SKRIPSI - ME - 141501

PERANCANGAN PROTOTYPE AUTOMATIC STACKING CRANE BERBASIS MIKROKONTROLLER ATMEGA 2560

FAJAR ANDIK CAHYONO
NRP : 4213 106 005

Dosen Pembimbing
Ir. Sardono Sarwito. M. Sc.
Indra Ranu Kusuma. S.T. M. Sc.

JURUSAN TEKNIK SISTEM PERKAPALAN
FAKULTAS TEKNOLOGI KELAUTAN
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA 2016

FINAL PROJECT – ME - 141501

DESIGN of PROTOTYPE AUTOMATIC STACKING CRANE-BASED MICROCONTROLLER ARDUINO MEGA 2560

FAJAR ANDIK CAHYONO
NRP : 4213 106 005

Conselor Lecture
Ir. Sardono Sarwito. M. Sc.
Indra Ranu Kusuma. S.T. M. Sc.

DEPARTMENT OF MARINE ENGINEERING
FACULTY OF OCEAN TECHNOLOGY
INSTITUTE TECHNOLOGY OF SEPULUH NOPEMBER
SURABAYA 2016

LEMBAR PENGESAHAN

PERANCANGAN PROTOTYPE AUTOMATIC STACKING CRANE BERBASIS MIKROKONTROLER ARDUINO MEGA 2560

SKRIPSI

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Teknik
Pada
Bidang Studi Marine Electrical and Automation System (MEAS)
Program S-1 Jurusan Teknik Sistem Perkapalan
Fakultas Teknologi Kelautan
Institut Teknologi Sepuluh Nopember

Oleh :
Fajar Andik Cahyono
Nrp. 4213106005

Disetujui oleh Dosen Pembimbing Skripsi :

1. Ir. Sardono Sarwito, M. Sc. ( )
   NIP.1960 0319 1987 01 1001
2. Indra Ranu Kusuma, ST. M.Sc. ( )
   NIP.1979 0327 2003 12 1001

SURABAYA
Januari, 2016

LEMBAR PENGESAHAN

PERANCANGAN PROTOTYPE AUTOMATIC STACKING CRANE BERBASIS MIKROKONTROLER ARDUINO MEGA 2560

SKRIPSI

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Teknik
Pada
Bidang Studi Marine Electrical and Automation System (MEAS)
Program S-1 Jurusan Teknik Sistem Perkapalan
Fakultas Teknologi Kelautan
Institut Teknologi Sepuluh Nopember

Oleh :
Fajar Andik Cahyono
NRP. 4213106005

Disetujui oleh Ketua Jurusan Teknik Sistem Perkapalan

Dr. Eng M. Badrus Zaman, S.T., M.T. ( )
NIP. 1977 0802 2008 01 1007

SURABAYA
Januari, 2016

PERANCANGAN PROTOTYPE AUTOMATIC STACKING CRANE BERBASIS MIKROKONTROLER ARDUINO MEGA 2560

NamaMahasiswa : Fajar Andik Cahyono
NRP : 4213106005
Jurusan : Teknik Sistem Perkapalan FTK-ITS
DosenPembimbing : Ir. Sardono Sarwito, M.Sc
Indra Ranu Kusuma. S.T. M. Sc

Abstrak

Pelabuhan adalah salah satu infrastruktur penunjang transportasi laut yang merupakan pintu gerbang keluar masuk barang dan penumpang. Fungsi dan peran pelabuhan sangat penting dalam mendukung sistem transportasi untuk pengembangan suatu wilayah dalam pengiriman barang yang akan di distribusikan ke daerah tujuan.

Crane adalah suatu alat yang digunakan untuk mengangkat atau memindahkan muatan berat dan banyak digunakan di pelabuhan untuk proses loading – unloading container ke truk.

Dalam penelitian ini akan dibuat prototype automatic stacking crane yang menggunakan modul mikrokontroller arduino mega 2560. Terdapat kekurangan dari perancangan ini adalah tegangan yang masuk ke motor bersumber dari arduino yaitu 5 v, sehingga perputaran motor tidak maksimal. Selain itu, respon input masih banyak terdapat prosedur yang masih kurang sesuai.

Kata kunci : **Crane*, Mikrokontroller, Prototype.***

“Halaman ini sengaja dikosongkan”

DESIGN of PROTOTYPE AUTOMATIC STACKING CRANE-BASED MICROCONTROLLER ARDUINO MEGA 2560

Student Name : **Fajar Andik Cahyono**
NRP : **4213 106 005**
Department : **Marine Engineering FTK-ITS**
Supervisor : **Ir. Sardono Sarwito, M.Sc**
Indra Ranu Kusuma. S.T. M. Sc

Abstract

The port is one of the supporting infrastructure for maritime transport which is the gate out of the incoming goods and passengers. The function and role of the port is very important in supporting the transportation system for the development of a region in the delivery of the items to be distributed to the destination.

Crane is a tool used to lift or move heavy payloads and widely used in ports for loading – unloadingprocess of container to truck. In this research will be made prototype automaticstacking crane who uses arduino mega 2560 mikrokontroller module.

Design of prototype automatic stacking crane by using the mikrokontroller form of the arduino can work well according to the design that had been designed before. There is a lackof design this is the voltage that goes to the motor from the arduino that is 5 v, so therotation of the motor is not the maximum. In addition, there are still plenty of inputresponse procedures are still less appropriate.

Keyword : Crane, Mikrokontroller, Prototype

“Halaman ini sengaja dikosongkan”

KATA PENGANTAR

Dengan mengucapkan puji dan syukur Alhamdulillah kepada Allah SWT yang telah memberikan rahmat dan hidayahNya berupa kesehatan dan ilmu pengetahuan sehingga penulis dapat menyelesaikan tugas akhir ini pada waktu yang telah ditentukan.

Tugas akhir ini dikerjakan sebagai persyaratan untuk memperoleh gelar Sarjana Teknik di Jurusan Teknik Sistem Perkapalan, Fakultas Teknologi Kelautan, Institut Teknologi Sepuluh Nopember. Di dalam penyusunan dan pengerjaan laporan ini, penyusun banyak menerima kritik, bantuan dan masukkan. Untuk itu di dalam kesempatan ini penulis mengucapkan terima kasih yang sebesar besarnya kepada pihak-pihak yang telah membantu penulis untuk menyelesaikan tugas akhir ini, penulis mengucapkan terima kasih kepada :

1. Bapak Dr. Eng. M. Badrsus Zaman, S.T. M.T selaku ketua jurusan Teknik Sistem Perkapalan
2. Bapak Ir. Sardono Sarwito M. Sc. selaku dosen pembimbing Tugas Akhir I, yang telah memberikan bimbingannya dalam penyelesaian Tugas Akhir ini.
3. Bapak Indra Ranu Kusuma. ST. M. Sc selaku dosen pembimbing Tugas Akhir II, yang telah memberikan bimbingannya dalam penyelesaian Tugas Akhir ini.

4. Bapak Irfan Syarif Arief. S.T. M.T selaku dosen wali, yang selama 4 semester ini medukung dan memberikan ilmu dan nasehat yang bermanfaat.
5. Bapak-bapak Dosen beserta Staf Jurusan Teknik Sistem Perkapalan yang tidak dapat disebut satu-persatu, yang telah banyak memberikan ilmu pengetahuan, bimbingan dan bantuannya selama penulis kuliah.
6. Orang tua tercinta yang selalu memberikan dukungan dan doa restunya sehingga penulis mendapat kelancaran dalam pengerjaan Tugas Akhir ini.
7. Rizkia Amalinda yang selalu memberi dukungan dan semangat baik secara langsung maupun tidak langsung
8. Boy Hendra Waramory, A'ang Kunaifi, Firman Rachmad W selaku sahabat penulis yang selalu memberikan semangat dan menemani dengan sabarnya saat pengerjaan Tugas Akhir ini.
9. Teman-teman lintas jalur genap 2013 yang selalu memberikan semangat dan ilmu selama 4 semester perkuliahan sampai dengan pengerjaan Tugas Akhir ini.
10. Teman-teman keluarga di Laboratorium "Marine Electrical and Automation System (MEAS) yang selalu memberikan semangat dan ilmu selama pengerjaan tugas akhir ini.
11. Semua pihak secara langsung maupun tidak langsung membantu penulis dalam menyusun Tugas Akhir ini.

Semoga semua kebaikan dan bantuan yang diberikan secara ikhlas dan tulus dari semua pihak kepada penulis dalam menyusun Tugas Akhir ini dibalas oleh Allah SWT. Penulis berharap semoga Tugas Akhir ini dapat memberi manfaat bagi semua pihak. Sekian dan Terima Kasih.

Surabaya, Januari 2016

Hormat Penulis

Halaman ini sengaja dikosongkan

DAFTAR ISI

“Halaman ini sengaja dikosongkan”

DAFTAR GAMBAR

Halaman ini sengaja dikosongkan

DAFTAR TABEL

Halaman ini sengaja dikosongkan

BAB I

PENDAHULUAN

1.1. Latar Belakang

Pelabuhan adalah salah satu infrastruktur penunjang transportasi laut yang merupakan pintu gerbang keluar masuk barang dan penumpang. Fungsi dan peran pelabuhan sangat penting dalam mendukung sistem transportasi untuk pengembangan suatu wilayah dalam pengiriman barang yang akan di distribusikan ke daerah tujuan.

Crane adalah suatu alat yang digunakan untuk mengangkat atau memindahkan muatan berat dan banyak digunakan di pelabuhan untuk proses loading – unloading container ke truk. crane mempunyai aturan bagaimana prosedur mengangkat suatu container, ada sebuah kabel dengan payload menggantung dan akan bergerak mengangkat maupun menurunkan beban ke lokasi yang dinginkan.

Adapun permasalahan ini perlu untuk di selesaikan bahwa automatic stacking crane mempunyai sistem kerja yang rumit dalam menjalankan fungsinya yaitu suatu alat yang digunakan untuk mengangkat atau memindahkan muatan berat dan banyak digunakan di pelabuhan untuk proses loading – unloading container, tentunya mempunyai sistem yang rumit, dari sistem yang inilah penggunaan sistem kontrol sangat perlu dilakukan. Dalam hal ini dapat di simulasikan menggunakan prototype

1.2. Perumusan Masalah

Berdasarkan prinsip mikrokontroller yang dapat mengendalikan mesin sesuai dengan yang diinginkan, sehingga mikrokontroller dapat diaplikasikan untuk pembuatan simulasi prototype crane di pelabuhan. Maka diambil permasalahan sebagai berikut:

1. Bagaimana perancangan prototype automatic stacking crane menggunakan mikrokontroller ?
2. Bagaimana perancangan program pada pembuatan simulasi ?

1.3. Batasan Masalah

Untuk lebih menfokuskan permasalahan dalam perancangan prototype tugas akhir ini, maka akan dibatasi permasalahannya sebagai berikut :

1. Jenis Mikrokontroller yang dibahas menggunakan Arduino Mega 2560.
2. Tidak mengubah sistem kerja dari crane.

1.4. Tujuan Penulisan

1. Tujuan dari pembahasan tugas akhir merupakan sebagai pembelajaran tentang Mikrokontroller agar memahami tentang pemograman dan control. Memberikan gambaran cara kerja mikrokontroller dalam pengaplikasian pada pembuatan simulasi prototype crane pada proses bongkar muat container di pelabuhan.

1.5. Manfaat Penulisan

Adapun manfaat yang dapat diperoleh dari penulisan tugas akhir adalah:

1. Manfaat yang diperoleh dari penulisan tugas akhir ini adalah mengetahui sistem kerja dari program kerja mikrokontroler pada pembuatan prototype simulasi crane dan memberikan pengetahuan tentang mikrokontroller baik dari cara kerja maupun peralatan.

Halaman ini sengaja dikosongkan

BAB II

TINJAUAN PUSTAKA

2.1 Pengertian Mikrokontroller

Mikrokontroller adalah sebuah sistem komputer fungsional dalam sebuah chip. Di dalamnya terkandung sebuah inti prosesor memori (sejumlah kecil RAM, memori program, atau keduanya), dan perlengkapan input output.

Dengan kata lain, mikrokontroller adalah suatu alat elektronika digital yang mempunyai masukan dan keluaran serta kendali dengan program yang bisa ditulis dan dihapus dengan cara khusus, cara kerja mikrokontroller sebenarnya membaca dan menulis data. Mikrokontroller merupakan komputer didalam chip yang digunakan untuk mengontrol peralatan elektronik, yang menekankan efisiensi dan efektifitas biaya. Secara harfiahnya bisa disebut "pengendali kecil" dimana sebuah sistem elektronik yang sebelumnya banyak memerlukan komponen-komponen pendukung seperti IC TTL dan CMOS dapat direduksi/diperkecil dan akhirnya terpusat serta dikendalikan oleh mikrokontroller ini. Dengan menggunakan mikrokontroller sistem kerja alat yang akan dibuat bisa dirancang melalui pemrograman dengan berbagai macam mikrokontroller yang tersedia.

2.2. Platform Arduino

Arduino merupakan sebuah platform yang dikembangkan dengan tujuan untuk memudahkan penggunaan mikrokontroller dan rangkaian elektronika. . Dengan Arduino, diharapkan hal teknis tidak lagi menjadi penghalang bagi para individu kreatif untuk mengembangkan kreasinya.

Meskipun Arduino ditujukan untuk memudahkan penggunaan mikrokontroler dengan bekal kemampuan teknis yang minimal, hal tersebut bukan berarti Arduino tidak cocok bagi para praktisi elektronika, khususnya pada bidang mikrokontroler. Dalam kasus tertentu Arduino akan sangat

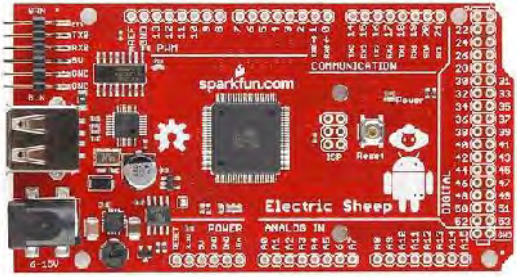
membantu, karena dengan penggunaan mikrokontroler yang semakin mudah, maka konsentrasi pengerjaan dapat dialihkan pada perancangan sistem mikrokontroler yang lebih kompleks.
Beberapa fitur yang ditawarkan oleh Arduino adalah:

- Multiplatform environment, Arduino IDE dapat berjalan pada berbagai macam sistem operasi (Windows, Linux, Macintosh).
- Software Arduino IDE yang mudah digunakan.
- Open source baik hardware maupun software sehingga pengguna Arduino dapat membuat ataupun memodifikasi platform Arduino secara mandiri.
- Investasi untuk platform Arduino relatif murah apabila dibandingkan dengan kehandalan dan dukungan yang didapatkan.
- Sumber informasi yang luas, karena banyak forum dan artikel yang membahas tentang penggunaan Arduino.
- Arduino dikembangkan dalam lingkungan pendidikan sehingga Arduino cocok digunakan sebagai langkah awal bagi pemula yang ingin memasuki dunia elektronika.

2.3. Hardware Arduino

Hardware Arduino merupakan sebuah modul mikrokontroler yang telah dilengkapi dengan program bootloader untuk membantu proses pengisian program. Sampai saat penulisan materi ini, kebanyakan modul Arduino menggunakan mikrokontroler Atmel AVR 8-bit RISC, kecuali Arduino Due yang sudah menggunakan mikrokontroler 32-bit keluarga ARM Cortex-M3.

Gambar 2.1 Arduino Mega ADK – *official board*



Gambar 2.2 *Electric Sheep – compatible board*

Karena Arduino merupakan *project* yang bersifat *open source*, maka kalangan umum dibebaskan untuk membuat *hardware* Arduino dalam versinya masing-masing. Beberapa produsen telah membuat dan merilis produk Arduino *compatible* yang tidak kalah menarik. Karena tidak ada standar pasti untuk hal kompatibilitas, maka tingkat kompatibilitas Arduino untuk masing-masing produk pun dapat berbeda mulai dari dukungan penuh (*clone*), dukungan penuh dengan penambahan/pengurangan fitur, hingga sebatas peletakan pinout yang sama.

Berikut adalah deskripsi umum dari sebagian pin pada *board* Arduino:

- *Reset (active low)* – pin untuk mengulang eksekusi program/*sketch* Arduino dari awal.
- 3V3 – pin untuk jalur tegangan 3.3 VDC (output dan input).
- 5V – pin untuk jalur tegangan 5 VDC (output dan input).
- GND – pin yang terhubung dengan jalur ground.
- Vin – pin yang terhubung dengan jalur catu daya.
- Analog In (A0, A1, A2, …) – pin input untuk tegangan analog. Pin-pin ini juga dapat difungsikan sebagai pin I/O digital.
- SDA/SCL – pin data dan c*lock* yang digunakan untuk komunikasi serial TWI.
- RX/TX – pin *receiver* dan *transmitter* untuk komunikasi serial UART.
- Digital (0, 1, 2, …) – pin I/O dengan sinyal high/low.
- PWM – pin digital yang memiliki fitur output PWM (sebagai output analog).
- AREF – pin input tegangan referensi eksternal untuk acuan pembacaan tegangan analog.

Deskripsi fungsi pin di atas dapat berbeda untuk beberapa *board* Arduino. Informasi lebih pasti dapat ditemukan pada masing-masing panduan *board* Arduino yang digunakan.

