BAB III

PERANCANGAN DAN PEMBUATAN

3.1 Perancangan

Secara umum, perancangan lampu led otomatis ini terdiri dari dua bagian dasar, yaitu bagian perangkat lunak (*software*), dan bagian keras (*hardware*).

Gambar 3.1 Diagram Blok Perancangan Lampu Led Otomatis

Pada penelitian ini, tahapan awal yang dilakukan yaitu melakukan perancangan perangkat keras (*hardware*). Rancangan ini berupa desain rancangan sistem kendali yang merupakan media pengolahan data masukan (*input data*) dari perangkat masukan (*input device*) untuk menghasilkan data keluaran (*output data*) pada perangkat keluaran (*output device*) sesuai dengan yang direncanakan.

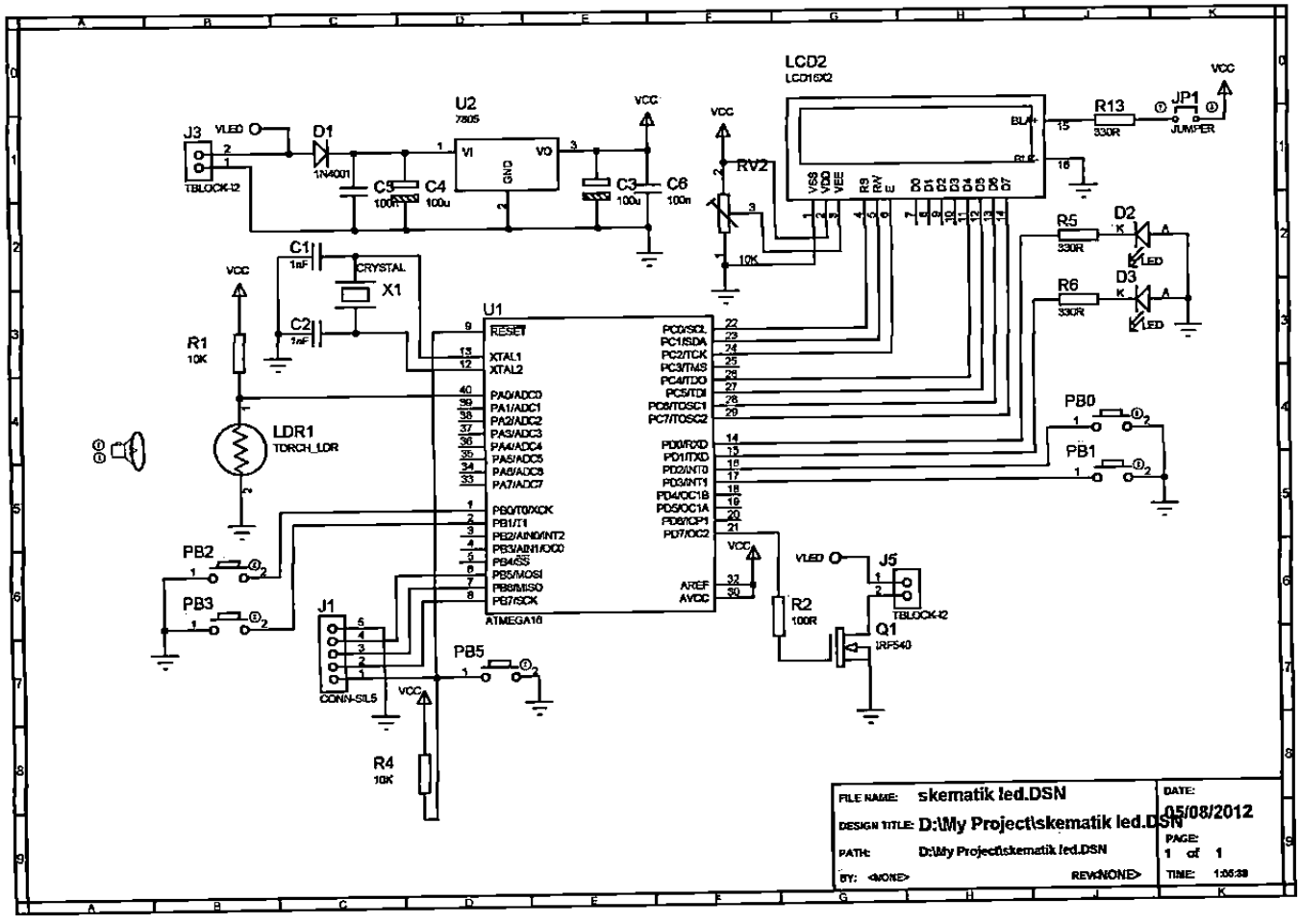
Tahapan berikutnya yaitu perancangan perangkat lunak (*software*), sebagai tindak lanjut dari hasil rancangan *hardware*. Rancangan ini berupa suatu program sistem kendali yang berfungsi untuk mengolah data masukan (*input data*) dari perangkat masukan (*input device*) untuk menghasilkan data keluaran (*output data*) pada perangkat keluaran (*output device*) sesuai dengan yang direncanakan. Rancangan *software* sebaiknya mengacu pada hasil rancangan *hardware*, sebab hasil rancangan *hardware* dipergunakan sebagai medium untuk mensimulasikan

