BAB III

PERENCANAAN DAN PERANCANGAN ALAT

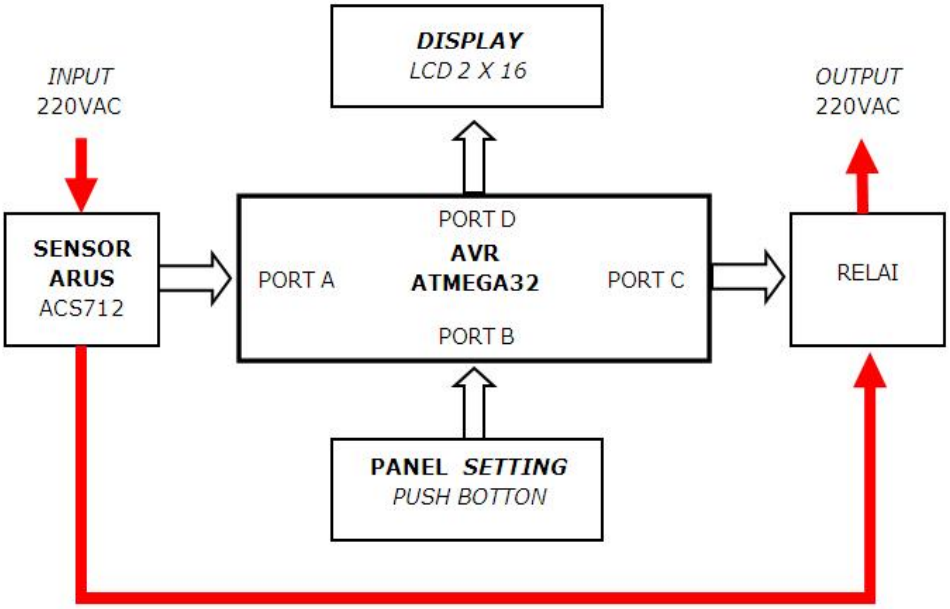
Energi listrik merupakan salah satu kebutuhan utama masyarakat, seiring meningkatnya pertumbuhan dan kesejahteraan masyarakat membuat kebutuhan energi listrik juga meningkat. Masalah pembatasan arus beban lebih merupakan hal yang sangat penting pada pengguna daya listrik rumah kost, tanpa adanya pembatasan arus beban lebih pemilik rumah kost akan sulit menentukan besaran daya dan tagihan biaya listrik pada masing-masing kamar kost.

Pembagi daya listrik umumnya sudah memiliki sistem keamanan berupa MCB (*Miniature Circuit Breaker*), namun sistem keamanan itu masih berupa thermis dan mekanik.

Gambar 3.1. MCB (*Miniature Circuit Breaker*)

Terkadang pembatasan arus pada pembagi daya listrik konvensional tidak sesuai dengan batas arus yang tertera pada masing-masing komponen pembagi daya listrik akibatnya pemutusan listrik tidak secara tepat, karena sistem keamanannya masih bersifat thermis dan mekanis sehingga kurang sensitif terhadap perubahan arus.

Untuk itu, penulis mencoba membuat sebuah sistem distribusi listrik digital yang bisa mendeteksi perubahan arus dan bisa membatasi arus dengan pembatasan yang bisa di *setting* sesuai kebutuhan. Komponen utama pada sistem ini adalah sensor arus ACS712, mikrokontroler ATMega32, *relay* pemutus dan *push botton* yang digunakan untuk mengatur batasan *setting*. Berikut blok diagram sistem distribusi listrik digital:



Gambar 3.2. Blok Sistem Distribusi Listrik Digital

Cara kerja dari sistem distribusi listrik digital ini adalah berbasis mikrokontroler AVR ATMega32 dengan sensor arus ACS712. Mikrokontroler AVR ATMega32 merupakan bagian utama pada sistem ini yang berfungsi sebagai pengendali utama (*central prosessing unit*), sedangkan sensor arus ACS712 berfungsi sebagai sensor arus yang akan mendeteksi besaran arus, hasil dari sensor arus ACS712 tadi dimasukkan ke *port* ADC *(Analog to Digital Converter*)

mikrokontroler AVR ATMega32 untuk di proses dan hasilnya akan di tampilkan pada LCD 2x16.

Push botton digunakan untuk memasukkan data *setting*an, data *setting*an tersebut akan di proses dan dibandingkan dengan data arus yang sudah diproses oleh mikrokontroler, jika data arus melebihi data setting maka mikrokontroler akan memerintahkan *relay* untuk memutus arus listrik.

3.1. Perancangan *Software*

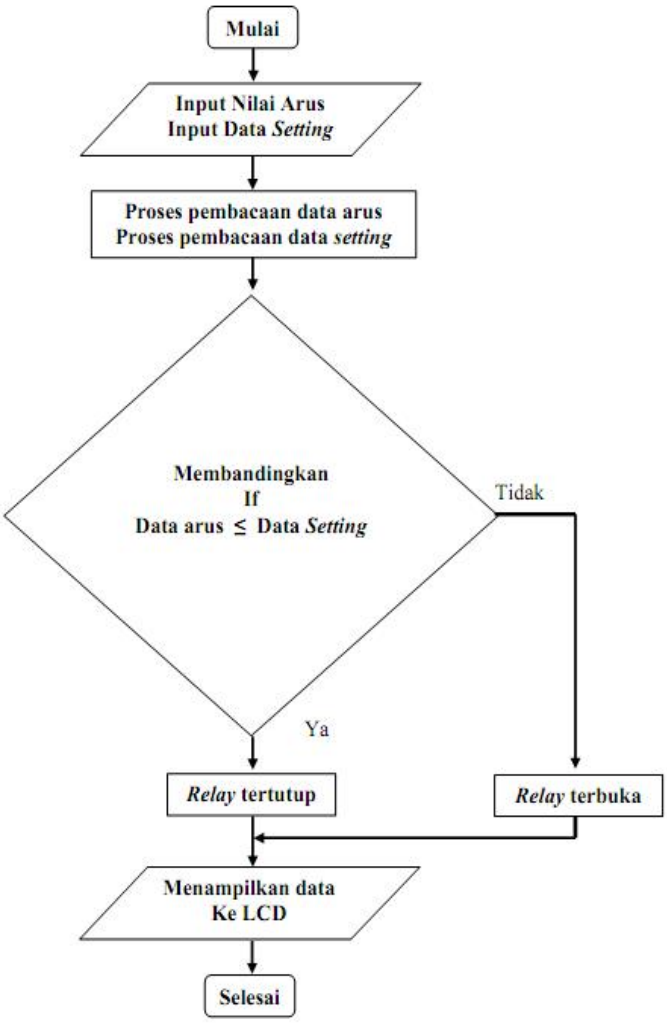
Pada perancangan *software* sistem distribusi listrik digital ini menggunakan bahasa C dan *CodeVisionAVR* sebagai *compiler*nya. Bahasa C digunakan untuk membuat program ini dibutuhkan untuk mengatur kinerja dari *hardware* sehingga dapat berjalan sesuai dengan yang diharapkan. Sedangkan *CodeVision AVR* digunakan sebagai alat bantu pemrograman (*programming toll*) yang bekerja dalam lingkungan pengembangan perangkat lunak (*sofware*) yang terintegrasi (*Intregrated Development Enviroment*, IDE).

