PERANCANGAN ALAT UKUR FREKUENSI, INDUKTANSI, KAPASITANSI (*fLC*) METER BERBASIS MIKROKONTROLER AT90S2313

TUGAS AKHIR

Diajukan sebagai Salah Satu Syarat
untuk Memperoleh Gelar Sarjana
Jurusan Teknik Elektro

Oleh :

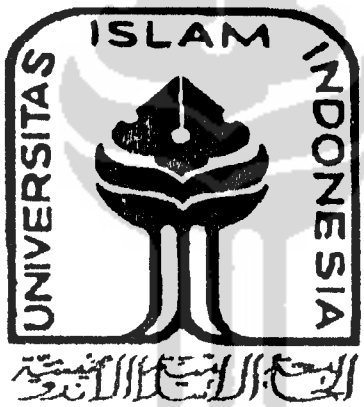
Nama : **Muhammad Supriadi**
No.Mahasiswa : **01 524 096**

JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM INDONESIA
YOGYAKARTA
2007

PERANCANGAN ALAT UKUR
FREKUENSI, INDUKTANSI, KAPASITANSI (*fLC*) METER
BERBASIS MIKROKONTROLER AT90S2313

TUGAS AKHIR

Diajukan sebagai Salah Satu Syarat
untuk Memperoleh Gelar Sarjana
Jurusan Teknik Elektro



Oleh :

Nama : Muhammad Supriadi
No.Mahasiswa : 01 524 096

JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM INDONESIA
YOGYAKARTA
2007

LEMBAR PENGESAHAN PEMBIMBING

PERANCANGAN ALAT UKUR FREKUENSI, INDUKTANSI, KAPASITANSI (*fLC*) METER BERBASIS MIKROKONTROLER AT90S2313

TUGAS AKHIR

Oleh :

Nama : Muhammad Supriadi

No.Mahasiswa : 01 524 096

Yoyakarta, April 2007

Pembimbing I

Ir. Hj. Budi Astuti. MT

Pembimbing II

Medilla Kusriyanto, ST

LEMBAR PENGESAHAN PENGUJI

PERANCANGAN ALAT UKUR
FREKUENSI, INDUKTANSI, KAPASITANSI (*fLC*) METER
BERBASIS MIKROKONTROLER AT90S2313

TUGAS AKHIR

Oleh :

Nama : Muhammad Supriadi
No.Mahasiswa : 01 524 096

Telah Dipertahankan di Depan Sidang Penguji sebagai
Salah Satu Syarat untuk Memperoleh Gelar Sarjana
Jurusan Teknik Elektro Fakultas Teknologi Industri
Universitas Islam Indonesia

Yogyakarta, April 2007

Tim Penguji

Ir, Hj, Budi Astuti, MT
Ketua

Medilla Kusriyanto, ST
Anggota I

Dwi Ana Ratna Wati, ST
Anggota II

Mengetahui,
Ketua Jurusan Teknik Elektro Fakultas Teknologi Industri
Universitas Islam Indonesia

Ir. Yuwono, ST, Msc

Halaman Persembahan

Kupersembahkan Karyaku Ini

Untuk

Kedua Orang Tuaku Tercinta

Ayahanda Sadimin dan Ibunda Laminem

Kakak-kakakku

Sudarso, Sudarsi, Sulastri, Sulasmi, Tirto Waluyo

Adikku

Subono Magribi(Alm)

Istri & Anakku

Indi Huliyana & Aisya Keisa Adilla

Orang-orang yang Sangat Aku Sayangi

Terimakasih semua

MOTTO

Apakah selain hak (kebenaran itu), tidak lain kecuali kesesatan.
(yunus:32)

Janganlah kamu putus harapan dari rahmat Allah. Sesungguhnya Allah dapat mengampunkan semua dosa, sungguh Ia maha pengampun dan penyayang.
(az-zumar:53)

Berpikirlah sebelum berkata-kata, bisa jadi sebuah kata yang kita anggap ringan untuk diucapkan, akan tetapi sangat berat dan menyakitkan bagi orang lain yang mendengarnya.
(my word)

KATA PENGANTAR

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

Alhamdulillah, Puji dan syukur marilah senantiasa kita panjatkan kehadirat Allah SWT, karena hanya berkat rahmat dan karunia-Nyalah akhirnya penulis dapat menyelesaikan Laporan Tugas Akhir ini sebagai syarat untuk meraih gelar Sarjana Teknik di Jurusan Teknik Elektro Universitas Islam Indonesia.Tak lupa Salawat serta salam senantiasa tercurah pada junjungan kita Nabi Besar Muhammad SAW .

Selama pembuatan alat dan penyusunan laporan penulis tidak lepas dari hambatan dan keterbatasan, namun berkat bantuan dan dukungan dari berbagai pihak masalah tersebut dapat diatasi. Oleh karena itu, sudah sepatutnya penulis mengucapkan terima kasih kepada :

1. Allah SWT, Dzat yang maha Rahman dan Rahim, atas ijin dan kuasaNya lah laporan ini dapat diselesaikan.
2. Rasulullah Muhammad SAW sebagai Uswatun Hasanah bagi seluruh umat.
3. Bapak Tito Yuwono, ST. M.sc selaku Ketua Jurusan Teknik Elektro Fakultas Teknologi industri Universitas Islam Indonesia.
4. Ibu Ir. Hj. Budi Astuti, MT selaku dosen pembimbing Tugas Akhir I
5. Bapak Medila kusriyanto,ST selaku dosen pembimbing Tugas Akhir II.
6. Bapak, Mama dan kakak-kakakku tercinta yang selalu mendoakan, memberikan dukungan dan semangat selama ini.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

ABSTRAKSI

Di pasaran Indonesia, sulit ditemui FLC Meter digital, dan jika ada, harganya masih sangat mahal. Untuk kebutuhan perancangan alat yang presisi, alat ini sangat dibutuhkan. Mengingat pentingnya kebutuhan alat ukur dalam bidang elektronika yang dapat untuk mengukur L(induktasi), C(Kapasitansi), dan F(Frekuensi), maka diperlukan alat FLC Meter Digital untuk keperluan instrumentasi. Penelitian dilakukan dengan perancangan FLC Meter Digital berbasis Mikrokontroler AT90S2313, yang kemudian dilakukan eksperimen dan pengujian kemampuan pengukuran kapasitansi, pengukuran induktansi, dan penghitungan frekuensi. Dalam penelitian juga diteliti signal keluaran dari osilator dan kharakteristik system. Rancangan FLC Meter Digital berbasis Mikrokontroler AT90S2313 yang telah direalisasikan dalam bentuk *prototype* memiliki beberapa bagian utama, yaitu L/C Switch, F Switch, Callibration Switch, Display LCD 2 baris, dan pengendali utama yang menggunakan AT90S2313. Pada rancangan tersebut, FLC memiliki tiga macam pengukuran, yaitu pengukuran frekuensi, pengukuran induktansi dan kapasitansi. FLC Meter Digital berbasis Mikrokontroler AT90S2313 memiliki kemampuan penghitungan kapasitansi dengan nilai *error* rata-rata 0,44%, dan dapat melakukan pengukuran 0pF hingga 2000pF. FLC Meter Digital berbasis Mikrokontroler AT90S2313 memiliki kemampuan penghitungan induktansi dengan nilai *error* rata-rata 5,51% dan telah mampu menghitung nilai induktansi 0nH hingga 650uH. FLC Meter Digital berbasis Mikrokontroler AT90S2313 memiliki kemampuan penghitungan frekuensi dengan nilai *error* maksimum 0,08%. FLC Meter tersebut telah teruji mampu untuk mengukur frekuensi hingga 2,1 MHz. Dengan sumber tegangan 8,5V, FLC Meter Digital berbasis Mikrokontroler AT90S2313 membutuhkan arus 57,6mA, dan memiliki daya 0,4896 watt, yang membuat kondisi FLC Meter dapat bekerja dengan normal.

BAB I

PENDAHULUAN

1.1. LATAR BELAKANG MASALAH

Untuk kemajuan penelitian dan pengembangan, diperlukan dukungan berupa alat instrumentasi yang memadai. Berbagai variasi alat instrument elektronika banyak beredar di pasaran dunia. Perlu disadari bahwa instrumentasi yang dimiliki Indonesia telah cukup jauh tertinggal dengan negara-negara maju.

Alat instrumentasi yang diadakan oleh para peneliti di negara maju tentu saja berdasarkan kebutuhan dan pengalaman panjang yang telah mereka lalui. Sehingga alat instrumentasi yang bervariasi tersebut, bagi pemula akan menjadikan bahan pemikiran yang sulit untuk dicerna. Tetapi, bagi para ahli dengan jam terbang tinggi, mereka dengan mudahnya menentukan instrumen mana saja yang mereka butuhkan, bahkan mereka dapat menentukan sebuah instrumen yang belum ada sebelumnya, dan mereka pikir harus ada.

Ketika penelitian ini dibuat, di Lab Elektronika UII belum ditemukan alat ukur Induktansi dan Kapasitansi. Hal ini menarik perhatian untuk dilakukan penelitian dan perancangan untuk kebutuhan instrumentasi yang belum ada tersebut.

Di sisi lain, ketika ditemui para perancang dan *maintenance* sistem komunikasi, mereka sangat memerlukan beberapa instrument, diantaranya adalah L meter , Frekuensi meter, dan C meter. Hal ini mengingat, dalam perancangan

sistem RF, digunakan komponen utama yang sangat sensitif, yaitu L dan C. Kesalahan sedikit pada penentuan nilai L dan C akan cukup berpengaruh pada hasil sistem komunikasi tersebut. Bila nilai L pada osilator terlalu besar atau terlalu kecil, maka nilai frekuensi osilasi akan berubah pula.

Sebagai contoh lain dalam bidang komunikasi, yaitu ketika seorang teknisi alat komunikasi menemukan sebuah kapasitor yang dicurigai bergeser nilai kapasitansinya karena rusak, maka nilai kapasitansi itu tak akan mudah diukur dengan multi meter yang ada pada umumnya. Nilai kapasitansi itu hanya bisa diukur dengan C Meter. Padahal dalam bidang telekomunikasi, penggantian kapasitor karena bergesernya nilai kapasitansi adalah cukup sering dilakukan.

Dengan demikian dapat kita ketahui betapa pentingnya kebutuhan alat ukur dalam bidang elektronika yang dapat untuk mengukur L(induktasi), C(Kapasitansi), dan F(Frekuensi).

1.2. RUMUSAN MASALAH

Berdasarkan latar belakang masalah, maka ditemukan rumusan masalah yang perlu ditemukan solusinya, yaitu bagaimanakah merancang, membuat dan unjuk kerja FLC (Frekuensi, Induktansi, Kapasitansi) Meter berbasis Mikrokontroler Atmel, AT90S231.

1.3. BATASAN MASALAH

Dalam merancang FLC Meter berbasis Mikrokontroler AT90S2313 diberikan batasan masalah sebagai berikut, Pengendalian menggunakan

mikrokontroler Atmel AVR AT90S2313 dan pengukuran dilakukan dengan menampilkan nilai frekuensi, kapasitansi, dan induktansi.

Nilai (*range*) pengukuran pada ***fLC*** meter adalah :

1) Untuk pengukuran frekuensi meter **0 Hz – 2.1 MHz**
2) Untuk pengukuran frekuensi meter **2,2 uH – 150 uH**
3) Untuk pengukuran frekuensi meter **10 pF – 1500 pF**

1.4. TUJUAN PENELITIAN

Tujuan penelitian ini adalah untuk menghasilkan sebuah alat ukur yang mampu digunakan sebagai pengukur nilai Frekuensi, Induktasi, dan Kapasitansi.,serta menguji alat ukur yang telah dihasilkan sehingga dapat diperoleh informasi kemampuan/spesifikasi dan unjuk kerja alat ukur.

1.5. SISTEMATIKA PENULISAN

Penulisan Laporan Penelitian ini meliputi 5 bab yang tersusun dengan susunan sebagai berikut:

1.5.1. BAB I : PENDAHULUAN

Bab ini membahas tentang masalah apa saja yang muncul dan sisi lain gejala yang memungkinkan untuk diambil sebagai solusi atas masalah tersebut. Yang kemudian hal tersebut dimasukkan dalam Latar Belakang masalah, selanjutnya dalam bab ini akan dibahas rumusan masalah, dan juga tujuan penelitian untuk memperjelas langkah penelitian.

1.5.2. BAB II : LANDASAN TEORI

Semua teori yang diperlukan dalam perancangan *hardware* maupun *software* akan dibahas dalam bab ini. Dengan demikian dapat dijadikan acuan untuk langkah penelitian selanjutnya.

1.5.3. BAB III : PERANCANGAN ALAT

Dalam bab ini akan dibahas keseluruhan perancangan alat, *hardware* maupun *software*. Selanjutnya akan dibahas dan dilakukan analisis masalah implementasinya.

1.5.4. BAB IV : PENGUJIAN DAN ANALISA DATA

Alat yang telah dirancang, diimplementasikan, diuji dan dianalisis atas hasil pengujian , yang hasilnya dimasukkan dalam BAB IV. Dalam BAB ini akan menjadi acuan utama dalam penarikan kesimpulan.

1.5.5. BAB V : PENUTUP

Pada bab ini berisi tentang kesimpulan atas rancangan dan hasil pengamatan, yang selanjutnya dituliskan saran-saran berkaitan dengan penelitian yang telah dilakukan.

BAB II

DASAR TEORI

2.1 Induktor

Dalam teori fisika dikenal medan listrik dengan simbol **B** dan satuannya Tesla (T). Besar akumulasi medan listrik B pada suatu luas area **A** tertentu difenisikan sebagai besar *magnetic flux*. Simbol yang biasa digunakan untuk menunjukkan besar *magnetic flux* ini adalah ϕ dan satuannya Weber (Wb = $T.m^2$). Secara matematis besarnya adalah :

$$\phi = BA \tag{2.1}$$

Lalu bagaimana jika kawat tembaga itu dililitkan membentuk koil atau kumparan. Jika kumparan tersebut dialiri listrik maka tiap lilitan akan saling menginduksi satu dengan yang lainnya. Medan listrik yang terbentuk akan segaris dan saling menguatkan. Komponen yang seperti inilah yang dikenal dengan induktor selenoid.

Berdasarkan teori medan magnet, induktor adalah komponen yang dapat menyimpan energi magnetik. Energi ini direpresentasikan dengan adanya tegangan **emf** (*electromotive force*) jika induktor dialiri listrik. Secara matematis tegangan emf ditulis :

$$E = -L\frac{di}{dt} \qquad (2.2)$$

Jika dibandingkan dengan rumus hukum Ohm **V=RI**, maka kelihatan ada kesamaan rumus. Jika R disebut resistansi dari resistor dan V adalah besar tegangan jepit jika resistor dialiri listrik sebesar I. Maka **L** adalah induktansi dari induktor dan **E** adalah tegangan yang timbul jika induktor dilairi listrik. Tegangan emf di sini adalah respon terhadap perubahan arus fungsi dari waktu terlihat dari rumus **di/dt**. Sedangkan bilangan negatif sesuai dengan hukum **Lenz** yang mengatakan efek induksi cenderung melawan perubahan yang menyebabkannya. Hubungan antara emf dan arus inilah yang disebut dengan **induktansi**, dan satuan yang digunakan adalah **(H) Henry**.

Arus listrik yang melewati kabel, jalur-jalur pcb dalam suatu rangkain berpotensi untuk menghasilkan medan induksi. Ini yang sering menjadi pertimbangan dalam mendesain pcb supaya bebas dari efek induktansi terutama jika *multilayer*. Tegangan emf akan menjadi penting saat perubahan arusnya fluktuatif. Efek emf menjadi signifikan pada sebuah induktor, karena perubahan arus yang melewati tiap lilitan akan saling menginduksi. Ini yang dimaksud dengan *self-induced*. Secara matematis induktansi pada suatu induktor dengan jumlah lilitan sebanyak **N** adalah akumulasi flux magnet untuk tiap arus yang melewatinya :

$$L = \frac{N\phi}{i} \tag{2.3}$$

Gambar 2.1. Induktor Solenoid

Dari pemahaman fisika, elektron yang bergerak akan menimbulkan medan elektrik di sekitarnya. Berbagai bentuk kumparan, persegi empat, setegah lingkaran ataupun lingkaran penuh, jika dialiri listrik akan menghasilkan medan listrik yang berbeda. Penampang induktor biasanya berbentuk lingkaran, sehingga diketahui besar medan listrik di titik tengah lingkaran adalah :

$$B = \mu\mu_0 ni \tag{2.4}$$

Jika dikembangkan, n adalah jumlah lilitan N relatif terhadap panjang induktor l. Secara matematis ditulis :

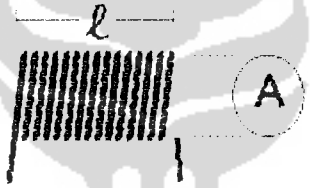
$$n = \frac{N}{\ell} \tag{2.5}$$

Lalu i adalah besar arus melewati induktor tersebut. Ada simbol μ yang dinamakan permeability dan μ_0 yang disebut permeability udara vakum. Besar

permeability μ tergantung dari bahan inti (*core*) dari induktor. Untuk induktor tanpa inti (*air winding*) $\mu = 1$.

Jika rumus-rumus di atas di subsitusikan maka rumus induktansi (rumus 3) dapat ditulis menjadi :

$$L = \frac{\mu\ \mu_0 N^2 A}{\ell} \qquad (2.6)$$



Gambar 2.2. Induktor Solenoid dengan inti (*core*)

L : induktansi dalam H (Henry)

μ : permeability inti (*core*)

μ_0 : permeability udara vakum

μ_0 : $4\pi \times 10^{-7}$

N : jumlah lilitan induktor

A : luas penampang induktor (m^2)

l : panjang induktor (m)

Rumus 2.6 tersebut di atas adalah rumus untuk menghitung nilai induktansi dari sebuah induktor. Tentu saja rumus ini bisa dibolak-balik untuk menghitung jumlah lilitan induktor jika nilai induktansinya sudah ditentukan.

Induktor sangat banyak digunakan dalam elektronika industri dan telekomunikasi. Induktor dengan inti udara memiliki induktansi yang kecil dengan orde *milihenry* dan digunakan untuk frekuensi radio (di atas 1 MHz). Biasanya digunakan untuk radio *tuning circuit* dan *filter circuit*. Induktor berinti besi memiliki induktansi lebih tinggi dengan orde *henry*. Biasa digunakan untuk aplikasi frekuensi rendah, umumnya digunakan pada frekuensi di bawah 100 KHz. Induktor berinti ferrit dioperasikan pada frekuensi antara 100 KHz dan 100MHz.

