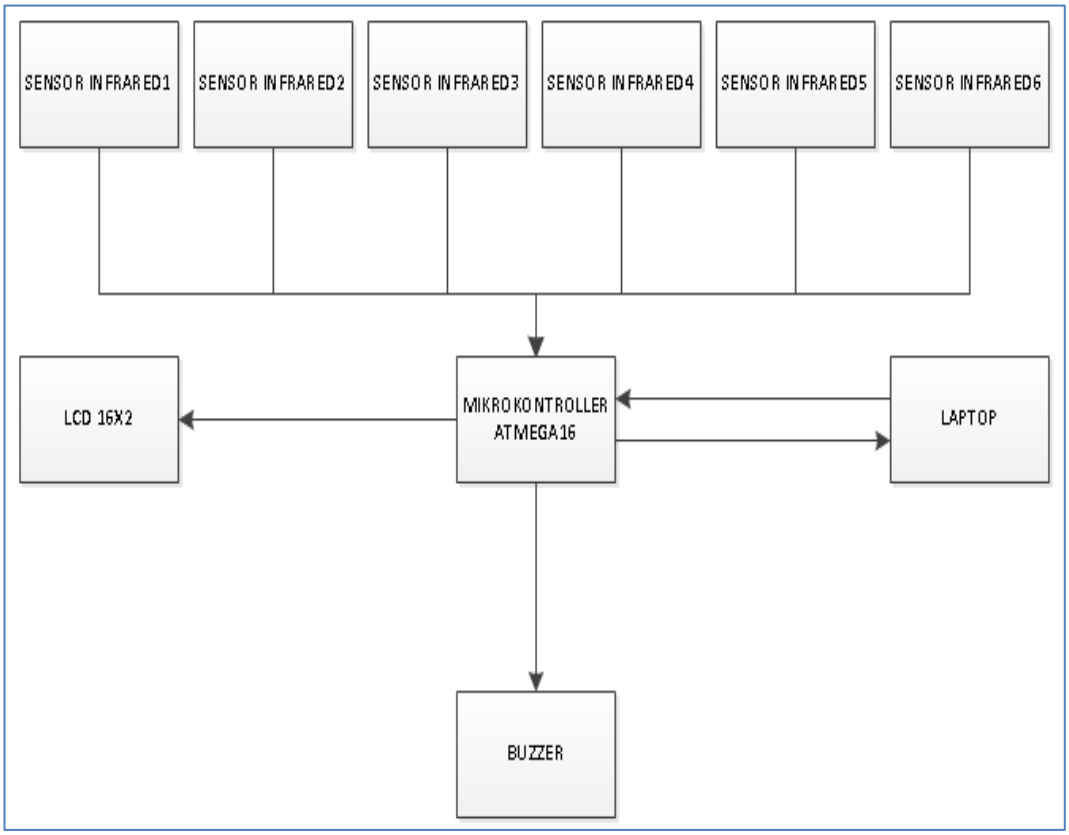
BAB III

PERENCANAAN DAN PEMBUATAN

3.1. Blok Diagram

Blok diagram alat sensor parkir yang telah dibuat adalah sebagai berikut:



Gambar III.1 Blok Diagram Alat

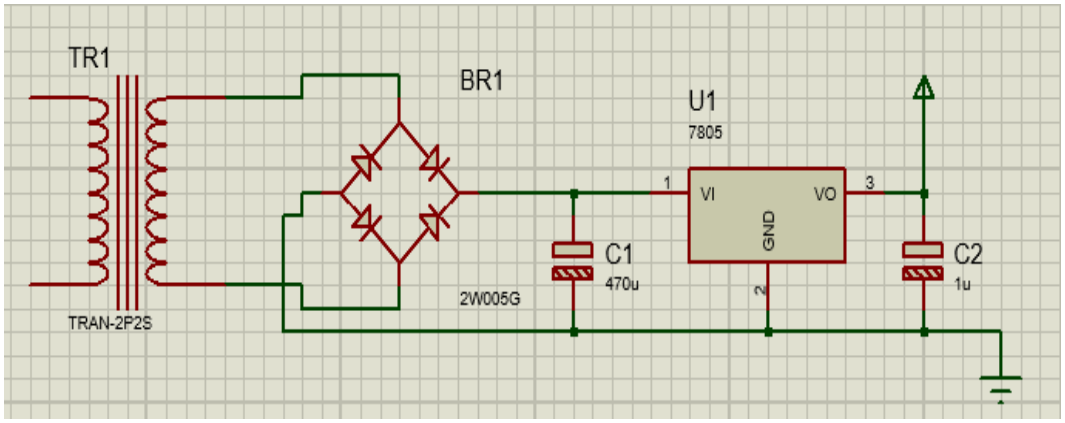
Pada alat ini, alat telah diaktikan sebuah catu daya, sistem akan memeriksa input dari sensor *infrared*. Setelah mendapat input dari sensor *infrared* maka mikrokontroler akan mengolah inputan tersebut sehingga akan tampil pada layar LCD untuk jumlah slot parkir baik yang terisi maupun yang masih kosong. Kemudian mikrokontroller juga akan mengirimkan data input dari sensor *infrared* menuju PC.

Data dikirimkan mikrokontroler ke komputer menggunakan komunikasi serial yang sudah tersedia. Kemudian komputer akan mengolah data inputan tersebut sehingga dapat memonitoring slot parkir yang tersedia. Sementara fungsi buzzer adalah sebagai indikator atau penanda bahwa slot parkir sudah penuh terisi..

