

Cara Mudah **MERAKIT** **ROBOT** untuk Pemula

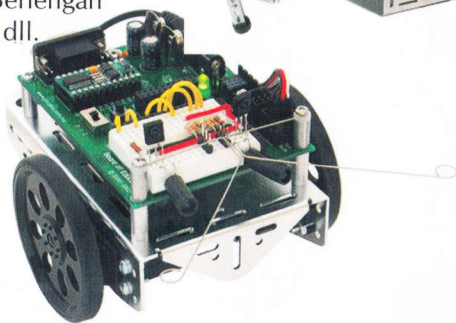
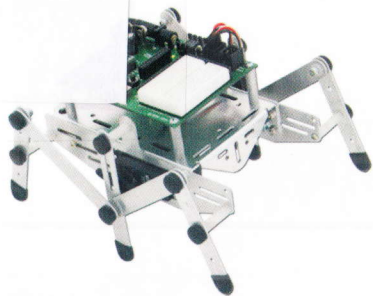
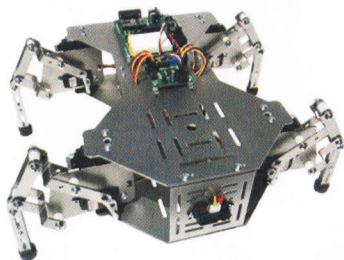


Mengupas Tuntas
Segala Hal tentang Robot

- Komponen-Komponen Dasar Sebuah Robot
- Dasar-Dasar Elektronika Pembuat Robot

USTAKAAN
RSIPAN
WA TIMUR

iat Robot Cerdas dengan Boe-Bot
iat Sendiri Robot *Line Tracker*
iat Robot Berkaki dan Berlengan
iat Robot Cerdas KRCI, dll.



Yusep Nur Jatmika

Cara
Mudah
MERAKIT
ROBOT
untuk Pemula

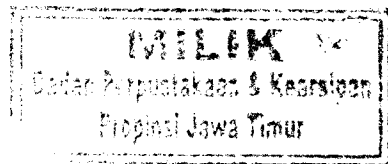
Yusep Nur Jatmika

**Cara
Mudah
MERAKIT
ROBOT
untuk Pemula**

Jaminan Kepuasan

Apabila Anda mendapatkan buku ini dalam keadaan cacat produksi (di luar kesengajaan kami), seperti halaman kosong atau terbalik, silakan ditukar di toko tempat Anda membeli atau langsung kepada kami dan kami akan menggantinya segera dengan buku yang bagus.





412.658/BPEIP/2014

CARA MUDAH MERAKIT ROBOT UNTUK PEMULA

Yusep Nur Jatmika

Editor: Putri Erine Nareswati

Tata Sampul: A. Budi

Tata Isi: Bambang

Pracetak: Wardi

Cetakan Pertama, Juli 2011

Penerbit

FlashBooks

Sampangan Gg. Perkutut No. 325-B

Jl. Wonosari, Baturetno

Banguntapan Jogjakarta

Telp: (0274) 4353776, 7418727

Fax: (0274) 4353776

E-mail: redaksi_divapress@yahoo.com

Blog: www.blogdivapress.com

Website: www.divapress-online.com

Distributor Tunggal

Buku Kita

Jl. Kelapa Hijau No. 22 RT. 006/03

Jagakarsa, Jakarta Selatan 12620

Telp. (021) 7888-1850 (hunting)

Fax. (021) 7888-1860

www.distributorbukukita.com

Sumber gambar cover: www.elexp.com



Pengantar Penulis

Dewasa ini, robot memegang peranan yang semakin penting di dalam kehidupan manusia. Di Indonesia, misalnya, robot dapat ditemukan di banyak lini kehidupan kita, mulai dari urusan menjinakkan bom sampai *assembly line* di pabrik-pabrik otomotif. Sedangkan untuk levael yang tidak terlalu serius, kita bisa menyaksikan menjamurnya perlombaan robotika dengan tema beragam.

Untuk itu, membuat robot dengan biaya murah serta inovatif merupakan hal yang menarik bagi banyak kalangan, khususnya pelajar dan pemula. Dengan alasan itu pula, penulis kemudian meluncurkan buku ini. Buku ini akan sangat tepat jika dibaca oleh kalangan-kalangan penggemar

robot, baik pelajar maupun para pemula. Buku ini membahas secara detail bagaimana proses pembuatan robot cerdas yang andal dan inovatif dengan sangat mudah.

Sehingga, tidak ada cara lain untuk bisa mahir membuat robot selain mencobanya secara langsung. Karena itu, Anda disarankan untuk bisa memiliki buku ini agar mahir berinovatif. Sebagai panduan praktis, buku ini juga dilengkapi dengan berbagai contoh program aplikasi siap jalan yang patut Anda coba.

Penulis juga mengucapkan terima kasih kepada Parallax.com yang telah mengizinkan penulis untuk mengadopsi materi tentang robot ke dalam buku ini. Tentunya, buku ini tidak terlepas dari segala kesalahan dan kekurangan, baik dari segi penyajian maupun penulisan. Saya berharap mudah-mudahan buku ini bisa bermanfaat dan berguna untuk kita semua. *Amin!*

Yusep Nur Jatmika



Daftar Isi

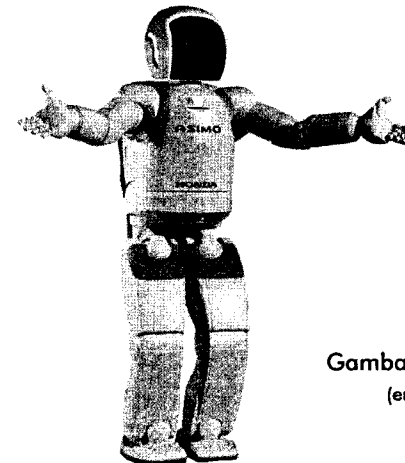
Pengantar Penulis	5
Daftar Isi.....	7
Bab 1 Komponen-Komponen Dasar Sebuah Robot	
Robot	9
A. Sensor Robot	11
B. Sistem Pergerakan Robot.....	20
C. Komponen Energi Sebuah Robot.....	25
D. Microcontroller.....	29
Bab 2 Dasar-Dasar Elektronika Pembuat Robot	
Robot	49
A. Resistor	51
B. Kapasitor (Kondensator).....	58
C. Dioda.....	70
D. Sirkuit Terintegrasi (Integrated Circuit) .	75

E. Bilangan-Bilangan dalam Elektronika Digital.....	83
Bab 3 Membuat Robot Cerdas dengan Boe-Bot.....	93
A. Merakit dan Mencoba Boe-Bot.....	93
B. Robot Whisker.....	103
C. Kecerdasan Buatan dan Pengambilan Keputusan.....	109
D. Navigasi dengan Inframerah.....	115
E. Kendalikan Robot dengan Deteksi Jarak	121
Bab 4 Membuat Sendiri Robot Line Tracker	129
A. Robot Line Tracker	130
B. Lapangan Uji Coba.....	136
Bab 5 Membuat Robot Berkaki dan Berlengan.....	157
A. Robot Berkaki (Crawler Boe-Bot).....	157
B. Robot Berkaki Hex (Quad Crawler).....	164
C. Lengan Robot.....	174
D. Kode Program.....	175
Bab 6 Membuat Robot Cerdas KRCI.....	181
A. Tahap Perencanaan	182
B. Tahap Pembuatan	183
C. Tahap Uji Coba	190
D. Latihan.....	192
Daftar Pustaka.....	195



Bab 1

Komponen-Komponen Dasar Sebuah Robot



Gambar 1. Robot Asimo
(englishclasslog.wordpress.com)

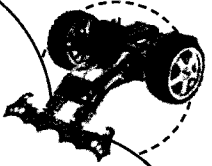
Kata robot pertama kali diperkenalkan oleh seorang penulis dari Czech bernama Karel pada tahun 1921. Kata Robot berasal dari kata *robota*, yang berarti pekerja sendiri. Sejarah robot bermula ketika Jacques de Vaucanson membuat sistem otomatis pada tahun 1938, yaitu dengan membuat bebek mekanik yang dapat memakan, mencincang biji-bijian, serta membuka dan menutupkan sayapnya. Kemudian, pada tahun 1796, Hisashine Tanaga di Jepang

berhasil membuat mainan mekanik yang dapat menghidangkan teh dan menulis huruf kanji. Lalu, pada tahun 1926, Nikola Tesla mendemonstrasikan perahu boot yang dapat dikontrol dengan radio. Pada tahun 1928, Makoto Nishimura membuat robot pertama di Jepang.

Sejalan dengan perkembangan teknologi elektronika, maka perkembangan robot ini melaju pesat, seperti tahun 1948 ketika William Grey Walter membuat robot elektronik otomatis pertama di mana robot ini dapat merespons cahaya dan dapat melakukan kontak dengan objek dari luar. Pada tahun 1954, saat dimulainya zaman digital, sebuah robot digital yang dapat deprogram ditemukan oleh George Devol.

Pada abad modern ini, sudah ada bermacam-macam robot yang dicipta dan digunakan seperti dalam industri, rumah sakit, transportasi, pendidikan, dan kehidupan sehari-hari seperti robot yang digunakan untuk mengecat mobil, merakit komponen elektronik, humanoid robot yaitu robot yang memiliki muka, mampu berjalan, dan bertindak seperti manusia. Oleh karena itu, untuk membuat sebuah robot, minimal dibutuhkan 4 komponen utama yang merupakan dasar dalam pembuatan sebuah robot.

Pertama, sensor. Sensor adalah komponen yang dapat merespons kondisi lingkungan yang diberikan. Sensor ini dapat berupa sensor cahaya, suara, suhu, tekanan. *Kedua*, *actuator* atau penggerak. *Actuator* adalah komponen yang menghasilkan gerak mekanik. *Actuator* ini dapat berupa motor, relay, pneumatik, atau hidrolik *actuator*. *Ketiga*, tenaga atau *power*. Biasanya, komponen energi penggerak di dalam robot adalah berupa baterai, sinar matahari, atau catur daya. *Keempat*, *microcontroller*. Komponen ini digunakan sebagai pusat pemikir untuk memproses data dari sensor dan memerintahkan *actuator* untuk bertindak.



Sensor adalah komponen yang dapat merespons kondisi lingkungan yang diberikan. Sensor ini dapat berupa sensor cahaya, suara, suhu, tekanan.

A. Sensor Robot

Dalam teknologi robotika banyak sekali sensor yang digunakan. Pemakaian sensor dalam sebuah robot sangat tergantung pada fungsi robot itu sendiri. Dari sudut pandang robot, sensor dapat diklasifikasikan dalam dua kategori, yaitu sensor lokal (*on-board*) yang dipasang di tubuh robot dan sensor global yang dipasang di luar robot, tetapi

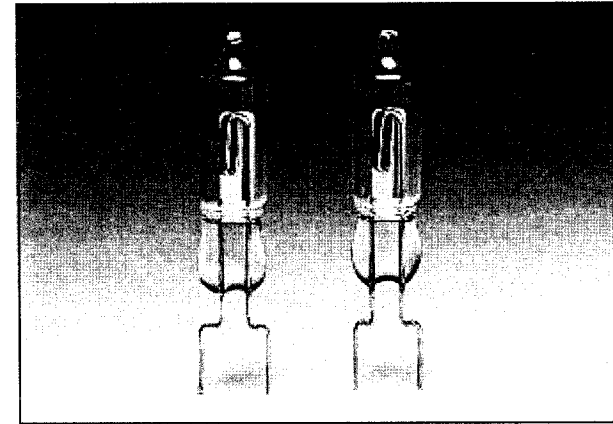
masih dalam lingkungannya. Data dari sensor global ini dikirim balik kepada robot melalui komunikasi nirkabel. Klasifikasi sensor berdasarkan output dan aplikasinya dapat dibedakan menjadi lima.

1. Sensor-Sensor Keperluan Khusus

Sensor jenis ini merupakan sensor yang digunakan secara spesifik untuk robot-robot dengan tujuan tertentu. Contohnya, sensor api untuk robot yang difungsikan untuk memadamkan api, sensor medan magnet pada kompas digital untuk menentukan arah robot, sensor PIR untuk mendeteksi gerakan manusia, dan lain sebagainya.

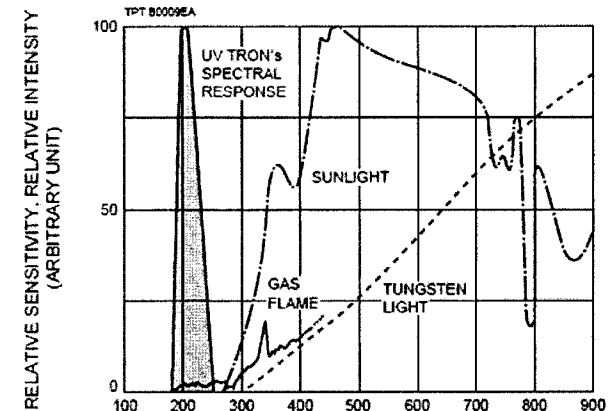
2. Sensor Api (Hamamatsu UVTRON)

Hamamatsu UVTRON R2868 adalah sebuah sensor yang mendeteksi adanya nyala api yang memancarkan sinar ultraviolet. Pancaran cahaya ultraviolet dari sebuah nyala lilin berjarak 5 meter dapat dideteksi oleh sensor ini. Sensor ini juga dapat mendeteksi beberapa fenomena yang tak tampak, seperti transmisi tegangan tinggi.



Gambar 2. UVTRON R2868

(nugi-imagination.co.cc)



Gambar 3. Grafik respons UVTRON

(green-elektronik.blogspot.com)

Gambar 3 menunjukkan respon UVTRON dibandingkan dengan cahaya matahari, nyala api gas,

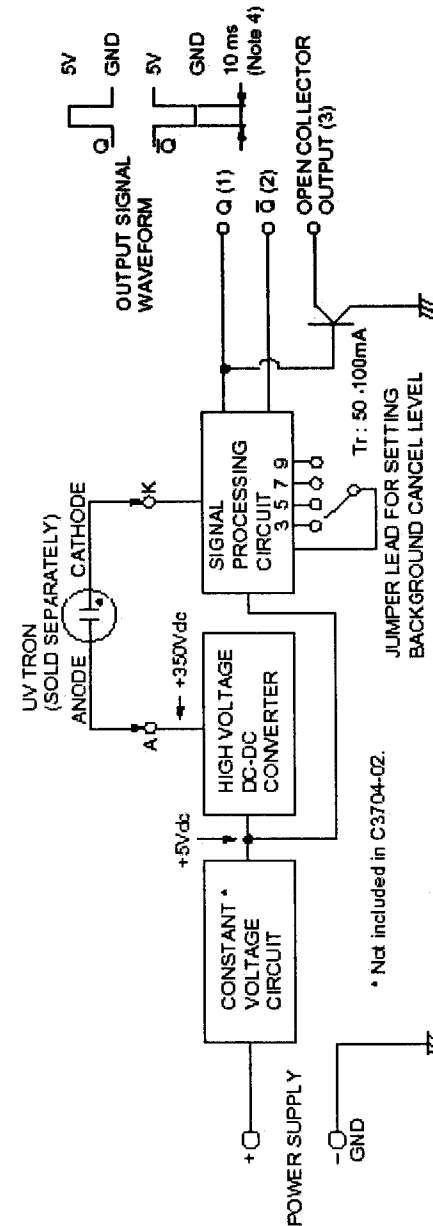
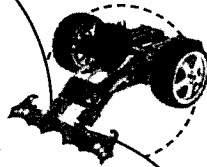
maupun cahaya Tungsten. Agar sensor UVTRON ini dapat terhubung pada sistem *microcontroller*, maka diperlukan rangkaian pengondisi sinyal yang berfungsi mengubah respons dari UVTRON menjadi pulsa yang dapat dikenali oleh sistem *microcontroller*. Dengan Modul C3704, maka respons UVTRON akan diproses menjadi pulsa-pulsa selebar 10 mS dan arus maksimum 100 mA. Keluaran modul ini menggunakan konfigurasi *open collector*.

Pada modul ini, *power supply* 5 volt diubah menjadi 350 volt DC melalui bagian *High Voltage DC to DC Converter* untuk mengaktifkan sensor UVTRON. Sedangkan *Signal Processing Circuit* berfungsi untuk mengatur berapa jumlah pulsa yang masuk dari sensor UVTRON selama 2

detik, yang akan direspons oleh C3704 menjadi pulsa selebar 10 mS. Pada kondisi standar, digunakan *setting* 3 pulsa dalam 2 detik.

Namun, untuk kondisi di mana banyak cahaya liar lainnya, *setting* dapat diubah menjadi 5, 7, atau 9 pulsa,

Agar sensor UVTRON dapat terhubung pada sistem *microcontroller*, maka diperlukan rangkaian pengondisi sinyal yang berfungsi mengubah respons dari UVTRON menjadi pulsa yang dapat dikenali oleh sistem *microcontroller*.



Gambar 4. Rangkaian C3704
(nugi-imagination.co.cc)

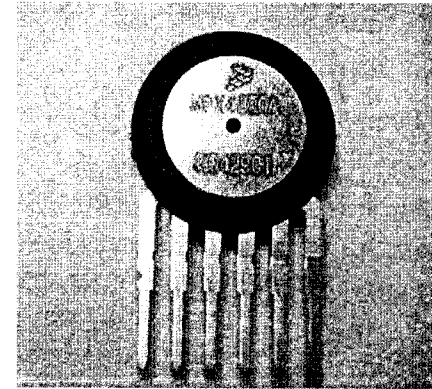
sehingga sensitivitas dari C3704 menjadi lebih rendah.

Keluaran dengan pulsa sebesar 10 mS ini selanjutnya dapat dihubungkan langsung pada sistem *microcontroller*, seperti DST-51, DST-52, ataupun DST-R8C, di mana program pada sistem *microcontroller* tersebut akan mendeteksi adanya perubahan kondisi *input* dengan periode 10 mS sebagai indikasi adanya nyala api dalam area 5 meter.

3. Sensor Tekanan (MPX4100)

MPX4100 adalah sebuah sensor tekanan yang sudah dilengkapi dengan rangkaian pengondisi sinyal dan temperatur *kalibrator* yang membuat sensor menjadi stabil terhadap perubahan suhu. Untuk akurasi pengukuran sensor ini, digunakan teknik *micromachine*, *thin film metallization*, dan proses *bipolar semiconductor*.

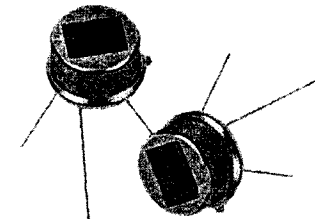
Dengan adanya rangkaian pengondisi sinyal, sensor ini dapat terhubung langsung pada *analog to digital converter*. Rangkaian pengondisi sinyal menghasilkan tegangan analog dengan skala penuh (*full scale*) hingga 5 volt.



Gambar 5. Sensor tekanan MPX4100
(parts.usbid.com)

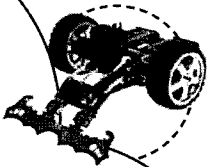
4. Sensor Infrareas Pasif/PIR (RE200B)

PIR atau *Passive Infrared* adalah sebuah sensor yang biasa digunakan untuk mendeteksi keberadaan manusia. Aplikasi ini biasa digunakan sebagai sistem alarm pada rumah-rumah atau perkantoran. Proses kerja sensor ini dilakukan dengan cara mendeteksi adanya radiasi panas tubuh manusia, yang diubah menjadi perubahan tegangan.



Gambar 6. Sensor jenis PIR RE200B
(alibaba.com)

Namun, perubahan tegangan pada PIR sangatlah kecil, yaitu berkisar pada ordo 10 hingga 20 milivolt, atau bahkan lebih kecil lagi. Perubahan tersebut sangat tergantung dari beberapa faktor, yaitu panas tubuh dari manusia yang dideteksi, jarak dengan sensor, maupun suhu lingkungan.



Oleh karena itu, diperlukan rangkaian penguat *noninverting amplifier* terlebih dahulu.

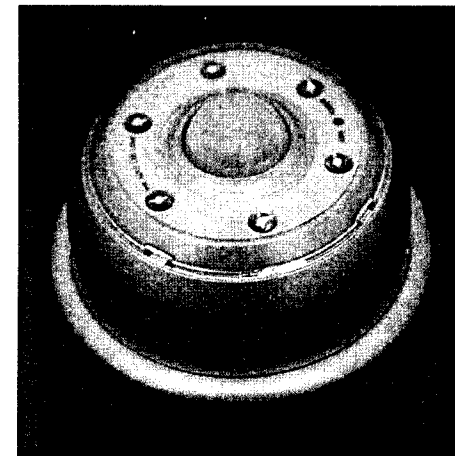
Penguatan yang ada pada *noninverting amplifier* yang ada pada gambar 6 dapat ditentukan dengan rumus berikut:

$$A = \frac{R_f}{R_i} = \frac{1}{10} \text{ M/K} = 100 \times$$

Pengguna dapat mengatur nilai R_f dan R_i dengan memutar variabel resistor 1M dan 10K yang ada pada Modul OP-01 agar ayunan tegangan tersebut dapat diketahui oleh *microcontroller* ataupun rangkaian-rangkaian digital lainnya. Oleh karena itu, diperlukan sebuah *comparator* yang akan mengubah

ayunan tegangan tersebut menjadi kondisi logika, 0 dan 1. Ayunan tegangan ini tidak selalu terjadi pada nilai tegangan yang sama. Artinya, sebagai contoh, untuk ayunan sebesar 0,2 volt dapat terjadi pada tegangan 3,8 volt ke 4 volt atau dapat juga pada 3 volt ke 3,2 volt, dan lain-lain. Hal ini akan mempersulit bagian *comparator* dalam menentukan tegangan pembanding.

5. Rangkaian PIR Detector



Gambar 7. PIR detector jenis JP10
(image.made-in-china.com)

Dengan memasang JP10 pada Modul OP-01, Anda dapat memantau perubahan tegangan *output comparator* melalui 8 bit LED Indicator yang ada

pada modul ini. Pengguna dapat menghubungkan keluaran dari *comparator* ini ke *microcontroller* atau rangkaian-rangkaian digital lainnya.

B. Sistem Pergerakan Robot

Salah satu contoh penggerak robot adalah servo motor. Saya akan banyak membahas pemrograman servo motor dan motor DC, karena kita akan menggunakan motor tersebut sebagai penggerak utama robot/*mobile robot*. *Mobile robot* (robot yang sering bergerak) memerlukan suatu penggerak agar dapat berpindah. Ada beberapa jenis penggerak pada *mobile robot*. *Pertama*, *mobile robot* yang digerakkan dengan roda dan terdapat dua penggerak yang bekerja secara diferensial. Ini adalah model yang umum untuk *mobile robot* dengan biaya yang murah, misalnya menggunakan motor DC atau servo motor. *Kedua*, *mobile robot* yang digerakkan dengan roda dan terdapat satu penggerak serta satu kemudi. *Ketiga*, *mobile robot* yang digerakkan dengan pergerakan seperti kaki.

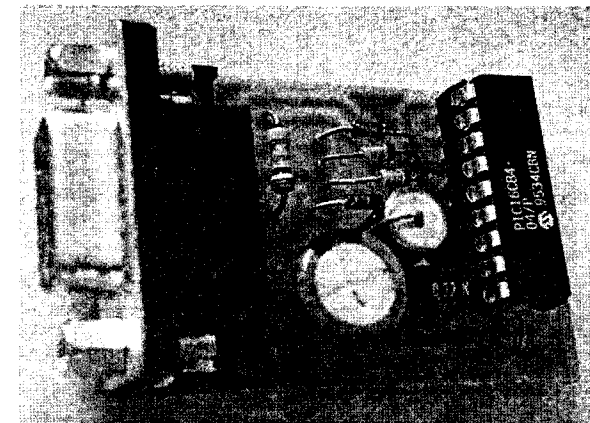
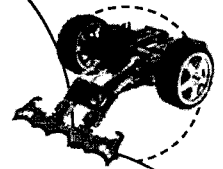
1. DC Motor

DC motor atau dinamo adalah motor yang paling banyak digunakan untuk *mobile robot*. DC motor tidak berisik dan dapat memberikan daya yang mema-

dai untuk tugas-tugas berat. Motor DC standar berputar secara bebas, berbeda dengan *stepper* motor. Untuk mengetahui berapa banyak putaran, biasanya digunakan mekanisme *feedback* menggunakan *shaft encoder*. Gambar berikut menampilkan skema motor DC yang dapat memperoleh arus yang memadai dari penguatan dua buah transistor.

DC motor atau dinamo adalah motor yang paling banyak digunakan untuk *mobile robot*.

DC motor tidak berisik dan dapat memberikan daya yang memadai untuk tugas-tugas berat. Motor DC standar berputar secara bebas, berbeda dengan *stepper* motor.



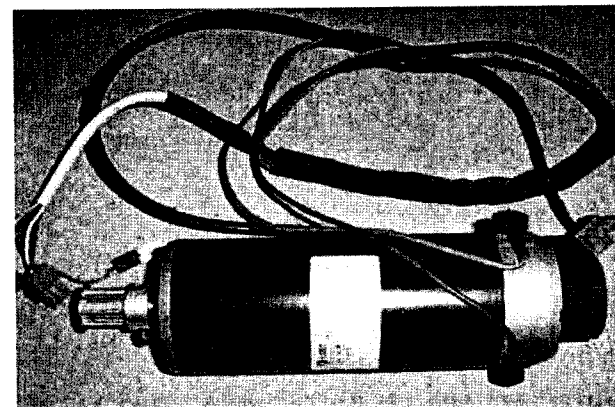
Gambar 8. Penggerak motor DC dengan transistor (sfprime.net)

Sinyal yang kita berikan ke *input* transistor akan mengaktifkan transistor, lalu arus yang memadai dapat menggerakkan motor DC ke arah yang kita inginkan.

2. Servo Motor

Servo motor standar dilengkapi dengan motor DC untuk mengendalikan posisi sebuah robot. Servo motor dapat diputar/diposisikan hingga 180 derajat, sedangkan servo motor continuous dapat berputar hingga 360 derajat. Biasanya, servo motor ini digunakan untuk mengendalikan gerak dari *toys* (mainan), seperti model mobil, pesawat, perahu, dan helikopter.

Karena servo secara luas digunakan untuk hobi-hobi tertentu, tentunya variasinya pun akan sangat banyak dijual. Alat penggerak ini banyak tersedia dan harganya murah. Servo motor adalah DC motor kualitas tinggi yang memenuhi syarat untuk digunakan pada aplikasi servo, seperti *closed control loop*. Motor tersebut harus dapat menangani perubahan yang cepat pada posisi, kecepatan, dan percepatan, serta harus mampu menangani *intermittent torque*.



Gambar 9. Servo motor kualitas tinggi
(takumpu.com)

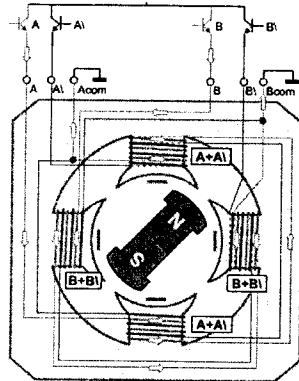
Servo adalah DC motor dengan tambahan elektronika untuk kontrol PW dan digunakan untuk tujuan *hobbyist*, misalnya pada model pesawat terbang, mobil, atau kapal. Servo mempunyai tiga kabel, yaitu Vcc, ground, dan PW input. Tidak seperti PWM pada DC Motor, *input* sinyal untuk servo tidak digunakan untuk mengatur kecepatan, tetapi digunakan untuk mengatur posisi dari putaran servo.

Sinyal PW yang digunakan untuk servo mempunyai frekuensi 50 Hz, sehingga pulsa dibuat setiap 20 ms. Sebagai contoh, sebuah pulsa 0,7 ms akan memutar *disk* servo ke posisi kiri, dan pulsa 1,7 ms akan memutar *disk* ke posisi kanan. Kekurangan servo adalah ia tidak menyediakan *feedback*. Umumnya, kita membeli servo continuous

karena dapat berputar 360 derajat. Namun, anehnya harganya lebih murah dibandingkan servo standar yang derajat putarannya terbatas.

3. Stepper Motor

Stepper motor digunakan untuk daya gerak, pengarah, dan pengendali posisi robot. Motor ini adalah komponen yang terintegrasi sebagai alat kendali komputer untuk industri. *Stepper* motor sangat unik, karena dapat dikendalikan dengan sirkuit digital dan sangat ideal untuk menggerakkan robot secara rotasi (berputar) dan posisi linear/garis lurus. Karena digunakan secara luas pada industri, *stepper* motor memiliki banyak variasi, baik ukuran maupun spesifikasi nya. Motor *stepper* biasanya terdiri atas unipolar dan bipolar.



Gambar 10. Skema *stepper* motor
(blog.ub.ac.id)

Perlu diketahui, tidak semua *stepper* motor memutar rotor dengan nilai yang sama setiap langkahnya. *Stepper* dibuat dengan derajat rotasi yang berbeda-beda per langkahnya. Untuk menghasilkan derajat rotasi yang optimal, sangat bergantung pada penggunaan *stepper* itu sendiri. Putaran *stepper* bergantung pada nilai digital yang diberikan ke *stepper* tersebut.

C. Komponen Energi Sebuah Robot

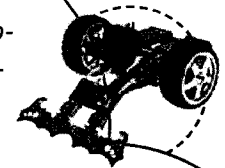
Ada tiga komponen energi dari sebuah robot. Berikut ketiga komponen energi yang dimaksudkan:

1. Tenaga Surya

Tenaga surya menghasilkan tenaga listrik dari cahaya matahari. Jika tenaga surya tersebut cukup kuat, kita dapat mengoperasikan robot secara langsung. Biasanya, robot dengan tenaga surya dirancang sekecil mungkin agar da-

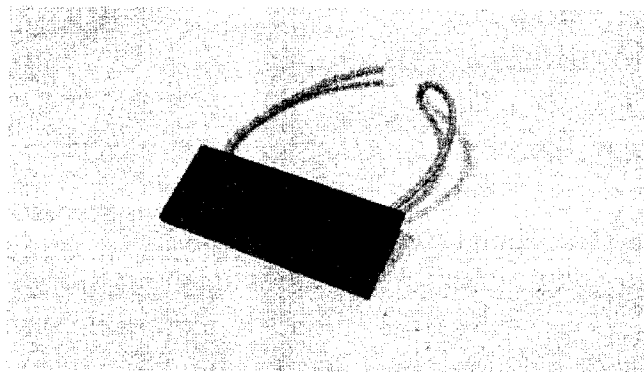
Tidak semua *stepper* motor memutar rotor dengan nilai yang sama setiap langkahnya.

Stepper dibuat dengan derajat rotasi yang berbeda-beda per langkahnya. Untuk menghasilkan derajat rotasi yang optimal, sangat bergantung pada penggunaan *stepper* itu sendiri.



pat berfungsi sesuai kebutuhan. Pengurangan ukuran berat robot akan mengurangi pula tenaga listrik yang dibutuhkan untuk Bergeraknya robot dan daya penggerakannya itu sendiri. Pengurangan berat robot merupakan hal yang sangat penting dalam merancang tenaga dari sebuah robot, baik dengan tenaga baterai dan tenaga *charging*.

Metode lain dalam penggunaan tenaga surya adalah tidak menggerakkan robot secara langsung, tetapi digunakan sebagai sumber tenaga untuk baterai isi ulang (*recharging*). Dengan metode ini, kita dapat mengurangi tenaga surya yang diperlukan untuk menggerakkan robot secara langsung. Namun demikian, tentunya robot yang menggunakan baterai isi ulang hanya bisa berfungsi tergantung dari kekuatan baterai dalam jangka waktu tertentu saja.



