

PEMBUATAN SUMBER ARUS AC YANG FREKUENSINYA
BISA DIATUR SECARA OTOMATIS

SKRIPSI

Sebagai salah satu syarat untuk memperoleh gelar Sarjana Sains
dalam bidang fisika

oleh:

RINDA SURYA ARIYANI

0310930047 – 93



JURUSAN FISIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS BRAWIJAYA
MALANG
2010

LEMBAR PENGESAHAN SKRIPSI
PEMBUATAN SUMBER ARUS AC YANG FREKUENSINYA
BISA DIATUR SECARA OTOMATIS

Oleh:
RINDA SURYA ARIYANI
0310930047 – 93

Setelah dipertahankan di depan Majelis Penguji pada tanggal
09 Februari 2010
dan dinyatakan memenuhi syarat untuk memperoleh gelar Sarjana
Sains dalam bidang fisika

Pembimbing I

Pembimbing II

DR. Eng. Didik R. Santoso
NIP. 196906301994021001

Drs. Johan A.E. Noor, M.Sc.
NIP. 196503251990021004

Mengetahui,
Ketua Jurusan Fisika
Fakultas MIPA Universitas Brawijaya

Adi Susilo, Phd.
NIP. 196312271991031002

EMBAR PERNYATAAN

Saya yang bertanda tangan di bawah ini :

Nama : Rinda Surya Ariyani

NIM : 0310930047 – 93

Jurusan : Fisika

Penulis skripsi berjudul :

Pembuatan Sumber Arus AC Yang Frekuensinya Bisa Diatur Secara Otomatis.

Dengan ini menyatakan bahwa :

1. Tugas Akhir ini adalah benar-benar karya saya sendiri, dan bukan hasil plagiat dari karya orang lain. Karya-karya yang tercantum dalam Daftar Pustaka TA ini, semata-mata digunakan sebagai acuan/referensi.
2. Apabila dikemudian hari diketahui bahwa isi TA saya merupakan hasil plagiat, maka saya bersedia menanggung akibat hukum dari keadaan tersebut.

Demikian pernyataan ini dibuat dengan segala kesadaran

Malang, 09 Pebruari 2010

Yang menyatakan

Rinda Surya Ariyani

NIM. 0310930047 – 93

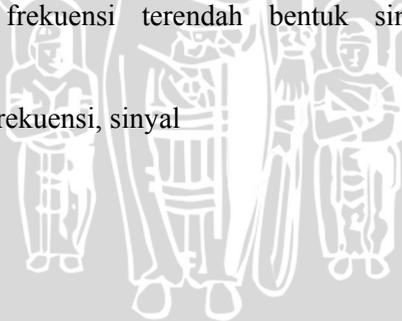
PEMBUATAN SUMBER ARUS AC YANG FREKUENSINYA BISA DIATUR SECARA OTOMATIS

ABSTRAK

Pengukuran merupakan kegiatan yang bertujuan untuk mengetahui kondisi fisis suatu objek seperti impedansi suatu bahan. Impedansi suatu bahan bisa diukur dengan menggunakan piranti spektrometer impedansi. Sumber arus AC merupakan salah satu bagian penting dalam piranti spektrometer impedansi.

Sumber arus bolak-balik (AC) terdiri dari dua modul utama yaitu modul sumber arus yang memiliki komponen utama generator sinyal XR2206 dan modul akuisisi data dan kontrol yang memiliki komponen utama mikrokontroler PIC16F877. Kedua modul ini bekerja secara terintegrasi sehingga bisa menghasilkan sinyal keluaran berupa gelombang sinusoidal yang bisa diamati pada *user interface*. Jangkauan frekuensi dari sinyal keluaran yang dihasilkan adalah 0,01 Hz sampai dengan 1 kHz. Amplitudo sinyal keluaran untuk frekuensi 10 Hz mencapai 3 volt, 1 Hz mencapai 2,5 volt dan untuk orde frekuensi yang lebih rendah hanya mencapai orde milivolt. Frekuensi sinyal ini bisa dipilih sesuai dengan kebutuhan. Bentuk sinyal keluaran pada frekuensi tinggi terlihat sangat bagus sedangkan untuk frekuensi terendah bentuk sinyalnya sudah mengalami distorsi.

Kata kunci : arus, frekuensi, sinyal



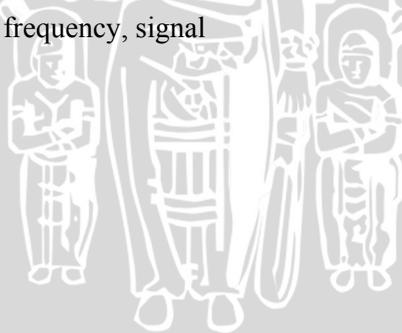
THE MAKING OF AC CURRENT SOURCE THAT HAS AUTOMATICALLY ARRANGEABLE FREQUENCY

ABSTRACT

Measurement is an activity for knowing the physics condition of an object, such as material's impedance. A material's impedance can be measured by using impedance spectrometer. AC current source is one of the principal element in impedance spectrometer.

Alternating current source consists of two principal moduls. They are current source modul, that has signal generator XR2206 as primary element, and data acquition and control modul, that has microcontroller PIC16F877 as the primary element. They work integratedly in order to produce sinusoidal wave output signal that can be perceived at user interface. The frequency range of output signal is 0,01 Hz to 1 kHz. Amplitudo's output signal at 10 Hz can reach 3 volt, at 1 Hz reach 2,5 volt an at lower frequency can reach in milivolt level only. The frequency of output signal can be selected according to user's needed. Output signal at high frequency is a good waveform, but the distortion of waveform can be seen at the lowest frequency of output signal

Key word : current, frequency, signal



KATA PENGANTAR

Segala puji dan syukur ke hadirat Allah SWT atas segala rahmat, hidayah dan karuniaNya sehingga penulis dapat menyelesaikan Tugas Akhir dan Penulisan Skripsi dengan judul “Pembuatan Sumber Arus AC Yang Frekuensinya Bisa Diatur Secara Otomatis” yang merupakan salah satu syarat untuk memperoleh gelar Sarjana Sains pada bidang fisika.

Oleh karena itu dengan segenap hati penulis ingin menyampaikan terima kasih kepada :

1. Bapak Adi Susilo, Phd. selaku ketua Jurusan Fisika Universitas Brawijaya Malang,
2. Bapak Dr. Eng, Didik R. Santoso dan Bapak Drs. Johan A.E. Noor. MSc selaku dosen pembimbing, atas arahan dan kesabaran dalam membimbing penulis dalam menyelesaikan tugas akhir ini..
3. Seluruh Dosen Penguji dan Bapak / Ibu Dosen di Jurusan Fisika Universitas Brawijaya atas ilmu yang telah diberikan selama penulis mengenyam bangku kuliah.
4. Seluruh Karyawan dan Staf di Jurusan Fisika atas bantuannya dalam melancarkan administrasi perkuliahan selama ini serta atas semangat dan dorongannya.
5. Papa, Mama, Ibu, Bapak, Suami tercinta Devid Yuliasir dan calon anakku serta adek-adekku sayang yang selalu mendoakan dan mendukung aktivitasku selama ini “ini secuil persembahanku buat cinta kalian, gak ada yang bisa mengalahkan kasih sayang kalian di dunia ini..”
6. Teman-teman Fisika 2003, atas persahabatan dan kekomplitannya, karena kalian hidupku jadi penuh warna...dan temen2 Fisika di semua Angkatan “Thanks for all memories..”
7. Teman-teman lab, Nobi, Rsy, Ephy, Baha, Fendi, Angga, dll “Terima kasih atas dukungannya”
8. Serta semua pihak yang turut membantu dan tidak mungkin penulis sebutkan satu persatu.

Penulis sangat menyadari bahwa penulisan skripsi ini masih jauh dari sempurna, oleh karena itu penulis mengharapkan kritik dan saran yang bersifat membangun demi kesempurnaan tulisan ini.

Akhirnya penulis mengharapkan semoga segala yang tertulis dalam skripsi ini bisa memberikan manfaat kepada pembaca, dapat memberikan wawasan mengenai sumber arus AC dan bisa dijadikan referensi dalam penelitian selanjutnya.

Malang, Pebruari 2010

Penulis



DAFTAR ISI

	Halaman
HALAMAN JUDUL.....	i
HALAMAN PENGESAHAN.....	ii
HALAMAN PERNYATAAN.....	iii
ABSTRAK.....	iv
ABSTRACT.....	v
KATA PENGANTAR.....	vii
DAFTAR ISI.....	ix
DAFTAR GAMBAR.....	xi
DAFTAR TABEL.....	xiii
DAFTAR LAMPIRAN.....	xv
BAB I PENDAHULUAN	
1.1 Latar Belakang.....	1
1.2 Perumusan Masalah.....	2
1.3 Batasan Masalah.....	2
1.4 Tujuan Penelitian.....	2
1.5 Manfaat Penelitian.....	2
BAB II TINJAUAN PUSTAKA	
2.1 Sumber Arus dan Sumber Tegangan.....	3
2.2 Generator Sinyal AC.....	4
2.3 Penguat Operasional (<i>OP-Amp</i>).....	5
2.4 Pengubah Tegangan ke Arus (<i>V-I Converter</i>).....	7
2.5 Motor Stepper.....	8
2.6 Mikrokontroler.....	11
2.7 Komunikasi Serial.....	13
2.8 Bahasa Pemrograman Borland Delphi.....	15
BAB III METODELOGI PENELITIAN	
3.1 Waktu dan Tempat Penelitian.....	17
3.2 Tahapan Penelitian.....	17
3.3 Perancangan Sistem Peralatan.....	19
3.3.1 Diagram blok perancangan alat.....	19
3.3.2 Perancangan perangkat keras.....	20
3.3.3 Perancangan perangkat lunak.....	30
3.4 Metode Pengujian Sistem Peralatan.....	35
3.4.1 Pengujian perangkat keras.....	35

3.4.2 Pengujian perangkat lunak.....	36
BAB IV PEMBAHASAN	
4.1 Hasil Pengujian Perangkat Keras.....	39
4.1.1 Hasil pengujian generator sinyal.....	39
4.1.2 Hasil pengujian selektor arus.....	51
4.2 Hasil Pengujian Perangkat Lunak.....	53
BAB V PENUTUP	
5.1 Kesimpulan.....	57
5.2 Saran / Rekomendasi.....	57
DAFTAR PUSTAKA	59
LAMPIRAN	61



DAFTAR GAMBAR

	Halaman
Gambar 2.1 Sumber tegangan ekuivalen.....	3
Gambar 2.2 Sumber arus ekuivalen	4
Gambar 2.3 (a) sinyal pada masukan tak membalik, (b) sinyal pada masukan membalik.....	6
Gambar 2.4 Rangkaian penyangga.....	6
Gambar 2.5 Rangkaian linier sederhana.....	7
Gambar 2.6 Pengubah tegangan ke arus aktif	8
Gambar 2.7 (a) Lilitan motor stepper unipolar dan (b) irisan melintang dari motor stepper	10
Gambar 2.8 Lilitan motor bipolar.....	10
Gambar 2.9 Lilitan motor bifiliar	10
Gambar 2.10 Pin mikrokontroler PIC16F877	12
Gambar 2.11 Pin out konektor DB9	14
Gambar 2.12 (a) Konektor DB9 <i>male</i> dan (b) Konektor DB9 <i>female</i>	14
Gambar 2.13 Pemasangan pin DB 9 untuk komunikasi serial biasa.....	15
Gambar 2.14 <i>Worksheet</i> bahasa pemrograman Borland Delphi 7	15
Gambar 3.1 Diagram alir tahapan penelitian.....	18
Gambar 3.2 Diagram blok sistem peralatan sumber arus AC .	19
Gambar 3.3 Amplitudo sinyal keluaran sebagai fungsi dari resistor, R_3 , pada pin 3	21
Gambar 3.4 Rangkaian generator sinyal dengan menggunakan XR2206	22
Gambar 3.5 Grafik nilai resistor yang dibolehkan untuk jangkauan frekuensi tertentu.....	23
Gambar 3.6 Rangkaian pengubah arus ke tegangan.....	26
Gambar 3.7 Diagram blok modul kontrol dan akuisisi data....	27
Gambar 3.8 Konfigurasi kaki pin IC HIN232	28
Gambar 3.9 Rangkaian pompa muatan pada HIN232	29
Gambar 3.10 Diagram alir perangkat lunak PIC16F877	31
Gambar 3.11 Diagram alir perangkat lunak <i>user interface</i>	34
Gambar 3.12 Rangkaian untuk pengujian kalibrasi amplitudo .	36

Gambar 4.1	Grafik perbandingan nilai frekuensi sinyal keluaran dengan frekuensi hitung pada (a) nilai kapasitor pewaktu 1 nF, (b) nilai kapasitor pewaktu 10 nF, (c) nilai kapasitor pewaktu 100 nF, (d) nilai kapasitor pewaktu 1000 nF.....	40-42
Gambar 4.2	Grafik hubungan resistor pewaktu dengan frekuensi hitung pada nilai kapasitor pewaktu 1nF.....	43
Gambar 4.3	Rangkaian pengujian hubungan frekuensi sinyal keluaran dengan resistor pewaktu.....	44
Gambar 4.4	Grafik hubungan antara R_T dengan frekuensi sinyal keluaran pada nilai kapasitor (a) 1 nF, (b) 10 nF, (c) 100 nF, (d) 1 μ F, (e) 10 μ F dan (f) 100 μ F.....	45-48
Gambar 4.5	Sinyal keluaran dari generator sinyal	48
Gambar 4.6	Grafik hubungan R_3 terhadap amplitudo sinyal pada nilai kapasitor pewaktu tertentu	50
Gambar 4.7	Bentuk sinyal keluaran pada RT sama dengan 100 Ω dan, (a) orde 1 kHz; (b) orde 100 Hz; (c) orde 10 Hz; (d) orde 1 Hz; (e) orde 0,1 Hz dan (f) orde 0,01 Hz	51
Gambar 4.8	Tampilan <i>user interface</i>	54
Gambar 4.9	Bentuk sinyal keluaran pada frekuensi 1 Hz	55



DAFTAR TABEL

	Halaman
Tabel 4.1. Perbandingan nilai frekuensi sinyal dan frekuensi hitung	39
Tabel 4.2. Konversi nilai tegangan kalibrasi ke nilai R_T pada pengujian selektor frekuensi	45
Tabel 4.3. Hasil pengujian selektor arus	52



DAFTAR LAMPIRAN

	Halaman
Lampiran 1. Data hasil pengujian perangkat keras	61
Lampiran 1.1 Pengujian hubungan frekuensi sinyal keluaran dengan resistor pewaktu	61
Lampiran 1.2 Pengujian kalibrasi amplitudo	62
Lampiran 2. <i>Source Code</i> Program	63
Lampiran 2.1 <i>Source code</i> mikrokontroller (assembly)	63
Lampiran 2.2 <i>Source code user interface</i> (delphi 7)	75
Lampiran 3. Gambar Sinyal Keluaran pada masing – masing nilai kapasitor (C_T) dan Resistor Pewaktu (R_T)	85
Lampiran 4. Data hasil pengujian perangkat lunak	95
Lampiran 5. Foto alat	97



BAB I

PENDAHULUAN

1.1. Latar Belakang

Pengukuran merupakan kegiatan yang bertujuan untuk mengetahui kondisi fisis suatu obyek. Kondisi fisis ini bisa berupa dimensi objek (seperti luas, volume) maupun sifat fisis internal obyek tersebut. Sifat fisis internal bisa berupa suhu, kekerasan, dan juga impedansi suatu bahan.

Pengukuran impedansi dari suatu bahan bisa dilakukan dengan beberapa metode. Salah satu metode yang bisa digunakan adalah dengan mencatu arus bolak-balik (AC) melalui sepasang elektrode dan mengukur beda potensial yang timbul. Apabila pengukuran beda potensial tersebut dilakukan melalui elektrode yang sama dengan elektrode pencatu, metodenya dinamakan metode pengukuran dua-terminal (2T) dan bila dilakukan sepasang elektrode lainnya dinamakan metode pengukuran empat-terminal (4T).

Salah satu alat yang bisa digunakan dalam pengukuran impedansi, terutama impedansi kompleks, adalah spektrometer impedansi. Spektrometer impedansi kompleks digunakan pada bermacam-macam aplikasi, meliputi garis transmisi dan analisis jaringan biomedik. Pengukuran spektrometer impedansi dibuat dalam domain waktu atau frekuensi.

Pada domain waktu, spektrometer impedansi ditampilkan dengan mengeksitasi sampel atau sistem untuk dianalisis dengan pulsa arus elektrik dan mengukur pulsa tegangan yang dibentuk kembali. Spektrometer impedansi pada domain frekuensi (*frequency impedance spectrometer*) melibatkan pembuatan pengukuran pada frekuensi yang berbeda, baik dengan pembersihan frekuensi (seperti pada perolehan impedansi/ penganalisis fase), atau dengan membuat pengukuran diskrit pada frekuensi yang dipilih untuk perkiraan spektrum impedansi.

Berdasarkan metode yang telah dijelaskan sebelumnya, sistem instrumentasi dari spektrometer impedansi ini dapat dibangun atas beberapa piranti pendukung yaitu pembangkit arus dan pengukur beda potensial. Oleh karena itu pembuatan sumber arus AC sangat

diperlukan sebagai pembangkit arus untuk mendukung kinerja dari spektrometer impedansi. Karena arus yang diharapkan pada sumber arus ini merupakan arus bolak-balik dengan beberapa frekuensi yang dianggap cukup rendah, maka sumber arus ini perlu dikendalikan secara otomatis dengan bantuan komputer (terprogram).

1.2. Perumusan Masalah

Rumusan masalah pada penelitian ini berdasarkan latar belakang di atas adalah:

1. Bagaimana perancangan dan pembuatan sumber arus AC yang frekuensinya bisa diatur secara otomatis?
2. Bagaimana karakteristik arus yang bisa dihasilkan dari sumber arus AC tersebut?

1.3. Batasan Masalah

Batasan masalah pada penelitian ini adalah :

1. Penelitian ini terdiri dari perancangan, pembuatan dan pengujian alat.
2. Arus yang dibangkitkan merupakan satu jenis arus bolak-balik yang merupakan sinyal sinusoidal dengan jangkauan frekuensi 10 mHz – 1 kHz.

1.4. Tujuan Penelitian

Tujuan dari penelitian ini adalah untuk membuat sebuah sumber arus AC yang frekuensinya bisa diatur sebagai piranti pendukung pada sistem instrumentasi spektrometer impedansi frekuensi ultra rendah yang memiliki jangkauan frekuensi 10 mHz – 1 kHz .

1.5. Manfaat Penelitian

Penelitian ini diharapkan dapat menghasilkan sumber arus AC yang frekuensinya bisa diatur yang dapat dijadikan sebagai piranti pendukung pada sistem instrumentasi spektrometer impedansi frekuensi rendah.

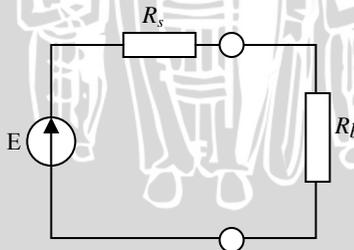
BAB II TINJAUAN PUSTAKA

2.1 Sumber Arus dan Sumber Tegangan

Arus listrik terjadi karena adanya aliran pembawa muatan listrik (yakni elektron pada logam-logam dan ion-ion pada benda cair dan gas) yang disebabkan oleh perbedaan potensial antara dua titik dalam suatu rangkaian atau tegangan listrik antara kedua titik tersebut. Tegangan listrik tersebut dapat timbul karena pada bagian tertentu dari rangkaian terdapat gaya gerak listrik (g.g.l).

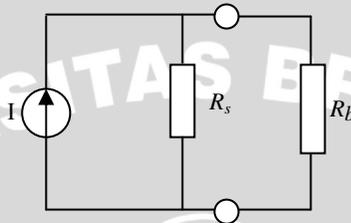
Alat-alat yang dapat memberikan gaya gerak listrik dinamakan sebagai sumber daya. Selain itu, sumber daya juga berupa generator atau pembangkit arus listrik bolak-balik maupun arus searah, akumulator dan juga alat-alat lain yang merupakan unsur-unsur yang aktif dalam rangkaian listrik. Dalam analisis rangkaian, sumber daya ini harus dinyatakan dengan pernyataan ekivalennya, yang dapat berupa sumber tegangan atau sumber arus (Margunadi, 1990).

Sumber tegangan ekivalen berupa hubungan seri antara suatu sumber tegangan yang ideal dengan nilai g.g.l yang sama dengan tegangan yang terukur antara kedua jepitannya dalam keadaan tidak terbeban (misalnya = E) dengan suatu tahanan R_s yang sama dengan tahanan dalam sumber daya tersebut. Untuk sumber tegangan searah, sumber tegangan ekivalennya dapat dilihat pada Gambar 2.1 dengan R_b merupakan tahanan beban.



Gambar 2.1 Sumber tegangan ekivalen.

Sumber arus ekivalen berupa hubungan paralel antara suatu sumber arus ideal I dengan suatu tahanan R_s yang nilainya sama dengan nilai tahanan dalam sumber daya tersebut (Gambar 2.2). Nilai dipilih sedemikian rupa, sehingga dipenuhi $IR_s = E$ dengan E sama dengan tegangan yang terukur antara dua jepitan sumber tanpa beban terhubung padanya.



Gambar 2.2 Sumber arus ekivalen.

Penggunaan rangkaian ekivalen ini tergantung pada keadaan yang ada. Pada umumnya lebih banyak digunakan sumber tegangan ekivalen. Tetapi apabila nilai R_s sangat besar dibandingkan dengan tahanan beban, maka lebih disukai untuk memandang sumber daya tersebut sebagai sumber arus (Margunadi, 1990).

Sumber arus adalah sebuah alat elektronik yang memberikan/mengirim atau menyerap arus listrik. Dalam teori rangkaian, sebuah sumber arus ideal adalah sebuah elemen rangkaian dimana arus yang melaluinya bebas dari tegangan sepanjang rangkaian tersebut (Anonimous(a), 2007).

Tidak ada sumber arus nyata yang ideal dan semuanya memiliki hambatan dalam yang terbatas (tidak satupun bisa menyediakan tegangan tak terbatas). Akan tetapi, hambatan dalam dari sumber arus fisis secara efektif dimodelkan dalam analisa rangkaian dengan mengkombinasikan sebuah resistansi tak nol secara paralel dengan sumber arus ideal.

2.2 Generator Sinyal AC

Pembangkit sinyal (generator sinyal) digunakan untuk membangkitkan sebuah sinyal, gelombang atau sumber *noise*. Fungsi dari pembangkit sinyal adalah untuk menghasilkan arus bolak-balik (AC) dengan frekuensi dan amplitudo yang diinginkan dengan

modulasi yang sesuai untuk rangkaian yang diukur atau diuji. Pembangkit sinyal dapat membangkitkan sinyal yang memiliki amplitudo yang sangat tepat. Pada kebanyakan pembangkit sinyal, pengukur keluaran (*output*) termasuk dalam peralatan untuk mengatur dan mempertahankan keluaran pada level standar dalam jangkauan frekuensi yang lebar (Anonimous(b), 2007).

Pada saat pembangkit sinyal digunakan, sinyal keluaran yang diuji dihubungkan pada rangkaian yang sedang diuji. Hasil dari pengujian sinyal kemudian ditarik melalui peralatan dengan menggunakan voltmeter elektronik atau osiloskop. Pada kebanyakan pembangkit sinyal, terdapat jaringan terkalibrasi dari resistor-resistor yang disebut sebagai *attenuator*. *Attenuator* dalam pembangkit sinyal digunakan untuk mengatur tegangan pada sinyal keluaran. Hanya *attenuator* yang terkalibrasi secara akurat bisa digunakan karena kekuatan sinyal dari pembangkit harus diatur untuk menghindari kelebihan kapasitas penerimaan sinyal dari rangkaian.

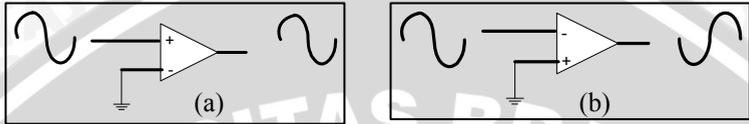
Pembangkit gelombang bisa berupa instrumen portabel, instrumen tetap, instrumen yang berbasis PC atau berupa modul. XR 2206 merupakan salah satu jenis pembangkit sinyal monolit pada rangkaian terintegrasi (*Integrated Circuit*) yang bisa menghasilkan gelombang sinus, kotak, segitiga yang berkualitas tinggi dan bentuk gelombang yang berupa pulsa yang memiliki kestabilan dan tingkat akurasi yang tinggi. Bentuk gelombang keluaran, baik amplitudo maupun frekuensinya, bisa dimodulasikan oleh tegangan external. Frekuensi operasi bisa ditentukan secara eksternal pada jangkauan 0,01 Hz sampai lebih dari 1 MHz (Exar,1997).

2.3 Penguat Operasional (*Op-Amp*)

Op-amp pada dasarnya adalah sebuah *differential amplifier* (penguat diferensial) yang memiliki dua masukan. Kedua masukan tersebut adalah masukan membalik (*inverting input*) dan masukan tak membalik (*non-inverting input*). Perbedaan dari kedua masukan ini adalah sebagai berikut (Putra, 2002):

- a. Jika sinyal melalui masukan tak membalik atau positif (+) maka keluarannya sefase dengan masukannya. Jika masukannya positif maka keluarannya juga positif (lihat Gambar 2.3 (a)).

- b. Jika sinyal melalui masukan membalik atau negatif (-) maka keluarannya berbeda fase 180° atau setengah siklus. Jika sinyalnya positif maka keluarannya menjadi negatif (dibalik) (lihat Gambar 2.3 (a)).

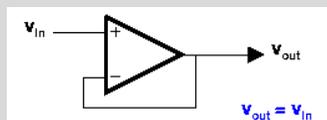


Gambar 2.3 (a) Sinyal pada masukan tak membalik, (b) sinyal pada masukan membalik.

Op-amp ideal memiliki *open loop gain* (penguatan *loop* terbuka) yang tak terhingga besarnya. Penguatan yang sebesar ini membuat op-amp menjadi tidak stabil, dan penguatannya menjadi tidak terukur (*infinite*). Untuk itulah, *negative feedback* (umpanbalik negatif) diperlukan, sehingga op-amp dapat dirangkai menjadi aplikasi dengan nilai penguatan yang terukur (*finite*). Impedansi masukan pada op-amp ideal adalah tak terhingga, sehingga arus input pada tiap masukannya adalah 0 (Aswan, 2004).