Proses awal pemrograman adalah inisialisasi mikrokontroler kemudian dilanjutkan pada proses pengambilan data dari sensor arus ACS712. Data tersebut dibaca oleh mikrokontroler melaui port ADC *(Analog to Digital Converter*) yang ada pada microkontroler dan di proses.

Data hasil proses tadi merupakan besaran arus listrik yang di ukur oleh sensor arus ACS712, data tersebut kemudian ditampilkan pada LCD 2x16 dan di bandingkan dengan *setting* arus yang sudah dimasukkan melalui *push botton*, jika

data arus tersebut melebihi dari batas *setting* yang dimasukkan maka microkontroler akan memerintahkan *relay* untuk memutus sumber listrik.

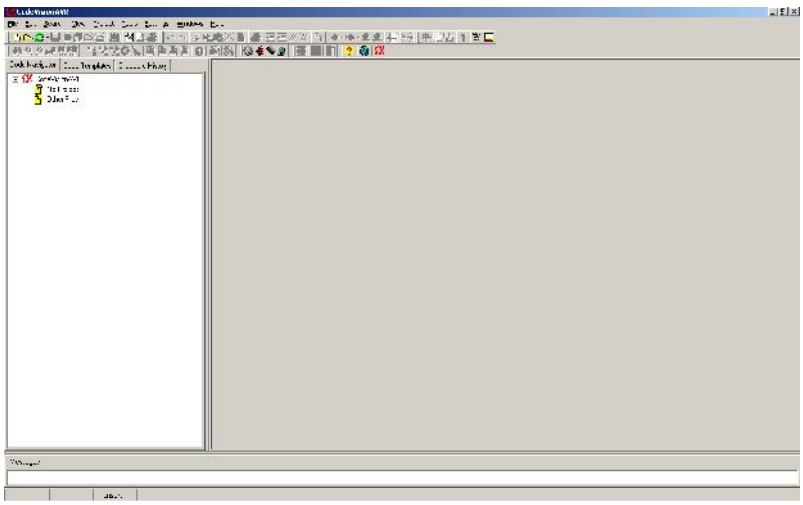
Untuk menormalkan sumber listrik yang diputus oleh mikrokontroler melalui *relay* cukup dengan menekan tombol *reset* yang ada pada panel *setting*, tombol *reset* yang ada pada panel *setting* merupakan tombol *reset* lokal yang di buat melalui program dengan cara menormalkan *relay* pemutus saja, sehingga tidak mempengaruhi kinerja dari mikrokontroler. Proses pemrograman dapat dilihat pada gambar *flowchart* 3.3.



Gambar 3.3. *Flowchart* untuk program utama

3.1.1. Pemograman Menggunakan *CodeVisionAVR*

CodeVisionAVR merupakan sistem *software C-cross compiler*, dimana program dapat ditulis menggunakan bahasa C. Dengan menggunakan pemograman bahasa bahsa C diharapkan waktu disain (*deleloping time*) akan menjadi lebih singkat. Setelah program dalam bahasa C ditulis dan dilakukan kompilasi tidak terdapat kesalahan/*error*, maka proses download dapat dilakukan. Mikrokontroler AVR ATMega32 mendukung sistem *download* secara ISP (*In-System Programing*).



Gambar 3.4. Tampilan *CodeVisionAVR*

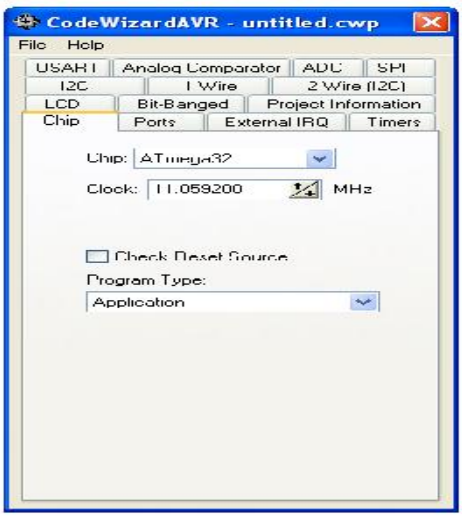
Untuk memulai pemograman menggunakan *CodeVisionAVR* pilih pada menu **File>New**, kemudian akan muncul kotak dialog pada gambar 3.5.

Gambar 3.5. Tampilan *New File*

Pilih project kemudian tekan OK, maka akam muncul kotak dialog pada gambar 3.6 berikut.

