TUGAS AKHIR – TE 145561

PENGAMAN KENDARAAN BERMOTOR DENGAN REMOTE ANDROID BERBASIS GSM, GPS, DAN *BLUETOOTH*

Nasrul Arifianto
NRP 2214030070
Moch Aan Fahrizal
NRP 2214030095

Dosen Pembimbing
Eko Pramunanto, ST., MT.

PROGRAM STUDI KOMPUTER KONTROL
Departemen Teknik Elektro Otomasi
Fakultas Vokasi
Institut Teknologi Sepuluh Nopember
Surabaya 2017

FINAL PROJECT – **TE 145561**

MOTOR VEHICLE SAFETY WITH REMOTE ANDROID-BASED GSM, GPS, AND BLUETOOTH

Nasrul Arifianto
NRP 2214030070
Moch Aan Fahrizal
NRP 2214030095

Advisor
Eko Pramunanto, ST., MT.

COMPUTER CONTROL STUDY PROGRAM
Electrical and Automation Engineering Department
Faculty of Vocational
Sepuluh Nopember Institute of Technology
Surabaya 2017

PERNYATAAN KEASLIAN
TUGAS AKHIR

Dengan ini saya menyatakan bahwa isi sebagian maupun keseluruhan Tugas Akhir saya dengan judul "**PENGAMAN KENDARAAN BERMOTOR DENGAN REMOTE ANDROID BERBASIS GSM, GPS, DAN *BLUETOOTH***" adalah benar-benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diijinkan dan bukan merupakan karya pihak lain yang saya akui sebagai karya sendiri.

Semua referensi yang dikutip maupun dirujuk telah ditulis secara lengkap pada daftar pustaka.

Apabila ternyata pernyataan ini tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Surabaya, 18 Juli 2017

Moch Aan Fahrizal
NRP 2214030095

Nasrul Arifianto
2214030070

-----Halaman ini sengaja dikosongkan-----

PENGAMAN KENDARAAN BERMOTOR DENGAN REMOTE ANDROID BERBASIS GSM, GPS, DAN *BLUETOOTH*

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Ahli Madya
Pada
Program Studi Komputer Kontrol
Departemen Teknik Elektro Otomasi
Fakultas Vokasi
Institut Teknologi Sepuluh Nopember

Menyetujui :

Dosen Pembimbing

Eko Pramunanto, ST., MT.
NIP. 19661203 199412 1 001

SURABAYA
JULI, 2017

-----Halaman ini sengaja dikosongkan-----

PENGAMAN KENDARAAN BERMOTOR DENGAN REMOTE ANDROID BERBASIS GSM, GPS, DAN *BLUETOOTH*

Nama : **Moch Aan Fahrizal**
Nasrul Arifianto
Pembimbing : **Eko Pramunanto, ST.,MT.**

ABSTRAK

Pada era modern ini teknologi berkembang dengan pesat, tapi berbeda dengan teknologi pengaman sepeda motor saat ini. Dikarenakan pengaman sepeda motor saat ini belum dapat mengamankan dengan baik, contohnya sepeda motor yang diparkir kemudian dicuri dengan cara didorong dan ketika terjadi perampasan di jalan secara paksa dengan senjata tajam. Untuk itu perlu adanya sistem pengaman kendaraan yang dapat mencegah kasus kriminalitas seperti itu. Pengaman sepeda motor dengan memanfaatkan GSM, *bluetooth* dan GPS menjadi solusi yang tepat. Untuk memudahkan mengakses sistem pengaman tersebut dibutuhkan suatu aplikasi yang berfungsi sebagai remot kendali untuk mengaktifkan sistem keamanan dengan memanfaatkan komunikasi *bluetooth*, maka dibuatlah aplikasi Android sebagai remot kendali pengaman sepeda motor. Dari aplikasi tersebut terdapat tombol untuk memilih mode keamanan yang ingin digunakan, terdapat mode parkir saat kendaraan diparkir dan mode *driver* saat berkendara. Terdapat juga fitur untuk menemukan lokasi motor dengan bantuan GPS, yang dapat kita akses dengan cara mengirim pesan pada sistem pengaman. Dari hasil pengujian, pengiriman data dari *handphone* ke sistem pengaman dapat terkirim dengan jarak maksimal 14,6 m dan dari jarak tersebut, sistem pengaman akan aktif ketika terjadi perampasan di jalan secara paksa dengan memutus aliran listrik pada CDI, memblokir aliran bensin dan sirine akan berbunyi, GPS tidak berfungsi dengan baik ketika kendaraan berada di dalam gedung dikarenakan satelit tidak dapat mengirim data koordinat kendaraan. Sehingga keberhasilan dari keseluruhan sistem mencapai 80%.

Kata Kunci : Sistem Pengaman, *Bluetooth*, GPS, Aplikasi Android

-----Halaman ini sengaja dikosongkan-----

MOTOR VEHICLE SAFETY WITH REMOTE ANDROID-BASED GSM, GPS, AND BLUETOOTH

Name **: Moch Aan Fahrizal**
Nasrul Arifianto
Advisor **: Eko Pramunanto, S.T.,M.T.**

ABSTRACT

In this modern era, technology develops very fast. But the current motorcycle safety is technology different, because the current motorcycle safety is not safety enough for motorcycle, for example when a parked motorcycle was stolen in a way encouraged without owner knowing and when that happens the robbery on the road by force with a sharp weapons. For it takes a motorcycle safety system which can prevent such crime cases. Motorcycle safety by utilizing GSM, bluetooth and GPS the rights solution for such a thing. To make it easier to access the security system an application is needed as a remote control to activate the security system by utilizing bluetooth communications, so an a Android application is made as remote control motorcycle Safety. Of the application there is a button to choose the security mode you want to use, there is a parking mode while the vehicle is parked and drivers while driving. There is also a feature to find the location of the motor with the help of GPS, we can access by means of sending a message to the security system. Of test results obtained, the delivery of data from the mobile phone to the protection system can be delivered with maximum distance 14.6 m and of the distance protection system will be active when deprivation forced on the road by way of severing the flow of electricity on a CDI, blocking the flow of gasoline and siren will beep as a warning sign, GPS does not function properly when the vehicle is in the building so that the satellite can not send data coordinates of the vehicle. So the success of the whole system reaches 80%.

Keywords : *Security System, Bluetooth, GPS, Android Applications*

-----Halaman ini sengaja dikosongkan-----

KATA PENGANTAR

Pertama-tama marilah kita ucapkan dan puji syukur kepada Allah SWT Yang Maha Esa karena limpahan rahmat yang telah diberikan oleh-Nya kepada kita semua.

Tugas Akhir ini disusun untuk memenuhi sebagian persyaratan guna menyelesaikan pendidikan Diploma-3 pada Departemen Teknik Elektro Otomasi, Fakultas Vokasi, Institut Teknologi Sepuluh Nopember Surabaya dengan judul:

PENGAMAN KENDARAAN BERMOTOR DENGAN REMOTE ANDROID BERBASIS GSM, GPS, DAN *BLUETOOTH*

Penulis mengucapkan terima kasih atas bantuan dan motivasi yang diberikan baik secara langsung ataupun tidak langsung dalam pengerjaan Tugas Akhir kepada:

1. Kedua orang tua penulis yang senantiasa memberikan doa dan dukungan.
2. Bapak Eko Pramunanto,S.T.,M.T., selaku Dosen Pembimbing pada Departemen Teknik Elektro Otomasi, Fakultas Vokasi, Institut Teknologi Sepuluh Nopember Surabaya
3. Angkatan ANDROMEDA yang senantiasa berjuang bersama-sama dalam Tugas Akhir.

Penulis menyadari bahwa penyusunan laporan ini masih belum sempurna. Untuk itu, penulis sangat mengharapkan adanya kritik dan saran demi kemajuan di kemudian hari. Akhir kata, penulis mohon maaf apabila ada kesalahan selama kegiatan Tugas Akhir berlangsung hingga tersusunnya laporan ini.

Surabaya, 18 Juli 2017

Penulis

-----Halaman ini sengaja dikosongkan-----

DAFTAR ISI

HALAMAN

HALAMAN JUDUL i
HALAMAN JUDUL iii
PERNYATAAN KEASLIAN TUGAS AKHIR v
HALAMAN PENGESAHAN vii
ABSTRAK ix
ABSTRACT xi
KATA PENGANTAR xiii
DAFTAR ISI xv
DAFTAR GAMBAR xix
DAFTAR TABEL xxiii

BAB I PENDAHULUAN 1
1.1 Latar Belakang 1
1.2 Permasalahan 1
1.3 Batasan Masalah 2
1.4 Tujuan 2
1.5 Metodelogi Penelitian 2
1.6 Sistematika Laporan 3
1.7 Relevansi 4

BAB II TEORI DASAR 5
2.1 Arduino Pro Mini 5
2.2 Modul SIM808 6
2.3 *Reed Switch* 6
2.4 *Driver Relay* 7
2.5 *Power Supply* 8
2.6 *Power Bank* 8
2.7 AT *Command* 8
2.8 GPS (*Global Positioning System*) 9
2.9 *Selenoid Valve* 9
2.10 CDI 10
2.11 Sirine 10
2.12 *Operating System* Android 11
2.13 Komunikasi *Bluetooth* 13
2.14 App Inventor 14

BAB III PERANCANGAN DAN PEMBUATAN ALAT 15
3.1 Perancangan *Hardware* 16
3.1.1 Rangkaian *Power Supply*.....16
3.1.2 Rangkaian *Driver Relay* 18
3.1.3 Penempatan Sensor *Reed Switch* 20
3.1.4 Penempatan Bagian Box dan Supply pada Kendaraan 21
3.1.5 Konfigurasi Arduino Pro Mini dengan SIM808 22
3.2 Perancangan Software 23
3.2.1 *Flowchart* 23
3.2.2 AT *Command* SIM808 28
3.2.3 Bagian Deklarasi 28
3.2.4 Kepala Program.....29
3.2.5 Program Kepala.....31
3.2.6 Setup Mode dan Monitor Program 34
3.2.7 Perancangan Aplikasi Android.....43

BAB IV PENGUJIAN DAN PENGUKURAN 55
4.1 Pengukuran Rangkaian *Power Supply*.....55
4.2 Pengukuran Jarak Koneksi *Bluetooth*.....56
4.3 Pengujian Pengiriman Data dari Aplikasi Android ke SIM808.....57
4.4 Pengujian pengiriman SMS ke SIM808 58
4.5 *AT Command* 58
4.6 Data Tegangan *Selenoid Valve* dan Sirine.....59
4.7 Data *Reed Switch*.....60
4.8 Data GPS 61
4.9 Pengujian Sistem Keseluruhan 63

BAB V PENUTUP 69
5.1 Kesimpulan.....69
5.2 Saran.....69

DAFTAR PUSTAKA.....71

LAMPIRAN A 73
A.1. Perancangan Perangkat Lunak.....73
A.2. *Block* Program pada App Inventor.....94
A.3. Konfigurasi Rangkaian *Driver Relay*, Arduino dan SIM808 98
A.4. Konfigurasi Rangkaian *Power Supply* dan *Power Bank*....98

LAMPIRAN B 99
B.1. DATASHEET SIM808 99
B.2. DATASHEET ARDUINO PRO MINI 104
B.3. DATASHEET TRANSISTOR BC547..... 105

DAFTAR RIWAYAT HIDUP..... 107

-----Halaman ini sengaja dikosongkan-----

DAFTAR GAMBAR

HALAMAN

Gambar 2.1 Arduino Pro Mini..... 6
Gambar 2.2 Modul SIM808 6
Gambar 2.3 *Reed Switch*..... 7
Gambar 2.4 (a) Skema *Relay* (b) *Relay* 7
Gambar 2.5 *Selenoid Valve* 9
Gambar 2.6 CDI pada Kendaraan..... 10
Gambar 2.7 Sirine..... 10
Gambar 3.1 Diagram Fungsional Alat..... 15
Gambar 3.2 Skematik Rangkaian *Power Supply* 17
Gambar 3.3 Rangkaian *Power Supply* 18
Gambar 3.4 Skematik Rangkaian *Driver Relay*..... 18
Gambar 3.5 Rangkaian *Driver Relay.* 19
Gambar 3.6 Bagian Kontrol Sistem Keamanan Sepeda Motor 19
Gambar 3.7 Letak Sirine Pada Sepeda Motor 20
Gambar 3.8 Letak Selenoid Valve Pada Sepeda Motor..... 20
Gambar 3.9 Letak Sensor *ReedSwitch* pada Sepeda Motor. 21
Gambar 3.10 Letak *Box Supply* Sistem Pengaman Kendaraan Bermotor 21
Gambar 3.11 Letak Box Kontrol Sistem Pengaman Kendaraan Bermotor. 22
Gambar 3.12 Konfigurasi Arduino dengan SIM808 22
Gambar 3.13 *Flowchart* Bagian 1 24
Gambar 3.14 *Flowchart* Bagian 2a..... 25
Gambar 3.15 *Flowchart* Bagian 2b 26
Gambar 3.16 *Flowchart* Bagian 3 27
Gambar 3.17 Deklarasi *Library*, Konstanta dan Variabel 29
Gambar 3.18 Void *Setup* Program Arduino 30
Gambar 3.19 Program Utama Perulangan..... 31
Gambar 3.20 Sub Program Utama *Accept Bluetooth*. 32
Gambar 3.21 Sub Program Baca Pesan..... 33
Gambar 3.22 Pengolahan SMS Perintah ADMIN..... 35
Gambar 3.23 Pengolahan SMS Perintah USER 36
Gambar 3.24 Monitoring Perintah USER..... 37
Gambar 3.25 Perintah *Dissble*..... 38

Gambar 3.26 Perintah Dukungan Sistem. 38
Gambar 3.27 Perintah Menggunakan Mode Parkir. 39
Gambar 3.28 Interupt Pada Mode Parkir. 40
Gambar 3.29 *Loop* Mode Parkir 41
Gambar 3.30 Perintah Menggunakan Mode Driver. 41
Gambar 3.31 *Loop* Pada Mode Driver. 42
Gambar 3.32 Monitoring Koneksi *Bluetooth*. 42
Gambar 3.33 Tampilan *Design View* 43
Gambar 3.34 Rancangan Tampilan *Log In* pada Aplikasi Android. 44
Gambar 3.35 Rancangan Tampilan Menu Utama 45
Gambar 3.36 Rancangan Tampilan Menu Mode Keamanan 45
Gambar 3.37 Rancangan Tampilan Menu Kirim Pesan 46
Gambar 3.38 Rancangan Tampilan pada Mode USER 46
Gambar 3.39 *Flowchart* Aplikasi Android 47
Gambar 3.40 Realisasi Tampilan *Log In* pada Aplikasi Android 49
Gambar 3.41 (a) Realisasi Tampilan Menu Utama pada Aplikasi Android (b) Realisasi Tampilan Menu Mode Keamanan pada Aplikasi Android 49
Gambar 3.42 (a) Realisasi Tampilan Menu Kirim Pesan (b) Realisasi Tampilan pada Mode User 50
Gambar 3.43 *Block* Program Status Baterai 52
Gambar 3.44 *Block* Program Kirim Data Berulang 52
Gambar 3.45 *Block* Program Kirim Data *Driver* 53
Gambar 3.46 *Block* Program Kirim Data Parkir 53
Gambar 3.47 *Block* Program Kirim SMS 54
Gambar 3.48 Block Program Starter Google Maps 54
Gambar 4.1 Pengukuran Rangkaian *Power Supply* saat Diberi Beban 55
Gambar 4.2 Void *Accept Bluetooth* 57
Gambar 4.3 Pengujian Data GPS 62
Gambar 4.4 Kondisi Awal Sepeda Motor 63
Gambar 4.5 Tampilan Mode Parkir Aktif 64
Gambar 4.6 Sepeda Motor Dihidupkan Secara Paksa 64
Gambar 4.7 Sepeda Motor Dicuri dengan Cara Didorong 65
Gambar 4.8 Sistem Keamanan Telepon Pengguna 65
Gambar 4.9 Tampilan Mode *Driver* Aktif 66
Gambar 4.10 Pengujian Mode *Driver* 66
Gambar 4.11 Tampilan *Dissable* Sistem 67

Gambar 4.12 (a) Kirim Pesan Lokasi Motor (b) Pesan Masuk Koordinat Lokasi Kendaraan..... 68
Gambar 4.13 Tampilan pada Google Maps..... 68
Gambar A.1 Konfigurasi Rangkaian *Driver Relay*, Arduino dan SIM808..... 98
Gambar A.2 Konfigurasi Rangkaian *Power Supply* dan *Power Bank* 98

-----Halaman ini sengaja dikosongkan-----

DAFTAR TABEL

HALAMAN

Tabel 3.1 Penjelasan *Icon* pada Aplikasi Android 50
Tabel 4.1 Hasil Pengukuran *Power Supply* Tanpa Beban 55
Tabel 4.2 Hasil Pengukuran *Power Supply* dengan Beban 56
Tabel 4.3 Hasil Pengukuran Jarak Koneksi *Bluetooth* 56
Tabel 4.4 Pengujian Pengiriman Data dari Aplikasi Android 57
Tabel 4.5 Pengujian Pengiriman SMS ke SIM808 58
Tabel 4.6 Pengukuran Kecepatan Respon SIM808 59
Tabel 4.7 Pengukuran Tegangan *Valve* 60
Tabel 4.8 Pengukuran Tegangan Sirine 60
Tabel 4.9 Pengukuran Jarak *Reed Switch* 61
Tabel 4.10 Pengujian GPS 61
Tabel 4.11 Pengujian Lokasi Kendaraan. 67

-----Halaman ini sengaja dikosongkan-----

BAB I
PENDAHULUAN

1.1 Latar Belakang

Pada zaman yang moderen ini perkembangan teknologi berkembang pesat, tapi berbeda dengan teknologi pengaman kendaraan saat ini. Dikarenakan pengaman kendaraan saat ini masih belum bisa mengamankan kendaraan dengan baik. Contohnya pengaman dengan menggunakan magnet yang masih bisa dibobol, pengaman menggunakan sensor getaran untuk mendeteksi bahwa mesin dinyalakan secara paksa yang tidak akan berfungsi ketika kendaraan dicuri dengan cara di dorong, apalagi saat terjadi perampasan kendaraan secara paksa menggunakan senjata tajam, tidak ada pengaman saat kejadian itu terjadi. Untuk itu diperlukan sebuah sistem pengaman kendaraan yang dapat mencegah kasus kriminalitas seperti itu.

Pengaman kendaraan bermotor dengan memanfaatkan GSM, *bluetooth* dan GPS menjadi solusi yang tepat untuk hal seperti itu. Untuk mempermudah mengakses sistem pengaman tersebut dibutuhkan suatu aplikasi yang berfungsi sebagai remot kendali untuk mengaktifkan sistem keamanan dengan memanfaatkan komunikasi *bluetooth*, maka dibuatlah aplikasi Android sebagai remot kendali pengaman kendaraan bermotor. Terdapat tiga tombol pada aplikasi yaitu tombol mode parkir berfungsi mengaktifkan pengaman kendaraan pada saat parkir, pengaman bekerja saat kendaraan dihidupkan tanpa sepengetahuan pengguna dan ketika kendaraan didorong, tombol mode *driver* berfungsi mengaktifkan pengaman saat berkendara, bekerja saat koneksi *bluetooth* antara *handphone* pengguna dan *bluetooth* pada kendaraan terputus, dan tombol *dissable* yang berfungsi untuk mematikan sistem keamanan

1.2 Permasalahan

Pengaman kendaraan saat ini belum bisa dihandalkan oleh pemilik sepeda motor, dikarenakan jika pemilik dalam keadaan jauh dari tempat parkir sepeda motor tersebut maka pemilik tidak dapat memantau keadaan sepeda motornya, dan tidak adanya sistem saat terjadi insiden perampasan kendaraan secara mendadak.

1.3 Batasan Masalah

Batasan masalah dalam pembutan alat Tugas Akhir yang berjudul Pengaman Kendaraan Bermotor dengan Remote Android Berbasis GSM, GPS, dan *Bluetooth* diuraikan sebagai berikut :

1. Daya baterai *handphone* yang digunakan untuk konektivitas dengan *bluetooth* cepat habis yang mengharuskan mematikan sistem keamanan.
2. Ketika motor hilang, pelacakan melalui komunikasi SMS hanya bisa ketika alat mendapatkan sinyal provider yang cukup.
3. Pengguna tidak bisa melihat keadaan pulsa pada *sim card*.
4. Menggunakan *handphone* berbasis Android

1.4 Tujuan

Tujuan dari Tugas Akhir ini adalah membuat suatu sistem keamanan kendaraan bermotor yang dapat dikendalikan melalui *handphone* untuk mencegah tindakan pencurian dan perampasan kendaraan bermotor. Dari uraian tersebut, maka dapat dibagi menjadi dua tujuan dalam Tugas Akhir ini, yaitu:

1. Membuat aplikasi Android yang berfungsi sebagai remot kendali pengaman kendaraan bermotor.
2. Mengkombinasikan komunikasi SIM808 dengan Arduino dan perangkat Android**.**
3. Mengembangkan sistem keamanan kendaraan yang sudah ada.

1.5 Metodelogi Penelitian

Penelitian dilakukan melalui beberapa tahapan metodelogi yaitu, tahap persiapan, tahap perancangan dan pembuatan alat, tahap pengujian dan analisa, dan yan terakhir adalah penyusunan laporan berupa buku Tugas Akhir.

Pada tahap persiapan dilakukan studi literatur dengan pencarian data, bahan, literatur, dan melakukan perbandingan dengan alat yang sudah ada di pasaran. Dimana literatur diperoleh dari makalah desain dan pembuatan alat pengaman sepeda motor dengan sistem kontrol Arduino, mengontrol *on/off* suatu sistem dengan menggunakan *bluetooth*. Pada tahap perancangan dan pembuatan alat akan dilakukan perancangan perangkat keras (*hardware*) dan perangkat lunak (*software*) yang menunjang alat Tugas Akhir supaya berjalan

dengan baik. *software* yang akan dibuat adalah program pada Arduino serta aplikasi pada Android**.**

Pada tahap terakhir akan dilakukan pengujian dan pengukuran terhadap alat atau komponen yang digunakan agar mempermudah mahasiswa untuk memecahkan permasalahan pada alat yang dikerjakan. Kemudian dari hasil pengujian dan pengukuran akan ditarik kesimpulan dari pengujian alat yang sudah dilakukan. Tahap terakhir metode yang dilakukan adalah penyusunan laporan dalam bentuk buku Tugas Akhir.

1.6 Sistematika Laporan

Penyusunan laporan Tugas Akhir ini akan dibagi menjadi lima Bab dengan sistematika sebagai berikut :

Bab I. Pendahuluan

Bab ini meliputi latar belakang, permasalahan, batasan masalah, tujuan, metodelogi penelitian, sistematika laporan, dan relevansi.

Bab II. Teori Dasar

Bab ini menjelaskan tentang teori dasar dari perangkat keras (*hardware*) yang digunakan yaitu, Arduino Pro Mini, Modul SIM808, *power supply, power bank, operating system* Android**,** *reed switch, driver relay, AT Command,* GPS (*Global Positioning System), Selenoid valve,* CDI, Sirine, komunikasi *bluetooth* dan perangkat lunak (*software*) yang dibuat menggunakan App Inventor .

Bab III. Perancangan dan Pembuatan Alat

Bab ini membahas tentang perancangan dan pembuatan perangkat keras (*hardware*) dan perangkat lunak (*software*) yang menunjang alat Tugas Akhir.

Bab IV. Pengujian dan Pengukuran

Bab ini memuat tentang pemaparan hasil pengujian dan pengukuran alat pada keadaan sebenarnya. Seperti pengujian *power supply*, pegukuran jarak koneksi *bluetooth*, pengujian pengiriman data dari aplikasi Android serta pengujian pengiriman SMS ke SIM808

Bab V. Penutup

Bab ini berisi kesimpulan dan saran dari hasil pembahasan yang telah diperoleh.

1.7 Relevansi

1. Untuk meningkatkan sistem pengaman pada kendaraan.
2. Pengendalikan sistem pengaman dengan menggunakan *handphone*
3. Mengatasi masalah pencurian kendaraan yang semakin meningkat.

BAB II
TEORI DASAR

Bab ini membahas mengenai teori dasar peralatan dari Tugas Akhir yang berjudul Pengaman Kendaraan dengan Remote Android Berbasis GSM**,** GPS, dan *Bluetooth* yang meliputi Arduino Pro Mini**,** Modul SIM808**,** *power supply, power bank, operating system* Android, *reed switch, driver relay, AT Command***,** GPS (*Global Positioning System), Selenoid valve,* CDI, Sirine, komunikasi *bluetooth* dan perangkat lunak (*software*) yang dibuat menggunakan App Inventor

2.1 Arduino Pro Mini

Arduino Pro Mini adalah *board* mikrokontroler yang dulunya menggunakan ATMega168 dan kemudian ditingkatkan menggunakan ATMega328. *Board* ini memiliki 14 *pin* digital *input/output*, 8 *pin* analog, tombol *reset*, sebuah resonator. Karena tidak terdapat *pin header* yang tersambung dengan konektor *board* terdapat lubang-lubang yang dapat dipasang *pin header* sendiri untuk mengkoneksikan pada konektor sesuai kebutuhan. Terdapat 6 *pin header* yang dapat dihubungkan ke kabel FTDI atau ke kabel USB adapter lainnya untuk memberikan tegangan dari USB dan berkomunikasi antara komputer dengan Arduino Pro Mini. *Board* ini memiliki tegangan 5V dan menjalankan *bootloader* dengan frekuensi kristal 16MHz. Adapun beberapa pin memiliki fungsi khusus.

