



UNIVERSITAS INDONESIA

**RANCANG BANGUN SISTEM PENGATURAN PASOKAN LISTRIK
PADA PEMBANGKIT HIBRIDA**

SKRIPSI

Kadek Eri Mahardika

0906602761

FAKULTAS TEKNIK

DEPARTEMEN TEKNIK ELEKTRO

Depok

Desember 2011



UNIVERSITAS INDONESIA

**RANCANG BANGUN SISTEM PENGATURAN PASOKAN
LISTRIK PADA PEMBANGKIT HIBRIDA**

SKRIPSI

**Diajukan sebagai salah satu syarat untuk memperoleh gelar
sarjana teknik**

Kadek Eri Mahardika

0906602761

FAKULTAS TEKNIK

DEPARTEMEN TEKNIK ELEKTRO

Depok

Desember 2011

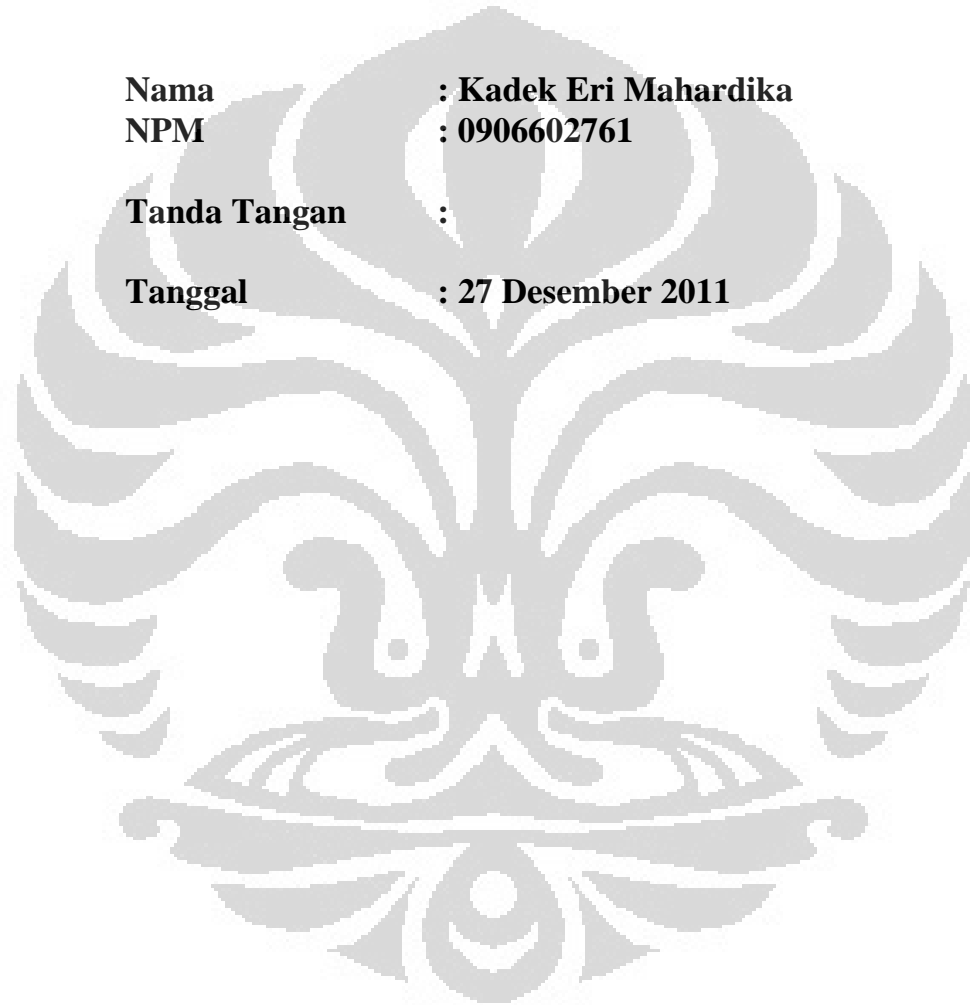
HALAMAN PERNYATAAN ORISINALITAS

Skripsi ini adalah hasil karya saya sendiri, dan semua sumber baik yang dikutip maupun yang dirujuk telah saya nyatakan dengan benar.

Nama : Kadek Eri Mahardika
NPM : 0906602761

Tanda Tangan :

Tanggal : 27 Desember 2011



HALAMAN PENGESAHAN

Skripsi diajukan oleh :

Nama : Kadek Eri Mahardika

NPM : 0906602761

Program Studi : Teknik Elektro

Judul Skripsi : Rancang Bangun Sistem Pengaturan Pasokan Listrik Pada Pembangkit Hibrida

Telah berhasil dipertahankan dihadapan Dewan Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana Teknik pada Program Studi Teknik Elektro, Fakultas Teknik, Universitas Indonesia

DEWAN PENGUJI

Pembimbing : Dr. Abdul Halim, M.eng (.....)

Penguji 1 : Ir. Aries Subiantoro, M.SEE (.....)

Penguji 2 : Prof. Drs. Benyamin Kusumoputro Meng., Dr.Eng (.....)

Ditetapkan di : Depok

Tanggal : 11 Januari 2012

KATA PENGANTAR

Puji syukur penulis panjatkan kehadirat Ida Sang Hyang Widhi Wasa atas segala wahrenugrahanya, sehingga penulis bisa menyelesaikan Skripsi di Universitas Indonesia dengan judul “ *Rancang Bangun Sistem Pengaturan Pasokan Listrik Pada Pembangkit Hibrida*”.

Dalam kesempatan ini penulis mengucapkan terima kasih kepada:

1. Bapak Dr. Abdul Halim, M.eng, atas kesabaran dalam membimbing penulis.
2. Bapak dan Ibu, atas segala do'a, nasehat, kasih sayang dan dukungannya sehingga penulis dapat menyelesaikan Skripsi ini.
3. Kakak saya Erman dan Kadek Yoni, atas semua dukungan dan do'anya.
4. Putu Eka Damanyanti atas semangat dan kasih sayangnya yang diberikan kepada penulis sehingga Skripsi ini selesai.
5. Sahabatku di Bali, Indra S, Vanagosi, Ardi, Yuda, Panca, Ngurah, Dek Win, Sak Ika, Wiwik, Sudria, Sak Yuli, Dian, Agung.
6. Sahabatku Bang Kafahri, Bang Daniel, Bang Ariel, Martin Chorazon, Putri Shaniya, Bang Umar W, yang telah senantiasa menemani dalam keadaan susah maupun senang.
7. Rekan- rekan Ekstensi Elektro angkatan 2009 kekompakan kita semoga terus terjaga sampai nanti kita menjadi alumni.
8. Rekan – rekan Kost Benteng Gading dan Graha Satria, terima kasih doanya. Semoga kita terus bersahabat.

Dan pihak-pihak lain yang tidak dapat disebut satu persatu yang juga berperan bagi penulis dalam penyelesaian laporan Seminar ini Penulis menyadari bahwa laporan ini masih jauh dari sempurna baik dalam materi maupun cara penulisannya. Harapan penulis semoga hasil penulisan laporan Skripsi ini dapat berguna bagi penulis dan pembaca.

27 Desember 2011

Penulis

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI
TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademik Universitas Indonesia, saya yang bertanda tangan di bawah ini:

Nama : Kadek Eri Mahardika

NPM : 0906602761

Program Studi : Teknik Elektro

Departemen : Teknik Elektro

Fakultas : Teknik

Jenis karya : Skripsi

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Indonesia **Hak Bebas Royalti Noneksklusif (*Non-exclusive Royalty-Free Right*)** atas karya ilmiah saya yang berjudul :

Rancang Bangun Sistem Pengaturan Pasokan Listrik Pada Pembangkit Hibrida

Beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Noneksklusif ini, Universitas Indonesia berhak menyimpan, mengalihmedia/formatkan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan mempublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

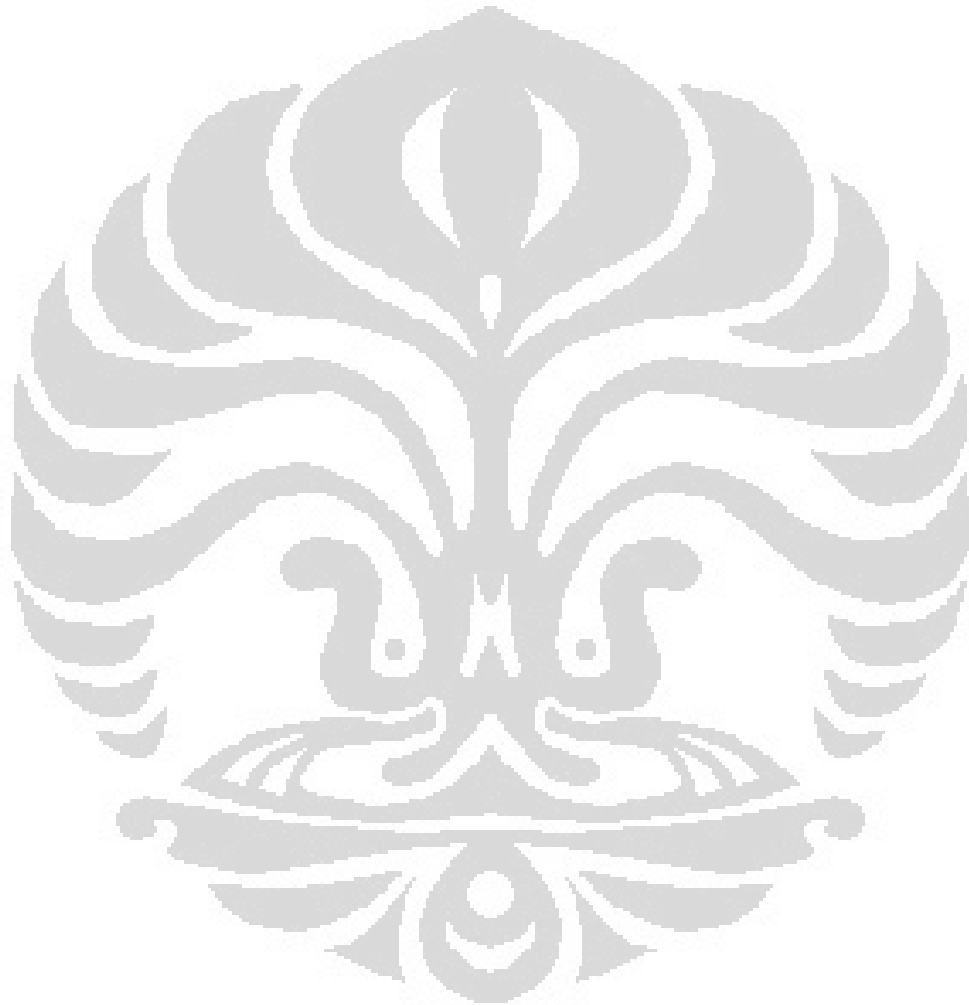
Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Depok

Pada tanggal : 27 Desember 2011

Yang menyatakan

(Kadek Eri Mahardika)



ABSTRAK

Nama : Kadek Eri Mahardika
Program Studi : Teknik Elektro
Judul : Rancang Bangun Sistem Pengaturan Pasokan Listrik Pada Pembangkit Hibrida

Salah satu syarat pembangunan ekonomi suatu negara adalah ketersediaan energi listrik. Energi listrik saat ini tidak hanya dipasok dari sumber energi fosil seperti BBM, gas dan batubara tetapi sudah memanfaatkan sumber energi terbarukan seperti sel surya. Penggunaan sumber energi terbarukan terus diperbesar karena memberikan manfaat lingkungan yang signifikan. Pemilihan sumber energi untuk memasok listrik ke beban yang ada merupakan tema yang penting kedepannya terutama tema bagaimana menjadikan sumber energi terbarukan sebagai sumber listrik utama. Untuk memilih pasokan listrik ini, diperlukan suatu alat yang mengatur secara otomatis pasokan listrik yang akan diberikan ke beban. Sistem pengaturan ini memprioritaskan sumber energi terbarukan. Dalam penelitian ini telah dikembangkan suatu alat yang mengatur secara otomatis sumber pasokan energi listrik. Sistem pasokan listrik yang terdiri dari beberapa jenis sumber ini disebut pembangkit hibrida. Sumber pasokan listrik dapat berupa PLN, genset dan batere yang terhubung dengan panel sel surya. Listrik dari panel surya merupakan sumber utama. Ketika pasokan dari panel surya tidak ada, maka listrik dipasok dari PLN. Tetapi apabila listrik dari PLN tidak ada atau sedang dalam kondisi pemadaman maka listrik dipasok dari Genset. Mekanisme pengaturan ini dilakukan dengan mikrokontroler Atmega 16, yang diprogram dengan menggunakan bahasa C. Alat pengaturan ini juga dapat berfungsi sebagai AMF (automatic main failure) genset. Dari hasil pengujian alat, didapatkan bahwa alat berfungsi sesuai dengan rancangan deskripsi kerjanya.

Kata kunci : sumber energi terbarukan, pengaturan otomatis pasokan listrik, pembangkit hibrida, mikrokontroler Atmega 16.

ABSTRACT

Name : Kadek Eri Mahardika
Field of Study : Electrical Engineering
Title : Development of Electric Power Supply Regulator For Hybrid Power Plant.

One of the requirements of a nation's economic development is the availability of electrical energy. Electrical energy is supplied not only from fossil energy sources such as oil, gas and coal but also from renewable energy sources such as solar cells. Usage of renewable energy sources continues to be enlarged, because it provides significant environmental benefits. One of important themes regarding use of renewable energy source is how to select energy source to be supplied to load, especially how to prioritize renewable energy sources as electrical energy resources. To choose energy resources automatically, a tool is required. In this research, a tool to regulated electric power supply has been developed. Power supply system consists of several kinds of sources that is called hybrid power plant. Sources of electricity supply can be either PLN, generator set and battery that are connected with solar cell panels. Electricity from solar cell panels is the main source. When the supply of solar cell panels do not exist, then the electricity is supplied from PLN. But when the electricity from PLN does not exist or are under condition of the electricity outage the supply done from Genset. Regulation mechanism is carried out by using microcontroller ATmega16, which is programmed using C language. This tool can also function as AMF (automatic main failure)of Genset. From testing result, it was found that tool has shown good performance.

Keyword : Renewable energy sources, automatic regulator power supply, power hybrids, microcontroller Atmega 16.

DAFTAR ISI

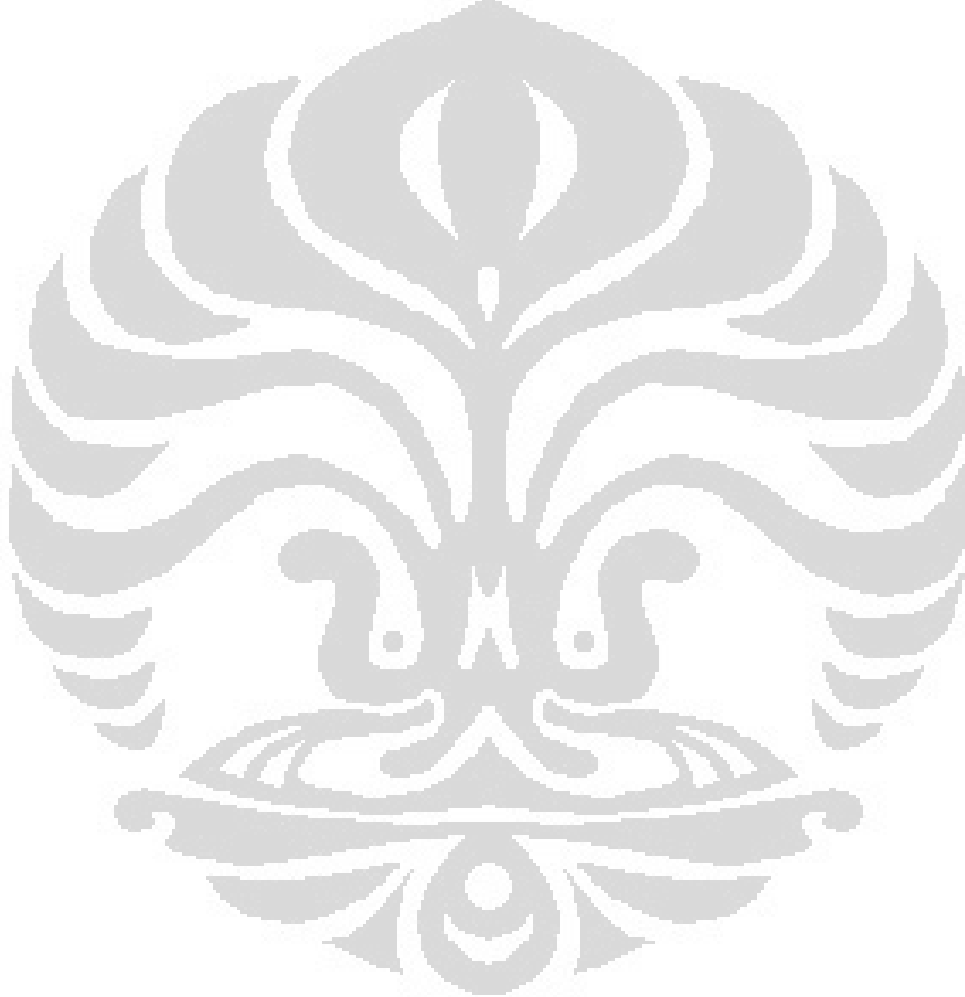
HALAMAN JUDUL.....	i
LEMBAR PERNYATAAN ORISINALITAS.....	ii
LEMBAR PENGESAHAN.....	iii
KATA PENGANTAR.....	iv
LEMBAR PERSETUJUAN PUBLIKASI KARYA ILMIAH.....	v
ABSTRAK.....	vi
ABSTRACT.....	vii
DAFTAR ISI.....	viii
DAFTAR GAMBAR.....	xi
DAFTAR TABEL.....	xii
DAFTAR LAMPIRAN.....	xiii
BAB I PENDAHULUAN.....	1
1.1 Latar Belakang.....	1
1.2 Perumusan Masalah.....	3
1.3 Tujuan.....	4
1.4 Batasan Masalah.....	4
1.5 Metodologi Penulisan.....	4
1.6 Sistematika Penulisa.....	5
BAB II DASAR TEORI.....	6
2.1 Sistem Pembangkit Listrik Tenaga Hibrida.....	6
2.2 Sensor.....	6
2.2.1 Sensor Temperatur.....	6
2.2.1.1 Thermistor.....	7
2.2.1.2 Resistance Thermal Detector (RTD).....	8
2.2.1.3 Termokopel.....	11
2.2.1.4 Dioda sebagai Sensor Temperatur.....	13
2.2.2 Sensor Level.....	14
2.2.2.1 Menggunakan Pelampung.....	14
2.2.2.2 Menggunakan Tekanan.....	15
2.2.2.3 Menggunakan Cara Thermal.....	15
2.2.2.4 Menggunakan Cara Optik.....	17
2.2.2.4 Menggunakan Sinar Laser.....	17
2.2.2.4 Menggunakan Prisma.....	18
2.2.3 Sensor Over Voltage.....	18
2.3 Perkembangan Mikrokontroler.....	19
2.4 Mikrokontroler Keluarga AVR.....	20
2.5 Perlengkapan Dasar Mikrokontroler.....	22
2.6 Arsitektur ATmega 16.....	24
2.7 Konfigurasi Pin Atmega 16.....	26
2.8 Mikrokontroler AVR dan Bahasa C.....	27
2.9 Sekilas Tentang CodeVisionAVR.....	29
2.10 Konsep I/O Pada mikrokontroler AVR ATmega 16.....	34
2.11 Konsep LCD.....	35
2.12 Konsep ADC.....	37

BAB III PERANCANGAN DAN REALISASI.....	41
3.1 Dasar Perancangan.....	41
3.2 Tahapan Perancangan.....	42
3.3 Spesifikasi Alat.....	43
3.4 Diskripsi Kerja.....	43
3.5 Perancangan dan Realisasi Hadware.....	49
3.5.1 Perancangan dan Realisasi Sensor Suhu.....	49
3.5.2 Perancangan Sensor Lever Fuel.....	51
3.5.3 Perancangan dan Realisasi Sensor Kondisi Aki Solarcell	52
3.5.4 Perancangan rangkaian driver.....	53
3.5.5 Perancangan dan Realisasi Modul Kontrol	56
3.5.6 Perancangan Kontruksi Box panel	59
3.5.7 Tabel Input dan Output Mikrokontroler.....	60
3.6 Perancangan Dan Realisasi Software.....	61
 BAB IV PENGUJIAN DAN ANALISA	 65
4.1 Tujuan Pengujian.....	65
4.2 Pengujian Kinerja Alat.....	65
4.2.1 Pengujian Input.....	65
4.2.2 Pengujian Output.....	66
4.2.3 Pengujian Mode Manual.....	66
4.2.4 Pengujian Mode Otomatik.....	67
4.3 Tujuan Analisa.....	68
4.4 Analisa.....	68
4.4.1 Analisa Mode manual.....	68
4.4.2 Analisa Mode Otomatik.....	69
 BAB V KESIMPULAN DAN SARAN.....	 70
5.1 Kesimpulan.....	70
5.2 Saran.....	70

DAFTAR GAMBAR

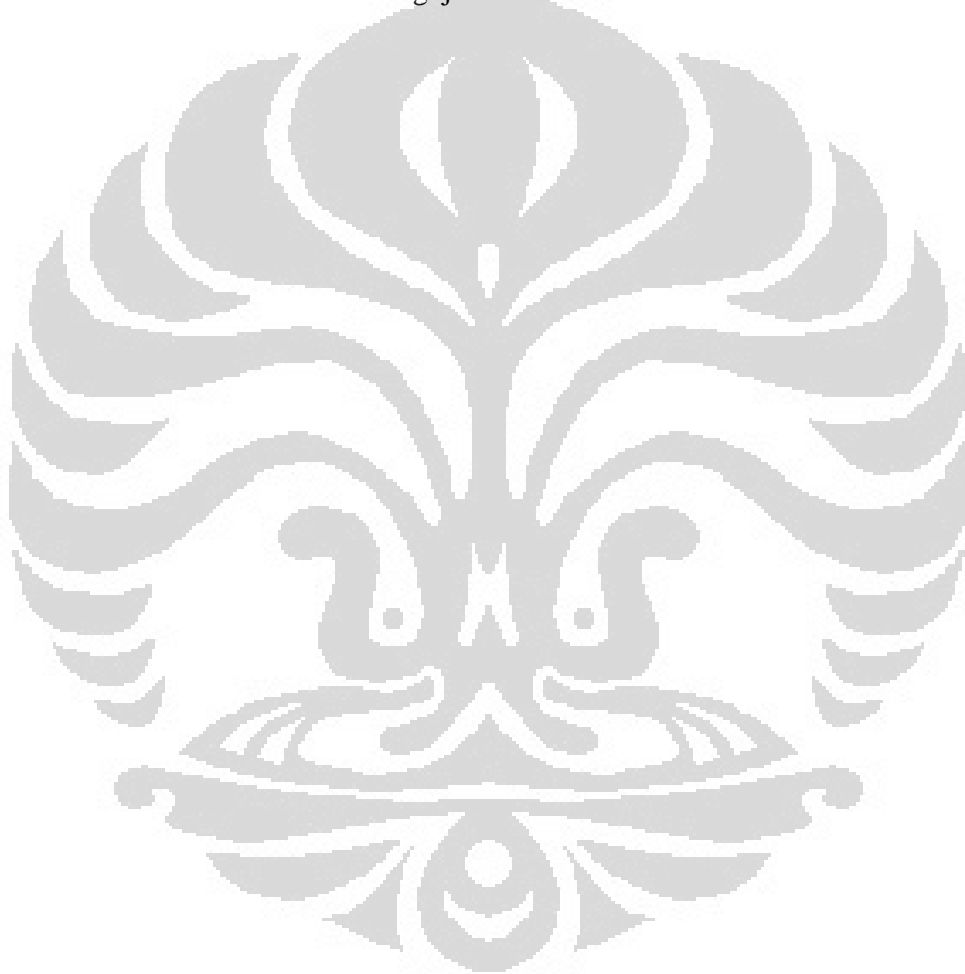
Gambar 2.1	Karakteristik sensor temperature.....	7
Gambar 2.2	Konfigurasi Thermistor.....	8
Gambar 2.3	<i>Konstruksi RTD</i>	9
Gambar 2.4	Resistansi versus Temperatur untuk variasi RTD metal.....	10
Gambar 2.5	Jenis RTD.....	10
Gambar 2.6	Rangkaian Penguat linier.....	10
Gambar 2.7	Arah gerak electron jika logam dipanaskan.....	11
Gambar 2.8	Beda potensial pada Termokopel.....	12
Gambar 2.9	Rangkaian penguat tegangan junction termokopel.....	12
Gambar 2.10	Karateristik beberapa tipe termokopel.....	13
Gambar 2.11	Contoh rangkaian dengan dioda sebagai sensor temperature.....	13
Gambar 2.12	Contoh rangkaian dengan IC sensor.....	14
Gambar 2.13	Rangkaian peubah arus ke tegangan untuk IC termo sensor.....	14
Gambar 2.14	Sensor Level Menggunakan Pelampung.....	15
Gambar 2.15	Sensor Level Menggunakan Sensor Tekanan.....	15
Gambar 2.16	Teknik Penyensoran Level Cairan Cara Thermal.....	16
Gambar 2.17	Blok Pengolahan Sensor Level Menggunakan Cara Thermal	17
Gambar 2.18	Sensor Level menggunakan Sinar Laser.....	18
Gambar 2.19	Sensor Level menggunakan Prisma.....	18
Gambar 2.20	Over Voltage Detector.....	19
Gambar 2.21	Blok Diagram ATMEGA16.....	25
Gambar 2.22	Tata Letak Kaki ATMEGA16.....	27
Gambar 2.23	Tampilan <i>New File</i>	31
Gambar 2.24	Tampilan <i>Option di Wizard CVAVR</i>	31
Gambar 2.25	Tampilan <i>IC port</i>	32
Gambar 2.26	Tampilan <i>Setting Port CVAVR</i>	32
Gambar 2.27	<i>Setting ADC</i>	33
Gambar 2.28	<i>Setting LCD</i>	33
Gambar 2.29	Tampilan <i>Save Generate Project</i>	34
Gambar 2.30	<i>setting LCD menggunakan CodeVision AVR</i>	36
Gambar 2.31	<i>Setting ADC</i>	40
Gambar 3.1	Flowchart Tahapan Perancangan dan Realisasi.....	42
Gambar 3.2	Blok Diagram sistem kontrol	43
Gambar 3.3	<i>Picture Diagram</i> Sistem Kontrol	44
Gambar 3.4	Flowchart Cara Kerja	45
Gambar 3.5	Flowchart Mode Otomatis.....	46
Gambar 3.6	Flowchart Mode Manual.....	47
Gambar 3.7	Rangkaian Sensor LM35.....	49
Gambar 3.8	Rangkaian Sensor LM35.....	50
Gambar 3.9	Sensor Level.....	51
Gambar 3.10	Sensor Level.....	51
Gambar 3.11	Rangkaian sensor over dan kondisi <i>accu</i>	53
Gambar 3.12	<i>Optocoupler</i>	54
Gambar 3.13	Skema Rangkaian <i>Driver Optocoupler</i>	55

