

**IMPLEMENTASI JARINGAN SYARAF TIRUAN BERBASIS
METODE BACKPROPAGATION SEBAGAI PENGENDALI
KECEPATAN MOTOR DC**

TUGAS AKHIR

Diajukan untuk melengkapi salah satu syarat untuk memperoleh gelar sarjana
Teknik pada Jurusan Teknik Elektro Universitas Islam Indonesia
Yogyakarta



oleh :

Nama : Sri Utami Kusumadewi

No. Mahasiswa : 01 524 109

**JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM INDONESIA
YOGYAKARTA**

2006

LEMBAR PENGESAHAN DOSEN PEMBIMBING

IMPLEMENTASI JARINGAN SYARAF TIRUAN

BERBASIS METODE *BACKPROPAGATION* SEBAGAI

PENGENDALI KECEPATAN MOTOR DC

TUGAS AKHIR

Oleh :

Nama : Sri Utami Kusumadewi

No. Mahasiswa : 01524109



Yogyakarta, Desember 2006

Pembimbing I,

Pembimbing II,

(Wahyudi Budi Pramono, ST.)

(Dwi Ana Ratna Wati, ST.)

LEMBAR PENGESAHAN PENGUJI
IMPLEMENTASI JARINGAN SYARAF TIRUAN
BERBASIS METODE *BACKPROPAGATION* SEBAGAI
PENGENDALI KECEPATAN MOTOR DC

TUGAS AKHIR

oleh:

Nama : Sri Utami Kusumadewi

No. Mahasiswa : 01 524 109

Telah dipertahankan di Depan Sidang Penguji sebagai
Salah Satu Syarat untuk Memperoleh Gelar Sarjana
Teknik Elektro Fakultas Teknologi Industri
Universitas Islam Indonesia
Yogyakarta, 29 Desember 2006

Tim Penguji,

IR.Hj Budi Astuti, MT

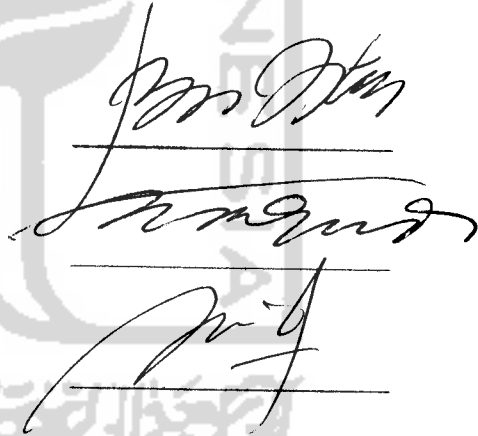
Ketua

Wahyudi Budi Pramono, ST

Anggota I

Dwi Ana Ratna Wati, ST

Anggota II



Mengetahui,

Ketua Jurusan Teknik Elektro
Fakultas Teknologi Industri
Universitas Islam Indonesia



(Wahyudi Budi Pramono, ST, M.Sc)

HALAMAN PERSEMBAHAN

Kinorsembahkan Tugas Akhir ini
Untuk :

Bapakku Mukardi dan ibuku Hersutami tercinta,

"Terima kasih selalu memberikan yang terbaik buat Dewi.

Dan tak henti hentinya melimpahi kasih sayang dan doa".

Adeku Dharma,

"Eratkan tangan kita untuk membahagiakan bapak dan ibu
tercinta"

Seluruh keluarga besar dari pihak bapak dan ibu

"Terima kasih atas dukungan dan doa"

Semoga menjadi kenangan yang indah dan
tak terlupakan
Amien Ya Rabbal 'Alami

MOTTO

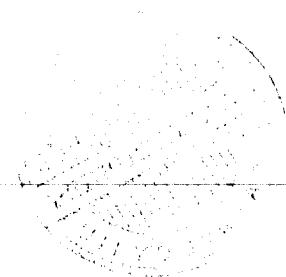
*Teruslah maju karena waktu takkan mundur
(my word)*

*"Bersabarlah menjalani segala hal yang tak terpecahkan dalam hatimu dan
cobalah mencintai pertanyaan-pertanyaan.
Hidupkan senantiasa pertanyaan-pertanyaan itu. Perlahan, tanpa perlu
mencermatinya, mungkin engkau akan sampai pada jawaban yang engkau cari"
(Rainer Maria Rilke 1875 – 1926)*

*"Allah tidak membebani seseorang melebihi kemampuan yang diberikan-Nya,
dan sesudah kesukaran Allah pasti akan memberikan kelapangan."
(Qs. Ath Thalaq : 7)*

*"Sesungguhnya doa adalah ibadah."
(HR, Imam empat)*

KATA PENGANTAR



Assalamu'alaikum Wr. Wb.

Alhamdulillah, segala puji dan syukur ditunjukkan hanya bagi Allah SWT, sang Esa pemilik alam semesta. Semoga kesejahteraan diberikan bagi rosul-Nya, Muhammad SAW, sang mustofa. Atas rahmat dan taufik-Nya penulis dapat menyelesaikan tugas akhir dengan judul **“Implementasi Jaringan Syaraf Tiruan Berbasis Metode *Backpropagation* Sebagai Pengendali Kecepatan Motor DC”** dengan lancar.

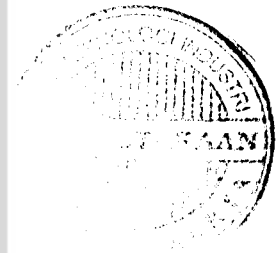
Adapun maksud dan tujuan penyusunan tugas akhir ini adalah untuk melengkapi salah satu syarat dalam menempuh gelar sarjana pada Jurusan Teknik Elektro Universitas Islam Indonesia, Yogyakarta. Disamping itu untuk menambah pengetahuan terhadap ilmu yang telah dipelajari di bangku perkuliahan untuk diterapkan di masyarakat.

Pada kesempatan ini dengan segala rasa syukur dan kerendahan hati penulis sampaikan ucapan terima kasih yang sebesar-besarnya kepada :

1. Kedua orang tua tercinta yang telah memberikan dorongan serta doa restu dalam menggapai cita-cita penulis.
2. Bpk Tito Yuwono, ST. selaku Ketua Jurusan Teknik Elektro Universitas Islam Indonesia Yogyakarta.

ABSTRAK

Motor DC banyak digunakan di industri. Kecepatan motor DC mudah diatur dalam suatu rentang kecepatan yang lebar dengan pengaturan tegangan masukannya. Pengaturan kecepatan motor DC menggunakan sistem manual terdapat kelemahan yang disebabkan *human error* dalam memprediksi tegangan yang masuk sesuai dengan beban pada motor DC. Untuk mengatasi kelemahan tersebut dikembangkan sistem kendali otomatis dengan memanfaatkan perangkat lunak yaitu Jaringan Syaraf Tiruan. Pengendali kecepatan motor DC menggunakan sistem Jaringan Syaraf Tiruan berbasis metode *Backpropagation* dan *interface* (antarmuka) menggunakan program Delphi. Untuk mengatur pengiriman dan penerimaan data dari sensor *optocoupler* ke komputer maupun sebaliknya digunakan sebuah mikrokontroler AT 89C51. Sebagai driver, juga digunakan mikrokontroler AT 89C2051 untuk membangkitkan pulsa – pulsa PWM. Hasil yang diperoleh dari menunjukkan masih terdapat error rata-rata sebesar 14,88 rpm.



DAFTAR ISI

HALAMAN JUDUL	i
LEMBAR PENGESAHAN PEMBIMBING	ii
LEMBAR PENGESAHAN PENGUJI.....	iii
HALAMAN PERSEMBAHAN	iv
MOTTO	v
KATA PENGANTAR.....	vi
ABSTRAKSI	viii
DAFTAR ISI	ix
DAFTAR TABEL	xii
DAFTAR GAMBAR.....	xiii
BAB I. PENDAHULUAN.....	1
1.1. Latar Belakang Masalah	1
1.2. Rumusan Masalah.....	2
1.3. Batasan Masalah	2
1.4. Tujuan Penelitian	2
1.5. Metodologi penelitian.....	3
1.6. Sistematika Penulisan	3
BAB II. DASAR TEORI.....	5
2.1. Jaringan Syaraf Tiruan.....	5
2.1.1. Metode <i>Backpropagation</i>	6
2.1.2. Algoritma <i>Backpropagation</i>	7
2.2. Antar Kanal Serial	9

2.2.1. Komunikasi Serial Port I/O dengan Borland Delphi.....	10
2.2.2. Serial Data Format.....	11
2.3 IDE (Intergrated Development Environment).....	12
2.4. Motor Arus Searah (DC)	13
2.4.1 Prinsip Kerja	14
2.4.2 Karakteristik Motor DC.....	14
2.4.3 Pengaturan Kecepatatan Motor DC.....	16
2.5. Mikrokontroller.....	17
2.5.1. Arsitektur Mikrokontroller AT89C2051.....	18
2.5.2 Arsitektur Mikrokontroller AT89C51.....	21
2.6. Optocoupler sebagai sensor kecepatan.....	24
BAB III. PERANCANGAN SISTEM	25
3.1 Perancangan Perangkat Keras.....	27
3.1.1. Sistem Minimum Mikrokontroller AT89C51 dan AT89C2051	27
3.1.2. <i>Konverter</i> Tingkat RS-232	29
3.1.3. Driver.....	29
3.1.4. Optocoupler.....	30
3.1.5. PC (Personal Computer).....	31
3.2 Perancangan Software	32
3.2.1. Perancangan IDE Delphi	33
3.2.2. Struktur Jaringan Syaraf Tiruan.....	33
3.3.3 Software Mikrokontroller.....	35

3.3. Pergerakan Motor DC.....	37
BAB IV. PENGAMATAN DAN ANALISA.....	40
4.1. Pengujian Keluaran Menggunakan Motor DC.....	41
4.2. Analisa Bentuk PWM.....	45
BAB V. PENUTUP.....	49
5.1. Kesimpulan.....	49
5.2. Saran.....	50
DAFTAR PUSTAKA.....	51
LAMPIRAN	



DAFTAR TABEL

Tabel 2.1 Fungsi port Konektor DB9	11
Tabel 2.2 Pemakaian motor DC berdasarkan karakteristiknya.....	14
Tabel 2.3 Fungsi Port 3 pada Mikrokontroler.....	20
Tabel 4.1. Hubungan <i>Duty cycle</i> dengan Kecepatan Motor DC.....	41
Tabel 4.2. Hasil rata – rata pengambilan data motor DC.....	42



DAFTAR GAMBAR

Gambar 2.1. Arsitektur algoritma <i>Backpropagation</i>	7
Gambar 2.2. Konektor DB9	10
Gambar 2.3. Diagram Struktur Data 8N1	12
Gambar 2.4. <i>Layout Editor</i>	13
Gambar 2.5. Rangkaian Ekuivalen Motor DC.....	14
Gambar 2.6 Karakteristik Motor DC dengan Eksitasi Terpisah	16
Gambar 2.7. Konfigurasi Pin.....	19
Gambar 2.8. Diagram Blok AT89C2051.....	19
Gambar 2.9. Memori Data Internal.....	22
Gambar 2.10. Konfigurasi Mikrokontroler AT89C51.....	22
Gambar 2.11. Blok Diagram Mikrokontroler AT89C51.....	22
Gambar 3.1. Blok diagram Pengendali Motor DC.....	25
Gambar 3.2. Skema Rangkaian Pembangkit Pulsa.....	28
Gambar 3.3. Rangkaian umum Max-232.....	29
Gambar 3.4. Rangkaian <i>Driver</i> Motor DC.....	30
Gambar 3.5 Rangkaian <i>Optocoupler</i>	31
Gambar 3.6 <i>Flowchart</i> dengan Pengendali Kecepatan Motor DC.....	32
Gambar 3.7 Tampilan GUI untuk Motor DC	33
Gambar 3.8. Arsitektur Jaringan Syaraf Tiruan.....	34
Gambar 3.9 <i>Flowchart</i> Diagram Pengaturan Kecepatan Motor DC.....	36

Gambar 3.10. <i>Flowchart</i> Pembangkit PWM.....	37
Gambar 4.1 Grafik perbandingan antara <i>Duty cycle</i> dengan Kecepatan Motor DC.....	40
Gambar 4.2. Grafik hasil perbandingan dengan kecepatan pada tampilan Delphi dengan hasil dari <i>Tachometer</i>	43
Gambar 4.3. Grafik hasil dari mengukur tegangan.....	44
Gambar 4.4 Tampilan Delphi untuk mengetahui perbedaan error	45
Gambar 4.5 Keluaran sinyal mikro pada saat <i>Duty Cycle</i> 10%.....	46
Gambar 4.6 Keluaran sinyal mikro pada saat <i>Duty Cycle</i> 40%.....	46
Gambar 4.7 Keluaran sinyal mikro pada saat <i>Duty Cycle</i> 70%.....	46
Gambar 4.8 Keluaran sinyal MOSFET pada saat <i>Duty Cycle</i> 10%.....	47
Gambar 4.9 Keluaran sinyal MOSFET pada saat <i>Duty Cycle</i> 50%.....	47
Gambar 5.0 Keluaran sinyal MOSFET pada saat <i>Duty Cycle</i> 80%	47

BAB I

PENDAHULUAN



1.1 Latar Belakang

Komputer telah menjadi kebutuhan pokok dalam dunia moderen saat ini, yang digunakan untuk membantu kebutuhan manusia untuk menyelesaikan pekerjaan sehari – hari. Pada mulanya komputer merupakan alat yang ukurannya besar dan mahal, sehingga orang-orang sangat sulit memiliki komputer. Namun perkembangan nyata pada saat ini membuat komputer menjadi alat yang sangat dibutuhkan sebagai alat kontrol atau pengendali industri, rumah tangga dan bidang pendidikan.

Dengan komputer banyak program – program yang dapat dibuat untuk membantu manusia dalam menyelesaikan tugasnya diantaranya adalah Borland Delphi. Delphi merupakan bahasa pemrograman yang banyak digunakan membuat aplikasi untuk keperluan teknik, kedokteran, ekonomi, militer, permainan anak-anak dan lain sebagainya. Bahasa pemrograman yang digunakan adalah bahasa pemrograman pascal atau kemudian disebut dengan bahasa pemrograman Delphi.

Motor DC banyak dipakai dalam dunia industri maupun rumah tangga. Motor DC yang beredar di perusahaan sebenarnya sudah menggunakan bahasa logika sederhana yang dapat digerakkan secara manual seperti saklar *on-off* oleh operator (manusia), sebagian sudah ada yang menggunakan *mikrokontroller*, logika *fuzzy*, Jaringan Syaraf Tiruan (JST) maupun kendali-kendali lainya sebagai pengendali motor DC tersebut. Barang –barang yang dihasilkan oleh motor DC yang digunakan di dunia industri akan

lebih menghasilkan produk yang bagus dan memiliki tingkat presisi yang tinggi apabila kesalahan dari faktor manusia dapat diperkecil.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan diatas, maka dapat diambil suatu rumusan masalah sebagai berikut: “Bagaimana mengendalikan kecepatan motor DC dengan menggunakan Jaringan Syaraf Tiruan Berbasis Metode *Backpropagation* ?“

1.3 Batasan Masalah

Dengan adanya batasan masalah, penulisan dapat lebih menyederhanakan dan mengarahkan penelitian dan pembuatan sistem agar tidak menyimpang dari apa yang diteliti dan dikembangkan, Batasan-batasannya adalah sebagai berikut :

1. Penelitian difokuskan pada internal I/O dengan Delphi untuk menghubungkan komputer dengan *peripheral devices*, khususnya untuk untuk menampilkan sedikit error yang dihasilkan oleh kecepatan motor
2. Sensor kecepatan yang digunakan adalah optocoupler
3. Pembuatan sistem ini menggunakan perangkat lunak delphi 7.0
4. Motor DC 0,5 Hp, eksitasi terpisah
5. Pembelajaran JST dengan Matlab.