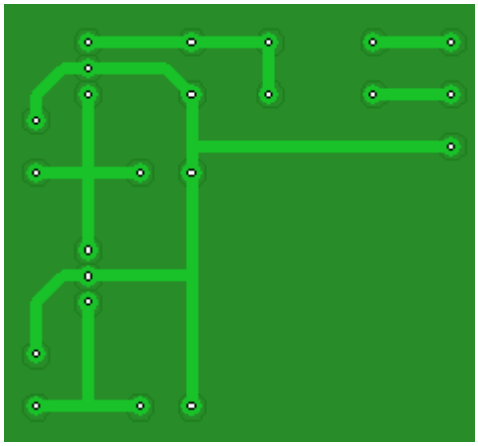
3.2. Perencanaan Catu Daya (*Power Supply*)

Catu daya adalah sebuah alat yang digunakan untuk menurunkan tegangan AC sehingga tegangan tersebut dapat digunakan komponen-komponen lain agar dapat bekerja sebagaimana mestinya. Catu daya yang dibuat menggunakan Trafo *Step down*, dioda, kapasitor, IC Regulator 7805.

Cara kerja catu daya ini adalah ketika Trafo CT mendapat tegangan AC sebesar ±220V, maka Trafo ini akan menurunkan tegangan *output* ±12V dengan arus 1A. Kemudian tegangan AC tersebut akan masuk ke dioda untuk disearahkan sehingga akan menghasilkan arus searah (DC). Setelah itu arus akan masuk ke kapasitor, IC Regulator 7805. Tugas IC Regulator ini adalah untuk menstabilkan tegangan yang dihasilkan sesuai dengan jenis dan kebutuhannya. Dalam sistem rangkaian ini tegangan DC yang dibutuhkan adalah 5V.



Gambar III.2 Rangkaian Catu Daya



Gambar III.3 *Layout* PCB Catu Daya

3.3. Perencanaan Input

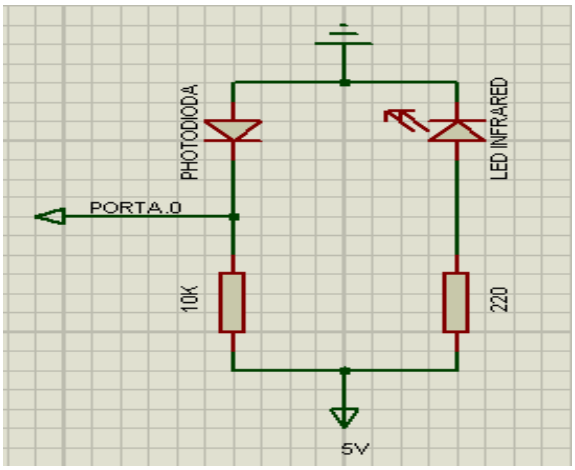
3.3.1. Perencanaan Sensor *Infrared*

Rangkaian sensor *infrared* merupakan salah satu rangkaian yang digunakan sebagai input ke mikrokontroler. Rangkaian ini berada pada PORTA, yaitu PORT yang dapat digunakan untuk mengkonversi sinyal analog yang didapat oleh mikrokontroler menjadi sinyal *digital* (ADC).

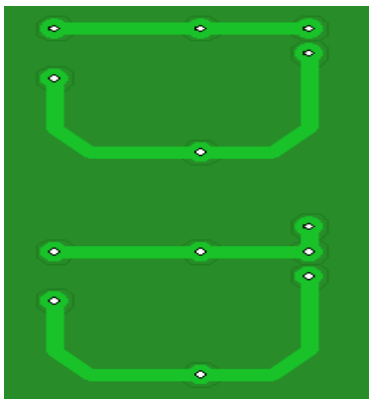
Pada rangkaian sensor ini, *photodioda* digunakan sebagai sensor cahaya dan LED *Infrared* sebagai sumber cahaya. Ketika LED *Infrared* menembakkan cahaya ke *photodioda*, *photodioda* akan menerima cahaya dalam intensitas yang tinggi dan akan memiliki nilai resistansi yang rendah sehingga memberikan tegangan keluaran yang sangat kecil. Hal ini akan dibaca oleh mikrokontroler ke dalam bentuk digital, yaitu berlogika 0.

Sebaliknya jika cahaya yang ditembakkan LED *Infrared* terhalang oleh sesuatu, maka *photodioda* akan menerima intensitas cahaya dalam jumlah yang kecil sehingga nilai resistansi *photodioda* akan tinggi dan ini akan mengakibatkan tegangan keluaran yang dikeluarkan *photodioda* cukup besar. Hal ini pula akan

dibaca oleh mikrokontroler kedalam bentuk digital, yaitu berlogika 1. Nilai tegangan inilah yang digunakan sebagai inputan PORTA.0 – PORTA.5 pada mikrokontroler yang telah dibuat.



