PROTOTIPE SISTEM KEAMANAN PINTU PAGAR DENGAN MULTI-*INPUT* BERBASIS ARDUINO

DEDEN LUTHFI FERMANA

5215116399

Skripsi ini Ditulis untuk Memenuhi Sebagian Persyaratan dalam Memperoleh Gelar Sarjana

PROGRAM STUDI PENDIDIKAN TEKNIK ELEKTRONIKA

JURUSAN TEKNIK ELEKTRO

FAKULTAS TEKNIK

UNIVERSITAS NEGERI JAKARTA

2015

LEMBAR PENGESAHAN SKRIPSI

Nama Dosen	Tanda Tangan	Tanggal
Drs. Pitoyo Yuliatmojo, MT **(Dosen Pembimbing I)**	.....	13 Juli 2015
Muhammad Yusro, S.Pd, MT **(Dosen Pembimbing II)**	.....	13 Juli 2015

PENGESAHAN PANITIA UJIAN SIDANG

Nama Dosen	Tanda Tangan	Tanggal
Drs. Mufti Ma'sum, M.Pd **(Ketua Sidang)**	.....	13 Juli 2015
Dr. Baso Maruddani, MT **(Dosen Penguji)**	.....	13 Juli 2015
Efri Sandi, S.Pd, MT **(Dosen Ahli)**	.....	13 Juli 2015

Tanggal Lulus : 7 Juli 2015

LEMBAR PERNYATAAN

Saya yang bertanda tangan dibawah ini menyatakan bahwa:

1. Karya tulis skripsi saya dengan judul "Prototipe Sistem Keamanan Pintu Pagar Dengan Multi-*Input* Berbasis Arduino" adalah asli dan belum pernah diajukan untuk mendapatkan gelar akademik sarjana, baik di Universitas Negeri Jakarta maupun di perguruan tinggi lain.
2. Karya tulis ini adalah murni gagasan, rumusan dan penelitian saya sendiri dengan arahan dosen pembimbing.
3. Dalam karya tulis ini tidak terdapat karya atau pendapat yang telah ditulis atau dipublikasikan orang lain, kecuali secara tertulis dengan jelas dicantumkan sebagai acuan dalam naskah dengan disebutkan nama pengarang dan dicantumkan dalam daftar pustaka.
4. Pernyataan ini saya buat dengan sesungguhnya dan apabila di kemudian hari terdapat penyimpangan dan ketidakbenaran dalam pernyataan ini, maka saya bersedia menerima sanksi akademik berupa pencabutan gelar yang telah diperoleh karena karya tulis ini, serta sanksi lainnya sesuai dengan norma yang berlaku di Universitas Negeri Jakarta.

Jakarta, 29 Juni 2015
Yang membuat pernyataan

Deden Luthfi Kermana
NIM: 5215116399

KATA PENGANTAR

Puji syukur penulis panjatkan kehadirat Allah SWT yang telah memberikan rahmat serta karunia-Nya, sehingga penulis dapat menyelesaikan penyusunan skripsi yang berjudul "Prototipe Sistem Keamanan Pintu Pagar Dengan Multi-*Input* Berbasis Arduino".

Penyusunan skripsi ini dilakukan untuk memenuhi syarat mendapatkan gelar Sarjana Pendidikan Teknik Elektronika FT-UNJ. Penulis menyadari bahwa skripsi ini jauh dari sempurna sehingga penulis membutuhkan kritik dan saran yang sifatnya membangun untuk lebih menyempurnakan skripsi ini.

Dalam kesempatan ini penulis dengan tulus hati mengucapkan terima kasih kepada:

1. Drs. Wisnu Djatmiko, MT, Selaku Ketua Jurusan Teknik Elektro FT-UNJ.
2. Drs. Pitoyo Yuliatmojo, MT, Selaku Ketua Program Studi Pendidikan Teknik Elektronika FT-UNJ sekaligus Dosen Pembimbing I.
3. Muhammad Yusro, S.Pd, MT, Selaku Dosen Pembimbing II.
4. Dr. Moch. Sukardjo, M.Pd, selaku Pembimbing Akademik.
5. Kedua orang tua dan seluruh keluarga yang selalu mendoakan untuk kelancaran dan keberhasilan dalam studi.

Semoga segala bantuan yang telah diberikan senantiasa mendapatkan ridho dan balasan yang lebih baik dari Allah SWT. Akhir kata, penulis berharap semoga skripsi ini dapat bermanfaat bagi yang membacanya.

Penulis

Deden Luthfi Fermana
NIM: 5215116399

ABSTRAK

Deden Luthfi Fermana, Prototipe Sistem Keamanan Pintu Pagar Dengan Multi-Input Berbasis Arduino. Skripsi. Jakarta, Program Studi Pendidikan Teknik Elektronika, Jurusan Teknik Elektro, Fakultas Teknik Universitas Negeri Jakarta, 2015.

Tujuan dari penelitian ini adalah penulis mampu untuk mendesain, membuat dan menguji sistem keamanan pada pintu pagar dengan multi-input berbasis arduino yang dirancang dalam sebuah prototipe.

Sisem keamanan yang dibuat yaitu untuk membatasi akses masuk dengan multi-input meliputi kontrol jarak jauh dengan smartphone android melalui bluetooth, kontrol menggunakan keypad untuk input password dan kontrol menggunakan voice recognition untuk input perintah suara. Selain itu, sistem ini juga memiliki pendeteksi penyusup menggunakan sensor getaran dan sensor cahaya. Metode yang digunakan adalah research and development yang dibagi menjadi beberapa tahap, yaitu tahap penelitian dan pengumpulan informasi, tahap perencanaan, tahap pengembangan, tahap uji coba dan tahap perbaikan.

Berdasarkan penelitian yang telah dilakukan, sistem kemanan pada pintu pagar berjalan dengan baik mulai dari kendali android, password dan voice recognition. Sensor getaran dan sensor cahaya sebagai pendeteksi penyusup juga berjalan dengan baik.

Sistem keamanan pintu pagar dengan multi-input dapat memberikan keamanan penuh tanpa mempersulit akses terhadap pengguna. Dengan adanya pendeteksi penyusup, orang yang mencoba masuk dengan memanjat ataupun mendorong paksa pintu pagar dapat diketahui.

Kata Kunci: Prototipe, keamanan, multi-input, bluetooth, password, voice recognition.

ABSTRACT

Deden Luthfi Fermana, Prototype of Gate Security System With Multi-Input Based Arduino. Minithesis. Jakarta, Studies Program Electronics Engineering Education, Department of Electrical Engineering, Faculty of Engineering, State University of Jakarta, 2015.

The purpose of this research is the writer can design, create and test the security system of gate with multi-input based arduino that designed on a prototype.

Security system that made is for restricting access with multi-input includes control remotely with smartphone android via bluetooth, control with keypad for input password and control with voice recognition for input voice command. Moreover, this system also has an intruder detector uses vibration sensor and light sensor. The method that use in this research is research and development that devided into several stages includes research and information collecting, planning, development, test and revision.

Based on the research that has been done, the security system of the gate is working properly, began of android control, password control and voice recognition control. Vibration sensor and light sensor as intruder detector also working properly.

Security system of the gate can provide full safety without complicate access to the user. With intruder detector, people who tried to climb the gate or open forcibly can be known.

Keyword: Prototype, security, multi-input, bluetooth, password, voice recognition.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB 1
PENDAHULUAN

1.1 Latar Belakang Masalah

Teknologi keamanan pada sebuah bangunan saat ini semakin berkembang seiring dengan maraknya aksi kejahatan, terutama pada bagian terluar yaitu pintu pagar. Minimnya sistem keamanan pada pintu pagar ini memudahkan para penjahat masuk ke dalam rumah. Semakin banyak sistem keamanan yang digunakan pada pintu pagar maka akan meminimalisir terjadinya aksi penerobosan paksa pada pintu pagar. Sistem keamanan yang canggih tentunya diperlukan agar memberikan keamanan lebih dengan tanpa mempersulit penggunaannya bagi pengguna.

Dalam perkembangan teknologi saat ini banyak yang dapat diterapkan sebagai sistem keamanan, salah satunya kunci yang berupa *password.* Sistem *password* adalah sistem penguncian dimana untuk membuka kunci itu memerlukan masukkan yang berupa kumpulan huruf, angka ataupun kombinasi keduanya yang membentuk sebuah kode yang sebelumnya sudah diatur, jadi jika kode yang dimasukkan tidak sesuai yang telah ditentukan, kunci tidak akan terbuka. Sistem *password* memiliki kekurangan utama, yaitu pada pengguna yang terkadang lupa kode, sehingga dibutuhkan antisipasi untuk hal tersebut.

Selain sistem keamanan dengan teknologi *password*, ada teknologi saat ini yang dapat dimanfaatkan untuk sistem keamanan, yaitu teknologi pengenal suara *(voice recognition/speech recognition).* Teknologi pengenal suara adalah teknologi dimana sebuah alat yang dapat mendeteksi ucapan manusia dalam beberapa bahasa.

Ucapan tersebut dapat dijadikan sebuah perintah atau kata kunci untuk memberikan suatu aksi sehingga dapat dimanfaatkan sebagai sistem keamanan.

Sistem kendali jarak jauh juga dapat dimanfaatkan sebagai sistem keamanan pintu pagar, yaitu teknologi kendali jarak jauh dengan *smartphone* android yang memanfaatkan komunikasi *bluetooth*. Komunikasi *bluetooth* adalah komunikasi tanpa kabel *(wireless)* yang dapat mencapai jarak 10 meter. Sistem ini memanfaatkan *smartphone* android sebagai *remote control* untuk memberikan perintah ke modul *bluetooth* yang terhubung dengan mikrokontroler.

Dalam sistem keamanan pintu pagar juga membutuhkan sensor yang dapat mendeteksi getaran jika tetap ada yang mencoba membuka paksa. *Accelerometer* adalah sebuah alat yang mendeteksi perubahan posisi, sehingga sangat cocodapat dijadikan sebagai sensor getaran. Dalam penggunaannya, sistem keamanan dengan sensor getaran dihubungkan pada sebuah alarm dimana saat terdeteksi getaran, alarm akan berbunyi. Selain sensor getaran, sensor cahaya juga dibutuhkan untuk ditempatkan pada bagian atas agar ketika ada orang yang masuk dengan cara memanjat, dapat terdeteksi oleh sistem.

Untuk dapat menggabungkan beberapa sistem *input* dan sensor getar, membutuhkan mikrokontroler sebagai pengendali dan pemroses. Mikrokontroler yang digunakan yaitu arduino. Arduino adalah sebuah mikrokontroler yang bersifat *open source*, artinya pengguna bebas merancang sistem dengan arduino sesuai kebutuhannya. Arduino dirancang untuk memudahkan penggunaan elektronik dalam berbagai bidang. Dalam *hardware*nya, arduino memiliki Atmel AVR dan untuk pemrogramannya, arduino memiliki bahasa pemrograman sendiri.

Latar belakang tersebut yang menjadi dasar untuk peneliti membuat sebuah sistem keamanan pada pintu pagar dengan multi-*input* berbasis mikrokontroler arduino, yang diterapkan pada sebuah prototipe.

1.2 Identifikasi Masalah

Berdasarkan latar belakang masalah diatas maka dapat diidentifikasi beberapa masalah sebagai berikut:

1. Bagaimana membuat sistem kendali dengan aplikasi android berbasis arduino dan modul *bluetooth*.
2. Bagaimana membuat sistem kendali dengan *password* berbasis arduino dan *keypad*.
3. Bagaimana membuat sistem kendali dengan suara berbasis arduino dan *voice recognition module.*
4. Bagaimana membuat sistem keamanan dengan sensor getaran berbasis arduino dan *accelerometer*.
5. Bagaimana membuat sistem keamanan dengan sensor cahaya berbasis arduino dan *photo diode*.
6. Bagaimana cara membuat aplikasi berbasis *smarthphone* android (apk) dengan *app inventor.*

1.3 Pembatasan Masalah

Agar penyelesaian masalah yang dilakukan tidak meyimpang dari ruang lingkup yang ditentukan, maka akan dilakukan pembatasan masalah. Batasan masalah dalam penelitian ini adalah:

1. Perancangan sistem ini menggunakan arduino mega 2560 sebagai mikrokontroler utamanya dan arduino IDE untuk pemrogramannya.
2. Komunikasi antara arduino dan beberapa *input* meliputi modul *bluetooth*, *keypad*, *voice recognition*, sensor getaran (*accelerometer*) dan sensor cahaya (*photo diode*).
3. Arduino uno digunakan sebagai bagian dari sub sistem *voice recognition.*
4. Perancangan dan pembuatan aplikasi android menggunakan *app inventor*.

1.4 Perumusan Masalah

Berdasarkan latar belakang, identifikasi dan pembatasan masalah yang telah diuraikan, timbul beberapa pertanyaan yang merupakan rumusan masalah penelitaian, yaitu sebagai berikut:

1. Bagaimana merancang dan membuat sistem keamananan pintu pagar berbasis arduino.
2. Bagaimana mengkombinasikan antara sistem kendali *smartphone* android, *password* dan *voice recognition* dengan sistem keamanan sensor getaran (*accelerometer*) dan sensor cahaya (*photo diode*).

1.5 Tujuan Penelitian

Sesuai dengan masalah yang telah dirumuskan dan diidentifikasi, maka tujuan dari penelitian ini adalah:

1. Untuk merancang dan membuat sistem keamanan pintu pagar dengan multi-*input* berbasis arduino yang disimulasikan dalam suatu prototipe sehingga dapat mewakili pengaplikasian pada pintu pagar yang sebenarnya.
2. Untuk meningkatkan keamanan pada pintu pagar rumah sehingga dapat meminimalisir tingkat kriminalitas terutama pencurian.

1.6 Kegunaan Penelitian

Penelitian ini diharapkan dapat bermanfaat baik dari segi keilmuan dan dari segi masyarakat:

1. Dari segi keilmuan untuk mengaplikasikan ilmu Elektronika dan Instrumentasi yang telah dipelajari di Universitas Negeri Jakarta sebagai media yang baik.
2. Dari segi masyarakat atau mahasiswa Univesitas Negeri Jakarta, masyarakat dan mahasiswa dapat menerapkan sistem keamanan pada penelitian ini dalam kehidupan nyata sehingga masyarakat merasa lebih aman dan praktis

BAB II
KERANGKA TEORITIK, KERANGKA BERPIKIR DAN HIPOTESIS PENELITIAN

2.1 Kerangka Teoritik

2.1.1 Definisi Prototipe

Prototipe adalah contoh atau model pertama dari sesuatu atau disebut juga bentuk dasar dari sesuatu[1]. Prototipe dibuat sebagai contoh atau model dari sesuatu yang digunakan untuk mewakili bentuk dan mensimulasikan fungsi dari alat yang sebenarnya. Dengan adanya prototipe ini dapat dibuat alat sebenarnya dengan bentuk dan fungsi yang sama sesuai dengan yang direncanakan.

2.1.2 Definisi Keamanan Pintu Pagar

Keamanan pintu pagar terdiri dari tiga kata yang memiliki arti masing-masing. Pertama, definisi dari kata keamanan. Menurut kamus besar bahasa Indonesia yaitu keadaan yang bebas dari bahaya[2]. Selanjutnya definisi dari pintu pagar. Menurut kamus besar bahasa Indonesia, pintu berarti tempat untuk masuk dan keluar[3]. Sedangkan pagar berarti sesuatu yang digunakan untuk membatasi[4].

Dari definisi keempat kata tersebut dapat diartikan bahwa sistem keamanan pintu pagar adalah seperangkat alat yang tergabung dan membentuk satu kesatuan untuk memberikan rasa terbebas dari bahaya dengan membatasi akses masuk baik

[1] Kamus Umum Bahasa Indonesia, Perpustakaan Nasional Jakarta, 1994, hlm 1093.
[2] KBBI, *aman*, KBBI Daring (Dalam Jaringan/Online), diakses dari http://kbbi.web.id/aman pada tanggal 22 Maret 2015 pukul 01.30.
[3] Ibid http://kbbi.web.id/pintu.
[4] Ibid http://kbbi.web.id/pagar.

untuk manusia ataupun kendaraan. Sistem ini digunakan agar memberikan rasa aman dan nyaman kepada pengguna serta lebih memberikan kepraktisan dan keefisienan.

2.1.3 Komponen Pendukung Sistem Kendali

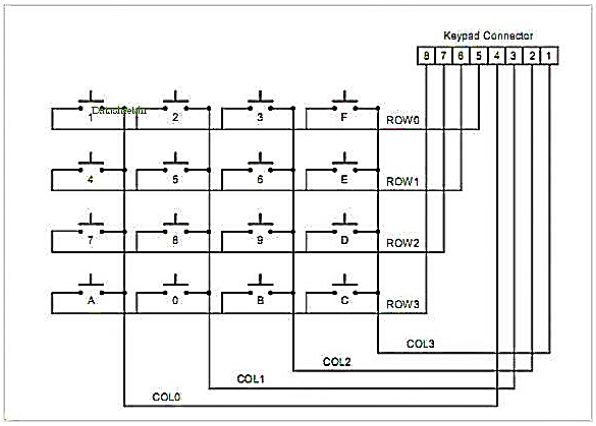
Sistem kendali terdiri dari kata sistem dan kendali. Definisi pertama dari sistem adalah susunan, seperangkat, atau sekumpulan sesuatu yang terhubung atau terkait sedemikian rupa sehingga membentuk sesuatu secara keseluruhan. Definisi kedua dari sistem adalah susunan komponen fisik yang terhubung atau terkait sedemikian rupa sehingga membentuk dan bertindak sebagai keseluruhan unit. Sedangkan kata kendali atau kontrol biasanya diartikan mengatur, mengarahkan, atau perintah. Dari kombinasi definisi kata sistem dan kontrol diatas, sistem kendali adalah suatu susunan komponen fisik yang terhubung atau terkait sedemikian rupa untuk memerintah, mengarahkan, atau mengatur sistem sendiri atau sistem lain[5]. Secara singkatnya sistem kendali berarti sekumpulan komponen yang saling terhubung membentuk suatu alat kerja sesuai dengan yang direncanakan.

2.1.3.1 *Keypad Matrix* 4x4

Keypad adalah perangkat elektronika yang membutuhkan interaksi manusia dengan kata lain membutuhkan aksi daripada manusia atau disebut dengan istilah HMI (*Human Machine Interface).* Konfigurasi *keypad* yang disusun secara matriks

[5] Joseph J. DiStefano, Allen R. Stubberud dan Ivan J. Williams, *Schaum's Outline of Feedback and Control Systems, 2nd Edition,* (Los Angeles: McGraw-Hill, 1990), hlm 1.

ini dimaksudkan agar menghemat penggunaan pin masukkan[6]. *Keypad* 4x4 ini terdiri dari 4 baris dan 4 kolom dimana disetiap persilangan baris dan kolom tersebut diletakkan saklar *push button*. Konfigurasi pin dari *keypad* matriks 4x4 digambarkan pada gambar 2.1 berikut.



Gambar 2.1 Konfigurasi Pin *Keypad Matrix* 4 x 4

Dalam penelitian ini, peneliti menggunakan *keypad* untuk masukkan yang berupa *password*. Kata *password* sendiri mempunyai definisi yaitu sebuah karakter yang hanya diketahui oleh badan tertentu (misalnya: pengguna) yang digunakan untuk mengautentikasi identitas pengguna sistem komputer dan/atau untuk mengizinkan akses ke sumber daya sistem[7].

2.1.3.2 *Voice Recognition*

Voice recognition atau pengenalan suara adalah teknologi dengan suara atau bicara, kata atau frasa yang diucapkan oleh manusia diubah menjadi sinyal-sinyal listrik, dan sinyal-sinyal ini diubah menjadi pola pengkodean yang telah

---

[6] Elektronika dasar, *Matrix Keypad 4x4 Untuk Mikrokontroler*, Komponen, diakses dari http://elektronika-dasar.web.id/artikel-elektronika/matrix-keypad-4x4-untuk-mikrokontroler/ pada tanggal 22 Maret 2015 pukul 02.30

[7] Meltem Sönmez Turan, dkk, *Recommendation for Password-Based Key Derivation*, (USA: NIST Special Publication 800-132, 2010), hlm 3.

ditetapkan[8]. Sinyal-sinyal suara yang dibaca didasarkan pada intonasinya, artinya suara semua orang dapat dideteksi baik laki-laki ataupun perempuan.

Teknologi ini sering dikenal juga dengan nama *speech recognition* atau *speaker recognition* yang berarti proses mengenali siapa yang berbicara atas dasar informasi yang diperoleh dari gelombang suara secara otomatis. Teknik ini akan memungkinkan untuk memverifikasi identitas orang yang mengakses sistem, yaitu akses kontrol dengan suara di berbagai layanan[9].

Secara tidak langsung, teknologi pengenal suara ini digunakan untuk sistem keamanan dengan cara membatasi akses terhadap suatu sistem guna mencegah hal-hal yang tidak diinginkan seperti pencurian, peretasan dan sebagainya. Selain untuk sistem keamanan, teknologi pengenal suara juga digunakan sebagai sarana untuk mempermudah kerja manusia dalam memberikan suatu perintah ke dalam suatu sistem tertentu.

2.1.3.3 *Bluetooth*

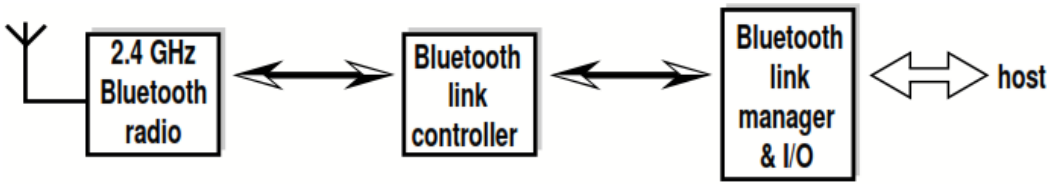
Bluetooth adalah sebuah teknologi komunikasi *wireless* (tanpa kabel) yang beroperasi dalam pita frekuensi 2,4 GHz *unlicensed* ISM *(Industrial, Scientific and Medical)* dengan menggunakan sebuah *frequency hopping tranceiver* yang mampu menyediakan layanan komunikasi data dan suara secara real-time antara *host-to-host bluetooth* dengan jarak jangkauan layanan yang terbatas[10]. Jarak yang dapat dijangkau oleh koneksi *bluetooth* yaitu tidak lebih dari 10 meter.

---

[8] Jim Baumann, *Voice Recognition*, Human Interface Technology Laboratory Wahington, diakses dari http://www.hitl.washington.edu/research/knowledge_base/virtual-worlds/EVE/I.D.2.d.VoiceRecognition.html pada tanggal 28 April 2015 pukul 20.00

[9] Chin-Hui Lee, Frank K. Soong dan Kuldip K. Paliwal, *Automatic Speech and Speaker Recognition, advanced topics,* (USA:Kluwer Academic Publisher, 1996), hlm 31.

[10] Bluetooth Architecture, *Specification of the Bluetooth System*, Bluetooth SIG, 2003, hlm 91.



Gambar 2.2 Blok Diagram Fungsional Sistem *Bluetooth*[11]

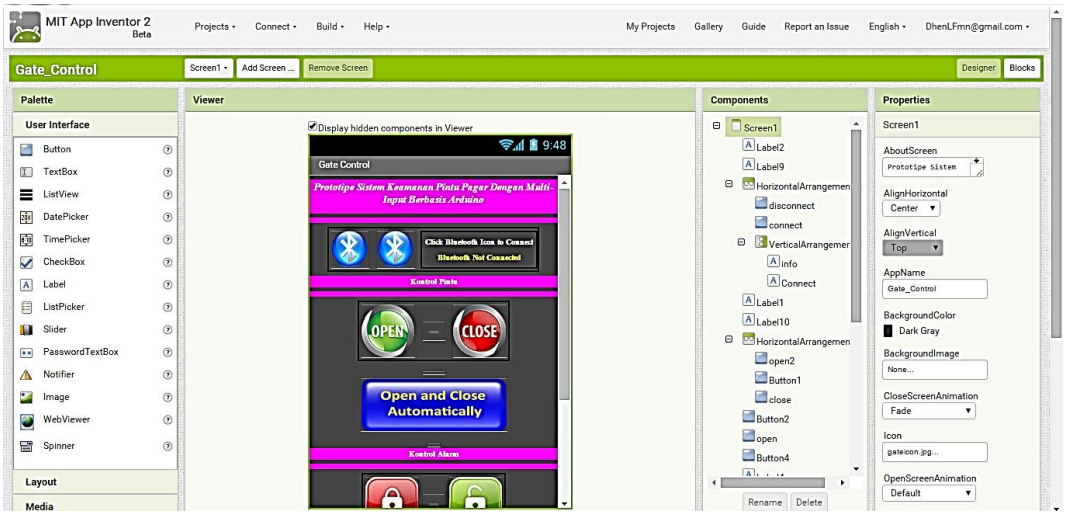
Jika dilihat pada gambar 2.2, sistem *bluetooth* terdiri dari unit radio *transceiver*, unit *link controller* dan unit *link manager*. *Link controller* bekerja sebagai penghubung perangkat keras radio *transceiver* ke *baseband processing* dan layer protokol fisik. Sedangkan *link manager* berfungsi untuk melakukan autentikasi dan konfigurasi terhadap *host*.

2.1.3.4 *App Inventor*

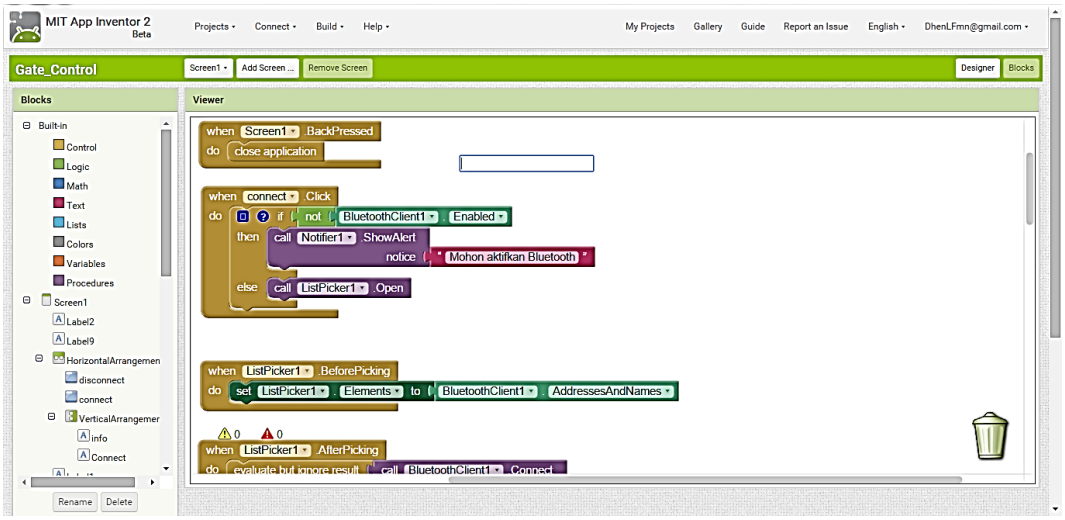
App inventor adalah alat *drag-and-drop* visual untuk membangun aplikasi seluler pada platform Android. Pengguna merancang antarmuka (tampilan visual) dari sebuah aplikasi berbasis web menggunakan GUI (*Graphical User Interface*), kemudian pengguna menentukan jalannya aplikasi dengan menyusun blok-blok seolah-olah pengguna sedang menyusun sebuah *puzzle*[12]. Aplikasi yang sekarang dikelola oleh MIT (*Massachusetts Institute of Technology*) ini memungkinkan pengguna baru untuk dapat membuat aplikasi yang berjalan pada sistem operasi

[11] Bluetooth Specification version 1.1, *Specification of the Bluetooth System*, Bluetooth SIG, 1999, hlm 41.
[12] David Wolber, dkk, *App Inventor, Create Your Own Android Apps,* (Canada: O'Reilly, 2011), hlm xv.

android dengan mudah. Tampilan pada aplikasi *app inventor* dapat dilihat pada gambar 2.3 dan 2.4 berikut ini.



Gambar 2.3 Tampilan Desain GUI *App Inventor*



Gambar 2.4 Tampilan Blok Program *App Inventor*

App inventor memiliki potensi besar untuk meningkatkan minat bagi perempuan dan orang lainnya yang kurang mengerti dalam hal ilmu pemrograman komputer. Siswa dapat belajar membuat aplikasi android sambil bermain dengan ponsel dan tablet mereka dan bahkan untuk pemula, dapat membuat aplikasi dalam

lingkungan yang menarik dan intuitif[13]. Hal tersebut menjadikan siapa saja dapat mempelajari cara membuat aplikasi android dengan mudah dan praktis tanpa harus mempelajari bahasa pemrograman yang kompleks.

2.1.4 Komponen Pendukung Sistem Keamanan

Sistem keamanan secara umum adalah sekumpulan komponen yang terhubung sedemikian rupa yang bertujuan untuk memberikan rasa aman dan bebas dari bahaya. Komponen-komponen tersebut berupa sensor dan tranduser.

Sensor didefinisikan sebagai perangkat non-unsur, biasanya didasarkan pada transduser, mampu mengkonversi kuantitas masukan fisik non-listrik menjadi kuantitas *output* listrik dan mengolahnya sesuai dengan algoritma yang diberikan, untuk memberikan keluaran yang sesuai untuk antarmuka ke proses sistem kontrol seperti komputer. Sedangkan transduser didefinisikan sebagai perangkat dasar dalam bidang pengukuran tertentu, yaitu konversi kuantitas masukan non-listrik fisik (besaran ukur) menjadi kuantitas *output* listrik[14]. Dengan kata lain, transduser adalah variabel keluaran dari sensor.