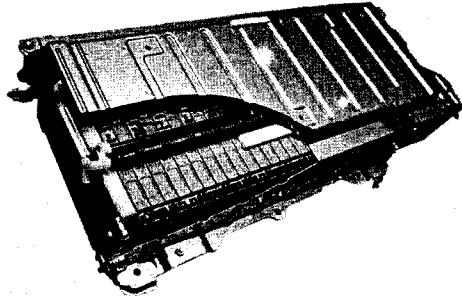
Gambar 11. Sel surya mikro
(browndoggadgets.com)

Metode ketiga adalah kombinasi dari dua metode sebelumnya, yang sekarang sering disebut sebagai *solar engine* (mesin surya). Komponen utama dari *solar engine* adalah tenaga surya itu sendiri, kapasitor, dan sirkuit pemacu. Ketika terkena cahaya, tenaga surya mulai mengisi kapasitor, sehingga kapasitor akan menyediakan tenaga listrik untuk menggerakkan robot. Secara rinci, setelah voltase dari kapasitor bertambah, otomatis sirkuit pemacu (*trigger circuit*) akan aktif. Ketika tenaga pada kapasitor sudah terpenuhi, sirkuit pemacu akan memicu SCR (*Stlicon Controlled Rectifier*) untuk mengeluarkan tenaga yang sudah tersimpan tadi ke kapasitor agar menggerakkan/memfungsikan robot.

Metode *solar engine* ini dapat digunakan untuk berbagai macam rancangan robot. Bahkan, teknologi terkini dapat menggunakan air untuk menghasilkan sumber tenaga robot.

2. Baterai

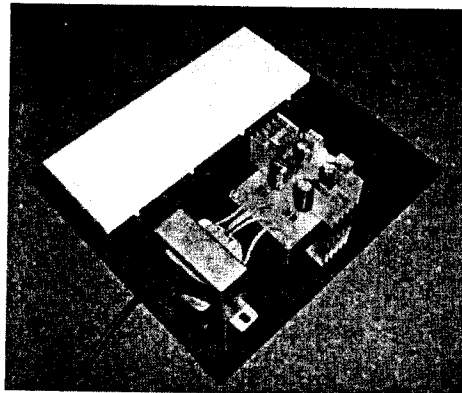
Sejauh ini, baterai adalah tenaga penggerak robot yang paling banyak digunakan, karena sangat mudah. Ada banyak sekali jenis baterai yang digunakan untuk menggerakkan sebuah robot, tetapi beberapa jenis baterai yang sangat umum digunakan adalah carbon-zinc, alkaline, nickel-cadmium, lead-acid, dan lithium.



Gambar 12. Baterai carbon-zinc
(cleangreencar.co.nz)

Untuk lomba robot, penulis menyarankan Anda membeli baterai yang dapat diisi ulang dengan kemampuan di atas 1.100 mAH dan tegangan 7,2 V.

3. Catu Daya



Gambar 13. Catu Daya 12 V
(telinks.files.wordpress.com)

Saat ini, sebagai regulator tegangan telah digunakan IC khusus seperti 7812 untuk regulator tegangan positif 12 V dan 7912 untuk regulator tegangan negatif -12 V. Gambar 13 menampilkan contoh rangkaian catu daya yang mampu mengeluarkan tegangan sangat stabil. CT adalah singkatan dari *Center Tap* yang berfungsi sebagai ground (0 volt).

D. Microcontroller

1. Apakah Microcontroller itu?

Jika kita bicara tentang *microcontroller*, maka tidak akan lepas dari definisi komputer itu sendiri. Mengapa? Sebab, ada beberapa kesamaan antara *microcontroller* dengan komputer atau *microcomputer*.

a. Kesamaan Microcontroller dengan Microcomputer

Beberapa kesamaan antara *microcontroller* dengan *microcomputer* adalah:

- 1) Sama-sama memiliki unit pengolah pusat atau yang lebih dikenal dengan CPU (*Central Processing Unit*);
- 2) CPU tersebut sama-sama menjalankan program dari suatu lokasi atau tempat, biasanya dari ROM (*Read Only Memory*) atau RAM (*Random Access Memory*);

- 3) Sama-sama memiliki RAM yang digunakan untuk menyimpan data-data sementara atau yang lebih dikenal dengan variabel-variabel; serta
- 4) Sama-sama memiliki beberapa keluaran dan masukan yang digunakan untuk melakukan komunikasi timbal-balik dengan dunia luar.

b. Perbedaan Microcontroller dengan Microcomputer

Lantas, apa yang membedakan antara *microcontroller* dengan *microcomputer*? Begitu mungkin pertanyaan yang ada di benak kita, saat kita membaca beberapa daftar kesamaan yang sudah saya tuliskan sebelumnya. Jelas antara kedua sama sekali berbeda, itu jawaban yang saya berikan kepada Anda. *Microcontroller* adalah versi mini dan untuk aplikasi khusus dari *microcomputer*.

Berikut saya berikan daftar kesamaan dengan menekankan pada perbedaan antara *microcontroller* dengan *microcomputer*:

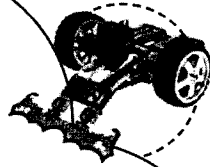
- 1) CPU pada *microcomputer* berada eksternal dalam suatu sistem, yang sampai saat ini kecepatan operasionalnya sudah mencapai tingkat lebih dari 2 GHz. Sedangkan CPU pada *microcontroller* berada internal dalam sebuah *chip*, yang kecepatan bekerja masih cukup

rendah, yaitu dalam orde MHz (misalnya, 24 MHz, 40 MHz, dan sebagainya). Kecepatan yang relatif rendah ini sudah mencukupi untuk aplikasi-aplikasi berbasis *microcontroller*.

- 2) CPU pada *microcomputer* menjalankan program dalam ROM atau yang lebih dikenal dengan BIOS pada saat awal dihidupkan, kemudian menjalankan program yang tersimpan dalam hard disk. Sedangkan CPU *microcontroller* sejak awal sudah menjalankan program yang tersimpan dalam ROM internal-nya (bisa berupa Mask ROM atau Flash PEROM). Sifat memori program ini *nonvolatile*, artinya tetap akan tersimpan walaupun tidak diberi catu daya.
- 3) RAM pada *microcomputer* bisa mencapai ukuran sekian MByte dan bisa di-*upgrade* ke ukuran yang lebih besar dan berlokasi di luar *chip* CPU-nya. Sedangkan RAM pada *microcontroller* ada di dalam *chip microcontroller* yang bersangkutan dan ukurannya sangat minim, misalnya 128 byte, 256 byte, dan seterusnya. Dan, dengan ukuran yang relatif kecil ini pun, dirasa cukup untuk aplikasi-aplikasi *microcontroller*.
- 4) Keluaran dan masukan pada *microcomputer* jauh lebih kompleks dibandingkan dengan

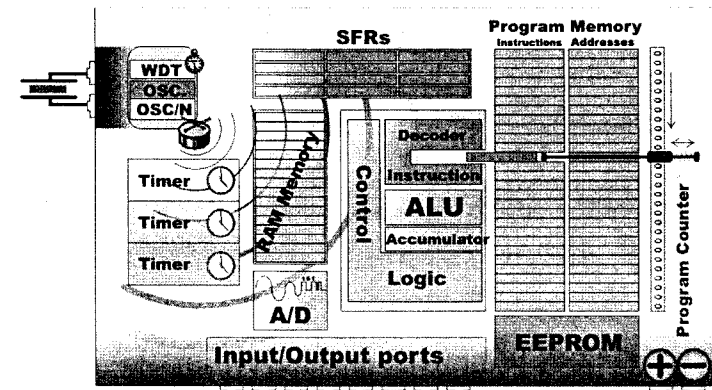
microcontroller, yang jauh lebih sederhana. Selain itu, pada *microcontroller* tingkat akses keluaran dan masukan bisa dalam satuan per bit.

Microcomputer merupakan komputer serbaguna atau *general purpose computer*, yang bisa dimanfaatkan untuk berbagai macam aplikasi (atau perangkat lunak).



- 5) Jika diamati lebih lanjut, bisa saya katakan bahwa *microcomputer* merupakan komputer serbaguna atau *general purpose computer*, yang bisa dimanfaatkan untuk berbagai macam aplikasi (atau perangkat lunak). Sedangkan *microcontroller* adalah *special purpose computer* atau komputer untuk tujuan khusus, hanya satu macam aplikasi saja.

Perhatikan gambar berikut agar Anda mendapatkan gambaran tentang *microcontroller*.



Gambar 14. Skema *microcontroller*
(agfi.staff.ugm.ac.id)

ALU, *instruction decoder*, *accumulator*, dan *control* merupakan “otak”-nya *microcontroller* yang bersangkutan. Jantungnya berasal dari detak OSC (lihat gambar 14). Sedangkan di sekeliling “otak” terdapat berbagai macam *peripheral*, seperti SFR (*Special Function Register*) yang bertugas menyimpan data-data sementara selama proses berlangsung, memori RAM yang tugasnya hampir sama seperti SFR, hanya saja tidak berhubungan langsung selama proses operasional *microcontroller*, ADC yang digunakan untuk mengubah data-data analog menjadi digital untuk diolah atau diproses lebih lanjut, EEPROM yang mempunyai fungsi sama seperti RAM, hanya saja tetap akan menyimpan data walaupun tidak mendapatkan sumber listrik/daya, dan *port-port* I/O

untuk masukan/luaran dalam melakukan komunikasi dengan pihak-pihak eksternal *microcontroller* (sensor dan aktuator).

c. Ciri Khas Microcontroller

Ciri-ciri khas dari microcontroller adalah:

- 1) "Tertanam" (*embedded*) dalam beberapa piranti (umumnya merupakan produk konsumen) atau yang dikenal dengan istilah *embedded system* atau *embedded controller*;
- 2) Terdedikasi untuk satu macam aplikasi saja (lihat contoh-contoh yang akan saya terangkan pada bagian berikutnya);
- 3) Hanya membutuhkan daya yang rendah (*low power*), sekitar 50 mwatt (coba Anda bandingkan dengan komputer yang bisa mencapai 50 watt lebih);
- 4) Memiliki beberapa keluaran maupun masukan yang terdedikasi untuk tujuan atau fungsi-fungsi khusus;
- 5) Kecil dan relatif lebih murah (seri AT89 di pasaran serendah-rendahnya bisa mencapai Rp15.000,00, sedangkan *basic stamp* bisa mencapai Rp500.000,00);
- 6) Lebih tahan banting, terutama untuk aplikasi-aplikasi yang berhubungan dengan mesin, otomotif, atau militer.

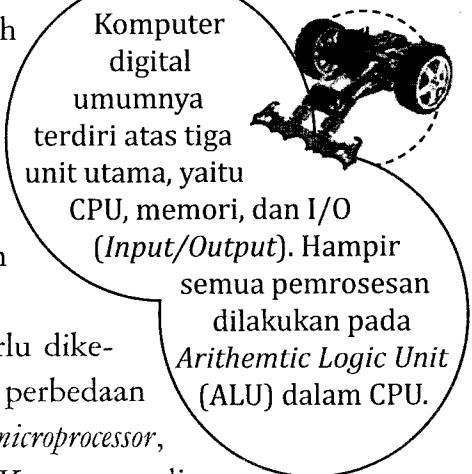
2. Perbedaan Microprocessor dan Microcontroller

Microprocessor yang umum terdapat dalam komputer biasanya bermerek INTEL, A.vlD, atau GYRIX. Pada sebuah komputer, terdapat *microprocessor* yang digunakan untuk memproses data dan mengkoordinasikan kerja sebuah komputer, yang dibantu oleh RAM (*Random Access Memory*), ROM (*Read Only Memory*), dan unit I/O pendukung lainnya.

Microprocessor merupakan bagian yang sangat penting dalam pengembangan ilmu pengetahuan dan teknologi. Pada tahun 1978, Intel memperkenalkan *microprocessor* 16 bit bernama 8086, yang merupakan pengembangan dari *microprocessor* sebelumnya, yaitu 8080/8085. *Microprocessor* 8086 adalah *microprocessor* dengan lebar bus data sebesar 16 bit secara internal dan eksternal.

Maksudnya, seluruh register lebarnya 16 bit dan ada bus data selebar 16 bit untuk mentransfer data ke dalam dan keluar CPU.

Sebelumnya, perlu diketahui terlebih dahulu perbedaan antara *minicomputer*, *microprocessor*, dan *microcontroller*. Komputer di-



gital umumnya terdiri atas tiga unit utama, yaitu CPU, memori, dan I/O (*Input/Output*). Hampir semua pemrosesan dilakukan pada *Arithmetic Logic Unit* (ALU) dalam CPU. Jika CPU dari sebuah komputer dibuat pada sebuah PCB, maka ini disebut sebagai *minicomputer*. *Microprocessor* adalah CPU yang dipaket menjadi 1 *chip*. Sedangkan *microcontroller* adalah keseluruhan komputer yang dibuat dalam 1 *chip*. Dengan berkembangnya teknologi *microprocessor* 8 bit dan 16 bit, muncul pula kebutuhan agar perangkat elektronika dapat dikemas sekecil mungkin, seperti Atari, Nintendo, Sega, PS2, dan peralatan hiburan serta peralatan rumah tangga seperti AC dan audio/video. Untuk mendukung hal tersebut, tidak dapat jika dilakukan oleh *microprocessor* standar. Sebab, *microprocessor* membutuhkan komponen eksternal tambahan seperti memori, pengolah analog ke digital, dan perangkat komunikasi serial. Oleh karena itu, dikembangkanlah *chip* yang di dalamnya sudah terdapat *microprocessor*, I/O pendukung, memori, dan ADC yang dikenal dengan istilah *microcontroller*.

Mungkin Anda akan bertanya, apa perbedaan *processor embedded* (prosesor yang dipasang pada sistem untuk tujuan tertentu) dengan *microcontroller*? Saat ini, mungkin sedikit sekali perbedaannya. Sebagai contoh, saat ini arsitektur standar prosesor yang berubah seperti *microcontroller* misalnya Motorola 68EC300,

Intel 386 EX, dan IBM PowerPC 403 GB. *Chip-chip* tersebut disebut sebagai *super-microcontrollers*. *Microcontroller* dapat juga disebut sebagai *one chip solution*, karena terdiri atas enam bagian berikut:

a. CPU (Central Processing Unit)

CPU merupakan bagian yang paling penting dari suatu *microprocessor*, yaitu untuk melakukan pemrosesan data.

b. RAM (Random Access Memory)

RAM digunakan untuk menyimpan data sementara.

c. EPROM/PROM/ROM (Erasable Programmable Read Only Memory)

ROM digunakan untuk menyimpan program yang bersifat permanen.

d. I/O (Input/Output) Serial dan Paralel

Unit ini berfungsi agar *microcontroller* dapat berkomunikasi secara serial atau paralel dengan PC dan peralatan standar digital lainnya.

e. Timer

Timer berguna untuk mengatur pewaktuan pada sistem berbasis *microcontroller*, misalnya untuk *delay* atau pencacah.

f. Interrupt Controller

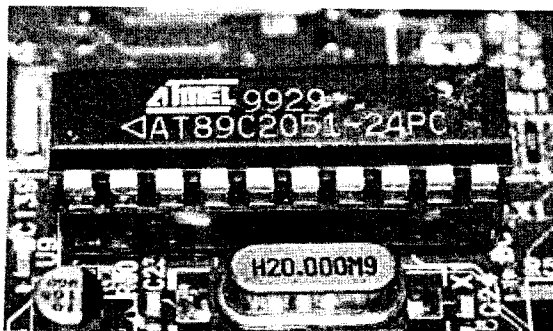
Interrupt controller berfungsi menangani suatu *request* pada saat *microcontroller* sedang *running*. Memang benar, bahwa *microcontroller*

digunakan untuk orientasi pengontrolan, seperti pengontrol temperatur, penampil display LCD, pemroses sinyal digital, pemroses dan pengontrol mesin-mesin industri, dan sebagainya. Dengan *microcontroller*, Anda dapat membuat robot hanya sebesar genggam tangan anak-anak.

3. Jenis-Jenis Microprocessor dan Microcontroller

a. Microcontroller ATMEL

Microcontroller keluaran ATMEL dapat dikatakan sebagai *microcontroller* terlaris dan termurah saat ini. *Chip microcontroller* ini dapat diprogram menggunakan port paralel atau serial. Selain itu, ia dapat beroperasi hanya dengan 1 *chip* dan beberapa komponen dasar seperti kristal, resistor, dan kapasitor. Silakan kunjungi situs www.atmel.com untuk melihat dan *download* informasi berbagai produk ATMEL.



Gambar 15. *Microcontroller* jenis ATMEL

(agfi.staff.ugm.ac.id)

Saya berharap, Anda dapat mengoleksi berbagai tipe *microcontroller* serta *kit*-nya sebagai perbandingan fitur, misalnya 8952051, 89551, 89353, Atmega8535 (mendukung ADC I channel), atau membeli *kit* hasil produksi, yang dapat Anda pesan di internet. Biasanya, orang Indonesia menggunakan *microcontroller* jenis ATMEL ini, selain *microcontroller* PIC.

Tabel 1. Beberapa Tipe Microcontroller ATMEL

Device	On-chip data memory (bytes)	On-chip program memory (flash)	No. Of 16-bit timer/counter	Digital I/O	Full duplex serial I/O	No of pin in (PDIP)	Precision on-chip analog comparator
AT89C51	128	4K	2	32	1	40	None
AT89C52	256	8K	3	32	1	40	None
AT89C55WD	256	20K	3	32	1	40	None
AT89C1051	64	1K	2	15	1	20	1
AT89C2051	128	2K	2	15	1	20	1
AT89C4051	128	4K	2	15	1	20	1
AT89LV52	256	8K	3	32	1	40	none

Untuk mempelajari lebih dalam mengenai pemrograman *microcontroller* ATMEL, Anda dapat membaca buku penulis sebelumnya.

b. Microcontroller PIC

PIC adalah keluarga *microcontroller* tipe RISC buatan Microchip Technology yang bersumber dari

PIC1650, yang dibuat oleh Divisi Mikroelektronika General Instruments. *Microcontroller* 8-bit PIC16CXX dan PIC17CXX dari *microchip* menggunakan teknologi CMOS. *Microcontroller* PIC terkenal karena performanya yang tinggi, biaya rendah, dan ukuran yang kecil. *Microcontroller* ini menggunakan arsitektur RICS. PIC16CXX hanya memiliki 33 *single-word* instruksi. Frekuensi operasi untuk 16CXX berjarak dari DC hingga 20 MHz. Seri ini dapat ditambahkan program memori eksternal hingga 64K word. PIC 17C42 memiliki beberapa pencacah/pewaktu dan kemampuan penanganan I/O, dan 16C71 telah terdapat 4 kanal 8-bit ADC.



Gambar 16. *Microcontroller* PIC 1650
(agfi.staff.ugm.ac.id)

Kita dapat memperoleh 12 kanal 10-bit ADC pada 17C752. Masih banyak lagi perangkat

microcontroller PIC lainnya yang tidak tertulis pada tabel. Fitur-fitur umum termasuk pewaktu, *watchdog*, ADC, memori data tambahan, komunikasi serial, keluaran *pulse width modulated* (PWM), dan memori ROM, EPROM, serta EEPROM.

Tabel 2. Beberapa Tipe *Microcontroller* PIC

Device	Pin (for DIP)	Digital I/O	ADC channel	EPROM x 12 word	RAM (bytes)
16C54	18	12	None	512	25
16C55	28	20	None	512	24
16C56	18	12	None	1K	25
16C57	28	20	None	2K	72
17C42A	40	33	None	2K	232
17C43	40	33	None	4K	454
17C44	40	33	None	8K	454
16C71	18	13	4 (8-bit ADC)	1K x 14	36
17C752	40	33	12 (10-bit ADC)	8K x 16	678

PIC pada awalnya dibuat menggunakan teknologi *General Instruments* 16 bit, yaitu CP1600. PIC dibuat pertama kali pada tahun 1975 untuk meningkatkan performa sistem pada UO. Saat ini, PIC telah dilengkapi dengan EPROM dan komunikasi serial, UART, kemel kontrol motor, serta memori program dari 572 word hingga 32 word 1. Word di sini sama dengan 1 instruksi bahasa *assembly* yang bervariasi mulai dari 72 hingga 16 bit, tergantung tipe PIC micro tersebut. Beberapa tipe PIC yang sangat terkenal

adalah PIC16F84A, PIC72F625, PIC12F629, PIC16F873, PIC16F874, dan PIC16F877. Saat ini, yang sedang tren adalah penggunaan *microcontroller* 16F84A dan 16F877, yang sudah mendukung ADC 8 *channel*. Untuk kontes robot KRCI, ada baiknya Anda mencoba PIC16F877, karena fitur-fiturnya mendukung pembuatan robot tersebut.

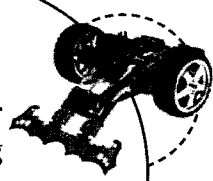
c. Microcontroller Maxim

Maxim merupakan salah satu produsen *chip* yang fokus pada komponen digital dan komunikasi, seperti *microcontroller*, akuisi data, dan komponen RF (frekuensi radio). Maxim cukup inovatif dengan meluncurkan *microcontroller* yang mendukung jaringan komputer, antara lain 80C400 yang berkecepatan tinggi. Anda dapat mengunjungi situs www.maxim-ic.com untuk melihat berbagai produk dan *download* data *sheet* atau contoh aplikasinya.

Beberapa *chip microcontroller* juga mendukung penggunaan *compiler* berbasis bahasa C, antara lain software Keil yang berfungsi sebagai *compiler C*, *macro assemblers*, *real-time kernels*, *debuggers*, dan simulator pada lingkungan IDE (*Interface Design Environment*) yang bagus. Saya sarankan, Anda mencoba *Evaluation Board* 80C400 dan MAXQ2000 untuk memahami fitur yang dimiliki *microcontroller* tersebut. Beberapa

chip microcontroller lain yang banyak digunakan adalah Basic Atom Arduino.

Beberapa *chip microcontroller* juga mendukung penggunaan *compiler* berbasis bahasa C, antara lain software Keil yang berfungsi sebagai *compiler C*, *macro assemblers*, *real-time kernels*, *debuggers*, dan simulator pada lingkungan IDE (*Interface Design Environment*) yang bagus.

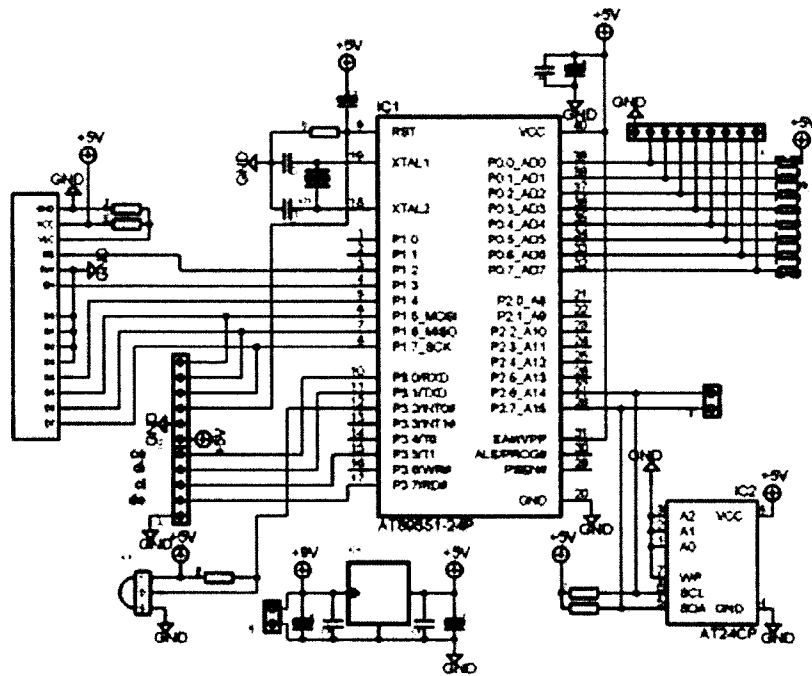


4. Mencoba Microcontroller

a. Microcontroller 89SXXX

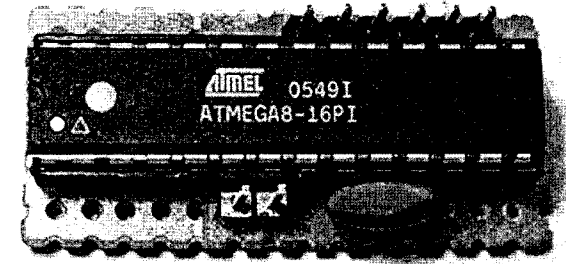
Microcontroller ATMEL

yang sering digunakan adalah 89551/52/8252. Lupakanlah *microcontroller* tipe 89c, karena sudah direkomendasikan untuk ditinggalkan oleh pabriknya. Untuk mencoba *microcontroller* bagi pemula, disarankan Anda membeli atau membuat sistem minimum *microcontroller* yang hanya terdiri atas 1 *chip*. Contoh berikut adalah ATMEL 89Sxxx Downloader yang mudah diperoleh dari situs internet.



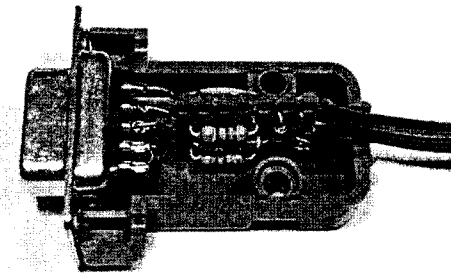
Gambar 17. Skema rangkaian AT89Sxxx
(2.bp.blogspot.com)

Untuk mengisinya, digunakan teknik ISP (*In System Programming*) yang telah didukung *microcontroller* versi 89Sxxx. Untuk itu, Anda dapat membuat *ISP Programmer* seperti contoh berikut, dengan menggunakan *software* ATMEL *Microcontroller* ISP. Rangkaian ini dihubungkan ke port p1.5, P7.6, P7.7, Reset, Cnd, dan Vcc *microcontroller*. Terima kasih kepada Bapak Tri (www.mytutorialcafe.com) atas rangkaian ini.

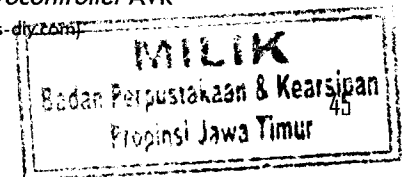


Gambar 18. *Microcontroller* ATMEL
(electronics-diy.com)

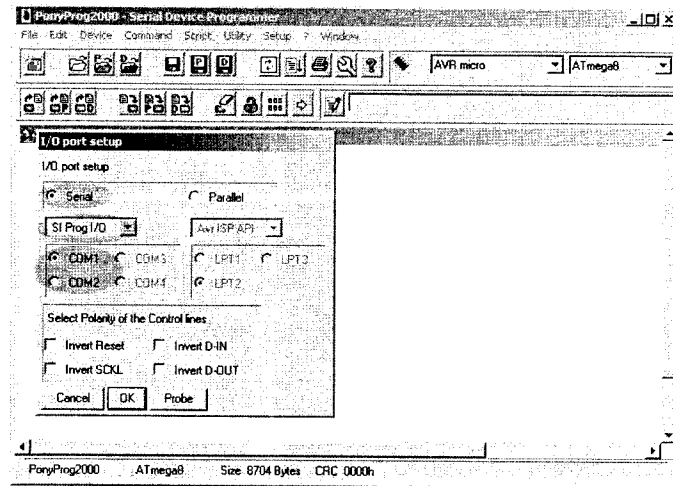
Bagi pemula, *microcontroller* yang belum memiliki *programmer microcontroller* AVR, tentu merupakan permasalahan serius. Oleh karena itu, mengapa tidak mencoba membuat *programmer* tersebut menggunakan rangkaian di bawah ini, dengan biaya yang terjangkau bagi para pelajar? Rangkaian ini dapat berjalan di hampir semua tipe *microcontroller* AVR yang umum beredar di pasaran.



Gambar 19. *Microcontroller* AVR
(electronics-diy.com)



Setelah alat selesai dibuat, maka proses selanjutnya adalah mengisi program ke dalam *microcontroller*. Yang perlu diperhatikan adalah konfigurasi I/O Port seperti gambar berikut:



Gambar 20. Konfigurasi I/O Port
(electronics-diy.com)

b. Belajar Membuat Program Assembly

Untuk memprogram *microcontroller*, biasanya digunakan bahasa *assembly*, PBASIC, atau c yang hasil akhirnya dikompilasi menjadi file ".hex". Jadi, tidak peduli bahasa sumbernya, hasil akhirnya pasti ".hex". Anda dapat membuat program *assembly* di editor Notepad. Saat menggunakan compiler ASM51, sebenarnya banyak pilihan untuk mengkompilasi

listing program bahasa assembler keluarga 8051, salah satunya ASM51. Prinsipnya, compiler ini akan mengubah file "*.ASM" menjadi "*.HEX". Berikut ini formatnya:

```
$MODxx
...
...
isi program bahasa assembler
...
...
END
```

Jadi, harus diawali dengan \$MODxx dan diakhiri dengan END. Pernyataan \$MOD merujuk kepada jenis IC xx yang digunakan. Contoh:

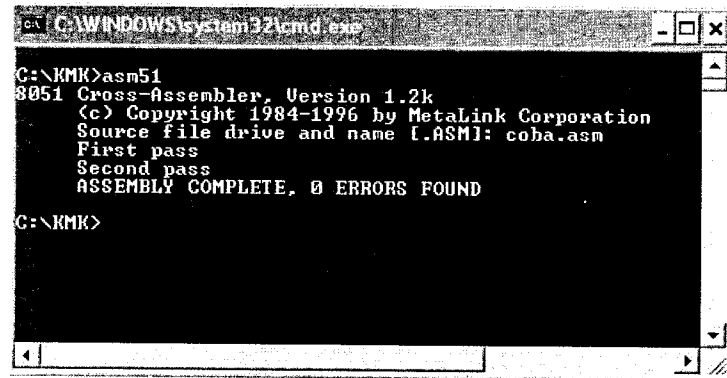
```
$MOD51; menggunakan IC xxx51 (bisa 89x51, 8051 dsb)
ORG      0000H
sSTART:  MOV P0,#0FH
        JMP      START
END
```

Sekarang, kita akan mengompilasikan contoh listing program di atas. Kita simpan dengan nama COBA.ASM. Sebelumnya, untuk memudahkan pastikan file/folder ASM51 dan file yang akan di kompilasi berada dalam satu folder. Sekarang, masuk ke *command prompt* (asumsi, menggunakan windows).

- Klik start - run - (ketik) cmd – enter
- Arahkan ke folder tempat ASM51 berada, lalu ketik

- ASM51 enter
- COBA.ASM enter

Perhatikan gambar berikut:



```
C:\WINDOWS\system32\cmd.exe
C:\NKK>asm51
8051 Cross-Assembler, Version 1.2k
(c) Copyright 1984-1996 by MetaLink Corporation
Source file drive and name [LASM]: coba.asm
First pass
Second pass
ASSEMBLY COMPLETE, 0 ERRORS FOUND
C:\NKK>
```

Gambar 21. Tampilan akhir sebuah program
(kelas-mikrokontrol.com)

Dari gambar 22 tersebut, tampak bahwa *assembly*-nya tuntas dan tidak ada error. Selamat mencoba.



Bab 2

Dasar-Dasar Elektronika Pembuat Robot

Belajar robot tanpa mengerti dasar-dasar elektronika, ibarat sayur tanpa garam. Sebagai bahan untuk memulai dasar pembelajaran tentang elektronika, seharusnya para pemula memulai kegiatan merakit robot dengan mempelajari dasar-dasar elektronika.

Dalam kehidupan sehari-hari, kita banyak menemui suatu alat yang mengadopsi elektronika sebagai basis teknologinya. Di rumah, kita sering melihat televisi, mendengarkan lagu melalui tape atau CD, mendengarkan radio, berkomunikasi dengan telepon. Di kantor, kita menggunakan komputer, mencetak dengan printer, mengirim pesan dengan *facsimile*, berkomunikasi dengan telepon. Di pabrik, kita memakai alat deteksi, mengoperasikan robot

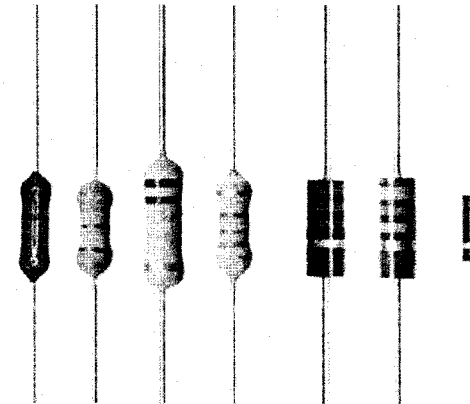
perakit, dan sebagainya. Bahkan, di jalan raya kita bisa melihat lampu lalu-lintas, lampu penerangan jalan yang secara otomatis hidup bila malam tiba, atau papan reklame yang terlihat indah berkelap-kelip; dan masih banyak contoh yang lainnya. Dari semua uraian tersebut kita dapat membuktikan bahwa pada zaman sekarang ini kita tidak akan lepas dari perangkat yang menggunakan elektronika sebagai dasar teknologinya.

