PENGATURAN KIPAS ANGIN BERDASARKAN SUHU RUANGAN MENGGUNAKAN LOGIKA FUZZY, BERBASIS MIKROKONTROLER AT89C52

TUGAS AKHIR

Diajukan sebagai Salah Satu Syarat
untuk Memperoleh Gelar Sarjana Teknik Elektro

oleh :

Nama : **Udin Saefudin**

No. Mahasiswa : **98 524 006**

JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM INDONESIA
JOGYAKARTA

2004

LEMBAR PENGESAHAN PEMBIMBING

TUGAS AKHIR

PENGATURAN SUHU RUANGAN DENGAN KIPAS ANGIN MENGGUNAKAN LOGIKA FUZZY, BERBASIS MIKROKONTROLER AT89C52

Oleh :

Nama : UDIN SAEFUDIN

No. Mahasiswa : 98.524.006

Yogyakarta, 08 Mei 2004

Pembimbing I,	Pembimbing II,
Ir. Hj. Budi Astuti	RM. Sisdarmanto Adinandra, ST

LEMBAR PENGESAHAN PENGUJI

PENGATURAN KIPAS ANGIN BERDASARKAN SUHU RUANGAN MENGGUNAKAN LOGIKA FUZZY, BERBASIS MIKROKONTROLER AT89C52

TUGAS AKHIR

Oleh :

Nama : **Udin Saefudin**

No. Mahasiswa : 98 524 006

Telah Dipertahankan Di Depan Penguji sebagai Salah satu Syarat untuk Memperoleh Gelar Sarjana Jurusan Teknik Elektro Fakultas Teknologi Industri Universitas Islam Indonesia

Jogjakarta, 08 Mei 2004

Tim Penguji	**Tanda Tangan**
Ir. Hj. Budi Astuti **Ketua**	
RM Sisdarmanto Adinandra, ST **Anggota I**	
Hendra Setiawan, ST **Anggota II**	

Mengetahui
Dekan Fakultas Teknologi Industri
Universitas Islam Indonesia

UNIVERSITAS ISLAM INDONESIA ★ YOGYAKARTA ★ FAKULTAS TEKNOLOGI INDUSTRI

(Ir. Bachrun Sutrisno, M.Sc)

HALAMAN PERSEMBAHAN

Kupersembahkan Tugas Akhir ini untuk Islam Agamaku,
Universitas Islam Indonesia, Almamaterku

Tugas Akhir ini kupersembahkan kepada kedua orangtuaku :
"Ayahanda H .Jupri & Ibunda Tersayang Hj .Rantisah"
Kepada adikku :
"Eneng, Ene dewi ,Efe, susu sulasiah"
serta keluarga besarku dan semua orang yang kucintai

*Tugas Akhir ini kupersembahkan kepada *Kake dan Neneiku* H .Sukanta, Hj.Ratnawati.*
Om abdurahman. Om Aan Yang baru menempuh Hidup Baru.

Tugas Akhir Ini Kupersembahkan kepada My Honey Hannah Derwent (Hanster) yang begitu besar kasih sayangnya dalam mendampingiku, serta pengorbananya yang tulus ikhlas, juga yang telah memberikan rasa tenang dan damai
dalam perjuangan menuntut ilmu dan masa depanku.

Tugas Akhir ini kupersembahkan kepada teman-teman seperjuangan "Rona , Doni, Wahab, Feri ST,Bondit, Andre ST, Lia ST, bayu, eldi, thanks for all dan semua teman-teman yang telah membantu memberikan dukungan serta motivasi untuk terus maju demi kesuksesanku

MOTTO

"Barang siapa menempuh jalan untuk mencari ilmu, maka Allah memudahkan jalan baginya menuju surga"

(HR. Muslim dan Abu Hurairah ra)

"...Katakanlah : Adakah sama orang-orang yang mengetahui dengan orang-orang yang tidak mengetahui? Sesungguhnya orang yng berakallah yang dapat menerima pelajaran"

(QS Az-Zumar : 9)

"Jadikanlah sabar dan sholat sebagai penolongmu. Dan sesungguhnya yang demikian itu sungguh berat, kecuali bagi orang-orang yang khusyu"

(QS Al-Baqarah : 45)

"Jadikanlah engkau pemaaf dan suruhlah orang mengerjakan yang ma'ruf, serta berpalinglah daripada orang-orang yang bodoh"

(QS Al-A'raaf : 199)

"Pandanglah kegagalan sebagai suatu peluang untuk belajar, sebagai suatu lompatan kreativitas, sebagai suatu kesempatan untuk menguji gagasan baru"

(Art Mortell)

"Derajat kemuliaan seseorang dapat dilihat dari sejauh mana dirinya punya nilai manfaat bagi orang lain"

(HR. Bukhori)

KATA PENGANTAR

Alhamdulillah, puji dan syukur kepada Allah SWT Tuhan semesta alam. Yang Maha Pengasih lagi Maha Penyayang. Ucapan syukur kehadirat-Nya akhirnya penulis dapat menyelesaikan Tugas Akhir ini sebagai syarat akhir untuk meraih gelar Sarjana Teknik di jurusan Teknik Elektro Universitas Islam Indonesia. Sholawat serta salam penulis haturkan kepada pemimpin umat, Nabi Muhammad SAW beserta para keluarganya, sahabatnya, dan semua umatnya yang tetap sedia menjalankan ajaran Islam. Semoga kita termasuk di dalamnya. Amin.

Penyusunan Tugas Akhir dengan judul "**Pengatur Kipas Angin Berdasarkan Suhu Ruangan Menggunakan Logika Fuzzy**" tidak terlepas dari bantuan semua pihak, baik berupa bimbingan, kritik, saran maupun do'a. Tugas Akhir ini adalah langkah pertama dari sekian ribu langkah menuju kesempurnaan maka dari itu sangat diharapkan ada teman dan adik-adik penulis yang meneruskan apa yang sudah penulis lakukan hingga tahap ini. Untuk itu, dalam kesempatan ini dengan segala kerendahan hati dan penghargaan yang tulus, penulis menghaturkan banyak terima kasih kepada:

1. Bapak Ir. Bachrudin Sutrisno, Msc selaku Dekan Fakultas Teknologi Industri Universitas Islam Indonesia.
2. Ibu Ir. Hj. Budi Astuti, selaku Ketua Jurusan Teknik Elektro Universitas Islam Indonesia dan Dosen Pembimbing I Tugas Akhir.

3. Bapak RM Sisdarmanto Adinandra ST, selaku Dosen Pembimbing II Tugas Akhir.
4. Segenap Dosen Teknik Elektro Universitas Islam Indonesia.
5. Segenap hormat kuserahkan pada kedua orang tuaku. H. Jupri dan Hj Rantisah, ribuan kilo jalan yang beliau tempuh lewati rintangan untuk aku anaknya, yang selalu memberikan kasih sayang, bimbingan, arahan dan do,a dalam setiap langkah dalam menuntut ilmu.
6. Untuk adik-adikku tercinta Eneng, Ene, Efe dan Ciyah yang telah memberikan motivasi.
7. Yang terkasih dinegari sebrang Hannah Derwent (Hanster), yang telah memberikan semangat yang tiada henti.
8. Untuk kawan-kawanku tercinta, Feri ST, Wahab, Doni, Eldi ST, Onay, Mas Enjen, Mas Yunus.
9. Untuk teman-teman angkatan 98' satu perjuangan maju terus pantang mundur, yang telah bersaing dengan sehat sehingga kita akan bertemu sepuluh tahun yang akan datang.

Kesempurnaan hanya milik Allah, oleh karena itu penulis mengharapkan saran dan masukan demi perbaikan tulisan ini dan semoga bermanfaat.

Yogyakarta, 08 Mei 2004

Penulis

Udin Saefudin

DAFTAR ISI

DAFTAR GAMBAR

Halaman

DAFTAR TABEL

Halaman

ABTRAKSI

Pengendalian temperatur banyak dimanfaatkan dalam bidang industri, baik untuk mempercepat suatu proses ataupun untuk memperlambat suatu proses. Oleh karenanya dibutuhkan sutau pengendali temperatur yang dapat beroperasi secara otomatis dan mampu mengendalikan cara "cerdas" model ruang suhu. Karakterstik suatu plant temperature adalah tidak linear, sehingga pengendali konvensional seperti pengendali proporsional, pengendali diferensial dan pengendali intgral maupun gabungan kombinasi ketiganya tidak sesuai untuk sistem yang tidak linier.

Dalam penelitian ini akan dirancang sistem pengendali temperatur ruangan yang berbasis algoritma *fuzzy* dengan pengendali mikrokontroler AT89C52 untuk mengendalikan kecepatan putaran kipas berdasarkan suhu ruangan.

Logika *fuzzy* adalah menerapkan pengendalian berdasarkan pengkaburan, atau suatu cara yang tepat untuk memetakan suatu ruang input kedalam suatu ruang output. Sehingga sesuai untuk karakter sistem yang tidak linier, pada perkembanganya logika *fuzzy* diterapkan dalam bidang kendali yang dikenal dengan istilah pengendali logika *fuzzy* (PLC). Algoritma pengendali *fuzzy* dapat diterapkan untuk mikrokontroler karena tidak memerlukan perhitungan matematis yang rumit sehingga dapat diimplementasikan dengan cepat.

Setpoint temperatur tidak pernah tercapai tapi akan beselisih beberapa derjat celcius terhadap suhu yang diinginkan (setpoint). Hal ini diakibatkan karena tidak adanya pengaturan laju pemanasan dan aturan-aturan logika *fuzzy* yang diterapkan. Putaran kipas akan berjalan ketika terjadi beda suhu atau temperatur antara setpoint dan suhu awal satu derajat celcius. Hal ini terjadi karena aturan fuzzifikasi yang dibuat.

BAB I

I.1. Latar belakang

Pada jaman sekarang ini peralatan kipas angin dapat dikendalikan dari jarak jauh atau dengan cara menggunakan *remote control,* didalam peralatan elektronik seperti kipas angin sudah dilengkapi dengan fasilitas pengatur kecepatan dan pengatur suhu namun masih dioperasikan secara manual.

Pabrik, industri, instansi perkantoran, hotel dan fasilitas lainya rata-rata telah menggunakan sistem pengendali suhu ruangan pada umumnya yang diaplikasikan menggunakan kipas angin untuk menjaga suhu, maka kipas angin yang berputar tidak ditentukan berdasarkan suhu yang diinginkan selama itu mekanisme kendali masih manual (mengubah-ubah tombol).

Pengendalian kecepatan putaran kipas angin dapat dilakukan secara otomatis. Kecepatan putaran bedasarkan bacaan sensor suhu dan pengolah mikrokontroler untuk mengendalikan putaran kipas.

Penelitian ini akan diimplementasikan secara digital berdasarkan pertimbangan bahwa kendali digital lebih tahan terhadap derau daripada kendali analog.

Pengolah mikrokontroler yang digunakan adalah mikrokontroler AT89C52 dengan kendali berbasis logika *fuzzy*

I.2. Rumusan masalah

Berdasarkan latar belakang diatas, maka permasalahan dapat di idenfikasi sebagai berikut:

1. Bagaimana mengatur kecepatan kipas angin sesuai dengan suhu ruangan
2. Bagaimana merancang perangkat keras dan perangkat-perangkat pendukung lainya, untuk mengatur kecepatan kipas atau merendahkan kipas angin.
3. Merancang dan mengimplementasikan logika *fuzzy* dengan menggunakan mikrokontroler AT89C52, yang dapat mengatur kipas angin, sesuai dengan suhu ruangan.

I.3. Batasan Masalah

Berdasarkan permasalahan yang ada diatas maka masalah yang difokuskan pada perancangan sistem pengendali putaran kipas angin berdasarkan suhu ruangan maka dibuat batasan masalah sebagai berikut:

1. Menggunakan sistem minimal mikrokontroler AT89C52 sebagai pengendali kecepatan putaran kipas angin berdasarkan suhu ruangan disertai model ruangan, keypad, 7 segment sebagai penampil setpoint dan suhu.
2. Menggunakan bahasa *assembler* untuk merancang program kendali
3. Penerapan logika *fuzzy* dengan tiga buah himpunan *fuzzy* berbentuk segitiga untuk mengendalikan putaran kipas angin berdasarkan suhu ruangan.
4. Rentang suhu yang digunakan adalah 25^0C sampai dengan 55^0C.

5. Motor yang dipakai motor AC rating 220V. Dan tidak mengukur kecepatan motor

I.4. Tujuan Pembuatan Tugas Akhir

Tujuan pembuatan tugas akhir ini untuk

1. Merancang dan membuat alat untuk mengatur kecepatan kipas angin.
2. Menggunakan mikrokontroler AT89C52 dengan logika *fuzzy* untuk mengendalikan kecepatan kipas angin berdasarkan suhu ruangan.

I.5. Sistematika Penulisan

BAB I : Pendahuluan

Berisi tentang latar belakang masalah, batasan masalah, tujuan penulisan, dan sistematika

BAB II : Landasan Teori

Bab ini memuat teori-teori yang berhubungan dengan pengendalian putaran kipas angin.

BAB III: Perancangan Sistem

Berisi tentang sistem perangkat keras AT89C52, perancang sistem kendali *fuzzy* untuk kendali putaran kipas angin.

BAB IV : Hasil Pengamatan dan Pembahasan

Bab ini berisi pengamatan hasil rancangan sitem pengendali dalam pengujian sistem serta pengamatanya.

BAB V : Kesimpulan dan Saran

Bab ini memuat kesimpulan dari pembahasan dan saran dari hasil yang diperoleh.

BAB II

DASAR TEORI

2.1. Himpunan *Fuzzy*

Himpunan *fuzzy* yang menjadi dasar dari sistem *fuzzy* merupakan pengembangan dari teori tradisional yang hanya membagi keanggotaannya menjadi dua keadaan saja yaitu aktif atau 1 dan tidak aktif atau 0. Teori klasik ini kemudian dimodifikasi pertama kali oleh Zadeh menjadi suatu teori himpunan yang beranggotakan nilai pada interval dari 0 sampai 1. Himpunan *fuzzy* ini lebih kompleks dari teori sebelumnya karena memiliki lebih banyak informasi, namun ketepatanya lebih tinggi dengan prosedur dan aturan yang mendasarnya.

Untuk mendefinisikan suatu himpunan, setiap elemen pada semesta wacana harus diberi nilai yang menunjukkan apakah elemen tersebut dalam himpunan atau tidak. Untuk himpunan bukan *fuzzy*, nilai ini terbatas pada 0 dan 1, yang menunjukkan anggota himpunan dan bukan anggota himpunan. Untuk suatu semesta wacana U, himpunan *fuzzy* ditentukan oleh fungsi keanggotaan yang memetakan anggota-anggota semesta wacana U ke suatu derajat yang nilainya berada pada interval 0 dan 1. Representasi himpunan *fuzzy* F dalam U ditunjukkan dalam pasangan elemen dan tingkat dari nilai keanggotaan:

$$F = \{(u, \mu_F(u)) \mid u \in U\} \tag{2.1}$$

Apabila U kontinyu, himpunan *fuzzy* F ditulis sebagai

$$F = \int_U \mu_F(u)/u \tag{2.2}$$

Dan bila U diskret, himpunan *fuzzy* ditunjukkan dalam

$$F = \Sigma \mu_F(u_i)/u_i \qquad (2.3)$$

Himpunan *fuzzy* mempunyai dua cara untuk mendefinisikan keanggotaannya, numeris dan fungsional. Definisi numeris menunjukkan derajat fungsi keanggotaan sebagai vektor nilai dengan dimensi sesuai tingkat diskretisasi, sedangkan secara fungsional ditunjukkan dalam persamaan analitik untuk tiap elemen yang didefinisikan. Fungsi keanggotaan yang sering dipakai yaitu fungsi S dan π (*gaussian)*, segitiga dan *trapesium,* serta eksponensial.

2.2. Fuzzifikasi

Fuzzifikasi adalah proses pemetaan dari variabel masukan ke dalam himpunan *fuzzy* dalam suatu semesta wacana. Proses fuzzifikasi ditunjukkan melalui persamaan:

$$X = \text{fuzzifier}\,(X_0) \qquad (2.4)$$

Xo merupakan vektor masukan himpunan tegas, x adalah vektor himpunan *fuzzy* hasil proses fuzzifikasi, dan fuzzifier adalah operator fuzzifikasi.

Beberapa strategi fuzzifikasi:

a. *Fuzzy singleton.* Nilai *fuzzy* ditunjukkan melalui suatu nilai tegas. Strategi ini yang paling sering digunakan karena natural dan paling mudah diimplementasikan.
b. *Fuzzy number.* Nilai *fuzzy* diberikan melalui suatu nilai tertentu dengan interval kepastian. *Fuzzy* number merupakan nilai subjektif karena nilai yang diberikan adalah penilaian, bukan hasil pengukuran.

c. *Fuzzy random.* Nilai yang dihasilkan merupakan nilai random yang bisa dikatakan sebagai nilai objektif karena menggunakan teori probabilitas dalam merepresentasikan data.

2.3. Sistem Pengambilan Keputusan (*Fuzzy Inference System*) dan Teknik *Reasoning*

Inferensi *fuzzy* dalam hal ini digunakan untuk merumuskan pemetaan himpunan input ke himpunan output dengan prinsip logika *fuzzy*. Inferensi ini menjadi basis peraturan dalam pengambilan keputusan. Aturan yang digunakan menggunakan aturan jika-maka (*If-Then*).

Dalam penalaran logika *fuzzy* ada dua tipe utama inferensi *fuzzy*:

1. Inferensi berdasar komposisi. Semua aturan dalam inferensi ini dikombinasikan dalam suatu relasi *fuzzy* dan dilihat sebagai sebuah aturan *fuzzy* jika-maka tunggal
2. Inferensi berdasar aturan individu. Setiap aturan menghasilkan sebuah keluaran himpunan *fuzzy* dan secara keseluruhan merupakan komposisi keluaran himpunan *fuzzy* yang dihasilkan setiap aturan.

Teknik *reasoning* digunakan untuk menentukan nilai yang akan digunakan sebagai masukan aksi kendali yang tepat. Dua buah teknik *reasoning* yang paling sering digunakan adalah MAX-MIN dan MAX-DOT. Untuk A' masukan himpunan *fuzzy* dan B' keluaran himpunan *fuzzy*, perumusan kedua metode sebagai berikut:

2.3.1. Metode MAX-MIN

Inferensi maksimum-minimum dapat dijelaskan dengan permasalahan sebuah basis aturan dengan dua aturan:

Aturan 1 jika x adalah A1 dan y adalah B1 maka z adalah C1

Aturan 2 jika x adalah A2 dan y adalah B2 maka z adalah C2

Setiap aturan diberi bobot dengan kuat bobot tertentu, kuat bobot ini sangat dipengaruhi oleh operator penghubung dalam aturan. Jika aturan dihubungkan dengan operator 'dan' maka operasi yang dilakukan adalah irisan (minimum), sedangkan jika aturan dihubungkan dengan operator 'atau' maka operasi yang dilakukan adalah gabungan (maksimum), jika kuat bobot dari aturan ke-i dilambangkan dengan αi, masukan *fuzzy* adalah xo dan yo maka keanggotan xo adalah himpunan A dilambangkan dengan $\mu A(x_0)$.

$$\alpha_1 = \mu A_1(x_0) \wedge \mu B_1(y_0). \quad (2.5)$$

$$\alpha_2 = \mu A_2(x_0) \wedge \mu B_2(y_0). \quad (2.6)$$

'$\wedge$' adalah operator 'dan' atau minimum dan kuat penyulutan untuk aturan-aturan diatas adalah:

$$\mu C(w) = (\alpha_1 \wedge \mu C_1(w)) \vee (\alpha_2 \wedge \mu C_2(w)) \quad (2.7)$$

Dengan $\mu C(w)$ adalah derajat angota keluaran.