Penguat operasional bisa diaplikasikan dalam beberapa rangkaian linier. Rangkaian linier adalah rangkaian yang keluarannya merupakan fungsi dari sinyal masukan tetapi yang sudah dikuatkan atau dengan kata lain sinyal keluarannya merupakan fungsi dari sinyal masukan yang bisa diramalkan dalam batas tertentu. Contoh dari rangkaian linier adalah penguat, rangkaian penjumlah, integrator, differensiator, regulator dan filter (Wasito, 2004). Rangkaian penyangga juga merupakan salah satu aplikasi dari penguat operasional.

Rangkaian penyangga atau *buffer* digunakan sebagai antarmuka untuk mengisolasi beban dari sumber atau masukan. Rangkaian penyangga mempunyai ciri-ciri memiliki impedansi masukan yang sangat tinggi dan memiliki impedansi keluaran yang sangat rendah (Putra, 2002).



Gambar 2.4 Rangkaian penyangga

Rangkaian umpan balik pada penguat penyangga merupakan hubungan langsung dari terminal keluaran ke terminal masukan membalik. Dengan hubungan ini, maka tegangan pada masukan negatif sama dengan tegangan keluaran, dimana sinyal masukan terhubung ke terminal tak membalik yang akan membuat tegangan pada masukan positif sama dengan tegangan sumber. Jadi dalam rangkaian ini penguatan tegangan selalu sama dengan satu atau dengan kata lain tegangan keluaran selalu mengikuti masukan.

2.4 Pengubah Tegangan ke Arus (*V-I Converter*)

Pengubah tegangan ke arus mengubah besaran listrik yang membawa informasi dari tegangan ke arus. Pengubah ini bertindak sebagai rangkaian linear dengan transfer rasio $k = I_{out}/V_{in}$ (mA/V) yang mempunyai dimensi konduktivitas.



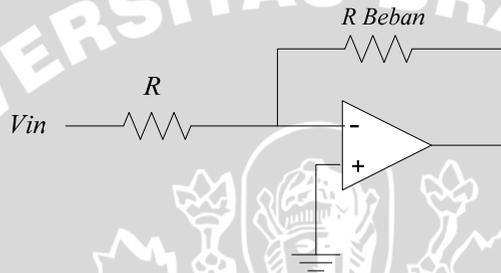
Gambar 2.5 Rangkaian linier sederhana.

Apabila sebuah tegangan V_{in} diberikan pada sebuah resistor R , arus yang sebanding akan mengalir pada rangkaian tersebut berdasarkan pada rumusan ‘tegangan menyebabkan arus’ pada hukum Ohm. Secara matematis hukum Ohm dijelaskan oleh persamaan di bawah ini.

$$I = \frac{V}{R}$$

Pada rangkaian linier sederhana yang ditunjukkan pada Gambar 2.5, resistor R menentukan besarnya arus yang mengalir melaluinya dan mengubah tegangan masukan menjadi arus keluaran yang sebanding. Oleh karena itu resistor ini bertindak sebagai pengubah tegangan ke arus (Anonymous (c), 2007).

Rangkaian pengubah tegangan ke arus dengan menggunakan resistor disebut sebagai pengubah tegangan ke arus pasif. Rangkaian ini memiliki kelemahan yaitu adanya tegangan beban yang timbul sehingga arus yang dihasilkan bukan merupakan representasi dari tegangan masukan yang diberikan. Oleh karena itu, kelemahan ini bisa diatasi dengan menggunakan bantuan op-amp dan menghasilkan rangkaian pengubah tegangan ke arus aktif (*active voltage to current converter*).



Gambar 2.6 Pengubah tegangan ke arus aktif

2.5 Motor Stepper

Secara umum, motor stepper digunakan dalam pengukuran dan pengendalian. Motor stepper merupakan aktuator elektromagnetik yang digerakkan dengan pengarahan dari sinyal digital. Hal ini berarti bahwa pada saat motor stepper diberi konfigurasi sinyal digital tertentu, motor stepper akan berputar sesuai dengan nomor spesifik dari perputarannya (Anonimous (a), 2008).

Motor stepper merupakan motor DC yang tidak memiliki komutator. Motor stepper memiliki dua bagian utama yaitu bagian yang bergerak (rotor) dan bagian yang diam (stator). Secara tipikal, motor stepper hanya mempunyai kumparan pada statornya, sedangkan pada bagian rotornya merupakan magnet permanen. Gambar 2.7 (b) menunjukkan konstruksi dasar motor stepper, dimana rotornya terletak pada bagian tengah.

Stator pada motor stepper memiliki beberapa kutub. Makin banyak kutubnya makin rumit konstruksinya. Setiap kutub memiliki lilitan yang menghasilkan medan magnet yang akan menggerakkan

rotor. Pemberian arus yang berurutan pada kutub-kutubnya menyebabkan medan magnet berputar yang akan menarik rotor ikut berputar (Prasetia, 2004).

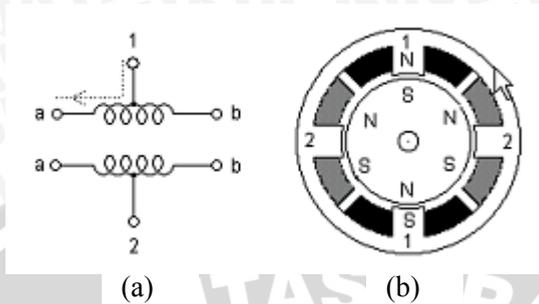
Motor stepper memiliki beberapa kelebihan dibanding motor biasa, yaitu (Anonimous (a), 2008):

- a. *Error* posisi tidak diakumulasikan. Akurasi yang tinggi dari pergerakan dimungkinkan terjadi bahkan di bawah pengendali loop terbuka.
- b. Penghematan pada sensor (sistem pengukuran) dan pengendali dimungkinkan ketika mode loop terbuka digunakan.
- c. Karena kealamian *increment* dari perintah dan pergerakan, motor stepper secara mudah beradaptasi pada aplikasi pengendali digital. Dengan kata lain, motor stepper dapat dengan mudah dikendalikan oleh mikroprosesor.
- d. Tidak ada masalah serius mengenai stabilitas, bahkan di bawah pengendalian loop terbuka.
- e. Motor stepper tanpa sikat sehingga tidak berkontribusi adanya percikan listrik.
- f. Kapasitas torsi dan kebutuhan daya bisa dioptimalkan dan tanggapannya bisa dikendalikan oleh *switch* elektronik.
- g. Konstruksi memiliki keuntungan yang nyata.

Selain itu, motor stepper juga memiliki beberapa kelemahan, yaitu (Anonimous (a), 2008):

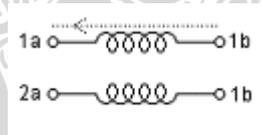
1. Motor stepper memiliki kapasitas torsi yang rendah dibanding dengan motor DC.
2. Motor stepper memiliki kecepatan yang terbatas, yaitu dibatasi oleh kapasitas torsi dan masalah pulsa yang hilang berkaitan dengan kegagalan sistem *switching* dari rangkaian *drive* (pengarah).
3. Motor stepper memiliki tingkat tegangan tinggi karena gerakan langkahnya.
4. *Error* yang besar dan osilasi bisa dihasilkan ketika sebuah pulsa hilang dibawah pengendali loop terbuka.

Ada tiga jenis motor stepper berdasarkan lilitannya, yaitu motor stepper unipolar, bipolar dan bifilar. Motor stepper unipolar dibangun dari 2 lilitan dimana masing-masing lilitan memiliki pusat lilitan (*taps*). Gambar 2.7 merupakan gambar lilitan pada motor stepper unipolar dan irisan melintang dari motor stepper.



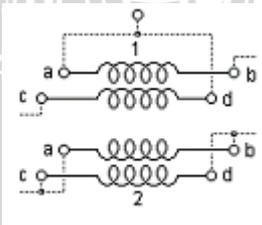
Gambar 2.7 (a) Lilitan motor stepper unipolar dan (b) irisan melintang dari motor stepper (Jones, 2004).

Motor stepper bipolar dibangun dari dua lilitan dan memiliki 4 kawat. Tidak seperti pada motor unipolar, motor bipolar tidak memiliki pusat lilitan. Keuntungan dari tidak adanya pusat lilitan ini adalah arus bisa melewati semua bagian lilitan pada satu waktu. Sebagai akibatnya, motor bipolar menghasilkan torsi yang lebih dibanding dengan motor-motor unipolar dengan ukuran yang sama (Jones, 2004).



Gambar 2.8 Lilitan motor bipolar (Jones, 2004).

Motor bifiliar memiliki rotor dan stator yang identik dengan motor bipolar, kecuali bahwa masing-masing lilitan dibangun dari 2 kawat yang diikat secara paralel satu sama lainnya. Sebagai akibatnya, motor bifiliar memiliki 8 kawat.



Gambar 2.9 Lilitan motor bifiliar (Jones, 2004).

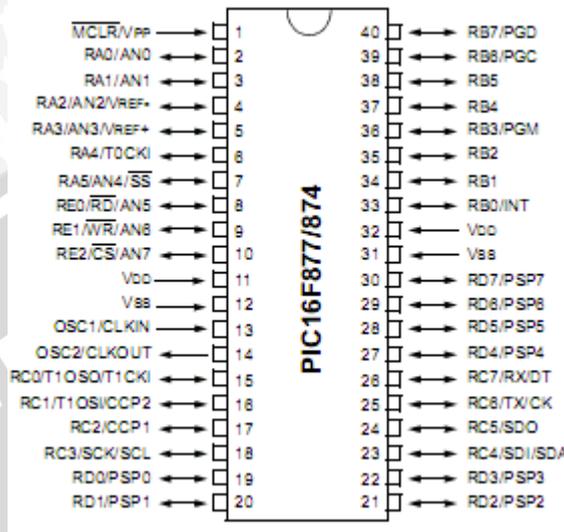
2.6 Mikrokontroler

Mikrokontroler merupakan perangkat yang telah dirancang dalam bentuk *chip* tunggal yang berfungsi untuk mengontrol sebuah sistem. Mikrokontroler dalam melakukan operasi terhadap sebuah sistem dilengkapi dengan instruksi-instruksi tertentu yang tertulis dalam deretan baris program yang layaknya disebut sebagai perangkat lunak (Atmel, 1997).

Beberapa alasan mengapa mikrokontroler dilibatkan pada sistem kontrol adalah sebagai berikut (Anonimous (b), 2008):

- i. Harga. Mikrokontroler dengan komponen-komponen rangkaian tambahan sangat lebih murah dibandingkan komputer dengan masukan/keluaran analog dan digitalnya.
- ii. Ukuran dan berat. Mikrokontroler lebih ringkas dan ringan dibanding dengan komputer.
- iii. Aplikasi yang sederhana. Jika aplikasi membutuhkan jumlah masukan/keluaran yang sangat sedikit dan kode yang relatif kecil, yang mana tidak membutuhkan jumlah memori yang luas dan tampilan LCD sederhana cukup sebagai *user interface*, sebuah mikrokontroler akan sangat sesuai untuk aplikasi ini.
- iv. Reliabilitas. Karena arsitektur lebih sederhana dibandingkan dengan komputer, mikrokontroler memiliki kemungkinan kecil untuk gagal.
- v. Kecepatan. Semua komponen dalam mikrokontroler ditempatkan pada potongan silikon tunggal. Oleh karena itu, aplikasi akan berjalan lebih cepat dibandingkan dengan yang dilakukan pada sebuah komputer.

Untuk mendukung proses berpikir, mikrokontroler memiliki beberapa bagian utama seperti CPU, ROM, RAM dan I/O. CPU (*Central Processing Unit*) berfungsi untuk membaca program yang tersimpan di dalam ROM dan melaksanakannya. ROM (*Read Only Memory*) merupakan memori yang hanya dapat dibaca saja untuk menyimpan program dari mikrokontroler itu sendiri. RAM (*Random Acces Memory*) merupakan memori yang dapat dibaca dan ditulis atau diisi berulang kali oleh pemrogram. I/O (*Input/Output*) merupakan bagian penghubung antara mikrokontroler dengan perangkat luar agar dapat berkomunikasi antara keduanya (Malik, 2003).



Gambar 2.10 Pin mikrokontroler PIC16F877 (Microchip, 2001)

Salah satu jenis mikrokontroler yang akan dipakai dalam penelitian ini adalah mikrokontroler PIC16F877. Mikrokontroler PIC16F877 beroperasi pada tegangan 5 volt. Fitur umum dari mikrokontroler ini yaitu memiliki CPU RICS dengan penampilan yang bagus, $8K \times 14$ words flash program memory dan 368×8 bytes Random Access Memory (RAM) serta mempunyai 5 port masukan/keluaran (I/O) yaitu port A, B, C, D dan E. Selain itu mikrokontroler ini memiliki fitur tambahan seperti: 8 kanal internal ADC dengan resolusi 10 bit, TIMER, Universal Synchronous Asynchronous Receiver Transmitter (USART) dan bisa diakses dengan menggunakan komunikasi paralel atau serial (Microchip, 2001).

Analog to Digital Converter (ADC) pada mikrokontroler ini berfungsi untuk mengubah besaran analog yang masuk ke mikrokontroler, melalui pin pada Port A yang telah diinisialisasi, menjadi besaran digital. Besaran digital yang didapatkan selanjutnya bisa mempermudah proses pengolahan data karena besaran digital

mudah diterjemahkan oleh mikrokontroler. Data digital ini selanjutnya juga bisa dikirimkan ke perangkat lain dari mikrokontroler.

Secara fisik, mikrokontroler PIC16F877 memiliki ukuran yang kecil dengan 40 pin dimana masing-masing pin memiliki fungsi yang bisa diubah sesuai dengan inisialisasi yang diperlukan (Gambar 2.10). Untuk bisa menjalankan sebuah mikrokontroler, diperlukan detak (*clock*) pemicu yang biasanya berasal dari oscillator luar. Besarnya frekuensi pada oscillator berkisar antara 32 kHz - 20 MHz. Oscillator dengan frekuensi tertentu dihubungkan pada mikrokontroler melalui kaki pin 13 dan 14 (Microchip, 2001).

2.7 Komunikasi Serial

Komunikasi serial adalah proses pengiriman data secara serial, yaitu data dikirim satu per satu. Jika dibandingkan dengan komunikasi paralel yang datanya dikirim secara serempak, komunikasi serial lebih lambat, akan tetapi komunikasi serial memiliki jangkauan yang lebih panjang dibanding komunikasi paralel.

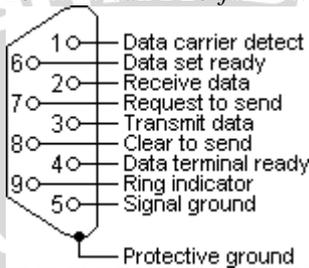
Ada dua jenis komunikasi serial, yaitu komunikasi sinkron dan komunikasi asinkron. Komunikasi serial sinkron membutuhkan detak (*clock*) pada saat pengiriman data sedangkan komunikasi serial asinkron tidak menggunakan sinyal detak pada saat pengiriman data. Penerima dan pengirim data pada komunikasi serial asinkron harus memiliki *baudrate* yang sama. *Baudrate* adalah banyaknya jumlah bit data yang dikirim tiap satu detik. Pengiriman data pada komunikasi ini juga harus diiringi oleh bit *start* dan bit *stop* sebagai penanda awal dan akhir data. Untuk menjaga integritas data bisa juga ditambahkan bit paritas.

Perangkat keras pada komunikasi serial port dibagi menjadi dua kelompok yaitu *Data Communication Equipment (DCE)* dan *Data Terminal Equipment (DTE)*. Contoh DCE adalah modem sedangkan DTE adalah terminal serial di komputer. Konektor port serial terdiri dari dua jenis yaitu konektor 25 pin atau biasa disebut dengan DB25 dan 9 pin atau biasa disebut dengan DB9 (Budiharto, 2004).

Spesifikasi elektronik dari serial port merujuk pada *Electronic Industry Association (EIA)* (Anonymous (c), 2008):

1. “Space” (logika 0) ialah tegangan antara + 3 hingga +25 V.
2. “Mark” (logika 1) ialah tegangan antara –3 hingga –25 V.
3. Daerah antara + 3V hingga –3V tidak didefinisikan/tidak terpakai
4. Tegangan open circuit tidak boleh melebihi 25 V.
5. Arus hubungan singkat tidak boleh melebihi 500 mA.

Pin out DB9 bisa memiliki fungsi yang berbeda-beda seperti yang ditunjukkan oleh Gambar 2.13. Pada aplikasinya, konektor DB9 ini terdiri dari DB9 *male* dan DB9 *female*.



Gambar 2.11 Pin out konektor DB9 (Bies, 2008).



(a)

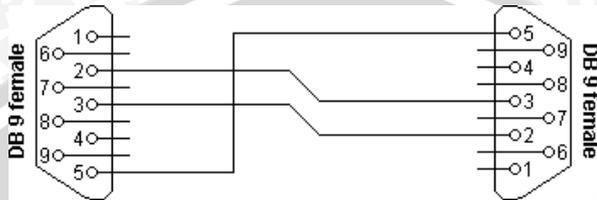
(b)

Gambar 2.12 (a) Konektor DB9 *male* dan (b) Konektor DB9 *female* (Anonimous (d), 2008).

Pin DB9 *Female* (betina) dengan DB9 *Male* (jantan) memiliki kesamaan fungsi dari masing-masing pin tetapi perbedaan terjadi pada saat pemasangan keduanya. Untuk komunikasi serial biasa maka pemasangan pin outnya adalah sebagai berikut (Anonimous (c), 2008):

- Pin TxD *male* ke pin RxD *female*
- Pin RxD *male* dihubungkan ke pin TxD *female*
- RTS dan CTS dihubung singkat

- DSR dan DTR dihubungkan singkat
- GND *male* dihubungkan ke GND *female*

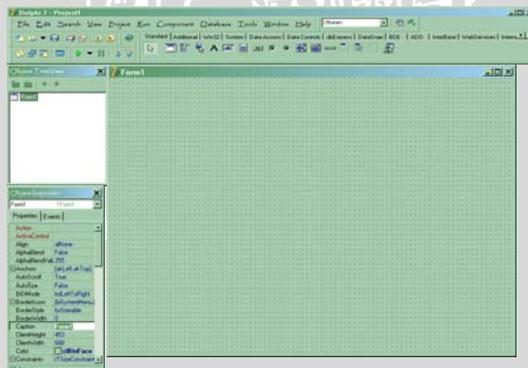


Gambar 2.13 Pemasangan pin DB 9 untuk komunikasi serial biasa (Bies, 2008).

2.8 Bahasa Pemrograman Borland Delphi

Borland Delphi atau yang biasa disebut sebagai delphi, merupakan sarana pemrograman aplikasi visual. Bahasa pemrograman yang digunakan adalah bahasa pemrograman pascal atau biasa disebut juga bahasa pemrograman Delphi. Delphi diluncurkan pertama kali pada tahun 1995 dan dirancang untuk beroperasi di bawah sistem windows (Anonimous, 2003).

Delphi menyediakan fasilitas untuk membuat aplikasi dengan antar muka visual secara mudah dan dapat memberikan hasil yang memuaskan. Gambar 2.14 menunjukkan contoh bidang kerja (*worksheet*) pemrograman Delphi 7.



Gambar 2.14 *Worksheet* bahasa pemrograman Borland Delphi 7.

Delphi memiliki sarana yang tangguh untuk pembuatan aplikasi, mulai dari sarana untuk pembuatan form, menu, toolbar hingga kemampuan untuk menangani pengelolaan basis data yang besar. Kelebihan-kelebihan yang dimiliki Delphi antara lain karena pada Delphi, form dan komponen-komponennya dapat dipakai ulang dan dikembangkan, mampu mengakses VBX, tersedia template aplikasi dan template form, memiliki lingkungan pengembangan visual yang dapat diatur sesuai kebutuhan, menghasilkan file kompilasi yang berjalan lebih cepat, serta kemampuan data dari bermacam-macam format (Anonimous, 2003).

Delphi juga bisa dilengkapi dengan penambahan komponen lain yang tidak terdapat pada komponen standar pada penginstalan awal program Delphi. Sebagai contoh komponen Comport pada CportLib yang berisi komponen-komponen untuk menghubungkan program delphi dengan perangkat keras luar. Keberadaan komponen Comport ini menambah kegunaan bahasa pemrograman Delphi dalam bidang instrumentasi dan pengukuran, yaitu sebagai *user interface* dari suatu instrumen.



BAB III

METODOLOGI PENELITIAN

3.1 Waktu dan Tempat Penelitian

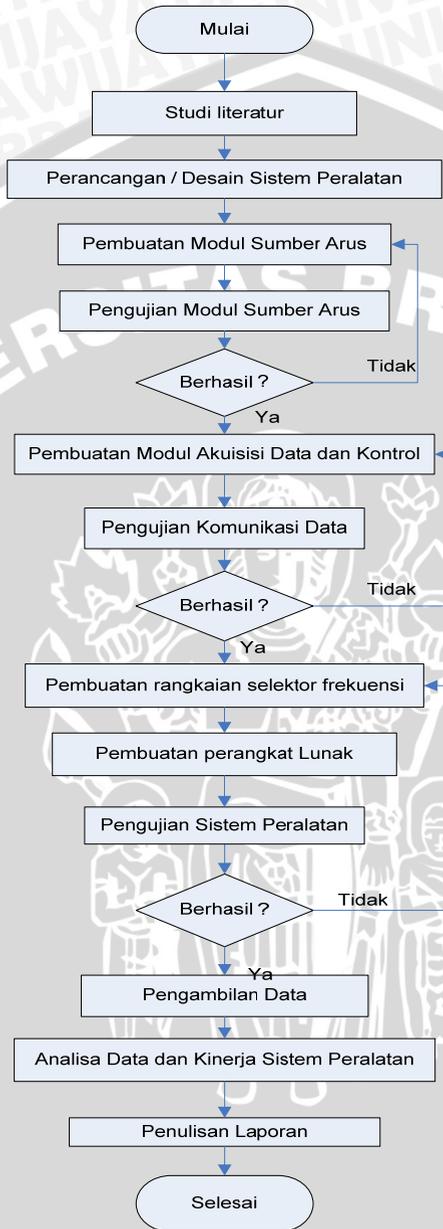
Penelitian ini dilaksanakan di Laboratorium Instrumentasi, Jurusan Fisika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Brawijaya. Penelitian ini dilaksanakan pada bulan Mei - September 2007 kemudian dilanjutkan pada bulan Agustus 2008 - Juli 2009.

3.2 Tahapan Penelitian

Tahapan penelitian yang dikerjakan adalah sebagai berikut :

1. Studi literatur.
2. Pembuatan rancangan/desain sistem peralatan.
3. Implementasi pembuatan sistem peralatan.
4. Pengujian sistem peralatan.
5. Percobaan pengambilan data dengan sistem peralatan yang dibuat.
6. Proses analisis kinerja peranti dan data yang diperoleh.
7. Pembuatan laporan.

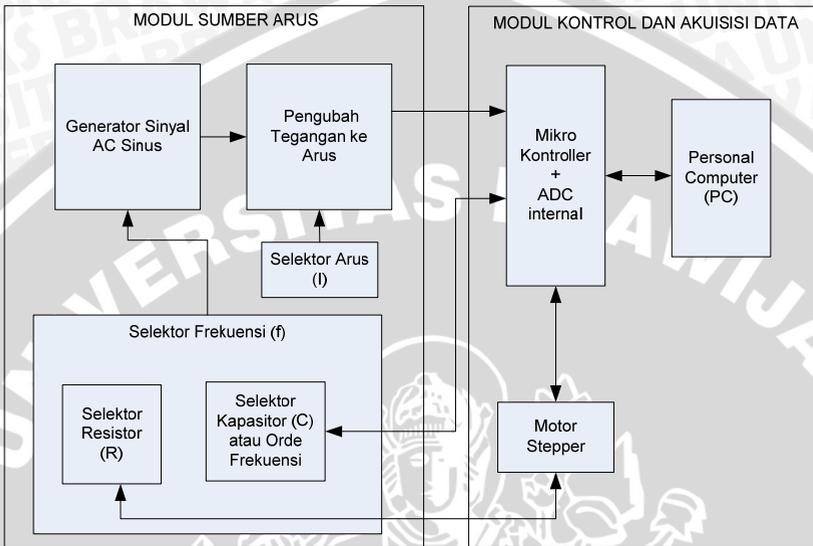
Tahapan penelitian ini tidak hanya bersifat terus menerus dari satu tahap ke tahap berikutnya. Ada proses evaluasi yang dilakukan agar sistem peralatan yang dibuat sesuai dengan yang diharapkan. Diagram alir tahapan penelitian yang terdapat pada Gambar 3.1 menyatakan apabila sistem peralatan masih belum dianggap berhasil, maka ditinjau kembali tahapan pembuatan alat.



Gambar 3.1 Diagram alir tahapan penelitian.

3.3 Perancangan Sistem Peralatan

3.3.1 Diagram blok perancangan alat



Gambar 3.2 Diagram blok sistem peralatan sumber arus AC.

Sistem peralatan sumber arus AC terdiri dari dua modul utama, yaitu :

1. Modul Sumber Arus.

Modul sumber arus adalah bagian dari sumber arus AC yang berfungsi untuk membangkitkan arus yang nantinya digunakan dalam pengukuran impedansi sistem fisis. Arus yang akan dihasilkan berupa arus sinusoidal dengan frekuensi tertentu yang bisa diubah-ubah dengan selektor frekuensi.

Karena sinyal yang dihasilkan oleh generator berupa tegangan maka diperlukan sebuah pengubah tegangan ke arus. Besarnya arus juga bisa diubah-ubah dengan menggunakan selektor arus yang dirangkaikan sebelum pengubah tersebut.

2. Modul Akuisisi Data dan Kontrol.

Modul akuisisi data dan kontrol berfungsi untuk mengendalikan kinerja keseluruhan sistem dan juga untuk akuisisi data. Pengendalian kinerja sistem yang dimaksud adalah modul ini berfungsi sebagai pengendali selektor frekuensi arus

sesuai dengan yang diinginkan dan juga sebagai pengendali keluar masuknya data dari modul sumber arus menuju ke mikrokontroler dan selanjutnya diteruskan ke PC.

