APLIKASI HOME MONITORING DAN KONTROL BERBASIS ANDROID

SKRIPSI

Disusun oleh :

SEPTIAN TRI HARJANTO

NIM 09.12.523

PROGRAM STUDI TEKNIK ELEKTRO S-1

KONSENTRASI TEKNIK KOMPUTER

FAKULTAS TEKNOLOGI INDUSTRI

INSTITUT TEKNOLOGI NASIONAL MALANG

2014

LEMBAR PERSETUJUAN

APLIKASI HOME MONITORING DAN KONTROL BERBASIS ANDROID

SKRIPSI

Disusun dan diajukan untuk melengkapi dan memenuhi persyaratan guna mencapai gelar Sarjana Teknik

Disusun oleh :

SEPTIAN TRI HARJANTO

NIM. 0912523

Diperiksa dan Disetujui,

Dosen Pembimbing I	Dosen Pembimbing II
Ir. F. Yudi Limpraptono, MT	**Bima Aulia Firmandani, ST**
NIP. Y. 1039500274	

Mengetahui,
Ketua Program Studi Teknik Elektro S-1

M. Ibrahim Ashari, ST, MT
NIP. P. 1030100358

JURUSAN TEKNIK ELEKTRO S-1
KONSENTRASI TEKNIK KOMPUTER
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
2014

SURAT PERNYATAAN ORISINALITAS

Yang bertanda tangan di bawah ini :

Nama	: SEPTIAN TRI HARJANTO
NIM	: 09.12.523
Program Studi	: Teknik Elektro S-1
Konsentrasi	: Teknik Komputer

Dengan ini menyatakan bahwa Skripsi yang telah dibuat adalah hasil karya sendiri, bukan merupakan plagiasi dari karya orang lain. Dalam Skripsi ini tidak memuat karya orang lain, kecuali dicantumkan sumbernya sesuai dengan ketentuan yang berlaku.

Demikian surat pernyataan ini saya buat dengan kesadaran penuh, apabila di kemudia hari terdapat pelanggaran atas surat pernyataan ini, saya selaku pembuat sekaligus penulis Skripsi ini bersedia menerima sanksinya.

Malang, 15 Februari 2014

Pembuat pernyataan,

Septian Tri Harjanto

NIM. 0912523

ABSTRAK

APLIKASI HOME MONITORING DAN KONTROL BERBASIS ANDROID

Septian Tri Harjanto, NIM 09.12.523
e-mail : *septiantriharjanto@yahoo.com*
Dosen Pembimbing : Ir. F. Yudi Limpraptono, MT dan Bima Aulia Firmadani, ST

Banyaknya kasus terjadinya kebocoran tabung gas LPG yang dapat mengakibatkan ledakan dan pencurian yang terjadi ketika tuan rumah berada di dalam rumah yang dapat membahayakan nyawa tuan rumah, serta kebiasaan membiarkan lampu yang tidak diperlukan tetap menyala sehingga mengakibatkan pemborosan energi listrik membuat kita harus meningkatkan kewaspadaan dan kepekaan terhadap hal-hal tersebut.

Android dengan kelebihan yang ada di dalamnya dapat dijadikan sebagai media untuk membantu kita dapat memonitor kebocoran gas LPG, sensor photodioda dan mengontrol lampu. Sensor MQ-6 yang berfungsi sebagai pendeteksi gas LPG, sensor *Led* dan *photodioda* sebagai pendeteksi gerakan serta *driver relay* yang berguna untuk memutus arus dan tegangan yang akan mematikan dan menghidupkan lampu kita gunakan sebagai perangkat keras. Dengan *software Eclipse* kita membuat aplikasi android yang nantinya akan dikombinasikan dengan perangkat keras. Dengan aplikasi ini, alarm pada android akan menyala apabila mendeteksi konsentrasi gas yang berlebihan dan apabila ada sesuatu yang terdeteksi oleh sensor photodioda, kemudian dari aplikasi ini membuat kita mudah untuk mematikan dan menghidupkan lampu.

Kata kunci : *Android*, *Eclipse*, *Home Monitoring*

KATA PENGANTAR

Alhamdulillah wa laaillahailallah wallaahuakbar, puji syukur penulis panjatkan sebanyak-banyaknya kepada Allah SWT karena dengan kehendak-Nya lah penelitian yang berjudul "Aplikasi *Home* Monitoring dan Kontrol Berbasis Android" dapat diselesaikan.

Penelitian ini dibuat untuk memenuhi salah satu syarat dalam mendapatkan gelar Sarjana Teknik dan atas keberhasilan yang telah tercapai, penulis ingin mengucapkan terima kasih sebanyak-banyaknya kepada :

- Allah SWT, Tuhan semesta alam.
- Nabi Muhammad SAW, sang pembawa kebenaran.
- Ibu, Bapak, kakak, keponakan dan seluruh keluarga penulis yang telah mendukung sepenuhnya untuk penelitian ini.
- Ir. F. Yudi Limpraptono, MT, selaku Dosen pembimbing 1.
- Bima Aulia Firmandani, ST, selaku Dosen pembimbing 2.
- Bapak Ir. Soeparno Djiwo, MT, selaku Rektor Institut Teknologi Nasional Malang.
- Bapak Ir. H. Anang Subardi, MT, selaku Dekan Fakultas Teknologi Industri ITN Malang.
- Bapak Ibrahim Ashari, ST, MT, selaku Ketua Jurusan Elektro. ITN Malang.
- Bapak Aryuanto Soetedjo, Dr. Eng., ST., MT, Selaku Sekretaris Jurusan Teknik Elektro ITN Malang.

Penulis menyadari bahwa masih banyaknya kekurangan yang terdapat pada penelitian ini, oleh karena itu penulis berharap para pembaca dapat memberikan kritik dan saran yang membangun agar penelitian ini menjadi lebih sempurna dan dapat bermanfaat bagi semua insan.

Malang, Maret 2014

Penulis

KATA PENGANTAR

Alhamdulillah, segala puji syukur penulis panjatkan kehadirat Allah SWT karena dengan berkat-Nya lah penelitian yang berjudul "Aplikasi Wave Monitoring dan Kontrol Berbasis Android" dapat diselesaikan.

Penelitian ini dilihat untuk memenuhi salah satu syarat dalam mendapatkan gelar Sarjana Teknik dan atas keberhasilan yang telah tercapai, penulis mengucapkan terima kasih sebanyak-banyaknya kepada :

- Allah SWT, Tuhan semesta alam.
- Nabi Muhammad SAW, yang panutanya kita semua.
- Ibu, Bapak, kakak, keponakan dan seluruh keluarga penulis yang telah memberikan sepenuhnya untuk penelitian ini.
- Ir. Yuli Christyono, MT., selaku Dosen pembimbing 1.
- Firda Ardila Fitri, ST, selaku Dosen pembimbing 2.
- Bapak dr. Soepeno Djiwo, MT., selaku Rektor Institut Teknologi Nasional Malang.
- Bapak Ir. H. Anang Subardi, MT., selaku Dekan Fakultas Teknologi Industri ITN Malang.
- Bapak Ibrahim Ashari, ST, MT., selaku Ketua Jurusan Teknik ITN Malang.
- Bapak/Ibu dosen Sotedjo, Dr. Eng., ST., MT. Selaku Sekretaris Jurusan Teknik Elektro ITN Malang.

Penulis menyadari bahwa masih banyaknya kekurangan yang terdapat pada penelitian ini oleh karena itu penulis berharap para pembaca dapat memberikan kritik dan saran yang membangun agar penelitian ini menjadi lebih sempurna dan dapat bermanfaat bagi semua insan.

Malang, Maret 2014

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR GRAFIK

BAB I
PENDAHULUAN

1.1. Latar Belakang

Keberadaan teknologi adalah untuk mempermudah pekerjaan ataupun kegiatan manusia. Dengan teknologi, pekerjaan yang sebelumnya dilakukan dalam jangka waktu yang lama ataupun memulai proses yang rumit dapat diselesaikan dengan lebih efektif dan efisien. Hal ini yang membuat teknologi berkembang dengan sangat pesat. Antara lain cabang teknologi ini adalah dibidang elektronika dan komputer, contoh menjamurnya perangkat elektronik. Melalui elektronika, dapat di rekayasa perangkat yang memiliki fungsi-fungsi tertentu. Kebutuhan akan perangkat elektronika muncul karena manusia menginginkan suatu kemudahan. Teknologi *smart phone* telah menjadi suatu yang populer saat ini diseluruh dunia apalagi dengan sistem *open source* yang telah diterapkan pada *smart phone* tersebut. Teknologi ini telah digunakan pada sebagian besar kehidupan sebagai bentuk perkembangan dan kemajuan teknologi.

Saat ini sistem pengendalian pada gedung ataupun ruangan masih menggunakan sistem manual, yaitu dengan cara menekan tombol *on/off* saklar lampu, sehingga hal ini kurang efisien dalam melakukan pengontrolan lampu listrik tersebut dan mengakibatkan pemborosan biaya karena pemakaian yang berlebihan. Disamping itu juga peralatan yang dikendalikan lebih dari satu buah, dan jarak masing-masing peralatan berjauhan karena ruangan yang sangat besar, maka ini tentu saja tidak menghemat waktu dan tenaga manusia.

Di layar kaca belakangan ini kita makin sering melihat kasus-kasus ledakan tabung gas yang dikarenakan kebocoran tabung maupun kelalaian dari pihak tuan rumah. Tidak sedikit korban jiwa yang berjatuhan karena hal ini, serta kerugian materi yang juga besar.

Selain itu, kita sering mendengar maraknya kasus yang terjadi yaitu pencurian. Pencurian tidak hanya terjadi pada saat rumah berada dalam keadaan tidak berpenghuni, bahkan pada saat penghuni berada di dalam

rumah pun pencuri masih banyak yang beraksi sehingga ketika kita bangun di pagi hari kita hanya dapat menyadari barang-barang berharga kita telah lenyap dijarah oleh pencuri. Oleh karena itu hal ini patut di cegah dengan kewaspadaan kita setiap saat.

Berdasarkan masalah yang dikemukakan diatas, penulis ingin merancang sistem monitor rumah menggunakan *smart phone* berbasis *android.* Jika menggunakan perangkat ini akan membantu kita mempermudah mengontrol rumah kita dari kejadian-kejadian yang tidak diharapkan seperti kebakaran akibat ledakan gas, pencurian, dan pemborosan energi listrik.

1.2. Rumusan Masalah

Permasalahan yang akan dibahas dalam penulisan skripsi ini dapat dirumuskan sebgai berikut :

1. Bagaimana merancang sistem sistem kontrol *home* monitoring.
2. Bagaimana mengirimkan data dari mikrokontroler ke *smart phone* menggunakan *Wi-fi.*
3. Bagaimana menampilkan hasil pengiriman data dari mikrokontroler ke *smart phone.*

1.3. Tujuan Penelitian

Berdasarkan permasalahan yang dikemukakan pada sub-bab 1.2, tujuan penulisan skripsi ini adalah :

1. Mempermudah pengontrolan lampu yang tidak diperlukan sehingga dapat membantu penghematan energi.
2. Tindakan pencegahan terjadinya pencurian dalam rumah.
3. Mengetahui terjadinya kebocoran gas yang dapat menyebabkan kebakaran rumah, maka penanggulangan dini dapat dilakukan.

1.4. Batasan Masalah

Agar permasalahan yang dibahas tidak terlalu meluas, maka ruang lingkup pembahasannya sebagai berikut :

1. Pada penelitian ini, yang di kontrol hanya lampu listrik dan alat pengontrolnya menggunakan mikrokontroler.
2. Aplikasi ini hanya untuk sistem operasi *android* dan *versi android*-nya adalah *jellybean.*
3. Penelitian ini memonitoring sensor photodioda, sensor gas LPG dan lampu rumah.
4. Pengiriman data menggunakan *Wi-fi.*
5. *Software* yang digunakan dalam penelitian ini adalah *Eclipse.*
6. *Hardware* (*relay, sensor-sensor* dan *mikrokontroler)* maupun rangkaian elektronika yang digunakan dalam penelitian ini tidak dijelaskan secara rinci.

1.5. Metodologi

Metode penelitian yang akan dilakukan dalam penyusunan skripsi ini adalah :

1. Kajian Literatur

   Yaitu kajian pustaka untuk mempelajari teori-teori yang terkait melalui literatur yang ada, yang berhubungan dengan permasalahan.
2. Langkah – langkah pengerjaan
   - Mengidentifikasi masalah
   - Pembuatan program
   - Pengujian program

1.6. Sistematika Penulisan

Sistematika dari pembahasan didalam skripsi ini adalah sebagai berikut :

BAB I : PENDAHULUAN

Pada bab ini berisikan latar belakang, rumusan masalah, tujuan, batasan masalah, metodologi penelitian, dan sistematika penulisan.

BAB II : KAJIAN PUSTAKA

Pada bab ini menjelasan tentang dasar-dasar teori tentang aplikasi yang digunakan.

BAB III : ANALISA DAN PERANCANGAN SISTEM

Pada bab ini menjelaskan tentang pembuatan program dan tampilan program serta alat dan bahan.

BAB IV : IMPLEMENTASI DAN PENGUJIAN APLIKASI

Pada bab ini berisi tentang pengimplementasian program dan hasil program dalam bentuk analisa.

BAB V : PENUTUP

Berisi kesimpulan dan saran.

BAB II
KAJIAN PUSTAKA

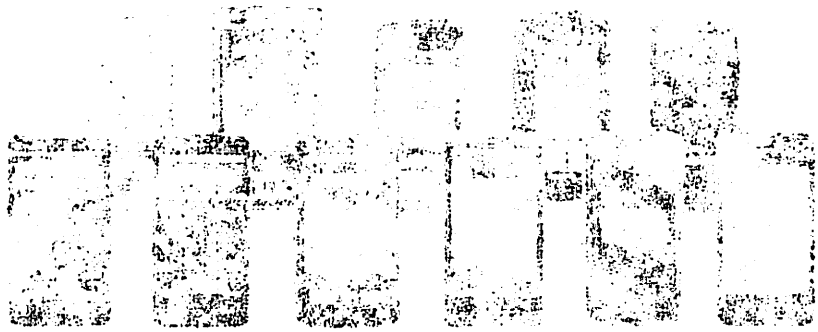
2.1. *Smart Phone*[12]

Telepon cerdas (*smart phone*) adalah telepon genggam yang mempunyai kemempuan tingkat tinggi, terkadang memiliki fungsi yang menyerupai komputer. Belum ada standar pabrik yang menentukan arti telepon cerdas secara mutlak. Bagi beberapa orang, telepon cerdas merupakan telepon yang bekerja menggunakan seluruh perangkat lunak sistem operasi yang menyediakan hubungan standar dan mendasar bagi pengembang aplikasi.

Bagi yang lainnya, telepon cerdas hanya merupakan sebuah telepon yang menyajikan fitur canggih seperti surel (surat elektronik), internet dan kemampuan untuk membaca buku elektronik (*e-book*) atau terdapat papan ketik (baik internal maupun eksternal) serta penghubung *VGA*. Namun untuk kalangan pengembang aplikasi, *smart phone* juga digunakan untuk objek penelitian karena mempunyai kemampuan yang sangat canggih. Dengan kata lain, telepon cerdas merupakan komputer kecil yang juga mempunyai kemampuan sebagai sebuah telepon.

Gambar 2.1. Contoh berbagai macam *Smart Phone*[12]

Pertumbuhan permintaan akan alat canggih yang mudah dibawa kemana-mana membuat kemajuan besar dalam prosesor, memori, layar dan sistem operasi yang diluar dari jalur telepon genggam sejak beberapa tahun ini.



2.2.Android[1]

Android adalah sistem operasi dengan sumber terbuka (*open source*) dan Google merilis kodenya dibawah lisensi *Apache*. Kode dengan *open source* yang diterapkan dan lisensi perizinan pada Android memungkinkan perangkat lunak dapat dimodifikasi secara bebas dan didistribusikan oleh para pembuat perangkat, operator nirkabel dan pengembang aplikasi. Selain itu, Android memiliki sejumlah besar komunitas pengembang aplikasi (*apps*) yang memperluas fungsionalitas perangkat, umumnya ditulis dalam versi kostumisasi bahasa pemrograman *Java*. Pada bulan Oktober 2012, ada sekitar 700.000 aplikasi tersedia untuk android dan sekitar 25 juta aplikasi telah diunduh dari *Google Play* (toko aplikasi utama Andoid).

Aplikasi Android dikembangkan dalam bahasa pemrograman java dengan menggunakan kit pengembangan perangkat lunak Android (*SDK*). *SDK* ini terdiri dari seperangkat perkakas pengembangan termasuk *debugger*, perpustakaan perangkat lunak, emulator *handset* yang berbasis *QEMU*, dokumentasi, kode sampel dan tutorial. Didukung secara resmi oleh lingkungan pengembangan terpadu (*IDE*) *Eclipse* yang menggunakan plugin *Android Development Tools* (*ADT*). Perkakas pengembangan lain yang tersedia diantaranya adalah *Native Development Kit* untuk aplikasi atau ekstensi dalam C atau C++, *Google App Inventor*, lingkungan visual untuk pemrogram pemula dan berbagai kerangka kerja aplikasi web selular lintas platform.[14]

Antarmuka pengguna Android adalah berdasarkan manipulasi langsung, menggunakan sentuhan bebas yang dibuat menyerupai kenyataan seperti menggesek, gerakan mencubit, menyentuh untuk memanipulasi objek layar. Perangkat tambahan seperti *akselerometer* (pengukur kecepatan), *giroskop* (penyeimbang gravitasi), *sensor jarak* dan perangkat lain juga dapat di tambahkan sesuai kegunaan. Contohnya merubah tampilan dari vertikal ke horizontal yang disesuaikan jika kita merubah arah telepon genggam dan lain-lain. Android juga memungkinkan pengguna untuk lebih mudah mengakses aplikasi-aplikasi yang ada di dalam telepon dengan bantuan *widget* (semacam jalan pintas / *shortcut*) yang dapat ditampilkan pada layar utama pada telepon

genggam. Dengan menggunakan bahasa Java, para pengembang aplikasi dapat membuat dan menginstal langsung ke dalam telepon genggam dengan "*.APK*" sebagai file ektensinya yang berisi tentang kelas-kelas, *layout* Java dan sebagainya.

Platform perangkat keras utama pada Android adalah 32-bit arsitektur *ARMv7*. Mendukung pada x86 dari proyek Android-x86, bahkan Google TV pun menggunakan versi x86 dari Android. Pada tahun 2013, salah satu sumber mengumumkan bahwa Android meluncurkan prosesor seri *i.MX*, *i.MX5X* dan *i.MX6X*. Hingga November 2013 versi terbaru Android setidaknya memerlukan 512MB *RAM* dan *ARMv7* 32-bit, *MIPS* atau prosesor arsitektur x86, bersama dengan *OpenGL ES* 2.0 *graphics processing unit* (GPU) yang kompatibel. Android mendukung *OpenGL ES* 1.1, 2.0 dan 3.0. Beberapa aplikasi secara eksplisit memerlukan versi tertentu dari *Open GL*, oleh karena itu versi *GPU* yang kompatibel sangat diperlukan untuk mejalankan aplikasi ini.

Dari awal muncul hingga kini, perkembangan sistem operasi Android telah menelurkan beberapa versi hingga sekarang seperti yang tampak pada Tabel 2.1.

Tabel 2.1 Versi Android

Versi	Nama Versi	Tanggal Rilis	Level API	Distribusi
4.4	*KitKat*	31 Oktober 2013	19	1.1%
4.3.x	*Jelly Bean*	24 Juli 2013	18	4.2%
4.2.x	*Jelly Bean*	13 November 2012	17	12.9%
4.1.x	*Jelly Bean*	9 Juli 2012	16	37.4%
4.0.3 – 4.0.7	*Ice Cream Sandwich*	16 Desember 2011	15	18.6%
3.2	*Honeycomb*	15 Juli 2011	13	0.1%
2.3.3 – 2.3.7	*Gingerbread*	9 Februari 2011	10	24.1%
2.2	*Froyo*	20 Mei 2010	8	1.6%

2.3.WiFi[6]

Wi-fi / Wifi / WiFi merupakan salah satu bentuk teknologi yang memanfaatkan perangkat elektronik untuk bertukar informasi / data tanpa menggunakan perangkat pengiriman data atau yang sering kita sebut juga dengan komunikasi nirkabel (melalui gelombang radio) di dalam sebuah jaringan komputer. Istilah *WiFi* juga sering disebut sebagai *WLAN (Wireless Local Area Network)* karena secara teknis operasional *WiFi* merupakan salah satu macam teknologi yang bekerja pada jaringan dan perangkat *WLAN*. Dengan kata lain, *WiFi* adalah merk dagang yang dibuat oleh pabrikan untuk perangkat telekomunikasi yang bekerja pada jaringan *WLAN* yang didasarkan kepada standar dan kualitas yang ditetapkan oleh *IEEE (Institute of Elelctrical and Electronics Engineers)*.

Sebuah alat yang mempunyai fasilitas *WiFi* dapat terhubung dengan sumber jaringan seperti internet melalui titik akses atau yang lebih familiar dengan sebutan *hotspot*. Hotspot memiliki jangkauan kira-kira sekitar 20 meter atau sekitar 65 kaki di dalam ruangan dan lebih luas lagi jika diluar ruangan. Namun, *WiFi* juga memungkinkan kita berkomunikasi langsung dari komputer satu dengan yang lainnya tanpa menggunakan hotspot. Ini dapat disebut juga sebagai komunikasi *WiFi ad hoc*.

Gambar 2.2. Logo *WiFi*
(sumber : wikipedia)

Beberapa spesifikasi *WiFi* yang paling familiar yang ditetapkan oleh *IEEE* dalam versi 802.11 pada Tabel 2.2.

Tabel 2.2. Spesifikasi *WiFi*

Spesifikasi	Kecepatan Transfer	Frekuensi Band	Kecocokan dengan versi lain
802.11a	11 Mb/s	~ 2.4 GHz	*b*
802.11b	54 Mb/s	~ 5 GHz	*a*
802.11g	54 Mb/s	~ 2.4 GHz	*b* , *g*
802.11n	100 Mb/s	~ 2.4 GHz	*b* , *g* , *n*

Spesifikasi *b* merupakan produk pertama yang dirilis oleh *WiFi*. Sedangkan produk *g* dan *n* merupakan produk yang mempunyai penjualan terbanyak tahun 2005.

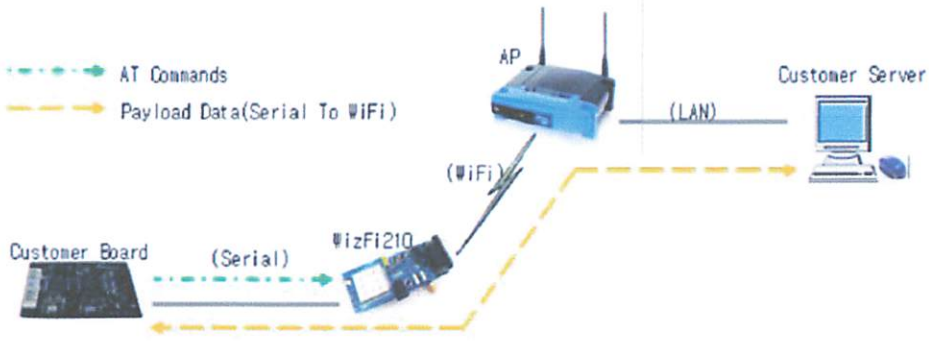
2.4.WizFi210[13]

Dalam penelitian ini penulis menggunakan perangkat *WizFi210* sebagai *Access Point* (pemancar sinyal *WiFi*) sekaligus sebagai konverter protokol *Wireless* TCP ke dalam format protokol Data Serial RS232.

Modul bersertifikasi dari keluarga *WizFi210* menawarkan kemudahan, kecepatan, dan harga yang efektif untuk perangkat dan alat pabrikan untuk menambah kemampuan *WiFi* pada produk mereka. Modul ini menyediakan serial *UART* antarmuka yang memungkinkan koneksi ke setiap desain yang telah menerapkan 8/16/32-bit mikrokontroler melalui perintah sederhana. *WizFi210* mendukung pengiriman data hingga mencapai 11Mbps dan kompatibel dengan 802.11b.

WizFi210 menyediakan kepada pengguna yang rata-rata membutuhkan kemampuan sumber daya *wireless* rendah pada chip dan yang telah ditanamkannya konverter perangkat lunak *Serial-to-WiFi* untuk jaringan *WiFi*. Perangkat lunak *Serial-to-WiFi* memungkinkan perangkat dan produk-produk pabrikan dengan mudah menghubungkan *WiFi* dengan perangkat mereka dengan dampak yang kecil pada program utama mikrokontroler mereka.

WizFi210 menyediakan semua perangkat keras dan lunak yang diperlukan untuk mengatur Serial (*UART*) *Link* berbasis PC atau mirokontroler eksternal. Seluruh data yang terdapat pada *WizFi210* telah diterapkan pada *WizFi220*.



Gambar 2.3.

Diagram Komunikasi *WizFi210* dengan Perangkat Lain

(*sumber : google.com*)

Fitur dan Keuntungan dari *WizFi210* :

- Dapat menghubungkan konektifitas *WiFi* dengan segala perangkat dengan mikrokontroler dan *Serial Host Interface* (*UART*).
- Ukuran kaki komponen yang minimal *Serial-to-Wifi* pada host mikrokontroler dan hanya sedikit merubah pada *firmware MCU* yang telah ada.
- *Offloading host* mikrokontroler lebih sedikit.
- Konfigurasi dan data komunikasi pada *AT Commands* lebih sederhana.
- *DHCP/Static IP, TCP/UDP, Server/Client* dan *DNS.*
- Ukurannya kecil dan memiliki fitur yang cukup lengkap sehingga tidak memerlukan perangkat tambahan seperti *Access Point.*
- Mengkonsumsi daya yang sangat rendah
- Mempunyai beberapa fungsi *interface* seperti *SPI, UART, GPIO, I2C, ADC* dan *JTAG.*
- Mempunyai fitur keamanan *WEP, WPA, WPA2-PSK* dan *Enterprise*

Spesifikasi dari *WizFi210* dapat dilihat pada Tabel 2.3, kemudian spesifikasi dari perangkat kerasnya dapat dilihat pada Tabel 2.4-13.

