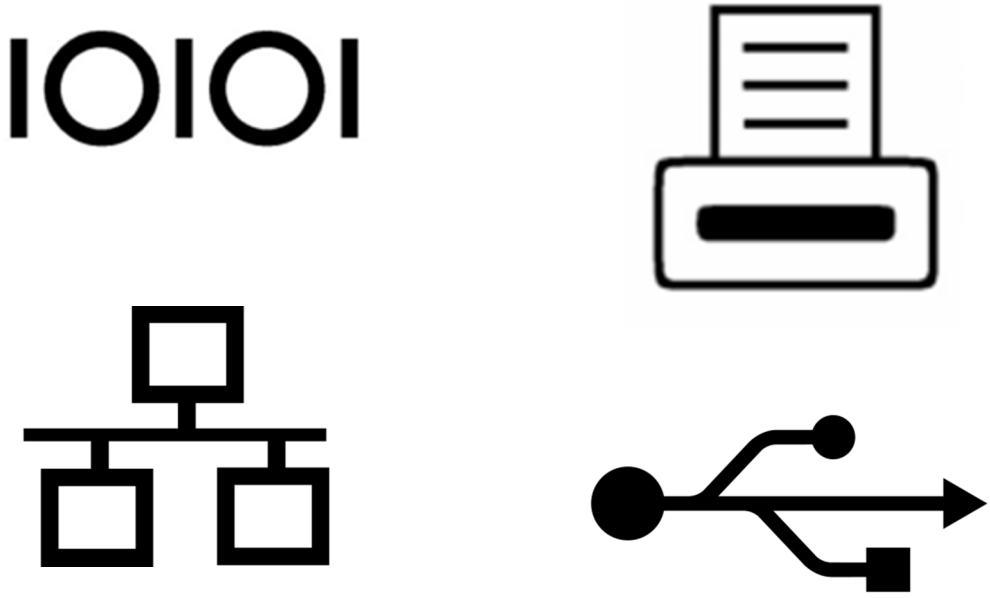
# MODUL PRAKTIKUM
# PENGANTARMUKAAN PERIPHERAL KOMPUTER**
## (INTERFACING)



NPM / NAMA : _____ /_____

KELAS / SHIFT : _____ /_____

PJ SHIFT : _____

Laboratorium Menengah Elektronika dan Komputer
Universitas Gunadarma

# TATA TERTIB PRAKTIKUM INTERFACING

1) Keterlambatan maksimal 15 menit untuk waktu tes pendahuluan. Tetapi masih bisa mengikuti praktikum. (setelah 30 menit tidak bisa masuk dengan alasan apapun).
2) Wajib membawa perlengkapan praktikum seperti : kartu praktikum, modul, tugas-tugas, apabila tidak membawa, maka tidak dapat mengikuti praktikum.
3) Wajib handphone disilent & tidak digunakan pada saat praktikum berlangsung.
4) Selama praktikum berlangsung wajib untuk berpakaian rapih & dimasukkan
   a) Pria : Kemeja, Celana Bahan / Jeans, Memakai Kaos Kaki, Sepatu Tertutup
   b) Wanita : Kemeja, Jeans (Celana, Rok) Bahan / Levis, Memakai Kaos Kaki Sepatu Tertutup.
5) Tidak boleh memakai aksesoris **kecuali** jam tangan.
6) Untuk praktikan yang berambut panjang harus dirapihkan rambutnya diikat.
7) Sepatu dipakai dengan benar & tidak diinjak bagian belakangnya.
8) Tidak boleh memakai Celana Cargo / banyak kantong.
9) Tidak boleh memakai Kemeja Distro / yang ada gambar-gambarnya.
10) Tidak boleh merokok, buang sampah sembarangan di lingkungan lab ( Gedung 1, Depok / Kalimalang) & tidak boleh menggunakan toilet dosen.

Depok / Kalimalang, _____

**PJ PRAKTIKUM INTERFACING**

# DAFTAR ISI

Percobaan

1

# Pengenalan Borland Delphi 7 dan Konsep Dasar Interfacing

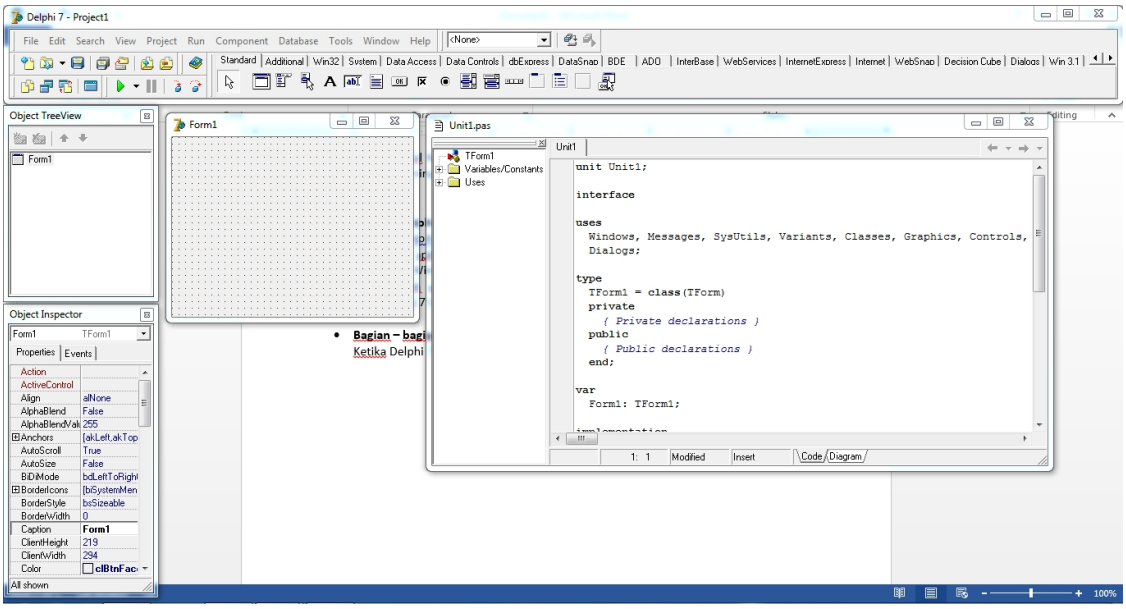
**Tujuan :**
1. Memahami bagian-bagian dan fungsi dari IDE Borland Delphi 7.
2. Memahami stuktur bahasa pemrograman Pascal.
3. Memahami konsep dasar Interfacing menggunakan port Paralel.

**Alat :**
1. PC.
2. Delphi 7.
3. Kabel Paralel Male – Female.
4. Modul Running Led.

**Dasar Teori :**

- **Borland Delphi 7**
  Borland Delphi 7 adalah sebuah perangkat lunak (software) yang digunakan untuk membuat aplikasi berbasis antarmuka grafis (GUI) di lingkungan sistem operasi Microsoft Windows. Delphi dibuat oleh perusahaan Borland Software Corporation. Delphi telah ada sejak 1993 dan versi yang digunakan dalam praktikum ini adalah Delphi versi 7.0 . Bahasa pemograman yang digunakan Delphi adalah bahasa Pascal.

- **Bagian – bagian IDE Delphi 7**
  Ketika Delphi 7 dijalankan, maka tampilannya akan seperti dibawah ini :




*Gambar 1.1 Tampilan IDE pada Delphi 7*

Bagian – bagian IDE pada Borland Delphi 7 :

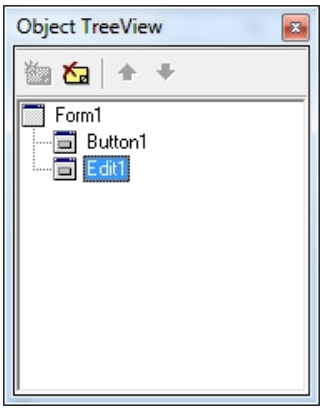
**Title Bar** merupakan suatu area bingkai diatas jendela (windows) utama akan tertulis nama project yang sedang aktif.

**Menu Bar** adalah panel berisi berbagai menu utama yang diikuti *drop-down* menu untuk berbagai keperluan.

**Toolbar** fungsinya mirip dengan Menu bar, namun berupa tombol yang digunakan untuk akses cepat.


*Gambar 1.2 Title Bar, Menu bar, dan Toolbar*

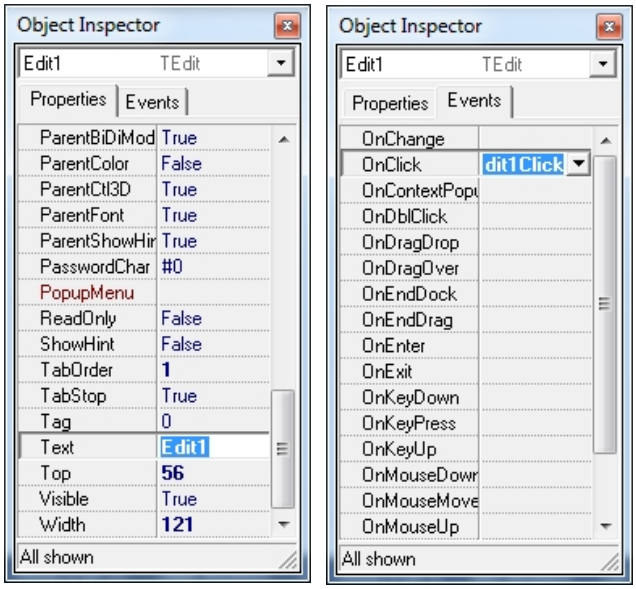
**Object TreeView** adalah jendela untuk menampilkan semua komponen dan *object* yang ada pada Form, tersusun dalam bentuk struktur pohon yang menggambarkan hubungan antar objek.




*Gambar 1.3 Object TreeView*

**Object Inspector** adalah jendela yang berfungsi untuk mengubah **properti, kejadian** (*event*), dan **metode** dari komponen / form terpilih. Object Inspector terdiri dari dua bagian/ tab yaitu *Properties* dan *Events*.

- **Tab Properties** berfungsi untuk mengatur nilai properti untuk Form maupun komponen yang terpilih.
- **Tab Events** berfungsi untuk menentukan tindakan/proses/metode yang akan diaktifkan jika suatu kejadian (*event*) yang bersesuaian terjadi.

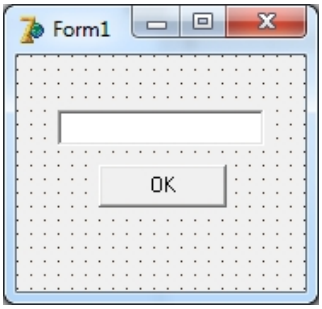



*Gambar 1.4 Object Inspector (Tab Properties dan Events)*

**Component Palette** merupakan area untuk memilih komponen yang akan digunakan oleh aplikasi. Komponen-komponen tersebut dikelompokkan pada suatu paket komponen, untuk memilih komponen tertentu haruslah memilih nama paketnya terlebih dahulu yaitu dengan memilih salah satu tab yang bertuliskan nama paket komponen tersebut.

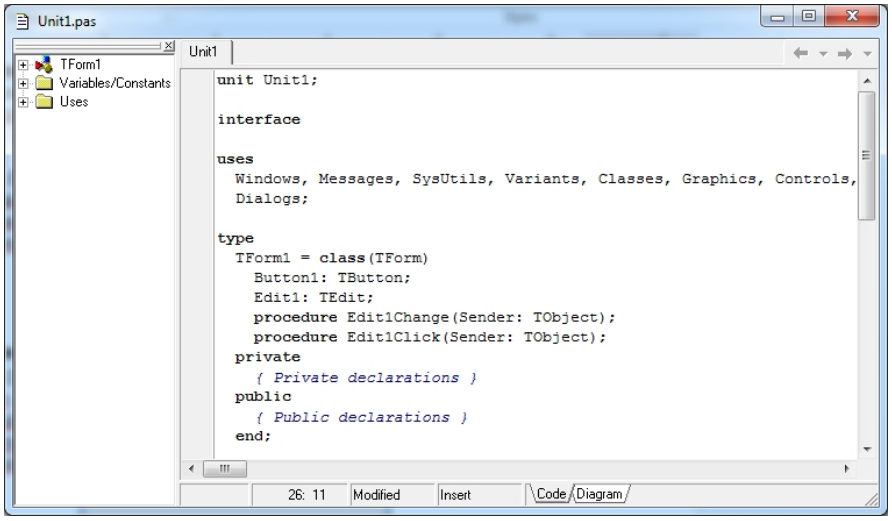

*Gambar 1.5 Component Palette*

**Form Designer** merupakan area untuk menyusun komponen yang akan digunakan sekaligus untuk mendesain tampilan dari aplikasi.




*Gambar 1.6 Form Designer*

**Code Editor** merupakan halaman untuk menyusun perintah-perintah yang sebagian besar dituliskan oleh programmer atau pengembang aplikasi.




*Gambar 1.7 Code Editor*

- **File – File Delphi**

  Sebuah proyek Delphi akan terdiri dari berberapa file. Beberapa file berfungsi untuk menyimpan source code program dan ada file lain yang menyimpan kode binari, gambar dan sebagainya. Karena setiap aplikasi terdiri dari beberapa file. Sangat disarankan agar menyimpan sebuah aplikasi pada satu folder agar ketika akan dilakukan perubahan terhadap aplikasi tersebut di Komputer lain, semua file yang ada tersedia dalam satu buah folder.

  Berbagai jenis file yang dibuat saat membangun aplikasi menggunakan Delphi adalah sebagai berikut:

| Ekstensi File | Keterangan |
|---|---|
| *.dpr | File-file proyek |
| *.dfm | File-file form |
| *.pas | File-file unit |
| *.dpk | File-file paket |
| *.res | File-file Resource |
| *.cfg | File-file konfigurasi proyek |
| *.dof | File-file pilihan proyek |
| *.dcu | File unit yang terkompilasi |
| *.exe | File yang dapat eksekusi |
| *.dsk | Pengaturan desktop |
| *.~* | File-file cadangan (backup)<br>Misal : *.~pas , *.~dpr |

*Tabel 1.1 File – File yang membangun aplikasi yang dibangun oleh Delphi*

- **Source Code pada Delphi**
  Secara umum bentuk struktur unit source code akan terbagi menjadi beberapa blok atau area, sebagai contoh suatu form mempunyai bentuk source code sebagai berikut:

```
unit Unit1;

interface

uses
  Windows, Messages, StdCtrls, Classes, Controls, .... ;

type
  TForm1 = class(TForm)
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

procedure TForm1. ......;
begin

end;

end.
```

  Penjelasannya sebagai berikut:
  - **Unit** merupakan tempat untuk nama unit tersebut yang sama dengan nama file-nya.
  - **Uses** merupakan area untuk menuliskan nama-nama unit lain yang merupakan pustaka yang diperlukan oleh unit ini.
  - **Interface** merupakan area untuk spesifikasi objek-objek (*types/class, variables*) yang ada pada unit dan dapat diakses oleh unit lain.
  - **Type** merupakan area untuk menuliskan definisi kelas objek yang ada pada unit. Kelas ini pun terbagi menjadi dua, yaitu **private** dan **public.**
    - **Private** merupakan area untuk menuliskan nama-nama variable maupun *procedure/function* yang hanya dikenal oleh objeknya sendiri.
    - **Public** merupakan area untuk menuliskan nama-nama variable maupun *procedure/function* yang bersifat public (dapat dikenal dari objek lain).
    - **End;** merupakan tanda akhir dari area klausa *type*.
  - **Const** merupakan area untuk mendeklarasikan konstanta-konstanta.
  - **Var** merupakan area untuk mendeklarasikan variabel-variabel.
  - **Implementation** merupakan area untuk penulisan-penulisan *procedure/function*.

- **Procedure atau function** merupakan subrutin atau kumpulan perintah untuk suatu tujuan tertentu dalam cakupan kecil. *Procedure/function* ini mempunyai area yang terdiri dari:
  - Area nama *procedure/function* beserta parameter-parameternya.
  - **Var** merupakan area deklarasi variabel lokal *procedure/function*.
  - **Begin** merupakan tanda awal proses yang terjadi pada *procedure/function*.
  - **End;** merupakan tanda akhir dari proses.
- **End.** merupakan batas akhir dari unit.

