Physikpraktikum
Sommersemester 2003
Protokoll für Versuch 4.1

Georg Kuschk,
Bert Wesarg,
Gruppe 12

02.06.2003

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Programmbausteine für die IEEE-488 Schnittstelle

1.1 Globale Variablen

Die Adressen der Spannungsquelle und des Multimeters an der IEEE-488 Schnittstelle.

```
const supply = 3;       (* Adresse der Spannungsquelle *)
const multi  = 20;      (* Adresse des Multimeters *)
```

1.2 Initialisierung der Geräte

Die Funktion *init* initialisiert beide Geräte an der IEEE-488 Schnittstelle.

```
function init : integer;
var  status : integer;
     value  : IB_Str;
 begin

  (* Initialisierung der IEEE Schnittstelle *)
  IB_Init;

  (* Spannungsquelle initialisieren *)
  Talk_to(supply, '*RST;*CLS;OUTP ON', Status);
  Talk_to(supply, '*OPC?', Status);
  Listen_from(supply, Value, Status);

  (* Multimeter initialisieren *)
  Talk_to(multi, '*RST;*CLS', Status);
  Talk_to(multi, '*OPC?', Status);
  Listen_from(multi, Value, Status);

  init := status;
 end;
```

1.3 Setzen einer Spannung an der Spannungsquelle

Die Funktion *set_u()* setzt an der Spannungsquelle die als Parameter übergebene Spannung *volt* an.

```
function set_u(volt : real) : integer;
var    ustr   : IB_Str;
       status : integer;
       value  : IB_Str;
 begin

  (* erzeugt einen string aus dem real Wert *)
  str(volt, ustr);

  (* sendet Anfrage an Spannungsquelle *)
  Talk_to(supply, 'APPL P6V, ' + ustr + ', 0.1', status);

  if status = 0 then
   begin
    (* fragt Spannungsquelle, ob letzter Befehl ausgefhrt wurde *)
    Talk_to(supply, '*OPC?', status);
    Listen_from(supply, Value, status);
   end;

  set_u := status;
 end;
```

1.4 Spannung am Multimeter auslesen

Die Funktion *get_u()* liest vom Multimeter die aktuelle Spannung aus, und weißt dem Parameter *volt* diesen Wert zu.

```
function get_u(var volt : real) : integer;
var  status : integer;
     value  : IB_str;
     code   : integer;
     valuereal : real;
begin
```

```
 (* fragt Multimeter nach aktuellem Spannungswert, *)
 (* in einem Mebereich von 10V und maximaler Auflsung*)
 Talk_to(multi, 'MEAS:VOLT:DC? 10V,MAX', status);

 (* liest Spannung vom Multimeter aus *)
 if status = 0 then
   Listen_from(multi, value, status);

 (* erzeugt real wert aus string *)
 val(copy(value,1, length(value)-1), valuereal, code);

 if (code <> 0) then
   writeln('realerror: ', value,' code: ', code);

 volt := valuereal;

 get_u:=status;
end;
```

2 Automatisierte Kennlinienaufnahme

Die Funktion *meas_range()* mißt im übergebenen Spannungsintervall *low-high*, mit Schrittweite *step*, die anliegende Spannung am Multimeter. Und schreibt die Wertepaare zeilenweise auf den Bildschirm und in die Datei *name*.

```
procedure meas_range(low : real; high : real; step : real ;name : String);
var f     : text;
    i     : integer;
    u1str, u2str : IB_Str;
    u2real : real;
begin

 if (step = 0.0) then exit;

 assign(f, name);
 rewrite(f);
```

```
  while abs(low) <= abs(high) do
   begin
    if set_u(low) = 0 then
     begin
      if get_u(u2real) = 0 then
       begin

        str(low:5:4,    u1str);
        str(u2real:5:4, u2str);
        writeln('U1: ', u1str, ',  U2: ', u2str);

        writeln(f, u1str, ' ', u2str);

       end; (* get_u *)
     end; (* set_u *)

     low:=low+step;
   end; (* while *)

 close(f);
end;
```

3 Programm zur automatisierten Kennlinienaufnahme

Das Hauptprogramm fragt den Nutzer nach dem Dateinamen ab und startet die automatisierte Kennlinienaufnahme im Intervall $0 - 5\ V$ mit einer Schrittweite von $0.1\ V$.

```
begin
  clrscr;

  write('Dateiname : ');
  readln(fname);
  writeln;

  if init=0 then
```

```
   meas_range(0.0, 5.0, 0.1, copy(fname,1,8)+'.dat');

  writeln;
  writeln('Messung abgeschlossen.');

  readln;

end.
```

4 Test des Programms

Da wir uns für Versuch 5.2, zur Aufnahme der Übertragungskennlinie von Gattern, entschlossen haben, testen wir unser Programm an einem Vierpol. Als Vierpol haben wir uns für einen Spannungsteiler entschieden. Dies hat mehrerer Vorteile. Zum einen wissen wir, was wir für Messwerte erwarten und zum anderen haben die verwendetet Bausteine eine sehr kleine Messunsicherheit, in dem Bereich, in dem wir später die Gatter messen. Das Schaltbild zu einem Spannungsteiler ist in Abbildung 1 zu sehen. Da wir eine Halbierung der Spannung U_1 wollten, setzten wir die Widerstände R_1 und R_2 gleich, und nutzten ein Potentiometer als Widerstand. Daraus ergab sich das Schaltbild in Abbildung 2.

