BAB IV
PENGUJIAN DAN ANALISA

4.1 Analisa dan Pengujian Alat

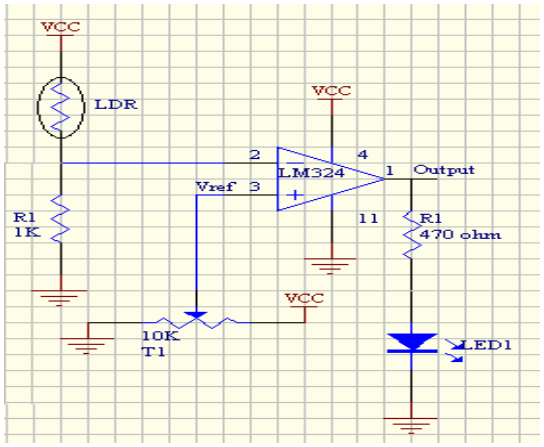
Pada bab pengujian dan analisa dilakukan pada perangkat keras dan perangkat lunak.

4.2 Perangkat Keras

Adapun perangkat keras yang akan dianalisi oleh penulis diantaranya yaitu rangkaian pada sensor LDR dan rangkaian relay.

.

4.2.1 Analisa dan Pengujian Sensor LDR



Gambar 4.1. *Rangkaian sensor LDR*

Berdasarkan hasil pengujian sensor didapat hasil keluaran yaitu ketika sensor terkena cahaya terang maka keluaran dari komparator adalah low sebaliknya ketika sensor tidak terkena cahaya (gelap) maka keluaran high. Untuk lebih jelasnya dapat dilihat pada tabel 4.1.

Tabel 4.1. Logika kondisi sensor

Kondisi Sensor	Logika	Led Indikator
Kena cahaya	Low	Mati
Tidak terkena cahaya	High	Nyala

4.2.2 Analisa Sensor dan Komparator

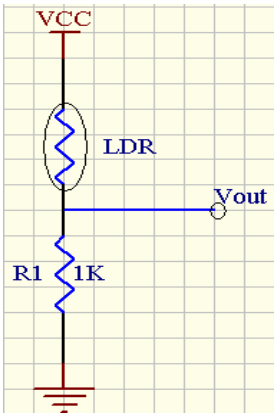
Setelah dilakukan pengukuran pada keluaran komparator LM324, maka didapatkan tegangan keluaran (Vout) pada saat sensor LDR menerima cahaya dan tidak menerima cahaya (gelap). Hasil keluaran tegangan dari komparator dapat dilihat pada tabel 4.2.

Tabel 4.2. *Hasil Pengukuran Sensor*

Sensor LDR Ke	Kondisi Terkena Cahaya			Kondisi Tidak Terkena Cahaya		
	Vin (Vout LDR)	Vout (Komparator)	Logika	Vin (Vout LDR)	Vout (Komparator)	Logika
1	4,92 V	0,24 V	Low	1,70 V	3,13 V	High
2	4,86 V	0,25 V	Low	0,89 V	3,14 V	High
3	3,37 V	0,25 V	Low	0,22 V	3,13 V	High
4	3.72 V	0,25 V	Low	0,07 V	3,13 V	High
5	4,16 V	0,25 V	Low	0,46 V	3,12 V	High
6	4,26 V	0,25 V	Low	0,6 V	3,09 V	High
7	4,12 V	0,22 V	Low	0,13 V	3,07 V	High
8	3,36 V	0,02 V	Low	0,13 V	3,07 V	High
9	3,29 V	0,02 V	Low	0,11 V	3,00 V	High
10	3,56 V	0 V	Low	0,23 V	3,00 V	High
11	4,53 V	0 V	Low	0,45 V	3,00 V	High
12	3,32 V	0 V	Low	0,15 V	3,04 V	High

Keterangan : Vin = tegangan sebelum masuk komparator (Vout dari LDR)
Vout = tegangan setelah masuk komparator

Selain mengukur keluaran tegangan pada komparator, penulis juga melakukan pengukuran terhadap tegangan referensi dan hasil *Vref* (tegangan referensi) sebesar 2 Volt.



Gambar 4.2. *Rangkaian LDR Pembagi Tegangan*

Karena LDR disusun sebagai rangkaian pembagi tegangan, maka untuk mengetahui nilai resistansinya digunakan persamaan 4.1

$$V_{out} = \frac{R_1}{R_1 + R_{LDR}} Vcc \quad \textbf{.....(4.1)}$$

$$V_{OUT}(R_1 + R_{LDR}) = R * Vcc$$

$$R_{LDR} = \frac{R_1 * Vcc - R_1 * Vout}{Vout} \quad \textbf{.....(4.2)}$$

Dengan menggunakan persamaan 4.2 dan data hasil pengukuran (tabel 4.2) diperoleh nilai R_{LDR} pada saat kondisi gelap dan pada saat kondisi terang adalah sebagai berikut:

- R_{LDR} pada saat kondisi nyala lampu (terang), untuk data dari sensor pertama adalah :

$$R_{LDR} = \frac{R_1 * Vcc - R_1 * Vout}{Vout}$$

$$R_{LDR} = \frac{1K\Omega * 5Volt - 1K\Omega * 4{,}92Volt}{4{,}92Volt}$$

$$R_{LDR} = \frac{5K - 4{,}92K}{4{,}92V}$$

$$R_{LDR} = \frac{0{,}08K}{4{,}92V} = 0{,}16\mathrm{K}\Omega$$

Maka R_{LDR} pada saat kondisi terang (nyala lampu) adalah 0,16 KΩ

- R_{LDR} pada saat kondisi mati lampu (gelap), untuk data dari sensor pertama adalah

$$R_{LDR} = \frac{R_1 * Vcc - R_1 * Vout}{Vout}$$

$$R_{LDR} = \frac{1K\Omega * 5Volt - 1K\Omega * 1{,}7Volt}{1{,}7Volt}$$

$$R_{LDR} = \frac{5K - 1{,}7K}{1{,}7V}$$

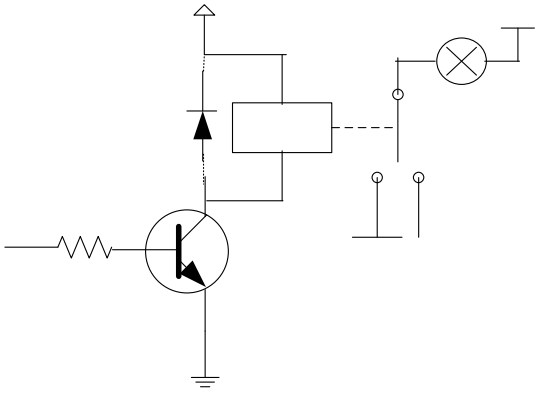
$$R_{LDR} = \frac{3{,}3K}{1{,}7V} = 1{,}94\mathrm{K}\Omega$$

Maka R_{LDR} pada saat kondisi gelap adalah 1,94 KΩ

Dari hasil pengukuran dan perhitungan di atas dapat diperoleh kesimpulan bahwa semakin terang cahaya yang diterima oleh LDR, maka tegangan *output* LDR akan semakin besar dan nilai resistansinya akan semakin kecil, dan sebaliknya jika cahaya yang diterima oleh LDR semakin kecil maka tegangan *output* LDR akan semakin kecil dan nilai resistansinya akan semakin besar.

4.2.3 Analisa Driver Lampu

Driver lampu ini berfungsi untuk mengendalikan beban yang dapat digunakan lampu. Untuk dapat mengontrol rangkaian sakelar 1 sampai 12, mikrokontroler harus mengirimkan data sinyal pulsa ”0” dan ”1”. Jika mikrokontroler memberikan data sinyal pulsa ”0” maka rangkaian sakelar digital berada dalam keadaan tidak aktif, tapi bila ada sinyal pulsa ”1” yang dikirimkam oleh mikrokontroler, maka rangkaian sakelar digital akan aktif. Relai yang digunakan pada rangkaian sakelar ini mempunyai supply tegangan sebesar 5 Volt dc.



Gambar 4.3. *Analisi rangkaian driver lampu*

Untuk mendapatkan nilai arus I_C, dilakukan pengukuran terlebih dahulu terhadap tahanan pada relay atau $R_{(relai)}$. Selanjutnya I_C dapat dicari dengan rumus seperti di bawah ini.

$$I_C = \frac{V_{(relai)}}{R_{(relai)}} = \frac{5V}{100} = 50mA$$

$$I_{B(sat)} = \frac{I_C}{\beta} = \frac{50mA}{100} = 0,5mA$$

Untuk arus basis I_B pada transistor adalah sebagai berikut :

$$I_B = \frac{V_B - VBE}{R_B} = \frac{4,92 - 0,85}{4K7}$$

$$I_B = \frac{4,07}{4K7} = 0,86mA$$

Keterangan : β = 100 ($V_{CE(sat)}$ dari datasheet transistor jenis NPN tipe 2N3904)

Dari perhitungan di atas didapatkan kesimpulan bahwa arus basis lebih besar dari arus basis saturasi ($I_B > I_{B(sat)}$), maka arus I_B akan membuat transistor ada dalam keadaan saturasi sehingga arus akan mengalir menuju relai yang akan menyebabkan swicth tertutup dan lampu akan terhubung tegangan PLN

Resistor pada kaki basis akan membatasi arus yang akan masuk ke transistor. sedangkan Diode 1N4002 berfungsi untuk menahan tegangan balik dari relai dari kondisi aktif ke kondisi tidak aktif. Pada gambar 4.3 terlihat rangkaian lampu