Tabel 2.3 Spesifikasi *WizFi210*

Spesifikasi	Penjelasan
Radio Protokol	IEEE 802.11b/g/n Compatible
Dukungan Aliran Data	11, 5.5, 2, 1 Mbps (802.11b)
Modulasi	DSSS dan CCK
RF Operating Frequency	2.4 – 2.497 GHz
Opsi Antena	Chip Antena dan U.FL konektor untuk antena tambahan
Protokol Jaringan	UDP, TCP/IP (IPv4), DHCP, ARP, DNS, HTTP/HTTPS Client and Server(*)
Konsumsi Daya	Standby = 34.0 μA(WizFi210) Menerima = 124.0 mA(WizFi210) Mengirim = 126.0 mA(WizFi210)
RF Output Power	8dBm ± 0.5dB (WizFi210) 17dBm ± 0.5dB (WizFi220)
Protokol Keamanan	WEP, WPA/WPA2 – PSK, Entrprise, EAP-FAST, EAP-TLS, EAP-TTLS, PEAP
I/O Antarmuka	UART, SPI(*), I2C(*), WAKE, ALARM, GPIOs
Sumber Daya	3.3V
Ukuran	32 x 23.5 x 3 mm

(*) didukung dengan software yang menyesuaikan

Tabel 2.4 Kondisi Operasi pada *WizFi210*

	Min	Type	Max	Unit
Operating Ambient Range	-40	-	+85	˚C
RTC Supply Voltage	1.2	3.3	3.6	V
Single Supply Voltage	3.0	3.3	3.6	V

Tabel 2.5 Spesifikasi Input Digital pada *WizFi210*

	Min	Type	Max	Unit
IO Supply Voltage	1.62/3	1.8/3.3	1.98/3.63	V
Input Low Voltage (V_{IL})	-0.3	-	$0.25V_{DDIO}$	V
Input High Voltage (V_{IH})	$0.8V_{DDIO}$	-	V_{DDIO}	V
Schmitt trig. Low to High threshold point (V_{T+})	1.5	-	-	V
Schmitt trig. High to Low Threshold point (V_{T-})	-	-	1	V

Tabel 2.6 Spesifikasi Output Digital pada *WizFi210*

	Min	Type	Max	Unit
IO Supply Voltage	1.62/3	1.8/3.3	1.98/3.63	V
Input Low Voltage (V_{IL})	0	-	0.4	V
Input High Voltage (V_{IH})	1.3	-	V_{DDIO}	V
Output rise time (t_{TLH})	-	-	7	V
Output fall time (t_{THL})	-	-	7	V

Tabel 2.7 Spesifikasi Digital I/O (*Tri-State*) pada *WizFi210*

	Min	Type	Max	Unit
IO Supply Voltage	1.62/3	1.8/3.3	1.98/3.63	-
Input Low Voltage (V_{IL})	-0.3	-	$0.25V_{DDIO}$	V
Input High Voltage (V_{IH})	$0.8V_{DDIO}$	-	V_{DDIO}	V
Schmitt trig. Low to High threshold point (V_{T+})	1.5	-	-	V
Schmitt trig. High to Low Threshold point (V_{T-})	-	-	1	V
Pull-Up Resistor (R_u)	0.05	-	1	MΩ
Pull-Down Resistor (R_d)	0.05	-	1	V
Output Low Voltage (V_{OL})	0	-	0.4	V
Output High Voltage (VR_{OH})	1.3	-	V_{DDIO}	ns
Output rise time @3.3V (t_{ToLH})	-	-	7	ns
Output rise time @3.3V (t_{ToHL})	-	-	7	ns
Input rise time (t_{TiLH})	-	-	7	ns
Input fall time (t_{TiHL})	-	-	7	ns

Tabel 2.8 Spesifikasi Input RTC pada *WizFi210*

	Min	Type	Max	Unit
RTC Supply Voltage	1.2	-	3.6	V
Input Low Voltage (V_{IL})	-0.3	-	$0.25V_{DDRTC}$	V
Input High Voltage (V_{IH})	$0.8V_{DDRTC}$	-	V_{DDRTC}	V
Schmitt trig. Low to High threshold point (V_{T+})	$0.57V_{DDRTC}$	-	$0.68V_{DDRTC}$	V
Schmitt trig. High to Low Threshold point (V_{T-})	$0.27V_{DDRTC}$	-	$0.35V_{DDRTC}$	V

Tabel 2.9

Spesifikasi Output RTC pada *WizFi210*

	Min	Type	Max	Unit
RTC Supply Voltage	1.2	-	3.6	V
Input Low Voltage (V_{OL})	0	-	0.4	V
Input High Voltage (V_{OH})	$0.8V_{DDRTC}$	-	V_{DDRTC}	V
Output rise time (t_{TLH})	19	-	142	ns
Output fall time (t_{THL})	21	-	195	ns

Tabel 2.10

Regulator Internal 1.8V pada *WizFi210*

	Min	Type	Max	Unit
Output Voltage (VOUT_1V8)	-	1.8	-	V
Maximum Output Current (IVOUT_V8)	-	250	300	mA
1.8V Regulator Enable "H" Voltage (EN_1V8)	1.0	-	VIN_3V3	V
1.8V Regulator Enable "L" Voltage (EN_1V8)	0	-	0.25	V

Tabel 2.11

Konsumsi Daya (VDD=3.3V, VDDIO=18V, Temp=25°C) pada *WizFi210*

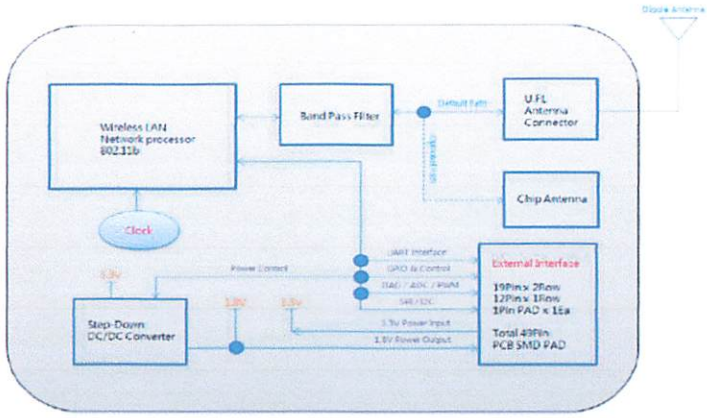
	Current	Power Consumption	Unit
Standby mode (Only V_{DDRTC} is active)	34.0 μA	112.2	μW
Idle mode (CPU running, WLAN disconnected)	10.0 mA	30.0	mW
Receive (-81dBm RX sens. @11Mbps)	124.0 mA	409.2	mW
Transmit (+8dBm at antenna port @11Mbps)	126.0 mA	415.8	mW
TCP, Standby = 10sec, Payload 100Byte	3.7 mA	12.21	mW
TCP, Standby = 1min, Payload 100Byte	0.6 mA	1.98	mW
UDP, Standby = 10sec, Payload 100Byte	3.6 mA	11.88	mW
UDP, Standby = 1min, Payload 100Byte	0.6 mA	0.98	mW

Tabel 2.12. Spesifikasi RF pada *WizFi210*

Spesifikasi	Penjelasan
Modulation Technique	DSSS for 1, 2Mbps CCK for 5.5, 11Mbps
Data Rate	IEEE 802.11b : 1, 2, 5.5 and 11Mbps
Receive Sensitivity	-84dBm ± 1dB @ 11Mbps -88dBm ± 1dB @ 5.5Mbps -90dBm ± 1dB @ 2Mbps -94dBm ± 1dB @ 1Mbps
Transmit Power (average)	802.11b : 8dBm ± 0.5dB @ 11Mbps
Frequency Range	USA : 2.400 ~ 2.483GHz Europe : 2.400 ~ 2.483GHz Japan : 2.400 ~ 2.497GHz China : 2.400 ~ 2.483GHz
Operating Channels	USA/Canada : 11 (1~11) Major EUROPE Countries : 13 (1~13) France : 4 (10~13) Japan : 14 for IEEE 802.11b (1~13 or 14) 13 for IEEE 802.11g (1~13) Korea/China : 13 (1~13)
Antenna	U.FL External Antenna Support External Antenna Pad Support 1dBi Chip Antenna (Optional)

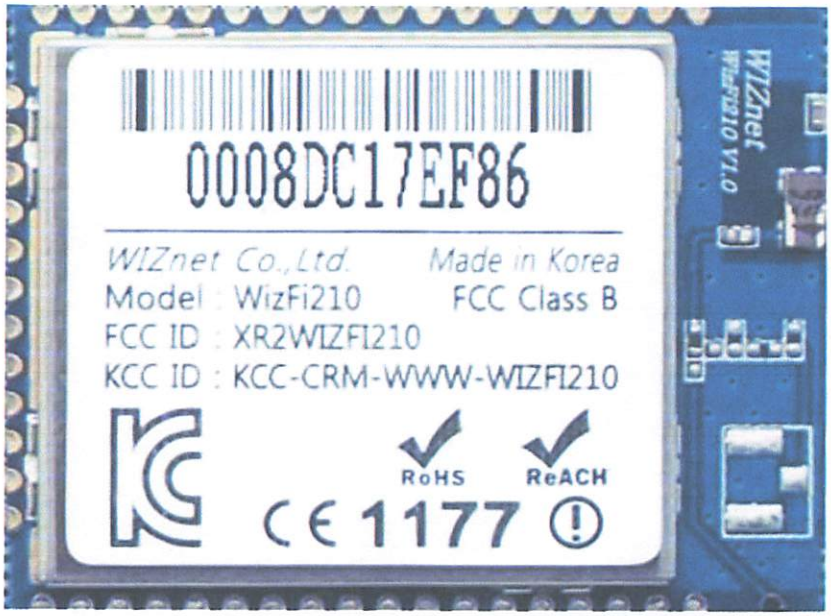
Tabel 2.13. LED Indikator pada *WizFi210*

Condition	D3(PWR)	D1(GPIO30)	D2(GPIO31)	D4(GPIO28)
ON Solid	Power On	-	Serial-to-WiFi OK	Associated
Blink(-1-)	-	Serial Data RX (Data Mode)	-	-
Blink(-1-1-)	-	Serial Data RX (AT Command Mode)	-	-
OFF	No Power	-	Serial-to-WiFi Error	Non Associated

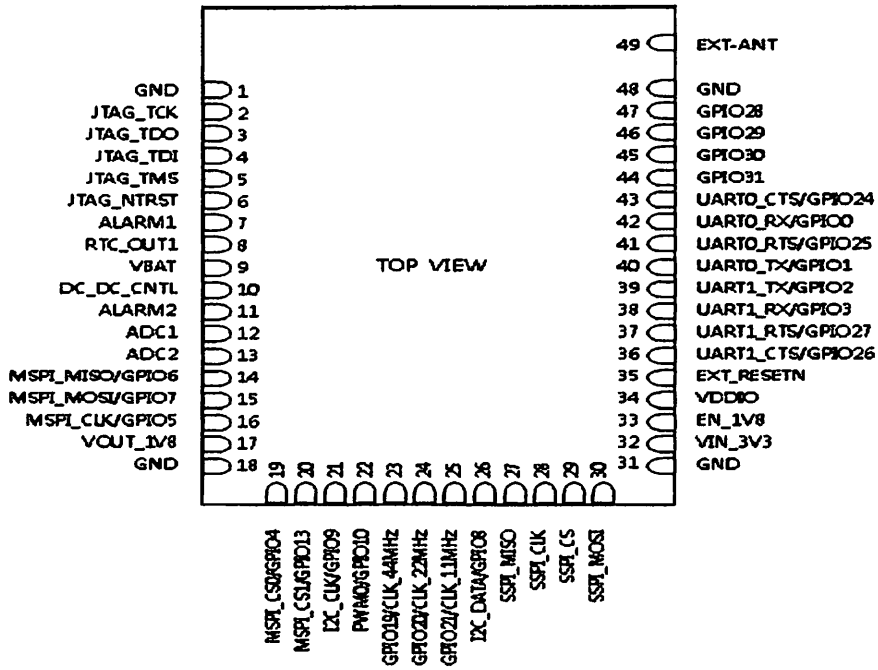


Gambar 2.4. Blog Diagram *WizFi210*[13]

Pada Gambar 2.5, terlihat bahwa *WizFi210* banyak terdapat pin-pin yang mempunyai fungsi yang berbeda yang dapat digunakan sesuai kebutuhan pengembang perangkat. Penjelasan masing-masing fungsi pin dapat dilihat pada Gambar 2.6.



Gambar 2.5. *WizFi210*[13]

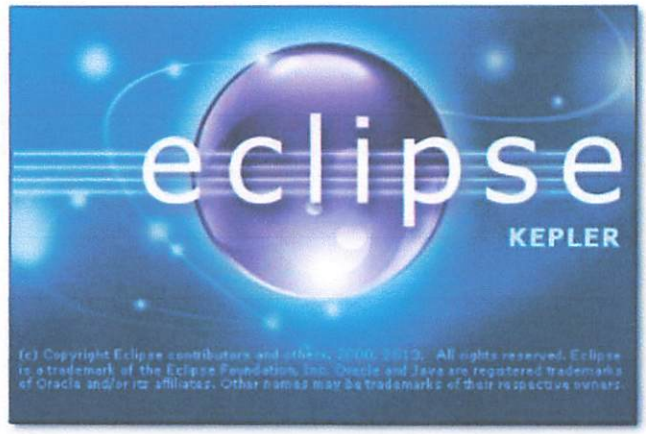


Gambar 2.6. Deskripsi PIN pada *WizFi210*[13]

2.5. Eclipse[8]

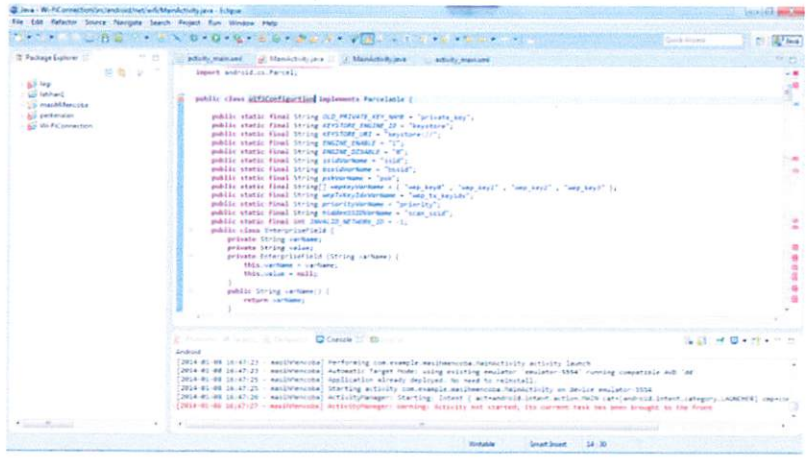
Eclipse adalah sebuah *IDE* (*integrated Development* Environment) untuk mengembangkan perangkat lunak dan dapat dijalankan di semua platform (*platform-independent*). Berikut ini adalah sifat dari Eclipse :

- Multi-platform : Target sistem operasi *Eclipse* adalah Microsoft Windows, Linux, Solaris, AIX, HP-UX dan Mac OS X.
- Mulit-language : *Eclipse* dikembangkan dengan bahasa pemrograman Java, akan tetapi Eclipse mendukung pengembangan aplikasi berbasis bahasa pemrograman lainnya, seperti *C/C++, Cobol, Python, Perl, PHP*, dan lain sebagainya.
- Multi-role : Selain sebagai *IDE* untuk pengembangan aplikasi, *Eclipse* pun bisa digunakan untuk aktivitas dalam siklus pengembangan perangkat lunak, seperti dokumentasi, test perangkat lunak, pengembangan web, dan lain sebagainya.



Gambar 2.7. Logo Eclipse Kepler[8]

Eclipse pada saat ini merupakan salah satu *IDE* terfavorit dikarenakan gratis dan *open source*, yang berarti setiap orang boleh melihat kode pemrograman perangkat lunak ini. Selain itu, kelebihan dari *Eclipse* yang membuatnya populer adalah kemampuannya untuk dapat dikembangkan oleh pengguna dengan komponen yang dinamakan *plug-in*.[10]



Gambar 2.8. Contoh tampilan editor pada Eclipse Kepler

2.6. Sensor *MQ-6* (pendeteksi gas)[2]

Bahan material sensitif yang terdapat pada sensor *MQ-6* adalah SnO_2 dengan tingkat konduktifitas yang rendah di udara bersih. Jika gas yang mudah terbakar muncul, konduktifitas sensor menjadi lebih tinggi seiring dengan meningkatnya konsentrasi gas. *MQ-6* memiliki sensitifitas yang tinggi

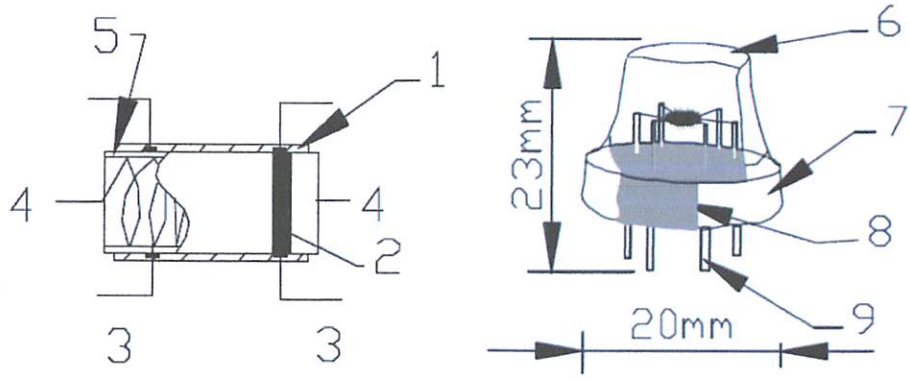
terhadap gas *Propane*, *Butane* dan *LPG*. Sensor ini dapat pula mendeteksi karakter gas yang berbeda seperti gas Metana, harga sensor ini cukup terjangkau dan cocok untuk aplikasi yang berbeda.

Karakter sensor MQ-6 :

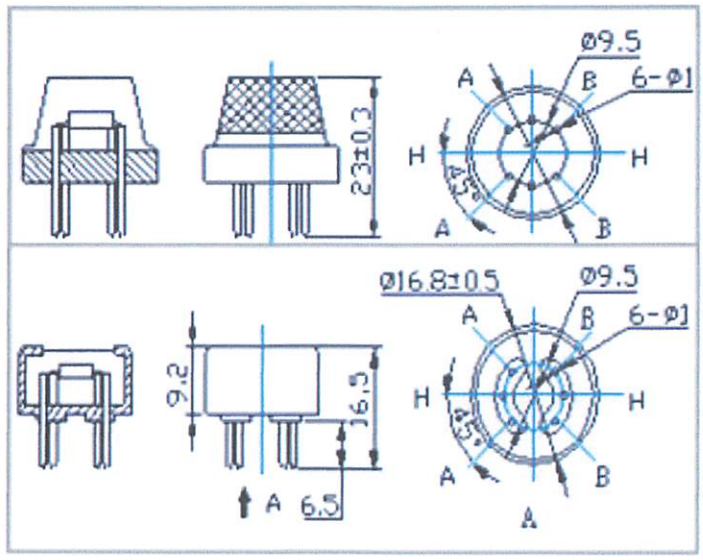
- Memiliki sensitifias yang baik terhadap gas yang mudah terbakar dalam area yang meluas.
- Sangat sensitif terhadap gas *Propane*, *Butane* dan *LPG*.
- Murah dan tahan lama.
- Rangkaian driver yang sederhana.

Tabel 2.14. Penjelasan Struktur Sensor *MQ-6* pada Gambar 2.9.

No.	Parts	Materials
1	Gas sensing layer	SnO_2
2	Electrode	Au
3	Electrode line	Pt
4	Heater coil	Ni-Cr alloy
5	Tubular ceramics	Al_2O_3
6	Anti-explotion network	Stainless steel gauze (SUS316 100-mesh)
7	Clamp ring	Copper plating Ni
8	Resin base	Bakelite
9	Tube pin	Copper plating Ni



Gambar 2.9. Struktur dan Konfigurasi Sensor *MQ-6*



Gambar 2.10. Parameter elektrik pengukuran rangkaian *MQ-6*

Struktur dan konfigurasi dari sensor *MQ-6* tertera pada Gambar 2.10, sensor tersusun oleh tabung keramik mikro (AL_2O_3), lapisan sensitif *Tin dioxide* (SnO_2), pengukur elektroda dan pemanas dipadukan menjadi seperti kerak yang terbuat dari plastik dan baja anti karat. Pemanas menyediakan kondisi kerja yang diperlukan bagi komponen sensitif. *MQ-6* memiliki 6 pin, 4 pin digunakan untuk mengambil dan menghantarkan sinyal sedangkan yang lain bekerja untuk mengalirkan panas.

Untuk mengetahui lebih jelas spesifikasi dari sensor gas *MQ-6*, lihat Tabel 2.15-17.

Tabel 2.15. Kondisi Kerja Standar Sensor *MQ-6*

Symbol	Parameter Name	Technical Condition	Remarks
V_c	Circuit Voltage	5V ± 0.1	AC or DC
V_H	Heating Voltage	5V ± 0.1	AC or DC
P_L	Load Resistance	20KΩ	
R_H	Heater Resistance	33Ω ± 5%	Room Tem
P_H	Heating Consumption	less than 750mW	

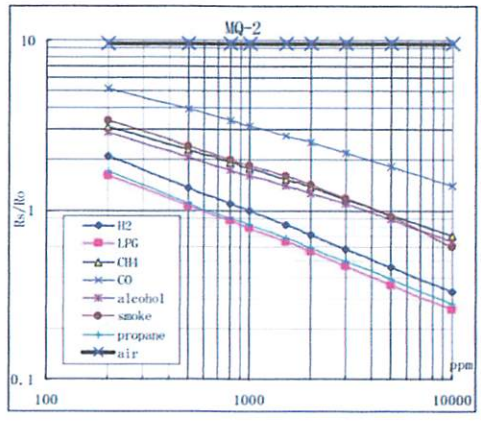
Tabel 2.16. Kepekaan Sensor *MQ-6* Terhadap Lingkungan

Symbol	Parameter Name	Technical Condition	Remarks
Tao	Using Tem	-10°C ~ 50°C	
Tas	Storage Tem	-20°C ~ 70°C	
R_H	Related Humidity	less than 95% Rh	
O_2	Oxygen Concentration	21%(standard condition) Oxygen concentration can effect sensitifity	Minimum value is over 2%

Tabel 2.17. Karakter Sensitifitas Sensor *MQ-6*

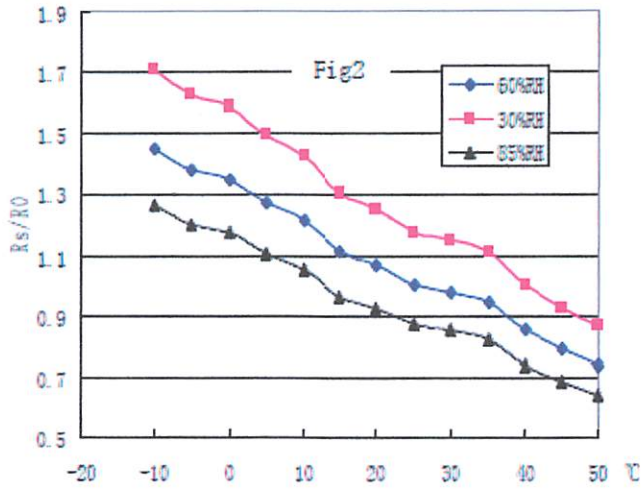
Symbol	Parameter Name	Technical Condition	Remarks
R_S	Sensing Resistance	10KΩ – 60KΩ (1000ppm LPG)	Detecting concentration scope : • 200-10000ppm LPG • ISO-Butane • Propane • LNG
α (1000/4000ppm LPG)	Concentration Slope Rate	≤ 0.6	
Standard Detecting Condition	Temp : 20°C ± 2°C Vc : 5V ± 0.1 Humidity : 65% ± 5% Vh : 5V ± 0.1		
Preheat Time	over 24 hour		

Nilai resistansi dari sensor *MQ-6* adalah untuk membedakan berbagai jenis dan konsetrasi gas. Jadi bila menggunakan komponen ini, penyesuaian sensitifitas sangatlah dibutuhkan. Disarankan bagi pengguna perangkat untuk menyesuaikan detektor, 1000ppm untuk konsentrasi nilai udara gas LPG dan penggunaan beban sekitar 20KΩ (10KΩ menjadi 47KΩ) untuk resistansinya (RL).



Grafik 2.1.

Karakter Sensitifitas terhadap jenis-jenis gas pada Sensor *MQ-6*



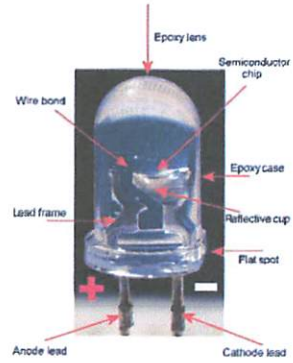
Grafik 2.2.

Karakteristik Sensor *MQ-6* terhadap temperatur dan kelembaban udara

2.7.Sensor *Fotodioda* dan *LED*[3]

Sensor yang penulis gunakan untuk mendeteksi gerak adalah *Photodioda* dan *LED*. Salah satu pertimbangan memilih sensor ini daripada menggunakan sensor-sensor yang lain adalah karena prinsip kerjanya yang simpel dan harga yang sangat terjangkau serta kemampuannya yang juga dapat bersaing dengan sensor-sensor gerak yang lain.

LED merupakan singkatan dari *Light Emiting Dioda*, merupakan produk temuan lain setelah dioda yang dapat memancarkan sinyal bila dibias maju. Gejala ini termasuk bentuk *electroluminescense*. Seperti sebuah dioda normal, *LED* terdiri dari sebuah bahan chip semikondukor yang di-dop dengan ketidakmurnian untuk menciptakan struktur yang disebut *p-n junction*.



Gambar 2.11. Lampu *LED*

LED adalah alat aplikatif dari cahaya tampak yang bersifat monokromatik. Cara keja dari alat ini adalah dengan merubah elektron menjadi foton. Elektron yang dialiri oleh sumber tegangan (panjar maju) akan mengalami medan elektromagnetik hingga menimbulkan arus listrik. Arus listrik ini yang kemudian akan menghidupkan dioda (*LED*) hingga foton dalam *LED* akan memancarkan energi dalam bentuk cahaya *LED*. Dalam *LED* dapat dipadang sebagai sebuah kristal. Kristal ini terdiri dari lubang (*hole*) dan elektron (*ion*), setiap elektron akan mengisi lubang yang kosong dalam rekombinasi ini disebabkan oleh hantaran arus listrik dari sumber tegangan (panjaru maju). Ketika elektron telah berkombinasi dengan lubang tadi, menyebabkan elektron terlepas dari energi ikatnya. Rekombinasi ini menghasilkan energi yang terlepas dari elektron. Energi yang terlepas inilah yang digunakan untuk memancarkan foton (rekombinasi radiaktif), sebagian lain digunakan untuk memanaskan partikel-partikel kristal (rekombinasi non-radiaktif). Pancaran cahaya ini merupakan cahaya sebuah *LED*.

Beberapa karakteristik dari *LED*, antara lain :

- Warna (panjang gelombang) ditentukan oleh *band-gap*
- Intensitas cahaya hasil berbanding lurus dengan arus
- Non linearitas tampak pada arus rendah dan tinggi

Warna-warna yang dipancarkan oleh sebuah *LED* tergantung dari selisih pita energi dari bahan yang membentuk *p-n junction*. Sebuah dioda normal biasanya terbuat dari silikon atau *germanium*, memancarkan cahaya tampak inframerah dekat. Tetapi, bahan yang digunakan untuk sebuah *LED* memiliki selisih pita energi antara cahaya inframerah dekat, tampak dan ultra ungu dekat. *LED* konvensional terbuat dari bahan mineral organik yang bervariasi, menghasilkan warna yang dijelaskan pada tabel 2.18.

Tabel 2.18. Bahan *LED* dan warna yang dihasilkan

Bahan	Warna
Alluminium Gallium Arsenide (AlGaAs)	Merah dan inframerah
Gallium Aluminium Phosphide (GaAlP)	Hijau
Gallium Arsenide / Phosphide (GaAsP)	Merah, oranye-merah, oranye dan kuning
Gallium Nitride (GaN)	Hijau, hijau murni (hijau emerald) dan biru
Gallium Phosphide (GaP)	Merah, kuning dan hijau
Zinc Selenide(ZnSe)	Biru
Indium Gallium Nitride (InGaN)	Hijau kebiruan dan biru
Indium Gallium Aluminium Phosphide (InGaAlP)	Oranye-merah, oranye, kuning dan hijau
Diamond (C)	Ultraviolet
Sapphire (Al_2O_3) as substrate	Biru

Inframerah radiasi adalah radiasi elektromagnetik dengan panjang gelombang yang lebih panjang dibandingkan dengan cahaya yang tampak, tetapi lebih pendek dari gelombang mikro. Cahaya inframerah mempunyai panjang gelombang sekitar 750nm sampai 1mm. *LED* yang memancarkan cahaya inframerah (~950nm) biasanya digunakan sebagai sumber cahaya pada sistem sensor, sedangkan *LED* cahaya tampak dipakai sebagai indikator, peraga dalam instrumen digital dan lain-lain.

Fotodioda, merupakan jenis dioda yang mendeteksi cahaya. *Fotodioda* merupakan sensor cahaya semikonduktor yang dapat mengubah besaran cahaya menjadi besaran listrik dan juga merupakan sebuah dioda dengan sambungan *p-n* yang dipengaruhi cahaya dalam kerjanya. Cahaya yang dapat dideteksi oleh *fotodioda* ini adalah mulai dari inframerah , cahaya tampak, ultra ungu sampai dengan sinar-X. Aplikasi *fotodioda* mulai dari penghitungan kendaraan di jalan umum hingga aplikasi yang digunakan untuk bidang medis.

Alat yang juga mempunyai kesamaan dengan *fotodioda* adalah *fototransistor* (*Phototransistor*). Komponen ini mempunyai sensitivitas yang lebih baik jika dibandingkan dengan *fotodioda*. Prinsip kerja dari *fotodioda*, jika sebuah sambungan *p-n* dibias maju dan diberikan cahaya padanya maka pertambahan arus sangat kecil, sedangkan jika sambungan *p-n* dibias mundur maka arus akan bertambah cukup besar. Ketika elektron-elektron yang dihasilkan itu masuk ke pita produksi, maka elektron-elektron itu akan mengalir ke arah positif sumber tegangan, sedangkan *hole* yang dihasilkan mengalir ke arah negatif sumber tegangan sehingga arus akan mengalir ke

dalam rangkaian. Besarnya pasangan elektron maupun *hole* yang dihasilkan tergantung dari besarnya intensitas cahaya yang dikenakan pada *fotodioda*. Cahaya yang dikenakan pada *fotodioda* akan mengakibatkan terjadinya pergeseran foton yang akan menghasilkan pasangan *electron-hole* di kedua sisi dari sambungan.