Gambar 3.6. Tampilan *option* di *CodeWizarrdAVR*

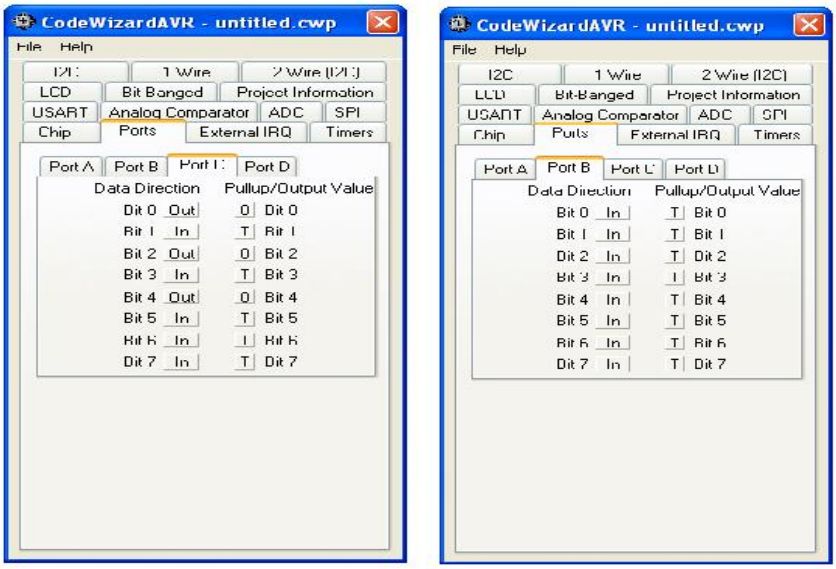
Pilih Yes untuk menggunakan *CodeWizardAVR*. *CodeWizardAVR* digunakan untuk membantu dalam mengenerate program, terutama dalam konfigurasi *Port*, *Timer*, penggunaan fasilitas-fasilitas seperti LCD, *interrupt* dan sebagainya. Pada sistem ini hanya memerlukan beberapa dari fasilitas *CodeWizardAVR*, antara lain *setting* chip, *port setting*, LCD, dan komunikasi serial menggunakan USART. Pertama kali yang harus dilakukan adalah men*setting* chip dan *clock* (kristal) yang digunakan. Pada gambar 3.7 terlihat chip dipilih ATMega32 karena sistem distribusi listrik digital ini berbasis mikrokontroler AVR ATMega32, sedangkan *clock* dipilih 11.059200 sesuai dengan kristal yang digunakan pada rangkaian minimum mikrokontroler AVR.



Gambar 3.7. Tampilan *setting chip*

3.1.1.1. *Setting Port* menggunakan *CodeWizardAVR*

Setting *port* dilakukan untuk memilih *port* mana saja yang akan dijadikan *input* atau *output*. Pada sistem distribusi daya digital ini, *port* C.0*, port* C.2 dan *port* C.4 dipilih sebagai *output* yang akan dihubungkan dengan motor relai. Setting *port C* dapat dilihat pada gambar 3.8. Data direction pada *port* C dipilih *out* dan *Pullup/Output Value* dipilih 0 , sehingga *port* C bisa digunakan sebagai *output*. Sedangkan *port* B.0 sampai *port* B.4 digunakan sebagai inputan dari *push botton* yang digunakan untuk men*setting* batasan arus pada sistem, setting *port B* dapat dilihat pada gambar 3.8.



Gambar 3.8. Tampilan *setting port*

List programnya akan tampil sebagai berikut :

#include *<mega32.h>*

// Input/Output Ports initialization

// Port A initialization

```
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In
Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T
State0=T
PORTA=0x00;
DDRA=0x00;

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

// Port C initialization
// Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out Func2=Out
Func1=Out Func0=Out
// State7=0 State6=0 State5=0 State4=0 State3=0 State2=0 State1=0
State0=0
PORTC=0x00;
DDRC=0x3F;
```

// Port D initialization

// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In

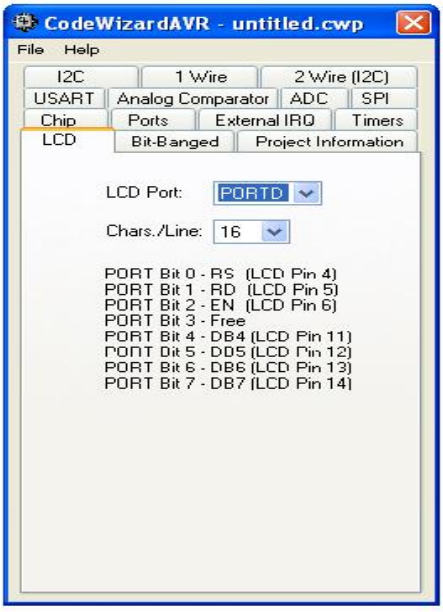
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T

PORTD=0x00;

DDRD=0x00;

3.1.1.2. *Setting* LCD menggunakan *CodeWizardAVR*

Untuk menampilkan data arus dan data *setting* maka digunakan LCD 2x16. Dengan menggunakan *CodeWizard AVR setting* LCD dapat dengan mudah dilakukan, seperti terlihat pada gambar 3.9 pada sistem distribusi listrik digital ini *Port* D dipilih sebagai jalur *input*an LCD.



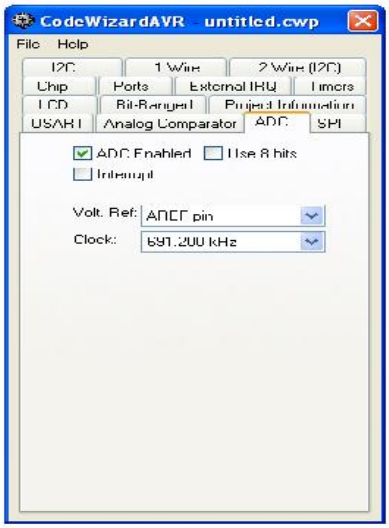
Gambar 3.9. Tampilan *setting* LCD

List programnya akan tampil sebagai berikut :

```
#include <mega32.h>
#include <stdio.h>
#include <delay.h>
#include <stdlib.h>
// Alphanumeric LCD Module functions
#asm
   .equ __lcd_port=0x12 ;PORTD
#endasm
#include <lcd.h>
```

