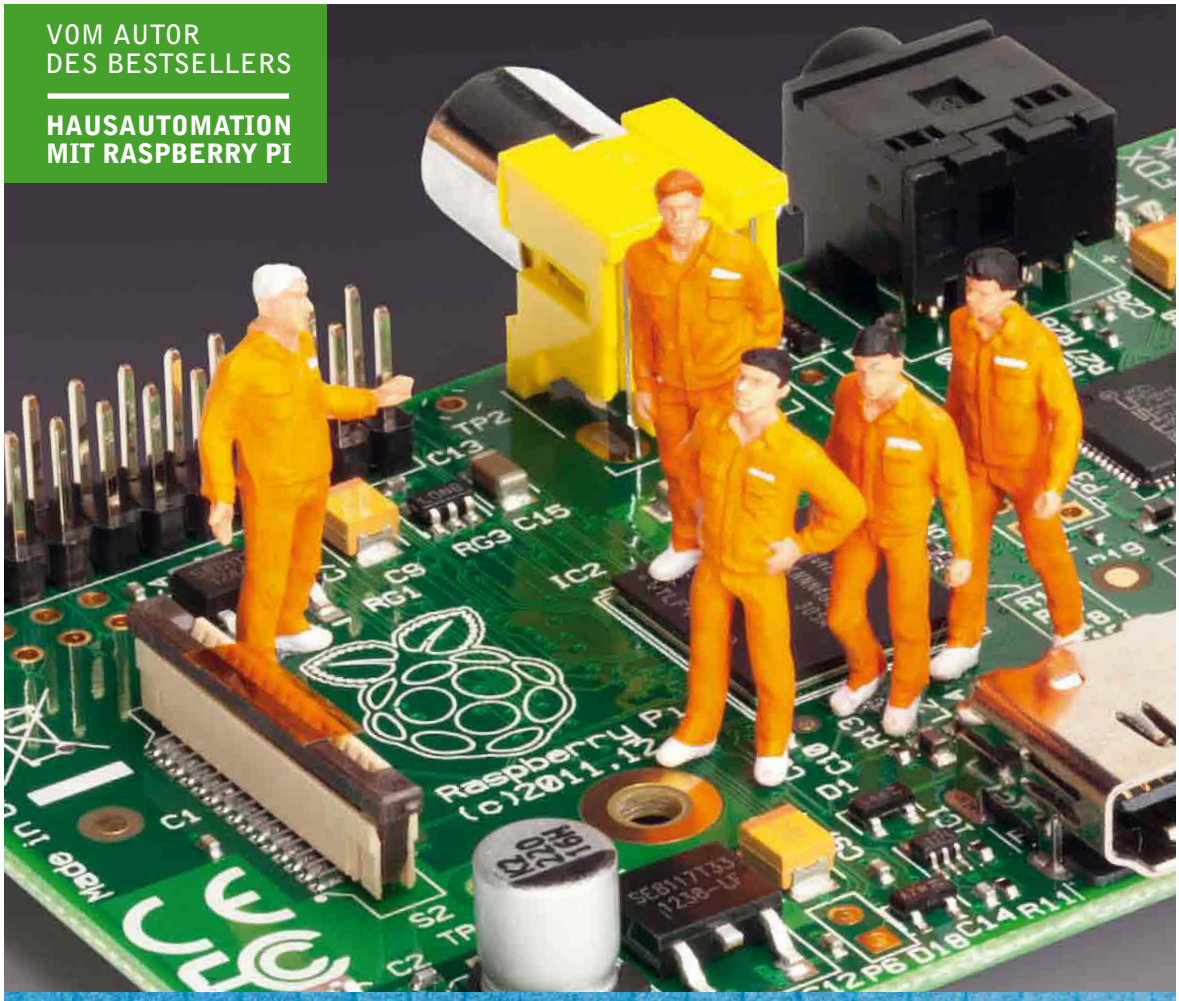
VOM AUTOR
DES BESTSELLERS

**HAUSAUTOMATION
MIT RASPBERRY PI**




FRUIT UP
YOUR
FANTASY

E. F. ENGELHARDT

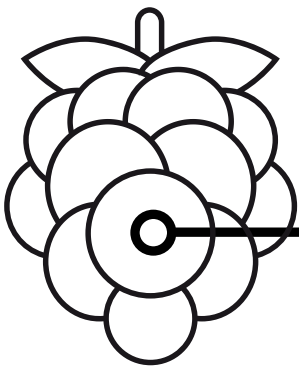
# ROBOTER MIT RASPBERRY PI

Mit Motoren, Sensoren, LEGO® und Elektronik eigene Roboter mit dem Pi bauen, die Spaß machen und Ihnen lästige Aufgaben abnehmen

FRANZIS

E. F. Engelhardt

# Roboter mit Raspberry Pi



FRUIT UP
YOUR
FANTASY

E. F. ENGELHARDT

# ROBOTER MIT RASPBERRY PI

Mit Motoren, Sensoren, LEGO® und Elektronik eigene Roboter mit dem Pi bauen, die Spaß machen und Ihnen lästige Aufgaben abnehmen

FRANZIS

Bibliografische Information der Deutschen Bibliothek

Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie;
detaillierte Daten sind im Internet über http://dnb.ddb.de abrufbar.



**Programmleitung:** Dr. Markus Stäuble
**Herausgeber:** Ulrich Dorn
**art & design:** www.ideehoch2.de
**Satz:** DTP-Satz A. Kugge, München

**ISBN 978-3-645-20343-2**

# Vorwort

Wer sich mit Mechanik und Robotik in Kombination mit dem Raspberry Pi beschäftigen möchte, der hat sich ein komplexes Themenfeld ausgesucht. Grundsätzlich ist neben einem Grundverständnis für die Elektronik und die GPIO-Anschlüsse auf dem Raspberry Pi der Wille notwendig, sich mit der Programmierung des Raspberry Pi auseinanderzusetzen.

Gerade wer sich noch nicht lange mit dem Thema Raspberry Pi beschäftigt, für den stellen die GPIO-Anschlüsse noch eine Wissenschaft dar, auch Änderungen im grundsätzlichen Setup des Raspberry Pi sind anfangs eine spannende Angelegenheit. Doch wenn Sie in Sachen Linux und Shell-Umgang schon fit sind, dann sollten Sie sich mit der Raspberry-Pi-Sprache Python beschäftigen: Hier bieten zig Codebeispiele umgehend passende Lösungen zu verschiedenen Problemstellungen an. Egal ob einfach und banal oder technisch kompliziert und verschachtelt mit mehreren Python-Bibliotheken: Mit diesem Buch werden Sie Schritt für Schritt zum Raspberry-Pi-Python-Experten, und Ihnen raubt anschließend keine nervige Fehlermeldung mehr den Schlaf.

Um Mechanik- und Robotik-Modelle zu bauen, bieten sich einige Möglichkeiten – die einfachste Lösung ist die Legokiste im Kinderzimmer. Angereichert mit LEGO®-Technic, lässt sich schon einiges bewegen und umsetzen – wer in Sachen Elektronik und Robotik hier Gas geben möchte, holt sich noch LEGO®-Mindstorms ins Haus. Mit und ohne Kinder: Hier sind einige Stunden Beschäftigung garantiert. Wer jenseits der traditionellen Legowelt mit Mindstorms eigene Lösungen bauen möchte, muss sich spätestens jetzt ernsthaft mit der Elektronik und der Programmierung auseinandersetzen. Hier lernen Sie die Gesetze der Physik und der Mechanik spielerisch kennen: Wer damit später spielen – oder besser, sich damit beschäftigen – will, sollte sich auch mit der Programmierung des Raspberry Pi auseinandersetzen. Hier bringt der Raspberry Pi eine Menge an Programmierschnittstellen mit.