Gambar 3.14	Skema Rangkaian <i>Driver Optocoupler</i>	55
Gambar 3.15	<i>PCB Driver Optocoupler</i>	56
Gambar 3.16	Modul kontrol	57
Gambar 3.17	Sistem <i>Minimum Mikrokontroler AVR Atmega 16</i>	58
Gambar 3.18	<i>Modul Isolated Input</i>	58
Gambar 3.19	Penampang depan <i>Box Panel</i>	59
Gambar 3.20	<i>Box Panel Sistem Kontrol</i>	60
Gambar 3.21	<i>Flowchart rancangan software</i>	62
Gambar 3.22	<i>Flowchart rancangan software</i> mode otomatis.....	63
Gambar 3.23	<i>Flowchart rancangan software</i> mode manual.....	64



DAFTAR TABEL

Tabel 2.1	Perbedaan seri AVR berdasarkan jumlah memori.....	22
Tabel 2.2	Beberapa Compiler C untuk mikrokontroler AVR.....	29
Tabel 2.3	Konfigurasi Pin LCD.....	35
Tabel 3.1	Tabel Input.....	63
Tabel 3.2	Tabel Output.....	63
Tabel 4.1	Pengujian Input.....	65
Tabel 4.2	Pengujian Output.....	66
Tabel 4.3	Tabel Hasil Pengujian mode Manual.....	67
Tabel 4.4	Tabel Hasil Pengujian mode Otomatis.....	68



BAB I

PENDAHULUAN

1.1 Latar Belakang

Pada zaman modern saat ini, energi listrik merupakan salah satu sumber energi yang sangat penting dalam kelangsungan hidup umat manusia. Dalam kesehariannya banyak peralatan rumah tangga, perusahaan, serta pabrik-pabrik yang menggunakan energi listrik untuk kelancaran aktivitasnya. Seiring dengan kemajuan zaman dan bertambahnya jumlah penduduk, maka jumlah rumah tinggal, perusahaan, dan pabrik ikut meningkat juga. Oleh karena itu kebutuhan akan pasokan listrik terus meningkat.

Banyak kegiatan yang memerlukan kontinuitas pasokan daya listrik, terutama dibidang industri dan pelayanan publik. Kontinuitas pasokan daya listrik ini sangat penting guna menghindari kerusakan peralatan listrik, kontinuitas proses produksi, dan kontinuitas proses pelayanan publik. Untuk menjaga kontinuitas daya listrik ini, maka industri-industri, dan gedung-gedung pelayanan publik membutuhkan cadangan daya yang dapat digunakan saat pasokan daya listrik dari PLN terganggu. Cadangan daya yang paling umum dipakai adalah *Generator Set* (GENSET).

Energi makin mahal, tarif listrik PLN akan terus dinaikkan sampai mencapai besaran tarif yang memungkinkan PLN menjadi BUMN yang sehat. Sebagai konsumen yang tergantung pada pasokan PLN, sepantasnya makin sadar betapa pentingnya menggunakan listrik seefisien dan seefektif mungkin. Penghematan akan menurunkan biaya penggunaan listrik, namun tidak harus menurunkan tingkat kenyamanan. Penghematan juga bernilai sosial. Jika setiap konsumen mau menghemat antara 5 s/d 10% saja, akan memberikan peluang pada saudara kita yang sedang menunggu sambungan listrik, bisa segera dipenuhi tanpa menunggu dibangunnya pembangkit baru.

Salah satu cara untuk melakukan penghematan penggunaan listrik dari PLN adalah dengan menggunakan energi alternatif seperti *solarcell* dan *mikrohidro*. Namun energi yang lebih memungkinkan diaplikasikan diberbagai daerah adalah

energi dari *solarcell*. Dengan memanfaatkan energi dari sinar matahari yang disimpan dalam sebuah battre / *accu* dalam bentuk tegangan DC, kemudian diubah ke tegangan AC. Energi alternatif ini kemungkinan hanya efektif digunakan dari pagi sampai sore hari saat masih ada sinar matahari. jadi untuk memasok energi pada saat malam hari kita masih menggunakan pasokan dari PLN. Sedangkan GENSET sendiri digunakan pada saat pasokan dari PLN tidak ada atau sedang terjadi pemadaman. Jadi dengan menggabungkan ketiga sumber energi (sistem pembangkit) ini kita mendapatkan sebuah sistem catu daya yang dapat menghemat penggunaan energi listrik dari PLN. Penggabungan ketiga sistem pembangkit ini biasanya disebut dengan sistem pembangkit listrik *hibrida*.

Untuk memenuhi kebutuhan akan sistem pengaturan pada sistem catu daya ini maka diperlukan suatu alat yang dapat bekerja secara otomatis. Alat ini digunakan untuk menurunkan *down-time* dan meningkatkan keandalan sistem catu daya listrik. Selain itu juga dapat mengendalikan perpindahan/transfer *Circuit Breaker* (CB) atau alat sejenis, dari catu daya utama (*sollarcell*) ke catu daya cadangan (PLN dan Genset) atau sebaliknya. Disamping kegunaan sebagai alat transfer, Alat ini juga dilengkapi dengan fungsi lain, diantaranya : start & stop *primover* (misal : mesin diesel), sensor-sensor *malfunction*, alat proteksi, indikator dan alarm.

Biasanya alat serupa menggunakan *Programmable Logic Controller* (PLC) sebagai kontroler utamanya. Namun karena harga dari sebuah PLC yang relatif mahal, maka hanya kalangan industri-industri sekala besar dan gedung-gedung besar yang mempunyai dana cukup untuk membeli dan memakai PLC sebagai kontrolernya. Sedangkan untuk kalangan industri menengah kebawah, gedung-gedung komersial sekala kecil atau bahkan untuk rumah tinggal hampir tidak mungkin untuk mengeluarkan dana untuk membeli PLC , yang harganya bisa hampir sama dengan harga Genset itu sendiri. Melihat hal ini penulis merasa tertarik untuk membuat sebuah kontrol yang tidak terlalu mahal tapi dengan keandalan yang sama dengan yang berbasis PLC. Adapun kontrol yang dapat dipakai adalah mikrokontroler AVR (*Alf and Vegard's Risc processor*) seri

ATMega16 dari Atmel. Dengan menggunakan mikrokontroler ini kita dapat membuat sebuah kontroler dengan biaya yang cukup ekonomis.

1.2 Perumusan Masalah

Dalam era kemajuan teknologi dan informasi sekarang ini, pertumbuhan industri dan bisnis menjadi semakin cepat dari waktu ke waktu. Namun hal ini juga mendorong penggunaan energi yang semakin tinggi dan menjadikan penggunaan energi menjadi salah satu kontributor besar biaya operasional yang harus dikeluarkan. Kondisi tersebut semakin diperburuk dengan isu-isu mengenai kenaikan Tarif Dasar Listrik (TDL) yang baru-baru ini semakin marak dibicarakan. Disamping itu akhir-akhir ini pemerintah marak mensosialisasikan penghematan energi khususnya energi listrik. Ini karena Negara kita masih kekurangan pembangkit tenaga listrik untuk melayani kenaikan jumlah pelanggan yang semakin meningkat. Disamping itu kebanyakan pembangkit listrik kita masih menggunakan energi dari batu bara dan minyak bumi, yang semakin hari sumber batu bara dan minyak bumi ini semakin mahal karena semakin menipis cadangannya.

Pembangkit dari sumber energi alternatif seperti solarcell sangat tergantung pada kondisi cuaca dan hanya bisa efektif digunakan mulai pagi sampai sore hari. Sedangkan PLN sebagai penyedia utama pasokan energi listrik tidak mungkin secara terus menerus dapat memberikan pasokan energi, gangguan pada saat pendistribusian bisa saja terjadi sewaktu-waktu tanpa bisa diprediksi. Oleh karena itu dibutuhkan alat yang dapat mengontrol sistem cadangan yang bekerja secara otomatis menghidupkan Genset guna membackup pasokan energi listrik apabila terjadi gangguan pada pembangkit listrik dari solarcell dan terjadi pemadaman secara mendadak dari PLN.

1.3 Tujuan

Adapun tujuan dari pembuatan skripsi ini adalah

1. Mengaplikasikan mikrokontroler ATmega16 sebagai kontroler sistem pengaturan pasokan listrik pada pembangkit hibrida.
2. Merancang dan membuat *hardware* dan *software* sistem pengaturan pasokan listrik pada pembangkit hibrida.

1.4 Batasan Masalah

Dalam skripsi ini masalah akan dibatasi pada :

1. Sistem pembangkit listrik *hibrida* yang di pakai hanya *solarcell* dan GENSET,
2. Langsung di aplikasikan ke GENSET dengan daya kecil.
3. Sensor – sensor *malfunction* yang terintegrasi hanya 4 buah yaitu *Low Fuel, Over Heat, Over Voltage* dan *Oil Alert*.
4. Semua Keadaan yang di deteksi oleh sensor akan terlihat di layar LCD, seperti Keadaan sumber energi alternatif (*solarcell*), *Fuel*, Suhu dari, serta besarnya tegangan yang keluar dari GENSET.
5. Dalam pengujiannya *Solarcell* diganti dengan *power supply* DC 12 Volt.
6. Dalam perancangan *software*, digunakan *software* CodeVisionAVR yang berbasis bahasa C.

1.5 Metodologi Penulisan

a. Studi Literatur

Studi literatur digunakan untuk membangun dasar-dasar teori yang diperlukan dalam penulisan skripsi , serta dalam proses perancangan yang berhubungan dengan sistem pengaturan pasokan listrik pada pembangkit hibrida dan Mikrokontroler ATmega16.

b. Analisa

Analisa ini dilakukan untuk menganalisa dari proses perancangan dan pembuatan alat yang akan dibuat. Mulai dari menganalisa kebutuhan pengguna (*user requirement*), sehingga dari sini kita bisa menentukan

spesifikasi, dan dimensi alat yang kita buat. Menganalisa sistem yang akan kita bangun apakah sesuai dengan kebutuhan pengguna (*user*) nantinya.

c. Perancangan

Penulis akan melakukan Perancangan dari hasil analisa

d. Pembuatan

Rancangan yang sudah dibuat dan dianalisa kemudian akan direalisasikan dalam bentuk pembuatan alat, sesuai dengan rancangan yang sudah dibuat

e. Pengujian

Pengujian dilakukan setelah alat selesai dibuat sesuai dengan rancangannya. Pengujian ini dilakukan untuk mengetahui apakah alat yang kita buat, sudah sesuai dengan sistem, dan spesifikasi yang kita rancang.

1.6 Sistematika Penulisan

Sistematika penulisan skripsi ini terdiri dari 5 BAB yaitu ; BAB I PENDAHULUAN. Pada bab ini akan berisikan mengenai latar belakang, tujuan, rumusan masalah, metodologi, dan sistematika dalam penulisan Skripsi. BAB II TEORI DASAR, Bab ini berisi penjelasan secara teori dari bahasan yang diambil dalam skripsi ini. BAB III PERANCANGAN DAN REALISASI ALAT, Pada bab ini akan dijelaskan mengenai proses perancangan dan realisasi alat. Mulai dari penentuan spesifikasi alat, pemilihan komponen, dan langkah pembuatan alat. BAB IV PENGUJIAN DAN ANALISIS, Bab ini menjelaskan hasil pengujian yang dilakukan terhadap alat yang dibuat. Hasil pengujian tersebut kemudian akan dianalisa. BAB V PENUTUP, Pada bab terakhir ini berisi kesimpulan dari seluruh proses penulisan Skripsi dan berisi saran dari penulis.

BAB II DASAR TEORI

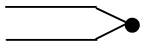



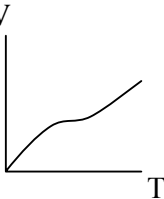
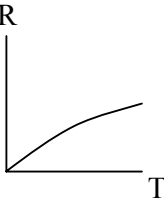
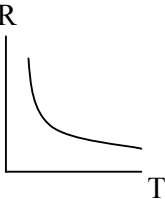
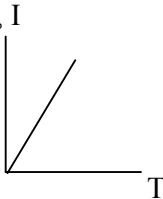
2.1 Sistem Pembangkit Listrik Tenaga Hibrida

Pembangkit Listrik Tenaga Hibrida (PLTH) adalah suatu sistem pembangkit listrik yang memadukan beberapa jenis pembangkit listrik, pada umumnya antara pembangkit listrik berbasis BBM dengan pembangkit listrik berbasis energi terbarukan. Merupakan solusi untuk mengatasi krisis BBM dan ketiadaan listrik di daerah terpencil, pulau-pulau kecil dan pada daerah perkotaan. Umumnya terdiri atas: modul foto voltaik, turbin angin, generator diesel, baterai, dan peralatan kontrol yang terintegrasi. Tujuan PLTH adalah mengkombinasikan keunggulan dari setiap pembangkit sekaligus menutupi kelemahan masing-masing pembangkit untuk kondisi-kondisi tertentu, sehingga secara keseluruhan sistem dapat beroperasi lebih ekonomis dan efisien. Mampu menghasilkan daya listrik secara efisien pada berbagai kondisi pembebanan Untuk mengetahui unjuk kerja sistem pembangkit hibrida ini, hal – hal yang perlu dipertimbangkan antara lain: karakteristik beban pemakaian dan karakteristik pembangkitan daya khususnya dengan memperhatikan potensi energy alam yang ingin dikembangkan berikut karakteristik kondisi alam itu sendiri, seperti pergantian siang malam, musim dan sebagainya.

2.2 Sensor-Sensor

2.3.1 Sensor Temperatur

Gambar 2.1 berikut memperlihatkan karakteristik dari beberapa jenis sensor suhu yang ada.

	Thermocouple	RTD	Thermistor	IC Sensor
				
				

Advantages	<ul style="list-style-type: none"> - self powered - simple - rugged - inexpensive - wide variety - wide temperature range 	<ul style="list-style-type: none"> - most stable - most accurate - more linear than thermocouple 	<ul style="list-style-type: none"> - high output - fast - two-wire ohms measurement 	<ul style="list-style-type: none"> - most linear - highest output - inexpensive
Disadvantage	<ul style="list-style-type: none"> - non linear - low voltage - reference required - least stable - least sensitive 	<ul style="list-style-type: none"> - expensive - power supply required - small ΔR - low absolute resistance - self heating 	<ul style="list-style-type: none"> - non linear - limited temperature range - fragile - power supply required - self heating 	<ul style="list-style-type: none"> - $T < 200^{\circ}\text{C}$ - power supply required - slow - self heating - limited configuration

Gambar 2.1. Karakteristik sensor temperature (Schuller, Mc.Name, 1986)

Setiap sensor suhu memiliki temperatur kerja yang berbeda, untuk pengukuran suhu disekitar kamar yaitu antara -35°C sampai 150°C , dapat dipilih sensor NTC, PTC, transistor, dioda dan IC hibrid. Untuk suhu menengah yaitu antara 150°C sampai 700°C , dapat dipilih thermocouple dan RTD. Untuk suhu yang lebih tinggi sampai 1500°C , tidak memungkinkan lagi dipergunakan sensor-sensor kontak langsung, maka teknis pengukurannya dilakukan menggunakan cara radiasi. Untuk pengukuran suhu pada daerah sangat dingin dibawah $65^{\circ}\text{K} = -208^{\circ}\text{C}$ ($0^{\circ}\text{C} = 273,16^{\circ}\text{K}$) dapat digunakan resistor karbon biasa karena pada suhu ini karbon berlaku seperti semikonduktor. Untuk suhu antara 65°K sampai -35°C dapat digunakan kristal silikon dengan kemurnian tinggi sebagai sensor.

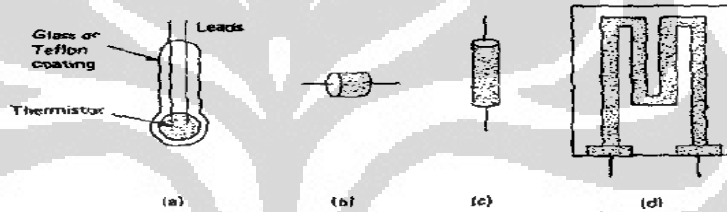
2.2.1.1 Termistor

Termistor atau tahanan *thermal* adalah alat semikonduktor yang berkelakuan sebagai tahanan dengan koefisien tahanan temperatur yang tinggi, yang biasanya negatif. Umumnya tahanan termistor pada temperatur ruang dapat berkurang 6% untuk setiap kenaikan temperatur sebesar 1°C . Kepekaan yang tinggi terhadap perubahan temperatur ini membuat termistor sangat sesuai untuk pengukuran, pengontrolan dan kompensasi temperatur secara presisi.

Termistor terbuat dari campuran oksida-oksida logam yang diendapkan seperti: mangan (Mn), nikel (Ni), cobalt (Co), tembaga (Cu), besi (Fe) dan uranium (U). Rangkuman tahanannya adalah dari $0,5 \Omega$ sampai 75Ω dan tersedia

dalam berbagai bentuk dan ukuran. Ukuran paling kecil berbentuk mani-manik (*beads*) dengan diameter 0,15 mm sampai 1,25 mm, bentuk piringan (*disk*) atau cincin (*washer*) dengan ukuran 2,5 mm sampai 25 mm. Cincin-cincin dapat ditumpukan dan di tempatkan secara seri atau paralel guna memperbesar disipasi daya.

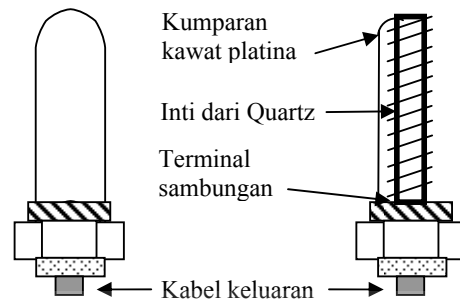
Dalam operasinya termistor memanfaatkan perubahan resistivitas terhadap temperatur, dan umumnya nilai tahanannya turun terhadap temperatur secara eksponensial untuk jenis NTC (*Negative Thermal Coeffisien*). Konfigurasi thermistor terdiri dari 4 macam diantaranya ; *coated-bead*, *disk*, *dioda case*, dan *thin film*, seperti yang lihat pada gambar 2.2.



Gambar 2.2. Konfigurasi Thermistor: (a) coated-bead
(b) disk (c) dioda case dan (d) thin-film

2.2.1.2. Resistance Thermal Detector (RTD)

RTD adalah salah satu dari beberapa jenis sensor suhu yang sering digunakan. RTD dibuat dari bahan kawat tahan korosi, kawat tersebut dililitkan pada bahan keramik isolator. Bahan tersebut antara lain; platina, emas, perak, nikel dan tembaga, dan yang terbaik adalah bahan platina karena dapat digunakan menyensor suhu sampai 1500°C . Tembaga dapat digunakan untuk sensor suhu yang lebih rendah dan lebih murah, tetapi tembaga mudah terserang korosi. Kontruksi dari RTD bisa dilihat pada gambar 2.3, sebuah inti keramik jenis Quartz, dililit dengan kawat platina.



Gambar 2.3. Konstruksi RTD

RTD memiliki keunggulan dibanding termokopel yaitu:

1. Tidak diperlukan suhu referensi
2. Sensitivitasnya cukup tinggi, yaitu dapat dilakukan dengan cara memperpanjang kawat yang digunakan dan memperbesar tegangan eksitasi.
3. Tegangan output yang dihasilkan 500 kali lebih besar dari termokopel
4. Dapat digunakan kawat penghantar yang lebih panjang karena noise tidak jadi masalah
5. Tegangan keluaran yang tinggi, maka bagian elektronik pengolah sinyal menjadi sederhana dan murah.

Resistance Thermal Detector (RTD) perubahan tahanannya lebih linear terhadap temperatur uji tetapi koefisien lebih rendah dari thermistor dan model matematis linier adalah:

$$R_T = R_0(1 + \alpha\Delta t) \quad (2.1)$$

dimana : R_0 = tahanan konduktor pada temperature awal (biasanya 0°C)

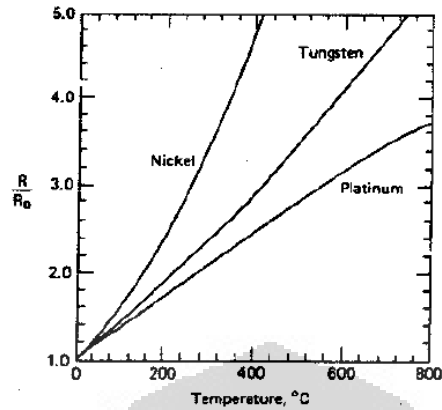
R_T = tahanan konduktor pada temperatur $t^\circ\text{C}$

α = koefisien temperatur tahanan

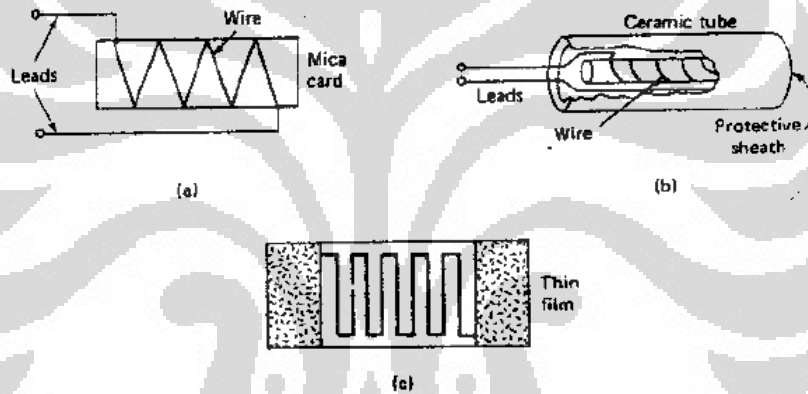
Δt = selisih antara temperatur kerja dengan temperatur awal

Sedangkan model matematis nonlinear kuadratik dari gambar 2.4 adalah:

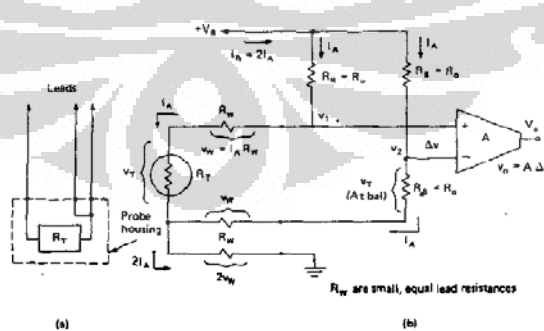
$$R_T = R_0(1 + AT - BT^2) \quad (2.2)$$



Gambar 2.4. Resistansi versus Temperatur untuk variasi RTD metal
 Gambar 2.5 merupakan bentuk lain dari Konstruksi RTD



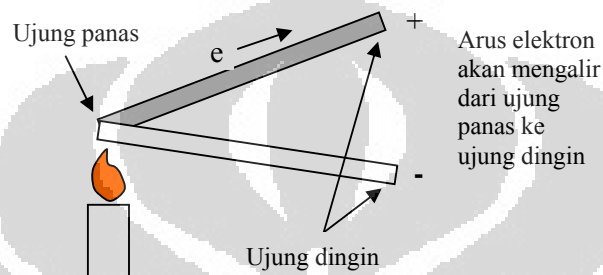
Gambar 2.5. Jenis RTD: (a) Wire (b) Ceramic Tube (c) Thin Film
 Gambar 2.6 merupakan gambar rangkaian Penguat untuk *three-wire* RTD



Gambar 2.6. (a) Three Wire RTD (b) Rangkaian Penguat linier; (b) Blok diagram rangkaian koreksi

2.2.1.3. Termokopel

Pembuatan termokopel didasarkan atas sifat thermal bahan logam. Seperti terlihat pada gambar 2.7, Jika sebuah batang logam dipanaskan pada salah satu ujungnya maka pada ujung tersebut elektron-elektron dalam logam akan bergerak semakin aktif dan akan menempati ruang yang semakin luas, elektron-elektron saling desak dan bergerak ke arah ujung batang yang tidak dipanaskan. Dengan demikian pada ujung batang yang dipanaskan akan terjadi muatan positif.