2.3.2. Metode MAX-DOT

Metode ini mengunakan operasi perkalian sebagai implikasi *fuzzy*, maka didapat persamaan sebagai berikut:

$$\mu C(w) = (\alpha_1 . \mu C_1(w)) \vee (\alpha_2 . \mu C_2(w)) \qquad (2.8)$$

2.4. Defuzzifikasi

Defuzzifikasi didefinisikan sebagai proses pemetaan dari himpunan *fuzzy* hasil inferensi ke dalam aksi kendali *non fuzzy*. Metode yang sering digunakan dalam defuzzifikasi adalah metode rerata maksimum (*mean of maxsimum*/MOM) dan metode rerata pusat (*center of average*/COA).

2.4.1. Metode rerata maksimum (MOM)

Metode ini memberikan keluaran pengendali dengan rumusan:

$$y^* = \frac{\sum_{j=1}^{n} \alpha_j H_j W_j}{\sum_{j=1}^{n} \alpha_j H_j} \qquad (2.9)$$

Dengan

- y^* = adalah hasil defuzzifikasi,
- H_j = adalah ketinggian maksimum fungsi keanggotaan keluaran untuk aturan ke-j,
- W_j = merupakan titik berat keanggotaan keluaran aturan ke-j,
- α_j = adalah kuat penyulutan (*fire strength*) pada aturan ke-j,
- n = menunjukkan jumlah aturan kendali.

2.4.2. Metode rerata pusat (COA)

Metode rerata pusat memberikan hasil defuzzifikasi (y*) dengan persamaan:

$$y^* = \frac{\sum_{j=1}^{n} \alpha_j M_j}{\sum_{j=1}^{n} \alpha_j A_j} \qquad (2.10)$$

dengan

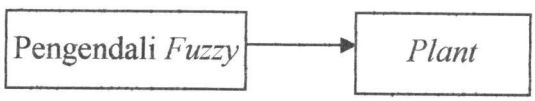
- α_j = adalah kuat penyulutan,
- M_j = momen fungsi keanggotaan himpunan *fuzzy* keluaran untuk aturan ke-j, dan
- A_j = merupakan luas daerah keluaran untuk aturan ke-j.

2.5. Pengendali Logika *Fuzzy* Proporsional dan Integral

Secara teoritis, tidak ada pengelompokan logika *fuzzy* ke dalam logika *fuzzy* proporsional dan integral. Pengelompokan ini dilakukan pada pengendali logika *fuzzy* berdasarkan hasil yang akan diperoleh.

2.5.1. Pengendali logika *fuzzy* proporsional

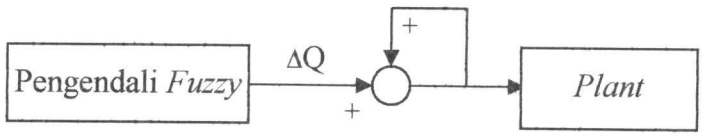
Pengendali ini mempergunakan logika *fuzzy* untuk memperoleh keluaran yang langsung akan diumpankan kepada penguat atau kepada sistem terkendali.Tanggapan sistem pada pengendali logika *fuzzy* proporsional merupakan respon terhadap keluaran pengendali pada saat tersebut.



Gambar 2.1 Struktur pengendali logika *fuzzy* proporsional

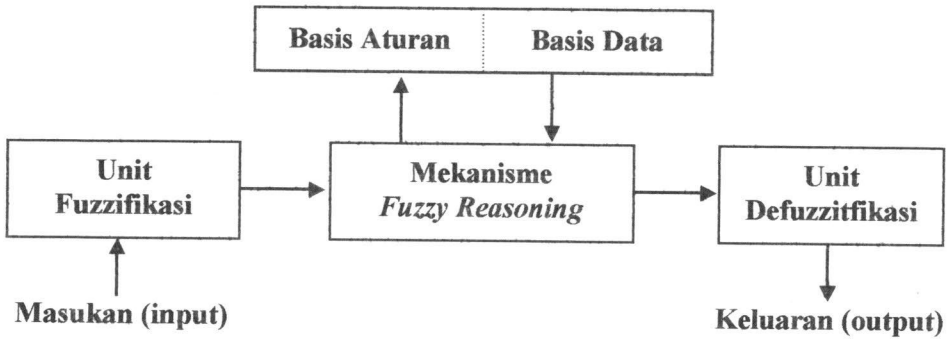
2.5.2. Pengendali logika *fuzzy* integral

Pada pengendali ini logika *fuzzy* dipergunakan untuk memperoleh perubahan keluaran (ΔQ) yang diumpankan ke penguat atau sistem terkendali.Tanggapan yang dikeluarkan sistem terkendali merupakan respon terhadap keluaran pengendali sebelumnya yang telah mengalami perubahan sebesar keluaran pengendali.



Gambar 2.2 Struktur pengendali logika *fuzzy* integral

Pengendali *fuzzy* secara umum merupakan pengendali kalang tertutup. Struktur dasar pengendali logika *fuzzy* terdiri dari unit fuzzifikasi, mekanisme penentuan keputusan dan basis aturan (*inference system*), dan unit defuzzifikasi, seperti diperlihatkan pada gambar 2.3.

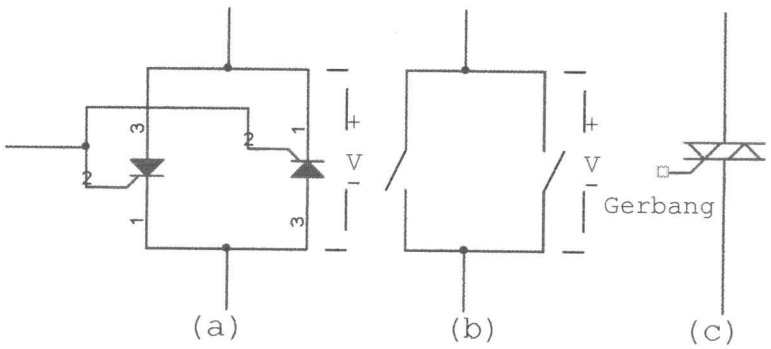


Gambar 2.3 Struktur dasar pengendali

2.6. Triac

TRIAC merupakan saklar elektronik yang sangat ideal untuk mengatur daya arus bolak-balik. Kombinasi TRIAC dan mikrokontroler menghasilkan sistem pengaturan daya yang sangat *fleksible* dan akurat.

Triac merupakan dua SCR yang terpasang paralel yang ekivalen dengan dua penahan yang dapat mengendalikan arus pada kedua arah, tegangan penyalaan tinggi, sehingga cara yang normal untuk menyalakan triac dengan menerapkan pemicu berupa tegangan maju. seperti gambar 2.4



Gambar 2.4 Triac (a) susunan ekivalen dengan sepasang SCR yang terpasang pararel (b) rangkaian ekivalen (c) lambang skematik

Apabila tegangan memiliki polaritas seperti pada gambar 2.4, suhu pemicu positif akan menutup saklar yang disebelah kiri. Saat tegangan memiliki polaritas terbalik, picu negatif akan menutup saklar yang disebelah kanan.

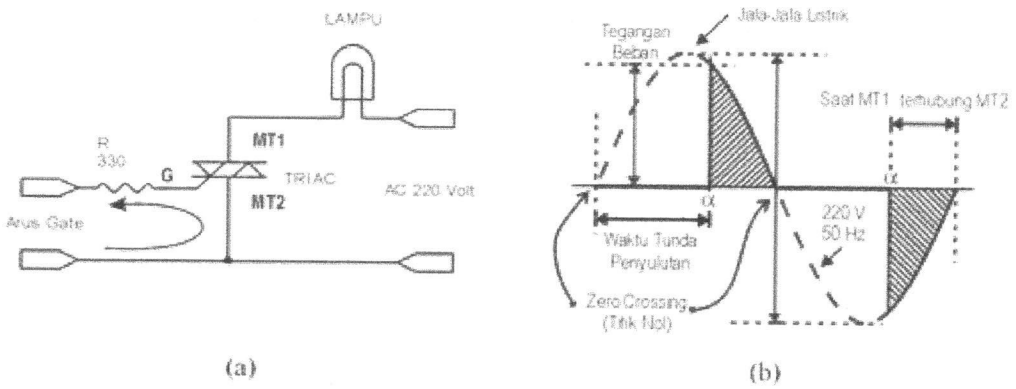
2.6.1. Pengatur Daya

Rangkaian dasar pemakaian TRIAC terlihat dalam gambar 2.5. Kaki MT1 dan MT2 merupakan saklar yang mengatur aliran arus beban yang berasal dari sumber tegangan bolak-balik (AC).Dalam keadaan normal kaki MT1 dan MT2 tidak terhubung, sehingga tidak ada arus beban yang mengalir, MT1 akan terhubung ke MT2 dan mengalirkan arus beban.

Arus gate hanya diperlukan untuk menghubungkan MT1 dan MT2, setelah itu MT1 akan tetap terhubung ke MT2 meskipun sudah tidak ada arus gate lagi. Pemberian arus gate sesaat untuk menghubungkan MT1 dan MT2 dikatakan sebagi menyulut atau (men-triger) TRIAC

MT1 akan tetap terhubung terus ke MT2 selama arus beban yang mengalir lebih besar dari arus minimum (*holding current*) sesuai dengan karaktristik masing-masing TRIAC.

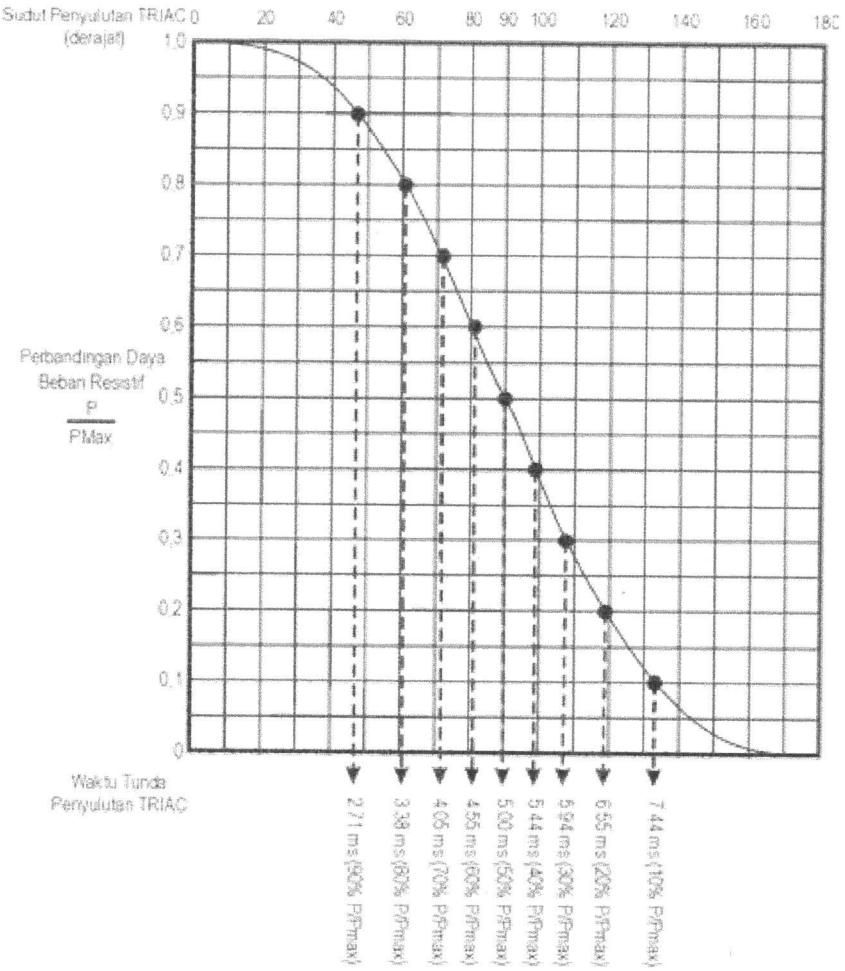
Sumber daya yang dipakai berasal dari tegangan bolak-balik pada daerah titik nol (*zero crossing*) dari tegangan bolak-balik, arus beban yang mengalir akan mengecil sampai kurang dari arus minimum yang diperlukan, akibatnya hubungan antara MT1 dan MT2 akan terputus dengan sendirinya,lihat gambar 2.5



Gambar 2.5 Dasar pengaturan dengan TRIAC

Daya yang disalurkan ke beban tergantung pada lamanya MT1 terhubung ke MT2 setiap setengah periode tegangan sinus dari jala-jala listrik, yakni bagian yang diarsir dalam gambar 2.6, pada saat-saat itulah beban menerima daya. Dengan demikian, daya yang disalurkan kebeban bias diatur dengan mengatur waktu tunda saat penyulutan TRIAC, terhitung pada saat tegangan sinus jala-jala

listrik mencapai titik nol. Teknik pengaturan daya semacam ini dikatakan sebagai teknik PWM.



Gambr 2.6 Sinus jala-jala listrik ¼ periode

2.7. Sistem Mikrokontroler AT89C52

2.7.1. Pendahuluan

Mikrokontroler merupakan sebuah sistem mikroprosesor yang dirancang untuk tujuan-tujuan khusus dan berupa rangkaian terintegrasi dalam sebuah *chip*. Mikrokontroler tersusun atas beberapa komponen:

- *Central Processing Unit* (CPU),
- *Register* sebagai pemroses utama,
- *Read only Memory* (ROM),
- *Random Acsess Memory* (RAM),
- *Timer*, dan *Input/Output* (I/O) sebagai pelengkap.

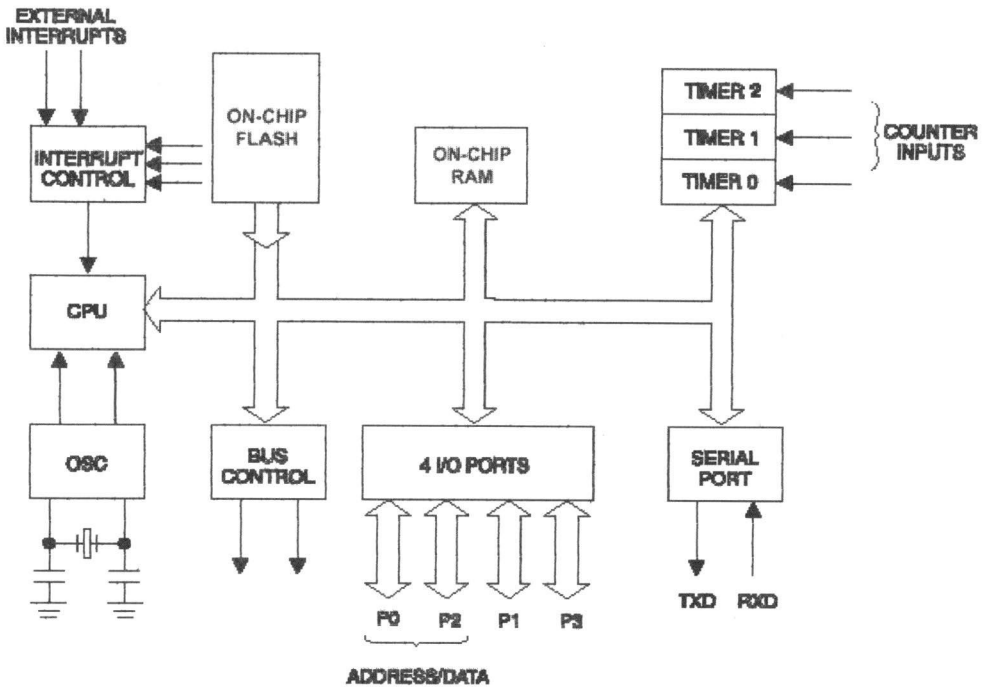
Mikrokontroler dirancang untuk aplikasi-aplikasi di bidang pengendalian sistem terutama dalam pengendalian sekuensial.

Mikrokontroler AT89C52 merupakan mikrokontroler yang diproduksi oleh *ATMEL* dan kompatibel dengan mikrokontroler standar industri MCS-5`1^{TM}. mikrokontroler AT89C52 merupakan mikrokontroler *CMOS* dengan 8-bit yang berdaya rendah dan dilengkapi dengan *Programmable and Eresable Read Only Memory* (PEROM) 8 K byte. Bagian lain dalam mikrokontroler AT89C52 adalah RAM 256 bytes, port I/O, *timer/counter* 16 bit, pengendali interupsi, pengendali bus, port serial *full duplex* dan *on-chip oscillator*.

2.7.2. Konfigurasi Pin

Mikrokontroler AT89C52 mempunyai 3 buah bentuk *chip* yaitu PDIP,PQFP/TQFP dan 68PLCC. Masing-masing pin dalam *chip* mikrokontroler AT89C52 memiliki spesifikasi:

a. Port 0 merupakan port I/O 8 bit dua arah, berfungsi sebagai port masukan atau keluaran. Port ini dikonfigurasi menjadi bus alamat orde rendah yang dimultiplek dengan bus data.



Gambar 2.7 Diagram AT89C52

b. Port 1, merupakan port I/O 8 bit dua arah, berfungsi sebagai port masukan atau keluaran. Sebagai tambahan pin P1.0/T2 dapat dikonfigurasi sebagai input *external counter timer/counter* 2 dan pin P1.1/T2EX dapat dikonfigurasi sebagai input trigger *timer/counter* 2.

c. Port 2, merupakan port yang berfungsi sebagai port masukan atau keluaran. Port ini berfungsi mengeluarkan bit-bit alamat atas (*high-order address bits*) selama pegambilan dari program memori eksternal dan selama pengaksesan ke data memori eksternal yang menggunakan alamat

16 bit. Selama proses pemrograman dan verifikasi PEROM port ini berfungsi menerima bit-bit alamat atas dan sinyal-sinyal kontrol.

d. Port 3, merupakan port masukan atau keluaran yang mempunyai fungsi khusus untuk tiap-tiap pin nya. Fungsi tiap-tiap pin pada port 3 ini pada tabel 2.1.

Tabel 2.1 Bit-bit dalam port 3

Bit	Fungsi
P3.0	RXD (Port masukan serial)
P3.1	TXD (Port keluaran serial)
P3.2	INT0 (Interupsi eksternal 0)
P3.3	INT1 (Interupsi eksternal 1)
P3.4	T0 (Masukan eksternal *timer* 0)
P3.5	T1 (Masukan eksternal *timer* 1)
P3.6	WR (*Strobe* penulisan data memori eksternal)
P3.7	RD (*Strobe* pembacaan data memori eksternal)

e. RST, merupakan masukan reset yang bekerja aktif tinggi.

f. ALE/PROG. ALE berfungsi untuk menahan bit-bit alamat bawah selama proses pengaksesan memori eksternal sedang PROG berfungsi sebagai masukan pulsa program selama pemrograman flash.

g. PSEN. Pin aktif rendah yang berfungsi sebagai sinyal keluaran yang mengaktifkan program memori eksternal.

h. $\overline{EA}/V_{PP}$, berfungsi untuk memungkinkan pengaksesan memori eksternal. Pin EA harus terhubung ke GND untuk memungkinkan pengambilan data dari lokasi program memori eksternal yang dimulai dari alamat 0000h sampai FFFFh. Dan pin EA harus terhubung ke V_{CC} eksekusi program internal. Pin ini juga berfungsi menerima tegangan V_{PP} selama pemrograman PEROM menggunakan mode pemrograman 12 volt Vpp.

i. XTAL1, berfungsi sebagai masukan ke penguat osilator pembalik dan ke rangkaian *clock* internal.

j. XTAL2, berfungsi sebagai keluaran penguat *osilator* pembalik.

2.7.3. Organisasi Memori

Mikrokontroler AT89C52 mempunyai ruang pengalamatan data memori dan program memori yang terpisah.

