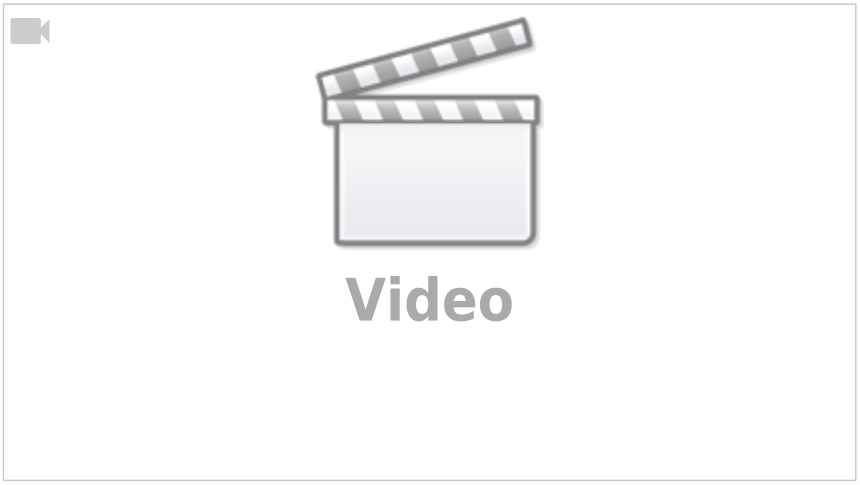
# SWR-Meter Kurzwelle

(siehe auch https://wiki.funkfreun.de/projekte/swr-meter_kurzwelle )

Für den Funkbetrieb auf Kurzwelle und dem Selbstbau von Antennen ist ein Stehwellen-Messgerät erforderlich. Viele Kurzwellen-Transceiver haben dieses zwar eingebaut, aber die Anzeige-Elemente werden oft auch für andere Messwerte benötigt. Spätestens jedoch bei der Verwendung eines Selbstbaugerätes, Senderverstärkers oder eines externen Tuners ist oft ein externes SWR-Meter notwendig.
Billige, fertige Analog-Geräte sind für ca. 50€ zu haben. Ein hochwertigerer Selbstbau auf Basis eines Mikrocontrollers ist jedoch nicht nur weit günstiger, sondern ließe sich um viele weitere Funktionen und Komponenten erweitern, wie z.B. eine serielle Ausgabe von Sendeleistung und SWR-Wert oder das Autotuning einer MagLoop-Antenne.




Das hier vorgestellte SWR-Meter besteht aus zwei Komponenten:

- **SWR-Messbrücke:**
  für verschiedene Leistungen anzupassen, gibt nur zwei analoge Spannungen aus.
- **Arduino Nano mit I²C-Textdisplay:**
  misst am ADC-Eingang die analogen Spannung und gibt die errechneten Werte auf dem Display und seriell aus.

Beide Komponenten sind beliebig austauschbar, so dass sich z.B. die SWR-Messbrücke auch an einem bereits vorhandenem Micocontroller-Projekt verwenden lässt.

## Dateien

Das Zip-Archiv mit Arduino-Code, Bestückungsanleitung, Fritzing-Projekt, Ätzvorlagen und bebilderter Aufbauanleitung ist hier zu finden: hshb-swr-meter.zip (letzte Aktualisierung 03-Jan-2017)

Für die Teile der SWR-Messbrücke gibt es einen Reichelt-Warenkorb.
Dazu wird nur noch ein beliebiger „Arduino Nano V3“, ein Textdisplay mit I2C-Adapter und ein Steckbrett benötigt. Das Gehäuse kann dafür beliebig selbst gebaut werden.
Die gesamten Materialkosten liegen bei ca. 25€

# Messbrücke

**Der Aufbau dieses SWR-Meters basiert auf der "SWR-Messbrücke mit Software-Korrektur der Diodenkennlinie" von Georg Latzel, DL6GL.**
Da unser SWR-Meter jedoch auch mit 5V betrieben werden soll (=Arduino-VCC, USB-Spannung), haben wir ein paar Schaltungsänderungen gemacht:

