Die serielle RS232 Schnittstelle

von Dr. Helmut Leimeister

Übersicht:

	Programm	Seite
Literatur, Geräte		2
Die RS232 Schnittstelle (Theorie)		3
Ausgang schalten	Ausgang_Eingang	4
Das Programm : Ausgang_Eingang		5
Halbleiter, Diode, Transistor (Theorie)		6
Blitzlicht	Ausgang_Eingang	7
Die Register der RS232 (Theorie)		8
Assembler-Programmierung	Ausgang-mit-ASM	9
Ampelschaltung	Ampel	10
Wechsel-Blinklicht	LED-Blinklicht	11
Zeitschaltung	Zeituhr	12
Die Eingänge der RS232		13
Berührungssensor	Ausgang_Eingang	14
Lichtschranke 1, Zeitmessung	Lichtschranke	15
Lichtschranke 2, Rolltreppe	Ausgang_Eingang	16
Ladezeit für einen Kondensator	Kondensator	17
Reaktionszeit	Reaktionszeit	18
Logik NOT; UND/ NAND; OR/ NOR; EXOR	Logik	19
Gatter: NOT; UND/NAND; OR/NOR; EXOR	Ausgang_Eingang	20,21

Literatur:

http://www.ak-modul-bus.de/
Die Firma AK-modul-bus vertreibt Elektronikbauteile, Elektronikbücher und die hier verwendeten Steckbuchsen.
Auf der Internetseite sind viele Versuche mit der RS232 Schnittstelle veröffentlicht.

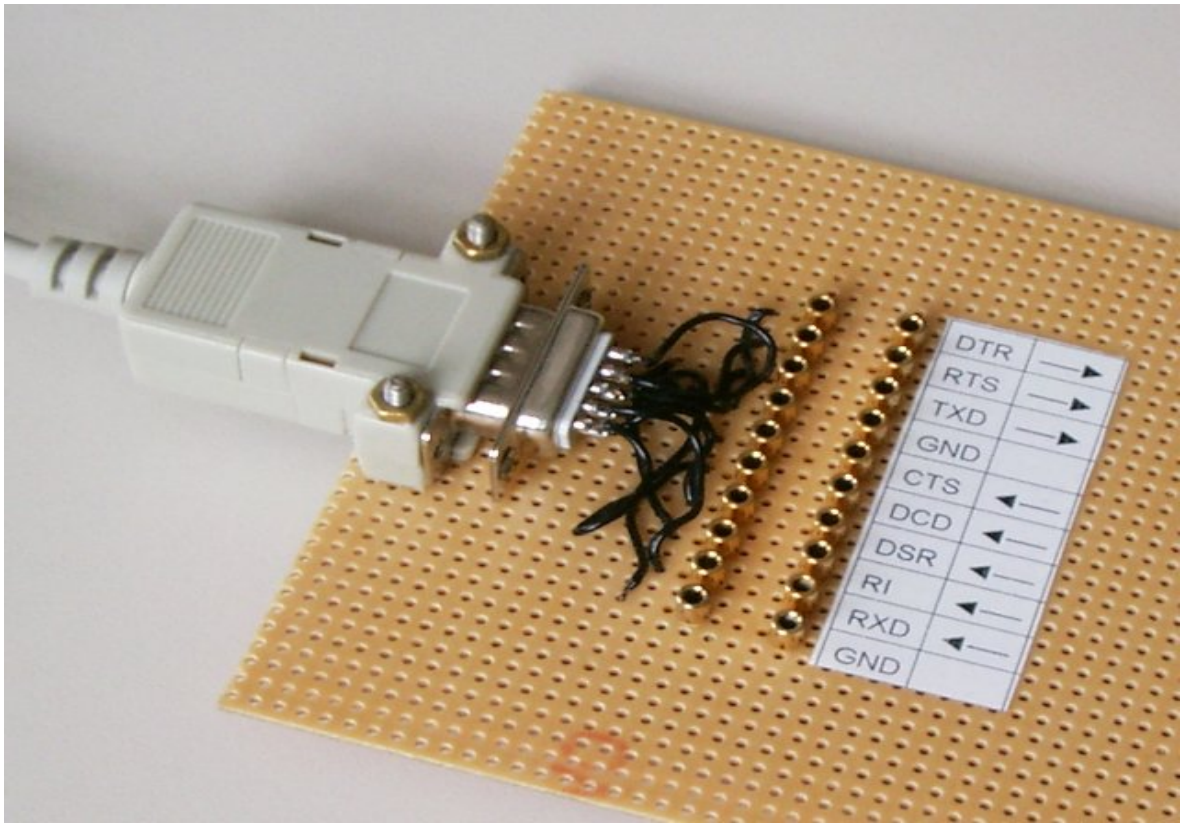
http://www.b-kainka.de/
Von B. Kainka ist das Buch: Messen, Steuern, Regeln unter Windows, Franzis Verlag. Es enthält eine CD mit Versuchen, die Programme in VB und den Treiber RSLine32.DLL.

Geräte:

Platine mit Streifenraster, 5.08 mm Rastermaß
20 Stück Einbaubuchsen, 2 mm
Sub-D Verlängerung 1:1, 9-pol
Sub-D Buchse, 9-pol

Das Bauteil:

Die Buchsen in die Löcher stecken und mit den Leiterbahnen verlöten. Danach werden die Leiterbahnen zwischen den beiden Buchsenreihen unterbrochen.



Die RS232 Schnittstelle

Die Ausgänge der seriellen Schnittstelle

DTR	data terminal ready	Endgerät bereit
RTS	request to send	Sendeteil einschalten
TXD	transmit data	Sendedaten

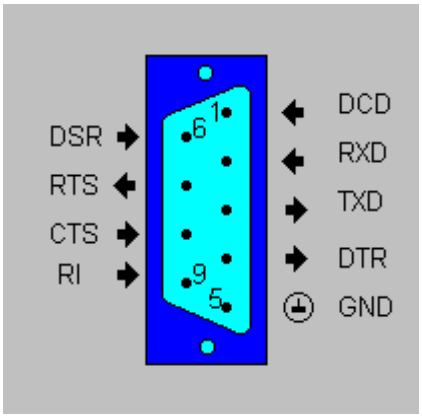
Die Eingänge der seriellen Schnittstelle

