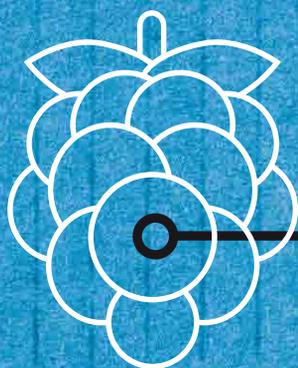
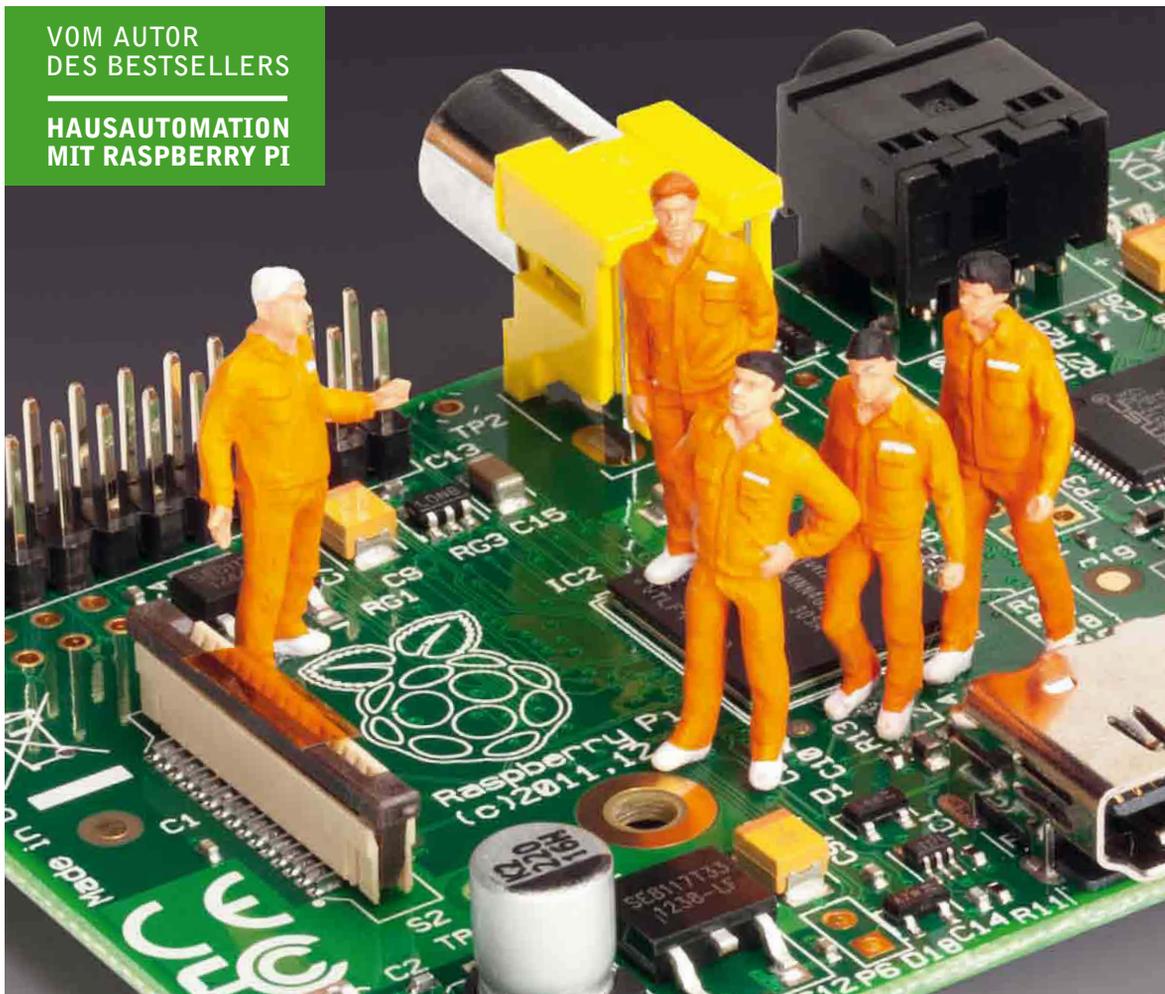


VOM AUTOR
DES BESTSELLERS

HAUSAUTOMATION
MIT RASPBERRY PI



FRUIT UP
YOUR
FANTASY

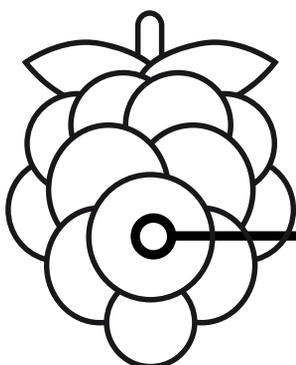
E. F. ENGELHARDT

ROBOTER MIT RASPBERRY PI

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

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

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.