Photodioda merupakan suatu jenis dioda yang resistansinya berubah-ubah kalau cahaya yang jatuh pada dioda berubah-ubah intensitasnya. Dalam gelap, nilai tahanannya sangat besar hingga praktis tidak ada arus yang mengalir. Semakin kuat cahaya yang jatuh pada dioda, maka tahanannya semakin kecil hingga arus yang mengalir menjadi semakin besar. Perbandingan arus pada saat dikenai cahaya dengan pada saat tidak dikenai cahaya ternyata cukup besar. Karakteristik ini diperlukan sebagai transducer cahaya. Umumnya *fotodioda* terbuat dari silikon dengan waktu reaksi ~1ns. Selanjutnya *fotodioda* juga dipergunakan untuk mengkonversi energi solar menjadi energi listrik. Karakteristik utama *fotodioda* adalah sebagai berikut :

- Tanggapan spektral (dinyatakan dalam A/lumen atau dengan %), untuk *fotodioda silikon* tanggapan maksimum pada panjang gelombang sekitar 800nm.
- Arus gelap adalah arus mundur *fotodioda* pada saat tak ada cahaya. Arus gelap ini bergantung pada suhu, biasanya arus gelap ini cukup besar dibandingkan dengan dioda hubungan (arus mundur) dalam orde nA atau μA tergantung pada luas permukaan perangkat.
- Efisiensi kuantum yaitu perbandingan jumlah pasangan *hole-elektron* yang terjadi secara optis dengan jumlah foton datang. Efisiensi ini lebih besar dari 90% pada panjang gelombang puncak.

Pada *fotodioda semikonduktor*, jika *p-n junction* dengan bias mundur disinari, akan terjadi perubahan arus yang hampir linier terhadap *flux* cahaya. Gejala ini dimanfaatkan pada *fotodioda semikonduktor*. Komponen ini terdiri atas *p-n junction* yang dibuat dalam plastik transparan. Radiasi hanya bisa diberikan pada satu permukaan *junction*. Sisi yang lain biasanya di cat hitam atau ditutupi lempengan logam. Komponen ini sangat kecil dengan order ukuran sepersepuluh inci. Jika *fotodioda* mendapat tegangan balik dengan nilai

sepersepuluhan volt, akan terjadi arus yang hampir konstan (tidak tergantung pada besarnya bias mundur). Arus "gelap" (*dark current*) berhubungan dengan arus saturasi mundur, karena pembentukan *carrier* minoritas secara termal. Jika cahaya dijatuhkan pada permukaan, akan terbentuk pasangan *carrier* yang kemudian akan berdifusi ke *junction* dan menyebrangi *junction* sehingga menimbulkan arus.

Arus saturasi mundur I_0 pada dioda *p-n* proporsional terhadap konsentrasi *carrier* minoritas pn_0 dan nn_o. Jika *junction* disinari, muncul sebuah *hole-electron* baru, proporsional dengan jumlah foton. Dengan demikian, bias mundur yang besar akan terbentuk arus $I = I_0 + I_s$, denga I_s sebagai arus *short-circuit* yang proporsional terhadap intensitas cahaya. Dengan demikian, karakteristik *volt-ampere fotodioda semikonduktor* adalah :

$$I = I_0 + I_s(1 - e^{v/vt\eta})$$

Nilai *V* positif untuk tegangan maju dan negatif untuk bias mundur. Parameter 'η' bernilai satu untuk germanium dan dua untuk silikon. *Vt* adalah tegangan ekuivalen untuk suhu.

Arus pada *fotodioda semikonduktor* terbias mundur bergantung pada difusi *carrier* minoritas di *junction*. Jika radiasi difokuskan pada satu titik kecil yang jauh dari *junction*, *carrier* minoritas terinjeksi bias melakukan rekombinasi sebelum berdifusi pada *junction*. Dengan demikian, arus yang mengalir menjadi lebih kecil dibandingkan kalau peristiwa ini pada posisi yang lebih dekat dengan *junction*.

Prinsip kerja kedua alat ini cukup sederhana, *LED* akan memancarkan cahaya ke objek dan *photodioda* akan menerima cahaya yang dipantulkan oleh objek tersebut. Intensitas cahaya yang diterima oleh *photodioda* akan mempengaruhi nilai resistansinya. Objek berupa warna biru dan merah akan memantulkan cahaya dengan intensitas yang berbeda. Warna merah akan memantulkan cahaya dengan intensitas yang lebih tinggi daripada warna hijau, sehingga nilai resistansinya akan berbeda. Semakin besar intensitas cahaya yang diterima oleh *photodioda*, maka nilai 15 resistansinya akan semakin kecil dan nilai tegangan *output*-nya akan semakin kecil pula. Perbedaan nilai tegangan *output* dari *photodioda* saat menerima cahaya pantulan dari warna

sepenuhnya yang akan terjadi jika hanya kontak tidak terputus pada beberapa titik tunggal. Arus "gelap" (dark current) berkurang dengan arus sebesar nilai arus penyearah ideal pada tegangan balik yang sama. Kondisi pengisian seperti ini dapat dijelaskan dengan pembentukan daerah deplesi yang kosong pembawa muatan bergerak di dalam medan listrik yang diterapkan.

Arus searah minoritas I_0 pada dioda p-n proporsional terhadap konsentrasi carrier minoritas p_{n0} dan n_{p0}. Jika panjang difusi minoritas lebih besar dari lebar daerah deplesi W, maka I_0 tergantung pada W. Dengan demikian, karakteristik volt-ampere dari sambungan adalah :

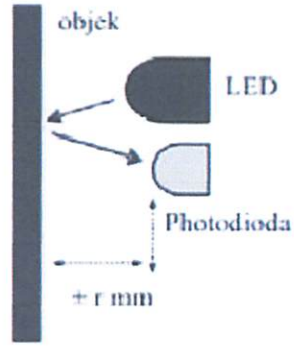
$$I = I_0 + I_G(1 - e^{qV/kT})$$

Nilai I_G positif untuk tegangan maju dan negatif untuk bias mundur. Parameter "q" bernilai satu untuk germanium dan dua untuk silikon. k adalah tegangan ekivalen suhu mutlak.

Arus pada frekuensi rendah terdiri atas arus mundur bergantung pada difusi carrier minoritas di luar area deplesi dan pada arus hasil kopling yang lebih dari pasangan carrier minoritas terpisah oleh medan. Jika kombinasi sebelum berdifusi ke luar area deplesi dapat diabaikan, arus yang mengalir merupakan jumlah lebih kecil dibandingkan kalau pembawa muatan positif yang lebih dulu dilepaskan.

Pada kecepatan tinggi ini cukup sederhana. LED akan memancarkan cahaya ke sekitar dan pada permukaan akan memantul cahaya yang dipancarkan oleh sambungan. Intensitas cahaya yang dipancarkan akan mempengaruhi nilai resistansi. Pada beban ringan atau hubung singkat akan memancarkan cahaya dengan intensitas yang lebih tinggi daripada saat sebelum nilai resistansinya akan berbeda. Semakin besar intensitas cahaya yang diterima oleh photodioda, maka nilai IS resistansinya akan semakin kecil dan nilai tegangan output-nya akan semakin kecil pula. Perbedaan nilai tegangan output dari photodioda saat memancarkan cahaya pantulan dan waktu

merah dan warna hijau akan dideteksi oleh rangkaian komparator. Tegangan referensi dapat diatur dengan memutar variabel resistor. Untuk dapat membedakan warna merah dan warna hijau, nilai tegangan referensi diatur sehingga memiliki nilai diantara nilai tegangan output dari *photodioda* saat menerima pantulan cahaya dari objek. Untuk mendapatkan hasil yang baik maka pemasangan sensor warna harus tertutup dan dipasang tegak lurus terhadap objek seperti yang ditunjukkan oleh gambar 2.12.

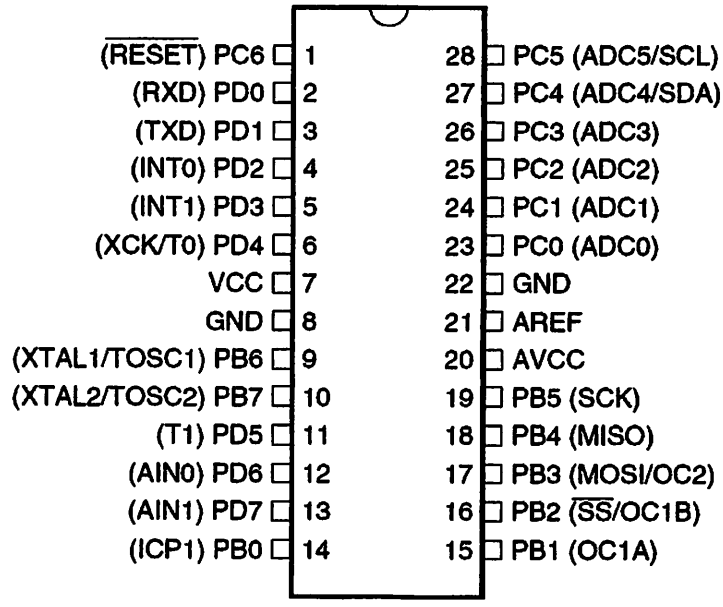


Gambar 2.12. Pemasangan sensor *fotodioda* dan LED yang benar

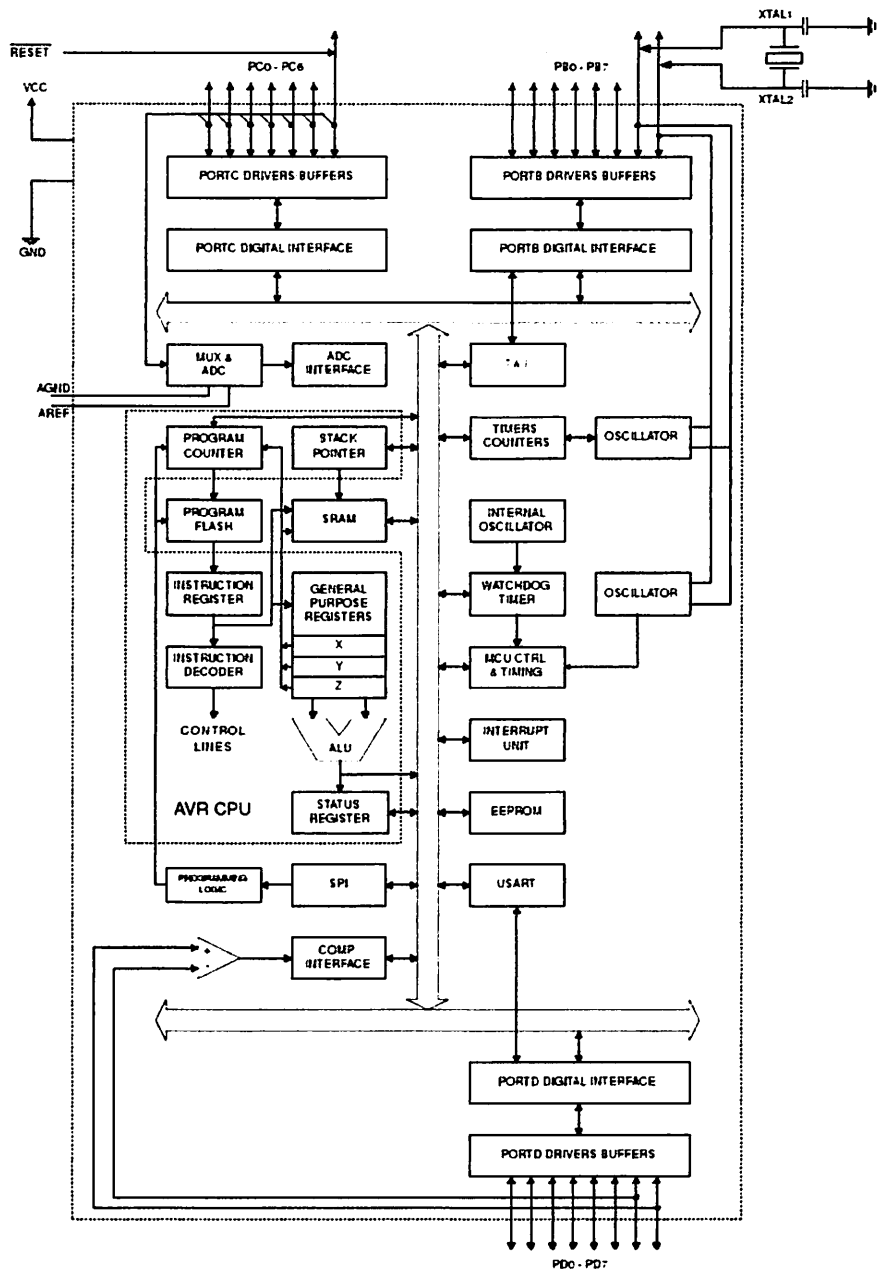
Untuk mendeteksi warna merah akan digunakan sensor *photodioda* yang disinari oleh *LED* berwarna merah. Pada saat *photodioda* menerima pantulan cahaya dari warna merah, nilai tegangan output pada *photodioda* akan lebih kecil dari tegangan referensi, sehingga output dari komparator akan bernilai "1". Sebaliknya, untuk mendeteksi warna hijau maka digunakan sensor *photodioda* yang disinari *LED* warna hijau. Pada saat *photodioda* menerima pantulan cahaya dari warna hijau, nilai tegangan output pada *photodioda* akan lebih kecil dari tegangan referensi sehingga output dari komparator akan bernilai "0". Sedangakan saat *photodioda* menerima pantulan cahaya dari warna merah, nilai tegangan outputnya akan lebih besar dari tegangan referensi, sehingga output dari komparator bernilai "1".

2.8.Mikrokontroler ATMega8[4]

ATMega8 adalah *low-power CMOS* 8-bit mikrokontroler yang berdasarkan pada arsitektur *AVR RISC*. Dengan mengeksekusi instruksi yang kuat dalam satu siklus *single clock*, *ATMega8* dapat menerima hingga 1MIPS per MHz yang memungkinkan perancang sistem untuk mengoptimalkan daya daripada mementingkan kecepatan pemrosesan.



Gambar 2.13. Konfigurasi Pin PDIP ATMega8



Gambar 2.14. Diagram Blok ATMega8

Atmel AVR CPU Core

AVR core (inti) menggabungkan banyak instruksi dan di set dengan 32 *general purpose* yang telah ter-registrasi. Seluruh 32 *general purpose* ini secara langsung terhubung ke *Arithmetic Logic Unit* (ALU), yang memungkinkan dua *register* yang berbeda dapat di akses sekaligus dalam satu instruksi tunggal dan dieksekusi dalam hitungan satu *clock.*

ATMega 8 menyediakan beberapa fitur :

- 8Kb In-*System Programmable Flash* dengan kemampuan *Rad-While-Write*.
- 512byte EEPROM.
- 1Kb SRAM.
- 23 *general purpose* I/O lines.
- 32 *general purpose working registers*.
- 3 fleksibel *Timer/Counter* dengan mode pembanding.
- Interupsi Internal dan Eksternal.
- Sebuah serial *USART* yang dapat diprogram.
- Sebuah ADC 6-*channel* (delapan saluran dalam paket TQFP dan QFN/MLF) dengan akurasi 10bit.
- *Programmable Watchdog Timer* dengan Osilator internal.
- 5 mode *software* penghemat daya yang dapat dipilih.

Ketika kondisi *Idle mode CPU* dihentikan tapi tetap membiarkan *SRAM, TIMER/COUNTER*, port *SPI* dan *interrupt system* untuk terus berfungsi. *Powerdown* mode menyimpan isi dari register tapi membekukan Osilator, memutuskan semua fungsi *chip* yang lain sampai *interrupt* berikutnya atau reset *hardware*. Dalam *Power-save* mode, *timer asynchronous* terus berjalan yang memungkinkan pengguna dapat mempertahankan *timer base*, sementara pada saat perangkat yang lain dalam keadaan tidak menyala. Mode ADC *Noise Reduction* menghentikan CPU dan semua modul I/O kecuali *asynchronous* timer dan ADC, untuk meminimalisir *switching noise* selama konversi ADC sedang berlangsung. Dalam mode *Standby*, Osilator *Crystal/Resonator* berjalan sementara perangkat lainnya dalam posisi non-aktif. Hal ini yang dapat memungkinkan *start-up* yang sangat cepat dan menggunakan daya yang rendah.

Perangkat ini diproduksi dengan teknologi Atmel yang mana memori mempunyai kepadatan yang tinggi. Memori program Flash dapat memprogram ulang *In-System* melalui serial antarmuka SPI. *Atmel ATMega8* adalah mikrokontroler canggih sehingga dapat memberikan solusi yang sangat fleksibel dan biaya yang efektif untuk banyak aplikasi *embedded* kontrol.

Tabel 2.19. Deskripsi Pin Atmel ATMega8

Nama	Deskripsi
Vcc	Tegangan suplai digital
GND	Ground
Port B (PB7, PB0) XTAL1/XTAL2/ TOSC1/TOSC2	Port B adalah 8-bit bi-direksional port I/O dengan internal pull-up resistor (yang dipilih dari tiap bit). Port B output buffer memiliki karakteristik drive yang simetris dengan kemampuan sink dan source yang tinggi. Sepagai inputan, pin Port B memiliki sumber arus yang rendah apabila pull-up resistor ditarik secara eksternal. Status Pin di Port B menjadi tri-stated ketika kondisi reset aktif, bahkan jika clock tidak berjalan.
Port C (PC5..PC0)	Port C adalah 7-bit bi-direksional port I/O dengan internal pull-up resistor (yang dipilih dari tiap bit). Port C output buffer memiliki karakteristik drive yang simetris dengan kemampuan sink dan source yang tinggi. Sepagai inputan, pin Port C memiliki sumber arus yang rendah apabila pull-up resistor ditarik secara eksternal. Status Pin di Port C menjadi tri-stated ketika kondisi reset aktif, bahkan jika clock tidak berjalan.
PC6/RESET	Jika RSTDISBL diprogram, PC6 digunakan sebagai pin I/O. Perlu diperhatikan bahwa karakteristik elektrik PC6 berbeda dari pin Port C yang lainnya. Jika RSTDISBL tidak terprogram, PC6 digunakan sebagai Reset input. Jika Low level pada pin ini lebih panjang dari panjang minimum pulsa, maka pin akan melakukan Reset. Bahkan jika clock tidak pada keadaan sedang berjalan. Pulsa terpendek tidak dapat menjamin untuk dapat melakukan Reset.
Port D (PD7..PD0)	Port D adalah 7-bit bi-direksional port I/O dengan internal pull-up resistor (yang dipilih dari tiap bit). Port D output buffer memiliki karakteristik drive yang simetris dengan kemampuan sink dan source yang tinggi. Sepagai inputan, pin Port D memiliki sumber arus yang rendah apabila pull-up resistor ditarik secara eksternal. Status Pin di Port D menjadi tri-stated ketika kondisi reset aktif, bahkan jika clock tidak berjalan.
RESET	Inputan Reset. Jika Low level pada pin ini lebih panjang dari panjang minimum pulsa, maka pin akan melakukan Reset. Bahkan jika clock tidak pada keadaan sedang berjalan. Pulsa terpendek tidak dapat menjamin untuk dapat melakukan Reset.
AV_{CC}	AV_{CC} adalah pin suplai tegangan untuk A/D konverter, Port C (3 .. 0) dan ADC (7 .. 6). Perlu secara eksternal untuk dapat tehubung dengan V_{CC}, bahkan jika ADC tidak digunanakan. Jika ADC digunakan, harus dihubungkan ke V_{CC} dengan low-pass filter. Perlu digaris bawahi bahwa di Port C (5 .. 4) menggunakan suplai tegangan digital, V_{CC}.
AREF	AREF adalah referensi pin analog untuk A/D konverter.
ADC7..6 (TQFP and QFN/MLF Package only)	Dalam paket TQFP dan QFN/MLF, ADC(7..6) dapat digunakan sebagai inputan analog ke A/D konverter. Pin-pin ini didukung dari pasokan analog dan berfungsi sebagai saluran ADC 10-bit

Arithmetic Logic Unit (ALU)

Atmel *ALU* yang memiliki *high-perfomance* beroperasi dalam hubungan langsung dengan semua 32 register *general purpose*. Dalam satu siklus *clock* tunggal, operasi aritmatika antar *general purpose* atau *register* dieksekusi dengan segera. Operasi ALU terbagi menjadi tiga kategori utama yaitu aritmatika, logika dan bit-fungsi. Beberapa implementasi dati arsitektur juga menyediakan *powerful-multiplier* yang mendukung keduanya untuk di-*signed/unsigned* dengan perkalian dan bilangan pecahan.

Status register berisi informasi tentang hasil dari instruksi aritmatika yang terakhir dieksekusi. Informasi ini dapat digunakan untuk merubah aliran program untuk melakukan operasi yang sifatnya kondisional. Perlu

Tabel 2.1 Deskripsi Pin Atmega 8535

Nama	Deskripsi
Vcc	Tegangan suplai digital
GND	Ground
Port B (PB7..PB0) XTAL1/TOSC1 XTAL2/TOSC2	Port B adalah port I/O 8-bit dengan internal pull-up resistor (yang dipilih untuk masing-masing bit). Output buffer Port B memiliki karakteristik simetris dengan kemampuan sink dan source yang tinggi. Sebagai input, pin Port B yang secara eksternal ditarik rendah akan mengalir arus jika pull-up resistor diaktifkan. Pin Port B adalah pin tri-state ketika kondisi reset menjadi aktif, bahkan jika clock tidak berjalan.
Port C (PC5..PC0)	Port C adalah 7-bit bidirectional port I/O dengan internal pull-up resistor (dipilih untuk masing-masing bit). Output buffer Port C memiliki karakteristik simetris dengan kemampuan sink dan source yang tinggi. Sebagai input, pin Port C yang secara eksternal ditarik rendah akan mengalir arus jika pull-up resistor diaktifkan. Pin Port C adalah pin tri-state ketika kondisi reset menjadi aktif, bahkan jika clock tidak berjalan.
Port D (PD7..PD0)	Port D adalah 8-bit bidirectional port I/O dengan internal pull-up resistor (dipilih untuk masing-masing bit). Output buffer Port D memiliki karakteristik simetris dengan kemampuan sink dan source yang tinggi. Sebagai input, pin Port D yang secara eksternal ditarik rendah akan mengalir arus jika pull-up resistor diaktifkan. Pin Port D adalah pin tri-state ketika kondisi reset menjadi aktif, bahkan jika clock tidak berjalan.
RESET	Reset input. Tegangan pada pin ini selama lebih dari panjang minimum pulse akan menyebabkan reset, bahkan jika clock tidak berjalan. Nilai lebih pendek tidak dijamin untuk menghasilkan reset.
AVcc	AVcc adalah pin supply tegangan untuk A/D converter, PC(3..0), dan ADC(7..6). Harus terhubung secara eksternal, bahkan jika ADC tidak digunakan. Jika ADC digunakan, harus terhubung melalui low pass filter. Perhatikan bagaimana menggunakan pin ini pada bab berikutnya.
AREF	AREF adalah referensi pin analog untuk A/D converter.
ADC7..6 (TQFP and CBFM MLF package only)	Pin ini hanya tersedia pada kemasan TQFP dan CBFM MLF. Mereka menyediakan fungsi yang sama sebagai PC(3..0). Lihat deskripsi PC(3..0).

Arithmetic Logic Unit (ALU)

Modul ALU yang memiliki fungsi aritmatika dan logika. Berhubungan dengan semua 32 register kerjanya. Dalam satu siklus clock ALU melakukan operasi aritmatika antar register atau antara register dan konstanta. Operasi ALU terbagi menjadi tiga kategori utama yaitu aritmatika, logika dan bit-fungsi. Biasanya mikrokontroler dari arsitektur juga menyediakan pemrosesan-perkalian yang memperkurang waktu eksekusan dengan perkalian dan hitungan pembagian.

Status register berisi informasi tentang hasil dari instruksi aritmatika yang terakhir dieksekusi. Informasi ini dapat digunakan untuk memicu aliran program untuk melakukan operasi yang berbeda kondisional. Pada

diperhatikan bahwa *Status Register* diperbaharui saat setelah semua operasi *ALU* terlebih dahulu dilakukan. Kegiatan ini dilakukan agar menghasilkan aliran kode yang lebih cepat dan komplit. *Status Register* tidak secara otomatis disimpan pada saat memasuki interupsi rutin dan akan dikembalikan pada saat kembali dari interupsi. Penyimpanan harus dilakukan oleh perangkat lunak.
AVR Status Register (*SREG*) didefinisikan sebagai berikut :

Bit	7	6	5	4	3	2	1	0	
	I	T	H	S	V	N	Z	C	SREG
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – I : Global Interrupt Enable**
  Global Interrupt Enable harus ditetapkan untuk dapat melakukan interupsi. *Individual interrupt* mengaktifkan kontrol selanjutnya yang nantinya akan melakukan *register* secara terpisah. Jika *register Global Interrupt Enable* dibersihkan, tak ada interupsi aktif yang dapat dilakukan secara independen kepada *individual interrupt*. I-bit dihapus oleh *hardware* setelah interupsi dilakukan dan diatur oleh RETI untuk mengaktifkan interupsi selanjutnya. I-bit juga dapat diatur dan dibersihkan oleh aplikasi dengan instruksi SEI dan CLI.
- **Bit 6 – T : Bit Copy Storage**
  Bit Copy menginstruksi *BLD* (*Bit Load*) dan *BST* (*Bit Storage*) menggunakan T-bit sebagai sumber dan tujuan pengoperasian bit. Bit yang berasal dari sebuah register pada *File Register* dapat di kopi kedalam T dengan instruksi *BST* dan bit yang berada didalam T dapat di kopi kedalam *register* yang berada didalam *File Register* dengan instruksi *BLD*.
- **Bit 5 – H : Half Carry Flag**
  Half Carry Flag-H mengindikasikan *Half Carry* didalam beberapa operasi aritmatik. *Half Carry* sangat berguna didalam aritmatik *BLD*.
- **Bit 4 – S : Sign Bit, S = N ⊕ V**
  S-bit selalu eksklusif atau antara *Negatif Flag*-N dan *Two's Complement Overflow Flag*-V

- **Bit 3 – V : Two's Complement Overflow Flag**

  Two's Complement Overflow Flag-V mendukung aritmatik *Two's Complement.*

- **Bit 2 – N : Negative Flag**

  Negative Flag-N mengindikasikan sebuah hasil negatif pada logika operasi dan aritmatik.

- **Bit 1 – Z : Zero Flag**

  Zero Flag-Z mengindikasikan sebuah hasil nol pada logika operasi dan aritmatik.

- **Bit 0 – C : Carry Flag**

  Carry Flag-C mengindikasikan bawaan pada logika operasi dan aritmatik.

General Purpose Register File

File Register dioptimalkan oleh instruksi dari *AVR Enhanced RISC.* Untuk mencapai performa dan fleksibilitas yang diinginkan, berikut ini skema input/output yang mendukung *File Register* :

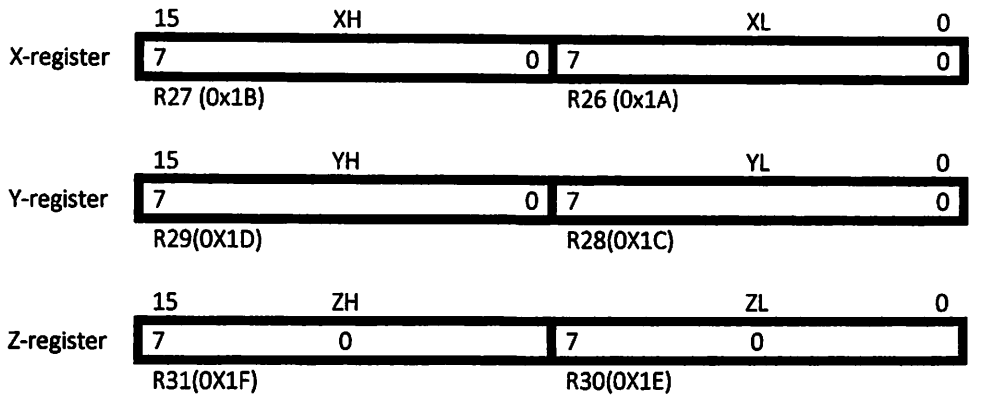
- Satu 8-bit outputan operan dan satu 8-bit inputan hasil
- Dua 8-bit outputan operan dan satu 8-bit inputan hasil
- Dua 8-bit outputan operan dan satu 16-bit inputan hasil
- Satu 16-bit outputan operan dan satu 16-bit inputan hasil

Gambar 2.15. (a) R0 (General Purpose Register) (b) Register Khusus

Sebagian besar perintah yang berupa operasi pada register memiliki bentuk seperti ADD R0, R1, R2 yang berarti menjumlahkan isi register R1 dengan isi register R2 dan hasilnya disimpan di register R0. Contoh lainnya adalah perintah MOV R0, R1 yang berarti menyalin isi register R1 ke register R0.

