RANCANG BANGUN INTERFACE UNTUK VISUALISASI PROSES SINKRONISASI GENERATOR SINKRON BERBASIS MIKROKONTROLER AT MEGA2560

Tugas Akhir
Untuk Memenuhi Sebagian Persyaratan
Derajat Sarjana S – 1 Jurusan Teknik Elektro

Oleh :

ANDRYO ISKANDAR

F1B 012 012

JURUSAN TEKNIK ELEKTRO

FAKULTAS TEKNIK

UNIVERSITAS MATARAM

2016

TUGAS AKHIR

RANCANG BANGUN INTERFACE UNTUK VISUALISASI PROSES SINKRONISASI GENERATOR SINKRON BERBASIS MIKROKONTROLER AT MEGA2560

Telah diperiksa dan disetujui oleh Tim Pembimbing :

1 Pembimbing Utama,

Tanggal: 22/10/2016

I Made Ari Nrartha, ST.,MT.
NIP. 19730404 199903 1 003

2 Pembimbing Pendamping,

Tanggal: 20/10/2016

I Made Budi Suksmadana, ST., MT.
NIP. 19710426 199903 1 002

Mengetahui,
Ketua Jurusan Teknik Elektro
Fakultas Teknik
Universitas Mataram

Sudi M. Al Sasongko, ST., MT.
NIP. 19670526199703 1 001

TUGAS AKHIR

RANCANG BANGUN INTERFACE UNTUK VISUALISASI PROSES SINKRONISASI GENERATOR SINKRON BERBASIS MIKROKONTROLER AT MEGA2560

Oleh:

Andryo Iskandar
F1B 012 012

Telah dipertahankan di depan Tim Penguji
Pada tanggal 8 Oktober 2016
dan dinyatakan telah memenuhi syarat mencapai derajat Sarjana S-1
Jurusan Teknik Elektro

Susunan Tim Penguji

1 Penguji 1

Supriono, ST., MT.
NIP. 19711126 199903 1 004

2 Penguji 2

Ida Bagus Fery Citarsa, ST., MT.
NIP. 19740226 199803 1 004

3 Penguji 3

Sabar Nababan, ST., MT.
NIP. 19710713 200003 1 001

Mataram, Oktober 2016
Dekan Fakultas Teknik
Universitas Mataram

Yusron Saadi, ST., M.Sc., Ph.D.
NIP. 19661020 199403 1 003

PERNYATAAN KEASLIAN TUGAS AKHIR

Dengan ini saya menyatakan bahwa skripsi ini adalah hasil karya saya yang belum pernah diajukan untuk mendapatkan gelar atau diploma pada perguruan tinggi manapun, dan bukan merupakan duplikasi sebagian atau seluruhnya dari karya orang lain yang diterbitkan atau yang tidak diterbitkan, kecuali kutipan berupa data atau informasi yang sumbernya dicantumkan dalam naskah dan Daftar Pustaka.

Pernyataan ini dibuat dengan sebenar-benarnya secara sadar dan bertanggung-jawab, dan saya bersedia menerima sanksi pembatalan skripsi apabila terbukti melakukan duplikasi terhadap karya ilmiah lain yang sudah ada.

Mataram, 22 Oktober 2016

Andryo Iskandar

F1B 012 012

KATA PENGANTAR

Puji Syukur penulis panjatkan kehadiran Allah Yang Maha Esa atas segala berkat, bimbingan, dan karunia-Nya, serta shalawat dan salam kepada Rasulullah SAW yang telah membawa ke alam ilmu pengetahuan sehingga penulis dapat menyelesaikan penyusunan Tugas Akhir dengan judul “RancangBangun Interface untukVisualisasi Proses Sinkronisasi Generator SinkronBerbasisMikrokontroler AT Mega2560”.

Pembuatan Tugas Akhir ini dilaksanakan di Laboratorium Sistem Tenaga Listrik, Jurusan Teknik Elektro Universitas Mataram. Tugas Akhir ini bertujuan untuk merancang visualisasi dalam proses sinkronisasi untuk dapat melihat secara visual syarat-syarat sinkronisasi generator sinkron yang berbasis mikrokontroller AT Mega2560 yang nantinya alat ini akan digunakan sebagai modul praktikum pada Laboratorium Sistem Tenaga Listrik. Tugas Akhir ini juga merupakan salah satu persyaratan kelulusan guna mencapai gelar kesarjanaan di Jurusan Teknik Elektro, Fakultas Teknik UNRAM.

Akhir kata semoga tidaklah terlampau berlebihan, bila penulis berharap hasil karya ini dapat bermanfaat bagi pembaca.

Mataram, 23Oktober 2016

Penulis

UCAPAN TERIMA KASIH

Tugas Akhir ini dapat diselesaikan berkat bimbingan dan dukungan ilmiah maupun materil dari berbagai pihak, oleh karena itu pada kesempatan ini penulis menyampaikan ucapan terima kasih yang setulus-tulusnya kepada:

1. Orang tua tercinta atas kasih sayang, do’a, dukungan, semangat dan segalanya
2. Bapak I Made Ari Nrartha ST.,MT. sebagai dosen pembimbing utama yang telah memberi bimbingan, saran, ilmu dan motivasi sehingga Tugas Akhir ini dapat terselesaikan
3. Bapak I Made Budi Suksmadana, ST., MT. sebagai dosen pembimbing pendamping atas saran, kritik, motivasi, dan tata cara penulisan laporan ilmiah yang baik pada Tugas Akhir ini
4. BapakI Ketut Wiryajati, ST., MT.sebagai dosen Pembimbing Akademik yang membimbing selama masa perkuliahan
5. Bapak Supriono, ST., MT,.Bapak Ida Bagus Fery Citarsa, ST., MT,. Bapak Sabar Nababan, ST.,MT, selaku dosen penguji pada Tugas Akhir ini.
6. Bapak Tohri, ST. selaku teknisi laboratorium Sistem Tenaga Listrik yang telah membantu dalam perlatan penunjang perancangan alat pada Tugas Akhir ini
7. Saudaraku yang terbaik Rypani Atmajaya atas do’a dan semangatnya.
8. Teman-teman angkatan 2012terutama bidang keahlian Sistem Tenaga Listrik yang menemani sehari-hari dalam proses perancangan maupun pengukuran, khususnya Fiqri Jamal, Kuraes Sihab dan Padlulloh, kalian luar biasa.
9. Geng Kapak atas persaudaraan, pengalaman dan petualangan yang takkan terlupa
10. Anak-anak Laboratorium Sistem Tenaga Listrik yang selalu membuat kebahagian dan canda tawa : Janah, Made, Chan, Rfiki, Hamdan, dan semua Genk Lab
11. Teman-teman Kontrakan, kalian teman hidup terbaik dalam perantauan dan membuat hidup lebih berwarna
12. Tim Futsal Teknik FC, Tim yang telah memberikan banyak trofi dan pengalaman
13. Semua pihak yang tidak dapat penulis sebutkan satu persatu, yang telah memberikan bimbingan kepada penulis dalam menyelesaikan Tugas Akhir ini.

Semoga Tuhan Yang Maha Esa memberikan imbalan yang setimpal atas bantuan yang diberikan kepada penulis.

DAFTAR ISI

LEMBAR PENGESAHAN..... i
DAFTAR ISI ii
DAFTAR GAMBAR iv
DAFTAR TABEL..... v

BAB I PENDAHULUAN..... 1
1.1 Latar belakang..... 1
1.2 Rumusan Masalah 2
1.3 Batasan Masalah 2
1.4Tujuan Penelitian 3
1.5 Manfaat Penelitian 3
1.6 Sistematika Penulisan Laporan..... 3
BAB II TINJAUAN PUSTAKA DAN DASAR TEORI 4
2.1 TinjauanPustaka ……..... 4
2.2 LandasanTeori……..... 5
2.2.1 Sistem Tenaga Listrik (STL)..... 5
a. SistemPembangkitanListrik..... 5
b. SistemtransmisidanDistribusi..... 6
c. Proteksi ……..... 7
2.2.2 Generator Sinkron……..... 7
2.2.3 OperasiParalel Generator..... 9
2.2.4 Minimum Sistem Arduino..... 14
a.Fitur-fiturarduino……. 14
b. Kelebihanarduino…… 15
2.2.5 Sensor Tegangan..... 15
2.2.6 Sensor Frekuensi..... 15
BAB III METODE PERANCANGAN..... 16
3.1 MetodologiPenelitian..... 16
3.2 LokasidanWaktuPenelitian..... 16
3.3 AlatdanBahanPenelitian…..... 16

3.3.1 Alat Penelitian..... 16
3.3.2 Bahan Penelitian……..... 17
3.4.Perancangan Sistem..... 17
3.5 PerancanganMasing-masing Blok..... 23
3.5.1 Rancangan Blok Untuk Couple Motor Dc dan Generator Sinkron..... 19
3.5.2 Rancangan Blok untuk Sensor Tegangan dan Sensor Frekuensi..... 19
a. Rangkaian Sensor Tegangan..... 19
b. Rangkaian Sensor Frekuensi..... 20
3.5.3 Perancangan Blok Interface danKonttrol Relay 21
a. Relay……..... 21
b. Pemrograman Mikrokontroller ATMega 2560 22
c. Interface Hasil Pembacaan 23
3.6 Perancangan Perangkat Lunak..... 25
3.6.1PemrogramanUntukMikrokontroller ATMega328…… 25
3.6.2 Interface Hasil Pengukuran..... 26
3.6.3 Diagram Alir Penelitian..... 28
3.6.3 Diagram Alir Arduino..... 29
3.6.3 Diagram Alir Matlab..... 30
BAB IV. PEMBAHASAN..... 35
4. Pengujian Sistem..... 35
4.1 Pengujian Rangkaian Sensor Tegangan..... 35
4.1.1 Pengujian Sensor Tegangan Fasa R pada PLN..... 37
4.1.2 Pengujian Sensor Tegangan Fasa S pada PLN..... 40
4.1.3 Pengujian Sensor Tegangan Fasa T pada PLN..... 43
4.1.3 Pengujian Sensor Tegangan Fasa R pada Generator..... 45
4.1.3 Pengujian Sensor Tegangan Fasa S pada Generator..... 48
4.1.3 Pengujian Sensor Tegangan Fasa T pada Generator..... 50
4.2 Pengujian Rangkaian Sensor Frekuensi..... 53
4.2.1 Pengujian Sensor Frekuensi pada PLN..... 56
4.2.2 Pengujian Sensor Frekuensi pada Generator..... 58
4.3 Pembuatan GUI (Grapic User Interface) Pada Komputer..... 60

4.4 Pengukuran dan Pengujian Keseluruhan Sistem..... 62
4.4.1 Pengukuran dan Pengujian Keseluruhan Sistem..... 62
4.4.2 Tampilan Interface Setelah Masuknya SinkronisasI..... 64
BAB V. KESIMPULAN DAN SARAN..... 65
DAFTAR PUSTAKA..... 66
LAMPIRAN..... 61

DAFTAR GAMBAR

Gambar 2.1 Kerja Parallel Generator……..... 9
Gambar 2.2 Besar Magnitude Tegangan..... 10
Gambar 2.3 Frekuensi..... 11
Gambar 2.4 Indikator Lampu Sama Fasa..... 12
Gambar 2.5 IndikatorLampuRotasiFasadanSamaFasa..... 13
Gambar 2.6 Board Arduino..... 14
Gambar 3.1 Diagram bloksistem*sinkronisasi generator sinkrondengan jala-jala PLN*..... 18
Gambar 3.2 Couple motor Dc dan Generator Sinkron..... 19
Gambar 3.3 Rangkaian Sensor Tegangan..... 20
Gambar 3.4 Sensor Frekuensi..... 21
Gambar 3.5 Relay optocoupler 4 channel..... 22
Gambar 3.6 Tampilan IDE Arduino..... 23
Gambar 3.7 Tampilan pada Matlab..... 24
Gambar 3.8 Diagram alir penelitian…..... 25
Gambar 3.9 Diagram AlirPerancanganAlat……. 27
Gambar 3.10 Diagram alir program pada arduino..... 29
Gambar 3.11 Diagram alir program pada matlab..... 30
Gambar 4.1 Pengujian trafo step down..... 33
Gambar 4.2 Gelombang primer dan sekunder trafo 1 sumber PLN..... 33
Gambar 4.3 Gelombang primer dan sekunder trafo 2 sumber PLN..... 33
Gambar 4.4 Gelombang primer dan sekunder trafo 3 sumber PLN..... 33

Gambar 4.5 Gelombang primer dan sekunder trafo 4 sumber PLN..... 34
Gambar 4.6 Gelombang primer dan sekunder trafo 5 sumber PLN..... 34
Gambar 4.7 Gelombang primer dan sekunder trafo 6 sumber PLN..... 35
Gambar 4.8 Sensor tegangan..... 35
Gambar 4.9 Pendekatan Linear..... 38
Gambar 4.10 Grafik pengujian sensor tegangan fasa R pada PLN..... 40
Gambar 4.11 Grafik pengujian sensor tegangan fasa S pada PLN..... 42
Gambar 4.12 Grafik pengujian sensor tegangan fasa T pada PLN..... 45
Gambar 4.13 Grafik pengujian sensor tegangan fasa R pada Generator..... 47
Gambar 4.14 Grafik pengujian sensor tegangan fasa S pada Generator..... 50
Gambar 4.15 Grafik pengujian sensor tegangan fasa T pada Generator..... 52
Gambar 4.16 Sensor Frekuensi..... 53
Gambar 4.17 Persamaan linear sensor frekuensi..... 55
Gambar 4.18 Grafik pengujian Sensor frekuensi pada PLN..... 57
Gambar 4.19 Grafik pengujian Sensor frekuensi pada Generator..... 59
Gambar 4.20 Main menu program interface..... 61
Gambar 4.21 Tampilan interface setelah sinkron..... 64
Gambar 4.21 Tampilan interface setelah sinkron..... 65
Gambar 4.21 Tampilan interface setelah sinkron..... 66
Gambar 4.21 Tampilan interface setelah sinkron..... 67

Daftar Tabel

Tabel 2.1 Spesifikasi Arduino Mega 2560..... 14
Tabel 3.1 Alat ukur yang digunakan..... 16
Tabel 3.2 Peralatan-peralatan yang digunakan..... 17
Tabel 4.1 Nilai tegangan input dan output fasa R pada PLN..... 37
Tabel 4.2 Persentase error masing-masing persamaan linear..... 38
Tabel 4.2
Tabel 4.3 Besar tegangan yang terukur pada analog dan interface..... 39
Tabel 4.4 Nilai tegangan input dan output fasa S pada PLN..... 40
Tabel 4.5 Besar tegangan yang terukur pada analog dan interface..... 42
Tabel 4.6 Nilai tegangan input dan output fasa T pada PLN..... 43

Tabel 4.7	Besar tegangan yang terukur pada analog dan interface.....	44
Tabel 4.8	Nilai tegangan input dan output fasa R pada Generator.....	45
Tabel 4.9	Besar tegangan yang terukur pada analog dan interface.....	47
Tabel 4.10	Nilai tegangan input dan output fasa S pada Generator.....	48
Tabel 4.11	Besar tegangan yang terukur pada analog dan interface.....	49
Tabel 4.12	Nilai tegangan input dan output fasa T pada Generator.....	50
Tabel 4.13	Besar tegangan yang terukur pada analog dan interface.....	52
Tabel 4.14	Rata-rata error sensor tegangan.....	53
Tabel 4.15	Nilai tegangan input dan output sensor frekuensi pada PLN	54
Tabel 4.16	Persentase error persamaan linear sensor frekuensi	56
Tabel 4.17	Besar frekuensi yang terukur pada analog dan interface PLN.....	57
Tabel 4.18	Besar frekuensi yang terukur pada analog dan interface Generator.....	59
Tabel 4.19	Rata-rata error sensor frekuensi.....	60
Tabel 4.20	Waktu Sinkronisasi.....	63

Daftar Lampiran

Lampiran 1 Foto-foto dan gambar penelitian

Lampiran 2 Listing program arduino dan matlab

ABSTRAK

Penelitian ini membahas tentang perancangan Visualisasi untuk melihat proses sinkronisasi antara generator sinkron dengan jala-jala PLN. Pada generatror sinkron dan jala-jala PLN dipasang 6 sensor tegangan dengan 3 sensor pada generator sinkron dan 3 sensor untuk jala-jala PLN, yang diletakkan pada tiap fasa untuk melihat besar tegangan pada setiap fasanya, selain itu juga untuk melihat besar frekuensi dipasang sensor frekuensi pada pada salah satu fasa generator sinkron dan jala-jala PLN. Data dari masing-masing sensor akan dikirim ke Arduino kemudian akan ditampilkan pada Matlab.

Waktu penyambungan antara generator sinkron dan jala-jala PLN secara otomatis yaitu sebesar 0.0401 detik setelah keempat syarat sinkronisasi terpenuhi yaitu fasa yang sama, tegangan yang sama, frekuensi yang sama, dan sudut fasa yang sama.

Pengujian sensor tegangan dan sensor frrekuensi menggunakan persamaan polinomial orde 1. Setelah melakukan pengujian pada sensor tegangan didapatkan persen error rata-rata sebesar 1.8% untuk sensor tegangan dan 0.44 % untuk sensor frekuensi.

Kata Kunci:

Visualisasi, Sinkronisasi, Sensor Tegangan, Sensor Frekuensi

ABSTRACT

This study discusses the design visualization to see the synchronization process between synchronous generator with PLN. In synchronous generator and PLN installed 6 voltage sensor with 3 sensors on synchronous generators and three sensors for PLN, which is placed on each phase to see a large voltage on each phase, and also to see the frequency of installed frequency sensor on the one-phase synchronous generator and PLN. Data from each sensor will be sent to the Arduino will then be displayed in Matlab.

The time between the connection of synchronous generators and PLN automatically in the amount of PLN 0.0401 seconds after the fourth synchronization requirements are the same phase, same voltage, same frequency and same phase angle.Testing the voltage sensor and frekuency sensor use order polynomial equation 1. After testing the voltage sensor error obtained percent on average by 1.8% for the voltage sensor and 12:44% to frequency sensor.

Keywords:

Visualization, Synchronization, Voltage Sensor, Frequency Sensor

BAB I
PENDAHULUAN

1.1 Latar Belakang

Sistem tenaga listrikdibagi ke dalam tiga bagian pentingyaitu, sistem pembangkitan, sistem transmisi dansistem distribusi. Semua bagian sistem tenagalistrik memilikiperanan yang pentingdalammenunjangketersediaan energi listrikkepelanggan. Apabilaketigabagianiniterjadimasalahmakaakan menyebabkan terganggunya penyaluranenergilistriksampai kepelanggan.

Salah satu bagian sistem tenaga listrik yang penting adalah pada sistem pembangkitan tenaga listrik, itu dikarenakan apabila terjadi pemadaman pada sistem ini maka secara otomatis supplai listrik ke konsumen menjadi tidak ada sehingga terjadi pemadaman listrik.

Masalah yang dihadapi pada sistem pembangkit adalah adanya kekurangan daya yang disuplai oleh generator yang disebabkan karena kebutuhan beban listrik yang semakin meningkat, itu menyebabkan generator tidak mampu untuk menanggung beban yang melebihi kapasitas yang bisa ditanggung, maka dari itu metode yang digunakan saat ini yaitu melakukan parallel generator yang bertujuan menggabung dua generator atau lebih untuk menghasilkan daya listrik yang lebih besar sehingga kekurangan daya listrik dapat teratasi.

Dalam metode parallel generator sinkron yang satu dengan generator sinkron yang lainnya harus terjadi kesinkronisasian. Adapunbeberapa syarat yang harus dipenuhi oleh setiap generator sehingga terjadi kesinkronisasian pada setiap generator tersebut sebelum bekerja bersamaan adalah, memiliki magnitude tegangan (V) generator yang harus sama, frekuensi yang dimiliki harus sama (f), sudut fase yang sama dan urutan fase yang sama.

Metode sinkronisasi generator sinkron dengan jala-jala PLN sudah dipratikkan pada praktikum mesin-mesin listrik di Laboratorium Sistem Tenaga Listrik pada percobaan 4. Pada percobaan tersebut digunakan alat untuk melihat sinkronisasi antara generator dengan jala-jalan PLN menggunakan alat *Sinkronizing Indikator* dengan indikasi sinkronisasi sudah terjadi apabila lampu pada alat tersebutdalam keadaan padam dikarenakan sudah tidak adanya perbedaan tegangan

dan frekuensi.Selain itu juga *Sinkronizing Indikator* ini sangat mengandalkan ketelitian dari manusia untuk melihat lampu-lampu pada alat tersebut.

Oleh karena itu diperlukan sebuah interface untuk visualisasi sinkronisasi generator sinkron jala-jala PLN agar bisa mengetahui sinkronisasisemua variable-variabel (frekuensi,tegangan dan sudut fasa) dengan lebih teliti dan pastidalam penggunaan skala laboratorium.

1.2Rumusan Masalah

Rumusaan masalah yang akan dibahas penulis dalam penelitian ini adalah :.

1. Bagaimana menguji sensor tegangan dan sensor frekuensi pada interface dalam proses sinkronisasi.

2. Bagaimana merancang dan membangun interface untuk visualisasi sinkronisasi generator sinkron dengan dengan jala-jala PLN berbasis mikrokontroller ATmega 2560.

1.3 Batasan Masalah

Adapun batasan permasalahan dalam penelitian ini adalah :

1. Penelitian ini menggunakan generator sinkron 0,3 kW yang ada pada Laboratorium Sistem Tenaga Listrik.
2. Interface tidak melakukan pengaturan variabel seperti frekuensi, tegangan dan beda fase pada saat melakukan sinkronisasi.
3. Urutan fasa diatur terlebih dahulu dengan melihat pada sumber 3 fasa jala-jala PLN dan Generator Sinkron.

1.4 Tujuan Penelitian

Tujuan dari penulis melakukan penelitian ini adalah :

1. Mendapatkan nilai tegangan dan frekuensi dengan akurasi yang baik dan tepat pada interface untuk melakukan sinkronisasi.
2. Mendapatkan interface untuk visualisasi sinkronisasi generator sinkron dengan generator sinkron atau generator sinkron dengan jala-jala PLN berbasis mikrokontroller ATmega 2560 (Arduino Mega).

1.5 Manfaat Penelitian

Penyusunan Tugas akhir ini dharapkan dapat bermanfaat

1. Bagi penulis untuk menambah ilmu pengetahuan mengenai caramerancangdanmembanguninterface untukvisualisasisinkronisasi generator sinkrondenganjala-jala PLN berbasismikrokontroller ATmega 2560.
2. Apabilanantinyaalatinisudahjadidiharapkandapatbergunauntukmodulpraktikum di LaboratoriumSistemTenagaListrik.
3. Diharapkan dapat menjadi bahan referensi/acuan di Universitas Mataram pada umumnya dan di Fakultas Teknik khususnya untuk penelitian yang berhubungan dengan perancangandanpembuataninterface untuk proses sinkronisasi generatorsinkrondenganbaikdantepat.

1.6 Sistematika Penulisan

Sistematika penulisan laporan tugas akhir ini adalah sebagai berikut:

1. BAB I PENDAHULUAN

   Bab I berisi penjelasan mengenai latar belakang, rumusan masalah, tujuan, batasanmasalah, metodologi, serta sistematika penulisan yang digunakan untuk menyusun laporan tugas akhir.
2. BAB II DASAR TEORI

   Bab II berisi dasar teori yang digunakan dalam melakukan analisis, perancangan, dan implementasi tugas akhir yang dilakukan pada bab-bab selanjutnya.
3. BAB III METODE PENELITIAN

   Berisi tentang rencana pelaksanaan, alat, bahan, jalannya perencanaan dan hasil yang diharapkan.
4. BAB IV HASIL DAN PEMBAHASAN

   Pada bab ini akan dibahas tentang hasil yang diperoleh dari perancangan dan pengujian *interface* yang dibuat.
5. BAB V KESIMPULAN DAN SARAN

   Merupakan bagian akhir yang berisikan kesimpulan dan saran yang didapatkan dari hasil dan pembahasan.

BAB II
LANDASAN TEORI

2.1. Tinjauan Pustaka

Berdasarkan penelitian-penelitian mengenai rancang bangun interface untuk visualisasi proses sinkronisasi yang telah dilakukan dengan hasil-hasil yang sudah di publikasikan baik secara Internasional, Nasional maupun Lokal sebagai berikut ini.

Effendi (2012), telah melakukan penelitian tentang Rancang Bangun Alat Sinkronisasi Otomatis Sistem Jaringan Pembangkit Listrik Tenaga Mikrohidro dengan Sistem Jaringan PLN. Pada tugas akhir ini,alat sinkronisasi otomatis yang dibuat menggunakan rangkaian detektor tegangan dan frekuensi dan hasilnya akan dibandingkan dalam sebuah IC mikrokontroller. Hasil pengujian didapatkan bahwa alat sinkronisasi dapat bekerja dengan baik dengan prosentase rata-rata kesalahan untuk pengujian tegangan listrik sebesar 2,35% dan prosentase rata-rata kesalahan untuk pengujian frekuensi sebesar 0,34%.

Nurcahyo (2012), telah melakukan penelitian tentang Sinkronisasi dan Pengamanan Modul Generator LAB-TST Berbasis PLC *("Software")*. Pada proyek akhir ini penulis membuat sebuah alat sinkronisasi dan pengamanan modul generator Lab-TST berbasis PLC yang diharapkan bisa mempermudah proses sinkronisasi generator. PLC yang akan digunakan adalah Zelio SR3B261BD yang berfungsi sebagai pengendali proses sinkronisasi generator berdasarkan input dari pembacaan sensor dan untuk mengontrol kerja kontaktor agar bekerja sesuai dengan parameter yang diinginkan. Dari hasil yang telah dilakukan generator akan sinkron apabila syarat kerja paralel telah terpenuhi yaitu ketika tegangannya sama 380 volt dan frekuensinya 50 Hz dengan urutan fasa yang sama.

