Raspberry-Pi-Voltmeter

Mit farbigen Anzeigen

Es werden nur wenige externe Bauelemente am Raspberry Pi benötigt, um Gleichspannungen bis 5 V zu messen und die Ergebnisse auf dem Monitor anschaulich in Farbe darzustellen. Dabei lässt sich die volle Bildschirmgröße nutzen, so dass diese Applikation gut als Basis für Demonstrationen zum Beispiel in Schulen geeignet ist.

Von
Hermann Nieder (D)

Der Autor ließ sich unter anderem von drei Elektor-Beiträgen zum Umstieg von Basic nach Python [1] anregen; also sollte das Programm für den Raspberry Pi (Rev. 2) in Python entworfen werden. Schließlich entstanden zwei Programmversionen. In beiden Fällen kommuniziert ein Raspberry Pi über einige seiner GPIO-Pins mit einem A/D-Wandler-Chip und zeigt dann die Messwerte auf dem angeschlossenen Bildschirm an.

Zusatzschaltung

Die Zusatzschaltung für den Raspberry Pi (**Bild 1**) besteht aus einem mit 8 bit auflösenden Analog-Digital-Wandler TLC549 [2] von Texas Instruments. Dieser Wandler mit seriellem Interface ist mittlerweile mehr als 30 Jahre (!) alt, leicht (natürlich als DIP) erhältlich und ebenso einfach anzuwenden. Der Wandler wurde hier noch mit einem Spannungsteiler (R1...R3) am Analogeingang ausgestattet. In der Testphase hat der Autor dort ein 10-k-Potentiometer angeschlossen und ein Digitalmultimeter als Vergleichsmessgerät eingesetzt.

Mit den beiden Miniaturtastern „HOLD1“ und „HOLD2“ lässt sich während einer Messung ein bestimmter Anzeigewert speichern (was auf der unten beschriebenen Anzeige jeweils auch durch eine Markierung über der Skala veranschaulicht wird). Beide gespeicherten Werte können mit dem Taster „RES“ wieder gelöscht werden, damit zwei andere Werte einschließlich der dazugehörigen Markierungen angezeigt werden können.

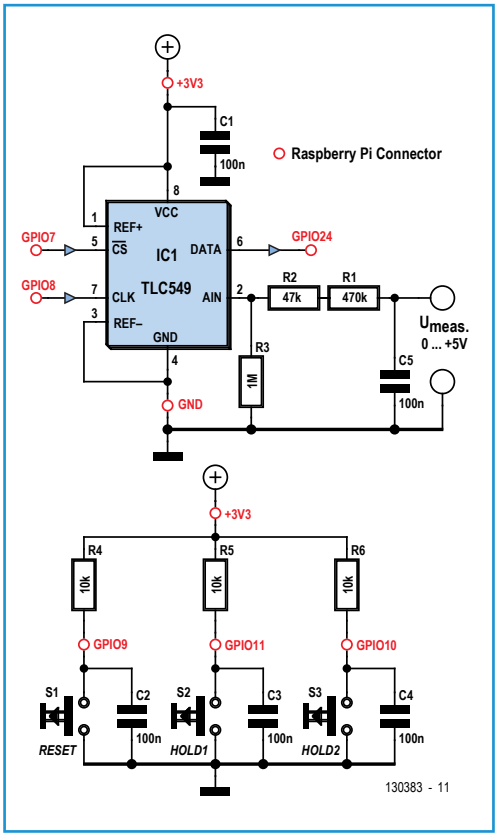

Bild 1. Zusatzschaltung zum Raspberry Pi mit einem „antiken“ AD-Wandler.