Selain 32 bilangan bulat, Cortex-M3 juga mendukung operasi floating point dengan presisi tunggal. Seperti yang terlihat pada gambar 2.15b, setiap register floating point akan menyimpan data yang berukuran 32 bit. Sebagai contoh, perintah VADD.F32 S0, S1, S2 akan menjumlahkan isi register S1 dan S2 lalu menyimpan hasilnya di register S0.

Arsitektur ARM Cortex-M3 memiliki 3 buah register khusus yang digunakan untuk pengalamatan secara tidak langsung (lihat gambar 2.16). Register-register tersebut adalah:



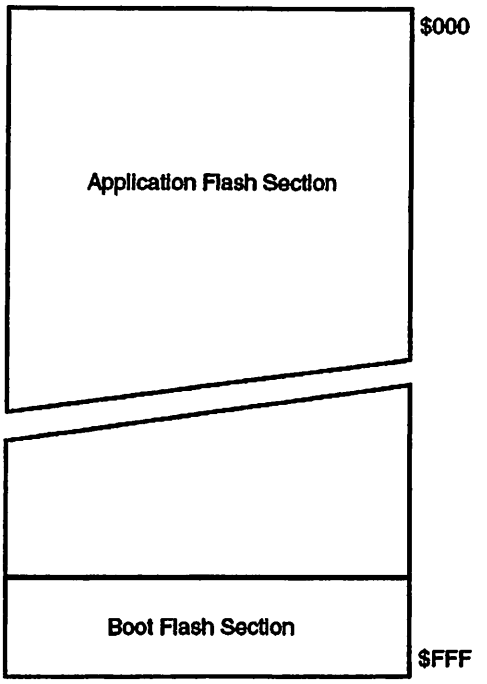
Gambar 2.16. X-register, Y-register dan Z-register

Memori AVR ATMega8

Bagian ini menjelaskan tentang memori dalam *Atmel® AVR® ATMega8*. Arsitekur AVR memililki dua ruang memori utama, memori *Data* dan *space Memori Program*. Selain itu ATMega8 memiliki memori EEPROM untuk penyimpanan data. Seluruh ketiga ruang pada memori adalah linier dan reguler. *ATMega8* memiliki 8Kb *On-Chip In-System Reprogrammable Flash Memory* untuk penyimpanan program. Karena semua instruksi AVR adalah meluas hingga 16-bit atau 32-bit, *Flash* diatur sebagai 4K x 16bit. Untuk keamanan perangkat lunak, ruang memori *Flash Program* dibagi menjadi dua bagian yaitu bagian *Program Boot* dan *Program Aplikasi*.

Memori *Flash* memiliki daya tahan setidaknya hingga 10.000 siklus *write/erase*. *ATMega8 Program Counter* (*PC*) mempunyai lebar 12 bit, sehingga menangani lokasi memori Program 4K.

Gambar 3.10. X-register, Y-register dan X-register



Gambar 2.17. Memori Program AVR

Memori Data SRAM

Lokasi Data Memori dibawah 1120 mengalamatkan *Register File*, I/O memori, dan data internal *SRAM*. 96 lokasi pertama mengalamatkan *Register File* dan *I/O Memory*, dan 1024 lokasi yang berikutnya menangani *Internal Data SRAM*.

Lima mode pengalamatan yang berbeda untuk sampul memori data : *Direct, Indirect with Displacement, Indirect, Indirect with Pre-Decrement* dan *Indirect with Post-increment*. Dalam *Register File, register R26* hingga *R31* memiliki fitur *pointer register* secara tidak langsung. Pengalamatan secara langsung mencakup semua *Data Space*.

Mode *Indirect with Displacement* menjangkau 63 alamat lokasi dari alamat dasar yang diberikan oleh *Y-register* dan *Z-register*. Ketika menggunakan mode pengalamatan *register* secara *Indirect*, seraca otomatis *Pre-Decrement* dan *Post-Increment*, alamat *register* X, Y dan Z menjadi berkurang atau bertambah. Seluruh 32 *General Purpose*, ke-64 I/O *Register*, dan 1024 byte data internal dalam ATMega8 semuanya dapat diakses melalui metode ini.

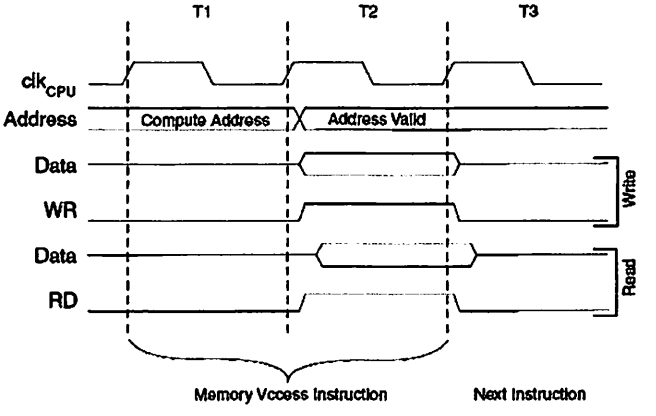
Register File	Data Address Space
R0	$0000
R1	$0001
R2	$0002
...	...
R29	$001D
R30	$001E
R31	$OO1F

I/O Registers	
$00	$0020
$01	$0021
$02	$0022
...	...
$3D	$005D
$3E	$005E
$3F	$005F

Internal SRAM
$0060
$0061
...
$045E
$045F

Gambar 2.18. Data Memori Map

Bagian ini merupakan penjelasan secara umum konsep waktu akses untuk akses memori internal. Akses data SRAM yang dilakukan dalam dua siklus clk_{CPU} seperti yang dijelaskan pada gambar 2.19.



Gambar 2.19. Siklus On-Chip data akses SRAM

Data Memori EEPROM

ATMega8 berisi 512byte memori data *EEPROM*. Hal ini diatur dalam ruang data yang terpisah, dimana pada tiap-tiap byte dapat dibaca dan ditulis. *EEPROM* ini setidaknya memiliki daya tahan hingga 100.000 siklus *write/erase*. Akses antara *EEPROM* dan data *CPU* yang dijelaskan dibawah ini, menentukan alamat *Register EEPROM* dan kontrol *Register EEPROM*.

Register EEPROM dapat diakses pada ruang I/O. Sebuah fungsi *self-timing* memungkinkan pengguna perangkat lunak dapat mendeteksi apabila byte berikutnya dapat ditulis. Jika kode pengguna berisi petunjuk yang menulis *EEPROM*, beberapa tindakan pencegahan harus dilakukan. Dalam *power supply* yang disaring ketat, *VCC* kemungkinan akan naik turun secara perlahan pada *Power-up/down*. Hal ini menyebabkan perangkat dalam beberapa waktu dapat dijalankan pada tegangan rendah dari nilai minimum yang ditentukan untuk frekuensi *clock* yang digunakan.

Ketika *EEPROM* dibaca, *CPU* dihentikan selama 4 siklus clock sebelum instruksi selanjutnya dijalankan. Ketika *EEPROM* in ditulis, *CPU* dihentikan selama 2 siklus clock sebelum instruksi selanjutnya dijalankan.

Bit	15	14	13	12	11	10	9	8	
	-	-	-	-	-	-	-	EEAR8	EEARH
	EEAR7	EEAR6	EEAR5	EEAR4	EEAR3	EEAR2	EEAR1	EEAR0	EEARL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R	R	R	R/W	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	X	
	X	X	X	X	X	X	X	X	

- **Bit 15 .. 9 – Res : Bit Cadangan (*Reserved Bits*)**
  Bit-bit ini dicadangkan dalam *ATMega8* dan akan selalu dibaca nol.
- **Bit 8 .. 0 – EAAR8 .. 0 : Alamat EEPROM**
  Alamat *Register EEPROM* (*EEARH* dan *EEARL*) menentukan alamat *EEPROM* dalam ruang 512byte *EEPROM*. Byte pada data *EEPROM* ditangani secara linier antara 0 dan 511. Nilai awal *EEAR* tidak terdefinisi. Sebuah nilai yang tepat harus ditulis sebelum *EEPROM* dapat diakses.

Bit	7	6	5	4	3	2	1	0	
	MSB							LSB	EEDR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 .. 0 – EEDR7 .. 0 : Data EEPROM**

  Untuk operasi *write* pada *EEPROM*, Register *EEDR* berisi data yang akan ditulis ke dalam alamat yang diberikan oleh *Register EEAR*. Untuk operasi *read*, *EEDR* berisi data yang dibaca dari *EEPROM* pada alamat yang diberikan oleh *EEAR*.

Bit	7	6	5	4	3	2	1	0	
	-	-	-	-	EERIE	EEMWE	EEWE	EERE	EECR
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	X	0	

- **Bit 7 .. 4 – Res : Bit Cadangan**

  Bit-bit ini dicadangkan pada *Atmel®AVR® ATMega8* dan akan selalu dibaca nol.

- **Bit 3 – EERIE : EEPROM Ready Interrupt Enable**

  Menuliskan nilai EERIE menjadi '1' memungkinkan *EEPROM Ready Interrupt* jika I bit pada SREG diatur. Menuliskan nilai *EERIE* menjadi '0' akan menonaktifkan *interrupt*. *EEPROM Ready Interrupt* menghasilkan interupsi yang konstan ketika nilai *EEWE* dihapus.

- **Bit 2 – EEMWE : EEPROM Master Write Enable**

  Bit *EEMWE* akan menentukan apakah pengaturan nilai '1' pada *EEWE* akan menyebabkan *EEPROM* ditulis pula. Ketika *EEMWE* diatur, pengaturan *EEWE* dalam siklus waktu empat *clock* akan menulis data ke *EEPROM* pada alamat dipilih. Jika keadaan *EEMWE* adalah '0', pengaturan *EEWE* tidak akan berpengaruh. Jika *EEMWE* bernilai '1' oleh *software*, *hardware* akan membersihkan bit menjadi '0' setelah empat siklus *clock*.

- **Bit 1 – EEWE : EEPROM Write Enable**

  Sinyal EEPROM Write Enable EEWE adalah *write strobe* ke *EEPROM.* Ketika alamat dan data diatur dengan benar, bit *EEWE* harus bernilai '1' pada *EEPROM.* Bit *EEMWE* harus bernilai '1' sebelum nilai logika pada *EEWE,* jika tidak maka *EEPROM* tidak akan mengambil alih. Berikut ini adalah prosedur yang harus dilakukan ketika ingin menulis pada *EEPROM*:

  - Tunggu hingga nilai *EEWE* '0'
  - Tunggu hingga nilai *SPMEN* pada *SPMCR* '0'
  - Tulis alamat *EEPROM* baru untuk *EEAR* (opsional)
  - Tulis data *EEPROM* baru untuk *EEDR* (opsional)
  - Tulis logika '1' untuk bit *EEMWE* saat menulis '0'untuk *EEWE* pada *EECR*
  - Dalam empat siklus *clock* setelah pengaturan *EEMWE,* tulis nilai logika menjadi '1' untuk *EEWE.*

- **Bit 0 – EERE : Read Enable**

  EEPROM membaca sinyal *EERE Read Enable* merupakan read strobe ke *EEPROM.* Ketika alamat yang benar di set ke dalam *Register EEAR,* bit *EERE* harus bernilai logika '1'untuk memicu *EEPROM* membaca. Akses membaca *EEPROM* membutuhkan satu instruksi dan data yang diminta akan segera tersedia. Ketika *EEPROM* dibaca, *CPU* dihentikan selama 4 siklus *clock* sebelum instruksi selanjutnya dijalankan.


INSTITUT TEKNOLOGI NASIONAL
ITN
MALANG

BAB III
ANALISA DAN PERANCANGAN SISTEM

Dalam bab ini akan di jelaskan tentang perancangan dari aplikasi *home monitoring berbasis android*. Perancangan akan terbagi dari beberapa sub bab, salah satunya alur sistem yang akan dijelaskan dengan *flowchart* dan blok diagram. Untuk lebih jelasnya kita lihat penjelasannya sebagai berikut :

3.1. Pengumpulan Data

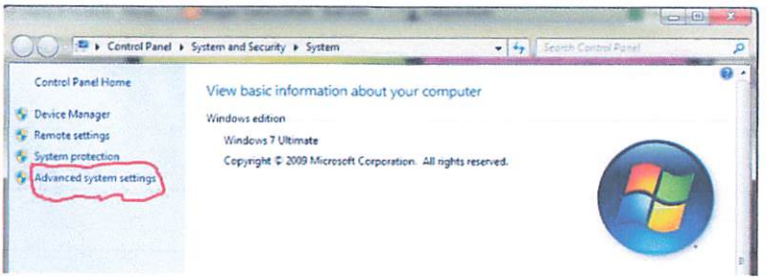
a. Pencarian referensi-referensi yang berkaitan dengan Android dan *software* Eclipse sebagai sarana untuk membuat aplikasi.
b. Mempelajari lebih dalam tentang komunikasi nirkabel yang akan diterapkan pada aplikasi.
c. Pencarian perangkat yang cocok untuk dijadikan alat pada jaringan komunikasinya.

3.2. Analisa Sistem

Setelah melihat beberapa hasil pengumpulan data seperti yang tertera pada sub-bab **3.1**, maka didapatkan beberapa analisa dari aplikasi maupun perangkat utama yang dibutuhkan, antara lain :

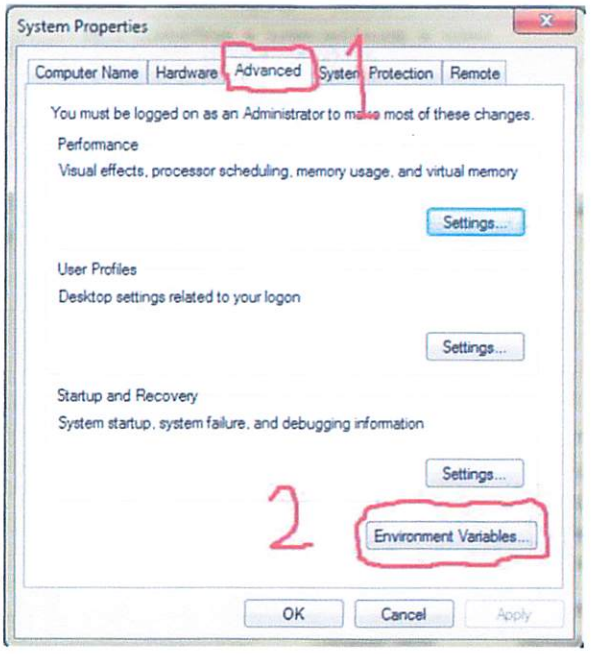
3.2.1. Instalasi *Java Development Kit* (*JDK*)

a. Siapkan instalasi *software JDK* yang anda miliki, jika belum memiliki dapat mengunduhnya melalui www.oracle.com. Setelah selesai langsung saja instal, biasanya ikuti saja langkah-langkah yang tertera pada jendela instalasinya dengan menekan tombol "*next*".
b. Setelah selesai, buka **Windows Explorer** caranya klik kanan pada kolom **My Computer - Properties** kemudian klik **Advance system settings.**



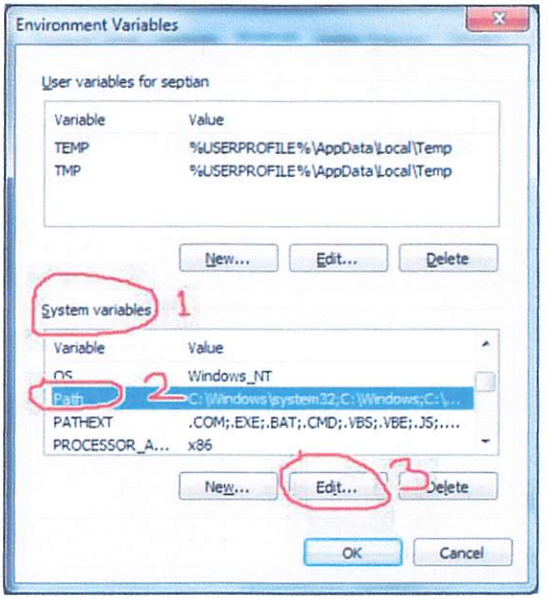
Gambar 3.1. Instalasi *Java Development Kit* (a)

c. Setelah itu muncul jendela *Properties*, kemudian pilih tab ***Advanced***, klik ***Environment Variables***.



Gambar 3.2. Instalasi *Java Development Kit* (b)

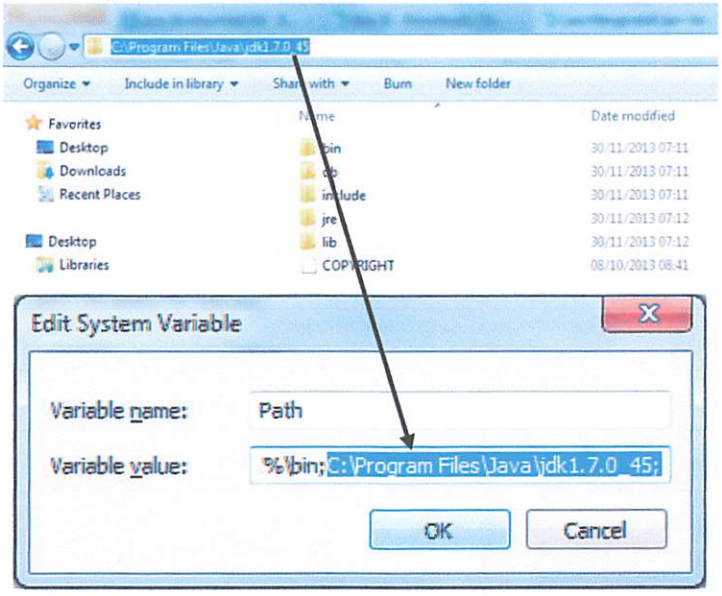
d. Setelah masuk jendela ***Environment Variables***, pada tab ***System Variables***, klik ***variable path – edit***.



Gambar 3.3. Instalasi *Java Development Kit* (c)

e. Setelah tombol edit di klik maka akan muncul jendela ***Edit System Variable***, pada ***Variable name*** sudah terisi otomatis dengan *path*, kemudian pada ***Variable value*** *paste*-kan dimana kita menginstal java. Disini penulis menginstal java pada direktori.

;C:\Program Files\Java\jdk1.7.0_45

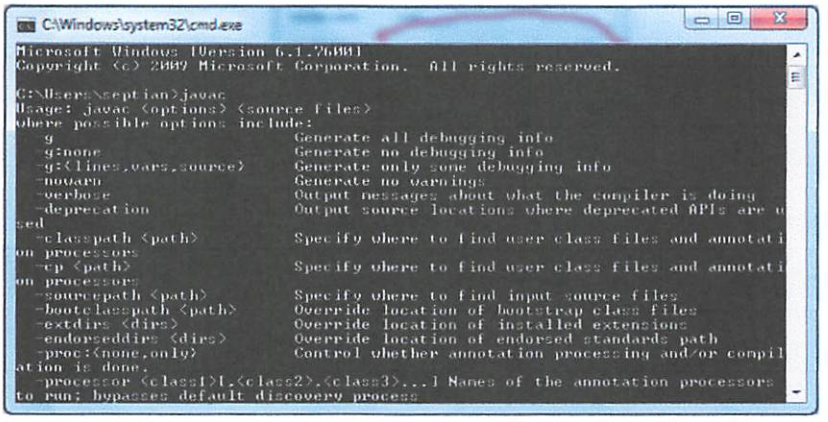


Gambar 3.4. Instalasi *Java Development Kit* (d)

f. Kemudian tekan 'OK' hingga jendela tertutup.

g. Jika ingin tau apa *JDK* telah terinstal dengan baik, kita bisa mengetes dengan membuka buka jendela *command prompt* (*cmd*), ***start->ketik 'cmd'->enter.***

h. Ketik ***javac***, apabila sukses maka akan keluar seperti gambar 3.5.



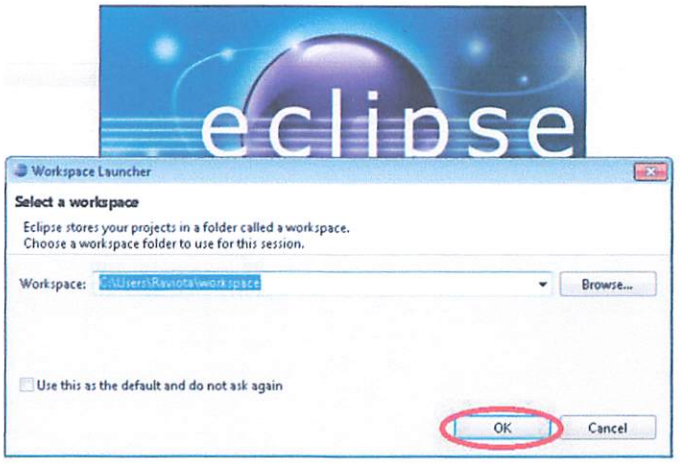
Gambar 3.5. Instalasi *Java Development Kit* (e)

i. Jika muncul seperti ini, ***'javac' is not recognized as an internal or external command, operable program or batch file***, berarti instalasi tidak dilakukan dengan benar maka *restart*-lah komputer saudara kemudian ulangi lagi hingga berhasil.

3.2.2. Instalasi *Eclipse Kepler*

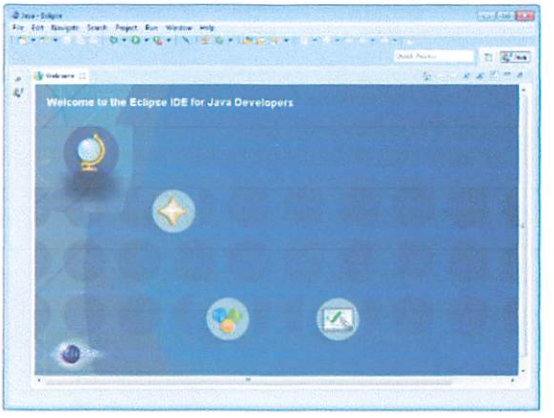
a. Hal pertama unduh *Eclipse Kepler* dan *ADT (Android Development Tools),* dapat dicari di *google*.

b. Kemudian instal *Eclipse Kepler* yang telah diunduh tersebut, kemudian akan muncul tampilan *workplace* yang akan menunjukkan dimana anda akan menyimpan hasil *compile project* anda, saya menyimpannya di dalam direktori 'D:\ ' kemudian tunggu hingga selesai.



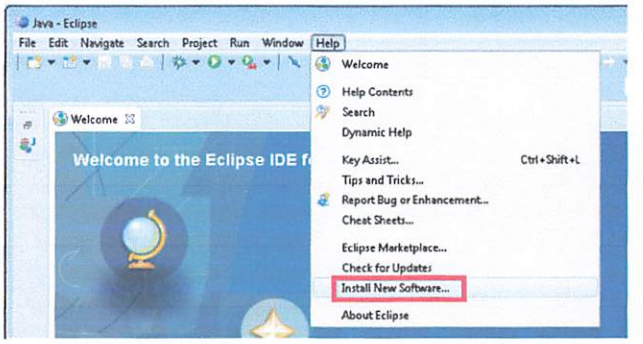
Gambar 3.6. Instalasi *software Eclipse* (a)

Setelah selesai akan muncul tampilan seperti gambar 3.7.



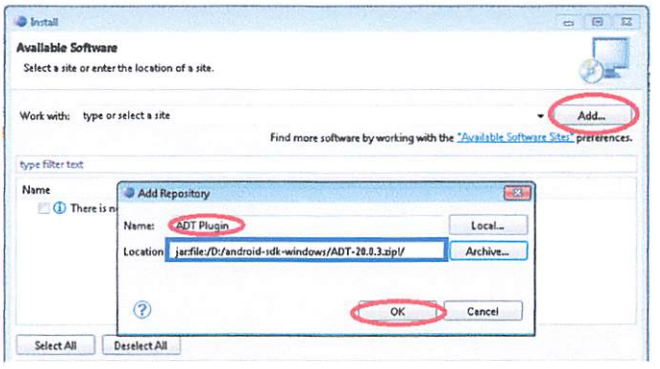
Gambar 3.7. Instalasi *software Eclipse* (b)

c. Langkah selanjutnya adalah menginstal *ADT* yang telah kita unduh sebelumnya.



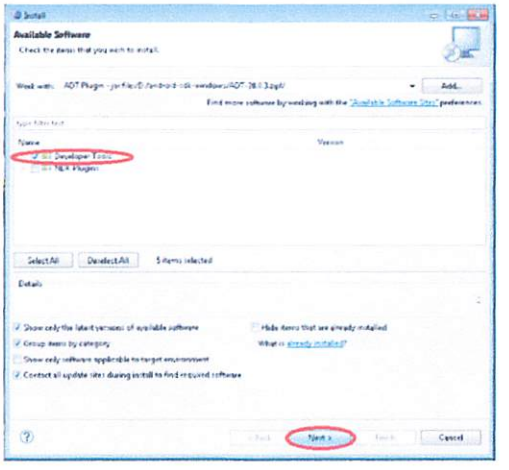
Gambar 3.8. Instalasi *software Eclipse* (c)

d. Setelah itu akan muncul jendela dan pilih ***add***. Kemudian isi kolom "*Name*" dengan "*ADT Plugin*" dan browse file *ADT Plugin* yang tadi telah diunduh. Kemudian klik *button* OK.



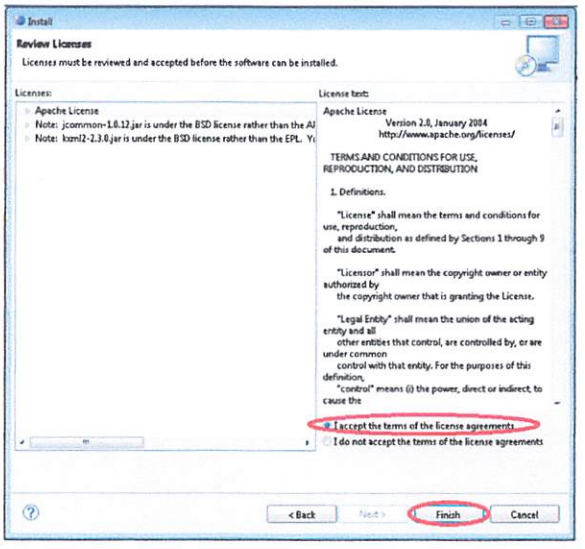
Gambar 3.9. Instalasi *software Eclipse* (d)

e. Pada dialog *Available Software*, centang *checkbox* yang berisi *Developer Tools* kemudian *next*.



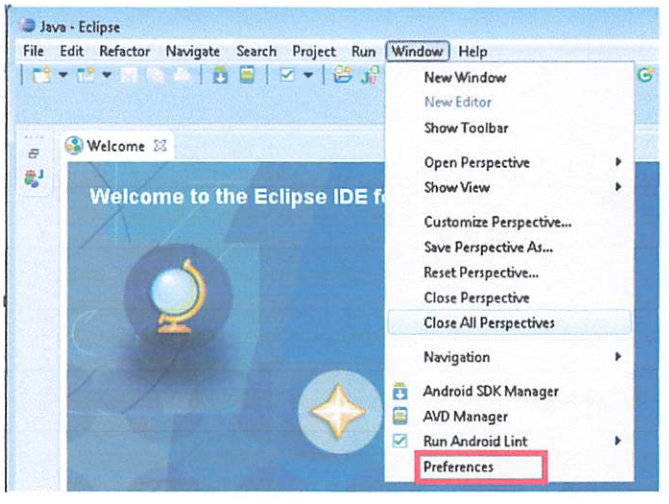
Gambar 3.10. Instalasi *software Eclipse* (e)

f. Akan muncul satu *form* lagi yang meminta kita untuk menyetujui lisensi dan klik pada *radio button* "*I accept the terms of the license agreement*", kemudian *Finish* dan *restart Eclipse* anda.