Prasetya (2012), telah melakukan penelitian tentang Sinkronisasi dan Pengamanan Modul Generator LAB-TST Berbasis PLC (Hardware). Pada proyek akhir ini digunakan PLC (Zelio) sebagai kendali utama dan pendeteksi parameter-parameter yang dibutuhkan, yaitu frekuensi, tegangan dan beda fasa. Penggunaan sensor-sensor yang dapat mendeteksi parameter-parameter sinkron serta tambahan kontroler dapat disusun menjadi suatu sistem yang handal dan membantu proses sinkronisasi yang sebelumnya masih manual. Pada pengujian tiap-tiap bagian tidak ditemukan permasalahan yang besar. Sensor tegangan memberikan nilai yang linear saat

pengukuran tegangan 0-400 v. Pembacaan sensor frekuensi juga memiliki nilai yang linear mulai 41-59 Hz. Hasil dari sinkronisasi ini kontaktor rele 1 nyala terlebih dahulu, kontaktor rele 2 nyala saat parameter tegangan 380 v, frekuensi 50 Hz, urutan phasa yang sama, dan tidak ada selisih sudut phasa antara kedua generator.

Graha (2012), telah melakukan penelitian tentang power management PLN-Genset pada Bank Indonesia Cabang Banjarmasin. Pemindahan aliran listrik antara sumber PLN dan Genset diperlukan suatu alat switch yang saling interlok satu sama lain dikenal dengan Change Over Switch (COS). Adanya power managemen pada modul GCP 30 dapat memberikan sistem parallel generator yang baik dimana dapat dilakukan secara otomatis dengan pembagian beban secara proposional dan pengaturan bahan bakar stabil dan dapat mengurangi pemakaian bahan bakar sebesar 15% dari data konsumsi bahan bakar.

2.2. Landasan Teori

2.2.1 Sistem Tenaga Listrik

Dalam sistem tenaga listrik terdapat tiga bagian yang sangat penting yaitu ;

a. Sistem pembangkitan listrik

b. Sistem transmisi

c. Sitem distribusi

Berikut penjelasan masing-masing bagian :

a. Sistem pembangkitan listrik

Salah satu cara yang paling ekonomis, mudah, dan aman untuk mengirimkan energi adalah melalui bentuk energi listrik (Zuhal, 1988). Pada pusat pembangkit, sumber daya energi primer seperti bahan bakar fosil (minyak,gas alam, dan batu bara), hidro, panas bumi dan nuklir diubah menjadi energi listrik. Generator sinkron mengubah energi mekanis yang dihasilkan pada poros turbin menjadi energi listrik tiga fasa.

Melalui transformator penaik tegangan (step-up transformator) energi listrik ini kemudian dikirimkan melalui saluran transmisi bertegangan tinggi menuju pusat-pusat beban. Peningkatan tegangan dimaksudkan untuk mengurangi jumlah arus yang mengalir pada saluran transmisi. Dengan demikian saluran transmisi bertegangan tinggi akan membawa aliran arus yang rendah dan berarti mengurangi rugi panas (heat loss) I^2R yang menyertainya. Ketika saluran transmisi mencapai pusat beban, tegangan

tersebut kembali diturunkan menjadi tegangan menengah, melalui transformator penurun tegangan (step-down transformator).

Di pusat-pusat beban yang terhubung dengan saluran distribusi, energi listrik ini diubah lagi menjadi bentuk-bentuk energi terpakai lainnya seperti energi mekanis (motor), penerangan, pemanas, pendingin dan sebagainya.

Pusat pembangkit berfungsi untuk mengkonversikan sumber daya energi primer menjadi energi listrik. Pusat pembangkit listrik konvensional mencakup :

1. Pusat Listrik Tenaga Uap (PLTU) ; minyak, gas alam, dan batubara.
2. Pusat Listrik Tenaga Air (PLTA).
3. Pusat Listrik Tenaga Gas (PLTG)
4. Pusat Listrik Tenaga Diesel (PLTD)
5. Pusat Listrik Tenaga Panas Bumi (PLTP)
6. Pusat Listrik Tenaga Nuklir (PLTN)

Di samping pembangkit listrik konvensional tersebut, saat ini tengah dikembangkan beberapa teknologi konversi untuk sumber daya energi baru seperti : biomassa, solar, limbah kayu, angin, gelombang laut dan sebagainya. Pembangkit listrik melalui cara magneto hidro dinamik (MHD) pada saat ini juga sedang memasuki tahap penelitian dan pengembangan yang intensif.

b. Sistem Transmisi dan Distribusi

Apabila saluran transmisi menyalurkan tenaga listrik bertegangan tinggi ke pusat-pusat beban dalam jumlah besar, maka saluran distribusi berfungsi membagikan tenaga listrik tersebut kepada pihak pemakai melalui saluran tegangan rendah.

Generator sinkron di pusat pembangkit biasanya menghasilkan tenaga listrik dengan tegangan antara 6-20 kV yang kemudian, dengan bantuan transformartor tegangan tersebut dinaikkan menjadi 150-500 kV. Saluran tegangan tinggi (STT) menyalurkan tenaga listrik menuju pusat penerima ; di sini tegangan diturunkan menjadi tegangan subtransmisi 70 kV. Pada gardu induk (GI), tenaga listrik yang diterima kemudian dilepaskan menuju trafo distribusi (TD) dalam bentuk tegangan menengah 20 kV. Melalui trafo distribusi yang tersebar di berbagai pusat-pusat beban, tegangan distribusi primer ini diturunkan menjadi tegangan rendah 220/380 V yang akhirnya diterima pihak pemakai.

Sistem tenaga listrik dirancang untuk dapat mengirim energi listrik dengan cara yang efisien dan aman kepada para langganan. Karakteristik dari permintaan energi listrik kadang kala membuat usaha tersebut sulit untuk dipenuhi. Meramalkan pertumbuhan beban dan usaha untuk memenuhi siklus beban harian dan beban tahunan secara memuaskan merupakan dua kesulitan yang harus diatasi.

Pertumbuhan rata-rata konsumsi listrik di Indonesia pada Pelita II dan Pelita III masing-masing mencapai 14,1% dan 12,7 % per tahun (Zuhal, 1988). Selama sepuluh tahun itu (1974/75 – 1983/84) konsumsi listrik total telah meningkat sebanyak tiga kali. Dalam pelita IV (1983/84 – 1988/89) pertumbuhan rata-rata pemakaian listrik diperkirakan sekitar 13-15% per tahun. Mengingat untuk membangun suatu pusat pembangkit tenaga listrik diperlukan waktu 8 sampai 10 tahun (Zuhal, 1988), maka para perencana sistem harus melihat kemungkinan perkembangan sistem tenaga 10 sampai 20 tahun ke muka. Hal tersebut diperlukan agar tersedia cukup waktu untuk memperkirakan dan memperbaiki perencanaan dalam perspektif jangka panjang.

c. Proteksi

Suatu gangguan atau kegagalan dalam keadaan bagaimanapun akan mempengaruhi arus normal pada sistem tenaga. Gangguan-gangguan yang terjadi dapat disebabkan oleh sambaran petir, hubungan singkat karena kejatuhan benda tertentu pada kawat penghantar, rusaknya isolasi dan lain sebagainya. Gangguan-gangguan tersebut dapat mengakibatkan lonjakan tegangan yang berlebihan, aliran arus yang sangat besar, bunga api listrik dan kegagalan sistem tenaga untuk beroperasi secara keseluruhan. Menjadi tugas insinyur listrik pula untuk merancang system proteksi dengan mengatur pemakaian sekering (fuse), pemutus daya (circuit breaker) dan system relay yang mampu menemukan gangguan dengan cepat serta memisahkannnya segera dari bagian system yang lain. Dengan rancangan system proteksi yang baik, gangguan-gangguan yang terjadi dapat dilokalisir pada daerah kejadian saja sehingga tidak mengganggu para langganan di daerah lain.

2.2.2 Generator Sinkron

Secara umum generator sinkron mempunyai kumparan stator dan kumparan rotor yang akan menghasilkan energi listrik.Generator sinkron merupakan mesin sinkron yang mengubah dari energi mekanik ke energi listrik. Energi mekanik berupa

putaran rotor yang digerakan oleh pengerak mula (prime mover) yang akan menghasilkan medan magnet putar dengan kecepatan dan arah putar yang sama dengan putaran rotor, hubungan antara kecepatan putar dengan putaran rotor disebut dengan frekuensi, persamaan antara kecepatan putar medan magnet dengan frekuensi listrik pada stator adalah :

$$n = \frac{120 \text{ x f}}{P} \tag{2.1}$$

Keterangan :

f = frekuensi listrik (Hz)

p = jumlah kutub

n = kecepatan putar rotor (rpm)

Medan putar yang dihasilkan pada rotor akandi induksikan pada kumparan jangkar yang menghasilkan fluks magnetik yang berubahubah besarannya terhadap waktu. Perubahan fluks magnetik akan menghasilkan ggl induksi pada ujung kumparan terdapat dalam persamaan :

$$E_{eff} = 4{,}44.\ f\ .\theta_m\ .\ N \tag{2.2}$$

Keterangan :

E_{eff} = ggl induksi efektif (Volt)

P = jumlah kutub

f = frekuensi listrik (Hz)

θ = fluks magnet (weber)

n = kecepatan putar rotor (rpm)

N = jumlah lilitan

Efesiensi generator merupakanperbandingan antara daya output generator berbanding lurus dengan daya input mekanis generator sebagai sebuah persamaan:

$$\eta = \frac{\text{Pout}}{\text{Pin}} \tag{2.3}$$

Keterangan :

η = effesiensi generator

Pout = Daya output generator (watt)

Pin = Daya mekanik output generator (watt)

Selanjutnya, dengan hukum kekekalan energy (perlu diketahui bahwa energy hanyalah produk dari daya dikalikan dengan waktu),daya input mekanis sama dengan daya output listrik ditambah dengan kerugian sehingga di peroleh persamaan :

Pin = Pout + Ploses (2.4)

Keterangan :

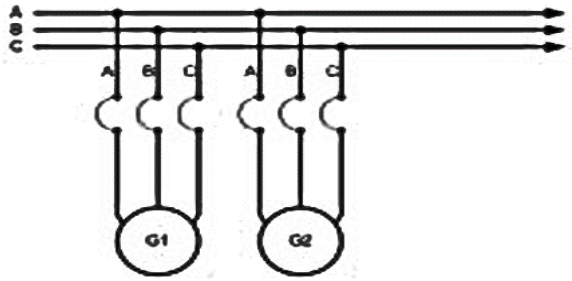
Pin = Daya mekanik output generator (watt)

Pout = Daya output generator (watt)

Plosses = Daya output generator (watt)

2.2.3 Operasi Paralel Generator

Paralel generator sinkron dapat dikatakan dengan menggabungkan dua buah generator atau lebih secara bersamaan sistem kerja paralel ini bertujuan untuk mendapatkan daya yang lebih besar, untuk efesiensi hal ini untuk menghemat biaya pemakaian operasional dan biaya pembelian kapasitas generator, dan menjamin kontinyutas ketersediaan daya listrik. Sistem paralel generator disebut juga dengan sinkronisasi generator



Gambar 2.1 Kerja Paralel Generator

(Graha, 2014)

Sebelum generator di paralel kondisi sinkron harus dipenuhi, adapun persyaratannya (Citarsa, 2013) yaitu :

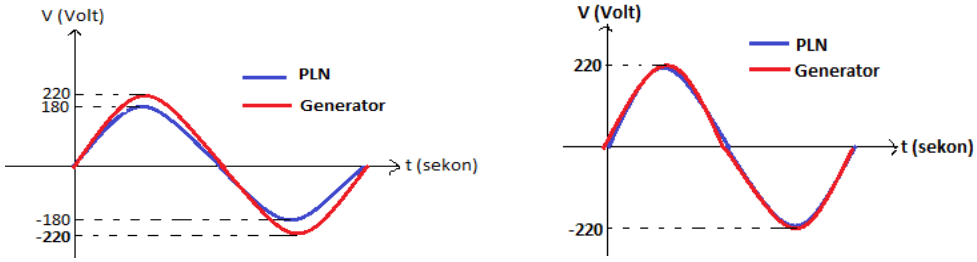
1. Tegangan atau magnitudeterminal dari generator sinkron harus samadengan tegangan terminal generator sinkron lainnya atau tegangan bus bar jaringan listrik yang akan dihubungkan dengan generator sinkron tersebut.
2. Putaran penggerak mula dari rotor generator sinkron harus memiliki kecepatan sedemikian rupa sehingga tegangan listrik yang dihasilkan generator sinkron memiliki frekuensi yang sama dengan frekuensi tegangan generator sinkron lainnya atau frekuensi tegangan bus bar jaringan listrik yang akan dihubungkan dengan generator sinkron tersebut.
3. Fasa (waktu awal munculnya gelombang) tegangan terminal dari generator sinkron harus sama dengan fasa tegangan terminal generator sinkron lainnya atau fsa

tegangan bus dari jaringan listrik yang akan dihubungkan dengan dengan generator sinkron tersebut. Untuk dapat tercapainya kondisi ini maka saklar penghubung generator sinkron dengan generator sinkron atau jaringan listrik harus ditutup pada (atau mendekati) saat kedua tegangan yang diparalel sudah memiliki fasa yang sama.

Untuk bisa memenuhi ketiga persamaan di atas maka diperlukan beberapa peralatan untuk menunjukkan indikasi bahwa ketiga syarat tersebut sudah terpenuhi. Untuk syarat pertama dibutuhkan voltmeter untuk menunjukkan bahwa tegangan terminal generator sinkron sudah sama dengan tegangan generator sinkron lainnya atau tegangan bus bar jaringan listrik. Untuk syarat kedua dan ketiga dibutuhkan lampu sinkronisasi untuk menunjukkan bahwa frekuensi dan fasa tegangan terminal generator sinkron sudah sama dengan frekuensi dan fasa generator sinkron lainnya atau tegangan bus bar jaringan listrik.

Adapaun contoh dari gambar yang akan ditampilkan pada Gambar 2.2 dan 2.3 di bawah ini :

a. Besar magnitude tegangan yang sama

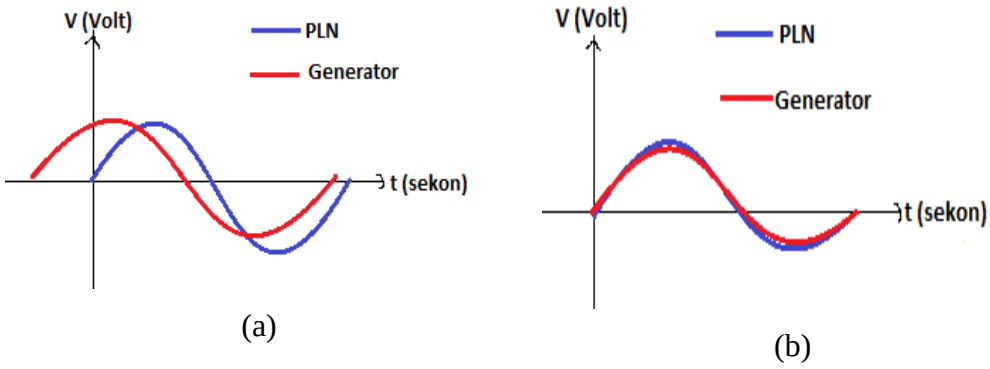


(a) (b)

Gambar 2.2 Besar magnitude tegangan

Pada Gambar 2.2 bagian (a) besar magnitude tegangan belum sinkron antara Generator Sinkron dengan Jala-jala PLN, sedangkan untuk Gambar 2.2 bagian (b) besar magnitude tegangan sudah sinkron antara Generator Sinkron dengan Jala-jala PLN.

b. Sudut Fase



(a) (b)

Gambar 2.3 Sudut Fase

Pada Gambar 2.3 bagian (a) Sudut Fase yang berbeda antara generator dan jala-jala PLN, sedangakan pada Gambar 2.3 bagian (b) Sudut Fase yang sama antara generator dan jala-jala PLN

Selain itu juga menurut Suparno (2000) proses penyambungan/paralel pembangkit dengan jaringan listrik harus didahului dengan proses sinkronisasi. Proses sinkronisasi dapat dilakukan jika syarat-syarat dibawah ini dipenuhi, antara lain besartegangannya sama, frekuensinya sama, urutan fase sama dan beda fasenya sama. Untuk menyamakan ke 4 syarat diatas generator diperlakukan sebagai berikut:

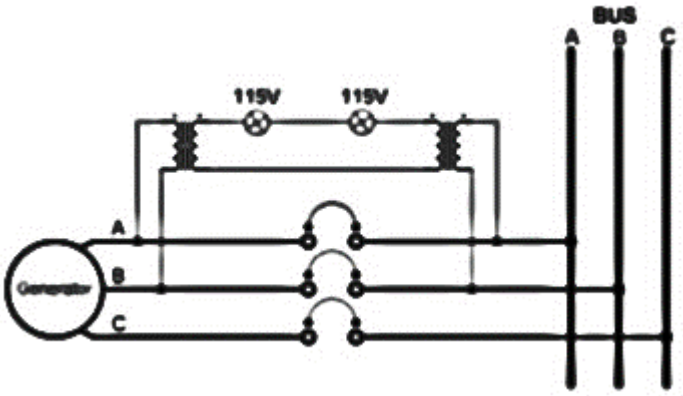
1. Penyamaan Tegangan. Atur tegangan generator dengan mengatur arus penguatan (Iex) sampai dengan tegangannya sama dengan tegangan jaringan PLN.
2. Penyamaan Frekuensi. Besarnya frekuensi tergantung dengan kecepatan putar motor.
3. Beda Fase Sama (Tegangannya Sefase). Tegangan generator dan tegangan jaringan PLN merupakan tegangan AC, sehingga besar tegangan setiap saat berbeda. Jika tegangan dengan urutan fase yang sama, diukur pada masing-masing tegangan fase yang sama (misal fase R generator dan fase jaringan) dengan menggunakan voltmeter, penunjukan voltmeter akan berubah-ubah sesuai dengan beda fase kedua tegangan tersebut. Jika voltmeter menunjuk angka nol maka kedua tegangan tersebut dikatakan sefase dan jika voltmeter menunjukkan angka tertentu, maka kedua tegangan tersebut mempunyai

beda fase. Untuk menyamakan fasenya dilakukan dengan mengatur frekuensi generator.

Strategi dalam memparalelkan alternator atau menambahkan sebuah generator sinkron pada jaringan sistem tenagayang telah ada harus dilakukan tahapan-tahapan sebagai berikut :

a. Generator yang akan ditambahkan dijalankan hingga mencapai kecepatan putar nominalnya.
b. Tahanan pengatur medannya diatur sedemikian hingga tegangan generatornya menjadi sedikit lebih tinggi daripada tegangan jaring. Tegangannya dapat diperiksa dengan menggunakan saklar pilih voltmeter.
c. Generator tadi kemudian dihubungkan dengan jaringan. Karena tegangannya sedikit lebih tinggi daripada tegangan jaring, alternator ini tidak akan bekerja sebagai motor.
d. Selanjutnya tahanan pengatur medannya diatur sedemikian hingga generator tersebut memikul sebagian dari beban jaring sistem yang dimasukinya. Besar beban alternator ini dapat dilihat dari penunjukan alat ukur amperemeternya.

Sinkronisasi mengandalkan peralatan dan ketelitian seorang operator ketika kedua atau beberapa generator aman untuk di paralel. Peralatan yang harus ada pada kondisi manual paralel adalah synchroscope, dobel volt meter, dobel frekuensi meter,lampu indikator kondisi parallel generator.

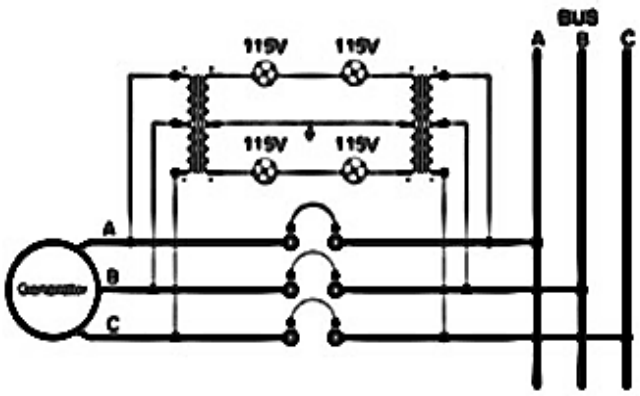


Gambar 2.4 Indikator lampu sama fasa

(Graha, 2014)

Pada Gambar 2.4 menunjukkan salah satu metode yang menggunakan dua lampu 115 Vac untuk memeriksa apakah dua tegangan berada dalam sefasa atau keluar

dari fasa. Ketika tegangan berada dalam fasa, lampu akan dipadamkan, dan ketika tegangan berada di luar fasa, lampu akan terang.



Gambar 2.5 Indikator lampu rotasi fasa dan sama fasa

(Graha,2014)

Pada Gambar 2.5 menunjukkan metode lain, menggunakan empat lampu 115 Vac, yang akan memeriksa rotasi fasa serta beda fasa seperti sebelumnya, ketika tegangan berada difasa, semua lampu akan dimatikan, dan ketika tegangan berada di luar fasa, semua lampu akan menyala. Jika pasang lampu bergantian terang dan gelap (dengan dua lampu gelap sementara dua lainnya terang) mengindikasikan urutan fasa tidak sama.

Pada pusat-pusat pembangkit tenaga listrik, biasanya menggunakan alat synchroscope untuk memparalel generator. Penggunaan alat ini telah dilengkapi dengan Voltmeter untuk memonitor kesamaan tegangan dan Frekuensi meter untuk kesamaan frekuensi.

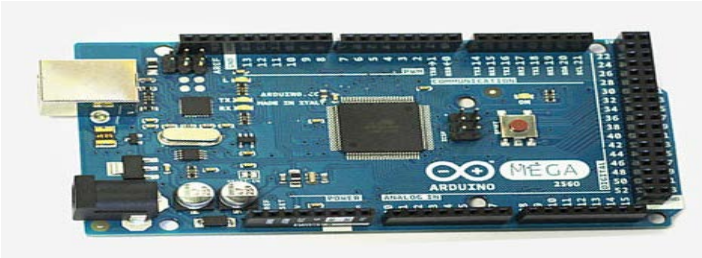
Peralatan sinkroskop berupa jarum penunjuk sebagai indicator yang akan bergerak ke arah berlawanan putaran jarum jam saat kecepatan putaran generator sinkron lebih lambat dari generator sinkron yang terlebih dahulu beroperasi (atau jaringan listrik) dan searah putaran jarum jam saat kecepatan putaran generator sinkron lebih cepat.

Ketepatan sudut phasa dapat dilihat dari synchroscope. Bila jarum penunjuk berputar berlawanan arah jarum jam berarti frekuensi generator yang baru masuk lebih rendah dan bila searah jarum jam berarti lebih tinggi. Pada saat jarum telah diam dan menunjuk pada kedudukan vertikal, berarti beda phasa generator dan jala-jala telah 0 (Nol) dan selisih frekuensi telah 0 (Nol), maka pada kondisi ini sakelar dimasukkan

(ON). Alat synchroscope tidak bisa menunjukkan urutan phasa jala-jala, sehingga perlu dipakai indikator urutan phasa jala-jala untuk memparalelkan generator.

2.2.4 Minimum Sistem Arduino

Pada penelitian ini digunakan Arduino Mega 2560.Arduino merupakan sebuah platform komputasi fisik berbasis papan mikrokontroler sederhana.Project arduino memiliki cakupan pengembangan yang luas karena bersifat *open source* dan mudah digunakan.



Gambar 2.6 Arduino Mega 2560

(www.arduino.cc (2015))

Arduino Mega 2560 memiliki spesifikasi sebagai berikut:

Tabel 2.1 Spesifikasi Arduino Mega 2560

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

Sumber (www.arduino.cc ,2015)

a. Kelebihan Arduino

Dibandingkan dengan mikrokontroller yang lain, Arduino memiliki beberapa kelebihan yang menbuatnya banyak dipilih untuk berbagai projek, kelebihan tersebut yaitu:

a. Tidak perlu perangkat chip programmer karena di dalamnya sudah ada bootloader yang akan menangani upload program dari komputer.
b. Sudah memiliki sarana komunikasi USB, sehingga pengguna Laptop yang tidak memiliki port serial/RS323 bisa menggunakannya.
c. Bahasa pemrograman relatif mudah karena software Arduino dilengkapi dengan kumpulan library yang cukup lengkap.
d. Memiliki modul siap pakai (shield) yang bisa ditancapkan pada board Arduino. Misalnya shield GPS, Ethernet, SD Card, dll.

2.2.5 Sensor Tegangan

Sensor tegangan digunakan sebuah transformator step-down hubungan Δ/Y dan resistor pembagi tegangan dipasang secara paralel antara phasa dengan netral. Sensor ini mendeteksi level tegangan pada Line Generator dan Jala-jala PLN dan akan dibaca oleh Arduino mega.

2.2.6 Sensor Frekuensi

Sensor frekuensi ini menggunakan *typical application* dari IC LM2907. Sensor ini akan mendeteksi setiap perubahan frekuensi pada generator. Frekuensi yang masuk ke sensor akan dikonversikan ke dalam bentuk tegangan dc, yang dipakai sebagai input pada ADC arduino.

BAB III
METODE PENELITIAN

3.1 Metodologi Penelitian

Penelitian ini melakukan perancangan *interface* untuk *Visualisasi* proses sinkronisasi Generator Sinkron dengan Jala-jala PLN menggunakan *Mikrokontroller AT Mega 2560*. Metode penelitian meliputi: lokasi dan waktu penelitian, alat dan bahan penelitian, perancangan sistem, perancangan perangkat keras dan perangkat lunak serta diagram alir penelitian.