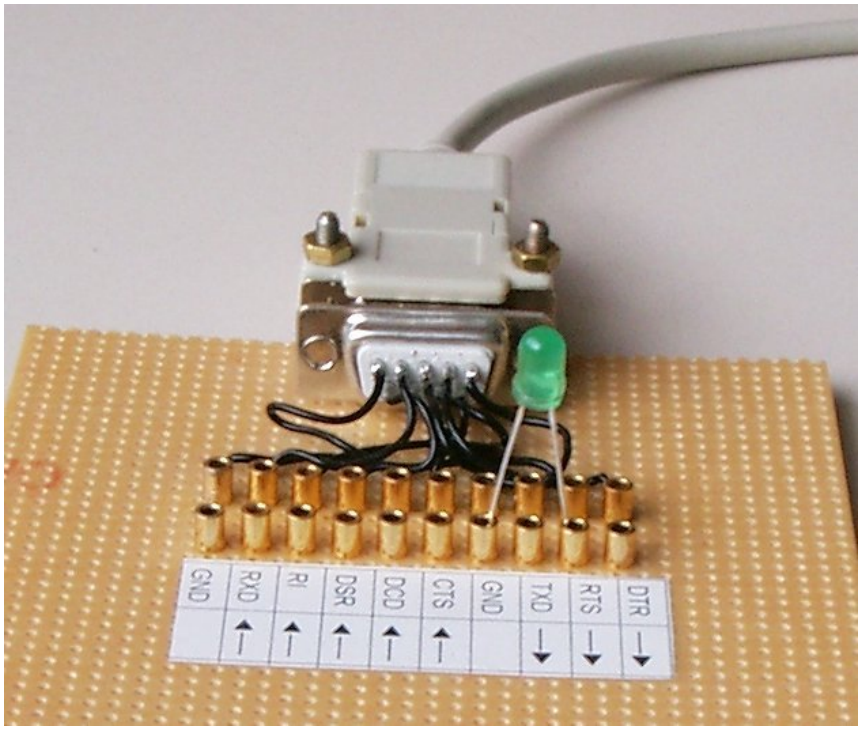
CTS	clear to send	Sendebereitschaft
DCD	data carrier detect	Empfangssignalpegel
DSR	data set ready	Betriebsbereitschaft
RI	ring indicator	ankommender Ruf
RXD (sollte man nicht verwenden)	receive data	Empfangsdaten

GND	ground	Masse

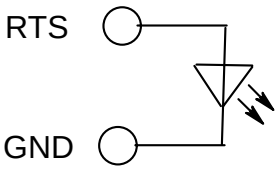
Jede Ausgangsleitung liefert eine Leerlaufspannung von +11,49 V und einen maximalen Strom von 10 mA...20 mA. Eine interne Strombegrenzung verhindert eine Überlastung auch im Kurzschlussfall.

Die meisten PCs verwenden zwei COM-Schnittstellen COM1 und COM2. Oft ist COM1 mit der Maus belegt. Die Programme suchen die freie COM2 für die Versuche.





Ausgang schalten



Programm Ausgang-Eingang verwenden

Man verbindet eine beliebige Ausgangsleitung und GND mit einer Leuchtdiode (LED). Durch Button klick schaltet man den Ausgang an oder aus. Die LED leuchtet, wenn der Ausgang positive Spannung hat. Ist die LED umgekehrt angeschlossen, leuchtet sie, wenn der Ausgang negative Spannung liefert.

Zur Programmierung der seriellen Schnittstelle wird die Funktionsbibliothek RSLine32.DLL eingesetzt. Sie muss sich im jeweils aktuellen Verzeichnis befinden.

Hinweis:

Eine DLL (Dynamic Link Library) ist aufgebaut wie eine Unit: Sie ist unabhängig von der verwendeten Programmiersprache und wird erst bei Bedarf in den Speicher geladen. Greifen mehrere Programme auf eine DLL zu, so ist diese nur einmal im Speicher vorhanden. Eine Unit wäre dann öfters, d. h. für jedes Programm vorhanden.

Das Programm Ausgang_Eingang

```
function OPENCOM (A: Integer): Integer; stdcall; external 'Rsline32.DLL';
procedure RTS (An: Integer); stdcall; external 'Rsline32.DLL';
procedure DTR (An: Integer); stdcall; external 'Rsline32.DLL';
procedure TXD (An: Integer); stdcall; external 'Rsline32.DLL';
function CTS (): integer; stdcall; external 'Rsline32.DLL';
function DCD (): integer; stdcall; external 'Rsline32.DLL';
function DSR (): integer; stdcall; external 'Rsline32.DLL';
function RI (): integer; stdcall; external 'Rsline32.DLL';

implementation
{$R *.DFM}

procedure TForm1.FormCreate(Sender: TObject);
begin
    if OPENCOM (1) = 0 then OPENCOM (2);  // die COM-Schnittstelle wählen
    DTR (0);     RTS (0);    TXD (0);                 // alle Ausgänge aus
end;

procedure TForm1.Button1Click(Sender: TObject);      // DTR an
begin  DTR (1);   panel1.color:= clLime;     end;

procedure TForm1.Button2Click(Sender: TObject);      // DTR aus
begin  DTR (0);   panel1.color:= clBtnFace;  end;

procedure TForm1.Button3Click(Sender: TObject);      // RTS an
begin  RTS (1);   panel2.color:= clLime;     end;

procedure TForm1.Button4Click(Sender: TObject);      // RTS aus
begin  RTS (0);   panel2.color:= clBtnFace;  end;

procedure TForm1.Button5Click(Sender: TObject);      // TXD an
begin  TXD (1);   panel3.color:= clLime;     end;

procedure TForm1.Button6Click(Sender: TObject);      // TXD aus
begin  TXD (0);   panel3.color:= clBtnFace;  end;

procedure TForm1.Timer1Timer(Sender: TObject);
begin
    if CTS  = 1 then panel4.color := clLime else panel4.color := clBtnFace;
    if DCD = 1 then panel5.color := clLime else panel5.color := clBtnFace;
    if DSR  = 1 then panel6.color := clLime else panel6.color := clBtnFace;
    if RI     = 1 then panel7.color := clLime else panel7.color := clBtnFace;
    if RXD = 1 then panel8.color := clLime else panel8.color := clBtnFace;
end;

procedure TForm1.FormClose(Sender: TObject; var Action: TCloseAction);
begin
     DTR (0);     RTS (0);    TXD (0);                      // alle Ausgänge aus
end;
```

Halbleiter

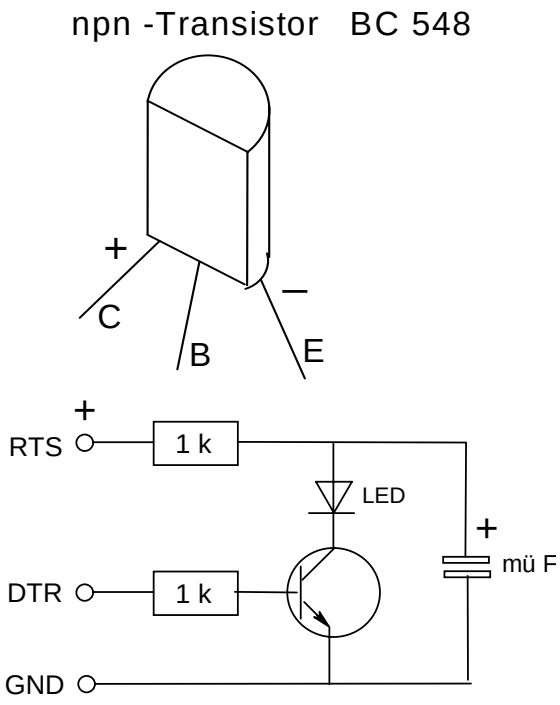
Diode: Leuchtdiode, Fotodiode

Transistor

Blitzlicht

Ein Kondensator kann elektrische Ladung speichern und kurzzeitig als einen stärkeren Strom abgeben.
Über das Programm (Ausgang-Eingang) schaltet man RTS an. Der Kondensator wird aufgeladen. Schaltet man jetzt DTR ein, so entsteht ein kurzer Blitz. Der Transistor wird schlagartig über die LED entladen.