- Der OpAmp TLC272 wird nur mit 5V betrieben (3.8V Arbeitsbereich an den Eingängen)
- Die 10k-Ohm-Widerstände an den invertierenden Eingängen haben wir entfernt.
  Der Ausgang der beiden OpAmps wird direkt auf den invertierenden Eingang gelegt (=Impedanzwandler). Durch die Impedanzwandlung wird der Messkreis weniger gedämoft, es sind längere Verbindungsleitungen verwendbar und der Arduino-ADC-Eingang bleibt durch Überspannungen geschützt.
- Ergänzen von 100kOhm-„Pulldowns“ als Spannungsteiler an den nicht-invertierenden Eingängen.
  Da der Arbeitsbereich des OpAmps durch die geringere Betriebsspannung eingeschränkt ist, wird die Messspannung (max 5V) halbiert. Da der OpAmp jedoch nicht nur bis 2.5V, sondern bis 3.8V verstärken kann, erhöht sich damit auch die maximal zu messende Leistung um ca. das 1,5fache. Das wird zwar im Arduino-Code nicht weiter umgerechnet, erhöht aber dennoch den maximalen möglichen Leistungsbereich.
- Entfernen des 4,7kOhm-Serienwiderstands am Ausgang. Da die Betriebsspannung halbiert wurde, ist eine Spannungsteilung für den darauf folgenden Microcontroller-Eingang nicht erforderlich und erhöht die Messgenauigkeit des AD-Wandlers.

Der Schaltplan, die Platine und die Bestückung sind hier zusammengefasst: hshb-swr-koppler.pdf
Die bebilderte Anleitung zum Aufbau des Messkopplers ist hier: hshb-swr-meter.pdf (als fehlerhaft markierte Bilder werden noch ersetzt)

# Arduino

Am Arduino sind der SWR-Koppler und das I2C-Textdisplay wie folgt anzuschließen:

| Modul | Pin Name | Arduino Pin |
|---|---|---|
| I2C-Textdisplay | SDA | A4 |
| | SCL | A5 |
| | VCC | 5V |
| | GND | GND |
| SWR-Messbrücke | F | A0 |
| | R | A1 |
| | + | 5V |
| | - | GND |

Der Arduino-Code ist auch im oben verlinkten Zip-Archiv enthalten.
Implentiert wurde ebenfalls die von Georg Latzel, DL6GL beschriebene und für die BAT43 ermittelte Korrektur der Diodenkennlinie (Leseempfehlung!).
Wie immer eine kurze Warnung vorab: ich bin kein Programmierer und es gibt die ein oder andere Stelle, die man noch verbessern könnte. Auch wenn der Code gut funktioniert, bin ich für jeden Tip per Mail an dl2ab@darc.de dankbar - man lernt ja schließlich nie aus ;)

Folgende Parameter sind von jedem selbst noch anzupassen:

- Die I²C-Adresse (und ggf. abweichende Pinbelegung) bei „LiquidCrystal_I2C lcd( …“
- Maximale Leistung (je nach Windungszahl) bei „ int maxPower = … “
- Die Korrektur der Diodenkennlinie bei „ bool diodeCorr = … “ *

Alle anderen Konfigurationsmöglichkeiten sind im Code auch dokumentiert.

Die Ausgabe erfolgt über die FLOAT-Variablen 'powerF' und 'swr'. Wer statt einem Textdisplay lieber eine andere Anzeige benutzen will (z.B. 7Segment, Nixie-Röhren oder zwei 8×8-LED-Bausteine), braucht nur den Code für das Textdisplay rauszuschmeißen und durch seinen eigenen zu ergänzen.

* Die Korrektur der Diodenkennlinie lässt sich mit „bool diodeCorr = 0;“ ausschalten. Damit ist der angezeigte Wert vergleichbar mit konventionellen analogen SWR-Metern, die diese Korrektur auch nicht vornehmen. Der Wert stimmt dadurch mit den Werten überein, die z.B. auch mein „Yaesu FT-990“ anzeigt.
Wie bei allen anderen SWR-Metern ist hier dann aber der Nachteil, dass der SWR-Wert bei unterschiedlichen Leistungen leicht abweichend ist (z.B. -0.15 bei 15W statt 50W), da die Diodenkennlinie sich bemerkbar macht.
Wer diese Korrektur wünscht, kann sie mit „bool diodeCorr = 1;“ vornehmen, verzichtet damit aber auch auf die Vergleichbarkeit mit konventionellen SWR-Metern.
Für die Leistungsmessung wird immer der genauere Kennlinien-korrigierte Wert genommen.

