BAB III

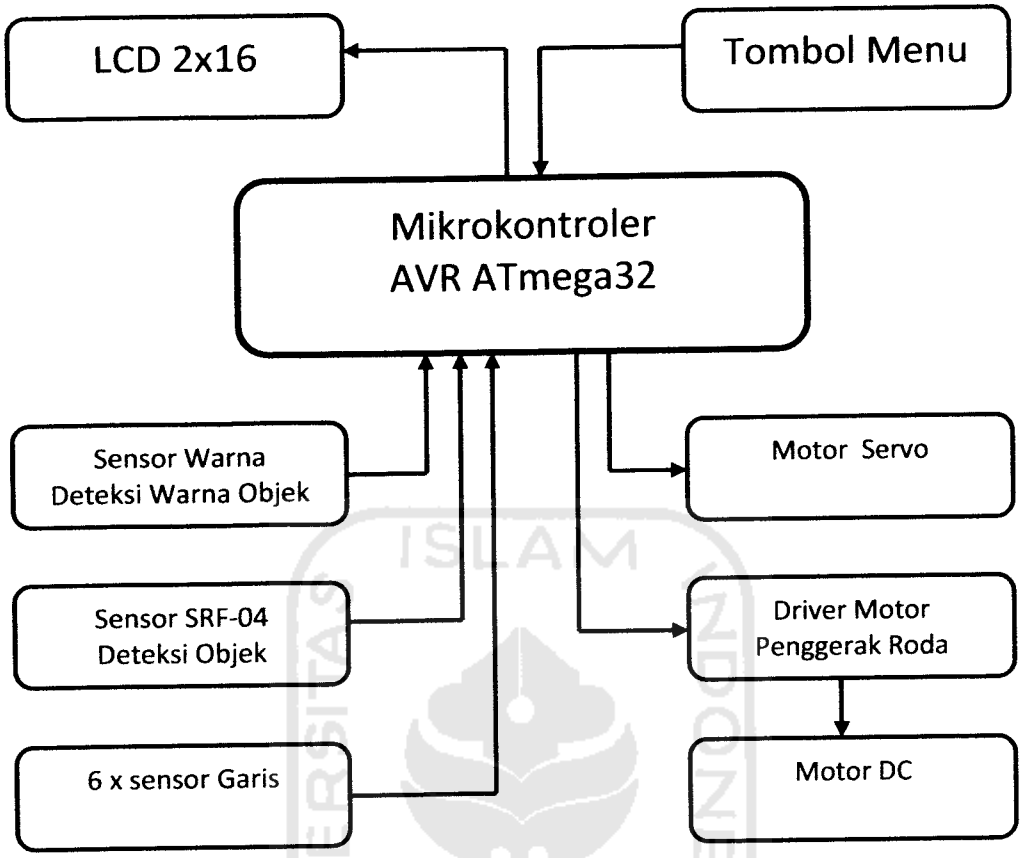
PERANCANGAN SISTEM

Pada bab ini akan dibahas tentang perancangan seluruh sistem robot yang dibuat baik *Hardware* maupun *Software*. Perancangan Hardware meliputi perancangan dan pembuatan mekanik robot. Perancangan *software* meliputi perancangan *flowchart* yang mendukung jalannya robot.

3.1. Blok Diagram

Perancangan secara umum dalam pembuatan sebuah robot pengangkut barang berdasarkan warna, yang terdiri dari sebuah sistem minimum AVR ATmega32 sebagai kendali utama dari berbagai masukan dari 6 pasang sensor garis, 1 buah sensor warna, 1 buah sensor SRF-04, 2 buah motor DC untuk penggerak roda, 2 motor servo untuk penggerakan gripper, serta menambahkan LCD 2x16 karakter sebagai penampil keadaan sistem dalam mengeksekusi program dalam pengolahan keadaan robot dalam program, sehingga kesalahan dan memperkecil nilai eror dalam pembacaan navigasi robot.

Berikut diagram blok yang digunakan dalam robot pengangkut barang berdasarkan warna :



Gambar 3.1. Blok Diagram Sistem

Dari diagram blok dapat kita perhatikan pada setiap komponen memiliki fungsi masing-masing, dan dikontrol pada mikrokontroler sebagai pengendali semua input dan output komponen blok diagram. Fungsi secara umum dari bagian-bagian blok diagram adalah sebagai berikut :

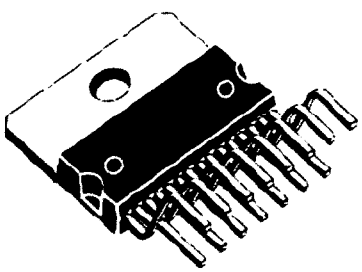
a. Mikrokontroler ATmega32 sebagai otak pengendali dari semua blok diagram yang terhubung, sinyal informasi yang masuk ke mikrokontroler diolah untuk mendapatkan intruksi output yang diinginkan.

b. Sensor jarak SRF-04 berfungsi untuk mengukur jarak antara robot dengan objek yang akan diangkat dan pindahkan. Dengan adanya sensor jarak maka robot dapat mendeteksi keberadaan objek yang akan dijepit.

c. Sensor warna menggunakan *photodioda* yang dikombinasikan dengan LED *super bright* merah, hijau dan biru atau yang umum disebut RGB. Pemilihan sensor menggunakan *photodioda* memiliki respon yang cukup cepat jika dibandingkan dengan LDR *(light Dependen Resistor).* Semakin cepat pembacaan dengan memberikan intruksi berulang -ulang akan menghasilkan pembacaan yang akurat.

d. Sensor garis menggunakan *photodioda* yang dipasangkan dengan *InfraRed* sebagai sumber cahaya untuk dipantulkan pada arena garis, sehingga robot dapat berjalan dengan sebuah informasi pantulan cahaya *InfraRed.*

e. Driver motor roda sebagai pengendali atau penguat untuk mengontrol motor dc sebagai penggerak roda robot. Driver ini menggunakan jenis driver L298D yang memiliki 2 buah output motor dan 4 buah input pengendali arah dan 2 buah pengendali pwm :



Tabel 3.2. Bentuk fisik IC L298

Tabel. 3.1. Pin kontrol IC L298

Pin	Input	Output	Input	Output
Input A	1	Motor maju	0	Motor Mundur
Input A	0		1	
Enable A	PWM Motor A	Nilai PWM mengatur kecepatan putaran		
Enable B	PWM Motor B			
Input B	1	Motor maju	0	Motor mundur
Input B	0		1	

f. Motor servo mengunakan 2 buah yang diantaranya untuk servo griper dan servo angkat lengan robot. Untuk servo tidak membutuhkan driver pengendali, karena didalam box motor sevo sudah dilengkapi komponen dan gear khusus buatan pabrik. Komunikasi yang sangat mudah dan dengan daya yang cukup rendah motor servo sangat cocok digunakan untuk lengan robot dengan memanfaatkan sudut putaran.

3.2. Perancangan Hardware Robot

Robot pengangkut barang berdsarkan warna didesain sesederhana mungkin agar robot tidak terlihat rumit dalam melakukan kerja. Perancangan robot ini didukung oleh rangkaian-rangkaian elektris yang membantu kinerja mikrokontroler untuk pengendali utama untuk semua sistem. Elcktronik yang dirancang menyesuaikan dengan mekanik robot yang telah dibuat sehingga tidak menghabiskan tempat dalam pemasangannya, hal ini dimaksudkan agar robot terlihat sederhana dan rapi dalam perancangannya.

Gambar 3.3. Mekanik Robot

3.2.1. Perancangan Mekanik Robot

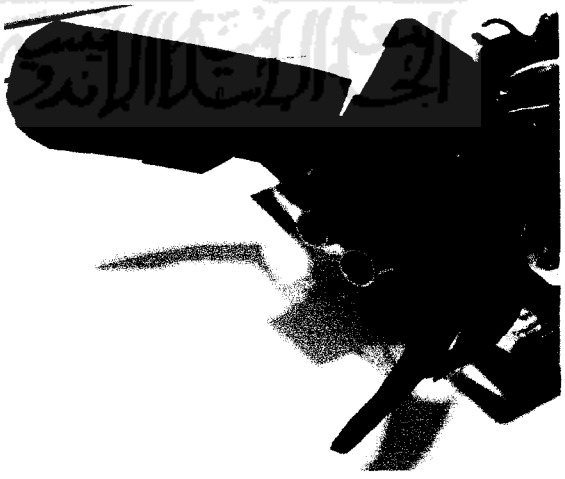
Perancangan mekanik meliputi beberapa bagian yaitu perancangan dimensi robot, perancangan mekanik roda, perancangan lengan griper.