1.4 Tujuan dan Manfaat

Tujuan dalam penyusunan laporan adalah untuk mempelajari dan memahami kinerja pengendali kecepatan Motor DC dengan menggunakan Jaringan Syaraf Tiruan yang berbasis metode *Backpropagation*.

1.5 Metodologi Penelitian

1. Pengumpulan Data

Data diperoleh dari studi pustaka berupa buku, artikel, makalah dan tutorial yang tersedia pada *website* di internet.

2. Studi Pustaka

Kajian terhadap hasil penelitian yang sudah ada, terkait dengan bidang pengendali motor DC dengan JST.

3. Perancangan dan Pembuatan

Setelah semua data terkumpul, maka dilakukan perancangan dan pembuatan alat, Dimana perancangan terdiri dari 2 hal yaitu :

1. Software

Membuat untuk memasukkan data set point kecepatan motor DC dengan menggunakan Program Delphi.

2. Hardware

Membuat motor driver.

4. Pengujian dan Analisis

Pengujian dilakukan setelah alat sudah jadi, dan dilanjutkan dengan analisis untuk masing – masing percobaan yang berkaitan dengan tugas akhir ini.

1.6 Sistematika Penulisan

Sistematika penulisan tugas akhir ini terdiri dari 5 (lima) bab bagian isi laporan, dengan penjelasan bab sebagai berikut :

BAB II

LANDASAN TEORI

2.1 Jaringan Syaraf Tiruan

Jaringan syaraf tiruan (JST) adalah sistem pemroses informasi yang memiliki karakteristik mirip dengan jaringan syaraf biologi, yang merupakan salah satu repretasi buatan otak manusia yang selalu mencoba untuk mensimulasikan proses pembelajaran pada otak manusia tersebut. Istilah buatan digunakan karena jaringan syaraf tiruan diimplementasikan dengan menggunakan program komputer yang mampu menyelesaikan sejumlah proses perhitungan selama pembelajaran.

Jaringan syaraf tiruan dibentuk sebagai generalisasi model matematika dari jaringan syaraf biologi, dengan asumsi bahwa :

1. Pemrosesan informasi terjadi pada banyak elemen sederhana (neuron).
2. Sinyal dikirimkan diantara neuron–neuron melalui penghubung-penghubung.
3. Untuk menentukan output, setiap neuron menggunakan fungsi aktivitas (biasanya bukan fungsi linier) yang dikenakan pada jumlahan input yang diterima. Besar output ini selanjutnya akan dibandingkan dengan suatu batas ambang.

Jaringan syaraf tiruan juga ditentukan dengan 3 hal :

- a. Pola hubungan antar neuron (disebut arsitektur jaringan).
- b. Metode untuk menentukan bobot penghubung (disebut metode *training* / *learning* / algoritma

c. Fungsi aktivitas.

JST dikenal juga sebagai model free-estimator, karena dibanding dengan cara perhitungan konvensional, JST tidak memerlukan atau menggunakan suatu model matematis dari permasalahan yang dihadapi.

2.1.1 Metode *Backpropagation*

Metode *Backpropagation* merupakan algoritma pembelajaran yang terawasi dan biasanya digunakan oleh perceptron dengan banyak lapisan untuk mengubah bobot-bobot yang terhubung dengan neuron-neuron yang ada pada lapisan tersembunyi. Metode *backpropagation* banyak diterapkan dalam berbagai bidang, seperti pengenalan pola, diagnosa kedokteran, klasifikasi gambar, menerjemahkan kode, dan bermacam-macam analisa pengenalan pola lainnya. Jaringan ini merupakan salah satu jenis yang mudah dipahami dan konsep belajarnya relatif sederhana, yaitu :

1. Belajar dari kesalahan.
2. Memasukan secara umpan maju (*feed forward*) pola-pola masukan.
3. Menghitung dan propagasi balik kesalahan yang bersangkutan.
4. Mengatur bobot-bobot koneksi.

Algoritma *backpropagation* menggunakan *error output* untuk mengubah nilai bobot-bobotnya dalam arah mundur (*backward*). Untuk mendapatkan *error*, tahap perambatan maju (*forward*) harus dikerjakan terlebih dahulu. Pada saat perambatan maju, neuron-neuron diaktifkan dengan fungsi aktivasi.

lapisan tersembunyi ini kemudian menghitung aktivasinya (z_{in_j}) dan mengirimkan sinyalnya ke tiap-tiap unit lapisan keluaran (z_j). Tiap unit keluaran (y_{in_k}) menghitung aktivasinya (y_k) untuk membentuk respon pada jaringan terhadap pola masukan yang diberikan.

Selama pelatihan, tiap unit keluaran membandingkan perhitungan aktivasinya (y_k) dengan nilai targetnya (t_k) untuk menentukan kesalahan pola tersebut dengan unit itu. Berdasarkan kesalahan ini, faktor δ_k ($k = 1, \dots, m$) dihitung. δ_k digunakan untuk menyebarkan kesalahan pada unit *output* (y_k) ke semua unit pada lapisan sebelumnya (unit-unit tersembunyi yang dihubungkan ke y_k) dan digunakan nantinya untuk mengubah bobot-bobot antara *output* dan lapisan tersembunyi. Dengan cara yang sama, δ_j ($j = 1, \dots, p$) dihitung untuk tiap unit tersembunyi (z_j). Untuk (δ_j) tidak perlu menyebarkan kesalahan kembali ke lapisan *input*, tetapi digunakan nantinya untuk mengubah bobot-bobot antara lapisan tersembunyi dan lapisan *input*-nya.

Setelah seluruh faktor δ telah ditentukan, bobot untuk semua lapisan diatur secara serentak. Pengaturan bobot w_{jk} (dari tiap-tiap unit lapisan tersembunyi ke tiap-tiap unit *output*) didasarkan pada faktor δ_k dan aktivasi z_{in_j} dari unit tersembunyi z_j . Pengaturan bobot v_{ij} (dari vektor unit *input* ke tiap-tiap unit lapisan tersembunyi) didasarkan pada faktor δ_j dan aktivasi unit *input* x_i .

Suatu jangka waktu (*epoch*) adalah satu set putaran vektor-vektor pelatihan. Beberapa *epoch* diperlukan untuk pelatihan sebuah jaringan syaraf tiruan *backpropagation*. Dalam algoritma ini dilakukan perbaikan bobot setelah masing-masing pola pelatihan disajikan. Setelah pelatihan selesai bobot-bobot yang telah diperbaiki disimpan.

2.2 Antarmuka Kanal Serial

Antarmuka kanal serial lebih kompleks atau sulit dibandingkan dengan antarmuka melalui kanal paralel, hal ini disebabkan karena :

1. Dari segi perangkat keras

Adanya proses konversi data paralel menjadi serial atau sebaliknya menggunakan piranti tambahan disebut UART (*Universal Asynchronous Receiver/Transmitter*).

2. Dari segi perangkat lunak

Lebih banyak register yang digunakan akan terlibat.

Namun di sisi lain antarmuka kanal serial menawarkan beberapa kelebihan dibandingkan secara paralel, antara lain :

1. Kabel untuk komunikasi serial bisa lebih panjang dibandingkan dengan paralel.

Data –data dalam komunikasi serial dikirimkan untuk logika '1' sebagai tegangan -3 s/d -25 volt dan untuk logika '0' sebagai tegangan +3 s/d +25 volt, dengan demikian tegangan dalam komunikasi serial memiliki ayunan tegangan maksimal 50 volt, sedangkan pada komunikasi paralel hanya 5 volt. Hal ini menyebabkan gangguan pada kabel – kabel panjang lebih mudah diatasi dibandingkan paralel.

2. Jumlah kabel serial lebih sedikit

Dengan kabel serial dapat menghubungkan dua perangkat komputer yang berjauhan dengan hanya 3 kabel untuk konfigurasi null modem, yaitu TXD (saluran kirim), RXD (saluran terima) dan *Ground*.

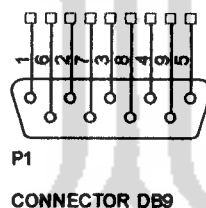
3. Untuk teknologi *Embedded System* , banyak mikrokontroler yang dilengkapi dengan komunikasi serial.

2.2.1 Komunikasi Serial Port I/O dengan Borland Delphi

Komunikasi serial adalah *low-level protocol* untuk komunikasi antara dua atau lebih *device*. Antarmuka port serial dengan Delphi menyediakan akses langsung ke *peripheral devices* seperti modem, printer dan instrument lainnya yang dihubungkan dengan ke port serial komputer.

Pada zaman sekarang sudah ada banyak pilihan antarmuka port serial antara lain RS-232, RS-422, dan RS-485. yang didukung dengan objek port serial dengan Delphi. Sekarang ini yang banyak dipakai adalah antarmuka standar yang digunakan untuk menghubungkan komputer dengan *peripheral devices* adalah RS-232.

Terdapat dua macam konektor port serial yaitu konektor DB-25 dan DB-9 keduanya dalam bentuk jantan (*male*) pada personal komputer.



Gambar 2.2 Konektor DB9

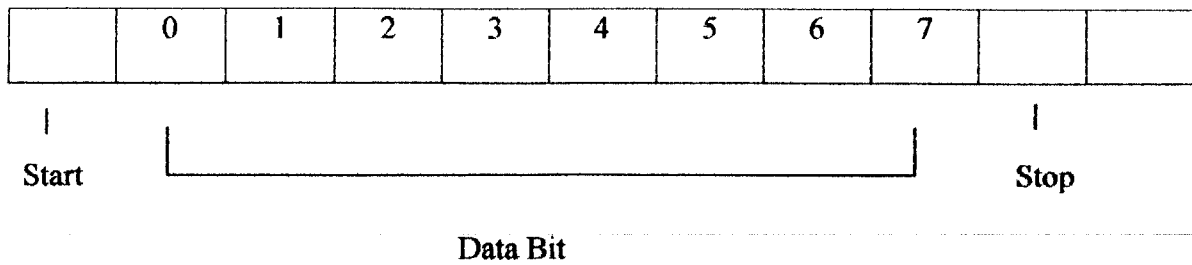
Tabel 2.1 fungsi port konektor DB9

Pin	Signal	Pin	Signal
1	Data Carrier Detect	6	Data Set Ready
2	Received Data	7	Request to Sent
3	Transmitted Data	8	Clear to Send
4	Data Terminal Ready	9	Ring Indicator
5	Signal Ground		

Antarmuka digital I/O sering digunakan pada sistem pengolahan data PC untuk mengontrol proses-proses, membangkitkan pola-pola pengujian dan untuk berkomunikasi dengan perangkat lain. Pada tiap-tiap kasus, parameter-parameter yang penting mencakup jumlah jalur digital yang tersedia, laju pemasukan dan pengeluaran data digital pada jalur-jalur tersebut dan kemampuan penggerakannya. Jika suatu jalur digital digunakan untuk mengontrol suatu kejadian seperti menghidupkan dan mematikan pemanas, motor atau lampu, maka tidak dibutuhkan laju data yang tinggi karena peralatan-peralatan tersebut tidak dapat merespon dengan cepat.

2.2.2 Serial Data Format

Sampai saat ini telah dibicarakan tentang komunikasi serial RS_232 berkaitan dengan komputer, komunikasi RS-232 bersifat asinkron (*asynchronous*), artinya sinyal *clock* tidak dikirimkan bersama dengan data, Masing – masing disinkronkan menggunakan bit start-nya dan clock internal pada masing –masing komputer.



Gambar 2.3 Diagram Struktur Data 8N1

Format untuk data port serial biasanya menggunakan notasi " *number off data bits parity type number off stop bits*". Gambar 2.3 diatas merupakan contoh yang menunjukkan komposisi data RS-232 dalam tingkat yang menggunakan format 8N1 (8 bit data, tidak ada paritas dan 1 bit stop), Jalur RS-232 , saat diam (*idle*) dalam kondisi 'Tertanda' – *Mark state* (logika 1). Suatu transmisi akan dimulai dengan sebuah bit start (logika 0), kemudian masing – masing bit dikirimkan berturutan diawali dengan LSB-nya dan diakhiri dengan bit stop (logika 1)

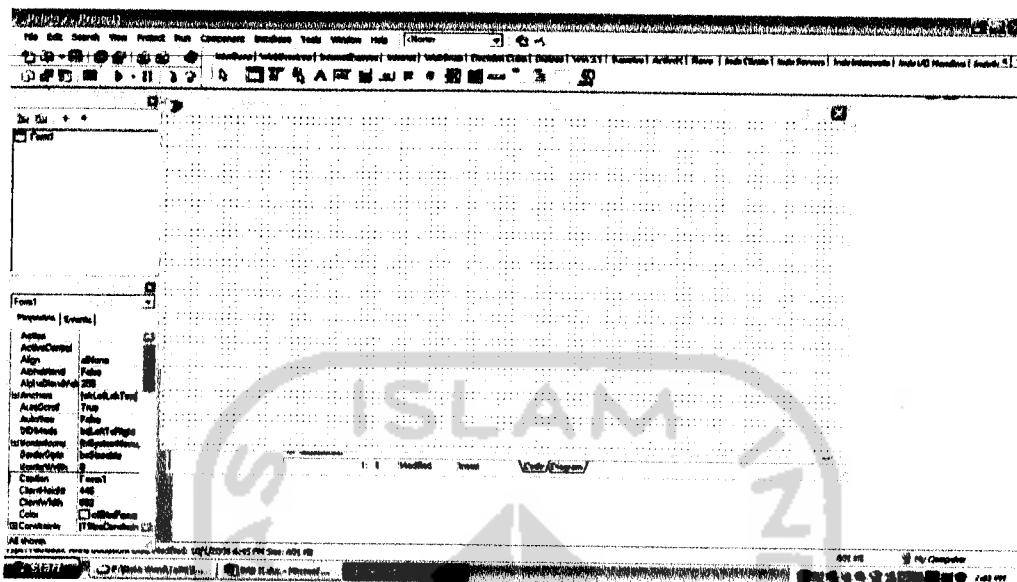
Pada gambar 2.3 juga terlihat bahwa bit berikut setelah bit stop adalah logika 0. ini artinya ada bagian lain yang dikirimkan dan ini merupakan bit startnya. Jika tidak ada data lagi yang dikirimkan maka jalur transmisi akan tetap dalam kondisi diam (logika 1).

2.3 IDE (*Integrated Development Environment*)

IDE pada dasarnya adalah yang menyediakan seluruh sarana yang diperlukan membangun, mencoba, mencari atau melacak kesalahan, serta mendistribusikan aplikasi.

From Designer adalah window kosong tempat merancang antarmuka pemakai (*user interface*) aplikasi tampilannya seperti pada gambar 2.4. pada from inilah

ditampilkan komponen-komponen sehingga aplikasi dapat berinteraksi dengan pemakainya.