- **Komponen Standard**
  Berikut ini adalah beberapa komponen yang sering digunakan saat pembuatan aplikasi dari Component Pallete Standard:

| No | Icon | Name | Fungsi |
|---|---|---|---|
| 1 |  | Pointer | Mengembalikan fungsi mouse ke defaultnya |
| 2 |  | Frame | Membentuk suatu frame terhadap obyek yang ada didalamnya |
| 3 |  | Main menu | Membuat menu Utama |
| 4 |  | Popup Menus | |
| 5 |  | label | Hanya untuk menampilkan Teks |
| 6 |  | Edit | Untuk menampilkan dan input data (1 baris) |
| 7 |  | Memo | Sama seperti edit tetapi mempunyai kapasitas lebih besar (lebih dari 1 baris) |
| 8 |  | Button | Digunakan untuk melakukan eksekusi terhadap suatu proses |
| 9 |  | Checkbox | Digunakan untuk menentukan pilihan lebih dari satu |
| 10 |  | Radio Button | Digunakan untuk menentukan pilhan, tetapi hanya satu pilhan yang bisa digunakan |
| 11 |  | List Box | Menmpilkan pilihan dalam bentuk list |
| 12 |  | Combo Box | Menampilkan pilihan dalam bentuk popup |
| 13 |  | Scroll Bar | Merupakan icon yang berupa baris status |
| 14 |  | Group Box | Digunakan untuk mengelompokan suatu icon |
| 15 |  | Radio Group | Digunakan untuk mengelompokan pilihan |

*Tabel 1.2 Komponen Standard*

**Percobaan 1A : Membuat Aplikasi Sederhana 1 (Menentukan bilangan Ganjil / Genap).**

a. Design Form


b. Komponen Properties

| Tab | Komponen | Properti | Nilai |
|---|---|---|---|
| | Form1 | Caption | Ganjil/Genap |
| Standard | Button1 | Caption | Proses |
| Standard | Button2 | Caption | Reset |
| Standard | Button3 | Caption | Keluar |
| Standard | Edit1 | Font (klik button •••)<br>Text<br>Width | Font Style: Bold, Size: 12<br>(dikosongkan)<br>50 |
| Standard | Label1 | Font (klik button •••)<br>Caption | Font Style: Bold, Size: 12<br>Masukan Angka: |
| Standard | Label2 | Font (klik button •••)<br>Caption | Font Style: Bold, Size: 12<br>Hasil: |
| Standard | Label3 | Font (klik button •••)<br>Caption | Font Style: Bold,Size: 12<br>Hasil |

c. Source Code
Lengkapi Source Code untuk komponen **Button1**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.Button1Click(Sender: TObject);
var
a : Integer;
begin
a := strtoint(Edit1.Text);
if a mod 2 = 0 then
begin
Label3.Font.Color := clBlue;
Label3.Caption := Edit1.Text + ' bilangan Genap';
end
else
begin
Label3.Font.Color := clRed;
```

```
Label3.Caption := Edit1.Text + ' bilangan Ganjil';
end
end;
```

Lengkapi Source Code untuk komponen **Button2**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.Button2Click(Sender: TObject);
begin
Edit1.Clear;
Label3.Font.Color := clBlack;
Label3.Caption := 'Hasil';
end;
```

Lengkapi Source Code untuk komponen **Button3**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.Button3Click(Sender: TObject);
begin
Application.Terminate;
end;
```

d. Save project, dengan cara klik **File > Save All,** dan save kedalam **satu folder**, disimpan pada **Desktop.**

e. Jalankan program yang telah dibuat dengan menekan tombol **F9** pada Keyboard, atau Klik **Run > Run** pada tool bar.

f. Jelaskan output yang ditampilkan pada program yang telah dibuat:

**Percobaan 1B : Membuat Aplikasi Sederhana 2 (Konversi Sistem Bilangan).**

(buatlah dengan Project baru, dengan cara klik **File > New > Application**)

a. Design Form


**Komponen Led0 – Led7 disusun dari kanan ke kiri.**
**Komponen Label2 – Label9 disusun dari kanan ke kiri.**
**Komponen Digit0 – Digit7 disusun dari kanan ke kiri.**

b. Komponen Properties

| Tab | Komponen | Properti | Nilai |
|---|---|---|---|
| | Form1 | Caption | Konversi Sistem Bilangan |
| Standard | Button1 | Caption | Konversikan |
| Standard | Button2 | Caption | Reset |
| Standard | Label1 | Font (klik button •••)<br>Caption | Font Style: Bold, Size: 10<br>Masukan Angka Desimal (0-255) : |
| Standard | Label2 – 9 (buat 8) | Font (klik button •••)<br>Caption<br>Aligenment | Font Style: Bold, Size: 10<br>D0, D1, D2, ….., D7<br>taCenter |
| Standard | Label10 | Font (klik button •••)<br>Caption | Font Style: Bold, Size: 10<br>Heksadesimal : |
| Standard | Edit1 | Font (klik button •••)<br>Text<br>Width | Font Style: Bold, Size: 10<br>(dikosongkan)<br>50 |
| Standard | Edit2 – 9 (buat 8) | Name<br>Font (klik button •••)<br>Text<br>Width | Digit0, Digit1, Digit2, ….., Digit7<br>Font Style: Bold, Size: 10<br>0<br>30 |
| Standard | Edit10 | Font (klik button •••)<br>Text<br>Width | Font Style: Bold, Size: 10<br>(dikosongkan)<br>50 |
| Additional | Shape1-8 | Name | Led0, Led1, Led2, ….., Led7 |

| | | | |
|---|---|---|---|
| | (buat 8) | Shape<br>Width<br>Height<br>Brush.Color | stCircle<br>30<br>30<br>clGray |

c. Source Code
Deklarasikan procedure **DisplayBin** dan **DisplayHex** secara private, dengan cara menambahkan perintah dibawah ini setelah kata **private { private declarations }** pada **Code Editor** :

```
procedure DisplayBin;
procedure DisplayHex;
```

Deklarasikan variabel global yang dibutuhkan, dengan cara menambahkan peritah dibawah ini setelah kata **var** pada Code Editor :

```
Led: array[0..7] of TShape;
Digit: array[0..7] of TEdit;
desimal: Byte;
```

Source Code untuk prosedur **DisplayBin,** ketikan setelah kata **implementation {$R *.dfm} :**

```
procedure TForm1.DisplayBin;
var
i, val: Integer;
begin
val := 1;
for i := 0 to 7 do
begin
if ((desimal AND val) = val) then
begin
Led[i].Brush.Color := clRed;
Digit[i].Text := '1';
end
else
begin
Led[i].Brush.Color := clGray;
Digit[i].Text := '0';
end;
val := val * 2;
end;
end;
```

Source Code untuk prosedur **DisplayHex,** ketikan setelah prosedur DisplayBin :

```
procedure TForm1.DisplayHex;
var
str : String;
begin
str := Format('%x',[StrToInt(Edit1.Text)]);
```

```
if (StrLen(PChar(str)) < 2) then str := '0' + str;
Edit10.Text := str;
end;
```

Lengkapi Source Code untuk komponen **Form1**, Event **OnCreate** seperti dibawah ini:

```
procedure TForm1.FormCreate(Sender: TObject);
var
i : Integer;
begin
for i := 0 to 7 do
begin
Led[i] := FindComponent(Format('Led%d',[i])) as TShape;
Digit[i] := FindComponent(Format('Digit%d',[i])) as TEdit;
end;
end;
```

Lengkapi Source Code untuk komponen **Button1**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.Button1Click(Sender: TObject);
begin
desimal := StrToInt(Edit1.Text);
DisplayBin();
DisplayHex();
end;
```

Lengkapi Source Code untuk komponen **Button2**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.Button2Click(Sender: TObject);
var
i : Integer;
begin
Edit1.Clear;
Edit10.Clear;
for i := 0 to 7 do
begin
Led[i].Brush.Color := clGray;
Digit[i].Text := '0';
end;
end;
```

d. Save project, dengan cara klik **File > Save All,** dan save kedalam **satu folder**, disimpan pada **Desktop.**

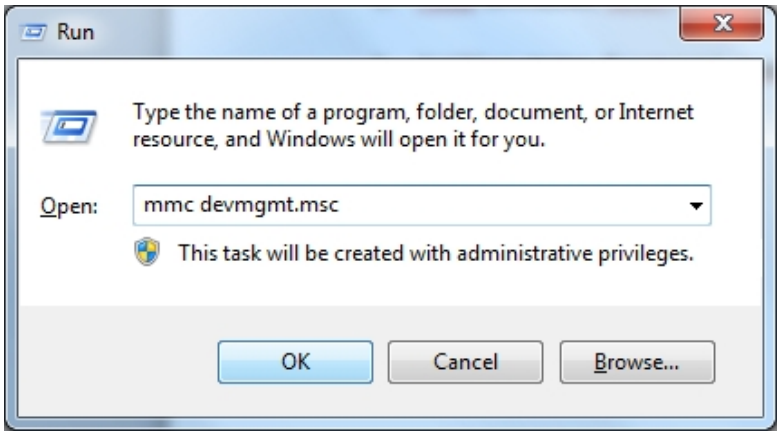
e. Jalankan program yang telah dibuat dengan menekan tombol **F9** pada Keyboard, atau Klik **Run > Run** pada tool bar.

f. Jelaskan output yang ditampilkan pada program yang telah dibuat :

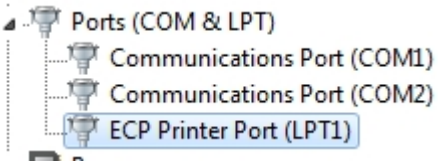
**Percobaan 1C : Mengetahui Alamat dan Mengakses Port Paralel (LPT1)**

a. Pada **Windows**, tekan **Windows Logo + R** secara bersamaan, maka akan muncul **Run.**

b. Ketikkan **mmc devmgmt.msc ,** kemudian klik **OK.**




c. Maka akan terbuka **Device Manager**, kemudian Expand pada bagian **Ports (COM & LPT).** Klik kanan Port **LPT1**, kemudian klik **Properties.**

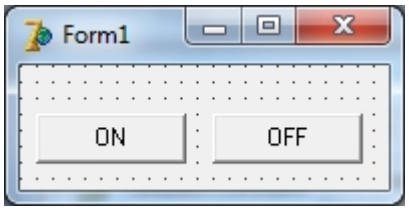



d. Klik tab **Resources,** Kemudian catat I/O Range Address untuk Port LPT1 :

I/O Range Address LPT1 :

e. buka **Delphi 7,** kemudian buatlah program berikut :

Design Form :




Komponen Properties :

| Tab | Komponen | Properti | Nilai |
|---|---|---|---|
| Standard | Button1 | Caption | ON |
| Standard | Button2 | Caption | OFF |

Source Code :
Deklarasikan fungsi **Out32** yang dipanggil dari file library **inpout32.dll**, dengan cara menambahkan program berikut sebelum kata **type** pada Code Editor:

```
function Out32(wAddr:word;bOut:byte):byte; stdcall;
external'inpout32.dll';
```

Lengkapi Source Code untuk komponen **Button1**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.Button1Click(Sender: TObject);
begin
Out32($378,255);
end;
```

