SKRIPSI

PERANCANGAN MODUL DATA AKUISISI PENGATURAN KECEPATAN MOTOR DC DENGAN MASUKN DAC MENGGUNAKAN MATLAB

DISUSUN OLEH:

HASBULLAH

105 82 00 634 10

PROGRAM STUDI TEKNIK LISTRIK

JURUSAN TEKNIK ELEKTRO FAKULTAS TEKNIK

UNIVERSITAS MUHAMMADIYAH MAKASSAR

2014

Tugas Akhir

"PERANCANGAN MODUL DATA AKUISISI PENGATURAN KECEPATAN MOTOR DC DENGAN MASUKN DAC MENGGUNAKAN MATLAB"

SKRIPSI :

Diajukan untuk memenuhi persyaratan guna memeroleh gelar Sarjana Teknik

pada Jurusan Elektro Fakultas Teknik

Universitas Muhammadiyah Makassar

Disusun Oleh :

HASBULLAH
105 82 00634 10

RAHMAT
105 82 00486 10

JURUSAN TEKNIK ELEKTRO FAKULTAS TEKNIK

UNIVERSITAS MUHAMMADIYAH MAKASSAR

2014

UNIVERSITAS MUHAMMADIYAH MAKASSAR

FAKULTAS TEKNIK

Jl. Sultan Alauddin No. 259 Telp. (0411) 866 972 Fax (0411) 865 588 Makassar 90221

بسم الله الرحمن الرحيم

HALAMAN PENGESAHAN

Tugas Akhir ini diajukan untuk memenuhi syarat ujian guna memperoleh gelar Sarjana Teknik (ST) Program Studi Teknik Elektro Jurusan Teknik Elektro Fakultas Teknik Universitas Muhammadiyah Makassar.

Judul Skripsi : **Perancangan Modul Data Akuisisi Pengaturan Kecepatan Motor DC dengan Masukan DAC Menggunakan MATLAB**

Nama : Hasbullah Rahmat

Stambuk : 105 82 0634 10

105 82 0486 10

Makassar, 09 Desember 2014

Telah Diperiksa dan Disetujui
Oleh Dosen Pembimbing:

Pembimbing I	Pembimbing II
Dr. Ir. Indra Jaya Mansur, MT.	**Ir. Abd. Hapid, MT.**

Mengetahui,
Ketua Jurusan Elektro

Umar Katu, ST., MT.
NBM : 990 410

UNIVERSITAS MUHAMMADIYAH MAKASSAR

FAKULTAS TEKNIK

Jl. Sultan Alauddin No. 259 Telp. (0411) 866 972 Fax (0411) 865 588 Makassar 90221

بسم الله الرحمن الرحيم

PENGESAHAN

Skripsi atas nama Hasbullah dengan nomor induk Mahasiswa 105 82 0634 10 dan Rahmat dengan nomor induk Mahasiswa 105 82 0486 10, dinyatakan diterima dan disahkan oleh Panitia Ujian Tugas Akhir/Skripsi sesuai dengan Surat Keputusan Dekan Fakultas Teknik Universitas Muhammadiyah Makassar Nomor : 085/05/A.4-II/XII/36/2014, sebagai salah satu syarat guna memperoleh gelar Sarjana Teknik pada Program Studi Teknik Elektro Jurusan Teknik Elektro Fakultas Teknik Universitas Muhammadiyah Makassar pada hari Rabu Tanggal 03 Desember 2014

Makassar, 16 Shafar 1436 H
09 Desember 2014 M

Panitia Ujian :

1. Pengawas Umum
   a. Rektor Universitas Muhammadiyah Makassar
      Dr. H. Irwan Akib, M.Pd. : .....
   b. Dekan Fakultas Teknik Universitas Hasanuddin
      Dr. –Ing. Ir. Wahyu H. Piarah, MSME. : .....
2. Penguji
   a. Ketua : Andi Faharuddin, ST., MT. : .....
   b. Sekertaris : A. Abd. Halik Lateko, ST., MT. : .....
3. Anggota : 1. Dr. H. Zulfajri Basri Hasanuddin, M.Eng. : .....
   2. Dr. Ir. Zahir Zainuddin, M.Sc. : .....
   3. Suryani, ST., MT. : .....

Mengetahui :

Pembimbing I	Pembimbing II
Dr. Ir. Indra Jaya Mansur, MT.	**Ir. Abd. Hapid, MT.**

Ketua Program Studi
Teknik Elektro

Uthar Katu, ST., MT.
NBM : 990 410

KATA PENGANTAR

Assalamu'alaikum Wr. Wb.

Puji syukur kehadirat Allah S.W.T. yang telah memberikan segala kenikmatan dan anugrah kesehatan, sehingga penulis dapat menyelesaikan Tugas Akhir ini sesuai rencana. Tugas Akhir ini diajukan sebagai salah satu syarat untuk mencapai gelar sarjana S1 di Jurusan Teknik Listrik Fakultas Teknik Universitas Muhammadiyah Makassar.

Telah banyak tenaga dan pikiran yang penulis curahkan untuk menyelesaikan Tugas Akhir ini. Banyak kesulitan teknis maupun non teknis yang ditemui selama pengerjaan, tanpa bantuan dari orang-orang yang peduli maka penulis tidak akan mampu menyelesaikan Tugas Akhir ini. Pada kesempatan ini penulis ucapkan terimakasih setulus-tulusnya kepada Yth :

1. Bapak Umar Katu, ST.,MT selaku ketua jurusan Teknik Elektro Unismuh Makassar
2. Bapak Dr. Ir. Indra Jaya Mansur, MT selaku dosen pembimbing I atas bimbingannya kepada penulis.
3. Bapak Ir. Abd. Hafid, MT selaku dosen pembimbing II atas bimbingannya kepada penulis.
4. Seluruh rekan – rekan FakTek "Manuver 10" atas dukungan yang diberikan selama pengerjaan tugas akhir.

Penulis hanya dapat berdoa kepada Allah S.W.T., semoga semua kebaikan yang telah diberikan kepada penulis mendapatkan balasan yang setimpal dari-Nya. Dan dengan segala kerendahan hati, penulis mengharapkan semoga tugas akhir ini dapat bermanfaat. Terimakasih.

Wassalamu'alaikum Wr. Wb.

Makassar 11 November 2014

Penulis

ABSTRAK

Hasbullah, Rahmat: **Perancangan Modul Data Akuisisi Pengaturan Kecepatan Motor DC Dengan Masukan DAC Menggunakan Matlab.**

Penelitian ini bertujuan untuk : (1) Untuk Mengatur putaran motor DC dengan masukan DAC menggunakan Matlab, (2) Untuk menghasilkan sebuah sistem akuisisi data dengan perangkat keras yang portable. Alat yang dirancang berupa pengatur dan penggerak putaran motor DC dengan inputan DAC yang dapat di perlambat dan di percepat sesuai dengan nilai inputan pada program matlab dengan memanfaatkan PPI sebagai alat kontrol pengirim sinyal ke rangkaian DAC sebagai alat pengonversi sinyal digital ke sinyal analog. Perangkat lunak pendekode telah dibuat agar proses dekoding berjalan efisien ketika beberapa program aplikasi mengakses sistem secara bersamaan. Untuk memudahkan pengguna mengakses sistem, skrip kode antar muka program aplikasi telah dibuat dalam bahasa matlab. Hasil penelitian ini menunjukkan bahwa PPI dan DAC mampu mengakuisisi data mode inputan minimum 60 bit (00111100 dan 1,5625 volt) dengan kecepatan 1066,41 rpm dan inputan maksimum 255 bit (11111111 dan 4,9 Volt) dengan kecepatan 922,852 rpm

Kata kunci: Akuisisi Data, Motor DC, Matlab, PPI, DAC

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I

PENDAHULUAN

1.1 Latar Belakang

Kecepatan putaran merupakan salah satu parameter penting yang harus dikendalikan dari motor DC. Pengaturan kecepatan putaran motor DC sering dilakukan dengan menggunakan rangkaian-rangkaian analog. Salah satu kelemahan pengaturan kecepatan putaran motor DC menggunakan rangkaian analog adalah sulitnya mengetahui berapa kecepatan putaran motor DC yang sebenarnya. Olehnya itu, bisa digunakan rangkaian kombinasi yaitu rangkaian analog sebagai pengatur kecepatan putaran motor DC dan rangkaian digital untuk menampilkan nilai kecepatan putaran motor DC.

Perkembangan teknologi saat ini mulai bergeser menuju proses otomatisasi dengan menggunakan komputer sebagai pusat pengontrolan peralatan elektronika. Perpaduan rangkaian analog, rangkaian digital, rangkaian terintegrasi, komputer hardware dan software menjadikan pengendalian sistem semakin mudah, akurat, fleksibel, dan lebih cerdas. Kelebihan pengaturan kecepatan putaran motor DC menggunakan komputer adalah tingkat akurasi yang tinggi, mudah dalam memonitoring kecepatan putaran serta pengendalian yang mudah karena pengaturannya menggunakan program (software) komputer.

Dalam aplikasi sistem pengendalian di industri yang menggunakan motor DC, kebutuhan industri untuk pengambilan dan pengolahan data menjadi semakin

komplek, semakin variatif dan semakin banyak. Oleh karena itu, dibutuhkan suatu perangkat yang dapat menangani kebutuhan tersebut, salah satunya adalah dengan system akuisisi data. Tugas utama dari sistem akuisisi data adalah mengakuisisi sinyal sensor yang biasanya berupa sinyal analog, mengubahnya menjadi sinyal digital dan memberikannya kepada sistem monitoring ataupun system pengendalian.

Beberapa fungsionalitas yang bersifat opsional kadang juga tersedia dalam modul sistem akuisisi data, seperti *filter*, Modulator dan sebagainya. Sistem akuisisi data tersebut biasanya dikendalikan oleh program, baik yang berjalan sebagai *embedded system* maupun program aplikasi dalam sebuah *personal computer* (PC).

Sebuah sistem akusisi data atau biasa dikenal *Data-Acquisition Sistem* (DAS) merupakan sistem instrumentasi elektronik terdiri dari sejumlah elemen yang secara bersama-sama bertujuan melakukan pengaturan, dan mengolah hasil pengukuran. Secara aktual DAS berupa *interface* antara lingkungan analog dengan lingkungan digital. Lingkungan analog meliputi transduser dan pengondisian sinyal dengan segala kelengkapannya, sedangkan lingkungan digital meliputi *digital alalog to digital converter* (DAC) dan selanjutnya *pemrosesan* yang dilakukan oleh Matlab atau sistem berbasis MATLAB 7.10.0.

1.2 Rumusan Masalah

A. Bagimana merancang modul data akuisisi pengaturan kecepatan motor DC dengan menggunakan matlab?

B. Bagaimana penggunaan DAC sebagai input untuk penggerak motor DC?

1.3 Tujuan Penelitian

Tujuan penelitian ini adalah:

1. Untuk Mengatur putaran motor DC dengan masukan DAC menggunakan Matlab
2. Untuk menghasilkan sebuah sistem akuisisi data dengan perangkat keras yang portable.

1.4 Manfaat Penelitian

Jika tujuan penelitian ini tercapai, maka hasil dari penelitian ini akan membawa beberapa manfaat:

1. Dapat mengatur putaran motor DC dengan masukan DAC menggunakan Matlab
2. Dapat menghasilkan sebuah sistem akuisisi data dengan perangkat keras yang portable.

1.5 Batasan Masalah

Dalam penelitian ini, masalah yang akan diteliti mempunyai batasan sesuai judul yang diajukan. Judul yang diajukan adalah "Perancangan Modul Data Akuisisi Pengaturan Kecepatan Motor DC Dengan Masukan DAC Menggunakan Matlab".

Yang dimaksud dengan perancangan disini adalah proses mulai dari merancang di atas kertas sampai membuat prototipe yang teruji. Sistem akuisisi data yang dirancang terdiri dari perangkat keras dan perangkat lunak. Perancangan sampai pengujian prototipe perangkat keras dibatasi sampai pada fungsi kerjanya sebagai perangkat akuisisi data.

Yang dimaksud dengan "Menggunakan Masukan Matlab " pada judul yang diajukan adalah bahwa perangkat keras yang dirancang nantinya akan dihubungkan ke komputer melalui masukan (line-in), yang sudah tersedia pada hampir semua komputer yang ada saat ini. Komputer yang dipakai dibatasi pada komputer pribadi (PC, Personal Computer).

1.6 Sistematika Penelitian

BAB I **PENDAHULUAN**

Berisikan tentang latar belakang penelitian, batasan masalah, tujuan penelitian, manfaat penelitian dan sistematika penelitian

BAB II TINJAUAN PUSTAKA

Berisikan tentang pokok pembahasan teori atau materi yang mendasari dalam pelaksanaan penelitian ini.

BAB III METODOLOGI

Berisikan tentang tempat pelaksanaan penelitian serta metode yang diterapkan dalam tugas akhir ini.

BAB IV HASIL DAN PEMBAHASAN

Berisikan tentang data pencapaian yang dilakukan dalam penelitian ini serta data pengujian penelitian.

BAB V PENUTUP

Kesimpulan dari hasil yang telah dicapai serta saran kepada pembaca untuk memanfaatkan dan mengembangkan penelitian ini.

BAB II

TINJAUAN PUSTAKA

2.1. Motor DC

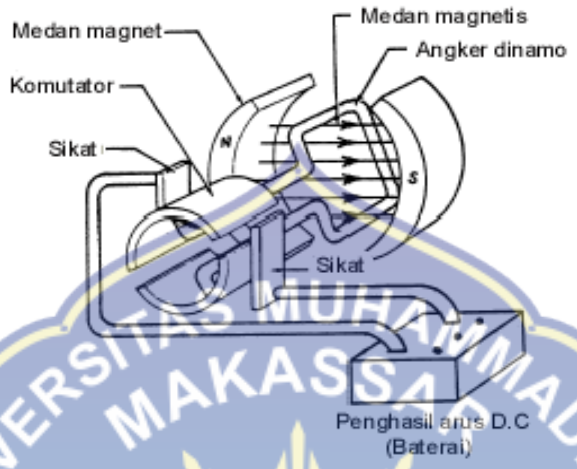
Motor listrik merupakan perangkat elektromagnetis yang mengubah energi listrik menjadi energi mekanik. Energi mekanik ini digunakan untuk, misalnya memutar *impeller* pompa, *fan* atau *blower*, menggerakan kompresor, mengangkat bahan,dll. Motor listrik digunakan juga di rumah (*mixer*, bor listrik, *fan* angin) dan di industri. Motor listrik kadang kala disebut "kuda kerja" nya industri sebab diperkirakan bahwa motor-motor menggunakan sekitar 70% beban listrik total di industri.

Motor DC memerlukan suplai tegangan yang searah pada kumparan medan untuk diubah menjadi energi mekanik, bagiaan utama Motor DC :

1. Kumparan medan pada motor dc disebut stator (bagian yang tidak berputar).
2. Kumparan jangkar disebut rotor (bagian yang berputar).

Jika terjadi putaran pada kumparan jangkar dalam pada medan magnet, maka akan timbul tegangan (GGL) yang berubah-ubah arah pada setiap setengah putaran, sehingga merupakan tegangan bolak-balik. Prinsip kerja dari arus searah adalah membalik phasa tegangan dari gelombang yang mempunyai nilai positif dengan menggunakan komutator, dengan demikian arus yang berbalik arah dengan kumparan jangkar yang berputar dalam

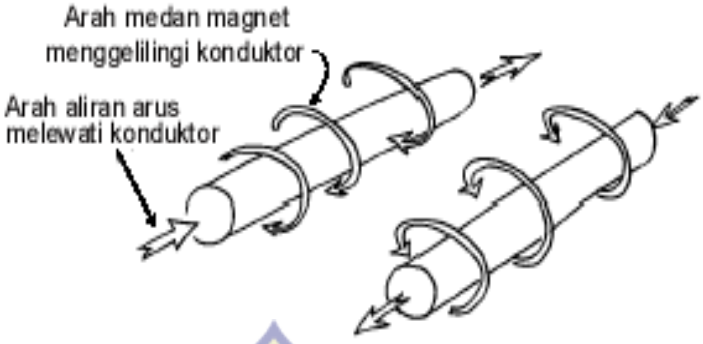
medan magnet. Bentuk motor paling sederhana memiliki kumparan satu lilitan yang bisa berputar bebas di antara kutub-kutub magnet permanen.



Gambar 2.1 Motor DC Sederhana

Catu tegangan dc dari baterai menuju ke lilitan melalui sikat yang menyentuh komutator, dua segmen yang terhubung dengan dua ujung lilitan. Kumparan satu lilitan pada gambar di atas disebut angker dinamo. Angker dinamo adalah sebutan untuk komponen yang berputar di antara medan magnet.

Jika arus lewat pada suatu konduktor, timbul medan magnet di sekitar konduktor. Arah medan magnet ditentukan oleh arah aliran arus pada konduktor.



Gambar 2.2 medan magnet yang membawa arus mengelilingi konduktor

Aturan Genggaman Tangan Kanan bisa dipakai untuk menentukan arah garis fluks di sekitar konduktor. Genggam konduktor dengan tangan kanan dengan jempol mengarah pada arah aliran arus, maka jari-jari anda akan menunjukkan arah garis fluks. Gambar 3 menunjukkan medan magnet yang terbentuk di sekitar konduktor berubah arah karena bentuk U.

Gambar 2.3 Medan magnet yang membawa arus mengelilingi konduktor

Gambar 2.4 Reaksi garis fluks.

Lingkaran bertanda A dan B merupakan ujung konduktor yang dilengkungkan (looped conductor). Arus mengalir masuk melalui ujung A dan keluar melalui ujung B.

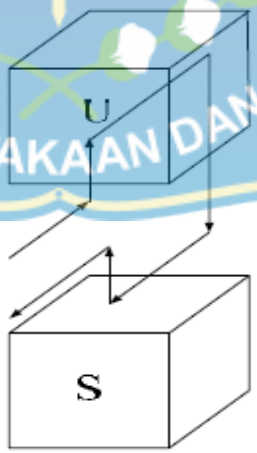
Medan konduktor A yang searah jarum jam akan menambah medan pada kutub dan menimbulkan medan yang kuat di bawah konduktor. Konduktor akan berusaha bergerak ke atas untuk keluar dari medan kuat ini. Medan konduktor B yang berlawanan arah jarum jam akan menambah medan pada kutub dan menimbulkan medan yang kuat di atas konduktor. Konduktor akan berusaha untuk bergerak turun agar keluar dari medan yang kuat tersebut. Gaya-gaya tersebut akan membuat angker dinamo berputar searah jarum jam.

Mekanisme kerja untuk seluruh jenis motor secara umum :

1. Arus listrik dalam medan magnet akan memberikan gaya.
2. Jika kawat yang membawa arus dibengkokkan menjadi sebah lingkaran / loop, maka kedua sisi loop, yaitu pada sudut kanan medan magnet, akan mendapatkan gaya pada arah yang berlawanan.

3. Pasangan gaya menghasilkan tenaga putar / torque untuk memutar kumparan.
4. Motor-motor memiliki beberapa loop pada dinamonya untuk memberikan tenaga putaran yang lebih seragam dan medan magnetnya dihasilkan oleh susunan elektromagnetik yang disebut kumparan medan.

Pada motor dc, daerah kumparan medan yang dialiri arus listrik akan menghasilkan medan magnet yang melingkupi kumparan jangkar dengan arah tertentu. Konversi dari energi listrik menjadi energi mekanik (motor) maupun sebaliknya berlangsung melalui medan magnet, dengan demikian medan magnet disini selain berfungsi sebagai tempat untuk menyimpan energi, sekaligus sebagai tempat berlangsungnya proses perubahan energi, daerah tersebut dapat dilihat pada gambar di bawah ini :



Gambar 2.5 Prinsip kerja motor dc

Agar proses perubahan energi mekanik dapat berlangsung secara sempurna, maka tegangan sumber harus lebih besar daripada tegangan gerak yang disebabkan reaksi lawan. Dengan memberi arus pada kumparan jangkar yang dilindungi oleh medan maka menimbulkan perputaran pada motor.

Dalam memahami sebuah motor, penting untuk mengerti apa yang dimaksud dengan beban motor. Beban dalam hal ini mengacu kepada keluaran tenaga putar / torque sesuai dengan kecepatan yang diperlukan. Beban umumnya dapat dikategorikan ke dalam tiga kelompok :

1. Beban torque konstan

   Adalah beban dimana permintaan keluaran energinya bervariasi dengan kecepatan operasinya namun torquenya tidak bervariasi. Contoh beban dengan torque konstan adalah corveyors, rotary kilns, dan pompa displacement konstan.

2. Beban dengan variabel torque

   Adalah beban dengan torque yang bervariasi dengan kecepatn operasi. Contoh beban dengan variabel torque adalah pompa sentrifugal dan fan (torque bervariasi sebagai kuadrat kecepatan). Peralatan Energi Listrik : Motor Listrik.

3. Beban dengan energi konstan

   Adalah beban dengan permintaan torque yang berubah dan berbanding terbalik dengan kecepatan. Contoh untuk beban dengan daya konstan adalah peralatan-peralatan mesin.