Revolusi besar-besaran terhadap elektronika terjadi sekitar tahun 1960-an, di mana saat itu mulai ditemukan suatu alat elektronika yang dinamakan transistor. Dengan alat ini, memungkinkan kita untuk membuat suatu alat dengan ukuran yang kecil, di mana sebelumnya alat-alat tersebut masih menggunakan tabung-tabung vakum yang ukurannya besar serta mengonsumsi listrik yang besar.

Hanya dalam kurun waktu 10 tahun sejak ditemukannya transistor, kemudian ditemukan sebuah rangkaian terintegrasi yang dikenal dengan IC (*Integrated Circuit*) yang merupakan rangkaian terpadu berisi puluhan, bahkan jutaan, transistor di dalamnya. Sehingga, kini kita bisa melihat sebuah perangkat elektronika yang semakin kecil bentuknya, tetapi semakin banyak fungsinya, seperti telepon genggam (*hand phone*). Yah, semua itu berkat revolusi

silikon sebagai bahan dasar pembuatan transistor dan IC atau *chip*.

A. Resistor



Gambar 1. Jenis-jenis resistor
(infoservicetv.com)

Resistor adalah salah satu komponen dasar elektronika. Hampir bisa dipastikan, semua rangkaian elektro pasti memiliki resistor sebagai salah satu elemennya. Resistor, sesuai namanya yang berarti penghambat, berfungsi untuk menghambat arus listrik yang mengalir pada sebuah rangkaian. Arus listrik yang mengalir dapat diatur sesuai dengan hukum ohm:

$$V \text{ (volt)} = I \text{ (ampere)} \cdot R \text{ (ohm)}$$

Dari rumus tersebut, dalam tegangan konstan, semakin besar nilai hambatan, maka semakin kecil arus yang mengalir, dan sebaliknya.

1. Fungsi Resistor

Fungsi-fungsi resistor adalah:

- menghambat arus listrik;
- pembagi tegangan;
- pengatur volume (potensiometer);
- pengatur kecepatan motor (*rheostat*); dan sebagainya tergantung desain komponen.

2. Jenis-Jenis Resistor

a. Berdasarkan Fungsinya

Dilihat dari fungsinya, resistor dapat dibagi menjadi empat:

- Resistor tetap (*fixed resistor*) – nilai hambatan konstan

Resistor yang biasanya dibuat dari nikelin atau karbon ini berfungsi sebagai pembagi tegangan,

Hampir bisa dipastikan, semua rangkaian elektro pasti memiliki resistor sebagai salah satu elemennya. Resistor, sesuai namanya yang berarti penghambat, berfungsi untuk menghambat arus listrik yang mengalir pada sebuah rangkaian.



mengatur atau membatasi arus pada suatu rangkaian, serta memperbesar dan memperkecil tegangan. Ada juga resistor yang dibuat khusus, yaitu resistor untuk tegangan tinggi (misalnya dalam televisi) yang terbuat dari selaput karbon dalam kapsul vakum, resistor megaohm tinggi (mencapai 10^6 Mohm) yang terbuat dari gelas semikonduktor (digunakan untuk FET, detector radiasi, elektrometer, resistor DIL/*Dual in Line*).

- Resistor tidak tetap (*variabel resistor*) – nilai hambatan konstan

Resistor ini berfungsi sebagai pengatur volume (mengatur besar kecilnya arus), *tone control* pada *sound system*, dan pengatur tinggi rendahnya nada (*bass/treble*). Selain itu, resistor ini juga berfungsi sebagai pembagi tegangan arus dan tegangan, misalnya potensiometer, trimpot (*trimmer potensiometer*), *rheostat*, dan *multiturn*.

- Resistor NTC dan PTC

NTC (*Negative Temperature Coefficient*) nilainya akan bertambah kecil bila terkena suhu panas. Sedangkan PTC (*Positive Temperature Coefficient*) nilainya akan bertambah besar bila temperaturnya menjadi panas.

4) LDR

LDR (*Light Dependent Resistor*) adalah suatu jenis resistor yang berubah hambatannya karena pengaruh cahaya. Bila terkena cahaya gelap, nilai tahanannya semakin besar, sedangkan bila terkena cahaya terang, nilainya menjadi semakin kecil.

b. Berdasarkan Bentuknya

Dilihat dari bentuknya, resistor memiliki bentuk standar atau gulungan kawat (*rheostat*), seperti IC (DIL).

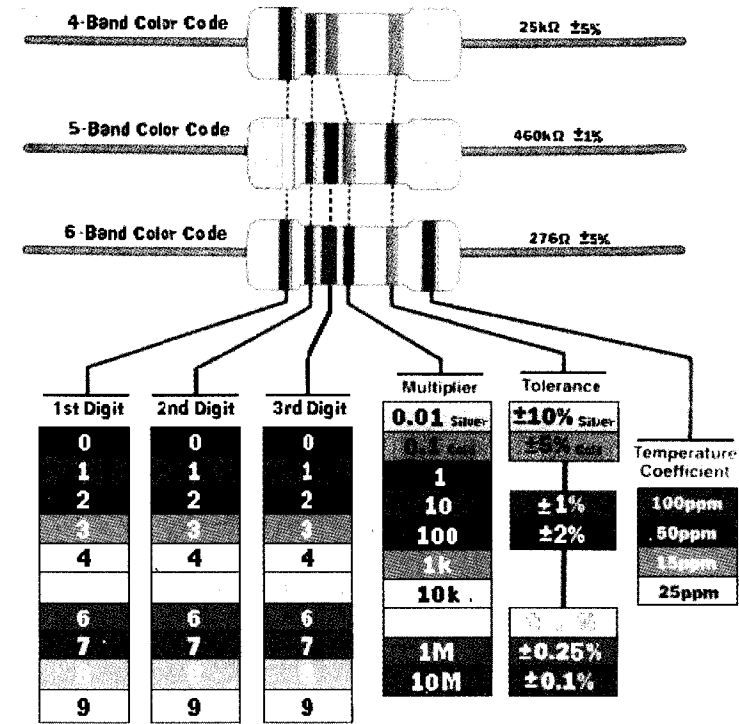
c. Berdasarkan Besarnya Daya Kerja

Sedangkan dilihat dari besarnya daya kerja yang digunakan, di pasaran terdapat beragam resistor yang memiliki daya yang berbeda, mulai dari $\frac{1}{8}$, $\frac{1}{4}$, $\frac{1}{2}$, 1, 2, 3, 5, 10, dan 20 watt. Resistor yang memiliki disipasi daya 5, 10, dan 20 watt biasanya berbentuk kubik berwarna putih atau silinder.

3. Mengetahui Nilai Resistor

a. Pita/Kode Warna

Pada resistor karbon jenis ini, kode-kode warna yang berbentuk pita menunjukkan besaran hambatan. Masing-masing warna mewakili hambatan tertentu.



Gambar 2. Kode warna pada resistor

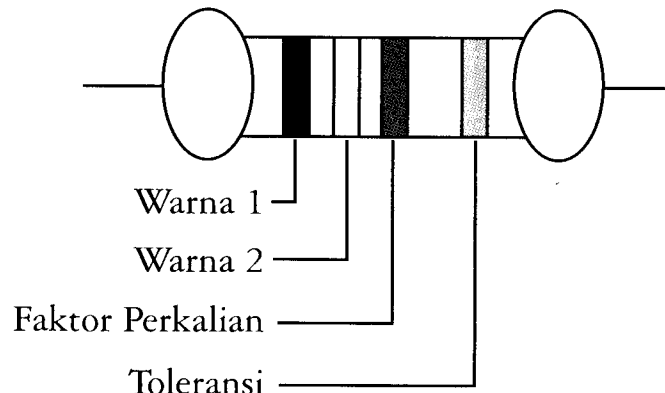
(belajardong.files.wordpress.com)

Tabel 1. Jenis warna dan nilai hambatannya

Pita	Resistor Tiga Pita	Resistor Lima Pita
Warna 1	nilai hambatan digit ^{1st}	nilai hambatan digit ^{1st}
Warna 2	nilai hambatan digit ^{2nd}	nilai hambatan digit ^{2nd}

Warna 3	faktor pengali (x 10 ⁿ)	nilai hambatan digit 3 rd
Warna 4	% toleransi	faktor pengali (x 10 ⁿ)
Warna 5	-	% toleransi
Warna 6	-	-

Contoh cara pembacaan:



Gambar 3. Menghitung nilai resistor

(belajardong.files.wordpress.com)

Warna 1 → Merah = 2

Warna 2 → Kuning = 4

Warna 3 → Jingga = x1.000

Warna 4 → Emas = 5%

Jadi, nilai resistansi resistor pada Gambar 3 adalah 24.000 Ohm = 24K.

$$R_{\text{maks}} = 24.000 + (5\% \times 24.000)$$

$$R_{\text{min}} = 24.000 - (5\% \times 24.000)$$

b. AVO meter (Ampere – Voltage – Ohmmeter)



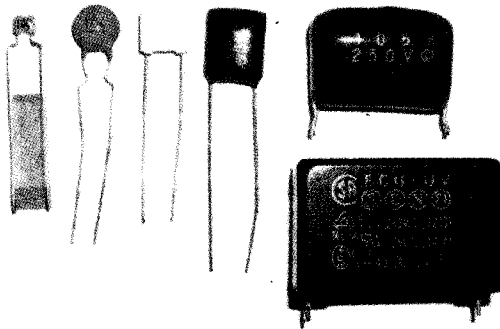
Gambar 4. AVO meter analog

(geraiproduk.com)

Untuk AVO meter analog (dengan skala besaran dan jarum penunjuk) dan AVO meter digital, besarnya hambatan dapat langsung diketahui dari jarum/angka yang ditunjukkan. Jangan lupa untuk mengatur faktor pengali sebelum mengukur × 1 ohm, × 10 ohm, atau × 1K ohm. Pada saat pertama kali mengukur, sebaiknya kita gunakan pengali terbesar (× 1K ohm) terlebih dahulu. Apabila pergeseran jarum terlalu sedikit atau angka terlalu besar, maka kecilkan faktor pengali.

B. Kapasitor (Kondensator)

Kapasitor (kondensator) yang dalam rangkaian elektronika dilambangkan dengan huruf "C" adalah suatu alat yang dapat menyimpan energi/muatan listrik di dalam medan listrik, dengan cara mengumpulkan ketidakseimbangan internal dari muatan listrik. Kapasitor ditemukan oleh Michael Faraday (1791–1867). Satuan kapasitor disebut Farad (F). Satu Farad = $9 \times 10^{11} \text{ cm}^2$, yang menandakan luas permukaan kepingan tersebut.



Gambar 5. Aneka jenis kapasitor
(ilmu-elektronika.co.cc)

Struktur sebuah kapasitor terbuat dari 2 buah plat metal yang dipisahkan oleh suatu bahan dielektrik. Bahan-bahan dielektrik yang umum dikenal adalah udara vakum, keramik, gelas, dan lain-lain. Jika kedua ujung plat metal diberi tegangan lis-

trik, maka muatan-muatan positif akan mengumpul pada salah satu kaki (elektroda) metalnya dan pada saat yang sama, muatan-muatan negatif terkumpul pada ujung metal yang satu lagi. Muatan positif tidak dapat mengalir menuju ujung kutub negatif dan sebaliknya muatan negatif tidak bisa menuju ke ujung kutub positif, karena terpisah oleh bahan dielektrik yang nonkonduktif. Muatan elektrik ini tersimpan selama tidak ada konduksi pada ujung-ujung kakinya. Di alam bebas, fenomena kapasitor ini terjadi pada saat terkumpulnya muatan-muatan positif dan negatif di awan.

Struktur sebuah kapasitor terbuat dari 2 buah plat metal yang dipisahkan oleh suatu bahan dielektrik. Bahan-bahan dielektrik yang umum dikenal adalah udara vakum, keramik, gelas, dan lain-lain.



1. Kapasitansi

Kapasitansi didefinisikan sebagai kemampuan dari suatu kapasitor untuk dapat menampung muatan elektron. Pada abad ke-18, Coulombs menghitung bahwa $1 \text{ coulomb} = 6,25 \times 10^{18}$ elektron. Kemudian, Michael Faraday membuat postulat bahwa sebuah kapasitor akan memiliki kapasitansi sebesar 1 farad jika dengan tegangan

1 volt dapat memuat muatan elektron sebanyak 1 coulombs. Dengan rumus dapat dituliskan bahwa:

$$Q = C V$$

Q = muatan elektron dalam C (coulombs)

C = nilai kapasitansi dalam F (farad)

V = besar tegangan dalam V (volt)

Dalam praktik pembuatan kapasitor, kapasitansi dihitung dengan mengetahui luas area plat metal (A), jarak (t) antara kedua plat metal (tebal dielektrik), dan konstanta (k) bahan dielektrik. Dengan rumus dapat di tulis:

$$C = (8,85 \times 10^{-12}) (k A/t)$$

Berikut adalah tabel contoh konstanta (k) dari beberapa bahan dielektrik yang disederhanakan.

Tabel 2. Konstanta bahan k

udara vakum	$k = 1$
aluminium oksida	$k = 8$
keramik	$k = 100-1.000$
gelas	$k = 8$
polyethylene	$k = 3$

Untuk rangkaian elektronik praktis, satuan farad adalah sangat besar. Umumnya, kapasitor yang ada di pasaran memiliki satuan μF , nF , dan pF .

1 Farad = 1.000.000 μF (mikro Farad)

1 μF = 1.000.000 pF (piko Farad)

1 μF = 1.000 nF (nano Farad)

1 nF = 1.000 pF (piko Farad)

1 pF = 1.000 $\mu\mu F$ (mikro-mikro Farad)

1 μF = 10^{-6} F

1 nF = 10^{-9} F

1 pF = 10^{-12} F

Konversi satuan penting untuk diketahui agar memudahkan saat membaca besaran sebuah kapasitor. Misalnya, saat sebuah kapasitor bertuliskan 0.047 μF , maka Anda dapat juga membacanya sebagai 0,047 μF atau 47 nF . Contoh lain, saat sebuah kapasitor bertuliskan 0.1 nF , maka Anda dapat membacanya sebagai 0,1 nF atau 100 pF .

Kondensator diidentikkan mempunyai dua kaki yang mempunyai dua kutub (positif dan negatif), memiliki cairan elektrolit, dan biasanya berbentuk tabung. Sedangkan kondensator jenis satunya kebanyakan nilai kapasitansinya lebih rendah, tidak mempunyai kutub positif atau negatif pada kakinya, kebanyakan berbentuk bulat pipih berwarna cokelat,

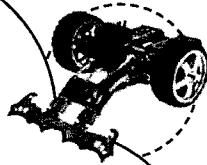
merah, hijau, dan lainnya seperti tablet atau kancing baju yang sering disebut kapasitor (*capacitor*).

2. Wujud dan Macam Kondensator

Berdasarkan kegunaannya, kondensator dapat di bedakan menjadi tiga, yaitu kondensator tetap (nilai kapasitansinya tetap tidak dapat diubah), kondensator elektrolit (*Electrolyte Condenser* = Elco), dan kondensator variabel (nilai kapasitansinya dapat diubah-ubah).

Pada kapasitor yang berukuran besar, nilai kapasitansi umumnya ditulis dengan angka yang jelas, lengkap dengan nilai tegangan maksimum dan polaritasnya. Misalnya, pada kapasitor Elco, dengan jelas tertulis kapasitansinya sebesar $100\mu\text{F}25\text{v}$ yang artinya kapasitor/kondensator tersebut memiliki nilai kapasitansi $100\mu\text{F}$ dengan tegangan kerja maksimal yang diperbolehkan sebesar 25 volt.

Kondensator diidentikkan mempunyai dua kaki yang mempunyai dua kutub (positif dan negatif), memiliki cairan elektrolit, dan biasanya berbentuk tabung.



Gambar 6. Jenis-jenis kondensator
(rv007.blogspot.com)

Kapasitor yang ukuran fisiknya kecil, biasanya hanya bertuliskan 2 (dua) atau 3 (tiga) angka saja. Jika hanya ada dua angka, satuannya adalah pF (*pico farads*). Sebagai contoh, kapasitor yang bertuliskan dua angka 47, maka kapasitansi kapasitor tersebut adalah 47 pF. Jika ada 3 digit, angka pertama dan kedua menunjukkan nilai nominal, sedangkan angka ketiga adalah faktor pengali. Faktor pengali sesuai dengan angka nominalnya, berturut-turut adalah $1 = 10$, $2 = 100$, $3 = 1.000$, $4 = 10.000$, $5 = 100.000$, dan seterusnya. Sebagai contoh, perhatikan tabel berikut:

Tabel 3. Contoh perhitungan pada sebuah kapasitor

104	105	222
$104 = 10 \times 10.000$	$105 = 10 \times 100.000$	$222 = 22 \times 100$
$= 100.000 \text{ pF}$	$= 1.000.000 \text{ pF}$	$= 2.200 \text{ pF}$
$= 100 \text{ nF}$	$= 1.000 \text{ nF}$	$= 2,2 \text{ nF}$
$= 0,1 \mu\text{F}$	$= 1 \mu\text{F}$	$= 0,0022 \mu\text{F}$

Untuk kapasitor polyester, nilai kapasitansinya bisa diketahui berdasarkan warna, seperti pada resistor. Seperti komponen lainnya, besarnya kapasitansi nominal pasti ada toleransinya. Pada Tabel 3, diperlihatkan nilai toleransi dengan kode-kode angka atau huruf tertentu. Dengan tabel tersebut, pemakai dapat dengan mudah mengetahui toleransi kapasitor yang biasanya tertera menyertai nilai nominal kapasitor. Misalnya, jika tertulis 104 X7R, maka kapasitansinya adalah 100 nF dengan toleransi $\pm 15\%$. Dari angka tersebut juga diketahui bahwa suhu kerja yang direkomendasikan adalah antara -55°C sampai $+125^\circ\text{C}$.

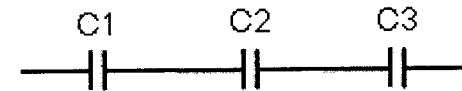
Dari penjelasan tersebut, bisa diketahui bahwa karakteristik kapasitor, selain kapasitansi, yang juga tak kalah pentingnya adalah tegangan dan temperatur kerja. Tegangan kerja adalah tegangan maksimum yang diizinkan, sehingga kapasitor masih dapat bekerja dengan baik. Misalnya kapasitor

10uF25V, maka tegangan yang bisa diberikan tidak boleh melebihi 25 volt dc. Umumnya, kapasitor-kapasitor polar bekerja pada tegangan DC dan kapasitor nonpolar bekerja pada tegangan AC.

Sedangkan temperatur kerja adalah batasan temperatur di mana kapasitor masih bisa bekerja dengan optimal. Misalnya, jika pada kapasitor tertulis X7R, maka kapasitor tersebut mempunyai suhu kerja yang direkomendasikan antara -55°C sampai $+125^\circ\text{C}$. Biasanya, spesifikasi karakteristik ini disajikan oleh pabrik pembuat *datasheet*.

3. Rangkaian Kapasitor

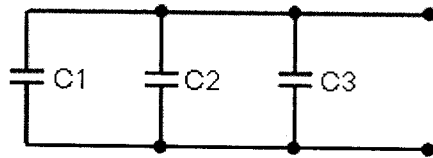
Rangkaian kapasitor secara seri akan mengakibatkan nilai kapasitansi total semakin kecil. Di bawah ini contoh kapasitor yang dirangkai secara seri:



Pada rangkaian kapasitor yang dirangkai secara seri berlaku rumus:

$$\frac{1}{C_{TOTAL}} = \frac{1}{C_1} + \frac{1}{C_2} + \frac{1}{C_3}$$

Rangkaian kapasitor secara paralel akan mengakibatkan nilai kapasitansi pengganti semakin besar. Di bawah ini contoh kapasitor yang dirangkai secara paralel.



Pada rangkaian kapasitor paralel berlaku rumus:

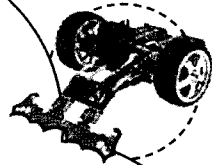
$$C_{TOTAL} = C_1 + C_2 + C_3$$

4. Fungsi Kapasitor

Fungsi penggunaan kapasitor dalam suatu rangkaian adalah:

- sebagai kopling antara rangkaian yang satu dengan rangkaian yang lain (pada PS = *Power Supply*);
- sebagai filter dalam rangkaian PS;

Rangkaian kapasitor secara seri akan mengakibatkan nilai kapasitansi total semakin kecil. Sedangkan rangkaian kapasitor secara paralel akan mengakibatkan nilai kapasitansi pengganti semakin besar.



- sebagai pembangkit frekuensi dalam rangkaian antena;
- untuk menghemat daya listrik pada lampu neon; dan
- untuk menghilangkan *bouncing* (loncatan api) bila dipasang pada saklar.

5. Tipe Kapasitor

Kapasitor terdiri atas beberapa tipe, tergantung dari bahan dielektriknya. Untuk lebih sederhana, kapasitor dapat dibagi menjadi tiga bagian, yaitu kapasitor *electrostatic*, *electrolytic*, dan *electrochemical*.

a. Kapasitor Electrostatic

Kapasitor *electrostatic* adalah kelompok kapasitor yang dibuat dengan bahan dielektrik dari keramik, film, dan mika. Keramik dan mika adalah bahan yang populer dan murah untuk membuat kapasitor yang kapasitansinya kecil. Kapasitor ini tersedia dari besaran pF sampai beberapa μF , yang biasanya untuk aplikasi rangkaian yang berkenaan dengan frekuensi tinggi, termasuk kelompok bahan dielektrik film. Bahan dielektrik film adalah bahan-bahan material seperti *polyester* (*polyethylene terephthalate* atau dikenal dengan sebutan *mylar*), *polystyrene*, *polypropylene*, *polycarbonate*, *metalized paper*, dan lainnya.

Mylar, MKM, dan MKT adalah beberapa contoh sebutan merek dagang untuk kapasitor dengan bahan-bahan dielektrik film. Umumnya, kapasitor kelompok ini adalah nonpolar.

b. Kapasitor Electrolytic

Kelompok kapasitor *electrolytic* terdiri dari kapasitor-kapasitor yang bahan dielektriknya adalah lapisan metal-oksida. Umumnya, kapasitor yang termasuk kelompok ini adalah kapasitor polar dengan tanda + dan - di badannya. Mengapa kapasitor ini dapat memiliki polaritas? Sebab, proses pembuatannya menggunakan elektrolisa, sehingga terbentuk kutub positif anoda dan kutub negatif katoda.

Telah lama diketahui beberapa metal seperti tantalum, aluminium, magnesium, titanium, niobium, zirconium, dan seng (zinc) mempunyai permukaan yang dapat dioksidasi, sehingga membentuk lapisan metal-oksida (*oxide film*). Lapisan oksidasi ini terbentuk melalui proses elektrolisa, seperti pada proses penyepuhan emas. Elektroda metal yang dicelup dalam larutan elektrolit (*sodium borate*) ini kemudian diberi tegangan positif (anoda) dan larutan elektrolit diberi tegangan negatif (katoda). Oksigen pada larutan elektrolit terlepas dan mengoksidasi permukaan plat metal. Contohnya, jika digunakan

aluminium, maka akan terbentuk lapisan Aluminium-Oksida (Al_2O_3) pada permukaannya.

Dengan demikian, berturut-turut plat metal (anoda), lapisan metal-oksida, dan elektrolit (katoda) membentuk kapasitor. Dalam hal ini, lapisan metal-oksida sebagai dielektrik. Lapisan metal-oksida ini sangat tipis, sehingga dengan dapat dibuat kapasitor yang kapasitansinya cukup besar.

Karena alasan ekonomis dan praktis, umumnya bahan metal yang banyak digunakan adalah aluminium dan tantalum, dengan alasan bahan yang paling banyak ditemukan dan harganya murah. Untuk mendapatkan permukaan yang luas, bahan plat aluminium ini biasanya digulung radial, sehingga diperoleh kapasitor yang kapasitansinya besar, contoh 100 μF , 470 μF , 4700 μF , dan lain-lain, yang sering juga disebut kapasitor *Elco*.

Bahan elektrolit pada kapasitor tantalum ada yang berbentuk cair, ada juga yang padat. Disebut elektrolit padat karena sebenarnya bukan larutan elektrolit yang menjadi elektroda negatif-nya, melainkan bahan lain yaitu manganese-dioksida. Dengan demikian, kapasitor jenis ini bisa memiliki kapasitansi yang besar, namun menjadi lebih ramping dan mungil. Selain itu, karena seluruh bagiannya padat, maka waktu kerja bahan ini (*lifetime*) menjadi lebih tahan lama. Kapasitor tipe ini juga memiliki arus

bocor yang sangat kecil. Jadi, dapat dipahami mengapa kapasitor tantalum menjadi relatif mahal.

c. Kapasitor Electro-chemical

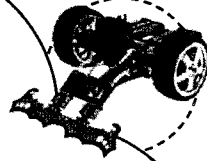
Satu jenis kapasitor lain adalah kapasitor *electrochemical*. Benda yang termasuk kapasitor jenis ini adalah baterai dan aki. Pada kenyataannya, baterai dan aki adalah kapasitor yang sangat baik, karena memiliki kapasitansi yang besar dan arus bocor (*leakage current*) yang sangat kecil. Tipe kapasitor jenis ini juga masih dalam pengembangan untuk mendapatkan kapasitansi yang besar namun kecil dan ringan, misalnya untuk aplikasi mobil elektrik, robot, dan telepon seluler.

C. Dioda

Dalam elektronika, dioda adalah komponen aktif bersaluran dua (dioda termionik mungkin memiliki saluran ketiga sebagai pemanas). Dioda mempunyai dua elektroda aktif, di mana isyarat listrik dapat mengalir. Dioda ini banyak digunakan karena ka-

Disebut elektrolit padat karena sebenarnya bukan larutan elektrolit

yang menjadi elektroda negatif-nya, melainkan bahan lain yaitu manganes-dioksida.



rakteristik satu arah yang dimilikinya. Dioda varikap (*VARIABLE CAPACITOR*/kondensator variabel) digunakan sebagai kondensator terkendali tegangan.



Gambar 7. Dioda
(rv007.blogspot.com)

Sifat kesearahan yang dimiliki sebagian besar jenis dioda seringkali disebut karakteristik menyearahkan. Fungsi paling umum dari dioda adalah untuk memperbolehkan arus listrik mengalir dalam suatu arah (disebut kondisi panjar maju) dan untuk menahan arus dari arah sebaliknya (disebut kondisi panjar mundur). Karenanya, dioda dapat dianggap sebagai versi elektronik dari katup pada transmisi cairan.

Dioda sebenarnya tidak menunjukkan kesearahan hidup-mati yang sempurna (benar-benar menghantar saat panjar maju dan menyumbat pada

panjang mundur), tetapi mempunyai karakteristik listrik tegangan arus tak linear kompleks yang bergantung pada teknologi yang digunakan dan kondisi penggunaannya. Beberapa jenis dioda juga mempunyai fungsi yang tidak ditujukan untuk penggunaan penyearahan.

Awalnya, dioda merupakan peranti kristal Cat's Whisker dan tabung hampa (juga disebut katup termionik). Namun, saat ini, dioda yang paling umum dibuat adalah dari bahan semikonduktor seperti silikon atau germanium.

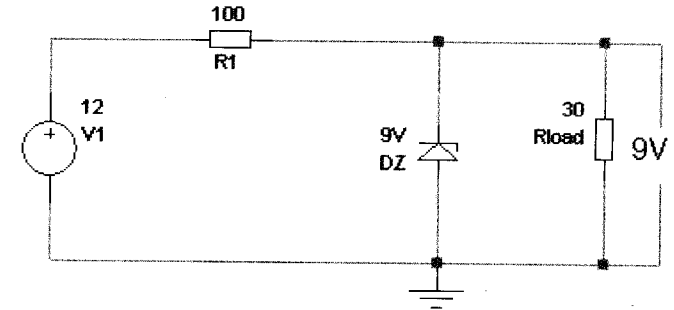
Dioda dapat dibedakan menjadi tiga jenis, yaitu *diode zener*, LED, dan *photodiode*.

1. Diode Zener

Ketika tegangan *reverse-bias* maksimum diberikan kepada dioda, maka arus listrik akan mengalir seperti layaknya pada keadaan *forward-bias*. Arus listrik ini tidak akan merusak dioda jika tidak melebihi dari apa yang telah ditentukan. Ketika tegangan *reverse-bias* ini dapat dikendalikan pada level tertentu, maka dioda ini disebut sebagai dioda zener.

Dioda zener memiliki nilai tegangan yang telah ditentukan dalam pembuatannya. Nilai tegangan ini mempunyai rentang dari beberapa volt hingga ratusan volt dan toleransi dioda zener berkisar antara 5%–10%. Pada aplikasinya di dalam rangkaian

elektronika, dioda zener berfungsi sebagai pengatur tegangan (regulator) dengan berperan sebagai beban.



Gambar 8. Skema rangkaian dioda zener
(ilmu-elektronika.co.cc)

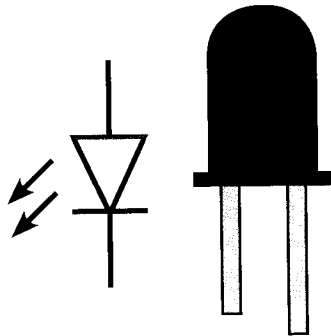
Dioda zener akan mengalirkan banyak arus listrik jika tegangan terlalu tinggi, dan mengurangi arus listrik jika tegangan terlalu rendah, sehingga menyebabkan tegangan stabil. Seperti pada contoh Gambar 8, tegangan dari sumber tegangan adalah 12 volt, tetapi tegangan yang terukur pada Rload adalah 9 volt, sama dengan nilai tegangan dioda zener.

Dioda zener memiliki nilai tegangan yang telah ditentukan dalam pembuatannya. Nilai tegangan ini mempunyai rentang dari beberapa volt hingga ratusan volt dan toleransi dioda zener berkisar antara 5%–10%.



2. LED

LED (*Light Emitting Diodes*) merupakan jenis dioda yang jika diberikan tegangan *forward*-bias akan menimbulkan cahaya dengan warna-warna tertentu, seperti merah, hijau, dan kuning.



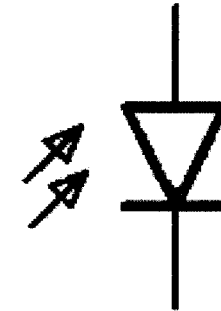
Gambar 9. Simbol LED
(ilmu-eletronika.co.cc)

Simbol LED hampir sama dengan simbol dioda, hanya saja pada simbol LED ditambahkan dua garis panah ke arah luar seperti ilustrasi pada Gambar 9. LED dalam rangkaian elektronika biasa digunakan sebagai lampu indikator.

3. Photodiode

Photodiode adalah dioda yang bekerja berdasarkan intensitas cahaya, di mana jika *photodiode* terkena cahaya, maka *photodiode* bekerja seperti dioda

pada umumnya. Akan tetapi, jika *photodiode* tidak mendapat cahaya, maka ia akan berperan seperti resistor dengan nilai tahanan yang besar, sehingga arus listrik tidak dapat mengalir.



Gambar 10. Photodiode
(ilmu-elektronika.co.cc)

Simbol dan bentuk *photodiode* hampir sama dengan LED, tetapi pada simbol *photodiode* arah dua panahnya menghadap ke dalam. *Photodiode* banyak digunakan sebagai sensor cahaya dalam dunia elektronika, karena sifatnya yang peka terhadap cahaya.

D. Sirkuit Terintegrasi (Integrated Circuit)

Sirkuit terintegrasi atau yang biasa juga disebut sebagai IC merupakan komponen elektronika yang terbuat dari kumpulan puluhan, ratusan, hingga ribuan transistor, resistor, dioda, dan komponen

elektronika lainnya.

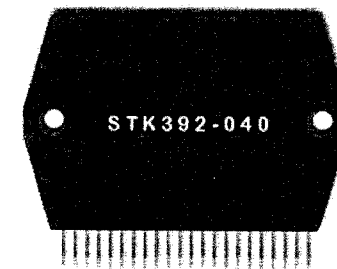
Kumpulan komponen-komponen tersebut dikemas dengan kompak sedemikian rupa hingga ukurannya tidak terlalu besar. IC dibuat agar memiliki fungsi tertentu, misalnya seperti penguat audio (audio amplifier), regulator tegangan, penerima gelombang radio, dan sebagainya.