Diese Schnittstellen werden auch benötigt, wenn es darum geht, den Raspberry Pi mit Robotern im Haushalt zu koppeln. In diesem Buch wird das an einem Vorwerk/Neato-Staubsaugerroboter demonstriert – hier wird das vorhandene Betriebssystem angezapft und mit dem Raspberry Pi gesteuert. Auch wenn Sie keinen Haushaltsroboter im Einsatz haben, die in diesem Projekt beschriebenen Kniffe wie das Anzapfen des USB-Anschlusses, die Steuerung eines Fahrzeugs über eine Webserversteuerung und vieles mehr lassen sich für weitere Raspberry-Pi-Projekte auch dank des vorliegenden Quellcodes schnell und bequem verwenden. Sie benötigen dafür ebenfalls Programmierkenntnisse – denn Mechanik und Robotik machen mehrere komplexe Denkvorgänge notwendig, damit beispielsweise klar ist, welche Motoren sich wie schnell bewegen müssen, um einen Arm zu bewegen und was mit

diesem erfasst werden kann. Außerdem müssen Sie sich überlegen, wie eine passende Steuerung in Form einer Software zu funktionieren hat, damit die Motoren im richtigen Moment das Gewünschte in der richtigen Reihenfolge tun.

Das Buch ist kein Einsteigerbuch zum Thema Raspberry Pi und Programmierung, doch nach dem Lesen werden Sie feststellen, dass die Mechanik und Robotik mit dem Raspberry Pi kein Hexenwerk ist. Sie brauchen aber etwas Zeit und Geduld sowie den Willen, auftretende Probleme selbst zu lösen. Wir wünschen Ihnen viel Spaß mit den Projekten!

Autor und Verlag

Sie haben Anregungen, Fragen, Lob oder Kritik zu diesem Buch? Sie erreichen den Autor per E-Mail unter *ef.engelhardt@gmx.de*.

# Inhaltsverzeichnis

# Lenken und Steuern mit der GPIO-Schnittstelle

Die zentrale Schnittstelle für das Messen, Steuern und Regeln von angeschlossenen Schaltern, Sensoren und Aktoren ist die GPIO-Schnittstelle (*General Purpose Input/Output*) des Raspberry Pi. Mit dieser Schnittstelle sind Sie für sämtliche Dinge in diesem Buch gerüstet – hier erweitern Sie den Raspberry Pi mit Schaltungen und Funktionen auf dem Steckboard, die später per Lötkolben in ein festes Platinendesign überführt werden können. Neben den Schaltungslösungen der Marke »Eigenbau« können Sie auch auf die hilfreiche Unterstützung in Form von zusätzlich zu erwerbenden Steck- und Erweiterungsboards für die Steuerung von Motoren aller Art zählen – beispielsweise gewöhnliche Gleichstrommotoren, Schritt- oder Steppermotoren, die gerade in der Robotik und bei Mechanikbasteleien mit dem Raspberry Pi eine wesentliche Rolle spielen.

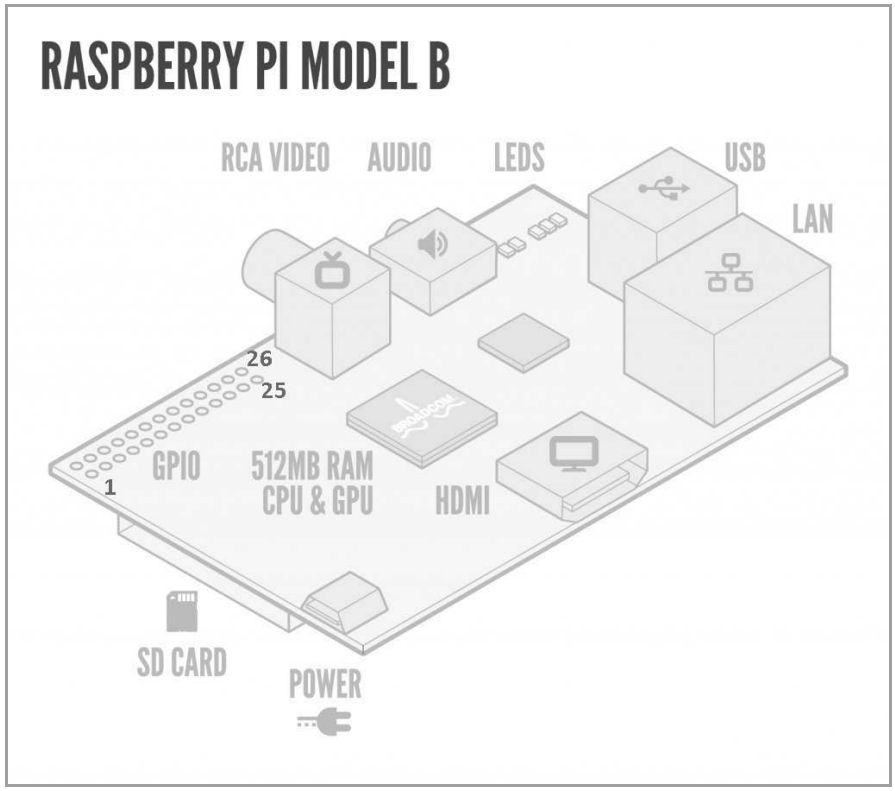



**Bild 1.1:** Der grundsätzliche Aufbau des Raspberry Pi und der GPIO-Pinleiste Modell B, Revision 2. (Grafik: *raspberrypi.org*)

Neben dem verbesserten Original werden zunehmend Klonplatinen im Internet angeboten, die auf die grundsätzliche Idee und Raspberry-Pi-Technik setzen. Erwähnenswert ist hier die Banana-Pi-Lösung, die nicht nur die gleichen Abmessungen wie das Original besitzt, sondern ebenfalls mit einer Pin-kompatiblen GPIO-Leiste ausgestattet ist, mit der bestehende Projekte eins zu eins auf eine leistungsfähigere Plattform gehievt werden können.

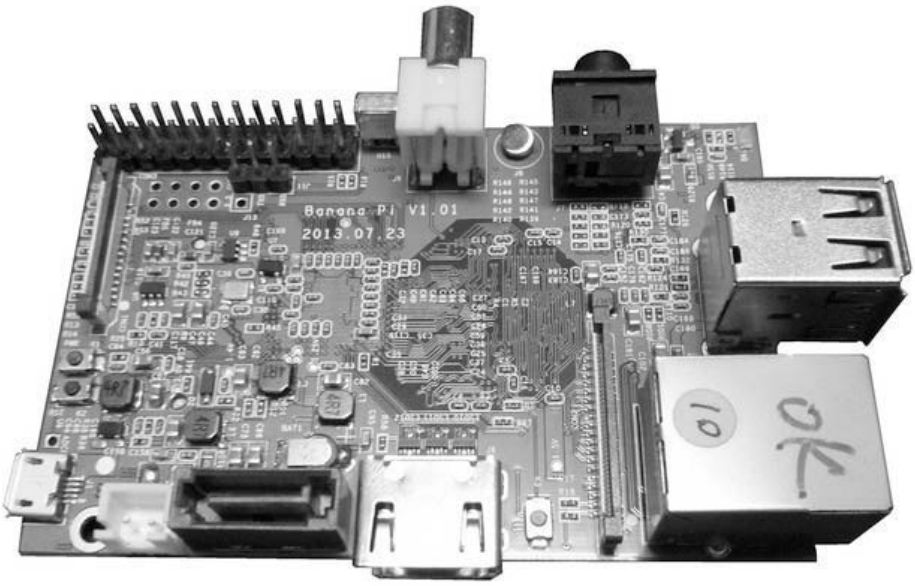



**Bild 1.2:** Stark verbesserter Raspberry-Pi-Klon, der nahezu die gleichen Abmessungen wie das Original besitzt und ebenfalls eine Pin-kompatible GPIO-Leiste bietet. (Abbildung: *Bananapi.org*)

Der im Vergleich zum Original mit 1 GB verdoppelte RAM-Speicher sowie der ARM Cortex A7-Dualkernprozessor mit 1 GHz Takt und die schnellere Grafikeinheit (Mali 400 GPU) beschleunigen die Platine enorm. Zwar stellt der bei Allnet (*www.allnet.de*) erhältliche Banana Pi mit einem Verkaufspreis von 69 Euro im Vergleich zum Original keine Low-Cost-Lösung mehr dar, er bringt jedoch zusätzlich ähnlich wie das Cubieboard einen SATA-Port mit, mit dem sich eine Festplatte mit bis zu 2 TB Kapazität anschließen lässt.