3.2.1.1. Mekanik Roda Robot

Robot memiliki 2 buah roda yang masing-masing roda menggunakan motor dc gearbox, pada bagian depan menggunakan roda ruble atau roda bebas, agar gerakan lebih bebas kesegala arah dan tidak membebani motor penggerak. Roda menggunakan lingkaran roda RC dengan mengganti karet lapisan luar, hal ini agar robot tidak terjadi selip saat melakukan pengereman.

3.2.1.2. Perancangan Lengan Griper

Lengan griper menggunakan 2 buah motor servo standard dengan putaran 180 derajad, 1 buah untuk bagian griper jepit dan 1 buah lagi untuk mengangkat lengan griper. Saat barang yang akan dipindahkan dijepit bagian lengan griper, selanjutnya agar pemindahan barang tidak mengenai lantai maka barang harus diangkat menggunakan servo angkat pada bagian lengan griper.



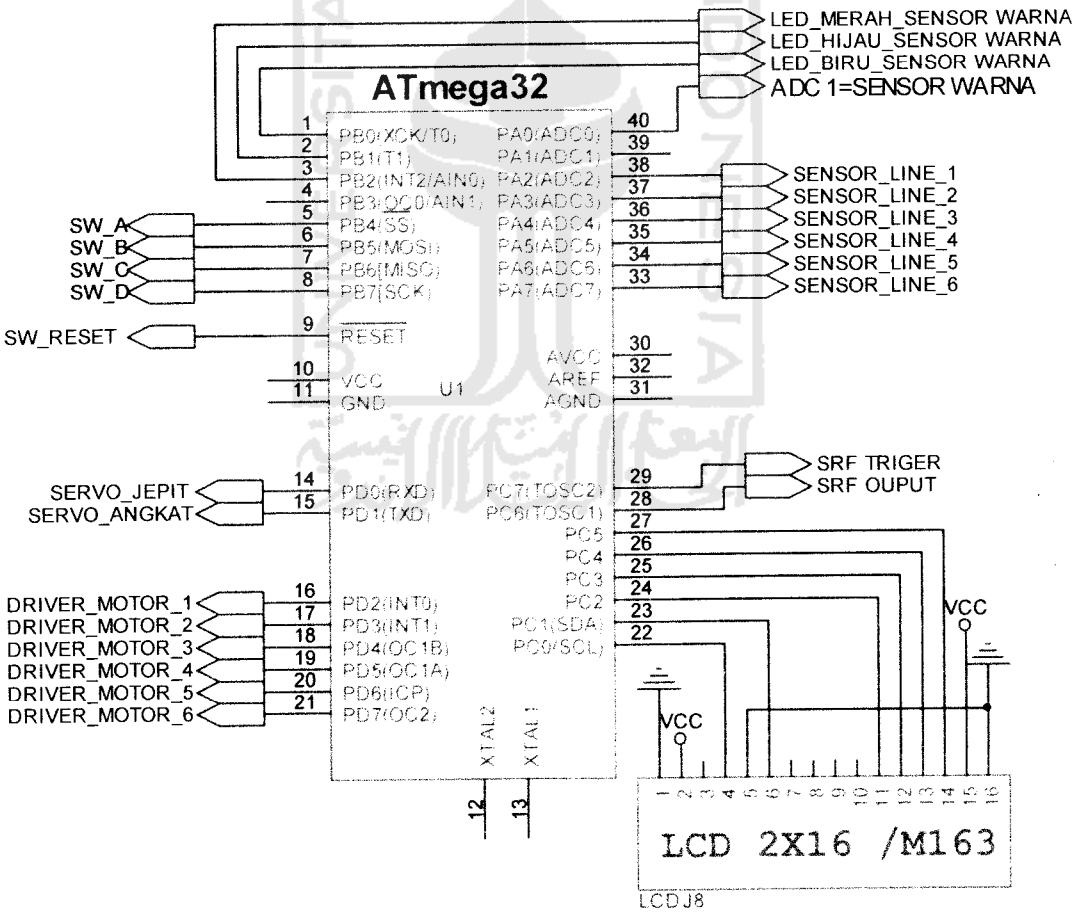
Gambar 3.4. Lengan Griper

3.2.2. Perancangan Elektronik

Perangcangan elektronik meliputi pembuatan sistem minimum AVR ATmega32, modul sensor warna, Sensor jarak SRF04, dan power supply untuk semua catu daya elektronik yang digunakan.

3.2.2.1. Sistem Minimum ATmega32

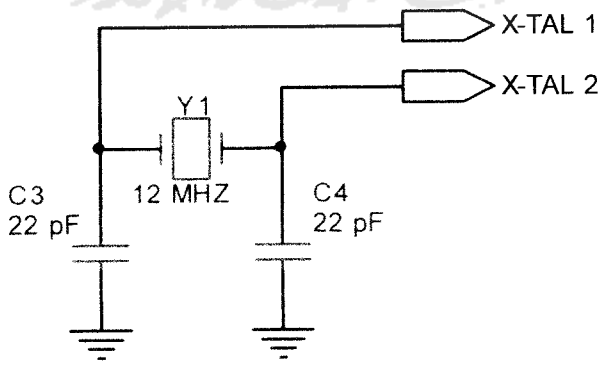
Pengendalian pada robot yaitu pada bagian sistem minimum ATmega32, yang terdiri dari rangkaian osilator, power On/Of, reset, dan beberapa port I/O.



Gambar 3.5. Skematik Sistem minimum AVR

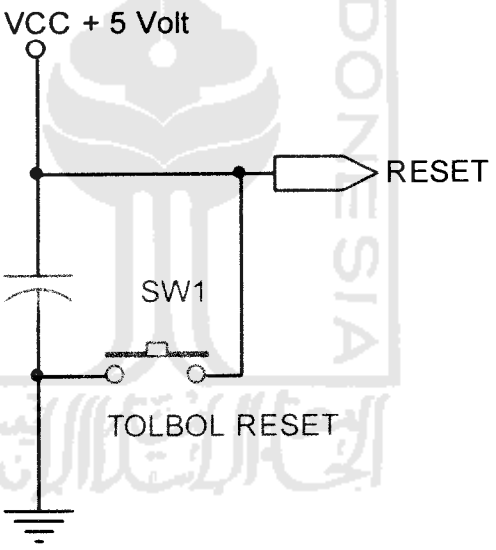
Keterangan gambar :

Sensor garis dihubungakan pada portA.2 s/d portA.7 yang masing-masing sensor dibaca menggunakan ADC internal mikrokontroler, sensor warna dihubungakan pada port A.0 dibaca menggunakan ADC chanel1. 3 buah led untuk sensor warna dihubungkan pada portB.0 – portB.2, 4 buah tombol menu untuk navigasi setingan program dihubungkan pada portB.4 – portB.7 dengan input active low (0). Motor servo dihubungkan pada portD.0 dan portD.1 untuk kedua servo mendapatkan masing-masing pin mikrokontroler untuk mengendalikan arah putaran motor servo. 6 buah pin untuk mengontrol driver motor DC dengan output PWM dan kontrol arah putaran. LCD dihubungkan pada Port C.0 =RS, Port C.2 = Enable, Port C.5 = D4 lcd, port C.5 =D5 lcd, port C.6= D6 lcd, port C.7 = D7 lcd. Sistem minimum membutuhkan osilator external sebagai pembangkit clock external yang berfungsi pembangkit dalam mengeksekusi intruksi program yang dituliskan dimemori program. Osilator eksternal menggunakan 11.059200 Hz dengan kapasitor masing 22 pF



Gambar 3.6. Clock Extal dan Kapasitor

Rangkaian *Reset* berfungsi untuk menjaga agar pada pin RST pada mikrokontroler selalu berlogika rendah saat mikro mengeksekusi program. Mikro direset pada transisi dari tegangan *low* ke tegangan *high.* Pada saat pin *reset* bernilai 1 (satu) saat pengisian kapasitor dan bernilai 0 (nol) saat kapasitor penuh. Pada saat sumber diaktifkan kapasitor terhubung singkat sehingga arus mengalir dari VCC ke pin reset sehingga pin reset bernilai 1 (satu) atau *high*, kemudian kapasitor mengisi ulang sampai nilai tegangan sama dengan VCC. Dengan demikian pin *reset* akan berlogika 0 (nol).