Sirkuit terintegrasi atau yang biasa juga disebut sebagai IC merupakan komponen elektronika yang terbuat dari kumpulan puluhan, ratusan, hingga ribuan transistor, resistor, dioda, dan komponen elektronika lainnya.



1. Jenis Sirkuit Terintegrasi

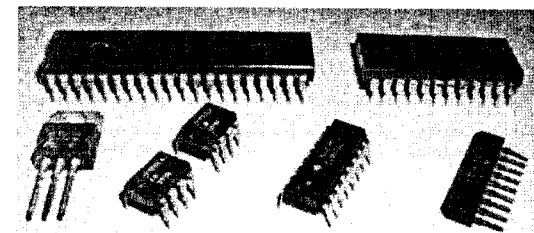
Menurut bagaimana sirkuit terintegrasi tersebut dibuat, sirkuit terintegrasi terbagi ke dalam dua kategori, yaitu *hybrid* dan *monolithic*. Sirkuit terintegrasi *hybrid* bisa juga disebut sebagai rangkaian miniatur elektronika, di mana di dalamnya terdapat transistor, dioda, kapasitor, resistor, bahkan koil yang dirangkai secara kompak pada PCB (*Printed Circuit Board*/papan sirkuit tercetak), yang kemudian dienkapsulasi menggunakan bahan *epoxy*. Contoh penggunaan sirkuit terintegrasi *hybrid* adalah sirkuit terintegrasi pada penguat audio (*audio amplifier*) "STK".



Gambar 11. Sirkuit terintegrasi

(ilmu-elektronika.co.cc)

Sekarang, sirkuit terintegrasi *hybrid* sudah jarang ditemui, karena ukurannya yang masih dianggap terlalu besar untuk perangkat elektronik pada saat ini. Akhirnya, sirkuit terintegrasi *monolithic* mulai menggeser keberadaan sirkuit terintegrasi *hybrid* dengan ukuran yang lebih kecil. Pada sirkuit terintegrasi *monolithic*, semua komponen (transistor, dioda, resistor, dan sebagainya) ditempatkan pada plat silikon yang sangat kecil, kemudian dienkapsulasi menggunakan plastik atau keramik.



Gambar 12. Jenis-jenis sirkuit terintegrasi

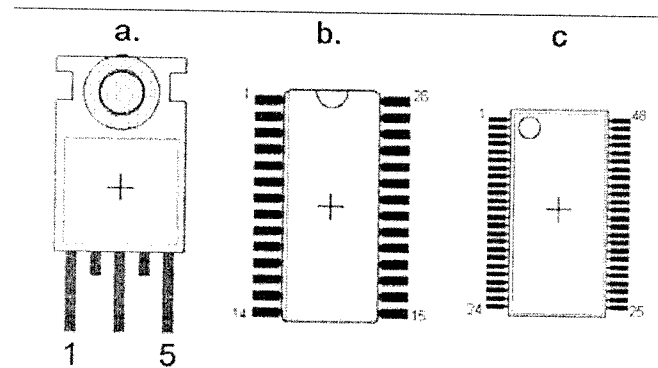
(ilmu-elektronika.co.cc)

Perbedaan antara *hybrid* dan *monolithic* terletak pada kapasitor dan kol. Pada sirkuit terintegrasi *monolithic*, komponen seperti kapasitor dan koil tidak dapat dimasukkan ke dalamnya, sehingga untuk menambahkan komponen tersebut perlu ditambahkan kaki-kaki di luar enkapsulasi untuk menghubungkan komponen tersebut dengan komponen yang ada di dalam sirkuit terintegrasi. Sementara, pada sirkuit terintegrasi *hybrid*, kapasitor dan koilnya sudah langsung dapat dimasukkan ke dalamnya.

2. Identifikasi Kaki-Kaki Sirkuit Terintegrasi

Pada umumnya, sirkuit terintegrasi memiliki jumlah kaki lebih dari tiga buah. Lalu, bagaimana mengidentifikasi kaki pertama, kedua, ketiga, dan seterusnya pada sebuah sirkuit terintegrasi/IC? Caranya adalah dengan melihat tanda-tanda khusus yang diberikan pada sebuah IC. Tanda khusus ini bisa berupa titik, logo perusahaan, lengkungan, dan sebagainya.

Pada contoh Gambar 13a., kaki pertama terletak pada kaki paling kiri dengan IC menghadap ke depan. Biasanya, jika IC ini dikemas (*packaging*) seperti pada Gambar 13a., maka kaki pertama selalu berada di paling kiri, dilanjutkan kaki kedua, dan seterusnya berada pada samping kanannya.

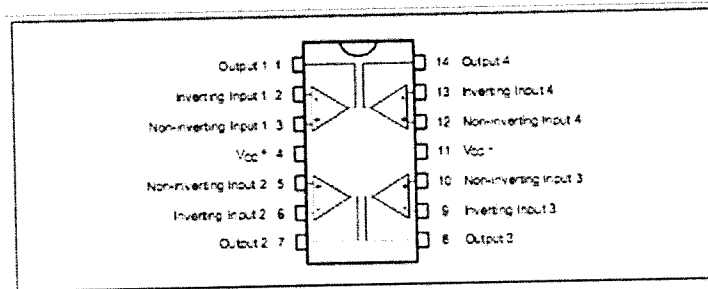


Gambar 13. Kaki sirkuit terintegrasi
(ilmu-elektronika.co.cc)

Pada gambar 13b., kaki pertama selalu ada di sebelah kiri dan kaki terakhir berada pada sebelah kanan tanda setengah lingkaran. Untuk kaki kedua, ketiga, dan seterusnya berada di bawahnya. Sedangkan pada Gambar 13c., kaki pertama berada pada tepat di mana tanda lingkaran berada, yakni di pojok kiri atas dengan posisi IC menghadap ke atas.

3. Klasifikasi Sirkuit Terintegrasi

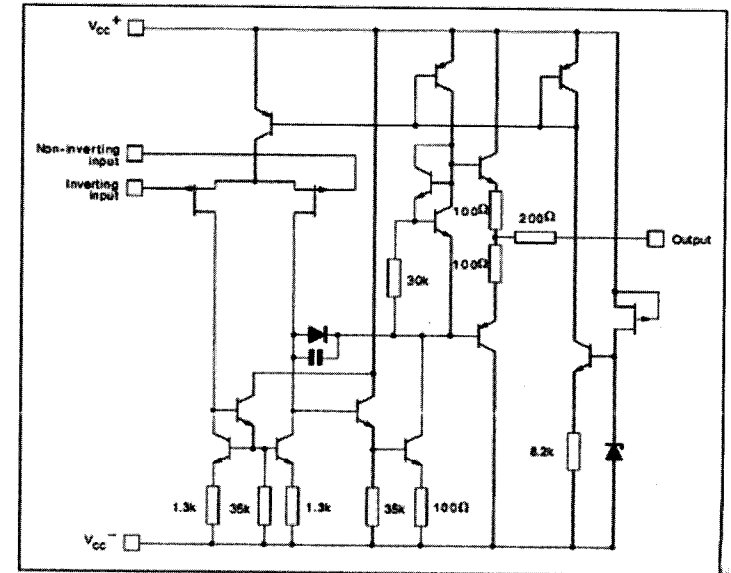
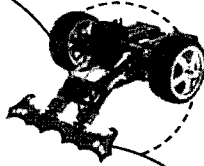
Klasifikasi sirkuit terintegrasi terbagi dalam tiga jenis, yaitu analog, digital, dan campuran (analog dan digital dalam satu IC). Pada IC analog, aplikasinya lebih ditujukan pada pengolahan sinyal-sinyal analog seperti pada *operational amplifier* (Op-Amp), penguat audio, regulator tegangan, dan sebagainya.



Gambar 14. Contoh IC analog
(ilmu-elektronika.co.cc)

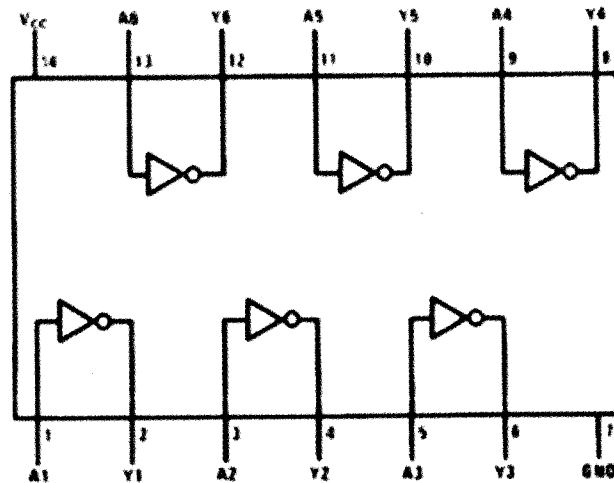
Gambar tersebut merupakan contoh gambar IC TL084 dari *SGS-Thomson Mikroelektronika* yang Op-Amp biasa digunakan pada rangkaian audio. Di dalam sirkuit terintegrasi tersebut, terdapat empat buah simbol Op-Amp (yang berbentuk segitiga) yang masing-masing masukan dan keluarannya terhubung pada satu kaki IC.

Pada IC analog, aplikasinya lebih ditujukan pada pengolahan sinyal-sinyal analog seperti pada *operational amplifier* (Op-Amp), penguat audio, regulator tegangan, dan sebagainya.



Gambar 15. Rangkaian untuk satu Op-Amp
(ilmu-elektronika.co.cc)

Gambar tersebut merupakan gambar rangkaian untuk satu Op-Amp. Jadi, dalam satu sirkuit terintegrasi TL084 terdapat empat buah rangkaian Op-Amp. Jika IC analog diaplikasikan sebagai pengolah sinyal-sinyal analog, maka pada IC digital diperuntukkan untuk mengolah sinyal-sinyal digital, di mana di dalamnya terdapat berbagai macam gerbang logika, flip-flop, *multiplexer*, dan sebagainya.



Gambar 16. Contoh IC digital DM74LS04

(ilmu-elektronika.co.cc)

Gambar tersebut merupakan salah satu contoh IC digital DM74LS04 dari *National Semiconductor* yang di dalamnya terdapat enam gerbang inverter, yang setiap masukan dan keluarannya terhubung pada setiap kaki IC.

Sedangkan klasifikasi terakhir yaitu sirkuit terintegrasi campuran, yang merupakan gabungan rangkaian analog dan digital, sehingga IC mampu mengolah kedua sinyal di dalamnya. Contoh dari jenis sirkuit terintegrasi ini adalah *Analog to Digital Converter* (ADC) yang merupakan penerjemah sinyal analog ke bentuk sinyal digital dan *Digital to*

Analog Converter (DAC), kebalikan dari ADC yang menerjemahksinyal digital ke bentuk analog.

E. Bilangan-Bilangan dalam Elektronika Digital

Dalam kehidupan sehari-hari, kita sudah terbiasa menghitung menggunakan bilangan desimal yang memiliki sepuluh simbol bilangan, yaitu 0, 1, 2, 3, 4, 5, 6, 7, 8, dan 9. Tetapi, tahukah Anda bahwa dalam elektronika digital bilangan yang digunakan adalah sistem bilangan yang tidak populer? Sistem bilangan yang digunakan adalah sistem bilangan biner yang hanya memiliki dua simbol bilangan, yaitu 0 dan 1. Selain itu, pada sistem *microprocessor* dan komputer, sistem bilangan yang digunakan adalah sistem bilangan heksadesimal yang memiliki enam belas simbol bilangan, yaitu 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.

1. Bilangan Biner

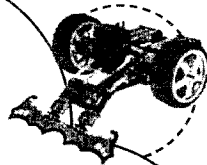
Sistem bilangan biner merupakan sistem bilangan yang memiliki dua simbol bilangan, yaitu 0 dan 1, sehingga sering disebut sebagai sistem bilangan basis 2. Pada tabel berikut diperlihatkan bagaimana pencacahan "0" sampai dengan "9" dalam sistem bilangan desimal dan biner.

Tabel 4. Sistem bilangan desimal dan biner

Sistem Bilangan Desimal	Sistem Bilangan Biner
0	0
1	1
2	10
3	11
4	100
5	101
6	110
7	111
8	1000
9	1001

Dalam pencacahan sistem bilangan biner, untuk menyatakan pencacahan 2 dinotasikan dengan "10" (satu nol), untuk menyatakan pencacahan 3 dinotasikan "11" (satu satu), dan untuk menyatakan pencacahan 9 dinotasikan "1001" (satu nol satu).

Sistem bilangan biner merupakan sistem bilangan yang memiliki dua simbol bilangan, yaitu 0 dan 1, sehingga sering disebut sebagai sistem bilangan basis 2.



2. Nilai Bagian

Dalam sistem bilangan desimal, kita mengenal nilai satuan, puluhan, ratusan, ribuan, dan seterusnya, ini disebut sebagai nilai bagian. Dalam sistem bilangan biner juga memiliki nilai bagian. Untuk lebih jelasnya, perhatikan tabel berikut.

Tabel 5. Nilai bagian untuk sistem bilangan biner

1	1	0	0	1	0	1	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

Nilai bagian untuk sistem bilangan biner dimulai dari kiri ke kanan adalah 2^0 , 2^1 , 2^2 , 2^3 , 2^4 , 2^5 , 2^6 , 2^7 , dan seterusnya.

3. Konversi Bilangan Biner ke Desimal atau Sebaliknya

Bagaimanakah cara mengubah bilangan biner ke desimal atau sebaliknya? Sebagai contoh, jika kita memiliki bilangan biner "1011101", untuk melakukan konversi ke bentuk bilangan desimal perhatikan tabel berikut ini.

Tabel 6. Konversi ke bentuk bilangan desimal

1	0	1	1	1	0	1
2^6	-	2^4	2^3	2^2	-	2^0
64	-	16	8	4	-	1

Carilah nilai bagian untuk masing-masing digit bilangan biner yang bernilai "1" (satu). Berdasarkan Tabel 6, diperoleh angka 64, 16, 8, 4, dan 1. Kemudian, jumlahkanlah nilai bagian tersebut ($64 + 16 + 8 + 4 + 1 = 93$). Jadi, bilangan biner 1011101 sama dengan 93 dalam bilangan biner.

Lalu, bagaimana jika ingin mengubah bilangan desimal ke biner? Sebagai contoh kita akan mengubah bilangan desimal "57" ke bilangan biner. Cara konversinya adalah sebagai berikut:

Tabel 7. Perhitungan konversi ke bentuk bilangan desimal

57	dibagi	2	=	28	sisa	1
28	dibagi	2	=	14	sisa	0
14	dibagi	2	=	7	sisa	0
7	dibagi	2	=	3	sisa	1
3	dibagi	2	=	1	sisa	1
1	dibagi	2	=	0	sisa	1

Berdasarkan Tabel 7, nilai biner diperoleh dari "sisa" hasil pembagian 2. Jadi, bilangan desimal "57" sama dengan "111001" dalam bilangan biner.

4. Bilangan Heksadesimal

Sistem bilangan heksadesimal terdiri dari enam belas simbol, yaitu "0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F". Keenam belas simbol tersebut biasa disebut sebagai sistem bilangan basis 16. Huruf "A, B, C, D, E, F" pada bilangan heksadesimal merupakan perwakilan simbol "10, 11, 12, 13, 14, 15" pada bilangan desimal.

Tabel 8. Sistem bilangan heksadesimal

Desimal	Biner	Heksadesimal
0	0	0
1	1	1
2	10	2
3	11	3
4	100	4
5	101	5
6	110	6
7	111	7
8	1000	8
9	1001	9
10	1010	A

11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F
16	10000	10
17	10001	11
18	10010	12
19	10011	13
20	10100	14

Tabel 8 merupakan perbandingan pencacahan antara tiga jenis sistem bilangan. Jadi, sekarang jika menuliskan bilangan "10" yang merupakan banyaknya objek, maka "10" bisa berarti "sepuluh", "dua", atau "enam belas", tergantung sistem bilangan yang digunakan.

Untuk membedakan sistem bilangan yang digunakan, maka ditambahkan subscript yang menandakan basis bilangan. Untuk desimal ditulis " 10_{10} ", biner ditulis " 10_2 ", dan heksadesimal ditulis " 10_{16} ".

5. Konversi Bilangan Heksadesimal ke Sistem Bilangan Lainnya

Konversi bilangan heksadesimal ke biner dilakukan dengan mengubah masing-masing digit bilangan heksadesimal ke ekuivalen empat bit bilangan biner.

Sebagai contoh:

$$D7_{16} = \dots? \text{ (biner)}$$

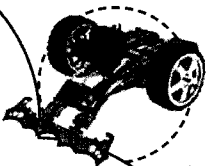
$$D_{16} = 1101_2$$

$$7_{16} = 0111_2$$

$$\text{Jadi, } D7_{16} = 11010111_2.$$

Untuk mengubah bilangan heksadesimal $D7_{16}$ ke bentuk biner, maka terlebih dahulu buat D_{16} menjadi bentuk biner, yaitu 1101_2 . Dengan cara yang sama, dibuat juga bentuk 7_{16} menjadi bentuk biner, yaitu 0111_2 . Jadi, bilangan heksadesimal $D7_{16}$ sama dengan 11010111_2 dalam bilangan biner.

Untuk mengubah bilangan heksadesimal $D7_{16}$ ke bentuk biner, maka terlebih dahulu buat D_{16} menjadi bentuk biner, yaitu 1101_2 . Dengan cara yang sama, dibuat juga bentuk 7_{16} menjadi bentuk biner, yaitu 0111_2 .



Lalu, bagaimana cara mengubah bilangan heksadesimal ke sistem bilangan desimal? Perhatikan contoh berikut, di mana akan dilakukan konversi dari $5AB_{16}$ ke bentuk bilangan desimal.

Tabel 9. Perhitungan bilangan heksadesimal

5	A	B	
\times	\times	\times	
16^2	16^1	16^0	
1280	160	11	= 1451

Pada tabel di atas, setiap bilangan heksadesimal dikalikan nilai bagiannya. Untuk 5_{16} dikalikan 16^2 menghasilkan 1280_2 . A_{16} sama dengan 10_2 dikalikan 16^1 menghasilkan 160_2 . B_{16} sama dengan 11_{10} dikalikan 16^0 menghasilkan 11_{10} . Hasil perkalian setiap bilangan heksadesimal terhadap nilai bagiannya dijumlahkan menghasilkan 1451_{10} . Jadi, $5AB_{16} = 1451_{10}$.

Sekarang bagaimana jika kebalikannya, yaitu mengubah bilangan desimal ke bentuk heksadesimal. Sebagai contoh adalah bilangan desimal 38_{10} akan diubah ke bentuk heksadesimal.

Tabel 10. Perhitungan bilangan heksadesimal

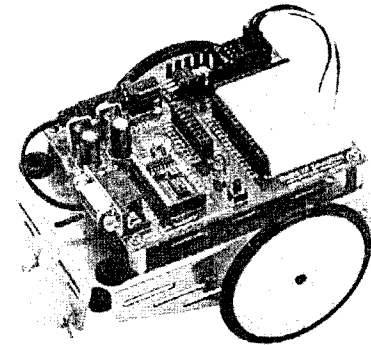
38	dibagi	16	=	2	sisanya	6
2	dibagi	16	=	0	sisanya	2

Caranya sama dengan seperti mengubah bilangan desimal ke biner, tetapi pada konversi bilangan desimal ke heksadesimal ini, bilangan yang ada dibagi dengan 16. Dari Tabel 10 didapat bahwa bilangan desimal 38_{10} sama dengan 26_{16} dalam bilangan heksadesimal.



Bab 3

Membuat Robot Cerdas dengan Boe-Bot



Gambar 1. Tampilan robot cerdas Boe-Bot

(web.njit.edu)

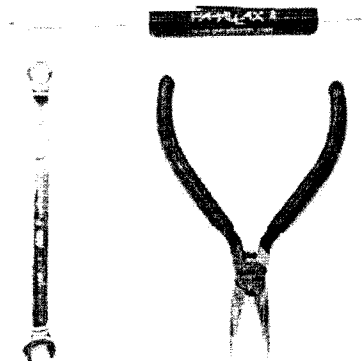
Pada bab ini, dibahas langkah-langkah dalam merakit dan memprogram robot cerdas Boe-Bot. Sangat penting untuk menyelesaikan bab ini, agar selanjutnya dalam membuat program yang berhubungan dengan Boe-Bot secara utuh dapat dilakukan.

A. Merakit dan Mencoba Boe-Bot

Dalam bagian ini, akan saya sertakan gambar-gambar untuk setiap langkahnya, agar memudahkan Anda saat merakit Boe-Bot secara benar. Ikutilah setiap instruksi yang diberikan secara teratur.

1. Perangkat Keras

Semua alat yang ditunjukkan pada Gambar 2 adalah alat-alat yang sering kita temukan dalam kehidupan sehari-hari.



Gambar 2. Peralatan yang dibutuhkan
(robotic-tutorials.blogspot.com)

Adapun komponen elektronika yang dibutuhkan dalam merakit Boe-Bot dapat Anda lihat dalam tabel berikut:

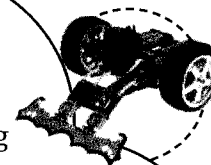
Tabel 1. Komponen-komponen yang dibutuhkan dalam merakit robot

Komponen	Jumlah (Buah)
IC LM 324	1
resistor 33 K Ω	2

resistor 10 K Ω	4
resistor 560 Ω	10
transistor (TR) 9013	8
<i>variable resistor</i> (VR)	2
LED indikator	4
LED superbright	2
photodiode	2
motor 3 volt	2
PCB metrik	1
baterai	1
saklar toggle	1

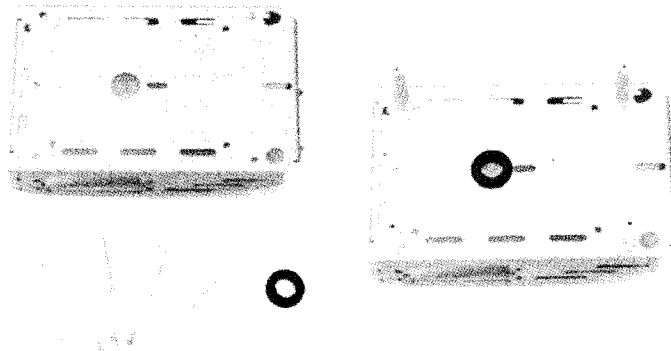
Perangkat keras yang dibutuhkan untuk memasang kerangka Boe-Bot (chasis Boe-Bot) adalah (4) penopang chasis 1"; (4) sekrup, 1A" 4-40; dan karet berbentuk bulat, 73/SZ.

Perangkat keras yang dibutuhkan untuk memasang kerangka Boe-Bot (chasis Boe-Bot) adalah (4) penopang chasis 1"; (4) sekrup, 1A" 4-40; dan karet berbentuk bulat, 73/SZ.



Langkah-langkah pemasangan:

- Masukkan karet (pada gambar berwarna hitam), ke dalam lubang pada posisi tengah chasis Boe-Bot.
- Pastikan karet telah diletakkan dengan benar pada chasis.
- Gunakan 4 sekrup 1A" 4-40 untuk menempel penopang chasis, seperti ditunjukkan pada Gambar 3.



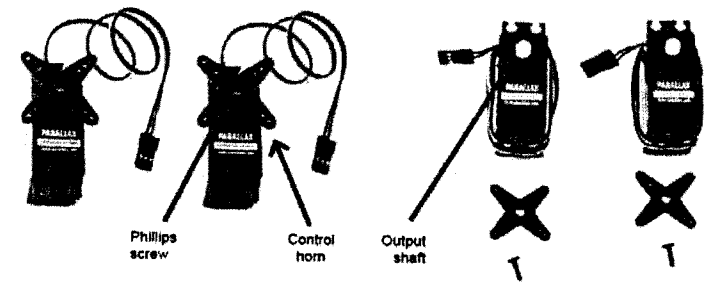
Gambar 3. Perangkat keras yang diperlukan (kiri) dan setelah dipasang (kanan)
(robotic-tutorials.blogspot.com)

2. Melepas Horn Pada Servo

Untuk melepas horn pada servo, matikan power dari BASIC Stamp dan servo Anda. Lepaskan semua baterai AA yang terpasang pada baterai pack serta matikan koneksi servo pada board.

Langkah-langkah pemasangan:

- Gunakan obeng (*screwdriver*) untuk memindahkan sekrup yang terpasang pada horn.
- Pastikan Anda menyimpan sekrup dengan baik, karena akan digunakan pada langkah berikutnya.



Gambar 4. Kontrol horn servo (kiri) dan setelah dipindahkan (kanan)
(parallax.com)

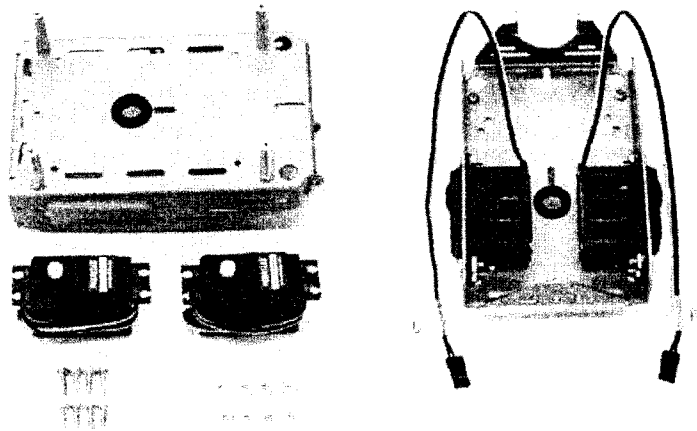
3. Memasang Servo ke Chasis

Daftar perangkat yang dibutuhkan untuk memasang servo ke chasis adalah chasis Boe Bot (yang telah dirakit); servo putar Parallax; (8) sekrup, 3/8 4-40; dan (8) nuts(ring), 4-40.

Langkah-langkah pemasangan:

- Pasang servo ke chasis dengan sekrup dan nuts(ring) yang telah tersedia.

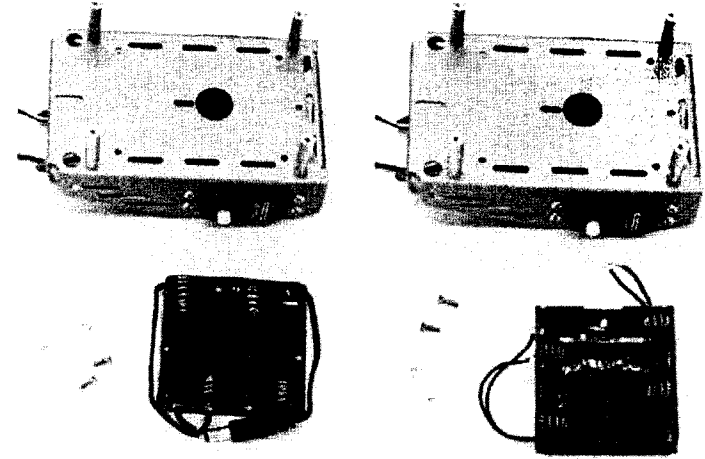
- b. Posisi servo pada chasis harus tepat, hingga masuk ke lubang berbentuk persegi seperti pada Gambar 5.
- c. Gunakan label untuk menandakan servo kiri dan servo kanan yang telah dipasang.



Gambar 5. Pemasangan servo pada chasis (kiri) dan setelah perakitan (kanan)
(parallax.com)

4. Pemasangan Battery Pack Pada Chasis

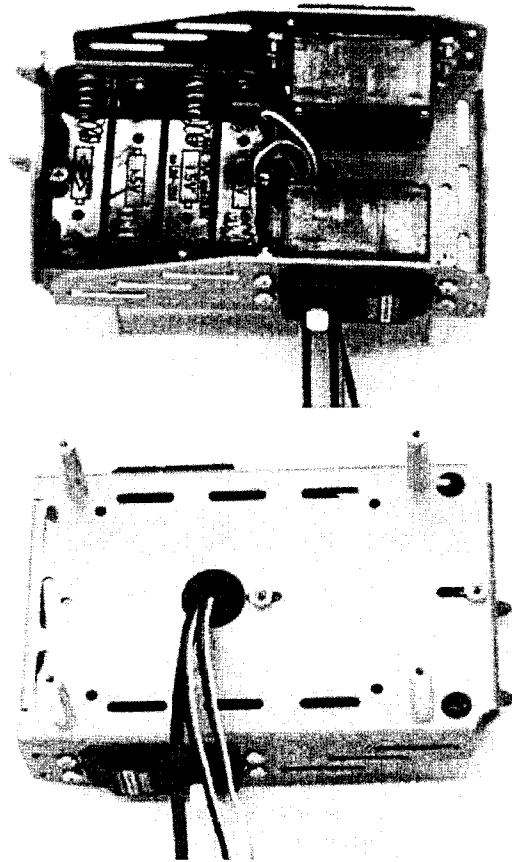
Gambar 6. merupakan dua perangkat yang berbeda. Gunakan perangkat pada sisi kiri jika Anda memiliki *Board of Education* dan sisi kanan jika menggunakan *Homework Board*.



Gambar 6. Tempat baterai untuk Boe-Bot
(parallax.com)

Langkah-langkah pemasangan:

- a. Gunakan sekrup dan nuts(ring) untuk memasang *battery pack* ke dalam chasis Boe-Bot.
- b. Pastikan Anda telah memasang sekrup dan ring dengan benar.
- c. Seperti ditunjukkan pada sisi kanan Gambar 7, tarik kabel power hingga melewati lubang pada tengah chasis.
- d. Tarik pula kabel yang ada pada servo.

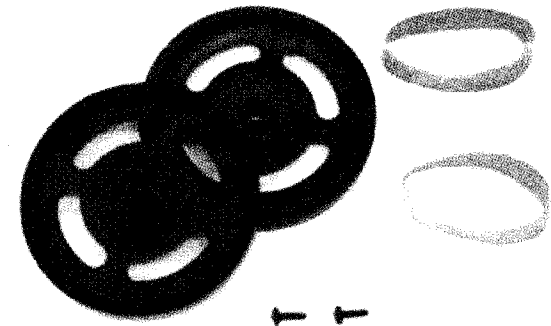


Gambar 7. *Battery Pack* yang terpasang tampak bawah (kiri) dan atas (kanan)
(parallax.com)

5. Pemasangan Roda Pada Boe-Bot

Perangkat yang dibutuhkan untuk roda Boe-Bot adalah (1) 7/76" *cotter pin* (seperti jarum; lihat Gambar 8), roda belakang berbentuk bulat (putih),

roda depan (warna hitam), *plastic machined wheels* (plastik pelapis roda), dan (2) sekrup yang dipasang pada bagian tengah roda belakang. Roda seperti ini dapat Anda beli secara terpisah, termasuk karet ban-talannya.

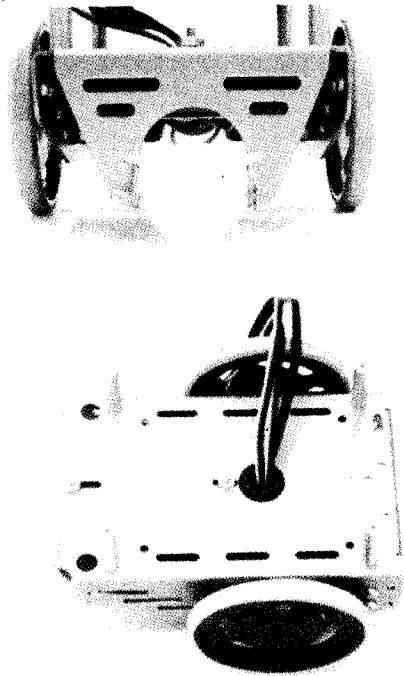


Gambar 8. Perangkat roda
(parallax.com)

Langkah-langkah pemasangan:

- Gambar 9 (kiri) menunjukkan roda belakang Boe-Bot yang telah terpasang pada chasis.
- Masukkan *cotter pin* pada roda belakang, melewati sisi kiri chasis, roda, dan sisi kanan chasis.
- Pastikan *cotter pin* telah terpasang dengan benar.
- Masukkan karet pelapis roda, masing-masing di sisi luar roda depan.