3.2 Lokasi dan Waktu Penelitian

Penelitian ini akan dilaksanakan di Laboratorium Sistem Tenaga , Jurusan Teknik Elektro, Fakultas Teknik Universitas Mataram. Waktu pelaksanaan penelitian ini direncanakan selama 5 bulan terhitung mulai bulan April 2016 sampai dengan selesai.

3.3 Alat dan Bahan Penelitian

Alat dan bahan yang dibutuhkan dalam penelitian ini adalah sebagai berikut:

3.3.1 Alat Penelitian

Perangkat keras yang digunakan pada penelitian ini adalah seperangkat komputer tipe Aspire One dengan processor Intel Atom, CPU N450 spec 1.66GHz dengan memori 1 GB. Dengan perangkat lunak sistem operasi Windows 7 Pro 32-bit, untuk simulasi digunakan MATLAB version 7.8.0 (R2011b), dan penyusunan laporan digunakan Microsoft Office 2007.

Selain alat di atas, penelitian ini juga menggunakan peralatan yang terdapat pada Laboratorium Sistem Tenaga Listrik, Teknik Elektro, Fakultas Teknik, Universitas Mataram seperti yang terdapat pada Tabel 3.1. di bawah ini :

Tabel 3.1. Alat Ukur yang Digunakan

No	Kode Alat	Alat Ukur	Jenis	Jumlah	Ket
1	DL2109T15	Voltmeter	Analog	1	-
2	LB 72727	Double Frekuensi Meter	Analog	1	-
4	LB73162	Synchronizing Indicator	Analog	1	-
3	Sunwa CD800a	Multimeter	Digital	1	-

Pengambilan data tegangan dan frekuensi menggunakan motor DC yang dikopel dengan generator sinkron. Peralatan-peralatan yang digunakan untuk pengambilan data tersebut ditabelkan pada Tabel 3.2.

Tabel 3.2. Peralatan-peralatan yang digunakan untuk pengambilan data

No	Kode Alat	Nama Alat	Jumlah	Keterangan
1	LB 725 77	DC power Supply	1	-
2	Tipe 73191	Motor DC shunt 0.3 kW	2	-
3	Tipe 73236	Generator Sinkron 0.3 kW	2	-
4	DL 100161	Power Supply 3 phase	2	-

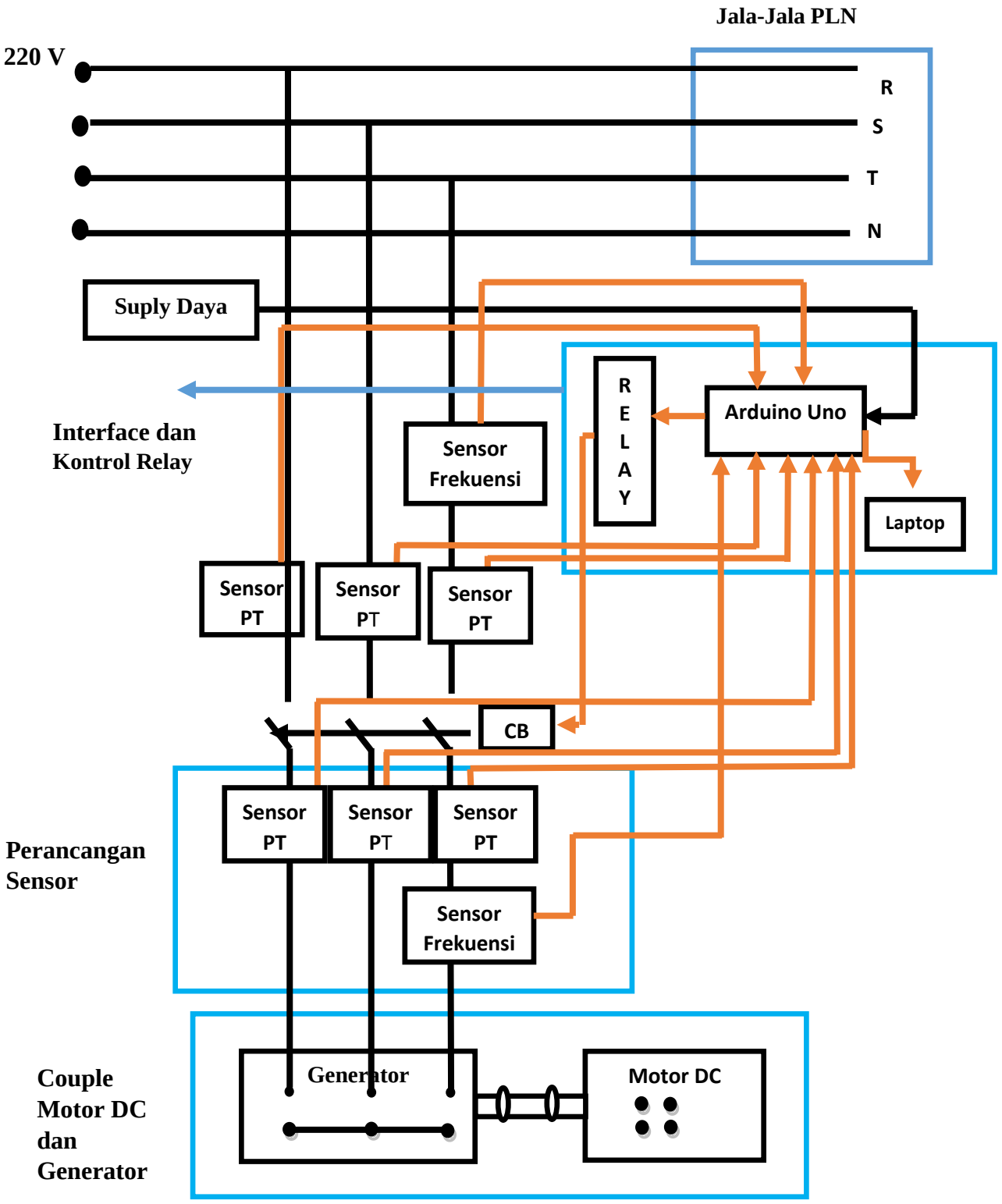
3.3.2 Bahan Penelitian

Bahan-bahan yang dibutuhkan pada penelitian ini adalah mikrokontroller ATMega 2560 dalam minimum sistem Arduino, komponen elektronika daya, konektor, sensor tegangan, sensor frekuensi dan relay optocoupler.

3.4 Perancangan Sistem

Diagram blok perencanaan perancangan *rancang* bangun *interface* untuk *visualisasi* sinkronisasi Generator Sinkron dengan Jala-jala PLN berbasis *mikrokontroller ATmega 2560*. Perancangan ini nantinya akan digunakan untuk melihat sinkronisasi antara generator sinkron dengan jala-jala PLN.

Pada Gambar 3.1 di bawah ini akan diperlihatkan mengenai blok diagram perancangan alat yang digunakan pada generator sinkron dengan Jala-jala PLN.



Gambar 3.1 Diagram blok sistem *sinkronisasi* generator sinkron dengan jala-jala *PLN*

Keterangan :

1. Saluran informasi **:**
2. Saluran Daya :

Cara kerja rangkaian adalah sebagai berikut :

Data-data tegangan dan frekuensi awalnya akan dideteksi oleh sensor tegangan dan sensor frekuensi. Sensor tegangan akan membaca masukan tegangan dengan range dari 0-5 Volt DC. Begitu juga dengan sensor frekuensi menggunakan IC LM 2907 yang mengkonversi frekuensi sebagai inputan menjadi nilai tegangan dengan nilai 0 – 5 Volt.

Kemudian data akan dibaca pada arduino, dengan komunikasi serial data akan dikirim ke Matlab untuk diolah dan ditampilkan. Setelah frekuensi,tegangan,dan sudut phasa sama antara generator dan jala-jala PLN maka arduino akan memberi perintah pada penggerak relay untuk kontaktor menutup sehingga akan terhubung antara generator sinkron dan jala-jala PLN.

3.5 Perancangan Masing-Masing Blok

3.5.1 Rancangan Blok untuk Couple Motor DC dan Generator Sinkron

Pada blok ini terdapat dua alat yaitu Motor DC 0.3 kW dengan Generator Sinkron 0.3 kW 3 fasa yang terdapat pada Laboratorium Sistem Tenaga Listrik dengan cara di couple untuk menghasilkan listrik 3 fasa.

Gambar 3.2. Couple motor dc dan generator sinkron

3.5.2. Perancangan Blok untuk Sensor Tegangan dan Frekuensi

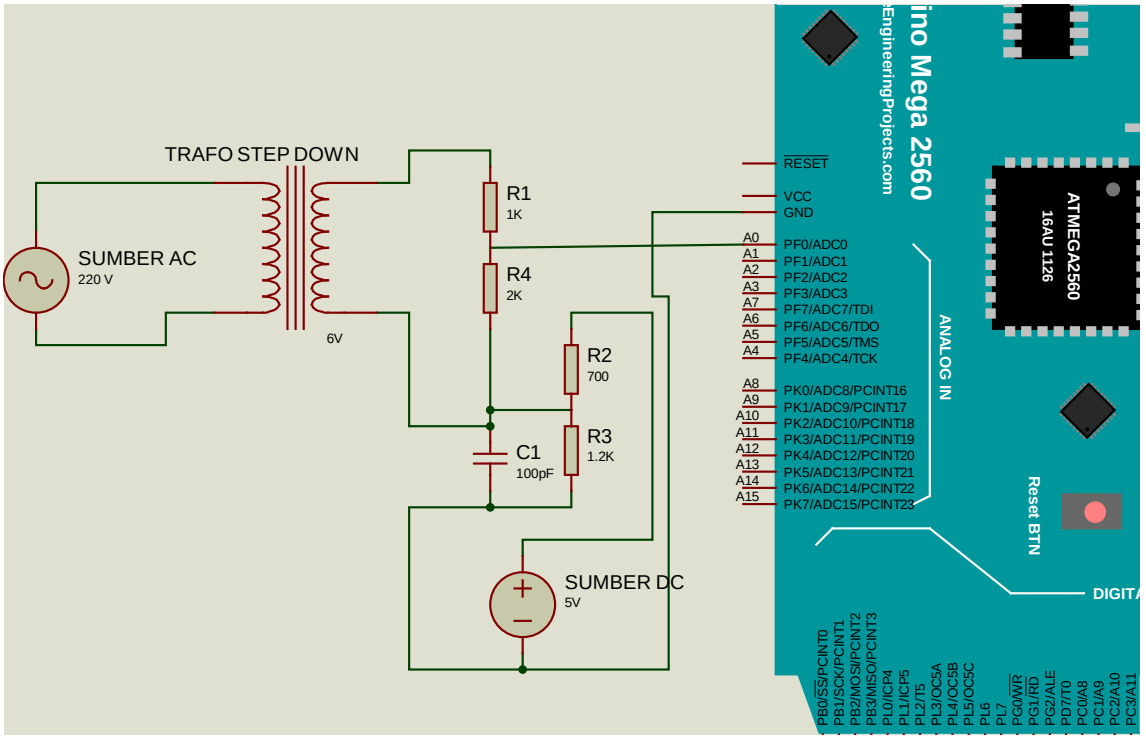
Pada blok ini terdapat dua jenis sensor yang akan dirancang yaitu sensor tegangan dan sensor frekuensi. Untuk sensor tegangan akan dibuat sebanyak enam sensor dan dua sensor frekuensi . Berikut perancangan kedua sensor tersebut :

a. Rangkaian Sensor Tegangan

Sensor tegangan yang digunakan berupa trafo tegangan step down yang rangkaian primernya merupakan sumber tegangan utama dari beban yang akan diukur dan output rangkaian sekundernya akan masuk ke rangkaian pembagi tegangan melalui resistor R1 dan R2. Rangkaian ini akan membagi tegangan sekunder trafo ($V_{sekunder}$) dengan Persamaan (3-1). Tegangan pada R2 (V_{R2}) akan masuk dan dibaca oleh analog input (A1) dari mikrokontroler. Output tegangan ini masih mengandung komponen negatif dari gelombang AC, sehingga perlu ditambahkan offset menggunakan tegangan 5V DC dari sumber luar. Sebelum ditambahkan, tegangan DC ini akan melalui resistor pembagi tegangan R3 dan R4 yang bernilai sama sehingga DC offset yang ditambahkan sebesar 2,5 V. Kapasitor C1 berfungsi sebagai kapasitor bypass untuk memfilter *ripple* dari 5V DC agar nilai hasil baca analognya tetap stabil.

Perhitungan untuk rangkaian pembagi tegangan adalah:

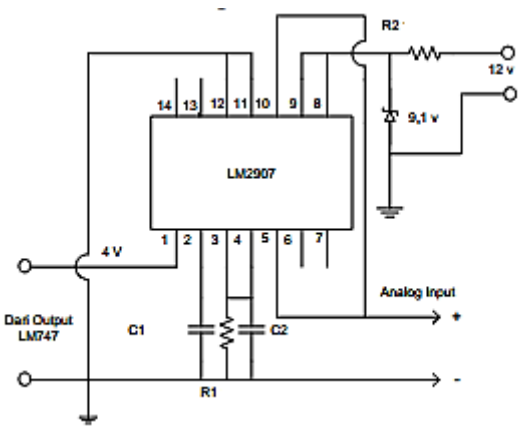
$$V_{R2} = \frac{R_2}{R_1+R_2} \times V_{sekunder} \quad \text{.....} \quad (3\text{-}1)$$



Gambar 3.3 Rangkaian sensor tegangan

b. Rangkaian Sensor Frekuensi

Sensor frekuensi ini menggunakan IC LM2907. Sensor ini akan mendeteksi setiap perubahan frekuensi pada generator. Frekuensi yang masuk ke sensor akan dikonversikan ke dalam bentuk tegangan dc, yang kemudian akan masuk ke ADC. Gambar rangkaian sensor frekuensi yang akan dirancang:



Gambar 3.4 Sensor frekuensi

Tegangan output ditentukan oleh persamaan sebagai berikut :

Vout = f_{in} x Vcc x R1 x C1 ……(3.2)

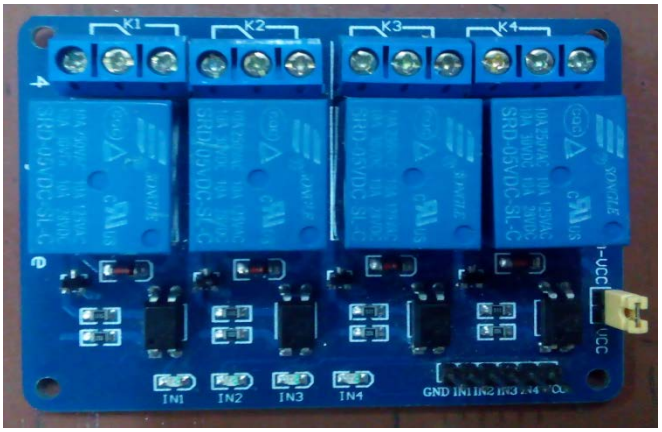
Pada sensor ini digunakan Vcc sebesar 8 Volt, R1 = 100 Kohm, C1=120 nF rangkaian ini dapat mendeteksi frekuensi samapai dengan kurang lebih 60 Hz. Vin maksimum adalah kurang lebih 4 Volt sehingga untuk mendeteksi frekuensi generator, tegangan generator diturunkan lebih dahulu dengan transformator step-down.

3.5.3 Perancangan Blok Interface dan Kontrol Relay

Pada Blok ini terdapat Relay, Arduino dan Laptop sebagai Interface. Penjelasan masing-masing bagian :

a. Relay

Relay yang digunakan adalah relay optocoupler yang digunakan untuk menggerakkan kontaktor yang terdapat pada Laboratorium Sistem Tenaga Listrik



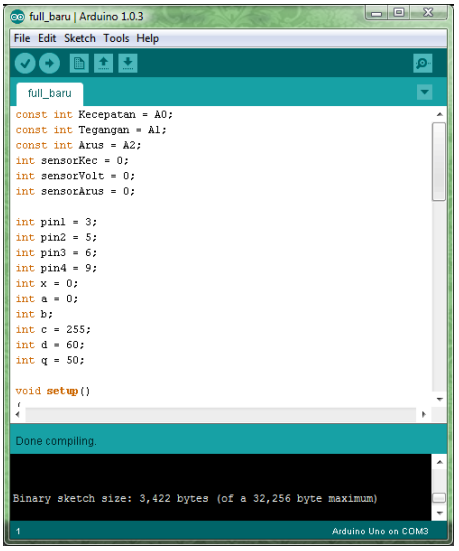
Gambar 3.5 Relay optocoupler 4 channel

b. Pemrograman Untuk Mikrokontroller ATMega 2560 (Arduino)

Mikrokontroller ATMega2560 yang terdapat di dalam minimum sistem Arduino menggunakan Bahasa Pemrograman Processing. Processing adalah bahasa pemrograman tingkat tinggi yang dialeknya sangat mirip dengan C++ dan Java. Software minimum sistem Arduino meliputi IDE untuk menulis program, *driver* untuk koneksi dengan computer, contoh program dan *library* untuk pengembangan program.

IDE Arduino adalah software yang ditulis dengan menggunakan Java. IDE Arduino terdiri dari: *Editor* program, sebuah window yang digunkan untuk menulis dan mengedit program dalam bahasa Processing. *Compiler*, sebuah modul yang mengubah kode program (bahasa Processing) menjadi kode biner.

Sebuah mikrokontroller tidak akan bisa memahami bahasa Processing. Yang bisa dipahami oleh mikrokontroller adalah kode biner. Itulah sebabnya *compiler* diperlukan dalam hal ini. *Uploader*, sebuah modul yang memuat kode biner dari komputer ke dalam memori di dalam papan Arduino.



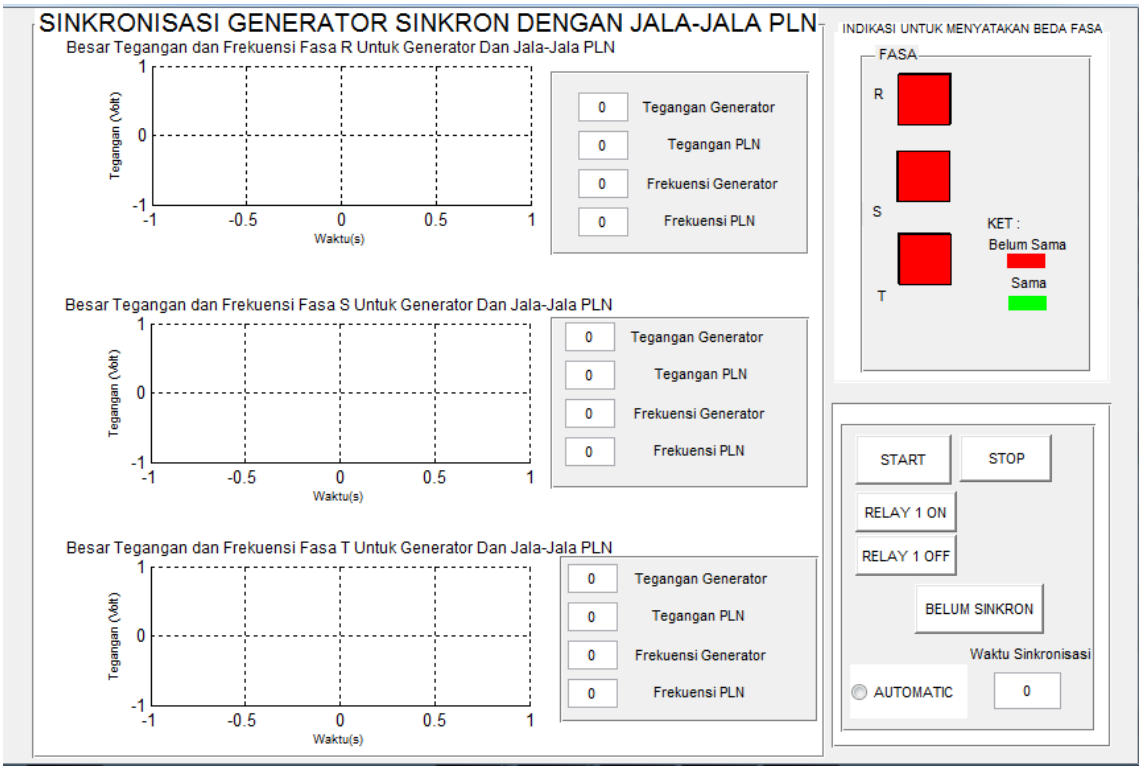
Gambar 3.6 Tampilan IDE arduino
(Djuandi, 2011)

c. Interface Hasil Pembacaan

Hasil pembacaan sensor akan ditampilkan pada komputer yang akan dibuat menggunakan *software* MATLAB version 7.8.0 (R2011a), dimana pada perancangan *software* antar muka ini akan dibagi menjadi beberapa tab yaitu:

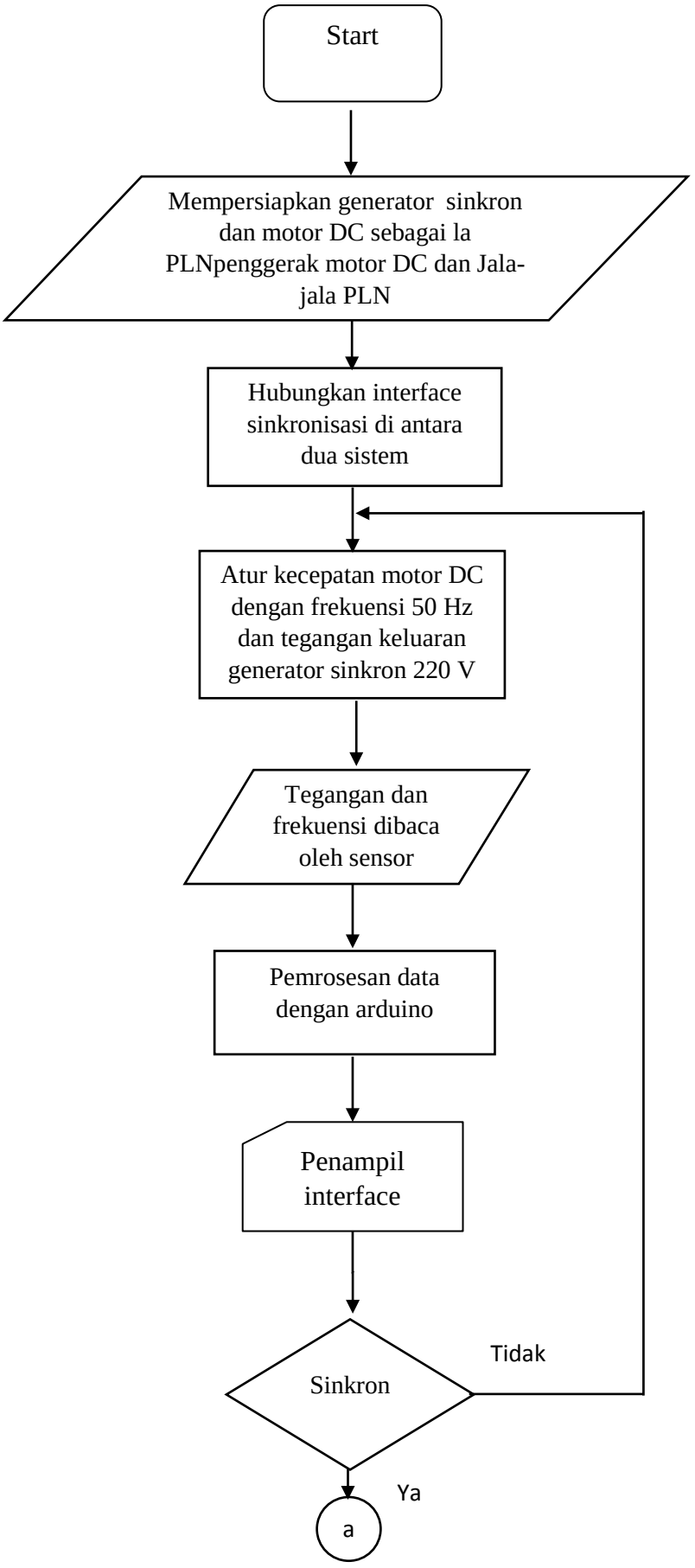
1. Tab pertama akan dirancang untuk menampilkan hasil keluaran dari sensor pada fase R generator sinkron yang akan menampilkan besar tegangan, frekuensi dan sudut fasa
2. Tab kedua akan dirancang untuk menampilkan hasil keluaran dari sensor pada fase S generator sinkron yang akan menampilkan besar tegangan frekuensi, dan sudut fasa.
3. Tab kedua akan dirancang untuk menampilkan hasil keluaran dari sensor pada fase T generator sinkron yang akan menampilkan besar tegangan frekuensi, dan sudut fasa