2.7.3.1. Memori Program

Memori program adalah memori yang digunakan untuk menyimpan program aktual mikrokontroler. Panjang memori program maksimal mencapai 64kbyte. Peta program memori internal pada AT89C52 berawal dari 0000h sampai 1FFFh, yang berarti kapasitas yang dimiliki sebesar 8 kbyte. Untuk alamat 2000h sampai FFFFh dialokasikan sebagai memori eksternal dengan kapasitas maksimal 56 kbyte.

2.7.3.2. Memori Data

Mikrokontroler AT89C52 memiliki memori data internal sebanyak 256 byte, dengan alamat dari 00h sampai 0FFh. 128 byte pertama memori dapat diakses melalui pengalamatan langsung maupun tak langsung, sedangkan 128 byte berikutnya hanya bisa diakses melalui pengalamatan tidak langsung. Hal tersebut disebabkan karena adanya register khusus yang menggunakan alamat yang sama dengan memori data bagian atas (80h sampai 0FFh). Peta memori alamat 128 byte bawah, terdiri dari 32 byte register yang dikelompokkan dalam 4 *register bank* (R0 - R7) pada alamat 00h sampai 1Fh, 16 byte memori bit (bit 00 -

bit 7F) pada alamat 20h sampai 2Fh, 208 byte memori secara umum yang digunakan sebagai RAM data pengguna (30h sampai 0FFh).

2.7.3.3. Register Fungsi Khusus (SFR)

SFR (*special function register*) merupakan memori yang mengatur fungsi tertentu dari prosesor AT89C52. SFR hanya dapat diakses melalui pengalamatan langsung dengan alamat 80h sampai 0FFh.

00	R0	R1	R2	R3	R4	R5	R6	R7	Reg. Bank 0
08	R0	R1	R2	R3	R4	R5	R6	R7	Reg. Bank 1
10	R0	R1	R2	R3	R4	R5	R6	R7	Reg. Bank 2
18	R0	R1	R2	R3	R4	R5	R6	R7	Reg. Bank 3
20	00	08	10	18	20	28	30	38	Bits 00-3F
28	40	48	50	58	60	68	70	78	Bits 40-7F
30									General
...									RAM
78									30 – 7F
80	P0	SP	DPL	DPH				PCON	Special
88	TCON	TMOD	TL0	TL1	TH0	TH1			Function
90	P1								Registers
98	SCON	SBUF							80 - FF
A0	P2								
A8	IE								
B0	P3								
B8	IP								
C0									
C8	T2CON		RCAP2L	RCAP2H	TL2	TH2			
D0	PSW								
D8									
E0	ACC								
E8									
F0	B								
F8									

Tabel 2.2 Peta RAM internal AT89C52

2.7.4. Register

Register-register dalam mikrokontroler *AT89C52* adalah antara lain *accumulator* (A), register B, register R, *Data Pointer* (DPTR), *Program Counter* (PC), dan *Stack Pointer* (SP). Masing-masing register ini dapat diterangkan sebagai berikut.

2.7.4.1. Register A dan B

Register A atau *accumulator* merupakan register umum untuk mengakumulasikan hasil dari instruksi-instruksi. *Accumulator* mempunyai lebar data 8 bit (1 byte) dan merupakan register yang paling sering dipakai. Hampir lebih dari separuh dari instruksi keluarga AT89C52 menggunakan register *accumulator*. Register B merupakan register penampung 8 bit (1 byte) tetapi terbatas hanya untuk perintah perkalian (MUL) dan perintah pembagian (DIV).

2.7.4.2. Register R

Register R adalah terdiri dari delapan buah register 8 bit yang dinamakan R0, R1, R2, R3, R4, R5, R6, dan R7. Register-register ini merupakan register serba guna dan dapat digunakan sebagai register penampung sementara. Khusus untuk register R0 dan R1, kedua register ini dapat digunakan untuk menyimpan alamat pointer.

2.7.4.3. DPTR, PC, dan SP

DPTR (*Data Pointer*) merupakan register 16 bit untuk menyimpan alamat pointer. DPTR digunakan untuk perintah-perintah pengaksesan memori eksternal yang mempunyai lebar alamat 16 bit. Sedang PC (*Program Counter*) merupakan register 2 byte yang memberitahu AT89C52 di mana instruksi selanjutnya akan

dilaksanakan. Saat AT89C52 inisialisasi, PC selalu berisi 0000h dan bertambah satu setelah sebuah perintah dieksekusi. Register PC tidak dapat diubah nilainya secara langsung dengan perintah MOV. Nilai PC akan berubah oleh perintah-perintah yang berhubungan dengan lompatan (JMP). Sementara itu *Stack Pointer* (SP) merupakan register 8 bit yang digunakan untuk menunjukkan di mana harga berikutnya yang akan diambil dari *stack*. Jika suatu harga dimasukan dalam *stack*, AT89C52 petama-tama akan menambah harga SP dan kemudian menyimpan harga tersebut pada alamat memori yang bersesuaian. Kemudian jika suatu harga diambil dari *stack*, maka AT89C52 akan mengambil harga dari *stack* dan kemudian mengurangi harga SP.

2.7.5. Mode Pengalamatan

AT89C52 mempunyai 5 buah mode pengalamatan, yaitu *immediate addressing*, *direct addressing*, *indirect addressing*, *External Direct*, dan *External Indirect*.

Immediate addressing atau pengalamatan segera adalah pengalamatan dengan memberikan nilai/harga yang akan disimpan ke dalama memori secara langsung. Contohnya perintah MOV A,#20h akan mengisi akumulator dengan harga 20h.

Direct addressing atau pengalamatan langsung merupakan pengalamatan yang mengambil nilai yang disimpan dalam memori secara langsung dari lokasi memori yang lain. Contoh perintah berupa MOV A,30h yang akan membaca data yang berada pada RAM internal dengan alamat 30h dan menyimpannya dalam *accumulator*.

Indirect addressing atau pengalamatan tidak langsung, digunakan untuk mengakses 128 byte RAM internal bagian atas. Pengalamatan ini memanfaatkan register-register penunjuk seperti R0 – R7 dan DPTR, serta menggunakan simbol @ untuk menunjukkan isi dari alamat lokasi memori yang ditunjukkan oleh suatu register penunjuk. Contoh perintah MOV A,@R0 akan mengisi *accumulator* dengan nilai pada alamat yang ditunjukkan oleh R0.

External Direct merupakan pengalamatan yang digunakan untuk mengakses memori eksternal yang berfungsi seperti pengalamatan langsung. Ada dua perintah yang dipergunakan mode pengalamatan ini:

MOVX A,@DPTR

MOVX @DPTR,A

Kedua perintah ini mempergunakan DPTR, yang mempunyai kapasitas 16 bit. Pada perintah pertama DPTR akan memuat alamat memori eksternal yang akan dibaca/ditulis dan DPTR menahannya. Isi dari alamat tersebut kemudian dibawa ke *accumulator*. Perintah kedua sebaliknya, membaca/menulis harga *accumulator* ke alamat eksternal yang ditunjukkan DPTR.

External Indirect digunakan untuk mengakses RAM eksternal dan menggunakan bentuk dari pengalamatan tidak langsung. Contoh perintah MOVX @R0,A nilai R0 pertama akan dibaca dan nilai akumulator kemudian ditulis pada alamat RAM eksternal tersebut.

2.7.6. *Timer*

Mikrokontroler AT89C52 mempunyai tiga buah *timer* 16 bit, masing-masing timer 0, timer 1 dan timer 2. Ketiga *timer* ini dapat difungsikan sebagai

timer atau penghitung waktu, pencacah dan sebagai pembangkit *baud rate* untuk komunikasi serial. Setiap *timer* dapat dikonfigurasi dan dibaca sendiri-sendiri.

Untuk mengkonfigurasi ketiga *timer* ini digunakan register-register kendali dalam SFR yaitu TMOD dan TCON untuk mengkonfigurasi timer 0 dan timer 1. Konfigurasi bit-bit pada kedua register ini ditunjukkan dalam tabel 2.5 dan tabel 2.6. Sedang tabel 2.7 menunjukkan kombinasi bit M0 dan M1 yang akan menentukan mode *timer.*

Register kendali TCON merupakan register yang dapat diakses per bit yang digunakan untuk mengaktifkan dan mematikan *timer*, menentukan jenis interupsi eksternal *timer* dan untuk menandakan adanya *overflow flag.* Sedang register TMOD bukan merupakan register yang dapat dialamati per bit. Kedua register ini digunakan untuk menentukan mode operasi dari timer 0 dan timer 1.

Tabel 2.3 Definisi bit-bit pada register TCON.

Nama	Bit	Fungsi
TF1	TCON.7	*Timer* 1 *overflow flag*
TR1	TCON.6	Bit *on/off* timer 1 (1 *timer* on/0 *timer* off)
TF0	TCON.5	Timer 0 *overflow flag*
TR0	TCON.4	Bit *on/off timer* 0 (1 *timer* on/0 *timer* off)
IE1	TCON.3	Interupsi 1 edge flag eksternal
IT1	TCON.2	Tipe interupsi 1 bit kendali
IE0	TCON.1	Interupsi 0 edge flag eksternal
IT0	TCON.0	Tipe interupsi 0 bit kendali

Tabel 2.4 Register kontrol TMOD

Timer 1				Timer 0			
GATE	C/T	M1	M0	C/T	C/T	M1	M0

GATE : Jika bit ini diset, *timer* hanya akan bekerja jika INT1 (P3.3) berlogika 1 Jika bit ini dinolkan, *timer* hanya bekerja jika TR1=1.

C/T : Pemilih fungsi *timer*/konter, jika diset 1 maka timer 1 berfungsi sebagai konter dan jika diset 0 berfungsi sebagai *timer.*

M1 dan M0 : Pemilih mode operasi.

Tabel 2.5 Kombinasi mode operasi timer 0 dan 1.

M1	M0	Mode	Operasi
0	0	0	*Timer* 13 bit
0	1	1	*Timer*/konter 16 bit
1	0	2	*Timer* auto reload 8 bit (pengisian otomatis)
1	1	3	*Timer terbagi* (8 bit)

Register T2MOD dan T2CON digunakan untuk mengkonfigurasi timer 2. T2CON merupakan register yang dapat diakses per bit yang digunakan untuk mengaktifkan dan mematikan *timer* 2, menentukan jenis interupsi eksternal *timer* dan untuk menandakan adanya *overflow flag.*

Nilai aktual pencacahan dari ketiga *timer* disimpan dalam SFR yaitu pada register TH0/TL0 untuk TIMER 0, TH1/TL1 untuk TIMER 1 dan TH2/TL2 untuk TIMER 2. Tiap-tiap register tersebut merupakan register 8 bit. TH menunjukkan byte atas dan TL menunjukkan byte bawah dari harga pencacahan masing-masing *timer.*

Tabel 2.6 Definisi bit-bit pada register T2CON.

Nama	Bit	Fungsi
TF2	T2CON.7	Timer 2 *overflow flag*
EXF2	T2CON.6	Timer 2 *external flag*
RCLK	T2CON.5	*Receive clock flag*
TLCK	T2CON.4	*Transmit clock flag*
EXEN2	T2CON.3	Timer *2 external enable flag*
TR2	T2CON.2	Bit *on/off timer* 2 (1 *timer* on/0 *timer* off)
C/T2	T2CON.1	Pilihan *timer/counter* (0=internal ; 1=eksternal)
CP/RL2	T2CON.0	*Capture/reload flag*

Tabel 2.7 Register kontrol T2MOD

-	-	-	-	-	-	T2OE	DCEN
Bit 7	**6**	**5**	**4**	**3**	**2**	**1**	**0**

-- : Tidak digunakan

T2OE : *Timer 2 Output Enable bit.*

DCEN : Jika diset, timer 2 dapat dikonfigurasi sebagai *up/down counter.*

2.7.7. Interupsi

Interupsi merupakan suatu kejadian yang akan menghentikan sementara jalannya program yang sedang berjalan untuk menjalankan suatu subrutin *(interrupt handler)* tertentu. Setelah subrutin interupsi selesai dikerjakan maka program yang dihentikan tadi akan dilanjutkan kembali secara normal. *Interrupt handler* akan dijalankan hanya jika ada kejadian (*event*) tertentu seperti *timer overflow*, *external interrupt* atau aktifnya port serial.

Setiap saluran interupsi mempunyai alamat vektor pelayanan interupsi yang tetap seperti diperlihatkan dalam tabel 2.8. Setelah layanan interupsi pada alamat vektor tertentu selesai dikerjakan maka untuk kembali ke program semula digunakan perintah RETI (*return from interrupt routine*).

Tabel 2.8 Alamat vektor pelayanan interupsi

Sumber Interupsi	Alamat Vektor
IE0	0003H
TF0	000BH
IE1	0013H
TF1	001BH
R1 & T1	0023H
TF2 & EXF2	002BH

Untuk mengkonfigurasi setiap saluran interupsi digunakan register *interrupt enable* (IE) pada SFR. Selain itu interupsi bisa dikonfigurasi level prioritasnya dengan mengatur bit-bit pada register *interrupt priority* (IP). Isi bit-bit pada register IE dan IP seperti pada tabel 2.9 dan tabel 2.10.

Tabel 2.9 Bit-bit register *interrupt enable* (IE).

Nama	Bit	Fungsi
EA	IE.7	Jika EA = 0 semua saluran interupsi tertutup, EA = 1 saluran interupsi bisa digunakan
-	IE.6	Tidak dipakai
ET2	IE.5	Pengaktifan interupsi jika *timer* 2 *overflow*
ES	IE.4	Interupsi melalui port serial
ET1	IE.3	Pengaktifan interupsi jika *timer* 1 *overflow*
EX1	IE.2	Pengaktifan interupsi melalui $\overline{\text{INT1}}$
ET0	IE.1	Pengaktifan interupsi jika *timer* 0 *overflow*
EX0	IE.0	Pengaktifan interupsi melalui $\overline{\text{INT0}}$

Tabel 2.10 : Bit-bit register *interrupt priotity* (IP).

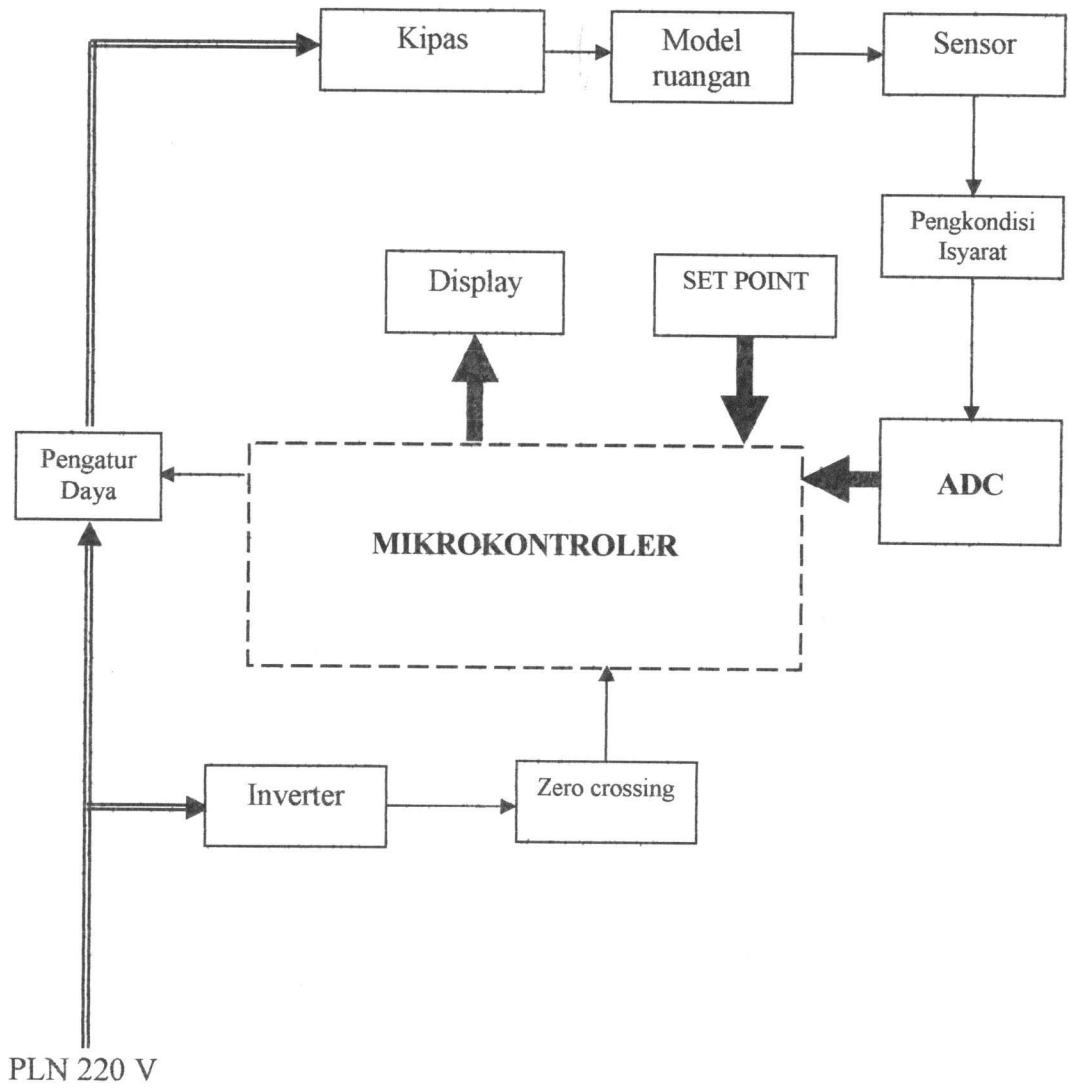
Nama	Bit	Fungsi
-	IP.7	Tidak dipakai
-	IP.6	Tidak dipakai
PT2	IP.5	Prioritas interupsi melalui *timer* 2
PS	IP.4	Prioritas interupsi melalui port serial
PT1	IP.3	Prioritas interupsi melalui *timer* 1
PX1	IP.2	Prioritas interupsi melalui $\overline{\text{INT1}}$
PT0	IP.1	Prioritas interupsi melalui *timer* 0
PX1	IP.0	Prioritas interupsi melalui $\overline{\text{INT0}}$

BAB III

PERANCANGAN SISTEM

3.1. Blok Diagram

Diagram kotak Sistem kendali Pengatur kecepatan kipas angin sesuai dengan suhu ruangan ditunjukan pada gambar 3.1



Gambar 3.1 Diagram blok pengaturan kipas angin berdasarkan suhu ruangan dengan menggunakan logika *Fuzzy*

Mikrokontroler digunakan sebagai unit utama untuk pengendalian. Mikrokontroler memperoleh masukan dari sensor suhu yang dikonversi menjadi sinyal digital oleh unit ADC. Data masukan akan diolah oleh mikrokontroler yang telah di program dengan algoritma kendali *fuzzy* untuk mengumpankan sinyal keluaran pada unit pengatur fase.

Unit pengatur fase mengubah sudut picu dengan acuan dari unit *zero crossing* detektor. Sehingga posisi acuan awal untuk menentukan letak posisi sudut picu dapat ditentukan berdasarkan acuan dari unit *zero crossing* detektor.

Perubahan sudut picu akan digunakan untuk mengendalikan aliran arus yang diumpankan pada kipas. Unit yang digunakan untuk mengendalikan aliran arus pada kipas adalah triac.

