RANCANG BANGUN ROBOT *LINE FOLLOWER* BERBASIS CAHAYA TAMPAK

(BAGIAN II)

TUGAS AKHIR

FAHMI DIYATI
081310213012

PROGRAM STUDI D3 OTOMASI SISTEM INSTRUMENTASI
DEPARTEMEN TEKNIK
FAKULTAS VOKASI
UNIVERSITAS AIRLANGGA
SURABAYA
2016

LEMBAR PERSETUJUAN TUGAS AKHIR

RANCANG BANGUN ROBOT *LINE FOLLOWER* BERBASIS CAHAYA TAMPAK (BAGIAN II)

TUGAS AKHIR

Sebagai Salah Satu Syarat Untuk Memperoleh Gelar Ahli Madya
Bidang Otomasi Sistem Instrumentasi
Pada Departemen Teknik Fakultas Vokasi
Universitas Airlangga

Oleh :

Fahmi Diyati

NIM. 081310213012

Disetujui Oleh :

Pembimbing	Konsultan
Winarno, S.SI., M.T.	Deny Arifianto, S.Si
NIP. 198109122015041001	NIK. 139111263

LEMBAR PENGESAHAN NASKAH TUGAS AKHIR

Judul	**: RANCANG BANGUN ROBOT *LINE FOLLOWER* BERBASIS CAHAYA TAMPAK (BAGIAN II)**
Penyusun	**: Fahmi Diyati**
NIM	**: 081310213012**
Pembimbing	**: Winarno, S.SI., M.T.**
Konsultan	**: Deny Arifianto S.Si**
Tanggal Ujian	**: 5 Agustus 2016**

Disetujui Oleh :

Pembimbing	**Konsultan**
Winarno, S.SI., M.T.	**Deny Arifianto, S.Si**
NIP. 198109122015041001	**NIK. 139111263**

Mengetahui :

Kepala Departemen Teknik	**Koordinator Program Studi D3 Otomasi Sistem Instrumentasi**
Ir. Dyah Herawatie, M.Si.	**Winarno, S.SI., M.T.**
NIP. 1967111 11993032002	**NIP. 198109122015041001**

PEDOMAN PENGGUNAAN TUGAS AKHIR

Tugas Akhir ini tidak dipublikasikan, namun tersedia di perpustakaan dalam lingkungan Universitas Airlangga. Diperkenankan untuk dipakai sebagai referensi kepustakaan, tetapi pengutipan seijin penulis dan harus menyebutkan sumbernya sesuai kebinasaan ilmiah.

Dokumen Tugas Akhir ini merupakan hak milik Universitas Airlangga.

KATA PENGANTAR

Puji syukur penulis panjatkan ke hadirat Tuhan Yang Maha Esa atas karunia serta hidayah-Nya, penulis dapat menyelesaikan tugas akhir yang berjudul "***Rancang Bangun*** **Robot *Line Follower* Berbasis Cahaya Tampak**" dengan baik. Shalawat serta salam semoga tetap tercurahkan kepada Rasulullah Muhammad SAW yang telah menunjukan jalan yang terang.

Tugas akhir ini,dapat selesai dengan baik berkat bantuan dari berbagai pihak. Oleh sebab itu, tidak lupa penulis mengucapkan terima kasih kepada semua pihak yang turut membantu dalam menyelesaikan tugas akhir ini, yang terhormat:

1. Bapak Winarno, S.Si., M.T., selaku Ketua Koordinator Program Studi D3 Otomasi Sistem Instrumentasi.
2. Bapak Winarno, S.Si., M.T., selaku Dosen Pembimbing yang tak henti-hentinya mendukung, memberikan ilmu, masukan dan membantu hingga terselesaikannya Proposal Tugas Akhir ini.
3. Bapak Deny Arifianto, S.Si, selaku Dosen Konsultan yang selalu memberikan bimbingan dan ilmu dalam pembuatan proposal tugas akhir ini. Semoga Allah membalas kebaikan bapak.
4. Kedua Orang Tua yang selalu mendoakan, memberi semangat dan dukungannya hingga penulis mampu menyelesaikan proposal tugas akhir ini.
5. Kepada Lek Mamah, Mas Fahrus, Mas Fauzan, Mas Fahrizal, Mbak Dayah, Mbak Frida, Mbak Maya, dek Fahrul, dan keponakan saya Faizan dan Fabian yang selalu memberikan dukungan.

6. Kepada Anak Rumpita yang selalu memberi semangat saat penulis mulai lelah.
7. Kepada abang Hendrik Junaedi sebagai *partner* TA (Tugas Akhir), terimakasih atas kerjasamanya.
8. Semua Dosen D3 Otomasi Sistem Instrumentasi yang selalu mengajar dengan baik.
9. Kepada keluarga ASTRAI yang telah banyak membantu dan memberikan banyak saran kepada penulis. Semoga sukses selalu.
10. Semua teman D3 Otomasi Sistem Instrumentasi 2013 yang selalu banyak membantu.

Harapan kami sebagai penulis adalah semoga dengan terselesaikannya Proyek Akhir ini, dapat bermanfaat bagi kami khususnya dalam pengembangan ilmu pengetahuan dan teknologi umunya dimasa yang akan datang. Sadar dengan keterbatasan waktu dan kemampuan yang dimiliki oleh penulis, hasil dari Proyek Akhir ini tentunya masih jauh dari kesempurnaan. mungkin untuk mencapai hasil yang terbaik. Oleh karena itu dengan segala kerendahan hati penyusun mengharapkan saran dan keritik demi penyempurnaan Proyek Akhir ini.

Surabaya, 18 Juli 2015

Penulis

Fahmi Diyati, 2016, ***Rancang Bangun Robot Line Follower Berbasis Cahaya Tampak (bagian II)***. Tugas akhir ini dibawah bimbingan Winarno, S.Si., M.T dan Deny Arifianto S.Si. Prodi D3 Otomasi Sistem Instrumentasi Departemen Teknik Fakultas Vokasi Universits Airlangga.

ABSTRAK

Robot *Line Follower* adalah suatu robot yang berjalan mengikuti garis yang memiliki warna berbeda dari lintasan yang dilaluinya. Dalam perancangan dan pengaplikasiannya, ada beberapa masalah yang harus dipecahkan yaitu perancangan *hardware* yang meliputi sistem mekanis robot dan perangkat elektroniknya, perancangan mekanik berupa desain sensor dan perancangan *software* untuk sistem pengendalian robot.

Dalam perancangannya Robot *Line Follower* dibagi menjadi tiga bagian umum yaitu bagian mata dalam hal ini berupa sensor cahaya untuk mendeteksi jalur robot pada suatu lintasan, bagian kaki yaitu berupa motor untuk pergerakan robot, serta bagian otak yaitu berupa IC Mikrokontroller ATMega16 sebagai pengendali robot.

Dari hasil pengujian jarak terhadap tegangan pada bidang warna hitam dan putih didapatkan jarak terbaik pada jarak 0cm yang didapatkan dari jarak antar LED dan *Photodiode*. Pada pengujian terhadap warna LED dan perbedaan bidang pantulan didapatkan hasil untuk bidang keramik (putih) adalah LED putih, kayu (coklat) adalah LED biru, karpet (Abu abu) adalah LED kuning, bannner (putih) adalah LED putih , dan lakban (hitam) adalah LED kuning. Dari hasil pengujian keseluruhan sistem didapatkan persentase keberhasilan 100%, dimana pengujian ini dilakukan pada bidang lintasan yang berbeda yaitu keramik putih dan banner putih dengan berbagai *desain* lintasan (garis hitam) yang berbeda. Hal ini menunjukkan bahwa sistem bekerja sesuai dengan yang di harapkan.

Kata Kunci: Robot *Line Follower*, sensor *photodiode* , pengambilan data

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1 Latar Belakang

Perkembangan Teknologi dan otomasi industri yang semakin pesat, canggih dan *modern* mendorong manusia untuk memenuhi kebutuhan hidupnya dengan cepat, tepat dan *efisien*. Salah satu teknologi yang berkembang saat ini adalah teknologi di bidang robotika. Banyak negara maju seperti Jepang, Amerika, Inggris dan Jerman yang menciptakan berbagai jenis robot untuk membantu dan mempermudah pekerjaan manusia di masa depan. Salah satu jenis robot yang banyak diciptakan adalah robot *line follower* dengan berbagai keistimewaan yang dimilikinya.

Robot *line follower* merupakan suatu jenis robot bergerak (*mobile* robot) yang mengikuti suatu garis pandu yang telah dibuat pada bidang lintasan. Garis yang dimaksud adalah garis berwarna hitam diatas permukaan berwarna putih atau sebaliknya, ada juga lintasan dengan warna lain dengan permukaan yang kontras dengan warna garisnya. Robot *line follower* telah dilengkapi sensor garis untuk mendeteksi warna garis pada bidang lintasan. Sensor garis terdiri dari LED (*Light Emiting Diode*) dan sensor *photodiode*. Dimana LED akan memancarkan cahaya dan pantulan cahayanya akan diterima oleh sensor *photodiode* sehingga ada perubahan tegangan yang di deteksi oleh sensor.

Melihat dari kurangnya penelitian di bidang robotika tentang *system robotic* khususnya pada robot *line follower* akan menimbulkan masalah seperti bagaimana pemilihan warna LED yang kepekaannya baik terhadap warna garis, bidang yang

digunakan untuk lintasan robot, maupun penempatan sensor garis yang baik pada robot. Oleh karena itu kami membuat penelitian tugas akhir "Rancang Bangun Robot *Line Follower* Berbasis Cahaya Tampak" agar robot mampu mengikuti garis yang telah ditentukan dengan baik.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah disebutkan, maka perumusan masalah dalam tugas akhir ini adalah:

1. Bagaimana desain sensor agar robot tidak keluar dari lintasan yang sudah ditentukan?
2. Bagaimana kinerja robot *line follower* berbasis cahaya tampak?

1.3 Batasan Masalah

Selain perumusan masalah diatas perlu diberikan pembatasan masalah agar pembahasan nantinya tidak meluas dan menyimpang dari tujuan awal.

Pembatasan pada proposal tugas akhir ini adalah:

1. Garis yang digunakan warna hitam
2. Bidang lintasan yang digunakan yaitu kayu, keramik dan karpet

1.4 Tujuan

Berdasarkan latar belakang yang telah disebutkan tujuan dalam pembuatan tugas akhir ini adalah:

1. Membuat desain sensor agar robot tidak keluar dari lintasan yang sudah ditentukan.
2. Mengetahui kinerja robot *line follower* berbasis cahaya tampak.

1.5 Manfaat

Adapun manfaat dalam pembuatan tugas akhir ini yaitu:

1. Robot ini diharapkan dapat bergerak mengikuti garis dengan baik
2. Penelitian ini diharapkan dapat menjadi *referensi* atau panduan dalam pembuatan robot *line follower* dengan berbagai keistimewaan.