Die Zusatzschaltung findet bequem auf einer der vielen im Handel erhältlichen Breadboards für den Raspberry Pi Platz, die auf den Expansion-Header der Mikrocontrollerplatine aufgesteckt werden. Die Versorgungsspannung von 3,3 V wird vom RPi bereitgestellt. Der Messbereich lässt sich durch zusätzliche in Reihe geschaltete Widerstände vor R1 erweitern. In diesem Fall sind beide Python-Programme

an einigen Stellen entsprechend anzupassen. Dies dürfte anhand der Kommentierung der Programmlistings aber nicht allzu schwer sein.

Programmversionen

Bild 2 zeigt einen Screenshot zu einer Variante der ersten Programmversion. Er zeigt drei Messwerte, in der Mitte (größer) der aktuelle. Es lassen sich außerdem mit den zwei Tastern zwei Messwerte als Digitalwert speichern. Zusätzlich zur Digitalanzeige ist eine quasianaloge Bargraphanzeige vorhanden. Die auf Tastendruck gespeicherten (und auch wieder gelöschten) Werte werden als Markierungsstrich über dem Bargraph dargestellt.

Wenn die Messwerte, die man speichern möchte, nicht zu dicht beieinander liegen, kann man die Markierungen mit einer geringfügig abgewandelten ersten Programmversion wie in **Bild 3** als Pfeilspitzen darstellen lassen.

In einer zweiten Programmversion (**Bild 4**) wird ein Voltmeter mit Zeiger simuliert, zusätzlich der Messwert darunter in Ziffern dargestellt. Auch in diesem Fall lassen sich zwei Messwerte speichern, indem man „HOLD1“ oder „HOLD2“ auf der Zusatzplatine mit dem TLC549 betätigt. Diese Messwerte werden außerdem auch über der Skala als Markierungsstriche dargestellt. Die beiden gespeicherten Messwerte und die Markierungen lassen sich genau wie oben bei Bedarf auch wieder löschen. Eine Abwandlung der zweiten Programmversion sieht man in **Bild 5**; ein Voltmeter mit einem Rahmen um das „Messwerk“.

In allen Programmversionen lässt sich beim Start des Programms jeweils eine von mehreren vorgegebenen Farben für den Hintergrund, eine Farbe für den Vordergrund und ein Faktor für die Darstellungsgröße wählen. Auch einen Namen für die Darstellung kann man eingeben.

Ausprobieren

Die Python-Programme des Autors kann man von der Elektor-Website downloaden [3], und zwar am besten mit einem ans Internet angeschlossenen Raspberry Pi selbst.

Legen Sie im Verzeichnis /home/pi nun ein weiteres Verzeichnis an, in das Sie die

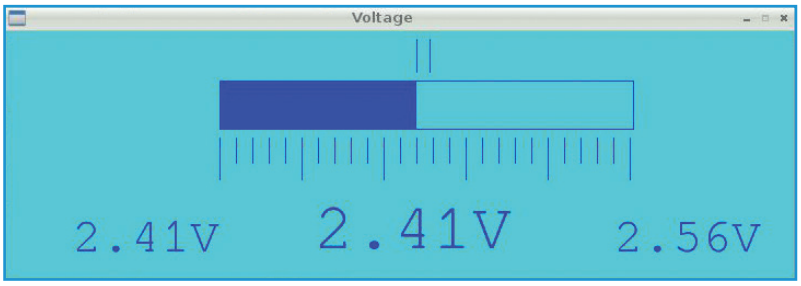

Bild 2. Digitalvoltmeter mit quasianaloger Anzeige.

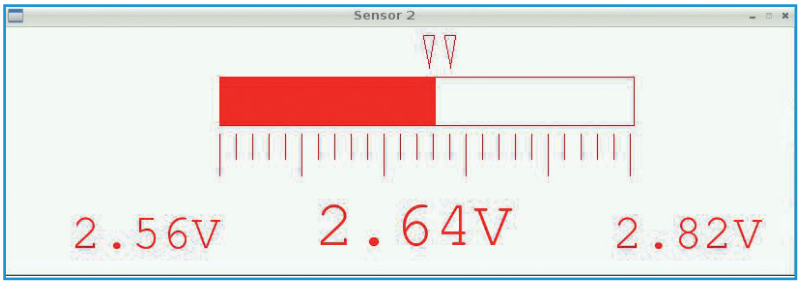

Bild 3. Die andersfarbige Variante mit zwei kleinen Pfeilen zur Anzeige der gespeicherten Werte.