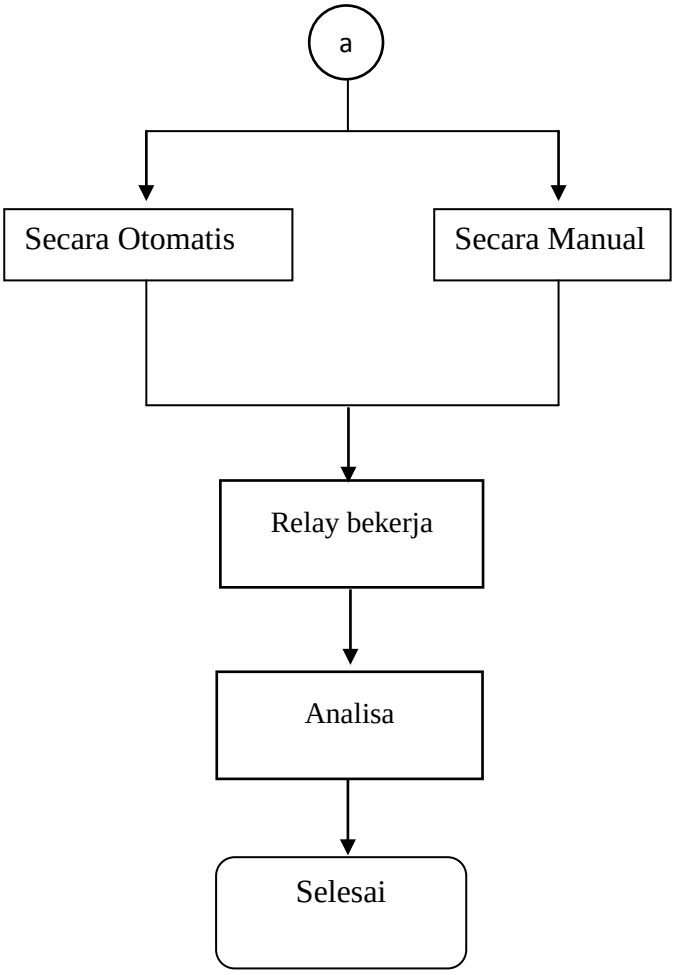
Interface hasil dari MATLAB dapat dilihat pada gambar 3.7 di bawah ini :



Gambar 3.7 Tampilan pada Matlab

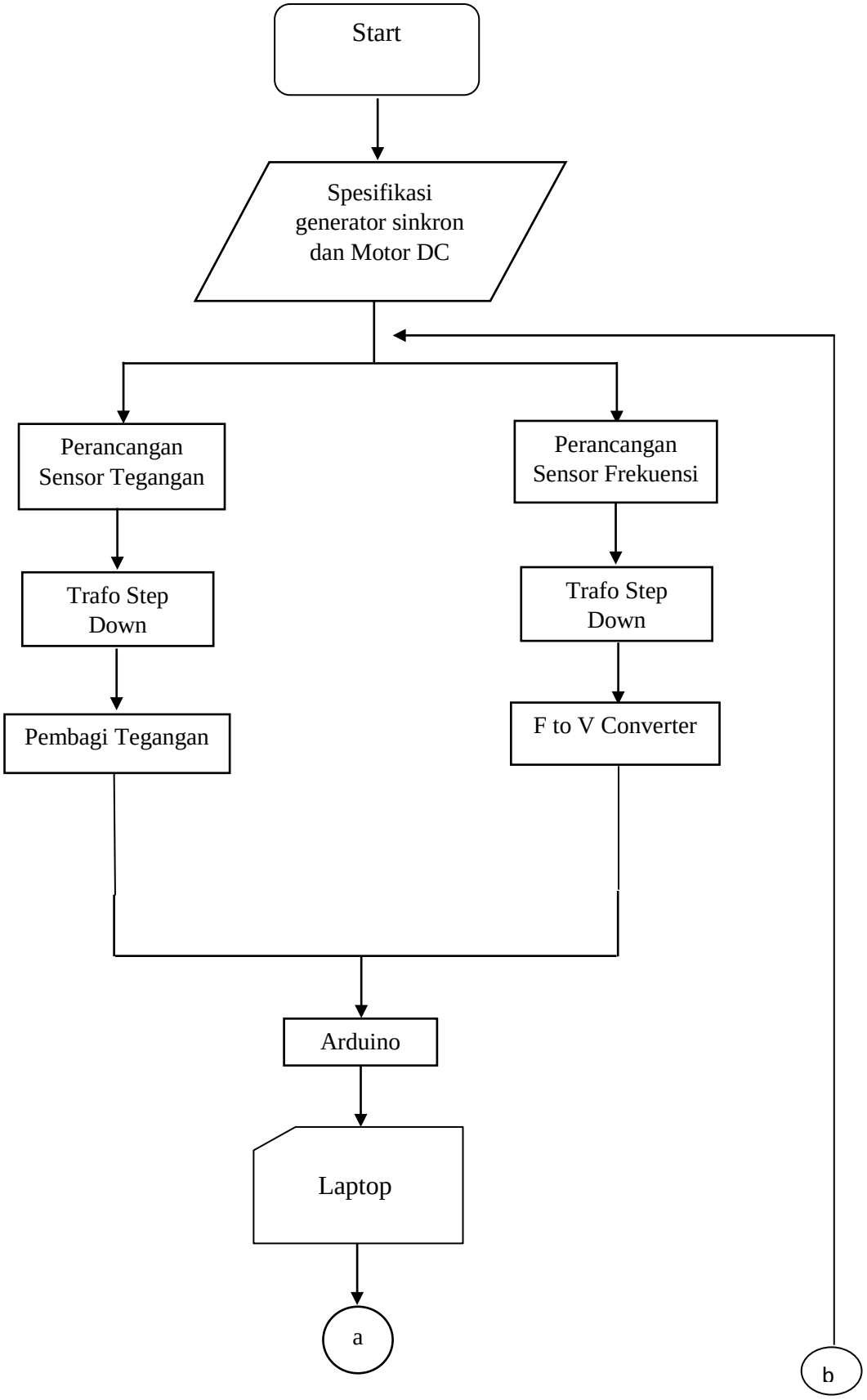
3.6 Diagram Alir Penelitian

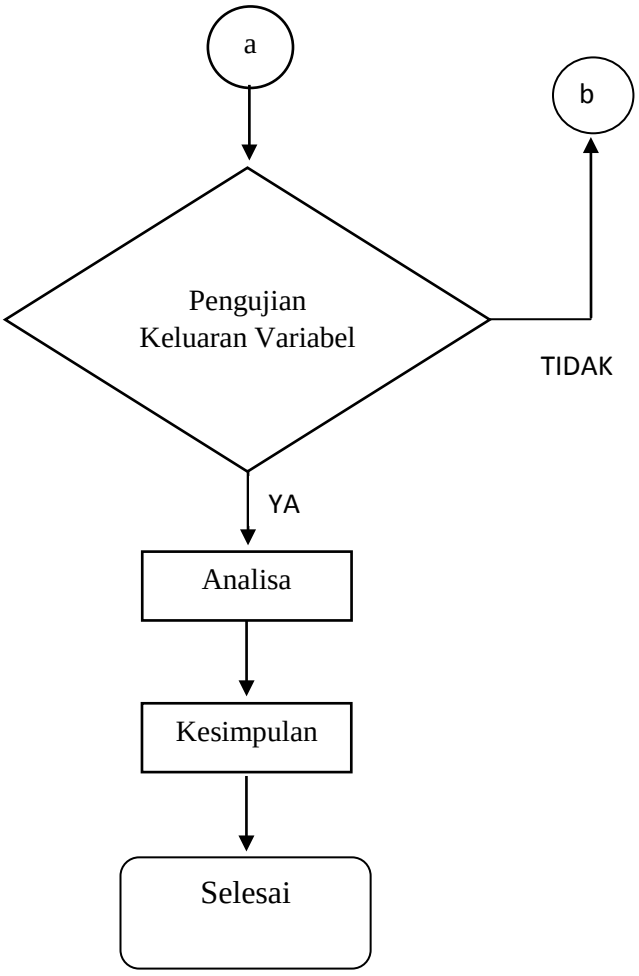




Gambar 3.8 Diagram Alir Penelitian

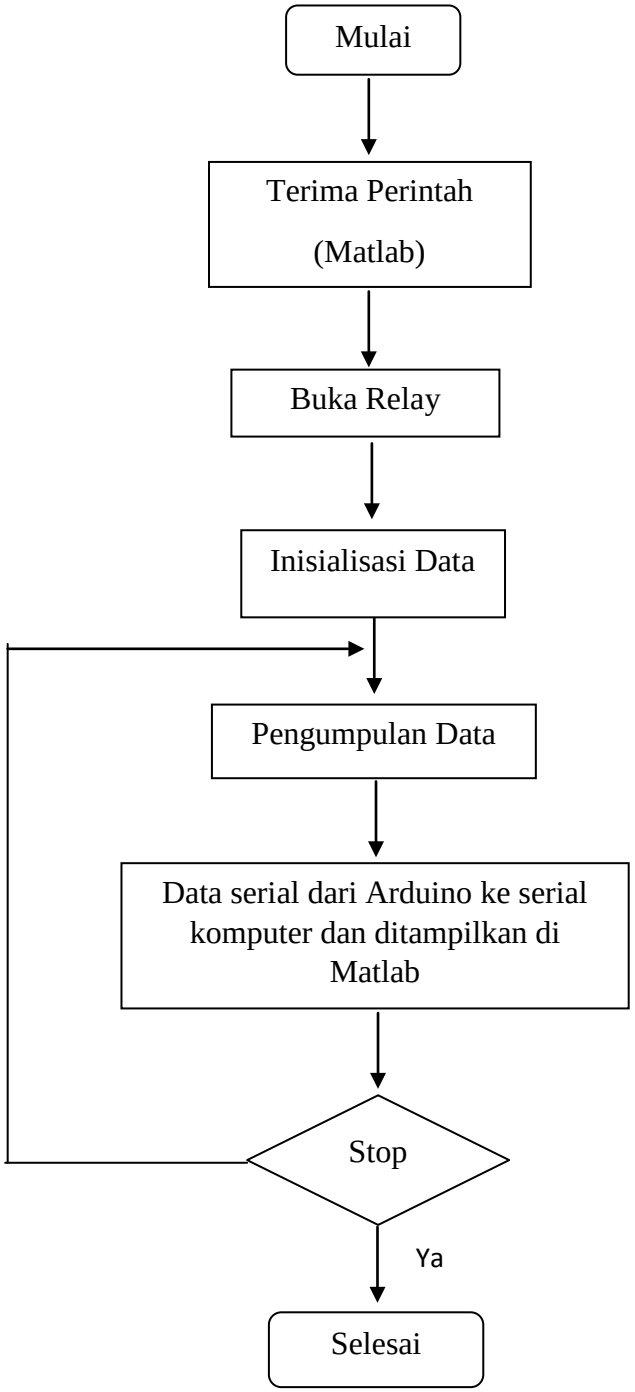
3.7 Diagram Alir Perancangan Alat





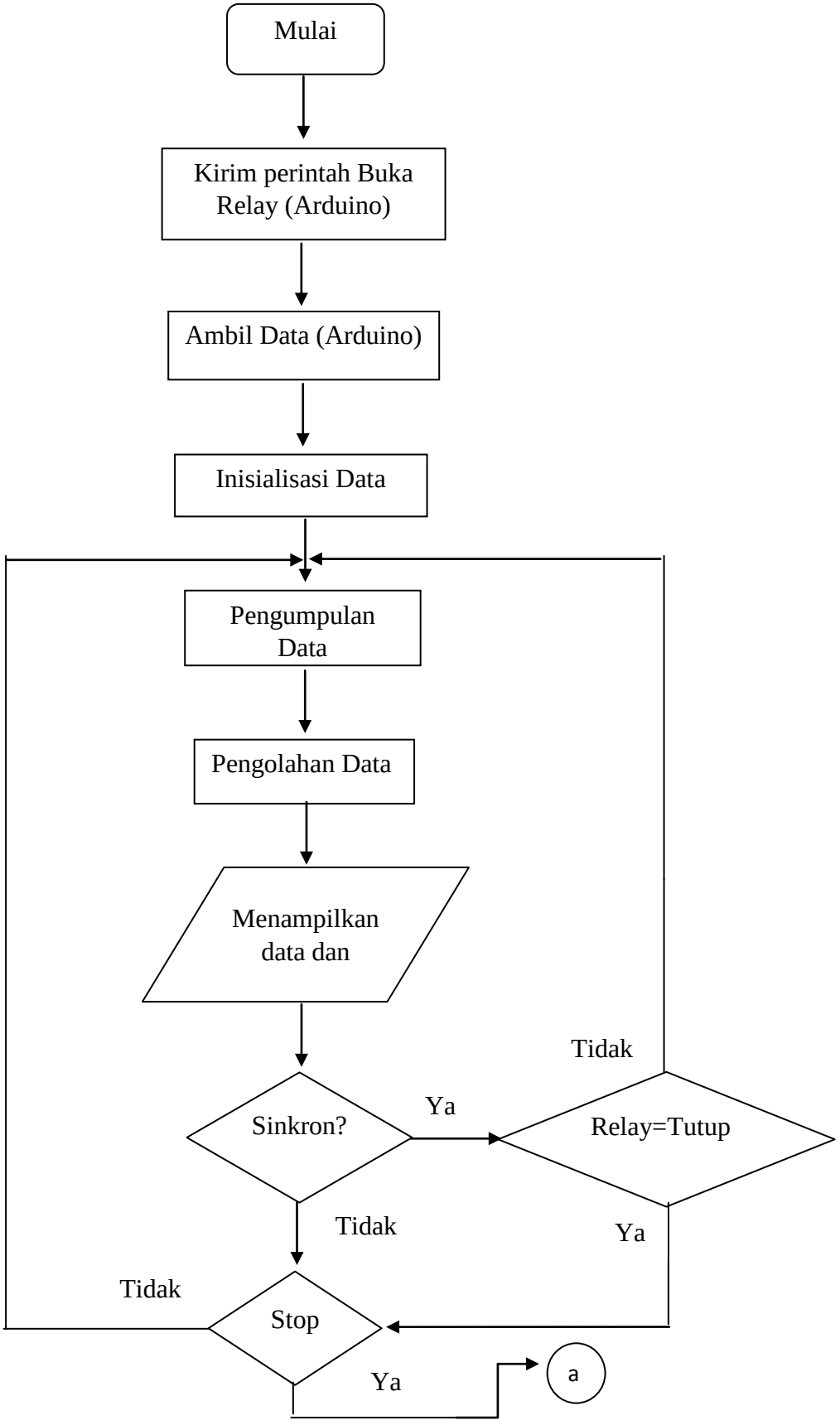
Gambar 3.9 Diagram Alir Perancangan Alat

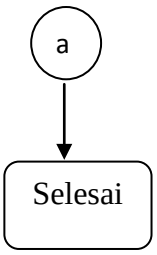
3.8. Diagram Alir Program pada Arduino



Gambar 3.10. Diagram Alir Program Pada Arduino

3.9. Diagram Alir Program Pada Matlab





Gambar 3.11. Diagram Alir Program Pada Matlab

BAB IV
PEMBAHASAN

4. Pengujian Sistem

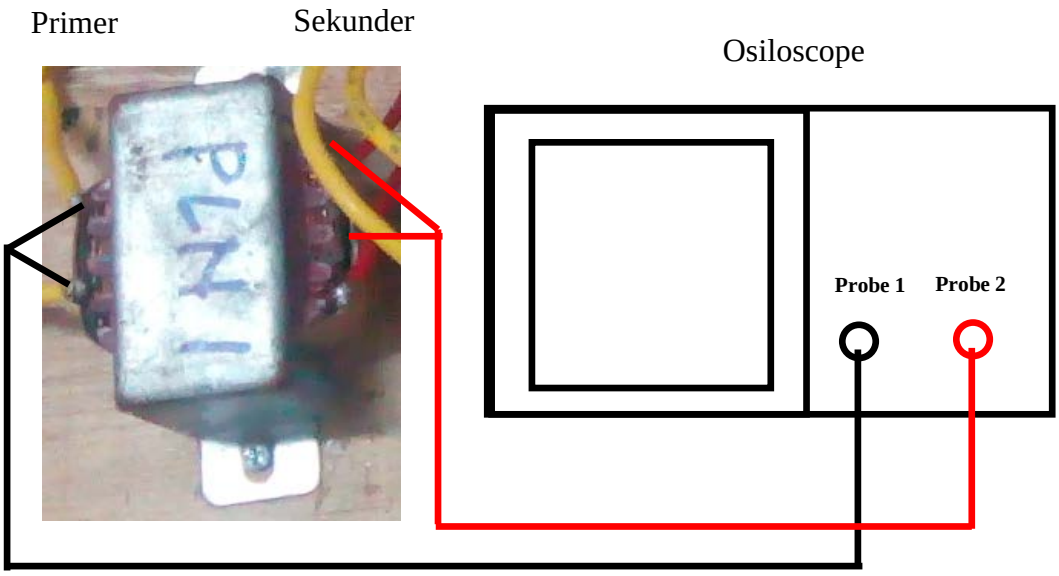
Pada penelitian ini, dilakukan perancangan dan pembuatan interface untuk visualisasi proses sinronisasi dengan tujuan untuk melihat nilai frekuensi,tegangan dan sudut fase saat terjadinya sinkronisasi, kemudian setelah sinkron akan dilakukan penyambungan secara otomatis dan manual antara generator sinkron dengan jala-jala PLN.

Untuk mengetahui kinerja dan kemampuan pada sistem perancangan, maka perlu dilakukan beberapa pengujian alat meliputi pengujian hardware (perangkat keras) dan pengujian software (perangkat lunak). Untuk memudahkan penulis dalam melakukan proses pengujian bahan, maka dilakukan pengujian secara terpisah dan secara keseluruhan. Adapun proses pengujian yang dilakukan meliputi:

1. Pengujian Trafo Step Down
2. Pengujian Rangkaian Sensor Tegangan
3. Pengujian Rangkaian Sensor Frekuensi
4. Pembuatan Antar Muka (*Interface*) pada PC Laptop
5. Pengukuran dan Pengujian Keseluruhan Sistem

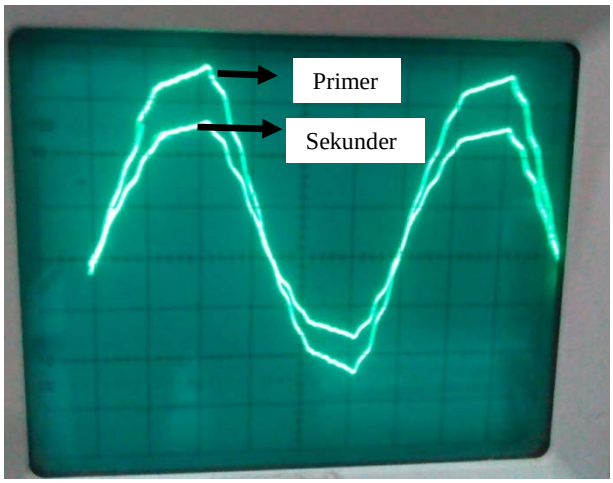
4.1. Pengujian Trafo Step Down

Sebelum melakukan pengujian terhadap sensor tegangan dan sensor frekuensi maka terlebih dahulu melakukan pengujian pada trafo step down. Pengujian dilakukan pada keenam trafo dengan membandingkan pada bagian primer dan sekunder untuk melihat sudut fase antara bagian primer dan sekunder. Pengujian dilakukan menggunakan osiloscope untuk melihat gelombang pada bagian primer dan sekundernya dengan menggunakan sumber PLN dan Generator. Berikut rangkaian pengujian trafo :

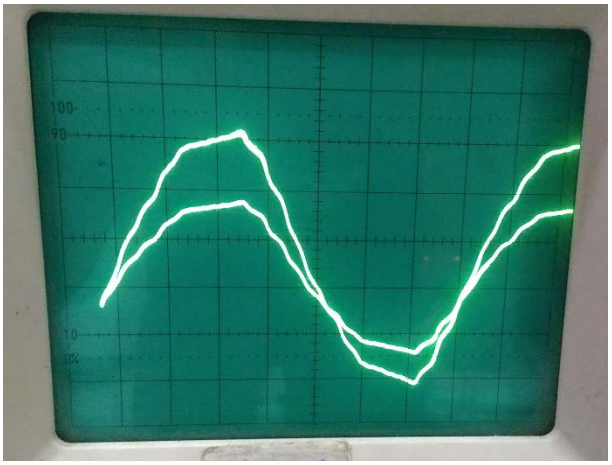


Gambar 4.1. Rangkaian pengujian trafo step down

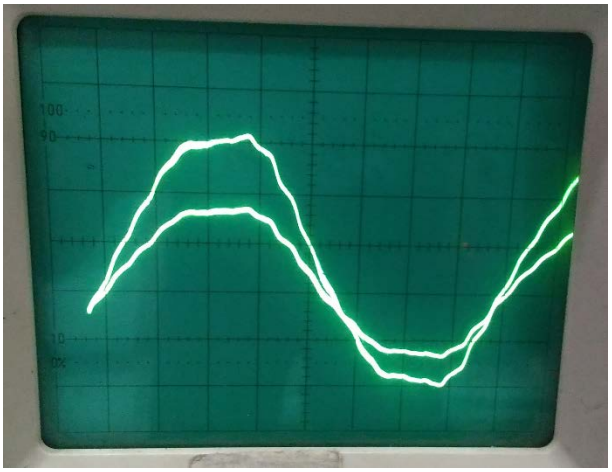
Pada Gambar 4.2- 4.4 dapat dilihat gelombang keluaran antara bagian primer dan sekunder dengan sumber Generator.



Gambar 4.2 Gelombang primer dan sekunder trafo 1 sumber generator



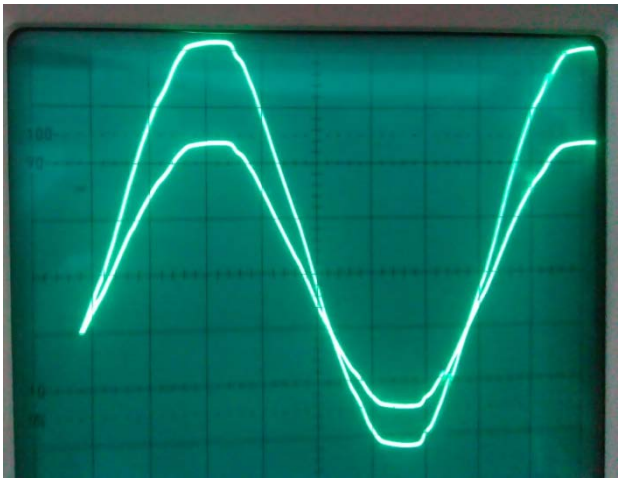
Gambar 4.3 Gelombang primer dan sekunder trafo 2 sumber generator



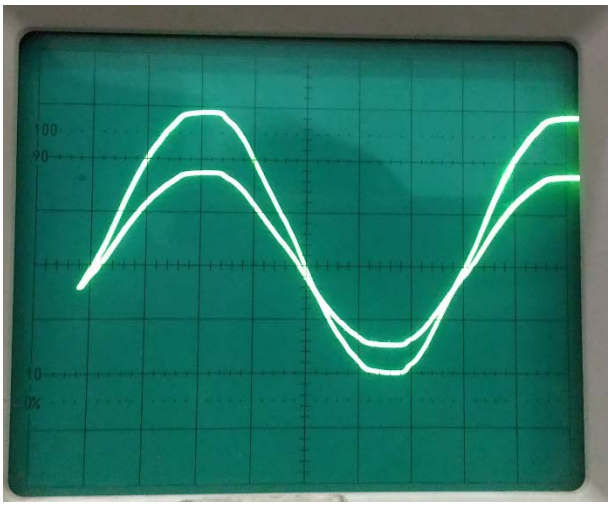
Gambar 4.4 Gelombang primer dan sekunder trafo 3 sumber generator

Pengujian gelombang keluaran pada trafo dengan sumber Generator dilakukan dengan memvariasikan frekuensi generator sehingga dapat dilihat walaupun frekuensi yang diberikan bervariasi besar sudut fase antara keluaran primer dan sekunder dengan sumber generator memiliki fase yang sama.

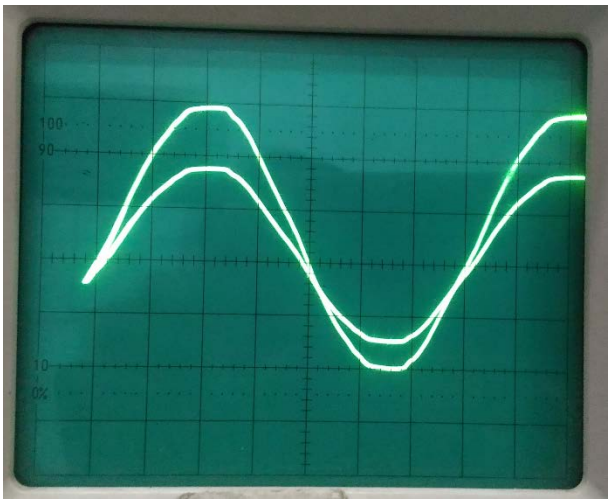
Selanjutnya dilakukan pengujian untuk bagian primer dan sekunder pada ke 6 trafo dengan sumber jala-jala PLN.



Gambar 4.5 Gelombang primer dan sekunder trafo 4 sumber PLN



Gambar 4.6 Gelombang primer dan sekunder trafo 5 sumber PLN

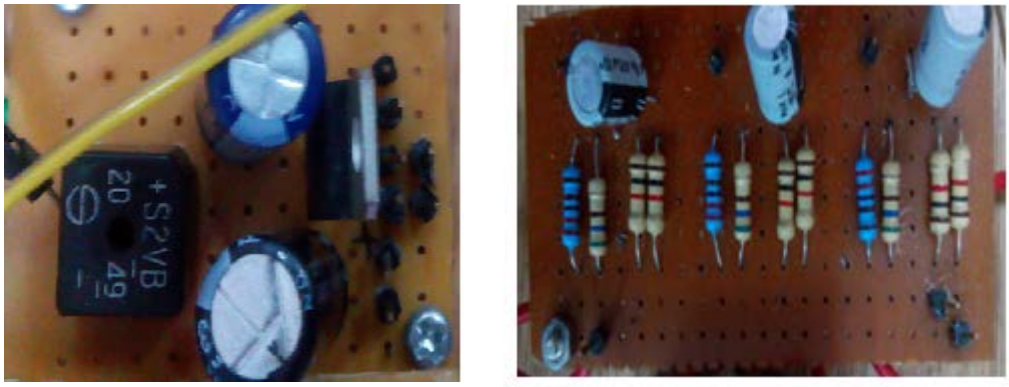


Gambar 4.7 Gelombang primer dan sekunder trafo 6 sumber PLN

Pada Gambar 4.4 - 4.6 menunjukkan sudut fase yang sama pada bagian primer dan sekunder dengan sumber listrik PLN.