BAB II

TINJAUAN PUSTAKA

2.1 Robot *Line Follower*

Robot *Line Follower* (Robot Pengikut Garis) adalah robot yang dapat berjalan mengikuti garis pada sebuah lintasan yang sudah ditentukan dan robot ini termasuk dalam kategori jenis robot *mobile* yang di desain untuk bekerja secara otomatis. Garis yang dimaksud adalah garis berwarna hitam diatas permukaan berwarna putih atau sebaliknya, ada juga lintasan dengan warna lain tetapi warna garis harus kontras dari bidang lintasannya. Robot ini juga di kenal dengan sebutan *Line Tracker*, *Line Tracer* Robot dan sebagainya. Cara kerja dari sistem robot *line follower* secara umum ialah dimulai dari pembacaan lintasan atau garis oleh sensor *photodiode* berserta LED yang mana intensitas pantulan sinar LED akan berbeda jika terkena bidang pantul yang gelap dengan bidang pantul yang lebih terang, dari perbedaan inilah dimanfaatkan sebagai pendeteksi lintasan atau garis dan selanjutnya diketahui nilai parameter ADC (*Analog to Digital Converter*) untuk setiap sensor garis. ADC merupakan piranti dari sebuah mikrokontrollet ATmega16 yang terletak pada PORTA. Nilai ADC berfungsi sebagai parameter untuk pembeda warna garis hitam atau putih dan juga untuk membandingkan nilai yang dibaca sensor *photodiode* mendeteksi objek pantul gelap atau terang. Setelah mendapatkan parameter nilai ADC pada setiap sensor maka parameter tersebut dijadikan eksekusi pembeda garis pada program AVR.

Robot *Line Follower* sendiri terdiri dari dua macam jenis yaitu Robot *Line Follower* Analog dan Robot *Line Follower* Digital.

Line follower analog ini mampu bergerak tanpa adanya penanaman *software* Khusus dari PC ke *chip* yang ada di robot, Dengan kata lain hanya komponen Elektronika yang menggerakan robot. Sedangkan *line follower* digital menggunakan mikrokontroller untuk melacak garis sehingga sistem pelacakan garisnya lebih halus jika dibandingkan *line follower* analog. *Line follower* digital memiliki mikrokontroller yang dapat diprogam dalamnya sesuai dengan keinginan pembuatnya. Karena kemampun itulah *line follower* digital mampu melewati jalur yang lebih rumit jika dibandingkan dengan *line follower* analog.

2.2 Sensor Garis

Sensor garis atau *proxymity* sensor adalah sensor yang berfungsi mendeteksi warna gelap atau warna terang, dimana warna gelap atau terang tersebut terdeteksi oleh sensor akibat pantulan cahaya lampu. Sensor garis terdiri dari LED dan sensor *photodiode*.

LED (*Light Emitting Diode*) merupakan jenis dioda semikonduktor yang dapat mengeluarkan energi cahaya ketika diberikan tegangan dan hanya mengalirkan arus listrik satu arah saja. LED dapat memancarkan cahaya karena menggunakan *dopping galium*, *arsenic* dan *phosporus*. Berbeda dengan dioda pada umumnya, kemampuan mengalirkan arus pada LED cukup rendah yaitu maksimal 20 mA.

Sensor *photodiode* adalah salah satu jenis sensor peka cahaya (*photodetector*). *Photodiode* akan mengalirkan arus yang membentuk fungsi linear terhadap intensitas cahaya yang diterima. Arus ini umumnya teratur terhadap *power density* (Dp). Perbandingan antara arus keluaran dengan *power density* disebut sebagai *current responsitivity*. Arus yang dimaksud adalah arus bocor ketika *photodiode* tersebut disinari dan dalam keadaan dipanjar mundur.

Prinsip kerja sensor pada rancangan sensor garis atau *proxymity* sensor dibawah ini yaitu, nilai resistansinya akan berkurang bila terkena cahaya dan bekerja pada kondisi *riverse* bias. Untuk pemberi pantulan cahayanya digunakan LED *superbright*, komponen ini mempunyai cahaya yang sangat terang, sehingga cukup untuk mensuplai pantulan cahaya ke *photodiode*. Berikut ini prinsip dan gambaran kerja dari sensor *photodiode.*

Gambar 2.1 Sensor Garis

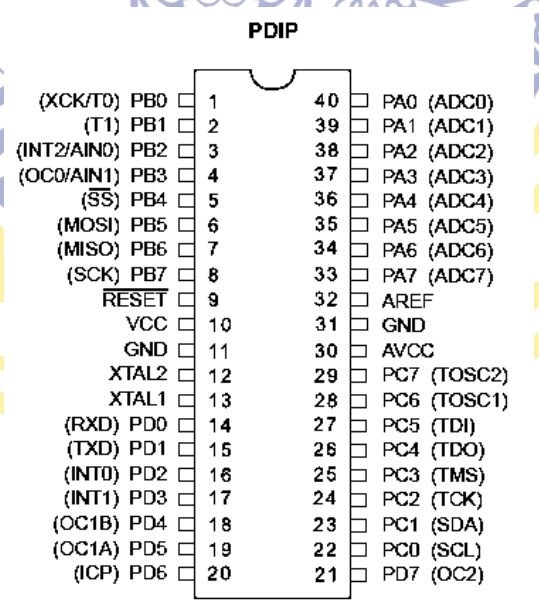
2.3 Mikrokontroller AVR ATMega16

Mikrokontroler *AVR* standart memiliki arsitektur 8 *bit*, dimana semua instruksi dikemas dalam kode 16-*bit*, dan sebagian besar intruksi dieksekusi dalam 1(satu) siklus *clock*. *AVR* berteknologi *RISC* (*Reduced Instruction Set*

Computing), sedangkan *MCS51* berteknologi *CISC (Complex Intruction Set Computing*). Pada sistem ini mikrokontroller digunakan sebagai pusat kontrol dalam proses pembacaan nilai ADC pada setiap sensor *photodiode*, LCD dan Motor DC.

AVR dapat dikelompokkan menjadi empat kelas, yaitu keluarga *Attiny*, keluarga *AT902xx*, keluarga *Atmega* dan keluarga *AT86RFxx*. Pada dasarnya yang membedakan masing-masing kelas adalah memori, *peripheral*, dan fungsinya. Mikrokontroler *AVR* yang berukuran lebih kecil antara lain *Atmega8*, *Attiny2313* dengan ukuran *Flash Memory* 2KB dengan dua *input analog*.

Mikrokontroler pada dasarnya diprogram dengan bahasa *assembler*,tetapi saat ini mikrokontroler dapat diprogram dengan menggunakan bahasa tingkat tinggi seperti *BASIC*, *PASCAL* atau *C*.



Gambar 2.2 Konfigurasi Pin Mikrokontroller AVR ATMega 16

2.3.1 Konfigurasi Pin Mikrokontroler ATMega16

Konfigurasi pin ATMEGA16 dengan kemasan 40 pin *Dual In-line Package* (DIP) dapat dilihat pada Gambar 2.2. dari gambar diatas dapat dijelaskan fungsi dari masing-masing pin ATMEGA16 sebagai berikut.

1. VCC merupakan pin yang berfungsi sebagai masukan catu daya.
2. GND merupakan pin *Ground*.
3. Saluran *I/O* sebanyak 32 buah, yaitu *Port* A, *Port* B, *Port* C, dan *Port* D.
4. ADC 10 *bit* sebanyak 8 *channel*.
5. Tiga buah *Timer/Counter* dengan kemampuan perbandingan.

2.3.2 ADC *(Analog to Digital Converter)*

ADC adalah kepanjangan dari *Analog to Digital Converter* yang artinya pengubah dari analog ke digital. fungsi dari ADC adalah untuk mengubah data analog menjadi data digital yang nantinya data digital tersebut dapat diproses oleh mikrokontroller.

Pada umumnya ADC merupakan sebuah piranti tambahan berupa IC, namun mikrokontroller jenis AVR memiliki fungsi internal ADC sebesar 10 bit sebanyak 8 kanal yang terletak pada PORT A. Setiap PIN pada PORT A dapat digunakan sebagai *input* analog secara terpiah tanpa ada keterkaitan dengan pin yang lainnya.

Tegangan referensi adalah suatu nilai tegangan yang digunakan oleh ADC sebagai penentu nilai tegangan masukan maksimum, dengan rumus sebagai berikut:

Vref =1/2 Vin Maksimum (2.1)

Tegangan resolusi adalah suatu nilai tegangan analog terkecil yang dapat dikonversi oleh ADC menjadi tegangan digital. Besar resolusi atau ketelitian ADC dipengaruhi oleh jumlah bit pada ADC. Resolusi ADC internal AVR sebesar 8 atau 10 bit. Adapun rumus tegangan resolusi adalah:

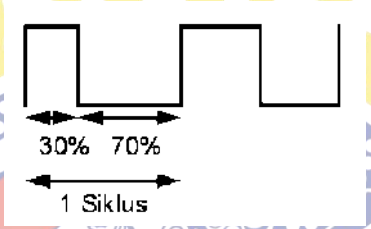
Vresolusi = Vin Maks/Resolusi ADC (2.2)

2.4 PWM (*Pulse Width Modulation*)

Pulse Width Modulation (PWM) secara umum adalah sebuah cara memanipulasi lebar sinyal yang dinyatakan dengan pulsa dalam suatu perioda, untuk mendapatkan tegangan rata-rata yang berbeda. Beberapa Contoh aplikasi PWM adalah pemodulasian data untuk telekomunikasi, pengontrolan daya atau tegangan yang masuk ke beban, regulator tegangan, *audio effect* dan penguatan, serta aplikasi-aplikasi lainnya.

Dalam PWM gelombang kotak, frekuensi tinggi dibangkitkan sebagai *output* digital. Untuk contoh, sebuah port bit secara kontinyu melakukan kegiatan saklar *on* dan *off* pada frekuensi yang relatif tinggi. Selanjutnya, bila sinyal diumpankan pada LPF (*low pass filter)*, tegangan pada *output filter* akan sama dengan *Root Mean Squere* (RMS) dari sinyal gelombang kotak. Selanjutnya tegangan RMS dapat divariasi dengan mengubah *duty cycle* dari sinyal.

DUTY CYCLE menyatakan fraksi waktu sinyal pada keadaan logika *high* dalam satu siklus. Satu siklus diawali oleh transisi *low to high* dari sinyal dan berakhir pada transisi berikutnya. Selama satu siklus, jika waktu sinyal pada keadaan *high* sama dengan *low* maka dikatakan sinyal mempunyai *duty cycle* 50 %. *Duty cycle* 20 % menyatakan sinyal berada pada logika 1 selama 1/5 dari waktu total.



Gambar 2.3 *Duty Cycle* 30%

2.5 Bahasa Pemograman C

Bahasa C merupakan salah satu bahasa pemograman computer yang dapat disebut sebagai *general-purpose language*, yaitu bahasa pemograman yang dapat digunakan untuk tujuan apa saja. Kerangka bahasa C terdiri dari fungsi main(), deklarasi variabel, perintah (*statement), library access* dan komentar-komentar. Fungsi main() merupakan fungsi utama yang wajib ada pada saat kita membuat program dengan bahasa C. Dalam sebuah *project* hanya ada 1 buah fungsi main() saja. Namun dalam bahasa C, tidak membatasi hanya 1 saja, melainkan untuk membuat fungsi-fungsi lain selain fungsi main() yang dapat mempermudah dalam membuat suatu program.