Induktor memiliki besaran yang disebut induktansi dengan lambang L, dengan satuan *henry*. Satu *henry* didefinisikan sebagai induktansi yang dihasilkan beda potensial 1 Volt ketika terjadi perubahan arus sebesar 1 A.

Induktor yang dihubung seri akan memiliki persamaan sebagai berikut :

$$L_{seri} = L1 + L2 + ... + Ln \qquad (2.7)$$

Maka, semakin banyak induktor yang dihubung seri, maka akan semakin besar pula nilai induktansinya.

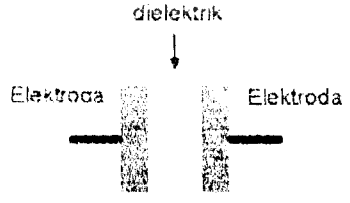
Sedangkan induktor yang dirangkai pararel, maka akan sesuai dengan persamaan berikut:

$$\frac{1}{L_{Paralel}} = \frac{1}{L_1} + \frac{1}{L2} + \ldots\ldots\ldots + \frac{1}{L_n} \qquad (2.8)$$

2.2 Kapasitor

2.2.1 Prinsip Dasar

Kapasitor adalah komponen elektronika yang dapat menyimpan muatan listrik. Struktur sebuah kapasitor terbuat dari 2 buah plat metal yang dipisahkan oleh suatu bahan dielektrik. Bahan-bahan dielektrik yang umum dikenal misalnya udara vakum, keramik, gelas dan lain-lain. Jika kedua ujung plat metal diberi tegangan listrik, maka muatan-muatan positif akan mengumpul pada salah satu kaki (elektroda) metalnya dan pada saat yang sama muatan-muatan negatif terkumpul pada ujung metal yang satu lagi. Muatan positif tidak dapat mengalir menuju ujung kutup negatif dan sebaliknya muatan negatif tidak bisa menuju ke ujung kutup positif, karena terpisah oleh bahan dielektrik yang non-konduktif. Muatan elektrik ini "tersimpan" selama tidak ada konduksi pada ujung-ujung kakinya. Di alam bebas, phenomena kapasitor ini terjadi pada saat terkumpulnya muatan-muatan positif dan negatif di awan.



Gambar 2.3. Prinsip Dasar Kapasitor

2.2.2 Kapasitansi

Kapasitansi didefenisikan sebagai kemampuan dari suatu kapasitor untuk dapat menampung muatan elektron. Coulombs pada abad 18 menghitung bahwa 1 coulomb = 6.25×10^{18} elektron. Kemudian Michael Faraday membuat postulat bahwa sebuah kapasitor akan memiliki kapasitansi sebesar 1 farad jika dengan tegangan 1 volt dapat memuat muatan elektron sebanyak 1 coulombs. Dengan rumus dapat ditulis :

$$\mathbf{Q = CV} \qquad (2.9)$$

Q = muatan elektron dalam C (coulombs)

C = nilai kapasitansi dalam F (farads)

V = besar tegangan dalam V (volt)

Dalam praktek pembuatan kapasitor, kapasitansi dihitung dengan mengetahui luas area plat metal (A), jarak (t) antara kedua plat metal (tebal dielektrik) dan konstanta (k) bahan dielektrik. Dengan rumusan dapat ditulis sebagai berikut :

$$\mathbf{C = (8.85 \times 10^{-12})\,(k\,A/t)} \qquad (2.10)$$

Untuk rangkain elektronik praktis, satuan farads adalah sangat besar sekali. Umumnya kapasitor yang ada di pasar memiliki satuan **uF** (10^{-6} **F**), **nF** (10^{-9} **F**) dan **pF** (10^{-12} **F**). Konversi satuan penting diketahui untuk memudahkan membaca besaran sebuah kapasitor. Misalnya 0.047uF dapat juga dibaca sebagai 47nF, atau contoh lain 0.1nF sama dengan 100pF.

2.2.3 Jenis Kapasitor

Kapasitor terdiri dari beberapa tipe, tergantung dari bahan dielektriknya. Untuk lebih sederhana dapat dibagi menjadi 3 bagian, yaitu kapasitor electrostatic, electrolytic dan electrochemical.

Kapasitor electrostatic adalah kelompok kapasitor yang dibuat dengan bahan dielektrik dari keramik, film dan mika. Keramik dan mika adalah bahan yang popular serta murah untuk membuat kapasitor yang kapasitansinya kecil. Tersedia dari besaran pF sampai beberapa uF, yang biasanya untuk aplikasi rangkaian yang berkenaan dengan frekuensi tinggi. Termasuk kelompok bahan dielektrik film adalah bahan-bahan material seperti polyester (polyethylene terephthalate atau dikenal dengan sebutan mylar), polystyrene, polyprophylene, polycarbonate, metalized paper dan lainnya. Mylar, MKM, MKT adalah beberapa contoh sebutan merek dagang untuk kapasitor dengan bahan-bahan dielektrik film. Umumnya kapasitor kelompok ini adalah non-polar.

Kelompok kapasitor electrolytic terdiri dari kapasitor-kapasitor yang bahan dielektriknya adalah lapisan metal-oksida. Umumnya kapasitor yang termasuk kelompok ini adalah kapasitor polar dengan tanda + dan - di badannya. Mengapa kapasitor ini dapat memiliki polaritas, adalah karena proses pembuatannya menggunakan elektrolisa sehingga terbentuk kutup positif anoda dan kutup negatif katoda.

Satu jenis kapasitor lain adalah kapasitor electrochemical. Termasuk kapasitor jenis ini adalah batere dan accu. Pada kenyataanya batere dan accu

adalah kapasitor yang sangat baik, karena memiliki kapasitansi yang besar dan arus bocor (leakage current) yang sangat kecil. Tipe kapasitor jenis ini juga masih dalam pengembangan untuk mendapatkan kapasitansi yang besar namun kecil dan ringan, misalnya untuk applikasi mobil elektrik dan telepon selular.

2.2.4 Konfigurasi Seri / Paralel

Hubungan paralel pada kapasitor akan menghasilkan nilai kapasitansi Cp sesuai persamaan :

$$C_P = C_1 + C_2 \tag{2.11}$$

Sedangkan kapasitor yang dihubung seri akan menghasilkan nilai kapasitansi Cs sesuai dengan persamaan :

$$\frac{1}{C_{Seri}} = \frac{1}{C_1} + \frac{1}{C_2} + \ldots\ldots + \frac{1}{C_n} \tag{2.12}$$

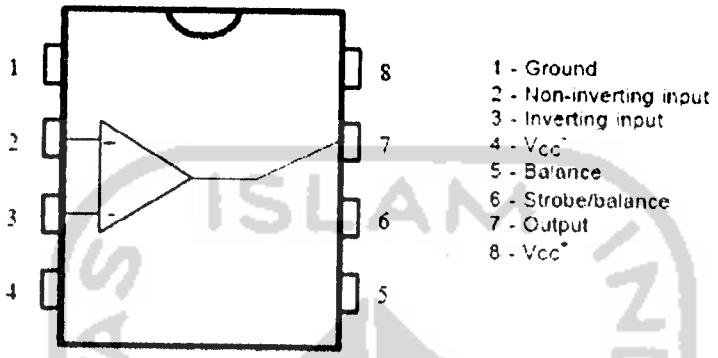
2.3 Op-Amp LM 311

Op-Amp atau *Operational Amplifier* adalah sebuah IC yang umumnya digunakan untuk penguat signal yang nilai penguatannya dapat dikontrol melalui sebuah resistor dan komponen lainnya. Umumnya, Op-Amp terdiri dari dua input dan 1 output. Keluaran dari penguat memiliki persamaan dengan rumus:

$$V_{out} = A(V_{noninverting} - V_{inverting}) \tag{2.13}$$

Dengan A sebagai nilai Penguatannya (*Gain*). $V_{NONINVERTING}$ dan $V_{INVERTING}$ adalah nilai masukan *inverting* dan *noninverting*.

Konfigurasi pin OP-AMP 311 dapat dilihat pada Gambar 2.4. IC ini memiliki 8 pin dengan sebuah OP-AMP di dalamnya. Masukan positif ada pada pin 2, masukan negatif ada pada pin 3, dan keluaran ada pada pin 7.



Gambar 2.4. OP-AMP 311

IC OP-AMP LM311 dapat dikategorikan sebagai pembanding (*comparator*) tegangan yang memiliki arus masukan yang kecil. IC ini juga didesain untuk dapat dioperasikan pada daerah tegangan yang cukup lebar, dari tegangan standar operasional amplifier ±12V, hingga +5V, yang digunakan untuk IC *logic*. Pada LM311, masukan maupun keluaran dapat diisolasikan dari *ground* sistemnya, dan keluarannya dapat mendorong beban yang diacukan ke *ground*, suplai positif atau negatifnya. Meskipun lebih lambat dari LM306 dan LM701, LM311 lebih tahan terhadap kondisi osilasi. LM311 memiliki arus masukan maksimum 250nA, tetapi IC ini dapat mendorong lampu atau relai, dengan mengalikan tegangan sampai 40V pada arus yang setinggi 50mA.

2.4 ASCII

ASCII (*American Standard Code for Information Interchange*) adalah *character encoding* yang berdasarkan *aplhabet* dalam bahasa Inggris. Kode ASCII digunakan untuk menampilkan tulisan dalam dalam komputer, perangkat elektronika, komunikasi, dan alat-alat lain yang bekerja dengan text.

ASCII dibuat pertama kali pada tahun 1963, dan dipublikasikan sebagai standar pada tahun 1967 dan diperbaiki pada tahun 1986. ASCII, sekarang memiliki 128 kharakter, dan memiliki 33 kharakter yang tidak dapat dicetak, dan memiliki 95 kharakter yang dapat dicetak. Sebanyak 95 kharakter yang dapat dicetak tersebut adalah nomor 32 hingga 126, Lihat Tabel 2.1.

Tabel 2.1. Tabel ASCII

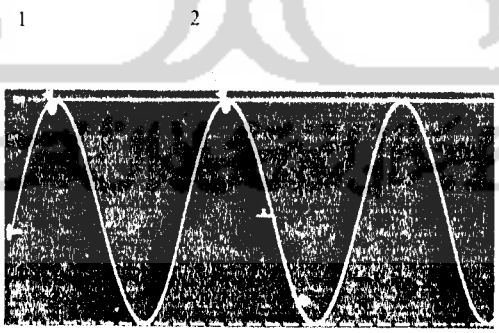
Binary	Dec	Hex	Glyph	Binary	Dec	Hex	Glyph	Binary	Dec	Hex	Glyph
0010 0000	32	20	SP	0100 0000	64	40	@	0110 0000	96	60	`
0010 0001	33	21	!	0100 0001	65	41	A	0110 0001	97	61	a
0010 0010	34	22	"	0100 0010	66	42	B	0110 0010	98	62	b
0010 0011	35	23	#	0100 0011	67	43	C	0110 0011	99	63	c
0010 0100	36	24	$	0100 0100	68	44	D	0110 0100	100	64	d
0010 0101	37	25	%	0100 0101	69	45	E	0110 0101	101	65	e
0010 0110	38	26	&	0100 0110	70	46	F	0110 0110	102	66	f
0010 0111	39	27	'	0100 0111	71	47	G	0110 0111	103	67	g
0010 1000	40	28	(	0100 1000	72	48	H	0110 1000	104	68	h

0010 1001	41	29	)	0100 1001	73	49	I	0110 1001	105	69	i
0010 1010	42	2A	*	0100 1010	74	4A	J	0110 1010	106	6A	j
0010 1011	43	2B	+	0100 1011	75	4B	K	0110 1011	107	6B	k
0010 1100	44	2C	,	0100 1100	76	4C	L	0110 1100	108	6C	l
0010 1101	45	2D	-	0100 1101	77	4D	M	0110 1101	109	6D	m
0010 1110	46	2E	.	0100 1110	78	4E	N	0110 1110	110	6E	n
0010 1111	47	2F	/	0100 1111	79	4F	O	0110 1111	111	6F	o
0011 0000	48	30	0	0101 0000	80	50	P	0111 0000	112	70	p
0011 0001	49	31	1	0101 0001	81	51	Q	0111 0001	113	71	q
0011 0010	50	32	2	0101 0010	82	52	R	0111 0010	114	72	r
0011 0011	51	33	3	0101 0011	83	53	S	0111 0011	115	73	s
0011 0100	52	34	4	0101 0100	84	54	T	0111 0100	116	74	t
0011 0101	53	35	5	0101 0101	85	55	U	0111 0101	117	75	u
0011 0110	54	36	6	0101 0110	86	56	V	0111 0110	118	76	v
0011 0111	55	37	7	0101 0111	87	57	W	0111 0111	119	77	w
0011 1000	56	38	8	0101 1000	88	58	X	0111 1000	120	78	x
0011 1001	57	39	9	0101 1001	89	59	Y	0111 1001	121	79	y
0011 1010	58	3A	:	0101 1010	90	5A	Z	0111 1010	122	7A	z
0011 1011	59	3B	;	0101 1011	91	5B	[	0111 1011	123	7B	{
0011 1100	60	3C	<	0101 1100	92	5C	\	0111 1100	124	7C	|
0011	61	3D	=	0101	93	5D	]	0111	125	7D	}

1101				1101				1101				
0011	62	3E	≥	0101	94	5E	^	0111	126	7E	~	
1110				1110				1110				
0011	63	3F	?	0101	95	5F	_					
1111				1111								

2.5 Osilator dan Gelombang

Sebuah osilator akan secara kontinyu menghasilkan sebuah signal listrik yang nilainya bervariasi terhadap waktu secara berulang-ulang. Karakteristik terpenting yang dimiliki oleh sebuah osilator adalah bentuk gelombang, amplitudo serta frekuensi dari signal yang dibangkitkan. Gelombang memiliki beberapa bentuk yang dikenal secara umum, yaitu gelombang kotak, sinus, dan segitiga (gigi gergaji). Gelombang tersebut memiki dua variabel utama, yaitu amplitudo dan panjang gelombang. Frekuensi pada gelombang tersebut memiliki nilai satu per panjang gelombang. Contoh gelombang sinus dapat dilihat pada Gambar 2.5.



Gambar 2.5. Gelombang Sinus

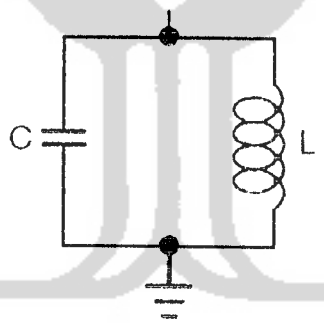
Pada gambar tersebut dapat diketahui sebagai sebuah gelombang sinus yang stabil nilai amplitudo dan frekuensinya. Untuk mengetahui panjang

gelombang dari gelombang sinus tersebut, dapat dihitung waktu dari titik 1 ke titik 2. Lamanya waktu gelombang, dari titik 1 ke titik 2 itulah yang disebut sebagai panjang gelombang. Dengan mengetahui nilai panjang gelombang tersebut, maka dapat diketahui pula nilai frekuensinya dengan rumus :

$$Frekuensi\,(HZ) = \frac{1}{Panjang\ Gelombang\,(\text{detik})} \qquad (2.14)$$

2.6 Osilator LC

Salah satu cara untuk menghasilkan osilasi adalah dengan menggunakan osilator LC, yaitu osilator yang dibentuk dengan menggunakan komponen utama berupa komponen induktif (L) dan komponen kapasitif (C).

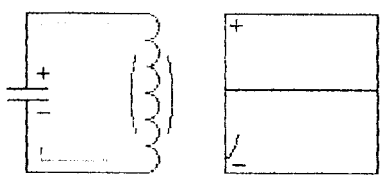


Gambar 2.6. L dan C yang dipararel, sebagai sumber osilasi

Osilator LC terdiri dari kapasitor dan inductor yang koneksikan dengan konfigurasi paralel. Prinsip kerja dari osilator LC yang digunakan untuk menghasilkan gelombang sinus adalah sebagai berikut:

1. Pertama-tama kapasitor diisi oleh tegangan *battery*,

2. Pada saat kapasitor diisi, satu plat kapasitor akan memiliki kelebihan elektron, dari pada plat yang lain. Proses ini disebut sebagai proses pengisian.
3. Ketika kapasitor dikosongkan dengan melalui salurannya, elektron akan kembali ke plat positif, sehingga akan membuat plat kapasitor menjadi netral. Proses ini disebut sebagai proses pengosongan.
4. Proses pengosongan tersebut akan berbeda ketika pengosongan kapasitor tersebut dilakukan melalui sebuah induktor, bukan melalui kawat langsung atau resistor. Ketika arus dilewatkan pada sebuah koil, maka medan magnet akan bangkit pada sekitar induktor. Medan magnet tersebut akan menghasilkan sebuah tegangan pada induktor yang berkebalikan dengan arak arus elektron.
5. Karena hal tersebut, kapasitor tidak dapat dikosongkan secara sepenuhnya dengan cepat seperti pada pengosongan langsung melalui kabel. Pada induktor yang lebih kecil, proses pengosongan kapasitor akan berjalan lebih cepat.
6. Pada saat kapasitor telah dikosongkan secara sepenuhnya melalui sebuah induktor, medan magnet akan mulai hilang di sekitar induktor. Tegangan induksi yang dihasilkan dari proses penurunan medan magnet akan mengisi kembali kapasitor secara berlawanan dari sebelumnya.
7. Sehingga kapasitor akan memulai untuk melakukan pengosongan melalui induktor lagi, sehingga akan membangkitkan sebuah medan magnet.
8. Proses tersebut di atas akan berlanjut hingga kapasitor telah selesai melakukan pengosongan melalui nilai resistansi yang dilalui.