- e. Gunakan sekrup yang tadi digunakan untuk memasang horn pada servo dan pasang ke roda bagian depan.

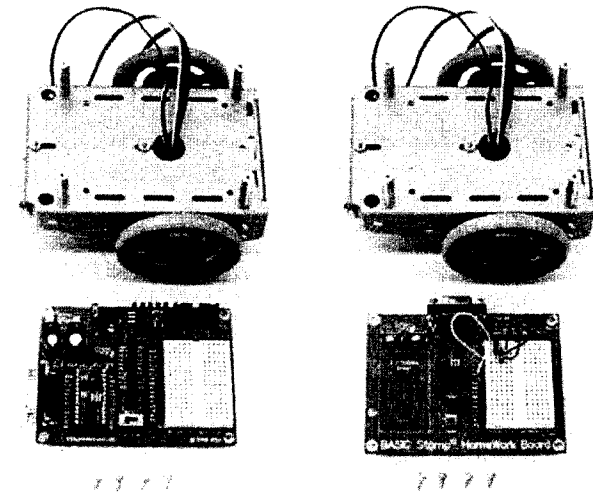


Gambar 9. Pemasangan roda belakang (kiri) dan roda depan/ penggerak (kanan)
(parallax.com)

6. Pemasangan Board ke Chasis Boe-Bot

Perhatikan posisi kabel pada Gambar 10 sebelum memasang board. Komponen-komponen yang dibu-

tuhkan saat memasang board adalah (1) chasis Boe-Bot, (4) sekrup, 7/4" 4-40, dan board BASIC stamp 2.



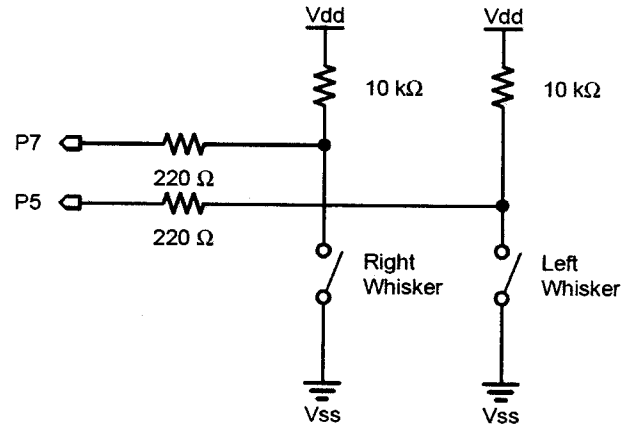
Gambar 10. Chasis Boe Bot dan board
(parallax.com)

Board dipasang di atas 4 buah sekrup tersebut, lalu dipasang baut untuk memperkuat posisi board tersebut.

B. Robot Whisker

Tibalah saatnya kita akan memprogram robot cerdas pertama, yaitu robot yang mampu mendeteksi penghalang. Jika robot menabrak penghalang, maka robot

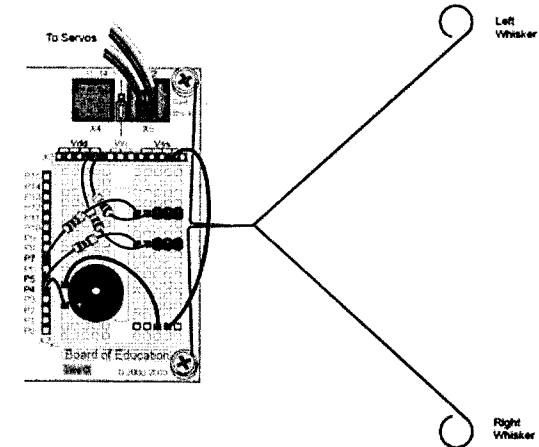
tersebut diprogram untuk mencari jalan lain yang bisa dilewati. Pendeteksian dilakukan menggunakan sungut (mirip sensor serangga seperti sensor pada semut).



Gambar 11. Skema whisker
(parallax.com)

1. Cara Mendeteksi Fungsi Whisker

Sebelumnya, pasanglah sungut pada robot, seperti rangkaian pada Gambar 12, kemudian gunakan sebuah program untuk mendeteksi apakah whisker telah berfungsi.



Gambar 12. Hasil pemasangan sungut
(parallax.com)

Program di bawah ini untuk menguji apakah whisker sudah berfungsi dengan benar, yaitu dengan menampilkan bit-bit yang dimasukkan pada register input P7 dan P5 (IN7 dan IN15). Setelah dijalankan, program akan memberikan *feedback* apakah robot dapat mendeteksi whisker.

Langkah-langkah pendeteksian:

- Hidupkan *power* pada board dan servo.
- Masukkan, simpan, dan jalankan TestWhiskers.bs2.
- Hogram menggunakan Debug Terminal. Jadi, pastikan kabel serial/USB tetap tersambung pada BASIC Stamp ketika program running.

```

TestWhiskers.bs2
` Robotics with the Boe-Bot - TestWhiskers.bs2
` Display what the I/O pins connected to the
whiskers sense.
` {$STAMP BS2}           ` Stamp directive.
` {$PBASIC 2.5}          ` PBASIC directive.
DEBUG "Kiri Kanan", CR,
"Left Right", CR,
"-----"
DO
DEBUG CR$RXY, 0, 3,
"P5 = ", BIN1 IN5,
"P7 = ", BIN1 IN7
PAUSE 50
Loop

```

- d. Jika nilai P7 dan P5 pada Debug Terminal menunjukkan angka 1, berarti whisker telah tersambung dengan benar.

2. Cara Memasang LED

LED yang kita pasang berfungsi untuk menampilkan status jika robot menabrak objek. Agar dapat melakukan aksinya saat salah satu sungut berbenturan dengan objek, sisipkan kondisi IF...THEN di antara perintah PAUSE 50 dan LOOP pada program tes whisker. Kemudian, beri nama tes LED.bs2, lalu dijalankan.

```

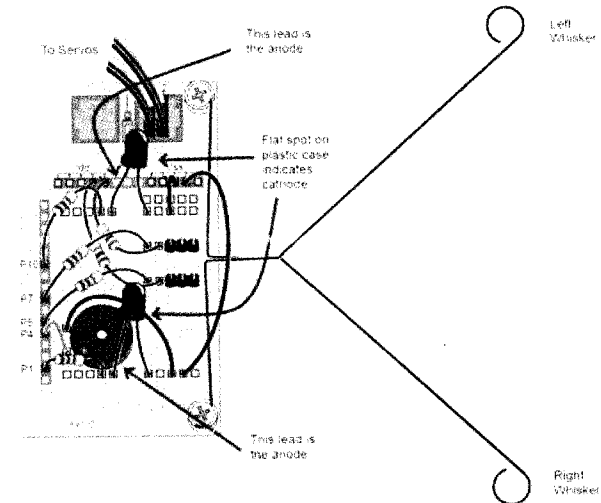
IF (IN7 = 0) THEN
HIGH 1
ELSE
LOW 1
ENDIF

```

```

IF (IN5 = 0) THEN
HIGH 10
ELSE
LOW 10
ENDIF

```



Gambar 13. Hasil pemasangan LED
(parallax.com)

Program berikut akan menginstruksikan pada Boe-Bot agar jika ada rintangan di depannya, ia berbalik dan berpindah arah dengan menggunakan subrutin IF...THEN. Setelah program dijalankan, biarkan Boe-Bot berjalan. Jika terjadi tabrakan, Boe-Bot akan berbalik arah.

```

' -----[ Title ]-----
' Robotics with the Boe-Bot - RoamingWithWhiskers.
bs2
' Boe-Bot uses whiskers to detect objects, and
navigates around them.
' {$STAMP BS2}           ' Stamp directive.
' {$PBASIC 2.5}          ' PBASIC directive.
DEBUG "Program Running!"
' -----[ Variables ]-----
pulseCount VAR Byte      ' FOR...NEXT loop counter.
' -----[ Initialization ]-----
FREQOUT 4, 2000, 3000    ' Signal program start/
reset.
' -----[ Main Routine ]-----
DO
IF (IN5 = 0) AND (IN7 = 0) THEN ' Both whiskers
detect obstacle
GOSUB Back_Up              ' Back up & U-turn (left
twice)
GOSUB Turn_Left
GOSUB Turn_Left
ELSEIF (IN5 = 0) THEN      ' Left whisker contacts
GOSUB Back_Up             ' Back up & turn right
GOSUB Turn_Right
ELSEIF (IN7 = 0) THEN      ' Right whisker contacts
GOSUB Back_Up             ' Back up & turn left
GOSUB Turn_Left
ELSE                       ' Both whiskers 1, no
contacts
GOSUB Forward_Pulse        ' Apply a forward pulse
ENDIF                     ' and check again
LOOP
' -----[ Subroutines ]-----
Forward_Pulse:             ' Send a single forward
pulse.
PULSOUT 13,850
PULSOUT 12,650
PAUSE 20
RETURN
Turn_Left:                 ' Left turn, about
90-degrees.
FOR pulseCount = 0 TO 20
PULSOUT 13, 650
PULSOUT 12, 650

```

C. Kecerdasan Buatan dan Pengambilan Keputusan

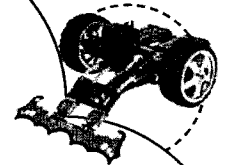
Ketika Boe-Bot bergerak, mungkin ia akan berhenti di pojok ruangan, sehingga tidak bisa bergerak. Artinya, ketika kondisi ini terjadi, whisker pada Boe-Bot akan menyentuh dinding sisi kiri dan akan bergerak ke kanan. Setelah maju ke depan, whisker kanan akan menyentuh dinding lagi, dan begitu seterusnya, sehingga Boe-Bot tidak bisa berpindah.

1. Melepaskan Boe-Bot dari Kondisi Pojok

Program sebelumnya dapat dimodifikasi untuk mendeteksi masalah dan bergerak menghindarinya. Trik yang digunakan adalah dengan menghitung selang jumlah waktu dari whisker dalam mendeteksi gangguan (kontak dengan gangguan).

Program berikut akan menggunakan IF...THEN bersarang. Dengan kata lain, program akan mengecek satu kondisi, jika benar, akan pindah ke kondisi lain

Ketika Boe-Bot bergerak, mungkin ia akan berhenti di pojok ruangan, sehingga tidak bisa bergerak. Artinya, ketika kondisi ini terjadi, whisker pada Boe-Bot akan menyentuh dinding sisi kiri dan akan bergerak ke kanan.



(masih dalam kondisi yang pertama). Untuk lebih jelasnya, berikut Pseudocode dari contoh yang akan digunakan:

```

IF (IN5 = 0) AND (IN7 = 0) THEN
  GOSUB Back_Up          ' Both whiskers detect
                           obstacle,
  GOSUB Turn_Left        ' back up & U-turn (left twice)
  GOSUB Turn_Left
  ELSEIF (IN5 = 0) THEN   ' Left whisker contacts
    GOSUB Back_Up        ' Back up & turn right
    GOSUB Turn_Right
  ELSEIF (IN7 = 0) THEN   ' Right whisker contacts
    GOSUB Back_Up        ' Back up & turn left
    GOSUB Turn_Left
  ELSE
    contacts              ' Both whiskers 1, no
                           contacts
    GOSUB Forward_Pulse   ' Apply a forward pulse &
                           check again
  ENDIF

```

Jalankan program Escapingcomers.bs2 dengan cara memasukkan, simpan, dan run EscapingComers.bs2. Kemudian, tes program dengan menekan sisi whisker secara bergantian (sebagai percobaan rintangan) untuk mengetes bagaimana cara Boe-Bot melepaskan diri dari rintangan.

```

' -----[ Title ]-----
' Robotics with the Boe-Bot - EscapingCorners.bs2
' Boe-Bot navigates out of corners by detecting
  alternating whisker presses.
' {$STAMP BS2}          ' Stamp directive.
' {$PBASIC 2.5}         ' PBASIC directive.
DEBUG "Program Running!"
' -----[ Variables ]-----
pulseCount VAR Byte     ' FOR...NEXT loop counter.
counter VAR Nib         ' Counts alternate
contacts.

```

```

old7 VAR Bit            ' Stores previous IN7.
old5 VAR Bit            ' Stores previous IN5.
' -----[ Initialization ]-----
FREQOUT 4, 2000, 3000   ' Signal program start/
reset.
counter = 1              ' Start alternate corner
count.
old7 = 0                 ' Make up old values.
old5 = 1
' -----[ Main Routine ]-----
DO
' --- Detect Consecutive Alternate Corners ---
' See the "How EscapingCorners.bs2 Works" section
  that follows this program.
  IF (IN7 <> IN5) THEN     ' One or other is pressed.
  IF (old7 <> IN7) AND (old5 <> IN5) THEN ' Different
    from previous.
    counter = counter + 1 ' Alternate whisker count
    + 1.
    old7 = IN7            ' Record this whisker
    press
    old5 = IN5            ' for next comparison.
    IF (counter > 4) THEN  ' If alternate whisker
      count = 4,
      counter = 1         ' reset whisker counter
      GOSUB Back_Up       ' and execute a U-turn.
      GOSUB Turn_Left
      GOSUB Turn_Left
    ENDIF                 ' ENDIF counter > 4.
  ELSE                    ' ELSE (old7=IN7) or
    (old5=IN5),
    counter = 1           ' not alternate, reset
    counter.
  ENDIF                   ' ENDIF (old7<>IN7) and '
    (old5<>IN5).
  ENDIF                   ' ENDIF (IN7<>IN5).
' --- Same navigation routine from
  RoamingWithWhiskers.bs2 -----
  IF (IN5 = 0) AND (IN7 = 0) THEN ' Both whiskers
    detect obstacle
    GOSUB Back_Up         ' Back up & U-turn
    (left twice)
    GOSUB Turn_Left
    GOSUB Turn_Left
  ELSEIF (IN5 = 0) THEN   ' Left whisker
    contacts

```

```

GOSUB Back_Up           ' Back up & turn right
GOSUB Turn_Right
ELSEIF (IN7 = 0) THEN   ' Right whisker contacts
GOSUB Back_Up           ' Back up & turn left
GOSUB Turn_Left
ELSE                     ' Both whiskers 1, no
contacts
GOSUB Forward_Pulse     ' Apply a forward pulse
ENDIF                   ' and check again
LOOP
' -----[ Subroutines ]-----
Forward_Pulse:          ' Send a single forward
pulse.
PULSOUT 13,850
PULSOUT 12,650
PAUSE 20
RETURN
Turn_Left:              ' Left turn, about
90-degrees.
FOR pulseCount = 0 TO 20
PULSOUT 13, 650
PULSOUT 12, 650
PAUSE 20
NEXT
RETURN
Turn_Right:
FOR pulseCount = 0 TO 20 ' Right turn, about
90-degrees.
PULSOUT 13, 850
PULSOUT 12, 850
PAUSE 20
NEXT
RETURN
Back_Up:                ' Back up.
FOR pulseCount = 0 TO 40
PULSOUT 13, 650
PULSOUT 12, 850
PAUSE 20
NEXT
RETURN

```

2. Cara Kerja Program Escaping Corners.bs2

Karena program di atas adalah hasil modifikasi program sebelumnya, maka berikut hanya akan

dijelaskan tag-tag tambahan saja. Dalam mendeteksi pojok dinding (*corner*), ada tiga variabel tambahan yang dibuat. Variabel *nibble counter* dapat memasukkan nilai antara 0 hingga 15. Karena nilai target kita dalam mendeteksi *corner* adalah 4, maka ukuran *variabel counter* dapat memenuhinya.

Selain variabel *corner*, digunakan juga variabel bit, yaitu *old7* dan *old5*, serta untuk menampung nilai sebelumnya pada *IN7* dan *IN5* yang juga termasuk variabel bit.

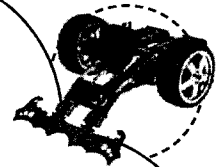
```

counter VAR Nib
old7 VAR Bit
old5 VAR Bit

```

Variabel-variabel di atas harus diberikan nilai awal. Variabel *nibble counter* dimulai dengan angka 1. Jika nilainya 4, yang berarti Boe-Bot *stuck* di pojok (*corner*), nilai ini akan di-*reset* lagi menjadi 1. Nilai variabel *old7* dan *old5* bergantung jumlah penekanan pada salah satu whisker sebelum program dijalankan. Hal ini perlu dilakukan karena adanya gangguan, sehingga harus dideteksi dengan membandingkan apakah "*IN5* = 1 dan *IN7* = 0" atau "*IN5* = 0 dan *IN7* = 1"? Sama halnya dengan *old5* dan *old7*, di mana nilai keduanya harus berbeda.

Dalam mendeteksi pojok dinding (*corner*), ada tiga variabel tambahan yang dibuat, yaitu variabel *nibble counter*, variabel *corner*, dan variabel bit.



```
counter = 1
old7 = 0
old5 = 1
```

Sekarang, untuk mendeteksi pojok (*corner*) secara bergantian, yang harus dicek pertama kali adalah whisker mana yang tertekan (bertubrukan). Untuk mendeteksi hal ini, kondisi yang harus ada adalah apakah nilai variabel IN7 berbeda dengan nilai IN5? Pada PBASIC, dapat digunakan notasi "<>" pada kondisi IF, seperti berikut:

```
IF (IN7 <> IN5) THEN
```

Jika benar salah satu whisker tertekan, selanjutnya kondisi yang harus dicek adalah apakah bentuk halangan/rintangan yang timbul sama seperti sebelumnya atau tidak. Dengan kata lain, apakah (old7 <> IN7) dan (old5 <> IN5)? Jika ya, nilai *counter* akan bertambah secara berurutan. Kemudian, set nilai old7 serta old5 agar sama dengan IN7 dan IN5.

```
IF (old7 <> IN7) AND (old5 <> IN5) THEN
counter = counter + 1
old7 = IN7
old5 = IN5
```

Jika kontak whisker (tubrukan) telah terjadi 4 kali secara berurutan, sesuai kondisi yang telah kita definisikan tadi, nilai *counter* akan di-set menjadi 1 (lagi) dan mengeksekusi gerak balik (U) untuk melepaskan diri dari rintangan pojok (*corner*).

```
IF (counter > 4) THEN
counter = 1
GOSUB Back_Up
GOSUB Turn_Left
GOSUB Turn_Left
This ENDIF ends the code block that is executed if
counter > 4.
ENDIF
```

D. Navigasi dengan Inframerah

Seperti yang kita ketahui, komunikasi wireless telah banyak digunakan oleh pengguna komputer untuk mentransfer data dan hal lain, misalnya pada *remote control* dan PDA (*Personal Digital*

Assistant) yang menggunakan frekuensi inframerah dalam komunikasi. Dengan beberapa alat yang tersedia pada paket Boe-Bot, BASIC stamp juga bisa menerima dan mentransmisikan sinyal inframerah.

Inframerah adalah cahaya atau radiasi elektromagnetik dengan frekuensi yang rendah. Salah satu penggunaan cahaya inframerah adalah untuk menyinari jalur robot dan memberikan tanda jika ada objek tertentu di jalur gerak robot.

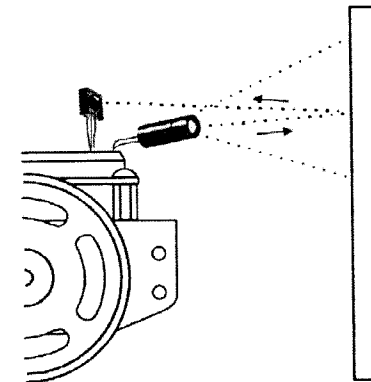


Beberapa robot menggunakan RADAR atau SONAR (terkadang juga disebut SODAR ketika digunakan di udara maupun air). Salah satu penggunaan cahaya inframerah adalah untuk menyinari jalur robot dan memberikan tanda jika ada objek tertentu di jalur gerak robot.

Infra berarti bawah/rendah. Jadi, inframerah adalah cahaya atau radiasi elektromagnetik dengan frekuensi yang rendah. Tabel berikut menunjukkan panjang gelombang beberapa warna dalam spektrum inframerah yang dihitung dalam satuan nanometer (nm).

Tabel 2. Warna dan panjang gelombang

Warna	Panjang Gelombang	Warna	Panjang Gelombang
violet	400	merah	780
biru	470	hampir merah	800–1.000
hijau	565	infrared	1.000–2.000
kuning	590	far infrared	2.000–10.000
oranye	630		

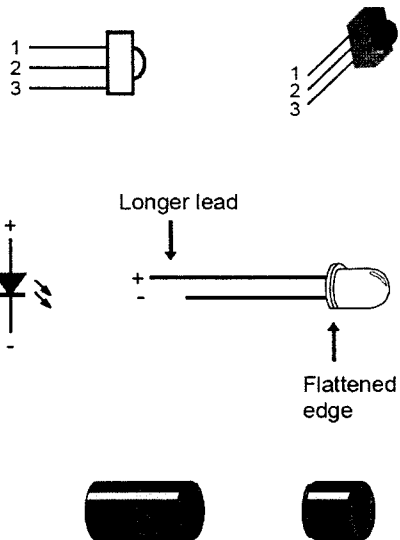


Gambar 14. Deteksi objek dengan IR *headlight*
(parallax.com)

1. Pemasangan dan Tes IR

Sekarang kita akan memasang perangkat yang dibutuhkan dan tes *transmitter* atau detektor dari inframerah. Dalam hal ini, perangkat-perangkat yang dibutuhkan adalah:

- detektor inframerah (2 buah),
- tR LED (2 buah),
- pelapis IR LED (2 buah),
- resistor 220 Ω (merah-merah-cokelat) (2 buah), dan
- resistor 1 k Ω (cokelat-hitam-merah) (2 buah).

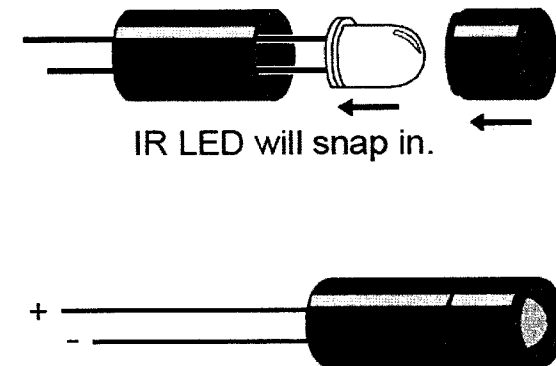


Gambar 15. detector IR (atas), LED IR (tengah), dan pelapis IR LED (bawah)
(parallax.com)

2. Pemasangan Headlight IR

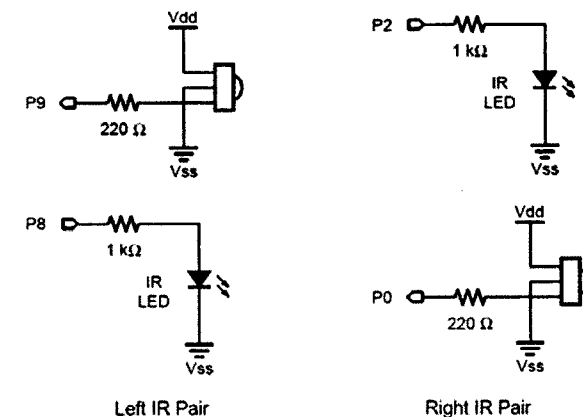
Untuk memasang headlight IR, ikuti langkah-langkah berikut:

- Masukkan LED inframerah ke dalam pelapisnya.
- Sesuaikan pemasangan seperti gambar berikut.



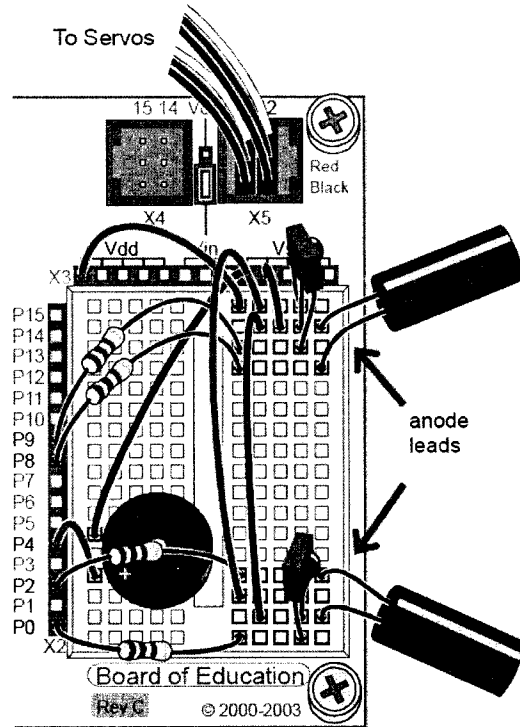
Gambar 16. Pemasangan selubung sensor
(parallax.com)

- Matikan *power* dari board dan servo.
- Buat sirkuit seperti skema berikut:



Gambar 17. Pasangan IR sebelah kanan dan kiri
(parallax.com)

- e. Setelah pemasangan selesai, board/stamp akan berbentuk seperti gambar berikut:

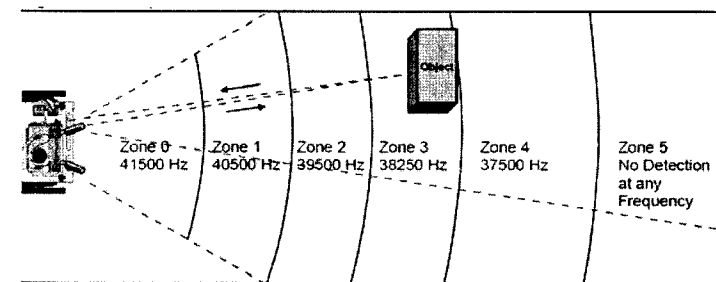


Gambar 18. Gambar board/stamp setelah pemasangan
(parallax.com)

Pada langkah selanjutnya, kita akan membuat program untuk menentukan/deteksi jarak dari suatu objek. Oleh karena itu, pastikan pemasangan IR pada board telah terpasang dengan benar.

E. Kendalikan Robot dengan Deteksi Jarak

Boe-Bot yang mampu menentukan jarak suatu objek, juga dapat diprogram untuk memindahkan objek tanpa bertabrakan dengannya. Dari sini, juga dapat dikembangkan Une Follower Boe-Bot. Jika sirkuit masih terpasang seperti pelajaran sebelumnya, pastikan LED IR telah memiliki resistor 1 k Ω . Alat-alat yang digunakan dalam tes deteksi jarak pada Boe-Bot adalah satu buah penggaris dan beberapa lembar kertas.



Gambar 19. Frekuensi dan zona Boe-Bot
(parallax.com)

Untuk mengetes setiap frekuensi detektor IR, perintah yang digunakan adalah FRESOOT, yang akan mengirim 5 tipe frekuensi berbeda. Kemudian, tes setiap frekuensinya untuk menentukan apakah detektor IR dapat melihat suatu objek di depannya atau tidak. Langkah setiap frekuensi tidak cukup menggunakan operator loop FOR...NH\$. Kita bisa

saja langsung menggunakan 5 perintah yang berbeda dalam mendefinisikan FREOOUT, tetapi akan terjadi pemborosan *space* (ruang) kode. Pendekatan terbaik dalam memasukkan daftar nilai yang digunakan secara berurutan adalah menggunakan perintah LOOKQP.

```
LookUP rindex, [Ni7ai), Ni7ai7, ..NilaiN], Variabel
```

Jika index adalah 0, maka nilai 0 pada kurung akan ditempatkan sebagai variabel. Jika index adalah 1, maka nilai 1 akan ditempatkan sebagai variabel. Nilai bisa didefinisikan hingga 256 nilai pada daftar. Pada contoh program berikut kita hanya akan menggunakan 5 nilai saja.

```
FOR freqselect = 0 TO 4
LOOKUP f regselect, [ 37 500, 38250, 3 9500, 40 50
0, 415 00 ], irFrequency
FREQOUT 8, 1, irFrequency
irDeEect = IN9
' masukkan kode di sini
NEXT
```

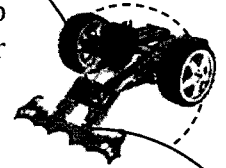
Pada loop pertama FOR...NEXT, nilai freqSelect adalah 0. Jadi, perintah LOOK(P akan menempatkan nilai 37500 pada variabel irFrequency. Karena irFrequency berisi 37500 setelah perintah LOOK(P, maka perintah FRESOOT akan mengirim frekuensi ke LED IR yang tersambung pada P8. Nilai IN9 kemudian akan tersimpan pada variabel irDetect.

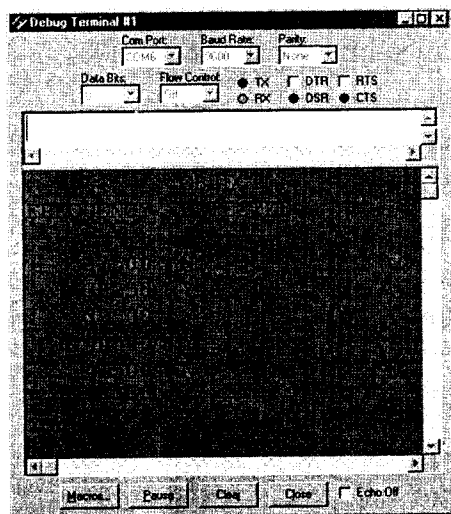
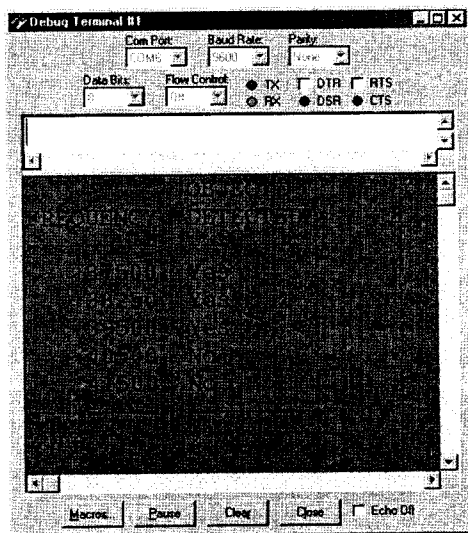
Begitu seterusnya hingga loop selesai dilakukan dengan nilai irFrequency yang terus berubah.

Program Test Left Frequency Sweep.bs2 akan melakukan dua hal. *Pertama*, menguji LED IR yang tersambung pada pg dan p9 untuk meyakinkan apakah keduanya berfungsi dengan benar. *Kedua*, menampilkan proses deteksi frekuensi dan keterangannya seperti yang ditunjukkan pada Gambar 20.

Ketika program di-run, debug terminal akan menampilkan proses pengukuran zona frekuensi. Akan ada banyak kemungkinan yang timbul dari hasil pengukuran ini. Hal tersebut sangat tergantung pada karakteristik filter IR detektor yang digunakan.

Untuk mengetes setiap frekuensi detektor IR, perintah yang digunakan adalah FRESOOT, yang akan mengirim 5 tipe frekuensi berbeda. Kemudian, tes setiap frekuensinya untuk menentukan apakah detektor IR dapat melihat suatu objek di depannya atau tidak.