3.3.2 Perancangan perangkat keras

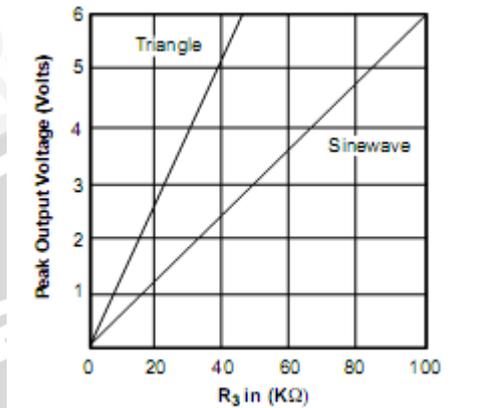
Seperti yang telah dijelaskan sebelumnya, sistem peralatan sumber arus AC memiliki dua bagian yaitu modul sumber arus dan modul kontrol dan akuisisi data. Modul-modul tersebut juga dibangun dari perangkat keras (*hardware*) dan perangkat lunak (*software*). Sub Bab ini akan menjelaskan tentang perangkat keras yang membangun kedua modul tersebut.

Perangkat keras dari modul sumber arus terdiri dari dua bagian yaitu generator sinyal AC sinus dan pengubah tegangan ke arus. Baik generator sinyal maupun pengubah tegangan ke arus merupakan kesatuan dari beberapa komponen sehingga bisa memberikan hasil dan fungsi yang diinginkan.

Generator sinyal pada alat ini menggunakan IC XR2206. XR2206 memiliki 14 pin dan dibangun dari empat bagian utama yang memiliki fungsi yang berbeda-beda. Bagian-bagian tersebut yaitu osilator yang diatur oleh tegangan (*Voltage-controlled Oscillator, VCO*), sebuah pengali dan pembentuk sinus analog (*analog multiplier and sine-shaper*), sebuah penguat penyangga dan sejumlah *switch* arus.

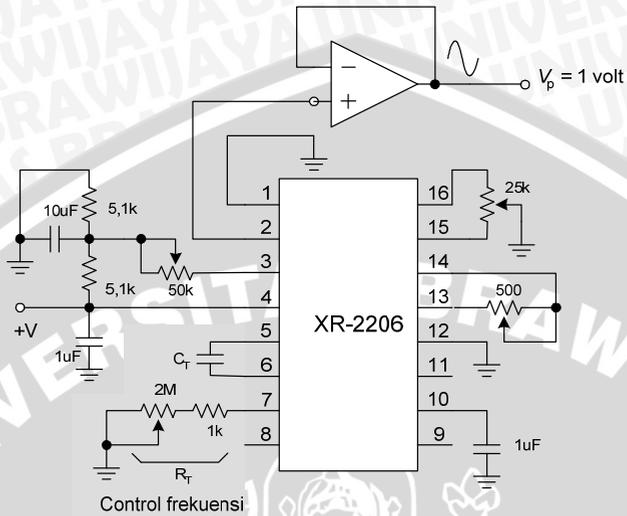
Rangkaian generator sinyal dapat dilihat pada Gambar 3.4. Pin 2 digunakan untuk sinyal keluaran yang berbentuk sinusoidal dan gelombang segitiga, sedangkan untuk keluaran gelombang kotak didapatkan dari pin 11. Sinyal keluaran sinus dirangkai dengan sebuah penguat penyangga eksternal.

Besarnya amplitudo atau level dari sinyal keluaran XR2206 (pin 2) dapat diatur dengan mengubah besarnya hambatan pada potensiometer yang terhubung pada pin 3. Besarnya level tegangan yang bisa dikeluarkan kira-kira sama dengan bias DC pada pin 3. Pin 3 dibiarkan dengan nilai tegangan antara $V+$ dan ground, untuk menghasilkan level keluaran DC sebesar $V+/2$.



Gambar 3.3 Amplitudo sinyal keluaran sebagai fungsi dari resistor, R_3 , pada pin 3 (Exar, 1997).

Amplitudo keluaran maksimum berbanding lurus dengan resistor eksternal (potensiometer 50 k Ω) yang terhubung pada pin 3. Gambar 3.3 menunjukkan bahwa setiap kenaikan 1 k Ω resistor akan menyebabkan kenaikan amplitudo sinyal keluaran sebesar 60 mV untuk keluaran gelombang sinus. R_3 yang digunakan pada modul ini adalah 50 k Ω , sehingga nilai maksimum amplitudo sinyal keluaran yang bisa dihasilkan adalah 3000 mV atau 3 volt.



Gambar 3.4 Rangkaian generator sinyal dengan menggunakan XR2206.

Untuk membangkitkan sinyal sinusoidal bisa dilakukan dengan menggunakan pengatur eksternal ataupun tidak. Pengaturan yang tidak menggunakan pengatur eksternal menyebabkan sinyal keluaran memiliki distorsi kurang dari 2,5%. Dengan menggunakan pengatur eksternal, distorsi bisa dikurangi hingga mencapai -0,5% (Exar, 1997).

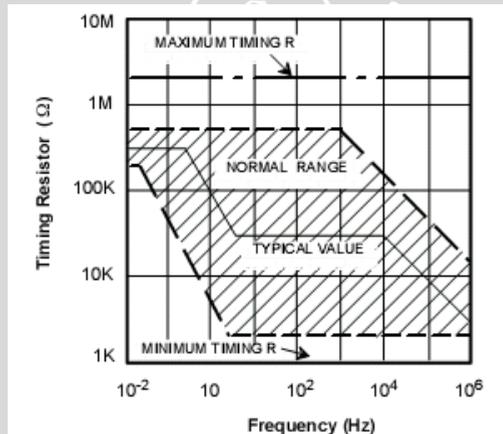
Gambar 3.4 merupakan rangkaian sinyal generator dengan menggunakan pengatur eksternal, yaitu menghubungkan kaki pin nomor 13, 15 dan 16 dengan resistor variabel atau potensiometer. Potensiometer yang dihubungkan dengan pin 13 mengatur resistor pembentuk sinus dan potensiometer yang terhubung pada pin 15 dan 16 mengatur kesimetrian bentuk dari gelombang sinus tersebut. Cara pengaturan pengurangan distorsi dengan menggunakan potensiometer-potensiometer ini adalah sebagai berikut :

1. Potensiometer pada kaki pin 15-16 pada titik tengah, kemudian potensiometer pada kaki pin 13 diatur untuk distorsi minimum.
2. Dengan kondisi potensiometer pada kaki pin 13 tetap, potensiometer pada kaki pin 15-16 diatur untuk memperoleh pengurangan distorsi berikutnya.

Frekuensi operasi dari sinyal keluaran, f_0 , ditentukan dengan kapasitor pewaktu eksternal, C_T , yang terhubung pin 5 dan 6, dan oleh resistor pewaktu, R_T , yang dihubungkan ke pin 7. Besarnya frekuensi diberikan oleh persamaan berikut :

$$f_0 = \frac{1}{R_T C_T} \text{ Hz}$$

Besarnya frekuensi bisa diubah dengan nilai R_T atau C_T yang berbeda-beda, untuk jangkauan frekuensi yang telah ditentukan (lihat Gambar 3.5). Kestabilan temperatur bisa optimum untuk nilai resistor antara 4 k Ω dan 200 k Ω . Nilai C yang direkomendasikan adalah dari 100 pF ke 100 μ F.



Gambar 3.5 Grafik nilai resistor yang dibolehkan untuk jangkauan frekuensi tertentu (Exar, 1997)

Untuk memperoleh nilai frekuensi yang berbeda-beda dari generator frekuensi ini, kapasitor pewaktu (C_T) dan resistor pewaktu (R_T) dirangkaikan sedemikian rupa sehingga memiliki nilai kapasitansi dan resistansi lebih dari satu jenis sehingga bisa dikonfigurasi. Kapasitor-kapasitor pewaktu yang digunakan sebesar : 1 nF, 10 nF, 100 nF, 1 μ F, 10 μ F dan 100 μ F. Untuk resistor pewaktu digunakan potensiometer 1 M Ω yang memiliki nilai resistansi berkisar antara 0 – 1 M Ω .

Pemilihan kapasitor dilakukan dengan perantara relai yang digerakkan oleh rangkaian transistor. Dengan menggunakan relai maka kontak yang terhubung dengan kapasitor pewaktu bisa dikendalikan dengan pemberian arus yang dikendalikan oleh logika “0” atau “1” yang dikirim dari program. Logika “0” mewakili arus rendah dan logika “1” mewakili arus tinggi. Rangkaian transistor berfungsi sebagai penggerak dimana bisa mempertahankan nilai arus dari sumber, dengan kata lain arus tidak mengalami penurunan nilai.

Relai merupakan komponen elektronik yang terdiri dari sebuah kumparan dengan sebuah inti, yang bila dialiri arus, menjadi magnetis dan yang menutup (kontak penutup) atau memutuskan (kontak pemutus) kontak-kontak apabila dialiri arus. Relai yang digunakan pada peralatan ini adalah relai dengan kontak *normally open*, yaitu relai yang jika tidak dialiri arus maka ia akan terbuka dan akan menutup jika dialiri arus. Dengan demikian, untuk memilih nilai kapasitor yang diinginkan cukup dengan memberikan logika “1” pada port yang terhubung dengan kapasitor tersebut.

Pemilihan resistor pewaktu dilakukan dengan cara yang berbeda seperti pemilihan kapasitor pewaktu. Karena resistor pewaktu hanya menggunakan sebuah potensiometer, maka pemilihan nilai resistor dilakukan dengan cara memutar potensiometer hingga menghasilkan nilai resistansi yang diinginkan.

Sinyal keluaran yang dihasilkan oleh modul ini harus memiliki nilai frekuensi tertentu. Untuk itu, nilai resistansi dan kapasitansi dari resistor pewaktu dan kapasitor pewaktu juga harus tertentu. Jika pada penjelasan sebelumnya nilai kapasitor pewaktu sudah dispesifikasikan maka untuk nilai resistansi juga harus dispesifikasikan.

Banyaknya nilai resistansi pada potensiometer menimbulkan sedikit kesulitan dalam menghasilkan nilai resistansi spesifik yang diinginkan, apalagi jika potensiometer diputar dengan cara manual. Nilai resistansi yang dihasilkan harus lebih dulu diukur dengan bantuan pengukur resistansi seperti ohmmeter. Hal ini sama sekali tidak efektif. Untuk itu pada penentuan nilai resistansi pewaktu ini dilakukan dengan bantuan motor stepper. Motor stepper ini memungkinkan pemilihan nilai resistansi yang lebih tepat, bisa dikontrol dan lebih efektif.

Motor stepper, seperti yang telah dijelaskan pada bab sebelumnya, pergerakannya dikendalikan oleh konfigurasi bit yang dikirimkan padanya. Pada modul ini, konfigurasi bit dikirimkan dari port I/O mikrokontroler. Hal ini akan dijelaskan pada bagian berikutnya.

Pengontrolan dari nilai resistansi ini juga dilakukan dengan merangkaikan potensiometer untuk resistor pewaktu dengan potensiometer 100 kΩ, selanjutnya dinamakan potensiometer pengendali. Potensiometer pengendali dirangkaikan dengan tegangan sumber sebesar 5 volt. Pemilihan nilai resistansi pada potensiometer 100 kΩ dan potensiometer 1 MΩ yang dibantu oleh motor stepper selalu menunjukkan posisi yang sama. Sehingga besarnya nilai resistansi yang dihasilkan keduanya secara ideal berbanding 1:10 untuk resistansi potensiometer 100 kΩ terhadap potensiometer 1 MΩ.

Potensiometer 100 kΩ berguna sebagai pembagi tegangan yang terhubung padanya. Pada posisi tertentu itu pula, keluaran yang terhubung dengan tegangan yang sebelumnya melewati potensiometer ini menunjukkan nilai tegangan tertentu. Nilai tegangan keluaran inilah yang nantinya menjadi masukan ke mikrokontroler untuk pengaturan perputaran motor stepper selanjutnya sehingga berada pada posisi nilai resistor pewaktu yang diinginkan.

Besarnya tegangan keluaran yang dihasilkan (V_o), menurut rumusan pembagi tegangan adalah sebagai berikut :

$$V_o = \frac{R_x}{R_{pot}} \times V_s$$

Dimana :

R_x = Besarnya resistansi yang dihasilkan pada posisi potensiometer 100 kΩ.

R_{pot} = Besarnya Resistansi maksimum potensiometer 100 kΩ.

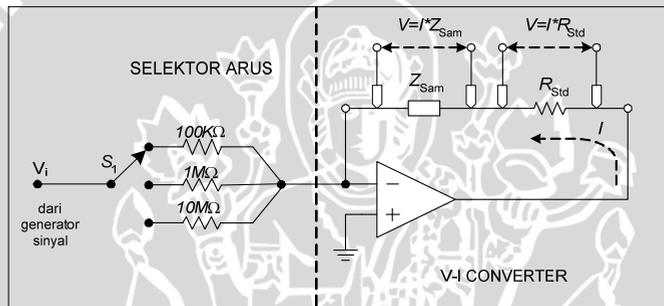
V_s = Tegangan sumber, yaitu sebesar 5 volt.

Bagian akhir dari generator sinyal ini adalah penguat penyangga. Penguat penyangga berfungsi untuk menyetabilkan amplitudo sinyal yang masuk padanya. Keluaran dari keseluruhan bagian ini masih

berupa tegangan. Untuk itu diperlukan rangkaian pengubah tegangan ke arus agar didapat besaran arus yang tentunya berupa sinyal sinusoidal.

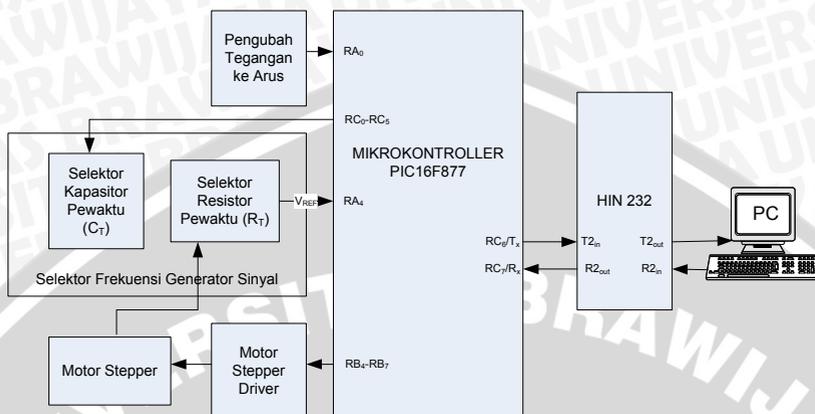
Bagian awal dari pengubah tegangan ke arus adalah selektor arus (lihat Gambar 3.6). Selektor ini digunakan untuk memilih besarnya arus yang akan digunakan untuk pengukuran impedansi. Sebelum menuju ke pengubah tegangan ke arus, sinyal yang dikeluarkan oleh generator sinyal masih berupa tegangan.

Pada Gambar 3.6. selektor arus dipilih dengan menghubungkan sinyal keluaran ke resistor dengan nilai $100\text{ k}\Omega$, $1\text{ M}\Omega$ atau $10\text{ M}\Omega$. Pemilihan arus ini dilakukan secara manual dengan menggunakan *jumper* sebagai penghubung.



Gambar 3.6 Rangkaian pengubah arus ke tegangan.

Perangkat keras modul kontrol dan akuisisi data terdiri dari mikrokontroler PIC16F877, motor stepper dan pemicunya (*driver*), serta komponen untuk komunikasi serial. Gambar 3.7 menggambarkan hubungan antar komponen yang membentuk modul kontrol dan akuisisi data. Pada modul ini, mikrokontroler PIC16F877 merupakan komponen utama yang mengatur semua kinerja modul.



Gambar 3.7 Diagram blok modul kontrol dan akuisisi data.

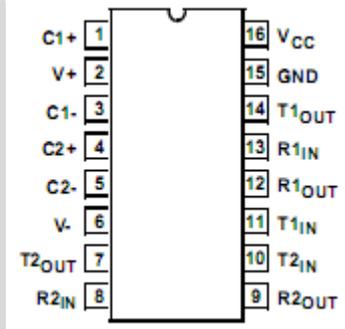
Seperti yang disebutkan pada bab sebelumnya, mikrokontroler PIC16F877 terdiri dari 40 kaki pin dengan fungsi yang berbeda-beda dan pada modul ini tidak semuanya digunakan. Kaki pin 1 merupakan pin $\overline{\text{MCLR}}$ dan kaki pin 13 dan 14 untuk masukan osilator eksternal. Kaki pin 11 dan 30 untuk masukan tegangan positif dan kaki pin 12 dan 31 untuk dihubungkan ke ground. Kaki pin lainnya digunakan untuk port masukan dan keluaran (lihat Gambar 2.10).

Pin $\overline{\text{MCLR}}$ atau pin nomor satu berfungsi untuk mengembalikan kondisi mikrokontroler pada keadaan semula (*reset*) pada saat diberi logika rendah. Pemberian logika rendah dilakukan oleh switch luar yang dihubungkan pada kaki pin ini. Menekan switch ini berarti me-reset kinerja mikrokontroler.

Osilator eksternal berguna untuk menghasilkan detak yang berperan penting dalam kinerja mikrokontroler. Mikrokontroler PIC16F877 ini bisa digerakkan dengan empat mode osilator yaitu *Low Power Crystal (LP)*, *Crystal/Resonator (XT)*, *High Speed Crystal/Resonator (HS)* dan *Resistor/Capacitor (RC)*. Keempat mode osilator ini memiliki jangkauan frekuensi yang berbeda-beda. Pada modul ini, osilator yang digunakan adalah mode High Speed crystal/resonator (HS) tepatnya osilator berfrekuensi 18,432 MHz.

Jumlah kaki pin untuk port masukan/keluaran mikrokontroler adalah sebanyak 33 buah yang secara garis besar dikelompokkan dalam lima port yaitu port A (6 kaki pin), port B (8 kaki pin), port C (8 kaki pin), port D (8 kaki pin) dan port E (3 kaki pin). Pada modul ini, port A dan port E, tepatnya Port A₀,A₁,A₂,A₃,A₅,E₀,E₁ dan E₂ digunakan sebagai masukan analog yang berasal dari modul sumber arus. Port A₀ digunakan sebagai masukan analog dari modul sumber arus. Port A₅ digunakan sebagai masukan analog untuk tegangan referensi yang digunakan dalam pengendalian resistor pewaktu. Port A₄ tidak digunakan dalam modul ini karena port ini berfungsi sebagai masukan *clock* (detak) untuk pengaturan TIMER0 dan sistem peralatan ini tidak menggunakan TIMER0. Port B yang digunakan dalam modul ini adalah port B₄ sampai port B₇. Empat port B ini dihubungkan pada motor stepper *driver* untuk mengendalikan perputaran motor stepper yang digunakan sebagai pengatur resistor pewaktu. Port C₀ sampai port C₅ dihubungkan pada pengatur kapasitor pewaktu pada modul sumber arus. Port C₆ dan C₇ merupakan pin untuk komunikasi serial. Port C₆ diset sebagai keluaran karena untuk transmisi data dan port C₇ diset sebagai masukan karena untuk penerimaan data.

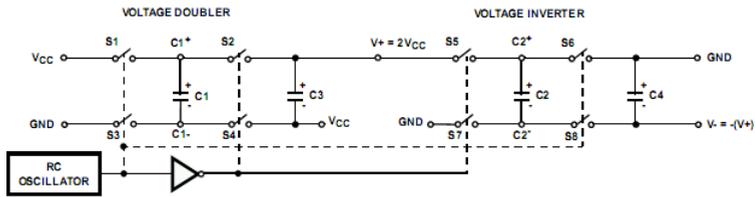
Port C₆ dan C₇ yang digunakan sebagai komunikasi serial ini tidak langsung dihubungkan ke PC melalui kabel serial DB9. Port ini dihubungkan dulu ke IC HIN232 untuk menyetabilkan tegangan.



Gambar 3.8 Konfigurasi kaki pin IC HIN232 (Intersil, 2005).

IC HIN232 merupakan keluarga IC RS232 yang banyak digunakan dalam komunikasi serial. HIN232 ini bekerja pada tegangan masukan 5 volt. Selain memiliki bagian untuk transmisi

dan penerimaan, IC ini juga memiliki bagian pompa muatan. Pompa muatan ini berfungsi untuk menghasilkan tegangan sebesar ± 10 volt dengan tegangan masukan maksimal 5 volt.



Gambar 3.9 Rangkaian pompa muatan pada HIN232 (Intersil, 2005).

Sebuah rangkaian ekuivalen dari pompa muatan digambarkan pada Gambar 3.9. Pompa muatan terdiri dari 2 bagian, yaitu: pengganda tegangan dan pembalik tegangan. Masing-masing bagian diarahkan oleh dua fase detak yang dibangkitkan secara internal untuk membangkitkan $+10$ V dan -10 V. Frekuensi detak nominal adalah sebesar 16 KHz. Selama fase satu dari clock, $C1$ dimuati oleh V_{CC} . Selama fase dua, tegangan pada $C1$ ditambahkan ke V_{CC} , dengan memproduksi sinyal yang melewati $C3$ sama dengan dua kali V_{CC} . Selama fase satu, $C2$ juga diberi muatan ke $2V_{CC}$, dan kemudian selama fase dua ia dibalik (*inverting*) untuk menghasilkan sinyal yang melewati $C4$ sama dengan $-2V_{CC}$. Pompa muatan menerima tegangan masukan sampai 5,5 V. Hambatan keluaran dari bagian pengganda tegangan ($V+$) mendekati 200Ω , dan impedansi keluaran dari bagian pembalik tegangan ($V-$) mendekati 450Ω . Akan tetapi, sebuah aplikasi tipikal menggunakan kapasitor $1 \mu\text{F}$ untuk $C1$ - $C4$ nilainya tidak kritis. Dengan menaikkan nilai dari $C1$ dan $C2$ akan merendahkan hambatan keluaran dari pengganda tegangan dan pembalik tegangan. Dengan meningkatkan nilai dari kapasitor reservoir, $C3$ dan $C4$, semakin merendahkan riak dari *supply* $V+$ dan $V-$. Besarnya $C1$, $C2$, $C3$ dan $C4$ pada modul ini adalah $10 \mu\text{F}$.

Perangkat keras lain yang digunakan dalam modul kontrol ini adalah motor stepper serta pemuncunya (*driver*). Fungsi utama dari motor stepper ini adalah untuk mengendalikan atau lebih tepatnya untuk memilih nilai resistor pewaktu dalam modul sinyal generator.

Motor stepper yang digunakan adalah motor stepper unipolar dengan tegangan masukan 12 volt. Pemicu yang digunakan adalah

HERARI HM006 dengan IC pemicu ULN2003. Masukan dari pemicu ini dihubungkan pada port-port mikrokontroler yang memberikan konfigurasi bit yang bisa menentukan perputaran posisi motor stepper. Keluarannya terhubung pada motor stepper.

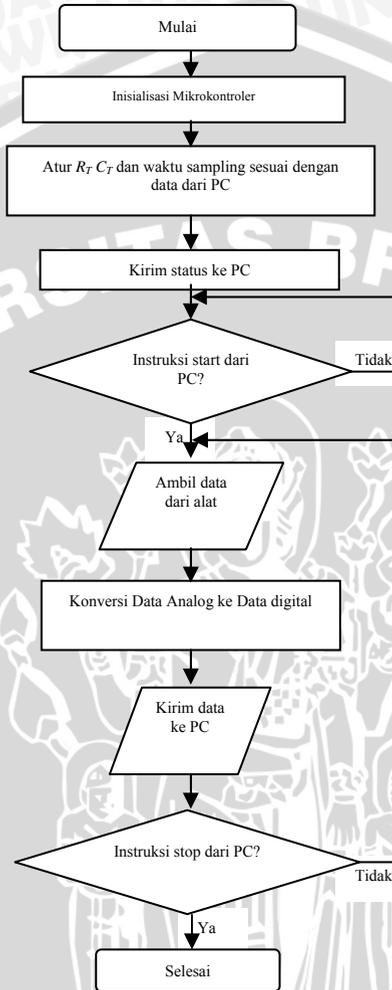
Motor stepper dihubungkan dengan potensiometer stereo yang memiliki dua buah potensiometer. Potensiometer yang pertama difungsikan sebagai resistor variabel yang terhubung dengan XR2206 dan potensiometer yang lainnya (potensiometer pengendali) dihubungkan dengan port masukan analog (Port A₅) pada mikrokontroler untuk pengendalian posisi. Nilai yang diterima oleh port ini memberikan informasi apakah posisi potensiometer pertama sudah sesuai dengan yang diinginkan. Apabila belum sesuai maka mikrokontroler mengendalikan perputaran motor stepper lagi melalui port B₄-B₇ sampai posisi dari potensiometer pertama sesuai dengan yang diinginkan.

3.3.3 Perancangan perangkat lunak

Perangkat lunak untuk sistem peralatan ini hanya terdapat pada modul kontrol dan akuisisi data. Perangkat lunak ini digunakan sebagai pendukung kinerja mikrokontroler PIC16F877 dan juga sebagai *user interface* pada PC.

Perancangan perangkat lunak pada mikrokontroler PIC16F877 menggunakan bahasa assembly (bahasa rakitan). Bahasa assembly merupakan bahasa pemrograman komputer tingkat rendah dimana bahasa ini merupakan notasi untuk bahasa mesin yang dapat dibaca oleh manusia. Bahasa assembly ini berbeda-beda sesuai dengan arsitektur mesin yang digunakan. Penulisan instruksi bahasa pemrograman pada PIC16F877 ini ada sebanyak 35 instruksi. Instruksi-instruksi tersebut ditujukan pada register-register yang terdapat pada mikrokontroler, baik register khusus maupun register tambahan.

Perancangan perangkat lunak ini berperan penting dalam kinerja keseluruhan sistem. Pengaturan, pengambilan data bahkan pengiriman data diatur dalam perangkat lunak. Diagram alir perangkat lunak yang digunakan pada sistem peralatan ini dapat dilihat pada Gambar 3.10.



Gambar 3.10 Diagram alir perangkat lunak PIC16F877.

Proses inisialisasi mikrokontroler adalah proses untuk mengatur fungsi dan peran dari port masukan-keluaran dan register-register tertentu pada mikrokontroler. Pada proses inisialisasi, dilakukan pengaturan pada port, sistem komunikasi, pemakaian timer dan pengaturan ADC internal. Selain itu, proses inisialisasi juga

merupakan proses yang menentukan kondisi awal mikrokontroler sebelum melakukan instruksi berikutnya. Sehingga pada proses ini juga dilakukan penghapusan register-register tambahan dan juga semua interupsi. Penentuan lokasi RAM register-register tambahan pada mikrokontroler ini dilakukan di luar program utama sebelum proses inisialisasi.