Serial: 0 (RX) dan 1 (TX), digunakan untuk menerima (RX) dan mengirimkan (TX) TTL data serial, *pin* ini terhubung ke TX-0 dan RX-1 *pin header* enam *pin*. Interupsi *eksternal*: 2 dan 3, pin ini dapat dikonfigurasi untuk memicu interupsi pada nilai rendah, naik atau turun, atau perubahan nilai. PWM: 3, 5, 6, 9, 10, dan 11, memberikan 8-bit PWM *output* dengan fungsi analogWrite (). SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK), pin ini mendukung komunikasi SPI, yang, meskipun disediakan oleh *hardware*, saat ini tidak termasuk dalam bahasa Arduino. LED: 13, ada *built-in* LED terhubung ke pin digital 13. Ketika pin adalah nilai HIGH, LED menyala, ketika pin LOW, LED mati. I2C A4 (SDA) dan A5 (SCL). Dukungan I2C (TWI) komunikasi menggunakan *library Wire*. Bentuk dari Arduino Pro Mini dapat dilihat pada Gambar 2.1.

Gambar 2.1 Arduino Pro Mini.

2.2 Modul SIM808

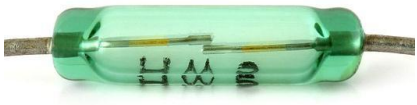
Modul SIM808 adalah modul GSM/GPRS Quad-Band 850/900/1800/1900MHz yang menggabungkan teknologi GPS untuk navigasi satelit dan *bluetooth* 3.0+EDR. Dengan fitur konsumsi daya ultra-rendah dalam mode tifur dan terintegrasi dengan sirkuit pengisian baterai Li-Ion. GPS memiliki sensitivitas yang tinggi dengan 22 tracking dan 66 receiver. Selain itu, juga mendukung A-GPS yang tersedia untuk lokasi dalam ruangan. Modul ini dikendalikan oleh perintah AT melalui UART dan mendukung level logika 3,3V dan 5V. Bentuk modul SIM808 dapat dilihat pada Gambar 2.2.

Gambar 2.2 Modul SIM808

2.3 *Reed Switch*

Reed Switch adalah sensor magnet yang akan aktif ketika berada didekat medan magnet. Cara kerja *reed switch* yaitu ketika dilewati medan magnet kedua lempengan tipis didalamnya akan tertarik dan akan terhubung, dan ketika medan listrik menjahui-nya posisi kedua lempengan akan terbuka kembali sehingga lempengan

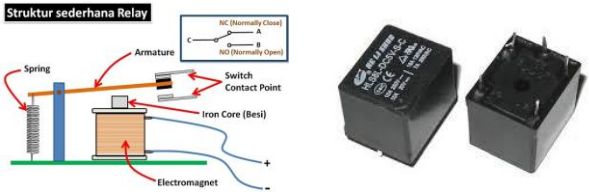
didalamnya berfungsi seperti sakelar pada umumnya. *Reed swich* hanya dapat dihubungkan ke beban yang kecil saja seperti *relay* maupun modul. Bentuk dari *reed switch* dapat dilihat pada Gambar 2.3



Gambar 2.3 *Reed Switch*

2.4 *Driver Relay*

Relay merupakan saklar yang di operasikan secara listrik. *Relay* memiliki 2 bagian utama yaitu koil elektromagnetik dan mekanik. Koil elekromagnetik berupakan sebuah kumparan yang akan menghasilkan medan magnet ketika dialiri arus listrik. Dan mekanik tersebut merupakan sebuah lempengan plat kecil yang terhubung dengan kontak keluran. *Relay* banyak macam tipe dan ukuran yang dapat dipilih sesuai dengan kriteria yang dibutuhkan seperti berapa banyak kontak yang dimiliki, daya hantar kontak tersebut dalam ukuran volt serta besar tegangan koil yang diperlukan melihat dari *supply* tegangan yang akan dipakai untuk mensupply *relay*.



(a) (b)

Gambar 2.4 (a) Skema *Relay* (b) *Relay*

Pada mekanik mempunyai 1 common dan 2 tipe kontak. Kontak NC (*normally close*) merupakan kontak yang pada keadaan normal tertutup dan pada saat aktif kontak tersebut terbuka. Sebaliknya kontak NO (*normally open*) pada kondisi awal kontak dalam keadaan terbuka dan setelah aktif akan menutup. Aktif tidaknya kontak tersebut diatur oleh koil. Jika koil di berikan arus listrik dan menghasilkan medan magnet. Medan magnet tersebut

akan menarik lempengan kontak sehingga kontak tersebut aktif untuk lebih jelasnya dapat dilihat pada Gambar 2.4 (a).

Driver relay merupakan sebuah rangkaian elektronika yang bertujuan untuk memperkuat arus listrik sehingga *relay* dapat bekerja dengan semestinya. Fungsi dari driver *relay* adalah sebagai penguat arus yang akan masuk ke koil *relay* difungsikan untuk kontroler yang mempunyai keluaran arus yang kecil seperti contohnya Arduino. Pada Arduino keluaran dari pin output tidak sampai melebihi 70mA. Sehingga tidak dapat mengaktifkan *relay* yang membutuhkan arus lebih dari 70mA. Untuk itulah driver *relay* ini diperlukan agar Arduino dapat mengontrol *relay* dengan keluaran arus yang kecil.

2.5 *Power Supply*

Power Supply adalah sebuah peralatan penyedia tegangan atau sumber daya untuk peralatan elektronika dengan prinsip mengubah tegangan listrik yang tersedia dari jaringan distribusi transmisi listrik menuju level yang diinginkan sehingga berimplikasi pada pengubahan daya listrik. Agar tegangan keluaran *power supply* lebih stabil, dapat digunakan suatu komponen IC yang disebut IC regulator, misalnya IC regulator 7812 atau IC regulator 7805. Hal ini memungkinkan keluaran DC catu daya dapat dibentuk sesuai kebutuhan. Pada umumnya *power supply* dapat menghasilkan *output* yang bersih, dengan tegangan output konstan dengan tingkat toleransi dari tegangan input, beban daya, juga suhu kerja, dengan tingkat konversi efisiensi 100%.

2.6 *Power Bank*

Power Bank atau bisa disebut dengan pengisi baterai portabel adalah suatu piranti yang digunakan untuk memasukan energi listrik ke dalam baterai yang bisa diisi ulang tanpa harus menghubungkan piranti tersebut pada outlet listrik. Pengisi baterai ini disebut portabel karena berbeda dengan pengisi baterai yang harus dihubungkan pada outlet listrik, pengisi portabel dapat digunakan tanpa harus menghubungkan pada perangkat listrik.

2.7 AT *Command*

AT *Command* (*Attention Command*) merupakan suatu perintah yang diberikan kepada perangkat elektronik yang mana dari *AT Command* tersebut mendapatkan feedback hasil dari perintah yang

dikirimkan. Dalam hal ini beberapa contoh perangkat yang menggunakan *AT Command* antara lain komputer, *bluetooth* dan modul simcom semua model. *AT Command* pada *bluetooth* biasanya digunakan untuk mengatur kecepatan *bit rate* komunikasi sedangkan pada produk simcom untuk kendali fitur dan informasi yang diperoleh.

2.8 GPS (*Global Positioning System*)

GPS digunakan untuk mencari lokasi atau tempat dari suatu benda. Merupakan sistem untuk menentukan letak di permukaan bumi dengan bantuan penyelarasan (*synchronization*) sinyal satelit. Sistem ini menggunakan 24 satelit yang mengirimkan sinyal gelombang mikro ke Bumi. Sinyal ini diterima oleh alat penerima di permukaan, dan digunakan untuk menentukan letak, kecepatan, arah, dan waktu. Untuk pengambilan GPS pada sim 808 dilakukan dengan mengirimkan *AT Command* berupa "AT+CGPSINF=0".

2.9 *Selenoid Valve*

Selenoid valve adalah katup yang digerakkan oleh energi listrik, mempunyai kumparan penggerak yang akan meghasilkan medan magnet jika dialiri oleh arus listrik AC maupun DC.pada Gambar 2.5 merupakan salah satu bentuk *selenoid valve.* Prinsip kerja dari *selenoid valve*/katup *selenoida* yaitu *selenoid* atau kumparan diberikan arus listrik maka kumparan tersebut akan menghasilkan medan magnet dan medan magnet tersebut akan menarik diafragma didalamnya sehingga selenoid tersebut aktif.

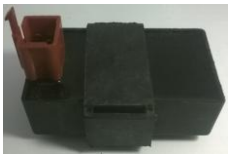
Gambar 2.5 *Selenoid Valve*

Terdapat dua tipe selenoid valve yaitu NC (*Normally Close)* dan NO (*Normally Open*). Pada tipe NC ketika diafragma berubah posisi karena tarikan medan magnet, katum yang semula tertutup

oleh diafragma akan terbuka dan sebaliknya pada tipe NO diafragma akan menutup katup ketika tertarik oleh medan magnet dari selenoid. Supply tegang untuk selenoid bermacam-macam besarnya dan dapat diberi arus AC maupun DC sesuai dengan tipe dan kriteria.

2.10 CDI

CDI atau *Capacitor Discharge Ignition* adalah sistem pengapian pada mesin pembakaran kendaraan bermotor dengan cara kerja menyimpan tegangan yang berasal dari *coil* kendaraan dan akan disalurkan ke busi untuk melakukan proses *discharge*. Pada Gambar 2.6 merupakan bentuk dari CDI. Selanjutnya dengan tegangan dari discharge CDI busi akan menghasilkan percikan api didalam ruang pembakaran sehingga ruang pembakaran yang berisikan udara jenuh dai bensin yang telah diproses di karburator akan menghasilkan tekanan akipat disulut oleh percikan api sehingga piston mampu bergerak.



Gambar 2.6 CDI pada Kendaraan

2.11 Sirine

Sirine adalah alat untuk membuat suara ribut. Berfungsi untuk memperingati masyarakat atau mengirimkan sinyal yang berisik dengan maksud tertentu, seperti tanda waktu habis, alarm, maupun peringatan bencana alam. Suara sirine yang cukup berisik dimaksudkan agar masyarakat tau dan mendengar dengan jelas. Sirine memiliki berbagai macam tipe, dan suara yang dihasilkan. Gambar 2.7 merupakan bentuk dari salah satu sirine.

Gambar 2.7 Sirine

2.12 *Operating System* Android

Operatif System Android adalah sistem operasi berbasis Linux yang dirancang untuk perangkat bergerak layar sentuh seperti telepon pintar dan komputer tablet. Android menyediakan platform terbuka bagi para pengembang untuk menciptakan aplikasi sendiri untuk digunakan pada bermacam peranti bergerak. Android awalnya dikembangkan oleh Android Inc, dengan dukungan finansial dari Google, yang kemudian membelinya pada tahun 2005, yang setelah itu terus menerus melakukan perkembangan ke teknologi yang lebih baik. Berikut ini merupakan perkembangan versi Android dari waktu ke waktu :

1. Android 1.0 (Astro)
   Sistem operasi ini pertama kali diperkenalkan pada tanggal 23 Desember 2008, dikarenakan masalah hak cipta nama Astro maka nama tersebut tidak jadi diresmikan. Ditemukan pula berbagai fitur-fitur synchronisasi yang berbasis layanan Google seperti Google Contact, Maps, Talk dan lain-lain.
2. Android 1.1 (Bender)
   Android versi 1.1 merupakan Android pertama yang memberikan sentuhan pada beberapa jenis aplikasinya seperti *system user interface* yang lebih baik. Dirilis pada 9 Februari 2009.
3. Android 1.5 (Cupcake)
   Diluncurkan pertama kali pada tahun 2009, fitur yang ditawarkan diantaranya ialah aplikasi kamus dan keyboard, kemampuan merekam dan memutar video dalam format MPEG-4 aplikasi widget yang lebih lengkap, serta kemampua transisi layar dan mengunggah video ke dalam Youtube.
4. Android 1.6 (Donut)
   Dirilis enam bulan setelah peluncuran OS Android versi Cupcake, terdapat penambahan dan pengembangan sistem kamera dan search engine atau mesin pencarian. Selain itu dilengkapi pula dengan dukungan bagi jaringan CDMA.
5. Android 2.0-2.1 (Eclair)
   Pertama kali dirilis pada 9 Desember 2009, terdapat peningkatan pada optimasi perangkat keras serta adanya Google Maps 3.1.2 serta penambahan browser dengan basis

HTML5. Fitur lampu kilat untuk kamera hingga 3,2 MP serta digital zion dan juga koneksi *Bluetooth* 2.1.

6. Android 2.2 (Froyo)
   Dirilis pada tanggal 20 Mei 2010, versi ini dapat menjalankan aplikasi Adobe Flash Player 10.1 serta menyertakan kemampuan menggunakan kartu memori SD untuk menyimpan aplikasi.
7. Android 2.3 (Gingerbread)
   Perkembangan pesat terjadi pada OS ini dengan optimasi kemampuan aplikasi serta Games dan disertakan *Near Field Communication* serta dukungan untuk penggunaan layar WXVGA.
8. Android 3.0-3.2.6 (Honeycomb)
   OS Android versi terbaru ini hadir pada bulan Februari 2011, versi ini menawarkan tampilan status bar yang semakin mudah untuk dilakukan kustomisasi oleh penggunanya. OS ini diluncurkan khusus mendukung perangkat tablet PC.
9. Android 4.0-4.0.4 (Ice Cream Sandwich)
   Hadir dengan tampilan yang lebih elegan dan menarik dibanding OS sebelumnya, semakin mempertegas konsistensi Android dalam melakukan transformasi dan perkembangan yang konsisten terhadap sistem operasi yang dimilikinya.
10. Android 4.1-4.3 (Jelly Bean)
    Terdapat pembaharuan peningkatan input keyboard yang kian lengkap dan adanya pencarian Google Now yang mampu memberikan berbagai informasi mengenai cuaca dan *traffic*.
11. Android 4.4+ (Kitkat)
    OS ini mampu memberikan tampilan status bar transparan serta dapat beroperasi secara optimal pada perangkat berspesifikasi rendah, selain itu juga Android memperbarui User Interface pada program Google Maps Navigation dan Alarm.
12. Android 5.+ (Lolipop)
    Pada Android versi ini terjadi pembaharuan dan peningkatan yang lebih terfokus pada hal yang berkaitan dengan desain dan performa. Dari segi performa, versi ini

menerapkan profesor 64 bit dan daya tahan baterai yang lebih baik.

13. Android 6.0+ (Marshmellow)
    Versi ini menyempurnakan asisten kontekstual dalam "Google Now on Tap"yang merupakan kemampuan baru dari aplikasi background jika perangkat sedang tidak digunakan, kemudian dukungan asli untuk pengenalan sidik jari dan konektor USB tipe-C.
14. Android 7.0 (Nougat)
    Terdapat peningkatan fitur dan kemampuan diantaranya ialah, Multi-window yang memungkinkan pengguna membuka dua aplikasi sekaligus secara split screen, fitur doze mode yang mengatur penggunaan baterai dengan membatasi aplikasi yang bekerja dibelakang layar.

2.13 Komunikasi *Bluetooth*

Bluetooth adalah sebuah teknologi komunikasi tanpa kabel (*wireless*) 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-host bluetooth* dengan jarak jangkauan yang terbatas (sekitar 10 meter). *Bluetooth* sendiri dapat berupa Cad yang bentuk dan fungsinya hampir sama dengan Cad yang digunakan untuk *Wireless Local Area Network* (WLAN) dan menggunakan frekuensi radio standar IEEE 802.11, hanya saja pada *bluetooth* mempunyai jarak jangkauan yang lebih pendek dan kemampuan transfer menggantikan atau menghilangkan penggunaan kabel di dalam melakukan pertukaran informasi.

Kanal komunikasi *bluetooth* dapat digunakan untuk komunikasi data dan suara, dengan dua jenis layanan asinkron dan sinkron yang dijelaskan sebagai berikut :

1. Transmisi asinkron (*Asynchronous Connection Les*/ ACL).
   Pada transmisi ini, setiap kali pengiriman data dilakukan perkarakter, antara satu karakter dengan yang lainnya tidak ada waktu yang tetap. Karakter dapat dikirimkan sekaligus atau beberapa karakter kemudian berhenti untuk waktu tidak tentu, kemudian dikirimkan lagi sisanya. Akibatnya penerima harus

melakukan sinkronisasi setiap kali pengiriman bit data, apakah data yang diterima sudah benar.

2. Transmisi sinkron (*Synchronous Connection Oriented*/ SCO).

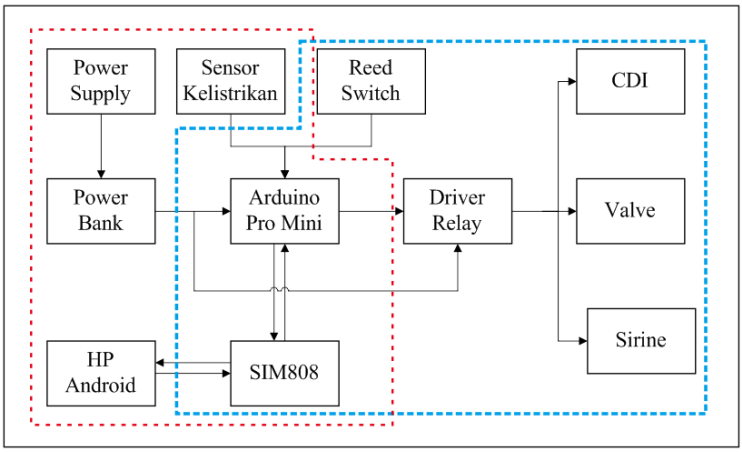
   Pada transmisi sinkron, blok atau france data dikirimkan setiap elemen 8 bit secara kontinu tanpa ada *delay*. Panjang tiap blok sama, waktu antara akhir dari bit terakhir dari satu karakter dan awal bit pertama karakter berikutnya harus nol atau kelipatan dari waktu satu karakter. Untuk mencapai sinkronisasi pengirim harus mengirim karakter khusus dan penerima harus mengenalinya. Transmisi sinkron digunakan untuk pengiriman *audio* atau suara dengan transmisi kecepatan tinggi, yang mentransmisikan satu blok data.

2.14 App Inventor

App Inventor adalah aplikasi web sumber terbuka yang awalnya dikembangkan oleh Google, dan saat ini dikelola oleh *Massachusetts Institute of Techology* (MIT). App Inventor memungkinkan pengguna baru untuk memprogram komputer untuk menciptakan aplikasi perangkat lunak bagi sistem operasi Android. App Inventor menggunakan antarmuka grafis, serupa dengan antarmuka pengguna pada Scratch dan StarLogo TNG, yang memungkinkan pengguna untuk men-*drag-and-drop* objek visual untuk menciptakan aplikasi yang bisa dijalankan pada perangkat Android. Dalam menciptakan App Inventor, Google telah melakukan riset yang berhubungan dengan komputasi edukasional dan menyelesaikan lingkungan pengembangan online Google.

BAB III
PERANCANGAN DAN PEMBUATAN ALAT

Bab ini akan membahas mengenai perencanaan dan pembuatan alat Tugas Akhir, yang meliputi perancangan perangkat keras (*hardware*) dan perangkat lunak (*software*) yang menunjang alat Tugas Akhir bekerja dengan baik. Diagram fungsional dari keseluruhan sistem Pengaman Kendaraan Bermotor dengan Remote Android berbasis GSM, *Bluetooth*, dan GPS pada Tugas Akhir ini terlihat seperti Gambar 3.1.



Gambar 3.1 Diagram Fungsional Alat

---------- : Bagian yang Dikerjakan Oleh Moch Aan Fahrizal

---------- : Bagian yang Dikerjakan Oleh Nasrul Arifianto

Gambaran kerja dari sistem secara keseluruhan yaitu pada sistem pengaman kendaraan ini terdapat dua mode keamanan yaitu mode parkir saat kendaraan diparkir dan mode *driver* saat berkendara. Terdapat aplikasi yang ter-instal pada *handphone* Android yang berisi tombol-tombol untuk mengganti mode keamanan. Untuk mengatur mode keamanan, *handphone* Android harus terkoneksikan terlebih dahulu dengan *bluetooth* SIM808. Saat mode parkir yang dipilih, ketika terjadi pencurian dengan membobol kontak dan saat mesin dinyalakan, sensor kelistrikan akan

memberikan sinyal ke Arduino untuk mengaktifkan sistem keamanan, *driver relay* akan bekerja untuk memutus arus pada CDI, mengaktifkan *valve* untuk memblokir aliran bensin, dan menyalakan sirine untuk tanda bahaya, seketika itu juga Arduino akan *misscall* ke pengguna untuk memberitahukan jika kendaraannya dalam bahaya. Saat mode *driver* yang dipilih, sistem keamanan ini memanfaatkan kelemahan dari koneksi *bluetooth* terhadap jarak untuk mengaktifkannya. Ketika kendaraan dirampas secara paksa oleh pencuri dan dibawa kabur saat itulah koneksi *bluetooth* dari *handphone* dan *bluetooth* pada kendaraan akan terputus karena terpaut jarak yang tidak bisa terjangkau oleh *bluetooth* maka sistem keamanan akan aktif, *driver relay* akan bekerja untuk memutus arus pada CDI, mengaktifkan *valve* untuk memblokir aliran bensin, dan membunyikan sirine tanda bahaya. Saat kendaraan dipinjam kita bisa mengubah *password bluetooth* untuk berjaga-jaga dengan mengirimkan SMS pada SIM808, mengirim pesan "USER" agar bisa dipinjam dan mengirim pesan "ADMIN" untuk mengembalikan *password bluetooth* seperti semula. Catu daya dari sistem keamanan ini didapatkan dari *power bank*, agar tidak cepat habis untuk men-*charge power bank* terdapat rangkaian *power supply* yang terhubung dengan kiprok motor dan tegangan berasal dari *coil* motor saat mesin dinyalakan. Pengguna dapat memonitoring lokasi kendaraan berada dengan mengirim pesan "LOKASI MOTOR", sistem pengaman akan mengirimkan koordinat kendaraan berada dan pengguna dapat mengakses koordinat dengan menggunakan Google Maps.

3.1 Perancangan *Hardware*

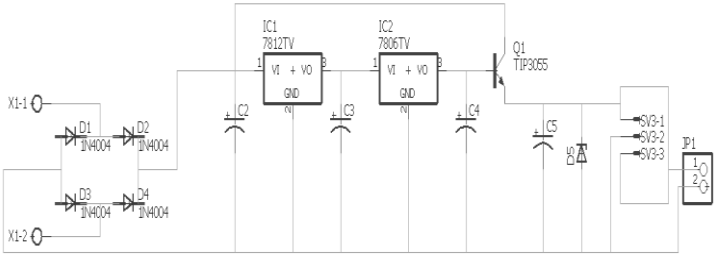
Perancangan *hardware* dilakukan dengan merancang rangkaian elektronika. Perancangan *hardware* ini meliputi :

1. Rangkaian *Power Supply*
2. Rangkaian *Driver Relay*
3. Konfigurasi Arduino Pro Mini dengan SIM808

3.1.1 Rangkaian *Power Supply*

Power supply adalah sebuah peralatan penyedia tegangan atau sumber daya untuk peralatan elektronika dengan prinsip mengubah tegangan listrik yang tersedia dari jaringan distribusi listrik menuju level yang diinginkan sehingga berimplikasi pada pengubahan daya listrik. Pada tugas akhir ini dibutuhkan *power supply* yang

menghasilkan tegangan ±5 Volt dan dapat digunakan untuk men-*charge* atau mengisi daya pada *power bank* yang digunakan sebagai catu daya untuk mencatu Arduino Pro Mini, modul SIM808, dan *driver relay* agar dapat bekerja dengan baik, memudahkan pengguna agar tidak melepas *power bank* pada pengaman kendaraan ini, dan juga dapat dipasang pada kendaraan bermotor yang sumber tengangannya melalui *coil* motor. *Coil* motor akan menghasilkan tegangan jika mesin kendaraan dihidupkan, besar tegangan yang dihasilkan oleh *coil* tergantung dari besar kecilnya tarikan *handle* gas pada kendaraan. Kemudian tegangan dari *coil* motor akan menuju kiprok motor untuk disearahkan tegangannya, dari kiprok motor yang merupakan pengearah untuk menngisi tegangan *accu* yang didapatkan dari tengangan koil kendaraan akan disambungkan menuju rangkaian *power supply* yang dibuat. Rangkaian *power supply* dapat dilihat pada Gambar 3.2.