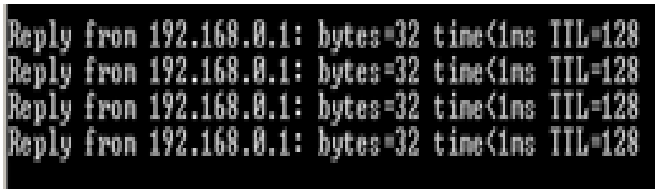
4.3 Analisa Perangkat Lunak

4.3.1 Perangkat Lunak

Adapun perangkat lunak yang akan di analisi oleh penulis yang meliputi pengaksesan komputer, proses komunikasi *client server*, proses pengiriman data dari *client* ke *server* ataupun sebaliknya, dan proses pengiriman data sensor dari *server* ke *client.*

4.3.1.1 Pengaksesan Komputer

Sebelum melakukan pengontrolan lampu operator harus mengetahui jaringan antara komputer dan *server* terhubung atau tidak. Pengontrolan lampu tidak dapat dilakukan jika koneksi jaringan antara *client* dan *server* tidak terhubung. Untuk mengetahui terhubungnya antara komputer *client* dan komputer *server* dapat dilakukan dengan pengaksesan komputer. Karena penulis menggunakan Windows maka pengaksesan komputer dilakukan pada *Command Prompt* dengan mengetik *ping 192.168.0.1* (*IP server*) pada komputer *client.* Jika terdapat status ”replay from 192.168.0.1” pada command prompt maka koneksi antara *client* dan *server* terkoneksi.



Gambar 4.4. *Pengaksesan terhadap server pada Command Prompt*

4.3.1.2 Proses Komunikasi Antara Client dan Server

Proses komunikasi antara *client* dan *server* terjadi pada *Winsock (Windows Socket).* Winsock pada *client* diberinama *wsock_client* dan pada *server* diberinama *wsock_server*. Port yang digunakan untuk berkomunikasi antara *client* dan *server* adalah port 5000. Pertama *wsock_client* akan mengirimkan *request* (permintaan) koneksi ke komputer *server* dengan *syntax ”wsock_client.connect”* . *Server* akan menunggu dan mendengarkan *(listen) request* dari *client* dengan syntax *”wsock_server.listen”*. *Wsock_server* akan menerima request koneksi dari *client* dengan syntax *”wsock_server.Accept requestID* ”. Setelah *server* menerima request dari *client* maka komunikasi antara *client* dan *server* dapat terjadi.

Adapun jenis protokol yang digunakan adalah protokol TCP *(Tranmission Control Protocol)* yaitu sebelum melakukan pertukaran data terlebih dahulu melakukan pembentukan hubungan *(handshake)* antara *client* dan *server*. Pembentukkan hubungan dilakukan dalam pembukaan hubungan TCP antara *client* dan *server*. Berikut ini adalah program untuk koneksi *client* dan *server* sehingga dapat terkoneksi.

Program pada *client* sebagai berikut :

```
Private Sub connect_Click()
wsock_client.RemotePort = 5000
wsock_client.RemoteHost = ip.Text
wsock_client.Close
wsock_client.connect
```

Program pada *server* sebagai berikut :

```
wsock_server.LocalPort = 5000
wsock_server.Listen
lbpesan.Caption = "Server Tidak Bekerja"
status.Caption = "Tidak Terhubung"

Private Sub wsock_server_ConnectionRequest(ByVal requestID As Long)
wsock_server.Close
wsock_server.Accept requestID
status.Caption = "Terhubung dengan " & wsock_server.RemoteHostIP
lbpesan.Caption = wsock_server.RemoteHostIP & "Terhubung"
End Sub
```

4.3.1.3 Proses Pengiriman Data Nyala atau Mati Lampu

Pertama pada program *client* membuat suatu variable, dimana variable tersebut akan diisi oleh sebuah data. Data tersebut akan di kirim ke PC *server* menggunakan syntax wsock_client.SendData. Berikut adalah ini adalah program pada *client* untuk pengiriman data ke *server* dan program pada *server* untuk menerima data dari *client*.

Program pada *client* untuk mengirim data (nyala/mati lampu) ke *server* :

```
Private Sub L1_Click()
If L1.Value = Checked Then
      R1 = "1ON"
   Else
      R1 = "1OFF"
   End If
      wsock_client.SendData R1
End Sub
```

Program pada *server* untuk menerima data (nyala/mati lampu) dari PC *client* :

```
Private Sub wsock_server_DataArrival(ByVal bytesTotal As Long)
wsock_server.GetData terima
If terima = "1ON" Then
   L1.Value = Checked
   ElseIf terima = "1OFF" Then
   L1.Value = Unchecked
```

4.3.1.4 Proses Pengiriman Data Sensor

Sebelum proses pengiriman data sensor, yang dilakukan pertama kali adalah membuat program menghidupkan atau mematikan lampu pada mikrokontroler. Adapun program untuk menghidupkan dan mematikan lampu pada mikrokontroler sebagai berikut :

```
G1L1_on:
        cjne A,#'a',G1L1_off
        clr P0.0
        acall G1_ldr1

G1L1_off:
        cjne A,#'b',G1L2_on
        setb P0.0
        acall delay
        acall G1_ldr1
```