Wie der Raspberry Pi bietet der Banana Pi die CSI/DSI-Schnittstellen, zwei USB-2.0-Steckplätze sowie je einen HDMI- und einen AV-Videoausgang, außerdem sind noch ein Mikrofon und als zusätzliche Schnittstelle ein Onboard-IR-Empfänger verbaut. Der Klon bringt eine Gigabit-Netzwerkschnittstelle mit, während beim Original mit Ethernet-Anschluss ein 100/10-MBit-Modul verbaut ist.

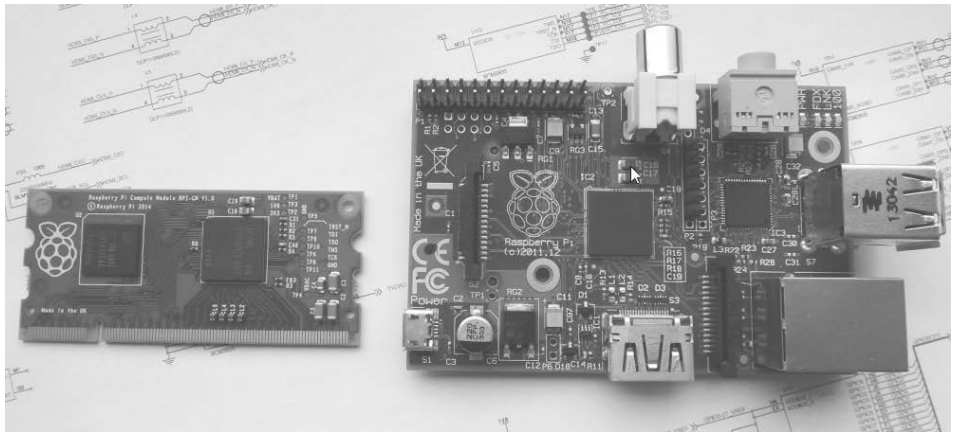


**Bild 1.3:** Die Raspberry Pi Foundation stellte im April 2014 das deutlich geschrumpfte Raspberry-Pi-Modul vor, das sich nun besser für kleinere, kompakte Anwendungen eignen soll. (Abbildung: Raspberry Pi Foundation)

Die Raspberry Pi Foundation setzt hier weiter auf Miniaturisierung: Das im April 2014 vorgestellte Compute Module ist mit dem BCM2835-Prozessor und 512 MB RAM wie ein »alter« Raspberry Pi ausgestattet. Statt des Speicherkartenslots sind hier 4 GB eMMc-Flash-Speicher direkt verlötet. Das Compute Modul sieht mit den Abmessungen von 67,7 × 30 mm nicht nur aus wie ein Speichermodul, sondern nutzt auch dieselbe Anschlussleiste wie ein DDR2 SODIMM. Dennoch ist es selbstverständlich, den Raspberry Pi nicht in einem RAM-Steckplatz des Computers einzustecken, sondern dafür das sogenannte »Breakout-Board« zu verwenden, das die bisher bekannten Anschlüsse zur Verfügung stellt.

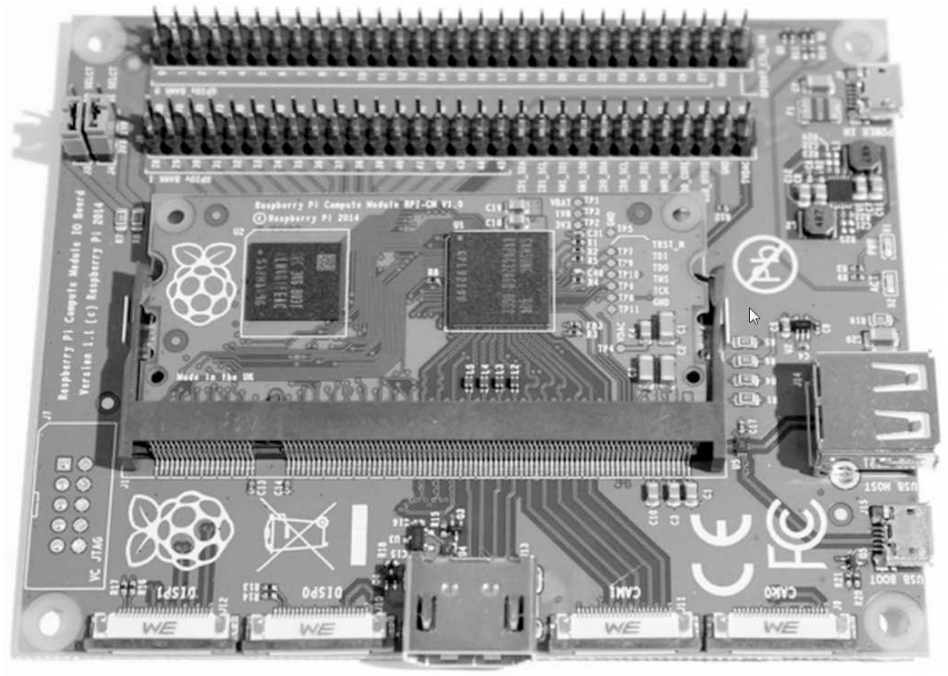


**Bild 1.4:** Das Raspberry-Pi-Modul wird in das flachere Breakout-Board eingesetzt, was bei mobileren Robotik-Anwendungen manchmal sinnvoller sein kann, wenn die Bauhöhe begrenzt ist. (Abbildung: Raspberry Pi Foundation)

Ab Juni 2014 soll der neue Raspberry Pi samt Breakout-Board im Handel erhältlich sein, im Mai 2014 wird bei einer Abnahme von 100 Stück für das Compute Modul ein Preis um die 30 Dollar fällig – für das Breakout-Board werden schätzungsweise weitere 10 bis 20 Dollar veranschlagt werden.

## 1.1 Betriebssystem und Treiber aktualisieren

Egal ob Sie selbst eine passende Schaltung im Eigenbau entwickeln oder die eine oder andere Fertiglösung zum Anschluss an den Raspberry Pi nutzen, das A und O ist der Zugriff per Software auf die Schnittstelle bzw. die Funktionen der einzelnen GPIO-Pins. Dafür stehen zahlreiche Möglichkeiten, Zubehör und Erweiterungsboards zur Verfügung, die in den nachfolgenden Kapiteln Schritt für Schritt erklärt und eingesetzt werden. Doch diese technischen Hilfsmittel sind oft wertlos, wenn das Grundsystem nicht stimmt.

Aus diesem Grund stellen Sie mit Kommandos wie `sudo apt-get update` und `sudo apt-get upgrade` sicher, dass der Raspberry Pi mit einem aktuellen Betriebssystem samt Treiber bestückt ist. Nach einem etwaigen Neustart per `sudo reboot` sollten Sie

hin und wieder auch das Kommando `sudo apt-get dist-upgrade` ausführen. Während sich das `apt-get upgrade`-Kommando primär um Anwendungen und Treiber kümmert, sorgt `apt-get dist-upgrade` für den aktuellen Kernel und installiert dessen Aktualisierungen.