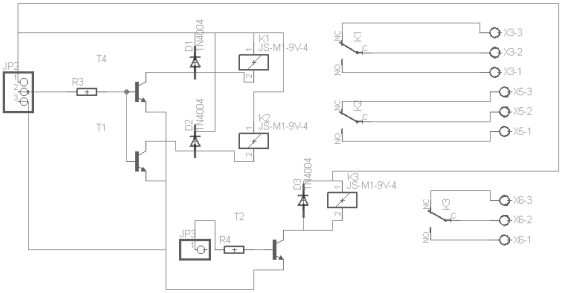
Gambar 3.2 Skematik Rangkaian *Power Supply*

Pada rangkaian di atas terdapat *rectifier* (penyearah) sebagai penyearah. Terdapat 4 kapasitor elco berkapasitas 3200 uF/25V, 1000uF/25V, 1000uF/16V, dan 470uF/16V yang berfungsi sebagai *filter*/penyaring untuk menghilangkan *ripple* yang sangat besar dari tegangan yang sudah disearahkan. Untuk menghasilkan tegangan dan arus DC yang stabil, diperlukan *voltage* regulator yang berfungsi untuk mengatur tegangan menjadi stabil dan membatasi tegangan agar sesuai dengan tegangan yang diinginkan. Pada rangkaian di atas *voltage* regulator yang digunakan adalah IC 7812 dan 7806, apabila tegangan yang masuk melebihi 12 Volt maka IC 7812 akan membatasi tegangan menjadi ±12 Volt, tegangan *output* dari IC 7812

akan dibatasi lagi oleh IC 7806 menjadi ±6 Volt, terdapat transistor TIP3055 yang berfungsi untuk menguatkan arus dan menurunkan sedikit tegangan, dikarenakan beban dari *power bank* yang besar, kemudian tegangan tersebut dibatasi oleh dioda zener 5,1 Volt agar keluarannya ±5,1 Volt. Terdapat sensor kelistrikan yang memberikan sinyal input ke mikrokontroler ketika kendaraan dinyalakan tanpa sepengetahuan pengguna untuk mengaktifkan sistem keamanan pada mode parkir. Rangkaian *power supply* yang sudah dibuat dapat dilihat pada Gambar 3.3.

Gambar 3.3 Rangkaian *Power Supply*

3.1.2 Rangkaian *Driver Relay*

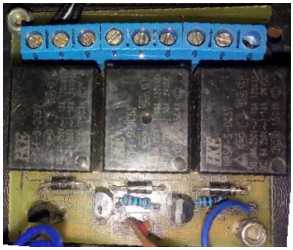


Gambar 3.4 Skematik Rangkaian *Driver Relay*

Pada Gambar 3.4 diatas transistor menggunakan tipe NPN BC 547 dikarenakan pada kondisi *switching* arus yang dapat dihantarkan oleh BC 547 sampai dengan 100mA sehingga *relay* yang mengkonsumsi arus di atas 70mA dapat diaktifkan melalui metode *switching* transistor. Jp 2 pin 1 dan 3 merupakan *supply* tegangan dengan arus yang lebih kuat dari luar dan jp 2 pin 2 dan jp3

berupakan masukan dari keluaran Arduino. Keluaran tersebut akan melalui kaki basis transistor sehingga ketika basis transistor di *triger* oleh tegangan maka kaki emitor dan kolektor transistor menjadi saklar penghubung sehingga koil *relay* dapat bekerja karena tehubung dengan *supply*.

Pada Gambar 3.4 diatas terdapat 2 *relay* bekerja sejara bersamaan dan 1 *relay* bekerja terpisah dikarenakan pada *relay* yang bekerja bersamaan digunakan untuk memutus ground pada CDI menggunakan *common* dan NC pin dan pada saat bersamaan *relay* yang satunya menghubungkan antara *valve* NO ke *power supply* sehingga aliran bensin dapat diblokir. Sedangkan yang 1 lagi merupakan kendali sirine yang bekerja secara terpisah agar dapat dikendalikan sendiri. *Supply* tegangan *valve* dan sirene didapat dari *accu*. Rangkaian *driver relay* yang telah dibuat dapat dilihat pada Gambar 3.5, Gambar 3.6 merupakan bentuk dari alat sistem kendaraan bermotor bagian kontrol yang terdiri dari Arduino, SIM808 dan *driver relay*. Pada gambar 3.7 merupakan letak dari sirine dan gambar 3.8 merupakan letak dari *selenoid valve* berada.



Gambar 3.5 Rangkaian *Driver Relay.*

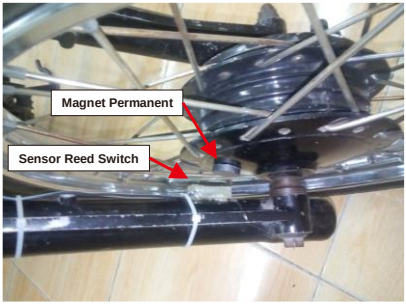
Gambar 3.6 Bagian Kontrol Sistem Keamanan Sepeda Motor

Gambar 3.7 Letak Sirine Pada Sepeda Motor

Gambar 3.8 Letak Selenoid Valve Pada Sepeda Motor

3.1.3 Penempatan Sensor *Reed Switch*

Sensor *Reed Switch* digunakan untuk pengaman pada mode parkir yang mana sensor ini akan merubah posisi kontak yang semula NO menjadi tertutup sehingga dapat mendeteksi perputaran roda pada kendaraan bermotor yang akan dikelolah lebih lanjut pada program Arduino. Pada Gambar 3.9 merupakan letak lokasi *Reed Switch* pada kendaraan bermotor. Ketika magnet tersebut melewati sensor reed switch satu kali maka reed switch akan memberikan perubahan kontak menjadi tertutup sebanyak 2 kali dikarenakan kontak reedswitch akan bereaksi pada saat melewati kutub utara dan kutub selatan magnet. Setelah itu Arduino akan mengelolah sinyal reedswitch tersebut menjadi data counter.



Gambar 3.9 Letak Sensor *ReedSwitch* pada Sepeda Motor.

3.1.4 Penempatan Bagian Box dan Supply pada Kendaraan

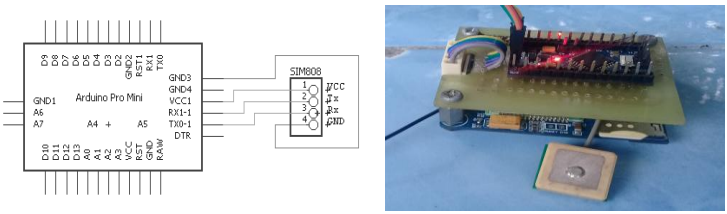
Penempatan dari sistem pengaman kendaraan bermotor diperlukan letak yang staregis sehingga tidak gampang di matikan. Sehingga sistem keamanan kendaraan bermotor ini tidak diletakkan didalam bos tetapi dimasukkan kedalam rongga kendaraan sehingga sulit untuk ditemukan. Letak dari sistem pengaman kendaraan bermotor ini dapat di lihat pada gambar 3.10 dan 3.11.

Gambar 3.10 Letak *Box Supply* Sistem Pengaman Kendaraan Bermotor

Gambar 3.11 Letak **Box** Kontrol Sistem Pengaman Kendaraan Bermotor.

3.1.5 Konfigurasi Arduino Pro Mini dengan SIM808

Dalam sistem pengaman kendaraan bermotor ini diperlukan Arduino sebagai mikrokontroler untuk menjalankan sistem sesuai dengan program yang telah dibuat dan modul SIM808 yang memilik fitur GSM, GPS, dan *bluetooth* yang digunakan sebagai komunikasi data yang akan digunakan pada alat Tugas Akhir ini. SIM808 menggunakan komunikasi serial melalui UART (*Universal Asynchronous Receiver Transmitter*) yang merupakan bagian perangkat keras komputer yang menerjemahkan bit-bit paralel data dan bit-bit serial. Oleh karena itu diperlukan konfigurasi antara mikrokontroler Arduino dengan SIM808 dengan menghubungkan pin Tx Arduino ke pin Rx SIM808 dan sebaliknya pin Rx Arduino ke pin Tx SIM 808. Tx disebut *transmitter* yang berfungsi untuk mengirim data atau mengeluarkan data yang akan diterima melalui Rx (*received*) dan sebaliknya Rx adalah jalur penerimaan data yang berguna menangkap data yang dikirim oleh Tx (*transmitter*). Konfigurasi antara Arduino dengan SIM808 dapat dilihat seperti Gambar 3.12.



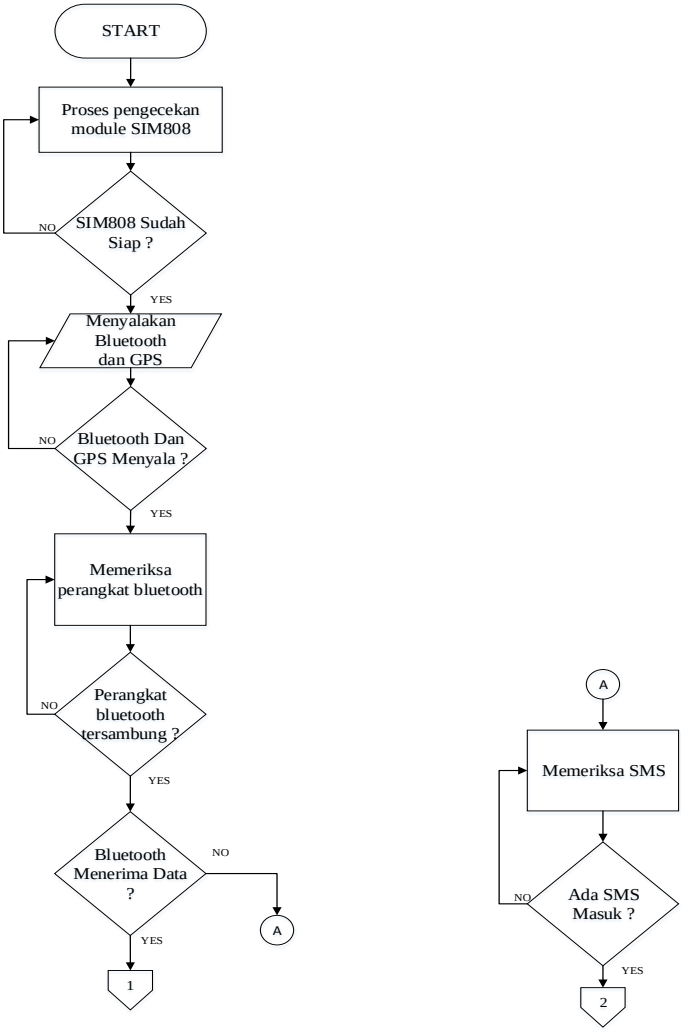
Gambar 3.12 Konfigurasi Arduino dengan SIM808

3.2 Perancangan Software

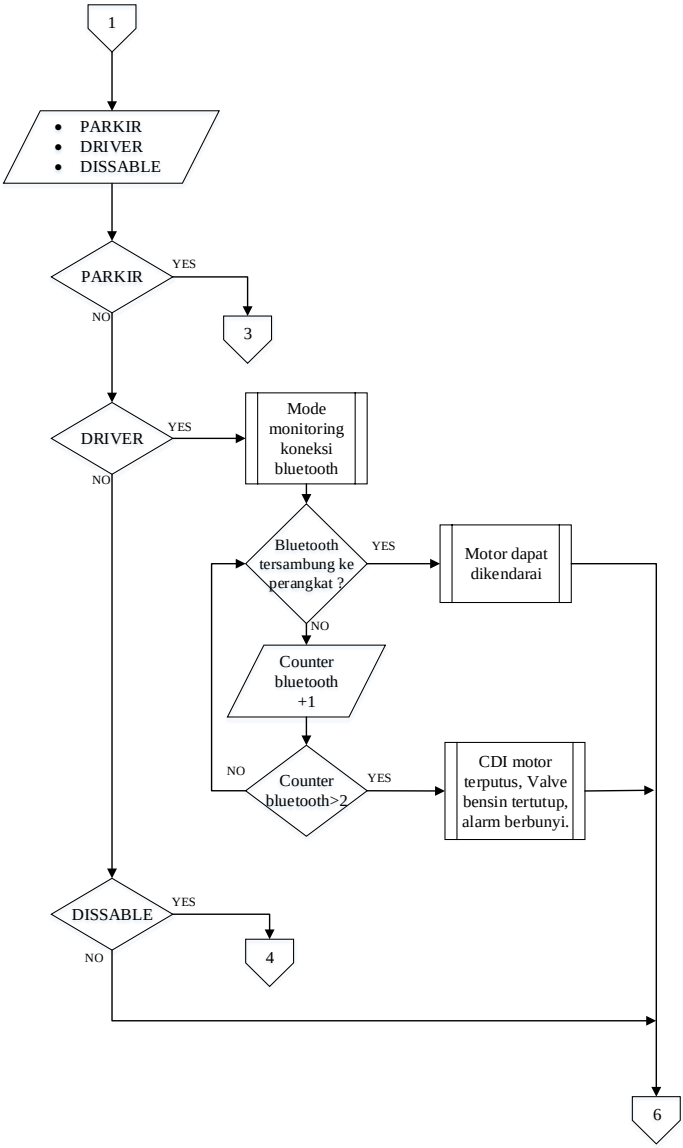
Dalam pembuatan alat ini, perangkat lunak yang dipakai untuk perancangan pemrograman pada aplikasi remote Android menggunakan aplikasi App Inventor yang berbasis *visual block progamming* sedangkan pemrograman untuk menerima data yang dikirimkan menggunakan Arduino**.**

3.2.1 *Flowchart*

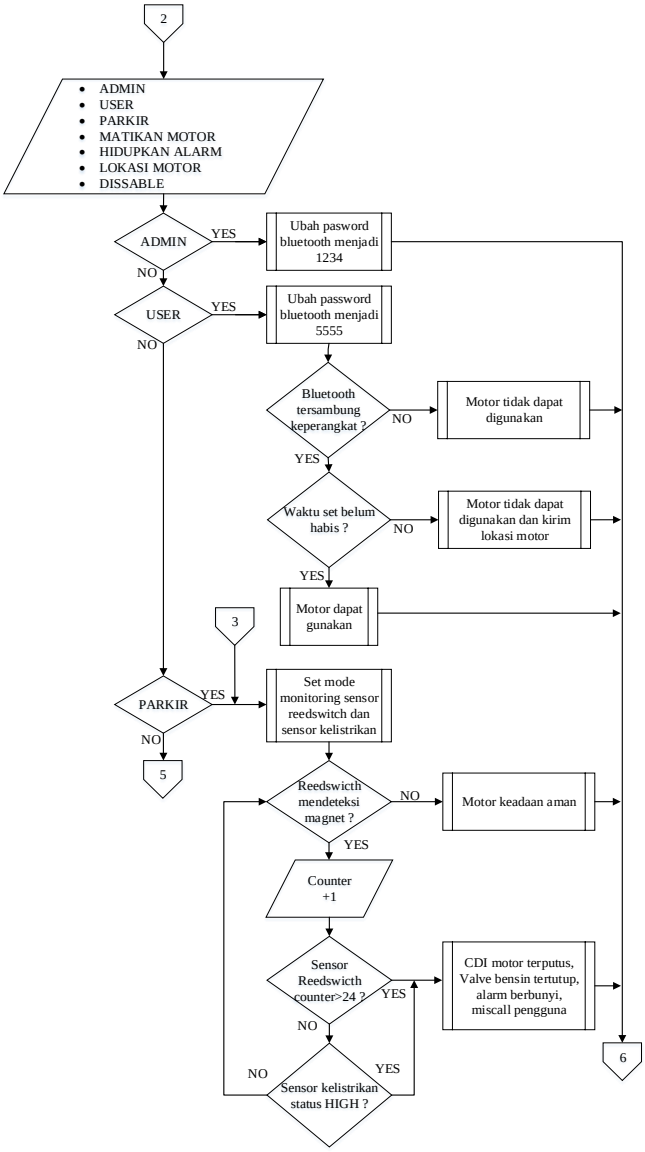
Pada *flowchart* pada Gambar 3.13 sampai Gambar 3.16 menjelaskan tentang cara kerja dari pengolahan data yang di terima dari SIM808 menjadi perintah, dan cara kerja dari masing masing mode yang ada.



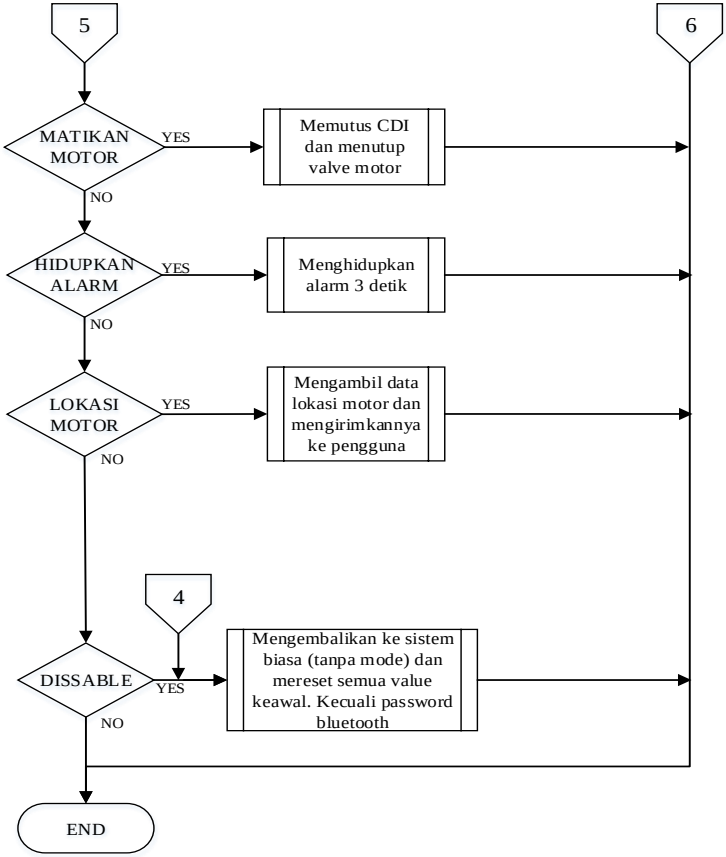
Gambar 3.13 *Flowchart* Bagian 1



Gambar 3.14 *Flowchart* Bagian 2a



Gambar 3.15 *Flowchart* Bagian 2b



Gambar 3.16 *Flowchart* Bagian 3

Pada *flowchart* telah dijelaskan bagaimana proses kerja dari pembuatan perangkat lunak sistem pengaman kendaraan bermotor ini. Yang nana pada saat awal setelah *start* hal yang perlu dilakukan adalah mempersiapkan semua perangkat agar dapat siap digunakan. Terdapat admin dan *user* yang mana *user* digunakan disaat motor dipinjam sedangkan admin untuk diri sendiri. Pada setiap mode memiliki proses keamanan yang berbeda beda.

Pada mode *driver* tujuan terhubungnya *bluetooth* dengan perangkat adalah ketika terjadi perampasan *bluetooth* akan terputus

karena jauh dari motor sehingga data yang dikirimkan *bluetooth* tidak akan sampai ke sistem. Sistem akan memastikan kalau benar-benar *bluetooth* terputus dengan cara menghitung data *loss*, ketika dipastikan putus maka sistem keamanan akan aktif yaitu *valve* bensin tertutup, mesin mati, dan alarm berbunyi. Berbeda lagi dengan mode parkir yang mana *reed swicth* akan membatasi jumlah rotasi roda karena *reed swicth* akan terus menghitung ketika mencapai satu rotasi penuh. Sehingga dengan jarak kurang lebih 20 meter maka sistem keamanan akan aktif, begitupun jika kontak diretas dan dinyalakan sistem keamanan juga akan aktif. Dan saat keamanan aktif maka *valve* bensin akan tertutup, CDI di putus, bunyi sirine serta menelepon pengguna untuk mengecek motornya.

3.2.2 AT *Command* SIM808

Module SIM808 dijalankan dengan cara menggunakan *AT Command*. AT *Command* yatiu sebuah komentar yang memiliki format tertentu yang dapat dimengerti oleh SIM808. Pada SIM808 terdapat ratusan tipe *AT Command* contohnya *AT command* yaitu "AT" (tanpa tanda petik) untuk mengetahui bahwa SIM808 sudah dapat bekerja atau mengalami masalah jika sistem bekerja akan merespon dengan "OK" jika tidak maka tidak ada respon atau terjadi masalah, AT+BTPOWER=1 untuk menyalakan *bluetooth* pada modul SIM808. Namun hanya beberapa *AT command* yang diperlukan dalam pemrograman alat ini. Keseluruhan *AT command* dan lama responnya dapat dilihat pada Bab 4.

3.2.3 Bagian Deklarasi

Pada bagian deklarai berisi sebuah pemanggilan *library*, pendaftaran konstanta atau variabel yang akan digunakan didalam program tersebut. Program dapat dilihat pada Gambar 3.17

```
#include <DFRobot_sim808.h>
#include <SoftwareSerial.h>

#define MESSAGE_LENGTH 160
#define BT_LENGTH 160
char message[MESSAGE_LENGTH];
int messageIndex = 0,BTstatus=0,adminservice=1,count=0;
int sensorlistrik=0,mode=0,countbt=0,koneksi=0,call=0,calling=0;
char btdata[BT_LENGTH];
char phone[16];
char datetime[24];
const int reedswitch=3;
const int listrik=2;
char tanggal[3];
char tanggal1[3];
char jam[3];
char jam1[3];
char menit[3];
char menit1[3];
char location[30];
DFRobot_SIM808 sim808(&Serial);
```

Gambar 3.17 Deklarasi *Library*, Konstanta dan Variabel

#include <DFRobot_SIM808.h> berfungsi memanggil *library* khusus SIM808 buatan Dfrobot. Didalam *library* tersebut berisikan program untuk memproses dan parse data yang diterima dari SIM808 melalui *AT command* sehingga pada *Skecth* Arduino menjadi lebih sederhana karena program di *library* siap panggil. Begitu pula dengan #include <SoftwareSerial.h> untuk memanggil *library* yang dipakai untuk digunakan dalam komunikasi serial. #define MESSAGE_LENGTH 160 dan #define BT_LENGTH2 160 digunakan untuk mendefinisikan banyaknya konstanta *value*. Lalu untuk tulisan awal berformat biru merupakan jenis variabel dan konstanta yang akan dipakai pada program tanda ";" berfungsi untuk mengakhiri statement pada baris tersebut. Dibaris terakhir terdapat DFRobot_SIM808 SIM808(&Serial) berfungsi untuk mendeklarasikan bahwa *library* DFRobot dapat menggunakan komunikasi serial. Komunikasi serial dilakukan pada pin RX dan TX pada masing-masing *device* yang saling terhubung berkebalikan.

Pada variabel yang dipakai kedalam program dapat mempengaruhi kinerja dari Arduino itu sendiri. Ketika variabel pada program melebihi 80% Arduino bekerja tidak maksimal.

3.2.4 Kepala Program

Pada setiap pemrograman terdapat sebuah kepala program kegunaannya yaitu untuk mengeset Arduino dengan kondisi tertentu

pada start awal agar program dibawahnya dapat bekerja dengan lancar dan terkendali. Kepala program tersebut berada didalam void setup. Void *setup* hanya di jalankan 1 kali. Dan pertama kali dijalankan sesaat setelah Arduino menyala.

```
void setup() {
  Serial.begin(9600);
  pinMode(reedswitch, INPUT);
  pinMode(listrik, INPUT);
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(7, OUTPUT);
  pinMode(8, OUTPUT);
  pinMode(LED_BUILTIN, OUTPUT);
  detachInterrupt(digitalPinToInterrupt(reedswitch));
  detachInterrupt(digitalPinToInterrupt(listrik));
  while(!sim808.init()) {
      Serial.print("Sim808 init error\r\n");
      delay(1000);
  }
  delay(3000);
  Serial.println("Init Success, module ready!");
  Serial.println("AT+CGPSPWR=1");
  delay (2000);
  Serial.println("AT+CGPSRST=0");
  delay (3000);
}
```

Gambar 3.18 Void *Setup* Program Arduino

Pada Gambar 3.18 merupakan program setup Arduino. Langkah pertama terlebih dahulu mengeset baud rate sebesar 9600 sehingga dapat berkomunikasi dengan SIM808 yang diset dengan baud rate yang sama. Setelah itu menentukan pin mode dari *input* dan *output*. Pada sistem ini menggunakan 2 sensor pengaman yang pertama yaitu *reed switch* yang akan mengkondisikan keadaannya sesuai dengan ada tidaknya medan magnet yang melintas didekatnya. Dan sensor listrik yang didapat dari *power supply* dan stabilisator yang mana ketika mesin menyala, maka keluaran stabilisator akan bermuatan energi listrik. Lalu setelah itu menentukan pin mode yang akan dipakai yang mana pin 4 digunakan untuk mematikan motor dan mengaktifkan *valve* elektrik dan pin 5 digunakan untuk membunyikan sirine, pin 6,7,8 digunakan sebagai lampu indikator pada tiap-tiap mode. Setelah itu, mennonaktifkan *interupt* terlebih dahulu dan mengaktifkan GPS pada saat awal mulai program.