Gambar 2.4. *Layout Editor*

2.4 Motor Arus Searah (DC)

Motor DC ialah suatu mesin yang berfungsi mengubah energi listrik arus searah (DC) menjadi energi gerak atau tenaga mekanik sehingga tenaga gerak tersebut menghasilkan putaran pada rotor. Motor dapat dibagi menjadi 2, yaitu motor arus searah (DC) dan motor arus bolak-balik (AC). Motor DC sendiri masih terbagi lagi menjadi motor DC berpenguatan sendiri yaitu motor DC shunt, motor DC kompon (pendek dan panjang), motor DC seri dan motor DC berpenguatan terpisah.

Secara mekanik motor DC sama persis seperti generator DC sehingga satu mesin bisa dioperasikan sebagai motor dan juga generator. Jika pada mesin DC itu menerima sumber tegangan, maka dia akan dioperasikan sebagai motor dan jika mesin DC tersebut rotornya diputar maka dia akan operasi sebagai generator.

2.4.1 Prinsip Kerja

Motor Dc bekerja berdasarkan hukum ampere dan hukum lorentz yaitu:

- Di sekitar penghantar yang dialiri arus listrik akan timbul medan magnet.
- Suatu penghantar yang dialiri arus listrik jika berada pada medan magnet akan mengalami suatu gaya yang disebut gaya lorentz.

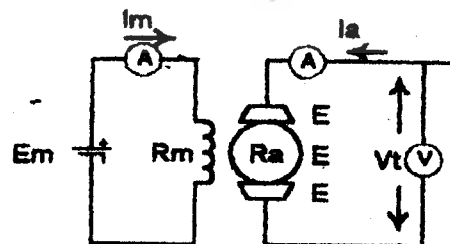
2.4.2 Karakteristik Motor DC

Karakteristik motor DC ditunjukkan pada Tabel 2.2.

Tabel 2.2 Pemakaian motor DC berdasarkan karakteristiknya

No.	Tipe motor	Karakteristik	Pemakaian
1.	Motor DC <i>shunt</i>	<ul style="list-style-type: none"> • Kecepatan mendekati konstan • Torsi start medium • Kecepatan dapat diatur 	<ul style="list-style-type: none"> • Pompa sentrifugal • Mesin perkakas • Kipas dan <i>blower</i> • Mesin bubut
2.	Motor DC seri	<ul style="list-style-type: none"> • Kecepatan bervariasi • Torsi start tinggi • Pengaturan kecepatan yang bervariasi 	<ul style="list-style-type: none"> • Trem listrik • Pesawat pengangkat (<i>crane</i>) • Mobil <i>trolley</i>
3.	Motor DC kompon	<ul style="list-style-type: none"> • Kecepatan bervariasi • Torsi start tinggi • Pengaturan kecepatan yang bervariasi 	<ul style="list-style-type: none"> • Elevator • Mesin potong dan pelubang • Mesin penggiling • Mesin penggilas

Di bawah ini adalah gambar rangkaian ekuivalen motor DC *shunt*, motor DC dengan penguatan paralel beserta persamaan yang berlaku :



Gambar 2.5 Rangkaian Ekuivalen Motor DC

Persamaan rangkaian untuk motor

$$I_m = E_m / R_m \quad (2.1)$$

$$V_t = E_a + I_a R_a + 2 \Delta E \quad (2.2)$$

Motor shunt mempunyai pengaturan kecepatan yang baik dan digolongkan sebagai motor dengan kecepatan konstan walaupun kecepatannya agak berkurang sedikit dengan bertambahnya beban, adapun persamaan kecepatannya adalah :

$$E_a = V_t - I_a R_a, \quad (2.1)$$

$$E_a = C n \Phi, \quad (2.2)$$

$$n = \frac{V_t - I_a R_a}{c \Phi} \quad (2.3)$$

Keterangan :

E_a = GGL Induksi

V_t = Tegangan sumber

n = Kecepatan motor (rpm)

I_a = Arus jangkar motor

R_a = Tahanan kumparan jangkar motor

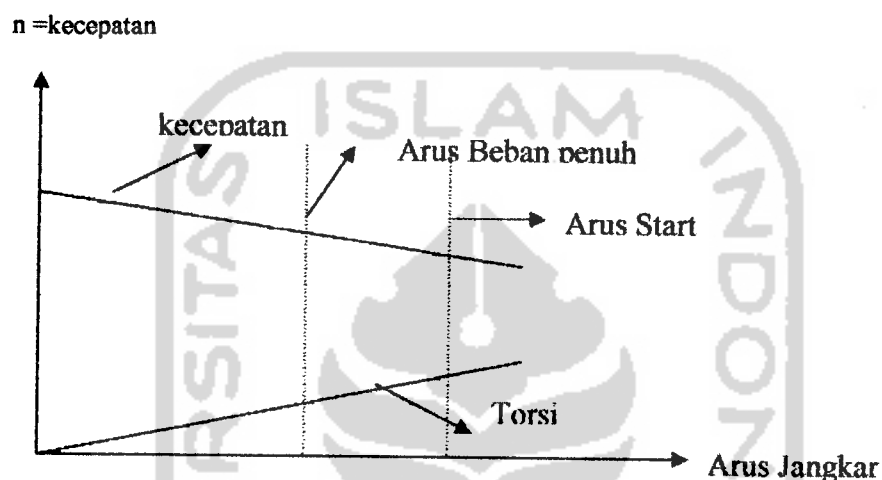
c = Konstanta motor

Φ = Fluks per kutub magnet motor (maxwel)

Dari persamaan 2.3 diketahui bahwa pada motor *shunt*, bertambahnya beban (arus jangkar bertambah) mengakibatkan kecepatan (n) menurun. Pada motor seri, bertambahnya beban (arus) akan menyebabkan bertambahnya harga fluks (Φ), karena fluks pada motor seri merupakan fungsi arus jangkar. Untuk harga arus jangkar sama dengan nol, harga fluks juga mendekati nol sehingga dari persamaan 2.3, diperoleh

harga n menuju tak terhingga. Sedangkan untuk harga I_a yang cukup besar, harga n akan mendekati nol.

Motor DC mempunyai pengaturan kecepatan yang baik. Dan bila beban ditambah maka kecepatan motornya akan melambat. GGL-lawan langsung berkurang karena motor bergantung pada kecepatan dan praktis fluks medan magnet adalah konstan. Karakteristik dari motor dapat dilihat dari gambar dibawah ini :



Gambar 2.6 Karakteristik Motor DC dengan Eksitasi Terpisah

Pada gambar 2.6, Torsi bertambah dalam hubungan yang linier dengan suatu kenaikan dalam arus jangkar, sedangkan kecepatan turun ketika arus jangkar naik. Sehingga harga kecepatan motor dapat mempunyai harga tetap antara maksimum dan minimum, maka motor *shunt* cocok digunakan untuk menggerakkan beban yang membutuhkan kecepatan konstan.

2.4.3 Pengaturan Kecepatan Motor DC

Pengaturan kecepatan memegang peranan penting dalam motor arus searah, karena motor arus searah mempunyai karakteristik torsi-kecepatan yang

menguntungkan dibandingkan dengan motor lainnya. Dari persamaan 2.1 sampai 2.3 diatas, dapat dilihat bahwa kecepatan (n) dapat diatur dengan mengubah besaran ϕ , V_t , dan R_a .

1. Pengaturan kecepatan dengan mengatur medan *shunt* (ϕ), dengan menyisipkan tahanan variabel yang dipasang seri terhadap kumparan medan (motor *shunt*), maka dapat diatur arus medan dan fluksnya. Rugi panas yang ditimbulkan sangat kecil pengaruhnya. Karena besarnya fluks yang dicapai oleh kumparan medan terbatas, kecepatan yang diaturpun akan terbatas.
2. Pengaturan kecepatan dengan mengatur tegangan (V_t), dikenal dengan metode *Ward Leonard*. Menghasilkan suatu pengaturan kecepatan yang sangat halus dan banyak dipakai untuk lift, mesin bubut dan lain-lain. Satu-satunya kerugian dalam sistem ini adalah biaya untuk penambahan generator dan penggerak awal.
3. Pengaturan kecepatan dengan mengatur tahanan (R_a), dengan menyisipkan tahanan variabel terhadap tahanan jangkar. Cara ini jarang dipakai, karena penambahan tahanan seri terhadap tahanan jangkar menimbulkan rugi panas yang cukup besar.

2.5 Mikrokontroller

Mikrokontroller merupakan sistem komputer yang seluruh atau sebagian besar elemennya dikemas dalam satu *chip*, sehingga sering juga disebut dengan *single chip microcomputer*. Mikrokontroller biasanya dikelompokkan dalam satu keluarga, masing – masing mikrokontroller mempunyai spesifikasi tersendiri tapi namun masih kompatibel dalam pemrogramannya. Misalnya keluarga MCS51 yang diproduksi oleh

ATMEL seperti AT89C51 dan AT89C52 dengan kapasitas memori program internal sebesar 8 kByte dan 3 buah timer, AT89C2051 jumlah port 2 dan sebagainya.

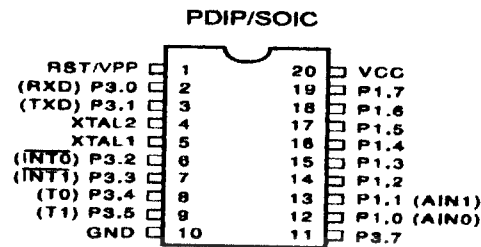
Bahasa pemrograman yang biasa digunakan adalah bahasa assembly, kemudian dikembangkan kompailer untuk bahasa tingkat tinggi. Untuk mikrokontroler MCS-51 bahasa tingkat tinggi yang dikembangkan antara lain, Basic, Pascal, dan C. Bahasa C paling banyak dikembangkan semisal *keil compiler* dari *keil corp* dan SDCC dari *Sandeep Dutta*.

2.5.1 Arsitektur Mikrokontroler AT89C2051

Mikrokontroler jenis AT89C2051 adalah sebuah CMOS mikrokomputer 8 bit bertegangan rendah yang memiliki performa tinggi dengan 2 kilo byte *Flash Programmable Erasable Read Only Memory* (PEROM). Mikrokontroler AT89C2051 menyediakan beberapa fitur standar, antara lain 2 K byte *flash memory*, RAM (*Random Acces Memory*) sebesar 128 byte, 15 jalur input/output, 2timer/counter 16 bit, 5 arsitektur interupsi jenis *two level*, sebuah serial port yang dapat membaca dan mengirim sinyal dua arah (*full duplex*), sebuah analog komparator yang sangat presisi, *oscilator on chip* dan sirkuit *clock*. Sebagai tambahan, mikrokontroler AT89C2051 juga didesain dengan logika statis untuk operasi penurunan frekuensi sampai titik nol (*frequency down to zero operation*) dan mendukung dua macam *power saving* software operasional mode. Pertama adalah mode idle yang melakukan penghentian CPU dengan mengijinkan RAM, timer/counter, serial port, dan sistem interupsi untuk terus melanjutkan operasinya. Kedua adalah mode *power down* yang melakukan penyimpanan isi dari RAM,

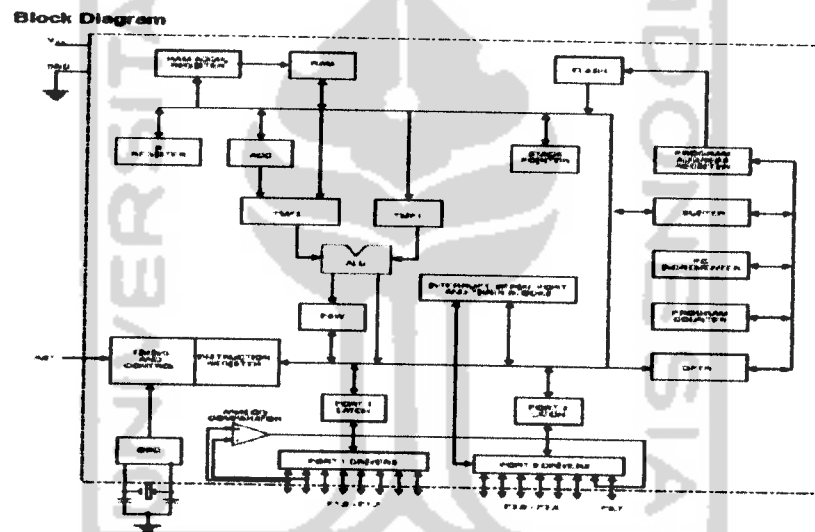
melakukan pembekuan oscilator serta menghentikan semua proses pada fungsi-fungsi chip yang lain sampai hardware reset berikutnya.

Konfigurasi pin



Gambar 2.7 Konfigurasi Pin

Diagram Blok AT89C2051



Gambar 2.8 Diagram Blok AT89C2051

Deskripsi dari pin AT89C2051 yaitu :

1. Vcc berfungsi sebagai suplai tegangan mikrokontroler.
2. GND digunakan untuk *ground*.
3. Port 1 adalah sebuah 8 bit input/output port yang 2 arah (*bidirectional I/O port*) pin port P1.2 sampai P1.7 menyediakan *pull up* secara internal. P1.0 dan P1.1 juga

berfungsi sebagai input positif dan input negatif yang bertanggung jawab pada pembandingan sinyal analog yang ada di dalam chip. Keluaran port 1.0 memuat arus sebesar 20 mA dan dapat digunakan untuk menyalakan LED secara langsung. Jika sebuah program mengakses port pin 1, maka port ini dapat digunakan sebagai port input. Ketika port pin 1.2 sampai 1.7 digunakan sebagai port input dan port-port tersebut diset secara *pulled low*, maka port-port tersebut dapat menghasilkan arus karena adanya internal *pull ups* tadi. Port 1 juga dapat menerima kode/data saat memori *flash* dalam kondisi diprogram atau saat proses verifikasi dilakukan.

4. Port 3 pin P3.0 sampai P3.5 adalah enam input/output pin yang dapat menerima kode/data secara dua arah (*bidirectional I/O port*) yang mempunyai fasilitas internal *pull ups*. P3.6 adalah sebuah *hardware* yang digunakan sebagai input dan output dari komparator on chip, tetapi pin tersebut tidak dapat diakses sebagai port input/output standar. Port 3 dapat mengeluarkan arus sebesar 20 mA. Port 3 juga menyediakan fungsi dari fitur spesial yang bervariasi dari mikrokontroler AT89C2051, antara lain :

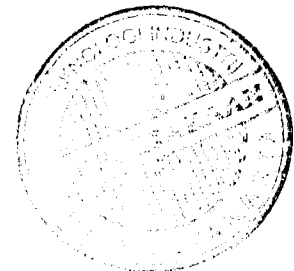
Tabel 2.3 fungsi port 3 pada mikrokontroller

Port Pin	Alternate function
P3.0	RXD (serial input port)
P3.1	TXD (serial output port)
P3.2	INT0 (external interrupt 0)
P3.3	INT1 (external interrupt 1)
P3.4	T0 (timer 0 external input)
P3.5	T1 (timer 1 external input)

5. RST berfungsi sebagai kaki untuk input sinyal reset. Semua input/output (I/O) akan kembali pada posisi nol (reset) secepatnya ketika kaki reset (RST) tersebut berlogika tinggi/*high condition*. Menahan pin RST untuk dua *cycles machine* ketika suatu oscilator sedang bekerja akan mengakibatkan resetnya semua sistem *device* yang ada ke dalam *zero position*.
6. XTAL 1 sebagai input kepada inverting amplifier oscilator dan memberi input kepada internal *clock* operating sirkuit.
7. XTAL 2 sebagai kaki output dari rangkaian inverting amplifier oscilator.