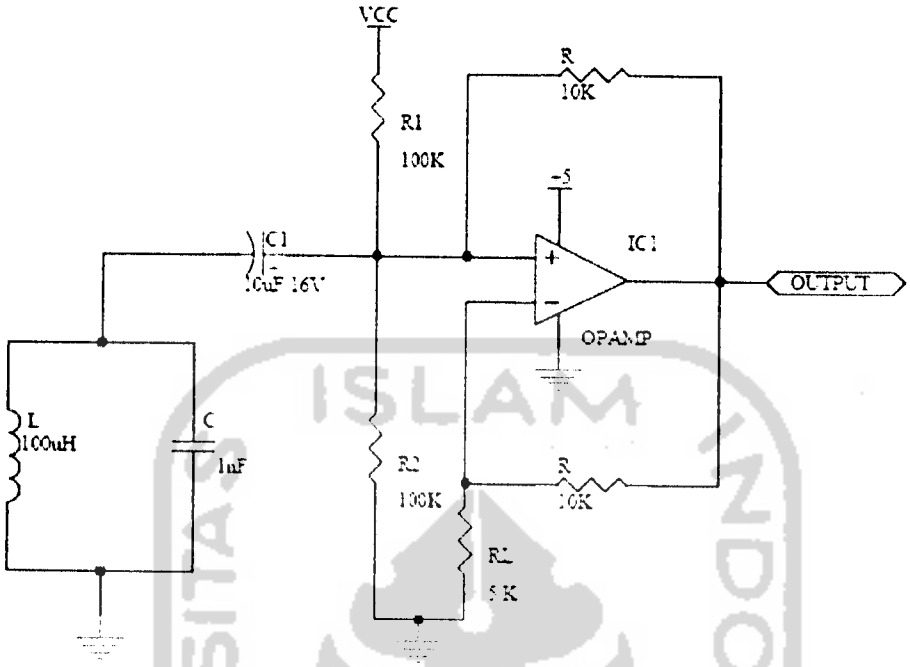
Gambar 2.7. L dan C saat pengisian dan Pengosongan

Secara teknis, dasar rangkaian LC menghasilkan gelombang sinus akan kehilangan tegangan pada setiap siklusnya. Untuk mengatasi hal tersebut, penambahan tegangan diterapkan untuk menjaga osilator dari kehilangan tegangan. Sehingga untuk menjaga osilator berjalan dengan baik, metode *switching* digunakan. Tabung hampa banyak digunakan dalam aplikasi ini, yang dalam perkembangannya digunakan persamaannya, seperti sebuah FET yang digunakan untuk menjaga rangkaian LC tetap berosilasi. Keunggulan dari penggunaan tabung hampa adalah tabung hampa dapat berosilasi pada frekuensi tinggi, hingga ribuan siklus per detiknya.

Op-Amp atau IC yang dirancang khusus dapat digunakan sebagai komponen rangkaian pembentuk rangkaian osilator. Salah satu contoh rangkaian osilator LC dapat dilihat pada Gambar 2.7. Pada gambar tersebut dapat dilihat osilator LC yang bekerja dengan frekuensi f sesuai dengan rumusan sebagai berikut :

$$f = \frac{1}{2\pi\sqrt{LC}} \qquad (2.15)$$

Berdasarkan rumus tersebut dapat diketahui nilai frekuensi f dengan berdasarkan pada nilai L sebagai nilai induktansi induktor dan C sebagai nilai kapasitansi kapasitor. Rangkaian tersebut dikenal sebagai rangkaian osilator LC yang memiliki jangkauan frekuensi yang lebih luas.



Gambar 2.8. Osilator LC yang menggunakan Op-Amp

Kapasitor C1 difungsikan sebagai kapasitor kopling DC yang memisahkan analisis DC antara rangkaian pararel L-C dengan rangkaian pengubah tegangan ke arus.

Resistor R1 dan R2 digunakan sebagai resistor yang membatasi arus dari tegangan VCC, yang digunakan untuk mensuplai tegangan dengan arus kecil pada rangkaian paralel L-C. Tegangan tersebut dibutuhkan untuk proses pengisian C agar osilator LC selalu berosilasi. dan tidak kehilangan tegangan osilasi.

Dengan kombinasi rangkaian Op-Amp dengan resistor R dan RL, maka dapat diperoleh rangkaian pengubah tegangan ke arus, hal ini karena arus beban

IL tergantung pada tegangan masukan. Tegangan masukan dalam Op-Amp ini yang paling dominan tertinggi adalah tegangan masukan 5VDC dari VCC.

Pada rangkaian tersebut, arus beban IL, besarnya tidak tergantung pada tahanan RL. Arus beban mengalir ke mana saja, sehingga rangkaian ini dapat menjadi sumber arus maupu penghisap arus. Arus beban IL ditentukan oleh :

$$I_L = \frac{V_{in}}{R} \qquad (2.16)$$

VIN di sini adalah tegangan VCC, dan R bernilai 10 K Ohm, sehingga dapat diketahui nilai IL berdasarkan persamaan 2.6 di atas.

I_L = 5 V / 10.000 Ohm

I_L = 0,5mA.

Harga IL yang positif menandakan bahwa arus mengalir ke bawah. Harga IL yang negatif artinya VL positif terhadap *ground*. Tegangan beban VL (bukan IL) tergantung pada tahanan beban RL dari

$$V_L = I_L \, x \, R_L \qquad (2.17)$$

Sehingga, dengan RL sebesar 10 K Ohm akan diperoleh

V_L = 0,5mA x 5 K Ohm,

V_L = 2,5 V

Untuk menjamin agar Op-Amp tidak jenuh, maka Vo harus diketahui dan dapat dihitung dari

$$V_O = 2V_L \tag{2.18}$$

Sehingga Vo dapat diperoleh dengan nilai

$V_O = 2 \times 2{,}5\ V$

$V_O = 5\ V$

Kondisi di atas diperhitungkan sebagai kondisi maksimum, sedangkan kondisi tegangan minimum akan terjadi sebagai akibat menurunnya nilai tegangan masukan VIN, karena terpengaruh dengan tegangan minus dari efek osilasi rangkaian paralel L dan C. Namun demikian, nilai RL dapat diubah-ubah untuk menyesuaikan arus masukan yang diperoleh dan hasil Vo maksimum yang mungkin diperoleh.

2.7 Arsitektur Mikrokontroler ATMEL AVR AT90S2313

AVR merupakan sebuah mikrokontroler dengan arsitektur RISC (*Reduce Instruction Set Computer*) 8 *bit* yang dikeluarkan Atmel *Corporation*. Sampai saat ini mikrokontroler keluarga AVR adalah ATTinyxx, ATmegaxxx, dan AT90Sxxxx.

AVR dikembangkan pertama kali oleh para peneliti di Trondheim, Norwegia, sebelum mereka diakuisisi oleh Atmel di tahun 1995. *Feature* inti AVR adalah 32 buah register identik 8-*bit* yang memiliki kemampuan dan sifat seperti sebuah *accumulator.* AVR dikembangkan berdasarkan arsitektur Harvard, yang memisahkan antara memori program dan memori data. Keunggulan arsitektur Harvard adalah kecepatan eksekusi yang tinggi. Susunan instruksinya bertipe RISC, tetapi variasinya cukup banyak. Dengan desain seperti itu, memungkinkan dihasilkan program yang efisien (sependek mungkin) dengan waktu eksekusi yang sangat cepat (kebanyakan dalam satu *clock*) dengan panjang tiap instruksi rata-rata 16 *bit.*

Feature yang dimiliki masing-masing seri keluarga AVR sedikit berbeda. Perbedaan ini menyebabkan beberapa instruksi tidak kompatibel antara satu seri dengan seri yang lain. Selain itu detail arsitekturnya juga sedikit berbeda, perbedaan tersebut di atas bukan sesuatu yang prinsipil tetapi merupakan konsekuensi logis yang timbul dari perbedaan *feature* masing-masing seri. Misalnya pada seri AT90S1200 tidak memiliki *feature* SRAM (*Static Random Access Memory*) otomatis instruksi yang melakukan akses ke SRAM seperti: *pop*, *push*, dll tidak didukung oleh seri AT90S1200. Arsitektur salah satu seri dari keluarga AVR, AT90S2313.

Dalam arsitektur tersebut terlihat bahwa tidak terdapat *accumulator*. Dengan demikian data *input* dan *output* dalam proses ALU disimpan dalam salah satu atau dua *register* dari 32 *register* yang disediakan. Terlihat juga bahwa *bus* data dan *bus* program terpisah secara fisik (tidak secara *logic*), *bus* data memiliki lebar 8

bit, sedangkan *bus* program memiliki lebar 16 *bit*.

Proses eksekusi sebuah perintah adalah sebagai berikut: kode dari *Flash Memory* di-*load* ke *instruksion register* melalui *bus* program dan langsung didekodekan menjadi sejumlah kontrol oleh *instruksion decoder*. Untuk mencapai kecepatan 1 MIPS per MHz AVR menerapkan *pipelining instruction*. *Pipelining* instruksi ini dilakukan dengan cara melakukan pengambilan instruksi selama prosesor mengeksekusi sebuah perintah.

2.8 Konfigurasi Pin AT90S2313

Konfigurasi *Pin* AT90S2313 dapat dilihat pada gambar 2.6.

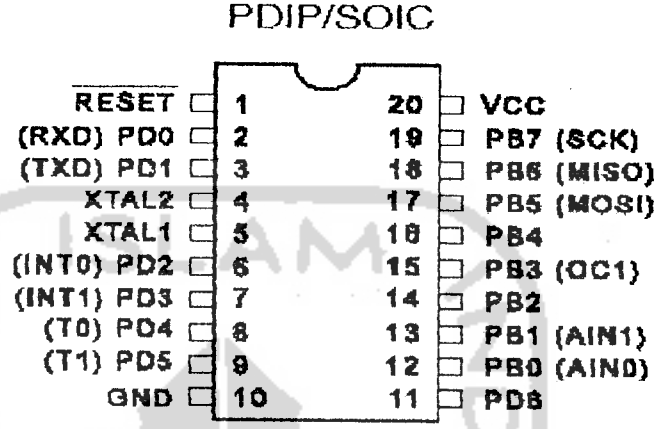
a. Vcc → Digunakan untuk masukan tegangan.

b. Reset → Digunakan untuk reset, pin ini dapat diaktifkan dengan *logic low* selama 50ns. Dengan syarat itu, maka mikrokontroler dapat dalam kondisi *reset*.

c. Gnd → ***Pin Ground.***

d. Port B → Merupakan 1 kelompok kelompok *8-bit bi-directional I O Port*. *Pin* ini pada kondisi *tri-state* ketika terjadi *reset*. Dapat diberi *Pull-Up* secara *internal*. Pada *port* ini juga tersedia fasilitas SCK, MISO dan MOSI yang dapat digunakan untuk keperluan *download*. Selain itu juga tersedia AIN0 dan AIN1 yang dapat digunakan sebagai

input Comparator. Sedangkan OC1 adalah keluaran untuk *Timer Counter*1.



Gambar 2.9. Konfigurasi Pin AT90S2313

c. Port D → Merupakan 1 kelompok *7-bit bi-directional I/O Port*. *Pin* ini pada kondisi *tri-state* ketika terjadi *reset*. Dapat diberi *Pull-Up* secara *internal*. RXD dan TXD dalam *Port* ini disediakan untuk fasilitas komunikasi UART. INT0 dan INT1 disediakan untuk operasi *external interupt*. Sedangkan T0 dapat digunakan untuk *triger* apabila akan digunakan *Timer0* dengan menggunakan *external clock*. Fasilitas lain dalam *Port* ini adalah T1, yang berfungsi seperti halnya T0, tetapi digunakan pada *Timer1*.

f. X-TAL 1 →Merupakan *input* dari *inverting* osilator.

g. X-TAL2 →Merupakan *output* dari *inverting* osilator.

2.9 Instruksi pada Mikrokontroler AT90S2313

Mikrokontroler AT90S2313 memiliki 118 perangkat instruksi. Perangkat instruksi mikrokontroler AVR dapat dibagi sebagai berikut :

a. Instruksi transfer data, instruksi ini berfungsi untuk tranfer data antara *register* ke *register*, memori ke memori, *register* ke memori, antarmuka ke *register* dan antar muka ke memori.

b. Instruksi aritmatika dan ***logic***, instruksi aritmatika meliputi penjumlahan, pengurangan, penambahan satu (*increament*), dan pengurangan satu (*decreament*). Instruksi logika dan manipulasi *bit*, yang melaksanakan operasi AND, OR, XOR, perbandingan, penggeseran dan komplemen data.

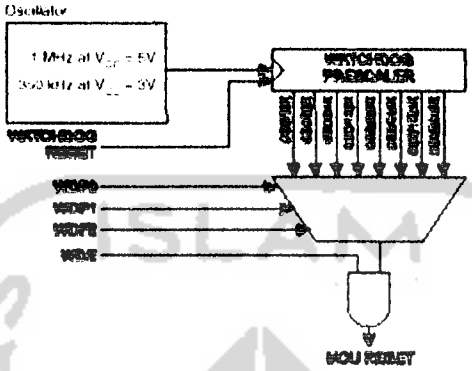
c. Instruksi *Bit* dan *Bit-Test*, yaitu instruksi untuk *setting* kondisi tiap *bit*, baik *set* maupun *clear*, bahkan ada beberapa variasi, seperti instruksi putar, hingga *watchdog reset*.

d. Instruksi percabangan, yang berfungsi mengubah urutan normal pelaksanaan suatu program menjadi sesuai yang dikehendaki. Dengan instruksi ini program yang sedang dilaksanakan akan mencabang ke suatu alamat tertentu. Instruksi percabangan dibedakan atas percabangan bersyarat dan percabangan tanpa syarat.

e. Instruksi *stack*, *I/O* dan kontrol, yang digunakan untuk mengatur penggunaan *stack*, membaca/menulis *port I/O* serta pengontrolan-pengontrolan.

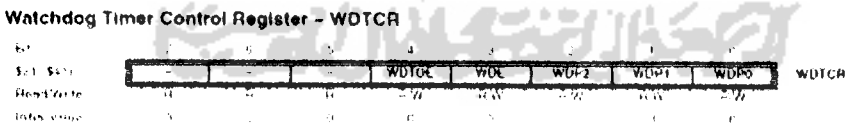
2.10 Watchdog Timer pada Mikrokontroler AT90S2313

Watchdog timer mendapatkan *clock* dari sebuah osilator tersendiri di



Gambar 2.10. Diagram Blok dalam *Watchdog Timer*

dalam *chip* mikrokontroler, pada tegangan 5 Volt, nilai *clock*-nya sebesar 1MHz. Adapun diagram yang ada dalam sistem *watchdog timer* adalah seperti dalam Gambar 2.10. Dengan memberikan nilai tertentu pada register WDTCR, yaitu dengan WDE, bit 3 diberi nilai *logic* 1 untuk mengakifkan *watchdog timer*, selanjutnya diberikan nilai pada WDP0, WDP1, dan WDP2 untuk pengaturan lamanya waktu *time-out*.



Dengan adanya tabel *watchdog time-out* pada Tabel 2.2 akan memudahkan dalam pemilihan nilai WDP, untuk menentukan seberapa lama nilai waktu *watchdog time-out* yang diinginkan. Sebagai contoh, apabila akan diinginkan *watchdog time-out* terjadi selama 15 ms, maka diperlukan nilai *hexa* pada

WDTCR sebesar 0x08. Karena semua nilai WDP bernilai 0, dan diperlukan nilai *logic* 1 pada bit WDE, yang berada pada bit 3.

Tabel 2.2. Tabel *Time-Out* pada *Watchdog Timer*

WDP2	WDP1	WDP0	Number of WDT Oscillator Cycles	Typical Time-out at V_{CC} = 3.0V	Typical Time-out at V_{CC} = 5.0V
0	0	0	16K cycles	47 ms	15 ms
0	0	1	32K cycles	94 ms	30 ms
0	1	0	64K cycles	0.19 s	60 ms
0	1	1	128K cycles	0.38 s	0.12 s
1	0	0	256K cycles	0.75 s	0.24 s
1	0	1	512K cycles	1.5 s	0.49 s
1	1	0	1,024K cycles	3.0 s	0.97 s
1	1	1	2,048K cycles	6.0 s	1.9 s

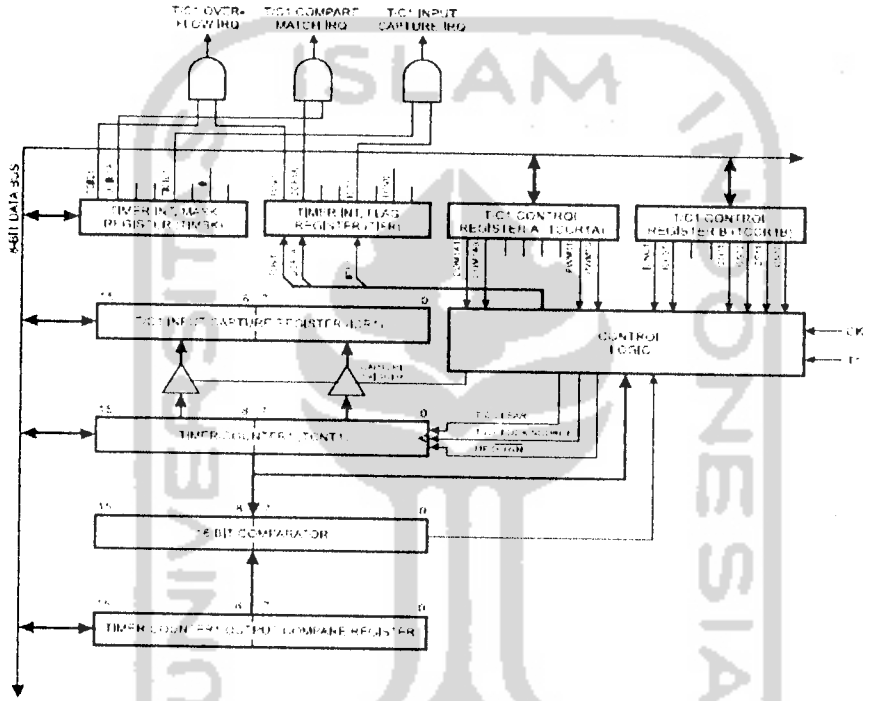
Selanjutnya tidak boleh untuk dilupakan bahwa setelah register WDTCR diberi nilai tersebut di atas, maka *watchdog timer* akan aktif. Pada beberapa tempat yang kemungkinan untuk dilalui program, harus diberikan perintah *reset* atas *watchdog timer*. Sehingga diperlukan *instruction "wdr"* pada beberapa bagian *source code*, dengan maksud menghindari terjadinya *time-out*.

Hal ini mengingat tujuan adanya *watchdog* adalah untuk mengatasi terjadinya *hank*. Karena saat terjadinya *hank*, maka tentu saja *watchdog* akan *time-out*. Sehingga program secara otomatis akan kembali ke awal, tanpa dilakukan *reset*.