Motor DC adalah peralatan elektromekanis yang mengubah daya listrik menjadi daya mekanis dengan sumber arus searah kecepatan putaran merupakan salah satu parameter penting yang harus dikendalikan dari motor DC. Pengaturan kecepatan putaran motor DC sering dilakukan dengan menggunakan rangkaian-rangkaian analog. Salah satu kelemahan pengaturan kecepatan putaran motor DC menggunakan rangkaian analog adalah sulitnya mengetahui berapa kecepatan putaran motor DC yang sebenarnya. Olehnya itu, bisa digunakan rangkaian kombinasi yaitu rangkaian analog sebagai pengatur kecepatan putaran motor DC dan rangkaian digital untuk menampilkan nilai kecepatan putaran motor DC.

Keuntungan utama motor DC adalah kecepatannya mudah dikendalikan dan tidak mempengaruhi kualitas pasokan daya. Motor DC ini dapat dikendalikan dengan mengatur:

1. Tegangan dinamo, meningkatkan tegangan dinamo akan meningkatkan keepatan.
2. Arus medan, menurunkan arus medan akan meningkatkan kecepatan.
3. Torka dan kecepatannya mudah dikendalikan.
4. Torka awalnya besar.
5. Performansinya mendekati linier.
6. Sistem kontrolnya relatif lebih murah dan sederhana.
7. Cocok untuk aplikasi motor servo karena respon dinamiknya yang baik.
8. Untuk aplikasi berdaya rendah, motor DC lebih murah dari motor AC

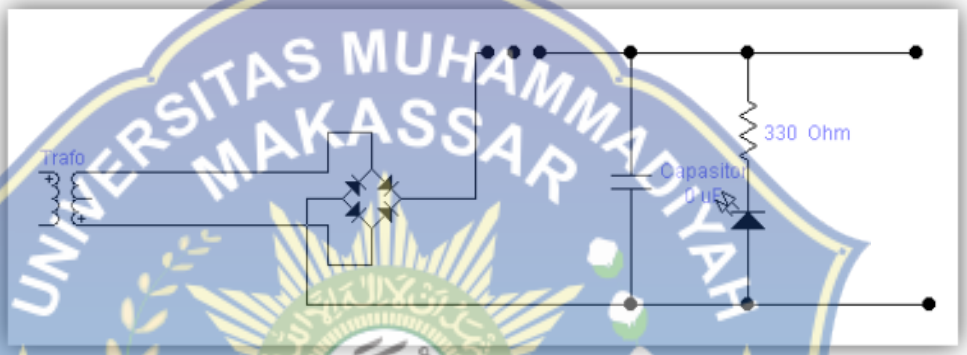
Perkembangan teknologi saat ini mulai bergeser menuju proses otomatisasi dengan menggunakan komputer sebagai pusat pengontrolan peralatan elektronika. Perpaduan rangkaian analog, rangkaian digital, rangkaian terintegrasi, komputer hardware dan software menjadikan pengendalian sistem semakin mudah, akurat, fleksibel, dan lebih cerdas. Kelebihan pengaturan kecepatan putaran motor DC menggunakan komputer adalah tingkat akurasi yang tinggi, mudah dalam memonitoring kecepatan putaran serta pengendalian yang mudah karena pengaturannya menggunakan program (software) komputer.

2.2 Sistem akuisisi Data

Sebuah sistem akusisi data atau biasa dikenal *Data-Acquisition Sistem* (DAS) merupakan sistem instrumentasi elektronik terdiri dari sejumlah elemen yang secara bersamasama bertujuan melakukan pengukuran, menyimpan, dan mengolah hasil pengukuran. Secara aktual DAS berupa *interface* antara lingkungan analog dengan lingkungan digital. Lingkungan analog meliputi transduser dan pengondisian sinyal dengan segala kelengkapannya, sedangkan lingkungan digital meliputi *analog to digital converter* (ADC), *digital to analog comverter* (DAC), dan selanjutnya *pemrosesan digital* yang dilakukan oleh sistem berbasis Matlab, untuk menghasilkan perangkat DAS yang memiliki program Matlab.

2.3 Power Supply

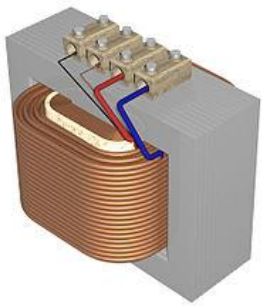
Power supply adalah alat atau sistem yang berfungsi untuk menyalurkan energi listrik atau bentuk energi jenis apapun yang sering digunakan untuk menyalurkan energi listrik. Secara prinsip rangkaian power supply adalah menurunkan tegangan AC , menyearah kan tegangan AC sehingga menjadi DC ,menstabilkan tegangan DC, yang terdiri atas :



Gambar 2.6 rangkaian power supply sederhana

2.3.1 Transformator

Tranformator biasanya berbentuk kotak dan terdapat lilitan - lilitan kawat email didalamnya. Tugas dari komponen ini adalah untuk menaikkan atau menurunkan tegangan AC sesuai kebutuhan.



Gambar 2.7 transformator

2.3.2 Dioda

komponen aktif yang memiliki dua kutub dan bersifat semikonduktor. Dioda juga bisa dialiri arus listrik ke satu arah dan menghambat arus dari arah sebaliknya. Dioda sebenarnya tidak memiliki karakter yang sempurna, melainkan memiliki karakter yang berhubungan dengan arus dan tegangan komplek yang tidak linier dan seringkali tergantung pada teknologi yang digunakan serta parameter penggunaannya.

gambar 2.8 simbol dioda

2.3.4 Kapasitor

Perangkat komponen elektronika yang berfungsi untuk menyimpan muatan listrik dan terdiri dari dua konduktor yang dipisahkan oleh bahan penyekat (dielektrik) pada tiap konduktor atau yang disebut keping. Kapasitor biasanya disebut dengan sebutan kondensator yang merupakan komponen listrik dibuat sedemikian rupa sehingga mampu menyimpan muatan listrik.

Prinsip kerja kapasitor pada umunya hampir sama dengan resistor yang juga termasuk ke dalam komponen pasif. Komponen

pasif adalah jenis komponen yang bekerja tanpa memerlukan arus panjar. Kapasitor sendiri terdiri dari dua lempeng logam (konduktor) yang dipisahkan oleh bahan penyekat (isolator). Penyekat atau isolator banyak disebut sebagai bahan zat dielektrik.

Gambar 2.9 gambar kapasitor

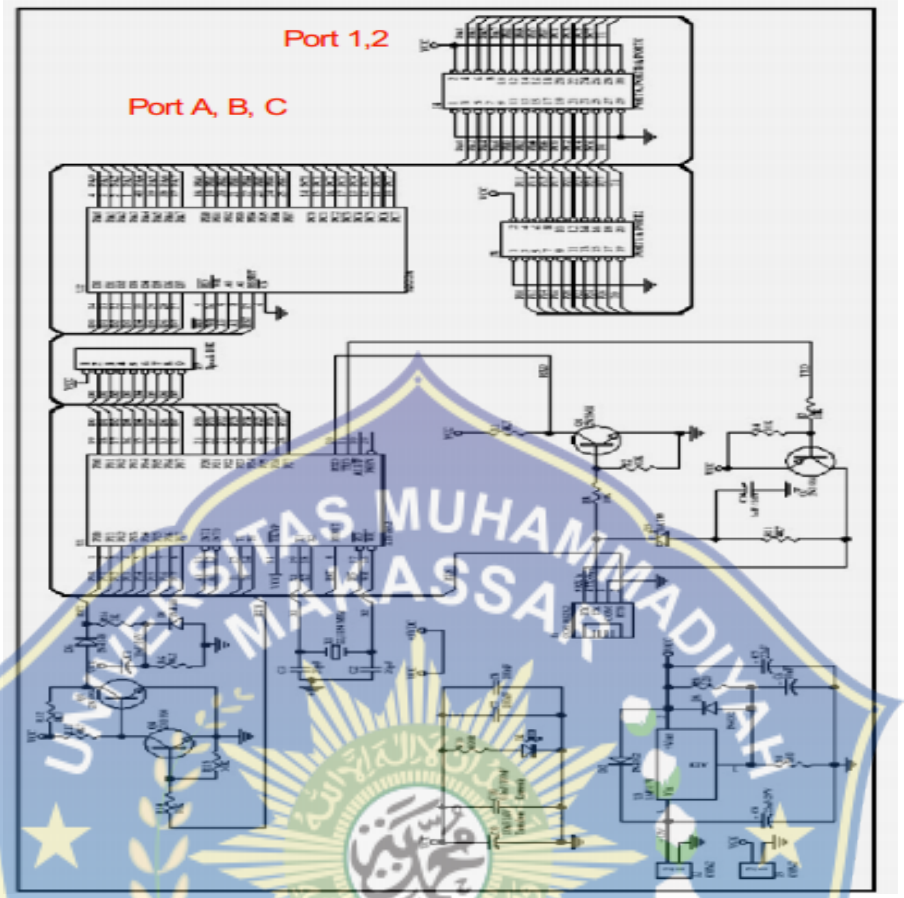
2.4 PC Link Serial PPI

PC-Link SERIAL PPI merupakan pengendali 40 bit jalur input/output melalui antarmuka UART RS-232 (jalur serial port) yang dapat dihubungkan ke komputer secara langsung atau melalui USB ke serial Adapter. Peralatan ini dibangun dari IC 82C55A, dimana IC tersebut merupakan general purpose programmable I/O device dengan tiga port yaitu port A, B, dan C. PC-Link SERIAL PPI juga menggunakan IC mikrokontrolel 89S51 dari Atmel dengan dua port, yaitu port 1 dan port 2. Keseluruhan port pada kit ini ada 5 buah, dan data I/O yang dapat diprogram pada masing port adalah data 8 bit.

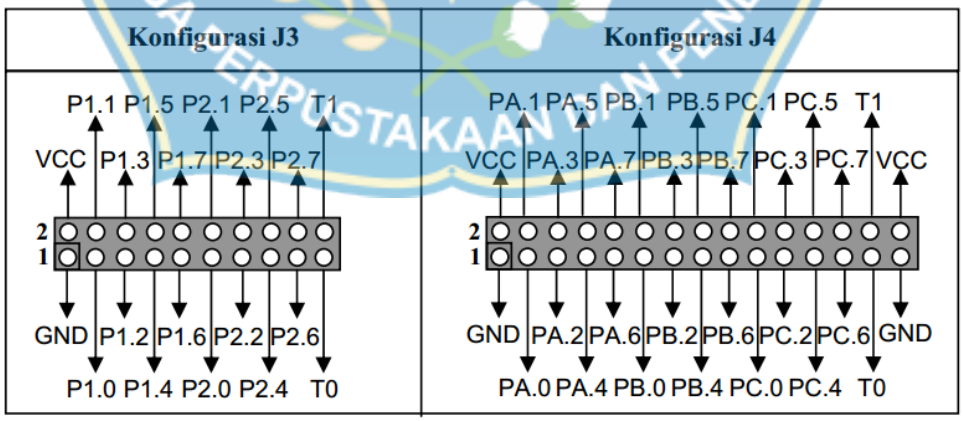
Gambar 2.10 PC-Link Serial PPI

Spesifikasi eksternal PPI adalah sebagai berikut:

1. Menggunakan antar muka UART RS-232
2. 4 pilihan baud rate
3. 16 bit jalur input/output (port 1 dan port 2) dengan level CMOS
4. 24 bit jalur programable peripheral interface 82C55 (Port A, Port B dan Port C) dengan level CMOS
5. 2 counter 16 bit (counter 0 dan counter 1) dengan level CMOS
6. Sumber tegangan input 12 VDC
7. Tersedia Voltage Regulator dengan tegangan output 5 VDC



Gambar 2.11 Diagram skematik PC-Link Serial PPI



Gambar 2.12 alokasi dan spesifikasi port

Tabel 2.1 Spesifikasi Port 1, 2, Counter 0 dan Counter 1 adalah sebagai berikut :

Simbol	Parameter	Nilai	satuan
I_{OL}	Arus saat output berlogika ‘0’	1,6	mA
I_{OH}	Arus saat output berlogika ‘1’	-10	µA

I_{OL} maksimum per pin adalah 10 mA

I_{OL} maksimum per port adalah 15 mA

I_{OL} maksimum untuk semua port adalah 71 mA

Tabel 2.2 Spesifikasi untuk Port A, B dan C adalah sebagai berikut :

Simbol	Parameter	Nilai	satuan
I_{OL}	Arus saat output berlogika ‘0’	2,5	mA
I_{OH}	Arus saat output berlogika ‘1’	-100	µA

2.5 DT-I/O DAC-08 (Digital To Analog Converter)

DAC-08 adalah Digital to Analog Converter berbasis AD7302 berukuran kecil yang membutuhkan catu daya tunggal. Aplikasinya antara lain untuk instruman bertenaga batterai serta sebagai sumber tegangan dan arus yang dapat diprogram.

Gambar 2.13 DT-I/O DAC-08 (Digital To Analog Converter)

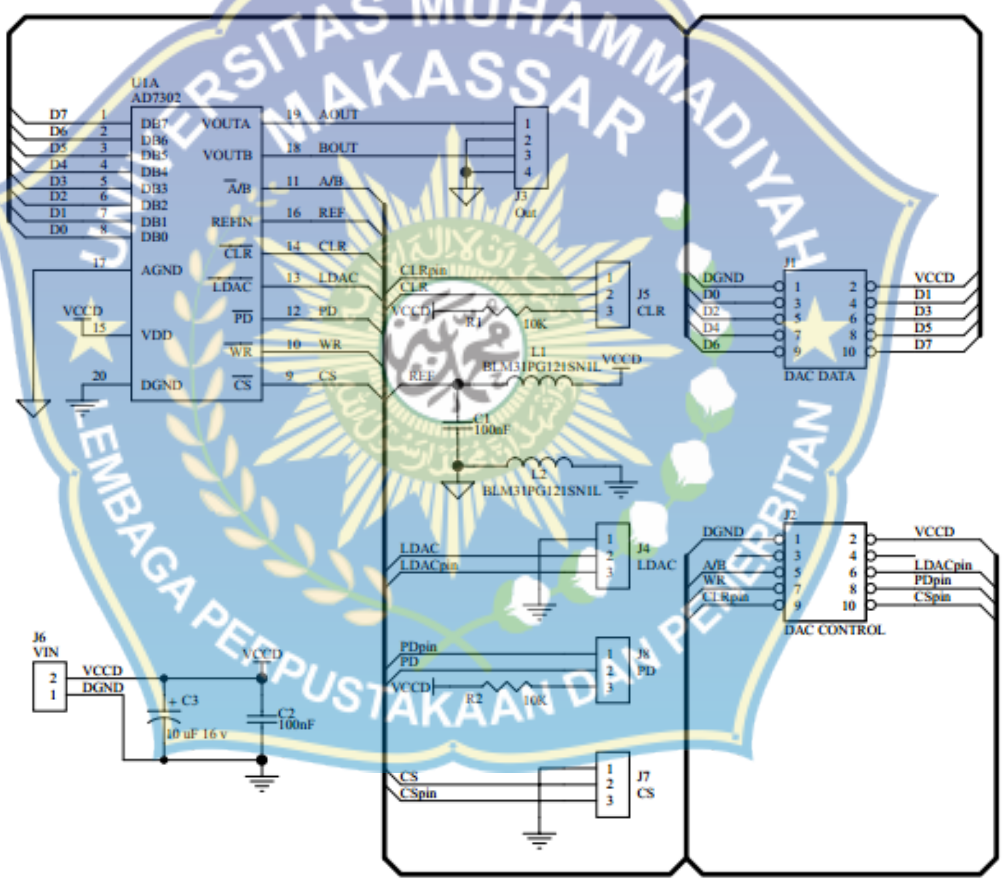
2.5.1 Fitur dan spesifikasi teknis:

1. Memiliki dua DAC 8 bit.
2. Tegangan kerja (VDD) dari + 2,7 VDC hingga + 5,5 VDC.
3. Tegangan referensi (Vref) = VDD/2
4. Fungsi power – Down.
5. Beroperasi secara Rail – To – Rail dengan setting time sekitar 1,2 µs.
6. Memiliki 2 mode :

a. Automatic update (untuk memperbaharui/update masing – masing outpu DAC secara real time)

b. Simultaneos update (untuk memperbaharui/update kedua output DAC secara bersamaan).

7. Selisih hasil pengukuran dan perhitungan maksimum 3 LSB (sekitar 59mV dengan menggunakan VDD = +5 VDCP).
8. Antar muka paralel kecepatan tinggi.
9. Dapat dihubungkan melalui pin I/O ataupun intel sistem bus.
10. Dilengkapi rutin – rutin siap pakai untuk DT-51 low cost series dan DT-51 minimum sistem ver. 3,0



Gambar 2.14 rangkaian DT-I/O DAC-08 (Digital To Analog Converter)

2.5.2 Cara menghitung Output DAC

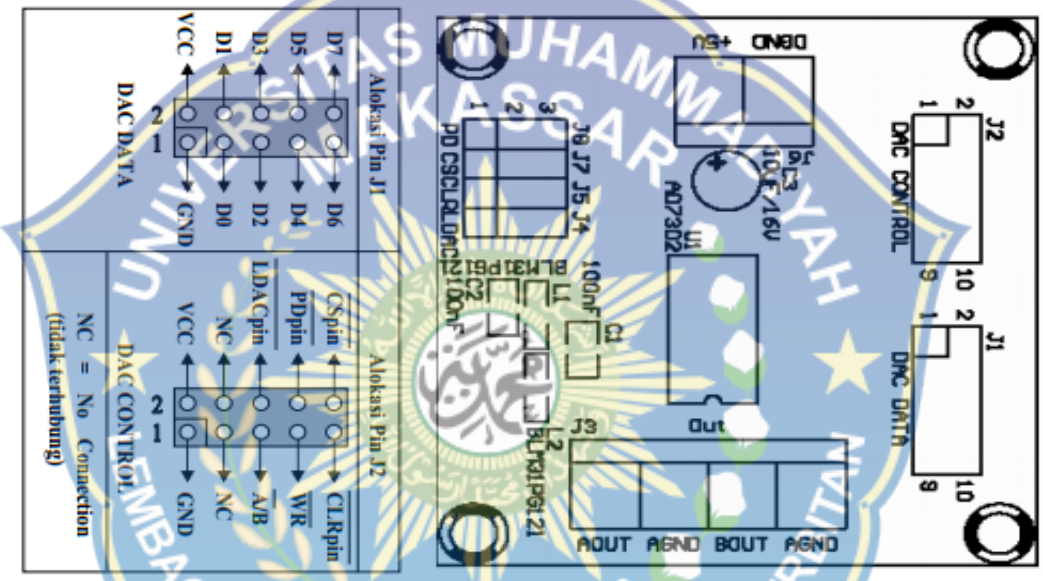
Vout = 2 x N x Vref/256

Dimana :

Vout : Tegangan Output

N : Nilai register output DAC

Vref : Tegangan referensi (=VDD/2)



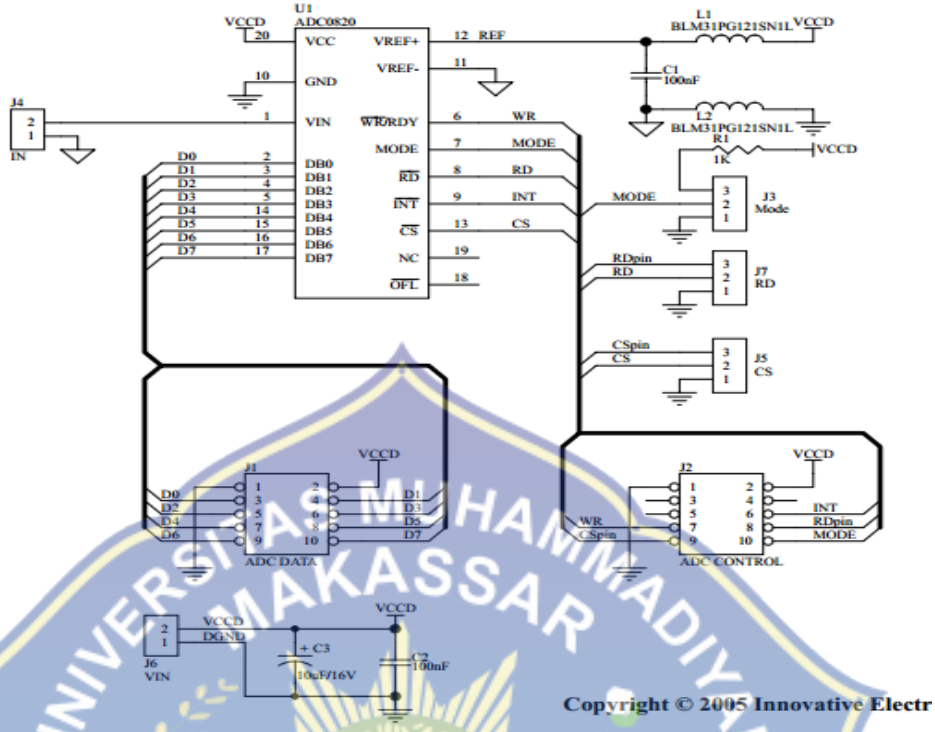
Gambar 2.15 tata letak DT-I/O DAC-08 (Digital To Analog Converter)

2.6 DT-I/O ADC-08 (Analog To Digital Converter)

ADC-08 adalah Analog to Digital Converter berbasis ADC0820 yang membutuhkan catu daya +5 VDC. Aplikasinya antara lain untuk pendeteksi tegangan dan mengubah data sensor analog menjadi digital.