Alle Angaben in diesem Buch wurden vom Autor mit größter Sorgfalt erarbeitet bzw. zusammengestellt und unter Einschaltung wirksamer Kontrollmaßnahmen reproduziert. Trotzdem sind Fehler nicht ganz auszuschließen. Der Verlag und der Autor sehen sich deshalb gezwungen, darauf hinzuweisen, dass sie weder eine Garantie noch die juristische Verantwortung oder irgendeine Haftung für Folgen, die auf fehlerhafte Angaben zurückgehen, übernehmen können. Für die Mitteilung etwaiger Fehler sind Verlag und Autor jederzeit dankbar. Internetadressen oder Versionsnummern stellen den bei Redaktionsschluss verfügbaren Informationsstand dar. Verlag und Autor übernehmen keinerlei Verantwortung oder Haftung für Veränderungen, die sich aus nicht von ihnen zu vertretenden Umständen ergeben. Evtl. beigefügte oder zum Download angebotene Dateien und Informationen dienen ausschließlich der nicht gewerblichen Nutzung. Eine gewerbliche Nutzung ist nur mit Zustimmung des Lizenzinhabers möglich.

© 2014 Franzis Verlag GmbH, 85540 Haar bei München

Alle Rechte vorbehalten, auch die der fotomechanischen Wiedergabe und der Speicherung in elektronischen Medien. Das Erstellen und Verbreiten von Kopien auf Papier, auf Datenträgern oder im Internet, insbesondere als PDF, ist nur mit ausdrücklicher Genehmigung des Verlags gestattet und wird widrigenfalls strafrechtlich verfolgt.

Die meisten Produktbezeichnungen von Hard- und Software sowie Firmennamen und Firmenlogos, die in diesem Werk genannt werden, sind in der Regel gleichzeitig auch eingetragene Warenzeichen und sollten als solche betrachtet werden. Der Verlag folgt bei den Produktbezeichnungen im Wesentlichen den Schreibweisen der Hersteller.

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

1	Lenken und Steuern mit der GPIO-Schnittstelle.....	11
1.1	Betriebssystem und Treiber aktualisieren	15
1.2	Analog-digital-Wandler MCP3008 nachrüsten	16
	Datenblatt prüfen, Funktionen verstehen.....	16
	MCP3008 auf dem Steckboard nutzen	17
	Programmierung des MCP3008 mit Python	21
	SPI-Schnittstelle aktivieren	26
	SPI-Nutzung ohne Umwege: py-spidev-Modul installieren	28
1.3	Joystick-Steuerung mit dem Raspberry Pi	30
	GPIO-Eingang schalten: Risiken und Nebenwirkungen.....	31
	Schaltungsdesign vom Steckboard auf die Rasterplatine	34
	Joystick-Steuerung mit Python.....	35
	Richtungsbestimmung mittels ADC-Werten	37
1.4	I ² C-Bus - Schnittstelle wecken und checken.....	41
	I ² C-Geräte und Raspberry-Pi-Revision.....	45
1.5	Schalten und walten mit Touchsensor	46
	Touch- und Drucksensor - Dateneingabe über den I ² C-Bus.....	47
	Flexibler Zugriff dank I ² C- und MRP121-Bibliothek.....	48
	Inbetriebnahme des MRP121-Touchsensors	50
2	Fahren und bremsen - Motorsteuerung mit dem Raspberry Pi.....	53
2.1	Die erste Schaltung - LEDs mit ULN2803A steuern	53
2.2	GPIO-Steuerung über die Konsole und Python	56
	Schalten per Konsole	57
2.3	Motoren und Steppermotoren.....	59
	Oft vernachlässigt: Spannungsversorgung des Motors	62
2.4	Motorsteuerung versus Motortreiber	63
	Mehr Kontrolle - Schrittmotorcontroller	64
2.5	Unipolaren Steppermotor mit ULN2803-IC steuern	65
	Schaltung auf Steckboard umsetzen.....	65
	Vollschritt- vs. Halbschrittverfahren im Detail	69
	Schritt für Schritt: Vollschritt- und Halbschrittverfahren einsetzen	70
	Vorwärts- und Rückwärtsbewegungen	74
2.6	Praktisch und sicher - USV für den Raspberry Pi.....	76
	Pi USV in Betrieb nehmen	77

	Ohne Strom nix los – Akkupack auswählen.....	78
	Pi-USV-Software in Betrieb nehmen	79
	Status der Pi USV erkennen.....	81
	Status der Pi USV mit Python auslesen	82
3	Pan/Tilt-Kamera im Eigenbau	85
3.1	Raspberry-Pi-Kamera im Robotik-Einsatz.....	86
	Kameramodul mit dem Raspberry Pi koppeln.....	86
	Inbetriebnahme per Software	87
	raspistill – Fotografieren über die Kommandozeile	91
	LED abschalten und heimlich fotografieren	93
	Programmierung der Raspberry-Pi-Kamera	93
3.2	Einzellösung: Tower-SG90-Servomotor.....	96
3.3	Hardware-PWM-Ausgang mit LED testen.....	99
3.4	Servoblaster-Treiber installieren.....	101
3.5	Motoren mit Servoblaster in Betrieb nehmen.....	103
3.6	Servomotor mit Python steuern	105
3.7	Pan/Tilt-Achse und Kamera steuern.....	106
3.8	Steuerung der Raspberry-Pi-Kamera.....	108
3.9	Bewegungen und Aufnahmen steuern.....	109
3.10	Hürden bei der Inbetriebnahme umgehen	114
	Automatischer Log-in: pi vom Start weg.....	114
	Autostart nach dem Einschalten.....	115
4	Haushaltshilfe: Staubsauger-Modding	117
4.1	Vorwerk vs. Neato – mehr als nur eine Kopie	118
	Einrichtung und Treiberinstallation	119
	Zugriff über PuTTY auf das Betriebssystem.....	124
4.2	Staubsauger über Raspberry Pi steuern	126
	Staubsaugerroboter mit Raspberry Pi verbinden	126
	minicom-Modemzugang zum Staubsauger einrichten.....	129
	minicom-Steuerung für den Staubsauger	131
	Staubsaugerkommandozeile im Überblick.....	134
	Python-Programmierung über python-serial	135
	Spazierfahrt mit der Kommandozeile – Staubsauger fortbewegen	138
	Zeitplanung für den Staubsauger.....	140
4.3	Staubsauger und Raspberry Pi koppeln.....	144
	Aufwecken aus dem Schlafmodus	144
	USB-Geräte über GPIO schalten.....	146
	Staubsauger mit dem Raspberry Pi verbinden.....	148
	Schaltung über Kommandozeile prüfen.....	150

