Raspberry-Pi-Rezepte

Teil 3

Von UART zu RS232

Von **Tony Dixon** (UK)

In der letzten Folge ging es um den Expansion-Header von RPi (Raspberry Pi) und dessen GPIOs bzw. seine Ein- und Ausgänge. Nun geht es weiter mit dem UART als Basis der seriellen Schnittstelle, dessen Signale ebenfalls am Expansion-Header bereitstehen.

Die serielle Schnittstelle ist wohl das grundlegendste aller Interfaces. Seit den frühen Tagen der Computerei steht diese Art der Verbindung von Peripherie zur Verfügung – sinnvollerweise also auch bei RPi. Eine Übersicht der RS232-Pins findet man im **Kasten**.

Serielle Schnittstellen

Beim UART (Universal Asynchronous Receiver/Transmitter) handelt es sich um einen von drei seriell operierenden Schnittstellen auf dem Expansion-Header. Die anderen beiden sind das I2C- und das SPI-Interface.

Die **Tabellen 1a** und **1b** listen die Belegungen des Expansion-Headers auf. Die UART-Signale TxD und RxD liegen an den Pins 8 und 10. Leider bietet der Expansion-Header außer diesen beiden nur noch ein weiteres UART-Signal, nämlich RTS an Pin 11 – Handshaking ist also nicht groß machbar.

Pegelwandlung

Zunächst braucht es einen Pegelwandler, der die UART-Signale mit 3,3-V-Pegel auf die „richtigen“ Pegel mit ±12V des RS232-Standards liftet. **Bild 1** zeigt die Einfachheit solch einer Wandlung. Ein MAX3232 oder ein äquivalentes IC genügt, um die korrekten RS232-Pegel zur Verfügung zu stellen. **Bild 2** zeigt den Hardware-Aufbau, bei dem ein kleines Add-on-Board eine „richtige“ RS232-Schnittstelle ermöglicht.

Serielle Konsole deaktivieren

Als Default-Einstellung der Raspbian-Distribution ist der Zugriff auf die Shell von RPi per Konsole über den UART möglich. Dies kann

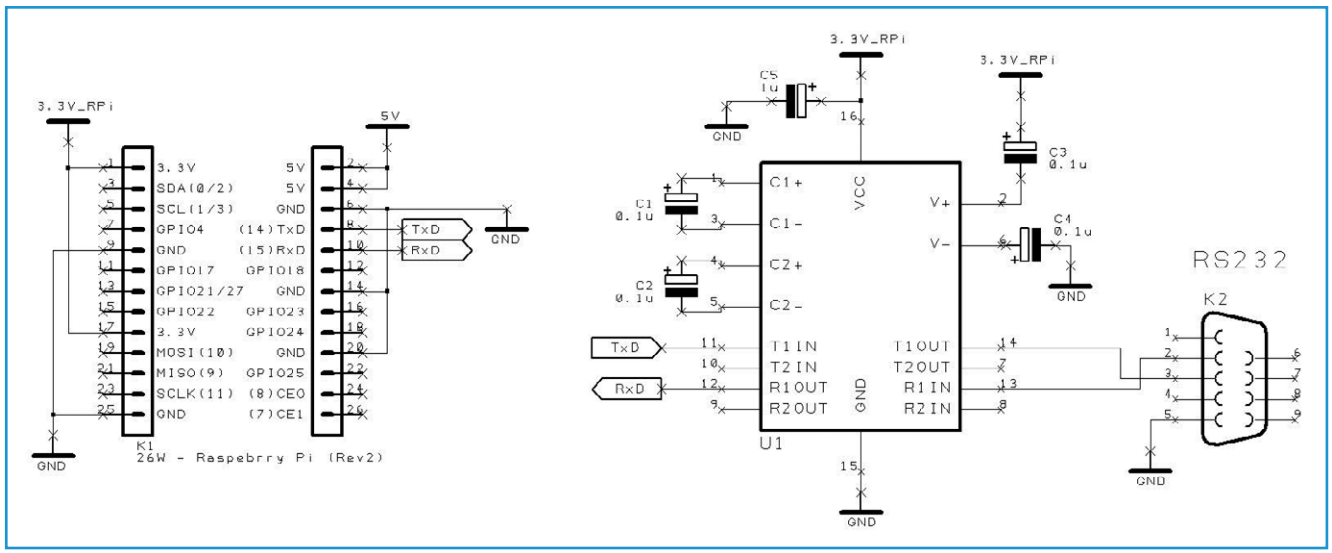

Bild 1.
Schaltung des RS232-Adapters für RPi.