2.6.1 Fitur & Spesifikasi Teknis

1. Resolusi ADC 8-bit.
2. Tegangan kerja (VCC) = Tegangan referensi (Vref) = +5 VDC.
3. Fungsi track-and-hold yang terintegrasi.
4. Tanpa clock eksternal.
5. Memiliki tiga operasi:
   a. RD (Read) Mode
   b. WR-RD (Write-Read) Mode
   c. WR-RD Stand Alone Operation
6. Waktu Konversi 2,5 µs pada Read Mode dan 1,5 µs pada Write-Read Mode dan WR-RD Stand Alone Operation.
7. Range input 0 VDC hingga +5 V (dengan VCC = +5 VDC).
8. Selisih hasil pengukuran dan penghitungan maksimum 1 LSB (sekitar 20 mV dengan menggunakan VCC = +5 VDC).
9. Tidak membutuhkan pengaturan zero atau full-scale adjust.
10. Antarmuka paralel dengan level tegangan CMOS atau TTL.
11. Dapat dihubungkan melalui pin I/O ataupun Intel System Bus (System Bus hanya mendukung WR-RD Mode).
12. Dilengkapi rutin-rutin siap pakai dalam bahasa Assembly untuk DT-51™ Low Cost Series dan DT-51™ Minimum System ver 3.0



Gambar 2.16 tata letak DT-I/O ADC-08 (Analog To Digital Converter)

2.6.2 Cara menghitung output ADC

Rumus output ADC

$$N = Vin/Vref \times 256$$

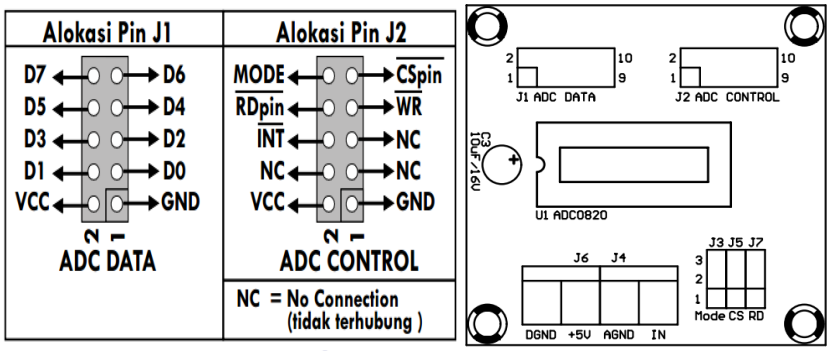
Dimana

N : nilai register output ADC

Vin : tegangan input

Vref : tegangan referensi (= VCC)

Selisih hasil penghitungan dan pengukuran dapat mencapai 1 LSB (sekitar 20 mV dengan menggunakan VCC = +5 VDC)



Gambar 2.17 tata letak DT-I/O ADC-08 (Analog To Digital Converter)

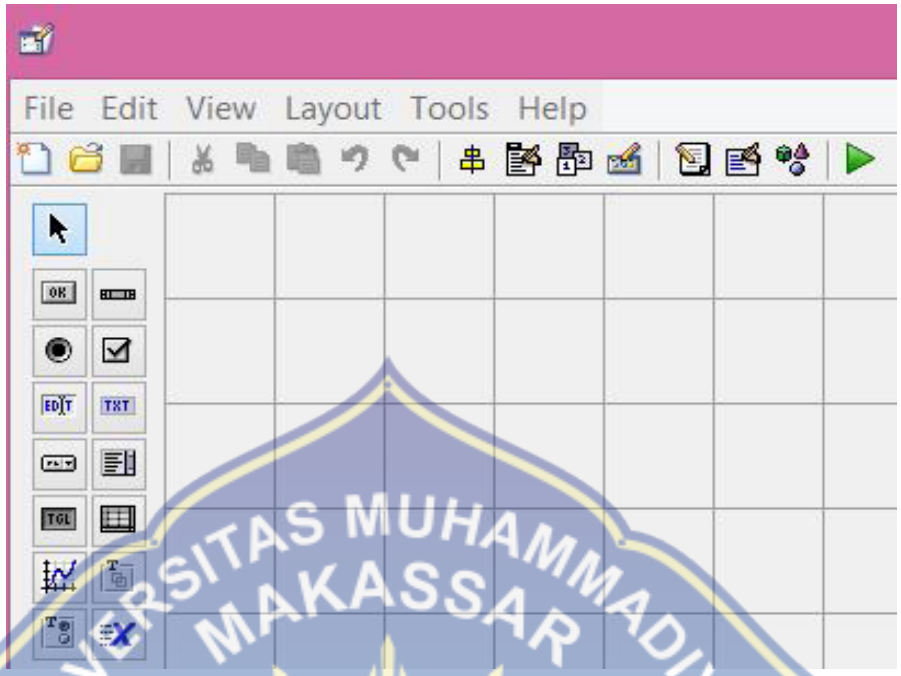
2.7 Matlab

Matlab merupakan bahasa tingkat tinggi dan lingkungan interaktif yang memungkinkan untuk melakukan tugas-tugas komputasi secara intensif lebih cepat dibandingkan dengan bahasa pemrograman tradisional seperti C, C++, dan Fortran.

Matlab adalah sebuah lingkungan komputasi numerikal dan bahasa pemrograman komputer generasi keempat.

GUI MATLAB

graphical user interface (GUI) adalah antar muka berupa gambar (pictorialinterface) untuk sebuah program. Tampilan jendela program GUI MATLAB diperlihatkan pada gambar 2.5.



Gambar 2.18 Jendela untuk program GUI MATLAB

Komponen yang terletak pada sebelah kiri jendela gambar 2.4 masing-masing disebut sebagaiPush Button, Edit Text, Static Text, dan sebagainya. Komponen-komponen tersebut dipindahkan ke sebelah kanan jendela jika dibutuhkan dan merupakan bagian dari program. Bagi mereka yangterbiasa bekerja dengan Borland Delphi atau Visual Basic, maka GUI MATLAB dengan kedua software tersebut cara kerjanya hampir sama.

Dewasa ini telah lazim digunakan program–program komputer sebagai alat bantu dalam menyelesaikan masalah engineering. Program-program komputer semacam MATLAB yang dikembangkan oleh MathWork banyak dipakai untuk keperluan perhitungan teknik, komputasi dan visualisasi dalam lingkungan yang terintegrasi, misalnya :

a. Komputasi dan matematis.

b. Pengembangan algoritma.

c. Akuisisi data (DAS).

d. Pemodelan, simulasi dan membuat prototype.

e. Analisis data, eksplorasi dan visualisasi.

f. Grafik pada bidang sains dan rekayasa.

g. Pengembangan aplikasi, termasuk membangun GUI (graphical user interface).

Perangkat akuisisi data bekerja sebagai antarmuka antara komputer dengan dunia luar. Fungsi utamanya adalah sebagai peranti yang mengubah data analog ke digital dan sebaliknya agar dapat ditafsirkan oleh komputer. Sebuah peranti DAS (perangkat keras DAS) umumnya mempunyai fungsi berikut:

h. Analog input.

i. Analog output.

j. Digital I/O.

k. Counter.

BAB III

METODE PENELITIAN

3.1 Waktu dan Tempat penelitian

a. Waktu Penelitian

Sasaran yang ingin dicapai pada usulan penelitian 1 bulan untuk merancang perangkat akuisisi data (DAS Adaptor) yang dapat dioperasikan pada software MATLAB. Perangkat DAS.

b. Tempat penelitian

penelitian dilaksanakan di Jalan Balang Caddi No 26 Makassar

3.2 Tahapan Penelitian

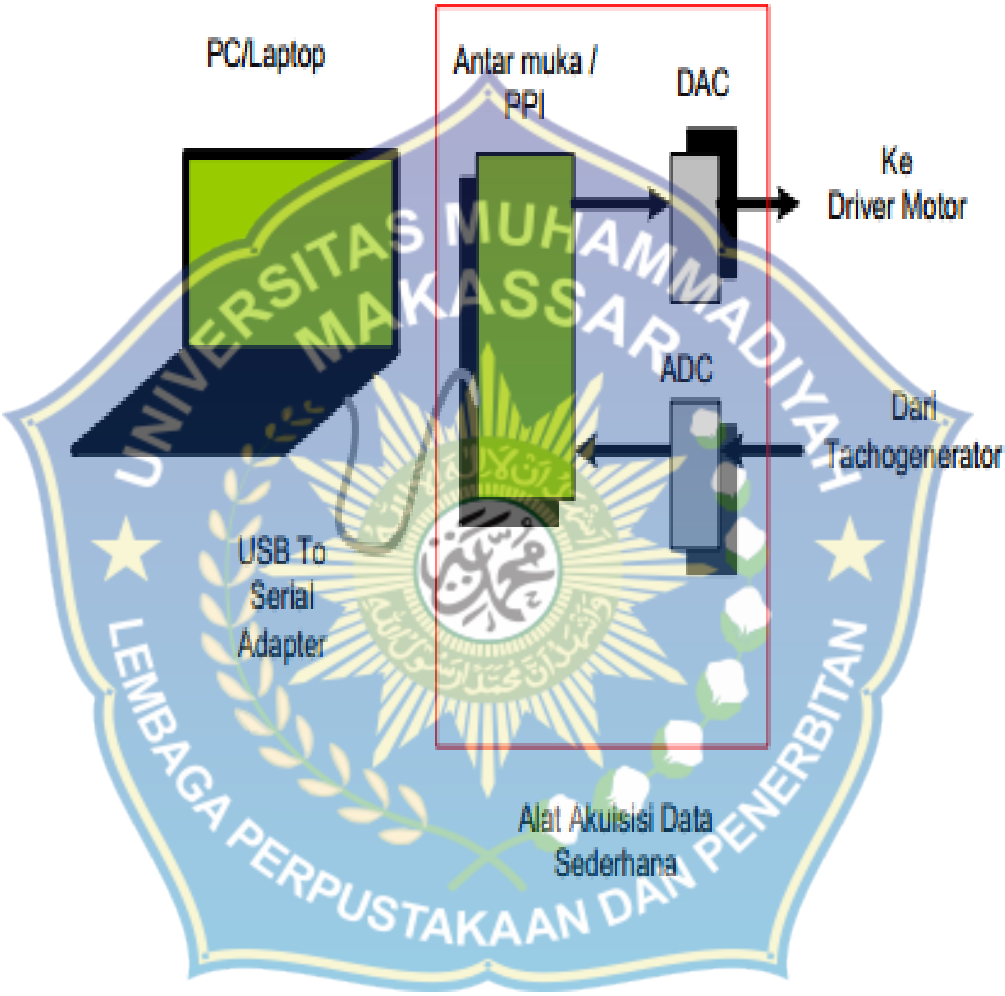
Tahapan yang dilakukan dalam penelitian Rancang Bangun Perangkat Akuisisi data Berbasis IC 82C55 Pada MATLAB, yaitu:

a. Studi literatur, menelusuri semua informasi yang berhubungan dengan desain dan produk DAS adaptor yang sudah terpakai pada MATLAB yang dapat menggerakan Motor DC. Misalnya perangkat DAS buatan National Instrumen mempunyai empat tipe, yaitu:

1. Dekstop DAS device, perangkat ini dihubungkan melalui slot PCI komputer. Software berjalan pada komputer.
2. Portable DAS devices, perangkat ini dihubungkan melalui port USB, sSoftware berjalan pada komputer.
3. Distributed DAS devices

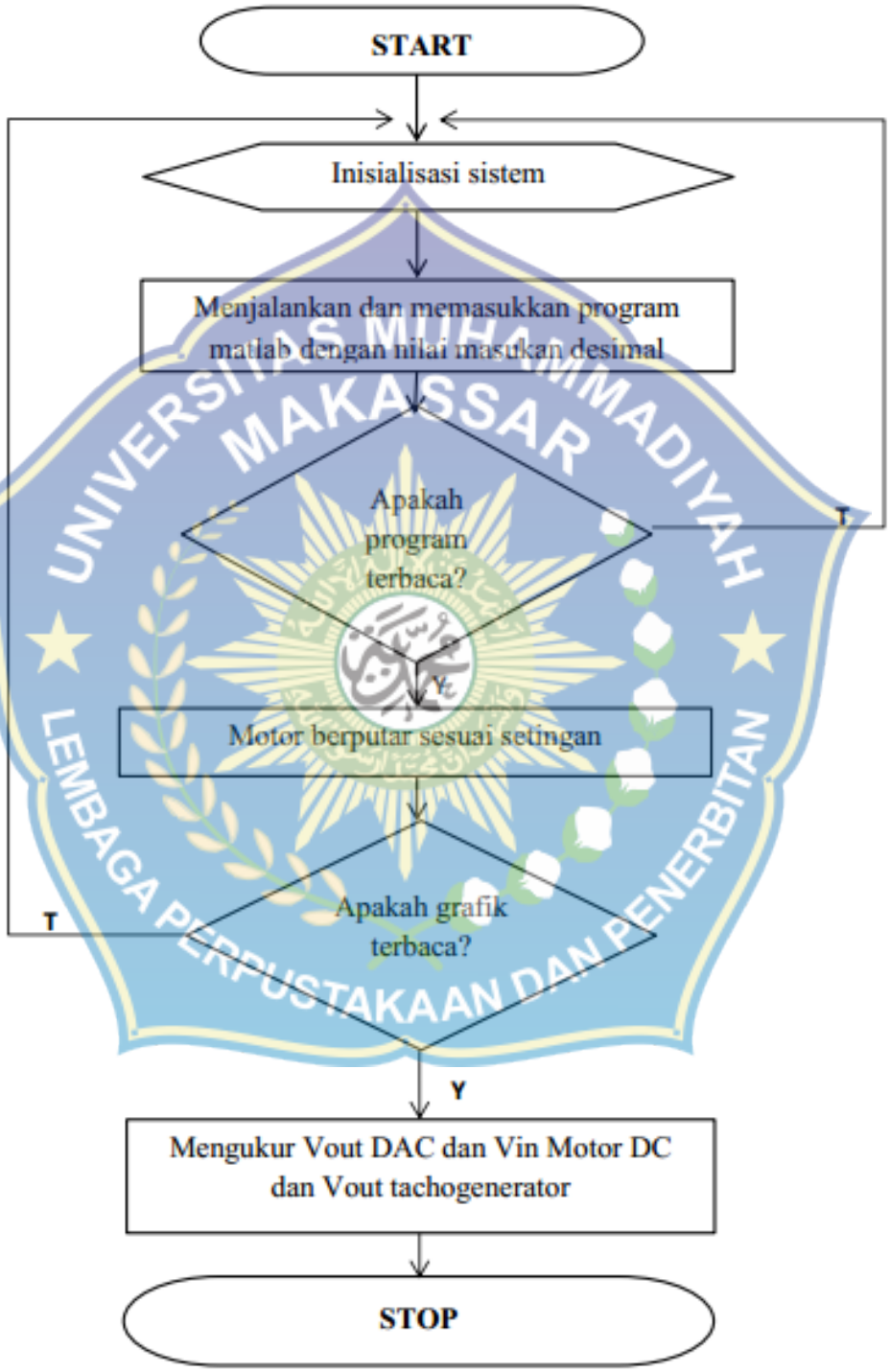
b. Tahap perancangan dan perakitan

Modul DAS pada penelitian ini dibangun dari beberapa subsistem seperti diperlihatkan pada gambar 3.1.



Gambar 3.1 Perangkat Akuisisi Data (DAS) menggunakan matlab

3.3 Flowchart Penelitian



3.4 Indikator Capaian

Perangkat yang dirancang dapat berjalan dengan software MATLAB 7.10.0 (R2010a) dengan menggunakan laptop dengan operasi sistem W7 32 bit.

BAB IV

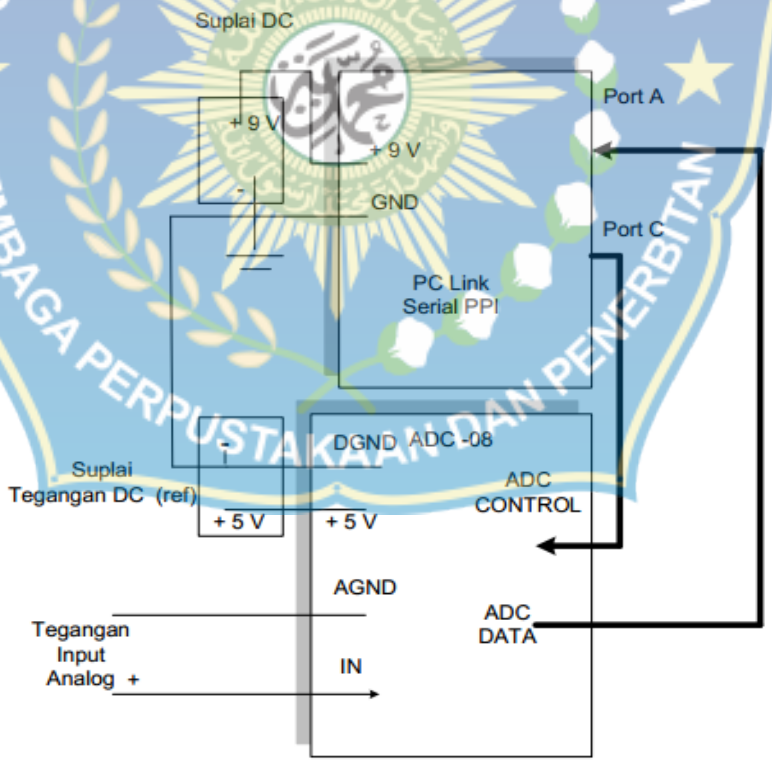
HASIL DAN PEMBAHASAN

4.1 Perancangan

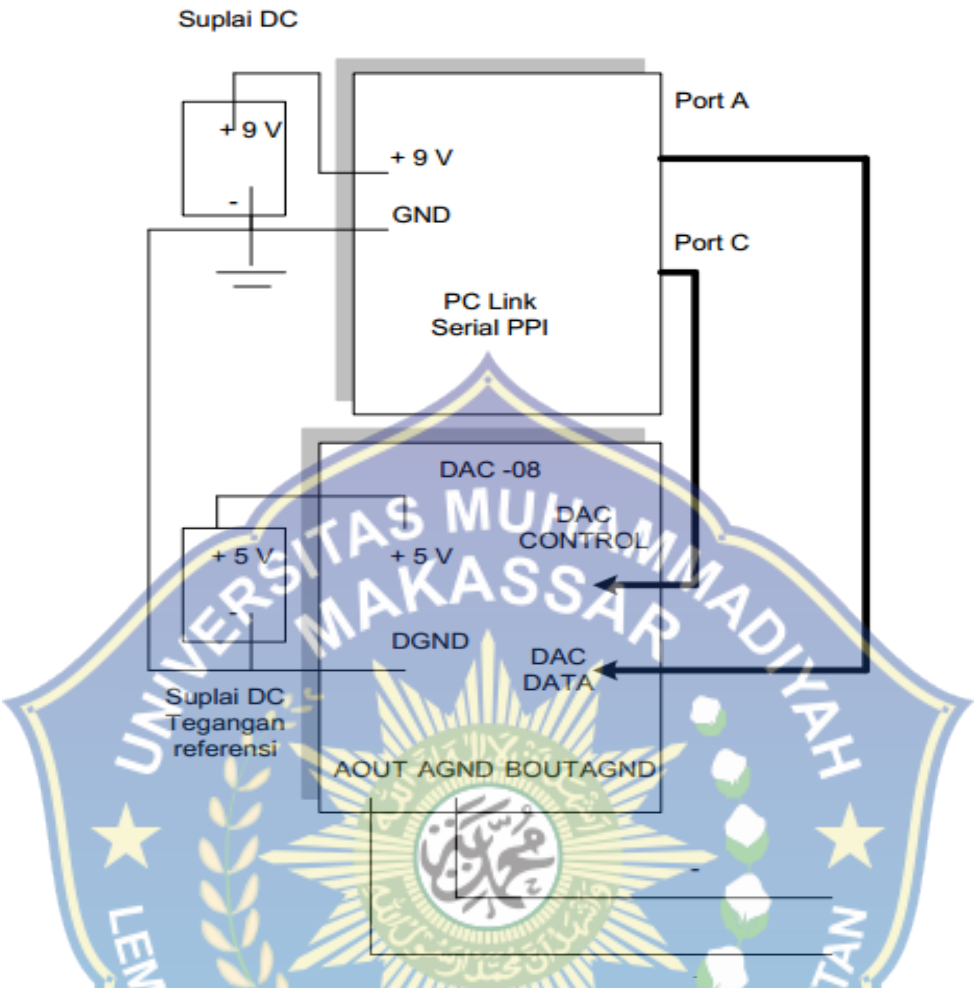
Setelah memiliki cukup referensi maka pada tahapan ini dibuat diagram perancangan sebagai berikut :

4.1.1 Tahapan pertama

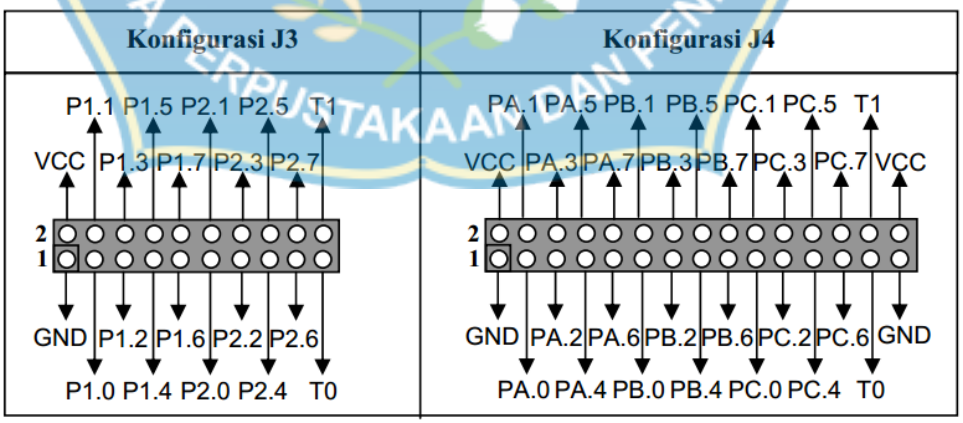
Mengoneksikan supply daya ke PPI dan selanjutnya ke rangkaian elektronika Analog to Digital Converter dan Digital to Analog Converter.