Gambar III.4 Rangkaian Sensor *Infrared*



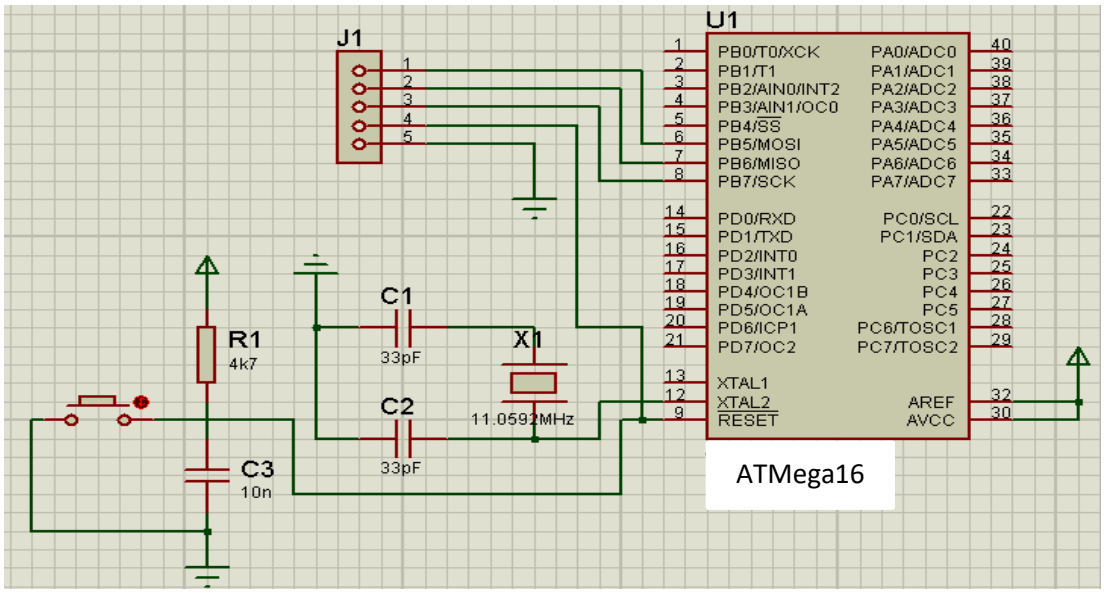
Gambar III.5 *Layout* PCB Sensor *Infrared*

3.4. Perencanaan Proses

3.4.1. Perencanaan Rangkaian Sistem Minimum Atmega16

Sistem minimum adalah rangkaian minimal dimana chip mikrokontroler dapat bekerja. ATMega16 membutuhkan tegangan (vcc) sebesar 5V dan akan bekerja pada frekuensi *oscillator* yang dipakai. Mikrokontroler ini mempunyai *oscillator* internal yang dapat digunakan sebagai penghasil *clock* yang menggerakan CPU.

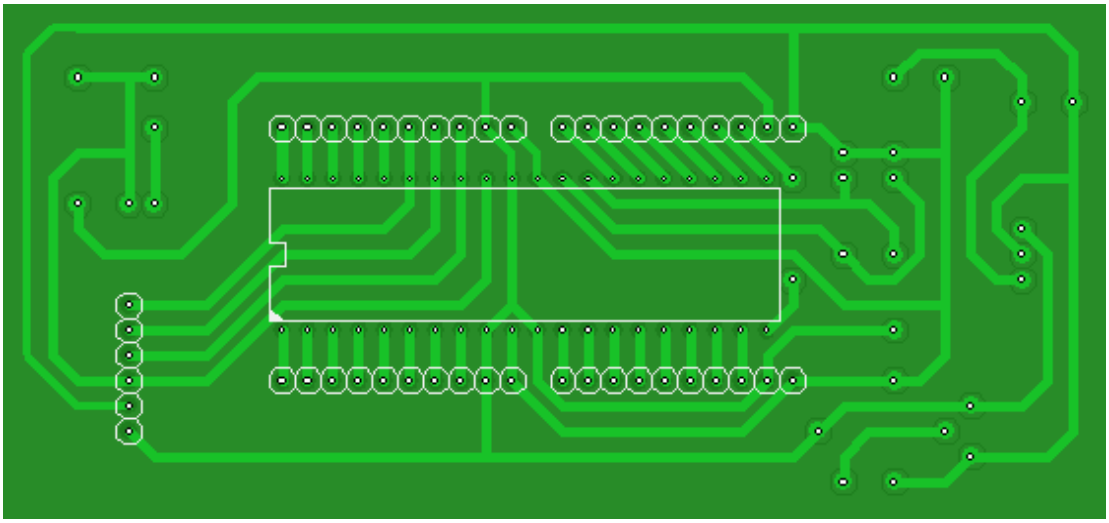
Pada sistem ini akan dipakai *Oscillator Crystal* dengan frekuensi 11,0592 MHz dan dipakai dua buah kapasitor 33pf yang dihubungkan dengan XTAL1 dan XTAL2 pada Mikrokontroler. Pemilihan *Crystal* dengan frekuensi ini adalah dengan pertimbangan agar kesalahan pada penentuan Kecepatan Transmisi (*baud rate*) dapat diperkecil ditingkat *interface* Komputer dan menghasilkan frekuensi penggerak stabil. Sistem minimum ini dilengkapi dengan pin-pin yang digunakan untuk men-*download* program ke mikrokontroler. Pin-pin tersebut adalah MOSI (PORTB.5), MISO (PORTB.6), SCK (PORTB.7), pin RESET, dan *Ground*.