2.4 Arduino IDE

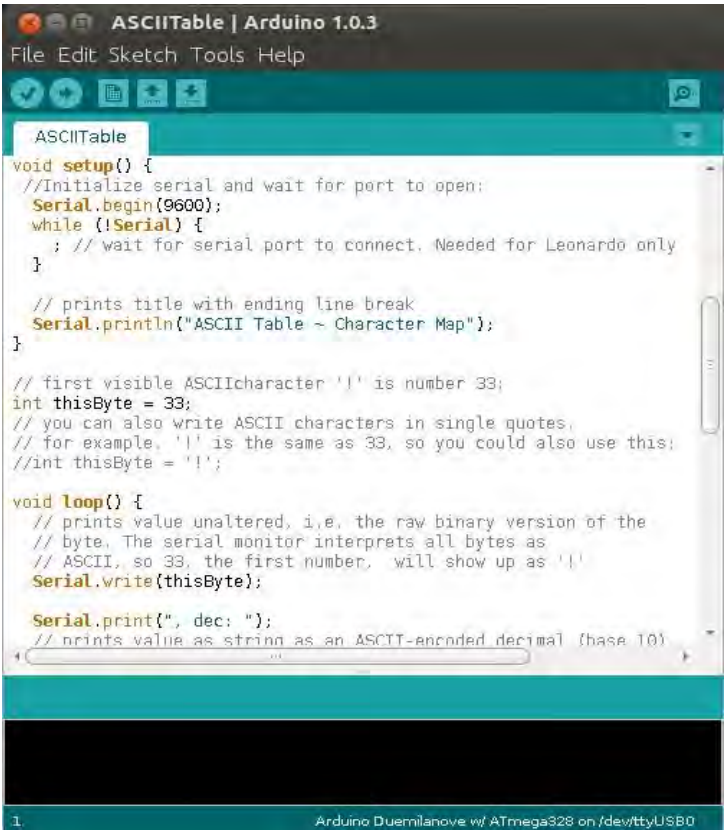
Pemrograman Arduino dapat dilakukan dengan menggunakan bantuan *software* Arduino IDE (*Integrated Development Kit)*. Arduino IDE merupakan software terpadu yang sudah menyediakan fitur-fitur yang diperlukan pengguna untuk menciptakan sebuah program/sketch Arduino.

Arduino IDE memiliki 3 bagian utama sebagai berikut:

- *Editor* – yaitu sebuah window dimana pengguna dapat menuliskan syntax sketch Arduino.

- *Compiler* – fitur yang digunakan untuk mengubah syntax sketch menjadi kode mesin yang dipahami oleh mikrokontroler.
- *Uploader* – fitur yang digunakan untuk mamasukkan kode mesin hasil proses dari compiler ke dalam memori pada mikrokontroller Arduino.

Adapun tampilan dari Arduino IDE adalah seperti pada Gambar :



Gambar 2.3 Tampilan Arduino IDE (*Integrated Development Kit)*

Beberapa bagian software yang akan sering digunakan oleh pengguna dapat ditemukan pada toolbar Arduino IDE. Toolbar Arduino IDE memiliki 6 buah tombol dengan deskripsi seperti pada Tabel 2.1.

Tabel 2.1 Deskripsi Toolbar Pada Arduino IDE

Tombol	Deskripsi
 Verify	untuk melakukan proses cek apakah terdapat kesalahan pada *sketch* atau tidak
 Upload	melakukan proses *compile* yang dilanjutkan dengan *upload sketch* ke *board* Arduino
 New	membuat *sketch* baru
 Open	untuk membuka kode pada bagian *examples* atau *sketch* yang telah dibuat sebelumnya
 Save	menyimpan *sketch* yang terbuka saat ini
 Serial Monitor	membuka Serial Monitor pada Arduino IDE

Fungsi lain dari software Arduino IDE dapat ditemukan pada 5 buah menu yang terletak di atas toolbar, yaitu File, Edit, Sketch, Tools, dan Help. Deskripsi bagian penting untuk masing-masing menu dapat ditemukan pada Tabel di bawah.

2.2 Tabel Deskripsi beberapa bagian menu pada Arduino IDE

Menu	Sub-menu	Deskripsi

File	Sketchbook	membuka *sketch* yang telah dibuat dan disimpan pada *folder* sketchbook
	Examples	membuka contoh *sketch* yang telah disediakan sebagai referensi pemrograman
	Save	menyimpan *sketch*
	Save As...	menyimpan *sketch* dengan nama atau lokasi yang berbeda
	Upload	melakukan proses *compile* yang dilanjutkan dengan *upload*
	Upload using programmer	*melakukan proses compile* yang dilanjutkan dengan *upload sketch* ke *board* Arduino
	Preferences	untuk melakukan konfigurasi Arduino IDE seperti lokasi *default folder* sketchbook,

Lanjutan..

Menu	Sub-menu	Deskripsi
Edit	Copy for Forum	menyalin *sketch* dalam *format* yang sesuai untuk *posting* di
	Copy as HTML	menyalin *sketch* dalam *format* HTML yang sesuai digunakan
	Verify/Compile	untuk melakukan proses cek apakah terdapat kesalahan pada *sketch* atau tidak

Sketch	Show Sketch Folder	membuka *folder* yang ditempati oleh *sketch*
	Add File...	menambahkan *source file* pada *sketch* (akan dibuka pada jendela *tab* baru)
	Import Library...	menambahkan *library* pada *sketch*
Tools	Serial Monitor	membuka Serial Monitor pada Arduino IDE
	Board	memilih jenis *board* Arduino yang digunakan
	Serial Port	memilih nomor serial *port* yang digunakan
	Programmer	memilih jenis *device programmer* yang digunakan
	Burn Bootloader	untuk melakukan proses pengisian *bootloader* pada *board* Arduino menggunakan *device programmer*
Help	-	berisi berbagai sumber referensi mengenai penggunaan, pemrograman, hingga *troubleshooting* Arduino

2.5 Struktur Dasar Pemrograman Arduino

Bahasa pemrograman yang digunakan untuk pembuatan sketch Arduino adalah C/C++. Karena Arduino mendukung penulisan sketch dalam bahasa C++, maka Arduino juga

mendukung konsep pemrograman berorientasi obyek dimana pengguna dimungkinkan untuk mendeklarasikan class pada program.

Sebuah sketch Arduino minimal harus menyertakan dua fungsi utama yaitu void setup() dan void loop().

```
void setup(){
// kode program
}

void loop(){
//kode program
}
```

void setup() merupakan sebuah fungsi yang hanya akan dijalankan sekali saja saat board Arduino dinyalakan. Seperti namanya, fungsi ini cocok digunakan sebagai bagian inisialisasi variable, inisialisasi pin I/O, dll. Meskipun suatu sketch tidak membutuhkan inisialisasi, fungsi ini harus tetap disertakan agar proses verify/compile sketch berhasil (tidak mengeluarkan pesan error).

Setelah fungsi void setup() selesai dieksekusi, Arduino akan menjalankan fungsi void loop(). Syntax yang ada dalam fungsi ini akan dijalankan berulang- ulang tanpa ada kondisi berhenti. Fungsi ini cocok digunakan sebagai bagian utama program Arduino.

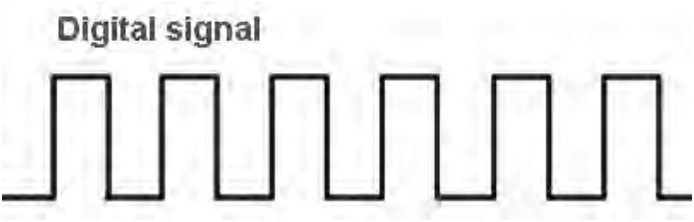
Sebagai catatan, selain dua fungsi sebelumnya pengguna dimungkinkan untuk membuat fungsi bentukan sendiri. Hal ini akan sangat membantu dalam membuat suatu sketch yang terstruktur dengan baik.

2.6. I/O (Input/Output) Digital

2.6.1 Sinyal Digital

Sinyal digital pada rangkaian elektronika merupakan sebuah sinyal yang hanya mengenal dua kondisi saja, yaitu high atau low pada suatu waktu. Pada sinyal digital tidak dikenal

adanya suatu keadaan di tengah kedua kondisi yang telah disebutkan sebelumnya. Beberapa contoh dari perangkat yang dapat bekerja menggunakan sinyal digital adalah switch, LED, relay, dll.



Gambar 2.4 Bentuk Sinyal Digital

Board Arduino dilengkapi dengan sejumlah pin I/O digital. Jumlah ini bervariasi mengikuti jenis dari board Arduino yang digunakan. Pada Arduino UNO, jumlah standar pin I/O digital adalah 14 buah (pin 0 – pin 13). Jumlah tersebut mengikutsertakan 6 buah pin yang memiliki fitur output PWM (*Pulse Width Modulation*) untuk keperluan output sinyal analog.

2.6.2 Pemrograman I/O Digital Arduino

Arduino IDE telah menyediakan fungsi siap pakai yang dapat dengan mudah digunakan untuk melakukan pengendalian pin l/O digital pada board Arduino. Fungsi-fungsi tersebut adalah:

- pinMode(pin, mode) - Fungsi ini digunakan untuk melakukan konfigurasi pada salah satu pin I/O board Arduino, yaitu sebagai input, input dengan pull-up, ataupun sebagai output. Sebagai catatan, secara default kondisi awal pin I/O board Arduino ketika baru dinyalakan adalah sebagai input.
- digitalWrite(pin, value) - Fungsi ini digunakan untuk memberikan kondisi high atau low pada salah satu pin I/O board Arduino.

- digitalRead(pin) - Fungsi ini digunakan untuk membaca kondisi logika pada salah satu pin I/O board Arduino.

Untuk memudahkan pengguna dalam hal pemrograman, Arduino IDE telah menyediakan beberapa konstanta yang dapat digunakan pada ketiga fungsi di atas. Konstanta tersebut adalah seperti pada Tabel 2.3

Tabel 2.3 Konstanta Untuk Pemrograman I/O Digital

Konstanta	Nilai	Digunakan pada fungsi
INPUT	0	pinMode(pin, mode)
OUTPUT	1	pinMode(pin, mode)
INPUT_PULLUP	2	pinMode(pin, mode)
HIGH	1	digitalWrite(pin, value)
LOW	0	digitalWrite(pin, value)

Contoh pengaksesan *output* digital pada Arduino adalah sebagai berikut:

```
//Deklarasi variable untuk menyimpan nilai/nomor pin LED yang
//akan digunakan.
int outputLED = 10;

//Deklarasi variable untuk menyimpan nilai delay.
int delayTime = 1000;

//Fungsi void setup() hanya akan dijalankan sekali saja pada saat
//awal board Arduino dinyalakan. Fungsi ini cocok digunakan
//sebagai bagian inisialisasi variable, inisialisasi hardware,
//dll.
void setup()
{
  //Fungsi pinMode digunakan untuk melakukan konfigurasi pin I/O
  //Arduino apakah sebagai INPUT, OUTPUT, atau INPUT_PULLUP.
  pinMode(outputLED, OUTPUT);
}
```

```
//Fungsi void loop() akan dijalankan berulang ulang secara
//sekuensial dari atas ke bawah. Fungsi ini umumnya digunakan
//untuk peletakan syntax program utama.
void loop()
{
  //digitalWrite digunakan untuk mengatur logika dari pin I/O
  //Arduino. Apabila konfigurasi pin adalah OUTPUT, maka fungsi
  //ini digunakan untuk memberikan sinyal output HIGH/LOW.
  //Apabila konfigurasi pin adalah INPUT, maka fungsi ini
  //digunakan untuk mengaktifkan/me-nonaktifkan fitur PULLUP.
  digitalWrite(outputLED, HIGH);
  //delay merupakan fungsi yang digunakan untuk memberikan waktu
  //tunda untuk eksekusi syntax selanjutnya. Satuan waktu delay
  //adalah milli detik. Apabila Arduino memasuki eksekusi delay,
  //maka Arduino tidak akan melakukan apa-apa selama waktu yang
  //ditentukan.
  delay(delayTime);
  digitalWrite(outputLED, LOW);
  delay(delayTime);
}
```

Contoh pengaksesan *input* digital pada Arduino adalah sebagai berikut:

```
//Deklarasi variable untuk menyimpan nilai/nomor pin LED yang
//akan digunakan.
int inputTC = A1;
int outputLED = 10;

//Deklarasi variable untuk menyimpan hasil pemanggilan fungsi
//digitalRead.
int stateTC;

//Fungsi void setup() hanya akan dijalankan sekali saja pada saat
//awal board Arduino dinyalakan. Fungsi ini cocok digunakan
//sebagai bagian inisialisasi variable, inisialisasi hardware,
//dll.
void setup()
{
  //Fungsi pinMode digunakan untuk melakukan konfigurasi pin I/O
  //Arduino apakah sebagai INPUT, OUTPUT, atau INPUT_PULLUP.
  pinMode(inputTC, INPUT_PULLUP);
  pinMode(outputLED, OUTPUT);
}

//Fungsi void loop() akan dijalankan secara berulang ulang secara
//sekuensial dari atas ke bawah. Fungsi ini umumnya digunakan
//untuk peletakan syntax program utama.
void loop()
{
```

```
  //digitalRead digunakan untuk melakukan pembacaan logika dari
  //pin I/O Arduino. Fungsi tersebut akan memberikan nilai
  //kembalian berupa angka 0(LOW)/1(HIGH) saat pemanggilannya.
  //Pada syntax di bawah nilai tersebut ditampung pada variabel
  //stateTC.
  stateTC = digitalRead(inputTC);
  //Nilai dari variable stateTC digunakan pada proses pemanggilan
  //fungsi digitalWrite.
  digitalWrite(outputLED, stateTC);

  //Dua baris syntax di atas dapat digantikan dengan syntax :
//  digitalWrite(outputLED, digitalRead(inputTC));
}
```

2.6.3 Dasar Pemrograman Digital I/O Arduino

2.6.3.1 Variable & Konstanta

Variable merupakan suatu lokasi pada memory yang dapat digunakan untuk menyimpan suatu nilai. Langkah pertama sebelum menggunakan variable adalah mendeklarasikannya. Pada deklarasi variable kita akan menentukan tipe data variable, nama variable (identifier), serta nilai awal (bersifat opsional).

```
int variable_satu;
//int merupakan tipe data variable.
//variable_satu merupakan nama/identifier dari variable.

byte variable_dua = 10;
//byte merupakan tipe data variable.
//variable_dua merupakan nama/identifier dari variable.
//10 merupakan nilai awal dari variable.
```

Tipe data suatu variable berkaitan dengan rentang nilai yang dapat ditampung oleh variable tersebut. Adapun beberapa tipe data yang sering digunakan pada pemrograman Arduino adalah seperti pada Tabel 2.3

Tabel 2.4 Beberapa Type Data *Variable* Pada Arduino IDE

Tipe data	Memori	Rentang nilai
char	1 *Byte*	-128 – 127

Byte	1 *Byte*	0 – 255
int	2 *Byte*	-32,768 – 32,767
unsigned int	2 *Byte*	0 – 65.535
long	4 *Byte*	-2,147,483,648 – 2,147,483,647
unsigned long	4 *Byte*	0 - 4,294,967,295
float	4 *Byte*	-3.4028235E+38 - 3.4028235E+38
double	4 *Byte*	-3.4028235E+38 - 3.4028235E+38

Hal lain yang perlu diperhatikan pada saat mendeklarasikan variable adalah scope. Scope yang dimaksud di sini adalah ruang lingkup variable tersebut agar dapat dikenali saat pemanggilannya. Scope dari variable dapat dibagi menjadi dua, yaitu local dan global.

```
char globalVariable;

void fungsi_1()
{
  char localVariable_satu;

  localVariable_satu = globalVariable + 10;
  //Pemanggilan localVariable_satu di sini diijinkan karena kode
  //program pada bagian ini masih mengenalinya.
  //Pemanggilan globalVariable di sini dijinkan karena variable
  //tersebut bersifat global, sehingga semua bagian kode program
  //mengenalinya.

  //Pemanggilan localVariable_dua di sini tidak diijinkan karena
  //variable tersebut bersifat local pada fungsi_2 saja.
}

void fungsi_2()
{
  char localVariable_dua;
```

```
  localVariable_dua = globalVariable - 10;
  //Pemanggilan localVariable_dua di sini diijinkan karena kode
  //program pada bagian ini masih mengenalinya.
  //Pemanggilan globalVariable di sini dijinkan karena variable
  //tersebut bersifat global, sehingga semua bagian kode program
  //mengenalinya.

  //Pemanggilan localVariable_satu di sini tidak diijinkan karena
  //variable tersebut bersifat local pada fungsi 1 saja.
}
```

2.6.3.2 Fungsi

Apabila pada suatu sketch terdapat list kode program yang cukup panjang dan sering digunakan, maka kita dapat membungkusnya ke dalam sebuah fungsi agar memudahkan proses pemrograman serta membuat sketch menjadi lebih terstruktur. Seperti layaknya sebuah variable, fungsi juga memiliki tipe data serta identifier. Selain kedua hal tersebut, fungsi memiliki parameter serta nilai kembalian (return value).

```
int perkalian(char angka1, byte angka2)
{
  int hasil = angka1 * angka2;
  return hasil;
}
//int merupakan tipe data dari fungsi.
//perkalian merupakan indentifier fungsi.
//angka1 dan angka2 merupakan parameter fungsi (opsional).
//return adalah perintah untuk memberikan suatu nilai sebagai
//hasil saat fungsi tersebut dipanggil. Perintah return akan
//mengakhiri eksekusi suatu fungsi.

void ledBlink()
{
  digitalWrite(10, HIGH); delay(500);
  digitalWrite(10, LOW); delay(500);
}
//Fungsi dapat tidak memiliki tipe data. Untuk hal tersebut kita
//dapat menggunakan statement void. Fungsi void tidak dapat
//melakukan return.

void setup()
{
  pinMode(10, OUTPUT);
}

void loop()
{
  int a = 5; int b = 6;
```

```
  //Contoh pemanggilan fungsi.

  int hasil = perkalian(a, b);
  //variable hasil mendapatkan nilai 30 dari return fungsi
  //perkalian.

  ledBlink();
}
```

