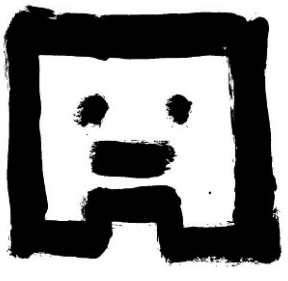
embedded projects JOURNAL
OPEN SOURCE SOFT-AND HARDWARE PROJECTS

Eine Open-Source Zeitschrift
zum Mitmachen!

all embedded



[PROJECTS]

- Die GNUBLIN Distribution
- WaWision - stiller Start
- Bildverarbeitung - eingebettete Systeme
- Prototyping Board Raspberry Pi
- Funkübertragung mit CC2500
- Entwicklungsumgebung DAVE von Infineon
- „Berrybush Pi“

creative commons

firma.embedded-projects.net

DAS HARDWARE FOR YOUR PROJECTS-PORTAL

Wussten Sie, dass wir eine Firma für kundenspezifische Entwicklungen mit Sitz in Augsburg sind?

Wir bieten:

Hardware, Software, Embedded, Software-Entwicklung, Mikrocontroller, Anwendungsentwicklung, Fachbeiträge/ Literatur, Schaltplan, Webentwicklung, Open-Source, E-Commerce, Platinen-layout, GNU/Linux

Kommen Sie vorbei!

embedded projects GmbH
HARDWARE FOR PROJECTS

Holzbachstraße 4, D-86152 Augsburg
Tel +49 (0) 821 279599-0
Fax +49 (0) 821 279599-20
info@embedded-projects.net

Anzeige

Einleitung

Ausgabe 01/2013

Embedded Projects Journal - Ausgabe No. 16

Einleitung

Diese Ausgabe ist für uns eine ganz besondere. Nach vier Jahren langer Arbeit haben wir uns endlich einen gemeinsamen Firmenurlaub verdient :-). Drei Tage „All Inklusive“ embedded world. Wir sind dieses Jahr mit einem eigenen Stand (Halle 5-414) auf der embedded world in Nürnberg vetreten.

Neben unserem Journal haben wir Gnublin und unser Open-Source Warenwirtschaftssystem WaWision dabei. Wer auch mal Prof. Dr. Hubert Högl von der Hochschule Augsburg kennenlernen möchte, kann die Chance nutzen, denn er ist ebenfalls die komplette Zeit mit dabei.

Wir freuen uns auf viele Gespräche, Ideen und Anregungen.

Auf unserer Homepage [1] kann man sich bis zur Messe kostenlose Tickets zuschicken lassen, solage der Vorrat reicht.

Benedikt Sauter

und das embedded projects Team!

[1] http://shop.embedded-projects.net

Hinweis: Zu jeden Artikel gibt es wieder einen Forumsbeitrag zur Diskussion

http://journal.embedded-projects.net/forum.php (Weiterleitung zu Mikrocontroller.net)

→shop.embedded-projects.net

HARDWARE FOR YOUR PROJECTS – ONLINESHOP

Design your GNUBLIN

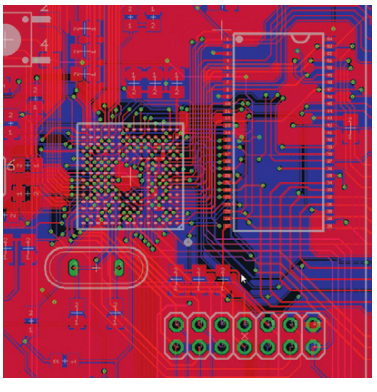
Sie suchen ein Board, das fast so wie GNUBLIN ist und brauchen davon nur eine kleine Menge pro Jahr? Wir passen Ihnen auf kurzem Weg die Schaltung an und ergänzen diese nach Ihrem Wunsch. Dank unserer internen Bestückung können wir Ihnen bereits ab kleinen Mengen ähnliche Stückpreise wie bei den GNUBLIN Boards in diesem Shop liefern.

→ **www.gnublin.org/designer**

Die einfachste Möglichkeit an ein eigenes embedded GNU/Linux Board zu kommen!

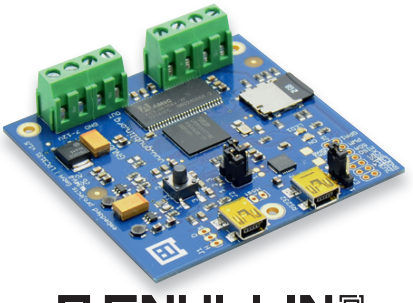
Einfach und genial ist das Produkt „Design your GNUBLIN“. Für alle, die kleine Serien von Platinen mit embedded GNU/Linux benötigen. Basierend auf der Schaltung der Boards der GNUBLIN Familie können wir einfach und schnell kundenspezifische Lösungen erstellen. Innerhalb kürzester Zeit können wir Ihnen die Boards von 1 bis ca. 1000 Stück liefern.

NEU: online GNUBLIN-Designer mit Kalkulator



Wir zeichnen den Schaltplan und das Layout ...

... bestücken mit einem Automaten in Augsburg ...



... und nehmen die Schaltung für Sie in Betrieb.

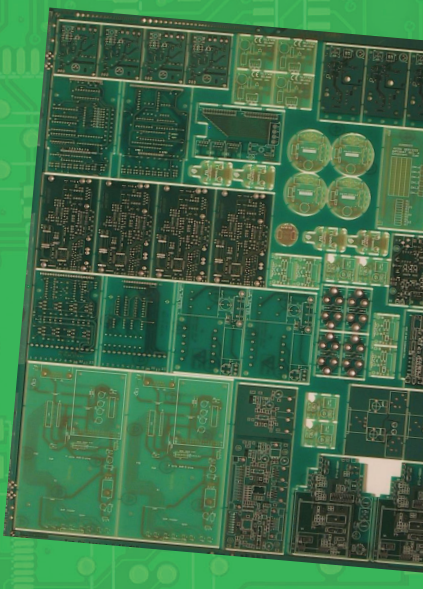
Holzbachstraße 4, D-86152 Augsburg
Tel +49 (0) 821 279599-0
Fax +49 (0) 821 279599-20
shop@embedded-projects.net

embedded projects GmbH

HARDWARE FOR PROJECTS

EURO CIRCUITS

Die Europäische Referenz für PCB Prototypen und Kleinserien



Kostensenkung durch Online-Pooling

- Keine Einrichtkosten
- Keine Mindesbestellwerte - ab der 1. Platine
- Sofortbestellung Online - ohne Vorkasse

Zeitgewinn durch Online-Daten-Check

- PCB Visualizer® - Sofort Design-Rule-Check
- Online Tipps für DFM (Design-For-Manufacturability)
- Online-Optimierung mit Galvano-Simulation

PCB proto – *spezieller Prototypen-Service für Entwickler, preiswert und schnell*

- 1 oder 2 LP in 2, 3, 5 oder 7 Arbeitstagen
- DRC-geprüft, professionelle Ausführung inkl. 2x Lötstopplack und 1x Bestückungsdruck, 150µm Technologie
- 1 x 100 x 80mm in 7AT - 2 Lagen 46.26 € - 4 Lagen 93.94 €
- 2 x 100 x 80mm in 7AT - 2 Lagen 36.28 € je LP - 4 Lagen 73.52 € je LP

Preise inkl. 19% MwSt und ohne Transportkosten

STANDARD pool – *die größte Auswahl an Eurocircuits Pooling Optionen*

- 1-8 Lagen 150µm Technologie-Leiterplatten
- ab 2 AT

TECH pool – *100µm-Technologie mit allen Pooling-Vorteilen*

- 2-8 Lagen 100µm Technologie-Leiterplatten
- ab 4 AT

IMS pool – *Aluminiumkern-Leiterplatten für hohe Wärmeableitung (z.B. LED-Anwendung)*

- Leiterplatten mit einlagig isoliertem Metallsubstrat
- ab 3 AT

On demand – *Alle Optionen im Nicht-Pooling für Spezialanwendungen*

- 1-16 Lagen bis 90µm-Technologie
- ab 2 AT

www.eurocircuits.de

Die GNUBLIN Distribution

Benedikt Sauter <sauter@embedded-projects.net>

Einleitung

Bis zum aktuellen Zeitpunkt hatten wir für GNUBLIN als Linux immer ein rudimentäres Dateisystem, das von der Hochschule Augsburg einmalig erstellt und von diesem Punkt aus regelmäßig von verschiedenen Stellen aus erweitert wurde. Es gab jedoch keinen gemeinsamen reproduzierbaren Nenner. Nach und nach entstand bei uns allen aber immer mehr der Wunsch eine eigene Distribution zu haben. Unsere Wünsche waren:

- Standard Debian als Basis
- Nightly Build (automatischer Buildprozess für die Distribution jede Nacht)
- GNUBLIN Tools für einfache Ansteuerung alle Peripherie (später API)

Perfekte Zusammenarbeit mit dem Installer um sich einfach eine neue Installation auf einer SD-Karte erstellen zu können.

Die GNUBLIN Distrubution bietet jetzt eine gemeinsame Plattform für alle Anwender des freien Linux Boards. Als Basis wird ein Debian Linux mit angepasstem Kernel, Bootloader etc. angeboten, um eine hoffentlich gut getestete und einfache Plattform für Entwicklungen und Bastelein zu haben.

Abb. 1: Die Gnublin Distribution basierend auf Debian

Installer / Quelltexte

Im Wiki[1] ist die ganze Distribution wesentlich ausführlicher beschrieben. Dennoch möchte ich kurz einen Einstieg zeigen. Im Wesentlich gibt es drei Nutzergruppen:

1) Den, der nur einfach eine neue SD-Karte erstellen möchte
2) Den, der Kernel, Bootloader und Programme selbst übersetzten können möchte
3) Den, der eine eigene Distribution pflegen möchte

Die Gruppe 1 ist wohl die größte Gruppe. Hier möchte man sich schnell und einfach eine SD-Karte mit der neusten Distribution erstellen können. Das geht mit unserem Installer ganz leicht. Läuft der Installer, zieht dieser sich automatisch das Image des Dateisystems, Kernels und Bootloaders von unseren Servern, entpackt diese lokal und kopiert sie auf eine (ebenfalls vom Installer) frisch partitionierte SD-Karte. Nach ca. 10 – 15 Minuten ist die SD-Karte erzeugt und kann direkt verwendet werden.

Die Installation ist in [2] beschrieben. Unter einem Ubuntu-System sieht es aktuell so aus:

wget http://gnublin.googlecode.com/files/gnublin-installer-1.3.1.1-beta-i386.deb

Anschliessend muss man es nur noch installieren:

```
sudo dpkg -i
gnublin-installer-
1.3.1.1-beta-i386.deb
```

Die Gruppe 2 benötigt nicht nur den Installer, sondern auch die Toolchain und Quelltexte des Kernels, Bootloader und ausserdem die Möglichkeit, sich ein Dateisystem erstellen zu können. Im Wiki ist diese Umgebung ebenfalls beschrieben [3].

Und schließlich die Gruppe 3, zu denen wir unter anderen auch zählen. Wir wollen regelmäßig eine neue Distribution mit den vielen Updates herausbringen. Wer sich dafür interessiert, findet die Infos unter dem Punkt „Nightly Build“ natürlich ebenfalls im Wiki und GIT.

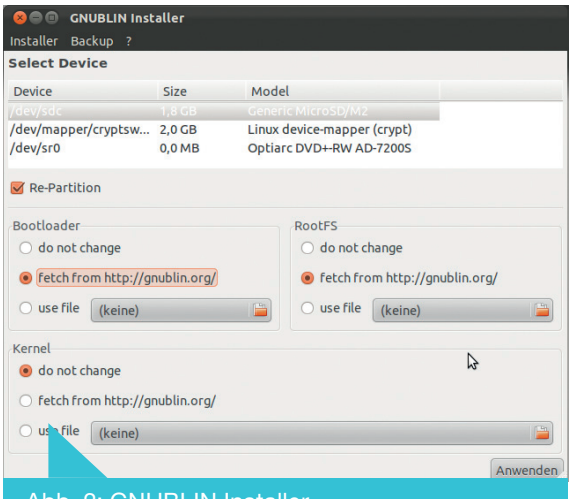


Abb. 2: GNUBLIN Installer

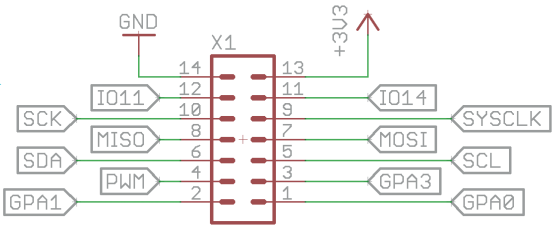
Messen, Steuern und Regeln

Warum verwendet man ein embedded GNU/Linux Board? Wahrscheinlich hauptsächlich, um kleine Steuerungen oder Messsysteme zu bauen. Mit dem Webserver etwas über ferne anzusteuern, zu überwachen, Daten zu erfassen oder per Mail zu versenden. Immer dann also, wenn man pysikalische Größen erfassen will oder ein Board in ein Gerät einbettet.

Genau dafür haben wir passend zum

Gnublin viele kleine Module entwickelt. Die Idee dabei ist, dass es zu jedem Modul eine 100%-ige Unterstützung in der GNUBLIN Distribution gibt. Das bedeutet: das Modul kann man einfach anstecken und mit einem passenden Befehl der mitgeliefert wird sofort ansprechen. Im Community-Wiki [1] gibt es vom Anfänger bis zum Forgeschrittenen die passenden Informationen.

Abb. 3: 14-poliger Stecker für Gnublin



Angebunden werden diese Module über einen 14-poligen für GNUBLIN standardisierten Stecker. Auf dem Stecker befindet sich I2C, SPI, GPIO-Pins, ein PWM-Ausgang, GND und 3.3V.

Ziel ist es eine große Vielfalt an Modulen zusammen zu bringen. Aktuell sieht die Liste wie folgt aus:

- Relaiskarte (Abb. 5)
- Temperatursensor LM75 (Abb. 7)
- Portexpander
- AD-Wandler Karte
- Schrittmotor Steuerung (Abb. 6)
- Display Einheit (Abb. 4)
- Zusätzliche RS232 Schnittstellen
- RS485
- CAN mit socket can Unterstützung
- Echtzeituhr mit Batteriepuffer

Für weitere Ideen und Anregungen haben wir immer ein offenes Ohr.

Abb. 6: GNUBLIN Schrittmotor

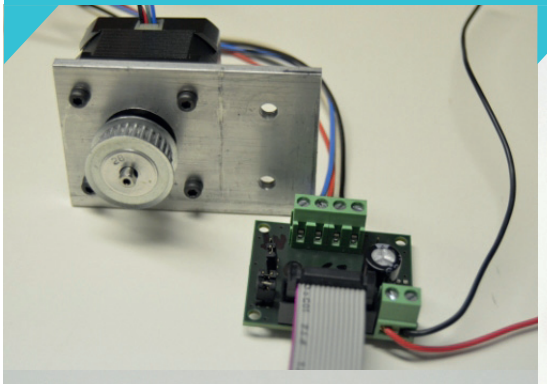
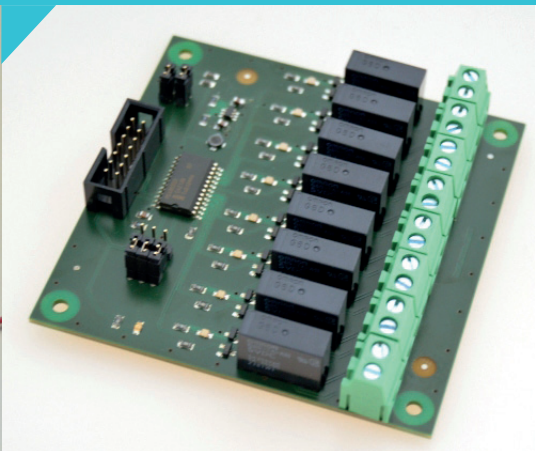


Abb. 5: GNUBLIN Relais



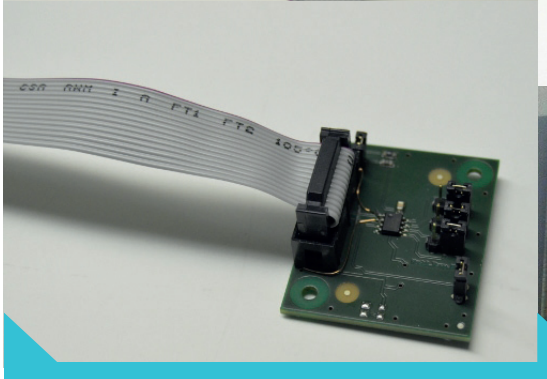


Abb. 7: GNUBLIN Temperatursensor

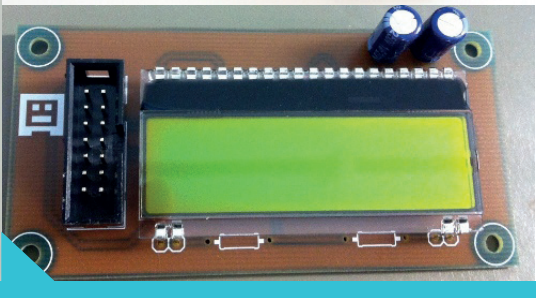


Abb. 4: GNUBLIN Display

Die Gnublin-Tools

Egal ob Temperatur, Schrittmotor, Display, AD-Wandler, PWM, SPI-Nachrichten oder Ähnliches gebraucht wird, in Gnublin gibt es jetzt eine Sammlung von Programmen, mit denen man das alles auf gleiche Art und Weise machen kann: Bei uns heißen sie Gnublin-Tools.