2.1.4.1 Sensor Getaran

Sensor getaran merupakan salah satu sensor yang dapat mengukur getaran suatu benda yang nantinya dimana data tersebut akan diproses untuk kepentingan percobaan ataupun di gunakan untuk mengantisipasi sebuah kemungkinan adanya

---

[13] Hal Abelson, *Teaching with App Inventor for Android*, ACM Digital Library, diakses dari http://dl.acm.org/citation.cfm?id=2157437 pada tanggal 25 April 2015 pukul 21.00.
[14] S. R. Ruocco, *Robot Sensors and Transducers*, (England: Halsted, 1987), hlm 2.

mara bahaya[15]. Seperti halnya dorongan paksa pada suatu benda dapat dideteksi oleh sensor getaran.

Sensor getaran pada dasarnya mendeteksi perubahan posisi yang cepat pada suatu benda baik perubahan posisi secara vertikal maupun horizontal dari posisi awal benda tersebut. Salah satu alat atau komponen yang dapat dijadikan sensor getaran adalah *accelerometer.*

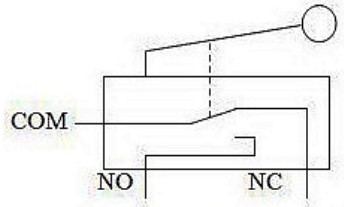
Accelerometer adalah sebuah transduser yang berfungsi untuk mengukur percepatan, mendeteksi dan mengukur getaran, ataupun untuk mengukur percepatan akibat gravitasi bumi[16]. Selain itu, *accelerometer* juga digunakan untuk mengukur percepatan ataupun getaran mesin, kendaraan ataupun benda lain yang tidak dipengaruhi gravitasi bumi.

2.1.4.2 *Limit Switch*

Limit switch adalah termasuk dalam kategori sensor mekanik, yaitu sensor yang digunakan untuk mengubah besaran mekanik menjadi besaran listrik. Pada sensor mekanik, keluaran sensor berubah sesuai perubahan gaya atau perubahan jarak (perpindahan), linier maupun rotasi[17]. Konstruksi dari *limit switch* dapat dilihat pada gambar 2.5 berikut ini.

---

[15] Komponen Elektronika, *Sensor Getaran*, Sensor, diakses dari http://www.komponenelektronika.biz/sensor-getaran.html pada tanggal 25 april 2015 pukul 22.00.
[16] Romi Wiryadinata, *Prinsip Kerja Sensor Accelerometer*, diakses dari http://wiryadinata.web.id/?p=22 pada tanggal 25 April 2015 pukul 22.30.
[17] Chogwang, *Pengertian dan Jenis Sensor*, diakses dari http://www.chogwang.com/2014/10/pengertian-dan-jenis-sensor.html pada tanggal 27 April 2015 pukul 22.00.



Gambar 2.5 Konstruksi *Limit Switch*

Prinsip kerja dari *limit switch* sama seperti saklar *push on*, yaitu hanya akan menghubung ketika katupnya ditekan pada batas penekanan tertentu yang telah ditentukan dan akan memutus ketika katup dilepas. *Limit switch* memiliki dua kontak, yaitu NO (*Normally Open*) dan NC (*Normally Close*).

2.1.4.3 Sensor Cahaya

Definisi dari sensor cahaya adalah komponen elektronika yang dapat memberikan perubahan besaran elektrik pada saat terjadi perubahan intensitas cahaya yang diterima oleh sensor cahaya tersebut[18]. Sensor cahaya terbagi menjadi dua tipe, pertama adalah sensor cahaya yang memberikan perubahan tegangan sesuai dengan intensitas cahaya yang diterima, sedangkan yang kedua adalah sensor cahaya yang memberikan perubahan resistansi sesuai dengan intensitas cahaya yang diterima.

Gambar 2.6 *Photo Diode*

[18] Zona Elektro, *Sensor Cahaya*, diakses dari http://zonaelektro.net/sensor-cahaya/ pada tanggal 27 April 2015 pukul 22.30.

Gambar 2.6 adalah sensor cahaya yang memberikan perubahan resistansi, yaitu *photo diode. Photo diode* adalah suatu dioda yang akan mengalami (Sensor Cahaya, 2015)resistansi pada terminal anoda dan katoda apabila terkena cahaya. Nilai resistansi pada *photo diode* akan semakin rendah jika intensitas cahaya yang diterima *photo diode* semakin tinggi.[19]

2.1.4.4 LCD (*Liquid Crystal Display*)

LCD (*Liquid Cristal Display*) adalah salah satu jenis *display* elektronik yang dibuat dengan teknologi CMOS *logic* yang bekerja dengan tidak menghasilkan cahaya tetapi memantulkan cahaya yang ada di sekelilingnya terhadap *front-lit* atau mentransmisikan cahaya dari *back-lit*[20].

Gambar *2*.7 LCD 16x2

Gambar 2.7 adalah LCD 16x2 dimana 16 menyatakan jumlah kolom dan 2 menyatakan jumlah baris. LCD ini sering digunakan untuk antarmuka antara pengguna dan mikrokontroler.

---

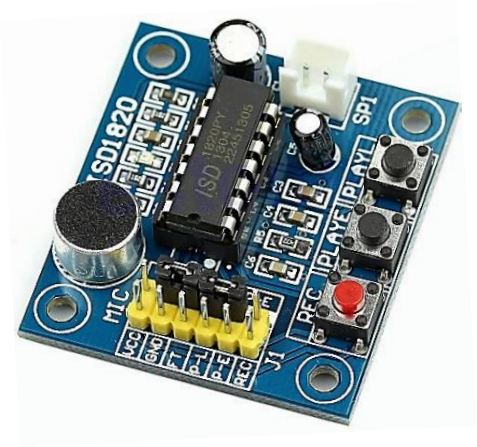
[19] Ibid.

[20] Elektronika dasar, *LCD (Liquid Crystal Display),* Komponen, diakses dari (DC Motors-Speed Controls-Servo Systems, 1977)pada tanggal 27 April 2015 pukul 22.40.

2.1.4.5 Modul ISD1820

Modul ISD1820 adalah modul *sound recording/playback* yang dapat merekam dan memainkan ulang rekaman audio dengan media penyimpanan terintegrasi (*non-volatile memory*) yang terintegrasi dalam *chip* tunggal ISD1820 ini[21]. Rekaman yang disimpan sebelumnya akan hilang ketika modul tersebut merekam audio yang lain.

Suara yang dapat direkam dengan modul ini berdurasi 8 hingga 12 detik, tergantung dari kualitas suara yang akan direkam, semakin tinggi kualitas suara yang akan direkam, maka semakin pendek durasi maksimal suara yang direkam.



Gambar 2.8 ISD1820 *Module*

Pada gambar 2.8, modul ISD1820 memiliki 3 *push button* yang masing-masing berfungsi untuk merekam, memainkan keseluruhan dan memainkan hanya ketika *button* ditekan.

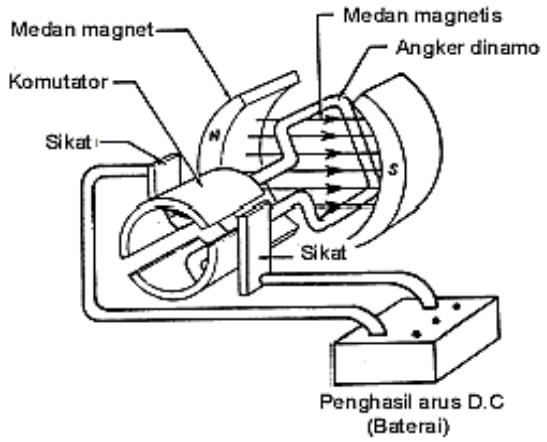
[21] Vcc2gnd, *Modul ISD1820*, diakses dari http://www.vcc2gnd.com/2014/01/IS1820-Sound-Module.html pada tanggal 27 April 2015 pukul 23.00.

2.1.5 Komponen Pendukung Sistem Mekanis

Mekanis memiliki arti secara mesin atau menurut cara kerja mesin[22]. Maka sistem mekanis dapat diartikan sebagai sekumpulan komponen yang saling terhubung dan membentuk suatu proses kerja dengan prinsip cara kerja mesin.

2.1.5.1 Motor DC

Motor DC adalah salah satu mesin pertama yang ditemukan untuk mengkonversi energi listrik menjadi energi gerak[23]. Motor DC menggunakan tegangan DC untuk menggerakkannya, jika polaritas dari tegangan yang diberikan pada motor DC dibalik, maka arah putarannya pun terbalik.



Gambar 2.9 Cara Kerja Motor DC Sederhana

Cara kerja motor DC secara sederhana bisa dilihat pada gambar 2.9. Ketika arus DC dari baterai ataupun catu daya mengalir melewati sikat dan menuju komutator, maka yang disebut angker dinamo akan berputar karena adanya medan magnet yang dihasilkan oleh kutub utara dan kutub selatan magnet.

---

[22] Kamus Umum Bahasa Indonesia, Perpustakaan Nasional Jakarta, 1994, hlm 882.
[23] Electro-Craft Corporation, *DC Motors-Speed Controls-Servo Systems*, (USA: Pergamon Press, 1977), hlm 2-1.

Dari gerak motor DC yang hanya berputar, dapat dikonversi menjadi berbagai macam gerak seperti gerak secara vertikal maupun horizontal dengan tambahan komponen *gear*, *belt*, ulir dan sebagainya.

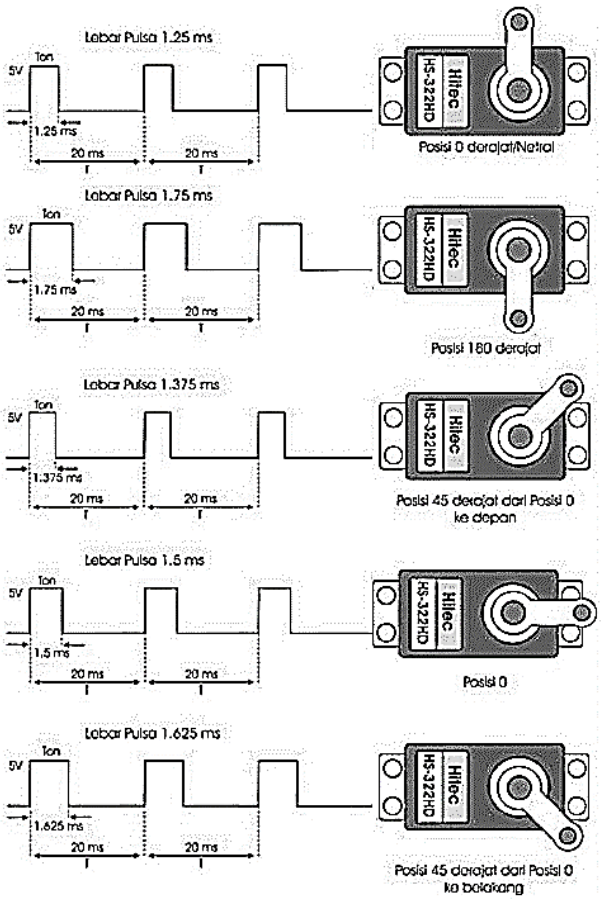
2.1.5.2 Motor Servo

Motor servo adalah motor DC yang dilengkapi rangkaian kendali dengan sistem *closed feedback* yang terintegrasi dalam motor tersebut[24]. Motor servo tersusun dari sebuah motor DC, *gearbox*, varistor/potensiometer dan rangkaian kontrol.

Motor servo terbagi menjadi jenis yaitu motor servo standar 180^o dan motor servo kontinyu. Servo 180^o dapat berputar ke dua arah (CW dan CCW) dengan defleksi sudut putar masing-masing 90^o. Sedangkan servo kontinyu dapat berputar ke dua arah (CW dan CCW) tetapi tanpa batasan defleksi sudut putar atau dapat berputar secara kontinyu.

Motor servo dikendalikan dengan memberikan sinyal modulasi lebar pulsa (PWM). Lebar pulsa yang diberikan akan menentukan posisi sudut putaran dari poros motor servo. Apabila motor servo diberikan pulsa dengan besar 1.5 ms mencapai gerakan 90°, maka bila kita berikan pulsa kurang dari 1.5 ms maka posisi mendekati 0° dan bila kita berikan pulsa lebih dari 1.5 ms maka posisi mendekati 180°.

[24] Elektronika dasar, *Motor Servo*, Teori Elektronika, diakses dari http://elektronika-dasar.web.id/teoril-elektronika/motor-servo/ pada tanggal 28 Maret 2015 pukul 00.30.



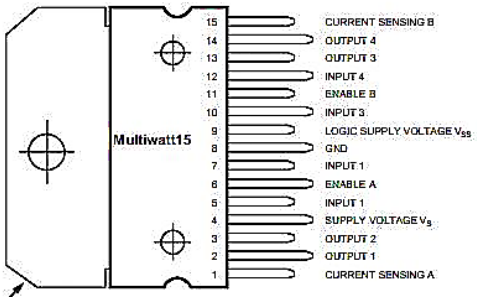
Gambar 2.10 Posisi Motor Servo Berdasarkan Lebar Pulsa

Pada gambar 2.10 menunjukkan bahwa ketika posisi motor servo pada 0^o, lebar pulsa yang diberikan kurang dari 1,5ms yaitu 1,25ms, ketika posisi motor servo pada 90^o, lebar pulsa yang diberikan adalah 1,5ms dan ketika posisi motor servo pada 180^o, lebar pulsa yang diberikan lebih dari 1,5ms yaitu 1,75ms.

2.1.5.3 Driver L298

L298 adalah jenis IC *driver* motor yang dapat mengendalikan arah putaran dan kecepatan motor DC ataupun motor stepper. *Driver* ini mampu mengeluarkan

output tegangan untuk motor DC dan motor stepper sebesar 50 volt[25]. IC L298 tersusun dari transistor-transistor logik (TTL) dengan gerbang *nand* dimana hal tersebut memudahkan pengguna untuk menentukkan arah putaran dari motor DC maupun motor stepper. IC L298 dapat mengendalikan 2 motor DC sekaligus tetapi hanya dapat mengendalikan 1 motor stepper dengan arus 2 Ampere.



Gambar 2.11 IC L298

Pada gambar 2.11 merupakan IC L298 yang ini memiliki total 15 pin. Diantaranya 4 pin *input*, 4 pin *output*, 2 pin *input* tegangan (5Volt dan 12 Volt), 2 pin PWM dan 3 pin lainnya dihubungkan dengan ground.

2.1.6 Arduino

Arduino adalah sebuah *platform* dari *physical computing* yang bersifat *open source* yang didasarkan atas papan masukan/keluaran (I/O) sederhana dan *development environment* yang mengimplementasikan bahasa pengolahan [26] . Arduino bisa digunakan untuk mengembangkan objek interaktif yang berdiri sendiri atau bisa juga dihubungkan dengan *software* di komputer. Dewasa ini arduino sangat banyak diminati oleh pemula dalam mempelajari tentang elektronika dan robotika karena bahasa yang digunakan dalam pemrograman arduino tergolong

[25] Kedairobot, *L298 Motor Driver*, diakses dari http://kedairobot.com/components/35-l298-motor-driver.html pada tanggal 27 April 2015 pukul 23.30.
[26] Massimo Banzi, *Getting Started with Arduino, First Edition,* (USA: O'Reilly, 2008), hlm 1.

mudah, yaitu menggunakan bahasas C dan dipermudah lagi dengan penyederhanaan menggunaka bantuan *libraries.* Arduino memiliki berbagai macam tipe dengan spek dan karakteristik yang berbeda-beda.

Dalam penelitian ini, arduino yang digunakan ada dua tipe, yaitu tipe uno dan mega 2560. Arduino uno digunakan khusus untuk mengolah masukan dari modul *voice recognition* karena modul tersebut hanya bisa didukung oleh arduino tipe uno. Sedangkan arduino tipe mega 2560 digunakan untuk sistem utamanya. Penggunaan arduino tipe mega 2560 dikarenakan keseluruhan sistem membutuhkan banyak pin masukan/keluaran sehingga peneliti menggunakan dua arduino dengan tipe yang berbeda.

Arduino memiliki perbedaan dibandingkan dengan produk-produk lain yang ada di pasaran. Perbedaan tersebut sebagai berikut:

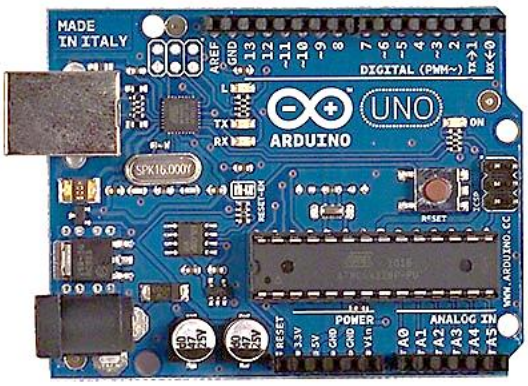
1. IDE arduino berupa multiplatform, artinya bisa dijalankan di berbagai sistem operasi seperti Windows, Macintosh, dan Linux.
2. Arduino didasarkan pada IDE prosesing yang artinya mudah digunakan oleh seniman dan perancang.
3. Pemrograman arduino menggunakan kabel USB sehingga sangat mudah digunakan dengan komputer-komputer modern saat ini.
4. *Software* dan *hardware* arduino bersifat *open-source* yang artinya pengguna dapat mengunduh rangkaian, *libraries* dan segala bentuk *software* yang berhubungan dengan arduino tanpa harus membayar.
5. Harga dari *hardware* arduino murah.
6. Terdapat banyak komunitas dari pengguna-pengguna arduino sehingga bisa membantu pengguna lain yang mendapat kesulitan.

7. Arduino dikembangkan dalam pendidikan sehingga pengguna baru bisa mengerti lebih cepat[27].

2.1.6.1 Arduino Uno

Arduino uno adalah papan mikrokontroler berbasis ATmega328 yang memiliki 14 pin digital I/O dimana 6 pin diantaranya dapat digunakan sebagai *output* PWM, sepasang pin komunikasi serial, 6 pin masukan analog, *clock speed* 16 MHz, koneksi USB, *jack* listrik, *header* ICSP dan tombol *reset*[28].

Arduino uno adalah arduino yang paling umum digunakan, terutama untuk pemula karena ukurannya tidak terlalu besar dan tidak terlalu kecil dan harganya pun relatif murah.



Gambar 2.12 Arduino Uno

Gambar 2.12 merupakan tampilan dari arduino uno, yang pada penelitian ini, peneliti menggunakan arduino uno sebagai bagian dari salah satu sistem *input*, yaitu pada sistem *voice recognition* dimana arduino uno ini digunakan untuk

---

[27] Ibid, hlm 2.
[28] Arduino, *Arduino Uno*, diakses dari http://www.arduino.cc/en/Main/ArduinoBoardUno pada tanggal 27 April 2015 pukul 20.00.

mendukung modul *voice recognition*. Hal tersebut dikarenakan modul *voice recognition* yang dipakai hanya dapat diprogram menggunakan arduino uno.

Karakteristik dari arduino uno sendiri dapat dilihat pada tabel 2.1 berikut ini.

Tabel 2.1 Karakteristik Arduino Uno[29]

No.	*Name*	*Value*
1	*Microcontroller*	Atmega328
2	*Operating Voltage*	5V
3	*Input Voltage (recommended)*	7-12V
4	*Input Voltage (limits)*	6-20V
5	Digital Pin I/O	14 (*of which* 6 *provide PWM output*)
6	Pin Analog *Input*	6
7	DC *Current* per Pin I/O	40 mA
8	DC *Current for* 3.3V *Pin*	50 mA
9	*Flash Memory*	32 KB *of which* 0,5 KB *used by bootloader*
10	SRAM	2 KB
11	EEPROM	1 KB
12	*Clock Speed*	16 MHz

2.1.6.2 Arduino Mega 2560

Arduino mega 2560 adalah papan mikrokontroler berbasis ATMega2560 dengan 54 pin digital I/O dimana 15 pin diantaranya bisa digunakan sebagai *output* PWM, 16 pin analog *input*, empat pasang komunikasi serial, *clock speed* 16 MHz, koneksi USB, *jack* listrik, *header* ICSP dan tombol *reset*[30].

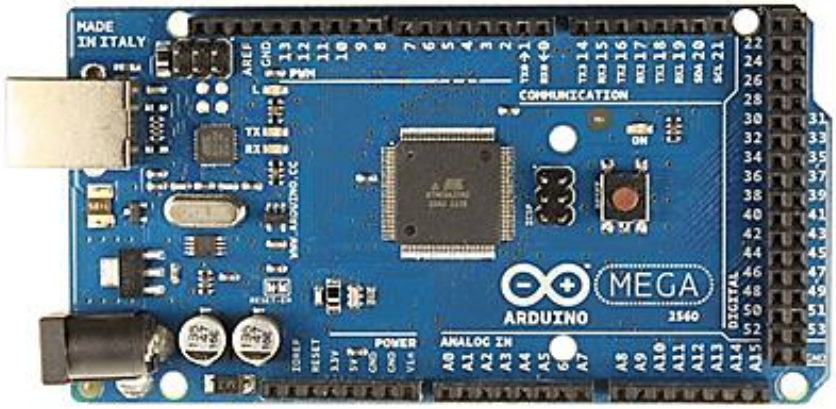
---

[29] Ibid

[30] Arduino, *Arduino Mega 2560*, diakses dari http://www.arduino.cc/en/Main/ArduinoBoardMega2560 pada tanggal 27 April 2015 pukul 20.15.

Arduino mega 2560 pada umumnya memiliki ukuran yang lebih besar dibandingkan dengan arduino tipe lainnya dikarenakan arduino ini memiliki lebih banyak pin dibandingkan dengan arduino tipe lain.

Pada penelitian ini, peneliti menggunakan arduino mega 2560 sebagai mikrokontroler utama dalam sistem, karena arduino tipe ini memiliki pin I/O lebih banyak, sehingga sangat mendukung untuk perancangan sistem yang memiliki banyak sub sistem. Bentuk dari arduino mega 2560 dapat dilihat pada gambar 2.13 berikut ini.



Gambar 2.13 Arduino Mega 2560

Selain dari bentuk dan tampilannya, ada juga beberapa karakteristik dari arduino mega 2560 untuk membedakannya dari arduino lain. Karakteristik tersebut bisa dilihat pada tabel 2.2 berikut ini.

Tabel 2.2 Karakteristik Arduino Mega 2560[31]

No.	*Name*	*Value*
1	*Microcontroller*	ATmega2560
2	*Operating Voltage*	5V
3	*Input Voltage (recommended)*	7-12V
4	*Input Voltage (limits)*	6-20V

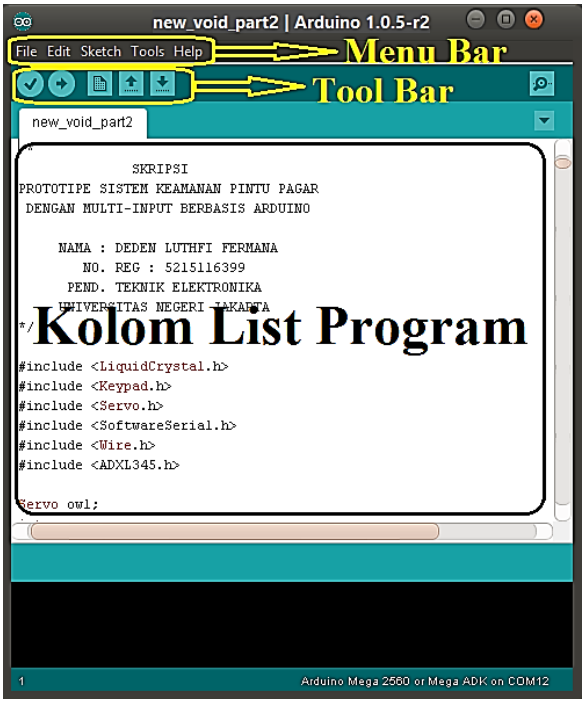
[31] Ibid

5	Digital Pin I/O	54 (*of which 15 provide PWM output*)
6	Pin Analog *Input*	16
7	DC *Current* per Pin I/O	40 mA
8	DC *Current for* 3.3V *Pin*	50 mA
9	*Flash Memory*	256 KB *of which* 8 KB *used by bootloader*
10	SRAM	8 KB
11	EEPROM	4 KB
12	*Clock Speed*	17 MHz

2.1.7 Arduino IDE

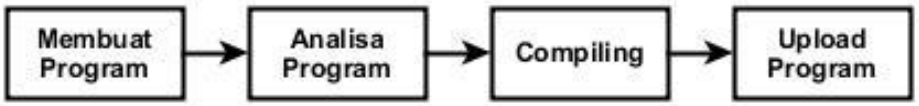
Arduino IDE (*Integrated Development Environment*) adalah sebuah *software* untuk menulis program, mengompilasi menjadi kode biner dan meng-*upload* ke dalam *memory* mikrokontroler[32]. Bahasa pemrograman yang digunakan adalah bahasa C. *Software* ini bersifat *open-source* dan menyediakan berbagai macam *library* secara gratis untuk pemrograman suatu komponen sehingga memudahkan pengguna dalam pemakaiannya.

[32] Muhammad Syahwil, *Panduan Mudah Simulasi & Praktek Mikrokontroler Arduino,* (Yogyakarta: Andi Offset, 2013), hlm 39.



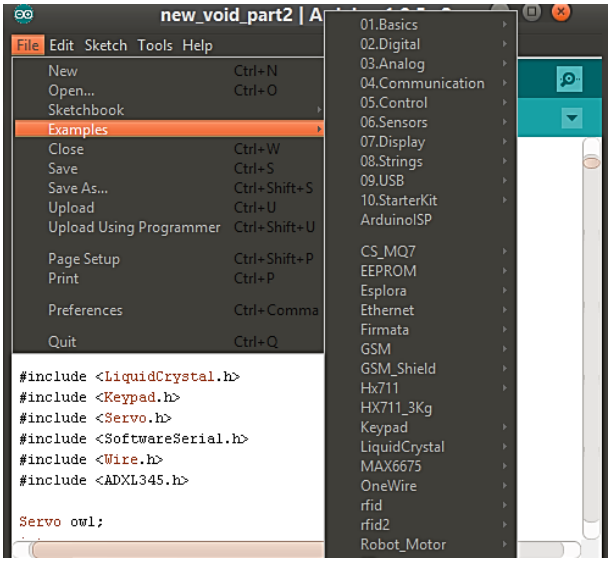
Gambar 2.14 Antarmuka Arduino IDE

Pada gambar 2.14 merupakan tampilan antarmuka dari *software* arduino IDE. Antarmuka yang ditampilkan sederhana serta *tool bar* dan menu-menunya mudah untuk dipahami terutama bagi pemula. Gambar 2.15 merupakan alur dalam pemrograman menggunakan arduino IDE dari mulai membuat program sampai memasukkan program ke arduino.



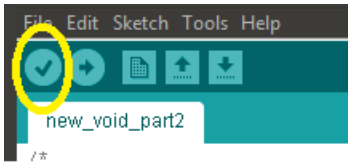
Gambar 2.15 Alur Pemrograman Arduino IDE

1. Membuat program pada arduino IDE yaitu menggunakan bahasa C. Program dapat dibuat dari awal ataupun mengambil contoh dari *library* yang sudah tersedia di dalamnya dengan cara mengklik menu *"File"* kemudian pilih *"Examples"*. Setelah itu akan muncul pilihan program yang sudah tersedia. Lihat pada gambar 2.16 berikut ini.



Gambar 2.16 *Libraries* Arduino IDE

2. Analisa program artinya mengecek dari awal program yang sudah dibuat atau diedit, apakah logikanya sudah benar atau belum.
3. *Compiling* dilakukan setelah program selesai dibuat. Caranya yaitu dengan mengklik simbol *checklist* pada *tool bar*. Lihat gambar 2.17 berikut ini.

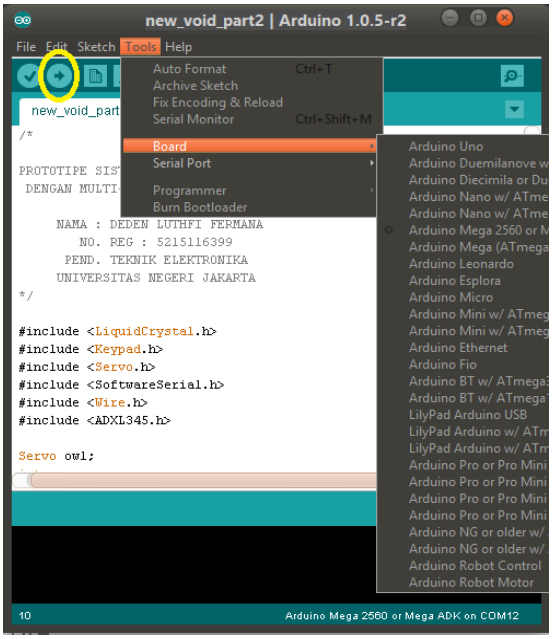


Gambar 2.17 *Compiling* Program Arduino

Jika setelah *compiling* muncul pesan *error*, itu berarti ada kesalahan pada program dan program harus diperbaiki sampai tidak ada pesan *error*.

4. Untuk meng*upload* program, hubungkan arduino dengan komputer sampai terdeteksi di komputer, kemudian pilih tipe arduino pada arduino IDE sesuai dengan arduino yang dihubungkan dengan komputer dengan cara mengklik *"Tools"* pada menu *bar* kemudian pilih *"Board"*. Setelah

itu, klik simbol panah di sebelah simbol *checklist* pada *tool bar*. Lihat gambar 2.18 berikut ini.