Hat das Installationsprogramm hier Änderungen durchgeführt, sollten Sie den Raspberry Pi mit `sudo reboot` neu starten und den dazugehörigen Neustart live mitverfolgen. Nur dann haben Sie die Gewissheit, dass bei der Kernel-Aktualisierung alles gut gegangen ist und alle Dienste wieder starten.

## 1.2 Analog-digital-Wandler MCP3008 nachrüsten

Anders als bei anderen Mikrocontroller-Boards, beispielsweise aus der Arduino-Ecke, fehlen dem Raspberry Pi trotz der Menge an GPIO-Anschlüssen die analogen Eingänge. Ist der Einsatz von kostengünstigen analogen Sensoren gewünscht, ist entweder der direkte Umweg über einen Arduino oder ein Analog-digital-Wandler-Board notwendig, falls auf dem Raspberry Pi bzw. im zu steuernden Programmcode genauere Messwerte verarbeitet werden sollen. Im Raspberry-Pi-Umfeld ist der Analog-digital-Wandler MCP3008-IC (Datenblatt: *http://bit.ly/OaQwQh* bzw. *http://bit.ly/1fkK3gv*) sehr verbreitet, der für kleines Geld im Elektronikhandel erhältlich ist.

### Datenblatt prüfen, Funktionen verstehen

Um die Funktionsweise und das Zusammenspiel der Anschlüsse zu durchschauen, damit Sie den Analog-digital-Wandler MCP3008-IC mit dem Raspberry Pi einsetzen können, benötigen Sie Informationen aus dem Datenblatt dazu, wie die Kommunikation von Mikrocontroller und MCP3008-Chip vonstatten geht.

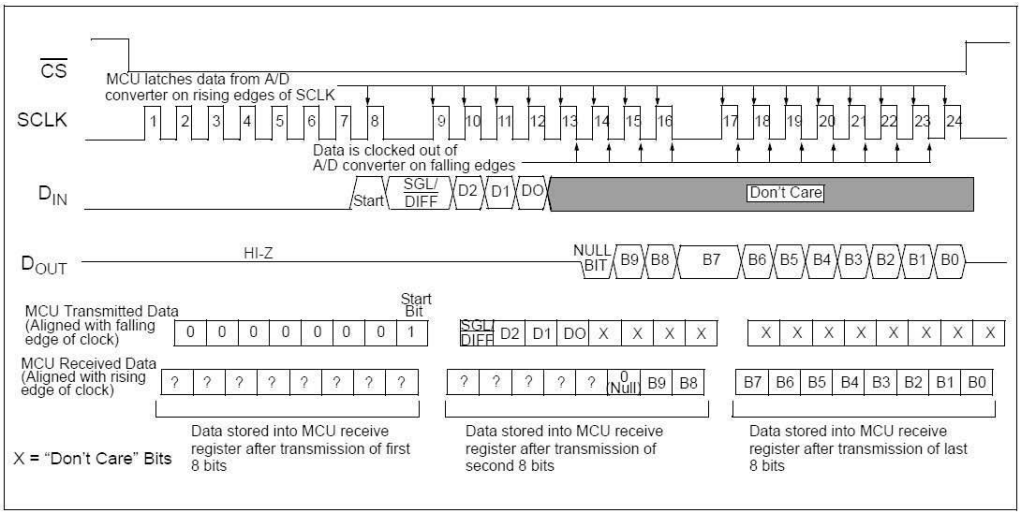



**Bild 1.5:** In der obersten Zeile des Datenblatts in der Abbildung sehen Sie das *Chip-Select-Signal*, das bei der dargestellten fallenden Flanke von HIGH auf LOW geht.

Um nun eine Datenübertragung anzutriggern, muss zunächst der Clock-Pin (`SCLK`) auf LOW-Pegel (GPIO-Pin auf `False`) gesetzt werden. Im nächsten Schritt benötigt der IC einen Befehl, damit dieser den Messwert des gewünschten Kanals zurückliefert. Der MCP3008 hat acht analoge Kanäle (`CH0` bis `CH7`), im Datenblatt wird das Kommando zum Auslesen des Messwerts in der obigen Abbildung bei *MCU Transmitted Data (Aligned with falling edge of clock)* mit dem Startbit 1 grafisch dargestellt. Hier bezeichnen die mit `D0`, `D1` und `D2` bezeichneten Bits den Binärwert des verwendeten Kanals. Somit sind neben den drei Bits für den Kanal noch das Startbit `1` und das `SGL/DIFF`-Bit notwendig, um den jeweiligen Kanal anzuschubsen, damit dieser anschließend den Messwert zum Mikrocontroller zurückschickt. Wie Sie dies in Programmcode umsetzen, lesen Sie, nachdem Sie die dazu passende Schaltung auf dem Steckboard umgesetzt haben.

## MCP3008 auf dem Steckboard nutzen

Um die Grundfunktionen des MCP3008-IC kennenzulernen, bauen Sie im nächsten Schritt eine Beispielschaltung mit einem Drehpotenziometer auf dem Steckboard auf und verbinden den MCP3008-IC mit der SPI-Schnittstelle sowie der Masse und Spannungsversorgung des Raspberry Pi. Der SPI (*Serial Peripheral Interface Bus*) lässt sich für die Signalübertragung zwischen dem Single Master (in diesem Fall dem Raspberry Pi) und einem oder mehreren Slave-Geräten bestens verwenden.

Die SPI-Schnittstelle selbst benötigt auf der GPIO-Leiste des Raspberry Pi die fünf Pins `SCLK` (*Serial Clock*, Taktgeber), `MOSI` (*Master Out, Slave In*), `MISO` (*Master In, Slave Out*) und `CE0`/`CE1` (*Slave Select*). Das `MISO`/`MOSI`-Pinpaar ist für die Full-Duplex-

Kommunikation zwischen dem Master und den Slaves zuständig, während über den CLK-Pin die angeschlossenen Geräte mit dem Master synchronisiert bleiben.

| Bauteil | Beschreibung | Händler/Bestell-nummer | Preis |
|---|---|---|---|
| Drehpotenziometer 10 kΩ | BOURNS – 3310Y-1-103L – POTENTIOMETER, 10K | Farnell/Bestellnummer 9353976 | ca. 2,88 Euro |
| MCP3008 | MICROCHIP – MCP3008-I/P – 10 BIT ADC, 2.7V, 8CH,SPI, 16DIP | Farnell/Bestellnummer 1627174 | ca. 2,59 Euro |

Durch die Beinchen lässt sich der Standard-IC für Experimentierzwecke bequem auf einem Steckboard und somit später auch auf der selbst bestückten Platine einsetzen.

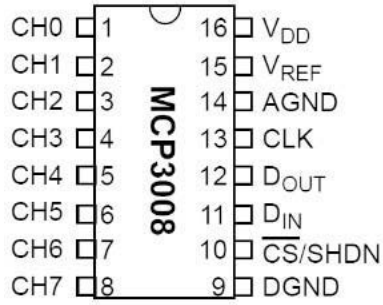



**Bild 1.6:** Zeigt die Kerbe des MCP3008-IC nach oben, erfolgt die Zählung der Anschlüsse wie beim Schreiben des Buchstabens U von links oben nach unten zu Anschlusspin 8, um dann wieder von rechts unten (Pin 9) nach oben zu Pin 16 zu gelangen.