2.11 Timer/Counter pada Mikrokontroler AT90S2313

Mikrokontroler AT90S2313 memiliki dua *timer counter*, yaitu *Timer 0* dan *Timer 1*. Timer 1 memiliki konfigurasi 16 bit. Diagram blok *Timer/Counter 1*

dapat dilihat pada Gambar 2.11. Dari diagram tersebut dapat diketahui beberapa blok penting dalam aplikasi *Timer 1*, yaitu *control register, control register, control logic, timer/counter 1(TCNT1), compare register, interupt mask, dan interupt flag*. Dari blok-blok tersebut, jalur datanya dihubungkan dengan *8-bits data bus*.



Gambar 2.11. Blok Diagram Timer/Counter

2.11.1. *Timer/Counter 1 Control Register*

Timer/Counter1 Control Register B – TCCR1B

Bit	7	6	5	4	3	2	1	0	
$2E ($4E)	ICNC1	ICES1	–	–	CTC1	CS12	CS11	CS10	TCCR1B
Read/Write	R/W	R/W	R	R	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

Pada bagian *Control Register*, dapat diatur *noise canceler* pada bit 7. dengan memberi nilai, satu, maka *noise canceler* akan aktif. Dengan nilai nol, maka *noise canceler* akan dalam kondisi tidak aktif.

Pada bit 0 hingga bit 2 digunakan untuk pengaturan CS10, CS11, dan CS12, yaitu pengaturan nilai *clock* yang digunakan untuk *timer1*. Namun demikian, apabila *timer/counter1* digunakan untuk mendeteksi masukan, maka dapat digunakan untuk mode *external pin T1, falling edge*, dengan nilai CS10 = 0, CS11 = 1, dan CS12 = 1.

2.11.2. *Timer/Counter1 TCNT1*

Untuk mendapatkan data dari *timer/counter* tersebut, dapat diperoleh data melalui register TCNT1. Data pada register ini memiliki konfigurasi 16 bit. Data pada register tersebut bersifat *read only*, sehingga data pada register tersebut hanya dapat dibaca saja, tanpa bisa diisi atau diubah secara *software*.

2.11.3. *Timer/Counter Interupt Mask Register*

Apabila akan digunakan *interupt overflow timer1*, maka dapat dilakukan set pada bit 7 pada register ini. Register ini dikenal sebagai register TIMSK, sehingga pada register TIMSK, bila diberi nilai 0x80, maka akan menyebabkan *interupt*, ketika terjadi *overflow*, pada nilai register TNCT1.

2.11.4. *Timer/Counter Interupt FLAG Register*

Register ini secara *sofiware* dikenal sebagai register TIFR. Register ini pada umumnya akan bekerja secara otomatis oleh mikrokontroler. Sebagai contoh, apabila terjadi *overflow* pada TCNT1, maka pada bit 7 secara otomatis akan diset 1 oleh mikrokontroler, yang menandakan *overflow*. Bit 7 tersebut, selanjutnya akan bernilai nol kembali ketika program *interupt timer* dibaca.

2.12 Bahasa C untuk Pemrograman Mikrokontroler

Pada mulanya bahasa computer digunakan untuk membantu dalam melakukan perhitungan telemeri. Ketika itu, bahasa yang digunakan masih primitive sekali karena masih berupa bahasa mesin yang hanya mengenal angka 1 dan 0. Selanjutnya bahasa mesin tersebut disederhanakan menjadi bahasa yang agak dipahami dengan menghadirkan beberapa *statement* khusus yang disebut dengan istilah *mnemonic*, seperti ADD, MOV, JMP, dan yang lainnya. Bahasa ini disebut dengan bahasa *assembly* yang masih termasuk ke dalam bahasa tingkat rendah (*low level language).*

Tahun 1969, laboratorium Bell AT&T di Muray Hill, New Jersey menggunakan bahasa *assembly* ini untuk mengembangkan system operasi UNIX. Setelah UNIX berjalan, Ken Thompson, seorang pengembang system di laboratorium tersebut mengembangkan *compiler* baru dengan nama bahasa B. Bahasa B digunakan untuk merevisi sistem operasi UNIX. Pada tahun 1971, sistem operasi UNIX kemudian ditulis ulang dengan menggunakan bahasa C,

yaitu bahasa pemrograman yang dikembangkan oleh Dennis Ritchie, seorang pengembang sistem di laboratorium yang sama.

Sampai sekarang, bahasa C masih digunakan untuk melakukan pengembangan program dan sistem operasi, diantaranya adalah sistem operasi Windows, dan Linux. Alasan itulah yang menjadikan bahasa C menjadi semakin populer di dunia pemrograman.

Bahasa C adalah bahasa pemrograman yang termasuk dalam *medium level language*, merupakan bahasa yang dekat dengan bahasa mesin. Dalam desain arsitektur AVR dipertimbangkan penggunaan bahasa C untuk AVR, sehingga diharapkan instruksi dengan bahasa C akan efisien untuk AVR.

Agar dapat menggunakan bahasa C untuk pemrograman AVR, maka diperlukan C *Compiler* untuk keperluan ini. Dalam penelitian ini digunakan *ATMEL AVR C Compiler*. Dalam *C Compiler* yang digunakan memiliki beberapa fasilitas untuk pemrograman bahasa C dan perlu diperhatikan beberapa hal sebagai berikut :

a. Fungsi

Pada dasarnya Program C tersusun atas sejumlah blok fungsi. Dalam fungsi memiliki beberapa bagian, yaitu nama fungsi, tubuh fungsi yang diawali dengan tanda '{' dan diakhiri dengan tanda '}'. Dalam bahasa C terdapat fungsi yang paling istimewa dan harus ada dalam setiap Program C, yaitu *Main Function*.

b. Preprosesor

Preprosesor dapat digunakan untuk mendefinisikan makro, untuk memanggil *header file*, dan untuk *setting* kondisi ketika kompilasi.

c. Variabel

Dalam penggunaan variabel biasanya perlu diperhatikan dalam deklarasi, inisialisasi, pemilihan tipe, dan kadang dipertimbangkan juga masalah penyimpanan variabel tersebut.

d. Instruksi Pengambilan Keputusan

Bahasa C memiliki beberapa istruksi pengambilan keputusan, diantaranya adalah pernyataan *if, if-else,* dan *switch.* Dalam penggunaannya kadang diperlukan beberapa operator logika, seperti *AND, OR,* atau *NOT.*

e. Pengulangan Proses

Sedangkan untuk keperluan pengulangan proses, dalam bahasa C disediakan beberapa instruksi yang dapat digunakan seperti pernyataan *do-while, while, for, break,* dan *goto.*

2.13 LCD

LCD yang digunakan dalam penelitian ini adalah LCD Hitachi 1 x 16, yang berarti memiliki 2 baris, 8 kolom. *Pin* yang digunakan dalam LCD ini adalah :

a. Data *bus*, memiliki 8 *pin*, tetapi hanya digunakan 4 *pin* saja, yaitu DB7, DB6, DB5, dan DB4.

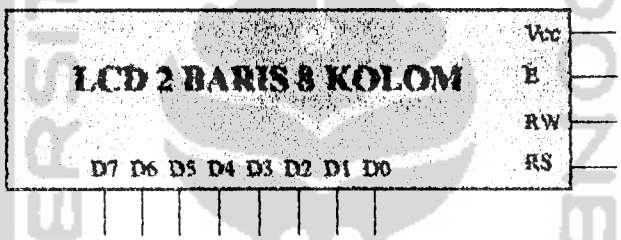
b. RS, yang apabila *logic* tinggi, menandakan adanya data *input*, apabila *logic* rendah menandakan adanya *input* instruksi.

c. R/W, yang merupakan *sign* untuk operasi baca-tulis, apabila dalam kondisi *high* berarti dalam kondisi *read*, apabila dalam kondisi *low* berarti dalam kondisi *Write*.

d. E, yang memberikan *sign enable signal*.

e. VCC, untuk *input* tegangan 5 Volt

f. Vss, untuk *input* tegangan *Ground*.



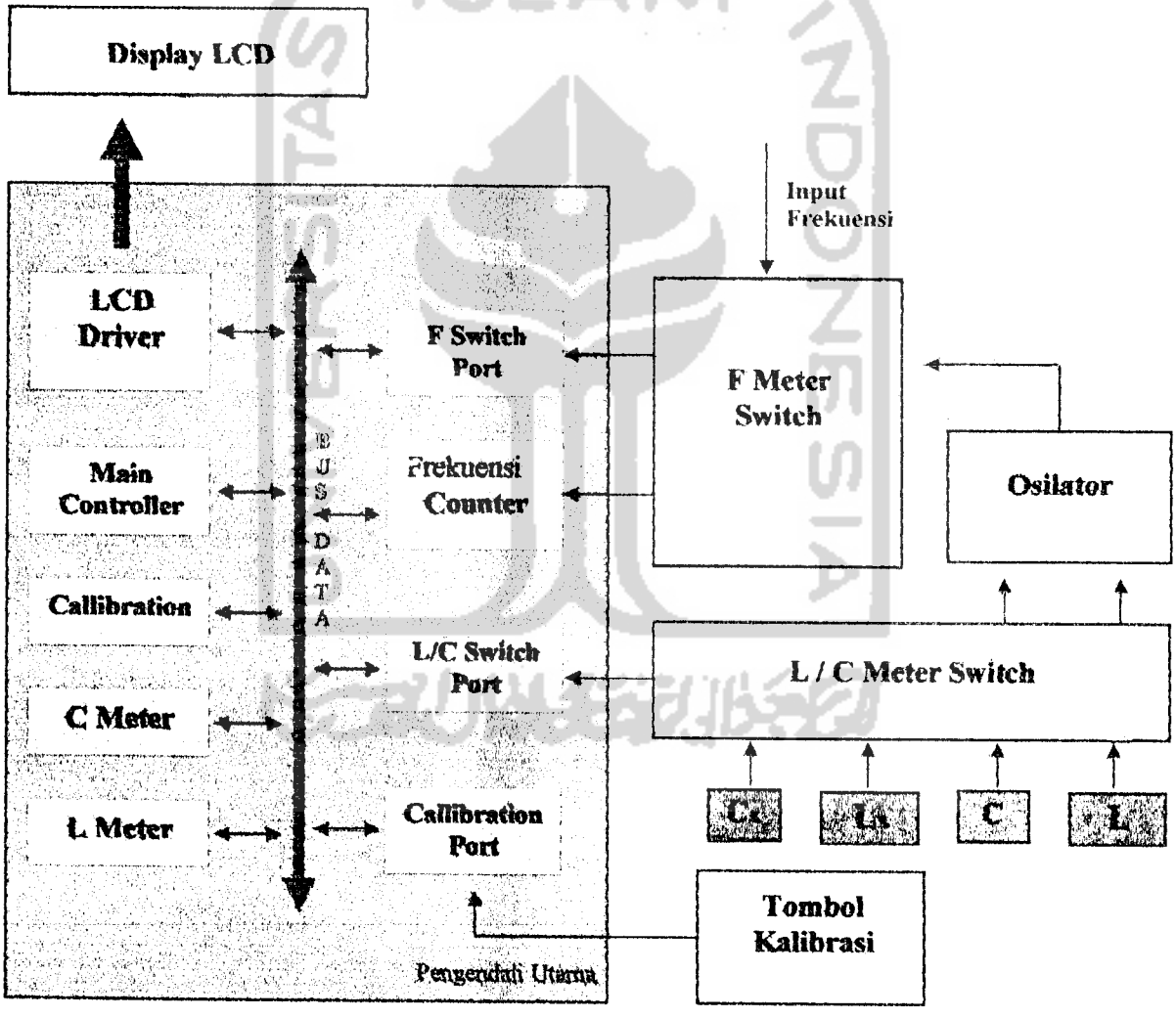
Gambar 2.12. LCD Hitachi 1 x 16

BAB III

PERANCANGAN SISTEM

3.1. Blok Diagram Sistem

Dari Gambar 3.1. dapat dilihat blok diagram sistem FLC Meter Digital berbasis mikrokontroler AT90S2313. Sistem tersebut memiliki beberapa bagian



Gambar 3.1. Diagram Blok *F L C Meter*

blok yang dibentuk dengan komponen elektronika dan *software*. Blok diagram di atas memiliki beberapa bagian blok, yaitu Blok Pengendali Utama (terdiri dari blok F Switch Port, L/C Switch Port, Frekuensi Counter, Callibration Port, C Meter, L Meter, Callibration, Main Controller dan LCD Driver), Tombol kalibrasi, L/C Meter Switch, Osilator, dan F Meter Switch. Pada blok pengendali utama, tiap bagian blok sistem dibentuk secara *software*, sehingga sesuai dengan rancangan blok sistem di atas.

3.1.1. L, C, Cx dan Lx

L dalam blok ini adalah induktor yang diposisikan untuk memberikan nilai induktansi yang sangat berpengaruh terhadap nilai frekuensi dari keluaran osilator. L berfungsi untuk keperluan kalibrasi dan penghitungan kapasitor.

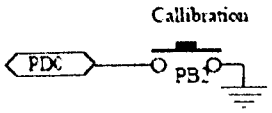
C dalam blok ini adalah kapasitor yang bersama dengan L, nilainya akan memberikan pengaruh pada nilai frekuensi keluaran osilator. C di sini sangat penting untuk keperluan kalibrasi dan penghitungan induktor.

Cx dalam blok ini adalah kapasitor yang akan diukur nilai kapasitansinya. Sedangkan Lx dalam blok ini adalah induktor yang akan diukur nilai induktansinya.

3.1.1.1. Tombol Kalibrasi

Tombol Kalibrasi digunakan untuk antarmuka dari pengguna ketika akan melakukan proses kalibrasi. Tombol kalibrasi ini dapat ditekan sebelum L/C meter

digunakan. Saat tombol ini ditekan, L/C Switch harus dalam posisi C, yang menunjuk pada C meter.

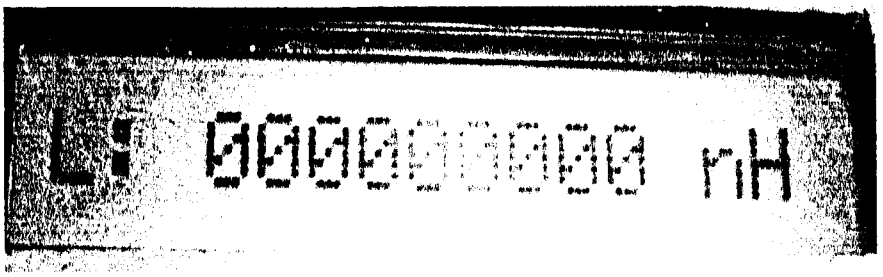


Gambar 3.2. Tombol Kalibrasi

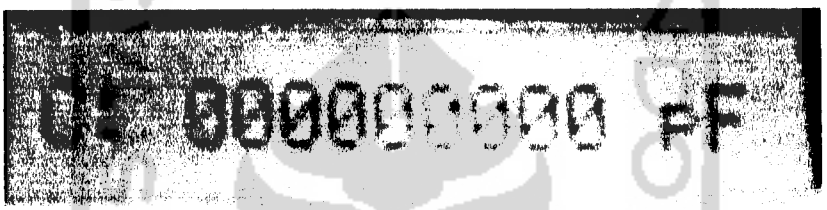
Tombol ini diperlukan mengingat, dalam pengukuran L/C memiliki keunikan tersendiri, yaitu pada kenyataannya, nilai frekuensi osilasi dapat berubah pada suhu yang berbeda. Sehingga, sebelum mulai dilakukan pengukuran, sebaiknya dilakukan proses kalibrasi, yaitu dengan cukup memposisikan pengukuran pada C Meter, kemudian ditekan tombol kalibrasi. Secara *hardware*, tombol ini cukup berupa *push button switch* yang dihubungkan menuju pin PD0 pada mikrokontroler AT90S2313.

3.1.1.2. L/C Switch

L/C Switch digunakan sebagai selektor untuk pengukuran L dan C. Pada pengukuran L, L/C Switch diposisikan pada posisi L, yaitu pada posisi *switch* keluar. Pada tampilan LCD akan keluar tampilan awal ketika tak ada komponen induktif terpasang seperti pada gambar 3.3.



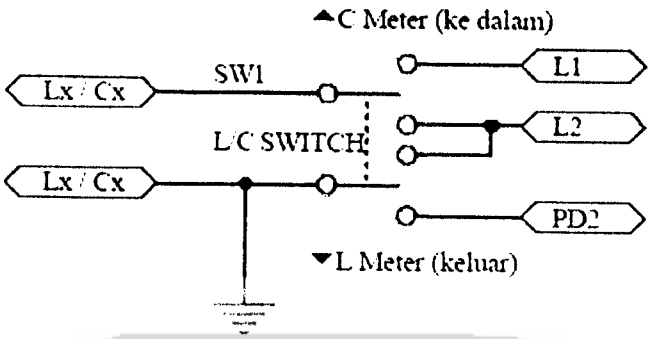
Gambar 3.3. Tampilan awal saat penghitungan L (induktansi)



Gambar 3.4. Tampilan awal saat penghitungan C (kapasitansi).

Pada pengukuran C, L/C Switch diposisikan pada posisi C, yaitu pada posisi *switch* ke dalam. Pada tampilan LCD akan keluar tampilan seperti pada gambar 3.4, tampilan seperti di gambar tersebut terjadi saat tak ada komponen kapasitif yang terpasang.

L/C *switch* secara *hardware* berupa *switch* DPDT yang memiliki 6 pin. Untuk mengetahui posisi *switch* saat ini, maka mikrokontroler mendeteksi posisi tombol ini melalui pin PD2.



Gambar 3.5. L/C Switch

Pada saat *switch* tersebut ditekan, maka pin PD2 akan bernilai 1, nilai ini akan disimpan dalam register yang dimiliki blok L/C Switch Port. Posisi tersebut dikenali oleh pengendali utama sebagai posisi C Meter.