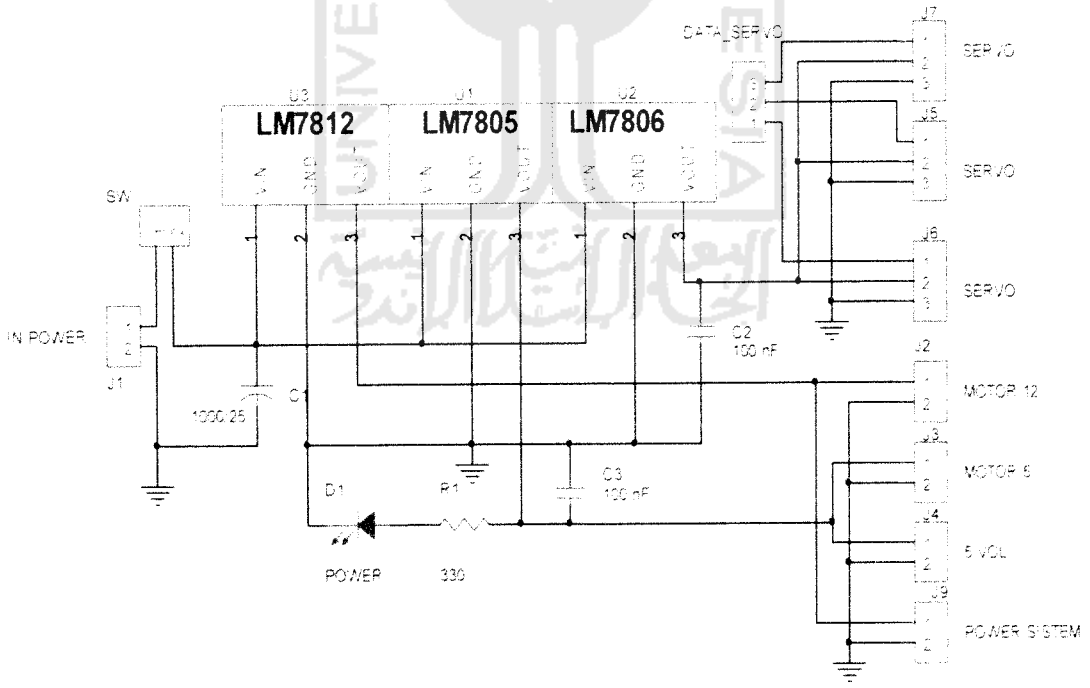
Gambar 3.7. Tombol Reset

Pada tugas akhir ini, mikrokontroler digunakan sebagai pengolah data utama yang memiliki fungsi pengolah data input dan output. Pin-pin mikrokontroler

memiliki fungsi khusus yang mendukung kinerja robot dalam menyelesaikan tugasnya.

3.2.2.2. Perancangan Power Supply

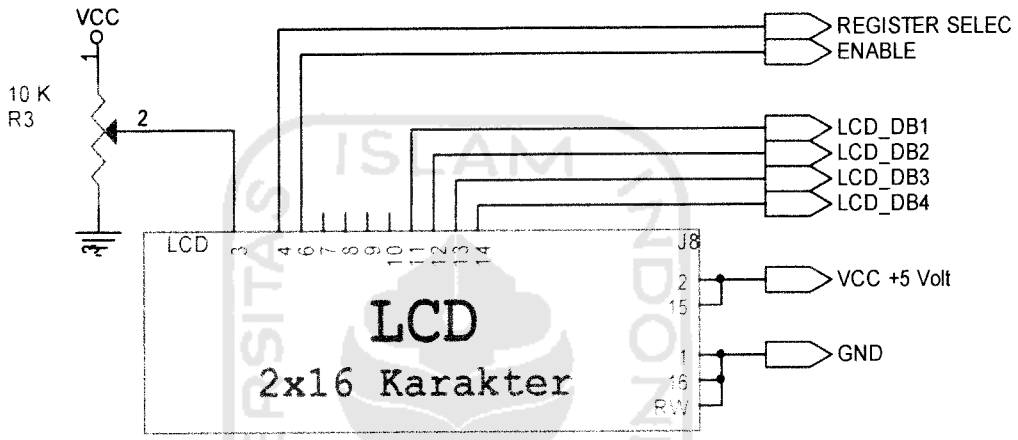
Robot yang dirancang mendapatkan power dari baterai, baterai sumber catu daya utama untuk mensupply semua bagian elektronik. Baterai yang digunakan adalah jenis lippo dengan besar tegangan 11.1 volt 1000 mA, Tegngan sebesar 11.1 volt harus dibagi ke bagian-bagian rangkaian lainnya. Power supply yang dirancang menggunakan IC regulator LM 7805 untuk menghasilkan tegangan 5 Volt untuk menyupply sensor dan sistem minimum, LM 7806 yang menghasilkan 6 Volt digunakan untuk menyupply tegangan motor servo.



Gambar 3.8. Rangkaian Power Supply

3.2.2.3. Rangkaian Penampil LCD

LCD adalah piranti elektronik untuk menampilkan hurup, angka dan simbol, lcd menggunakan 2x16 karakter atau memiliki 2 baris dan 16 kolom untuk tiap baris nya.



Gambar 3.9. LCD 2 x 16 Karakter

Lcd dikomunikasikan 4 buah jalur data dan 2 buah jalur kontrol pemilihan register dan enable. Pin control dan pin data dihubungkan di potC.

3.2.2.4. Rangkaian Sensor Garis

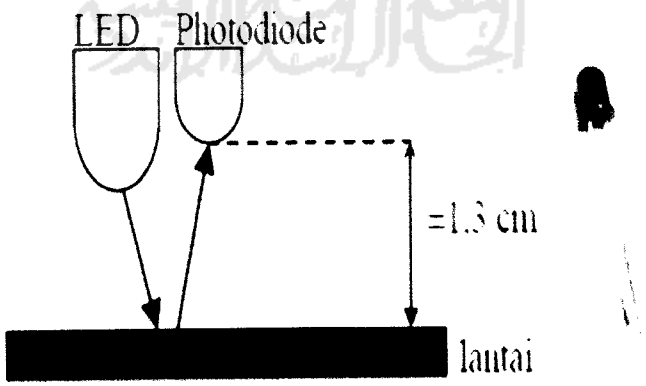
Sensor untuk penjejak garis digunakan 6 buah *photodioda* yang dipasangkan dengan *InfraRed.* Pada dasarnya *photodioda* mengalamai perubahan hambatan dikedua kutubnya bila terjadi perubahan intensitas cahaya yang

mengenainya. Perbandingan *photodioda*, *LDR* , dan *phototransistor* ditunjukkan pada tabel :

Tabel. 3.2. Perbandingan Sensor

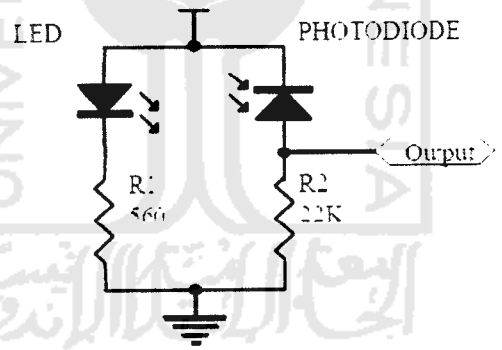
NO	Materi Perbandingan	LDR	Phototransistor	Prhotodioda
1	Perubahan Hambatan	Sedang	**Besar**	**Besar**
2	Perubahan Kenaikan	**Linier**	Tidak Linier	**Linier**
3	Penyerapan Cahaya	Tidak Fokus	**Fokus**	**Fokus**
4	Peka terhadap Ir	Tidak	**ya**	**ya**

Photodioda berfungsi sebagai pemandu jalannya robot dengan cara membaca perubahan intensitas cahaya yang mengenai lantai arena robot. Untuk dapat membedakan garis dan arena dasar maka dibutuhkan cahaya yang cukup yang bersumber dari *InfraRed* yang nantinya akan dipancarkan pada lantai arena dan hasil pantulan akan memberikan cahaya sensor *photodioda*. *Photodioda* dipasang sejajar dan berpasangan dengan *InfraRed* dan ke duanya tegak lurus terhadap lantai arena. Untuk lebih jelasnya dapat dilihat pada Gambar :



Gambar 3.10. Posisi sensor terhadap lantai

Pada prinsipnya perubahan hambatan pada *photodioda* akan dikombinasikan dengan rangkaian pembagi tegangan dan akan memberikan nilai tegangan yang bervariasi. Pada rangkaian pembagian tegangan tersebut, *photodioda* akan disusun seri dengan resistor referensi. Nilai hambatan tertinggi dari *photodioda* harus lebih besar dari nilai resistor referensi, dan nilai hambatan terendah dari *photodioda* harus lebih kecil dari resistor referensi agar dapat dicapai perubahan tegangan yang signifikan dan bagus. Perubahan intensitas cahaya pada arena juga sangat terpengaruh oleh cahaya lingkungan atau cahaya sekitar. Agar terhindar dari cahaya interferensi dari luar yang dapat mengganggu maka rangkaian sensor perlu diberi pelindung untuk mengurangi gangguan tersebut.