2.6.3.3 *Conditional Structure*

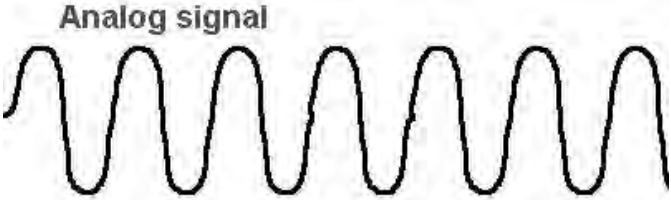
Conditional structure merupakan suatu cara dimana kita dapat mengendalikan eksekusi sketch/program yang umumnya sekuensial sehingga dapat melakukan eksekusi bagian tertentu saja berdasarkan suatu kondisi. Contoh syntax untuk conditional structure adalah sebagai berikut:

```
//Struktur if...else.
//Jika nilai variable_1 sama dengan 2.
if(variable_1 == 2)
{
  ... //Syntax yang akan dieksekusi.
}
//Jika nilai variable_1 kurang dari 7.
else if(variable_1 < 7)
{
  ... //Syntax yang akan dieksekusi.
}
//Jika nilai variable_1 lebih dari atau sama dengan 2 DAN
//nilai variable_1 kurang dari 10.
else if(variable_1 >= 2 && variable_1 < 10)
{
  ... //Syntax yang akan dieksekusi.
}
//Jika logika pin 10 HIGH ATAU logika pin 11 HIGH.
else if(digitalRead(10) || digitalRead(11))
{
  ... //Syntax yang akan dieksekusi.
}
//Jika logika pin 10 LOW.
else if(!digitalRead(10))
{
  ... //Syntax yang akan dieksekusi.
}
//Jika kondisi-kondisi di atas tidak terpenuhi.
else
{
  ... //Syntax yang akan dieksekusi.
}
```

```
//Struktur switch...case.
switch(variable_1)
{
  case 1: //Jika variable_1 bernilai 1.
    ... //Syntax yang akan dieksekusi.
    break;
  case 2: //Jika variable_1 bernilai 2.
    ... //Syntax yang akan dieksekusi.
    break;
  case 3: //Jika variable_1 bernilai 3.
    ... //Syntax yang akan dieksekusi.
    break;
  default: //Jika nilai-nilai di atas tidak terpenuhi.
    ... //Syntax yang akan dieksekusi.
    break;
}
```

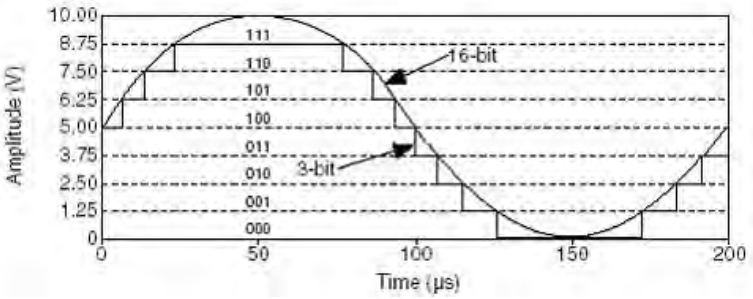
2.6.4 Sinyal Analog

Sinyal analog merupakan suatu sinyal kontinu yang memiliki nilai bervariasi untuk amplitudo, frekuensi, serta fasanya pada setiap waktu. Beberapa contoh dari perangkat yang bekerja menggunakan sinyal analog adalah mic condenser, potensiometer, force-sensing resistor, dll.

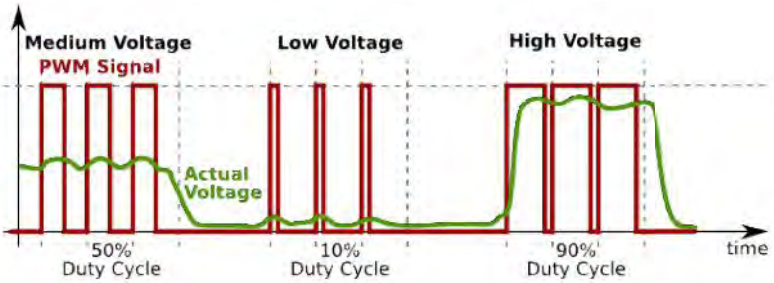


Gambar 2.5 Bentuk Sinyal Digital

Board Arduino menyediakan fitur ADC (Analog-to-Digital Converter) sebagai input analog dan pin digital PWM (Pulse-Width Modulation) sebagai output analog. Grafik konversi sinyal analog oleh ADC dan sinyal analog yang dihasilkan dengan metode PWM terdapat pada Gambar 2.6 dan Gambar 2.7



Gambar 2.6 Konversi sinyal analog oleh ADC



Gambar 2.7 Sinyal analog yang dihasilkan dengan metode PWM

2.6.5 Pemrograman I/O Analog Arduino

Arduino IDE telah menyediakan fungsi siap pakai yang dapat dengan mudah digunakan untuk melakukan pengendalian pin I/O analog pada board Arduino. Fungsi-fungsi tersebut adalah:

- analogReference (type)

Fungsi ini digunakan untuk melakukan konfigurasi nilai tegangan referensi yang digunakan untuk pembacaan sinyal analog. Pengisian parameter type dapat dilakukan menggunakan konstanta yang terdapat pada Tabel 2.4 dibawah.

Tabel 2.5 Konstanta untuk pengaturan tegangan referensi analog

Konstanta	Nilai tegangan referensi	Note
DEFAULT	3.3 VDC / 5 VDC	sesuai dengan tegangan kerja *board* Arduino
INTERNAL	1.1 VDC / 2.56 VDC	1.1 VDC untuk Atmega 168/328, 2.56 VDC untuk ATmega8, tidak tersedia pada Arduino Mega
INTERNAL1V1	1.1 VDC	hanya untuk Arduino Mega
INTERNAL2V56	2.56 VDC	hanya untuk Arduino Mega
EXTERNAL	0 VDC – 5 VDC	sesuai dengan nominal tegangan pada *pin* AREF

Sebagai catatan, nilai pada pin AREF tidak boleh melebihi batasan 0 VDC – 5 VDC (informasi lebih jelas terdapat pada panduan board Arduino yang digunakan). Apabila pin AREF dihubungkan dengan tegangan eksternal, maka nilai tegangan referensi harus dikonfigurasikan ke EXTERNAL sebelum menggunakan fungsi analogRead(pin). Apabila kedua hal tersebut terlewatkan, board Arduino dapat mengalami kerusakan.

- analogRead(pin)

Fungsi ini digunakan untuk melakukan pembacaan nilai tegangan analog pada salah satu pin input analog board Arduino. Fungsi ini akan memetakan nilai tegangan 0 – "nilai tegangan referensi" kedalam nilai integer 0 – 1023 (resolusi 10-bit). Apabila nilai tegangan analog melebihi nilai konfigurasi tegangan referensi, maka nilai tersebut akan dipetakan ke dalam angka 1023. Perlu diperhatikan bahwa nilai tegangan analog yang melebihi spesifikasi board Arduino dapat mengakibatkan

kerusakan. Sebagai contoh, apabila tegangan kerja Arduino UNO adalah 5 VDC dan nilai tegangan referensi dikonfigurasikan ke INTERNAL (2.56 VDC), maka tegangan analog dengan nilai 3.3 VDC, 4 VDC, dan 5 VDC akan dipetakan ke dalam angka yang sama, yaitu 1023.

- analogWrite(pin, value)

Fungsi ini digunakan untuk memberikan pulsa PWM pada pin digital board Arduino yang dilengkapi fitur tersebut. Duty cycle untuk pulsa PWM diatur pada parameter value dengan nilai 0 (0%) – 255 (100%). Nilai frekuensi untuk sinyal PWM ≈ 490 Hz. Contoh pengaksesan input analog pada Arduino adalah sebagai berikut:

```
int analogInput = A0;
int pwmLED = 10;
int analogData;

void setup()
{

}

void loop()
{
  //analogRead merupakan fungsi yang dapat digunakan untuk
  //melakukan pembacaan nilai tegangan analog pada pin Ax. Nilai
  //pembacaan tersebut memiliki rentang 0 - 1023.
  analogData = analogRead(analogInput);
  //Fungsi analogWrite akan digunakan sebagai output hasil
  //pembacaan fungsi analogRead. Adapun nilai untuk pemanggilan
  //fungsi analogWrite memiliki rentang 0 - 255 (berbeda dengan
  //rentang analogRead), maka dari itu akan digunakan bantuan
  //fungsi map. Fungsi map di bawah akan memetakan nilai 0 - 1023
  //kedalam angka 0 - 255 sehingga sesuai dengan rentang nilai
  //fungsi analogWrite.
  analogData = map(analogData, 0, 1023, 0, 255);
  analogWrite(pwmLED, analogData);

  //Syntax di atas dapat digantikan dengan syntax berikut.
//  analogWrite(pwmLED, map(analogRead(analogInput), 0, 1023, 0,
//  255));
}
```

Contoh pengaksesan output analog pada Arduino adalah sebagai berikut:

```
int pwmLED = 10;
int delayTime = 1000;

void setup()
{

}

void loop()
{
  //analogWrite merupakan fungsi untuk menuliskan sinyal PWM pada
  //pin Arduino yang memiliki fitur tersebut. Pemanggilan fungsi
  //ini mirip dengan pemanggilan fungsi digitalWrite, hanya saja
  //nomor pin harus disesuaikan dengan pin yang mendukung serta
  //nilai HIGH dan LOW diganti dengan nilai 0 - 255.
  analogWrite(pwmLED, 0);
  delay(delayTime);
  analogWrite(pwmLED, 127);
  delay(delayTime);
  analogWrite(pwmLED, 255);
  delay(delayTime);
}
```

2.6.6 Dasar Pemrograman I/O Analog Arduino

2.6.6.1 Interation Structure

Apabila kita ingin membuat syntax untuk menghidupkan kemudian mematikan sebuah LED sebanyak 5 kali, kita dapat menuliskannya sebagai berikut:

```
void setup()
{
  pinMode(10, OUTPUT);
}

void loop()
{
  digitalWrite(10, HIGH); delay(1000);
  digitalWrite(10, LOW); delay(1000);
  digitalWrite(10, HIGH); delay(1000);
  digitalWrite(10, LOW); delay(1000);
  digitalWrite(10, HIGH); delay(1000);
  digitalWrite(10, LOW); delay(1000);
  digitalWrite(10, HIGH); delay(1000);
  digitalWrite(10, LOW); delay(1000);
  digitalWrite(10, HIGH); delay(1000);
  digitalWrite(10, LOW); delay(1000);

  ...
}
```

Penulisan sketch dengan cara tersebut akan menjadi lebih tidak masuk akal apabila kita harus menghidupkan kemudian mematikan sebuah LED sebanyak 63 kali. Oleh karena hal tersebut, kita dapat menggunakan salah satu fitur dasar pemrograman pada Arduino, yaitu iteration yang disebut juga sebagai perulangan. Beberapa contoh syntax perulangan untuk mematikan kemudian menghidupkan LED sebanyak 63 kali adalah sebagai berikut:

```
//Struktur for.
for(int i = 0; i < 63; i++) //(inisialisasi; kondisi; increment)
{
  digitalWrite(10, HIGH);
  delay(1000);
  digitalWrite(10, LOW);
  delay(1000);
}

//int i --> merupakan bagian inisialisasi. Bagian ini berfungsi
//sebagai deklarasi yang akan dieksekusi sekali saja saat awal
//perulangan.
//i < 63 --> merupakan bagian kondisi yang akan menentukan
//berlangsung atau tidaknya eksekusi syntax dalam perulangan.
//Pada syntax di atas, baris perintah dalam perulangan akan terus
//dieksekusi sampai nilai variable i lebih dari atau sama dengan
//63.

//i++ --> merupakan bagian increment yang akan mengubah nilai
//variable pada bagian inisialisasi. Bagian ini akan dieksekusi
//sekali setiap kali syntax yang ada dalam perulangan selesai
//dieksekusi.
```

```
//Struktur while.
int i = 0; //Bagian inisialisasi.
while(i < 50) //Bagian kondisi.
{
  digitalWrite(10, HIGH);
  delay(1000);
  digitalWrite(10, LOW);
  delay(1000);

  i = i + 1; //Bagian increment.
}

//Perulangan menggunakan while tidak memiliki bagian inisialisasi
//serta bagian increment. Perulangan dengan while hanya
//menyediakan bagian kondisi saja. Maka dari itu bagian
//inisialisasi dan bagian increment dapat ditambahkan secara
//manual.
```

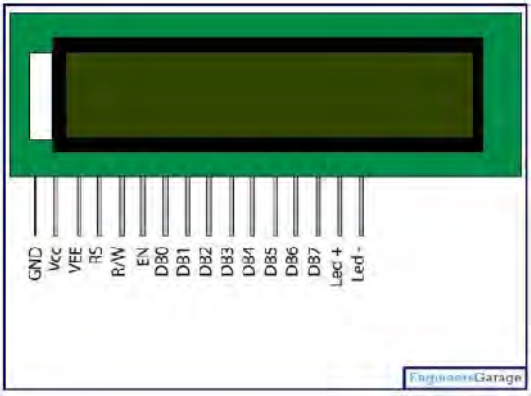
```
//Bagian kondisi pada perulangan while akan diperiksa terlebih
//dahulu sebelum mengeksekusi syntax yang ada di dalam while.
//Apabila kondisi terpenuhi, maka syntax yang ada di dalam while
//akan dieksekusi.
```

```
//Struktur do...while.
int i = 0; //Bagian inisialisasi.
do
{
  digitalWrite(10, HIGH);
  delay(1000);
  digitalWrite(10, LOW);
  delay(1000);

  i = i + 1; //Bagian increment.
}
while(i < 50); //Bagian kondisi.

//Pada perulangan dengan struktur do...while, syntax yang ada di
//dalam perulangan akan terlebih dahulu dieksekusi yang kemudian
//diikuti dengan pemeriksaan kondisi. Apabila kondisi masih
//memenuhi, maka eksekusi syntax dalam perulangan akan
//dilakukan kembali.
```

2.7 LCD (Liquid Crystal Display)

Digunakan untuk menampilkan sebuah perintah yang ditampilkan pada layar tersebut yang berasal dari perintah keypad pada prototype.

Gambar 2.8 LCD (Liquid Crystal Display)



Gambar 2.9 Port LCD (Liquid Crystal Display)

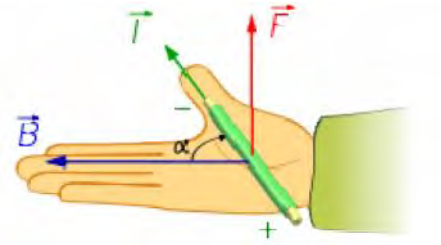
Tabel 2.6 Pin Description Port LCD (Liquid Crystal Display)

Pin No	Function	Name
1	Ground (0V)	Ground
2	Supply voltage; 5V (4.7V – 5.3V)	Vcc
3	Contrast adjustment; through a variable resistor	V_{EE}
4	Selects command register when low; and data register when high	Register Select
5	Low to write to the register; High to read from the register	Read/write
6	Sends data to data pins when a high to low pulse is given	Enable
7	8-bit data pins	DB0
8		DB1
9		DB2
10		DB3
11		DB4
12		DB5

13		DB6
14		DB7
15	Backlight V_{CC} (5V)	Led+
16	Backlight Ground (0V)	Led-

2.8 DC Geared Motor

Geared motor DC dapat didefinisikan sebagai perluasan dari motor DC. Motor Dc sendiri itu adalah motor yang mengkonversi energi listrik (sumber tegangan) ke energi mekanik (menghasilkan gerakan rotasi). Motor ini bekerja pada arus searah, motor DC bekerja pada prinsip gaya Lorentz. Gaya Lorentz adalah gaya (dalam bidang fisika) yang ditimbulkan oleh muatan listrik yang bergerak atau oleh arus listrik yang berada dalam suatu medan magnet, B. Jika ada sebuah penghantar yang dialiri arus listrik dan penghantar tersebut berada dalam medan magnetik maka akan timbul gaya yang disebut dengan nama gaya magnetik atau dikenal juga nama gaya lorentz. Arah dari gaya lorentz selalu tegak lurus dengan arah kuat arus listrik (l) dan induksi magnetik yang ada (B). Arah gaya ini akan mengikuti arah maju skrup yang diputar dari vektor arah gerak muatan listrik (v) ke arah medan magnet B.



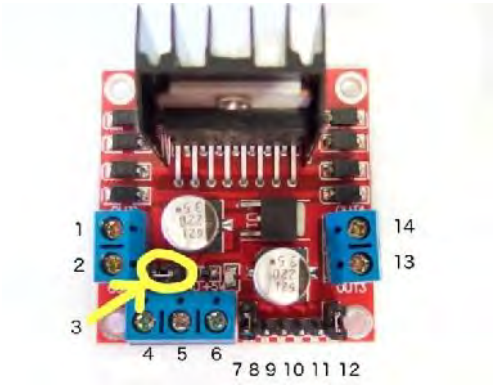
Gambar 2.10 Gaya Lorentz berdasarkan Kaidah Tangan Kanan

Geared Motor DC memiliki rakitan gigi yang melekat pada motor. Kecepatan motor dihitung dalam rotasi poros per menit dan disebut sebagai RPM. Rakitan gigi membantu dalam

meningkatkan torsi dan mengurangi kecepatan. Dengan menggunakan kombinasi gigi di gear motor, kecepatan dapat dikurangi untuk sesuai dengan yang diinginkan. Konsep ini dimana gigi mengurangi kecepatan kendaraan tetapi menambah torsi yang dikenal sebagai pengurangan gear.

Gambar 2.11 Struktur Eksternal Geared Motor DC

2.9 Motor Controller H-Bridge L298N



Gambar 2.12 H-Bridge L298N

Deskripsi bagian-bagian H-Bridge L298N :

1. DC motor 1 "+"
2. DC motor 1 "-"
3. 12V jumper - remove this if using a supply voltage greater than 12V DC. This enables power to the onboard 5V regulator
4. Connect your motor supply voltage here, maximum of 35V DC. Remove 12V jumper if >12V DC
5. GND
6. 5V output if 12V jumper in place, ideal for powering your Arduino (etc)
7. DC motor 1 enable jumper. Leave this in place when using a stepper motor. Connect to PWM output for DC motor speed control.
8. IN1
9. IN2
10. IN3
11. IN4
12. DC motor 2 enable jumper. Leave this in place when using a stepper motor. Connect to PWM output for DC motor speed control.
13. DC motor 2 "+"
14. DC motor 2 "-"

2.9.1 Pengendali DC Motor

Untuk mengendalikan satu atau dua DC Motor ini cukup mudah. Pertamasambungkan masing- masing motor untuk koneksi A dan B pada modul L298N. JikaAnda menggunakan dua motor untuk robot (dll) memastikan bahwa polaritas motoradalah sama pada kedua input. Sebaliknya Anda mungkin perlu untuk menukarmereka atas ketika Anda menetapkan kedua motors untuk maju dan pergi ke belakang.