3.1.1.3. Osilator LC

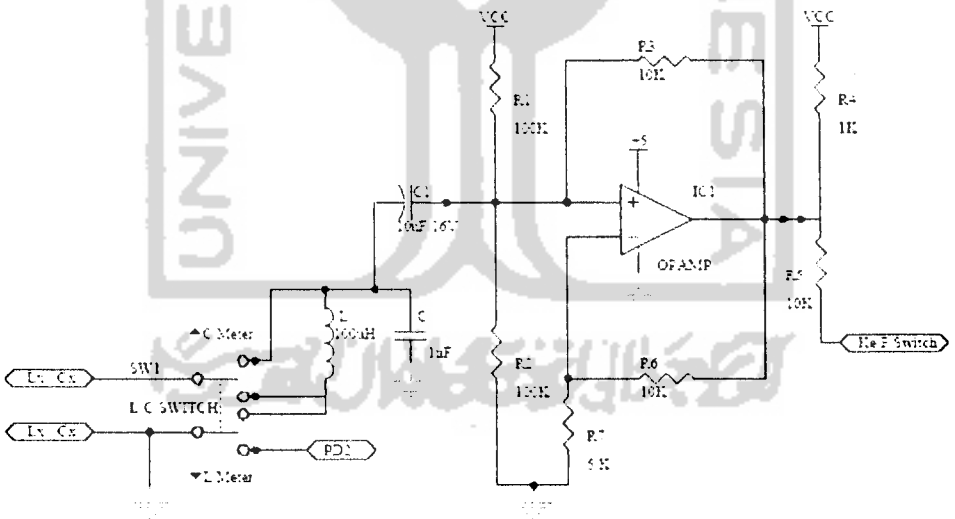
Blok osilator ini memiliki komponen utama berupa IC Op-Amp LM311 yang didukung dengan sumber osilasi yang utama yaitu rangkaian pararel L dan C. Osilator yang digunakan dalam penelitian ini, sebenarnya adalah osilator LC. Osilator ini dipilih, mengingat komponen yang akan diukur adalah L (induktor) dan C (kapasitor).

Dengan mengetahui nilai L, maka akan diketahui nilai C dengan menghitung nilai frekuensinya. Dengan mengetahui nilai C, maka akan diketahui nilai L dengan menghitung nilai frekuensinya. Frekuensi osilasi yang dapat dihasilkan dari rangkaian tersebut adalah sesuai dengan rumusan di bawah ini:

$$f = \frac{1}{2\pi\sqrt{L * C}}$$

Hal ini sesuai dengan rumus 2.7 pada BAB II, nilai frekuensi osilator dipengaruhi atas nilai L dan C.

Dalam rangkaian ini ditambahkan resistor R4, yang akan memberikan *pull-up* atas tegangan keluaran dari Op-Amp. Hal ini diperlukan untuk setiap rangkaian op-amp dengan signal AC yang akan dihubungkan dengan tegangan kerja standar TTL, maka diperlukan resistor *pull-up* sebesar 1 K Ohm.



Gambar 3.6. Rangkaian Osilator LC

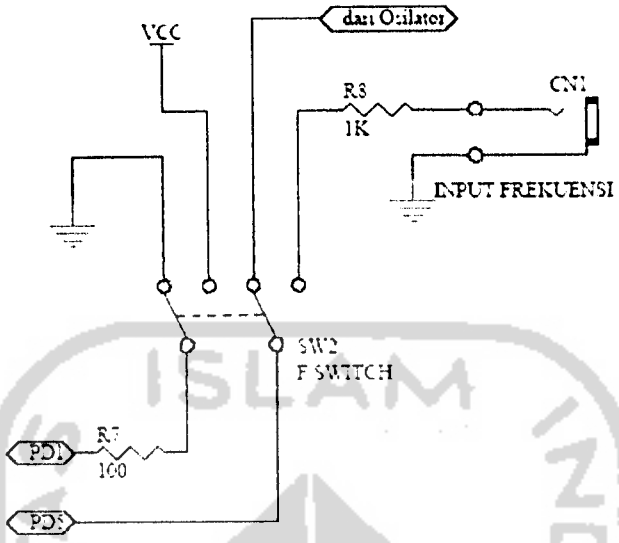
Sedangkan R5 digunakan untuk pengaman arus yang menuju ke mikrokontroler. Nilai dari resistor ini dapat disesuaikan dengan kondisi arus masukan yang menuju ke mikroknroler. Apabila arus yang masuk terlalu tinggi, maka semakin tinggi pula nilai resistor R5 yang diperlukan.

3.1.2. F Meter Switch

F Meter Switch digunakan sebagai saklar selektor untuk memilih apakah akan mengukur frekuensi (F) atau akan mengukur L/C (induktor/kapasitor).

Gambar 3.7. Tampilan awal saat penghitungan F (Frekuensi)

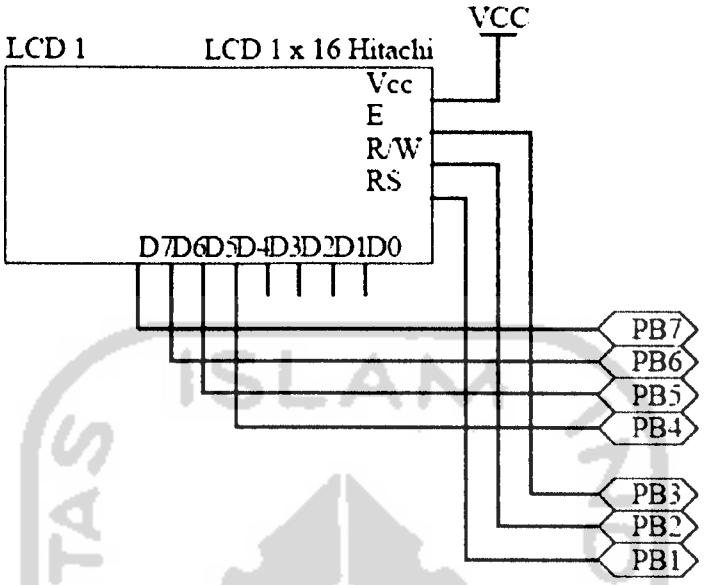
Apabila akan dilakukan pengukuran frekuensi, maka F Meter Switch harus ditekan, sehingga akan tertampil tampilan awal seperti pada gambar 3.7. Secara fisik, blok ini berupa *switch* DPDT, yang posisinya dideteksi oleh pengendali utama melalui blok F Switch Port, yang secara fisik berada pada pin PD1.



Gambar 3.8. F Meter Switch

3.1.3. LCD Display

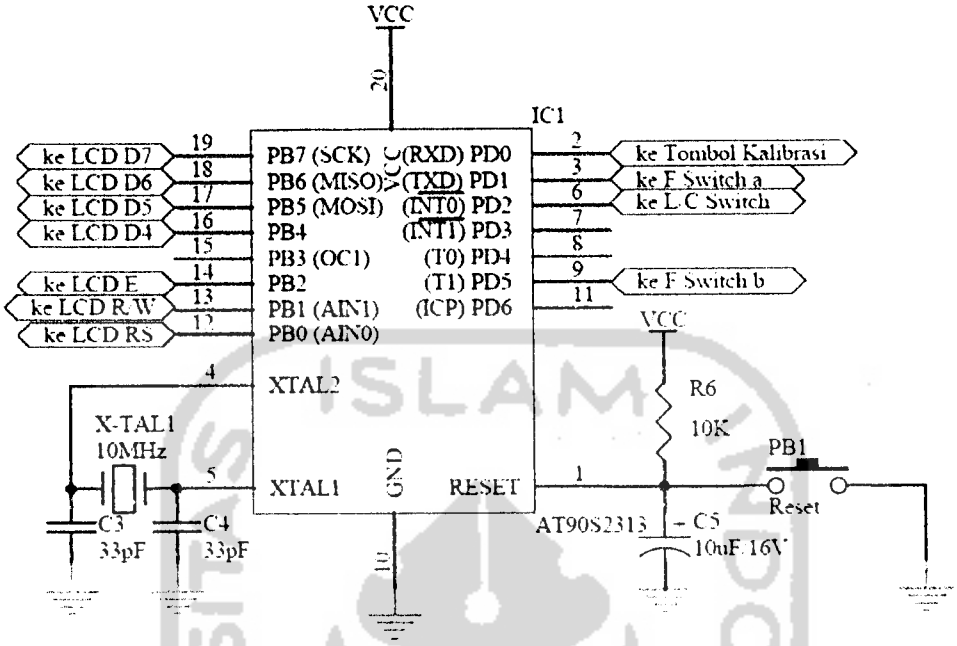
LCD Display diperlukan untuk menampilkan semua informasi berkaitan dengan proses dan hasil pengukuran nilai L (induktansi), C(kapasitansi), atau F(frekuensi). Pengguna juga dapat mengetahui posisi yang sedang diukur dengan melihat tampilan LCD. LCD yang digunakan dalam perancangan ini adalah LCD Hitachi 1x16, satu baris, 16 kolom.



Gambar 3.9. LCD Hitachi 1 x 16.

3.1.4. Pengendali Utama

Pada pengendali utama memiliki beberapa blok penting yang kesemuanya dirancang dengan menggunakan *software*. Pengendali utama memiliki komponen *hardware* yang utama yaitu mikrokontroler Atmel, AT90S2313. Mikrokontroler ini memiliki 20 pin dan menggunakan X-TAL 10MHz. Nilai X-TAL ini adalah sudah sesuai dengan nilai maksimum yang direkomendasikan oleh Atmel, yang tertulis dalam *data sheet*.



Gambar 3.10. Rangkaian Pengendali Utama

Beberapa bagian blok yang diformulasikan dalam bentuk software dalam pengendali utama adalah sebagai berikut:

3.1.4.1. F Switch Port

F Switch Port adalah Port yang digunakan untuk mendeteksi posisi *F* Meter Switch. Saat saklar *F* Meter Switch ditekan, maka port ini akan bernilai *logic* 1, saat saklar dilepas, maka port ini akan bernilai *logic* 0. Secara *hardware*, dalam mikrokontroler AT90S2313, port ini berada pada posisi port PD1.

3.1.4.2. L/C Switch Port

L C Switch Port adalah port yang digunakan oleh pengendali utama untuk mendeteksi posisi *L C* Switch. Dengan pendeteksian pada port ini, maka akan dapat diketahui apakah posisi *switch* pada posisi masuk atau pada posisi keluar. Secara *hardware*, posisi port ini ada pada port PD2 pada mikrokontroler, sehingga merupakan bagian dari PORTD.

3.1.4.3. Callibration Port

Callibration Port adalah port yang digunakan oleh pengendali utama untuk mendeteksi tombol. Ketika tombol kalibrasi ditekan, maka port ini akan mendeteksi penekanan tombol. yang datanya siap untuk diproses secara *software*. Secara *hardware*, posisi port ini ada pada port PD0 pada mikrokontroler, sehingga merupakan bagian dari PORTD.

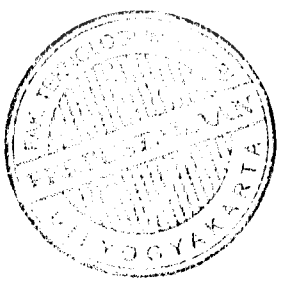
3.1.4.4. Frekuensi Counter

Untuk menghitung frekuensi, digunakan *source code* sebagai berikut

```
void FrequencyCounter(void){

TCCR1B=0x86; // start Timer1

//Mulai Counter...
timer0_start();
```



```
    while(1){
  if(i == 40)break;
  #asm("wdr")
  }

// Counter Ending
tTCNT1 = TCNT1;
 TCCR1B = 0x00; //stop Timer1
 TCCR0 = 0x00; //stop timer

// Overflow Processing
  tResult = ov ;
  tResult = tResult * 65536;

  FResult = tTCNT1;
  FResult = FResult + tResult;

}
```

Dari program di atas, untuk melakukan pengukuran frekuensi, maka dapat dilakukan pengukuran dengan menghitung banyaknya gelombang yang masuk ke mikrokontroler, selama 1 detik.

Untuk melakukan proses tersebut, maka sistem harus mempersiapkan Timer0 untuk menghitung lamanya waktu, 1 detik dan Timer1 yang digunakan untuk menghitung banyaknya pulsa yang masuk.

Dari *source code* di atas dapat diketahui proses awal dengan menghidupkan *timer1* dengan *code*

```
TCCR1B=0x86; // start Timer1
```

Yang segera dilanjutkan dengan menghidupkan *timer0* untuk menghitung waktu pendeteksian selama 1 detik

```
timer0_start();
```

Selanjutnya, *timer1* akan menghitung dan menyimpan data frekuensi, dan sistem akan menunggu saat berakhirnya 1 detik tersebut. Selama satu detik tersebut, program yang dikerjakan adalah

```
while(1){
if(i == 40)break;
#asm("wdr")
}
```

Karena *timing* yang digunakan dalam *timer0* adalah 25ms, maka diperlukan pengulangan waktu selama 40 kali untuk menghasilkan 1000 ms atau 1 detik.

Setelah satu detik, maka pembacaan kode program akan keluar dari proses *looping* di atas, yang selanjutnya akan dilakukan penyimpanan data hasil penghitungan frekuensi oleh *timer1*

```
tTCNT1 = TCNT1;
```

Selanjutnya, *timer0* dan *timer1* harus dihentikan aktifitasnya dengan pemberian kode

```
TCCR1B = 0x00; //stop Timer1
TCCR0 = 0x00; //stop timer
```

Untuk menghadapi *overflow* atas nilai yang dihasilkan dari penghitungan nilai frekuensi oleh *timer1*, maka dilakukan penghitungan berapa kali *overflow*, yang hsilnya dikalkulasi menjadi nilai frekuensi total

```
tResult = ov ;
tResult = tResult * 65536;

FResult = tTCNT1;
FResult = FResult + tResult;
```

Sehingga hasil akhir dari penghitungan frekuensi disimpan ke dalam variabel `FResult`.

3.1.4.5. LCD Driver

Untuk menampilkan nilai frekuensi hasil dari *F* Meter, nilai kapasitansi hasil dari *C* Meter, dan nilai induktansi hasil dari *L* Meter, maka diperlukan program LCD Driver yang memiliki *source code* sebagai berikut.

```
void LCDDriver(void){
lcd_gotoxy(2,0);
lcd_putsf(" ");
Ascii(mr);

//lcd_gotoxy(1,1);
Ascii(mp);

//lcd_gotoxy(2,1);
Ascii(ms);

//lcd_gotoxy(3,1);
Ascii(nr);

//lcd_gotoxy(4,1);
Ascii(np);

lcd_gotoxy(0,1);
//lcd_gotoxy(5,1);
```

```
Ascii(ns);

Ascii(pr);

//lcd_gotoxy(7,1);
Ascii(pp);

//lcd_gotoxy(8,1);
Ascii(ps);

  //lcd_gotoxy(6,1);
  if (Freq == 0)  lcd_putsf(" Hz ");
  else{
     if (PIND.2 == 1) lcd_putsf(" pF ");
     else lcd_putsf(" nH ");
  }

}
```

Sesuai dengan *source code* tersebut, untuk menampilkan nilai frekuensi, maka pada *display* ditampilkan satuan frekuensi dalam Hz. Untuk penampilan kapasitansi, maka nilai yang dihasilkan ditampilkan dengan satuan pF. Sedangkan untuk penampilan nillai induktansi, maka nilai yang dihasilkan ditampilkan dalam satuan nH.

Data yang menjadi masukan dalam proses ini adalah telah menjadi data dengan nilai digitalnya. Function LCDDriver di atas selanjutnya akan mengubah nilai desimal tersebut menjadi data *ASCII* yang nilainya langsung dikirimkan ke LCD 1x16 menjadi data tampilan sesuai dengan yang direncanakan.

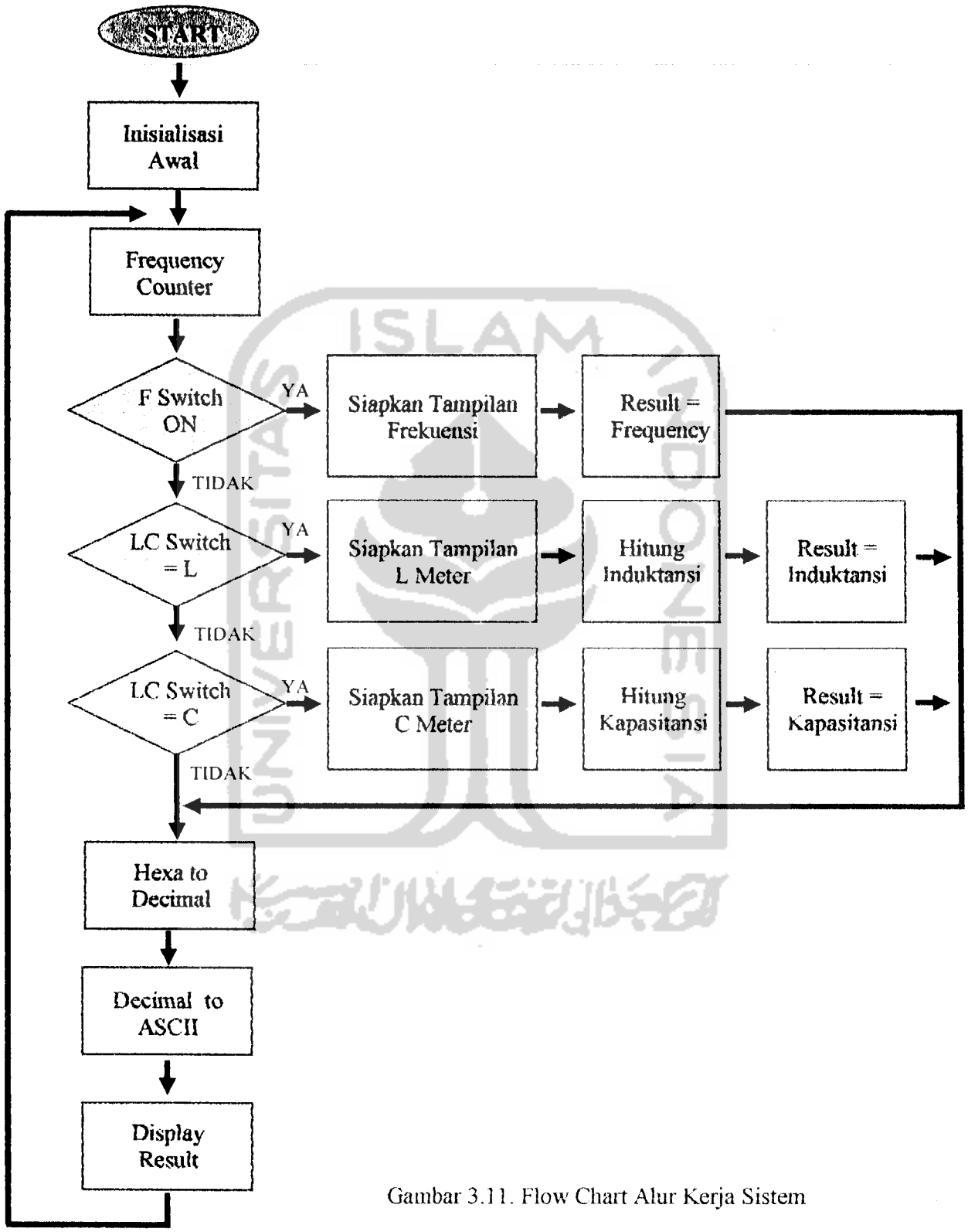
3.1.4.6. Main Controller

Main Controller dirancang dalam bentuk *software* yang memiliki alur kerja sesuai dengan *flow chart* cara kerja sistem pada gambar 3.11. Software pada *main controller* dirancang untuk mengendalikan sistem yang mengacu pada sistem pengendalian seperti pada gambar 3.11.