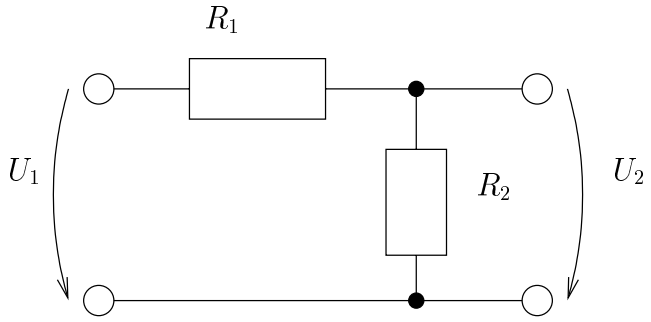

Abbildung 1: Spannungsteiler

$$\frac{U_1}{U_2} = \frac{R_1 + R_2}{R_1}$$

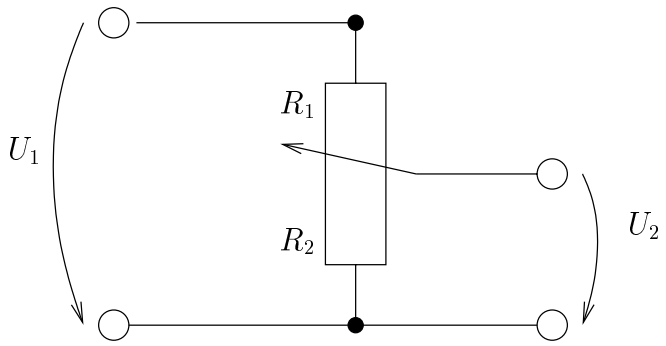

Abbildung 2: Spannungsteiler mit Potentiometer

$$\begin{aligned} \Rightarrow U_2 &= \frac{U_1 \cdot R_1}{R_1 + R_2} && , R_1 = R_2 \\ &= \frac{U_1 \cdot R_1}{2 \cdot R_1} \\ &= \frac{1}{2} U_1 \end{aligned} \tag{1}$$

Als Potentiometerwiderstand verwendetet wir den Baustein A273, dieser hat aht ein Bereich von $10 \times 100\ \Omega$, eine maximale Dauerbelastung von $0,08\ A$ und eine Messungenauigkeit von $0,1\%$. Damit erfüllt dieser Baustein genau unseren Vorstellungen, um unser Programm zu testen.
In der Abbildung 3 sind die Messwerte als $\circ$ dargestellt, und durch eine lineare Regression wurde eine Gerade durch die Messwerte gelegt, mit einem Anstieg von $m = 0.500486$ und einem Offset von $0.0\ V$.

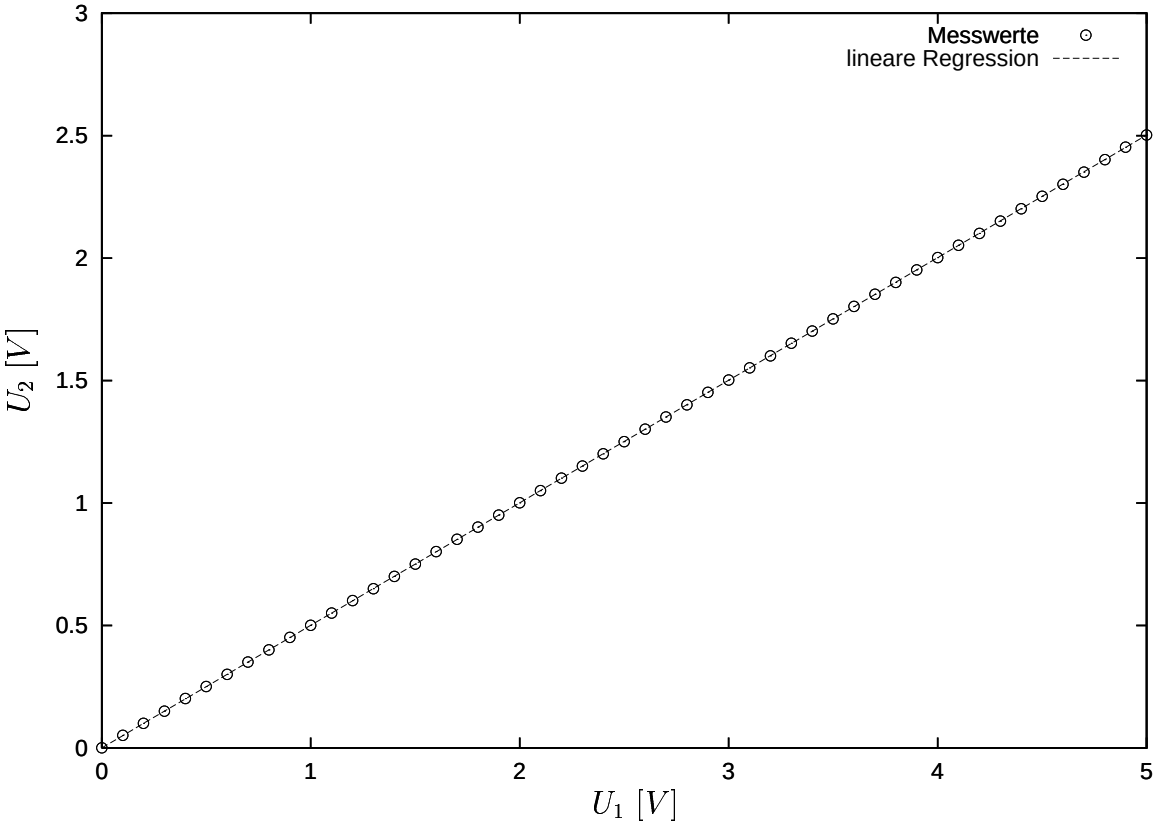

Abbildung 3: Spannungsteiler $U_2 = \frac{1}{2}U_1$