Das Prinzip für ein Tool sieht wie folgt aus:

Für Anfänger:

- Modul anstecken
- Befehl eingeben (ideal kein Parameter)
- Wert oder Ergebnis erhalten

Für Forgeschrittene:

- Modul anstecken
- Befehl mit Parameter angeben + Ausgabeformat bestimmen
- Abfangen der Ergebnisse mit einem eigenen Programm

Für Gurus:

- Modul anstecken + Interrupt-Handler und CS-Leitungen einstellen
- Befehl mit Parameter angeben + Ausgabeformat bestimmen
- Eigene Gnublin-Tools entwickeln

Das wichtige dabei ist, dass alle Programme gleich reagieren bzw. einheitliche Bedienung und Ausgaben garantieren. Dann kann man die Module perfekt in eigene Anwendungen integrieren.

Einschalten der LED auf dem Board:

```
gnublin-gpio -p 3 -o 1
```

Abfragen AD-Wandler:

```
gnublin-adcint -c 0
```

Minimale Ausgabe erzwingen

Tool	Beschreibung
gnublin-adcint	Internen ADC (max 4 Kanäle) abfragen
gnublin-adcmod	Externes ADC Modul abfragen
gnublin-cam	Webcam ansteuern (Snapshot oder Stream)
gnublin-dogm	Display Ansteuerung (2x16 Zeichen)
gnublin-lm75	Abfrage Temperatursensoren
gnublin-pwm	Erzeugen eines PWM-Signals
gnublin-rtc	Steuerung der Echtzeituhr (Uhrzeit setzen, auslesen oder übernehmen)
gnublin-step	Schrittmotor Ansteuerung
gnublin-wlan	Einrichten einer WLAN-Verbingung

Tabelle: GNUBLIN Tools

Startet man ein Gnublin-Tool standardmäßig, werden zum Teil diverse Meldungen oder Hinweise ausgegeben, um einen einfachen Einstieg in die Programme und Module zu gewährleisten. Möchte man aber solch ein Tool in eigenes Programm integrieren, ist es besser wenn man nur minimale Ausgaben erhält bzw. dann gleich auf das JSON Format, das im nächsten Abschnitt beschrieben ist, wechselt. Ist das JSON Format aber aufwändig, kann man einfach mit der Option -b (Abkürzung für bare) eine minimale Ausgabe erzwingen.

JSON für Datenaustausch

JSON [4] ist ein kompaktes Datenformat, das für Mensch und Maschine einfach lesbarer ist und daher zum Datenaustausch zwischen Anwendungen perfekt geeignet ist.

Eine JSON Ausgabe vom Temperatursensor sieht wie folg aus:

```
{ result‘:‘0‘, ‚temperature‘:‘21‘, ‚error _
msg‘:‘‘ }
```

Die Ausgabe erhält man, indem man einfach die Option -j an jeden „gnublin-Befehl“ anfügt. Schon wird jede Ausgabe per JSON gelifert.

Temperatur:

```
gnublin-lm75 -j
```

Das JSON Format kann man sehr einfach in interne Datenstrukturen verschiedenster Programmiersprachen überführen. Viele Programmiersprachen bieten mittlerweile fertige JSON Encode und – Decode Funktionen an.

PHP JSON

In PHP gibt es die Funktionen json_decode und json_encode. Um das JSON Format in ein assoziatives Array zu überführen verwendet man encode [http://www.php.net]:

```
string json _ encode ( mixed $value [, int
$options = 0 ]
```

Diese gibt eine Zeichenkette zurück, die die JSON-Darstellung von value beinhaltet.

value Der zu kodierende value. Kann von jedem Typ außer Ressource sein. Diese Funktion arbeitet nur mit UTF-8-kodierten Daten.

options Bitmaske bestehend aus JSON_HEX_QUOT, JSON_HEX_TAG, JSON_HEX_AMP, JSON_HEX_APOS und JSON_FORCE_OBJECT. Standardmäßig auf 0 gesetzt.

Möchte man weitere Daten in das JSON Format überführen, um Daten zwischen Programmen austauschen zu können, kann man ebenfalls einfach aus einem assoziativen Array ein JSON Format erzeugen:

```
mixed json _ decode ( string $json [, bool
$assoc = false [, int $depth = 512 ]] )
```

Dies konvertiert eine JSON-kodierte Zeichenkette in eine PHP-Variable.

json Der zu dekodierende json-String.

assoc Wenn TRUE, werden zurückgegebene Objekte in assoziative Arrays konvertiert. depth Benutzerspezifische Verschachtelungstiefe.

Python JSON

Ab der Version 2.6 gibt es JSON Funktionen „On-Board“.

```
import json

data = [ { ‚result‘:‘0‘, ‚voltage‘:‘2500‘, ‚error _ msg‘:‘‘ } ]

print ‚DATA:‘, repr(data)
```

Webanwendung als einfaches Demo

Zu guter Letzt ist das GNUBLIN Projekt um eine kleine PHP-Anwedung inkl. moderner JqueryUI [5] Oberfläche erweitert worden. Den Download findet man im Wiki unter [6].

Im nächsten Journal wird diese ausführlicher beschrieben.

Abb. 8: GNUBLIN Web Demo

Literatur

[1] http://wiki.gnublin.org

[2] http://wiki.gnublin.org/index.php/Gnublin_Installer

[3] http://wiki.gnublin.org/index.php/Gnublin_Distribution

[4] http://de.wikipedia.org/wiki/JavaScript_Object_Notation

[5] http://jqueryui.com/

[6] http://wiki.gnublin.org/index.php/Webanwendung

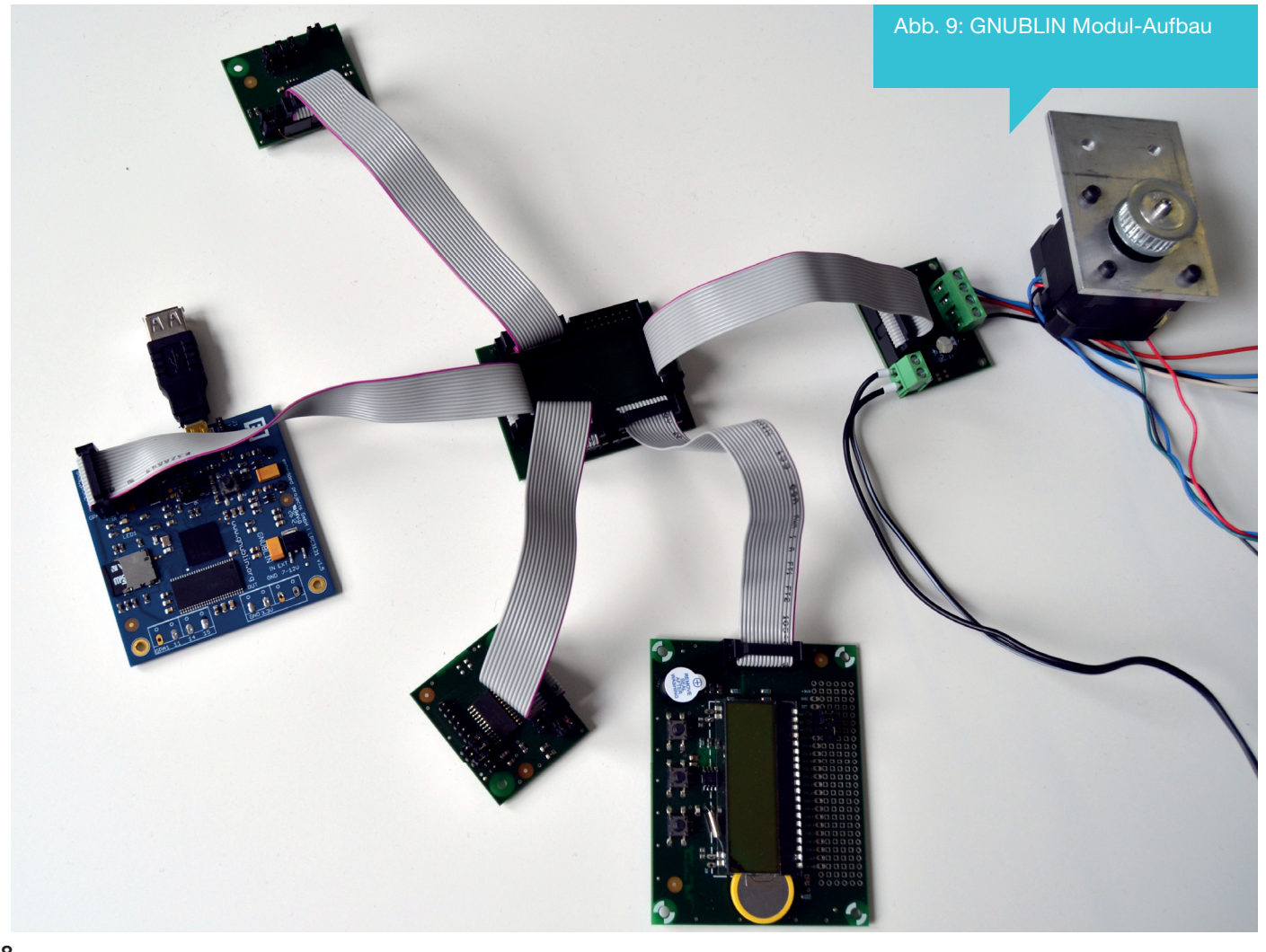
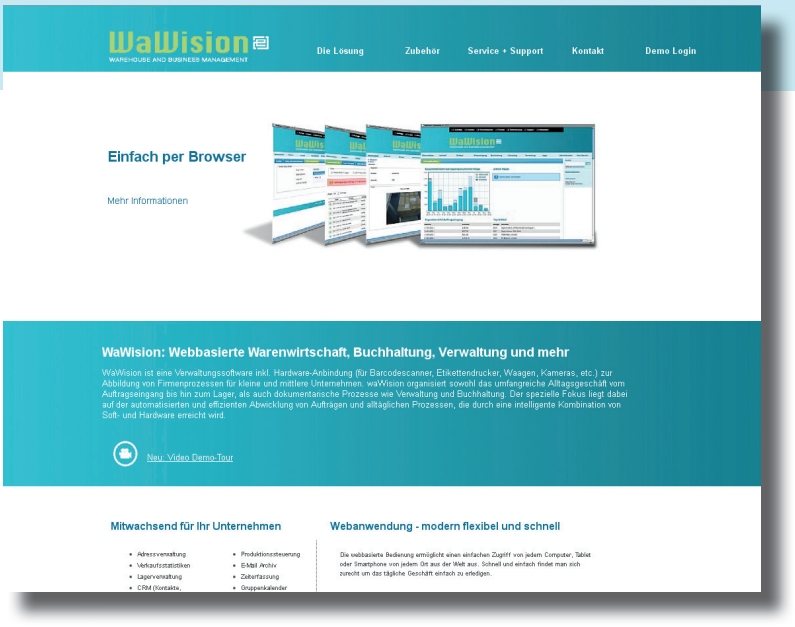


Abb. 9: GNUBLIN Modul-Aufbau

WaWision

Stiller Start eines Open-Source Warenwirtschaftssystems



Entstanden aus dem eigenen Bedarf - so startete die Idee für WaWision im Jahr 2008 nach Gründung unseres Unternehmens embedded projects GmbH. Auf der Suche nach einem passenden und bezahlbaren Warenwirschaftssystem und mit Anbindung an wOnline-Shops und Verwaltungstools entstand nach einer Marktanalyse immer mehr die Idee ein eigenes System zu entwickeln.

Das Ergebnis nach nun fast 5 Jahren ist das Warenwirtschaftssystem WaWision, das aktuell primär für Ingenieurbüros und kleine Produktionsstätten mit einem optionalen Online-Shop bzw. Lager-Versand ist. Alle klassischen Aufgaben wie Lager- und Stücklistenverwaltung, Versandwesen- und Online-Shop-Anbindung, Angebots-, Rechnungs- und Lieferscheinerstellung sind integriert. Alles was bei uns in der Firma benötigt ist, haben wir im System abgebildet.

Durch den offenen Quelltext kann das System flexibel an die eigenen Bedürfnisse angepasst werden. WaWision kann komplett über den Browser bedient werden. Für die Anbindung von Barcode-Lesegeräten und Etikettendruckern, elektronischen Waagen sowie Kameras im Warenein- und -ausgang haben wir die sogenannte Adapter-Box entwickelt. Die externen Geräte werden einfach an diese angesteckt und vom System automatisch erkannt.

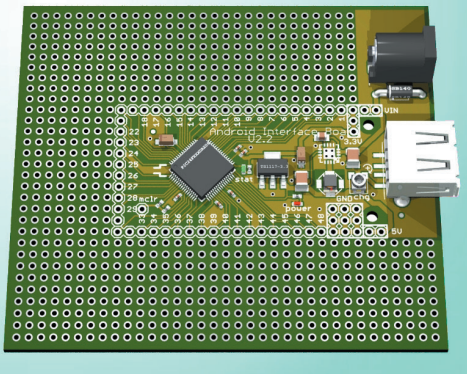
Zur Open-Source Version gibt es eine sogenannte Enterprise Version. Diese beinhaltet für Firmen den notwendigen Support und entsprechende Extraleistungen.

Wir freuen uns auf die nächsten 5 Jahre WaWision.

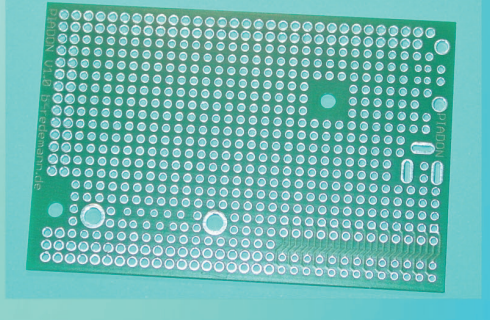
[1]http://www.wawision.org □

Anzeige

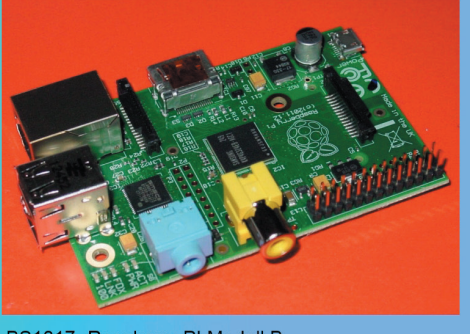
Elektronikbausätze, Bücher, Komplettsets



BS1013 - Bausatz Android Interface Board (IOIO-Clone) 37,00 €*



LP1001 - PIADON - Lochrasterboard für das Raspberry Pi 8,00 €



BS1017- Raspberry PI Modell B 42,00 €

Weitere Bausätze im Shop zu finden:
06104 USB-Modul mit FT232: 18,00 €*
06103 USB-Modul mit FT2232: 28,00 €*
BS1008 Ethernetmodul mit ENC28J60: 15,00 €*
07102 Programmer usbasp: 8,00 €*

0B103 Schrittmotormodul mit TMC222: 20,00 €*
BS1015 Schrittmotormodul mit L293: 8,00 €*
BS1016 Schrittmotormodul mit L298: 12,00 €*

BS1006 Arduino Clone: 15,00 €*
BL6101 USB-Buch: 39,00 €*
BL9110 AVR-Buch: 39,00 €*

Anbieter(kein Laden):
B. Redemann
Mahlower Str.204
14513 Teltow

Hier im Shop: www.b-redemann.de

Bildverarbeitung mit eingebetteten Systemen

Ulrich Hornung, Matthias Pohl, Michael Schäferling, Michael Seider und Gundolf Kiefer

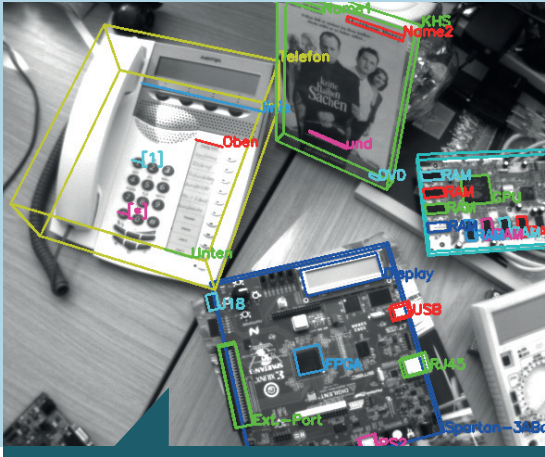


Abb. 1: MARS in Aktion - Verschiedene Gegenstände aus dem Labor werden automatisch vom System erkannt und passend beschriftet (vorgestellt auf der embedded world 2012 und Go-3D 2012)

Einleitung

Bildverarbeitung ist im Allgemeinen mit hoher Rechenleistung verbunden. Das fängt bei einfachen Algorithmen, wie der Anpassung von Helligkeit und Kontrast oder dem Filtern von Bildrauschen an und geht weiter über Kantendetektoren, Bildent- und -verzerrungen und Detektoren für Merkmale im Bild, die zum Erkennen und Tracken von Objekten benötigt werden. Ein Ende ist hier nicht in Sicht. Immer komplexere Algorithmen werden für immer bessere automatische Analyseverfahren benötigt.