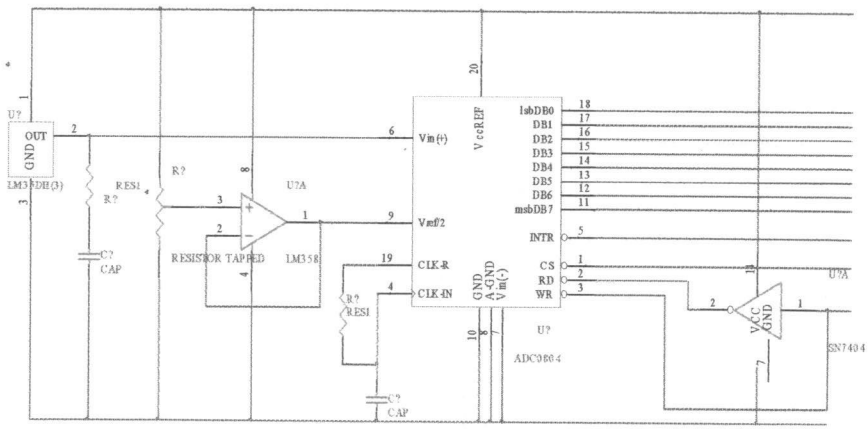
3.2. Perangkat Keras

3.2.1. Sensor LM 35 dan ADC

Sensor suhu LM 35 merupakan sensor temperatur dalam paket plastik type To-46 hermetic (kedap udara). LM 35 dapat membaca pengeseran suhu dan merubah suhu menjadi tegangan, yaitu ± 10.0 mV / ^{0}C, dan tegangan ini langsung dihubungkan dengan masukan ADC 0804 untuk dikonversi menjadi digital.

Inverter ADC 0804 menggunakan sistem pendekatan berturut–turut (*Successive Approximation*). Metode pendekatan berturut–turut merupakan proses pendekatan tegangan analog dengan mencoba satu bit setiap saat dimulai dari MSB. ADC 0804 dalam mengkonversi membutuhkan tegangan referensi pada pin 9. Agar tegangannya stabil dibutuhkan sebuah penyangga, maka dipakailah IC LM 358 dengan penguatan satu kali. Kemudian tegangan referensi tersebut dapat

di setel lewat VR yang sekaligus dapat mengkalibrasi sensor. Gambar 3.2 untainya dapat dilihat sebagai berikut :

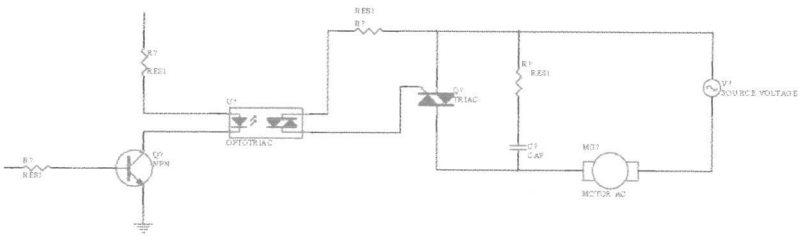


Gambar 3.2 Rangkaian sensor LM35 dengan ADC

3.2.2. Untai Pengatur Daya

Bagian pengatur daya dibentuk dengan R1, R2, transistor, MOC 3021, TRIAC Q1 dan kipas sebagai beban. Bagian ini langsung berhubungan dengan sumber tegangan jala-jala listrik 220 Volt, agar tegangan jala-jala terpisah dari bagian lainya, dipakai Opto Isolator MOC 3021 untuk menghubungkan mikrokontroler AT89C52 dengan TRIAC.

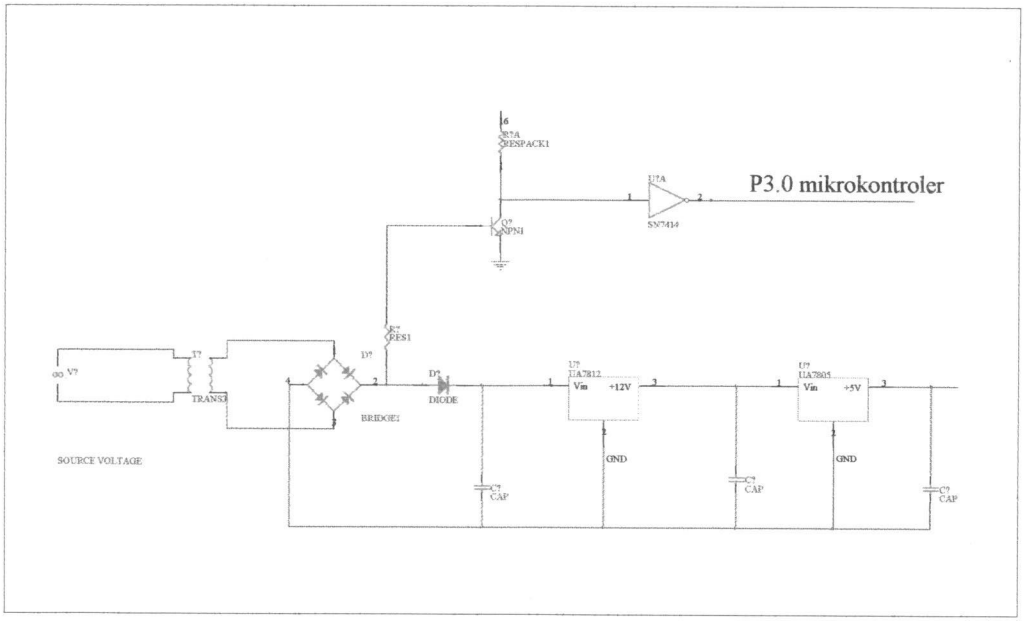
Bagian input dari MOC 3021 merupakan RED yang dinyala-padam-kan oleh AT89C52 lewat resistor R1,instuksi CLR P1.7 akan mengakibatkan port 1.7 menyalurkan arus dari Vcc lewat RED dari ground sehingga RED menyala. Mengakibatkan triac dibagian Output MOC 3021 menjadi 'on' dan mengalirlah arus gate TRIAC Q1 lewat resistor R2 selanjutnya TRIAC akan 'on' dan kipas akan menyala.



Gambar 3.3 Rangkaian MOC 3021

Agar bisa menentukan waktu tunda dengan tepat untuk mendapatkan hasil pengaturan daya yang akurat, mikrokontroler harus mengetahui saat titik nol (*zero crossing)* dari tegangan jala-jala listrik. Bagian ketiga merupakan bagian pemantau titik nol tegangan jala-jala listrik yang dibentuk dengan transistor NPN Q2, resistor R3 dan diode D2 dan IC pembalik 74LS04.

Rangkaian ini dihubungkan pada rangkian catu daya dengan cara memasukan Diode D2 antara jembatan diode D1 dan Kapasitor C5, maksudnya agar pada ujung kanan resistor R3 masih berupa tegangan searah yang belum diratakan. Tegangan ini setelah lewat transistor NPN Q2 berubah bentuk menjadi gelombang kotak yang diumpankan ke kaki IC74LS04 untuk dilakuakn pembalikan. Keluaran IC74LS04 diumpankan ke kaki P3.0/INT0 (kaki6) AT89C52. Setiap kali tegangan jala-jala listrik mulai meningalkan titik nol, gelombang kotak akan berubah dari '1' menjadi '0' yang merupakan sinyal permintaan intrupsi bagi AT89C52.



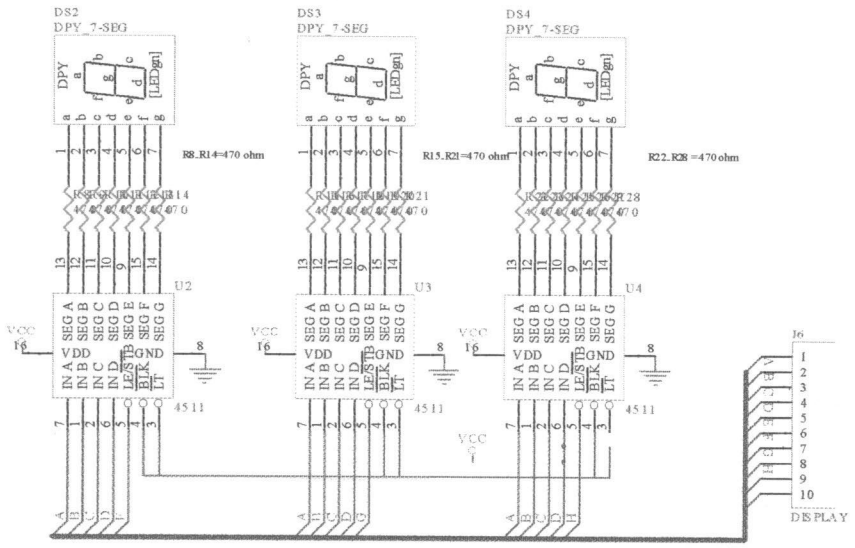
Gambar 3.4 Rangkaian *Zero Crossing*

3.2.3. Untai Penampil

Untai penampil ini digunakan untuk menampilkan suhu yang dikeluarkan oleh sensor, dan untuk mengetahui berapa suhu yang ada disekitar ruangan.

Tampilan menggunakan 3 buah 7 segmen yang dilengkapi dengan *driver* berupa IC CMOS 4511 dengan masukan BCD. Masukan IC driver ini berasal dari port A PPI 8255 pada papan sistem minimal mikrokontroler. Bit 0 sampai bit 3 menjadi masukan BCD ke *driver*, sedang bit 4 sampai bit 7 menjadi masukan *chip select* pada ketiga IC *driver* tersebut. IC *driver* akan aktif apabila mendapat masukan rendah atau nol pada pin *chip select*-nya.

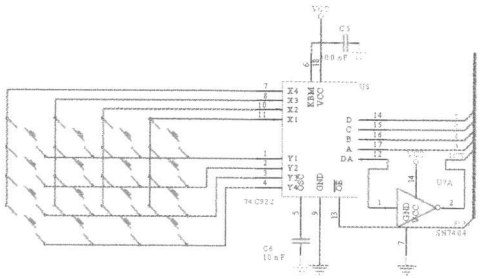
Sehingga untuk menampilkan besar suhu ruangan, mikrokontroler cukup dengan mengirimkan data suhu tersebut dan data posisi bilangan ke IC 4511 yang akan ditampilkan oleh 7 segment.



Gambar 3.5 Rangkaian Penampil

3.2.4. Keypad

Keypad untuk memasukan set point suhu, dengan menggunakan encoder IC 74C992. Prinsip kerja dari encoder ini setiap tombol dihubungkan dalam bentuk matrik, salah satu tombol yang ditekan akan di kodekan menjadi kode BCD pada port keluraran IC bersamaan dengan itu pula pin INT akan berlogika tinggi yang mengimpormasikan ada tombol yang ditekan. Karena pin INT pada mikrokontroler aktif rendah.



Gambar 3.6 Rangkaian keypad

3.2.5. Kipas Motor AC

Kipas motor AC ini untuk menurunkan temperatur ruangan, digunakan kipas yang diputar oleh motor arus bolak-balik, sebagaimana lazimnya mesin yang dapat dibagi dalam generator dan motor, maka demikian juga pada mesin induksi. Akan tetapi generator induksi jarang dipergunakan karena penampilan atau karkteristiknya yang kurang memuaskan. Sedang motor induksi justtru banyak dipergunakan sebagai tenaga penggerak dalam industri.

Prinsip kerja motor induksi adalah dihasilkannya medan magnet putar . Cara kerjanya sebagai berikut :

- Belitan stator yang dihubungkan dengan sumber satu fase menghasilkan medan magnet dalam celah udara dan berotasi dengan kecepatan sinkron.
- Medan magnet putar memotong badan penghantar pada rotor yang menimbulkan tegngan induksi.
- Tegngan induksi menyebabkan adanya arus pada kumparan rotor.
- Arus didalam medan magnet menimbulkan gaya pada rotor.
- Perbedaan relatif antara kecepatan motor putar stator dengan kecepatan putar rotor sedikit lebih lambat dari putaran medan stator.
- Bila perbandingan slip tegangan tidak akan terinduksi dan arus tidak akan mengalir pada kumparan rotor maka tidak akan dihasilkan torsi.
- Motor induksi disebut motor tak serempak atau asinkron.

3.3. Perancangan Program

Program yang digunakan adalah bahasa assembly untuk keluarga MCS51. Program ditulis dengan editor biasa dan dikompiler menggunakan kompiler ASM51. Setelah perancangan perangkat keras yang berupa sistem minimal AT89C52 serta perancangan Logika *Fuzzy* berhasil dilakukan maka langkah berikutnya, adalah merancang perangkat lunak sebagai langkah penerapan rancangan yang telah dibuat guna mencapai tujuan pengendalian .

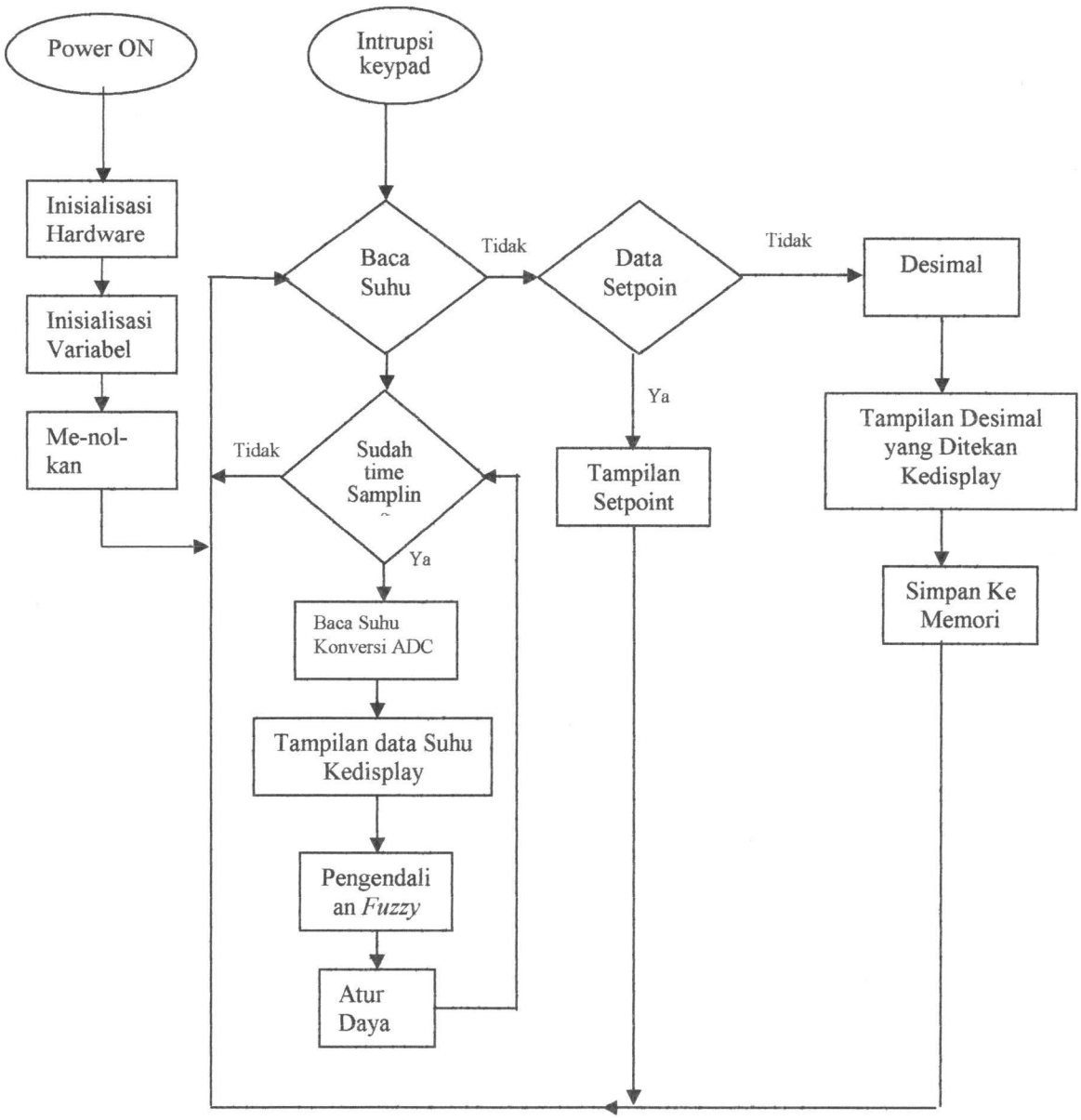
Perangkat lunak ini dibuat dengan bahasa assembler mikrokontroler ASM-51, kemudian dikompaile untuk menghasilkan file listing dan heksadesimal dengan kompiler khusus ASM-51. File heksadisimal kemudian didownload ke-flash ROM sistem minimal AT89C52 dengan menggunakan *down loader.*

3.3.1. Program Utama

Program utama menginisialisasi nilai variabel atau menentukan operasi suatau komponen. Pada tugas akhir ini inisialisasi komponen digunakan untuk menginisialisai PPI 8255 yang port-port nya dibuat sebagai keluaran. Sedangkan inisialisasi yang lain hanya memberi nilai awal yang lain untuk suatu variabel.

Register-register yang menampung variabel untuk tampilan diberi nilai nol. Kemudian program utama mendekteksi terjadinya penekanan tombol melalui flag yang dikirimkan oleh prosedur layanan intrupsi keypad. Prosedur layanan intrupsi keypad memberi tanda untuk program utama agar mengeksekusi rutin-rutin mengubah data set point.

Program utama juga menunggu flag waktu sampling untuk memerintahkan ADC mengkonversi sinyal suhu. Flag waktu sampling diaktifkan oleh prosedur layanan intrupsi eksternal nol. Bila flag waktu sampling set maka terjadi proses konversi dan kemudian program akan memanggil prosedur menampilkan suhu dan juga memanggil prosedur untuk melakukan pengendalian dengan algoritma *fuzzy*.



Gambar 3.7 Diagram alir Program Utama

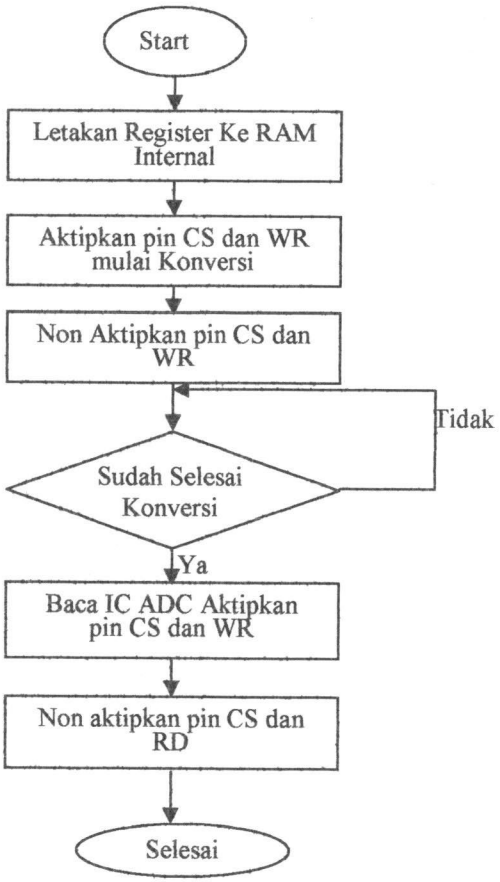
Pada saat alat dijalankan akan tampil menu utama. Menu utama ini akan menunggu masukan dari tombol 0..9 (keypad). RUN, STOP. Adapun penjelasan dari tombol-tombol penting ini dijelaskan pada tabel 3.1.

Tabel 3.1 Keterangan tombol-tombol perancangan perangkat lunak

No	Tombol	Keterangan
1	Keypad(0..9)	Memasukan data setpoint
2	RUN #	Menampilkan data suhu dan proses kendali *fuzzy* berjalan.
3	STOP *	Memasukan set point dan menghentikan kendali *fuzzy*.

3.3.2. Prosedur ADC

Pada prosedur ini digunakan untuk menangani pembacaan suhu ruangan dari sensor Lm35.ADC yang digunakan memiliki resolusi 8 bit. Prosedur membaca ADC dengan enable CS untuk mengaktifkan bus data ADC agar dapat dibaca oleh mikrokontroler kemudian membuat low pin WRT sesaat dan membuat set pin CS kemudian rutin menunggu samapai pin EOC low sehingga data dapat dibaca oleh mokrokontroler. Data hasil bacaan dari ADC disimpan dalam variabel PV. Diagram alir rutin ADC sepert pada gambar 3.8.