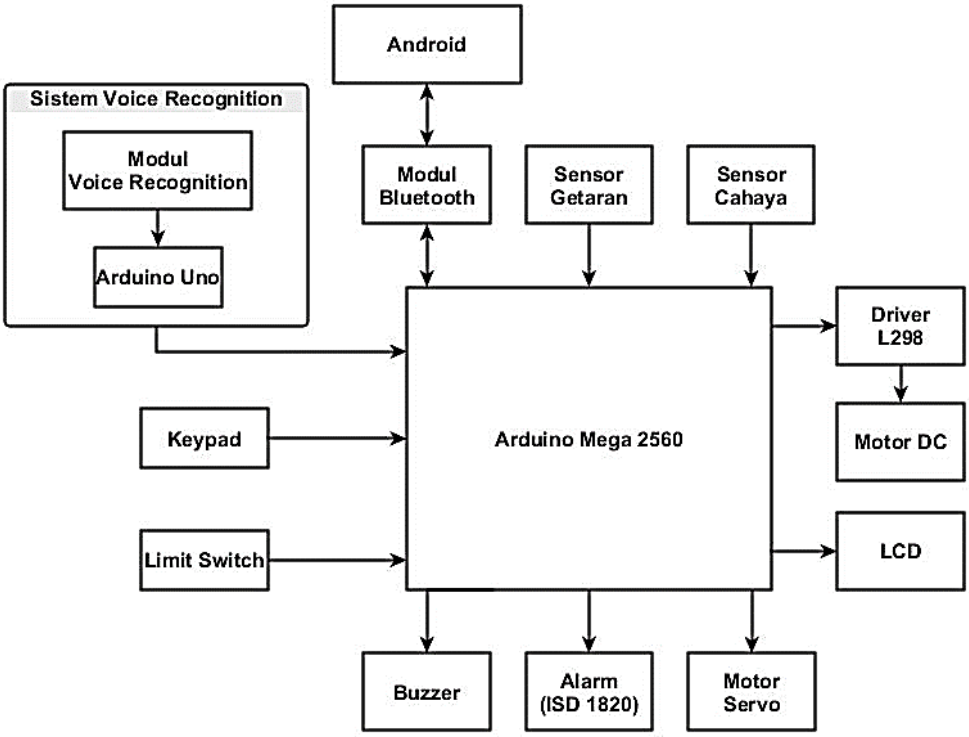


Gambar 2.18 *Upload* Program Arduino

2.2 Kerangka Berpikir

2.2.1 Blok Diagram Prototipe Sistem Keamanan Pintu Pagar dengan Multi-*Input* Berbasis Arduino

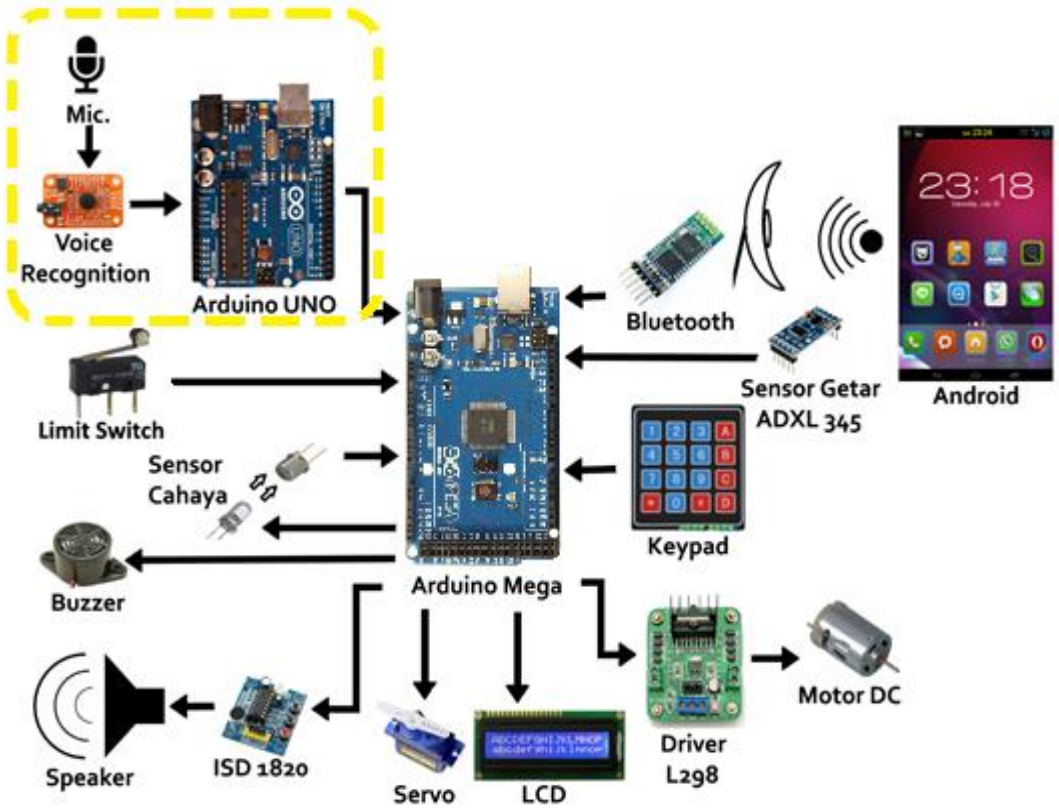
Sebelum membuat prototipe sistem keamanan pintu pagar dengan multi-*input* berbasis arduino, terlebih dahulu harus merancang susunan keseluruhan sistem. Rancangan tersebut dapat dilihat pada gambar 2.19 dan 2.20.



Gambar 2.19 Blok Diagram Prototipe Sistem Keamanan Pintu Pagar dengan Multi-*Input* Berbasis Arduino

Gambar 2.19 merupakan blok diagram dari prototipe sistem keamanan pintu pagar dengan multi-*input* berbasis arduino. Prototipe pintu pagar ini memiliki *input* dan *output* yang banyak, sehingga arduino mega 2560 digunakan sebagai mikrokontroler utamanya, karena memiliki lebih banyak pin I/O dibandingkan arduino tipe lainnya.

Selain blok diagram, rancangan sistem dari prototipe yang akan dibuat juga digambarkan dengan blok komponen pada gambar 2.20 berikut ini.



Gambar 2.20 Blok Komponen Prototipe Sistem Keamanan Pintu Pagar dengan Multi-*Input* Berbasis Arduino

Pada gambar 2.20 merupakan blok komponen pada prototipe pintu pagar dimana arah panah pada setiap komponen menunjukkan fungsinya. Arah panah yang menuju arduino mega 2560 adalah sebagai *input* sedangkan arah panah dari arduino mega 2560 adalah sebagai *output*.

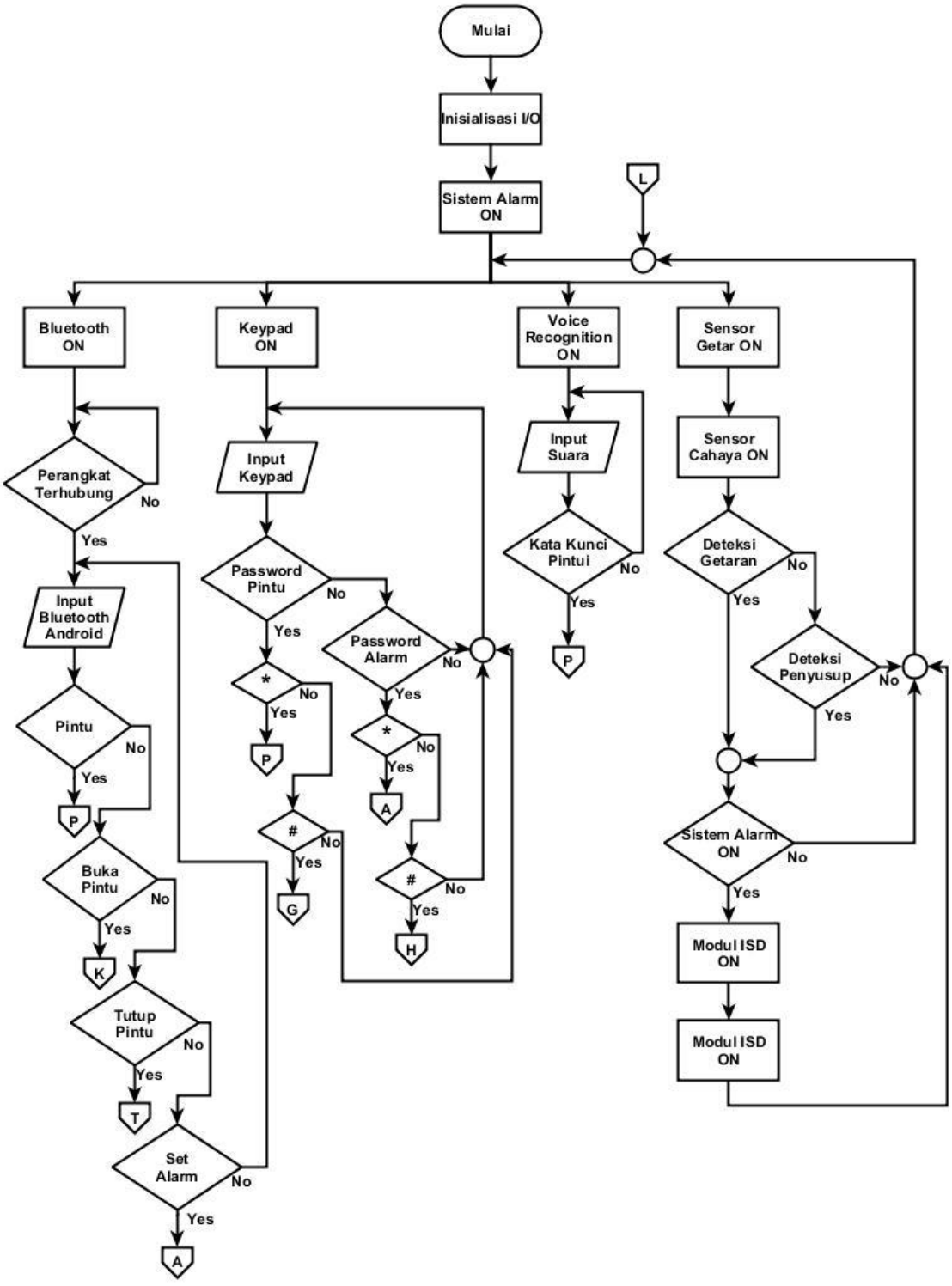
Masukan dari sistem keamanan ini terdiri dari modul *bluetooth* yang menerima perintah dari *smartphone* android, *keypad* yang masukannya berupa *password, voice recognition module* yang masukannya berupa perintah suara, *accelerometer* adxl345 yang digunakan sebagai sensor getaran, sensor cahaya *photo diode* yang digunakan sebagai pendeteksi manusia, kendaraan atau yang lainnya di jalur pintu ketika pintu pagar ditutup agar tidak menabrak juga sebagai pendeteksi manusia yang memanjat ke atas tembok pintu pagar dan *limit switch* sebagai sensor mekanik pada pintu pagar. Keluaran dari sistem keamanan ini terdiri

dari motor DC sebagai sistem mekanis buka dan tutup pada pintu pintu pagar, motor servo sebagai sistem pengunci pintu, LCD 16x2 sebagai indikator berupa teks, *speaker* sebagai indikator alarm ketika sensor getaran mendeteksi dorongan paksa pada pintu pagar dan *photo diode* mendeteksi adanya yang memanjat pada tembok pintu pagar serta *buzzer* sebagai indikator dari *keypad* ketika ditekan.

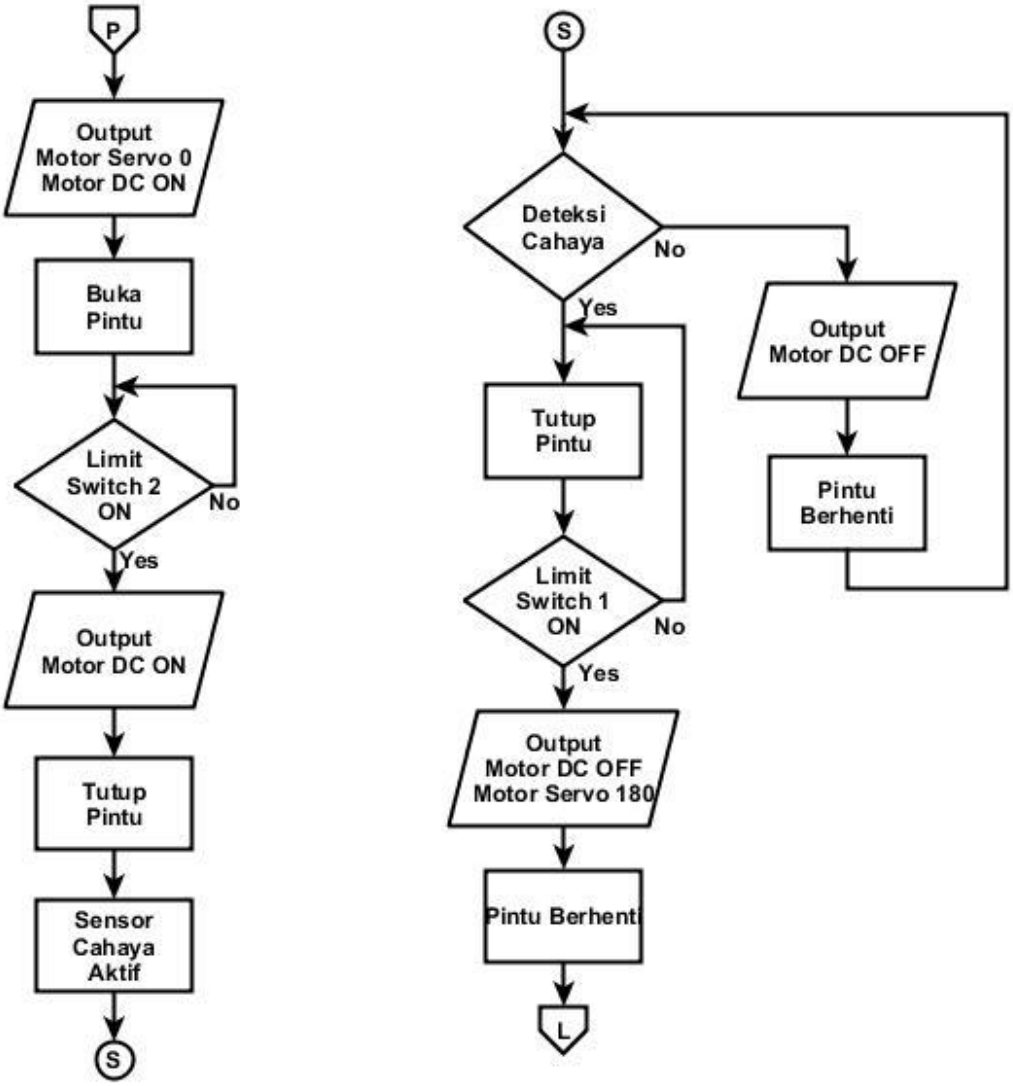
Adapun arduino uno yang digunakan dalam sistem *voice recognition* sebagai pendukung sistem tersebut, karena modul *voice recognition* yang dipakai dalam penelitian ini tidak dapat diproses menggunakan mikrokontroler selain arduino uno.

Setelah blok diagram dan blok komponen dirancang, selanjutnya dibuat alur kerja dari keseluruhan sistem pada prototipe pntu pagar ini yang dijelaskan dalam bentuk *flowchart* pada bahasan selanjutnya.

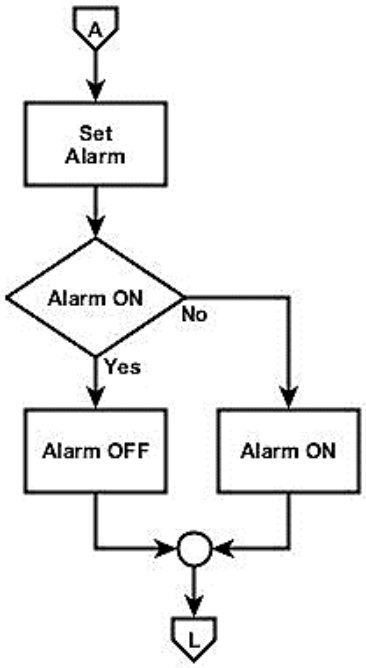
2.2.2 Flowchart Prototipe Sistem Keamanan Pintu Pagar dengan Multi-*Input* Berbasis Arduino



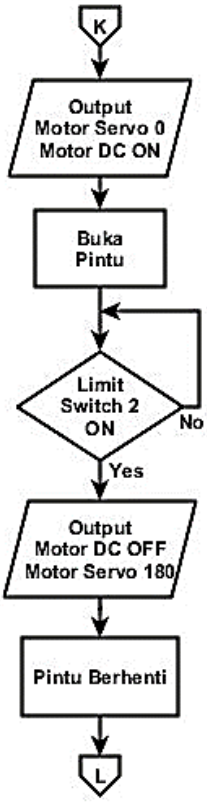
Gambar 2.21 *Flowchart* Prototipe 1



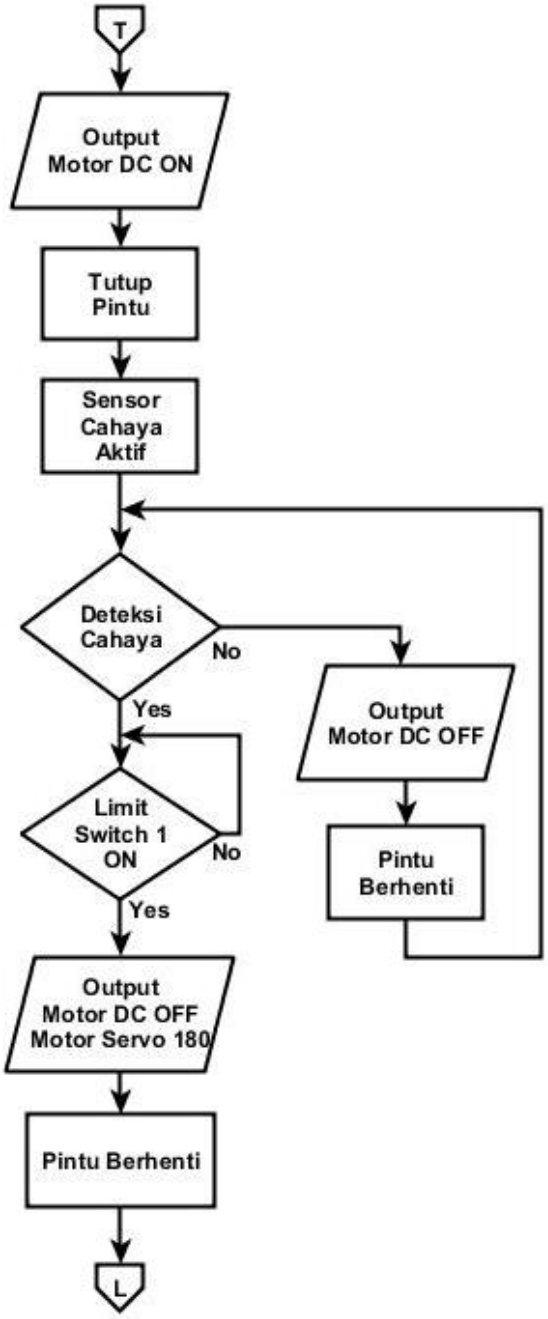
Gambar 2.22 *Flowchart* Prototipe 2



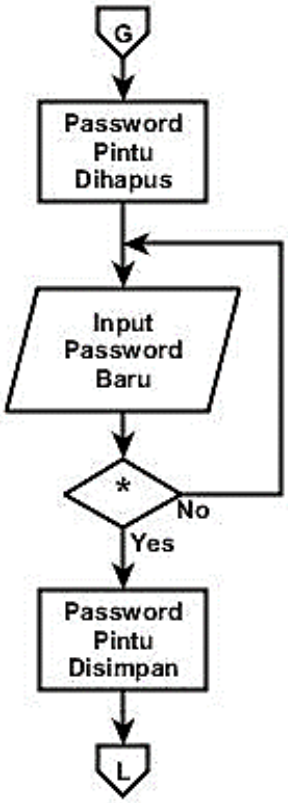
Gambar 2.23 *Flowchart* Prototipe 3



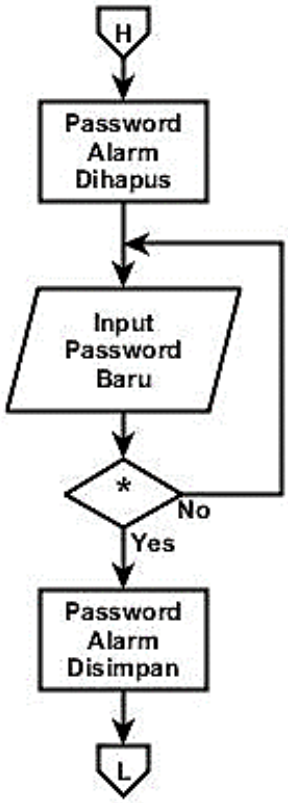
Gambar 2.24 *Flowchart* Prototipe 4



Gambar 2.25 *Flowchart* Prototipe 5



Gambar 2.26 *Flowchart* Prototipe 6



Gambar 2.27 *Flowchart* Prototipe 7

Pada gambar 2.21 *flowchart* prototipe 1, hal yang pertama dilakukan prototipe ketika prototipe pertama kali dijalankan adalah menginisialisasi pin I/O untuk menentukan masukan dan keluaran dari keseluruhan sistem. Setelah inisialisasi selesai, sistem akan mengaktifkan sistem alarm untuk kondisi awal dan mengaktifkan modul *bluetooth*, *keypad*, *voice recognition* dan sensor getar.

Pada sistem *bluetooth*, setelah modul *bluetooth* aktif, pengguna terlebih dahulu harus menghubungkan perangkat *smartphone* androidnya untuk mengirim perintah. Setelah terhubung, jika tombol "*Open and Close Automatically*" ditekan, maka sistem akan membuka pintu dan menutup pintu secara otomatis ketika pintu telah terbuka total. Jika tombol "*Open*" ditekan, maka sistem akan membuka pintu. Jika tombol "Close" ditekan, maka sistem akan menutup pintu. Jika tombol "ON" ditekan, maka sistem akan mengaktifkan sistem alarm. Jika tombol "OFF" ditekan, maka sistem akan menonaktifkan sistem alarm.

Pada *keypad*, jika *password* pintu yang dimasukkan (222222), maka setelahnya ada pilihan untuk menekan "*" atau "#". Jika setelahnya tombol "*" yang ditekan, maka sistem akan membuka pintu dan menutup pintu otomatis ketika pintu telah terbuka. Sedangkan jika setelahnya tombol "#" yang ditekan, maka sistem akan masuk menu pilihan untuk mengganti *password* pintu. Jika *password* keamanan yang dimasukkan (333333), maka setelahnya ada pilihan untuk menekan "*" atau "#". Jika setelahnya tombol "*" yang ditekan, maka sistem akan mengaktifkan sistem alarm ketika kondisi awalnya nonaktif dan sebaliknya, ketika kondisi awalnya aktif, maka sistem akan menonaktifkannya. Sedangkan jika setelahnya tombol "#" yang ditekan, maka sistem akan masuk pilihan untuk mengganti *password* keamanan. Jika *password* yang dimasukkan salah, maka

ketika ditekan tombol “*” ataupun “#” akan ada peringatan dan harus *input* ulang *password*.

Pada *voice recognition*, jika suara atau ucapannya adalah “*open*”, maka sistem akan membuka pintu dan menutup pintu otomatis ketika pintu telah terbuka. Jika suara atau ucapan tidak sesuai dengan kata kunci, maka tidak akan terjadi aksi apapun.

Pada sistem keamanan, yaitu pada sensor getaran dan sensor cahaya, jika sensor getaran mendeteksi getaran pada pintu pagar atau sensor cahaya mendeteksi penyusup memanjat pagar dengan kondisi sistem alarm aktif, maka sistem akan membunyikan suara alarm sampai batas waktu tertentu, tetapi jika sistem alarm tidak aktif, alarm tidak akan berbunyi.

Pada gambar 2.22 *flowchart* prototipe 2, ketika pintu terbuka sepenuhnya, akan mengaktifkan *limit switch* kedua dan membuat pintu langsung menutup kembali. Ketika proses menutup, sensor cahaya aktif untuk mendeteksi penghalang di jalur pintu. Jika di jalur pintu ada penghalang, maka pintu akan berhenti dan akan bergerak menutup kembali ketika sensor cahaya tidak mendeteksi adanya penghalang.

Pada gambar 2.23 *flowchart* prototipe 3 merupakan alur pengaktifan sistem alarm dengan pengkondisian. Jika kondisi awal sistem alarm aktif, maka sistem alarm akan nonaktif. Sedangkan jika kondisi awal sistem alarm nonaktif, maka sistem alarm akan aktif.

Pada gambar 2.24 *flowchart* prototipe 4 merupakan alur proses buka pintu dimana ketika pintu dalam proses membuka dan *limit switch* 2 aktif, pintu akan berhenti.

Pada gambar 2.25 *flowchart* prototipe 5 merupakan alur proses tutup pintu dimana ketika pintu dalam proses menutup dan *limit switch* 1 aktif, pintu akan berhenti dan kunci pintu akan aktif. Pada saat proses menutup, sensor cahaya juga aktif dan prosesnya sama dengan proses menutup pintu pada gambar 2.22.

Pada gambar 2.26 *flowchart* prototipe 6 merupakan alur sistem untuk mengganti *password* pintu. Pertama *password* sebelumnya akan dihapus kemudian sistem akan meminta untuk memasukkan *password* baru dan setelahnya ada pilihan untuk menekan tombol "*" untuk menyimpan *password* baru tersebut.

Pada gambar 2.27 *flowchart* prototipe 7 merupakan alur sistem untuk mengganti *password* alarm. Untuk prosesnya sama dengan pada gambar 2.26.

2.3 Hipotesis Penelitian

Berdasarkan dari kerangka teoritik dan kerangka berpikir yang telah dikemukakan sebelumnya, maka didapat hipotesis penelitian pada prototipe sistem keamanan pintu pagar dengan multi-*input* berbasis arduino ini bekerja sesuai dengan fungsinya, yaitu memiliki sistem keamanan dan multi-*input* yang baik.

BAB III
METODOLOGI PENELITIAN

3.1 Tempat dan Waktu Penelitian

3.1.1 Tempat Penelitian

Penelitian dilaksanakan di Bengkel Mekanik Fakultas Teknik Jurusan Teknik Elektro Universitas Negeri Jakarta.

3.1.2 Waktu Penelititian

Waktu penelitian dilaksanakan pada bulan Oktober 2014 sampai Maret 2015.

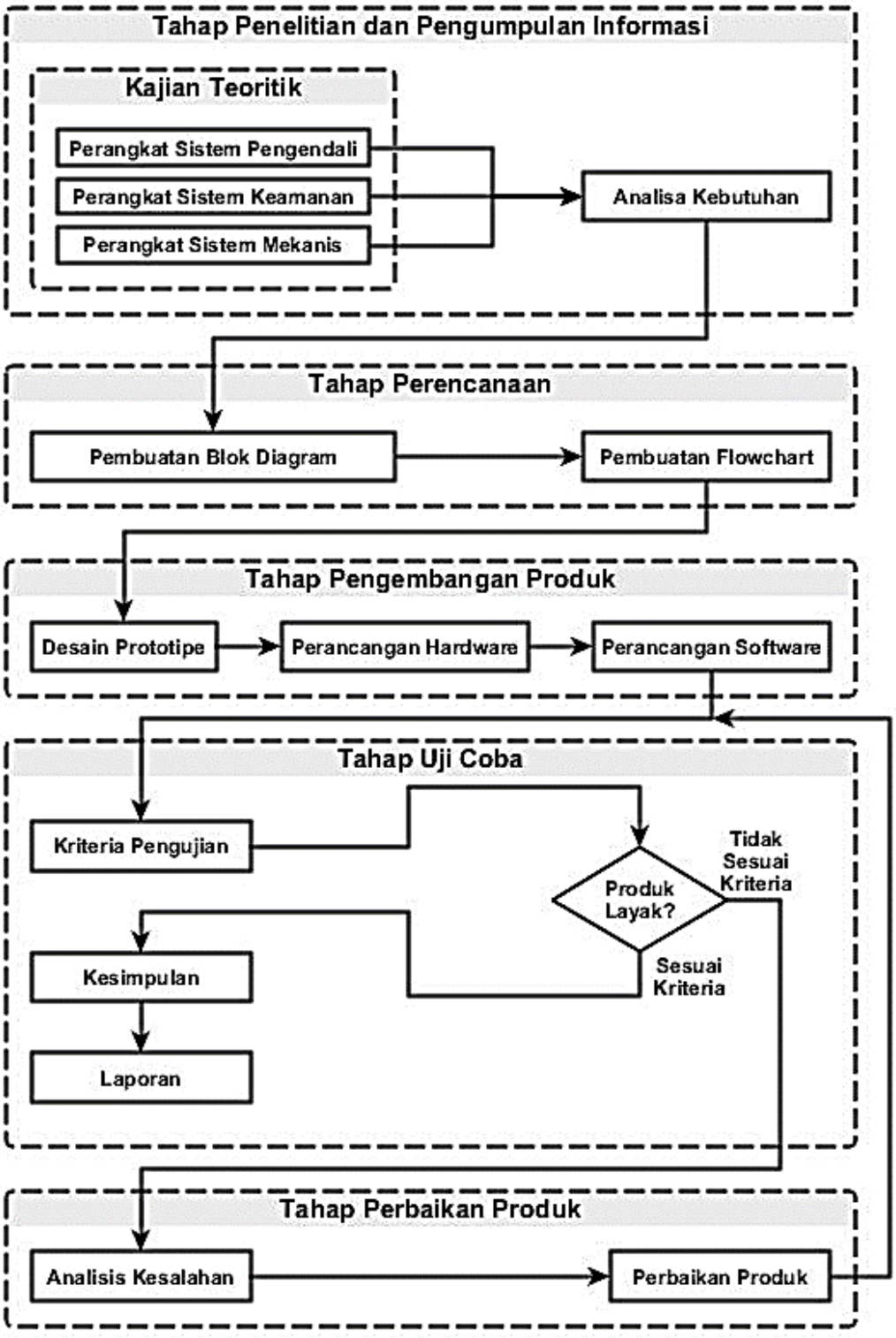
3.2 Metodologi Penelitian

Metodologi penelitian adalah langkah-langkah yang dilakukan dalam penelitian ilmiah yang bertujuan untuk mendapatkan hasil sehingga tujuan dari peneelitian tersebut dapat terpenuhi. Metodologi penelitian yang digunakan dalam menyelesaikan penelitian ini adalah menggunakan metodologi penelitian dan pengembangan (*Research and Development*). Borg and Gall menyatakan bahwa penelitian dan pengembangan (*Research and Development*), merupakan metode penelitian yang digunakan untuk mengembangkan atau memvalidasi produk-produk yang digunakan dalam pendidikan dan pembelajaran.[33]

[33] W.R Borg dan M. D. Gall, *Educational Research: An Introduction, Fifth Edition*, (New York: Longman, 1989), hlm 624.