3.1.1 Perancangan *Hardware*

Dalam perancangan *hardware* ini yang dihasilkan berupa *schematic diagram* rangkaian sistem kendali dan desain *layout* PCB (*Printed Circuit Board*). Oleh sebab itu, dalam perancangan *software* ini penulis menggunakan *software* Proteus, ISIS dan ARES.

Langkah-langkah perancangan yang dilakukan adalah sebagai berikut :

- Perancangan *schematic diagram* dengan menggunakan *software* PROTEUS ISIS.
- Perancangan *layout* PCB dengan menggunakan *software* PROTEUS ARES.



Gambar 3.2 Desain *Schematic Diagram* pada Proteus ISIS

Dikerjakan pada *software* Proteus ISIS, *schematic diagram* rangkaian sistem kendali atau rangkaian kontroler ini merupakan desain sistem rangkaian elektronis yang dibuat berdasarkan sistem minimum mikrokontroler ATMEGA16, yang menjadi pusat kendali pengolahan data masukan (*input data*) dari perangkat masukan (*input device*) yang berupa sensor cahaya jenis LDR (*Light Dependent Resistor*).

Mikrokontroler ATMEGA16 yang merupakan jantung dari rangkaian kontroler di atas akan menerima data masukan dari sensor LDR melalui PORTA.0. Data masukan dari sensor LDR berupa sinyal analog, sedangkan yang dapat diolah oleh mikrokontroler ATMEGA16 berupa sinyal digital. Sehingga data masukan itu terlebih dahulu dikonversi menggunakan fasilitas ADC (*Analog to Digital Converter*) pada mikrokontroler ATMEGA16.

Hasil konversi data analog yang telah diubah menjadi data digital tersebut, selanjutnya diolah menggunakan algoritma tertentu yang keluarannya berupa PWM (*Pulse Width Modulation*) pada PORTD.7/ OC2. PORTD.7/ OC2 terhubung pada keluaran yang berupa lampu LED 7W/23,8V. Diantara PORTD.7/OC2 dan keluaran terdapat rangkaian penguat tegangan MOSFET IRF 540 *n-channel* yang berfungsi untuk memberikan tegangan tambahan menghidupkan lampu LED yang mendapatkan *input* tegangan langsung dari *switching regulator* atau adaptor, sebab rangkaian kontroler ini hanya menghasilkan keluaran sebesar 12V.

J1 adalah *input* tegangan 5 Volt yang berasal dari *step down regulator*.

digunakan adalah xtal 16 MHz, agar dapat menghasilkan unit detik yang tepat. PORTD.4, PORTD.5, PORTB.6, dan PORTB.7 masing-masing dihubungkan dengan push button yang digunakan untuk memberikan input pada mikrokontroler. PORTD.6 dan PORTD.7, output mikrokontroler dihubungkan dengan LED sebagai LED indikator.