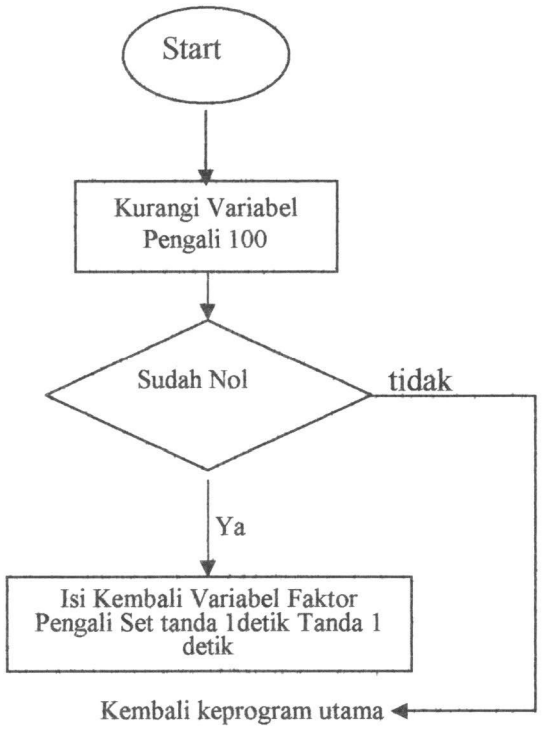
Gambar 3.8 Alir Konversi ADC

3.3.3. Prosedur *Zero Crossing*

Untai *zero crossing* menyidiakan sinyal yang digunakan sebagai pulsa untuk membangkitkan intrupsi eksternal. Rutin layanan intrupsi eksternal digunakan untuk mematikan Triac dan untuk mengendalikan posisi hidup atau matinya Triac.

Dalam rutin layanan intrupsi terdapat nilai perbandingan antara pulsa terhadap pulsa maksimal yang sebenarnya digunakan untuk membaca nilai tabel yang akan dipindahkan pada register TH0 dan TL0. Nilai-nilai pada TH0 dan TL0 tersebut adalah nilai-nilai yang akan mengendalikan sudut picu Triac.

Rutin layanan intrupsi juga digunakan untuk menentukan proses konversi berdasarkkan cacahan variabel TIX dan TS. Bila kedua variabel digunakan sebagai pencacah tersebut bernilai nol maka prosedur untuk membaca data dari ADC akan dilakukan. Diagram alir prosedur ini terdapat pada gambar 3.9



Gambar 3.9 Diagram Alir Prosedur Zerocrossing

3.4. Perancangan Pengedali Logika *Fuzzy*

Pada tugas akhir ini perancangan logika *fuzzy* dilakukan dengan bantuan perangkat lunak MATLAB versi 5.2 khususnya *fuzzy* editor. Metode inferensi yang digunakan adalah metode MAX-MIN, sedangkan proses defuzzifikasinya menggunakan metode *center of area* (COA). Penjelasan mengenai kedua metode ini telah dijelaskan pada dasar teori.

Sedangkan penetuan rancangan logika *fuzzy* itu sendiri didasarkan pada pertimbangan berikut:

1. Tujuan pengendalian
2. Keterbatasan kemampuan perangkat keras (*hardware*).

Dengan pertimbangan kedua hal diatas diperoleh dua masukan ke logika *fuzzy* yaitu error (err) dan perubahan Δerror (delta error) serta satu keluaran logika *fuzzy* (kel). Error (err) adalah perbedan antara set point (s_p) suhu dengan suhu saat itu (p_v),error ini mewakili kecepatan yang belum dicapai, dapat dirumuskan:

$$[s\text{-}p] - [p\text{-}v(k)] \qquad (3.2)$$

Delta error adalah error dikurangi error saat itu dikurangi error sebelumnya:

$$D\text{-}err(k) = Err(k)\text{-}Err(k\text{-}1) \qquad (3.3)$$

Suhu saat itu (p_v) merupakan nilai ADC yang dibaca. Selain pertimbangan diatas ada alasan lain digunakan kedua variabel tersebut sebagai masukan yaitu, penggunaan error saja sebagai masukan logika *fuzzy* hanya efektif untuk temperatur yang diinginkan itu besar berada dikawasan keanggotaan PE atau dikawasan keanggotaan NE untuk temperatur yang diinginkan kecil tidak akan tertangani karena batas fungsi keanggotaan error yang digunakan konstan, sehingga tujuan pengendalian tidak tercapai.

Pada persamaan 3.2 error diperoleh dengan mengurangi set point bila terdapat perubahan data bacaan ADC, sehingga error akan negatif atau positif. Bila error menjadi nol maka sudut picu terdapat pada titik dekat dengan titik nol, sehingga arus yang dihantarkan untuk memutar kipas sangat kecil. Sebaliknya bila

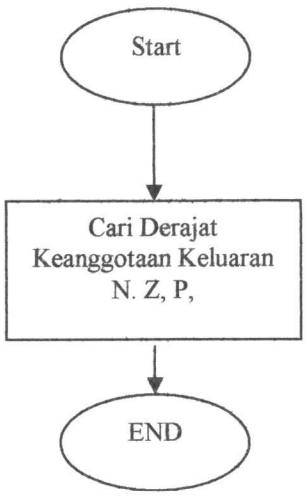
error yang terjadi positif dan maksimal, maka sudut picu akan terdapat jauh dari titik nol. Dari asumsi diatas dengan berdasarkan panjang maksimal bus data mikrokontroler maka semesta wacana untuk error minimal adalah 0^0C sedangkan semesta wacana untuk error maksimal adalah 25^0C atau minus 25^0C. Untuk memudahkan dalam perancangan program jangkauan nilai minus 25^0C dikonversi kedalam desimal menjadi -127 dan error maksimal 25^0C menjadi 127 kemudian semua error dipetakan kedalam daerah positif menjadi 0 sampai 255.

Perubahan error digunakan untuk mempercepat proses pengendalian serta untuk menghilangkan osilasi pada saat terjadi perubahan mendadak terhadap sistem yang dikendalikan. Seperti dalam perancangan himpunan semesta untuk variabel masukan error, himpunan semesta perubahan error adalah 0 untuk perubahan nol dan positif atau negatif untuk perubahan error maksimal.

3.4.1. Prosedur Fuzzifikasi

Pengendali *fuzzy* terdiri dari tiga bagian yaitu fuzzifikasi, inferensi dan defuzzifikasi pada bagian ini akan dijelaskan prosedur fuzzifikasi menggunakan bahasa asmbeler MCS-51. proses fuzzifikasi dimaksudkan memetakan domain masukan teramati (himpunan tegas) ke domain logika *fuzzy* (himpunan *fuzzy*) yang telah ditentukan sebelumnya. Dalam tugas akhir ini proses fuzzifikasi menggunakan metode *look up tabel* untuk menentukan derajat keanggotaan himpunan masukan. Metode *look up tabel* digunakan selain lebih mudah juga kerena keterbatasan mikrokontroller dalam perhitungan aritmatika diatas 8 bit. Derajat keanggotan masukan untuk masing-masing fungsi keanggotaan masukan

diberikan pada lampiran (listing program). Diagram alir yang memperlihatkan aliran program fuzzifikasi diperlihatkan pada gambar 3.10.



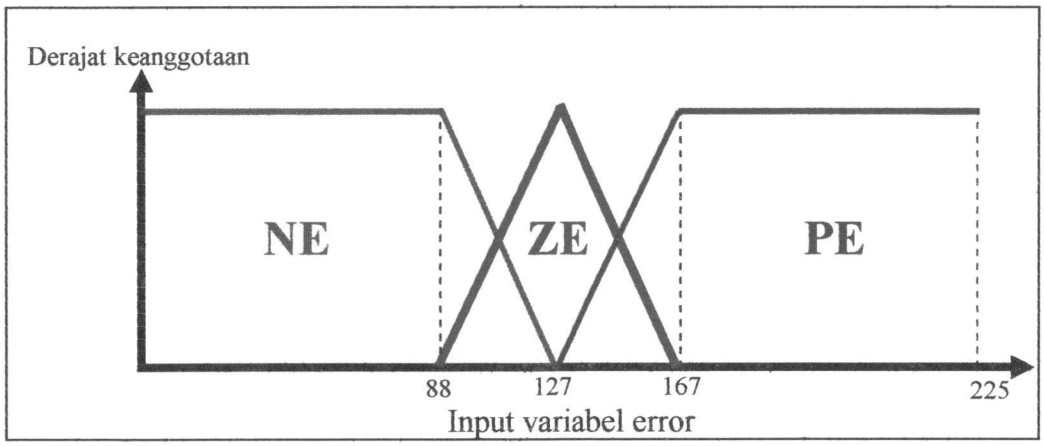
Gambar 3.10 diagram alir proses inferensi

3.4.2. Fuzzifikasi

Fungsi keangotaan diperoleh dengan menggunakan cara *trial and error* (coba-coba) diperoleh fungsi keanggotan error seperti pada gambar 3.11.

Keanggotaan himpunan *fuzzy* pada rancangan ini dinyatakan dalam definisi fungsional, yaitu dengan cara analisis untuk menentukan derajat keanggotaan untuk setiap elemen pada semesta pembicaraan.

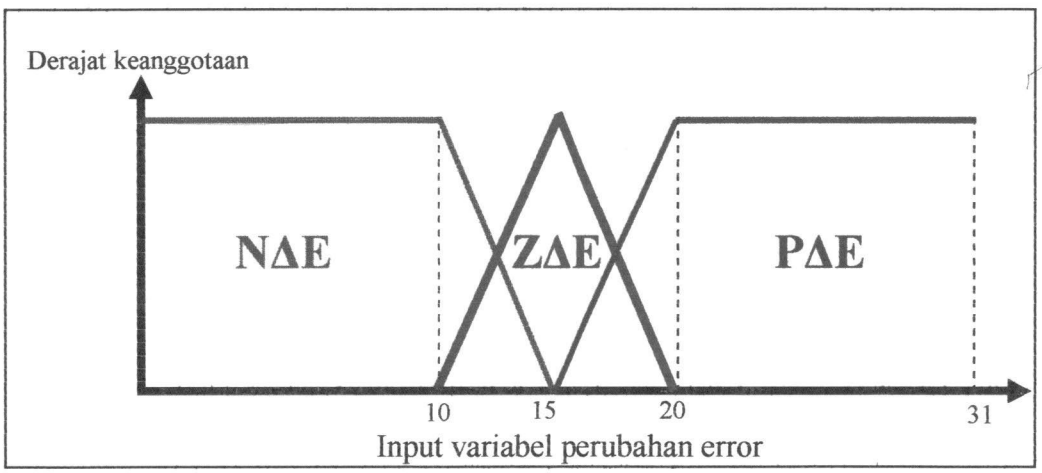
Fungsi keanggotaan *fuzzy* memiliki bentuk yang berbeda-beda tergantung pada pengalaman perancang. Namun untuk kemudahaan dari segi komputasi dipilih bentuk segitiga, untuk mewujudkan kedua bentuk tersebut dapat dilihat pada gambar 3.11.



Gambar 3.11 Fungsi keanggotaan masukan error (err)

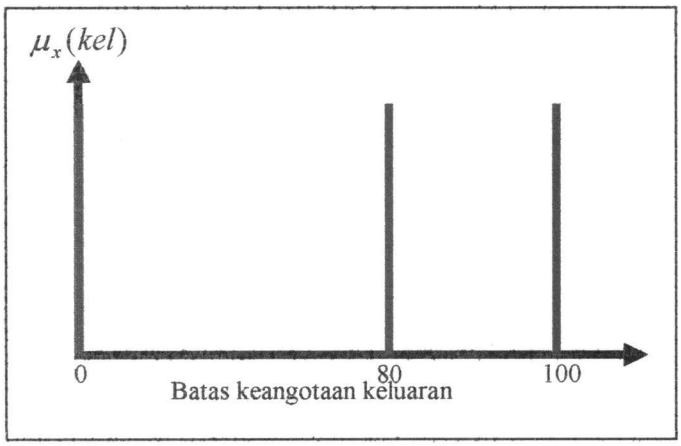
Fungsi keanggotaan input yang digunakan berupa gabungan fungsi keanggotaan segitiga dan trapesium. Error maksimal mencapai desimal 127 sesungguhnya adalah 25^{0}C.

Fungsi keanggotaan perubahan error ditentukan dengan cara mencoba memetakan variabel input error dan mengamati pengaruhnya terhadap kecepatan putaran kipas. Fungsi keanggotaan error seperti pada gambar 3.12.



Gambar 3.12 Fungsi keanggotaan masukan delta error (err)

Fungsi keangotaan keluaran digunakan fungsi keangotaan *singletone* (konstan). Batas keanggotaan keluaran oleh NE, ZE, dan PE ditentukan berdasarkan sudut picu yang diinginkan fungsi keanggotaannya diperlihatkan pada gambar 3.13 batas keanggotaan keluarnya diperoleh melalui pengujian berulang ke palnt sesungguhnya diperoleh masing-masing NE = 0, ZE = 80 dan PE = 100, pada gambar tersebut μ_x(kel) adalah derajat keanggotaan himpunan *fuzzy* keluaran x dengan nilai maksimum adalah 255 (sesuai dengan resulusi mikrokontroler).



Gambar 3.13 Batas fungsi keanggotaan keluaran

Jumlah keanggotaan *fuzzy* masing-masing variabel *ER*, ΔER, suhu *setpoint* dan *DI* tidak harus seperti yang ditentukan di atas. Jumlah keanggotaan dapat dikurangi ataupun ditambahkan, tergantung dari kriteria kebutuhan. Sebagai contoh, variabel *ER*, ΔER, dapat mempunyai keanggotaan *Negatif* (E),*Medium Error* (ME), *Zero* (ZE) dan *Positif* (PE).

3.4.3. Inferensi

Aturan-aturan logika *fuzzy* yang akan dipergunakan sangat tergantung pada sistem yang akan dikendalikan. Disamping itu tujuan pengendalian juga sangat mempengaruhi penentuan aturan-aturan kendali. Pengendalian suhu melalui pengaturan tegangan mempunyai aturan kendali yang unik. Tidak ada rumusan yang pasti dalam menentukan aturan-aturan *fuzzy* dan fungsi keanggotaan masukan dan keluaran. Pengetahuan seorang operator yang berpengalaman dalam pengendalian dapat dijadikan acuan dalam perancangan pengendali logika *fuzzy*.

Dalam menentukan fungsi keanggotaan himpunan *fuzzy*, yang perlu diperhatikan adalah nilai aktual variabel masukan dan keluaran dari pengendali logika *fuzzy* serta semesta pembicaraanya, yaitu rentang nilai yang mungkin dicapai. Faktor penting lain yang harus diperhatikan adalah *completeness*, yaitu suatu algoritma pengendali *fuzzy* harus mampu memutuskan aksi kendali yang tepat untuk semua masukan dalam semesta pembicaraan. Faktor *completeness* ini dipenuhi dengan merancang fungsi keanggotaan yang *overlapping* untuk menjamin semua nilai dapat diakomodasi oleh salah satu atau beberapa fungsi keanggotaan himpunan *fuzzy*.

Jumlah aturan *fuzzy* yang akan digunakan dalam basis aturan *fuzzy* dapat dipengaruhi oleh kriteria kualitas pengendalian yang ingin dicapai. Dalam pendekatan sistem ahli konvensional (*conventional expert system*), jika jumlah predikat untuk setiap masukan sebanyak *m* dan jumlah variabel masukan sistem sebanyak *n*, maka aturan sebanyak m^n jelas diperlukan untuk kelengkapan. Sementara basis aturan *fuzzy* memerlukan jumlah aturan yang lebih sedikit untuk

memenuhi kelengkapan. Jumlah aturan kendali *fuzzy* dapat lebih sedikit karena fungsi keanggotaan himpunan *fuzzy* yang *overlapping*, waktu komputasi akan lebih cepat dan kinerja pengendali akan lebih baik. Tidak ada prosedur umum yang dapat digunakan untuk menentukan jumlah aturan kendali *fuzzy* yang optimal, tetapi sejumlah faktor kriteria mempengaruhi pengambilan keputusan, yaitu kinerja pengendali, kinerja sistem keseluruhan, waktu komputasi, fungsi-fungsi keanggotaan untuk variabel kendali dan lain-lain.

Karena himpunan keanggotaan untuk masing-masing input logika *fuzzy* direncanakan hanya tiga keanggotaan yaitu negatif (NE), zero (ZE), positif (PE), maka aturan pengendaliannya (if then rule) akan menjadi 9 aturan. Aturan-aturan tersebut disusun berdasarkan tujuan pengendalian. Himpunan keanggotaan keluaran juga menggunakan tiga keanggotaan seperti dilihat pada tabel 3.2. Derajat keanggotaan masukan dibuat *singletone* untuk meningkatkan resolusi. Sedangkan batas-batas keanggotaan keluaran ditentukan berdasarkan kemampuan *timer* yang digunakan.

Aturan-aturan yang disusun digunakan untuk mengendalikan putaran motor AC untuk mengendalikan suhu ruangan tergantung temperatur suhu yang diinginkan.

Logika penentuan aturan kendali diberikan sebagai berikut

1. Jika error masih besar dan perubahan error positif berarti suhu yang diinginkan masih jauh, sudut picu akan dibuat lebar.
2. Jika error sudah kecil dan perubahan error positif kecil berarti suhu yang diinginkan hampir tercapai, sudut picu akan dibuat sempit.

3. Jika error kecil dan perubahan error nol berarti suhu yang diinginkan telah tercapai sehingga sudut picu akan di nol kan.

Berdasarkan tujuan pengendalian dan logika pemikiran diatas disusun 9 aturan kendali seperti diperlihatkan pada tabel 3.2

Pada perancangan ini digunakan aturan kendali aturan berjumlah.

Jika ada tiga masukan error dan ada tiga masukan *change error* maka akan ada tiga kali tiga aturan (*rules*).