3.1.1.3. *Setting* ADC menggunakan *CodeWizardAVR*

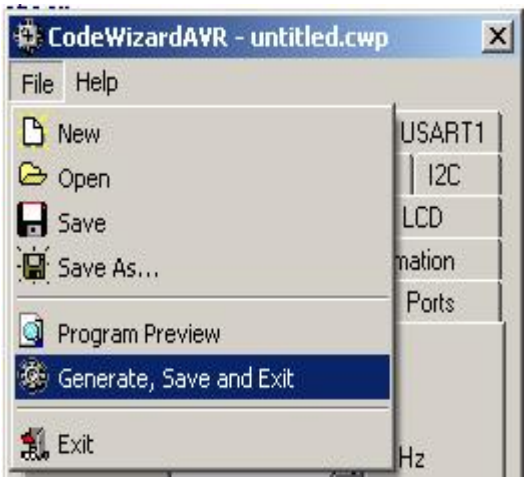
Untuk mendapatkan data arus dari modul ACS712 maka *output* dari modul ACS712 harus dihubungkan dengan *port* ADC pada mikrokontroler ATMega32, hal ini dikarenakan *output* dari modul ACS712 adalah masih berupa sinyal analog. Dengan dimasukkannya *output* dari modul ACS712 ke *port* ADC maka sinyal tersebut akan diubah oleh mikrokontroler menjadi sinyal digital sehingga dapat dibaca dan diproses oleh mikrokontroler. ADC yang digunakan pada sistem ini adalah ADC 10 *bit* untuk memudahkan dalam proses pengambilan data arus. seperti terlihat pada gambar 3.10.



Gambar 3.10. Tampilan *setting* ADC

3.1.1.4. Menyimpan Data Program

Setelah semua sudah setting sesuai kebutuhan dalam pemrograman, selanjutnya pilih menu *File,* pilih *Generate, Save and Exit* dan simpan pada direktori yang diinginkan. List program yang tersebut diatas akan tergabung menjadi satu, kemudian ditambah beberapa program tambahan agar *software* sesuai dengan kinerja *hardware.*



Gambar 3.11. Tampilan untuk menyimpan *project*

3.1.2. *Start* Program

Pada *start* program sistem distribusi listrik akan menampilkan angka nol pada LCD dan semua *relay* juga akan terbuka, hal ini disebabkan karena parameter *setting* belum dimasukkan, setelah *setting* dimasukkan tekan *reset* dan LCD akan menampilkan data *setting* dan arus pada LCD dan *relay* akan tertutup sehingga alat siap digunakan. Pada baris pertama LCD untuk menampilkan data arus dan baris kedua untuk menampilkan data *setting*.

3.1.3. Program untuk inisialisasi

Bagian ini berfungsi sebagai inisialisasi nilai awal dari parameter-parameter program, supaya program dapat berjalan sebagaimana mestinya. Potongan list program ini adalah sebagai berikut.

```
#include <mega32.h>
#include <stdio.h>
#include <delay.h>
#include <stdlib.h>
// Alphanumeric LCD Module functions
#asm
  .equ __lcd_port=0x12 ;PORTD
#endasm
#include <lcd.h>
#include <delay.h>
```

```
#define ADC_VREF_TYPE 0x40
// Read the AD conversion result
unsigned int read_adc(unsigned char adc_input)
{
ADMUX=adc_input | (ADC_VREF_TYPE & 0xff);
// Delay needed for the stabilization of the ADC input voltage
delay_us(10);
// Start the AD conversion
ADCSRA|=0x40;
// Wait for the AD conversion to complete
while ((ADCSRA & 0x10)==0);
ADCSRA|=0x10;
return ADCW;
}
float adcin1, adcin2,adcin3;
float adcina, adcinb,adcinc;
float adcinax, adcinbx,adcincx;
float v,w,x;
int alpha,i;
float maxadc1,maxadc2,maxadc3;
float  sample1[51];
float  sample2[51];
float  sample3[51];
```

char vt [33];

char wt [33];

char xt [33];

char adc1[33];

char adc2[33];

char adc3[33];

char disp1[33];

char disp2[33];

char disp3[33];

3.1.4. Pengambilan Data Arus

Pengambilan data dilakukan dengan cara memproses sinyal analog yang diterima dari modul ACS712 melalui *port* ADC pada microkontroller dan merubahnya menjadi sinyal digital.

Sinyal digital diproses menggunakan ADC 10 bit sehingga menjadi tegangan digital 0V sampai 5 Volt, setelah itu di proses ke dalam rumus sesuai dengan data sheet modul ACS712 yaitu:

Pin out = 2.5 ±(0.185 X I) Volt

Karena nilai pembacaan dari modul ACS712 yang diproses mikrokontroller fluktuatif dan nilainya berubah-ubah maka digunakan program mencari nilai maksimal dari data tersebut, program tersebut akan mengambil data nilai maksimum setiap 50 *ms*. Berikut potongan program dari pengambilan data nilai maksimum :

```
//=============================================//
      // *** == Proses Pengambilan Data Sample MCB == *** //
//=============================================//

   for (i==0; i<=50; i++)   {

      adcina  = read_adc(0);

      adcinb  = read_adc(1);

      adcinc  = read_adc(2);

      adcin1 = adcina*0.00488281; //== ADC 10 bit ==//

      adcin2 = adcinb*0.00488281; //== ADC 10 bit ==//

      adcin3 = adcinc*0.00488281; //== ADC 10 bit ==//

      adcinax   = (adcin1-2.5)/0.185; //== Rumus dari data sheet ==//

      adcinbx   = (adcin2-2.5)/0.185; //== Rumus dari data sheet ==//

      adcincx   = (adcin3-2.5)/0.185; //== Rumus dari data sheet ==//

      sample1[i]= adcinax;

      sample2[i]= adcinbx;

      sample3[i]= adcincx;

      //==== Mengambil Nilai Max ====//

        if (sample1[i] >= maxadc1)

          {maxadc1=sample1[i];}

        if (sample2[i] >= maxadc2)

          {maxadc2=sample2[i];}

        if (sample3[i] >= maxadc3)

          {maxadc3=sample3[i];}

       delay_ms(20);  }
```