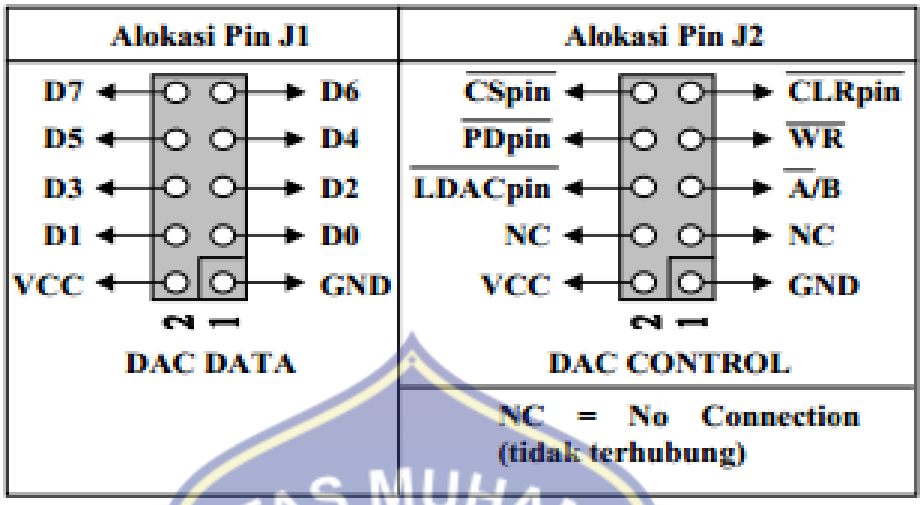
Gambar 4.1 Skema koneksi ADC-08 dengan PPI



Gambar 4.2 Skema koneksi DAC-08 dengan PPI



Gambar 4.3 alokasi port PPI



Gambar 4.4 alokasi port DAC

Pada gambar 4.1 Port A pada kit PC-Link Serial PPI digunakan untuk memberikan data 8 bit ke DAC-08 melalui DAC data. Koneksi antar pin diterangkan pada tabel dibawah ini.

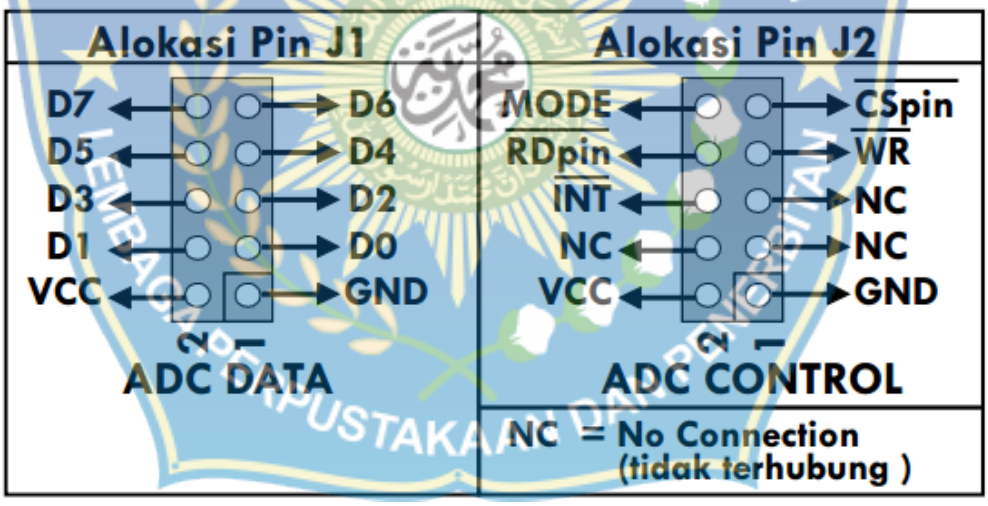
Tabel 4.1 koneksi antara PPI dan DAC (Port A ke DAC)

Port A (pada kit PPI)	DAC data (pada DAC)
PA.0	D0
PA.1	D1
PA.2	D2
PA.3	D3
PA.4	D4
PA.5	D5
PA.6	D6

PA.7	D7
GND	GND
VCC	VCC

Tabel 4.2 koneksi antara PPI dan DAC (Port C ke DAC)

Port C (pada kit PPI)	**DAC data (pada DAC)**
PC.7	$\overline{A}$/B
PC.6	$\overline{WR}$
PC.5	$\overline{LDAC}$pin



Gambar 4.5 Alokasi port ADC

pada kit PC link Serial PPI digunakan untuk menerima data 8 bit hasil konversi ADC melalui ADC DATA. Koneksi antar pin diperlihatkan pada tabel di bawah

Tabel 4.3 koneksi antara PPI dan ADC (Port 2 ke DAC)

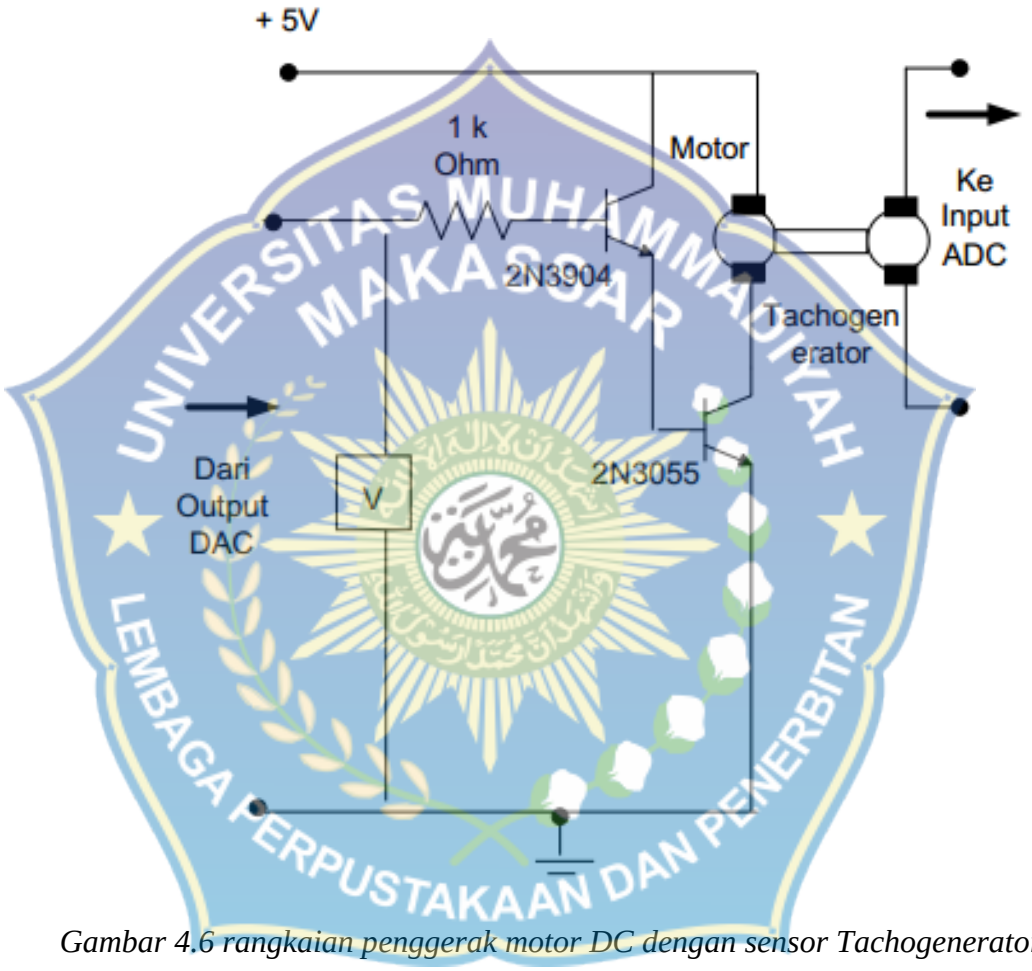
Port 2 **Pada Kit PPI**	**ADC data** **(Pada ADC)**
P2.0	D0
P2.1	D1
P2.2	D2
P2.3	D3
P2.4	D4
P2.5	D5
P2.6	D6
P2.7	D7
GND	GND
VCC	VCC

Tabel 4.4 koneksi antara PPI dan DAC (Port C ke DAC)

PORT C **(Pada PPI)**	**ADC CONTROL** **(Pada ADC)**
PC.0	$\overline{WR}$
PC.1	$\overline{CSpin}$
PC.2	MODE
PC.3	$\overline{RDpin}$

4.1.2 Tahapan ke dua

Membuat rangkaian elektronika penggerak motor DC dengan masukan output DAC sesuai dengan ketentuan seperti pada gambar dibawah ini :



Gambar 4.6 rangkaian penggerak motor DC dengan sensor Tachogenerator

4.2 Hasil Perancangan

Setelah melalui beberapa tahapan perancangan yang meliputi perancangan rangkaian elektronika, perancangan mekanik serta perancangan perangkat lunak (software), maka telah dihasilkan "judul skripsi" berikut tampilan dari hasil yang telah dirancang :

1 2 3 4 5 6 7

Berdasarkan gambar diatas maka dapat diberikan keterangan diantaranya sebagai berikut :

1. Power supply

   Power supply adalah alat atau sistem yang berfungsi untuk menyalurkan energi listrik atau bentuk energi jenis apapun yang sering digunakan untuk menyalurkan energi listrik.

2. Pc link PPI

   PC-Link serial PPI merupakan pengendali 40 bit jalur input/output melalui antar muka UART RS-232 yang dapat dihubungkan ke komputer secara langsung.

3. Digital To Analog Converter (DAC)

   Merupakan alat pengubah sinyal digital ke analog

4. Analog To Digital Converter

   Merupakan alat pengubah sinyal analog ke digital

5. Rangkaian motor DC

   Sebagai penggerak motor DC yang akan disuplay tegangan dari DAC

6. Motor DC

   Motor dc aplikasinya sebagai objek dari penelitian ini

7. Tacho generator

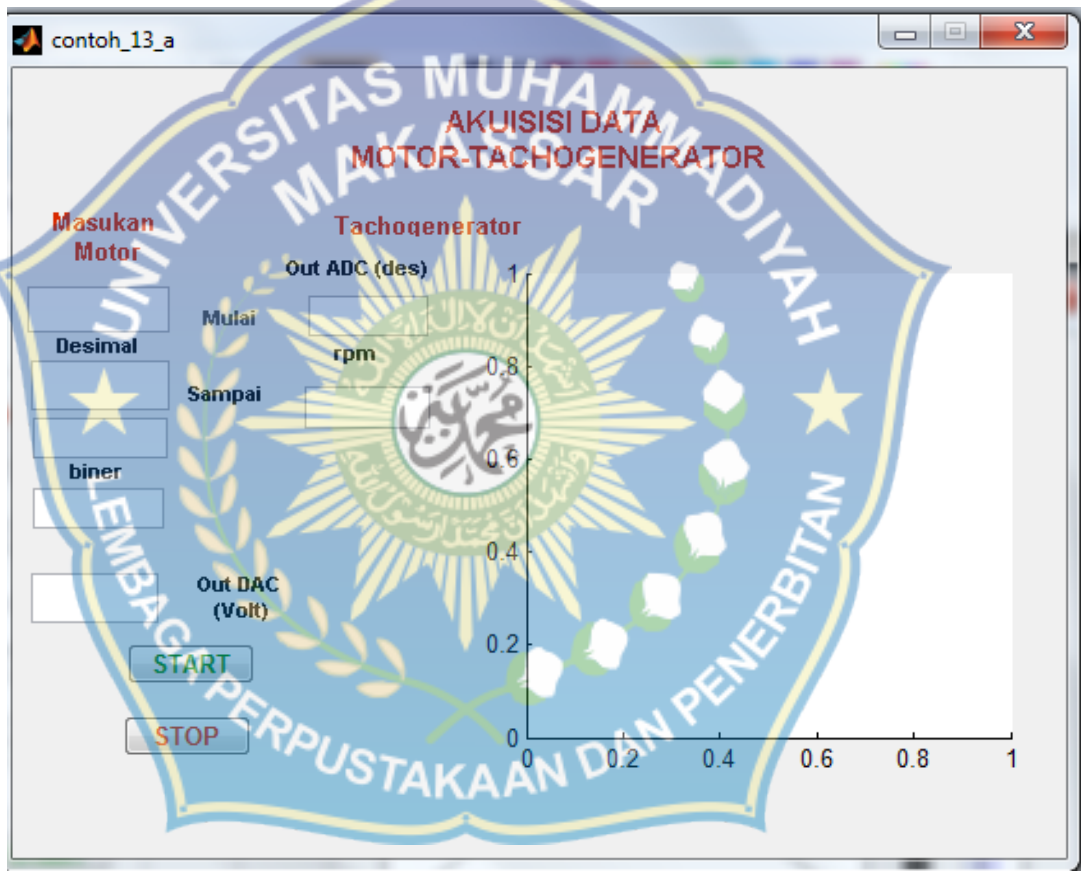
   Aplikasinya mengirim sinyal analog/ tegangan ke ADC

4.3 Analisis Perancangan

4.3.1 Pengujian rangkaian dengan menggunakan matlab

1. Tujuan pengujian

   Setelah merancang modul data akuisisi ini maka dilakukan pengujian untuk mengetahui batas maksimum dan minimum tegangan dalam pengendali motor DC yang di input menggunakan matlab.



Gambar 4.7 pengujian modul data akuisisi menggunakan matlab

2. Hasil pengujian pada DAC

Data hasil pengujian dapat dilihat pada tabel berikut :

Tabel 4.5 hasil pengujian masukan DAC menggunakan matlab

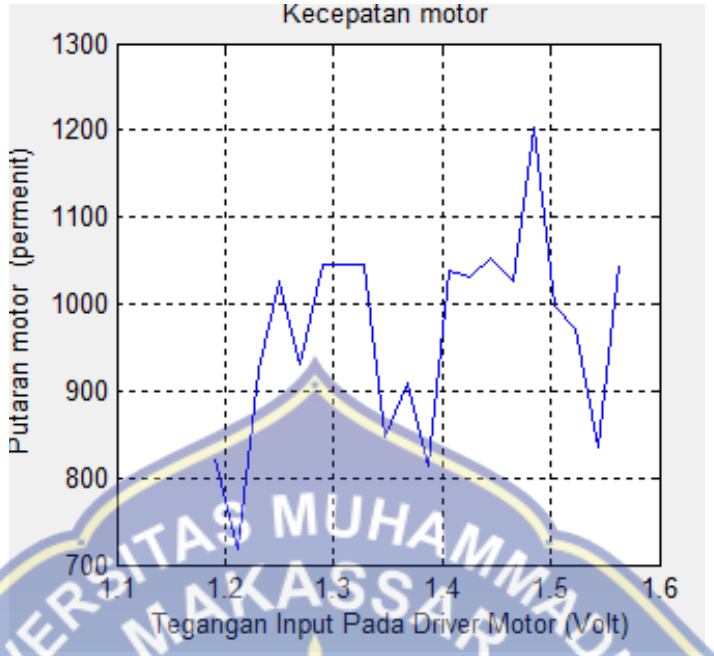
No	Masukan Decimal	Ekuivalen Biner	Keluaran DAC (Volt)	Keterangan
1	255	11111111	4.9 Volt	Motor ON
2	200	11001000	3.9 Volt	Motor ON
3	150	10010110	2.9 Volt	Motor ON
4	100	01100100	1.9 Volt	Motor ON
5	80	01010000	1.6 Volt	Motor ON
6	60	00111100	1.2 Volt	Motor ON
7	59	00111011	1.1 Volt	Motor OFF

3. Hasil pengujian pada ADC dan DAC

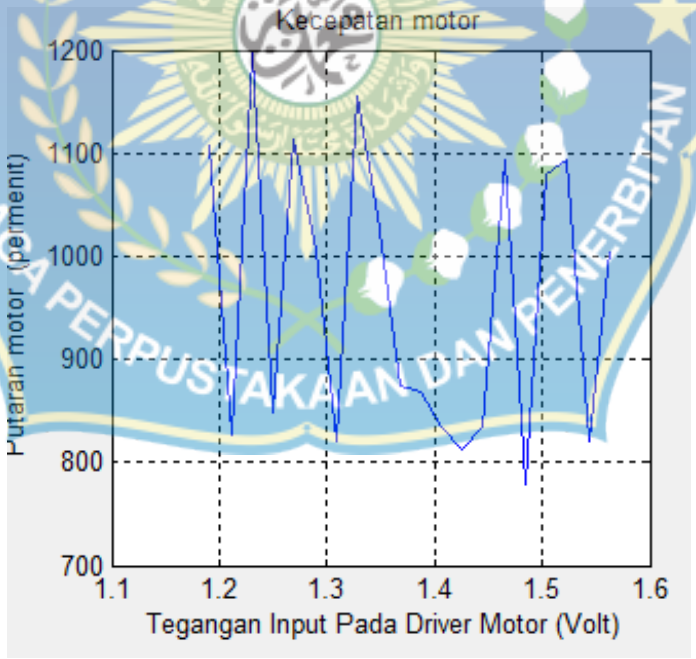
Data hasil pengujian dapat dilihat pada tabel berikut :

Tabel 4.5 hasil pengujian masukan ADC menggunakan matlab

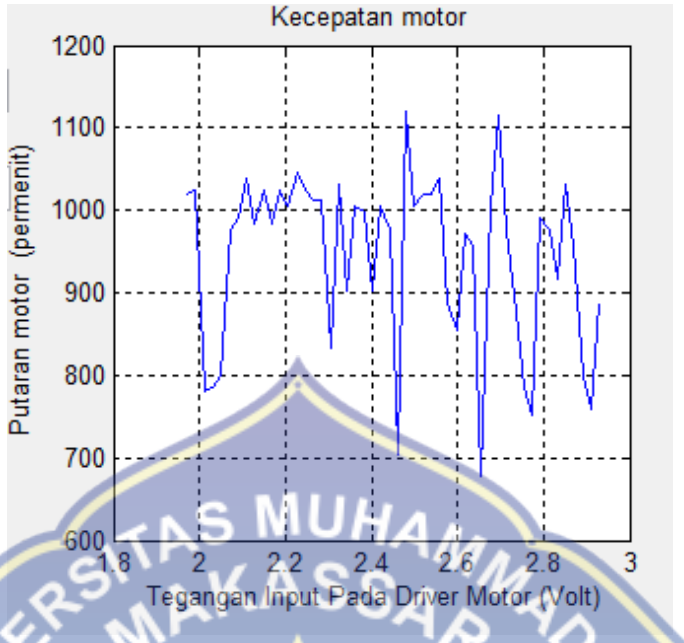
No.	Masukan Motor				Out Tegangan DAC (Volt)	Out ADC Desimal	RPM
	Desimal 1	Desimal 2	Biner 1	Biner 2			
1	60	80	00111100	01010000	1,6 Volt	153	1045,9
2	80	100	01010000	01100100	1,9 Volt	147	1004,88
3	100	150	01100100	10010110	2,9 Volt	130	888,672
4	150	200	10010110	11001000	3,9 Volt	182	1244,14
5	200	255	11001000	11111111	4,9 Volt	171	1168,95