Pada modul ini, proses inisialisasi mengatur port B, Port C₀-C₅ dan port D sebagai keluaran. Fungsi port B sebagai keluaran ini bisa diubah kemudian sesuai dengan kebutuhan kinerja sistem peralatan. TIMER1 digunakan dengan perbandingan skala (*prescaler*) 1:1 dan tidak menggunakan osilator khusus. Untuk tiap siklus atau instruksi dari mikrokontroler diperlukan waktu sebesar $F_{osc}/4$. Proses komunikasi mikrokontroler dan perangkat luar diatur melalui register TXSTA dan RCSTA. Sistem peralatan ini menggunakan komunikasi serial tak sinkron dengan kecepatan tinggi, transmisi 8 bit dan tidak menggunakan paritas. *Baudrate* yang digunakan sebesar 19.200 bps (*bit per second*). Pengaturan *baudrate* dilakukan dengan mengisikan angka tertentu (x) pada register SPBRG sesuai rumusan :

$$\text{Baud Rate} = \frac{F_{osc}}{16(x+1)}$$

Pengaturan ADC internal dilakukan dengan mengkonfigurasi bit-bit pada register ADCON1. Semua port A diatur sebagai masukan analog untuk ADC internal ini. Hasil konversi dari ADC, 10 bit, diletakkan pada register ADRESH dan ADRESL. ADRESH untuk bit yang lebih tinggi dan ADRESL untuk bit yang lebih rendah. Pada modul ini, hasil konversi diletakkan rata kiri dan hanya 8 bit yang digunakan. Hal ini berarti bahwa hanya hasil konversi pada ADRESH yang digunakan dan hasil pada ADRESL diabaikan.

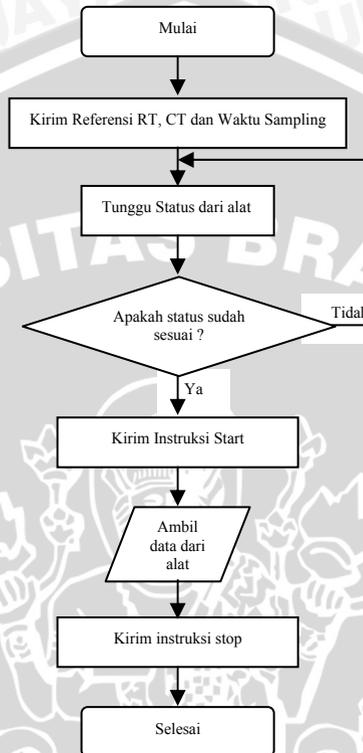
Setelah proses inisialisasi, mikrokontroler menerima referensi nilai frekuensi, yang terbagi dalam R_T dan C_T , referensi tegangan yang dikeluarkan dan mode sampling. Nilai-nilai ini disimpan dalam masing-masing register yang telah khusus dibuat dan kemudian diproses ke dalam sistem peralatan sehingga kondisi sistem peralatan sesuai dengan nilai referensi yang diinginkan oleh pengguna. Apabila sistem peralatan telah sesuai, mikrokontroler kembali mengirimkan status sistem peralatan dan menyatakan bahwa sistem peralatan telah siap untuk digunakan.

Untuk menjalankan dan menghentikan sistem peralatan, mikrokontroler menunggu instruksi *start* atau *stop* dari *user interface* pada PC. Setelah mendapatkan instruksi *start*, mikrokontroler mulai mengambil data dari sistem peralatan yang masih berupa data analog dan mengirimkannya ke PC setelah dikonversi menjadi data digital. Pada proses ini, nilai tegangan keluarannya diatur agar stabil sesuai dengan nilai amplitudo tegangan yang diinginkan. Instruksi *stop* bisa dikirimkan dari *user interface* apabila data yang telah diambil sudah dirasa cukup.

Pada perancangan perangkat lunak ini, hal yang perlu diperhatikan adalah bagaimana membangun perangkat lunak yang komunikatif antara perangkat lunak pada mikrokontroler dan pada *user interface*. Kedua perangkat lunak ini saling berhubungan dan memiliki fungsi masing-masing untuk membangun sebuah sistem peralatan yang diinginkan.

Sistem peralatan yang akan dibuat memerlukan *user interface* untuk mengoptimalkan pemanfaatannya. *User interface* digunakan sebagai perantara antara manusia dengan sistem peralatan agar bisa berkomunikasi dengan baik. Berkomunikasi dalam hal ini adalah manusia bisa memberikan perintah khusus pada peralatan dan peralatan bisa menjalankan sesuai dengan perintah yang diberikan oleh manusia. *User interface* biasanya berupa informasi grafik, teks maupun suara. Dalam penelitian ini, *user interface* dibangun dengan bahasa pemrograman delphi-7 yang bisa didesain agar lebih interaktif dengan pengguna melalui grafis.





Gambar 3.11 Diagram alir perangkat lunak *user interface*.

Bagan di atas merupakan diagram alir yang digunakan dalam membangun perangkat lunak *user interface*. Permulaan penggunaan sistem peralatan ini dilakukan dengan membuka koneksi atau komunikasi antara *user interface* pada PC dengan mikrokontroler pada sistem peralatan. Setelah komunikasi sudah siap, user interface mengirimkan nilai frekuensi yang diinginkan (R_T , C_T) dan waktu sampling. Setelah itu alat akan mengirimkan kembali data tersebut dan ditampilkan dalam user interface. Apabila datanya sama maka pengguna bisa menggunakan instruksi untuk pengambilan data dari alat dan menghentikannya ketika data sudah dirasa cukup.

3.4 Metode Pengujian Sistem Peralatan

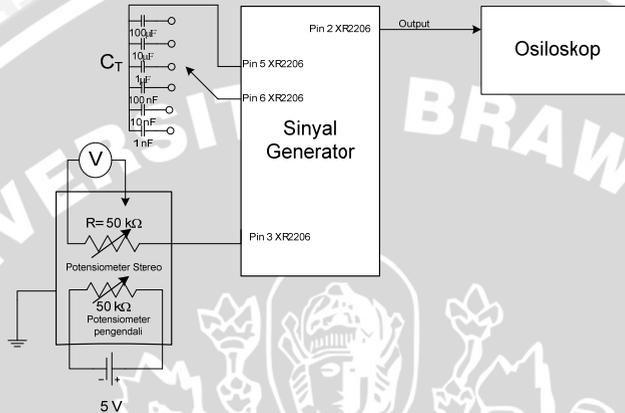
3.4.1 Pengujian perangkat keras

Pengujian perangkat keras terdiri dari pengujian generator sinyal dan selektor arus pada modul sumber arus. Pengujian modul akuisisi data dan kontrol tidak termasuk dalam pengujian ini, akan tetapi masuk pada pengujian perangkat lunak karena pengujian modul akuisisi data dan kontrol memerlukan abntuan perangkat lunak.

Pengujian generator sinyal merupakan pengujian untuk mengetahui bentuk sinyal keluaran yang dihasilkan oleh modul sinyal generator dengan mengubah konfigurasi resistor pewaktu dan kapasitor pewaktu yang berpengaruh terhadap frekuensi sinyal. Pemilihan resistor dan kapasitor pewaktu dilakukan secara manual tanpa bantuan program. Hasil keluaran sinyal langsung dihubungkan pada masukan osiloskop sehingga bisa terlihat bagaimana bentuk sinyalnya. Besarnya resistor dan kapasitor pewaktu diubah-ubah dan bentuk sinyal diperhatikan serta diamati apakah frekuensi dari sinyal pada osiloskop sesuai dengan yang diharapkan. Nilai resistor pewaktu yang digunakan adalah 125 k Ω , 166 k Ω , 200 k Ω , 250 k Ω , 333 k Ω , 500 k Ω dan 1000 k Ω . Nilai kapasitor pewaktu yang digunakan adalah 1 nF, 10 nF, 100 nF, 1 μ F, 10 μ F dan 100 μ F. Nilai kapasitor disini bertindak sebagai orde frekuensi sedangkan nilai resistor pewaktu sebagai nilai satuannya. Oleh karena itu pada pengujian ini besarnya nilai resistor pewaktu ditentukan dahulu.

Selain mengamati bentuk dan frekuensi sinyal, karakteristik sinyal lain yang diamati adalah besarnya amplitudo sinyal keluaran. Karakteristik amplitudo keluaran dilakukan dengan metode yang berbeda. Pengujian karakteristik amplitudo dilakukan dengan menggunakan potensiometer stereo 50 k Ω dimana potensiometer pengendali dihubungkan dengan tegangan 5 volt dan potensiometer lainnya dihubungkan dengan kaki pin nomor 3 dari IC XR2206. Ketika semua rangkaian terhubung (lihat Gambar 3.12), nilai potensiometer pengendali diubah-ubah mulai dari 0 volt hingga 5 volt dengan interval 0,25 volt. Pengukuran tegangan pada potensiometer pengendali ini dilakukan secara manual dengan menggunakan voltmeter. Pada masing-masing tegangan tersebut besarnya tegangan keluaran dari modul sumber arus diukur untuk masing-masing nilai kapasitor pewaktu. Nilai kapasitor pewaktu

diubah secara manual, yaitu dengan menggunakan jumper. Pada saat pengukuran nilai dari resistor pewaktu tetap. Tegangan keluaran dari modul sumber arus ini diukur dengan menggunakan osiloskop agar dapat diketahui juga bentuk sinyal keluaran.



Gambar 3.12 Rangkaian untuk pengujian kalibrasi amplitudo.

Pengujian selektor arus dilakukan untuk mengetahui apakah besarnya arus yang dihasilkan sesuai dengan yang diharapkan atau tidak. Besar arus yang dimaksud adalah besarnya amplitudo tegangan keluaran yang ditunjukkan oleh osiloskop pada saat salah satu nilai resistor selektor arus dipilih.

3.4.2 Pengujian perangkat lunak

Pengujian perangkat lunak dilakukan untuk mengetahui koneksi sistem perangkat keras dengan sistem perangkat lunak yang dibangun untuk menggerakkan perangkat keras tersebut dan juga untuk mengetahui komunikasi sistem apakah sudah berjalan dengan baik. Selain itu, pengujian ini juga dilakukan untuk mengetahui apakah keseluruhan komponen dari perangkat keras sudah bisa berfungsi dengan baik.

Pada pengujian perangkat keras, komponen yang diuji kebanyakan mengenai komponen yang bekerja atau dipilih secara analog. Pada pengujian perangkat lunak ini, komponen-komponen

yang bekerja atau dipilih secara digital juga ikut diuji. Komponen-komponen yang dimaksud antara lain motor stepper untuk pemilihan resistor pewaktu dan relai untuk pemilihan kapasitor pewaktu.

Setelah semua komponen sudah bisa bekerja sesuai fungsinya dan sudah bisa dikendalikan dengan perangkat lunak, sinyal keluaran dari modul sumber arus bisa diamati dalam tampilan perangkat lunak tersebut. Hasil dari sinyal keluaran ini kemudian dianalisa apakah sudah sesuai dengan yang diharapkan atau tidak. Tampilan sinyal yang diharapkan bisa menunjukkan hasil sinyal keluaran untuk orde rendah, dimana sinyal keluaran ini sulit diamati dengan perangkat lainnya.



BAB IV PEMBAHASAN

4.1 Hasil Pengujian Perangkat Keras

4.1.1 Hasil pengujian generator sinyal

Generator sinyal pada modul sumber arus terprogram bisa menghasilkan bentuk gelombang sinusoidal dengan frekuensi tertentu. Frekuensi yang dihasilkan berbeda-beda sesuai dengan konfigurasi resistor pewaktu dan kapasitor pewaktu. Pada pengujian ini, hanya dilakukan pengujian pada nilai kapasitor pewaktu 1 nF, 10 nF, 100 nF dan 1000 nF saja. Untuk nilai lainnya tidak disertakan karena keterbatasan alat dalam membaca frekuensi konkrit dari hasil sinyal. Hasil perbandingan dari nilai frekuensi sinyal keluaran dengan frekuensi hasil perhitungan dapat dilihat pada Tabel 4.1.

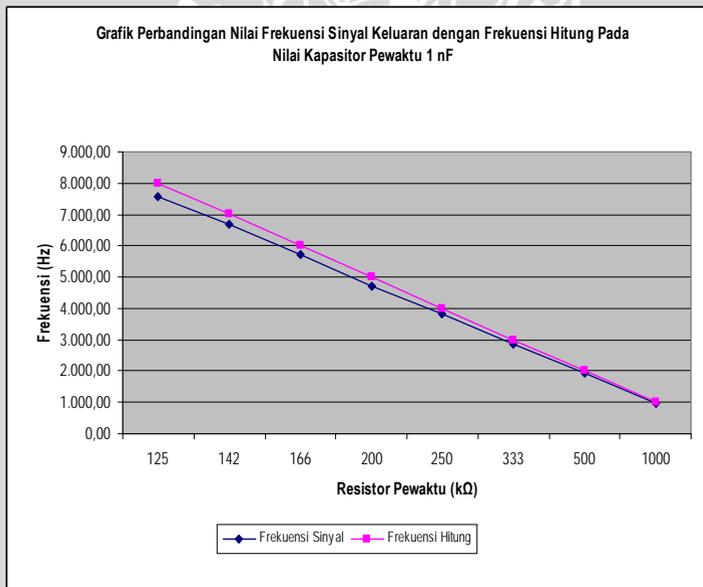
Tabel 4.1 Perbandingan nilai frekuensi sinyal dan frekuensi hitung

C_T (nF)	R_T (k Ω)	f sinyal (Hz)	f hitung (Hz)	C_T (nF)	R_T (k Ω)	f sinyal (Hz)	f hitung (Hz)
1	125	7.566,67	8.000,00	100	125	78,00	80,00
1	142	6.685,33	7.042,25	100	142	69,00	70,42
1	166	5.708,67	6.024,10	100	166	59,00	60,24
1	200	4.700,00	5.000,00	100	200	50,00	50,00
1	250	3.819,00	4.000,00	100	250	66,67	40,00
1	333	2.866,67	3.003,00	100	333	30,00	30,03
1	500	1.950,67	2.000,00	100	500	19,67	20,00
1	1000	978,00	1.000,00	100	1000	10,33	10,00
10	125	803,33	800,00	1000	125	9,00	8,00
10	142	712,67	704,23	1000	142	7,67	7,04
10	166	608,67	602,41	1000	166	6,67	6,02
10	200	506,00	500,00	1000	200	6,00	5,00
10	250	408,67	400,00	1000	250	4,33	4,00
10	333	310,00	300,30	1000	333	3,00	3,00
10	500	203,67	200,00	1000	500	1,33	2,00
10	1000	73,00	100,00	1000	1000	1,33	1,00

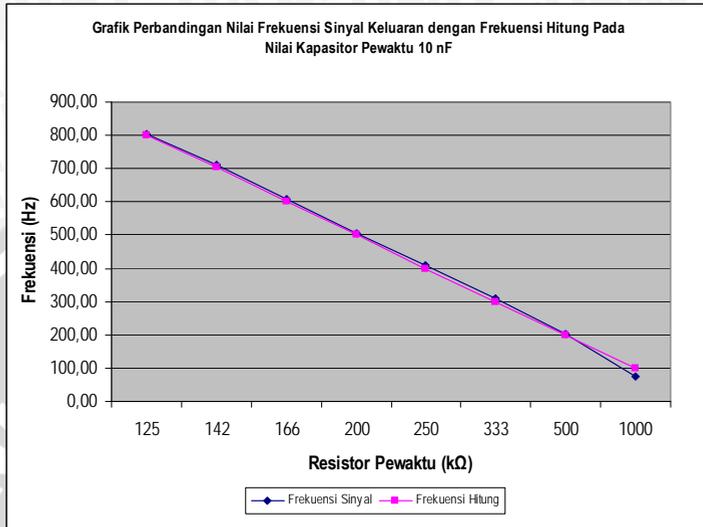
Secara garis besar, frekuensi sinyal keluaran hampir sama dengan frekuensi hasil perhitungan. Frekuensi hasil perhitungan didapatkan dari persamaan

$$f = \frac{1}{R_T C_T}$$

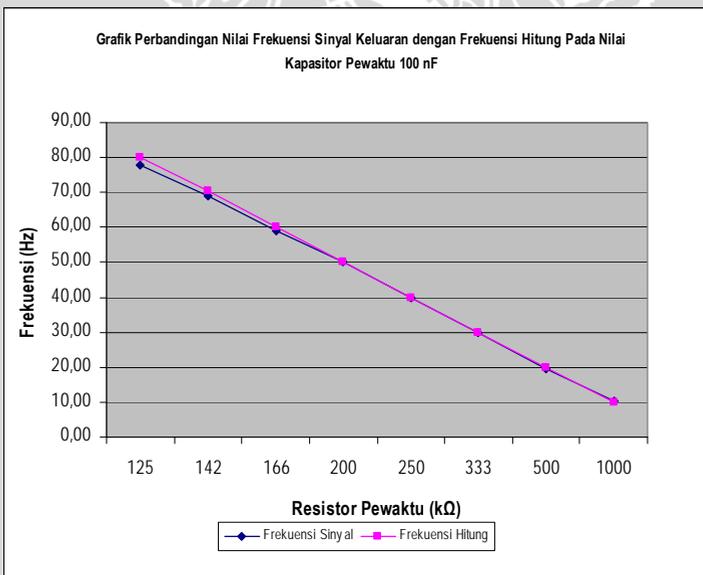
Nilai variasi dari kapasitor menurut tabel di atas bisa dijadikan sebagai acuan orde frekuensi. Kapasitor 1 nF untuk orde 1000 Hz, 10 nF untuk orde 100 Hz, 100 nF untuk orde 10 Hz dan 1000 nF untuk orde 1 Hz. Jika kapasitor selanjutnya (10 μ F dan 100 μ F) digunakan, maka untuk akan didapatkan orde 0,1 Hz dari kapasitor 10 μ F dan orde 0,01 Hz dari kapasitor 100 μ F. Jika ditinjau lebih dekat lagi nilai variasi dari resistor juga menghasilkan nilai satuan tertentu pada masing-masing orde.



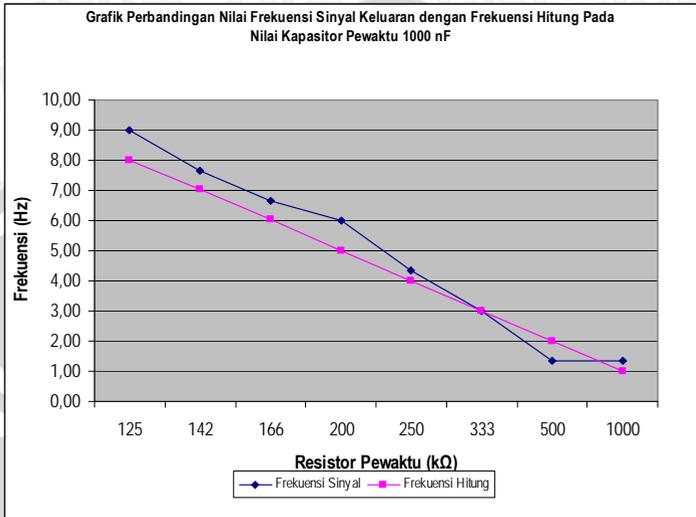
(a)



(b)



(c)



(d)

Gambar 4.1 Grafik perbandingan nilai frekuensi sinyal keluaran dengan frekuensi hitung pada (a) nilai kapasitor pewaktu 1 nF, (b) nilai kapasitor pewaktu 10 nF, (c) nilai kapasitor pewaktu 100 nF, (d) nilai kapasitor pewaktu 1000 nF.

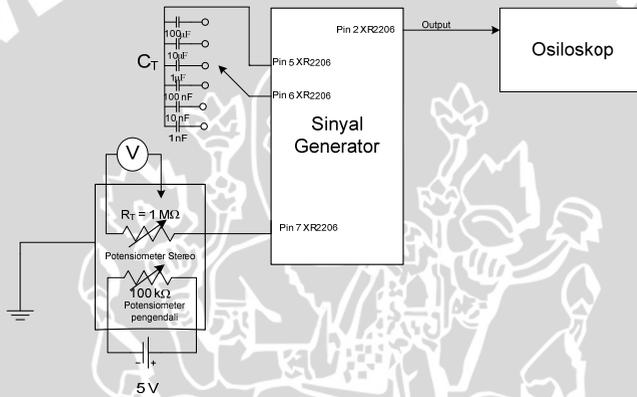
Grafik perbandingan di atas menunjukkan bahwa nilai R_T 125 kΩ hampir mendekati nilai satuan 8, 142 kΩ hampir mendekati nilai satuan 7 dan seterusnya. Nilai satuan 9, menurut grafik di atas juga bisa dijangkau oleh nilai resistansi di bawah 125 kΩ. Jangkauan nilai sumbu x (resistor pewaktu) pada grafik di atas tidak sama dari nilai resistansi yang satu lain. Grafik tersebut dibuat hanya bertujuan untuk membandingkan nilai frekuensi sinyal keluaran dengan frekuensi hitung saja, pada nilai resistansi yang sudah ditentukan.

Secara ideal, hubungan antara resistor pewaktu dengan frekuensi hitung untuk nilai kapasitor 1nF ditunjukkan pada Gambar 4.2. Nilai resistor 0 – 1000 kΩ dipilih untuk merepresentasikan nilai potensiometer. Grafik ini juga menunjukkan hubungan berbanding terbalik antara nilai resistor pewaktu dengan nilai frekuensinya.



Gambar 4.2 Grafik hubungan resistor pewaktu dengan frekuensi hitung pada nilai kapasitor pewaktu 1nF

Pengujian mengenai hubungan antara frekuensi sinyal keluaran dan resistor pewaktu dengan metode yang sedikit berbeda juga dilakukan. Nilai resistansi dipilih dari 0 sampai 1000 k Ω dengan jangkauan yang sama. Pada pengujian langsung nilai resistansi ini diwakili oleh tegangan referensi. Tegangan referensi yang dihubungkan pada potensiometer pengendali diubah dari 0 sampai 5 volt dengan jangkauan 0,5 volt. Besarnya frekuensi dan bentuk sinyal keluaran pada masing-masing C_T dan tegangan referensi tersebut diamati dan dicatat. Gambar 4.3 menunjukkan rangkaian pengujian ini.



Gambar 4.3 Rangkaian pengujian hubungan frekuensi sinyal keluaran dengan resistor pewaktu.

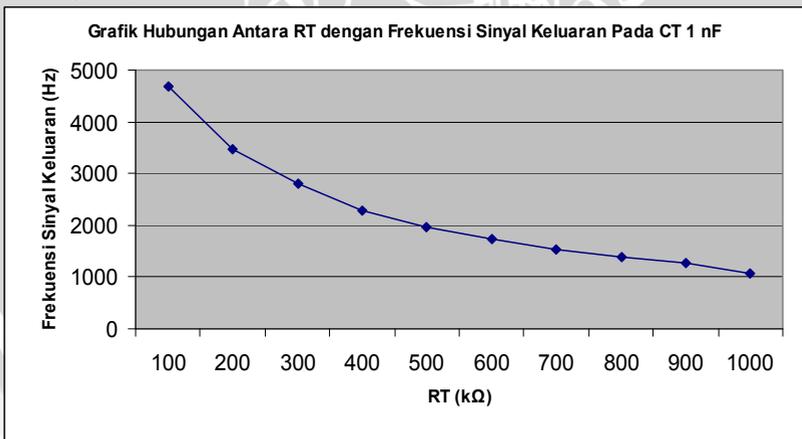
Hasil dari pengujian dalam bentuk grafik ditunjukkan pada Gambar 4.4. Hasil pengujian ini menunjukkan bentuk grafik melengkung yang sama dengan perbandingan ideal pada Gambar 4.2. Grafik pada gambar menunjukkan hubungan antara R_T dengan frekuensi sinyal keluaran. Tabel hasil pengujian bisa dilihat pada lampiran 1.1. Nilai R_T dikonversi dari nilai tegangan referensi

melalui persamaan
$$R_T = \frac{V_{ref}}{5 \text{ volt}} \times 1000 \text{ k}\Omega$$
 yang hasilnya ditunjukkan pada Tabel 4.2.

Tabel 4.2 Konversi nilai tegangan kalibrasi ke nilai R_T pada pengujian selektor frekuensi.

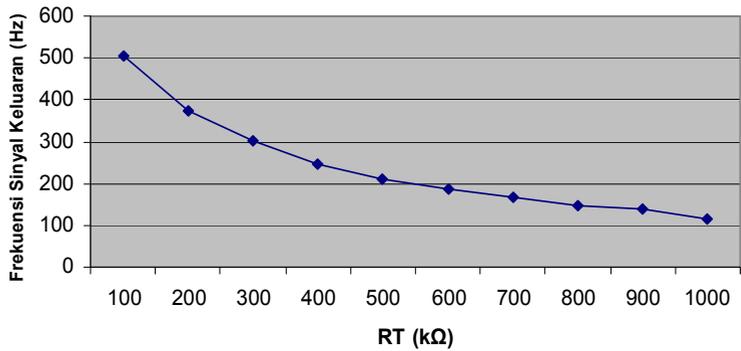
V_{ref} (volt)	R_T (k Ω)
0	0
0,5	100
1	200
1,5	300
2	400
2,5	500
3	600
3,5	700
4	800
4,5	900
5	1000

Pengujian ini dilakukan untuk semua nilai kapasitor pewaktu dan memiliki hasil atau hubungan yang hampir sama. Akan tetapi pada pada kapasitor 100 μF , pengujian tidak dapat dilanjutkan untuk nilai tegangan/resistor yang besar karena keterbatasan alat dalam membaca frekuensi yang sangat rendah. Untuk masing-masing kapasitor, grafik sengaja dipisah agar nampak jelas kurva hubungannya.