DTR muss ausgeschaltet werden, bevor der Kondensator neu geladen wird.
Man schaltet RTS ein und wieder aus.
Der Kondensator ist jetzt geladen und behält seine Ladung mehrere Minuten lang.



Programm Ausgang-Eingang verwenden

Die Register der RS232 Schnittstelle

Im Gerätemanager von WINDOWS findet man für die Schnittstelle COM1 die Angabe
Interrupt : 04
E/A-Bereich : 03F8 ... 03FF // 8 Werte
Datenbit : 8

Das heißt :
1) die Schnittstelle COM1 hat die Adresse 03F8,
2) die Schnittstelle hat 8 Register mit dem Offset 0 ... 7,
3) jedes Register ist 8 Bit breit.

Nach ihrer Funktion werden die Register verschieden benannt.
In den Registern haben die einzelnen Bits verschiedene Aufgaben: sie steuern die Leitungen DTR, RTS, usw.

Sende-Halte-Register

Offset: 0								
Bit Nr.	7	6	5	4	3	2	1	0

Interrupt-Freigabe-Register

Offset: 1								

Interrupt-Erkennungs-Register

Offset: 2								

Leitung-Steuer-Register

Offset: 3		TXD						

Modem-Steuer-Register (Ausgang setzen)

Offset: 4							RTS	DTR

Leitungs-Status-Register

Offset: 5								

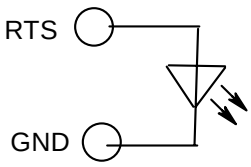
Modem-Status-Register (Eingang lesen)

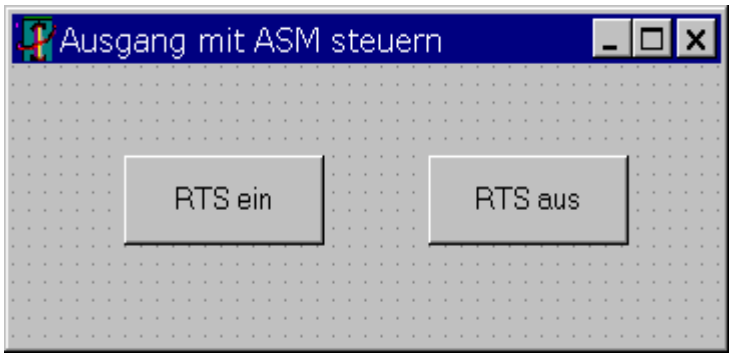
Offset: 6	DCD	RI	DSR	CTS				
Bit Nr.	7	6	5	4	3	2	1	0

Offset: 7								

Assembler-Programmierung

Man verbindet RTS und GND mit einer Leuchtdiode (LED). Mit dem Programm (Ausgang_mit_ASM) schaltet man den Ausgang RTS ein oder aus. Das Programm ist in Assembler geschrieben.



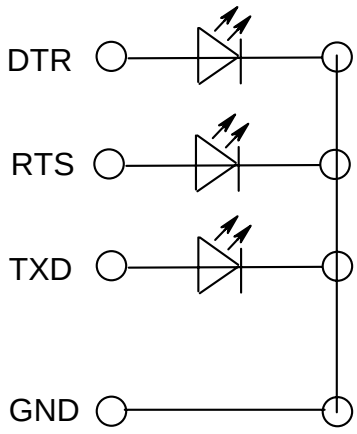


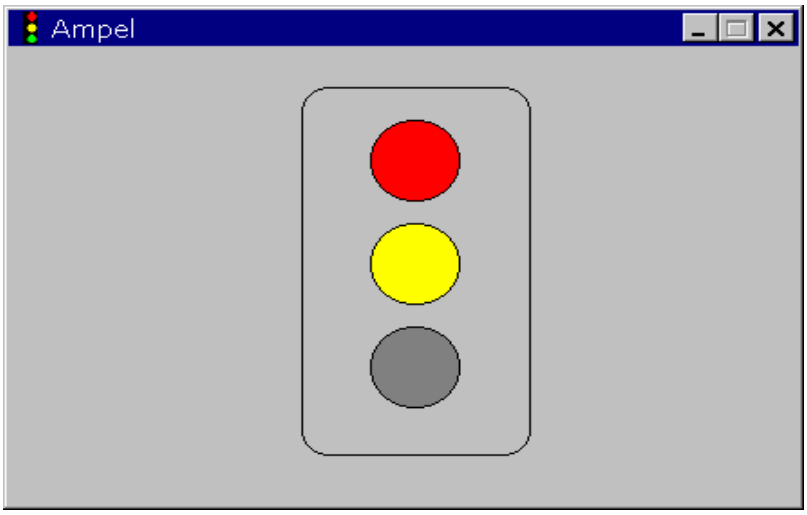
Programm Ausgang_mit_ASM verwenden

```
procedure TForm1.FormCreate(Sender: TObject);
begin
     if OPENCOM (1) = 0 then openCom (2);
     DTR (0);     RTS (0);    TXD (0);                // alle Ausgänge aus
end;

procedure TForm1.Button1Click(Sender: TObject);    // RTS ein
begin
    asm
       push ax                 // AX-Register vom Prozessor sichern
       push DX                 // DX-Register vom Prozessor sichern
       mov DX, 03F8h +4    // die serielle Schnittstelle und das 4.Register auswählen
       mov al, 2               // bin 0000.0010 ; RTS ist das 2. bit
       out DX, al              // RTS einschalten
       pop DX                  // den Wert wieder herstellen
       pop ax
    end
end;

procedure TForm1.Button2Click(Sender: TObject);    // RTS aus
begin
    asm
       push ax
       push DX
       mov DX, 03F8h +4    // die serielle Schnittstelle und das 4.Register
       mov al, 0               // bin 0000.0000
       out DX, al              // RTS ausschalten
       pop DX
       pop ax
    end
end;
```

Ampelschaltung

Man baut die Schaltung nach Plan auf. Im Programm (Ampel) steuert ein Timer die Ausgänge der Schnittstelle und die Farben der drei Shape-Komponenten.