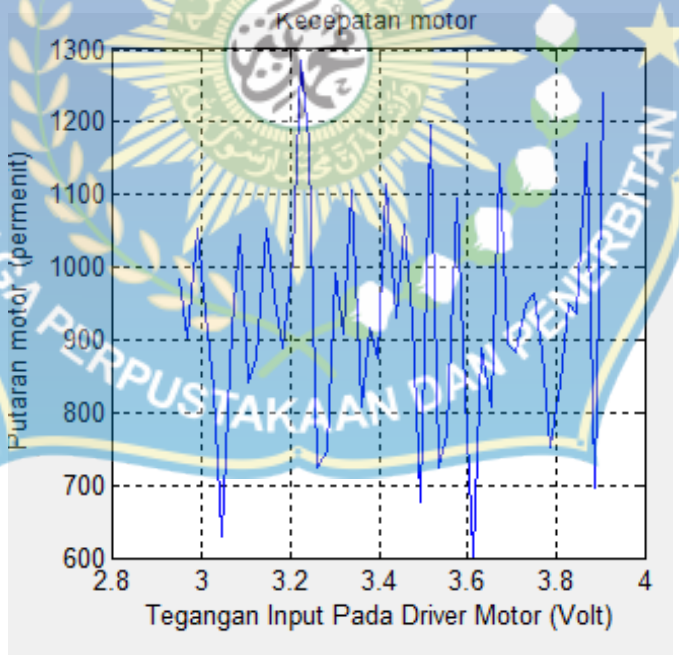
Gambar 4.8 grafik kecepatan putaran motor pada 1045,9 rpm



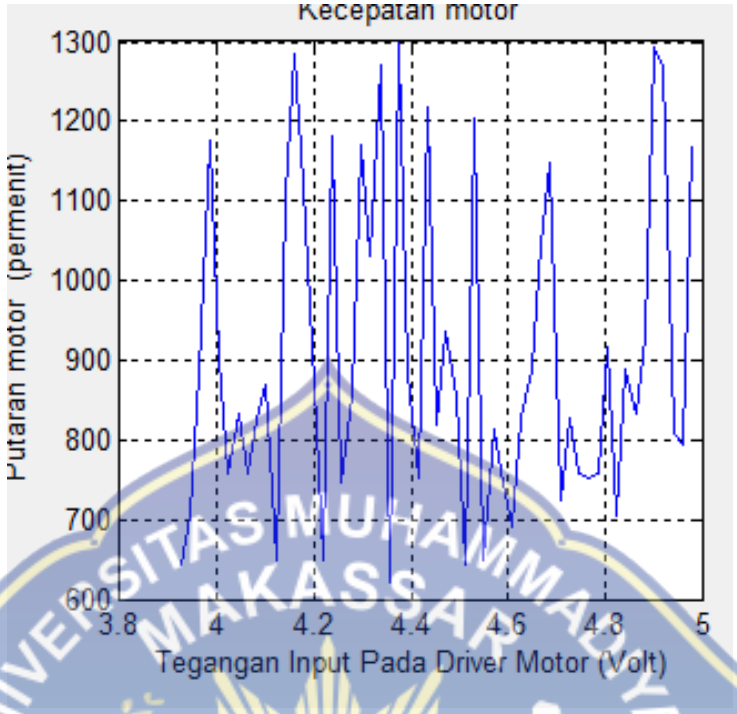
Gambar 4.9 grafik kecepatan putaran motor pada 1004,88 rpm



Gambar 4.10 grafik kecepatan putaran motor pada 888,672 rpm



Gambar 4.11 grafik kecepatan putaran motor pada 970,703



Gambar 4.12 grafik kecepatan putaran motor pada 1168,95

4.3.2 Pengukuran menggunakan DAC

Pengukuran dilakukan pada Vin rangkaian penggerak Motor DC dan Vm pada Motor DC. Data hasil pengukuran dapat dilihat pada tabel berikut :

Tabel 4.6 hasil pengukuran Vin dan Vm Motor DC

No	Masukan decimal	Ekuivalen Biner	Vin	Vm	Keterangan
1	255	11111111	3.8 VDC	3.4 VDC	Motor ON
2	200	11001000	3.0 VDC	3.5 VDC	Motor ON
3	150	10010110	2.3 VDC	3.8 VDC	Motor ON
4	100	01100100	1.6 VDC	4.0 VDC	Motor ON

5	80	01010000	1.3 VDC	4.2 VDC	Motor ON
6	60	00111100	1.0 VDC	3.1 VDC	Motor ON
7	59	00111011	0.9 VDC	0.0 VDC	Motor OFF

4.3.3 Perhitungan

Selisih hasil perhitungan dan pengukuran dapat mencapai 3 LSB (sekitar 59 mV dengan menggunakan VDD = +5 VDC)

Rumus output DAC

Vout = 2 x N x Vref/256

Dimana:

Vout : tegangan output

N : nilai register output DAC

Vref : tegangan referensi (= VDD/2)

1. Diketahui :

   N : 255

   Vref : 2.5 V

   Vout : . . . . ?

   Penyelesaian

   Vout : 2 x 255 x 2.5/256

   : 4.5 Volt

2. Diketahui :

   N : 200

   Vref : 2.5 V

   Vout : . . . . ?

   Penyelesaian

   Vout : 2 x 200 x 2.5/256

   : 3.9 Volt

3. Diketahui :

   N : 150

   Vref : 2.5 V

   Vout : . . . . ?

   Penyelesaian

   Vout : 2 x 150 x 2.5/256

   : 2.9 Volt

4. Diketahui :

   N : 100

   Vref : 2.5 V

   Vout : . . . . ?

   Penyelesaian

   Vout : 2 x 100 x 2.5/256

   : 1.9 Volt

5. Diketahui :

   N : 80

   Vref : 2.5 V

   Vout : . . . . ?

   Penyelesaian

   Vout : 2 x 80 x 2.5/256

   : 1.56 Volt

6. Diketahui :

   N : 60

   Vref : 2.5 V

   Vout : . . . . ?

   Penyelesaian

   Vout : 2 x 60 x 2.5/256

   : 1.17 Volt

7. Diketahui :

   N : 59

   Vref : 2.5 V

   Vout : . . . . ?

   Penyelesaian

   Vout : 2 x 59 x 2.5/256

   : 1.15 Volt

Tabel 4.7 hasil perhitungan Vout DAC

No	Masukan Decimal	Ekuivalen Biner	Vout DAC	Keterangan
1	255	11111111	4.5 Volt	Motor ON
2	200	11001000	3.9 Volt	Motor ON
3	150	10010110	2.9 Volt	Motor ON
4	100	01100100	1.9 Volt	Motor ON
5	80	01010000	1.56 Volt	Motor ON
6	60	00111100	1.17 Volt	Motor ON
7	59	00111011	1.15 Volt	Motor OFF

BAB V

PENUTUP

5.1 Kesimpulan

A. Dengan menggunakan masukan DAC , telah berhasil dirancang dan dibuat sebuah sistem akusisi data yang dapat mengatur kecepatan motor DC dengan menggunakan matlab. Sistem akuisisi yang dihasilkan memunyai spesifikasi sebagai berikut :

1. Tipe motor : Soder RF-300FA – 12350 5,9 VDC
2. Rentang masukan
   a. Minimum : 60 bit
      : 1,5625 Volt
      : 1066,41 rpm
   b. Maksimum : 255 bit
      : 4,9 Volt
      : 1168,95 rpm

B. Tegangan keluaran DAC menjadi masukan untuk rangkaian penggerak motor DC, oleh sebab itu putaran motor bisa lambat atau cepat tergantung tegangan keluaran DAC.

5.2 Saran

A. Perlu dibuat antar muka program dengan bahasa pemrograman yang lain, sehingga sistem akuisisi dapat dimanfaatkan oleh lebih banyak pengguna dengan penguasaan program yang beragam

B. Perlu dikembangkan rancangan yang menghasilkan sistem yang lebih tahan derau (Signa-to-Noise Ratio yang lebih tinggi), sehingga akurasi dan resolusinya dapat ditingkatkan.

C. Pada kasus pengukuran untuk analisis keteknikan kadang-kadang diperlukan suatu pengukuran banyak titik secara serempak, untuk tujuan ini dapat dikembangkan sistem akuisisi dengan tambahan perangkat sample and holdpada rangkaian masukan.

DAFTAR PUSTAKA

Iqbal, M., Imam, A.T, & Karyati, Y. *Sistem Akusisi Citra Stereo Menggunakan Matlab*. Depok: universitas Gunadarma

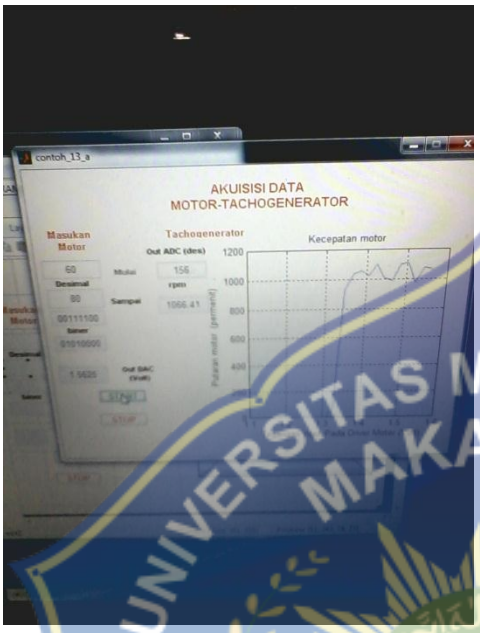
Labkomputer, Team. (2011). *Modul Praktikum Matlab*. Malang: divisi pendidikan dan pelatihan lembaga informasi dan komunikasi Universitas Muhammadiyah Malang

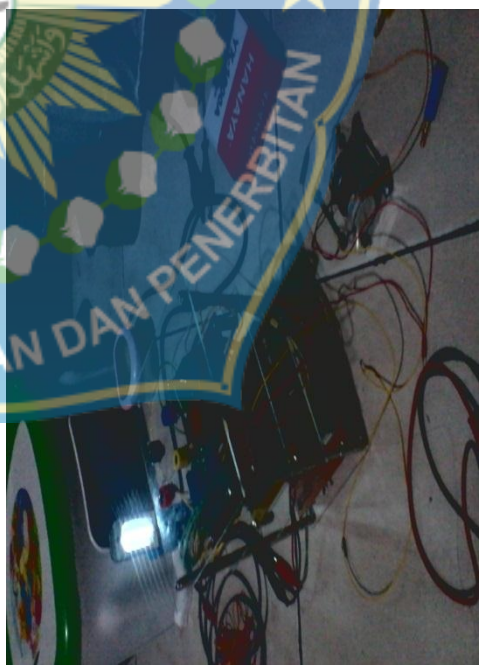
Murod, Hasan. (2005). *Perancangan Sistem Akuisisi Data Menggunakan Masukan Soundcard*. Yogyakarta: Universitas Gadjah Mada.

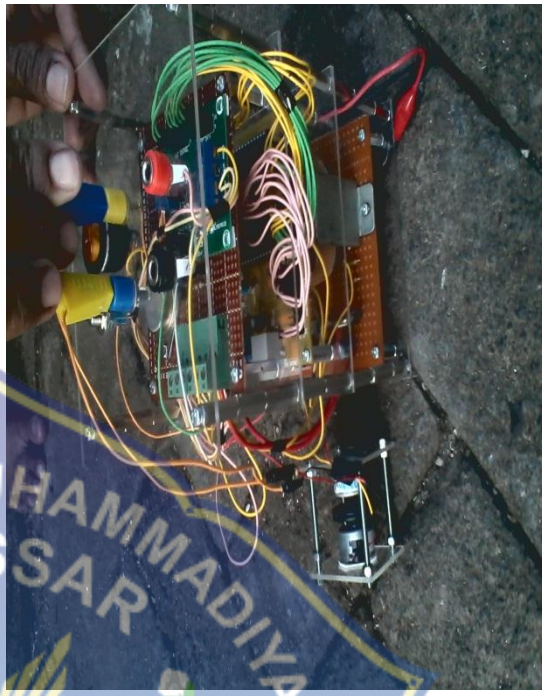
Manual Book *DT-I/O DAC – 08* (digital to analog converter). (2014)

Manual Book *PC – Link Serial PPI*. (2014)