Gambar 2.7. Arah gerak electron jika logam dipanaskan

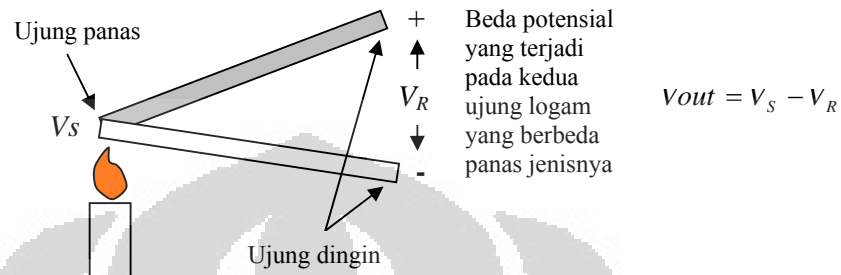
Kerapatan elektron untuk setiap bahan logam berbeda tergantung dari jenis logam. Jika dua batang logam disatukan salah satu ujungnya, dan kemudian dipanaskan, maka elektron dari batang logam yang memiliki kepadatan tinggi akan bergerak ke batang yang kepadatan elektronnya rendah, dengan demikian terjadilah perbedaan tegangan diantara ujung kedua batang logam yang tidak disatukan atau dipanaskan seperti terlihat pada gambar 2.8. Besarnya termolistrik yang dihasilkan menurut T.J Seeback (1821) yang menemukan hubungan perbedaan panas (T_1 dan T_2) dengan gaya gerak listrik yang dihasilkan E , Peltir (1834), menemukan gejala panas yang mengalir dan panas yang diserap pada titik *hot-juction* dan *cold-juction*, dan Sir William Thomson, menemukan arah arus mengalir dari titik panas ke titik dingin dan sebaliknya, sehingga ketiganya menghasilkan rumus sbb:

$$E = \underbrace{C_1(T_1-T_2)}_{\text{Efek Peltier}} + \underbrace{C_2(T_1^2 - T_2^2)}_{\text{Efek Thomson}} \quad (2.3)$$

Efek Peltier *Efek Thomson*

$$\text{atau } E = 37,5(T_1-T_2) - 0,045(T_1^2-T_2^2) \quad (2.4)$$

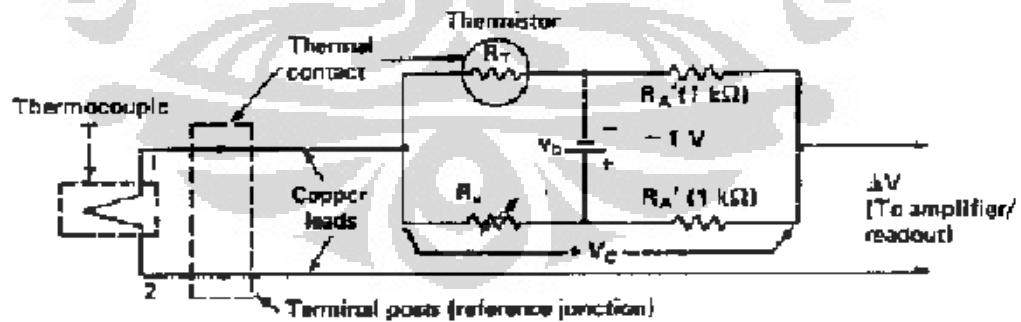
di mana 37,5 dan 0,045 merupakan dua konstanta C_1 dan C_2 untuk termokopel tembaga/konstanta.



Gambar 2.8. Beda potensial pada Termokopel

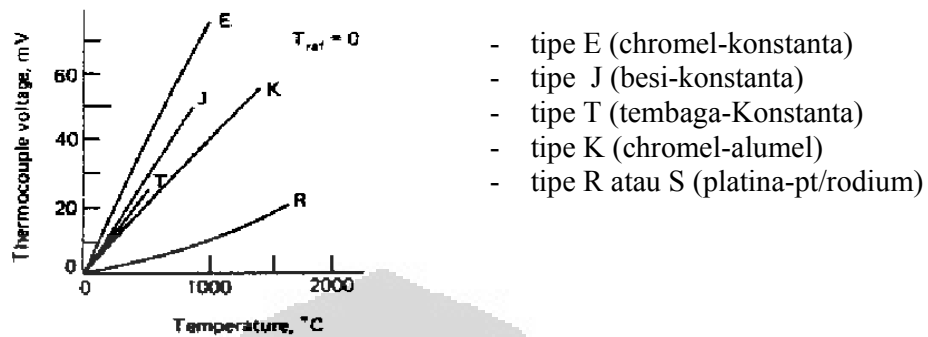
Bila ujung logam yang tidak dipanaskan dihubungkan singkat, perambatan panas dari ujung panas ke ujung dingin akan semakin cepat. Sebaliknya bila suatu termokopel diberi tegangan listrik DC, maka diujung sambungan terjadi panas atau menjadi dingin tergantung polaritas bahan dan polaritas tegangan sumber. Dari prinsip ini memungkinkan membuat termokopel menjadi pendingin.

Thermocouple sebagai sensor temperatur memanfaatkan beda *workfunction* dua bahan metal. Rangkaian kompensasi untuk Thermocouple diperlihatkan oleh gambar 2.9.



Gambar 2.9. Rangkaian penguat tegangan junction termokopel

Perilaku beberapa jenis thermocouple diperlihatkan oleh gambar 2.10



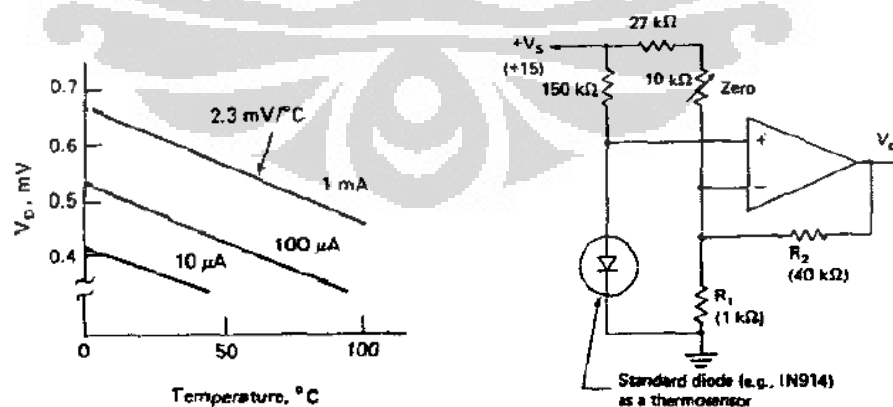
Gambar 2.10. Karakteristik beberapa tipe termokopel

2.2.1.4. Dioda sebagai Sensor Temperatur

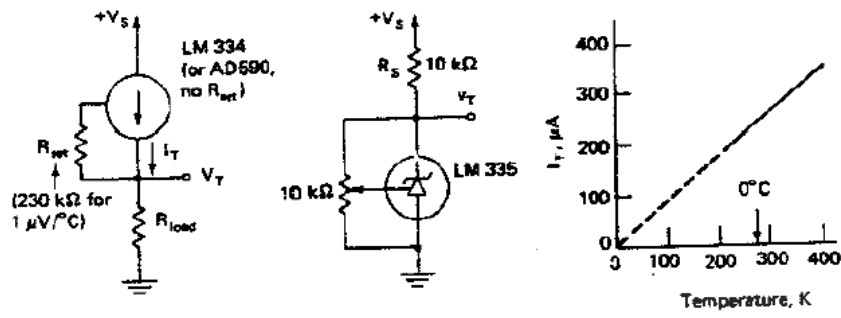
Dioda dapat pula digunakan sebagai sensor temperatur yaitu dengan memanfaatkan sifat tegangan *junction*

$$I_D = I_S \exp\left(\frac{V_D}{nV_T}\right) \quad (2.5)$$

Dimanfaatkan juga pada sensor temperatur rangkaian terintegrasi (memiliki rangkaian penguat dan kompensasi dalam chip yang sama). Gambar 2.11. memperlihatkan grafik hubungan antara tegangan keluaran sensor dengan kenaikan suhu, serta gambar rangkaian dioda sebagai sensor suhu. Sedangkan gambar 2.12 memperlihatkan skema rangkaian dengan IC sensor (LM 35).

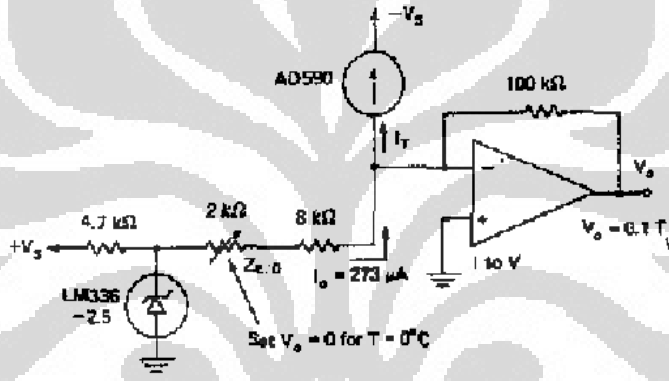


Gambar 2.11 Contoh rangkaian dengan dioda sebagai sensor temperature



Gambar 2.12 Contoh rangkaian dengan IC sensor

Gambar 2.13 merupakan rangkaian alternatif untuk mengubah arus menjadi tegangan pada IC sensor temperature



Gambar 2.13 Rangkaian peubah arus ke tegangan untuk IC termo sensor

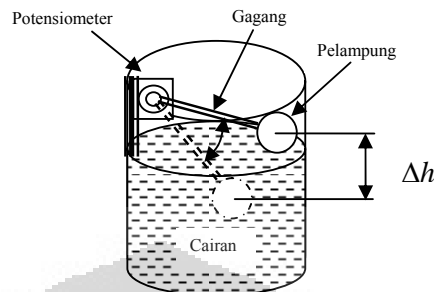
2.2.2 Sensor Level

Pengukuran level dapat dilakukan dengan bermacam cara antara lain dengan: pelampung atau displacer, gelombang udara, resistansi, kapasitif, ultra sonic, optic, thermal, tekanan, sensor permukaan dan radiasi. Pemilihan sensor yang tepat tergantung pada situasi dan kondisi sistem yang akan di sensor.

2.2.2.1 Menggunakan Pelampung

Cara yang paling sederhana dalam penyensor level cairan dapat dilihat pada gambar 2.14, adalah dengan menggunakan pelampung yang diberi gagang. Pembacaan dapat dilakukan dengan memasang sensor posisi misalnya

potensiometer pada bagian engsel gagang pelampung. Cara ini cukup baik diterapkan untuk tanki-tanki air yang tidak terlalu tinggi.

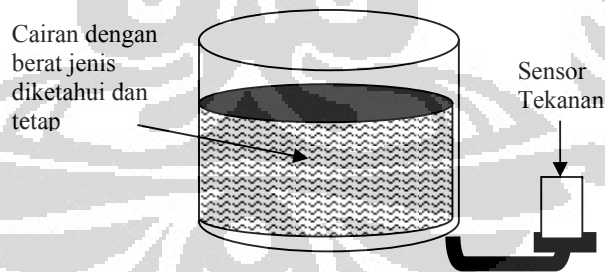


Gambar 2.14 Sensor Level Menggunakan Pelampung

2.2.2.2 Menggunakan Tekanan

Untuk mengukur level cairan dapat pula dilakukan menggunakan sensor tekanan yang dipasang di bagian dasar dari tabung seperti terlihat pada gambar 2.15. Cara ini cukup praktis, akan tetapi ketelitiannya sangat tergantung dari berat jenis dan suhu cairan sehingga kemungkinan kesalahan pembacaan cukup besar.

Sedikit modifikasi dari cara diatas adalah dengan cara mencelupkan pipa berisi udara kedalam cairan. Tekanan udara didalam tabung diukur menggunakan sensor tekanan, cara ini memanfaatkan hukum Pascal. Kesalahan akibat perubahan berat jenis cairan dan suhu tetap tidak dapat diatasi.



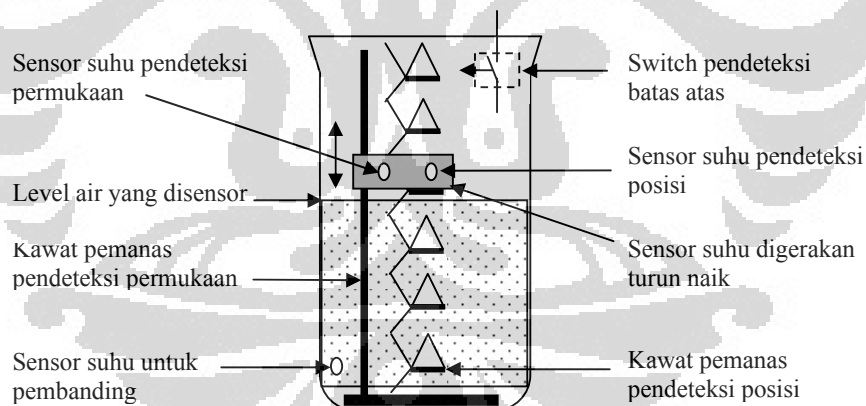
Gambar 2.15. Sensor Level Menggunakan Sensor Tekanan

2.3.2.3 Menggunakan Cara Thermal

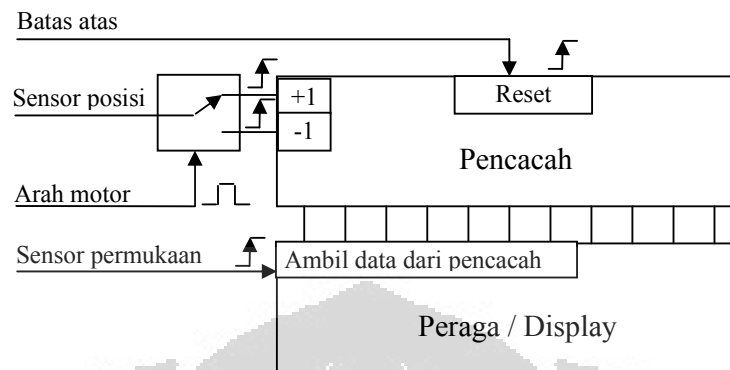
Seperti yang terlihat pada gambar 2.16, Teknik penyensoran ini didasarkan pada fakta penyerapan kalor oleh cairan lebih tinggi dibandingkan penyerapan kalor oleh uapnya, sehingga bagian yang tercelup akan lebih dingin dibandingkan bagian yang tidak tercelup. Kontruksi dasar sensor adalah terdiri dari sebuah

elemen pemanas dibentuk berliku-liku dan sebuah pemanas lain dibentuk tetap lurus. Dua buah sensor diletakkan berhadapan dengan bagian tegak dari pemanas, sebuah sensor tambahan harus diletakkan selalu berada dalam cairan yang berfungsi untuk pembandingan. Kedua sensor yang berhadapan dengan pemanas digerakkan oleh sebuah aktuator secara perlahan-lahan dengan perintah naik atau turun secara bertahap. Mula-mula sensor diletakkan pada bagian paling atas, selanjutnya sensor suhu digerakkan ke bawah perlahan-lahan, setiap terdeteksi adanya perubahan suhu pada sensor yang berhadapan pada pemanas berliku, maka dilakukan penambahan pencacahan terhadap pencacah elektronik. Pada saat sensor yang berhadapan dengan pemanas lurus mendeteksi adanya perubahan dari panas ke dingin, maka hasil pencacahan ditampilkan pada peraga aeperti yang terlihat pada gambar 2.17, yang merupakan blok diagram pengolahan dan pendisplayan sensor level menggunakan cara termal.

Sensor level cairan dengan cara thermal ini biasanya digunakan pada tanki-tanki boiler, karena selain sebagai sensor level cairan, juga dapat dipergunakan untuk mendeteksi gradien perubahan suhu dalam cairan.



Gambar 2.16 Teknik Penyensoran Level Cairan Cara Thermal



Gambar 2.17. Blok Diagram Pengolahan dan Pendisplayan Sensor Level Menggunakan Cara Thermal

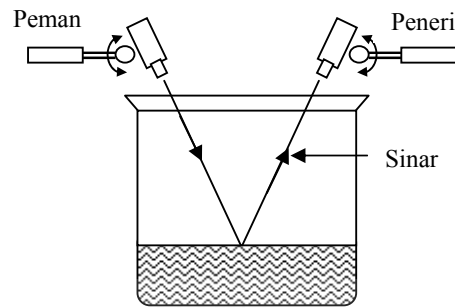
2.2.2.4 Menggunakan Cara Optik

Pengukuran level menggunakan optic didasarkan atas sifat pantulan permukaan atau pembiasan sinar dari cairan yang disensor. Ada beberapa carayang dapat digunakan untuk penyensoran menggunakan optic yaitu:

1. Menggunakan sinar laser
2. Menggunakan prisma

2.2.2.4.1 Menggunakan Sinar Laser

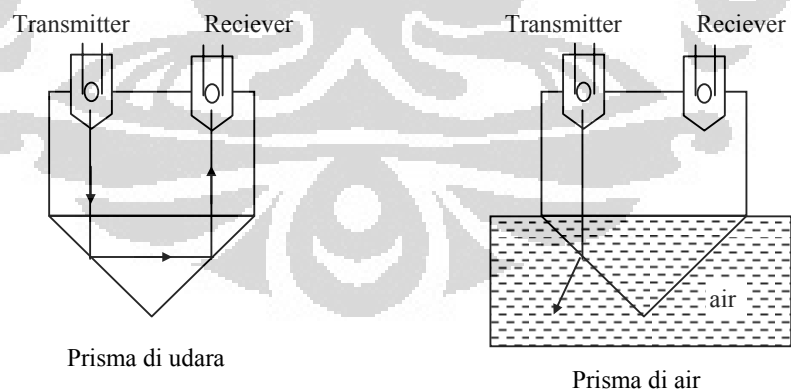
Pada gambar 2.18, Sinar laser dari sebuah sumber sinar diarahkan ke permukaan cairan, kemudian pantulannya dideteksi menggunakan detector sinar laser. Posisi pemancar dan detector sinar laser harus berada pada bidang yang sama. Detektor dan umber sinar laser diputar. Detektor diarahkan agar selalu berada pada posisi menerima sinar. Jika sinar yang datang diterima oleh detektor, maka level permukaan cairan dapat diketahui dengan menghitung posisi-posisi sudut dari sudut detektor dan sudut pemancar.



Gambar 2.18. Sensor Level menggunakan Sinar Laser

2.2.2.4.2 Menggunakan Prisma

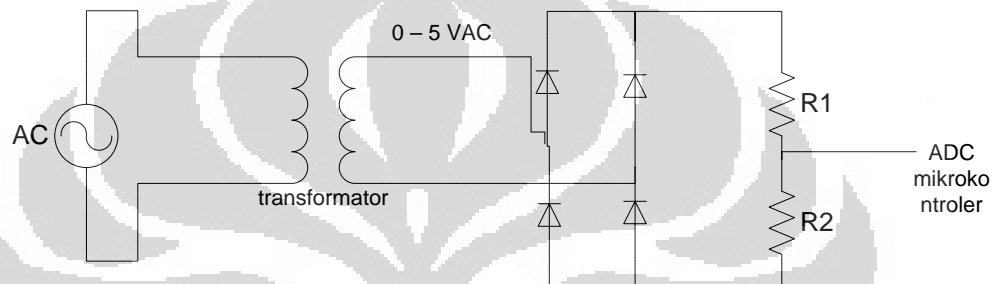
Teknik ini memanfaatkan harga yang berdekatan antara index bias air dengan index bias gelas. Sifat pantulan dari permukaan prisma akan menurun bila prisma dicelupkan kedalam air. Seperti yang terlihat pada gambar 2.19, prisma yang digunakan adalah prisma bersudut 45 dan 90 derajat. Sinar diarahkan ke prisma, bila prisma ditempatkan di udara, sinar akan dipantulkan kembali setelah melewati permukaan bawah prisma. Jika prisma ditempatkan di air, maka sinar yang dikirim tidak dipantulkan akan tetapi dibiaskan oleh air, Dengan demikian prisma ini dapat digunakan sebagai pengganti pelampung. Keuntungan yang diperoleh ialah dapat mereduksi ukuran sensor.



Gambar 2.19. Sensor Level menggunakan Prisma

2.3.3 Sensor Over Voltage

Pada saat Genset baru distart atau saat *running* biasanya sering terjadi lonjakan tegangan (*Over Voltage*), lonjakan tegangan ini dapat merusak peralatan listrik. Untuk menghindari kerusakan alat listrik karena terjadinya *Over voltage*. Maka pada Genset harus dilengkapi *Over voltage detector*, gunanya apabila terjadi *over voltage*, sistem dapat mendeteksinya dan memutuskan *switch* yang menghubungkan beban dengan Genset



Gambar 2.20. Over Voltage Detector

Gambar 2.20 merupakan rangkaian sederhana dari sebuah sensor *over voltage*. Pada sensor ini menggunakan sebuah transformator, dan penyerah tegangan AC ke DC, serta sebuah rangkaian pembagi tegangan.

2.3 Perkembangan Mikrokontroler

Pada awal perkembangannya (yaitu sekitar tahun 1970-an), sumber daya perangkat keras serta perangkat lunak mikrokontroler yang beredar masih sangat terbatas. Saat itu, sistem mikrokontroler hanya dapat diprogram secara khusus dengan perangkat yang dinamakan EPROM *programmer*. Sedangkan perangkat lunak yang digunakan umumnya berbasis bahasa assembler yang relatif sulit dipelajari.

Seiring dengan perkembangan teknologi *solid state* dan perangkat lunak komputer secara umum, saat ini pemrograman sistem mikrokontroler dirasakan relatif mudah dilakukan, terutama dengan digunakannya metode pemrograman *In system Programming* (ISP). Dengan menggunakan metode ini kita dapat memprogram sistem mikrokontroler sekaligus mengujinya pada sistem minimum

atau papan pengembang (*development board*) secara langsung tanpa perlu lagi perangkat “pembakar“ program atau emulator secara terpisah. Selain itu, ditinjau dari aspek perangkat lunak pemrogramannya, dewasa ini banyak alternatif bahasa aras tinggi dari pihak ke-tiga, baik gratis maupun komersil yang dapat digunakan. Penggunaan bahasa aras tinggi ini (seperti Pascal, C, basic dan sebagainya) selain akan menghemat waktu pengembangan, kode program yang disusun juga akan bersifat lebih modular dan terstruktur.

Dengan berbagai macam kelebihan yang dimiliki serta hal-hal yang menjadi bahan pertimbangan, dewasa ini mikrokontroler AVR 8 bit produk perusahaan Atmel adalah salah satu mikrokontroler yang banyak merebut minat kalangan profesional dan juga cocok dijadikan sarana berlatih bagi para pemula. Hal ini selain karena ragam fitur yang ditawarkan, juga disebabkan kemudahan untuk memperoleh mikrokontroler tersebut (berikut papan pengembangnya) di pasaran dengan harga yang relatif murah. Selain itu berkaitan dengan rancangan arsitekturnya, mikrokontroler AVR ini juga cocok diprogram dengan menggunakan bahasa pemrograman aras tinggi (terutama bahasa C).