Dari *flow chart* tersebut dapat diketahui alur kerja sesuai dengan prosedur sebagai berikut :

1. Start, program memulai pembacaan dari alamat 000, yang berisi alamat RESET. Alamat ini akan dibaca ketika pertama kali sistem dihidupkan atau ketika ditekan tombol reset.
2. Selanjutnya, program akan melakukan inisialisasi awal, yaitu pemberian nilai awal atas variabel yang akan digunakan dan *setting* atas bagian sistem yang akan digunakan.
3. Yang dilakukan pertama kali sistem setelah inisialisasi awal adalah proses penghitungan frekuensi. Dengan demikian, sistem telah segera melakukan proses utamanya. Proses penghitungan frekuensi ini dilakukan selama 1 detik.

4. Setelah nilai frekuensi yang masuk diketahui, maka sistem akan membaca nilai tombol *F* Switch. Jika tombol *F* Switch ON, maka sistem dalam kondisi sebagai frekuensi meter. Dengan berfungsinya sistem sebagai frekuensi meter, maka dapat dilakukan persiapan tampilan frekuensi meter. Nilai variabel *Result* diisi dengan nilai frekuensi yang telah diperoleh. Nilai variabel *result* tersebut nantinya akan digunakan untuk menampilkan data ke LCD.
5. Bila F Switch dalam kondisi OFF, maka dilakukan pengecekan atas *LC* Switch. Apabila posisi *LC* Switch ada pada posisi *L*, maka tampilan akan dipersiapkan sebagai tampilan bagi *L* Meter. Proses yang dijalankan selanjutnya adalah proses penghitungan nilai induktansi. Nilai variabel *Result* diisi dengan nilai hasil penghitungan nilai induktansi yang telah diperoleh. Dengan melalui nilai variabel *result* tersebut, nantinya nilai induktansi akan dapat segera ditampilkan ke LCD.
6. Tetapi, apabila posisi *LC* Switch ada pada posisi *C*, maka tampilan akan dipersiapkan sebagai tampilan bagi *C* Meter. Proses yang dijalankan selanjutnya adalah proses penghitungan nilai kapasitansi. Nilai variabel *Result* diisi dengan nilai hasil penghitungan nilai kapasitansi yang telah diperoleh. Dengan melalui nilai variabel *result* tersebut, nantinya nilai kapasitansi akan dapat segera ditampilkan ke LCD.



Gambar 3.11. Flow Chart Alur Kerja Sistem

7. Proses yang dilakukan berikutnya adalah dengan mengubah data hasil penghitungan yang telah diperoleh, yang berupa data hexa, menjadi data desimal. Data hexa dibutuhkan untuk memudahkan perhitungan dalam bahasa mesin, sedangkan data desimal sangat dibutuhkan ketika pembacaan dilakukan oleh orang awam, pada umumnya.
8. Data desimal tersebut diharapkan dapat tertampil melalui tampilan LCD dalam sesuai dengan nilai desimalnya, maka data harus dikirim ke LCD dalam bentuk data ASCII. LCD yang digunakan menggunakan standar data ASCII. Sehingga proses yang dilakukan dalam tahap ini adalah proses penghubahan data desimal ke data ASCII.
9. Setelah data *result* yang diperoleh menjadi data ASCII, maka data tersebut dapat dikirim menuju LCD, sehingga akan diperoleh nilai yang diinginkan.
10. Setelah data *result* ditampilkan ke LCD, maka segera dilakukan kembali proses di atas, yang dimulai dari nomor 3, yaitu dilakukan kembali proses penghitungan frekuensi. Sehingga proses nomor 3 hingga nomor 10 ini dilakukan berulang-ulang, hal ini diperlukan untuk keperluan perbaruan data secara terus menerus.

Secara *software*, alur kerja di atas dapat diformulasikan menjadi kode pmrograman, yang dalam bahasa *C*, diposisikan sebagai *Main Function*, atau fungsi utama. Fungsi utama tersebutlah yang akan menjadi pengendali utama atas

kode-kode *software* yang lain, dengan kata lain menjadi pengendali atas *fucntion-function* yang lain.

Adapun kode program dari *Main Fuction* tersebut adalah sebagai berikut:

```
void main(void)
{

// Port D init
PORTD=0xFF;

// Timer(s)/Counter(s) initialization
TIMSK = 0x82;

#asm("wdr")
WDTCR = 0x0F;

// LCD module initialization
lcd_init(16);

// Global enable interrupts
#asm("sei")
 //lcd_gotoxy(0,0);
 //lcd_putsf("DIGITL FLC METER");
 // delay_ms(2000);
  /*
 lcd_gotoxy(0,0);
 lcd_putsf("By: SUPRIADI UII");
 delay_ms(2000);
```

```
*/
 //lcd_clear();
#asm("wdr")


while (1)
      {

ov = 0;
TCNT1  = 0;
i = 0;

FrequencyCounter();
Callibration();

////////////////////////////////////////////
// Cek apakah L atau C yang sedang dihitung
   lcd_gotoxy(0,0);
   if(Freq == 0) {
     tResult = FResult;
     lcd_putsf("F:");
    }
   else{
  if (PIND.2 == 0){
    lcd_putsf("L:");
   if(FResult <= 130) tResult = 0;
     else{

  LMeter();

     }
```

Program pada *main function* yang dimiliki dalam sistem ini memiliki beberapa proses penting, yaitu inisialisasi awal, *frequency counter*, keputusan penghitungan *f*, *L*, dan *C* yang kesemuanya bekerja sesuai dengan alur pada *flow chart*.

Program diawali dengan pemberian nilai awal pada PORTD, yang berfungsi sebagai input. Selanjutnya diberikan nilai *setting* nilai TIMSK dan WDTCR. Setelah kesemua *setting* awal dilakukan, maka dilakukan *looping* yang membuat program akan selalu mendeteksi, dari *frequency counter* hingga *display* secara berulang-ulang.

Untuk memanggil fungsi penghitungan frekuensi dilakukan dengan kode program

```
FrequencyCounter();
```

Sedangkan untuk proses kalibrasi, dapat dikerjakan dengan memanggil fungsi kalibrasi,

```
Callibration();
```

Selanjutnya, program mulai mendeteksi posisi *F switch* yang jika nilainya sesuai, maka proses yang dilakukan adalah proses penghitungan frekuensi.

```
if(Freq == 0) {
    tResult = FResult;
```

```
        lcd_putsf("F:");
    }
```

Pendeteksian LC *Switch* dilakukan dengan kode berikut

```
if (PIND.2 == 0){
  lcd_putsf("L:");
 if(FResult <= 130) tResult = 0;
   else{

LMeter();

   }

}
 else{
      CMeter();

 }
```

PIND2 menjadi pin yang digunakan untuk pendeteksian tombol *LC switch*. Sesuai dengan program di atas, apabila nilai PIND2 adalah 0, maka sistem berfungi sebagai *L* Meter, sehingga dapat dipangil *function* untuk penghitungan induktansi,

```
LMeter();
```

Apabila nilai PIND2 adalah 1, maka sistem akan berfungsi sebagai *C* Meter, sehingga dapat dipanggil fungsi penghitungan kapasitansi dengan

```
CMeter();
```

Setelah nilai pengukuran diperoleh, maka dapat dikonversikan ke nilai desimal dengan kode

```
Hexa2Decimal();
```

Function tersebut akan mengubah nilai hexa ke nilai desimalnya, yang selanjutnya, nilainya siap untuk menuju proses penampilan ke LCD dengan bentuk ASCII, dengan memanggil *function* Display LCD.

```
// Display Process
LCDDriver();
```

3.1.4.7. Callibration

Proses kalibrasi, secara *software* dilakukan untuk menghitung sebagian dari rumus utama frekuensi osilator *LC*. Kode program yang digunakan adalah sebagai berikut:

```
void Callibration(void){

//  Constanta / Callibration
 if (PIND.0 == 0){
```

```
            Constant = FResult / 10;
            Constant = Constant * Constant;
        }

    }
```

Pada program tersebut dideteksi tombol kalibrasi dengan pendeteksian PIND0 pada PORTD, yaitu dengan kode

```
if (PIND.0 == 0)
```

apabila PIND0 ditekan, maka nilainya akan sesuai dengan nilai *logic* 0. Apabila nilainya telah sesuai dengan nilai tersebut, maka dapat dihitung nilai konstanta

```
Constant = FResult / 10;
```

Fresult adalah nilai frekuensi yang telah dihitung pada saat komponen L atau C belum dikoneksikan dari luar, sedangkan Constant adalah nilai konstantanya. Sebelum nilai konstanta diperoleh dari nilai frekuensi, maka nilai frekuensi dibagi 10, hal ini untuk mengantisipasi kelebihan nilai saat penghitungan L atau C. Dengan demikian nilai konstanta yang dihitung nilainya tidak terlalu besar. Dalam penghitungan di mikrokontroler yang masih 8 bit ini harus dihindari penghitungan yang menggunakan nilai pecahan, negatif, atau nilai yang membutuhkan bit

tinggi, seperti 32 bit, karena akan menurunkan kecepatan proses dan membutuhkan memori yang lebih besar.

Setelah diperoleh nilai konstanta awal, maka akan dapat dihitung hasil akhir nilai konstanta

```
Constant = Constant * Constant;
```

Dengan demikian nilai konstanta akan sesuai dengan nilai kuadrat dari nilai frekuensi saat komponen L dan C belum menggunakan komponen luar yang akan dihitung.

3.1.4.8. C Meter

Berdasarkan rumus frekuensi 2.7 pada BAB II, apabila akan dicari nilai kapasitor, maka dapat diperhitungkan dengan rumus:

$$f^2 = \frac{1}{4 \times 3{,}142857 \times L \times C} \quad (3.1)$$

$$f^2 = \frac{1}{39{,}51 \times LC} \quad (3.2)$$

$$f^2 = 0.0253 \frac{1}{LC} \quad (3.3)$$

$$f^2 = \frac{0.0253}{LC} \tag{3.4}$$

$$LC = \frac{0.0253}{f^2} \tag{3.5}$$

$$C = \frac{0.0253}{L} \times \frac{1}{f^2} \tag{3.6}$$

Apabila telah diketahui nilai induktansi L, dan frekuensi osilator f, maka akan dapat diketahui nilai C. Maka dengan menggunakan rumus 3.6, maka dapat dirumuskan nilai kapasitansi saat tak ada kapasitor yang dikoneksikan, C, dengan rumusan sebagai berikut :

$$C = \frac{0.0253}{L} \times \frac{1}{f_0^2} \tag{3.7}$$

dengan f_0 sebagai nilai frekuensi saat digunakan kapasitor internal C, dan L sebagai induktor internal. Dengan kondisi yang lain, dapat dirumuskan sebuah rumusan berikut:

$$Cx + C = \frac{0.0253}{L} \times \frac{1}{f_1^2} \tag{3.8}$$

Cx sebagai nilai kapasitansi kapasitor eksternal yang terpasang untuk diukur, f_1 sebagai nilai frekuensi osilator saat digunakan kedua kapasitor tersebut. Apabila dimisalkan nilai konstanta K_1 memiliki persamaan berikut :

$$K_1 = \frac{0.0253}{L} \tag{3.9}$$

Maka dengan menggabungkan rumus 3.7 dengan rumus 3.9, akan diperoleh persamaan baru sebagai berikut:

$$C = K_1 \times \frac{1}{f_1^2} \tag{3.10}$$

Sedangkan rumus 3.8 dengan rumus 3.9, bila digabungkan akan menjadi sebuah rumus baru 3.11 berikut.

$$Cx + C = K_1 \times \frac{1}{f_1^2} \tag{3.11}$$

Dari rumus 3.10 dapat diperoleh nilai K_1 menjadi

$$K_1 = C \times f_0^2 \tag{3.12}$$

Sehingga rumus 3.11 dengan 3.12 dapat digabungkan menjadi rumus 3.13 dengan metode substitusi.

$$Cx+C = C\,x\,F_o^{\,2}\,x\,\frac{1}{f_1^{\,2}} \qquad (3.14)$$

Yang dapat disederhanakan menjadi

$$C_p = \frac{C\,x\,f_o^{\,2}}{f_1^{\,2}} \qquad (3.15)$$

dengan Cp sebagai nilai kapasitansi pararel C dengan Cx. f_0 dapat juga disebut sebagai nilai frekuensi osilator saat dilakukan kalibrasi. Untuk mengaplikasikan rumus 3.15, dapat dilakukan dengan *sofiware* sebagai berikut :

```
void CMeter(void){
        lcd_gotoxy(0,0);
      lcd_putsf("C:");

         tResult = Constant / FResult;
       tResult = tResult * 100000;
     tResult = tResult / FResult;
    if(tResult < 1000) tResult = 0;
    else tResult = tResult - 1000;

}
```

Dari *source code* di atas, dapat diketahui proses implementasi atas rumus 3.15 di atas. Variabel `Constant` adalah nilai konstanta yang diperoleh dengan rumus

$$Constant = f_0^2 \quad (3.16).$$

Proses yang dilakukan dalam penghitungan ini dilakukan secara bertahap, hal ini mengingat apabila rumus di atas diterapkan secara langsung, maka mikrokontroler tidak mampu untuk melakukan penghitungan keseluruhan, dan dapat menyebabkan kesalahan nilai. Selain itu juga harus dihindari terjadinya *overflow*.

Pada awalnya, variabel `tResult` diisi dengan nilai *Constant* yang dibagi dengan nilai frekuensi osilator, maka hasil nilai akan lebih kecil dari nilai sebelumnya.

```
tResult = Constant / FResult;
```

Selanjutnya nilai tersebut harus ditingkatkan dahulu, yaitu dengan mengalikan terlebih dahulu dengan nilai kapasitansi 100000. Sehingga nilai baru akan meningkat, dan siap untuk dibagi dengan nilai frekuensi osilator `Fresult`.

```
tResult = tResult * 100000;
tResult = tResult / FResult;
```

Karena nilai kapasitansi internal nilainya 1000pF, maka nilai kapasitansi kapasitor yang sedang dihitung Cx dapat diperoleh dengan kode

```
tResult = tResult - 1000;
```

sehingga hasil akhir nilai kapasitansi kapasitor yang sedang dihitung akan tersimpan dalam variabel `tResult`.

3.1.4.9. L Meter

Dengan melanjutkan perhitungan rumus 3.5 di atas, apabila akan dicari nilai induktor yang belum diketahui, maka dapat dicari dengan rumus 3.5 yang dimodifikasi menjadi

$$L = \frac{0.0253}{C} x \frac{1}{f^2} \qquad (3.17)$$

Apabila telah diketahui nilai induktansi C, dan frekuensi osilator F, maka akan dapat diketahui nilai L. Maka dengan menggunakan rumus 3.17, maka dapat dirumuskan nilai induktansi saat tak ada induktor atau kapasitor yang dikoneksikan secara eksternal, dengan rumusan sebagai berikut :

$$L = \frac{0.0253}{C} x \frac{1}{F_o^2} \qquad (3.18)$$

dengan f_0 sebagai nilai frekuensi saat digunakan kapasitor internal C, dan L sebagai induktor internal. Dengan kondisi yang lain, dapat dirumuskan sebuah rumusan berikut:

$$L_x + L = \frac{0.0253}{C} x \frac{1}{f_1^2} \qquad (3.19)$$

Lx sebagai nilai induktansi induktor eksternal yang terpasang untuk diukur,f_1 sebagai nilai frekuensi osilator saat digunakan induktor tersebut. Apabila dimisalkan nilai konstanta K_2 memiliki peersamaan berikut :

$$K_2 = \frac{0.0253}{C} \qquad (3.20)$$

Maka dengan menggabungkan rumus 3.18 dengan rumus 3.20, akan diperoleh persamaan baru sebagai berikut:

$$L = K_2 \times \frac{1}{f_0^2} \qquad (3.21)$$

Sedangkan rumus 3.19 dengan rumus 3.20, bila digabungkan akan menjadi sebuah rumus baru 3.11 berikut.

$$L_x + L = K_2 x \frac{1}{f_1^2} \qquad (3.22)$$

Dari rumus 3.2.1 dapat diperoleh nilai K_2 menjadi

$$K_2 = L \times F_0^{\,2} \tag{3.23}$$

Sehingga rumus 3.22 dengan 3.23 dapat digabungkan menjadi rumus 3.24 dengan metode substitusi.

$$L_x + L = L \times f_0^{\,2} x \frac{1}{f_1^{\,2}} \tag{3.24}$$

Yang dapat disederhanakan menjadi

$$L_p = \frac{L_x \times f_0^{\,2}}{f_1^{\,2}} \tag{3.25}$$

dengan *Lp* sebagai nilai induktsi serial *L* dengan *Lx*. f_0 dapat juga disebut sebagai nilai frekuensi osilator saat dilakukan kalibrasi. Untuk mengaplikasikan rumus 3.15, dapat dilakukan dengan *software* sebagai berikut :

```
void LMeter(void){

        tResult = Constant / FResult;
        tResult = tResult * 68000;
      tResult = tResult / FResult;
    if(tResult < 68000) tResult = 0;
    else tResult = tResult - 68000;
}
```

Dari *source code* di atas, dapat diketahui proses implementasi atas rumus 3.15 di atas. Variabel `Constant` adalah nilai konstanta yang diperoleh dengan rumus

$$Constant = f_0^{\,2} \qquad (3.26).$$

Proses yang dilakukan dalam penghitungan ini dilakukan secara bertahap, hal ini mengingat apabila rumus di atas diterapkan secara langsung, maka mikrokontroler tidak mampu untuk melakukan penghitungan keseluruhan, dan dapat menyebabkan kesalahan nilai. Selain itu juga harus dihindari terjadinya *overflow*.