Untuk langkah-langkah dalam penelitian, Brog & Gall mengungkapkan bahwa siklus R&D tersusun dalam beberapa langkah penelitian sebagai berikut: penelitian dan pengumpulan informasi (*research and information collecting*), perencanaan (*planning*), pengembangan produk awal (*develop premilinary from of product*), uji coba pendahuluan (*main field testing*), perbaikan produk operasional (*operasional product revision*), uji coba operasional (*operasional field testing*), perbaikan produk akhir (*final product revision*), disminasi dan pendistribusian (*dissemination and distribution*).

Langkah-langkah dalam penelitian ini mengacu pada langkah-langkah yang dikemukakan oleh Borg & Gall yang kemudian dimodifikasi oleh peneliti menjadi lima tahap yaitu tahap penelitian dan pengumpulan informasi (*research and information collecting*), tahap perencanaan (*planning*), tahap pengembangan produk (*develop of product*), tahap uji coba (*field test*) dan tahap perbaikan produk (*product revision*). Lebih jelasnya dapat dilihat pada gambar 3.1 berikut ini.



Gambar 3.1 *Flowchart* Penelitian

3.2.1 Tahap Penelitian dan Pengumpulan Informasi

Tahap penelitian dan pengumpulan informasi disini merupakan analisis kebutuhan sistem. Kebutuhan suatu sistem pada umumnya yaitu perangkat *input* dan *output* yang digunakan dalam sistem tersebut.

Pada penelitian ini, perangkat-perangkat yang dibutuhkan untuk membuat prototipe sistem keamanan pintu pagar dengan multi-*input* berbasis arduino, telah dijelaskan pada bab sebelumnya yang meliputi perangkat pendukung sistem pengendali, perangkat pendukung sistem keamanan dan perangkat pendukung sistem mekanis.

3.2.2 Tahap Perencanaan

Pada tahap ini berisi kerangka berpikir peneliti dalam pembuatan prototipe sistem keamanan pintu pagar yang telah dijelaskan pada bab sebelumnya meliputi blok diagram yang dapat dilihat pada gambar 2.19, blok komponen yang dapat dilihat pada gambar 2.20 dan *flowchart* keseluruhan sistem yang terdapat pada gambar 2.21, gambar 2.22, gambar 2.23, gambar 2.24, gambar 2.25, gambar 2.26 dan gambar 2.27.

3.2.3 Tahap Pengembangan Produk

Tahap ini merupakan tahap perancangan sistem dari prototipe yang akan dibuat. Pada tahap ini, peneliti membagi menjadi tiga tahap utama yang meliputi perancangan desain prototipe, perancangan perangkat keras dan perancangan perangakat lunak.

3.2.4 Tahap Uji Coba

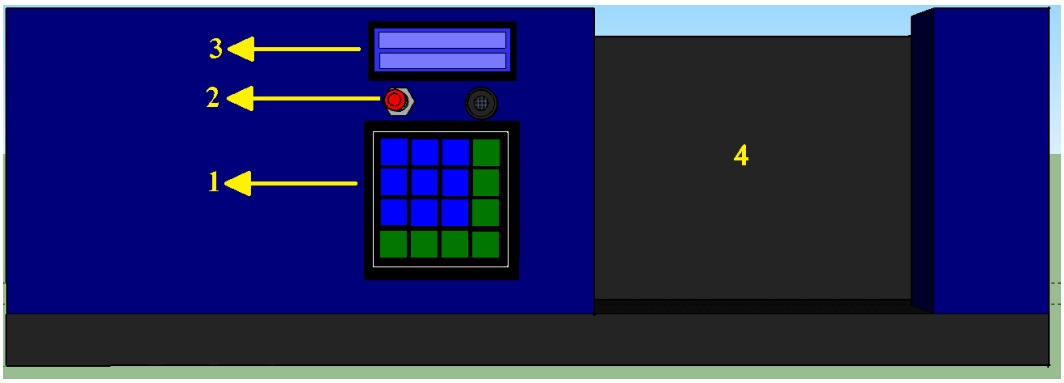
Tahap uji coba dilakukan setelah prototipe selesai dibuat. Pada tahap ini dilakukan pengujian kelayakan cara kerja prototipe untuk mengetahui kelayakan produk yang dihasilkan sesuai dengan kriteria yang telah ditentukan.

3.2.5 Tahap Perbaikan Produk

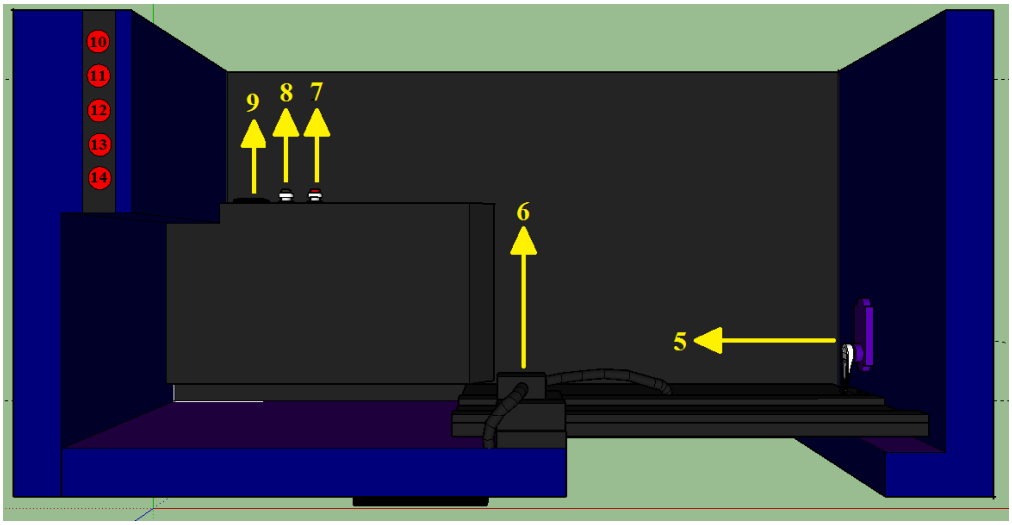
Tahap perbaiakan produk dilakukan ketika hasil uji coba tidak sesuai dengan perencanaan yang bertujuan untuk mencari kesalahan dan kekurangan pada sistem agar dapat diperbaiki sehingga mendapatkan hasil yang sesuai dengan perencanaan dan kriteria yang telah ditentukan.

3.3 Perancangan Desain Prototipe

Prototipe pintu pagar ini dibuat dari bahan triplek, akrilik dan kayu serta dilapisi *sticker* warna hitam dan biru dengan dimensi 70x37x24 cm^3. Desain dari prototipe dibuat dengan *software Sketchup* sehingga gambar yang dihasilkan berupa gambar tiga dimensi. Desain tersebut digambarkan pada gambar 3.2 dan gambar 3.3 berikut ini.



Gambar 3.2 Desain Prototipe Pintu Pagar Tampak Depan



Gambar 3.3 Desain Prototipe Pintu Pagar Tampak Atas

Keterangan dari gambar desain prototipe adalah sebagai berikut:

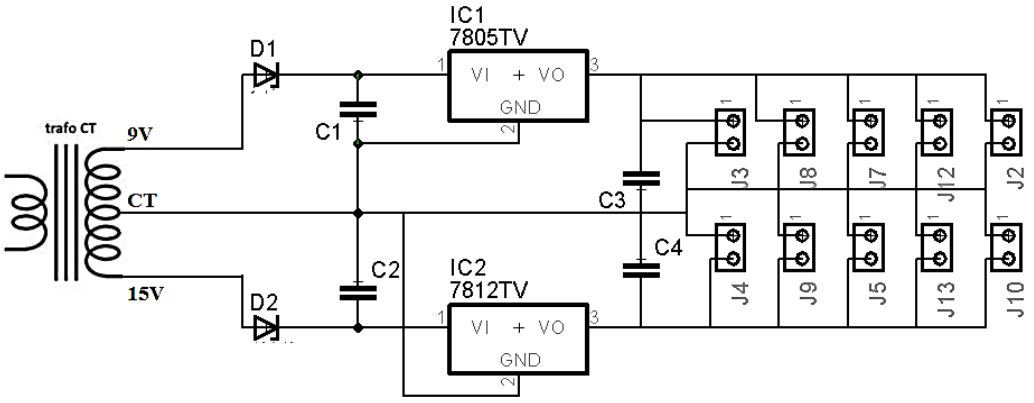
1. *Keypad.*
2. *Microphone.*
3. LCD *(Liquid Crystal Display).*
4. Pintu Pagar.
5. Motor Servo.
6. Sensor Getar.
7. *Button* untuk buka/tutup pintu.
8. *Button* untuk buka pintu dengan menutup otomatis.
9. Tombol Power.
10. Led indikator *standby*.
11. Led indikator tutup pintu.
12. Led indikator pintu tertutup dan terkunci.
13. Led indikator buka pintu.
14. Led indikator alarm berbunyi.

3.3.1 Perancangan Perangkat Keras

Perancangan perangkat keras pada prototipe ini yaitu berupa rangkaian – rangkaian dan komponen – komponen pendukung yang telah dijelaskan pada bab sebelumnya.

3.3.1.1 Rangkaian Catu Daya

Pada penelitian ini, peneliti menggunakan catu daya untuk mensuplai tegangan ke setiap rangkaian dan komponen utama ataupun pendukungnya yang digunakan dalam prototipe. Tegangan pada catu daya yang diperlukan adalah 12 Volt DC dan 5 Volt DC karena setiap komponen yang dipakai membutuhkan tegangan yang berbeda-beda. Untuk memenuhi kebutuhan tersebut, catu daya menggunakan IC regulator 7812 untuk menghasilkan tegangan 12 Volt DC dan IC regulator 7805 untuk menghasilkan tegangan 5 Volt DC. Untuk lebih jelasnya, dapat dilihat pada rancangan rangkaian catu daya yang digambarkan pada gambar 3.4 berikut ini.



Gambar 3.4 Rangkaian Catu Daya

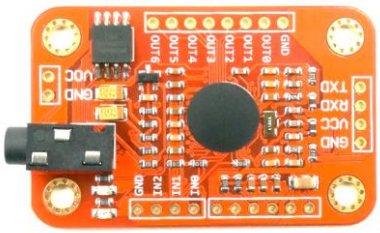
Rangkaian atau komponen yang menggunakan tegangan 5 Volt DC meliputi arduino, modul *voice recognition*, modul *bluetooth*, sensor getaran (adxl345), LCD

16x2, modul ISD 1820, *photo diode*, *buzzer* dan motor servo. Sedangkan rangkaian atau komponen yang menggunakan tegangan 12 Volt DC adalah modul *driver* motor DC L298.

3.3.1.2 Rangkaian Modul Voice Recognition

Pada penelitian ini, peneliti menggunakan *voice recognition module v3* (lihat gambar 3.5) dimana modul tersebut hanya kompatibel dengan salah satu arduino yaitu arduino uno. Modul ini bekerja pada tegangan antara 4,5-5,5volt dengan arus kurang dari 40mA. Untuk lebih jelasnya, berikut adalah fitur dari modul tersebut:

1. Mendukung hingga 80 perntah suara dimana masing-masing suara berdurasi 1500ms.
2. Bisa bekerja dengan 7 perintah suara dalam waktu yang sama.
3. Sudah memiliki *library* untuk arduino sehingga kompatibel dengan arduino (arduino uno).
4. Mudah dalam pemakaiannya karena tidak membutuhkan tambahan komponen lain[34].

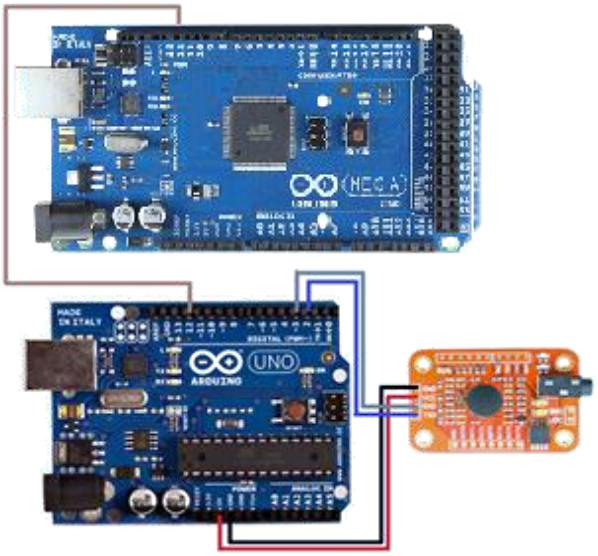


Gambar 3.5 *Voice Recognition Module V3*

[34]Voice Recognition Module V3 user-manual

Sebelum modul ini digunakan untuk mengenali suara, dilakukan “latihan” terlebih dahulu, maksudnya memasukkan frasa atau kata yang akan dijadikan kode, kemudian pengguna menyebutkan kata tersebut sampai terdeteksi, barulah kode tersebut bisa digunakan.

Pada rangkaian modul *voice recognition* ini menggunakan arduino uno sebagai bagian dari sub sistem pada sistem *input voice recognition*. Keluaran sub sistem tersebut sudah berupa logika 1 dan 0 yang menjadi masukan ke arduino mega sebagai mikrokontroler utamanya. Untuk lebih jelasnya bisa dilihat pada gambar 3.6 berikut ini.

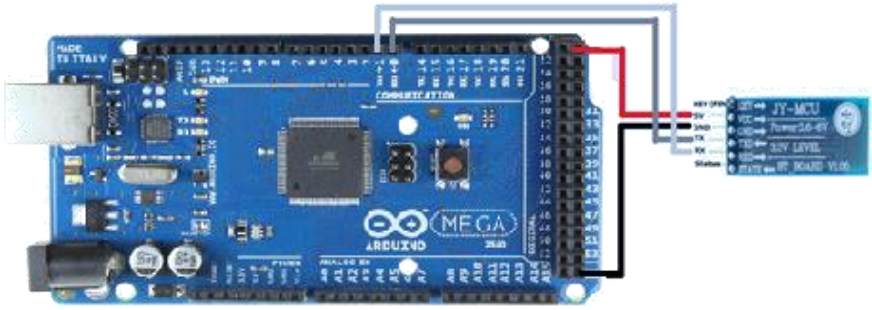


Gambar 3.6 Koneksi Pin *Voice Recognition* dengan Arduino

3.3.1.3 Rangkaian Modul *Bluetooth*

Pada penelitian ini, peneliti menggunakan modul *bluetooth* seri HC-05. Modul *bluetooth* HC-05 adalah modul *bluetooth* SPP (*Serial Port Protocol*) yang mudah digunakan untuk komunikasi serial *wireless* (nirkabel) yang mengkonversi port serial ke *bluetooth*. HC-05 menggunakan modulasi *bluetooth* V2.0 + EDR

(*Enchanced Data Rate*) 3 Mbps dengan memanfaatkan gelombang radio berfrekuensi 2,4 GHz[35].

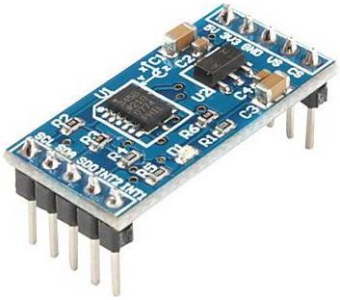


Gambar 3.7 Koneksi Pin Modul *Bluetooth* dengan Arduino

Pada gambar 3.7 merupakan koneksi pin dari modul *bluetooth* ke arduino dimana modul *bluetooth* dihubungkan dengan arduino melalui pin Tx dan Rx (komunikasi serial).

3.3.1.4 Rangkaian Sensor Getaran

Penggunaan sensor getaran pada penelitian ini menggunakan komponen yang dapat mendeteksi perubahan ke segala arah yaitu menggunakan *accelerometer* adxl345. Untuk lebih jelasnya bisa dilihat pada gambar 3.8 berikut ini.



Gambar 3.8 *Accelerometer* ADXL345

Gambar 3.8 adalah gambar *accelerometer* tipe adxl345. Sensor ini dapat bekerja pada 3 sumbu dengan resolusi tinggi (13bit) dan pengukurannya sampai

[35] Datasheet Modul Bluetooth HC-05

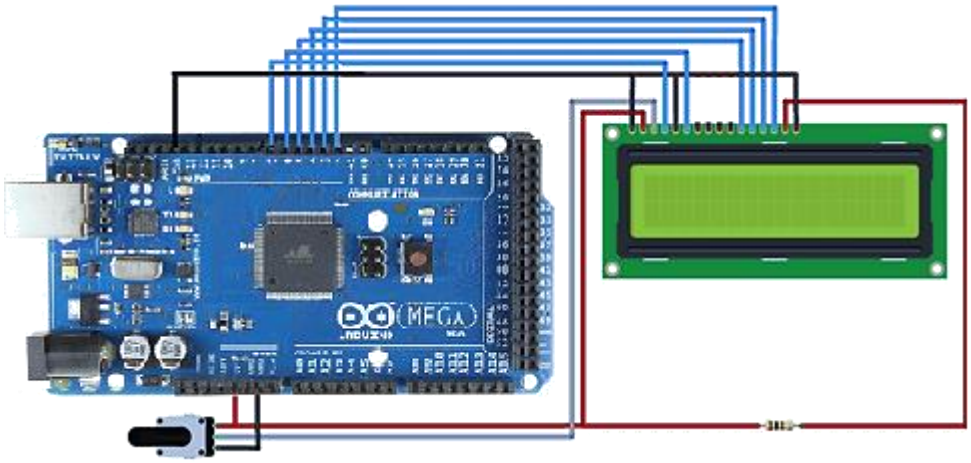
dengan ± 16g. Keluarannya berupa data digital (16bit) yang saling berpasangan dan dapat diakses melalui SPI ataupun I^2C.[36]

Untuk pengkoneksiannya dengan arduino mega 2560, adxl345 menggunakan pin SDA dan SCL yaitu terdapat pada pin 20 dan pin 21. Rangkaiannya dapat dilihat pada gambar 3.9 berikut ini.

Gambar 3.9 Koneksi Pin ADXL345 dengan Arduino

3.3.1.5 Rangkaian LCD 16x2

Pada prototipe pintu pagar ini, peneliti menggunakan LCD 16x2 untuk menampilkan informasi sederhana. LCD dihubungkan dengan arduino mega 2560 dan menggunakan tegangan 5 Volt DC.



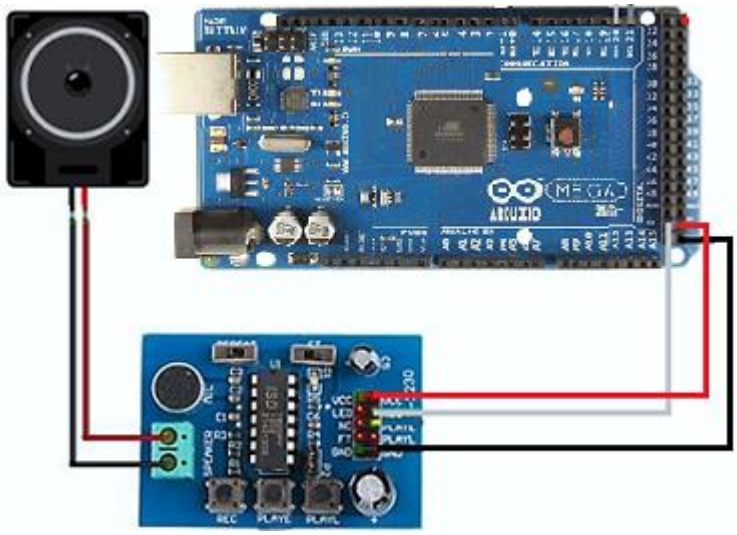
Gambar 3.10 Koneksi Pin LCD 16x2 dengan Arduino

[36] Datasheet adxl345

Pada gambar 3.10 merupakan rangkaian LCD 16x2 yang dihubungkan dengan arduino mega 2560 beserta dengan pin yang digunakan. Varistor digunakan untuk mengatur kecerahan pada LCD 16x2.

3.3.1.6 Rangkaian Modul ISD 1820

Indikator sistem keamanan yang dipakai dalam prototipe pintu pagar ini adalah berupa alarm. Dalam penelitian ini, peneliti menggunakan modul ISD 1820 sebagai sistem alarm yang digunakan, dimana yang sudah dijelaskan pada bab sebelumnya. Modul ISD 1820 dihubungkan dengan arduino mega 2560 dimana modul tersebut akan aktif ketika ada peringatan. Modul ini menggunakan tegangan 5 Volt DC dan *output*nya menggunakan speaker 8 ohm.



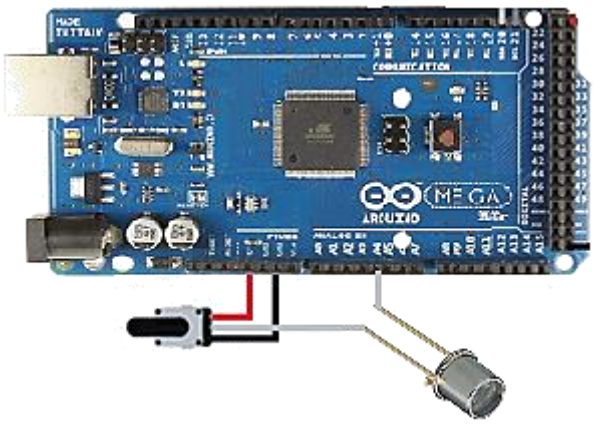
Gambar 3.11 Koneksi Pin Modul ISD 1820 dengan Arduino

Pada gambar 3.11 merupakan rangkaian modul ISD 1820 yang dihubungkan dengan arduino mega beserta pin yang dipakai. Sebelum digunakan, modul ISD 1820 harus merekam terlebih dahulu suara yang akan diputar sebagai alarm.

3.3.1.7 Rangkaian *Photo Diode*

Photo diode merupakan salah satu sensor cahaya dengan prinsip perubahan resistansi atau tahanan yang dipengaruhi oleh besar kecilnya intensitas cahaya yang diterima, semakin besar intensitas cahaya yang diterima, semakin kecil resistansinya.

Pada penelitian ini, *photo diode* ditembak dengan cahaya led *super bright*, dan digunakan pada pintu pagar sebagai pendeteksi orang ataupun kendaraan yang melewati pintu ketika pintu hendak menutup. Hal tersebut dimaksudkan agar ketika pintu sedang menutup, tetapi di jalur pintu masih ada yang menghalangi, pintu akan otimatis berhenti dan akan bergerak kembali ketika tidak ada penghalang.



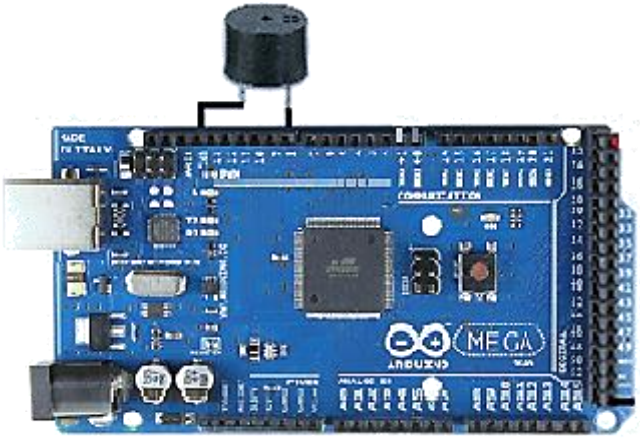
Gambar 3.12 Koneksi Pin *Photo Diode* dengan Arduino

Gambar 3.12 merupakan *photo diode* yang dikoneksikan dengan pin masukan analog pada arduino dan dihubungkan pula dengan varistor sebagai pengkalibrasinya. Penggunaan pin analog dimaksudkan agar pembacaan sensor lebih stabil dan mendapatkan hasil yang baik. Sensor cahaya yang digunakan dalam penelitian ini menggunakan tegangan 5 Volt DC.

3.3.1.8 Rangkaian *Buzzer*

Buzzer adalah komponen elektronika yang mengubah getaran listrik manjadi getaran suara. Penggunaan *buzzer* pada umumnya hanya sebagai indikator dari suatu masukan, proses maupun keluaran.

Pada prototipe pintu pagar ini, peneliti menggunakan *buzzer* untuk indikator pada *keypad*. Hal ini dimaksudkan agar ketika *keypad* ditekan, akan terindikasi bahwa *keypad* benar-benar telah ditekan ketika *buzzer* berbunyi. *Buzzer* yang digunakan adalah *buzzer* yang bekerja pada tegangan 5 Volt DC.



Gambar 3.13 Koneksi Pin *Buzzer* dengan Arduino

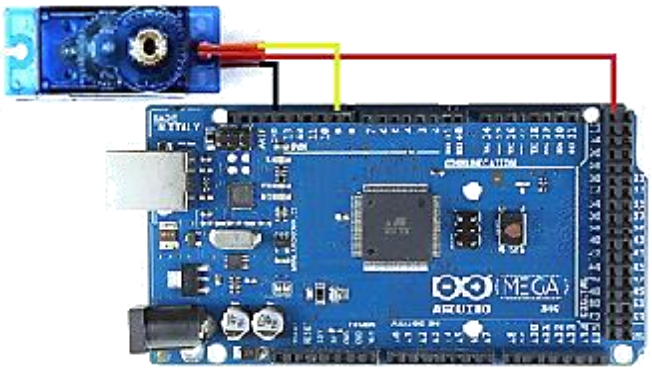
Pengkoneksian pin dari *buzzer* ke arduino ditunjukkan pada gambar 3.13 dimana *buzzer* hanya membutuhkan satu pin digital dan ground dalam penggunaannya. Jadi, ketika pin digital pada *buzzer* diberi logika *high*, maka *buzzer* akan berbunyi dan begitu pula sebaliknya, ketika pin digital pada *buzzer* diberi logika *low*, maka *buzzer* akan mati.

3.3.1.9 Rangkaian Motor Servo

Dalam pembuatan pintu pagar, selalu ada pengunci pintu dalam sistemnya yang dimaksudkan untuk pengamanan. Dalam hal ini peneliti menggunakan motor

servo untuk sistem pengunci pintu pagar. Motor servo yang digunakan adalah tipe mini servo dengan maksimal beban yang bisa digerakkan sebesar 1,5Kg.

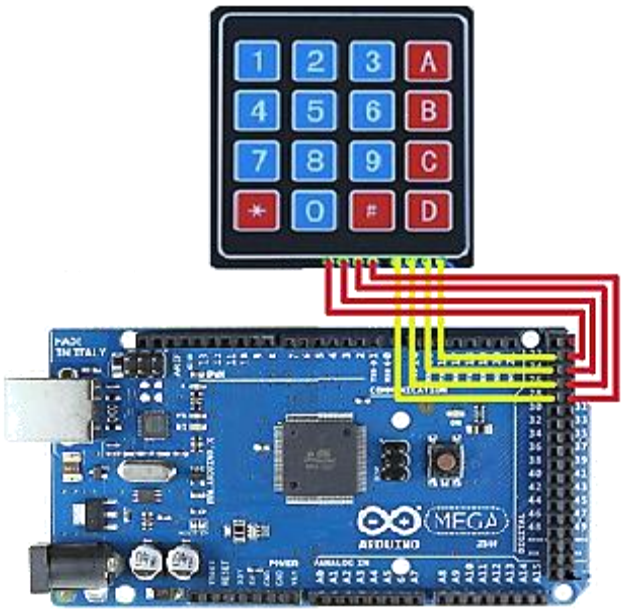
Motor servo ini memiliki 3 pin, yaitu satu untuk vcc, satu untuk ground dan satu untuk data. Tegangan untuk menggerakkan motor servo ini yaitu sebesar 5 Volt DC. Untuk pengkoneksian pin motor servo dengan arduino dapat dilihat pada gambar 3.14 berikut ini.



Gambar 3.14 Koneksi Pin Motor Servo dengan Arduino

3.3.1.10 Rangkaian *Keypad* 4x4

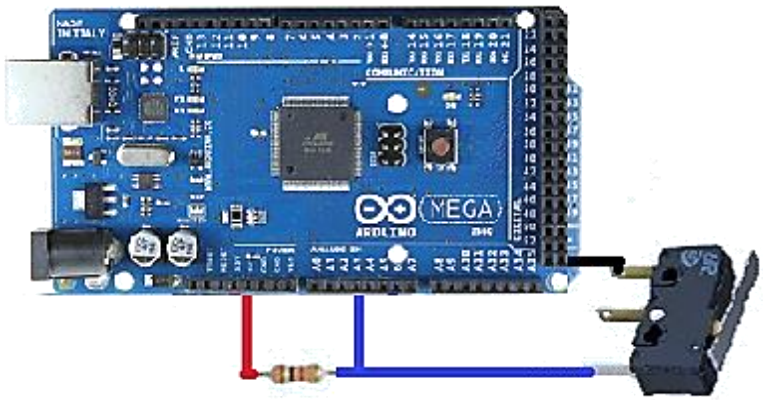
Keypad yang digunakan dalam penelitian ini adalah *keypad* dengan bahan membran yang memiliki 4 pin baris dan 4 pin kolom untuk dihubungkan dengan pin digital pada arduino sebanyak 8 pin. Pada gambar 3.15 berikut ini bisa dilihat pengkoneksian *keypad* dengan arduino.