3.1.5. Proses Memasukkan data *Setting*

Proses *setting* dilakukan dengan cara menekan tombol *push botton* pada panel *setting.* Dengan cara menekan tombol selektor dulu untuk menandai modul mana yang akan di *setting,* kemudian menekan tombol +/- untuk mengatur besaran *setting*. Setelah *setting* selesai kemudian tombol *reset* di tekan dan *relay* akan menutup atau terhubung. Berikut potongan program dari proses memasukkan data *setting* :

```
//================================================//
        // *** ==== Proses Selector ==== *** //
//================================================//
    if (PINB.0==1) {alpha=1;} //== Switch Select Modul 1 ==//
    if (PINB.2==1) {alpha=2;} //== Switch Select Modul 2 ==//
    if (PINB.4==1) {alpha=3;} //== Switch Select Modul 3 ==//
    if (PINB.1==1) {alpha=4;} //== Switch Select Disp 1 ==//
    if (PINB.3==1) {alpha=5;} //== Switch Select Disp 2 ==//
    if (PINB.7==1) {PORTC.0=0; PORTC.1=0; PORTC.2=0; PORTC.3=0;
PORTC.4=0; PORTC.5=0;} //== Switch Reset ==//**Reset Lokal**//
     // *** ==== ADJ Setting ==== *** //
    if (alpha==1)
        {
        if(PINB.6==1) {v = v + 0.5;}  //== Set + == Modul 1 ==//
        if(PINB.5==1) {v = v - 0.5;}  //== Set - == Modul 1 ==//
        }
```

```
if (alpha==2)
    {
    if(PINB.6==1) {w = w + 0.5;} //== Set + == Modul 2 ==//
    if(PINB.5==1) {w = w - 0.5;} //== Set - == Modul 2 ==//
    }
if (alpha==3)
    {
    if(PINB.6==1) {x = x + 0.5;} //== Set + == Modul 3 ==//
    if(PINB.5==1) {x = x - 0.5;} //== Set - == Modul 3 ==//
    }
```

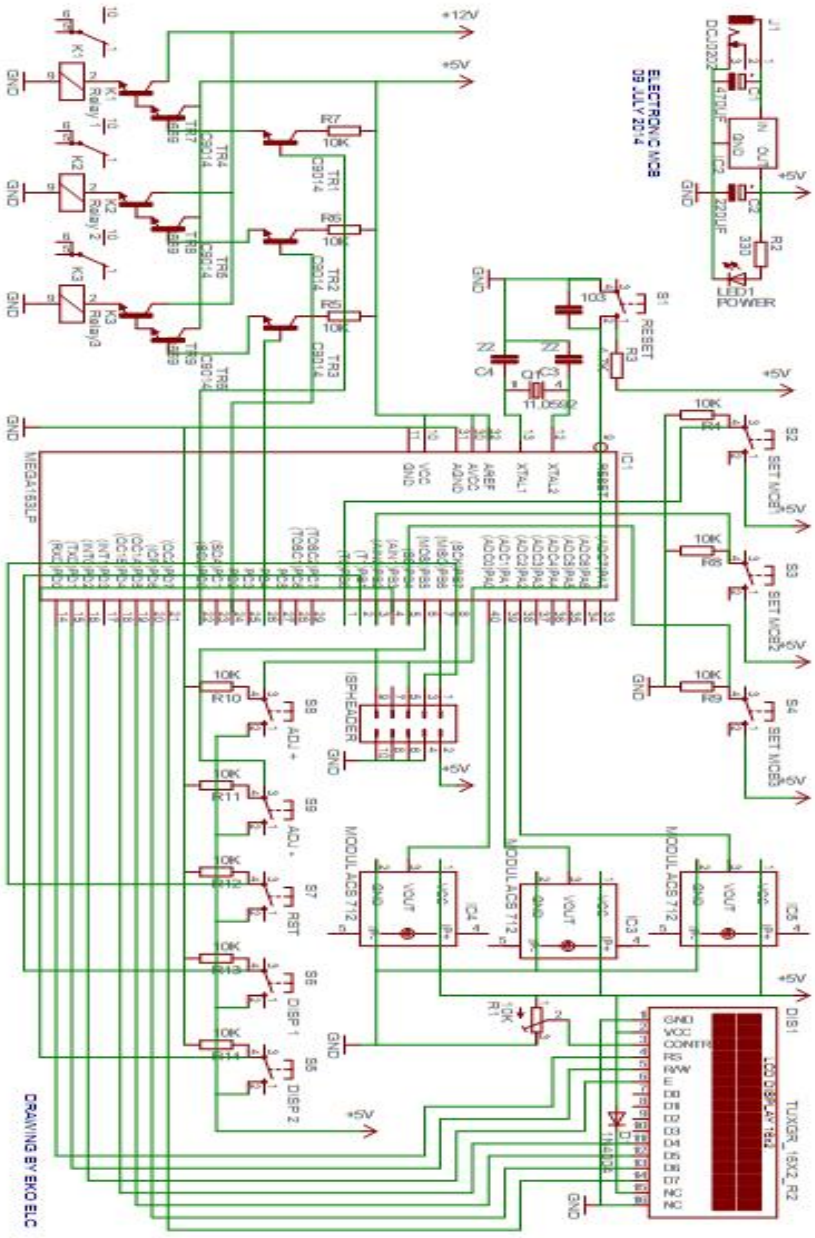
3.1.6. Membandingkan Data Arus dengan Data *Setting*

Data arus yang sudah di ambil kemudian dibandingkan dengan data *setting*, jika data arus melebihi dari data *setting* maka mikrokontroler akan memerintahkan *relay* melelui *port* B untuk memutus sumber listrik. Berikut potongan program dari proses membandingkan data arus dan data *setting* :

```
// *** ==== ADJ Setting ==== *** //
if (alpha==1)
    {
    if(PINB.6==1) {v = v + 0.5;}  //== Set + == Modul 1 ==//
    if(PINB.5==1) {v = v - 0.5;}  //== Set - == Modul 1 ==//
    }
```

```
    if (alpha==2)
        {
        if(PINB.6==1) {w = w + 0.5;} //== Set + == Modul 2 ==//
        if(PINB.5==1) {w = w - 0.5;} //== Set - == Modul 2 ==//
        }
    if (alpha==3)
        {
        if(PINB.6==1) {x = x + 0.5;} //== Set + == Modul 3 ==//
        if(PINB.5==1) {x = x - 0.5;} //== Set - == Modul  3 ==//
        }
//================================================//
        // *** ==== Proses Pembatasan Arus ==== *** //
//================================================//
         //== Modul 1 ==//
           if (maxadc1 >= v)
              { PORTC.0=1; } ==//relay 1//==
         //== Modul 2 ==//
           if (maxadc2 >= w)
              { PORTC.2=1; } ==//relay 2//==
         //== Modul 3 ==//
           if (maxadc3 >= x)
              { PORTC.4=1; } ==//relay 3//==
```