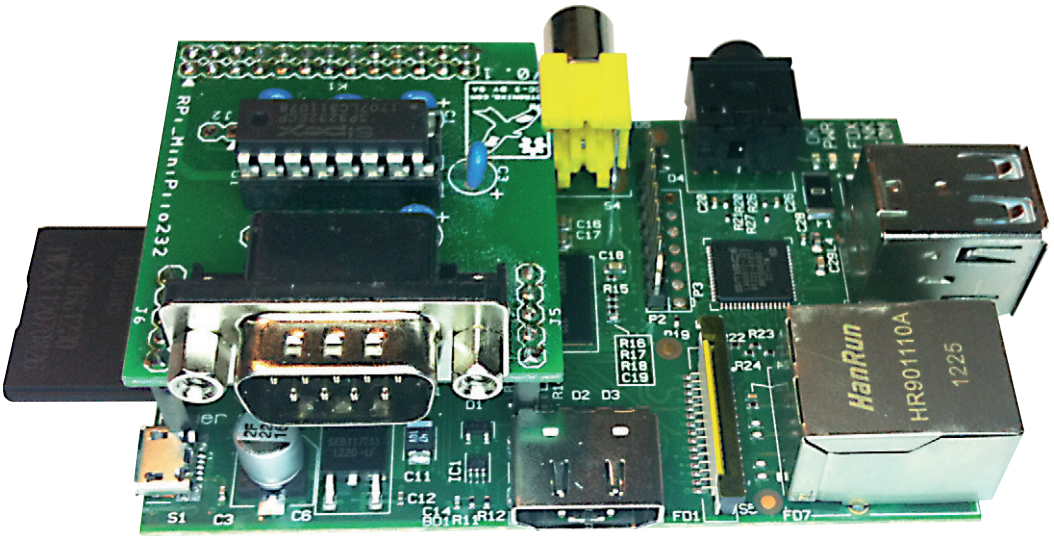

Bild 2.
RPi und das
RS232-Add-On-Board.

sehr nützlich sein, wenn man weder Bildschirm noch Tastatur an RPi angeschlossen hat. Bei der „gewöhnlichen“ Verwendung der seriellen Schnittstelle ist das aber ein Problem. Von daher muss man hierzu den seriellen Shell-Zugriff deaktivieren. Dies erfordert ein paar kleine Änderungen in den Dateien *cmdline.txt* und *inittab*. Zuvor empfiehlt sich eine Sicherungskopie der originalen Dateien:

```
sudo cp /boot/cmdline.txt /boot/cmdline.bak
sudo cp /etc/inittab /etc/inittab.bak
```

Um *cmdline.txt* via LXTerminal zu modifizieren gibt man ein:

```
sudo leafpad /boot/cmdline.txt
```

Nun muss man den String „console=ttyAMA0,115200“ suchen und durch die Konfigurationsparameter „kgdboc=ttyAMA0,115200“ ersetzen.

In *cmdline.txt* sollte dann Folgendes zu lesen sein:

```
dwc_otg.lpm_enable=0 console=ttyAMA0,115200 kgdboc=ttyAMA0,115200
console=tty1 root=/dev/mmcblk0p2 rootfstype=ext4 elevator=deadline rootwait
```

Nun sucht man „console=ttyAMA0,115200“ und ersetzt den String durch „kgdboc=ttyAMA0,115200“. Das Resultat ist dann:

```
dwc_otg.lpm_enable=0 console=tty1 root=/dev/mmcblk0p2 rootfstype=ext4 elevator=-deadline rootwait
```

Anschließend sichert man die Datei und beendet den Editor.
Nun muss noch die Datei „/etc/inittab“ editiert und dort der serielle Port „ttyAMA0“ deaktiviert werden. Hierzu tippt man:

```
sudo leafpad /etc/inittab
```

Am Ende der Datei sollte eine Zeile mit der Port-Adresse für „ttyAMA0“ stehen.
Hier setzt man ein Doppelkreuz („#“) oder ein Pfund-Zeichen an den Anfang der Zeile mit „ttyAMA0“.
Nun noch die Datei „/etc/inittab“ sichern und den Editor beenden.
Anschließend ist ein Neustart von RPi fällig.
Wenn man jetzt die serielle Schnittstelle für eigene Zwecke in eigenen Programmen verwenden will, nutzt man die Device-Adresse für „ttyAMA0“ zum Zugriff auf den seriellen Port.