Gambar III.6 Rangkaian Sistem Minimum ATMega16

Selain itu Mikrokontroler memiliki saluran *reset* aktif tinggi (*high*) sehingga saluran *reset* ini harus dijaga agar tetap berada pada kondisi rendah (*low*). Pin RESET digunakan untuk *me-reset* program (mulai keadaan awal 0000H) dengan memberikan sinyal *high* pada pin. Namun digunakan sebuah *resistor pull-down* yang dihubungkan dengan *ground* agar pin RESET tidak berada pada kondisi mengambang (*floating*). Agar mikrokontroler di *reset* pada

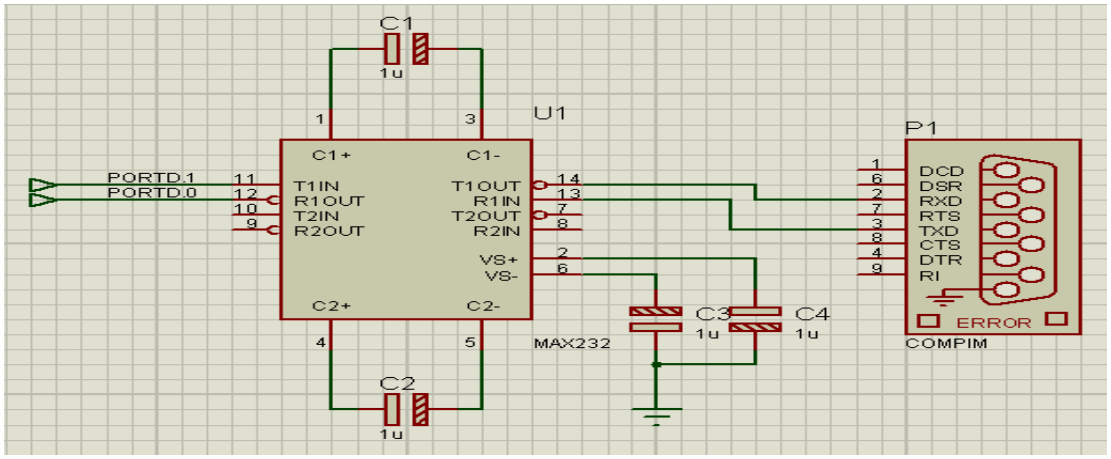
saat dihubungkan dengan sumber tegangan, maka pin *reset* dihubungkan dengan VCC melalui kapasitor 10nF.



Gambar III.7 *Layout* PCB Sistem Minimum ATMega16

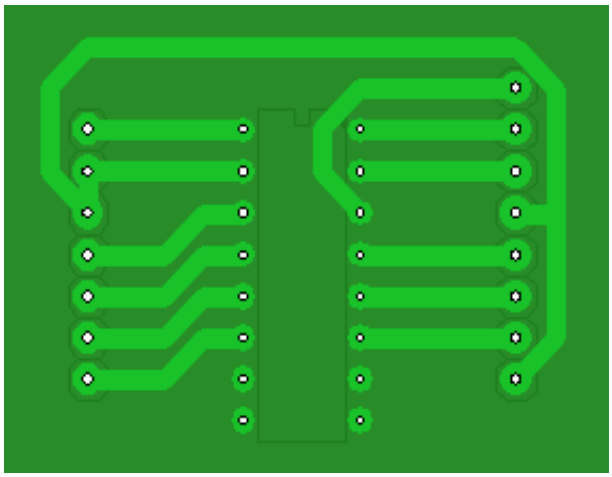
3.4.2. Perencanaan Rangkaian Komunikasi Serial RS232

Pada komunikasi serial antara PC dengan mikrokontroler ATMega16 yang digunakan adalah rangkaian RS232. RS232 diperlukan dalam komunikasi serial antara PC dan mikrokontroler karena adanya perbedaan tegangan pada jenis komunikasi ini.



Gambar III.8 Rangkaian Komunikasi Serial RS232

Pin 11 pada IC MAX232 terhubung dengan PORTD.1 mikrokontroler dan pin 12 pada IC MAX232 terhubung dengan PORTD.0 mikrokontroler. Sedangkan untuk komunikasi ke PC pin IC MAX232 yang digunakan adalah pin 13 dan pin 14, pin 13 terhubung oleh pin 3 (TXD) dan pin 14 terhubung oleh pin 2 (RXD) pada PC. RS232 memerlukan 4 buah kapasitor berukuran 1µF yang dipasang sesuai dengan gambar III.8.



Gambar III.9 Layout PCB Komunikasi Serial RS232

3.5. Perencanaan *Output*

3.5.1. Perencanaan Rangkaian LCD

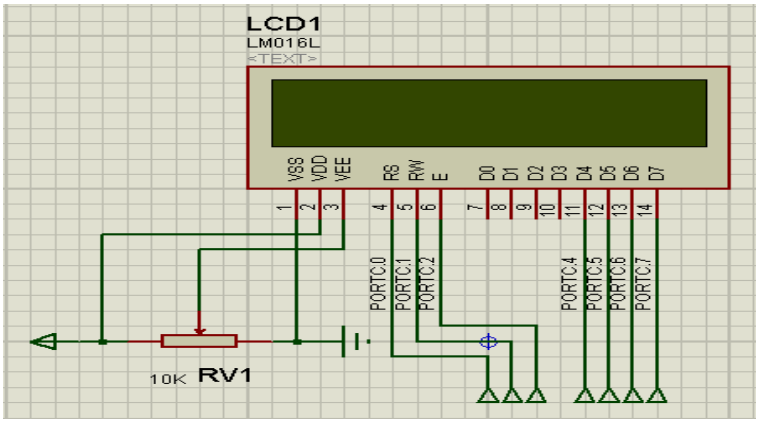
LCD 16x2 merupakan salah satu *output* dari skema rangkaian pintu keamanan yang menampilkan *character*, *string*, serta *indicator* hasil verifikasi dari mikrokontroler ATMega16. LCD 16x2 ini akan menampilkan slot parkir baik yang masih tersedia maupun yang sudah terisi.

