LAPORAN TUGAS AKHIR

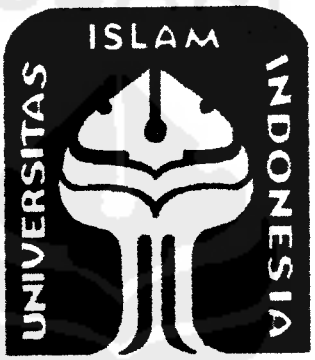
PENGENDALIAN LENGAN ROBOT

MELALUI JARINGAN KOMPUTER

Diajukan Guna Melengkapi Sebagian Syarat Untuk Mendapatkan Gelar Sarjana S1

Pada Program Studi Jurusan Teknik Elektro Fakultas Teknologi Industri

Universitas Islam Indonesia



Disusun Oleh :

Nama : ENGGAR ADI CAHYANTO

No. Mhs : 02 524 068

JURUSAN TEKNIK ELEKTRO

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ISLAM INDONESIA

YOGYAKARTA

2007

LEMBAR PENGESAHAN PEMBIMBING

PENGENDALIAN LENGAN ROBOT MELALUI JARINGAN KOMPUTER

TUGAS AKHIR

Oleh :

Nama : Enggar Adi Cahyanto

No. Mahasiswa : 02 524 068

Yogyakarta, Mei 2007

Pembimbing I	Pembimbing II
Ir.H.Suyamto	Yusuf Aziz Amrulloh,ST

LEMBAR PENGESAHAN DOSEN PENGUJI

PENGENDALIAN LENGAN ROBOT MELALUI JARINGAN KOMPUTER

TUGAS AKHIR

Disusun oleh :

Nama : Enggar Adi Cahyanto

No. Mhs : 02 524 068

Telah Dipertahankan Di Depan Sidang Penguji, Sebagai Salah Satu Syarat Untuk Memperoleh Gelar Sarjana Teknik Elektro, Fakultas Teknologi Industri, Universitas Islam Indonesia

Yogyakarta, Juni 2007

Tim Penguji

Ir. H. Suyamto
Pembimbing I

Yusuf Aziz Amrulloh, ST
Pembimbing II

Tito Yuwono, ST, M.Sc
Penguji

Mengetahui,
Ketua Jurusan Teknik Elektro
Fakultas Teknologi Industri
Universitas Islam Indonesia

(Tito Yuwono, ST, M.Sc)

HALAMAN PERSEMBAHAN

Syukur Alhamdulillah kepada Allah SWT, yang telah memberikan rahmat dan hidayah-NYA, sehingga dapat menyelesaikan tugas akhir ini.

Skripsi ini didedikasikan dan dipersembahkan sebagai ungkapan terima kasih dengan tulus kepada mereka yang sangat berarti dalam hidupku :

Ayahanda Tercinta, atas bimbingan, ketauladanan, pengorbanan, kesabaran dan do'a ayahanda adalah motivator utama dalam hidupku.

Ibunda Tersayang, wujud kasih sayangmu, kesabaran dan ketabahan serta keselarasan hidup yang telah ibunda tunjukkan telah mendewasakanku.

Kakakku Tercinta yang telah memberikan motivasi yang sangat berarti.

Untuk *my beloved* Vita Meria D.N yang selalu menyayangiku.

Terimakasih atas do'a serta kasih sayang dan kepercayaanya yang telah diberikan kepadaku.

Semoga menjadi kenangan yang indah dan tak terlupakan
Amien Ya Robbal 'Aalamien.....

MOTTO

- *"Jadikanlah Sabar dan Sholat sebagai penolongmu, dan sesungguhnya yang demikian itu sungguh berat, kecuali bagi yang khusuk"*

(QS. AL Baqarah 45)

- *"Allah meninggikan orang yang beriman di antara kamu dan orang-orang yang diberi ilmu pengetahuan, beberapa derajat."*

(QS. Almujaadalah 11)

- *"Sesungguhnya sesudah kesulitan itu ada kemudahan."*

(Q.S. Asy Syarh; 5)

- Pandanglah kegagalan sebagai suatu peluang untuk belajar, sebagai suatu lompatan kreativitas dan sebagai suatu kesempatan untuk menguji gagasan baru.

(Art Martell)

KATA PENGANTAR

بسم الله الرحمن الرحيم

Assalamu'alaikum Wr. Wb.

Puji syukur kehadirat Allah SWT, atas nikmat iman, rahmat, hidayah dan pikiran yang diberikan. Sehingga penulis dapat menyelesaikan sebuah tugas akhir dengan judul " Pengendalian Lengan Robot Melalui Jaringan Komputer". Tidak lupa shalawat serta salam selalu tercurah kepada Nabi Muhammad. SAW beserta keluarga dan para sahabatnya.

Adapun maksud dari penyusunan tugas akhir ini adalah untuk memenuhi kurikulum S-1 Jurusan Teknik Elektro, Fakultas Teknologi Industri, Universitas Islam Indonesia. Disamping itu juga untuk menambah pengetahuan atas disiplin ilmu yang telah dipelajari di bangku perkuliahan untuk diterapkan ke masyarakat.

Selama mengerjakan tugas akhir dan dalam penyusunan laporan, tidak lepas dari hambatan, namun berkat motivasi, informasi dan konsultasi dari berbagai pihak, semua masalah dapat diatasi. Untuk itu penyusun menyampaikan rasa hormat sebagai ungkapan terima kasih kepada :

1. Kedua orang tuaku yang senantiasa memberikan dukungan moril, materi dan do'a setiap saat.
2. Bpk Fathul Wahid, ST, M.Sc, selaku Dekan Fakultas Teknologi Industri Universitas Islam Indonesia.

3. Bpk Tito Yuwono, ST, M.Sc, selaku Ketua Jurusan Teknik Elektro Fakultas Teknologi Industri Universitas Islam Indonesia.
4. Bpk.Ir.H. Suyamto sebagai dosen pembimbing I.
5. Bpk Yusuf Aziz Amrulloh, ST selaku Sekretaris Jurusan Teknik Elektro dan sebagai dosen pembimbing II.
6. Ibu Dwi Anna Ratnawati, ST selaku Ka.Lab Kendali dan Dosen Pembimbing Akademik.
7. Segenap dosen Teknik Elektro Fakultas Teknologi Industri Universitas Islam Indonesia yang telah memberikan ilmunya.
8. Mas Heri selaku Kor.Lab Kendali dan Mas Agung selaku Kor.Lab Imel.
9. Seluruh mahasiswa jurusan Teknik Elektro UII dan teman-teman 2002 khususnya Rusdani, Dedi, Dede, Yudi, Hanggit, Endi, Mukti, Whinar, Cahyo, Eko, Iwan, Herdian, Bagus terima kasih atas bantuannya yang telah diberikan dan kalian akan selalu menjadi sahabat-sahabatku.
10. Teman-teman kontrakan Arda, Dana, Yusuf, Nanda, Doddi dan seluruh pihak yang tidak dapat disebutkan satu-persatu yang telah memberikan dukungan dan do'a.
11. *My beloved* Vita Meria D.N *thank's for all.*

Penulis menyadari bahwa penyusunan tugas akhir ini masih terdapat kesalahan dan kekurangan. Oleh karena itu, kritik dan saran yang membangun akan senantiasa

penulis terima dengan senang hati. Semoga laporan ini dapat bermanfaat kepada penulis pada khususnya dan kepada pembaca yang budiman pada umumnya, amin.

Wassalamu'alaikum Wr.Wb

Yogyakarta, Juni 2007

Enggar Adi Cahyanto

ABSTRAK

Pemanfaatkan jaringan komputer dapat untuk mendistribusikan suatu data secara tepat dan cepat. Jaringan komputer LAN menyediakan komunikasi berkecepatan tinggi pada komputer-komputer yang dihubungkan antara satu dengan yang lain, baik menggunakan 'hub' atau tidak. Untuk instalasi jaringan komputer tersebut digunakan dua metode pengkabelan, yaitu Kabel Lurus *(Straight)* dan Kabel Silang *(Crossover)*. Dengan digunakannya 'hub', alat ini dapat dikendalikan oleh beberapa *client*, tidak seperti halnya tanpa menggunakan 'hub yang hanya dapat dikendalikan oleh satu *client* saja. Agar komputer dapat melaksanakan tugas-tugas yang diinginkan, maka diperlukan *software* untuk menjalankan piranti-piranti masukan / keluaran yang sesuai dengan kebutuhan. *Borland Delphi* merupakan salah satu bahasa pemrograman tingkat tinggi yang digunakan untuk menjalankan semua rangkaian pengendali lengan robot melalui jaringan. Sistem pengendalian yang dibuat yaitu dengan membuat suatu *client*-server, dimana *client* sebagai komputer pengendali untuk menggerakan lengan robot dengan cara memberikan masukan berupa sudut (°) pada setiap *joint*-nya. Sedangkan server akan selalu *stand-by* menunggu *client* yang sekiranya akan mengakses alamat dari server tersebut. Ketika server diakses *client*, maka server sudah berada pada posisi siap menerima data masukan dan siap pula untuk meneruskan masukan tersebut ke mikrokontroler melalui port serial (RS-232). Setiap perlengkapan akan dikontrol melalui server yang kemudian akan diteruskan ke mikrokontroler untuk menggerakkan masing-masing dari motor servo yang digunakan sebagai *joint* pada lengan robot berbasis AT89C2051 ini. Dari hasil perancangan, alat ini dapat berfungsi seperti yang diinginkan meskipun ada kesalahan/*error* pada *joint* 1 yaitu saat diberi masukan sudut 60°, 70°, 80°,110° dan *error* dengan kesalahan terbesar yakni 1,66% saat sudut masukan 60°. Sedangkan *joint* 2, *joint* 3, dan *joint* 4 tidak terjadi *error* atau dengan kata lain *error*-nya 0. Ketidakpresisian hasil yang dicapai *end effector* lengan robot disebabkan ketidaktepatan posisi dan ukuran bagian-bagian lengan robot serta kekurangtelitian dalam kalibrasi dan perbedaan pulsa input juga sangat mempengaruhi keakuratan pergerakan motor servo.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I

PENDAHULUAN

1.1 Latar Belakang

Dunia telah menujukan adanya perkembangan teknologi yang semakin canggih. Dengan semakin berkembangnya teknologi, maka semakin berkembang pula perangkat – perangkatnya dan jaringan komputer salah satu dari sekian banyaknya perkembangan teknologi itu. Jaringan komputer sendiri merupakan gabungan antara teknologi komputer dan teknologi komunikasi. Gabungan teknologi ini melahirkan pengolahan data yang dapat didistribusikan secara tepat dan cepat.

Agar komputer dapat melaksanakan tugas-tugas yang diinginkan, maka diperlukan *software* untuk menjalankan piranti-piranti masukan dan keluaran yang sesuai dengan kebutuhan. *Borland Delphi* merupakan salah satu bahasa pemrograman yang digunakan untuk menjalankan semua rangkaian pengendali peralatan robot melalui jaringan komputer.

Melalui jaringan komputer ini diharapkan pemakai dapat mengendalikan robot secara jarak jauh. Setiap perlengkapan akan dikontrol melalui server yang kemudian diteruskan ke mikrokontroler melalui *port serial* dari server untuk diproses perintahnya dan diteruskan ke *driver* dari robot tersebut sehingga robot dapat bergerak sesuai dengan perintah yang telah diberikan oleh *client*. Data yang diberikan oleh *client* berupa sudut (°) dan urutan motor servo ke berapa yang dioperasikan sebagai *joint* pada lengan robot tersebut.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan sebelumnya, maka dapat diambil suatu rumusan masalah : "Bagaimanakah cara mengendalikan lengan robot melalui jaringan komputer".

1.3 Batasan Masalah

Dengan adanya batasan masalah, maka akan lebih mudah dalam menyederhanakan dan mengarahkan penelitian agar tidak menyimpang dari apa yang diteliti. Hal-hal yang akan diterangkan dalam laporan tugas akhir ini sekiranya adalah sebagai berikut :

1. Penulisan difokuskan pada cara mengendalikan lengan robot melalui jaringan komputer.
2. Komunikasi antar komputer merupakan komunikasi satu arah, yakni server hanya dapat menerima data dari *client*.

1.4 Tujuan Penelitian

Tujuan penelitian pada tugas akhir ini adalah : Mengendalikan lengan robot melalui jaringan komputer.

1.5 Manfaat

1. Memberikan kemudahan bagi manusia untuk mengendalikan lengan robot dari jarak jauh.

1.6 Sistematika Penulisan

Dalam penulisan, sistematika penulisannya terdiri dari lima bab yaitu :

BAB I PENDAHULUAN

Dalam bab ini dijelaskan tentang latar belakang masalah yang isinya menjelaskan kegunaan dari hasil perancangan tugas akhir, perumusan masalah yaitu bagaimana cara mengendalikan lengan robot melalui jaringan komputer, batasan masalah yang isinya menjelaskan batasan dalam penulisan tugas akhir agar penulisan tidak menyimpang dari apa yang dibuat dalam tugas akhir ini, tujuan penulisan dan sistematika penulisan yang isinya menjelaskan urutan dalam penulisan laporan tugas akhir.

BAB II DASAR TEORI

Dalam bab ini dijelaskan tentang teori-teori yang digunakan dalam perancangan dan pembuatan peralatan. Yang pertama dasar teori tentang robot, yang kedua dasar teori tentang mikrokontroler, yang ketiga dasar teori tentang *PWM(Pulse Width Modulation)*, yang keempat dasar teori tentang motor servo, yang kelima dasar teori *IP Address*, yang keenam dasar teori tentang komunikasi RS232, dan yang terakhir dasar teori tentang jenis perangkat lunak yang digunakan.

BAB III PERANCANGAN SISTEM

Dalam bab ini dijelaskan mengenai perancangan sistem dan desain perangkat kerasnya. Perancangan sistem kendali lengan robot melalui jaringan komputer dibagi menjadi 2 tahap, yaitu perancangan perangkat keras yang meliputi perancangan sistem minimal mikrokontroler dan unit konverter ke RS232 . Sedangkan perancangan perangkat lunak yaitu pembuatan program dengan bahasa C *(Borland Delphi 7)* dan program untuk mikrokontroler AT89C2051 yang akan di-*download* sistem minimal mikrokontroler.

BAB IV PENGUJIAN, ANALISA DAN PEMBAHASAN

Dalam bab ini dijelaskan hasil dari pengujian alat yang dibuat dan kemudian di analisa. Pengamatan dan pengujian pada bab ini dilakukan pada beberapa bagian, yang pertama pengamatan pada *joint1, joint2, joint3, joint4* saat diberi masukan sudut oleh *client*, yang kedua pengamatan difokuskan pada instalasi komponen jaringan dengan kabel *UTP*, dan yang terakhir pengamatan pada *client-server*.

BAB V PENUTUP

Dalam bab ini berisi kesimpulan-kesimpulan dari peralatan yang dibuat dan berisi saran-saran guna perbaikan dan pengembangan alat ini.

BAB II

DASAR TEORI

2.1 Robot

Ilmu robotika merupakan ilmu yang sangat penting dan berpengaruh dalam perkembangan teknologi masa kini. Fungsi dari robot sendiri untuk saat ini sangatlah terasa bermanfaat terutama dalam dunia industri yang mana dapat membantu dalam proses produksi suatu produk tertentu. Ilmu robotika dapat dikatakan sebagai ilmu yang komplek, karena ilmu robotika merupakan kumpulan dari beberapa disiplin ilmu. Robot dapat dikatakan sangat bermanfaat dalam dunia industri karena robot memiliki keunggulan antara lain mampu beroperasi dalam suhu yang tinggi, juga mampu bekerja sepanjang waktu tanpa mengenal lelah, dan mempunyai tingkat produktivitas yang tinggi.

2.1.1 Definisi Robot

Robot berasal dari kata *robota* yang berasal dari bahasa ceko yang artinya budak atau pekerja. Unsur-unsur utama yang harus terkandung dalam robot adalah :

1. Bekerja secara otomatis
2. Dapat diprogram
3. Mampu melaksanakan tugas tertentu sesuai dengan program

Secara umum definisi robot itu sendiri adalah suatu alat multi fungsi otomatis yang dapat diprogram untuk melakukan pekerjaan tertentu. Sedangkan tujuan dari

pembuatan robot sendiri adalah untuk menggantikan fungsi manusia dan pekerjaan yang tidak dapat dilakukan oleh manusia. Selain itu robot juga dapat diprogram agar dapat bekerja secara otomatis. Keunggulan sistem robot dibandingkan dengan manusia antara lain sbb :

1. Tidak lelah
2. Dapat bekerja pada keadaan/kondisi tekanan tinggi
3. Tahan terhadap lingkungan yang berbahaya

2.1.2 Struktur Robot

Struktur yang dimiliki oleh robot pada umumnya terdiri dari :

1. Manipulator

   Adalah tubuh yang membentuk robot. Bentuk-bentuk manipulator antara lain bodi, pergelangan *(wrist)* dan lengan *(arm)*. Seperti halnya yang ditunjukkan oleh Gambar 2.1.

2. *End effector*

   End effector adalah ujung dari manipulator yang akan dikontrol, seperti penjepit, bor, jarum, pemanas.