Sämtliche acht analogen Eingangskanäle sind auf Pin 1 bis 8 auf der linken Seite untergebracht, während rechts die Stromversorgungs- und Schnittstellenanschlüsse an Pin 9 bis 16 Platz finden. Die Spannung an den acht analogen Eingangskanälen (CH0 bis CH7) darf laut Datenblatt –0,6 V bis $V_{DD}$+0,6 V nicht unter-/überschreiten, da der Schaltkreis sonst beschädigt werden könnte. Nach der Messung der Anschlüsse über den Vergleich mit der am V-REF-($V_{REF}$-)Pin anliegenden Spannung erfolgt die Berechnung der Differenz in einer Auflösung von 1,024 (10 Bit, $2^{10}$), die anschließend an die Datenschnittstelle übergeben wird.

| Bemerkung | Bezeichnung | Pin | O | Pin | Bezeichnung | Bemerkung |
|---|---|---|---|---|---|---|
| Kanal 1 | CH0 | 1 | | 16 | $V_{DD}$ | Versorgungsspannung liegt zwischen 2,7 V und 5,5 V mit $V_{DD}$ (max.) von 7 V. |
| Kanal 2 | CH1 | 2 | | 15 | $V_{REF}$ | Referenzspannung, mit der die jeweils an den analogen Anschlüssen anliegende Spannung verglichen und die Abweichung berechnet werden kann. |
| Kanal 3 | CH2 | 3 | | 14 | $A_{GND}$ | Masse-Signal analoger Schaltkreise. |
| Kanal 4 | CH3 | 4 | | 13 | CLK | Taktsignal. |
| Kanal 5 | CH4 | 5 | | 12 | $D_{OUT}$ | Digitaler Ausgang, SPI-Schnittstelle. |
| Kanal 6 | CH5 | 6 | | 11 | $D_{IN}$ | Digitaler Eingang, SPI-Schnittstelle. |
| Kanal 7 | CH6 | 7 | | 10 | CS | Falls Steuereingang (Cable Select, Chip Select, Chip Enable) low, ist der Schaltkreis aktiv. |
| Kanal 8 | CH7 | 8 | | 9 | $D_{GND}$ | Masse-Signal, digitaler Schaltkreis. |

Durch das Drehpotenziometer haben Sie die Möglichkeit, am Analogeingang die Spannung manuell zu regulieren. Durch die Drehung des Stellrads des Drehpotenziometers ändern sich die Widerstandsparameter und somit auch die Werte des Spannungsteilers, die sich auf die anliegende Spannung auswirken. Diese wird anschließend mit der Referenzspannung verglichen und kann somit digital als Vergleichswert dargestellt werden.

Steckt das MCP3008-IC-Modul im Steckboard, nutzen Sie für die Verbindung zum Raspberry Pi am besten die gewohnten Jumperkabel. Die einzelnen Pins auf der rechten Seite des MCP3008-IC sind anhand der obigen Tabelle bzw. dem Datenblatt (*http://bit.ly/OaQwQh*) schnell zugeordnet und an der GPIO-Reihe bestückt.

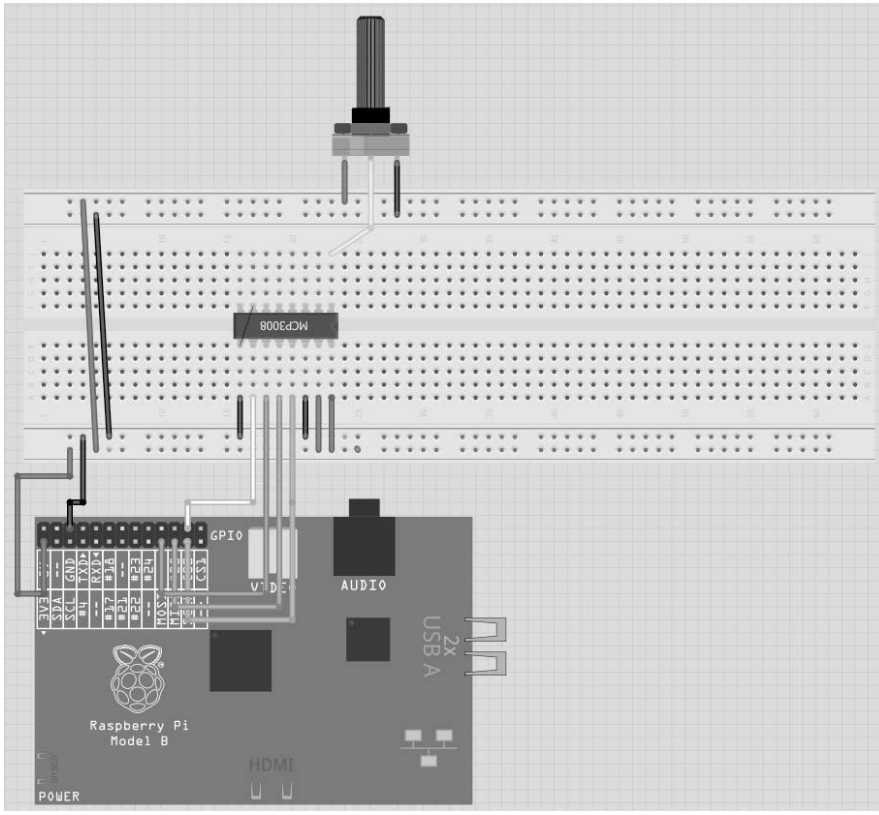



**Bild 1.7:** Das notwendige Potenziometer (oben) wird außen an Masse (–) angeschlossen und mit dem 3,3-V-Anschluss des Raspberry Pi (Rev. 2) verbunden. Der Mittelanschluss des Potenziometers wird an Eingang `CH0` – also Pin 1 des MCP3008-IC – angeschlossen.

So werden in diesem Beispiel Pin 15 und 16 mit 3,3 V, Pin 14 und Pin 9 mit Masse, Pin 13 mit GPIO11 (P1/23), Pin 12 mit GPIO09 (P1/21), Pin 11 mit GPIO10 (P1/19) und Pin 10 des IC mit GPIO08 (P1/24) mit den Anschlüssen des Raspberry Pi verbunden.

Für den ersten Test wird mit dem 10K-Potenziometer ein gewöhnliches, einfaches Potenziometer für den Steckboardeinsatz verwendet. Das Potenziometer wird an den beiden äußeren Anschlüssen je mit Masse (–) und dem 3,3 V-Anschluss verbunden. Der Mittelanschluss des Potenziometers wird an Eingang `CH0` – also Pin 1 des MCP3008-IC – angeschlossen.

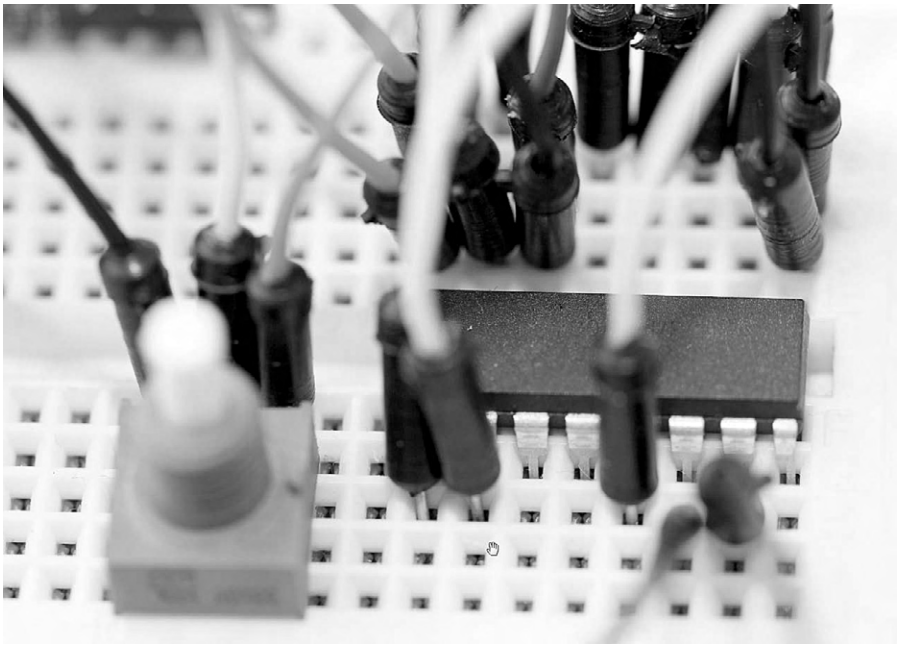


**Bild 1.8:** Raspberry Pi im SPI-Einsatz: Auf das Steckboard ist zusätzlich noch ein Drehpotenziometer gesteckt, dessen Mittelanschluss mit dem MCP3008-Pin 1 (Kanal `CH0`) verbunden ist.