PORT yang digunakan LCD untuk berhubungan dengan mikrokontroler adalah PORTC. PORTC merupakan pin I/O dua arah dan memiliki fungsi khusus. Berikut ini adalah konfigurasi pin LCD 16x2:

Tabel III.1 Konfigurasi Pin LCD 16x2

No.	Pin	Fungsi
1	Vss	OV (GND)
2	Vcc	5V
3	VLC	LCD *Contras Voltage*
4	RS	*Register Select*; H : Data *Input*; L : *Intruction Input*
5	RW	H : *Read*; L : *Write*
6	EN	*Enable Signal*
7	D0	*Data Bus* 8 bit (tidak digunakan)
8	D1	
9	D2	
10	D3	
11	D4	*Data Bus* 4 bit (digunakan)
12	D5	
13	D6	
14	D7	
15	V+BL	Positif *Backlight Voltage* (4-4,2 V; 50-200mA)
16	V-BL	Negatif *Backlight Voltage* (0V; GND)

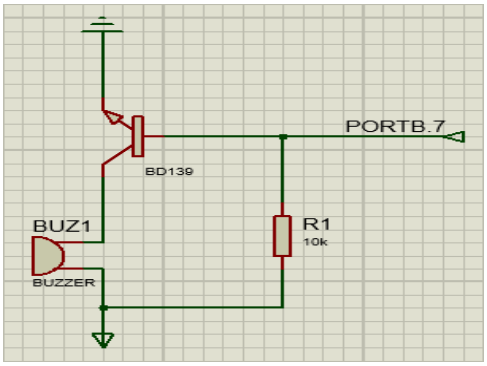
Dari tabel diatas dapat dijelaskan bahwa LCD ini membutuhkan tegangan 5V dengan variabel *resistor* 10K sebagai pengatur kontras pada monitor LCD. LCD ini dapat membaca karakter yang dikirimkan oleh mikrokontroler melalui *data bus* 4 bit. Berikut ini skema rangkaian LCD 16x2 yang terhubung dengan mikrokontroler.



Gambar III.10 Skema Rangkaian LCD 16x2

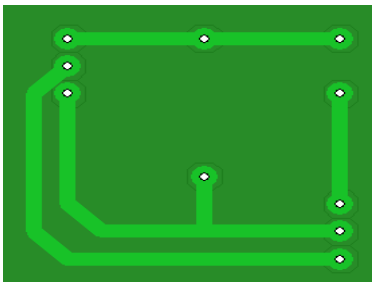
3.5.2. Perencanaan Buzzer

Buzzer merupakan salah satu indikator suara yang digunakan untuk memberi tanda sebuah kondisi. *Buzzer* ini akan aktif pada dengan tegangan 6V-12V yang dikendalikan langsung oleh mikrokontroler ATMega16 melalui PORTA.6. *Buzzer* ini akan berbunyi jika slot parkir pada gedung sudah terisi penuh.



Gambar III.11 Skema Rangkaian *Buzzer*

Cara kerja dari rangkaian diatas adalah ketika tidak ada data dari mikrokontroler atau data *low* (0), maka arus ke basis akan 0 sehingga hubungan antara kolektor dan emitter pada transistor akan terputus dan *buzzer* tidak akan aktif. Ketika mikrokontroler mengeluarkan data 1 melalui PORTA.6 menuju basis transistor maka kolektor akan terhubung dengan *emitter*, karena *emitter* terhubung dengan *ground* maka akan terjadi aliran arus dari kolektor ke *emitter* yang melalui *buzzer*, sehingga *buzzer* akan aktif.