3. Penggerak *(driver)*

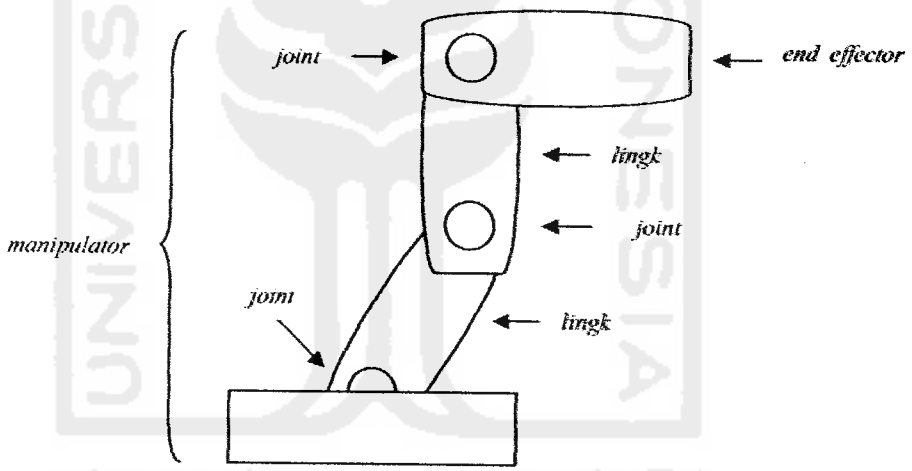
   Pengontrolan robot dapat berupa mikroprosessor, PLC, FPGA, maupun komputer dengan memanfaatkan berbagai macam bahasa pemrograman yang bisa digunakan untuk mengontrol robot seperti *Borland Delphi, Visual Basic, dan Matlab.*

4. Sensor

   Sensor biasanya digunakan untuk mengetahui parameter-parameter tertentu yang diperlukan robot agar robot bekerja sesuai dengan program. Ada beberapa sensor yang sering digunakan di dalam robot seperti sensor posisi, kecepatan, torsi, suhu.

5. Pengendali *(Controller)*

   Pengendali robot ada yang dari dalam ada pula yang dari luar. Pengendali robot yang berasal dari dalam berasal dari parameter-parameter yang dihasilkan oleh sensor-sensor yang dimiliki robot.



Gambar 2.1 Manipulator

2.1.3 Lengan Robot

Lengan robot merupakan salah satu bagian dari robot. Penggunaan dari lengan robot ini sangat banyak dan beragam dalam dunia pendidikan maupun di

dunia industri. Fungsi yang utama dari lengan robot ini adalah untuk menggantikan tangan manusia di beberapa tempat yang sekiranya tidak mungkin dijangkau oleh manusia karena merupakan daerah berbahaya atau daerah yang memiliki kondisi tertentu misalnya suhu yang panas/dingin, zat kimia yang membahayakan, daya angkut yang besar, sehingga lengan robot ini berfungsi menggantikan tangan manusia tersebut.

2.1.4 Tenaga Penggerak Robot

Terdapat beberapa macam tenaga penggerak robot, antara lain :

1. Elektris

   Penggerak elektris memanfaatkan energi listrik untuk menggerakkan berbagai jenis motor AC, DC, dan servo.

2. Hidrolis

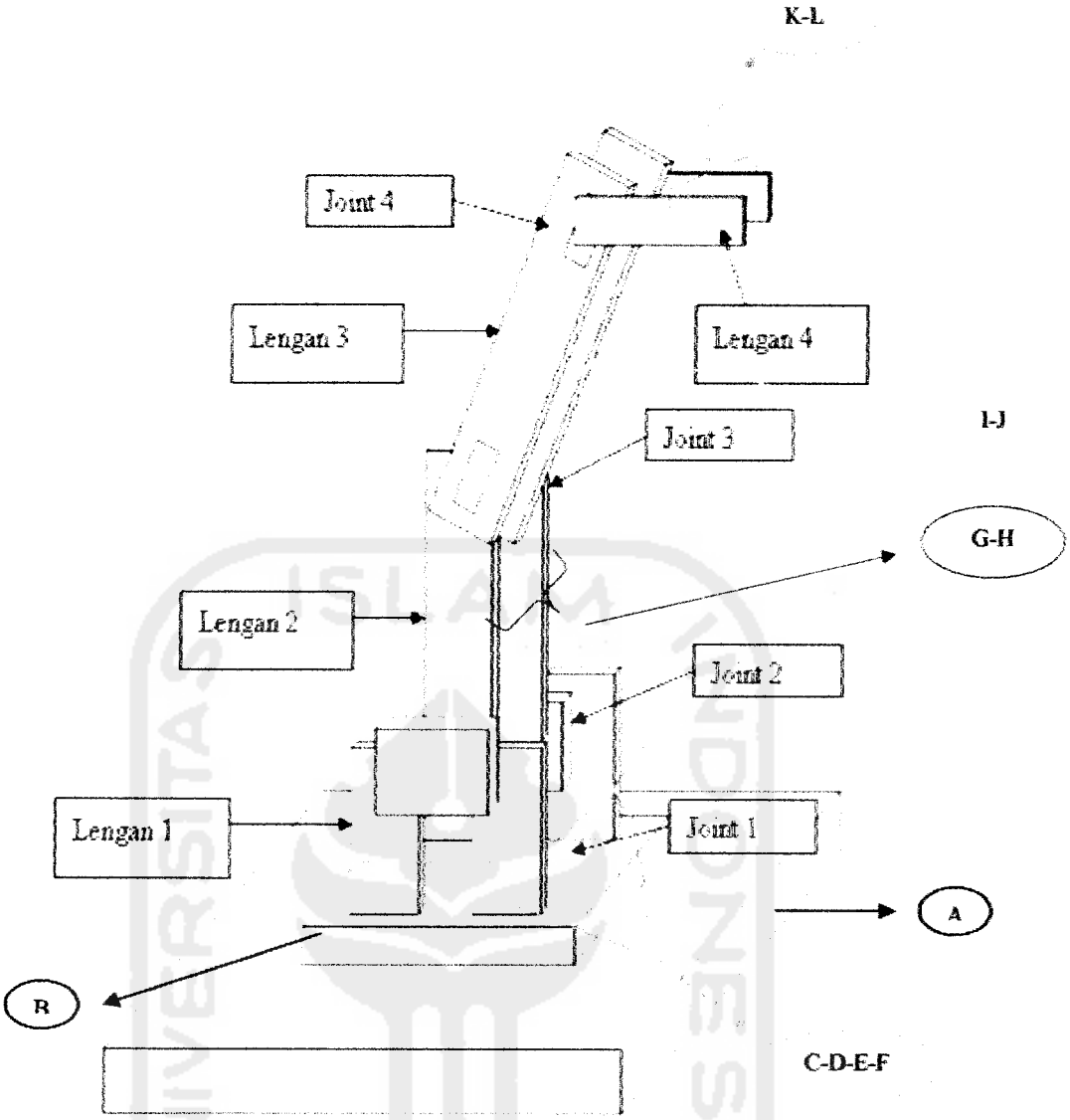
   Tenaga penggerak berupa cairan.

3. Pneumatik

   Tenaga penggerak berupa gas.

4. Mekanis

   Dapat berupa *relay* atau hanya sebagai penguat saja. Dalam prakteknya penggerak mekanis lebih sering dimanfaatkan dalam transmisi tenaga.



Gambar 2.2 Arsitektur lengan robot

Keterangan gambar :

A : Landasan dasar robot

B : Sebagai tumpuan yang menyangga robot sekaligus sebagai bagian dari lengan 1, pada B ini disusun C, D, E dan F.

G, H, I, J : Mempunyai panjang yang sama, namun berbeda penggunaannya. G dan H merupakan lengan 2, sedangkan I dan J merupakan lengan 3.

K dan L : Merupakan ujung manipulator robot

2.1.5 Pengendalian Robot

Pada umumnya robot dikendalikan pada *joint-jointnya*, sedangkan menggerakkan robot merupakan bagaimana memindahkan posisi *end effector* menuju suatu posisi mengikuti pola lintasan acuan trayektori. Data acuan tersebut dimanfaatkan untuk membangkitkan sinyal kontrol sudut *joint* (trayektori *joint*) melalui mekanisme *invers kinematic*, yaitu mekanisme untuk mendapatkan nilai-nilai sudut *joint* dari posisi dan orientasi *end effector* yang telah diketahui. Penyelesaian ini akan mentransformasikan gerakan manipulator dari *operational space* menjadi *joint space*. *Invers kinematik* dapat diselesaikan dengan banyak cara seperti *invers transform, screw algebra, dual matrix, dual quartenian, interactive* dan pendekatan geometri.

2.1.6 Kinematika Manipulator

Pada lengan robot dikenal adanya istilah kinematika manipulator. Kinematika manipulator adalah ilmu yang mempelajari gerakan manipulator, dengan mengabaikan gaya yang menyebabkan gerakan tersebut. Sedangkan manipulator merupakan sekumpulan *link* yang terhubung dengan *joint* tanpa memperhatikan gaya dan penyebabnya.

Secara umum penomoran *link* berasal dari bawah, yaitu dari paling dekat dengan *joint* utama. Pada kinematika manipulator, posisi dan orientasi harus

didefinisikan. Deskripsi posisi harus menentukan posisi titik andalan suatu sistem koordinat acuan (referensi). Diskripsi posisi juga harus menentukan vektor posisi, sedangkan orientasi harus menentukan sistem koordinat pada objek. Selain itu juga harus menyatakan sistem koordinat objek relatif terhadap titik acuan yang digunakan. Terdapat dua jenis kinematika manipulator, yaitu :

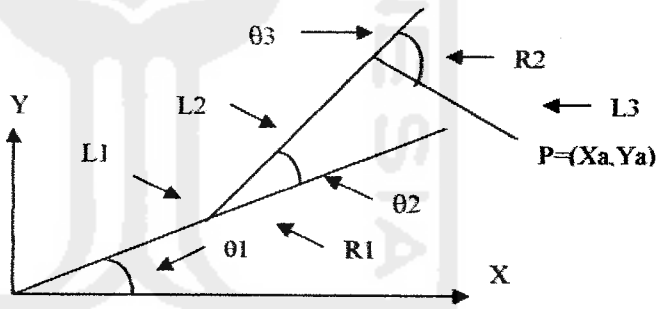
1. *Forward Kinematic*

   Yaitu masukan yang berupa sudut *joint* diubah menjadi posisi dan orientasinya, persamaan *forward kinematic* secara umum adalah :

   $R1 = (L1 \cos\theta1, L1 \sin\theta1)$ ..... (2.1)

   $R2 = (L2 \cos(\theta2 + \theta1), L2 \sin(\theta2 + \theta1))$ ..... (2.2)

   $P = (L3 \cos(\theta3 + \theta2 + \theta1), L3 \sin(\theta3 + \theta2 + \theta1))$ .....(2.3)



Gambar 2.3 *Forward Kinematic*

2. *Inverse Kinematic*

   Yaitu masukan yang berupa posisi dan orientasi menggerakkan *joint* sehingga membentuk sudut tertentu.

2.2 Mikrokontroler AT89C2051

Mikrokontroler adalah suatu terobosan teknologi mikroprosesor dan mikrokomputer, yang mana teknologi ini adalah teknologi semikonduktor dengan kandungan transistor yang lebih banyak, namun hanya membutuhkan ruang yang kecil serta dapat diproduksi secara massal (dalam jumlah banyak) sehingga harganya menjadi lebih murah.

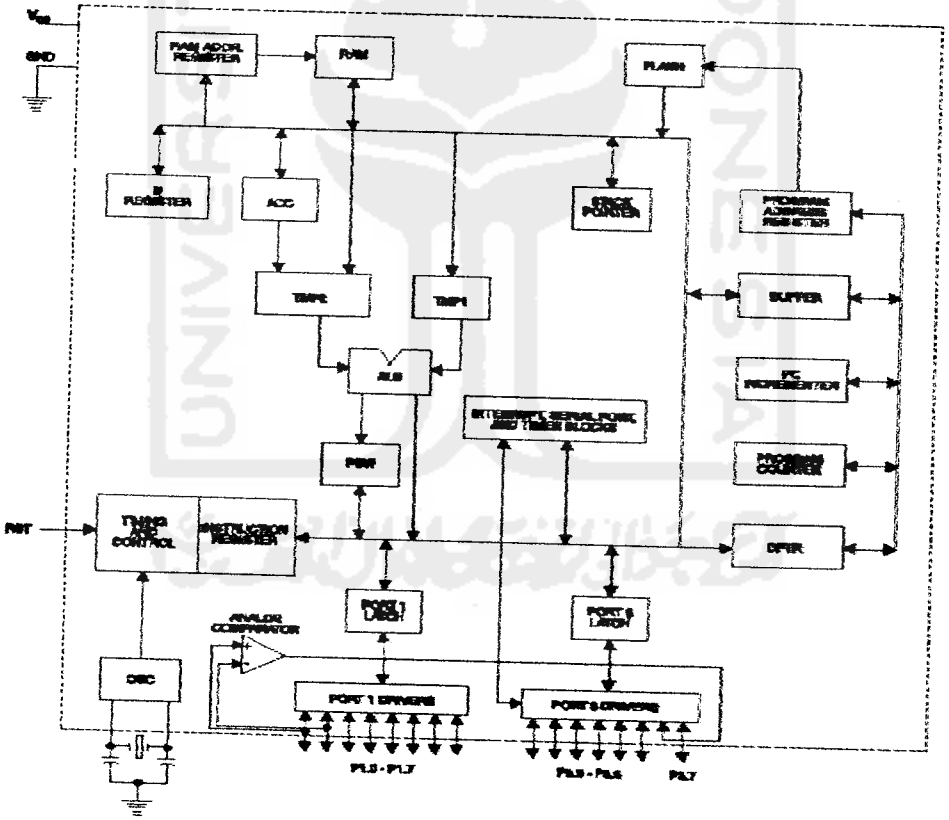
Mikrokontroler ini kemampuan digitalnya adalah menirukan fungsi otak manusia, sehingga meliputi fungsi atau instruksi aritmatika (berhitung), logika (mempertimbangkan suatu kondisi), dan memori. Mikrokontroler ini berbeda halnya dengan mikroprosesor yang hanya pemrosesannya terdiri dari *Central Processing Unit* (CPU) dan register-register, tanpa memori, tanpa I/O, dan periperal yang dibutuhkan oleh suatu sistem supaya dapat bekerja. Namun apabila mikroprosesor ini dikombinasikan dengan I/O dan memori (RAM dan ROM) akan dihasilkan sebuah mikrokontroler yang mana kombinasi dari komponen-komponen ini sudah terdapat dalam satu chip *Integrated Circuit* (IC).

Arsitektur mikrokontroler AT89C2051 merupakan sebuah mikrokomputer yang terdiri dari 8 bit dengan *low power supply* dan performa yang tinggi. Perintah-perintah dan fungsi pin-pinnya sama dengan mikrokontroler keluarga 8051 yang merupakan suatu standar industri.

Mikrokontroler AT89C2051 memiliki spesifikasi sebagai berikut :

1. Kompatibel dengan produk dan program assembler MCS-51.

2. Dapat menyimpan program sebesar 2 K Bytes Flash memory dan tahan diisi atau dihapus sampai 1000 kali.
3. Dapat bekerja pada tegangan 2,7 V sampai 6 V.
4. 128 x 8 bit internal RAM.
5. Dua buah timer/counter 16 bit.
6. 5 sumber interupsi.
7. Tersedia port serial (UART) yang dapat diprogram.
8. Terdapat fasilitas On Chip Analog Comparator (komparator tegangan analog).



Gambar 2.4 Diagram blok mikrokontroler AT89C2051

Dalam AT89C2051 selain CPU *(Central Prossesing Unit)* juga terintegrasi di dalamnya :

1. RAM *(Random Acces Memory)* sebesar 128 *bytes*. RAM merupakan tempat menyimpan sementara yang akan terhapus apabila sistem mikrokontroler dimatikan.
2. ROM *(Read Only Memory)* sebesar 2 *Kbytes*, ROM ini berisikan program-program yang akan dijalankan oleh mikrokontroler. ROM hanya dapat dibaca tidak dapat ditulis pada saat eksekusi program. Untuk menghapus program di ROM ada berbagai macam cara yang disesuaikan dengan jenis ROM tersebut, yaitu :
   a. Untuk EPROM *(Erasable Programable ROM)* dapat dihapus dengan menggunakan sinar ultra violet selama kurang lebih 15 menit, setelah itu dapat ditulis lagi program baru dengan menggunakan EPROM Programmer.
   b. Untuk EEPROM *(Electric Erasable Programable ROM)* dapat dihapus dengan memberikan tegangan 5 volt selama beberapa saat pada pin tertentu, setelah itu dapat ditulis program kembali dengan menggunakan EEPROM Programer.
3. Register pewaktu *(Timer Register)* sebanyak 2 buah yaitu timer 0 dan timer 1 yang masing-masing berkapasitas 16 bit. Register ini digunakan sebagai :

a. *Delay* atau jarak waktu sebagai contoh penggunaannya mikrokontroler memberikan waktu kepada sebuah piranti I/O yang dikontrolnya untuk bekerja selama rentang waktu tertentu. Hal ini memerlukan *delay*.

b. *Counter* atau pencacah, mikrokontroler mempunyai kemampuan untuk mencacah (menghitung) pulsa dari luar misalnya dari signal generator.

c. *Baud Rate* Serial Komunikasi yaitu tekanan transfer dapat diubah-ubah sesuai dengan kebutuhan.