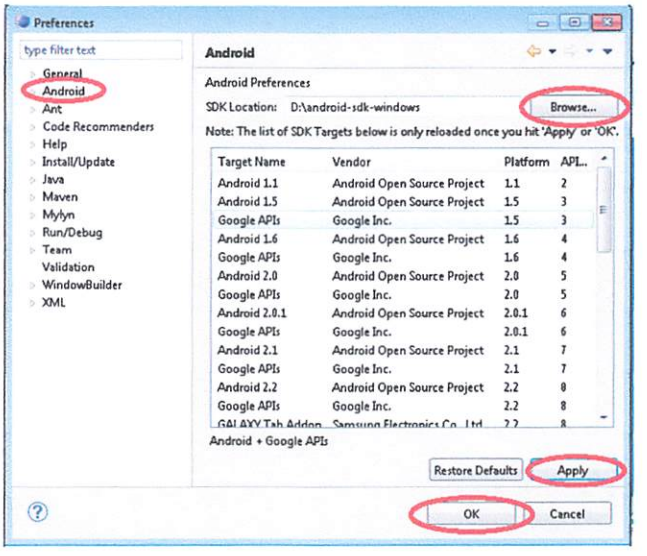
Gambar 3.11. Instalasi *software Eclipse* (f)

g. Untuk proses terakhir adalah menghubungkan dengan *SDK*. Pertama masuk ke menu "***Window –> Preferences***"



Gambar 3.12. Instalasi *software Eclipse* (g)

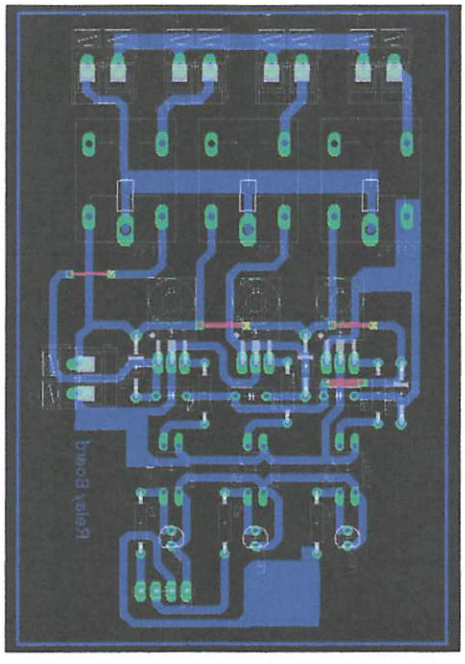
h. Akan muncul sebuah form lagi, kemudian klik pada tab "Android", pada dialog tersebut klik tombol ***Browse*** kemudian arahkan ***path*** / lokasi folder ***android-sdk-windows*** anda kemudian "*Apply*" dan "OK".



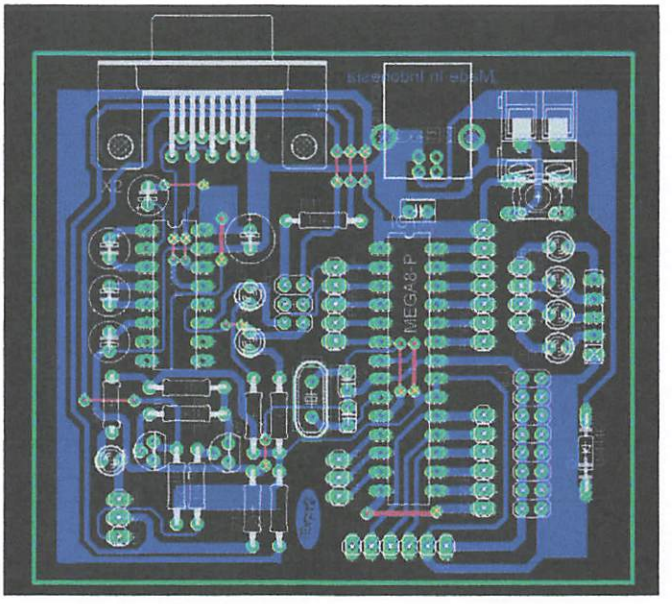
Gambar 3.13. Instalasi *software Eclipse* (h)

3.2.3. Perancangan Hardware

Penulis membuat PCB dengan *software* ***Eagle***, buatlah rangkaian hingga menjadi seperti gambar 3.14-16.

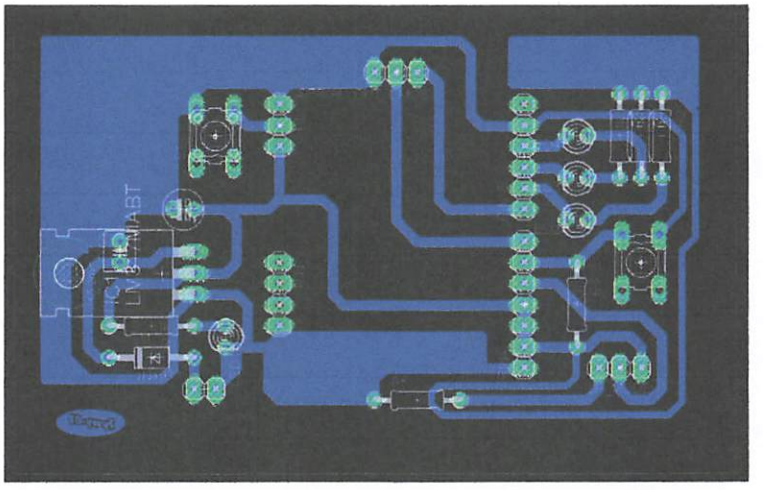


Gambar 3.14. Rangkaian *Driver Relay*

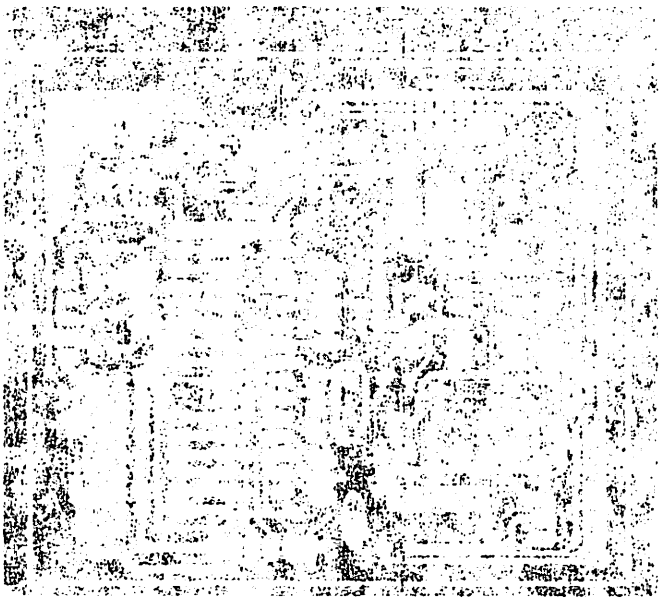


Gambar 3.15

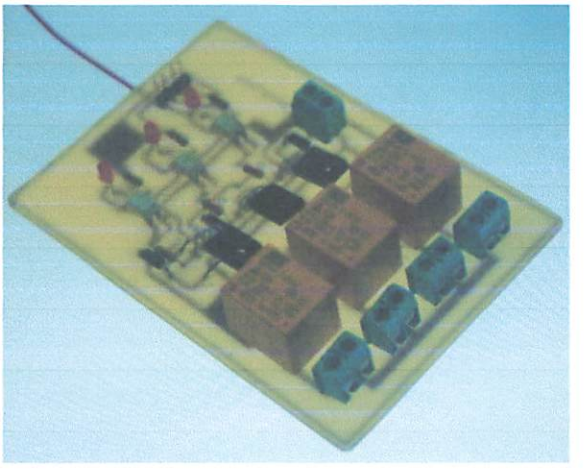
Rangkaian Minimum Sistem Mikrokontroler ATMega8 dan Rangkaian Level Konverter Dari TTL ke CMOS



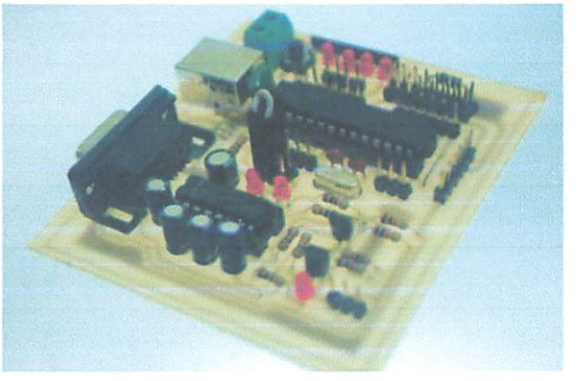
Gambar 3.16. Rangkaian *Adapter* dan *Regulator WizFi210*



Setelah itu rakitlah komponen hingga menjadi seperti gambar 3.17-19.

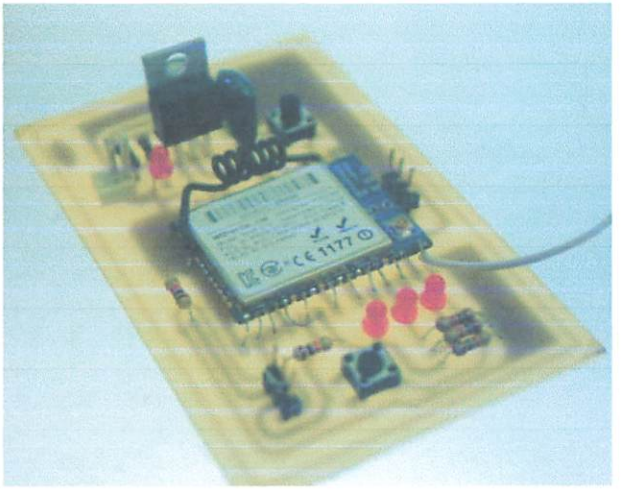


Gambar 3.17. Penerapan rangkaian *Driver Relay* pada PCB



Gambar 3.18

Penerapan rangkaian Minimum Sistem ATMega8 pada PCB

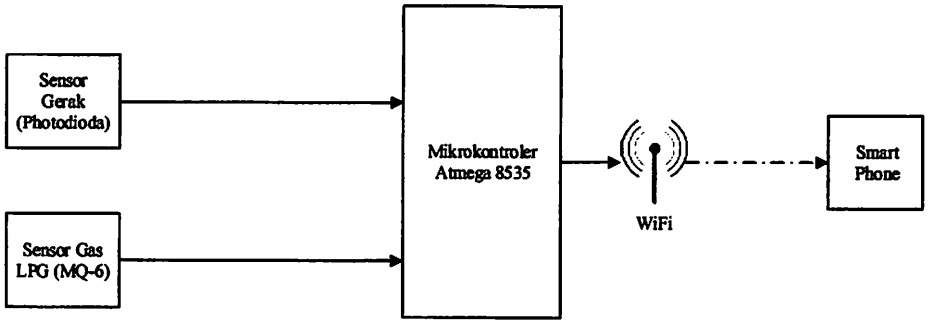


Gambar 3.19. Penerapan rangkaian WizFi210 pada PCB

3.3. Perancangan Sistem

Rancangan yang akan dibuat adalah bagaimana sensor-sensor dan lampu yang terdapat pada suatu rangkaian elektronika dapat diperintah melalui telepon genggam. Mulai dari rancangan tampilan utama pada telepon genggam hingga proses pengiriman datanya.

Berikut gambaran diagram blok untuk memonitor sensor yang ditunjukkan oleh gambar 3.20.



Gambar 3.20. Diagram Blok Aplikasi Smart Home

Penjelasan diagram blok pada gambar 3.20 adalah sebagai berikut :

1. *Sensor Photodioda dan LED*

   Berfungsi mengirimkan data saat sensor diaktifkan dan telah terdeteksi adanya pergerakan dari manusia yang melewati sensor.

2. *Sensor Gas LPG (Sensor MQ-6)*

   Mengirimkan data saat sensor di aktifkan dan telah terdeteksi adanya perubahan gas terhadap resistansi sensor.

3. *Mikrokontroler*

   Berfungsi sebagai pemroses masukan dari sensor-sensor yang masuk dan mematikan atau menghidupkan saklar elektrik.

4. *WiFi*

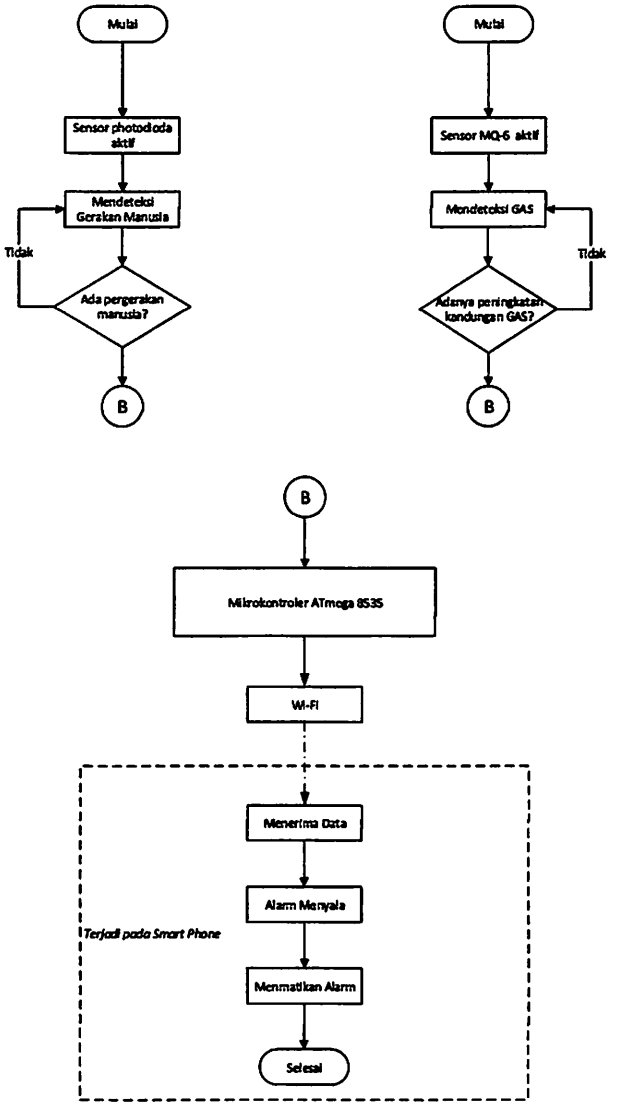
   Membaca dan memproses data-data yang akan dikirimkan ke receiver ataupun sebaliknya yaitu smartphone yang berfungsi sebagai pengontrol. Sinyal yang dikirimkan berupa data-data biner.

5. *Smart Phone*

   Berfungsi sebagai *display*, dan pusat kontrol untuk lampu. Bila ada pemberitahuan terjadinya kondisi bahaya dari sensor photodioda maupun sensor gas *LPG*, maka pada alarm akan menyala pada *smartphone*.

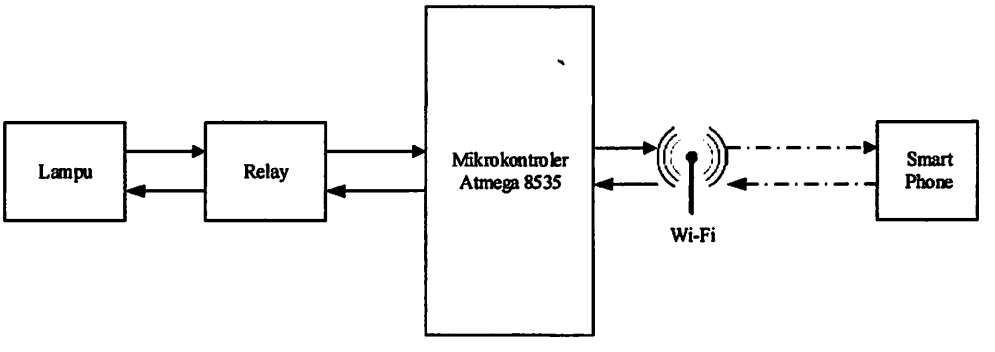
Penjelasan cara kerja dari diagram blok yang tertera pada gambar 3.20 adalah jika cahaya *LED* pada sensor photodioda terhalang oleh orang atau adanya benda yang menyebabkan tidak ada cahaya yang sampai menuju sensor *photodioda*, maka mikrokontroler akan mengirimkan isyarat atau tanda ke *smartphone* melalui *WiFi* yang merupakan penghantar komunikasi antara *smartphone* dengan *mikrokontroler*. Begitu juga dengan pendeteksi Gas *LPG*, jika sensor *MQ-6* merasakan atau mendeteksi adanya konsentrasi gas yang berlebihan, maka sensor akan mengirimkan tanda atau isyarat yang akan dikirimkan dari mikrokontroler menuju *smartphone* melalui *WiFi* yang menggunakan *WizFi210* sebagai perangkatnya.

Gambar 3.21 menunjukkan diagram alir proses pemberitahuan pada sensor ke *smart phone*. Diawali dengan pengaktifannya sensor-sensor yang terpasang, sensor-sensor tersebut mendeteksi hal yang menjadi tugasnya yaitu sensor photodioda mendeteksi adanya gerakan dan sensor MQ-6 mendeteksi kadar gas LPG. Jika terdeteksi gerakan atau gas, maka sensor akan mengirimkan sinyal ke mikrokontroler. Kemudian mikrokontroler mengirimkan data ke *smartphone* melalui *WiFi*. Pada akhirnya alarm pada *smartphone* akan menyala menandakan kepada pengguna adanya gangguan antara sensor photodioda dan sensor gas.



Gambar 3.21. *Flowchart* proses pemberitahuan pada sensor

Sementara untuk diagram blok dan *flowchart* pengotrolan lampu ditunjukkan pada gambar 3.22 dan gambar 3.23.



Gambar 3.22. Diagram Blok Pengontrolan Lampu

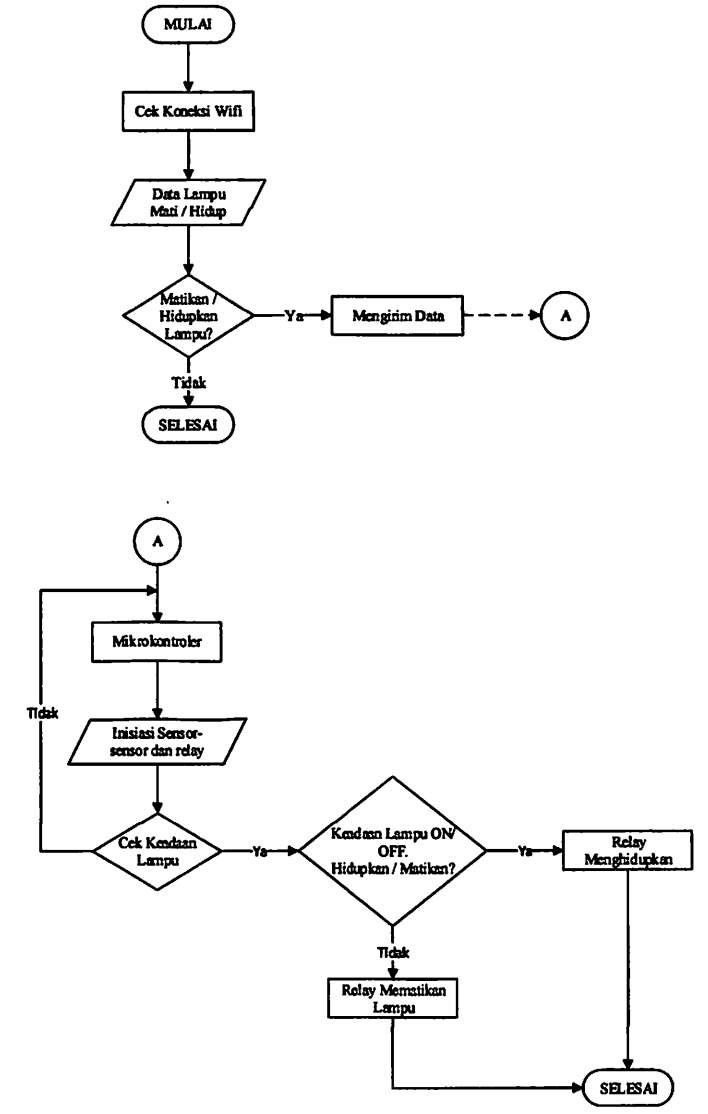
Penjelasan diagram blok yang ditunjukkan oleh gambar 3.22 adalah sebagai berikut :

1. *Lampu*, adalah objek yang akan dikontrol oleh *user* yang menggunakan aplikasi dalam *smartphone*.
2. *Relay / Driver Relay, Mikrokontroler, Wi-Fi, dan Smartphone* Banyaknya *driver relay* tergantung dari seberapa banyak jumlah lampu yang di gunakan. Setiap *driver relay* berfungsi untuk menghidupkan atau mematikan beban lampu. Pada rangkaian ini juga terdapat *toggle switch* yang berfungsi sebagai saklar manual lampu.
3. *Panah ke kanan,* menujukkan bahwa data keadaan lampu pada saat aplikasi dinyalakan dan akan dikirimkan melewati *relay* dan *mikrokontroler* yang perngiriman datanya dilakukan melalui media *WiFi* sebelum akhirnya terlihat pada tampilan aplikasi utama yaitu pada *smart phone*.
4. *Panah ke kiri*, setelah lampu ruangan mengirimkan datanya menuju *smart phone* ketika *user* ingin mematikan/menghidupkan lampu yang dalam kondisi tidak diperlukan atau yang sedang diperlukan maka *smart phone* akan mengirimkan data kembali melalui *WiFi* dan diterima oleh *mikrokontroler* pada akhirnya perintah akan di eksekusi oleh *relay* untuk mematikan / menghidupkan lampu.

Penjelasan cara kerja diagram blok pada gambar 3.22 adalah ketika semua perangkat aktif dan terhubung, *smart phone* akan mendeteksi keadaan lampu apakah aktif atau tidak. Melalui mikrokontroler, aktif atau tidaknya lampu dapat diketahui. Setelah mengetahuinya, dengan *smartphone* kita memberi perintah kepada mikrokontroler untuk mematikan atau menghidupkan lampu melalui *WiFi*. Kemudian mikrokontroler akan memerintahkan *driver relay* yang bertugas sebagai eksekutor yaitu menyalakan atau mematikan lampu. Dengan cara kerja driver yang dapat memutuskan dan menghantarkan arus dan tegangan, *driver relay* adalah alat yang tepat untuk digunakan untuk hal ini. Setelah *driver relay* mendapatkan perintah yang

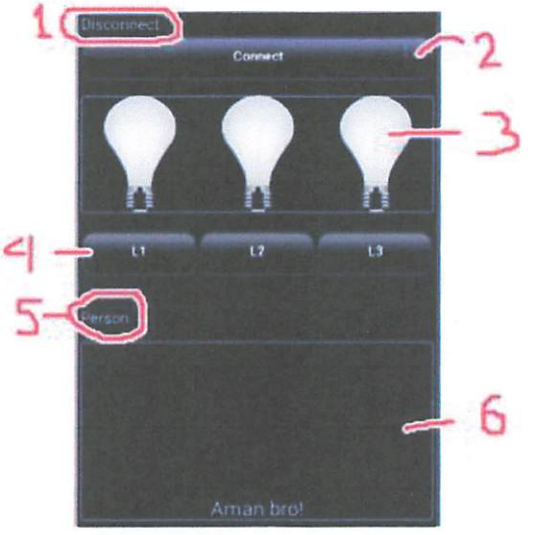
disalurkan oleh mikrokontroler, maka ia akan mematikan atau menghidupkan lampu yang diperlukan atau yang tidak diperlukan.

Berikut ini gambar 3.23 yaitu *flowchart* dari pengontrol lampu .



Gambar 3.23. Flowchart Pengontrol Lampu

Peracangan tampilan pada *smart phone* ditunjukkan oleh Gambar 3.24 serta untuk penjelasan gambarnya terdapat pada Tabel 3.1.



Gambar 3.24. Perancangan sementara tampilan utama pada *smart phone*

Tabel.3.1. Penjelasan Gambar 3.24

No.	Penjelasan
1	Status konektifitas, jika kondisi *WiFi* tersambung dengan baik maka akan ditunjukkan pada *text* ini
2	Tombol untuk menghubungkan *handphone* dengan perangkat elektronika
3	Indikator lampu apakah menyala atau tidak
4	Tombol untuk menyalakan masing-masing lampu
5	Indikator keadaan sensor pendeteksi gerak aktif, jika merasakan adanya pergerakan seseorang maka tulisan ini berubah secara otomatis
6	Tampilan indikator gas yang terdeteksi oleh sensor

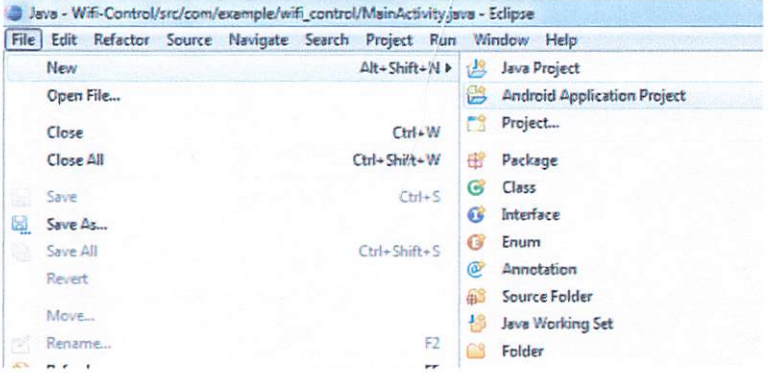
BAB IV
IMPLEMENTASI DAN PENGUJIAN APLIKASI

4.1. Implementasi Sistem

Dalam bab ini akan menjelaskan tentang implementasi dan pengujian dari aplikasi ***home monitoring berbasis android***. Dalam hal ini, *script* yang digunakan akan dibahas dan beberapa penjelasan secara umum mengenai perintah-perintah dan komunikasinya. Diataranya tentang bagaimana perangkat keras bekerja dan mengirimkan data menuju ke *smart phone* yang ber-*sistem operasi*-kan android serta bagaimana pembuatan *script* pada aplikasi *Eclipse Kepler* dalam menerima dan mengirimkan data pada *smart phone*.

4.1.1. Pembuatan Aplikasi dengan Eclipse

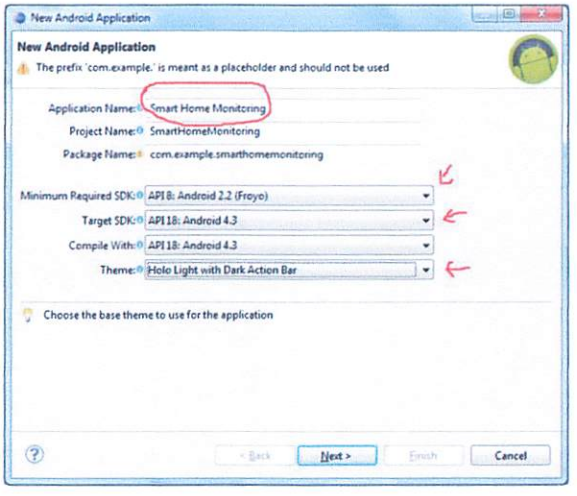
Pertama-tama bukalah aplikasi *Eclipse* yang sebelumnya sudah diinstal pada komputer anda. Kemudian pada jendela utama kiri atas, pilih *File > New > Android Application Project*.



Gambar 4.1. Membuat *Project* baru

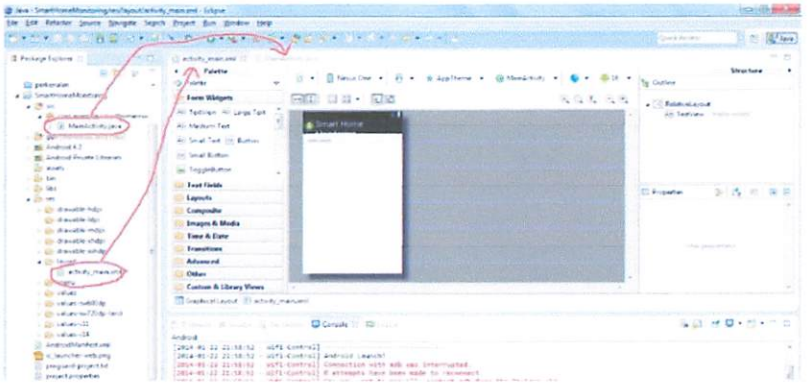
Akan muncul jendela baru, masukkan nama aplikasi pada kolom yang tersedia. Kemudian aturlah versi android minimal dan maksimal yang dapat di instal. Ketahuilah secara pasti versi berapa android yang akan anda pakai nanti, karena jika versi android anda kurang atau lebih dari yang anda atur maka nantinya aplikasi tidak akan bisa diinstal. Anda juga dapat menentukan tema apa yang akan anda gunakan nanti dengan

memilih, tepat dibawah kolom pengaturan versi android. Kemudian tekan tombol "*next*" hingga "*finish*". Saran saya jangan merubah konfigurasi yang telah ada untuk memudahkan pengerjaan. Namun, jika anda telah meguasai *software Eclipse*, silahkan anda buat dengan sesuka hati.



Gambar 4.2. Pemberian identitas pada Aplikasi

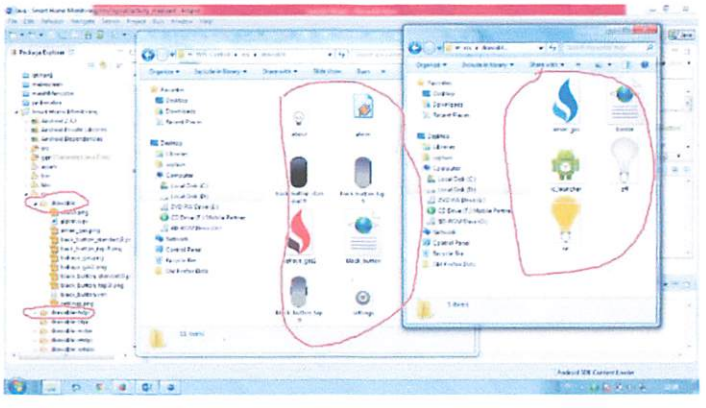
Setelah itu akan muncul jendela dimana kita akan membuat aplikasi dengan bahasa Java. Disini terdapat *activity* utama dengan dua form "*MainActivity.java*" dan "*activty_main.xml*" yang kita gunakan sebagai tempat membuat aplikasi. Form *MainActivity.java* merupakan tempat untuk membuat *source code*-nya dan form *activity-main.xml* kita gunakan untuk membuat tampilannya. Disinilah tempat kita untuk bereksperimen.