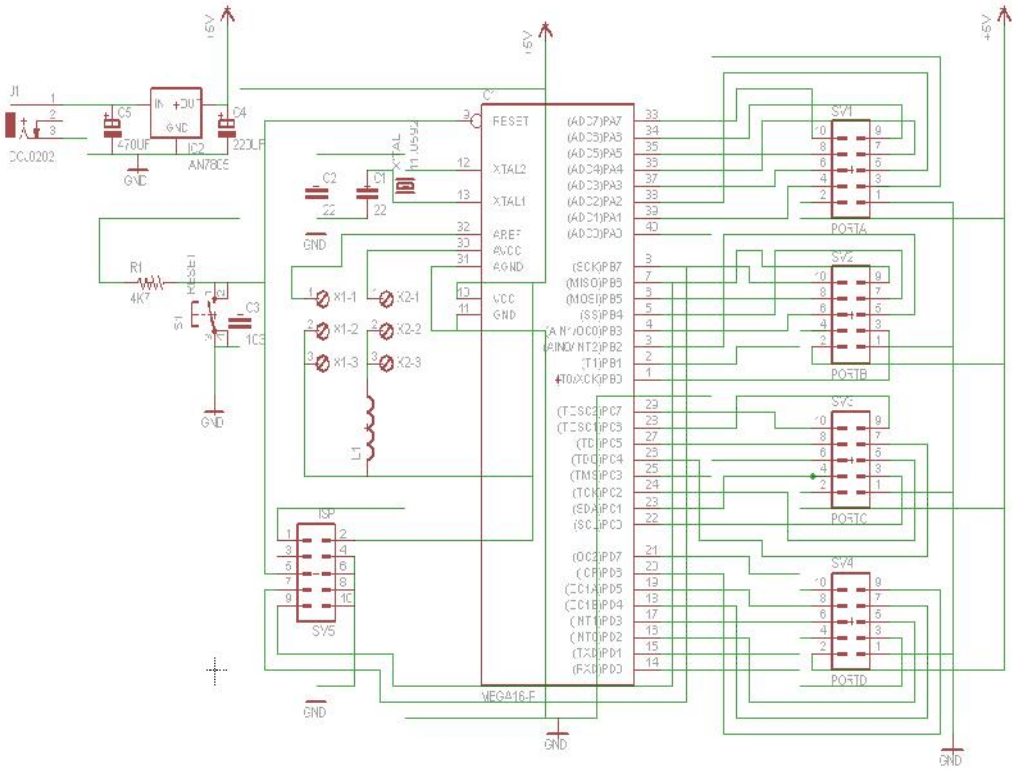
3.2. Perancangan *Hardware*



Gambar 3.12. Rangkaian skematik Sistem Distribusi Listrik Digital

Rancangan *hardware* sistem distribusi listrik digital ini terbagi atas beberapa bagian seperti terlihat pada gambar 3.12, yaitu bagian utama berupa sistem minimun mikrokontroler AVR ATMega32, bagian *inputan* berupa modul ACS712 dan *push botton switch*, bagian *output* berupa unit *display* LCD 2x16 dan *driver relay,* dan terakhir bagian *suplai* tegangan dengan catu daya 5 volt. Setiap bagian mempunyai hubungan dengan mikrokontroler dan fungsi-fungsi khusus yang nantinya dijelaskan satu persatu pada sub bab berikut ini.

3.2.1. Sistem Minimun Mikrokontroler ATMega32



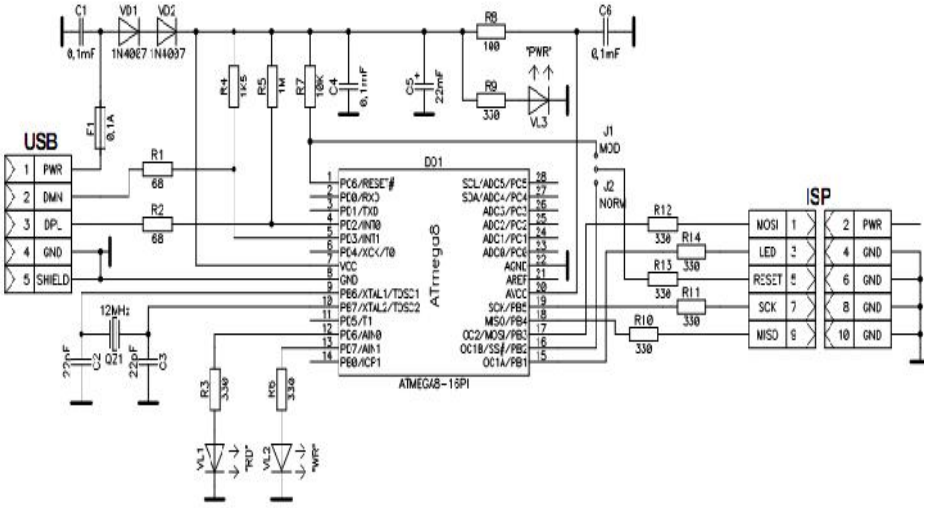
Gambar 3.13. Sistem minimum mikrokontroler ATMega32

Mikrokontroler ATMega32 dalam tugas akhir ini digunakan untuk memproses data dari modul ACS712 yang berupa sinyal analog dan merubahnya

menjadi sinyal digital melalui ADC yang ada pada mikrokontroler tersebut, memprosesnya dan menampilkannya pada LCD 2x16.