4.2 Pengujian Rangkaian Sensor Tegangan

Pengujian sensor tegangan ini bertujuan untuk mendapatkan nilai keluaran sensor yang presisi. Rangkaian sensor tegangan pada penelitian ini menggunakan transformator 1 phasa dengan input tegangan primer 220 Volt AC dan tegangan output sekunder 6 Volt AC yang kemudian akan dilakukan pembagian tegangan, sebelum masuk ke arduino untuk dibaca perlu dilakukan penggeseran gelombang sehingga yang masuk ke arduino tegangan dalam level DC. Untuk membuat tegangan dalam level AC menjadi DC diperlukan sumber DC 5 Volt yang berasal dari sumber luar kemudian tegangan DC diberikan ke rangkaian .



DC 5 V

Gambar 4.8 Sensor Tegangan

Selajutnya dilakukan pengukuran dari keluaran sensor dengan nilai tegangan input yang berbeda-beda (20-220 V) dengan output dari sensor tegangan dengan maksimal tegangan sebesar 5V DC.

Untuk menguji sensor tegangan pada generator fasa R, pada penelitian ini digunakan power supplai yang ada di Laboratorium Sistem Tenaga agar tegangan yang masuk ke dalam sensor dapat di ubah-ubah sehingga hasilnya semakin besar tegangan input sensor maka tegangan output sensor juga akan semakin besar. Dari hasil pengujian tersebut kemudian sebagai contoh digunakan data Tabel 4.1. untuk menentukan persamaan polinomial yang memiliki persentase error yang kecil.

Dari persamaan polinomial yang memiliki persentase error kecil akan digunakan untuk kalibrasi sensor tegangan baik pada PLN maupun pada Generator. Pendekatan linear (orde 1,orde 2 dan orde 3) ini menggunakan fungsi ‘polyfit.m’ dari MATLAB yaitu sebagai berikut:

4.2.1 Pengujian Sensor Tegangan Fasa R pada Sumber Listrik PLN

Pengujian sensor tegangan pada fasa R pada PLN dapat dilakukan pengujian setelah didapatkan hasil pengukuran antara tegangan input AC yang diukur menggunakan Voltmeter analog yang besar tegangan inputan sampai 220 volt dengan tegangan DC yang terbaca pada interface dengan besar tegangan maksimal 5 V DC. Berikut-berikut data hasil pengukuran dapat dilihat pada tabel 4.1 di bawah ini :

Tabel 4.1 Nilai tegangan sensor pada PLN fasa R

No	Tegangan Input (Vac)	Tegangan Output (Vdc)
1	20	2.69
2	40	2.87
3	60	3.07
4	80	3.26
5	100	3.44
6	120	3.61
7	140	3.81
8	160	4
9	180	4.2
10	200	4.4
11	220	4.6

Dari Tabel 4.1 di atas dapat dibuat pendekatan polinomial orde 1, orde 2 dan orde 3 menggunakan fungsi polyfit yang ada di Matlab.

a. Pendekatan Polinomial Orde 1

```
ac = [20  40 60 80 100 120 140 160 180 200 220];
dc = [2.69 2.87 3.07 3.26 3.44 3.61 3.81 4 4.2 4.4 4.6];
p = polyfit(ac,dc,1)
p = 0.0095    2.4913 (dengan 0.0095 adalah nilai a dan 2.4913 adalah
nilai b)
```

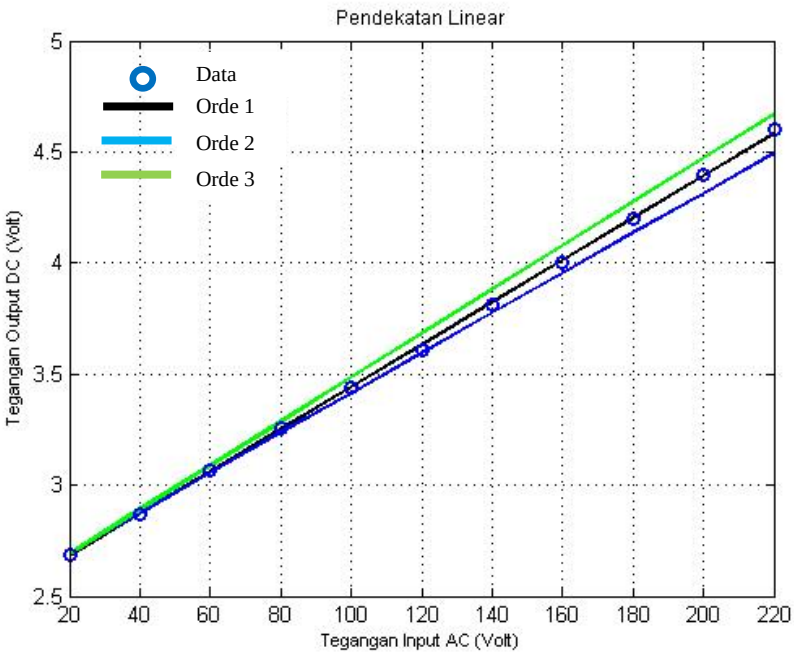
b. Pendekatan Polinomial Orde 2

```
ac = [20  40 60 80 100 120 140 160 180 200 220];
dc = [2.69 2.87 3.07 3.26 3.44 3.61 3.81 4 4.2 4.4 4.6];
p = polyfit(ac,dc,2)
p = 0.0000    0.0090    2.5152 (dengan 0.0000 adalah nilai a, 0.0090
adalah nilai b dan 2.5152 adalah nilai c)
```

c. Pendekatan Polinomial Orde 3

```
ac = [20  40 60 80 100 120 140 160 180 200 220];
dc = [2.69 2.87 3.07 3.26 3.44 3.61 3.81 4 4.2 4.4 4.6];
p = polyfit(ac,dc,2)
p = 0.0000   -0.0000    0.0099    2.4927 (dengan 0.0000 adalah nilai
a, -0.0000 adalah nilai b, 0.0099 adalah nilai c dan 2.4927 adalah
nilai d)
```

Dari ketiga orde di atas dapat dibuat grafik persamaan polinomial untuk dapat melihat persamaan orde yang lebih bagus dan memiliki error yang kecil.



Gambar 4.9. Pendekatan polinomial

Dari Gambar 4.9 di atas dapat diketahui persentase error pada setiap orde sehingga pendekatan orde mana yang lebih bagus dapat digunakan pada kalibrasi sensor tegangan. Berikut salah satu perhitungan untuk mencari persentase error persamaan polinomial orde 1

Tabel 4.2 Perhitungan persentase rerarata error Orde 1

a	X	b	Orde 1 Y = aX +b	Pembanding (Tegangan DC)	Error (orde 1 – Pembanding)
0.0095	20	2.4913	2.6813	2.69	0.32
0.0095	40	2.4913	2.8713	2.87	0.045
0.0095	60	2.4913	3.0613	3.07	0.283
0.0095	80	2.4913	3.2513	3.26	0.266
0.0095	100	2.4913	3.4413	3.44	0.03
0.0095	120	2.4913	3.6313	3.61	0.59
0.0095	140	2.4913	3.8213	3.81	0.29
0.0095	160	2.4913	4.0113	4	0.275
0.0095	180	2.4913	4.2013	4.2	0.03
0.0095	200	2.4913	4.3913	4.4	0.19
0.0095	220	2.4913	4.5813	4.6	0.406
Rerata error orde 1					**0.24**

Selanjutnya untuk rerata persentase error selanjutnya dapat dilihat pada Tabel 4.3 dibawah ni.

Tabel 4.3. Persentase error masing-masing persamaan linear

No	Persamaan Polinomial	Error (%)
1	Orde 1	0.24
2	Orde 2	0.9561
3	Orde 3	1.07

Dari Tabel 4.3 dapat menunjukkan besar persentase error masing-masing persamaan polinomial dengan persentase error terkecil pada orde 1. Oleh karena itu maka pendekatan yang digunakan untuk kalibrasi sensor tegangan menggunakan persamaan polinomial orde 1.

Hasil pendekatan linear orde 1 mengunakan funsgi ‘polyfit.m’, didapatkan persamaan hubungan tegangan input sensor terhadap tegangan output sensor, yaitu Persamaan 4.1.

$$V_{sensor\ tegangan} = 0.0095 * Tegangan_{sensor} + 2.4913\ Volt \qquad (4.1)$$

Persamaan 4.1 dapat dibuat persamaan untuk melihat hasil pengukuran yang terbaca pada interface dengan output tegangan AC (0-220 Volt) dengan yang terukur pada alat ukur analog (0-220 V) melalui Persamaan 4.2 berikut ini :

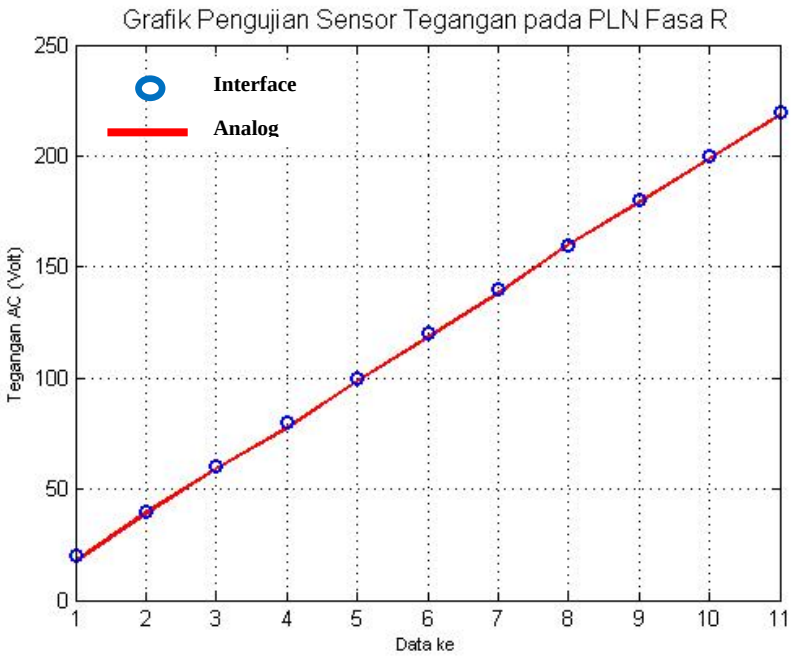
$$Tegangan\ PLN1 = \frac{\left(\frac{b(1)*5}{1023} - 2.4913\right)}{0.0095}\ Volt \quad (4.2)$$

Dari persamaan 4.2 di atas didapatkan hasil perbandingan nilai tegangan antara yang terbaca pada interface dengan alat ukur analog pada Tabel 4.4 :

Tabel 4.4. Tegangan pada Alat Ukur Analog dan Interface

Data ke	**Tegangan Menggunakan Alat Ukur Analog (VAC)**	**Tegangan Terbaca di Interface (VAC)**	**Error (%)**
1	20	17,4	13
2	40	39,02	2.45
3	60	58,94	1.76
4	80	77,386	3.3
5	100	98,408	1.6
6	120	118,37	1.41
7	140	137,82	1.52
8	160	159,89	0.06
9	180	178,81	0.66
10	200	198,78	0.61
11	220	218,23	0.8
Rerata			**2.47**

Pada Tabel 4.4 dapat dibuat grafik pengujian sensor tegangan fasa R pada PLN dengan membandingkan antara alat ukur analog dengan yang terbaca pada interface.



Gambar 4.10 Grafik Pengujian Sensor Tegangan Pada PLN Fasa R

4.2.2 Pengujian Sensor Tegangan Fasa S pada PLN

Pengujian sensor tegangan fasa S pada PLN dilakukan setelah mengetahui output tegangan DC yang terbaca pada sensor dengan tegangan input AC yang diberikan. Tabel 4.5 memperlihatkan hasil baca sensor setelah diberikan tegangan input AC.

Tabel 4.5 Nilai tegangan sensor fasa S PLN

No	Tegangan Input (Vac)	Tegangan Output (Vdc)
1	20	2.75
2	40	2.97
3	60	3.19
4	80	3.42
5	100	3.63
6	120	3.83
7	140	4.05
8	160	4.26
9	180	4.4
10	200	4.6
11	220	4.8

Data Tabel 4.5 dapat dibuat persamaan linear orde 1 dengan menguji nilai inputan AC dengan inputan dari 20-220 V dengan ouput DC dengan tegangan maksimal 5 V. Berikut script untuk mengetahui persamaan polinomial orde 1 :

```
ac = [20  40 60 80 100 120 140 160 180 200 220];
dc = [2.75 2.97 3.19 3.42 3.63 3.83 4.05 4.26 4.4 4.6 4.8];
p = polyfit(ac,dc,1)
p = 0.0102   2.5818 (Dengan 0.0102 sebagai nilai a dan 2.5818 sebagai
nila b)
```

Hasil pendekatan linear orde 1 menggunakan funsgi ‘polyfit.m’, didapatkan persamaan hubungan tegangan input sensor terhadap tegangan output sensor, yaitu Persamaan 4.3.

$$V_{sensor\ tegangan} = 0.0102 * Tegangan_{sensor} + 2.5818\ Volt \qquad (4.3)$$

Dari persamaan 4.3 dapat dibuat persamaan untuk melihat hasil pengukuran yang terbaca pada interface dengan output tegangan AC (20-220 Volt) dengan yang terukur pada alat ukur analog (20-220 V) melalui persamaan 4.4 di bawah ini :

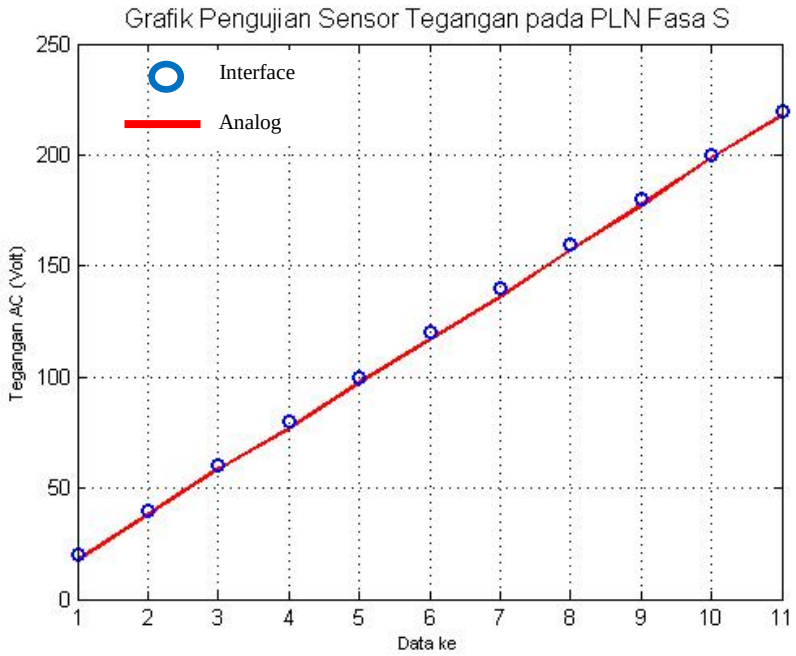
$$Tegangan\ PLN2 = \frac{\left(\frac{b(2)*5}{1023} - 2.5818\right)}{0.0102}\ Volt \qquad (4.4)$$

Dari persamaan 4.4 di atas didapatkan hasil perbandingan nilai tegangan antara yang terbaca pada interface dengan alat ukur analog pada Tabel 4.6 :

Tabel 4.6. Tegangan pada Alat Ukur Analog dan Interface

Data ke	Tegangan Menggunakan Alat Ukur Analog (VAc)	Tegangan Terbaca di Interface (VAc)	Error (%)
1	20	17.2	14
2	40	38,09	4.7
3	60	58,76	2.06
4	80	76,53	4.3
5	100	97,404	2.5
6	120	116,62	2.8
7	140	135,84	2.9
8	160	157,26	1.7
9	180	177,03	1.65
10	200	198,45	0.7
11	220	217,67	1.05
Rerata			**0.03**

Dari Tabel 4.6 dapat dibuat grafik pengujian sensor tegangan fasa S pada PLN dengan membandingkan antara alat ukur analog dengan yang terbaca pada interface.



Gambar 4.11 Grafik Pengujian Sensor Tegangan pada PLN fasa S

4.2.3 Pengujian Sensor Tegangan Fasa T pada PLN

Pengujian sensor tegangan fasa T pada PLN dilakukan setelah mengetahui output tegangan DC yang terbaca pada sensor dengan tegangan input AC yang diberikan. Tabel 4.7 memperlihatkan hasil baca sensor setelah diberikan tegangan input AC.

Tabel 4.7 Nilai tegangan sensor pada PLN fasa T

No	Tegangan Input (Vac)	Tegangan Output (Vdc)
1	20	2.74
2	40	2.96
3	60	3.17
4	80	3.39
5	100	3.6
6	120	3.8
7	140	4.02
8	160	4.23
9	180	4.41
10	200	4.6
11	220	4.81

Dari tabel 4.7 dapat dibuat persamaan polinomial orde 1 dengan mengetahui nilai inputan AC dengan besar tegangan dari 20-220 V dan tegangan output DC dengan maksimal tegangan 5 V DC. Berikut script untuk mengetahui persamaan polinomial orde 1 :

```
ac = [20  40 60 80 100 120 140 160 180 200 220};
dc = [2.74 2.96 3.17 3.39 3.6 3.8 4.02 4.23 4.41 4.6 4.81];
p = polyfit(ac,dc,1)
p = 0.0103   2.5538 (dengan 0.0103 adalah nilai a dan 2.5538 adalah
nilai b)
```

Hasil pendekatan polinomial orde 1 mengunakan funsgi ‘polyfit.m’, didapatkan persamaan hubungan tegangan input sensor terhadap tegangan output sensor, yaitu pada Persamaan 4.5.

$$V_{sensor\ tegangan} = 0.0103 * Tegangan_{sensor} + 2.5538\ Volt \qquad (4.5)$$

Dari persamaan 4.5 dapat dibuat persamaan untuk melihat hasil pengukuran yang terbaca pada interface dengan output tegangan AC (20-220 Volt) dengan yang terukur pada alat ukur analog (20-220 V) melalui persamaan 4.6 di bawah ini :

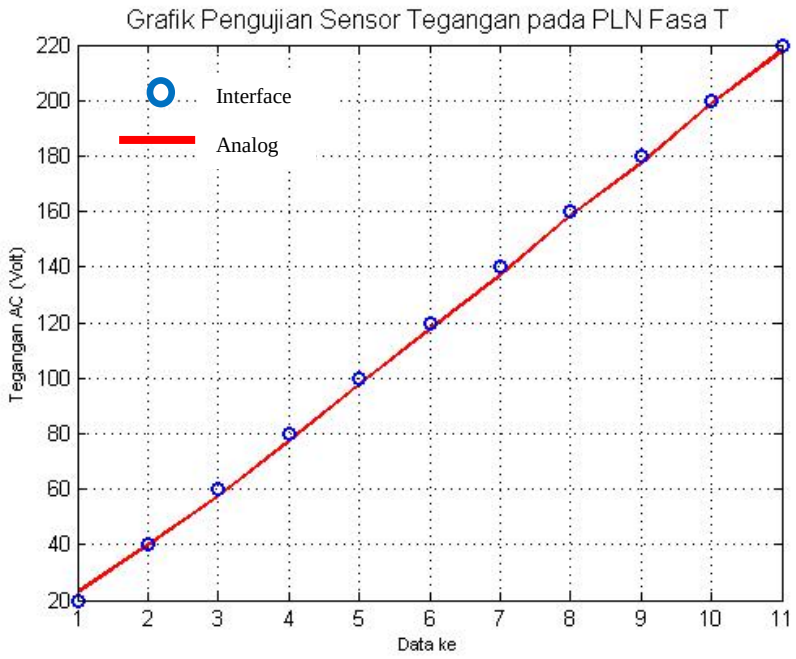
$$Tegangan\ PLN3 = \frac{\left(\frac{b(3)*5}{1023} - 2.5538\right)}{0.0103}\ Volt \qquad (4.6)$$

Dari persamaan 4.6 di atas didapatkan hasil perbandingan nilai tegangan antara yang terbaca pada interface dengan alat ukur analog pada tabel 4.8 :

Tabel 4.8 Tegangan Pada Alat Ukur Analog dan Interface

Data ke	**Tegangan Menggunakan Alat Ukur Analog (VAC)**	**Tegangan Terbaca di Interface (VAC)**	**Error (%)**
1	20	22.93	14.65
2	40	39,76	0.6
3	60	57,146	4.7
4	80	77,23	3.4
5	100	97,87	2.13
6	120	117,42	1.35
7	140	136,97	2.1
8	160	158,69	0.8
9	180	177,07	1.6
10	200	198,88	1.2
11	220	217,89	0.95
Rerata			**3.13**

Data tabel 4.8 dapat dibuat grafik pengujian sensor tegangan fasa T pada PLN dengan membandingkan antara alat ukur analog dengan yang terbaca pada interface.



Gambar 4.12 Grafik Pengujian Sensor Tegangan Fasa T pada PLN

4.2.4 Pengujian Sensor Tegangan Fasa R pada Generator

Pengujian sensor tegangan fasa R pada Generator dilakukan setelah mengetahui output tegangan DC yang terbaca pada sensor dengan tegangan input AC yang diberikan. Tabel 4.9 memperlihatkan hasil baca sensor setelah diberikan tegangan input AC.

Tabel 4.9 Nilai tegangan sensor pada Generator fasa R

No	Tegangan Input (Vac)	Tegangan Output (Vdc)
1	20	2.7
2	40	2.9
3	60	3.08
4	80	3.24
5	100	3.46
6	120	3.53
7	140	3.71
8	160	3.9
9	180	4.08
10	200	4.4
11	220	4.59

Dari tabel 4.9 dapat dibuat persamaan polinomial orde 1 dengan mengetahui nilai inputan AC dengan besar tegangan dari 20-220 V dan tegangan output DC dengan maksimal tegangan 5 V DC. Berikut script untuk mengetahui persamaan polinomial orde 1 :

```
ac = [20  40 60 80 100 120 140 160 180 200 220};
dc = [2.7 2.9 3.08 3.24 3.46 3.53 3.71 3.9 4.08 4.4 4.59];
p = polyfit(ac,dc,1)
p = 0.0091    2.5071 (Dengan 0.0091 adalah nilai a dan 2.5071 adalah
nilai b)
```

Hasil pendekatan polinomial orde 1 mengunakan funsgi ‘polyfit.m’, didapatkan persamaan hubungan tegangan input sensor terhadap tegangan output sensor, yaitu Persamaan 4.7.

$$V_{sensor\ tegangan} = 0.0091 * Tegangan_{sensor} + 2.5071\ Volt \qquad (4.7)$$

Dari persamaan 4.7 dapat dibuat persamaan untuk melihat hasil pengukuran yang terbaca pada interface dengan output tegangan AC (0-220 Volt) dengan yang terukur pada alat ukur analog (0-220 V) melalui persamaan 4.8 di bawah ini :

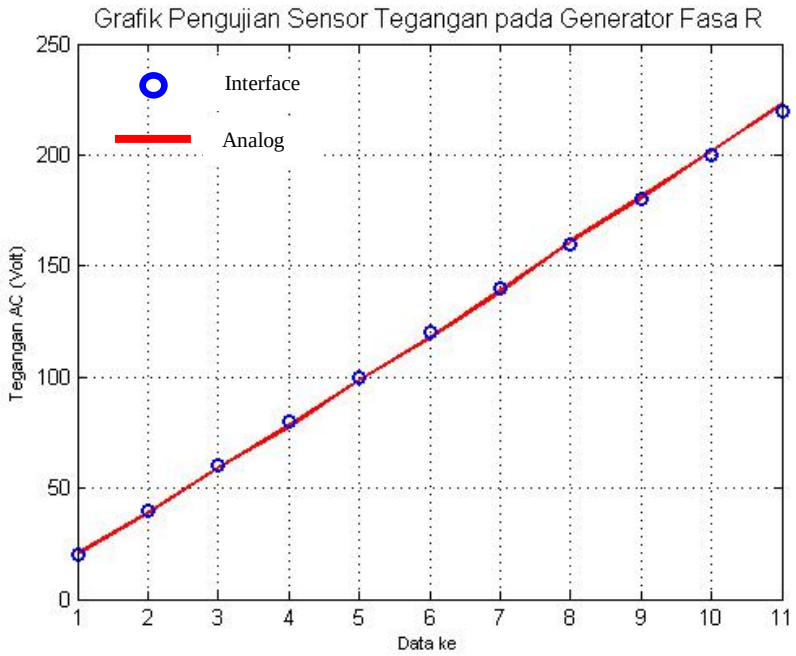
$$Tegangan\ Generator1 = \frac{\left(\frac{b(4)*5}{1023} - 2.5071\right)}{0.0091}\ Volt \qquad (4.8)$$

Dari persamaan 4.8 di atas didapatkan hasil perbandingan nilai tegangan antara yang terbaca pada interface dengan alat ukur analog dapat dilihat pada Tabel 4.10 .

Tabel 4.10 Tegangan pada Interface dengan Alat Ukur Analog

Data ke	**Tegangan Menggunakan Alat Ukur Analog (VAC)**	**Tegangan Terbaca di Interface (VAC)**	**Error (%)**
1	20	20,56	2.8
2	40	38,48	3.8
3	60	58,89	1.85
4	80	77,69	2.8
5	100	98,638	1.62
6	120	117,43	2.1
7	140	138,38	1.1
8	160	160,94	0.58
9	180	180,81	0.45
10	200	201,76	0.54
11	220	222,63	1.19
Rerata			**1.7**

Dari Tabel 4.10 dapat dibuat grafik pengujian sensor tegangan pada Generator fasa R dengan membandingkan antara alat ukur analog dengan yang terbaca pada interface.