4.4	Roboter über die Webseite steuern	151
	Python-Zugriff über Browser - Bottle im Einsatz	152
4.5	Videostreaming installieren und einbinden	158
	Streaming-Werkzeug laden und installieren	159
	MJPEG-Streamer als Live-View-Quelle	162
	Live-View und Steuerung verheiraten	165
	Fotografieren mit dem Vorwerk/Neato-Staubsauger	170
4.6	Drahtlos-Raspberry-Pi einrichten	171
	Raspberry Pi mit drahtloser Stromversorgung	172
	Akkupack und USV für Raspberry Pi kombinieren.....	173
	WLAN-Netzwerk einrichten und Verbindung aufnehmen	174
	Umschalten zwischen WLAN-Verbindungen	178
	WLAN-Verbindung mit Python steuern.....	180
4.7	Staubsaugerroboter mit dem Smartphone steuern.....	183
	USB-Debugging-Modus - Smartphone einrichten	183
	Staubsaugerroboter mit dem Smartphone koppeln	185
5	Schrauben, löten, programmieren: RC-Car-Modding	187
5.1	Basis für das RaspiRoboCAR-Projekt.....	188
5.2	Lenken und Steuern über die Tastatur	191
5.3	Google-Streetview-RC-Car mit der Raspberry-Pi-Kamera	203
6	LEGO® Pi mit Mindstorms EV3 und LEGO®-Technic	205
6.1	Viel kreativer Spielraum für Technikfantasien	205
6.2	LEGO®-Technic und LEGO®-Mindstorms mit Raspberry Pi aufmotzen	208
6.3	BrickPi: LEGO®-Mindstorms im Eigenbau	209
	BrickPi-Treiber in Betrieb nehmen.....	210
	BrickPi-Schnittstellen aktivieren.....	212
	Python-Bibliothek für BrickPi installieren.....	213
	Motoren und Sensoren im BrickPi-Einsatz.....	215
6.4	Legokran- und -greifer-Steuerung mit dem Raspberry Pi.....	216
	Basis, Neigung und Greifer: drei Motoren für den Kran	218
6.5	LEGO®-Modding: Mindstorms im Eigenbau.....	224
	LEGO®-Steine mit LED-Birnen nachrüsten	225
	Servomotor-Modding für LEGO®-Technic	226
	LEGO®-Extrem-Modding: bis zu 16 Servomotoren steuern.....	228
	Adressbelegung für den Anschluss am I ² C-Bus.....	230
	Mehrere Servomotoren im Zusammenspiel	232

A Anhang	239
A Python-Basics auf dem Raspberry Pi.....	239
LED-Steuerung mit Python	241
Schneller Zugriff über die Wiring-Pi-API.....	245
Raspberry-Pi-Revision 2: zusätzlicher GPIO-Sockel	248
 Stichwortverzeichnis	 253



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.