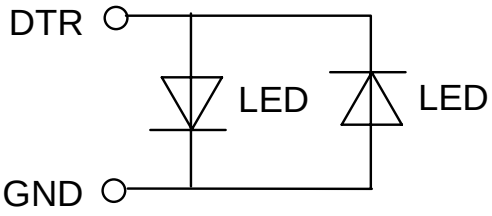


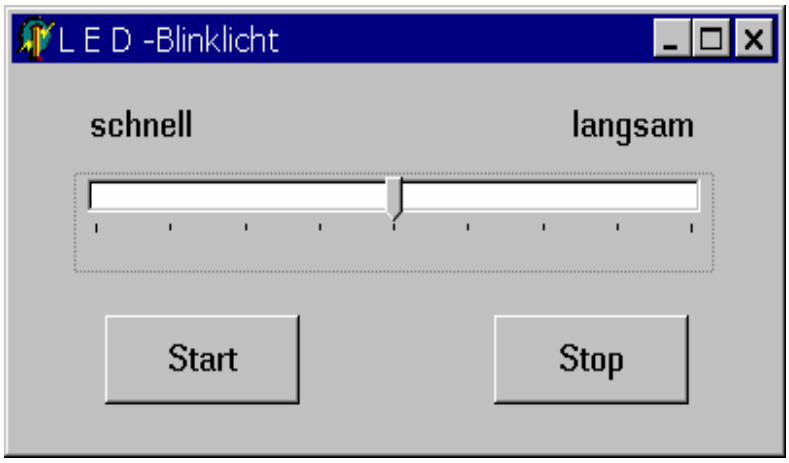
Programm Ampel verwenden

```
procedure TForm1.FormCreate(Sender: TObject);
var i: byte;
begin
     if OPENCOM (1) = 0 then  openCom (2);
     RTS (0);       TXD (0);       DTR (0);                    // alle Ausgänge aus
     oft:= 0;
end;

procedure TForm1.Timer1Timer(Sender: TObject);
begin
    case oft of
       0: begin
            shape1.Brush.Color:=clred;    shape2.Brush.Color:=clgray;
            shape3.Brush.Color:=clgray;
            dtr(1);              rts(0);              txd(0);
            inc(oft);
          end;
       1: begin
            shape1.Brush.Color:=clred;    shape2.Brush.Color:=clyellow;
            shape3.Brush.Color:=clgray;
            dtr(1);              rts(1);              txd(0);
            inc(oft);
          end;
       2: begin
            shape1.Brush.Color:=clgray;  shape2.Brush.Color:=clgray;
            shape3.Brush.Color:=cllime;
            dtr(0);              rts(0);              txd(1);
            oft:= 0;
         end;
    end;
end;
```

Wechsel-Blinklicht

Man verbindet DTR und Masse (GND) mit zwei LEDs, wie in der Skizze angegeben. Die LEDs sind verschieden gepolt eingesetzt, also leuchten sie auch bei unterschiedlicher Stromrichtung. Das Programm (LED-Blinklicht) schaltet DTR abwechselnd ein- und aus. Eine TrackBar (Win32-Komponente) regelt den Timer und so die Blinkfrequenz. Man kann das Blinken abschalten und neu starten.



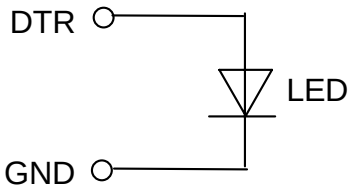


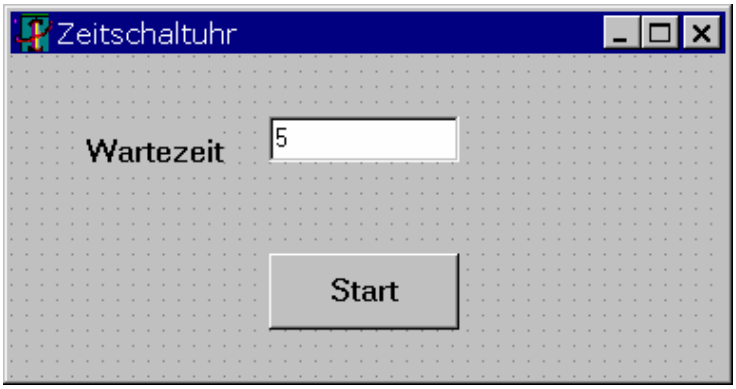
Programm LED-Blinklicht verwenden

```
procedure TForm1.Timer1Timer(Sender: TObject);
begin
    DTR (schalten);                  // DTR ein
    schalten:= not schalten and 1;   // NOT schalten AND 1  invertiert den Wert
end;

   // var schalten: byte = 0 ;       //  ist global definiert
```

Zeitschaltung

Man baut eine einfache Zeitschaltuhr. Es wird die gewünschte Zeit eingeben und die Uhr gestartet. Mit Ablauf der eingestellten Zeit geht die angeschlossene LED an.





Programm Zeituhr verwenden

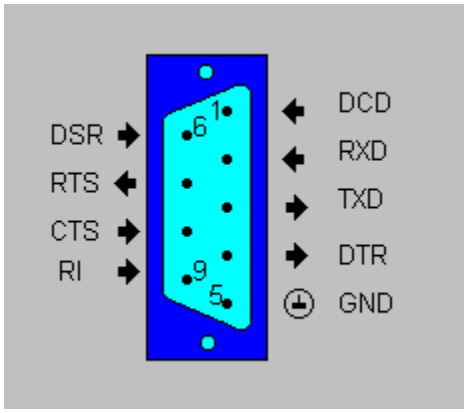
```
procedure TForm1.FormCreate(Sender: TObject);
begin
    if OPENCOM (1)= 0 then openCom (2);
    DTR (0);        RTS (0);        TXD (0);
end;

procedure TForm1.Button1Click(Sender: TObject);    // start
var zeit, zeit1: integer;
begin
    DTR (0);
    zeit := strToint (edit1.text);                 // so lange ist die Wartezeit
    zeit1:= GetTickCount ;                         // Startzeit
    repeat;
        application.processMessages;
    until (getTickCount -zeit1 > zeit );
    DTR (1);
end;

procedure TForm1.FormClose(Sender: TObject; var Action: TCloseAction);
begin
    DTR (0);      RTS (0);       TXD (0);
end;
```