**Arduino-Sketch** SWR-Example.ino (Arduino 1.6.9, Stand 03.01.2017, ist auch im oben verlinkten Zip-Archiv enthalten)

```
//
//    .-------------------------------------------------------------------
-------------------------------------------------------------------------
---------------.
//    |
SWR-Meter Example
|
//    '-------------------------------------------------------------------
-------------------------------------------------------------------------
--------------'
//     by Daniel Wendt-Fröhlich, DL2AB (BugReports/Changes to
danielwf@hackerspace-bremen.de, dl2ab@darc.de) for
//          "Hackerspace Bremen e.V." https://hackerspace-bremen.de / HSHB
Amateur Radio Group http://hshb.de/afu
//     License CC-by-SA 3.0 - 11/2016-01/2017 - Bremen(GER) --
http://creativecommons.org/licenses/by-sa/3.0/de/
//
//
//     Use SWR-circuit from here
http://dl6gl.de/amateurfunk/swr-messbruecke-mit-software-korrektur-der-diode
nkennlinie (without parts left of 33k-Resistor for freq-counter).
//     For 5V-Operation some changes are needed:
//       - Remove 10k-Resistors at OpAmps and connect outputs to inverted
inputs directly
//       - Add 100k to noninverted-inputs of OpAmps
//       - remove inline 4.7k at output, we need only the Pulldown 4,7k-
```

```
Resistor und Poti for fine adjustments.
//
//     NewLiquidCrystal
https://bitbucket.org/fmalpartida/new-liquidcrystal/wiki/Home see website
for authors and license
//
//
//
//     ----------------------------------------------Connections----------
-----------------------------------------------
//        Display SDA: A4 (check/modify 'LiquidCrystal_I2C lcd' for your
used I2C-LCD-Adapter, I2C-Adress, PinOut)
//        Display SCL: A5
//        SWR-Bridge F: A0
//        SWR-Bridge R: A1
//
//
//    .-------------------------------------------------------------------
-------------------------------------------------------------------------
---------------.
//    |                                                          Libraries
and Settings
|
//    '-------------------------------------------------------------------
-------------------------------------------------------------------------
---------------'
//     ----------------------------------------------Display--------------
-----------------------------------------------
#include <Wire.h>
#include <LiquidCrystal_I2C.h>                                    // I2C-
LCD-Library, included in Arduino-IDE
LiquidCrystal_I2C lcd(0x3F, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);       // Set
the LCD I2C address, SainSmartLCD2004 0x3F, maybe try 0x38 or 0x20 or 0x27
bool DisplayRefresh = 1;
unsigned long printMillis = 0;
unsigned long serialMillis = 0;


//     ----------------------------------------------SWR-Meter-Parameters----
-------------------------------------------------


// Configuration
int maxPower = 90;                                    // maximum Power in Watt
(incl. corrections) , maxPower = (5*N/1.414)² * 50ohm | [N=7 windings on
coupler => 10W] [N=9=>20W] [N=14=>50W] [N=20=>100W] [N=28=>200W]
int swrFpin = 0;                                      // SWR-Coupler
Connections
int swrRpin = 1;
bool diodeCorr = 0;                                   // 1 = correction of
diode characteristic
                                                      // 0 = no correction,
```

```
comparable to analog SWR-Meters. only used for SWR, power will be always
corrected.

// Parameters for used Diode, see
http://dl6gl.de/digitales-swr-power-meter/6-messbruecken
float a = 0.25161;                                  // Values for BAT43-
Diode-correction
float b = -0.85798;
float c = 1;

// Values for Calculation
float areadF;                                       // analog readings
float areadR;
float voltageF;                                     // calculated voltage
(peak)
float voltageR;
float calcF;                                        // corrected values
float calcR;
float powerF;                                       // calculated power
float powerPeak;
float swr;                                          // calculated swr

void readMeter(){
  areadF = analogRead(swrFpin);                     // read Voltage
  areadR = analogRead(swrRpin);