2.5.2 Arsitektur Mikrokontoller AT89C51

Mikrokontroller AT89C51 terdapat 4 port, yang mana setiap portnya memiliki masing-masing 8 bit yang mempunyai fungsi-fungsi tertentu. Mikrokontroller AT89C51 juga memiliki ruang alamat memori data dan program yang terpisah. Pemisahan memori program dan data tersebut membolehkan memori data diakses dengan alamat 8-bit, sehingga dapat dengan cepat dan mudah disimpan dan dimanipulasi oleh CPU 8-bit. Memori RAM AT89C51 mempunyai 256 byte, 128 byte ditempati secara paralel oleh *Special Function Register*(SFR), ini berarti bahwa 128 byte atas mempunyai alamat yang sama dengan SFR, walaupun secara fisik terpisah.



Fungsi-fungsi pin mikrokontroller AT89C51

a. Port 0 (P0.0-P0.7)

Port ini adalah port I/O *bi-directional*. Port ini jika diberikan logika 1 dalam kondisi mengangap akan menjadi input *hight-impendace*, port ini juga menjadi bus alamat dan jalur data untuk byte rendah jika ada memori luar yang digunakan.

b. Port 1 (P1.0-P1.7)

Port ini adalah port I/O *bi-directional* dengan internal *pull-up*. Dua kaki port 1 memiliki ragam fungsi, yang mempunyai fungsi khusus :

T2(P1.0) : Masukan eksternal pencacah 2

T2EX(P1.1) : Pemicu reload pewaktu 2

c. Port 2 (P2.0-P2.7)

Port ini adalah port I/O *bi-directional* dengan internal *pull-up*. Port juga menjadi bus jalur alamat dan jalur data byte tinggi jika ada memori luar yang digunakan (pengalamatan 16 bit).

d. Port 3 (P3.0-P3.7)

Port ini adalah port I/O *bi-directional* dengan internal *pull-up*. Potr 3 dapat digunakan sebagai fungsi khusus yaitu :

RxD (P3.0) : Serial Input Port

TxD (P3.1) : Serial Output Port

INT0 (P3.2) : Eksternal Interrupt

INT1 (P3.2) : Eksternal Interrupt

T0 (P3.4) : Timer 0 ekstenal input

- T1 (P3.5) : Timer 0 eksternal input
 WR (P3.6) : Eksternal data memory *write strobe*
 RD (P3.7) : Eksternal data memory *read strobe*

2.6 Optocoupler (sensor kecepatan)

Optocoupler merupakan komponen yang mampu mengubah efek cahaya menjadi sinyal listrik, selanjutnya komponen yang mampu mengubah efek fisika seperti panas, cahaya, getaran mekanik dan efek-efek fisika lain menjadi listrik disebut transduser.

Optocoupler termasuk dalam optik transduser, komponen ini terdiri emiter cahaya dan sensor yang diproduksi dalam bentuk paket plastik dan dapat diberi lensa atau filter untuk menaikkan kepekaannya.

Optocoupler benar-benar paket elektronik murni. Jalur cahaya didalamnya, yang biasanya inframerah, sungguh-sungguh tertutup dalam paket, ini menyebabkan terjadinya transfer energi listrik dalam salah satu arah, dari IRED ke foto detector, sambil mempertahankan isolasi listrik diantara kedua sirkuit

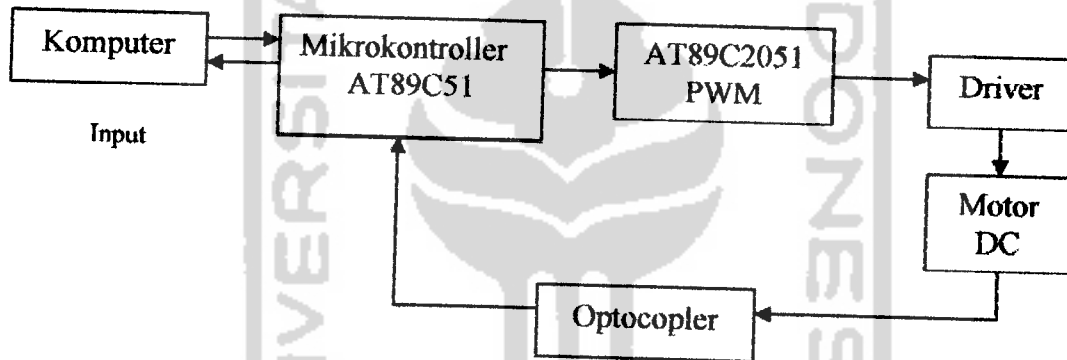


Gambar 2.12 *Optocoupler*

BAB III

PERANCANGAN SISTEM

Dalam bab III akan dibahas tentang perancangan sistem yang didalamnya terdapat perancangan rangkaian elektronik, dan sistem pengendali motor DC 150 volt dengan komunikasi serial berbasis Delphi berdasarkan teori – teori yang dibahas pada bab sebelumnya. Untuk lebih memudahkan pemahaman cara kerja dari sistem yang akan dibuat, maka gambar 3.1 memperlihatkan blok diagram dari perancangan sistem.



Gambar 3.1. Blok diagram pengendali kecepatan motor DC

Pada gambar blok diatas pada masing-masing bagian mempunyai fungsi-fungsi yang berlainan tetapi saling berkaitan. Adapun fungsi-fungsi dari bagian – bagian tersebut adalah :

a. Input

Untuk memasukan nilai kecepatan pada set point, maka bagian yang digunakan sebagai input masukan adalah dengan *scrollbar*. Nilai input tersebut dimasukan melalui tools IDE Delphi dengan batasan nilai input maksimum adalah 1750.

b. PC (*Personal Computer*) dengan GUI Delphi

Blok ini berfungsi sebagai *interface* (antarmuka) antara *user* dengan komputer/PC sebagai pengendali. Disini semua proses komputasi dieksekusi, mulai dari masukan yang berupa kecepatan (rpm) yang akan menghasilkan kecepatan dari optocoupler dan keluaran grafik yang telah diuji dengan jaringan syaraf tiruan dengan metode *Backpropagation*. Setelah semua data yang diperlukan diproses, maka komputer/PC akan mengirimkan data dan menerima data dari sistem minimum mikrokontroler AT89C51 pengiriman dan penerimaan data ini dilakukan dengan komunikasi serial.

c. Mikrokontroler AT89C51 dan Mikrokontroler AT89C2051

Pada blok ini Mikrokontroler AT89C51 adalah sebagai pengatur kecepatan motor DC dan juga sebagai (*interface*) antarmuka dengan komputer. Mikrokontroler AT89C2051 adalah sebagai pembangkit PWM..

d. Driver

Tegangan keluaran dari mikrokontroler sebesar 0-5 volt, sedangkan motor yang digunakan adalah motor DC 150 volt, sehingga memerlukan rangkaian motor driver untuk menaikkan tegangan keluaran dari mikrokontroler sebesar 5 volt menjadi 150 volt.

e. Motor DC

Motor DC ini akan bergerak sesuai dengan lebar pulsa yang diberikan sinyal kontrol. Sehingga besarnya kecepatan (Rpm) motor DC tergantung pada sinyal kontrol yang dikeluarkan oleh mikrokontroler, maka besar kecepatan (Rpm) motor

DC dapat sesuai dengan keinginan pemakai sesuai dengan batas (*range*) yang telah ditentukan.

f. Optocoupler

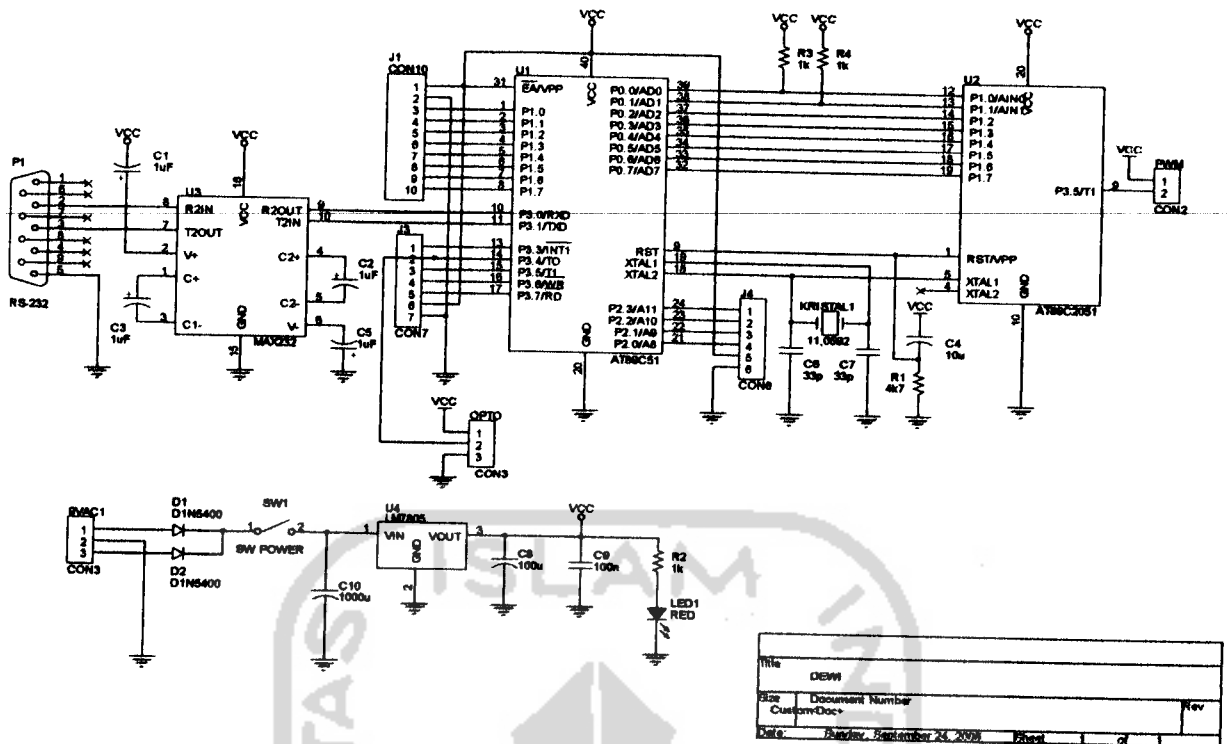
Optocoupler merupakan sensor kecepatan yang berfungsi untuk mengkonversi putaran ke frekuensi dalam bentuk pulsa untuk menghitung kecepatan motor DC.

3.1 Perancangan Perangkat Keras

Pada penelitian ini perangkat keras yang dipergunakan berupa sebuah motor DC 150 V 3, 2 A, yang didukung dengan perlengkapan-perengkapan seperti : mikrokontroler AT89C51, AT89C2051, Optocoupler dan Motor driver.

3.1.1 Sistem Minimum Mikrokontroler AT89C51 dan AT89C2051

Sistem minimum mikrokontroler AT89C51 digunakan sebagai *interface* antar PC dan motor DC atau lebih tepatnya sebagai pengatur kecepatan, pengirim dan penerima kecepatan motor DC, sedangkan AT89C2051 adalah sebagai pembangkit PWM (*Pulse Width Modulation*) dan untuk kebutuhan *download* program dibutuhkan program yang digunakan adalah program Isp-30. Di bawah ini adalah skema rangkaian pembangkit pulsa.



Gambar 3.2 Skema Rangkaian Pembangkit Pulsa

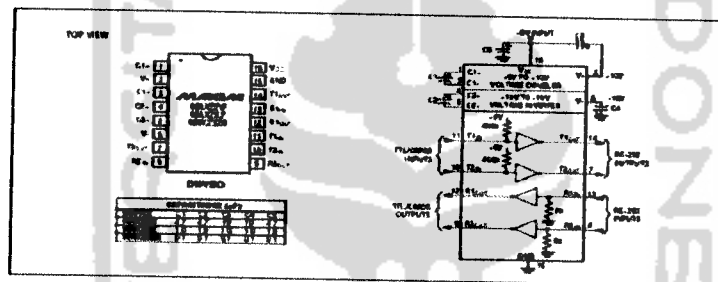
Rangkaian sistem pengatur kecepatan dan pembangkit PWM ini memakai 2 mikrokontroller yaitu AT89C51 dan AT89C2051. Kristal yang dipakai adalah 11.0592 Mhz, sehingga akan memberikan clock sebesar 0,9216 Mhz atau periode sebesar 1,0851s. Penggunaan kristal berfrekuensi 11,0592 Mhz ini adalah untuk menghindari kesalahan yang terjadi pada saat menggunakan baudrate yang tinggi, dan pada rangkaian *oscillator* ini digunakan dua buah kapasitor 33pF.

Mikrokontroller direset pada transisi tegangan rendah ke tegangan tinggi, oleh karena itu pada pin RST dipasang kapasitor yang terhubung ke VCC dan resistor ke ground yang akan menjaga RST bernilai 1 saat pengisian kapasitor dan bernilai nol pada saat kapasitor penuh. Pada saat sumber tegangan diaktifkan kapasitor terhubung dengan singkat sehingga arus mengalir dari VCC langsung ke kaki RST sehingga reset berlogika 1, kemudian kapasitor terisi hingga tegangan pada

kapasitor sama dengan VCC, Pada saat itu kapasitor terisi penuh. Dengan demikian tegangan reset akan turun menjadi 0 sehingga kaki RST berlogika 0.

3.1.2 Konverter Tingkat RS-232

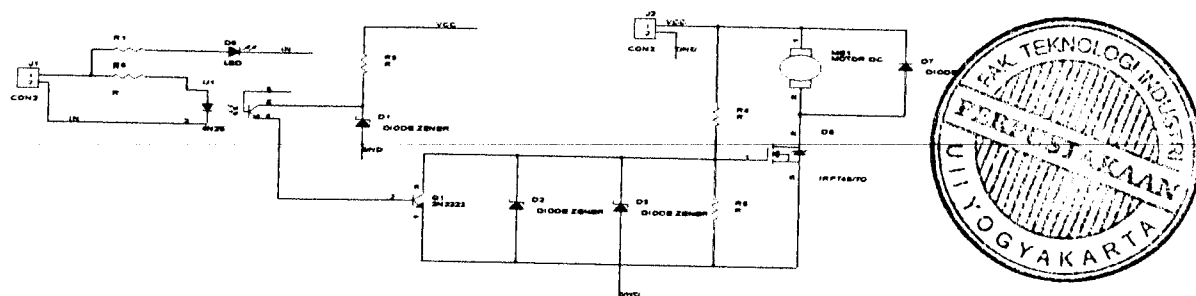
Max -232 adalah sebuah IC yang termasuk dalam *level converter* RS-232, yang memiliki sebuah Charge Pump yang dapat menghasilkan tegangan +10 volt dan -10 volt dari catu daya tunggal 5 volt. Max -232 juga memiliki dua penerima dan dua pengirim pada kemasan yang sama, sehingga tidak memerlukan dua IC dalam proses pengiriman dan penerimaan data, sehingga mikrokontroler dapat menerima data yang dikirimkan PC melalui saluran serial



Gambar 3.3 Rangkaian Umum Max -232

3.1.3 Driver

Rangkaian RS-232 menggunakan catu daya dari PC yang memiliki tegangan stabil. Untuk rangkaian RS-232 dan mikrokontroler membutuhkan tegangan 5 volt DC, sedangkan untuk motor DC membutuhkan tegangan maksimal sampai 150 volt, oleh sebab itu untuk membangkitkan tegangan dari 5 Volt menjadi 150 Volt dibutuhkan driver, dengan input pulsa dari mikrokontroler sehingga dengan tegangan maksimal akan mendapatkan kecepatan motor yang maksimal juga.