3.2.5 Program Kepala

```
void loop() {
delay(200);
acceptBT();
delay(200);
sim808.isBTdata(btdata);
Serial.println(btdata);
BTproses(btdata);
while(koneksi==1){
  sim808.isBTdata(btdata);
  Serial.println(btdata);
  delay(300);
  BTproses(btdata);
  delay(1000);
  }
bacapesan();
delay(300);
sim808.getGPSlokasi(location);
Serial.println(location);
delay(300);
loopmode();
}
```

Gambar 3.19 Program Utama Perulangan

Void *loop* digunakan untuk menjalankan sebuah program berulang kali. Semua void yang didesain untuk diproses berulang kali harus dimasukkan kedalam void *loop*. Pada Gambar 3.19 hal yang pertama dilakukan yaitu mengecek keadaan *bluetooth*. Dapat dilihat pada Gambar 3.20 yang mana *delay* pada skecth ini bertujuan untuk menunggu SIM808 memproses perintah yang telah diberikan agar tidak terjadi error. Pertama kali yang dilakukan pada bagian acceptBT adalah memanggil library yaitu SIM808.isBTstatus(); untuk melihat status *bluetooth*. *Status* ini digunakan untuk mengetahui apakah *bluetooth* pada modul SIM808 ini mati, menyala, atau menerima permintaan koneksi dari perangkat lain. Status yang diberikan yaitu bernilai 0 saat *bluetooth* mati, bernilai 5 saat *bluetooth* menyala dan bernilai 25 pada saat *bluetooth* menerima permintaan koneksi dari perangkat lain. Sayangnya pada pemberian status ini SIM808 tidak memberikan status yang berbeda antara *bluetooth* yang hanya menyala dengan *bluetooth* yang sedang terhubung dengan perangkat lain yang hanya diberikan nilai status sama yaitu 5. Hal ini mempersulit teknik pengolahan data sistem

sehingga membutuhkan kecepatan pengolahan data yang lama untuk memastikan *bluetooth* tersebut terhubung perangkat lain atau standby.

```
void acceptBT(){
  delay(800);
  BTstatus = sim808.isBTstatus();
  delay(200);
  BTstatus = sim808.isBTstatus();
  Serial.println(BTstatus);
  delay(400);
  while(BTstatus==0){
  Serial.println("AT+BTSPPGET=1");
  delay (2000);
  Serial.println("AT+BTPOWER=1");
  delay (3000);
  BTstatus = sim808.isBTstatus();
  }
  delay (1000);
  while (BTstatus==25){
  Serial.println("AT+BTACPT=1");
  delay(3000);
  BTstatus = sim808.isBTstatus();
  }
}
```

Gambar 3.20 Sub Program Utama *Accept Bluetooth*.

Pencarian status *bluetooth* diberikan selama 2x dikarenakan pada momen pertama saat meminta status pasa *bluetooth* yang terjadi adalah timbul error data dikarenakan SIM808 yang belum siap untuk mengirim data tersebut serta Arduino yang juga belum siap menerima data yang pertama. Sehingga dibutuhkan 2x perintah agar bendapatkan hasil dari *bluetooth* status yang valid. Seteah itu masuk kedalam while yang mana while yang pertama ini bertujuan agar *bluetooth* tetap menyala sehingga saat nilai BTstatus=0 akan melakukan perulangan hingga Btstatus tidak sama dengan 0 lagi, AT+BTPOWER=1 digunakan untuk menyalakan *bluetooth* sedangkan AT+BTSPPGET=1 digunakan untuk membuka data *bluetooth* secara manual. lalu pada while yang kedua digunakan untuk meng accept permintaan koneksi dari perangkat lain yang akan mengontrolnya. AT+BTACPT=1 merupakan format memberi ijin koneksi kepada perangkat lain sehingga smartphone akan dapat mengirimkan data kepasa sistem pengaman.

Lalu setelah mendapatkan status dari *bluetooth* selanjutnya yaitu untuk memeriksa apakah ada data yang masuk atau tidak dengan memanggil program dari *library* yaitu SIM808.is-

BTdata(btdata). Yang akan memberikan hasil dari pembacaan pada variabel btdata. Setelah itu hasil tersebut akan diproses melalui BTproses(btdata) yang akan menentukan mode apa yang akan dijalankan.

Didalam while (koneksi==1) merupaka sebuah program yang digunakan untuk pengecekan apakah *bluetooth* tersambung dengan perang atau tidak. Difungsikan saat menggunakan mode *driver* maupun mode user. Variabel koneksi=1 hanya didapatkan ketika btdata menemukan data PING. Dikarekanan data PING dikirim secara berulang-ulang untuk menghindari penumpukan data yang bisa berakibat sistem lagging sehingga data tersebut dibuka terus menerus melalui while hingga tidak terdeteksi adanya data ping maka dapat diteruskan ke program selanjutnya.

```
void bacapesan(){
  messageIndex = sim808.isSMSunread();
    Serial.print("messageIndex: ");
    Serial.println(messageIndex);
   if (messageIndex > 0) {
      sim808.readSMS(messageIndex, message, MESSAGE_LENGTH, phone, datetime);
      delay(3000);
      sim808.deleteSMS(messageIndex);
      Serial.print("From number: ");
      Serial.println(phone);
      Serial.print("Datetime: ");
      Serial.println(datetime);
      Serial.print("Recieved Message: ");
      Serial.println(message);
      delay(5000);
      prosespesan(message);
}
}
```

Gambar 3.21 Sub Program Baca Pesan

Setelah dari pengontrolan *bluetooth* selanjutnya yaitu pengontrolan lewat pesan singkat atau SMS. Didalam program void loop tersebut diketahui memanggil sub program yang lain yaitu bacapesan yang berisikan seperti Gambar 3.21.

Messageindex digunakan untuk membetahukan jumlah pesan baru masuk yang diterima, seteah ada pesan masuk makan messageindex akan lebih besar dari 0 sehingga dapat diteruskan untuk membaca SMS dengan memanggil library "SIM808.readSMS(mes-sageIndex, message, MESSAGE_LENGTH, phone, datetime);" data tersebut akan masuk pada bagian variabel message untuk isi pesan, phone untuk nomor

yang mengirimkannya dan datetime untuk tanggal penerimaan pesan. Setelah itu SIM808 membutuh kan waktu 3 detik untuk dapat stabil sehingga memberikan *delay* adalah langkah agar SIM808 dapat bekerja maksimal kembali. Setelah itu pesan dang sudah dibaca di hapus lalu pada bagian terakhir yaitu memproses isi pesan tersebut.

Lalu pada bagian yang terakhir yaitu untuk mengambil data GPS dengan memanggilnya dari *library* dengan menggunakan "SIM808.getGPSlokasi(location);" dan data tersebut akan diletakkan pada bagian variabel location yang akan berisi koordinat latitude longitude. Ketika GPS tidak menerima data dari satelit maka silai location akan 0.000000,0.000000. yang menandakan bahwah GPS tidak dapat menerima sinyal data. Setelah itu program akan masuk kedalam bagian moniring yaitu loopmode.

3.2.6 Setup Mode dan Monitor Program

Pada bagian ini menjelaskan tentang memproses sebuah data yang masuk masuk baik melalui *bluetooth* maupun melalui SMS. Data yang masuk melalui *bluetooth* akan diproses pada void "BTproses(String btdata);" sedangkan data yang masuk lewat SMS akan diproses melalui "void prosespesan(String message);" proses ini lebih mirip dengan void *setup* yang berjalan 1x untuk menyiapkan Arduino pada posisi siap digunakan. Makan pada void proses bertujuan menjalankan 1x program ini untuk mengatur keperluan agar pada saat monitor program berjalan dengan baik. Pada proses yang berjalan 1x ini pada program digunakan untuk menentukan mode apa yang akan digunakan yang dapat dilihat pada variabel mode lalu mengatur posisi dari indikaor led pada sistem dan yang terakhir yaitu mengatur variabel pada posisi awal sesuai kebutuhkan.

Dan yang terakhir yaitu pada loopmode merupakan bagian dari *monitoring* pada mode yang telah dipakai, seperti contohnya saat mode parkir yang mana *counter* pada *reed swicth* di monitor terus yang berarti program harus berjalan berulang ulang untuk memeriksa keadaan. Sesuai dengan *flowchart* yang telah dibahas ada beberapa hal yang dapat dilakukan lewat SMS namu tidak dapat dilakukan dengan *bluetooth* dan sebaliknya. Hal yang dapat dilakukan dengan SMS antara lain seperti mengubah mode admin atau user dikarenakan admin dan user akan merestart dan mengganti pasoword *bluetooth* yang berakibat koneksi *bluetooth* akan terputus maka

penggantian user dan admin hanya dapat dilakukan lewat SMS sebaliknya pada mode driver diwajibkan *bluetooth* dan sistem untuk saling terhubung maka untuk memasuki mode driver tidak bisa melalui SMS. Berikut merupakan mengaturan dari subsistem yang telah dibuat.

```
if( message.indexOf("ADMIN") >= 0 ){
  detachInterrupt(digitalPinToInterrupt(reedswitch));
  detachInterrupt(digitalPinToInterrupt(listrik));
  Serial.println("AT+BTPOWER=0");
  delay (2000);
  Serial.println("AT+BTPOWER=1");
  delay (2000);
  Serial.println("AT+BTPAIRCFG=1,1234");
  delay (2000);
  Serial.println("AT+BTPAIRCFG=1,1234");
  delay (2000);
  Serial.println("AT+BTPOWER=0");
  delay (3000);
  adminservice=1;
  Serial.println("MODE ADMIN");
  digitalWrite(5, HIGH);
  delay(500);
  digitalWrite(5, LOW);
  delay(50);
  digitalWrite(5, HIGH);
  delay(500);
  digitalWrite(5, LOW);
  }
```

Gambar 3.22 Pengolahan SMS Perintah ADMIN

Pada Gambar 3.22 jika pada isi pesan menemukan kata "ADMIN" maka proses yang akan dijalankan 1x yaitu mematikan interupt yang hanya akan aktif ketika mode parkir. Lalu mematikan *bluetooth* dan menghidupkannya kembali lalu mengganti password *bluetooth* menjadi 1234 setelah itu *bluetooth* dimatikan dan variable adminservice menjadi 1 yang menandakan bahwa dapat menggunakan mode *driver* maupun parkir setelah itu merupakan indikator bunyi sirine.

```
if( message.indexOf("USER") >= 0 ){
  detachInterrupt(digitalPinToInterrupt(reedswitch));
  detachInterrupt(digitalPinToInterrupt(listrik));
  Serial.println("AT+BTPOWER=0");
  delay (2000);
  Serial.println("AT+BTPOWER=1");
  delay (2000);
  Serial.println("AT+BTPAIRCFG=1,5555");
  delay (2000);
  Serial.println("AT+BTPAIRCFG=1,5555");
  delay (2000);
  Serial.println("AT+BTPOWER=0");
  delay (3000);
  digitalWrite(6, HIGH);
  digitalWrite(8, HIGH);
  adminservice=0;
  sim808.getDateTime(tanggal, jam, menit);
  memcpy(tanggal1, tanggal, 3);
  memcpy(jam1, jam, 3);
  memcpy(menit1, menit, 3);
  Serial.println("MODE USER");
  mode=3;
  digitalWrite(5, HIGH);
  delay(300);
  digitalWrite(5, LOW);
  delay(100);
  digitalWrite(5, HIGH);
  delay(300);
  digitalWrite(5, LOW);
  }
```

Gambar 3.23 Pengolahan SMS Perintah USER

Pada Gambar 3.23 jika kata USER ditemukan pada isi pesan maka penggantian password *bluetooth* seperti langkah pada mode ADMIN namun menggunakan password yang berbeda yaitu 5555, setelah itu dikarenakan mode user maka adminservice=0 dan mengambil waktu mulai melalui library "SIM808.getDate-Time(tanggal, jam, menit);" dan data tersebut dikopikan kedalam variabel tanggal1 untuk data tanggal jam1 untuk data jam dan menit1 untuk data menit lalu mode di set menjadi mode=3.

```
if(adminservice==0){
  Serial.println("modeuser");
    if(countbt<2){
      digitalWrite(4,LOW);
  }
    if(countbt>2){
      digitalWrite(4,HIGH);
    }
sim808.getDateTime(tanggal, jam, menit);
Serial.println();
Serial.print("Waktu Sekarang: Tanggal ");
Serial.print(tanggal);
Serial.print(" ");
Serial.print(jam);
Serial.print("-");
Serial.println(menit);
Serial.print("Waktu Mulai: Tanggal ");
Serial.print(tanggal1);
Serial.print(" ");
Serial.print(jam1);
Serial.print("-");
Serial.println(menit1);
Serial.println("User mode berlangsung 1x24jam");
    if(tanggal!=tanggal1){
      if(jam==jam1){
      digitalWrite(4,HIGH);
      sim808.sendSMS(phone,location);
      adminservice=1;
      }
    }
  }
```

Gambar 3.24 Monitoring Perintah USER

Sebelumnya pada set mode telah ditentukan bahwa admin service=0 sehingga sub program pada Gambar 3.24 akan berjalan yaitu jika countbt kurang dari 2 maka motor dapat dinyalakan jika lebih dari 2 maka motor mati, countbt merupakan penanda bahwa *bluetooth* pada smartphone masih terhubung dengan sistem atau tidak. Lalu pada kondisi kedua sistem akan memperbarui tanggal terus hingga ketika lebih dari 24jam maka motor tidak dapat dinyalakan lagi dan sistem mengirim lokasi motor ke pemilik. Hal ini dapat dilakukan dikarenakan didalam program pada if yang pertama tanggal!=tanggal1(tidak sama dengan) yang berarti tanggal harus berbeda yaitu besok lalu pada if yang ketua yaitu jam=jam1 yang berarti besok pada jam yang sama sehingga dapat disimpulkan bahwa mode user akan aktif selama 1x24jam dan setelah habis akan kembali kedalam mode admin.

```
if( message.indexOf("DISSABLE") >= 0 ){
  if(adminservice==1){
  detachInterrupt(digitalPinToInterrupt(reedswitch));
  detachInterrupt(digitalPinToInterrupt(listrik));
  Serial.println("DISSABLE AKTIF");
  digitalWrite(4,LOW);
  digitalWrite(6, LOW);
  digitalWrite(7, HIGH);
  digitalWrite(8, LOW);
  digitalWrite(5, HIGH);
  delay(500);
  digitalWrite(5, LOW);
  koneksi=0;
  countbt=0;
  mode=0;}
  else{
  sim808.sendSMS(phone,"Anda masih berada dalam mode user,ganti ke mode admin terlebih dahulu");
  }
  }
```

Gambar 3.25 Perintah *Dissble*

Pada Gambar 3.25 merupakan program untuk melakukan dissable pada sistem. Pada perintah dissable merupakan perinah untuk mengem-balikan status sistem ke kondisi standby yang sehingga pada kondisi dissable ini variable yang merupakan integer akan dikembalikan nilainya seperti semula.

```
if( message.indexOf("LOKASI MOTOR") >= 0 ){
  sim808.sendSMS(phone,location);
  }
if( message.indexOf("MATIKAN MOTOR") >= 0 ){
  detachInterrupt(digitalPinToInterrupt(reedswitch));
  detachInterrupt(digitalPinToInterrupt(listrik));
  digitalWrite(4,HIGH);
  digitalWrite(5, HIGH);
  delay(500);
  digitalWrite(5, LOW);
  }
if( message.indexOf("HIDUPKAN ALARM") >= 0 ){
  detachInterrupt(digitalPinToInterrupt(reedswitch));
  detachInterrupt(digitalPinToInterrupt(listrik));
  digitalWrite(5, HIGH);
  delay(3000);
  digitalWrite(5, LOW);
  }
```

Gambar 3.26 Perintah Dukungan Sistem.

Pada Gambar 3.26 adalah program tambahan untuk mendukung sisem. Pada sistem pengaman terdapat perintah pendukung pengguna yaitu seperti mencari lokasi motor, lalu mematikan motor ketika tidak menginginkan untuk memasang mode tertentu dan yang terakhir menghidupkan sirine selama 3 detik yang berguna saat berada di parkiran dan lupa lokasi motor.

```
if( message.indexOf("PARKIR") >= 0 ){
  if(adminservice==1){
  attachInterrupt(digitalPinToInterrupt(reedswitch), sensor1 , RISING);
  attachInterrupt(digitalPinToInterrupt(listrik), sensor2, RISING);
  calling=0;
  call=0;
  count=0;
  sensorlistrik=0;
  digitalWrite(4,LOW);
  digitalWrite(5,LOW);
  digitalWrite(6, LOW);
  digitalWrite(7, LOW);
  digitalWrite(8, HIGH);
  mode=1;
  digitalWrite(5, HIGH);
  delay(500);
  digitalWrite(5, LOW);
  delay(50);
  digitalWrite(5, HIGH);
  delay(200);
  digitalWrite(5, LOW);
  }
  else{
  sim808.sendSMS(phone,"Anda masih berada dalam mode user,ganti ke mode admin terlebih dahulu");
  }
}
}
```

Gambar 3.27 Perintah Menggunakan Mode Parkir.

Pada Gambar 3.27 pada mode parkir, jika sudah pada mode admin maka perintah untuk mode parkir dapat dilakukan namun jika belum siap maka mengirimkan SMS ke pengguna untuk merubah ke mode admin. Untuk pelakukan proses ke mode parkir terlebih dahulu mengaktifkan *interupt* untuk sensor *reed swicth* dan sensor listrik yang dapat dilihat pada Gambar 3.21 lalu mereset semua integer yang berhubungan dengan mode parkir dan memberikan tanda bel ketika mode telah siap.

Pada Gambar 3.28 merupakan penggunaan *interupt* pada saat mode parkir. Pada Arduino Pro Mini pin *interupt* diletakkan pada pin 2 dan 3 yang dipakai untuk sensor *reed swicth* dan sensor listrik ketika terjadi respon dari kedua pin tersebut maka program didalam *interupt* akan berjalan langsung tanpa mempedulikan posisi pembacaan *sketch* Arduino yang berjalan. Pada sensor 1 *reed swicth* akan menghitung *count* bila kendaraan didorong yang mana reedswitch diletakan pada garpu motor dan magnet diletakan pada area putar motor seperti pada gambar 3.11 ketika dilewati magnet 1 kali maka akan merubah kontak reed switch sebanyak 2x pada pengambilan data bab 4 didapatkan untuk mendapatkan jarak 20 meter membutuhkan counter sebanyak 24. Ketika *count* lebih dari 24 maka kendaraan akan menghidupkan sistem keamanan, yaitu alarm,

mematikan mesin dan memblokir bensin dengan *valve*. sedangkan pada sensor2 ketika kendaraan retas dan dihidupkan sistem keamanan juga akan aktif.

```
void sensor1(){
  Serial.print("jumlah counter:");
  Serial.println(count);
  digitalWrite(LED_BUILTIN, HIGH);
  delay(500);
  digitalWrite(LED_BUILTIN, LOW);
  delay(500);
  count++;
  if(count>24){
  digitalWrite(4,HIGH);
  digitalWrite(5,HIGH);
  digitalWrite(6,HIGH);
  digitalWrite(7,HIGH);
  digitalWrite(8,HIGH);
  }
}
void sensor2(){
  sensorlistrik=1;
  digitalWrite(4,HIGH);
  digitalWrite(5,HIGH);
  digitalWrite(6,HIGH);
  digitalWrite(7,HIGH);
  digitalWrite(8,HIGH);
}
```

Gambar 3.28 Interupt Pada Mode Parkir.

Pada Gambar 3.29 merupakan mode *loop* program untuk memantau keadaan dari *count reed swicth* dan sensor listrik bila salah satu keadaan dari if terpenuhi maka sistem akan menelepon pengguna selama 25 detik 2x dan mengirimkan SMS bahwa motor dalam bahanya. Ketika panggilan ke2 selesai maka alarm pada motor akan mati namun tidak dengan *valve* dan CDI yang masih dalam keadaan aktif, yaitu *valve* yang memblokir aliran bensin dan CDI yang groundnya telah diputus oleh *relay*. Mode parkir juga dapat dilakukan melaui koneksi *bluetooth*. Proses yang dilakukan hampir sama hanya saja pada *bluetooth* yang dideteksi adalah data PARKIR1.

```
if(mode==1){
  Serial.println("modeparkir");
  if(count>24){
      detachInterrupt(digitalPinToInterrupt(reedswitch));
      detachInterrupt(digitalPinToInterrupt(listrik));
      call=1;
    }
  if(sensorlistrik==1){
     detachInterrupt(digitalPinToInterrupt(reedswitch));
     detachInterrupt(digitalPinToInterrupt(listrik));
     call=1;
    }
  if(call==1){
    if(calling<1){
     sim808.sendSMS(phone,"Motor anda dalam bahaya");
     }
    while(calling<2){
     sim808.callUp(phone);
     delay(25000);
     Serial.println("ATH");
     delay(800);
     Serial.println("ATH");
     delay(1000);
     calling++;
    }
    digitalWrite(5,LOW);
    }
      mode=1;
    }
```

Gambar 3.29 *Loop* Mode Parkir

```
if( btdata.indexOf("DRIVER1") >= 0 ){
  if(adminservice==1){
  detachInterrupt(digitalPinToInterrupt(reedswitch));
  detachInterrupt(digitalPinToInterrupt(listrik));
  countbt=0;
  Serial.println("DRIVER AKTIF");
  digitalWrite(4,LOW);
  digitalWrite(5,LOW);
  digitalWrite(6, HIGH);
  digitalWrite(7, LOW);
  digitalWrite(8, LOW);
  digitalWrite(5, HIGH);
  delay(200);
  digitalWrite(5, LOW);
  delay(50);
  digitalWrite(5, HIGH);
  delay(200);
  digitalWrite(5, LOW);
  mode=2;
  }
  else{
  sim808.sendSMS(phone,"Anda masih berada dalam mode user,ganti ke mode admin terlebih dahulu");
  }
  }
```

Gambar 3.30 Perintah Menggunakan Mode Driver.

Pada Gambar 3.30 merupakan perintah untuk menjalankan mode *driver*. Perintah hanya dapat dilakukan ketika berada dalam mode admin. Integer yang berkaitan dengan mode *driver* akan di kembalikan ke nilai 0 serta menonaktifkan *interupt* dan set pada *loop* mode untuk masuk ke mode=2.

```
if(mode==2){
  Serial.println("modedriver");
  if(countbt>2){
   digitalWrite(4,HIGH);
   digitalWrite(5,HIGH);
   digitalWrite(6,HIGH);
   digitalWrite(7,HIGH);
   digitalWrite(8,HIGH);

  }
  mode=2;
}
```

Gambar 3.31 *Loop* Pada Mode Driver.

Ketika sistem sudah masuk kedalam mode *driver* maka mode tersebut akan memberikan nilai mode=2 sehingga sistem keamanan menyala ketika count BT=2 yang mana koneksi *bluetooth* dengan sistem terputus.lebih jelasnya dapat dilihat pada Gambar 3.31.

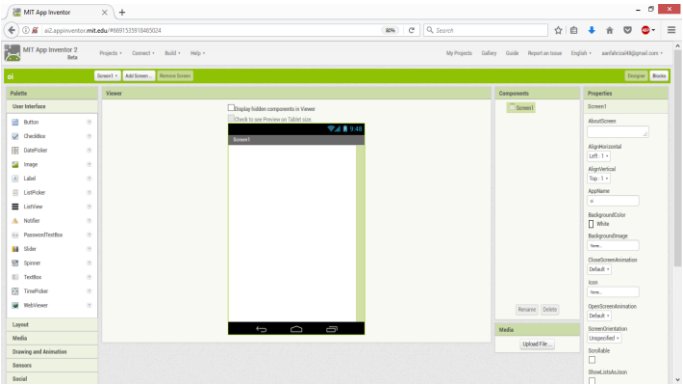
```
if(mode==2){
  if( btdata.indexOf("PING") >= 0 ){
    koneksi=1;
    countbt=0;
    }
  else{
    koneksi=0;
    countbt++;
  }
  }
if(adminservice==0){
  if( btdata.indexOf("PING") >= 0 ){
    koneksi=1;
    countbt=0;
    }
  else{
    koneksi=0;
    countbt++;
  }
}
}
```

Gambar 3.32 Monitoring Koneksi *Bluetooth***.**

Pada Gambar 3.32 menjelaskan tentang proses pertambahnya countbt. Pada mode driver dan user komunikasi *bluetooth* bada smartphone akan mengirimkan data PING ke sistem sehingga sistem akan mendeteksi hal tersebut dan countbt akan menjadi 0 namun ketika terputus maka count akan bertambah sehingga pada loop ke 3 tidak menerima data ping maka sistem keamanan akan aktif.

3.2.7 **Perancangan Aplikasi** Android

Aplikasi Android yang dibuat akan digunakan sebagai remote untuk mengontrol mode keamanan pada kendaraan. Perancangan *software* pada Android menggunakan aplikasi App Inventor yang dapat kita akses secara *online* dengan mengunjungi alamat *web* dan harus *log in* terlebih dahulu dengan menggunakan akun Gmail, bahasa pemrogramannya menggunakan pemrograman *visual block*. Setelah berhasil *log in* akan tampil halaman utama pada App Inventor buat desain aplikasi yang akan dibuat dengan *klik new project* kemudian akan muncul tampilan *design view* seperti Gambar 3.33.

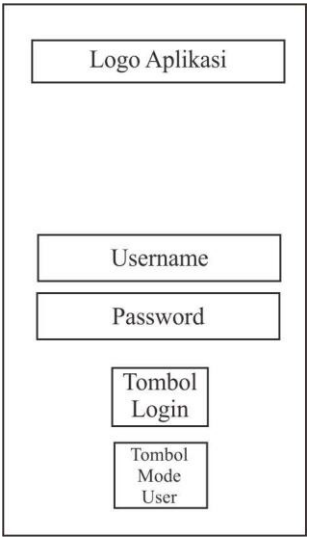


Gambar 3.33 Tampilan *Design View*

Pada *design view* terdapat komponen-komponen seperti tombol-tombol, *textbox*, label, gambar dan komponen tak terlihat seperti memasukan suara, sistem waktu dan lain-lain yang dapat membantu kita untuk merancang sebuah *user interface*. Untuk memasukan setiap komponen kita perlu men-*drag and drop* komponen-komponen tersebut ke dalam *viewer*. *Viewer* sendiri

merupakan sebuah tampilan *handphone* yang berfungsi sebagai tempat untuk merancang *user interface* yang kita buat.