Ist die Schaltung verkabelt, kann der Raspberry Pi wieder an die Stromversorgung angeschlossen und eingeschaltet werden. Im nächsten Schritt wird die manuelle Ansteuerung des MCP3008-IC über Python realisiert.

## Programmierung des MCP3008 mit Python

Aus dem vorigen Abschnitt wird ersichtlich, dass das MCP3008-IC eine 10-Bit-Auflösung, also $2^{10}$ mögliche Bitkombinationen bietet. Im Umkehrschluss heißt das, dass Sie bei der vorliegenden Eingangspannung bzw. Referenzspannung von 3,3 V für jede einzelne Bitänderung in dem Bereich von 0 bis 1023 einen anderen Messwert im Bereich von 0 bis 3,3 V haben.

Für jedes einzelne Bit ist in diesem Fall die Änderung leicht zu berechnen, indem Sie die Referenzspannung durch die Auflösung teilen (3,3 V / 1024), was 0,00322265625 V bzw. 3,22265625 mV entspricht. Die Stromstärke bleibt gleich, bei einer Drehung des Potenziometers ändert sich der Widerstand und somit auch die anliegende Spannung. Nach dem Vergleich mit der Referenzspannung wird die Differenz an die Datenschnittstelle übergeben.

| MCP3008-Pin | MCP3008 | Poti-Pin 10 kΩ | Raspberry-Pi-Bezeichnung | Raspberry-Pi-Pin | Wiring Pi |
|---|---|---|---|---|---|
| 1 | CH0 | 3 | – | – | – |
| 16 | $V_{DD}$ | 2 | 3.3V | 1 | – |
| 15 | $V_{REF}$ | – | 3.3V | 1 | – |
| 16 | $V_{DD}$ | – | 3.3V | 1 | – |
| 11 | $D_{IN}$ | – | GPIO10/SPI MOSI | 19 | 12 |
| 12 | $D_{OUT}$ | – | GPIO9/SPI MISO | 21 | 13 |
| 13 | CLK | – | GPIO11/SPI0_SCLK | 23 | 14 |
| 10 | CS/SHDN | – | GPIO8/SPI0_CE0_N | 24 | 10 |
| 9 | $D_{GND}$ | – | Masse | 6 | – |
| 14 | $A_{GND}$ | 1 | Masse | 6 | – |

Ist der Raspberry Pi mit dem Steckboard und auch mit dem MCP3008-IC verbunden, lassen sich nur dann Messwerte auslesen und verarbeiten, wenn an einem oder mehreren der acht verfügbaren Analogeingänge auch ein Signal anliegt. Die einfachste Methode ist in diesem Fall der Anschluss eines Drehpotenziometers, um per Drehung die Werte des Spannungsteilers manuell anzupassen.

```
cd ~
mkdir mcp3008
cd mcp3008
nano mcp3008-step1.py
```

Ist die Python-GPIO-Library installiert, prüfen Sie mit einem Python-Skript die Schaltung auf dem Steckboard. Die SPI-Library kommt in einem späteren Beispiel zum Einsatz. Die Beispieldatei `mcp3008-step1.py` finden Sie im Quellcodeverzeichnis `MCP3008`. Zunächst legen Sie die genutzten GPIO-Ein- und Ausgänge fest. In diesem Beispiel wird die BCM-Zählung auf dem Board Raspberry Pi Revision 2 genutzt – achten Sie bei der Verwendung einer älteren Revision auf die korrekte Zuordnung der GPIO-Nummern.

Für das Auslesen der Werte an den Eingängen des MCP3008-IC wird die Funktion `getAnalogData` genutzt, die mit den entsprechenden GPIO-Nummern der verwendeten Pins aufgerufen wird. Nach der Festlegung des Kanals kann die im Datenblatt beschriebene Syntax für das Auslesen der am jeweiligen Kanal anliegenden Spannung angewendet werden.

Mit der Zeile `GPIO.output(CS_Pin, False)` erzeugt das Skript eine steil abfallende Signalflanke (von 3,3 V auf 0 V) und aktiviert per Cable Select (CS) den A/D-Wandler des IC.

| Binärwert | Kanal |
|---|---|
| 000 | 0 |
| 001 | 1 |
| 010 | 2 |
| 011 | 3 |
| 100 | 4 |
| 101 | 5 |
| 110 | 6 |
| 111 | 7 |

Die mit dem `pushcmd`-Kommando festgelegte Bitfolge wird über den `MOSI`-Kanal ausgegeben, und mit der Zeile `pushcmd = poti_channel` wird die Nummer des Analogeingangs, welche durch drei Bits dargestellt wird, festgelegt. Durch das bitweise ODER in der Zeile `pushcmd |= 0b00011000` werden die drei Adressbits von rechts eingefügt, und anschließend wird die Bitfolge über die `for`-Schleife gesendet.

| Binärfolge | Bemerkung |
|---|---|
| 0b00011000 | 000 gibt die verwendete Kanaladresse (CH0) in binärer Schreibweise an. |
| 0b00011000 | SGL-(Signal-)Bit. |
| 0b00011100 | Startbit. |