Setelah proses program menghidupkan atau mematikan lampu maka, selanjutnya adalah membuat program untuk pengecekan status lampu (nyala/mati) pada mikrokontroler Berikut ini adalah program untuk pengecekan status lampu pada mikrokontroler.

```
G1_ldr1:
        acall delay
        clr p1.0
        acall delay
        jb p1.0,ldr1
        mov r0,#'A'
        acall kirim
        acall delay
        acall periksa
        sjmp G1_ldr2

ldr1:
        acall delay
        mov r0,#'B'
        acall kirim
        acall delay
        acall periksa
```

Jika kedua program program diatas bisa berjalan, selanjutnya adalah membuat program untuk proses pengiriman status lampu pada PC *server*. Status lampu akan dikirim oleh sensor LDR ke PC *server* melalui RS232 akan diterima oleh program Visual Basic melalui MSComm. Pada program *server* akan menyimpan data sensor tersebut pada buffer setelah itu disimpan ke dalam suatu variabel, kemudian variabel tersebut akan dikirim ke *client* menggunakan syntax *wsock_server.sendData*. Syntax sendData berfungsi untuk mengirim data pada koneksi terbuka yang sedang aktif. Proses pengiriman data sensor kurang lebih satu menit dari mikrokontoler ke PC *server* yang akan dilanjutkan ke PC *client*. Untuk lebih jelas, waktu yang dibutuhkan dalam proses pengiriman data sensor dapat dilihat pada list program mikro.

Misal data yang dikirim oleh mikorokontroler adalah ”A” dan variabel yang digunakan adalah S1, maka ”A” akan di masukkan dalam variabel S1 (S1=”A”). Setelah itu *wsock_server* akan mengirim ke *client* dengan *syntax wsock_server* .SendData S1.

Kemudian pada program *client* data tersebut akan diterima pada *buffer internal winsock* menggunakan syntax getData. Dibawah ini merupakan program *server* untuk mengirim data sensor. Pertama inisialisasi untuk komunikasi port serial kemudian proses menghidupkan dan mematikan lampu dari PC *server* ke mikrokontroler setelah itu proses pengiriman data sensor dari mikrokontroler ke PC *server*.

Program *server* untuk inisilisasi port serial :

```
Private Sub Form_Load()
MSComm1.CommPort = 1
MSComm1.Settings = "9600,N,8,1"
MSComm1.InputLen = 0
MSComm1.PortOpen = True
End Sub
```

Program menghidupkan dan mematikan lampu dari PC *server* ke mikrokontroler.

```
Private Sub L1_Click()
Timer15.Enabled = False
If L1.Value = Checked Then
MSComm1.Output = "a"
Else
MSComm1.Output = "b"
End If
End sub
```

Program pengiriman data sensor dari mikrokontroler ke PC *server*.

```
Private Sub Timer1_Timer()
   If MSComm1.Input = "A" Then
        h1.Visible = True
   ElseIf MSComm1.Input = "B" Then
        h1.Visible = False
   End If
  Timer1.Enabled = False
  Timer15.Enabled = True
End sub
```

Program untuk pengiriman data sensor dari PC *server* ke PC *client* sebagai berikut :

```
Private Sub Timer15_Timer()
   Dim buffer1 As String
   buffer1 = MSComm1.Input
   h = h + 1
If h < 15 Then
   If buffer1 <> "" Then
      Text1.Text = Left$(buffer1, 1)
      If Text1.Text = "A" Then
         h1.Visible = True
         S1 = "A"
         wsock_server.SendData S1
      ElseIf Text1.Text = "B" Then
         h1.Visible = False
         S2 = "B"
         wsock_server.SendData S2
```

Program untuk menerima data sensor pada *client* :

```
        Dim terima As String
        Private Sub wsock_client_DataArrival(ByVal bytesTotal As Long)
        wsock_client.GetData terima

            If terima = "A" Then
               h1.Visible = True
            ElseIf terima = "B" Then
               h1.Visible = False
```

Program pengiriman data sensor dari sensor 1 sampai sensor 12 pada dasarnya sama hanya karakter yang dikirim dari mikrokontroler yang berbeda. Demikian juga sama halnya untuk program pengiriman data sensor dari PC *client* ke PC *server* dan program penerimaan data sensor pada *client.* Program-program diatas merupakan sebagian dari program yang dibuat oleh penulis untuk lebih lengkapnya dapat dilihat pada list program pada lampiran.