Gambar 4.3. Layer *MainActivity.java* dan *activity main.xml*

Agar lebih mudah dalam pengerjaan, lebih baik kita membuat *layout*-nya terlebih dahulu. Bukalah tampilan *activity_main.xml*, disana terdapat dua *layer* yang bisa digunakan yaitu *Graphical Layout* dan *activity_main.xml* yang berada pada pojok kiri bawah pada tampilan *layout editor*. Kita bisa membuat *layout* dengan dua cara, yang pertama kita membuat *script* pada layer *activity_main.xml* dengan cara membuat *script layout* manual dengan bahasa java. Cara yang kedua adalah kita bisa membuat *layout* dalam *layer Graphical Layout* dengan opsi "*drag and drop*" yang telah diterapkan pada *Eclipse* yang juga menjadi salah satu kemudahan yang diberikan untuk pengembang aplikasi pemula. Karena saya termasuk pemula, maka saya menggunakan cara kedua untuk tahap pembuatan *layout*-nya.

Hal pertama yang dilakukan adalah menyiapkan semua gambar yang akan dimasukkan ke dalam aplikasi nanti, gambar-gambar itulah yang nantinya akan menjadi partikel-partikel yang akan dihubungkan dengan *script* java yang akan dibuat pada *layer main_activity.java*. Kemudian tinggal kita klik dan seret ke dalam file *drawable* atau *file* yang berada dibawahnya seperti yang dijelaskan pada gambar 4.4.

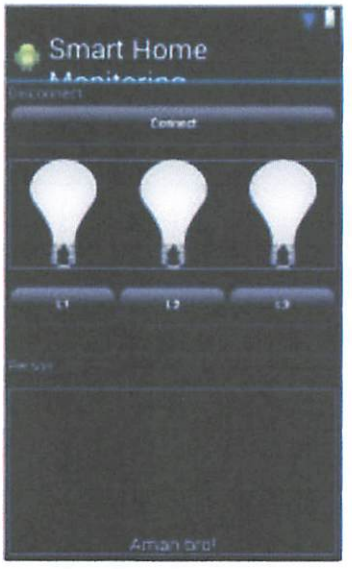


Gambar 4.4. Mamasukkan komponen eksternal pada Eclipse

Setelah semua komponen telah masuk ke dalam *software Eclipse*, maka secara otomatis akan tersimpan dengan permanen. Jika sumber komponen asli di dalam *folder* asalnya dihapus, maka tidak akan berpengaruh dalam *Eclipse*. Seperti beberapa *software* yang ada, jika

sumber gambar atau komponen yang lainnya dihapus maka aplikasi yang telah jadi tidak bisa di *run* atau dijalankan dengan keterangan tidak adanya sumber komponen yang telah dimasukkan.

Kembali ke pembuatan *layout*, susun dan atur komponen yang tadinya sudah dimasukkan ke dalam *software* hingga seperti gambar 4.5.



Gambar 4.5.Tampilan Layout Aplikasi Home Monitoring

Perlu diketahui, jika kita membuat layout di dalam *layer Graphical Layout* maka pada *layer activity_main.xml* akan secara otomatis menulis *script* dari komponen yang kita buat di layer *Graphical Layout*. Berikut ini *script* pada *layer activity_main.xml*.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/llMain"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:padding="2dip" >

    <LinearLayout
        android:id="@+id/Linear1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >
```

```
    <TextView
        android:id="@+id/status"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Disconnect" />
</LinearLayout>

<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical" >

    <ToggleButton
        android:id="@+id/btnConnected"
        style="@style/button_style"
        android:layout_width="fill_parent"
        android:layout_height="40dp"
        android:textOff="Connect"
        android:textOn="Disconnect" />
</LinearLayout>

<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="0dp"
    android:layout_marginTop="12dp"
    android:layout_weight="1"
    android:background="@drawable/border"
    android:orientation="horizontal" >

    <ImageView
        android:id="@+id/lamp1"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:src="@drawable/off" />

    <ImageView
        android:id="@+id/lamp2"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:src="@drawable/off" />

    <ImageView
        android:id="@+id/lamp3"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:src="@drawable/off" />
</LinearLayout>

<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="0dp"
    android:layout_marginTop="14dp"
    android:layout_weight="3"
    android:orientation="horizontal" >
```

```
    <Button
        android:id="@+id/btnLamp1"
        style="@style/button_style"
        android:layout_width="0dp"
        android:layout_height="40dp"
        android:layout_weight="1"
        android:text="L1" />

    <Button
        android:id="@+id/btnLamp2"
        style="@style/button_style"
        android:layout_width="0dp"
        android:layout_height="40dp"
        android:layout_weight="1"
        android:text="L2" />

    <Button
        android:id="@+id/btnLamp3"
        style="@style/button_style"
        android:layout_width="0dp"
        android:layout_height="40dp"
        android:layout_weight="1"
        android:text="L3" />
</LinearLayout>

<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="0dp"
    android:layout_marginTop="12dp"
    android:layout_weight="1"
    android:orientation="vertical" >

    <ProgressBar
        android:id="@+id/pgSuhu"
        style="?android:attr/progressBarStyleHorizontal"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content" />
</LinearLayout>

<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="match_parent"
    android:layout_weight="1"
    android:orientation="horizontal" >

    <TextView
        android:id="@+id/tvSuhu"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="Person" />
</LinearLayout>

<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="0dp"
```

```
        android:layout_marginTop="12dp"
        android:layout_weight="1"
        android:background="@drawable/border"
        android:orientation="vertical" >

        <ImageView
            android:id="@+id/gasPict"
            android:layout_width="match_parent"
            android:layout_height="140dp"
            android:scaleType="centerInside" />

        <TextView
            android:id="@+id/stateGas"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:gravity="center_horizontal"
            android:text="Aman bro!"
            android:textSize="18dp" />
    </LinearLayout>

</LinearLayout>
```

Setelah kita selesai membuat tampilan atau *layoutnya*, sekarang saatnya kita membuat *script* untuk perintah pada komponen-komponen yang telah ada. Kini kita berpindah menuju ke tampilan *MainActivity.java* untuk menulis satu per satu perintah apa saja yang akan digunakan. Berikut ini adalah *script* dan penjelasannya.

```
public class MainActivity extends Activity {

    private static int REFRESH_TIME = 300;

    private boolean setLamp1Click = false;
    private boolean setLamp2Click = false;
          private boolean setLamp3Click = false;


    private Timer timer;

    public byte sensorSuhuVal = 0;
    public byte sensorGasVal = 0;
          public byte x = 0;
```

Membuat variabel untuk timer yang digunakan untuk mengirim data ke *hardware* secara periodik setiap 300ms.

```
public char[] dataYangDikirim = new char[4];
    public byte[] dataYangDiterima = new byte[20];
```

Membuat variabel *array* bertipe *char* untuk menampung data yang akan dikirim ke perangkat keras sebanyak 4 data.

```
private Handler mHandler;
```

Membuat *Handler* (sistem penanganan) pada konsep *Threading* aplikasi.

```
public TCPClient mTcpClient;
```

Membuat turunan (*Inheritance*) dari kelas *TCPClient*.

```
private connectTask contask;
    private boolean runningState = true;
```

Buat turunan dari kelas *connectTask*.

```
float curVolume, maxVolume, leftVolume, rightVolume;
int priority = 1;
int no_loop = 0;
float normal_playback_rate = 1f;
```

Variabel yang digunakan untuk pengolahan suara yang akan dipakai sebagai alarm jika *Level* gas telah mencapai nilai tertentu.

```
SoundPool soundPool;
```

Membuat turunan dari kelas *Soundpool* untuk penanganan *audio* (*Tone*) yang difungsikan sebagai alarm gas.

```
HashMap<Integer, Integer> soundPoolMap;
```

Pemetaan dari file audio.

```
@Override
    protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    if (Build.VERSION.SDK_INT >Build.VERSION_CODES.GINGERBREAD)
    {
    StrictMode.ThreadPolicy policy = new
    StrictMode.ThreadPolicy.Builder().permitAll().build();
    StrictMode.setThreadPolicy(policy);
    }
```

Fungsi yang merupakan turunan dari kelas *Activity* yang akan dijalankan ketika aplikasi pertama kali dijalankan.

```
setContentView(R.layout.activity_main);
```

Memanggil *layout* (*.xml*) kedalam aplikasi utama.

```
AudioManager audioManager = (AudioManager)
getSystemService(Context.AUDIO_SERVICE);
curVolume =
audioManager.getStreamVolume(AudioManager.STREAM_MUSIC);
maxVolume =
audioManager.getStreamMaxVolume(AudioManager.STREAM_MUSIC);
leftVolume = curVolume / maxVolume;
rightVolume = curVolume / maxVolume;

soundPool = new SoundPool(30, AudioManager.STREAM_MUSIC, 100);
soundPoolMap = new HashMap<Integer, Integer>();
```

Instruksi untuk pengaturan / konfigurasi *Soundpool*.

```
soundPoolMap.put(1,soundPool.load(this, R.drawable.alarm, 1));
```

Instruksi ini digunakan untuk memanggil suara yang akan digunakan pada saat alarm berbunyi nanti.

```
final ImageView lamp1Pict = (ImageView)findViewById
(R.id.lamp1);
final TextView tvState = (TextView) findViewById(R.id.status);
```

Instruksi ini untuk menggabungkan *Widget* (komponen) dalam *file .xml* ke dalam *script .java* (utama).

```
contask = new connectTask();
```

Perintah "*new*" yang tertera digunakan untuk membuat objek dari kelas *contask*, proses ini dinamakan *Constructor*.

```
dataYangDikirim[0] = '0';
dataYangDikirim[1] = '0';
dataYangDikirim[2] = '0';
dataYangDikirim[3] = ';';
```

Untuk format data yang dikirimkan menggunakan format data berupa karakter yang akan mengatur nyala dan matinya lampu pada perangkat hardware. Yang dimaksud paket data adalah variabel array yang berisi elemen untuk menyatakan status lampu. Setiap pengiriman paket data tersebut selalu diakhiri dengan karakter ";" yang bertujuan untuk mereset *index buffer* pada mikrokontroler. Lampu nyala depresentasikan dengan karakter "1" dan untuk mematikan lampu dipresentasikan dengan krakter "0". Contohnya misalkan terdiri dari tiga lampu yang masing-masing kita beri karakter "0" jadi identitas lampu yang mati adalah "000;" untuk setiap masing-masing lampu. Jika ingin menyalakan lampu pertama maka data yang dikirimkan adalah "100;". Jika menyalakan lampu ke dua "010;" dan seterusnya untuk lampu ketiga.

```
final Button lamp1 = (Button) findViewById(R.id.btnLamp1);
```

Perintah untuk mengintegerasikan *Widget* yang ada pada *file .xml* dengan *script* utama

```
lamp1.setOnClickListener(new OnClickListener() {
@Override
public void onClick(View arg0) {
if (setLamp1Click == false) {
lamp1Pict.setImageResource(R.drawable.on);
    setLamp1Click = true;

    dataYangDikirim[0] = '1';
    dataYangDikirim[3] = ';';

} else {

    dataYangDikirim[0] = '0';
    dataYangDikirim[3] = ';';

    lamp1Pict.setImageResource(R.drawable.off);
    setLamp1Click = false;
    }
}
});
```

Membuat *listener* (pendengar) yang digunakan untuk menunggu *event* dari *user* yang berupa *event OnClick*. Jadi, ketika ada perintah "*click*" pada *button 1* maka *Method* (fungsi) *OnClick* akan dijalankan. Dalam method *OnClick* ini dimanfaatkan untuk meng-*update* nilai variabel *dataYangDikirim* dengan eleman "0" menjadi "1"yang artinya menyalakan lampu. Perintah "else" yang tertera mengartikan jika *button 1* ditekan sekali lagi maka nilai variabel kembali bernilai "0" yang artinya lampu akan kembali mati.

```
final ImageView lamp2Pict = (ImageView)
findViewById(R.id.lamp2);

final Button lamp2 = (Button) findViewById(R.id.btnLamp2);
lamp2.setOnClickListener(new OnClickListener() {

@Override
public void onClick(View arg0) {

if (setLamp2Click == false) {

    dataYangDikirim[1] = '1';
    dataYangDikirim[3] = ';';

    lamp2Pict.setImageResource(R.drawable.on);
    setLamp2Click = true;
} else {

    dataYangDikirim[1] = '0';
```

```
    dataYangDikirim[3] = ';';

    lamp2Pict.setImageResource(R.drawable.off);
    setLamp2Click = false;
    }
}
});

final ImageView lamp3Pict = (ImageView)
findViewById(R.id.lamp3);

final Button lamp3 = (Button) findViewById(R.id.btnLamp3);
lamp3.setOnClickListener(new OnClickListener() {

@Override
public void onClick(View arg0) {

if (setLamp3Click == false) {

    dataYangDikirim[2] = '1';
    dataYangDikirim[3] = ';';

    lamp3Pict.setImageResource(R.drawable.on);
    setLamp3Click = true;
} else {

    dataYangDikirim[2] = '0';
    dataYangDikirim[3] = ';';

    lamp3Pict.setImageResource(R.drawable.off);
    setLamp3Click = false;
    }
}
});
```

Untuk penjelasan *button* lampu ke-2 dan ke-3 sama seperti *button* pertama.

```
final ToggleButton btnConnected = (ToggleButton)
findViewById(R.id.btnConnected);
btnConnected.setOnClickListener(new OnClickListener() {

@Override
public void onClick(View arg0) {
if (btnConnected.isChecked()) {
    try {
          if (runningState) {
                contask.execute("");
          }
    tvState.setText("Connected to :" + mTcpClient.SERVERIP + "
    Port : 5000");
```

Perintah *toogle button* yang digunakan adalah untuk melakukan koneksi perangkat keras berdasarkan *IP* dan *Port* yang telah ditentukan menggunakan protokol *TCP/IP*. Perlu diketahui bahwa *WizFi*-lah yang dikonfigurasikan sebagai servernya dan penentuan menggunakan protokol komunikasi *TCP/IP* dikonfigurasikan didalamnya. Jadi, dari *handphone* hanya mengikuti konfigurasi yang ditetapkan oleh perangkat.

```
lamp1.setEnabled(true);
lamp2.setEnabled(true);
lamp3.setEnabled(true);

runningState = false;

} catch (Exception e) {
tvState.setText("State : " + e.getMessage());
btnConnected.setChecked(false);
}
```

Ketika proses koneksi berhasil, maka *widget button* lampu 1 sampai lampu 3 yang masing-masing secara berturut-turut telah diberi nama *lamp1, lamp2* dan *lamp3* akan di-*enable*-kan terlebih dahulu agar dapat dikontrol.

```
timer = new Timer();
timer.schedule(new TimerTask() {

@Override
public void run() {
    try {
    mTcpClient.sendMessage(dataYangDikirim);
    } catch (Exception ex) {
    ex.getMessage();
    }
}
}, 0, 300);
```

Membuat timer yang digunakan untuk melakukan data secara terus-menerus selama 300ms dan hanya dapat dihentikan bila *ToggleButton* (*btnConnected*) di-*Disconnect*-kan (diputuskan).

```
} else {
tvState.setText("Disconnected");
```
//instruksi untuk stop timer//
```
    try {
    timer.cancel();
    timer.purge();
    } catch (Exception ex) {
    ex.toString();
    }
```
//disable *button* ketika koneksi dengan perangkat keras ditutup//
```
lamp1.setEnabled(false);
lamp2.setEnabled(false);
lamp3.setEnabled(false);
}
}
});
```
//dibawah ini masih termasuk di dalam *method onCreate* yang difungsikan untuk menginisialisasi *btnLamp1*, *lamp3* ke keadaan *disable* kecuali telah dilakukan proses koneksi kembali//
```
lamp1.setEnabled(false);
lamp2.setEnabled(false);
lamp3.setEnabled(false);
}
```

Jika status *btnConnected* berubah menjadi *disconnect* maka *statement* inilah yang akan dijalankan.

Kemudian berikut ini adalah *Method* / *Fungsi* yang digunakan untuk *parsing* (pemisahan) data yang dikirim oleh perangkat keras. Format perngiriman data yang dikirimkan oleh perangkat keras adalah : (nilai sensor orang) ; (nilai sensor gas).

Contohnya, 1;124 artinya bahwa ada orang didalam suatu ruangan dan nilai sensor gas adalah 124 (desimal). Dengan keterangan sensor orang adalah bernilai "1" jika ada orang atau objek yang terdeteksi dan bernilai "0" ketika tidak mendeteksi apapun.

```
public void updateValue(String bs) {
String[] items = bs.split(";");
TextView txDataDiterima = (TextView) findViewById(R.id.tvSuhu);
ImageView gasPict = (ImageView) findViewById(R.id.gasPict);
TextView txStateGas = (TextView) findViewById(R.id.stateGas);

txDataDiterima.setText("Ada Orang : " + items[0]);

    try {
```

//instruksi untuk melakukan konversi data dari nilai *string* ke nilai *integer* (bilangan bulat)//

```
            int gasVal = Integer.parseInt(items[1]);
```

//instruksi untuk melakukan proses peringatan berdasarkan level gas. Jika masih dalam *range* 0-25, maka status masih aman. Jika dalam *range* 26-100, maka status menjadi siaga. Jika dalam range 101-255, maka status bahaya gas rawan meledak//

```
            if ((gasVal > 0) && (gasVal <25)) {
            gasPict.setBackgroundResource(R.drawable.aman_gas);
            txStateGas.setText("Aman!");
    } else if ((gasVal >= 26) && (gasVal < 100)) {
    gasPict.setBackgroundResource(R.drawable.bahaya_gas1);
    txStateGas.setText("Bahaya!");
    } else if ((gasVal >= 101) && (gasVal <= 255)) {
    gasPict.setBackgroundResource(R.drawable.bahaya_gas);
    txStateGas.setText("Bahaya!!!");
```

//jika nilai gas sudah pada tingkatan bahaya, maka akan membunyikan alarm peringatan menggunakan kelas *soundpool*//

```
    soundPool.play(1, leftVolume, rightVolume, priority,
    no_loop,normal_playback_rate);
    }
    } catch (Exception ex) {
    ex.getMessage();

    }
}
```

//*method* yang akan dijalankan ketika *user* menekan tombol menu yang difungsikan untuk mengatur *IP* dan keterangan aplikasi//

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
getMenuInflater().inflate(R.menu.main, menu);
return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {

    switch (item.getItemId()) {
    case R.id.iSettings:
    Intent i = new Intent(this, Preferences.class);
    startActivity(i);
    return true;
    case R.id.iAbout:
    try {
    PackageInfo manager = getPackageManager().getPackageInfo(
    getPackageName(), 0);
    Toast toast = Toast.makeText(this,

    this.getString(R.string.app_name) + " " +
    manager.versionName, 1000);
    toast.show();
    } catch (Exception e) {

    }
```

```
return true;
default:
return super.onOptionsItemSelected(item);
}
}
```

Kelas yang digunakan untuk melakukan *multi threading* dengan *AsyncTask*. Konsep ini melakukan *Task* (pekerjaan) dikerjakan secara paralel dimana pekerjaan ini dilakukan pada *background* tanpa mengiterupsi program utama yang berjalan.

```
public class connectTask extends AsyncTask<String, String,
TCPClient> {

    @Override
    protected TCPClient doInBackground(String... message) {
//membuat TCPClient Object dari kelas TCPClient//
    mTcpClient = new TCPClient(new
    TCPClient.OnMessageReceived() {
          @Override
          public void messageReceived(String message) {
          publishProgress(message);
          }
    });
    mTcpClient.run();

return null;
}

    @Override
    protected void onProgressUpdate(String... values) {
    super.onProgressUpdate(values);
//instruksi dalam Method onProgressUpdate akan dieksekusi jika ada
data yang masuk. Jika ada data yang masuk maka Method updateValue
akan dijalankan untuk melakukan prosedur parsing data yang diterima//
    updateValue(values[0]);

    }

    @Override
    protected void onCancelled() {
    super.onCancelled();
    mTcpClient.stopClient();
    }
}

}
```

Pada *TCPClient.java* berfungsi untuk menghubungkan kepada jaringan yang telah disediakan oleh *WizFi210* dengan memasukkan kelas *IP* dan *port* yang akan digunakan agar dapat secara langsung terhubung dengan perangkat keras. Pada *layer* ini kita memasukkan *script* berikut.

```
package com.example.wifi_control;

import android.util.Log;
import java.io.*;
import java.net.InetAddress;
import java.net.Socket;

public class TCPClient {

    private String serverMessage;
    public static final String SERVERIP = "192.168.1.1";
```

//192.168.1.1 merupakan *IP* dari *WizFi210* yang dikonfigurasikan sebagai *server*//

```
    public static final int SERVERPORT = 5000;
```

//5000 merupakan port yang dipakai oleh *server*//

```
    private OnMessageReceived mMessageListener = null;
    private boolean mRun = false;

    PrintWriter out;
    BufferedReader in;
```

//dibawah ini merupakan konstruktor kelas pada saat *Object* dibuat//

```
    public TCPClient(OnMessageReceived listener) {
           mMessageListener = listener;
    }
```

//*method* yang digunakan untuk proses pengiriman data//

```
    public void sendMessage(char[] data) {
           if (out != null && !out.checkError()) {
                  out.write(data);
                  out.flush();
           }
    }
```

//*method* yang dijalankan ketika *user* melakukan stop aplikasi//

```
    public void stopClient() {
           mRun = false;
    }

    public void run() {

           mRun = true;

           try {
```

//deklarasi alamat *IP* yang digunakan oleh alamat *server* (*WizFi210*)//

```
InetAddress serverAddr = InetAddress.getByName(SERVERIP);

Log.e("TCP Client", "C: Connecting...");
```

//membuat soket untuk dapat melakukan komunikasi dengan server (*WizFi*), protokol yang digunakan adalah *TCP*//

```
Socket socket = new Socket(serverAddr, SERVERPORT);

        try {
```

//instruksi untuk konfigurasi *PrintWriter* out agar dapat digunakan untuk menampung data yang dikirimkan ke *Socket TCP*//

```
out = new PrintWriter(new BufferedWriter(new o
utputStreamWriter(socket.getOutputStream())), true);

        Log.e("TCP Client", "C: Sent.");

        Log.e("TCP Client", "C: Done.");
```

//instruksi untuk konfigurasi *BufferReader* yang digunakan untuk menampung data yang dikirim ke *Socket TCP*//

```
in = new BufferedReader(new InputStreamReader
(socket.getInputStream()));
```

//selama *mRun=true* maka *statement* dalam *while* akan dijalankan terus (*looping*)//

```
while (mRun) {
serverMessage = in.readLine();
```

//membuat *listener* menunggu jika ada data yang masuk ke dalam soket *TCP*//

```
if (serverMessage != null && mMessageListener != null) {
        mMessageListener.messageReceived(serverMessage);
        }
```

//reset pesan dari *server*//

```
serverMessage = null;
}

Log.e("RESPONSE FROM SERVER", "S: Received Message: '"
+ serverMessage + "'");

} catch (Exception e) {

        Log.e("TCP", "S: Error", e);
```

//menutup soket//

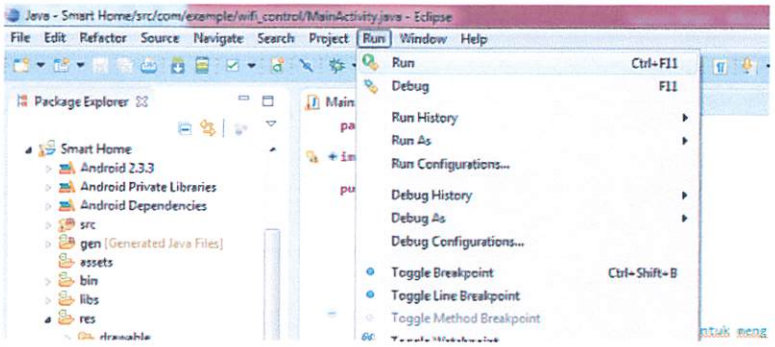
```
} finally {
        socket.close();
        }
} catch (Exception e) {
        Log.e("TCP", "C: Error", e);
        }
}
```

//mendeklarasikan *interface*. Instruksi ini akan dipanggil pada kelas *AsyncTask doInBackground* yang berada pada kelas utama *MainActivity.java*//

```
    public interface OnMessageReceived {
            public void messageReceived(String message);
    }
}
```

4.2. Pengujian Aplikasi

Jika semua *script* telah dimasukkan dan telah siap, maka hal pertama kita *run* untuk melihat tampilan dalam *emulator* android yang ada pada *Eclipse* dengan cara pilih *Run>Run* pada tools yang berada di bagian atas jendela. Atau kita bisa menekan *Ctrl+F11* pada *keyboard.*

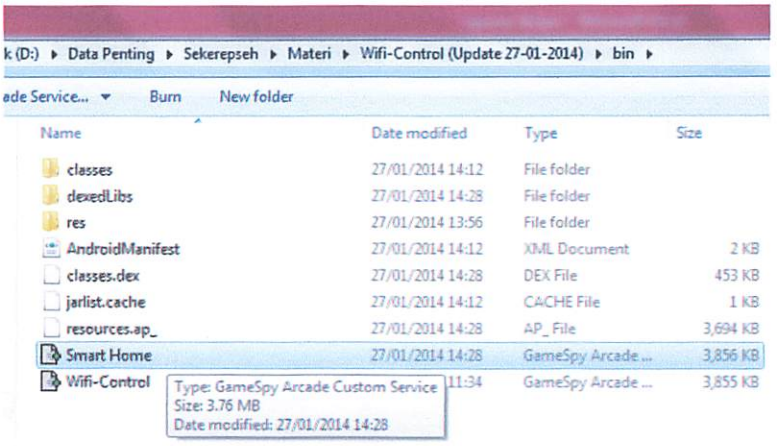


Gambar 4.6. *Running* Aplikasi pada Eclipse

Ketika sudah, maka akan keluar jendela baru hasil *running* tadi seperti pada gambar 4.7.

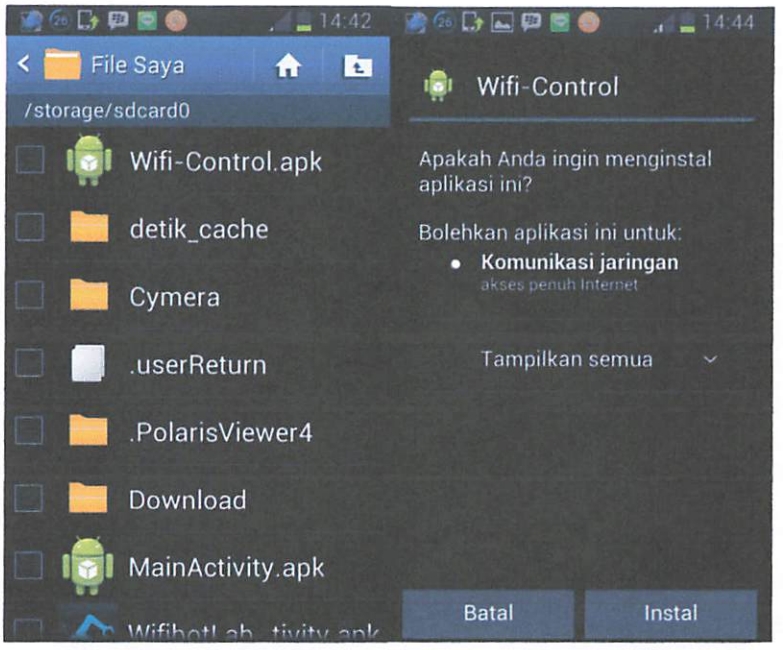
Gambar 4.7. Hasil tampilan *running* pada *emulator* android

Hasil *running* tadi adalah tanda bahwa kita bisa menginstal aplikasi ke *smart phone*. Langkah selanjutnya tujulah *work space* atau tempat kita mengerjakan aplikasi. Tujulah *folder bin* yang ada di dalamnya kemudian *copy* ke dalam *smart phone*, karena nanti pada *smart phone* lah aplikasi ini di instal.