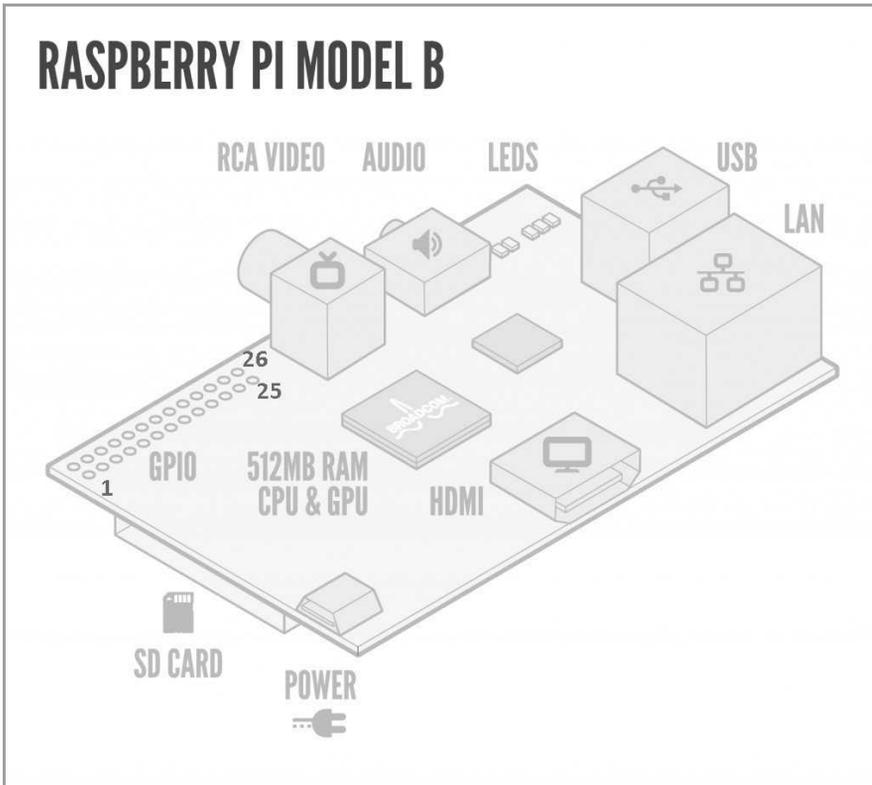


Bild 1.1: Der grundsätzliche Aufbau des Raspberry Pi und der GPIO-Pinleiste Modell B, Revision 2. (Grafik: raspberrypi.org)

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.

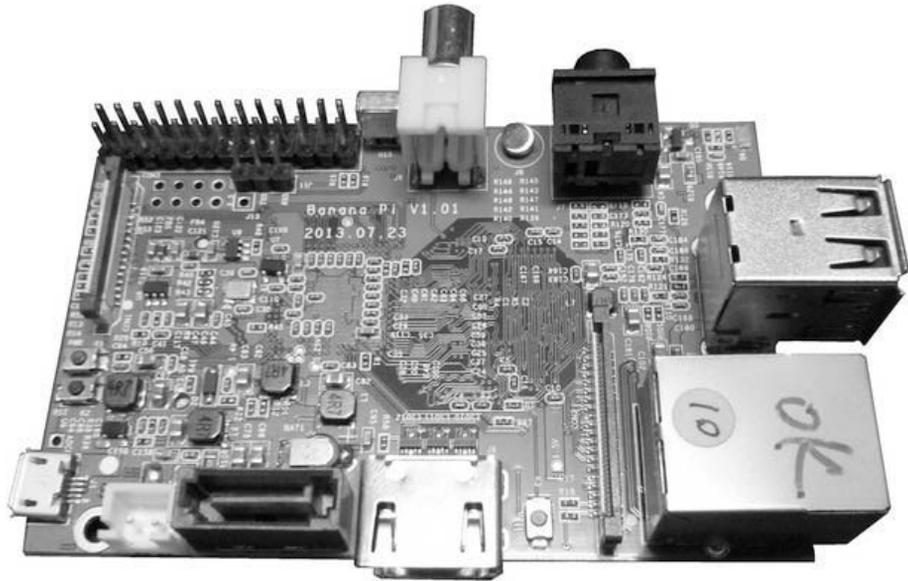


Bild 1.2: Stark verbesserter Raspberry-Pi-Klon, der nahezu die gleichen Abmessungen wie das Original besitzt und ebenfalls eine Pin-kompatible GPIO-Leiste bietet. (Abbildung: *Bananapi.org*)

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.

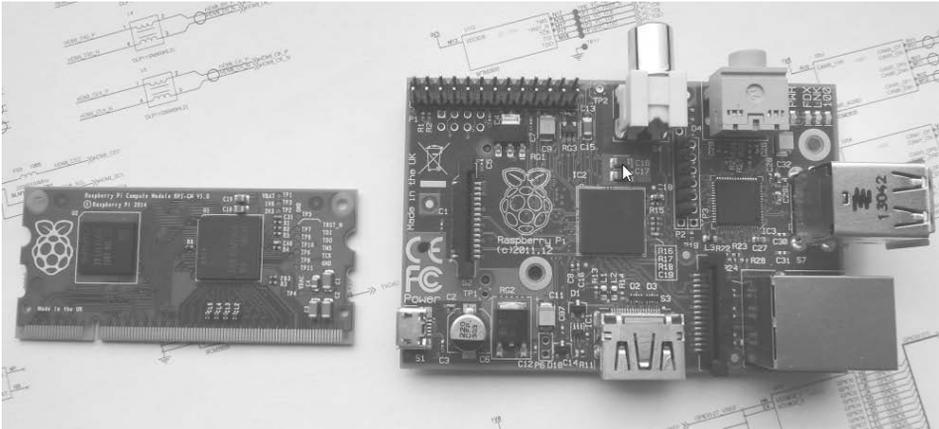


Bild 1.3: Die Raspberry Pi Foundation stellte im April 2014 das deutlich geschrumpfte Raspberry-Pi-Modul vor, das sich nun besser für kleinere, kompakte Anwendungen eignen soll. (Abbildung: Raspberry Pi Foundation)

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 $67,7 \times 30$ 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.