Lengkapi Source Code untuk komponen **Button2**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.Button1Click(Sender: TObject);
begin
Out32($378,0);
end;
```

f. Save project, dengan cara klik **File > Save All,** dan save kedalam **satu folder**, disimpan pada **Desktop.** Kemudian Copykan file **inpout32.dll** kedalam file tersebut.

g. Hubungkan Modul Running Led dengan PC menggunakan Kabel Paralel.

h. Jalankan program yang telah dibuat dengan menekan tombol **F9** pada Keyboard, atau Klik **Run > Run** pada tool bar.

i. Amati perubahan yang terjadi pada modul running led ketika **Button1** atau **Button2** di klik. Jelaskan output yang ditampilkan pada modul running led.

**Percobaan 1D : Latihan Mandiri**

Kembangkan kembali Percobaan 1A, namun menambahkan output di modul running led sebagai berikut :

- Apabila angka yang dimasukan **Ganjil**, maka semua Led akan **mati**.
- Apabila angka yang dimasukan **Genap**, maka semua Led akan **menyala**.

Source Code pada **Button1,** event **OnClick :**

| | |
|---|---|
| **Nama Asisten** | |
| **Tgl. Praktikum** | |
| **Paraf Asisten** | |
| | |

Percobaan

2

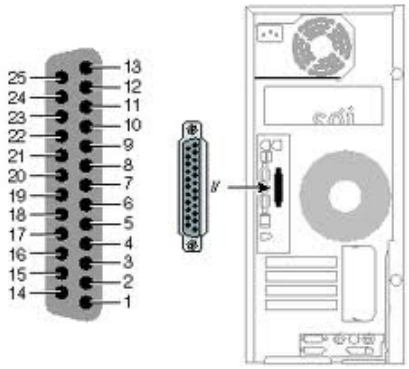
# Komunikasi Paralel pada PC melalui Port LPT1

**Tujuan :**
1. Memahami tentang komunikasi data secara paralel.
2. Memahami cara kerja transmisi data secara paralel melalui port LPT1.
3. Memanfaatkan transmisi data paralel untuk kendali perangkat elektronika sederhana.

**Alat :**
1. PC.
2. Delphi 7.
3. Kabel Paralel Male – Female.
4. Modul Running Led.

**Dasar Teori :**

- **Konektor Port Paralel DB-25**
  Port Paralel merupakan port yang dapat mentransfer data **8 bit sekaligus dalam satu periode clock**. Jalur 8 bit ini sudah diwujudkan dalam 8 buah jalur kabel pada Port DB-25. Penggunaan Port Paralel pada umumnya digunakan untuk menghubungkan Printer ke Komputer. Sehingga sering disebut juga Port Printer.




*Gambar 2.1 Port Paralel pada PC*

IEEE 1284, merupakan standarisasi untuk Port Paralel yang dipublikasikan pada tahun 1994 mendefinisikan 5 (lima) macam mode transfer data Port Paralel:
1. Compatibility Mode
2. Nibble Mode (Mode transfer dengan lebar data 4 bit)
3. Byte Mode (Mode transfer dengan lebar data 8 bit)
4. EPP (Enhance Parallel Port)
5. ECP (Extended Capabilities Parallel Port)

Port Paralel ini terhubungkan dengan dunia luar melalui konektor DB25, yang terbagi atas tiga kelompok register, yaitu:
1. Data Register / Data Port (DP)
2. Control Register / Control Port (CP)
3. Status Register / Status Port (SP)

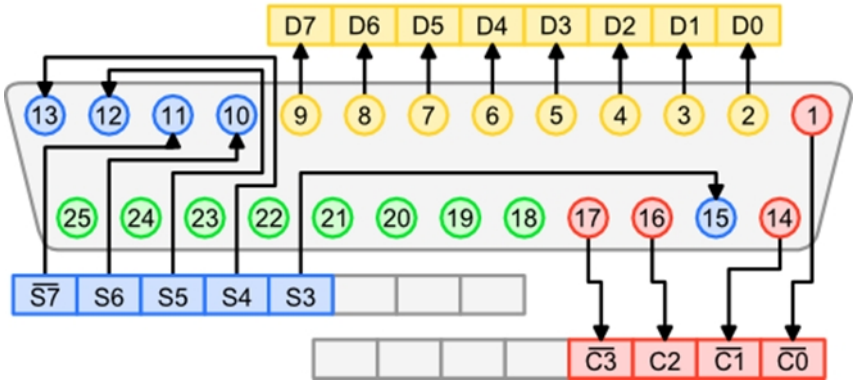
Fungsi dari tiap-tiap register pada Port DB-25 :

**Data Register** digunakan sebagai data bus dan bersifat sebagai port output.

**Control Register** digunakan sebagai control handshaking dengan Printer dan bersifat sebagai port input atau output.

**Status Register** berfungsi untuk mengetahui info status yang dikirim oleh Printer dan bersifat sebagai port input.

Pinout Port DB-25 sebagai berikut :




*Gambar 2.2 Pin out konektor port Paralel DB-25*

| Nomor Pin | Nama Sinyal | Arah | Register – Bit | Invers* |
|---|---|---|---|---|
| 1 | Strobe | In/Out | Control - 0 | Ya |
| 2 | Data 0 | Out | Data - 0 | Tidak |
| 3 | Data 1 | Out | Data - 1 | Tidak |
| 4 | Data 2 | Out | Data - 2 | Tidak |
| 5 | Data 3 | Out | Data - 3 | Tidak |
| 6 | Data 4 | Out | Data - 4 | Tidak |
| 7 | Data 5 | Out | Data - 5 | Tidak |
| 8 | Data 6 | Out | Data - 6 | Tidak |
| 9 | Data 7 | Out | Data - 7 | Tidak |
| 10 | Acknowledge | In | Status - 6 | Tidak |
| 11 | Busy | In | Status - 7 | Ya |
| 12 | Paper-Out | In | Status - 5 | Tidak |
| 13 | Select | In | Status - 4 | Tidak |
| 14 | Linefeed | In/Out | Control - 1 | Ya |

| | | | | |
|---|---|---|---|---|
| 15 | Error | In | Status - 3 | Tidak |
| 16 | Initialize | In/Out | Control - 2 | Tidak |
| 17 | Select-Printer | In/Out | Control - 3 | Ya |
| 18-25 | Ground | - | - | - |

*Tabel 2.1 Pinout Port Paralel DB-25*

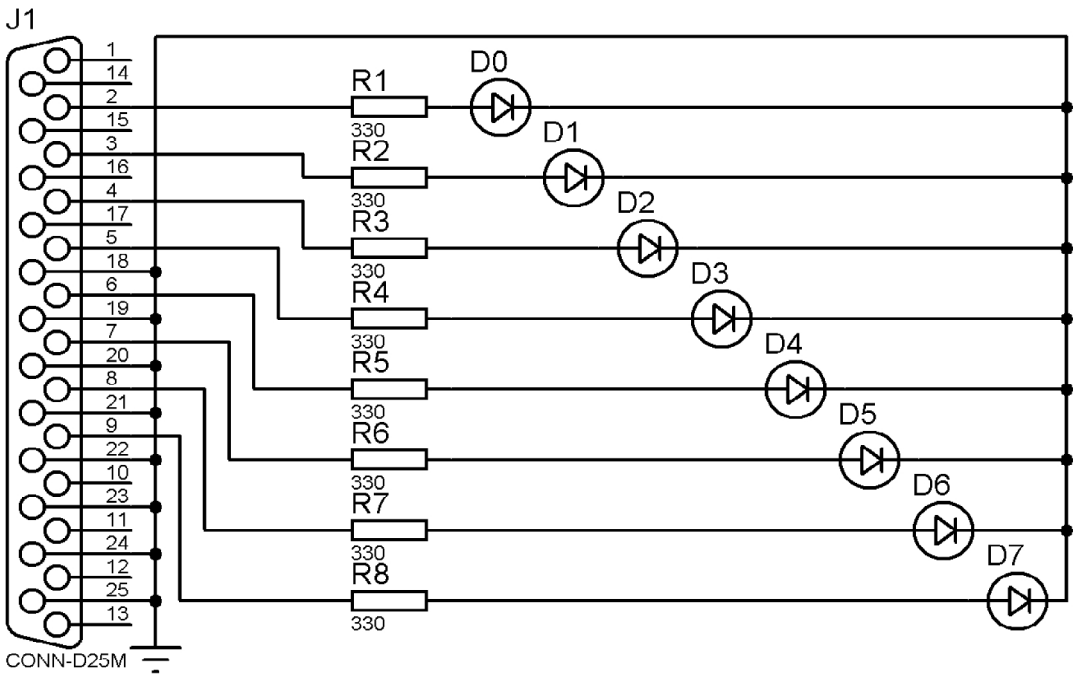
*Invers yang dimaksud adalah aktif low ( bernilai True, apabila logika Low)

Untuk mengakses Port Paralel DB-25, tiap-tiap register pada Port DB-25 sudah dialokasikan port address pada PC.

| Register | LPT1 | MSB | | | | | | | LSB |
|---|---|---|---|---|---|---|---|---|---|
| | | Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Data Register (Base Address + 0) | 378h | Pin | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 |
| Status Register (Base Address + 1) | 379h | | ~11 | 10 | 12 | 13 | 15 | | | |
| Control Register (Base Address + 2) | 37Ah | | | | | | ~17 | 16 | ~14 | ~1 |

*Tabel 2.2 Address Port Paralel DB25*

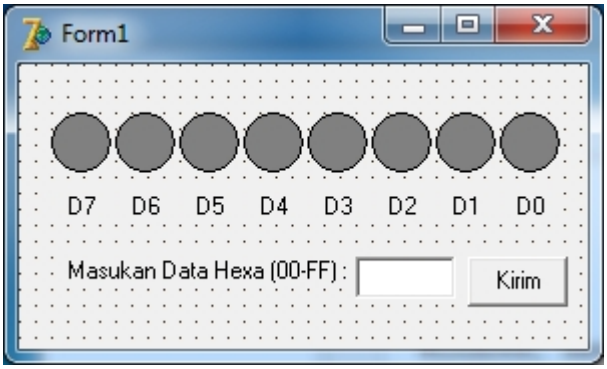
- **Modul Running Led**
  Modul yang digunakan adalah suatu PCB dengan 8 buah LED yang dipasang secara berurutan dengan kabel paralel yang telah dipasang konektor DB-25 diujungnya.




*Gambar 2.3 Schematic Modul Running Led*

**Percobaan 2A : Analisa Transmisi Data melalui port Paralel LPT1**

a. Design Form




**Komponen Led0 – Led7 disusun dari kanan ke kiri.**
**Komponen Label1 – Label8 disusun dari kanan ke kiri.**

b. Komponen Properties

| Tab | Komponen | Properti | Nilai |
|---|---|---|---|
| Standard | Label1-8 (buat 8) | Caption<br>Alignment<br>Height | D0, D1, D2, ….., D7<br>taCenter<br>30 |
| Standard | Label9 | Caption | Masukan Data Hexa (00-FF): |
| Standard | Edit1 | Text<br>Width | (dikosongkan)<br>30 |
| Standard | Button1 | Caption<br>Width | Kirim<br>50 |
| Additional | Shape1-8 (buat 8) | Name<br>Shape<br>Width<br>Height<br>Brush.Color | Led0, Led1, Led2, ….., Led7<br>stCircle<br>30<br>30<br>clGray |

c. Source Code
Deklarasikan fungsi **Out32** yang dipanggil dari file library **inpout32.dll**, dengan cara menambahkan program berikut sebelum kata **type** pada Code Editor:

```
function Out32(wAddr:word;bOut:byte):byte; stdcall;
external'inpout32.dll';
```

Deklarasikan procedure **NyalaLed** secara private, dengan cara menambahkan perintah dibawah ini setelah kata **private { private declarations }** pada Code Editor :

```
procedure NyalaLed;
```

Deklarasikan variabel global yang dibutuhkan, dengan cara menambahkan peritah dibawah ini setelah kata **var** pada Code Editor :

```
data : Byte;
Led: array[0..7] of TShape;
```

Source Code untuk prosedur **NyalaLed,** ketikan setelah kata **implementation {$R *.dfm} :**

```
procedure TForm1.NyalaLed;
var
i, val: Integer;
begin
val := 1;
for i := 0 to 7 do
begin
if ((data AND val) = val) then
begin
Led[i].Brush.Color := clRed;
end
else
begin
Led[i].Brush.Color := clGray;
end;
val := val * 2;
end;
end;
```

Lengkapi Source Code untuk komponen **Form1**, Event **OnCreate** seperti dibawah ini:

```
procedure TForm1.FormCreate(Sender: TObject);
var
i : Integer;
begin
for i := 0 to 7 do
begin
Led[i] := FindComponent(Format('Led%d',[i])) as TShape;
end;
end;
```

Lengkapi Source Code untuk komponen **Button1**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.Button1Click(Sender: TObject);
begin
data := StrToInt('$'+Edit1.Text);
Out32($378,data);
NyalaLed();
end;
```

Lengkapi Source Code untuk komponen **Edit1**, Event **OnKeyPress** seperti dibawah ini:

```
procedure TForm1.Edit1KeyPress(Sender: TObject; var Key: Char);
begin
if Key = #13 then Button1Click(Sender);
end;
```

d. Save project, dengan cara klik **File > Save All,** dan save kedalam **satu folder**, disimpan pada **Desktop.** Kemudian Copykan file **inpout32.dll** kedalam file tersebut.

e. Hubungkan Modul Running Led dengan PC menggunakan Kabel Paralel.

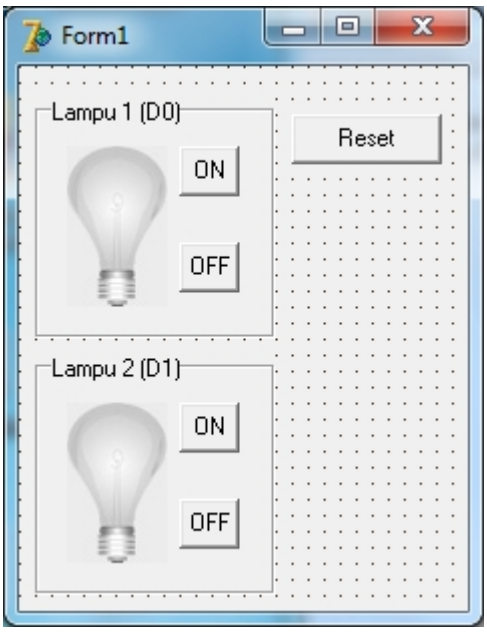
f. Jalankan program yang telah dibuat dengan menekan tombol **F9** pada Keyboard, atau Klik **Run > Run** pada tool bar.

g. Amati perubahan yang terjadi pada modul running led ketika button **Kirim** di Klik atau **Edit1** ditekan Enter. Jelaskan output yang ditampilkan pada modul running led:

**Percobaan 2B : Mengubah data secara Per-bit melalui port LPT1.**

(buatlah dengan Project baru, dengan cara klik **File > New > Application**)

a. Design Form




**Penyusunan Komponen:**
**Image1, btn_on1, btn_off1 berada pada GroupBox dengan caption Lampu 1 (D0).**
**Image2, btn_on2, btn_off2 berada pada GroupBox dengan caption Lampu 2 (D1).**

b. Komponen Properties

| Tab | Komponen | Properti | Nilai |
|---|---|---|---|
| Standard | Button1,3 | Name<br>Caption<br>Width | btn_on1, btn_on2<br>ON<br>30 |
| Standard | Button2,4 | Name<br>Caption<br>Width | btn_off1, btn_off2<br>OFF<br>30 |
| Standard | Button5 | Caption | Reset |
| Standard | Groupbox1-2 (buat 2) | Caption<br>Width<br>Height | Lampu 1 (D0), Lampu 2 (D1)<br>120<br>120 |
| Additional | Image1-2 (buat 2) | Picture (di klik button •••)<br><br>Width<br>Height | Klik **Load..** , pilih gambar lampu mati (**off.jpg)**<br>50<br>80 |

c. Source Code
Deklarasikan fungsi **Out32** dan **Inp32** yang dipanggil dari file library **inpout32.dll**, dengan cara menambahkan program berikut sebelum kata **type** pada Code Editor:

```
function Out32(wAddr:word;bOut:byte):byte; stdcall;
external'inpout32.dll';
function Inp32(wAddr:word):byte; stdcall;
external 'inpout32.dll';
```

Deklarasikan variabel global yang dibutuhkan, dengan cara menambahkan peritah dibawah ini setelah kata **var** pada Code Editor :

```
data, baru: Byte;
```

Lengkapi Source Code untuk komponen **btn_on1**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.btn_on1Click(Sender: TObject);
begin
data := Inp32($378);
baru := data OR $01;
Out32($378,baru);
Image1.Picture.LoadFromFile('on.jpg');
btn_on1.Enabled := False;
btn_off1.Enabled := True;
end;
```

Lengkapi Source Code untuk komponen **btn_off1**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.btn_off1Click(Sender: TObject);
begin
data := Inp32($378);
baru := data AND $FE;
Out32($378,baru);
Image1.Picture.LoadFromFile('off.jpg');
```

```
btn_on1.Enabled := True;
btn_off1.Enabled := False;
end;
```

Lengkapi Source Code untuk komponen **btn_on2**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.btn_on2Click(Sender: TObject);
begin
data := Inp32($378);
baru := data OR $02;
Out32($378,baru);
Image2.Picture.LoadFromFile('on.jpg');
btn_on2.Enabled := False;
btn_off2.Enabled := True;
end;
```

Lengkapi Source Code untuk komponen **btn_off2**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.btn_off2Click(Sender: TObject);
begin
data := Inp32($378);
baru := data AND $FD;
Out32($378,baru);
Image2.Picture.LoadFromFile('off.jpg');
btn_on2.Enabled := True;
btn_off2.Enabled := False;
end;
```

Lengkapi Source Code untuk komponen **Button5**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.Button5Click(Sender: TObject);
begin
Out32($378,0);
Image1.Picture.LoadFromFile('off.jpg');
Image2.Picture.LoadFromFile('off.jpg');
btn_on1.Enabled := True;
btn_off1.Enabled := True;
btn_on2.Enabled := True;
btn_off2.Enabled := True;
end;
```

d. Save project, dengan cara klik **File > Save All,** dan save kedalam **satu folder**, disimpan pada **Desktop.** Kemudian Copykan file **inpout32.dll, on.jpg, off.jpg** kedalam file tersebut.

e. Hubungkan Modul Running Led dengan PC menggunakan Kabel Paralel.

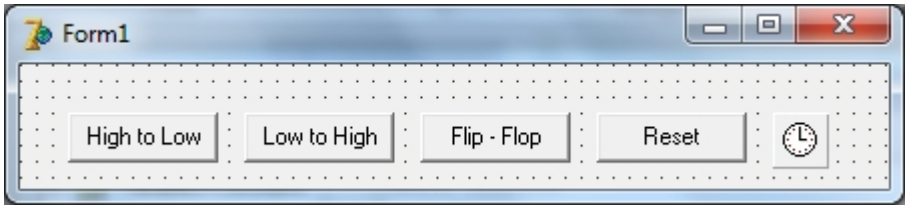
f. Jalankan program yang telah dibuat dengan menekan tombol **F9** pada Keyboard, atau Klik **Run > Run** pada tool bar.

g. Jelaskan output yang ditampilkan pada program dan modul running led :

**Percobaan 2C : Membuat Running Led**

(buatlah dengan Project baru, dengan cara klik **File > New > Application**)

a. Design Form




b. Komponen Properties

| Tab | Komponen | Properti | Nilai |
|---|---|---|---|
| Standard | Button1 | Caption | High to Low |
| Standard | Button2 | Caption | Low to High |
| Standard | Button3 | Caption | Flip - Flop |
| Standard | Button4 | Caption | Reset |
| System | Timer1 | Enabled<br>Interval | False<br>1500 |

c. Source Code
Deklarasikan fungsi **Out32** yang dipanggil dari file library **inpout32.dll**, dengan cara menambahkan program berikut sebelum kata **type** pada Code Editor:

```
function Out32(wAddr:word;bOut:byte):byte; stdcall;
external'inpout32.dll';
```

Source Code untuk prosedur **Delay,** ketikan setelah kata **implementation {$R *.dfm} :**

```
procedure Delay(mSec: Integer);
var StartCount: longint;
begin
StartCount:=GetTickCount;
repeat
Application.ProcessMessages;
until (GetTickCount-StartCount) >= mSec;
end;
```

Lengkapi Source Code untuk komponen **Button1**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.Button1Click(Sender: TObject);
begin
Out32($378,$80);
delay(500);
Out32($378,$40);
delay(500);
Out32($378,$20);
delay(500);
Out32($378,$10);
delay(500);
Out32($378,$08);
delay(500);
Out32($378,$04);
delay(500);
Out32($378,$02);
delay(500);
Out32($378,$01);
end;
```

Lengkapi Source Code untuk komponen **Button2**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.Button2Click(Sender: TObject);
begin
Out32($378,$01);
delay(500);
Out32($378,$02);
delay(500);
Out32($378,$04);
delay(500);
Out32($378,$08);
delay(500);
Out32($378,$10);
delay(500);
Out32($378,$20);
delay(500);
Out32($378,$40);
delay(500);
Out32($378,$80);
end;
```

Lengkapi Source Code untuk komponen **Button3**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.Button3Click(Sender: TObject);
begin
Timer1.Enabled := True;
end;
```

Lengkapi Source Code untuk komponen **Button4**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.Button4Click(Sender: TObject);
```