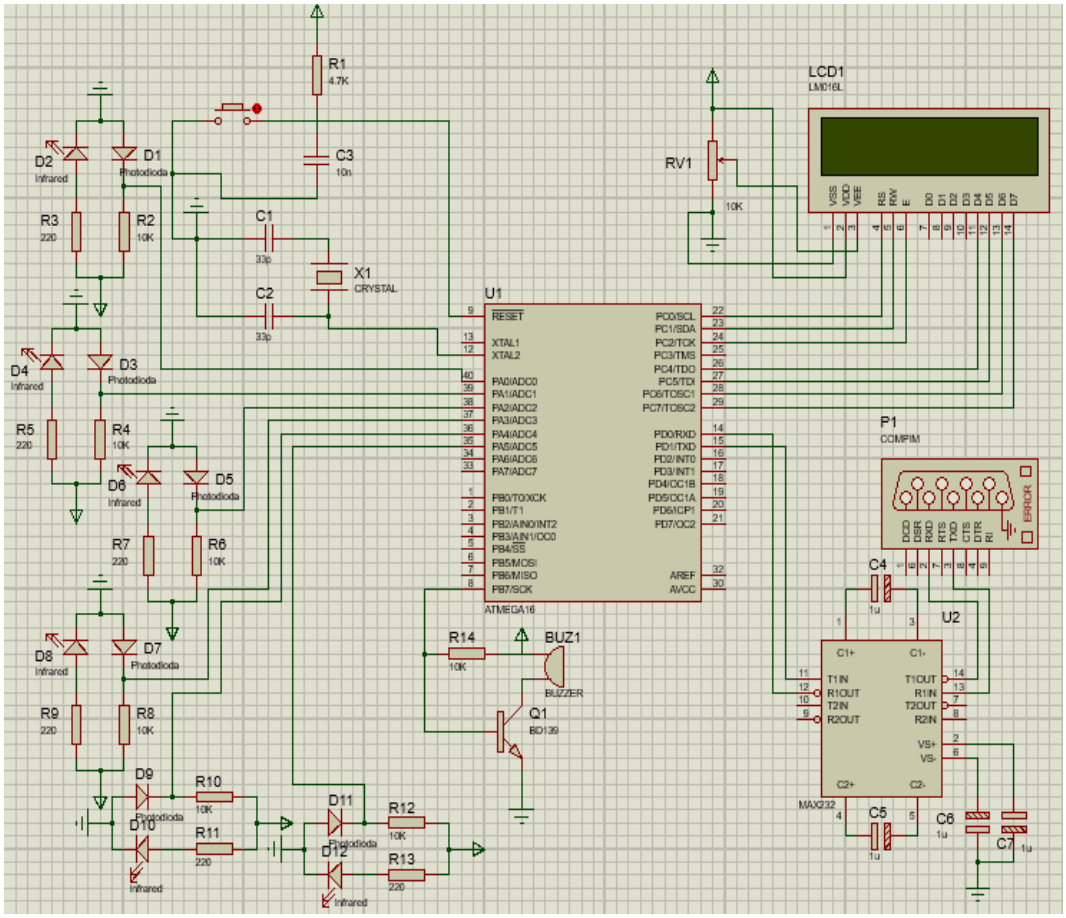


Gambar III.12 *Layout* PCB *Buzzer*

3.6. Rangkaian Keseluruhan

3.6.1. Skematik Rangkaian

Dari skema rangkaian alat yang dibuat dapat dijelaskan bahwa terdapat 6 *sub* rangkaian yang terhubung dengan mikrokontroler ATMega16 yaitu rangkaian rangkaian catu daya (*power supply*), sistem minimum ATMega16, rangkaian sensor *infrared*, rangkaian komunikasi serial antara mikrokontroler dan PC, rangkaian LCD, rangkaian *buzzer*. Rangkaian – rangkaian tersebut saling berhubungan satu sama lain sehingga menjadi satu-kesatuan alat pintu keamanan yang dibuat secara miniatur. Berikut ini skema rangkaian alat secara keseluruhan yang telah dibuat.

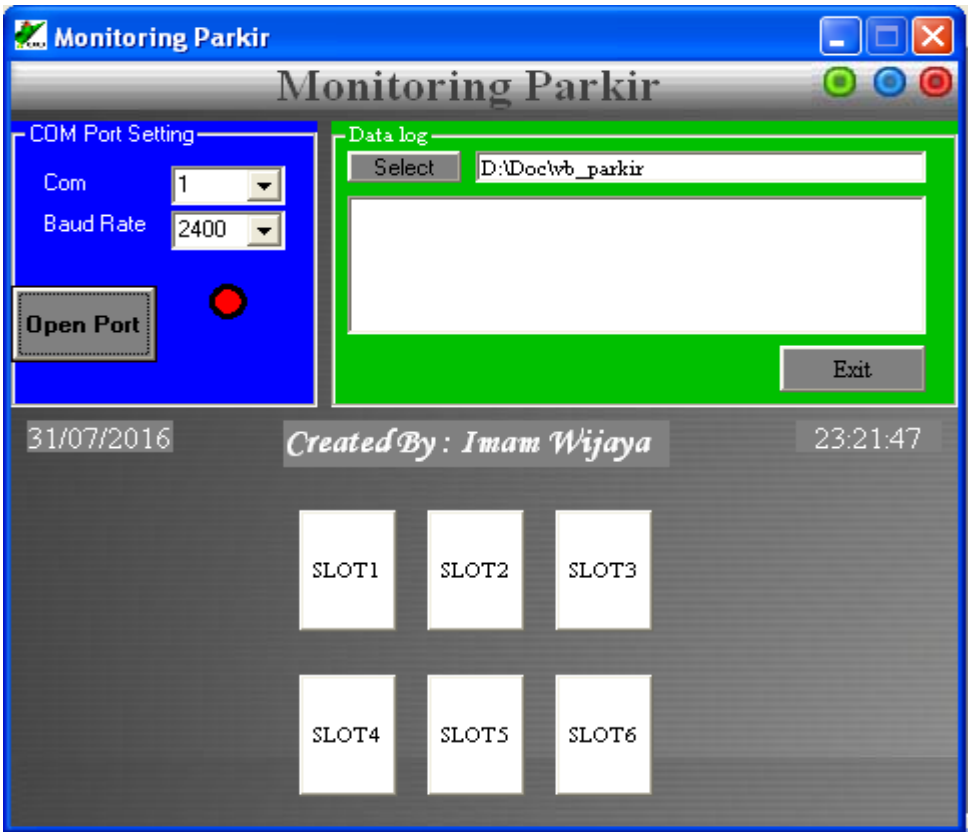


Gambar III.13 Skema Rangkaian Alat

3.6.2. Cara Kerja Alat

Berikut ini adalah proses kerja alat secara keseluruhan mulai dari awal pemberian catu daya.

a. Ketika catu daya dihubungkan maka semua rangkaian menerima tegangan yang telah diatur kebutuhannya sehingga rangkaian siap bekerja.
b. Sensor infrared akan mengirimkan sinyal kepada mikrokontroller untuk masing-masing slot pada area parkir.
c. Mikrokontoler ATMega16 akan memproses data inputan dari sensor infrared kemudian akan ditampilkan pada layar LCD dan akan dikirimkan pula ke komputer melalui komunikasi serial.
d. Di dalam komputer data inputan tersebut akan diolah dengan program *Visual Basic*, dan akan ditampilkan pada layar aplikasi visual basic slot area parkir.