Gambar 4.13 Grafik Pengujian Sensor Tegangan Fasa R Generator

4.2.5 Pengujian Sensor Tegangan Fasa S pada Generator

Pengujian sensor tegangan fasa S pada Generator dilakukan setelah mengetahui output tegangan DC yang terbaca pada sensor dengan tegangan input AC yang diberikan. Tabel 4.11 memperlihatkan hasil baca sensor setelah diberikan tegangan input AC.

Tabel 4.11 Nilai tegangan sensor fasa S pada Generator

No	Tegangan Input (Vac)	Tegangan Output (Vdc)
1	20	2.72
2	40	2.91
3	60	3.08
4	80	3.23
5	100	3.44
6	120	3.51
7	140	3.69
8	160	3.88
9	180	4.06
10	200	4.37
11	220	4.59

Dari Tabel 4.11 dapat dibuat persamaan polinomial orde 1 dengan mengetahui nilai inputan AC dengan besar tegangan dari 20-220 V dan tegangan output DC dengan maksimal tegangan 5 V DC. Berikut script untuk mengetahui persamaan linear orde 1 :

```
ac = [20  40 60 80 100 120 140 160 180 200 220};
dc = [2.72 2.91 3.08 3.23 3.44 3.51 3.69 3.88 4.06 4.37 4.59];
p = polyfit(ac,dc,1)
p = 0.0089    2.5156 (dengan 0.0089 adalah nilai a dan 2.5156 adalah
nilai b)
```

Hasil pendekatan polinomial orde 1 menggunakan funsgi ‘polyfit.m’, didapatkan persamaan hubungan tegangan input sensor terhadap tegangan output sensor, yaitu pada persamaan 4.9.

$$V_{sensor\ tegangan} = 0.0089 * Tegangan_{sensor} + 2.5156\ Volt \qquad (4.9)$$

Dari persamaan 4.9 dapat dibuat persamaan untuk melihat hasil pengukuran yang terbaca pada interface dengan output tegangan AC (20-220 Volt) dengan yang terukur pada alat ukur analog (20-220 Volt) melalui persamaan 4.10 di bawah ini :

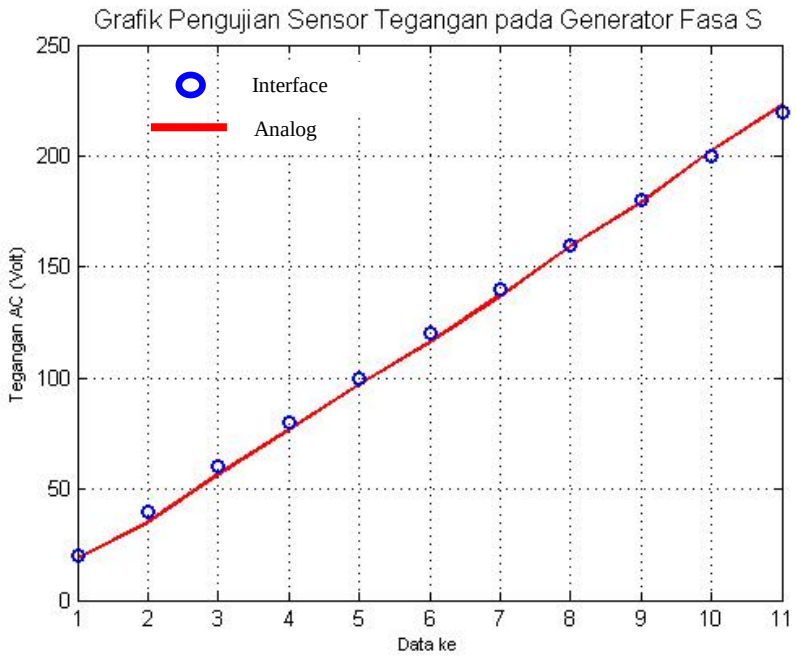
$$Tegangan\ Generator2 = \frac{\left(\frac{b(5)*5}{1023} - 2.5156\right)}{0.0089}\ Volt \qquad (4.10)$$

Dari persamaan 4.10 di atas didapatkan hasil perbandingan nilai tegangan antara yang terbaca pada interface dengan alat ukur analog pada tabel 4.12 :

Tabel 4.12. Tegangan pada Interface dan Alat Ukur Analog

Data ke	**Tegangan Menggunakan Alat Ukur Analog (VAC)**	**Tegangan Terbaca di Interface (VAC)**	**Error (%)**
1	20	18,5	7.5
2	40	34,78	9.3
3	60	56,52	5.8
4	80	76,77	4.03
5	100	96,822	3.52
6	120	116,09	3.25
7	140	136,82	2.2
8	160	159,32	0.4
9	180	179,06	0.52
10	200	202,09	0.88
11	220	222,65	1.3
Rerata			**3.5**

Pada tabel 4.12 dapat dibuat grafik pengujian sensor tegangan pada fasa S Generator dengan membandingkan antara alat ukur analog dengan yang terbaca pada interface.



Gambar 4.14 Grafik Pengujian Sensor Tegangan Fasa S pada Generator

4.2.6 Pengujian Sensor Tegangan Fasa T pada Generator

Pengujian sensor tegangan fasa T pada Generator dilakukan setelah mengetahui output tegangan DC yang terbaca pada sensor dengan tegangan input AC yang diberikan. Tabel 4.13 memperlihatkan hasil baca sensor setelah diberikan tegangan input AC.

Tabel 4.13 Nilai tegangan sensor pada fasa T Generator

No	Tegangan Input (Vac)	Tegangan Output (Vdc)
1	20	2.69
2	40	2.87
3	60	3.07
4	80	3.26
5	100	3.44
6	120	3.61
7	140	3.81
8	160	4
9	180	4.2
10	200	4.4
11	220	4.6

Dari Tabel 4.13 dapat dibuat persamaan polinomial orde 1 dengan mengetahui nilai inputan AC dengan besar tegangan dari 20-220 V dan tegangan output DC dengan maksimal tegangan 5 V DC. Berikut script untuk mengetahui persamaan polinomial orde 1 :

```
ac = [20  40 60 80 100 120 140 160 180 200 220};
dc = [2.69 2.87 3.07 3.26 3.44 3.61 3.81 4 4.2 4.4 4.6];
p = polyfit(ac,dc,1)
p = 0.0090    2.5007 (dengan 0.0090 adalah nilai a dan 2.5007 adalah
nilai b)
```

Hasil pendekatan polinomial orde 1 menggunakan funsgi ‘polyfit.m’, didapatkan persamaan hubungan tegangan input sensor terhadap tegangan output sensor, yaitu Persamaan 4.11.

$$V_{sensor\ tegangan} = 0.0090 * Tegangan_{sensor} + 2.5007\ Volt \qquad (4.11)$$

Dari persamaan 4.11 dapat dibuat persamaan untuk melihat hasil pengukuran yang terbaca pada interface dengan output tegangan AC (0-220 Volt) dengan yang terukur pada alat ukur analog (0-220 V) melalu persamaan 4.12 di bawah ini :

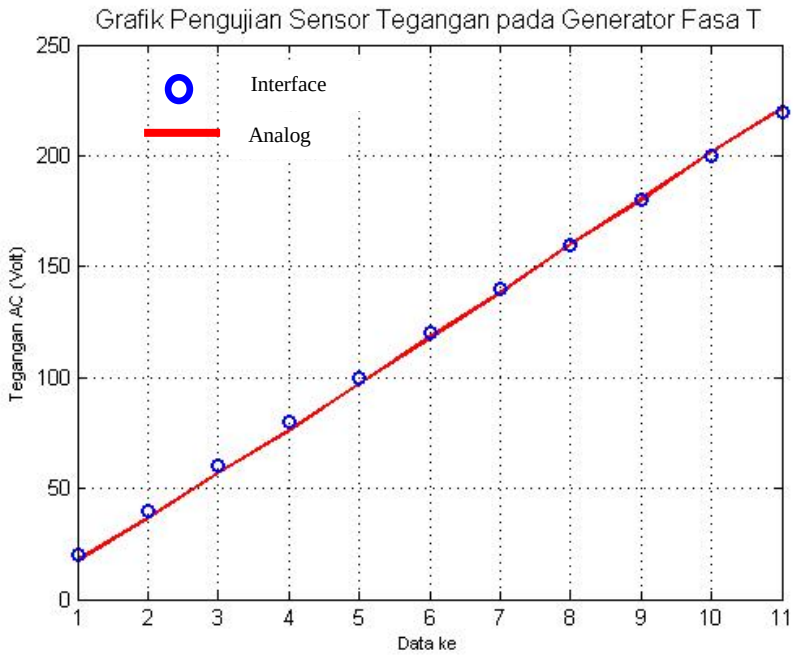
$$Tegangan\ Generator2\ = \frac{\left(\frac{b(5)*5}{1023} - 2.5007\right)}{0.0090}\ Volt \qquad (4.12)$$

Dari persamaan 4.12 di atas didapatkan hasil perbandingan nilai tegangan antara yang terbaca pada interface dengan alat ukur analog pada tabel 4.14 :

Tabel 4.14. Perbandingan Nilai Tegangan Antara yang Terbaca pada Interface dengan Alat Ukur Analog

Data ke	**Tegangan Menggunakan Alat Ukur Analog (VAC)**	**Tegangan Terbaca di Interface (VAC)**	**Error (%)**
1	20	17,47	2.8
2	40	36,25	9.3
3	60	56,65	5.5
4	80	76,11	4.8
5	100	97,0	3
6	120	117,87	1.77
7	140	137,8	1.5
8	160	160,10	0.06
9	180	180,03	0.01
10	200	201,38	0.69
11	220	221,31	0.45
Rerata			**0.02**

Dari Tabel 4.14 dapat dibuat grafik pengujian sensor tegangan pada Generator fasa T dengan membandingkan antara alat ukur analog dengan yang terbaca pada interface.



Gambar 4.15 Grafik Pengujian Sensor Tegangan Fasa T pada Generator

Dari semua sensor tegangan pada generator dan PLN dapat dihitung rerata persen error untuk semua sensor tegangan. Berikut tabel 4.15 untuk rata-rata error sensor tegangan.

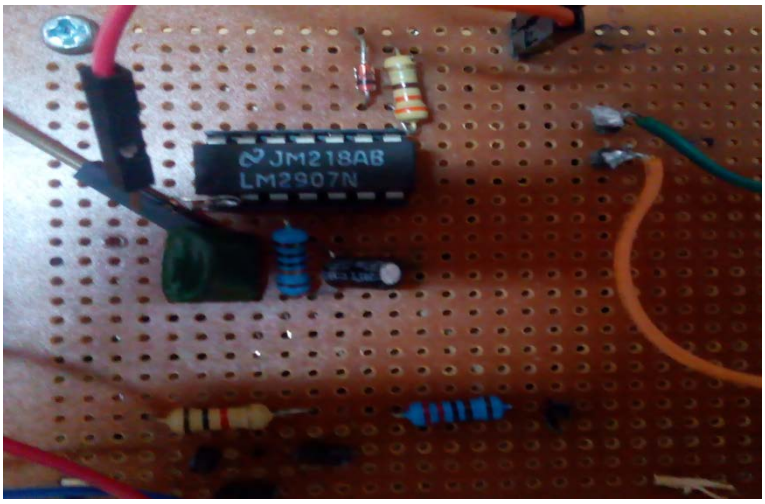
Tabel 4.15 Rerata error sensor tegangan.

NO	Jenis		Error (%)
1	**Generator**	1	1.7
2		2	3.5
3		3	0.02
4	**PLN**	1	2.47
5		2	0.03
6		3	3.13
Rerata			**1.8**

4.3 Pengujian Rangkaian Sensor Frekuensi

Sensor frekuensi yang digunakan pada penelitian menggunakan IC LM2907 . IC LM2907 merupakan IC yang inputannya berupa frekuensi dengan outputnya adalah tegangan. IC LM 2907 pada penelitian ini membaca frekuensi dari 46 sampai dengan 55 Hz.

Pengujian rangkaian sensor frekuensi ini bertujuan untuk mengetahui kesamaan dari sensor frekuensi yang digunakan. Pengujian dilakukan dengan mengubah-ubah nilai frekuensi dengan cara mengubah-ubah kecepatan pada motor DC yang digunakan.



Gambar 4.16 Sensor Frekuensi

Berdasarkan prinsip dari IC LM 2907 yang merubah masukan nilai frekuensi menjadi tegangan, nilai tegangan output bergantung dari frekuensi masukan yang diberikan, tegangan VCC dengan maksimal 12 Volt DC, Kapasitor dan Resistor yang digunakan.

Dalam penelitian ini digunakan dua sensor frekuensi, dimana satu sensor membaca frekuensi dari Generator dan yang satunya lagi digunakan untuk membaca frekuensi dari jala-jala PLN, tegangan output dari sensor dibaca dari interface.

Tabel 4.16 Nilai frekuensi input dan tegangan output

No	**Frekuensi Input Sensor (Hz)**	**Tegangan Output Sensor 1 (Volt DC)**	**Tegangan Output Sensor 2 (Volt Dc)**
1	46	1.43	1.55
2	47	1.47	1.58
3	48	1.5	1.61
4	49	1.52	1.64
5	50	1.54	1.68
6	51	1.56	1.71
7	52	1.59	1.73
8	53	1.61	1.76
9	54	1.65	1.79
10	55	1.67	1.82

Dari Tabel 4.16 menunjukan untuk mendapatkan frekuensi yang berubah-ubah, maka pada penelitian ini digunakan motor DC dengan mengatur kecepatan motor, semakin cepat kecepatan motor maka frekuensi yang diperoleh semakin besar,semakin besarnya nilai frekuensi ini mengakibatkan tegangan keluaran sensor akan berubah semakin besar pula sehingga dapat dikatakan bahwa sensor bekerja.

Dari hasil pengujian tersebut kemudian dibuat pendekatan untuk mendapatkan persamaan yang digunakan sebagai pemrograman pembacaan sensor. Pendekatan polinomial orde 1, orde 2 dan orde 3 yang dapat dibuat berdasarkan data-data hasil pengujian Sensor Frekuensi IC LM2907 pada PLN sebagai contoh menggunakan fungsi ‘polyfit.m’ dari MATLAB adalah sebagai berikut:

a. Persamaan polinomial orde 1

```
fre = [46 47 48 49 50 51 52 53 54 55];
teg = [1.43 1.47 1.5 1.52 1.54 1.56 1.59 1.61 1.65 1.67];
p = polyfit(fre,teg,1)
p = 0.0255    0.2685 (dengan 0.0255 adalah nilai a dan 0.2685 adalah
nilai b)
```

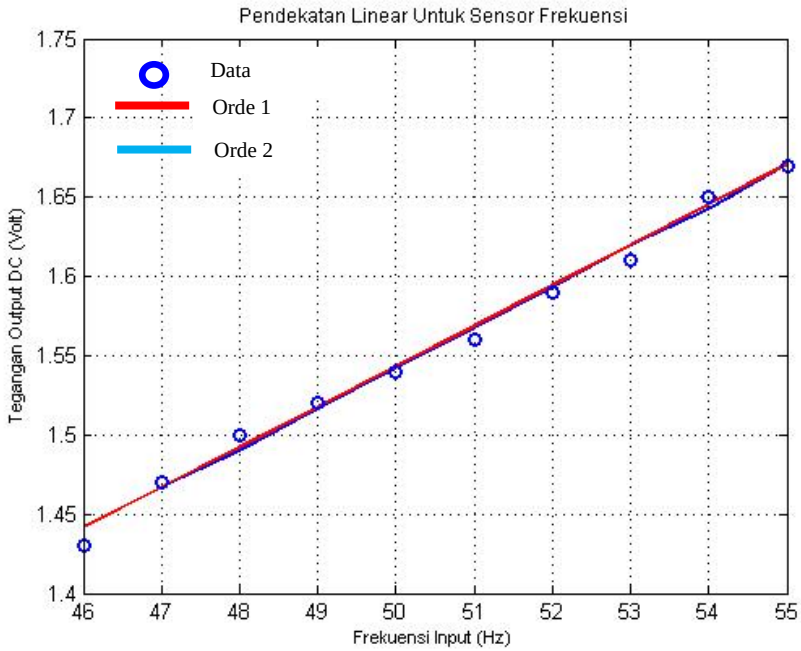
b. Persamaan polinomial orde 2

```
fre = [46 47 48 49 50 51 52 53 54 55];
teg = [1.43 1.47 1.5 1.52 1.54 1.56 1.59 1.61 1.65 1.67];
p = polyfit(fre,teg,2)
p = -0.0000    0.0255    0.2685 (dengan -0.000 adalah nilai a 0.0255
adalah nilai b dan 0.2685 adalah nila c)
```

c. Persamaan polinomial orde 3

```
fre = [46 47 48 49 50 51 52 53 54 55];
teg = [1.43 1.47 1.5 1.52 1.54 1.56 1.59 1.61 1.65 1.67];
p = polyfit(fre,teg,3)
p = 0.0003   -0.0383    1.9538  -32.0668 (dengan 0.0003 adalah nilai
a, -0.0383 adalah nilai b, 1.9538 adalah nilai c dan -32.0668 adalah
nilai d)
```

Dari persamaan masing-masing pendekatan linear di atas dapat dibuat grafik untuk dapat melihat pendekatan linear yang memiliki persentase error lebih kecil untuk dapat digunakan pada kalilbrasi sensor frekuensi. Berikut grafik pendekatan linear untuk masing-masing orde :



Gambar 4.17. Persamaan Linear dan Persamaan Sensor Frekuensi

Dari Gambar 4.17 di atas dapat dihitung persentase error pada setiap orde sehingga pendekatan orde mana yang lebih bagus dapat digunakan pada kalibrasi sensor tegangan. Tabel 4.17 persentase error untuk masing-masing orde :

Tabel 4.17. Persentase error masing-masing persamaan linear

No	Persamaan Polinomial	Rerata Error (%)
1	Orde 1	0.364
2	Orde 2	0.366
3	Orde 3	50

Dari Tabel 4.17 dapat dilihat besar persentase error masing-masing persamaan linear dengan persentase error terkecil pada orde 1. Oleh karena itu maka pendekatan yang digunakan untuk kalibrasi sensor tegangan menggunakan persamaan linear orde 1.

4.3.1 Pengujian Sensor Frekuensi pada PLN

Untuk melakukan pengujian sensor frekuensi pada PLN maka digunakan data-data frekuensi inputan dengan output tegangan DC yang terukur pada sensor. Berikut script untuk menentukan persamaan polinomial orde 1 :

```
fre = [46 47 48 49 50 51 52 53 54 55];
teg = [1.43 1.47 1.5 1.52 1.54 1.56 1.59 1.61 1.65 1.67];
p = polyfit(fre,teg,1)
p = 0.0255    0.2685 (dengan 0.0255 adalah nilai a dan 0.2685 adalah
nilai b)
```

Hasil persamaan polinomial orde 1 menggunakan funsgi ‘polyfit.m’, didapatkan persamaan hubungan frekuensi input sensor frekuensi terhadap keluaran tegangan DC , yaitu Persamaan 4.13.

$$V_{sensor\ frekuensi} = 0.0255 \quad * Frekuensi_{sensor} + 0.2685\ Volt \qquad (4.13)$$

Dari persamaan 4.13 dapat dibuat persamaan untuk melihat hasil pengukuran yang terbaca pada interface dengan output frekuensi (46 – 55 Hz) dengan yang terukur pada alat ukur analog (46 – 55 Hz) melalu persamaan 4.14 di bawah ini :

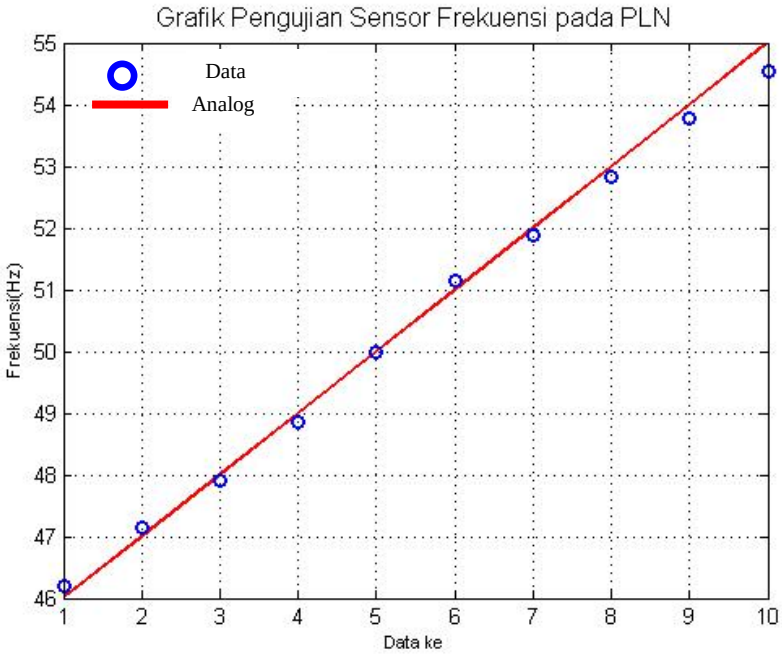
$$Frekuensi\ PLN\ = \frac{\left(\frac{b(7)*5}{1023} - 0.2685\right)}{0.0255}\ Hz \qquad (4.14)$$

Dari persamaan 4.14 di atas didapatkan hasil perbandingan nilai frekuensi antara yang terbaca pada interface dengan alat ukur analog pada Tabel 4.18:

Tabel 4.18 Frekuensi pada Interface dengan Alat Ukur Analog

Data ke	**Frekuensi Menggunakan Alat Ukur Analog (Hz)**	**Frekuensi Terbaca di Interface (Hz)**	**Error (%)**
1	46	46,208	0.4
2	47	47,155	0.32
3	48	47,913	0.2
4	49	48,860	0.4
5	50	49,997	0.2
6	51	51,133	0.2
7	52	51,89	0.38
8	53	52,838	0.37
9	54	53,786	0.5
10	55	54,540	0.9
Rerata			**0.38**

Dari Tabel 4.18 dapat dibuat grafik pengujian sensor frekuensi pada PLN dengan membandingkan antara alat ukur analog dengan yang terbaca pada interface.



Gambar 4.18 Grafik Pengujian Sensor Frekuensi pada PLN

4.3.2 Pengujian Sensor Frekuensi Generator

Untuk melakukan pengujian sensor frekuensi pada Generator maka digunakan data-data frekuensi inputan dengan output tegangan DC yang terukur pada sensor. Berikut script untuk menentukan persamaan polinomial orde 1 :

```
fre = [46 47 48 49 50 51 52 53 54 55];
teg = [1.55 1.58 1.61 1.64 1.68 1.71 1.73 1.76 1.79 1.82];
p = polyfit(fre,teg,1)
p = 0.0300    0.1720
```

Hasil persamaan polinomial menggunakan funsgi ‘polyfit.m’, didapatkan persamaan hubungan frekuensi input sensor frekuensi terhadap keluaran tegangan DC , yaitu Persamaan 4.15.

$$V_{sensor\ frekuensi} = 0.0300 \quad * Frekuensi_{sensor} + 0.1720 Volt \tag{4.15}$$

Dari persamaan 4.15 dapat dibuat persamaan untuk melihat hasil pengukuran yang terbaca pada interface dengan output frekuensi (46 – 55 Hz) dengan yang terukur pada alat ukur analog (46 – 55 Hz) melalu persamaan 4.16 di bawah ini :

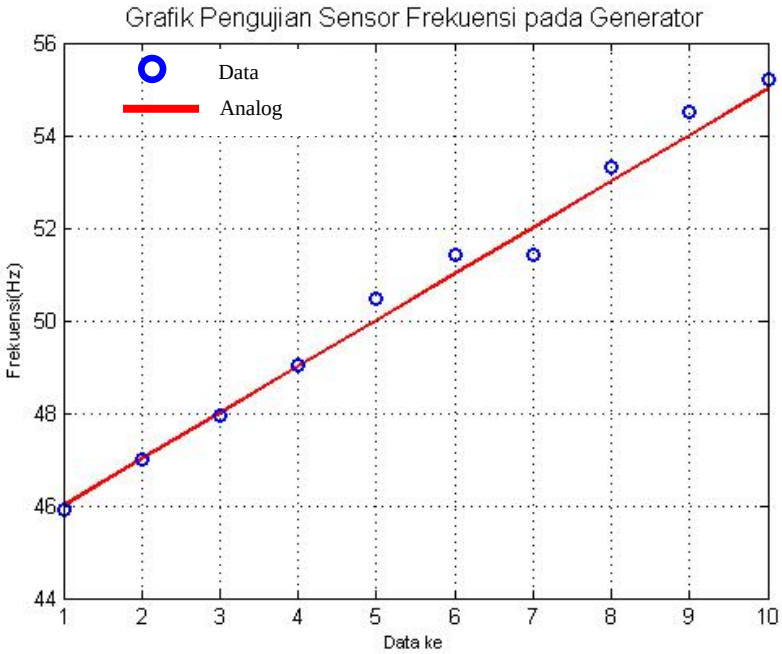
$$Frekuensi\ Generator\ = \frac{\left(\frac{b(7)*5}{1023} - 0.1720\right)}{0.0300}\ Hz \tag{4.16}$$

Dari persamaan 4.16 di atas didapatkan hasil perbandingan nilai frekuensi antara yang terbaca pada interface dengan alat ukur analog pada tabel 4.19 :

4.19. Frekuensi Pada Interface dengan Alat Ukur Analog

Data ke	Frekuensi Menggunakan Alat Ukur Analog (Hz)	Frekuensi Terbaca di Interface (Hz)	Error (%)
1	46	45.9	0.21
2	47	47.01	0.02
3	48	47.95	0.1
4	49	49.02	0.04
5	50	50.49	0.98
6	51	51.43	0.84
7	52	51.43	1.09
8	53	53.3	0.56
9	54	54.5	0.92
10	55	55.2	0.36
Rerata			**0.51**

Dari Tabel 4.19 dapat dibuat grafik pengujian sensor frekuensi pada Generator dengan membandingkan antara alat ukur analog dengan yang terbaca pada interface.