```
begin
Out32($378,$00);
Timer1.Enabled := False;
end;
```

Lengkapi Source Code untuk komponen **Timer1**, Event **OnTimer** seperti dibawah ini:

```
procedure TForm1.Timer1Timer(Sender: TObject);
begin
Out32($378,$80);
delay(100);
Out32($378,$40);
delay(100);
Out32($378,$20);
delay(100);
Out32($378,$10);
delay(100);
Out32($378,$08);
delay(100);
Out32($378,$04);
delay(100);
Out32($378,$02);
delay(100);
Out32($378,$01);
delay(100);
Out32($378,$02);
delay(100);
Out32($378,$04);
delay(100);
Out32($378,$08);
delay(100);
Out32($378,$10);
delay(100);
Out32($378,$20);
delay(100);
Out32($378,$40);
delay(100);
Out32($378,$80);
delay(100);
end;
```

d. Save project, dengan cara klik **File > Save All,** dan save kedalam **satu folder,** disimpan pada **Desktop.** Kemudian Copykan file **inpout32.dll** kedalam file tersebut.

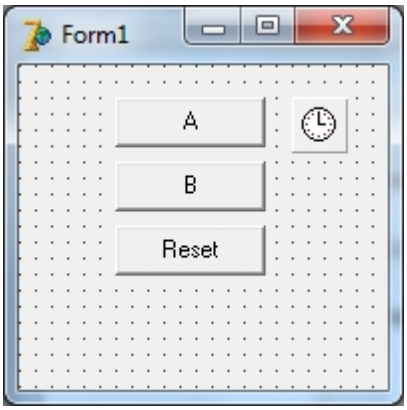
e. Hubungkan Modul Running Led dengan PC menggunakan Kabel Paralel.

f. Jalankan program yang telah dibuat dengan menekan tombol **F9** pada Keyboard, atau Klik **Run > Run** pada tool bar.

g. Jelaskan output yang ditampilkan pada modul running led :

**Percobaan 2D : Latihan Mandiri**

a. Design Form

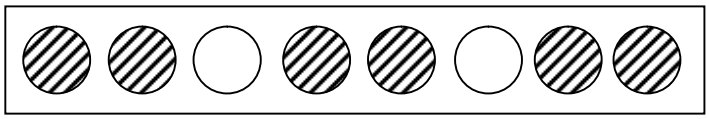



b. Komponen Properties

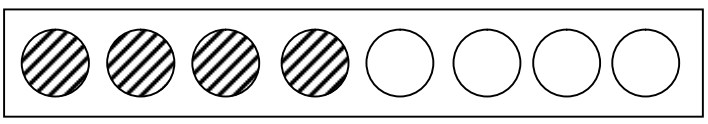
| Tab | Komponen | Properti | Nilai |
|---|---|---|---|
| Standard | Button1 | Caption | A |
| Standard | Button2 | Caption | B |
| Standard | Button3 | Caption | Reset |
| System | Timer1 | Enabled<br>Interval | False<br>1000 |

c. Output yang diinginkan

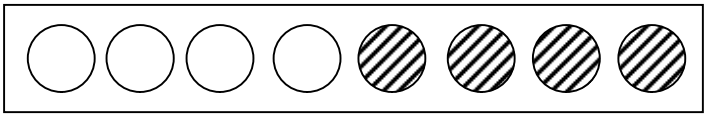
- Apabila diklik tombol **A**, maka Led akan menampilkan :



- Apabila diklik tombol **B**, maka Led akan menampilkan running led :
  Kondisi 1 :



Kondisi 2 :



Jeda antara Kondisi 1 dengan Kondisi 2 adalah 300 ms, seecara terus-menerus.

 = Menyala  = Mati

- Apabila diklik tombol **Reset**, maka Led akan mati, dan **Timer1** tidak aktif.

d. Source Code:

| | |
|---|---|
| **Nama Asisten** | |
| **Tgl. Praktikum** | |
| **Paraf Asisten** | |
| | |

Percobaan

3

# Komunikasi Serial Asinkron pada PC melalui Port COM1

**Tujuan :**
1. Memahami tentang komunikasi data secara serial dengan standar RS-232.
2. Memahami cara kerja transmisi data secara serial melalui port COM1.
3. Memahami dan memanfaatkan konfigurasi *Null Modem*.

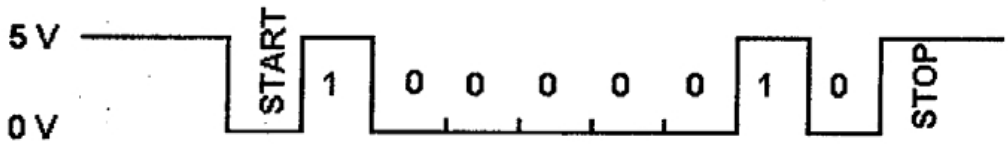
**Alat :**
1. 2 unit PC.
2. Delphi 7.
3. Kabel Serial dengan Konfigurasi *Null Modem*.

**Dasar Teori :**

- **Komunikasi Serial Pada PC**

  Pada IBM PC kompatibel port serialnya termasuk jenis Asinkron. Komunikasi data serial ini dikerjakan oleh UART (*Universal Asynchronous Receiver Transmitter*). IC UART dibuat khusus untuk mengubah data paralel menjadi data serial, dan menerima data serial yang kemudian diubah kembali menjadi data paralel. IC UART 8250 dari Intel merupakan salah satunya.

  Pada UART, kecepatan pengiriman data (*baud rate*) dan *fase clock* pada sisi transmitter dan pada sisi receiver harus sinkron. Untuk itu diperlukan sinkronisasi antara transmitter dan receiver. Hal ini dilakukan oleh bit 'Start' dan bit 'Stop'. Ketika saluran transmisi dalam keadaan *idle*, output UART adalah dalam keadaan logika '1'. Ketika transmitter ingin mengirimkan data, output UART akan diset lebih dulu ke logika ‘0’ untuk waktu satu bit. Sinyal ini pada receiver akan dikenali sebagai sinyal 'Start' yang digunakan untuk mensinkronkan fase clocknya sehingga sinkron dengan fase clock transmitter. selanjutnya, data akan dikirim secara serial dari bit paling rendah (bit 0) sampai bit tertinggi. Selanjutnya, akan dikirim sinyal 'Stop' sebagai akhir dari pengiriman data serial. Cara pemberian kode data yang disalurkan tidak ditetapkan secara pasti.




*Gambar 3.1 Pengiriman huruf ‘A’ tanpa bit paritas*

Kecepatan transmisi (*baud rate*) dapat dipilih bebas dalam rentang tertentu. Baud rate yang umum dipakai adalah 9600, 19200, 115200 (bit/detik). Dalam komunikasi data serial, baudrate dari kedua yang berhubungan harus diatur pada kecepatan yang sama. Selanjutnya, harus ditentukan panjang data (6,7 atau 8 bit), paritas (genap ganjil atau tanpa paritas), dan jumlah bit 'Stop' (1, 1.5, atau 2 bit).

- **RS-232**

  Standar RS-232 ditetapkan oleh *Electronic Industry Association and Telecomunication Industry Association* pada tahun 1962. Nama lengkapnya adalah *EIA/TIA-232 Interface Between Data Terminal Equipment and Data Circuit-Terminating Equipment Employing Serial Binary Data Interchange*. Standar ini menyangkut komunikasi data antara Computer (*Data Terminal Equipment / DTE*) dengan alat-alat pelengkap komputer (*Data Circuit-terminating Equipment / DCE*). Standar RS-232 inilah yang digunakan pada port serial yang ada pada PC.

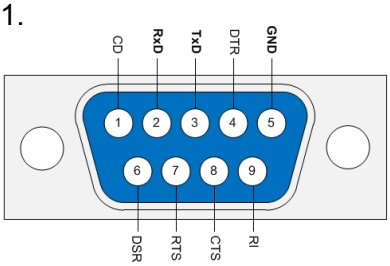
  Standar sinyal serial RS-232 memiliki ketentuan sebagai berikut :
  - Sebuah '**Mark**' (Logika 1) terletak antara -25 V hingga -3 V.
  - Sebuah '**Space**' (Logika 0) terletak antara +3 V hingga +25 V.
  - Tegangan -3 V hingga +3 V adalah *invalid level* (tidak memiliki level logika).
  - Panjang kabel maksimum 15 meter.

  Berikut Contoh pengiriman huruf 'A' pada RS-232 :


*Gambar 3.2 Pengirimah huruf 'A' pada RS-232*

- **Konfigurasi Port Serial**

  Gambar 3.3 dibawah ini adalah gambar konektor port serial DB-9 pada bagian belakang CPU. Pada PC biasanya kita dapat menemukan konektor port serial DB-9 yang biasa dinamai COM1.




*Gambar 3.3 Konektor DB-9 pada belakang PC*

| Nomor Pin | Nama Sinyal | Arah | Keterangan |
|---|---|---|---|
| 1 | DCD | In | Data Carrier Detect |
| 2 | RxD | In | Receive Data |
| 3 | TxD | Out | Transmit Data |
| 4 | DTR | Out | Data Terminal Ready |
| 5 | GND | - | Ground |
| 6 | DSR | In | Data Set Ready |
| 7 | RTS | Out | Request to Send |
| 8 | CTS | In | Clear to Send |
| 9 | RI | In | Ring Indicator |

*Tabel 3.1 Konfigurasi Pin dan nama sinyal konektor DB-9*

Fungsi tiap pin pada port DB-9 :

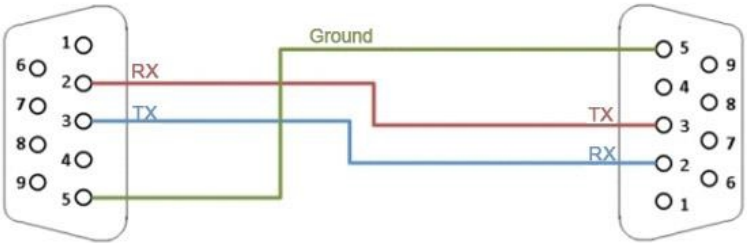
- **Data Carrier Detect (DCD),** Digunakan DCE untuk memberitahukan ke DTE bahwa pada terminal masukan ada data masuk.
- **Received Data (RxD),** untuk penerimaan data.
- **Transmit Data (TxD),** untuk pengiriman data.
- **Data Terminal Ready (DTR),** untuk memberitahu bahwa DTE siap melakukan komunikasi.
- **Ground (GND)**
- **Data Set Ready (DSR),** untuk memberitahu DTE bahwa DCE siap untuk melakukan komunikasi.
- **Request to Send (RTS),** untuk menginformasikan DCE bahwa DTE siap untuk melakukan pertukaran data.
- **Clear to Send (CTS),** untuk memberitahukan bahwa DCE siap melakukan pertukaran data.
- **Ring Indicator (RI),** akan aktif jika modem mendeteksi sinyal dering dari saluran telepon.

- **Konfigurasi kabel *Null Modem***

  Konfigurasi kabel *Null Modem* digunakan untuk menghubungkan 2 perangkat DTE (Misalkan menghubungkan 2 PC). Konfigurasi ini hanya membutuhkan 3 Kabel, yaitu RxD, TxD, dan Ground. Untuk melakukan komunikasi serial antar 2 PC, antara 2 PC tersebut harus memiliki kesamaan parameter.


*Gambar 3.4 2 Penrangkat DTE dihubungkan dengan kabel konfigurasi Null Modem*

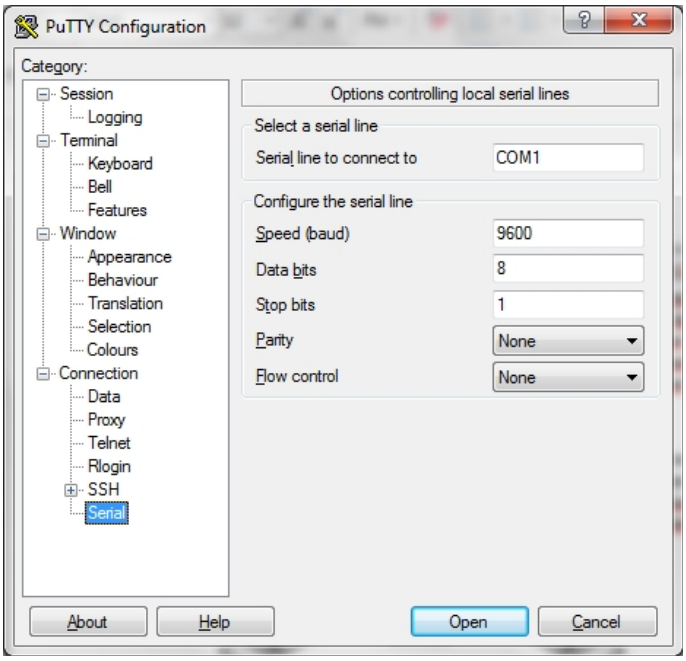
Konfigurasi kabel *Null Modem* dapat dilihat pada gambar dibawah ini :



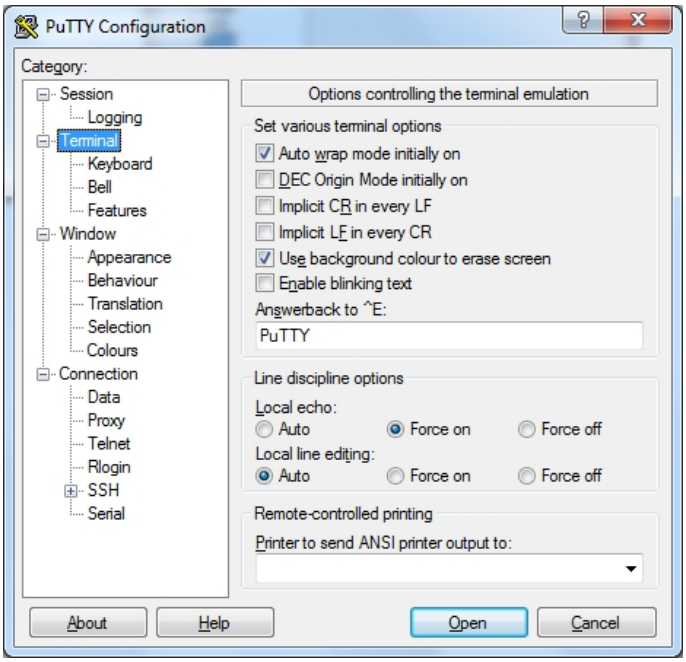

*Gambar 3.5 Konfigurasi Null Modem DB-9 tanpa Handshaking*

**Percobaan 3A : Komunikasi Antar 2 PC menggunakan Software *PuTTY***

a. Hubungkan kedua PC dengan kabel serial konfigurasi *Null Modem* pada port COM1 *(Konektor DB-9 yang ada di depan PC).*
b. Jalankan program PuTTY.
c. Pada **Category**, Klik **Connection – Serial**. Aturlah seperti gambar di bawah ini:




d. Pada **Category,** Klik **Terminal**. Aturlah seperti gambar dibawah ini:




e. Pada **Category,** Klik **Session.** Kemudian pada **Connection type:,** klik **Serial.** Kemudian klik **Open.**

f. Ketikkan beberapa kata pada terminal PuTTY, dan amati hasilnya.

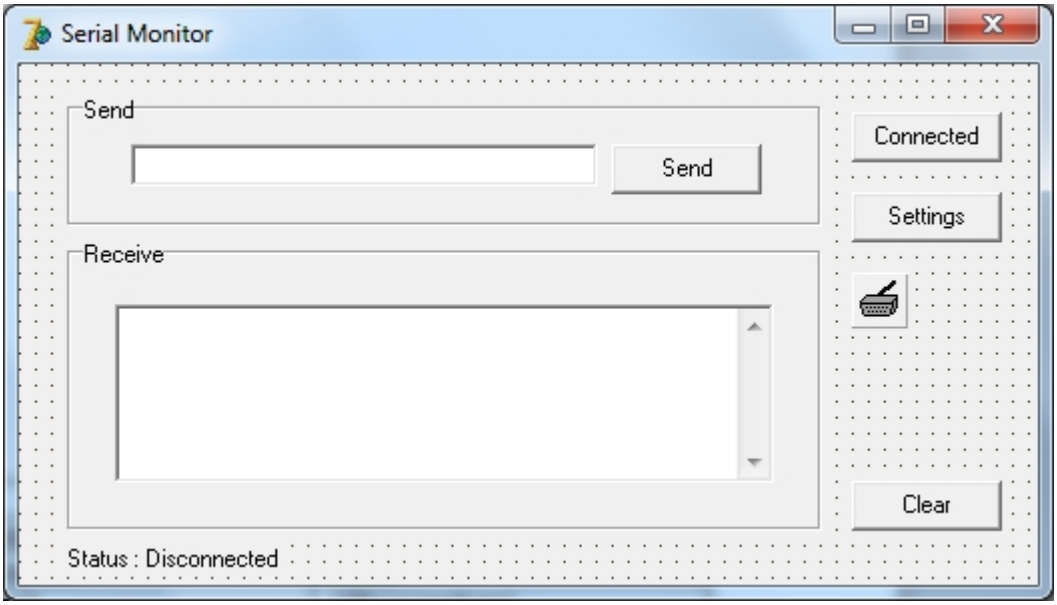
g. Jelaskan output yang dihasilkan :

| |
|---|
| |

h. **Close Program** PuTTY, jika sudah selesai.

**Percobaan 3B : Membuat program *Serial Monitor* dengan Delphi 7**

a. Design Form




b. Komponen Properties

| Tab | Komponen | Properti | Nilai |
|---|---|---|---|
| | Form1 | Caption | Serial Monitor |
| Standard | GroupBox1 | Caption | Send |
| Standard | Edit1 | Text | *(dikosongkan)* |
| Standard | Button1 | Caption | Send |
| Standard | GroupBox2 | Caption | Receive |
| Standard | Memo1 | Lines (di klik button •••)<br>ReadOnly<br>ScrollBars | *(kosongkan string list)*<br>True<br>ssVertical |
| Standard | Button2 | Caption | Connected |
| Standard | Button3 | Caption | Settings |
| CPortLib | ComPort1 | | |
| Standard | Button4 | Caption | Clear |
| Standard | Label1 | Caption | Status : Disconnected |

c. Source Code

Lengkapi Source Code untuk komponen **Button1**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.Button1Click(Sender: TObject);
var
kirim : String;
begin
kirim := Edit1.Text + #13#10;
ComPort1.WriteStr(kirim);
Edit1.Text := #10;
end;
```