In dieses sehr interessante Forschungsthema kann man sogar selbst recht einfach einsteigen. Viele Algorithmen sind in Open-Source-Bibliotheken, wie zum Beispiel OpenCV [5], in C/C++ implementiert und können benutzt werden, ohne sich großes Expertenwissen vorher aneignen zu müssen.

Die Arbeitsgruppe „Effiziente eingebettete Systeme“ an der Hochschule Augsburg befasst sich damit, wie man genau solche Algorithmen in eingebetteten System betreiben kann, und dabei die Schwächen dieser Systeme umgeht, um eine hohe Performance zu erreichen.

Dabei wird, neben Spezial-Hardware in FPGAs, auch stark von Open-Source-Software Gebrauch gemacht. Neben der OpenCV-Bibliohek wird auch Linux eingesetzt, das sich als Betriebssystem der Embedded-Plattformen sehr gut eignet. Die folgenden Projekte sind in den letzten zwei Jahren gewachsen und werden auch aktuell noch weiterentwickelt.

MARS - Ein mobiles Augmented-Reality-System

Als Teil des BMBF-Projektes „Triokulus“ wurde ein mobiles Augmented-Reality-System (MARS) entwickelt, welches reale Objekte erkennt und deren Lage im Raum bestimmt. Anschließend wird die berechnete Position genutzt um zusätzliche Informationen (siehe Abb. 1) über das erkannte Objekt einzublenden.

Als Basis für dieses System dient das Pandaboard (siehe http://pandaboard.org), dessen Hardware weitgehend der eines aktuellen Smartphones entspricht und mit Batterien betrieben wird. Hauptbestandteil ist ein OMAP4430-System-on-Chip mit zwei Cortex-A9-Kernen, die mit 1 GHz getaktet werden. Darauf läuft Ubuntu-Linux und die MARS-Software.

Die Software empfängt die Bilder von einem Kamerasystem mit zwei intelligenten FPGA- Kameras, die bereits einen wesentlichen Teil der Bildverarbeitung - die Detektion markanter Punkte - in Hardware erledigt. Die in den Kameras gewonnenen Informationen werden zusammen mit den Bilddaten an das Hauptsystem übertragen und dort zur Objekterkennung und Posenbestimmung verwendet.

Den wichtigsten Teil des Systems stellt der SURF-Algorithmus dar (SURF = „Speeded Up Robust Features“, siehe [1]), und wird verwendet um markante Punkte aus dem Bild zu extrahieren. Diese Punkte können anschließend mit gespeicherten Punkten aus einer Datenbank verglichen werden, um bekannte Objekte auf dem Bild wiederzuerkennen. Anschließend wird die exakte Lage dieser Punkte genutzt, um die 3D-Position und Lage des Objekts genau zu berechnen. SURF ist auch in OpenCV implementiert und es existieren zahlreiche quelloffene Software- und GPU-Implementierungen. Er kann somit recht schnell auch ohne Spezial-Hardware genutzt werden.

Das MARS-System kann beispielsweise für Schulungszwecke genutzt werden, um dem Benutzer mit Hilfe von Augmented-Reality die Funktionen eines neuen Gerätes beizubringen.

Ein weiteres interessantes Einsatzgebiet ist ein autonomes Robotersystem, welches bestimmte Objekte mit Hilfe von Kameras erkennt und anschließend mit ihnen interagieren kann (siehe Abb. 3).

NITRA - Ein Hardware-Modul zur Korrektur von Linsenverzerrung und Rektifizierung von Kamerabildern in Echtzeit

NITRA („Non-linear Image Transformation“) ist ein Hardware-Modul, welches die Entzerrung und Rektifizierung von Kamerabildern „on-the-fly“ ermöglicht. Wie beispielsweise im linken Bild von Abbildung 2 zu sehen ist, sind Weitwinkelaufnahmen meist mit starken Bildverzerrungen verbunden. Um das entzerrte Bild zu erstellen tastet das NITRA-Modul das Originalbild nicht-linear mit Hilfe von Polynomen ab. Dies erlaubt eine sehr flexible Transformation der Bilddaten. Durch die Realisierung in einem FPGA-Chip und die direkte Anbindung an einen Bildsensor in der intelligenten Kamera werden zudem sehr geringe Verzögerungen bei gleich bleibender Framerate erreicht. Wer keinen FPGA zur Hand hat, hat die Möglichkeit das alles auch mit OpenCV umzusetzen. Diese Software-Bibliohek bietet nicht nur die

Algorithmen zur Entzerrung, sondern auch alles Nötige drumherum:

- Eine einfache Möglichkeit um gewöhnliche USB-Webcams anzusteuern.
- Kamerakalibrierungs-Funktionen, für eine und zwei Kameras.

Nach dem Entzerren des Bildes können anschließend weitere Algorithmen aus OpenCV darauf angewendet werden. Die Berechnung von Tiefenkarten ist ein interessantes Beispiel. Diese können mit zwei kalibrierten Kameras berechnet werden und geben dann Auskunft über die Entfernung von Objekten in der Umgebung. Allerdings erlaubt eine reine Softwarelösung gerade im Embedded-Bereich oft nur eine schwache Performanz im Vergleich zu einer Implementierung im FPGA.

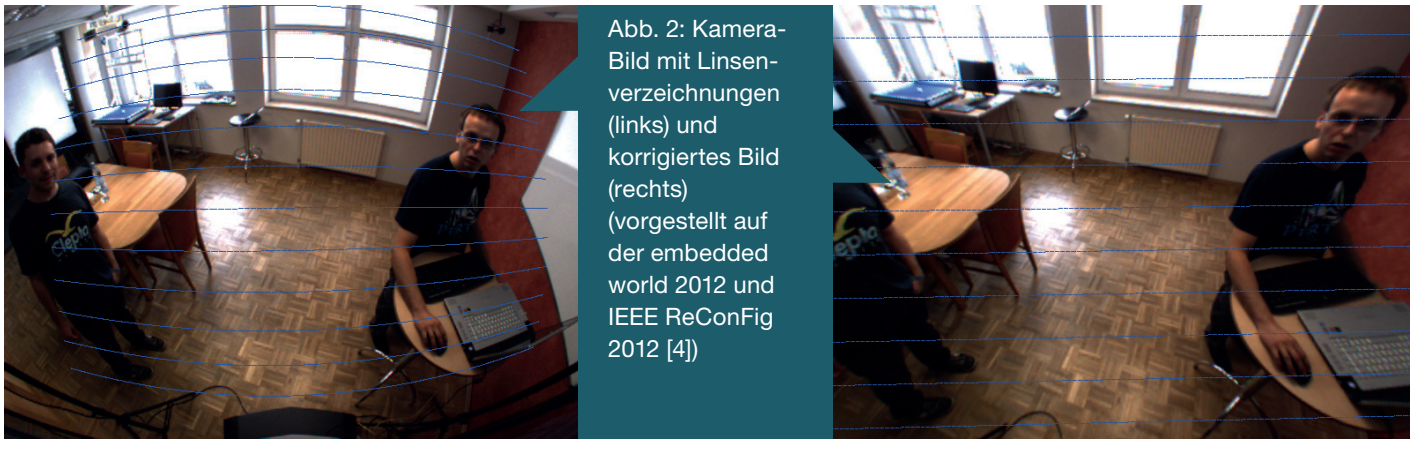
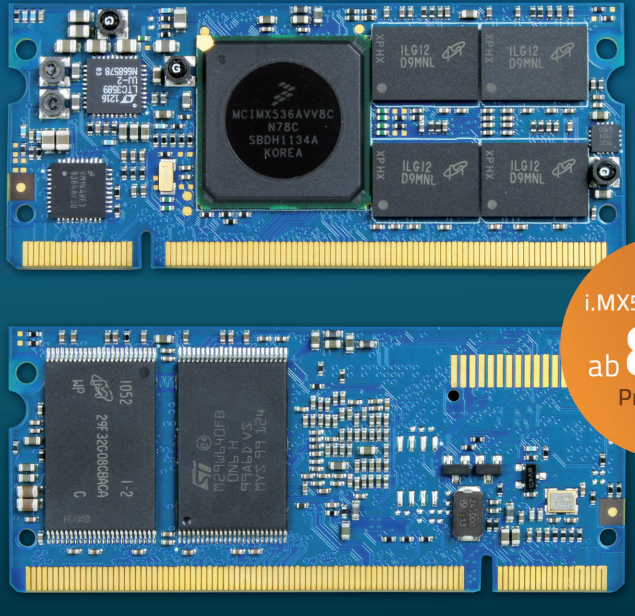


Abb. 2: Kamera-Bild mit Linsenverzeichnungen (links) und korrigiertes Bild (rechts) (vorgestellt auf der embedded world 2012 und IEEE ReConFig 2012 [4])

Anzeige

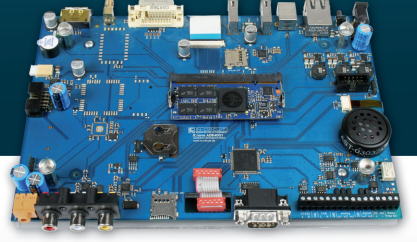
ICnova i.MX536 SODIMM
High End Cortex-A8 SODIMM-200 Modul



ICnova i.MX536 SODIMM ab 85€ Preis exkl. MwSt.

WWW.IC-BOARD.DE
WEBSHOP IN-CIRCUIT GMBH

FEATURES
- Freescale i.MX536 Cortex-A8 Prozessor mit bis zu 800 MHz (1GHz in iMX535 Version)
- 512 MByte DDR3-RAM
- 8 MByte paralleler NOR-Flash für schnelles Booten
- 4 GByte NAND Flash
- integrierter 10/100 MBit Ethernet PHY
- Schnittstellen: Ethernet, USB Host+Device, SATA, UARTs, SPIs, CAN, ISO7816, LCD (TTL and LVDS), Camera IF...
- die meisten Pins sind auch als GPIO verwendbar
- Alle Spannungsregler integriert, Eingangsspannung 5V + -10%, Leistungsaufnahme typ. 1,5-2W
- SODIMM-200 Formfaktor (2.5V-Version-Sockel)
- Temperaturbereich -20°C bis +70°C
- zugelassen für Industrie und Wohnbereich
- passendes Entwicklungsboard ADB4001



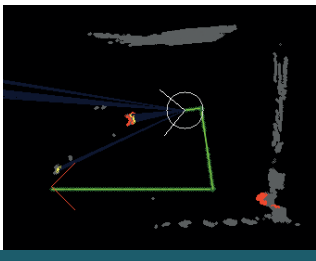
IN-CIRCUIT
ENGINEERING AS A PASSION

In-Circuit GmbH · Königsbrücker Str. 69 · D-01099 Dresden www.in-circuit.de

© MEDIENKOLLEKTIV.DE

Spotty

Die beiden vorab vorgestellten Projekte wurden in einem studentischen Projekt an der Hochschule Augsburg namens „Spotty“ genutzt um einen Staubsaugerroboter iRobot Roomba 531 so zu programmieren, dass er autonom eine unbekannte Umgebung kartografiert um anschließend eine intelligente Route zum Abfahren des Raumes berechnen zu können (siehe Abb. 3). Dieses Projekt setzt ebenfalls auf das Pandaboard und Linux in Kombination mit zwei FPGA-Kameras.



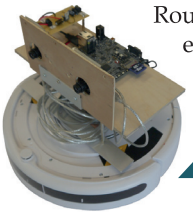


Abb. 3: Staubsaugerrobottor „Spotty“ (links) kartografiert selbständig die Umgebung. Das Bild oben zeigt eine Karte die der Roboter selbständig von der Umgebung erstellt hat (Grau sind Hindernisse, grün der bisher gefahrene Weg, rot sind bekannte Objekte die im Raum entdeckt wurden). Bildquelle: Projektgruppe Spotty - Christine Demharter, Klaus Göttling, Alexander Senger, Thomas Spanrunft

Open Source in der Lehre

Neben den genannten Forschungs- und Entwicklungsarbeiten werden im Labor verschiedene Lernplattformen für und zum Teil von Studierende(n) entwickelt. Der „Hands-On Audio Player“ ist ein FPGA-basiertes System, mit dem Studierende sowohl die Treiberprogrammierung unter Linux als auch die FPGA-Programmierung und den Entwurf digitaler Schaltungen trainieren können (siehe Abb. 4). Der HiCoVec-Prozessor ist ein flexibel konfigurierbarer Vektorprozessor, der als Open-Source-Projekt an der Hochschule Augsburg im Rahmen von studentischen Arbeiten entstanden ist (siehe Abb. 5).

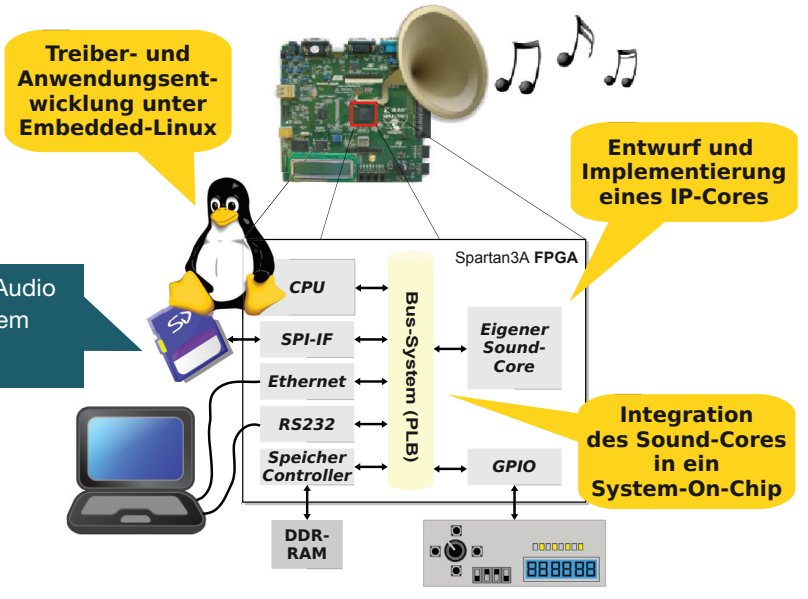


Abb. 4: Der „Hands-On Audio Player“, realisiert auf einem „Spartan 3A Starter Kit“

Abb. 5: Der HiCoVec-Prozessor in Aktion. Die Hardware und Software zu diesem Autorennspiel wurde komplett von Studenten entwickelt und ist in einem kleinen FPGA (Spartan 3A-700) realisiert

Literatur

[1] Herbert Bay, Andreas Ess, Tinne Tuytelaars und Luc Van Gool: „Speeded-Up Robust Features (SURF)“, Computer Vision and Image Understanding (CVIU), 2008, Ausgabe 110, S. 346-359.

[2] Michael Schäferling, Ulrich Hornung und Gundolf Kiefer: „Object Recognition and Pose Estimation on Embedded Hardware: SURF-Based System Designs Accelerated by FPGA Logic“, International Journal of Reconfigurable Computing, Ausgabe 2012, Artikel-ID: 368351

[3] Ulrich Hornung, Michael Schäferling, Gundolf Kiefer und Andreas Becher: „Markerlose optische Objekt- und Posenbestimmung mit eingebetteter Hardware“, Go-3D - Computergraphik für die Praxis, 2012, S. 37-49

[4] Matthias Pohl, Michael Schäferling, Gundolf Kiefer, Plamen Petrow, Egmont Woitzel und Frank Papenfuß: „An Efficient and Scalable Architecture for Real-Time Distortion Removal and Rectification of Live Camera Images“, IEEE International Conference on ReConFigurable Computing and FPGAs, 2012

[5] OpenCV-Homepage: http://opencv.willowgarage.com/

Kontakt und weitere Informationen:

Homepage der Arbeitsgruppe:
http://ees.informatik.hs-augsburg.de

E-Mail:
gundolf.kiefer@hs-augsburg.de

Ein Prototyping-Board für den Raspberry Pi

Martin Schmidt, Michel Schmidt – Martins Bastelstube

Stefan Enderle – qfix robotics

Einführung

Seit Frühjahr 2012 ist der Einplatinencomputer „Raspberry Pi“ erhältlich – ein kompletter, unter Linux lauffähiger PC für unter 50 € auf der Fläche einer Kreditkarte. Die Platine enthält einen mit 700MHz getakteten ARM 11 samt 256 oder 512MB SDRAM, 2 × USB, Ethernet, SD-Kartenleser und mehr. Die Videoausgabe erfolgt in hoher Auflösung über die eingebaute HDMI-Buchse oder in niedriger Auflösung als Composite-Video-Signal über eine Cynch-Buchse. Und für Bastler stehen bis zu 8 GPIOs, SPI, I2C und UART zu Verfügung. Das Ganze läuft mit lediglich 5V bei max. 700mA, versorgt über die Micro-USB-Buchse oder über die GPIO-Steckerleiste.

