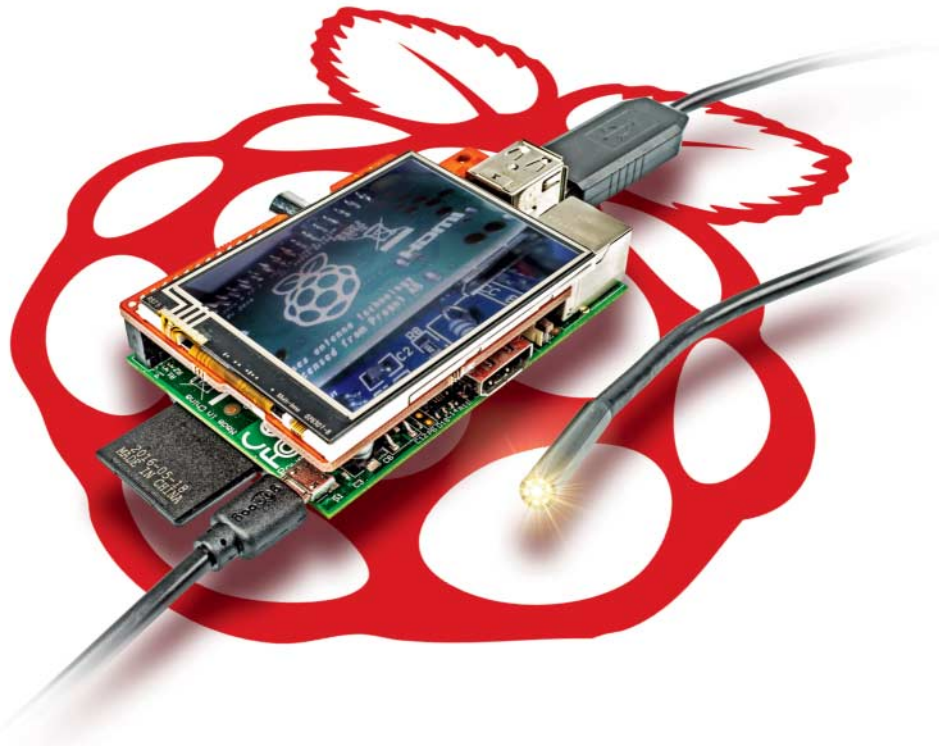




Schlüsselloch-spion

Raspberry Pi als autarkes Endoskop mit Kamera und Display

Was verstopft den Abfluss? Wo verlief noch einmal das Kabel hinter der Verkleidung? Und ist der Rahmen wirklich rostfrei? Ein selbstgebautes Endoskop aus einem alten Raspberry Pi, einem billigen Display und einer wasserdichten Miniatur-USB-Kamera beantwortet diese Fragen.

Von Mirko Dölle

Schon eine winzige Öffnung von nicht einmal 6 Millimetern Durchmesser genügt, um den Kamerakopf einzuführen – doch ein Bild kann man sich so noch nicht machen: Die winzigen wasserdichten USB-Endoskop-Ka-

meras von eBay für 10 bis 25 Euro haben kein Display. Notebooks und Tablets als Monitorersatz sind groß, teuer und unhandlich, um damit im Motorraum oder unter dem Spültisch zu hantieren. Die Lösung ist ein alter Raspberry Pi, der für 10 bis 20 Euro um ein Display und ein Powerpack ergänzt wird. Das Ergebnis ist ein kompaktes, preisgünstiges Endoskop.

Die Teileliste ist kurz, neben einem beliebigen Raspberry Pi – selbst ein alter Raspi Zero lässt sich problemlos verwenden – benötigen Sie lediglich noch ein Display, ein USB-Powerpack für die Stromversorgung und natürlich die USB-Endoskop-Kamera. Letztere kostet bei eBay zwischen 10 und 25 Euro, je nachdem, wie lang der Kameraschlauch sein soll. Es gibt sie auch in verschiedenen Größen – wir haben uns für das 5-Meter-Modell mit 5,5-Millimeter-Kopf inklusive LEDs zur Be-

leuchtung für 15 Euro entschieden. Die etwas größeren Modelle mit 7-Millimeter-Kopf werden oft mit Winkelspiegeln oder Haken als Zubehör geliefert.