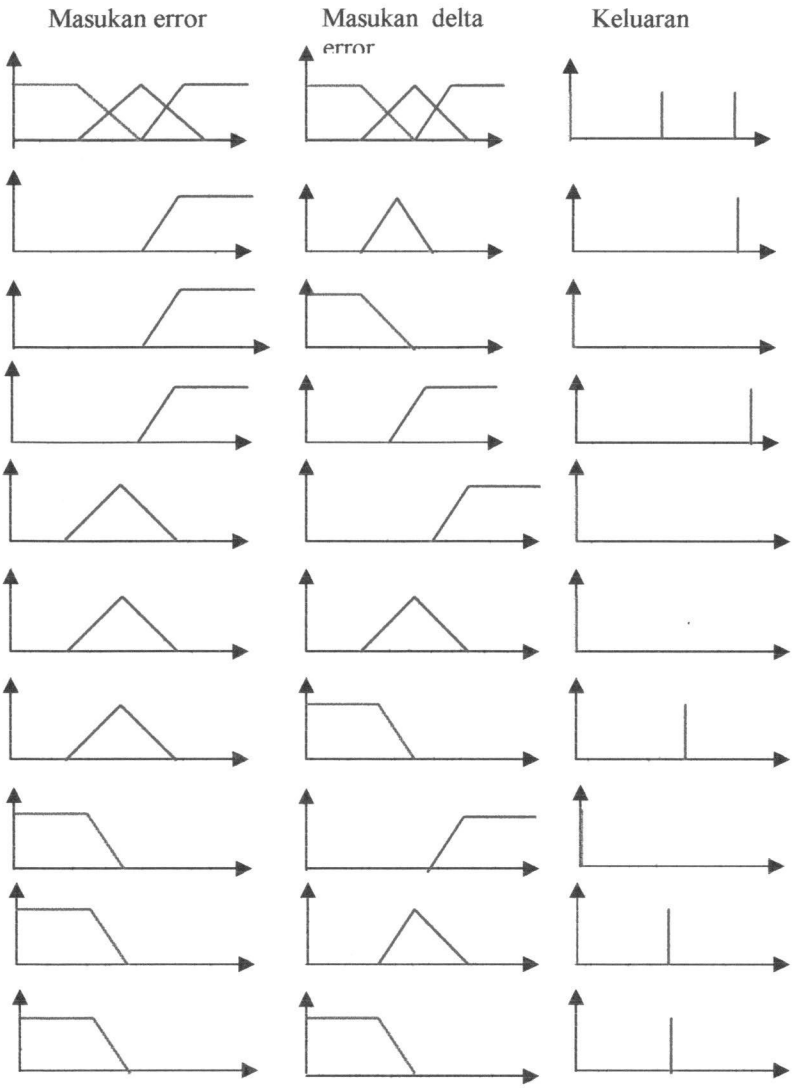
Dimana aturan tiga yang dibuat berdasarkan pengalaman, asumsi pembuat dan konsep aturan sederhana memakai pendekatan *hearistik*

Tabel 3.2 penentuan himpunan aturan

No aturan	Err	Δ Err	Kel	Keterangan (Err, Δ Err,kel
1	P	Z	P	Erorr Positif, delta err nol, keluaran maksimal
2	Z	N	N	Error nol, delta err negatif, keluaran minimum
3	N	Z	N	Error negatif, delta err nol, keluaran minimum
4	Z	P	P	Error nol, delta err positif, keluaran maksimal
5	Z	Z	Z	Error nol, delta err nol, keluaran nol
6	P	N	P	Error positif, delta err negatif, keluaran nol
7	N	N	N	Error negatif, delta err negatif, keluaran minimum
8	N	P	N	Error negatif,delta err positif, keluaran nol
9	P	P	P	Error positif,delta err positif, keluaran maksimal

Gambar 3.14 berikut ini memperlihatkan secara grafis proses inferensi menggunakan metode MAX-MIN untuk tujuan pengendalian putaran kipas berdasarkan suhu, gambar 3.14 juga memperlihatkan alasan digunakanya variabel Δerr sebagai masukan logika *Fuzzy*. Suhu yang diharapkan (setpoint) adalah 30^0C, pada kondisi awal himpunan masukan logika *fuzzy* 40^0C, keluaran defuzzifikasi adalah sebagai berikut.

Keanggotaan masukan error tidak mengalami perubahaan. Gambar 3.14 memperlihatkan fungsi keanggotaan masukan posisi (p_v) dan gambar 3.14 memperlihatkan fungsi keanggotaan keluaran setelah mengalami perubahan dan pengujian sesuai dengan tujuan pengendalian.



Gambar 3.14 proses inferensi dengan metode MAX-MIN

3.4.4. Prosedur Defuzzifikasi

Proses defuzzifikasi merupakan bagian akhir dari suatu logika *fuzzy* yang bertujuan untuk memetakan himpunan *fuzzy* hasil inferensi ke himpunan keluaran (himpunan tegas) yang berupa aksi kendali. Ada beberapa metode defuzzifikasi yang sering digunakan sepert *Mean OF Maxsimum* (MOM), dan *Center OF Area*

(COA). Untuk penelitian ini digunakan metode COA dikarenakan kemudahan dalam komputasi, sifat kontinuitas dan kemudahan implementasinya. Fungsi keanggotaan keluaran yang digunakan adalah fungsi keanggotaan *singleton.* Untuk tujuan pengendalian suhu

3.4.5. Penegasan (Defuzzifikasi)

Setelah dilakukan evaluasi masukan menerapkan basis aturannya, pengedali logika *fuzzy* menghasilkan keluaran untuk diberikan pada sistem yang dikendalikan misalnya dengan cara mengeluarkan tegangan atau nilai arus listrik pada nilai tertentu untuk mengendalikan kecepatan putaran kipas angin. Pengendali logika *fuzzy* harus mengubah variabel keluaran *fuzzy* menjadi nilai-nilai tegas yang dapat digunakan untuk mengedalikan sistem. Proses ini disebut sebagai penegasan (defuzzifikasi).

Keluaran logika *fuzzy* (aksi kendali) digunakan untuk mengendalikan saklar triac untuk mengatur lamanya buka dan menutupnya saklar sehingga sebanding dengan daya yang disalurkan untuk mengatur putaran kipas. Besarnya aksi kendali sama dengan keluaran logika *fuzzy*, yang dirumuskan:

$$\text{Kel (k)} = \textit{defuzzifer}\ (\text{kel}) \qquad (3.3)$$

Defuzzifer yang digunakan untuk penelitian ini adalah metode *center of area* (COA), berdasarkan gambar 3.13 diperoleh hasil defuzzifikasi untuk tujuan pengendalian putaran kipas angin sesuai dengan suhu ruangan.

$$kel = \frac{\mu_{NE}(kel).0 + \mu_{ZE}(kel).80 + \mu_{PE}(kel).100}{\mu_{NE}(kel) + \mu_{ZE}(kel) + \mu_{PE}(kel)}$$

Kel adalah hasil proses defuzzifikasi, $\mu_{NE}(kel), \mu_{ZE}(kel), \mu_{PE}(kel)$ yang merupakan derajat keanggotaan keluaran masing-masing untuk NE, ZE, dan PE. 0, 80, dan 100 Adalah batas keanggotaan keluaran masing-masing untuk NE, ZE dan PE.

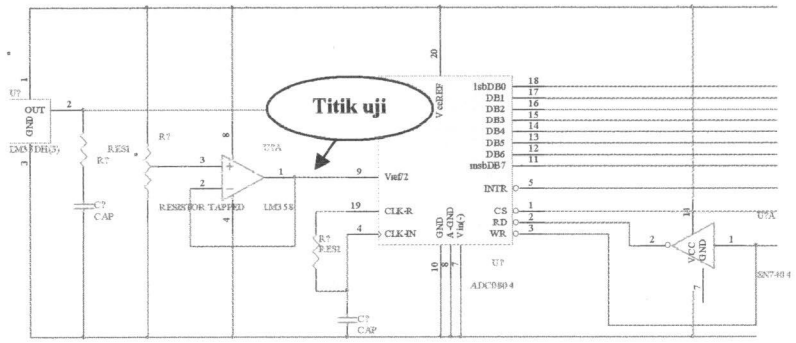
BAB IV

ANALISIS DAN PEMBAHASAN

4.1. Kalibrasi sensor suhu

Sensor suhu yang digunakan adalah LM 35 buatan *national semiconductor*, dalam *data sheet* yang dilampirkan dinyatakan bahwa LM 35 mempunyai sensitivitas 10 mV/derajat celcius.

Pengamatan dilakukan dengan menggunakan rangkaian berikut.



Gambar 4.1 Titik uji pengamatan untuk keluaran sensor suhu

Operasional amplifier yang digunakan diatas adalah sebagai *buffer* untuk menjaga agar tegangan referensi tetap sebesar 2,5 Volt.

Pada saat suhu disekitar LM 35 100 ^{o}C maka sebenarnya keluaran (Vi) tegangannya 1 Volt sebelum mendapat penguatan. Ini dapat dijadikan sebagai patokan pengkalibarsian untuk menentukan tegangan (Vi).

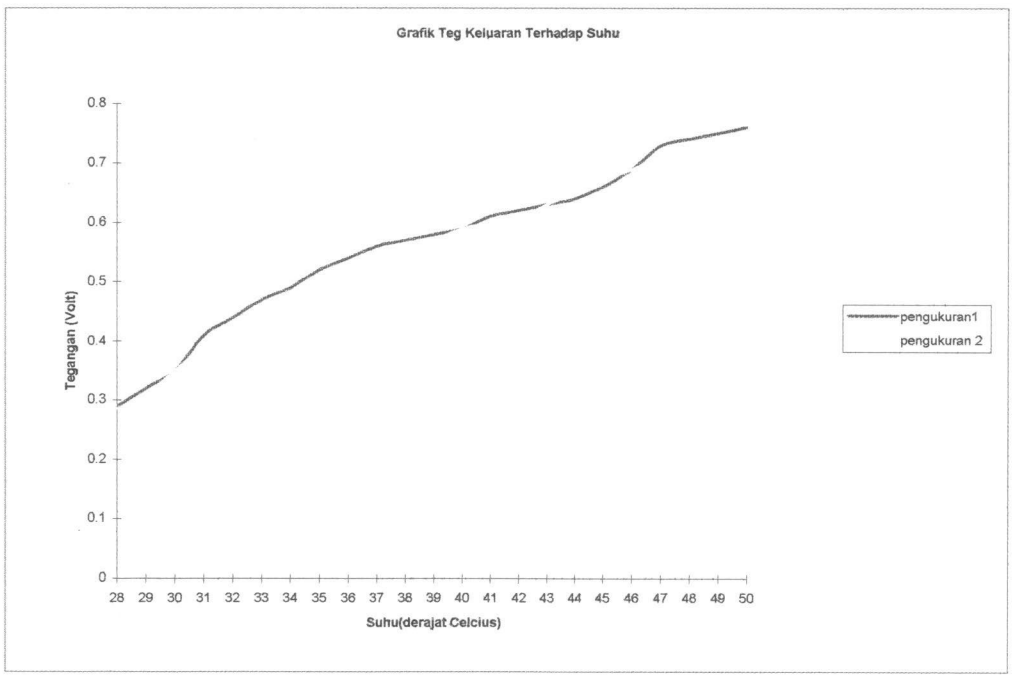
Suhu pada model ruang yang terbaca oleh sensor akan menghasilkan tegangan dan akan dihubungkan dengan masukan op amp, selanjutnya pada op amp akan mendapatkan data output yang penuh pada ADC.

Tabel 4. 1. Pengkalibrasian LM 35

NO	Suhu	Pengukuran 1	Pengukuran 2
1	28 ^{0}C	0,28 V	0,28 V
2	29 ^{0}C	0,31	0,32 V
3	30 ^{0}C	0,35 V	0,35 V
4	31 ^{0}C	0,39 V	0,40 V
5	32 ^{0}C	0,41 V	0,44 V
6	33 ^{0}C	0,43 V	0,47 V
7	34 ^{0}C	0,45 V	0,49 V
8	35 ^{0}C	0,49 V	0,52 V
9	36 ^{0}C	0,51 V	0,54 V
10	37 ^{0}C	0,53 V	0,56 V
11	38 ^{0}C	0,55 V	0,57 V
12	39 ^{0}C	0,57 V	0,58 V
13	40 ^{0}C	0,59 V	0,59 V
14	41 ^{0}C	0,60 V	0,61 V
15	42 ^{0}C	0,61 V	0,62 V
16	43 ^{0}C	0,63 V	0,63 V
17	44 ^{0}C	0,65 V	0,64 V
18	45 ^{0}C	0,67 V	0,66 V
19	46 ^{0}C	0,69 V	0,69 V

20	47 ^{0}C	0,71 V	0,71 V
21	48 ^{0}C	0,73 V	0,74 V
22	49 ^{0}C	0,74 V	0,75 V
23	50 ^{0}C	0,75 V	0,76 V

Pengukuran tegangan keluran dilakukan dengan menggunkan voltmeter digital dan dibandingkan dengan suhu ruangan terukur termometer. Termometer yang digunakan bukan merupakan termometer standard untuk kalibrasi suhu.



Gambar 4.2 Grafik suhu terhadap tegangan keluaran sensor

Hasil pengujian tampak seperti pada tabel 4.1 dan gambar 4.2 yang menggambarkan bahwa hasil pengujian suhu telah menunjukan kelinieran walaupun belum sesuai benar dengan koralitenstik yang ada pada data sheet.

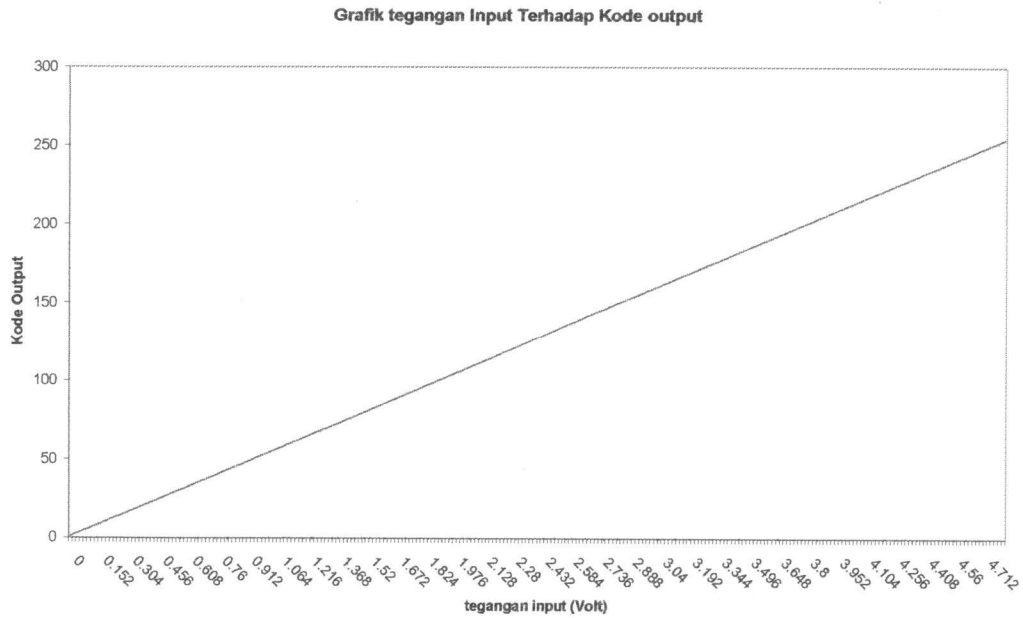
Belum sesuainya kelinieran yang didapat dengan data sheet juga dikarenakan termometer yang digunakan untuk kalibrasi suhu bukan merupakan termometer standard yang digunakan untuk pengkalibrasian, sehingga pengukuran oleh termometer tidak dapat digunakan sebagai acuan pengukuran yang valid.

4.2. Pengujian ADC

Proses pengaksesan data dari ADC. Pada port P1.1 dihubungkan kaki *CS* dan port P1.2 dihubungkan dengan kaki WRT pada ADC. Dengan memberi pulsa sesaat pada kaki CS dan WRT maka ADC akan menjadikan aktif dan siap akan mengkonversi sinyal analog keluaran dari sensor arus. Pada saat konversi sinyal EOC akan *low*, jika konversi berhasil maka menunggu kapan waktu akan berakhir. Waktu konversi berakhir pada saat input atau output di data sheet waktu konversi berlangsung selama 100 nano detik (data sheet, 1989).

Setelah selesai konversi maka pada input atau output akan mendapat sinyal *high* instruksi selanjutnya adalah membaca data yang telah dikonversi.

Pengujian ADC dilakukan dengan mengamati hasil konversi pada kaki data ADC (D0-D7) dengan melakukan variasi tegangan pada input analog ADC. Hasil pengujian seperti pada gambar 4.2.



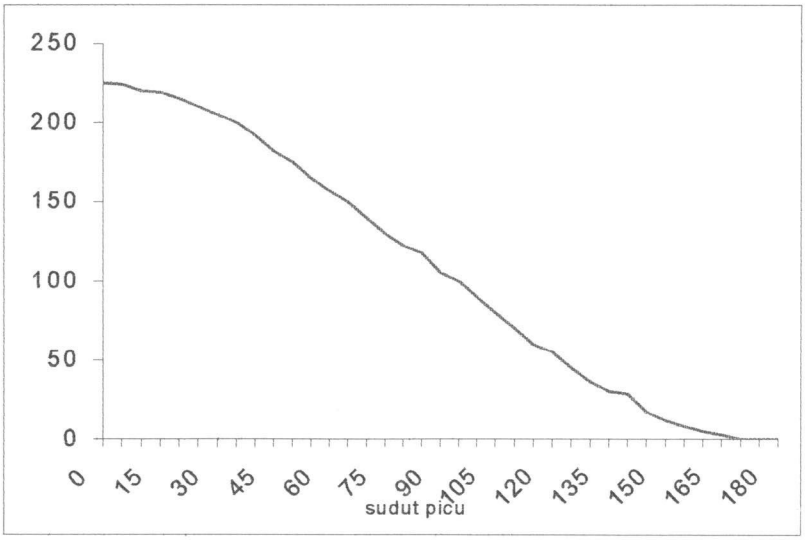
Gambar 4.3 grafik hubungan tegangan analog input terhadap kode output

Grafik pada gambar 4.3 menunjukan bahwa ADC 0804 mempunyai karakteristik linier antara masukan tegangan terhadap kode data keluaran.

Pada rangkaian sebenarnya ADC yang digunakan mempunyai jangkah dan pengaturan nol yang disesuaikan dengan spesifikasi yang telah ditentukan pada rancangan.

4.3. Pengamatan Watak Proses yang Dikendalikan

Pengamatan sudut picu dilakukan dengan melihat posisi pulsa terhadap titik balik gelombang (titik nol) dengan osiloskop. Pengamatan tegangan dilakukan dengan mengukur tegangan pada kipas menggunakan voltmeter.



Gambar 4.4 Hubungan sudut picu dan tegangan hasil pengamatan

Keterangan:

1. Dari gambar, terlihat bahwa hubungan sudut picu dengan tegangan yang dihasilkan adalah tidak linier.
2. Bentuk matematis dari watak sudut picu dengan tegangan tidak perlu dicari karena logika *fuzzy* tidak memerlukan model matematis sistem.
3. Berdasarkan hasil pengamatan diatas, kita tetapkan fungsi keanggotaan sudut picu, yaitu besar = 0 = 100% tegangan max, sedang = 60 = 70% tegangan max, kecil = 120 = sekitar 25% tegangan max, dan kecil 155 = 3% tegangan max.

4.4. Pengamatan Perubahan Tegangan *Thrystor* Terhadap Suhu Set Point Tertentu

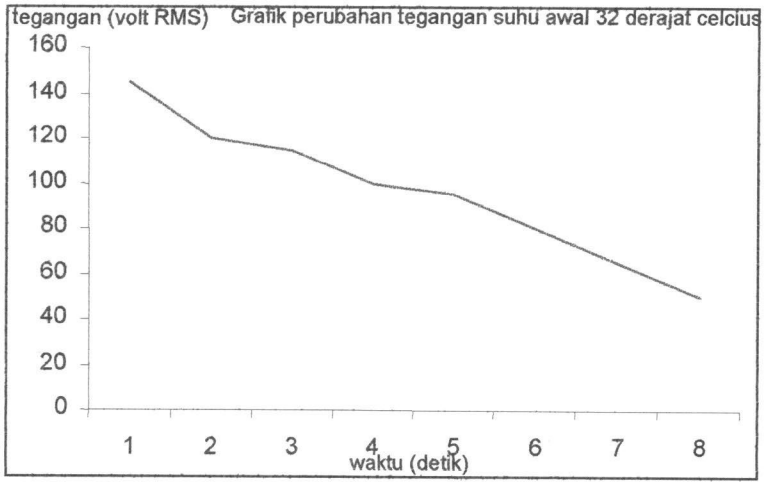
Pengamatan ini dimaksudkan untuk mengetahui kecepaatan perubahan tegangan *thrystor* terhadap suhu set point tertentu. Pengamatan dilakukan dengan

melihat perubahan tegangan *thrystor* terhadap waktu dengan menggunakan voltmeter sehingga tegangan yang diamati adalah merupakan tegangan *root mean square* (RMS) dari gelombang pensaklaran *thrystor*.

Alasan digunakannya pengamatan tegangan ini adalah besarnya tegangan RMS dapat teramati perubahan dan tegangan yang teramati merupakan representasi dari tegangan yang digunakan oleh kipas untuk berputar.

Hasil pengamatan akan menjelaskan perubahan putaran kipas dengan mengamati perubahan tegangan melalui volt meter.

Gambar grafik menjelaskan pengamatan perubahan tegangan terhadap waktu dengan suhu awal 32 derajat celsius dan suhu setpoint sebesar 29 derajat celcius.