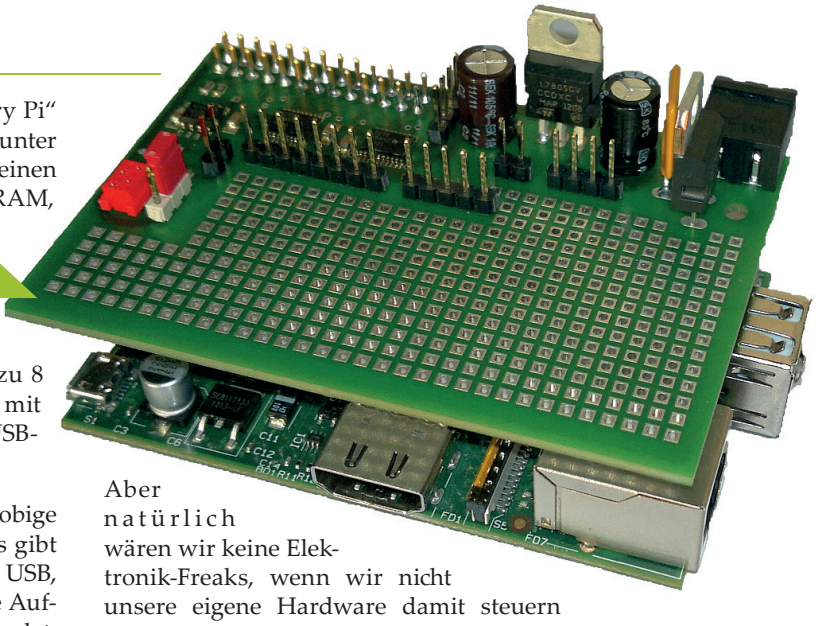


Abb. 1: Prototyping-Board

Der Vollständigkeit halber muss erwähnt werden: Die obige Beschreibung gilt für das „Model B“ des Raspberry Pi. Es gibt auch ein „Model A“ mit einfacherer Ausstattung (nur 1× USB, kein Ethernet, nur 256MB SDRAM). Wir haben für unsere Aufbauten und Tests bisher ausschließlich das Model B verwendet, wobei das Model A prinzipiell genauso verwendbar wäre. Die Inbetriebnahme des Systems gestaltet sich als recht einfach. Tastatur und Maus können über die USB-Ports angeschlossen werden, ein Fernseher an den HDMI-Anschluss. Das komplette Linux-Dateisystem wird auf einer SD-Karte gespeichert. Diese kann man sich fertig installiert besorgen oder selbst mit frei im Internet verfügbaren Images beschreiben. Und schon kann es losgehen.

Aber natürlich wären wir keine Elektronik-Freaks, wenn wir nicht unsere eigene Hardware damit steuern wollten. Also ran an die GPIOs – aber Achtung: 3,3V! Um mit unserer bisherigen Elektronik kompatibel zu sein oder per I2C zu kommunizieren, wären 5V geschickter. Das war der Punkt, an dem wir uns entschieden, eine Aufsteckplatine für den Raspberry Pi zu bauen, die ein Experimentierfeld enthält und die GPIOs samt I2C-Bus mit 5V-Pegel zur Verfügung stellt. Eine Spannungsversorgung für das Board und den Pi sollte auch gleich mit drauf .

Raspberry Pi Hardware

Im Detail beinhaltet der Raspberry Pi Model B folgende Hardware:

- CPU: ARM1176JZF-S (700 MHz)
- SoC: Broadcom BCM2835
- GPU: Broadcom VideoCore IV
- SDRAM: 512MB (vor Oktober 2012 nur 256MB)
- Ext.Memory: SD/MMC/SDIO-Kartenleser
- Video: Composite, HDMI
- Audio: 3,5 mm-Klinke, HDMI
- LAN: 10/100 MBit Ethernet-Controller (SMSC LAN9512)
- IO: 8 GPIO-Pins, SPI, I²C, UART

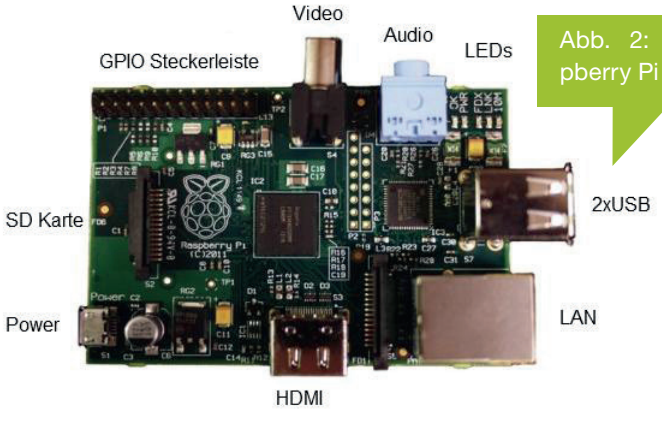


Abb. 2: Raspberry Pi

Funktion	Pin		Pin	Funktion
3,3V	1		2	5V
GPIO 0 (SDA)	3		4	
GPIO 1 (SCL)	5		6	GND
GPIO 4 (GPCLK0)	7		8	GPIO 14 (TxD)
	9		10	GPIO 15 (RxD)
GPIO 17	11		12	GPIO 18 (PCM_CLK)
GPIO 21 (PCM_DOUT)	13		14	
GPIO 22	15		16	GPIO 23
	17		18	GPIO 24
GPIO 10 (MOSI)	19		20	
GPIO 9 (MISO)	21		22	GPIO 25
GPIO 11 (SCLK)	23		24	GPIO 8 (CE0)
	25		26	GPIO 7 (CE1)

Tabelle: Pinbelegung der GPIO Steckerleiste

Prototyping Board

Die Idee des vorgestellten Prototyping-Boards ist es, drei maßgebliche Funktionen bereitzustellen:

1) Spannungsversorgung von Raspberry Pi und Prototyping Boards
2) Einfachen Anschluss der GPIOs und des I2C durch Pegelwandler
3) Experimentierfeld

Hier ein Bild des Prototyping-Boards (Version 1) (siehe Abb. 3).

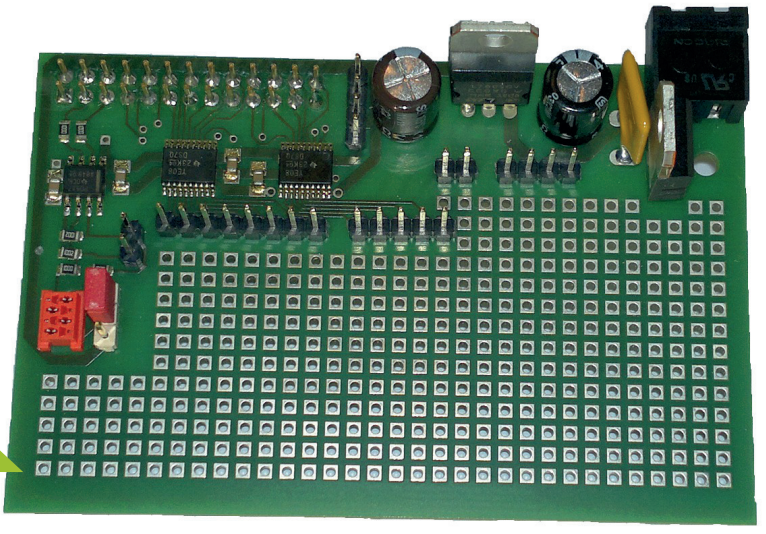


Abb. 3: Prototyping-Boards (Version 1)

1. Spannungsversorgung

Im rechten oberen Teil des Bildes erkennt man die Spannungsversorgung samt Verpolungsschutzdiode, Sicherung und 5V-Festspannungsregler. Hierdurch kann das Board mit 7V bis 15V Eingangsspannung betrieben werden. Statt des Festspannungsreglers kann natürlich auch ein integrierter DC/DC-Wandler bestückt werden, um weniger Verlustleistung und damit Wärme zu produzieren.

Der Raspberry Pi wird über den GPIO-Stecker mit versorgt, so dass man für ihn kein separates Netzteil braucht. Das ist sehr praktisch bei mobilen, batteriegespeisten Anwendungen (Roboter etc.)

2. Pegelwandler

Der TXB0108 bietet eine relativ einfache Möglichkeit, acht GPIO-Leitungen bidirektional z.B. von 3,3V auf 5V Pegel zu heben. Allgemein gesprochen können zwei unterschiedliche Spannungen als jeweilige Versorgungsspannung angelegt und die IO-Pins dann auf dem entsprechenden Pegel benutzt werden. Der TXB0108 erkennt von selbst, in welcher Richtung die Daten fließen, und schaltet sich entsprechend um.

Für den I2C-Bus existiert das gleiche Prinzip mit dem Baustein PCA9517.

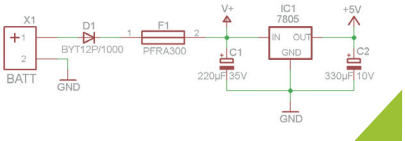


Abb. 4: Spannungsversorgung

3. Experimentierfeld

Der restliche Platz des Boards wurde als Experimentierfeld im 2.54mm Raster ausgelegt. Hier können relativ einfach eigene Ideen realisiert werden.

Fazit

Das vorgestellte Prototyping-Board für den Raspberry Pi ist in der Version 1 schon recht gut für eigene Experimente einsetzbar. Wir planen, in einem weiteren Artikel die ersten Projekte mit dem Prototyping-Board vorzustellen.

Weitere Information gibt es unter

bastelstube.rocci.net

und

www.qfix-robotics.de.

Bei qfix können auch leere oder SMD-bestückte PrototypingBoard-Platinen bezogen werden.

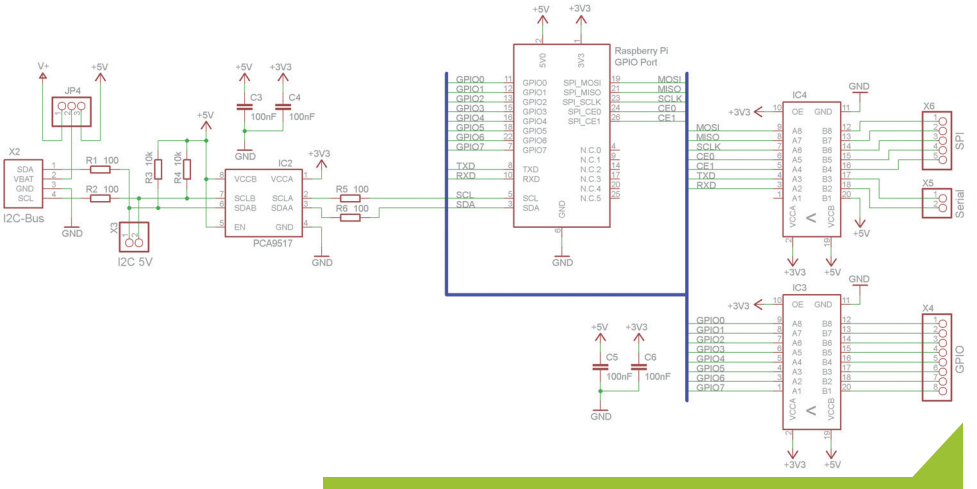


Abb. 5: GPIO-Stecker des Pi (Mitte) und Pegelwandler

Ein Übertragungsystem mit dem CC 2500 von TI

Jens Reineke, Thomas Grötzbach, Harald Görl

Einleitung

Die Basis für das hier vorgestellte Übertragungssystem ist der Transceiverchip CC2500 von Texas Instruments. Dieser Chip arbeitet im Frequenzband von 2400 MHz bis 2483,5 MHz. Er ist zu einem Preis von unter fünf Euro leicht beschaffbar und hat einen geringen Leistungsaufnahme von max. 3,3 V · 17 mA≈0,05 W im Empfangs/RX-Modus und max. 3,3 V·21,2 mA≈0,07 W im Sende/TX-Modus. Der Transceiverchip unterstützt mehrere Modulationsarten und besitzt eine einstellbare Datenrate von bis zu 500 kbps. Weitere Chip Funktionen sind:

- Paket orientiertes Senden/Empfangen
- CRC (Cyclic Redundancy Check)
- Frequency Hopping
- Digital RSSI (Received Signal Strength Indication) output
- Zwei separate 64 Byte große RX und TX FIFOs
- SPI-Interface

In Abbildung 1 ist das Blockdiagramm des CC2500 dargestellt. Hier sind sowohl der Sendestrang als auch der Empfangsstrang des Transceiverchips zu sehen. Die empfangenen Signale gelangen über die Anschlüsse RF_P/RF_N in den Chip, werden dann zuerst analog verarbeitet z.B. gefiltert und gelangen von dort in den ADC. Der ADC wandelt die analogen Signale in digitale Signale um, welche danach demoduliert werden. Im FEC forward error correction werden die nun digitalisierten Daten einer Fehlerprüfung unterzogen und gegebenenfalls korrigiert. Der Packet Handler entpackt die Nutzdaten aus dem Paket, welches neben diesen Daten auch Adressenfelder, Längenfelder und weitere Daten enthält, siehe Abbildung 1. Die Nutzdaten werden in den RXFIFO geschoben und können über das Digitale Interface ausgelesen werden.

Die zu sendenden Daten passieren die gemeinsam genutzten Funktionsgruppen in umgedrehter Reihenfolge wie die im Vorfeld beschriebenen empfangenen Daten. Danach werden die zu sendenden Daten in die gewünschte Form z.B. 2-FSK, GFSK, ASK/OOK oder MSK moduliert und anschließend mittels mehrerer analoger Baugruppen für das Senden vorbereitet. Die Steuerung und das Einstellen der Chipparameter des CC2500 erfolgt über das Beschreiben seiner internen 8-Bit-langen Register.

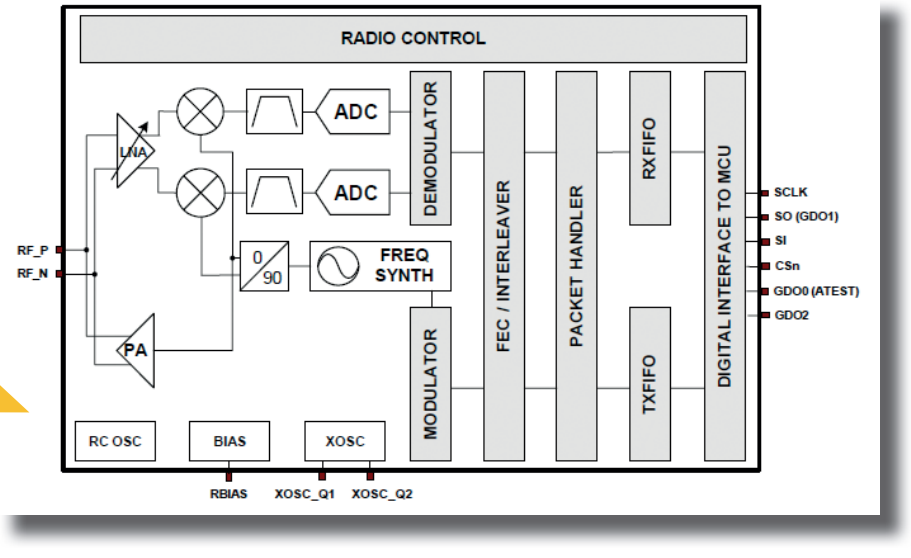


Abb. 1: Blockdiagramm CC2500 (Quelle: Datenblatt CC2500 von Texas Instruments)

Aufbau eines Datenpaketes

In Abbildung 2 ist der Aufbau eines per Funk übertragenen Paketes des CC2500 zu sehen. Die ersten Bytes des Paketes enthalten die Preamble. Diese geht dem Sync word voraus und enthält eine programmierte Schwelle, um eine zufällige Erfassung des Sync words im Rauschen zu verhindern. Das Sync word dient der Byte-Synchronisation. Es folgt das Length field, welches die Länge der Payload einschließlich des Längenbytes und der CRC-Checksumme angibt. Die nächsten Bits sind das Address field, welches die Zieladresse enthält. Im Data field sind die eigentlichen Nutzdaten enthalten, wobei im ersten Byte die Nutzdatenlänge in Bytes steht. Das letzte Paketfeld ist das CRC-16 Feld, in dem eine Checksumme enthalten ist, mit dessen Hilfe die Daten auf Fehler geprüft werden.

Die Preamble und das Sync word des Datenpakets werden vom CC2500 automatisch eingefügt. Es können lediglich die Anzahl der Preamble Bytes und die Länge des Sync word mit 16 oder 32 Bit bestimmt werden. Für das Length field kann eine feste Länge oder eine von der Payload abhängige variable Lände eingestellt werden. Das Address field und das CRC-16 Feld können optional zum Paket hinzugefügt werden. Hierfür ist in den entsprechenden Steuerregistern des Chips ein Aktivierungsbit zu setzen. Nach dieser Aktivierung fügt der CC25000 die entsprechenden Felder automatisch zum Paket hinzu.