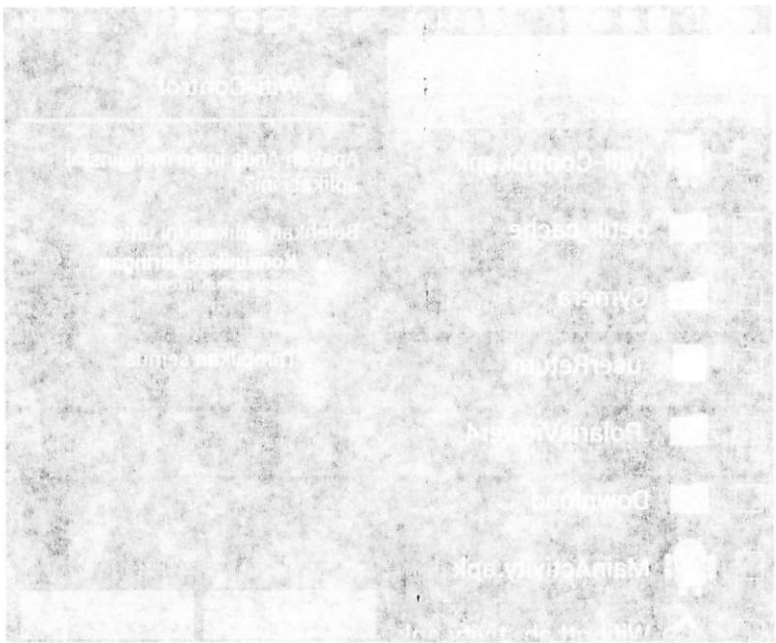


Gambar 4.8. Aplikasi yang di *copy* ke *smartphone*

Setelah file terkirim, buka file pada *smart phone* kemudian klik dan instal kemudian buka.

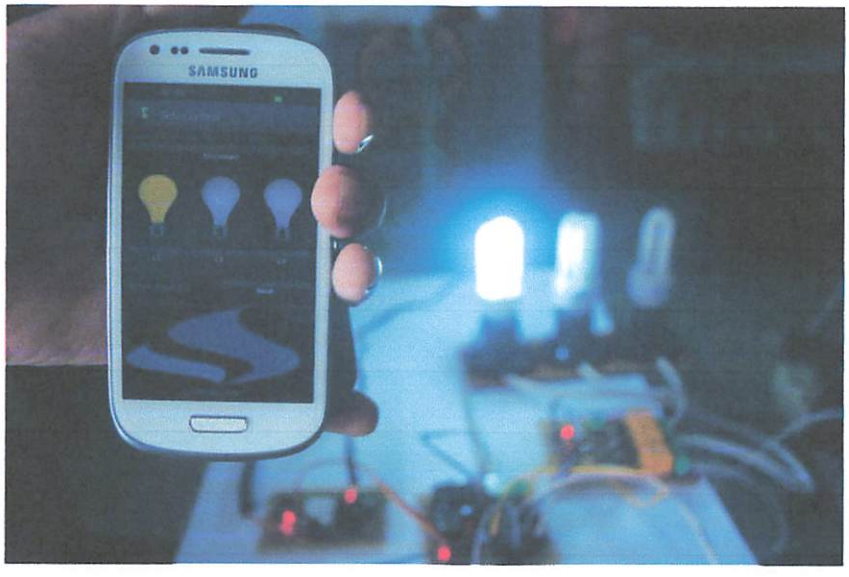


Gambar 4.9. Instalasi aplikasi pada *smartphone*

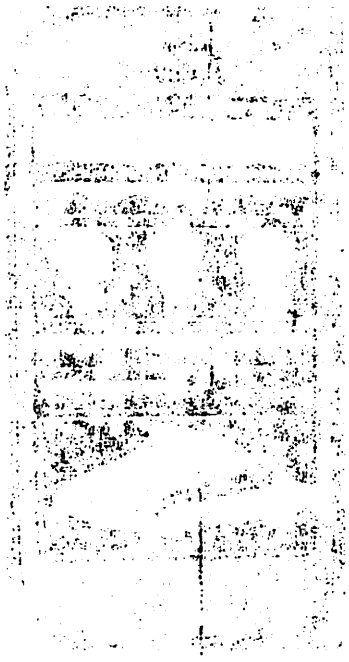


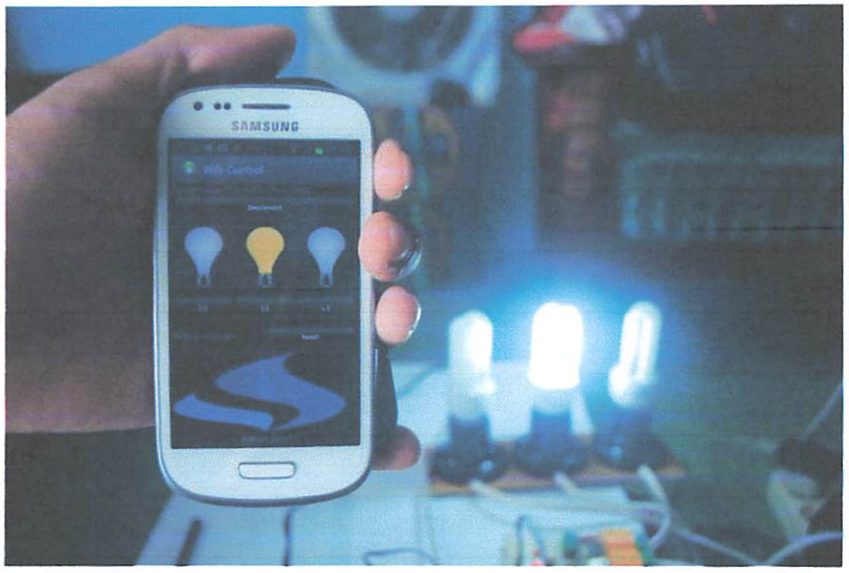
Gambar 4.10. Tampilan hasil instalasi aplikasi

Setelah terinstal dengan baik, tinggal kita buka dan coba dengan *hardware* yang telah dibuat.

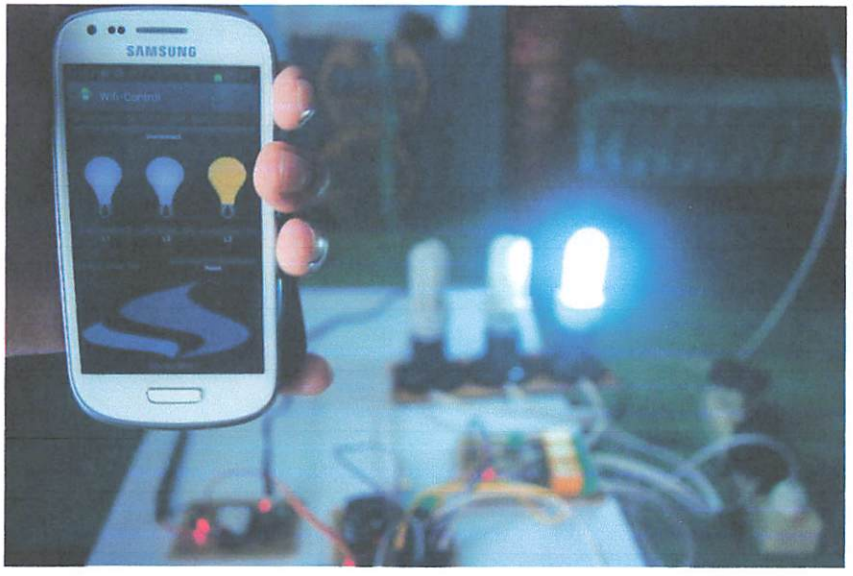


Gambar 4.11. Menyalakan lampu ke-1 melalui smart phone

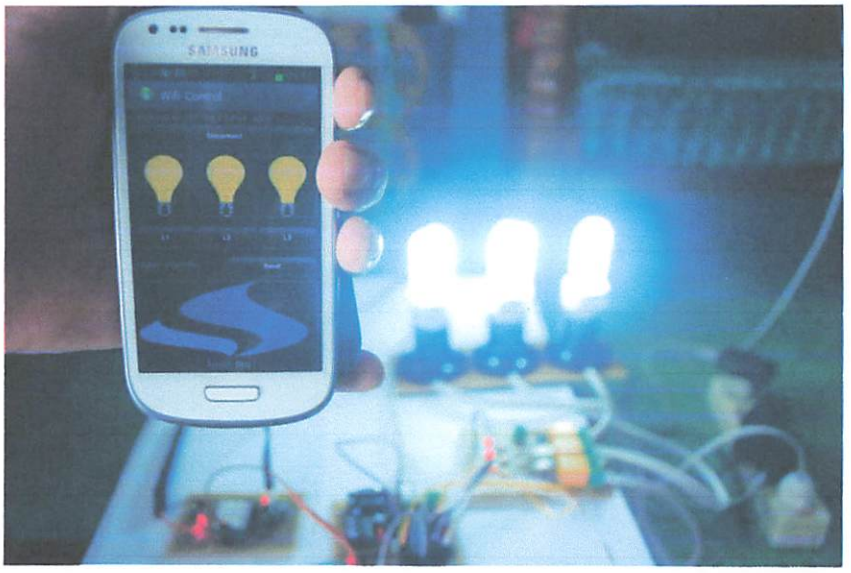




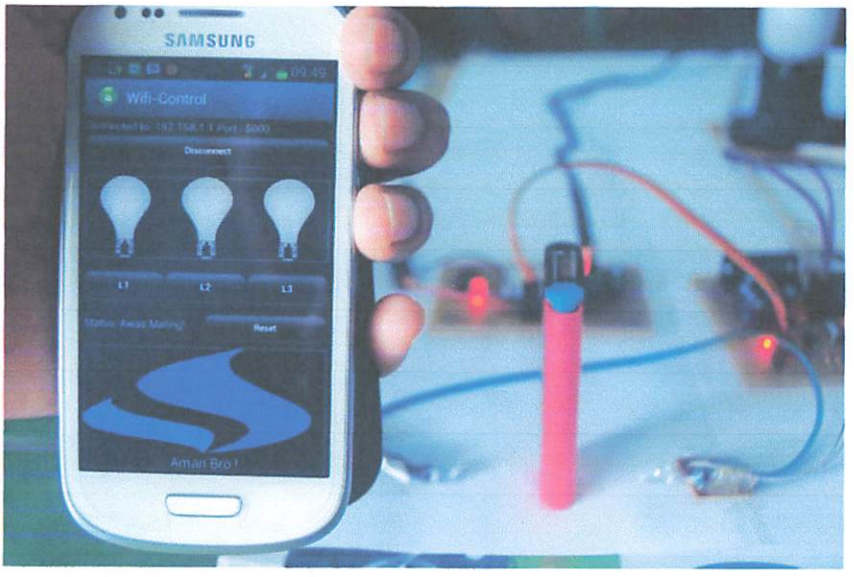
Gambar 4.12. Menyalakan lampu ke-2 melalui *smartphone*



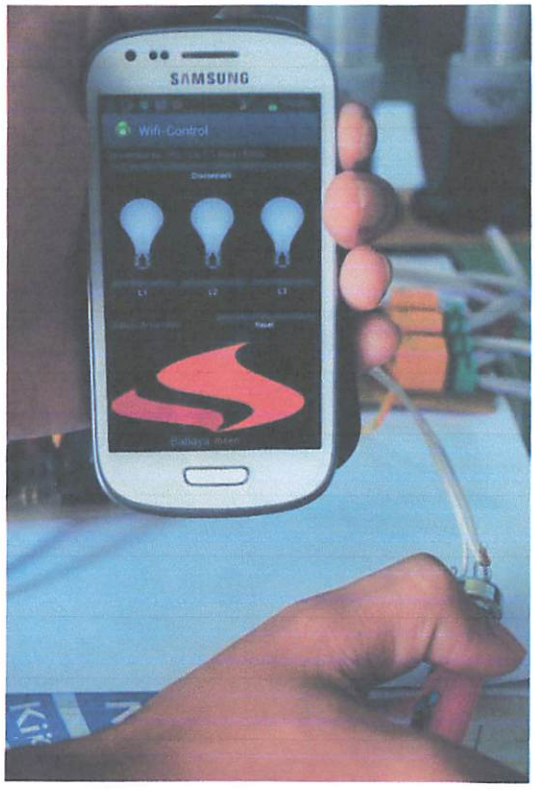
Gambar 4.13. Menyalakan lampu ke-3 melalui *smartphone*



Gambar 4.14. Menyalakan ketiga lampu bersamaan melalui *smartphone*



Gambar 4.15. Alarm yang muncul ketika sensor *photodioda* merasakan gangguan



Gambar 4.16. Tampilan ketika sensor *MQ-6* merasakan gas

Tabel 4.1.

Keadaan lampu terhadap kontrol melalui *hand phone*

Konektifitas	Kondisi Tombol Lampu	Kondisi Tombol			Kondisi Lampu		
		1	2	3	1	2	3
Terputus	Tidak dapat ditekan	X	X	X	O	O	O
Terhubung	Dapat ditekan	O	O	I	O	O	I
		O	I	O	O	I	O
		O	I	I	O	I	I
		I	O	O	I	O	O
		I	O	I	I	O	I
		I	I	O	I	I	O
		I	I	I	I	I	I

Keterangan :

Pada Tombol Lampu :
X = Tombol tidak dapat ditekan
I = Tombol ditekan
O = Tombol tidak ditekan

Pada Lampu:
I = Kondisi Lampu menyala
O = Kondisi Lampu mati

Tabel 4.2.

Keadaan *hand phone* pada saat terhubung dengan perangkat keras dan kondisi sensor aktif

Sensor	Keadaan		Alarm		
			Teks	Warna Gambar	Suara
Photodioda	Tidak mendeteksi pergerakan		Status : Aman	-	Non-aktif
	Pergerakan terdeteksi		Status : Awas Maling !	-	Aktif
MQ-6	Tidak mendeteksi konsentrasi gas		Aman !	Biru / Biru	Non-aktif
	Konsentrasi gas terdeteksi	Rendah	Aman !	Biru / Biru	Non-aktif
		Medium	Waspada !	Merah / Merah	Aktif

Keterangan :
Nilai konsentrasi gas adalah 8 bit (1 byte)
Rendah = 0 ppm – 999 ppm
Waspada = 1000 ppm – 10.000 ppm

Tabel 4.3.

Hasil percobaan tanpa penghalang antara sumber sinyal dengan penerima sinyal

Jenis *handphone*	Versi Android	Jenis Percobaan		Jarak (meter)													
				1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Samsung Galaxy SIII mini	4.1 (*JellyBean*)	Lampu		X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
		Gas LPG	Gambar	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
			Suara	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
		Sensor Photodioda	Text	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
			Suara	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
	4.2 (*JellyBean*)	Lampu		I	I	I	I	I	I	I	I	I	I	I	I	I	I	I
		Gas LPG	Gambar	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I
			Suara	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I
		Sensor Photodioda	Text	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I
			Suara	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I
	4.3 (*JellyBean*)	Lampu		I	I	I	I	I	I	I	I	I	I	I	I	I	I	I
		Gas LPG	Gambar	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I
			Suara	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I
		Sensor Photodioda	Text	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I
			Suara	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I
Samsung Galaxy Ace 3 GT-S7270	4.1 (*JellyBean*)	Lampu		X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
		Gas LPG	Gambar	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
			Suara	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
		Sensor Photodioda	Text	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
			Suara	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
	4.2 (*JellyBean*)	Lampu		I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
		Gas LPG	Gambar	I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
			Suara	I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
		Sensor Photodioda	Text	I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
			Suara	I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
	4.3 (*JellyBean*)	Lampu		I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
		Gas LPG	Gambar	I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
			Suara	I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
		Sensor Photodioda	Text	I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
			Suara	I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
	4.1 (*JellyBean*)	Lampu		I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
		Gas LPG	Gambar	I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
			Suara	I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
		Sensor Photodioda	Text	I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
			Suara	I	I	I	I	I	I	I	I	I	I	I	I	X	X	X

Tabel 4.2

Kesalahan dalam proses data [illegible] dengan perangkat keras dan kondisi

[illegible]

[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

[illegible]

[illegible] [illegible] [illegible] [illegible] [illegible] [illegible] [illegible]

[illegible] [illegible] [illegible] [illegible] [illegible] [illegible] [illegible]

[illegible] [illegible] [illegible] [illegible] [illegible] [illegible] [illegible]

Tabel 4.3

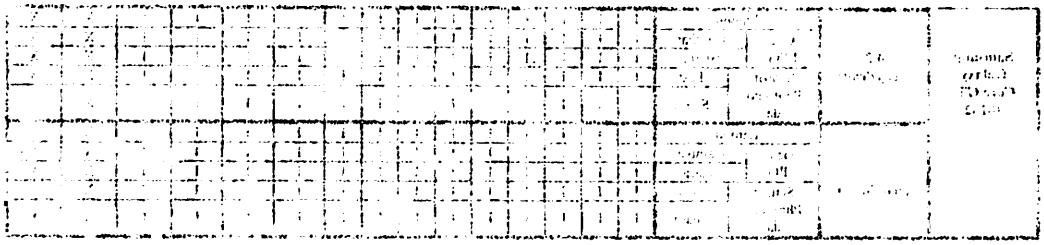
[illegible] [illegible] [illegible] [illegible] [illegible] [illegible] [illegible]

Samsung Galaxy Core GT-18262	4.2 (*JellyBean*)	Lampu		I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
		Gas LPG	Gambar	I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
			Suara	I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
		Sensor Photodioda	Text	I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
			Suara	I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
	4.3 (*JellyBean*)	Lampu		I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
		Gas LPG	Gambar	I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
			Suara	I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
		Sensor Photodioda	Text	I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
			Suara	I	I	I	I	I	I	I	I	I	I	I	I	I	X	X

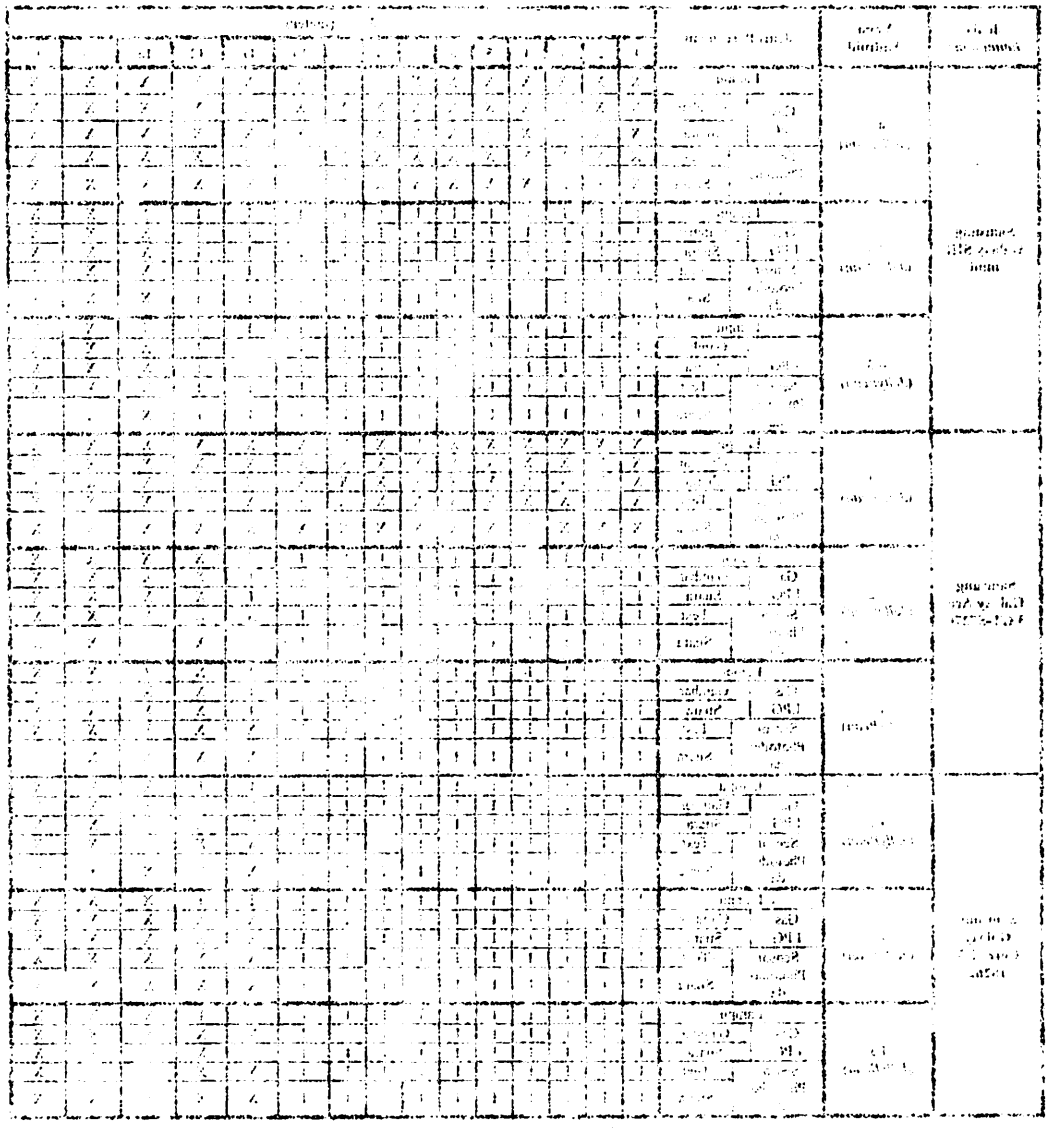
Tabel 4.4.

Hasil percobaan dengan penghalang pintu antara sumber sinyal dengan penerima sinyal

Jenis *handphone*	Versi Android	Jenis Percobaan		Jarak (meter)														
				1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Samsung Galaxy SIII mini	4.1 (*JellyBean*)	Lampu		X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
		Gas LPG	Gambar	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
			Suara	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
		Sensor Photodioda	Text	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
			Suara	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
	4.2 (*JellyBean*)	Lampu		I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
		Gas LPG	Gambar	I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
			Suara	I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
		Sensor Photodioda	Text	I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
			Suara	I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
	4.3 (*JellyBean*)	Lampu		I	I	I	I	I	I	I	I	I	I	I	I	I	X	X
		Gas LPG	Gambar	I	I	I	I	I	I	I	I	I	I	I	I	X	X	X
			Suara	I	I	I	I	I	I	I	I	I	I	I	I	X	X	X
		Sensor Photodioda	Text	I	I	I	I	I	I	I	I	I	I	I	I	X	X	X
			Suara	I	I	I	I	I	I	I	I	I	I	I	I	X	X	X
Samsung Galaxy Ace 3 GT-S7270	4.1 (*JellyBean*)	Lampu		X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
		Gas LPG	Gambar	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
			Suara	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
		Sensor Photodioda	Text	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
			Suara	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
	4.2 (*JellyBean*)	Lampu		I	I	I	I	I	I	I	I	I	I	I	X	X	X	X
		Gas LPG	Gambar	I	I	I	I	I	I	I	I	I	I	I	X	X	X	X
			Suara	I	I	I	I	I	I	I	I	I	I	I	X	X	X	X
		Sensor Photodioda	Text	I	I	I	I	I	I	I	I	I	I	X	X	X	X	X
			Suara	I	I	I	I	I	I	I	I	I	I	X	X	X	X	X
	4.3 (*JellyBean*)	Lampu		I	I	I	I	I	I	I	I	I	I	I	X	X	X	X
		Gas LPG	Gambar	I	I	I	I	I	I	I	I	I	I	I	X	X	X	X
			Suara	I	I	I	I	I	I	I	I	I	I	I	X	X	X	X
		Sensor Photodioda	Text	I	I	I	I	I	I	I	I	I	I	I	X	X	X	X
			Suara	I	I	I	I	I	I	I	I	I	I	I	X	X	X	X
Samsung Galaxy Core GT-18262	4.1 (*JellyBean*)	Lampu		I	I	I	I	I	I	I	I	I	I	I	I	X	X	X
		Gas LPG	Gambar	I	I	I	I	I	I	I	I	I	I	X	X	X	X	X
			Suara	I	I	I	I	I	I	I	I	I	I	X	X	X	X	X
		Sensor Photodioda	Text	I	I	I	I	I	I	I	I	I	I	X	X	X	X	X
			Suara	I	I	I	I	I	I	I	I	I	I	X	X	X	X	X
	4.2 (*JellyBean*)	Lampu		I	I	I	I	I	I	I	I	I	I	I	I	X	X	X
		Gas LPG	Gambar	I	I	I	I	I	I	I	I	I	I	X	X	X	X	X
			Suara	I	I	I	I	I	I	I	I	I	I	X	X	X	X	X
		Sensor Photodioda	Text	I	I	I	I	I	I	I	I	I	I	X	X	X	X	X
			Suara	I	I	I	I	I	I	I	I	I	I	X	X	X	X	X
	4.3 (*JellyBean*)	Lampu		I	I	I	I	I	I	I	I	I	I	I	I	X	X	X
		Gas LPG	Gambar	I	I	I	I	I	I	I	I	I	I	X	X	X	X	X
			Suara	I	I	I	I	I	I	I	I	I	I	X	X	X	X	X
		Sensor Photodioda	Text	I	I	I	I	I	I	I	I	I	I	X	X	X	X	X
			Suara	I	I	I	I	I	I	I	I	I	I	X	X	X	X	X



Tabel 4.4



Melihat hasil dari percobaan pada tabel 4.3 dan 4.4, dapat diketahui bahwa kualitas perangkat mempengaruhi jarak dari komunikasi nirkabel. Pada tabel 4.3, perangkat telepon genggam *Samsung SIII mini*-lah yang mempunyai kemampuan yang paling baik dalam hal komunikasi nirkabel. Karena dipengaruhi oleh spesifikasi komunikasi yang terbilang lebih unggul jika dibandingkan dengan kedua perangkat percobaan lainnya, untuk lebih jelasnya dapat dilihat pada situs resmi telepon genggam samsung www.samsung.com. Untuk versi android dari telepon genggam *Samsung SIII mini* dan *Samsung Ace 3 GT-S7270* tidak dapat menggunakan versi *JellyBean* 4.1 karena versi asli dari kedua telepon genggam tersebut adalah versi 4.2. Perlu diketahui, setiap perangkat dapat ditingkatkan (*upgrade*) versi androidnya ke versi-versi yang berada di atasnya, namun tidak bisa diturunkan (*downgrade*). Serta mengapa versi android tidak terlalu berpengaruh terhadap jarak, karena versi androidnya masih dalam satu kategori yaitu *JellyBean*.[12]

Kemudian untuk penentuan titik "*waspada*" pada konsentrasi gas, jika nilai gas mencapai nilai 1000 ppm dan dihirup secara terus – menerus dapat mengakibatkan mengantuk, mimpi hingga meninggal.[15]

Tabel 4.5.

Hasil percobaan sensitifitas sensor gas dengan sumber gas dalam ruangan normal dan letak kebocoran gas mengarah ke sensor

	Jarak sumber gas dengan sensor (sentimeter)																			
	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
Sensor MQ-6	I	I	I	I	I	I	I	I	I	I	I	I	I	I	X	X	X	X	X	X

Keterangan : Sumber gas yang digunakan adalah korek gas yang menggunakan gas *butane*.
I = mendeteksi gas
X = tidak dapat mendeteksi

Tabel 4.6.

Hasil percobaan sensitifitas sensor *Photodioda* dan cahaya *LED*

	Jarak sumber gas dengan sensor (sentimeter)																			
	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45
Photodioda	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	X	X

Keterangan : I = dapat menerima cahaya *LED*
X = tidak dapat menerima cahaya *LED*

Sedangkan hasil dari pengujian pada tabel 4.3 dan 4.4 dapat dikatakan bahwa kualitas pengiriman informasi pada alat komunikasi infrared radio tabel 4.3 pengujian telepon menggunakan Samsung SGH-A636 yang memproduksi komunikasi yang paling baik dalam hal komunikasi nirkabel. Karena dipengaruhi oleh spesifikasi komunikasi yang terdukung lebih unggul jika dibandingkan dengan kedua pemakaian percobaan lainnya, untuk lebih jelasnya dapat dilihat pada tabel 4.4 pengujian Samsung SGH-A636 dan Siemens C55-SL75 tidak dapat menggunakan versi Software 4.1 karena versi asli dari kedua telepon genggam tersebut adalah versi 4.2. Perlu diketahui, ketika proses pengujian dapat di lihat/dikatakan (signifikan) versi Software ke versi-versi yang berada di bawah nomor versi diturunkan (downgrade). Sama halnya versi software di atas nomor versi tersebut terhadap jarak koneksi versi software maka dalam satu kategori yang sama.[4]

Kemudian untuk pengujian alat "receiver" pada komunikasi gps jika mati atau mengalami nilai 1000 pun dan dibiarkan secara terus - menerus dapat mengakibatkan pengiriman yang tertunda.[5]

Tabel 4.5

Hasil pengujian sensitifitas receiver pengiriman number gps dalam keadaan normal dan letak koneksi gps menyerah ke server

Keterangan : Simbol yang digunakan adalah koneksi gps yang menyimulasikan gps receiver.
I = menghasilkan gps
X = tidak dapat menerima

Tabel 4.6

Hasil pengujian sensitifitas sensor receiver dari tabel 4.5

Keterangan : I = dapat menerima sinyal
X = tidak dapat menerima sinyal

BAB V
KESIMPULAN DAN SARAN

5.1. Kesimpulan

Berdasarkan dari hasil pengujian aplikasi *home monitoring* berbasis android, dapat di ambil beberapa kesimpulan sebagai berikut :

1. Jangkauan *Wi-Fi* tergantung dari merk *handphone* serta tergantung dari adanya penghalang yang terdapat antara sumber sinyal dengan *handphone*.
2. Alat ini dapat diaplikasikan untuk pengendali lampu, monitoring sensor photodioda dan konsentrasi gas LPG.
3. Terdapat kelemahan dari sensor *photodioda* antara lain jarak cahaya yang terbatas hanya sampai 43 cm.