2.4 Mikrokontroler Keluarga AVR.

Secara historis mikrokontroler seri AVR pertama kali diperkenalkan ke pasaran sekitar tahun 1997 oleh perusahaan Atmel, yaitu sebuah perusahaan yang sangat terkenal dengan produk mikrokontroler seri AT89S51/52-nya yang sampai sekarang masih banyak digunakan di lapangan. Tidak seperti mikrokontroler seri AT89S51/52 yang masih mempertahankan arsitektur dan set instruksi dasar mikrokontroler 8031 dari perusahaan INTEL. Mikrokontroler AVR ini diklaim memiliki arsitektur dan set instruksi yang benar-benar baru dan berbeda dengan arsitektur mikroontroler sebelumnya yang diproduksi oleh perusahaan tersebut. Konsep dan istilah-istilah dasarnya hampir sama, pemrograman level *assembler*-nya pun relatif tidak jauh berbeda.

Berdasarkan arsitekturnya, AVR merupakan mikrokontroler RISC (*Reduce Instruction Set Computer*) dengan lebar bus data 8 bit. Berbeda dengan sistem AT89S51/52 yang memiliki frekuensi kerja seperduabelas kali frekuensi oscilator, frekuensi kerja mikrokontroler AVR ini pada dasarnya sama dengan frekuensi

oscillator, sehingga hal tersebut menyebabkan kecepatan kerja AVR untuk frekuensi *oscillator* yang sama, akan dua belas kali lebih cepat dibandingkan dengan mikrokontroler keluarga AT89S51/52.

Dengan instruksi yang sangat variatif (mirip dengan sistem CISC-*Complex Instruction Set Computer*) serta jumlah register serbaguna (*general Purpose Register*) sebanyak 32 buah yang semuanya terhubung secara langsung ke ALU (*Arithmetic Logic Unit*), kecepatan operasi mikrokontroler AVR ini dapat mencapai 16 MIPS (enam belas juta instruksi per detik) sebuah kecepatan yang sangat tinggi untuk ukuran mikrokontroler 8 bit yang ada di pasaran sampai saat ini.

Untuk memenuhi kebutuhan dan aplikasi industri yang sangat beragam, mikrokontroler keluarga AVR ini muncul di pasaran dengan tiga seri utama: tinyAVR, ClasicAVR (AVR), megaAVR. Berikut ini beberapa seri yang dapat kita jumpai di pasaran:

ATtiny13	AT90S2313	ATmega103
ATtiny22	AT90S2323	ATmega128
ATtiny22L	AT90S2333	ATmega16
ATtiny2313	AT90S4414	ATmega162
ATtiny2313V	AT90S4433	ATmega168
ATtiny26	AT90S8515	ATmega8535

Keseluruhan seri AVR ini pada dasarnya memiliki organisasi memori dan set instruksi yang sama (sehingga dengan demikian jika kita telah mahir menggunakan salah satu seri AVR, untuk beralih ke seri yang lain akan relatif mudah). Perbedaan antara tinyAVR, AVR dan megaAVR pada kenyataannya hanya merefleksikan tambahan-tambahan fitur yang ditawarkannya saja (misal adanya tambahan ADC internal pada seri AVR tertentu, jumlah Port I/O serta memori yang berbeda, dan sebagainya). Diantara ketiganya, megaAVR umumnya memiliki fitur yang paling lengkap, disusul oleh AVR, dan terakhir tinyAVR.

Untuk memberi gambaran yang lebih jelas, tabel 2.1 berikut memperlihatkan perbedaan ketiga seri AVR ditinjau dari jumlah memori yang dimilikinya.

Tabel 2.1. Perbedaan seri AVR berdasarkan jumlah memori

Mikrokontroler AVR		Memori (byte)		
Jenis	Paket IC	Flash	EEPROM	SRAM
TinyAVR	8–32 pin	1 – 2K	64 – 128	0 – 128
AVR (classic AVR)	20–44 pin	1 – 8K	128 – 512	0 – 1 K
MegaAVR	32–64 pin	8 – 128 K	512 – 4 K	512 – 4 K

Seperti terlihat pada tabel tersebut, Semua jenis AVR ini telah dilengkapi dengan memori *flash* sebagai memori program. Tergantung serinya, kapasitas memori *flash* yang dimiliki bervariasi dari 1K sampai 128 KB. Secara teknis, memori jenis ini dapat diprogram melalui saluran antarmuka yang dikenal dengan nama *Serial Peripheral Interface* (SPI) yang terdapat pada setiap seri AVR tersebut. Dengan menggunakan perangkat lunak programmer (*downloader*) yang tepat, pengisian memori *Flash* dengan menggunakan saluran SPI ini dapat dilakukan bahkan ketika *chip* AVR telah terpasang pada sistem akhir (*end system*), sehingga dengan demikian pemrogramannya sangat fleksibel dan tidak merepotkan pengguna (Secara praktis metoda ini dikenal dengan istilah *In System Programming* (ISP), sedangkan perangkat lunaknya dinamakan *In System Programmer*).

Untuk penyimpanan data, mikrokontroler AVR menyediakan dua jenis memori yang berbeda: *Electrically Erasable Programmable Read Only Memory* (EEPROM) dan *Static Random Access memory* (SRAM). EEPROM umumnya digunakan untuk menyimpan data-data program yang bersifat permanen, sedangkan SRAM digunakan untuk menyimpan data variabel yang dimungkinkan berubah setiap saat. Kapasitas simpan data kedua memori ini bervariasi tergantung pada jenis AVR-nya (lihat tabel 2.1). Untuk seri AVR yang tidak memiliki SRAM, penyimpanan data variabel dapat dilakukan pada *register* serbaguna yang terdapat pada CPU mikrokontroler tersebut.

Selain seri-seri diatas yang sifatnya lebih umum, Perusahaan Atmel juga memproduksi beberapa jenis mikrokontroler AVR untuk tujuan yang lebih khusus dan terbatas, seperti seri AT86RF401 yang khusus digunakan untuk aplikasi

wireless remote control dengan menggunakan gelombang radio (RF), seri AT90SC yang khusus digunakan untuk peralatan sistem-sistem keamanan kartu SIM GSM, pembayaran via internet, dan lain sebagainya.

2.5 Perlengkapan Dasar Mikrokontroler

2.5.1 CPU

Unit pengolah pusat, *Central Processing Unit* (CPU) terdiri atas dua bagian yaitu unit pengendali *Controlling Unit* (CU) serta unit aritmatika dan logika *Aritmatic Logic Unit* (ALU). Fungsi utama unit pengendali adalah untuk mengambil, mengkode, dan melaksanakan urutan instruksi sebuah program yang tersimpan dalam memori. Sedangkan unit aritmatika dan perhitungan bertugas untuk menangani operasi perhitungan maupun boolean dalam program.

2.5.2 Alamat

Pada mikroprosesor/mikrokontroler, apabila suatu alat dihubungkan dengan mikrokontroler maka harus ditetapkan terlebih dahulu alamat (*address*) dari alat tersebut secara unik. Untuk menghindari terjadinya dua alat bekerja secara bersamaan yang mungkin akan menyebabkan kerusakan.

2.5.3 Data

Mikrokontroler ATmega16 mempunyai lebar bus data 16 bit. Merupakan mikrokontroler CMOS 16 bit daya-rendah berbasis arsitektur RISC yang ditingkatkan.

2.5.4 Pengendali

Selain bus alamat dan bus data mikroprosesor/mikrokontroler dilengkapi juga dengan bus pengendali (*control bus*), yang fungsinya untuk sinkronisasi operasi mikroprosesor/mikrokontroler dengan operasi rangkaian luar.

2.5.5 Memori

Mikroprosesor/mikrokontroler memerlukan memori untuk menyimpan program/data. Ada beberapa tingkatan memori, diantaranya *register internal*,

memori utama, dan memori massal. Sesuai dengan urutan tersebut waktu aksesnya dari yang lebih cepat ke yang lebih lambat.

2.5.6 RAM

Random Acces Memory (RAM) adalah memori yang dapat dibaca atau ditulis. Data dalam RAM akan terhapus bila catu daya dihilangkan. Oleh karena itu program mikrokontroler tidak disimpan dalam RAM. Ada dua teknologi yang dipakai untuk membuat RAM, yaitu RAM *static* dan RAM *dynamic*.

2.5.7 ROM

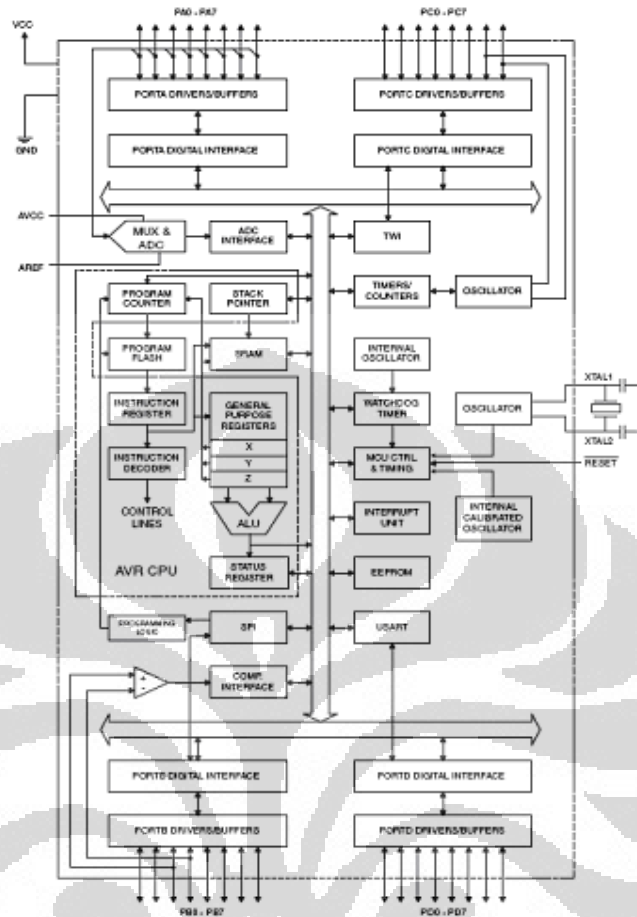
Read Only Memory (ROM) merupakan memori yang hanya dapat dibaca. Data dalam ROM tidak akan terhapus meskipun catu daya dimatikan. Oleh karena itu ROM dapat digunakan untuk menyimpan program. Ada beberapa jenis ROM antara lain ROM murni, PROM, EPROM, EAPROM. ROM adalah memori yang sudah diprogram oleh pabrik, PROM dapat diprogram oleh pemakai sekali saja. Sedangkan EPROM merupakan PROM yang dapat diprogram ulang.

2.5.8 Input / Output

Input output (I/O) dibutuhkan untuk melakukan hubungan dengan piranti di luar sistem. I/O dapat menerima data dari alat lain dan dapat pula mengirim data ke alat lain. Ada dua perantara I/O yang dipakai, yaitu piranti untuk hubungan serial (UART) dan piranti untuk hubungan paralel (PIO).

2.6 Arsitektur ATmega16

Dari gambar 2.21 dapat dilihat bahwa ATmega16 memiliki bagian sebagai berikut



Gambar 2.21. Blok Diagram ATMEGA16

- Saluran I/O sebanyak 32 buah, yaitu port A, Port B, Port C, dan Port D
- ADC (*Analog to Digital Converter*) 10 bit sebanyak 8 chanel
- Tiga buah timer/counter dengan kemampuan perbandingan.
- CPU yang terdiri dari 32 buah register.
- 131 instruksi andal yang umumnya hanya membutuhkan 1 siklus clock.
- Watchdog Timer dengan osilator internal
- Dua buah timer/counter 8 bit
- Satu buah timer/counter 16 bit
- Tegangan operasi 2.7V- 5.5V pada ATMEGA16
- Internal SRAM sebesar 1KB

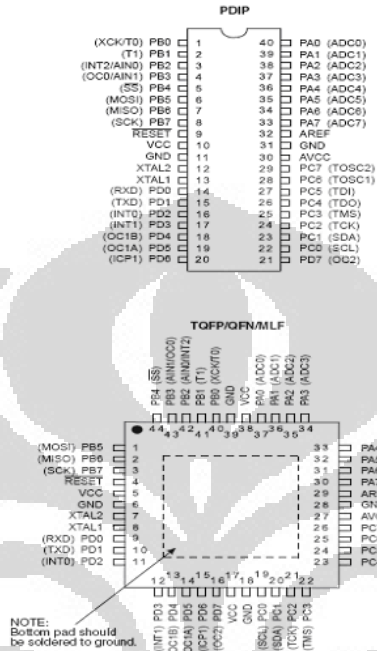
- Memori flash sebesar 16 KB dengan kemampuan Read While Write
- Unit interupsi internal dan eksternal.
- Port antar muka SPI
- EEPROM sebesar 512 byte yang dapat deprogram saat operasi
- Antarmuka komparator analog
- 4 chanel PWM
- 3x8 general purpose register
- Hampir mencapai 16 MIPS pada kristal 16 Mhz
- Port USART programable untuk komunikasi serial.

2.7 Konfigurasi Pin ATmega16

Gambar 2.22 merupakan susunan kaki standar 40 pin DIP mikrokontroler AVR ATmega16.

- VCC merupakan pin masukan positif catu daya. Setiap peralatan elektronika digital tentunya butuh sumber catudaya yang umumnya sebesar 5 V, itulah sebabnya di PCB kit mikrokontroler selalu ada IC regulator 7805.
- GND sebagai pin Ground
- Port A (PA0...PA7) merupakan pin I/O dua arah dan dapat deprogram sebagai pin masukan ADC.
- Port B (PB0...PB7) merupakan pin I/O dua arah dan pin fungsi khusus, yaitu timer/counter, komparator analog, dan SPI.
- Port C (PC0...PC7) merupakan pin I/O dua arah dan pin fungsi khusus, yaitu TWI, komparator analog, dan timer Osilator.
- Port D (PD0...PD7) merupakan pin I/O dua arah dan pin fungsi khusus, yaitu komparator analog, interupsi eksternal, dan komunikasi serial.
- Reset merupakan pin yang digunakan untuk mereset mikrokontroler.
- XTAL 1 dan XTAL 2 sebagai pin masukan clock eksternal. Suatu mikrokontroler membutuhkan sumber clock agar dapat mengeksekusi intruksi yang ada di memori. Semakin tinggi nilai kristalnya, maka semakin cepat mikrokontroler tersebut.

- AVCC sebagai pin masukan tegangan untuk ADC
- AREF sebagai masukan tegangan referensi.



Gambar 2.22. Tata Letak Kaki ATMEGA16

2.8 Mikrokontroler AVR dan Bahasa C

Tak dapat disangkal, dewasa ini penggunaan bahasa pemrograman aras tinggi (seperti C, Basic, Pascal, Forth dan sebagainya) semakin populer dan banyak digunakan untuk memprogram sistem mikrokontroler. Berdasarkan sifatnya yang sangat fleksibel dalam hal keleluasaan pemrogram untuk mengakses perangkat keras, Bahasa C merupakan bahasa pemrograman yang paling cocok dibandingkan bahasa-bahasa pemrograman aras tinggi lainnya.

Bahasa C merupakan salah satu bahasa pemrograman yang paling populer untuk pengembangan program-program aplikasi yang berjalan pada sistem *microprocessor* (komputer). Karena kepopulerannya, vendor-vendor perangkat lunak kemudian mengembangkan *compiler* C sehingga menjadi beberapa varian berikut: Turbo C, Borland C, Microsoft C, Power C, Zortech C dan lain

sebagainya. Untuk menjaga portabilitas, compiler-compiler C tersebut menerapkan ANSI C (ANSI: *American National Standards Institute*) sebagai standar bakunya. Perbedaan antara *compiler-compiler* tersebut umumnya hanya terletak pada pengembangan fungsi-fungsi *library* serta fasilitas *Integrated Development Environment (IDE)* nya saja.

Relatif dibandingkan dengan bahasa aras tinggi lain, bahasa C merupakan bahasa pemrograman yang sangat fleksibel dan tidak terlalu terikat dengan berbagai aturan yang sifatnya kaku. Satu-satunya hal yang membatasi penggunaan bahasa C dalam sebuah aplikasi adalah semata-mata kemampuan imajinasi *programmer*-nya saja. Sebagai ilustrasi, dalam program C kita dapat saja secara bebas menjumlahkan karakter huruf (misal 'A') dengan sebuah bilangan bulat (misal '2'), dimana hal yang sama tidak mungkin dapat dilakukan dengan menggunakan bahasa aras tinggi lainnya. Karena sifatnya ini, seringkali bahasa C dikategorikan sebagai bahasa aras menengah (*mid level language*).

Dalam kaitannya dengan pemrograman mikrokontroler, Tak pelak lagi bahasa C saat ini mulai menggeser penggunaan bahasa aras rendah *assembler*. Penggunaan bahasa C akan sangat efisien terutama untuk program mikrokontroler yang berukuran relatif besar. Dibandingkan dengan bahasa *assembler*, penggunaan bahasa C dalam pemrograman memiliki beberapa kelebihan berikut: Mempercepat waktu pengembangan, bersifat modular dan terstruktur, sedangkan kelemahannya adalah kode program hasil kompilasi akan relatif lebih besar (dan sebagai konsekuensinya hal ini terkadang akan mengurangi kecepatan eksekusi).

Khusus pada mikrokontroler AVR, untuk mereduksi konsekuensi negatif diatas, Perusahaan Atmel merancang sedemikian sehingga arsitektur AVR ini efisien dalam mendekode serta mengeksekusi instruksi-instruksi yang umum dibangkitkan oleh *compiler* C (Dalam kenyataannya, pengembangan arsitektur AVR ini tidak dilakukan sendiri oleh perusahaan Atmel tetapi ada kerja sama dengan salah satu vendor pemasok *compiler* C untuk mikrokontroler tersebut, yaitu IAR C.

Tabel 2.2. Beberapa Compiler C untuk mikrokontroler AVR

Compiler C	Platform	Keterangan
IAR C	-DOS -Windows	Komersil
CodeVisionAVR	-Windows	Komersil,
ImageCraft's C	-DOS -Windows -Linux	Komersil
AVR-GCC	-DOS -Windows	General Public Licence
C-AVR	-Windows	Komersil
Small C for AVR	-DOS	Komersil
GNU C for AVR	-Linux	General Public Licence
LCC-AVR	-Linux, -Windows	Free
Dunfields AVR	-Windows	Komersil

Seperti halnya *compiler C* untuk sistem *microprocessor*, di pasaran ada beberapa varian *compiler C* untuk memprogram sistem mikrokontroler AVR yang dapat dijumpai (lihat tabel 2.5).

Dengan beberapa kelebihan yang dimilikinya, saat ini CodeVisionAVR produk Perusahaan Pavel Haiduc merupakan *compiler C* yang relatif banyak digunakan dibandingkan *compiler-compiler C* lainnya.

2.9 Sekilas Tentang CodeVisionAVR

CodeVisionAVR pada dasarnya merupakan perangkat lunak pemrograman mikrokontroler keluarga AVR berbasis bahasa C. Ada tiga komponen penting yang telah diintegrasikan dalam perangkat lunak ini: *Compiler C*, IDE dan Program generator.

Berdasarkan spesifikasi yang dikeluarkan oleh perusahaan pengembangnya, *Compiler C* yang digunakan hampir mengimplementasikan semua komponen

standar yang ada pada bahasa C standar ANSI (seperti struktur program, jenis tipe data, jenis operator, dan *library* fungsi standar-berikut penamaannya). Tetapi walaupun demikian, dibandingkan bahasa C untuk aplikasi komputer, *compiler* C untuk mikrokontroler ini memiliki sedikit perbedaan yang disesuaikan dengan arsitektur AVR tempat program C tersebut ditanamkan (*embedded*).

Khusus untuk *library* fungsi, disamping *library* standar (seperti fungsi-fungsi matematik, manipulasi *String*, pengaksesan memori dan sebagainya), CodeVisionAVR juga menyediakan fungsi-fungsi tambahan yang sangat bermanfaat dalam pemrograman antarmuka AVR dengan perangkat luar yang umum digunakan dalam aplikasi kontrol. Beberapa fungsi *library* yang penting diantaranya adalah fungsi-fungsi untuk pengaksesan LCD, komunikasi I²C, IC *Real time Clock* (RTC), sensor suhu LM75, *Serial Peripheral Interface* (SPI) dan lain sebagainya.

Untuk memudahkan pengembangan program aplikasi, CodeVisionAVR juga dilengkapi IDE yang sangat *user friendly*. Selain menu-menu pilihan yang umum dijumpai pada setiap perangkat lunak berbasis Windows, CodeVisionAVR ini telah mengintegrasikan perangkat lunak *downloader* (*in system programmer*) yang dapat digunakan untuk mentransfer kode mesin hasil kompilasi ke dalam sistem memori mikrokontroler AVR yang sedang diprogram.

Selain itu, CodeVisionAVR juga menyediakan sebuah tool yang dinamakan dengan *Code Generator* atau CodeWizardAVR. Secara praktis, *tool* ini sangat bermanfaat membentuk sebuah kerangka program (*template*), dan juga memberi kemudahan bagi *programmer* dalam peng-inisialisasian *register-register* yang terdapat pada mikrokontroler AVR yang sedang diprogram. Dinamakan *Code Generator*, karena perangkat lunak CodeVision ini akan membangkitkan kode-kode program secara otomatis setelah fase inisialisasi pada jendela CodeWizardAVR selesai dilakukan.

CodeVisionAVR merupakan *software C-cross compiler*, dimana program dapat ditulis dengan bahasa C. Dengan menggunakan pemrograman bahasa-C diharapkan waktu disain (*deleloping time*) akan menjadi lebih singkat. Setelah program dalam bahasa-C ditulis dan dilakukan kompilasi tidak terdapat kesalahan

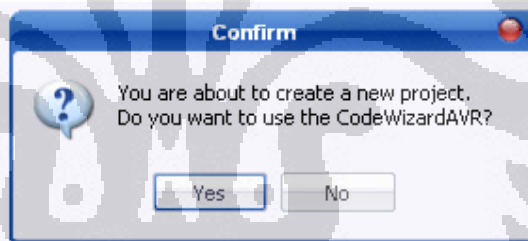
(*error*) maka proses download dapat dilakukan. Mikrokontroler AVR mendukung sistem download secara ISP.