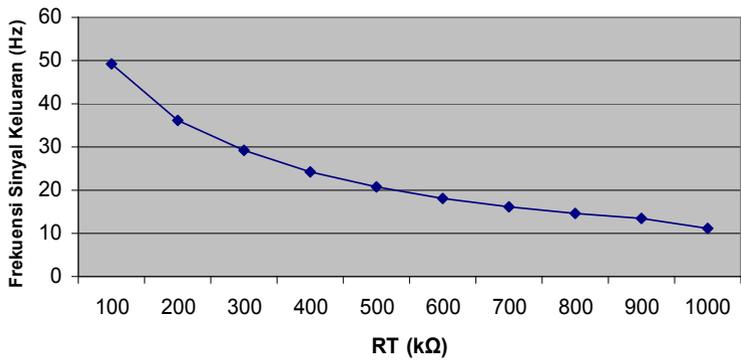
(a)

Grafik Hubungan Antara RT dengan Frekuensi Sinyal Keluaran Pada CT 10 nF

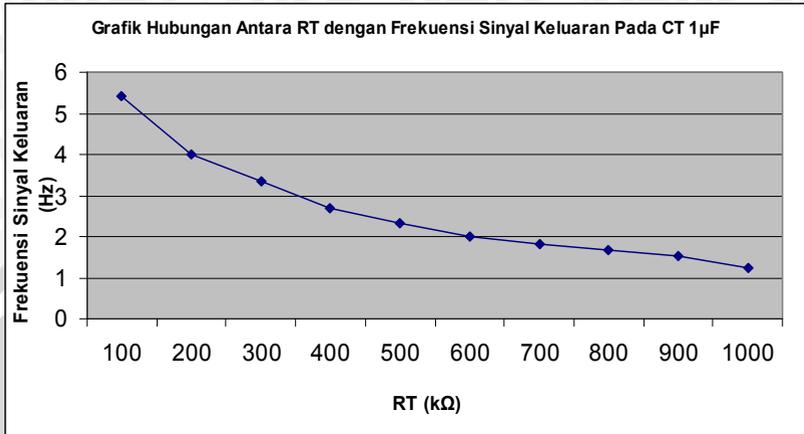


(b)

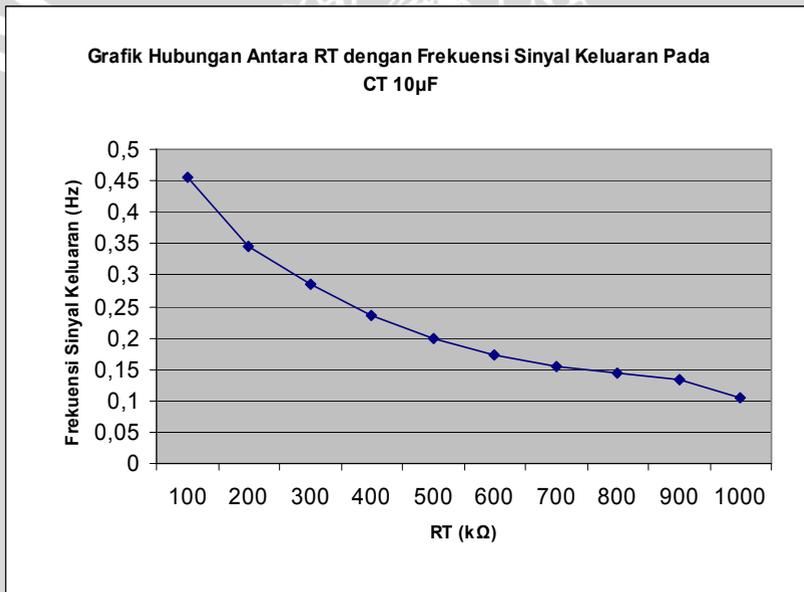
Grafik Hubungan Antara RT dengan Frekuensi Sinyal Keluaran Pada CT 100 nF



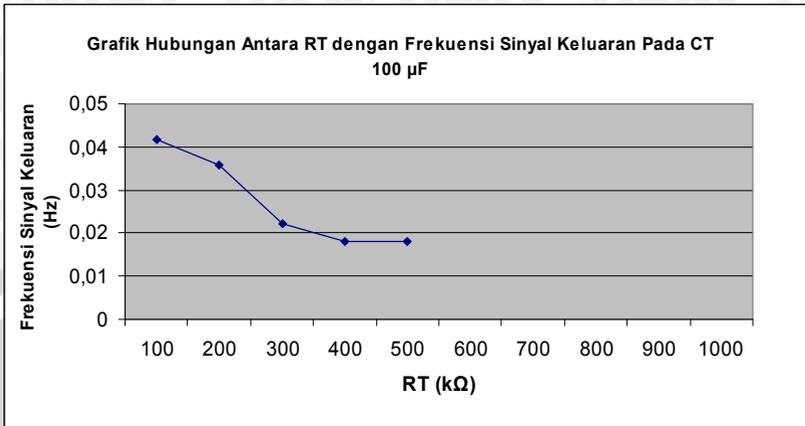
(c)



(d)

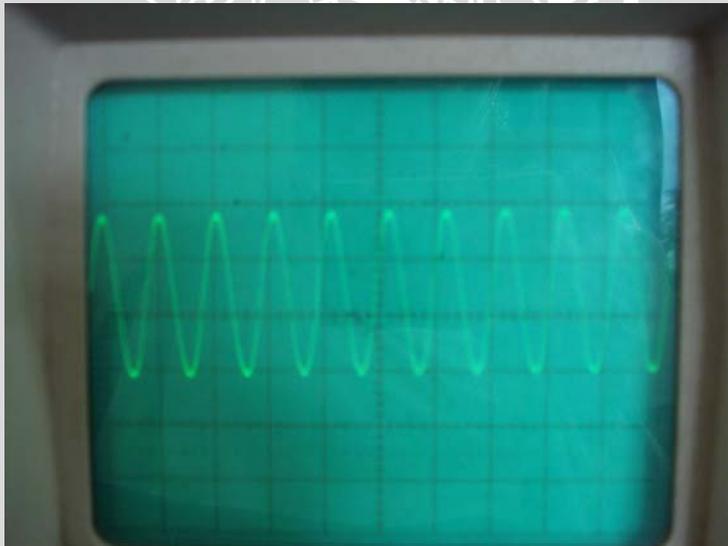


(e)



(f)

Gambar 4.4 Grafik hubungan antara R_T dengan frekuensi sinyal keluaran pada nilai kapasitor (a) 1 nF, (b) 10 nF, (c) 100 nF, (d) 1 μF , (e) 10 μF dan (f) 100 μF .



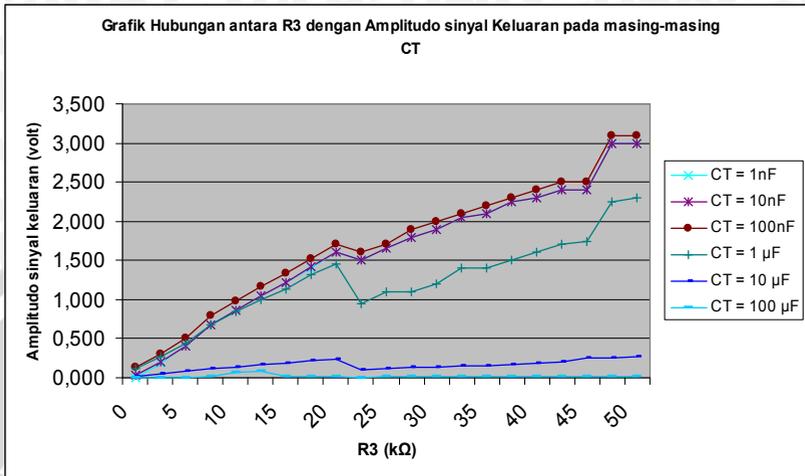
Gambar 4.5 Sinyal keluaran dari generator sinyal.

Pengatur eksternal yang digunakan pada generator sinyal (seperti yang dijelaskan pada bab sebelumnya) memungkinkan pengguna

untuk menyesuaikan bentuk sinus dari sinyal keluaran. Pada awal pengujian generator sinyal, bentuk sinyal sinus masih belum sesuai. Setelah menyatur kesimetrian dan amplitudonya, bentuk sinyal keluaran sinus ini bisa bagus. Gambar 4.5 merupakan salah satu bentuk sinyal keluaran dari generator sinyal pada saat pengujian sinyal generator.

Hasil pengujian karakteristik amplitudo sinyal keluaran menyatakan bahwa perubahan resistor eksternal, R_3 , mempengaruhi besar amplitudo keluaran dengan hubungan berbanding lurus. Akan tetapi grafik yang didapatkan dari pengujian ini tidak sebagus yang dicantumkan dalam karakteristik XR2206. Ada beberapa penyimpangan yang didapatkan, mungkin ini dikarenakan kurang presisi dalam pengukuran nilai potensiometer sesuai dengan yang ditentukan.

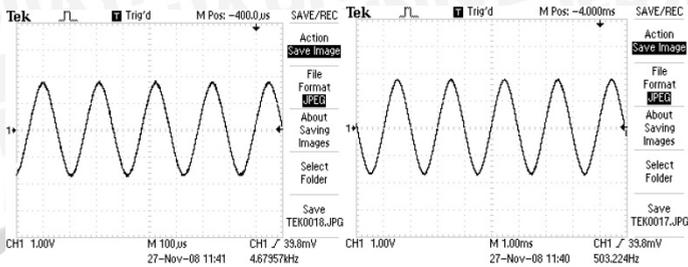
Gambar 4.6 menunjukkan bahwa besarnya kapasitor pewaktu yang digunakan juga mempengaruhi besar amplitudo keluaran. Kapasitor pewaktu 1 nF, 10 nF dan 100 nF bisa menjangkau semua nilai amplitudo tegangan pada range 0 – 3 volt. Hal ini berarti bahwa orde frekuensi 1000 Hz, 100 Hz dan 10 Hz bisa dihasilkan dengan baik pada perangkat ini. Kapasitor 1 μ F belum bisa menjangkau amplitudo maksimum sehingga untuk menghasilkan frekuensi pada orde 10 Hz sebaiknya menggunakan tegangan dibawah 2,5 volt. Lain halnya dengan kapasitor 10 μ F dan 100 μ F. Besar amplitudo keluaran yang dihasilkan sangat kecil di bawah 0,3 volt bahkan sampai orde milivolt. Jadi, walaupun frekuensi sinyal yang dikeluarkan dari modul ini bisa mencapai orde 0,01 Hz akan tetapi besar amplitudonya tidak bisa tinggi.



Gambar 4.6 Grafik hubungan R_3 terhadap amplitudo sinyal pada nilai kapasitor pewaktu tertentu.

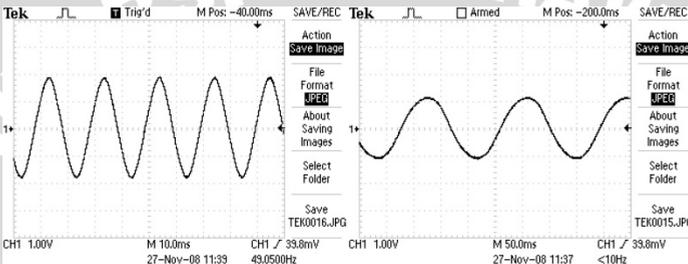
Pola grafik yang ditunjukkan pada Gambar 4.6 tersebut mengalami pola turun pada nilai tertentu. Seharusnya setiap kenaikan nilai R_3 besarnya amplitudo juga naik. Penurunan grafik ini bisa dikarenakan pada saat pengambilan data pembacaan nilai tegangan referensi yang mengacu pada nilai R_3 kurang tepat. Penyebab lainnya bisa juga dikarenakan adanya guncangan yang menyebabkan perubahan nilai R_3 , mengingat potensiometer yang digunakan pada R_3 sangat sensitif. Nilai amplitudo sinyal keluaran yang dihasilkan pada C_T 1 nF dan 10nF hampir sama sehingga pada grafik C_T 1nF tidak terlihat.

Bentuk sinyal keluaran yang dihasilkan pada berbagai orde frekuensi, yang ditunjukkan pada pengujian ini, secara garis besar sama yakni berbentuk sinyal sinusoidal. Akan tetapi pada orde rendah (0,1 Hz dan 0,01 Hz) bentuk sinyal sinusoidal mulai tidak beraturan. Gambar 4.7 menunjukkan bentuk sinyal sinusoidal pada masing-masing orde dengan nilai R_T yang sama yaitu 100 k Ω . Bentuk sinyal bisa diamati secara lengkap pada lampiran 3.



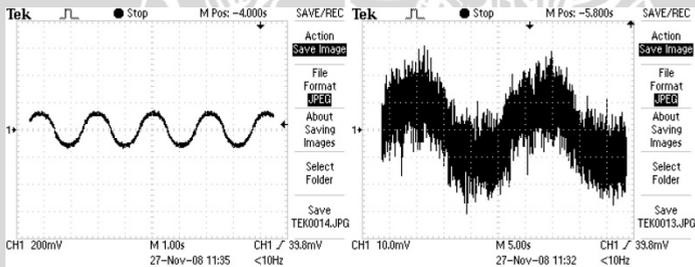
(a)

(b)



(c)

(d)



(a)

(b)

Gambar 4.7 Bentuk sinyal keluaran pada R_T sama dengan 100Ω dan, (a) orde 1 kHz; (b) orde 100 Hz; (c) orde 10 Hz; (d) orde 1 Hz; (e) orde 0,1 Hz dan (f) orde 0,01 Hz.

4.1.2 Hasil Pengujian selektor arus

Pengujian modul sumber arus berikutnya adalah pengujian selektor arus. Selektor arus di sini adalah resistor-resistor yang dihubungkan langsung pada keluaran generator sinyal. Gambar 4.8

merupakan grafik perbandingan amplitudo sinyal arus dari ketiga selektor arus. Masing-masing selektor di ambil dalam 3 keadaan frekuensi sinyal. Keadaan frekuensi pertama pada konfigurasi $C_T = 1 \mu\text{F}$ dan $R_T = 1 \text{M}\Omega$, keadaan kedua $C_T = 100 \text{nF}$ dan $R_T = 1 \text{M}\Omega$ dan keadaan ketiga $C_T = 10 \text{nF}$ dan $R_T = 1 \text{M}\Omega$. Ketiga keadaan tersebut dipilih karena sinyal keluaran pada osiloskop bisa terlihat jelas pada keadaan ini. Sebelum pengujian ini dilakukan, sinyal keluaran dari generator sinyal diatur sehingga memiliki amplitudo 1 volt. Data yang diambil langsung pada pengujian ini adalah berupa tegangan. Data ini kemudian dikonversi menjadi arus melalui persamaan hukum ohm untuk masing-masing selektor arus.

$$I = \frac{V}{R_{\text{selektor}}}$$

Tabel 4.2 menunjukkan hasil pengujian selektor arus. Di dalam tabel ini, $R_A = 10 \text{M}\Omega$, $R_B = 1 \text{M}\Omega$ dan $R_C = 100 \text{k}\Omega$. Ketiga selektor ini menghasilkan besar arus dengan orde masing-masing. Pada Tabel 4.3 dapat dilihat bahwa selektor arus R_A menghasilkan arus pada orde $0,1 \mu\text{A}$, selektor R_B menghasilkan arus pada orde $1 \mu\text{A}$ dan selektor R_C menghasilkan arus pada orde $10 \mu\text{A}$. Hal ini sesuai dengan yang diharapkan.

Tabel 4.3 Hasil pengujian selektor arus.

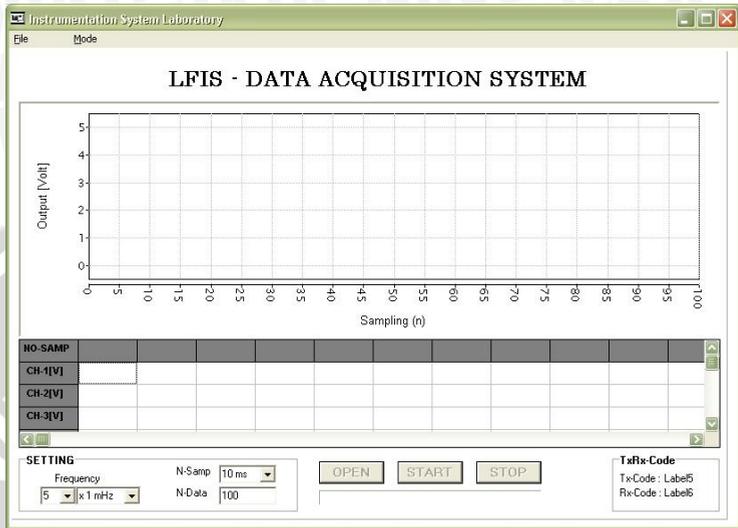
Keadaan Frekuensi	R Selektor	V1 (volt)	V2 (volt)	V3 (volt)	V rata-rata	I (μA)
1	R_A	0,150	0,150	0,150	0,150	0,0150
2	R_A	0,150	0,150	0,150	0,150	0,0150
3	R_A	0,150	0,145	0,145	0,147	0,0147
1	R_B	0,800	0,800	0,800	0,800	0,800
2	R_B	0,775	0,775	0,775	0,775	0,775
3	R_B	0,775	0,763	0,763	0,767	0,767
1	R_C	1,45	1,45	1,45	1,45	14,50
2	R_C	1,50	1,50	1,50	1,50	15,00
3	R_C	1,50	1,50	1,50	1,50	15,00

4.2 Hasil Pengujian Perangkat Lunak

Seperti yang dijelaskan pada bab sebelumnya, tujuan utama dari pengujian perangkat lunak ini adalah untuk mengetahui kinerja keseluruhan sistem yang dikendalikan secara terprogram (otomatis) oleh perangkat lunak, terutama pada selektor frekuensi. Perangkat lunak yang dibangun pada mikrokontroller dan *user interface* bekerja secara berkesinambungan. *Listing* program pada kedua perangkat lunak ini bisa dilihat pada lampiran 2.

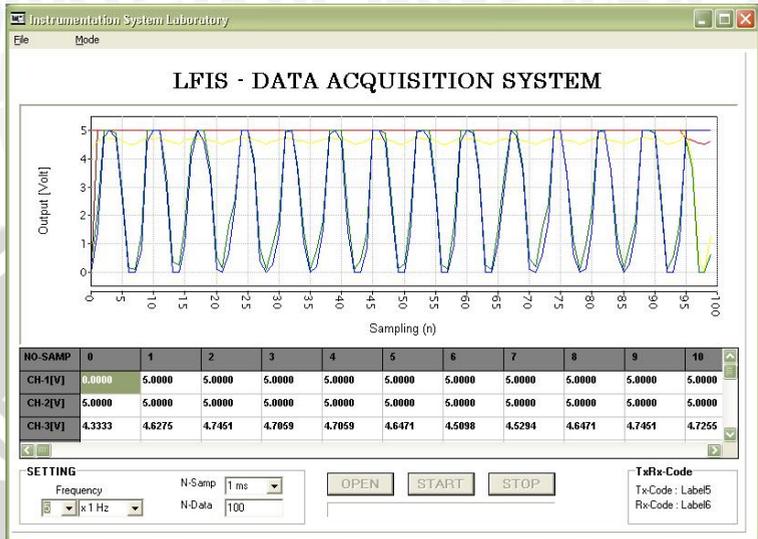
Gambar 4.11 merupakan tampilan dari *user interface*. Pada *user interface* ini terdapat menu untuk pemilihan frekuensi sinyal keluaran, jumlah data dan waktu sampling. Pada menu mode terdapat dua sub menu yaitu *online* dan *offline*. Mode *online* dipilih ketika akan mengambil data dari alat. Pada mode ini dapat dipilih *baudrate* untuk komunikasi serial kemudian tombol *open* diaktifkan. Ketika tombol *open* dipencet, data frekuensi, jumlah data dan waktu sampling dikirim ke mikrokontroller sehingga mikrokontroller bisa langsung mengatur perputaran motor stepper untuk pemilihan resistor pewaktu dan pengaktifan relai untuk kapasitor pewaktu yang dipilih. Ketika semua alat sudah siap mengirimkan data, maka mikrokontroller kembali mengirimkan data frekuensi dan yang lainnya. Pengiriman data kembali ini dilakukan hanya untuk penyesuaian saja. Apabila data sudah sesuai maka komunikasi dinyatakan baik dan data bisa diambil dengan memencet tombol *start*.





Gambar 4.8 Tampilan *user interface*.

Hasil pengujian dengan menggunakan perangkat lunak ini tidak dilakukan untuk semua nilai frekuensi. Fokus utama dari pengujian ini adalah mengetahui bentuk sinyal yang dihasilkan pada masing – masing tingkatan frekuensi terutama frekuensi rendah. Gambar 4.12 merupakan grafik hasil pengujian untuk frekuensi 1 Hz. Dari gambar tersebut bisa dilihat bahwa sinyal yang dihasilkan berbentuk sinusoidal akan tetapi masih ada sedikit distorsi. Distorsi ini diakibatkan oleh waktu sampling pada pengambilan data yang mungkin kurang pendek. Apabila dilakukan secara kontinu bentuk sinyal ini akan bisa seperti pada pengujian sinyal generator secara manual.



Gambar 4.9 Bentuk sinyal keluaran pada frekuensi 1 Hz.

Hasil pengujian untuk tingkatan frekuensi yang lain bisa dilihat pada lampiran 4. Secara garis besar bentuk sinyal yang dihasilkan juga tidak halus dan mengalami distorsi.

BAB V PENUTUP

5.1. Kesimpulan

Kesimpulan yang dapat diambil berdasarkan penelitian ini antara lain sebagai berikut :

1. Sumber arus AC terdiri dua bagian utama yaitu modul sumber arus dan modul akuisisi data dan kontrol. Sumber arus berfungsi untuk membangkitkan gelombang sinusoidal yang berupa tegangan yang kemudian diubah menjadi arus oleh pengubah tegangan ke arus (*VI converter*). Modul akuisisi data dan kontrol berfungsi untuk mengendalikan kinerja alat sehingga menghasilkan sinyal keluaran yang bisa ditampilkan dalam bentuk grafis.
2. Sinyal keluaran yang dihasilkan merupakan sinyal sinusoidal dengan frekuensi konstan yang bisa dipilih sesuai dengan kebutuhan. Jangkauan frekuensi yang dihasilkan adalah 0,01 Hz sampai dengan 10 Hz. Amplitudo sinyal pada orde frekuensi 10 Hz bisa mencapai 3 volt, pada orde frekuensi 1 Hz mencapai 2,5 volt dan pada orde frekuensi yang lebih rendah hanya dapat menghasilkan amplitudo pada orde milivolt. Bentuk sinyal keluaran pada frekuensi orde tinggi masih teratur sedangkan pada frekuensi orde paling rendah bentuk sinyal sudah mengalami distorsi.

5.2. Saran / Rekomendasi

Berdasarkan kesimpulan di atas, maka rekomendasi yang bisa penulis sampaikan adalah perlunya ditambahkan modul penguat sinyal terutama pada keluaran sinyal frekuensi rendah agar semua sinyal keluaran bisa dihasilkan pada amplitudo yang sama.

DAFTAR PUSTAKA

- Anonimous. 2003. **Panduan Praktis Pemrograman Delphi 7.0**. Penerbit Andi. Yogyakarta. Hal: 1 – 3.
- Anonimous (a). 2007. **Current Source**. http://en.wikipedia.org/wiki/Current_source. Diakses tanggal 05 Juni 2007 Pukul 13:42 WIB.
- Anonimous (b). 2007. **About Signal Generators and Waveform Generators**. http://test-equipment.globalspec.com/LearnMore/Labware_Test_Measurement/Lab_Test_Equipment/Signal_Function_Waveform_Generators. Diakses tanggal 02 September 2007 Pukul 18:45.
- Anonimous (c). 2007. **Voltage to Current Converter**. http://en.wikipedia.org/wiki/Voltage-to-current_converter. Diakses tanggal 04 September 2007 pukul 20:04 WIB.
- Anonimous (a). 2008. **Lecture 8-Stepper Motors**. http://hibp.ecse.rpi.edu/~connor/education/IEE/lectures/Lecture_8_Stepper_motors.pdf. Diakses Tanggal 21 Pebruari 2008 pukul 06:39 WIB.
- Anonimous (b). 2008. **Tutorial : Microchip PIC16F877 Microcontroller**. <http://www.pages.drexel.edu/~cy56/PIC.htm>. Diakses tanggal 18 Maret 2008 pukul 11:54 WIB.
- Anonimous (c). 2008. **Interfacing Port Pararel Komputer, Serial dan PPI 8255**. <http://www.toko-elektronika.com/tutorial/paralel.html>. Diakses tanggal 16 April 2008 pukul 7:50 WIB.
- Anonimous (d). 2008. **DB Connector Guide**. <http://www.cablestogo.com/resources/db.asp>. Diakses tanggal 18 Maret 2008 pukul 12:56.
- Aswan. 2004. **Operational Amplifier (Analisa Rangkaian Op Amp Popular)**. <http://www.electroniclab.com/index.php?action=html&fi>

- d=52. Diakses tanggal 2 September 2007 pukul 19:39 WIB
- Atmel, 1997. **AT89C51 : 8-Bit Microcontroller with 4K bytes Flash.**
- Bies, Lammert. 2008. **RS232 Serial Cable Layout.**
<http://www.lammertbies.nl/comm/cable/RS-232.html>.
Diakses tanggal 27 Mei 2008 pukul 10:24 WIB.
- Budiharto, Widodo Ssi, Mkom.2004. **Interfacing Komputer dan Mikrokontroler.** PT. Elex Media Komputindo.Jakarta
- Exar. 1997. **XR-2206 Monolithic Function Generator.**
www.datasheetcatalog.com/datasheet_pdf/X/R/2/2/XR2206.shtml. Diakses tanggal 24 Juli 2008 Pukul 14:25.
- Intersil. 2005. **Intersil HIN232, HIN236, HIN237, HIN238, HIN239, HIN240, HIN241 Data Sheet.** http://measure.feld.cvut.cz/groups/edu/x38NRP/fn3138_HI232.pdf. Diakses tanggal 26 September 2007 pukul 11:13 WIB.
- Jones, Dr. Douglas W. 2004. **Stepping Motors Fundamental.**
<http://www.cs.uiowa.edu/~jones/step/an907a.pdf>. Diakses tanggal 21 Pebruari 2008 pukul 06:38 WIB.
- Malik, Moh.Ibnu. 2003. **Belajar Mikrokontroler PIC 16F84.** Gava Media. Yogyakarta
- Margunadi, Ir. A.R. 1990. **Teori Rangkaian (Dasar - Dasar).** Erlangga. Jakarta. Hal :42 - 43
- Microchip. 2001. **PIC16F87X Data Sheet 28/40-pin 8-bit CMOS Flash Microcontroller, Microchip Technology Inc.**
<http://www.microchip.com/downloads/en/DeviceDoc/30292c.pdf>. Diakses tanggal 17 Oktober 2007.
- Prasetia, Retna dan Catur Edi Widodo. 2004. **Teori dan Praktek Interfacing Port Pararel dan Port Seroal Komputer dengan Visual Basic 6.0.** Penerbit Andi. Yogyakarta. Hal 67.
- Putra, Agfianto Eko. 2002. **Penapis Aktif Elektronika: Teori dan Praktek.** C.V. Gava Media. Yogyakarta. Hal 3, 5.
- Wasito. 2004. **Vademekum Elektronika.** PT Gramedia Pustaka Utama. Jakarta. Hal 347.