Gambar 3.4 Rangkaian Driver Motor DC

Tegangan suplay yang digunakan adalah tegangan AC 100 Volt, yang telah disearahkan. Prinsip kerja MOSFET sama dengan transistor atau *relay*. Bila MOSFET mendapat tegangan pada gate-nya maka akan menyala. Dan pada driver maka diperoleh tegangan yang masuk ke MOSFET adalah 19,5 Volt. Dengan adanya dioda zener 15 Volt maka dibatasi untuk tegangan masukan MOSFET adalah tidak lebih dari 15 Volt.

Rangkaian diatas juga menggunakan dioda 6 Amp yang dipasang secara paralel dengan motor DC. Ini dilakukan agar tidak terjadi arus balik dari motor DC yang nantinya akan merusak MOSFET karena adanya induktansi didalam motor DC.

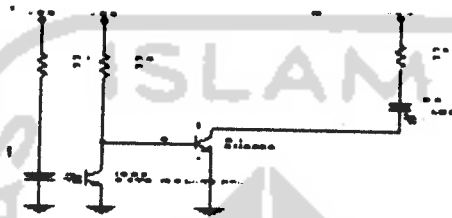
3.1.4 Sensor Kecepatan (Optocoupler)

Motor DC yang digunakan adalah motor DC 150 Volt, motor ini dikopel dengan piringan plastik yang diberi lubang sebanyak 30 lubang. Tiap-tiap lubang akan dibaca oleh *optocoupler* berupa pulsa-pulsa. *Optocoupler* berfungsi untuk mengkonversi putaran ke frekuensi dalam bentuk pulsa untuk menghitung kecepatan putaran motor

DC yang kemudian menjadi masukan ke mikrokontroler. Karena piringannya mempunyai 30 lubang, maka untuk setiap putaran motor akan membangkitkan pulsa sebanyak 30. Time Sampling pembacaan kecepatan motor DC selama 200 ms, jadi perhitungannya kecepatan motor DC

$$(RPM) = \text{nilai timer} * 300 / 30. \quad (3.1)$$

Nilai timer = jumlah gelombang yang masuk selama 200ms.



Gambar 3.5 Rangkaian Optocoupler

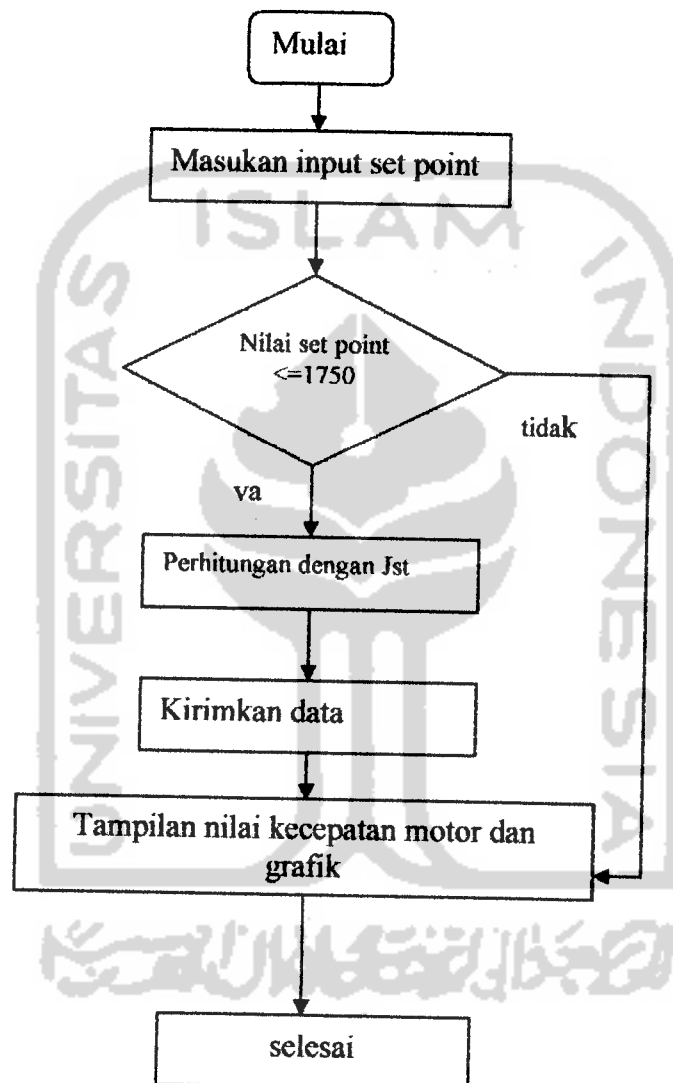
3.1.5 PC (Personal Komputer)

PC (*personal computer*) berfungsi sebagai pengolah data masukan sekaligus *interface* antara *user* dengan motor DC. Data yang akan digunakan untuk mengendalikan motor DC terlebih dahulu disimulasikan di dalam PC dengan menggunakan jaringan syaraf tiruan (JST) dan kemudian ditampilkan dalam program IDE yang menggunakan komunikasi serial pada Delphi untuk menjalankan motor DC.

Data akan dimasukan oleh user dengan menggerakkan nilai-nilai set point (0-1750 Rpm) melalui mouse PC, setelah itu data akan dikirimkan ke mikrokontroler melalui serial RS-232 menuju mikrokontroler AT89C51 untuk diproses pengaturan kecepatannya dan membangkitkan PWM di mikrokontroler AT89C2051.

3.2 Perancangan Software

Pada perancangan ini dibutuhkan tiga program untuk menjalankan motor DC, yang pertama adalah GUI Delphi, kedua program pengatur kecepatan, dan ketiga pembangkit PWM (*Pulse Width Modulation*).

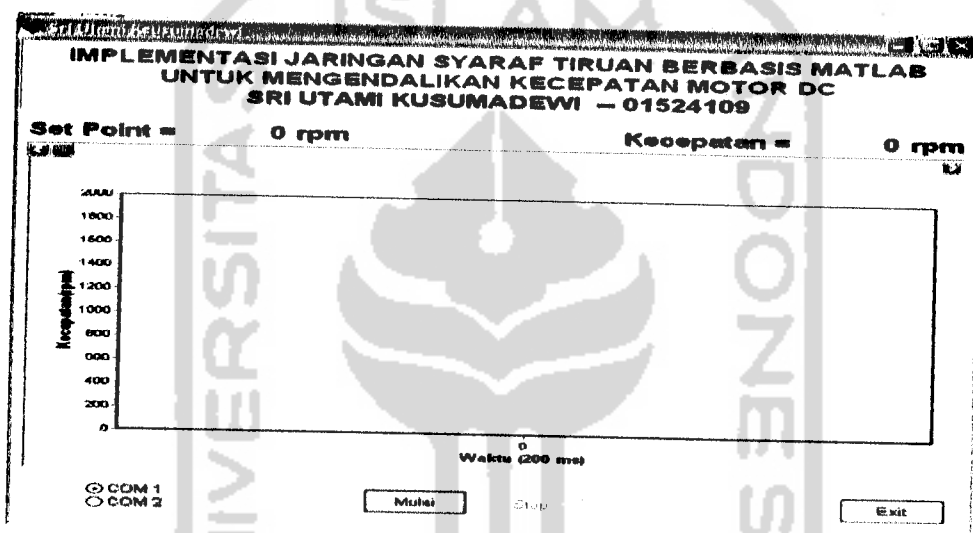


Gambar 3.6 *Flowchart* dengan Pengendali Kecepatan Motor DC

3.2.1 Perancangan IDE Delphi

Dalam pembuatan IDE (*integrated development environment*) dibutuhkan kreatifitas pembuat untuk mendapatkan tampilan yang menarik, Dalam pembuatan IDE sudah tersedia *form-form* dan *tool-tool* di dalam Delphi.

Dalam simulasi menggunakan IDE, pertama untuk menjalankannya penulis akan menekan tombol mulai untuk menjalankan programnya, setelah itu untuk memasukan nilai-nilai set point dengan cara menggeserkan *Scroll Bar* pada tampilan. Setelah itu maka akan keluar nilai kecepatan dalam Rpm dan tampilan grafiknya.



Gambar 3.7 Tampilan IDE untuk Motor DC

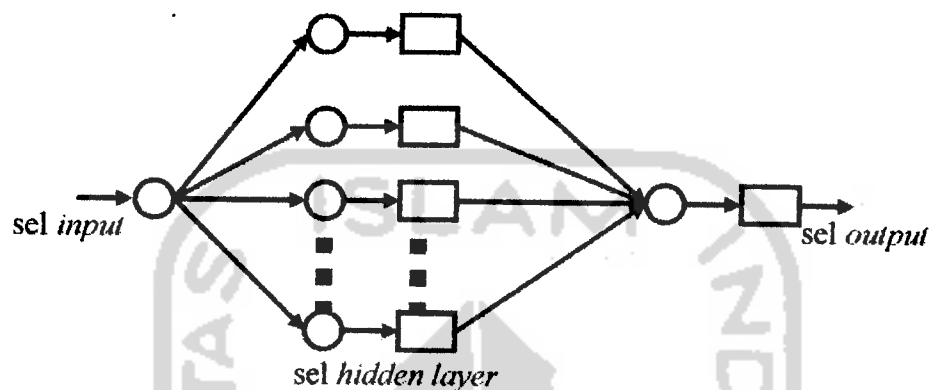
Dibawah ini contoh penggalan program dari M-file Delphi yang digunakan untuk menjalankan Motor DC.

3.2.2 Struktur Jaringan Syaraf Tiruan

JST mempunyai proses pembelajaran yang dilakukan secara *on-line /continue* (terus –menerus) , sehingga JST membutuhkan hasil pengendaliannya (kecepatan yang dihasilkan motor) untuk memperbaiki tanggapan motor, sistem masukan dalam jaringan

syaraf adalah berupa kecepatan yang diinginkan, sedangkan keluaran jaringan syaraf berupa kecepatan yang keluar dari optocoupler, dan hasil akhirnya adalah kecepatan yang dihasilkan oleh motor DC.

Dibawah ini merupakan arsitektur jaringan yang digunakan, yaitu memiliki satu neuron pada masukan, dan satu neuron pada lapisan keluaran.



Gambar 3.8 Arsitektur Jaringan Syaraf Tiruan

Jaringan syaraf tiruan yang digunakan sebagai pengendali kecepatan motor DC ini menggunakan metode *backpropagation*. Metode *backpropagation* termasuk jenis jaringan yang *autoassociative*, yaitu *range* masukan yang di proses ke dalam jaringan sama dengan *range* hasil yang dikeluarkannya.

Penentuan notasi atau tata nama yang digunakan pada pelatihan JST adalah sebagai berikut :

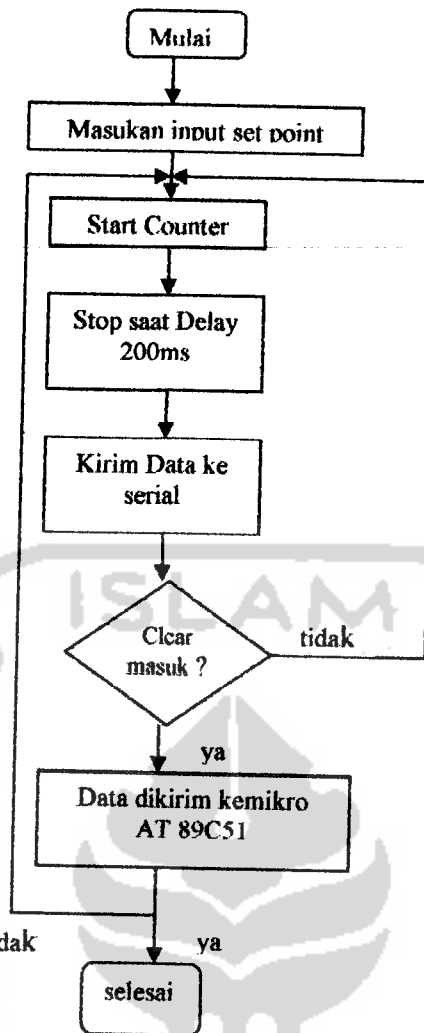
- x : Vektor masukan kedalam jaringan
- v : Bobot pada lapisan masukan ke lapisan tersembunyi
- v0 : Bias pada lapisan masukan ke lapisan tersembunyi
- w : Bobot pada lapisan lapisan tersembunyi ke lapisan keluaran

- w0 : Bias pada lapisan tersembunyi ke lapisan keluaran
- z : Sel pada lapisan tersembunyi
- y : Sel pada lapisan keluaran

3.2.3 *Software Mikrokontroller*

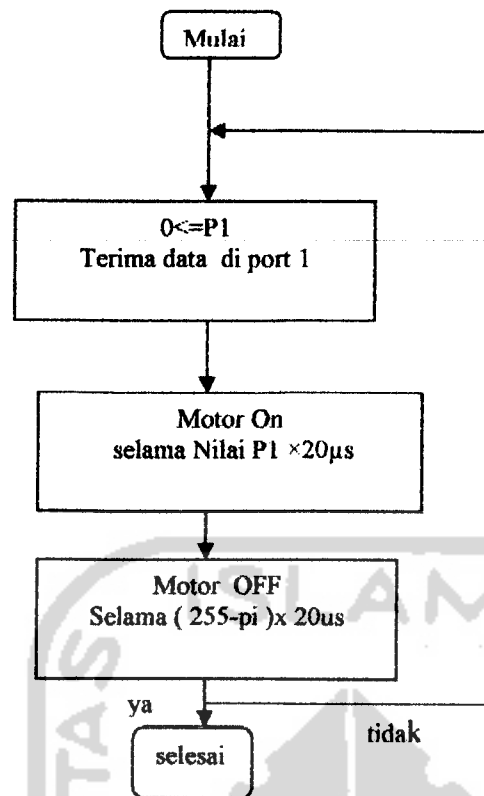
Program mikrokontroller ada dua yaitu untuk pengatur kecepatan dan untuk pembangkit PWM, sehingga menggunakan dua mikrokontroller terdiri dari mikrokontroller AT89C51 dan mikrokontroller AT89C2051. Bahasa pemrograman yang digunakan adalah bahasa C, karena bahasa C lebih mudah dipahami dibandingkan bahasa *assembler*.

Di bawah ini merupakan *flowchart* sebagai pengatur kecepatan pada mikrokontroller AT89C51. Dimana masukan dari serial dijadikan pengatur kecepatan motor DC sehingga setelah data diterima maka kecepatan putaran motor DC akan berubah sesuai dengan data yang masuk. Pada program ini *baudrate* yang digunakan adalah 57600 bps. Data yang masuk dari SBUF akan ditampung pada suatu fungsi, kemudian setelah lengkap datanya akan diproses.