1. Untuk memulai bekerja dengan CodeVisionAVR pilih pada menu **File > New**. Maka akan muncul kotak dialog seperti pada gambar dibawah :



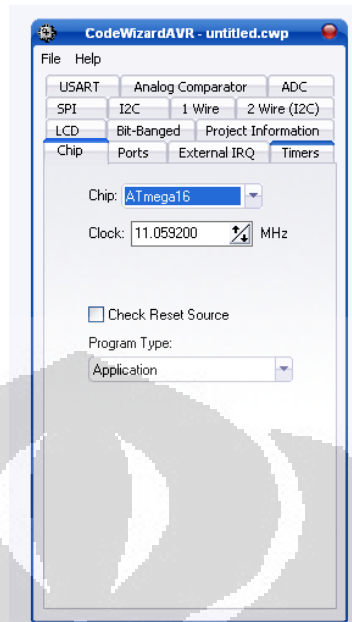
Gambar 2.23. tampilan *New File*

2. Pilih *Project* lalu klik OK, kemudian akan muncul kotak dialog seperti dibawah ini :

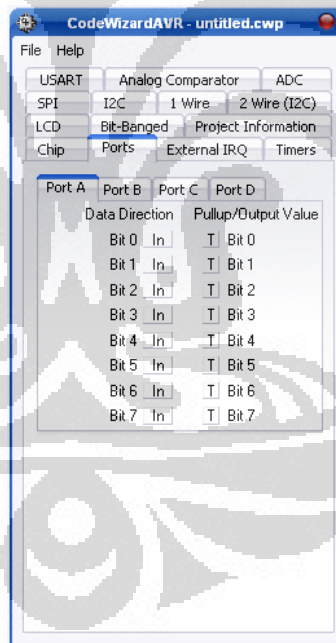


Gambar 2.24. Tampilan *Option di Wizard CVAVR*

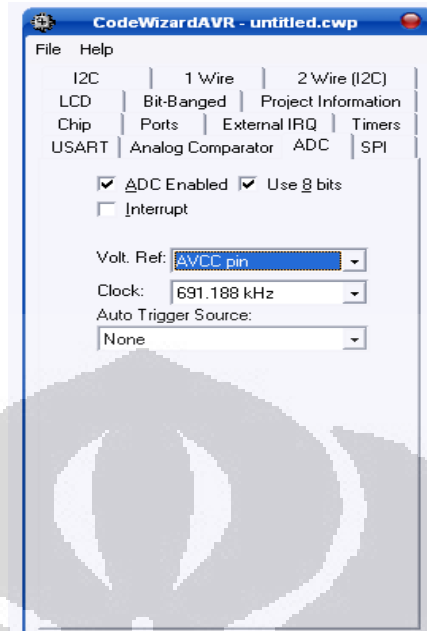
3. Pilih **Yes** untuk menggunakan CodeWizardAVR. CodeWizardAVR digunakan untuk membantu dalam *generate* program, terutama dalam konfigurasi *chip* mikrokontroler, baik itu konfigurasi Port, Timer, penggunaan fasilitas-fasilitas seperti LCD, *interrupt*, dan sebagainya seperti yang terlihat pada gambar 2.25 sampai 2.28 . CodeWizardAVR ini sangat membantu *programmer* untuk setting *chip* sesuai keinginan. Berikut ini tampilan CodeWizardAVR untuk *setting Chip* dan *Port* dari mikrokontroler



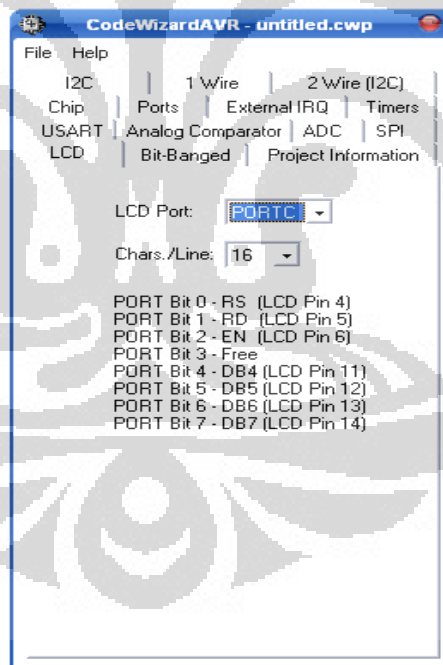
Gambar 2.25. Tampilan *IC port*



Gambar 2.26. Tampilan *Setting Port CAVR*



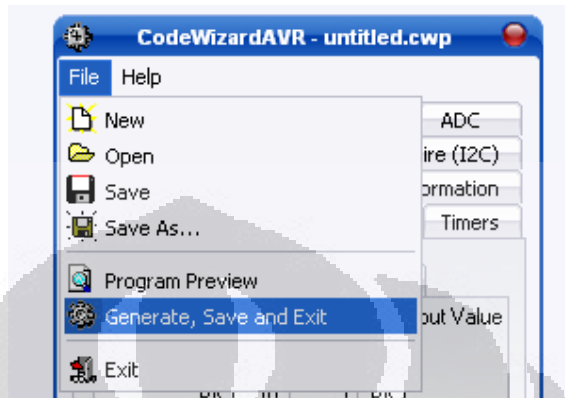
Gambar 2.27. *Setting ADC*



Gambar 2.28. *Setting LCD*

4. Untuk selanjutnya fasilitas-fasilitas lainnya dapat disetting sesuai kebutuhan dari pemrograman. Setelah selesai dengan CodeWizardAVR, selanjutnya pada menu File, pilih Generate,

Save and Exit dan simpan pada direktori yang diinginkan seperti yang ditunjukkan gambar 2.29.



Gambar 2.29. Tampilan *Save Generate Project*

2.10 Konsep I/O pada mikrokontroler AVR ATmega16

Pada mikrokontroler ATmega16 terdapat empat buah *port* yaitu PA, PB, PC, dan PD yang semuanya dapat diprogram sebagai input maupun *output*. Pin I/O pada mikrokontroler AVR dapat dikonfigurasi sebagai input atau output, dengan cara mengubah isi *I/O register Data Direction Register*. Misalnya, jika ingin port B dikonfigurasi sebagai output, maka Data Direction Register port B (DDRB) harus diset sebagai 0xFFH (sama dengan 255). Jika sebagai input, maka diset 0x00H (sama dengan 0).

V_{OH} (*output high voltage*) ialah tegangan pada pin I/O mikrokontroler ketika ia mengeluarkan logika “1” dengan besar sekitar 4.2 V dan arus sebesar 20 mA (I_{OH}). Setiap pin I/O mikrokontroler AVR memiliki *internal pull up*. Misalnya Port B dikonfigurasi sebagai input dan *internal pull-up*nya diaktifkan, maka $DDRB=00H$ dan $PORTB=00H$. Untuk mendeteksi input pada salah satu *port*, dapat digunakan fungsi $PINx$, sedangkan untuk mendeteksi per pin pada suatu *port* dapat digunakan fungsi $Pinx.bit$.

2. 11 Konsep Liquid Crystal Display

Liquid Crystal Display (LCD) ialah modul penampil yang banyak digunakan karena tampilanya yang menarik. LCD yang paling digunakan saat ini adalah LCD M1632 *refurbish* karena harganya cukup murah. LCD M1602 merupakan modul dengan tampilan 2x16 (2 baris x 16 kolom) dengan konsumsi daya rendah. Modul tersebut dilengkapi dengan mikrokontroler yang didesain khusus untuk mengendalikan LCD. *Chip* S6A0069 yang berfungsi sebagai pengendali LCD memiliki *Character Generator Read Only Memory* (CGPROM), *Character Generator Random Access Memory* (CGRAM), dan *Display Data Random Access Memor* (DDRAM).

LCD yang umum ada yang panjangnya hingga 40 karakter (2x40 dan 4x40), dimana kita menggunakan DDRAM untuk mengatur tempat penyimpanan karakter tersebut.

CGRAM merupakan memori untuk menggambarkan pola sebuah karakter, dimana bentuk dari karakter dapat diubah-ubah sesuai dengan keinginan. Namun, memori akan hilang saat *power supply* tidak aktif sehingga pola karakter akan hilang.

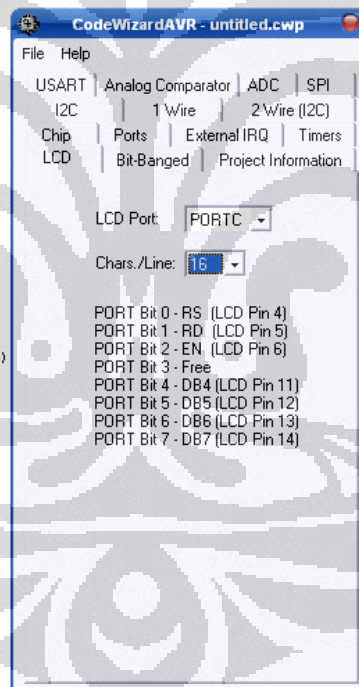
Driver LCD seperti S6A0069 memiliki dua register yang askesnya diatur menggunakan pin RS. Pada saat RS berlogika 0 (*Low*), *register* yang diaskes adalah perintah, sedangkan pada saat RS berlogika 1(*high*), *register* yang diaskes adalah *register* data. Berikut table pin untuk LCD M1602

Tabel 2.3. Konfigurasi Pin LCD

No	Pin	Function
1	Vss	0V (GND)
2	Vcc	5V
3	VLC	LCD Contras Voltage
4	RS	Register Select;H(1):Data input;L(0):Instructin input
5	RD	H(1): read; L(0): Write
6	EN	Enable signal
7	D0	

Table 2.3. Lanjutan

No	Pin	Function
8	D1	Data Bus
9	D2	
10	D3	
11	D4	
12	D5	
13	D6	
14	D7	
15	V+BL	Positif Backlight Voltage
16	V-BL	Negatif Backlight Voltage



Gambar 2.30. setting LCD menggunakan CodeVision AVR

D1-D7 pada LCD berfungsi untuk menerima data dari mikrokontroler. Untuk menerima data, pin 5 pada LCD (RD) harus berlogika 0 (*low*), dan berlogika 1 (*high*) untuk mengirim data mikrokontroler. Setiap kali menerima atau mengirim

data, untuk mengaktifkan LCD diperlukan sinyal E (*Chip Enable*) dalam bentuk perpindahan logika 1 ke logika 0.

Sedangkan pin RS (*Register Selector*), berguna untuk memilih *Instruction Register* (IR) atau *Data Register* (DR). Jika nilai RS *Low* (0) dan RD *Low* (0), maka akan dilakukan operasi penulisan data ke DDRAM atau CGRAM. Sedangkan jika RS berlogika *High*(1) dan RD berlogika *High* (1), akan membaca data dari DDRAM atau CGRAM ke *register* DR. Karakter yang akan ditampilkan ke display disimpan di memori DDRAM. Lokasi karakter yang akan ditampilkan ke display mempunyai alamat tertentu pada memori DDRAM.

Misalnya, alamat DDRAM 00H berisi data 30H (nilai ASCII untuk angka 0), maka akan tampil dibaris 1 kolom 1 angka 0. proses penampilannya dengan cara: kontroler LCD mengambil data pada alamat DDRAM 00H. Data 30H digunakan sebagai alamat untuk mengambil data *display* LCD pada memori CGROM atau CGRAM, kemudian data pada CGROM/CGRAM diambil dan ditampilkan ke display.

Fungsi *busy flag* pada LCD untuk indikator apakah kontroler LCD sudah siap menerima perintah atau data selanjutnya. Jika *busy flag* berlogika 1, maka perintah atau data yang dikirim oleh mikrokontroler tidak akan diproses, tetapi jika berlogika 0, maka perintah atau data yang dikirim oleh mikrokontroler akan diproses.

Untuk menampilkan karakter atau *string* ke LCD tidak sulit, karena didukung pustaka yang telah disediakan oleh CodeVision AVR. Kita tidak harus memahami secara mendalam karakteristik LCD. Perintah tulis dan inisialisasi sudah disediakan oleh *library* LCD dari CodeVision AVR. Berikut adalah cara *setting* LCD menggunakan CodeVision AVR

2.12 Kosep *Analog to Digital Converter* (ADC)

Keunggulan mikrokontroler AVR ATMEGA16 dibandingkan pendahulunya adalah :

- ❖ Sudah terintegrasinya ADC 10 bit sebanyak 8 saluran

- ❖ 13-260 μ S *conversion time*
- ❖ Mencapai 15kSPs pada resolusi maksimum
- ❖ *Optional left adjustment* untuk ADC result readout
- ❖ Interupsi pada *ADC Conversion Complete*
- ❖ *Sleep Mode noise canceler*

Input ADC pada mikrokontroler dihubungkan 8 *channel* Analog *multiplexer* yang digunakan untuk *single ended input channels*. Jika sinyal input dihubungkan ke masukan ADC dan satu jalur lagi terhubung ke *Ground*, maka disebut *single ended input*. Jika input ADC terhubung kedua buah input ADC, disebut sebagai *differential input*, yang dapat dikombinasikan sebanyak 16 kombinasi. Empat kombinasi terpenting antara lain kombinasi input diferensial (ADC0 dengan ADC1 dan ADC2 dengan ADC3) dengan penguatan yang dapat diatur. ADC0 dan ADC2 sebagai tegangan input negatif, sedangkan ADC1 dan ADC3 sebagai tegangan input positif. Besar penguatan yang dapat dibuat yaitu 20dB (10x) atau 46dB (200x) pada tegangan input diferensial sebelum proses konversi ADC.

Proses inisialisasi ADC meliputi proses penentuan *clock*, tegangan referensi, format output data, dan mode pembacaan. Register yang perlu diset nilainya adalah *ADC Multiplexer Selection Register (ADMUX)*, *ADC Control and Status Register (ADCSRA)*, dan *Special Function IO Register (SFIO)*. ADMUX merupakan register 8 bit yang berfungsi menentukan tegangan referensi ADC, format data output, dan saluran ADC yang digunakan.

Untuk memilih *channel* ADC mana yang akan digunakan (*single ended* atau diferensial), atur nilai MUX4 :0, misalnya *channel* ADC0 sebagai input ADC, maka MUX4 :0 diberi nilai 00000B.

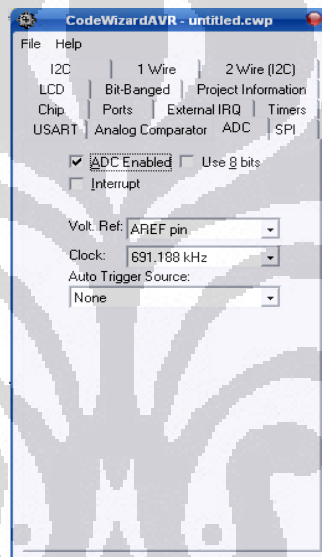
Tegangan referensi ADC dapat dipilih antara lain pada pin AREF, pin AVCC, atau menggunakan tegangan referensi internal sebesar 2.56V. Agar fitur ADC mikrokontroler dapat digunakan, maka ADEN (*ADC Enable*, dalam I/O register ADCSRA) harus diberi nilai 1.

Setelah konversi selesai (ADIF high), hasil konversi dapat diperoleh pada register hasil (ADCL, ADCH). Untuk konversi *single ended*, hasilnya ialah :

$$ADC = \frac{V_{in} \cdot 1024}{V_{ref}}$$

Dimana V_{in} ialah tegangan pada input yang dipilih dan V_{ref} merupakan tegangan referensi. Jika hasil ADC=000H, maka menunjukkan tegangan input sebesar 0V, jika hasil ADC=3FFH menunjukkan tegangan input sebesar tegangan referensi dikurangi 1 LSB.

Kita dapat mengkonfigurasi fasilitas ADC pada CodeVision AVR sebagai berikut seperti yang terlihat pada gambar 2.31 :



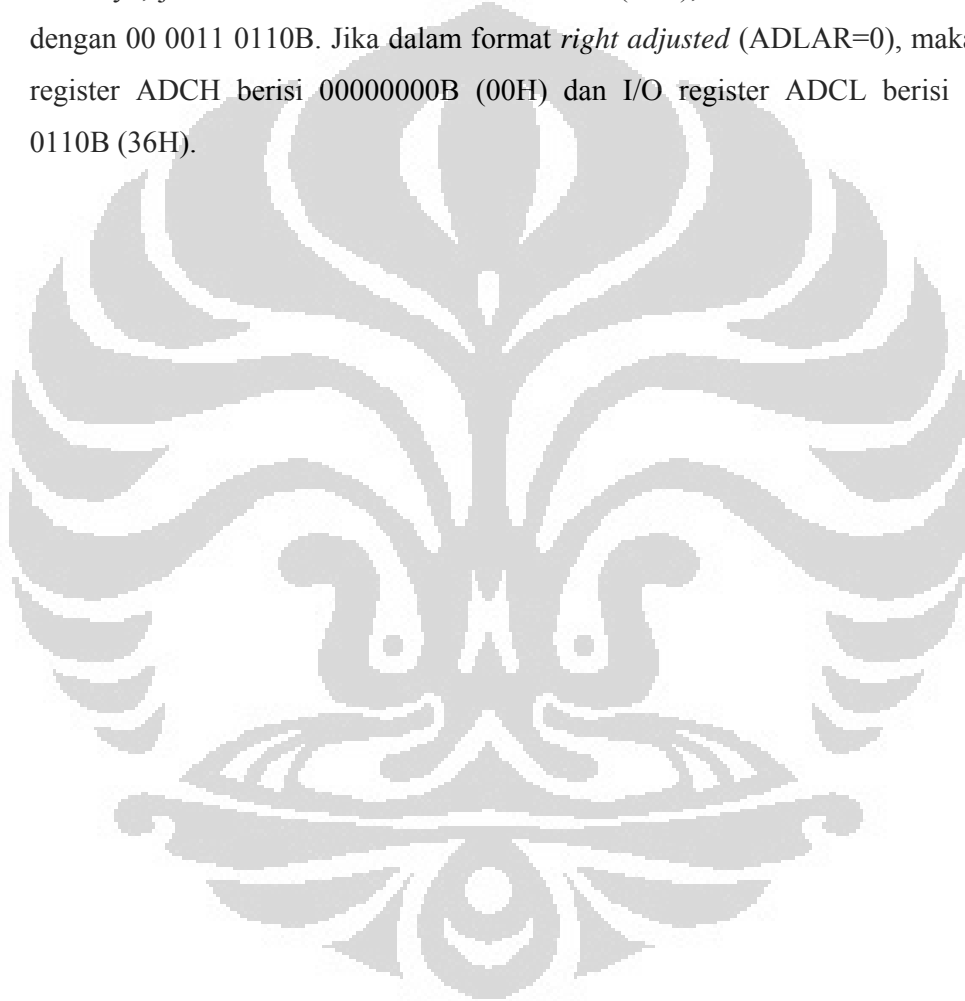
Gambar 2.31. *Setting ADC*

Sebagai contoh, jika diberikan V_{in} sebesar 0.2 V dengan V_{ref} 5V, maka hasil konversi ADC adalah 41. jika menggunakan differensial *channel*, hasilnya adalah 40.46, yang bila dikenakan bisa sekitar 39,40,41 karena ketelitian ADC ATMEGA16 sebesar +/- 2 LSB. Jika yang digunakan saluran differensial, maka hasilnya ialah :

$$ADC = \frac{(V_{positif} - V_{negatif}) \cdot GAIN \cdot 512}{V_{ref}}$$

Dimana V_{positif} ialah tegangan pada input pin positif, V_{negatif} ialah tegangan input pada pin negatif, GAIN ialah faktor penguatan, dan V_{ref} ialah tegangan referensi yang digunakan.

Dengan mencentang *ADC Enable* akan mengaktifkan *on-chip* ADC. Dengan mencentang *Use 8 bits*, maka hanya 8 bit terpenting yang digunakan. Hasil konversi 10 bit dapat dibaca pada ADC Data Register ADCH dan ADCL. Misalnya, jika hasil konversi ADC bernilai 54 (36H), dalam 10 bit biner ditulis dengan 00 0011 0110B. Jika dalam format *right adjusted* (ADLAR=0), maka I/O register ADCH berisi 00000000B (00H) dan I/O register ADCL berisi 0011 0110B (36H).



BAB III

PERANCANGAN DAN REALISASI ALAT

3.1 Dasar Perancangan

Dalam merancang suatu alat ada beberapa faktor yang harus dipertimbangkan baik aspek teknis maupun non-teknis. Aspek teknis berarti alat tersebut dibuat berdasarkan kebutuhan untuk menunjang suatu fungsi kerja tertentu dan harus memenuhi persyaratan-persyaratan tertentu. Sedangkan aspek non-teknis diartikan sebagai aspek diluar aspek teknis seperti segi ekonomis dan komersial.

Dalam skripsi ini pembuatan alat lebih ditekankan pada aspek teknis, tetapi walaupun demikian faktor ekonomis tetap ditentukan, seperti dalam pemilihan komponen yang digunakan.

Sesuai dengan fungsinya bahwa alat ini harus mampu untuk mengendalikan berbagai sistem pembangkit listrik, guna terciptanya penghematan energi listrik dari PLN, serta mengendalikan kerja GENSET secara cepat dan kontiniu, saat terjadi gangguan pada sumber listrik *solarcell* dan PLN. Maka alat ini harus mempunyai kemampuan-kemampuan sebagai berikut.

1. Keandalan yang tinggi, artinya tujuan yang ingin dicapai harus mempunyai faktor kegagalan yang rendah. Untuk itu alat ini harus mempunyai kualitas komponen yang baik. Selain dari itu gangguan yang terjadi harus dapat segera diketahui. Sehingga operator dengan segera bisa memperbaikinya serta menghindari kerusakan yang lebih fatal. Oleh karena itu alat ini harus mampu memberikan indikasi pada setiap gangguan yang terjadi. Adapun gangguan yang perlu di indifikasi yaitu adalah :

- Level bahan bakar mesin.
- Temperatur mesin
- Level Oli mesin
- *Over voltage*

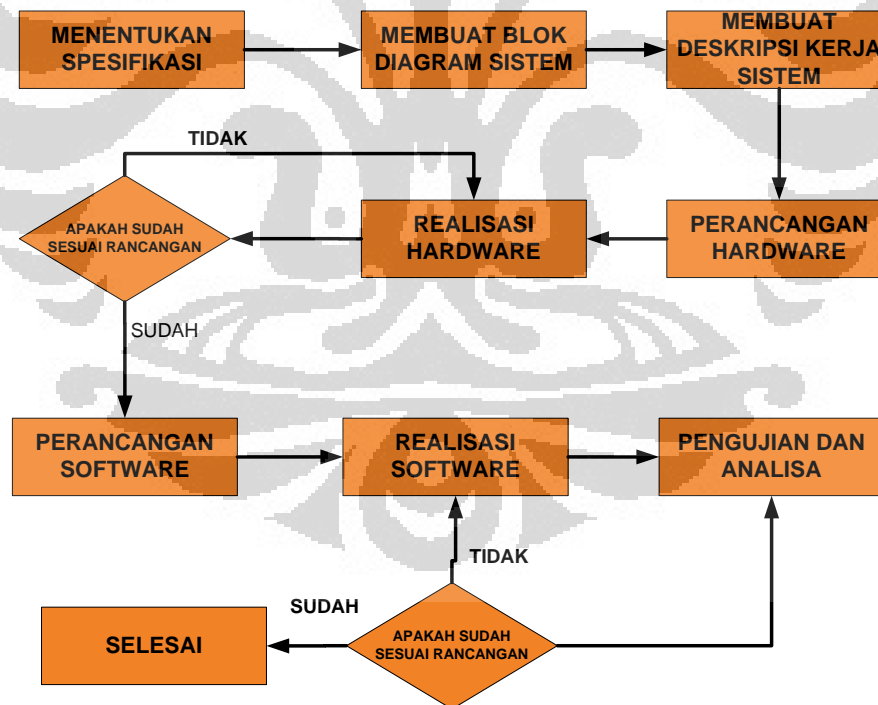
Bentuk indikasi yang dapat diberikan harus yang mudah diketahui baik dari ruang dimana alat ini ditempatkan atau dari tempat lain. Dan juga indikasi ini harus jelas memberikan informasi gangguan apa yang sedang

terjadi. Untuk memenuhi hal tersebut dapat digunakan sinyal dari tampilan LCD dan dari cahaya lampu indikator untuk memberikan indikasi gangguan yang sedang terjadi.

2. Mudah dalam pengoperasian, untuk memenuhi kriteria ini, maka perlu diperhatikan hal-hal sebagai berikut :
 - Pemasangan sakelar, LED, LCD, dan *push on-off* pada panel muka harus dibuat sedemikian rupa sehingga tidak membingungkan selama pengoperasian alat tersebut.
 - Penandaan dan penamaan komponen atau alat yang terpasang pada panel muka harus jelas dan singkat.