Lampiran 1

Data hasil pengujian perangkat keras

1.1. Pengujian Hubungan Frekuensi sinyal keluaran dengan resistor pewartu

No	V_{ref} (volt)	R_T (k Ω)	Frekuensi untuk masing-masing C_T (Hz)					
			1 nF	10 nF	100 nF	1 μ F	10 μ F	100 μ F
1	0	0	686,600	95,077	9,590	1,071	100	95
2	0,5	100	4.670,000	503,125	49,050	5,405	0,455	0,042
3	1	200	3.457,000	372,692	36,330	4,000	0,345	0,036
4	1,5	300	2.790,000	300,747	29,350	3,333	0,286	0,022
5	2	400	2.290,000	246,917	24,061	2,703	0,235	0,018
6	2,5	500	1.970,000	211,322	20,624	2,326	0,200	0,018
7	3	600	1.729,000	186,578	18,210	2,000	0,174	
8	3,5	700	1.529,000	164,904	16,067	1,818	0,154	
9	4	800	1.378,000	148,264	14,460	1,667	0,143	
10	4,5	900	1.279,800	137,521	13,440	1,538	0,133	
11	5	1000	1.059,000	113,694	11,104	1,250	0,105	

1.2. Pengujian Kalibrasi amplitudo

No	V_{ref} (volt)	R_3 (k Ω)	Amplitudo Sinyal untuk masing-masing C_T (volt)					
			1 nF	10 nF	100 nF	1 μ F	10 μ F	100 μ F
1	0,00	0,0	0,005	0,035	0,128	0,120	0,024	0,004
2	0,25	2,5	0,190	0,200	0,305	0,270	0,046	0,004
3	0,50	5,0	0,400	0,410	0,510	0,440	0,080	0,008
4	0,75	7,5	0,680	0,680	0,800	0,680	0,116	0,025
5	1,00	10,0	0,860	0,860	0,980	0,840	0,140	0,075
6	1,25	12,5	1,040	1,040	1,160	1,000	0,170	0,080
7	1,50	15,0	1,220	1,220	1,340	1,140	0,190	0,016
8	1,75	17,5	1,410	1,420	1,520	1,320	0,220	0,020
9	2,00	20,0	1,600	1,600	1,700	1,450	0,240	0,022
10	2,25	22,5	1,500	1,500	1,600	0,950	0,100	0,008
11	2,50	25,0	1,650	1,650	1,700	1,100	0,120	0,012
12	2,75	27,5	1,800	1,800	1,900	1,100	0,140	0,012
13	3,00	30,0	1,900	1,900	2,000	1,200	0,140	0,014
14	3,25	32,5	2,050	2,050	2,100	1,400	0,155	0,014
15	3,50	35,0	2,100	2,100	2,200	1,400	0,160	0,014
16	3,75	37,5	2,250	2,250	2,300	1,500	0,175	0,016
17	4,00	40,0	2,300	2,300	2,400	1,600	0,190	0,018
18	4,25	42,5	2,400	2,400	2,500	1,700	0,200	0,018
19	4,50	45,0	2,400	2,400	2,500	1,750	0,260	0,018
20	4,75	47,5	3,000	3,000	3,100	2,250	0,260	0,022
21	5,00	50,0	3,000	3,000	3,100	2,300	0,270	0,024

Lampiran 2

2.1. Source code mikrokontroler (assembly)

```
list p=16f877
#include <P16f877.INC>
ERRORLEVEL -302

;ram usage
RREF equ 21h
CREf equ 22h
MODE equ 23h ;konstanta sampling time
CH0 equ 24h ;adc channel 1
CH1 equ 25h ;adc channel 2
CH2 equ 26h ;adc channel 3
CH3 equ 27h ;adc channel 4
CH4 equ 28h ;adc channel 5(untuk referensi RREF)
CNTAD equ 29h
CONT1 equ 2Ah
CONT2 equ 2Bh
RREF1 equ 2Ch
RREF2 equ 2Dh

;-----
org 0x00
goto START

org 0x10

;-----
;MAIN PROGRAM
;-----
START call init ; initialization
call setting ; receive Vref + CREf + mode (from PC)
call posisi_cap ; posisi saklar kapasitor
call motor_Rref ; cari posisi Rref
call ms_100 ; delay 100 ms, nunggu PC siap menerima data
call kirim_status ; transmit RREF + CREf + mode (to PC)

;----- mode operasi
bcf STATUS,RP0
movfw MODE
bcf STATUS,Z
xorlw 'A'
btfsc STATUS,Z
goto modeA
movfw MODE
```



```

bcf      STATUS,Z
xorlw   'B'
btfst   STATUS,Z
goto    modeB
movfw   MODE
bcf      STATUS,Z
xorlw   'C'
btfst   STATUS,Z
goto    modeC
movfw   MODE
bcf      STATUS,Z
xorlw   'D'
btfst   STATUS,Z
goto    modeD
movfw   MODE
bcf      STATUS,Z
xorlw   'E'
btfst   STATUS,Z
goto    modeE
movfw   MODE
bcf      STATUS,Z
xorlw   'F'
btfst   STATUS,Z
goto    modeF
movfw   MODE
bcf      STATUS,Z
xorlw   'G'
btfst   STATUS,Z
goto    modeG
movfw   MODE
bcf      STATUS,Z
xorlw   'H'
btfst   STATUS,Z
goto    modeH

```

```

modeA   call   triger_start; PC send "M" char
        call   ms_100      ; nunggu PC siap menerima data
loa     call   AD_Conv     ; 5 channels 8 bits A/D conversion
        call   send_data ; send data to PC
        call   ms_1
        call   triger_stop; PC send 'D' char
        goto   loa

modeB   call   triger_start; PC send "M" char
        call   ms_100      ; nunggu PC siap menerima data
lob     call   AD_Conv     ; 5 channels 8 bits A/D conversion

```

	call	send_data ; send data to PC
	call	ms_2
	call	triger_stop ; PC send 'D' char
	goto	lob
modeC	call	triger_start ; PC send "M" char
	call	ms_100 ; nunggu PC siap menerima data
loc	call	AD_Conv ; 5 channels 8 bits A/D conversion
	call	send_data ; send data to PC
	call	ms_5
	call	triger_stop ; PC send 'D' char
	goto	loc
modeD	call	triger_start ; PC send "M" char
	call	ms_100 ; nunggu PC siap menerima data
lod	call	AD_Conv ; 5 channels 8 bits A/D conversion
	call	send_data ; send data to PC
	call	ms_10
	call	triger_stop ; PC send 'D' char
	goto	lod
modeE	call	triger_start ; PC send "M" char
	call	ms_100 ; nunggu PC siap menerima data
loe	call	AD_Conv ; 5 channels 8 bits A/D conversion
	call	send_data ; send data to PC
	call	ms_50
	call	triger_stop ; PC send 'D' char
	goto	loe
modeF	call	triger_start ; PC send "M" char
	call	ms_100 ; nunggu PC siap menerima data
lof	call	AD_Conv ; 5 channels 8 bits A/D conversion
	call	send_data ; send data to PC
	call	ms_100
	call	triger_stop ; PC send 'D' char
	goto	lof
modeG	call	triger_start ; PC send "M" char
	call	ms_100 ; nunggu PC siap menerima data
log	call	AD_Conv ; 5 channels 8 bits A/D conversion
	call	send_data ; send data to PC
	call	ms_500
	call	triger_stop
	goto	log
modeH	call	triger_start ; PC send "M" char

```

loh      call    ms_100          ; nunggu PC siap menerima data
        call    AD_Conv      ; 5 channels 8 bits A/D conversion
        call    send_data    ; send data to PC
        call    sec_1
        call    triger_stop  ; PC send 'D' char
        goto   loh

```

```

-----
init:
-----

```

```

        clrf    STATUS
        bcf     STATUS,RP0    ;BANK0
        movlw   0X00          ;init timer1, pres 1:1, clock=Fosc/4, osc disable
        movwf  T1CON
        movlw   0x90          ;osilator RC1,RC2 = off
        movwf  RCSTA
        bsf     STATUS,RP0    ;BANK1
        clrf   TRISB         ;PORTB as output
        movlw   0x24          ;BRGH=1
        movwf  TXSTA
        movlw   D'59'         ;19.200 bps at 18.432 MHz
        movwf  SPBRG
        clrf   ADCON1        ;left justify, init ADC
        bcf   STATUS,RP0     ;BANK0
        clrf   CH0           ;channel0
        clrf   CH1           ;channel1
        clrf   CH2           ;channel2
        clrf   CH3           ;channel3
        clrf   CH4           ;channel4
        clrf   PORTB
        clrf   PORTC
        clrf   RREF
        clrf   CREF
        clrf   MODE          ;mode sampling
        clrf   CNTAD
        clrf   CONT1
        clrf   CONT2
        clrf   PIR1         ;clear all interrupt
        movfw  RCREG
        return

```

```

-----
setting: ;baca setting alat (RREF, CREF, mode) kiriman PC
-----

```

```

        btfs   PIR1,RCIF
        goto   $-1
        movfw  RCREG

```

```

movwf RREF ;(R-referensi)
btfs PIR1,RCIF
goto $-1
movfw RCREG
movwf CREF ;(C-referensi)
btfs PIR1,RCIF
goto $-1
movfw RCREG
movwf MODE ;(sampling-time)
incf RREF,0
movwf RREF1
decf RREF,0
movwf RREF2
return

```

posisi_cap:

```

bcf STATUS,RP1
bsf STATUS,RP0 ;bank1
movlw B'10000000'
movfw TRISC ;poRT RC operation
bcf STATUS,RP0 ;bank0
bcf PORTC,0
bcf PORTC,1
bcf PORTC,2
bcf PORTC,3
bcf PORTC,4
bcf PORTC,5
movfw CREF
bcf STATUS,Z
xorlw D'1'
btfsc STATUS,Z
bsf PORTC,0
movfw CREF
bcf STATUS,Z
xorlw D'2'
btfsc STATUS,Z
bsf PORTC,1
movfw CREF
bcf STATUS,Z
xorlw D'4'
btfsc STATUS,Z
bsf PORTC,2
movfw CREF
bcf STATUS,Z
xorlw D'8'

```

```

btfsc STATUS,Z
bsf PORTC,3
movfw CREF
bcf STATUS,Z
xorlw D'16'
btfsc STATUS,Z
bsf PORTC,4
movfw CREF
bcf STATUS,Z
xorlw D'32'
btfsc STATUS,Z
bsf PORTC,5
return

```

motor_Ref:

```

bcf STATUS,RP0
clrf PORTB
call AD_Conv ;5 channels 8 bits A/D conversion
mot bcf STATUS,RP0
call komparasi
call AD_Conv ;5 channels 8 bits A/D conversion
movfw CH4
bcf STATUS,Z
xorwf RREF,W
btfsc STATUS,Z
return
movfw CH4
bcf STATUS,Z
xorwf RREF1,W
btfsc STATUS,Z
return
movfw CH4
bcf STATUS,Z
xorwf RREF2,W
btfsc STATUS,Z
return
goto mot

```

komparasi:

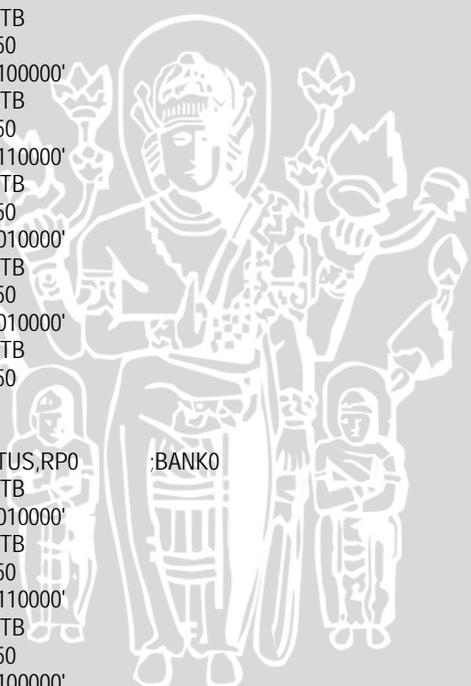
```

bcf STATUS,RP0
movfw CH4
bcf STATUS,C
subwf RREF,W ;CH4-VREF
btfsc STATUS,C

```

	goto	pkanan	
	goto	pkiri	
pkanan	bcf	STATUS,RPO	;BANKO
	clrf	PORTB	
	movlw	B'10000000'	
	movwf	PORTB	
	call	ms_50	
	movlw	B'11000000'	
	movwf	PORTB	
	call	ms_50	
	movlw	B'01000000'	
	movwf	PORTB	
	call	ms_50	
	movlw	B'01100000'	
	movwf	PORTB	
	call	ms_50	
	movlw	B'00100000'	
	movwf	PORTB	
	call	ms_50	
	movlw	B'00110000'	
	movwf	PORTB	
	call	ms_50	
	movlw	B'00010000'	
	movwf	PORTB	
	call	ms_50	
	movlw	B'10010000'	
	movwf	PORTB	
	call	ms_50	
	return		

pkiri	bcf	STATUS,RPO	;BANKO
	clrf	PORTB	
	movlw	B'00010000'	
	movwf	PORTB	
	call	ms_50	
	movlw	B'00110000'	
	movwf	PORTB	
	call	ms_50	
	movlw	B'00100000'	
	movwf	PORTB	
	call	ms_50	
	movlw	B'01100000'	
	movwf	PORTB	
	call	ms_50	
	movlw	B'01000000'	



```
movwf PORTB
call ms_50
movlw B'11000000'
movwf PORTB
call ms_50
movlw B'10000000'
movwf PORTB
call ms_50
movlw B'10010000'
movwf PORTB
call ms_50
return
```

```
-----
;
; kirim_status: ; kirim status uC ke PC
;
;-----
```

```
bcf STATUS,RP0
movfw RREF ;send R-referensi
call send
movfw CREF ;send C-referensi
call send
movfw MODE ;send mode sampling
call send
return
```

```
-----
;
; triger_start:
;
;-----
```

```
bcf STATUS,RP0 ;BANK0
btffs PIR1, RCIF
goto $-1
movfw RCREG
bcf STATUS,Z
xorlw 'M'
btffs STATUS,Z
goto START ; jika tidak 'M' maka restart
return
```

```
-----
;
; triger_stop:
;
;-----
```

```
bcf STATUS,RP0 ;BANK0
btffs PIR1, RCIF
goto st
movfw RCREG
bcf STATUS,Z
xorlw 'D'
btffs STATUS,Z
```

```
st      goto      START      ; jika tidak 'D' maka restart
      nop
      return
```

```
-----
AD_Conv:
```

```
      bcf      STATUS,RP0      ;BANK0
      movlw   0x81      ;ch0
      call    ADC
      movwf   CH0
      movlw   0x89      ;ch1
      call    ADC
      movwf   CH1
      movlw   0x91      ;ch2
      call    ADC
      movwf   CH2
      movlw   0x99      ;ch3
      call    ADC
      movwf   CH3
      movlw   0xA1      ;ch4
      call    ADC
      movwf   CH4
      return
```

```
-----
ADC:
```

```
      movwf   ADCON0
      bcf      PIR1,ADIF
      call    delay_15us
      bsf      ADCON0, 2
      btfs    ADCON0, 2
      goto    $-1
      movfw   ADRESH
      return
```

```
-----
delay_15us:
```

```
      movlw   D'15'
      movwf   CNTAD
      decfsz  CNTAD,f
      goto    $-1
      return
```

```
-----
send_data:
```

```
      movfw   CH0      ;data Ch0
      call    send
      movfw   CH1      ;data ch1
      call    send
```

```
movfw CH2 ;data ch2
call send
movfw CH3 ;data ch3
call send
movfw CH4 ;data ch4
call send
return
```

send:

```
bcf STATUS,RP0
btfss PIR1,TXIF
goto $-1
movwf TXREG
return
```

sec_1:

```
s1 movlw D'10'
movwf CONT2
call ms_100
decfsz CONT2,f
goto s1
return
```

ms_500:

```
1500 movlw D'5'
movwf CONT2
call ms_100
decfsz CONT2,f
goto 1500
return
```

ms_100:

```
1100 movlw D'100'
movwf CONT1
bcf STATUS,RP0
bcf PIR1,TMR1IF
movlw 0XEE
movwf TMR1H
movlw 0X32
movwf TMR1L
bsf T1CON,TMR1ON ;start to count
btfss PIR1,TMR1IF
goto $-1
decfsz CONT1,f
goto 1100
return
```

```

;-----
ms_50:
movlw   D'50'
movwf   CONT1
bcf     STATUS,RP0
bcf     PIR1,TMR1F
movlw   0XEF
movwf   TMR1H
movlw   0XD0
movwf   TMR1L
bsf     T1CON,TMR1ON ;start to count
btfs   PIR1,TMR1F
goto    $-1
decfsz CONT1,f
goto    I50
return

```

```

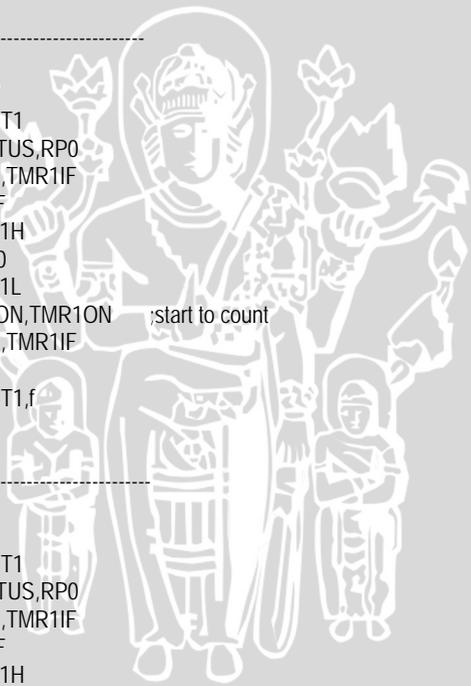
;-----
ms_10:
movlw   D'10'
movwf   CONT1
bcf     STATUS,RP0
bcf     PIR1,TMR1F
movlw   0XEF
movwf   TMR1H
movlw   0XD0
movwf   TMR1L
bsf     T1CON,TMR1ON ;start to count
btfs   PIR1,TMR1F
goto    $-1
decfsz CONT1,f
goto    L_10
return

```

```

;-----
ms_5:
movlw   D'5'
movwf   CONT1
bcf     STATUS,RP0
bcf     PIR1,TMR1F
movlw   0XEF
movwf   TMR1H
movlw   0XD0
movwf   TMR1L
bsf     T1CON,TMR1ON ;start to count
btfs   PIR1,TMR1F
goto    $-1
decfsz CONT1,f

```



```

goto      I5
return

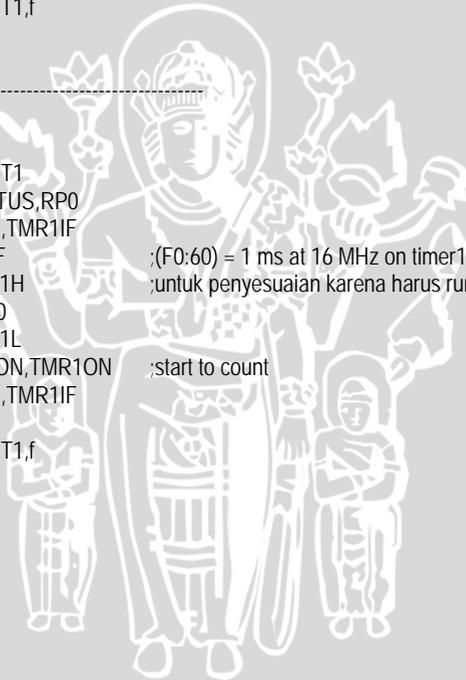
;-----
ms_2:
movlw    D'2'
movwf    CONT1
I2      bcf      STATUS,RP0
        bcf      PIR1,TMR1IF
        movlw   0XEF
        movwf   TMR1H
        movlw   0XD0
        movwf   TMR1L
        bsf     T1CON,TMR1ON ;start to count
        btfss  PIR1,TMR1IF
        goto    $-1
        decfsz CONT1,f
        goto    I2
        return
;-----

```

```

ms_1:
movlw    D'1'
movwf    CONT1
I1      bcf      STATUS,RP0
        bcf      PIR1,TMR1IF
        movlw   0XEF ;(F0:60) = 1 ms at 16 MHz on timer1
        movwf   TMR1H ;untuk penyesuaian karena harus run line lain
        movlw   0XD0
        movwf   TMR1L
        bsf     T1CON,TMR1ON ;start to count
        btfss  PIR1,TMR1IF
        goto    $-1
        decfsz CONT1,f
        goto    I1
        return

```



2.2. Source code user interface (delphi 7)

unit Unit1;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, TeeProcs, TeEngine, ChaRT, Grids, ExtCtrls, Menus, ExtDlgs,
ComCtrls, StdCtrls, CPOrt, comobj, Series, Buttons, OleServer, ExcelXP;

type

```
TForm1 = class(TForm)
  MainMenu1: TMainMenu;
  File1: TMenuItem;
  File_Open: TMenuItem;
  File_Save: TMenuItem;
  File_Print: TMenuItem;
  OpenFileDialog1: TOpenDialog;
  SaveDialog1: TSaveDialog;
  PrintDialog1: TPrintDialog;
  ComPoRT: TComPoRT;
  File_Exit: TMenuItem;
  ChaRT1: TChaRT;
  Series1: TFastLineSeries;
  Mode1: TMenuItem;
  Online1: TMenuItem;
  Offline1: TMenuItem;
  StringGrid1: TStringGrid;
  Button_StaRT: TButton;
  Button_Open: TButton;
  Button_Stop: TButton;
  Panel1: TPanel;
  Series2: TFastLineSeries;
  oExel1: TMenuItem;
  GBSetting: TGroupBox;
  CBNilai: TComboBox;
  CBOrde: TComboBox;
  CBTSamp: TComboBox;
  ENData: TEdit;
  Label1: TLabel;
  Label3: TLabel;
  Label2: TLabel;
  GBDate: TGroupBox;
  Label7: TLabel;
  Label5: TLabel;
  Label8: TLabel;
```



```

Label6: TLabel;
ProgressBar1: TProgressBar;
LJudul: TLabel;
Series3: TFastLineSeries;
Series4: TFastLineSeries;
Series5: TFastLineSeries;
procedure Button_OpenClick(Sender: TObject);
procedure File_OpenClick(Sender: TObject);
procedure File_SaveClick(Sender: TObject);
procedure File_PrintClick(Sender: TObject);
procedure File_ExitClick(Sender: TObject);
procedure FormCreate(Sender: TObject);
Procedure ClearStringGrid;
Procedure ClearBuffer;
procedure delay(lama:cardinal);
procedure ComPoRTxChar(Sender:TObject;Count:integer);
procedure Button_StaRTClick(Sender: TObject);
procedure Button_StopClick(Sender: TObject);
procedure Online1Click(Sender: TObject);
procedure Offline1Click(Sender: TObject);
procedure oExel1Click(Sender: TObject);

private
  { Private declarations }
public
  { Public declarations }
end;

var
  Form1: TForm1;

implementation

uses Printers;
{$R *.DFM}
var
  RSBuffer:string;
  ChInBuf:integer;
  stop:boolean;
  Ref,Cref:byte;
  pesan,cekpesan:string[9];
  CH1,CH2,CH3,CH4,CH5: array[0..1048576] of byte;
  RCH1,RCH2,RCH3,RCH4,RCH5: array[0..1048576] of real; //angka desimal
  sbx,sbxmax:real;
  sigdata,tunda:integer;
  sampl:Char;

```

```
procedure TForm1.FormCreate(Sender: TObject);
```

```
begin
```

```
StringGrid1.FixedRows:=1;
```

```
StringGrid1.FixedCols:=1;
```

```
stringgrid1.Font.Color := clBlack;
```

```
stringgrid1.Font.size := 8;
```

```
stringgrid1.Font.Style := [fsBold];
```

```
stringgrid1.Cells[0,0] := ' NO-SAMP';
```

```
stringgrid1.Cells[0,1] := ' CH-1[V]';
```

```
stringgrid1.Cells[0,2] := ' CH-2[V]';
```

```
stringgrid1.Cells[0,3] := ' CH-3[V]';
```

```
stringgrid1.Cells[0,4] := ' CH-4[V]';
```

```
stringgrid1.Cells[0,5] := ' CH-5[V]';
```

```
end;
```

```
procedure TForm1.ComPoRTRxChar(Sender:TObject;Count:integer);
```

```
var
```

```
str:string;
```

```
begin
```

```
ComPoRT.ReadStr(str,count);
```

```
RSBuffer:=RSBuffer + str;
```

```
ChInBuf:=length(RSBuffer);
```

```
end;
```

```
procedure TForm1.ClearBuffer;
```

```
begin
```

```
RSBuffer:='';
```

```
ChInBuf:=0;
```

```
end;
```

```
Procedure TForm1.ClearStringGrid;
```

```
var
```

```
kolom,baris: integer;
```

```
begin
```

```
for baris :=0 to 5 do
```

```
begin
```

```
for kolom:=1 to sigdata+1 do
```

```
StringGrid1.Cells[kolom,baris]:='';
```

```
end;
```