4.3.1.5 Komponen Penunjang

Komponen-komponen yang menunjang terbentuknya sistem komunikasi antara *client* dan *server* pada perangkat lunak dengan sistem operasi adalah sebagai berikut. Windows API adalah sebuah antarmuka dari berbagai macam rutin atau fungsi-fungsi yang tersedia dalam *Dynamic Link Libraries* (DLL). Pada sistem operasi Windows terdiri dari ratusan *file DLL*, dan *file DLL* ini merupakan salah satu file yang sangat mendukung sistem komunikasi antara aplikasi dan sistem operasi.

Adapun file-file penunjang tersebut adalah sebagai berikut :

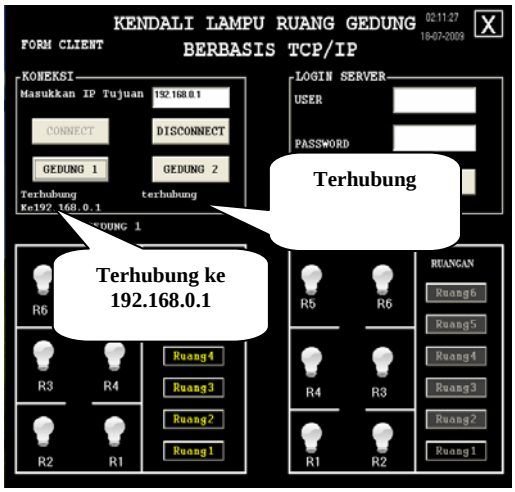
a. MSWINSCK.OCX
b. ws2_32.dll
c. mswsock.dll
d. wsock.dll32

File–file tersebut telah tersedia pada sistem operasi *Microsoft® Windows* sehingga penulis mudah untuk membuat aplikasinya.

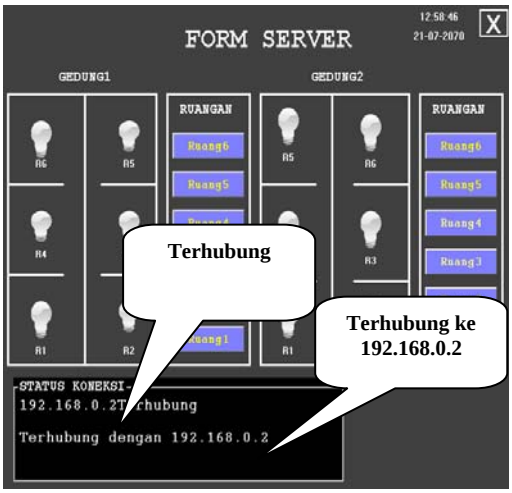
4.4 Pengujian Perangkat Lunak

4.4.1 Pengujian Koneksi Antara Client dan Server

Pertama yang dilakukan adalah *running* kedua program *client* dan *server* kemudian memasukkan IP tujuan (IP server 192.168.0.1) jika terkoneksi maka ada status ”terhubung ke 192.168.0.1” pada *form client* dan ”terhubung ke 192.168.0.2” pada *server*. Pada gambar 4.6 dan gambar 4.7 dapat dilihat koneksi *server* dan *client* sedang terhubung.

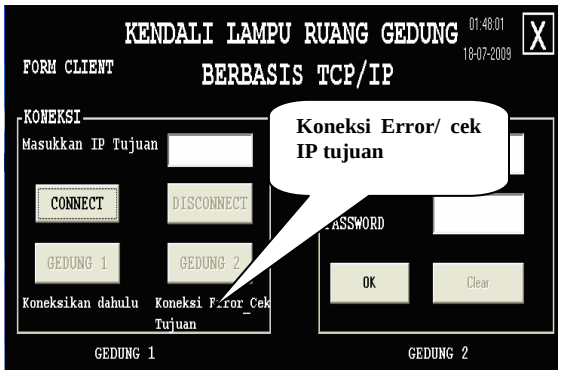


Gambar 4.5. *Tampilan pada client saat terkoneksi ke server*

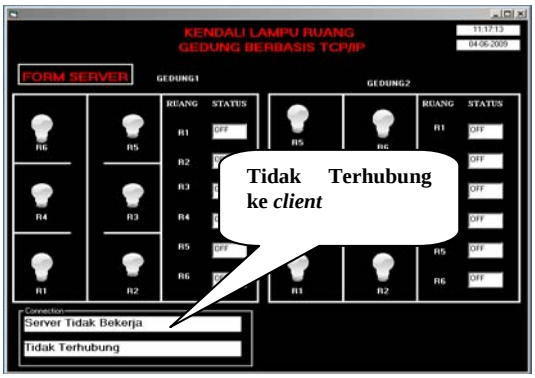


Gambar 4.6. *Tampilan pada server saat terkoneksi ke client*

Apabila salah memasukkan IP atau tidak memasukkan IP tujuan maka koneksi antara *client* dan *server* tidak terhubung dan pengontrolan lampu tidak dapat dilakukan. Sehingga status pada *client* ”Koneksi Error_CekIP” dan ”Tidak terhubung” pada *server* untuk lebih jelas dapat dilihat pada gambar 4.8 dan gambar 4.9.