3.2 Tahapan Perancangan

Dalam perancangan terdapat beberapa tahapan yang akan dilaksanakan seperti ditunjukkan pada diagram alir dibawah ini



Gambar 3.1. Flowchart Tahapan Perancangan dan Realisasi

3.3 Spesifikasi Alat

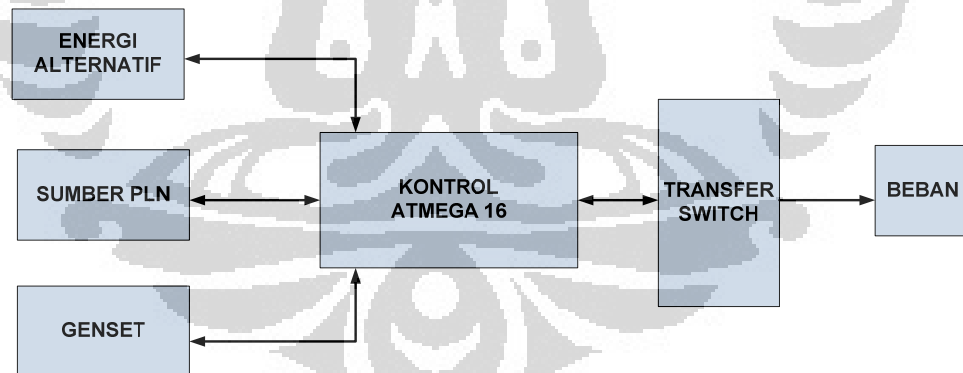
Adapun Alat yang akan dirancang memiliki spesifikasi :

- Daya GENSET : 0,9 KVA-2 KVA
- Daya Sistem pembangkit listrik *solarcell* : 0,5 KVA – 2 KVA
- Tegangan : 220 VAC / 1 PHASA
- Frekuensi : 50 Hz
- Kontroller : ATMEGA 16
- Rating tegangan output sensor : 0-5 V DC
- *Fault indicator* : LCD & Lampu Indikator
- *Fault detector* : *Fuel, Engine Temperatur, Over Voltage, Oil fault, Solar cell accu warning, PLN fault*
- *Operation* : *Manual & Automatic*

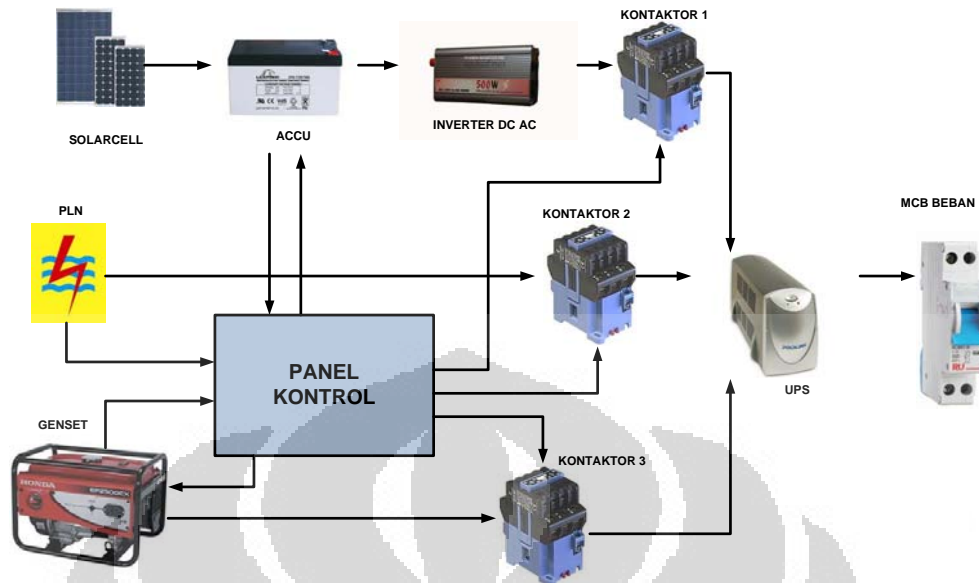
3.4 Diskripsi Kerja

3.4.1 Blok Diagram

Dibawah ini adalah blok diagram dari sistem pengontrolan



Gambar 3.2. Blok Diagram sistem kontrol

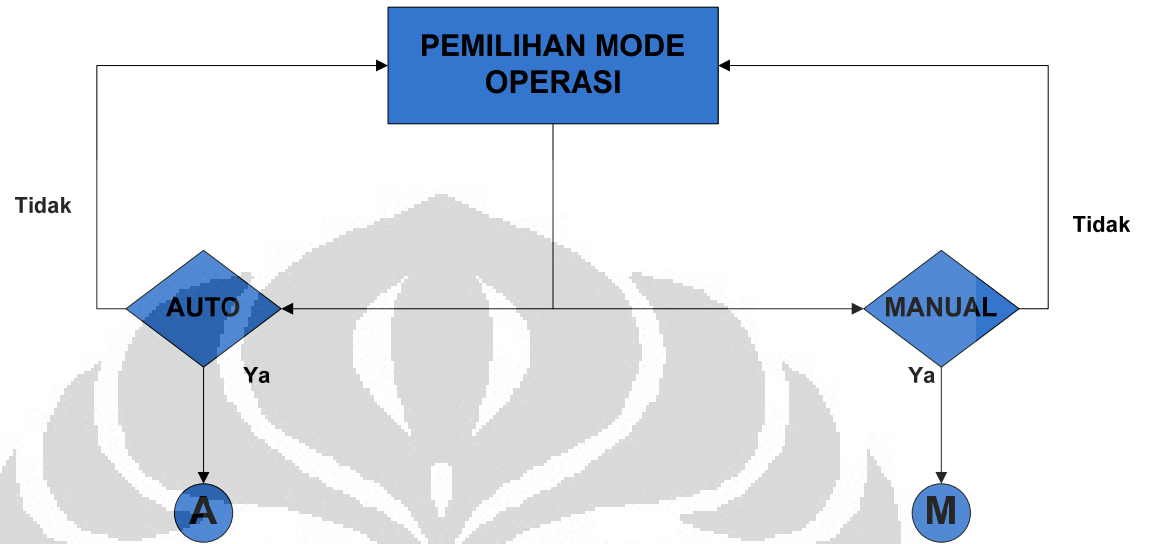


Gambar 3.3 . *Picture Diagram Sistem Kontrol*

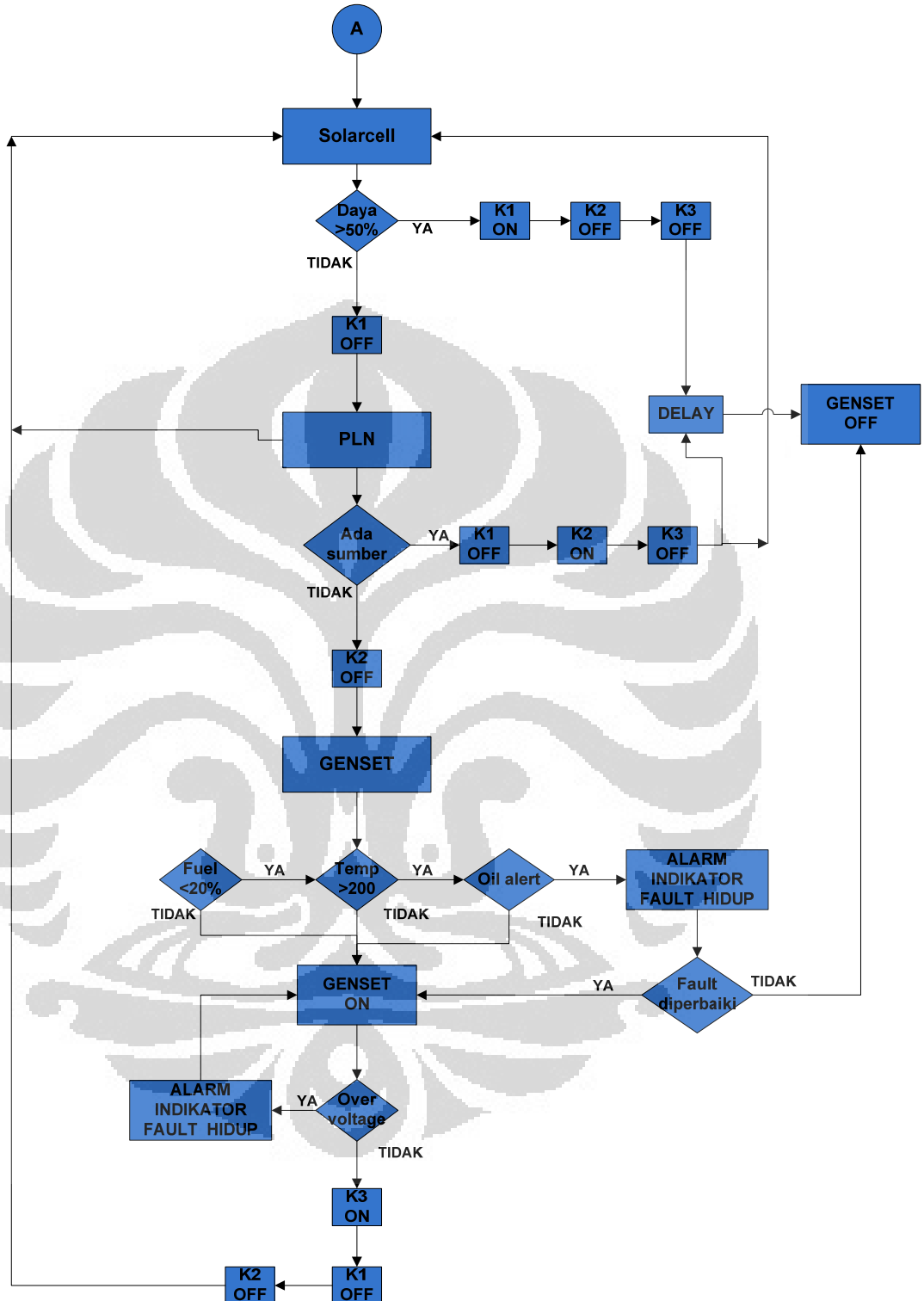
Panel kontrol berfungsi untuk mengatur *on / off* kontaktor K1, K2, K3. Kontaktor K1 berfungsi menghubungkan beban dengan pembangkit listrik *solarcell*, Kontaktor K2 menghubungkan beban dengan sumber listrik PLN, kontaktor K3 menghubungkan beban dengan GENSET. Disamping itu panel kontrol berfungsi untuk mendeteksi *fault* yang terjadi pada masing –masing sumber listrik. Pada sumber solarcell sistem kontrol dapat mendeteksi keadaan *accu solarcell* misalnya *accu* kekurangan daya untuk mensuplay listrik kebeban karena malam hari solarcell tidak bisa mengisi daya pada *accu*, sehingga sistem akan mengalihkan supplay listrik dari *solarcell* ke sumber PLN. Panel kontrol juga dilengkapi dengan sensor PLN, apabila sumber listrik dari PLN dalam keadaan *fault* (pemadaman) maka sistem kontrol akan mengalihkan supplay listrik beban ke pembangkit listrik yang lain, misalnya ke GENSET atau ke *solarcell*. Panel control ini juga dapat *menstart* dan *mengoff* kan mesin GENSET secara otomatis saat sumber listrik dari solarcell dan PLN tidak bisa mensupplay listrik ke beban serta mendeteksi *fault* yang terjadi pada GENSET, misalnya *oil fault*, *fuel fault*, *tempratur fault*, *voltage fault*. UPS disini berfungsi sebagai backup listrik saat proses *switching* antara ke 3 pembangkit listrik.

3.4.2 Flowchart Cara Kerja Alat

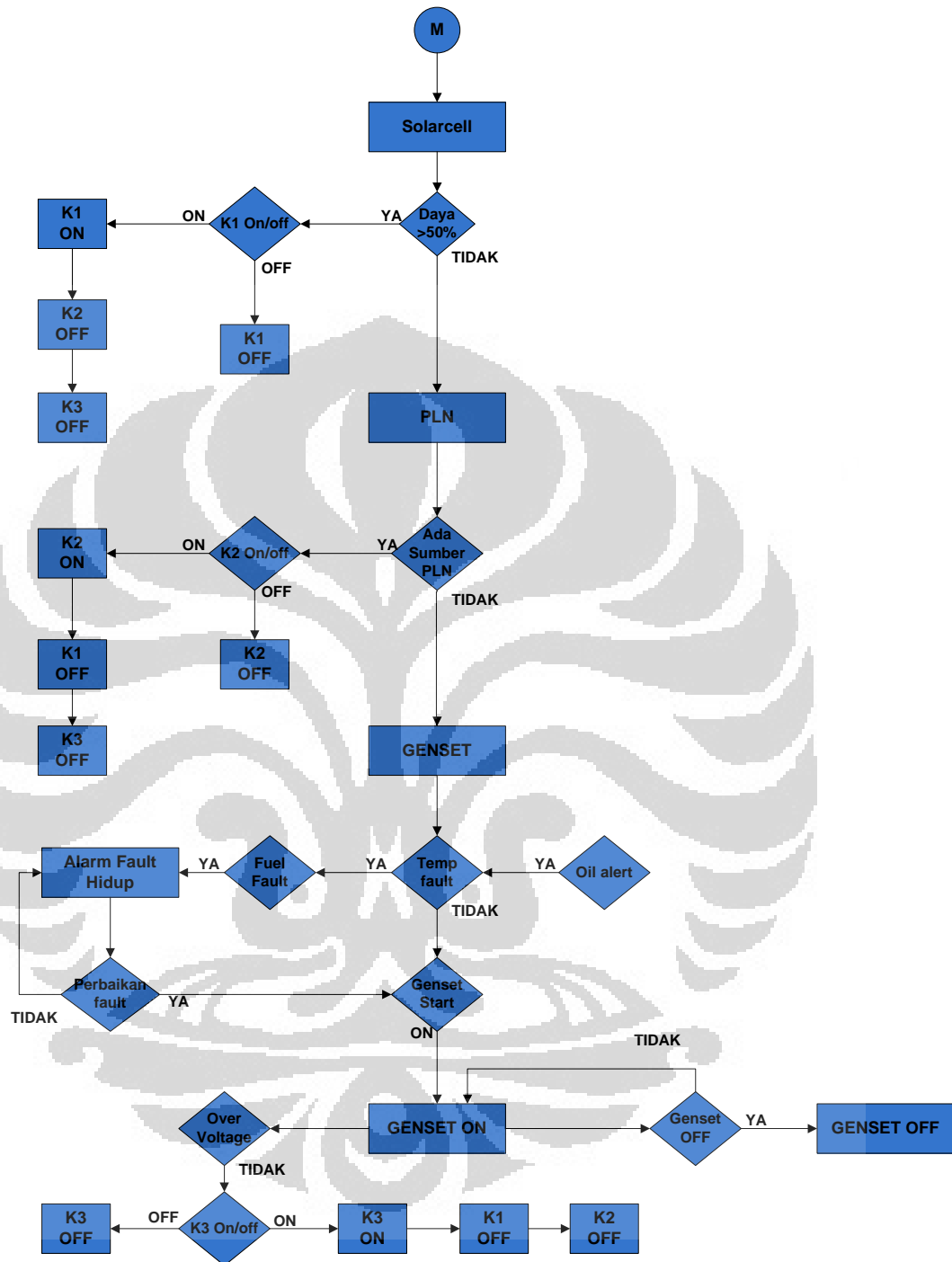
Berikut ini adalah tampilan flowchart proses kerja .



Gambar 3.4. Flowchart Cara Kerja Alat



Gambar 3.5. Flowchart Cara Kerja Mode Otomatis



Gambar 3.6. Flowchart Cara Kerja Mode Manual

Adapun cara kerja Alat adalah terbagi menjadi dua mode operasi, yaitu mode Otomatis dan Manual. Pada mode otomatis, Sistem akan memprioritaskan sumber listrik dari energi alternatif (*Solarcell*). Bila sumber dari *solarcell* berkurang, di indikasikan dengan berkurangnya daya dari *accu* untuk menyimpan daya listrik dari *solarcell* samapi 50 %, maka sistem akan melakukan *switching* dari sumber *solarcell* ke sumber dari PLN. K1 akan OFF dan K2 akan ON. Saat proses *switching* ini beban sementara di *dibackup* dari UPS, sehingga beban tetap tersupply arus listrik tanpa terjadi pemadaman sementara. Setelah proses *switching* beban akan di catu dayanya dari sumber PLN.

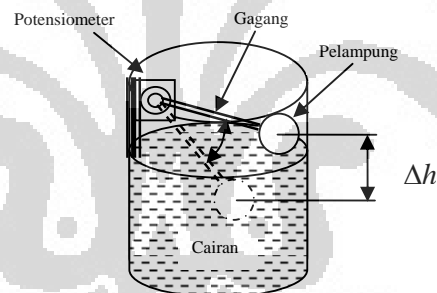
Bila sumber dari PLN *off* karena terjadi pemutusan dari pihak PLN, atau terjadi gangguan dalam saluran distribusi, maka K2 akan otomatis *off*, dan Sistem secara otomatis akan *men-start* GENSET. Bila terjadi *fault*, misalnya terjadi *Low Oil, Low Fuel, Over Heat*, GENSET tidak akan bisa hidup. Beberapa detik setelah Genset hidup, maka K2 akan ON secara otomatis. Bila saat GENSET dalam keadaan menyala terjadi *fault* misalnya terjadi *Low Oil, Low Fuel, Over Heat*, maka GENSET akan mati, dan bila terjadi *Over Voltage* GENSET tidak padam tapi K2 akan *Off*. Selama proses *switching* dari PLN ke GENSET (biasanya membutuhkan waktu +- 1 menit) beban sementara akan mendapat sumber atau di *backup* dari UPS, sehingga beban tetap mendapat pasokan daya. Jika sumber PLN ada dan sumber dari *solarcell* juga ada, maka sistem akan memprioritaskan mensupply beban dari sumber *solarcell*. K1 ON, atau bila *solarcell* tidak dapat mensupply energi listrik, maka sistem akan kembali mensupply beban dari sumber PLN, K2 ON, K3 akan langsung OFF.

Mode manual biasanya dioperasikan pada saat operator melakukan *maintenance* GENSET. Operasi manual dimulai dengan menetapkan *toggle swith* pada posisi *manual*. Selanjutnya dengan mengoperasikan saklar K1, K2 dan K3 pada posisi *on* atau *off*. K1 merupakan Kontaktor yang menghubungkan beban ke sumber energi alternatif (*solarcell*), K1 dapat dioperasikan bila daya dari *accu solarcell* lebih dari 50 %. K2 menghubungkan beban ke sumber PLN, K2 bisa dioperasikan apabila sumber listrik PLN tidak dalam keadaan pemadaman. K3 menghubungkan beban ke GENSET. Operasi K1, K2 dan K3 saling mengunci,

Dengan rangkaian ini, tegangan keluaran rangkaian ini jauh lebih stabil dibandingkan tegangan keluaran rangkaian dasar di atas. Dengan demikian akurasi pengukuran telah dapat ditingkatkan. Tegangan keluaran *op-amp* dapat langsung diumpankan ke rangkaian ADC untuk kemudian datanya diolah lebih lanjut oleh mikrokontroler.

3.5.2 Perancangan Sensor Lever Fuel

Sensor level bahan bakar yg dipakai adalah sensor jenis pelampung. Sensor ini bekerja berdasarkan perubahan resistansi. Perubahan resistansi ini di sebabkan karena berubahnya posisi pelampung saat terjadi perubahan level cairan. Gambar 3.9 merupakan contoh pemasangan sensor level, dan gambar 3.10 merupakan bentuk asli dari sensor level *fuel*. Nantinya dalam pengujian sensor level *fuel* akan dilakukan dengan simulasi mengubah kedudukan pelampung secara manual tanpa dilakukan langsung ke tangki.



Gambar 3.9. Sensor Level



Gambar 3.10. Sensor Level

3.5.3 Perancangan dan Realisasi Sensor Kondisi Aki Solarcell dan Over Voltage GENSET

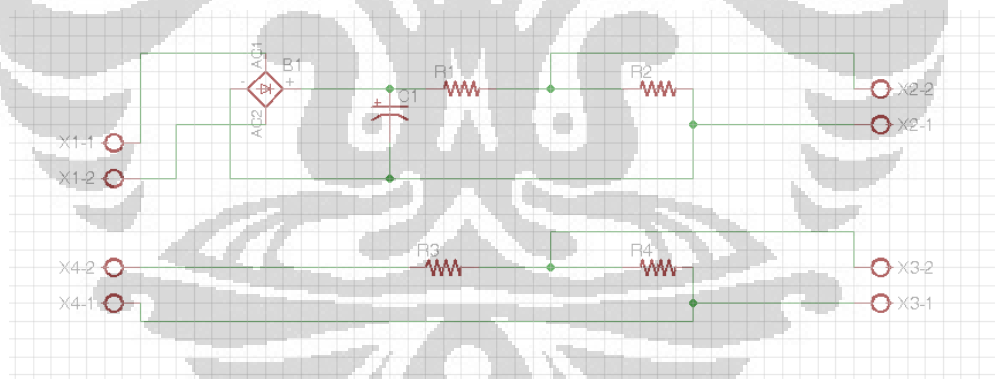
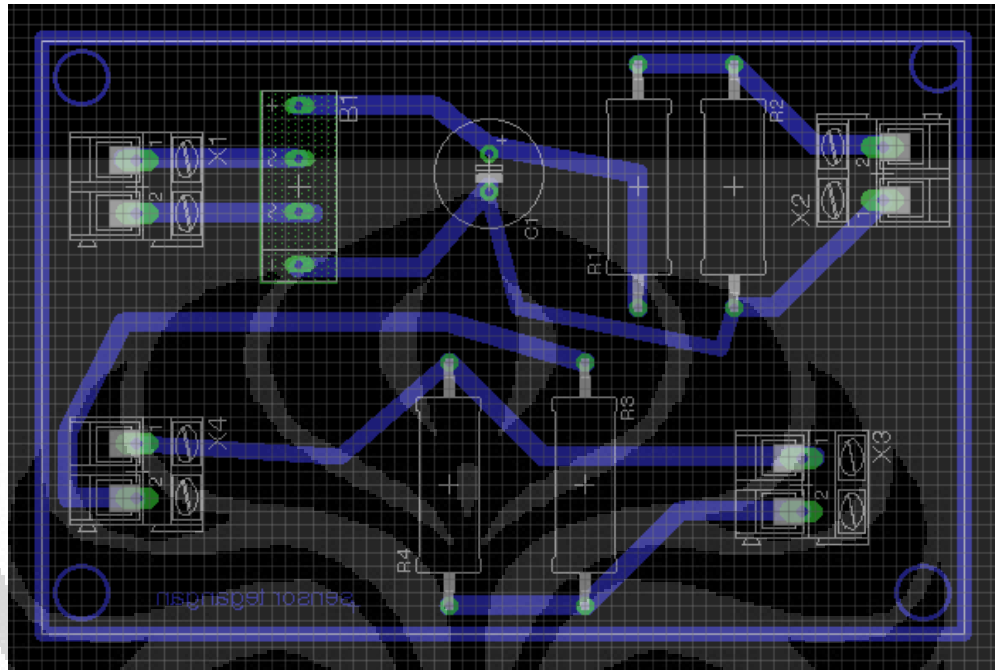
Sensor kondisi aki, merupakan sensor yang mendeteksi besarnya tegangan *output* dari aki. Karena kondisi aki melemah di indikasikan dengan turunnya tegangan keluaran dari aki. Biasanya tegangan keluaran saat aki harus di *charge* lagi adalah dibawah 10 Volt DC. Untuk mendekteksi ini maka pada aki harus dipasang sebuah sensor tegangan, yang nantinya dibaca oleh oleh mikrokontroler dan ditampilkan hasilnya di layar LCD. Cara kerja rangkaian sangat sederhana, pertama ADC0 digunakan sebagai input tegangan yang akan diukur. Karena tegangan yang sampai ke ADC0 atau V_s harus < 5 volt (bila > 5 volt akan merusak mikrokontroler), maka untuk mendapatkan *range* pengukuran sekitar V_{in} 12 Volt, maka diperlukan sebuah rangkaian pembagi tegangan yang dibentuk oleh R1 dan R2, dimana hubungan antara V_s (tegangan masuk ke ADC0) dan V_{in} (tegangan yang diukur) adalah $V_s = V_{in} * R_2 / (R_1 + R_2)$. Selain sebagai pembagi tegangan, rangkaian ini juga berguna untuk membatasi jumlah arus yang masuk ke ADC0. Saya mempergunakan R1=18K dan R2=4.7K yaitu nilai-nilai resistor yang umum dijumpai di pasaran, dengan V_{in} maksimum yang akan saya ukur adalah 12 volt DC. Maka kita mendapatkan data teknis sebagai berikut :

- Arus yang melewati ADC0 sekitar $= 12 / (18K + 4.7K) = 0.000529$ A.
- Tegangan V_s maksimum $= 12 * 4.7K / (18K + 4.7K) = 2.485$ Volt (masih aman dan jauh dari 5 volt).

Karena ADC yang digunakan di ATMEGA-16 berbasiskan kepada 10 bits, maka tegangan 0 volt akan direpresentasikan oleh 0 dan tegangan maksimum 5 volt akan direpresentasikan dengan $2^{10} - 1 = 1023$. Konsekuensinya, tegangan V_s maksimum di atas akan direpresentasikan dengan $= 1023 * 2.485 / 5 = 508$. Jadi dalam perhitungan yang dilakukan oleh mikrokontroler nantinya dibutuhkan sebuah faktor koreksi sebesar $f_c = 12 / 508$.

Untuk sensor over voltage tegangan AC juga sama, tegangan AC dari genset diturunkan dulu dari 220 Volt AC ke 5 Volt AC, menggunakan Transformator. Setelah itu tegangan AC 5 Volts di jadikan tegangan DC 5 volt dengan diode.

Baru kemudian masuk ke rangkaian pembagi tegangan. Adapun bentuk papan PCB rangkaian dari sensor ini terlihat pada gambar 3.11



Gambar 3.11. Rangkaian sensor over dan kondisi *accu*

3.5.4 Perancangan rangkaian driver

Rangkaian driver merupakan suatu rangkaian penggerak yang tujuan utamanya adalah mengaktifkan relay yang merupakan saklar dari interlock antara suplay *solarcell*, PLN dan Genset. Rangkaian driver ini terdiri dari 2 bagian yaitu *Optocoupler* dan rangkaian *switching transistor*.

3.5.4.1 Optocoupler

Rangkaian driver ini menggunakan sistem *interface* dimana asistem analog dan system digital terisolasi atau terpisah. Pada perancangan ini *system interface* yang digunakan adalah optoisolator atau yang lebih dikenal dengan *optocoupler* dengan seri 4N33 sehingga lebih aman apabila terjadi gangguan pada beban.