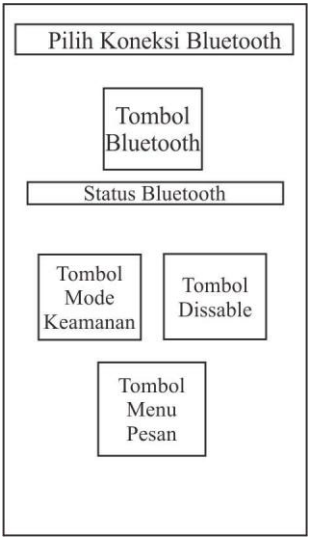
Tampilan pertama pada *software* ini adalah tampilan *log in* yang terdapat *username* dan *password* untuk memproteksi keamanan agar tidak sembarang orang bisa mengakses aplikasi ini. Rancangan tampilan *log in* dari aplikasi yang dibuat seperti Gambar 3.34.



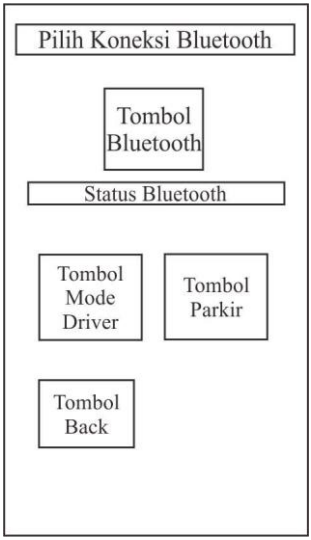
Gambar 3.34 Rancangan Tampilan *Log In* pada Aplikasi Android.

Setelah pengguna berhasil memasukan *username* dan *password* dengan benar akan tampil menu utama pada aplikasi Android seperti Gambar 3.35. Pada tampilan ini terdapat icon untuk memilih dan mengkoneksikan *bluetooth*, menampilkan menu untuk memilih mode keamanan, menu untuk mengirim pesan, men-*dissable* sistem keamanan, dan terdapat sistem untuk men-*dissable* sistem keamanan saat kapasitas baterai *handphone* kurang dari 10 persen pada saat mode *driver* diaktifkan dan akan mengirimkan data dari handphone Android ke mikrokontroler Arduino untuk mematikan sistem keamanan. Pada menu mode keamanan terdapat dua pilihan yaitu mode parkir dan mode *driver*, rancangan tampilan pada mode keamanan dapat dilihat pada Gambar 3.36. Pada menu untuk mengirim pesan pengguna dapat mengirim pesan, dengan tombol

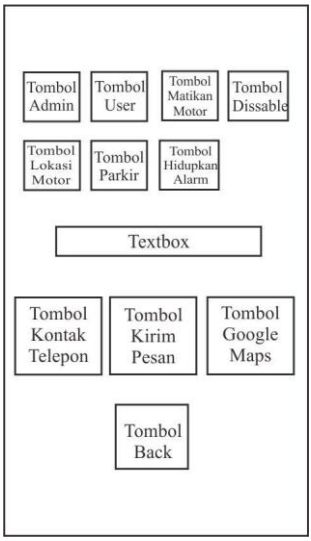
yang isi pesannya telah diatur sedemikian rupa agar pengguna tidak repot menuliskan pesan yang akan dikirimkan dan terdapat tombol untuk membuka aplikasi Google Maps, yang dapat dilihat pada Gambar 3.37. Saat sistem berada pada mode USER, maka pengguna dapat mengakses aplikasi pada menu USER seperti pada Gambar 3.38.



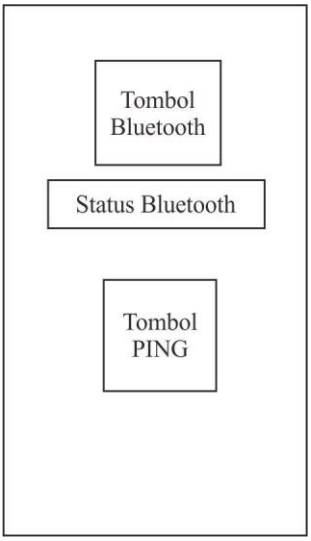
Gambar 3.35 Rancangan Tampilan Menu Utama



Gambar 3.36 Rancangan Tampilan Menu Mode Keamanan

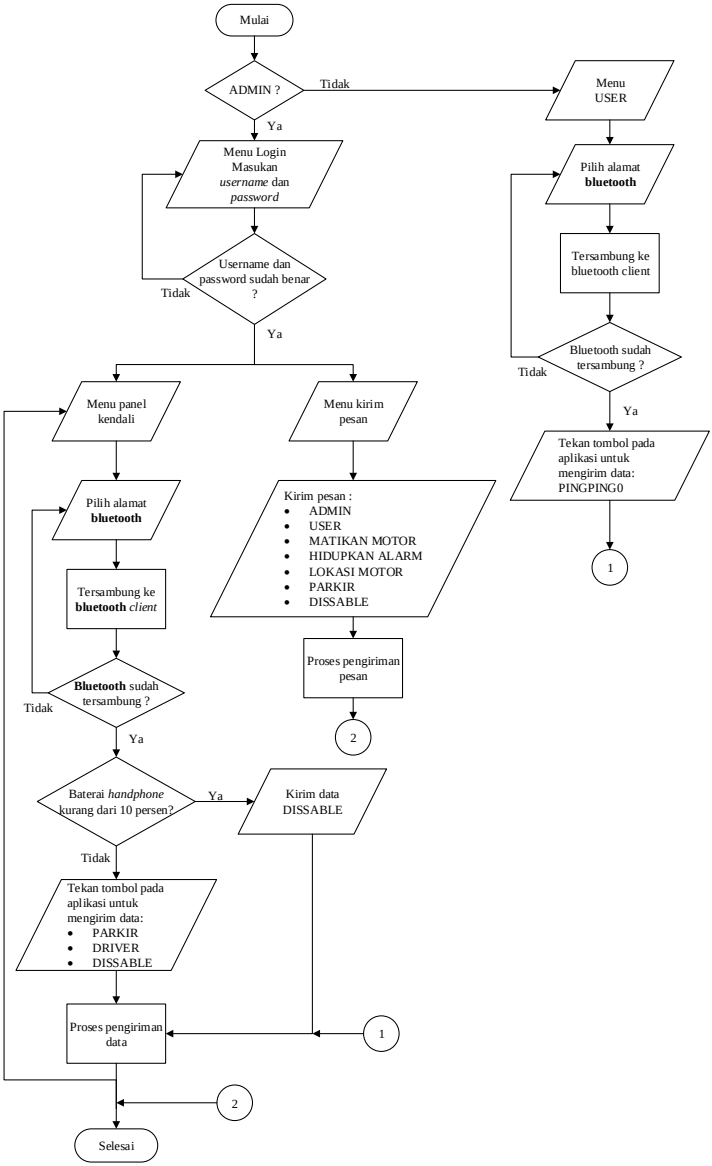


Gambar 3.37 Rancangan Tampilan Menu Kirim Pesan



Gambar 3.38 Rancangan Tampilan pada Mode USER

Dari perancangan interface aplikasi Android sistem kerja dari aplikasi Android dijelaskan pada flowchart seperti pada Gambar 3.39.

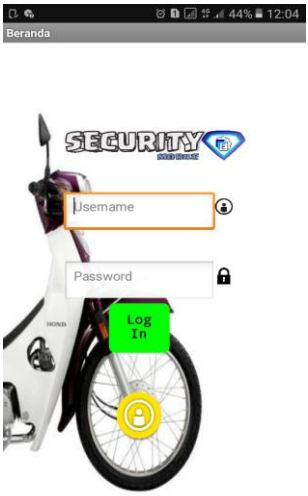


Gambar 3.39 *Flowchart* Aplikasi Android

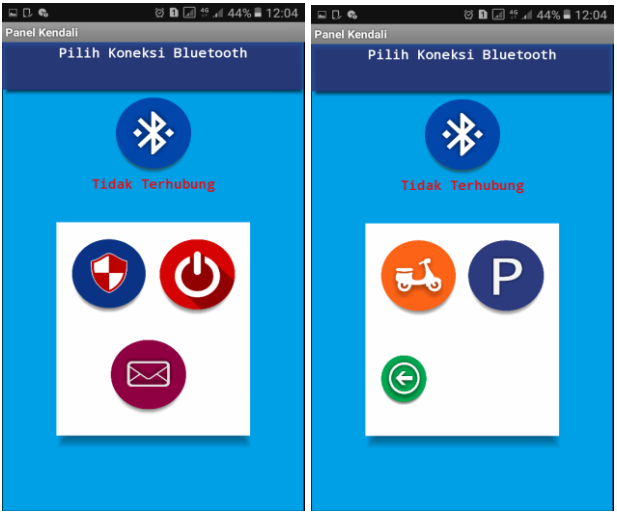
Saat membuka aplikasi terdapat 2 menu untuk mengakses aplikasi yaitu, menu untuk ADMIN dan menu untuk USER. Jika mengakses menu USER akan tampil layar untuk mengkoneksikan *bluetooth*, setelah *bluetooth* sudah tersambung pengguna menekan tombol untuk mengaktifkan keamanan saat berkendara. Jika mengakses menu ADMIN pengguna terlebih dahulu memasukan username dan password dengan benar, kemudian masuk ke menu panel kendali dan mengkoneksikan *bluetooth* terlebih dahulu untuk mengaktifkan mode keamanan, terdapat tombol parkir, *driver*, dan dissable. Tombol parkir akan mengaktifkan keamanan saat kendaraan di parkir, tombol driver akan mengaktifkan keamanan saat berkendara, dan tombol *dissable* untuk menon-aktifkan sistem keamanan.

Pada menu panel kendali terdapat juga menu untuk kirim pesan, yang dimana kita dapat mengakses sistem keamanan melalui SMS. Terdapat berbagai tombol untuk menuliskan pesan secara otomatis, pesan ADMIN untuk mengganti *password bluetooth* menjadi 1234, pesan USER untuk mengganti *password bluetooth* menjadi 5555, pesan HIDUPKAN ALARM untuk menghidupkan sirine selama 3 detik, pesan MATIKAN MOTOR untuk mematikan motor, pesan LOKASI MOTOR untuk memperoleh data koordinat kendaraan berada, pesan PARKIR untuk mengaktifkan mode parkir, dan pesan DISSABLE untuk menon-aktifkan sistem keamanan.

Setelah perancangan interface pada aplikasi Android, hasil dari perancangan tersebut dapat dilihat pada Gambar 3.40, Gambar 3.41, dan Gambar 3.42.

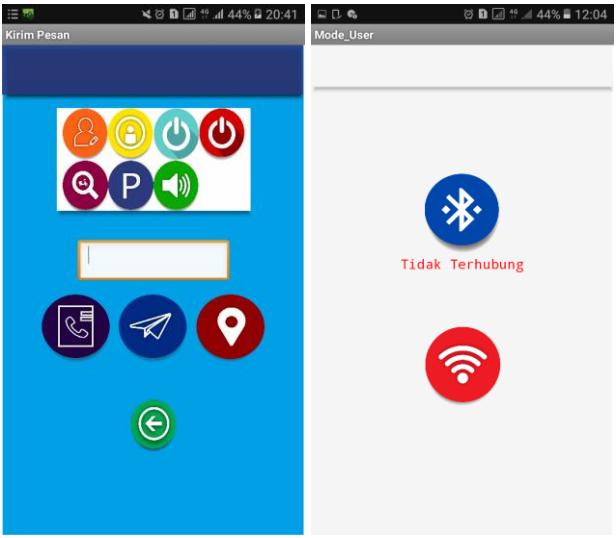


Gambar 3.40 Realisasi Tampilan *Log In* pada Aplikasi Android



(a) (b)

Gambar 3.41 (a) Realisasi Tampilan Menu Utama pada Aplikasi Android (b) Realisasi Tampilan Menu Mode Keamanan pada Aplikasi Android



(a) (b)

Gambar 3.42 (a) Realisasi Tampilan Menu Kirim Pesan (b) Realisasi Tampilan pada Mode User

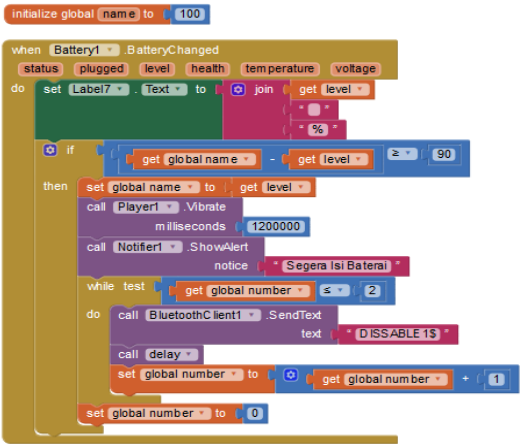
Dari realisasi tampilan aplikasi Android penjelasan tiap tombol dapat dilihat pada Tabel 3.1.

Tabel 3.1 Penjelasan *Icon* pada Aplikasi Android

Icon	Penjelasan
	Berfungsi untuk memilih dan mengkoneksikan *blue-tooth client.* .
	Berfungsi untuk menampilkan layar menu untuk mengirim pesan
	Berfungsi untuk menampilkan layar yang berisi mode keamanan
	Berfungsi untuk mematikan sistem keamanan
	Berfungsi untuk mengganti mode keamanan menjadi mode *driver.*
	Berfugsi untuk kembali ke menu utama

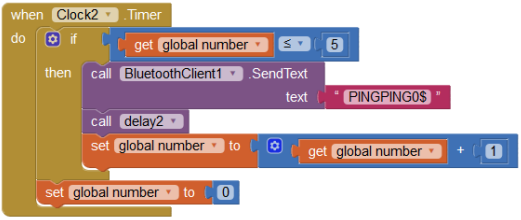
Icon	Penjelasan
	Berfungsi untuk mengganti mode keamanan menjadi mode parkir.
	Berfungsi untuk pergi ke menu utama setelah memasukan *username* dan *password* dengan benar.
Tidak Terhubung	Untuk mengetahui status *bluetooth* apakah sudah terkoneksikan dengan *bluetooth client*.
	Berfungsi untuk menampilkan teks “USER” pada *textbox*
	Berfungsi untuk menampilkan teks “ALARM” pada *textbox*
	Berfungsi untuk menampilkan teks “LOKASI MOTOR” pada textbox
	Berfungsi untuk menampilkan teks “ADMIN” pada *textbox*
	Berfungsi untuk menampilkan teks “MATIKAN MOTOR” pada *textbox*
	Berfungsi untuk memilih kontak telepon pada *handphone*
	Berfungsi untuk mengirimkan pesan sesuai dengan nomer yang dituju
	Berfungsi untuk mengirimkan data “PING” pada saat sistem keamanan pada mode USER.
	Berfungsi untuk membuka aplikasi Google Maps ketika telah menerima data GPS berupa Latitude dan Longtitude untuk mengetahui posisi kendaraan berada.

Setelah pembuatan *user interface* selesai, langkah selanjutnya adalah menyusun *code block* dan *block* komponen yang terdapat pada *block editor*, untuk masuk ke dalam *block editor* tekan *blocks* yang terdapat pada sisi kanan atas. Penyusunan *block-block* tersebut digunakan untuk mengatur jalannya program sesuai dengan yang diinginkan. Berikut ini sebagian penjelasan tentang *block-block* yang sudah disusun sesuai jalannya program.



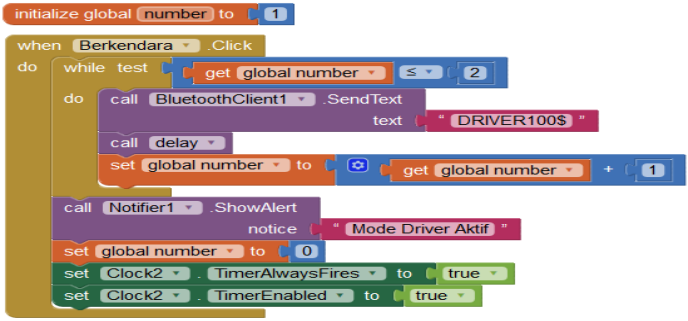
Gambar 3.43 *Block* Program Status Baterai

Pada Gambar 3.43 berfungsi untuk mengetahui status baterai *handphone*, dikarenakan saat berkendara dikhawatirkan koneksi *bluetooth* terputus akibat *handphone* kehabisan baterai, maka dari itu pada *block* program diatas akan mengirimkan data "DISSABLE1" saat kondisi baterai kurang dari 10 persen untuk mematikan sistem keamanan, agar tidak terjadi hal yang tidak diinginkan saat berkendara.



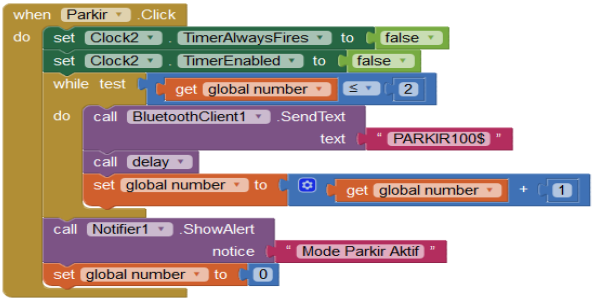
Gambar 3.44 *Block* Program Kirim Data Berulang

Pada Gambar 3.44 *block* program di atas berfungsi untuk mengirim data "PINGPING0" secara terus menerus ketika tombol mode *driver* ditekan. Program akan berhenti mengirim ketika tombol *dissable* ditekan, dan saat koneksi *bluetooth* terputus.



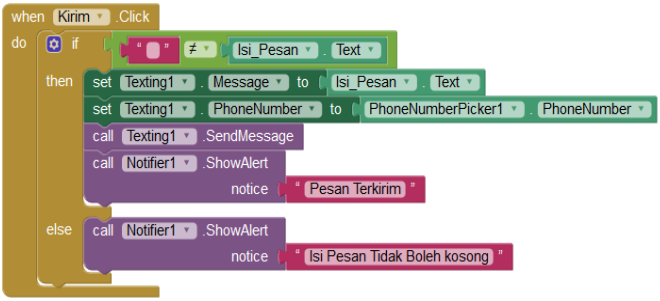
Gambar 3.45 *Block* Program Kirim Data *Driver*

Pada Gambar 3.45 merupakan *block* program untuk mengirim data "DRIVER100" yang digunakan untuk mengaktifkan mode *driver* pada sistem keamanan. Pada *block* tersebut terdapat perintah perulangan *while* yang berfungsi untuk mengirimkan data berulang selama dua detik, dikarenakan saat mengirim data sekali tidak terbaca oleh Arduino sehingga sistem tidak akan berfungsi. Setelah mengirim data selama detik program akan otomatis mengaktifkan Clock 2 untuk mengirim data secara terus menerus.



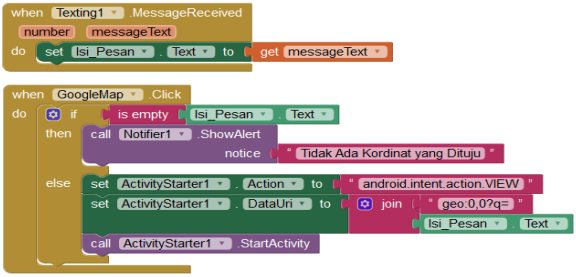
Gambar 3.46 *Block* Program Kirim Data Parkir

Pada Gambar 3.46 merupakan *block* program untuk mengirimkan data "PARKIR1000" yang digunakan untuk mengaktifkan sistem keamanan pada mode parkir. Pada *block* program ini juga terdapat perintah perulangan, sama halnya dengan *block* program sebelumnya yaitu untuk mengirim data berulang selama dua detik dan akan menon-aktifkan Clock 2.



Gambar 3.47 *Block* Program Kirim SMS

Pada Gambar 3.47 merupakan *block* program yang digunakan untuk mengirim SMS. Isi pesan SMS sudah diatur sedemikian rupa ,dengan menekan berbagai tombol dengan format teks yang sudah ditentukan sebelumnya. Isi pesan tidak boleh kosong, jika kosong maka akan muncul notifikasi "Isi Pesan Tidak Boleh Kosong".



Gambar 3.48 Block Program Starter Google Maps

Pada Gambar 3.48 merupakan *block* program yang berfungsi untuk membuka aplikasi Google Maps setelah pengguna telah menerima data GPS berupa koordinat posisi kendaraan, data tersebut dikirimkan melalui SMS dari sistem keamanan yang terpasang pada kendaraan, oleh karena itu terdapat *block* program untuk menerima SMS yang nantinya akan menampilkan isi pesan ke dalam *textbox*. Setelah menerima koordinat pengguna dapat menekan tombol yang tersedia untuk membuka aplikasi Google Maps dan menunjukkan posisi kendaraan berada.

BAB IV
PENGUJIAN DAN PENGUKURAN

Pada bab ini berisi tentang pengujian dan pengukuran dari alat yang dibuat untuk mengetahui apakah tujuan dari pembuatan alat telah sesuai dengan yang diharapkan atau tidak, maka diperlukan pengujian dan pengukuran dari alat yang sudah dibuat.

4.1 Pengukuran Rangkaian *Power Supply*

Pengukuran rangkaian *power supply* yang telah dibuat ini bertujuan untuk mengetahui berapa besar tegangan yang dikeluarkan oleh rangkaian tersebut berdasarkan tegangan *input* yang diperoleh dari kiprok kendaraan bermotor dan nantinya digunakan untuk mengisi daya pada *power bank.* Pengukuran rangkaian *power supply* ini dilakukan dengan cara mengukur tegangan *input* dari kiprok dan tegangan *output* dari *power supply* ketika tanpa beban dan saat ada beban dengan menggunakan AVO digital. Pada Gambar 4.1 merupakan contoh pengukuran rangakaian *power supply* yang sedang diukur tegangannya saat diberi beban *power bank*, sedangkan Tabel 4.1 dan Tabel 4.2 merupakan hasil dari pengukuran yang telah dilakukan.

Gambar 4.1 Pengukuran Rangkaian *Power Supply* saat Diberi Beban

Tabel 4.1 Hasil Pengukuran *Power Supply* Tanpa Beban

No.	Tegangan Input (VDC)	Tegangan Output (VDC)
1.	8,5	5,4
2.	12,1	5,4

No.	Tegangan Input (VDC)	Tegangan Output (VDC)
3.	16,1	5,4
4.	24,5	5,4

Tabel 4.2 Hasil Pengukuran *Power Supply* dengan Beban

No.	Tegangan Input (VDC)	Tegangan Output (VDC)
1.	8,5	5,34
2.	12,1	5,34
3.	16,1	5,34
4.	24,5	5,34

Dari hasil pengukuran *power supply* yang terdapat pada Tabel 4.1 dan Tabel 4.2 dapat diketahui bahwa tegangan *Power supply* saat tidak ada beban 5,4 VDC dan saat ada beban 5,3 VDC.

4.2 Pengukuran Jarak Koneksi *Bluetooth*

Pengukuran jarak koneksi *bluetooth* ini bertujuan untuk mengetahui berapa jarak jangkauan koneksi *bluetooth* dari *handphone* Android dengan modul SIM808, dari data yang diperoleh dapat diketahui pada jarak berapa koneksi *bluetooth* terputus, karena pada alat Tugas Akhir yang dibuat memerlukan status *bluetooth* saat koneksinya terputus untuk mengaktifkan sistem keamanan. Dalam proses pengukuran menggunakan dua *handphone* yang berbeda dengan tujuan untuk mengetahui berapa jauh jarak koneksi *bluetooth* antara *handphone* dengan modul SIM808, saat koneksi terputus dapat diketahui berapa jauh jarak koneksi *bluetooth* dari masing-masing *handphone*. Pada Tabel 4.3 merupakan hasil dari pengujian dan pengukuran jarak koneksi *bluetooth* yang telah dilakukan.

Tabel 4.3 Hasil Pengukuran Jarak Koneksi *Bluetooth*

No.	Tipe Handphone	Jarak Koneksi *Bluetooth*
1.	Samsung Galaxy J2 Prime	14 m
2.	LG Magna	24 m

Dari hasil pengukuran jarak koneksi *bluetooth* dapat diketahui bahwa setiap *handphone* memiliki jarak koneksi yang berbeda, pada pengukuran ini menggunakan 2 *handphone,* menggunakan

handphone Samsung Galaxy J2 Prime sejauh 14 m dan menggunkan *handphone* LG Magna sejauh 24 m, dari jarak tersebut sistem keamanan ketika berkendara akan aktif.

