
Lampiran 4. Skematik Rangkaian Driver Motor DC



Title	**RANGKAIAN DRIVER MOTOR DC**		
Size: A4	Number:		Revision:
Date:	6-Feb-1998	Sheet of	
File:	C:\TA_KU\DRIVERM.SCH	Drawn By:	BENNY


C9
CENTRONIK
U?
74LS241
U?D
74LS125
U?A
74LS125
U?B
74LS125
A?
MAX154
74LS245
U?
A?
74LS175
A?
74LS175
A?
MX7226
U?C
74LS125
Title
ADC dan DAC LPT1
Size
A4
Number
Revision
Date: 6-Feb-1998
Sheet of
File: C:\TA_KUL\PTTA.SCH
Drawn By: BENNY

Lampiran 6. Fuzzy Model Untuk Kontroler 2 Joint



Inputs

Timer1 (Timer1)

Timer2 (Timer2)

Error1

DError1

Error2

DError2

Outputs

Timer1

Timer2

Control1

Control2

Fuzzy Variables

Timer1 is Zerro (0, 1, symmetric inclusive)

Timer1 is NotMax (249, 0, left inclusive)

Timer1 is Max (250, 0, right inclusive)

Timer2 is Zerro (0, 1, symmetric inclusive)

Timer2 is NotMax (249, 0, left inclusive)

Timer2 is Max (250, 0, right inclusive)

Error1 is Zerro (128, 2, symmetric inclusive)

Error1 is SmNeg (124, 4, symmetric inclusive)

Error1 is SmPos (132, 4, symmetric inclusive)

Error1 is MedNeg (116, 6, symmetric inclusive)

Error1 is MedPos (140, 6, symmetric inclusive)

Error1 is LrgNeg (100, 12, symmetric inclusive)

Error1 is LrgPos (156, 12, symmetric inclusive)

Error1 is MaxNeg (128, 24, right exclusive)

Error1 is MaxPos (128, 24, left exclusive)

Error1 is NoChg (DError1, 2, symmetric inclusive)

Error1 is SmDrop (DError1, 4, right exclusive)

Error1 is SmRise (DError1, 4, left exclusive)

Error1 is MedDrop (DError1, 6, right exclusive)

Error1 is MedRise (DError1, 6, symmetric inclusive)

Error1 is LrgDrop (DError1, 12, right exclusive)

Error1 is LrgRise (DError1, 12, symmetric inclusive)

Error1 is MaxDrop (DError1, 24, right exclusive)

Error1 is MaxRise (DError1, 24, left exclusive)

Error2 is Zerro (128, 2, symmetric inclusive)

Error2 is SmNeg (124, 4, symmetric inclusive)

Error2 is SmPos (132, 4, symmetric inclusive)

Error2 is MedNeg (116, 6, symmetric inclusive)

Error2 is MedPos (140, 6, symmetric inclusive)

Error2 is LrgNeg (100, 12, symmetric inclusive)

Error2 is LrgPos (156, 12, symmetric inclusive)

Error2 is MaxNeg (128, 24, right exclusive)

Error2 is MaxPos (128, 24, left exclusive)

Error2 is NoChg (DError2, 2, symmetric inclusive)

Error2 is SmDrop (DError2, 4, right exclusive)

Error2 is SmRise (DError2, 4, left exclusive)

Error2 is MedDrop (DError2, 6, right exclusive)

Error2 is MedRise (DError2, 6, symmetric inclusive)

Error2 is LrgDrop (DError2, 12, right exclusive)

Error2 is LrgRise (DError2, 12, symmetric inclusive)

Error2 is MaxDrop (DError2, 24, right exclusive)

Error2 is MaxRise (DError2, 24, left exclusive)

Rules

// Start awal Control1:

If Timer1 is Zerro then Control1 = 128

If Timer1 is NotMax then Contro1l + 0

// Bagian proporsional Control1:

If Error1 is Zerro then Control1 = 128

If Error1 is MaxPos then Control1 = 255

If Error1 is MaxNeg then Control1 = 0

If Error1 is LrgPos then Control1 = 233

If Error1 is LrgNeg then Control1 = 23

If Error1 is MedPos then Control1 = 173

If Error1 is MedNeg then Control1 = 83

If Error1 is SmdPos then Control1 = 143

If Error1 is SmNeg then Control1= 113

// Rule pemisah:

If Timer1 is NotMax then Control1 + 0

// Bagian Derivativ Control1:

If Error1 is NoChg then Control1 + 0

If Error1 is MaxRise then Control1 + 40

If Error1 is MaxDrop then Control1 - 40

If Error1 is LrgRise then Control1 + 20

If Error1 is LrgDRop then Control1 - 20

If Error1 is MedRise then Control1 + 6

If Error1 is MedDrop then Control1 - 6

If Error1 is SmdRise then Control1 + 2

If Error1 is SmDrop then Control1 - 2

// Pengatur kecepatan proses Control1:

If Timer2 is NotMax then Timer2 + 20

If Timer1 is Max then Timer1 = 1

If Timer1 is NotMax then Timer1 + 20

// Start awal Control2:

If Timer2 is Zerro then Control2 = 128

If Timer2 is NotMax then Contro2 + 0

// Bagian proporsional Control2:

If Error2 is Zerro then Control2 = 128

If Error2 is MaxPos then Control2 = 255

If Error2 is MaxNeg then Control2 = 0

If Error2 is LrgPos then Control2 = 233

If Error2 is LrgNeg then Control2 = 23

If Error2 is MedPos then Control2 = 173

If Error2 is MedNeg then Control2 = 83

If Error2 is SmdPos then Control2 = 143

If Error2 is SmNeg then Control2 = 113

// Rule pemisah:

If Timer2 is NotMax then Control2 + 0

// Bagian Derivativ Control2:

If Error2 is NoChg then Control2 + 0

If Error2 is MaxRise then Control2 + 40

If Error2 is MaxDrop then Control2 - 40

If Error2 is LrgRise then Control2 + 20

If Error2 is LrgDRop then Control2 - 20

If Error2 is MedRise then Control2 + 6

If Error2 is MedDrop then Control2 - 6

If Error2 is SmdRise then Control2 + 2

If Error2 is SmDrop then Control2 - 2

// Pengatur kecepatan proses Control2:

If Timer2 is Max then Timer2 = 1

DAFTAR RIWAYAT HIDUP

Benedictus Indrajaya lahir di desa Cerme Kecamatan Grogol Kediri Jawa Timur pada tanggal 27 Mei 1972, putra dari FX. Sudarsono dan MM. Tri Rahayu. Menempuh pendidikan dasar di SDN Grogol I Kediri dan lulus tahun 1985. Pendidikan Menengah di SMPK Putra Kediri lulus pada tahun 1988. Melanjutkan pendidikan atas di SMAK St. Albertus Malang dan lulus pada tahun 1991. Kemudian diterima menjadi mahasiswa di Teknik Elektro ITS pada tahun 1992 dan diharapkan lulus pada ujian tugas akhir periode wisuda Maret 2000. Selama berada di ITS dan jurusan Teknik Elektro penulis aktif di salah satu Unit Kegiatan Mahasiswa dengan menjadi ketua UK Aeromodeling, dan di kegiatan kemahasiswaan Himpunan Mahasiswa teknik Elektro dengan menjadi pengurus serta ketua panitia Lomba Cipta Elektroteknik Nasional 1996. Di bidang studi Elektronika penulis pernah menjadi asisten praktikum Rangkaian Listrik dan praktikum Elektronika.