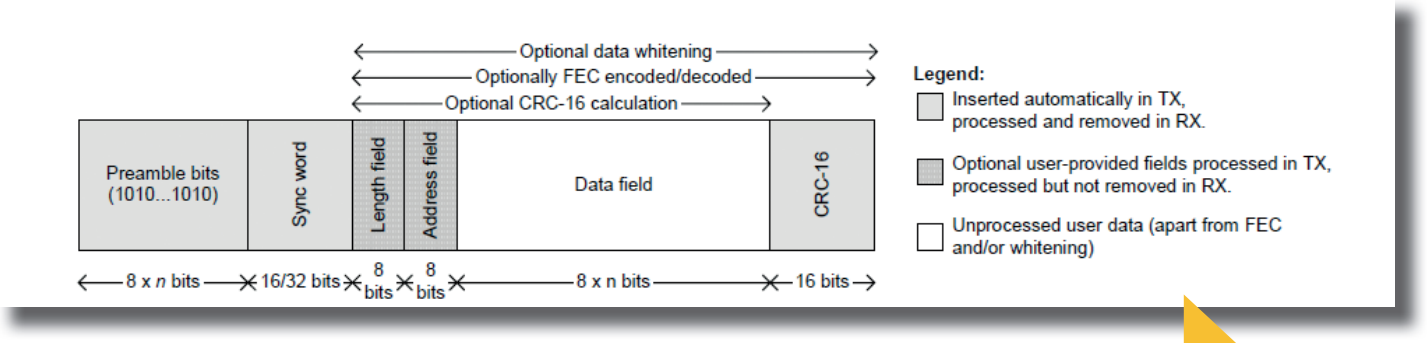
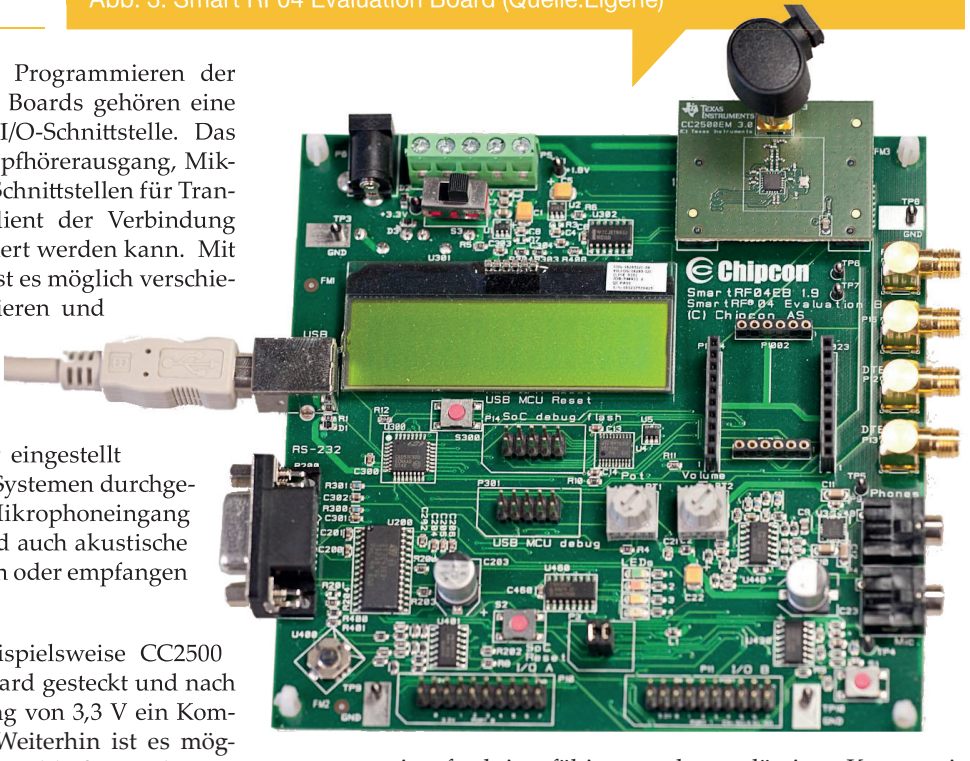


Abb. 2: Paketaufbau CC2500 (Quelle: Datenblatt CC2500 von Texas Instruments)

Smart RF04 Evaluation Board

Um die Entwicklung des Transceiversystems zu vereinfachen und die ersten Hürden mit dem Umgang des Transceiversystems zu meistern, ist das Entwicklungskit „Smart RF04 Evaluation Board“ erhältlich. Sowohl der CC2500, als auch die entsprechende Antenne werden auf Modulen auf das Entwicklungsboard aufgesteckt, so dass beispielsweise auch eigene Antennendesigns getestet werden können, wie im Bild 3 rechts oben zu sehen. Zudem können erste Übertragungstest mit Hilfe einer Windows-Software gestartet und die Parameter des Funkchips getestet werden.

Abb. 3: Smart RF04 Evaluation Board (Quelle:Eigene)



Eigenschaften

Diese Boards dienen dem Testen und Programmieren der CC2500 Serie. Zu den Bestandteilen des Boards gehören eine USB-, EIA-232-, USB-Debugging- und I/O-Schnittstelle. Das Board enthält auch ein Display, einen Kopfhörerausgang, Mikrophoneingang, einen Joystick und zwei Schnittstellen für Transceiverplatinen. Die USB-Schnittstelle dient der Verbindung mit einem PC, über den das Board gesteuert werden kann. Mit Hilfe der USB- und der I/O-Schnittstelle ist es möglich verschiedene Transceiversysteme zu programmieren und zu steuern. Über das Display und den Joystick ist eine vom PC unabhängige Steuerung des Evaluation Boards möglich. So können an einem angeschlossenen Transceiver verschiedene Parameter eingestellt und Funkübertragungen zwischen zwei Systemen durchgeführt werden. Kopfhörerausgang und Mikrophoneingang ermöglichen es mit dem Evaluation Board auch akustische Signale wandeln und per Funk übertragen oder empfangen zu können.

Mit dem Evaluation Board können beispielsweise CC2500 Transceiverplatinen in das Evaluation Board gesteckt und nach dem Anlegen einer Spannungsversorgung von 3,3 V ein Kommunikationstest durchgeführt werden. Weiterhin ist es möglich einen PC über USB an das Board anzuschließen und einen Funkbetrieb mittels der PC-Software Smart RF Studio aufzubauen. Mit Hilfe des Evaluation Boards und der Software Smart RF Studio werden Eigenentwicklungen eines CC2500 Transceiver Boards deutlich unterstützt. Beim Testen einer selbst entwickelten CC2500-basierten Schaltung stellt das Entwicklungskit einen garantiert funktionsfähigen und zuverlässigen Kommunikationspartner dar. Die für Tests notwendigen Registerwerte des CC2500 können mit Hilfe des Smart RF Studio gesetzt, angesehen und ausgewertet werden, um Rückschlüsse für die Registereinstellungen der Eigenentwicklung zu ziehen.

Funkbetrieb des Evaluation Boards über das Smart RF Studio

In diesem Kapitel wird die Inbetriebnahme des Evaluation Boards in Verbindung mit der Bedienung der Smart RF Studio Software erklärt.

Herunterladen und Installieren der Smart RF Studio Software von der Internetseite des Herstellers. http://www.ti.com/tool/smartrftm-studio

Aufstecken der CC2500 Transceiverplatine auf das Evaluation Board: Stiftleisten P1 und P2

Verbinden des PCs und des Smart RF04 Evaluation Boards mittels einer USB-Leitung

Starten der Smart RF Studio Software

Wechseln auf das Blatt „2.4 GHz“ (im Feld „List of connected devices“ ist das verbundene Evaluation Board aufgeführt)

Doppelklick auf den Button CC2500 (es öffnet sich das Device Control Panel, siehe Abbildung 4)

Bei einer Kommunikation zwischen zwei Evaluation Bords sind die Punkte 2 bis 6 mit dem gleichen oder einem zweiten PC zu wiederholen. Über die Bedienoberflächen „Packet TX“ und „Packet RX“ der Software können dann einfach Datenpakete ausgetauscht werden und so ein einfacher Test stattfinden. Die Bedienoberfläche „Continuous RX“ zeichnet einen Graph mit der Sendestärke in dBm über die Zeit auf.

Es hat sich bei den ersten Kommunikationsversuchen als sehr hilfreich erwiesen, in der Bedienoberfläche „Packet TX“ die Funktion „Add seq. Number“ zu deaktivieren. Dadurch wird die Sequenznummer nicht mitgesendet und somit scheidet eine falsche Sequenzangabe als Fehlerursache aus.

An dieser Stelle sei nochmals darauf hingewiesen, dass das Evaluation Board zwar eine Erleichterung beim Einstieg darstellt, da die Programmierung eines Mikrocontrollers mit angeschlossenem CC2500 Funkchip entfällt und ein Sende- bzw. Empfangsteil quasi sofort zur Verfügung steht. Trotzdem ist ein Einstieg auch völlig ohne ein solches Board möglich, Wissen über die Programmierung eines Mikrocontrollers vorausgesetzt.

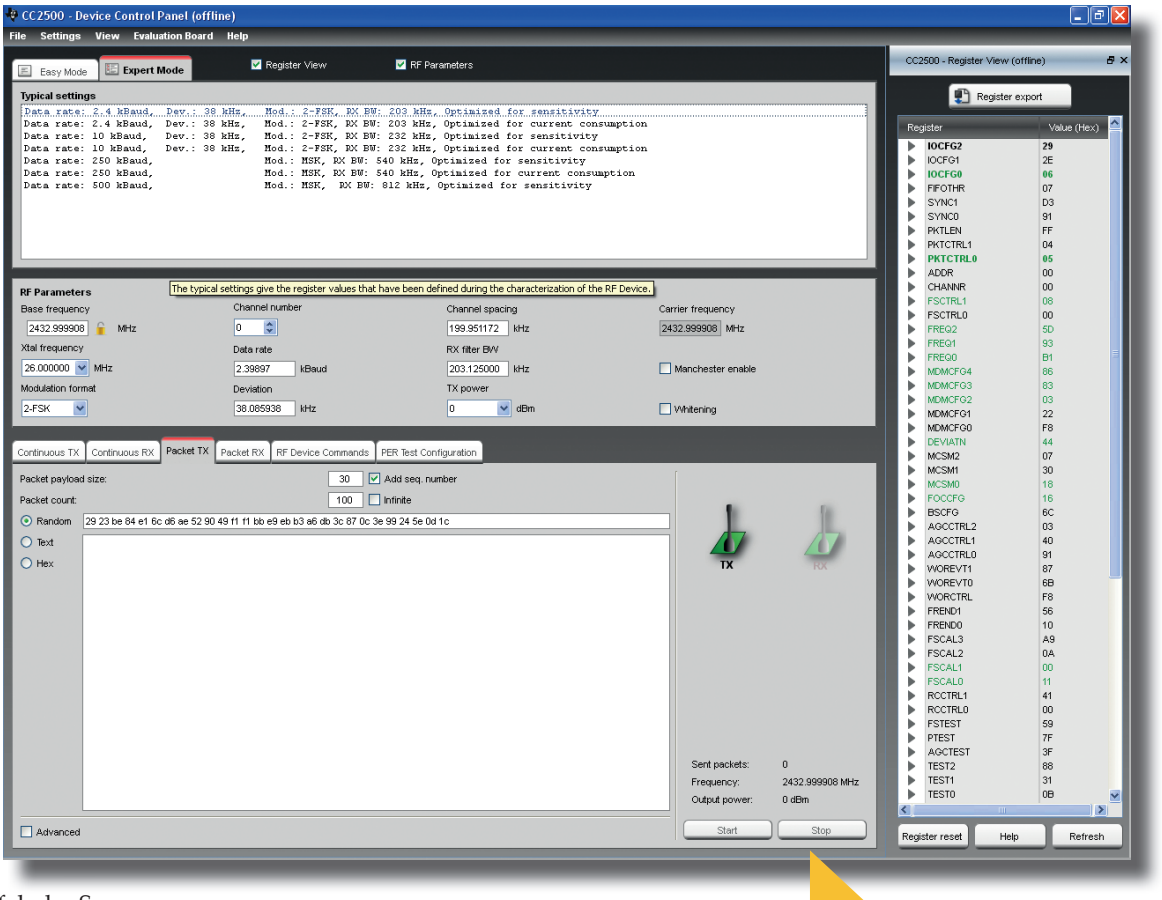


Abb. 4: Device Control Panel (Quelle:Eigene)

Hardware

Der CC2500 wird im QLP-Bauform geliefert und kann mit entsprechender Löterfahrung in eigene Designs verbaut werden. Neben dem Aufwand des Verlötens dieses Chipgehäuses muss zudem beachtet werden, dass vor allen die Antennenanschlüsse entsprechend der angegebenen Parameter und Designvorlagen beschaltet werden. Daneben sind Platinenmaterial, Substrat und die entsprechende Dielektrizitätskonstante sowie Platinenschichtendicke von großer Bedeutung. Bei den gezeigten Prototypen wurde FR4-Platinenmaterial mit 0,18 µm Schichtstärke eingesetzt. Bei anderem Material muss das Design entsprechend angepasst werden. Entsprechende Designregeln der Hochfrequenztechnik (hier 2,4 GHz) sind zu beachten, wie etwa der korrekte Umgang beim Design von Streifenleitungen usw.!

Anzeige

a. Fertiges CC2500-Modul

Wenn man diesen Aufwand umgehen will, so bieten sich fertige CC2500-Module an, die bei den typischen Online-Auktionshäusern für etwa 10 Dollar angeboten werden und meistens einen 12-poligen DIP-Anschluss besitzen. Derartige Module, wie auch in Abbildung 5 zu sehen, können dann direkt an einem Mikrocontroller über SPI verwendet werden. Dabei ist nur zu beachten, dass diese Module mit maximal 3,6 Volt betrieben werden dürfen und daher u.U. eine Pegelanpassung bei 5 Volt Controllern nötig ist.

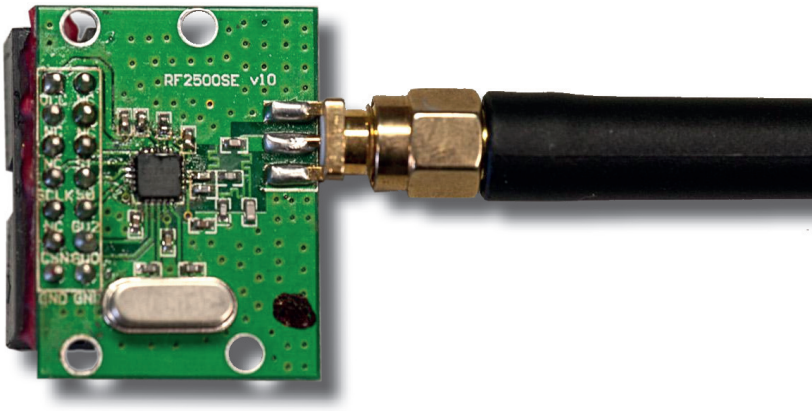


Abb. 5: Fertiges CC2500-Modul mit Stabantenne (Quelle:Eigene)

b. Realisierung mit Dipolantenne

Platzsparender ist jedoch der Verzicht auf externe Antennen und die Realisierung von Platinenantennen, die direkt auf der Transceiverplatine mit CC2500 Transceiver-Chip realisiert werden. Für die Lösung mit einer Dipolantenne erzeugt der Chip seinen Takt über einen externen 26 MHz Quarz mit den beiden Ziehkondensatoren. Als Indikator für eine vorhandene Versorgungsspannung ist zudem eine LED mit Vorwiderstand verbaut. Um die Baugröße zu minimieren wurde bei dieser Platine eine Dipolantenne direkt auf der Platine realisiert, wobei das Kupfer auf der Rückseite im Antennenbereich zwingend entfernt sein muss, wie in Abbildung 6 gezeigt ist. Die Abmessungen der Dipolantenne wurden der Design Note DN004 „Folded Dipole Antenna for CC25xx“ von Texas Instruments entnommen. Die Dipolantenne wurde so konstruiert, dass man sie direkt und ohne weitere Bauteile an den Transceiver-Chip anschließen kann.

Abb. 6: Platine mit Dipolantenne (Quelle: Eigene)

c. Inverted F-Antenne

Noch platzsparender sind sogenannte F-Antennen, die einen asymmetrisch gespeisten Monopol darstellen. Typisch für diese Bauform sind die Schleifenführung und der nahe dem Einspeisepunkt liegende Schluss gegen Masse.Alle Abmessungen der F-Antenne wurden der Application Note AN043 „Small Size 2.4 GHz PCB antenna“ von Texas Instruments entnommen. Sie besitzt eine Abmessung von nur 15,2 mm·5,7 mm und wurde so konstruiert, dass sie an eine 50 Ohm Quelle angeschlossen werden muss. Deshalb ist an den Antennenausgang des CC2500 ein Anpassungsnetzwerk wie in Abbildung 8 zu sehen mit Kondensatoren und Spulen anzubringen, welches diese Bedingung erfüllt. Auch hier gilt ein möglichst HF-taugliches Design umzusetzen.