Die Treiberunterstützung unter Raspbian ist einwandfrei, sämtliche von uns ausprobierten Kameras wurden automatisch erkannt und als `/dev/video0` eingebunden. Auch praktisch alle Mini-TFT-Displays mit SPI (Serial Peripheral Interface) funktionieren dank des generischen Framebuffer-Treibers von Noralf Trønnes am Raspi. Diese gibt es reichlich für wenig Geld bei eBay, die kleinsten schon für unter 10 Euro inklusive Porto. Displays, die sich als Shield auf den GPIO-Anschluss des Raspi stecken lassen, kosten 20 Euro oder mehr. Ähnliche Modelle, die nur für den Arduino beworben werden oder für die es gar keine Kompatibilitätsangaben gibt, funktionieren genauso gut, denn letztlich kommt es nur auf die Anschlusspins und die Controller-Chips der Displays an. Solange das Display eine SPI-Schnittstelle hat, stehen die Chancen gut, dass es am Raspi funktioniert.

Seriell angesteuert

SPI ist ein synchroner Vollduplex-Datenbus, auf dem der Bus-Master mit einem oder mehreren Slaves kommuniziert. Dafür besitzt der Raspberry Pi sechs Anschlusspins: Masse, Takt, Datenein- und -ausgang sowie zwei Select-Pins. Die Tabelle auf Seite 82 unten listet die Pins, ihre Benennung beim Raspi, mögliche alternative Bezeichnungen und die Funktion auf.

Die Kommunikation ist simpel: Der Master versendet im selbst erzeugten Takt der SCLK-Leitung seine Daten über MOSI an die Slaves. Gibt es eine Antwort, erhält er sie sogleich über die MISO-Leitung zurück. Mithilfe des Select-Pins erfahren die Slaves, dass sie gemeint sind und nicht ein anderer Busteilnehmer.

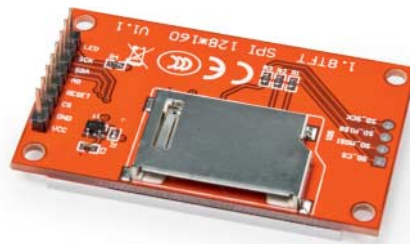
In der Praxis bereitet die Benennung der Display-Pins das meiste Kopfzerbrechen, denn sie ist wenig einheitlich und mitunter verwirrend. Auch sind längst nicht immer alle Pins vorhanden. Displays ohne Touchscreen etwa empfangen lediglich Daten, weshalb sie keine Antwort-Leitung benötigen und der MISO-Anschluss entfällt. Da es nur noch einen Datenpin gibt, nennen Hersteller ihn mitunter nicht MOSI, sondern SDA (Serial Data). Diese Pin-Bezeichnung wiederum ist typisch für den I²C-Bus und wird deshalb leicht verwechselt.

Neben der Datenleitung benötigt ein Display außerdem noch ein Pin für Chip Select (CS) und eins für die Umschaltung zwischen Daten und Kommandos (DC für „Data/Command“, manchmal auch AO oder A0). Häufig gibt es noch ein Reset-Pin und für die Hintergrundbeleuchtung einen weiteren Anschluss. Masse und Versorgungsspannung braucht das Display natürlich ebenfalls.