Lengkapi Source Code untuk komponen **Button2**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.Button2Click(Sender: TObject);
begin
if Button2.Caption = 'Connected' then
begin
ComPort1.Connected := True;
Button2.Caption := 'Disconnected';
Label1.Caption := 'Status : Connected with ' + Comport1.Port;
end
else
begin
ComPort1.Connected := False;
Button2.Caption := 'Connected';
Label1.Caption := 'Status : Disconnected';
end
end;
```

Lengkapi Source Code untuk komponen **Button3**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.Button3Click(Sender: TObject);
begin
Comport1.ShowSetupDialog;
end;
```

Lengkapi Source Code untuk komponen **Button4**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.Button4Click(Sender: TObject);
begin
Edit1.Clear;
Memo1.Clear;
end;
```

Lengkapi Source Code untuk komponen **ComPort1**, Event **OnRxChar** seperti dibawah ini:

```
procedure TForm1.ComPort1RxChar(Sender: TObject; Count: Integer);
var
terima : String;
```

```
begin
ComPort1.ReadStr(terima, Count);
Memo1.Text := Memo1.Text + terima;
end;
```

Lengkapi Source Code untuk komponen **Edit1**, Event **OnKeyPress** seperti dibawah ini:

```
procedure TForm1.Edit1KeyPress(Sender: TObject; var Key: Char);
begin
if Key = #13 then Button1Click(Sender);
end;
```

d. Save project, dengan cara klik **File > Save All,** dan save kedalam **satu folder,** disimpan pada **Desktop.**

e. Hubungkan kedua PC menggunakan kabel serial konfigurasi *Null Modem*.

f. Jalankan program yang telah dibuat dengan menekan tombol **F9** pada Keyboard, atau Klik **Run > Run** pada tool bar.

g. Klik button **Settings** pada Form.

Atur Settings seperti dibawah ini :

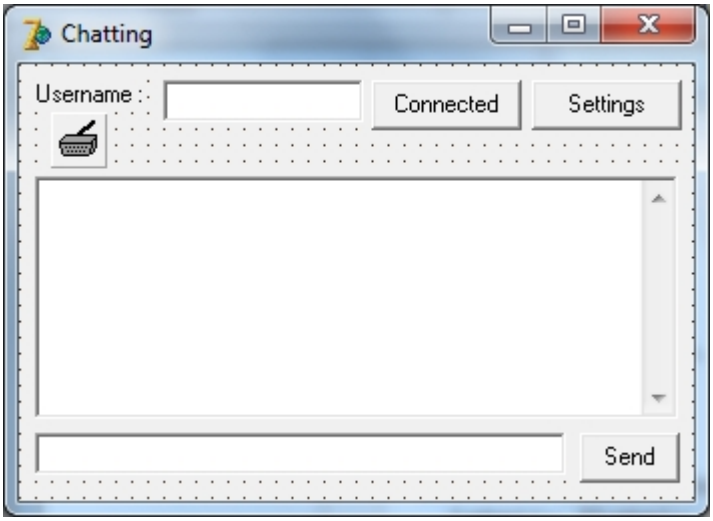
- Port : COM1
- Baud rate : 9600
- Data bits : 8
- Stop bits : 1
- Parity : None
- Flow control : None

Kemudian klik **OK**.

h. Kemudian klik button **Connected**, Isi **Edit1** dengan beberapa karakter, Kemudian amati perubahan yang terjadi pada program yang telah dibuat ketika button **Send** di Klik atau **Edit1** ditekan Enter. Jelaskan output yang dihasilkan:

**Percobaan 3B : Latihan Mandiri**

a. Design Form




b. Komponen Properties

| Tab | Komponen | Properti | Nilai |
|---|---|---|---|
| | Form1 | Caption | Chatting |
| Standard | Label1 | Caption | Username: |
| Standard | Edit1 | Text | *(dikosongkan)* |
| Standard | Button1 | Caption | Connected |
| Standard | Button2 | Caption | Settings |
| Standard | Memo1 | Lines (di klik button •••)<br>ReadOnly<br>ScrollBars | *(kosongkan string list)*<br>True<br>ssVertical |
| Standard | Edit2 | Text | *(dikosongkan)* |
| Standard | Button3 | Caption | Send |
| CPortLib | ComPort1 | | |

c. Source Code *(Lengkapi kode program di bawah ini).*
Lengkapi Source Code untuk komponen **Button1**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.Button1Click(Sender: TObject);
begin
if Button1.Caption = 'Connected' then
begin
_________________________     {menghubungkan akses ke port serial}
Button1.Caption := 'Disconnected';
Edit1.Enabled := False;
end
else
begin
_________________________     {memutuskan akses ke port serial}
Button1.Caption := 'Connected';
Edit1.Enabled := True;
end
end;
```

Lengkapi Source Code untuk komponen **Button2**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.Button2Click(Sender: TObject);
begin
_____________________________   {menampilkan setup untuk port serial}
end;
```

Lengkapi Source Code untuk komponen **Button3**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.Button3Click(Sender: TObject);
var
kirim : ________;          {variabel kirim, dengan tipe data string}
begin
kirim := Edit1.Text + ‘ >> ‘ + Edit2.Text + #13#10;
Memo1.Text := Memo1.Text + kirim;
ComPort1.____________(kirim);  {mengirim nilai variabel kirim ke port serial}
Edit2.Text := #10;
end;
```

Lengkapi Source Code untuk komponen **ComPort1**, Event **OnRxChar** seperti dibawah ini:

```
procedure TForm1.ComPort1RxChar(Sender: TObject; Count: Integer);
var
terima : String;
begin
ComPort1.__________(terima, Count); {menerima data dari port serial}
Memo1.Text := Memo1.Text + terima;
end;
```

Lengkapi Source Code untuk komponen **Edit2**, Event **OnKeyPress** seperti dibawah ini:

```
procedure TForm1.Edit2KeyPress(Sender: TObject; var Key: Char);
begin
if Key = #13 then Button3Click(Sender);
end;
```

d. Save project, dengan cara klik **File > Save All,** dan save kedalam **satu folder**, Disimpan pada **Desktop.**

e. Hubungkan kedua PC menggunakan kabel serial konfigurasi *Null Modem*.

f. Jalankan program yang telah dibuat dengan menekan tombol **F9** pada Keyboard, atau Klik **Run > Run** pada tool bar.

g. Klik button **Settings** pada Form.

Atur Settings seperti dibawah ini :

- Port : COM1
- Baud rate : 9600
- Data bits : 8
- Stop bits : 1
- Parity : None
- Flow control : None

Kemudian klik **OK**.

h. Kemudian klik button **Connected**, Isi **Edit1** untuk Username, dan **Edit2** untuk data yang akan dikirim, Kemudian amati perubahan yang terjadi pada program yang telah dibuat ketika button **Send** di Klik atau **Edit2** ditekan Enter. Jelaskan output yang dihasilkan, dan berikan kesimpulan :

| | |
|---|---|
| **Nama Asisten** | |
| **Tgl. Praktikum** | |
| **Paraf Asisten** | |
| | |

Percobaan

4

# Komunikasi Serial Asinkron PC dengan Mikrokontroler AT89S51

**Tujuan :**
1. Memahami pemrograman komunikasi serial pada Mikrokontroler AT89S51, baik sebagai penerima atau pengirim data.
2. Mampu membuat aplikasi antarmuka sederhana dengan Delphi 7 sebagai penghubung antara PC dengan Mikrokontroler AT89S51 secara Komunikasi Serial.

**Alat :**
1. 1 unit PC.
2. Software Delphi 7, MIDE-51, dan PuTTY.
3. 1 unit MinSys AT89S51 dilengkapi Port Serial.
4. Adaptor dan Kabel Serial.

**Dasar Teori :**

- **Komunikasi Serial Pada Mikrokontroler AT89S51**

  Mikrokontroler AT89S51 memiliki kemampuan untuk berkomunikasi secara serial melalui pin RxD (P3.0) dan TXD (P3.1). Satu hal yang perlu diingat adalah tingkat tegangan komunikasi kedua pin serial menggunakan tingkat tegangan TTL. Pada prinsipnya, komunikasi serial adalah komunikasi dimana transmisi data dilakukan per bit. Interface serial hanya membutuhkan jalur yang sedikit (umumnya hanya 2 jalur), sehingga lebih menghemat pin jika dibandingkan dengan interface paralel. Komunikasi serial ada dua macam, asynchronous serial dan synchronous serial :

  a. **Synchronous serial** adalah komunikasi dimana hanya ada satu pihak pengirim atau penerima) yang menghasilkan clock dan mengirimkan clock tersebut bersama-sama dengan data. Contoh pengunaan synchronous serial terdapat pada transmisi data keyboard.

  b. **Asynchronous serial** adalah komunikasi dimana kedua pihak (pengirim dan penerima) masing-masing menghasilkan clock namun hanya data yang ditransmisikan, tanpa clock. Agar data yang dikirim sama dengan data yang diterima, maka kedua frekuensi clock harus sama dan harus terdapat sinkronisasi. Setelah adanya sinkronisasi, pengirim akan mengirimkan datanya sesuai dengan frekuensi clock pengirim dan penerima akan membaca data sesuai dengan frekuensi clock penerima. Contoh penggunaan asynchronous serial adalah pada Universal Asynchronous Receiver Transmitter (UART) yang digunakan pada serial port (COM) komputer.

  MCS-51 mendukung komunikasi secara asinkron, bahkan tiga dari empat serial mode yang dimiliki MCS-51 kompatibel dengan UART. MCS-51 memiliki 4 mode komunikasi serial. Mode 0 berupa synchronous serial (shift register), sedangkan tiga mode yang lain berupa asynchronous serial (UART). Pada semua mode, pengiriman dilakukan jika ada instruksi yang mengisi nilai register SBUF. Sedangkan pada saat penerimaan, data yang diterima akan disimpan pada register SBUF.

**Mode 0 :**
Mode 0 adalah 8 bit shift register dimana data dikirimkan dan diterima melalui pin RXD sedangkan clock dikirimkan dan diterima melalui pin TXD. Pengiriman data 8 bit dilakukan dengan mengirimkan Least Significant Bit (LSB) terlebih dahulu. Pada mode 0, baud rate yang gunakan adalah sebesar 1/12 dari frekuensi osilator.

**Mode 1 :**
Pada mode 1, jumlah data yang dikirimkan sebanyak 10 bit yang terdiri dari start bit, 8 bit data (LSB terlebih dahulu), dan stop bit. Ada proses penerimaan, nilai stop bit akan dimasukkan ke RB8 secara otomatis.Pada proses pengiriman, stop bit akan diberi nilai '1' secara otomatis. Pada mode 1, baudrate yang digunakan dapat diatur melalui Timer 1.

**Mode 2 :**
Pada mode 2, jumlah data yang dikirimkan sebanyak 11 bit yang terdiri dari start bit, 8 bit data (LSB terlebih dahulu), bit ke-9, dan stop bit. Pada proses pengiriman, nilai bit ke 9 dapat diatur dengan mengisi nilai
TB8. Pada proses penerimaan, bit ke 9 akan dimasukkan ke RB8 secara otomatis. Pada mode 2, baudrate yang dapat digunakan adalah sebesar 1/64 frekuensi osilator atau 1/32 frekuensi osilator jika SMOD bernilai '1'.

**Mode 3 :**
Mode 3 hampir sama dengan mode 2. Perbedaannya terdapat pada baud rate yang digunakan. Jika mode 2 menggunakan baudrate yang pasti, mode 3 menggunakan baud rate yang dihasilkan oleh Timer 1.

- **Pemrograman Mikrokontroler MCS-51 dalam Komunikasi Serial**

Dalam melakukan komunikasi serial, Ada beberapa register yang terlibat untuk melakukan komunikasi serial ini, diantaranya :

**PCON (Power Control)**
Register Power Control beralamat di 87H berguna untuk mengatur kebutuhan daya mikrokontroler. Dengan adanya register pengatur daya ini memungkinkan mikrokontroler ke mode "idle" atau "sleep" yang mana akan lebih menghemat pemakaian daya. Selain itu ada bit-bit pada register PCON ini untuk mengatur Baud rate pada serial port. Bit-bit pada PCON adalah sebagai berikut:

MSB LSB

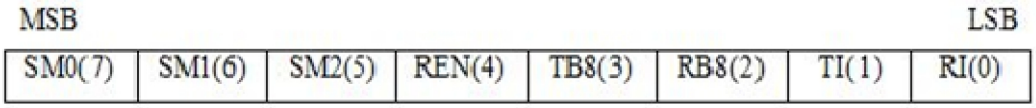
| SMOD (7) | (6) | (5) | (4) | GF1 (3) | GFO (2) | PD (1) | IDL (0) |
|---|---|---|---|---|---|---|---|

Keterangan :

| Simbol | Posisi Bit | Fungsi |
|---|---|---|
| SMOD | PCON.7 | Modifikasi Baudrate. Diset 1 untuk Baudrate 2 kali lipat menggunakan Timer 1 untuk mode 1,2,3. Diset 0 untuk mendapat nilai baudrate dari Timer 1 |

| | | |
|---|---|---|
| - | PCON.6 | Tidak Digunakan |
| | PCON.5 | |
| | PCON.4 | |
| GF1 | PCON.3 | Flag untuk fungsi umum |
| GF0 | PCON.2 | Flag untuk fungsi umum |
| PD | PCON.1 | Bit untuk fungsi operasi Power Down<br>Aktif jika berlogika 1 |
| IDL | PCON.0 | Bit untuk fungsi operasi Idle Mode<br>Aktif jika berlogika 1 |

**SCON (Serial Port Control)**
Register SCON terdiri dari :




Keterangan :

| Simbol | Posisi Bit | Fungsi |
|---|---|---|
| SM0 | SCON.7 | *Serial Mode 0*<br>Port Serial mode 0 |
| SM1 | SCON.6 | *Serial Mode 1*<br>Port Serial mode 1 |
| SM2 | SCON.5 | *Serial Mode 2*<br>Mengaktifkan komunikasi multiprosesor dalam mode 2 dan 3 |
| REN | SCON.4 | *Reception Enable* |
| TB8 | SCON.3 | *Transmit Bit 8*<br>Mengirimkan bit ke-9 yang diaktifkan pada mode 2 dan 3 |
| RB8 | SCON.2 | *Receive Bit 8*<br>Menerima bit ke-9 yang diaktifkan pada mode 2 dan 3 |
| TI | SCON.1 | *Transmit Interrupt flag*<br>Di set oleh hardware untuk mendeteksi *flag* yang menunjukkan suatu byte telah komplit dikirimkan |
| RI | SCON.0 | *Receive Interrupt flag*<br>Di set oleh hardware untuk mendeteksi *flag* yang menunjukkan suatu byte telah komplit diterima. |

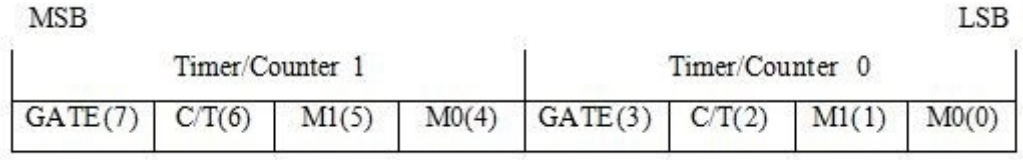
Konfigurasi Mode SM0 dan SM1 :

| SM0 | SM1 | Serial Mode | Keterangan | Baudrate |
|---|---|---|---|---|
| 0 | 0 | 0 | 8 bit Shift register | Osilator/12 |
| 0 | 1 | 1 | 8 bit UART | Set oleh timer 1 |
| 1 | 0 | 2 | 9 bit UART | Osilator/32 |
| 1 | 1 | 3 | 9 bit UART | Set oleh timer 1 |

**SBUF (Serial Data Buffer)**
Register SBUF ini berada pada alamat memory 99H, fungsi dari register ini untuk menyimpan data sementara yang akan dikirimkan dan diterima, setelah serial port di konfigurasi, maka penulisan ke SBUF akan memulai pengiriman secara serial.