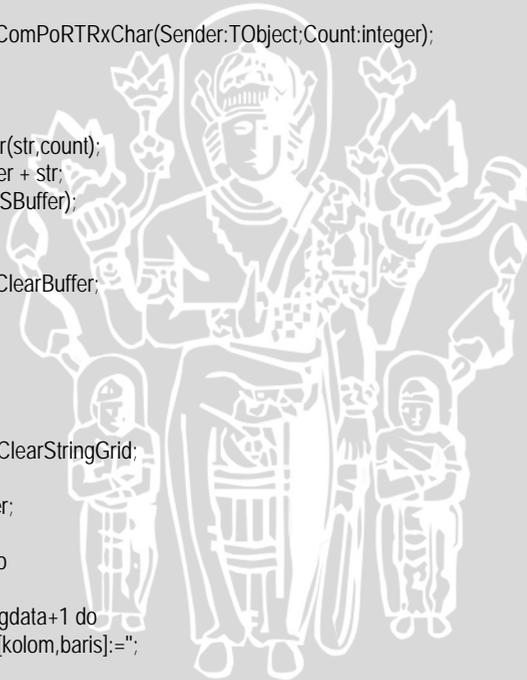
```
end;
```

```
procedure TForm1.Delay(lama:cardinal);
```

```
var
```

```
ref:cardinal;
```

```
begin
```



```
ref:=GetTickCount;  
repeat  
    Application.ProcessMessages;  
until ((GetTickCount-Ref)>=lama);  
end;
```

```
procedure TForm1.Button_OpenClick(Sender: TObject);  
begin
```

```
    if not CompoRT.Connected then CompoRT.Open else close;
```

```
    if (CBNilai.Text='1') then Rref:=255;
```

```
    if (CBNilai.Text='2') then Rref:=230;
```

```
    if (CBNilai.Text='3') then Rref:=204;
```

```
    if (CBNilai.Text='4') then Rref:=179;
```

```
    if (CBNilai.Text='5') then Rref:=153;
```

```
    if (CBNilai.Text='6') then Rref:=127;
```

```
    if (CBNilai.Text='7') then Rref:=102;
```

```
    if (CBNilai.Text='8') then Rref:=77;
```

```
    if (CBNilai.Text='9') then Rref:=51;
```

```
    if (CBOorde.Text='x 10 mHz') then Cref:=32;
```

```
    if (CBOorde.Text='x 100 mHz') then Cref:=16;
```

```
    if (CBOorde.Text='x 1 Hz') then Cref:=8;
```

```
    if (CBOorde.Text='x 10 Hz') then Cref:=4;
```

```
    if (CBOorde.Text='x 100 Hz') then Cref:=2;
```

```
    if (CBOorde.Text='x 1 kHz') then Cref:=1;
```

```
    if (CBTSamp.Text='1 ms') then sampl:='A';
```

```
    if (CBTSamp.Text='2 ms') then sampl:='B';
```

```
    if (CBTSamp.Text='5 ms') then sampl:='C';
```

```
    if (CBTSamp.Text='10 ms') then sampl:='D';
```

```
    if (CBTSamp.Text='50 ms') then sampl:='E';
```

```
    if (CBTSamp.Text='100 ms') then sampl:='F';
```

```
    if (CBTSamp.Text='500 ms') then sampl:='G';
```

```
    if (CBTSamp.Text='1 s') then sampl:='H';
```

```
    pesan:=inttostr(Rref)+'-'+inttostr(Cref)+'-'+sampl;
```

```
    Label5.Caption:=pesan;
```

```
    ClearBuffer;
```

```
    CompoRT.Write(Rref,1);
```

```
    CompoRT.Write(Cref,1);
```

```
    ComPoRT.WriteStr(sampl);
```

```
    repeat
```

```
        Application.ProcessMessages;
```

```
    until ChInBuf >= 3; // nunggu response
```

```
    cekpesan:=inttostr(Ord(RSBuffer[1]))+'-'+inttostr(Ord(RSBuffer[2]))+'-'+RSBuffer[3];
```

```
    Label6.Caption:=cekpesan;
```

```

if (cekpesan=pesan)then
begin
if messagedlg('Communication OK, PUSH START',mtconfirmation,[mbok],0)=mrok
then
begin
Label5.Caption:=cekpesan;
Button_Open.Enabled:=false;
Button_StaRT.Enabled:=true;
Button_Stop.Enabled:=false;
end
end
else
begin
Button_Open.Enabled:=true;
Button_StaRT.Enabled:=false;
Button_Stop.Enabled:=false;
beep;
if messagedlg('Communication Error, Application
Terminated',mtconfirmation,[mbok],0)=mrok
then application.Terminate;
end;
end;

```

```

procedure TForm1.Button_StaRTClick(Sender: TObject); //inti program
var
i :integer;
procedure HITUNG;
begin
CH1[i]:= ord(RSBuffer[(i*5)]);
CH2[i]:= ord(RSBuffer[(i*5+1)]);
CH3[i]:= ord(RSBuffer[(i*5+2)]);
CH4[i]:= ord(RSBuffer[(i*5+3)]);
CH5[i]:= ord(RSBuffer[(i*5+4)]);
RCH1[i]:=((CH1[i]*5/255));
RCH2[i]:=((CH2[i]*5/255));
RCH3[i]:=((CH3[i]*5/255));
RCH4[i]:=((CH4[i]*5/255));
RCH5[i]:=((CH5[i]*5/255));
stringgrid1.Cells[i+1,0] := inttostr(i); //untuk nomor
stringgrid1.Cells[i+1,1] := FormatFloat('0.0000', (RCH1[i]));
stringgrid1.Cells[i+1,2] := FormatFloat('0.0000', (RCH2[i]));
stringgrid1.Cells[i+1,3] := FormatFloat('0.0000', (RCH3[i]));
stringgrid1.Cells[i+1,4] := FormatFloat('0.0000', (RCH4[i]));
stringgrid1.Cells[i+1,5] := FormatFloat('0.0000', (RCH5[i]));
if i=0 then
begin

```

```

Series1.Clear;
Series2.Clear;
Series3.Clear;
Series4.Clear;
Series5.Clear;
end;
Series1.AddXY(i,RCH1[i],"cMaroon");
Series2.AddXY(i,RCH2[i],"cRed");
Series3.AddXY(i,RCH3[i],"cYellow");
Series4.AddXY(i,RCH4[i],"cGreen");
Series5.AddXY(i,RCH5[i],"cBlue");
end;

```

```

begin
Button_Open.Enabled:=false;
Button_Start.Enabled:=false;
Button_Stop.Enabled:=true;
Series1.Clear;
Series2.Clear;
Series3.Clear;
Series4.Clear;
Series5.Clear;
ClearStringGrid;
sigdata:=stRToint(ENData.Text);
sbxmax:=sigdata;
ChaRT1.BottomAxis.Maximum:=sbxmax;
StringGrid1.ColCount:=sigdata+1;

```

```

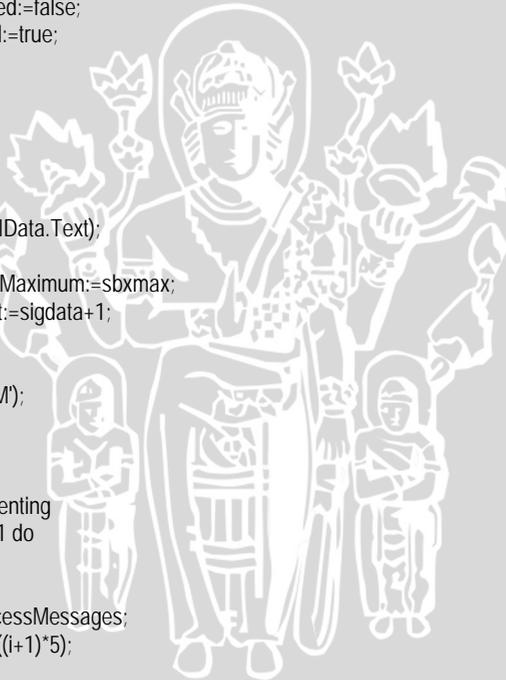
ClearBuffer;
CompoRT.WriteStr('M');
stop:=false;
repeat
ClearStringGrid;
ClearBuffer; //penting
for i:=0 to sigdata-1 do
begin
repeat
Application.ProcessMessages;
until ChInBuf >= ((i+1)*5);
HITUNG;
end
until stop;

```

```

end;
procedure TForm1.File_SaveClick(Sender: TObject);
var
datanya:TextFile;

```



```

jumdat,i:integer;
begin
jumdat:=sigdat;
if savedialog1.Execute then
begin
assignfile(datanya,savedialog1.FileName);
rewrite(datanya);
writeln(datanya,copy(ENData.Text,1,6)); //number of data
writeln(datanya,copy(CBT Samp.Text,1,6)); //sampling time
for i:= 0 to jumdat-1 do
begin
writeln(datanya, CH1[i]);
writeln(datanya, CH2[i]);
writeln(datanya, CH3[i]);
writeln(datanya, CH4[i]);
writeln(datanya, CH5[i]);
end;
closefile(datanya);
end;
end;

```

```

procedure TForm1.File_OpenClick(Sender: TObject);

```

```

var
datanya:TextFile;
i,k:integer;
spl_time,jumdat:string[6];
kanal1,kanal2,kanal3,kanal4,kanal5:string[3];
begin
if opendialog1.Execute then
begin
assignfile(datanya,opendialog1.FileName);
reset(datanya);
readln(datanya,jumdat); //number of data
readln(datanya,spl_time); //sampling time
sigdata:=stRToint(jumdat);
for k:= 0 to sigdata-1 do
begin
readln(datanya,kanal1);
CH1[k]:=stRToint(kanal1);
readln(datanya,kanal2);
CH2[k]:=stRToint(kanal2);
readln(datanya,kanal3);
CH3[k]:=stRToint(kanal3);
readln(datanya,kanal4);
CH4[k]:=stRToint(kanal4);
readln(datanya,kanal5);

```

```

CH5[k]:=stRToint(kanal5);
RCH1[k]:=(CH1[k]*5/255);
RCH2[k]:=(CH2[k]*5/255);
RCH3[k]:=(CH3[k]*5/255);
RCH4[k]:=(CH4[k]*5/255);
RCH5[k]:=(CH5[k]*5/255);
end;
if (spl_time = '1 ms') then tunda:= 10;
if (spl_time = '2 ms') then tunda:= 20;
if (spl_time = '5 ms') then tunda:= 50;
if (spl_time = '10 ms') then tunda:= 100;
if (spl_time = '50 ms') then tunda:= 500;
if (spl_time = '100 ms') then tunda:= 1000;
if (spl_time = '500 ms') then tunda:= 5000;
if (spl_time = '1 s') then tunda:= 10000;
CBTSamp.Text:= spl_time;
ENDData.Text:= jumdat;

ClearBuffer;
ClearStringGrid;
Series1.Clear;
Series2.Clear;
Series3.Clear;
Series4.Clear;
Series5.Clear;
StringGrid1.ColCount:=sigdata+1;
sbxmax:=sigdata;
ChaRT1.BottomAxis.Maximum:=sbxmax;
for i:=0 to sigdata-1 do
begin
  sbx:=i;
  stringgrid1.Cells[i+1,0] := ' '+ inttostr(i);
  stringgrid1.Cells[i+1,1] := FormatFloat('0.0000',(RCH1[i]));
  Series1.AddXY(sbx,RCH1[i],"clMaroon");
  stringgrid1.Cells[i+1,2] := FormatFloat('0.0000',(RCH2[i]));
  Series2.AddXY(sbx,RCH2[i],"clRed");
  stringgrid1.Cells[i+1,3] := FormatFloat('0.0000',(RCH3[i]));
  Series3.AddXY(sbx,RCH3[i],"clYellow");
  stringgrid1.Cells[i+1,4] := FormatFloat('0.0000',(RCH4[i]));
  Series4.AddXY(sbx,RCH4[i],"clGreen");
  stringgrid1.Cells[i+1,5] := FormatFloat('0.0000',(RCH5[i]));
  Series5.AddXY(sbx,RCH5[i],"clBlue");
end;
end;
end;
procedure TForm1.File_PrintClick(Sender: TObject);

```

```

Var tmpH,tmpW,tmpWMargin,tmpHMargin:Longint; { margins }
  OldOrientation:=TPrinter.Orientation;
begin
  if PrintDialog1.Execute then
  begin
    Screen.Cursor := crHourGlass;
    OldOrientation:=Printer.Orientation; { <-- save paper orientation }
    Printer.Orientation:=poLandscape; { <-- Force Horizontal paper }
    try
      Printer.BeginDoc;    { <-- start printer job }
    try
      { CALCULATE HORIZONTAL MARGIN }
      tmpW:=Printer.PageWidth;
      tmpWMargin:=Round(5.0*tmpW/100.0); { <-- 5% margins }
      tmpW:=tmpW-2*tmpWMargin; { <-- left and right margins }
      { CALCULATE VERTICAL MARGIN }
      tmpH:=Printer.PageHeight;
      tmpHMargin:=Round(10.0*tmpH/100.0); { <-- 10% margins }
      tmpH:=Round((tmpH-2*tmpHMargin)/2); { <-- bottom and top margins }
      { print chart }
      ChaRT1.PrintPartial( Rect( tmpWMargin,tmpHMargin,
                                tmpWMargin+tmpW,tmpHMargin+tmpH ) );
      Printer.EndDoc; { <-- end job and print !! }
    except
      on Exception do { just in case an error happens... }
      Begin
        Printer.Abort;
        Printer.EndDoc;
        Raise;
      end;
    end;
  finally
    Printer.Orientation:=OldOrientation; { <-- restore paper orientation }
    Screen.Cursor:=crDefault; { <-- restore cursor }
  end;
end;
end;

procedure TForm1.File_ExitClick(Sender: TObject);
begin
  beep;
  if messagedlg('Are you sure to exit?',mtconfirmation,[mbytes,mbcancel],0)=mryes then
  application.Terminate;
end;
procedure TForm1.Button_StopClick(Sender: TObject);
begin

```

```
CompoRT.WriteStr('D');
Button_Open.Enabled:=true;
Button_StarT.Enabled:=false;
Button_Stop.Enabled:=false;
CompoRT.Close;
ClearBuffer;
stop:=true;
exit;
end;
procedure TForm1.Online1Click(Sender: TObject);
begin
  Button_Open.Enabled:=true;
  CompoRT.ShowSetupDialog;
end;
```

```
procedure TForm1.Offline1Click(Sender: TObject);
begin
```

```
  Button_Open.Enabled:=false;
end;
```

```
procedure TForm1.oExel1Click(Sender: TObject);
```

```
var
```

```
  fa,g:variant;
```

```
  bar:integer;
```

```
begin
```

```
  fa:=createoleobject('Excel.application');
```

```
  fa.visible:=true;
```

```
  fa.workbooks.add;
```

```
  fa.workbooks[1].worksheets[1].name:='DATA-LFIS';
```

```
  g:=fa.workbooks[1].worksheets['DATA-LFIS'];
```

```
  g.cells[1,1]:='NO-SAMP';
```

```
  g.cells[1,2]:='CH-1';
```

```
  g.cells[1,3]:='CH-2';
```

```
  g.cells[1,4]:='CH-3';
```

```
  g.cells[1,5]:='CH-4';
```

```
  g.cells[1,6]:='CH-5';
```

```
  for bar:=2 to sigdata+1 do
```

```
  begin
```

```
    g.cells[bar,1]:=bar-1;
```

```
    g.cells[bar,2]:=RCH1[bar-1];
```

```
    g.cells[bar,3]:=RCH2[bar-1];
```

```
    g.cells[bar,4]:=RCH3[bar-1];
```

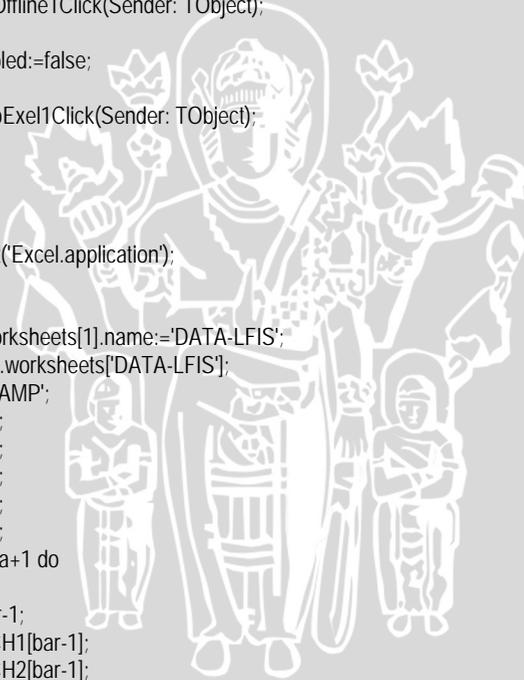
```
    g.cells[bar,5]:=RCH4[bar-1];
```

```
    g.cells[bar,6]:=RCH5[bar-1];
```

```
  end;
```

```
end;
```

```
END.
```

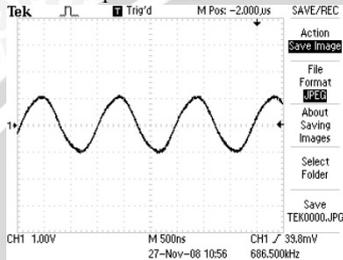


Lampiran 3

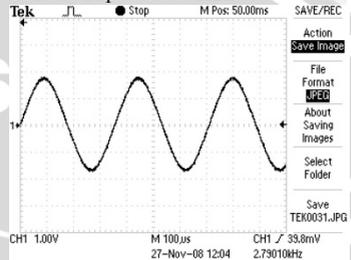
Gambar Sinyal Keluaran pada masing – masing nilai kapasitor (C_T) dan Resistor Pewaktu (R_T)