4. Port Input Output AT89C2051 mempunyai 2 buah port yang dapat dikontrol sebagai I/O yaitu P1 dan P3. Sebuah port mempunyai 8 pin atau 8 bit. Meskipun semua port dapat dikontrol, masing-masing port mempunyai fungsi yang berbeda, yaitu :

   a. Port 1 adalah masukan /keluaran 8 bit dengan nama masing-masing P1.0 s/d P1.7 yang bersifat dua arah. Port 1 sudah dipasang resistor *pullup* secara internal. Jika logika satu dituliskan pada port 1 maka keluaran akan berlogika satu dan dapat digunakan sebagai masukan. Selain itu Port 1 juga memiliki fungsi lain, yaitu :

      1. Port 1 sebagai masukan byte kode program saat pemrograman dan keluaran kode program saat verifikasi
      2. Port P1.0 membutuhkan resistor *pullup* dan merupakan masukan positif (AN0) untuk komparator analog

3. Port P1.1 membutuhkan resistor *pullup* dan merupakan masukan negatif (AN1) untuk komparator analog

b. Port 3 sama dengan Port 1 yaitu masukan /keluaran 8 bit dengan nama masing-masing P3.0 s/d P3.7 yang bersifat dua arah. Port 3 sudah dipasang resistor *pullup* secara internal. Jika logika satu dituliskan pada port 3 maka keluaran akan berlogika satu dan dapat digunakan sebagai masukan. Selain itu Port 3 juga memiliki fungsi khusus seperti pada Tabel 2.1 :

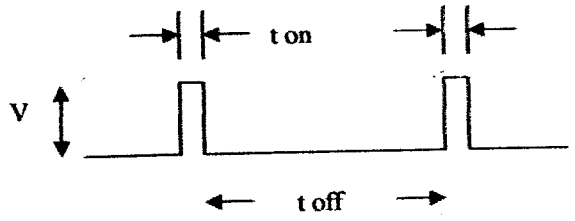
Tabel 2.1 Fungsi khusus port 3

Pin Port	Fungsi Khusus	AT89C2051
P3.0	RXD (port masukan serial)	Ada
P3.1	TXD(port keluaran serial)	Ada
P3.2	INT0(interupsi eksternal 0, aktif rendah)	Ada
P3.3	INT1(interupsi eksternal 1, aktif rendah)	Ada
P3.4	T0(masukan eksternal timer 0)	Ada
P3.5	T1(masukan eksternal timer1)	Ada
P3.6	WR(signal tulis untuk memori eksternal, aktif rendah)	Tidak ada
P3.7	RD(signal baca untuk memori eksternal, aktif rendah)	Tidak ada

5. Kontrol Interupsi, Interupsi yang dilayani oleh AT89C2051 dapat berasal dari :

a. Piranti diluar AT89C2051, untuk interupsi ini IC AT89C2051 menyediakan dua buah kontrol yaitu INT0 dan INT1 (pin P3.3 dan pin P3.2).

b. Timer register baik timer 0 dan timer 1.



Gambar 2.5 PWM

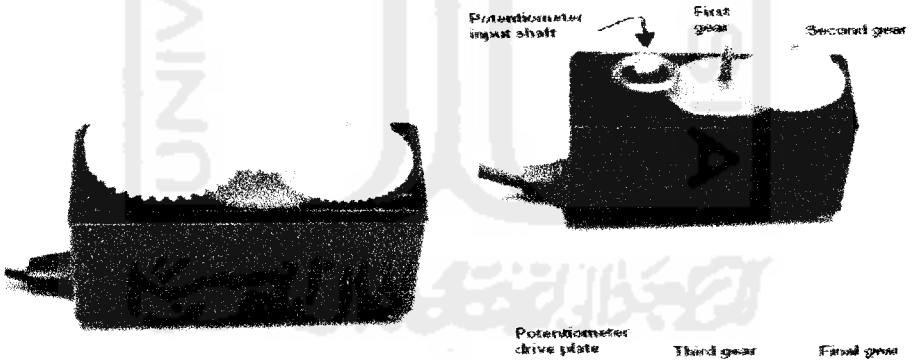
2.4 Motor Servo

Motor servo adalah motor yang memiliki 3 buah input yang dapat digerakkan dengan memberikan input berupa PWM kepada salah satu kakinya, sedangkan 2 kaki yang lain digunakan sebagai input power. Alat ini sangat bagus bila digunakan dalam sistem kendali pada mobil radio *control*, perahu, pesawat terbang model, mobil dan robot. Akan tetapi karena harganya yang mahal dibandingkan dengan motor DC biasa ataupun motor AC, maka hanya beberapa kalangan saja yang sering menggunakannya. Motor servo adalah motor khas yang disertai dengan rangkaian tambahan atau *feedback* posisi untuk menentukan posisi sumbu motor servo yang diinginkan. Rangkaian *feedback* posisi itu berupa potensiometer yang dapat berputar sesuai dengan putaran *gear* utama motor servo. Rangkaian ini senantiasa membandingkan nilai yang diberikan sesuai dengan perintah program terhadap nilai aktual potensiometer itu sendiri. Selisih atau *error* dari perbandingan kedua nilai tersebut akan dijadikan rujukan untuk menentukan arah posisinya, sehingga motor akan berputar untuk mengeleminasi *error* tersebut, sehingga akan tercapai posisi yang diinginkan. Potensiometer dari motor servo ada yang dapat diatur dari luar *cassing*, tetapi ada pula yang tidak sehingga untuk mengubah standar posisi dari

pergerakan servo, *cassing* servo harus dibuka terlebih dahulu. Setelah didapatkan nilai yang diinginkan, *cassing* servo dapat ditutup kembali.

2.4.1 Servo Standar

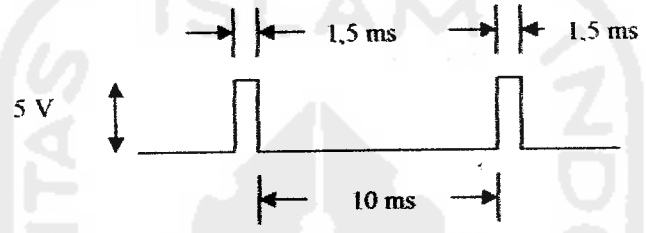
Ditinjau dari bentuk mekanik, servo dapat dibagi menjadi dua jenis yaitu yang belum dimodifikasi dan motor yang sudah dimodifikasi. Motor servo yang belum dimodifikasi hanya dapat berputar sejauh 90 derajat ke kiri nol *position* servo dan 90 derajat ke kanan nol *position* dari servo itu sendiri. Dengan kata lain, servo itu hanya dapat berputar sejauh 180 derajat sedangkan servo yang belum dimodifikasi memiliki posisi tengah (nol *position*) yang merupakan titik tengah antara awal dan akhir jangkauan motor servo. Gambar 2.5 menunjukkan bentuk dan bagian *gear* motor servo.



Gambar 2.6 Bentuk dan bagian *gear* motor servo

Servo didesain untuk menterjemahkan lebar pulsa positif (+5) antara 1-2 (ms) lebar pulsa ini merupakan pengontrol gerakan servo. Gerakan ini diatur oleh 2 keadaan yaitu keadaan 1 (+5v) dan 0 (0V). Lebar pulsa dapat bervariasi antara 10

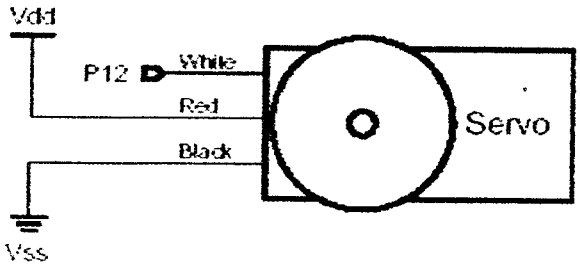
sampai 40 (ms). Pergerakan servo dibawa oleh pulsa 1 (+5). Keadaan rendah (pulsa 0) hanya akan menentukan kehalusan pergerakan dari servo itu sendiri. Jika lebar pulsa terlalu lebar, akan mengakibatkan gerakan putaran servo kurang halus. Lebar pulsa rendah antara 10 sampai 40 (ms) merupakan rekomendasi agar gerakan servo bisa halus. Seperti tampak pada Gambar 2.6 motor servo yang dibuat *parallax* dianjurkan untuk memberikan lebar pulsa 10 milidetik untuk pulsa rendah.



Gambar 2.7 Standar pulsa input

Pada servo yang belum dimodifikasi dan potensionya belum diubah jika diberikan pulsa sebesar 1,5 (ms) servo akan berada pada posisi netral. Jika lebar pulsa on (+5V) yang diberikan lebih kecil, servo akan berputar menunjukkan 0 derajat. Servo yang diberi pulsa lebih dari 1,5 (ms) akan berputar menuju ke 180 derajat.

Pada Gambar 2.7 ditunjukkan bahwa, motor servo memiliki 3 buah kabel input, yaitu : merah untuk power 5 volt, hitam untuk *ground* dan kuning untuk sinyal. Ini merupakan warna standar pada sebuah servo. Warna kabel untuk sinyal ini bisa putih ataupun kuning, kabel ini untuk sinyal pengendali servo yang dihubungkan dengan I/O mikrokontroler.



Gambar 2.8 Pengkabelan motor servo

2.4.2 Torsi Motor Servo

Torsi merupakan satuan gaya, dalam servo biasanya dinyatakan *oz/inc atau kg/cm*. Torsi yang dinyatakan dalam produk biasanya merupakan torsi angkat, bukan torsi tahan. Sehingga hal ini sangat penting dalam pemilihan servo. Pada servo buatan *Parallax* jika hanya diberi tegangan pada power dan *ground*, maka servo tidak akan bekerja. Namun pada servo buatan *Hitec*, tegangan power dan *ground* akan membuat servo diam dengan memberikan torsi tahan. Pada servo buatan *Hitec*, pulsa dapat berupa tegangan positif dari 5 volt sampai dengan 6 volt. Dengan naiknya tegangan pada pulsa yang digunakan untuk mengendalikan servo, torsi yang diberikan akan semakin besar.

2.5 *IP (Internet Protokol) Address*

Dalam mendesain sebuah jaringan komputer terutama yang terhubung dengan internet, kita perlu menentukan *IP Address* untuk setiap komputer dalam jaringan tersebut. *IP Address* (yang terdiri dari 32 bit ini) akan ditempatkan dalam *header*

setiap paket data yang dikirim oleh komputer lain, serta akan digunakan untuk menentukan rute yang harus dilalui oleh paket data.

Setiap komputer yang terhubung ke jaringan harus memiliki 1 *IP Address* dan 1 alamat *IP Address* hanya boleh dimiliki oleh satu komputer. *IP Address* merupakan bilangan biner 32 bit yang dipisahkan oleh tanda pemisah berupa tanda titik setiap 8 bit-nya dan setiap 8 bit ini disebut *oktet*. Bentuk *IP address* adalah sebagai berikut :

xxxxxxxx . xxxxxxxx . xxxxxxxx . xxxxxxxx

Setiap simbol 'x' dapat digantikan oleh angka 0 dan 1 misalnya sebagai berikut :

11000000000010100001111000000010

Notasi *IP Address* dengan bilangan seperti diatas tidaklah mudah dibaca dan ditulis. Untuk memudahkannya secara khusus dibagi ke dalam empat *oktet* (8 bit) :

11000000	**00001010**	**00011110**	**00000010**
192	10	30	2

Selanjutnya untuk memudahkan pembacaan, masing-masing *oktet* dapat diterjemahkan ke dalam bilangan desimal dengan *range* 0 sampai dengan 255 :

192.10.30.2

Jika dilihat dari bentuknya, *IP Address* terdiri atas 4 buah bilangan *oktet* (8 bit). Nilai terbesar dari bilangan biner 8 bit yaitu 255 ($= 2^7 + 2^6 + 2^5 + 2^4 + 2^3 + 2^2 + 2^1 + 1$). Jumlah keseluruhan *IP Address* adalah 255 x 255 x 255 x 255. *IP Address* sebanyak inilah yang harus dibagi-bagikan keseluruh pengguna jaringan internet diseluruh dunia.

2.6 *Borland Delphi 7.0*

Untuk menjalankan semua rangkaian pengendali lengan robot melalui jaringan ini digunakan *Borland Delphi 7*. Bahasa dari *Delphi* merupakan bahasa pemrograman yang dapat diatur sedemikian rupa sehingga dapat bekerja sama dengan piranti lain sehingga membentuk suatu sistem pengendali.

Perkembangan bahasa pemrograman berlangsung dengan pesat, mulai dari bahasa tingkat rendah (yang lebih dekat dengan *hardware*) seperti bahasa *assembly*, ataupun bahasa tingkat tinggi, seperti bahasa C dan *Pascal*. Pada bahasa ini tergolong susah mengembangkannya dan membuat tampilannya menjadi menarik, apalagi bila menggunakan bahasa pemrograman ini dalam bentuk grafik. Perkembangan bahasa pemrograman terus berlanjut dengan dikeluarkannya *Windows* oleh pihak *Microscoft*. Pada *Windows* diperkenalkan model *OOP(Object Oriented Programming)* yaitu yang lebih menampilkan kemudahan dan tampilan yang menarik (seperti *windows-windows* pada aplikasi yang sering digunakan pada *MS Word*). *Delphi* merupakan salah satu program yang berbasis pada *OOP*, jadi dengan bahasa *delphi* dapat dibuat program-program yang menarik untuk dilihat dan fleksibel serta *user friendly*.

2.6.1 Form *Borland Delphi*

Berbeda dengan *Pascal*, pada *Borland Delphi* dikenal *OOP(Object oriented programming)*. Jadi bila pada *Pascal* tampilannya menjemukan, pada *delphi* tampilannya dapat diatur semenarik mungkin pada *form* yang digunakan. Untuk mengatur tampilan *form* caranya cukup mudah, hanya dengan menaruh komponen -

komponen yang diinginkan pada *form* tersebut, dan mengfungsikan masing-masing komponen sesuai dengan yang diinginkan.

2.6.2 Unit *Borland Delphi*

Setiap perubahan pada *form* akan berakibat perubahan pada unit. Untuk pindah dari *form* ke unit, dapat dilakukan dengan menekan tombol **F12**. Berikut ini adalah bentuk unit yang diberikan delphi saat pertama kali membuka sebuah *form* :

```
Unit Unit1;
Interface

Uses
Windows, Messages, Sys Utils, Classes, Graphics, Controls, Forms, dialogs;

Type
TForm1 = class(tForm)

Private
{ private declarations }

Public
{ Public declarations }

end;
var
Form1 : TForm1;

Implementation
{SR * .DFM}
end.
```

2.6.3 Komponen *Borland Delphi*

Dalam membuat program, delphi telah menyediakan banyak kemudahan yaitu dengan disediakannya komponen-komponen. Komponen ini merupakan sebuah prosedur/program yang sudah di kompile dan langsung dapat digunakan, sesuai dengan fungsinya masing-masing. Untuk menggunakan komponen ini dengan meng-klik komponen yang diinginkan, kemudian klik di *form*, maka komponen tersebut akan muncul di *form*.

Kegunaan dari beberapa komponen tersebut adalah sebagai berikut:

a. *Button/ Bitbtn*

Komponen ini biasa digunakan sebagai tombol kendali. Perbedaan antara bitbtn dengan btn adalah : bitbtn dapat menyisipkan warna pada tombol dan *icon* tertentu, lain halnya dengan menggunakan btn.

b. *Panel*

Panel befungsi untuk mengelompokan komponen-komponen didalamnya.

c. *Label*

Kita dapat menamakan atau memberi keterangan pada program.

d. *Edit*

Edit berfungsi sebagai masukan data (*input*) dalam bentuk *string*, dari bentuk *string* dapat diubah menjadi bentuk *integer* atau bentuk lainnya. Yang kemudian dapat digunakan untuk operasi selanjutnya.

e. *Chart*

Dengan chart data-data yang telah dianalisa, dapat ditampilkan kedalam grafik, sehingga memudahkan untuk menganalisanya.

f. *Stringgrid*

Stringgrid berguna untuk menaruh data *string* kedalam bentuk kolom tabel, seperti pada Excel.

g. *Popup Menu*

Popup menu berfungsi sebagai perintah yang aktif bila meng-klik kanan *mouse*, Untuk mengaktifkannya harus mengaktifkan *popup menu* pada komponen yang diinginkan, caranya : ubah pada *object inspector*.

h. *Main menu*

Contoh *main menu* adalah *option* pada tiap aplikasi program, dengan komponen ini, fungsi-fungsi program dapat ditaruh seperti pada aplikasi umumnya.

i. *Combo Box*

Combo Box berfungsi sebagai petunjuk untuk pemilihan berbagai masukan.

j. *Check Box*

Bila komponen ini di *check* maka ada aplikasi yang bisa disetting untuk bekerja dibawahnya.

k. *Radio Button*

Prinsip kerjanya hampir sama dengan *check box*, Cuma tampilannya saja yang berbeda.

2.6.4 *Object Inspector*

Object inspector berguna sebagai menu pilihan dari masing-masing komponen. Dengan *object inspector* komponen-komponen yang digunakan dapat dimanipulasi.

2.6.5 Komponen Serial *Delphi*

Untuk mengakses data melalui port serial digunakan *Borland Delphi* salah satunya, dengan menggunakan komponen *ComPort*. Pada menu komponen standar yang digunakan pada *Borland Delphi 7* tidak terdapat komponen *ComPort* sehingga untuk dapat menggunakan komponen tersebut terlebih dahulu komponen *ComPort* diinstallkan ke *Delphi 7*. Cara menginstall komponen *ComPort* ke *Delphi 7* adalah sebagai berikut:

1. Buat nama folder baru didalam folder *Delphi 7*, kemudian ekstrak file *ComPort*. Zip ke folder baru. File *ComPort*.Zip dapat di download pada www.sourceforge.net/projects/comport
2. Klik menu komponen kemudian klik *install packages* kemudian buka file CportLib7.dpk, kemudian klik tombol *"compile"*.
3. Kemudian buka file DsgnCPort7.dpk, kemudian klik tombol *"compile"* setelah *dicompile* kemudian klik tombol *install*.