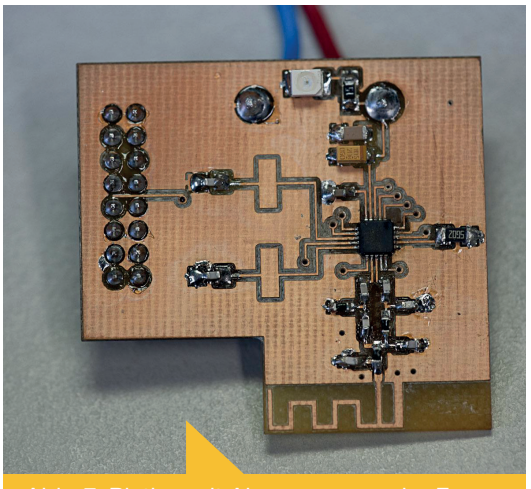
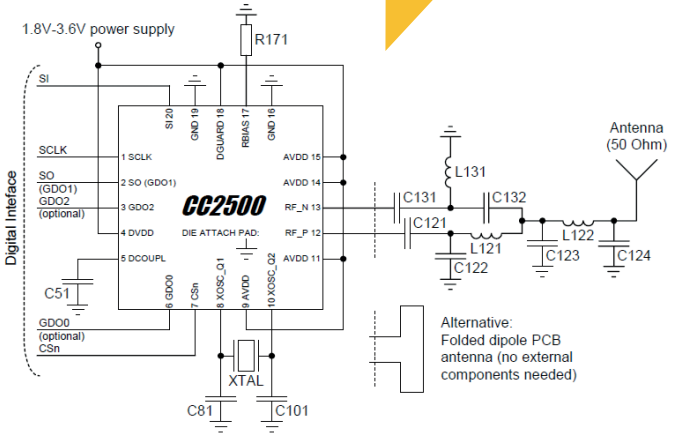


Abb. 7: Platine mit Abmessungen der F-Antenne (Quelle: Eigene)

Abb. 8: Beschaltung CC2500 (Quelle: Datenblatt CC2500 von Texas Instruments)



Beschreiben und Auslesen der CC2500 Register und Funktionsweise der Command Strobe Register

Hier wird davon ausgegangen, dass der Leser die typische Programmierung der SPI-Schnittstelle des verwendeten Controllers bereits kennt. Die Initialisierung und Verwendung der SPI-Controller wird hier nicht näher beschrieben. Die einzelnen Funktionen der Register des CC2500 sind im Manual von TI (PRELIMINARY Data Sheet (Rev.1.2), SWR-

S040a) ausführlich dokumentiert. In diesem Abschnitt werden daher nur die Besonderheiten der wichtigen Register für die Kommunikation mit dem Chip erklärt. Eingesetzt wurde dazu ein Controller vom Typ STM32F107, andere Controller nutzen ihre SPI Schnittstelle entsprechend.

Zum Auslesen der CC2500 Register ist zuerst das Registeradressbyte des auszulesenden Registers an den Chip zu senden. Das höchstwertige Bit dieses Registeradressbytes ist das read/write Bit, mit dem signalisiert wird, ob vom Register gelesen oder geschrieben wird. Das darauffolgende Bit ist das burst Bit. Dieses entscheidet, ob nur auf das entsprechende Register zugegriffen werden soll oder absteigend auch auf die darunter befindlichen Register. Danach folgen die sechs Register Adressbits. Nach dem Schreiben des Registeradressbytes muss zwingend das SPI-Register des Mikrocontrollers ausgelesen werden, da alte noch nicht gelesene Daten von neuen Daten nicht überschrieben werden. Abschließend ist ein 0x00 Byte zu versenden. Darauf werden die angeforderten Daten an den Mikrocontroller übersandt. Die Daten können dann aus dem SPI-Datenregister des Controllers gelesen werden.

```
/*Wait for SPIy Tx buffer empty*/
while (SPI _ I2S _ GetFlagStatus(SPI3, SPI _ I2S _ FLAG _ TXE) == RESET);
/*Send SPIz data;
Hier wird die Anfrage für das Lesen der Daten gesandt, reg ist die
Registeradresse*/
SPI _ I2S _ SendData(SPI3,reg);
/*Wait for SPIz data reception*/
while (SPI _ I2S _ GetFlagStatus(SPI3, SPI _ I2S _ FLAG _ RXNE)== RESET);/**/
/*Read SPIz received data*/
SPI _ I2S _ ReceiveData(SPI3);
/*Wait for SPIy Tx buffer empty*/
while (SPI _ I2S _ GetFlagStatus(SPI3, SPI _ I2S _ FLAG _ TXE) == RESET);
/*Send SPIz data*/
SPI _ I2S _ SendData(SPI3, 0x00);
/*Wait for SPIz data reception*/
while (SPI _ I2S _ GetFlagStatus(SPI3, SPI _ I2S _ FLAG _ RXNE)== RESET);/**/
/*Read SPIz received data;
Hier werden die Daten auf die gestellte Anfrage ausgelesen */
help = SPI _ I2S _ ReceiveData(SPI3);
```

Der Unterschied zwischen Auslesen und Beschreiben von CC2500 Registern besteht darin, dass das read/write Bit beim Beschreiben auf write gesetzt wird. Desweiteren enthält das zweite zu versendende Byte nach dem Registeradressbyte die Daten, die in das Register geschrieben werden sollen. Im Folgenden ist ein exemplarischer Code für die Nutzung des SPI-Interfaces zu sehen:

Neben den normalen Registern des CC2500 stehen die Command Strobe Register zur Verfügung. Diese Register dienen der Steuerung des Sende- und Empfangsablaufs im CC2500. Jedes der Command Strobe Register wirkt wie ein Befehl, der direkt ausgeführt wird, wenn das Registeradressbyte zum Transceiver-Chip versandt wird. Es ist kein zweites Datenbyte erforderlich. Die Register sind in Tabelle 1 aufgelistet.

Tabelle 1: Commend Strobe Register (Quelle: Datenblatt CC2500 von Texas Instruments)

Address offset: 0x00

Reset value: 0x0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BIDI MODE	BIDI OE	CRC EN	CRC NEXT	DFF	RX ONLY	SSM	SSI	LSB FIRST	SPE	BR [2:0]			MSTR	CPOL	CPHA
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Empfangen von Daten über die Funkschnittstelle

Um Daten mittels des CC2500 zu empfangen, gibt es sowohl einen Polling- als auch einen Interrupt-Betrieb. Für den Polling-Modus sind folgende Schritte durchzuführen:

1) Übersenden des command strobe SIDLE (geht in den IDLE Zustand)
2) Übersenden des command strobe SFRX (setzt den Receive-FIFO zurück)
3) Übersenden des command strobe SRX (Frequency synthesizer setup)
4) Übersenden des command strobe SRX (geht in den Receive-RX-Modus)

Wenn man sich anschließend im RX-Modus befindet, dann ist der CC2500 empfangsbereit.

Softwareseitig ist nun wie folgt vorzugehen:

- Auslesen des FIFO Statusbytes. Dieses gibt die Anzahl der empfangenen Bytes zurück. Sollte es einen FIFO-Overflow geben, ist das höchstwertige Bit des Statusbytes eins.
- Steht im FIFO Statusbyte ein Wert größer null, sind neue Daten vorhanden. Ist der Wert des Statusbytes kleiner 0x80, gibt es keinen FIFO-Overflow, d.h. der FIFO kann ausgelesen werden.
- Das erste Byte des FIFOs enthält die Länge des empfangenen Datenpaketes. Beispiel a: das Paket ist 28 Byte groß.
- Nun werden die restlichen Daten ausgelesen. Beispiel a: 28 Byte Daten – 1 Byte Länge = 27 Byte.
- Sind alle Daten ausgelesen, müssen noch die zwei CRC-Bytes ausgelesen werden. Mit diesen Bytes kann die Integrität der Daten geprüft werden.
- Sollte das Datenpaket größer als der FIFO sein, sind in Schritt 4 nur die Anzahl an Bytes der eingestellten FIFO Größe – 1 Byte Länge – 1 Byte (um den FIFO nicht zu leeren) auszulesen.
- Dann sind die Punkte 1 und 2 zu wiederholen. Beim erneuten Auslesen des FIFOs gibt es nun kein Längenbyte mehr. So können nun alle restlichen Bytes des Datenpaktes ausgelesen werden.
- Nach dem Auslesen der Daten ist erneut der command strobe SRX zu übersenden. Der CC2500 befindet sich wieder im Receive-Modus.
- Sollte ein FIFO-Overflow vorhanden sein, ist zuerst der command strobe SFRX und anschließend wieder der command strobe SRX an den Chip zu senden.
- Im Interrupt-Betrieb werden die Daten in einer eigenen Interrupt-Service-Routine ausgelesen. Der Sprung in diese Routine wird durch einen externen Interrupt des CC2500 ausgelöst. Hierfür ist der Pin GD0 des CC2500 verantwortlich, der bei Interrupt-Signalisierung auf Low gezogen wird. Der Transceiver-Chip wird so konfiguriert, dass jedes neue Datenpaket im FIFO zu einer Auslösung des GD0 Pins führt. Diese Leitung muss dann an einer Interrupt-fähigen Eingangsleitung des Mikrocontrollers verbunden sein.

Senden der Daten über die Funkschnittstelle

Das Senden von Daten kann ebenfalls mit Polling oder über Interrupts gesteuert werden. Beim Versenden von Daten über den CC2500 ist folgendermaßen vorzugehen:

- Übersenden des command strobe SIDLE (geht in den IDLE Zustand)
- Übersenden des command strobe SFTX (setzt den Transmit-FIFO zurück)
- Übersenden des command strobe STX (Einstellung des Frequenz Synthesizers)
- Übersenden des command strobe SFSTXON (kalibriert und aktiviert den Frequenz Synthesizer)
- Schreiben des Längenbytes in den TX-FIFO (im Längenbyte befindet sich die Anzahl der Daten Bytes)
- Schreiben der Datenbytes in den TX-FIFO
- Übersenden des command strobe STX (geht in den Transmit-Zustand und überträgt die Daten)

Reichweitentest

Beim Reichweitentest wurden die Dipolantenne und die im Paket enthaltene Stabantenne des CC2500 Entwicklungskits von Texas Instruments auf ihre Sendereichweite getestet und miteinander verglichen.

Für den Reichweitentest wurde der Mikrocontroller zur Steuerung der Transceiverplatine so programmiert, dass er alle empfangenen Daten an den Absender zurücksendet und einen Zeitstempel anhängt. Mittels des Smart RF04 Evaluation Board und der Software Smart RF Studio wurden entsprechende Datenpakete zum Messobjekt versandt und von dort wieder zurückgeschickt. Über das Smart RF Studio konnten nun die zurückkommenden Daten und die Empfangsstärke in dBm abgelesen werden. Dieses Senden und Empfangen wurde über eine immer größere Distanz in einer Bodenhöhe von ca. 2 Metern durchgeführt, bis Fehler auftraten oder keine Daten mehr am Evaluation Board empfangen wurden. Bei den Reichweitentests sind die in Tabelle 2 gelisteten Daten ermittelt worden.

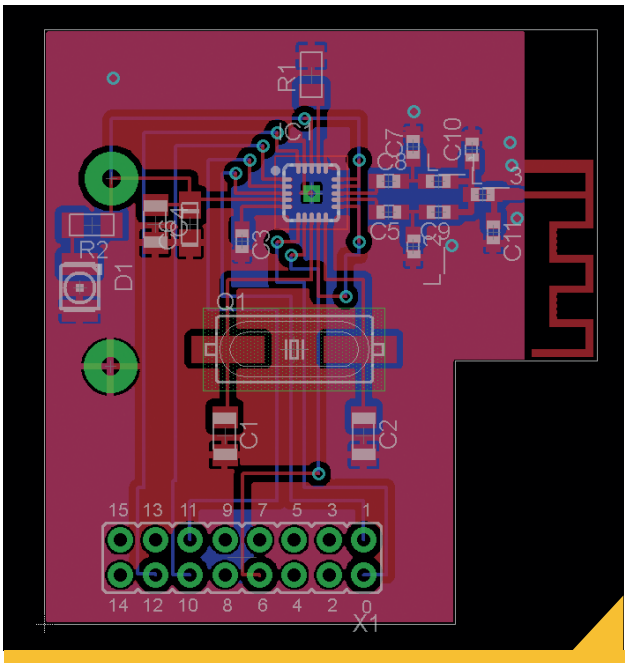
Anhand der Werte in Tabelle 2 ist zu erkennen, dass die Leistung der selbst gefertigten Dipolantenne nur unwesentlich von der gekauften Antenne abweicht. Überraschenderweise besitzen beide Antennen gleichermaßen eine maximale Sendeentfernung von ca. 220 Metern und durch Drehen der Antenne ist nur wenig Richtcharateristik erkennbar. Diese experimentell ermittelten Werte waren stets reproduzierbar und wurden ohne großen Messaufwand (eher qualitativ) durchgeführt. Somit ist die billigere Dipolantenne auf Grund einer praktisch identischen Leistung, der kleineren Bauform und der einfachen Herstellung für das hier entwickelte Transceiversystem geeigneter.

Entfernung in Metern zwischen SDK und Messobjekt	Empfangene Sendeleistung in dBm: Dipolantenne	Originalantenne
221,00	-95,00	-95,50
178,50	-91,00	-91,00
136,50	-90,50	-90,00
93,50	-84,00	-82,00
51,00	-76,00	-73,50
8,50	-59,00	-56,50

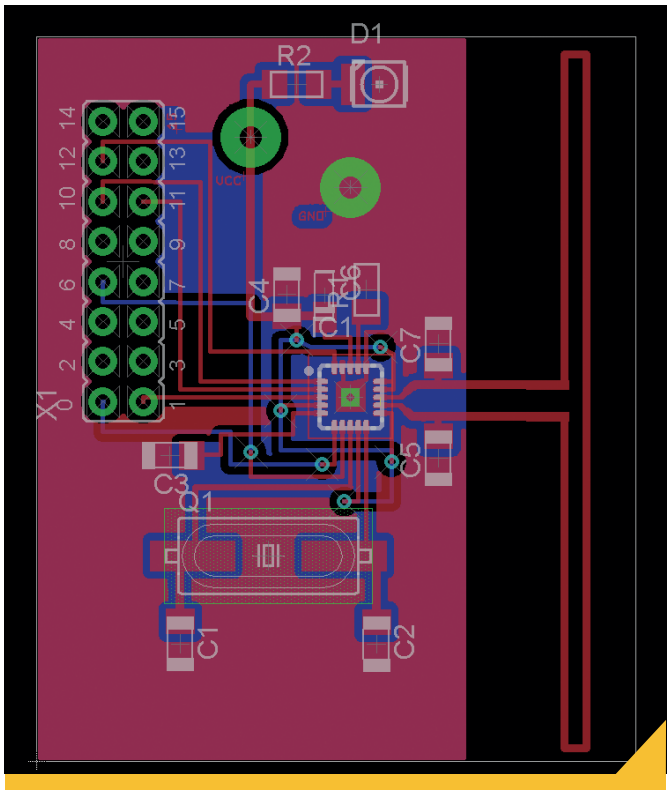
Tabelle 2: Werte des Reichweitentests (Quelle:Eigene)

Weiterer Einsatz

Bei einer weiteren Verwendung des Transceiversystems kann das bestehende Programm auf Zusatzfunktionen erweitert werden, wie z.B. die Umsetzung eines Frequencyhopping-Verfahrens, welches Störungen bei der Übertragung verringern und somit eine fehlerresistentere Kommunikation ermöglichen würde. Das Funksystem könnte auch auf mehrere Teilnehmer erweitert werden, wie man es beispielsweise im ZigBee-Standard (IEEE 802.15.4) mit dem ZigBee Wireless Personal Area Network (PAN) umgesetzt hat.