Berikut adalah contoh penulisan program dalam bahasa C :

```
#include<mega16.h>
#include <delay.h>
#define sensor PINA.0
#define LCD PORTC.0
//variabel global
Void main(void)
{
//variabel local
DDRA=0X00;
PORTA=0XFF;
DDRA=0X00;
POORTA=0XFF;
While (1)
{
…….
……
};
```

Prepocessor

Digunakan untuk memasukkan (*include*) *text* dari

Penempatan variabel global untuk komentar program utama

Inisialisai

Program akan berulang terus karena syarat *while* (1) akan selalu menghasilkan nilai benar (*true*)

2.6 *Code Vision* AVR

*CodeVision*AVR merupakan *software C-cross compiler*, dimana program dapat ditulis menggunakan bahasa-C. Dengan menggunakan pemrograman bahasa-C diharapkan waktu *design* (*developing time*) akan menjadi lebih singkat. Setelah program dalam bahasa-C ditulis dan dilakukan kompilasi tidak terdapat kesalahan maka proses *download* dapat dilakukan. Mikrokontroler AVR mendukung *system download* secara ISP *(In-system Programming*). *CodeVision* AVR merupakan yang terbaik bila dibandingkan dengan kompiler – kompiler yang lain karena beberapa kelebihan yang dimiliki oleh *CodeVision*AVR antara lain :

1. Menggunakan IDE (*Intergrated Development Environment*).
2. Fasilitas yang disediakan lengkap (program *editing*, program *compiler*, downloading *program*) serta tampilanya yang terlihat menarik dan mudah dimengerti. Dapat melakukan pengaturan sedemikian rupa sehingga membantu memudahkan dalam penulisan program.
3. Mampu membangkitkan kode program secara otomatis dengan menggunakan fasilitas CodeWizardAVR.
4. Memiliki fasilitas untuk *downloading* program langsung dari *CodeVision*AVR dengan menggunakan *hardware* khusus seperti Atmel STK500, Kanda *System* STK200+ / 300 dan beberapa *hardware* lain yang telah didefinisikan oleh *CodeVision*AVR.

5. Memiliki fasilitas *debugger* sehingga dapat menggunakan *software compiler* lain untuk memeriksa kode *assembler*-nya, contohnya AVRStudio.
6. Memiliki terminal komunikasi serial yang terintregasi dalam *CodeVision*AVR sehingga dapat digunakan untuk membantu pemeriksaan program yang telah dibuat khususnya yang menggunakan fasilitas komunikasi serial USART.

Selain itu, *CodeVision*AVR juga menyediakan sebuah *tool* yang dinamakan *Code Generator* atau *CodeWizard*AVR yang merupakan salah satu fasilitas yang disediakan oleh *CodeVision*AVR yang dapat digunakan untuk mempercepat penulisan *listing* program. Fasilitas ini sangat membantu untuk mengetahui nama *register* yang akan digunakan untuk mengatur *mode* kerja fitur-fitur yang ada dalam mikrokontroler.

2.7 ProgISP

Software ini digunakan dalam pemrograman mikrokontroler khususnya saat melakukan *download File* *. HEX ke dalam memori mikrokontroler AT90, AT*tiny*, ATmega atau MCS-51. *Software* ini bersifat *portable* jadi tidak perlu di *install* terlebih dahulu.

2.8 GUI (*Graphic User Interface*)

GUI (*Graphic User Interface*) adalah Antar muka komputer yang berbasiskan grafis. Citra grafis yang ditampilkan di layar komputer yang memungkinkan untuk mengakses aplikasi *software* dengan memakai menu

dropdown, *dialog box*, *radio button*, *check box*, *panel*, *tabs*, *toolbar*, *icon shortcuts* dan *tool* lain. Atau bisa juga dikatakan jenis antarmuka pengguna yang memungkinkan orang untuk berinteraksi dengan program dengan lebih banyak dengan gambar daripada perintah *teks*. Sebuah GUI grafis menawarkan ikon, dan *visual* indikator, sebagai lawan dari antarmuka berbasis *teks*. GUI juga dikenal sebagai tombol menu. Tombol menu disini untuk mengatur *set point* pada sebuah program dengan menggunakan tombol *push button*.

BAB III

METODE PENELITIAN

3.1 Tempat dan Waktu Penelitian

Perencanakan dan pembuatan alat ini dilakukan di Laboratorium Bengkel Mekanik. Program Studi D3 Otomasi Sistem Instrumentasi, Departemen Fisika, Fakultas Sains dan Teknologi Universitas Airlangga selama kerang lebih 4 bulan yang dimulai dari blan April 2015 sampai juli 2015.

3.2 Alat dan Bahan

3.2.1 Alat

Alat-alat yang diperlukan untuk membuat sistem ini yaitu multimeter yang digunakan untuk menguji rangkaian dan jalur PCB, catu daya untuk memberikan tegangan pada rangkaian yang digunakan dan juga untuk *supllay* motor dc, solder untuk menyolder komponen ke PCB dan PC sebagai perantara untuk menuliskan program pada AVR.

Perangkat *software* yang digunakan untuk membuat sistem ini yaitu *CodeVision* (AVR) untuk menuliskan program pada sistem dan USBISP sebagai *downloader* digunakan untuk mengunggah program dari AVR ke mikrokontroller.

3.2.2 Bahan

Bahan-bahan yang diperlukan untuk membuat sistem pada robot ini yaitu sensor garis yang terdiri dari LED dan *photodiode* sebagai pendeteksian warna garis Robot *Line Follower*, mikrokontroller ATmega16 sebagai kontroler, *Driver* Motor DC VNH2SP30 sebagai control pergerakan motor dc, motor dc sebagai penggerak jalannya robot, alumunium sebagai kerangka robot, *Acrylic* sendiri sebagai *body* untuk robot.

3.3 Prosedur Penelitian

Prosedur penelitian yang dilakukan pada penulisan ini terdiri dari beberapa tahap, yaitu tahap persiapan, tahap perancangan, tahap perwujudan mekanik, tahap perwujudan *hardware*, tahap perwujudan *software*, tahap pengujian sistem, dan analisis data. Untuk lebih jelasnya, berikut grafik tahapan prosedur penelitian:

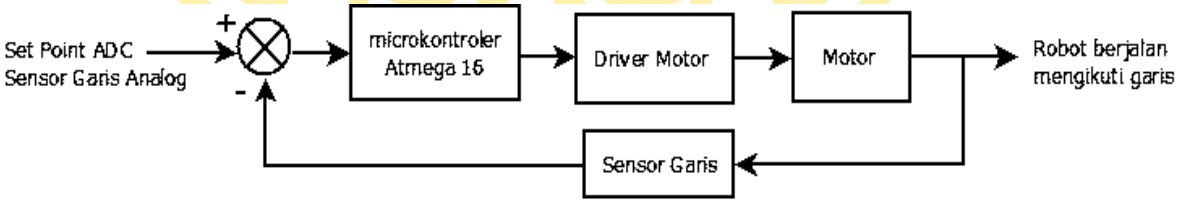
Gambar 3.1 Diagram Blok Prosedur Penelitian

3.3.1 Tahap Persiapan

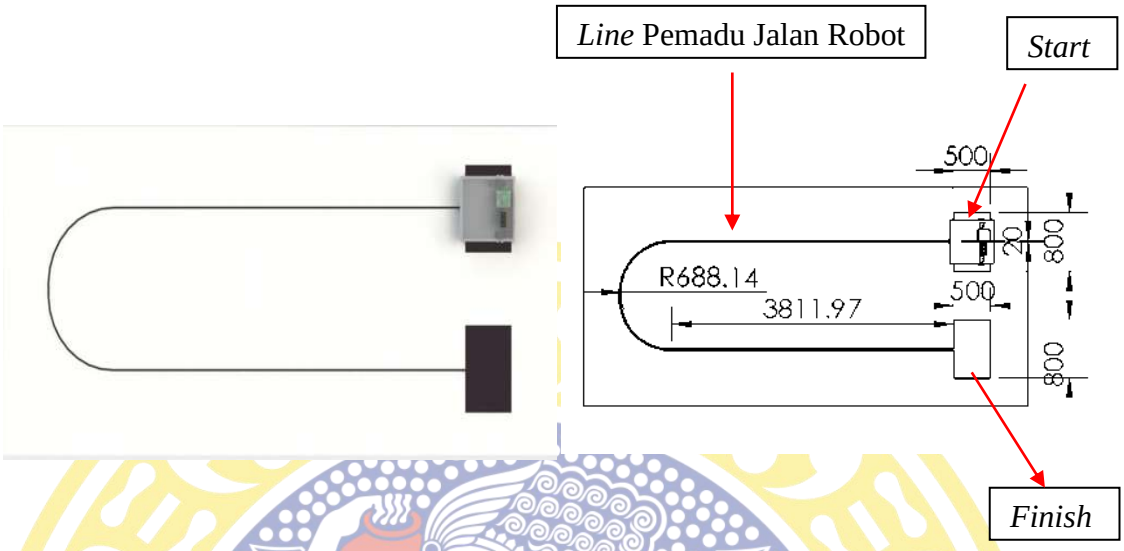
Tahap persiapan merupakan tahapan awal dalam melakukan penelitian, pada tahap ini penulis melakukan studi literatur dengan mencari berbagai acuan baik melalui buku, jurnal, tugas akhir maupun artikel dengan narasumber yang jelas dan terpercaya dengan tujuan untuk melengkapi literatur mengenai penelitian ini. Dan juga penulis menyiapkan alat dan bahan yang diperlukan diperlukan dalam penelitian ini untuk mempersiapkan menuju ke tahap selanjutnya.

3.3.2 Tahap Perancangan Alat

Tahap perancangan alat terdiri dari perancangan *hardware* dan perancangan mekanik sistem alat. Sensor yang digunakan adalah sensor garis, yang merupakan sensor pengukur warna garis melalui LED sebagai *Transmitter* dan *Photodiode* sebagai R*eceiver*. Kemudian *output* yang dihasilkan akan dibaca oleh ADC mikrokontroller. ADC digunakan untuk mengkonversi keluaran sensor dari sinyal analog menjadi sinyal digital kemudian menentukan parameter nilai ADC untuk garis hitam dan juga garis putih. Mikrokontroler bertugas untuk mengatur gerakan motor saat sensor garis mendeteksi garis.



Gambar 3.2 Diagram Kontrol Sistem



(a) (b)

Gambar 3.3 jalur *line follower*

(a) Gambar jalur Robot *line follower*

(b) Dimensi jalur Robot *line follower*

a) Perancangan *hardware*

Tahap perancangan *hardware* terdiri atas pembuatan rangkaian elektronik untuk robot *line follower*. Adapun rancangan *hardware* dari *prototype* robot yang akan dibuat adalah sebagai berikut : rangkaian minimum *system*, rangkaian sensor, rangkaian *driver* motor, dan rangkaian LCD.

b) Tahap Pemrograman *Software*

Tahap pembuatan *software* meliputi pembuatan inisialisasi program, pembuatan *flowchart* program, dan pembuatan program untuk mengeksekusi rancangan *hardware* dan mekanik yang telah dibuat. *Software* yang digunakan yakni *CodeVision* AVR. Program yang akan dibuat dalam sistem ini meliputi

perbedaan warna garis untuk lintasan robot dengan menggunakan sensor *photodiode* dan driver motor DC untuk mengendalikan jalannya robot.

Tahapan awal yang dilakukan sistem tersebut berdasarkan *software* yang dibuat yakni mengetahui keluaran sensor *photodiode* dimana keluaran sinyal analog dari sensor *photodiode* akan dibaca oleh ADC mikrokontroller dan masukan tersebut dikonversi menjadi sinyal digital yang menjadi parameter nilai untuk warna putih dan warna hitam. Jika nilai ADC yang dibaca sama dengan parameter yang ditentukan maka motor DC akan bergerak sesuai instruksi yang diinginkan. Untuk mengandalikan pergerakan dari motor DC menggunakan *driver* motor, dimana segala pergerakan akan dijalankan sesuai dengan instruksi yang telah diprogram pada AVR.