5.2. Saran

Untuk para pengembang aplikasi yang ingin mengembangkan penelitian ini lebih lanjut, penulis menyarankan beberapa hal terkait skripsi ini, yaitu :

1. Dapat dikembangkan lagi pada sistem monitoring, tidak hanya gas LPG dan pencuri serta tidak hanya mengotrol lampu.
2. Dapat ditambahkan *database* pada aplikasi agar dapat melihat perubahan yang terjadi pada waktu lampau.
3. Tidak hanya dapat di aplikasikan dalam rumah tangga, namun di semua tempat yang memerlukan.

DAFTAR PUSTAKA

[1] Android Developer. (2007). Android Developer Guide. Retrieved July 2013, from http://developer.android.com.

[2] Anonim. MQ-6 Semiconductor Sensor for LPG. Hanwei Sensors. http://www.hwsensor.com.

[3] Anonim. 2005. *Fairchild Semiconductor Corporation QSD2030F Rev. 1.2.0.* http://www.fairchildsemi.com.

[4] Atmel®. 2013. *8-bit Atmel with 8Kbytes In-Programmable Flash Rev. 2486AA.02/2013.*

[5] F. D. Rumagit, J. O. Wuwung, S. R. U. A. Sompie, B. S. Narasiang. *Perancangan Sistem Switching 16 Lampu Secara Nirkabel Menggunakan Remote Control.* Manado : UNSRAT

[6] Hendrantoro, Gamantyo. 2008. "Teknologi komunikasi Nirkabel : Perkembangan Terkini dan Peluang Indonesia". Pidato Pengukuhan untuk Jabatan Guru Besar Dalam Bidang Sistem Komunikasi Nirkabel dan Propagasi Radio.

[7] Laksono, B., & Dedi, R. (2009). Mobile Application. Bandung: Politeknik Telkom.

[8] Raharjo, Budi., Heryanto, Herman., Haryanto, Arif. 2012. Mudah Belajar Java (Revisi Kedua). Informatika Bandung : Bandung.

[9] Safaat, N. S. (2011). Pemrograman Aplikasi Mobile Smartphone Dan Tablet PC. Bandung: Informatika.

[10] Samsono, Hadi M. Z. 2011. Performance & Monitoring Network. Surabaya: Institut Teknologi Sepuluh Nopember.

[11] Silva, Vladimir (11 March 2009). *Practical Eclipse Rich Client Platform Projects* (1st ed.). Apress. p. 352. ISBN 1-4302-1827-4.

[12] Song, Hee-Chan. Analysis of The Global Smart Phone Market and The Strategies of Its Major Player. University of Texas : Dallas.

[13] Wiznet Co. 2013. *WizFi210 Datasheet (version 1.20).* http://www.wiznet.co.kr.

[14] Yudistira, Yuan. 2011. *Membuat Aplikasi iPhone, Android, & Blackberry Itu Gampang.* Jakarta : mediakita.

[15] BPKN, *Data Kasus Ledakan Tabung Gas,* Jakarta, 2010.

LAMPIRAN

ГУЛЬБУТИ

Lampiran I

Script Pemrograman Eclipse

(*MainActivity.java*)

```
package com.example.wifi_control;

import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.net.InetSocketAddress;
import java.net.Socket;
import java.util.HashMap;
import java.util.Timer;
import java.util.TimerTask;

import android.R.bool;
import android.R.drawable;
import android.media.AudioManager;
import android.media.SoundPool;
import android.net.Uri;
import android.os.AsyncTask;
import android.os.Build;
import android.os.Bundle;
import android.os.Handler;
import android.os.Looper;
import android.os.Message;
import android.os.StrictMode;
import android.preference.PreferenceManager;
import android.app.Activity;
import android.content.BroadcastReceiver;
import android.content.ComponentName;
import android.content.ContentResolver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.content.IntentSender;
import android.content.ServiceConnection;
import android.content.SharedPreferences;
import android.content.IntentSender.SendIntentException;
import android.content.pm.ApplicationInfo;
import android.content.pm.PackageInfo;
import android.content.pm.PackageManager;
import android.content.pm.PackageManager.NameNotFoundException;
import android.content.res.AssetManager;
import android.content.res.Resources;
import android.content.res.Resources.Theme;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteDatabase.CursorFactory;
import android.graphics.Bitmap;
```

```
import android.graphics.drawable.Drawable;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.CompoundButton;
import android.widget.CompoundButton.OnCheckedChangeListener;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.TimePicker.OnTimeChangedListener;
import android.widget.Toast;
import android.widget.ToggleButton;

public class MainActivity extends Activity {

	private static int REFRESH_TIME = 300;

	private boolean setLamp1Click = false;
	private boolean setLamp2Click = false;
	private boolean setLamp3Click = false;

	private Timer timer;
	public byte x = 0;

	public char[] dataYangDikirim = new char[5];
	public byte[] dataYangDiterima = new byte[20];

	private Handler mHandler;

	public TCPClient mTcpClient;
	private Socket socket;

	private connectTask contask;
	private boolean runningState = true;

	float curVolume, maxVolume, leftVolume, rightVolume;
	int priority = 1;
	int no_loop = 0;
	float normal_playback_rate = 1f;

	SoundPool soundPool;

	HashMap<Integer, Integer> soundPoolMap;

	private String statusOrang = null;

	@Override
	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);

	if (Build.VERSION.SDK_INT>Build.VERSION_CODES.GINGERBREAD){
		StrictMode.ThreadPolicy policy = new
	StrictMode.ThreadPolicy.Builder().permitAll().build();
			StrictMode.setThreadPolicy(policy);
	}
```

```
setContentView(R.layout.activity_main);

AudioManager audioManager = (AudioManager)
getSystemService(Context.AUDIO_SERVICE);
curVolume =
audioManager.getStreamVolume(AudioManager.STREAM_MUSIC);
maxVolume =
audioManager.getStreamMaxVolume(AudioManager.STREAM_MUSIC);
leftVolume = curVolume / maxVolume;
rightVolume = curVolume / maxVolume;

soundPool = new SoundPool(30, AudioManager.STREAM_MUSIC, 100);
soundPoolMap = new HashMap<Integer, Integer>();

soundPoolMap.put(1, soundPool.load(this, R.drawable.alarm, 1));
soundPoolMap.put(2, soundPool.load(this, R.drawable.alarm_maling,
1));

final ImageView lamp1Pict = (ImageView) findViewById(R.id.lamp1);
final TextView tvState = (TextView) findViewById(R.id.status);

contask = new connectTask();

dataYangDikirim[0] = '0';
dataYangDikirim[1] = '0';
dataYangDikirim[2] = '0';
dataYangDikirim[3] = '0';
dataYangDikirim[4] = ';';

final Button lamp1 = (Button) findViewById(R.id.btnLamp1);
lamp1.setOnClickListener(new OnClickListener() {
      //
      @Override
      public void onClick(View arg0) {
      // TODO Auto-generated method stub

if (setLamp1Click == false) {
lamp1Pict.setImageResource(R.drawable.on);
setLamp1Click = true;

dataYangDikirim[0] = '1';
dataYangDikirim[4] = ';';

} else {

dataYangDikirim[0] = '0';
dataYangDikirim[4] = ';';


lamp1Pict.setImageResource(R.drawable.off);
setLamp1Click = false;
            }
      }
});
```

```
final ImageView lamp2Pict = (ImageView)
findViewById(R.id.lamp2);
final Button lamp2 = (Button) findViewById(R.id.btnLamp2);
lamp2.setOnClickListener(new OnClickListener() {

	@Override
	public void onClick(View arg0) {
		// TODO Auto-generated method stub

	if (setLamp2Click == false) {

	dataYangDikirim[1] = '1';
	dataYangDikirim[4] = ';';


lamp2Pict.setImageResource(R.drawable.on);
setLamp2Click = true;
} else {

dataYangDikirim[1] = '0';
dataYangDikirim[4] = ';';


lamp2Pict.setImageResource(R.drawable.off);
setLamp2Click = false;
		}
	}
});

final ImageView lamp3Pict = (ImageView) findViewById(R.id.lamp3);
final Button lamp3 = (Button) findViewById(R.id.btnLamp3);
lamp3.setOnClickListener(new OnClickListener() {

	@Override
	public void onClick(View arg0) {
		// TODO Auto-generated method stub

	if (setLamp3Click == false) {

	dataYangDikirim[2] = '1';
	dataYangDikirim[4] = ';';

lamp3Pict.setImageResource(R.drawable.on);
setLamp3Click = true;
} else {

dataYangDikirim[2] = '0';
dataYangDikirim[4] = ';';

lamp3Pict.setImageResource(R.drawable.off);
setLamp3Click = false;
		}
	}
});
```

```
final ToggleButton btnConnected = (ToggleButton)
findViewById(R.id.btnConnected);
btnConnected.setOnClickListener(new OnClickListener() {

	@Override
	public void onClick(View arg0) {
	// TODO Auto-generated method stub

	mTcpClient = new TCPClient();
	if (btnConnected.isChecked()) {
	try {
	if (runningState) {
	contask.execute("");
	}

tvState.setText("Connected to :" + mTcpClient.SERVERIP+ " Port :
5000");

lamp1.setEnabled(true);
lamp2.setEnabled(true);
lamp3.setEnabled(true);

runningState = false;
} catch (Exception e) {
tvState.setText("State : " + e.getMessage());
btnConnected.setChecked(false);
}
timer = new Timer();
timer.schedule(new TimerTask() {

@Override
public void run() {
// TODO Auto-generated method stub
try {

mTcpClient.sendMessage(dataYangDikirim);
dataYangDikirim[3] = '0';
} catch (Exception ex) {
ex.getMessage();

		}
	}
}, 0, REFRESH_TIME);
} else {

tvState.setText("Disconnected");

try {
	timer.cancel();
	timer.purge();
	mTcpClient.stopClient();
} catch (Exception ex) {
	ex.toString();
}
	lamp1.setEnabled(false);
	lamp2.setEnabled(false);
	lamp3.setEnabled(false);
```

```
                }
            }
        });

        Button btnReset = (Button) findViewById(R.id.btnReset);
        btnReset.setOnClickListener(new OnClickListener() {

        @Override
        public void onClick(View arg0) {
            // TODO Auto-generated method stub
            dataYangDikirim[3] = '1';
            }
        });

        lamp1.setEnabled(false);
        lamp2.setEnabled(false);
        lamp3.setEnabled(false);
}

public void updateValue(String bs) {
        String[] items = bs.split(";");

        TextView txDataDiterima = (TextView) findViewById(R.id.tvSuhu);
        ImageView gasPict = (ImageView) findViewById(R.id.gasPict);
        TextView txStateGas = (TextView) findViewById(R.id.stateGas);

        try {
        int orangVal = Integer.parseInt(items[0]);
        switch (orangVal) {
        case 0:
        statusOrang = "Aman Bro";
        break;
        case 1:
        statusOrang = "Awas Maling!";
        soundPool.play(2, leftVolume, rightVolume, priority, no_loop,
        normal_playback_rate);
        break;
        default:
        break;

        }
} catch (Exception ex) {
//
}

txDataDiterima.setText("Status: " + statusOrang);
try {
int gasVal = Integer.parseInt(items[1]);
if ((gasVal > 0) && (gasVal < 25)) {
gasPict.setBackgroundResource(R.drawable.aman_gas);
txStateGas.setText("Aman!");
} else if ((gasVal >= 26) && (gasVal < 100)) {
gasPict.setBackgroundResource(R.drawable.bahaya_gas1);
txStateGas.setText("Bahaya!");
} else if ((gasVal >= 101) && (gasVal <= 255)) {
gasPict.setBackgroundResource(R.drawable.bahaya_gas);
txStateGas.setText("Bahaya!!!");
```

```
soundPool.play(1, leftVolume, rightVolume, priority, no_loop,
normal_playback_rate);
        }
} catch (Exception ex) {
//
        }
}
public void Integratesocket(Socket socket) {
        this.socket = socket;
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
getMenuInflater().inflate(R.menu.main, menu);
return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {
switch (item.getItemId()) {
case R.id.iSettings:
Intent i = new Intent(this, Preferences.class);
startActivity(i);
return true;
case R.id.iAbout:
        try {
        PackageInfo manager = getPackageManager().getPackageInfo(
        getPackageName(), 0);
        Toast toast = Toast.makeText(this,
        this.getString(R.string.app_name) + " "
        + manager.versionName, 1000);
        toast.show();
        } catch (Exception e) {
        //
        }
        return true;
        default:
        return super.onOptionsItemSelected(item);
        }
}

        public class connectTask extends AsyncTask<String, String,
TCPClient> {

@Override
protected TCPClient doInBackground(String... message) {

        mTcpClient = new TCPClient(new TCPClient.OnMessageReceived() {
        @Override
        // here the messageReceived method is implemented
        public void messageReceived(String message) {
        // this method calls the onProgressUpdate
        publishProgress(message);
                }
        });
        mTcpClient.run();
```

```
        return null;
        }

@Override
protected void onProgressUpdate(String... values) {
        super.onProgressUpdate(values);

        updateValue(values[0]);

        }

@Override
protected void onCancelled() {
        // TODO Auto-generated method stub
        super.onCancelled();
        mTcpClient.stopClient();
        }
}
}
```

(*Preferences.java*)

```
package com.example.wifi_control;

import android.os.Bundle;
import android.preference.PreferenceActivity;

public class Preferences extends PreferenceActivity {

    @Override
    public void onCreate(Bundle savedInstanceState){
        super.onCreate(savedInstanceState);

        addPreferencesFromResource(R.xml.prefs);
    }
}
```

(*TCPclient.java*)

```
package com.example.wifi_control;

import android.util.Log;
import java.io.*;
import java.net.InetAddress;
import java.net.Socket;

public class TCPClient {

        private String serverMessage;
        public static final String SERVERIP = "192.168.1.1";
        public static final int SERVERPORT = 5000;    private
        OnMessageReceived mMessageListener = null;
        private boolean mRun = false;
        public Socket socket;
        public String msg;
```

```
	PrintWriter out;
	BufferedReader in;
	public TCPClient(OnMessageReceived listener) {
		mMessageListener = listener;
	}
	public TCPClient(){
		//
	}
	public void sendMessage(char[] data) {
		if (out != null && !out.checkError()) {
			out.write(data);
			out.flush();
		}
	}
	public void stopClient() {
		mRun = false;
	}
	public void run() {
		mRun = true;
		try {
		InetAddress serverAddr = InetAddress.getByName(SERVERIP);
		socket = new Socket(serverAddr, SERVERPORT);
		try {
		out = new PrintWriter(new BufferedWriter(
		new OutputStreamWriter(socket.getOutputStream())), true);
		in = new BufferedReader(new InputStreamReader(
		socket.getInputStream()));
		while (mRun) {
		serverMessage = in.readLine();
		if (serverMessage != null && mMessageListener != null) {
		mMessageListener.messageReceived(serverMessage);
		}
		serverMessage = null;
		}
		} catch (Exception e) {
		Log.e("TCP", "S: Error", e);
		} finally {
			// Close Socket
		socket.close();
		}
			} catch (Exception e) {
		Log.e("TCP", "C: Error", e);
		}
	}
	public interface OnMessageReceived {
		public void messageReceived(String message);
	}
}
```

INSTITUT TEKNOLOGI NASIONAL
FAKULTAS TEKNOLOGI INDUSTRI
JURUSAN TEKNIK ELEKTRO S-1
Jl. Raya Karanglo, Km. 2 MALANG

PERMOHONAN PERSETUJUAN SKRIPSI

Yang Bertanda Tangan Dibawah Ini:

Nama : Septian Tri Hargianto
N I M : 09.12.523
Semester : IX / Sembilan
Fakultas : Teknologi Industri
Jurusan : Teknik Elektro S-I
Konsentrasi : **TEKNIK ENERGI LISTRIK**
TEKNIK ELEKTRONIKA
TEKNIK KOMPUTER DAN INFORMATIKA
TEKNIK KOMPUTER (circled)
TEKNIK TELEKOMUNIKASI
Alamat : Perum. Permata Jingga Blok BB.15 no. 29

Dengan ini kami mengajukan permohonan untuk mendapatkan persetujuan untuk membuat SKRIPSI Tingkat Sarjana. Untuk melengkapi permohonan tersenut, bersama ini kami lampirkan persyaratan-persyaratan yang harus dipenuhi.
Adapun persyaratan- persyaratan pengambilan SKRIPSI adalah sebagai berikut:

1. Telah melaksanakan semua praktikum sesui dengan konsentrasinya (.....)
2. Telah lulus dan menyerahkan laporan Praktek Kerja (.....)
3. Telah lulus seluruh mata kuliah keahlian (MKB)sesuai konsentrasinya (.....)
4. Telah menempuh matakuliah> 134 sks dengan IPK > 2 dan tidak ada nilai E (.....)
5. Telah mengikuti secara aktif kegiatan seminar Skripsi yang diadakan Jurusan (.....)
6. Memenuhi persyaratan administrasi (.....)

Demikian permohonan ini untuk mendapatkan penyelesaian lebih lanjut dan atas perhatiannya kami ucapkan terima kasih.

Telah diteliti kebenarannya data tersebut diatas
Recording Teknik Elektro S-I

(.....)

Malang, 26 September 2013
Pemohon

(Septian .....)

Disetujui
Ketua Prodi Teknik Elektro S-I

M. Ibrahim Ashari, ST, MT
NIP. P. 1030100358

Mengetahui
Dosen Wali

(.....)

Catatan:
Bagi mahasiswa yang telah memenuhi persyaratan mengambil SKRIPSI agar membuat proposal dan mendapat persetujuan dari Ketua Prodi T. elektro S-I

1 A131.5 / 138 3.13
2 .....
3 .....

Form S-1a

BERITA ACARA RAPAT PERSETUJUAN JUDUL/PROPOSAL SKRIPSI

PROGRAM STUDI TEKNIK ELEKTRO S-1

. Konsentrasi : Teknik Komputer

nggal : 19 Oktober 2013

NIM	0912523
Nama	SEPTIAN TRI HARJANTO
Judul yang diajukan	APLIKASI SMART HOME MONITORING BERBASIS ANDROID
Disetujui/~~Ditolak~~	Ok
Catatan: - Di blok diagram ditambahkan pengontrolan lampu.	
Pembimbing yang diusulkan: 1. F. Yudi Limpraptono, ST. MT 2. Bima Aulia, ST	

Menyetujui

1. Koordinator Dosen Kelompok Keahlian

(Dr. Eng. Aryuanto Soetedjo, ST, MT)

2. Dosen Kelompok Keahlian (Terlampir)

: Coret yang tidak perlu

PROGRAM STUDI TEKNIK ELEKTRO S-1
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
Kampus II : Jl. Raya Karanglo Km. 2 Telp. (0341) 417636 Malang

PERNYATAAN KESEDIAAN DALAM PEMBIMBINGAN SKRIPSI

Sesuai permohonan dari mahasiswa/i :

Nama : **SEPTIAN TRI HARJANTO**
Nim : **0912523**
Semester : **IX (Sembilan)**
Jurusan : **Teknik Elektro S-1**
Konsentrasi : **Teknik Komputer**

Dengan ini menyatakan bersedia/tidak bersedia*) Membimbing skripsi dari mahasiswa tersebut, dengan judul :

" APLIKASI SMART HOME MONITORING BERBASIS ANDROID"

Demikian surat pernyataan ini kami buat agar dapat dipergunakan seperlunya.

Hormat Kami

Ir. F. Yudi Limpraptono, MT
Y.1039500274

*) Coret yang tidak perlu

PROGRAM STUDI TEKNIK ELEKTRO S-1
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
Kampus II : Jl. Raya Karanglo Km. 2 Telp. (0341) 417636 Malang

PERNYATAAN KESEDIAAN DALAM PEMBIMBINGAN SKRIPSI

Sesuai permohonan dari mahasiswa/i :

Nama	: **SEPTIAN TRI HARJANTO**
Nim	: **0912523**
Semester	: **IX (Sembilan)**
Jurusan	: **Teknik Elektro S-1**
Konsentrasi	: **Teknik Komputer**

Dengan ini menyatakan bersedia/~~tidak bersedia~~*) Membimbing skripsi dari mahasiswa tersebut, dengan judul :

" APLIKASI SMART HOME MONITORING BERBASIS ANDROID"

Demikian surat pernyataan ini kami buat agar dapat dipergunakan seperlunya.

Hormat Kami

Bima Aulia Firmandani, ST

Catatan :

Setelah disetujui agar formulir ini Diserahkan mahasiswa/I yang bersangkutan kepada jurusan untuk diproses lebih lanjut

*) Coret yang tidak perlu

PROGRAM STUDI TEKNIK ELEKTRO S-1
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG

Kampus II : Jl. Raya Karanglo Km. 2 Telp. (0341) 417636

FORMULIR BIMBINGAN SKRIPSI

Nim : 09.12.523
Nama : SEPTIAN TRI HARJANTO
Masa Bimbingan : Semester Ganjil - 2013/2014
Judul : APLIKASI HOME MONITORING DAN KONTROL BERBASIS ANDROID

No.	Tanggal	Uraian	Paraf
1	12 Nov 2013	Bimbingan Bab I, II, III	
2	22 Nov 2013	Bimbingan Bab IV	
3	24 Des 2013	Konsultasi Hardware	
4	21 Jan 2014	Konsultasi Bab V	
5	3 Feb 2014	Revisi Hasil Pengujian	
6	5 Feb 2014	Bimbingan Keseluruhan Skripsi	
7	5 Feb 2014	Presentasi Hardware & Skrip Aplikasi	
8			
9			
10			

Malang,
Dosen Pembimbing

Ir. F. Yudi Limpraptono, MT
Y.1039500274

PROGRAM STUDI TEKNIK ELEKTRO S-1
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG

Kampus II : Jl. Raya Karanglo Km. 2 Telp. (0341) 417636

FORMULIR BIMBINGAN SKRIPSI

Nim : 09.12.523
Nama : SEPTIAN TRI HARJANTO
Masa Bimbingan : Semester Ganjil / 2013-2014
Judul : APLIKASI HOME MONITORING DAN KONTROL BERBASIS ANDROID

No.	Tanggal	Uraian	Paraf
1	11 Nov 2013	Bimbingan BAB I, II, III	
2	20 Nov 2013	Bimbingan Bab IV	
3	22 Nov 2013	Revisi Bab I, II, III	
4	3 Des 2013	Bimbingan hardware	
5	13 Feb 2014	Konsultasi Kompre	
6	15 Feb 2014	Revisi Bab IV	
7	22 Mar 2014	Konsultasi Perbaikan Ujian Skripsi	
8	24 Mar 2014	Bimbingan Revisi Referensi Gas	
9			
10			

Malang,
Dosen Pembimbing

Bima Aulia Firmandani, ST

PERKUMPULAN PENGELOLA PENDIDIKAN UMUM DAN TEKNOLOGI NASIONAL MALANG
INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
FAKULTAS TEKNIK SIPIL DAN PERENCANAAN
PROGRAM PASCASARJANA MAGISTER TEKNIK
PT. BNI (Persero) MALANG
BANK NIAGA MALANG
Kampus I : Jl.Bendungan Sigura-gura No.2 Telp. (0341) 551431 (Hunting), Fax (0341) 553015 Malang 65145
Kampus II : Jl. Raya Karanglo, Km 2 Telp. (0341) 417636 Fax. (0341) 417634 Malang

BERITA ACARA UJIAN SKRIPSI
FAKULTAS TEKNOLOGI INDUSTRI

NAMA	: SEPTIAN TRI HARJANTO
NIM	: 09.12.523
JURUSAN	: TEKNIK ELEKTRO S-1
KONSENTRASI	: TEKNIK KOMPUTER
MASA BIMBINGAN	: SEMESTER GANJIL 2013/2014
JUDUL	: **APLIKASI HOME MONITORING DAN KONTROL BERBASIS ANDROID**

Dipertahankan dihadapan Tim Penguji Ujian Skripsi Jenjang Program Strata Satu (S-1) pada :

Hari : **Selasa**
Tanggal : **18 Februari 2014**
Dengan Nilai : **77,75 (B+)**

PANITIA UJIAN SKRIPSI

Ketua Jurusan Prodi

M. Ibrahim Ashari, ST, MT
NIP. Y. 1030100358

Sekertaris Jurusan Prodi

Dr. Eng. Aryuanto Soetedjo, ST, MT
NIP. Y. 1030800417

ANGGOTA PENGUJI

Dosen Penguji I

Dr. Eng. Aryuanto Soetedjo, ST, MT
NIP. Y. 1030800417

Dosen Penguji II

Ir. Eko Nurcahyo, MT
NIP. Y. 1028700172

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
PROGRAM STUDI TEKNIK ELEKTRO S-1
KONSENTRASI TEKNIK KOMPUTER

FORMULIR PERBAIKAN SKRIPSI

Dalam pelaksanaan ujian skripsi jenjang strata satu (S-1) jurusan T.Elektro konsentrasi Teknik Komputer, maka perlu adanya perbaikan skripsi untuk mahasiswa :

Nama : Septian Tri Harjanto
NIM : 09.12.523
Program Studi : Teknik Elektro S-1
Konsentrasi : Teknik Komputer
Judul Skripsi : Aplikasi Home Monitoring dan Kontrol Berbasis Android

No.	Penguji	Tanggal	Uraian	Paraf
1	Penguji I	18 Februari 2014	1. Monitor status lampu. 2. Referensi pengaturan status gas. 3. Analisis hasil pengujian. 4. Kata "smart" dihilangkan. 5. Pengujian terhadap OS yang berbeda pada setiap *handphone*.	
2	Penguji II	18 Februari 2014	1. Tambahkan batasan masalah untuk jenis *handohone*. 2. Kata "smart" pada judul dihilangkan. 3. Perbaiki kesimpulan Nomor 4. 4. Lakukan pengujian untuk jenis *handphone* yang sama dengan versi OS yang berbeda.	

Disetujui :

Dosen Penguji I

Dr. Eng. Aryuanto Soetedjo, ST, MT
NIP. Y. 1030800417

Dosen Penguji II

Ir. Eko Nurcahyo, MT
NIP. Y. 1028700172

Mengetahui :

Dosen Pembimbing I

Ir. F. Yudi Limpraptono, MT
Y. 1039500274

Dosen Pembimbing II

Bima Aulia Firmandani, ST

Persembahan

to God,

Thanks for giving me the joys, loves, hopes, sacrifices, pains, and all kind of the emotion. Without you, I'm nothing.

"اَللّٰهُ اَكْبَرُ"

27th March 2014

BIOGRAFI PENULIS

Septian Tri Harjanto dilahirkan pada tanggal 13 September 1991 di kota Balikpapan, Kalimantan Timur. Merupakan buah perkawinan terakhir dari tiga bersaudara dari pasangan bapak Sudjarno dan ibu Helena Yulia H, S.Pd, MM. Penulis memulai pendidikannya di TK/TPA At-Taqwa Balikpapan pada tahun 1995 dan lulus pada tahun 1997. Pada pertengahan tahun 1997 penulis melanjutkan pendidikannya di SD Kemala Bhayangkari I Balikpapan dan lulus pada tahun 2003. Kemudian melanjutkan jenjang pendidikan Sekolah Menengah Pertama dan Sekolah Menengah Atas berturut-turut di SMP Negeri I Balikpapan lulusan tahun 2006 dan SMA Patra Dharma Balikpapan lulusan tahun 2009. Hingga kini berhasil mendapatkan gelar S-1 nya pada perguruan tinggi Institut Teknologi Nasional Malang jurusan Teknik Elektro S-1 konsentrasi Teknik Komputer diawali pada pertengahan tahun 2009 hingga diwisuda pada 22 Maret 2014 dengan skripsinya yang berjudul "Aplikasi Home Monitoring dan Kontrol Berbasis Android".