Pada awalnya, variabel `tResult` diisi dengan nilai *Constant* yang dibagi dengan nilai frekuensi osilator. maka hasil nilai akan lebih kecil dari nilai sebelumnya.

```
tResult = Constant / FResult;
```

Selanjutnya nilai tersebut harus ditingkatkan dahulu, yaitu dengan mengalikan terlebih dahulu dengan nilai induktansi 68000 nH. Sehingga nilai baru akan meningkat, dan siap untuk dibagi dengan nilai frekuensi osilator `Fresult`.

```
tResult = tResult * 68000;
tResult = tResult / FResult;
```

Karena nilai induktansi internal nilainya 68000nH, maka nilai induktansi induktor yang sedang dihitung *Lx* dapat diperoleh dengan kode

```
tResult = tResult - 68000;
```

sehingga hasil akhir nilai kapasitansi kapasitor yang sedang dihitung akan tersimpan dalam variabel `tResult`.

3.1.4.10. Power Supply

Power Supply yang digunakan dalam sistem ini adalah Power Supply yang menggunakan koneksi dari adaptor atau *battery*, yang diseleksi dengan *switch* SW3. Sehingga masukan tegangan dapat dipilih, apakah akan menggunakan *battery* 9 VDC atau akan menggunakan Adaptor 6 – 12VDC. Tegangan masukan tersebut akan diregulasi dengan komponen utama berupa IC LM7805. Dengan IC tersebut, akan dapat diperoleh tegangan keluaran 5VDC.

BAB IV

PENGUJIAN ALAT

4.1. Pengujian Kapasitansi Meter

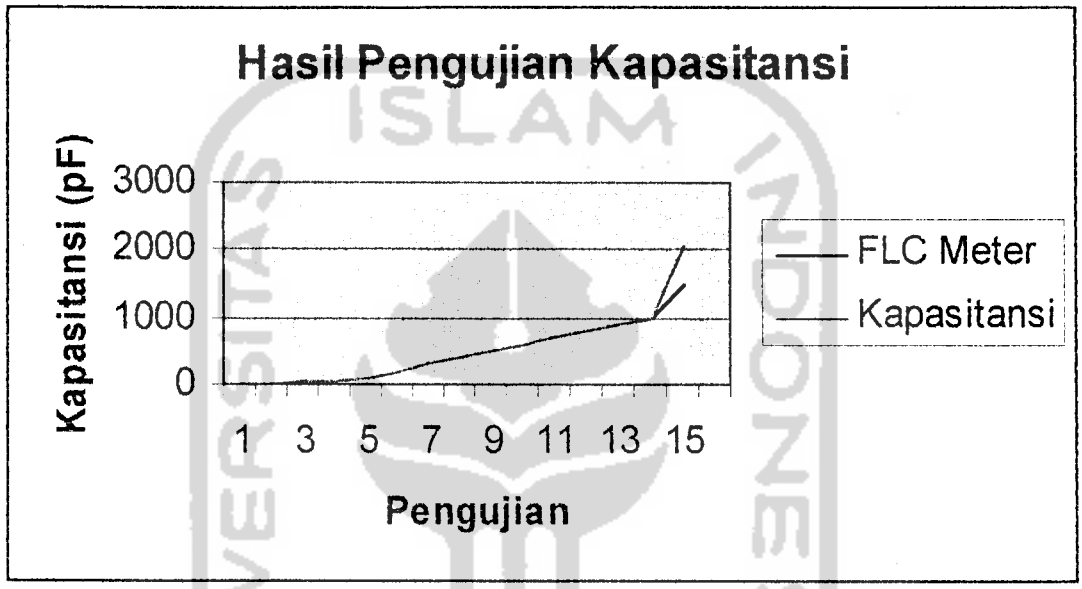
Berdasarkan hasil pengujian yang telah dilakukan, diperoleh hasil pengujian kemampuan pengukuran kapasitansi pada fungsi C Meter pada FLC Meter digital berbasis mikrokontroler AT90S2313.

Tabel 4.1. Tabel Hasil Pengujian Kapasitansi

NO	Nilai yg Tertulis Kapasitor	Hasil Pengukuran	*Error* Hasil terhadap Nilai	% *Error*
1	10 pF	10 pF	0	0,00%
2	20 pF	20 pF	0	0,00%
3	33 pF	33 pF	0	0,00%
4	66 pF	67 pF	-1	1,52%
5	100 pF	98 pF	2	2,00%
6	200 pF	200 pF	0	0,00%
7	300 pF	305 pF	-5	1,67%
8	400 pF	400 pF	0	0,00%
9	500 pF	501 pF	-1	0,20%
10	600 pF	599 pF	1	0,17%
11	700 pF	702 pF	-2	0,29%
12	800 pF	801 pF	-1	0,13%
13	900 pF	900 pF	0	0,00%
14	1000 pF	995 pF	5	0,50%
15	1500 pF	1502 pF	-2	0,13%
			error rata-rata	0,44%

Dari hasil pengujian di atas, dapat diketahui bahwa FLC Meter mampu mengukur kapasitor dari 0 pF hingga 1500pF dengan *error* rata-rata sebesar

0,44%. Untuk pengukuran di atas nilai tersebut, FLC Meter mengalami kendala meningkatnya nilai *error*. Sebagai contoh, pada pengukuran kapasitor 2000 pF, diperoleh nilai hasil pengukuran 2043, sehingga nilai *error*-nya 2,15%.



Gambar 4.1. Grafik Hasil Pengujian Kapasitansi

Dari grafik gambar 4.1 dapat dilihat begitu identiknya antara hasil pengukuran dengan nilai sebenarnya hingga nilai 1000pF atau 1nF. Hal ini terjadi karena *error* yang dimiliki masih sangat kecil, yaitu di bawah 1%.

4.2. Pengujian Induktansi Meter

Hasil pengujian kemampuan pengukuran induktansi pada fungsi L Meter pada FLC Meter digital berbasis mikrokontroler AT90S2313 dapat diketahui berdasarkan atas hasil pengujian yang telah dilakukan sesuai dengan tabel 4.2.

Tabel 4.2. Tabel Hasil Pengujian Pengukuran Induktasi

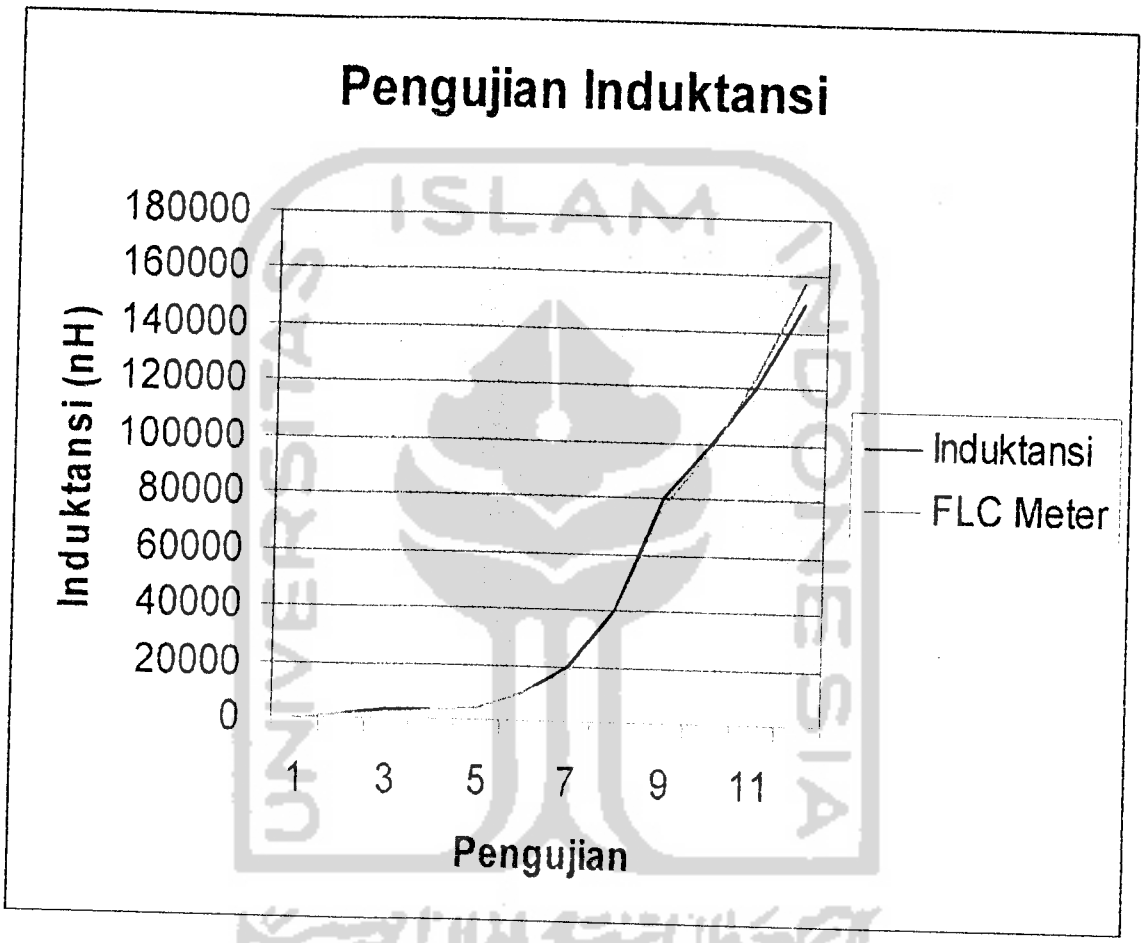
L (nH)	LX (nH)	LS (nH)	LX HASIL PEMBACAAN	Error	% Error
68000	0	68000	0	0	0,00%
68000	2200	70200	2.000	200	9,09%
68000	3300	71300	3.000	300	9,09%
68000	3900	71900	3.500	400	10,26%
68000	4700	72700	4.300	400	8,51%
68000	10000	78000	9.800	200	2,00%
68000	20000	88000	18.000	2.000	10,00%
68000	40000	108000	39.000	1.000	2,50%
68000	80000	148000	77.000	3.000	3,75%
68000	100000	168000	98.000	2.000	2,00%
68000	120000	188000	124.300	-4.300	3,58%
68000	150000	218000	158.000	-8.000	5,33%
				Error rata-rata:	5,51%

Dari tabel 4.2 di atas, dapat diketahui bahwa pengukuran induktansi yang telah dilakukan dengan menggunakan FLC Meter Digital berbasis Mikrokontroller memiliki nilai *error* rata-rata 5,51%. Induktansi yang menjadi perhitungan sebagai induktor pada Osilator LC adalah induktor internal L dan induktor eksternal LX yang dihubung seri, sehingga memiliki nilai induktansi seri LS. Yang tertampil dalam pembacaan di LCD adalah nilai induktansi LX yang merupakan induktansi dari induktor yang ingin untuk diukur nilainya.

Pada pengukuran induktansi tersebut, penelitian ini memiliki kendala sulitnya mencari sampel induktor yang bervariasi atau memiliki nilai sama dengan jumlah banyak. Harga induktor juga relatif lebih mahal dari kapasitor.

Berdasarkan grafik pada gambar 4.2, dapat diketahui hasil pengujian induktansi yang memiliki *error* rata-rata 5,51%. Dalam grafik tersebut jelas

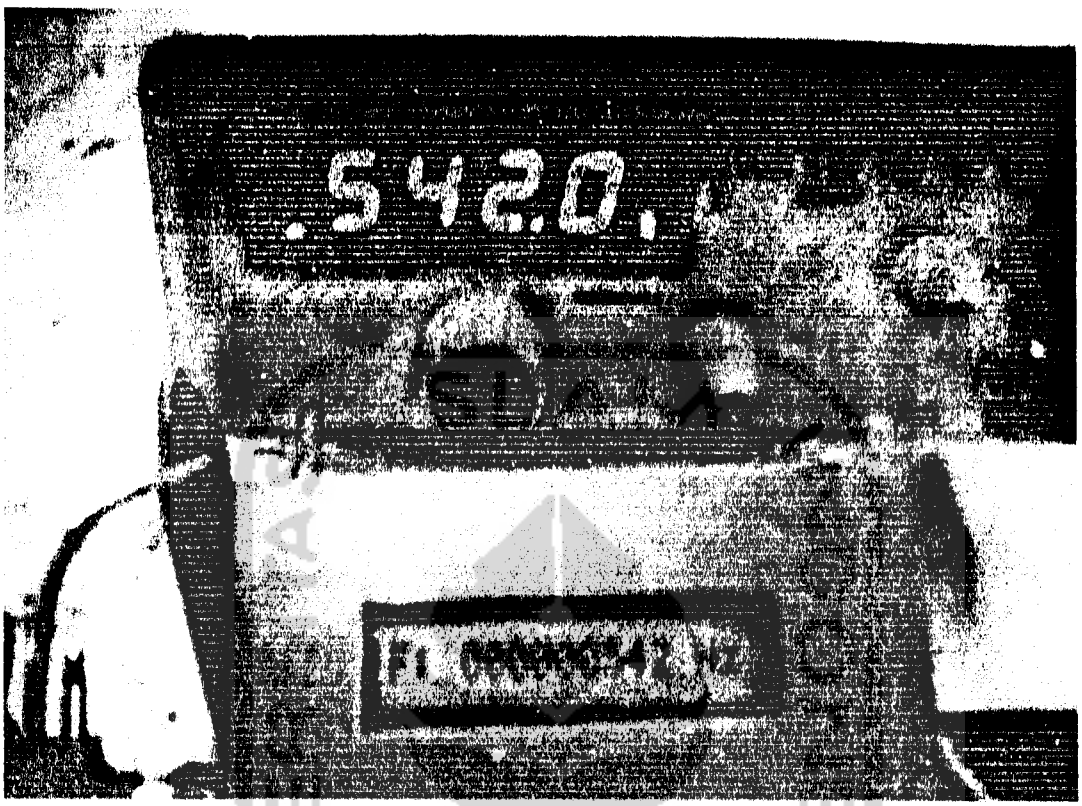
terlihat perbedaan antara grafik nilai induktansi aslinya dengan hasil pengukuran. Namun demikian, bentuk kedua grafik tersebut masih cenderung memiliki kemiripan.



Gambar 4.2. Grafik Hasil Pengujian Induktansi

4.3. Pengujian Frekuensi Meter

Pengujian nilai frekuensi yang dilakukan seperti pada gambar 4.3 dan 4.4. Pada gambar 4.3 dapat dilihat foto hasil pengukuran frekuensi pada 542 Hz dengan tingkat *error* 0%.



Gambar 4.3. Pengujian frekuensi pada 542 Hz

Pada gambar 4.4 dapat dilihat hasil pengukuran pada frekuensi 2,1 MHz. Pada nilai frekuensi maksimal yang dimiliki AFG, FLC Meter masih dapat mengukur frekuensinya dengan baik, dengan nilai *error* yang relatif masih kecil.

Karena keterbatasan alat, maka pengukuran tak dapat dilakukan pada frekuensi yang lebih tinggi, karena AFG yang tersedia hanya mampu mengeluarkan frekuensi sebesar 2 MHz.

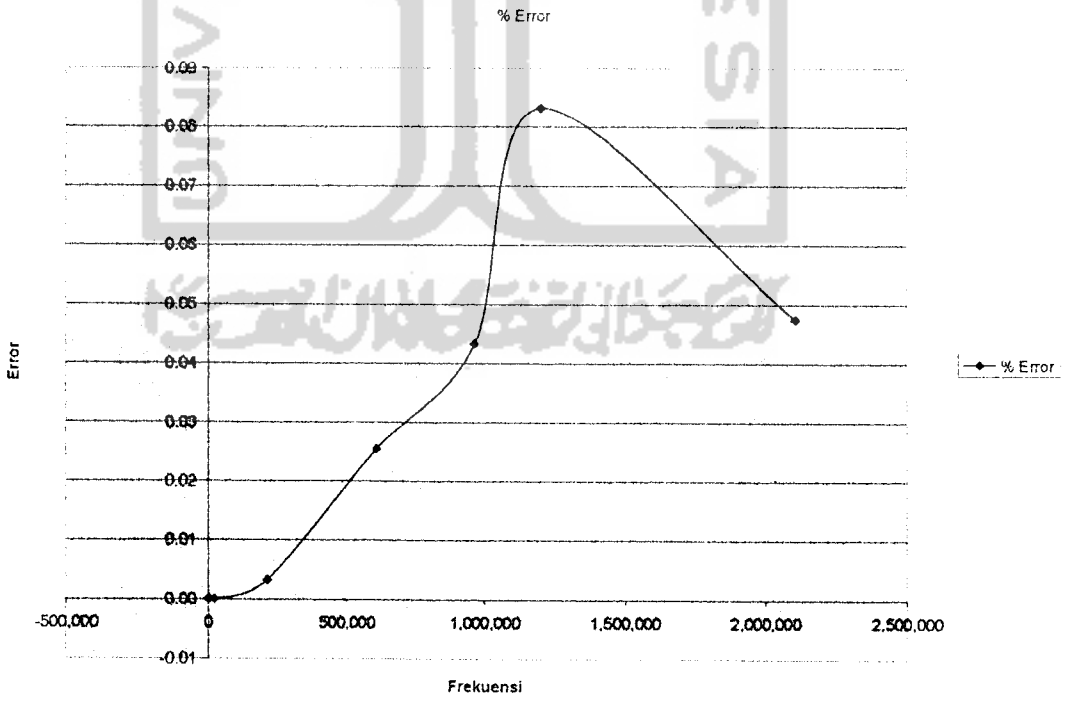
Gambar 4.4. Pengujian frekuensi pada 2,1 Hz

Dengan berdasarkan hasil pengujian yang telah dilakukan, diperoleh hasil pengujian kemampuan pengukuran frekuensi pada fungsi F Meter pada FLC Meter digital berbasis mikrokontroler AT90S2313. Hasil penelitian tersebut dapat dilihat pada Tabel 4.3 berikut.

Berdasarkan tabel hasil pengujian frekuensi tersebut, dapat diketahui nilai frekuensi berdasarkan hasil pengukuran dengan FLC Meter Digital. Berdasarkan hasil pengukuran tersebut pada tabel 4.3, dapat diketahui nilai error maksimum sebesar 0,08%.