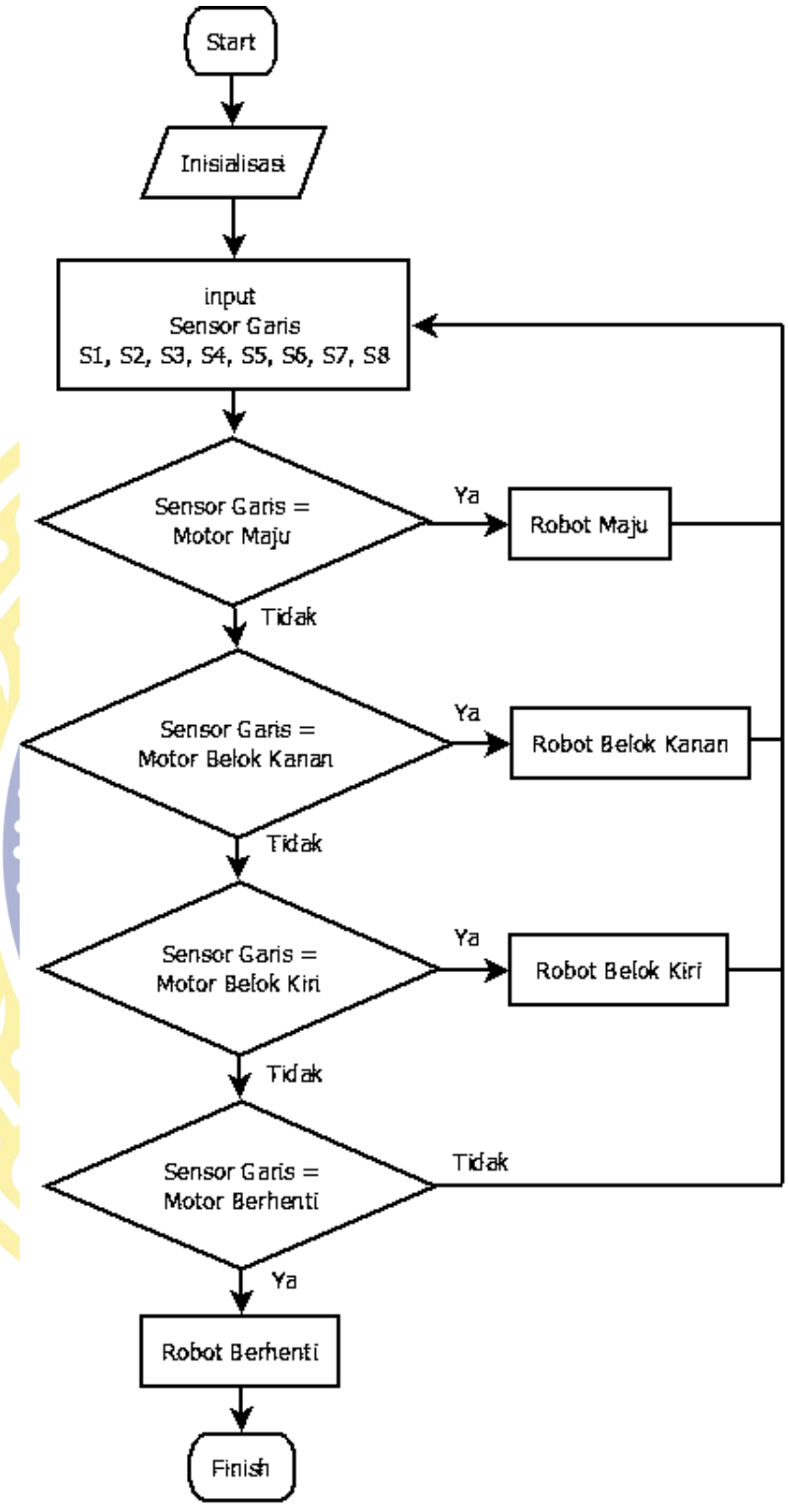
Untuk mengsinkronisasikan antara *software* dan *hardware* agar sistem dapat bekerja sesuai dengan tujuan dari pembuatan rancang bangun ini, maka diperlukan tabel pengamatan pada masing-masing *port*. Pada tabel 3.1 merupakan pengalamatan untuk *port-port* yang akan digunakan dalam pembuatan *software* pada mikrokontroller dalam sistem ini.

Tabel 3.1 Pengalamatan untuk *Port-Port* pada Mikrokontroller

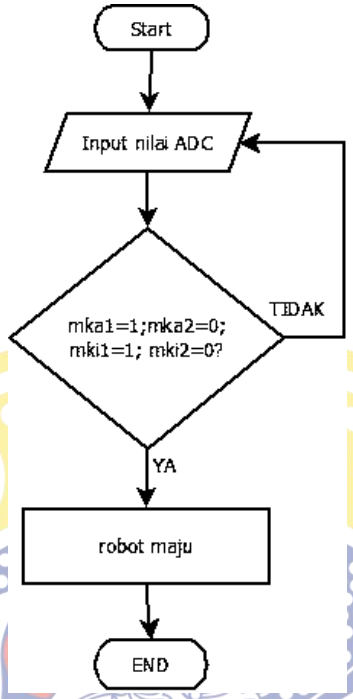
PIN Mikrokontroller	**Rangkaian Hardware**
LCD	PortB
Sensor analog	PortA (ADC)
OK	PortC4
Back	PortC5

Up	PortC6
Down	PORTC7
Start	PortC0
Stop	PortC1
Driver kanan	PortD0 dan PortD1
Pmwkanan	PortD4
Driver kiri	PortD2 dan PortD3
Pwmkiri	PortD5

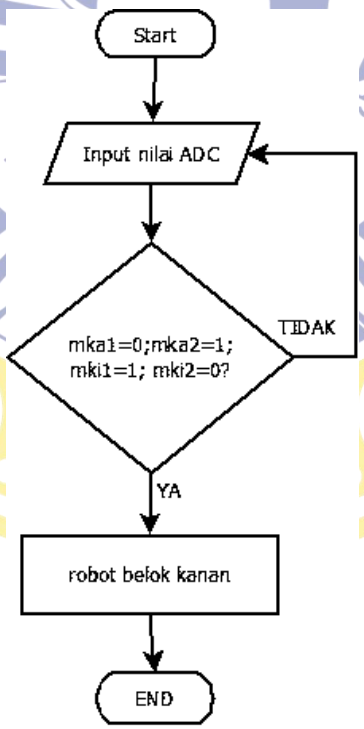
Tabel pengalamatan tersebut berfungsi untuk mempermudah dalam pembuatan *software* dari rancang bangun, agar antara *hardware* dan *software* yang dirancang saling *sinkron* satu sama lain. Setelah dilakukan pengalamatan *port* maka langkah selanjutnya yang harus dilakukan adalah membuat inisialisasi pada program. Ketika inisialisasi masing-masing program sudah ditentukan, maka langkah selanjutnya yang dilakukan adalah membuat *flowchart* dari masing-masing program, berikut *flowchart* secara keseluruhan dari sistem rancang bangun:



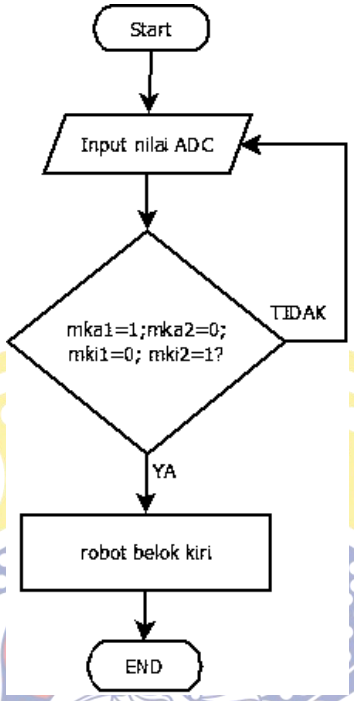
Gambar 3.4 *Flowchart* Utama



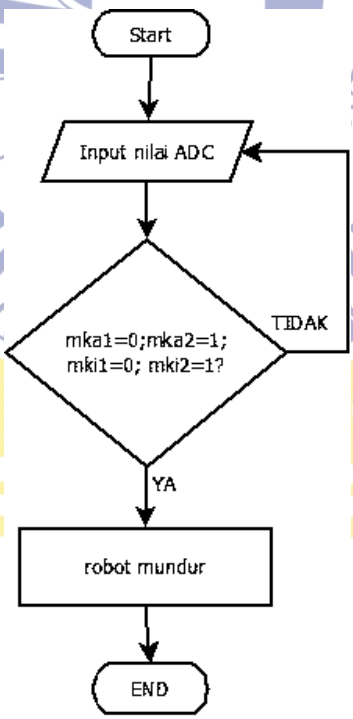
Gambar 3.5 *Flowchart* Pergerakan Maju Robot



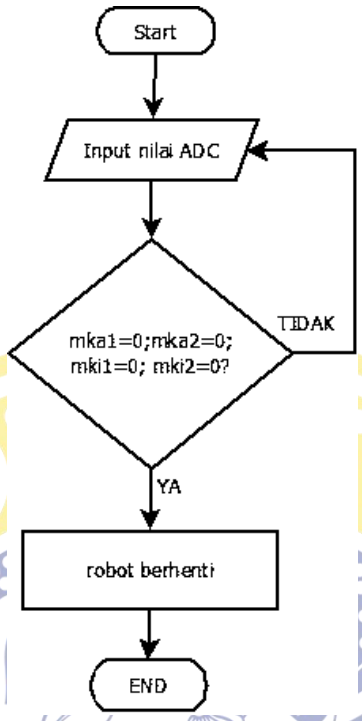
Gambar 3.6 *Flowchart* Pergerakan Belok Kanan Robot



Gambar 3.7 *Flowchart* Pergerakan Belok Kiri Robot



Gambar 3.8 *Flowchart* Pergerakan Mundur Robot



Gambar 3.9 *Flowchart* Pergerakan Berhenti Robot

3.4 Tahap Pengujian Sistem dan Analisa Data

Tahap pengujian sistem dan analisi data bertujuan untuk mengetahui kinerja *hardware*, *software* maupun sistem pada secara keseluruhan. Uji kinerja alat dan analisis data pada bagian *software* yang dilakukan dalam tugas akhir ini adalah sebagai berikut:

3.4.1 Tahap Pengujian Sensor

Pengujian pada sensor *photodiode* yaitu berupa pengambilan data sensor dengan cara mengetahui besarnya keluaran sensor dari beberapa tahap pengujian yaitu tahap pengujian sensor dengan perbedaan jarak antara sensor *photodiode* dan LED, perbedaan jarak antara rangkaian sensor *photodiode* dan bidang pantul,

perbedaan warna LED, dan juga perbedaan bidang pantul yang digunakan untuk lintasan.

Dari dua tahap awal pengujian sensor yaitu pengujian keluaran sensor dengan perbedaan jarak antara sensor *photodiode* dan LED dan juga pengujian keluaran sensor dengan perbedaan jarak rangkaian sensor dan bidang pantul maka dari dua tahap pengujian tersebut didapatkan nilai parameter ADC untuk setiap sensor *photodiode*.

Tahap pengujian sensor selanjutnya yaitu dengan mengetahui keluaran sensor yang menggunakan beberapa warna LED sebagai *transmitter*. Warna LED yang digunakan yaitu merah, hijau, kuning, dan biru. Tahap pengujian ini digunakan untuk membandingkan keluaran sensor dengan perbedaan warna LED.