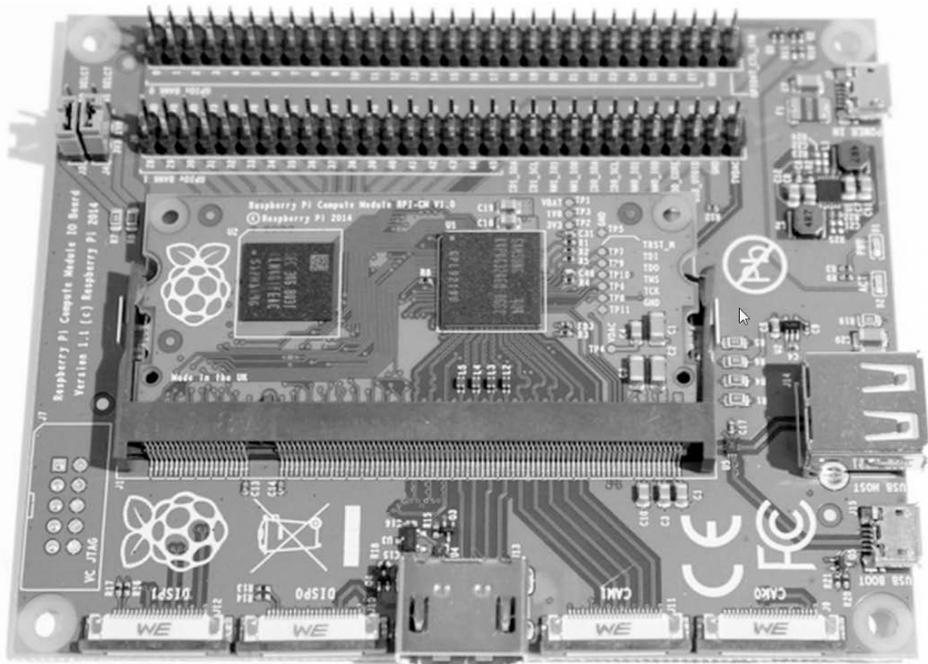


Bild 1.4: Das Raspberry-Pi-Modul wird in das flachere Breakout-Board eingesetzt, was bei mobileren Robotik-Anwendungen manchmal sinnvoller sein kann, wenn die Bauhöhe begrenzt ist. (Abbildung: Raspberry Pi Foundation)

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.

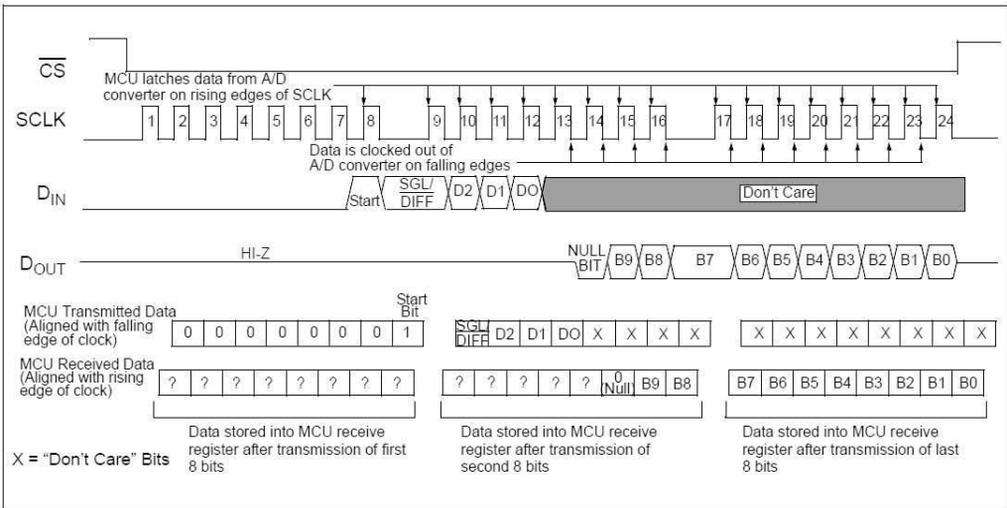


Bild 1.5: In der obersten Zeile des Datenblatts in der Abbildung sehen Sie das *Chip-Select-Signal*, das bei der dargestellten fallenden Flanke von HIGH auf LOW geht.

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/Bestellnummer	Preis
Drehpotenziometer 10 k Ω	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.

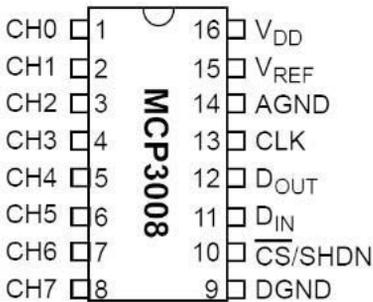


Bild 1.6: Zeigt die Kerbe des MCP3008-IC nach oben, erfolgt die Zählung der Anschlüsse wie beim Schreiben des Buchstabens U von links oben nach unten zu Anschlusspin 8, um dann wieder von rechts unten (Pin 9) nach oben zu Pin 16 zu gelangen.

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 $-0,6\text{ V}$ bis $V_{\text{DD}}+0,6\text{ V}$ 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, 2¹⁰), die anschließend an die Datenschnittstelle übergeben wird.

Bemerkung	Bezeichnung	Pin	0	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.