4.3 Pengujian Pengiriman Data dari Aplikasi Android ke SIM808

Pengujian ini bertujuan untuk mengetahui apakah aplikasi Android yang dibuat dapat mengirimkan data ke modul SIM808 dan sudah sesuai dengan yang diharapkan atau tidak. Pengujian dilakukan dengan menghubungkan SIM808 ke Arduino, dengan menuliskan *listing* program seperti dibawah ini :

```
void acceptBT(){
  delay(800);
  BTstatus = sim808.isBTstatus();
  delay(200);
  BTstatus = sim808.isBTstatus();
  Serial.println(BTstatus);
  delay(400);
  while(BTstatus==0){
  Serial.println("AT+BTSPPGET=1");
  delay (2000);
  Serial.println("AT+BTPOWER=1");
  delay (3000);
  BTstatus = sim808.isBTstatus();
  }
  delay (1000);
  while (BTstatus==25){
  Serial.println("AT+BTACPT=1");
  delay(3000);
  BTstatus = sim808.isBTstatus();
  }
}
```

Gambar 4.2 Void *Accept Bluetooth*

Setelah menuliskan *listing* program seperti contoh di atas, aplikasi Android dapat kita uji dengan menekan tombol yang digunakan untuk mengirim data. Pada Tabel 4.4 merupakan hasil pengujian aplikasi Android.

Tabel 4.4 Pengujian Pengiriman Data dari Aplikasi Android

No.	Tombol	Data	Jarak	Keterangan
1.	Parkir	"PARKIR100"	1 m	Terkirim
2.	Driver	"DRIVER100"	5 m	Terkirim
3.	Dissable	"DISSABLE1"	8 m	Terkirim

No.	Tombol	Data	Jarak	Keterangan
4.	Parkir	"PARKIR100"	12 m	Terkirim
5.	Driver	"DRIVER100"	14 m	Terkirim
6.	Dissable	"DISSABLE1"	15 m	Tidak Terkirim

Dari hasil pengujian pengiriman data dari aplikasi Android dapat diketahui bahwa data dapat terkirim pada jarak maksimal 14 m menggunakan komunikasi *bluetooth***.**

4.4 **Pengujian pengiriman** SMS **ke** SIM808

Pengujian ini bertujuan untuk mengetahui seberapa cepat respon SIM808 dalam menerima SMS. Pengujian dilakukan dengan cara mengirim SMS ke SIM808 dan mencatat waktu saat mengirim dan saat SIM808 menerima SMS. Pada Tabel 4.5 merupakan hasil pengujian dari pengiriman SMS ke SIM808

Tabel 4.5 Pengujian Pengiriman SMS ke SIM808

No.	Hari/Tanggal	Waktu Pengiriman	Waktu Penerimaan	Keterangan
1.	Rabu/07 Juni 2017	14:53:48	14:54:10	ADMIN
2.	Rabu/07 Juni 2017	14:55:52	14:56:09	USER
3.	Rabu/07 Juni 2017	14:56:30	14:56:48	LOKASI MOTOR

4.5 *AT Command*

AT Command merupakan cara komunikasi antara SIM808 dengan Arduino. Melalui *AT Command* dapat diperoleh data yang diperlukan melalui layanan yang disediakan oleh SIM808. Lama respon dibutuhkan untuk membuat suatu perangkat lunak. Dikarenakan dika pada saat masih *buffer* lalu diberikan *AT Command* lagi yang terjadi adalah SIM808 mengirimkan tulisan error. Untuk itu lama respon data diperlukan dalam membuat suatu perangkat lunak. Cara untuk mendapatkan lama respon daa tersebut dapat digunakan aplikasi SSCom yang mana *AT Command* dapat diketik secara manual dan dapat dihitung manual untuk lama responnya. Setelah lama respon didapat tinggal dimasukkan ke *delay*

untuk dapat menemukan data yang tepat data tersebut dapat dilihat pada Tabel 4.6.

Tabel 4.6 Pengukuran Kecepatan Respon SIM808

Comand	Lama Respon (Second)	Respon
AT	0,5	OK
AT+BTPWR=0	2	OK
AT+BTPWR=1	2,5	OK
AT+BTSPPGET=1	1	+AT+BTSPPGET=1 OK
AT+CGNSPWR=1	3	OK
AT+BTSTATUS?	1	+BTSTATUS: <Status>
AT+BTACPT=1	1	OK
AT+BTSPPGET=3,5	0,5	+BTSPPGET:<data>

Dari hasil kecepatan respon yang diperoleh maka data kecepatan respon ini akan dipakai untuk merancang sketsa program yang mana apabila SIM808 sedang memproses data lalu perintah baru masuk maka sim 808 akan mengirimkan data yang masih sedang diproses yang secara langsung akan mempengaruhi proses bekerjanya sistem pengaman sehingga *delay* yang dimasukkan harus melebihi lama respon setidaknya 0,2 detik lebih lama dari lama respon sehingga Arduino dapat membaca informasi data dari SIM808 dengan benar.

4.6 Data Tegangan *Selenoid Valve* dan Sirine

Data elektrik merupakan data yang berhubungan dengan tegangan, arus maupun hambatan. Adapun yang diukur disini yaitu tengangan *valve* dan sirine. Pada tegangan *valve* yang perlu dilakukan adalah menghubungkan kabel power *valve* dengan tegangan untuk mengeceknya cukup diubah tegangan perlahan lahan namun untuk kabel power *valve* di hubungkan sesaat setiap ganti tegangan. Hingga ketemu tegangan yang cukup untuk mengaktifkan *valve*. Hal ini juga sama dilakukan untuk mengambil data dari sirine Tabel 4.7 dan 4.8 merupakan data *valve* dan sirine yang telah diperoleh.

Tabel 4.7 Pengukuran Tegangan *Valve*

No.	Tegangan (volt) dc	Aktif
1	2	0
2	4	0
3	8	0
4	11	0
5	11,91	1

Dari data *valv*e yang telah didapat pada karakteristik valve tersebut mempunyai tegangan kerja 12 volt dan pada saat dilakukan pengujian *valve* tersebut dapat bekerja pada tegangan 11,91 volt, sehingga dengan data tersebut didapatkan tegangan kerja dari *valve* tersebut antara 11,91-12 volt dikarenakan 12 merupakan tegangan pada karakteristik *valve* sehingga tegangan 12 volt merupakan tegangan yang direkomendasikan.

Tabel 4.8 Pengukuran Tegangan Sirine

No.	Tegangan (volt) dc	Aktif
1	1	0
2	2	0
3	3	0
4	4	0
5	5	1

Dari karakteristik sirine, sirine dapat bekerja dengan tegangan kerja sebesar 12v lalu saat dilakukan pengujian sirine dapat bekerja pada tegangan 5 volt namun suara yang dihasilkan tidak sekeras 12 volt sehingga sirine dapat bekerja dalam range 5-12 volt namun pada range sejauh itu ditakutkan akan dapat cepat merusak sirine dikarenakan tidak mendapat tegangan supply cukup sehingga direkomendasikan menggunakan tegangan 12 volt.

4.7 Data *Reed Switch*

Untuk mengambil data *reed switch* yang perlu dilakukan adalah dengan mengetahui jari-jari velg kendaraan. Setelah dilakukan pengukuran velg kendaraan diketahui bahwa jari-jarinya sebesar 27cm dan dengan menggunakan rumus $2.\pi.R$ didapatkan keliling kendaraan sebesar 169,6cm. Target yang diinginkan pada sistem keamanan kendaraan yaitu jika jarak melebihi 20m maka

sistem pengaman kendaraan akan aktif. Pada sensor *reed switch* yang dihitung adalah jumlah putaran pada kendaraan. Sehingga didapatkan 20m=2000cm, 2000:169,6cm= 11,79 kali putaran atau jika dibulatkan menjadi 12 kali putaran. Pada *reed switch* ketika dilewati medan magnet satu kali maka *reed switch* tersebut akan merubah posisi kontaknya 2 kali dikarenakan lempengan kontak *reed switch* ditarik oleh medan magnet pada kutub utara dan kutup selatan. Lalu aktifnya *reed switch* tersebut akan dijadikan satuan *counter* pada Arduino sehingga untuk mencapai jarak 20 m diperlukan 12 x 2 = 24 *counter*. Sehingga ketika dilakukan hasil pengukuran didapatkan hasil pada Tabel 4.9.

Tabel 4.9 Pengukuran Jarak *Reed Switch*

No	Jumlah *counter*	Jarak
1	24	20 m
2	24	19,8 m

Dari data *reed switch* yang telah didapatkan ini yaitu pada jumlah 24 *counter* mendapatkan hasil jarak 20 dan 19,8 m dengan 2 kali pengukuran sehingga hasil tersebut telah mendekati jarak 20 m yaitu sesuai dengan target yang diinginkan.

4.8 **Data** GPS

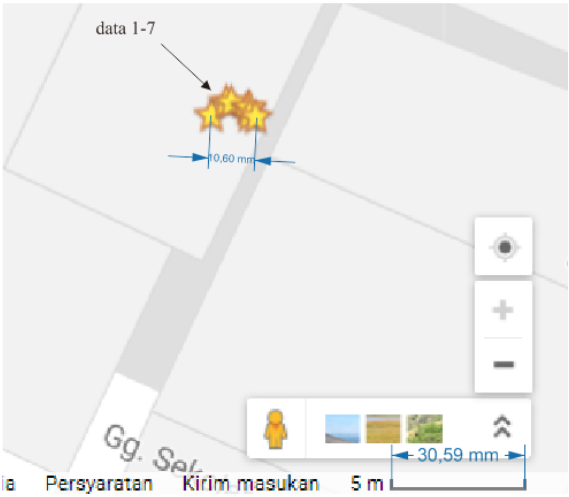
Untuk mengukur data GPS dapat menggunakan aplikasi SSCOM dan memanggil info secara manual yaitu dengan cara hubungkan SIM808 dengan PC buka aplikasi SSCOM pada PC. Lalu seting tempat port berada dan ketik AT+CGPSPWR=1 untuk menyaakan GPS, lalu ketik AT+CGPSRST=0, untuk masuk ke mode *cold*, dan AT+CGPSINF =0 untuk mengambil data, data pada koma pertama adalah mode, lalu kedua adalah *latitude* dan yang ketiga *longitude*. Data GPS dapat dilihat pada Tabel 4.10.

Tabel 4.10 Pengujian GPS

Data ke	*Latitude*	*Longitude*
1	-7.226922	112.757585
2	-7.226918	112.757590
3	-7.226918	112.757592
4	-7.226920	112.757597
5	-7.226920	112.757598

Data ke	*Latitude*	*Longitude*
6	-7.226922	112.757598
7	-7.226923	112.757600

Dari data yang telah didapat untuk mengetahui dimana letak lokasi tersebut dapat menggunakan aplikasi peta google map dengan menginputkan data <latitude>,<longitude> contohnya seperti -7.226922, 112.757585 dan setelah itu pada aplikasi googlemap tersebut dapat menampilkan lokasi dari hasil data tersebut. Hasil dari ke 7 data GPS yang diambil pada tempat yang sama dapat dilihat pada gambar 4.3 dan pada Gambar 4.3 merupakan hasil range area dari ke 7 titik poin tersebut.



Gambar 4.3 Pengujian Data GPS

Dari data 1 sampai 7 yang telah di temukan lokasinya melalui google maps lalu di ambillah 2 lokasi yang jaraknya paling jauh untuk menengetahui apakah GPS masih dalam keadaan baik atau tidak. Pada skala baru yang telah dibuat didapatkan hasil jarak terjauh melalui aplikasi CorelDRAW yang dapat mengukur panjang dari gambar. Didapatkan skala jarak 30,59 mm = 5m dengan jarak terjauh data yaitu 10,50mm sehingga kita dapat menemukan jarak lokasi yang paling jauh yaitu dengan melalui perbandingan

(30,59mm/5000mm)=(10,60mm/x) didapatkan x= 1732,59mm atau 1,7 meter. Range GPS pada karakteristik modul SIM808 adalah 5meter. Dari hasil tersebut dapat di simpulkan bahwa GPS masih berada didalam range aman.

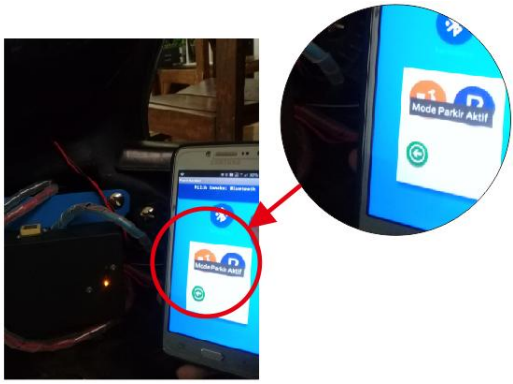
Data pada GPS tidak dapat diambil ketika posisi alat tersebut pada keadaan tertutup yang mana terdapat material tebal diatasnya contohnya didalam gedung bangunan bertingkat yang mengakibatkan sinyal GPS tidak dapat ditangkap oleh SIM808.

4.9 Pengujian Sistem Keseluruhan

Pengujian sistem keseluruhan ini bertujuan untuk mengetahui seberapa besar keberhasilan dari alat yang sudah dibuat. Pada Gambar 4.4 merupakan gambar kondisi sepeda motor sebelum sistem keamanan diaktifkan dan sepeda motor dalam keadaan tidak terkunci..

Gambar 4.4 Kondisi Awal Sepeda Motor

Pengujian pertama sistem keamanan diaktifkan pada mode parkir, dengan menekan tombol parkir pada aplikasi Android. Dari aplikasi Android akan mengirimkan data PARKIR10 melalui *bluetooth*, sehingga mode parkir aktif dan mengaktifkan *interupt* pada sensor dan menyalakan lampu indikator berwarna kuning yang dapat dilihat pada Gambar 4.5.



Gambar 4.5 Tampilan Mode Parkir Aktif

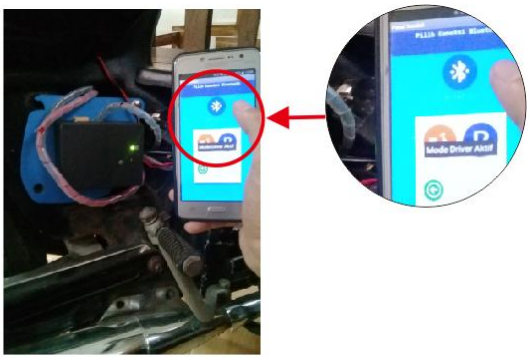
Pada mode parkir sistem keamanan akan aktif saat sepeda motor dihidupkan secara paksa oleh pencuri tanpa sepengetahuan pengguna yang dapat dilihat pada Gambar 4.6 dan saat sepeda motor dicuri dengan cara didorong seperti Gambar 4.7. Seketika itu sirine akan aktif, valve akan memblokir aliran bensin, dan aliran listrik dari CDI akan diputus oleh driver relay. Untuk memberitahu pengguna bahwa sepeda motor dalam bahaya, sistem keamanan akan telepon *handphone* pengguna yang dapat dilihat seperti Gambar 4.8.

Gambar 4.6 Sepeda Motor Dihidupkan Secara Paksa

Gambar 4.7 Sepeda Motor Dicuri dengan Cara Didorong

Gambar 4.8 Sistem Keamanan Telepon Pengguna

Pengujian kedua sistem keamanan diaktifkan pada mode driver, pengguna menekan tombol driver pada aplikasi Android, dan akan mengirimkan data DRIVER untuk mengaktifkan mode driver, dengan indikasi lampu indikator hijau menyala dan akan mengaktifkan sistem pengaman saat berkendara, indikasi lampu indikator hijau menyala dapat dilihat seperti pada Gambar 4.9. Saat berkendara aplikasi Android mengirimkan data "PING" secara terus menerus, jika koneksi *bluetooth* tersebut terputus yang mengakibatkan sistem tidak menerima data "PING" maka sistem keamanan akan aktif. Pengujian mode *driver* dapat dilihat seperti pada Gambar 4.10, pengujian dilakukan dengan cara menjauhkan sepeda motor dengan *handphone* pengguna.

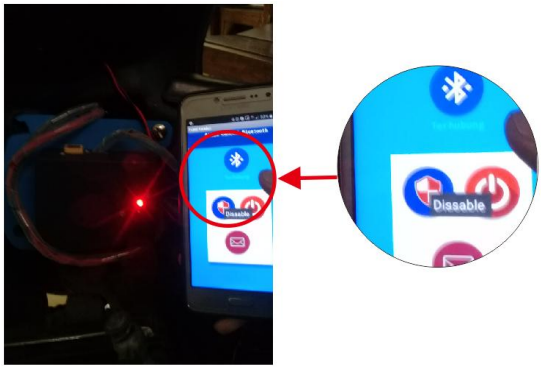


Gambar 4.9 Tampilan Mode *Driver* Aktif

Gambar 4.10 Pengujian Mode *Driver*

Pada mode ini sistem keamanan akan aktif dikarenakan koneksi *bluetooth* terputus dan tidak bisa mengirimkan data "PING". Seketika itu motor akan mati dikarenakan aliran listrik dari CDI terputus, *valve* akan aktif untuk memblokir aliran bensin, dan sirine akan berbunyi sebagai tanda peringatan.

Untuk menon-aktifkan sistem keamanan pengguna dapat menekan tombol dissable pada aplikasi Android, dan mengirimkan data DISSABLE dengan indikasi lampu merah indikator menyala, yang dapat dilihat seperti Gambar 4.11.



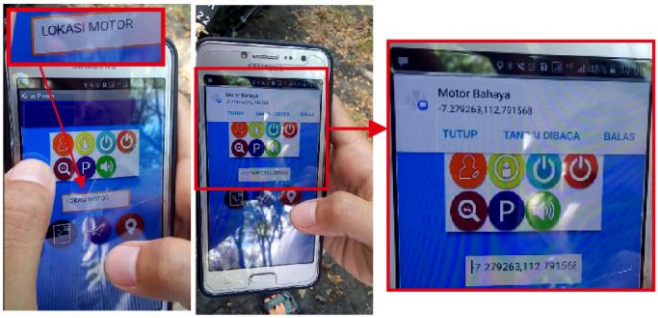
Gambar 4.11 Tampilan *Dissable* Sistem

Tabel 4.11 Pengujian Lokasi Kendaraan.

Pengujian Ke-	Lattitude	Longtitude	Lokasi
1	0.00000	0.00000	Tidak ditemukan
2	-7.226922	112.757585	Tenggumung Baru Mulya 1
3	-7.226918	112.757590	Tenggumung Baru Mulya 1
4	-7.279263	112.791568	Jl. Taman Alumni ITS
5	-7.278885	112.792260	Departemen Teknik Mesin Industri

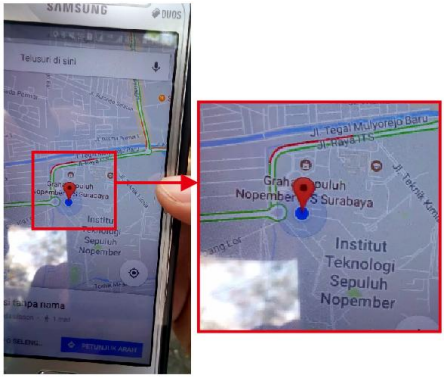
Pada Tabel 4.11 merupakan data GPS yang didapatkan dari beberapa lokasi, pada pengujian ke 1 data GPS 0.00000, 0.00000 yang berarti GPS belum dapat menemukan koordinat kendaraan berada dikarenakan tidak dapat menerima data dari satelit saat berada di dalam gedung.

Pengujian lokasi kendaraan dilakukan dengan cara mengakses GPS untuk menemukan lokasi kendaraan, pengguna terlebih dahulu mengirimkan pesan lewat aplikasi Android dengan menekan tombol LOKASI MOTOR dan mengirimkan pesan ke sistem keamanan seperti pada Gambar 4.12. Setelah itu menunggu balasan pesan yang berisikan koordinat lokasi kendaraan berada, kemudian menekan tombol Google Maps yang akan menunjukkan lokasi kendaraan berada, tampilan dari Google Maps dapat dilihat seperti Gambar 4.13.



(a) (b)

Gambar 4.12 (a) Kirim Pesan Lokasi Motor (b) Pesan Masuk Koordinat Lokasi Kendaraan



Gambar 4.13 Tampilan pada Google Maps

BAB V
PENUTUP

5.1 Kesimpulan

Dari hasil pengujian dan pengukuran alat Tugas Akhir yang dibuat dapat ditarik kesimpulan sebagai berikut :

1. Tegangan pada *power supply* harus lebih besar dari tegangan pada *power bank*, setelah dibuat *rangkaian power supply* dapat diketahui bahwa tegangan output saat tidak diberi beban sebesar 5,4VDC dan saat diberi beban sebesar 5,3VDC
2. Setiap *handphone* yang berbeda mempunyai jarak koneksi *bluetooth* yang berbeda pula sehingga pengaman kendaraan pada saat mode *driver* akan berfungsi pada jarak yang berbeda-beda tergantung dari *handphone* yang digunakan.
3. Pengiriman data dari aplikasi Android ke sistem pengaman pada kendaraan dapat diterima pada jarak maksimal 14,6m.
4. Kecepatan pengiriman SMS tergantung dari beberapa faktor yaitu pada provider sendiri dan kuatnya sinyal pada saat pengiriman.

5.2 Saran

Saran untuk alat Pengaman Kendaraan Bermotor dengan Remote Android Berbasis GSM, GPS, dan *Bluetooth* adalah :

1. Pada aplikasi Android perlu ditambahkan fitur *running in the background* agar aplikasi dapat bekerja di belakang layar.
2. Sebaiknya ditambahkan *feedback* pada aplikasi agar pengguna dapat mengetahui keadaan terbaru kendaraannya.

-----Halaman ini sengaja dikosongkan-----

DAFTAR PUSTAKA

[1] Arduino, 2016, Arduino-Introduction. https://www.Arduino.cc/en/Main/ArduinoBoardUno. (diakses tanggal 7 Maret 2017).

[2] Ecadio, 2017, Belajar dan Mengenal Arduino Pro Mini. http://ecadio.com/belajar-dan-mengenal-Arduino-pro-mini (diakses tanggal 21 Maret 2017).

[3] Arrland, Perkembangan Sistem Operasi Android, http://www.meroket.com/perkembangan-sistem-operasi-Android/ (diakses tanggal 21 Maret 2017).

[4] Wikipedia, 2017, Android (Sistem Operasi), https://id.wikipedia.org/wiki/Android_(sistem_operasi) (diakses tanggal 21 Maret 2017).

[5] SIMCOM, 2015, SIM808. http://simcom.ee/modules/GSM-gprs-gnss/SIM808/. (diakses tanggal 22 Maret 2017).

[6] Github, 2016, DFRobot_SIM808. https://github.com/DFRobot/DFRobot_SIM808. (diakses tanggal 22 Maret 2017).

[7] Elektro Indonesia, 2001, *BLUETOOTH*: Teknologi Komunikasi Wireless untuk Layanan Multimedia dengan Jangkauan-Terbatas, http://www.elektroindonesia.com/elektro/khu36.html (diakses tanggal 23 Maret 2017).

[8] Wikipedia, 2015, Sistem Pemosisi Global. https://id.wikipedia.org/wiki/Sistem_Pemosisi_Global#cite_note-GPS_overview_from_JPO-3. (diakses tanggal 23 Maret 2017).

[9] Putu Nopa Gunawan, 2011, Laporan Praktikum Rangkaian Listrik dan Rangkaian Logika (Power Supply), https://www.scribd.com/document/105790297/Power-Supply (diakses tanggal 24 Maret 2017).

[10] Wikipedia, 2010, App Inventor, https://id.wikipedia.org/wiki/App_Inventor, (diakses tanggal 4 April 2017).
[11] Ahmad Fajar Prasetyo, 2016, App Inventor untuk Pemula, http://www.academia.edu/10362093/Surya_Univ-Appinventor-bagi-pemula-by-Ahmad-Fajar-Prasetiyo, (diakses 10 April 2017).