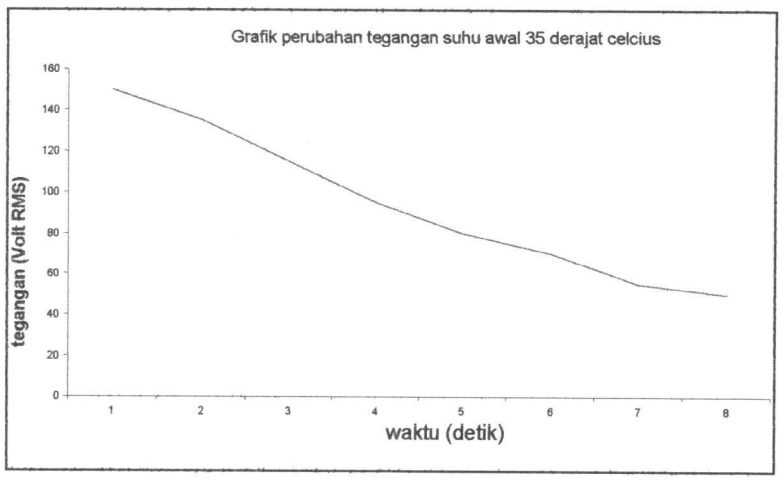
Gambar 4.5 Grafik perubahan suhu awal 32 derjat celcius

Hasil pengamatan menunjukan bahwa perubahan tegangan untuk laju pendinginan tidak linier karena suhu pendinginan yang dihasilkan kipas tidak

linier. Hasil pengamatan juga menunjukan bahwa kecepatan pendinginan perubahannya lambat untuk setiap titik pengamatan.

Bila beda suhu antara suhu awal dan suhu yang diinginkan tidak besar maka tegangan awal tidak terlalu besar. Hal ini menunjukan bahwa putaran kipas tidak dibuat cepat oleh algoritma *fuzzy*.

Aturan kendali *fuzzy* yang diterapkan bila beda suhu yang terjadi kecil adalah memberikan keluran maksimal tetapi hasil dari inferensi tidak dapat menyebabkan tegangan untuk kipas menjadi maksimal. Karena hasil inferensi yang digunakan untuk membaca *look-up table* mengarahkan sudut picu lebih dari 60 derajat terhadap satu siklus (180 derajat maksimal).



Gambar 4.6 Grafik perubahan tegangan suhu awal 35 derajat celcius

Bila perbedaan suhu semakin tinggi maka perubahan tegangan semakin lambat dan kecepatan perubahan tegangan terjadi pada waktu suhu berbeda sebesar 2 derajat celcius dari suhu yang diinginkan. Perubahan semakin cepat karena putaran kipas melambat dan suhu yang diinginkan telah mendekati sehingga tampak bahwa perubahan tegangan semakin cepat yang menunjukan

bahwa putaran kipas menjadi lambat dan berhenti ketika mencapai beda suhu sebesar 0,5 derajat celcius.

Peranan algoritma *fuzzy* tampak dari hasil yang diperoleh berdasarkan grafik diatas. Beda suhu yang tinggi antara suhu awal dan setpoint (suhu yang diinginkan) menyebabkan tegangan pada *thrystor* sangat besar. Hal ini menandakan bahwa error berada pada fungsi keanggotaan positif dan perubahan error berada pada keangotaan nol yang menyebabkan keluaran akan berada pada batas maksimal sehingga pembacaan *look-up table* akan berada pada posisi 1 kurang dari 60 derajat untuk sinyal 180 derajat.

Thrystor akan mengalirkan arus penuh pada saat tersebut dan putaran kipas menjadi sangat cepat untuk mendinginkan suhu. Tetapi karena error yang besar maka kecepatan putaran awal kipas akan lama daripada kecepatan putaran kipas berikutnya karena menjaga agar pendinginan cepat terjadi yang dikendalikan oleh algoritma *fuzzy* sesuai dengan aturan yang dibuat dalam perancangan yang dijelaskan pada bab 3.

4.5. Pengamatan Perubahan Suhu Terhadap Waktu Pendinginan

Pengamatan ini dimaksudkan untuk mengetahui kecepatan pendinginan dengaan melakukan pengamatan terhadap suhu awal dan suhu akhir dengan waktu yang diperlukan untuk mencapai suhu akhir.

Hasil pengamatan ditunjukan pada tabel 4.2. Pada tabel 4.2 terdapat data setpoint yang merupakan suhu yang diinginkan sebenarnya, tetapi hasil pengamatan hanya dilakukan ketika suhu mencapai suhu akhir.

Tabel 4.2 Hasil pengamatan pada model ruang

No	Suhu Awal	Set Point	Suhu Akhir	Waktu
1	30	28,5	28,6	120dt
2	31	28,5	28,6	140dt
3	32	28,5	28,6	160dt
4	33	28,5	28,6	140dt
5	34	28,5	28,6	210dt
6	35	28,5	28,6	200dt
7	36	28,5	28,6	180dt
8	37	28,5	28,6	190dt
9	38	28,5	28,6	220dt
10	39	28,5	28,6	250dt

4.6. Kemampuan Pengendalian

Sistem dikatakan dapat dikendalikan adalah apabila dapat menghasilkan keluaran sistem sesuai dengan yang diharapkan. Logika *fuzzy* dengan mikrokontroler dapat mengendalikan keluaran sistem sehingga keluaran sistem sesuai dengan yang diharapkan pada keadaan tunaknya.

Dengan mengacu dari deretan data yang didapatkan, terlihat keluaran sesekali sedikit melebihi ataupun sedikit dari set point yang diharapkan. Namun kesalahan keadaan tunak yang ditimbulkan tidak besar, dengan maksimal 0,5 derajat Celcius. Kesalahan yang terjadi juga terlihat tidak memliki pola dan cenderung dikategorikan sebagai suatu pola acak.

Kesalahan yang terjadi pada setiap pengendalian yaitu +- 0,5 derajat Celcius dapat dianalisa adalah akibat :

1. Akurasi ADC adalah 0,1 derajat Celcius / bit. Perancangan dengan akurasi 0,1 derajat Celcius mengacu pada datasheet LM35 sebagai sensor suhu yang memiliki akurasi tipikal 10 mV/derajat celcius dan hasil yang linier.
2. Perubahan yang dapat dideteksi untuk menghitung kesalahan adalah 0,1 dajat Celcius.

BAB V

PENUTUP

5.1 Kesimpulan

1. Setpoint temperatur tidak pernah tercapai tapi akan beselisih beberapa derjat celcius terhadap suhu yang diinginkan (setpoint). Hal ini diakibatkan karena tidak adanya pengaturan laju pemanasan dan aturan-aturan logika *fuzzy* yang diterapkan .
2. Model ruangan suhu yang dirancang sangat tergantung suhu sekitarnya, akibatnya hasil pembacaan proses value pada display dipengaruhi oleh temperatur sekitar model ruang.
3. Pengaturan fase untuk pengendalian putaran kipas tidak menghasilkan pengendalian putaran yang halus karena daya yang tersalurkan untuk megendalikan kipas dengan teknik posisi fase tidak terdapat beda yang signifikan.

5.2 Saran

1. Fungsi keanggotaan hendaknya dapat ditambahkan untuk memperoleh pengendalian sistem temperatur seperti yang diinginkan.
2. Untuk memperoleh hasil yang optimal keluaran pengendali tidak hanya tidak hembusan kipas tetapi juga pengendalian laju pemanasan bola lampu.
3. Sistem pengendali ini dapat dikembangkan dengan menghubungkan mikrokontroler dan komputer. Dimana komputer digunakan untuk

pengolahan dan pengaturan logika *fuzzy* serta untuk mengolah hasil pengukuran.

4. Pengolah dapat menggunakan mikrokontroler dengan lebar BUS yang besar (16) dan sensor suhu yang memiliki sensivitas tinggi serta ADC yang mempunyai resolusi yang tinggi sehingga pengendalian sistem dapat diperoleh dengan peresisi yang tinggi.

DAFTAR PUSTAKA

1. Budi Susanto, 2002, Power Kontrol.
2. Ir.Hj. Budi Astuti, 1999, Teknik Tenaga Listrik,Universitas Isalam Indonesia.
3. Malvino Albert Paul, Ph, D, *Prinsip-Prinsip Elektronika*, Erlangga, 1992.
4. *National Semiconductor Data sheet,*1995.
5. Putra Agfianto Eko, 2002, *Belajar Mikrokontroler AT89C51/53/53 Teori dan Aplikasi,* Gava Media, Yogyakarta.
6. Sri Kusumadewi (Edisi 2) 2001 *Analisis dan desain sistem fuzzy.*
7. Sri Kusumadewi, 2002, *Analisi dan Desain Sistem Fuzzy Menggunakan Tool BOX matlab,* Graha Ilmu.
8. Wasito, S, 1991, *Data Sheet Book 1*, Gramedia, Jakarta.
9. Wasito, S, 1985, *Data Sheet Book 1, Data IC Linier, TTL, dan CMOS*, PT Elex Media Komputindo, Kelompok Gramedia, Jakarta.