**Timer / Counter**
Mikrokontroler AT89S51 dilengkapi dengan dua perangkat Timer/Counter yang masing-masing dinamakan Timer 0 dan Timer 1. Pencacah biner untuk Timer 0 dibentuk dengan register Timer 0 Low byte (TL0) dan register Timer 0 High byte (TH0). Pencacah biner untuk Timer 1 dibentuk dengan register Timer 1 Low byte (TL1) dan register Timer 1 High byte (TH1). Untuk mengatur kerja Timer/Counter dipakai dua register tambahan yang dipakai bersama oleh Timer 0 dan Timer 1. Register tersebut adalah Timer Mode Control (TMOD) dan Timer Control (TCON). Adanya Timer/Counter menambah fungsionalitas dari mikrokontroler ini. Dengan adanya Timer/Counter maka dapat digunakan misalnya untuk menghitung kejadian (event), untuk menghasilkan baud rate, atau untuk menghitung waktu.

**TMOD (Timer Mode Control)**
Register TMOD Terdiri dari :




Keterangan :

| Simbol | Alamat Bit | Fungsi |
|---|---|---|
| GATE | TMOD.7 dan TMOD.3 | Mengatur saluran sinyal denyut. Bila 0, saluran sinyal denyut hanya diatur bit TRx. Bila 1, kaki INT0 atau INT1 dipakai juga untuk mengatur saluran sinyal denyut. |
| C/T | TMOD.6 dan TMOD.2 | Mengatur sumber sinyal denyut yang diumpankan ke pencacah biner. Bila 0, Sinyal denyut diperoleh dari osilator Kristal yang |

| | | |
|---|---|---|
| | | frekuensinya sudah dibagi 12. Bila 1, maka sinyal denyut diperoleh dari kaki T0 atau kaki T1 |
| M1 | TMOD.5 dan TMOD.1 | Mode bit 1 |
| M0 | TMOD.4 dan TMOD.1 | Mode bit 0 |

Konfigurasi Mode Timer/Counter:

| M1 | M0 | Mode | Keterangan |
|---|---|---|---|
| 0 | 0 | 0 | 13-bit Timer |
| 0 | 1 | 1 | 16-bit Timer |
| 1 | 0 | 2 | 8-bit Timer |
| 1 | 1 | 3 | Split Mode |

Register TCON terdiri dari :

MSB LSB

| TF1(7) | TR1(6) | TF0(5) | TR0(4) | IE (3) | IT (2) | IE0(1) | IT0(0) |
|---|---|---|---|---|---|---|---|

Keterangan :

| Simbol | Posisi Bit | Fungsi |
|---|---|---|
| TF1 | TCON.7 | *Timer 1 Overflow flag*<br>*Flag* yang menandakan hitungan Timer 1 *Overflow* |
| TR1 | TCON.6 | *Timer 1 Run control bit*<br>Jika berlogika 1 = Timer 1 berjalan<br>Jika berlogika 0 = Timer 1 berhenti |
| TF0 | TCON.5 | *Timer 0 overflow flag*<br>*Flag* yang menandakan hitungan Timer 0 *Overflow* |
| TR0 | TCON.4 | *Timer 0 run control bit*<br>Jika berlogika 1 = Timer 0 berjalan<br>Jika berlogika 0 = Timer 0 berhenti |
| IE1 | TCON.3 | *Interrupt 1 Edge flag*<br>Berlogika 1, jika *Interrupt External 1* terdeteksi.<br>Berlogika 0, jika Intruksi Interrupt dijalankan. |

| | | |
|---|---|---|
| IT1 | TCON.2 | *Interrupt 1 type control bit*<br>Jika berlogika 1, maka INT1 (interrupt luar 1) dikenali pada sisi turun sinyal.<br>Jika berlogika 0, maka suatu saat akan dikenali saat suatu sinyal berlogika rendah. |
| IE0 | TCON.1 | *Interrupt 0 edge flag*<br>Berlogika 0, jika *Interrupt External 0* terdeteksi.<br>Berlogika 1, jika Intruksi Interrupt dijalankan. |
| IT0 | TCON.0 | *Interrupt 0 type control bit*<br>Jika berlogika 1, maka INT0 (interrupt luar 0) dikenali pada sisi turun sinyal.<br>Jika berlogika 0, maka suatu saat akan dikenali saat suatu sinyal berlogika rendah. |

**Baud rate** adalah frekuensi clock yang digunakan dalam pengiriman dan penerimaan data. Satuan baud rate pada umumnya adalah bps (bit per second), yaitu jumlah bit yang dapat ditransmisikan per detik.

Konfigurasi Baud rate untuk pemograman komunikasi serial dapat di buat dari sistem clock mikrokontroler atau dengan menggunakan Timer 1, jika Timer 1 dioperasikan pada mode 2 (8 bit auto reload), maka baud rate diberikan melalui persamaan berikut:

$$Baud\ Rate = \frac{2^{SMOD} x\ f}{384\ x\ (256 - TH1)}$$

Maka untuk mendapatkan nilai TH1 adalah :

$$TH1 = 256 - \left( \frac{f/_{384}}{Baudrate} \right) \quad \text{Jika bit SMOD berlogika 0}$$

$$TH1 = 256 - \left( \frac{f/_{192}}{Baudrate} \right) \quad \text{Jika bit SMOD berlogika 1}$$

**Contoh :** Diketahui MinSys AT89S51 menggunakan frekuensi kristal 11,059 MHz dan mengharapkan baud rate 9600 bps, maka nilai TH1 adalah …

- Menggunakan **SMOD berlogika 0**

TH1 = 256 – ((Frek. Osilator / 384) / Baud rate)
= 256 – (11059000 / 384) / 9600)
= 256 – 28799,4791 / 9600)
= 256 – (2,93)
= 256 - 3
= 253 = **0FDh**

Jika kita mengharapkan baud rate 19200 bps, maka :

TH1 = 256 – ((Frek. Osilator / 384) / Baud rate)
= 256 – ((11059000 / 384) / 19200)
= 256 – ((28799.4791) / 19200)
= 256 – 1,5
= 254,5

Satu hal yang harus diperhatikan dalam pengaturan baud rate adalah nilai baud rate dan nilai TH1 diusahakan harus tepat dan bukan merupakan pembulatan. Untuk komunikasi serial kecepatan tinggi, pembulatan terhadap nilai-nilai tersebut dapat mengakibatkan kekacauan dalam proses pengiriman atau penerimaan. Jika terdapat nilai pecahan, user disarankan untuk mengganti osilator dengan frekuensi yang sesuai. Untuk komunikasi dengan kecepatan rendah, toleransi terhadap kesalahan cukup besar sehingga pembulatan masih boleh dilakukan.

Untuk menghindari TH1 berupa pecahan, maka kita menggunakan persamaan yang menggunakan **SMOD berlogika 1,** Sehingga **:**

TH1 = 256 – ((Frek.Kristal / 192) / Baud rate)
= 256 – ((11059000 / 192) / 19200)
= 256 – ((57598,95) / 19200)
= 256 – (2,999)
= 256 - 3
= 253 = **0FDh**

Jadi nilai TH1 yang harus diisikan adalah OFDH. Berdasarkan hal diatas jika diberikan kristal dengan frekuensi 11,059 MHz maka untuk mengkonfigurasi serial port (UART) memiliki baud rate 19200 bps harus dilakukan hal-hal berikut :

- Konfigurasikan serial port pada mode 1 atau 3
- Konfigurasikan Timer 1 pada mode 2 (8 bit)
- Set TH1 pada niai 253 = OFDH untuk menghasilkan 19200 bps
- Set bit SMOD (PCON.7)

Beberapa konfigurasi Baud rate dibawah ini :

| Serial | | | | *Timer* 1 | | |
|---|---|---|---|---|---|---|
| **Mode** | ***Baud Rate*** | **Frekuensi Osilator** | **SMOD** | **C/T** | **Mode** | ***Reload*** |
| 0 | 1,6667 Mbps (max.) | 20 MHz | X | X | X | X |
| 2 | 625 Kbps (max.) | 20 MHz | 1 | X | X | X |
| 1, 3 | 104,1667 Kbps (max.) | 20 MHz | 1 | 0 | 2 | FFh |
| 1, 3 | 19,2 Kbps | 11,0592 MHz | 1 | 0 | 2 | FDh |
| 1, 3 | 9,6 Kbps | 11,0592 MHz | 0 | 0 | 2 | FDh |
| 1, 3 | 4,8 Kbps | 11,0592 MHz | 0 | 0 | 2 | FAh |
| 1, 3 | 2,4 Kbps | 11,0592 MHz | 0 | 0 | 2 | F4h |
| 1, 3 | 1,2 Kbps | 11,0592 MHz | 0 | 0 | 2 | E8h |
| 1, 3 | 137,5 bps | 11,9856 MHz | 0 | 0 | 2 | 1Dh |
| 1, 3 | 110 bps | 6 MHz | 0 | 0 | 2 | 72h |
| 1, 3 | 110 bps | 12 MHz | 0 | 0 | 1 | FEEBh |

**Interrupt**

Interrupt merupakan fitur penting pada suatu mikrokontroler. Dengan adanya interrupt, maka mikrokontroler dapat menghentikan proses yang sedang dijalankan dan melaksanakan rutin interrupt. Contoh interrupt adalah jika suatu perhitungan pada Timer telah melimpah (overflow) maka akan terjadi interrupt TF0 dan TF1. Interrupt ini akan memberitahukan pada mikrokontroler bahwa hitungan Timer / Counter telah melimpah. Atau jika serial port menerima data, maka akan terjadi interrupt serial port sehingga mikrokontroler akan mengetahui bahwa ada data yang datang dari serial port. Setelah interrupt diketahui, maka bergantung pada program yang telah diberikan kepada mikrokontroler hal apakah yang harus dikerjakan.

Pada Mikrokontroler AT89S51, Ada dua buah Special Function Register yang digunakan untuk mengontrol interrupt, yaitu IE (Interupt Enable) alamat A8H dan IP (Interupt Priority Control) alamat B8H. IE digunakan untuk mengontrol interrupt mana saja yang akan diaktifkan, sedangkan IP akan menentukan interrupt mana yang memiliki prioritas tinggi dan interrupt mana pula yang memiliki prioritas rendah.

Bit-bit pada register IE adalah sebagai berikut :

| MSB | | | | | | | LSB |
|---|---|---|---|---|---|---|---|
| EA(7) | (6) | (5) | ES(4) | ET1(3) | EX1(2) | ET0(1) | EX0(0) |

Keterangan :

| Simbol | Posisi Bit | Fungsi |
|---|---|---|
| EA | IE.7 | *Enable All*<br>Mengaktifkan seluruh interupsi, Jika berlogika 1 |
| - | IE.6<br>IE.5 | Tidak Digunakan |
| ES | IE.4 | *Enable Serial*<br>Mengaktifkan interupsi Serial, Jika berlogika 1 |
| ET1 | IE.3 | *Enable Timer 1*<br>Mengaktifkan interupsi Timer 1, Jika berlogika 1 |
| EX1 | IE.2 | *Enable External Interrupt 1*<br>Mengaktifkan interupsi External 1, Jika berlogika 1 |
| ET0 | IE.1 | *Enable Timer 0*<br>Mengaktifkan interupsi Timer 0, Jika berlogika 1 |
| EX0 | IE.0 | *Enable External Interrupt 0*<br>Mengaktifkan interupsi External 0, Jika berlogika 1 |

Pada mikrokontroler MCS-51 ada 5 buah sumber interrupt. masing masing sumber interrupt memiliki vector alamat masing-masing. Berikut daftar interrupt pada mikrokontroler MCS-51 dan alamat vektor interrupt :

| Nama Interrupt | Flag | Alamat Vektor Interrupt | Prioritas |
|---|---|---|---|
| Interrupt Luar 0 | lEO | 3H | 1 (tertinggi) |
| Timer 0 | TFO | BH | 2 |
| Interrupt Luar 1 | IE1 | 13H | 3 |
| Timer 1 | TH | 1BH | 4 |
| Serial | TI atau RI | 23H | 5 (terendah) |

Jika register IE (Interupt Enable) digunakan untuk mengatur interrupt mana saja yang aktif, maka register IP (Interrupt Priority Control) digunakan untuk mengatur prioritas dari interrupt. Bit -bit pada register IP didefinisikan sebagai berikut :

MSB LSB

| (7) | (6) | (5) | PS(4) | PT1(3) | PX1(2) | PT0(1) | PX0(0) |
|---|---|---|---|---|---|---|---|

Keterangan :

| Simbol | Posisi Bit | Fungsi |
|---|---|---|
| - | IP.7<br>IP.6<br>IP.5 | Tidak Digunakan |

| | | |
|---|---|---|
| PS | IP.4 | *Serial Interrupt Priority*<br>Memprioritaskan interupsi Serial |
| PT1 | IP.3 | *Timer 1 Interrupt Priority*<br>Memprioritaskan interupsi Timer 1 |
| PX1 | IP.2 | *External Interrupt 1 Priority*<br>Memprioritaskan interupsi External 1 |
| PT0 | IP.1 | *Timer 0 Interrupt Priority*<br>Memprioritaskan interupsi Timer 0 |
| PX0 | IP.0 | *External Interrupt 0 Priority*<br>Memprioritaskan interupsi External 0 |

**Percobaan 4A : Komunikasi Serial PC ► Mikrokontroler**

a. Jalankan **MIDE-51**.

b. Klik **File > New,** Kemudian ketikkan program *Assembly* berikut ini :

```
$mod51

ORG 00H
SJMP INIT_SERIAL

ORG 23H
SJMP TERIMA

INIT_SERIAL: MOV PCON,#00H
             MOV SCON,#50H
             MOV TMOD,#20H
             MOV TH1,#0FDH
             SETB TR1
             SETB ES
             SETB EA

LOOP : SJMP LOOP

TERIMA : MOV A,SBUF
         CLR RI
         MOV P2,A
         RETI

END
```

c. Save Program dengan cara klik **File > Save**, Save file dengan ekstensi ***.asm** kemudian klik **Build > Build** untuk *compile***.**

d. Jika tidak ada error, maka akan menghasilkan file *.hex, dan lakukan Flashing Mikrokontroler dengan file *.hex.
***Catatan : Cara Flashing Mikrokontroler dapat ditanyakan pada Asisten***

e. Hubungkan PC dengan Mikrokontroler dengan kabel Serial.

f. Jalankan **PuTTY**

g. Pada **Category**, Klik **Connection – Serial.** Kemudian aturlah seperti dibawah ini :

- Serial line to connect to : <pilih port COM yang terhubung dengan µC>
- Speed (Baud) : 9600
- Data bits : 8
- Stop bits : 1
- Parity : None
- Flow control : None

***Catatan : Daftar port COM yang aktif, dapat dilihat pada Device Manager.***

h. Pada **Category**, Klik **Terminal,** Kemudian atur seperti dibawah ini :
   - Local echo : pilih **Force On**
   - Local line editing : pilih **Auto**

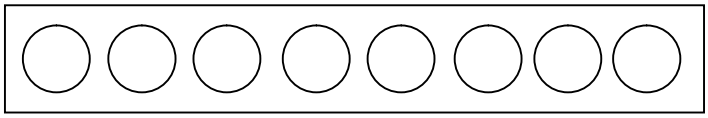
i. Pada **Category,** Klik **Session.** dan pada **Connection type:,** Pilih **Serial,** Kemudian klik **Open.**

j. Ketikkan satu karakter pada terminal PuTTY, dan Amati Hasilnya.