Gambar 3.15 Koneksi Pin *Keypad* 4x4 dengan Arduino

3.3.1.11 Rangkaian Limit Switch

Limit switch merupakan salah satu sensor mekanik yang mempunyai prisnsip kerja seperti saklar tukar. Pada penelitian ini, *limit switch* digunakan untuk membatasi gerak dari pintu pagar bak ketika membuka ataupun menutup.



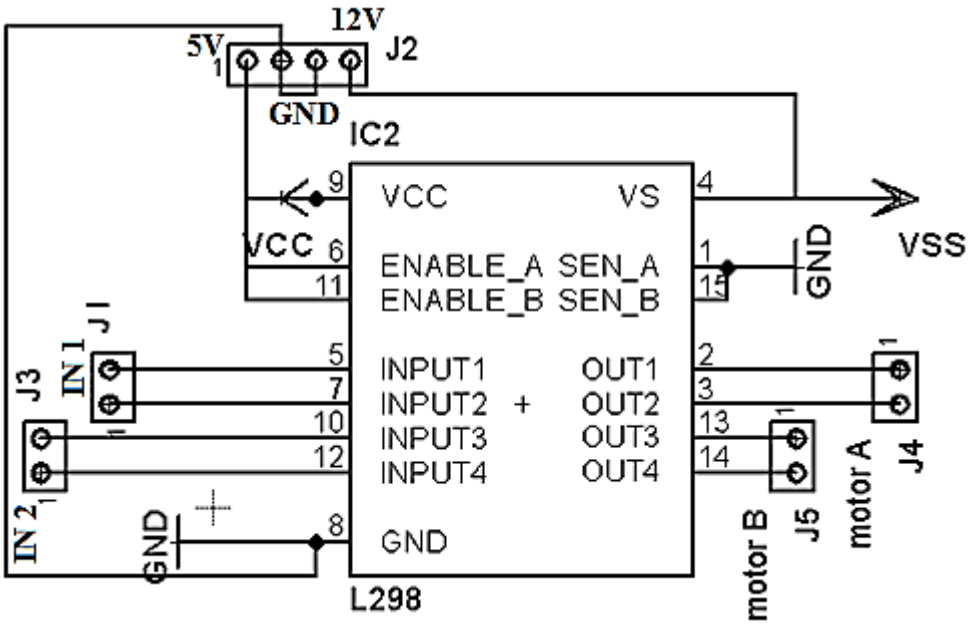
Gambar 3.16 Koneksi Pin *Limit Switch* dengan Arduino

Pada gambar 3.16 merupakan koneksi pin antara *limit switch* dengan arduino dimana pin arduino yang dipakai adalah pin analog. Resistor digunakan sebagai *pull up* pada rangkaian *limit switch.*

3.3.1.12 Rangkaian *Driver*

Motor DC yang digunakan untuk penggerak buka tutup pintu, harus menggunakan *driver* motor agar motor dapat bergerak secara bolak balik. *Driver* yang digunakan dalam penelitian ini adalah *driver* IC L298.

Driver L298 bekerja pada tegangan 5 Volt DC dan 12 Volt DC untuk keluarannya. Pada gambar 3.17 merupakan rangkaian *driver* L298. Tegangan *input* 5 Volt dihubungkan pada pin 9 (Vcc) dan tegangan *input* 12 Volt dihubungkan pada pin 4 (Vss).



Gambar 3.17 Rangkaian *Driver* Motor DC L298

3.3.2 Perancangan Perangkat Lunak

Perangkat lunak dalam penelitian ini adalah berupa program untuk arduino dan aplikasi android. Seperti yang telah dijelaskan pada bab sebelumnya, untuk pemrograman arduino menggunakan pernagkat lunak arduino IDE dan untuk pemrograman android menggunakan *app inventor*.

3.3.2.1 Perancangan Program Arduino

Peneliti menggunkan arduino IDE 1.0.5 untuk pemrograman arduino pada prototipe pintu pagar ini. Dalam pemrograman ini, ditentukan penggunaan pin *input* dan *output* yang digunakan pada arduino dengan perangkat-perangkat yang telah dijelaskan sebelumnya. Untuk penggunaan pin *input* pada arduino, dapat dilihat pada tabel 3.1 berikut ini.

Tabel 3.1 Penggunaan Pin *Input* pada Arduino Mega 2560 dengan Perangkat *Input*

No.	Perangkat *Input*	Pin Perangkat *Input*	Pin Arduino Mega 2560
1	*Keypad*	Baris 1	23
		Baris 2	25
		Baris 3	27
		Baris 4	29
		Kolom 1	28
		Kolom 2	26
		Kolom 3	24
		Kolom 4	22
2	Sistem *Voice Recognition*	12 (Arduino Uno)	12
3	*Limit Switch 1*	*Output* Analog	A2
4	*Limit Switch 2*	*Output* Analog	A3
5	*Photo Diode 1*	*Output* Analog	A4
6	*Photo Diode 2*	*Output* Analog	A5
7	Sensor getaran (*Accelerometer* ADXL345)	SDA	20 (SDA)
		SCL	21 (SCL)

Untuk penggunaan pin *output* pada arduino dengan perangkat *output*, dapat dilihat pada tabel 3.2 berikut ini.

Tabel 3.2 Penggunaan Pin *Output* pada Arduino Mega 2560 dengan Perangkat *Output*

No.	Perangkat *Output*	Pin Perangkat *Output*	Pin Arduino Mega 2560
1	*Driver* Motor DC	*Input* 1	30
		Input 2	31
2	Modul ISD 1820	*Input* Play	53
		Input Vcc	52
3	Led *Standby*	*Input* Vcc	51
4	Led Tutup	*Input* Vcc	50
5	Led Buka	*Input* Vcc	49
6	Led Kunci	*Input* Vcc	48
7	Led Alarm	*Input* Vcc	47
8	*Buzzer*	*Input* Power	8
9	Motor Servo	*Input* Data	9

Selain perangkat *input* dan *output*, ada perangkat yang menggunakan komunikasi serial, yaitu modul *bluetooth*. Untuk penggunaan pin komunikasi serial pada arduino dengan *bluetooth*, dapat dilihat pada tabel 3.3 berikut ini.

Tabel 3.3 Penggunaan Pin Komunikasi Serial pada Arduino Mega 2560 dengan Perangkat Komunikasi Serial

No.	Perangkat Komunikasi Serial	Pin Perangkat Komunikiasi Serial	Pin Arduino Mega 2560
1	Modul *Bluetooth* HC-05	Tx	0 (Rx)
		Rx	1 (Tx)

Sebelumnya telah dijelaskan bahwa penggunaan arduino uno adalah sebagai bagian dari sub sistem *voice recognition*. Untuk penggunaan komunikasi *voice recognition* dengan arduino uno dapat dilihat pada tabel 3.4 berikut ini.

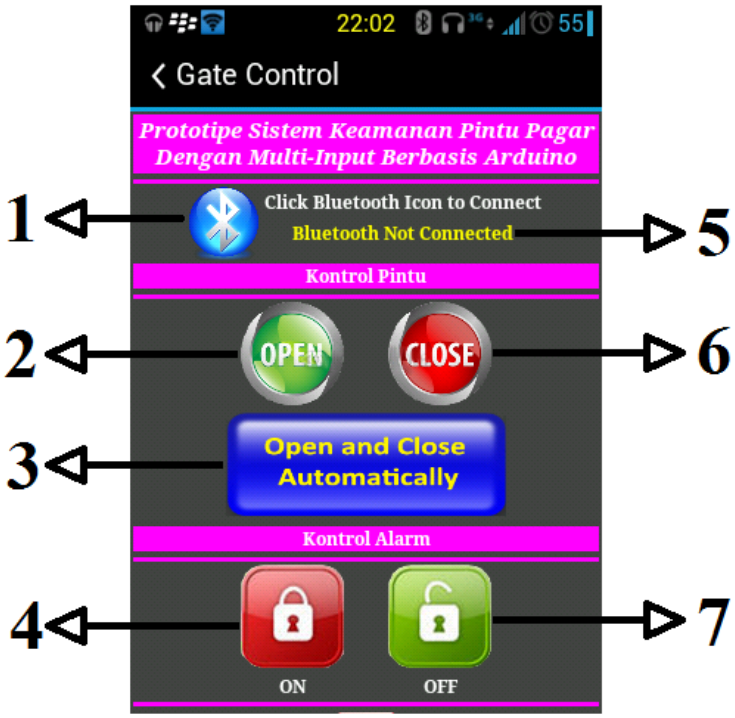
Tabel 3.4 Penggunaan Pin pada Arduino Uno dengan Modul Voice Recognition

No.	Perangkat Komunikiasi Serial	Pin Perangkat Komunikiasi Serial	Pin Arduino Uno
1	Modul *Voice Recognition*	Tx	2
		Rx	3

3.3.2.2 Perancangan APK

Salah satu *input* untuk mengendalikan pintu pagar pada penelitian ini adalah dengan kendali *smartphone* android melalui media *bluetooth*. Hal pertama yang dilakukan sebelum *smartphone* android dapat dijadikan sebagai pengendali pintu pagar adalah memasang aplikasi pendukungnya yang dibuat sesuai kebutuhan prototipe pintu pagar.

Aplikasi yang dibuat berekstensi apk, yaitu aplikasi yang didukung oleh sistem operasi android yang didesain dan diprogram dengan *app inventor* yang telah dijelaskan pada bab sebelumnya. Tampilan aplikasi tersebut dapat dilihat pada gambar 3.18 berikut ini.



Gambar 3.18 APK Pengendali Pintu Pagar

Keterangan:

1. Memilih perangkat *bluetooth* yang akan dihubungkan dengan android.
2. Tombol untuk membuka pintu pagar.
3. Tombol untuk membuka pintu pagar, tetapi ketika pintu pagar telah terbuka penuh, akan langsung menutup otomatis.
4. Tombol untuk mengaktifkan sistem alarm.
5. Label indikator koneksi android dengan perangkat *bluetooth*. Jika perangkat *bluetooth* terhubung, label akan berubah menjadi *"Bluetooth is Conneted"* tetapi jika tidak, label menjadi *"Bluetooth Not Connected"*.
6. Tombol untuk menutup pintu.
7. Tombol untuk mematikan sistem alarm.

3.4 Instrumen Penelitian

Instrumen yang digunakan dalam penelitian ini terdiri dari:

1. Komputer/laptop dengan spesifikasi sebagai berikut:
   a. *Processor Intel*(R) *Core*(TM) i3 CPU M 370 @2,40GHz.
   b. *Memory* 2,00 GB RAM
   c. Sistem Operasi *Windows* 8.1 Pro 32 bit

2. *Software* Pendukung:
   a. Arduino IDE 1.0.5, yang digunakan untuk memrogram *board* Arduino.
   b. *Eagle* 6.5, yang digunakan untuk membuat gambar skematik dan layout rangkaian pada PCB.
   c. *Google SketchUp* 2014, yang digunakan untuk membuat desain perancangan maket.
   d. *Microsoft Office Word* 2013, yang digunakan untuk penulisan.
   e. MIT *App Inventor*, yang digunakan untuk membuat aplikasi android.
   f. YED *Graph Editor*, yang digunakan untuk pembuatan blok diagram dan *flowchart*.

3. *Hardware* Pendukung:
   a. Mini *electric drill* (Bor tangan kecil).
   b. Solder listrik.
   c. Multimeter Analog
   d. Multimeter Digital

e. *Cutter*.

f. Gunting.

g. Palu.

h. Tang Jepit.

i. Tang Potong.

j. Gergaji.

k. Obeng.

3.5 Teknik Analisis Data

Analisis data merupakan membandingkan data hasil pengujian dengan kriteria pengujian yang telah ditentukan oleh peneliti agar peneliti dapat menyatakan berhasil tidaknya sistem yang telah dibuat.

Ada beberapa kriteria pengujian yang dilakukakan dalam penelitian ini, yaitu pengujian sistem pengendali, pengujian sistem keamanan, pengujian sistem mekanis dan pengujian tegangan *output* catu daya.

3.5.1 Pengujian Sistem Pengendali

Pengujian pada sistem pengendali meliputi pengujian sistem *bluetooth*, sistem *keypad* dan sistem *voice recognition*.

3.5.1.1 Kriteria Pengujian Sistem *Bluetooth*

Pengujian sistem *bluetooth* dilakukan dengan kriteria sebagai berikut:

1. Menguji seberapa jauh jarak koneksi *bluetooth* dapat dilakukan dengan adanya halangan ataupun tanpa halangan.

2. Menguji berfungsi atau tidaknya aplikasi android yang telah dibuat sesuai dengan yang direncanakan..

Pengujian jarak koneksi android dengan perangkat *bluetooth* pada prototipe pintu pagar dapat dilihat pada tabel 3.5 berikut ini.

Tabel 3.5 Kriteria Pengujian Jarak Koneksi *Bluetooth*

No.	Jarak (meter)	Kondisi	Kriteria Pengujian	Hasil Pengujian
1	2	Tanpa Halangan	Terkoneksi dengan perangkat *bluetooth*	
		Ada Halangan		
2	4	Tanpa Halangan		
		Ada Halangan		
3	6	Tanpa Halangan		
		Ada Halangan		
4	8	Tanpa Halangan		
		Ada Halangan		
5	10	Tanpa Halangan		
		Ada Halangan		
6	12	Tanpa Halangan		
		Ada Halangan		

Pengujian berfungsi atau tidaknya sesuai kriteria dari aplikasi android yang telah dibuat, dapat dilihat pada tabel 3.6 berikut ini.

Tabel 3.6 Kriteria Pengujian Aplikasi Android

No.	Tombol yang Ditekan	Kriteria Pengujian	Hasil Pengujian
1	*Icon Bluetooth*	Tampilan memilih perangkat *bluetooth* yang tersedia	
2	*Open*	Pintu pagar terbuka	
3	*Close*	Pintu pagar tertutup	

4	*Open and Close Automatically*	Pintu pagar terbuka dan tertutup otomatis	
5	*ON*	Sistem alarm aktif	
6	*OFF*	Sistem alarm nonaktif	
7	*Exit*	Keluar	

3.5.1.2 Kriteria Pengujian Sistem *Keypad*

Pengujian sistem *keypad* dilakukan dengan cara memasukkan *password* dimana jika *password* yang dimasukkan sesuai, sistem akan bekerja sesuai yang telah ditentukan. Kriteria pengujian dari sistem *keypad* adalah sebagai berikut:

1. Memasukkan *password* "222222" lalu menekan "*" untuk membuka pintu dan menutup secara otomatis setelah terbuka penuh.
2. Memasukkan *password* "222222" lalu menekan "#" untuk mengganti *password* buka pintu.
3. Memasukkan *password* "333333" lalu menekan "*" untuk mengaktifkan sistem alarm jika sistem alarm nonaktif.
4. Memasukkan *password* "333333" lalu menekan "*" untuk menonaktifkan sistem alarm jika sistem alarm aktif.
5. Memasukkan *password* "333333" lalu menekan "#" untuk mengganti *password* sistem alarm.

Untuk kriteria pengujian berfungsinya *password* pada sistem *keypad*, dapat dilihat pada tabel 3.7 berikut ini.

Tabel 3.7 Kriteria Pengujian Sistem *keypad*

No.	*Input Keypad*	Kriteria Pengujian	Hasil Pengujian
1	Memasukkan *password* "222222" lalu menekan "*"	Pintu pagar terbuka dan tertutup otomatis setelah terbuka penuh	
2	Memasukkan *password* "222222" lalu menekan "#"	Mengganti *password* buka pintu	
3	Memasukkan *password* "333333" lalu menekan "*" ketika sistem alarm aktif	Sistem alarm nonaktif	
4	Memasukkan *password* "333333" lalu menekan "*" ketika sistem alarm nonaktif	Sistem alarm aktif	
5	Memasukkan *password* "333333" lalu menekan "#"	Mengganti *password* sistem alarm	

3.5.1.3 Kriteria Pengujian Sistem *Voice Recognition*

Pengujian pada sistem *voice recognition* dilakukan dengan cara mengucapkan kata sesuai dengan yang telah diprogram atau direkam dalam modul *voice recognition*. Kriteria dari pengujian sistem *voice recognition* adalah:

1. Mengucapkan *"Open"* untuk membuka pintu pagar dan menutup secara otomatis ketika pintu terbuka penuh.
2. Mengetahui tegangan *output* arduino uno pada sistem *voice recognition.*

Untuk lebih jelasnya, dapat dilihat pada tabel 3.8 berikut ini.

Tabel 3.8 Kriteria Pengujian Sistem *Voice Recognition*

No.	Kata Kunci	*Output* Arduino Uno	Kriteria Pengujian		Hasil Pengujian	
			Pada Prototipe	Tegangan *Output* Arduino Uno	Pada prototipe	Tegangan *Output* Arduino Uno
1	*"Open"*	Pin 12	Pintu pagar terbuka dan tertutup otomatis setelah terbuka penuh	5 Volt DC		

3.5.2 Pengujian Sistem Keamanan

Pengujian sistem keamanan hanya dapat dilakukan ketika sistem alarm aktif. Pengujian sistem keamanan ini meliputi pengujian pada sensor getaran dan sensor cahaya (*photo diode).*

3.5.2.1 Kriteria Pengujian Sensor Getaran

Sensor getaran akan aktif ketika menerima getaran. Kriteria dari pengujian ini adalah ketika sensor getaran dan sistem alarm aktif, maka alarm akan berbunyi. Untuk lebih jelasnya, dapat dilihat pada tabel 3.9 berikut ini.

Tabel 3.9 Kriteria Pengujian Sensor Getaran

No.	Sensor Getaran	Kondidsi Pendukung	Kriteria Pengujian	Hasil Pengujian
1	Deteksi getaran	Sistem alarm aktif	Alarm berbunyi	
2	Deteksi getaran	Sistem alarm nonaktif	Alarm tidak berbunyi	

3.5.2.2 Kriteria Pengujian *Photo Diode*

Photo diode yang digunakan berjumlah dua, yang pertama sebagai pendeteksi penghalang pada jalur pintu pagar dan yang kedua sebagai pendeteksi pemanjat tembok. Kriteria pengujian *photo diode* adalah sebagai berikut:

1. Ketika *photo diode* 1 menerima cahaya dari led *super bright* saat pintu sedang menutup, maka pintu akan terus menutup.
2. Ketika *photo diode* 1 tidak menerima cahaya led *super bright* saat pintu sedang menutup, maka pintu akan berhenti.
3. Ketika *photo diode* 2 menerima cahaya dari led *super bright* dan sistem alarm aktif, maka alarm tidak akan berbunyi.
4. Ketika *photo diode* 2 menerima cahaya dari led *super bright* dan sistem alarm nonaktif, maka alarm tidak akan berbunyi.
5. Ketika *photo diode* 2 tidak menerima cahaya dari led *super bright* dan sistem alarm aktif, maka alarm akan berbunyi.
6. Ketika *photo diode* 2 tidak menerima cahaya dari led *super bright* dan sistem alarm nonaktif, maka alarm tidak akan berbunyi.
7. Mengetahui tegangan *output* dari *photo diode* saat menerima cahaya maupun tidak.

Pengujian sensor cahaya sesuai dengan kriteria yang ditentukan dapat dilihat pada taabel 3.10 berikut ini.

Tabel 3.10 Kriteria Pengujian *Photo Diode*

No.	Kondisi Sensor	Kondisi Pendukung	Kriteria Pengujian	Hasil Pengujian
1	*Photo diode* 1 menerima cahaya dari led *super bright*	Pintu sedang menutup	Pintu terus menutup	
2	*Photo diode* 1 tidak menerima cahaya dari led *super bright*	Pintu sedang menutup	Pintu berhenti	
3	*Photo diode* 2 menerima cahaya dari led *super bright*	Sistem alarm aktif	Alarm tidak berbunyi	
4	*Photo diode* 2 menerima cahaya dari led *super bright*	Sistem alarm nonaktif	Alarm tidak berbunyi	
5	*Photo diode* 2 tidak menerima cahaya dari led *super bright*	Sistem alarm aktif	Alarm berbunyi	
6	*Photo diode* 2 tidak menerima cahaya dari led *super bright*	Sistem alarm nonaktif	Alarm tidak berbunyi	

Tegangan *input* yang digunakan pada *photo diode* adalah sebesar 5 Volt DC. Tabel 3.11 adalah kriteria pengujian tegangan *output* pada *photo diode*.

Tabel 3.11 Kriteria Pengujian Tegangan *Output Photo Diode*

No.	Kondisi Sensor	Kriteria Pengukuran	Hasil Pengukuran
1	*Photo diode* 1 menerima cahaya dari led *super bright*	0 Volt DC	
2	*Photo diode* 1 tidak menerima cahaya dari led *super bright*	5 Volt DC	
3	*Photo diode* 2 menerima cahaya dari led *super bright*	0 Volt DC	
4	*Photo diode* 2 tidak menerima cahaya dari led *super bright*	5 Volt DC	

3.5.3 Pengujian Sistem Mekanis

Pengujian pada sistem mekanis adalah pengujian berfungsi tidaknya perangkat-perangkat yang berhubungan dengan mekanis. Perangkat-perangkat tersebut meliputi motor DC dan motor servo.

3.5.3.1 Kriteria Pengujian Motor DC

Kriteria pengujian motor DC adalah ketika sistem menerima perintah untuk membuka pintu, maka motor DC akan berputar berlawanan arah jarum jam. Sedangkan ketika sistem menerima perintah untuk menutup pintu, maka motor DC akan berputar searah jarum jam. Selain itu, kriteria motor DC juga mengetahui tegangan *output* dari *driver* motor DC. Tabel 3.12 berikut ini merupakan kriteria dari motor DC.

Tabel 3.12 Kriteria Pengujian Motor DC

No.	Perintah	Kriteria Pengujian	Hasil Pengujian
1	Buka pintu	Motor DC berputar berlawanan arah jarum jam	
2	Tutup pintu	Motor DC berputar searah jarum jam	

Driver motor DC menggunakan tegangan *input* 5 Volt DC sebagai pengontrolnya dan 12 Volt DC sebagai penggerak motornya. Pada tabel 3.13 berikut ini merupakan kriteria pengujian tegangan motor DC.

Tabel 3.13 Kriteria Pengujian Tegangan Motor DC

No.	Kondisi Motor DC	Kriteria Pengukuran	Hasil Pengukuran
1	Bergerak	12 Volt DC	
2	Tidak bergerak	0 Volt DC	

3.5.3.2 Kriteria Pengujian Motor Servo

Motor servo merupakan sistem mekanis dari kunci pintu pagar. Kriteria pengujian motor servo adalah ketika pintu terkunci, posisi motor servo pada 0^o dan ketika pintu tidak terkunci, posisi motor servo pada 90^o. Kriteria pengujian motor servo dapat dilihat pada tabel 3.14 berikut ini.

Tabel 3.14 Kriteria Pengujian Motor Servo

No.	Kondisi Pintu Pagar	Kriteria Pengujian	Hasil Pengujian
1	Terkunci	Posisi servo pada 0^o	
2	Tidak Terkunci	Posisi servo pada 90^o	

3.5.4 Pengujian Tegangan *Output* Catu Daya

Catu daya merupakan sumber tegangan ke semua sistem. Dalam kriteria pengujiannya, catu daya yang menggunakan IC regulator 7805 maka tegangan *output*nya sebesar 5 Volt DC dan jika menggunakan IC regulator 7812 maka tegangan *output*nya sebesar 12 Volt DC. Pada tabel 3.15 ini adalah kriteria pengujian tegangan *output* pada catu daya.

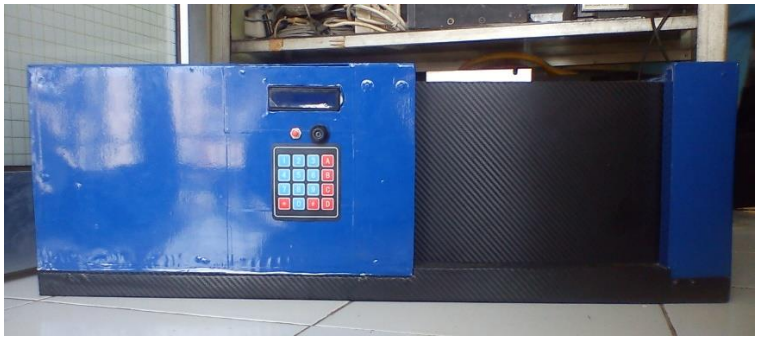
Tabel 3.15 Kriteria Pengujian Tegangan *Output* Catu Daya

No.	IC Regulator	Kriteria Pengukuran	Hasil Pengukuran
1	7805	5 Volt DC	
2	7812	12 Volt DC	

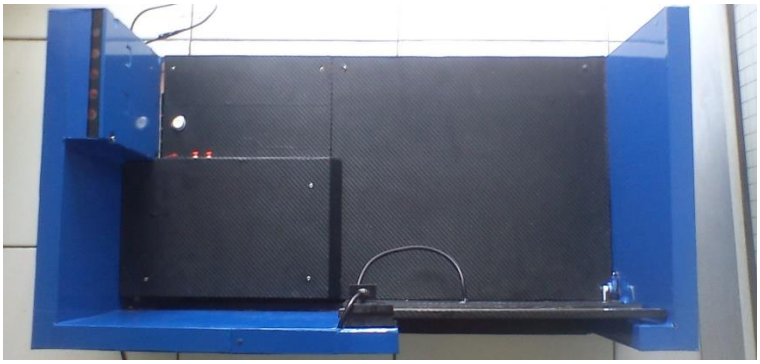
BAB IV
HASIL PENELITIAN DAN PEMBAHASAN

4.1 Hasil Penelitian

Sesuai dengan blok diagram beserta *flowchart* yang dijelaskan pada bab sebelumnya, maka prototipe sistem keamanan pintu pagar dengan multi-*input* berbasis arduino diimplementasikan oleh peneliti pada gambar 4.1 dan gambar 4.2.



Gambar 4.1 Prototipe Pintu Pagar (tampak depan)



Gambar 4.2 Prototipe Pintu Pagar (tampak atas)

4.1.1 Hasil Pengujian Sistem Pengendali

Pengujian sistem kendali yang telah dilakukan meliputi pengujian sistem *bluetooth*, pengujian sistem *keypad* dan pengujian sistem *voice recognition* dengan hasil sebagai berikut.

4.1.1.1 Hasil Pengujian Sistem *Bluetooth*

Pengujian yang dilakukan pada sistem *bluetooth* adalah pengujian jarak koneksi *bluetooth* dengan penghalang dan tanpa penghalang serta pengujian aplikasi android yang telah dibuat. Hasil dari pengujian jarak koneksi *bluetooth* dalapat dilihat pada tabel 4.1 berikut.

Tabel 4.1 Hasil Pengujian Jarak Koneksi *Bluetooth*

No.	Jarak (meter)	Kondisi	Kriteria Pengujian	Hasil Pengujian
1	2	Tanpa Halangan	Terkoneksi dengan perangkat *bluetooth*	Berhasil
		Ada Halangan		Berhasil
2	4	Tanpa Halangan		Berhasil
		Ada Halangan		Berhasil
3	6	Tanpa Halangan		Berhasil
		Ada Halangan		Berhasil
4	8	Tanpa Halangan		Berhasil
		Ada Halangan		Berhasil
5	10	Tanpa Halangan		Berhasil
		Ada Halangan		Tidak Berhasil
6	12	Tanpa Halangan		Tidak Berhasil
		Ada Halangan		Tidak Berhasil

Dalam pengujian aplikasi android, terlebih dahulu dilakukan pengkoneksian dengan perangkat *bluetooth* pada sistem yang akan dikendalikan, setelah itu, aplikasi android dapat digunakan.

Hasil dari pengujian aplikasi android dapat dilihat pada tabel 4.2 berikut ini.

Tabel 4.2 Hasil pengujian Aplikasi Android

No.	Tombol yang Ditekan	Kriteria Pengujian	Hasil Pengujian
1	*Icon Bluetooth*	Tampilan memilih perangkat *bluetooth* yang tersedia	Berhasil
2	*Open*	Pintu pagar terbuka	Berhasil
3	*Close*	Pintu pagar tertutup	Berhasil
4	*Open and Close Automatically*	Pintu pagar terbuka dan tertutup otomatis	Berhasil
5	ON	Sistem alarm aktif	Berhasil
6	OFF	Sistem alarm nonaktif	Berhasil
7	*Exit*	Keluar	Berhasil

4.1.1.2 Hasil Pengujian Pada Sistem *Keypad* 4x4

Pengujian pada *keypad* yaitu berupa *input*an *password* yang akan dicocokkan dengan *password* yang sudah diprogram sebelumnya sehingga jika *password* tidak sesuai, sistem tidak akan merespon. Hasil dari pengujian sistem *keypad* ini dapat dilihat pada tabel 4.3 berikut ini.

Tabel 4.3 Hasil Pengujian Sistem *Keypad*

No.	*Input Keypad*	Kriteria Pengujian	Hasil Pengujian
1	Memasukkan *password* “222222” lalu menekan “*”	Pintu pagar terbuka dan tertutup otomatis setelah terbuka penuh	Berhasil
2	Memasukkan *password* “222222” lalu menekan “#”	Mengganti *password* buka pintu	Berhasil
3	Memasukkan *password* “333333” lalu menekan “*” ketika sistem alarm aktif	Sistem alarm nonaktif	Berhasil

4	Memasukkan *password* "333333" lalu menekan "*" ketika sistem alarm nonaktif	Sistem alarm aktif	Berhasil
5	Memasukkan *password* "333333" lalu menekan "#"	Mengganti *password* sistem alarm	Berhasil

4.1.1.3 Hasil Pengujian Sistem *Voice Recognition*

Pengujian pada sistem *voice recognition* dilakukan dengan cara mengucapkan kata yang sesuai dengan kata yang telah diprogram. Jika hal tersebut berhasil, maka sistem akan merespon dimana *output* pada arduino uno dalam sistem *voice recognition* tersebut akan memberikan perintah ke arduino mega sebagai mikrokontroler utamanya, yaitu berupa logika *high* atau *low* yang akan diproses menjadi keluaran pada prototipe.