Tabel 3.1 Pin-pin I/O yang digunakan pada ATMEGA16

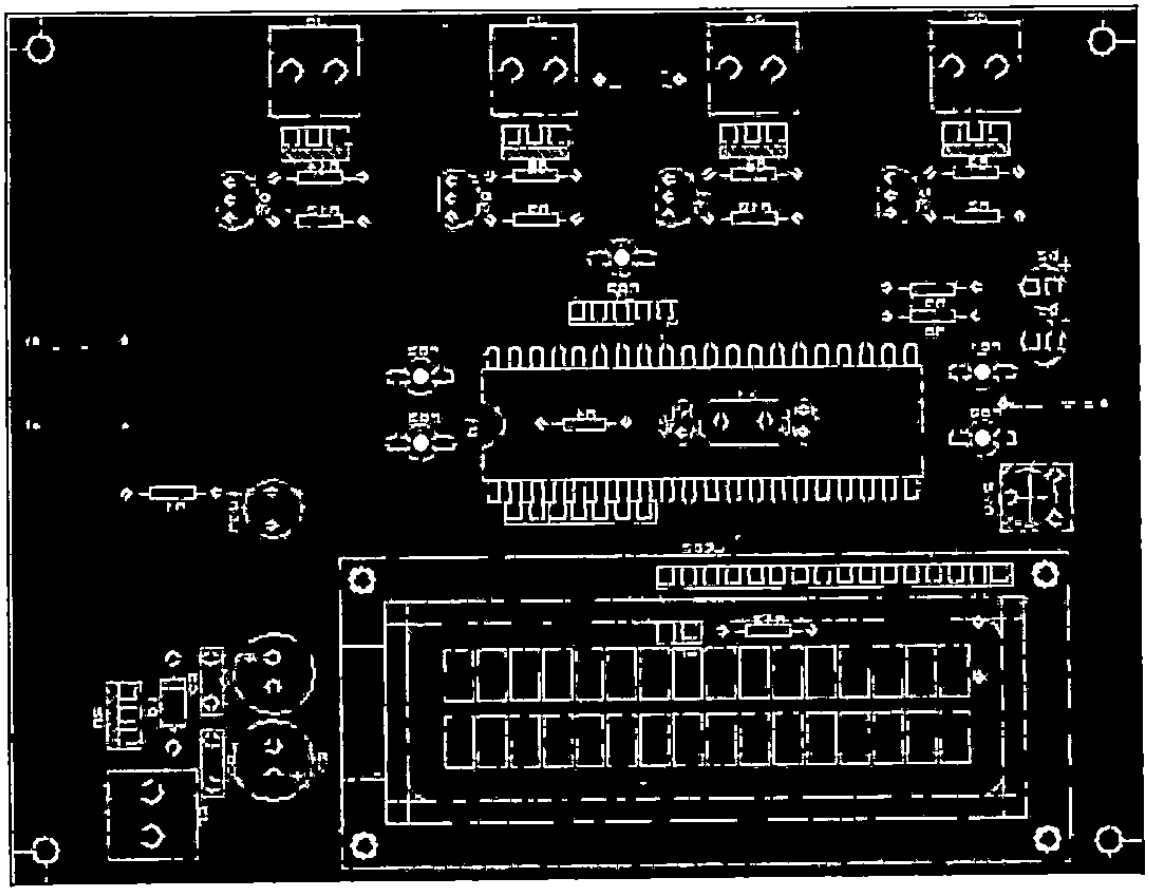
Nama Port	Pin No	Pin I/O	Tipe Pin	Fungsi
PORT A	40	0	I/O	LDR
	39	1	I/O	-
	38	2	I/O	-
	37	3	I/O	-
	36	4	I/O	-
	35	5	I/O	-
	34	6	I/O	-
	33	7	I/O	-
AREF	32	-	I/O	-
GND	31	-	I/O	-
AVCC	30	-	I/O	-
PORT C	29	7	I/O	To LCD (D4)
	28	6	I/O	To LCD (D3)
	27	5	I/O	To LCD (D2)
	26	4	I/O	To LCD (D1)