LAMPIRAN KEGIATAN





LAMPIRAN PROGRAM

LAMPIRAN MANUAL BOOK

```
function varargout = contoh_13_a(varargin)
% CONTOH_13_A M-file for contoh_13_a.fig
%      CONTOH_13_A, by itself, creates a new CONTOH_13_A or raises
the existing
%      singleton*.
%
%      H = CONTOH_13_A returns the handle to a new CONTOH_13_A or
the handle to
%      the existing singleton*.
%
%      CONTOH_13_A('CALLBACK',hObject,eventData,handles,...) calls
the local
%      function named CALLBACK in CONTOH_13_A.M with the given input
arguments.
%
%      CONTOH_13_A('Property','Value',...) creates a new CONTOH_13_A
or raises the
%      existing singleton*.  Starting from the left, property value
pairs are
%      applied to the GUI before contoh_13_a_OpeningFcn gets called.
An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to contoh_13_a_OpeningFcn via
varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows
only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help contoh_13_a

% Last Modified by GUIDE v2.5 16-Oct-2014 03:33:00

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @contoh_13_a_OpeningFcn, ...
                   'gui_OutputFcn',  @contoh_13_a_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
```

```
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before contoh_13_a is made visible.
function contoh_13_a_OpeningFcn(hObject, eventdata, handles,
varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to contoh_13_a (see VARARGIN)

% Choose default command line output for contoh_13_a
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes contoh_13_a wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = contoh_13_a_OutputFcn(hObject, eventdata,
handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;



function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%        str2double(get(hObject,'String')) returns contents of edit1
as a double
%_____________________________________________________________

mulai = str2num (get (handles.edit1, 'String'));
handles.mulai = mulai;
```

```
guidata (hObject, handles);


% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit2 as text
%        str2double(get(hObject,'String')) returns contents of edit2
as a double
%_____________________________________________________________
sampai = str2num (get (handles.edit2, 'String'));
handles.sampai = sampai;
guidata (hObject, handles);


% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function edit3_Callback(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit3 as text
%        str2double(get(hObject,'String')) returns contents of edit3
as a double


% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit5_Callback(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit5 as text
%        str2double(get(hObject,'String')) returns contents of edit5
as a double


% --- Executes during object creation, after setting all properties.
function edit5_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function edit6_Callback(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit6 as text
%        str2double(get(hObject,'String')) returns contents of edit6
as a double


% --- Executes during object creation, after setting all properties.
function edit6_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit9_Callback(hObject, eventdata, handles)
% hObject    handle to edit9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit9 as text
%        str2double(get(hObject,'String')) returns contents of edit9
as a double


% --- Executes during object creation, after setting all properties.
function edit9_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit10_Callback(hObject, eventdata, handles)
% hObject    handle to edit10 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit10 as text
%        str2double(get(hObject,'String')) returns contents of
edit10 as a double


% --- Executes during object creation, after setting all properties.
function edit10_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit10 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit11_Callback(hObject, eventdata, handles)
% hObject    handle to edit11 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit11 as text
%        str2double(get(hObject,'String')) returns contents of
edit11 as a double


% --- Executes during object creation, after setting all properties.
function edit11_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit11 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
%_____________________________________________________________
mulai =handles.mulai;
sampai =handles.sampai;
bin_mulai =dec2bin (mulai,8);
bin_sampai =dec2bin (sampai,8);

set(handles.edit3, 'string', bin_mulai);
set(handles.edit12, 'string', bin_sampai);

%____________________ COBA DAQ ____________________________
% dummy_input = DAQ_DAC_up (data_input);
% dummy_input = DAQ_DAC_finite (data_input);
% dummy_input = DAQ_DAC_down (data_input);
 UP_DAC = DAQ_DAC_up (255);
     tunda (2);

   %  sampai =77;
   %  mulai =66;
     delta =sampai-mulai;

     bb = delta;
     aa = 1;



for siklus = aa:bb
    k=siklus;

 %_____________________________________________________________
 %  input dari program, output digital to analog ke motor
 %_____________________________________
 v_driver = mulai+siklus;
 %_________________________________
     finite_DAC = DAQ_DAC_finite ( v_driver);
```

```
 V_DAC_in_des (k)  = v_driver; % desimal
 Vref =5/2;

Vout (k) = 2* V_DAC_in_des (k) * Vref/256;
  V_DAC_out (k) = Vout (k); % Volt

 %_____________________________________________________________
 % Input analog dari tacho ke analog to digital converter
     des_V = DAQ_ADC_Volt (1000);


     V_ADC_out_des (k) =des_V (1);
     V_ADC_out_volt (k) =des_V (2)
     tunda (0.01)

end
% Down_DAC = DAQ_DAC_down ( 0);
     tunda (1)
% kalibrasi 1 V =350 rpm
rpm =  V_ADC_out_volt *350;

 disp (' _______DAC_____________ADC_____________')
 disp ('  V_output (Volt)      rpm  Vout (desimal)')
data_out =[V_DAC_out' rpm'  V_ADC_out_des']

% kalibrasi 1 V =350 rpm
teg_driver_motor = V_DAC_out;
plot (teg_driver_motor, rpm);
xlabel (' Tegangan Input Pada Driver Motor (Volt)')
ylabel (' Putaran motor  (permenit)')
title (' Kecepatan motor ')
grid

%_____________________________________________________________

V_ADC_bin =V_ADC_out_des (k);
set(handles.edit9, 'string', V_DAC_out (k));
set(handles.edit10, 'string', V_ADC_bin);
set(handles.edit11, 'string', rpm (k));



% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
%_____________________________________________________
Down_DAC = DAQ_DAC_down ( 0);
     tunda (1)
delete(handles.figure1)

function edit12_Callback(hObject, eventdata, handles)
% hObject    handle to edit12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit12 as text
%        str2double(get(hObject,'String')) returns contents of
edit12 as a double


% --- Executes during object creation, after setting all properties.
function edit12_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

12 CONVERTER ANALOG KE DIGITAL (ADC)

Converter digital ke analog adalah sebuah alat yang berfungsi untuk mengubah sinyal analog (tegangan dan arus listrik) menjadi sinyal digital. Misalnya mengubah tegangan analog 0 Volt sampai 5 Volt menjadi data digital 8 bit 00000000 biner sampai 11111111 biner. Pada bagian ini akan dijelaskan program kit Analog to Digital Converter berbasis ADC0820 menggunakan MATLAB

12.1 ADC-08

ADC-08 adalah Analog to Digital Converter berbasis ADC0820 yang membutuhkan catu daya tunggal +5 VDC. Aplikasinya antara lain untuk pendeteksi tegangan dan mengubah data sensor analog menjadi digital.

Fitur dan spesifikasi teknis :

- Resolusi ADC 8-bit.
- Tegangan kerja (VCC) = Tegangan referensi (Vref) = +5 VDC.
- Fungsi track-and-hold yang terintegrasi.

- Tanpa clock eksternal.
- Memiliki tiga operasi:
  - RD (Read) Mode
  - WR-RD (Write-Read) Mode
  - WR-RD Stand Alone Operation
- Waktu Konversi 2,5 µs pada Read Mode dan 1,5 µs pada Write-Read Mode dan WRRD Stand Alone Operation.
- Range input 0 VDC hingga +5 V (dengan VCC = +5 VDC).
- Selisih hasil pengukuran dan penghitungan maksimum 1LSB (sekitar 20 mV dengan menggunakan VCC = +5 VDC).
- Tidak membutuhkan pengaturan zero atau full-scale adjust.
- Antarmuka paralel dengan level tegangan CMOS atau TTL.
- Dapat dihubungkan melalui pin I/O ataupun Intel System Bus (System Bus hanya mendukung Write-Read Mode).

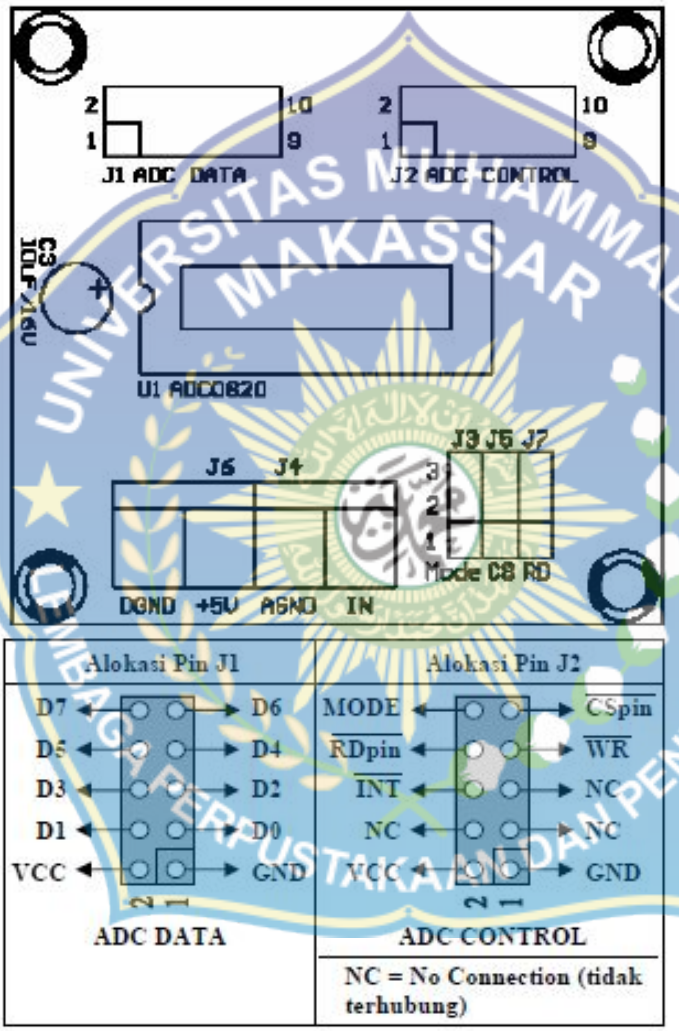
Cara menghitung output ADC

Rumus output ADC: N = Vin x 256 /Vref

dimana N : nilai register output ADC

Vin : tegangan input

Vref: tegangan referensi (= VCC)

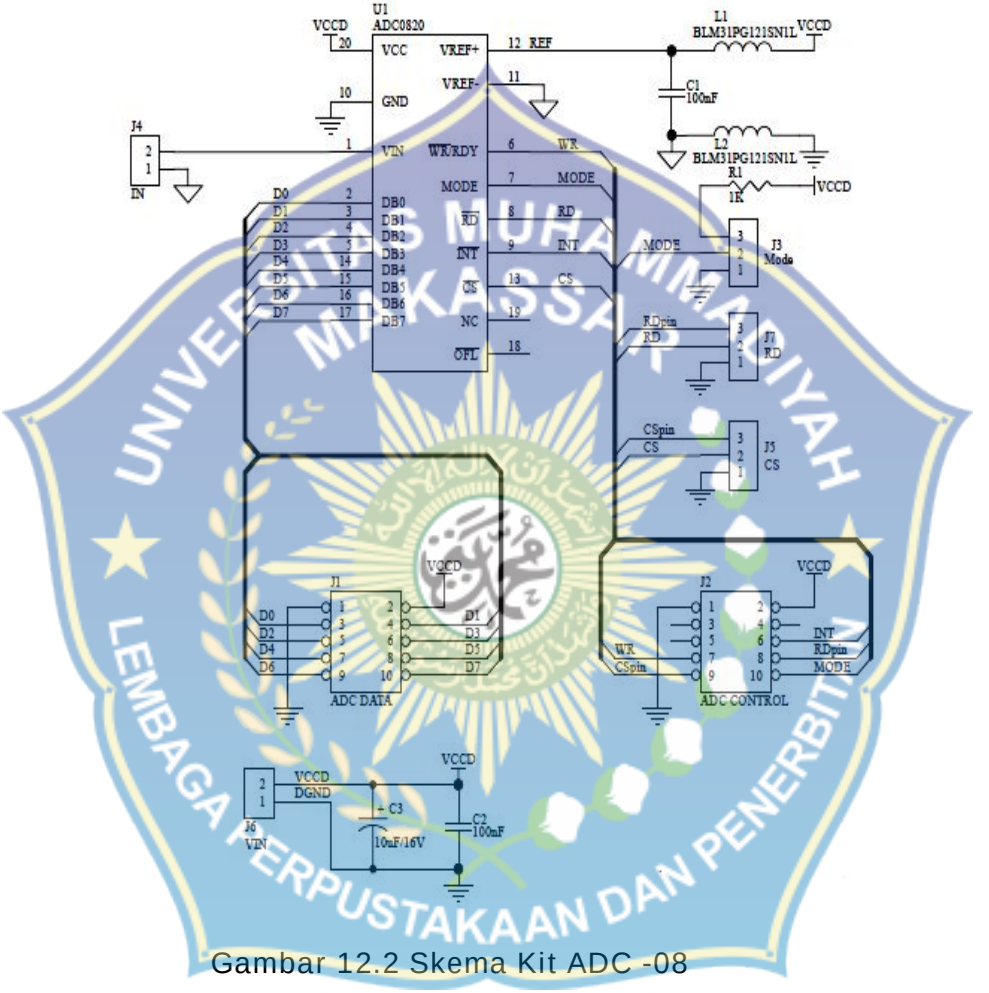


Gamabr 12.1 Tata letak komponen kit ADC -08

Selisih hasil penghitungan dan pengukuran dapat mencapai 1 LSB (sekitar 20 mV dengan menggunakan VCC = +5 VDC)

Pengaturan Jumper

Jumper	Posisi	Keterangan
MODE (J3)	1-2	MODE terhubung ke Ground, hanya dapat menggunakan Read Mode
	2-3	MODE terhubung ke VCC via resistor, semua mode dapat digunakan
CS (J5)	1-2	CS (chip select) terhubung ke Ground, modul ADC selalu terpilih
	2-3	CS terhubung ke pin CSpin, modul ADC dapat dipilih oleh program
RD (J7)	1-2	RD terhubung ke Ground, hanya dapat menggunakan Write-Read Mode atau WR-RD Stand Alone Operation
	2-3	RD terhubung ke pin RDpin, semua mode dapat digunakan



Gambar 12.2 Skema Kit ADC -08

(Sumber Quick Start DAC -08, Innovative Electronics)

Contoh 12.1

Dengan menggunakan persamaan untuk keluaran ADC-08, tentukan keluaran ADC dalam bentuk data 8 bit (D7-D0) jika ADC mendapat masukan tegangan analog 0, 1, 2, 3, dan 4 Volt. Tegangan referensi 5 Volt.

Solusi

Dari persamaan

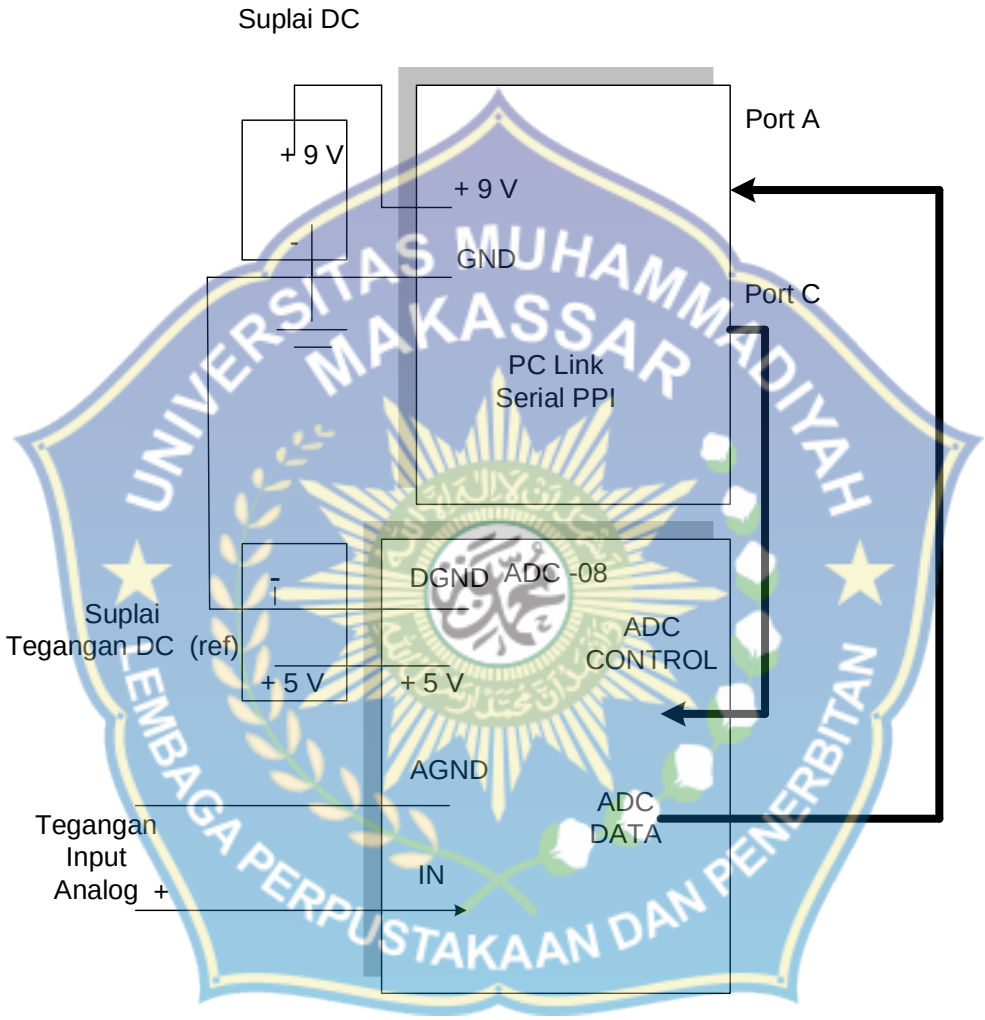
N = Vin x 256 / Vref

N = Vin 256/5

Vin	N (desimal)	D7 D6 D5 D4 D3 D2 D1 D0 (Data)
0	0	00000000
1	51.2	00110011
2	102.4	01100110
3	153.6	10011001
4	204.8	11001100

12.2 PENGUJIAN ADC-08 DENGAN PC Link Serial PPI

Skema kit ADC-08 yang dihubungkan dengan kit PC Link Serial PPI 8255 diperlihatkan pada gambar 12.3



Gambar 12.3 Skema koneksi ADC- 08 dengan PPI

Pada gambar 12.3, port 2 pada kit PC link Serial PPI digunakan untuk menerima data 8 bit hasil konversi ADC melalui ADC DATA. Koneksi antar pin diperlihatkan pada tabel di bawah

Port 2 (Pada Kit PPI)	ADC DATA (Pada ADC)	
P2.0	D0	
P2.1	D1	
P2.2	D2	
P2.3	D3	
P2.4	D4	
P2.5	D5	
P2.6	D6	
P2.7	D7	
GND	GND	
VCC	VCC	
Port C (Pada PPI)	ADC CONTROL (Pada ADC)	
PC.0	$\overline{WR}$	
PC.1	$\overline{CSpin}$	
PC.2	MODE	
PC.3	$\overline{RDpin}$	

Untuk memogram port 2 agar berfungsi sebagai input (mmbaca data dari keluaran ADC) dilakukan menurut urutan berikut (lihat bab 4).

- Instruksi agar port 2 sebagai input pada kit PPI
  fwrite (s,49)
- Instruksi membaca data
  fread(s,1,'uint8');

Seperti yang diperlihatkan pada gambar 12.3 untuk keperluan kontrol ADC digunakan pin 0 sampai 3 pada port C dengan operasi set/reset. Urutan programnya adalah

- Mengirin Instruksi ke Control Word
  fwrite(s,38)
- Intruksi set/reset pada pin PC.0
  fwrite(s,0)(logika 0) atau
  fwrite(s,1) (logika 1)
- Intruksi set/reset pada pin PC.1
  fwrite(s,2), atau
  fwrite(s,3)
- Intruksi set/reset pada pin PC.2
  fwrite(s,4); % PC.6
  fwrite(s,5);% PC.5
- Intruksi set/reset pada pin PC.3
  fwrite(s,4); % PC.6

fwrite(s,5);% PC.5

Contoh 12. 2

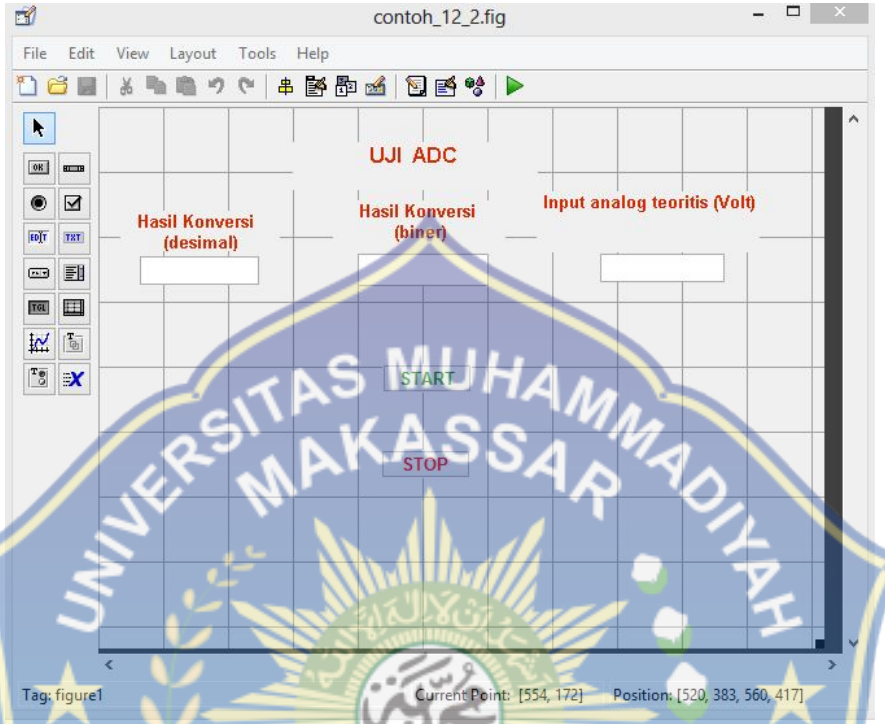
Rangkaian gambar 12.3 memperoleh tegangan masukan analog 0 sampai 5 Volt yang diatur melalui potensiometer. Posisi jumper pada kit DAC diatur sesuai tabel berikut

Jumper	Posisi
MODE (J3)	2-3
CS (J5)	2-3
RD (J7)	2-3

Buat program MATLAB dalam format GUI untuk membaca data biner 8 bit hasil konversi ADC untuk masukan tegangan analog antara 0-5 Volt. Tampilan GUI diperlihatkan pada gambar 12.4.

Solusi

Koneksi komputer atau laptop ke kit PC Link Serial PPI menggunakan kabel USB to Serial BAFO, dengan nomor Com serial yang terdeteksi adalah Com 11.



Gambar 12.4 Tampilan GUI Contoh 12.2

Static Text 1	String	UJI ADC
	FontSize	12.0
	FontWeight	bold
	ForegroundColor	pilih
Static Text 2	String	Hasil Konversi (desimal)
	FontSize	10.0

	FontWeight	bold
	ForegroundColor	pilih
Static Text 3	String	Hasil Konversi (biner)
	FontSize	10.0
	FontWeight	bold
	ForegroundColor	pilih
Static Text 4	String	Input analog teoritis (Volt)
	FontSize	10.0
	FontWeight	bold
	ForegroundColor	pilih
Edit Text 1	String	kosongkan
Edit Text 2	String	kosongkan
Edit Text 3	String	kosongkan
Push Button 1	String	START
	FontSize	10.0
	FontWeight	bold
	ForegroundColor	pilih
Push Button 2	String	STOP
	FontSize	10.0
	FontWeight	bold
	ForegroundColor	pilih

- Klik kanan komponen Push button 1, kemudian pilih View Callbacks > Callback maka muncul fungsi Push button 1 Callback

```
function pushbutton1_Callback(hObject, eventdata, handles)

[isikan]
%_____________________________________________
clc
%--------------jalankan m-file 2x-------------------------

for ulang =1:2
data_adc =fungsi_ADC (ulang);
end

u=size (data_adc);
for n=1:u(2)
  if data_adc(n)~=250

    data_baca  =data_adc (n);
  end
end

biner_ADC =dec2bin (data_baca,8)
% RUMUS ADC N= Vin* 256/Vref
%  Vref = VCC =+ 5 Volt
vref =5;
Vin_teoritis = data_baca * vref /256
```

```
set(handles.edit1, 'string', data_baca);
set(handles.edit2, 'string', biner_ADC);
set(handles.edit3, 'string', Vin_teoritis);
%_____
```

Gambar 12.5 Hasil program Contoh 12.2

- Klik kanan komponen Push button 2, kemudian pilih View Callbacks > Callback maka muncul fungsi Push button 2 Callback

```
function pushbutton2_Callback(hObject, eventdata, handles)

[isikan]
delete(handles.figure1)
```

m-file fungsi_ADC

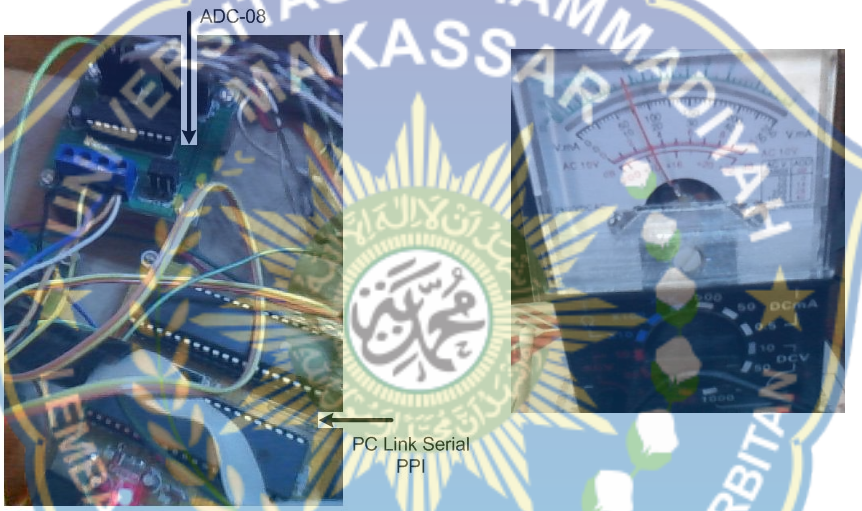
```
function data_adc =fungsi_ADC (xxx);

% Akses Bit Set/ Reset
% BACA data analog menjadi data digital
% masukan banyaknya data array
% inisialisasi serial com.
%=========================================================
s = serial('COM11'); %assigns the object s to serial port

set(s, 'InputBufferSize', 128); %number of bytes in inout buffer
set(s, 'FlowControl', 'none');
set(s, 'BaudRate', 9600);
set(s, 'Parity', 'none');
set(s, 'DataBits', 8);
set(s, 'StopBit', 1);
set(s, 'Timeout',100);
```

```
disp(get(s,'Name'));
prop(1)=(get(s,'BaudRate'));
prop(2)=(get(s,'DataBits'));
prop(3)=(get(s, 'StopBit'));
prop(4)=(get(s, 'InputBufferSize'));
prop;
disp([num2str(prop)]);
%============buka serial port====================
fopen(s);          %opens the serial port
% delay ambil sampel t, delay control t0
t=0.1;
t0=0.00001;
%_____________________________________

% masukkan nilai digital ekivalen desimal
% akses per byte desimal ke port A
% inisialisasi 128, control word 38, tulis port A 32
%
j=0;

for step =1:3
for k=0:2 ; % data
  j=j+1;

%---------------------KONTROL ADC ---------
% WR PC.0,  CS pin PC.1, Mode PC.2, RD pin PC.3
% low  0/1     low    2/3      high  4/5       low 6/7
```

```
% jumper 2-3 semua


% port  akses perbit
%          fwrite(s,38 );% cont word
 %           tunda (t0);
% tidak pakai fwrite (s,128);% inisialisasi basic i/o
% jika akses perbit port c
          % instr write port C  pebit


%--------------------KONTROL ADC ---------

        fwrite(s,38 );% cont word
       tunda (t0);
%-----------------------------------------

% nilai desimal pin  0-1  2-3   4-5 6-7
 %----------------------------------------
if step ==1
 fwrite(s,2 );% pc.0
end
if step==2
     fwrite(s,3 );% pc.0
else

 fwrite(s,2 );% pc.0
end
```

```
% DATA BACA ADC
% _____________
 % data baca dari port 2
       fwrite (s,49); % baca dari port 2
          tunda (t0);



          data_baca  =fread(s,1,'uint8');
          data_adc (j) =data_baca;