Untuk hasil dari pengujian sistem *voice recognition* tersebut, dapat dilihat pada tabel 4.4 berikut ini.

Tabel 4.4 Hasil Pengujian Sistem *Voice Recognition*

No.	**Kata Kunci**	***Output* Arduino Uno**	**Kriteria Pengujian**		**Hasil Pengujian**	
			Pada Prototipe	**Tegangan *Output* Arduino Uno**	**Pada prototipe**	**Tegangan *Output* Arduino Uno**
1	*"Open"*	Pin 12	Pintu pagar terbuka dan tertutup otomatis setelah terbuka penuh	5 Volt DC	Berhasil	4,86 Volt DC

4.1.2 Hasil Pengujian Sistem Keamanan

Pengujian pada sistem keamanan meliputi pengujian pada sensor getaran dan *photo diode*. Pada *photo diode*, tidak hanya diuji berfungsi atau tidaknya komponen tersebut, tetapi diuji juga nilai tegangan keluarannya.

4.1.2.1 Hasil Pengujian Sensor Getaran

Sensor getaran digunakan untuk mendeteksi adanya pembukaan pintu secara paksa oleh pihak yang tidak diinginkan. Hal terebut dipengaruhi juga dengan kondisi dari aktif atau tidaknya sistem alarm. Pada pengujian sensor getar, didapatkan hasil yang dipaparkan dalam tabel 4.5 berikut ini.

Tabel 4.5 Hasil Pengujian Sensor Getaran

No.	Sensor Getar	Kondisi Pendukung	Kriteria Pengujian	Hasil Pengujian
1	Deteksi getaran	Sistem alarm aktif	Alarm berbunyi	Berhasil
2	Deteksi getaran	Sistem alarm nonaktif	Alarm tidak berbunyi	Berhasil

4.1.2.2 Hasil Pengujian *Photo Diode*

Photo diode yang digunakan sebagai sensor cahaya dalam penelitian ini berjumlah dua buah dengan fungsi masing-masing yang berbeda. *Photo diode* 1 diuji untuk mendeteksi penghalang pada jalur pintu ketika pintu sedang menutup yang dimaksudkan untuk menjaga keamanan pengguna agar tidak terjepit oleh pintu pagar. Sedangkan *photo diode* 2 diuji untuk mendeteksi orang asing yang memanjat tembok untuk memaksa masuk tanpa melewati pintu pagar. Hasil dari

pengujian tersebut dapat dilihat pada tabel 4.6 berikut dengan kondisi pendukungnya.

Tabel 4.6 Hasil Pengujian *Photo Diode*

No.	Kondisi Sensor	Kondisi Pendukung	Kriteria Pengujian	Hasil Pengujian
1	*Photo diode* 1 menerima cahaya dari led *super bright*	Pintu sedang menutup	Pintu terus menutup	Berhasil
2	*Photo diode* 1 tidak menerima cahaya dari led *super bright*	Pintu sedang menutup	Pintu berhenti	Berhasil
3	*Photo diode* 2 menerima cahaya dari led *super bright*	Sistem alarm aktif	Alarm tidak berbunyi	Berhasil
4	*Photo diode* 2 menerima cahaya dari led *super bright*	Sistem alarm nonaktif	Alarm tidak berbunyi	Berhasil
5	*Photo diode* 2 tidak menerima cahaya dari led *super bright*	Sistem alarm aktif	Alarm berbunyi	Berhasil
6	*Photo diode* 2 tidak menerima cahaya dari led *super bright*	Sistem alarm nonaktif	Alarm tidak berbunyi	Berhasil

4.1.2.3 Hasil Pengujian Tegangan *Output Photo Diode*

Selain diuji berfungsi atau tidaknya, *photo diode* juga diuji tegangan keluarannya. Hasil dari pengujian tegangan *photo diode* dapat dilihat pada tabel 4.7 berikut ini.

Tabel 4.7 Hasil Pengujian Tegangan *Output Photo Diode*

No.	Kondisi Sensor	Kriteria Pengukuran	Hasil Pengukuran
1	*Photo diode* 1 menerima cahaya	0 Volt DC	0,87-3,61 Volt DC
2	*Photo diode* 1 tidak menerima cahaya	5 Volt DC	4,85 Volt DC
3	*Photo diode* 2 menerima cahaya	0 Volt DC	3,62 Volt DC
4	*Photo diode* 2 tidak menerima cahaya	5 Volt DC	4,66 Volt DC

4.1.3 Hasil Pengujian Sistem Mekanis

Pengujian pada sistem mekanis yaitu pengujian pada komponen-komponen pendukung mekanis meliputi motor DC sebagai penggerak utama pintu pagar dan motor servo sebagai sistem pengunci pintu pagar.

4.1.3.1 Hasil Pengujian Motor DC

Motor DC diuji berdasarkan fungsinya dimana motor DC harus dapat berputar dua arah. Pada tabel 4.8 merupakan hasil dari pengujian motor DC.

Tabel 4.8 Hasil Pengujian Motor DC

No.	Perintah	Kriteria Pengujian	Hasil Pengujian
1	Buka pintu	Motor DC berputar berlawanan arah jarum jam	Berhasil
2	Tutup pintu	Motor DC berputar searah jarum jam	Berhasil

4.1.3.2 Hasil Pengujian Tegangan *Output* Motor DC

Tegangan keluaran yang diukur merupakan tegangan keluaran pada *driver* motor DC dimana tegangan tersebut yang akan dihubungkan langsung dengan motor DC. Hasil dari pengujian tegangan motor DC dapat dilihat pada tabel 4.9 berikut ini.

Tabel 4.9 Hasil Pengujian Tegangan *Output* Motor DC

No.	Kondisi Motor DC	Kriteria Pengukuran	Hasil Pengukuran
1	Bergerak	12 Volt DC	11,90 Volt DC
2	Tidak bergerak	0 Volt DC	0 Volt DC

4.1.3.3 Hasil Pengujian Motor Servo

Motor servo sebagai sistem pengunci pintu pagar diuji menurut fungsinya, apakah motor servo tersebut bekerja sesuai dengan perencanaan atau tidak. Hasil pengujian tersebut dapat dilihat pada tabel 4.10.

Tabel 4.10 Hasil Pengujian Motor Servo

No.	Kondisi Pintu Pagar	Kriteria Pengujian	Hasil Pengujian
1	Terkunci	Posisi servo pada 0^o	Berhasil
2	Tidak Terkunci	Posisi servo pada 90^o	Berhasil

4.1.4 Hasil Pengujian Tegangan *Output* Catu Daya

Tegangan keluaran dari catu daya atau sumber tegangan yang digunakan dalam penelitian ini adalah 5 Volt DC dan 12 Volt DC. Pengujian ini dilakukan untuk menentukan tegangan keluaran dari catu daya sesuai atau tidaknya dengan perencanaan.

Hasil dari pengujian tegangan keluaran tersebut dapat dilihat pada tabel 4.11 berikut ini.

Tabel 4. 11 Hasil Pengujian Tegangan *Output* Catu Daya

No.	IC Regulator	Kriteria Pengukuran	Hasil Pengukuran
1	7805	5 Volt DC	4,99 Volt DC
2	7812	12 Volt DC	12,02 Volt DC

4.2 Pembahasan

Setelah hasil dari keseluruhan pengujian dilakukan, maka prototipe sistem keamanan pintu pagar ini dapat dikatakan sesuai dengan perencanaan, tetapi ada beberapa keterbatasan pada prototipe pintu pagar ini. Berikut ini adalah pembahasan-pembahasan pada hasil yang didapat.

Pembahasan pertama yaitu pada sistem pengendali yang meliputi sistem *bluetooth*, sistem *keypad* dan sistem *voice recognition.* Pada pengujian *bluetooth*, terdapat keterbatasan yaitu jarak koneksi *bluetooth* hanya sampai 10 meter dengan tanpa halangan. Sesuai dengan hasil pada tabel 4.1, jika jarak koneksi sejauh 10 meter atau lebih dan ada penghalang, maka *bluetooth* pada android tidak dapat terkoneksi dengan perangkat *bluetooth* pada prototipe pintu pagar. Hal tersebut juga mempengaruhi pada pengujian aplikasi android sebagai pengontrol pintu pagar dimana pada saat pengujian aplikasi android tersebut harus pada jarak kurang dari 10 meter. Di luar keterbatasan itu, semua fungsi dari aplikasi android yang telah dibuat, berjalan sesuai dengan perencanaan sebelumnya berdasarkan hasil pada tabel 4.2.

Selanjutnya yaitu hasil pada pengujian sistem *keypad* yang terdapat pada tabel 4.3. Berdasarkan hasil tersebut, sistem *keypad* sebagai *input*an *password* pada prototipe pintu pagar berjalan dengan baik sesuai dengan perencanaan. Bahkan *password* dapat diganti langsung dari *input*an *keypad* itu sendiri tanpa memprogram ulang.

Berdasarkan pada tabel 4.4, hasil dari pengujian sistem *voice recognition* berjalan dengan baik, akan tetapi ada keterbatasan, yaitu sistem sulit mendeteksi kata yang diucapkan ketika kondisi luar sangat bising sehingga perlu dilakukan beberapa kali sampai sistem mendeteksi kata yang sesuai dengan perencanaan.

Pembahasan kedua yaitu pembahasan pada hasil pengujian sistem keamanan yang meliputi hasil pengujian sensor getaran dan *photo diode* atau sensor cahaya. Berdasarkan hasil pengujian pada tabel 4.5, sensor getaran berfungsi dengan baik, ketika mendeteksi getaran dan sistem alarm dalam keadaan aktif, alarm langsung berbunyi sesuai dengan perencanaan. Selanjutnya hasil pengujian pada tabel 4.6, yaitu hasil pengujian fungsi dari sensor cahaya atau *photo diode*. *Photo diode* 1 sebagai pendeteksi penghalang jalur pintu pagar ketika pintu menutup berhasil mendeteksi jika ada penghalang orang ataupun kendaraan di jalur pintu dengan indikasi pintu akan berhenti bergerak agar orang atau kendaraan yang belum sepenuhnya melewati pintu, tidak akan terjepit. Pada *photo diode* 2 juga berhasil mendeteksi ketika ada yang melewati bagian atas tembok pintu pagar sesuai dengan perencanaan.

Selain diuji fungsinya, *photo diode* juga diuji tegangan keluarannya yang ditunjukkan pada tabel 4.7 dimana hasil pengukuran tegangan keluaran pada *photo diode* 1 memiliki nilai antara 0,86-3,61 Volt DC ketika terkena cahaya karena

intensitas cahaya yang diterima *photo diode* berubah-ubah sesuai dengan lebar pintu yang terbuka sehingga memiliki batas atas dan bawah pada tegangan keluarannya. Sedangkan pada hasil pengukuran tegangan keluaran *photo diode* 2 ketika menerima cahaya, memiliki nilai tegangan yang jauh lebih besar dibandingkan dengan kriteria pengukurannya karena jauhnya jarak sensor cahaya.

Pembahasan ketiga yaitu pembahasan hasil dari pengujian pada sistem mekanis yang meliputi hasil pengujian pada motor DC dan motor servo. Hasil dari pengujian motor DC ditunjukkan pada tabel 4.8 dimana pengujian yang dilakukan adalah sesuai atau tidaknya arah perputaran motor berdasarkan perintah yang dikirim dari sistem dan dapat dilihat bahwa hasil dari pengujian sesuai dengan perencanaan. Pengujian pada tegangan motor DC pun sesuai dengan perencanaan jika dilihat pada tabel 4.9. Selanjutnya hasil pengujian pada motor servo dimana fungsinya sebagai pengunci pada pintu pagar. Jika dilihat pada tabel 4.10, hasil dari pengujian motor servo berfungsi sesuai dengan perencanaan.

Pembahasan terakhir yaitu pembahasan pada hasil pengujian tegangan keluaran pada rangkaian catu daya. Dalam penelitian ini, rangkaian catu daya mengguanakan IC regulator 7805 dan 7812 sehingga memiliki 2 keluaran, yaitu 5 Volt DC dan 12 Volt DC. Berdasarkan hasil pengujian pada tabel 4.11, tegangan keluaran pada rangkaian cau daya sesuai dengan perencanaan.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan perencanaan sampai dengan hasil dalam penelitian prototipe sistem keamanan pintu pagar dengan multi-*input* berbasis arduino ini, dapat disimpulkan bahwa:

1. Prototipe sistem keamanan pintu pagar dengan multi-*input* berbasis arduino telah sesuai dengan perencanaan dan tujuan dari penelitian. Prototipe memiliki sistem pengendali dan sistem keamanan serta keseluruhan *input* dapat dikombinasikan menjadi sistem multi-*input*.
2. Sistem pengendali dalam prototipe pintu pagar ini adalah beberapa *input*an meliputi sistem *bluetooth* dengan pengendali android, sistem *keypad* dengan *input password* dan sistem *voice recognition* dengan *input* kata atau ucapan. Hal tersebut dimaksudkan untuk membatasi akses masuk pada pintu pagar. Pada sistem *keypad*, *password* yang digunakan dapat diganti langsung melalui *keypad* itu sendiri tanpa diprogram ulang.
3. Sistem keamanan pada prototipe pintu pagar ini menggunakan sensor getaran sebagai pendeteksi getaran pada pintu pagar dimana getaran tersebut disebabkan oleh orang yang masuk tanpa ijin dengan cara memaksa memdorong dan mendobrak pintu pagar tersebut agar terbuka.

Pada bagian atas ditempatkan sensor cahaya sebagai pendeteksi orang yang masuk dengan cara memanjat. Jika ada orang yang masuk dan terdeteksi sensor tersebut, alarm akan berbunyi. Sensor cahaya juga ditempatkan pada jalur pintu untuk mendeteksi penghalang, sehingga tidak akan terjadi kecelakaan terjepit pintu ketika pintu pagar meunutup.

5.2 Saran

Dalam penelitian prototipe sistem keamanan pintu pagar ini, pasti memiliki beberapa kekurangan. Berdasarkan hasil dari penelitian dan kesimpulan, terdapat beberapa saran untuk pengembangan pada penelitian ini, yaitu sebagai berikut:

1. Menggunakan perangkat sistem pengendali nirkabel yang jarak koneksinya dapat lebih jauh dari *bluetooth*.
2. Penggunaan modul *voice recognition* yang lebih sensitif dan tidak terlalu terpengaruh oleh kebisingan di sekitar.
3. Menambah kamera sebagai pengenal wajah (*Face Detector*).
4. Menambah indikasi selain alarm pada sistem keamanan.

.

DAFTAR PUSTAKA

Abelson, H. *Teaching with App Inventor for Android*. ACM Digital Library. Diambil dari: http://dl.acm.org/citation.cfm?id=2157437 (2015, April 25).

Banzi, M. 2008. *Getting Started with Arduino.* USA: O'Reilly.

Baumann, J. *Voice Reognition.* Human Interface Technology Laboratory. Diambil dari: http://www.hitl.washington.edu/research/knowledge_base/virtual-worlds/EVE/I.D.2.d.VoiceRecognition.html (2015, April 28).

Borg, W. R. 1989. *Educational Research: An Inroduction, Fifth Edition.* New York: Longman.

Chin-Hui Lee, F. K. 1996. *Automatic Speech and Speaker Recognation Advanced Topics.* USA: Kluwer Academic Publisher.

Chogwang. *Pengertian dan Jenis Sensor*. Chogwang. Diambil dari: http://www.chogwang.com/2014/10/pengertian-dan-jenis-sensor.html (2015, April 27).

DC Motors-Speed Controls-Servo Systems. 1977. USA: Electro-Craft Corporation.

DiStefano, J. J. 1990. *Schaum's Outline of Feedback and Control System 2nd Edition.* Los Angeles: McGraw-Hill.

[FT] Fakultas Teknik. 2012. *Buku Pedoman Skripsi/ Komprehensif/Karya Inovatif (S1)*. Jakarta: Fakultas Teknik, Universitas Negeri Jakarta.

LCD (Liquid Crystal Display). Elektronika Dasar. Diambil dari: http://elektronika-dasar.web.id/teori-elektronika/lcd-liquid-cristal-display/ (2015, April 27).

Matrix Keypad 4x4 Untuk Mikrokontroler. Elektronika Dasar. Diambil dari: http://elektronika-dasar.web.id/artikel-elektronika/matrix-keypad-4x4-untuk-mikrokontroler/ (2015, Maret 22).

Ruocco, S. R. (1987). *Robot Sensors and Transducers.* England: Halsted.

Sensor Cahaya. Zona Elektro. Diambil dari: http://zonaelektro.net/sensor-cahaya/ (2015, April 27).

Sensor Getaran. Komponen Elektronika. Diambil dari: http://www.komponenelektronika.biz/sensor-getaran.html (2015, April 25).

Specification of the Bluetooth System. 2003. Bluetooth SIG.

Spesification of the Bluetooth Systems. 1999. Bluetooth SIG.

Syahwil, Muhammad. 2013. *Panduan Mudah Simulasi & Praktek Mikrokontroler Arduino.* Yogyakarta: Andi Offset.

Turan, M. S. 2010. *Recommendation for Password-Based Key Derivation.* USA: NIST Special Publication.

Wiryadinata, R.. *Prinsip Kerja Accelerometer*. Wiryadinata. Diambil dari http://wiryadinata.web.id/?p=22 (2015, April 25)

Wolber, D. 2011. *App Inventor, Create Your Own Android Apps.* Canada: O'Reilly.

LAMPIRAN

Lampiran 1 List Program Arduino

1. List Program Pada Arduino Mega 2560

```
/*
           SKRIPSI
   PROTOTIPE SISTEM KEAMANAN
PINTU PAGAR DENGAN MULTI-INPUT
       BERBASIS ARDUINO

  NAMA : DEDEN LUTHFI FERMANA
     NO. REG : 5215116399
   PEND. TEKNIK ELEKTRONIKA
  UNIVERSITAS NEGERI JAKARTA
*/

#include <LiquidCrystal.h>
#include <Keypad.h>
#include <Servo.h>
#include <SoftwareSerial.h>
#include <Wire.h>
#include <ADXL345.h>

Servo owl;
int pos;
const int resetPin = 44;
int tx       = 1;
int rx       = 0;
int buka     = 30;
int tutup    = 31;
int voice    = 12;
int alarm    = 53;
int alarmpow = 52;
int standby  = 51;
int ledtutup = 50;
int ledbuka  = 49;
int ledkunci = 48;
int ledalarm = 47;
int tombol1  = A0;
int tombol2  = A1;
int limit1   = A2;
int limit2   = A3;
int sencah   = A4;
int sencah1  = A5;
int password = 100;
int photo    = 200;
int keamanan = 100;
int bukaonly = 100;
int resetState = 0;
int tombolA, tombolB, limitA,
limitB, sencahA, sencahB, vrm;

char inSerial[15];
SoftwareSerial bluetooth(tx,
rx);
LiquidCrystal lcd(7, 6, 5, 4,
3, 2);
const byte ROWS = 4;
const byte COLS = 4;
const int nada[8]={

131,175,262,349,523,698,1047,1
397};
char keys[ROWS][COLS] = {
  {
    '1','2','3','A'        }
  ,
  {
    '4','5','6','B'        }
  ,
  {
    '7','8','9','C'        }
  ,
  {
    '*','0','#','D'        }
};
char berita[16]={
  ' ',' ',' ',' ',
  ' ',' ',' ',' ',
  ' ',' ',' ',' ',
  ' ',' ',' ',' '};
char tampil[16]={
  ' ',' ',' ',' ',
  ' ',' ',' ',' ',
  ' ',' ',' ',' ',
  ' ',' ',' ',' '};
byte rowPins[ROWS] = {
  23,25,27,29};
byte colPins[COLS] = {
  28, 26, 24, 22};
Keypad keypad = Keypad(
makeKeymap(keys), rowPins,
colPins, ROWS, COLS );
byte k=4,n=0;

String readString, jawab,
unlock="222222",
aktif="333333";
ADXL345 adxl;

void setup() {
  Serial.begin(9600);
```

```
  pinMode(voice,    INPUT);
  pinMode(tombol1,  INPUT);
  pinMode(tombol2,  INPUT);
  pinMode(limit1,   INPUT);
  pinMode(limit2,   INPUT);
  pinMode(sencah,   INPUT);
  pinMode(sencah1,  INPUT);
  pinMode(buka,    OUTPUT);
  pinMode(tutup,   OUTPUT);
  pinMode(tx,      OUTPUT);
  pinMode(rx,       INPUT);
  pinMode(alarm,   OUTPUT);
  pinMode(alarmpow,OUTPUT);
  pinMode(standby, OUTPUT);
  pinMode(ledtutup,OUTPUT);
  pinMode(ledbuka, OUTPUT);
  pinMode(ledkunci,OUTPUT);
  pinMode(ledalarm,OUTPUT);
  pinMode(8,       OUTPUT);
  pinMode(32,      OUTPUT);
  pinMode(33,      OUTPUT);
  pinMode(46,      OUTPUT);
  pinMode(resetPin, INPUT);
  owl.attach(9);
  pos=0;
  owl.write(pos);
  digitalWrite(46,      HIGH);
  digitalWrite(32,      HIGH);
  digitalWrite(33,       LOW);
  digitalWrite(buka,     LOW);
  digitalWrite(tutup,    LOW);
  digitalWrite(alarm,    LOW);
  digitalWrite(standby,  LOW);
  delay(1000);
  digitalWrite(standby, HIGH);
  digitalWrite(ledtutup,
HIGH);
  delay(1000);
  digitalWrite(ledtutup,
LOW);
  digitalWrite(ledkunci,HIGH);
  delay(1000);
  digitalWrite(ledkunci, LOW);
  digitalWrite(ledbuka,HIGH);
  delay(1000);
  digitalWrite(ledbuka, LOW);
  digitalWrite(ledalarm,HIGH);
  delay(1000);
  digitalWrite(ledalarm, LOW);
  lcd.begin(16, 2);
  lcd.print("INPUT
PASSWORD:");
  lcd.setCursor(0, 1);
  lcd.print("..GATE
LOCKED..");
  kunci_pintu();

adxl.powerOn();
  //set activity/ inactivity
thresholds (0-255)

adxl.setActivityThreshold(75);
//62.5mg per increment

adxl.setInactivityThreshold(75
); //62.5mg per increment
  adxl.setTimeInactivity(10);
// how many seconds of no
activity is inactive?
  //look of activity movement
on this axes - 1 == on; 0 ==
off
  adxl.setActivityX(1);
  adxl.setActivityY(1);
  adxl.setActivityZ(1);
  //look of inactivity
movement on this axes - 1 ==
on; 0 == off
  adxl.setInactivityX(1);
  adxl.setInactivityY(1);
  adxl.setInactivityZ(1);
  //setting all interupts to
take place on int pin 1
  //I had issues with int pin
2, was unable to reset it
  adxl.setInterruptMapping(
ADXL345_INT_ACTIVITY_BIT,
ADXL345_INT1_PIN );
  adxl.setInterruptMapping(
ADXL345_INT_INACTIVITY_BIT,
ADXL345_INT1_PIN );
  //register interupt actions
- 1 == on; 0 == off
  adxl.setInterrupt(
ADXL345_INT_ACTIVITY_BIT,
1);
  adxl.setInterrupt(
ADXL345_INT_INACTIVITY_BIT,
1);
}

void (*pseudoReset) (void)=0;

void baca_input()
{
  tombolA =
analogRead(tombol1);
  tombolB =
analogRead(tombol2);
  limitA  =
analogRead(limit1);
  limitB  =
analogRead(limit2);
  sencahA =
analogRead(sencah);
  sencahB =
analogRead(sencah1);
```

```
  vrm     =
digitalRead(voice);

//  Serial.print(tombolA);
//  Serial.print(" ");
//  Serial.print(tombolB);
//  Serial.print(" ");
//  Serial.print(limitA);
//  Serial.print(" ");
//  Serial.print(limitB);
//  Serial.print(" ");
//  Serial.print(sencahA);
//  Serial.print(" ");
//  Serial.println(sencahB);

 if (password==100 &&
limitA<=60)
 {
    lcd.setCursor(0, 0);
    lcd.print("INPUT
PASSWORD:");
  }

  if (password==100 &&
limitA>=60 && limitB>=60)
  {
    lcd.setCursor(0, 0);
    lcd.print("..PLEASE
WAIT..");
  }

  if (password==200)
  {
    lcd.setCursor(0, 0);
    lcd.print("PASS FOR OPEN
:");
  }

  if (password==300)
  {
    lcd.setCursor(0, 0);
    lcd.print("PASS FOR
ALARM:");
  }

  if (vrm==HIGH)
  {
    keamanan=200;
    bukaonly=100;
    lcd.setCursor(0, 1);
    lcd.print("COMMAND
RECEIVED");
    buka_pintu();
  }

  if (sencahB>=900 &&
keamanan==100)
  {
    digitalWrite(alarm,
HIGH);
    digitalWrite(alarmpow,
HIGH);
    digitalWrite(ledalarm,
HIGH);
  }

  if (tombolA<=60)
  {
    keamanan=200;
    photo=200;
    bukaonly=200;
    lcd.setCursor(0, 1);
    lcd.print("COMMAND
RECEIVED");
    buka_pintu();
  }

  if (tombolB<=60)
  {
    keamanan=200;
    lcd.setCursor(0, 1);
    lcd.print("COMMAND
RECEIVED");
    tutup_pintu();
  }

  if (limitB<=60 &&
bukaonly==100)
  {
    tutup_pintu();
  }

  if (limitB<=60 &&
bukaonly==200)
  {
    buka_only();
  }

  if (limitA<=60 &&
photo==300)
  {
    kunci_pintu();
    bukaonly=200;
    keamanan=100;
  }

  if (sencahA>=920 &&
photo==300)
  {
    jeda();
  }

  if (sencahA<=920 &&
photo==300)
  {
    tutup_pintu();
  }
```

```
}

void baca_bluetooth()
{
 while (Serial.available())
  {
    delay (3);
    char c = Serial.read();
    readString += c;
  }
  if (readString.length() >0)
  {

Serial.println(readString);
    if (readString == "buka")
    {
      lcd.setCursor(0, 1);
      lcd.print("COMMAND
RECEIVED");
      bukaonly=100;
      buka_pintu();
    }

    if (readString ==
"bukaonly")
    {
      lcd.setCursor(0, 1);
      lcd.print("COMMAND
RECEIVED");
      bukaonly=200;
      buka_pintu();
    }

    if (readString == "tutup")
    {
      lcd.setCursor(0, 1);
      lcd.print("COMMAND
RECEIVED");
      tutup_pintu();
    }

    if (readString ==
"alarmon")
    {
      keamanan=100;
      lcd.print("
");
      lcd.setCursor(0, 1);
      lcd.print("  ALARM IS ON
");
      delay(2000);
      lcd.setCursor(0, 1);
      lcd.print("
");
    }

    if (readString ==
"alarmoff")
    {
      keamanan=200;
      lcd.setCursor(0, 0);
      lcd.print("
");
      lcd.setCursor(0, 1);
      lcd.print("  ALARM IS
OFF  ");
      delay(3000);
      lcd.setCursor(0, 1);
      lcd.print("
");
    }
    readString ="";
  }
}

void baca_keypad()
{
  char key = keypad.getKey();
  if (key != NO_KEY)
  {
    berita[n]=key;
    tampil[n]='*';
    tone(8, nada[k],250);
    n++;
    if(n>15)
    {

for(n=1;n<=15;n++)berita[n-
1]=berita[n];
      n=15;
      berita[n]=' ';
    }
    lcd.setCursor(n-1, 1);
    lcd.print(key);
    delay(500);
    lcd.setCursor(0, 1);
    lcd.print(tampil);
    noTone(4);
  }

  if(key=='*' &&
password==200)
  {

for(n=0;n<6;n++)jawab+=berita[
n];
    lcd.setCursor(0,1);
    lcd.print("PASSWORD
CHANGED");
    delay(2000);
    unlock=jawab;
    password=100;
    for(n=0;n<=15;n++)
    {
      berita[n]=' ';
      tampil[n]=' ';
    }
    lcd.setCursor(0, 1);
```

```
    lcd.print(tampil);
    n=0;
    jawab=String();
  }

  if(key=='*' &&
password==300)
  {

for(n=0;n<6;n++)jawab+=berita[
n];
    lcd.setCursor(0,1);
    lcd.print("PASSWORD
CHANGED");
    delay(2000);
    aktif=jawab;
    password=100;
    for(n=0;n<=15;n++)
    {
      berita[n]=' ';
      tampil[n]=' ';
    }
    lcd.setCursor(0, 1);
    lcd.print(tampil);
    n=0;
    jawab=String();
  }

  if(key=='*' &&
password==100)
  {

for(n=0;n<6;n++)jawab+=berita[
n];
    if(jawab==unlock)
    {
      lcd.setCursor(0, 1);
      lcd.print("ACCESS
GRANTED!!");
      delay(1000);
      lcd.setCursor(0, 1);
      lcd.print("
");
      bukaonly=100;
      buka_pintu();
    }

    else if(jawab==aktif &&
keamanan==100)
    {
      keamanan=200;
      lcd.setCursor(0, 1);
      lcd.print("  ALARM IS
OFF  ");
      delay(2000);
      lcd.setCursor(0, 1);
      lcd.print("
");
    }

    else if(jawab==aktif &&
keamanan==200)
    {
      keamanan=100;
      lcd.setCursor(0, 1);
      lcd.print("  ALARM IS ON
");
      delay(2000);
      lcd.setCursor(0, 1);
      lcd.print("
");
    }

    else if(jawab!=unlock &&
jawab!=aktif)
    {
      lcd.setCursor(0, 1);
      lcd.print("WRONG
PASSWORD!!");
      tone(10, nada[1],500);
      delay(100);
    }
    delay(1500);
    for(n=0;n<=15;n++)
    {
      berita[n]=' ';
      tampil[n]=' ';
    }
    lcd.setCursor(0, 1);
    lcd.print(tampil);
    n=0;
    jawab=String();
  }

  if(key=='#')
  {

for(n=0;n<6;n++)jawab+=berita[
n];
    if(jawab==unlock)
    {
      lcd.clear();
      lcd.setCursor(0,0);
      lcd.print("PASS FOR OPEN
: ");
      lcd.setCursor(0, 0);
      lcd.print("
");
      password=200;
    }

    else if(jawab==aktif)
    {
      lcd.clear();
      lcd.setCursor(0,0);
      lcd.print("PASS FOR
ALARM :");
      lcd.setCursor(0, 0);
```

```
      lcd.print("                ");
      password=300;
    }

    else if(jawab!=unlock && jawab!=aktif)
    {
      lcd.setCursor(0, 1);
      lcd.print("WRONG PASSWORD!!");
      tone(10, nada[1],500);
      delay(100);
    }
    delay(1500);
    for(n=0;n<=15;n++)
    {
      berita[n]=' ';
      tampil[n]=' ';
    }
    lcd.setCursor(0, 1);
    lcd.print(tampil);
    n=0;
    jawab=String();
  }
  delay(100);
}

void baca_getar()
{
  int x,y,z;
  adxl.readAccel(&x, &y, &z);
  byte interrupts = adxl.getInterruptSource();
  //inactivity

if(adxl.triggered(interrupts, ADXL345_INACTIVITY))
  {
    digitalWrite(alarm, LOW);
    digitalWrite(ledalarm, LOW);
    digitalWrite(alarmpow, LOW);
  }
  //activity

if(adxl.triggered(interrupts, ADXL345_ACTIVITY) && limitA<=60 && keamanan==100)
  {
    digitalWrite(alarm, HIGH);
    digitalWrite(alarmpow, HIGH);
    digitalWrite(ledalarm, HIGH);
  }
}

void buka_pintu()
{
  photo=200;
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("PLEASE WAIT....");
  delay(1000);
  lcd.setCursor(0, 1);
  lcd.print("OPEN  THE GATE");
  pos = 90;
  owl.write(pos);
  digitalWrite(ledkunci,LOW);
  delay(1000);
  digitalWrite(tutup,    LOW);
  digitalWrite(ledtutup, LOW);
  digitalWrite(buka,    HIGH);
  digitalWrite(ledbuka, HIGH);
}

void buka_only()
{
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("                ");
  lcd.setCursor(0, 1);
  lcd.print("..GATE OPENED..");
  digitalWrite(buka,    LOW);
  digitalWrite(ledbuka, LOW);
  digitalWrite(tutup,   LOW);
  digitalWrite(ledtutup,LOW);
}

void tutup_pintu()
{
  photo=300;
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("PLEASE WAIT....");
  lcd.setCursor(0, 1);
  lcd.print("CLOSE  THE GATE");
  digitalWrite(buka,     LOW);
  digitalWrite(ledbuka,  LOW);
  digitalWrite(ledtutup,HIGH);
  digitalWrite(tutup,   HIGH);
}

void kunci_pintu()
{
  photo=200;
  delay(500);
  digitalWrite(buka,     LOW);
```

```
  digitalWrite(tutup,    LOW);
  digitalWrite(ledbuka,  LOW);
  digitalWrite(ledtutup, LOW);
  digitalWrite(ledkunci,HIGH);
  delay(1000);
  pos = 0;
  owl.write(pos);
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("
");
  lcd.setCursor(0, 1);
  lcd.print("..GATE
LOCKED..");
  bukaonly=200;
}

void jeda()
{
  digitalWrite(buka,    LOW);
  digitalWrite(tutup,   LOW);
  digitalWrite(ledbuka, LOW);
  digitalWrite(ledtutup,LOW);
  lcd.setCursor(0, 1);
  lcd.print("  GATE STOPPED
");
  delay(200);
}

void loop()
{
  resetState =
digitalRead(resetPin);
  if (resetState==HIGH)
  {
    pseudoReset();
  }
  baca_input();
  baca_bluetooth();
  baca_keypad();
  baca_getar();
}
```