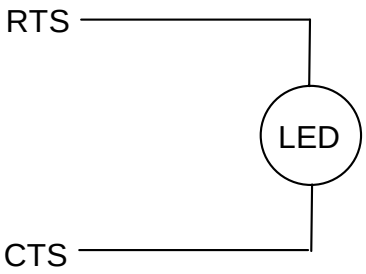
Die Eingänge der RS232 Schnittstelle

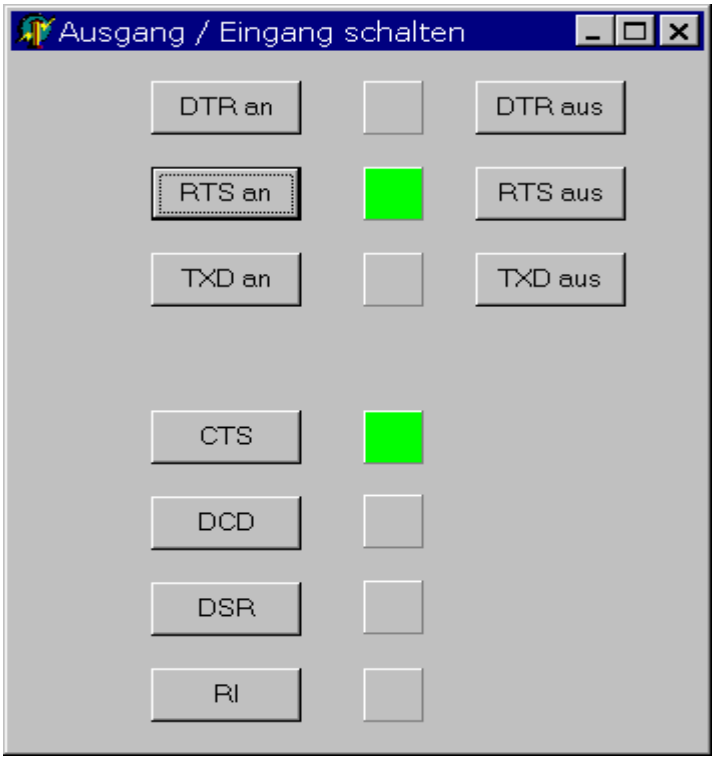
CTS	clear to send	Sendebereitschaft
DCD	data carrier detect	Empfangssignalpegel
DSR	data set ready	Betriebsbereitschaft
RI	ring indicator	ankommender Ruf
RXD	receive data	Empfangsdaten

Alle Eingangsleitungen sind für Anschlussspannungen bis +/-12V ausgelegt und für die direkte Verbindung mit den Ausgängen der RS232-Schnittstelle vorgesehen.
Bei vielen Versuchen verbindet man daher eine Ausgangs- mit einer Eingangsleitung und fragt ihren Zustand ab. Eine positive Spannung oberhalb ca. 1,25 V wird als logische Eins erkannt. Eine Spannung unter ca. 1 V erscheint als logische Null . Ein offener Eingang erscheint als Null.

Das Programm Ausgang-Eingang schaltet die Ausgangsleitung ein, und fragt den Zustand der Eingangsleitung ab. Ist eine Ausgangsleitung über die Lampe mit einer Eingangsleitung verbunden, so erscheinen beide Symbole grün.



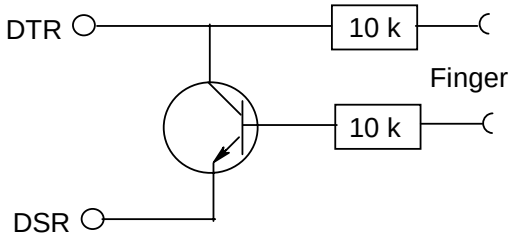




Programm Ausgang-Eingang verwenden

Berührungssensor

Die Leitung DTR hat Spannung. Über die Finger wird ein sehr schwacher Strom von ca. 50 Mikroampere an die Basis weitergeleitet. Der Transistor schaltet durch, er verstärkt den Strom und schaltet die Eingangsleitung DSR ein. Das Signalplättchen wird grün gefärbt.

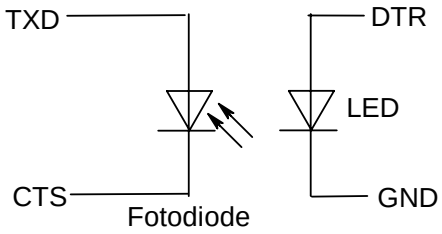


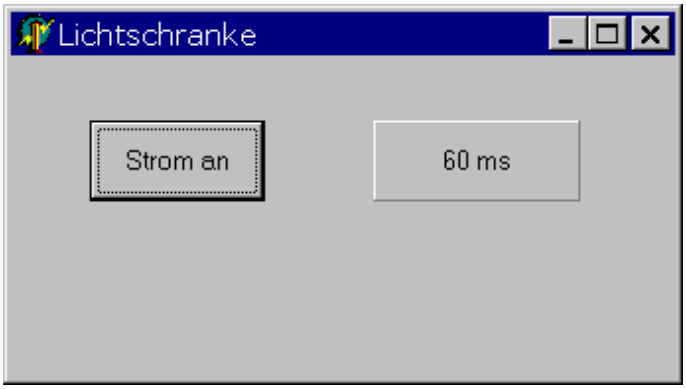
Programm Ausgang-Eingang verwenden

Lichtschranke 1 (Zeit messen)

Bei geringer Helligkeit ist der Widerstand der Fotodiode groß. Es fließt ein schwacher Strom und die Eingangsleitung wird als (AUS) erkannt.
Bei großer Helligkeit ist der Widerstand der Fotodiode klein, es fließt ein starker Strom und die Eingangsleitung wird als (EIN) erkannt.

Die Fotodiode dient als Eingabeelement. Der PC unterscheidet zwischen hell und dunkel.