Selanjutnya, Hubungkan power supply Anda - positif untuk pin 4 pada modul dannegatif/GND ke pin 5. Jika Anda memberikan terserah 12V Anda dapat meninggalkandi jumper

12V (titik 3 dalam gambar di atas) dan 5V akan tersedia dari pin 6 padamodul. Ini dapat dimasukkan ke Arduino Anda 5V pin untuk daya dari catu dayamotor. Jangan lupa untuk menghubungkan Arduino GND ke pin 5 pada modul serta untuk menyelesaikan rangkaian.

2.10 Sensor Rotary Encoder

Rotary Encoder adalah suatu komponen elektro mekanis yang memiliki fungsi untuk memonitoring posisi anguler pada suatu poros yang berputar. Dari perputaran benda tersebut data yang termonitoring akan diubah ke dalam bentuk data digital oleh rotary encoder berupa lebar pulsa kemudian akan dihubungkan ke kontroler (Mikrokontroler/PLC). Berdasarkan data yang di dapat berupa posisi anguler (sudut) kemudian dapat diolah oleh kontroler sehingga mendapatkan data berupa kecepatan, arah, dan posisi dari perputaran porosnya.

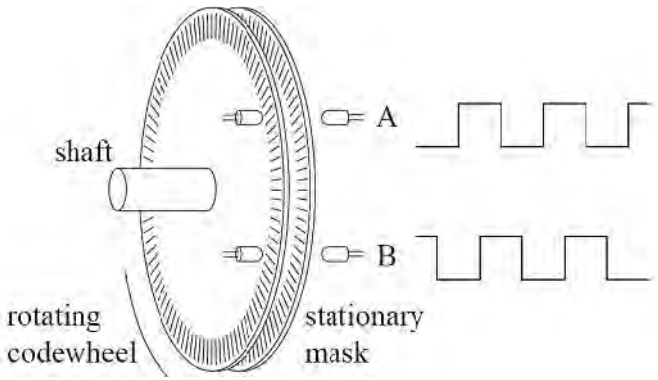
Penerapan dari penggunaan Rotary Encoder sering dijumpai pada robot-robot yang membutuhkan kepresisian tinggi dalam hal posisi seperti Robot Mechanum dan Robot Omni, selain itu untuk robot berjenis Differential Drive (ex : Robot Tank) juga disarankan menggunakan Rotary Encoder untuk mengatur agar kecepatan putar motor di roda kiri dan kanan bisa sama.

2.10.1 Konstruksi Rotary Encoder

Konstruksi Rotary Encoder berupa piringan tipis yang biasanya di kopel dengan poros yang berputar, umumnya di kopel langsung dengan shaft motor. Piringan tipis tersebut terdapat lubang di sepanjang pinggir lingkarannya. Di bagian sisi-sisi piringan terdapat sebuah led dan phototransistor di bagian bersebrangan. Fungsi dari lubang-lubang yang berada di sepanjang pinggir lingkaran tersebut akan menghantarkan cahaya led ke phototransistor, sebaliknya jika cahaya led tidak menembus lubang piringan maka cahaya akan tertahan. Piringan tersebut akan berputar sesuai dengan kecepatan putaran motor sehingga phototransistor akan saturasi ketika cahaya led menembus lubang-lubangnya.

Pada saat saturasi phototransistor akan menghasilkan pulsa dengan range +0.5V s/d +5V. Semakin banyak lubang yang berada pada piringan tentu saja semakin banyak pulsa yang dihasilkan selama satu putaran, hal tersebut berbanding lurus dengan tingkat akurasi yang dihasilkan oleh rotary encoder.
Ada 2 jenis rotary yang umum beredar di pasaran yaitu 1. Incremental Rotary Encoder dan 2. Absolute Rotary Encoder.

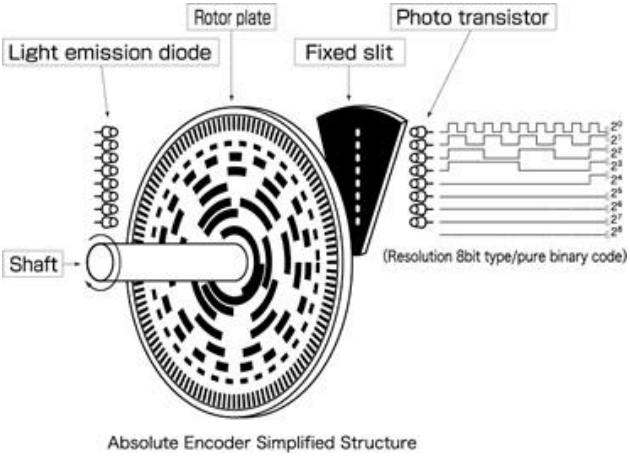
1. Incremental Rotary Encoder



Gambar 2.13 Struktur Incremental Rotary Encoder

Tipe Incremental Rotary Encoder merupakan tipe rotary encoder yang paling sederhana karena hanya dapat mengukur perubahan sudut relatifnya saja. Karena kurangnya akurasi dari incremental rotary encoder ini perlu ditambahkan satu lagi sensor optik untuk menentukan arah putaran porosnya. Dua buah sensor optik dipasang pada sudut yang berbeda sehingga arah putaran dapat diketahui, biasanya sering disebut Channel A dan Channel B.

2. Absolute Rotary Encoder



Gambar 2.14 Struktur Absolute Rotary Encoder

Piringan pada Absolute Rotary Encoder telah di desain sedemikian rupa sehingga setiap posisi dari poros dapat diterjemahkan berupa kode digital yang unik. Cincin-cincin (lubang) yang tersusun pada piringan jenis rotary ini tersusun dari bagian tengah piringan hingga ke tepi. Dimulai dari bagian tengah piringan cincin akan bertambah dua kali lipat hingga bagian pinggirnya. Sebagai contoh apabila piringan dengan 10 segmen, bagian dalamnya memiliki sebanyak 16 cincin maka bagian terluar cincin akan berjumlah 32767. Bilangan yang dihasilkan saat pembacaan nilai terhadap Absolute Rotary Encoder ini adalah berupa bilangan biner karena memilki kelipatan dua dari setiap cincinnya.

2.11 Sensor Reed Switch

Red switch adalah saklar listrik dioperasikan oleh medan magnet diterapkan. Ini diciptakan di Bell Telephone Laboratories pada tahun 1936 oleh WB Ellwood. Ini terdiri dari sepasang kontak pada alang-alang logam besi dalam amplop kaca tertutup

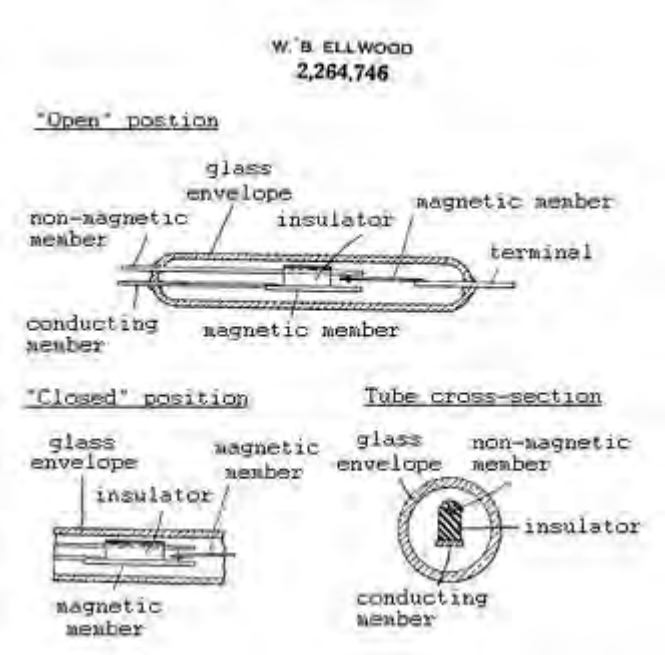
rapat. Kontak mungkin biasanya terbuka, menutup ketika medan magnet hadir, atau biasanya tertutup dan membuka ketika medan magnet diterapkan. Switch dapat digerakkan oleh coil, membuat relay buluh, atau dengan membawa sebuah magnet dekat dengan saklar. Setelah magnet ditarik jauh dari switch, saklar buluh akan kembali ke posisi semula.

Gambar 2.15 Sensor Reed Switch

Reed switch berisi sepasang (atau lebih) dari magnet, fleksibel, buluh logam yang bagian ujung dipisahkan oleh celah kecil ketika saklar terbuka. Alang-alang yang tertutup rapat di ujung-ujung amplop kaca tubular.Sebuah medan magnet (dari elektromagnet atau magnet permanen) akan menyebabkan alang-alang untuk datang bersama-sama, sehingga menyelesaikan sebuah sirkuit listrik. Kekakuan dari alang-alang menyebabkan mereka memisahkan, dan membuka sirkuit , ketika medan magnet berhenti.

Sejak kontak saklar buluh disegel dari atmosfer, mereka dilindungi terhadap korosi atmosfer. The hermetis penyegelan saklar buluh membuat cocok untuk digunakan dalam ledakan di mana percikan kecil dari switch konvensional akan merupakan bahaya.Salah satu kualitas penting dari switch adalah sensitivitas , jumlah medan magnet yang diperlukan untuk menjalankan itu . Sensitivitas diukur dalam satuan Ampere - berubah , sesuai dengan arus dalam kumparan dikalikan dengan jumlah putaran . Khas tarik -in kepekaan untuk perangkat komersial dalam 10 sampai 60 AT jangkauan. Semakin rendah AT, semakin sensitif saklar buluh. Juga, reed switch kecil , yang memiliki bagian yang

lebih kecil , lebih sensitif terhadap medan magnet , sehingga lebih kecil amplop kaca switch buluh, semakin sensitif itu



Gambar 2.16 Struktur Sensor Reed Switch

2.12 Limit Switch

Limit switch merupakan jenis saklar yang dilengkapi dengan katup yang berfungsi menggantikan tombol. Prinsip kerja limit switch sama seperti saklar Push ON yaitu hanya akan menghubung pada saat katupnya ditekan pada batas penekanan tertentu yang telah ditentukan dan akan memutus saat saat katup tidak ditekan. Limit switch termasuk dalam kategori sensor mekanis yaitu sensor yang akan memberikan perubahan elektrik saat terjadi perubahan mekanik pada sensor tersebut. Penerapan dari limit switch adalah sebagai sensor posisi suatu benda (objek)

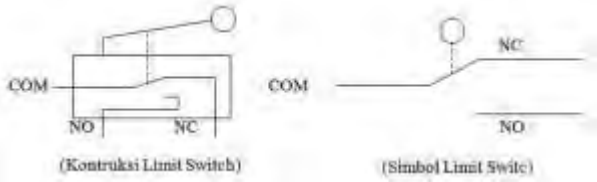
yang bergerak. Simbol limit switch ditunjukan pada gambar berikut.

Gambar 2.17 Bentuk Dan Simbol Limit Switch

Limit switch umumnya digunakan untuk :

- ✓ Memutuskan dan menghubungkan rangkaian menggunakan objek atau benda lain.
- ✓ Menghidupkan daya yang besar, dengan sarana yang kecil.
- ✓ Sebagai sensor posisi atau kondisi suatu objek.

Prinsip kerja limit switch diaktifkan dengan penekanan pada tombolnya pada batas/daerah yang telah ditentukan sebelumnya sehingga terjadi pemutusan atau penghubungan rangkaian dari rangkaian tersebut. Limit switch memiliki 2 kontak yaitu NO (Normally Open) dan kontak NC (Normally Close) dimana salah satu kontak akan aktif jika tombolnya tertekan. Konstruksi dan simbol limit switch dapat dilihat seperti gambar di bawah :



Gambar 2.18 Kontruksi dan Simbol Limit Switch

"Halaman ini sengaja dikosongkan"

BAB III

METODOLOGI

3.1. Identifikasi dan Perumusan Masalah

Tahap ini dilakukan untuk mengidentifikasikan dan merumuskan masalah yang ada. Pada skripsi ini, permasalahan yang diambil adalah sistem simulasi prototype pada automatic stacking crane

3.2. Studi Literatur

Tahap ini merupakan tahap pembelajaran mengenai teori-teori dasar yang akan dibahas pada penulisan skripsi ini. Sumber yang diambil berasal dari buku-buku, paper, internet, tutorial, regulasi, dan lain-lain yang mendukung pembahasan skripsi ini.

3.3. Pengumpulan Data

Pengumpulan data dilakukan untuk mendapatkan informasi mengenai sistem maupun cara kerja dari crane yang akan digunakan untuk pembuatan simulasi prototype

3.4. Analisa Data

Analisa data merupakan tahap di mana hasil data-data yang telah didapat, di analisa dan direncanakan sebuah sistem dengan dilakukannya penentuan peralatan yang digunakan dalam pembuatan alat pada simulasi prototype crane yang akan di rancang.

3.5. Perancangan Alat

Pada tahap perancangan peralatan dilakukan sebuah perancangan mekanik maupun sistem kerjanya agar dalam pembuatan alatnya bisa terstruktur dengan baik sampai alat tersebut beroperasi sesuai dengan apa yang telah di rencanakan sebelumnya.

3.6. Pembuatan Alat

Setelah perancangan peralatan dilakukan, maka selanjutnya adalah pembuatan peralatan. Pada tahap ini dilakukan pembuatan mekanik atau kontruksi dari crane tersebut, kemudian berlanjut pada pembuatan program sesuai dengan prinsip dan cara kerja daripada fungsi crane itu sendiri.

3.7. Pengujian Alat

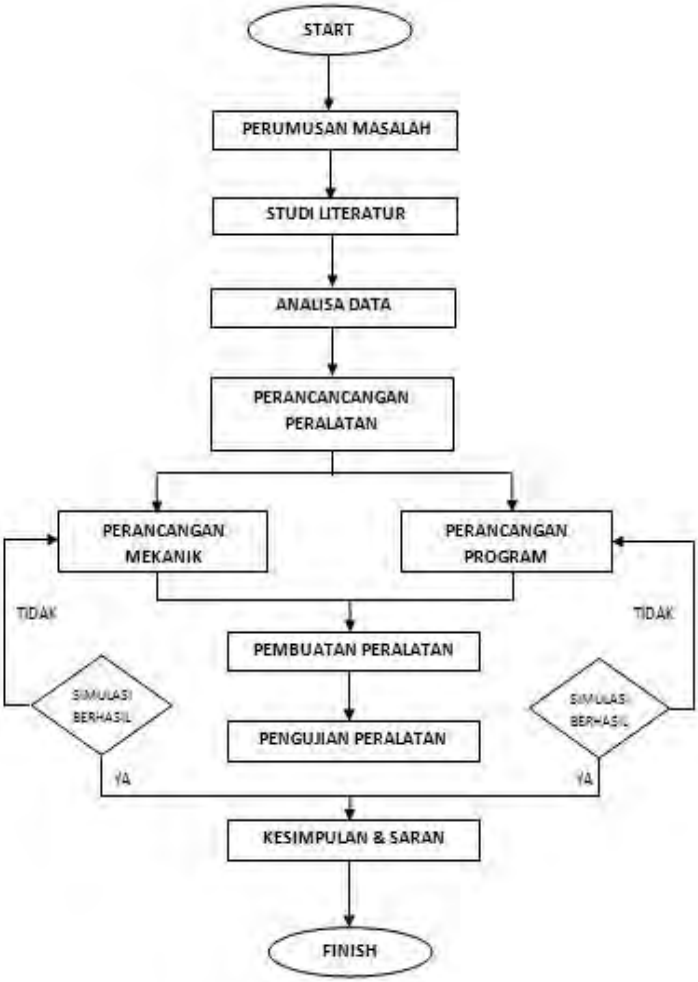
Pengujian peralatan dilakukan dengan mencocokkan perancangan alat yang dibuat dengan apa yang telah direncanakan sebelumnya. Pengujian dilakukan juga bertujuan agar dapat mengetahui apakah peralatan dapat berjalan dengan baik sesuai dengan yang diinginkan atau tidak.

3.8. Kesimpulan dan Saran

Langkah terakhir dalam penyusunan skipsi ini adalah pembuatan kesimpulan dari keseluruhan proses yang telah dilakukan sebelumnya serta memberikan jawaban atas permasalahan yang ada. Setelah membuat kesimpulan adalah memberikan saran berdasarkan hasil analisa untuk dijadikan dasar pada penelitian selanjutnya, baik terkait secara langsung pada skripsi ini ataupun data-data dan metodologi yang nantinya akan di referensi.

3.9. Flowchart

Proses perencanaan pembuatan prototype dapat dilihat pada flowchart sebagai berikut:



Gambar 3.1. Flowchat Pengerjaan Tugas Akhir

Halaman ini sengaja dikosongkan

BAB IV

ANALISA DAN PEMBUATAN PERALATAN

4.1. Pembuatan Prototype Automatic Stacking Crane

Crane adalah suatu alat yang digunakan untuk mengangkat atau memindahkan muatan berat dan banyak digunakan di pelabuhan untuk proses loading – unloading container ke truk. crane mempunyai aturan bagaimana prosedur mengangkat suatu container, ada sebuah kabel dengan payload menggantung dan akan bergerak mengangkat maupun menurunkan beban ke lokasi yang dinginkan.