2. List Program Arduino Uno

```
/*
            SKRIPSI
   PROTOTIPE SISTEM KEAMANAN
PINTU PAGAR DENGAN MULTI-INPUT
       BERBASIS ARDUINO

  NAMA : DEDEN LUTHFI FERMANA
     NO. REG : 5215116399
   PEND. TEKNIK ELEKTRONIKA
  UNIVERSITAS NEGERI JAKARTA
*/

#include <SoftwareSerial.h>
#include
"VoiceRecognitionV3.h"

/**
  Connection
  Arduino
VoiceRecognitionModule
   2   ------->     TX
   3   ------->     RX
*/
VR myVR(2,3);

uint8_t records[7]; // save
record
uint8_t buf[64];

int buka     = 12;
const int resetPin = 11;
int resetState = 0;

#define buka0 (0)
#define buka1 (1)
#define buka2 (2)
#define buka3 (3)
/**
  @brief   Print signature, if
the character is invisible,
           print hexible value
instead.
  @param   buf     --> command
length
           len     --> number
of parameters
*/
void printSignature(uint8_t
*buf, int len)
{
  int i;
  for(i=0; i<len; i++){
    if(buf[i]>0x19 &&
buf[i]<0x7F){
      Serial.write(buf[i]);
    }
    else{
      Serial.print("[");
      Serial.print(buf[i],
HEX);
      Serial.print("]");
    }
  }
}
```

```
/**
  @brief   Print signature, if
the character is invisible,
           print hexible value
instead.
  @param   buf  -->  VR module
return value when voice is
recognized.
             buf[0]  -->
Group mode(FF: None Group,
0x8n: User, 0x0n:System
             buf[1]  -->
number of record which is
recognized.
             buf[2]  -->
Recognizer index(position)
value of the recognized
record.
             buf[3]  -->
Signature length
             buf[4]~buf[n] -->
Signature
*/
void printVR(uint8_t *buf)
{
  Serial.println("VR
Index\tGroup\tRecordNum\tSigna
ture");

  Serial.print(buf[2], DEC);
  Serial.print("\t\t");

  if(buf[0] == 0xFF){
    Serial.print("NONE");
  }
  else if(buf[0]&0x80){
    Serial.print("UG ");

Serial.print(buf[0]&(~0x80),
DEC);
  }
  else{
    Serial.print("SG ");
    Serial.print(buf[0], DEC);
  }
  Serial.print("\t");

  Serial.print(buf[1], DEC);
  Serial.print("\t\t");
  if(buf[3]>0){
    printSignature(buf+4,
buf[3]);
  }
  else{
    Serial.print("NONE");
  }
  Serial.println("\r\n");
}

void setup()
{
  /** initialize */
  myVR.begin(9600);

  Serial.begin(115200);
  Serial.println("Elechouse
Voice Recognition V3
Module\r\nControl LED
sample");

  pinMode(buka, OUTPUT);
  pinMode(resetPin, INPUT);
  digitalWrite(buka, LOW);

  if(myVR.clear() == 0){
    Serial.println("Recognizer
cleared.");
  }else{
    Serial.println("Not find
VoiceRecognitionModule.");
    Serial.println("Please
check connection and restart
Arduino.");
    while(1);
  }

  if(myVR.load((uint8_t)buka0)
>= 0){
    Serial.println("buka0
loaded");
  }

  if(myVR.load((uint8_t)buka1)
>= 0){
    Serial.println("buka1
loaded");
  }

  if(myVR.load((uint8_t)buka2)
>= 0){
    Serial.println("buka2
loaded");
  }

  if(myVR.load((uint8_t)buka3)
>= 0){
    Serial.println("buka3
loaded");
  }
}

void (*pseudoReset) (void)=0;

void loop()
{
  resetState =
digitalRead(resetPin);
  if (resetState==HIGH)
```

```
  {
    pseudoReset();
  }
  int ret;
  ret = myVR.recognize(buf,
50);
  if(ret>0){
    switch(buf[1]){
      case buka0:
        digitalWrite(buka,
HIGH);
        delay(5000);
        digitalWrite(buka,
LOW);
        break;
      case buka1:
        digitalWrite(buka,
HIGH);
        delay(5000);
        digitalWrite(buka,
LOW);
        break;
      case buka2:
        digitalWrite(buka,
HIGH);
        delay(5000);
        digitalWrite(buka,
LOW);
        break;
      case buka3:
        digitalWrite(buka,
HIGH);
        delay(5000);
        digitalWrite(buka,
LOW);
        break;

      default:
        Serial.println("Record
function undefined");
        break;
    }
    /** voice recognized */
    printVR(buf);
  }
}
```

3. List Program Arduino Uno Untuk Merekam *Voice Recognition*

```
/**

**********************************
**********************************
******************
 * @file
vr_sample_train.ino
 * @author  JiapengLi
 * @brief   This file provides
a demostration on
 * how to train
VoiceRecognitionModule to
record your voice

**********************************
**********************************
******************
 * @note:
 * Use serial command to
control
VoiceRecognitionModule. '
 * All commands are case
insensitive. Default serial
baud rate 115200.
 *
 * COMMAND        FORMAT
EXAMPLE
Comment
 *
 * train          train (r0)
(r1)...           train 0 2
45               Train records
 * load           load (r0)
(r1) ...            load 0 51
2 3              Load records
 * clear          clear
clear
remove all records in
Recognizer
 * record         record /
record (r0) (r1)...  record /
record 0 79       Check record
train status
 * vr             vr
vr
Check recognizer status
 * getsig         getsig (r)
getsig 0                   Get
signature of record (r)
 * sigtrain       sigtrain (r)
(sig)            sigtrain 0
ZERO            Train one
record(r) with signature(sig)
 * settings       settings
settings
Check current system settings

**********************************
**********************************
******************
```

```
 * @section  HISTORY
 *
 * 2013/06/13    Initial
version.
 */
#include <SoftwareSerial.h>
#include
"VoiceRecognitionV3.h"

/**
 * Connection
 * Arduino
VoiceRecognitionModule
 * 2   ------->     TX
 * 3   ------->     RX
 */
VR myVR(2,3);    // 2:RX 3:TX,
you can choose your favourite
pins.

/**************************
**************************
***************/
/** declare print functions */
void printSeperator();
void printSignature(uint8_t
*buf, int len);
void printVR(uint8_t *buf);
void printLoad(uint8_t *buf,
uint8_t len);
void printTrain(uint8_t *buf,
uint8_t len);
void
printCheckRecognizer(uint8_t
*buf);
void printUserGroup(uint8_t
*buf, int len);
void printCheckRecord(uint8_t
*buf, int num);
void
printCheckRecordAll(uint8_t
*buf, int num);
void printSigTrain(uint8_t
*buf, uint8_t len);
void
printSystemSettings(uint8_t
*buf, int len);
void printHelp(void);

/**************************
**************************
***************/
// command analyze part
#define CMD_BUF_LEN      64+1
#define CMD_NUM     10
typedef int
(*cmd_function_t)(int, int);
uint8_t cmd[CMD_BUF_LEN];
uint8_t cmd_cnt;
uint8_t *paraAddr;
int receiveCMD();
int checkCMD(int len);
int checkParaNum(int len);
int findPara(int len, int
paraNum, uint8_t **addr);
int compareCMD(uint8_t *para1
, uint8_t *para2, int len);

int cmdTrain(int len, int
paraNum);
int cmdLoad(int len, int
paraNum);
int cmdTest(int len, int
paraNum);
int cmdVR(int len, int
paraNum);
int cmdClear(int len, int
paraNum);
int cmdRecord(int len, int
paraNum);
int cmdSigTrain(int len, int
paraNum);
int cmdGetSig(int len, int
paraNum);
int cmdSettings(int len, int
paraNum);
int cmdHelp(int len, int
paraNum);
/** cmdList, cmdLen,
cmdFunction has correspondence
*/
const char
cmdList[CMD_NUM][10] = {  //
command list table
  {
    "train"  }
  ,
  {
    "load"  }
  ,
  {
    "clear"  }
  ,
  {
    "vr"  }
  ,
  {
    "record"  }
  ,
  {
    "sigtrain"  }
  ,
  {
    "getsig"  }
  ,
  {
    "Settings"  }
  ,
```

```
  {
    "test"  }
  ,
  {
    "help"  }
  ,
};
const char cmdLen[CMD_NUM]= {
// command length
  5,  //  {"train"},
  4,  //  {"load"},
  5,  //  {"clear"},
  2,  //  {"vr"},
  6,  //  {"record"},
  8,  //  {"sigtrain"},
  6,  //  {"getsig"},
  8,  //  {"Settings"},
  4,  //  {"test"},
  4,  //  {"help"}
};
cmd_function_t
cmdFunction[CMD_NUM]={      //
command handle
fuction(function pointer
table)
  cmdTrain,
  cmdLoad,
  cmdClear,
  cmdVR,
  cmdRecord,
  cmdSigTrain,
  cmdGetSig,
  cmdSettings,
  cmdTest,
  cmdHelp,
};

/*****************************
*****************************
***************/
/** temprory data */
uint8_t buf[255];
uint8_t records[7]; // save
record

void setup(void)
{
  myVR.begin(9600);

  /** initialize */
  Serial.begin(115200);
  Serial.println(F("Elechouse
Voice Recognition V3 Module
\"train\" sample."));

  printSeperator();
  Serial.println(F("Usage:"));
  printSeperator();
  printHelp();
  printSeperator();
  cmd_cnt = 0;
}

void loop(void)
{
  int len, paraNum, paraLen,
i;

  /** receive Serial command
*/
  len = receiveCMD();
  if(len>0){
    /** check if the received
command is valid */
    if(!checkCMD(len)){

      /** check parameter
number of the received command
*/
      paraNum =
checkParaNum(len);

      /** display the receved
command back */
      Serial.write(cmd, len);

      /** find the first
parameter */
      paraLen = findPara(len,
1, &paraAddr);

      /** compare the received
command with command in the
list */
      for(i=0; i<CMD_NUM;
i++){
        /** compare command
length */
        if(paraLen ==
cmdLen[i]){
          /** compare command
content */
          if(
compareCMD(paraAddr, (uint8_t
*)cmdList[i], paraLen) == 0 ){
            /** call command
function */
            if(
cmdFunction[i](len, paraNum)
!= 0){

printSeperator();

Serial.println(F("Command
Format Error!"));

printSeperator();
            }
```

```
            break;
          }
        }
      }

      /** command is not
supported*/
      if(i == CMD_NUM){
        printSeperator();

Serial.println(F("Unkonwn
command"));
        printSeperator();
      }
    }
    else{
      /** received command is
invalid */
      printSeperator();

Serial.println(F("Command
format error"));
      printSeperator();
    }
  }

  /** try to receive recognize
result */
  int ret;
  ret = myVR.recognize(buf,
50);
  if(ret>0){
    /** voice recognized,
print result */
    printVR(buf);
  }
}

/**
 * @brief   receive command
from Serial.
 * @param   NONE.
 * @retval  command length, if
no command receive return -1.
 */
int receiveCMD()
{
  int ret;
  int len;
  unsigned long start_millis;
  start_millis = millis();
  while(1){
    ret = Serial.read();
    if(ret>0){
      start_millis = millis();
      cmd[cmd_cnt] = ret;
      if(cmd[cmd_cnt] ==
'\n'){
        len = cmd_cnt+1;
        cmd_cnt = 0;
        return len;
      }
      cmd_cnt++;
      if(cmd_cnt ==
CMD_BUF_LEN){
        cmd_cnt = 0;
        return -1;
      }
    }

    if(millis() - start_millis
> 100){
      cmd_cnt = 0;
      return -1;
    }
  }
}

/**
 * @brief   compare two
commands, case insensitive.
 * @param   para1  -->
command buffer 1
 * para2  -->  command buffer
2
 * len    -->  buffer length
 * @retval  0  --> equal
 * -1  --> unequal
 */
int compareCMD(uint8_t *para1
, uint8_t *para2, int len)
{
  int i;
  uint8_t res;
  for(i=0; i<len; i++){
    res = para2[i] - para1[i];
    if(res != 0 && res !=
0x20){
      res = para1[i] -
para2[i];
      if(res != 0 && res !=
0x20){
        return -1;
      }
    }
  }
  return 0;
}

/**
 * @brief   Check command
format.
 * @param   len  -->  command
length
 * @retval  0  -->  command is
valid
 * -1  -->  command is invalid
 */
```

```
int checkCMD(int len)
{
  int i;
  for(i=0; i<len; i++){
    if(cmd[i] > 0x1F && cmd[i]
< 0x7F){

    }
    else if(cmd[i] == '\t' ||
cmd[i] == ' ' || cmd[i] ==
'\r' || cmd[i] == '\n'){

    }
    else{
      return -1;
    }
  }
  return 0;
}

/**
 * @brief   Check the number
of parameters in the command
 * @param   len  -->  command
length
 * @retval  number of
parameters
 */
int checkParaNum(int len)
{
  int cnt=0, i;
  for(i=0; i<len; ){
    if(cmd[i]!='\t' &&
cmd[i]!=' ' && cmd[i] != '\r'
&& cmd[i] != '\n'){
      cnt++;
      while(cmd[i] != '\t' &&
cmd[i] != ' ' && cmd[i] !=
'\r' && cmd[i] != '\n'){
        i++;
      }
    }
    i++;
  }
  return cnt;
}

/**
 * @brief   Find the specified
parameter.
 * @param   len       -->
command length
 * paraIndex -->  parameter
index
 * addr      -->  return
value. position of the
parameter
 * @retval  length of
specified parameter
 */
int findPara(int len, int
paraIndex, uint8_t **addr)
{
  int cnt=0, i, paraLen;
  uint8_t dt;
  for(i=0; i<len; ){
    dt = cmd[i];
    if(dt!='\t' && dt!=' '){
      cnt++;
      if(paraIndex == cnt){
        *addr = cmd+i;
        paraLen = 0;
        while(cmd[i] != '\t'
&& cmd[i] != ' ' && cmd[i] !=
'\r' && cmd[i] != '\n'){
          i++;
          paraLen++;
        }
        return paraLen;
      }
      else{
        while(cmd[i] != '\t'
&& cmd[i] != ' ' && cmd[i] !=
'\r' && cmd[i] != '\n'){
          i++;
        }
      }
    }
    else{
      i++;
    }
  }
  return -1;
}

int cmdHelp(int len, int
paraNum)
{
  if(paraNum != 1){
    return -1;
  }
  printSeperator();
  printHelp();
  printSeperator();
  return 0;
}

/**
 * @brief   Handle "train"
command
 * @param   len     -->
command length
 * paraNum --> number of
parameters
 * @retval  0 --> success
 * -1 --> Command format error
 */
```

```
int cmdTrain(int len, int
paraNum)
{
  int i, ret;
  if(paraNum < 2 || paraNum >
8 ){
    return -1;
  }

  for(i=2; i<=paraNum; i++){
    findPara(len, i,
&paraAddr);
    records[i-2] = atoi((char
*)paraAddr);
    if(records[i-2] == 0 &&
*paraAddr != '0'){
      return -1;
    }
  }
  printSeperator();
  ret = myVR.train(records,
paraNum-1, buf);
  //  ret =
myVR.train(records, paraNum-
1);
  if(ret >= 0){
    printTrain(buf, ret);
  }
  else if(ret == -1){
    Serial.println(F("Train
failed."));
  }
  else if(ret == -2){
    Serial.println(F("Train
Timeout."));
  }
  printSeperator();
  return 0;
}

/**
 * @brief   Handle "load"
command
 * @param   len     -->
command length
 * paraNum --> number of
parameters
 * @retval  0 --> success
 * -1 --> Command format error
 */
int cmdLoad(int len, int
paraNum)
{
  int i, ret;
  if(paraNum < 2 || paraNum >
8 ){
    return -1;
  }
  for(i=2; i<=paraNum; i++){
    findPara(len, i,
&paraAddr);
    records[i-2] = atoi((char
*)paraAddr);
    if(records[i-2] == 0 &&
*paraAddr != '0'){
      return -1;
    }
  }
  //  myVR.writehex(records,
paraNum-1);
  ret = myVR.load(records,
paraNum-1, buf);
  printSeperator();
  if(ret >= 0){
    printLoad(buf, ret);
  }
  else{
    Serial.println(F("Load
failed or timeout."));
  }
  printSeperator();
  return 0;
}

/**
 * @brief   Handle "clear"
command
 * @param   len     -->
command length
 * paraNum --> number of
parameters
 * @retval  0 --> success
 * -1 --> Command format error
 */
int cmdClear(int len, int
paraNum)
{
  if(paraNum != 1){
    return -1;
  }
  if(myVR.clear() == 0){
    printSeperator();

Serial.println(F("Recognizer
cleared."));
    printSeperator();
  }
  else{
    printSeperator();
    Serial.println(F("Clear
recognizer failed or
timeout."));
    printSeperator();
  }
  return 0;
}
```

```
/**
 * @brief   Handle "vr"
command
 * @param   len     -->
command length
 * paraNum --> number of
parameters
 * @retval  0 --> success
 * -1 --> Command format error
 */
int cmdVR(int len, int
paraNum)
{
  int ret;
  if(paraNum != 1){
    return -1;
  }
  ret =
myVR.checkRecognizer(buf);
  if(ret<=0){
    printSeperator();
    Serial.println(F("Check
recognizer failed or
timeout."));
    printSeperator();
    return 0;
  }
  printSeperator();
  printCheckRecognizer(buf);
  printSeperator();
  return 0;
}

/**
 * @brief   Handle "record"
command
 * @param   len     -->
command length
 * paraNum --> number of
parameters
 * @retval  0 --> success
 * -1 --> Command format error
 */
int cmdRecord(int len, int
paraNum)
{
  int ret;
  if(paraNum == 1){
    ret =
myVR.checkRecord(buf);
    printSeperator();
    if(ret>=0){
      printCheckRecordAll(buf,
ret);
    }
    else{
      Serial.println(F("Check
record failed or timeout."));
    }
    printSeperator();
  }
  else if(paraNum < 9){
    for(int i=2; i<=paraNum;
i++){
      findPara(len, i,
&paraAddr);
      records[i-2] =
atoi((char *)paraAddr);
      if(records[i-2] == 0 &&
*paraAddr != '0'){
        return -1;
      }
    }

    ret =
myVR.checkRecord(buf, records,
paraNum-1);    // auto clean
duplicate records
    printSeperator();
    if(ret>=0){
      printCheckRecord(buf,
ret);
    }
    else{
      Serial.println(F("Check
record failed or timeout."));
    }
    printSeperator();
  }
  else{
    return -1;
  }
  return 0;
}

/**
 * @brief   Handle "sigtrain"
command
 * @param   len     -->
command length
 * paraNum --> number of
parameters
 * @retval  0 --> success
 * -1 --> Command format error
 */
int cmdSigTrain(int len, int
paraNum)
{
  int ret, sig_len;
  uint8_t *lastAddr;
  if(paraNum < 2){
    return -1;
  }

  findPara(len, 2, &paraAddr);
  records[0] = atoi((char
*)paraAddr);
```

```
  if(records[0] == 0 &&
*paraAddr != '0'){
    return -1;
  }

  findPara(len, 3, &paraAddr);
  sig_len = findPara(len,
paraNum, &lastAddr);
  sig_len +=( (unsigned
int)lastAddr - (unsigned
int)paraAddr );

  printSeperator();
  ret =
myVR.trainWithSignature(record
s[0], paraAddr, sig_len, buf);
  //  ret =
myVR.trainWithSignature(record
s, paraNum-1);
  if(ret >= 0){
    printSigTrain(buf, ret);
  }
  else{
    Serial.println(F("Train
with signature failed or
timeout."));
  }
  printSeperator();

  return 0;
}

/**
 * @brief   Handle "getsig"
command
 * @param   len     -->
command length
 * paraNum --> number of
parameters
 * @retval  0 --> success
 * -1 --> Command format error
 */
int cmdGetSig(int len, int
paraNum)
{
  int ret;
  if(paraNum != 2){
    return -1;
  }

  findPara(len, 2, &paraAddr);
  records[0] = atoi((char
*)paraAddr);
  if(records[0] == 0 &&
*paraAddr != '0'){
    return -1;
  }

  ret =
myVR.checkSignature(records[0]
, buf);

  printSeperator();
  if(ret == 0){

Serial.println(F("Signature
isn't set."));
  }
  else if(ret > 0){

Serial.print(F("Signature:"));
    printSignature(buf, ret);
    Serial.println();
  }
  else{
    Serial.println(F("Get sig
error or timeout."));
  }
  printSeperator();

  return 0;
}

/**
 * @brief   Handle "test"
command
 * @param   len     -->
command length
 * paraNum --> number of
parameters
 * @retval  0 --> success
 * -1 --> Command format error
 */
int cmdTest(int len, int
paraNum)
{
  printSeperator();
  Serial.println(F("TEST is
not supported."));
  printSeperator();
  return 0;
}

int cmdSettings(int len, int
paraNum)
{
  int ret;
  if(paraNum != 1){
    return -1;
  }
  ret =
myVR.checkSystemSettings(buf);
  if( ret > 0){
    printSeperator();
    printSystemSettings(buf,
ret);
    printSeperator();
```

```
  }
  else{
    printSeperator();
    Serial.println(F("Check
system settings error or
timeout"));
    printSeperator();
  }
  return 0;
}

/*****************************
*****************************
*****************/
/**
 * @brief   Print signature,
if the character is invisible,
 * print hexible value
instead.
 * @param   buf     -->
command length
 * len     --> number of
parameters
 */
void printSignature(uint8_t
*buf, int len)
{
  int i;
  for(i=0; i<len; i++){
    if(buf[i]>0x19 &&
buf[i]<0x7F){
      Serial.write(buf[i]);
    }
    else{
      Serial.print(F("["));
      Serial.print(buf[i],
HEX);
      Serial.print(F("]"));
    }
  }
}

/**
 * @brief   Print signature,
if the character is invisible,
 * print hexible value
instead.
 * @param   buf  -->  VR
module return value when voice
is recognized.
 * buf[0]  -->  Group mode(FF:
None Group, 0x8n: User,
0x0n:System
 * buf[1]  -->  number of
record which is recognized.
 * buf[2]  -->  Recognizer
index(position) value of the
recognized record.
 * buf[3]  -->  Signature
length
 * buf[4]~buf[n] --> Signature
 */
void printVR(uint8_t *buf)
{
  Serial.println(F("VR
Index\tGroup\tRecordNum\tSigna
ture"));

  Serial.print(buf[2], DEC);
  Serial.print(F("\t\t"));

  if(buf[0] == 0xFF){
    Serial.print(F("NONE"));
  }
  else if(buf[0]&0x80){
    Serial.print(F("UG "));

Serial.print(buf[0]&(~0x80),
DEC);
  }
  else{
    Serial.print(F("SG "));
    Serial.print(buf[0], DEC);
  }
  Serial.print(F("\t"));

  Serial.print(buf[1], DEC);
  Serial.print(F("\t\t"));
  if(buf[3]>0){
    printSignature(buf+4,
buf[3]);
  }
  else{
    Serial.print(F("NONE"));
  }
  Serial.println(F("\r\n"));
}

/**
 * @brief   Print seperator.
Print 80 '-'.
 */
void printSeperator()
{
  for(int i=0; i<80; i++){
    Serial.write('-');
  }
  Serial.println();
}

/**
 * @brief   Print recoginizer
status.
 * @param   buf  -->  VR
module return value when voice
is recognized.
```

```
 * buf[0]      -->  Number of
valid voice records in
recognizer
 * buf[i+1]    -->  Record
number.(0xFF: Not
loaded(Nongroup mode), or not
set (Group mode)) (i= 0, 1,
... 6)
 * buf[8]      -->  Number of
all voice records in
recognizer
 * buf[9]      -->  Valid
records position indicate.
 * buf[10]     -->  Group mode
indicate(FF: None Group, 0x8n:
User, 0x0n:System)
 */
void
printCheckRecognizer(uint8_t
*buf)
{
  Serial.print(F("All voice
records in recognizer: "));
  Serial.println(buf[8], DEC);
  Serial.print(F("Valid voice
records in recognizer: "));
  Serial.println(buf[0], DEC);
  if(buf[10] == 0xFF){
    Serial.println(F("VR is
not in group mode."));
  }
  else if(buf[10]&0x80){
    Serial.print(F("VR is in
user group mode:"));

Serial.println(buf[10]&0x7F,
DEC);
  }
  else{
    Serial.print(F("VR is in
system group mode:"));
    Serial.println(buf[10],
DEC);
  }
  Serial.println(F("VR
Index\tRecord\t\tComment"));
  for(int i=0; i<7; i++){
    Serial.print(i, DEC);
    Serial.print(F("\t\t"));
    if(buf[i+1] == 0xFF){
      if(buf[10] == 0xFF){

Serial.print(F("Unloaded\tNONE
"));
      }
      else{
        Serial.print(F("Not
Set\t\tNONE"));
      }
    }
    else{
      Serial.print(buf[i+1],
DEC);
      Serial.print(F("\t\t"));
      if(buf[9]&(1<<i)){

Serial.print(F("Valid"));
      }
      else{

Serial.print(F("Untrained"));
      }
    }

    Serial.println();
  }
}

/**
 * @brief   Print record train
status.
 * @param   buf  -->  Check
record command return value
 * buf[0]     -->  Number of
checked records
 * buf[2i+1]  -->  Record
number.
 * buf[2i+2]  -->  Record
train status. (00: untrained,
01: trained, FF: record value
out of range)
 * (i = 0 ~ buf[0]-1 )
 * num  -->  Number of trained
records
 */
void printCheckRecord(uint8_t
*buf, int num)
{
  Serial.print(F("Check "));
  Serial.print(buf[0], DEC);
  Serial.println(F("
records."));

  Serial.print(num, DEC);
  if(num>1){
    Serial.println(F(" records
trained."));
  }
  else{
    Serial.println(F(" record
trained."));
  }

  for(int i=0; i<buf[0]*2; i
+= 2){
    Serial.print(buf[i+1],
DEC);
```

```
    Serial.print(F("\t--
>\t"));
    switch(buf[i+2]){
    case 0x01:

Serial.print(F("Trained"));
      break;
    case 0x00:

Serial.print(F("Untrained"));
      break;
    case 0xFF:
      Serial.print(F("Record
value out of range"));
      break;
    default:
      Serial.print(F("Unknown
Stauts"));
      break;
    }
    Serial.println();
  }
}

/**
 * @brief   Print record train
status.
 * @param   buf  -->  Check
record command return value
 * buf[0]     -->  Number of
checked records
 * buf[2i+1]  -->  Record
number.
 * buf[2i+2]  -->  Record
train status. (00: untrained,
01: trained, FF: record value
out of range)
 * (i = 0 ~ buf[0]-1 )
 * num  -->  Number of trained
records
 */
void
printCheckRecordAll(uint8_t
*buf, int num)
{
  Serial.print(F("Check
255"));
  Serial.println(F("
records."));

  Serial.print(num, DEC);
  if(num>1){
    Serial.println(F(" records
trained."));
  }
  else{
    Serial.println(F(" record
trained."));
  }
  myVR.writehex(buf, 255);
  for(int i=0; i<255; i++){
    if(buf[i] == 0xF0){
      continue;
    }
    Serial.print(i, DEC);
    Serial.print(F("\t--
>\t"));
    switch(buf[i]){
    case 0x01:

Serial.print(F("Trained"));
      break;
    case 0x00:

Serial.print(F("Untrained"));
      break;
    case 0xFF:
      Serial.print(F("Record
value out of range"));
      break;
    default:
      Serial.print(F("Unknown
Stauts"));
      break;
    }
    Serial.println();
  }
}

/**
 * @brief   Print check user
group result.
 * @param   buf  -->  Check
record command return value
 * buf[8i]    -->  group
number.
 * buf[8i+1]  -->  group
position 0 status.
 * buf[8i+2]  -->  group
position 1 status.
 * ...                ...
 * buf[8i+6]  -->  group
position 5 status.
 * buf[8i+7]  -->  group
position 6 status.
 * (i = 0 ~ len)
 * len  -->  number of checked
groups
 */
void printUserGroup(uint8_t
*buf, int len)
{
  int i, j;
  Serial.println(F("Check User
Group:"));
  for(i=0; i<len; i++){
    Serial.print(F("Group:"));
    Serial.println(buf[8*i]);
```