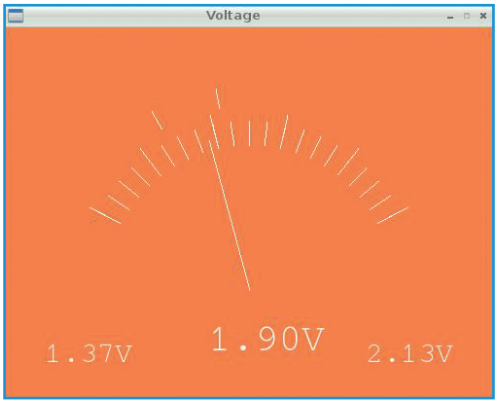

Bild 4. Voltmeter mit Zeigerdarstellung und Digitalanzeige.

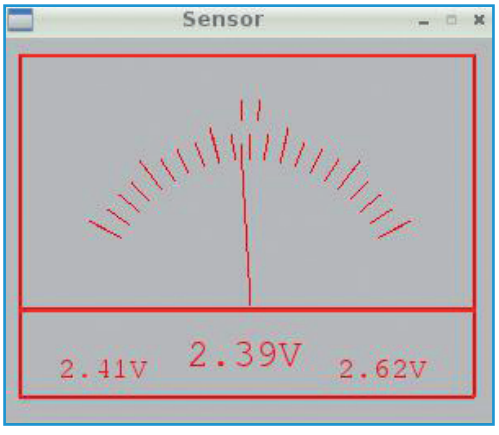

Bild 5. Die „Luxusausführung“ mit Rahmen um das Messwerk.

Dateien im Zip-Format kopieren und danach entpacken.
Zuvor ist noch die Pygame-Bibliothek zu installieren, die in allen Programmbeispielen des Autors verwendet wird. Wenn sie die Debian-Version des Betriebssystems benutzen, geben Sie dazu im Terminal folgende Zeile ein:

```
sudo apt-get install python-game
```

Nun kann man die Programme des Autors erproben.
Sie wechseln dazu im Dateimanager in das Verzeichnis mit den Pythondateien und starten das Programm mit:

```
sudo python ADW_PTR_D.py
```

(bzw. `sudo python ADW_PTR_E.py` bei der Programmversion in englischer Sprache).

Anschließend gibt man im Terminal eine der vorgesehenen Hintergrundfarben ein (mit ENTER bestätigen), dann eine der genannten Vordergrundfarben, dann eine Bezeichnung für die Darstellung sowie einen Faktor für die Größe der Darstellung auf dem Bildschirm. Unmittelbar darauf öffnet sich das Fenster mit der Anzeige.

Sollte die Darstellung wegen des ausgewählten Faktors zu groß oder zu klein sein, können Sie das Programm im Terminal mit Strg + C unterbrechen und es erneut starten, um an der entsprechenden Stelle einen anderen Faktor für die Größe der Darstellung zu wählen. In ähnlicher Weise, wie dies oben beschrieben ist, gehen Sie bei den anderen Programmversionen vor.

Im Beispiel in **Bild 6** wurde als Hintergrundfarbe „white“, als Vordergrundfarbe für den Zeiger, die Skala und die Messwertanzeige „red“ gewählt. Als Faktor für die Größe auf dem Bildschirm wurde in Bild 6 der Wert 0.7 eingegeben.

Bild 7 zeigt die notwendigen Eingaben im Terminal, damit eine Messspannung am Eingang des TLC549 durch den Raspberry Pi in blauer Farbe auf gelbem Hintergrund dargestellt wird.