Gambar 4.19 Grafik Pengujian Sensor Frekuensi pada Generator

Dari kedua sensor frekuensi pada generator dan PLN dapat dihitung persentase error rata-rata sensor frekuensi. Berikut Tabel 4.20 untuk rata-rata error sensor frekuensi

Tabel 4.20 Error Rerata sensor frekuensi

No	Jenis	Error (%)
1	PLN	0.38
2	Generator	0.51
Rerata		**0.44**

4.4 Pembuatan GUI (*Grapic User Interface*) pada Komputer

Hasil pengukuran dan pengujian yang dilakukan membutuhkan media sebagai tempat ditampilkannya hasil pengukuan dan pengujian tersebut, untuk itu diperlukan suatu program untuk menampilkan secara visual dari ADC (Arduino) pada port serial. *Software* yang digunakan untuk menampilkan secara visual tegangan, arus dan kecepatan dari Motor Induksi Tiga Phase ini adalah MATLAB. Pembacaan serial dengan *Software* dapat menggunakan fungsi serial. Contoh pembacaan data dari serial sebagai berikut:

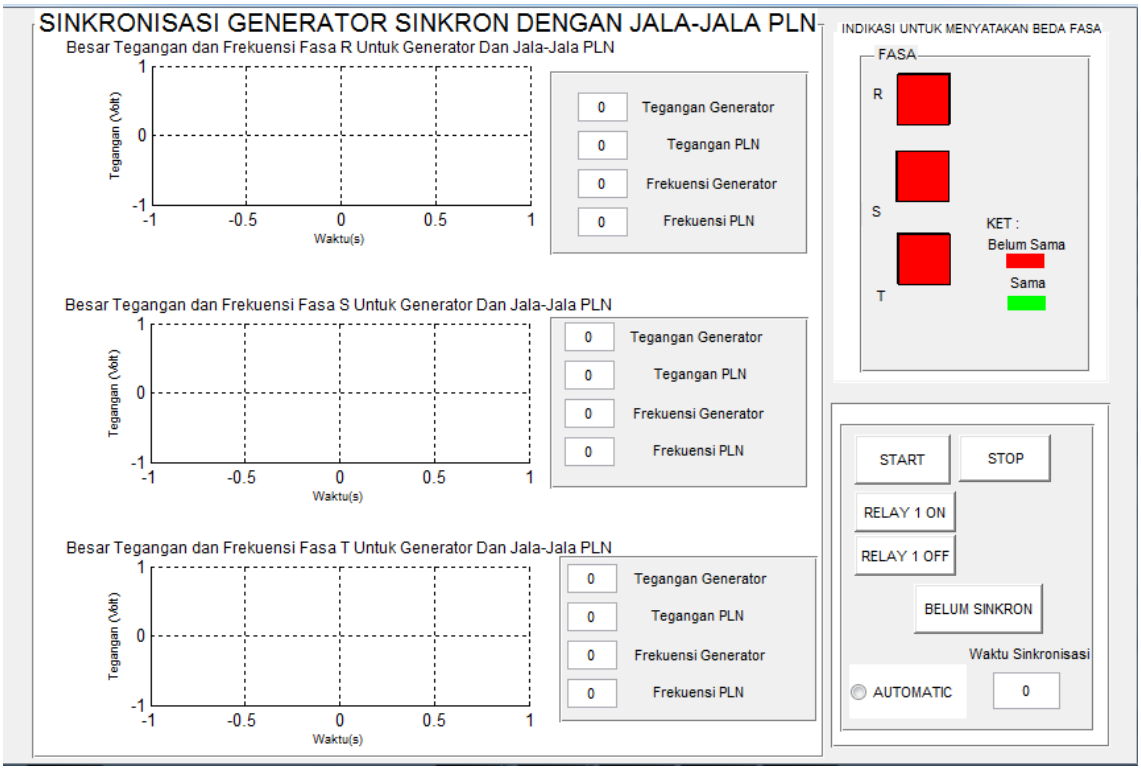
```
delete(instrfind({'Port'},{'COM4'}));
arduino = serial('COM4');
set (arduino, 'Baudrate',115200);
fopen(arduino);
b = fscanf(arduino, '%f %f %f');
```

Perintah delete(`instrfind({'Port'},{'COM4'})`berfungsi untuk menghapus variabel pada Port COM4. Variable ‘arduino’ membaca port serial pada COM4 dengan kecepatan transfer data sebesar 115200 bit/sec. kemudian dibuka dengan perintah fopen,dan ditulis nilai float yang terbaca dengan perintah fscanf pada variabel ‘b’. Variabel ‘b’ akan terisi sembilan data integer untuk masing-masing nilai tegangan, frekuensi dan waktu.

Hasil pembacaan dari port serial kemudian diolah pada softwere untuk ditampilkan secara grafis. Setiap data tegangan, frekuensi dan waktu yang diterima dari port serial adalah sebesar 10 bit. Sepuluh bit berarti rentang data kecepatan dan suhu antara 0 – 1023. Nilai 0 – 1023 secara analog terukur tegangan 0 sampai 5 volt, dibutuhkan program pada *Software* yang bisa menterjemahkan nilai tersebut ke dalam tegangan dan frekuensi yang sesuai dengan tegangan dan frekuensi pada jala-jala PLN dan generator pada persamaan yang ada masing-masing sensor.

Listing program untuk interface pengukuran arus dan kecepatan motor induksi tiga phase untuk penelitian ini dilampirkan pada lampiran. Pembuatan program untuk

visualisasi menggunakan GUI (*Grapich User Interface*) dari MATLAB. Berikut GUI yang telah dibuat untuk penelitian ini.



Gambar 4.20 Main Menu Program *Interface*

Fungsi setiap bagian yang ada pada main menu tersebut adalah sebagai berikut:

Kolom Menu

Tombol Start : Digunakan untuk memulai pengambilan data yang dimulai instruksi dari Matlab yang kemudian di Matlab akan membaca keluaran masukan dari port serial (RS232) yang berisi informasi tegangan dan frekuensi dari generator dan jala-jala PLN dalam bentuk bit (8 kali 10 bit). Nilai bit tersebut dengan program MATLAB dikonversi menjadi nilai tegangan V) dan frekuensi (Hz). kemudian nilai ini ditampikan pada grafik secara real time.

Tombol Stop : Digunakan untuk menghentikan pengambilan data,

Tombol Relay 1 On : Digunakan untuk menginstruksikan arduino untuk menutup relay 1 yang semula berada pada posisi terbuka.

Tombol Relay 1 Off : Digunakan untuk menginstruksikan arduino untuk membuka relay 1 yang semula berada pada posisi tertutup.

Tombol Automatic : Digunakan untuk mengontrol relay untuk membuka pada saat semua variable sudah sinkron.

4.5 Pengukuran dan Pengujian Keseluruhan Sistem

Pengujian keseluruhan sistem dimaksudkan untuk melihat kinerja dari setiap perangkat yang telah dirancang mulai dari perangkat keras berupa sensor-sensor untuk pengukuran, dan perangkat lunak berupa antarmuka yang dirancang pada PC, apakah telah mampu bekerja sesuai dengan yang diingikan.

Perangkat yang telah dibuat akan dihubungkan dengan generator sinkron dan jala-jala PLN. Pengukuran dan pengujian keseluruhan sistem ini meliputi pengukuran nilai tegangan 3 fasa dan frekuensi pada jala-jala PLN dan generator sinkron. Selain itu dari nilai tegangan dan frekuensi akan dicari nilai sudut fasa pada masing-masing fasa sehingga dapat dilihat besar sudut fasa pada masing-masing fasa pada generator dan jala-jala PLN.

4.4.1 Pengukuran Waktu Masuknya Sinkronisasi

Untuk mengetahui keandalan sistem dalam penggabungan antara generator dan jala-jala PLN dalam seberapa cepat menghubungkan maka dilakukan percobaan sebanyak 20 kali untuk menggabungkan generator dan jala-jala PLN secara otomatis pada saat keduanya sudah sinkron, ditunjukkan pada Tabel 4.20 di bawah ini :

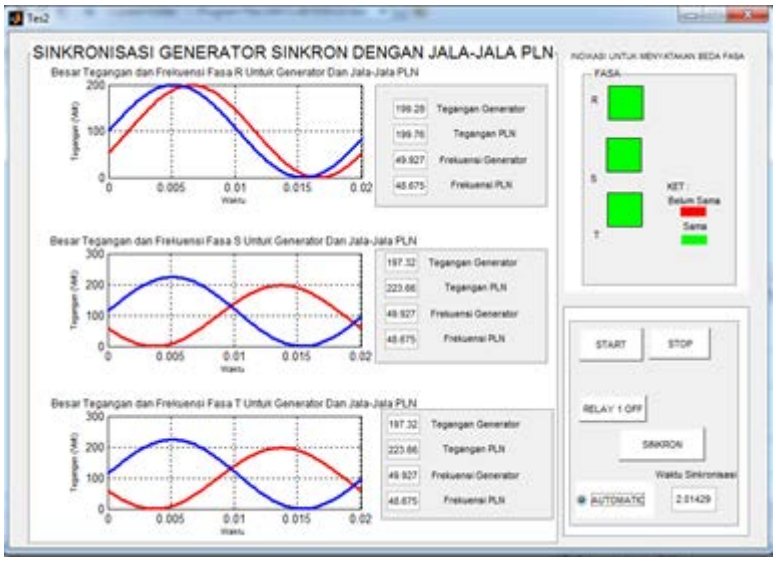
Tabel 4.21. Waktu Sinkronisasi Antara Generator dan Jala-Jala PLN

No	Percobaan	Waktu (S)
1	Ke-1	0.0375
2	Ke-2	0.0517
3	ke-3	0.0468
4	Ke-4	0.0331
5	Ke-5	0.0471
6	Ke-6	0.0457
7	Ke-7	0.0423
8	Ke-8	0.0340
9	Ke-9	0.0341
10	Ke-10	0.0398
11	Ke-11	0.0390
12	Ke-12	0.0436
13	Ke-13	0.0413
14	Ke-14	0.0444
15	Ke-15	0.0332
16	Ke-16	0.0341
17	Ke-17	0.0331
18	Ke-18	0.0423
19	Ke-19	0.0453
20	Ke-20	0.0342
Rata-rata		0.0401

Data Tabel 4.21 menunjukkan rerata waktu sinkronisasi sebesar 0.0401 detik untuk menyambung relay setelah pembacaan pada interface menyatakan sudah sinkron.

4.4.2 Tampilan Interface Setelah Masuknya Sinkronisasi

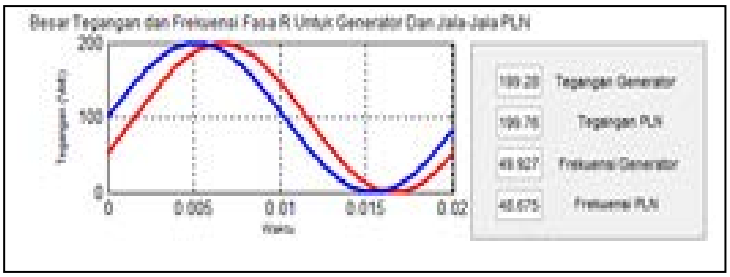
Setelah proses tersambungnya sumber PLN dan Generator maka perlu diketahui proses selanjutnya pada interface untuk melihat besar magnitude gelombang pada setiap fasa antara generator dan PLN. Berikut dapat dilihat tampilan interface setelah terjadinya proses sinkronisasi.



Gambar 4.21 Tampilan interface setelah sinkron

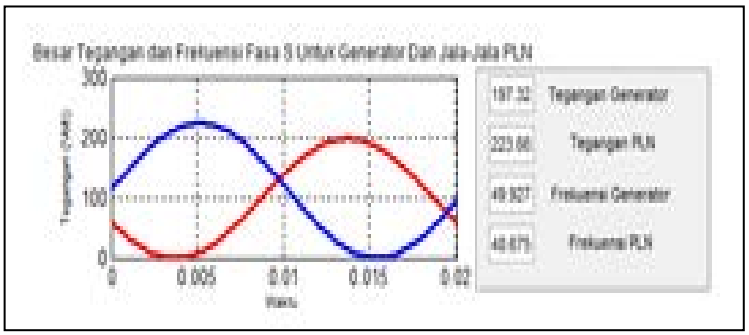
Pada Gambar 4.21 menunjukkan tampilan interface setelah terjadinya sinkronisasi antara generator dan Jala-jala PLN. Magnitude gelombang keluaran pada setiap fasa pada pada PLN maupun pada generator mendekati nilai yang sama dengan nilai sudut fase yang berbeda.

Pada fasa R antara generator dan jala-jala PLN setelah terjadinya sinkronisasi fase gelombang memiliki sedikit perbedaan dengan magnitude tegangan yang sama, berikut dapat dilihat gelombang keluaran pada fasa R antara generator dan jala-jala PLN.

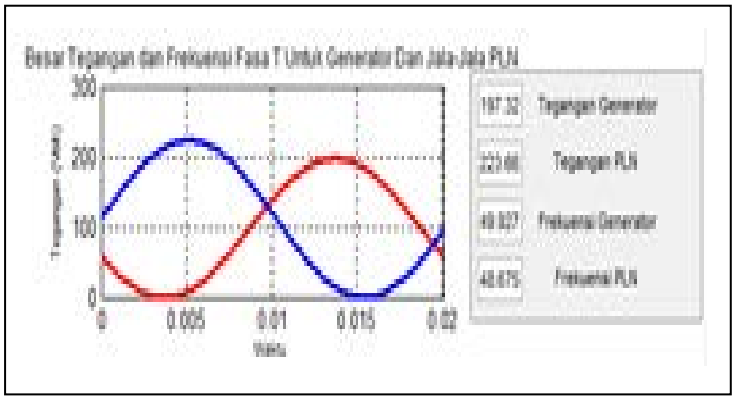


Gambar 4.22 Gelombang keluaran pada Fasa R

Kemudian untuk gelombang pada fasa S dan T pada saat setelah terjadinya sinkronisasi antara generator dan jala-jala PLN memiliki sudut fase dengan beda yang besar antara generator dan jala-jala PLN dengan magnitude yang sama besar. Berikut gelombang keluaran pada generator dan jala-jala PLN pada saat setelah terjadinya sinkronisasi untuk fasa S dan T.



Gambar 4.23 Gelombang keluaran pada Fasa S



Gambar 4.24 Gelombang keluaran pada Fasa T

Dari ketiga fasa tersebut dapat disimpulkan bahwa pada saat setelah terjadinya proses sinkronisasi antara generator dan jala-jala PLN memiliki sudut fase yang berbeda untuk ketiga fase antara generator dan jala-jala PLN dengan besar magnitude dan frekuensi yang sama.

BAB V
PENUTUP

5.1 Kesimpulan

Berdasarkan pengujian dan analisa yang telah dilakukan pada Penelitian ini,dapat diambil beberapa kesimpulan sebagai berikut:

1. Antarmuka (*Interface*) yang dirancang pada computer telah dapat menampilkan grafik hasil pengukuran dari sensor, dan menampilkan nilai hasil pengukuran secara *real time*, kemudian dari hasil pengukuran telah dapat melakukan penyalaan relay secara otomatis sebagai saklar untuk menyambungkan sinkronisasi antara Generator Sinkron ke jala-jala PLN.
2. Pengukuran nilai tegangan dan frekuensi dengan menggunakan persamaan linear orde 1 pada interface dengan alat ukur analog pada Laboratorium Sistem Tenaga Listrik memiliki persentase error 1.8% untuk sensor tegangan dan 0.44% untuk sensor frekuensi.
3. Proses penyambungan setelah terjadinya sinkronisasi antara generator dan jala-jala PLN memiliki waktu rata-rata penyambungan relay sebesar 0.0401 detik.

5.2 Saran

Dalam pengerjaan dan penyelesaian tugas akhir ini tentu tidak lepas dari berbagai macam kekurangan dan kelemahan, baik pada sistem maupun pada peralatan yang telah dibuat. Untuk memperbaiki kekurangan tersebut, maka perlu dilakukan hal-hal sebagai berikut:

1. Pemilihan dan penggunaan komponen yang tepat sehingga saat *pembuatan perangkat keras* untuk interface agar didapatkan dengan harga yang optimal.
2. Dapat digunakan sebagai tambahan modul pratikum mesin-mesin listrik.

DAFTAR PUSTAKA

Arduino Mega2560. (*www.arduino.cc/en/Main/ArduinoBoardMega2560*)

Boldea Ion, ,2006, *Syncrononous Generators, The Electric Generator hand book*, United state of America.

Chomat Miroslav, 2011. "*Electric Machines And Drives*," intech.

Citarsa, I.B.F., 2013, *" Buku Ajar Mata Kuliah Mesin Listrik I"* , Jurusan Teknik Elektro Fakultas Teknik Universitas Mataram.

Efendi, M, 2012. *Rancang Bangun Alat Sinkronisasi Otomatis Sistem Jaringan Pembangkit Listrik Tenaga Mikrohidro dengan Sistem Jaringan PLN.*

Graha, Setia, 2012. *Power management PLN-Genset pada Bank Indonesia Cabang Banjarmasin.* Teknik Elektro Politeknik Negeri Banjarmasin.

Nrartha, I.M. A, Sultan, Muljono, A.B, Suksmadana, I.M.B, 2013. *Rancang Bangun Interface untuk Visualisasi Kecepatan dan Suhu Motor DC Shunt Berbeban (Sebagai Modul Praktikum di Laboratorium Sistem Tenaga).* Teknik Elektro Universitas Mataram.

Nurcahyo, A, Chusn. Y, Ningsih. S, 2012. *Sinkronisasi dan pengamanan modul generator LAB-TST berbasis PLC (Software).* Jurusan Teknik Elektro Industri, Politeknik Elektronika Negeri Surabaya (PENS).

Prasetya. T, Yahya. C. A, Ningsih. S, 2012. *Sinkronisasi dan pengamanan modul generator LAB-TST berbasis PLC (Hardware).* Jurusan Teknik Elektro Industri, Politeknik Elektronika Negeri Surabaya (PENS).

Setia Graha, 2014. *Power management PLN-Genset pada Bank Indonesia Cabang Banjarmasin.*Volume 6, No. 2, Desember2014 : 55 - 102

Suparno, Drs. 2000. *Mesin Listrik 2*. Jakarta :Erlangga.

Thumann Albert, Franze Harry, 2009."*Efficient Electrical system Design Handbook*", The Fairmont Press, United state of America..

Wangiyana, I.M.S.A., 2016. "*Perancangan Sistem Mini SCADA Radio Telemetry 433MHZ"*, Jurusan Teknik Elektro Fakultas Teknik Universitas Mataram.

Woodward, 2004. “*Application GCP-30 Series Genset Control*” .

Woodward, 2004. “*Governing Fundamental and Power Management*”.

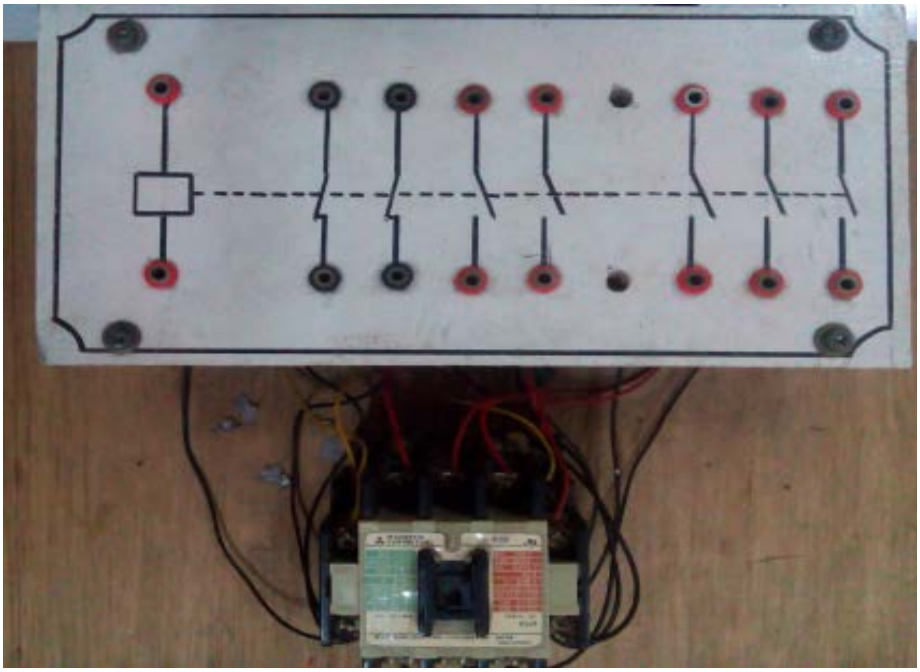
Zuhal, 1988. *Dasar Teknik Tenaga Listrik dan Elektronika Daya*, Jakarta, PT Gramedia Pustaka Utama.

Lampiran 1. Foto-foto dan gambar penelitian

Motor DC 0.3 kW dan Generator Sinkron 0.3 kW

Sumber DC

Voltmeter

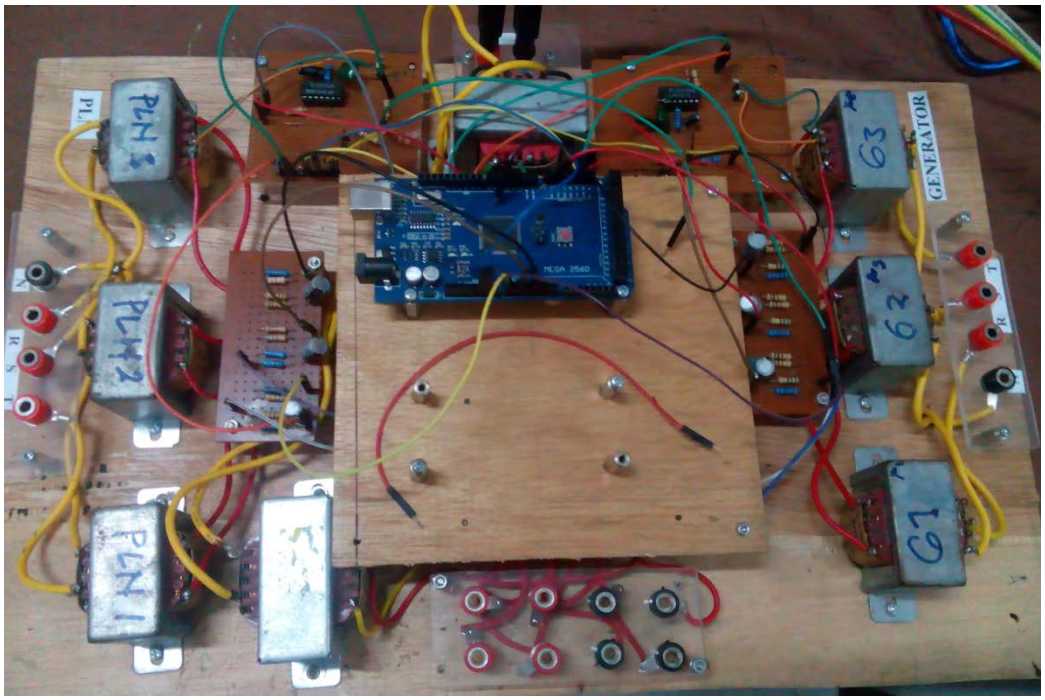


Kontaktor 3 Fasa

Double Frequensi Meter

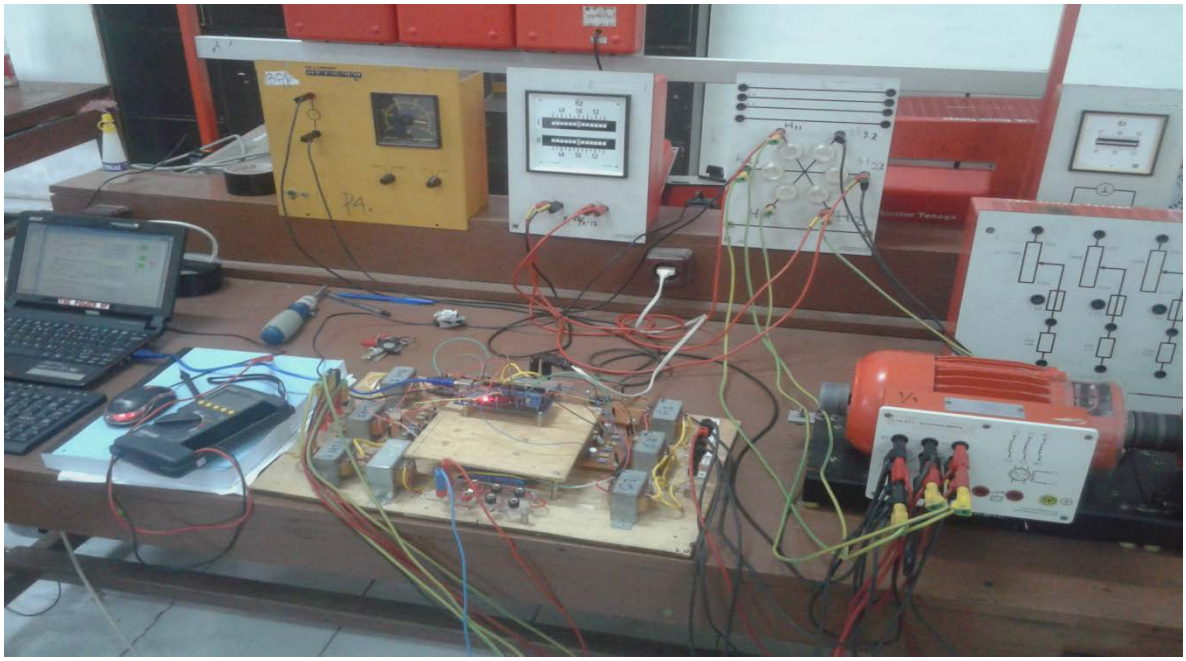
Sinkronising Indikator

Sumber AC 3 Fasa



Perangkat Keras Interface

Pengujian Keseluruhan Sistem



Pengujian Keseluruhan Sistem

Lampiran 2 : Listing Program Visualisasi *Sinkronisasi* Generator dengan Jala-Jala PLN