Pada perancangan alat ini, *optocoupler* mendapat input dari mikrokontroler melalui pin-pin dari PORTC. Apabila pin-pin dari mikrokontroler bernilai 1 maka *optocoupler* mendapat pulsa high dari pin-pin tersebut. Pada saat pulsa high (+5 V) diode led yang berada didalam *optocoupler* akan memancarkan cahaya , sehingga menyebabkan *receiver* dari optocoupler akan saturasi. Hal ini akan menyebabkan tegangan 5 volt akan mengalir ke kaki basis transistor. Gambar 3.12 merupakan salah satu contoh bentuk fisik dari optocoupler



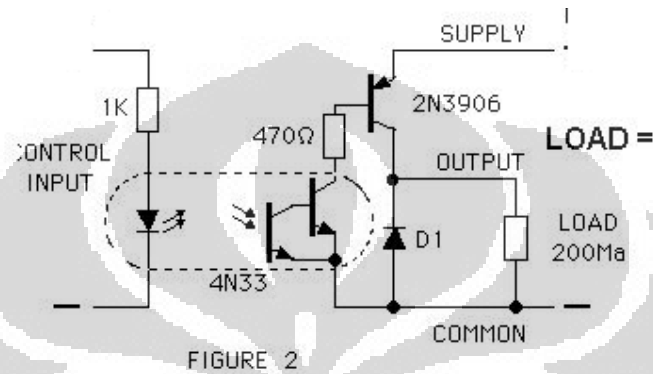
Gambar 3.12. *Optocoupler*

3.5.4.2 Rangkaian Switching Transistor

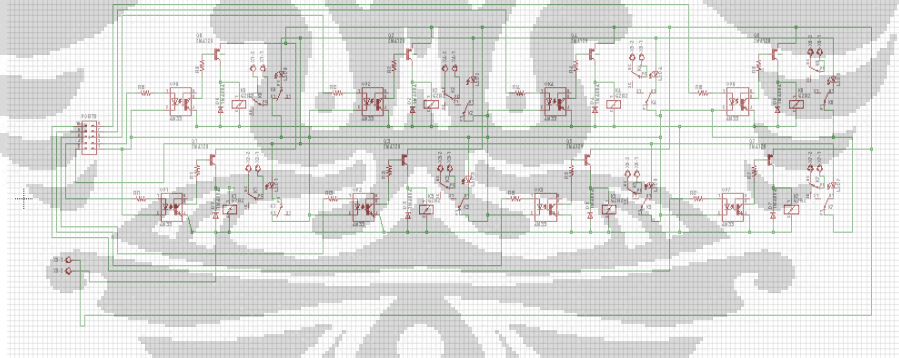
Untuk menggerakkan relay, *optocoupler* membutuhkan bantuan sebuah rangkaian yang disebut dengan *switching transistor* seperti terlihat pada gambar 3.13. Rangkaian *switching transistor* ini biasa disebut juga dengan rangkaian transistor sebagai saklar, dan transistor yang digunakan pada rangkaian ini adalah jenis PNP dengan tipe 2N3906. Rangkaian transistor sebagai saklar ini merupakan lanjutan dari rangkaian optocoupler yang tujuannya untuk menggerakkan relay.

Pada saat *optocoupler energize* maka tegangan 5 volt yg berasal dari rangkaian catu daya akan diteruskan ke kaki Basis transistor 2N3906, akibat dari

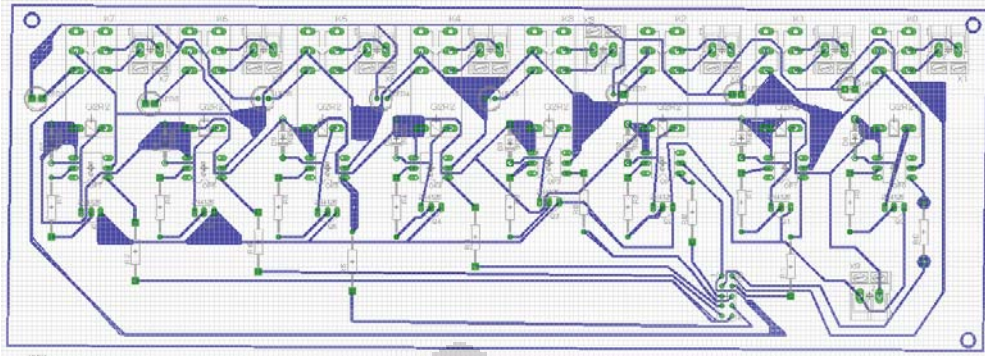
hal tersebut maka transistor berubah keadaanya dari *cut off* menjadi saturasi. Pada saat keadaan saturasi kaki kolektor dan kaki emitor akan terhubung , pada kondisi inilah transistor dikatakan berfungsi sebagai saklar. Pada saat saturasi maka salah satu koil dari relay akan terhubung dengan catudaya 12 V dan koil yang lainnya telah terhubung dengan ground sehingga relay akan bekerja. Gambar 3.15 merupakan bentuk PCB dari dari rangkaian *Driver Optocoupler* .



Gambar 3.13. Skema Rangkaian *Driver Optocoupler*



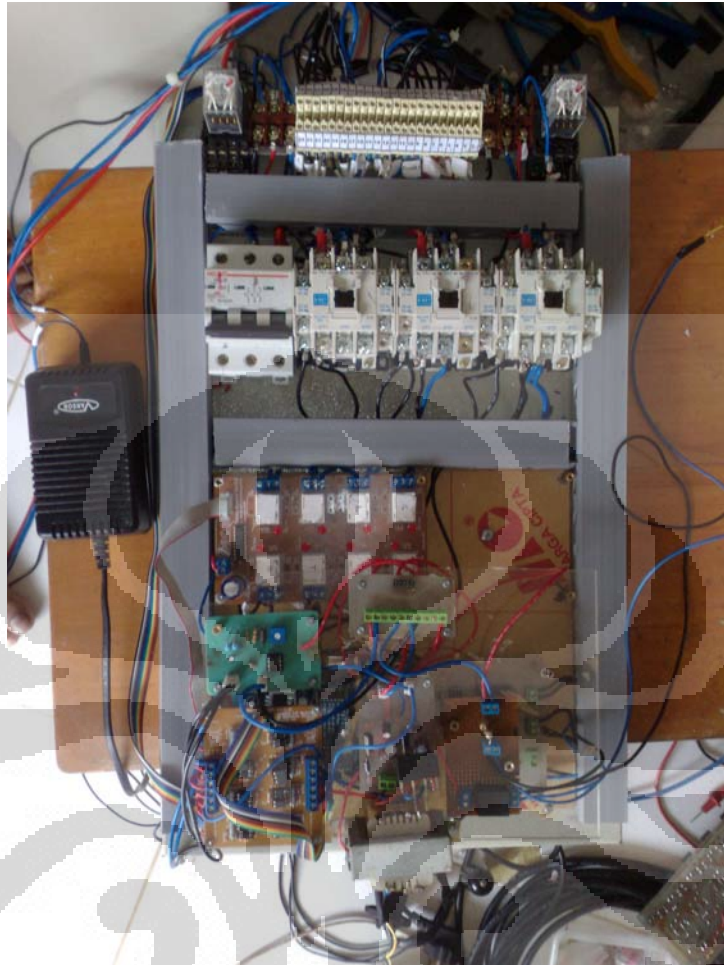
Gambar 3.14. Skema Rangkaian *Driver Optocoupler*



Gambar 3.15. PCB Driver Optocoupler

3.5.5 Perancangan dan Realisasi Modul Kontrol Alat

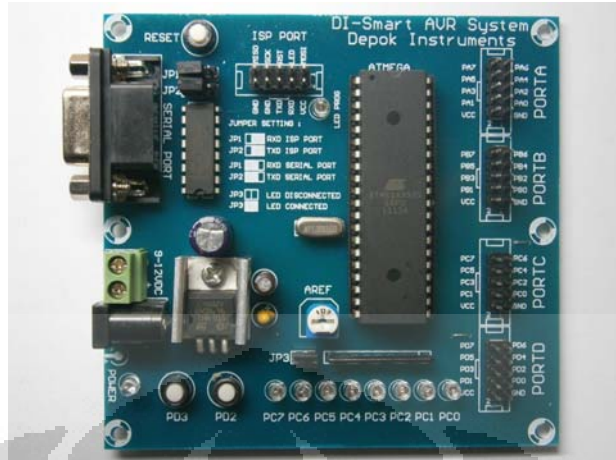
Gambar 3.16 merupakan modul kontrol Alat yang penulis buat, ini terdiri dari rangkaian kontrol utama yang berisi *chip* Atmega16, dan rangkaian pendukung lainnya rangkaian pendukung dari masing-masing sensor, rangkaian *Driver* yang berfungsi mengoperasikan output tegangan AC, rangkaian power supply, rangkaian kontaktor, serta *lineup terminal*. Rangkaian-rangkaian ini ditempatkan dalam suatu *Box panel* yang terbuat dari plat 3mm. Dalam *box* ini, didepannya terdapat panel kontrol yang berisikan *push on* START dan *push off* STOP engine, *toggle switch* MANUAL/AUTO operation, *push on-off* K1,K2,dan K3, *Key switch*, *Pilot Lamp* untuk lampu indikator indikator *fault*, dan LCD 16x2 karakter.



Gambar 3.16. modul kontrol

3.5.5.1 Sistem Minimum AVR ATmega 16

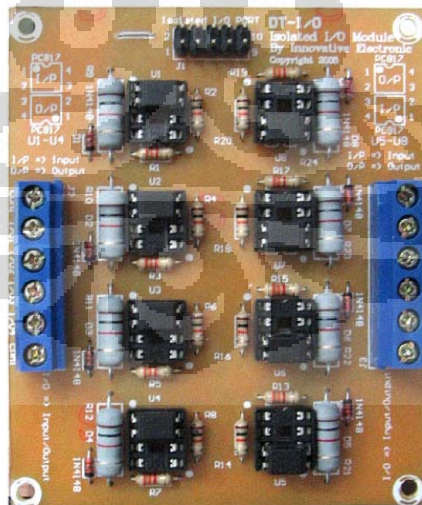
Dalam perancang sistem kontrol ini penulis menggunakan Sistem Minimum Mikrokontroler AVR dari Depok Instrument seperti yang terlihat pada gambar 3.17. Modul dengan sistem minimum mikrokontroler AVR ini digunakan sebagai pengatur kerja alat yang penulis rancang.



Gambar 3.17. Sistem *Minimum Mikrokontroler AVR Atmega 16*

3.5.5.2 Rangkaian Isolated Input

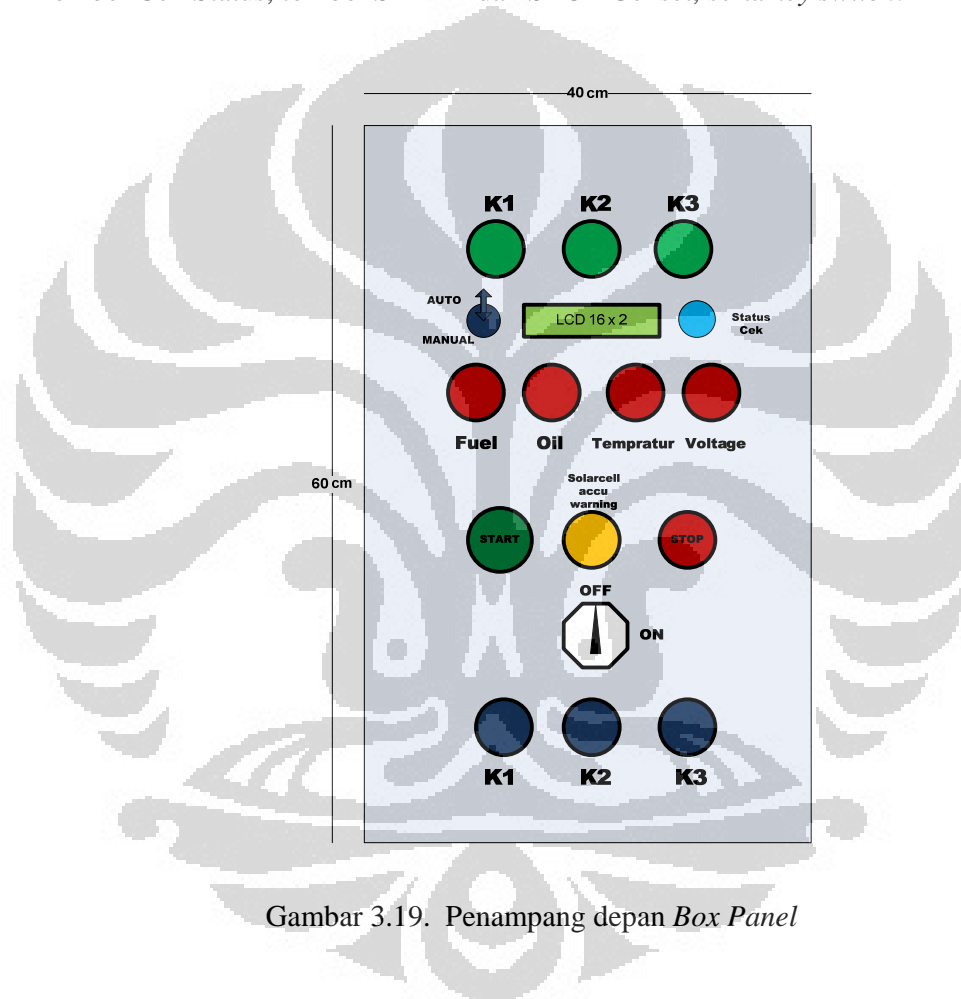
Dalam Perancangan rangkaian input untuk sistem ini, penulis memakai Modul Isolated Input dari Innovative Electronics seperti yang terlihat pada gambar 3.18. Modul Isolated Input ini merupakan modul input yang terisolasi secara optical terhadap tegangan input. Rangkaian Isolated ini memakai Optocoupler sebagai isolatednya.



Gambar 3.18. *Modul Isolated Input*

3.5.6 Perancangan Kontruksi Box Panel

Modul Kontrol Sistem ini diletakkan dalam sebuah Box Panel yang terbuat dari Plat besi. Ukuran Box Panel yang dipakai adalah 60 x 40 x 20 cm. pada gambar 3.19, terlihat disisi luar dari panel di susun berbagai *interface* indikator dan input, seperti lampu indikator K1,K2, K3, lampu indikator *fault* yang terjadi, LCD 16 x 2 , *toggle switch* pemilihan Mode Manual / Automatis, Tombol Cek Status, tombol START dan STOP Genset, serta *key switch*.



Gambar 3.19. Penampang depan *Box Panel*



Gambar 3.20. *Box Panel Sistem Kontrol*

3.5.7 Tabel Input dan Output Mikrokontroler

Tabel ini merupakan penentuan perangkat input/output, juga pengalamatan serta fungsi masing-masing peralatan I/O. Berikut adalah tabel input dan output yang akan digunakan dalam pembuatan kontrol :

Tabel 3.1. Tabel Input

Tabel Input	
Nama Input	Port
Accu Solarcell detctor	Port A.0
Fuel Sensor	Port A.1
tempratur sensor	Port A.2
voltage sensor	Port A.3
charger sensor	Port A.4
PLN sensor	Port B.0
Status Cek	Port B.1
Auto/Manual	Port B.2
Start Genset	Port B.3
Stop Genset	Port B.4
K1	Port B.5
K2	Port B.6
K3	Port B.7

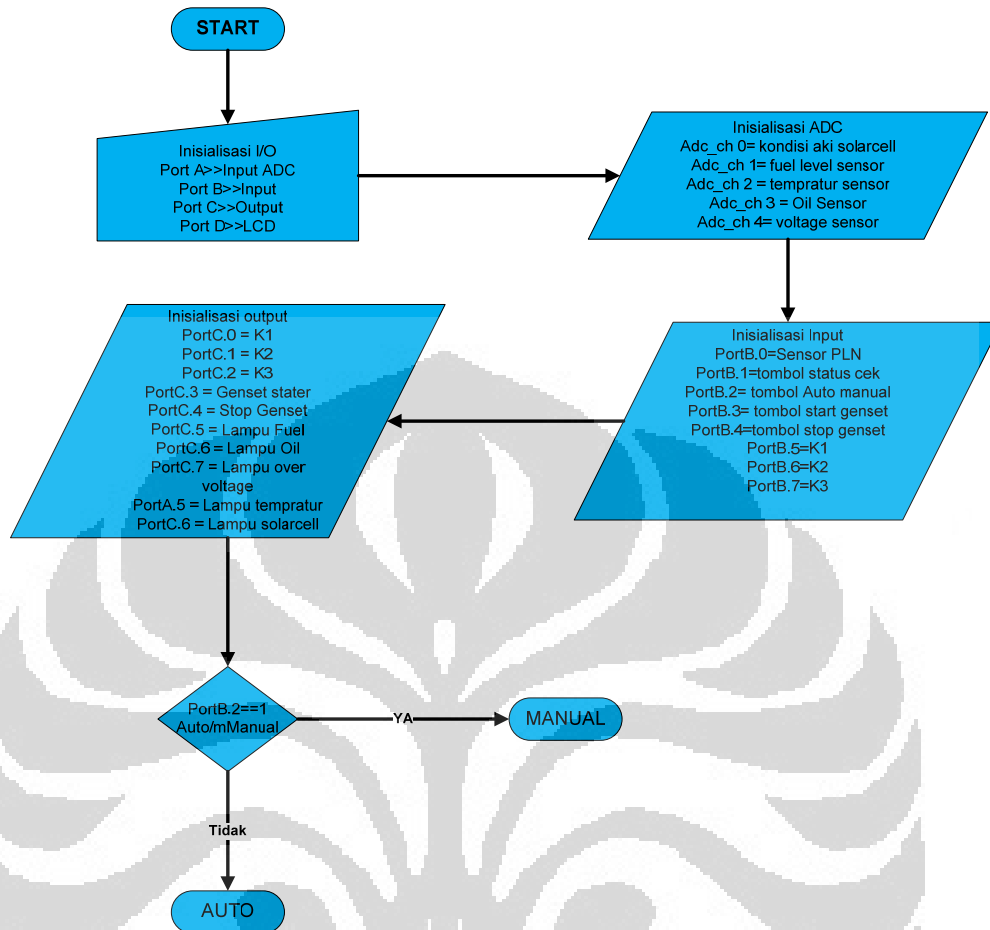
Tabel 3.1 .Tabel Output

Tabel Output	
Nama Output	Port
K1	Port C.0
K2	Port C.1
K3	Port C.2
Genset relay On	Port C.3
Genset relay Off	Port C.4
Lampu Fuel	Port C.5
Lampu Oil/charger	Port C.6
lampu over volage	Port C.7
Lampu over heat	Port A .5
Lampu Solarrcell warning	Port A .6
charger switch	Port A .7

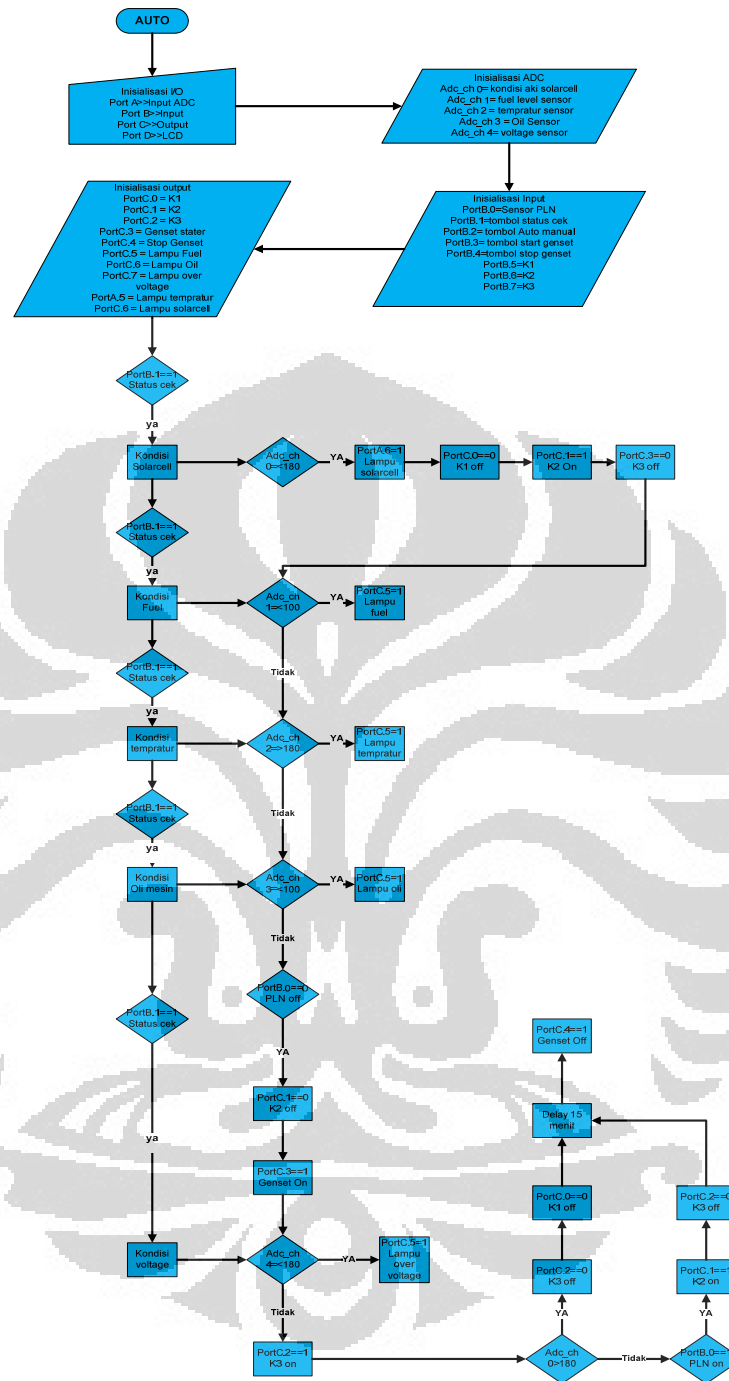
3.6 Perancangan Software

3.6.1 Flowchart Perancangan Software

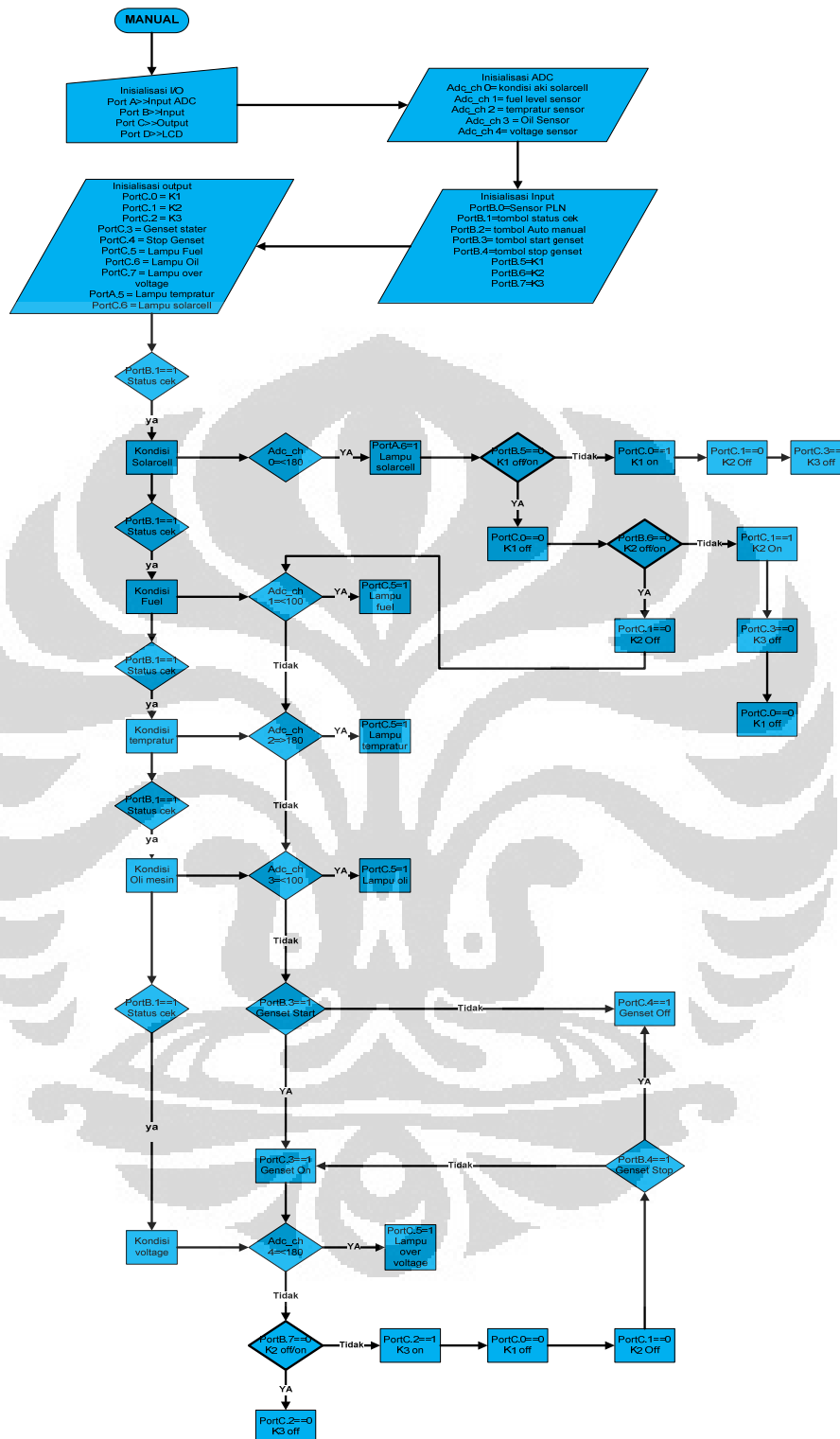
Dalam perancangan software, penulis menggunakan CodevisionAVR yang berbasis bahasa C untuk pemrograman mikrokontrolernya. List program pengontrolan alat ini terlampir di lampiran. Dibawah ini merupakan *flowchart* dari rancangan software untuk sistem pengaturan Pasokan listrik pada pembangkit hibrida.