%---------------------KONTROL ADC ---------

         fwrite(s,38 );% cont word
        tunda (t0);
%-----------------------------------------

 fwrite(s,0 );% pc.0

end
end
dummy_xxx = xxx* 0;
fclose(s);
delete(s)
```

Pada gambar 12.5 ditampilkan hasil konversi ADC yaitu 10011111 biner atau 159 desimal. Tegangan yang masuk ke perangkat ADC berdasarkan rumus yang diberikan sebelumnya adalah 3.10547 Volt. Masukan analog yang masuk ke ADC hasil pengukuran Voltmeter juga sekitar 3 Volt, seperti tampak pada gambar 12.6.



Gambar 12.6 Hasil pengukuran tegangan masuk pada ADC

Latihan

1. Pada Rangkaian 12.3, jika port A digantikan dengan port 1 untuk membaca data konversi ADC, dan port 2 dipakai untuk mengontrol ADC control pada kit

ADC menggantikan port C, tulislah kode MATLAB untuk akes ke dua port tersebut.

2. Sama dengan contoh 12.2, ukur tegangan yang masuk pada input ADC dengan Voltmeter kemudian jalankan program dan catat hasil konversi ADC untuk tegangan 1, 2, 3, dan 4 Volt.

DT-I/O ADC-08

Analog to Digital Converter

ADC-08 adalah Analog to Digital Converter berbasis ADC0820 yang membutuhkan catu daya +5 VDC. Aplikasinya antara lain untuk pendeteksi tegangan dan mengubah data sensor analog menjadi digital.

Fitur & Spesifikasi Teknis

1. Resolusi ADC 8-bit.
2. Tegangan kerja (VCC) = Tegangan referensi (Vref) = +5 VDC.
3. Fungsi track-and-hold yang terintegrasi.
4. Tanpa clock eksternal.
5. Memiliki tiga operasi:
   - RD (Read) Mode
   - WR-RD (Write-Read) Mode
   - WR-RD Stand Alone Operation
6. Waktu Konversi 2,5 µs pada Read Mode dan 1,5 µs pada Write-Read Mode dan WR-RD Stand Alone Operation.
7. Range input 0 VDC hingga +5 V (dengan VCC = +5 VDC).
8. Selisih hasil pengukuran dan penghitungan maksimum 1 LSB (sekitar 20 mV dengan menggunakan VCC = +5 VDC).
9. Tidak membutuhkan pengaturan zero atau full-scale adjust.
10. Antarmuka paralel dengan level tegangan CMOS atau TTL.
11. Dapat dihubungkan melalui pin I/O ataupun Intel System Bus (System Bus hanya mendukung WR-RD Mode).
12. Dilengkapi rutin-rutin siap pakai dalam bahasa Assembly untuk DT-51™ Low Cost Series dan DT-51™ Minimum System ver 3.0.

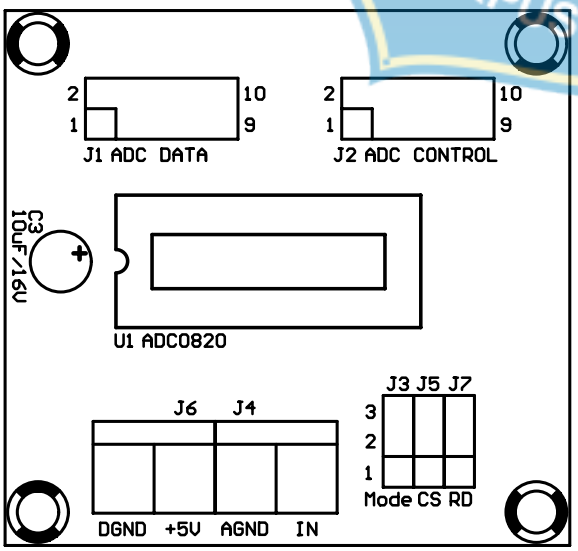
Cara Menghitung Output ADC

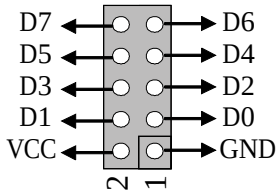
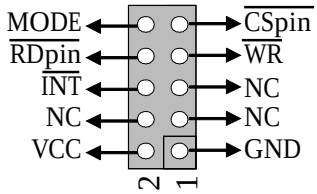
Rumus output ADC: $N = \frac{Vin}{Vref} \times 256$

dimana
N : nilai register output ADC
Vin : tegangan input
Vref : tegangan referensi (= VCC)

Selisih hasil penghitungan dan pengukuran dapat mencapai 1 LSB (sekitar 20 mV dengan menggunakan VCC = +5 VDC).

Tata Letak dan Koneksi



Alokasi Pin J1	Alokasi Pin J2
	
ADC DATA	ADC CONTROL
	NC = No Connection (tidak terhubung)

Koneksi DT-51™ Low Cost Series dengan ADC-08 secara I/O

DT-51™ Low Cost Series		ADC-08	
Port 1		ADC-DATA	
Pin	Nama	Pin	Nama
1	GND	1	GND
2	VCC	2	VCC
3	P1.0	3	D0
4	P1.1	4	D1
5	P1.2	5	D2
6	P1.3	6	D3
7	P1.4	7	D4
8	P1.5	8	D5
9	P1.6	9	D6
10	P1.7	10	D7

Port 3		ADC-CONTROL	
Pin	Nama	Pin	Nama
1	GND	1	GND
2	VCC	2	VCC
6	P3.3	6	INT
7	P3.4	7	WR
8	P3.5	8	RDpin
9	P3.6 (pin ini tidak tersedia pada DT-51™LC Nano System)	9	CSpin (pada DT-51™ LC Nano System, jumper CS dapat dihubungkan ke posisi 1-2)
10	P3.7	10	MODE

Koneksi DT-51™ Minimum System ver 3.0 dengan ADC-08 secara System Bus

DT-51™ Minimum System ver 3.0		ADC-08	
Data & CS		ADC-DATA	
Pin	Nama	Pin	Nama
9	AD0	3	D0
10	AD1	4	D1
11	AD2	5	D2
12	AD3	6	D3
13	AD4	7	D4
14	AD5	8	D5
15	AD6	9	D6
16	AD7	10	D7

Data & CS		ADC CONTROL	
Pin	Nama	Pin	Nama
15	CS6	9	CSpin

Control		ADC CONTROL	
Pin	Nama	Pin	Nama
1	VCC	2	VCC
2	GND	1	GND
4	I1	6	INT
7	WR	7	WR
8	RD	8	RDpin

Keterangan: MODE tidak dihubungkan ke I/O DT-51™ Minimum System ver 3.0 sehingga harus dipilih menggunakan jumper.

Pengaturan Jumper

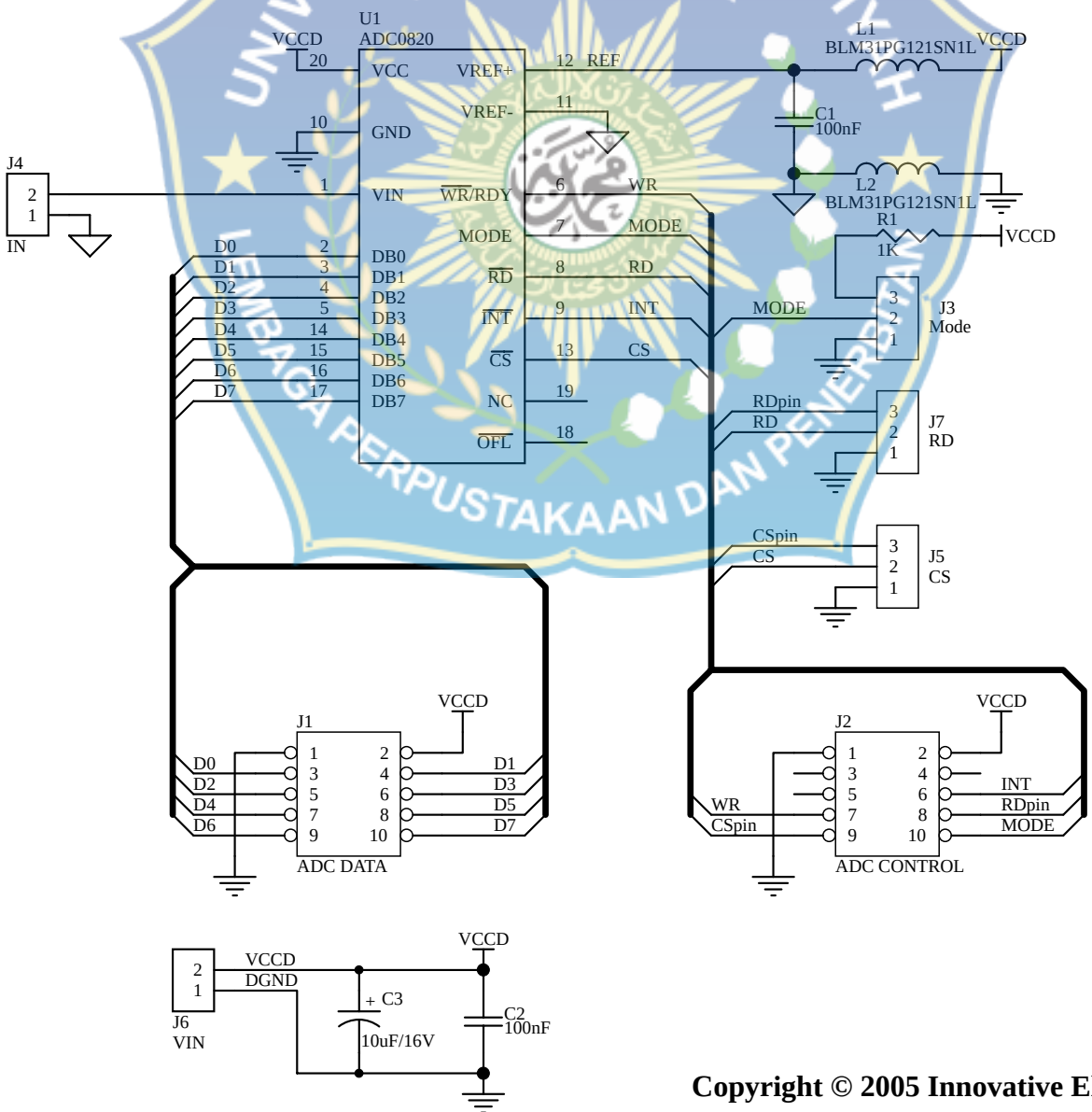
Jumper	Posisi	Keterangan
MODE (J3)	1-2	MODE terhubung ke Ground, hanya dapat menggunakan Read Mode
	2-3	MODE terhubung ke VCC via resistor, semua mode dapat digunakan
CS (J5)	1-2	CS (chip select) terhubung ke Ground, modul ADC selalu terpilih
	2-3	CS terhubung ke pin CSpin, modul ADC dapat dipilih oleh program
RD (J7)	1-2	RD terhubung ke Ground, hanya dapat menggunakan Write-Read Mode atau WR-RD Stand Alone Operation
	2-3	RD terhubung ke pin RDpin, semua mode dapat digunakan

Prosedur Testing

Dengan DT-51™ Low Cost Micro System:
1. Hubungkan ADC-08 dengan DT-51™ Low Cost Micro System secara I/O.
2. Pindah jumper RD dan MODE ke posisi 2-3.
3. Pindah jumper CS ke posisi 1-2.
4. Beri tegangan supply +9 VDC pada DT-51™ Low Cost Series.
5. Programlah ADIOWRRD.HEX ke dalam DT-51™ Low Cost Series.
6. Jalankan LCADDA.EXE, pilih COM port yang digunakan, dan pilih mode ADC Write Read Mode.
7. Tekan "Update" untuk membaca hasil konversi input ADC secara berulang-ulang.
8. Tekan "Stop" untuk menghentikan proses pembacaan.

Dengan DT-51™ Minimum System ver 3.0:
1. Hubungkan ADC-08 dengan DT-51™ Minimum System ver 3.0 secara System Bus.
2. Pindah jumper RD, CS, dan MODE ke posisi 2-3.
3. Beri tegangan supply +9 VAC pada DT-51™ Minimum System ver 3.0.
4. Download-lah ADSB.HEX ke dalam DT-51™ Minimum System ver 3.0.
5. Jalankan LCADDA.EXE, pilih COM port yang digunakan, dan pilih mode ADC Write Read Mode.
6. Tekan "Update" untuk membaca hasil konversi input ADC secara berulang-ulang.
7. Tekan "Stop" untuk menghentikan proses pembacaan.

Rutin, prosedur testing, dan kerangka program yang lain dapat Anda pelajari pada MANUAL ADC-08.PDF.



PC-Link

SERIAL PPI

PC-Link					
Serial (COM) Port	*Parallel (LPT) Port*	*USB*	*Firewire*	*ISA slot*	*PCI slot*
✓					

Quick Start

1. PENDAHULUAN

PC-Link SERIAL PPI merupakan pengendali 40 bit jalur input/output melalui antarmuka UART RS-232 yang dapat dihubungkan ke komputer secara langsung. Contoh aplikasi dari SERIAL PPI adalah sebagai pengendali tampilan LED, sebagai pembaca kondisi saklar, penghitung pulsa counter, dan lain-lain.

Untuk manual dan source-source yang lebih lengkap dapat dilihat di dalam CD.

2. SPESIFIKASI EKSTERNAL SERIAL PPI

Spesifikasi Eksternal SERIAL PPI adalah sebagai berikut:

- Menggunakan antarmuka UART RS-232.
- 4 pilihan Baud Rate.
- 16 bit jalur Input/Output (Port 1 dan Port 2) dengan level CMOS.
- 24 bit jalur Programmable Peripheral Interface 82C55 (Port A, Port B, dan Port C) dengan level CMOS.
- 2 Counter 16 bit (Counter 0 dan Counter 1) dengan level CMOS.
- Sumber tegangan input 12 VDC.
- Tersedia Voltage Regulator dengan tegangan output 5 VDC.

3. SISTEM YANG DIANJURKAN

Perangkat Keras:

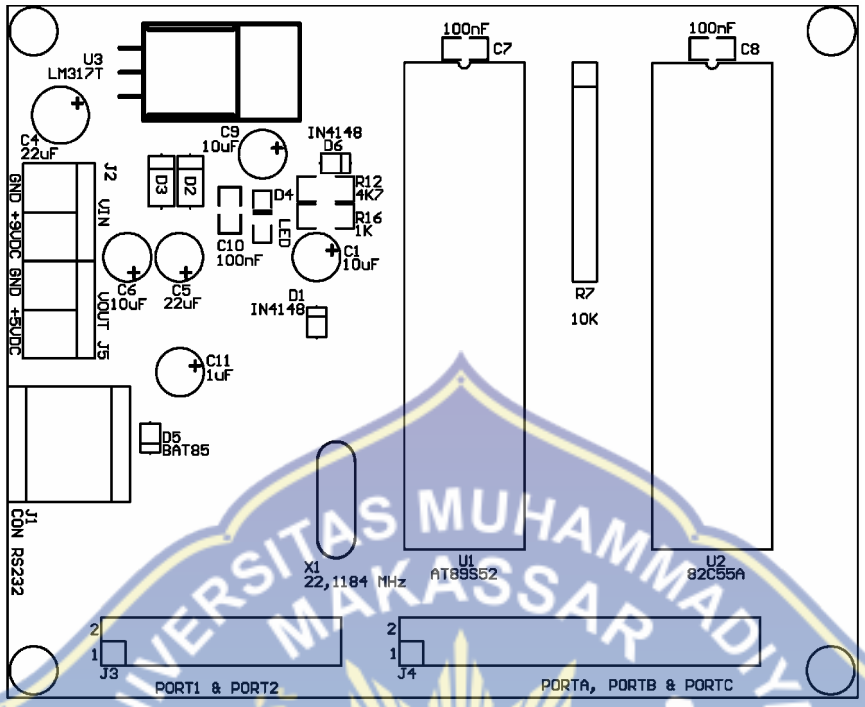
- PC AT™ / Pentium® IBM Compatible dengan Serial Port (COM1 / COM2).
- CD-ROM Drive.
- Ruang hard disk minimum 2 Mbytes.

Perangkat Lunak:

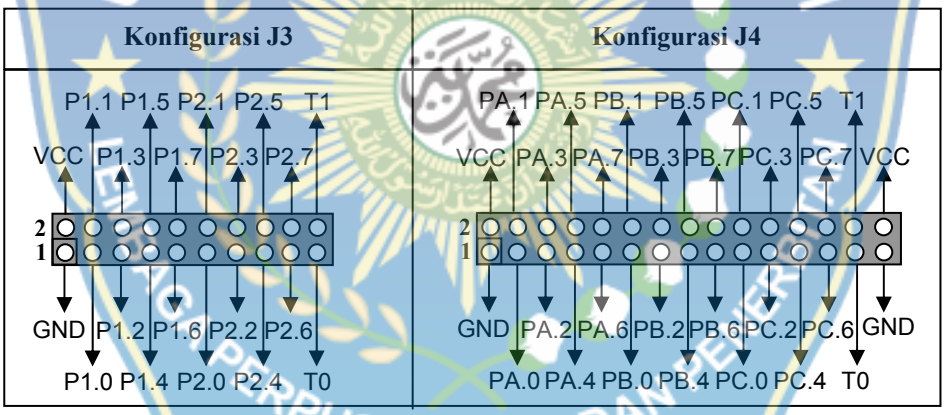
- Sistem Operasi MS-DOS®, PC-DOS™, atau Windows® 9x ke atas.
- Borland© Delphi 5.0 atau Turbo© Pascal 7.0.
- File-file dalam CD:
  SERPPI.EXE, PASPPI.EXE, SERLIB.DLL, SERPAS.TPU, MANUAL SERIAL PPI.PDF, QUICK START SERIAL PPI.PDF, 89S52.PDF, 82C55.PDF

4. PERANGKAT KERAS SERIAL PPI

4.1 TATA LETAK KOMPONEN SERIAL PPI



4.2 ALOKASI DAN SPESIFIKASI PORT



Spesifikasi untuk Port 1, 2, Counter 0, dan Counter 1 adalah sebagai berikut:

Simbol	Parameter	Nilai	Satuan
I_{OL}	Arus saat output berlogika ‘0’	1,6	mA
I_{OH}	Arus saat output berlogika ‘1’	-10	µA

I_{OL} maksimum per pin adalah 10 mA.
I_{OL} maksimum per port adalah 15 mA.
I_{OL} maksimum untuk semua port adalah 71 mA.

Spesifikasi untuk Port A, B, dan C adalah sebagai berikut:

Simbol	Parameter	Nilai	Satuan
I_{OL}	Arus saat output berlogika ‘0’	2,5	mA
I_{OH}	Arus saat output berlogika ‘1’	-100	µA

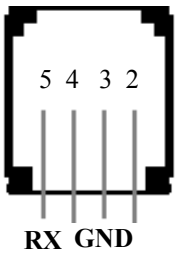
Keterangan lebih lanjut dapat dilihat pada datasheet IC yang bersangkutan.

4.3 HUBUNGAN SERIAL PPI DENGAN KOMPUTER

SERIAL PPI dapat dihubungkan dengan COM port komputer atau dengan kontroler lain yang juga memiliki interface UART RS-232. Perhatikan hubungan jalur komunikasinya.

COM Port Komputer DB9 Male	SERIAL PPI RJ11 Female
RTS (Pin 7)	RTS (Pin 2)
GND (Pin 5)	GND (Pin 3)
TX (Pin 3)	TX (Pin 4)
RX (Pin 2)	RX (Pin 5)

J1 Tampak Depan



RX GND
TX RTS

4.4 MENCOBA SERIAL PPI DENGAN SERPPI.EXE

- Hubungkan kabel serial antara SERIAL PPI dan komputer.
- Hubungkan perangkat output (misalnya rangkaian LED) ke port SERIAL PPI.
- Hubungkan sumber tegangan. Hubungkan juga referensi Ground antara rangkaian tambahan dengan Ground pada SERIAL PPI.
- Jalankan program SERPPI.EXE (under Windows®), pilih COM yang digunakan dan tekan tombol **Start**. LED yang dihubungkan pada port SERIAL PPI akan menyala bergantian.

4.5 MENCOBA SERIAL PPI DENGAN PASPPI.EXE

- Hubungkan kabel serial antara SERIAL PPI dan komputer.
- Hubungkan perangkat output (misalnya rangkaian LED) ke port SERIAL PPI.
- Hubungkan sumber tegangan. Hubungkan juga referensi Ground antara rangkaian tambahan dengan Ground pada SERIAL PPI.
- Jalankan program PASPPI.EXE (under DOS). Ketik COM port yang digunakan dan tekan Enter. LED yang dihubungkan pada port SERIAL PPI akan menyala bergantian.

5. PERANGKAT LUNAK SERIAL PPI

Waktu yang dibutuhkan SERIAL PPI mulai menyala hingga siap dioperasikan (Start-up Time) = 25 ms.

5.1 SPESIFIKASI UART RS-232

Secara default, komunikasi UART RS-232 bekerja pada **Baud Rate 9600 bps, 8 Data Bit, No Parity Bit, 1 Stop Bit, No Flow Control**. Pilihan baud rate yang lain terdapat pada **bagian 5.6.**

5.2 COMMAND

Semua transaksi data selalu dimulai dengan mengirimkan Command ke SERIAL PPI.

COMMAND

M2	M1	M0	X	X	X	X	X

Mode

M2	M1	M0	MODE
0	0	0	*Tidak Terpakai*
0	0	1	Byte Transfer
0	1	0	Bit Set/Reset
0	1	1	Counter
1	0	0	Baud Rate
1	1	X	*Tidak Terpakai*

5.3 BYTE TRANSFER

COMMAND

0	0	1	P2	P1	A1	A0	R/$\overline{W}$

Mode

Command untuk Byte Transfer akan mengakses salah satu dari Port 1, Port 2, Port A, Port B, Port C, atau Control Word. Mengakses lebih dari satu port pada saat yang sama tidak diperkenankan.
Proses tulis/output dan proses baca/input dibedakan oleh R/$\overline{W}$. R/$\overline{W}$ diberi nilai 1 untuk proses pembacaan dari port dan diberi nilai 0 untuk proses penulisan ke port.
Proses baca akan diikuti oleh satu byte hasil pembacaan port yang dikirim oleh modul SERIAL PPI. Proses tulis harus diikuti oleh satu byte yang akan dikirimkan ke port.

Mode	P2	P1	A1	A0	R/$\overline{W}$	Proses
001	0	0	0	0	0	Tulis ke Port A
	0	0	0	0	1	Baca dari Port A
	0	0	0	1	0	Tulis ke Port B
	0	0	0	1	1	Baca dari Port B
	0	0	1	0	0	Tulis ke Port C
	0	0	1	0	1	Baca dari Port C
	0	0	1	1	0	Tulis ke Control Word
	0	0	1	1	1	*Tidak Terpakai*
	0	1	0	0	0	Tulis ke Port 1
	0	1	0	0	1	Baca dari Port 1
	0	1	0	1	X	*Tidak Terpakai*
	0	1	1	X	X	*Tidak Terpakai*
	1	0	0	0	0	Tulis ke Port 2
	1	0	0	0	1	Baca dari Port 2
	1	0	0	1	X	*Tidak Terpakai*
	1	0	1	X	X	*Tidak Terpakai*
	1	1	X	X	X	*Tidak Terpakai*

Secara default, kondisi semua port pada saat awal adalah:
- Port 1 dan Port 2 : sebagai input, bernilai FFH
- Port A, Port B, Port C : Tidak terprogram sebagai input maupun output

(Keterangan lebih lanjut mengenai Port A, Port B, dan Port C terdapat dalam Manual SERIAL PPI atau datasheet 82C55).

5.4 BIT SET/RESET

COMMAND

0	1	0	P2/$\overline{P1}$	D2	D1	D0	V

Mode

Command untuk Bit Set/Reset akan menulis/output salah satu bit dari Port 1 atau Port 2.

Pemilihan bit ditentukan oleh nilai D0, D1, dan D2. Sedangkan nilai yang dituliskan ke bit ditentukan oleh V.

D2	D1	D0	Bit yang Diakses
0	0	0	Bit 0
0	0	1	Bit 1
0	1	0	Bit 2
0	1	1	Bit 3
1	0	0	Bit 4
1	0	1	Bit 5
1	1	0	Bit 6
1	1	1	Bit 7

Mode	P2/$\overline{P1}$	D2	D1	D0	V	Proses
010	0	X	X	X	0	Clear/Reset (memberi logika 0) pada Bit dari Port 1
	0	X	X	X	1	Set (memberi logika 1) pada Bit dari Port 1
	1	X	X	X	0	Clear/Reset (memberi logika 0) pada Bit dari Port 2
	1	X	X	X	1	Set (memberi logika 1) pada Bit dari Port 2

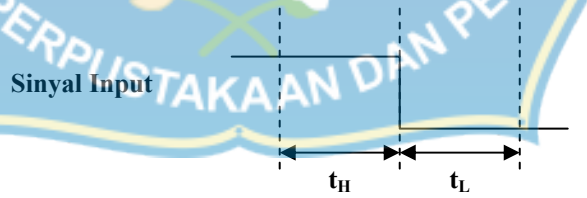
5.5 COUNTER

COMMAND

0	1	1	0	0	C1/$\overline{C0}$	GET/ $\overline{INIT}$	START/ $\overline{STOP}$

Mode

Command untuk Counter akan mengakses salah satu dari Counter 0 atau Counter 1.
Start dan **Stop** berfungsi untuk mengaktifkan atau menghentikan counter dan hanya bekerja apabila **Init** diberi nilai 0.
Register Counter 0 dan Counter 1 akan bertambah jika pin yang bersangkutan (T0 atau T1) mendapatkan sinyal sesuai timing diagram berikut.



Simbol	Keterangan	Nilai	Satuan
t_H	Durasi sinyal berlogika '1' sebelum transisi	0,55	µs
t_L	Durasi sinyal berlogika '0' setelah transisi	0,55	µs

Pembacaan dengan Get akan diikuti oleh dua byte hasil pembacaan nilai counter yang dikirim oleh modul SERIAL PPI. Data pertama adalah nilai Least Significant Byte, sedangkan data kedua adalah nilai Most Significant Byte.

Mode	00	C1/$\overline{C0}$	GET/$\overline{INIT}$	START/$\overline{STOP}$	Proses
011	00	0	0	0	Menghentikan Counter 0
	00	0	0	1	Mengaktifkan Counter 0
	00	0	1	0	Get Counter 0 Data
	00	0	1	1	*Tidak Terpakai*
	00	1	0	0	Menghentikan Counter 1
	00	1	0	1	Mengaktifkan Counter 1
	00	1	1	0	Get Counter 1 Data
	00	1	1	1	*Tidak Terpakai*
	01	X	X	X	*Tidak Terpakai*
	1X	X	X	X	*Tidak Terpakai*

Secara default, kondisi awal Counter 0 dan Counter 1 adalah tidak aktif.

5.6 BAUD RATE

COMMAND

1	0	0	0	0	D2	D1	D0

Mode

Command untuk Baud Rate akan mengaktifkan salah satu dari beberapa setting Baud Rate. Pemilihan setting baud rate ditentukan oleh nilai D0, D1, dan D2.

Mode	00	D2	D1	D0	Baud Rate
100	00	0	0	0	9600 bps
	00	0	0	1	19200 bps
	00	0	1	0	38400 bps
	00	0	1	1	57600 bps
	00	1	0	0	115200 bps
	01	1	X	1	*Tidak Terpakai*
	1X	X	X	X	*Tidak Terpakai*

Secara default, kondisi baud rate pada saat awal adalah:
- Baud Rate : 9600 bps

5.7 RUTIN DLL DAN TPU

SERIAL PPI memiliki library berupa file SERLIB.DLL yang dapat digunakan oleh bahasa pemrograman yang dapat mengakses file tersebut (misalnya Borland© Delphi, Borland© C++, dll). Selain itu SERIAL PPI juga memiliki unit berupa file SERPAS.TPU yang dapat digunakan oleh bahasa pemrograman Turbo Pascal. Kedua file ini dapat digunakan untuk mempermudah user dalam pemrograman.
Berikut ini adalah rutin-rutin yang digunakan dalam SERLIB.DLL dan SERPAS.TPU:

IOFlag
Fungsi : Memeriksa status komunikasi terakhir.
Tipe : Function

Input : -
Output : IOFlag Tipe : Boolean
Keterangan : -
Metode : Panggil rutin ini untuk memeriksa apakah SERIAL PPI membalas dengan kode Acknowledge (FAH) pada komunikasi terakhir.
Jika command terakhir dibalas dengan FAH oleh SERIAL PPI, maka IOFlag bernilai = True.
Jika command terakhir tidak dibalas dengan FAH oleh SERIAL PPI, IOFlag bernilai = False.

COMBaud(Rate)

Fungsi : Mengubah baud rate pada COM port komputer.
Tipe : Function
Input : Rate Tipe : Longint
Output : COMBaud Tipe : Boolean
Keterangan :

- ❖ Nilai Rate yang valid = 9600, 19200, 38400, 57600, dan 115200.
- ❖ Pemanggilan rutin ini hanya mengubah baud rate komputer, baud rate SERIAL PPI tidak berubah.

Metode : Isikan nilai Rate lalu panggil rutin ini seperti contoh.
Jika nilai Rate valid, maka baud rate akan berubah dan COMBaud bernilai = True.
Jika nilai Rate tidak valid, maka baud rate tidak akan berubah dan COMBaud bernilai = False.
Contoh : COMBaud(19200) akan mengubah baud rate komputer menjadi 19200 bps.

DeviceBaud(Rate)

Fungsi : Mengubah baud rate SERIAL PPI dan COM port komputer.
Tipe : Function
Input : Rate Tipe : Longint
Output : DeviceBaud Tipe : Boolean
Keterangan :

- ❖ Nilai Rate yang valid = 9600, 19200, 38400, 57600, dan 115200.
- ❖ Pemanggilan rutin ini masih menggunakan baud rate sebelumnya.
- ❖ Pemanggilan rutin ini akan mengubah baud rate komputer dan SERIAL PPI.

Metode : Isikan nilai Rate lalu panggil rutin ini seperti contoh.
Jika nilai Rate valid dan SERIAL PPI dapat menjawab komunikasi, maka baud rate akan berubah dan DeviceBaud bernilai = True.
Jika nilai Rate tidak valid atau SERIAL PPI tidak dapat menjawab komunikasi, maka baud rate tidak akan berubah dan DeviceBaud bernilai = False.
Contoh : DeviceBaud(19200) akan mengubah baud rate komputer dan SERIAL PPI menjadi 19200 bps.

CountStatus(CounterSelect, Run)

Fungsi : Mengaktifkan atau menghentikan Counter.
Tipe : Function
Input : CounterSelect Tipe : Byte
Run Tipe : Boolean
Output : CountStatus Tipe : Boolean

Keterangan :

- ❖ Nilai CounterSelect yang valid = 0 dan 1.

Metode : Pilih Counter yang akan diaktifkan dengan mengisi CounterSelect.
Jika Run diberi nilai True, maka Counter yang ditentukan akan aktif dan input Counter tersebut akan terbaca oleh SERIAL PPI.
Jika Run diberi nilai False, maka Counter yang ditentukan akan dihentikan dan input Counter tersebut tidak akan terbaca.

Jika nilai CounterSelect valid dan SERIAL PPI dapat menjawab komunikasi, maka kondisi Counter akan berubah dan CountStatus akan bernilai = True.
Jika nilai CounterSelect tidak valid atau SERIAL PPI tidak dapat menjawab komunikasi, maka kondisi Counter tidak akan berubah dan CountStatus akan bernilai = False.

Contoh : CountStatus(0,True) akan menjalankan Counter 0.

CountRead(CounterSelect)

Fungsi : Membaca nilai Counter.
Tipe : Function
Input : CounterSelect Tipe : Byte
Output : CountRead Tipe : Word
Keterangan :

- ❖ Nilai CounterSelect yang valid adalah 0 dan 1.

Metode : Pilih Counter yang akan diaktifkan dengan mengisi CounterSelect.
Jika nilai CounterSelect valid dan SERIAL PPI dapat menjawab komunikasi, maka nilai Counter yang dipilih akan berada pada CountRead dan register internal SERIAL PPI akan kembali ke nilai 0.
Jika nilai CounterSelect tidak valid atau SERIAL PPI tidak dapat menjawab komunikasi, maka CountRead akan bernilai = 0 dan register internal SERIAL PPI tidak akan kembali ke nilai 0.
Contoh : CountRead(1) akan membaca nilai Counter 1.

PortWrite(IOport, DataOut)

Fungsi : Menuliskan data ke Port.
Tipe : Function
Input : IOport Tipe : String
DataOut Tipe : Byte
Output : PortWrite Tipe : Boolean
Keterangan :

- ❖ Nilai IOport yang valid = '1', '2', 'CW' atau 'cw', 'A' atau 'a', 'B' atau 'b', dan 'C' atau 'c'.
- ❖ Penulisan ke CW(Control Word) dapat berfungsi untuk menentukan mode Port A, B, dan C atau untuk menjalankan Bit Set/Reset untuk Port C.
- ❖ Penulisan ke Port A, B, atau C dapat dilakukan jika port dikonfigurasi sebagai output.

Metode : Pilih Port yang akan diakses dengan mengisi IOport.
Nilai yang akan dikirimkan ke Port diisikan ke DataOut.
Jika nilai IOport valid dan SERIAL PPI dapat menjawab komunikasi, maka nilai DataOut akan dikirimkan ke Port dan PortWrite bernilai = True.
Jika nilai IOport tidak valid atau SERIAL PPI tidak dapat menjawab komunikasi, maka nilai DataOut tidak akan dikirimkan ke Port dan PortWrite bernilai = False.
Contoh : PortWrite('1',20) akan menuliskan nilai 20 ke Port 1.

PortRead(IOport)

Fungsi : Membaca data dari Port.
Tipe : Function
Input : IOport Tipe : String
Output : PortRead Tipe : Byte
Keterangan :

- ❖ Nilai IOport yang valid = '1', '2', 'A' atau 'a', 'B' atau 'b', dan 'C' atau 'c'.
- ❖ Pada pembacaan Port 1 dan 2, semua bit Port akan diberi logika = 1 sebelum Port dibaca.
- ❖ Pembacaan dari Port A, B, atau C dapat dilakukan jika port dikonfigurasi sebagai input.

Metode : Pilih Port yang akan diakses dengan mengisi IOport.

Jika nilai IOport valid dan SERIAL PPI dapat menjawab komunikasi, maka hasil pembacaan Port akan berada pada PortRead.
Jika nilai IOport tidak valid atau SERIAL PPI tidak dapat menjawab komunikasi, maka PortRead akan bernilai = 0.

Contoh : PortRead('2') akan membaca nilai Port 2.

BitWrite(IOport, BitSelect, SetReset)

Fungsi : Memberi logika 0 atau 1 ke bit Port 1 atau Port 2.
Tipe : Function
Input : IOport Tipe : Byte
BitSelect Tipe : Byte
SetReset Tipe : Boolean
Output : BitWrite Tipe : Boolean
Keterangan :

- Nilai IOport yang valid = 1 dan 2.
- Nilai BitSelect yang valid = 0, 1, 2, 3, 4, 5, 6, dan 7.

Metode : Pilih Port yang akan diakses dengan mengisi IOport.
Pilih Bit yang diakses dengan mengisi BitSelect.
Jika SetReset bernilai = True, maka Bit akan diberi logika = 1.
Jika SetReset bernilai = False, maka Bit akan diberi logika = 0.
Jika nilai IOport dan BitSelect valid dan SERIAL PPI dapat menjawab komunikasi, maka BitWrite akan bernilai = True.
Jika nilai IOport atau BitSelect tidak valid atau SERIAL PPI tidak dapat menjawab komunikasi, maka BitWrite akan bernilai = False.
Contoh : BitWrite(1,5,True) akan memberi logika 1 pada P1.5.

COMSetPort(PortSer)

Fungsi : Memilih COM port.
Tipe : Function
Input : PortSer Tipe : String
Output : COMSetPort Tipe : Boolean
Keterangan :

- Nilai PortSer yang valid = 'COMn' dimana n adalah nomor COM port yang digunakan.
- Nilai PortSer umumnya hanya berkisar antara 'COM1' dan 'COM2', namun ada komputer yang memiliki 'COM3' atau lebih.
- Rutin ini hanya terdapat pada SERLIB.DLL.

Metode : Gunakan rutin ini untuk memilih COM port sebelum membukanya.
Jika rutin ini tidak digunakan sebelum membuka COM port, maka COM port yang digunakan adalah COM1.
Jika nilai PortSer valid, maka COMSetPort bernilai = True.
Jika nilai PortSer tidak valid, maka COMSetPort bernilai = False.
Contoh : COMSetPort('COM2') akan membuka COM2.

Penting!
Jika COM port sedang terbuka lalu diubah menggunakan rutin COMSetPort, maka COM port sebelumnya akan ditutup secara otomatis dan port yang baru akan dibuka secara otomatis.
Setting Baud Rate komputer tidak akan berubah meskipun COM port diubah dengan COMSetPort.

COMOpen(Stat)

Fungsi : Membuka atau menutup COM port.
Tipe : Function
Input : Stat Tipe : Boolean
Output : COMOpen Tipe : Boolean
Keterangan :

- Rutin ini hanya terdapat pada SERLIB.DLL.

Metode : Jika COMSetPort tidak digunakan sebelum rutin ini, maka COM port yang diakses adalah COM1.
Jika Stat bernilai = True, maka COM port akan terbuka.
Jika Stat bernilai = False, maka COM port akan ditutup.

Jika proses membuka COM port dapat dilakukan, COMOpen akan bernilai = True.
Jika proses membuka/menutup COM port tidak dapat dilakukan atau COM port sudah tertutup, COMOpen akan bernilai = False.

Contoh : COMOpen(True) akan membuka COM port.

Penting!
Sebelum COM port terbuka atau setelah COM port ditutup, komunikasi dengan SERIAL PPI tidak dimungkinkan.
Jika COM port sedang terbuka lalu dipindah menggunakan rutin COMSetPort, maka COM port sebelumnya akan ditutup secara otomatis dan port yang baru akan dibuka secara otomatis.

COMSet(PortSer)

Fungsi : Memilih COM port.
Tipe : Procedure
Input : PortSer Tipe : String
Output : -
Keterangan :
- ❖ Nilai PortSer yang valid = 'COM1' dan 'COM2'.
- ❖ Rutin ini hanya terdapat pada SERPAS.TPU.

Metode : Pilih COM port dengan mengisi PortSer.
Rutin ini hanya akan melakukan pemindahan COM port yang digunakan.
Contoh : COMInit('COM1') akan memindah jalur komunikasi ke COM 1.

COMInit(PortSer)

Fungsi : Memilih COM port dan melakukan inisialisasi.
Tipe : Procedure
Input : PortSer Tipe : String
Output : -
Keterangan :
- ❖ Nilai PortSer yang valid = 'COM1' dan 'COM2'.
- ❖ Rutin ini hanya terdapat pada SERPAS.TPU.

Metode : Pilih COM port dengan mengisi PortSer.
Rutin ini akan melakukan inisialisasi terhadap COM port yang dipilih dan mengatur baud rate komputer menjadi 9600 bps.
Contoh : COMInit('COM1') akan memilih COM 1 dengan baud rate 9600 bps.

Penting!
Setelah COMInit digunakan, baud rate komputer menjadi 9600 bps, namun baud rate SERIAL PPI tidak akan berubah.

DeviceReset

Fungsi : Melakukan reset pada SERIAL PPI.
Tipe : Procedure
Input : -
Output : -
Keterangan :-
Metode : Rutin ini akan mereset SERIAL PPI agar sistem kembali seperti saat baru pertama kali dinyalakan (default).
Contoh : Pemanggilan DeviceReset akan mengembalikan baud rate komputer dan SERIAL PPI menjadi 9600 bps, Port 1 dan Port 2 sebagai input, Port A, Port B, dan Port C tidak terprogram, serta kedua Counter tidak aktif.

5.8 CONTOH APLIKASI DAN PROGRAM

Berikut ini adalah cuplikan program dalam Borland© Delphi 5.0 untuk mengakses Port 1 dan Counter 0.