L2.1. Listing Program Pada Matlab

```
function varargout =Tes2(varargin)
% TES2 MATLAB code for Tes2.fig
%      TES2, by itself, creates a new TES2 or raises the existing
%      singleton*.
%
%      H = TES2 returns the handle to a new TES2 or the handle to
%      the existing singleton*.
%
%      TES2('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in TES2.M with the given input arguments.
%
%      TES2('Property','Value',...) creates a new TES2 or raises the
%      existing singleton*.  Starting from the left, property value pairs
are
%      applied to the GUI before Tes2_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property
%      applicationg
%      stop.  All inputs are passed to Tes2_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only
one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help Tes2

% Last Modified by GUIDE v2.5 22-Aug-2016 14:24:36

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Tes2_OpeningFcn, ...
                   'gui_OutputFcn',  @Tes2_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before Tes2 is made visible.
```

```
function Tes2_OpeningFcn(hObject, eventdata, handles, varargin)

axes(handles.axes1)
%Membuat grafik Axes1
    x1 = -1;x2 = 1;
    y1 = -1;y2 = 1;
    judul = 'Besar Tegangan dan Frekuensi Fasa R Untuk Generator Dan
Jala-Jala PLN';
    set(gca,'Title',text('String',judul,'Color','black','FontSize',9))
    set(get(gca,'XLabel'),'String','Waktu(s)','FontSize',7)
    set(gca,'XLim',[x1 x2])
    set(get(gca,'YLabel'),'String','Tegangan (Volt)','FontSize',7)
    set(gca,'YLim',[y1 y2])
    grid on
%end Axes1

axes(handles.axes4)
%Membuat grafik Axes4
    x1 = -1;x2 = 1;
    y1 = -1;y2 = 1;
    judul = 'Besar Tegangan dan Frekuensi Fasa S Untuk Generator Dan
Jala-Jala PLN';
    set(gca,'Title',text('String',judul,'Color','black','FontSize',9))
    set(get(gca,'XLabel'),'String','Waktu(s)','FontSize',7)
    set(gca,'XLim',[x1 x2])
    set(get(gca,'YLabel'),'String','Tegangan (Volt)','FontSize',7)
    set(gca,'YLim',[y1 y2])
    grid on
%end Axes4
axes(handles.axes5)
%Membuat grafik Axes5
    x1 = -1;x2 = 1;
    y1 = -1;y2 = 1;
    judul = 'Besar Tegangan dan Frekuensi Fasa T Untuk Generator Dan
Jala-Jala PLN';
    set(gca,'Title',text('String',judul,'Color','black','FontSize',9))
    set(get(gca,'XLabel'),'String','Waktu(s)','FontSize',7)
    set(gca,'XLim',[x1 x2])
    set(get(gca,'YLabel'),'String','Tegangan (Volt)','FontSize',7)
    set(gca,'YLim',[y1 y2])
    grid on
%end Axes5

% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Tes2 (see VARARGIN)

% Choose default command line output for Tes2
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);
```

```
% UIWAIT makes Tes2 wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = Tes2_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)

% User selected the second item
% Proceed with callback...
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu1
contents as cell array
%        contents{get(hObject,'Value')} returns selected item from
popupmenu1


% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in pushbutton5.
function pushbutton5_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in pushbutton6.
function pushbutton6_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in pushbutton8.
function pushbutton8_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)



function maks_teg_gen1_Callback(hObject, eventdata, handles)
% hObject    handle to maks_teg_gen1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of maks_teg_gen1 as text
%        str2double(get(hObject,'String')) returns contents of
maks_teg_gen1 as a double


% --- Executes during object creation, after setting all properties.
function maks_teg_gen1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to maks_teg_gen1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function maks_teg_pln1_Callback(hObject, eventdata, handles)
% hObject    handle to maks_teg_pln1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of maks_teg_pln1 as text
%        str2double(get(hObject,'String')) returns contents of
maks_teg_pln1 as a double



% --- Executes during object creation, after setting all properties.
function maks_teg_pln1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to maks_teg_pln1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function maks_frek_gen1_Callback(hObject, eventdata, handles)
% hObject    handle to maks_frek_gen1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of maks_frek_gen1 as text
%        str2double(get(hObject,'String')) returns contents of
maks_frek_gen1 as a double



% --- Executes during object creation, after setting all properties.
function maks_frek_gen1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to maks_frek_gen1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
```

```
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function maks_frek_pln1_Callback(hObject, eventdata, handles)
% hObject    handle to maks_frek_pln1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of maks_frek_pln1 as text
%        str2double(get(hObject,'String')) returns contents of
maks_frek_pln1 as a double


% --- Executes during object creation, after setting all properties.
function maks_frek_pln1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to maks_frek_pln1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function maks_teg_gen2_Callback(hObject, ~, handles)
% hObject    handle to maks_teg_gen2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of maks_teg_gen2 as text
%        str2double(get(hObject,'String')) returns contents of
maks_teg_gen2 as a double


% --- Executes during object creation, after setting all properties.
function maks_teg_gen2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to maks_teg_gen2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function maks_teg_pln2_Callback(hObject, eventdata, handles)
% hObject    handle to maks_teg_pln2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of maks_teg_pln2 as text
%        str2double(get(hObject,'String')) returns contents of
maks_teg_pln2 as a double


% --- Executes during object creation, after setting all properties.
function maks_teg_pln2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to maks_teg_pln2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function maks_frek_gen2_Callback(hObject, eventdata, handles)
% hObject    handle to maks_frek_gen2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of maks_frek_gen2 as text
%        str2double(get(hObject,'String')) returns contents of
maks_frek_gen2 as a double


% --- Executes during object creation, after setting all properties.
function maks_frek_gen2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to maks_frek_gen2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function maks_frek_pln2_Callback(hObject, eventdata, handles)
% hObject    handle to maks_frek_pln2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of maks_frek_pln2 as text
%        str2double(get(hObject,'String')) returns contents of
maks_frek_pln2 as a double


% --- Executes during object creation, after setting all properties.
function maks_frek_pln2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to maks_frek_pln2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit9_Callback(hObject, eventdata, handles)
% hObject    handle to edit9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit9 as text
%        str2double(get(hObject,'String')) returns contents of edit9 as a
double


% --- Executes during object creation, after setting all properties.
function edit9_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit10_Callback(hObject, eventdata, handles)
% hObject    handle to edit10 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit10 as text
%        str2double(get(hObject,'String')) returns contents of edit10 as
a double


% --- Executes during object creation, after setting all properties.
function edit10_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit10 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit11_Callback(hObject, eventdata, handles)
% hObject    handle to edit11 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit11 as text
%        str2double(get(hObject,'String')) returns contents of edit11 as
a double


% --- Executes during object creation, after setting all properties.
function edit11_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit11 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit12_Callback(hObject, eventdata, handles)
% hObject    handle to edit12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit12 as text
%        str2double(get(hObject,'String')) returns contents of edit12 as
a double



% --- Executes during object creation, after setting all properties.
function edit12_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



% --- Executes on button press in cb_menutup.
function cb_menutup_Callback(hObject, eventdata, handles)
% hObject    handle to cb_menutup (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)



% --- Executes on button press in pushbutton11.
function pushbutton11_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton11 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)



% --- Executes on button press in SINKRONASI.
function SINKRONASI_Callback(hObject, eventdata, handles)
% hObject    handle to SINKRONASI (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
function maks_teg_gen3_Callback(hObject, eventdata, handles)
% hObject    handle to maks_teg_gen3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of maks_teg_gen3 as text
%        str2double(get(hObject,'String')) returns contents of
maks_teg_gen3 as a double


% --- Executes during object creation, after setting all properties.
function maks_teg_gen3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to maks_teg_gen3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function maks_teg_pln3_Callback(hObject, eventdata, handles)
% hObject    handle to maks_teg_pln3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of maks_teg_pln3 as text
%        str2double(get(hObject,'String')) returns contents of
maks_teg_pln3 as a double


% --- Executes during object creation, after setting all properties.
function maks_teg_pln3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to maks_teg_pln3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function maks_frek_gen3_Callback(hObject, eventdata, handles)
% hObject    handle to maks_frek_gen3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of maks_frek_gen3 as text
%        str2double(get(hObject,'String')) returns contents of
maks_frek_gen3 as a double


% --- Executes during object creation, after setting all properties.
function maks_frek_gen3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to maks_frek_gen3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function maks_frek_pln3_Callback(hObject, eventdata, handles)
% hObject    handle to maks_frek_pln3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of maks_frek_pln3 as text
%        str2double(get(hObject,'String')) returns contents of
maks_frek_pln3 as a double


% --- Executes during object creation, after setting all properties.
function maks_frek_pln3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to maks_frek_pln3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in tombol_start.
function tombol_start_Callback(hObject, eventdata, handles)
```

```
automatic = get(handles.Automatic, 'Value');

axes(handles.axes1)
cla;
axes(handles.axes4)
cla;
axes(handles.axes5)
cla;
global bro;
bro =1;
global arduino;
global berhenti;
global index;
berhenti = 0;
delete(instrfind({'Port'},{'COM4'}));
arduino = serial('COM4');
set (arduino, 'Baudrate',115200);
fopen(arduino);
pause(1);

global waktu;
global data;
waktu = 0;
data = 0;
index =0;
waktuData = 0;
tegangan_pln1 = 0;
tegangan_pln2 = 0;
tegangan_pln3 = 0;
tegangan_gen1 = 0;
tegangan_gen2 = 0;
tegangan_gen3 = 0;
i = 0;
jmlhS = 0;
waktu = 1:50;
while(true)
    if(berhenti == 0)
        fprintf(arduino,'%s','ambil data');
        disp('ambil data');
         pause(.2);

            b = fscanf(arduino, '%f %f %f')

           if length(b)== 6
             for ir = 1:50,
               if ir>1
                b = fscanf(arduino, '%f %f %f')
               end
                    tegangan_pln1(ir) = ((b(1)*5/1023)-(2.4913))/0.0095;
                    if tegangan_pln1(ir)<0,
                        tegangan_pln1(ir)=0;
                    end
                     tegangan_pln2(ir) = ((b(2)*5/1023)-(2.5818))/0.0102;
                    if tegangan_pln2(ir)<0,
```

```
                        tegangan_pln2(ir)=0;
                    end
                    tegangan_gen1(ir) = ((b(3)*5/1023)-(2.5071))/0.0091;
%(2.5538))/0.0085
                    if tegangan_gen1(ir)<0,
                        tegangan_gen1(ir)=0;
                    end
                    tegangan_gen2(ir) = ((b(4)*5/1023)-(2.5156))/0.0089;
%(2.5007))/0.0088
                    if tegangan_gen2(ir)<0,
                        tegangan_gen2(ir)=0;
                    end
                    frek_pln(ir) = ((b(5)*5/1023)-0.2685)/0.0228;
                    if frek_pln(ir)<0,
                        frek_pln(ir)=0;
                    end
                    frek_gen(ir) = ((b(6)*5/1023)-0.1720)/0.0270;
                    if frek_gen(ir) <0,
                        frek_gen(ir)=0;
                    end
            end
            %Menghaluskan grafik
            maxGen1 = max(tegangan_gen1);
            maxPln1 = max(tegangan_pln1);
            maxGen2 = max(tegangan_gen2);
            maxPln2 = max(tegangan_pln2);

            Tgen = 1/max(frek_gen);
            Tpln = 1/max(frek_pln);
            T11 = 0;
            if Tgen < Tpln,
                T11 = Tgen
            else
                T11 = Tpln;
            end
            sudutTegPln1 = 0;
            sudutTegGen1 = 0;
            sudutTegPln2 = 0;
            sudutTegGen2 = 0;
            for it = 1:49,
                if tegangan_pln1(it) < tegangan_pln1(it+1),
                    sudutTegPln1 = asind(tegangan_pln1(1)/220);
                    break;
                elseif tegangan_pln1(it) > tegangan_pln1(it+1),
                    sudutTegPln1 = 180 - asind(tegangan_pln1(1)/220);
                    break;
                end
            end
            for it = 1:49,
                if tegangan_pln2(it) < tegangan_pln2(it+1),
                    sudutTegPln2 = asind(tegangan_pln2(1)/220);
                    break;
                elseif tegangan_pln2(it) > tegangan_pln2(it+1),
                    sudutTegPln2 = 180 - asind(tegangan_pln2(1)/220);
                    break;
```

```
                end
            end
            for it = 1:49,
                if tegangan_gen1(it) < tegangan_gen1(it+1),
                    sudutTegGen1 = asind(tegangan_gen1(1)/220);
                    break;
                elseif tegangan_gen1(it) > tegangan_gen1(it+1),
                    sudutTegGen1 = 180 - asind(tegangan_gen1(1)/220);
                    break;
                end
            end
            for it = 1:49,
                if tegangan_gen2(it) < tegangan_gen2(it+1),
                    sudutTegGen2 = asind(tegangan_gen2(1)/220);
                    break;
                elseif tegangan_gen2(it) > tegangan_gen2(it+1),
                    sudutTegGen2 = 180 - asind(tegangan_gen2(1)/220);
                    break;
                end
            end
            if bro == 1;
                if (sudutTegPln1 - sudutTegGen1 < 0.9) || (sudutTegPln1 -
sudutTegGen1 < -5)

                    set(handles.MERAH_R, 'Visible', 'Off');
                else
                    set(handles.MERAH_R, 'Visible', 'on');
                end
                 if (sudutTegPln1 - sudutTegGen1 <0.9 ) || (sudutTegPln1
- sudutTegGen1 < -5)
                    set(handles.MERAH_S, 'Visible', 'Off');
                    set(handles.MERAH_T, 'Visible', 'Off');
                else
                    set(handles.MERAH_S, 'Visible', 'on');
                    set(handles.MERAH_T, 'Visible', 'on');
                end
            end

            set(handles.maks_frek_pln1, 'String', max(frek_pln));
            set(handles.maks_frek_pln2, 'String', max(frek_pln));
            set(handles.maks_frek_pln3, 'String', max(frek_pln));
            set(handles.maks_frek_gen1, 'String', max(frek_gen));
            set(handles.maks_frek_gen2, 'String', max(frek_gen));
            set(handles.maks_frek_gen3, 'String', max(frek_gen));

            set(handles.maks_teg_gen1, 'String', max(tegangan_gen1));
            set(handles.maks_teg_gen2, 'String', max(tegangan_gen2));
            set(handles.maks_teg_gen3, 'String', max(tegangan_gen2));
            set(handles.maks_teg_pln1, 'String', max(tegangan_pln1));
            set(handles.maks_teg_pln2, 'String', max(tegangan_pln2));
            set(handles.maks_teg_pln3, 'String', max(tegangan_pln2));

            xx = 0:0.0498:2*pi;
            yGen1 = (maxGen1/2) * (sin(xx - sudutTegGen1*pi/180) +1);
            yPln1 = (maxPln1/2) * (sin(xx - sudutTegPln1*pi/180) +1);
```

```
            yGen2 = (maxGen2/2) * (sin(xx - sudutTegGen2*pi/180) +1);
            yPln2 = (maxPln2/2) * (sin(xx - sudutTegPln2*pi/180) +1);

            axes(handles.axes1)
            plot((xx/6.2748)*Tgen, yGen1, 'r','LineWidth',2.5);
            hold on;
            plot((xx/6.2748)*Tpln, yPln1, 'b','LineWidth',2.5);
            hold off;
            judul = 'Sensor 1 Perubahan Tegangan dan Frekuensi terhadap
Waktu';

set(gca,'Title',text('String',judul,'Color','black','FontSize',9))
            set(get(gca,'XLabel'),'String','Data ke-','FontSize',7)
                set(gca,'XLim',[0 T11]);
            set(get(gca,'YLabel'),'String','Tegangan
(Volt)','FontSize',7)
            grid on
            axes(handles.axes4)
            %Membuat grafik Axes4
            plot((xx/6.2748)*Tgen, yGen2, 'r','LineWidth',2.5);
            hold on;
            plot((xx/6.2748)*Tpln, yPln2, 'b','LineWidth',2.5);
            hold off;
                judul = 'Sensor 2 Perubahan Tegangan dan Frekuensi
terhadap Waktu';

set(gca,'Title',text('String',judul,'Color','black','FontSize',9))
                set(get(gca,'XLabel'),'String','Data ke-','FontSize',7)
                set(gca,'XLim',[0 T11]);
                set(get(gca,'YLabel'),'String','Tegangan
(Volt)','FontSize',7)
                grid on
            %end Axes4
            axes(handles.axes5)
            %Membuat grafik Axes5
            plot((xx/6.2748)*Tgen, yGen2, 'r','LineWidth',2.5);
            hold on;
            plot((xx/6.2748)*Tpln, yPln2, 'b','LineWidth',2.5);
            hold off;
                judul = 'Sensor 3 Perubahan Tegangan dan Frekuensi
Terhadap Waktu';

set(gca,'Title',text('String',judul,'Color','black','FontSize',9))
                set(get(gca,'XLabel'),'String','Data ke-','FontSize',7)
                set(gca,'XLim',[0 T11]);
                set(get(gca,'YLabel'),'String','Tegangan
(Volt)','FontSize',7)
                grid on
            %end Axes5
            sinkronn = get(handles.SINKRONASI, 'String');
            automatic = get(handles.Automatic, 'Value');

            if (max(frek_pln)- max(frek_gen) < 5 || max(frek_pln)-
max(frek_gen) > -5) && (max(tegangan_gen1) - max(tegangan_pln1) > -5 ||
max(tegangan_gen1) - max(tegangan_pln1) > -5)
```

```
                if  max(tegangan_gen1) >= 220 && max(tegangan_pln1)>= 220
&& max(frek_pln)>= 49 && max(frek_gen) >= 49 && (sudutTegPln2 -
sudutTegGen2 <0.5)  || (sudutTegPln2 - sudutTegGen2 < -3)
                    set(handles.SINKRONASI, 'String', 'SINKRON');
                    if automatic == 1;
                   pause(2);
                     fprintf(arduino,'%s','relay 1 on');
                    end
                else
                    set(handles.SINKRONASI, 'String', 'BELUM SINKRON');
                end
            else
                set(handles.SINKRONASI, 'String', 'BELUM SINKRON');
            end

            pause(2);
        end



        elseif(berhenti == 1)

        break;
    end
end

% hObject    handle to tombol_start (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in pushbutton15.
function pushbutton15_Callback(hObject, eventdata, handles)
global berhenti;
if( berhenti == 0)
    berhenti = 1;
end
% hObject    handle to pushbutton15 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in relay1.
function relay1_Callback(hObject, eventdata, handles)
% hObject    handle to relay1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global berhenti;
global arduino;
global bro;
if( berhenti == 0)
    bro = 0;
    set(handles.cb_menutup, 'Visible', 'on');
    fprintf(arduino,'%s','relay 1 on');
```

```
     pause(1);
end


% --- Executes on button press in relay2.
function relay2_Callback(hObject, eventdata, handles)
% hObject    handle to relay2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global berhenti;
global arduino;
if( berhenti == 0)
    set(handles.cb2_on, 'Visible', 'on');
    fprintf(arduino,'%s','relay 2 on');
    disp('relay 2 on');
     pause(1);
end


% --- Executes on button press in radiobutton2.
function radiobutton2_Callback(hObject, eventdata, handles)
% hObject    handle to radiobutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of radiobutton2


% --- Executes on key press with focus on radiobutton2 and none of its
controls.
function radiobutton2_KeyPressFcn(hObject, eventdata, handles)
% hObject    handle to radiobutton2 (see GCBO)
% eventdata  structure with the following fields (see UICONTROL)
%   Key: name of the key that was pressed, in lower case
%   Character: character interpretation of the key(s) that was pressed
%   Modifier: name(s) of the modifier key(s) (i.e., control, shift)
pressed
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in Automatic.
function Automatic_Callback(hObject, eventdata, handles)
% hObject    handle to Automatic (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of Automatic
automatic = get(handles.Automatic, 'Value');
if automatic == 1
    set(handles.relay1, 'Visible', 'Off');
    %set(handles.relay2, 'Visible', 'Off');
else
    set(handles.relay1, 'Visible', 'On');
    %set(handles.relay2, 'Visible', 'On');
```

```
end


% --- Executes on button press in Relay1_off.
function Relay1_off_Callback(hObject, eventdata, handles)
% hObject    handle to Relay1_off (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global berhenti;
global arduino;
if( berhenti == 0)
    fprintf(arduino,'%s','relay 1 off');
     pause(1);
end


% --- Executes on button press in Relay2_off.
function Relay2_off_Callback(hObject, eventdata, handles)
% hObject    handle to Relay2_off (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global berhenti;
global arduino;
if( berhenti == 0)
    fprintf(arduino,'%s','relay 2 off');
     pause(1);
end

% --- Executes on button press in cb2_on.
function cb2_on_Callback(hObject, eventdata, handles)
% hObject    handle to cb2_on (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)



function edit19_Callback(hObject, eventdata, handles)
% hObject    handle to Lampu_fasaT (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Lampu_fasaT as text
%        str2double(get(hObject,'String')) returns contents of
Lampu_fasaT as a double


% --- Executes during object creation, after setting all properties.
function Lampu_fasaT_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Lampu_fasaT (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit18_Callback(hObject, eventdata, handles)
% hObject    handle to Lampu_fasaS (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Lampu_fasaS as text
%        str2double(get(hObject,'String')) returns contents of
Lampu_fasaS as a double



% --- Executes during object creation, after setting all properties.
function Lampu_fasaS_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Lampu_fasaS (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit17_Callback(hObject, eventdata, handles)
% hObject    handle to MERAH_R (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of MERAH_R as text
%        str2double(get(hObject,'String')) returns contents of MERAH_R as
a double



% --- Executes during object creation, after setting all properties.
function MERAH_R_CreateFcn(hObject, eventdata, handles)
% hObject    handle to MERAH_R (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
```

```
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

L2.2. Listing Program Mikrokontroller

```
const short sen_teg_pln1 = A0;
const short sen_teg_pln2 = A2;
const short sen_teg_pln3 = A4;
const short sen_teg_gen1 = A1;
const short sen_teg_gen2 = A3;
const short sen_teg_gen3 = A7;

const short sen_frek_pln = A6;
const short sen_frek_gen = A5;

const short relay1 = 48;
const short relay2 = 51;
bool mulai = HIGH;

bool kirimData = LOW;
String Rel1 = "";
String Rel2 = "";

int cah = -1;
int teg_pln1[50];
int teg_pln2[50];
int teg_pln3[50];
int teg_gen1[50];
int teg_gen2[50];
int teg_gen3[50];

int frek_pln[50];
int frek_gen[50];

String matData;
double waktu;
```

```
double waktuStart = 0 ;
void setup() {
  // put your setup code here, to run once:
  pinMode(relay1,OUTPUT);
  digitalWrite(relay1,HIGH);
  pinMode(relay2,OUTPUT);
  digitalWrite(relay2,HIGH);
  Serial.begin(115200);
}

void loop() {
  // put your main code here, to run repeatedly:

    if(Serial.available()>0){
        matData = Serial.readString();
        if (matData == "ambil data"){
          kirimData = HIGH;
        }
        else if (matData == "relay 1 on"){
          Rel1 = "ON";
        }
        else if (matData == "relay 1 off"){
          Rel1 = "OFF";
        }
       else if (matData == "relay 2 on"){
          Rel2 = "ON";
        }
        else if (matData == "relay 2 off"){
          Rel2 = "OFF";
        }
    }

    if(cah<50){
      cah = cah +1;
      teg_pln1[cah] = analogRead(sen_teg_pln1);
      teg_pln2[cah] = analogRead(sen_teg_pln2);
      //teg_pln3[cah] = analogRead(sen_teg_pln3);
```

```
   teg_gen1[cah] = analogRead(sen_teg_gen1);
   teg_gen2[cah] = analogRead(sen_teg_gen2);
   //teg_gen3[cah]= analogRead(sen_teg_gen3);

   frek_pln[cah] = analogRead(sen_frek_pln);
   frek_gen[cah] = analogRead(sen_frek_gen);
 }

if(Rel1== "ON"){
   digitalWrite(relay1,LOW);
   Rel1 = "";
}
if(Rel1== "OFF"){
   digitalWrite(relay1,HIGH);
   Rel1 = "";
}
if(Rel2 == "ON"){
   digitalWrite(relay2,LOW);
   Rel2 = "";
}
if(Rel2 == "OFF"){
   digitalWrite(relay2,HIGH);
   Rel2 = "";
}
if(kirimData){
     Serial.println(cah);
    for(int x=0;x<50;x++){
     Serial.print(teg_pln1[x+1]); // 1
     Serial.print(" ");
     Serial.print(teg_pln2[x+1]); // 2
     Serial.print(" ");
     //Serial.print(teg_pln3[x+1]); // 3
     //Serial.print(" ");
     Serial.print(teg_gen1[x+1]); // 4
     Serial.print(" ");
     Serial.print(teg_gen2[x+1]); // 5
     Serial.print(" ");
```

```
        //Serial.print(teg_gen3[x+1]); // 6
        //Serial.print(" ");
        Serial.print(frek_pln[x+1]); // 7
        Serial.print(" ");
        Serial.println(frek_gen[x+1]); // 8
      }
      kirimData = LOW;
      cah = -1; //reset
   }
}
```