Installation der Serial Library in Python

Wie schon erwähnt, ist Python schon standardmäßig in der Raspian-Distribution dabei, aber zur Verwendung der seriellen Schnittstelle ist die Installation einer geeigneten seriellen Hardware-Library notwendig. pySerial ist so eine Python-Library für serielle Interfaces, die leider nicht schon im Raspian-Paket enthalten ist. Also muss man sie noch downloa-

RS232-Pin-Belegung				
Pin	**Signal**	**Beschreibung**	**DTE**	**DCE**
1	DCD	Data Carrier Detected	IN	OUT
2	RD (or RxD)	Receive Data	IN	OUT
3	TD (or TxD)	Transmit Data	OUT	IN
4	DTR	Data Terminal Ready	OUT	IN
5	GND	Signal Ground	GND	GND
6	DSR	Data Set Ready	IN	OUT
7	RTS	Ready to Send	OUT	IN
8	CTS	Clear to Send	IN	OUT
9	RI	Ring Indicator	IN	OUT

Tabelle 1a.			
Pin-Name	**Pin-Funktion**	**Alternative**	**RPi.GPIO**
P1-02	5.0V	-	-
P1-04	5.0V	-	-
P1-06	GND	-	-
P1-08	GPIO14	UART0_TXD	RPi.GPIO8
P1-10	GPIO15	UART0_RXD	RPi.GPIO10
P1-12	GPIO18	PWM0	RPi.GPIO12
P1-14	GND	-	-
P1-16	GPIO23		RPi.GPIO16
P1-18	GPIO24		RPi.GPIO18
P1-20	GND	-	-
P1-22	GPIO25		RPi.GPIO22
P1-24	GPIO8	SPI0_CE0_N	RPi.GPIO24
P1-26	GPIO7	SPI0_CE1_N	RPi.GPIO26

Tabelle 1b.				
Pin-Name	**Board Revision 1**		**Board Revision 2**	
	Pin-Funktion	**Alternative**	**Pin-Funktion**	**Alternative**
P1-01	3.3V	-	3.3V	-
P1-03	GPIO0	I2C0_SDA	GPIO2	I2C1_SDA
P1-05	GPIO1	I2C0_SCL	GPIO3	I2C1_SCL
P1-07	GPIO4	GPCLK0	GPIO4	GPCLK0
P1-09	GND	-	GND	-
P1-11	GPIO17	RTS0	GPIO17	RTS0
P1-13	GPIO21		GPIO27	
P1-15	GPIO22		GPIO22	
P1-17	3.3V	-	3.3V	-
P1-19	GPIO10	SPI0_MOSI	GPIO10	SPI0_MOSI
P1-21	GPIO9	SPI0_MISO	GPIO9	SPI0_MISO
P1-23	GPIO11	SPI0_SCLK	GPIO11	SPI0_SCLK
P1-25	GND	-	GND	-
Hinweis: I2C0_SDA und I2C0_SCL (GPIO0 & GPIO1) sowie I2C1_SDA und I2C1_SCL (GPIO2 & GPIO3) sind über 1,8-kΩ-Pull-up-Widerstände mit 3,3 V verbunden.				

den und installieren. Hierzu startet man eine LXTerminal-Session und tippt (siehe **Bild 3**):

```
sudo apt-get install python-serial
```

Beispiel: serial.py

Nachdem pySerial installiert ist, kann man ein kleines Testprogramm schreiben und damit Zeichen an ein Terminal-Programm auf einem PC schicken. Hierzu doppelklickt man auf das IDLE-Icon auf dem Desktop des RPi, um die Python-Shell und die IDE zu starten (siehe **Bild 4**).