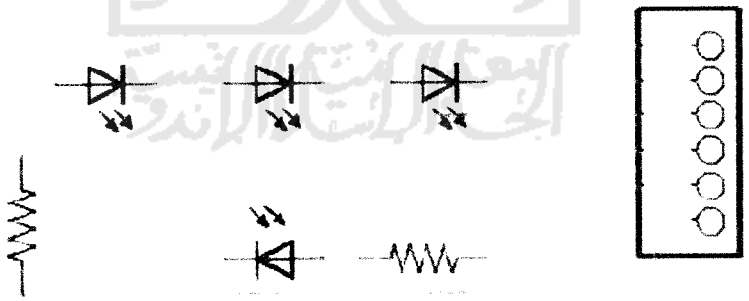


Gambar 3.11. Rangkaian sensor garis

Dari perubahan nilai tegangan pada rangkaian pembagi tegangan pada sensor maka diperoleh perubahan yang cukup untuk dibaca ADC internal pada mikrokontroler dengan resolusi ADC 10-bit sehingga nilai tegangan output sensor yang berada pada tegangan antara 0 volt hingga 5 volt akan dibaca dan dikonversikan bilangan desimal ADC antara 0 hingga 1023.

3.2.2.5. Rangkaian Sensor Warna

Pada dasarnya sensor warna ini hampir sama prinsip kerjanya dengan sensor penjejak garis. Untuk rangkaian penerima digunakan *photodiode* untuk menangkap intensitas cahaya yang berbeda-beda dan proses pembagian tegangannya sama dengan proses sensor penjejak garis. Sensor warna terdiri dari 1 buah *photodiode* dan 3 buah LED *superbright* yang terdiri dari warna primer RGB (*Red, Green, Blue*). Ketiga LED warna tersebut akan menyala bergantian dan diatur oleh pin mikrokontroler dengan proses sistem bergantian(*scanning*). Keluaran dari rangkaian pembagi tegangan sensor akan dibaca oleh pin ADC internal mikrokontroler. Setiap warna LED yang dipancarkan akan memantulkan intensitas cahaya yang berbeda-beda pada setiap warna objek. Pada skripsi ini objek hanya akan diberi tiga warna objek yaitu warna merah, hijau, dan biru.



Gambar 3.12. Rangkaian sensor warna

3.2.2.6. Tombol Menu

Tombol yang digunakan dalam robot ini yaitu tombol *push button.* Ada 4 buah tombol yang terpasang dan terhubung pada PORTB. Tombol ini digunakan sebagai pemilih menu pada layar penampil LCD 2x16. Selain digunakan sebagai tombol mulai (*start*) pada robot, tombol ini juga digunakan untuk mengatur nilai konstanta-konstanta kecepatan motor nantinya.

Empat tombol ini terhubung dengan mikrokontroler dengan mode aktif low (0). Rangkaian 4 buah tombol dapat dilihat pada Gambar :

Gambar 3.13. Rangkaian Tombol Menu

3.2.2.7. Baterai

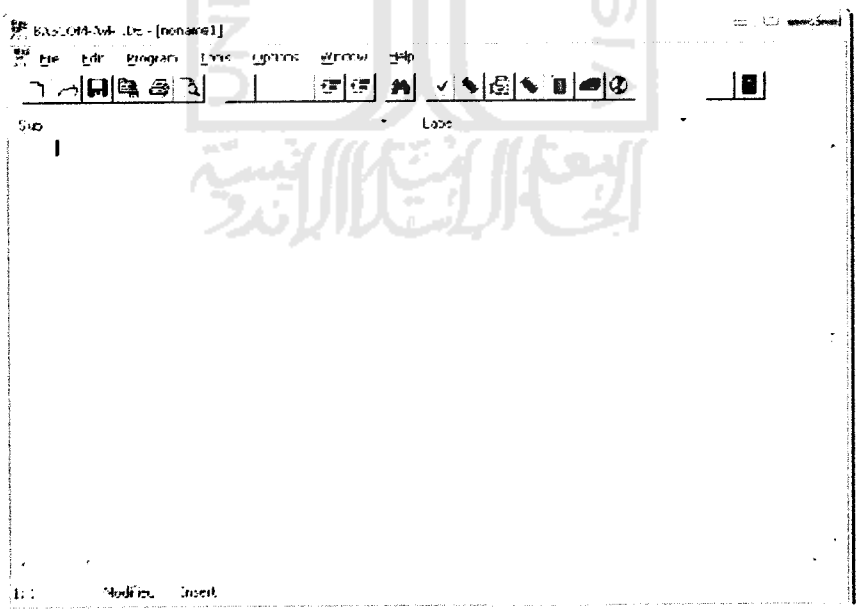
Baterai merupakan bagian yang sangat penting karena digunakan sebagai sumber energi bagi robot ini. Baterai yang digunakan adalah baterai jenis Litium Polymer (Li-Po) yang dapat diisi ulang (*charging*). Baterai jenis ini memiliki kemampuan penyimpanan energi yang cukup lama dengan arus yang cukup besar

3.2.2.8. Perancangan Perangkat Lunak (*Software)*

Perancangan perangkat lunak berisi tentang rancangan program mikrokontroler dan diagram alir (*flowchart*) tiap proses kerja robot untuk setiap bagiannya meliputi berbagai proses, antara lain:

- Pembacaan sensor garis dan kalibrasi
- Pengenalan objek dan proses pengangkutan
- Proses kontrol PWM motor
- Proses pemilihan jalan robot (*mapping*).

Pemrograman mikrokontroler dibuat menggunakan program *Basic Compiller* (*BASCOM*) AVR versi 1.11.9.5. Hasil pemrograman dari *BASCOM* AVR di-*compile* menjadi file berekstensi *.hex, yang akan di-*download* ke mikrokontroler ATMega32 dengan software PonyProg2000 atau sejenisnya.

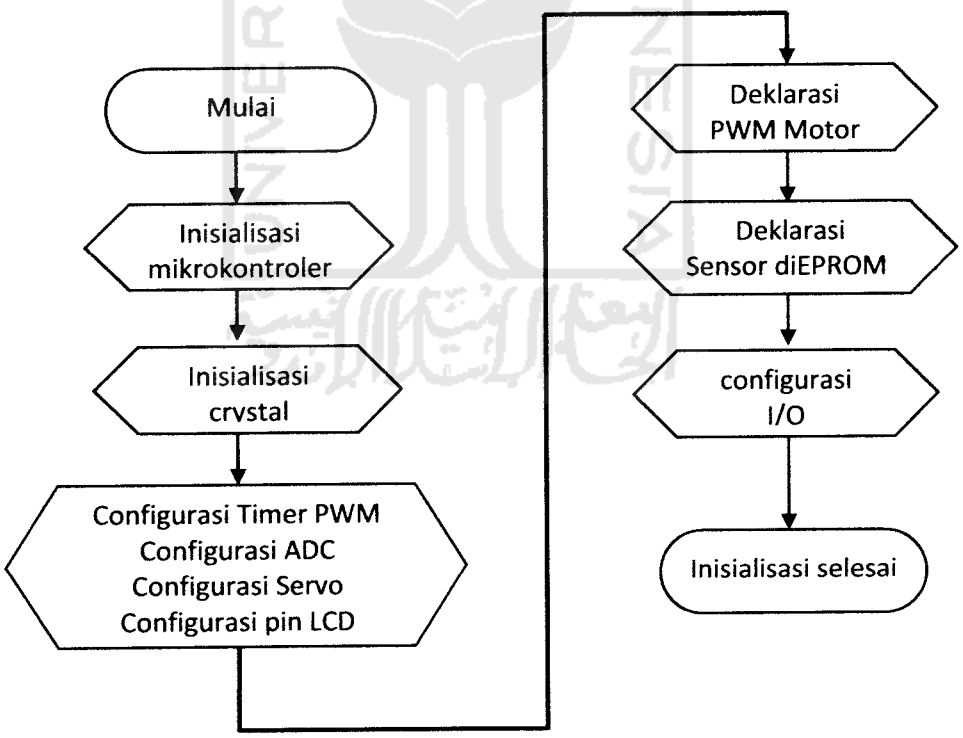


Gambar 3.15. Lembar Kerja pembuatan program

Dalam memulai penulisan program ada beberapa hal yang harus definisikan atau dideklarasikan. Yang perlu didefinisikan antara lain:

- Jenis mikrokontroler yang digunakan
- Detak *clock crystal* pada mikrokontroler
- Konfigurasi fasilitas mikrokontroler seperti *timer*, adc, servo, dan konfigurasi LCD.
- Deklarasi *variable* yang digunakan.
- Pengaturan PORT sebagai masukan, keluaran, atau fungsi khusus.
- Penginisialisasi nama dan konstanta.

Rancangan program awal robot pengakut barang yang terdiri dari deklarasi *variable* dan konfigurasi mikrokontroler.