Gambar 20. Hasil tes IR
(parallax.com)

Langkah selanjutnya adalah masukkan, simpan, dan jalankan program `TestLeftFrequencySweep.bs2`. Gunakan selembar kertas untuk menutupi LED IR dan mendeteksi jarak.

```
' -----[ Title ]-----
' Robotics with the Boe-Bot -
TestLeftFrequencySweep.bs2
' Test IR detector distance responses to frequency
sweep.
' {$STAMP BS2}      ' Stamp directive.
' {$PBASIC 2.5}    ' PBASIC directive.
Page 274 · Robotics with the Boe-Bot
' -----[ Variables ]-----
freqSelect VAR Nib
irFrequency VAR Word
irDetect VAR Bit
distance VAR Nib
' -----[ Initialization ]-----
DEBUG CLS,
" OBJECT", CR,
"FREQUENCY DETECTED", CR,
"-----"
' -----[ Main Routine ]-----
DO
distance = 0
FOR freqSelect = 0 TO 4
LOOKUP freqSelect, [37500,38250,39500,40500,41500],
irFrequency
FREQOUT 8,1, irFrequency
irDetect = IN9
distance = distance + irDetect
DEBUG CRSRXY, 4, (freqSelect + 3), DEC5 irFrequency
DEBUG CRSRXY, 11, freqSelect + 3
IF (irDetect = 0) THEN DEBUG "Yes" ELSE DEBUG "No "
PAUSE 100
NEXT
DEBUG CR,
"-----", CR,
"Zone ", DEC1 distance
LOOP
```

Selanjutnya, akan saya berikan cara menampilkan deteksi jarak dua detector yang ada pada Boe-Bot. Berikut program yang digunakan untuk menampilkannya. Masukkan program dan coba lihat hasilnya.

```

\ -----[ Title ]-----
\ Robotics with the Boe-Bot - DisplayBothDistances.
bs2
\ Test IR detector distance responses of both IR
LED/detector pairs to
\ frequency sweep.
\ {$STAMP BS2}      \ Stamp directive.
\ {$PBASIC 2.5}     \ PBASIC directive.
\ -----[ Variables ]-----
freqSelect          VAR    Nib
irFrequency          VAR    Word
irDetectLeft        VAR    Bit
irDetectRight       VAR    Bit
distanceLeft         VAR    Nib
distanceRight        VAR    Nib
\ -----[ Initialization ]-----
DEBUG CLS,
    "IR OBJECT ZONE", CR,
    "Left Right", CR,
    "-----"
\ -----[ Main Routine ]-----
DO
GOSUB Get_Distances
GOSUB Display_Distances
LOOP
\ -----[ Subroutine - Get_Distances ]-----
Get_Distances:
distanceLeft = 0
distanceRight = 0
FOR freqSelect = 0 TO 4
LOOKUP freqSelect, [37500,38250,39500,40500,41500],
irFrequency
FREQOUT 8,1,irFrequency
irDetectLeft = IN9
distanceLeft = distanceLeft + irDetectLeft
FREQOUT 2,1,irFrequency
irDetectRight = IN0

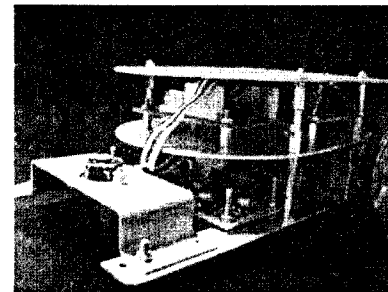
```

```

distanceRight = distanceRight + irDetectRight
PAUSE 100
NEXT
RETURN
\ -----[ Subroutine - Display_Distances ]-----
Display_Distances:
DEBUG      CRSRXY,2,3, DEC1 distanceLeft,
          CRSRXY,9,3, DEC1 distanceRight
RETURN

```

Membuat Sendiri Robot Line Tracker



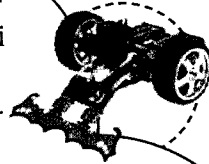
Gambar 1. Robot *line tracker*
(images.myfilehost.us)

Pada bab ini kita akan membahas cara membuat robot *line tracker* yang dapat bergerak mengikuti *track* berupa garis hitam setebal 3 cm. Garis hitam tersebut disusun membentuk sejumlah persimpangan. Robot diprogram untuk dapat menghitung jumlah persimpangan yang sudah dilaluinya, kemudian belok sesuai dengan arah yang diinginkan. Untuk membaca garis, robot dilengkapi dengan sensor proximity yang dapat membedakan antara garis hitam dengan lantai putih. Sensor proximity ini dapat dikalibrasi untuk menyesuaikan pembacaan sensor terhadap kondisi pencahayaan ruangan, sehingga pembacaan

sensor selalu akurat.

Agar pergerakan robot menjadi lebih halus, maka kecepatan robot diatur sesuai kondisi pembacaan sensor *proximity*. Jika posisi robot menyimpang dari garis, maka robot akan melambat. Namun, jika robot tepat berada di atas garis, maka robot akan bergerak cepat. Robot juga dapat kembali ke garis pada saat benar-benar terlepas dari garis. Hal ini bisa dilakukan, karena robot selalu mengingat kondisi terakhir pembacaan sensor. Jika kondisi terakhir adalah di sebelah kiri garis, maka robot akan bergerak ke kanan, demikian pula sebaliknya.

Agar pergerakan robot menjadi lebih halus, maka kecepatan robot diatur sesuai kondisi pembacaan sensor *proximity*. Jika posisi robot menyimpang dari garis, maka robot akan melambat. Namun, jika robot tepat berada di atas garis, maka robot akan bergerak cepat.

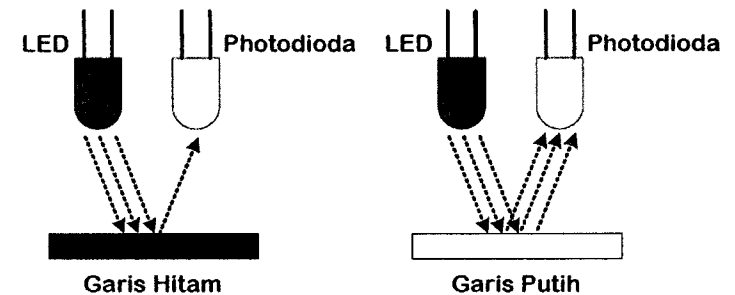


A. Robot Line Tracker

1. Sensor Proximity

Sensor *proximity* bisa kita buat sendiri. Prinsip kerjanya pun sederhana, yaitu hanya dengan memanfaatkan sifat cahaya yang akan dipantulkan jika

mengenai benda berwarna terang dan akan diserap jika mengenai benda berwarna gelap. Sebagai sumber cahaya, kita gunakan LED (*Light Emitting Diode*) yang akan memancarkan cahaya merah. Dan, untuk menangkap pantulan cahaya LED, kita gunakan *photodiode*. Jika sensor berada di atas garis hitam, maka *photodiode* akan menerima sedikit sekali cahaya pantulan. Tetapi, jika sensor berada di atas garis putih, maka *photodiode* akan menerima banyak cahaya pantulan. Berikut ilustrasinya untuk semakin memperjelas pemahaman Anda:

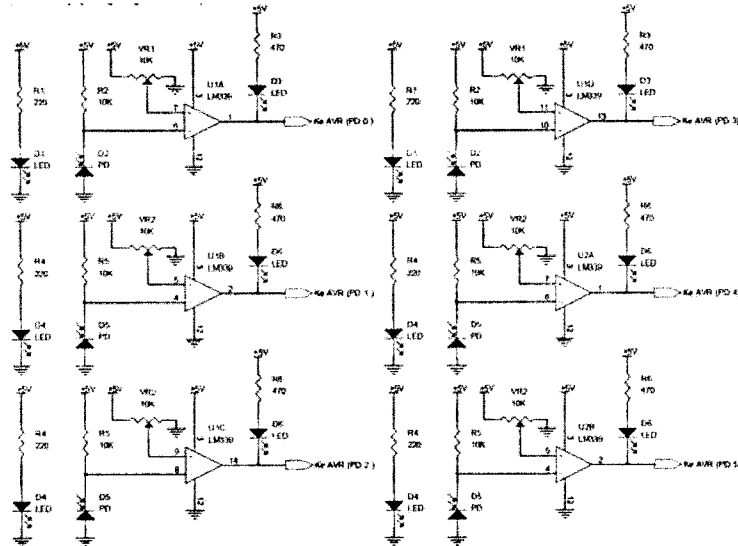


Gambar 2. Prinsip kerja sensor *proximity*; cahaya pantulan sedikit (kiri) dan cahaya pantulan banyak (kanan)

(images.myfilehost.us)

Sifat dari *photodiode* adalah jika semakin banyak cahaya yang diterima, maka nilai resistansi diodanya semakin kecil. Dengan melakukan sedikit modifikasi, maka besaran resistansi tersebut dapat diubah men-

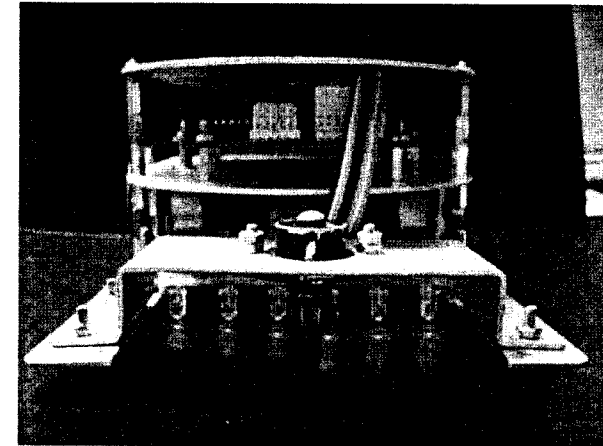
jadi tegangan. Dengan demikian, jika sensor berada di atas garis hitam, maka tegangan keluaran sensor akan kecil, demikian pula sebaliknya. Berikut adalah gambar rangkaian sensor *proximity* yang digunakan pada robot ini:



Gambar 3. Rangkaian sensor *proximity*
(images.myfilehost.us)

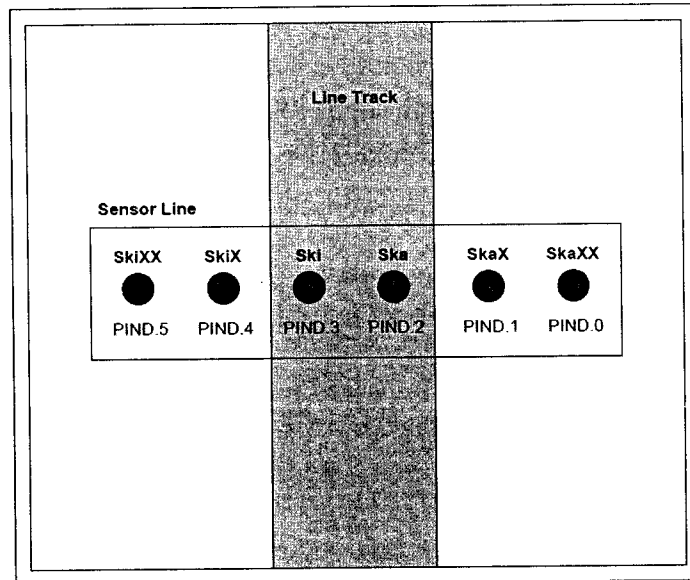
Agar dapat dibaca oleh *microcontroller*, maka tegangan sensor harus disesuaikan dengan level tegangan TTL, yaitu 0–1 volt untuk logika 0 dan 3–5 volt untuk logika 1. Hal ini bisa dilakukan dengan memasang *operational amplifier* yang difungsikan sebagai komparator. *Output* dari *photodiode* yang

masuk ke *input inverting op-amp* akan dibandingkan dengan tegangan tertentu dari variabel resistor VR. Tegangan dari VR inilah yang kita atur agar sensor *proximity* dapat menyesuaikan dengan kondisi cahaya ruangan.



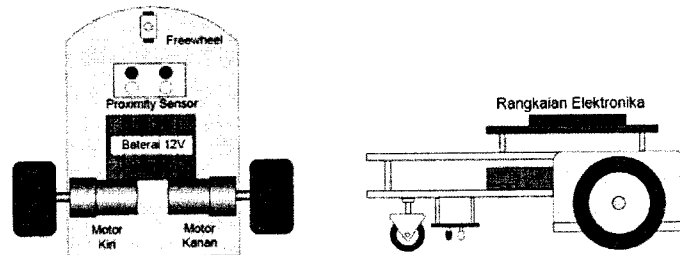
Gambar 4. Posisi pemasangan sensor *proximity* pada robot
(images.myfilehost.us)

Sensor *proximity* terdiri dari enam pasang LED dan *photodiode* yang disusun sedemikian rupa, sehingga jarak antara satu sensor dengan yang lainnya lebih kecil dari lebar garis hitam. Perhatikan gambar berikut:



Gambar 5. Jarak antarsensor *proximity*
(images.myfilehost.us)

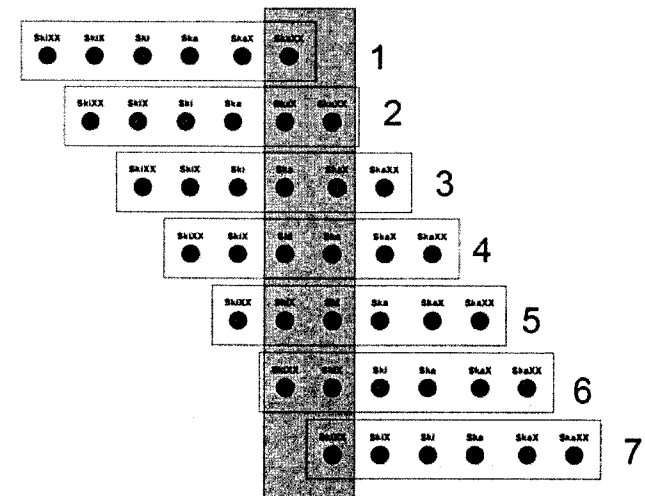
2. Rancangan Mekanik Robot



Gambar 6. Mekanik robot; robot tampak atas (kiri) dan tampak samping (kanan)
(images.myfilehost.us)

3. Algoritma Pergerakan Robot

Sebelum membuat program, kita perlu untuk mendefinisikan seluruh kemungkinan pembacaan sensor *proximity*. Dengan demikian, kita dapat menentukan pergerakan robot, dengan tujuan untuk menjaga agar robot selalu berada tepat di atas garis. Berikut adalah beberapa kemungkinan pembacaan garis oleh sensor *proximity*:



Gambar 7. Kemungkinan posisi sensor *proximity* pada *line*
(images.myfilehost.us)

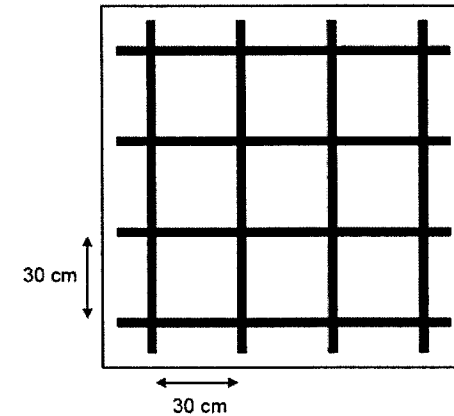
Setelah mengetahui kemungkinan-kemungkinan posisi sensor, selanjutnya kita harus mendefinisikan aksi dari setiap kondisi tersebut. Perhatikan tabel berikut:

Tabel 1. Aksi pergerakan robot

Posisi Sensor	Aksi Robot	Roda Kiri	Roda Kanan
1	belok kanan tajam	maju cepat	berhenti
2	belok kanan sedang	maju cepat	maju lambat
3	belok kanan ringan	maju cepat	maju sedang
4	maju lurus	maju cepat	maju cepat
5	belok kiri ringan	maju sedang	maju cepat
6	belok kiri sedang	maju lambat	maju cepat
7	belok kiri tajam	berhenti	maju cepat

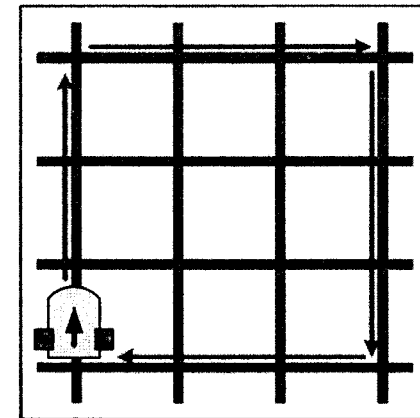
B. Lapangan Uji Coba

Lapangan uji coba berupa garis-garis hitam di atas lantai berwarna putih. Garis hitam disusun membentuk beberapa persimpangan. Ukuran tiap kotak adalah 30×30 cm. Ketebalan garis hitam adalah 3 cm. Garis hitam ini bisa dibuat menggunakan isolasi hitam, kemudian ditempel pada lantai atau kertas karton berwarna putih.



Gambar 8. Lapangan uji coba
(images.myfilehost.us)

Dalam aplikasi ini, robot akan bergerak mengikuti kotak terluar lapangan. Posisi awal robot seperti terlihat pada gambar berikut:

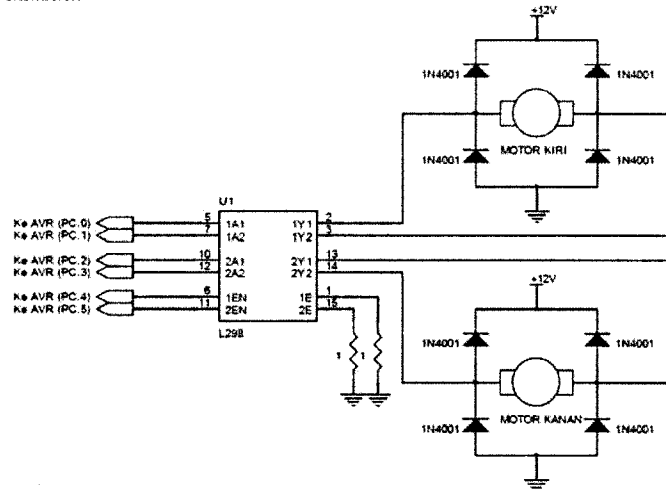
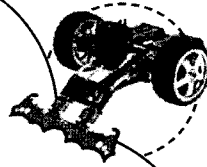


Gambar 9. Pergerakan robot di lapangan
(images.myfilehost.us)

1. Driver Motor DC

Untuk menggerakkan dua buah motor DC, digunakan IC H-Bridge Motor Driver L298, yang mampu memberikan arus maksimum sebesar 1A ke tiap motor. *Input* L298 ada 6 jalur, yang terdiri dari *input* data arah pergerakan motor dan *input* untuk PWM (*Pulse Width Modulation*). Untuk mengatur kecepatan motor, pada *input* PWM akan diberikan lebar pulsa yang bervariasi dari *microcontroller*.

Untuk menggerakkan dua buah motor DC, digunakan IC H-Bridge Motor Driver L298, yang mampu memberikan arus maksimum sebesar 1A ke tiap motor.



Gambar 10. Rangkaian *driver* motor DC

(images.myfilehost.us)

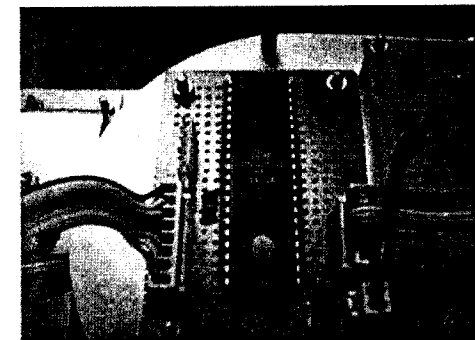
Untuk menentukan arah pergerakan motor, maka pada *input* L298 harus diberikan kondisi sesuai dengan tabel berikut:

Tabel 2. Tabel kebenaran *driver* motor

Motor 1						Motor 2					
1A1	1A2	1EN	1Y1	1Y2	Aksi Motor	2A1	2A2	2EN	2Y1	2Y2	Aksi Motor
0	0	1	0	0	Free Running Stop	0	0	1	0	0	Free Running Stop
0	1	1	0	12V	CW	0	1	1	0	12V	CW
1	0	1	12V	0	CCW	1	0	1	12V	0	CCW
1	1	1	12V	12V	Fast Stop	1	1	1	12V	12V	Fast Stop
x	x	0	0	0	Free Running Stop	x	x	0	0	0	Free Running Stop

2. AVR Microcontroller

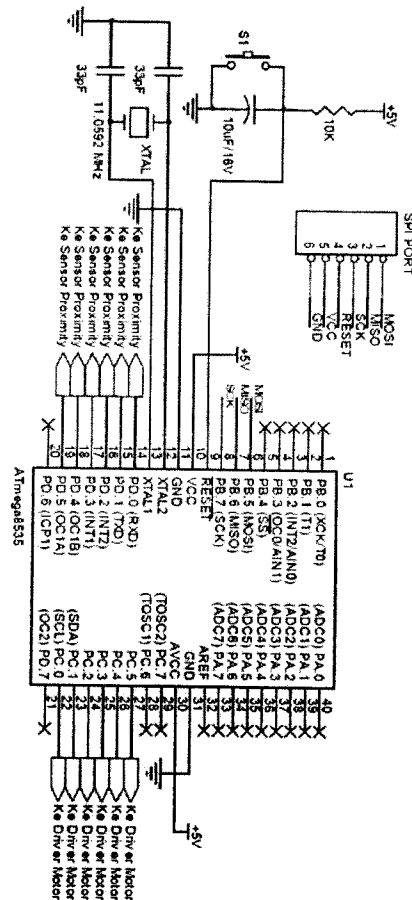
Sebagai "otak" robot, digunakan *microcontroller* AVR jenis ATmega 8535 yang akan membaca data dari sensor *proximity*, mengolahnnya, kemudian memutuskan arah pergerakan robot.



Gambar 11. *Microcontroller* Atmega 8535 pada robot

(images.myfilehost.us)

Pada robot *line track* ini, keluaran sensor *proximity* dihubungkan ke PortD.0 dan PortD.5 pada *microcontroller*. Sedangkan *driver* motor dihubungkan ke PortC.0 s/d PortC.5 seperti terlihat pada gambar berikut:



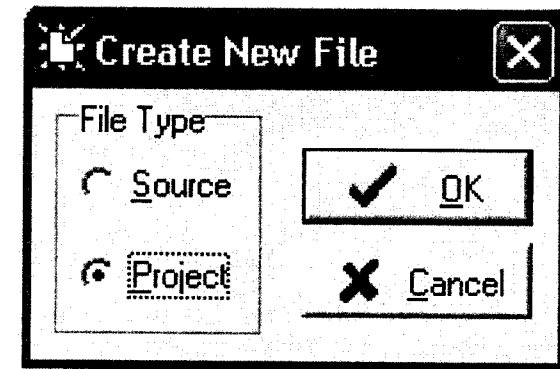
Gambar 12. Microcontroller ATmega 8535

(images.myfilehost.us)

3. Membuat Source Code

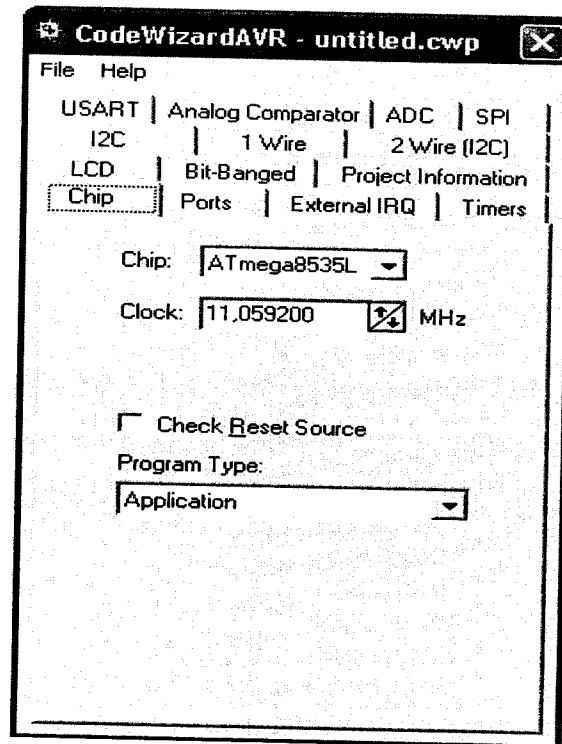
Source code secara lengkap dapat dilihat pada Lampiran A. *Source code* dibuat dengan menggunakan software CodeVisionAVR. Adapun langkah-langkah pembuatannya adalah sebagai berikut:

- Jalankan CodeVisionAVR, kemudian klik "File" → "New", lalu pilih "Project".

Gambar 13. Membuat *source code*

(images.myfilehost.us)

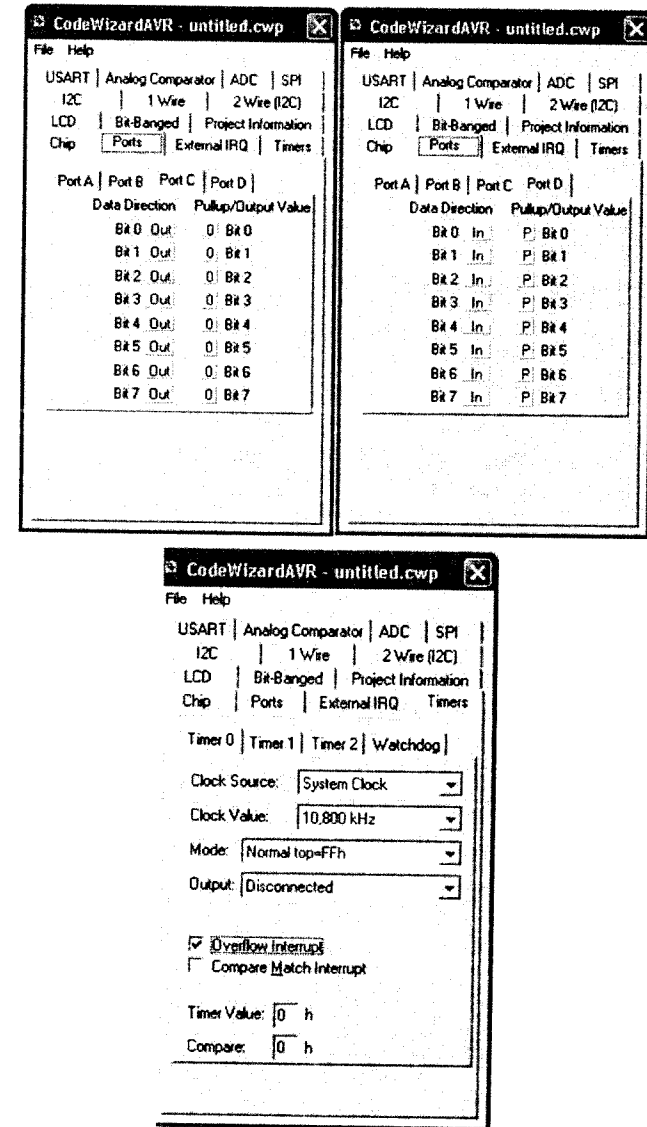
- Ketika keluar tampilan dengan pertanyaan, "Do you to use the CodeWizardAVR?" maka klik "Yes".
- Pilih *chip* yang digunakan, yaitu *chip* ATmega8535L dan *clock* 11.059200 MHz.



Gambar 14. ATmega 8535L pada tampilan program
(images.myfilehost.us)

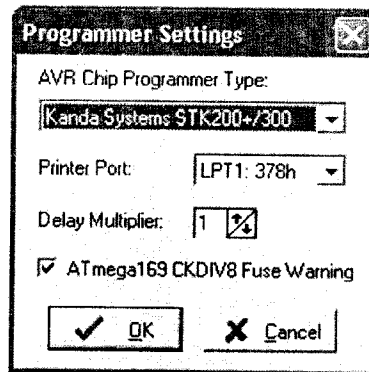
d. Lakukan *setting* sebagai berikut:

Pada bagian *Port*, tulislah Port C sebagai *Output* dan Port D sebagai *Input-up*. Sedangkan pada *Timers*, pilihlah "Timer 0" dengan Clock Value "10,800 KHz". Setelah itu, aktifkan "Overflow Interrupt".



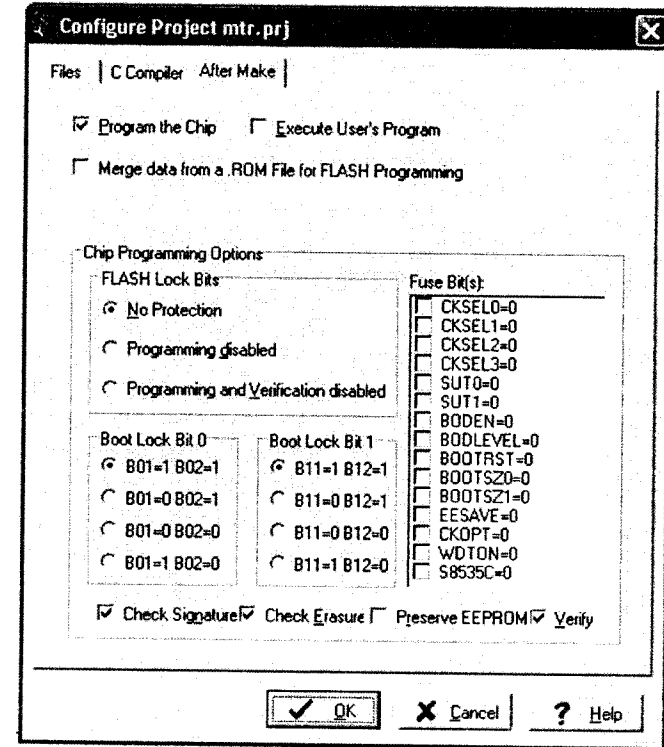
Gambar 15. Membuat tampilan *port* dan *timer* pada program
(images.myfilehost.us)

- e. Klik “File” → “Generate”, “Save”, and “Exit”.
- f. Buat *source code* seperti pada Lampiran A.
- g. Setelah selesai membuat *source code*, klik “Setting” → “Programmer”.
- h. Pada AVR Chip Programmer Type, pilih “Kanda System STK200+/300”, Printer Port pilih “LPT1: 378h”, Delay Multiplier pilih “1”, dan aktifkan “ATmega 169 CKDIV8 Fuse Warning”. Setelah itu tekan “OK”.



Gambar 16. Tipe AVR pada tampilan program
(images.myfilehost.us)

- i. Klik “Project” → “Configure”, kemudian pilih menu “After Make” dan aktifkan “Program the Chip”. Jika sudah, klik “OK”.
- PERHATIAN: Jangan mengubah *setting* apa pun pada menu ini. Sebab, jika Anda salah memilih, chip tidak akan bisa digunakan lagi!!!



Gambar 17. Tampilan project pada program
(images.myfilehost.us)

- j. Untuk meng-*compile project*, klik “Project” → “Make”.
- k. Jika tidak ada *error*, maka *file* siap di-*download* ke *chip*. Pastikan koneksi kabel *downloader* dan *chip* sudah terpasang dengan benar.
- l. Nyalakan *power supply* dan klik “Program”. Kemudian, tunggu hingga proses *download* selesai.

4. Penjelasan Source Code

Berikut adalah penjelasan tiap bagian dari *source code*.

- a. Membuat definisi *port* yang digunakan sebagai berikut:

```
#define SkiXX    PIND.0    // Sensor kiri terluar
#define SkiX     PIND.1    // Sensor kiri luar
#define Ski      PIND.2    // Sensor kiri tengah
#define Ska      PIND.3    // Sensor kanan tengah
#define SkaX     PIND.4    // Sensor kanan luar
#define SkaXX    PIND.5    // Sensor kanan terluar
#define EnKi     PORTC.4    // Enable L298 untuk motor kiri
#define dirA_Ki  PORTC.0    // Direction A untuk motor kiri
#define dirB_Ki  PORTC.1    // Direction B untuk motor kiri
#define EnKa     PORTC.5    // Enable L298 untuk motor
kanan
#define dirC_Ka  PORTC.2    // Direction C untuk motor kanan
#define dirD_Ka  PORTC.3    // Direction D untuk motor kanan
```

- b. Menentukan *library* yang digunakan sebagai berikut:

```
#include <mega8535.h> // Library untuk chip
ATmega8535
#include <delay.h>    // Library delay
```

- c. Membuat *variable* sebagai pengingat kondisi pembacaan sensor *line* terakhir.

```
bit x;
```

- d. Membuat ISR Timer 0/*Interrupt Service Routine* Timer 0.

Perlu diketahui bahwa:

- 1) ISR ini digunakan untuk menghasilkan pulsa PWM yang mampu mengendalikan motor kiri dan kanan melalui bit EnKi dan EnKa.