Das Holen bzw. das Auslesen der Daten ist ebenfalls über eine `for`-Schleife realisiert. Hier nehmen Sie einen Pegelwechsel mit den beiden Befehlen `GPIO.output(CLK_Pin, True)` und `GPIO.output(CLK_Pin, False)` vor und lesen die Datenschnittstelle anschließend Bit für Bit aus. Liegt am `MISO`-Anschluss ein Pegel an, ist das letzte Bit eine `1`, sonst bleibt der Wert `0`. Auch diese Zuordnung wird mit der bitweisen ODER-Operation durchgeführt. Zu guter Letzt wird der in der Variablen `poti_channel_value` gesicherte Wert von der Funktion zurückgegeben.

```
# -*- coding: utf-8 -*-
#!/bin/python
import RPi.GPIO as GPIO
import time, os
#--------------------------------------------------------------------------
# Roboter mit Raspberry Pi
# E.F.Engelhardt, Franzis Verlag, 2014
# --------------------------------------------------------------------------
# Das Skript nutzt die Analogeingaenge des MCP3008-IC
# und liest diese ueber SPI-Bus aus.
# Datei mcp3008-step1.py
```

```
#  ------------------------------------------------------------------------
GPIO.setmode(GPIO.BCM) #  GPIO Mode
#  ------------------------------------------------------------------------
# Konfiguration Eingangskanal und GPIOs
poti_channel = 0  # Analog-Digital-Kanal 0
CLK_Pin = 11 # Clock -> Pin 13 / GPIO11 / SPI0_SCLK/ Pin 23 / wiringPi = 14
DIN_Pin = 10 # Digital in -> Pin 11 / GPIO10 / SPI MOSI / Pi Pin 19 /
wiringPi = 12
DOUT_Pin = 9 # Digital out -> Pin 12 / GPIO9 / SPI MISO / Pin 21 / wiringPi
= 13
CS_Pin = 8 # Cable Select -> Pin 10 / GPIO8 / SPI0_CE0_N / Pin 24 /
wiringPi = 10
#  ------------------------------------------------------------------------
# GPIO konfigurieren
GPIO.setup(CLK_Pin, GPIO.OUT)
GPIO.setup(DIN_Pin, GPIO.OUT)
GPIO.setup(DOUT_Pin, GPIO.IN)
GPIO.setup(CS_Pin, GPIO.OUT)
# ------------------------------------------------------------------------
# Funktionen
#
def getAnalogData(poti_channel, CLK_Pin, DIN_Pin, DOUT_Pin, CS_Pin):
    # Pegel definieren
    GPIO.output(CS_Pin, True)
    GPIO.output(CS_Pin, False)
    GPIO.output(CLK_Pin, False) #
    pushcmd = poti_channel
    pushcmd |= 0b00011000 # = Hex 0x18 (1:Startbit, 1:Single/ended)
                      #  Kommando zum Werte-Abruf des Kanals
    # Bitfolge senden
    for i in range(5):
        GPIO.output(DIN_Pin, bool(pushcmd & 0b10000))
        # 4. Bit prüfen und mit 0 anfangen
        # Clocksignal negative Flanke erzeugen
        GPIO.output(CLK_Pin, True)
        GPIO.output(CLK_Pin, False) # Taktsignal MCP anschubsen, um Bit auf
MOSI zu nehmen
        pushcmd <<= 1 # Bitfolge eine Stelle nach links verschieben.
        # Damit kommt das nächste Bit an Position 5 und kann im nächsten
Schleifenlauf
        # ausgeworfen werden.
    # Daten holen
    poti_channel_value = 0 # Wert auf 0 resetten
    for i in range(11):
        GPIO.output(CLK_Pin, True)
        GPIO.output(CLK_Pin, False)
        poti_channel_value <<= 1 # Um eins nach links schieben
```

```
        if(GPIO.input(DOUT_Pin)): # Liegt an MISO-Pegel an, dann wird
            poti_channel_value |= 0x01 # letztes Bit eine 1, sonst bleibt
es 0
    time.sleep(0.5)
    return poti_channel_value
# ------------------------------------------
# Start Python-Skript
# ------------------------------------------
os.system('clear')
print "[mcp3008] Messung Drehpotenziometer"
try:
  while True:
    tmp_value = getAnalogData(poti_channel, CLK_Pin, DIN_Pin, DOUT_Pin,
CS_Pin)
    print "Wert Kanal Nr.", poti_channel, "=", tmp_value
    time.sleep(2) # Sekunden warten
except KeyboardInterrupt:
  # CTRL-C gedrueckt
  # Reset GPIO
  print "[mcp3008] Messung abgebrochen."
  GPIO.cleanup()
# ------------------- EOF  ------------------------
```

Durch die Drehung des Stellrads des Drehpotenziometers ändern sich die Widerstandsparameter und somit auch die Werte des Spannungsteilers, die sich auf die anliegende Spannung auswirken. Diese wird anschließend mit der Referenzspannung verglichen und kann somit digital als Vergleichswert dargestellt werden.

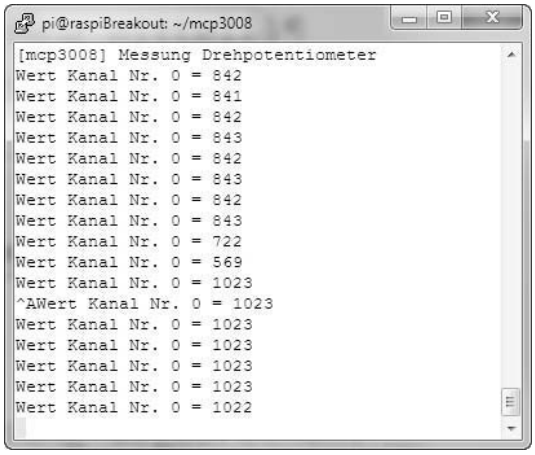



**Bild 1.9:** Stellrad am Anschlag: Mit Ausgabe des Werts `1023` erreicht das Potenziometer den höchsten digitalen Wert. Der Jitter schwankt zwischen den Werten `1022` und `1023`.

Läuft das Skript in einer Dauerschleife, kann es vorkommen, dass der angezeigte Wert etwas schwankt, obwohl das Drehpotenziometer nicht betätigt wurde. Die

Ursache ist das sogenannte Jittering, bei dem die Übergänge zwischen zwei aufeinanderfolgenden unterschiedlichen Bits für schwankende Ausgaben sorgen.

Um die Messung bzw. die Darstellung der Messwerte zu verbessern, nutzen Sie am besten eine Normalisierung im Python-Skript. Hier können Sie beispielsweise die Wartezeit in der `while`-Schleife nutzen, um mehrere Messungen zusammenzufassen und sich stattdessen anschließend den errechneten, normalisierten Mittelwert ausgeben zu lassen.

```
# Datei mcp3008-step2.py
# ---------------------------------------------
# Start Python-Skript
# ---------------------------------------------
os.system('clear')
print "[mcp3008] Messung Drehpotenziometer"
anz=3
# Drei hintereinanderfolgende Messungen werden gemittelt (anz).
try:
  while True:
    sum=0
    for i in range(anz):
        tmp = getAnalogData(poti_channel, CLK_Pin, DIN_Pin, DOUT_Pin,
CS_Pin)
        sum=(sum+tmp)
        print "Summe =", sum
        time.sleep(0.15)
    tmp_value = sum / anz
    print "Wert Kanal Nr.", poti_channel, "=", tmp_value
    time.sleep(2) # Sekunden warten
except KeyboardInterrupt:
  # CTRL-C gedrueckt
  # Reset GPIO
  print "[mcp3008] Messung abgebrochen."
  GPIO.cleanup()
# ------------------- EOF  ------------------------
```

Die angepasste Version mit den normalisierten Messwerten ist in der Projektdatei `mcp3008-step2.py` gespeichert. Im nächsten Schritt aktivieren Sie die SPI-Schnittstelle des Raspberry Pi, um Zugriff auf das angeschlossene MCP3008-IC-Modul zu erhalten.

## SPI-Schnittstelle aktivieren

Das *Serial Peripheral Interface* (kurz SPI) ist ein einfaches, serielles Protokoll, das die Kommunikation zwischen dem Raspberry Pi und dem MCP3008-IC übernimmt. Hier steuert der Master (Raspberry Pi) die Kommunikation, und der Slave (MCP3008-IC)

führt Befehle aus. Für die Kommunikation werden einfache Byte-Befehlscodes verwendet, die als Spannungsgröße codiert übertragen werden.

| SPI | Name der Signale/ Leitungen auf Master | Name der Signale/ Leitungen auf Slave |
|---|---|---|
| Taktleitung/Clock | CLK, SCLK, SCK | CLK, SCLK, SCK |
| Daten vom Master zum Slave (Schreiboperation) | MOSI, SIMO | Dateneingang eines Bauteils: SDI, DI, SI, IN |
| Daten vom Slave zum Master (Leseoperation) | MISO, SOMI | Datenausgang eines Bauteils: SDO, DO, SO, OUT |
| Slave-Ansprache/-Auswahl | SS (Slave Select), CS (Chip Select) | - |

Der Raspberry Pi besitzt zwei Slave-Anschlüsse (CE0 und CE1, Raspberry-Pi-Pin 24 und 26), mit denen zwei SPI-Geräte bei einer maximalen Taktrate von 32 MHz gleichzeitig verwendet werden können. Ist der Raspberry Pi gestartet, prüfen Sie in der Datei `/etc/modprobe.d/raspi-blacklist.conf` die Zeile, die das Modul `spi-bcm2708` »blacklistet«. Hier achten Sie darauf, dass sich vor der genannten Zeile das Rautensymbol `#` befindet, um die Anweisung auszukommentieren.