Gambar 3.16. Diagram alir proses inisialisasi mikrokontroler

Listing program kofigurasi jenis mikrokontroler dan *clock crystal* yang dipakai adalah sebagai berikut:

```
$regfile = "m32def.dat"
$crystal = 11,059200
```

Konfigurasi fasilitas khusus pada mikrokontroler dan sintak pembantu dari *BASCOM* AVR dapat dituliskan pada program berikut ini:

```
Config Timer1= Pwm , Pwm = 8 , Compare A Pwm = Clear Down
Config Compare B Pwm = Clear Down , Prescale = 64
Config Servos=2, Servo1=Portd.0,Servo2=Portd.1,Reload = 10
Config Adc= Single,Prescaler=Auto
Config Lcdpin=Pin,Db4=Portc.3,Db5=Portc.2,Db6=Portc.4
Config Lcdpin=Pin,Db7=Portc.5,E=Portc.1,Rs=Portc.0
Config Lcd = 16 * 2
Deflcdchar 0 , 4 , 14 , 31 , 31 , 14 , 4 , 32 , 32
Cls
Cursor Off
```

Digunakan Timer1 sebagai pembangkit sinyal PWM dengan resolusi 8 bit, pembagian *prescale* = 64 untuk mendapatkan frequensi sebesar ± 100 Hz. Konfigurasi untuk pin motor servo dan jumlah servo yaitu 2 buah. ADC yang digunakan adalah *mode single* dengan *prescaler auto* untuk mendapatkan waktu konversi ADC tercepat. Konfigurasi LCD untuk setiap pin dan jenis LCD 2 x 16.

Untuk deklarasi *variable* digunkan perintah Dim diikuti tipe datanya. Deklarasi untuk beberapa *variable* proses kalibrasi sensor dituliskan pada program berikut ini:

```
Dim X As Byte , Kecepatan As Byte
Dim Ep_kp As Eram Byte , Ep_kd As Eram Byte , Ep_ki As Eram
Byte
Dim Kp As Byte , Kd As Byte , Ki As Byte
Dim Eror_lalu As Integer , Eror As Integer ,
```

```
Dim Motkan As Integer , Motkir As Integer
Dim P As Integer,D As Integer,I As Integer, Z_i As Integer
Dim Temp_kan As Word , Temp_kir As Word
Dim Sp_kan As Integer , Sp_kir As Integer
Dim Delta_eror As Integer , Pid As Integer

Dim Dat_warna As Word ,Data_srf As Word
Dim Warna_merah As Word,Warna_biru As Word,Warna_hijau As Word
Dim Perjalan As Byte
Dim Set_simpang As Byte , Ep_simpang As Eram Byte
Dim Lines As Byte,X_max_e As Eram Byte,
Dim X_max As Byte,Titip As Byte
Dim Ep_ln1 As Eram Word , Ep_ln2 As Eram Word ,
Dim Ep_ln3 As Eram Word ,
Dim Ep_ln4 As Eram Word , Ep_ln5 As Eram Word ,
Dim Ep_ln6 As Eram Word ,
Dim Ep_ln7 As Eram Word , Ep_ln8 As Eram Word
Dim Ref_ln1 As Word , Ref_ln2 As Word , Ref_ln3 As Word ,
Dim Ref_ln4 As Word ,
Dim Ref_ln5 As Word , Ref_ln6 As Word , Ref_ln7 As Word ,
Dim Ref_ln8 As Word
Dim Dat_ln1 As Word , Dat_ln2 As Word ,
Dim Dat_ln3 As Word , Dat_ln4 As Word , Dat_ln5 As Word ,
Dim Dat_ln6 As Word ,
Dim Dat_ln7 As Word , Dat_ln8 As Word
Dim Hi_ln1 As Word , Hi_ln2 As Word , Hi_ln3 As Word ,
Dim Hi_ln4 As Word ,
Dim Hi_ln5 As Word , Hi_ln6 As Word , Hi_ln7 As Word ,
Dim Hi_ln8 As Word
Dim Lo_ln1 As Word , Lo_ln2 As Word , Lo_ln3 As Word ,
Dim Lo_ln4 As Word ,
Dim Lo_ln5 As Word , Lo_ln6 As Word , Lo_ln7 As Word ,
Dim Lo_ln8 As Word ,
Dim Waktu_muter As Word
```

Konfigurasi pin mikrokontroler sebagai masukan atau keluaran diatur pada register DDRX dengan nilai 0 sebagai masukan dan nilai 1 sebagai keluaran, programnya sebagai berikut:

```
Ddrc.6 = 0
Ddrc.7 = 1
Ddrb = &B11100000
Portb = &HFF
Ddra.1 = 1
```

```
Porta.1 = 1
Portd = &HFF
Portc = 255
```

3.2.2.9. Perancangan dan kalibrasi sensor

Perubahan intensitas cahaya yang mengenai *photodiode* akan ditandai dengan perubahan nilai tegangan pada keluaran sensor hasil pembagi tegangan. Nilai tegangan ini akan dibaca oleh ADC 10-bit internal mikrokontroler dan dikonversikan menjadi nilai bilangan desimal pada sebuah *variable* antara 0 hingga 1023, nilai konversi dapat dihitung dari persamaan :

$$\text{Nilai ADC} = \frac{Vin}{VReff} x1024$$

Mikrokontroler pada robot ini konfigurasi nilai referensi ADC diambil dari pin AVCC yaitu 5 volt sehingga nilai VReff adalah 5 volt. Untuk membaca dan mengambil data ADC digunakan perintah Getadc(channel). Sub program dalam pembacaan ADC adalah sebagai berikut:

```
Baca_adc:
Start Adc
Dat_ln1 = Getadc(2)
Dat_ln2 = Getadc(2)
Dat_ln3 = Getadc(3)
Dat_ln4 = Getadc(4)
Dat_ln5 = Getadc(5)
Dat_ln6 = Getadc(6)
Dat_ln7 = Getadc(7)
```

```
Dat_ln8 = Getadc(7)
Stop Adc
        Return
```

Dari pengambilan data sensor tersebut kemudian nilai ADC dibandingkan dengan nilai referensi untuk mendapat logika sensor apakah itu hitam atau putih dan data untuk setiap sensor diubah menjadi nilai per-bit. Proses konversi ini digunakan untuk mempermudah pembacaan logika sensor dan proses ini untuk menggantikan fungsi *comparator*. Jumlah sensor garis pada robot ini adalah enam buah *photodioda* dan ADC yang terpakai adalah *channel* 0 hingga 5 dan hasil konversi akan mengubah nilai tiap bit *variable* Lines. Program untuk konversinya adalah sebagai berikut:

```
Conversi:
Gosub Baca_adc
If Dat_ln1 > Ref_ln1 Then Ln0 = 1
If Dat_ln1 < Ref_ln1 Then Ln0 = 0
If Dat_ln2 > Ref_ln2 Then Ln1 = 1
If Dat_ln2 < Ref_ln2 Then Ln1 = 0
If Dat_ln3 > Ref_ln3 Then Ln2 = 1
If Dat_ln3 < Ref_ln3 Then Ln2 = 0
If Dat_ln4 > Ref_ln4 Then Ln3 = 1
If Dat_ln4 < Ref_ln4 Then Ln3 = 0
If Dat_ln5 > Ref_ln5 Then Ln4 = 1
If Dat_ln5 < Ref_ln5 Then Ln4 = 0
If Dat_ln6 > Ref_ln6 Then Ln5 = 1
If Dat_ln6 < Ref_ln6 Then Ln5 = 0
If Dat_ln7 > Ref_ln7 Then Ln6 = 1
If Dat_ln7 < Ref_ln7 Then Ln6 = 0
If Dat_ln8 > Ref_ln8 Then Ln7 = 1
If Dat_ln8 < Ref_ln8 Then Ln7 = 0
Return
```