LAMPIRAN A

A.1. Perancangan Perangkat Lunak

- Listing Program

```
#include <DFRobot_SIM808.h>
#include <SoftwareSerial.h>

#define MESSAGE_LENGTH 160
#define BT_LENGTH 160
char message[MESSAGE_LENGTH];
int messageIndex = 0,BTstatus=0,adminservice=1,count=0;
int sensorlistrik=0,mode=0,countbt=0,koneksi=0,call=0,calling=0;
char btdata[BT_LENGTH];
char phone[16];
char datetime[24];
const int reedswitch=3;
const int listrik=2;
char tanggal[3];
char tanggal1[3];
char jam[3];
char jam1[3];
char menit[3];
char menit1[3];
char location[30];
DFRobot_SIM808 SIM808(&Serial);

void setup() {
  Serial.begin(9600);
  pinMode(reedswitch, INPUT);
  pinMode(listrik, INPUT);
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(7, OUTPUT);
  pinMode(8, OUTPUT);
  pinMode(LED_BUILTIN, OUTPUT);
  detachInterrupt(digitalPinToInterrupt(reedswitch));
  detachInterrupt(digitalPinToInterrupt(listrik));
```

```
  while(!SIM808.init()) {
     Serial.print("SIM808 init error\r\n");
     delay(1000);
  }
  delay(3000);
  Serial.println("Init Success, module ready!");
  Serial.println("AT+CGPSPWR=1");
  delay (2000);
  Serial.println("AT+CGPSRST=0");
  delay (3000);
     }
void sensor1(){
  Serial.print("jumlah counter:");
  Serial.println(count);
  digitalWrite(LED_BUILTIN, HIGH);
  delay(500);
  digitalWrite(LED_BUILTIN, LOW);
  delay(500);
  count++;
  if(count>24){
  digitalWrite(4,HIGH);
  digitalWrite(5,HIGH);
  digitalWrite(6,HIGH);
  digitalWrite(7,HIGH);
  digitalWrite(8,HIGH);
  }
}
void sensor2(){
  sensorlistrik=1;
  digitalWrite(4,HIGH);
  digitalWrite(5,HIGH);
  digitalWrite(6,HIGH);
  digitalWrite(7,HIGH);
  digitalWrite(8,HIGH);
}
void loop() {
delay(200);
acceptBT();
delay(200);
```

```
SIM808.isBTdata(btdata);
Serial.println(btdata);
BTproses(btdata);
while(koneksi==1){
  SIM808.isBTdata(btdata);
  Serial.println(btdata);
  delay(300);
  BTproses(btdata);
  delay(1000);
  }
bacapesan();
delay(300);
SIM808.getGPSlokasi(location);
Serial.println(location);
delay(300);
loopmode();
}
void bacapesan(){
  messageIndex = SIM808.isSMSunread();
    Serial.print("messageIndex: ");
    Serial.println(messageIndex);
  if (messageIndex > 0) {
 SIM808.readSMS(messageIndex, message, MES-
SAGE_LENGTH, phone, datetime);
      delay(3000);
      SIM808.deleteSMS(messageIndex);
      Serial.print("From number: ");
      Serial.println(phone);
      Serial.print("Datetime: ");
      Serial.println(datetime);
      Serial.print("Recieved Message: ");
      Serial.println(message);
      delay(5000);
      prosespesan(message);
}
}


void prosespesan(String message){
```

```
if( message.indexOf("ADMIN") >= 0 ){
  detachInterrupt(digitalPinToInterrupt(reedswitch));
  detachInterrupt(digitalPinToInterrupt(listrik));
  Serial.println("AT+BTPOWER=0");
  delay (2000);
  Serial.println("AT+BTPOWER=1");
  delay (2000);
  Serial.println("AT+BTPAIRCFG=1,1234");
  delay (2000);
  Serial.println("AT+BTPAIRCFG=1,1234");
  delay (2000);
  Serial.println("AT+BTPOWER=0");
  delay (3000);
  adminservice=1;
  Serial.println("MODE ADMIN");
  digitalWrite(5, HIGH);
  delay(500);
  digitalWrite(5, LOW);
  delay(50);
  digitalWrite(5, HIGH);
  delay(500);
  digitalWrite(5, LOW);
  }
if( message.indexOf("USER") >= 0 ){
  detachInterrupt(digitalPinToInterrupt(reedswitch));
  detachInterrupt(digitalPinToInterrupt(listrik));
  Serial.println("AT+BTPOWER=0");
  delay (2000);
  Serial.println("AT+BTPOWER=1");
  delay (2000);
  Serial.println("AT+BTPAIRCFG=1,5555");
  delay (2000);
  Serial.println("AT+BTPAIRCFG=1,5555");
  delay (2000);
  Serial.println("AT+BTPOWER=0");
  delay (3000);
  digitalWrite(6, HIGH);
  digitalWrite(8, HIGH);
  adminservice=0;
```

```
  SIM808.getDateTime(tanggal, jam, menit);
  memcpy(tanggal1, tanggal, 3);
  memcpy(jam1, jam, 3);
  memcpy(menit1, menit, 3);
  Serial.println("MODE USER");
  mode=3;
  digitalWrite(5, HIGH);
  delay(300);
  digitalWrite(5, LOW);
  delay(100);
  digitalWrite(5, HIGH);
  delay(300);
  digitalWrite(5, LOW);
  }
 if( message.indexOf("LOKASI MOTOR") >= 0 ){
  SIM808.sendSMS(phone,location);
  }
 if( message.indexOf("MATIKAN MOTOR") >= 0 ){
  detachInterrupt(digitalPinToInterrupt(reedswitch));
  detachInterrupt(digitalPinToInterrupt(listrik));
  digitalWrite(4,HIGH);
  digitalWrite(5, HIGH);
  delay(500);
  digitalWrite(5, LOW);
  }
 if( message.indexOf("HIDUPKAN ALARM") >= 0 ){
  detachInterrupt(digitalPinToInterrupt(reedswitch));
  detachInterrupt(digitalPinToInterrupt(listrik));
  digitalWrite(5, HIGH);
  delay(3000);
  digitalWrite(5, LOW);
  }
 if( message.indexOf("DISSABLE") >= 0 ){
  if(adminservice==1){
  detachInterrupt(digitalPinToInterrupt(reedswitch));
  detachInterrupt(digitalPinToInterrupt(listrik));
  Serial.println("DISSABLE AKTIF");
  digitalWrite(4,LOW);
  digitalWrite(6, LOW);
```

```
  digitalWrite(7, HIGH);
  digitalWrite(8, LOW);
  digitalWrite(5, HIGH);
  delay(500);
  digitalWrite(5, LOW);
  koneksi=0;
  countbt=0;
  mode=0;}
  else{
  SIM808.sendSMS(phone,"Anda masih berada dalam mode user,ganti ke mode admin terlebih dahulu");
  }
  }
if( message.indexOf("PARKIR") >= 0 ){
  if(adminservice==1){
  attachInterrupt(digitalPinToInterrupt(reedswitch), sensor1 , RISING);
  attachInterrupt(digitalPinToInterrupt(listrik), sensor2, RISING);
  calling=0;
  call=0;
  count=0;
  sensorlistrik=0;
  digitalWrite(4,LOW);
  digitalWrite(5,LOW);
  digitalWrite(6, LOW);
  digitalWrite(7, LOW);
  digitalWrite(8, HIGH);
  mode=1;
  digitalWrite(5, HIGH);
  delay(500);
  digitalWrite(5, LOW);
  delay(50);
  digitalWrite(5, HIGH);
  delay(200);
  digitalWrite(5, LOW);
  }
  else{
```

```
  SIM808.sendSMS(phone,"Anda masih berada dalam mode
user,ganti ke mode admin terlebih dahulu");
  }
}
}


void acceptBT(){
  delay(800);
  BTstatus = SIM808.isBTstatus();
  delay(200);
  BTstatus = SIM808.isBTstatus();
  Serial.println(BTstatus);
  delay(400);
  while(BTstatus==0){
  Serial.println("AT+BTSPPGET=1");
  delay (2000);
  Serial.println("AT+BTPOWER=1");
  delay (3000);
  BTstatus = SIM808.isBTstatus();
  }
  delay (1000);
  while (BTstatus==25){
  Serial.println("AT+BTACPT=1");
  delay(3000);
  BTstatus = SIM808.isBTstatus();
  }
}

void BTproses(String btdata){
if( btdata.indexOf("DISSABLE1") >= 0 ){
  if(adminservice==1){
  detachInterrupt(digitalPinToInterrupt(reedswitch));
  detachInterrupt(digitalPinToInterrupt(listrik));
  Serial.println("DISSABLE AKTIF");
  digitalWrite(4,LOW);
  digitalWrite(6, LOW);
  digitalWrite(7, HIGH);
  digitalWrite(8, LOW);
```

```
  digitalWrite(5, HIGH);
  delay(500);
  digitalWrite(5, LOW);
  koneksi=0;
  countbt=0;
  mode=0;
  }
  else{
  SIM808.sendSMS(phone,"Anda masih berada dalam mode user,ganti ke mode admin terlebih dahulu");
  }
  }
if( btdata.indexOf("PARKIR1") >= 0 ){
  if(adminservice==1){
  attachInterrupt(digitalPinToInterrupt(reedswitch), sensor1 , RISING);
  attachInterrupt(digitalPinToInterrupt(listrik), sensor2, RISING);
  count=0;
  sensorlistrik=0;
  calling=0;
  call=0;
  Serial.println("PARKIR AKTIF");
  digitalWrite(4,LOW);
  digitalWrite(6, LOW);
  digitalWrite(7, LOW);
  digitalWrite(8, HIGH);
  digitalWrite(5, HIGH);
  delay(500);
  digitalWrite(5, LOW);
  delay(50);
  digitalWrite(5, HIGH);
  delay(200);
  digitalWrite(5, LOW);
  mode=1;
  }
  else{
  SIM808.sendSMS(phone,"Anda masih berada dalam mode user,ganti ke mode admin terlebih dahulu");
```

```
  }
  }
if( btdata.indexOf("DRIVER1") >= 0 ){
  if(adminservice==1){
  detachInterrupt(digitalPinToInterrupt(reedswitch));
  detachInterrupt(digitalPinToInterrupt(listrik));
  countbt=0;
  Serial.println("DRIVER AKTIF");
  digitalWrite(4,LOW);
  digitalWrite(5,LOW);
  digitalWrite(6, HIGH);
  digitalWrite(7, LOW);
  digitalWrite(8, LOW);
  digitalWrite(5, HIGH);
  delay(200);
  digitalWrite(5, LOW);
  delay(50);
  digitalWrite(5, HIGH);
  delay(200);
  digitalWrite(5, LOW);
  mode=2;
  }
  else{
  SIM808.sendSMS(phone,"Anda masih berada dalam mode
user,ganti ke mode admin terlebih dahulu");
  }
  }
if(mode==2){
  if( btdata.indexOf("PING") >= 0 ){
    koneksi=1;
    countbt=0;
    }
  else{
    koneksi=0;
    countbt++;
  }
  }
if(adminservice==0){
  if( btdata.indexOf("PING") >= 0 ){
```

```
    koneksi=1;
    countbt=0;
    }
  else{
    koneksi=0;
    countbt++;
  }
}
}

void loopmode(){
if(adminservice==0){
  Serial.println("modeuser");
    if(countbt<2){
      digitalWrite(4,LOW);
  }
    if(countbt>2){
      digitalWrite(4,HIGH);
    }
SIM808.getDateTime(tanggal, jam, menit);
Serial.println();
Serial.print("Waktu Sekarang: Tanggal ");
Serial.print(tanggal);
Serial.print(" ");
Serial.print(jam);
Serial.print("-");
Serial.println(menit);
Serial.print("Waktu Mulai: Tanggal ");
Serial.print(tanggal1);
Serial.print(" ");
Serial.print(jam1);
Serial.print("-");
Serial.println(menit1);
Serial.println("User mode berlangsung 1x24jam");
    if(tanggal!=tanggal1){
      if(jam==jam1){
      digitalWrite(4,HIGH);
      SIM808.sendSMS(phone,location);
      adminservice=1;
```

```
    }
   }
  }
if(mode==0){
  Serial.println("dissable");
  mode=0;
  }
if(mode==1){
  Serial.println("modeparkir");
  if(count>24){
     detachInterrupt(digitalPinToInterrupt(reedswitch));
     detachInterrupt(digitalPinToInterrupt(listrik));
     call=1;
    }
  if(sensorlistrik==1){
    detachInterrupt(digitalPinToInterrupt(reedswitch));
    detachInterrupt(digitalPinToInterrupt(listrik));
    call=1;
   }
  if(call==1){
   if(calling<1){
    SIM808.sendSMS(phone,"Motor anda dalam bahaya");
    }
   while(calling<2){
    SIM808.callUp(phone);
    delay(25000);
    Serial.println("ATH");
    delay(800);
    Serial.println("ATH");
    delay(1000);
    calling++;
   }
   digitalWrite(5,LOW);
  }
   mode=1;
  }
  if(mode==2){
   Serial.println("modedriver");
   if(countbt>2){
```

```
    digitalWrite(4,HIGH);
    digitalWrite(5,HIGH);
    digitalWrite(6,HIGH);
    digitalWrite(7,HIGH);
    digitalWrite(8,HIGH);

   }
   mode=2;
  }
 }
```

- *Library* yang digunakan

  - Cek sim status

```
bool DFRobot_SIM808::checkSIMStatus(void)
{
   char gprsBuffer[32];
   int count = 0;
   SIM808_clean_buffer(gprsBuffer,32);
   while(count < 3) {
      SIM808_send_cmd("AT+CPIN?\r\n");

SIM808_read_buffer(gprsBuffer,32,DEFAULT_TIME
OUT);
      if((NULL != strstr(gprsBuffer,"+CPIN:
READY"))) {
         break;
      }
      count++;
      delay(300);
   }
   if(count == 3) {
      return false;
   }
   return true;
}
```

  - Send SMS

```
bool DFRobot_SIM808::sendSMS(char *number, char
*data)
{
      if(!SIM808_check_with_cmd("AT+CMGF=1\r\n",
"OK\r\n", CMD)) { // Set message mode to ASCII
       return false;
    }
    delay(500);
       SIM808_flush_serial();
       SIM808_send_cmd("AT+CMGS=\"");
       SIM808_send_cmd(number);
       if(!SIM808_check_with_cmd("\"\r\n",">",CMD))
{
       return false;
    }
    delay(1000);
    SIM808_send_cmd(data);
    delay(500);
    SIM808_send_End_Mark();
    return SIM808_wait_for_resp("OK\r\n", CMD);
}
```

- SMS unread

```
 char DFRobot_SIM808::isSMSunread()
 {
    char gprsBuffer[48];
    char *s;

 SIM808_check_with_cmd("AT+CMGF=1\r\n","OK\r
 \n",CMD);
    delay(1000);

    SIM808_send_cmd(F("AT+CMGL=\"REC
 UNREAD\",1\r\n"));

    SIM808_clean_buffer(gprsBuffer,31);

 SIM808_read_buffer(gprsBuffer,30,DEFAULT_TIM
 EOUT);
```

```
    if(NULL != ( s = strstr(gprsBuffer,"OK"))) {
      delay(50);
      return 0;
    } else {
      SIM808_wait_for_resp("OK\r\n", CMD);
      SIM808_send_cmd("AT+CMGL=\"REC
UNREAD\",1\r\n");
      SIM808_clean_buffer(gprsBuffer,48);

SIM808_read_buffer(gprsBuffer,47,DEFAULT_TIM
EOUT);
            if(NULL != ( s =
strstr(gprsBuffer,"+CMGL:"))) {
        s = strstr(gprsBuffer,":");
        if (s != NULL) {
          SIM808_wait_for_resp("OK\r\n", CMD);
          return atoi(s+1);
        }
      } else {
        return -1;

      }
    }
    return -1;
}
```

- *Bluetooth* status

```
char DFRobot_SIM808::isBTstatus()
{
  char gprsBuffer[48];
  char *s;
  SIM808_send_cmd(F("AT+BTSTATUS?\r\n"));
  SIM808_clean_buffer(gprsBuffer,31);

SIM808_read_buffer(gprsBuffer,30,DEFAULT_TIME
OUT);
  if(NULL != ( s = strstr(gprsBuffer,"OK"))) {
            delay(50);
```

```
        return 0;
    } else {
        SIM808_wait_for_resp("OK\r\n", CMD);
        SIM808_send_cmd("AT+BTSTATUS?\r\n");
        SIM808_clean_buffer(gprsBuffer,48);

SIM808_read_buffer(gprsBuffer,47,DEFAULT_TIME
OUT);
        if(NULL != ( s = strstr(gprsBuffer,"+BTSTATUS:"))) {
            s = strstr(gprsBuffer,":");
            if (s != NULL) {
                SIM808_wait_for_resp("OK\r\n", CMD);
                return atoi(s+1);
            }
        } else {
            return -1;
        }
    }
    return -1;
}
```

- GPS status

```
char DFRobot_SIM808::isGPSstatus()
{
    char gprsBuffer[48];
    char *s;
    SIM808_send_cmd(F("AT+CGPSPWR?\r\n"));
    SIM808_clean_buffer(gprsBuffer,31);

SIM808_read_buffer(gprsBuffer,30,DEFAULT_TIME
OUT);
    if(NULL != ( s = strstr(gprsBuffer,"OK"))) {
        delay(50);
        return 0;
    } else {
        SIM808_wait_for_resp("OK\r\n", CMD);
        SIM808_send_cmd("AT+CGPSPWR?\r\n");
```

```
        SIM808_clean_buffer(gprsBuffer,48);

    SIM808_read_buffer(gprsBuffer,47,DEFAULT_TIME
    OUT);
        if(NULL != ( s = strstr(gprsBuffer,"+CGNSPWR:
    "))) {
            s = strstr(gprsBuffer,":");
            if (s != NULL) {
              SIM808_wait_for_resp("OK\r\n", CMD);
              return atoi(s+1);
            }
        } else {
            return -1;

        }
      }
      return -1;
    }
```

- *Bluetooth* data

```
    bool DFRobot_SIM808::isBTdata(char *btdata)
    {   int i=0;
      char gprsBuffer[48];
        char *p,*p2,*s;
        SIM808_send_cmd(F("AT+BTSPPGET=3\,10\r\n
    "));
      SIM808_clean_buffer(gprsBuffer,31);

    SIM808_read_buffer(gprsBuffer,30,DEFAULT_TIME
    OUT);
      if(NULL != ( s = strstr(gprsBuffer,"OK"))) {
        delay(50);
        return 0;
      } else {
        SIM808_wait_for_resp("OK\r\n", CMD);
        SIM808_send_cmd("AT+BTSPPGET=3\,10\r\n");
        SIM808_clean_buffer(gprsBuffer,48);
```

```
SIM808_read_buffer(gprsBuffer,47,DEFAULT_TIME
OUT);
      if(NULL        !=        (        s        =
strstr(gprsBuffer,"+BTSPPGET:"))) {
                s = strstr((char *)(s),",");
      s = s + 1;
      p = strstr((char *)(s),"\$");
      if (NULL != s) {
         i = 0;
         while (s < p) {
           btdata[i++] = *(s++);
         }
         btdata[i] = '\0';
                    }
                                    i=0;
              return true;
      }

   }
}
```

- Read SMS

```
bool DFRobot_SIM808::readSMS(int messageIndex,
char *message, int length, char *phone, char *datetime)
{
    int i = 0;
   char gprsBuffer[80 + length];
     char num[4];
   char *a,*a2,*b;


SIM808_check_with_cmd("AT+CMGF=1\r\n","OK\r\
n",CMD);
   delay(500);
     SIM808_send_cmd("AT+CMGR=");
     itoa(messageIndex, num, 10);
     SIM808_send_cmd(num);
     SIM808_send_cmd("\r\n");
```

```
SIM808_clean_buffer(gprsBuffer,sizeof(gprsBuffer));
   SIM808_read_buffer(gprsBuffer,sizeof(gprsBuffer));
   if(NULL != ( b = strstr(gprsBuffer,"+CMGR:"))){
      a = strstr(b,",");
                  a2 = a + 2;
      a = strstr((char *)(a2), "\"");
      if (NULL != a) {
         i = 0;
         while (a2 < a) {
            phone[i++] = *(a2++);
         }
         phone[i] = '\0';
      }
      a = strstr((char *)(a2),",");
      a2 = a + 1;
      a = strstr((char *)(a2), ",");
      a2 = a + 2;
      a = strstr((char *)(a2), "\"");
      if (NULL != a) {
         i = 0;
         while (a2 < a) {
            datetime[i++] = *(a2++);
         }
         datetime[i] = '\0';
      }
      if(NULL != ( b = strstr(b,"\r\n"))){
         i = 0;
         a = b + 2;
         while((*a != '\r')&&(i < length-1)) {
            message[i++] = *(a++);
         }
         message[i] = '\0';
      }
      return true;
   }
   return false;
}
```

```
bool DFRobot_SIM808::readSMS(int messageIndex,
char *message,int length)
{
   int i = 0;
   char gprsBuffer[100];
     char num[4];
   char *a,*b;


SIM808_check_with_cmd("AT+CMGF=1\r\n","OK\r\
n",CMD);
   delay(500);
      SIM808_send_cmd("AT+CMGR=");
      itoa(messageIndex, num, 10);
      SIM808_send_cmd(num);
      SIM808_send_cmd("\r\n");

SIM808_clean_buffer(gprsBuffer,sizeof(gprsBuffer));

SIM808_read_buffer(gprsBuffer,sizeof(gprsBuffer),DE
FAULT_TIMEOUT);
   if(NULL != ( b = strstr(gprsBuffer,"+CMGR:"))){
      if(NULL != ( b = strstr(b,"\r\n"))){
         a = b + 2;
         while((*a != '\r')&&(i < length-1)) {
            message[i++] = *(a++);
         }
         message[i] = '\0';
         return true;
      }
   }
   return false;
}
```

- Delete SMS

```
bool DFRobot_SIM808::deleteSMS(int index)
{
     char num[4];
```

```
    SIM808_send_cmd("AT+CMGD=");
      itoa(index, num, 10);
      SIM808_send_cmd(num);
      return
SIM808_check_with_cmd("\r","OK\r\n",CMD);
}
```

- Get date time

```
bool                DFRobot_SIM808::getDateTime(char
*tanggal,char *jam,char *menit)
{   byte i = 0;
   char gprsBuffer[50];
   char *p,*s,*s2,*p2;
     SIM808_flush_serial();
   SIM808_send_cmd("AT+CCLK?\r\n");
   SIM808_clean_buffer(gprsBuffer,50);

SIM808_read_buffer(gprsBuffer,50,DEFAULT_TIME
OUT);
   if(NULL != ( s = strstr(gprsBuffer,"+CCLK:"))) {
        s = strstr((char *)(s),"\"");
     s = s + 1;
     p = strstr((char *)(s),"/");
              s = strstr((char *)(s),"/");
              s = s + 1;
              p = strstr((char *)(s),"/");
              s = strstr((char *)(s),"/");
              s = s + 1;
              p = strstr((char *)(s),",");
     if (NULL != s) {
        i = 0;
        while (s < p) {
          tanggal[i++] = *(s++);
        }
        tanggal[i] = '\0';
     }
              s = strstr((char *)(s),",");
              s = s + 1;
              p = strstr((char *)(s),":");
```

```
    if (NULL != s) {
       i = 0;
       while (s < p) {
         jam[i++] = *(s++);
       }
       jam[i] = '\0';
    }
                s = strstr((char *)(s),":");
                s = s + 1;
                p = strstr((char *)(s),":");
    if (NULL != s) {
       i = 0;
       while (s < p) {
         menit[i++] = *(s++);
       }
       menit[i] = '\0';
    }
    return true;
  }
  return false;

}
```

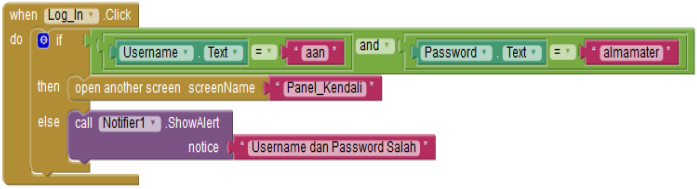
- Get GPS lokasi

```
bool DFRobot_SIM808::getGPSlokasi(char *location)
{   int i = 0;
  char gprsBuffer[96];
  char *p,*p2,*s,*s2;
  delay(500);
    SIM808_send_cmd("AT+CGPSINF=0\r\n");
  SIM808_clean_buffer(gprsBuffer,96);
  SIM808_read_buffer(gprsBuffer,96);
  if(NULL != ( s = strstr(gprsBuffer,"+CGPSINF:
")))){
    p = strstr(s,",");
            p2 = p + 1;
    p = strstr((char *)(p2), ",");
    s2 = p + 1;
    s = strstr((char *)(s2), "\,");
```

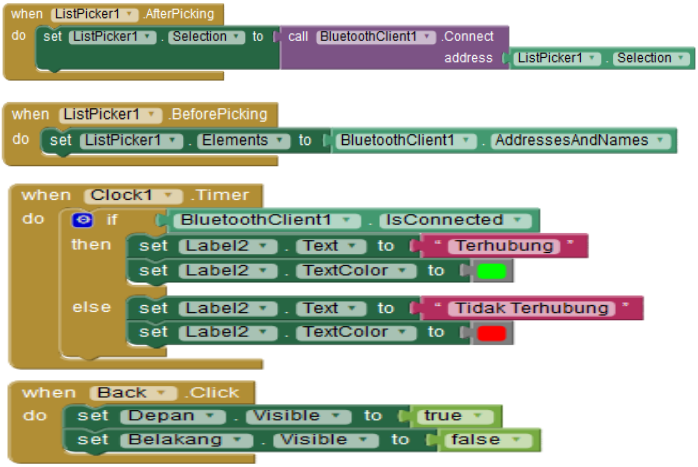
```
            if (NULL != p) {
               i = 0;
               while (p2 < s) {
                  location[i++] = *(p2++);
               }
               location[i] = '\0';
               }
            return true;
         }
         return false;

      }
```