Listing in Python

Der AD-Wandler TLC549 wird über die GPIO-Pins 7, 8 und 24 des Raspberry Pi angesteu-

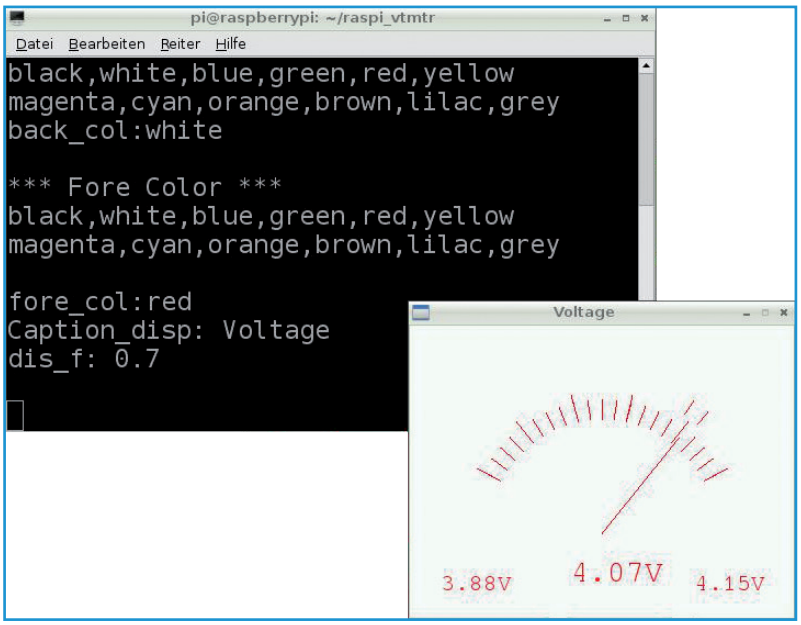

Bild 6. Simulation eines Messwerks mit Zeiger und die dafür notwendigen Eingaben im LXTerminal des Raspberry Pi.

ert. GPIO7 wurde in der Python-Software als AD_CS und GPIO8 als AD_Clk definiert, damit der Code verständlicher wird. Das Datenblatt des AD-Wandlers [3] zeigt, wie die Pins bedient werden müssen, damit der Baustein einen Messwert einliest. An seinem Ausgang DATA werden nacheinander alle acht Bits des Ergebnisses zur Weiterverarbeitung angeboten. Dieser ist mit GPIO24 verbunden, der in der Software als AD_Dat definiert wurde. In beiden Programmversionen geschieht die

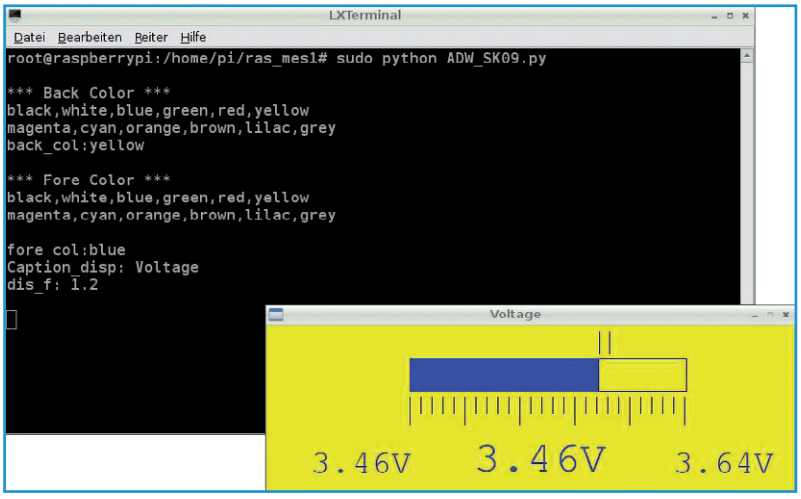

Bild 7. Die Eingaben im LXTerminal für die abgebildete Simulation.