Nilai referensi untuk tiap sensor didapat dari proses perekaman data pada saat kalibrasi. Pada saat proses kalibrasi mikrokontroler akan menyimpan nilai tertinggi ADC (lantai putih) dan nilai terendah ADC (lantai hitam). Nilai tengah

diantara nilai tertinggi dan terendah akan digunakan sebagai nilai referensi yang akan digunakan sebagai pembanding diproses konfersi. Persamaan untuk mencari nilai referensi ditunjukan pada persamaaan :

$$\text{Nilai referensi ADC} = \frac{ADC\ Tertinggi + ADC\ terendah}{2}$$

Proses pengambilan data saat proses kalibrasi bisa dilakukan secara otomatis atau manual. Dari pengambilan data tersebut kemudian didapat nilai referensi dari persamaan diatas nilai referensi tersebut kemudian akan disimpan dalam EEPROM sehingga nilai tersebut tidak akan hilang walaupun mikrokontroler dimatikan suplai energinya. Nilai EEPROM akan selalu dipindah ke RAM mikrokontroler saat robot awal dihidupkan. Proses perekaman atau kalibrasi dapat dijelaskan lebih lengkap dengan melihat program berikut ini:

```
Reff_step:
Cls
Lcd "<oke>"
Locate 2 , 1
Lcd " cari putih "
Waitms 300
Do

Gosub Baca_adc
Loop Until Sw_a = 0
Hi_ln1 = Dat_ln1
Hi_ln2 = Dat_ln2
Hi_ln3 = Dat_ln3
Hi_ln4 = Dat_ln4
Hi_ln5 = Dat_ln5
```

```
Hi_ln6 = Dat_ln6
Hi_ln7 = Dat_ln7
Hi_ln8 = Dat_ln8


Cls
Locate 1 , 12
Lcd "<oke>"
Locate 2 , 1
Lcd " cari hitam "
Waitms 300
Do

Gosub Baca_adc
Loop Until Sw_d = 0
Lo_ln1 = Dat_ln1
Lo_ln2 = Dat_ln2
Lo_ln3 = Dat_ln3
Lo_ln4 = Dat_ln4
Lo_ln5 = Dat_ln5
Lo_ln6 = Dat_ln6
Lo_ln7 = Dat_ln7
Lo_ln8 = Dat_ln8
```

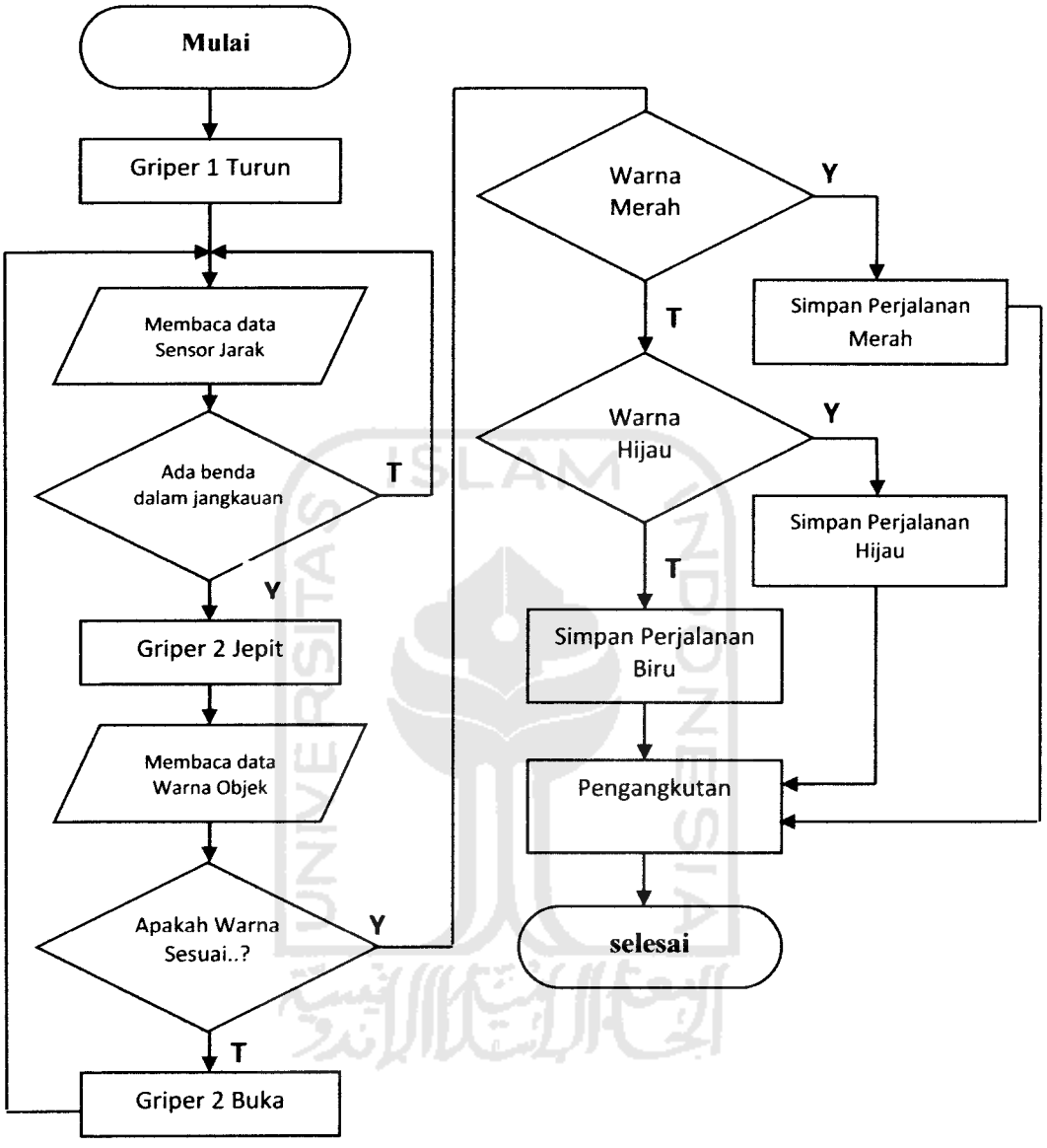
Setelah melakukan perekaman data tertinggi dan terendah maka dari kedua data tersebut akan dicari nilai referensi yang nantinya akan disimpan pada *variable* EEPROM. *Listing* programnya sebagai berikut:

```
Ref_ln1 = Hi_ln1 + Lo_ln1
Ref_ln2 = Hi_ln2 + Lo_ln2
Ref_ln3 = Hi_ln3 + Lo_ln3
Ref_ln4 = Hi_ln4 + Lo_ln4
Ref_ln5 = Hi_ln5 + Lo_ln5
Ref_ln6 = Hi_ln6 + Lo_ln6
Ref_ln7 = Hi_ln7 + Lo_ln7
Ref_ln8 = Hi_ln8 + Lo_ln8


Ref_ln1 = Ref_ln1 / 2
Ref_ln2 = Ref_ln2 / 2
Ref_ln3 = Ref_ln3 / 2
Ref_ln4 = Ref_ln4 / 2
```

```
Ref_ln5 = Ref_ln5 / 2
Ref_ln6 = Ref_ln6 / 2
Ref_ln7 = Ref_ln7 / 2
Ref_ln8 = Ref_ln8 / 2

Ep_ln1 = Ref_ln1
Ep_ln2 = Ref_ln2
Ep_ln3 = Ref_ln3
Ep_ln4 = Ref_ln4
Ep_ln5 = Ref_ln5
Ep_ln6 = Ref_ln6
Ep_ln7 = Ref_ln7
Ep_ln8 = Ref_ln8
```

3.2.2.10. Pengenalan Objek dan proses pengangkutan

Dalam proses pengenalan objek, robot diusahakan dapat mengenali ada tidaknya keberadaan objek. Robot juga akan mencoba mengenali warna objek untuk proses pemetaan peletakan jalur target. Objek pada awal pengambilan sudah diatur letaknya dan jarak antara awal robot mulai (*home*) berjarak kurang lebih 25 cm dari titik peletakan objek. Robot akan berhenti di awal pengangkutan dan titik target jika sensor garis membaca sebuah simpang pertigaan atau semua sensor mengenai garis hitam. Untuk proses pengenalan objek dapat dijelaskan pada diagram alir



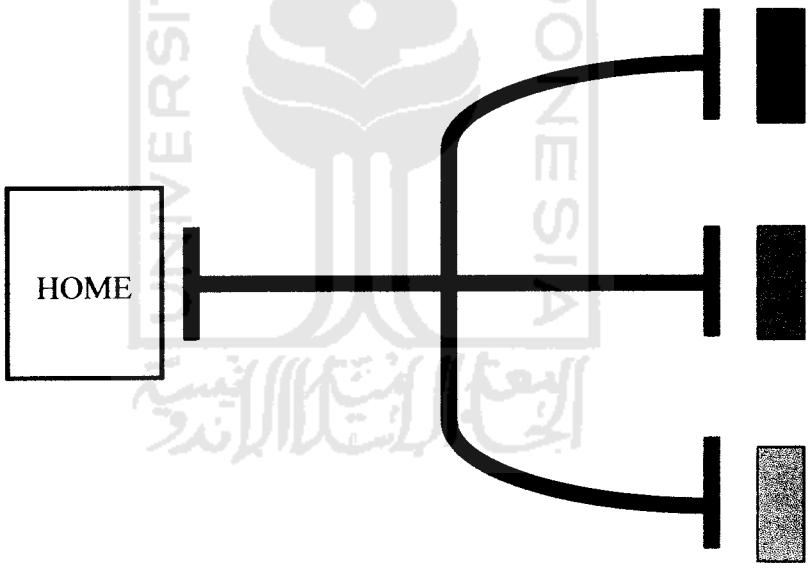
Gambar 3.17. Pengenalan Objek dan proses pengangkutan

```
Scan_objek:
Gosub Grip_turun
Gosub Jepit_buka

Baca_jarak:
   Init_srf = 1
   Waitus 10
   Init_srf = 0
   Data_srf = 0
   Do
      Waitus 1
      Incr Data_srf
      If Echo_srf = 1 Then Exit Do
      Loop Until Data_srf = 1000
     Data_srf = 0

Do
Waitus 1
Incr Data_srf
If Echo_srf = 0 Then Exit Do
If Data_srf > 3000 Then Exit Do
Loop
Data_srf = Data_srf / 10
Return

Baca_warna_jarak:
     Gosub Baca_jarak
     Start Adc
     Dat_warna = Getadc(0)
     Stop Adc
     Dat_warna = Dat_warna
     Return

Baca_warna:
  Led_merah = 0
  Led_biru = 1
  Led_hijau = 0
  Waitms 20
  Gosub Baca_warna_jarak
  Warna_biru = Dat_warna
  Waitms 10

  Led_merah = 1
  Led_biru = 0
  Led_hijau = 0
  Waitms 20
  Gosub Baca_warna_jarak
```

```
Warna_merah = Dat_warna
Waitms 10

Led_merah = 0
Led_biru = 0
Led_hijau = 1
Waitms 20
Gosub Baca_warna_jarak
Warna_hijau = Dat_warna * 10
Waitms 30
Return
```