Gambar 3.9 *Flowchart* Diagram Pengatur Kecepatan Motor DC

Mikrokontroller AT 89C2051 dipakai sebagai pembangkit PWM(*Pulse Width Modulation*) .setiap data yang masuk dari mikrokontroller AT89C51 maka pulsa ON dan OFF akan berubah sesuai dengan data yang masuk. Dalam perancangan PWM menggunakan resolusi sebesar $5100\mu\text{s}$, maka perlu delay sebesar $20\mu\text{s}$ ($255 \times 20 \mu\text{s} = 5100\mu\text{s}$). Dibawah ini *flowchart* dari mikrokontroller AT89C2051



Gambar 3.10 *Flowchart* pembangkit PWM

3.3 Pergerakan Motor DC

Untuk menjalankan motor DC dibutuhkan masukan tegangan suplay 150 volt , ketika driver sudah dihubungkan dengan tegangan supaly, agar motor tidak bergerak maka set point masukan diberi nilai 0 (nol) dahulu. Masukan motor DC dibatasi sesuai dengan karateristik dari motor DC yaitu dibatasi maksimum set point 1750 Rpm.

Motor yang disimulasikan diambil dari data motor yang ada pada laboratorium instalasi dan mesin listrik dasar. Berikut ini data *board* yang ada pada motor sebenarnya ;

Type :GSDT

Exciting shunt

Rpm = 1750

Armature control = ~Rpm

Field control = ~ Rpm

Bearing : DE 6203ZZ NDE 6204ZZ

Serial number R2A30812

Weight : 19 kg

Date 1992

Design : -

Output = 0.5 HP

Rating cont : -Class ins F

- Class ins f

Armature : - Voltage = 150 V

- Current = 3.2 A

- Resistance = 208.33 Ohm

- Indutance = 0.01 H

Field : - Voltage = 100 v

- Current = 0.48 A

- Resistance = 46.875 Ohm

- Indutance = 0,03H



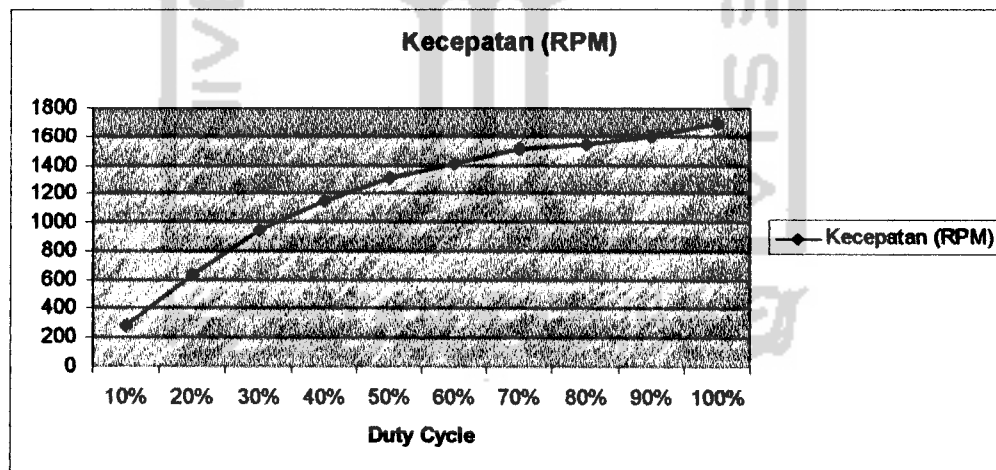
BAB IV

ANALISIS DAN PEMBAHASAN



Pada bab ini akan dibahas tentang pengujian sistem untuk membentuk sistem yang baik. Materi pengujian sistem meliputi tegangan motor, kecepatan motor dan bentuk sinyal PWM jika dikendalikan oleh *user* dengan memasukkan angka pada *layout* yang ada pada tampilan.

Untuk mendapatkan hasil kecepatan dari motor DC, maka terlebih dahulu dilakukan perbandingan terhadap masukan yang berupa *duty cycle* (%) dan keluarannya berupa kecepatan putaran motor DC dari motor yang sebenarnya. Dari data yang didapat maka akan diketahui pada saat *duty cycle* 100% kecepatan motor adalah 1687,8 rpm. Pengujian hanya diberikan masukan tegangan jangkar sebesar 100 volt, dikarenakan apabila tegangan jangkar dimaksimalkan maka pengaman atau sekering pada penyearah sering putus karena arusnya terlalu besar.



Gambar 4.1. Grafik perbandingan antara *duty cycle* dengan kecepatan motor DC

Tabel 4.1 hubungan Duty Cycle dengan kecepatan motor DC

Duty cycle	Tachometer (RPM)	Tegangan
10%	284,6	15
20%	635,5	46
30%	940	64
40%	1153,1	76
50%	1305,8	92
60%	1408,9	100
70%	1503,1	103
80%	1546,5	107
90%	1596,6	112
100%	1687,8	121

Dari tabel 4.1, dapat diketahui bahwa pada saat tegangan jangkar 100 volt kecepatan maksimum dapat mencapai 1687,8 rpm. Hal ini disebabkan karena usia motor yang cukup lama dan penggunaan yang sering dilakukan, sehingga menyebabkan perubahan pada piranti motor tidak sesuai lagi dengan standarisasi dari pabrik saat awal motor diproduksi. Data yang terdapat pada motor akan sesuai jika usia dan penggunaan motor masih dalam usia dan penggunaan wajar.

Dengan menggunakan data yang sama pada motor sebenarnya, data input dan output dari hasil simulasi dijadikan sebagai masukan dan target pada pelatihan.

4.1 Pengujian Keluaran Menggunakan motor DC

Data masukan yang diambil tabel 4.2 adalah diperoleh dari pengamatan pada tampilan delphi, tachometer, dan multimeter yang diukur melalui tegangan masukan pada motor DC.

Pada pengujian motor DC dengan tegangan masukan awal minimal 100 Volt menghasilkan kecepatan 100 rpm. Setelah motor DC jalan nilai tegangan akan naik secara bertahap. Apabila motor diberikan input masukan langsung tegangan diatas 150Volt maka akan merusak kinerja dari motor karena arusnya terlalu besar maka pengaman atau sekering pada penyearah sering putus.

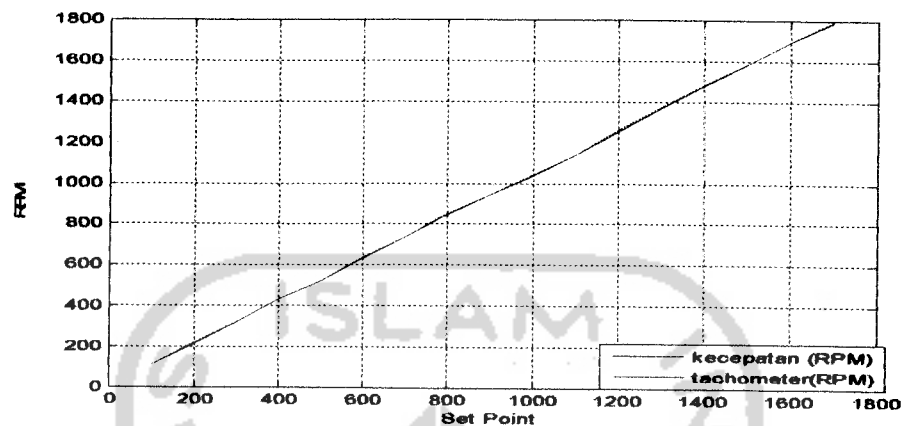
Tabel 4.2 Hasil rata – rata pengambilan data motor DC

Pengambilan Data Rata-rata			
Set Point	Kecepatan(Rpm)	Tachometer (RPM)	Tegangan Mtr
100	100	98	8
200	203	201	14
300	307	304	22
400	413	414	31
500	513	510	36
600	610	611	45
700	703	703	50
800	813	814	56
900	910	909	60
1000	1017	1018	70
1100	1110	1118	75
1200	1213	1210	85
1300	1323	1324	90
1400	1433	1434	100
1500	1530	1530	105
1600	1620	1623	115
1700	1733	1730	125

Untuk pengujian dengan motor DC real, setpoint dengan hasil kecepatan optocoupler yang di simulasikan Jaringan Syaraf Tiruan akan terdapat selisih yang disebut error. Besarnya error dapat dihitung dengan menggunakan perhitungan *mean square error* (MSE)

$$MSE = \frac{\sum \sqrt{(\text{kecepatan pada setpoint} - \text{kecepatan pada optocoupler})^2}}{n} \quad (4.1)$$

Pada gambar 4.2 akan ditampilkan grafik dari hasil keluar rata –rata tabel 4.2 Grafiknya akan ditampilkan 2 buah yaitu grafik perbandingan kecepatan tampilan Delphi dengan alat *Tachometer*, dan grafik perbandingan input set point dengan tegangan jangkar motor DC.



Gambar 4.2 Grafik hasil dari perbandingan dengan kecepatan pada tampilan Delphi dengan hasil dari tachometer

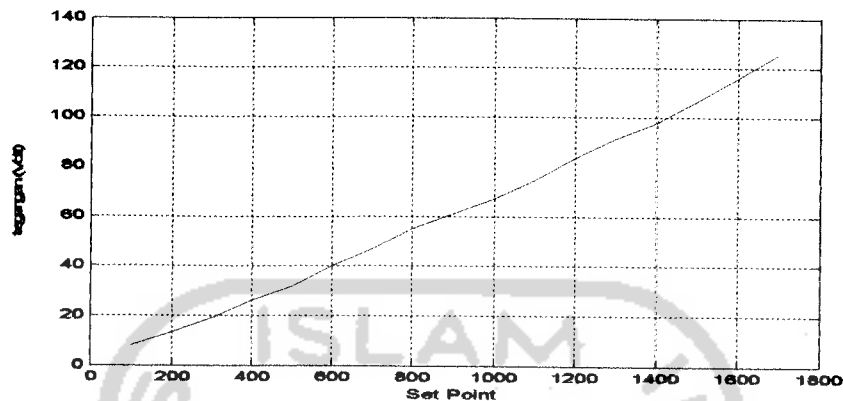
Dengan melihat dari gambar 4.2, maka hasil yang dibaca oleh optocoupler yang diolah oleh delphi, hasilnya mendekati masukan set point, sehingga dapat dihitung MSE (*mean square error*) :

$$\frac{\sum \sqrt{(\text{kecepatan pada setpoint} - \text{kecepatan pada optocoupler})^2}}{n} \cong \frac{253}{17} \cong 14.88 \text{RPM}$$

Hasil dari alat tachometer yang ditampilkan delphi sesuai masukkan set point dibandingkan dengan hasil *optocoupler* akan terlihat hasil MSE (*Mean Square error*) hampir sama:

$$\frac{\sum \sqrt{(\text{kecepatan pada setpoint} - \text{kecepatan pada tachometer})^2}}{n} \cong \frac{248.5}{17} \cong 14.62 \text{RPM}$$

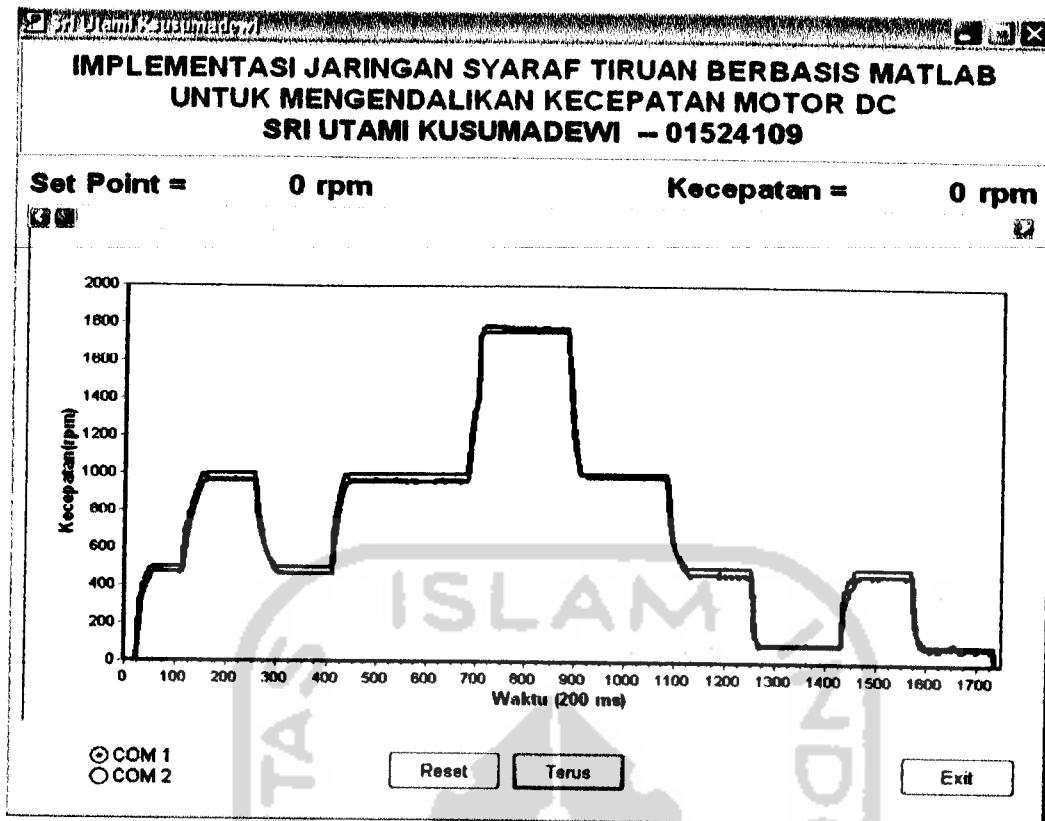
Dimana besarnya Root Mean Square Error (RMSE) atau rata – rata selisih kecepatan dari *optocopler* adalah 14.88 rpm, sedangkan dari perhitungan secara manual menggunakan *tachometer* selisihnya =14.66 rpm dari 17 data masukan kecepatan yang diambil secara acak.



Gambar 4.3 Grafik hasil dari mengukur tegangan.

Dilihat dari gambar 4.3, maka akan terlihat tegangan yang masuk dalam motor DC semakin tinggi sesuai dengan nilai masukan set pointnya. Tegangan yang masuk pada motor DC ini menghasilkan putaran (kecepatan) pada motor DC yang akan dibaca oleh *optocopler*.

Gambar 4.4, merupakan tampilan Delphi dimana garis warna biru adalah pengirim data masukan set point dan warna merah adalah masukan dari *optocopler* yang diolah dengan Jaringan Syaraf Tiruan dalam waktu 0,2 s (*optocopler* akan membaca 1(satu) putaran piringan).