	25	3	I/O	-
	24	2	I/O	To LCD (E)
	23	1	I/O	To LCD (RW)
	22	0	I/O	To LCD (RS)
PORT D	21	0	I/O	To LED
	20	1	I/O	-
	19	2	I/O	-
	18	3	I/O	-
	17	4	I/O	Tombol
	16	5	I/O	Tombol
	15	6	I/O	LED indikator
	14	7	I/O	LED indikator
XTAL1	13	-	I/O	-
XTAL2	12	-	I/O	-
GND	11	-	I/O	-
VCC	10	-	I/O	-
RESET	9	-	I/O	-
PORT B	8	7	I/O	-
	7	6	I/O	-
	6	5	I/O	-
	5	4	I/O	-
	4	3	I/O	-

	3	2	I/O	-
	2	1	I/O	Tombol
	1	0	I/O	Tombol

Dalam pembuatan rangkaian ini diperlukan pemahaman yang utuh mengenai sistem minimum dari mikrokontroler ATMEGA16 yang akan dirancang. Sistem rangkaian yang akan dirancang diusahakan menggunakan rangkaian yang seringkas mungkin dan dengan pengkabelan yang baik. Sebab, rangkaian tersebut bekerja pada frekuensi yang relatif tinggi, sehingga peka terhadap *noise* dari luar.

Setelah menyelesaikan perancangan *schematic diagram*, selanjutnya adalah membuat *layout* PCB yang dirancang menggunakan *software* PROTEUS ARES. ARES secara otomatis akan menghubungkan titik-titik komponen dari *schematic diagram* yang telah dibuat sebelumnya di PROTEUS ISIS. Berikut adalah *layout* PCB yang dihasilkan menggunakan teknik *auto routing*. Meskipun, teknik ini juga terkadang masih terdapat *error*, sehingga sangat perlu untuk mengecek kembali hasil *auto routing*



Gambar 3.3 *Layout* PCB di Proteus ARES

3.1.2 Perancangan *Software*

Dalam perancangan *software* ini yang dihasilkan berupa program sistem kendali. Oleh sebab itu, dalam perancangan *software* ini penulis menggunakan *software* pemrograman *Code Vision AVR* (CV AVR) yang menggunakan bahasa pemrograman C. Perancangan program ini dibuat mengacu dari hasil desain sistem kendali pada perancangan *hardware* yang dilakukan sebelumnya.

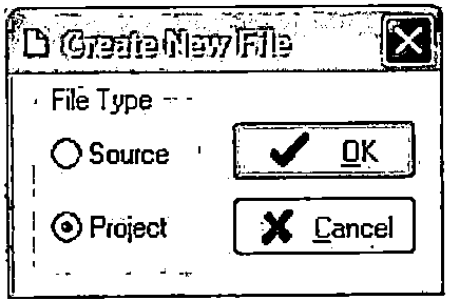
Code Vision AVR

Pemrograman dengan menggunakan *software* ini lebih mudah karena tersedia dengan bahasa pemrograman C. Selain itu, *Code Vision*

pemakai tinggal meng-klik fitur-fitur yang tersedia di dalamnya untuk membuat inisialisasi ataupun fungsi-fungsi sesuai properti yang tampil. Kemudahan yang didapatkan dalam pengoperasian CVAVR jika menggunakan *code wizard*, antara lain :

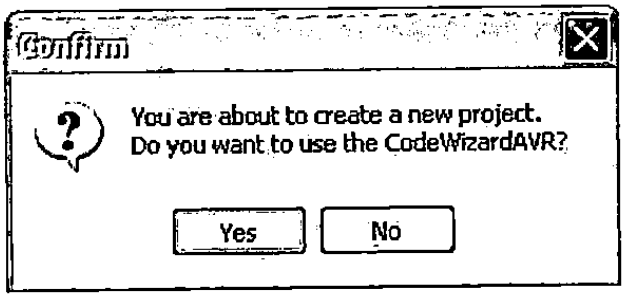
- **Membuat dan Pemilihan Chip dan Frekuensi Xtall**

  Langkah pertama dalam menggunakan CVAVR adalah membuat sebuah *project* baru, dengan meng-klik *create new project.*