LAMPIRAN

```
rogram lengkap  fuzzy
efinisi dan lokasi alamat Port PPI 8255
rtA         equ         2000H
rtB         equ         2001H
rtC         equ         2002H
lwrd        equ         2003H
luaran      equ         5000H
ariabels
ag          equ         20h
luhan       equ         22h
tuan        equ         23h
cahan       equ         24h
g0          equ         25h         ;pecahan
g1          equ         26h         ;satuan
g2          equ         27h         ;puluhan
x           equ         28h
            equ         29h
d1          equ         2ah         ;pecahan
d2          equ         2bh         ;satuan
d3          equ         2ch         ;puluhan
ta_kpd      equ         2dh
ya          equ         2eh
ariabel fuzifikasi
            equ         2fh         ;data suhu dari adc
tpoint      equ         30h         ;data suhu setelan
ror         equ         31h
rbef        equ         32h
err         equ         33h
r_pos       equ         34h
r_nol       equ         35h
r_neg       equ         36h
errpos      equ         37h
errneg      equ         38h
errnol      equ         39h
tput        equ         3ah
tnol        equ         3bh
tpos        equ         3ch
tneg        equ         3dh
ariabel untuk defuzifikasi
li1         equ         3eh
li2         equ         3fh
li3         equ         40h
li4         equ         41h
ks          equ         42h
n           equ         43h
n1          equ         44h
n2          equ         45h
n3          equ         46h
outneg      equ         47h
outpos      equ         48h
outnol      equ         49h
ariabel untuk bagi 16 bit
.vidend     equ         4ah
.vidend1    equ         4bh
.visor      equ         4ch
sil         equ         4dh
g1          equ         4eh
g2          equ         4fh
g3          equ         50h

lags control
C                       bit         P1.0        ;tanda konversi adc selesai
                        bit         P1.1        ;chip select adc
T                       bit         P1.2        ;write atau not read adc
suhu                    bit         P1.3        ;led data suhu
point                   bit         P1.4        ;led setpoint
```

```
;Y                      bit         P1.5        ;keypad diambil datanya
;IAC                    bit         P1.7        ;pengendali triac
;ecimal                 bit         Flag.0      ;tanda tombol desimal ditekan
.ntang                  bit         Flag.1      ;tanda tombol * ditekan
;gar                    bit         Flag.2      ;tanda tombol # ditekan
;tik                    bit         Flag.3      ;tanda telah 1 detik
.me_sampling            bit         flag.4      ;tanda time sampling
;lesai                  bit         flag.5
;ri                     bit         flag.6
            ORG         0000h
;i:         ;Ljmp       Rst
            ORG         0003h
            ;Ljmp       Itsra
            ORG         0013h
            ;Ljmp       Itsrb
            org         4000h       ;Program dimulai disini
;st:        ljmp        mulai
            org         4003h       ;Program keypad
;srа:       lcall       itsr0
            reti
            org         400bh       ;Program sulut triac
            Lcall penyulutan
            reti
            org         4013h       ;Program zero crossing
;srb:       lcall       TitikNol
            reti
            org         4033h       ;Awal penulisan Program
*****************************************
inisialisasi
*****************************************
;nisialisasi:
            mov         dptr,#ctlwrd
            mov         a,#99h
            movx        @dptr,a
            mov         seg0,#11100000b         ;seg  kanan
            mov         seg1,#11010000b         ;seg tengah
            mov         seg2,#10110000b         ;seg kiri
            mov         pad1,#00
            mov         pad2,#00
            mov         pad3,#00
            mov         tix,#100
            mov         ts,#200
            mov         flag,#00

            mov         pv,#00
            mov         setpoint,#00h
            mov         error,#00h
            mov         c_err,#00h
            mov         errbef,#00h

            mov         c_errneg,#00h
            mov         c_errnol,#00h
            mov         c_errpos,#00h

            mov         err_neg,#00h
            mov         err_nol,#00h
            mov         err_pos,#00h

            mov         mfoutneg,#00h
            mov         mfoutnol,#00h
            mov         mfoutpos,#00h

            mov         maks,#00h
            mov         min,#00h
            mov         min1,#00h
            mov         min2,#00h
```

```
            mov         min3,#00h
            mov         outneg,#00h
            mov         outnol,#80
            mov         outpos,#100
            mov         output,#10h

            mov         kali1,#00h
            mov         kali2,#00h
            mov         kali3,#00h
            mov         kali4,#00h
            mov         dividend,#00h
            mov         dividend1,#00h
            mov         divisor,#00h
            mov         hasil,#00h

            setb        ex0
            setb        it0
            setb        ea
            ret
;*****************************************
;rogram intrupsi keypad
;*****************************************
;sr0:        clr         ie.0
            push        dph
            push        dpl
            push        acc
            mov         dptr,#porta
            clr         KEY                     ;aktifkan port ic keypad
            movx        a,@dptr
            setb        KEY                     ;non aktifkan port ic keypad
            anl         a,#00001111b
            mov         data_kpd,a

            cjne        a,#0bh,desimal
            setb        bintang
            sjmp        desimal3

;esimal:     cjne        a,#0ah,desimal2
            setb        pagar
            sjmp        desimal3

;esimal2:    setb        Decimal

;esimal3:    pop         acc
            pop         dpl
            pop         dph
            setb        ie.0
            ret
;*****************************************
;elay:       mov         r7,#0ffh
            mov         r6,#0ffh
;0:          djnz        r6,l1
;1:          djnz        r7,l0
            mov         r7,#0ffh
            mov         r6,#0ffh
;2:          djnz        r6,l3
;3:          djnz        r7,l2
            mov         r7,#0ffh
            mov         r6,#0ffh
;4:          djnz        r6,l5
;5:          djnz        r7,l4
            ret

;itung_error:
            mov         a,pv
            clr         c
```

```
            subb        a,setpoint
            jc          pv_besar
            add         a,#127
            jc          err_ovp
            mov         error,a
            ret

:r_ovp:
            mov         a,#0ffh
            mov         error,a
            ret

/_besar:
            mov         a,setpoint
            clr         c
            subb        a,pv
            mov         r7,a
            mov         a,#127
            subb        a,r7
            jc          err_ovn
            mov         error,a
            ret

:r_ovn:
            mov         a,#00
            mov         error,a
            ret

itung_c_error:
            clr         c
            mov         a,error
            subb        a,errbef
            jc          errbef_besar
            add         a,#15
            jc          nt1
            clr         c
            cjne        a,#30,nt
t:          jc          c_errovp
t1:         mov         c_err,#30
            ret

_errovp:
            mov         c_err,a
            ret

rrbef_besar:
            clr         c
            mov         a,errbef
            subb        a,error
            mov         r7,a
            mov         a,#15
            clr         c
            subb        a,r7
            jc          c_errovn
            mov         c_err,A
            ret

_errovn:
            mov         c_err,#00
            ret

abel:
            push        dph
            push        dpl
            mov         a,error
            mov         dptr,#mfen
```

```
        movc    a,@a+dptr
        mov     err_neg,a

        mov     a,error
        mov     dptr,#mfez
        movc    a,@a+dptr
        mov     err_nol,a

        mov     a,error
        mov     dptr,#mfep
        movc    a,@a+dptr
        mov     err_pos,a

        mov     a,c_err
        mov     dptr,#mfcen
        movc    a,@a+dptr
        mov     c_errneg,a

        mov     a,c_err
        mov     dptr,#mfcez
        movc    a,@a+dptr
        mov     c_errnol,a

        mov     a,c_err
        mov     dptr,#mfcep
        movc    a,@a+dptr
        mov     c_errpos,a
        pop     dpl
        pop     dph
        ret

ifikasi:
        acall   hitung_error
        acall   hitung_c_error
        acall   tabel
        ret

erensi:
        acall   hit_mfoutneg
        acall   hit_mfoutnol
        acall   hit_mfoutpos
        ret

_mfoutneg:
        mov     r7,err_neg
        mov     r6,c_errneg
        acall   minimum
        mov     a,min
        mov     min1,a
        mov     r7,err_neg
        mov     r6,c_errnol
        acall   minimum
        mov     a,min
        mov     min2,a
        mov     r7,err_nol
        mov     r6,c_errneg
        acall   minimum
        mov     a,min
        mov     min3,a
        acall   maksimum
        mov     a,maks
        mov     b,#2
        div     ab
        mov     mfoutneg,a
        ret
```

```
.t_mfoutpos:
            mov         r7,err_nol
            mov         r6,c_errpos
            acall       minimum
            mov         a,min
            mov         min1,a
            mov         r7,err_pos
            mov         r6,c_errnol
            acall       minimum
            mov         a,min
            mov         min2,a
            mov         r7,err_pos
            mov         r6,c_errpos
            acall       minimum
            mov         a,min
            mov         min3,a
            acall       maksimum
            mov         a,maks
            mov         b,#2
            div         ab
            mov         mfoutpos,a
            ret

it_mfoutnol:
            mov         r7,err_neg
            mov         r6,c_errpos
            acall       minimum
            mov         a,min
            mov         min1,a
            mov         r7,err_nol
            mov         r6,c_errnol
            acall       minimum
            mov         a,min
            mov         min2,a
            mov         r7,err_pos
            mov         r6,c_errneg
            acall       minimum
            mov         a,min
            mov         min3,a
            acall       maksimum
            mov         a,maks
            mov         b,#2
            div         ab
            mov         mfoutnol,a
            ret

subrutin mencari nilai minimum
inimum:
            clr         c
            mov         a,r6
            subb        a,r7
            jc          banding
            mov         a,r7
            mov         min,a
            ret

anding:
            mov         a,r6
            mov         min,a
            ret
subrutin mencari nilai maksimum
aksimum:
            clr         c
            mov         a,min1
            subb        a,min2
            jc          besar_min2
```

```
            mov         a,min1
            subb        a,min3
            jc          besar_min3
            mov         a,min1
            mov         maks,a
            ret

esar_min2:
            clr         c
            mov         a,min2
            subb        a,min3
            jc          besar_min3
            mov         a,min2
            mov         maks,a
            ret

esar_min3:
            mov         a,min3
            mov         maks,a
            ret

efuzifikasi:
            mov         a,mfoutneg
            mov         b,outneg
            mul         ab
            mov         kali1,b
            mov         kali2,a
            mov         a,mfoutnol
            mov         b,outnol
            mul         ab
            mov         kali3,b
            mov         kali4,a
            mov         a,mfoutpos
            mov         b,outpos
            mul         ab
            clr         c
            add         a,kali2
            add         a,kali4
            mov         dividend1,a
            clr         c
            mov         a,b
            add         a,kali1
            add         a,kali3
            mov         dividend,a
            mov         a,mfoutneg
            clr         c
            add         a,mfoutnol
            add         a,mfoutpos
            jc          jumlah_ov
            mov         divisor,a
            ljmp        bagi

umlah_ov:
            mov         divisor,#0ffh
agi:
            lcall       bagi16ke8
            clr         c
            mov         a,hasil
            subb        a,outnol
            jc          out_neg
            add         a,output
            jc          out_ovp
            mov         output,a
            ret

ut_ovp:
```

```
            mov         output,#0ffh
            ret
ut_neg:
            clr         c
            mov         a,outnol
            subb        a,hasil
            mov         r7,a
            mov         a,output
            clr         c
            subb        a,r7
            jc          out_ovn
            mov         output,a
            ret

ut_ovn:
            mov         output,#00h
            ret

************************************************
program pembagian 16bit dengan 8bit
************************************************
agi16ke8:
            clr         a
            cjne        a,divisor,ovku
embagian_oleh_nol:
            mov         hasil,#0h
            ret
vku:
            mov         hasil,a
            mov         r4,#8
            mov         r5,dividend1
            mov         r6,dividend
            mov         r7,a
            mov         a,r6
            mov         b,divisor
            div         ab
            mov         r6,b
imes_two:
            mov         a,r5
            rlc         a
            mov         r5,a
            mov         a,r6
            rlc         a
            mov         r6,a
            mov         a,r7
            rlc         a
            mov         r7,a

ompare:
            cjne        a,#0,done
            mov         a,r6
            cjne        a,divisor,done
            cjne        r5,#0,done
one:
            cpl         c

uild_quotient:
            mov         a,hasil
            rlc         a
            mov         hasil,a
            jnb         acc.0,loop

ubstract:
            mov         a,r6
            subb        a,divisor
            mov         r6,a
```

```
            mov         a,r7
            subb        a,#0
            mov         r7,a

oop:
            djnz        r4,times_two
            clr         ov
            ret

************************************************
lai:        lcall       delay
            lcall       inisialisasi

nu:         mov         r0,#seg2
            mov         r1,#pad3

nu1:        jnb         Decimal,menu2           ;ada tombol desimal ditekan
            clr         Decimal
            sjmp        menu4

nu2:        jnb         bintang,menu3           ;tombol * ditekan (setpoint)
            clr         bintang
            clr         LDsuhu
            setb        LDpoint                 ;data setpoint
            lcall       setvalue2               ;pogram nulis setpoint ke
splay
            sjmp        menu

nu3:        jnb         pagar,menu1 ;tombol # ditekan (data suhu)
            clr         pagar
            setb        LDsuhu                  ;data suhu
            clr         LDpoint
            lcall       setvalue                ;pogram nulis setpoint
simal ke heksa
            ljmp        kendali

nu4:        mov         a,data_kpd
            mov         @r1,a
            mov         dptr,#portb
            add         a,@r0
            movx        @dptr,a
            anl         a,#00001111b
            add         a,#0f0h
            movx        @dptr,a
            dec         r1
            dec         r0
            cjne        r0,#24h,menu1
            sjmp        menu

ndali:      lcall       delay
            clr         ea
            mov         TMOD,#1                 ;Mode Kerja Timer0
            setb        IT1
            setb        EX1
            setb        EA
            ;MOV        daya,#255   ;p/pmax = 0,5
            mov         dptr,#keluaran

oops:
            clr         Decimal
            clr         pagar
            jb          bintang,menu2
            Lcall       ubah                    ;keluaran di acc
            Lcall       ubahformat  ;masukan di acc
            jnb         detik,loops2
            clr         detik
```

```
            Lcall       tampil
oops2:      ;jb         selesai,loops
            jnb         time_sampling,loops
            clr         time_sampling
            Lcall       fuzifikasi
            Lcall       inferensi
            Lcall       defuzifikasi
            mov         errbef,error
            mov         daya,output
            ;mov        pv,output
            mov         a,pv
            movx        @dptr,a
            ;inc        dptr
            mov         a,dph
            cjne        a,#05fh,loops
            mov         a,dpl
            cjne        a,#0ffh,loops
            setb        selesai
            setb        LDpoint
            sjmp        loops
************************************************************
ubah biner/hex ke desimal 7 segmen, masukan di acc
************************************************************
bahformat:
            mov         puluhan,#00h
            mov         satuan,#00h
            mov         pecahan,#00h
            add         a,#250d
            mov         keri,c
            clr         c
            mov         b,a

lh:         jb          keri,plh1
            subb        a,#64h
            jc          ubh_stn

lh1:        mov         a,b
            clr         c
            subb        a,#64h
            mov         b,a
            inc         puluhan
            jnc         plh
            clr         keri
            sjmp        plh

bh_stn:     mov         a,b
ulh:        mov         b,a
            clr         c
            subb        a,#0ah
            jc          ubh_pch
            inc         satuan
            sjmp        pulh

bh_pch:     mov         pecahan,b
            ret

;****************************************
;subrutin tampilan data suhu ke 7 segmen
;****************************************
ampil:      push        dph
            push        dpl
            mov         a,#0f0h
            clr         acc.4
            add         a,pecahan
            mov         dptr,#portB
            movx        @dptr,a
```

```
            setb        acc.4
            movx        @dptr,a
            ;
            mov         a,#0f0h
            clr         acc.5
            add         a,satuan
            mov         dptr,#portB
            movx        @dptr,a
            setb        acc.5
            movx        @dptr,a
            ;
            mov         a,#0f0h
            clr         acc.6
            add         a,puluhan
            mov         dptr,#portB
            movx        @dptr,a
            setb        acc.6
            movx        @dptr,a
            pop         dpl
            pop         dph
            ret
************************************
erintah konversi adc,hasil di acc
************************************
ah:         push        dph
            push        dpl
            clr         CS                      ;enable CS (1 uS)
            clr         WRT                     ;enable WR (1 uS)
            setb        WRT                     ;disable WR
            setb        CS                      ;disable CS
eckadc:     jb          EOC,checkadc            ;selesai?
            mov         dptr,#portC
            clr         CS                      ;enable CS
            setb        WRT                     ;enable RD
            movx        a,@dptr                 ;load data
            mov         pv,a
            setb        CS                      ;disable CS
            pop         dpl
            pop         dph
            ret
************************************
utin interupsi titik nol
************************************
tikNol:
            push        acc
            push        dph
            push        dpl
            clr         TRIAC                   ;matikan LED MOC3021
            mov         DPTR,#Isi_TH0           ;byte pertama
            mov         A,daya                  ;nilaip/pmax
            movc        a,@A+DPTR               ;ambil nilai tunda
            mov         TH0,A                   ;simpan ke timer0
            mov         DPTR,#Isi_TL0           ;byte ke 2
            mov         A,daya                  ;nilai p/pmax
            movc        A,@A+DPTR               ;ambil nilai tunda
            mov         TL0,A                   ;simpan ke Timer0
            setb        ET0                     ;aktipkan intrupsi Timer0
            setb        TR0                     ;hidupkan timer0
            djnz        tix,exit    ;decrement tix, jika tidak segera keluar
            mov         tix,#100                ;reset variabel ticks
            setb        detik                   ;tanda telah satu detik
it:         djnz        ts,exit2
            mov         ts,#200
            setb        time_sampling
it2:        pop         dpl
            pop         dph
```

```
          pop         acc
          ret
****************************************
Rutin delay timer 0
****************************************
enyulutan:
          clr         TR0         ;hentikkan timer0
          clr         ET0         ;tanpa interupsi timer0
          setb        TRIAC       ;hidupkan LED MOC3021
          nop
          nop
          ;clr        TRIAC
          ret
************************************************************************
subrutin mengubah masukan dari 7 segmen ke dalam bentuk Heksadesimal
************************************************************************
etvalue:
          mov         r0,#pad3                ;r0 untuk register suhu (BCD)
          mov         r2,#0                   ;baca tabel B to H brapa
ali?
          mov         r1,#pag1    ;r1 untuk register heksa
          mov         pag1,#0
          mov         pag2,#0
          mov         pag3,#0
************************************************
subrutin untuk mengubah BCD ke Heksadesimal
************************************************
_TO_H:    mov         b,@r0
          mov         a,r2
          mov         dptr,#BtoH
          movc        a,@a+dptr
          mul         ab
          inc         r1
          add         a,@r1
          mov         @r1,a
          dec         r1
          mov         a,b
          add         a,@r1
          mov         @r1,a
          dec         r0
          inc         r2
          mov         a,r2
          cjne        a,#3,B_TO_H
          clr         c
          mov         a,pag2
          subb        a,#250d
          ;jnc        lho
          ;mov        setpoint,#0
          ;ret
ho:       mov         setpoint,a
          ret
****************************************
subrutin tampilan setpoin ke 7 segmen
****************************************
etvalue2:
          mov         a,#0f0h
          clr         acc.4
          add         a,pad1      ;pecahan
          mov         dptr,#portB
          movx        @dptr,a
          setb        acc.4
          movx        @dptr,a
          ;
          mov         a,#0f0h
          clr         acc.5
          add         a,pad2      ;satuan
```

```
        mov         dptr,#portB
        movx        @dptr,a
        setb        acc.5
        movx        @dptr,a
        ;
        mov         a,#0f0h
        clr         acc.6
        add         a,pad3      ;puluhan
        mov         dptr,#portB
        movx        @dptr,a
        setb        acc.6
        movx        @dptr,a
        ret

*************************************
abel Fuzifikasi Error dan Change Error
*************************************
encari fungsi keanggotaan negatif
en:     db          255         ;0
        db          255
        db          255
        db          255
        db          255
        db          255
        db          255
        db          255
        db          255
        db          255         ;10
        db          255
        db          255
        db          255
        db          255
        db          255
        db          255
        db          255
        db          255
        db          255         ;20
        db          255
        db          255
        db          255
        db          255
        db          255
        db          255
        db          255
        db          255
        db          255         ;30
        db          255
        db          255
        db          255
        db          255
        db          255
        db          255
        db          255
        db          255
        db          255         ;40
        db          255
        db          255
        db          255
        db          255
        db          255
        db          255
        db          255
```

```
db          255
db          255
db          255         ;50
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255         ;60
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255         ;70
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255         ;80
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          249
db          242
db          236         ;90
db          230
db          223
db          217
db          210
db          204
db          198
db          191
db          185
db          179
db          172         ;100
db          166
db          159
db          153
db          147
db          140
db          134
db          128
db          121
db          115
db          108         ;110
db          102
db          96
db          89
```

```
db      83
db      77
db      70
db      64
db      57
db      51
db      45          ;120
db      38
db      32
db      26
db      19
db      13
db      6
db      0
db      0
db      0
db      0           ;130
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
```

```
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
```

```
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0


mencari fungsi keanggotaan error nol
fez:    db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
```

```
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          6
db          13
db          19
db          26
db          32
db          38
db          45
db          51
db          57
db          64
db          70
db          77
db          83
db          89
db          96
db          102
db          108
db          115
db          121
db          128
db          134
db          140
db          147
db          153
db          159
db          166
db          172
db          179
db          185
db          191
```

```
db          198
db          204
db          210
db          217
db          223
db          230
db          236
db          242
db          249
db          255
db          249
db          242
db          236
db          230
db          223
db          217
db          210
db          204
db          198
db          191
db          185
db          179
db          172
db          166
db          159
db          153
db          147
db          140
db          134
db          128
db          121
db          115
db          108
db          102
db          96
db          89
db          83
db          77
db          70
db          64
db          57
db          51
db          45
db          38
db          32
db          26
db          19
db          13
db          6
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
```

```
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
db          0
```

```
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0

ıencari fungsi keanggotaan error positif
fep:    db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
        db      0
```

```
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
db      0
```

```
db          0
db          0
db          0
db          0
db          0
db          6
db          13
db          19
db          26
db          32
db          38
db          45
db          51
db          57
db          64
db          70
db          77
db          83
db          89
db          96
db          102
db          108
db          115
db          121
db          128
db          134
db          140
db          147
db          153
db          159
db          166
db          172
db          179
db          185
db          191
db          198
db          204
db          210
db          217
db          223
db          230
db          236
db          242
db          249
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
```

```
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
db          255
```

```
          db          255


ıencari fungsi keanggotaan error negatif
:cen:     db          255
          db          255
          db          255
          db          255
          db          255
          db          255
          db          255
          db          255
          db          255
          db          255
          db          219
          db          182
          db          146
          db          109
          db          73
          db          36
          db          0
          db          0
          db          0
          db          0
          db          0
          db          0
          db          0
          db          0
          db          0
          db          0
          db          0
          db          0
          db          0
          db          0
          db          0


mencari fungsi kanggotaan prubahan error nol
fcez:     db          0
          db          0
          db          0
          db          0
          db          0
          db          0
          db          0
          db          0
          db          0
          db          36
          db          73
          db          109
          db          146
          db          182
          db          219
          db          255
          db          219
          db          182
          db          146
          db          109
          db          73
          db          36
          db          0
          db          0
          db          0
          db          0
          db          0
          db          0
```

```
            db          0
            db          0
            db          0

mencari fungsi kanggotaan error positif
fcep:       db          0
            db          0
            db          0
            db          0
            db          0
            db          0
            db          0
            db          0
            db          0
            db          0
            db          0
            db          0
            db          0
            db          0
            db          0
            db          0
            db          36
            db          73
            db          109
            db          146
            db          182
            db          219
            db          255
            db          255
            db          255
            db          255
            db          255
            db          255
            db          255
            db          255

;*****************************
;tabel bcd to hex
;*****************************
3toH:       db          64h
            db          0ah
            db          01h

;*** Tabel waktu tunda
Isi_TH0:
ib          0E5h                    ;1
ib          0E6h        ;5h         ;2
ib          0E6h
ib          0E6h
ib          0E6h
ib          0E6h
ib          0E6h
ib          0E6h
ib          0E6h
ib          0E6h
ib          0E6h
ib          0E6h
ib          0E6h
ib          0E6h
ib          0E6h
ib          0E6h
db          0E6h
db          0E6h
db          0E7h
db          0E7h
db          0E7h
```

```
b               0E7h
b               0E7h
b               0E7h
b               0E7h
b               0E7h
b               0E7h
b               0E7h
b               0E7h
b               0E7h
b               0E7h
b               0E7h
b               0E7h
b               0E8h
b               0E8h
b               0E8h
b               0E8h
b               0E8h
b               0E8h
b               0E8h
b               0E8h
b               0E8h
b               0E8h
b               0E8h
b               0E8h
b               0E8h
b               0E8h
b               0E8h
b               0E8h
b               0E9h
b               0E9h
b               0E9h
b               0E9h
b               0E9h
b               0E9h
b               0E9h
b               0E9h
b               0E9h
b               0E9h
b               0E9h
b               0E9h
b               0E9h
b               0E9h
b               0EAh
b               0EAh
b               0EAh
b               0EAh
b               0EAh
b               0EAh
b               0EAh
b               0EAh
b               0EAh
b               0EAh
b               0EAh
b               0EAh
b               0EAh
b               0EAh
b               0EAh
b               0EAh
b               0EBh
b               0EBh
b               0EBh
b               0EBh
b               0EBh
b               0EBh
b               0EBh
b               0EBh
```

0EBh
0EBh
0EBh
0EBh
0EBh
0EBh
0ECh
0ECh
0ECh
0ECh
0ECh
0ECh
0ECh
0ECh
0ECh
0ECh
0ECh
0ECh
0ECh
0ECh
0ECh
0ECh
0EDh
0EDh
0EDh
0EDh
0EDh
0EDh
0EDh
0EDh
0EDh
0EDh
0EDh
0EDh
0EDh
0EDh
0EDh
0EDh
0EDh
0EDh
0EDh
0EEh
0EEh
0EEh
0EEh
0EEh
0EEh
0EEh
0EEh
0EEh
0EEh
0EEh
0EEh
0EEh
0EEh
0EEh
0EEh
0EEh
0EFh
0EFh
0EFh
0EFh
0EFh
0EFh
0EFh
0EFh
0EFh
0EFh
0EFh
0EFh
0EFh
0EFh
0EFh
0EFh

0EFh
0F0h
0F0h
0F0h
0F0h
0F0h
0F0h
0F0h
0F0h
0F0h
0F0h
0F0h
0F0h
0F0h
0F0h
0F0h
0F0h
0F1h
0F1h
0F1h
0F1h
0F1h
0F1h
0F1h
0F1h
0F1h
0F1h
0F1h
0F1h
0F1h
0F1h
0F1h
0F1h
0F2h
0F2h
0F2h
0F2h
0F2h
0F2h
0F2h
0F2h
0F2h
0F2h
0F2h
0F2h
0F2h
0F2h
0F3h
0F3h
0F3h
0F3h
0F3h
0F3h
0F3h
0F3h
0F3h
0F3h
0F3h
0F3h
0F3h
0F3h
0F3h
0F3h
0F3h
0F4h
0F4h
0F4h
0F4h
0F4h

```
        0F4h
        0F4h
        0F4h
        0F4h
        0F4h
        0F4h
        0F4h
        0F4h
        0F4h
        0F4h
        0F5h
        0F5h
        0F5h
        0F5h
        0F5h
        0F5h
        0F5h
        0F5h
        0F5h
        0F5h
        0F5h
        0F5h
        0F5h
        0F5h
        0F5h
        0F5h
        0F6h
        0F6h
        0F6h
        0F6h
        0f6h
        0f6h
        0f6h
        0f6h
        0f6h
        0f6h
        0f6h
        0f6h
        0f6h
        0f6h
        0f6h
        0f6h

i_TL0:
        048h        ;0
        014h        ;59h,6Ah,7Bh,8Ch,9Dh,AEh,BFh,D0h,E1h,F2h,03h;1
        025h        ;12
        036h
        047h
        058h
        069h
        07Ah
        08Bh
        09Ch
        0ADh
        0BEh
        0CFh
        0E0h
        0F2h
        003h
        014h
        025h
        036h
        047h
        058h
        069h
        07Ah
        08Bh
```

```
            09Ch
            0ADh
            0BEh
            0CFh
            0E0h
            0F1h
            002h
            013h
            024h
            035h
            046h
            057h
            068h
            079h
            08Ah
            09Bh
            0ACh
            0BDh
            0CEh
            0DFh
            0F0h
            001h
            012h
            023h
            034h
            046h
            057h
            068h
            079h
            08Ah
            09Bh
            0ACh
            0BDh
            0CEh
            0DFh
            0F0h
            001h
            012h
            023h
            034h
            045h
            056h
            067h
            078h
            089h
            09Ah
            0ABh
            0BCh
            0CDh
            0DEh
            0EFh
            000h
            011h
            022h
            033h
            044h
            055h
            066h
            077h
            088h
            099h
            0ABh
            0BCh
            0CDh
            0DEh
            0EFh
```

```
b       000h
b       011h
b       022h
b       033h
b       044h
b       055h
b       066h
b       077h
b       088h
b       099h
b       0AAh
b       0BBh
b       0CCh
b       0DDh
b       0EEh
b       0FFh
b       010h
b       021h
b       032h
b       043h
b       054h
b       065h
b       076h
b       087h
b       098h
b       0A9h
b       0BAh
b       0CBh
b       0DCh
b       0EDh
b       0FEh
b       010h
b       021h
b       032h
b       043h
b       054h
b       065h
b       076h
b       087h
b       098h
b       0A9h
b       0BAh
b       0CBh
b       0DCh
b       0EDh
b       0FEh
b       00Fh
b       020h
b       031h
b       042h
b       053h
b       064h
b       075h
b       086h
        097h
        0A8h
        0B9h
        0CAh
        0DBh
        0ECh
        0FDh
        00Eh
        01Fh
        130h
        241h
        3?h
```

063h
075h
086h
097h
0A8h
0B9h
0CAh
0DBh
0ECh
0FDh
00Eh
01Fh
030h
041h
052h
063h
074h
085h
096h
0A7h
0B8h
0C9h
0DAh
0EBh
0FCh
00Dh
01Eh
02Fh
040h
051h
062h
073h
084h
095h
0A6h
0B7h
0C9h
0DAh
0EBh
0FCh
00Dh
01Eh
02Fh
040h
051h
062h
073h
084h
095h
0A6h
0B7h
0C8h
0D9h
0EAh
0FBh
00Ch
01Dh
02Eh
03Fh
050h
061h
072h
083h
94h
5h
h