Layout der Inverted F-Antennen Platine (Quelle: Eigene)



Layout der Dipolantennen-Platine

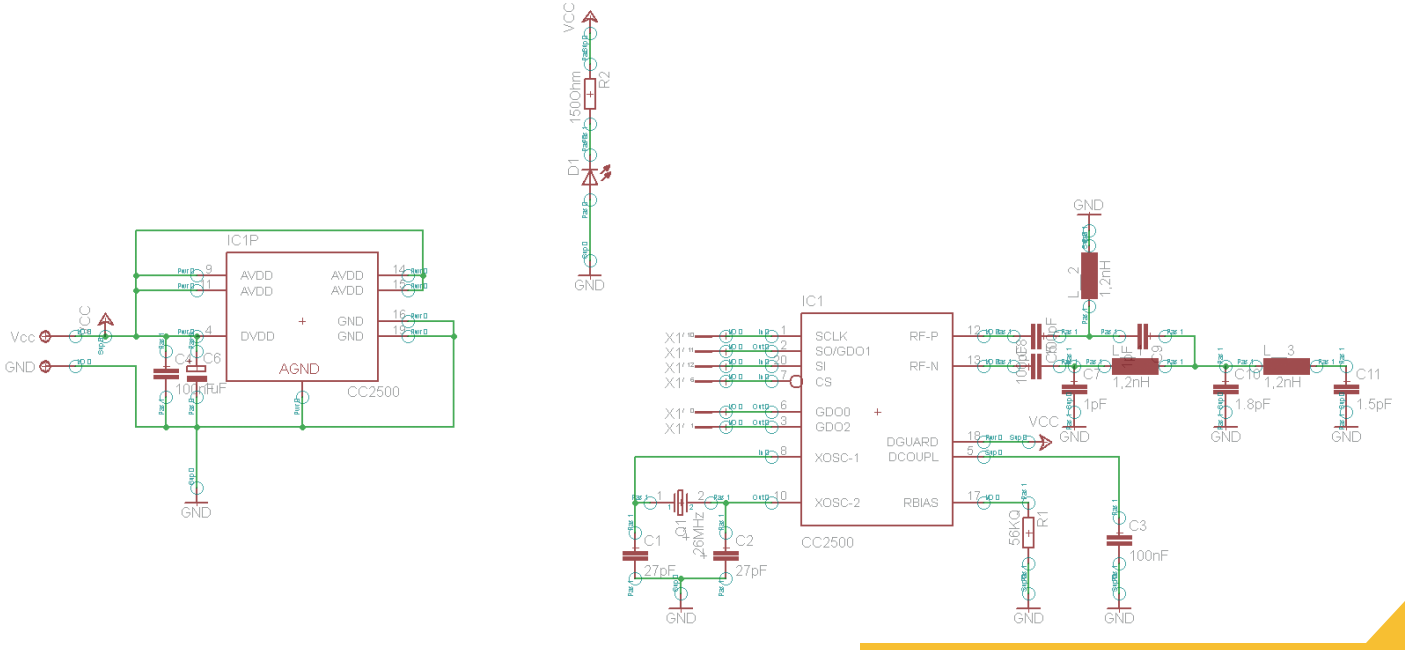
Literaturverzeichnis

[1] Texas Instruments CC2500 PRELIMINARY Data Sheet (Rev.1.2), SWRS040a

[2] Texas Instruments Application Note N049 Software for CC1100/CC2500 and MSP430, SWRA141

[3] Texas Instruments Design Note DN004 Folded Dipole Antenna for CC25xx, SWRA118

[4] Texas Instruments Application Note AN043 Small Size 2.4 GHz PCB antenna, SWRA117D



Schaltplan der Inverted F-Antennen-Platine

Lesen Sie die neue Elektor ein Jahr lang in der ultimativen GOLD-Mitgliedschaft und profitieren Sie von allen Premium-Vorteilen!

elektor GOLD membership

Die Elektor-GOLD-Jahresmitgliedschaft bietet Ihnen folgende Leistungen/Vorteile:

- Sie erhalten **10 Elektor-Hefte** (8 Einzelhefte + 2 Doppelausgaben Januar/Februar und Juli/August) pünktlich und zuverlässig frei Haus.
- **Extra:** Jedes Heft steht Ihnen außerdem als PDF zum sofortigen Download unter www.elektor-magazine.de (für PC/Notebook) oder via App (für Tablet) bereit.
- **Neu & Exklusiv:** Sie erhalten alle 2 Wochen per E-Mail ein neues Extra-Schaltungsprojekt (frisch aus dem Elektor-Labor).
- **Neu & Exklusiv:** Wir gewähren Ihnen bei jeder Online-Bestellung 10% Rabatt auf alle unsere Webshop-Produkte – dauerhaft!
- **Neu & Exklusiv:** Der Online-Zugang zum neuen Community-Bereich www.elektor-labs.com bietet Ihnen zusätzliche Bauprojekte und Schaltungsideen.
- **Extra:** Die neue Elektor-Jahrgangs-DVD (Wert: 27,50 €) ist bereits im Mitgliedsbeitrag inbegriffen. Diese DVD schicken wir Ihnen sofort nach Erscheinen automatisch zu.
- **Extra:** Top-Wunschprämie (im Wert von 30 €) gibts als Dankeschön GRATIS obendrauf!

UMWELTSCHONEND – GÜNSTIG – GREEN

Möchten Sie Elektor lieber im elektronischen Format beziehen? Dann ist die neue GREEN-Mitgliedschaft ideal für Sie!
Die GREEN-Mitgliedschaft bietet (abgesehen von den 10 Printausgaben) alle Leistungen und Vorteile der GOLD-Mitgliedschaft.

elektor GREEN membership

Jetzt Mitglied werden unter www.elektor.de/mitglied!

Ein Blick über den Tellerrand

Infineon Mikrocontroller

von Georg Huba <Georg.Huba@infineon.com>

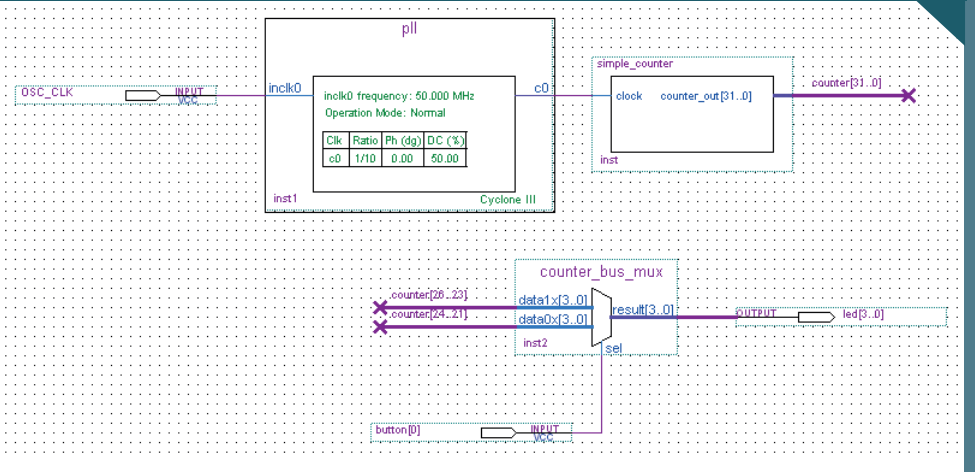
Vorwort Benedikt Sauter: Vor kurzem erhielt ich einen Anruf von David Chang (Infineon Technologies). Er hat im Unternehmen die Aufgabe bekommen Open-Source in das Unternehmen zu bringen bzw. Open-Source vom Unternehmen zu geben. Mit diesem und eventuell weiteren Artikeln wollen wir einen Blick über unseren gewohnten Tellerrand wagen. Wie sieht die Welt hinter AVR, ARM & Co. aus. Neben der Entwicklungsumgebung, die in diesem Artikel vorgestellt wird, interessiert uns natürlich: Was bieten die Infineon Prozessoren für Vorteile, erhält man Datenblätter und Dokumentationen, wo kann man die Prozessoren kaufen und wie sieht es mit einer Community bzw. offenem Support aus?

Die Entwicklungsumgebung DAVE™ von Infineon

Stellen Sie sich vor Sie sind SW-Entwickler für Embedded-Mikrocontroller und Ihr Chef legt Ihnen ein Evaluation-Board mit einem interessanten neuen Mikrocontroller auf den Tisch, mit der Bitte, damit so schnell wie möglich einen BLDC-Motor zum Laufen zu bringen. Sie schauen sich das Board an, sehen schnell um welchen Typ und Hersteller es sich handelt und Sie wissen, dass jeder Mikrocontroller-Hersteller, der was auf sich hält Unterstützung für Motorsteuerung anbietet. Schnell haben Sie Zugriff auf eine Projektsammlung mit fertigen Treibern für die Peripherie. Glücklicherweise wird der von Ihnen benutzte Motor unterstützt und nach 1 - 2 Tagen können sie ihrem Chef Vollzug melden. Im Überschwang der Freude über das gute und schnelle Ergebnis soll nun die Steuerung um eine weitere Achse erweitert werden und die Regeldaten sollen über eine SPI Schnittstellen auf einem LCD dargestellt werden, und… Mit dem ersten Erfolgserlebnis im Rücken durchstöbern Sie Projektsammlungen, Libraries und Referenzbeispiele aufs Neue und werden höchstwahrscheinlich fündig: Für jedes der einzelnen Problemstellungen gibt es Referenzprojekte oder SW-Module mit denen Sie eine jeweils individuelle Lösung entwickeln können, allein, das war es nicht, was Ihr Chef möchte, er wollte, dass alles zusammen gleichzeitig auf einem Chip funktioniert: Die unabhängige Regelung 2er Motoren, plus Ein-/ Ausgabemöglichkeiten plus ein bisschen Kommunikation, plus….

Um dies zu erreichen, müssen Sie sicherstellen, dass sich die verwendeten SW-Module und die darunterliegende HW nicht in die Quere kommen, und dies kann Aufwendig werden. Sie müssen die verfügbaren HW-Ressourcen inklusive der Pin-Zuordnung richtig aufteilen, dann müssen Sie in den benützen SW-Modulen den Initialisierungs-Code und die Registerzugriffe entsprechend ihrer HW-Ressourcenaufteilung anpassen und wenn Sie das dann alles gemacht haben, ihr Chef vorbeikommt und noch eine Zusatzfunktion fordert, kann es sein, dass Sie mit der Ressourcenfrage wieder von vorne beginnen müssen…..

Abb. 1: Graphische Design-Oberfläche eins FPGA-Tools. Die verschiedenen Funktionskomponenten (HW und SW) können konfiguriert und verbunden werden. Das Tool löst die Bedingungen für die Ressourcenzuordnung auf den FPGA-Chip automatisch, konfliktfrei auf.



Die Lösung: “Constraint Logic Programming“

Diese beschriebene Situation soll das Dilemma zeigen, in der wir uns als embedded SW-Entwickler befinden, wenn wir bei der Entwicklung neuer HW-orientierter SW-Projekte auf vordefinierte SW-Komponenten zurückgreifen. Diese Methodik wird üblicherweise “Component Based Programming“ oder “Component based SW Engineering“ genannt und basiert darauf, dass eine Problemstellung in weitgehend unabhängige Teile zerlegt wird, die dann mit vordefinierten und getesteten SW-Komponenten gelöst werden kann. Dieses Konzept steht auch eng in Zusammenhang mit der objektorientierten Programmierung. Die Kombination dieser SW-Komponenten sollte laut Theorie konfliktfrei möglich sein. Die

ist auch weitgehend der Fall, wenn die SW-Komponenten keine HW-Ressourcen exklusiv beanspruchen. Die ist jedoch bei der Programmierung von Mikrocontrollern genau nicht der Fall, ein PWM-Kanal ist üblicherweise fest einem Motor zugeordnet und kann nicht gleichzeitig den 2ten Motor ansteuern.

Will man die Vorteile von "Component Based Programming" wie einfache, schnelle, sicherer und flexible SW Entwicklung auf die HW-orientierte-Programmierung von Mikrokontrollern konsequent anwenden, muss man sich der Frage nach der HW-Ressourcenaufteilung auf Grund sich ändernden Randbedingungen stellen. Viele aktuelle Ansätze auf Basis von normalen Libraries oder Beispielprogrammen tun dies nicht. Aber auch modernere Ansätze, wie die AUTOSAR-Methode, auch eine komponentenbasierte Entwicklungsmethode aus der Automobiltechnik hat diese Problematik mehr oder weniger elegant ausgelagert. Dort wurde der Komponentenbasierte Ansatz bewusst von der Mikrocontroller-HW getrennt. Die Schnittstelle zur HW wird über die MCAL-Programmierschnittstellen (Mikrocontroller Abstraction Layer) definiert und muss in der Konfigurationsphase manuell den jeweils verfügbaren HW-Ressourcen zugeschlüsselt werden.

FPGA-Design-Tools dagegen haben sich der Herausforderung der Ressourcenzuordnung nach sich ändernden Randbedingungen bereits früh gestellt. Ein einfaches FPGA-Design für ein LED-Blinky -Projekt könnte im Schematic-Editor wie in Bild 1 aussehen: Eine Takterzeugung, ein Counter und ein Multiplexer, der über einen externen Taster gesteuert wird und an dessen (bestimmten) Ausgang LEDs angeschlossen sind. Bei der Erzeugung des VHDL-Code und des FPGA-Bitstreams prüft das Tool die HW-Anforderungen und bildet diese auf die verfügbaren HW-Ressourcen des gewählten FPGA-Chips ab, unter Berücksichtigung aller Randbedingungen wie Signalverbindung, elektrische Eigenschaften und Pin-Zuordnung. Eine Erweiterung des FPGA-Designs oder eine Übertragung auf ein anderes FPGA ist jederzeit möglich, solange es eine Auflösung zwischen geforderten Ressourcen und verfügbaren Ressourcen gibt. Das dahinterliegende SW Konzept, um eine solche Auflösung zu finden wird in der Literatur als "Constraint Logic Programming" bezeichnet.

Abb. 2: Gesteuert von einer externe Puls-Quelle soll nach genau 10 Pulse ein PWM-Signal mit einem Duty-Cycle von 1% gestartet werden. Der Duty-Cycle soll kontinuierlich bis 90% erhöht werden und dann soll das PWM-Signal nach 500 Pulse automatisch anhalten und der Puls-Counter soll für einen neuen Durchlauf reaktiviert werden.

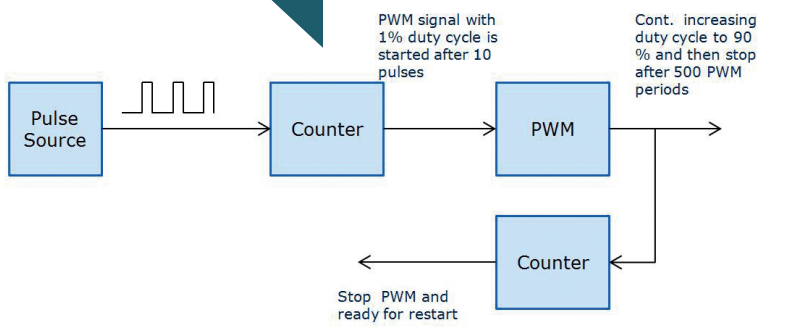
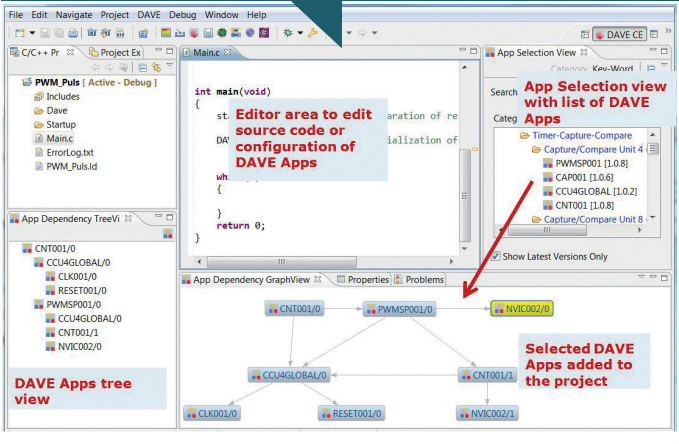
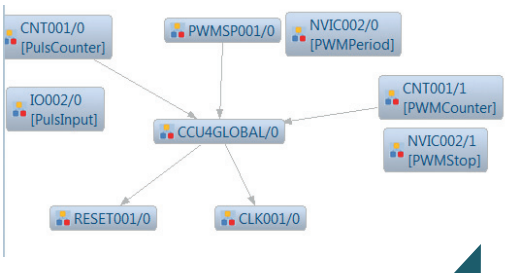


Abb. 3: Screen-Shot DAVE-Entwicklungsplattform mit den verschiedenen Views zu DAVE-Apps.



Ein Ressource-Solver in DAVE™ ist die Lösung

Die neue kostenlose Entwicklungsplattform DAVE in der Version 3 hat sich von den FPGA-Tools inspirieren lassen. Mit DAVE kann der Programmierer, je nach Anforderung, Applikationsorientierte SW-Komponenten in sein Projekt einfügen, konfigurieren und verbinden. Ein Ressource-Solver findet automatisch eine funktionierende Ressourcenzuordnung, unter Berücksichtigung aller Randbedingungen wie Signalverbindung zwischen der Peripherie, oder der Pin-Zuordnung. Entsprechend der Ressourcenzuordnung erfolgt dann die Generierung der Library, so dass der Programmierer nur noch die definierten HW-ressourcenunabhängigen API (Application Programming Interface) in seiner SW einsetzen muss, um die gewünschte Funktionalität zu erreichen. Nun kann Ihr Chef mit neuen Anforderungen vorbeikommen so oft er will, Sie instanziieren einfach neun SW-Komponenten, genannt DAVE-Apps, um die zusätzlichen Anforderungen zu erfüllen und um das leidige Ressource-Management und entsprechende Anpassung der Library kümmert sich DAVE.



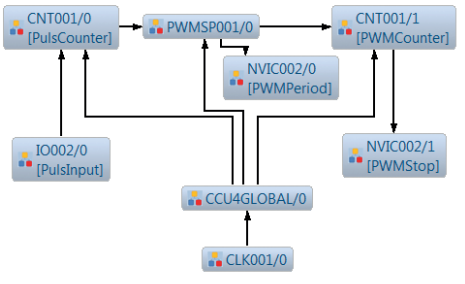


Abb. 4: Zeigt die ausgewählten DAVE-Apps (IO002, CNT001, PWMSP001, NVIC001) plus die automatisch angeforderten DAVE-Apps, um Ressourcen für die gewählten DAVE Apps bereitzustellen.
Im S/W App-Connectivity-View (links) wird die SW Abhängigkeit der DAVE-Apps dargestellt. Im H/W Connectivity-View wird die Signal-/ Event- Verbindung dargestellt.