Gambar 3.4 Membuat *project* baru

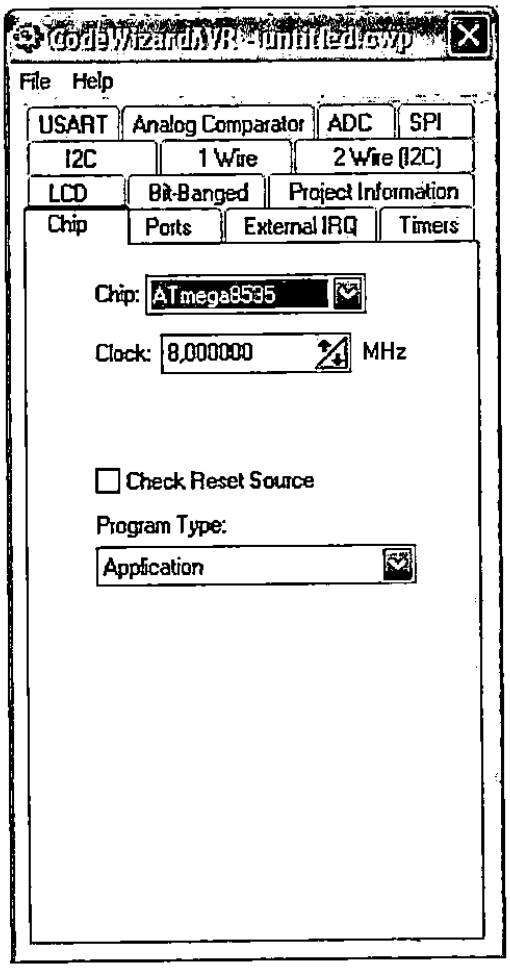
Maka akan muncul pertanyaan apakah anda ingin memanfaatkan bantuan *code wizard.*



Gambar 3.5 Pemilihan bantuan *code wizard*

Jika memilih "Yes", maka akan masuk pada *code wizard.* Langkah pertama yang harus dilakukan pada *code wizard* adalah

digunakan dalam *project*. Pemilihan chip dan frekuensi Xtall dapat dilihat pada gambar dibawah ini :



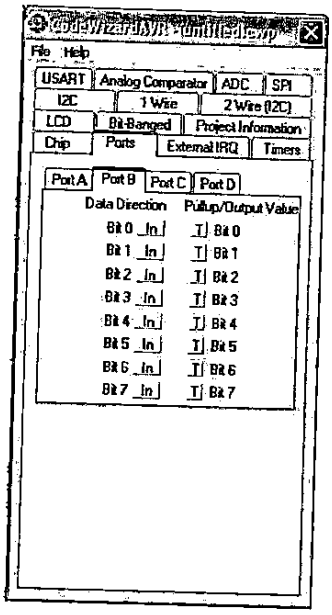
Gambar 3.6 Pemilihan Chip dan Frekuensi Xtall yang akan dipakai

- **Inisialisasi Port I/O**

Inisialisasi Port berfungsi untuk memilih fungsi port sebagai *input* atau sebagai *output*. Pada konfigurasi port sebagai *output* dapat dipilih pada saat awal setelah reset kondisi port berlogika 1 atau 0, sedangkan pada konfigurasi port sebagai *input* terdapat dua pilihan yaitu kondisi pin *input pull-down* atau *pull-up*,

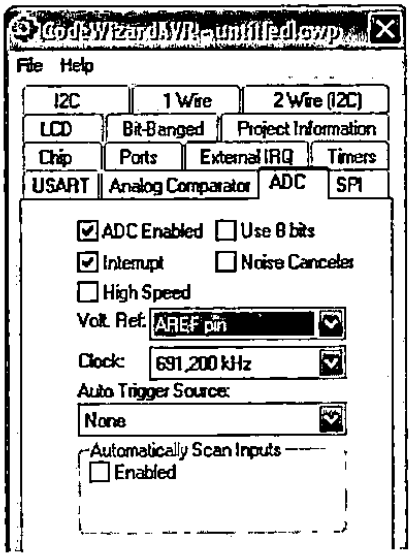
selalu berlogika 1. Setiap port berjumlah 8 bit, konfigurasi dari port dapat diatur sesuai dengan kebutuhan. Peraturan konfigurasi dapat dilakukan per-bit, jadi dalam satu port dapat difungsikan sebagai input dan output dengan nilai *default*-nya berbeda-beda.