Jenis automatic stacking crane mempunyai sistem kerja yang rumit dalam menjalankan fungsinya yaitu suatu alat yang digunakan untuk mengangkat atau memindahkan muatan berat dan banyak digunakan di pelabuhan untuk proses loading – unloading container, tentunya mempunyai sistem yang rumit, dari sistem yang inilah penggunaan sistem kontrol sangat perlu dilakukan. Dalam hal ini dapat di simulasikan menggunakan prototype

Pada pembuatan kontruksi prototype secara garis besar meliputi perancangan mekanik dan program. Perancangan mekanik diantaranya meliputi :

1. Pembuatan kontruksi prototype crane.
2. Perancangan trolley
3. Perancangan hoist dan gripper
4. Perancangan box container

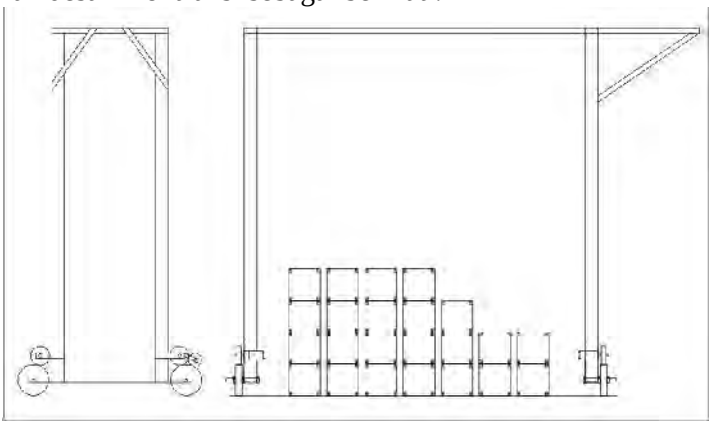
Sedangkan untuk perancangan programnya meliputi :

1. Program pada Arduino Mega 2560

4.1.1 Pembuatan Kontruksi Prototype Crane

Pada pembuatan alat ini menggunakan alumunium dengan pertimbangan bahanya yang ringan dan cukup kuat untuk

menunjang sistem mekanik yang akan dibuat nantinya. Berikut adalah desain kontruksi sebagai berikut :



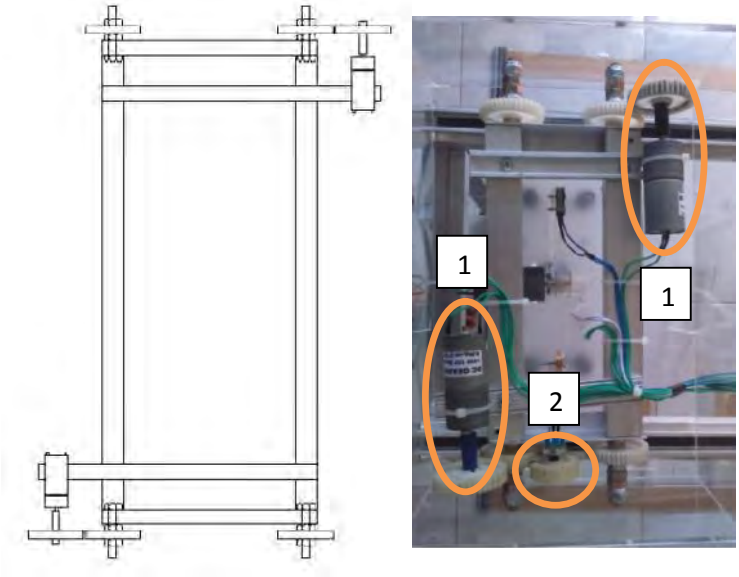
Gambar 4.1 Kontruksi Prototype Crane

4.1.2 Perancangan Trolley

Trolley disini digunakan untuk proses pergeragan secara vertigal yang pada alat ini berfungsi sebagai pemindahan container dari terminal ke truk maupun sebaliknya atau bisa juga untuk perpindahan lokasi container di dalam terminal itu sendiri.

Pada pembuatan hoist terdapat komponen dan spesifikasi alat diantaranya :

1. Motor DC geared ratio 1:21 12V 38rpm torsi 8,6kg/cm @12V
2. Sensor rotary encoder



Gambar 4.2 Kontruksi Trolley Crane

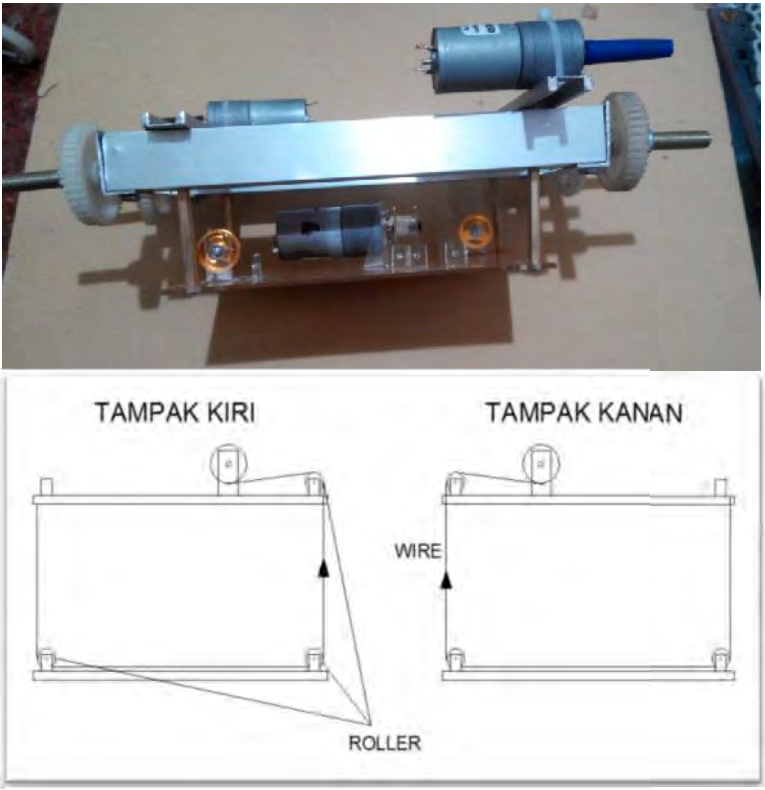
4.1.3 Perancangan Hoist dan Gripper

Hoist digunakan untuk proses naik dan turun nya container sedangkan gripper sendiri digunakan untuk penjepit atau pengunci container pada peroses pengambilan loading unloading. Hoist dan gripper di desain sedemikian rupa sehingga diharapkan dapat beroperasi sesuai dengan proses kerja yang diinginkan.

Pada pembuatan hoist dan gripper terdapat komponen beserta spesifikasi alat diantaranya :

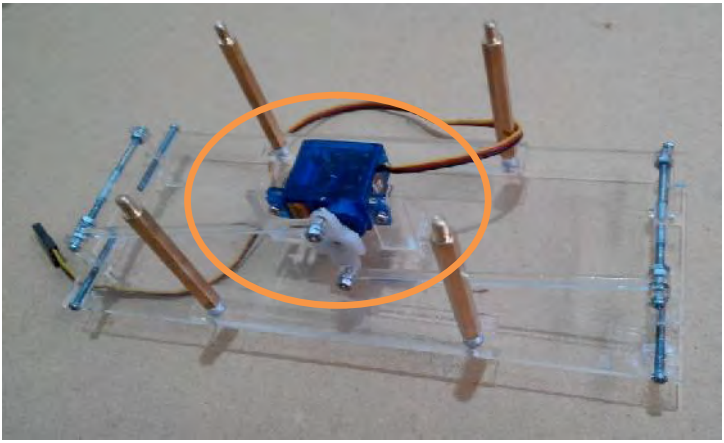
- ✓ Motor DC geared ratio 1:220 12V 38rpm torsi 8,6kg/cm @12V
- ✓ Motor mini servo
- ✓ Limit switch
- ✓ Sensor reed switch

Untuk naik turunya gripper pada proses pengambilan maupun peletakan kontainer digunakan motor DC geared yang dihubungkan melalui wire/tali dengan melewati roller yang terhubung pada gripper. Pada proses turun pendeteksi gripper jika sudah mencapai benda yang ada dibawah menggunakan limit switch yang akan aktif/mati ketika gripper telah mencapai benda yang ada dibawahnya. Sedangkan untuk pendeteksi naiknya gripper pada posisi diatas menggunakan sensor reed switch yg akan mendeteksi gripper apabila sudah mencapai titik paling atas.

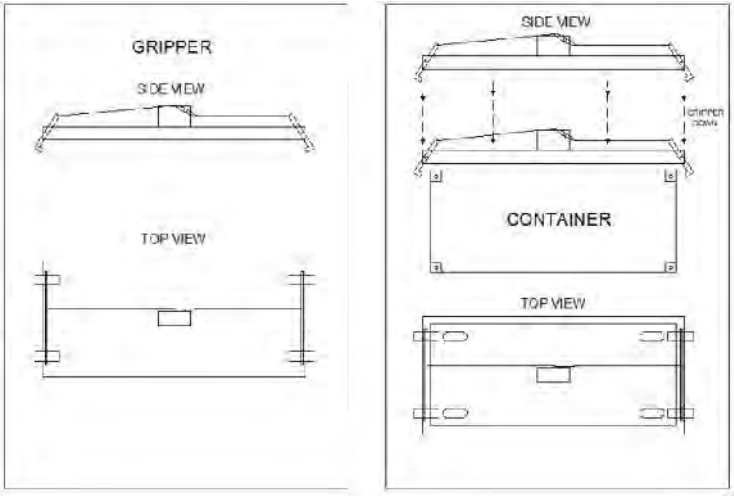


Gambar 4.3 Kontruksi Hoist Crane

Untuk proses penjepit container digunakan motor mini servo yang dihubungkan dengan engkol yang dapat berputar dan dapat mengunci dari sisi samping dari container itu sendiri. Dapat dilihat kontruksi gripper menggunakan motor mini servo pada gambar berikut :



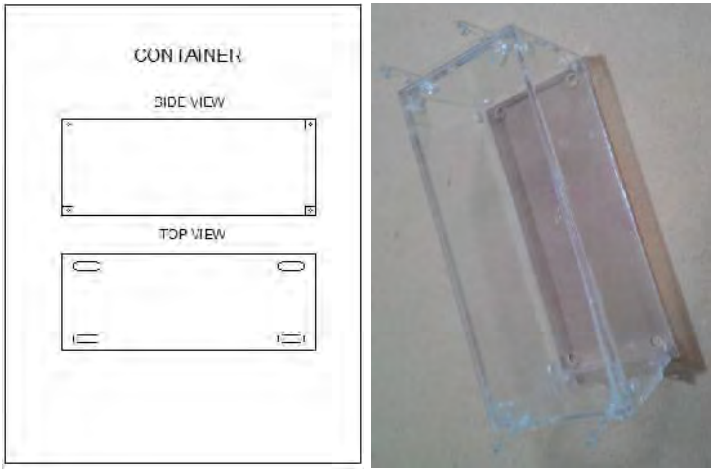
Gambar 4.4 Motor Mini Servo pada gripper



Gambar 4.5 Kontruksi Gripper Crane

4.1.4 Perancangan Box Container

Desain dari box container menggunakan skala dengan perbaindingan 1 : 40 , ukuran containernya adalah sebagai berikut : panjang 15cm, lebar 6,25cm dan tinggi 6,25cm



Gambar 4.6 Kontruksi Box Container

4.1.5 Pembuatan Program Pada Arduino Mega 2560

Arduino adalah merupakan sebuah board minimum system mikrokontroler yang bersifat open source. Didalam rangkaian board arduino terdapat mikrokontroler AVR seri ATMega 328 yang merupakan produk dari Atmel.

Arduino mega 2560 adalah papan mikrokontroler ATmega 2560 berdasarkan (datasheet) memiliki 54 digital pin input / output (dimana 15 dapat digunakan sebagai output PWM), 16 analog input, 4 UART (hardware port serial), osilator kristal 16 MHz, koneksi USB, jack listrik, header ICSP, dan tombol reset. Ini berisi semua yang diperlukan untuk mendukung mikrokontroler, hanya menghubungkannya ke komputer dengan kabel USB atau power dengan adaptor AC-DC atau baterai.

Gambar 4.7 *Board* Arduino Mega 2560
(https://www.arduino.cc/)

Tabel 4.1 Deskripsi Spesifikasi Data pada Arduino Mega 2560

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz
Length	101.52 mm
Width	53.3 mm
Weight	37 g

Langkah – langkah pembuatan program pada peralatan prototype menggunakan mikrokontroller Arduino Mega 2560.

4.1.5.1 Pembuatan program pada Keypad

Keypad disini adalah sebagai *interface* antara manusia dan alat itu sendiri untuk melakukan proses/perintah yang dilakukan

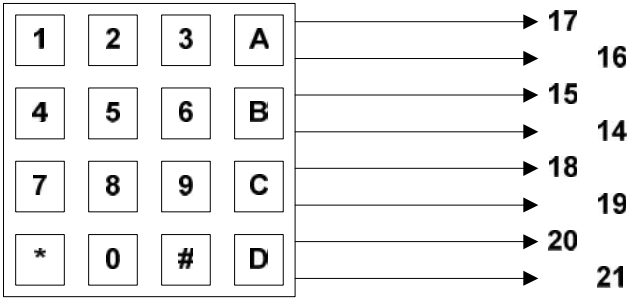
dengan menekan tombol sesuai dengan fungsi dari masing-masing tombol itu sendiri.

Pembuatan program pada keypad secara garis besar berfungsi sebagai perintah untuk melakukan proses loading-unloading pada peralatan ini yang melibatkan semua komponen yang ada pada alat tersebut. Keypad yang digunagan pada alat ini adalah keypad 4x4.

Gambar 4.8 Tampilan pada Keypad 4x4

Dengan fungsi masing-masing tombol disini adalah :

- 1, 2, 3 s/d 9 : Posisi container
- A : Tombol Start
- B : Tombol Stop
- C : Tombol Save Posisi Awal
- D : Tombol Save Posisi Awal
- * : Gripper UP-DOWN
- # : Gripper Penjepit
- 0 : Homing Posisi Awal
- Selector : Mode Auto/Manual



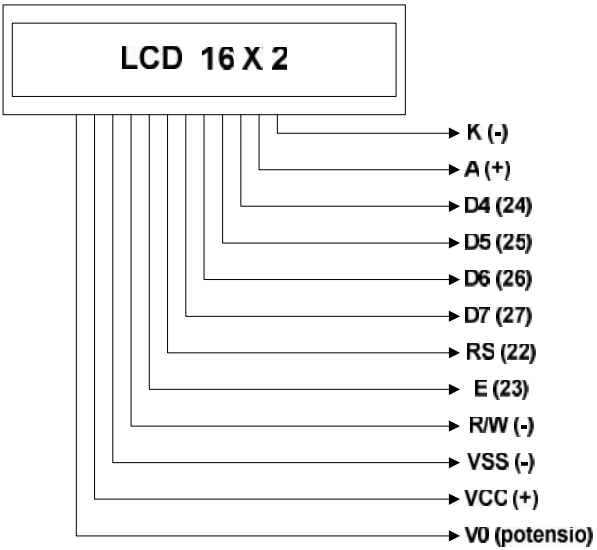
Gambar 4.9 konfigurasi keypad 4x4 yang tersambung pada Pin *board* Arduino Mega

Pada keypad 4x4 konfigurasi sambungan Pin pada *board* arduino digunakan rowPins/baris (17, 16, 15, 14) sedangkan colPins/kolom (18, 19, 20, 21)
Berikut adalah tampilan window pada sofware Arduino :

Gambar 4.10 Tampilan Window Program pada Keypad

4.1.5.2 Pembuatan program pada tampilan LCD

Pembuatan program pada LCD berfungsi untuk menampilkan perintah maupun posisi dari tujuan awal sampai ke tujuan akhir pada program yang diakhiri dengan tulisan finis. LCD jga bertujuan untuk mempermudah melihat cara kerja dari program yang dijalankan. Jenis LCD yang digunakan adalah jenis 16x2 caracter.



Gambar 4.11 konfigurasi LCD 16x2 yang tersambung pada Pin *board* Arduino Mega

Tabel 4.2 Pin Description Port LCD (Liquid Crystal Display)

Pin No	Function	Name
1	Ground (0V)	Ground
2	Supply voltage; 5V (4.7V – 5.3V)	Vcc
3	Contrast adjustment; through a variable resistor	V_{EE}

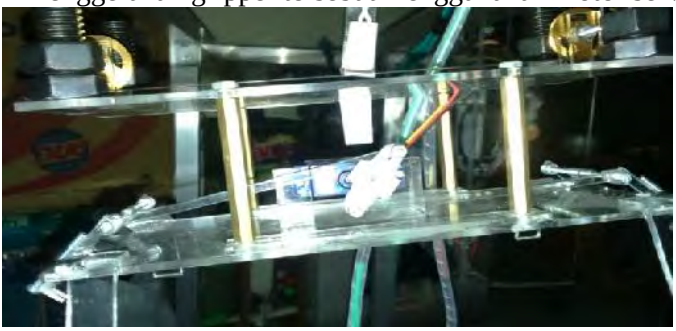
4	Low to write to the register; High to read from the register	Read/write
5	Sends data to data pins when a high to low pulse is given	Enable
6	8-bit data pins	DB0
7		DB1
8		DB2
9		DB3
10		DB4
11		DB5
12		DB6
13		DB7
14	Backlight V_{CC} (5V)	Led+
15	Backlight Ground (0V)	Led-

Berikut adalah tampilan window pada sofware Arduino :

Gambar 4.12 Tampilan window Program Pada Tampilan LCD

4.1.5.3 Pembuatan Program Gripper

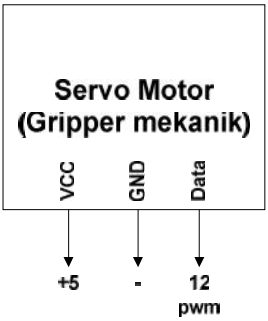
Pembuatan program pada Gripper berfungsi sebagai penjepit container box pada proses pengambilan maupun peletakan yang sebelumnya telah di setting peletakan box container tersebut. Untuk menggerakan gripper tersebut menggunakan motor servo.