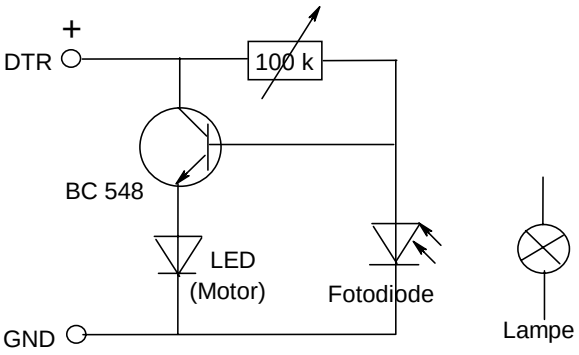
Programm Lichtschranke verwenden

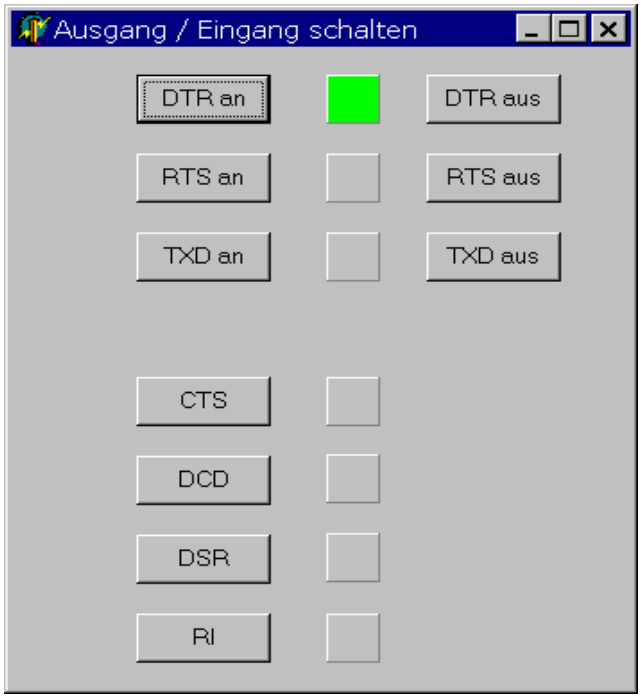
Aufgabe

1) Die Schaltung aufbauen und das Verhalten der Fotodiode überprüfen. Sie arbeitet richtig, wenn das kurze Drahtstück am (+) Pol angeschlossen ist.
2) Zwischen Fotodiode und Lampe zieht man ein Blatt Papier vorbei. Das Programm misst die Dunkelzeit. Aus der Zeit und der Blattgröße berechnet man die Geschwindigkeit der Bewegung.

```
procedure TForm1.FormCreate(Sender: TObject);
begin
    if OPENCOM (1) =0 then openCom (2);     DTR (0);  RTS (0);  TXD (0);
end;

procedure TForm1.Button1Click(Sender: TObject);          // Strom an
begin
    DTR (1);          // Ausgangsleitung (DTR an
    TXD (1);          // Ausgangsleitung (TXD) an
    zeit:= 0;         // eine globale Variable
end;

procedure TForm1.Button2Click(Sender: TObject);          // Strom aus
begin    DTR (0);  RTS (0); TXD (0);       end;

procedure TForm1.Timer1Timer (Sender: TObject);          // Timerintervall = 1 ms
begin
   if CTS = 0 then begin                                 // messen, wenn die
      inc (zeit);                                        // Lichtschranke dunkel ist
      panel1.caption := intTostr (zeit)+ ' ms';
   end
   else zeit:= 0;
end;
```

Lichtschranke 2 (Rolltreppe)





Programm Ausgang-Eingang verwenden

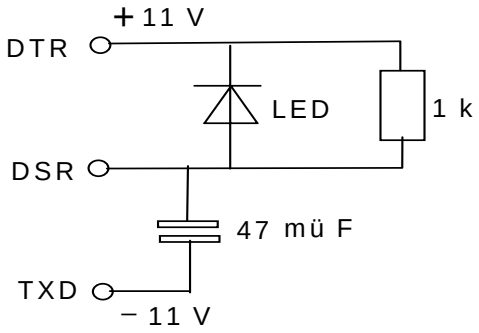
Unterbricht man den Lichtweg zwischen Lampe und Fotodiode, so leuchtet die LED auf, bzw. der Motor läuft an. Diese Schaltung ist an Rolltreppen oder automatischen Türöffnern zu finden.

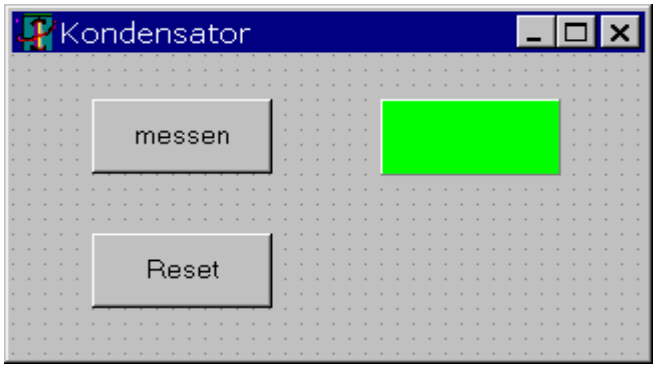
Beachte:

Die Fotodiode wird mit dem kurzen Draht an den (+) Pol angeschlossen, die LED mit dem kurzen Draht an den (-) Pol.

Ladezeit für einen Kondensator

Man lädt ein Kondensator (47 µF) über einen Widerstand (1 k) auf. Die Ladezeit hängt von der Kapazität des Kondensators und von der Größe des Widerstand ab. Die Ladezeit wird vom Programm gemessen und auf der grünen Schaltfläche angezeigt. Nach dem Ladevorgang wird die Ausgangsleitung (DTR) auf NULL gesetzt und der Kondensator über die Diode entladen.
Man wiederholt den Versuch mit einem Kondensator von (500 µF).



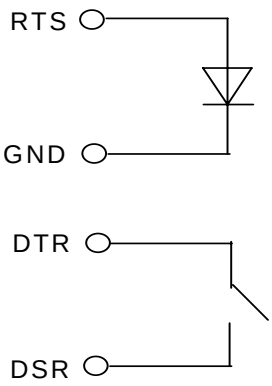


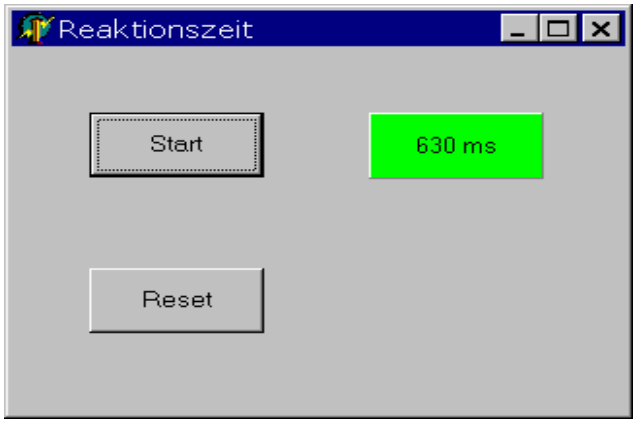
Programm Messen verwenden

```
procedure TForm1.FormCreate(Sender: TObject);
begin
    if OPENCOM (1)= 0 then openCom (2);
    DTR (0);    RTS (0);    TXD (0);
end;

procedure TForm1.Button1Click(Sender: TObject);    // messen
var zeit1, zeit2: integer;
begin
    DTR (1);                                       // Ausgang ein, Kondensator wird
                                                   // geladen
    zeit1:= getTickCount;
    While DSR =0 do begin
       zeit2:= getTickCount;
    end;
    panel1.caption := intTostr (zeit2- zeit1)+ ' ms';
    DTR (0);                                       // der Kondensator wird entladen
end;

procedure TForm1.Button2Click(Sender: TObject);
begin
    panel1.caption:= ' ';
end;
```

Reaktionszeit

Messen der Reaktionszeit mit dem PC. Das Programm schaltet nach zufälliger Wartezeit die LED ein. Man muss dann so schnell wie möglich den angeschlossenen Tastschalter drücken. Die Reaktionszeit wird gemessen. Typische Werte liegen bei 200 Millisekunden (200 ms = 0,2 s).