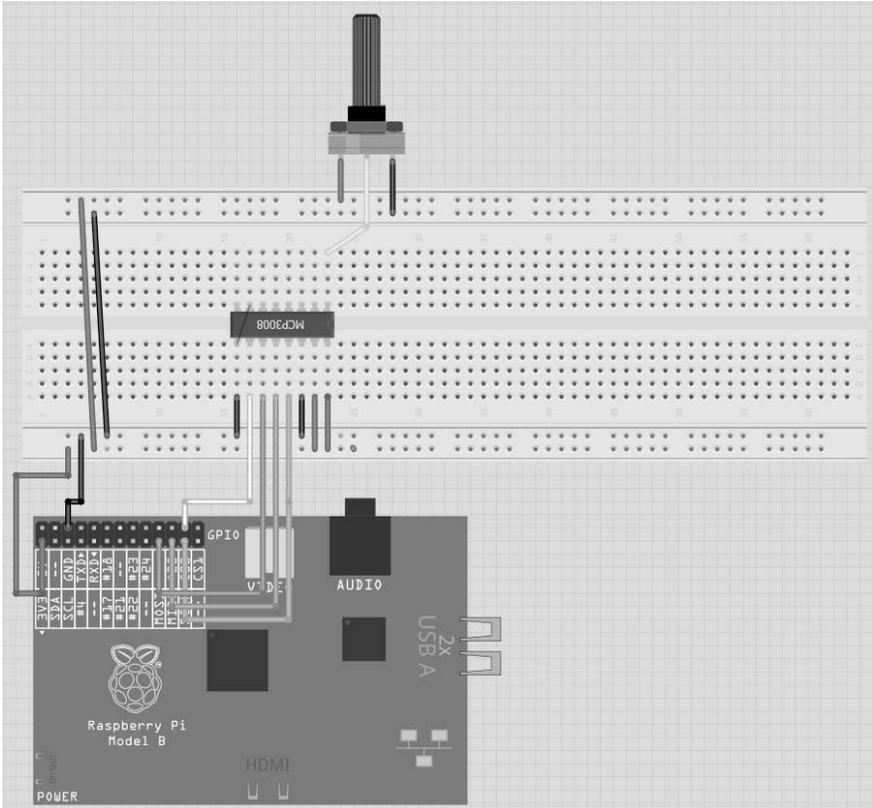


Bild 1.7: Das notwendige Potenziometer (oben) wird außen an Masse (-) angeschlossen und mit dem 3,3-V-Anschluss des Raspberry Pi (Rev. 2) verbunden. Der Mittelanschluss des Potenziometers wird an Eingang **CH0** – also Pin 1 des MCP3008-IC – angeschlossen.

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.

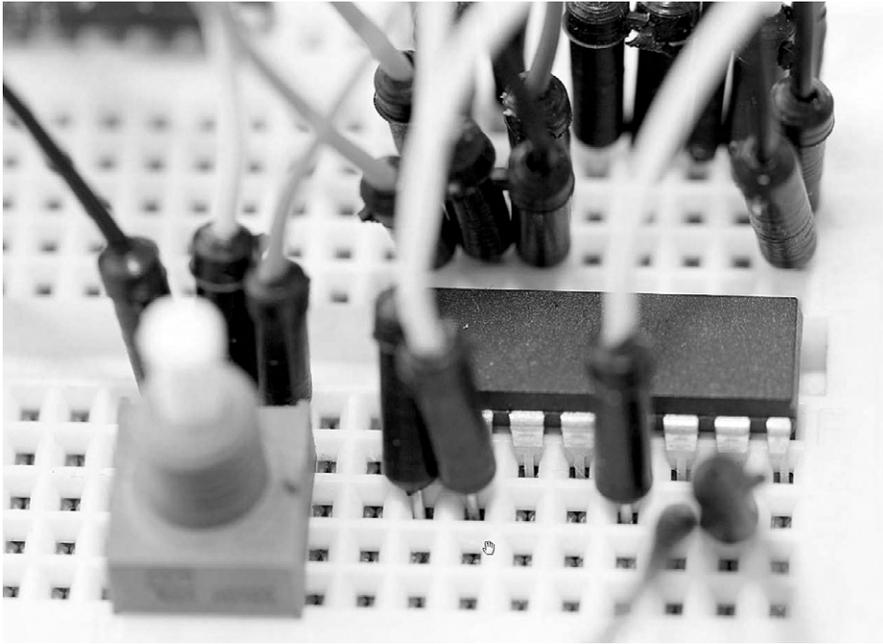


Bild 1.8: Raspberry Pi im SPI-Einsatz: Auf das Steckboard ist zusätzlich noch ein Drehpotenziometer gesteckt, dessen Mittelanschluss mit dem MCP3008-Pin 1 (Kanal CH0) verbunden ist.

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 Eingangsspannung 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 ($3,3 \text{ V} / 1024$), 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 k Ω	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}	-	GPI010/SPI MOSI	19	12
12	D _{OUT}	-	GPI09/SPI MISO	21	13
13	CLK	-	GPI011/SPI0_SCLK	23	14
10	CS/SHDN	-	GPI08/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 Signalfanke (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
<code>0b00011000</code>	000 gibt die verwendete Kanaladresse (CH0) in binärer Schreibweise an.
<code>0b00011000</code>	SGL-(Signal-)Bit.
<code>0b0001100</code>	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
DOOUT_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(DOOUT_Pin, GPIO.IN)
GPIO.setup(CS_Pin, GPIO.OUT)
# -----
# Funktionen
#
def getAnalogData(poti_channel, CLK_Pin, DIN_Pin, DOOUT_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-Abwurf 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