Gambar 4.13 Tampilan Motor mini Servo Pada Gripper

Motor mini servo pada gripper akan berputar ke kanan-kiri yang pada program di arduino IDE sebesar :

```
int s_max_pos = 80; //maximum servo opening
int s_min_pos = 40; //minimum servo closed
int s_speed = 50;//servo gripper delay speed
int z_speed = 80;   //can be sett
```



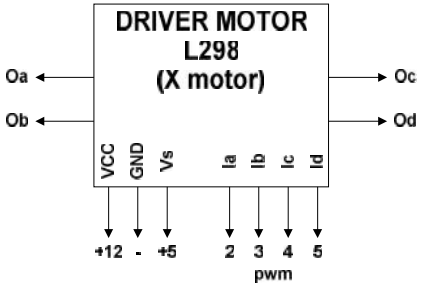
Gambar 4.14 konfigurasi motor mini servo yang tersambung pada Pin *board* Arduino Mega

Berikut adalah tampilan window pada sofware Arduino :

Gambar 4.15 Tampilan window Program Pada Gripper

4.1.5.4 Pembuatan Program Pergerakan Trolley (x_Axis)

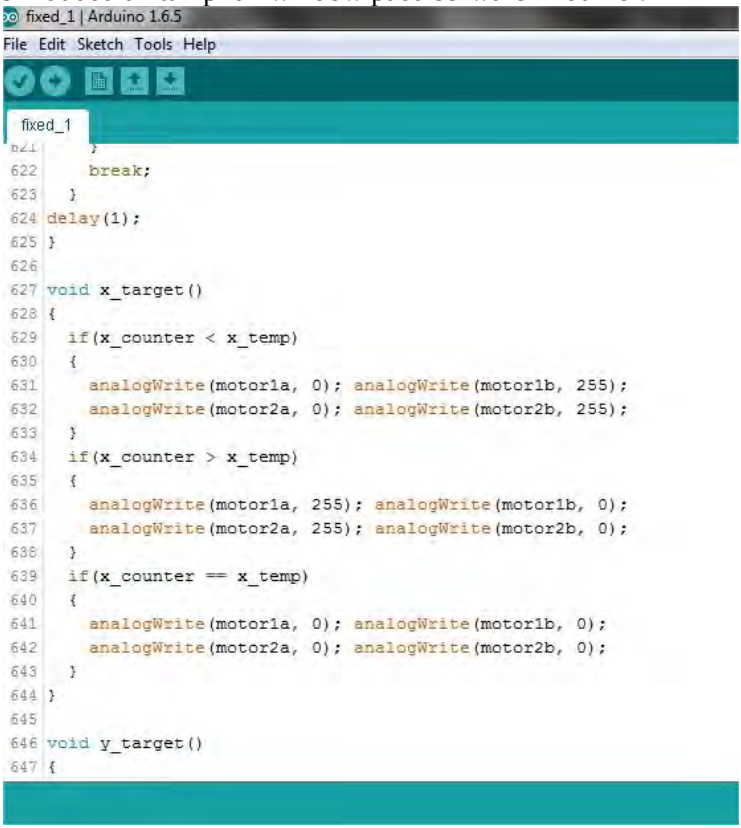
Pembuatan program pada pergerakan trolley yang bergerak secara x_axis menggunakan dua motor DC geared dengan spesifikasi ratio 1:220 12volt 38rpm dan satu sensor rotary encoder.



Gambar 4.16 konfigurasi driver motor (x_axix) yang tersambung pada Pin *board* Arduino Mega

Keterangan :
Oa & Ob = Output DC Geared motor trolley 1
Oc & Od = Output DC Geared motor trolley 2

Berikut adalah tampilan window pada sofware Arduino :

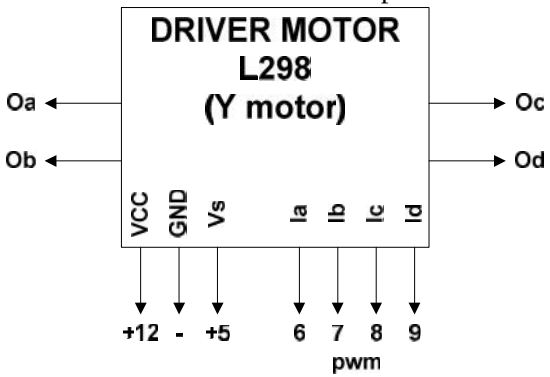


Gambar 4.17 Tampilan Program Pada Pergerakan Trolley

4.1.5.5 Pembuatan Program Pergerakan maju-mundur (Y_Axis)

Pembuatan program pada pergerakan motor maju-mundur atau (y_axis) pada pengambilan maupun peletakan box container

yang sudah disetting tata letak penempatanya. Pada pergerakan maju-mundur secara y_axis menggunakan dua motor DC geared dengan spesifikasi ratio 1:121 12volt 400rpm

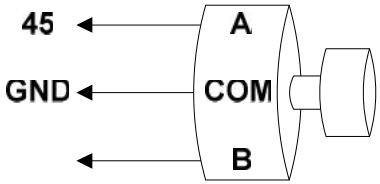


Gambar 4.18 konfigurasi driver motor (y_axix) yang tersambung pada Pin *board* Arduino Mega

Keterangan :

Oa & Ob = Output DC Geared motor maju-mundur 1
Oc & Od = Output DC Geared motor maju-mundur 2

Untuk setting posisi pada proses ppengambilan kontainer agar bisa secara otomatis berhenti sesuai *set position* yang diinginkan maka digunakan sensor encoder. Sensor ini akan menghitung berapa pulse putaran yang akan diberikan pada motor tersebut dengan dihubungkan melalui roda gigi pada motor DC geared sehingga motor akan berhenti sesuai jumlah pulse yang diberikan pada sensor encoder tersebut.



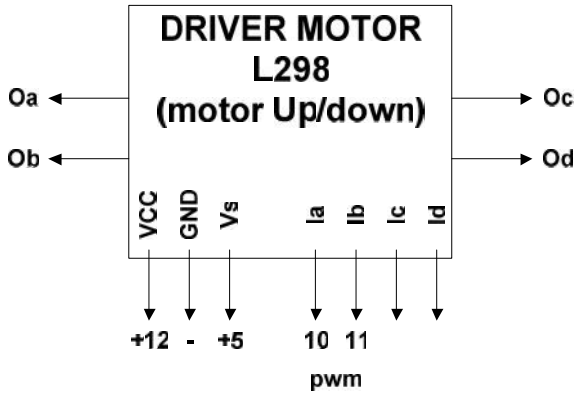
Gambar 4.19 konfigurasi sensor encoder yang tersambung pada Pin *board* Arduino Mega

Berikut adalah tampilan window pada sofware Arduino :

Gambar 4.20 Tampilan Program Pada Pergerakan Motor maju-mundur (y_axis)

4.1.5.6 Pembuatan Program Pergerakan Up-Down Pada Gripper

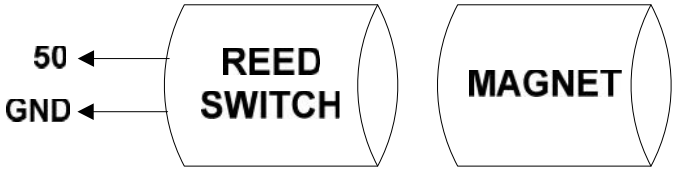
Pembuatan program pada pergerakan up-down pada gripper berfungsi untuk proses pengambilan maupun peletakan box container Pada pergerakan up-down pada gripper menggunakan satu motor DC geared dengan spesifikasi ratio 1:121 12volt 400rpm dan menggunakan dua sensor yaitu sensor reedswitch dan limit switch yang masing-masing digunakan pada proses up-down pada gripper tersebut.



Gambar 4.21 konfigurasi driver motor (up-down) yang tersambung pada Pin *board* Arduino Mega

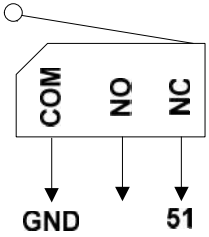
Keterangan :
Oa & Ob = Output DC Geared Up/down
Oc & Od = -

Untuk proses up/naik digunakan sensor reed switch pada pendeteksi gripper apabila telah mencapai maksimum posisi paling atas (up-limit). Dengan menempatkan sesnsor reed switch maka pada saat magnet yg terpasang pada gripper akan dideteksi oleh reed switch yang ada di atas dengan jarak ±1cm maka motor up/down akan berhenti.



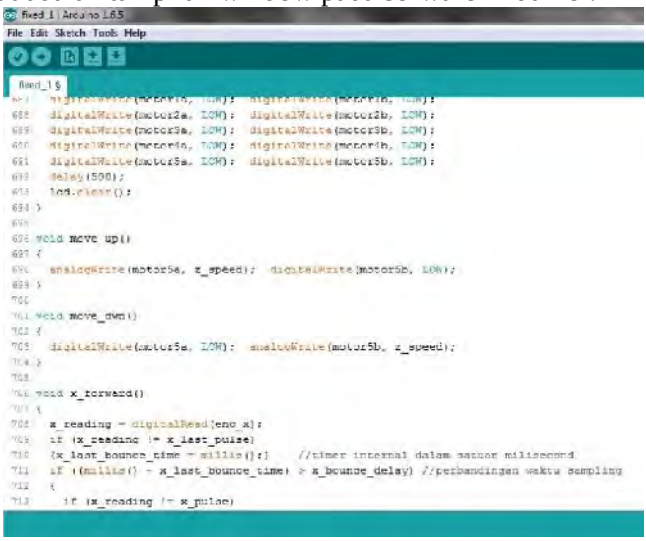
Gambar 4.22 konfigurasi sensor reed switch (up-limit) yang tersambung pada Pin *board* Arduino Mega

Pada proses down/turun digunakan limit switch pada pendeteksi gripper apabila telah mencapai maksimum posisi paling bawah (down-limit). Dengan menempatkan limit switch maka pada saat posisi turun apabila sudah mencapai posisi maksimum bawah atau pada posisi benda kerja (kontainer) maka limit switch akan berada pada kondisi NO (Normali Open).

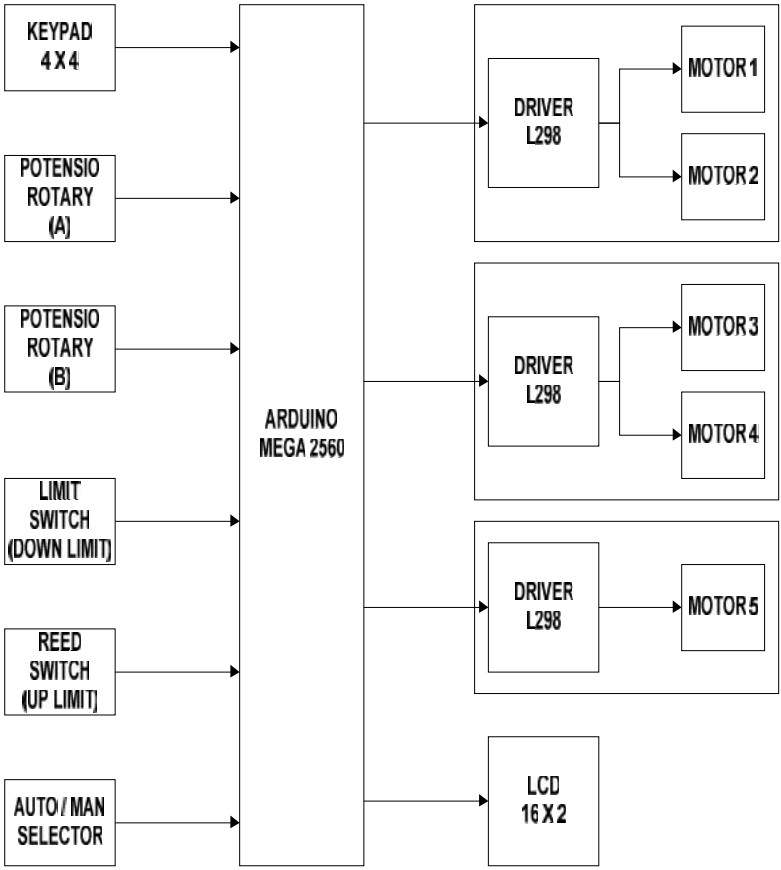


Gambar 4.23 konfigurasi sensor limit switch (down-limit) yang tersambung pada Pin *board* Arduino Mega

Berikut adalah tampilan window pada sofware Arduino :



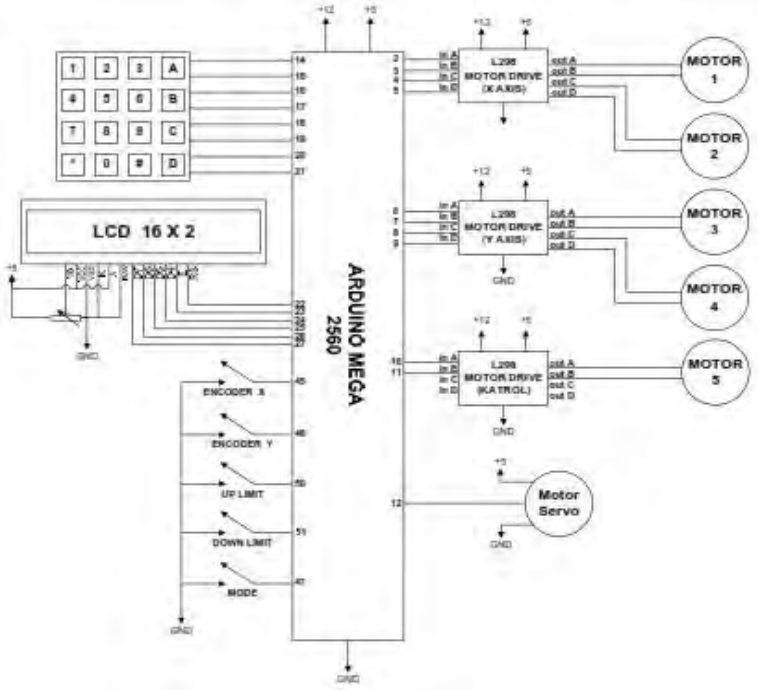
Gambar 4.24 Tampilan window Program Pada Pergerakan Up-Down Gripper



Gambar 4.25 Blok Diagram sistem secara keseluruhan

Keterangan :

Motor 1 = DC geared motor penggerak trolley
Motor 2 = DC geared motor penggerak trolley
Motor 3 = DC geared motor penggerak maju-mundur
Motor 4 = DC geared motor penggerak maju-mundur
Motor 5 = DC geared motor penggerak Up/Down



Gambar 4.26 Wiring Diagram Rangkaian Keseluruhan

Berikut adalah prosedur cara kerja dari peralatan dan juga alamat Pin yang digunakan pada modul Arduino :

Prosedur auto

1. Tekan tujuan (1 - 0)
2. Tekan tombol simpan (C)
3. Tekan tombol simpan (D)
4. Tekan start (A)

Prosedur manual

1. Tekan tujuan (1-0)
2. Tekan grip down/up (*)
3. Tekan grip/ungrip (#)

Tombol Keypad

1 – 0	koordinat
*	grip up/down
#	grip/ungrip
A	start
B	stop
C	set posisi awal
D	set posisi akhir

Tabel Wiring Input/Output Pin Pada Arduino Mega 2560

NAMA	PIN	FUNGSI	JENIS
INPUT			
Mode	47	Pilih mode otomatis / manual	Saklar/switch
Encoder X	45	Membaca input dari putaran motor sumbu X	Potensio rotary
Encoder Y	46	Membaca input dari putaran motor sumbu Y	Potensio rotary
Down limit	51	Membaca posisi terbawah katrol	Limit switch
Up limit	50	Membaca posisi teratas katrol	Reed Switch
Keypad 4x4	17, 16, 15, 14-(baris) 18, 19, 20,	Sebagai input untuk panel pengguna	Keypad 4x4

	21-(kolom)		
OUTPUT			
Servo Gripper	12(~pwm)	Menggerakkan lengan mekanis untuk mencengkram box kontainer	Servo Motor
Motor1a	2(~pwm)	Menggrakkan crane sumbu X	DC GEAR MOTOR 1:220
Motor1b	3(~pwm)		DC GEAR MOTOR 1:220
Motor2a	4(~pwm)	Menggrakkan crane sumbu X	DC GEAR MOTOR 1:220
Motor2b	5(~pwm)		DC GEAR MOTOR 1:220
Motor3a	6(~pwm)	Menggrakkan crane sumbu Y	DC GEAR MOTOR 1:20
Motor3b	7(~pwm)		DC GEAR MOTOR 1:20
Motor4a	8(~pwm)	Menggrakkan crane sumbu Y	DC GEAR MOTOR 1:20
Motor4b	9(~pwm)		DC GEAR MOTOR 1:20
Motor5a	10(~pwm)	Mengangkat katrol	DC GEAR MOTOR 1:220
Motor5b	11(~pwm)		DC GEAR MOTOR 1:220
LCD 16 X 2	22 rs 23 e 24 d4 25 d5 26 d6 27 d7	Untuk display	LCD 16 x 2

4.2 Uji Coba Protptype

Dalam melakukan uji coba untuk mengetahuincara kerja dari prototype dapat bekerja sesuai dengan yang diinginkan dibutuhkan beberapa komponen yang dibutuhkan seperti :

- Arduino mega 2560

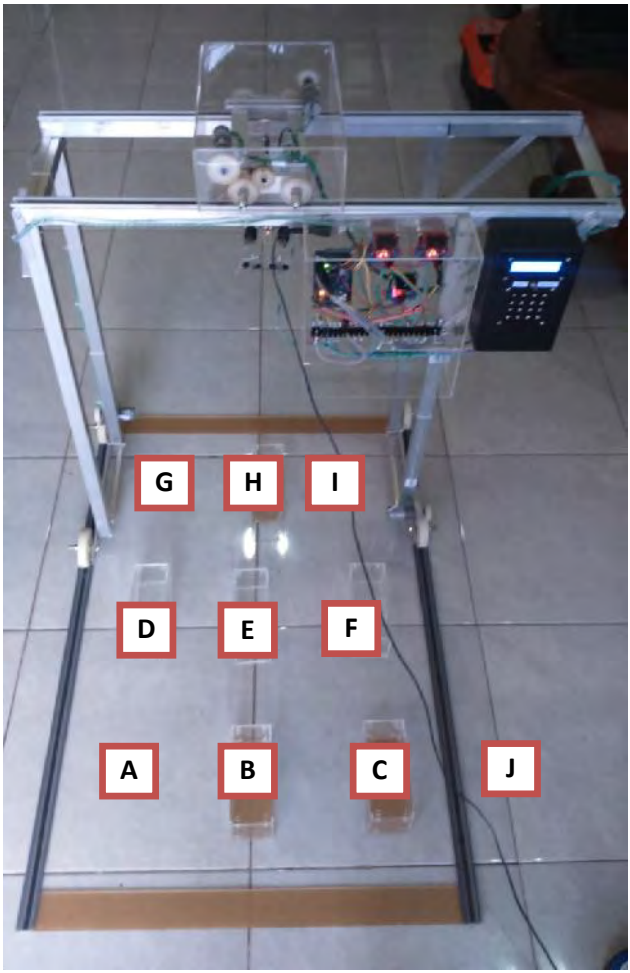
  Digunakan sebagai pusat proses pada keseluruhan sistem baik membaca input dan mengeluarkan output yang dibutuhkan oleh pengguna sesuai dengan program yang telah dirancang untuk memindahkan miniatur container pada posisi yang ingin dituju.
- Driver motor L298

  Digunakan sebagai penguat sinyal input PWM yang berasal dari arduino mega 2560 untuk menggerakkan motor. Driver motor L298 diaplikasikan untuk menggerakkan motor pada sumbu X, sumbu Y dan katrol
- Potensio rotary

  Digunakan untuk membaca posisi dari pergerakan crane masing-masing pada sumbu X dan sumbu Y. Output daripada potensio rotary berupa pulsa yang berasal dari pergerakan poros potensio rotary yang terhubung pada motor penggerak sumbu X dan sumbu Y, setiap satu kali putaran potensio terdapat 20 pulsa output yang dapat diolah oleh arduino untuk mentukan posisi.
- Limit switch

  Digunakan untuk membaca batas bawah daripada katrol dengan memanfaatkan tingkat beban tali katrol.
- Reed switch

  Digunakan untuk membaca batas atas daripada katrol saat ada kontak dengan magnet yang terhubung pada posisi puncak katrol.