BAB III

PERANCANGAN SISTEM

3.1 Perancangan Sistem

Perancangan sistem kendali lengan robot melalui jaringan komputer dibagi menjadi 2 tahap, yaitu perancangan perangkat keras yang meliputi perancangan sistem minimal mikrokontroler dan unit konverter ke RS232 . Sedangkan perancangan perangkat lunak yaitu pembuatan program dengan bahasa C *(Borland Delphi 7)* dan program untuk mikrokontroler AT89C2051 yang akan di-*download* sistem minimal mikrokontroler.

3.1.1 Perancangan Perangkat Keras

Perancangan perangkat keras harus disesuaikan dengan proses pengendalian objek yang akan dikendalikan. Pada sistem pengendalian lengan robot ini, proses yang dikendalikan dapat digambarkan seperti pada Gambar 3.1.

Pengaturan kecepatan motor untuk bergerak dari sudut satu ke sudut yang lainnya dilakukan dengan mengubah nilai kecepatan pada program AT89C2051, dimana sebenarnya *Set Point* akan menentukan *duty cycle* sinyal *Pulse Width Modulation (PWM)* yang diterima motor, sehingga tegangan

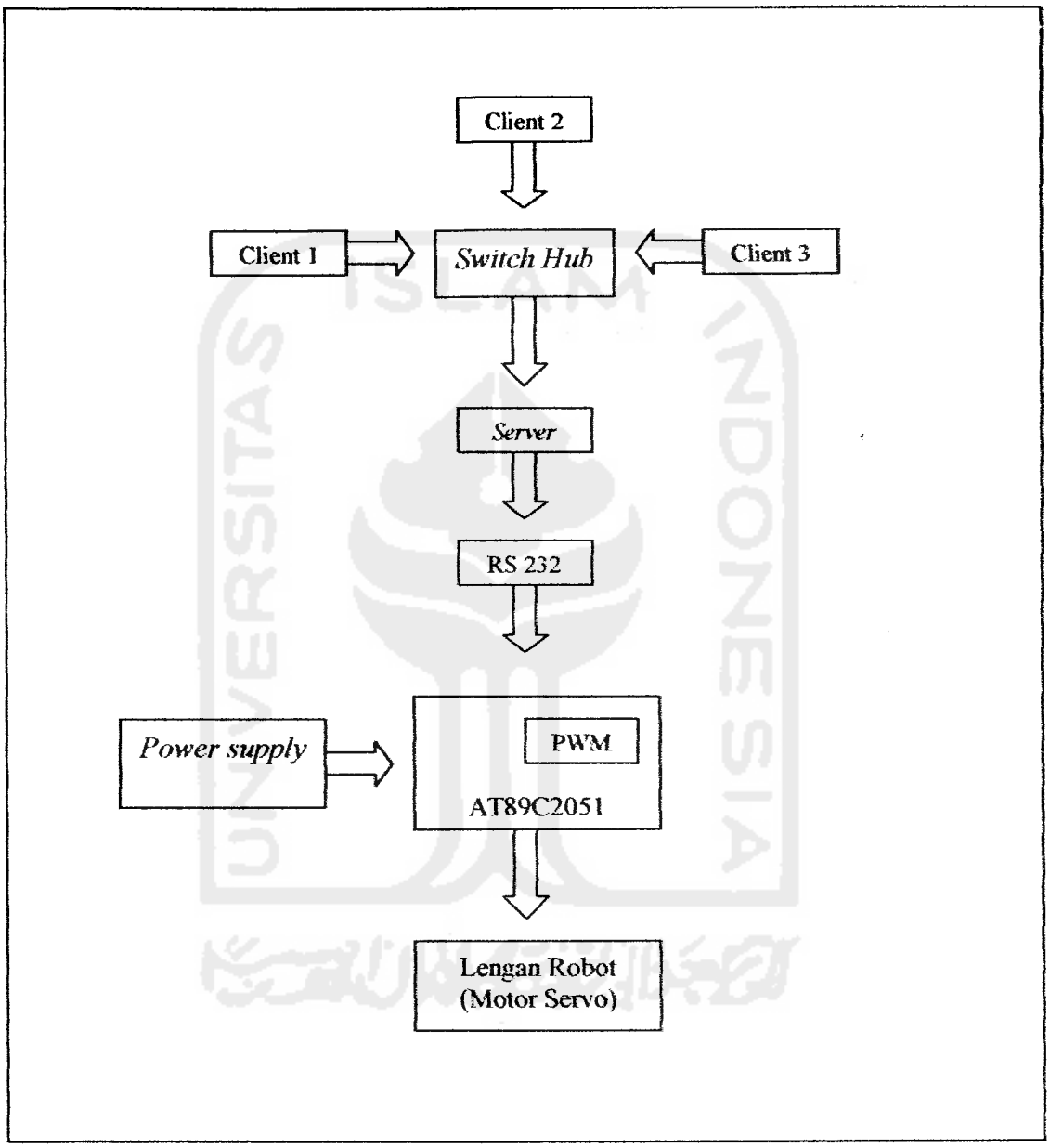
c. Port serial yaitu melalui register TI *(transmit interupt)* atau RI *(receive interupt)*.

6. Port serial berfungsi untuk komunikasi serial dengan CPU , sepasang TX *(transmiter)* dan RX *(receiver)* (pin P3.1 dan pin P3.0).
7. Jika RST diberi logika tinggi dalam waktu 2 siklus mesin maka mikrokontroler akan di*reset* serta RST berfungsi sebagai masukan tegangan pemrograman (VPP) pada saat pemrograman memori *flash internal*.
8. Oscilator ini dibangkitkan oleh kristal ataupun dari TTL *(Transitor Trasistor Logic)* luar. Semakin besar frekuensi yang dipakai oleh osilator maka semakin cepat siklus mesin dari 89C2051 berarti semakin cepat pula kemampuan 89C2051 mengeksekusi suatu program.

2.3 *PWM (Pulse Width Modulation)*

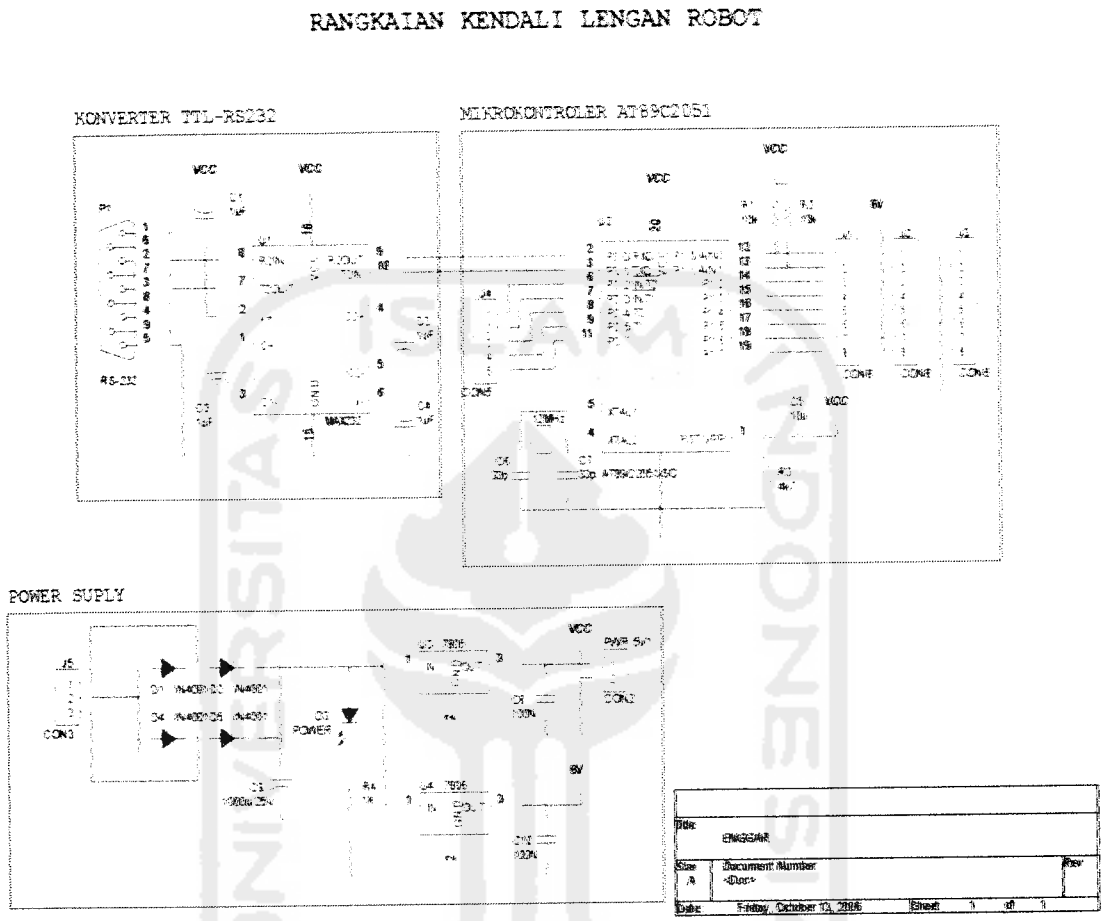
*PWM (Pulse Width Modulation)*adalah pemodulasian terhadap sebuah lebar pulsa. Modulasi yang dilakukan pada *PWM* adalah mudulasi *duty cycle*, yaitu perbandingan antara *t on* dan *t off*. Namun dalam tugas akhir ini, yang sangat diperlukan adalah *t on*. *t on* dibuat lebarnya antara 0,8 (*ms*) sampai 2,2 (*ms*) agar dapat menggerakkan motor servo yang ada dalam lengan robot. Tampak pada Gambar 2.4. V adalah tinggi pulsa, yaitu antara 3 sampai 5 volt. Besarnya arus tidak terlalu berpengaruh, sinyal *PWM* ini akan dibangkitkan dengan memanfaatkan timer yang ada dalam mikrokontroler AT89C2051.

motor akan ditentukan oleh berapa persen *duty cycle* sinyal tersebut. Dengan demikian kecepatan motor akan ditentukan sesuai dengan nilai *Set Point*.



Gambar 3.1 Diagram blok sistem kendali lengan robot melalui jaringan

Secara detail dapat digambarkan rangkaian perangkat keras dari lengan robot tersebut diatas tampak seperti pada Gambar 3.2 berikut ini :

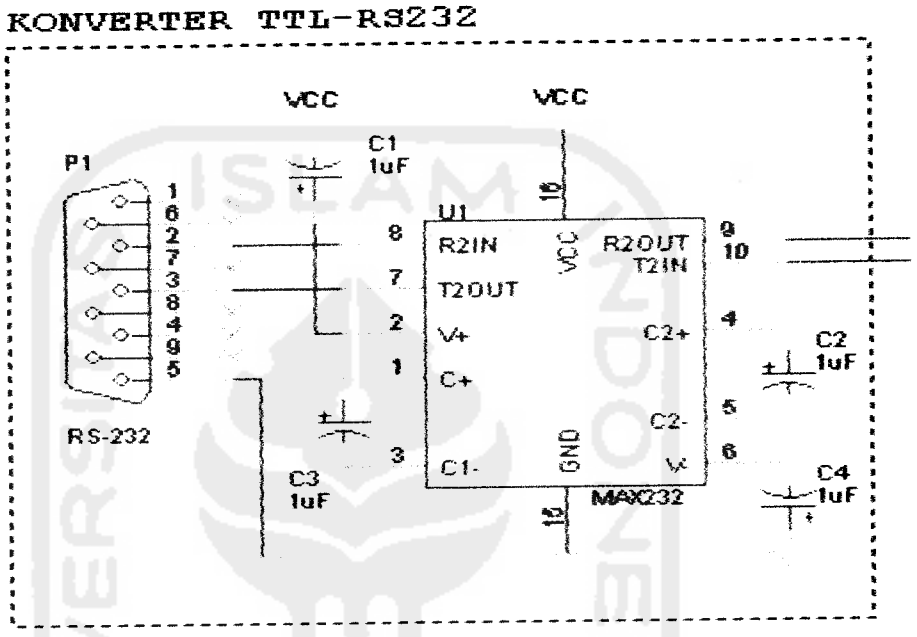


Gambar 3.2 Skematik rangkaian kendali lengan robot menggunakan AT89C2051

3.1.2 Konverter TTL - RS232

Komunikasi serial RS232 digunakan sebagai antarmuka antara komputer dengan mikrokontroler. Agar level tegangan data serial dari mikrokontroler setara dengan level tegangan komunikasi port serial *PC*, diperlukan IC MAX232 untuk mengubah ke tegangan TTL/CMOS logic level RS232. Dapat dilihat pada Gambar 3.3

yang menunjukkan rangkaian konverter TTL - RS232 dimana IC MAX232 menggunakan sistem komunikasi simplex sehingga difungsikan untuk mengubah dari aras tegangan logika TTL menjadi aras tegangan logika komputer (RS232). Lihat Gambar 3.4.



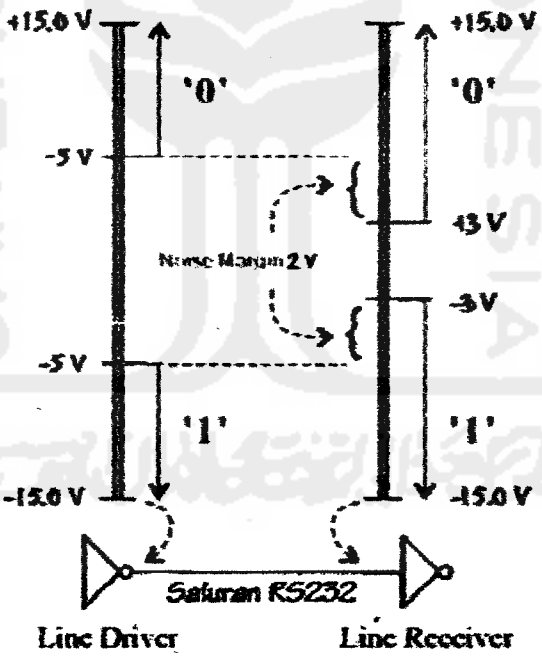
Gambar 3.3 Rangkaian konverter TTL – RS232

Saluran data yang dipakai RS232 adalah tranmisi saluran tunggal. Tranmisi saluran tunggal memakai satu utas kabel untuk mengirim satu sinyal, informasi logika ditafsirkan dari beda tegangan terhadap ground. Untuk pengiriman banyak sinyal, maka :

a. Dalam standar RS232, tegangan antara +3 sampai +15 volt pada *input line receiver* dianggap sebagai level tegangan "0", dan tegangan antara -3 sampai -15 volt dianggap sebagai level tegangan "1".

b. Agar *output line driver* dapat dihubungkan dengan baik, tegangan *output line driver* berkisar antara +5 sampai +15 volt untuk menyatakan level tegangan "0", dan berkisar antara -5 sampai -15 volt untuk menyatakan level tegangan "1".

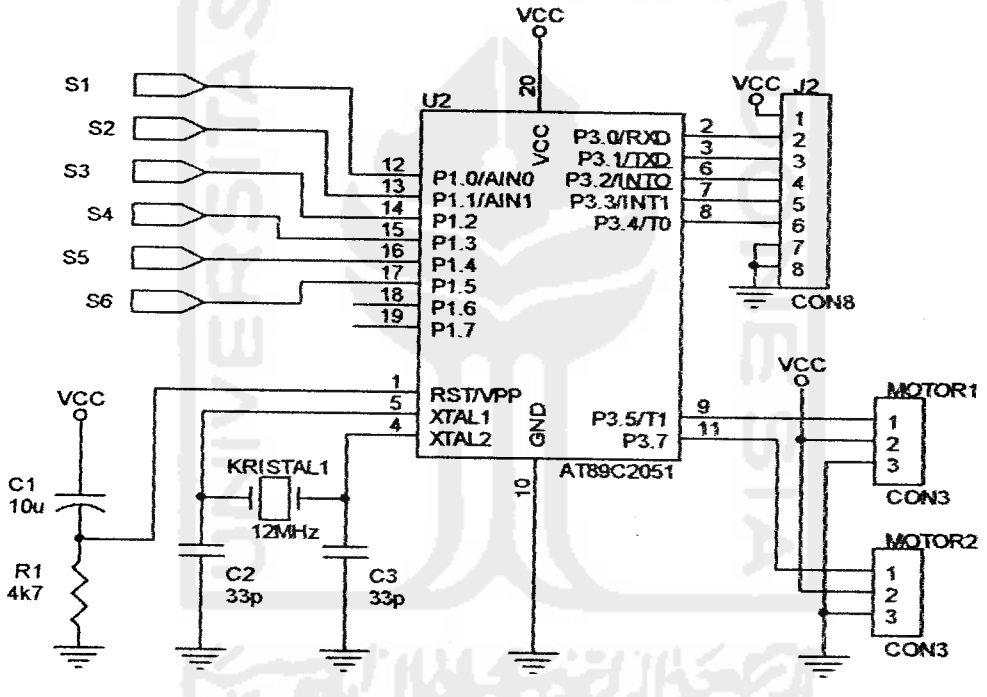
c. Beda tegangan sebesar 2 volt ini disebut sebagai *noise margin* dari RS232.