Dari gambar 3.13. menggambarkan rangkaian mikrokontroler ATMega32 secara *singlechip*. Rangkaian sistem minimum ini terdiri atas kristal, kapasitor, dan resistor dengan *suplai* tegangan sebesar 5 Volt DC. Pada *pin reset* terhubung dengan kapasitor 10 nF, resistor 4K7 Ω, dan sebuah *push button*, rangkaian *reset* tersebut berfungsi untuk mengembalikan mikrokontroler ke kondisi awal. *Reset* terjadi bila diberi logika 1 selama minimal 2 *cycle* pada kaki RST dan setelah *pin* RST kembali low, mikrokontroler akan mulai menjalankan program. Rangkaian kristal pada mikrokontroler berfungsi untuk menghasilkan pulsa, rangkaian ini terdiri dari kristal 11.059 MHz dan dua buah kapasitor bernilai 22 pF.

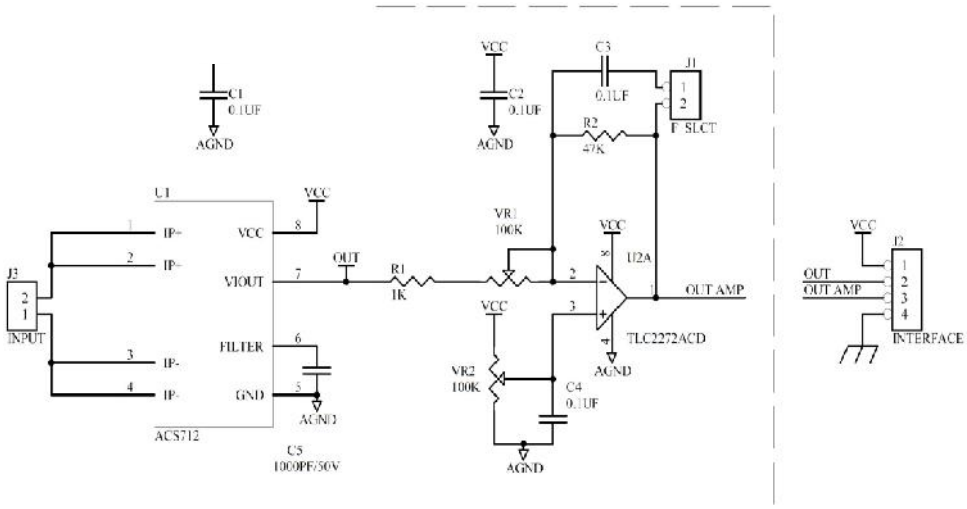
Seperti yang sudah dijelaskan diatas, fungsi dari mikrokontroler ATMega32 adalah sebagai pengendali utama (*central prosesing unit*) yang memproses dan mengendalikan sistem distribusi listrik digital dengan memasukkan semua program kedalam chip ATMega32. Pada tugas akhir ini, untuk mendownload program digunakan *port* yang tersedia di ATMega32 yaitu SCK (*pin* 8), MISO (*pin* 7), MOSI (*pin* 6), dan RESET (*pin* 9). Mendownload program ke Atmega32 melalui modul *downloader* AVR 910 USB *programer* sebagai interface dari PC ke mikrokontroler AVR ATMega32, dengan rangkaian seperti terlihat pada gambar 3.14. dibawah ini.



Gambar 3.14. Rangkaian *downloader* AVR 910 USB programer

3.2.2. Rangkaian Modul ACS712

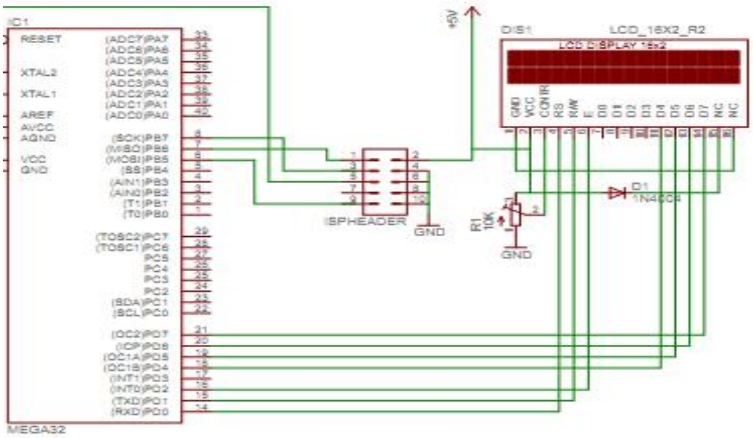
Modul ACS712 merupakan suatu modul sensor arus yang mengunakan IC sensor arus linier berbasis *Hall-Effect* ACS712 produksi Allegro. Sensor arus ini dapat digunakan untuk mengukur arus AC atau DC. Modul ACS 712 yang digunakan pada alat ini sudah dilengkapi dengan rangkaian OpAmp sehingga sensitivitas pengukuran arus dapat lebih ditingkatkan dan dapat mengukur perubahan arus yang lebih kecil. Sensor ini digunakan pada aplikasi-aplikasi di bidang industri, komersial, maupun komunikasi. Contoh aplikasinya antara lain untuk sensor kontrol motor, deteksi dan manajemen penggunaan daya, sensor untuk *switch-mode power supply*, sensor proteksi terhadap *overcurrent*, dan lain sebagainya. Berikut gambar rangkaian modul ACS712 :



Gambar 3.15. Rangkaian Modul ACS712

Dari gambar 3.15 dapat dilihat *output* dari sensor arus ACS712 dihubungkan dengan penguat operasional amplifier, *pin* 2 dari IC TLC2272ACD merupakan *pin input* dan terhubung dengan VR1 yang berfungsi untuk mengatur gain atau sensitivitas dari sensor. *Pin* 3 terhubung dengan VR2 *offset* rangkaian tersebut.