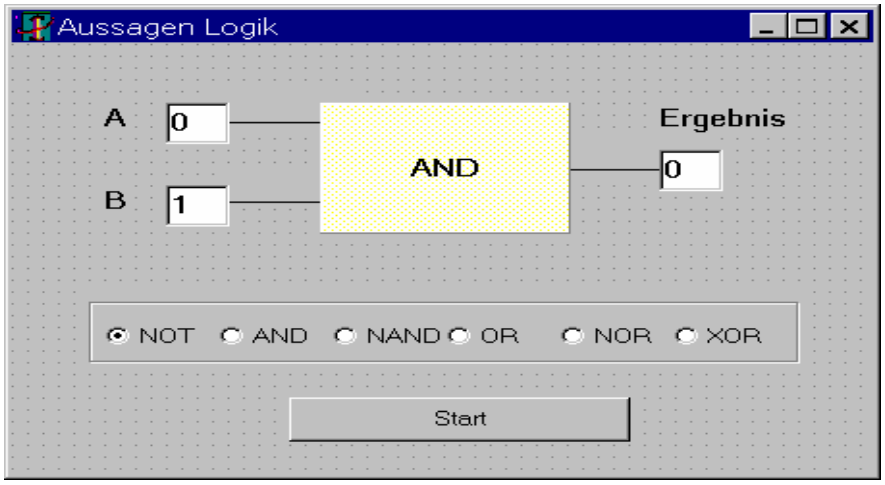




Programm Reaktionszeit verwenden

```
procedure TForm1.FormCreate(Sender: TObject);
begin
    if openCom (1) = 0 then openCom (2);
    DTR (0);   RTS (0);   TD (0);
end;

procedure TForm1.Button1Click(Sender: TObject);        // Start
var zeit1, zeit2, zufallZeit, warten: integer;
begin
    panel1.caption:= ' ';
    randomize;
    ZufallZeit := (random (5)+1) * 1000;               // Werte von 1 bis 5 sek
    zeit1:= getTickCount;
    repeat
        zeit2:= getTickCount;
    until zeit2- zeit1 >= zufallZeit;
    RTS (1);                                           // RTS ein, LED leuchtet
    DTR (1);                                           // DTR ein, Leitung zum Taster
    if DSR () = 0 then begin
        zeit1:= getTickCount;
        repeat
            zeit2:= getTickCount;
        until DSR () = 1;
        warten := zeit2- zeit1;
    end;
    panel1.caption := intTostr (warten) + ' ms';
    RTS (0);    DTR (0);                               // Leitung aus
end;
```

Aussagen-Logik



Programm Logik verwenden

Eingang		logische Verknüpfung					
A	B	NOT (A)	AND	NAND	OR	NOR	EXOR
0	0	1	0	1	0	1	0
1	0	0	0	1	1	0	1
0	1	1	0	1	1	0	1
1	1	0	1	0	1	0	0
		(Not)	(Reihenschaltung)		(Parallelschaltung)		

Aufgabe

1) Man gibt in die Edit-Fenster Zahlenwerte ein und wählt eine logischen Verknüpfung.
2) Bei <Start> wird das Ergebnis ausgegeben.

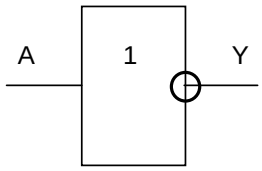
Methoden

logische (bitweise) Verknüpfung

```
procedure TForm1.Button1Click(Sender: TObject);
var a, b, c: byte;
begin
     A:= strToint (edit1.text);  B:= strToint (edit2.text);
    case radioGroup1.Itemindex of
       0: C:= not A AND 1;            // NOT
                                      // NOT invertiert alle Bits, es entsteht ein negativer Wert
                                      // + 1 dazu addiert ergibt das logische NOT
       1: C:= A AND B;                      // AND
       2: C:= not (A AND B) AND 1;          // NAND
       3: C:= A or B;                       // OR
       4: C:= not (A or B) AND 1;           // NOR
       5: C:= A xor B;                      // EXOR
    end;
    panel1.caption:= radioGroup1.items[radioGroup1.itemIndex];
    edit3.text:= intTostr (C);
end;
end.
```

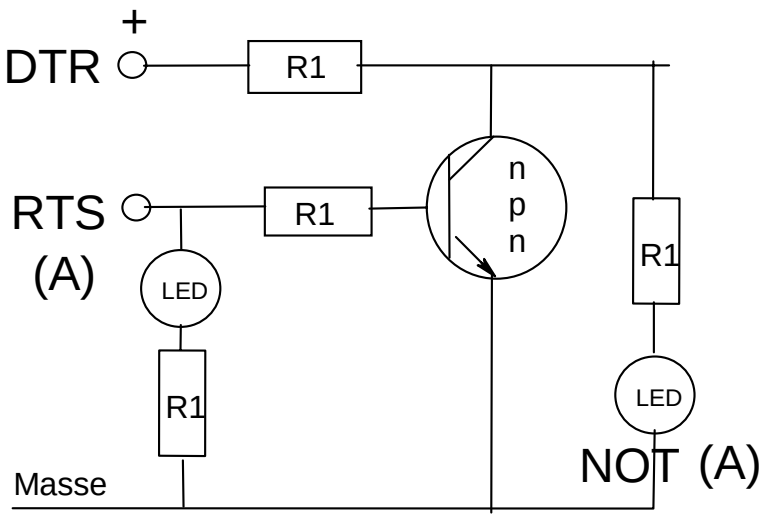
Gatter

NOT



BC548 (NPN)

R1= 10 k
U=11,5 V

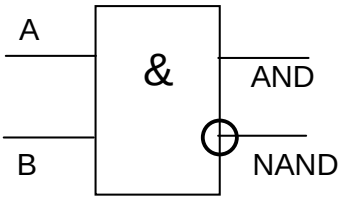


Programm Ausgang-Eingang verwenden.