Gambar 4.27 Prototype Secara Keseluruhan

Tabel 4.2 Fungsi Tombol Pada Keypad

Tombol	Mode Manual	Mode Otomatis
1	Bergerak pada posisi A	Ambil data A

2	Bergerak pada posisi B	Ambil data B
3	Bergerak pada posisi C	Ambil data C
4	Bergerak pada posisi D	Ambil data D
5	Bergerak pada posisi E	Ambil data E
6	Bergerak pada posisi F	Ambil data F
7	Bergerak pada posisi G	Ambil data G
8	Bergerak pada posisi H	Ambil data H
9	Bergerak pada posisi I	Ambil data I
0	Bergerak pada posisi J	Ambil data J
*	Katrol naik/turun	-
#	Katrol jepit/lepas	-
A	-	Start
B	Stop	
C	-	Simpan posisi target
D	-	Simpan posisi tujuan

Dalam pengoperasian alat tersebut ada dua mode yang bisa digunagan yaitu : mode manual dan mode auto, Petunjuk pengoperasian manual pada perangkat dapat dilakukan ketika posisi saklar mode berada pada posisi manual.

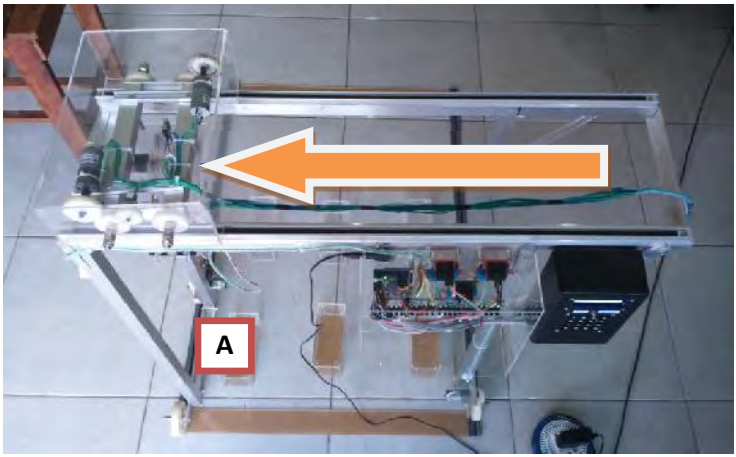
1. Pengujian Pada Mode Manual.

Gambar 4.28 pengujian posisi selektor pada mode manual

Pengoperasian sistem dapat dilakukan sesuai dengan tabel petunjuk penggunaan sistem. Pengguna cukup menekan tombol 0-9 seperti gambar dibawah ini pemindahan pada posisi “A”.

Gambar 4.29 Pengoperasian Keypad Dengan Perintah Pada Posisi“A”

Setelah tombol 1 ditekan maka trolley akan bergerak ke koordinat titik “A” sesuai dengan set posisi penempatan kontainer yang ada pada gambar sebelumnya. Gambar dibawah ini menunjukan pergerakan trolley pada posisi koordinat titik “A”.



Gambar 4.30 Pergerakan Trolley Pada Koordinat titik “a”

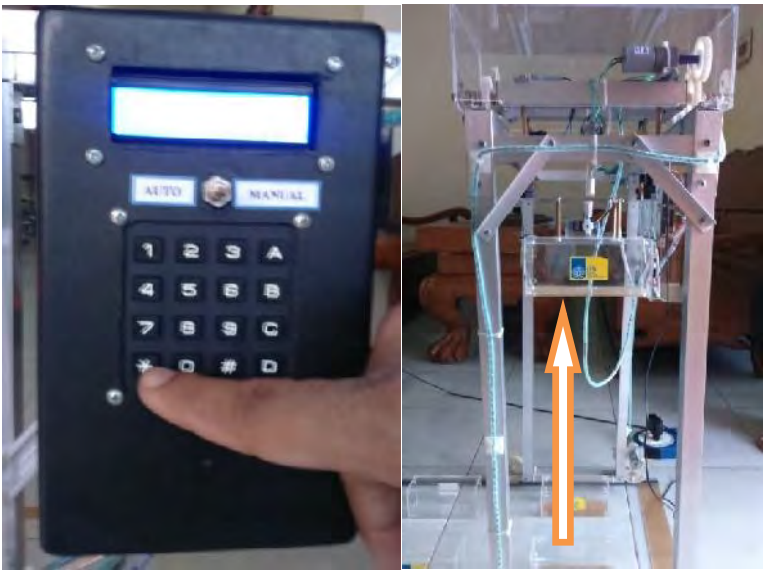
Langkah selanjutnya adalah pengambilan container yang ada dititik “A” dengan menurunkan gripper sampai dengan berada ke kontainer yang berada pada titik “A” tersebut. Dengan cara menekan tombol “*”

Gambar 4.31 Proses Gripper Turun

Setelah gripper turun maka langkah selanjutnya adalah perintah penjepit dengan cara menekan tombol “*” maka kontainer akan terkunci

Gambar 4.32 Prosses Gripper Mengunci Kontainer

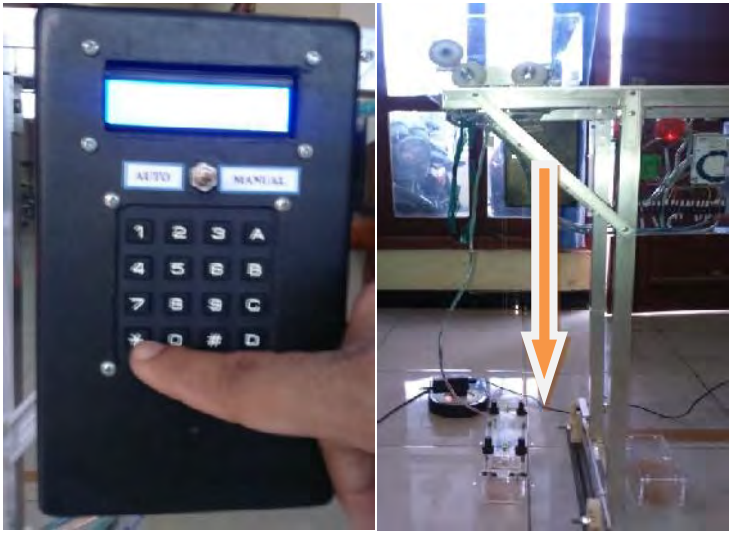
Setelah kontainer terkunci maka langkah berikutnya adalah perintah naik yaitu dengan cara menekan tombol “*” maka gripper dengan kontainer yang telah terkunci tadi akan terangkat naik sampai batas paling atas dengan sensor reed switch yang akan mendeteksi gripper berada pada posisi maksimal Up/naik. Berikut adalah gambar proses gripper naikdengan tombol “*” untuk perintah naik nya :



Gambar 4.33 Proses Gripper Naik

Setelah proses gripper naik pada posisi maksimal atas maka langkah selanjutnya adalah menekan tombol dengan lokasi yang diinginkan yang disini adalah pemilihan penempatan lokasi adalah “J”. Setelah didapat lokasi yang telah dituju maka proses berikutnya adalah peletakan gripper pada posisi tersebut dengan cara mekan tombol “*” kemudian dilanjutkan dengan tombol “#” setelah kontainer mencapai posisi bawah maka gripper yang dalam kondisi mengunci kontainer akan terlepas dan bisa dilanjutkan dengan perintah berikutnya sesuai dengan posisi kontainer yang ingin dipindahkan berturut-turut sesuai dengan langkah-langkah yang telah dilakukan sebelumnya.

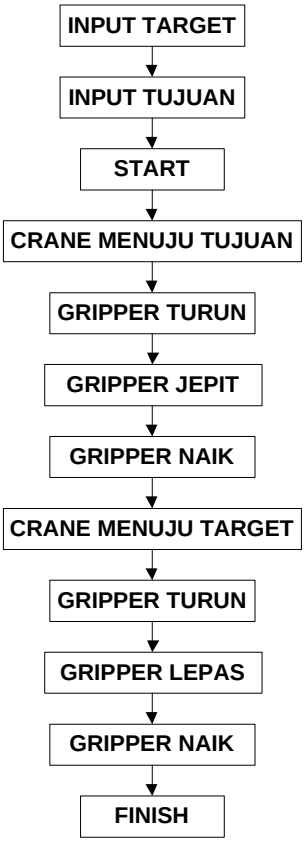
Gambar 4.34 Proses Trolley ke Posisi “J”



Gambar 4.35 Proses Gripper Turun

2. Pengujian Pada Mode Auto.

Untuk menggerakkan crane secara otomatis dapat dilakukan dengan negarahkan selektor pada posisi autokemudian input data terlebih dahulu. Alur kerja daripada sistem berjalan secara sistematis dan berurutan setelah tombol start ditekan.



Gambar 4.35 Blok Diagram Alur Pengoperasian Mode Auto

Untuk melakukan input target terlebih dahulu pengguna harus mengisi data target dengan cara memilih posisi target sesuai dengan posisi 0 – 9

Gambar 4.36 Tombol Perintah Input Target Pada Mode Auto

lalu tekan tombol “C” untuk menyimpan posisi tujuan. Sedangkan untuk melakukan input tujuan pengguna dapat memilih posisi dengan cara menekan angka 0 – 9 lalu tekan tombol “D” untuk menyimpan posisi target. Setelah mengisi posisi target dan tujuan makan pengguna dapat memulai prosedur dengan cara menekan tombol “A”

Gambar 4.37 Fungsi Start, Stop, Posisi Awal dan Posisi Akhir

Dari hasil uji coba alat tersebut didapatkan kecepatan pada proses pemindahan kontainer dengan data dibawah ini :

Kecepatan motor trolley	= 3,99 m/min
Kecepatan motor gripper up (load)	= 1,56 m/min
Kecepatan motor gripper up(unload)	= 1,74 m/min

Kecepatan motor gripper down(load)	= 1,56 m/min
Kecepatan motor gripper down(unload)	= 2,28 m/min
Kecepatan motor gripper maju-mundur	= 3,03 m/min
Grip/ungrip	= 2 detik

"Halaman ini sengaja dikosongkan"

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan analisa dan uji coba alat serta pembahasan yang telah dilakukan, maka dapat diambil kesimpulan sebagai berikut :

1. Perancangan prototype automatic stacking crane dengan menggunakan mikrokontroller berupa arduino dapat bekerja dengan baik sesuai dengan desain yang telah dirancang sebelumnya.
2. Saat diberikan perintah manual dan dilakukan inisialisasi pergerakan gripper cukup stabil meskipun ada sedikit guncangan karena desain mekanik nya kurang sempurna .
3. Kekurangan dari perancangan ini adalah tegangan yang masuk ke motor bersumber dari arduino yaitu 5 v, sehingga perputaran motor tidak maksimal. Selain itu, respon input masih banyak terdapat prosedur yang masih kurang sesuai.
4. Peletakan kabel pada gripper dan juga kabel untuk trolley perlu mekanisme peletakan yang baik agar tidak menggangu pergerakan dari trolley maupun gripper pada proses kerja alat tersebut.
5. Dari hasil uji coba alat tersebut didapatkan kecepatan pada proses pemindahan kontainer dengan kecepatan motor trolley 3,99m/min, kecepatan motor gripper up(load) 1,56 m/min, Kecepatan motor gripper up(unload) 1,74 m/min, Kecepatan motor gripper down(load) 1,56 m/min, Kecepatan motor gripper down(unload) 2,28 m/min, Kecepatan motor gripper maju-mundur 3,03 m/min, Grip/ungrip 2 detik

5.2 Saran

Adapun saran yang dapat disampaikan oleh penulis setelah selesainya skrips iadalah:

1. Pengerjaan skripsi ini masih banyak kekurangan sehingga untuk kedepannya dapat dikembangkan lagi. Dari kontruksi sebaiknya dibuat menggunakan peralatan khusus dengan material yang lebih baik sehingga kerja alat bisa maksimal
2. Pembuatan mekanik yang lebih baik akan membuat sistem kerja dari alat akan sesuai dengan yang kita inginkan dan menghindari terjadinya error pada saat uji coba karena sistem mekanik sangat penting untuk memaksimalkan program yang akan kita buat.
3. Pemahaman software sangat penting agar tidak menemui kesulitan pada saat pembuatan programnya.
4. Pengembangan prototype crane masih banyak yang bisa dilakukan sesuai dengan kreatifitas yang nantinya bisa diaplikasikan pada peralatan yang sesungguhnya guna untuk meringankan pekerjaan manusia terutama pada proses bongkar muat container pada pelabuhan.

DAFTAR PUSTAKA

Arduino, 2015. **Built-In Examples**
https://www.arduino.cc/en/Tutorial/BuiltInExamples

Container Crane, "DC Drive" 2013
https://containercrane.wordpressCom/2009/10/19/dc-drive/

Kaunang, Jason."*Perangkat Transmisi Otomatis Pada Motor Berbasis Mikrokontroller*", jurusan Teknik Elektro Universitas Widya Kartika, Surabaya 2015

Malvino Albert Paul,*"Prinsip-Prinsip Elektronika Edisi Ketiga Jilid 2"*, Penerbit Erlangga, Jakarta, 1996

M.H Rashid, "*Power Electronic Circuit, Device, and Applications"*, 3rd Ed., Prentice Hall, 2004

Nuzula, Arma," *Analisa Modernisasi Sistem Otomasi Pada Container Crane Dengan Menggunakan Plc Siemens*" Jurusan Sistem Perkapalan FTK-ITS, Surabaya 2015