Gambar 3.21. *Flowchart* rancangan software



Gambar 3.22. Flowchart rancangan software mode otomatis



Gambar 3.23. Flowchart rancangan software mode manual

BAB IV

PENGUJIAN DAN ANALISA

4.1 Tujuan Pengujian

Pengujian ini dilakukan untuk memperoleh data yang digunakan untuk mengukur sejauh mana kinerja sistem yang telah kita rancang. Dengan melakukan pengujian ini kita dapat mengetahui kondisi dari alat yang kita rancang .

4.2 Pengujian Kinerja Sistem

Pengujian kinerja sistem ini dilakukan dengan mencoba semua Mode, baik *Mode Automatic*, maupun *Mode Manual* serta pengujian masing masing sensor dan *input output*.

4.2.1 Pengujian Input

Pengujian input ini bertujuan untuk mengetahui kondisi dari semua input yang terpasang pada sistem. Berikut adalah table pengujian Input,

Tabel 4.1. Pengujian Input

Nama Input	Pengujian	Parameter		Kondisi	
		Tegangan keluaran isolated input saat tombol ditekan	Voltage	OK	NG
Tombol Push ON/OFF K1	Tombol ditekan		4,92	√	
Tombol Push ON/OFF K2	Tombol ditekan		4,90	√	
Tombol Push ON/OFF K3	Tombol ditekan		4,92	√	
Tombol Push ON Start GENSE	Tombol ditekan		4,87	√	
Tombol Push ON/OFF Stop Gen	Tombol ditekan		4,86	√	
Toggle Switch Auto Manual	Toggle di pidahkan		4,90	√	
Tombol Push ON Status Cek	Tombol ditekan		4,87	√	

4.2.1 Pengujian Output

Pengujian output ini bertujuan untuk mengecek kondisi dari masing - masing output yang di pakai pada sistem. Pengujian dilakukan pada saat system berada pada mode operasi manual.

Tabel 4.2. Pengujian Output

Nama Output	Alamat I/O	Pengujian	Parameter			Kondisi	
				V (Volt)	OK	NG	
Kontaktor K1	Port C.0	Push ON/OFF K1 di tekan	Menyala	219	√		
Kontaktor K2	Port C.1	Push ON/OFF K2 di tekan	Menyala	219	√		
Kontaktor K3	Port C.2	Push ON/OFF K3 di tekan	Menyala	219	√		
Lampu indikator Fuel fault	Port C.5	posisikan dlm keadaan fault	Menyala	12	√		
Lampu indikator Oil Alert	Port C.6	posisikan dlm keadaan fault	Menyala	12	√		
Lampu Indikator Over Voltage	Port C.7	Kondisi Voltage di posisikan dlm keadaan fault	Menyala	12	√		
Lampu indikator over heat	Port A .5	Kondisi Suhu di posisikan dlm keadaan fault	Menyala	12	√		
Lampu indikator solarcell warning	Port A .6	Kondisi aki solarcell di posisikan dlm keadaan fault	Menyala	12	√		
Relay Starting genset	Port C.3	Push button Start Genset di tekan	Menyala	5	√		
Relay Stop genset	Port C.4	Push button Stop genset di tekan	Menyala	5	√		

4.2.3 Pengujian Mode Manual

Pengujian ini dilakukan bertujuan untuk mengetahui kerja dari sistem yang kita rancang pada saat sistem di operasikan pada mode manual. Pengujian ini diawali dengan memposisikan *toggle switch* pada posisi ‘manual’, kemudian mengikuti pengujian sesuai tabel 4.3. pengujian dilakukan beberapa kali dan dilakukan pada saat pada saat berbeban. Beban yang dipakai dalah beban lampu dan kipas angin.

Tabel 4.3. Tabel Hasil Pengujian mode Manual

Tabel Pengujian Mode Manual																						
Input											Output											
K1	K2	K3	Start Genset	Sensor PLN	Stop genset	sensor fuel	Sensor Suhu	Sensor tegangan	sensor oli	sensor accu	Kontaktor 1	Kontaktor 2	Kontaktor 3	Lampu Indikator solarcell	Lampu Indikator PLN	Lampu Indikator Genset	GENSET	Lampu Indikator Fuel	Lampu Indikator heat	Lampu Indikator Oli	Lampu Indikator Over voltage	Lampu Indikator solarcell warning
1	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
1	1	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0
0	1	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1
0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	1	1	0	1	0	0	0	0	0	0	0	0	1	1	1	1	0
0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	1	0

Keterangan:
■ : Input /tombol ditekan (diberi logika 1)
■ : sensor di posisikan dalam keadaan fault
■ : output dalam kondisi hidup (berlogika 1)

Hasil pengujian dari tabel 4.3, memperlihatkan apabila kita memberi nilai logika 1 (menekan tombol) pada salah satu input (disini diambil contoh input *push on/off* K1), maka kalau dilihat dari deskripsi dan *flowchart* kerja, output yang harus bekerja adalah Kontaktor 1 dan Lampu indikator *solarcell*. Apabila memberikan logika 1 pada input sensor, berarti dalam pengujiaannya sensor dikondisikan dalam keadaan fault, maka lampu indikator masing-masing sensor akan berlogika 1 (menyala).

4.2.4 Pengujian Mode Otomatis

Pengujian ini dilakukan bertujuan untuk mengetahui kerja dari sistem yang kita rancang pada saat sistem dioperasikan pada mode otomatis. Pengujian ini diawali dengan memposisikan *toggle switch* pada posisi ‘otomatis’, kemudian mengikuti pengujian sesuai tabel 4.4. Pengujian dilakukan pada saat saat berbeban. Adapun beban yang dipakai adalah beban lampu dan kipas angin.

Tabel 4.4. Tabel hasil pengujian mode otomatis

Tabel pengujian pada mode Automatic

Input						Output											
Sensor PLN	sensor fuel	Sensor Suhu	Sensor tegangan	sensor oli	sensor accu	Kontaktor 1	Kontaktor 2	Kontaktor 3	Lampu indikator solarcell	Lampu Indikator PLN	Lampu Indikator Genset	GENSET	Lampu Indikator Fuel	Lampu Indikator heat	Lampu Indikator Oli	Lampu Indikator Over voltage	Lampu Indikator solarcell warning
0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	1	0	0	1	0	0	0	0	0	0	1
1	0	0	0	0	1	0	0	1	0	0	1	1	0	0	0	0	1
1	1	1	0	1	1	0	0	1	0	0	0	0	1	1	1	0	1
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	0	0	0	0	0	0	0	1	1	1	1	1
0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0
1	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0

Keterangan :

1 : Sensor dikondisikan dalam keadaan fault

1 : output dalam kondisi bekerja /hidup

Pada pengujian tabel 4.4, apabila *input* sensor di berikan logika 1 (sensor diposisikan pada keadaan *fault*), misalkan sensor *accu solarcell*, mengindikasikan terjadi *fault*, maka *output* yang bekerja adalah kontaktor 2, lampu indikator *solarcell* dan lampu indikator *solarcell warning*.

4.3 Tujuan Analisa

Analisa ini dilakukan untuk membandingkan hasil dari pengujian yang telah kita lakukan dengan deskripsi atau cara kerja yang telah kita rancang. Serta menganalisa kemungkinan kegagalan atau *error* yang terjadi pada alat yang kita rancang.

4.4. Analisa

4.4.1. Analisa Pada Saat Mode Manual

Dari beberapa kali pengujian yang telah dilakukan pada mode manual dalam keadaan berbeban, kerja alat yang telah dibuat sudah sesuai dengan deskripsi alat yang dirancang. Tidak terjadi kegagalan atau *error* yang menyebabkan alat tidak berfungsi sebagaimana mestinya, hanya lampu indicator PLN yang tidak menyala, ini karena lampu sudah tidak bisa digunakan dan harus dilakukan penggantian. Ini terlihat dari hasil pengujian tabel 4.3, semua pengujian input menghasilkan *output* yang bekerja sesuai deskripsi dan *flowchart* kerja yang telah dirancang.

4.4.2 Analisa Pada Saat Mode Otomatis

Dari hasil pengujian yang telah dilakukan pada mode otomatis, secara garis besar alat sudah berfungsi sesuai dengan deskripsi alat yang dirancang., di indikasikan dari tabel hasil pengujian pada tabel 4.4. Namun masih ada beberapa kondisi yang memungkinkan alat tidak bisa merespon, diantaranya adalah ;

1. Kondisi saat *accu solarcell* baru terisi 60 %, alat akan langsung menghubungkan supplay ke *solarcell*. Ini akan mengakibatkan *accu solarcell* tidak akan terisi penuh, karena setiap kondisi daya *accu* sudah diatas 50 %, alat akan langsung mengalihkan supplay listrik ke *solarcell*.
2. Kondisi dimana mesin GENSET mengalami kerusakan tiba-tiba, misalkan kerusakan pada sistem pengapian mesin, sehingga menyebabkan mesin GENSET tidak bisa hidup. Bila ini terjadi maka saat alat melakukan *switching* ke pembangkit GENSET, akan menghidupkan motor stater mesin secara terus menerus tanpa berhenti, sebelum dimatikan secara manual oleh operator.
3. Apabila terjadi kerusakan pada UPS, alat tidak bisa memberikan indikator yang mengindikasikan UPS dalam keadaan *fault*.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dari hasil perancangan dan pengujian yang telah dilaksanakan, maka dapat disimpulkan :

1. Rancang bangun sistem pengaturan pasokan listrik pada pembangkit hibrida yang telah direalisasikan sudah bekerja sesuai dengan deskripsi dan *flowchart* kerja yang penulis rancang.
2. Ada beberapa kondisi yang tidak dapat di deteksi dari alat ini diantaranya :
Kondisi saat *accu* solarcell baru terisi 60 %, alat akan langsung menghubungkan supplay ke *solarcell*. Ini akan mengakibatkan *accu solarcell* tidak akan terisi penuh, karena setiap kondisi daya battere sudah diatas 50 %, alat akan langsung mengalihkan supplay listrik ke *solarcell* kondisi saat mesin GENSET tiba-tiba mengalami kerusakan, seperti kerusakan sistem pengapian mesin, ini akan menyebabkan alat akan menstarter mesin secara terus menerus. Kondisi dimana apabila UPS mengalami kerusakan (*fault*), alat tidak bisa memberikan indikator bila UPS mengalami kerusakan.

5.2 Saran

Untuk pengembangan lebih lanjut penulis menyarankan :

1. Melakukan proses pemanasan dan pengecekan mesin genset minimal 1 kali dalam 1 minggu, untuk menghindari GENSET tidak bisa hidup.
2. Penambahan sensor atau indikator untuk mengetahui kerusakan pada UPS, misalnya sensor kondisi *accu* dari UPS, karena biasanya yang sering bermasalah pada UPS adalah *accunya*.

DAFTAR REFERENSI

- [1] Budiharto, Widodo. 2008. *Panduan Praktikum Mikrokontroler AVR ATMEGA 16*
Jakarta: PT Elex Media Komputindo
- [2] Heryanto, M. Ary, Wisnu Adi P. *Pemrograman Bahasa C untuk Mikrokontroler ATMEGA 8535*, Yogyakarta : ANDI
- [3] Andrianto, Heri. *Pemrograman Mikrokontroler AVR ATMEGA 16 Menggunakan Bahasa C*, Bandung : INFORMATIKA
- [4] <http://www.alldatasheet.com/datasheet.pdf/pdf/77784/AUK/SPC817M.html>
- [5] <http://www.atmel.com/products/AVR>
- [6] <http://www.avrfreaks/viewproject>
- [7] <http://www.ats-amf.com>
- [8] <http://yd1chs.wordpress.com/2010/08/15/simple-avr-lcd-voltmeter/>
- [9] <http://elektro-kontrol.blogspot.com/p/avr-projects.html>
- [10] Moran, Michael J dan Howard N. Shapiro. 2008. Fundamentals of Engineering Thermodynamics. USA: Jhon Wiley & Sons

```

/*****
This program was produced by the
CodeWizardAVR V1.25.3 Standard
Automatic Program Generator
© Copyright 1998-2007 Pavel Hai Duc, HP InfoTech s.r.l.
http://www.hpinfotech.com

Project :
Version :
Date : 12/22/2011
Author : F4CG
Company : F4CG
Comments:

Chip type : ATmega16
Program type : Application
Clock frequency : 11.004000 MHz
Memory model : Small
External SRAM size : 0
Data Stack size : 256
*****/

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

// Alphanumeric LCD Module functions
#asm
.equ __lcd_port=0x12 ;PORTD
#endasm
#include <lcd.h>

#define ADC_VREF_TYPE 0x40
void oil(void);
void minyak(void);
void solar(void);
void tempr(void);
void auto(void);
void manual(void);
void genset_hidup(void);
void genset_mati();
void solar_cell(void);
void bahan_bakar(void);
void suhu_mesi n(void);
void over_voltage(void);
void sensor_oli(void);
void kondisi(void);
int sol arcell, oli, suhu, vol tage, fuel, ac, dataK1, dataK2, dataK3, dataG;
int tegangan;
char temp[8];
float tegangan_terukur, suhu_terukur, vol tage_ac, minyak_terukur,
hasil, persen;
// Read the AD conversion result
unsigned int read_adc(unsigned char adc_input)
{
ADMUX=adc_input | (ADC_VREF_TYPE & 0xff);
// Start the AD conversion
ADCSRA|=0x40;
// Wait for the AD conversion to complete
while ((ADCSRA & 0x10)==0);

```

```

ADCSRA|=0x10;
return ADCW;
}

// Declare your global variables here

void main(void)
{
int nilai, lampau; // Declare your local variables here

// Input/Output Ports initialization
// Port A initialization
// Func7=In Func6=Out Func5=Out Func4=In Func3=In Func2=In Func1=In
Func0=In
// State7=T State6=0 State5=0 State4=T State3=T State2=T State1=T
State0=T
PORTA=0x00;
DDRA=0x60;

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

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

// Port D initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In
Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTD=0x00;
DDRD=0x00;

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

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer 1 Stopped
// Mode: Normal top=FFFFh
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer 1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;

```

```

OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer 2 Stopped
// Mode: Normal top=FFh
// OC2 output: Disconnected
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
MCUCR=0x00;
MCUCSR=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x00;

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

// ADC initialization
// ADC Clock frequency: 687.750 kHz
// ADC Voltage Reference: AVCC pin
// ADC Auto Trigger Source: None
ADMUX=ADC_VREF_TYPE & 0xff;
ADCSRA=0x84;

// LCD module initialization
lcd_init(16);
nilai=1;
lampau=1;
#asm ("sei")
while (1)
{
char buf[33];
sol ar();
temp r();
mi nyak();

if (PINB.2==1){
auto();
}
else{

manual ();
}

if(PINB.1==0){
lampau=0;
}
if (PINB.1==1){
if (lampau==0){
nilai=nilai*2;
if (nilai>32){
nilai=1;
lcd_clear();
}
lampau=1;
}
}
}

```



```

    }
    lcd_gotoxy(14, 0);
    printf(buf, "%i", nilai);
    lcd_puts(buf);
    if (nilai == 1){

        solar_cell();
    }

    if (nilai == 2){
        delay_ms(100);
        lcd_clear();
        bahan_bakar();
    }
    if (nilai == 4){
        delay_ms(100);
        lcd_clear();
        suhu_mesi n();
    }
    if (nilai == 8){
        delay_ms(100);
        lcd_clear();
        over_vol tage();
    }
    if (nilai == 16){
        delay_ms(100);
        lcd_clear();
        sensor_oli ();
    }
    if (nilai == 32){
        delay_ms(100);
        lcd_clear();
        kondisi ();
    }
};

void kondisi (void)
{
    if(PINB. 2==1){
        lcd_gotoxy(0, 1);
        lcd_putsf("AUTO OPERATION");
    }
    else {
        lcd_gotoxy(0, 1);
        lcd_putsf("MANUAL OPERATION");
    }
}

void solar_cell (void)
{
    #asm("sei")
    tegangan=read_adc(0);
    tegangan_terukur=(float) tegangan*12/508;
    ftoa(tegangan_terukur, 1, temp);
    lcd_gotoxy(0, 0);
    lcd_puts(temp);
    lcd_gotoxy(4, 0);
    lcd_putsf("V");
    if(tegangan_terukur>=10){
        persen=tegangan_terukur-10;
        hasil =(float)persen*50;
        ftoa(hasil, 1, temp);
        lcd_gotoxy(7, 0);
        lcd_puts(temp);
        lcd_gotoxy(12, 0);
        lcd_putsf("%");
    }
    else {
        persen=0;
        delay_ms(100);
    }
}

```

```

    lcd_clear();
    lcd_gotoxy(7, 0);
    lcd_putsf("0%");
}
if(tegangan_terukur >= 10){
    lcd_gotoxy(0, 1);
    lcd_putsf(" battre ok");}

else {
    lcd_gotoxy(0, 1);
    lcd_putsf("battre lemah");
}
}
}

void solar(void)
{
    tegangan=read_adc(0);
    tegangan_terukur=(float)tegangan*12/508;
    if(tegangan_terukur>=11){
        PORTA. 6=0;
    }
    else{
        PORTA. 6=1;
    }
}

void bahan_bakar (void)
{
    #asm("sei ")
    fuel =read_adc(1);
    mi nyak_terukur=(float) fuel *18/409;
    ftoa(mi nyak_terukur, 1, temp);
    lcd_gotoxy(0, 0);
    lcd_puts(temp);
    lcd_gotoxy(5, 0);
    lcd_putsf("Li ter");
    lcd_gotoxy(1, 1);
    lcd_putsf("bahan bakar");
}

void mi nyak(void)
{
    fuel =read_adc(1);
    mi nyak_terukur=(float) fuel *18/409;
    if(mi nyak_terukur<2){
        PORTC. 5=1;
    }
    else{
        PORTC. 5=0;
    }
}

void suhu_mesi n (void)
{
    suhu=read_adc(2);
    suhu_terukur=(float)suhu*500/1023;
    ftoa(suhu_terukur, 1, temp);
    lcd_gotoxy(0, 0);
    lcd_puts(temp);
    lcd_gotoxy(5, 0);
    lcd_putchar(0xDf);
    lcd_gotoxy(1, 1);
    lcd_putsf(" suhu mesi n");
}

void tempr(void)
{
    suhu=read_adc(2);
    suhu_terukur=(float)suhu*500/1023;
    ftoa(suhu_terukur, 1, temp);
    if (suhu_terukur>40){
        PORTA. 5=1;
    }
}

```

```

    }
    else{
        PORTA. 5=0;
    }
}

void over_vol tage (void)
{
    ac=read_adc(3);
    vol tage_ac=(float)ac*220/430;
    ftoa(vol tage_ac, 2, temp);
    lcd_gotoxy(0, 0);
    delay_ms(10);
    lcd_puts(temp);
    lcd_gotoxy(1, 1);
    lcd_putsf("over vol tage");
}

void sensor_oli (void)
{
    if (PINA. 4==0){
        PORTA. 7=0;
        lcd_gotoxy(0, 0);
        lcd_putsf(" Oil Alert");
    }
    else {
        PORTA. 7=1;
        lcd_gotoxy(0, 0);
        lcd_putsf(" Oil OK");}
    lcd_gotoxy(1, 1);
    lcd_putsf("sensor oli");
}

void oil (void)
{
    if (PINA. 4==0){
        PORTA. 7=0;
    }
    else {
        PORTA. 7=1;
    }
}

void auto (void)
{
    solarcell=read_adc(0);
    fuel=read_adc(1);
    suhu=read_adc(2);
    vol tage=read_adc(3);
    if (tegangan_terukur<10){
        PORTC. 0=0;
        PORTC. 1=1;
        PORTC. 2=0;
        genset_mati ();

        if (PINB. 0==1){
            PORTC. 2=1;
            PORTC. 1=0;
            PORTC. 0=0;
            genset_hidup();
            if ((tegangan_terukur>10)&&(PINB. 0==0)){
                PORTC. 2=0;
                PORTC. 1=0;
                PORTC. 0=1;
                delay_ms(10);
                genset_mati ();
            }
        }
    }
    else {
        PORTC. 2=0;
        PORTC. 0=1;
        PORTC. 1=0;
    }
}

```

```

        delay_ms(10);
        genset_mati ();
    }
}

void genset_hidup(void){
    fuel=read_adc(1);
    suhu=read_adc(2);
    voltase=read_adc(3);
    oli=read_adc(4);
    if ((suhu_terukur<40)&&(mi nyak_terukur>2)){
        PORTC. 4=0;
        PORTC. 3=1;
    }
    else {
        genset_mati ();
    }
}

void genset_mati (void){ // prosedur mematikan
    genset kondisi auto
    PORTC. 3=0;
    PORTC. 4=1;
}

void manual (void){ // prosedur manual
    operasi
    sol arcell =read_adc(0);
    fuel =read_adc(1);
    suhu=read_adc(2);
    voltase=read_adc(3);
    oli=read_adc(4);
    if
    ((PINB. 3==1)|| (dataG==0xff)|| (suhu_terukur>40)|| (mi nyak_terukur<2)){
        // tombol start genset ditekan
        PORTC. 3=0;
        PORTC. 4=1;
        // relay starting genset hidup bila kondisi oli, fuel, dan suhu
        dalam keadaan normal dan relay stop genset mati
    }
    else{
        PORTC. 3=1;
        PORTC. 4=0;
    };
    if (PINB. 4==1){ // genset di mati kan bila kondisi
    fuel, oli dan suhu tidak normal dan tombol stop ditekan
        dataG=0xff;
        PORTC. 4=1;
        // relay genset off di hidupkan
    }
    else {
        dataG=0x00;
        PORTC. 4=0;
    };
    if
    ((PINB. 5==1)|| (dataK2==0xff)|| (dataK3==0xff)|| (tegangan_terukur<10)){
        // menghidupkan K1 dengan syarat kondisi batree sol arcell OK, dan K2
        serta K3dalam keadann OFF
        dataK1=0x00;
        // data dari K1 untuk data interlock
        PORTC. 0=0;
        // kontaktor K1 hidup
    }
    else {
        PORTC. 0=1;
        // kontaktor k1 mati
        dataK1=0xff;
        // data k1 bernilai bila k1 off
    }
}

```

```
};  
if((PINB.6==1)|| (dataK1==0xff)||dataK3==0xff){ //  
menghidupkan K2 dengan syarat K1 dan K2 mati serta PLn on  
dataK2=0x00;  
// data dari K2 bernilai 0  
PORTC.1=0;  
// kontaktor k2 hidup  
}  
else{  
dataK2=0xff;  
// data K2 bernilai 1  
PORTC.1=1;  
// kontaktor k2 off  
};  
if ((PINB.7==1)|| (dataK1==0xff)|| (dataK2==0xff)) //  
{  
menghidupkan k3 dengan syarat K1 dan K2 mati, serta tidak terjadi over  
voltage  
dataK3=0;  
// data K3 bernilai 0  
PORTC.2=0;  
// kontaktor 3 hidup  
}  
else {  
dataK3=0xff;  
// data K3 bernilai 1  
PORTC.2=1;  
// kontaktor K3 mati  
};  
}
```