3.1. Kapasitor C_1 (1 nF)

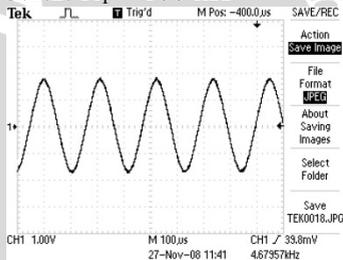
3.1.1. $R_T = 0 \text{ k}\Omega$



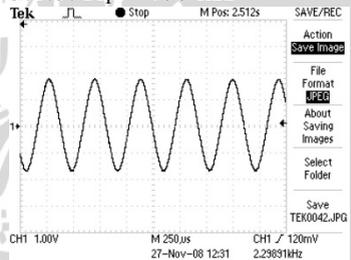
3.1.4. $R_T = 300 \text{ k}\Omega$



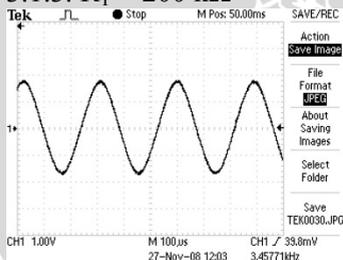
3.1.2. $R_T = 100 \text{ k}\Omega$



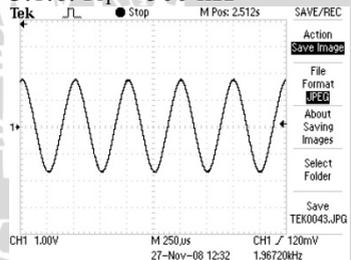
3.1.5. $R_T = 400 \text{ k}\Omega$



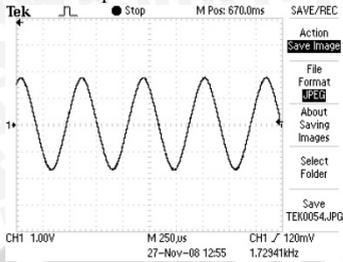
3.1.3. $R_T = 200 \text{ k}\Omega$



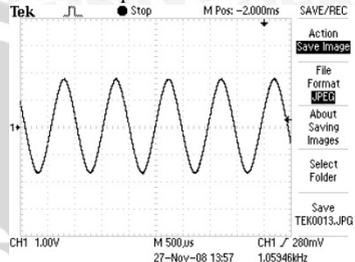
3.1.6. $R_T = 500 \text{ k}\Omega$



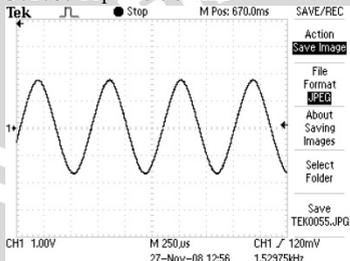
3.1.7. $R_T = 600 \text{ k}\Omega$



3.1.11. $R_T = 1000 \text{ k}\Omega$

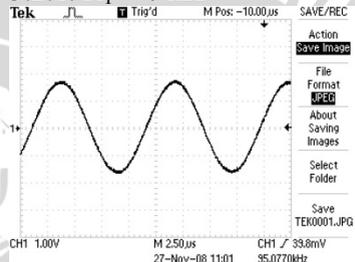


3.1.8. $R_T = 700 \text{ k}\Omega$

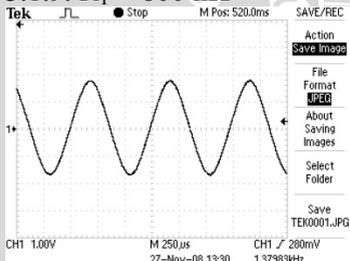


3.2 Kapasitor C2 (10 nF)

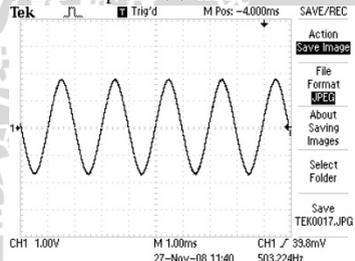
3.2.1. $R_T = 0 \text{ k}\Omega$



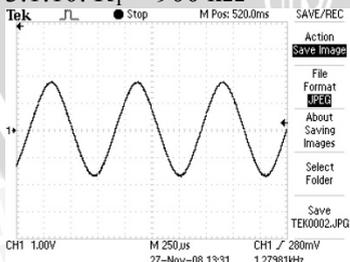
3.1.9. $R_T = 800 \text{ k}\Omega$



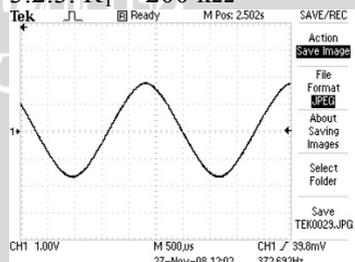
3.2.2. $R_T = 100 \text{ k}\Omega$



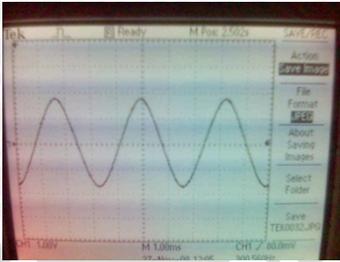
3.1.10. $R_T = 900 \text{ k}\Omega$



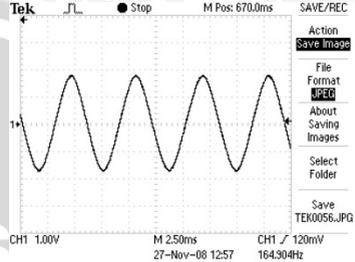
3.2.3. $R_T = 200 \text{ k}\Omega$



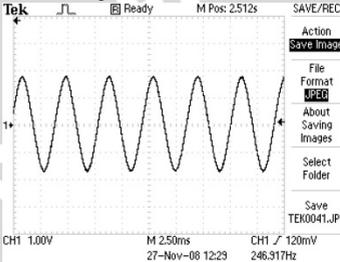
3.2.4. $R_T = 300 \text{ k}\Omega$



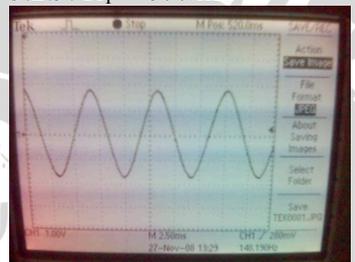
3.2.8. $R_T = 700 \text{ k}\Omega$



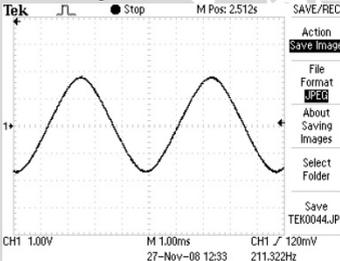
3.2.5. $R_T = 400 \text{ k}\Omega$



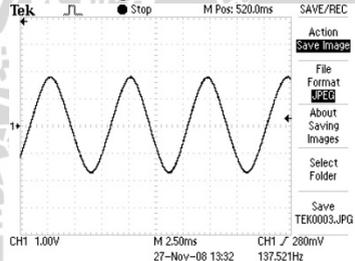
3.2.9. $R_T = 800 \text{ k}\Omega$



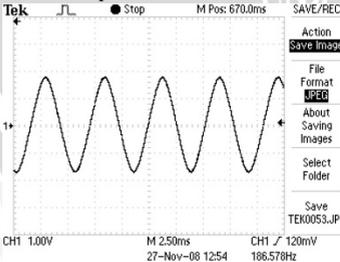
3.2.6. $R_T = 500 \text{ k}\Omega$



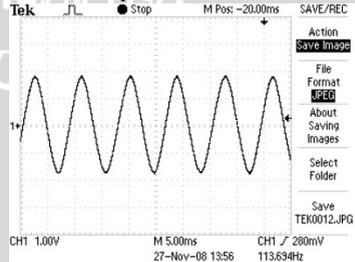
3.2.10. $R_T = 900 \text{ k}\Omega$



3.2.7. $R_T = 600 \text{ k}\Omega$

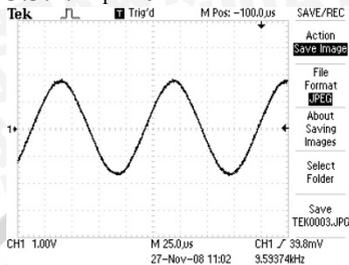


3.2.11. $R_T = 1000 \text{ k}\Omega$

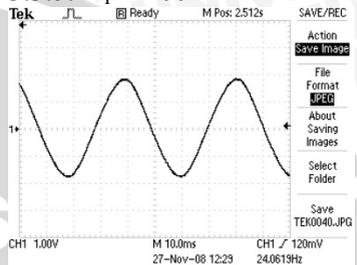


3.3. Kapasitor C3 (100 nF)

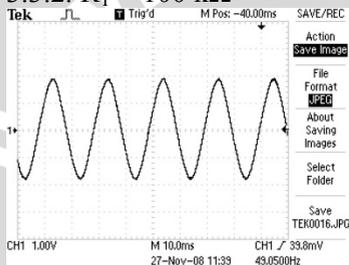
3.3.1. $R_T = 0 \text{ k}\Omega$



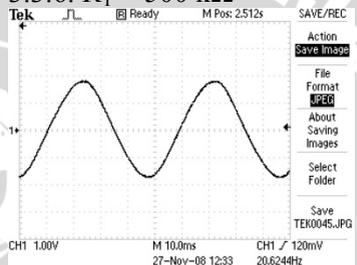
3.3.5. $R_T = 400 \text{ k}\Omega$



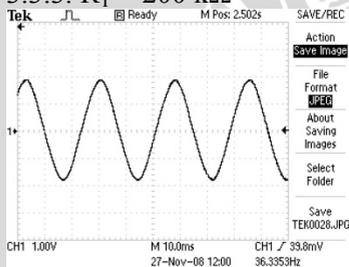
3.3.2. $R_T = 100 \text{ k}\Omega$



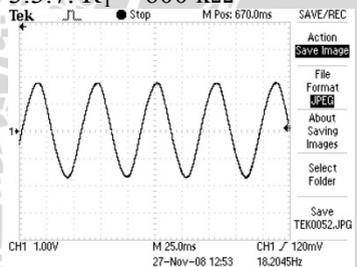
3.3.6. $R_T = 500 \text{ k}\Omega$



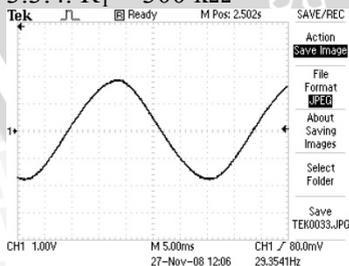
3.3.3. $R_T = 200 \text{ k}\Omega$



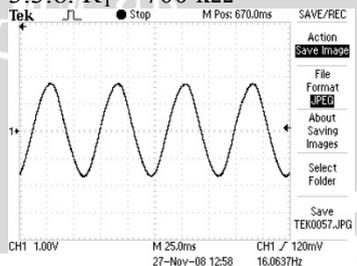
3.3.7. $R_T = 600 \text{ k}\Omega$



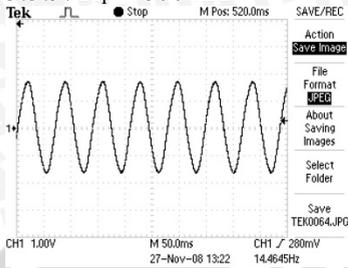
3.3.4. $R_T = 300 \text{ k}\Omega$



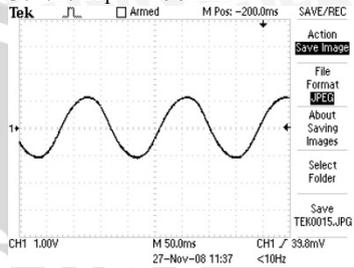
3.3.8. $R_T = 700 \text{ k}\Omega$



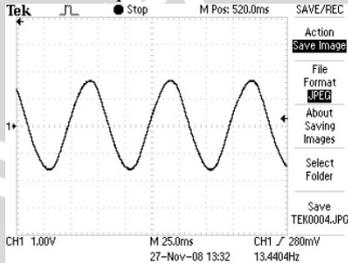
3.3.9. $R_T = 800\text{ k}\Omega$



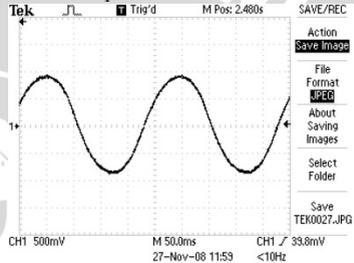
3.4.2. $R_T = 100\text{ k}\Omega$



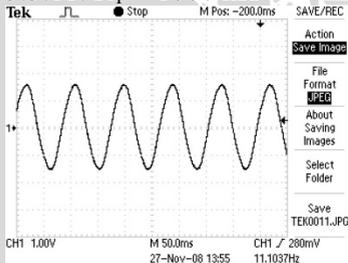
3.3.10. $R_T = 900\text{ k}\Omega$



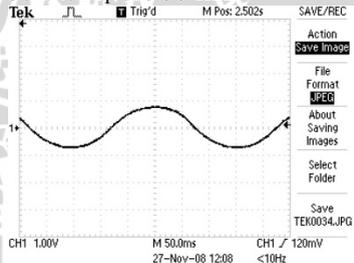
3.4.3. $R_T = 200\text{ k}\Omega$



3.3.11. $R_T = 1000\text{ k}\Omega$

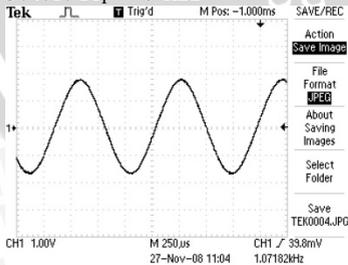


3.4.4. $R_T = 300\text{ k}\Omega$

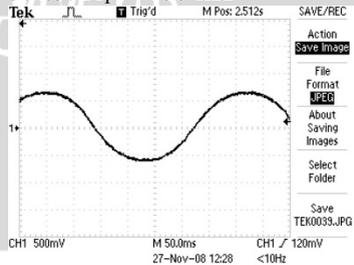


3.4. Kapasitor C4 (1 μf)

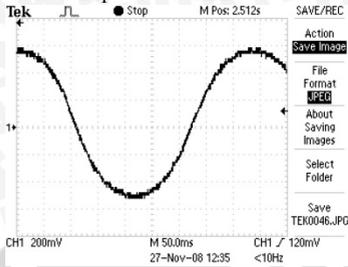
3.4.1. $R_T = 0\text{ k}\Omega$



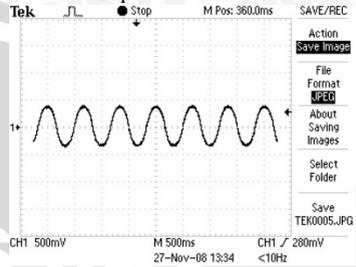
3.4.5. $R_T = 400\text{ k}\Omega$



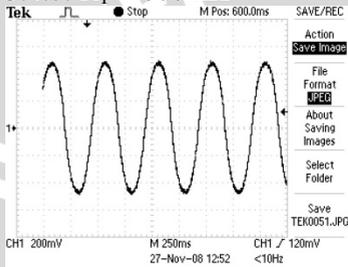
3.4.6. $R_T = 500 \text{ k}\Omega$



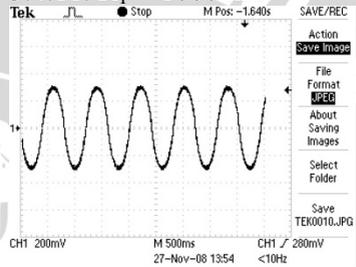
3.4.10. $R_T = 900 \text{ k}\Omega$



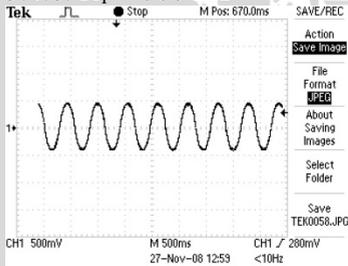
3.4.7. $R_T = 600 \text{ k}\Omega$



3.4.11. $R_T = 1000 \text{ k}\Omega$

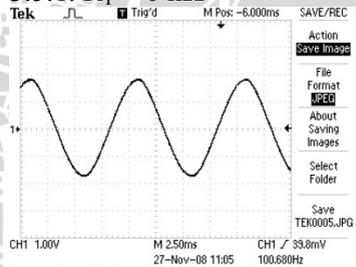


3.4.8. $R_T = 700 \text{ k}\Omega$

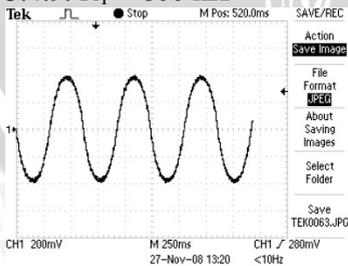


3.5. Kapasitor C5 (10 μ F)

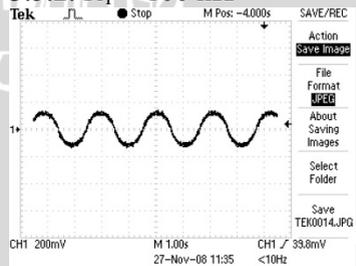
3.5.1. $R_T = 0 \text{ k}\Omega$



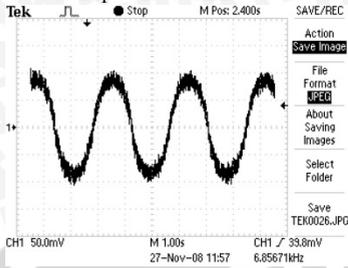
3.4.9. $R_T = 800 \text{ k}\Omega$



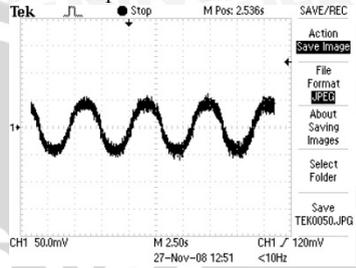
3.5.2. $R_T = 100 \text{ k}\Omega$



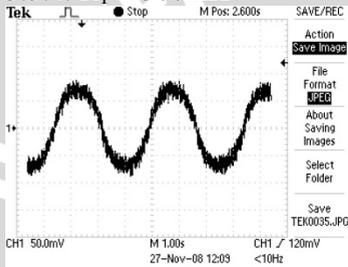
3.5.3. $R_T = 200 \text{ k}\Omega$



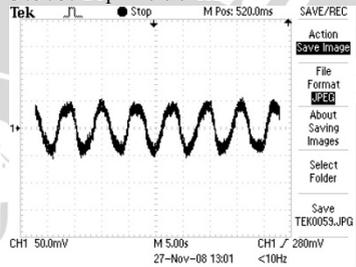
3.5.7. $R_T = 600 \text{ k}\Omega$



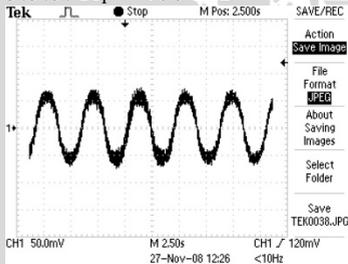
3.5.4. $R_T = 300 \text{ k}\Omega$



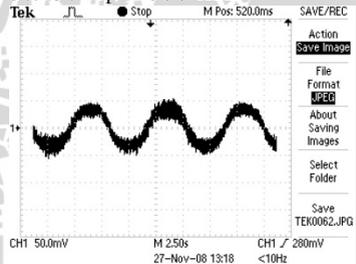
3.5.8. $R_T = 700 \text{ k}\Omega$



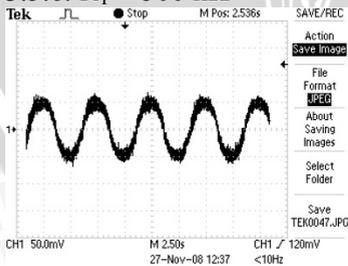
3.5.5. $R_T = 400 \text{ k}\Omega$



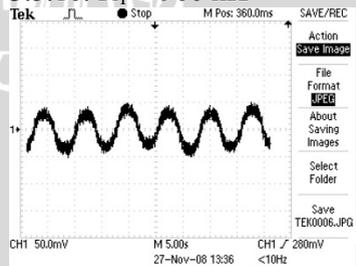
3.5.9. $R_T = 800 \text{ k}\Omega$



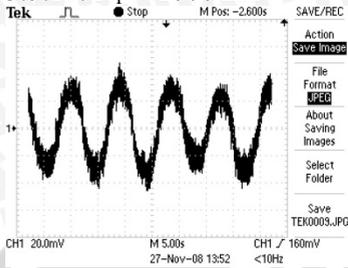
3.5.6. $R_T = 500 \text{ k}\Omega$



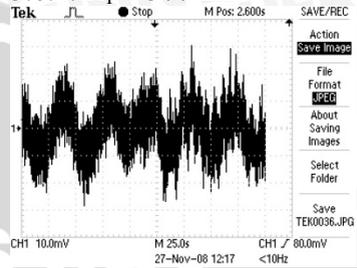
3.5.10. $R_T = 900 \text{ k}\Omega$



3.5.11. $R_T = 1000\text{ k}\Omega$

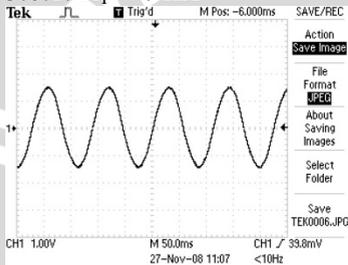


3.6.4. $R_T = 300\text{ k}\Omega$

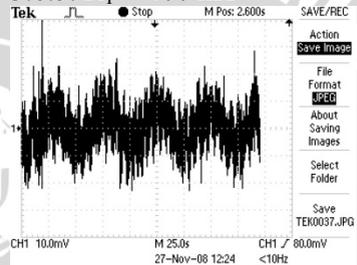


3.6. Kapasitor C6 (100 μ F)

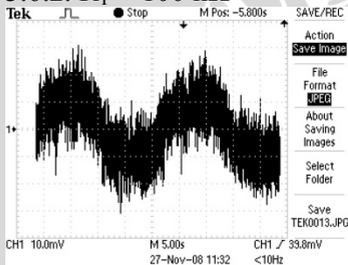
3.6.1. $R_T = 0\text{ k}\Omega$



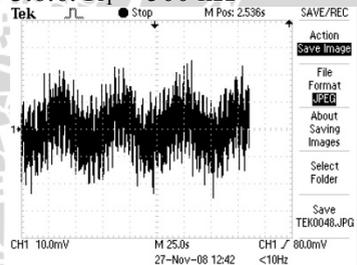
3.6.5. $R_T = 400\text{ k}\Omega$



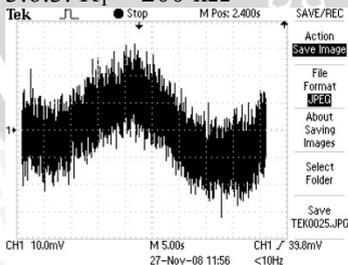
3.6.2. $R_T = 100\text{ k}\Omega$



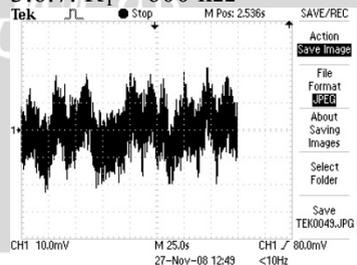
3.6.6. $R_T = 500\text{ k}\Omega$



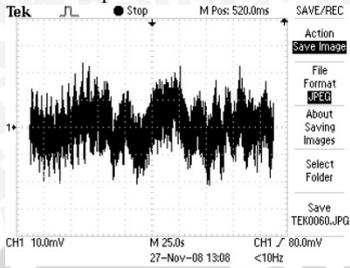
3.6.3. $R_T = 200\text{ k}\Omega$



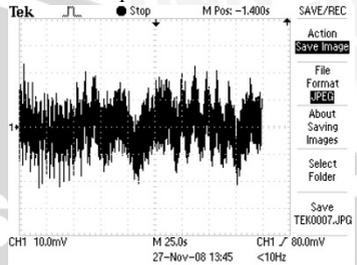
3.6.7. $R_T = 600\text{ k}\Omega$



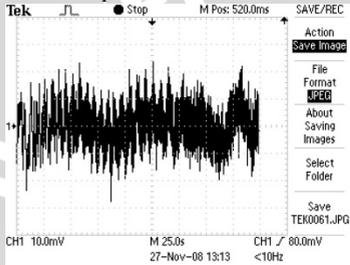
3.6.8. $R_T = 700 \text{ k}\Omega$



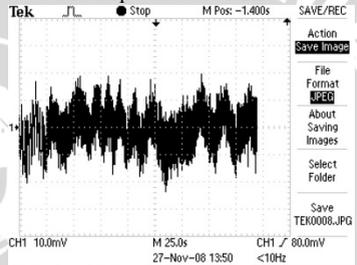
3.6.10. $R_T = 900 \text{ k}\Omega$



3.6.9. $R_T = 800 \text{ k}\Omega$



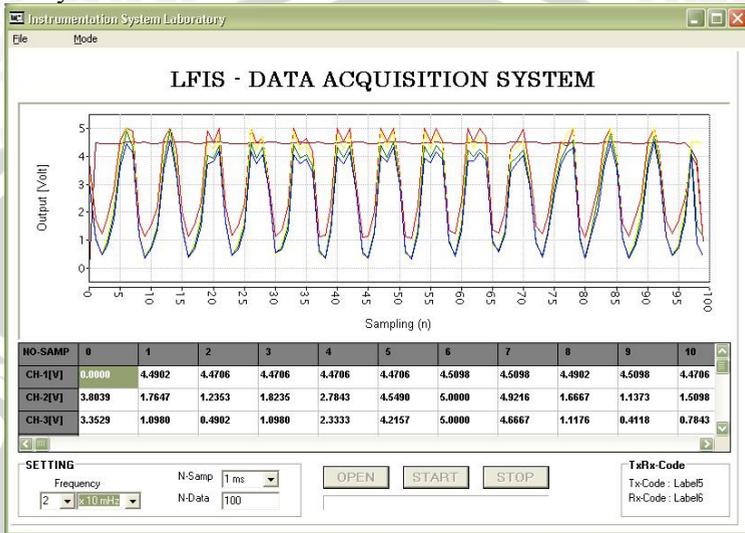
3.6.11. $R_T = 1000 \text{ k}\Omega$



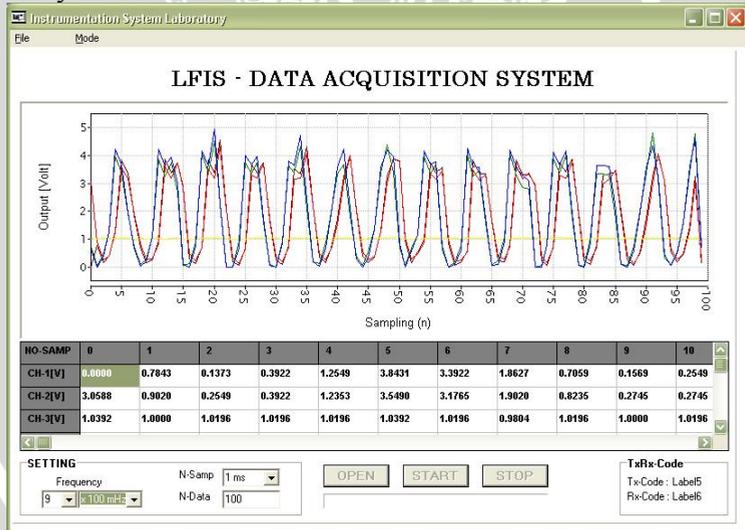
Lampiran 4

Data Hasil Pengujian Perangkat Lunak

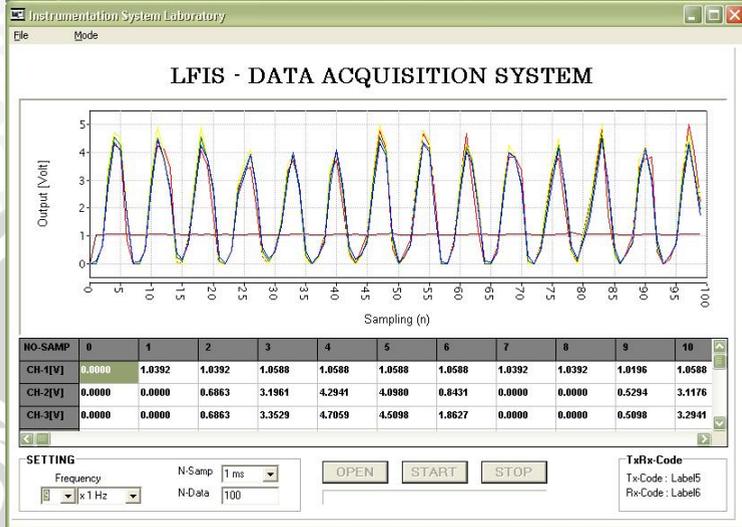
4.1 Sinyal keluaran frekuensi 20 mHz



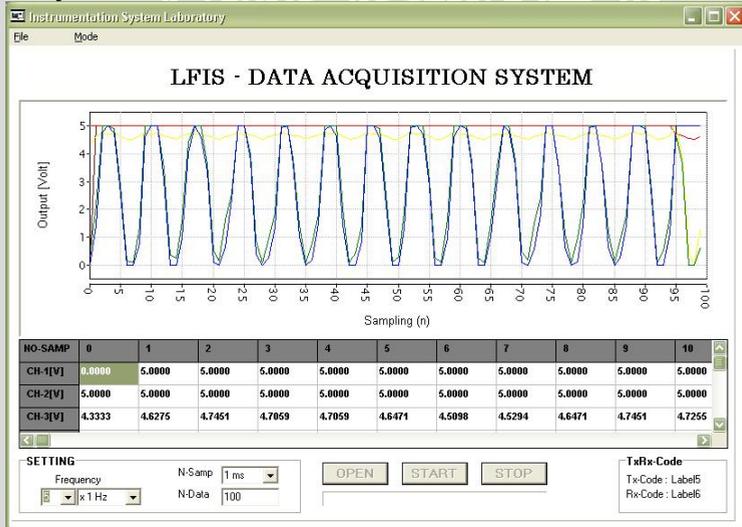
4.2 Sinyal keluaran frekuensi 900 mHz



4.3 Sinyal keluaran frekuensi 9 Hz



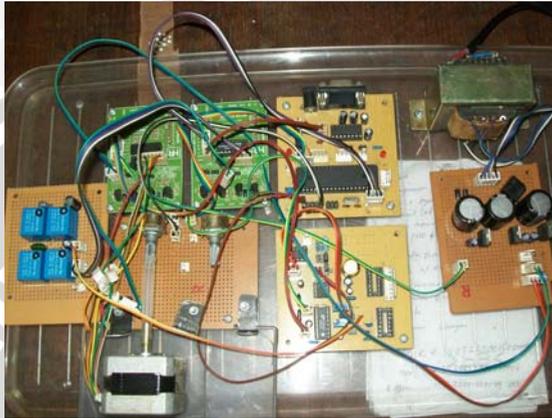
4.4 Sinyal keluaran frekuensi 1 Hz



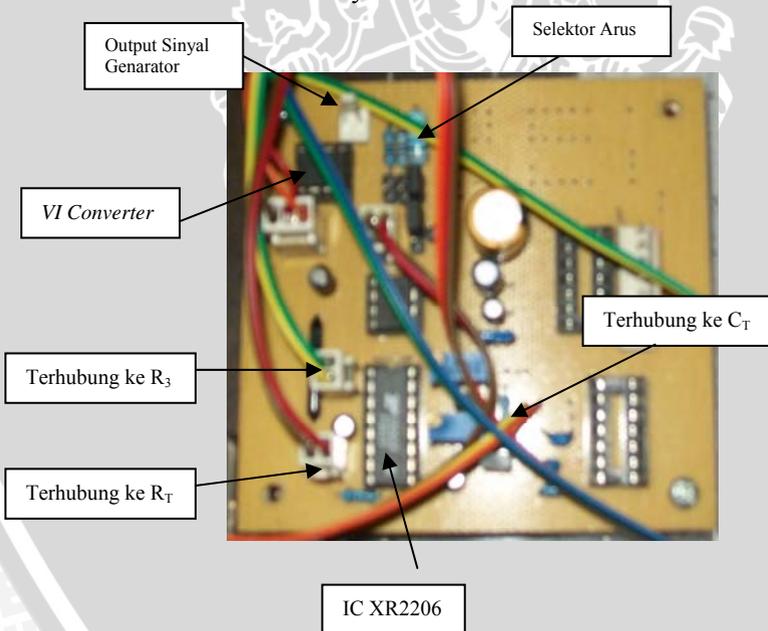
Lampiran 5

Foto alat

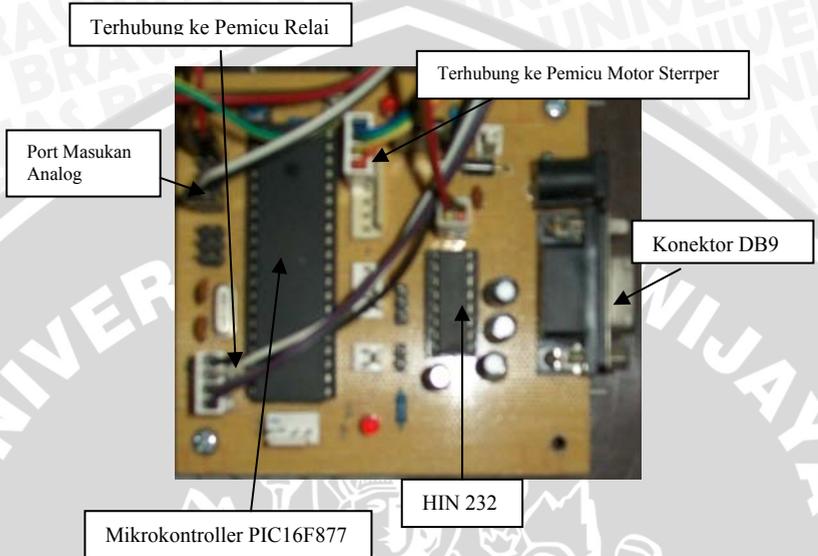
5.1. Foto Alat Keseluruhan



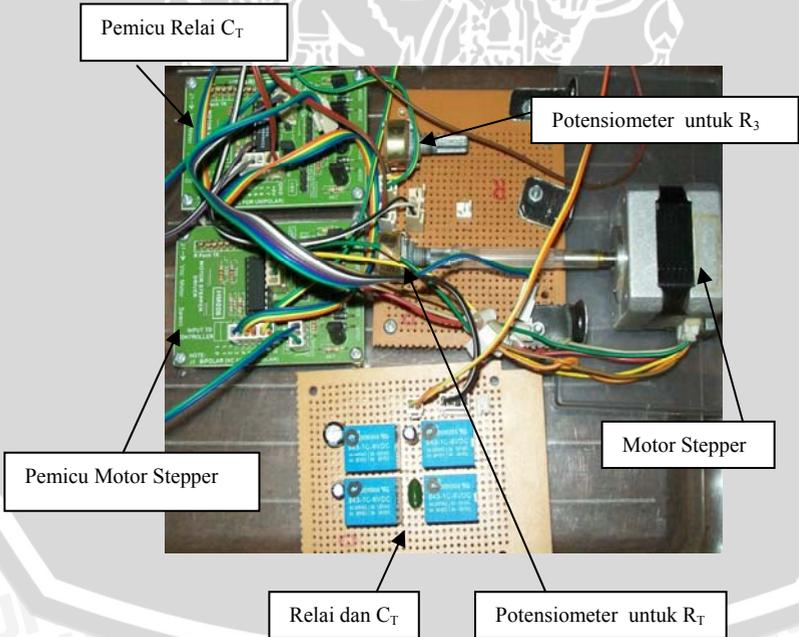
5.2. Modul Generator Sinyal



5.3. Modul Akuisisi Data dan Kontrol



5.4. Selektor Frekuensi Sinyal Keluaran



5.5. Power Supply