Yang terakhir yaitu tahap pengujian sensor dengan perbedaan bidang pantul yang digunakan untuk lintasan robot. Ada beberapa bidang pantul yangg digunakan yaitu keramik warna abu-abu gelap, keramik warna putih, dan juga kayu. Tahap pengujian ini digunakan untuk membandingkan keluaran sensor dengan perbedaan bidang lintasan robot.

3.4.2 Tahap Kontrol Kerja Motor

Pengujian pada kontrol kerja motor berupa pengambilan data *input* logika *high* atau *low* pada *channel driver* motor untuk menentukan arah putaran motor bergerak searah jarum jam *Clock Wise* (CW) atau bergerak berlawanan arah jarum jam *Clock Centre Wise* (CCW). Pengujian ini dilakukan untuk mengetahui kesesuaian kerja *software* dan *hardware* pada pergerakan robot.

3.4.3 Tahap Pengujian Software

Pengujian *software* pada penelitian ini meliputi pengujian respon *hardware* terhadap program yang sudah ditransmisikan ke dalam mikrokontroler. Tahapan pengujian ini juga digunakan untuk mengetahui apakah alat sudah bisa membaca dan mengeksekusi perintah dari program yang sudah dibuat atau tidak.

3.4.4 Tahap Pengujian Sistem Secara Keseluruhan

Pengujian sistem secara keseluruhan bertujuan untuk mengetahui kinerja *hardware*, *software* maupun sistem pada alat secara keseluruhan yang dilakukan dengan melihat tingkat keberhasilan sistem dari pengaturan pembacaan sensor hingga eksekusi keluaran sistem berupa pergerakan motor untuk jalannya robot. Adapun persamaan yang digunakan yaitu membagi banyaknya kesalahan dengan banyaknya percobaan yang dilakukan kemudian dikalikan dengan persentase untuk mengetahui persentase keberhasilan.

BAB IV

HASIL DAN PEMBAHASAN

4.1 Perancangan Perangkat Lunak (*Software*)

Setelah pembuatan mekanik dan perangkat keras *hardware* selesai, selanjutnya yaitu dilanjutkan dengan pembuatan perangkat lunak (*software*). Pembuatan *software* pada sistem ini yaitu meliputi pengujian sensor dengan menampilkan nilai ADC pada setiap sensor analog, pengujian respon putaran dengan memberikan logika 0 atau 1 pada setiap *chanel* motor, pengujian data keseluruhan sensor terhadap respon gerakan motor dan pengujian keseluruhan sistem.

4.2 Pengujian Sensor Garis

Pengujian sensor garis analog dilakukan dengan menampilkan nilai keluaran ADC dari setiap sensor analog yaitu S1-S8. Nilai keluaran sensor akan diterima oleh ADC mikrokontroller 10 bit dan ditampilkan pada LCD untuk setiap sensor analog. Untuk pengujian sensor garis analog dilakukan pada bidang warna putih dan bidang warna hitam. Pada tabel berikut data nilai ADC yang diperoleh dari setiap sensor analog.

Tabel 4.1 Pengujian nilai ADC pada bidang putih

Sensor Analog	Nilai ADC
S1	929
S2	900
S3	895
S4	879
S5	904
S6	913
S7	919
S8	928

Tabel 4.2 Pengujian nilai ADC pada bidang hitam

Sensor Analog	Nilai ADC
S1	1816
S2	1007
S3	999
S4	1020
S5	1016
S6	999
S7	1008
S8	1007

Berdasarkan data yang diperoleh dari tabel tersebut dapat disimpulkan bahwa parameter nilai ADC untuk sensor yang mendeteksi warna putih yaitu kurang dari 950 sedangkan parameter nilai ADC untuk sensor yang mendeteksi warna hitam yaitu lebih dari 950. Berikut program untuk mengkonversi nilai ADC yang dihasilkan menjadi masukan untuk respon motor:

```
#include <mega16.h>
#include <stdio.h>
#include <delay.h>
unsigned char buff[46];
unsigned int S1,S2,S3,S4,S5,S6,S7,S8;

void baca_adc(){
S1=read_adc(0);
S2=read_adc(1);
S3=read_adc(2);
S4=read_adc(3);
S5=read_adc(4);
S6=read_adc(5);
S7=read_adc(6);
S8=read_adc(7);
```

Program baca ADC

```
lcd_gotoxy(0,1);
sprintf(buff, "%3d %3d %3d %3d", S1, S2, S3, S4);
lcd_puts(buff);
lcd_gotoxy(0,0);
sprintf(buff, "%3d %3d %3d %3d", S5, S6, S7,S8);
lcd_puts(buff);
}
```

Program untuk menampilkan nilai ADC pada

Berdasarkan listing di atas diketahui bahwa sensor photodiode terletak pada PORT ADC mikrokontroller yaitu PORT A0-A7. Keluaran dari sensor akan dikonversi menjadi digital dan di tampilkan pada LCD.

4.3 Pengujian Driver Motor

Pengujian ini dilakukan pada motor untuk pergerakan robot, yang bertujuan untuk mengetahui kesesuaian kerja *software* dengan *hardware* yang telah dibuat. Pengujian pada motor sebagai pergerakan robot untuk *control* arah putaran motor yang dikendalikan dengan *driver* motor, dilakukan dengan cara pengambilan data keadaan *high* atau *low port* arah perputaran motor untuk pergerakan robot. Berikut data pengujian pada motor kanan dan motor kiri:

Tabel 4.3 Pengujian Putaran Motor Kanan

Motor Kanan		Keterangan
Mka1 (PORT D2)	Mka2 (PORT D3)	
1	0	CW
0	1	CCW

Tabel 4.4 Pengujian Putaran Motor Kiri

Motor Kiri		Keterangan
Mki1 (PORT D0)	Mki2 (PORT D1)	
1	0	CW
0	1	CCW

Dari hasil pengujian dapat diketahui bahwa jika Mka1 diberi logika "1" dan Mka2 diberi logika "0" maka respon putaran motor searah jarum jam (CW) sedangkan jika Mka1 diberi logika "0" dan Mka2 diberi logika "1" maka respon putaran motor berlawanan arah jarum jam (CCW). Dan jika Mki1 diberi logika "1" dan Mki2 diberi logika "0" maka respon putaran motor searah jarum jam (CW) sedangkan jika Mki1 diberi logika "0" dan Mki2 diberi logika "1" maka respon putaran motor berlawanan arah jarum jam (CCW).

Berikut adalah program untuk pengujian respon motor:

```
#include <mega16.h>
#include <stdio.h>
#include <delay.h>
#define mki1  PORTD.0
#define mki2  PORTD.1
#define mka1  PORTD.2
#define mka2  PORTD.3
#define pwmki OCR1B
#define pwmka OCR1A
```

Inisialisasi port mikrokontroller pada setiap *driver* motor

```
void go(unsigned int L , unsigned int R){
pwmki=L;
pwmka=R;
}
```

Program untuk jalannya robot

```
void maju(){
mka1=1;mka2=0;
mki1=1;mki2=0;
}
```

Robot bergerak maju

```
void mundur(){
mka1=0;mka2=1;
mki1=0;mki2=1;
}
```

Robot bergerak mundur

```
void beka(){
mka1=0;mka2=1;
mki1=1;mki2=0;
}
```

Robot bergerak belok kanan

```
void beki(){
mka1=1;mka2=0;
mki1=0;mki2=1;
}
```

Robot bergerak belok kiri

```
void henti(){
mka1=0;mka2=0;
mki1=0;mki2=0; }
```

Robot berhenti

Berdasarkan program di atas dapat diketahui bahwa pada *driver* motor kanan dan motor kiri terletak pada PORT D0-D3 pada mikrokontroller. Dari program tersebut didapatkan data untuk pengujian respon motor terhadap pemberian logika “0” atau “1” pada setiap *port* mikrokontroller yang terhubung dengan *chanel driver* motor. Berikut adalah tabel hasil pengujian respon motor:

Tabel 4.5 Pengujian Respon Kedua Motor

Motor Kanan		Motor Kiri		Respon Motor
PORTD2	PORTD3	PORTD0	PORTD1	
1	0	1	0	Maju
0	1	0	1	Mundur
1	0	0	1	Belok kiri (beki)
0	1	1	0	Belok Kanan (beka)
0	0	0	0	Berhenti

Berdasarkan tabel di atas dapat diketahui bahwa pemberian logika “0” atau “1” pada setiap *chanel driver* motor mempengaruhi arah kerja motor. Dari hasil tersebut dapat diketahui bahwa *input* berupa logika sesuai dengan *output* yang diharapkan.

Setelah mendapatkan *input* logika yang sesuai dengan yang diinginkan maka pengujian selanjutnya yaitu dengan menguji PWM (*Pulse with Modulation*) motor dengan perbedaan variasi sudut. Berikut adalah tabel pengujian PWM terhadap variasi sudut:

Tabel 4.6 Pengujian Respon Motor Terhadap Variasi Sudut

Sudut (º)	Keterangan
0	Mampu
10	Mampu
20	Mampu
30	Mampu
40	Mampu
50	Mampu
60	Mampu
70	Mampu
80	Mampu
90	Mampu
100	Tidak Mampu
110	Tidak Mampu
120	Tidak Mampu
130	Tidak Mampu
140	Tidak Mampu
150	Tidak Mampu
160	Tidak Mampu
170	Tidak Mampu
180	Tidak Mampu

Dari Tabel 4.6 dapat diketahui bahwa respon pergerakan robot terhadap variasi sudut jalur yang diberikan terhadap PWM yang sudah ditentukan yaitu 180 rpm pada sudut 0º-90º robot masih mampu berjalan dengan PWM yang sama dan pada sudut 100º-180º sudah tidak mampu berjalan.

4.4 Pengujian Seluruh Sistem

Pengujian seluruh sistem dilakukan dengan melihat tingkat keberhasilan sistem dari pengaturan parameter masukan nilai ADC pada setiap sensor analog, hingga keluaran sistem berupa respon gerakan motor DC terhadap jalannya robot. Pengambilan data dilakukan dengan merubah parameter ADC pada masing-masing sensor analog sehingga akan diketahui eksekusi dari sistem. Untuk parameter ADC yang digunakan untuk eksekusi program yaitu nilai ADC<950 untuk mendeteksi bidang putih dan nilai ADC>950 untuk mendeteksi garis hitam. Pengujian ini diambil dari jarak sensor dan *photodiode* yang sudah ditentukan yaitu 0,9 cm dan juga jarak modul sensor dengan bidang pantul yaitu 1,5 cm. Hasil dari pengujian keseluruhan data sensor terhadap respon motor dapat dilihat pada tabel 4.7 berikut:

Tabel 4.7 Pengujian Keseluruhan Data Sensor Terhapap Jalan Robot

Sensor Analog								Jalan Robot
S1	S2	S3	S4	S5	S6	S7	S8	
<950	<950	<950	>950	>950	<950	<950	<950	Maju
<950	<950	<950	<950	>950	<950	<950	<950	Maju
<950	<950	<950	>950	<950	<950	<950	<950	Maju
<950	<950	<950	<950	>950	>950	<950	<950	Maju
<950	<950	<950	>950	>950	>950	<950	<950	Maju
<950	<950	>950	>950	<950	<950	<950	<950	Maju
<950	<950	>950	>950	>950	<950	<950	<950	Maju
<950	<950	<950	<950	<950	>950	<950	<950	Maju
<950	<950	>950	<950	<950	<950	<950	<950	Maju
>950	>950	>950	>950	<950	<950	<950	<950	Maju
<950	<950	<950	<950	>950	>950	>950	>950	Maju
<950	<950	<950	<950	<950	>950	>950	<950	Belok Kiri

<950	<950	<950	<950	<950	>950	>950	>950	Belok Kiri
<950	<950	<950	<950	<950	<950	<950	>950	Belok Kiri
<950	<950	<950	<950	<950	>950	>950	>950	Belok Kiri
<950	<950	<950	<950	>950	>950	>950	<950	Belok Kiri
<950	>950	>950	<950	<950	<950	<950	<950	Belok Kanan
>950	>950	<950	<950	<950	<950	<950	<950	Belok Kanan
>950	<950	<950	<950	<950	<950	<950	<950	Belok Kanan
>950	>950	>950	<950	<950	<950	<950	<950	Belok Kanan
<950	>950	>950	>950	<950	<950	<950	<950	Belok Kanan
<950	<950	<950	<950	<950	<950	<950	<950	Mundur
>950	>950	>950	>950	>950	>950	>950	>950	Berhenti

Dari hasil percobaan pada Tabel dapat diketahui kinerja dari keseluruhan sistem yakni berupa tindakan eksekusi sistem bekerja pada *set point* yang telah ditentukan untuk setiap sensor analog. Eksekusi sistem dapat dilihat hasilnya dari respon gerakan motor untuk jalannya robot.