```
//Tulis ke Port 1
procedure TForm1.SendP1Click(Sender: TObject);
begin
      if not PortWrite('1',dataout) then showmessage('Error');
```

```
      if IOFlag then edit1.text:='Success' else edit1.text:='Error';
end;

//Baca Port 1
procedure TForm1.ReceiveP1Click(Sender: TObject);
begin
      r1.text:=inttohex(PortRead('1'),2);
      if IOFlag then edit1.text:='Success' else edit1.text:='Error';
end;

//Aktifkan Counter 0
procedure TForm1.StartC0Click(Sender: TObject);
begin
      if not CountStatus(0,True) then showmessage('Error');
      if IOFlag then edit1.text:='Success' else edit1.text:='Error';
end;

//Baca register Counter 0
procedure TForm1.GetC0Click(Sender: TObject);
begin
      g0.text:=inttohex(CountRead(0),4);
      if IOFlag then edit1.text:='Success' else edit1.text:='Error';
end;

//Hentikan Counter 0
procedure TForm1.StopC0Click(Sender: TObject);
begin
      if not CountStatus(0,False) then showmessage('Error');
      if IOFlag then edit1.text:='Success' else edit1.text:='Error';
end;
```

Berikut ini adalah cuplikan program dalam Turbo© Pascal 7.0 untuk mengakses Port 2.7 per bit, mengubah baud rate, serta melakukan reset.

```
{mengubah bit P2.7 secara toggle}
if p27 then
begin
  if BitWrite(2,nl,false) then
  begin
    p27:=false;
    textbackground(4);
    clrscr;
  end;
end
else
begin
  if BitWrite(2,nl,true) then
  begin
    p27:=true;
    textbackground(2);
    clrscr;
    end;
  end;
end;
{mengubah baud rate SERIAL PPI}
if ym=2 then DeviceBaud(19200)
{mengubah baud rate komputer}
else if ym=3 then ComBaud(19200)
{melakukan reset}
else if ym=4 then DeviceReset;
```

5.9 KERANGKA PROGRAM

Bagi user yang ingin membuat program aplikasi SERIAL PPI dengan menggunakan rutin yang sudah ada, maka file SERLIB.DLL atau SERPAS.TPU harus digunakan/dipanggil.

SERLIB.DLL merupakan library yang akan selalu digunakan untuk setiap aplikasi SERIAL PPI yang menggunakan pemrograman under Windows®.

SERPAS.TPU merupakan unit yang akan selalu digunakan untuk setiap aplikasi SERIAL PPI yang menggunakan pemrograman Pascal (under DOS).
Kerangka pemrograman SERIAL PPI menggunakan Borland© Delphi 5.0 adalah sebagai berikut :

```
unit SERIALPPI;

interface

uses
  Windows, Forms;                                { Isi sesuai kebutuhan }

type
  TForm1 = class(TForm)
        .                          { Deklarasi komponen/prosedur/fungsi user }
        .
        .
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;
        .                                    { Deklarasi variabel program user }
        .
        .
implementation

{$R *.DFM}

        .                                                { Nama rutin SerLib.dll }
stdcall; external 'SerLib.dll';

        .                          { Panggil rutin SerLib.dll sesuai kebutuhan }
        .
        .

        .                                        { Prosedur/fungsi program user }
        .
        .
end.
```

Kerangka pemrograman SERIAL PPI menggunakan Turbo© Pascal 7.0 adalah sebagai berikut:

```
program SERIALPPI;

uses
  Dos, Crt, Serpas;                              { Isi sesuai kebutuhan }

const
        .                                  { Deklarasi konstanta program user }
        .
        .

type
        .                          { Deklarasi komponen/prosedur/fungsi user }
        .
        .

var
        .                                    { Deklarasi variabel program user }
        .
        .

        .                                       { Prosedur/fungsi program user }
        .
        .
```

```
        .                   { Panggil rutin SerPas.tpu sesuai kebutuhan }
        .
        .

begin
        .                                           { Program utama user }
        .
        .
end.
```

♦ *Terima Kasih atas kepercayaan Anda menggunakan produk kami, bila ada kesulitan, pertanyaan atau saran mengenai produk ini silahkan menghubungi technical support kami :*

support@innovativeelectronics.com


UNIVERSITAS MUHAMMADIYAH
MAKASSAR
LEMBAGA PERPUSTAKAAN DAN PENERBITAN