Tabel 4.3. Tabel Hasil Pengujian Penghitungan Frekuensi

Nilai Frekuensi pada FLC Meter	Nilai Frekuensi pada AFG instek GFG-8020H	% ERROR FLC Meter terhadap AFG	Waktu Hitung
2.17 Hz	2.17 Hz	0 %	2 detik
24 Hz	24 Hz	0 %	3 detik
310 Hz	310 Hz	0 %	3 detik
585 Hz	585 Hz	0 %	3 detik
903 Hz	903 Hz	0 %	4 detik
1.021 Hz	1.021 Hz	0 %	4 detik
2.169 Hz	2.169 Hz	0 %	4 detik
3.4 Hz	3.4 Hz	0 %	5 detik
21.59 Hz	21.59 Hz	0 %	6 detik
213.693 Hz	213.7 Hz	0 %	6 detik
608.745 Hz	608.9 Hz	0.03 %	10 detik
965.082 Hz	965.5 Hz	0.04 %	16 detik
1.201.000 Hz	1.202.000 Hz	0.08 %	20 detik
2.107.000 Hz	2.108.000 Hz	0.05 %	33 detik



Gambar 4.5. Grafik nilai *error* atas pengukuran nilai frekuensi

Dari berdasarkan penelitian tersebut di atas, dapat diketahui bahwa AFG yang digunakan memiliki keterbatasan pada nilai frekuensi maksimum yang mampu dihasilkan, dalam hal ini adalah 2.108.000 Hz, atau kurang lebih adalah 2,1 MHz.

Sedangkan *error* pembacaan frekuensi dengan FLC Meter terhadap AFG INSTEK GFG-8020H adalah sesuai dengan grafik Gambar 4.3. Dari grafik tersebut dapat diketahui bahwa *error* yang terjadi masih relatif kecil, dan cenderung memiliki nilai 0%, dan memiliki nilai *error* maksimum 0,08%.

Waktu proses penghitungan hingga menampilkan ke *display* memiliki waktu yang bervariasi tergantung besarnya nilai frekuensi yang dihitung. Berdasarkan tabel di atas, dapat diketahui bahwa semakin besar nilai frekuensi yang dihitung, maka semakin besar pula waktu yang digunakan untuk menghitung dan menampilkan ke LCD. Hal ini terjadi karena semakin besar nilai yang dihitung, maka semakin besar pula nilai yang digunakan dalam *software* untuk dihitung. Maka dengan mikrokontroler 8 bit, proses itu membutuhkan waktu yang cukup lama, seperti pada penelitian yang telah dilakukan.

4.4. Pengujian Kharakteristik Kelistrikan

Pengujian kelistrikan dilakukan untuk menguji kebutuhan kelistrikan pada sistem FLC Meter digital berbasis mikrokontroler AT90S2313. Dengan mengetahui spesifikasi kelistrikannya, maka dapat digunakan dalam pengambilan keputusan berkaitan dengan kelistrikan sistem. Pengujian kelistrikan dilakukan

pada masukan DC, baik arus dan tegangan, sehingga nantinya akan dapat diperhitungkan nilai daya yang dibutuhkan sistem untuk bekerja.

Berdasarkan atas hasil pengujian kharakteristik kelistrikan yang telah dilakukan, maka dapat diperoleh hasil sesuai dengan tabel 4.4 berikut. Dari tabel tersebut dapat diketahui bahwa pada nilai tegangan 8,5V adalah tegangan yang paling efektif untuk digunakan sebagai sumber tegangan pada sistem FLC Meter Digital. Berdasarkan tabel tersebut, pada tegangan ini, daya yang dimiliki lebih kecil dan tampilan LCD dalam keadaan normal. Dengan demikian, pada FLC Meter ini dapat dipasangkan *battery* internal 9VDC.

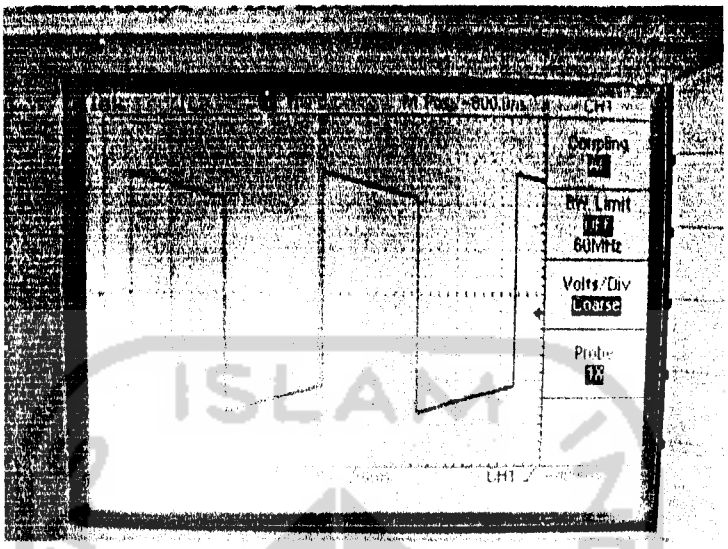
Tabel 4.4. Tabel Hasil Pengujian Kharakteristik Kelistrikan

SELEKTOR PADA ADAPTOR	TEGANGAN SEBENARNYA	ARUS BEBAN	DAYA (P) P = V x I	TAMPILAN LCD
12 V	13,72 V	59 mA	0,80948 Watt	Normal
9 V	10,2 V	58,6 mA	0,59772 Watt	Normal
7.5 V	8,5 V	57,6 mA	0,4896 Watt	Normal
6 V	7,3 V	39,2 mA	0,28616 Watt	berkedip

4.5. Pengujian Signal Osilator LC

Osilator LC yang digunakan dalam FLC Meter digital memiliki sigal osilasi yang diuji dalam dua kondisi. Hal ini karena dua kondisi ini adalah kondisi yang sangat berbeda dalam osilator mengeluarkan signal dengan frekuensi osilasinya.

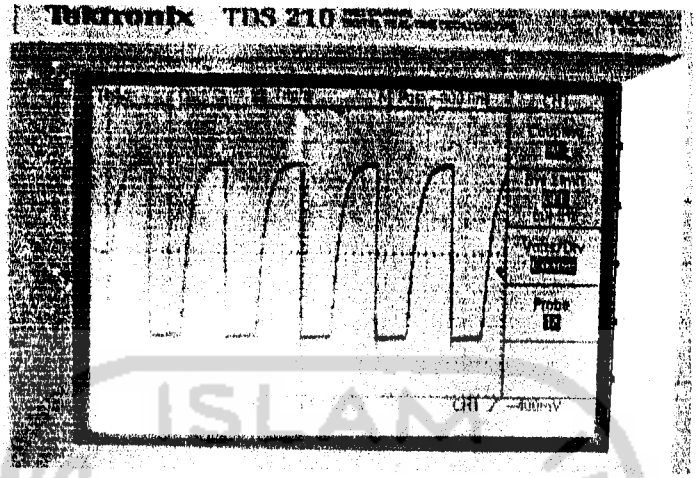
Kondisi pertama adalah kondisi saat osilator tidak sedang terhubung dengan komponen L atau C dari luar. Pada kondisi tersebut, osilator mengeluargan signal osilasi seperti pada gambar 4.6.



Gambar 4.6. Grafik signal keluaran osilator saat induktor tidak terhubung dengan osilator

Dengan osiloskom digital Tektronix, terlihat bahwa osilator mengeluarkan gelombang yang lebih mirip dengan gelombang kotak, yang memiliki frekuensi osilasi 110Hz. Hal ini terjadi ketika posisi induktor L belum terhubung pada osilator, atau dalam kondisi terbuka.

Kondisi kedua adalah kondisi saat osilator sedang terhubung dengan komponen L atau C dari luar. Pada kondisi tersebut, osilator mengeluargan signal osilasi seperti pada gambar 4.7. Pada kondisi ini, osilator telah memiliki komposisi L dan C.



Gambar 4.7. Grafik signal keluaran osilator saat induktor terhubung dengan osilator

Dari penelitian tersebut, diperoleh hasil penelitian berupa gelombang, yang bentuknya lebih mirip dengan gelombang sinus. Gelombang yang dihasilkan tersebut, nilainya memiliki frekuensi yang bervariasi, tergantung pada besarnya L dan C yang dikoneksikan secara eksternal. Sebagai sampel, berikut hasil nilai frekuensi pada saat pengukuran induktor dengan FLC Meter Digital berbasis Mikrokontroler AT90S2313.

Tabel 4.6. Tabel Hasil Pengujian Frekuensi Osilator

L (nH)	LX (nH)	LS (nH)	Frekuensi Osilator
68000	tak ada	tak terhingga	110 Hz
68000	hubung singkat (0)	68000	617.203,00 Hz
68000	2200	70200	607.550,00 Hz
68000	3900	71900	601.774,00 Hz
68000	4700	72700	597.236,00 Hz
68000	10000	78000	578.204,00 Hz
68000	120000	188000	367.848,00 Hz
68000	150000	218000	338.547,00 Hz

Sesuai dengan tabel di atas, saat induktor eksternal LX tidak terpasang, maka osilator mengeluarkan frekuensi 110Hz. Tetapi pada saat LX digantikan dengan kawat, maka induktor L akan terhubung ke osilator, sehingga LC Osilator akan memiliki kandungan L. Maka frekuensi keluaran osilator memiliki nilai frekuensi 617.203 Hz.

BAB V

KESIMPULAN DAN SARAN

5.1. Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan, diperoleh beberapa kesimpulan sebagai berikut:

1. Rancangan FLC Meter Digital berbasis Mikrokontroler AT90S2313 yang telah direalisasikan dalam bentuk *prototype* memiliki beberapa bagian utama, yaitu L/C Switch, F Switch, Callibration Switch, Display LCD 2 baris, dan pengendali utama yang menggunakan AT90S2313. Pada rancangan tersebut, FLC memiliki tiga macam pengukuran, yaitu pengukuran frekuensi, pengukuran induktansi dan kapasitansi.
2. FLC Meter Digital berbasis Mikrokontroler AT90S2313 memiliki kemampuan penghitungan kapasitansi dengan nilai *error* rata-rata 0,44%, dan dapat melakukan pengukuran 0pF hingga 2000pF.
3. FLC Meter Digital berbasis Mikrokontroler AT90S2313 memiliki kemampuan penghitungan induktansi dengan nilai *error* rata-rata 5,51% dan telah mampu menghitung nilai induktansi 0nH hingga 650uH.
4. FLC Meter Digital berbasis Mikrokontroler AT90S2313 memiliki kemampuan penghitungan frekuensi dengan nilai *error* maksimum 0,08%. FLC Meter tersebut telah teruji mampu untuk mengukur frekuensi hingga 2,1 MHz.

5. Dengan sumber tegangan 8,5V, FLC Meter Digital berbasis Mikrokontroler AT90S2313 membutuhkan arus 57,6mA, dan memiliki daya 0,4896 watt, yang membuat kondisi FLC Meter dapat bekerja dengan normal.

5.2. Saran

Berdasarkan penelitian yang telah dilakukan dan berdasarkan basil *prototype* yang telah diperoleh, dapat analisis kekurangan dan kelebihan dari alat yang telah dirancang. Namun demikian, kekurangan-kekurangan tersebut dapat diatasi dalam penelitian-penelitian yang selanjutnya.

1. AT90S2313 masih memiliki keterbatasan memori program untuk aplikasi ini, sehingga diharapakan dapat digunakan mikrokontroler dengan memori yang lebih besar untuk fleksibilitas dalam pemrograman.
2. AT90S2313 memiliki keterbatasan kecepatan pengolahan data penghitungan karena masih menggunakan system 8-bit. Untuk meningkatkan kecepatan pengolahan, maka diperlukan mikrokontroler dengan kecepatan yang lebih tinggi dan memiliki sistem pengolahan 32-bit.
3. Untuk pengukuran kapasitansi yang lebih besar diperlukan metode pengukuran yang lain untuk mendapatkan keakuratan pada nilai pengukuran tersebut.
4. Untuk pengujian yang lebih lanjut pada penghitungan frekuensi dengan FLC Meter Digital berbasis Mikrokontroler AT90S2313, diperlukan AFG yang dapat mengeluarkan frekuensi yang lebih tinggi, hingga 100MHz.

DAFTAR PUSTAKA

1. Hartono, Jogiyanto., MBA, Ph.D, (2000) , *Konsep Dasar Pemrograman Bahasa C, Andi Yogyakarta.*
2. Kadir,Abdul.,(1994),*"Pemrograman Dasar Turbo C untuk IBM PC"*, Yogyakarta, ANDI OFFSET.
3. Malvino, Albert Paul., Ph.D., E.E., (1993), *"Electronic Principles Fifth Edition"*, Singapore, McGrow-Hill Book Co.
4. Petruzella, Frank D,. (2002), Elektr*onik Industri*, Andi Yogyakarta.
5. Pratomo, Andi., (2005), *Panduan Praktis Pemrograman AVR* `
6. Turley, Jim., (1997), *"ATMEL AVR brings RISC to 8-Bit World: Better Performance than Other 8-Bit Chips With Same Low Cost"*, Microprocessor Report. *MicrokontrolerAT90S2313*, Andi Ofset..
7. Woollard, Barry., (1993), *Elektronika Praktis*, PT Pradnya Paramita, Jakarta
8. _____, (2000), *"Data Sheet AVR RISC Microcontroller"*, San Jose, ATMEL Co.

```
/*****************************************
Chip type           : AT90S2313
Clock frequency     : 10.000000 MHz
Oleh                : Muhammad Supriadi
NIM                 : 01 524 096
*****************************************/

#include <90s2313.h>

// Alphanumeric LCD Module functions
#asm
  .equ __lcd_port=0x18
#endasm
#include <lcd.h>
#include <delay.h>

#define End 1
#define Ready 0
#define Freq PIND.1

unsigned int tTCNT1;
unsigned char  ps, pp, pr,  ns, np, nr, ms, mp, mr, i, ov;
eeprom unsigned long int Constant;
unsigned long int FResult, tResult, tResult2 ; //, CResult ; //, LConstant,
CConstant;
void Ascii(unsigned char data);


// Timer 0 overflow interrupt service routine
interrupt [TIM0_OVF] void timer0_ovf_isr(void)
{
  TCNT0 = 0x0C; //reload counter value
  i++;

}



// Timer 1 output compare interrupt service routine
//interrupt [TIM1_COMP] void timer1_comp_isr(void)
interrupt [TIM1_OVF] void timer1_ovf_isr(void)
{
ov++;
```

```
}


void Ascii(unsigned char data){
 if (data == 0)lcd_putsf("0");
 else if (data == 1)lcd_putsf("1");
  else if (data == 2) lcd_putsf("2");
   else if (data == 3) lcd_putsf("3");
    else if (data == 4) lcd_putsf("4");
     else if (data == 5) lcd_putsf("5");
      else if (data == 6)lcd_putsf("6");
       else if (data == 7)lcd_putsf("7");
        else if (data == 8) lcd_putsf("8");
         else if (data == 9) lcd_putsf("9");
}


 void Display(void){


 lcd_gotoxy(2,0);
 lcd_putsf(" ");
 Ascii(mr);

 //lcd_gotoxy(1,1);
 Ascii(mp);

 //lcd_gotoxy(2,1);
 Ascii(ms);

 //lcd_gotoxy(3,1);
 Ascii(nr);

 //lcd_gotoxy(4,1);
 Ascii(np);

 lcd_gotoxy(0,1);
 //lcd_gotoxy(5,1);
 Ascii(ns);

 Ascii(pr);
```

```
//lcd_gotoxy(7,1);
Ascii(pp);

//lcd_gotoxy(8,1);
Ascii(ps);

 //lcd_gotoxy(6,1);
 if (Freq == 0)  lcd_putsf(" Hz ");
 else{
        if (PIND.2 == 1) lcd_putsf(" pF ");
        else lcd_putsf(" nH ");
 }

}


void timer0_start(void)
{
 TCCR0 = 0x00; //stop timer
 TCNT0 = 0x0C; //set count
 TCCR0 = 0x05; //start timer
}

void FrequencyCounter(void){
TCCR1B=0x86; // start Timer1

////////////////////////////////////////////
//Mulai Counter...

timer0_start();

  while(1){
   if(i == 40)break;
   #asm("wdr")
   }

////////////////////////////////////////////
// Counter Ending
tTCNT1 = TCNT1;
 TCCR1B = 0x00; //stop Timer1

 TCCR0 = 0x00; //stop timer
```

```
////////////////////////////////////////////
// Overflow Processing
 tResult = ov ;
 tResult = tResult * 65536;

 FResult = tTCNT1;
 FResult = FResult + tResult;

}

void Callibration(void){

// Constanta / Callibration
 if (PIND.0 == 0){
     Constant = FResult / 10;
     Constant = Constant * Constant;
 }

}

 void LMeter(void){

         tResult = Constant / FResult;

         tResult = tResult * 683000;
       tResult = tResult / FResult;
       tResult = tResult * 1000;
     if(tResult < 68300) tResult = 0;
     else tResult = tResult - 68300;
 }


 void CMeter(void){
          lcd_gotoxy(0,0);
         lcd_putsf("C:");

         tResult = Constant / FResult;

         tResult = tResult * 10300;
       tResult = tResult / FResult;
       tResult = tResult * 10;
```

```
    if(tResult < 1030) tResult = 0;
    else tResult = tResult - 1030;

}


void Hexa2Decimal(void){
// Hexa to Decimal
ps = 0;
pp = 0;
pr = 0;
ns = 0;
np = 0;
nr = 0;
ms = 0;
mp = 0;
mr = 0;

      //tResult  = 543210;
      tResult2 = tResult;

      while (1){

       if(tResult2 <= 0)break;


       tResult2--;
       ps++;
        #asm("wdr")
       if(ps == 10){
         ps = 0;
         pp++;
       }
       if(pp == 10){
         pp = 0;
         pr++;
       }
       if(pr == 10){
         pr = 0;
         ns++;
       }
       if(ns == 10){
         ns = 0;
```

```
      np++;
     }
     if(np == 10){
      np = 0;
      nr++;
     }
     if(nr == 10){
      nr = 0;
      ms++;
     }
     if(ms == 10){
      ms = 0;
      mp++;
     }
     if(mp == 10){
      mp = 0;
      mr++;
     }
     if(mr == 10){
      mr = 0;
     }
    //
   };

}


void main(void)
{
// Input/Output Ports initialization

// Port D initialization
PORTD=0xFF;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK = 0x82;

#asm("wdr")
WDTCR = 0x0F;

// LCD module initialization
lcd_init(16);
```

```
// Global enable interrupts
#asm("sei")

 //lcd_clear();
#asm("wdr")


while (1)
    {

ov = 0;
TCNT1  = 0;
i = 0;

FrequencyCounter();
Callibration();

//////////////////////////////////////////
// Cek apakah L atau C yang sedang dihitung
  lcd_gotoxy(0,0);
  if(Freq == 0) {
     tResult = FResult;
     lcd_putsf("F:");
  }
  else{
 if (PIND.2 == 0){
  lcd_putsf("L:");
 if(FResult <= 130) tResult = 0;
   else{

        LMeter();

  }

 }
  else{
        CMeter();

  }
}
  #asm("wdr")
```