Setelah keseluruhan sistem diuji untuk mendapatkan hasil dari kinerja alat mulai dari awal sampai akhir, untuk mendapatkan hasil yang diharapkan maka dilakukan beberapa percobaan. Pengujian robot dilakukan mulai awal sampai dengan robot berhasil dijalankan. Pada Tabel 4.8 berikut adalah hasil percobaan keseluruhan program:

Tabel 4.8 Pengujian Seluruh Sistem

Percobaan ke-	Hasil Pengujian Jalan Robot
1	Berhasil
2	Berhasil
3	Berhasil
4	Berhasil
5	Berhasil
6	Berhasil
7	Berhasil
8	Berhasil
9	Berhasil
10	Berhasil

Dari pengujian tersebut dianggap berhasil jika robot bisa berjalan dari garis *start* sampai garis *finish* pada jalur atau lintasan robot yang sudah ditentukan yaitu banner putih dan keramik putih dengan perbedaan desain lintasan (garis

hitam) pada bidang lintasan. Berdasarkan percobaan diatas dapat diperoleh keberhasilan dari uji program keseluruhan.

$$\text{Persentase Keberhasilan} = \frac{nPercobaan - nKesalahan}{nPercobaan} \times 100\%$$

$$= \frac{10-0}{10} \times 100\%$$

$$= 100\ \%$$

Dari perhitungan diatas didapatkan persentase keberhasilan sistem yaitu 100% sehingga dapat diketahui bahwa sistem bekerja sesuai dengan hasil yang diharapkan.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dari kegiatan pengujian tugas akhir dengan judul "Rancang Bangun Robot *Line Follower* Berbasis Cahaya Tampak" dapat menarik suatu kesimpulan sebagai berikut :

1. Dari penelitian ini didapatkan hasil berupa rancang bangun robot *line follower* berbasis cahaya tampak dengan desain mekanik sensor lurus berdampingan dimana jarak antar sensor sebesar 2 cm, jarak sensor *photodiode* dengan LED sebesar 0,9 cm, dan jarak modul sensor garis dengan bidang pantul sebesar 1,5 cm, sedangkan untuk pembuatan sistem *software* yaitu dilakukan dengan pemberian nilai ADC sebagai *control* motor pada setiap sensor garis dimana parameter nilai ADC untuk bidang hitam >950 dan bidang putih <950.
2. Dari hasil pengujian seluruh sistem dapat diketahui kinerja dari sistem yakni dengan menganalisa beberapa hasil percobaan dan membandingkan tingkat kaberhasilan dan kegagalan sistem, dapat diketahui bahwa sistem mempunyai persentase keberhasilan 100% dimana pengujian ini dilakukan pada bidang lintasan yang berbeda yaitu keramik putih dan banner putih dengan berbagai *desain* lintasan (garis hitam) yang berbeda. Hal ini menunjukkan bahwa sistem dapat bekerja sesuai dengan hasil yang diharapkan.

5.2 Saran

Saran yang dapat penulis sampaikan adalah Robot ini masih terdapat banyak kekurangan, diharapkan untuk penelitian selanjutnya bisa dilengkapi dengan *software* optimasi PID atau *Fuzzy logic* sehingga didapatkan pergerakan yang *smooth*.

DAFTAR PUSTAKA

Baskara. “Dasar Teori ATMEGA16”. http:// baskarapunya. blogspot.co.id/ 2012/ 09/ dasar-teori-atmega16.html . diakses pada 17 desember 2015 jam 01:57

Elizer,I.P.G. "Merancang Driver Motor DC“. http:// www.geyosoft.com/ 2014/ merancang-driver-motor-dc. diakses pada 17 desember 2015 jam 01:48

Falani, Achmad Zakki; Setyawan Budi. 2015. “Robot Line Follower Berbasis Mikrokontroler ATmega 16 dengan Menampilkan Status Gerak Pada LCD”. Surabaya : Fakultas Ilmu Komputer Prodi Sistem Komputer, Universitas Narotama.

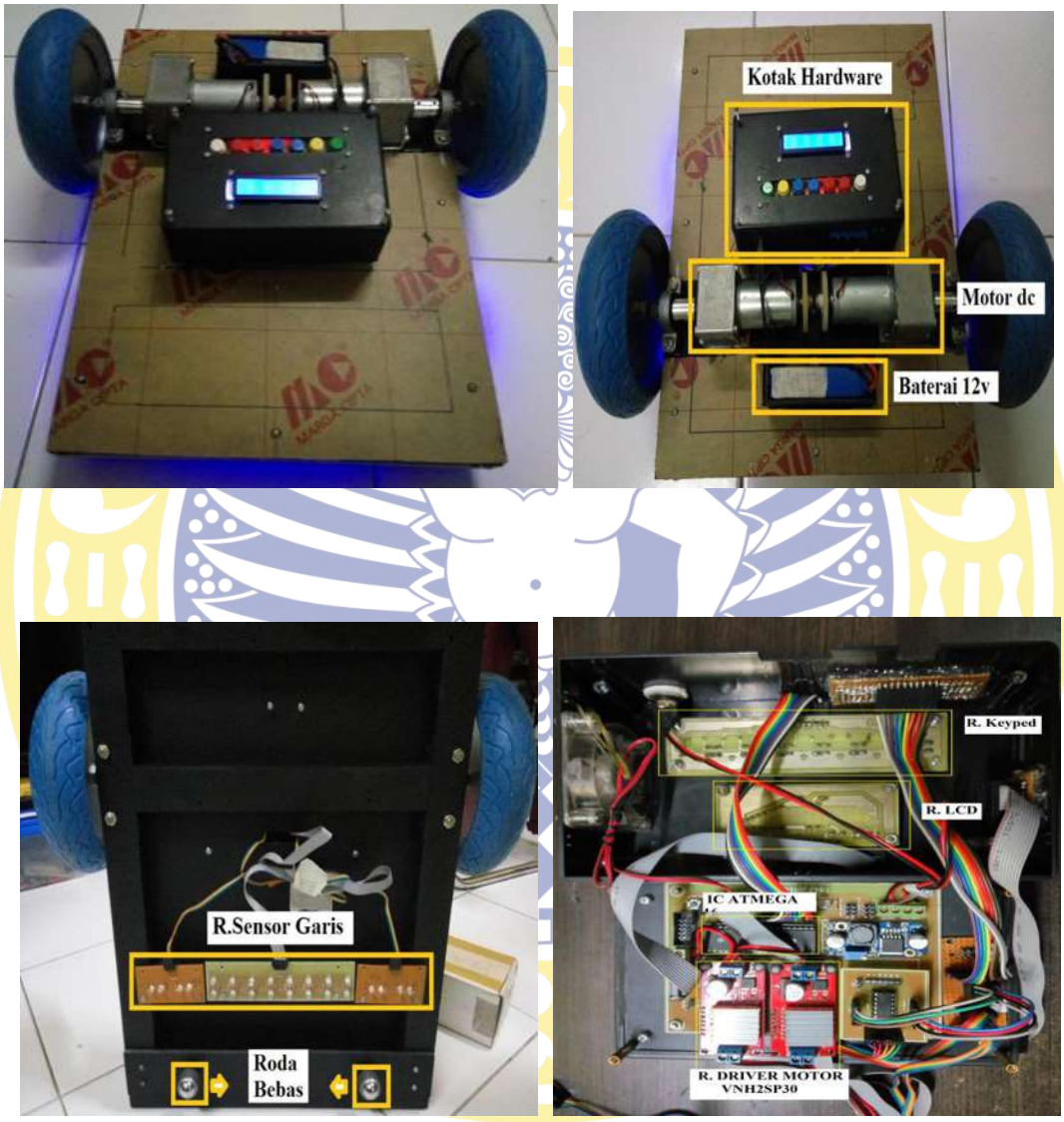
Immersa Lab. “Sistem Minimum Mikrokontroler” http:// www.immersa-lab.com/ sistem-minimum-mikrokontroler.html . diakses pada 17 desember 2015 jam 1:51

Janis, Daisy A. N; David Pang; J. O. Wuwung. 2014. “Rancang Bangun Robot Pengantar Makanan Line Follower”. Manado : Jurusan Teknik Elektro FT, UNSRAT.

Margiono, Abdillah. “Sensor Cahaya *Photodiode”* http:// margionoabdil. blogspot.co.id/ 2015/03/ sensor-cahaya-photo-diode.html. Diakses pada 17 Desember 2015 jam 00:52