  voltageF = areadF * 0.0048828125;                 // measuredVoltage =
analogRead * (5/1024)
  voltageR = areadR * 0.0048828125;

  calcF = pow(voltageF,b);                          // Formular according to
http://dl6gl.de/amateurfunk/swr-messbruecke-mit-software-korrektur-der-diode
nkennlinie
  calcF = calcF * a;                                // CorrectionFactor = a *
( voltage ^ b) + c
  calcF = calcF + c;
  calcF = calcF * voltageF;                         // U_hf = voltage *
CorrectionFactor

  calcR = pow(voltageR,b);
  calcR = calcR * a;
  calcR = calcR + c;
  calcR = calcR * voltageR;

  powerF = (maxPower * 8) * ( sq(calcF) / 50 );     // with maxPower=ADC=2.5V
-> Padc = (2.5V)² / 50 = 0,125 ==> maxPower / 0,125 = maxPower * 8
                                                    // than (maxPower * 8 ) *
Padc ==> (maxPower * 8 ) * (voltageF² / 50ohm)


  if (diodeCorr == 1) swr = (calcF + calcR) / (calcF - calcR);         //
calculation of SWR
```

```
  if (diodeCorr == 0) swr = (areadF + areadR) / (areadF - areadR);
}


//    .-------------------------------------------------------------------
-------------------------------------------------------------------------
---------------.
//    |                                                          SETUP
|
//    '-------------------------------------------------------------------
-------------------------------------------------------------------------
---------------'

void setup() {

  Serial.begin(9600);                              // setup serial
connection

  lcd.begin(16,2);                                 // initialize the 16x2-
lcd, backlight is lit
  lcd.backlight();                                 // switch backlight on
  lcd.setCursor(4,0); lcd.print("Arduino-");       // set display-cursor x,y
and print text
  lcd.setCursor(4,1); lcd.print("SWR-Meter");
  delay(2000);
  lcd.clear();


}

//    .-------------------------------------------------------------------
-------------------------------------------------------------------------
---------------.
//    |                                                          LOOP
|
//    '-------------------------------------------------------------------
-------------------------------------------------------------------------
---------------'


void loop() {

  readMeter();

  if (powerF > powerPeak) powerPeak = powerF;
  if (printMillis < millis() ) {
    lcd.setCursor(0,0);                              // LCD Output 1st line
for Power
    lcd.print("PWR ");                               //
    if (powerPeak<100)lcd.print(" ");                   // clears empty
spaces for smaller values and
```

```
    if (powerPeak<10)lcd.print(" ");                    //  moves value for
fixed position
    lcd.print(powerPeak,0);                             // print power
    lcd.print("W ");

    char wattMeter[7] = "       ";                     // meter for power,
cleared array
    for (byte i=1; i<=6; i++) {
      if (powerPeak > (i * maxPower/7) ) wattMeter[i-1] = (char)0xFF;
    }
    if (powerPeak > maxPower) wattMeter[6] = '!';
    lcd.setCursor(9,0); lcd.print(wattMeter);


    lcd.setCursor(0,1); lcd.print("SWR ");            // LCD Output 2nd line
for SWR
    if (swr < 0.1 ) swr = 0;                            // needed for
disconnected SWR-bridge
    if (swr > 9.99) {lcd.setCursor(4,1); lcd.print(">10!");}            //
if SWR > 10 - usually if SWR-bridge is disconnected
    if (swr <= 9.99) {lcd.setCursor(4,1); lcd.print((float)swr,2);}     //
print SWR-value

    char swrMeter[7] = "       ";                       // meter for swr,
cleared array
    if (swr > 1,2 ) swrMeter[0]=(char)0xFF;
    for (byte i=1;i<=6; i++) {
      if (swr > ((i*0.5)+1)){
        swrMeter[i] = (char)0xFF;
        if (3 <= ((i*0.5)+1)) swrMeter[i] = '!';
      }

    }
    lcd.setCursor(9,1); lcd.print(swrMeter);

    printMillis = millis() + 200;
    powerPeak = 0;

    if (powerF > 0){
      Serial.print(powerF,0);
      Serial.print(" Watt; SWR ");
      Serial.println(swr,2);
    }
  }


}
```

From:
https://wiki.hackerspace-bremen.de/ - **Hackerspace Bremen e.V.**

Permanent link:
**https://wiki.hackerspace-bremen.de/projekte/swr-meter_kurzwelle**

Last update: **2022-11-17 22:34**