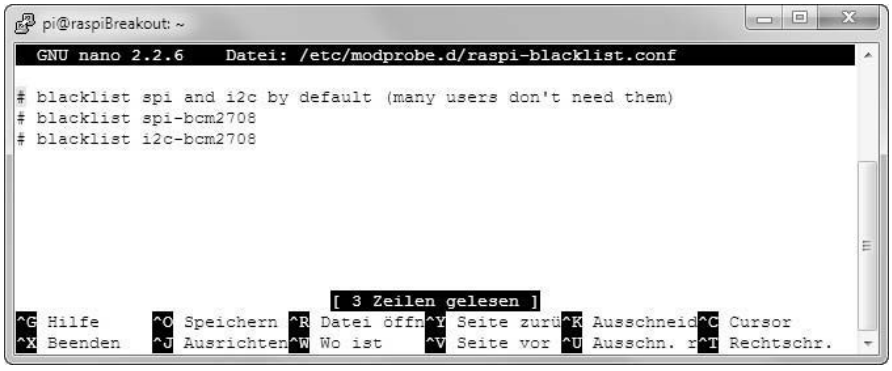



**Bild 1.10:** Mit dem Kommando `sudo nano /etc/modprobe.d/raspi-blacklist.conf` starten Sie den `nano`-Editor und fügen am Anfang der Zeile, in der sich das `spi-bcm2708`-Modul befindet, ein `#`-Symbol ein.

Nach der Änderung speichern Sie per Tastenkombination Strg+X, dann Y und schließlich Enter die Datei und starten mit dem `reboot`-Kommando den Raspberry Pi neu, damit die Änderung aktiv wird.

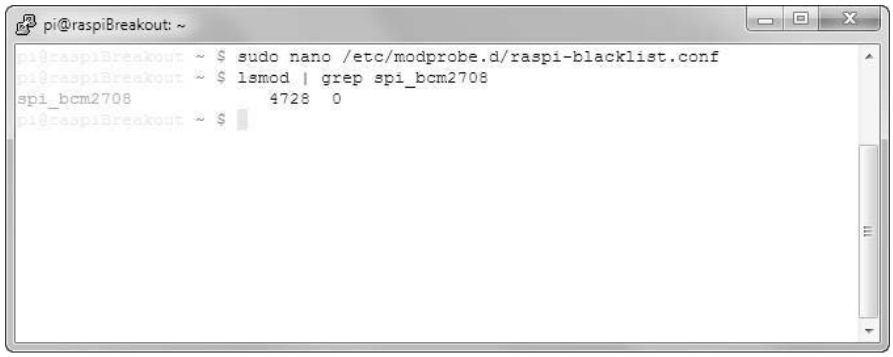



**Bild 1.11:** Nach dem Neustart des Raspberry Pi per `sudo reboot`-Kommando prüfen Sie, ob das notwendige Modul automatisch gestartet worden ist.

Um nun herauszufinden, ob das Modul betriebsbereit zur Verfügung steht, nutzen Sie den Befehl:

```
lsmod | grep spi_bcm2708
```

Anschließend sollte der Eintrag `spi_bcm2708` in der Bildschirmausgabe erscheinen. Wenn nicht, nutzen Sie den Befehl `sudo insmod spi_bcm2708`, um das Modul manuell zu starten. Ist das Modul ordnungsgemäß geladen, installieren Sie im nächsten Schritt die SPI-Erweiterung für den bequemen Zugriff auf die SPI-Schnittstelle mit Python.

## SPI-Nutzung ohne Umwege: py-spidev-Modul installieren

Warum das Rad neu erfinden, wenn dafür bereits sinnvolle und praktische Alternativen zur Verfügung stehen? Das trifft auch auf die GPIO-Schnittstelle und Erweiterungen zu, und hier existieren für Schnittstellen viele verschiedene Module und Bibliotheken, die vor allem eines gemeinsam haben: Sie sind kostenlos über die Entwicklerplattformen *github.com* oder *sourceforge.net* erhältlich und stellen einfach zu nutzende Funktionen und Schnittstellen zur Verfügung. So existiert auch für den SPI-Bus (*Serial Peripheral Interface Bus*) eine praktische Erweiterung, die Sie im Python-Skript einbauen und umgehend auf dem Raspberry Pi einsetzen können.

# Stichwortverzeichnis

## T



## U



## V



## W

E. F. ENGELHARDT

# ROBOTER MIT RASPBERRY PI

**Roboter sind längst Teil unseres täglichen Lebens, als Rasenmäher, Staubsauger oder auch in der Industrie. Für Maker sind Roboter ein optimales Betätigungsfeld: Handwerk, Elektronik und Informatik verbinden sich zur perfekten Symbiose. Der Raspberry Pi ist als vollwertiger Minicomputer das perfekte Gehirn für Ihren Roboter. E. F. Engelhardt zeigt Ihnen, wie Sie am besten die Synapsen mit Leben füllen.**

Für maschinelle Bewegung bieten sich Motoren an. Damit diese Bewegungen einer gewissen Genauigkeit folgen, sind Schrittmotoren erforderlich. Wie Sie Motoren über die GPIO ansteuern, zeigt Ihnen ein ausführliches Kapitel. Damit der Roboter nicht einfach ohne Strom stehen bleibt, lernen Sie, wie eine USV integriert werden kann. Damit ist der Tagesausflug mit Roboter gesichert.

Das erlernte Wissen setzt Engelhardt direkt in konkreten Projekten um: Bauen Sie eine eigene Pan-/Tilt-Kamera. Ein RC-Car lässt sich sehr einfach mit dem Pi steuern und der Staubsaugerroboter ist auch schnell angezapft. Mit der GPIO und dem zuvor gezeigten Basiswissen lassen sich vielfältige Roboterprojekte umsetzen.

Die LEGO®-Mindstorms-Serie stellt bereits Roboter zur Programmierung zur Verfügung. Über den BrickPi lassen sich diese sehr gut mit dem Raspberry Pi steuern – eine aufregende Kombination. Mit dem BrickPi können Sie auch wunderbar einen Roboter mit LEGO®, aber ohne LEGO®-Mindstorms, realisieren. Holen Sie die alten LEGO®-Steine hervor und machen Sie den Lötkolben heiß! Und nicht vergessen: Türe schließen, sonst läuft der Roboter raus.

## Aus dem Inhalt:

- Lenken und Steuern mit der GPIO-Schnittstelle
- SPI-Schnittstelle aktivieren
- Analog-digital-Wandler MCP3008
- Joystick-Steuerung
- Touch- und Drucksensor
- Motorsteuerung mit dem Raspberry Pi
- Motoren und Steppermotoren
- USV für den Raspberry Pi
- Pan-/Tilt-Kamera im Eigenbau
- Haushaltshilfe: Staubsauger-Modding
- Staubsauger über Raspberry Pi steuern
- Roboter über eine Webseite steuern
- Staubsaugerroboter mit dem Smartphone steuern
- Schrauben, löten, programmieren: RC-Car-Modding
- Google-Streetview-RC-Car
- LEGO®-Pi mit Mindstorms EV3 und LEGO®-Technic
- BrickPi: LEGO®-Mindstorms im Eigenbau
- LEGO®-Kran- und -greifer-Steuerung

## Über den Autor:

E.F. Engelhardt, Jahrgang 1975, hat bereits über 40 Computerbücher veröffentlicht – und keines dieser Bücher ist wie andere Computerbücher: Der Autor beginnt direkt mit der Praxis, ohne langatmige, weitschweifende und überflüssige Technikerläuterungen.

E.F. Engelhardt ist Autor des Bestsellers „Hausautomation mit Raspberry Pi". Hier hat er eindrucksvoll seine große Erfahrung mit dem Raspberry Pi gezeigt. Und er hat immer noch nicht genug: Dieses Mal werden Roboter gebaut und programmiert. Wie in allen seinen Büchern hat er die Projekte komplett selbst entwickelt. Sie haben als Leser damit die Sicherheit, dass alles funktioniert.

**Der komplette Quellcode aus dem Buch auf www.buch.cd**

9 783645 603430