Wir haben uns bei eBay nach billigen Displays umgesehen und für gerade einmal 9 Euro inklusive Versand ein 1,8-Zoll-TFT mit 128 × 160 Pixeln und SPI-Interface gefunden. Die Beschriftung der Anschlusspins ist gewohnt kryptisch, durch Ausschuss der bekannten Pins konnten wir die Originalbeschriftung letztlich „übersetzen“ und so einen Verdrahtungsplan für den Raspberry Pi entwerfen. Das Ergebnis sehen Sie in der Tabelle auf Seite 82 rechts unten. Versuchsweise können Sie das Display auf einem Breadboard verdrahten, für die feste Installation sollten Sie aber besser ein Flachbandkabel an die Pins des Displays und auf der anderen Seite eine Postenbuchse für den GPIO-Anschluss des Raspi löten.

Farbenquizz

Ein weiterer Knackpunkt ist der Display-Controller, er muss vom Framebuffer-Treiber unterstützt werden oder wenigstens zu einem unterstützten Typen kompatibel sein – denn fast jeder Controller spricht ein anderes Protokoll. Welcher Controller in welchem Display steckt, lässt sich aber meist nur erraten. Mit etwas



Kennzeichnungswirrwarr: Statt SCLK und MISO gibt es bei diesem Display SCK und SDA, was eigentlich zum I²C-Bus gehört. Die Daten-Kommando-Umschaltung wurde von DC in AO umbenannt.

Glück hat das der eBay-Verkäufer wie in unserem Fall in den Artikeldetails korrekt angegeben – es sollte ein ST7735 sein.

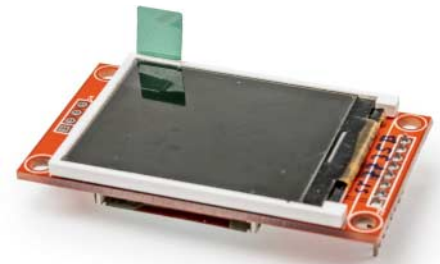
Ein Hinweis ist, kein Scherz, die Farbe der Abziehlasche der Display-Schutzfolie, in Fachkreisen als „tab“ bekannt. So steckt in Displays mit „green tab“ ein anderer Controller als in denen mit „black tab“ oder „red tab“; oder man findet Kompatibilitätsangaben, dass ein bestimmtes Display mit „green tab“ kompatibel zu einem anderen mit „black tab“ ist. Wir hüten uns deshalb tunlichst, bei unseren Displays in der Bastelkiste die Schutzfolie abzuziehen – und schreiben zusätzlich Farbe und Controller-Typ mit wasserfestem Stift irgendwo auf die Platine.

Ob der Controller unterstützt wird, recherchieren Sie am besten über die Kernelmodule. Zunächst einmal ermitteln Sie mit `modinfo fbftft_device` das Verzeichnis des Basistreibers und schauen hinein. Dort sollten Sie eine Datei mit dem Namen des Controllers finden, in unserem Fall `fb_st7735r.ko`. Der Controller wird also augenscheinlich unterstützt und das Kernelmodul heißt `fb_st7735r`. Finden Sie Ihren Controller nicht in der Liste der Kernelmodule, sollten Sie im Internet nachsehen, ob der Controller vielleicht ein Nachfolgemodell eines anderen ist oder zu einem anderen kompatibel, für den es einen Treiber gibt.

Mit dem Wissen um den Treiber und die Verdrahtung mit dem Raspi können Sie das Display bereits in Betrieb nehmen, indem Sie das Kernelmodul mit allen Informationen zur Verdrahtung als Parameter laden:

```
modprobe fbftft_device \
  custom name=fb_st7735r \
  speed=32000000 \
  gpios=reset:23,cs:8,dc:24,led:25 \
  rotate=90
```

Die beim Parameter `gpios` angegebenen Nummern beziehen sich auf die interne



Die am Display überstehende grüne Lasche hilft nicht nur, die Schutzfolie zu entfernen – sie ist außerdem der einzige Hinweis darauf, welcher Controller hier verbaut ist.

Nummerierung des GPIO-Busses, nicht auf die Pin-Nummern am GPIO-Anschluss. Zwei Parameter sind neu: Die Taktfrequenz des SPI-Busses (`speed`) und die Drehung des Displays (`rotate`) um 90 Grad, um vom Hochformat ins Querformat umzuschalten.