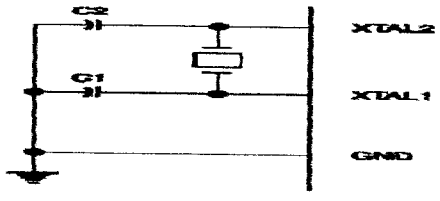
Gambar 3.4 Level tegangan RS232

3.1.3 Sistem Minimum Mikrokontroler

Pengendalian lengan robot ini dilakukan melalui sistem minimum mikrokontroler yang terdiri dari rangkaian osilator, *power on reset* dan catu daya, seperti tampak pada Gambar 3.5. Rangkaian osilator tersebut digunakan untuk membangkitkan *clock*. Rangkaian osilator seperti terlihat pada Gambar 3.6. Pada rangkaian osilator ini digunakan dua buah kapasitor 33 pF dan kristal sebesar 11,0592 MHz.



Gambar 3.5 Sistem minimum AT89C2051



Gambar 3.6 Rangkaian osilator

Sedangkan rangkaian *power on reset* berfungsi untuk menjaga agar pin RST mikrokontroler selalu berlogika rendah saat mikrokontroler mengeksekusi program. Untuk dapat mengeksekusi program dari awal program (alamat 00H) maka mikrokontroler akan di*reset* secara otomatis saat catu daya pertama kali dihidupkan dimana untuk *reset* otomatis ini dilakukan oleh *power on reset.*

Sistem minimum Mikrokontroler AT89C2051 digunakan sebagai pengolah dan sekaligus sebagai unit penyimpan program. *PWM (Pulse Width Modulation)* akan dibangkitkan melalui program dengan menggunakan mikrokontroler.

3.1.4 *Power Supply*

Sebagai pemasok tegangan digunakan transformator 2A CT, untuk membangkitkan tegangan pada RS232 dan mikrokontroler digunakan LM7805 sedangkan untuk memasok tegangan pada motor servo digunakan LM7806. Seperti tampak pada Gambar 3.7 yang merupakan rangkaian dari *power supply*

Gambar 3.7 Rangkaian *power supply*

3.2. Perancangan Perangkat Lunak

Bahasa pemrograman yang digunakan untuk pengendalian lengan robot pada tugas akhir ini yaitu bahasa C. Bahasa C merupakan merupakan bahasa *low-level* yang memungkinkan pemrogram untuk menetapkan setiap detail dalam logika algoritma untuk mencapai efisiensi komputasi yang maksimum. Namun bahasa C juga merupakan bahasa *high-level* yang dapat menyembunyikan detail arsitektur komputer sehingga meningkatkan efisiensi pemrograman.

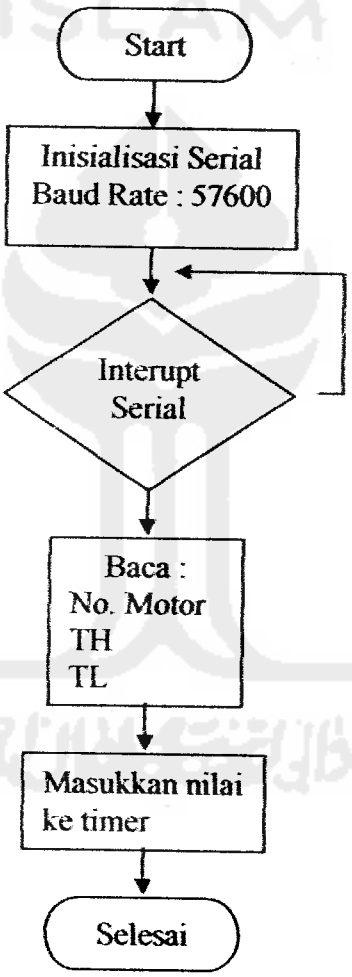
Bahasa C berada satu langkah di atas bahasa yang berorientasi pada mesin, namun tetap satu langkah di bawah sebagian besar bahasa yang berorientasi pada persoalan. Bahasa C cukup dekat dengan komputer untuk dapat memberikan kendali yang besar terhadap detail implementasi pemakaian, namun cukup jauh untuk mengabaikan detail *hardware*. Karena inilah bahasa C suatu ketika dipandang sebagai bahasa *high-level* dan pada saat yang lain dilihat sebagai bahasa *low-level*.

3.2.1 Perancangan Program Mikrokontroler AT89C2051

Program yang digunakan pada pemograman mikorokontroler ini menggunakan bahasa C dengan SDCC *(Small Device C Compiler)*. Gambar 3.8 merupakan *flowchart* dari program pengendalian lengan robot, sedangkan untuk listing programnya dapat dilihat pada halaman lampiran.

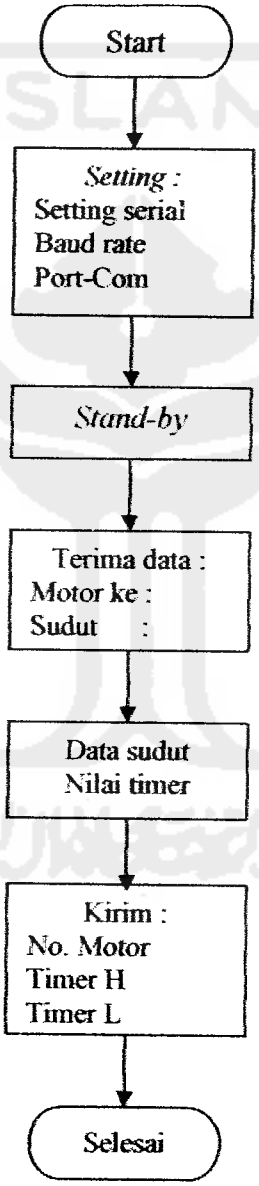
3.2.2 Perancangan Program *Borland Delphi 7*

Program yang digunakan pada pemograman *delphi* ini menggunakan *Borland delphi 7*. Gambar 3.9 merupakan *flowchart* dari program *delphi* sebagai *client*, sedangkan untuk Gambar 3.10 merupakan *flowchart* dari program delphi sebagai server dan listing program untuk *client* ataupun server dapat dilihat pada halaman lampiran.

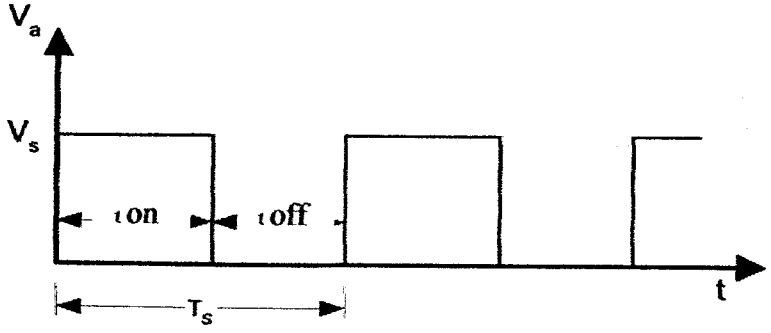


Gambar 3.8 *Flowchart* program pengendalian lengan robot pada mikrokontroler AT89C2051

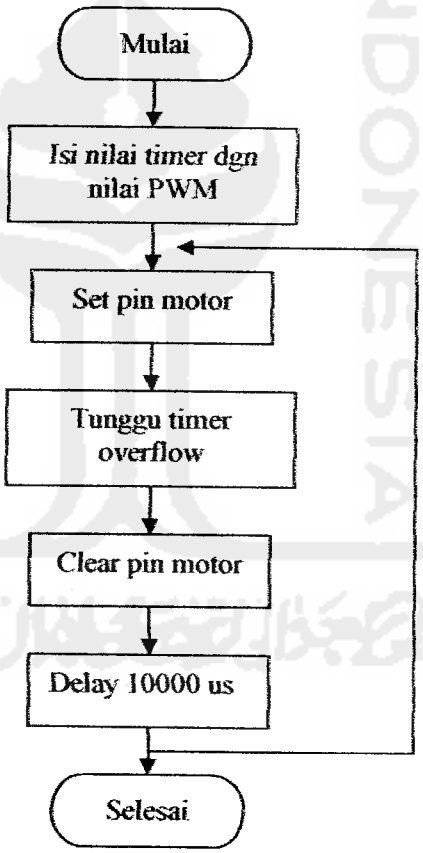
Frekuensi yang dibangkitkan *PWM* adalah 200 Hz. Sedangkan program untuk membangkitkan frekuensi 200Hz dengan menggunakan bahasa C yaitu dengan cara mengeluarkan tegangan tinggi (logika 1) di PORT A selama T_{on} µdetik, dan 0 selama 1-T_{on} µdetik. Dengan demikian akan dihasilkan *PWM* dengan *duty cycle* T_{on} x 100%. Gambar 3.12 merupakan *flowchart* dari pembuatan program *PWM*.



Gambar 3.10 *Flowchart* program *delphi* sebagai server



Gambar 3.11 Bentuk gelombang *PWM*



Gambar 3.12 *Flowchart* untuk pembuatan program *PWM*

3.2.4 Pembangkit Pulsa

Mikrokontroler AT89C2051 memiliki fasilitas 2 buah timer yang dapat diset sebagai *timer* maupun *counter*. Perangkat *timer/counter* tersebut merupakan perangkat keras yang terpadu dalam mikrokontroler AT89C2051. Untuk mengaksesnya digunakan register khusus yang tersimpan dalam *SFR (Special Function register)*. Pada *SFR* ini terdapat register *Tlx (timer low)* dan *Thx (timer high)*. *Timer* digunakan untuk membangkitkan pulsa yang bertegangan 5 volt (*on*) dan mati (*off*). *Thx* diisi nilai *high bit* dari FFFFh (65535) yang dikurangi nilai *timer* yang diinginkan, sedangkan *low bit* diisikan ke *Tlx*. Beberapa variasi *PWM* yang dibangkitkan tergantung dari masukan, sehingga didapatkan beberapa nilai *PWM* yang bervariasi antara 400 (ms) sampai 2500 (ms).

BAB IV

PENGUJIAN, ANALISA DAN PEMBAHASAN

Pada bab ini akan dibahas mengenai keseluruhan pengujian alat. Pengujian alat meliputi pengujian kemampuan lengan robot untuk bergerak, baik untuk perangkat keras ataupun perangkat lunak.

4.1 Pengujian

Hasil pengujian pengendalian *end effector* lengan robot dengan mikrokontroler pada *joint* 1 ditunjukkan pada Tabel 4.1 berikut ini :

Tabel 4.1 Hasil pengujian pada *joint* 1

No	Masukan Sudut (°)	Sudut *joint* terukur			Kesalahan $=\left\lvert\left(\frac{rata-rata\cdot error}{masukan\cdot sudut}\right)\times 100\%\right\rvert$
		Coba1/ error	Coba2 /error	Coba3/ error	
1	0	0/0	0/00	0/0	0
2	10	10/0	10/0	10/0	0
3	20	20/0	20/0	20/0	0
4	30	30/0	30/0	30/0	0
5	40	40/0	40/0	40/0	0
6	50	50/0	50/0	50/0	0
7	60	65/1	64/1	64/1	1,66
8	70	73/1	74/1	74/1	1,42
9	80	84/1	85/1	84/1	1,25
10	90	90/0	90/0	90/0	0
11	100	100/0	100/0	100/0	0
12	110	114/1	113/1	110/0	1,36
13	120	120/0	120/0	120/0	0
14	130	130/0	130/0	130/0	0
15	140	140/0	140/0	140/0	0
16	150	150/0	150/0	150/0	0
17	160	160/0	160/0	160/0	0
18	170	170/0	170/0	170/0	0
19	180	180/0	180/0	180/0	0

Pengujian pada *joint* 1 ini dengan mengabaikan *joint* 2, *joint* 3, dan *joint* 4. Diperoleh data seperti tampak pada Tabel 4.1, *error* terbesar pada *joint* 1 ini ditunjukkan pada saat sudut masukan 60° dengan *error* 1,66 %.

$$error = \left| \left(\frac{rata - rata \cdot error}{masukan \cdot sudut} \right) \times 100\% \right| \quad \text{..... (4.1)}$$

Tabel 4.2 Hasil pengujian pada *joint* 2

No	Masukan Sudut (°)	Sudut *joint* terukur			Kesalahan $= \left\| \left(\frac{rata - rata \cdot error}{masukan \cdot sudut} \right) \times 100\% \right\|$
		Coba1/ error	Coba2 /error	Coba3/ error	
1	110	110/0	110/0	110/0	0
2	100	100/0	100/0	100/0	0
3	90	90/0	90/0	90/0	0
4	80	80/0	80/0	80/0	0
5	70	70/0	70/0	70/0	0
6	60	60/0	60/0	60/0	0
7	50	50/0	50/0	50/0	0
8	40	40/0	40/0	40/0	0
9	30	30/0	30/0	30/0	0
10	20	20/0	20/0	20/0	0
11	10	10/0	10/0	10/0	0
12	0	0/0	0/0	0/0	0

Pengujian pada *joint* 2 ini dengan mengabaikan *joint* 1, *joint* 3, dan *joint* 4. Dari hasil yang diperoleh tidak terdapat kesalahan pada *joint* 2. Sudut masukan yang diberikan dari komputer *client* sama dengan sudut yang terukur, sehingga *error* = 0.

Tabel 4.3 Hasil pengujian pada *joint* 3

No	Masukan Sudut (°)	Sudut *joint* terukur			Kesalahan $= \left\| \left(\frac{rata - rata \cdot error}{masukan \cdot sudut} \right) \times 100\% \right\|$
		Coba1 /error	Coba2 /error	Coba3 /error	
1	100	100/0	100/0	100/0	0
2	90	90/0	90/0	90/0	0
3	80	80/0	80/0	80/0	0
4	70	70/0	70/0	70/0	0
5	60	60/0	60/0	60/0	0
6	50	50/0	50/0	50/0	0
7	40	40/0	40/0	40/0	0
8	30	30/0	30/0	30/0	0
9	20	20/0	20/0	20/0	0
10	10	10/0	10/0	10/0	0
11	0	0/0	0/0	0/0	0

Pengujian pada *joint* 3 ini dengan mengabaikan *joint* 1, *joint* 2, dan *joint* 4. Dari hasil yang diperoleh tidak terdapat kesalahan pada *joint* 3. Sudut masukan yang diberikan dari komputer *client* sama dengan sudut yang terukur, sehingga *error* = 0.

Hasil pengujian pada *joint* 4 dibawah ini dengan mengabaikan *joint* 1, *joint* 2, dan *joint* 3. Dari hasil yang diperoleh tidak terdapat kesalahan pada *joint* 4. Sudut masukan yang diberikan dari komputer *client* sama dengan sudut yang terukur, sehingga *error* = 0.

Pada pengujian masing-masing *joint* diatas diketahui besarnya pergerakan dari tiap-tiap *joint*. Dari data yang dihasilkan oleh *joint* 2, *joint* 3, dan *joint* 4

ketika diberi masukan yang sama dan pulsa yang sama hasilnya tidak mengalami suatu kesalahan/*error*. Perbedaan yang terjadi pada *joint* 1 dimana banyak terjadi *error*, kemungkinan karena perbedaan *setting*-an hambatan yang ada di dalam motor servo itu sendiri.

Sebenarnya keempat motor servo tersebut memiliki torsi yang berbeda-beda, motor servo Hitec HS564MG mempunyai respon yang lebih lambat dari motor servo lainnya sedangkan motor servo tersebut mempunyai torsi yang besar pula. Motor servo HS-311 dan HS-322 menggunakan PCB yang sama yaitu HS-322, namun dengan perbedaan besaran komponen yang digunakan maka dapat memberikan torsi yang berbeda-beda.

Tabel 4.4 Hasil pengujian pada *joint* 4

No	Masukan Sudut (°)	Sudut *joint* terukur			Kesalahan $= \left(\frac{rata - rata \cdot error}{masukan \cdot sudut} \right) \times 100\%$
		Coba1 /error	Coba2 /error	Coba3 /error	
1	0	0/0	0/0	0/0	0
2	10	10/0	10/0	10/0	0
3	20	20/0	20/0	20/0	0
4	30	30/0	30/0	30/0	0
5	40	40/0	40/0	40/0	0
6	50	50/0	50/0	50/0	0
7	60	60/0	60/0	60/0	0
8	70	70/0	70/0	70/0	0
9	80	80/0	80/0	80/0	0
10	90	90/0	90/0	90/0	0
11	100	100/0	100/0	100/0	0

4.2 Pengujian Lebar Pulsa

Dengan menggunakan kristal 11,0592 MHz, maka pewaktuan dalam mikrokontroler ditunjukkan pada persamaan 4.2 sebagai berikut :

$$Frekuensi = \frac{frek.kristal}{12} \quad \text{..... (4.2)}$$

$$= \frac{11,0592}{12}$$

$$= 0,921 \text{ MHz}$$

$$T = \frac{1}{f} \quad \text{..... (4.3)}$$

Sehingga dengan kristal tersebut dapat memberikan waktu 1,085069 µs. Jadi lebar pulsa sesungguhnya adalah lebar pulsa dikalikan 1,085069 µs.