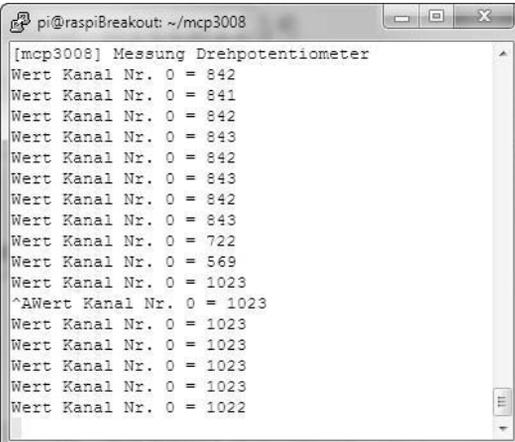
```

```

        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.



```

pi@raspiBreakout: ~/mcp3008
[mcp3008] Messung Drehpotenziometer
Wert Kanal Nr. 0 = 842
Wert Kanal Nr. 0 = 841
Wert Kanal Nr. 0 = 842
Wert Kanal Nr. 0 = 843
Wert Kanal Nr. 0 = 842
Wert Kanal Nr. 0 = 843
Wert Kanal Nr. 0 = 842
Wert Kanal Nr. 0 = 843
Wert Kanal Nr. 0 = 843
Wert Kanal Nr. 0 = 722
Wert Kanal Nr. 0 = 569
Wert Kanal Nr. 0 = 1023
^A Wert Kanal Nr. 0 = 1023
Wert Kanal Nr. 0 = 1022

```

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.



```

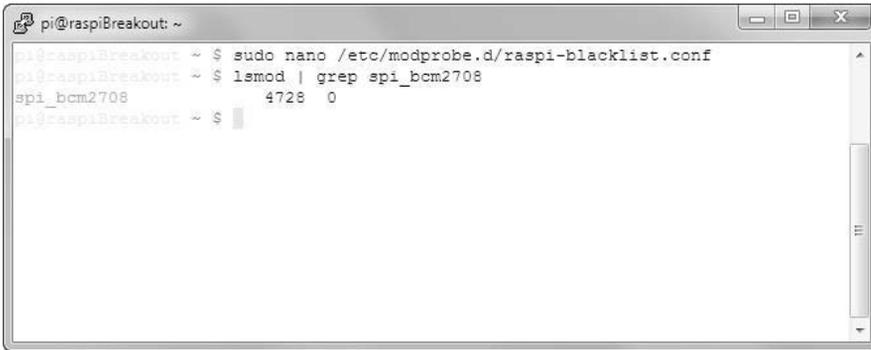
pi@raspiBreakout: ~
GNU nano 2.2.6 Datei: /etc/modprobe.d/raspi-blacklist.conf
# blacklist spi and i2c by default (many users don't need them)
# blacklist spi-bcm2708
# blacklist i2c-bcm2708

[ 3 Zeilen gelesen ]
^C Hilfe      ^O Speichern  ^R Datei öffn  ^Y Seite zurü  ^K Ausschneid ^C Cursor
^X Beenden    ^J Ausrichten ^W Wo ist      ^V Seite vor  ^U Ausschn.   ^T Rechtschr.

```

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.



```
pi@raspiBreakout: ~  
pi@raspiBreakout ~ $ sudo nano /etc/modprobe.d/raspi-blacklist.conf  
pi@raspiBreakout ~ $ lsmod | grep spi_bcm2708  
spi_bcm2708          4728  0  
pi@raspiBreakout ~ $
```

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

/

/dev/servoblaster 104
/dev/ttyACM0 129
/etc/modprobe.d/raspi-blacklist.conf 43
/etc/rc.local 114

A

A4988 64
Adafruit 103
ADC-Werte 37
Analog-digital-Wandler 16
Arduino 77
Aufnahmen, mit Skript steuern 109

B

Banana-Pi-Lösung 12
basis_turn_left() 219
basis_turn_right() 219
bcm_host.h 160
Beaglebone 77
Betriebssystem
 auffrischen 87
 update 15
 upgrade 15
Bewegungen, mit Skript steuern 109
Bottle 152
bottle.py 152
bottle_pisauger.py 167
BrickPi 209
BrickPi, Treiber 210

C

Cable Select 22
chmod 43
clean 132

Clean 134
cleanup 76
con 136
con.write 136
config.txt 93
cpuinfo 45
CSI/DSI-Schnittstellen 13
CSI-Anschluss 86

D

DiagTest 134
dir 197
direction 57
dmesg 42, 127
dpkg 79
Drahtlose Stromversorgung, Raspberry Pi
 172
Drehmoment 62
Drehpotenziometer 18, 25
Duracell Powermat 172

E

echo 104
Elektrolytkondensator 229
Enable Camera 90

F

feedback 137
Firmware 161
Firmware auffrischen 87
Freilaufdioden 62
Frequenz 100
fsck 76
Funkauto 187

G

GetAccel 134
 getadcChannel 35
 GetAnalogSensors 134
 GetSchedule 134, 140
 GetVersion 134
 git 29
 git pull origin 246
 GitHub 94, 152
 Google-Streetview-RC-Car 203
 GPIO, Wiring Pi steuern 250
 GPIO.cleanup() 244
 GPIO-Bibliothek 239
 GPIO-Eingang 31
 GPIO-Schnittstelle 11
 GPU 90