Gambar di bawah ini menunjukkan *setting* konfigurasi pada Port B dengan kombinasi *input* dan *output* yang berbeda-beda *default*-nya.



Gambar 3.7 Inisialisasi Port I/O

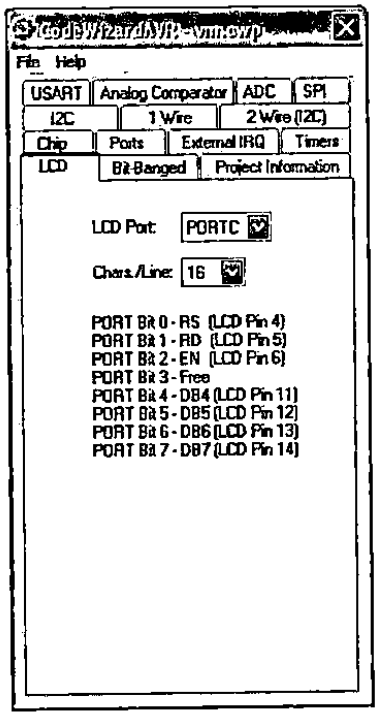
- **Inisialisasi Port ADC**



Gambar 3.8 Inisialisasi Port ADC

Analog to Digital Converter (ADC) berfungsi untuk mengubah tegangan analog ke data digital. Digunakan pada sensor, yang keluarannya masih analog. Fasilitas ADC internal pada ATMega16 ada 8 port, yaitu di PA.0 sampai PA.7.

- **Inisialisasi Port LCD**



Hasil perancangan program sistem kendali lampu otomatis dengan menggunakan *software* CV AVR adalah sebagai berikut :

```
#include <mega16.h>

#include <delay.h>

// Alphanumeric LCD functions
#include <alcd.h>
#include <stdio.h>
#define ADC_VREF_TYPE 0x00

// 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;
}
#define pwmA OCR2
#define LDR read_adc(0)
// Declare your global variables here
char lcd_buffer[33];

void main(void)
{
// Declare your local variables here
```

```
// 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=0x01;
DDRB=0x00;

// Port C 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
PORTC=0x00;
DDRC=0x00;

// Port D initialization
// Func7=Out Func6=In Func5=Out Func4=Out Func3=In Func2=In Func1=In Func0=In
// State7=0 State6=T State5=0 State4=0 State3=T State2=T State1=T State0=T
PORTD=0x00;
DDRD=0xB0;

// Timer/Counter 0 initialization
```

```
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=0xFF
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer1 Stopped
// Mode: Fast PWM top=0x00FF
// OC1A output: Non-Inv.
// OC1B output: Non-Inv.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0xA1;
TCCR1B=0x08;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: 62,500 kHz
```

```
// Mode: Fast PWM top=0xFF
// OC2 output: Non-Inverted PWM
ASSR=0x00;
TCCR2=0x6E;
TCNT2=0x00;
OCR2=0x00;

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

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

// USART initialization
// USART disabled
UCSRB=0x00;

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

// ADC initialization
// ADC Clock frequency: 1000,000 kHz
// ADC Voltage Reference: AREF pin
// ADC Auto Trigger Source: ADC Stopped
ADMUX=ADC_VREF_TYPE & 0xff;
```

```
// SPI initialization
// SPI disabled
SPCR=0x00;

// TWI initialization
// TWI disabled
TWCR=0x00;

// Alphanumeric LCD initialization
// Connections are specified in the
```