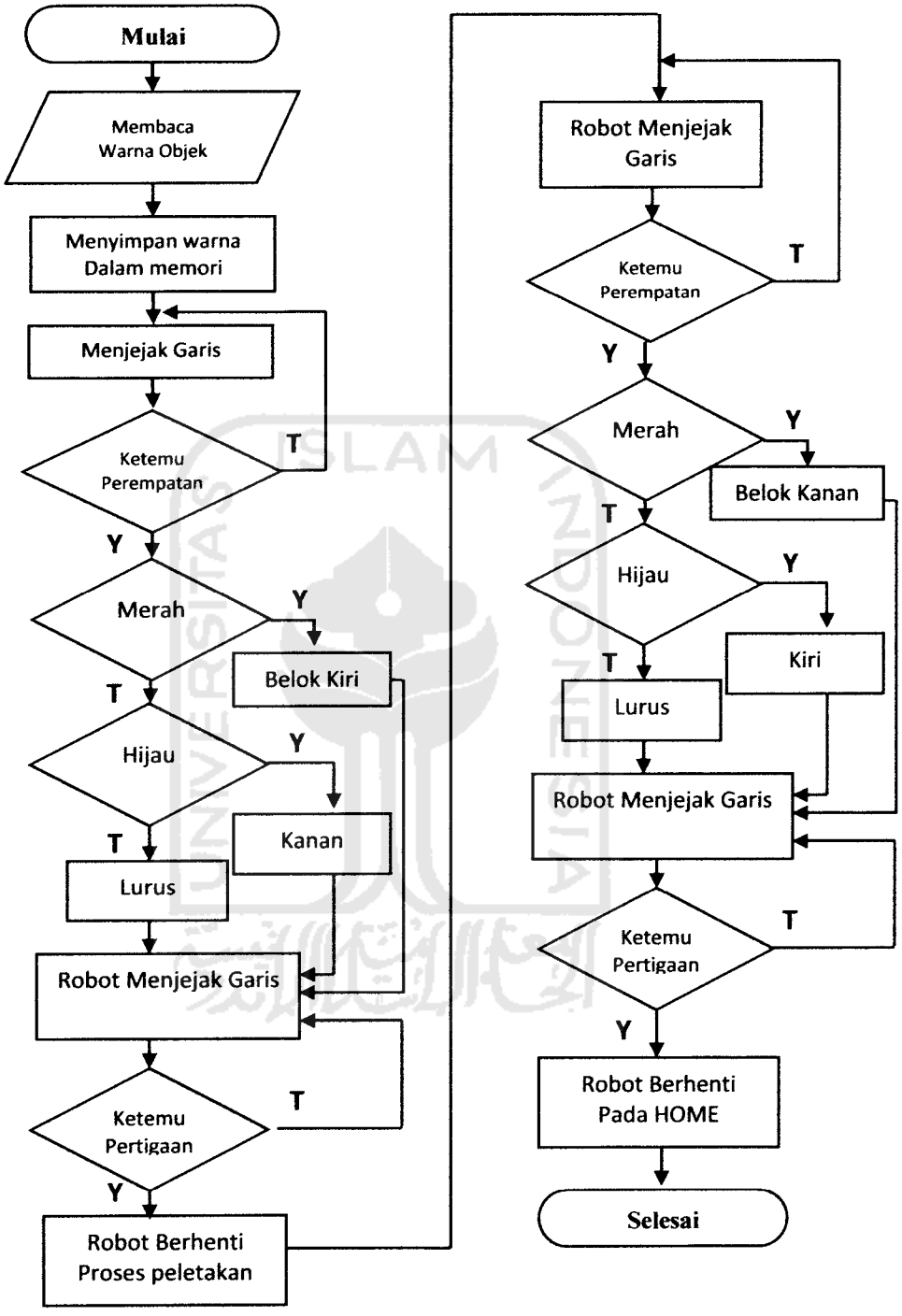
3.2.2.11. Proses Pemilihan Jalan Robot

Sebuah *mobile* robot harus memiliki panduan atau alur perjalanan dalam melaksanakan tugasnya. Dalam hal ini digunakan sistem pemetaan(*mapping*). Robot ini memiliki algoritma pemetaan dalam menentukan arah perjalanan. Kasus yang diberikan pada robot ini yaitu dapat mengangkut barang (objek) dari satu sumber (*home*) dan meletakannya pada sebuah target dengan memilih salah satu jalur perjalanan target yang bergantung dari warna pembacaan objek kemudian dapat kembali ke posisi awal (*home*). Bentuk dari contoh arena pada kasus ini ditunjukan pada gambar dibawah ini :



Gambar 3.18. Arena atau Lapangan Robot

Robot pengangkut barang ini memiliki panduan dalam hal pemilihan jalur yaitu dengan cara sensor garis mendeteksi adanya sebuah persimpangan jalan antara perempatan. Proses dari pemilihan jalur ini dapat dijelaskan dalam diagram alir sebagai berikut:



Gambar 3.19. Diagram alir proses pemetaan arena

```
Gosub Baca_warna
If Warna_merah > 500 And Warna_biru < 500 And Warna_hijau <
500 Then
Perjalan = Merah
Elseif Warna_merah < 500 And Warna_biru > 500 And Warna_hijau
< 200 Then
Perjalan = Biru
Elseif Warna_merah > 300 And Warna_biru > 500 And Warna_hijau
> 150 And Warna_hijau < 500 Then
Perjalan = Hijau

Else
Gosub Jepit_buka
Gosub Mundur_dikit
Goto Scan_objek
End If
Gosub Grip_naik
Gosub Mundur_dikit
Gosub Balik_kiri

Do
Call Ngeline
      Gosub Conversi
      If Ln2 = 0 And Ln3 = 0 And Ln4 = 0 And Ln5 = 0 Then
      Exit Do
      Loop
            If Perjalan = Merah Then
            Gosub Belok_kiri
            Elseif Perjalan = Biru Then
            For X = 1 To 30
            Call Ngeline
            Next X
            Else
            Gosub Belok_kanan
            End If

Do
Call Ngeline
Gosub Conversi
If Ln2 = 0 And Ln3 = 0 And Ln4 = 0 And Ln5 = 0 Then
Exit Do
Loop

Gosub Ngerem
Gosub Grip_turun
Gosub Jepit_buka
Gosub Maju_dikit
```

```
Gosub Mundur_dikit
Gosub Grip_naik
Gosub Jepit_tutup

If Perjalan = Merah Then
Gosub Balik_kanan
Elseif Perjalan = Biru Then
Gosub Balik_kanan
Else
Gosub Balik_kiri
End If

Do
Call Ngeline
Gosub Conversi
If Ln2 = 0 And Ln3 = 0 And Ln4 = 0 And Ln5 = 0 Then
Exit Do
Loop

If Perjalan = Merah Then
Gosub Belok_kanan
Elseif Perjalan = Biru Then
For X = 1 To 40
Call Ngeline
Next X
Else
Gosub Belok_kiri
End If
Loop
```

Dari proses awal pengangkutan objek, robot telah menyimpan data warna objek yang sedang diangkut dan disimpan dalam variable perjalanan. Setelah menemukan perempatan, robot akan memilih apakah akan tetap berjalan lurus, belok kiri, atau belok kanan. Demikian juga saat robot dalam perjalanan pulang menuju *home* awal.