Flüssig im Takt

Die Taktfrequenz des SPI-Busses entscheidet darüber, wie schnell das Bild des Displays aktualisiert werden kann. Die Faustregel ist: Je höher die Auflösung, desto höher muss auch der SPI-Takt sein, um ein flüssiges Videobild mit hoher Wiederholrate zu bekommen. Allerdings steigt dadurch auch die Störanfälligkeit, insbesondere bei langen Kabeln. Es können Bildfehler auftreten oder der Raspi stürzt nach einer Weile ab. Werte von 8 bis 16 MHz sollten selbst bei Kabellängen von 30 Zentimetern und mehr störungsfrei funktionieren, bei kurzen Kabeln sind sogar 64 MHz möglich.

Die meisten Displays arbeiten mit 32 MHz (`speed=32000000`) zuverlässig und ausreichend schnell, mit diesem Wert sollten Sie beginnen. Die tatsächliche Taktfrequenz liegt übrigens etwas niedriger, da der Raspi mit einer Grundfrequenz von 250 MHz arbeitet und diese nur schrittweise halbieren kann – statt 64 MHz wählt der Treiber automatisch den nächst niedrigeren Wert 62,5 MHz (Divisor 4) und statt 32 MHz sind es tatsächlich nur 31,25 MHz (Divisor 8).

Standardmäßig verwendet Raspbian den internen Framebuffer des Grafikchips, `/dev/fb0`, für alle Ausgaben. Damit Raspbian das externe Display `/dev/fb1` nutzt, müssen Sie in der Datei `/boot/cmdline.txt` vor dem Parameter `rootnowait` folgende Parameter einfügen:

```
fbcon=map:10 fbcon=font:VGA8x8 ↵
FRAMEBUFFER=/dev/fb1
```

EndoskoPi



1 Stunde



25–50 Euro



Raspberry Pi oder Raspi Zero



Display mit SPI-Schnittstelle,
USB-Endoskop-Kamera,
Powerpack



keins



leicht

schwer



Nur oberflächlich angefressen oder durchgerostet? Mit dem Endoskop kann c't-Autor Thomas Koch den Rahmen seines VW Käfers von innen begutachten.

Zudem müssen Sie den SPI- und den Display-Treiber mit den notwendigen Parametern während des Bootvorgangs laden lassen. Das Laden erledigen die beiden folgenden Einträge, die Sie am Ende der Datei `/etc/modules` anfügen:

```
spi_bcm2835
fbtft_device
```

Die notwendigen Parameter für das Modul `fbtft_device` tragen Sie in der neuen Datei `/etc/modprobe.d/raspi.conf` genau so ein, wie Sie sie zuvor von Hand beim Laden des Moduls angegeben hatten:

```
options fbtft_device
└─ custom name=fb_st7735r
└─ speed=32000000
└─ gpios=reset:23,cs:8,dc:24,led:25
└─ rotate=90
```

Die Änderungen werden erst nach einem Neustart wirksam, der aber vorerst nicht nötig ist, wenn Sie das Modul bereits von Hand geladen haben.

Mini-X

Für die grafische Darstellung ist – wie auf einem großen Monitor – der X-Server zuständig. Aufgrund der geringen Auflösung der günstigen Displays, mehr als 320×240 Pixel bekommt man selten für kleines Geld, ist es nicht sinnvoll, einen Window Manager zu benutzen. Auch die Maus ist überflüssig, deshalb starten Sie den X-Server allein und ohne Maus:

```
FRAMEBUFFER=/dev/fb1 X -nocursor \
-retro &
```