Tabel 4.5 Hasil pengujian lebar pulsa pada *joint* 1

No	Lebar Pulsa (µs)	Sudut	Waktu (µs)
1	2063	0	2238,49
2	1289	90	1398,65
3	526	180	570,74

Tabel 4.6 Hasil pengujian lebar pulsa pada *joint* 2

No	Lebar Pulsa (µs)	Sudut	Waktu (µs)
1	991	0	1075,30
2	1401	45	1520,18
3	1788	90	1940,10

Tabel 4.7 Hasil pengujian lebar pulsa pada *joint* 3

No	Lebar Pulsa (µs)	Sudut	Waktu (µs)
1	1393	0	1511,50
2	2185	90	2370,87
3	2266	100	2458,76

Tabel 4.8 Hasil pengujian lebar pulsa pada *joint* 4

No	Lebar Pulsa (µs)	Sudut	Waktu (µs)
1	1410	0	1529,94
2	1032	45	1119,79
3	653	90	708,55

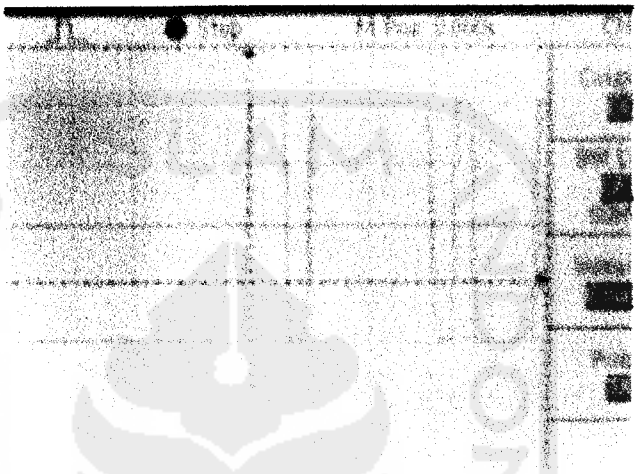
4.3 Bentuk Gelombang *PWM(Pulse Width Modulation)*

Gelombang *PWM* yang dikirim ke motor servo memiliki bentuk yang berbeda-beda tergantung masukan sudut yang diberikan oleh *client*. Gambar 4.1 menunjukkan bentuk gelombang *PWM* ketika *joint 1* diberi masukan sudut 50°.

Gambar 4.1 Bentuk gelombang PWM pada *joint 1*

ketika diberi masukan sudut 50°

Pada Gambar 4.1 pulsa hidup (t *on*), tidak sama lebarnya dengan pulsa mati (t *off*). Modulasi yang dilakukan pada *PWM* adalah modulasi *duty cycle*, yaitu perbandingan antara t *on* dan t *off*. V adalah tinggi pulsa, yaitu antara 3 sampai 5 volt. Gambar 4.2 dibawah ini menunjukkan bentuk gelombang data serial ketika diberi masukan sudut 100°.



Gambar 4.2 Bentuk gelombang data serial

Baudrate 57600 memiliki nilai *Th* 255, dimana 1 *clock* merupakan nilai awal dari nilai *Th* hingga 256, maka kecepatan kirim per-*clock* dapat dihitung seperti berikut ini :

$Th = 256 - 255$

$= 1$

1 kali kirim butuh 16 *byte data* (dalam 1 siklus ada 2 x *byte data* yang dikirim)

$$= \frac{1}{osc/12} = \frac{1}{12/osc} \quad \text{.....(4.4)}$$

$$= \frac{1}{12/11{,}0592}$$

= 1,08507 µs ⟶ (sekali kirim 1 *byte data*)

16 *byte data* = 16 x 1,08507 µs ..... (4.5)

= 17,3611 µs

Maka dalam 1 detik mampu mengirim data :

$$= \frac{1}{17{,}3611} \quad \text{..... (4.6)}$$

= 57600 kali

4.4 Instalasi Komponen Jaringan dengan Kabel *UTP*

Unshielded Twisted Pair Wire (UTP) merupakan sepasang kabel yang di-*twist*/dililit satu sama lain dengan tujuan untuk mengurangi *interferensi* listrik yang dapat terdiri dari dua, empat atau lebih pasangan kabel (umumnya yang dipakai dalam jaringan komputer terdiri dari 4 pasang kabel/8 kabel). *UTP* dapat mempunyai *transfer rate* 10 Mbps sampai dengan 100 Mbps, tetapi mempunyai jarak yang pendek yaitu maksimum 100 m.

Pada umumnya di Indonesia warna kabel yang terlilit/di-*twist* adalah (orange-putih orange), (hijau-putih hijau), (coklat-putih coklat dan biru-putih biru). Sebagai ujung dari kabel *UTP* digunakan konektor RJ-45 seperti tampak pada Gambar 4.3, penggunaan kabel *UTP* ini merupakan pilihan yang paling efisien dalam pengembangan jaringan komputer berkecepatan tinggi 10 Mbps s/d 100 Mbps.

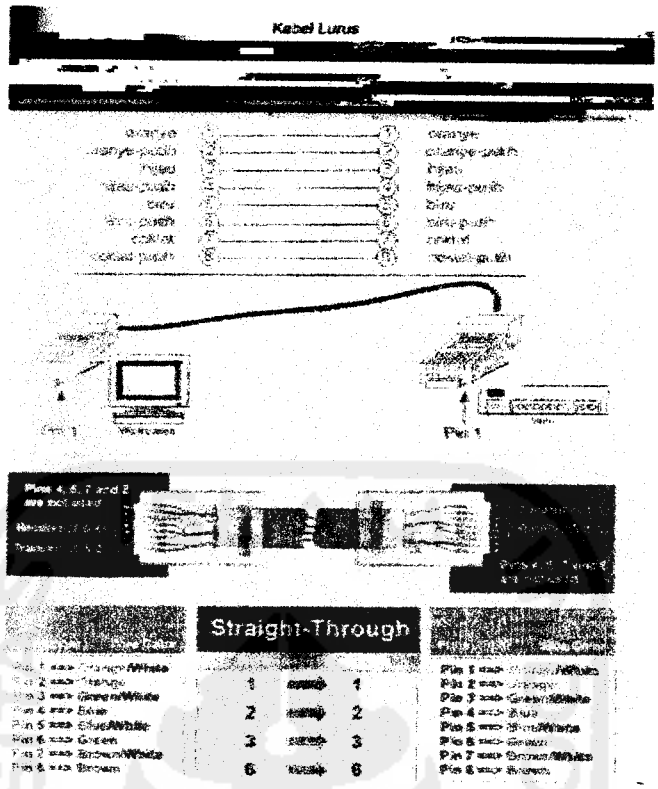
Gambar 4.3 Konektor RJ-45

Pengkabelan menggunakan kabel *UTP* memiliki 2 metode, yaitu :

1. Kabel Lurus *(Straight Cable)*
2. Kabel Silang *(Crossover Cable)*

Pada Gambar 4.4 menunjukkan pengkabelan *Straight*/lurus digunakan untuk menghubungkan antara *workstation* dengan 'hub', sedangkan kabel *Crossover*/silang digunakan untuk menghubungkan antara 'hub' dengan 'hub' atau antara dua komputer tanpa 'hub' seperti yang ditunjukkan pada Gambar 4.6.

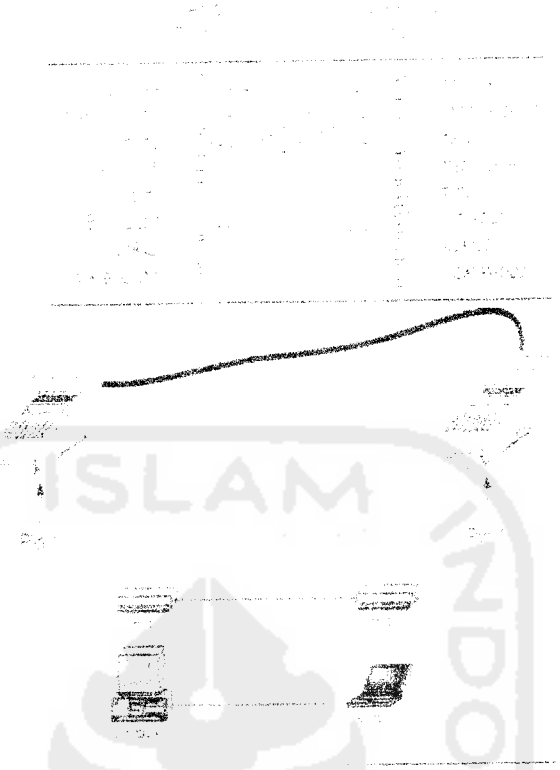
Sistem jaringan komputer yang dapat berkomunikasi satu dengan yang lainnya atau biasa disebut *Local Area Networking (LAN)* baik menggunakan 'hub' atau pun tidak, harus diperhatikan metode kombinasi penyambungan konektor *UTP*-nya. Penyambungan warna kabel tersebut yaitu dimulai dari kiri konektor ke kanan dan juga harus memperhatikan penomoran pin konektor RJ-45 untuk penyambungan dengan metode *crossover*/silang.



Gambar 4.4 Pengkabelan *UTP* dengan metode *Straight*/Lurus

Dari Gambar 4.5 terlihat bahwa sistem pengkabelan *Crossover*/silang dimaksudkan agar data yang dikirim dari komputer 1 ke komputer 2 atau dari 'hub' 1 ke 'hub' 2 dapat diterima begitu pula sebaliknya sehingga dapat berkomunikasi.

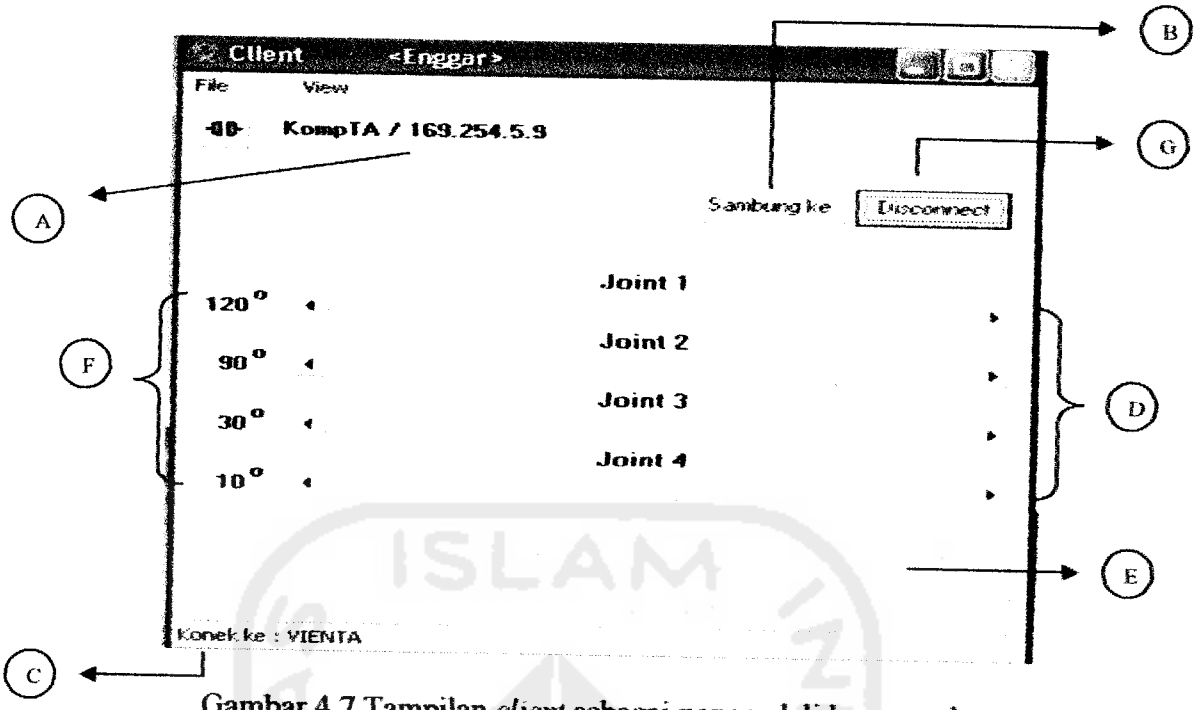
Gambar 4.5 Sistem pengkabelan *crossover*/silang



Gambar 4.6 Pengkabelan *UTP* dengan metode *crossover*/silang

4.5 *Client* – Server

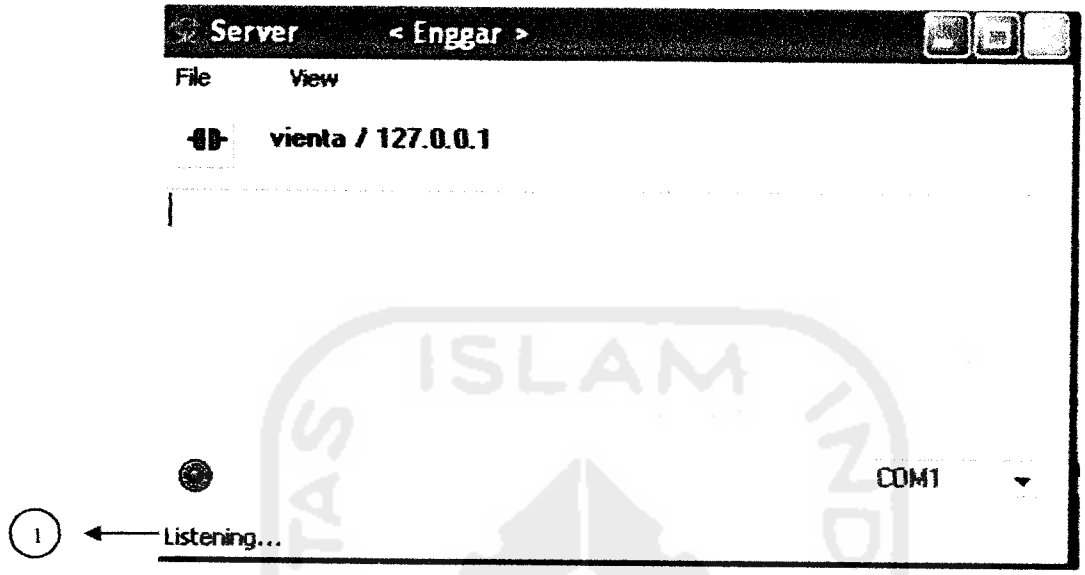
Client ini berfungsi sebagai pengendali dari lengan robot yang telah diprogram sedemikian rupa agar lengan robot dapat bergerak sesuai dengan masukan yang diberikan oleh *client*.)Saat *client* diaktifkan maka nama dan alamat dari komputer sebagai *client* akan muncul .)Masukan yang diberikan oleh *client* berupa nilai sudut ( ) yang nantinya masing-masing *joint* pada lengan robot akan bergerak sesuai dengan masukan yang diberikannya. Gambar 4.7 merupakan tampilan *client* sebagai pengendali lengan robot.



Gambar 4.7 Tampilan *client* sebagai pengendali lengan robot

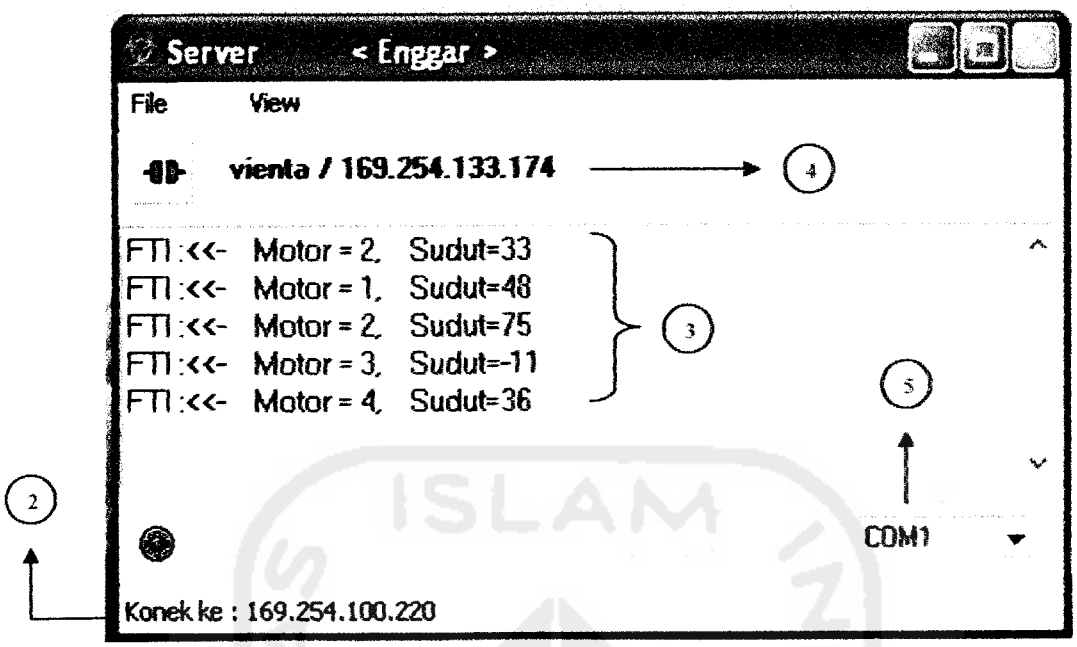
Sebelum client ini dioperasikan, maka harus disambungkan terlebih dahulu dengan server. Hal ini dimaksudkan untuk memastikan tujuan kemana kita akan mengirimkan data. [B])Dengan meng-klik *view* atau 'sambung ke' maka akan tampil nama-nama komputer yang sudah terhubung dengan *client*, selanjutnya klik nama dari komputer yang mana dalam hal ini sebagai server. [C])Secara otomatis akan terhubung dengan server dan ditunjukkan dengan munculnya tulisan " Konek ke : ..... " di sebelah kiri pojok bawah dari tampilan *client*. Sebagai contoh pada gambar diatas " Konek ke : VIENTA". [D])Ketika *scrollbar* digeser ke kanan atau ke kiri, maka nilai sudut (°) akan berubah, [E])tombol "SEND" siap ditekan apabila *scrollbar* sudah digerakkan sesuai dengan sudut yang diinginkan untuk dikirim ke server. Data yang *client* kirim akan diterima oleh server, dimana server akan selalu aktif ketika komputer

dinyalakan. [6])Merupakan tombol untuk memutuskan sambungan ke server. Berikut Gambar 4.8 yang merupakan tampilan server ketika berada pada posisi *stand-by* .