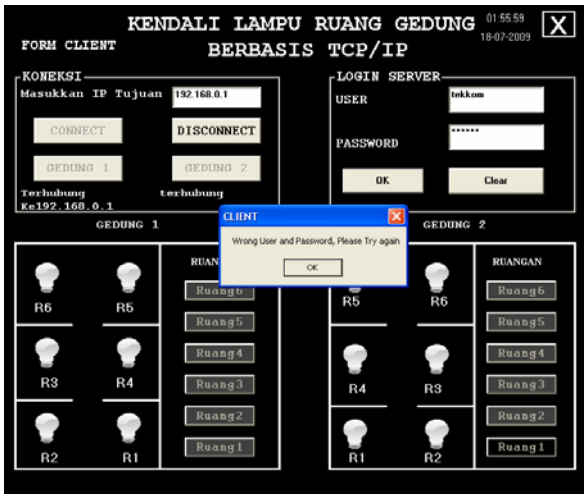
Gambar 4.7. *Tampilan pada client saat tidak terkoneksi ke server*



Gambar 4.8. *Tampilan pada server saat tidak terkoneksi ke client*

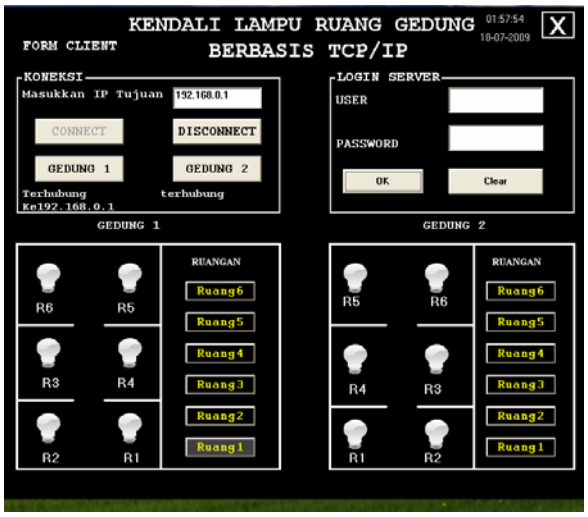
4.4.2 Pengujian Kontrol Lampu

Setelah memasukkan IP tujuan dengan benar dan status antara *client server* saling terhubung kemudian *client* akan memasukkan *user* dan *password server* untuk security pada aplikasi *client*. Sebelum mendapatkan ijin dari *server, client* tidak dapat mengakses aplikasi. Oleh karena itu *client* harus memasukkan *user* dan *password server* dengan benar. Pada gambar 4.10 adalah tampilan *client* salah memasukkan *user* dan *password server*.



Gambar 4.9. *Tampilan client ketika salah memasukkan user dan password server*

Jika *client* memasukkan *user* dan *password server* dengan benar maka aplikasi pada *client* dapat di akses sehingga dapat melakukan pengontrolan lampu (tombol untuk menghidupkan dan mematikan lampu akan di aktikan). Berikut ini adalah tampilan *client* setelah mendapat ijin akses dari *server*.

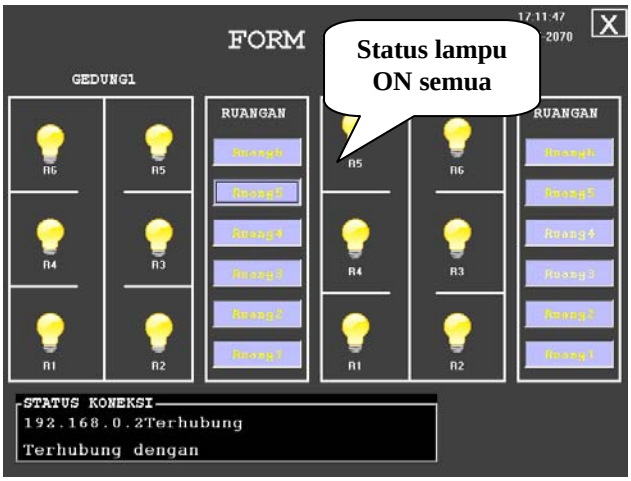


Gambar 4.10. *Tampilan client setelah mendapat ijin akses dari server*

Pada program utama *(form client)* terdapat dua pilihan gedung yaitu gedung 1 atau gedung 2. Dimana kedua gedung terdapat masing-masing 6 ruangan. Jika kita memilih gedung 1 maka tombol gedung 1 di klik secara langsung tombol pada gedung 1 akan aktif dan tombol gedung 2 akan tidak aktif demikian sebaliknya. Cara pengujian kontrol lampu, tombol gedung dan ruangan di pilih (klik) semua maka status lampu akan ”ON” semua.