Der Parameter `-retro` sorgt für ein Muster auf dem Bildschirmhintergrund, das erleichtert die Beurteilung der Bildqualität. Fehlt nur noch eine Anwendung, die das Bild der USB-Kamera auf das Display bringt: Wir haben uns für `luvcview` entschieden, das Sie leicht über die Paketverwaltung nachinstallieren können. Es hat den Vorteil, dass sich die Videoauflösung der Kamera per Kommandozeilenparameter einstellen lässt und das Bild links oben in der Anwendung angezeigt wird – die Bedienelemente befinden sich unterhalb des Bilds. Der Aufruf von `Luvcview` erfolgt so:

```
DISPLAY=:0 luvcview -s 160x120
```

Da das Display acht Pixelzeilen mehr hat als das Video, sehen Sie auf dem Miniatur-Display Teile der Bedienelemente am

unteren Bildrand, was aber nicht weiter stört. Bei einem Display mit 320×240 Pixeln würden Sie einfach die USB-Kamera mit der nächst höheren Videoauflösung, ebenfalls 320×240 Pixel, betreiben und hätten so ein Vollbild ohne Bedienelemente.

Damit der X-Server und `Luvcview` beim Booten des Raspi automatisch gestartet werden, laden Sie das Init-Skript `endoscope` von ct.de/yp4m herunter und kopieren es nach `/etc/init.d`. Es enthält die Aufrufe von X-Server und `Luvcview`. Letzteres in einer Schleife, sodass `Luvcview` immer wieder gestartet wird, so lange der X-Server läuft: Auf unserem Testsystem hatten wir einen Wackelkontakt am USB-Anschluss, sodass sich das Programm wegen fehlender Kamera häufig beendete.

Mit dem folgenden Befehl schalten Sie das Init-Skript scharf, sodass es beim nächsten Booten ausgeführt wird:

```
update-rc.d endoscope enable 2 3 4 5
```

Mit einer Powerbank ausgerüstet lässt sich der EndoskoPi überall im Haus und in der Werkstatt einsetzen – etwa um nach Ablagerungen in Rohren zu suchen, hinter Verkleidungen zu sehen oder Motoren durchs Zündkerzenloch zu begutachten. Bleibt nur noch ein Problem: Den Raspi geordnet wieder herunterzufahren. Dazu finden Sie unter [2] eine Anleitung zum Nachrüsten eines Tasters an Pin 5 und 6 des GPIO-Anschlusses sowie ein Python-Skript nebst Systemd-Job, mit dem Sie den Raspi auf Tastendruck neu starten, ausschalten und auch wieder einschalten können.

(mid@ct.de) **ct**

Literatur

- [1] Dr. Till Harbaum, *Raspi mit Touch, Touchscreens und fingerfreundliche Bedienung für den Raspberry Pi*, c't 19/16, S. 174
 [2] George Ilyes, *Raspberry Pi Shutdown/Reset/Start Button*: <http://gilyes.com/pi-shutdown-button/>

Init-Skript: ct.de/yp4m

SPI-Interface des Raspberry Pi			
GPIO-Pin	Typ	Bezeichnung	Funktion
23 (GPIO 11)	Out	SCLK (System Clock), SCK	Taktsignal
19 (GPIO 10)	Out	MOSI (Master Out, Slave In), SDO (Serial Data Out)	Datenausgang
21 (GPIO 9)	In	MISO (Master In, Slave Out), SDI (Serial Data In)	Dateneingang
24,26 (GPIO 8,7)	Out	CE (Chip Enable), CS (Chip Select), SS (Slave Select)	Slave-Auswahl
25	GND	GND (Ground)	Masse

Verdrahtung 1,8"-TFT-Display		
Display	Funktion	Raspi GPIO
1 (VCC)	+3,3V	17 (3.3V)
2 (GND)	Masse	20 (GND)
3 (CS)	SPI Chip Select	24 (SPI_CEO_N, GPIO 8)
4 (Reset)	Reset	16 (GPIO 23)
5 (AO)	Data/Command	18 (GPIO 24)
6 (SDA)	SPI MOSI	19 (SPI_MOSI, GPIO 10)
7 (SCK)	SPI Clock	23 (SPI_CLK, GPIO 11)
8 (LED)	Backlight	22 (GPIO 25)