Aufgabe

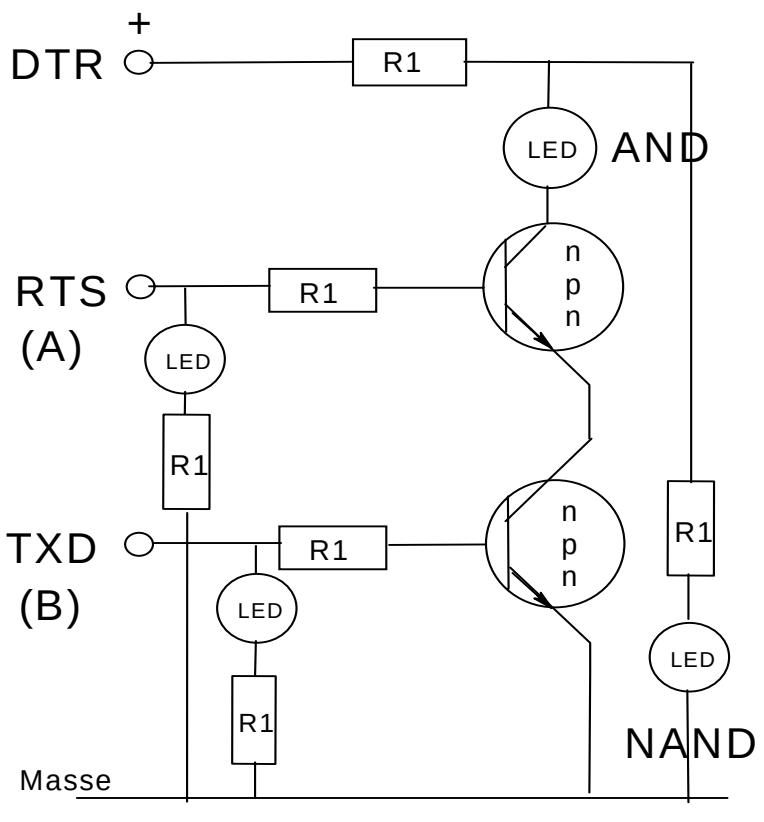
1) DTR ist an und bildet die Stromversorgung für den Transistor.
2) Die Leitung RTS ein- und ausschalten.
3) Der Ausgang Y zeigt entgegengesetzte Spannung wie DTR.

AND / NAND

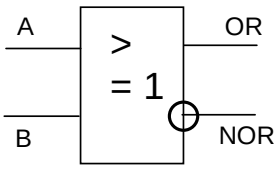
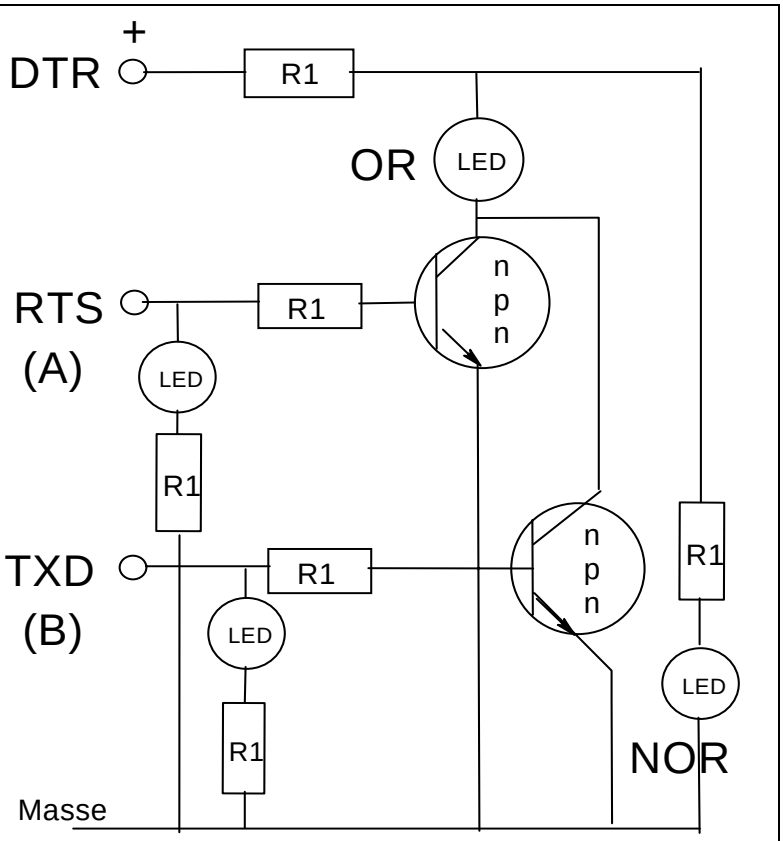


BC548 (NPN)

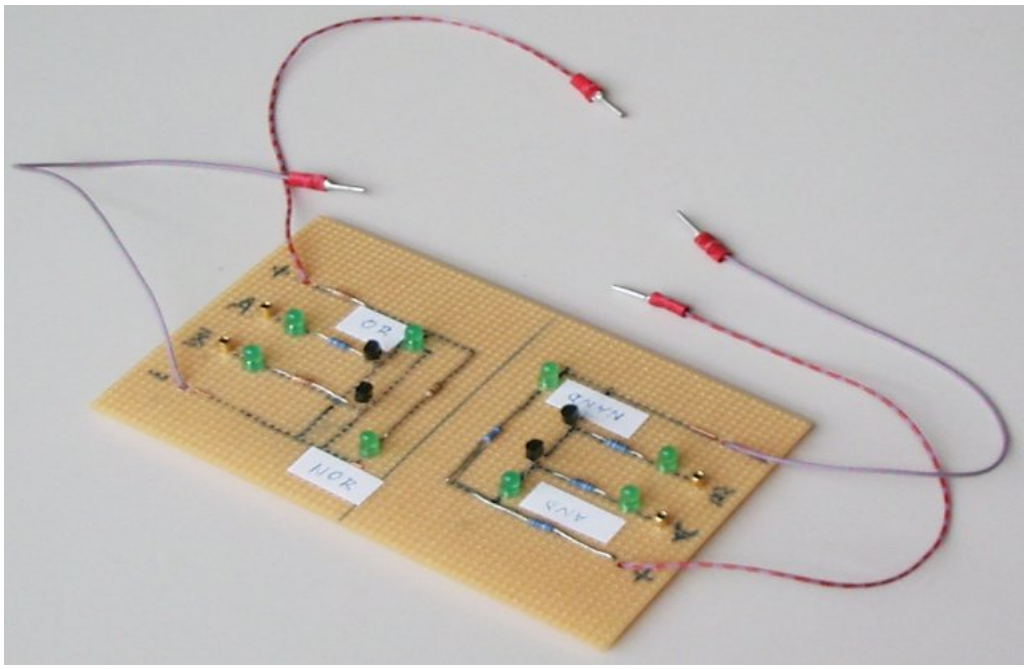
R1= 10 k
U= 11,5 V



Programm Ausgang-Eingang verwenden.

OR / NOR	 BC548 (NPN) R1= 10 k U= 11,5 V	

Programm Ausgang-Eingang verwenden.



Anschrift des Autors: Dr. Helmut Leimeister, Freiherr-vom-Stein-Gymnasium, 57518 Betzdorf