3.2.3. Rangkaian LCD 2x16 dengan Mikrokontroler ATMega32

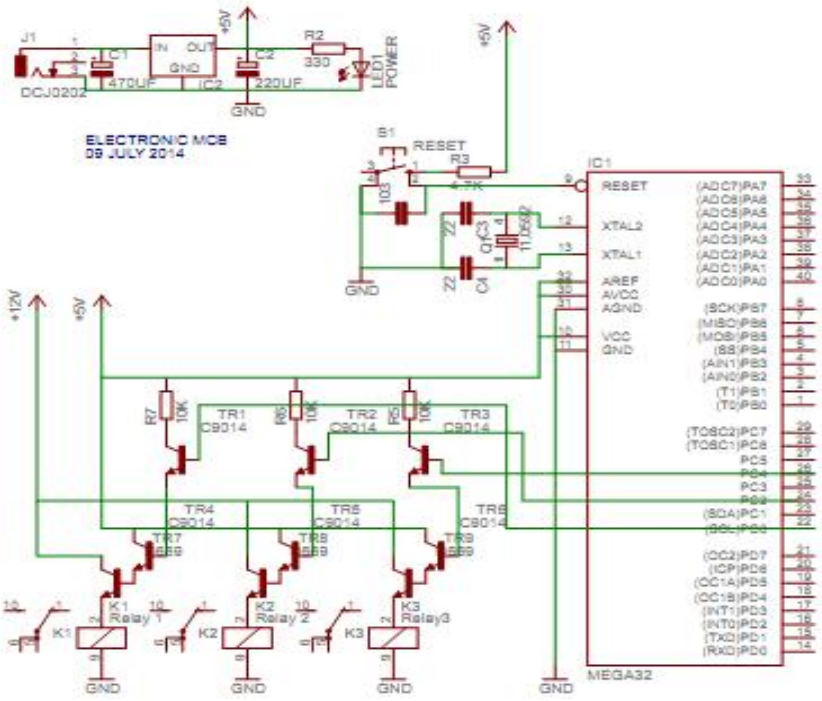


Gambar 3.16. Rangkaian LCD 2x16 dengan mikrokontroler ATMega32

LCD 2x16 sangat mudah diaplikasikan pada mikrokontroler AVR ATMega32. LCD 2x16 artinya LCD tersebut didesain untuk menampilkan data sebanyak 2 baris 16 kolom. Pada gambar 3.16 merupakan rangkaian *interface* LCD dengan mikrokontroler AVR ATMega32. Pada rangkaian tersebut dapat dilihat *pin* 1 dari LCD terhubung dengan *ground*, *pin* 2 merupakan VCC dari LCD dihubungkan dengan pin VCC mikrokontroler AVR ATMega32. Sedangkan *pin* 3 berfungsi mengatur kontras LCD, sehingga pada rangkaian diatas *pin* 3 terhubung dengan sebuah *varibel resistor* (VR) 10 KΩ. LCD memerlukan tiga jalur kontrol dan delapan jalur data (untuk mode 8 *bit*) atau empat jalur data (untuk mode 4 *bit*). Ketiga jalur kontrol yang dimaksud adalah *pin* 4 (RS), *pin* 5 (R/W), *pin 6* (E). *Pin* 4 (RS) merupakan *pin register selection signal* yang berfungsi memilih *register* yang akan digunakan, *pin* ini terhubung pada *pin* 14 (PD.0). Untuk *pin* 5 (R/W) merupakan *pin read/write signal* yang bila diberi logika '0', akan terpilih *write.* dan bila diberi logika '1', maka akan terpilih *read.* Pada saat *pin* R/W berlogika rendah, informasi pada jalur data berupa pengiriman data ke LCD (*write*). Sedangkan ketika *pin* R/W berlogika tinggi, berarti sedang dilaksanakan pengambilan data dari LCD (*read*). *Pin* 5 LCD ini dihubungkan dengan *pin* 15 (PD.1). Sedangkan *pin* 6 (E) adalah *pin Enable* yang berfungsi mengaktifkan *read/write* data, yang digunakan untuk memberitahu LCD kalau kita akan berkomunikasi dengannya. Sebelum mengirim data ke LCD *pin* ini di buat berlogika tinggi dahulu. Kemudian jalur kontrol yang lain di*setting*, pada saat bersamaan data yang akan dikirim ditempatkan pada jalur data. Setelah semua siap, *pin Enable* dibuat berlogika rendah. Transisi dari logika tinggi ke logika

rendah ini akan memberitahu LCD untuk mengambil data pada jalur kontrol dan jalur data. *Pin Enable* dihubungkan dengan *pin* 16 (PD.2) pada mikrokontroler AVR ATMega32.

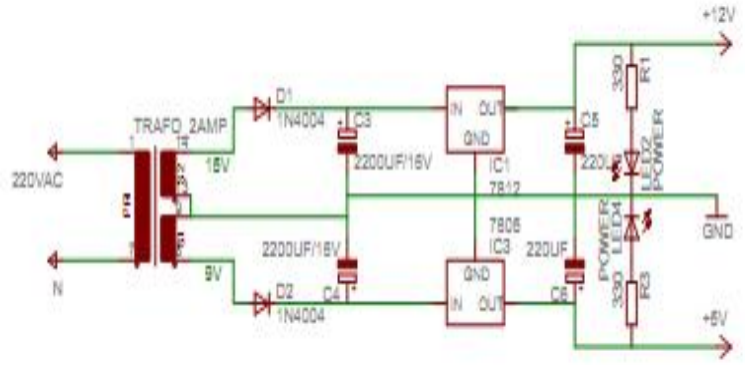
3.2.4. Rangkaian *Driver Relay*



Gambar 3.17. Rangkaian *driver Relay* dengan mikrokontroler ATMega32

Pada tugas akhir ini *driver relay* menggunakan rangkaian sakelar transistor, rangkaian sakelar transistor digunakan untuk menguatkan arus dan tegangan yang diterima dari mikrokontroller sebesar 5 Volt menjadi 12 Volt karena *relay* yang digunakan membutuhkan tegangan 12 Volt DC .

3.2.5. Rangkaian Catu Daya



Gambar 3.18. Rangkaian catu daya

Catu daya merupakan suatu rangkaian yang paling penting bagi sistem elektronika. Ada dua sumber catu daya yaitu sumber AC dan sumber DC. Sumber AC yaitu sumber tegangan bolak-balik, sedangkan sumber tegangan DC merupakan sumber tegangan searah. Catu daya yang dibutuhkan pada alat ini adalah 12 VDC dan 5 VDC. Tegangan 12 VDC digunakan untuk menggerakkan *relay* dan tegangan 5 VDC digunakan untuk mensuplai mikrokontroler AVR ATMega32, Modul ACS712 dan LCD. Tegangan 12 VDC dan 5 VDC disuplai dari transformator 2 Ampere.