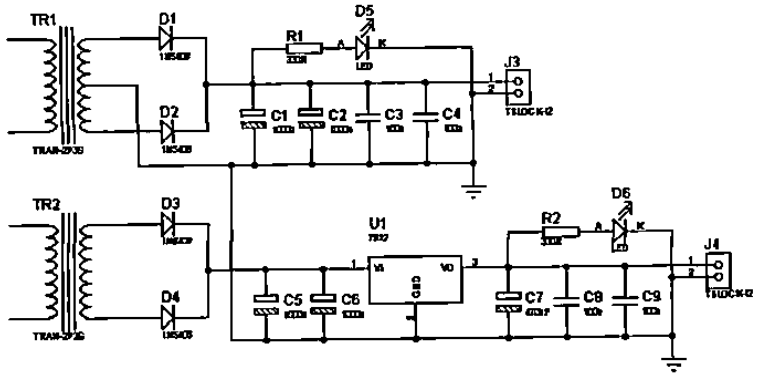
```
lcd_gotoxy(10,1);
sprintf(lcd_buffer,"%.2fV   ",0.0049*read_adc(0));
lcd_puts(lcd_buffer);

 if  (LDR<=20)                  {pwmA=0; TCCR2=0x00;}
 else                if          ((LDR>20)&&(LDR<=50))
 {pwmA=12.5;TCCR2=0x6E;}
 else                if          ((LDR>50)&&(LDR<=100))
 {pwmA=25;TCCR2=0x6E;}
 else                if          ((LDR>100)&&(LDR<=150))
 {pwmA=37.5;TCCR2=0x6E;}
 else                if          ((LDR>150)&&(LDR<=200))
 {pwmA=50;TCCR2=0x6E;}
 else                if          ((LDR>250)&&(LDR<=300))
 {pwmA=62.5;TCCR2=0x6E;}
 else                if          ((LDR>300)&&(LDR<=350))
 {pwmA=75;TCCR2=0x6E;}
 else                if          ((LDR>350)&&(LDR<=400))
 {pwmA=87.5;TCCR2=0x6E;}
 else                if          ((LDR>400)&&(LDR<=450))
 {pwmA=100;TCCR2=0x6E;}
 else                if          ((LDR>450)&&(LDR<=500))
 {pwmA=112.5;TCCR2=0x6E;}
 else                if          ((LDR>500)&&(LDR<=550))
 {pwmA=125;TCCR2=0x6E;}
 else                if          ((LDR>550)&&(LDR<=600))
 {pwmA=137.5;TCCR2=0x6E;}
 else                if          ((LDR>600)&&(LDR<=650))
 {pwmA=150;TCCR2=0x6E;}
 else                if          ((LDR>650)&&(LDR<=700))
 {pwmA=162.5;TCCR2=0x6E;}
 else                if          ((LDR>700)&&(LDR<=750))
 {pwmA=175;TCCR2=0x6E;}
```

```
        else        if        ((LDR>750)&&(LDR<=800))
        {pwmA=187.5;TCCR2=0x6E;}
        else        if        ((LDR>800)&&(LDR<=850))
        {pwmA=200;TCCR2=0x6E;}
        else        if        ((LDR>850)&&(LDR<=900))
        {pwmA=212.5;TCCR2=0x6E;}
        else        if        ((LDR>900)&&(LDR<=950))
        {pwmA=225;TCCR2=0x6E;}
        else        if        ((LDR>950)&&(LDR<=1000))
        {pwmA=237.5;TCCR2=0x6E;}
        else if (LDR>1000)
        {pwmA=255;TCCR2=0x6E;}
        else;
      }
}
```

3.1.3 Perancangan *Switching Regulator*

Pada penelitian ini dipergunakan *switching regulator* yang berfungsi untuk menurunkan tegangan dan menyearahkan tegangan AC yang diterima dari listrik tegangan tinggi 220 Volt menjadi tegangan DC yang sesuai dengan rangkaian kontroler lampu LED otomatis.



Switching regulator atau adaptor ini memiliki dua keluaran. Keluaran yang pertama berfungsi untuk menurunkan tegangan sesuai dengan kebutuhan tegangan masukan dari rangkaian kontroler yaitu sebesar 12 Volt. Pada rangkaian kontrolernya sendiri sudah terdapat *step down regulator* untuk meregulasi tegangan masukan tersebut menjadi sebesar 5 Volt untuk mengakomodasi kinerja mikrokontroler ATMEGA16. Hal ini juga bertujuan untuk keamanan penggunaan mikrokontroler yang hanya bisa digunakan jika menggunakan tegangan masukan dikisaran 4,5V-5,5V saja.