LAMPIRAN

Hasil Rancang Bangun Robot *Line Follower* Berbasis Cahaya Tampak



```
/*****************************************************
This program was produced by the
CodeWizardAVR V2.05.0 Professional
Automatic Program Generator
© Copyright 1998-2010 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com

Project : family robot
Version :
Date    : 7/26/2016
Author  : NeVaDa
Company :
Comments:


Chip type           : ATmega16
Program type        : Application
AVR Core Clock frequency: 11.059200 MHz
Memory model        : Small
External RAM size   : 0
Data Stack size     : 256
*****************************************************/

#include <mega16.h>
#include <stdio.h>

#include <delay.h>
#define mki1  PORTD.0
#define mki2  PORTD.1
#define mka1  PORTD.2
#define mka2  PORTD.3
#define pwmki OCR1B
#define pwmka OCR1A
#define start PINC.0
#define stop PINC.1
#define ok PINC.4
```

```
#define back PINC.5
#define up PINC.6
#define down PINC.7

// Alphanumeric LCD Module functions
#asm
   .equ __lcd_port=0x18 ;PORTB
#endasm
#include <alcd.h>

#define ADC_VREF_TYPE 0x00

// Read the AD conversion result
unsigned int read_adc(unsigned char adc_input)
{
ADMUX=adc_input | (ADC_VREF_TYPE & 0xff);
// Delay needed for the stabilization of the ADC input voltage
delay_us(10);
// Start the AD conversion
ADCSRA|=0x40;
// Wait for the AD conversion to complete
while ((ADCSRA & 0x10)==0);
ADCSRA|=0x10;
return ADCW;
}

void go(unsigned int L , unsigned int R){
pwmki=L;
pwmka=R;
}
void maju(){
 mka1=1;mka2=0;
 mki1=1;mki2=0;
}

void mundur(){
 mka1=0;mka2=1;
```

```
 mki1=0;mki2=1;
}

void beka(){
 mka1=0;mka2=1;
 mki1=1;mki2=0;
}

void beki(){
 mka1=1;mka2=0;
 mki1=0;mki2=1;
}

void henti(){
 mka1=0;mka2=0;
 mki1=0;mki2=0;

}

// Declare your global variables here

unsigned char buff[46];
unsigned int S1,S2,S3,S4,S5,S6,S7,S8;
int step;


//########################## BACA ADC ########################
void baca_adc(){
    S1=read_adc(0);
    S2=read_adc(1);
    S3=read_adc(2);
    S4=read_adc(3);
    S5=read_adc(4);
    S6=read_adc(5);
    S7=read_adc(6);
    S8=read_adc(7);
    lcd_gotoxy(0,1);
```

```
    sprintf(buff, "%3d %3d %3d %3d", S1, S2, S3, S4);
    lcd_puts(buff);
    lcd_gotoxy(0,0);
    sprintf(buff, "%3d %3d %3d %3d", S5, S6, S7,S8);
    lcd_puts(buff);
}



//##### Variable nilai eeprom #####
   eeprom int adc0 = 100;
   eeprom int adc1 = 100;
   eeprom int adc2 = 100;
   eeprom int adc3 = 100;
   eeprom int adc4 = 100;
   eeprom int adc5 = 100;
   eeprom int adc6 = 100;
   eeprom int adc7 = 100;

//######################### Loading Screen ########################
void loading()
{
   int i;
   lcd_gotoxy(4,0);
   lcd_putsf("TA 2016");              // loading screen
   for(i=2;i<=14;i++)
                {
                lcd_gotoxy(i,1);
                lcd_putchar(0xFF);
                delay_ms(75);
                }
}

//################## SAVE TAMPIL ####################
void convert(unsigned int hasil)
```

```
{
unsigned int i;

i=hasil/100;
lcd_putchar(48+i);
i=hasil%100;
i/=10;
lcd_putchar(48+i);
hasil%=10;
lcd_putchar(48+hasil);
}


//################################ SAVE EEPROM ################################
void tulis_data_ke_eeprom()
{
   adc0 = adc0;
   adc1 = adc1;
   adc2 = adc2;
   adc3 = adc3;
   adc4 = adc4;
   adc5 = adc5;
   adc6 = adc6;
   adc7 = adc7;
};

void save_eeprom()
{
      int j;
      lcd_gotoxy(0, 0);
      lcd_putsf("  Saving EEPROM ");
      for(j=2;j<=14;j++)
   {
   lcd_gotoxy(j,1);
   lcd_putchar(0xFF);
   delay_ms(50);
   }
```

```
};


//################################ GUI ################################
typedef unsigned char byte;
flash byte char0[8]={
0b0001000,
0b0000100,
0b0000010,
0b1111111,
0b1111111,
0b0000010,
0b0000100,
0b0001000
};

void define_char(byte flash *pc,byte char_code)
{
byte i,a;
a=(char_code<<3) | 0x40;
for (i=0; i<8; i++) lcd_write_byte(a++,*pc++);}

void gui()
{
   lcd_clear();
menu1:          //adc
   delay_ms(150);
   lcd_gotoxy(2,0);
   lcd_putsf("Set ADC");
   lcd_gotoxy(2,1);
   lcd_putsf("Start Robot");
   lcd_gotoxy(0,0);
   lcd_putchar(0);
   if (ok==0){lcd_clear();goto setadc0;}
   if (down==0){lcd_clear(); goto menu2;}
   if (up==0){lcd_clear(); goto start_robot;}
   if (start==0){lcd_clear(); step=0;}
```

```
    goto menu1;

menu2:          // start robot
    delay_ms(150);
    lcd_gotoxy(2,0);
    lcd_putsf("Set ADC");
    lcd_gotoxy(2,1);
    lcd_putsf("Baca Sensor");
    lcd_gotoxy(0,1);
    lcd_putchar(0);
    if (ok==0){lcd_clear(); goto bacaadc;}
    if (down==0){lcd_clear(); goto menu3;}
    if (up==0){lcd_clear(); goto menu1;}
    if (start==0){lcd_clear(); step=0;}
    goto menu2;

menu3:          // start robot
    delay_ms(150);
    lcd_gotoxy(2,0);
    lcd_putsf("Baca Sensor");
    lcd_gotoxy(2,1);
    lcd_putsf("Start Robot");
    lcd_gotoxy(0,1);
    lcd_putchar(0);
    if (ok==0){lcd_clear(); goto start_robot;}
    if (down==0){lcd_clear(); goto menu1;}
    if (up==0){lcd_clear(); goto menu2;}
    if (start==0){lcd_clear(); step=0;}
    goto menu3;

bacaadc:
    delay_ms(150);
    if (ok==0){lcd_clear(); baca_adc();}
    goto bacaadc;

setadc0:
    delay_ms(150);
```

```
    lcd_gotoxy(2,0);
    lcd_putsf("ADC0   ADC1");
    lcd_gotoxy(2,1);
    lcd_putsf("ADC2   ADC3");
    lcd_gotoxy(0,0);
    lcd_putchar(0);
    if (ok==0){lcd_clear(); goto settingadc0;}
    if (down==0){lcd_clear(); goto setadc1;}
    if (up==0){lcd_clear(); goto setadc7;}
    if (back==0){lcd_clear(); goto menu1;}
    if (start==0){lcd_clear(); step=0;}
    goto setadc0;

setadc1:
    delay_ms(150);
    lcd_gotoxy(2,0);
    lcd_putsf("ADC0   ADC1");
    lcd_gotoxy(2,1);
    lcd_putsf("ADC2   ADC3");
    lcd_gotoxy(7,0);
    lcd_putchar(0);
    if (ok==0){lcd_clear(); goto settingadc1;}
    if (down==0){lcd_clear(); goto setadc2;}
    if (up==0){lcd_clear(); goto setadc0;}
    if (back==0){lcd_clear(); goto menu1;}
    if (start==0){lcd_clear(); step=0;}
    goto setadc1;

setadc2:
    delay_ms(150);
    lcd_gotoxy(2,0);
    lcd_putsf("ADC0   ADC1");
    lcd_gotoxy(2,1);
    lcd_putsf("ADC2   ADC3");
    lcd_gotoxy(0,1);
    lcd_putchar(0);
    if (ok==0){lcd_clear(); goto settingadc2;}
```

```
    if (down==0){lcd_clear(); goto setadc3;}
    if (up==0){lcd_clear(); goto setadc1;}
    if (back==0){lcd_clear(); goto menu1;}
    if (start==0){lcd_clear(); step=0;}
    goto setadc2;

setadc3:
    delay_ms(150);
    lcd_gotoxy(2,0);
    lcd_putsf("ADC0   ADC1");
    lcd_gotoxy(2,1);
    lcd_putsf("ADC2   ADC3");
    lcd_gotoxy(7,1);
    lcd_putchar(0);
    if (ok==0){lcd_clear(); goto settingadc3;}
    if (down==0){lcd_clear(); goto setadc4;}
    if (up==0){lcd_clear(); goto setadc2;}
    if (back==0){lcd_clear(); goto menu1;}
    if (start==0){lcd_clear(); step=0;}
    goto setadc3;

setadc4:
    delay_ms(150);
    lcd_gotoxy(2,0);
    lcd_putsf("ADC4   ADC5");
    lcd_gotoxy(2,1);
    lcd_putsf("ADC6   ADC7");
    lcd_gotoxy(0,0);
    lcd_putchar(0);
    if (ok==0){lcd_clear(); goto settingadc4;}
    if (down==0){lcd_clear(); goto setadc5;}
    if (up==0){lcd_clear(); goto setadc3;}
    if (back==0){lcd_clear(); goto menu1;}
    if (start==0){lcd_clear(); step=0;}
    goto setadc4;

setadc5:
```

```
    delay_ms(150);
    lcd_gotoxy(2,0);
    lcd_putsf("ADC4   ADC5");
    lcd_gotoxy(2,1);
    lcd_putsf("ADC6   ADC7");
    lcd_gotoxy(7,0);
    lcd_putchar(0);
    if (ok==0){lcd_clear(); goto settingadc5;}
    if (down==0){lcd_clear(); goto setadc6;}
    if (up==0){lcd_clear(); goto setadc4;}
    if (back==0){lcd_clear(); goto menu1;}
    if (start==0){lcd_clear(); step=0;}
    goto setadc5;

setadc6:
    delay_ms(150);
    lcd_gotoxy(2,0);
    lcd_putsf("ADC4   ADC5");
    lcd_gotoxy(2,1);
    lcd_putsf("ADC6   ADC7");
    lcd_gotoxy(0,1);
    lcd_putchar(0);
    if (ok==0){lcd_clear(); goto settingadc6;}
    if (down==0){lcd_clear(); goto setadc7;}
    if (up==0){lcd_clear(); goto setadc5;}
    if (back==0){lcd_clear(); goto menu1;}
    if (start==0){lcd_clear(); step=0;}
    goto setadc6;

setadc7:
    delay_ms(150);
    lcd_gotoxy(2,0);
    lcd_putsf("ADC4   ADC5");
    lcd_gotoxy(2,1);
    lcd_putsf("ADC6   ADC7");
    lcd_gotoxy(7,1);
    lcd_putchar(0);
```

```
    if (ok==0){lcd_clear(); goto settingadc7;}
    if (down==0){lcd_clear(); goto setadc0;}
    if (up==0){lcd_clear(); goto setadc6;}
    if (back==0){lcd_clear(); goto menu1;}
    if (start==0){lcd_clear(); step=0;}
    goto setadc7;

//############################### ubah nilai adc ############################
settingadc0:                        // mengubah nilai adc0
    delay_ms(150);
    if (up==0){adc0++;
          if (adc0>1000) adc0=0;
          else if (adc0<0) adc0=1000;};
    if (down==0){adc0--;
          if (adc0>1000) adc0=0;
          else if (adc0<0) adc0=1000;};
    if (ok==0){ tulis_data_ke_eeprom(); save_eeprom(); lcd_clear(); goto setadc0;};
    lcd_gotoxy(2,0);
    lcd_putsf("ADC0 :");
    convert(adc0);
    lcd_gotoxy(0,0);
    lcd_putchar(0);
    if (back==0){lcd_clear();goto setadc0;}
    if (start==0){lcd_clear(); step=0;}
    goto settingadc0;

settingadc1:                        // mengubah nilai adc1
    delay_ms(150);
    if (up==0){adc1++;
          if (adc1>1000) adc1=0;
          else if (adc1<0) adc1=1000;};
    if (down==0){adc1--;
          if (adc1>1000) adc1=0;
          else if (adc1<0) adc1=1000;};
    if (ok==0){ tulis_data_ke_eeprom(); save_eeprom(); lcd_clear(); goto setadc1;};
    lcd_gotoxy(2,0);
    lcd_putsf("ADC1 :");
```

```
    convert(adc1);
    lcd_gotoxy(0,0);
    lcd_putchar(0);
    if (back==0){lcd_clear();goto setadc1;}
    if (start==0){lcd_clear(); step=0;}
    goto settingadc1;

settingadc2:                    // mengubah nilai adc2
    delay_ms(150);
    if (up==0){adc2++;
          if (adc2>1000) adc2=0;
          else if (adc2<0) adc2=1000;};
    if (down==0){adc2--;
          if (adc2>1000) adc2=0;
          else if (adc2<0) adc2=1000;};
    if (ok==0){ tulis_data_ke_eeprom(); save_eeprom(); lcd_clear(); goto setadc2;};
    lcd_gotoxy(2,0);
    lcd_putsf("ADC2 :");
    convert(adc2);
    lcd_gotoxy(0,0);
    lcd_putchar(0);
    if (back==0){lcd_clear();goto setadc2;}
    if (start==0){lcd_clear(); step=0;}
    goto settingadc2;

settingadc3:                    // mengubah nilai adc3
    delay_ms(150);
    if (up==0){adc3++;
          if (adc3>1000) adc3=0;
          else if (adc3<0) adc3=1000;};
    if (down==0){adc3--;
          if (adc3>1000) adc3=0;
          else if (adc3<0) adc3=1000;};
    if (ok==0){ tulis_data_ke_eeprom(); save_eeprom(); lcd_clear(); goto setadc3;};
    lcd_gotoxy(2,0);
    lcd_putsf("ADC3 :");
    convert(adc3);
```

```
    lcd_gotoxy(0,0);
    lcd_putchar(0);
    if (back==0){lcd_clear();goto setadc3;}
    if (start==0){lcd_clear(); step=0;}
    goto settingadc3;

settingadc4:                    // mengubah nilai adc4
    delay_ms(150);
    if (up==0){adc4++;
          if (adc4>1000) adc4=0;
          else if (adc4<0) adc4=1000;};
    if (down==0){adc4--;
          if (adc4>1000) adc4=0;
          else if (adc4<0) adc4=1000;};
    if (ok==0){ tulis_data_ke_eeprom(); save_eeprom(); lcd_clear(); goto setadc4;};
    lcd_gotoxy(2,0);
    lcd_putsf("ADC4 :");
    convert(adc4);
    lcd_gotoxy(0,0);
    lcd_putchar(0);
    if (back==0){lcd_clear();goto setadc4;}
    if (start==0){lcd_clear(); step=0;}
    goto settingadc4;

settingadc5:                    // mengubah nilai adc5
    delay_ms(150);
    if (up==0){adc5++;
          if (adc5>1000) adc5=0;
          else if (adc5<0) adc5=1000;};
    if (down==0){adc5--;
          if (adc5>1000) adc5=0;
          else if (adc5<0) adc5=1000;};
    if (ok==0){ tulis_data_ke_eeprom(); save_eeprom(); lcd_clear(); goto setadc5;};
    lcd_gotoxy(2,0);
    lcd_putsf("ADC5 :");
    convert(adc5);
    lcd_gotoxy(0,0);
```

```
    lcd_putchar(0);
    if (back==0){lcd_clear();goto setadc5;}
    if (start==0){lcd_clear(); step=0;}
    goto settingadc5;

settingadc6:                    // mengubah nilai adc6
    delay_ms(150);
    if (up==0){adc6++;
          if (adc6>1000) adc6=0;
          else if (adc6<0) adc6=1000;};
    if (down==0){adc6--;
          if (adc6>1000) adc6=0;
          else if (adc6<0) adc6=1000;};
    if (ok==0){ tulis_data_ke_eeprom(); save_eeprom(); lcd_clear(); goto setadc6;};
    lcd_gotoxy(2,0);
    lcd_putsf("ADC6 :");
    convert(adc6);
    lcd_gotoxy(0,0);
    lcd_putchar(0);
    if (back==0){lcd_clear();goto setadc6;}
    if (start==0){lcd_clear(); step=0;}
    goto settingadc6;

settingadc7:                    // mengubah nilai adc7
    delay_ms(150);
    if (up==0){adc7++;
          if (adc7>1000) adc7=0;
          else if (adc7<0) adc7=1000;};
    if (down==0){adc7--;
          if (adc7>1000) adc7=0;
          else if (adc7<0) adc7=1000;};
    if (ok==0){ tulis_data_ke_eeprom(); save_eeprom(); lcd_clear(); goto setadc7;};
    lcd_gotoxy(2,0);
    lcd_putsf("ADC7 :");
    convert(adc7);
    lcd_gotoxy(0,0);
    lcd_putchar(0);
```

```
   if (back==0){lcd_clear();goto setadc7;}
   if (start==0){lcd_clear(); step=0;}
   goto settingadc7;

start_robot:
   if (ok==0) {lcd_clear(); step=0;}
}

void proses(){

                if(S1<adc0&&S2<adc1&&S3<adc2&&S4>adc3&&S5>adc4&&S6<adc5&&S7<ad
                c6&&S8<adc7){maju();go(180,180);}    //4,5
     else
                if(S1<adc0&&S2<adc1&&S3<adc2&&S4>adc3&&S5<adc4&&S6<adc5&&S7<ad
                c6&&S8<adc7){maju();go(180,180);}   //4
     else
                if(S1<adc0&&S2<adc1&&S3<adc2&&S4<adc3&&S5>adc4&&S6<adc5&&S7<ad
                c6&&S8<adc7){maju();go(180,180);}   //5
     else
                if(S1<adc0&&S2<adc1&&S3<adc2&&S4<adc3&&S5>adc4&&S6>adc5&&S7<ad
                c6&&S8<adc7){maju();go(180,180);}  //56
     else
                if(S1<adc0&&S2<adc1&&S3>adc2&&S4>adc3&&S5<adc4&&S6<adc5&&S7<ad
                c6&&S8<adc7){maju();go(180,180);}   //34
     else
                if(S1<adc0&&S2<adc1&&S3<adc2&&S4>adc3&&S5>adc4&&S6>adc5&&S7<ad
                c6&&S8<adc7){maju();go(180,180);}   //456
     else
                if(S1<adc0&&S2<adc1&&S3>adc2&&S4>adc3&&S5>adc4&&S6<adc5&&S7<ad
                c6&&S8<adc7){maju();go(180,180);}  //345
     else
                if(S1<adc0&&S2<adc1&&S3>adc2&&S4>adc3&&S5>adc4&&S6>adc5&&S7<ad
                c6&&S8<adc7){maju();go(180,180);}   //3456
```

```
else
        if(S1<adc0&&S2<adc1&&S3<adc2&&S4<adc3&&S5<adc4&&S6>adc5&&S7<ad
        c6&&S8<adc7){beki();go(180,180);}   //6
else
        if(S1<adc0&&S2<adc1&&S3<adc2&&S4<adc3&&S5<adc4&&S6<adc5&&S7>ad
        c6&&S8<adc7){beki();go(180,180);}   //7
else
        if(S1<adc0&&S2<adc1&&S3<adc2&&S4<adc3&&S5<adc4&&S6<adc5&&S7<ad
        c6&&S8>adc7){beki();go(180,180);}   //8
else
        if(S1<adc0&&S2<adc1&&S3<adc2&&S4<adc3&&S5<adc4&&S6>adc5&&S7>ad
        c6&&S8<adc7){beki();go(180,180);}   //67
else
        if(S1<adc0&&S2<adc1&&S3<adc2&&S4<adc3&&S5<adc4&&S6<adc5&&S7>ad
        c6&&S8>adc7){beki();go(180,180);}   //78
else
        if(S1<adc0&&S2<adc1&&S3<adc2&&S4<adc3&&S5>adc4&&S6>adc5&&S7>ad
        c6&&S8<adc7){beki();go(180,180);}   //567
else
        if(S1<adc0&&S2<adc1&&S3<adc2&&S4<adc3&&S5<adc4&&S6>adc5&&S7>ad
        c6&&S8>adc7){beki();go(180,180);}   //678
else
        if(S1<adc0&&S2<adc1&&S3<adc2&&S4<adc3&&S5>adc4&&S6>adc5&&S7>ad
        c6&&S8>adc7){beki();go(180,180);}   //5678

else
        if(S1>adc0&&S2<adc1&&S3<adc2&&S4<adc3&&S5<adc4&&S6<adc5&&S7<ad
        c6&&S8<adc7){beka();go(180,180);}   //1
else
        if(S1<adc0&&S2>adc1&&S3<adc2&&S4<adc3&&S5<adc4&&S6<adc5&&S7<ad
        c6&&S8<adc7){beka();go(180,180);}   //2
else
        if(S1<adc0&&S2<adc1&&S3>adc2&&S4<adc3&&S5<adc4&&S6<adc5&&S7<ad
        c6&&S8<adc7){beka();go(180,180);}   //3
else
        if(S1>adc0&&S2>adc1&&S3<adc2&&S4<adc3&&S5<adc4&&S6<adc5&&S7<ad
        c6&&S8<adc7){beka();go(180,180);}   //12
```

```
    else
        if(S1<adc0&&S2>adc1&&S3>adc2&&S4<adc3&&S5<adc4&&S6<adc5&&S7<ad
        c6&&S8<adc7){beka();go(180,180);}    //23
    else
        if(S1>adc0&&S2>adc1&&S3>adc2&&S4<adc3&&S5<adc4&&S6<adc5&&S7<ad
        c6&&S8<adc7){beka();go(180,180);}    //123
    else
        if(S1<adc0&&S2>adc1&&S3>adc2&&S4>adc3&&S5<adc4&&S6<adc5&&S7<ad
        c6&&S8<adc7){beka();go(180,180);}    //234
    else
        if(S1>adc0&&S2>adc1&&S3>adc2&&S4>adc3&&S5<adc4&&S6<adc5&&S7<ad
        c6&&S8<adc7){beka();go(180,180);}   //1234

    else
        if(S1<adc0&&S2<adc1&&S3<adc2&&S4<adc3&&S5<adc4&&S6<adc5&&S7<ad
        c6&&S8<adc7){mundur();go(180,180);}

    else
        if(S1>adc0&&S2>adc1&&S3>adc2&&S4>adc3&&S5>adc4&&S6>adc5&&S7>ad
        c6&&S8>adc7){henti();}
};
void main(void)
{
// Declare your local variables here

// Input/Output Ports initialization
// Port A initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTA=0x00;
DDRA=0x00;

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