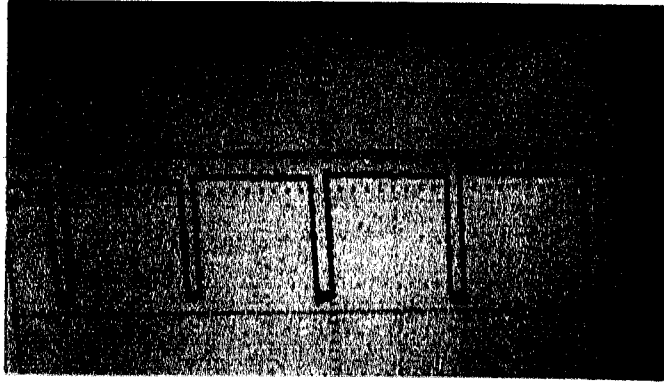


Gambar 4.4 Tampilan Delphi untuk mengetahui perbedaan error

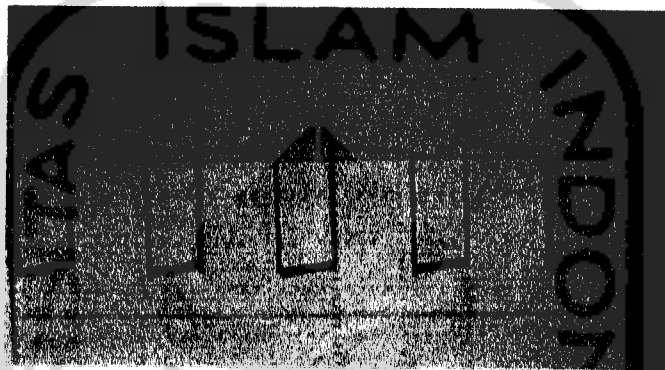
4.2 Analisa bentuk PWM

Pada gambar 4.5 sampai 5.0 menunjukkan hasil PWM dengan *duty cycle* menggunakan frekuensi penyaklaran sebesar 196Hz. Kecepatan motor tergantung pada tegangan rata-rata yang diterima terminal motor, karena pembuatan program PWM aktif low maka gambar PWM terbalik supaya ketika mikrokontroller jalan maka driver tidak akan langsung jalan. Pada saat output drivernya dalam keadaan 0 maka dia akan hidup sedangkan ketika output drivernya dalam keadaan 1 maka akan mati.

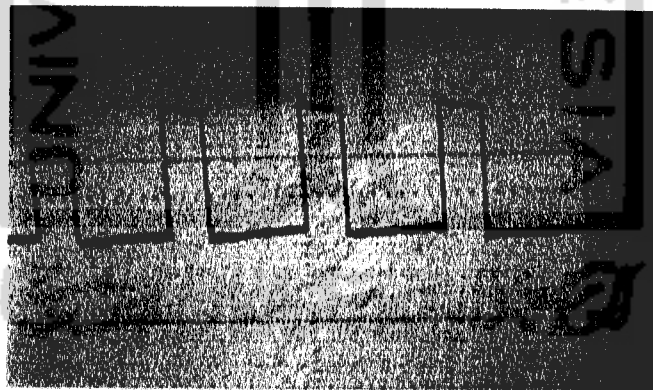
- Keluaran sinyal mikro pada saat *Duty cycle* :



Gambar 4.5 Pada saat *Duty cycle* 10%

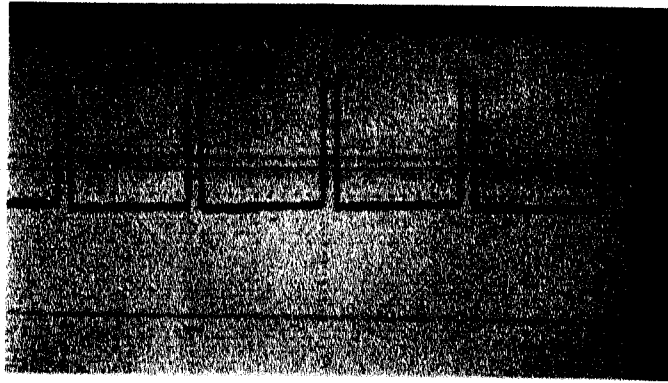


Gambar 4.6 Pada saat *Duty cycle* 40%

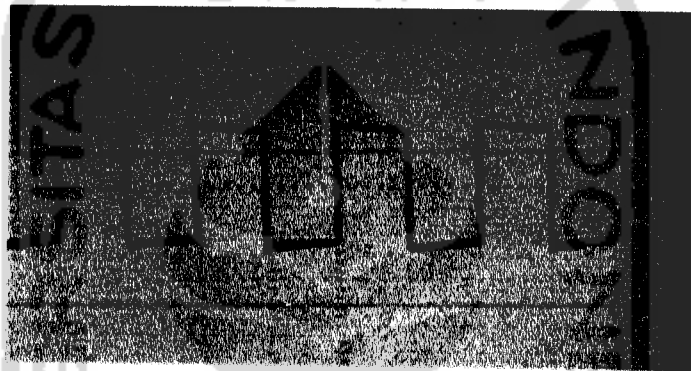


Gambar 4.7 Pada saat *Duty cycle* 70%

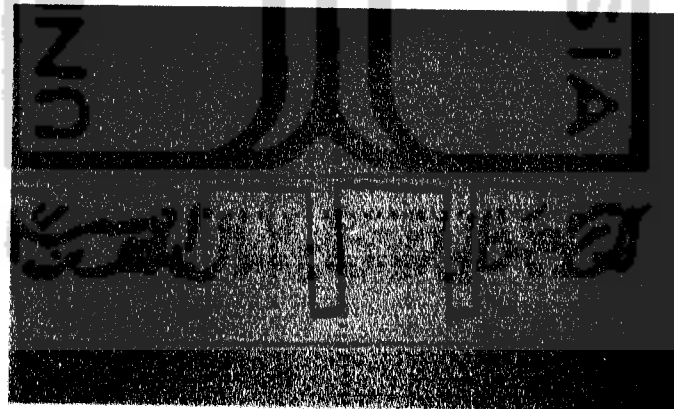
- Keluaran sinyal MOSFET pada saat *Duty cycle* :



Gambar 4.8 Pada saat *Duty cycle* 10%



Gambar 4.9 Pada saat *Duty cycle* 50%



Gambar 5.0 Pada saat *Duty cycle* 80%

Dengan melihat dari hasil sinyal PWM dari keluaran mikro dan MOSFET terlihat adanya perbedaan yaitu hasil dari sinyal keluaran mikro kebalikkan dari hasil sinyal keluaran MOSFET. Metode yang digunakan dalam pengaturan kecepatan motor DC adalah metode *Chopper*.

Metode chopper adalah prinsip perubahan converter DC/DC statis dimana konversi tegangan dilakukan power semikonduktor yang fungsinya sebagai saklar statis yang berubah saat frekuensi berulang-ulang tinggi.

$$D = \frac{T_{ON}}{T} \quad (4.2)$$

$$V_2 = V_1 \frac{T_{ON}}{T} = V_1 D \quad (4.3)$$

Keterangan :

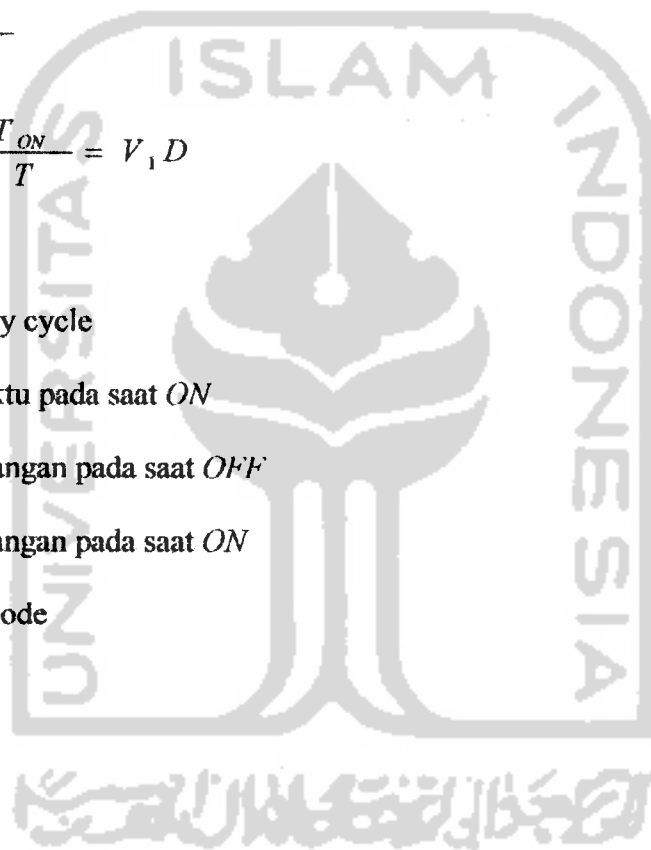
D = Duty cycle

T_{ON} = waktu pada saat *ON*

V_2 = tegangan pada saat *OFF*

V_1 = tegangan pada saat *ON*

T = periode



BAB V

PENUTUP

5.1 KESIMPULAN

Setelah selesainya uraian pada tahap-tahap pembahasan, pembuatan dan pengujian sistem pada tugas akhir ini, maka dapat diambil kesimpulan sebagai berikut :

1. Pengiriman data secara serial mempunyai kelebihan dibandingkan secara paralel, yaitu kabel untuk komunikasi serial bisa lebih panjang dibandingkan dengan paralel, dan kabel serial lebih sedikit. Dan untuk menghubungkan dua perangkat computer yang berjauhan hanya cukup dengan 3 kabel yaitu TXD, RXD, dan Ground.
2. Struktur terbaik jaringan syaraf tiruan untuk sistem kendali kecepatan motor DC terdiri dari 5 sel neuron lapisan *input*. Lapisan tersembunyi terdiri dari 2 lapisan, dimana lapisan tersembunyi pertama memiliki 5 sel neuron, lapisan tersembunyi kedua terdiri dari 1 sel neuron. (*Mean Square Error*) MSE dengan fungsi aktivasi setiap lapisan menggunakan fungsi *purelin*.
3. Setelah dilakukan pengujian dengan komunikasi serial dua arah, yang menggunakan Delphi maka antara output dari pengujian Jaringan Syaraf Tiruan dengan pengujian langsung dengan motor DC, hasilnya memperkecil *error* terhadap kecepatan Motor DC. Dimana hasil MSE (*mean square error*) dari membaca optocoupler adalah 14.88 rpm.

5.2 SARAN

Untuk pengembangan pengendalian motor DC dengan serial dua arah selanjutnya penulis menyarankan :

1. Dalam merancang hardware, alat ukur dan perangkat lain yang dipilih harus mempunyai kualitas bagus untuk mendapatkan hasil yang lebih baik.
2. Coba menggunakan komunikasi serial ke dalam matlab untuk menampilkan masukan yang berasal dari pengiriman data hardware ke software..



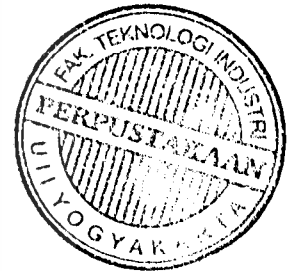
DAFTAR PUSTAKA

- Achyanto, Djoko, 1997. *Mesin-mesin Listrik Edisi ke-Empat*. Jakarta : Erlangga
- Budioko, Totok, 2005. *Belajar dengan Mudah dan Cepat Pemrograman Bahasa C sengan SDCC (Small Device C Compiler) pada Mikrokontroler AT89X051/AT89C51/52 Teori, Simulasi dan Aplikasi*. Yogyakarta: Gava Media
- Hartato, Dwi ; Raharjo, Suwanto, 2005, *Microcontoler AT89C2051*, Yogyakarta : Andi
- Instalasi dan Mesin Listrik, Laboratorium, 2004. *Modul praktikum Teknik Tenaga Listrik*. Yogyakarta: Teknik Elektro Fakultas Teknologi Industri Universitas Islam Indonesia
- Malik, Jaja Jamaludin, 2006, *Kumpulan Latihan Pemrograman Delphi*, Yogyakarta : Andi
- Putra, Afgianto Eko, 2002, *Teknik Antarmuka Komputer, Konsep Dan Aplikasi*, Yogyakarta: Graha Ilmu
- Panduan Praktis Pemrograman Borland Delphi 7.0, Yogyakarta : Andi ; Semarang : Wahana Komputer
- Sari, Indria, 2005. *Implementasi Jaringan Syaraf Tiruan Berbasis Metode Back Propagation Sebagai pengendali Kecepatan Motor DC dengan Komunikasi Serial Berbasis Matlab*. Skripsi, Tidak Diterbitkan. Yogyakarta: Teknik Elektro Fakultas Teknologi Industri Universitas Islam Indonesia

Vithayathil, Josep, 1995, *Power Electronics Principles and Applications*,
California Polytechnic State University : McGraw-Hill ; New York

Wiryadinata, Romi, 2005, *Simulasi Jaringan Syaraf Tiruan Berbasis
Metode Back Propagation Sebagai pengendali Kecepatan Motor DC.*
Skripsi, Tidak Diterbitkan. Yogyakarta: Teknik Elektro Fakultas
Teknologi Industri Universitas Islam Indonesia

Wardhana, Lingga ; Suwastono Addin, 2005, *Belajar Sendiri Pembuatan
Skematik Rangkaian Elektronis dan Layout PCB Menggunakan
OrCAD Release 9.1*, Yogyakarta: Andi



LAMPIRAN



➤ **Lampiran 1, Program pada Delphi**

unit Unit1;

interface

uses

Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
TeEngine, Series, ExtCtrls, TeeProcs, Chart, StdCtrls, CPortCtl, CPort,
ActnList, StdActns, ComCtrls, Menus, XPMan;

type

TForm1 = class(TForm)

Chart1: TChart;

ComPort: TComPort;

StatusBar1: TStatusBar;

Panel1: TPanel;

btSave: TButton;

Timer1: TTimer;

btReset: TButton;

btMulai: TButton;

Series2: TLineSeries;

btStop: TButton;

btContinue: TButton;

btExit: TButton;

rbCom1: TRadioButton;

rbCom2: TRadioButton;

XPManifest1: TXPManifest;

Panel2: TPanel;

ScrollBar1: TScrollBar;

Panel3: TPanel;

Label7: TLabel;

Label1: TLabel;

Panel4: TPanel;

Label2: TLabel;

Label4: TLabel;

Label5: TLabel;

lbPV: TLabel;

Label3: TLabel;

Label6: TLabel;

Button1: TButton;

Series1: TLineSeries;

Label8: TLabel;

procedure FormCreate(Sender: TObject);

procedure ComPortRxChar(Sender: TObject; Count: Integer);

procedure FormClose(Sender: TObject; var Action: TCloseAction);




```

procedure btMulaiClick(Sender: TObject);
procedure btResetClick(Sender: TObject);
procedure btStopClick(Sender: TObject);
procedure btContinueClick(Sender: TObject);
procedure btExitClick(Sender: TObject);
procedure Close1Click(Sender: TObject);
procedure rbCom1Click(Sender: TObject);
procedure rbCom2Click(Sender: TObject);
procedure ScrollBar1Change(Sender: TObject);
private
  { Private declarations }
public

  { Public declarations }
end;

var
  Form1: TForm1;
  i: single;
  rpm,j: extended;
  data_serial: byte;
  setpoint,present_value,left_max: integer;
  time1: TDateTime;

  x   : Array[1..2] of extended;
  v   : Array[1..2,1..5] of extended;
  v0  : Array[1..5] of extended = (-0.76786,-0.54192,-0.73045,-0.25363,-0.089317);
  z   : Array[1..5] of extended;

  fz  : Array[1..5] of extended;
  w   : Array[1..3,1..5] of extended;
  w_t : Array[1..5,1..3] of extended;
  w0  : Array[1..3] of extended = (1.7165,-1.0705,0.33936);
  y   : Array[1..3] of extended;

  fy  : Array[1..3] of extended;
  w_out : Array[1..3] of extended = (-2.1318,1.8786,-0.69901);
  w_out_t : Array[1..1,1..3] of extended;

  w0_out: extended;
  y_out,x_old: extended;
  d,k,present_vallue: integer;

implementation

{$R *.DFM}