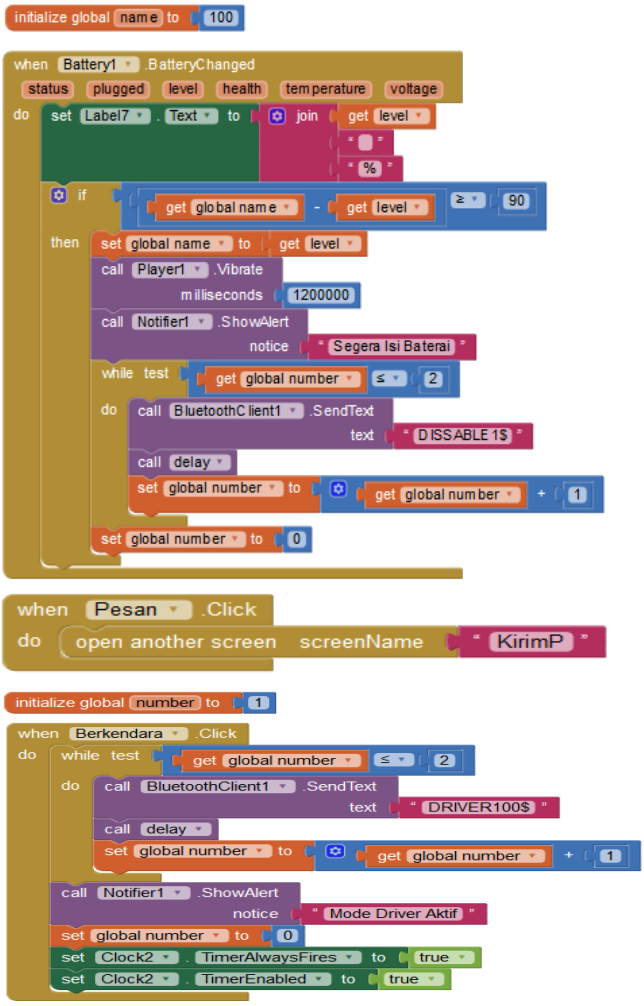
A.2. *Block* Program pada App Inventor

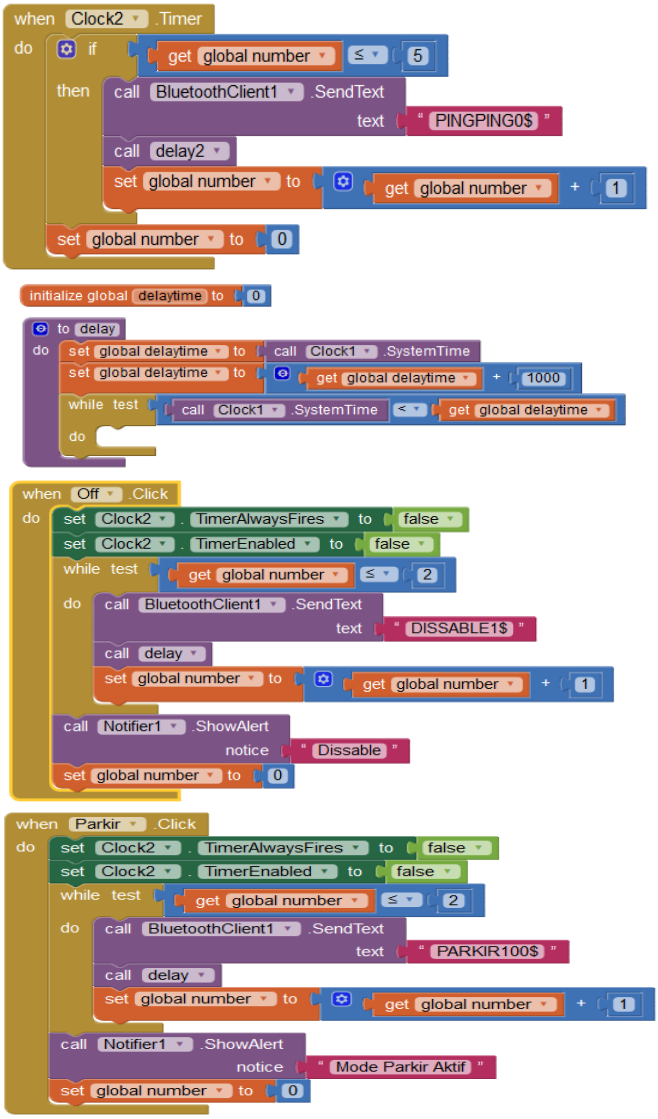
a. *Block* Program pada Menu *Log In*



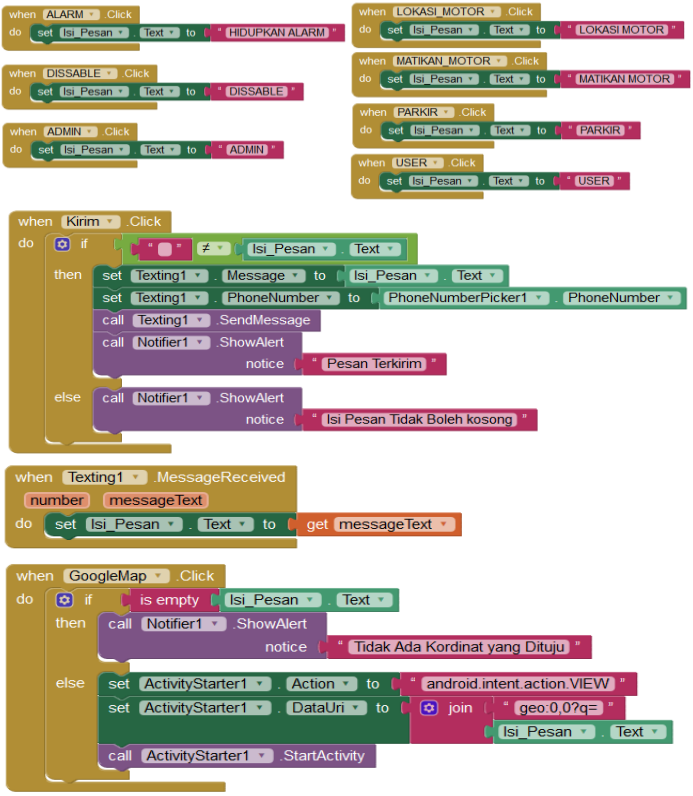
b. *Block* Program pada Menu Utama



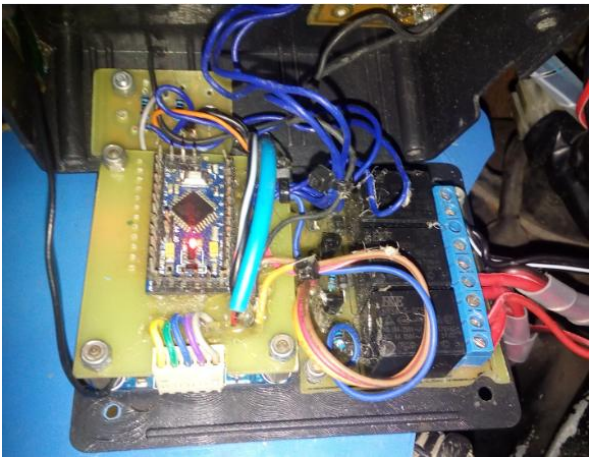

initialize global name to 100
when Battery1 .BatteryChanged
status plugged level health temperature voltage
do set Label7 . Text to join get level
" "
" % "
if get global name - get level ≥ 90
then set global name to get level
call Player1 .Vibrate
milliseconds 1200000
call Notifier1 .ShowAlert
notice " Segera Isi Baterai "
while test get global number ≤ 2
do call BluetoothClient1 .SendText
text " DISSABLE1$ "
call delay
set global number to get global number + 1
set global number to 0
when Pesan .Click
do open another screen screenName " KirimP "
initialize global number to 1
when Berkendara .Click
do while test get global number ≤ 2
do call BluetoothClient1 .SendText
text " DRIVER100$ "
call delay
set global number to get global number + 1
call Notifier1 .ShowAlert
notice " Mode Driver Aktif "
set global number to 0
set Clock2 . TimerAlwaysFires to true
set Clock2 . TimerEnabled to true


when Clock2 .Timer
do if get global number ≤ 5
then call BluetoothClient1 .SendText
text " PINGPING0$ "
call delay2
set global number to get global number + 1
set global number to 0
initialize global delaytime to 0
to delay
do set global delaytime to call Clock1 .SystemTime
set global delaytime to get global delaytime + 1000
while test call Clock1 .SystemTime < get global delaytime
do
when Off .Click
do set Clock2 . TimerAlwaysFires to false
set Clock2 . TimerEnabled to false
while test get global number ≤ 2
do call BluetoothClient1 .SendText
text " DISSABLE1$ "
call delay
set global number to get global number + 1
call Notifier1 .ShowAlert
notice " Dissable "
set global number to 0
when Parkir .Click
do set Clock2 . TimerAlwaysFires to false
set Clock2 . TimerEnabled to false
while test get global number ≤ 2
do call BluetoothClient1 .SendText
text " PARKIR100$ "
call delay
set global number to get global number + 1
call Notifier1 .ShowAlert
notice " Mode Parkir Aktif "
set global number to 0

c. *Block* Program pada Menu Kirim Pesan

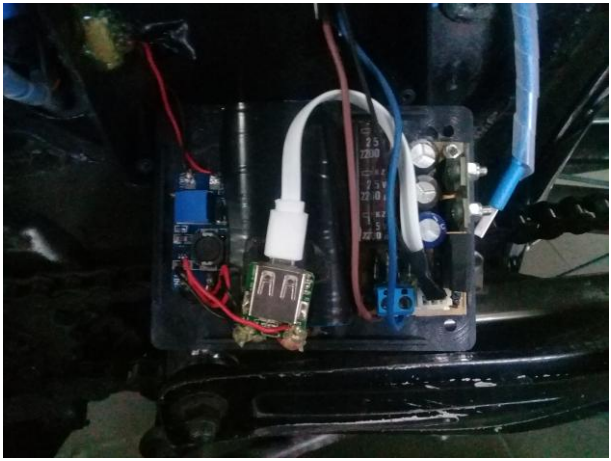


A.3. Konfigurasi Rangkaian *Driver Relay,* Arduino dan SIM808



Gambar A.1 Konfigurasi Rangkaian *Driver Relay*, Arduino dan SIM808

A.4. Konfigurasi Rangkaian *Power Supply* dan *Power Bank*



Gambar A.2 Konfigurasi Rangkaian *Power Supply* dan *Power Bank*

LAMPIRAN B

B.1. DATASHEET SIM808

1 Introduction

This document describes SIM808 hardware interface in great detail. This document can help user to quickly understand SIM808 interface specifications, electrical and mechanical details. With the help of this document and other SIM808 application notes, user guide, users can use SIM808 to design various applications quickly.

2 SIM808 Overview

Designed for global market, SIM808 is integrated with a high performance GSM/GPRS engine, a GPS engine and a BT engine. The GSM/GPRS engine is a quad-band GSM/GPRS module that works on frequencies GSM 850MHz, EGSM 900MHz, DCS 1800MHz and PCS 1900MHz. SIM808 features GPRS multi-slot class 12/ class 10 (optional) and supports the GPRS coding schemes CS-1, CS-2, CS-3 and CS-4. The GPS solution offers best-in-class acquisition and tracing sensitivity, Time-To-First-Fix (TTFF) and accuracy.

With a tiny configuration of 24*24*2.6mm, SIM808 can meet almost all the space requirements in user applications, such as M2M, smart phone, PDA, tracker and other mobile devices.

SIM808 has 68 SMT pads, and provides all hardware interfaces between the module and customers' boards.

- Support 4*4*2 keypads.
- One full modem serial port.
- One USB, the USB interfaces can debug, download software.
- Audio channels which include a microphone input and a receiver output.
- One SIM card interface.
- Charging interface.
- Programmable general purpose input and output.
- Support Bluetooth function.
- Support PWM and ADC.
- PCM/SPI/SD card interface, only one function can be accessed synchronously. (Default function is PCM).

SIM808 is designed with power saving technique so that the current consumption is as low as 1mA in sleep mode (GPS engine is powered down).
SIM808 integrates TCP/IP protocol and extended TCP/IP AT commands which are very useful for data transfer applications. For details about TCP/IP applications, please refer to *document [2]*.

2.1 SIM808 Key Features

Table 1: SIM808 GSM/GPRS engine key features

Feature	Implementation
Power supply	3.4V ~ 4.4V
Power saving	Typical power consumption in sleep mode is 1mA (BS-PA-MFRMS=9, GPS engine is powered down)
Charging	Supports charging control for Li-Ion battery
Frequency bands	• SIM808 Quad-band: GSM 850, EGSM 900, DCS 1800, PCS 1900. SIM808

	can search the 4 frequency bands automatically. The frequency bands also can be set by AT command "AT+CBAND". For details, please refer to *document [1]*. • Compliant to GSM Phase 2/2+
Transmitting power	• Class 4 (2W) at GSM 850 and EGSM 900 • Class 1 (1W) at DCS 1800 and PCS 1900
GPRS connectivity	• GPRS multi-slot class 12 (default) • GPRS multi-slot class 1~12 (optional)
Temperature range	• Normal operation: -40℃ ~+85℃ • Storage temperature -45℃~ +90℃
Data GPRS	• GPRS data downlink transfer: max. 85.6 kbps • GPRS data uplink transfer: max. 85.6 kbps • Coding scheme: CS-1, CS-2, CS-3 and CS-4 • PAP protocol for PPP connect • Integrate the TCP/IP protocol. • Support Packet Broadcast Control Channel (PBCCH) • CSD transmission rates：2.4，4.8，9.6，14.4 kbps
CSD	• Support CSD transmission
USSD	• Unstructured Supplementary Services Data (USSD) support
SMS	• MT, MO, CB, Text and PDU mode • SMS storage: SIM card
SIM interface	Support SIM card: 1.8V, 3V
External antenna	Antenna pad

Audio features	Speech codec modes: ● Half Rate (ETS 06.20) ● Full Rate (ETS 06.10) ● Enhanced Full Rate (ETS 06.50 / 06.60 / 06.80) ● Adaptive multi rate (AMR) ● Echo Cancellation ● Noise Suppression
Serial port and USB interface	**Serial port:** ● Full modem interface with status and control lines, unbalanced, asynchronous. ● 1200bps to 115200bps. ● Can be used for AT commands or data stream. ● Support RTS/CTS hardware handshake and software ON/OFF flow control. ● Multiplex ability according to GSM 07.10 Multiplexer Protocol. ● Autobauding supports baud rate from 1200 bps to 115200bps. **USB interface:** ● Can be used as debugging and firmware upgrading.
Phonebook management	Support phonebook types: SM, FD, LD, RC, ON, MC.
SIM application toolkit	GSM 11.14 Release 99
Real time clock	Support RTC
Alarm function	Can be set by AT command
Physical characteristics	Size: 24*24*2.6mm Weight: 3.5g

2.3 SIM808 Functional Diagram

The following figure shows a functional diagram of SIM808:

- The GSM baseband engine
- The GPS engine
- Flash
- The GSM radio frequency part
- The antenna interface
- The other interfaces

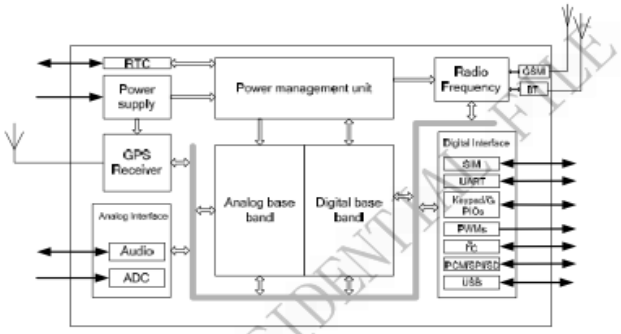


Figure 1: SIM808 functional diagram

3.1 Pin out Diagram

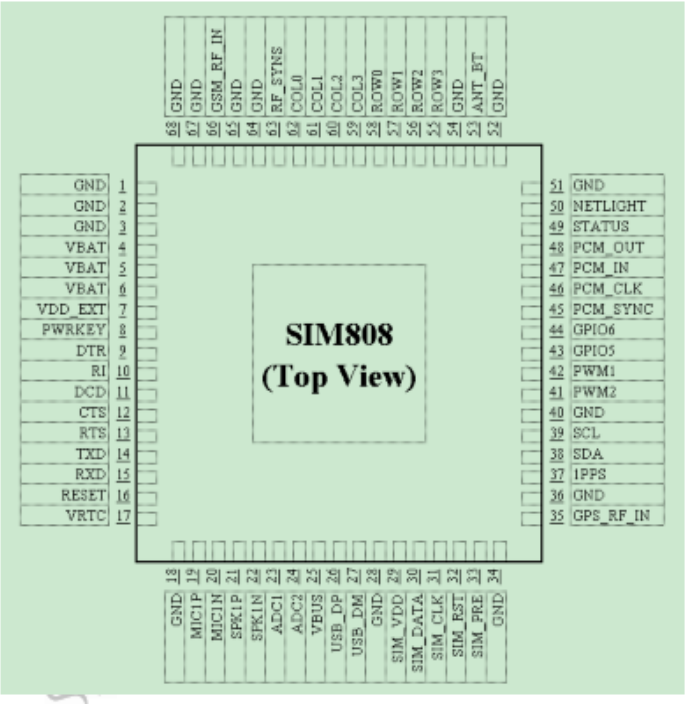


Figure 2: SIM808 pin out diagram (Top view)

4.9 Bluetooth

SIM808 supports Bluetooth function, customer only needs to design the Bluetooth antenna, and then customer can operate Bluetooth conveniently by AT commands. For the detail commands about Bluetooth please refer to *document [1]*

- Fully compliant with Bluetooth specification3.0 + EDR
- Support operation with GPS and GSM/GPRS worldwide radio systems
- Fully integrated PA provides 10dbm output power
- Up to 4 simultaneous active ACL links
- Support sniff mode

Support PCM interface and built-in programmable transcoders for liner voice with transmission.

4.9.1 Bluetooth Antenna Interface

The module provides a Bluetooth antenna interface named ANT_BT. External antenna must be matched properly to achieve best performance, so the matching circuit is necessary, the connection is recommended as the following figure:

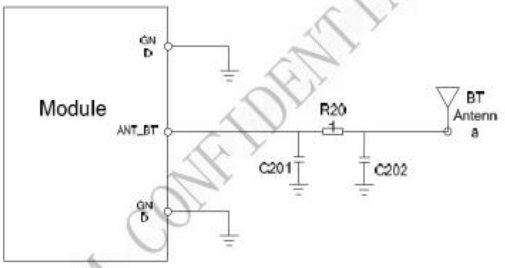
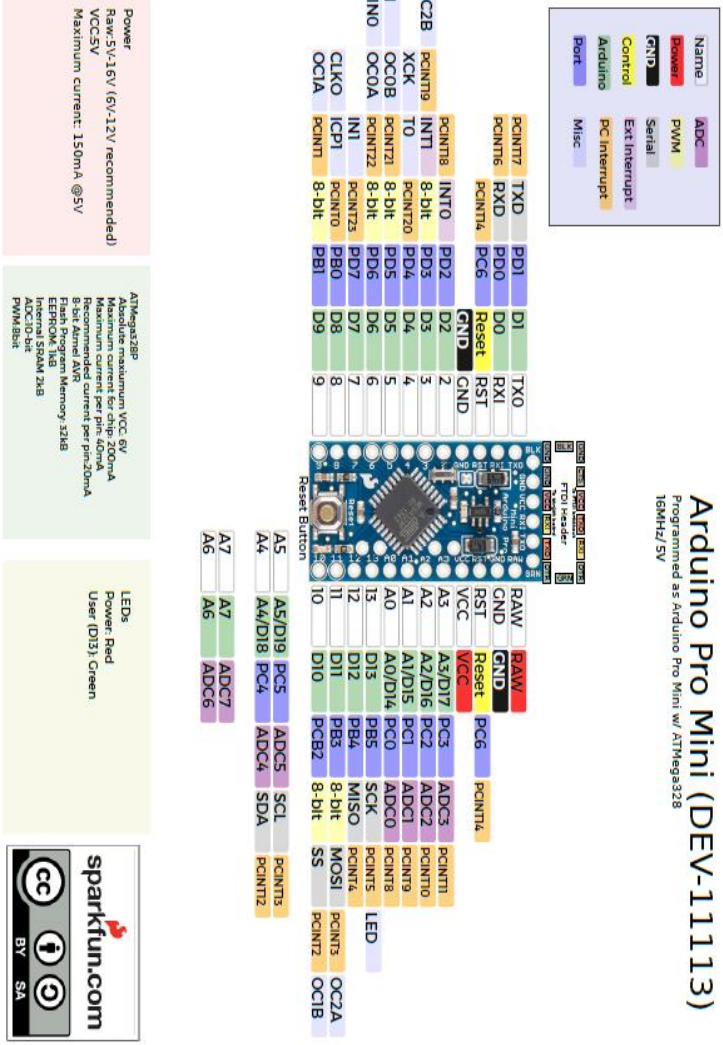


Figure 35: Bluetooth antenna matching circuit

R201, C201, C202 are the matching circuit, the values depend on antenna debug result. Normally R201 is 0Ω, C201 and C202 are not mounted.

B.2. DATASHEET ARDUINO PRO MINI



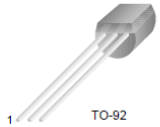
B.3. DATASHEET TRANSISTOR BC547

FAIRCHILD
SEMICONDUCTOR®

BC546/547/548/549/550

Switching and Applications

- High Voltage: BC546, V_{CEO}=65V
- Low Noise: BC549, BC550
- Complement to BC556 ... BC560



TO-92

1. Collector 2. Base 3. Emitter

NPN Epitaxial Silicon Transistor

Absolute Maximum Ratings T_a=25°C unless otherwise noted

Symbol	Parameter	Value	Units
V_{CBO}	Collector-Base Voltage : BC546	80	V
	: BC547/550	50	V
	: BC548/549	30	V
V_{CEO}	Collector-Emitter Voltage : BC546	65	V
	: BC547/550	45	V
	: BC548/549	30	V
V_{EBO}	Emitter-Base Voltage : BC546/547	6	V
	: BC548/549/550	5	V
I_C	Collector Current (DC)	100	mA
P_C	Collector Power Dissipation	500	mW
T_J	Junction Temperature	150	°C
T_{STG}	Storage Temperature	-65 ~ 150	°C

Electrical Characteristics T_a=25°C unless otherwise noted

Symbol	Parameter	Test Condition	Min.	Typ.	Max.	Units
I_{CBO}	Collector Cut-off Current	V_{CB}=30V, I_E=0			15	nA
h_{FE}	DC Current Gain	V_{CE}=5V, I_C=2mA	110		800	
V_{CE} (sat)	Collector-Emitter Saturation Voltage	I_C=10mA, I_B=0.5mA		90	250	mV
		I_C=100mA, I_B=5mA		200	600	mV
V_{BE} (sat)	Base-Emitter Saturation Voltage	I_C=10mA, I_B=0.5mA		700		mV
		I_C=100mA, I_B=5mA		900		mV
V_{BE} (on)	Base-Emitter On Voltage	V_{CE}=5V, I_C=2mA	580	660	700	mV
		V_{CE}=5V, I_C=10mA			720	mV
f_T	Current Gain Bandwidth Product	V_{CE}=5V, I_C=10mA, f=100MHz		300		MHz
C_{ob}	Output Capacitance	V_{CB}=10V, I_E=0, f=1MHz		3.5	6	pF
C_{ib}	Input Capacitance	V_{EB}=0.5V, I_C=0, f=1MHz		9		pF
NF	Noise Figure : BC546/547/548	V_{CE}=5V, I_C=200µA		2	10	dB
	: BC549/550	f=1KHz, R_G=2KΩ		1.2	4	dB
	: BC549	V_{CE}=5V, I_C=200µA		1.4	4	dB
	: BC550	R_G=2KΩ, f=30~15000MHz		1.4	3	dB

h_{FE} Classification

Classification	A	B	C
h_{FE}	110 ~ 220	200 ~ 450	420 ~ 800

-----Halaman ini sengaja dikosongkan-----

DAFTAR RIWAYAT HIDUP

Nama	: Nasrul Arifianto
TTL	: Surabaya, 10 September 1995
Jenis Kelamin	: Laki-laki
Agama	: Islam
Alamat	: Tenggumung Baru Mulya 1/29A, Surabaya
Telpn/Hp	: 085648594191
Email	: nasroel.stemba@gmail.com

RIWAYAT PENDIDIKAN

1. 2001-2007 : SDN Sidotopo Wetan 255 Surabaya
2. 2007-2010 : SMPN 15 Surabaya
3. 2010-2014 : SMKN 5 Surabaya
4. 2014-2017 : Departemen Teknik Elektro Otomasi Program Studi Komputer Kontrol – Fakultas Vokasi (FV) Institut Teknologi Sepuluh Nopember

PENGALAMAN KERJA

1. Kerja Praktek di CV.Global Teknik
2. Kerja Praktek di PT. Perusahaan Gas Negara Distribusi Wilayah II

PENGALAMAN ORGANISASI

1. Staff Departemen Kewirausahaan Himpunan Mahasiswa D3 Teknik Elektro
2. Staff Departemen Kewirausahaan Badan Eksekutif Fakultas Teknologi Industri

-----Halaman ini sengaja dikosongkan-----

DAFTAR RIWAYAT HIDUP

Nama	: Moch Aan Fahrizal
TTL	: Surabaya, 02 Maret 1996
Jenis Kelamin	: Laki-laki
Agama	: Islam
Alamat	: Wisma Lidah Kulon Blok xi/16, Surabaya
Telpn/Hp	: 08967709212
Email	: aanfahrizal48 @gmail.com

RIWAYAT PENDIDIKAN

1. 2002-2008 : SDN Gunungsari IV Surabaya
2. 2008-2011 : SMPN 10 Surabaya
3. 2011-2014 : SMAN 18 Surabaya
4. 2014-2017 : Departemen Teknik Elektro Otomasi Program Studi Komputer Kontrol – Fakultas Vokasi (FV) Institut Teknologi Sepuluh Nopember

PENGALAMAN KERJA

3. Kerja Praktek di APD Jawa Timur PT. PLN Persero

PENGALAMAN ORGANISASI

3. OSIS SMPN 10 Surabaya 2009-2010
4. Wakil Ketua OSIS SMAN 18 Surabaya
5. Staff Departemen Kesejahteraan Mahasiswa Himpunan Mahasiswa D3 Teknik Elektro
6. Ketua Biro Akademik dan Kesehatan Departemen Kesejahteraan Mahasiswa Himpunan Mahasiswa D3 Teknik Elektro