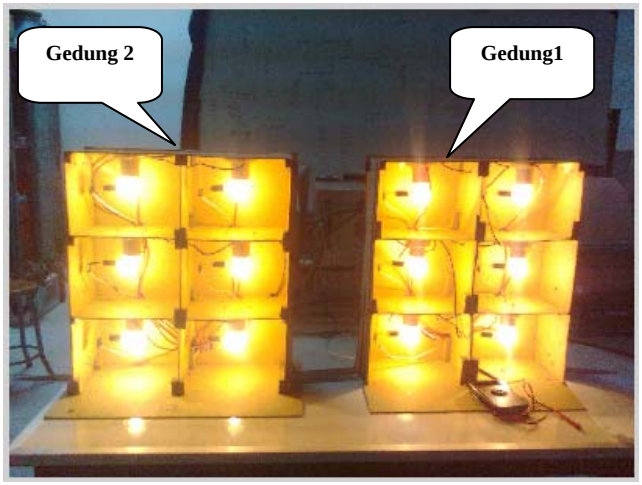


Gambar 4.11. *Tampilan pada client pada saat lampu ON semua*



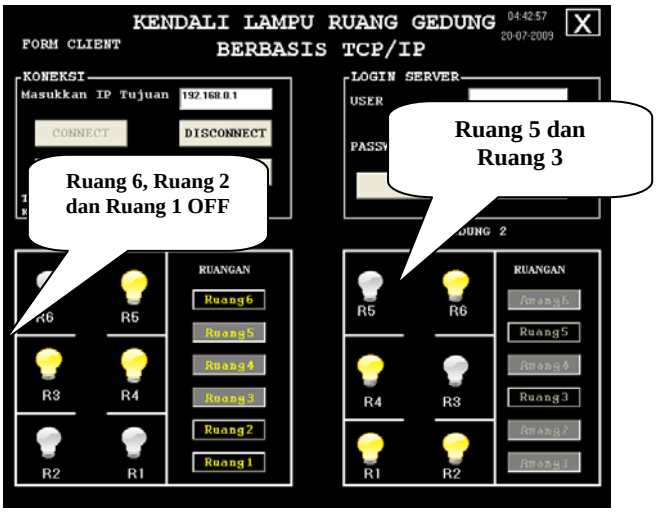
Gambar 4.12. *Tampilan pada server pada saat lampu ON semua*

Dibawah ini merupakan pengujian pada objek dengan kondisi lampu gedung 1 dan gedung 2 ”ON” dengan tampilan sebagai berikut:

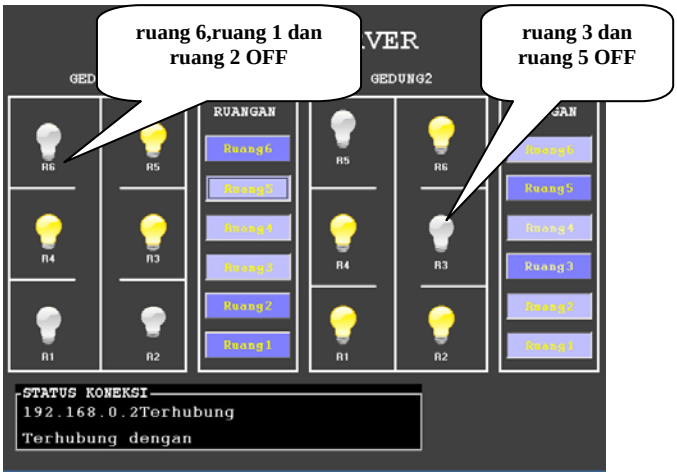


Gambar 4.13. *Tampilan gedung 1 dan gedung 2 ON semua pada objek*

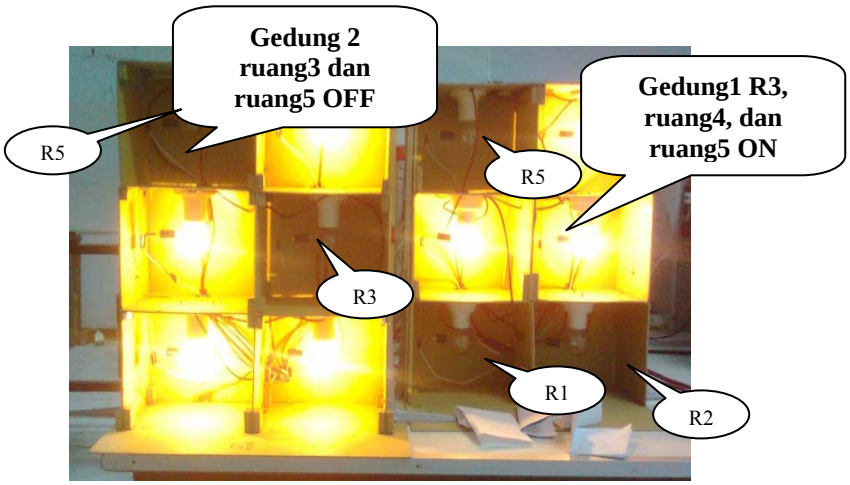
Dibawah ini merupakan pengujian pengontrolan lampu secara acak pada gedung 1 dan gedung 2 dengan tampilan sebagai berikut.