H

H.264 92
 Halbschrittbetrieb 70
 Halbschrittverfahren 70
 help 132
 Help 134
 help SetSchedule 141

I

i2c-bcm2708 42
 I²C-Bus 41
 i2cdetect 44, 48
 i2c-dev 42
 i2c-tools 43
 if-Anweisung 51
 import-Kommando 29
 inittab 114
 input_file.so 159
 input_raspicam.so 159

J

Joystick-Modul 30
 Joystick-Steuerung 35

K

Kameramodul 86
 Kamerasensor 87
 Klonplatinen 12
 Kommandozeile 91
 Kransteuerung 216

L

L293D 63
 L298N Dual H-Bridge 189
 LED 54, 225
 LED abschalten 93
 LED-Lämpchen 225
 LEGO[®] 205
 LEGO[®]-Extrem-Modding 228
 LEGO[®]-Mindstorms 208
 LEGO[®]-Mindstorms 205
 LEGO[®]-Mindstorms, Bauanleitungen 216
 LEGO[®]-NXC-Sprache 208
 LEGO[®]-Servomotor 226
 LEGO[®]-Technic 208
 LEGO[®]-Technic, Bauanleitungen 216
 Legoelektrik 208
 Legomotoren 208
 Legosensoren 215
 Lenken 191
 lsusb 126, 150

M

make 246
 MCP3008, Programmierung 21
 MCP3008-IC 16, 18
 Microsoft Robotics Developer Studio 209
 minicom 129
 minicom-Emulationsfenster 133
 Mini-USB-Hub 148
 MISO 17
 mjpg-streamer 159
 MJPG-Streamer 158
 modprobe 43

MOSI 17
Motor 59
 bipolar 59
 Spannungsversorgung 62
 unipolar 59
Motorcontroller 63, 190
motorrichtung 197
Motorschaltung 53
Motorsteuerung 63
Motortreiber 63
moveCam 112
Multimeter 60

N

Neato 118
NiMH 77

O

Oszilloskop 99

P

Pan/Tilt-Kamera 85
Pan/Tilt-Kameraarm 109
PCA9685 229
Phasenwinkel 100
pi 109
Pi USV 77
Pololu A4988 64
Pull-up-Schaltung 36
Pulsweite 97
Pulsweitenmodulation 97
PuTTY 126
pwm 106
PWM 97
pwm-servo.py 105
Python 105, 239
python-serial-API 135

R

Raspberry Pi
 Betriebssystem 15

 drahtlose Stromversorgung 172
 Modell B 11
 Revision 2 248
Raspberry Pi Foundation 14
Raspberry-Pi-Kamera 86
Raspberry-Pi-Kamera, Programmierung
 93
Raspbian 90
raspi-blacklist.conf 43
raspi-config 89
RaspiRoboCAR 188
raspistill 91, 92, 94
raspistill 109
raspid 91, 94
RC-Car 187
reboot 27
Referenzspannung 25
Relais 147
Repository-Verwaltung 94
Richtungsbestimmung 37
Roboterarm 45544 217
Rotor 59
RPi.GPIO 97

S

Schrittmotorcontroller 64
seq 70
Serial Peripheral Interface 26
Servoblaster 103
Servomotor 86, 96, 103, 224
 Python 105
 Servoblaster 101
 steuern 105
setmode 242
SetMotor 138
SetSchedule 134
SetSystemMode 134
SetTime 134
SetWallFollower 134
Skript
 gpiostart.sh 58

- mjpeg.sh 167
- pantilt-cam.sh 109
- pwm-led.py 99
- Software-PWM 101
- SourceForge 94
- Spannungsversorgung 62
- SPI-Library 22
- SPI-Schnittstelle 17
- Spulenwiderstand 64
- start 106
- Stator 59
- Staubsauger 117
- Staubsauger, minicom-Steuerung 131
- Staubsaugerroboter
 - am Raspberry Pi 126
 - Treiber 120
- stepCount 70
- stepCounter 70
- stepCounterSum 70
- Steppermotor 59, 62
- Steuern 191
- Streetview-RC-Car 203
- Stromversorgung 77
- sudo reboot 28

T

- takeSnapshot 112
- tar 246
- Tastatur
 - lenken 191
 - steuern 191
- Terminal-Emulation 135

- testmode 137
- TestMode on/off 135
- Touchsensor 47
- Tower SG90 96
- TowerProMG995 224
- Türspion 93

U

- uln_clean_motor_gpio.py 68
- uln_full_half.py 70
- ULN2803A-IC 54
- ULN2803-IC 63, 188
- Upload 135
- USB-Kabel 148
- usbserial 127

V

- value 57, 58
- Versionsstand 89
- Vollschrittverfahren 69
- Vorwerk 118

W

- Web Server Gateway Interface 152
- wget 245
- while 50
- Wiring Pi 246
- Wiring-Pi-API 244
- wpa_passphrase 178
- write 151
- WSGI 152



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: Türen 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 langweilige, weitschweifende und überflüssige Technikerfäulungen.

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



Besuchen Sie
unsere Website
www.franzis.de

FRANZIS