Keluaran yang kedua berfungsi untuk menaikkan tegangan keluaran dari rangkaian kontroler. Tegangan keluaran yang dihasilkan oleh rangkaian kontroler ini adalah 12V, setelah melalui rangkaian penguat tegangan. Namun, tegangan tersebut tidak mencukupi untuk menghidupkan lampu LED 7 Watt yang menjadi keluaran dari rangkaian kontroler ini. Lampu LED tersebut membutuhkan tegangan tercatat mencapai 23,8 Volt.

Sebab lain dari penggunaan *switching regulator* ini adalah kontinyuitas konsumsi sumber daya listrik untuk menyalakan lampu LED pada ruangan yang direncanakan, yang lebih tahan lama jika dibandingkan menggunakan baterai.

3.2 Pembuatan

Pembuatan ini meliputi realisasi rancangan seluruh rangkaian tersebut di atas. Dimulai dari pengadaan bahan, persiapan alat, pengerjaan, dan pengujian. Penelitian mengenai tugas akhir ini dikerjakan selama 5 bulan,

dilakukan di Universitas Muhammadiyah Yogyakarta dengan menggunakan peralatan yang tersedia di Laboratorium Teknik Elektro.

3.2.1 Pengadaan Bahan

Bahan-bahan yang digunakan pada tahap pembuatan adalah sebagai berikut :

- 1 lembar PCB (13cm x 10cm)
- Bahan kimia $FeCl_3$
- Fotokopi transparansi *layout* PCB
- Terminal blok
- *Tools box*
- Komponen elektronika yang berupa :
  - Mikrokontroler ATMEGA16
  - Sensor LDR (*Light Dependent Resistor*)
  - LED 7 Watt/ 23,8 Volt merk SUJICON
  - LCD 2x16 character
  - Trafo CT 3 Ampere
  - LM 7812
  - LM 7805
  - *Xtal* 16 MHz
  - MOSFET IRF 540
  - LED indikator 3mm
  - Kabel
  - Beberapa komponen pendukung lainnya, seperti resistor, kapasitor,

3.2.2 Persiapan Alat

Alat-alat yang digunakan dalam pembuatan ini antara lain :

- Solder dan tenol
- Laptop (*software* Proteus, dan Code Vision AVR)
- USB ISP *downloader*
- Setrika listrik
- Bor listrik dan mata bornya
- *Cutter*
- *Tools set* (obeng, tang, tang potong)
- *Desoldering attractor*
- *Laboratory Power Supply*
- Multimeter
- *Function Generator* GW Insteck SFG-2010
- *Digital Storage Oscilloscope* GW Insteck GDS-2102

3.2.3 Perangkat Lunak (*Software*)

Perangkat lunak yang digunakan adalah :

- Aplikasi program Proteus, ISIS SCH dan ARES PCB *Layout*
- Aplikasi program Code Vision AVR

3.3 Pengerjaan

Pengerjaan dimulai dengan pembuatan PCB seluruh rangkaian yang telah dirancang sebelumnya. Teknik pembuatan PCB yang dilakukan adalah teknik

Langkah awal yang harus dilakukan adalah *layout* PCB dicetak menggunakan printer, selanjutnya print out tersebut di fotokopi transparansi. Hasil transparansi tersebut dipanaskan dan ditekan pada permukaan PCB menggunakan setrika listrik. Setelah yakin semua hasil setrika berpindah tempat dari transparansi ke PCB, diamkan sejenak hingga panas pada permukaan PCB berangsur-angsur mendingin. Ketika permukaan PCB telah dingin, maka lapisan transparansi dapat dilepas secara perlahan.

Proses selanjutnya, PCB dilarutkan dalam larutan $FeCl_3$, agar jalurnya dapat tercetak. Untuk mempercepat proses pelarutan, goyangkan wadah tempat pelarutan secara kontinyu. Setelah jalur tercetak, tahapan selanjutnya yaitu pengeboran lubang-lubang komponen dan pembersihan jalur tembaga pada PCB.