"Halaman ini sengaja dikosongkan"

```
//Library
#include <Keypad.h>
#include <Servo.h>
#include <EEPROM.h>
#include <LiquidCrystal.h>

//EEPROM address
int data_x = 127;
int data_y = 128;
byte data_counter_x;
byte data_counter_y;

//LCD 16x2 configuration
LiquidCrystal lcd(22,23,24,25,26,27);

//keypad configuration
const byte ROWS = 4;
const byte COLS = 4;
char hexaKeys[ROWS][COLS] =
{ {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'E','0','F','D'}};

byte rowPins[ROWS] = {17, 16, 15, 14};
byte colPins[COLS] = {18, 19, 20, 21};
Keypad customKeypad = Keypad(
makeKeymap(hexaKeys), rowPins, colPins,
ROWS, COLS);
```

```
//INPUT
int up_limit = 50;
int dwn_limit = 51;
int enc_x = 45;
int enc_y = 46;
int mode = 47;
//OUTPUT (Use PWM Pin only)
Servo m_servo; //Servo gripper motor

int motor1a = 2;  //x axis motor
int motor1b = 3;

int motor2a = 4;  //x axis motor
int motor2b = 5;

int motor3a = 6;  //y axis motor
int motor3b = 7;

int motor4a = 8;  //y axis motor
int motor4b = 9;

int motor5a = 10;  //z axis motor
int motor5b = 11;

//INITIAL
int delay_process = 500;
int s_speed = 50;  //servo gripper delay
speed
int z_speed = 80;  //can be sett
int s_max_pos = 80; //maximum servo opening
```

```
int s_min_pos = 40;  //minimum servo closed
position *fixed position by measurement
int s_pos = 0;

//pengengenalan posisi
char locA = 'A';
int x_pos_a = 0;    //row 1
int y_pos_a = 0;     //col 1

char locB = 'B';
int x_pos_b = 50;    //row 2
int y_pos_b = 0;     //col 1

char locC = 'C';
int x_pos_c = 100;    //row 3
int y_pos_c = 0;     //col 1

char locD = 'D';
int x_pos_d = 0;    //row 4
int y_pos_d = 50;     //col 1

char locE = 'E';
int x_pos_e = 50;    //row 2
int y_pos_e = 50;    //col 2

char locF = 'F';
int x_pos_f = 100;    //row 3
int y_pos_f = 50;    //col 2

char locG = 'G';
int x_pos_g = 0;    //row 4
```

```
int y_pos_g = 100;    //col 2

char locH = 'H';
int x_pos_h = 50;    //row 2
int y_pos_h = 100;    //col 3

char locI = 'I';
int x_pos_i = 100;    //row 3
int y_pos_i = 100;    //col 3

char locJ = 'J';
int x_pos_j = 150;    //row 4
int y_pos_j = 0;    //col 3

//initial condition
int s_stat = 0;
int x_temp = 0;
int y_temp = 0;
int x_tgt = 0;
int y_tgt = 0;
int x_dest = 0;
int y_dest = 0;
int sequence = 0;
char loc_temp;
char tgt_temp;
char des_temp;

//x axis counter initialization
int x_reading;
int x_counter = 0;
int x_pulse;
```

```
int x_last_pulse = HIGH;

long x_last_bounce_time = 0;
long x_bounce_delay = 1;

//y axis counter initialization
int y_reading;
int y_counter = 0;
int y_pulse;
int y_last_pulse = HIGH;

long y_last_bounce_time = 0;
long y_bounce_delay = 1;

void setup() {
  //attach INPUT_PULLUP to GND (LOW if
Contact, HIGH if Open)
  lcd.begin(16,2);
  lcd.print("FAJAR ANDIK C.");
  lcd.setCursor(0, 1);
  lcd.print("4213106005");
  delay(2000);
  lcd.clear();

  Serial.begin(9600);
  pinMode(up_limit,       INPUT_PULLUP);
//digital pin only
  pinMode(dwn_limit,      INPUT_PULLUP);
//digital pin only
  pinMode(enc_x,          INPUT_PULLUP);
//digital pin only
```

```
  pinMode(enc_y,           INPUT_PULLUP);
//digital pin only
  pinMode(mode,            INPUT_PULLUP);
//digital pin only
  m_servo.attach(12); //servo pin
  pinMode(motor1a,   OUTPUT); //pwm pin only
  pinMode(motor1b,   OUTPUT); //pwm pin only
  pinMode(motor2a,   OUTPUT); //pwm pin only
  pinMode(motor2b,   OUTPUT); //pwm pin only
  pinMode(motor3a,   OUTPUT); //pwm pin only
  pinMode(motor3b,   OUTPUT); //pwm pin only
  pinMode(motor4a,   OUTPUT); //pwm pin only
  pinMode(motor4b,   OUTPUT); //pwm pin only
  pinMode(motor5a,   OUTPUT); //pwm pin only
  pinMode(motor5b,   OUTPUT); //pwm pin only

  //homing(); //use routine if needed
  x_counter = EEPROM.read(data_x);
  y_counter = EEPROM.read(data_y);
  x_temp = 150;
  y_temp = 0;
  sequence = 3;
}

void loop() {
  // put your main code here, to run
repeatedly:
  up_limit = digitalRead(50);
  dwn_limit = digitalRead(51);
  enc_x = digitalRead(45);
  enc_y = digitalRead(46);
```

```
  mode = digitalRead(47);

  //x_counter = EEPROM.read(data_x);
  //y_counter = EEPROM.read(data_y);

  char key = customKeypad.getKey();
  if(key == '1')
  {
    if(mode == HIGH)
    {
      x_temp = x_pos_a;
      y_temp = y_pos_a;
      lcd.setCursor(0, 0);
      lcd.clear();
      lcd.print("pos A");
      sequence = 3;
    }
    else
    {
      x_temp = x_pos_a;
      y_temp = y_pos_a;
      loc_temp = locA;
      lcd.setCursor(0, 0);
      lcd.clear();
      lcd.print("ke pos A?");
    }
  }
  else if(key == '2')
  {
    if(mode == HIGH)
    {
```

```
      x_temp = x_pos_b;
      y_temp = y_pos_b;
      lcd.setCursor(0, 0);
      lcd.clear();
      lcd.print("pos B");
      sequence = 3;
    }
    else
    {
      x_temp = x_pos_b;
      y_temp = y_pos_b;
      loc_temp = locB;
      lcd.setCursor(0, 0);
      lcd.clear();
      lcd.print("ke pos B?");
    }
  }
  else if(key == '3')
  {
    if(mode == HIGH)
    {
      x_temp = x_pos_c;
      y_temp = y_pos_c;
      lcd.setCursor(0, 0);
      lcd.clear();
      lcd.print("pos C");
      sequence = 3;
    }
    else
    {
      x_temp = x_pos_c;
```

```
      y_temp = y_pos_c;
      loc_temp = locC;
      lcd.setCursor(0, 0);
      lcd.clear();
      lcd.print("ke pos C?");
    }
  }
  else if(key == '4')
  {
    if(mode == HIGH)
    {
      x_temp = x_pos_d;
      y_temp = y_pos_d;
      lcd.setCursor(0, 0);
      lcd.clear();
      lcd.print("pos D");
      sequence = 3;
    }
    else
    {
      x_temp = x_pos_d;
      y_temp = y_pos_d;
      loc_temp = locD;
      lcd.setCursor(0, 0);
      lcd.clear();
      lcd.print("ke pos D?");
    }
  }
  else if(key == '5')
  {
    if(mode == HIGH)
```

```
    {
      x_temp = x_pos_e;
      y_temp = y_pos_e;
      lcd.setCursor(0, 0);
      lcd.clear();
      lcd.print("pos E");
      sequence = 3;
    }
    else
    {
      x_temp = x_pos_e;
      y_temp = y_pos_e;
      loc_temp = locE;
      lcd.setCursor(0, 0);
      lcd.clear();
      lcd.print("ke pos E?");
    }
  }
  else if(key == '6')
  {
    if(mode == HIGH)
    {
      x_temp = x_pos_f;
      y_temp = y_pos_f;
      lcd.setCursor(0, 0);
      lcd.clear();
      lcd.print("pos F");
      sequence = 3;
    }
    else
    {
```

```
      x_temp = x_pos_f;
      y_temp = y_pos_f;
      loc_temp = locF;
      lcd.setCursor(0, 0);
      lcd.clear();
      lcd.print("ke pos F?");
    }
  }
  else if(key == '7')
  {
    if(mode == HIGH)
    {
      x_temp = x_pos_g;
      y_temp = y_pos_g;
      lcd.setCursor(0, 0);
      lcd.clear();
      lcd.print("pos G");
      sequence = 3;
    }
    else
    {
      x_temp = x_pos_g;
      y_temp = y_pos_g;
      loc_temp = locG;
      lcd.setCursor(0, 0);
      lcd.clear();
      lcd.print("ke pos G?");
    }
  }
  else if(key == '8')
  {
```

```
    if(mode == HIGH)
    {
      x_temp = x_pos_h;
      y_temp = y_pos_h;
      lcd.setCursor(0, 0);
      lcd.clear();
      lcd.print("pos H");
      sequence = 3;
    }
    else
    {
      x_temp = x_pos_h;
      y_temp = y_pos_h;
      loc_temp = locH;
      lcd.setCursor(0, 0);
      lcd.clear();
      lcd.print("ke pos H?");
    }
  }
  else if(key == '9')
  {
    if(mode == HIGH)
    {
      x_temp = x_pos_i;
      y_temp = y_pos_i;
      lcd.setCursor(0, 0);
      lcd.clear();
      lcd.print("pos I");
      sequence = 3;
    }
    else
```

```
    {
      x_temp = x_pos_i;
      y_temp = y_pos_i;
      loc_temp = locI;
      lcd.setCursor(0, 0);
      lcd.clear();
      lcd.print("ke pos I?");
    }
  }
  else if(key == '0')
  {
    if(mode == HIGH)
    {
      x_temp = x_pos_j;
      y_temp = y_pos_j;
      lcd.setCursor(0, 0);
      lcd.clear();
      lcd.print("pos J");
      sequence = 3;
    }
    else
    {
      x_temp = x_pos_j;
      y_temp = y_pos_j;
      loc_temp = locJ;
      lcd.setCursor(0, 0);
      lcd.clear();
      lcd.print("ke pos J?");
    }
  }
  else if(key == 'A')
```

```
  {
    if(mode == LOW && sequence == 0)
    {
      lcd.setCursor(0, 0);
      lcd.clear();
      lcd.print("Auto start!!");
      lcd.setCursor(0, 1);
      lcd.print(tgt_temp);
      lcd.setCursor(5, 1);
      lcd.print("ke");
      lcd.setCursor(10, 1);
      lcd.print(des_temp);
      if(sequence == 0)
      {
      sequence = 5;
      }
  }
    //jump to starting sequential process
  }
  else if(key == 'B')
  {
    lcd.setCursor(0, 0);
    lcd.print("Stop!!");
    stop_all();
    sequence = 0;  //reset to zero
  }
  else if(key == 'C')
  {
    x_tgt = x_temp;  //write to x target
position
```

```
    y_tgt = y_temp;  //write to y target
position
    tgt_temp = loc_temp;
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print(tgt_temp);
  }
  else if(key == 'D')
  {
    x_dest = x_temp;  //write to x
destination
    y_dest = y_temp;  //write to y
destination
    des_temp = loc_temp;
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print(des_temp);
  }
  else if(key == 'E')
  {
      if((dwn_limit == LOW || up_limit !=
LOW) && mode == HIGH)
      {sequence = 1;}
      if(up_limit == LOW && mode == HIGH)
      {sequence = 2;}
      if(up_limit != LOW && dwn_limit != LOW
&& mode == HIGH)
      {sequence = 2;}
  }
  else if(key == 'F')
```

```
  {                    //grip/ungrip manual
mode
    if(s_stat == 0 && mode == HIGH)
    {ungrip();}
    else if(s_stat == 1 && mode == HIGH)
    {grip();}
  }

  switch(sequence)  //urutan jalan program.
yang dijalankan adalah case ke n dimana n =
sequence
  {
    case 1:
    move_up();      //memanggil void dengan
nama tersebut
    if(up_limit == LOW)
    {
      zero_speed();
      sequence = 0;
    }
    break;          //keluar dari program
case

    case 2:
    move_dwn();
    if(dwn_limit == LOW)
    {
      zero_speed();
      sequence = 0;
    }
    break;
```

```
    case 3:
    if(x_counter < x_temp)
    {
      Serial.println("+");
      x_forward();
    }
    if(x_counter > x_temp)
    {
      Serial.println("-");
      x_reverse();
    }
    if(x_counter == x_temp)
    {
    sequence = 4;
    }
    break;

    case 4:
    if(y_counter < y_temp)
    {
      Serial.println("++");
      y_forward();
    }
    if(y_counter > y_temp)
    {
      Serial.println("--");
      y_reverse();
    }
    if(y_counter == y_temp)
    {
```

```
    sequence = 0;
    }
    break;

    case 5:
    x_temp = x_tgt;
    y_temp = y_tgt;
    if(x_counter < x_temp)
    {
      Serial.println("+");
      x_forward();
    }
    if(x_counter > x_temp)
    {
      Serial.println("-");
      x_reverse();
    }
    if(x_counter == x_temp)
    {
    sequence = 6;
    }
    break;

    case 6:
    if(y_counter < y_temp)
    {
      Serial.println("++");
      y_forward();
    }
    if(y_counter > y_temp)
    {
```

```
      Serial.println("--");
      y_reverse();
    }
    if(y_counter == y_temp)
    {
    sequence = 7;
    }
    break;

    case 7:
    move_dwn();
    if(dwn_limit == LOW)
    {
      zero_speed();
      sequence = 8;
    }
    break;

    case 8:
    grip();
    sequence = 9;
    break;

    case 9:
    x_temp = x_dest;
    y_temp = y_dest;
    move_up();
    if(up_limit == LOW)
    {
      zero_speed();
      sequence = 10;
```

```
    }
    break;

    case 10:
    if(y_counter < y_temp)
    {
      Serial.println("++");
      y_forward();
    }
    if(y_counter > y_temp)
    {
      Serial.println("--");
      y_reverse();
    }
    if(y_counter == y_temp)
    {
    sequence = 11;
    }
    break;

    case 11:
    if(x_counter < x_temp)
    {
      Serial.println("+");
      x_forward();
    }
    if(x_counter > x_temp)
    {
      Serial.println("-");
      x_reverse();
    }
```

```
    if(x_counter == x_temp)
    {
    sequence = 12;
    }
    break;

    case 12:
    move_dwn();
    if(dwn_limit == LOW)
    {
      zero_speed();
      ungrip();
      sequence = 13;
    }
    break;

    case 13:
    move_up();
    if(up_limit == LOW)
    {
      zero_speed();
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("Finish!!");
      sequence = 0;
    }
    break;
  }
delay(1);
}
```

```
void x_target()
{
  if(x_counter < x_temp)
  {
analogWrite(motor1a, 0);
analogWrite(motor1b, 255);
analogWrite(motor2a, 0);
analogWrite(motor2b, 255);
  }
  if(x_counter > x_temp)
  {
analogWrite(motor1a, 255);
analogWrite(motor1b, 0);
analogWrite(motor2a, 255);
analogWrite(motor2b, 0);
  }
  if(x_counter == x_temp)
  {
analogWrite(motor1a, 0);
analogWrite(motor1b, 0);
analogWrite(motor2a, 0);
analogWrite(motor2b, 0);
  }
}

void y_target()
{
  if(y_counter < y_temp)
  {
analogWrite(motor3a, 0);
analogWrite(motor3b, 120); //asli 100
```

```
analogWrite(motor4a, 0);
analogWrite(motor4b, 120);
  }
  if(y_counter > y_temp)
  {
analogWrite(motor3a, 120);
analogWrite(motor3b, 0);
analogWrite(motor4a, 120);
analogWrite(motor4b, 0);
  }
  if(y_counter == y_temp)
  {
analogWrite(motor3a, 0);
analogWrite(motor3b, 0);
analogWrite(motor4a, 0);
analogWrite(motor4b, 0);
  }
}

void ungrip()
{
    for(s_pos = s_min_pos; s_pos <=
s_max_pos; s_pos++)
    {m_servo.write(s_pos); delay(s_speed);}
//time to reach pulse
    s_stat = 1;
}

void grip()
{
```

```
    for(s_pos = s_max_pos; s_pos >=
s_min_pos; s_pos--)
    {m_servo.write(s_pos);  delay(s_speed);}
//time to reach pulse
    s_stat = 0;
}

void zero_speed()
{
  digitalWrite(motor5a, LOW);
  digitalWrite(motor5b, LOW);
}

void stop_all()
{
digitalWrite(motor1a, LOW);
digitalWrite(motor1b, LOW);
digitalWrite(motor2a, LOW);
digitalWrite(motor2b, LOW);
digitalWrite(motor3a, LOW);
digitalWrite(motor3b, LOW);
digitalWrite(motor4a, LOW);
digitalWrite(motor4b, LOW);
digitalWrite(motor5a, LOW);
digitalWrite(motor5b, LOW);
  delay(500);
  lcd.clear();
}

void move_up()
{
```

```
analogWrite(motor5a, z_speed);
digitalWrite(motor5b, LOW);
}

void move_dwn()
{
digitalWrite(motor5a, LOW);
analogWrite(motor5b, z_speed);
}

void x_forward()
{
  x_reading = digitalRead(enc_x);
  if (x_reading != x_last_pulse)
  {x_last_bounce_time = millis();}
//timer internal dalam satuan milisecond
  if ((millis() - x_last_bounce_time) >
x_bounce_delay) //perbandingan waktu
sampling
  {
    if (x_reading != x_pulse)
    {
      x_pulse = x_reading;
      if (x_pulse == LOW)
      {
        x_counter++;
        EEPROM.write(data_x, x_counter);
      }
    }
  }
  x_last_pulse = x_reading;
```

```
  x_target();
}

void x_reverse()
{
  x_reading = digitalRead(enc_x);
  if (x_reading != x_last_pulse)
  {x_last_bounce_time = millis();}
  if ((millis() - x_last_bounce_time) >
x_bounce_delay)
  {
    if (x_reading != x_pulse)
    {
      x_pulse = x_reading;
      if (x_pulse == LOW)
      {
        x_counter--;
        EEPROM.write(data_x, x_counter);
      }
    }
  }
  x_last_pulse = x_reading;
  x_target();
}

void y_forward()
{
  int y_reading = digitalRead(enc_y);
  if (y_reading != y_last_pulse)
  {y_last_bounce_time = millis();}
```

```
  if ((millis() - y_last_bounce_time) >
y_bounce_delay)
  {
    if (y_reading != y_pulse)
    {
      y_pulse = y_reading;
      if (y_pulse == LOW)
      {
         y_counter++;
         EEPROM.write(data_y, y_counter);
      }
    }
  }
  y_last_pulse = y_reading;
  y_target();
}

void y_reverse()
{
  int y_reading = digitalRead(enc_y);
  if (y_reading != y_last_pulse)
  {y_last_bounce_time = millis();}
  if ((millis() - y_last_bounce_time) >
y_bounce_delay)
  {
    if (y_reading != y_pulse)
    {
      y_pulse = y_reading;
      if (y_pulse == LOW)
      {
         y_counter--;
```

```
          EEPROM.write(data_y, y_counter);
        }
      }
    }
    y_last_pulse = y_reading;
    y_target();
}
```

BIODATA PENULIS

Penulis dilahirkan di Lamongan pada tanggal 28 Oktober 1991 anak pertama dari dua bersaudara. Penulis menempuh pendidikan formal yaitu di SDN Plabuhanrejo, SMP Negeri 1 Mantup dan SMK NEGERI 1 Cerme-Gresik (lulus tahun 2010). Setelah itu penulis melanjutkan studi di Politeknik Perkapalan Negeri Surabaya (PPNS) dengan pilihan D3 Teknik Kelistrikan Kapal (lulus tahun 2013). Setelah lulus dari PPNS penulis melanjutkan studi lanjut S1 di Jurusan Teknis Sistem Perkapalan Fakultas Teknologi Kelautan Institut Teknologi Sepuluh Nopember Surabaya. Di Jurusan Teknik Sistem Perkapalan ini, penulis mengambil bidang studi Marine Electrical and Automation System (MEAS). Selama mengikuti perkuliahan, penulis sering mengikuti kegiatan baik seminar maupun training yang diadakan Himpunan Mahasiswa Teknik Sistem Perkalan (HIMASISKAL). Penulis juga mengikuti kegiatan sebagai instruktur untuk praktikum di laboratorium Marine Electrical and Automation System (MEAS).

Fajar Andik Cahyono
Jurusan Teknik Sistem Perkapalan FTK-ITS
fajarandik@ymail.com