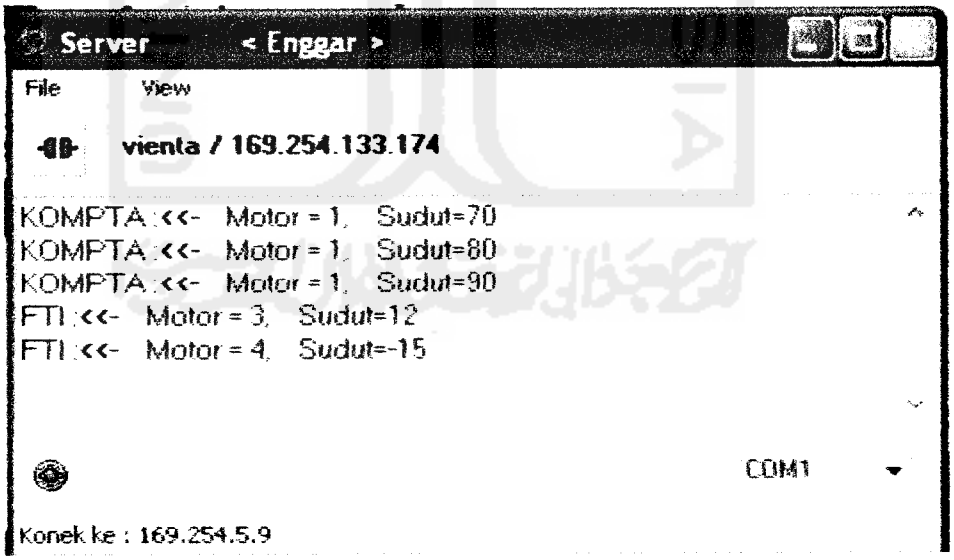
Gambar 4.8 Tampilan server ketika berada pada posisi *stand-by*

[1])Server ini akan selalu *listening* ketika tidak ada yang mengakses alamat dari server itu sendiri. [2])Pada saat alamat dari server tersebut diakses oleh *client*, maka tampilan "*listening*" akan berubah menjadi "Konek ke : 169.254.100.220", dimana angka-angka tersebut merupakan alamat dari *client* yang telah mengaksesnya. Seperti tampak pada Gambar 4.9 berikut ini, [3])Saat *client* memberikan data masukan kepada server, maka tampilan pada server tampak seperti pada gambar tersebut. [4])Tampilan alamat dan nama dari komputer sebagai server. [5]) Identifikasi port serial yang digunakan untuk mengirim data masukan dari *client* ke server pada port COM 1 ke *driver* lengan robot.



Gambar 4.9 Tampilan server ketika diakses *client* dan diberi masukan data dari *client*

Ketika server diakses oleh dua komputer atau lebih, maka tampilan dari server tampak seperti pada Gambar 4.10 berikut ini :



Gambar 4.10 Tampilan ketika server diakses oleh dua komputer *client*

Pada saat memberikan masukan sudut dari 2 komputer *client* atau lebih, server akan menerima mana yang terlebih dahulu diterima oleh server. Meskipun pengiriman sudut masukan dilakukan dalam waktu yang bersamaan, server akan tetap menerima data masukan tersebut secara berurutan mana yang terlebih dahulu diterimanya. Akan tetapi kemampuan untuk membedakan selisih waktu dari data masukan yang diberikan tidaklah mutlak selalu dapat dibedakan mana yang lebih dahulu diterima oleh server. Apabila terjadi interupsi yang bersamaan maka akan mengganggu kinerja dari sistemnya, baik sistem pada lengan robot atau pada server. Untuk menormalkan kembali dari sistem yang terganggu pada *driver* lengan robot, cukup dengan memutuskan catu daya-nya dan menyambungkannya kembali.

BAB V

PENUTUP

5.1 Kesimpulan

Berdasarkan hasil perancangan serta pengujian sistem maka dapat disimpulkan beberapa hal yaitu :

1. Pada *joint* 1 terdapat *error* ketika diberi masukan sudut 60°, 70°, 80°,110° dan *error* dengan kesalahan terbesar yakni 1,66% saat sudut masukan 60°.
2. Pada *joint* 2, *joint* 3, dan *joint* 4 tidak terjadi *error* atau dengan kata lain *error*-nya 0.
3. Ketidakpresisian hasil yang dicapai *end effector* lengan robot disebabkan ketidaktepatan posisi dan ukuran bagian-bagian lengan robot serta kekurangtelitian dalam kalibrasi dan perbedaan pulsa input juga sangat mempengaruhi keakuratan pergerakan motor servo.
4. Interupsi yang bersamaan dapat mengganggu kinerja dari sebuah sistem.
5. Pengkabelan dengan metode *crossover* dimaksudkan agar data yang dikirim oleh suatu komputer atau 'hub' dapat diterima oleh komputer lain atau 'hub' lain karena posisi dari kedua pin memiliki fungsi yang sama tiap pin-nya.

5.2 Saran

Karena sistem pengendalian lengan robot ini dinilai masih banyak kekurangannya maka untuk pengembangan selanjutnya disarankan untuk :

- Kedepan hendaknya lengan robot ini diperbaharui dengan menyempurnakan mekanik dari lengan robot itu sendiri dengan maksud agar kemampuan bergeraknya bisa lebih sempurna lagi.

DAFTAR PUSTAKA

1. Budioko, Totok, 2005, *"Belajar dengan mudah dan cepat pemograman Bahasa C dengan SDCC Pada Mikrokontroler AT89X051/AT89C51/52"*, Gava Media, Jogjakarta.
2. Malvino, Paul, Albert, Ph.D. *"Prinsip - Prinsip Elektronika"*, Erlangga, Jakarta.
3. Martina, Inge, 2002, *"36 Jam Belajar Komputer Pemrograman Internet dengan Delphi "*, Elex Media Komputindo, Jakarta
4. Putra, Afgianto Eko, 2002, *"Belajar Mikrokontroler AT89C51/52/55"*, Gava Media,Yogyakarta
5. Ruwano, Nino Guevara, 2006, *"Berkarya dengan Mikrokontroler AT89C2051"*, Elex Media Komputindo, Jakarta
6. S. Wasito, *"Data Sheet Book"*, PT Elek Media Komputindo, Jakarta.
7. Sopandi, Dede, 2004, *"Instalasi dan Konfigurai Jaringan Komputer"*, Informatika, Bandung
8. www.atmel.com
9. www.elektronikaindonesia.com
10. www.ilmukomputer.com
11. www.sourceforge.net/projects/comport

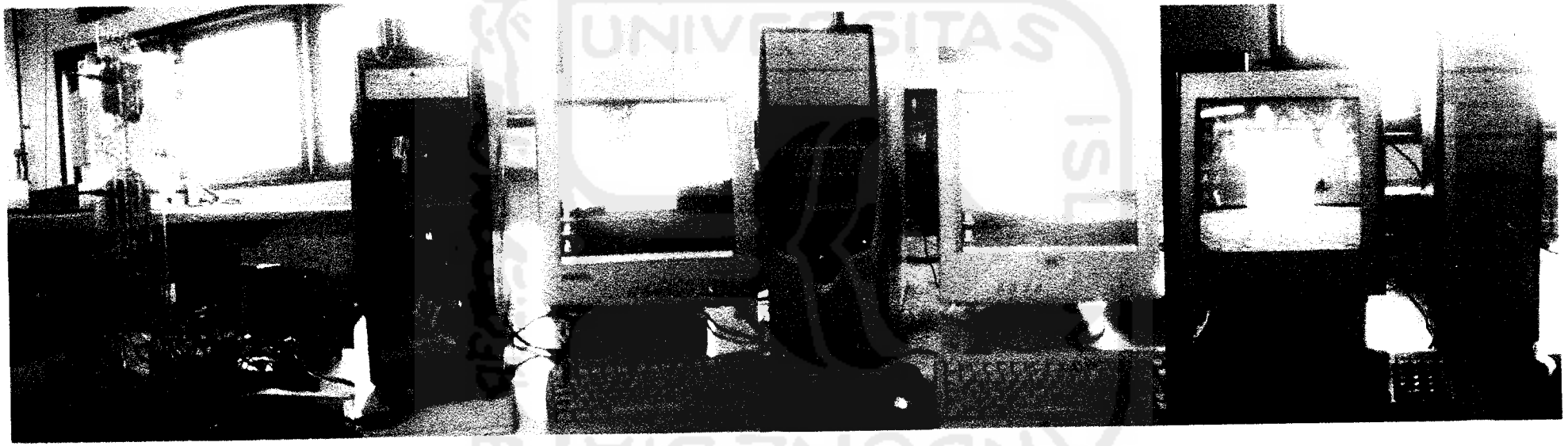
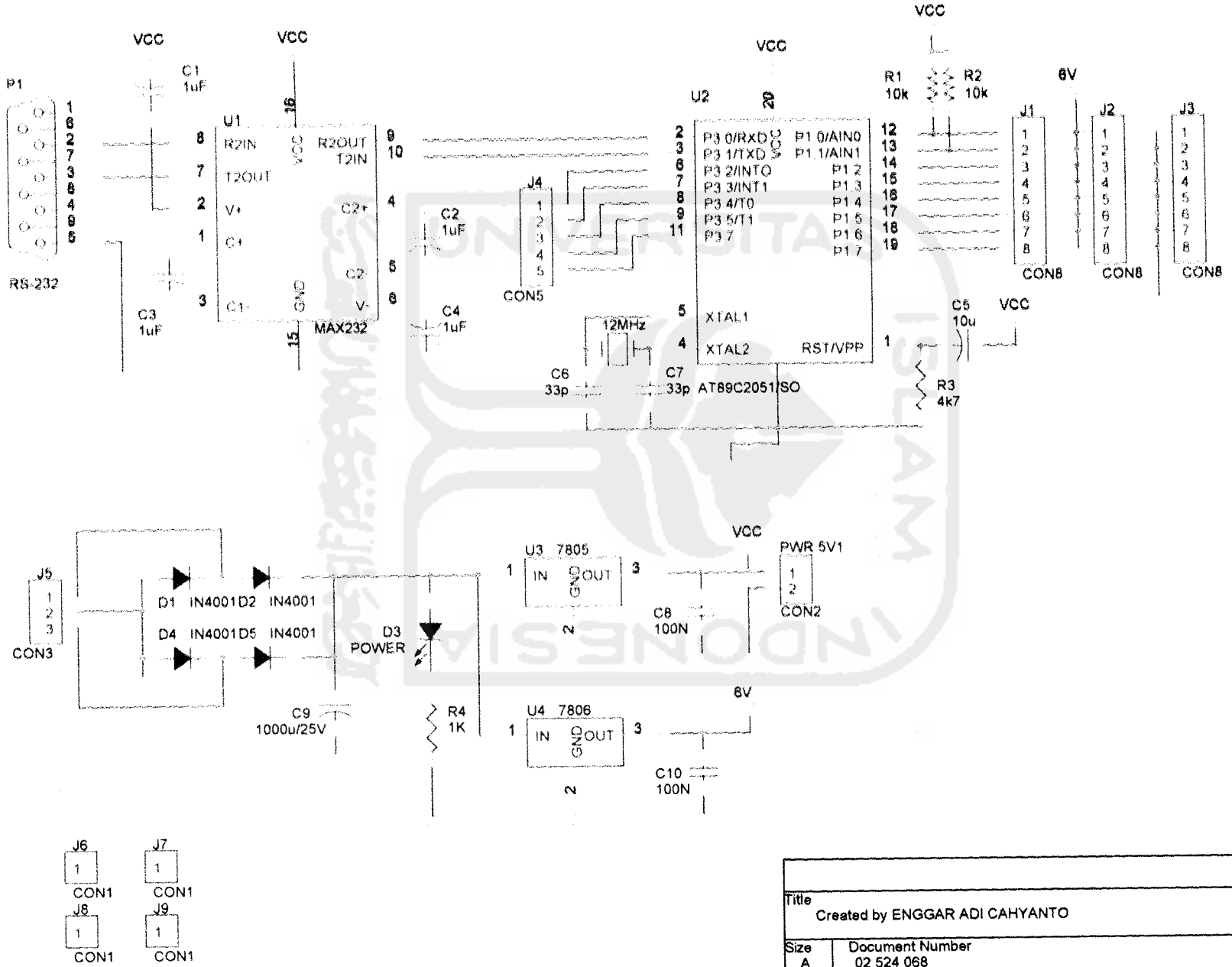


Foto Lengan Robot, Server, Hub, dan 2 Client



Title	Created by ENGGAR ADI CAHYANTO	
Size A	Document Number 02 524 068	Rev
Date: Monday, October 09, 2006	Sheet 1 of 1	

LAMPIRAN

```
//preprosisi untuk mikrokontroler keluarga MCS-51
#include<8051.h>


//Motor di Port-1
#define PortMotor  P1

//Pin-pin motor
code unsigned char Motor[4]={16,32,64,128};

//Nilai awal timer untuk posisi link 90derajat
unsigned char Thh[4]={0xfa,0xf9,0xf7,0xfd};
unsigned char Tll[4]={0xfa,0x03,0x76,0x72};

//variabel MotorNo 8-bit
unsigned char MotorNo;


                                        //fungsi tunda 1ms
void Delay(unsigned int j)
{
    int i;                              //variable 16-bit
    while(j)                            //kerjakan selama
nilai j tidak nol
    {
        for(i=0;i<140;i++);             //delay 1ms
        j--;                            //decrement j
    }
}

                                        //sub rutin untuk
membangkitkan PWM
void PutarMotor()
{
    unsigned char i;                    //variable 8-bit
positif

    for(i=0;i<4;i++)                    //ulang 4 kali untuk 4
motor
    {
        PortMotor=Motor[i];             //Set Pin motor
        TH0=Thh[i];                     //Masukan nilai timer
        TL0=Tll[i];
        TR0=1;                          //jalankan timer
        while(!TF0);                    //Tunggu sampai
selesai mencacah
        TF0=0;                          //Clear Flag timer
        PortMotor=0;                    //Clear Pin Motor
    }
}

                                        //Insialisasi
komunikasi serial
void InitSerial()
{
    TMOD=0x21;                          //Timer-1 Mode-2(UART)
dan timer-0 mode-1(16-bit timer)
    SCON = 0x52;                        //Receive enable
    PCON = 0x80;                        //Pengganda baudrate
    TH1  = 255;                         //Baudrate 57600 u/
```

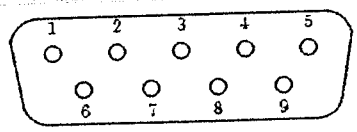
```
Fosc=11.0592MHz
    TR1  = 1;                          //Jalankan timer-1
    ES=1;                              //Enable interupsi
serial
    EA=1;                              //Enable interupsi
global
}

                                       //Fungsi untuk
menerima karakter dari komputer
char GetC()
{
    while(!RI);                        //tunggu sampai RI set
(ada karakter datang)
    RI=0;                              //Clear RI
    return(SBUF);                      //Kembalikan karakter
yang masuk di SBUF
}

                                       //Pelayanan interupsi
serial
void InterupsiSerial() interrupt 4
{
    char c;                            //variable 8-bit
    if(RI)                             //jika ada karakter
masuk
    {
        RI=0;                          //Clear RI
        c=SBUF;                        //Pindahkan karakter
yang masuk di SBUF ke variable c
        if(c<4)                        //jika nilai c kurang
dari 4
        {
            Thh[c]=GetC();             //ambil karakter
selanjutnya (Nilai Timer High)
            Tll[c]=GetC();             //ambil karakter
selanjutnya (Nilai Timer Low)
            MotorNo=c;                 //MotorNo adalah nilai
c
        }
    }
    TI=0;
}

//Fungsi utama
void main()
{
    InitSerial();                      //panggil subrutin
InitSerial();

    while(1)                           //Kerjakan terus
menerus
    {
        PutarMotor();                  //Bentuk PWM tiap
motor (Ton)
        Delay(10);                     //Pulsa rendah (Toff),
10ms
    }
}
```



The pins and signals associated with the 9-pin connector are described below. Refer to the RS-232 standard for a description of the signals and pin assignments used for a 25-pin connector.

Table 10-1: Serial Port Pin and Signal Assignments

Pin	Label	Signal Name	Signal Type
1	CD	Carrier Detect	Control
2	RD	Received Data	Data
3	TD	Transmitted Data	Data
4	DTR	Data Terminal Ready	Control
5	GND	Signal Ground	Ground
6	DSR	Data Set Ready	Control
7	RTS	Request to Send	Control
8	CTS	Clear to Send	Control
9	RI	Ring Indicator	Control

The term "data set" is synonymous with "modem" or "device," while the term "data terminal" is synonymous with "computer."

> **Note** The serial port pin and signal assignments are with respect to the DTE. For example, data is transmitted from the TD pin of the DTE to the RD pin of the DCE.