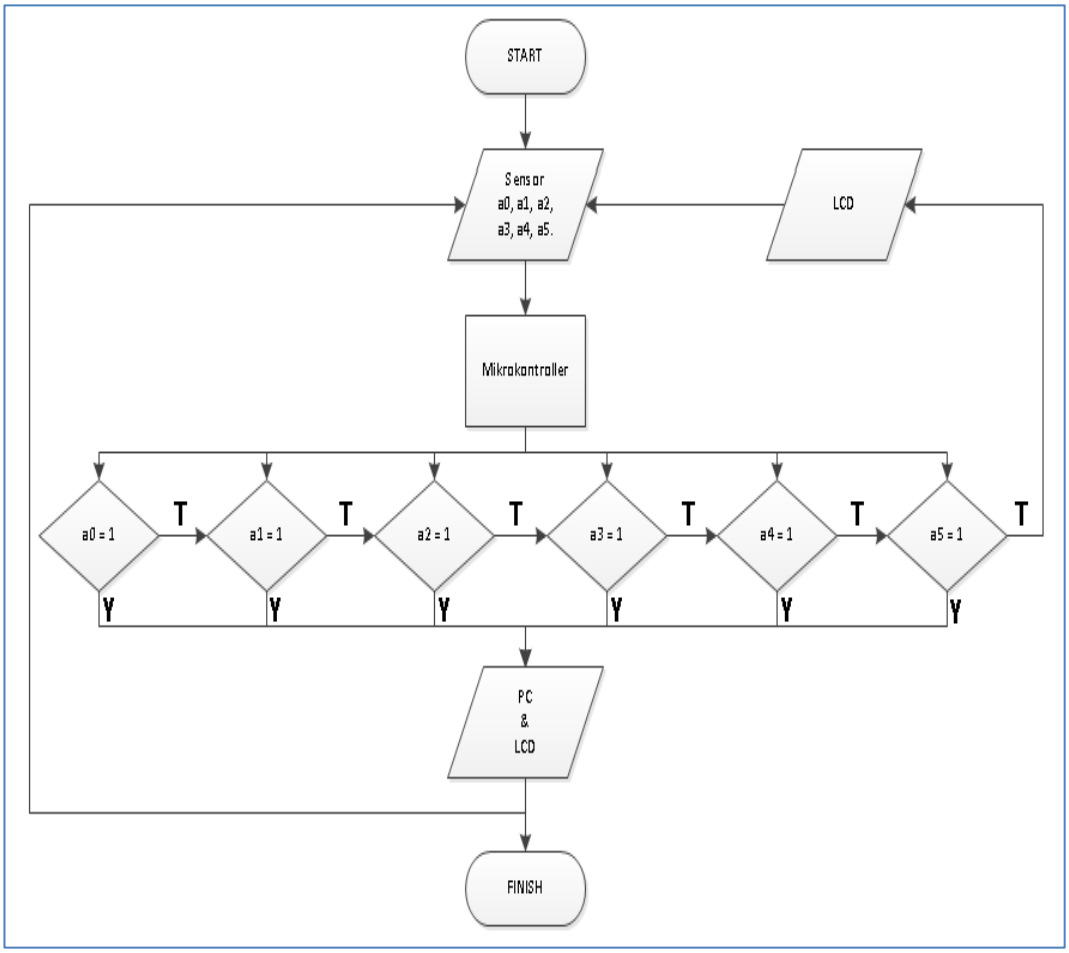


Gambar III.14. Tampilan Program *Visual Basic*

3.7. Perencanaan Program

3.7.1. *Flowchart* Program

Berikut ini diagram alur program pada alat sensor parkir gedung yang telah dibuat.



Gambar III.15 Diagram Alur Program

Diagram alur atau biasa disebut dengan *flowchart* merupakan penggambaran secara grafik dari langkah-langkah dan urutan-urutan prosedur dari suatu program. Diagram alur diatas menggunakan dua buah program utama, yaitu pemrograman pada mikrokontroler ATMega16 dan pemrograman pada PC.

Start pada flowchart menandakan bahwa sistem pada alat telah siap digunakan. Sistem akan memeriksa input dari sensor *infrared* yang disimbolkan

dengan huruf a0-a5. Setelah mendapat input dari sensor *infrared* maka mikrokontroler akan mengolah inputan tersebut sehingga akan tampil pada layar LCD untuk jumlah slot parkir baik yang terisi maupun yang masih kosong. Kemudian mikrokontroller juga akan mengirimkan data input dari sensor *infrared* menuju PC.

Data dikirimkan mikrokontroler ke komputer menggunakan komunikasi serial yang sudah tersedia. Kemudian komputer akan mengolah data inputan tersebut sehingga dapat memonitoring slot parkir yang tersedia. Sementara fungsi buzzer adalah sebagai indikator atau penanda bahwa slot parkir pada gedung sudah penuh terisi.