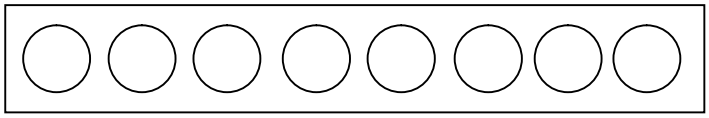
k. Data Pengamatan :

** Amati nyala led yang ada pada MinSys*
** Isilah **Kode ASCII** berbentuk **Heksa**, dari **Nyala LED** representasi **Biner***

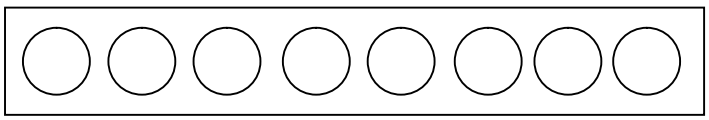
Jika ditekan ‘a’ pada keyboard : Kode ASCII:



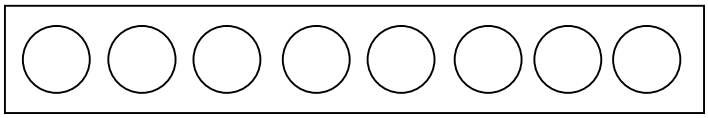
Jika ditekan ‘B’ pada keyboard :



Jika ditekan ‘1’ pada keyboard :



Jika ditekan ‘=’ pada keyboard :



Keterangan :

 = Menyala = Logika 0  = Mati = Logika 1

l. Jelaskan output yang dihasilkan :

m. **Jangan close** terminal PuTTY, karena akan digunakan selanjutnya.

**Percobaan 4B : Komunikasi Serial Mikrokontroler ► PC**

a. Jalankan **MIDE-51**.

b. Klik **File > New,** Kemudian ketikkan program *Assembly* berikut ini :

```
$mod51

ORG 00H
MOV P2,#0FFH
MOV P3,#0FFH

PIL      : JNB P3.2,MULAI
           SJMP PIL

MULAI    : ACALL INIT_SERIAL
           MOV R3,#15H
           MOV DPTR,#KATA

ULANG    : CLR A
           MOVC A,@A+DPTR
           ACALL KIRIM
           INC DPTR
           ACALL DELAY
           DJNZ R3,ULANG
           SJMP PIL

INIT_SERIAL : MOV PCON,#00H
              MOV SCON,#40H
              MOV TMOD,#20H
              MOV TH1,#0FDH
              SETB TR1
              SETB ES
              SETB EA
              RET

KATA : DB "Switch P3.2 Ditekan !",0

KIRIM : CLR TI
        MOV SBUF,A
        RET

DELAY : MOV R2,#07FH

LAGI :  MOV R1,#07FH
        DJNZ R1,$
        DJNZ R2,LAGI
        RET
END
```

c. **Save program** dengan cara klik **File > Save,** kemudian klik **Build > Build** untuk *compile.*

d. Jika tidak ada error, maka akan menghasilkan file *.hex, dan lakukan Flashing Mikrokontroler dengan file *.hex.
***Catatan : Cara Flashing Mikrokontroler dapat ditanyakan pada Asisten.***

e. Hubungkan PC dengan Mikrokontroler dengan kabel Serial.

f. Membuka kembali program PuTTY yang sebelumnya digunakan.

g. Klik **icon PuTTY** pada **Title Bar,** kemudian klik **Reset Terminal.**

h. Kemudian tekan button **P3.2** pada MinSys, dan amati output yang tampil pada terminal PuTTY.

i. Jelaskan output yang dihasilkan :

| |
|---|
| |

j. Close terminal PuTTY.

**Percobaan 4C : Aplikasi Kendali Nyala LED dengan Komunikasi Serial**

- **Membuat Program Mikrokontroler**

a. Jalankan **MIDE-51.**

b. Klik **File > New,** Kemudian ketikkan program *Assembly* berikut ini :

```
$mod51

ORG 00H
SJMP INIT_SERIAL

ORG 23H
TERIMA: MOV A,SBUF
        CLR RI
        SJMP LAMPU1_ON

INIT_SERIAL : MOV PCON,#00H
              MOV SCON,#50H
              MOV TMOD,#20H
              MOV TH1,#0FDH
              SETB TR1
              SETB ES
              SETB EA
```

```
LOOP : SJMP LOOP

LAMPU1_ON   : CJNE A,#61H,LAMPU1_OFF
              CLR P2.0

LAMPU1_OFF  : CJNE A,#62H,LAMPU2_ON
              SETB P2.0

LAMPU2_ON   : CJNE A,#31H,LAMPU2_OFF
              CLR P2.1

LAMPU2_OFF  : CJNE A,#32H,TERIMA
              SETB P2.1
              SJMP TERIMA
END
```

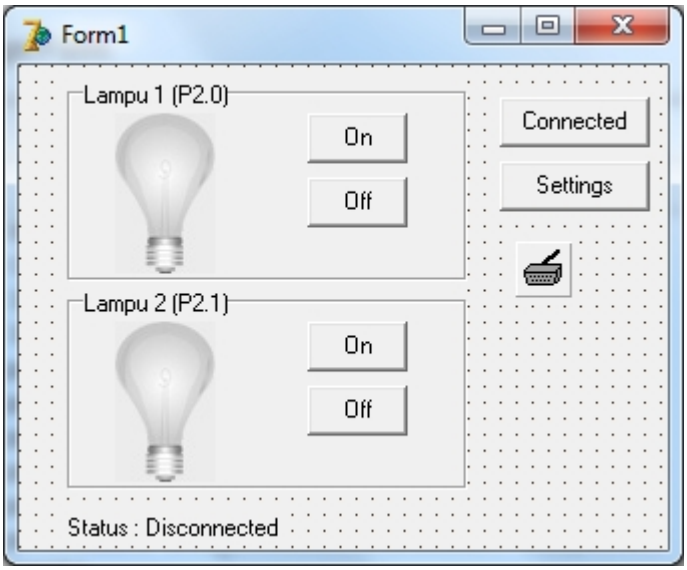
c. **Save program** dengan cara klik **File > Save,** kemudian klik **Build > Build** untuk *compile.*

d. Jika tidak ada error, maka akan menghasilkan file *.hex, dan lakukan Flashing Mikrokontroler dengan file *.hex.
***Catatan : Cara Flashing Mikrokontroler dapat ditanyakan pada Asisten.***

e. Hubungkan PC dengan Mikrokontroler dengan kabel Serial.

- **Membuat Program Antarmuka dengan Delphi 7**

a. Design Form




b. Komponen Properties

| Tab | Komponen | Properti | Nilai |
|---|---|---|---|
| Standard | GroupBox1 | Caption | Lampu 1 (P2.0) |
| Additional | Image1 | Picture (di klik button •••) | Klik **Load..** , pilih gambar lampu mati (**off.jpg**) |

| | | | |
|---|---|---|---|
| | | Width<br>Height | 50<br>80 |
| Standard | Button1 | Name<br>Caption<br>Enabled | btn_on1<br>On<br>False |
| Standard | Button2 | Name<br>Caption<br>Enabled | btn_off1<br>Off<br>False |
| Standard | GroupBox2 | Caption | Lampu 2 (P2.1) |
| Additional | Image2 | Picture (di klik button •••)<br><br>Width<br>Height | Klik **Load..** , pilih gambar lampu mati (**off.jpg**)<br><br>50<br>80 |
| Standard | Button3 | Name<br>Caption<br>Enabled | btn_on2<br>On<br>False |
| Standard | Button4 | Name<br>Caption<br>Enabled | btn_off2<br>Off<br>False |
| Standard | Button5 | Caption | Connected |
| Standard | Button6 | Caption | Settings |
| Standard | Label1 | Caption | Status : Disconnected |
| CPortLib | ComPort1 | | |

c. Source Code

Lengkapi Source Code untuk komponen **btn_on1**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.btn_on1Click(Sender: TObject);
begin
Comport1.WriteStr('a');
Image1.Picture.LoadFromFile('on.jpg');
btn_on1.Enabled := False;
btn_off1.Enabled := True;
end;
```

Lengkapi Source Code untuk komponen **btn_off1**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.btn_off1Click(Sender: TObject);
begin
Comport1.WriteStr('b');
Image1.Picture.LoadFromFile('off.jpg');
btn_on1.Enabled := True;
btn_off1.Enabled := False;
end;
```

Lengkapi Source Code untuk komponen **btn_on2**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.btn_on2Click(Sender: TObject);
begin
Comport1.WriteStr('1');
```

```
Image2.Picture.LoadFromFile('on.jpg');
btn_on2.Enabled := False;
btn_off2.Enabled := True;
end;
```

Lengkapi Source Code untuk komponen **btn_off2**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.btn_off2Click(Sender: TObject);
begin
Comport1.WriteStr('2');
Image2.Picture.LoadFromFile('off.jpg');
btn_on2.Enabled := True;
btn_off2.Enabled := False;
end;
```

Lengkapi Source Code untuk komponen **Button5**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.Button5Click(Sender: TObject);
begin
if Button5.Caption = 'Connected' then
begin
ComPort1.Connected := True;
btn_on1.Enabled := True;
btn_off1.Enabled := True;
btn_on2.Enabled := True;
btn_off2.Enabled := True;
Label1.Caption := 'Status : Connected with ' + ComPort1.Port;
Button5.Caption := 'Disconnected';
end
else
begin
ComPort1.Connected := False;
btn_on1.Enabled := False;
btn_off1.Enabled := False;
btn_on2.Enabled := False;
btn_off2.Enabled := False;
Image1.Picture.LoadFromFile('off.jpg');
Image2.Picture.LoadFromFile('off.jpg');
Label1.Caption := 'Status : Disconnected';
Button5.Caption := 'Connected';
end
end;
```

Lengkapi Source Code untuk komponen **Button6**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.Button6Click(Sender: TObject);
begin
Comport1.ShowSetupDialog;
end;
```

d. Save project, dengan cara klik **File > Save All,** dan save kedalam **satu folder**, disimpan pada **Desktop.** Kemudian Copykan file **on.jpg**, **off.jpg** kedalam file tersebut.

e. Jalankan program yang telah dibuat dengan menekan tombol **F9** pada Keyboard, atau Klik **Run > Run** pada tool bar.

f. Klik button **Settings** pada Form.

Atur Settings seperti dibawah ini :

- Port : <pilih port COM yang terhubung dengan µC>
- Baud rate : 9600
- Data bits : 8
- Stop bits : 1
- Parity : None
- Flow control : None

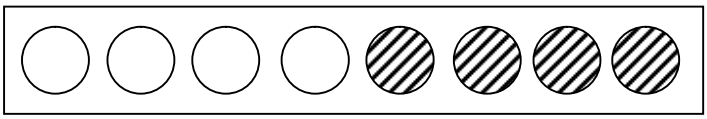
Kemudian klik **OK**.

g. Kemudian klik button **Connected**, Klik **Button1-4**, Kemudian amati perubahan yang terjadi pada program yang telah dibuat dan MinSys. Jelaskan output yang dihasilkan :

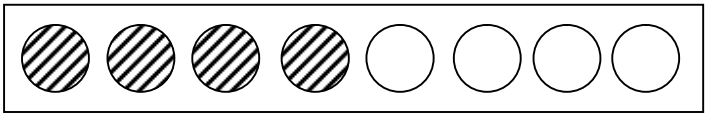
**Percobaan 4D : Latihan Mandiri**

Buatlah Program Mikrokontroler dengan inputan penekanan pada keyboard dalam terminal PuTTY, dimana penekanan pada keyboard, dapat mengatur nyala LED, dengan aturan :

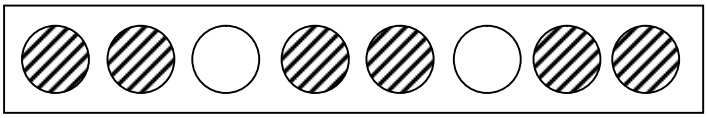
a. Jika ditekan ‘1’ pada keyboard :



b. Jika ditekan ‘2’ pada keyboard :



c. Jika ditekan ‘3’ pada keyboard :



Keterangan :

 = Menyala = Logika 0  = Mati = Logika 1

Kode Program pada MinSys :

| | |
|---|---|
| **Nama Asisten** | |
| **Tgl. Praktikum** | |
| **Paraf Asisten** | |
| | |

Percobaan

5

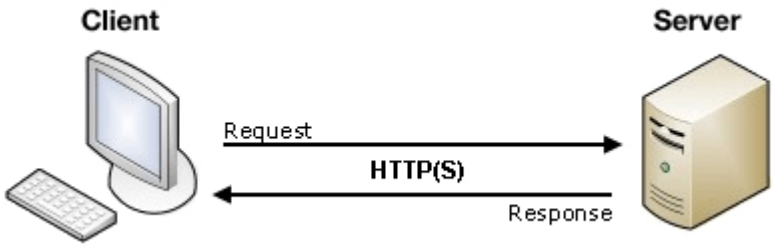
# Konsep Komunikasi Client - Server

**Tujuan :**
1. Memahami tentang komunikasi data dengan arsitektur Client-Server.
2. Memahami cara kerja dari komunikasi data secara Client-Server.
3. Mengenal konsep Socket Programming pada komunikasi Client-Server.

**Alat :**
1. 2 PC.
2. Delphi 7.
3. Kabel LAN dengan konfigurasi *Crossover*.

**Dasar Teori :**

- ***Client-Server***
  *Client-Server* adalah suatu arsitektur jaringan di mana tiap-tiap komputer atau proses dalam jaringan itu ada yang sebagai client, dan ada yang sebagai server. Proses – Proses yang ada di client memerlukan layanan dari proses - proses di server. Server adalah komputer-komputer atau proses-proses yang disesdiakan untuk mengendalikan dan mengkoordinasi disk drive (file server), printer, atau traffic jaringan. Sementara itu client adalah PC – PC atau workstation dimana pengguna menjalankan aplikasi. Client membututhkan server-server untuk sumber daya (resources) seperti file-file, piranti-piranti, dan juga pemrosesan sumber daya.




*Gambar 5.1 Arsitektur Client-Server*

- **TCP (*Transmission Control Protocol)***
  TCP (Transmission Control Protocol) merupakan protokol yang bersifat connection-oriented, artinya komunikasi yang melewatinya membutuhkan handshaking untuk mengatur koneksi end-to-end. Koneksi dapat dibuat dari client ke server, dan kemudian banyak data dapat dikirimkan melalui koneksi itu. Aplikasi berbasis TCP biasanya membutuhkan ketepatan data hingga 100% tapi tidak memperdulikan lama pengiriman, sedangkan aplikasi berbasis UDP biasanya adalah aplikasi yang tidak terlalu mempedulikan ketepatan data tapi sangat peduli dengan delay pengiriman. Contoh aplikasi TCP adalah web browser, sedangkan UDP adalah Video Converence. Untuk membangun aplikasi hal pertama yang perlu dilakukan adalah menganalisa jenis aplikasi kita, kebutuhan bandwidth, kebutuhan ketersampaian data dan sensitifitas terhadap delay. Berdasarkan hal ini kita bisa menentukan protokol apa yang kita gunakan, entah TCP atau UDP.