Signal States

Signals can be in either an *active* state or an *inactive* state. An active state corresponds to the binary value 1, while an inactive state corresponds to the binary value 0. An active signal state is often described as *logic 1*, *on*, *true*, or a *mark*. An inactive signal state is often described as *logic 0*, *off*,

```
 < CLIENT >
unit main;

interface

uses
  Windows, winsock, Messages, SysUtils, Classes, Graphics,
Controls, Forms, Dialogs,
  Menus, StdCtrls, Buttons, ScktComp, ExtCtrls, ComCtrls,
  CPort, CPortCtl;

type
  TChatForm = class(TForm)
    MainMenu1: TMainMenu;
    File1: TMenuItem;
    Exit1: TMenuItem;
    FileConnectItem: TMenuItem;
    FileListenItem: TMenuItem;
    Bevel1: TBevel;
    Panel1: TPanel;
    N1: TMenuItem;
    SpeedButton1: TSpeedButton;
    Disconnect1: TMenuItem;
    ClientSocket: TClientSocket;
    Panel2: TPanel;
    Button4: TButton;
    Button3: TButton;
    Label1: TLabel;
    View1: TMenuItem;
    AllKomputer1: TMenuItem;
    ComPort1: TComPort;
    Panel3: TPanel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    StatusBar1: TStatusBar;
    ScrollBar1: TScrollBar;
    ScrollBar2: TScrollBar;
    ScrollBar3: TScrollBar;
    ScrollBar4: TScrollBar;
    Button1: TButton;
    Label10: TLabel;
    Label11: TLabel;
    Label12: TLabel;
    Label13: TLabel;
    procedure FileConnectItemClick(Sender: TObject);
    procedure Exit1Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure ServerSocketError(Sender: TObject; Number:
Smallint;
      var Description: string; Scode: Integer; const Source,
      HelpFile: string; HelpContext: Integer; var
CancelDisplay: Wordbool);
    procedure Disconnect1Click(Sender: TObject);
    procedure ClientSocketConnect(Sender: TObject;
```

```
      Socket: TCustomWinSocket);
    procedure ServerSocketAccept(Sender: TObject;
      Socket: TCustomWinSocket);
    procedure ServerSocketClientDisconnect(Sender: TObject;
      Socket: TCustomWinSocket);
    procedure Button3Click(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure StatusBar1Resize(Sender: TObject);
    procedure AllKomputer1Click(Sender: TObject);
    procedure ScrollBar1Change(Sender: TObject);
    procedure ScrollBar2Change(Sender: TObject);
    procedure ScrollBar3Change(Sender: TObject);
    procedure ScrollBar4Change(Sender: TObject);
    procedure Button1Click(Sender: TObject);
  protected
    IsServer: Boolean;
  end;

var
  ChatForm: TChatForm;
  Server: String;
  Host, IP, Err: string;
  NilaiTimer: integer;
  TimerH,TimerL:byte;
  FlagMotor1,FlagMotor2,FlagMotor3,FlagMotor4:byte;

implementation

uses Unit1;

{$R *.DFM}
// Fungsi buat dapet nama & Ip kompi kita
function GetIPFromHost
(var HostName, IPaddr, WSAErr: string): Boolean;
type
  Name = array[0..100] of Char;
  PName = ^Name;
var
  HEnt: pHostEnt;
  HName: PName;
  WSAData: TWSAData;
  i: Integer;
begin
  Result := False;
  if WSAStartup($0101, WSAData) <> 0 then begin
    WSAErr := 'Winsock is not responding."';
    Exit;
  end;
  IPaddr := '';
  New(HName);
  if GetHostName(HName^, SizeOf(Name)) = 0 then
  begin
    HostName := StrPas(HName^);
    HEnt := GetHostByName(HName^);
    for i := 0 to HEnt^.h_length - 1 do
     IPaddr :=
      Concat(IPaddr,
      IntToStr(Ord(HEnt^.h_addr_list^[i])) + '.');
    SetLength(IPaddr, Length(IPaddr) - 1);
    Result := True;
```

```
  end
  else begin
   case WSAGetLastError of
    WSANOTINITIALISED:WSAErr:='WSANotInitialised';
    WSAENETDOWN       :WSAErr:='WSAENetDown';
    WSAEINPROGRESS    :WSAErr:='WSAEInProgress';
   end;
  end;
  Dispose(HName);
  WSACleanup;
end;
//---------------------------------------------------------


procedure TChatForm.FileConnectItemClick(Sender: TObject);
begin
  Button3.Click;
end;

procedure TChatForm.Exit1Click(Sender: TObject);
begin
  ClientSocket.Close;
  Close;
end;

procedure TChatForm.FormCreate(Sender: TObject);
begin
  //FileListenItemClick(nil);
  if GetIPFromHost(Host, IP, Err) then
    Label1.Caption := Host + ' / ' + IP
  else
    MessageDlg(Err, mtError, [mbOk], 0);
end;

procedure TChatForm.ServerSocketError(Sender: TObject; Number:
Smallint;
  var Description: string; Scode: Integer; const Source,
HelpFile: string;
  HelpContext: Integer; var CancelDisplay: Wordbool);
begin
  ShowMessage(Description);
end;

procedure TChatForm.Disconnect1Click(Sender: TObject);
begin
  Button4.Click;
end;

procedure TChatForm.ClientSocketConnect(Sender: TObject;
  Socket: TCustomWinSocket);
begin
  Statusbar1.Panels[0].Text := 'Konek ke : ' +
Socket.RemoteHost;
end;

procedure TChatForm.ServerSocketAccept(Sender: TObject;
  Socket: TCustomWinSocket);
begin
  IsServer := True;
  Statusbar1.Panels[0].Text := 'Konek ke : ' +
```

```
Socket.RemoteAddress;
end;

procedure TChatForm.ServerSocketClientDisconnect(Sender:
TObject;
  Socket: TCustomWinSocket);
begin
  Statusbar1.Panels[0].Text := 'Listening...';
end;

procedure TChatForm.Button3Click(Sender: TObject);
begin
  if ClientSocket.Active then ClientSocket.Active := False;
  if InputQuery('Alamat Komputer', 'IP / Nama Komputer',
Server) then
    if Length(Server) > 0 then
      with ClientSocket do
      begin
        Host := Server;
        Active := True;
        FileListenItem.Checked := False;
      end;
end;

procedure TChatForm.Button4Click(Sender: TObject);
begin
  ClientSocket.Active := False;
  Statusbar1.Panels[0].Text := 'Disconnect...';
end;

procedure TChatForm.StatusBar1Resize(Sender: TObject);
begin
  StatusBar1.Anchors:=[akBottom];
end;

procedure TChatForm.AllKomputer1Click(Sender: TObject);
begin
  Form1.Show;
end;

procedure TChatForm.ScrollBar1Change(Sender: TObject);
begin
    FlagMotor1:=1;
    button1.Enabled:=true;
    Label2.Caption:=floattoStr(scrollbar1.Position);
end;

procedure TChatForm.ScrollBar2Change(Sender: TObject);
begin
    FlagMotor2:=1;
    button1.Enabled:=true;
    Label3.Caption:=floattoStr(scrollbar2.Position);
end;

procedure TChatForm.ScrollBar3Change(Sender: TObject);
begin
    FlagMotor3:=1;
    button1.Enabled:=true;
    Label4.Caption:=floattoStr(scrollbar3.Position);
end;
```

```
procedure TChatForm.ScrollBar4Change(Sender: TObject);
begin
     FlagMotor4:=1;
     button1.Enabled:=true;
     Label5.Caption:=floattoStr(scrollbar4.Position);
end;


procedure TChatForm.Button1Click(Sender: TObject);
var val,delay:integer;
begin
  delay:=500;
  if FlagMotor1=1 then
  begin
    val:=scrollbar1.Position;
    val:=val+1000;
    ClientSocket.Socket.SendText(floattoStr(val));
    FlagMotor1:=0;
    sleep(delay);
  end;

  if FlagMotor2=1 then
  begin
    val:=scrollbar2.Position;
    val:=val+2000;
    ClientSocket.Socket.SendText(floattoStr(val));
    FlagMotor2:=0;
    sleep(delay);
  end;

  if FlagMotor3=1 then
  begin
    val:=scrollbar3.Position;
    val:=val+3000;
    ClientSocket.Socket.SendText(floattoStr(val+90));
    FlagMotor3:=0;
    sleep(delay);
  end;

  if FlagMotor4=1 then
  begin
    val:=scrollbar4.Position;
    val:=val+4000;
    ClientSocket.Socket.SendText(floattoStr(val+90));
    FlagMotor4:=0;
    sleep(delay);
  end;
  button1.Enabled:=false;

//ClientSocket.Socket.SendText(floattoStr(scrollbar1.Position+
1000));
end;

end.
```

```
< SERVER >
unit main;

interface

uses
  Windows, winsock, Messages, SysUtils, Classes, Graphics,
Controls, Forms, Dialogs,
  Menus, StdCtrls, Buttons, ScktComp, ExtCtrls, ComCtrls,
  CPort, CPortCtl;

type
  TChatForm = class(TForm)
    MainMenu1: TMainMenu;
    File1: TMenuItem;
    Exit1: TMenuItem;
    FileListenItem: TMenuItem;
    Bevel1: TBevel;
    Panel1: TPanel;
    Memo2: TMemo;
    N1: TMenuItem;
    SpeedButton1: TSpeedButton;
    Disconnect1: TMenuItem;
    ServerSocket: TServerSocket;
    Label1: TLabel;
    View1: TMenuItem;
    AllKomputer1: TMenuItem;
    ComPort1: TComPort;
    Panel3: TPanel;
    ScrollBar1: TScrollBar;
    ScrollBar2: TScrollBar;
    ScrollBar3: TScrollBar;
    ScrollBar4: TScrollBar;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    btSend: TButton;
    ComLed1: TComLed;
    Button2: TButton;
    cb1: TComboBox;
    StatusBar1: TStatusBar;
    procedure FileListenItemClick(Sender: TObject);
    procedure Exit1Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure ServerSocketError(Sender: TObject; Number:
Smallint;
      var Description: string; Scode: Integer; const Source,
      HelpFile: string; HelpContext: Integer; var
CancelDisplay: Wordbool);
    procedure ClientSocketConnect(Sender: TObject;
      Socket: TCustomWinSocket);
    procedure ServerSocketClientRead(Sender: TObject;
      Socket: TCustomWinSocket);
    procedure ServerSocketAccept(Sender: TObject;
      Socket: TCustomWinSocket);
```

```
    procedure ServerSocketClientConnect(Sender: TObject;
      Socket: TCustomWinSocket);
    procedure ClientSocketDisconnect(Sender: TObject;
      Socket: TCustomWinSocket);
    procedure ServerSocketClientDisconnect(Sender: TObject;
      Socket: TCustomWinSocket);
    procedure Button4Click(Sender: TObject);
    procedure StatusBar1Resize(Sender: TObject);
    procedure AllKomputer1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
  protected
    IsServer: Boolean;
  end;

var
  ChatForm: TChatForm;
  Server: String;
  Host, IP, Err: string;
  NilaiTimer: integer;
  TimerH,TimerL:byte;
  tabelmotor1 :array[1..181] of integer = (
2063,   2053,   2044,   2034,   2025,   2017,   2009,   2000,
1993,   1985,
1977,   1969,   1962,   1954,   1946,   1939,   1931,   1923,
 1915,   1907,
1900,   1891,   1883,   1875,   1866,   1858,   1849,   1840,
1831,   1822,
1814,   1804,   1795,   1786,   1776,   1767,   1758,   1748,
1738,   1729,
1711,   1710,   1700,   1690,   1681,   1671,   1661,   1651,
1642,   1632,
1633,   1613,   1604,   1594,   1585,   1575,   1566,   1557,
1548,   1538,
1530,   1520,   1511,   1503,   1494,   1485,   1476,   1468,
1459,   1451,
1436,   1434,   1426,   1417,   1409,   1401,   1393,   1385,
1377,   1369,
1358,   1353,   1345,   1337,   1329,   1322,   1314,   1306,
1298,   1290,
1289,   1275,   1267,   1259,   1251,   1243,   1235,   1227,
1219,   1212,
1203,   1195,   1187,   1179,   1171,   1163,   1155,   1146,
 1138,   1130,
1117,   1113,   1104,   1096,   1087,   1079,   1070,   1061,
1053,   1044,
1040,   1026,   1018,   1009,   1000,    991,    983,    974,
 965,    956,
 946,    939,    930,    921,    913,    904,    896,    887,
  879,   871,
 860,    854,    846,    838,    830,    822,    814,    806,
 798,    790,
 782,    775,    768,    760,    753,    745,    738,    731,
 723,    716,
 713,    701,    694,    687,    679,    671,    664,    656,
 648,    640,
 628,    622,    613,    604,    594,    584,    573,    562,
 551,    538,
 526);
```

```
implementation

uses Unit1;

{$R *.DFM}
// Fungsi buat dapet nama & Ip kompi kita
function GetIPFromHost
(var HostName, IPaddr, WSAErr: string): Boolean;
type
  Name = array[0..100] of Char;
  PName = ^Name;
var
  HEnt: pHostEnt;
  HName: PName;
  WSAData: TWSAData;
  i: Integer;
begin
  Result := False;
  if WSAStartup($0101, WSAData) <> 0 then begin
    WSAErr := 'Winsock is not responding."';
    Exit;
  end;
  IPaddr := '';
  New(HName);
  if GetHostName(HName^, SizeOf(Name)) = 0 then
  begin
    HostName := StrPas(HName^);
    HEnt := GetHostByName(HName^);
    for i := 0 to HEnt^.h_length - 1 do
     IPaddr :=
      Concat(IPaddr,
      IntToStr(Ord(HEnt^.h_addr_list^[i])) + '.');
    SetLength(IPaddr, Length(IPaddr) - 1);
    Result := True;
  end
  else begin
   case WSAGetLastError of
    WSANOTINITIALISED:WSAErr:='WSANotInitialised';
    WSAENETDOWN       :WSAErr:='WSAENetDown';
    WSAEINPROGRESS    :WSAErr:='WSAEInProgress';
   end;
  end;
  Dispose(HName);
  WSACleanup;
end;
//---------------------------------------------------------


procedure TChatForm.FileListenItemClick(Sender: TObject);
begin
  FileListenItem.Checked := not FileListenItem.Checked;
  if FileListenItem.Checked then
  begin
    ServerSocket.Active := True;
    Statusbar1.Panels[0].Text := 'Listening...';
  end
  else
  begin
    if ServerSocket.Active then
      ServerSocket.Active := False;
```

```
    Statusbar1.Panels[0].Text := '';
  end;
end;

procedure TChatForm.Exit1Click(Sender: TObject);
begin
  ServerSocket.Close;
  Close;
end;

procedure TChatForm.FormCreate(Sender: TObject);
begin

  FileListenItemClick(nil);
  if GetIPFromHost(Host, IP, Err) then
    Label1.Caption := Host + ' / ' + IP
  else
    MessageDlg(Err, mtError, [mbOk], 0);
end;

procedure TChatForm.ServerSocketError(Sender: TObject; Number:
Smallint;
  var Description: string; Scode: Integer; const Source,
HelpFile: string;
  HelpContext: Integer; var CancelDisplay: Wordbool);
begin
  ShowMessage(Description);
end;

procedure TChatForm.ClientSocketConnect(Sender: TObject;
  Socket: TCustomWinSocket);
begin
  Statusbar1.Panels[0].Text := 'Konek ke : ' +
Socket.RemoteHost;
end;


procedure TChatForm.ServerSocketClientRead(Sender: TObject;
  Socket: TCustomWinSocket);
var
  txt: string;
  val,head:integer;
  valf:single;
begin
  txt:=Socket.ReceiveText;
  val:=strtoint(txt);
  head:=val div 1000;
  val:=val mod 1000;
  valf:=val;
  if head>2 then valf:=valf-90;
  Memo2.Lines.Add(Socket.RemoteHost + ' :<<-    Motor =
'+inttostr(head)+',    Sudut='+floattostr(valf));
  if comport1.Connected then
  begin
    if head=1 then
    begin
      NilaiTimer:=$ffff - tabelmotor1[val+1];
      Comport1.WriteStr(#0);
    end
    else if head=2 then
```

```
    begin
      NilaiTimer:=$ffff - round(valf*8.6 + 1014);
      Comport1.WriteStr(#1);
    end
    else if head=3 then
    begin
      NilaiTimer:=$ffff - round(valf*8.8 + 1399);
      Comport1.WriteStr(#2);
    end
    else if head=4 then
    begin
      NilaiTimer:=$ffff - round(valf*-8.4 + 1410);
      Comport1.WriteStr(#3);
    end;

    TimerH:=NilaiTimer div 256;
    TimerL:=NilaiTimer mod 256;
    Comport1.Write(TimerH,1);
    Comport1.Write(TimerL,1);
  end;
  sleep(10);
end;

procedure TChatForm.ServerSocketAccept(Sender: TObject;
  Socket: TCustomWinSocket);
begin
  IsServer := True;
  Statusbar1.Panels[0].Text := 'Konek ke : ' +
Socket.RemoteAddress;
end;

procedure TChatForm.ServerSocketClientConnect(Sender: TObject;
  Socket: TCustomWinSocket);
begin
  Memo2.Lines.Clear;
end;

procedure TChatForm.ClientSocketDisconnect(Sender: TObject;
  Socket: TCustomWinSocket);
begin
  FileListenItemClick(nil);
end;

procedure TChatForm.ServerSocketClientDisconnect(Sender:
TObject;
  Socket: TCustomWinSocket);
begin
  Statusbar1.Panels[0].Text := 'Listening...';
end;

procedure TChatForm.Button4Click(Sender: TObject);
begin
  ServerSocket.Active := True;
  Statusbar1.Panels[0].Text := 'Listening...';
end;

procedure TChatForm.StatusBar1Resize(Sender: TObject);
begin
  StatusBar1.Anchors:=[akBottom];
end;
```

```
procedure TChatForm.AllKomputer1Click(Sender: TObject);
begin
  Form1.Show;
end;

procedure TChatForm.Button2Click(Sender: TObject);
begin
  if comport1.Connected then
  begin
    comport1.Connected:=false;
    button2.Caption:='Connect';
    cb1.Enabled:=true;
  end
  else
  begin
    comport1.Port:=cb1.Text;
    comport1.Connected:=true;
    button2.Caption:='Disconnect';
    cb1.Enabled:=false;
  end
end;


end.
```