3.7.2. Konstruksi Sistem (*Coding*)

Berikut ini program yang sudah tertanam pada mikrokontroller ATMega16.

```
/*****************************************************
This program was produced by the
CodeWizardAVR V2.03.4 Standard
Automatic Program Generator
© Copyright 1998-2008 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com

Project :
Version :
Date    : 25/07/2016
Author  :
```

```
Company :
Comments:


Chip type           : ATmega16
Program type        : Application
Clock frequency     : 11,059200 MHz
Memory model        : Small
External RAM size   : 0
Data Stack size     : 256
*****************************************************/
#include <mega16.h>
// Alphanumeric LCD Module functions
#asm
   .equ __lcd_port=0x15 ;PORTC
#endasm
#include <lcd.h>
// Standard Input/Output functions
#include <stdio.h>
#include <delay.h>
// Declare your global variables here
int status1, status2, status3, status4, status5, status6;
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=P State6=P State5=P State4=P State3=P State2=P State1=P
State0=P
PORTA=0xFF;
DDRA=0x00;
// Port B initialization
// Func7=Out Func6=Out Func5=Out Func4=Out Func3=In Func2=In
Func1=In Func0=In
// State7=1 State6=1 State5=1 State4=1 State3=T State2=T State1=T
State0=T
PORTB=0xF0;
DDRB=0xF0;
// 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=Out Func5=Out Func4=Out Func3=In Func2=In
Func1=In Func0=In
// State7=0 State6=0 State5=0 State4=0 State3=T State2=T State1=T
State0=T
PORTD=0x00;
DDRD=0xF0;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=FFh
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;
// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer 1 Stopped
// Mode: Normal top=FFFFh
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
```

```
// Timer 1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
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: Timer 2 Stopped
// Mode: Normal top=FFh
// OC2 output: Disconnected
ASSR=0x00;
TCCR2=0x00;
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
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART Receiver: On
// USART Transmitter: On
// USART Mode: Asynchronous
// USART Baud Rate: 9600
UCSRA=0x00;
UCSRB=0x18;
UCSRC=0x86;
UBRRH=0x00;
UBRRL=0x47;
// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
SFIOR=0x00;
```

```
// LCD module initialization
lcd_init(16);
lcd_putsf("INITIALIZING...");
lcd_gotoxy(0,1);
lcd_putsf(".............");
delay_ms(5000);
lcd_clear();
lcd_gotoxy(2,0);
lcd_putsf("1");
lcd_gotoxy(4,0);
lcd_putsf("2");
lcd_gotoxy(6,0);
lcd_putsf("3");
lcd_gotoxy(8,0);
lcd_putsf("4");
lcd_gotoxy(10,0);
lcd_putsf("5");
lcd_gotoxy(12,0);
lcd_putsf("6");
status1=0;
status2=0;
status3=0;
status4=0;
```

```
status5=0;
status6=0;
PORTB.7=0;
while (1)
      {
       if((PINA.0==1)&&(status1==0))
      {
      lcd_gotoxy(2,1);
      lcd_putsf("X");
      printf("11");
      delay_ms(100);
      status1=1;
      PORTB.7=1;
      delay_ms(500);
      PORTB.7=0;
      }
      if((PINA.0==0)&&(status1==1))
      {
      lcd_gotoxy(2,1);
      lcd_putsf("V");
      printf("21");
      delay_ms(100);
      status1=0;
      PORTB.7=1;
```

```
delay_ms(500);
PORTB.7=0;
}
if((PINA.1==1)&&(status2==0))
{
lcd_gotoxy(4,1);
lcd_putsf("X");
printf("12");
delay_ms(100);
status2=1;
PORTB.7=1;
delay_ms(500);
PORTB.7=0;
}
if((PINA.1==0)&&(status2==1))
{
lcd_gotoxy(4,1);
lcd_putsf("V");
printf("22");
delay_ms(100);
status2=0;
PORTB.7=1;
delay_ms(500);
PORTB.7=0;
```

```
}
if((PINA.2==1)&&(status3==0))
{
lcd_gotoxy(6,1);
lcd_putsf("X");
printf("13");
delay_ms(100);
status3=1;
PORTB.7=1;
delay_ms(500);
PORTB.7=0;
}
if((PINA.2==0)&&(status3==1))
{
lcd_gotoxy(6,1);
lcd_putsf("V");
printf("23");
delay_ms(100);
status3=0;
PORTB.7=1;
delay_ms(500);
PORTB.7=0;
}
```

```
if((PINA.3==1)&&(status4==0))
{
lcd_gotoxy(8,1);
lcd_putsf("X");
printf("14");
delay_ms(100);
status4=1;
PORTB.7=1;
delay_ms(500);
PORTB.7=0;
}
if((PINA.3==0)&&(status4==1))
{
lcd_gotoxy(8,1);
lcd_putsf("V");
printf("24");
delay_ms(100);
status4=0;
PORTB.7=1;
delay_ms(500);
PORTB.7=0;
}

if((PINA.4==1)&&(status5==0))
```

```
{
lcd_gotoxy(10,1);
lcd_putsf("X");
printf("15");
delay_ms(100);
status5=1;
PORTB.7=1;
delay_ms(500);
PORTB.7=0;
}
if((PINA.4==0)&&(status5==1))
{
lcd_gotoxy(10,1);
lcd_putsf("V");
printf("25");
delay_ms(100);
status5=0;
PORTB.7=1;
delay_ms(500);
PORTB.7=0;
}
if((PINA.5==1)&&(status6==0))
{
lcd_gotoxy(12,1);
```

```
            lcd_putsf("X");
            printf("16");
            delay_ms(100);
            status6=1;
            PORTB.7=1;
            delay_ms(500);
            PORTB.7=0;
            }
            if((PINA.5==0)&&(status6==1))
            {
            lcd_gotoxy(12,1);
            lcd_putsf("V");
            printf("26");
            delay_ms(100);
            status6=0;
            PORTB.7=1;
            delay_ms(500);
            PORTB.7=0;
            }
            // Place your code here
            };
}
```