Ansteuerung des TLC549 mit den nebenstehenden Programmzeilen.

Für die digitale Anzeige und die grafische Darstellung müssen jeweils folgende Umrechnungen vorgenommen werden:

In der Programmversion 1 muss für die Bargraphdarstellung eines aktuellen Messwerts und auch für die jeweilige digitale Anzeige die zuvor angezeigte Farbfläche beziehungsweise der zuvor angezeigte Wert stets vorher gelöscht werden. Dies geschieht, indem zuerst ein „gefülltes“ Rechteck unsichtbar gezeichnet, also mit der Hintergrundfarbe dargestellt wird. Erst dann wird die dem aktuellen Wert entsprechende neue farbige Fläche gezeichnet und der aktuelle Spannungswert digital angezeigt.

In ähnlicher Weise wurde das Listing für die Programmversion 2 gestaltet. Der Aufwand ist allerdings erheblich größer als für die erste Version. Die Werte für die Anfangs- und Endpunkte der Linien für die Skalenstriche sind mit den Winkelfunktionen Sinus und Cosinus zu bestimmen. Der Zeigerausschlag des simulierten Voltmeters entspricht jeweils dem momentanen Messwert, also sind im Programm auch dafür Berechnungen mit den Winkelfunktionen Sinus und Cosinus vorzunehmen.
Der Zeiger ragt in Anlehnung an ein reales Messinstrument in der jeweiligen Position in die Skala hinein. Ähnliches gilt für die Bestimmung der beiden Markierungen, deren Richtung der momentanen Richtung des simulierten Zeigers beim Drücken des zugeordneten Tasters entspricht.
Vor der Abbildung der momentanen Zeigerposition wird anders als in der ersten Programmversion jeweils der ganze Bewegungsbereich des Zeigers (Kreisausschnittsfläche) gelöscht. Weitere Einzelheiten lassen sich den Anmerkungen in den jeweiligen Programmlistings entnehmen.

(130383)

Ansteuerung des TLC549

```
def ADein():
    GPIO.output(AD_Clk,GPIO.LOW)# AD_Clk auf 0
    AD_res=0
    for n in range(10):
        time.sleep(0.05)
        GPIO.output(AD_CS,GPIO.HIGH)# AD_CS auf 1
        MSB=128
        time.sleep(0.001)
        GPIO.output(AD_CS,GPIO.LOW)# AD_CS auf 0
        time.sleep(0.0005)
        AD_Wert=0
        for z in range(8):
            if (GPIO.input(AD_Dat)):
                AD_Wert=AD_Wert+MSB
            GPIO.output(AD_Clk,GPIO.HIGH)# AD_Clk auf 1
            time.sleep(0.0005)
            GPIO.output(AD_Clk,GPIO.LOW)# AD_Clk auf 0
            MSB=MSB>>1
            time.sleep(0.0005)
            Ergebnis=AD_Wert
        GPIO.output(AD_CS,GPIO.HIGH)# AD_CS auf 1
        AD_res=AD_res+AD_Wert
    AD_res=AD_res/10
    Ergebnis=AD_res
    return Ergebnis
```

Umrechnungen

```
def Umrechnung(Wert):
    value=Wert
    value=value*1960 #fuer 5V
    value=value/1000
    pic_value=value/2 #fuer Bargraphdarstellung
    einer=value/100
    rest_z=value % 100
    zehntL=rest_z/10
    hundertL=rest_z%10
    return einer, zehntL, hundertL, pic_value,value
...
```

Weblinks

[1] „Von Basic nach Python“, Elektor 5/2013, 6/2013 sowie 7/8 2013, www.elektor-magazine.de

[2] www.ti.com/lit/ds/symlink/tlc549.pdf

[3] www.elektor-magazine.de/130383

[4] http://lxde.org