Gambar 4.14. *Tampilan kontrol lampu gedung 1 dan 2 secara acak pada client*



Gambar 4.15. *Tampilan kontrol lampu gedung 1 dan 2 secara acak pada server*

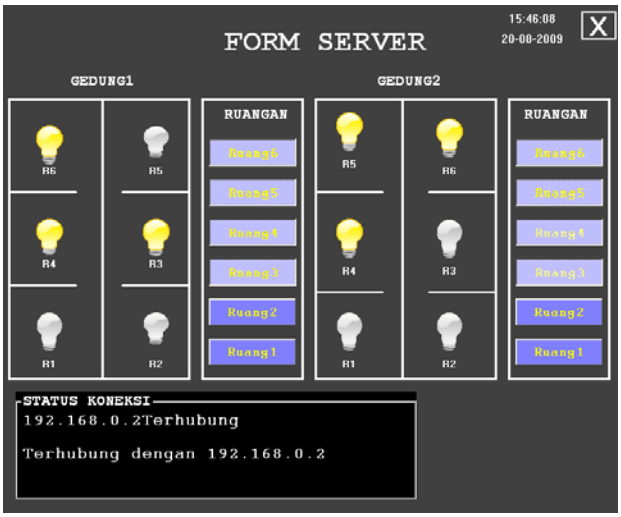


Gambar 4.16. *Tampilan kontrol lampu pada objek secara acak*

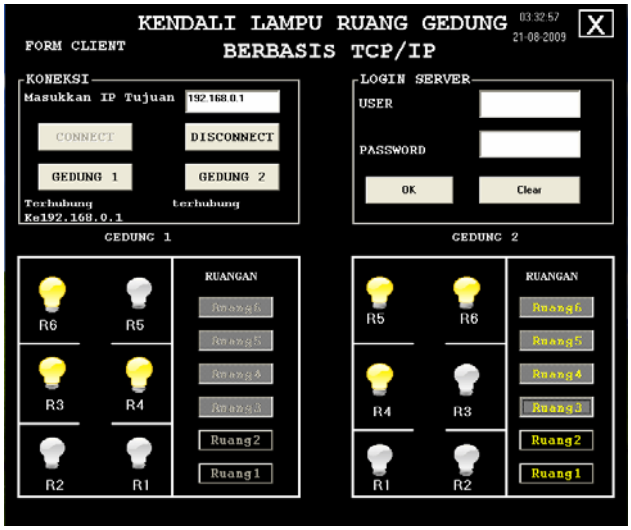
4.4.3 Pengujian Lampu Rusak

Pada pengujian untuk kasus lampu tidak berfungsi atau rusak, lampu yang dinyalakan adalah lampu ruang 3, ruang 4, ruang 5 dan ruang 6 pada gedung 1, serta ruang 3, ruang 4, ruang 5 dan ruang 6 pada gedung 2. Lampu pada ruang 5 pada gedung 1 dan lampu diruang 3 pada gedung 2 dikondisikan tidak berfungsi atau rusak, dengan cara menutup sensor LDR dengan penghalang berwarna hitam. Sensor akan membaca status lampu tersebut kemudian dikirim ke mikrokontroler dan akan diterima oleh *server* untuk dikirim kembali ke *client*.

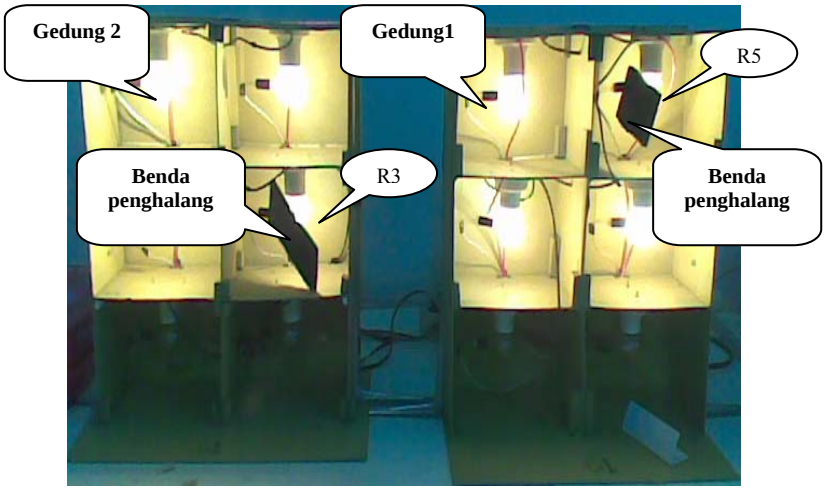
Berikut adalah tampilan dari penjelasan diatas:



Gambar 4.17. *Tampilan pada client*



Gambar 4.18. *Tampilan pada server*



Gambar 4.19. *Tampilan pada objek*