```
// Port C initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTC=0b11111111;
DDRC=0x00;

// Port D initialization
// Func7=In Func6=In Func5=Out Func4=Out Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=0 State4=0 State3=T State2=T State1=T State0=T
PORTD=0b11000000;
DDRD=0b00111111;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=0xFF
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: 10.800 kHz
// Mode: Fast PWM top=0x00FF
// OC1A output: Non-Inv.
// OC1B output: Non-Inv.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0xA1;
TCCR1B=0x0D;
TCNT1H=0x00;
```

```
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer2 Stopped
// Mode: Normal top=0xFF
// 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=0x00;

// USART initialization
// USART disabled
UCSRB=0x00;

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

```
// ADC initialization
// ADC Clock frequency: 691.200 kHz
// ADC Voltage Reference: AREF pin
// ADC Auto Trigger Source: ADC Stopped
ADMUX=ADC_VREF_TYPE & 0xff;
ADCSRA=0x84;

// SPI initialization
// SPI disabled
SPCR=0x00;

// TWI initialization
// TWI disabled
TWCR=0x00;

// Alphanumeric LCD initialization
// Connections specified in the
// Project|Configure|C Compiler|Libraries|Alphanumeric LCD menu:
// RS - PORTB Bit 0
// RD - PORTB Bit 1
// EN - PORTB Bit 2
// D4 - PORTB Bit 4
// D5 - PORTB Bit 5
// D6 - PORTB Bit 6
// D7 - PORTB Bit 7
// Characters/line: 16
lcd_init(16);
define_char(char0,0);

loading();

gui();

while (1)
    {
    // Place your code here
```

```
      if (step==0){baca_adc(); proses();
                  if (stop==0){step=99;};}
      if (step==99){baca_adc(); henti();
                  if (start==0){step=0;}
      } ;
      }
}
```