- **Port pada TCP**
  Aplikasi client menggunakan nomor port untuk memberitahu mesin tujuan dan service TCP mana yang diinginkannya. Server untuk aplikasi tertentu menggunakan well-known port untuk mengetahui koneksi dari client yang meminta servicenya. Port – port yang digunakan dalam transport layer menggunakan 16-bit integer (0 – 65535), dengan satu sama lain harus berbeda (unique). Pada saat client ingin membangun koneksi dengan Server, client harus mengetahui port dari server yang dituju dan protokol apa yang digunakan (UDP atau TCP).

  Client di sisi sebaliknya, umumnya menggunakan ephemeral port atau biasa disebut short-lived ports. Nomor pada port ephemeral yang digunakan oleh client diberikan oleh Transport Protocol. Client tidak perlu tahu nomor port ephemeral yang digunakan, yang jelas semua port ephemeral yang digunakan pasti bersifat unique.

  The Internet Assigned Numbers Authority (IANA) telah mengelompokkan nomor – nomor port yang dibagi menjadi tiga bagian :

  1. Well-known ports: 0 – 1023. Pada range ini merupakan nomor – nomor port yang telah digunakan oleh IANA. Contoh nya adalah Web server yang menggunakan port 80, FTP menggunakan 21 dll.
  2. Registered ports: 1024 – 49252. Nomor – nomor port pada range ini tidak digunakan oleh IANA, IANA mengelompokkan port – port ini untuk dapat digunakan sebagai server untuk TCP atau UDP. Contohnya antara port 6000 sampai 6063 digunakan untuk X Windows server. Aplikasi yang kita gunakan juga bisa menggunakan port ini.
  3. Private ports: 49152 – 65535. Nomor – nomor port pada range ini adalah ephemeral port. Namun tentu saja tidak menutup kemungkinan nilai ephemeral port mempunya nilai diluar range ini, hal tersebut bergantung juga dari Sistem Operasi yang digunakan.

  Jadi dapat disimpulkan bahwa koneksi TCP memiliki 1 buah Local IP address, local port number, foreign ip address dan foreign port number.

- **Socket**
  Socket adalah mekanisme komunikasi jaringan yang memungkinkan terjadinya pertukaran data antar program atau proses antar mesin.

  Socket terdiri dari elemen-elemen utama sebagai berikut :
  1. Local IP
  2. Local Port
  3. Remote IP
  4. Remote Port

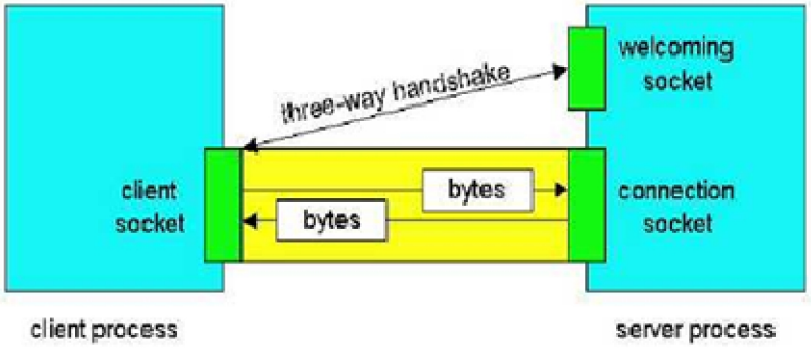
  dalam komunikasi antara dua pihak, tentunya harus digunakan kesepakatan aturan dan format yang sama agar komunikasi dapat dimengerti. Seperti halnya dua orang yang menggunakan bahasa yang sama, maka bahasa di sini berfungsi sebagai protokol. Protokol yang digunakan dalam socket dapat menggunakan TCP ataupun UDP.

Contoh komunikasi sederhana adalah komunikasi antara komputer A dan komputer B. Baik komputer A maupun komputer B harus memiliki identitas unik, yang direpresentasikan oleh IP masing-masing.

Komunikasi yang terjadi melalui port, sehingga baik komputer A maupun komputer B harus memiliki port yang dapat diakses satu sama lain.

- **Cara Kerja Socket menggunakan TCP**

  Cara kerja Socket yang menggunakan TCP dapat digambarkan oleh di bawah ini :

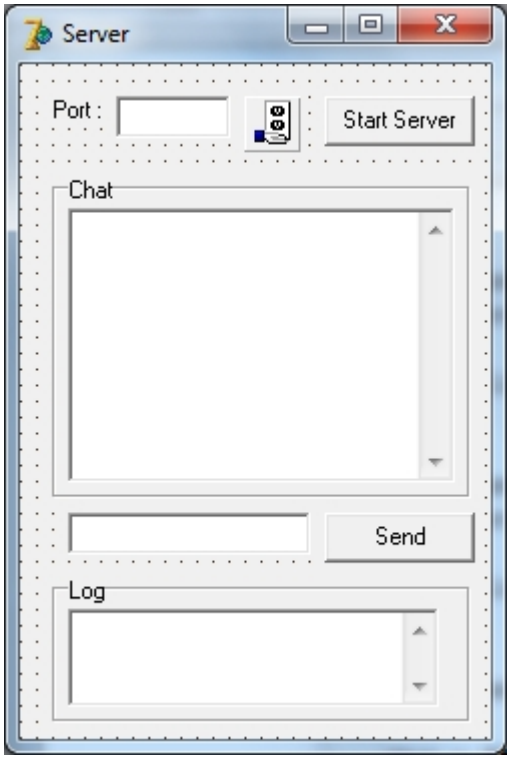



Detail dari proses tersebut adalah :

1. Untuk bisa melakukan koneksi client server, program server harus berjalan terlebih dahulu.
2. Di sisi server disediakan sebuah socket, yang disebut welcoming socket yang fungsinya untuk mendeteksi adanya permintaan koneksi dari sisi client.
3. Di sisi client terdapat client socket. Jika ingin menghubungi server, maka melalui client socket-nya, client membuat inisialisai koneksi ke welcoming socket milik server, dengan mode three-way handshake.
4. Setelah welcoming socket menerima inisialisasi koneksi dari client socket, aplikasi server akan membuat connection socket di sisi server. Dengan connection socket ini, client socket dan connection socket berinteraksi satu sama lain untuk mengirim dan menerima data.
5. Client membaca data yang dikirim oleh server dari client socket-nya. Kemudian menampilkan data tersebut di monitor.

**Percobaan 5A : Membuat Aplikasi Server Sederhana**

a. Design Form




b. Komponen Properties

| Tab | Komponen | Properti | Nilai |
|---|---|---|---|
| Standard | Label1 | Caption | Port : |
| Standard | Edit1 | Text | *(dikosongkan)* |
| Internet | ServerSocket1 | | |
| Standard | Button1 | Caption | Start Server |
| Standard | GroupBox1 | Caption | Chat |
| Standard | Memo1 | Lines (di klik button •••)<br>ReadOnly<br>ScrollBars | *(kosongkan String List)*<br>True<br>ssVertical |
| Standard | Edit2 | Text | *(dikosongkan)* |
| Standard | Button2 | Caption<br>Enabled | Send<br>False |
| Standard | GroupBox2 | Caption | Log |
| Standard | Memo2 | Lines (di klik button •••)<br>ReadOnly<br>ScrollBars | *(kosongkan String List)*<br>True<br>ssVertical |

c. Source Code
Lengkapi Source Code untuk komponen **Button1**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.Button1Click(Sender: TObject);
begin
if Button1.Caption = 'Start Server' then
begin
```

```
ServerSocket1.Port := StrToInt(Edit1.Text);
ServerSocket1.Active := True;
Edit1.Enabled := False;
Memo2.Lines.Add('['+ TimeToStr(now)+'] '+'Server Start');
Button1.Caption := 'Stop Server';
Button2.Enabled := True;
end
else
begin
ServerSocket1.Active := False;
Edit1.Enabled := True;
Memo2.Lines.Add('['+ TimeToStr(now)+'] '+'Server Stop');
Button1.Caption := 'Start Server';
Button2.Enabled := False;
end
end;
```

Lengkapi Source Code untuk komponen **Button2**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.Button2Click(Sender: TObject);
var
i : Integer;
Str : String;
begin
Str := '['+ TimeToStr(now)+'] Server >> '+ Edit2.Text;
Edit2.Text := #10;
Memo1.Lines.Add(Str);
for i:=0 to ServerSocket1.Socket.ActiveConnections-1 do
ServerSocket1.Socket.Connections[i].SendText(Str);
end;
```

Lengkapi Source Code untuk komponen **ServerSocket1**, Event **OnClientConnect** seperti dibawah ini:

```
procedure TForm1.ServerSocket1ClientConnect(Sender: TObject;
  Socket: TCustomWinSocket);
begin
Memo2.Lines.Add('['+ TimeToStr(now)+'] '+'Client Connected : ' +
Socket.RemoteAddress);
end;
```

Lengkapi Source Code untuk komponen **ServerSocket1**, Event **OnClientDisconnect** seperti dibawah ini:

```
procedure TForm1.ServerSocket1ClientDisconnect(Sender: TObject;
  Socket: TCustomWinSocket);
begin
Memo2.Lines.Add('['+ TimeToStr(now)+'] '+'Client Disconnected : ' +
Socket.RemoteAddress);
end;
```

Lengkapi Source Code untuk komponen **ServerSocket1**, Event **OnClientRead** seperti dibawah ini:

```
procedure TForm1.ServerSocket1ClientRead(Sender: TObject;
  Socket: TCustomWinSocket);
begin
Memo1.Lines.Add(Socket.ReceiveText);
end;
```

Lengkapi Source Code untuk komponen **Edit2**, Event **OnKeyPress** seperti dibawah ini:

```
procedure TForm1.Edit2KeyPress(Sender: TObject; var Key: Char);
begin
if Key = #13 then Button2Click(Sender);
end;
```

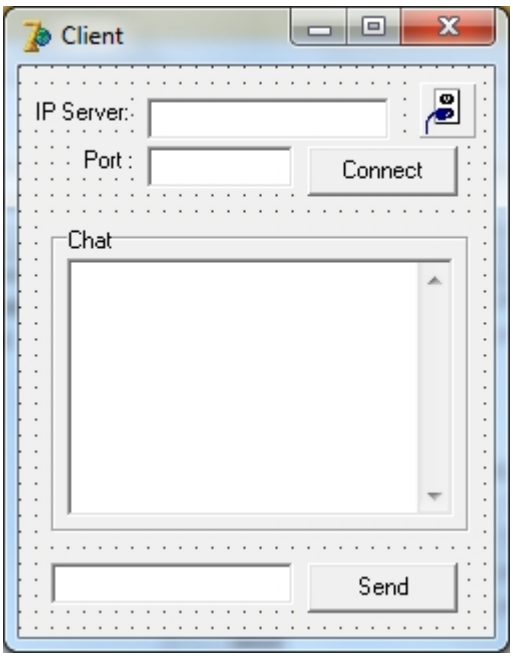
d. Save project, dengan cara klik **File > Save All,** dan save kedalam **satu folder** dengan nama **Server.**

e. Jalankan program yang telah dibuat dengan menekan tombol **F9** pada Keyboard, atau Klik **Run > Run** pada tool bar.

f. Buatlah dengan Project baru, dengan cara klik **File > New > Application.**

**Percobaan 5B : Membuat Aplikasi Client Sederhana**

a. Design Form




b. Komponen Properties

| Tab | Komponen | Properti | Nilai |
|---|---|---|---|
| Standard | Label1 | Caption | IP Server : |
| Standard | Edit1 | Text | *(dikosongkan)* |

| | | | |
|---|---|---|---|
| Standard | Label2 | Caption | Port : |
| Standard | Edit1 | Text | *(dikosongkan)* |
| Internet | ClientSocket1 | | |
| Standard | Button1 | Caption | Connect |
| Standard | GroupBox1 | Caption | Chat |
| Standard | Memo1 | Lines (di klik button •••)<br>ReadOnly<br>ScrollBars | *(kosongkan String List)*<br>True<br>ssVertical |
| Standard | Edit2 | Text | *(dikosongkan)* |
| Standard | Button2 | Caption<br>Enabled | Send<br>False |

c. Source Code
Lengkapi Source Code untuk komponen **Button1**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.Button1Click(Sender: TObject);
begin
if Button1.Caption = 'Connect' then
begin
ClientSocket1.Host := Edit1.Text;
ClientSocket1.Port := StrToInt(Edit2.Text);
ClientSocket1.Active := True;
Edit1.Enabled := False;
Edit2.Enabled := False;
Button2.Enabled := True;
Button1.Caption := 'Disconnect';
end
else
begin
ClientSocket1.Active := False;
Edit1.Enabled := True;
Edit2.Enabled := True;
Button2.Enabled := False;
Button1.Caption := 'Connect';
end
end;
```

Lengkapi Source Code untuk komponen **Button2**, Event **OnClick** seperti dibawah ini:

```
procedure TForm1.Button2Click(Sender: TObject);
var
Str : String;
begin
Str := '['+TimeToStr(now)+'] Client >> '+ Edit3.Text;
ClientSocket1.Socket.SendText(Str);
Memo1.Lines.Add(Str);
Edit3.Text := #10;
end;
```

Lengkapi Source Code untuk komponen **Edit3**, Event **OnKeyPress** seperti dibawah ini:

```
procedure TForm1.Edit3KeyPress(Sender: TObject; var Key: Char);
begin
if Key = #13 then Button2Click(Sender);
end;
```

Lengkapi Source Code untuk komponen **ClientSocket1**, Event **OnRead** seperti dibawah ini:

```
procedure TForm1.ClientSocket1Read(Sender: TObject;
  Socket: TCustomWinSocket);
begin
Memo1.Lines.Add(Socket.ReceiveText);
end;
```

d. Save project, dengan cara klik **File > Save All,** dan save kedalam **satu folder** dengan nama **Server.**

e. Jalankan program yang telah dibuat dengan menekan tombol **F9** pada Keyboard, atau Klik **Run > Run** pada tool bar.

**Percobaan 5C : Uji Coba**

- Ujicoba dengan **localhost (127.0.0.1)**

  a. Buka kembali folder **Server** dan **Client** yang berisikan project yang telah dibuat.

  b. Jalankan file berekstensi **.exe** pada masing-masing folder.

  c. Pada form **Server**, masukan port dengan port yang tidak digunakan (misalkan : 50), kemudian klik **Start Server.**

  d. Pada form **Client,** masukan port dengan port yang digunakan pada form **Server.** Kemudian masukan **IP Server** dengan **127.0.0.1,** Kemudian klik **Connect.**

  e. Lakukan pengiriman data dari **Client ke Server,** dan **Server ke Client.**

  f. Amati perubahan pada masing-masing form.

  g. Klik **Disconnect** pada form **Client**, dan **Stop Server** pada form **Server.**

- Ujicoba dengan **2 PC**

  a. Hubungkan kedua PC dengan **Kabel LAN (Cross)**. Kemudian tentukan PC mana yang digunakan sebagai **Server**, dan **Client**.

  b. Konfigurasikan IP Address, baik PC yang dijadikan **Server**, dan **Client**. dengan cara tekan tombol **Windows Logo + R** pada Keyboard**,** kemudian masukan perintah **ncpa.cpl ,** kemudian klik **OK.**

c. Klik kanan **Local Area Connection,** Kemudian klik **Properties**.

d. Double klik **Internet Protocol Version 4 (TCP/IPv4).**

e. Klik **Use the following IP address:,** Kemudian atur sebagai berikut :

**Pada PC Server :**

IP Address : 192.168.24.1
Subnet Mask : 255.255.255.0
Default Gateway : 192.168.24.2

**Pada PC Client :**

IP Address : 192.168.24.2
Subnet Mask : 255.255.255.0
Default Gateway : 192.168.24.1

Kemudian klik **OK**.

**Pengaturan IP Address, Subnet Mask, dan Default Gateway dapat ditentukan sendiri, namun harus dalam satu **network**.*

f. Pada form **Server**, masukan port dengan port yang tidak digunakan (misalkan : 50), kemudian klik **Start Server**.

g. Pada form **Client,** masukan port dengan port yang digunakan pada form **Server.** Kemudian masukan **IP Server** dengan **IP Address Server,** Kemudian klik **Connect.**

h. Amati perubahan pada masing-masing form. Apa output yang dihasilkan pada program yang telah dibuat, Jelaskan !

| |
|---|
| |

| | |
|---|---|
| **Nama Asisten** | |
| **Tgl. Praktikum** | |
| **Paraf Asisten** | |
| | |