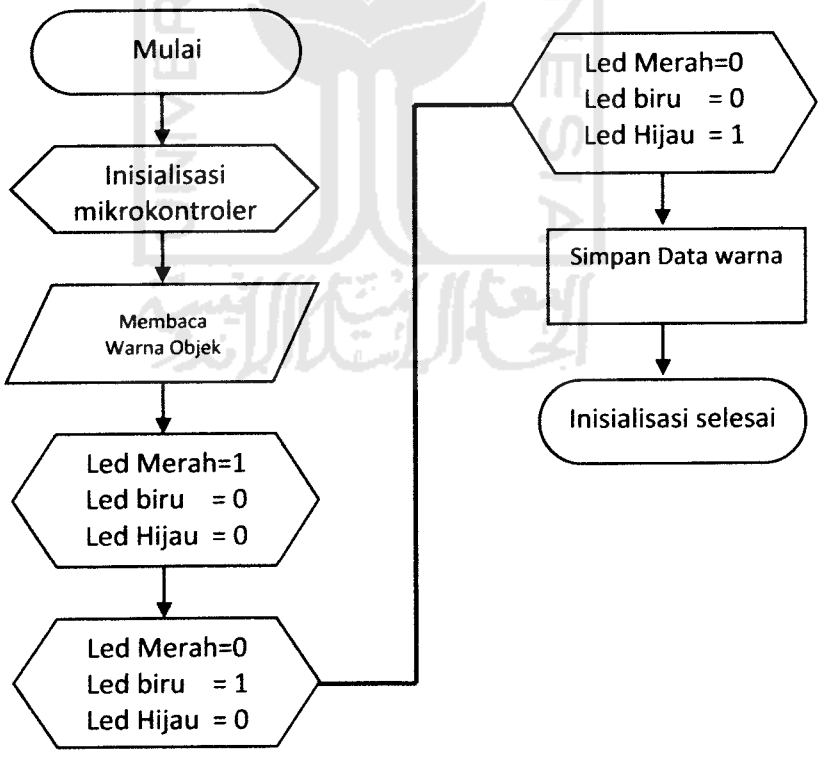
Proses penyimpanan data warna hanya sekali saat robot akan mengambil objek yang akan dipindahkan. Setelah mendapatkan data warna yang telah sesuaikan

dalam pengambilan data robot sudah dapat menentukan perjalan dalam penempatan objek yang akan dipindahkan.

3.2.2.12. Sub flowchart dan program masing-masing sensor

Sub *flowchart* dan program untuk masing-masing sensor yang di terapkan dalam aplikasi robot pemindah barang berdasarkan warna, bagian ini digunakan untuk mempermudah menganilisa bagian program yang digunakan setelah nantinya akan digabungkan dengan program keseluruhan.

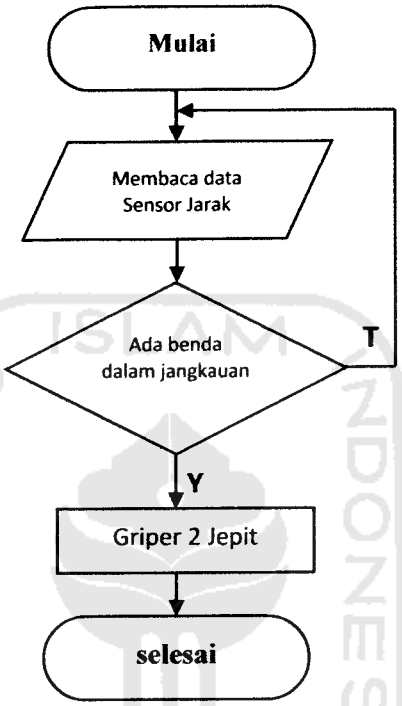
3.2.2.13.1. Sensor Warna



Gambar 3.20. Diagram alir proses pembacaan warna objek

```
Do
Gosub Baca_warna
Locate 2 , 1
Lcd  "R:"  ;  Warna_merah  ;  "  B:"  ;  Warna_biru  ;  "  G:"  ;
Warna_hijau ; "      "
Loop

Baca_warna:
      Led_merah = 1
      Led_biru = 0
      Led_hijau = 1
      Waitms 40
      Gosub Baca_warna_jarak
      Warna_biru = Dat_warna
      Waitms 10
      Led_merah = 1
      Led_biru = 1
      Led_hijau = 0
      Waitms 40
      Gosub Baca_warna_jarak
      Warna_hijau = Dat_warna
      Waitms 10
      Led_merah = 0
      Led_biru = 1
      Led_hijau = 1
      Waitms 40
      Gosub Baca_warna_jarak
      Warna_merah = Dat_warna
      Waitms 10
      Led_merah = 1
      Led_biru = 1
      Led_hijau = 1
      Waitms 40
      Return

 Baca_warna_jarak:
      Start Adc
      Dat_warna = Getadc(0)
      Stop Adc
      Dat_warna = Dat_warna
      Return
```

3.2.2.13.2. Sensor Jarak

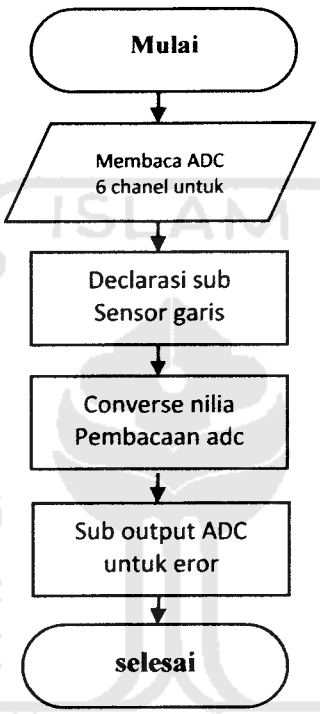


Gambar 3.21. Diagram alir proses pembacaan Sensor jarak

```
Baca_jarak:
   Init_srf = 1
   Waitus 10
   Init_srf = 0
   Data_srf = 0
   Do
      Waitus 1
      Incr Data_srf
      If Echo_srf = 1 Then Exit Do
      Loop Until Data_srf = 1000
           Data_srf = 0

           Do
           Waitus 1
           Incr Data_srf
           If Echo_srf = 0 Then Exit Do
```

```
If Data_srf > 3000 Then Exit Do
Loop
Data_srf = Data_srf / 10
Return
```

3.2.2.13.3. Pembacaan ADC Sensor Garis



Gambar 3.22. Diagram alir proses pembacaan Sensor Garis

```
Baca_adc:
  Start Adc
   Dat_ln1 = Getadc(2)
   Dat_ln2 = Getadc(2)
   Dat_ln3 = Getadc(3)
   Dat_ln4 = Getadc(4)
   Dat_ln5 = Getadc(5)
   Dat_ln6 = Getadc(6)
   Dat_ln7 = Getadc(7)
   Dat_ln8 = Getadc(7)
   Stop Adc
   Return
```

```
Declare Sub Ngeline
  Ln0 Alias Lines.0
  Ln1 Alias Lines.1
  Ln2 Alias Lines.2
  Ln3 Alias Lines.3
  Ln4 Alias Lines.4
  Ln5 Alias Lines.5
  Ln6 Alias Lines.6
  Ln7 Alias Lines.7

Conversi:
Gosub Baca_adc
If Dat_ln1 > Ref_ln1 Then Ln0 = 1
If Dat_ln1 < Ref_ln1 Then Ln0 = 0
If Dat_ln2 > Ref_ln2 Then Ln1 = 1
If Dat_ln2 < Ref_ln2 Then Ln1 = 0
If Dat_ln3 > Ref_ln3 Then Ln2 = 1
If Dat_ln3 < Ref_ln3 Then Ln2 = 0
If Dat_ln4 > Ref_ln4 Then Ln3 = 1
If Dat_ln4 < Ref_ln4 Then Ln3 = 0
If Dat_ln5 > Ref_ln5 Then Ln4 = 1
If Dat_ln5 < Ref_ln5 Then Ln4 = 0
If Dat_ln6 > Ref_ln6 Then Ln5 = 1
If Dat_ln6 < Ref_ln6 Then Ln5 = 0
If Dat_ln7 > Ref_ln7 Then Ln6 = 1
If Dat_ln7 < Ref_ln7 Then Ln6 = 0
If Dat_ln8 > Ref_ln8 Then Ln7 = 1
If Dat_ln8 < Ref_ln8 Then Ln7 = 0
Return

Sub Ngeline
Gosub Conversi
Select Case Lines
Case &B11111100 : Eror = 8
Case &B11111000 : Eror = 6
Case &B11111011 : Eror = 4
Case &B11110011 : Eror = 2
Case &B11110111 : Eror = 1
Case &B11100111 : Eror = 0
Case &B11101111 : Eror = -1
Case &B11001111 : Eror = -2
Case &B11011111 : Eror = -4
Case &B00011111 : Eror = -6
Case &B00111111 : Eror = -8
End Select
End Sub
```