- 2) ISR Timer 0 dieksekusi secara periodik ketika Timer 0 *overflow*. Lama eksekusi sangat tergantung dari nilai.
- 3) Timer/Counter 0 (TCNT0).
- 4) Periode pulsa ditentukan oleh TCNT0 dan nilai maksimumnya 0xFF atau 255d.
- 5) *Duty cycle* PWM untuk motor kiri ditentukan oleh nilai "lpwm" dengan nilai maksimum 255.
- 6) *Duty cycle* PWM untuk motor kanan ditentukan oleh nilai "rpwm" dengan nilai maksimum 255.

```
unsigned char xcount,lpwm,rpwm; // Definisi
variable
// Timer 0 overflow interrupt service routine
interrupt [TIM0_OVF] void timer0_ovf_isr(void)
{
    // Place your code here
    xcount++; // xcount=xcount+1
    if(xcount<=lpwm)EnKi=1; // EnKi=1 jika xcount <=
    lpwm
    else EnKi=0; // EnKi=0 jika xcount >
    lpwm
    if(xcount<=rpwm)EnKa=1; // EnKa=1 jika xcount <=
    rpwm
    else EnKa=0; // EnKa=0 jika xcount >
    rpwm
    TCNT0=0xFF; // Timer0 Value Menentukan
    periode pulsa PWM
}
```

- e. Membuat subrutin agar robot bergerak maju.

```
void maju()
{
    dirA_Ki=1;dirB_Ki=0; // Motor kiri maju
    dirC_Ka=1;dirD_Ka=0; // Motor kanan maju
}
```

f. Membuat subrutin agar robot belok ke kiri.

```

void belok_kiri()
{
  unsigned int i;
  lpwm=50;          rpwm=50;      // Kecepatan pelan
  delay_ms(60);      // Robot dimajukan
  sedikit
  dirA_Ki=0;dirB_Ki=1;      // Motor kiri
  mundur
  dirC_Ka=1;dirD_Ka=0;      // Motor kanan
  maju
  for(i=0;i<=1000;i++)      while (!SkiXX ||!SkiX)
  {};
  for(i=0;i<=1000;i++)      while  SkiXX || SkiX)
  {};
  lpwm=0; rpwm=0;          // Robot berhenti

```

Pada program tersebut, tampak ada 2 “for while” yang masing-masing diulang 1.000 kali untuk memastikan bahwa sensor benar-benar membaca sebuah garis, bukan noda atau kotoran yang ada di lapangan.

1) `for(i=0;i<=1000;i++) while (!SkiXX ||!SkiX) {};`

Robot akan terus belok kiri selama sensor `SkiXX=0` atau `SkiX=0` (sensor berada di atas garis hitam).

2) `for(i=0;i<=1000;i++) while (SkiXX || SkiX) {};`

Robot tetap belok kiri selama sensor `SkiXX=1` atau `SkiX=1` (sensor berada di atas permukaan putih)

g. Membuat subrutin agar robot belok ke kanan.

```

void belok_kanan()
{
  unsigned int i;
  lpwm=50; rpwm=50;      // Kecepatan pelan
  delay_ms(60);          // Robot dimajukan sedikit
  dirA_Ki=1;dirB_Ki=0;    // Motor kiri maju
  dirC_Ka=0;dirD_Ka=1;    // Motor kanan mundur
  for(i=0;i<=1000;i++) while (!SkaXX ||!SkaX) {};
  for(i=0;i<=1000;i++) while ( SkaXX || SkaX) {};
  lpwm=0; rpwm=0;        // Robot berhenti
}

```

h. Membuat subrutin membaca *line*.

```

unsigned char sensor;
void scan_rule1()
{ maju(); // Robot bergerak maju
  sensor=PIN; // PIN diberi nama sensor
  sensor&=0b00111111; // sensor di-AND-kan dengan
  0b00111111
  switch(sensor)
  { case 0b00111110: rpwm=0; lpwm=200; x=1; break;
    case 0b00111100: rpwm=50; lpwm=200; x=1; break;
    case 0b00111101: rpwm=75; lpwm=200; x=1; break;
    case 0b00111001: rpwm=100; lpwm=200; x=1; break;
    case 0b00111011: rpwm=150; lpwm=200; x=1; break;
    case 0b00110011: rpwm=200; lpwm=200; break;
    case 0b00110111: rpwm=200; lpwm=150; x=0; break;
    case 0b00100111: rpwm=200; lpwm=100; x=0; break;
    case 0b00101111: rpwm=200; lpwm=75; x=0; break;
    case 0b00001111: rpwm=200; lpwm=50; x=0; break;
    case 0b00011111: rpwm=200; lpwm=0; x=0; break;
    case 0b00111111: break;
    if(x) {lpwm=50; rpwm=0; break;}
    else {lpwm=0; rpwm=50; break;}
  }
}

```

Variabel *x* berfungsi sebagai pengingat posisi terakhir robot terhadap garis. Jika robot berada di

kanan garis, maka $x = 0$. Jika robot berada di kiri garis, maka $x = 1$. Ketika robot lepas dari *track*, maka program akan membaca kondisi variabel x , sehingga dapat ditentukan arah gerak robot agar robot dapat kembali ke garis, seperti terlihat pada instruksi berikut:

```
if(x)      {lpwm=50;      rpwm=0;      break;}
else      {lpwm=0;      rpwm=50;     break;}

```

Dari instruksi tersebut, jika $x = 1$ maka robot belok kanan dan jika $x = 0$ maka robot belok kiri.

i. Membuat subrutin membaca persimpangan

```
void scan_count(unsigned char count)
{ unsigned int i;
  unsigned char xx=0;
  while(xx<count)
  { for(i=0;i<1000;i++) while((sensor &
    0b00011110)!=0b00000000) scan_rule1();
    for(i=0;i<1000;i++) while((sensor &
    0b00011110)==0b00000000) scan_rule1();
    xx++;
  }
}

```

Variable count digunakan untuk menentukan jumlah persimpangan yang harus dilewati. *Variable xx* berisi jumlah persimpangan yang telah dilewati. Nilainya akan bertambah 1 ketika kondisi 4 sensor tengah membaca garis hitam semua, kemudian membaca garis putih semua.

j. Membuat *main* program.

```
void main(void)
{
.
.
.
.
while (1)
{
  // Place your code here
  scan_count(3); // Maju 3 persimpangan
  belok_kanan(); // Belok kanan
};
}

```

5. Lampiran A; Source Code "Line Tracker Robot"

```
/******
This program was produced by the
CodeWizardAVR V1.24.0 Standard
Automatic Program Generator
© Copyright 1998-2003 HP InfoTech s.r.l.
http://www.hpinfotech.ro
e-mail:office@hpinfotech.ro

```

```
Project      :
Version      :
Date         : 17/11/2007
Author       : hendawan
Company      :
Comments     :

```

```
Chip type           : ATmega8535L
Program type        : Application
Clock frequency     : 11,059200 MHz
Memory model        : Small
External SRAM size  : 0
Data Stack size     : 128

```

```
*****/
#define SkiXX      PIND.0
#define SkiX       PIND.1
#define Ski        PIND.2
#define Ska        PIND.3

```

```

#define SkaX      PIND.4
#define SkaXX     PIND.5

#define EnKi      PORTC.4
#define dirA_Ki   PORTC.0
#define dirB_Ki   PORTC.1
#define EnKa      PORTC.5
#define dirC_Ka   PORTC.2
#define dirD_Ka   PORTC.3
#include <mega8535.h>
#include <delay.h>
bit x;
unsigned char xcount,lpwm,rpwm;
// Timer 0 overflow interrupt service routine
interrupt [TIM0_OVF] void timer0_ovf_isr(void)
{
    // Place your code here
    xcount++;
    if(xcount<=lpwm)EnKi=1;
    else EnKi=0;
    if(xcount<=rpwm)EnKa=1;
    else EnKa=0;
    TCNT0=0xFF; // Timer0 Value Menentukan periode
    pulsa PWM
}
void maju()
{
    dirA_Ki=1;dirB_Ki=0;
    dirC_Ka=1;dirD_Ka=0;
}
void belok_kiri()
{
    unsigned int i;
    lpwm=50; rpwm=50;
    delay_ms(60); // dimajukan sedikit
    dirA_Ki=0;dirB_Ki=1;
    dirC_Ka=1;dirD_Ka=0;
    for(i=0;i<=1000;i++) while (!SkiXX ||!SkiX) {};
    for(i=0;i<=1000;i++) while ( SkiXX || SkiX) {};
    lpwm=0; rpwm=0;
}
void belok_kanan()
{
    unsigned int i;
    lpwm=50; rpwm=50;
    delay_ms(60); // dimajukan sedikit
    dirA_Ki=1;dirB_Ki=0;

```

```

    dirC_Ka=0;dirD_Ka=1;
    for(i=0;i<=1000;i++) while (!SkaXX ||!SkaX) {};
    for(i=0;i<=1000;i++) while ( SkaXX || SkaX) {};
    lpwm=0; rpwm=0;
}
// Declare your global variables here
unsigned char sensor;
void scan_rule1()
{
    maju();
    sensor=PIND;
    sensor&=0b00111111;
    switch(sensor)
    {
        case 0b00111110: rpwm=0;   lpwm=200; x=1; break;
        case 0b00111100: rpwm=50;  lpwm=200; x=1; break;
        case 0b00111101: rpwm=75;  lpwm=200; x=1; break;
        case 0b00111001: rpwm=100;  lpwm=200; x=1; break;
        case 0b00111011: rpwm=150;  lpwm=200; x=1; break;
        case 0b00110011: rpwm=200;  lpwm=200; x=1; break;
        case 0b00110111: rpwm=200;  lpwm=150; x=0; break;
        case 0b00100111: rpwm=200;  lpwm=100; x=0; break;
        case 0b00101111: rpwm=200;  lpwm=75;  x=0; break;
        case 0b00001111: rpwm=200;  lpwm=50;  x=0; break;
        case 0b00011111: rpwm=200;  lpwm=0;   x=0; break;
        case 0b00111111: rpwm=200;  lpwm=0;   x=0; break;
        if(x) {lpwm=50; rpwm=0; break;}
        else {lpwm=0;  rpwm=50; break;}
    }
}
void scan_count(unsigned char count)
{
    unsigned int i;
    unsigned char xx=0;
    while(xx<count)
    {
        for(i=0;i<1000;i++) while((sensor &
            0b00011110)!=0b00000000) scan_rule1();
        for(i=0;i<1000;i++) while((sensor &
            0b00011110)==0b00000000) scan_rule1();
        xx++;
    }
}
void main(void)
{
    // Declare your local variables here
    // Input/Output Ports initialization
    // Port A initialization
    // Func0=In Func1=In Func2=In Func3=In Func4=In
    Func5=In Func6=In Func7=In

```

```
// State0=T State1=T State2=T State3=T State4=T
State5=T State6=T State7=T
PORTA=0x00;
DDRA=0x00;

// Port B initialization
// Func0=In Func1=In Func2=In Func3=In Func4=In
Func5=In Func6=In Func7=In
// State0=T State1=T State2=T State3=T State4=T
State5=T State6=T State7=T
PORTB=0x00;
DDRB=0x00;

// Port C initialization
// Func0=Out Func1=Out Func2=Out Func3=Out Func4=Out
Func5=Out Func6=Out Func7=Out
// State0=0 State1=0 State2=0 State3=0 State4=0
State5=0 State6=0 State7=0
PORTC=0x00;
DDRC=0xFF;

// Port D initialization
// Func0=In Func1=In Func2=In Func3=In Func4=In
Func5=In Func6=In Func7=In
// State0=P State1=P State2=P State3=P State4=P
State5=P State6=P State7=P
PORTD=0xFF;
DDRD=0x00;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: 10,800 kHz
// Mode: Normal top=FFh
// OC0 output: Disconnected
TCCR0=0x05;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer 1 Stopped
```

```
// Mode: Normal top=FFFFh
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
```

```
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;
```

```
// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer 2 Stopped
// Mode: Normal top=FFh
// OC2 output: Disconnected
```

```
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;
```

```
// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
```

```
MCUCR=0x00;
MCUCSR=0x00;
```

```
// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x01;
```

```
// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter
1: Off
// Analog Comparator Output: Off
ACSR=0x80;
SFIOR=0x00;
```

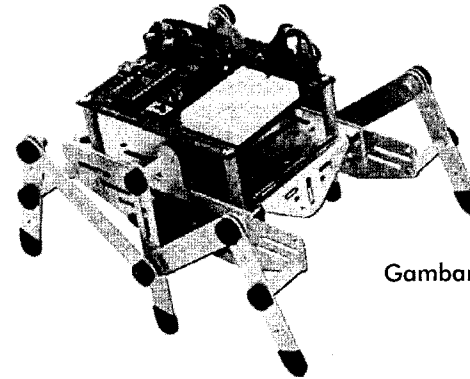
```
// Global enable interrupts
#asm("sei")
```

```
while (1)
{
    // Place your code here
    scan_count(3);
    belok_kanan();
};
}
```



Bab 5

Membuat Robot Berkaki dan Berlengan



Gambar 1. Crawler Kit Boe-Bot

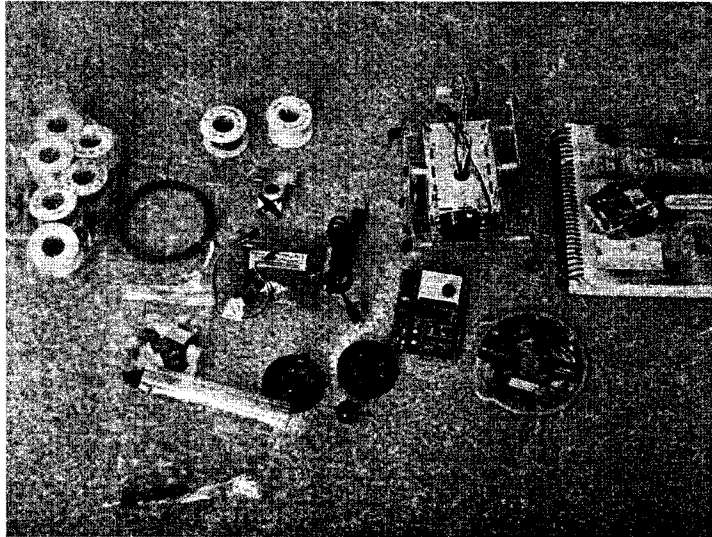
(generationrobots.com)

Robot berkaki sangat menarik untuk dibuat, karena membutuhkan teknik pergerakan aktuator robot yang lebih rumit. Begitu juga untuk lengan robot (*arm robot*). Pada bab ini, akan dicontohkan cara mengendalikan gerak robot berkaki dan lengan robot menggunakan servo *controller*.

A. Robot Berkaki (Crawler Boe-Bot)

Boe-Bot mempunyai kit tambahan agar dapat bergerak menggunakan kaki, yang dikenal sebagai

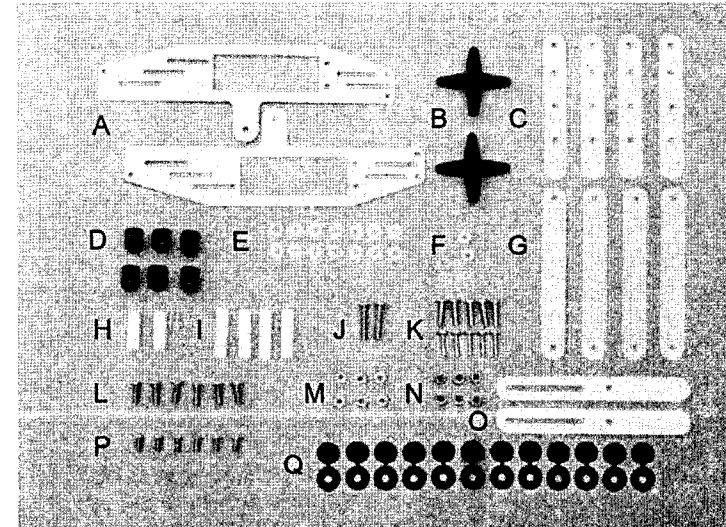
Crawler Kit. Anda sebenarnya dapat membuat sendiri kaki Boe-Bot tersebut, asalkan peralatan yang akan Anda gunakan betul-betul lengkap.



Gambar 2. Peralatan yang digunakan
(img.photobucket.com)

Berikut akan saya sajikan langkah-langkah untuk membuat robot berkaki:

1. Sediakan komponen yang diperlukan. Anda dapat menggunakan lempengan seng atau logam lainnya sebagai bahannya.



Gambar 3. Komponen yang dibutuhkan
(images.mylifehost.us)

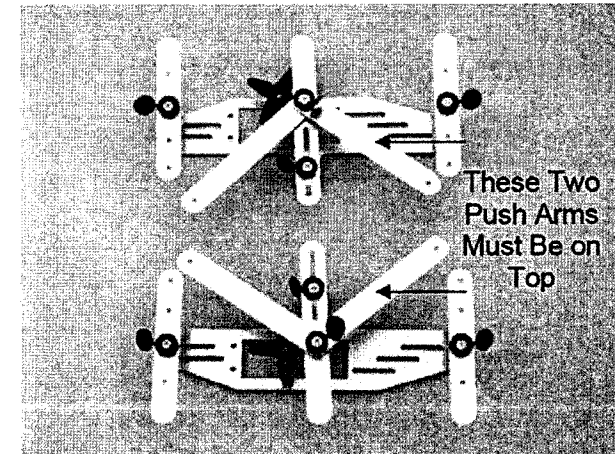
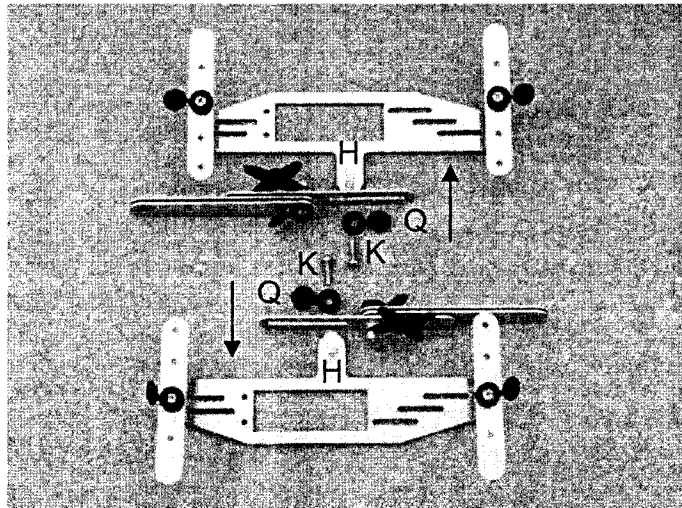
Keterangan gambar:

- A. Crawler Sides (2 buah)
- B. Servo horns (2 buah)
- C. Legs (4 buah)
- D. Rubber feet (6 buah)
- E. Nylon washers (14 buah)
- F. 3 mm (1/8") Nylon Spacer (2 buah)
- G. Push Arms (4 buah)
- H. 19 mm (3/4") Nylon Standoffs (2 buah)
- I. 25 mm (1") Nylon Standoff (4 buah)
- J. M3x18 Phillips Pan Head Screw (2 buah)
- K. M3x12 Phillips Pan Head Screw (10 buah)

- L. M3x10 Phillips Pan Head Screw (6 buah)
- M. M3 Hex Nut (6 buah)
- N. M3 nylon insert locknuts (6 buah)
- O. Middle Leg (2 buah)
- P. M3x6 Phillips Pan Head Screw (6 buah)
- Q. Plastic Screw Covers (12 buah)

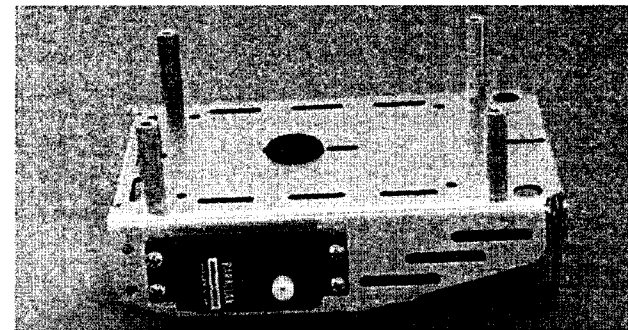
Komponen-komponen untuk membuat kaki tersebut, dapat Anda modifikasi atau buat sendiri. Yang penting, bentuknya mirip dan dapat bergerak dengan baik.

2. Rangkailah komponen-komponen tersebut seperti gambar berikut:



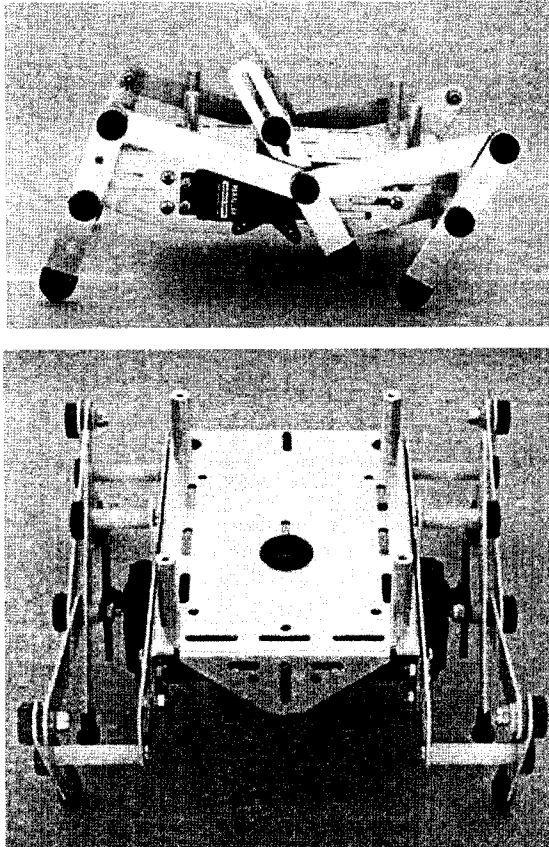
Gambar 4. Gelang kaki Boe-Bot yang dirakit
(images.myfilehost.us)

Berikut contoh hubungan antara tulang-tulang kaki dengan servo motor, sehingga perputaran servo motor menyebabkan kaki bergerak.



Gambar 5. Hubungan servo dengan tulang kaki
(images.myfilehost.us)

3. Pasanglah tulang-tulang kaki tersebut ke badan, sehingga tampil seperti gambar berikut:



Gambar 6. Hasil akhir pemasangan kaki Boe-Bot
(images.myfilehost.us)

4. Buatlah kode agar Boe-Bot berkaki mampu bergerak. Berikut adalah contoh kode Boe-Bot

berkaki yang mampu bergerak maju serta belok ke kiri dan ke kanan, lalu berjalan mundur dengan *delay* beberapa detik:

```
BoeBotKaki.bs2:
'Robot Boe Bot Berkaki
' { $srA }. { P BS2 }
' { $PBASTC 2.5 }
pufse_count VAR Word

'--- Inisialisasi
LOW 12
LOW 13

'--- Rutin utama-----
main:
forward:
FOR pulse_count = 1 TO 100
PULSOUT 12,500
PULSOUT 13,1000
PAUSE 20
NEXT
PAUSE 700
left_turn:
FOR pufse_count = 1 TO 90
PULSOUT 12,500
PULSOUT 13,500
PAUSE 20
NEXT
PAUSE 700

right_turn:
FOR pulse_coun! = 1 TO 180
PULSOUT 12,1000
PULSOUT 13,1000
PAUSE 20
NEXT
PAUSE 700

left_turn:
FOR pulse_count = 1 To 90
PULSOUT 12,500
PULSOUT 13,500
PAUSE 20
```

```

NEXT
PAUSE 700

backward:
FOR pulse_count. = 1 TO 100
  PULSOUT 12,1000
  PULSOUT 13,500
  PAUSE 20
NEXT
PAUSE 700
STOP

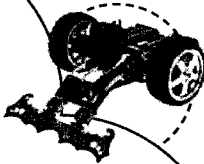
```

B. Robot Berkaki Hex (Quad Crawler)

Crust crawler menyediakan kit robot berkaki 4 dan berkaki 6, yang disebut *Quad Crawler* dan *Hex Crawler*. Pada buku ini hanya akan dicontohkan robot berkaki 4 (*Quad crawler*), karena sudah memadai sebagai dasar pengembangan robot berkaki.

Di sini, tugas Anda adalah merancang robot berkaki seperti kaki kepiting yang dapat berjalan ke segala arah. Robot ini sudah dibuat oleh salah satu peserta kontes robot di Indonesia.

Crust crawler menyediakan kit robot berkaki 4 dan berkaki 6, yang disebut *Quad Crawler* dan *Hex Crawler*. Robot berkaki 4 (*Quad crawler*), karena sudah memadai sebagai dasar pengembangan robot berkaki.



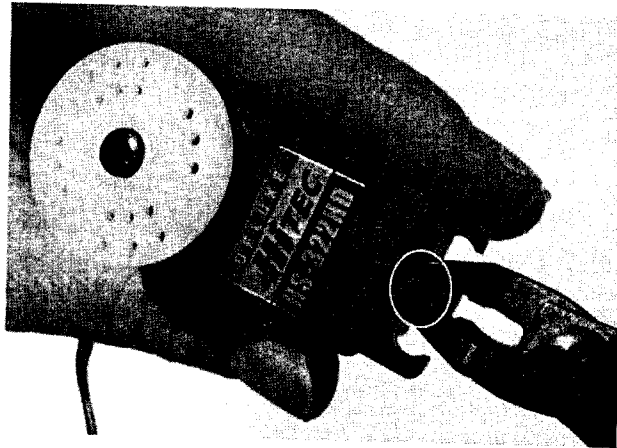
1. Bahan-Bahan yang Diperlukan

Jika Anda membeli kit *Quad crawler*, komponen yang diperoleh antara lain:

- a. Quad crawler robot metal kit (1 buah)
- b. Quad crawler robot manual (1 buah)
- c. BASIC stamp manual (1 buah)
- d. Hitec servos (8 buah)
- e. Parallax servo controller/PSC (1 buah)
- f. BASIC stamp 2 module (1 buah)
- g. Board of education programming board (1 buah)
- h. 7-segment LED (1 buah)
- i. 1K ohm resistors (7 buah)
- j. Resistor 10K ohm (2 buah)
- k. Pushbutton (2 buah)
- l. Resistor 220 ohm (2 buah)
- m. Bag 3" jumper wires (1 buah)
- n. Kabel serial (1 buah)
- o. Parallax CD-ROM (1 buah)
- p. Distance sensor (1 buah)
- q. Custom sensor cable (1 buah)

Quad crawler menggunakan servo motor berdaya tinggi dan bermerek Hitec. Pastikan Anda memotong salah satu bagian di servo motor tersebut agar dapat

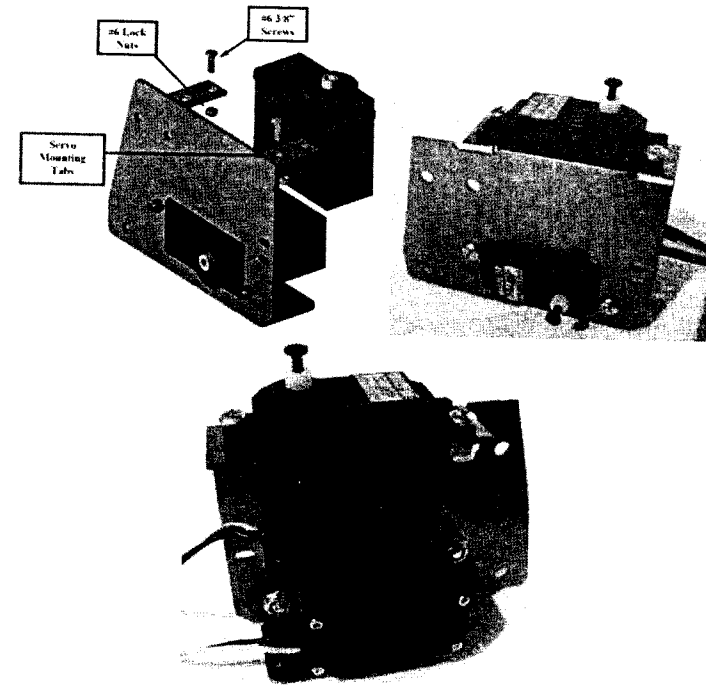
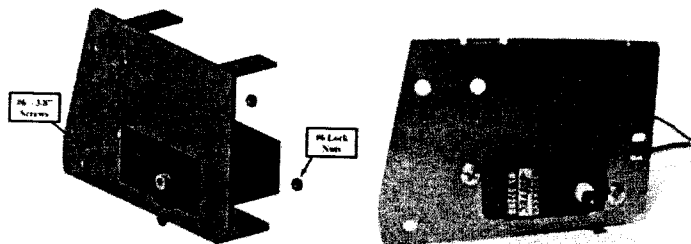
dipasang dengan pas ke badan robot seperti gambar berikut:



Gambar 7. Merapikan servo motor
(images.myfilehost.us)

2. Pemasangan Servo Motor Pada Badan Robot

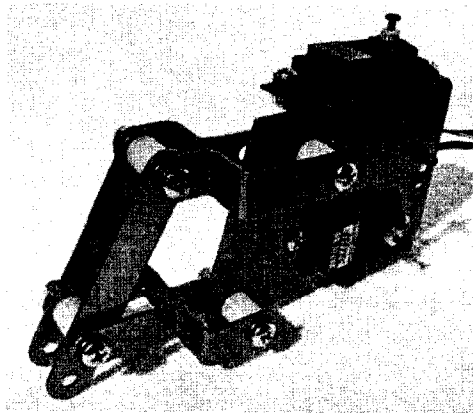
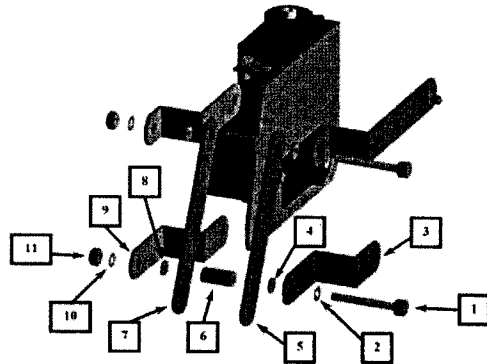
Gambar di bawah menampilkan bentuk pemasangan servo motor pada badan robot. Perhatikanlah bahwa dalam pemasangan ini, gunakan baut yang sesuai untuk servo motor.



Gambar 8. Pemasangan servo motor di badan robot
(images.myfilehost.us)

3. Komponen Penyangga Kaki Robot

Pada kaki robot, digunakan banyak persendian yang harus dapat bergerak dengan mudah. Oleh karena itu, dibutuhkan berbagai komponen penyangga dari bahan plastik sebagai berikut:



Gambar 9. Tulang kaki yang sudah dipasang
(images.myfilehost.us)

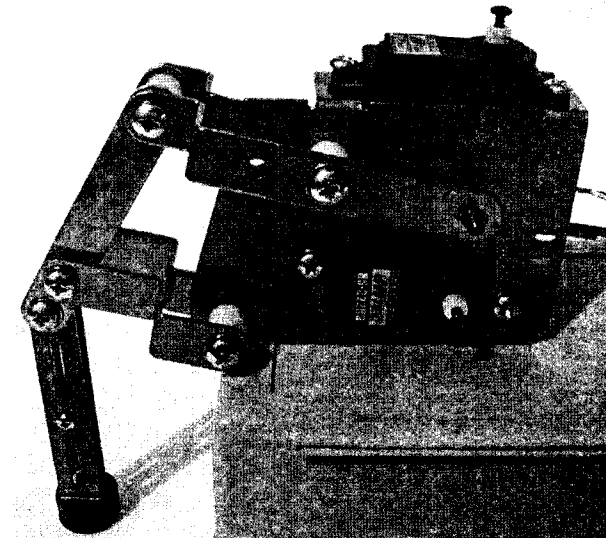
Keterangan gambar:

1. #8 1 1/4" Screw
2. #8 Stainless Flat Washer
3. Lower Horizontal Leg Brace
4. #8 Flat Nylon Spacer
5. Vertical Leg Brace

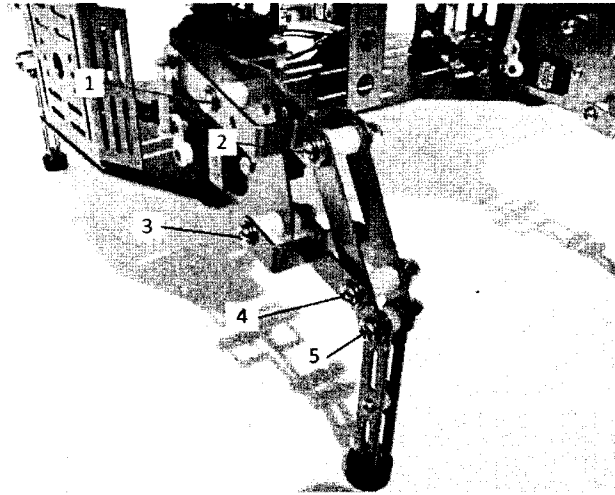
6. #8 1/2" Nylon Spacer
7. Vertical Leg Brace
8. #8 Flat Nylon Spacer
9. Lower Horizontal Leg Brace
10. #8 Stainless Flat Washer
11. #8 Lock Nut

4. Memastikan Pergerakan Kaki Robot

Gambar berikut menampilkan bagaimana servo motor terhubung ke kaki agar mampu menggerakkan kaki ke atas dan ke bawah.