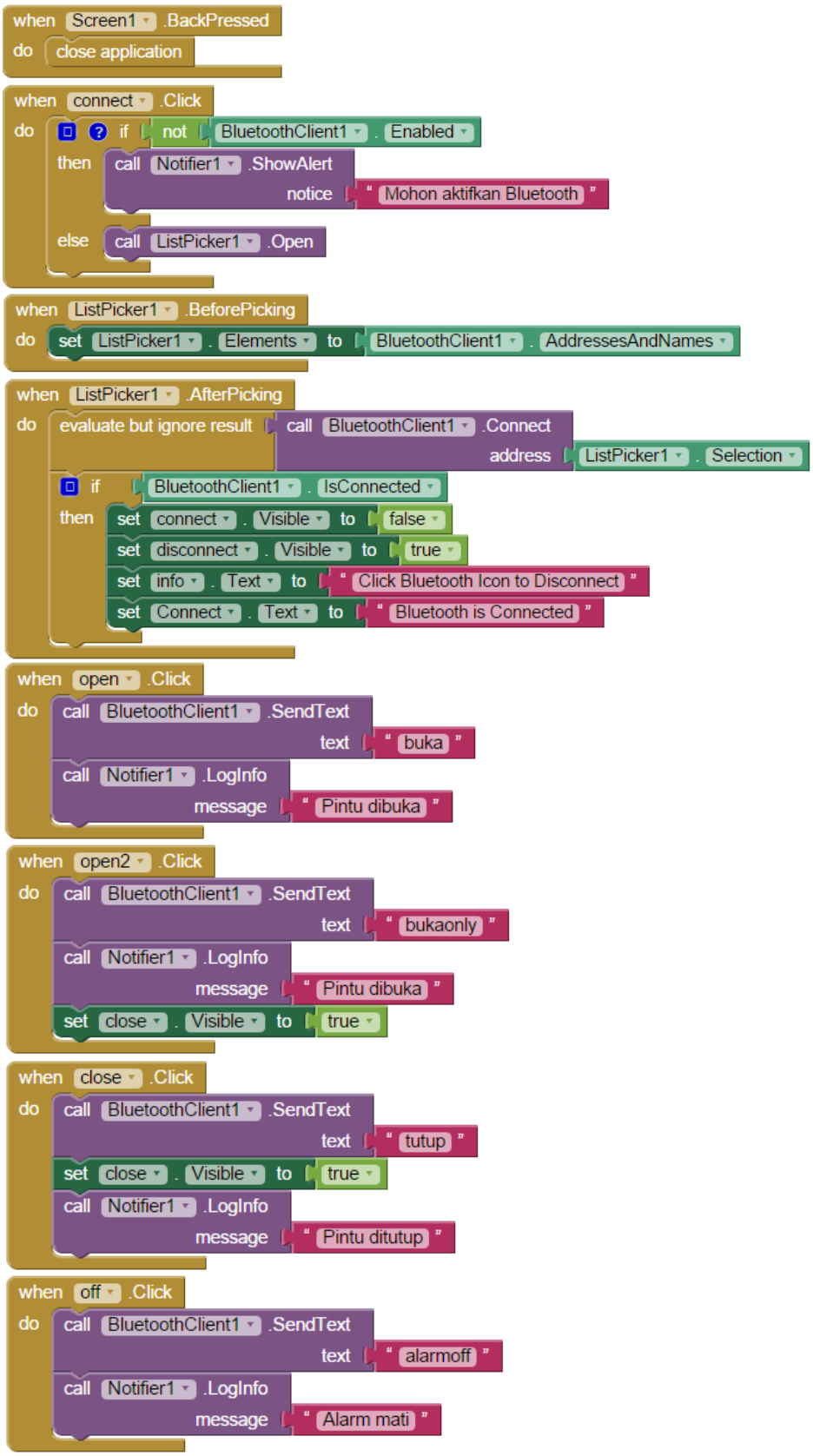
```
    for(j=0; j<7; j++){
      if(buf[8*i+1+j] ==
0xFF){

Serial.print(F("NONE\t"));
      }
      else{

Serial.print(buf[8*i+1+j],
DEC);
        Serial.print(F("\t"));
      }
    }
    Serial.println();
  }
}

/**
 * @brief   Print "load"
command return value.
 * @param   buf  -->  "load"
command return value
 * buf[0]    -->  number of
records which are load
successfully.
 * buf[2i+1]  -->  record
number
 * buf[2i+2]  -->  record load
status.
 * 00 --> Loaded
 * FC --> Record already in
recognizer
 * FD --> Recognizer full
 * FE --> Record untrained
 * FF --> Value out of range"
 * (i = 0 ~ (len-1)/2 )
 * len  -->  length of buf
 */
void printLoad(uint8_t *buf,
uint8_t len)
{
  if(len == 0){
    Serial.println(F("Load
Successfully."));
    return;
  }
  else{
    Serial.print(F("Load
success: "));
    Serial.println(buf[0],
DEC);
  }
  for(int i=0; i<len-1; i +=
2){
    Serial.print(F("Record
"));
    Serial.print(buf[i+1],
DEC);
    Serial.print(F("\t"));
    switch(buf[i+2]){
    case 0:

Serial.println(F("Loaded"));
      break;
    case 0xFC:
      Serial.println(F("Record
already in recognizer"));
      break;
    case 0xFD:

Serial.println(F("Recognizer
full"));
      break;
    case 0xFE:
      Serial.println(F("Record
untrained"));
      break;
    case 0xFF:
      Serial.println(F("Value
out of range"));
      break;
    default:

Serial.println(F("Unknown
status"));
      break;
    }
  }
}

/**
 * @brief   Print "train"
command return value.
 * @param   buf  -->  "train"
command return value
 * buf[0]    -->  number of
records which are trained
successfully.
 * buf[2i+1]  -->  record
number
 * buf[2i+2]  -->  record
train status.
 * 00 --> Trained
 * FE --> Train Time Out
 * FF --> Value out of range"
 * (i = 0 ~ len-1 )
 * len  -->  length of buf
 */
void printTrain(uint8_t *buf,
uint8_t len)
{
  if(len == 0){
    Serial.println(F("Train
Finish."));
    return;
  }
  else{
```

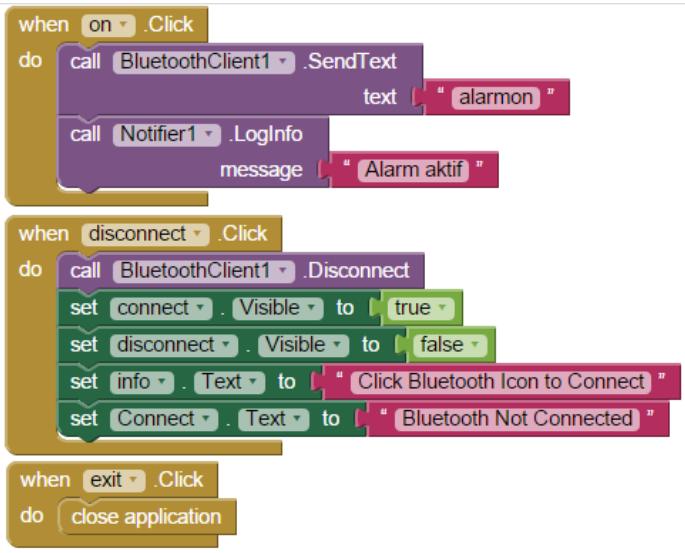
```
    Serial.print(F("Train
success: "));
    Serial.println(buf[0],
DEC);
  }
  for(int i=0; i<len-1; i +=
2){
    Serial.print(F("Record
"));
    Serial.print(buf[i+1],
DEC);
    Serial.print(F("\t"));
    switch(buf[i+2]){
    case 0:

Serial.println(F("Trained"));
      break;
    case 0xFE:
      Serial.println(F("Train
Time Out"));
      break;
    case 0xFF:
      Serial.println(F("Value
out of range"));
      break;
    default:
      Serial.print(F("Unknown
status "));
      Serial.println(buf[i+2],
HEX);
      break;
    }
  }
}

/**
 * @brief   Print "sigtrain"
command return value.
 * @param   buf  -->
"sigtrain" command return
value
 * buf[0]  -->  number of
records which are trained
successfully.
 * buf[1]  -->  record number
 * buf[2]  -->  record train
status.
 * 00 --> Trained
 * F0 --> Trained, signature
truncate
 * FE --> Train Time Out
 * FF --> Value out of range"
 * buf[3] ~ buf[len-1] -->
Signature.
 * len  -->  length of buf
 */
void printSigTrain(uint8_t
*buf, uint8_t len)
{
  if(len == 0){
    Serial.println(F("Train
With Signature Finish."));
    return;
  }
  else{
    Serial.print(F("Success:
"));
    Serial.println(buf[0],
DEC);
  }
  Serial.print(F("Record "));
  Serial.print(buf[1], DEC);
  Serial.print(F("\t"));
  switch(buf[2]){
  case 0:

Serial.println(F("Trained"));
    break;
  case 0xF0:
    Serial.println(F("Trained,
signature truncate"));
    break;
  case 0xFE:
    Serial.println(F("Train
Time Out"));
    break;
  case 0xFF:
    Serial.println(F("Value
out of range"));
    break;
  default:
    Serial.print(F("Unknown
status "));
    Serial.println(buf[2],
HEX);
    break;
  }
  Serial.print(F("SIG: "));
  Serial.write(buf+3, len-3);
  Serial.println();
}

/**
 * @brief   Print "settings"
command return value.
 * @param   buf  -->
"settings" command return
value
 * buf[0]  -->  number of
records which are trained
successfully.
 * buf[1]  -->  record number
 * buf[2]  -->  record train
status.
 * 00 --> Trained
 * F0 --> Trained, signature
truncate
 * FE --> Train Time Out
```

```
 * FF --> Value out of range"
 * buf[3] ~ buf[len-1] -->
Signature.
 * len  -->  length of buf
 */

const unsigned int
io_pw_tab[16]={
  10,  15,  20,  25,  30,  35,
40,  45,
  50,  75,  100, 200, 300,
400, 500, 1000
};

void
printSystemSettings(uint8_t
*buf, int len)
{

  switch(buf[0]){
  case 0:
  case 3:
    Serial.println(F("Baud
rate: 9600"));
    break;
  case 1:
    Serial.println(F("Baud
rate: 2400"));
    break;
  case 2:
    Serial.println(F("Baud
rate: 4800"));
    break;
  case 4:
    Serial.println(F("Baud
rate: 19200"));
    break;
  case 5:
    Serial.println(F("Baud
rate: 38400"));
    break;
  default:
    Serial.println(F("Baud
rate: UNKONOWN"));
    break;
  }

  switch(buf[1]){
  case 0:
  case 0xFF:
    Serial.println(F("Outpu IO
Mode: Pulse"));
    break;
  case 1:
    Serial.println(F("Outpu IO
Mode: Toggle"));
    break;
  case 2:
    Serial.println(F("Outpu IO
Mode: Clear(When recognized)
"));
    break;
  case 3:
    Serial.println(F("Outpu IO
Mode: Set(When recognized)"));
    break;
  default:
    Serial.println(F("Output
IO Mode: UNKONOWN"));
    break;
  }

  if(buf[2] > 15){
    Serial.println(F("Pulse
width: UNKONOWN"));
  }
  else{
    Serial.print(F("Pulse
Width: "));

Serial.print(io_pw_tab[buf[2]]
, DEC);
    Serial.println(F("ms"));
  }

  if(buf[3] == 0 || buf[3] ==
0xFF){
    Serial.println(F("Auto
Load: disable"));
  }
  else{
    Serial.println(F("Auto
Load: enable"));
  }

  switch(buf[4]){
  case 0:
  case 0xFF:
    Serial.println(F("Group
control by external IO:
disabled"));
    break;
  case 1:
    Serial.println(F("Group
control by external IO: system
group selected"));
    break;
  case 2:
    Serial.println(F("Group
control by external IO: user
group selected"));
    break;
  default:
    Serial.println(F("Group
control by external IO:
UNKNOWN STATUS"));
    break;
```

```
  }
}

void printHelp(void)
{
  Serial.println(F("COMMAND
FORMAT
EXAMPLE
Comment"));
  printSeperator();
  //  Serial.println(F("------
------------------------------
------------------------------
------------------------------
--------------"));
  Serial.println(F("train
train (r0) (r1)...
train 0 2 45
Train records"));
  Serial.println(F("load
load (r0) (r1) ...
load 0 51 2 3
Load records"));
  Serial.println(F("clear
clear
clear
remove all records in
Recognizer"));
  Serial.println(F("record
record / record (r0) (r1)...
record / record 0 79
Check record train status"));
  Serial.println(F("vr
vr
vr
Check recognizer status"));
  Serial.println(F("getsig
getsig (r)
getsig 0                    Get
signature of record (r)"));
  Serial.println(F("sigtrain
sigtrain (r) (sig)
sigtrain 0 ZERO
Train one record(r) with
signature(sig)"));
  Serial.println(F("settings
settings
settings
Check current system
settings"));
  Serial.println(F("help
help
help
print this message"));}
```

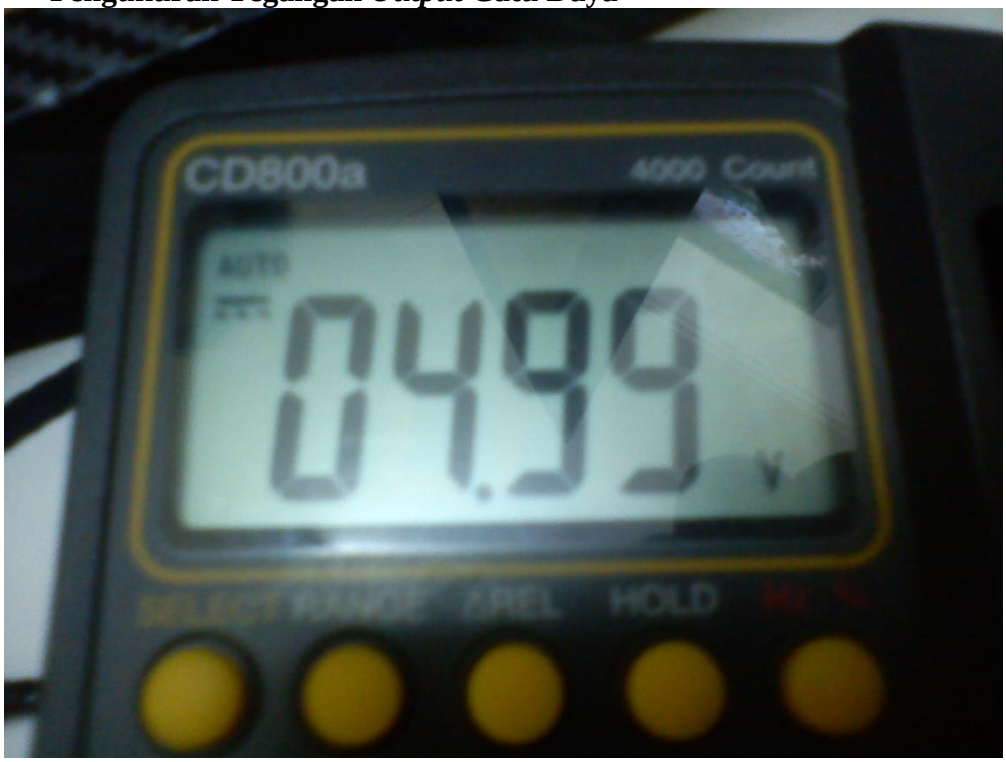
Lampiran 2 Blok Program *App Inventor*



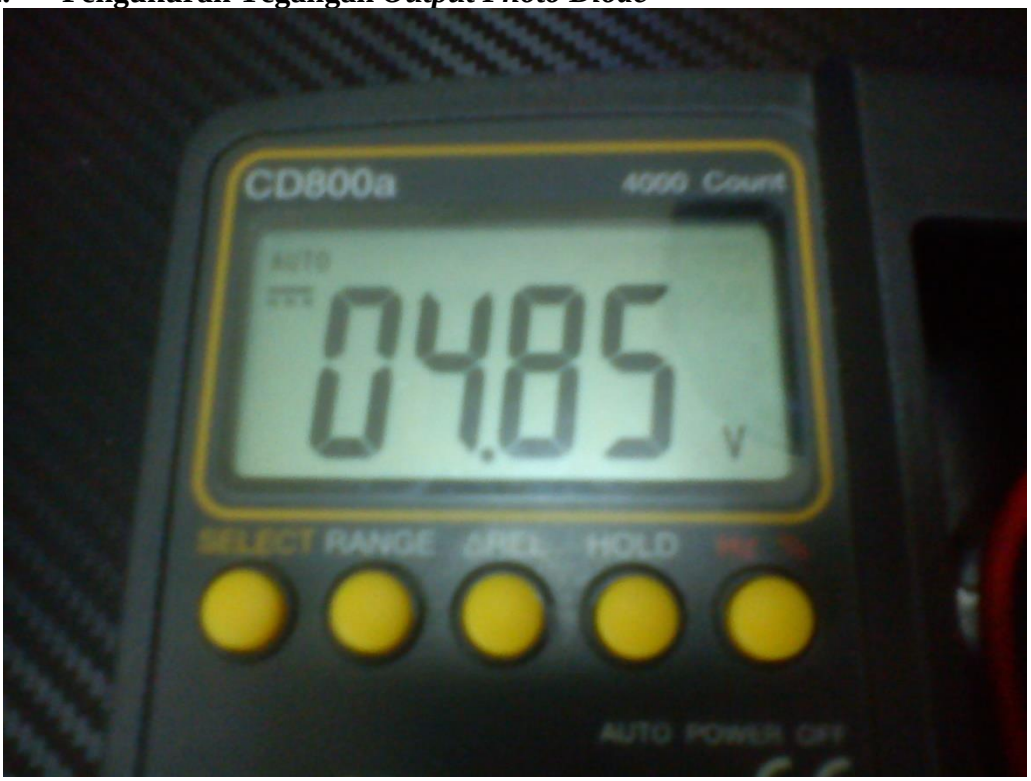


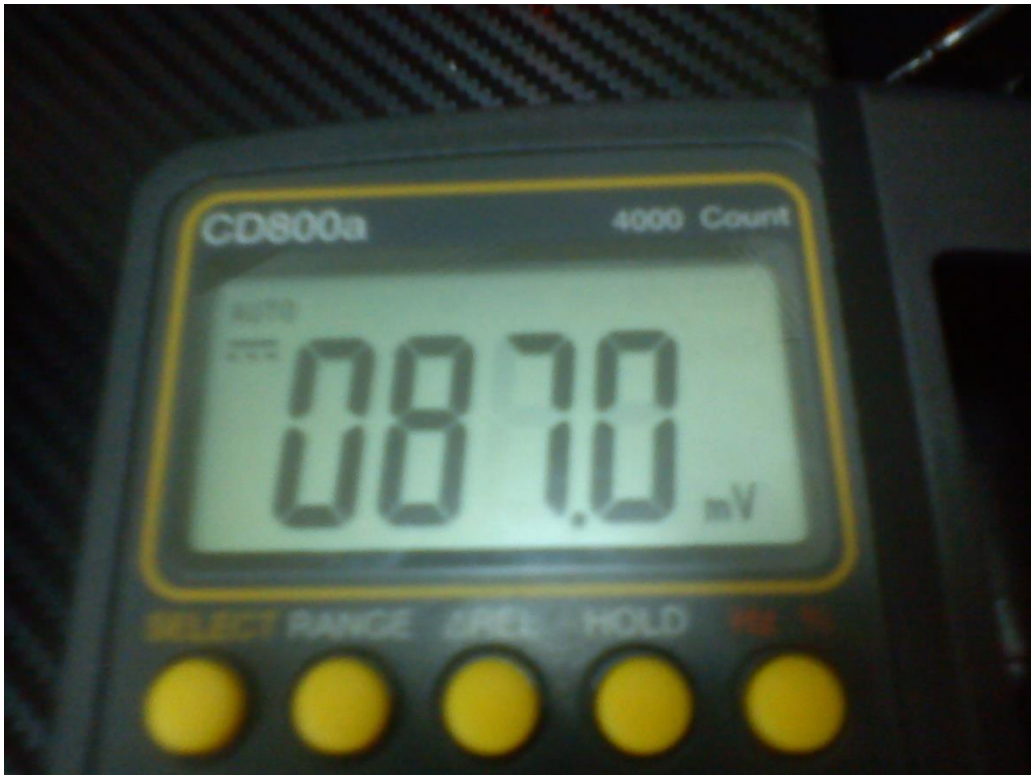
Lampiran 3 Foto Pengukuran

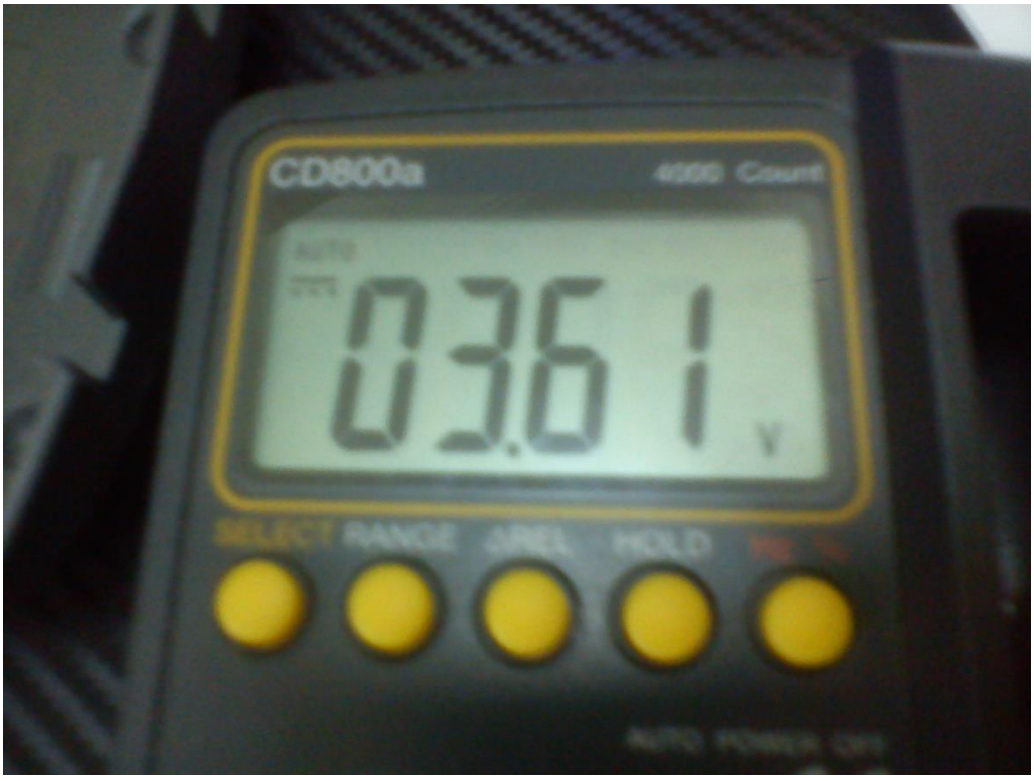
1. Pengukuran Tegangan *Output* Catu Daya



2. Pengukuran Tegangan *Output Photo Diode*




CD800a
4000 Count
087.0
mV
SELECT
RANGE
REL
HOLD

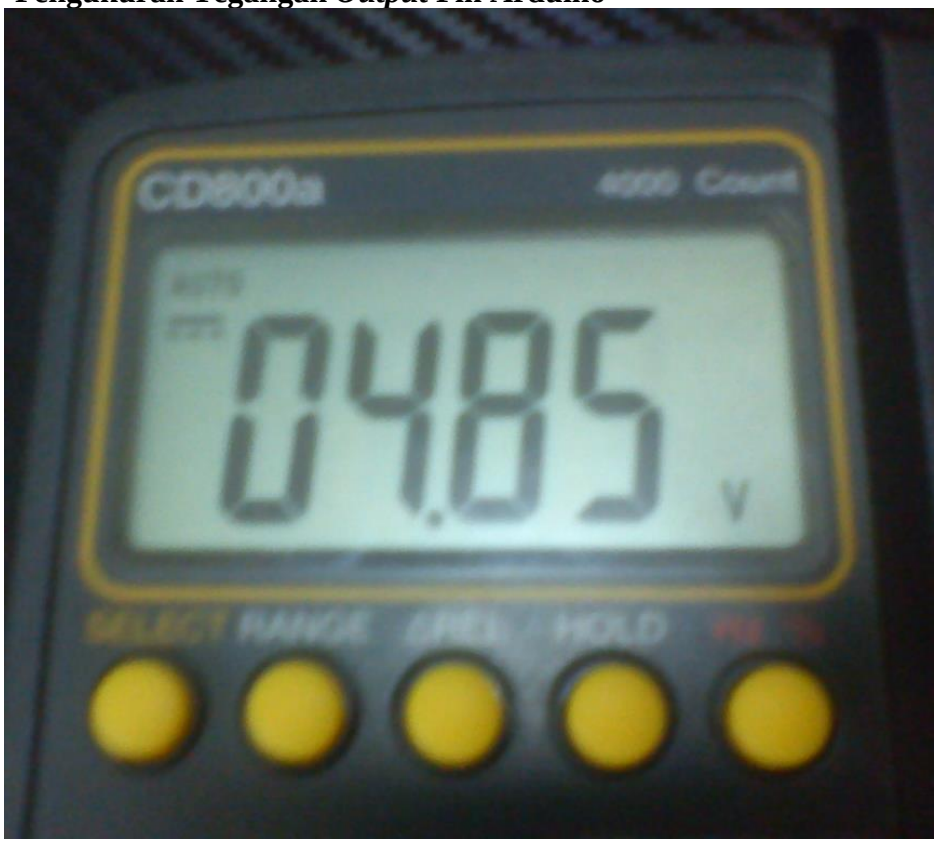

CD800a
4000 Count
03.61
V
SELECT
RANGE
REL
HOLD
AUTO POWER OFF


CD800a
4000 Count
AUTO
3.623
V
SELECT RANGE ΔREL HOLD Hz %


CD800a
4000 Count
AUTO
04.66
V
SELECT RANGE ΔREL HOLD Hz %

3. Pengukuran Tegangan *Output* Motor DC

4. Pengukuran Tegangan *Output* Pin Arduino



Riwayat Hidup

Deden Luthfi Fermana, lahir di Bogor 06 Oktober 1993. Penulis bertempat tinggal di Kp. Setu Tonggoh RT 01/05 Desa Leuwimekar Kecamatan Leuwiliang Kabupaten Bogor, Jawa Barat. Mahasiswa yang hobi bermain musik, browsing dan futsal ini, pernah menempuh pendidikan dasar di SDN Leuwimekar. Setelah itu, penulis melanjutkan pendidikan menengah di MTsN Model Babakansirna dan SMAN 1 Leuwiliang. Setelah menyelesaikan pendidikan SMA, penulis melanjutkan pendidikan ke jenjang lebih tinggi di Universitas Negeri Jakarta Jurusan Teknik Elektro Program Studi Pendidikan Teknik Elektronika. Penulis dapat dihubungi melalui alamat e-mail di dhenlfmn@gmail.com.