Man wählt den Eintrag „File“ aus dem Menü; dadurch wird auch der Editor gestartet. Im Editor gibt man das Mini-Programm nach **Listing 1** so ein, dass es so aussieht wie in **Bild 5**.

Anschließend sichert man dieses Programm, geht zu LXTerminal und macht mit den folgenden Befehlen das Programm ausführbar:

```
chmod +x serial.py
```

Jetzt kann man das Programm mit dem folgenden Befehl starten:

```
sudo ./serial.py
```

Wenn RPi seriell mit einem PC verbunden ist und dort ein Terminal-Programm läuft, sollte man „Hello Elektor“ angezeigt bekommen.

(130151)

Weblinks:

Raspberry Pi:
www.raspberrypi.org

Library pySerial:
http://pypi.python.org/pypi/pyserial

Add-on-Board MiniPiio RS232:
www.dtronixs.com

Listing 1.

```
#!/usr/bin/python

import serial

ser = serial.Serial (‘/dev/ttyAMA0’, 115200, timeout=1)
ser.write (“Hello Elektor”)
ser.close ()
```

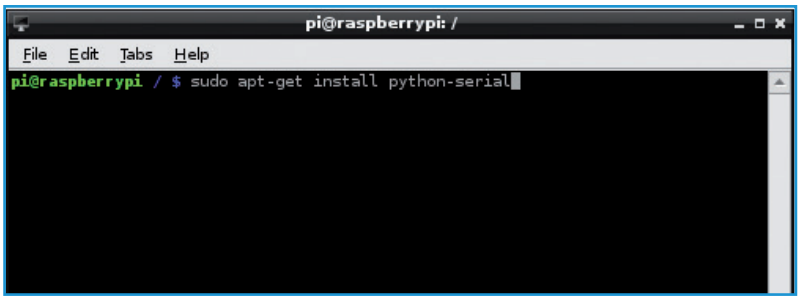

Bild 3. LXTerminal.

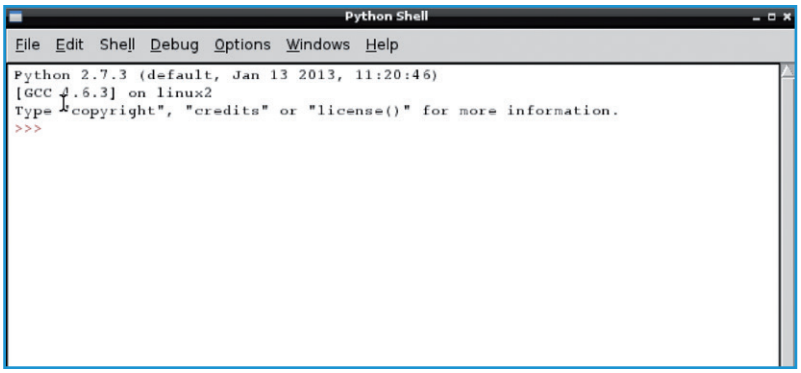

Bild 4. Die Python-Shell IDLE.

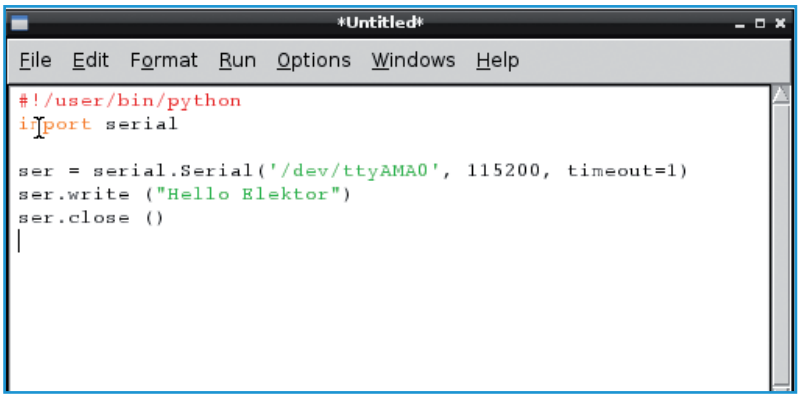

Bild 5. Der Editor der IDE.