Gambar 10. Penghubung plastik antara tulang dengan servo
(images.myfilehost.us)



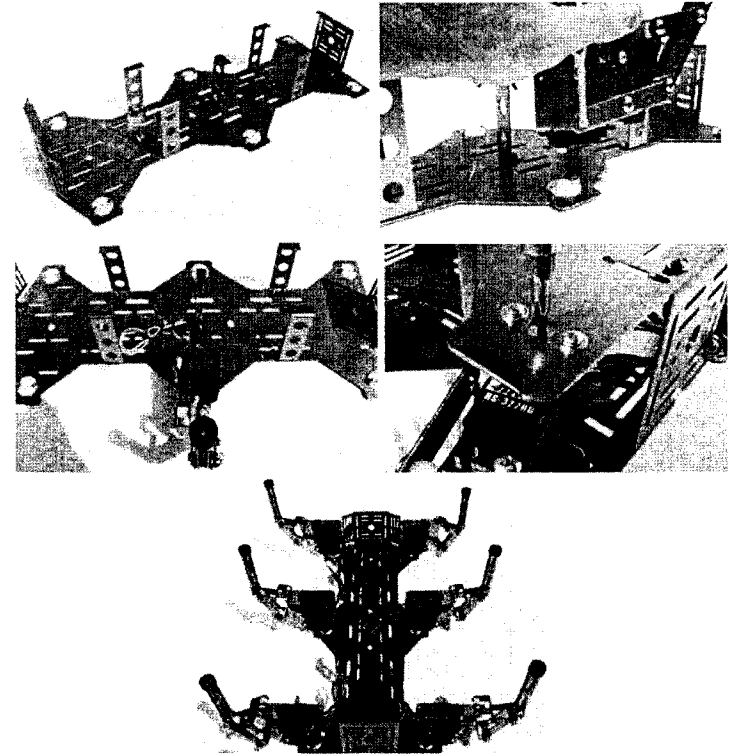
Gambar 11. Pengaturan pada kaki robot
(images.myfilehost.us)

Untuk memastikan pergerakan kaki robot yang sempurna, ada beberapa yang harus diperhatikan, yaitu:

- gunakan baut disertai bantalan yang memadai, sehingga persendian dapat bergerak tidak seret;
- pastikan lower leg severtikal mungkin untuk akurasi terbaik;
- pastikan juga titik ini diatur, sehingga pergerakan kaki dapat maksimal;
- pada bagian ini, atur baut agar tulang kaki dapat bergerak dengan mudah;
- pada bagian ini, atur juga baut agar posisi tulang paling bawah kokoh.

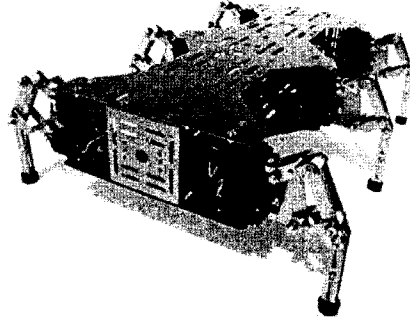
5. Perakitan dan Pemasangan Servo Motor

Gambar berikut menampilkan kerangka robot yang baru dirakit dan dipasang servo motor.



Gambar 12. Rangkaian pemasangan servo pada kerangka lengan robot
(images.myfilehost.us)

Quad crawler merupakan jenis robot yang menggunakan servo *controller*, sehingga mampu mengendalikan banyak servo motor.

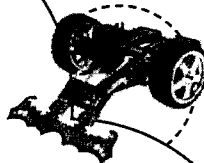


Gambar 13. *Basic stamp* dan *servo controller* di posisi atas pada *hex crawler*
(images.myfilehost.us)

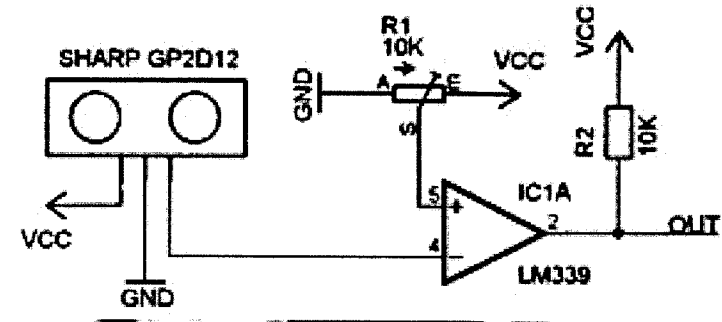
6. Penggunaan Sensor Sharp GP2D12

Sensor sharp GP2D12 digunakan untuk membaca jarak. Sensor ini menggunakan prinsip pantulan sinar inframerah. Dalam aplikasinya, nilai tegangan yang keluar dari sensor berbanding terbalik dengan hasil pembacaan jarak yang dikomparasikan dengan

Sensor sharp GP2D12 yang digunakan untuk membaca jarak, pada dasarnya menggunakan prinsip pantulan sinar inframerah. Dalam aplikasinya, nilai tegangan yang keluar dari sensor berbanding terbalik dengan hasil pembacaan jarak yang dikomparasikan dengan tegangan referensi komparator.



tegangan referensi komparator. Rangkaian sistem komparator pembacaan jarak dengan sensor sharp GP2D12 ini, disajikan pada gambar berikut:



Gambar 14. Skema rangkaian sensor sharp GP2S12
(images.myfilehost.us)

Prinsip kerja dari rangkaian komparator sensor sharp GP2D12 pada gambar tersebut adalah jika sensor mengeluarkan tegangan melebihi tegangan referensi, maka keluaran dari komparator akan berlogika rendah. Jika tegangan referensi lebih besar dari tegangan sensor, maka keluaran dari komparator akan berlogika tinggi.

Selain menggunakan komparator, untuk mengakses sensor jarak sharp GP2D12, dapat digunakan prinsip ADC. Atau, dengan kata lain, mengolah sinyal analog dari pembacaan sensor sharp GP2D12 ke bentuk digital dengan bantuan pemrograman. Dalam pemrograman BASCOM-AVR, untuk meng-

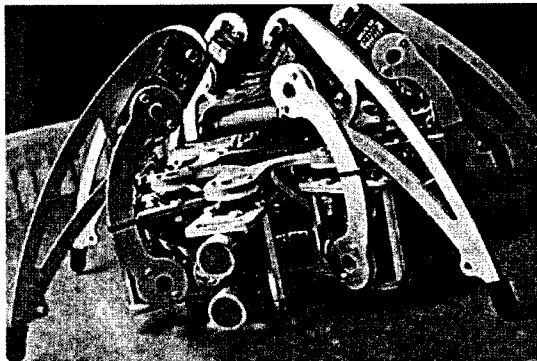
akses sensor ini, dapat digunakan fasilitas akses ADC yang cukup mudah.

C. Lengan Robot

Robot humanoid membutuhkan lengan untuk mengambil barang yang biasanya dikenal dengan lengan robot. Perangkat yang dibutuhkan untuk membuat lengan robot ini adalah:

1. kit *microcontroller* basic stamp,
2. servo QWS S04 2bh,
3. gripper,
4. servo hitec/gws/parallax/lainnya,
5. body acrylic 3–4 mm, dan
6. program visual studio .net 2008.

Rakitlah lengan robot seperti gambar di bawah ini, di mana servo terpasang mulai dari P0-P5.



Gambar 15. Lengan robot menggunakan servo Smaff Arm Robot
(rosesana.com)

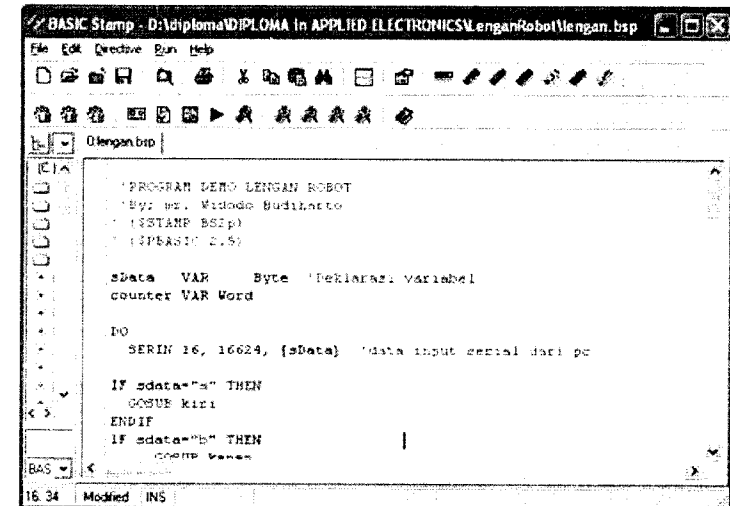
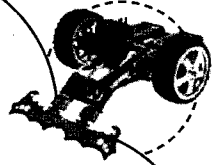
D. Kode Program

Gunakan *Basic Stamp Compiler* untuk membuat dan mengisi program ke mikro. Sedangkan untuk menerima data serial, gunakan fungsi **SERIN**. Berikut adalah kode-kodenya:

Untuk membuat dan mengisi program ke mikro, gunakanlah

Basic Stamp Compiler.

Sedangkan untuk menerima data serial, gunakanlah fungsi **SERIN**.



Gambar 16. Tampilan kode program sebuah lengan robot
(images.myfilehost.us)

1. Kode Program Agar Robot Melakukan Aksi Tertentu

Robot akan bekerja berdasarkan data yang diterima dari PC berupa data (a–j), yang kemudian diterjemahkan untuk melakukan aksi tertentu.

```
Lengranrobot.bsp:
'PROGRAM DEMO IJENGAN ROBOT Smart Arm Robot
'Copyright e-Technology Center 2010
'Dengan 6 servo dan Gripper
` {$srAMP Bs2p}
` {$PBAsrC 2.5}
sData VAR Byte 'Deklarasi variabel
counter VAR Word

DO
SERIN 16, 6624, [sData] 'data input serial dari pc

IF sdata="a" THEN
  GOSUB kiri
ENDIF
IF sdata="b" THEN
  GOSUB kanan
ENDIF
IF sdata="c" THEN
  GOSUB atas
ENDIF
IF sdata="d" THEN
  GOSUB bawah
ENDIF
...
IF sdata="e" THEN
  GOSUB latas
ENDIF
...

LOOP
'Definisi fungsi
kiri:
  FOR counter=1 TO 3
    PUISOUT 15,2500
  NEXT
```

```
RETURN

kanan:
  FOR counter=1 TO 3
    PUISOUT 1.5,200
  NEXT
  RETURN

bawah:
  FOR counter=1 TO 6
    PULSOUT 0,300
    PULSOUT 7,3500
  NEXT
  RETURN

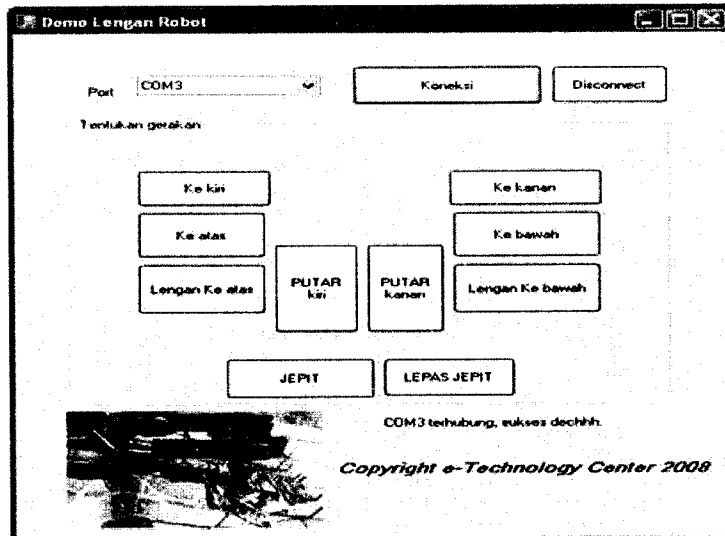
atas:
  FOR counter=1 TO 20
    PULSOUT 0,3500
    PUT,SOUT 7,300
  NEXT
  RETURN

....
cengkeram:
  FOR counter=1 To 4
    PULSOUT 3,300
  NEXT
  RETURN

bukacengkeram:
  FOR counter=1 TO 4
    PULSOUT 3,2500
  NEXT
  RETURN
```

2. Kode Program untuk Tampilan Grafis

Untuk tampilan grafis, gunakan Visual studio .net 2008.



Gambar 17. Tampilan program lengan robot

(images.myfilehost.us)

```
Public Class Form1
    Dim WithEvents serialPort As New IO.Ports.
    serialPort
    Private Sub Form1_Load(ByVal sender As System.
    Object, ByVal e As
    System.EventArgs) Handles MyBase.Load
        displayPort ( )
    End Sub
    Private Sub displayPort () 'mencari daftar com
    port
        For I As Integer = 0 To My.Computer.Ports.
        SerialPortNames.Count - 1
            cbport.Items.Add(My.
            Computer.Ports.SerialPortNames(i))
        Next
        cbport.SelectedIndex = 0
    End Sub
    Private Sub Button2_Click(ByVal sender As System.
    Object, ByVal e As
    System.EventArgs) Handles Button2.Click
        serialPort.Write ( "b" )
    End Sub
```

```
Private Sub Button1_Click(ByVal sender As System.
Object, ByVal e As
System.EventArgs) Handles Button1.Click
    serialPort.Write ( "a" )
End Sub
Private Sub Button7_Click(ByVal sender As System.
Object, ByVal e As
System.EventArgs) Handles Button7.Click
    If serialPort.IsOpen Then
        serialPort.Close ( )
    End If
    Try
        With serialPort
            .PortName = cbport.Text
            .BaudRate = 9600
            .Parity = IO.Ports.Parity.None
            .DataBits = 8
            .StopBits = IO.Ports.StopBits.One
        End With
        serialPort.Open()
        Label3.Text = cbport.Text & " terhubung,
        sukses dechhh."

        Catch ex As Exception
            MsgBox(ex.ToString)
        End Try
    End Sub

    Private Sub Button5_Click(ByVal sender As System.
    Object, ByVal e As System.EventArgs) Handles
    Button5.Click
        serialPort.Write("I") 'kirim data
    End Sub

    ...

    Private Sub Button9_Click(ByVal sender As System.
    Object, ByVal e As System.EventArgs) Handles
    Button9.Click
        serialPort.Write("f") 'kirim data
    End Sub

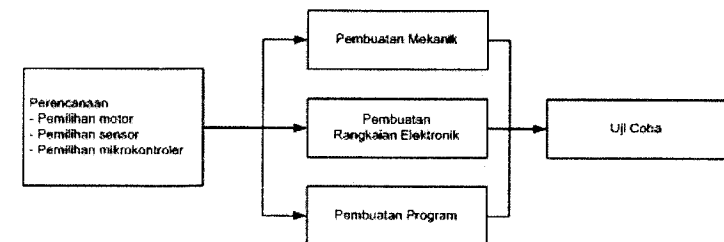
    Private Sub Button10_Click(ByVal sender As System.
    Object, ByVal e As System.EventArgs) Handles
    Button10.Click
        serialPort.Write("g") 'kirim data
    End Sub
```

```
Private Sub Button11_Click(ByVal sender As System.  
Object, ByVal e As System.EventArgs) Handles  
Button11.Click  
    serialPort.Write("h") ' kirim data  
End Sub  
End Class
```



Membuat Robot Cerdas KRCI

Secara garis besar, tahap-tahap pembuatan robot dapat dilihat pada gambar berikut:



Gambar 1. Skema pembuatan robot KRCI
(download.ebookgratis.info)

Ada tiga tahapan dalam pembuatan robot agar robot tersebut dapat digunakan sesuai keinginan.

Ketiga tahapan tersebut meliputi perencanaan (pemilihan hardware dan design), pembuatan (pembuatan mekanik, elektronik, dan program), dan uji coba.

Ada tiga tahapan dalam pembuatan robot agar robot tersebut dapat digunakan sesuai keinginan.

Ketiga tahapan tersebut meliputi perencanaan (pemilihan hardware dan design), pembuatan (pembuatan mekanik, elektronik, dan program), dan uji coba.



A. Tahap Perencanaan

Dalam tahap ini, kita akan mencoba merencanakan apa yang ingin dibuat. Sederhananya, kita hendak membuat robot yang seperti apa atau robot yang berguna untuk apa? Dalam tahap perencanaan ini, yang perlu kita tentukan adalah:

1. Dimensi; yaitu panjang, lebar, tinggi, dan perkiraan berat dari robot. Robot KRI mempunyai tinggi sekitar 1 m, sedangkan tinggi robot KRCI sekitar 25 cm.
2. Struktur material; apakah dari aluminium, besi, kayu, plastik, dan sebagainya.
3. Cara kerja robot; berisi bagian-bagian robot dan fungsi dari bagian-bagian itu, misalnya lengan, *conveyor*, *lift*, *power supply*, dan sebagainya.

4. Sensor-sensor yang akan digunakan robot.
5. Mekanisme; bagaimana sistem mekanik yang harus dibuat agar robot dapat menyelesaikan tugas.
6. Metode pengontrolan; yaitu bagaimana robot dapat dikontrol dan digerakkan, *microprocessor* yang digunakan, dan blok diagram sistem yang dibuat.
7. Strategi untuk memenangkan pertandingan; jika memang robot itu akan diikuti lomba/kontes robot Indonesia/internasional.

B. Tahap Pembuatan

Ada tiga pekerjaan yang harus dilakukan dalam tahap ini, yaitu pembuatan mekanik, elektronik, dan *programming*. Masing-masing tahapan membutuhkan tenaga dengan spesialisasi yang berbeda-beda, yaitu spesialis mekanik (bidang ilmu yang cocok adalah teknik mesin dan teknik industri), spesialis elektronika (bidang ilmu yang cocok adalah teknik elektro), dan spesialis *programming* (bidang ilmu yang cocok adalah teknik informatika).

1. Pembuatan Mekanik

Setelah gambaran garis besar bentuk robot dirancang, maka rangka dapat mulai dibuat. Umumnya,

rangka robot KRI terbuat dari aluminium kotak atau aluminium siku. Satu ruas rangka terhubung satu sama lain dengan keling aluminium. Keling adalah semacam paku aluminium yang berguna untuk menempelkan lembaran logam dengan erat. Sedangkan untuk rangka robot KRCI lebih variatif lagi, karena bisa juga terbuat dari plastik atau besi panjang seperti jeruji.

2. Pembuatan Sistem Elektronika

Bagian sistem elektronika dirancang sesuai dengan fungsi yang diinginkan, misalnya untuk menggerakkan motor DC diperlukan bridge, sedangkan untuk menggerakkan relay diperlukan saklar transistor.

a. Memahami Sensor-Sensor yang Digunakan

Sensor-sensor yang akan digunakan, harus dipelajari dan dipahami cara kerjanya, misalnya:

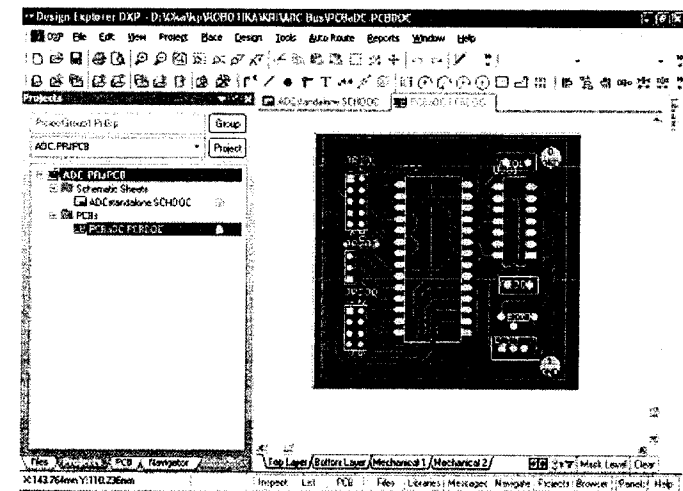
- 1) sensor jarak, bisa menggunakan SRF04, GP2D12, atau merakit sendiri modul sensor ultrasonik atau inframerah;
- 2) sensor arah, bisa menggunakan sensor kompas CMPS03 atau Dinsmore;
- 3) sensor suhu, bisa menggunakan LM35 atau sensor yang lain;
- 4) sensor nyala api/panas, bisa menggunakan UVTron atau Thermopile; atau

5) sensor *line follow/line detector*, bisa menggunakan led dan phototransistor.

b. Tahapan Pembuatan Sistem Elektronika

Pembuatan sistem elektronika ini meliputi tiga tahap:

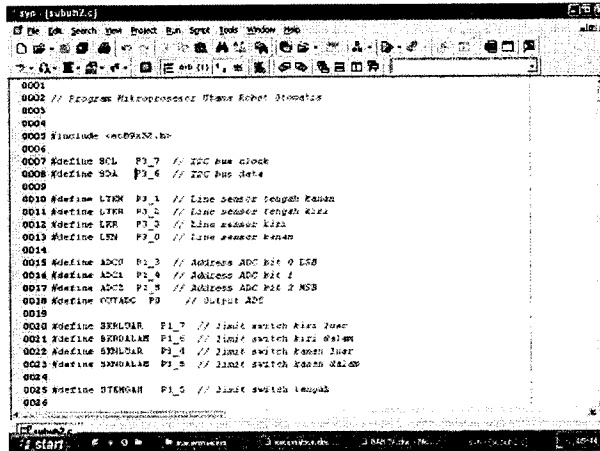
- 1) tahap design PCB, misalnya dengan program Altium DXP;
- 2) tahap pencetakan PCB, bisa dengan Pro-board; dan
- 3) tahap perakitan dan pengujian rangkaian elektronika.



Gambar 2. Tampilan program sensor jarak
(alfan90.student.umm.ac.id)

3. Pembuatan Software/Program

Pembuatan *software* dilakukan setelah alat siap untuk diuji. *Software* ini ditanamkan (di-download) pada *microcontroller*, sehingga robot dapat berfungsi sesuai dengan yang diharapkan.



Gambar 3. Tampilan program untuk *microcontroller*
(alfan90.student.umm.ac.id)

a. Tahap-Tahap Pembuatan Program

1) Perancangan algoritma atau alur program

Untuk fungsi yang sederhana, algoritma dapat dibuat langsung pada saat menulis program. Untuk fungsi yang lebih kompleks, algoritma dibuat dengan menggunakan flow chart.

2) Penulisan program

Penulisan program dapat menggunakan bahasa C, Assembly, Basic, atau bahasa yang paling dikuasai.

3) Compile dan download

Compile dan download adalah mentransfer program yang kita tulis kepada robot yang dibuat.

b. Contoh Kode Program Robot KRCI

KRCI.bs2:

```
'DEMO PROGRAM ROBOT KRCI
' {$sRAMP BS2P}
' {$PBASTC 2.s}
' {$sRAMP Bs2p}
' {$PBASTC 2.5}
```

```
trigger CON 13
scale CoN $0cd
i VAR Byte
ping PIN 0
terhubung ke p0
ping2 PIN 1
ping3 PIN 2
```

```
inir4 coN 4
echo4 CoN 3
'sensor sisi kiri
init5 CON 5
echo5 CON 6
ping4 PIN 7
```

```
isHigh CON 1
islow CON 0
pulse PIN 15
time VAR Word
```

```
rawtoin CON 889
rawtoom CON 2257
rawDist VAR Word
```

```

rawdistkanandepan VAR Word
rawdistkiridepan VAR Word
rawdistbelakaog VAR Word
distkanan VAR Word
distkiri VAR Word
'berjalan 1 arah 1 uS
'jarak=(echo tirire) / faktor konversi
'gunakan 74 inchi (73.14 uS per inchi)
'gunakan 29 untuk sentimeter ( 29.033 per 1 cm)
convfac CON 74 ' use inches
counter VAR Byte
status VAR Word
status=0
LOW 14
DO
    GOSUB get_flame
    GOSUB getjing ' baca jarak depan
    GOSUB getjingkanandepan
    GOSUB GETjingkiridepan
    coSUB getjingbelakang
    cosuB srfkanan
    GOSUB srfkiri
    DEBUG "Status api ? " : DEBUG DEC status, CR
    DEBUG "jarak depan " : DEBUG DEC rawDist **
    RawToCm,CR
    DEBUG "jarak kanan depan " : DEBUG DEC
    rawdistkanandepan ** rawtocm ,CR
    ' Jika tidak ada halangan
    IF rawDis! ** Ra\rToCm >15
    tOW 14 'malikan kipas
    ' maju terus
    Low 1-0
    LOW 11
    LOW 12
    HIGH 13
    PAUSE 10
    ENDIF
    IF rawDist ** RawToCm <=15
        HIGH 1-1 'berhenti
        HIGH 13
        HIGH 10
        HIGH 12
        PAUSE 30
        HIGH 14 'hidupkan kipas
    PAUSE 500.0
    st.atuS=0
    PAUSE 5

```

```

ENDIF
di depan
AND status=1 THEN
AND status=1 THEN

IF rawDist ** RawToCm <=15 AND status=0 THEN
LOW 14 'matikan
HIGH 0
    LOW 10 'mundur
HIGH ].].
LOW 12
tow 13
PAUSE 1000
    HIGH 11 'stop
    HIGH 13
    HIGH 10
    HIGH 12
EXIT
    ENDTF

LOOP
get_flame:'deteksi api
PULSIN puLse, 1, time
IF (time>o) THEN
    DEBUG DEC time, ' unit",CR
    st.atuS=1
    PAUSE 5
    EI,SE
    DEBUG "gak ada api" ,CR
    ENDTF
    PAUSE 5
    RETURN

get_ping :
    Ping = IsLow
    PULSOUT ping, trigger
    PULSTN ping, isHigh, rawDist
    rawDist=rawDist */Scale
    rawDist =rawDist/2
RETURN

get_pingkanandepan :
    ping2 =Is1,ow
    PULSOUT ping2, trigger
    PULSFN ping2, isHigh, rawDistkanandepan
    rawDistkanandepan=rawDistkanandepan */Scale
    rawDistkanandepan =rawDistkanandepan/2

```

```

RETURN

get_pingkiridepan :
    ping3 =Islow
    PULSOUT ping3, trigger
    PULSIN ping3, isHigh, rawDistkiridepan
    rawDistkiridepan=rawDistkiridepan */Scale
    rawDistkiridepan =rawDis tkiridepan/ 2
RETURN

get_pingbelakang :
    ping4 =Islow
    PULSOUT ping4, trigger
    PULSIN ping4, isHigh, rawDistbelakang
    rawDistbelakang=rawDistbelakang * /Scale
    rawDistbelakang=rawDistbelakang/ 2
RETURN

srfkanan:
    PULSOUT init4,5 ' 10us init pulse
    PULSIN echo4,1,distkanan , measure echo time
    distkanan=distkanan/convfac, convert to inch
RETURN

srfkiri:
    PULSOUT init5,5 ' 10us init pulse
    PULSIN echo5,1,distkiri ' measure echo time
    distkiri=distkiri/convfac ' convert to inch
RETURN

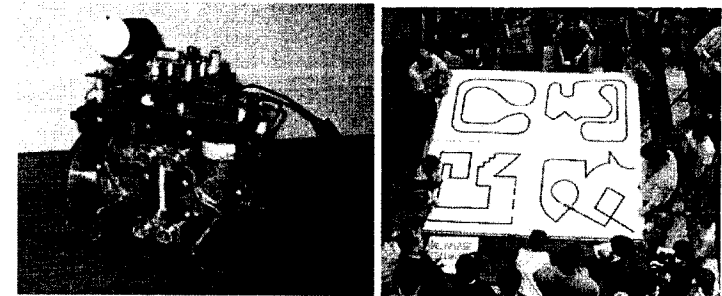
```

Jika ingin menambahkan sensor kompas, contoh program lengkap dari aplikasi kompas dapat dilihat di CD Program bernama Teskompas.bs2.

C. Tahap Uji Coba

Setelah kita men-*download* program ke *microcontroller* (otak robot), berarti kita telah siap melakukan tahapan terakhir dalam membuat robot, yaitu uji coba.

Untuk KRCI, uji coba dilakukan pada arena seluas sekitar 4×4 meter dan berbentuk seperti puzzle. Dalam arena KRCI ini, diletakkan lilin-lilin yang harus dipadamkan oleh robot cerdas pemadam api. Berikut adalah contoh gambar robot pemadam api Ted Larsorn di arena Kontes Robot Cerdas Indonesia (KRCI).



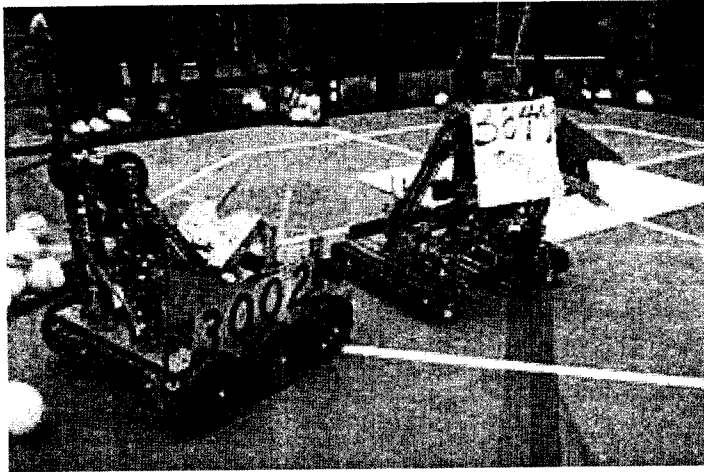
Gambar 4. Uji coba robot KRCI
(robotgames.net)

Untuk lomba robot KRI, dibutuhkan ruangan yang lebih besar, yaitu sekitar 15×15 meter. Dalam Kontes Robot Indonesia (KRI) 2008, misalnya, masing-masing robot harus meraih target (bola/kubus) yang diletakkan

Untuk KRCI, uji coba dilakukan pada arena seluas sekitar 4×4 meter dan berbentuk seperti puzzle. Dalam arena KRCI ini, diletakkan lilin-lilin yang harus dipadamkan oleh robot cerdas pemadam api.



di tempat yang tinggi. Jadi, sebuah robot harus bisa naik di atas robot yang lain untuk meraih target tersebut (seperti panjat pinang).



Gambar 5. Uji coba robot KRCI pada sebuah line
(robotgames.net)

D. Latihan

1. Kembangkan robot KRCI yang Anda buat agar mampu mengangkat benda menggunakan *gripper* dan meletakkan benda tersebut ke posisi *home*, seperti pada robot KRCI *expert battle*.
2. Kembangkan sistem gerakan robot di atas, agar dapat berjalan dengan tingkat kestabilan yang tinggi menggunakan Fuzzy logic atau ANFIS.
3. Cobalah kemampuan motor DC Vexta agar robot Anda dapat bergerak sangat cepat.

4. Cobalah kembangkan robot KRCI berbiaya murah menggunakan:
 - a. DT sense Flame Detector berbasis IR dengan jarak deteksi maksimal 40 cm.
 - b. DT Sense IR Proximity berbasis IR.



Daftar Pustaka

- Rusmadi, Dedy. 2005. *Aneka Rangkaian Elektronika Alarm dan Bel Listrik*. Bandung: Pioner Jaya.
- Team IE. 2006. *Panduan Praktis Mikrokontroler Keluarga AVR*. Surabaya: Innovative Electronic.
- Tim Pustena ITB. 2010. *Jago Membuat Robot; Dunia Komputer*. Bekasi: Tanpa Penerbit.
- Widodo, Budiharto. 2006. *Membuat Sendiri Robot Cerdas*. Jakarta: Elex Media Komputindo.
- . 2008. *Panduan Praktikum Mikrokontroler AVR ATmega16*, Jakarta: Elex Media Komputindo.

Website:

agfi.staff.ugm.ac.id

arduino.cc

cnt121.com

de-ca-de.co.tv

en.wikipedia.org

fahmizalee's.wordpress.com

himakomers.com

ikisuryo.blogdetik.com

ilmu-elektronika.co.cc

indo-ware.com

jhonsryo.blogspot.com

kangmuzaini.wordpress.com

ndoware.com

nextsys.web.id

parallax.com

pentrioloquist.wordpress.com

repository.unand.ac.id

rikosibigo.blogspot.com

robot-electronics.co.uk

robotic-explorer.com

robotika.blog.gunadarma.ac.id

sonoku.com

toko-elektronika.com

toko-robot.com

upm.comeze.com

widodo.com