```

```

procedure TForm1.FormCreate(Sender: TObject);
begin

```

```

  v[1,1]:=0.47008;
  v[1,2]:=-0.81074;
  v[1,3]:=-0.20179;
  v[1,4]:=-0.38432;
  v[1,5]:=1.0015;
  v[2,1]:=0.52384;
  v[2,2]:=-0.087159;
  v[2,3]:=-0.96354;
  v[2,4]:=0.64264;
  v[2,5]:=-0.11054;
  w[1,1]:=1.1328;
  w[1,2]:=0.64938;
  w[1,3]:=1.4516;
  w[1,4]:=0.067861;
  w[1,5]:=-0.1943;
  w[2,1]:=0.58644;
  w[2,2]:=0.093118;
  w[2,3]:=-0.16147;
  w[2,4]:=-0.65332;
  w[2,5]:=-0.31559;
  w[3,1]:=0.74758;
  w[3,2]:=-0.54839;
  w[3,3]:=-0.77743;
  w[3,4]:=-0.72149;
  w[3,5]:=-0.63254;
  w0_out:= 1.7760;
  i:=0;
  left_max:=0;
end;

```

```

procedure TForm1.ComPortRxChar(Sender: TObject; Count: Integer);
begin

```

```

  ComPort.Read(data_serial,1);

  x[1]:=(setpoint-data_serial)/175; //error = setpoint-data sekarang
  x[2]:=(x[1]-x_old)/175;          //derror = error sekarang-error sebelumnya
  // z = v0+(x*v);
  for k := 1 to 5 do
  begin
    z[k]:=x[1]*v[1,k]+x[2]*v[2,k]+v0[k];
  end;

  //fz = purelin(z);

```

```

for k := 1 to 5 do
begin
  fz[k]:=z[k];
end;

//w' --> matrik 5x3
for d := 1 to 3 do
begin
  for k := 1 to 5 do
  begin
    w_t[d,k]:=w[k,d];
  end;
end;

// y = w0+(fz*w'); --> matrik 1x3
for k := 1 to 3 do
begin
  y[k]:=fz[1]*w_t[1,k]+fz[2]*w_t[2,k]+fz[3]*w_t[3,k]+fz[5]*w_t[5,k]+w0[k];
end;

// w_out' --> matrik 3x1
for d := 1 to 3 do
begin
  w_out_t[d,1]:=w_out[d];
end;

//fy = purelin(y);
for k := 1 to 3 do
begin
  fy[k]:=y[k];
end;

//y_out = w0_out+(fy*w_out');
y_out:=w0_out+fy[1]*w_out_t[1,1]+fy[2]*w_out_t[1,2]+fy[3]*w_out_t[1,3];

y_out:=y_out+w_out[1];

x_old:=x[1];

present_vallue:=round(y_out+175);

rpm:=data_serial * 10;
if rpm>2000 then rpm:=2000;
Series1.AddXY(i,setpoint,"");
Series2.AddXY(i,rpm,"");

```

```
lbPV.Caption:=FloatToStrF(rpm,ffFixed,5,0);
chart1.LeftAxis.Maximum:=2000;
i:=i+1;
end;
```

```
comport.Write(present_value,1);
```

```
procedure TForm1.FormClose(Sender: TObject; var Action: TCloseAction);
begin
  scrollbar1.Position:=0;
  ComPort.Close;
end;
```

```
procedure TForm1.btMulaiClick(Sender: TObject);
begin
  btMulai.Visible:=false;
  btReset.Visible:=true;
  btContinue.Visible:=false;
  btStop.Visible:=true;
  btStop.Enabled:=true;

  comport.Connected:=true;
  timer1.Enabled:=true;
  chart1.AddSeries(Series2);
end;
```

```
procedure TForm1.btResetClick(Sender: TObject);
begin
  btMulai.Visible:=true;
  btReset.Visible:=false;
  btContinue.Visible:=false;
  btStop.Visible:=true;
  btStop.Enabled:=false;
  comport.Connected:=false;
  series1.Clear;
  series2.Clear;
  i:=0;
end;
```

```
procedure TForm1.btStopClick(Sender: TObject);
begin
  btContinue.Visible:=True;
  btStop.Visible:=false;
  comport.Connected:=false;
end;
```

```
procedure TForm1.btContinueClick(Sender: TObject);
begin
  btContinue.Visible:=false;
  btStop.Visible:=true;
  comport.Connected:=true;
end;
```

```
procedure TForm1.btExitClick(Sender: TObject);
begin
  form1.Close;
end;
```

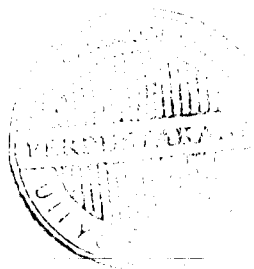
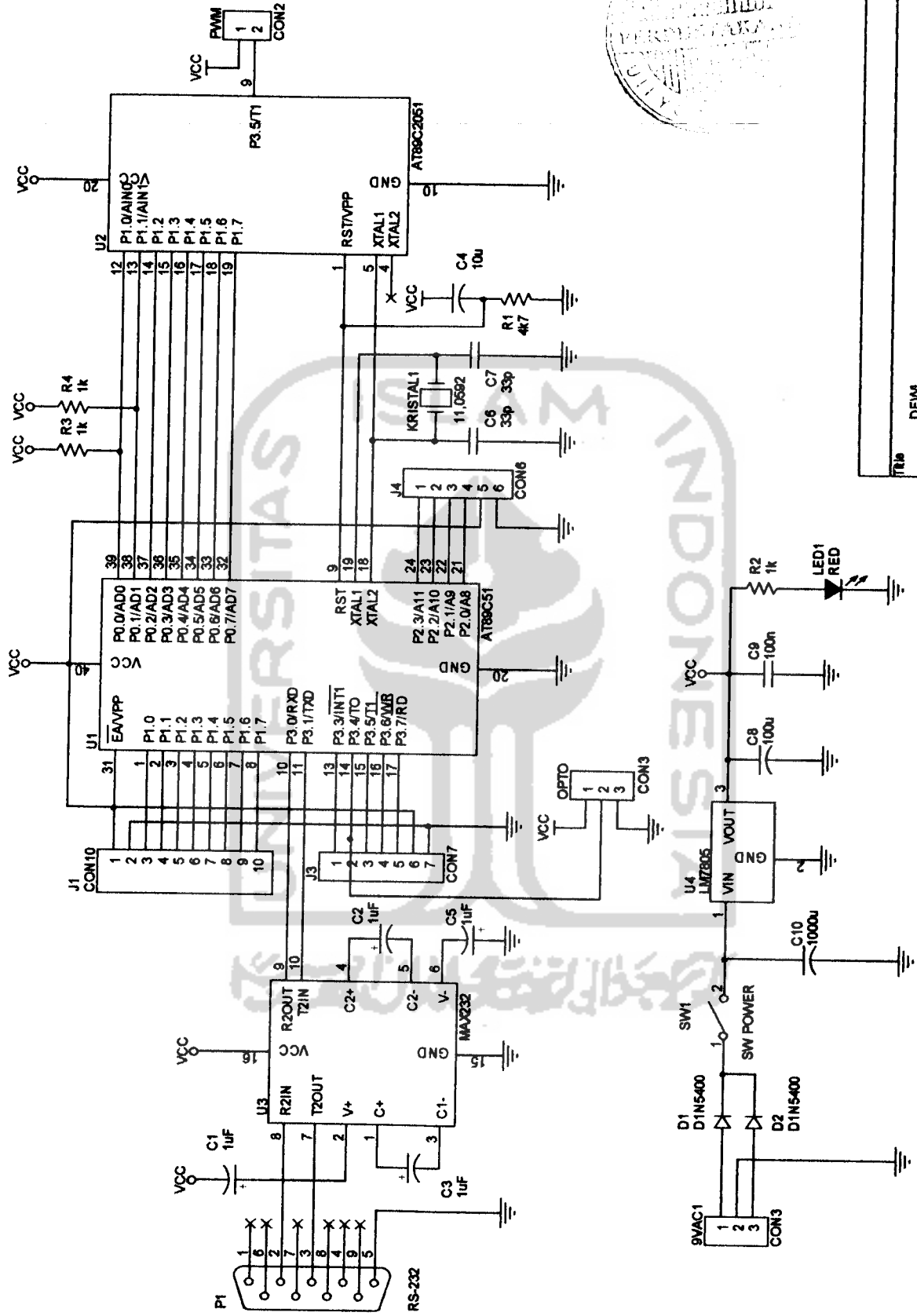
```
procedure TForm1.Close1Click(Sender: TObject);
begin
  btExit.Click;
end;
```

```
procedure TForm1.rbCom1Click(Sender: TObject);
begin
  rbcom2.Checked:=false;
  comport.Port:='COM1';
end;
```

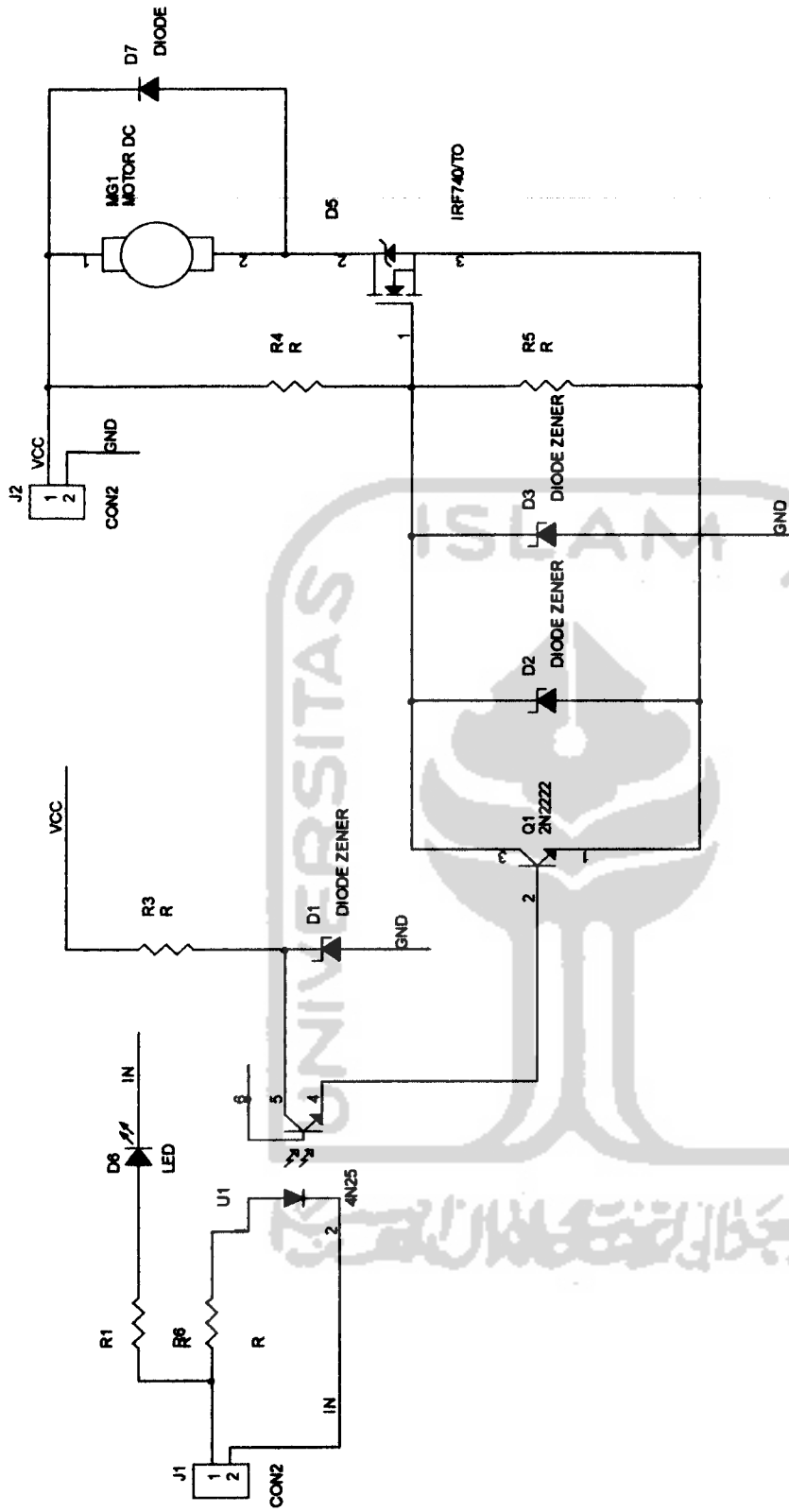
```
procedure TForm1.rbCom2Click(Sender: TObject);
begin
  rbcom1.Checked:=false;
  comport.Port:='COM2';
end;
```

```
procedure TForm1.ScrollBar1Change(Sender: TObject);
var SetpointJST: byte;
begin
  setpoint:=scrollbar1.Position;
  if comport.Connected then comport.Write(setpoint,1);
  setpoint:=setpoint *10;
  label4.Caption:=inttostr(setpoint);
end;
```

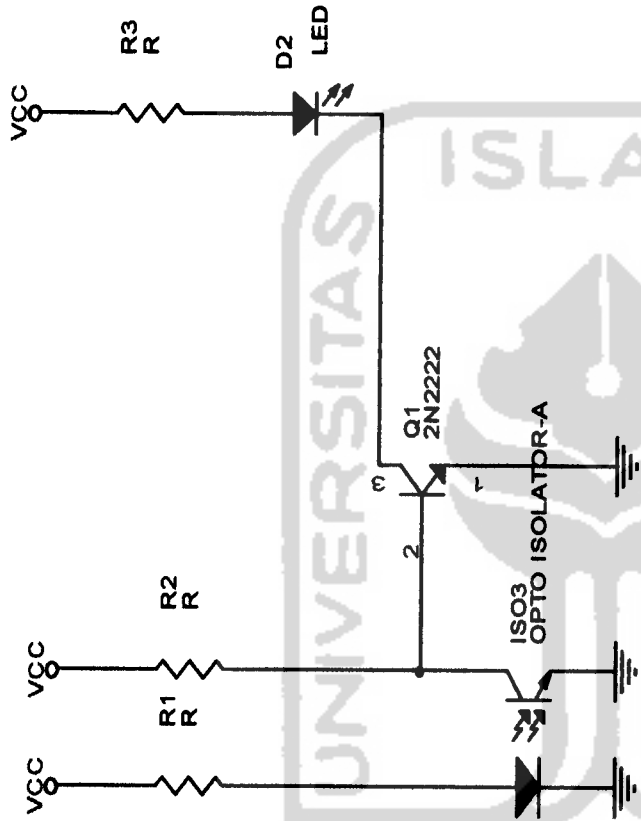
```
end.
```

Title		DEWM
Size	Document Number	
Customer	Doc>	
Date:	Sunday, September 24, 2006	Sheet 1 of 1



Title		<Title>
Size	Document Number	Rev
A	<Doc>	<Rev Code>
Date:	Tuesday, January 09, 2007	Sheet 1 of 1



Title		optocoupler	
Size	Document Number	Rev	<Rev Code>
A	Sri Utami Kusumadewi	01 524 109	
Date:	Monday, January 07, 2002	Sheet	1 of 1