Wie funktioniert das nun in der Praxis?

Ein kleines Beispiel zeigt am besten wie einfache es ist embbeded SW mit DAVE zu entwickeln. Die Anforderung: Es soll ein PWM-Signal in Abhängkeit von einem Pulsgeber erzeugt werden. Bild 2 zeigt die Details. DAVE kann kostenlos von der Infineon Webseite heruntergeladen und danach installiert werden (www.infineon.com/dave). Die applikationsorientierten SW-Komponenten werden DAVE-Apps genannt, da sie die Funktionalität von kleinen Applikationselementen bis zu einer vollständigen Applikation abdecken können und einfach wie Smart-Phone-Apps aus dem Web heruntergeladen werden können. Die auf diese Weise lokal auf dem PC verfügbaren DAVE-Apps können dann mit dem App-Selektor ausgewählt und dem SW Projekt zugefügt werden. Bild 3 zeigt einen entsprechenden Screen-Shot der DAVE-Entwicklungs-Platform.

Im Folgenden sollen nun die jeweiligen Entwicklungsschritte erläutert werden, um die SW entsprechend den Anforderungen zu entwickeln:

- Schritt 1: Auswahl der DAVE-Apps. Um die Signale des Pulsgebers zu zählen, wählen wir eine Count-App plus eine I/O App, zur Erzeugung der PWM-Signale eine PWM-App und um das PWM Signal nach 100 pulsen mit 90% Duty Cycle abzuschalten wird eine weitere Count-App instanziiert. Zusätzlich werden noch 2 Interrupt-Apps (NVIC-Apps) für den Duty-Cycle-Update und der Reinitialisierung benötigt. Bild 4 zeigt den graphischen DAVE-App-View nachdem diese DAVE-Apps für das Projekt instanziiert wurden. Dabei zeigt sich bereits die erste Besonderheit. Es sind mehr DAVE Apps zu sehen als ausgewählt wurden. Ein DAVE-App kann die Ressourcen einer anderen DAVE-App anfordern. Dies ist der Fall bei der Clock-App, da die Count- und PWM-Instanzen auch eine konfigurierten Clock-Tree benötigen der von der Clock-App bereitgestellt wird.

- Schritt 2: Verbindung der HW-Signale. DAVE-Apps sind nicht nur simple SW-Komponenten, die ausschließlich eine Programmierschnittstelle (API) zur Verfügung stellen, sondern auch HW-Signale beinhalten, die direkt mit den Signalen anderer DAVE-Apps verbunden werden können. Um die hier geforderte Funktionalität erreichen, müssen folgende Signale verbunden werden: Der Status-Ausgang der Count-App (Instanz 0) mit dem Start-Eingang der PWM-App. Status-Ausgang der PWM-App mit dem Count-Input der Count-App (Instanz 1). Event-Signale mit dem jeweiligen Interrupt-Eingang der beide NVIC-App-Instanzen. Bild 5 zeigt den Signal-Connection-View in dem die verschiedenen HW-Signale verbunden wurden.

App	Signal	Connected To	App	Signal
CNT001/0[PulsCounter]				
	Global	<----	CCU4GLOBAL/0	Global
	Status	---->	PWMSP001/0	Input External Start
CNT001/1[PWMCounter]				
	Event Count Match Interrupt	---->	NVIC002/1[PW...	NVIC Interrupt
	Global	<----	CCU4GLOBAL/0	Global
PWMSP001/0				
	Direct Output Pin	<----	Internally Conn...	PWM Output
	Global	<----	CCU4GLOBAL/0	Global
	Period Match Interrupt	---->	NVIC002/0[PW...	NVIC Interrupt
	PWM Status	---->	CNT001/1[PW...	Input

Abb. 5: Verbundene HW-Signale, grün sind Signale die automatisch verbunden werden, die blauen Signale sind die vom Entwickler verbundenen Signale

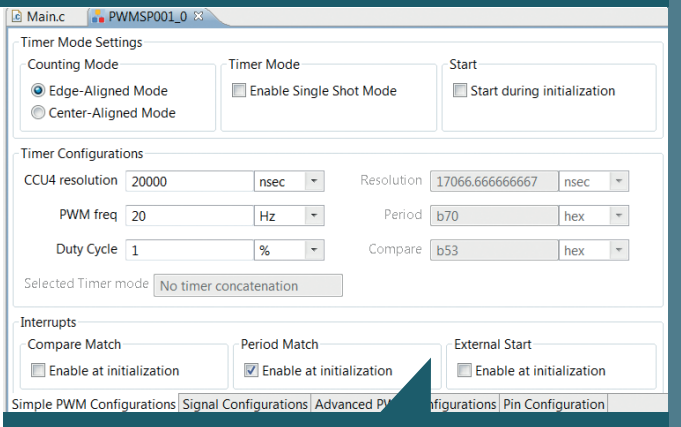


Abb. 6: Graphische User Interface um DAVE Apps zu konfigurieren, hier am Beispiel der PWMSP001 App

- Schritt 3: Konfiguration der DAVE-Apps über das graphische User-Interface. Durch Doppelklick auf eine DAVE-App öffnet sich das graphische User-Interface. Entsprechend der Konfigurationsmöglichkeiten der jeweiligen DAVE-App kann die gewünschte Funktionalit, der Initialisierungsstatus, etc eingestellt werden. Für die Count-App Instanz 0 definieren wir ein Count-Match-Event nach 10 Pulsen, während wir für die Instanz 1 ein Count-Match-Event nach 500 Pulsen definieren. Die Konfigurationen der PWM-App zeigt Bild 6. Bei den NVIC-Apps wird die Interrupt Priorität festgelegt und der Name der Handler-Funktion.

- Schritt 4: Manuelle Pin-Festlegung. Die bisherigen Einstellungen und Konfigurationen der DAVE-Apps sind nicht auf bestimmte HW-Ressourcen bezogen. Bisher überlassen wir es dem Solver die Peripherie-Ressourcen auszuwählen. Bei der Pin-Auswahl ist es oftmals notwendig Vorgaben zu machen, um beispielsweise die SW auf ein vorhandenes Board zu portieren. Dazu gibt es einen Manual-Pin-Assignment-View (zunächst in Tabellenform, später auch als Graphik) in dem beispielsweise das PWM-Ausgangssignal und der Puls-Eingang einem bestimmten Pin zugeordnet werden sollen (Bild 7).

- Schritt 5: HW-Ressourcenzuordnung und Libray-Code-Generation. Um sicherzustellen, dass die gewünschten Funktionen auch auf den Chip

App	Resource	Port-Pin/Pin Number
PWMSP001/0		
	pin_directoutput	P3.9 / #12
IO002/0[PulsInput]		
	pin	P2.3 / #71

Abb. 7: Manuell Zuordnung der Pins damit der Ressource-Solver die Randbedingen eines vorhandenden PCBs bei der Ressource-Zuordnung berücksichtigen kann

"passen", wird der Ressource-Solver automatisch nach bestimmten Ressourcen relevanten Transaktionen gestartet. Man kann den Ressource-Solver auch explizit, manuell starten. Das Ergebnis: die DAVE-Apps wurden automatisch auf die verfügbaren Chip-Ressourcen konfliktfrei unter Berücksichtigung aller Randbedingungen zugeordnet. Entsprechend den zugeordneten Ressourcen erfolgt danach die Erzeugung des Library-Code. Je DAVE App werden spezifische Header-Files und C-Source-Files erzeugt, die die Initialisierungsfunktionen und die API für die verschiedenen Funktionsaufrufe beinhalten. Obgleich es bei dem erzeugten Code um c-code handelt, folgt die Codeerzeugung vorgaben der objektorientierten Programmierung. Der erzeugte Library-Code ist unabhängig von der Anzahl der DAVE App-Instanzen, je Instanz wird eine Datenstruktur erzeugt, in der die jeweilige Konfiguration sowie Ressource Infor-

mationen gespeichert sind. Im Grunde sind die DAVE-Apps Klassen, die beim Zufügen zu eine Projekt als deren jeweiliges Objekt instanziiert werden.

- Schritt 6: Benutzung der API im User-Code. Nach diesen ersten 5 Schritten sind alle HW relevanten Aufgaben der SW-Entwicklung erledigt und man kann sich voll und ganz auf das abstrakte Programmieren konzentrieren. In unserem Fall ist es lediglich notwendig 2 Interrupt-Handler-Funktionen zu definieren: Eine Funktion, die von der PWM-Period-Match getriggert wird, die im Wesentlichen den Duty-Cycle update durchführt und nach Erreichen des 90% Duty-Cycles die Count-App-Instanz 0 startet eine weitere Funktion, die vom Count-Event-Match der Count-App Instanz 1 getriggert wird und die im Wesentlichen eine neue Sequenz vorbereitet. Bild 8 zeigt die beiden Code-Sequenzen. Die Funktionsnamen beginnen jeweils mit dem App-Namen als Präfix gefolgt von einem funktionalen Suffix, das erste Argument der Funktion ist fast immer ein Zeiger auf die Daten-Struktur der jeweiligen App-Instanz. Mit dem in DAVE verfügbaren kostenlosen GNU-Compiler und TASKING-Debugger kann der Code kompiliert und auf dem Zielsystem getestet werden.

```
int main(void)
{
    status_t status;        // Declaration of return variable for DAVE3 APIs
    DAVE_Init();            // Initialization of DAVE Apps
    status = CNT001_Start(&CNT001_Handle0); // Start pulse counter
    status = CNT001_Start(&CNT001_Handle1); // Start PWM counter
    status = PWMSP001_Start(&PWMSP001_Handle0); // Start PWM
    while(1)
    {
```

```
void Period_Handler(void)  // interrupt handler for PWM period match
    {
    static float duty1 = 1;
    status_t status;

    if (duty1 < 91){
        status = PWMSP001_SetDutyCycle(&PWMSP001_Handle0, duty1); // change  PWM duty cycle
        duty1 ++;
    }
    else {
    duty1 = 1;
    status = PWMSP001_DisableEvent(&PWMSP001_Handle0,PWMSP001_PERIODMATCHEVENT); // disable period interrupt
    status = CNT001_ResetCounter(&CNT001_Handle0); // reset pulse counter
```

```
void PWM_Count_Handler(void)  // interrupt handler for PWM count match
    {
    status_t status;
    status = PWMSP001_SetDutyCycle(&PWMSP001_Handle0, 1);
    status = PWMSP001_Stop(&PWMSP001_Handle0);
    status = CNT001_ResetCounter(&CNT001_Handle1);
    status = PWMSP001_Start(&PWMSP001_Handle0);
    status = PWMSP001_EnableEvent(&PWMSP001_Handle0,PWMSP001_PERIODMATCHEVENT);
```

Abb. 8: Mit wenigen Funktionsaufrufen kann dieses Beispiel programmiert werden. Die von DAVE bereitgestellten Library-Funktionen starten jeweils mit dem Namen der DAVE-App, gefolgt von einer funktionalen Beschreibung, das erste Argument ist immer der Zeiger auf die Struktur der DAVE-App-Instanz

- Das fertige Projekt kann auch aus der auf dem Web verfügbaren DAVE-Project-Library geladen werden, Key-Word: Elektronik-17

Jetzt kann es richtig losgehen

Obgleich es sich hier um ein sehr einfaches Beispiel handelt, zeigt sich doch das Einsparpotential welches In DAVE steckt, auch ein erfahrener SW-Entwickler würde für diese Aufgabe ohne DAVE ein Mehrfaches der Zeit benötigen. Wenn sich die Anforderungen nun erweitern oder ändern, kann man auf das existierende Projekt einfach aufsetzen und weitere benötigte DAVE-Apps instanzieren. Die heute vorhanden DAVE-Apps-Library beinhaltet bereits ein Vielzahl von DAVE-APPs, welche einen breiten Anwendungsbereich abdecken und die ständig kontinuierlich erweitert werden. Bild 9 zeigt eine Übersicht über die aktuell verfügbaren DAVE Apps. Das einzigartige an DAVE und am Konzept der DAVE-Apps ist die Flexibilität mit der die gewünschte Funktionalität durch die Komposition verschiedener DAVE-Apps und entsprechende HW-Signal-Verbindung "zusammengebaut" werden kann. Es ermöglicht die volle und umfangreiche Komponentenbasierte Nutzung der HW ohne das man sich im Details in das HW-Referenz-Manual vertiefen muss. Das Konzept unterstützt auch die Einbindung existierender SW-Module oder kommerzielle SW-Lösungen von Drittanbietern. Dabei geht es im Wesentlichen darum, dass die benötigten HW-Ressourcen dieser SW-Module vom Ressource-Solver nicht für andere Zwecke verwendet werden. Die geschieht durch die künftige Möglichkeit, dass HW-Ressourcen reserviert werden oder aber, dass solche Komponenten durch ein dedizierte DAVE-App unterstützt werden, wie zum Beispiel die Konfiguration (über erzeugte Header-Files) und der flexiblen Ressourcenzuordnung. Dieser Ansatz wird auch dadurch gefördert, dass in Zukunft eine umfangreiches DAVE-App-Devlopment-Kit kostenlos zur Verfügung gestellt wird. Viele SW-Entwickler die bereits lange mit einer existierenden Tool-Chain arbeiten ist vielleicht nicht wohl beim Gedanken auf ein neues Tool umsteigen zu müssen, wenn sie die Vorteile von DAVE nutzen wollen. Auch hier zeigt sich DAVE flexibel. DAVE ist zwar eine komplette Entwicklungsplattform, man kann sich aber auf die Funktion der komponentenbasierten Code-Erzeugung beschränken und den Code in andere Tool-Chains importieren. Manche Toolanbieter arbeiten sogar daran nur das DAVE-Plug-in zur Konfiguration der DAVE Apps und zu Erzeugung des Library-Codes in deren eigene Eclipse basierte Tool-Chain zu integrieren. Aktuell werden von DAVE die Bausteine der XMC4500-Serie von Infineon unterstützt. All relevanten DAVE-Apps werden zur-

Basic Applications
- ❑Timer
- ❑CAN
- ❑I2S, I2C, UART, SPI
- ❑ADC, Filter,…
- ❑Delta Sigma demodulator
- ❑ Waveform generation
- ❑Resolver
- ❑PWM
- ❑Timer, Capture, Counter
- ❑Position Interface
- ❑Ethernet / Phy
- ❑USB
- ❑Touch
- ❑Flash
- ❑Temp. Control

Middle Ware
- ❑ CMSIS RTOS
- ❑USB stack, class drivers
- ❑TCP/IP stack plus HTTP, FTP, SNMP
- ❑ File System
- ❑GUI lib

Complex Applications
- ❑Motor Control
- ❑Power Conversion
- ❑Webserver
- ❑Modbus

Service Apps
- ❑DMA, Interrupt, I/O, CRC, AES, Power Mgmt.….

Abb. 9: Überblick über die kostenlos verfügbare DAVE-Apps: Ein großer Block sind die Basic Applications und Service Apps. Es steht aber auch ein komplettes Set von Middle-Ware-Applikationen wie CMSIS-RTOS, File-System, Graphic-Bibliothek, USB- und TCP/IP-Stack zur Verfügung. Von großer Hilfe sind auch die DAVE-Apps für komplexe Applikationen wie Motor-Control, Webserver, Modbus, etc.

zeit zügig auf die XMC4400-, XMC4200- und XMC4100-Serie sowie die XMC 1000-Familie portiert. Insbesondre auch mit dem geplante DAVE-App-SDK sollte die Anzahl der verfügbaren DAVE-Apps kontinuierlich weiter zunehmen. Mehr Komponenten bedeuten mehr Optionen Ihre Anforderungen zu lösen. Stellen Sie sich vor Sie sind SW-Entwickler und Ihre Arbeit ist geprägt von Termindruck und sich ändernden Anforderungen, mit DAVE und DAVE-Apps meistern Sie diese Anforderungen souverän. Für mehr Informationen zu der 32-bit Industrie Mikrocontroller basierend auf den ARM® Cortex™ M4: www.infineon.com/xmc

Technische Unterstützung und Plattformen die helfen

Infineon FORUM: Wir haben ein kostenloses Infineon Forum wo unsere Experten und andere Mitglieder technischen Fragen rund um DAVE und Mikrocontrollern beantworten. Außerdem findet Ihr auf unseren Facebook-Tabs, neben den Trainings, weitere spannende Themen zu Infineon Mikrocontrollern.

www.infineon.com/Forum

@Facebook: Hat euch dieser Artikel gefallen? Auf unserer Infineon Microcontroller Facebookseite habt ihr die Möglichkeit uns direktes Feedback zu geben und Fragen in die Runde zu stellen.

www.facebook.com/infineon.microcontroller

Corporate Facebook Channel: www.facebook.com/infineon

Auch wir werden auf der Embedded World 2013 in Nürnberg sein und wir haben auch ein Highlight für euch. Besuchen sie uns gerne an unserem Stand. Am Studentsday bekommt ihr ein kostenloses XMC4500 Development Kits solange der Vorrat reicht

Links

DAVE: www.infineon.com/dave

Support: www.infineon.com/dave-support

Anzeige

Sicherheit?

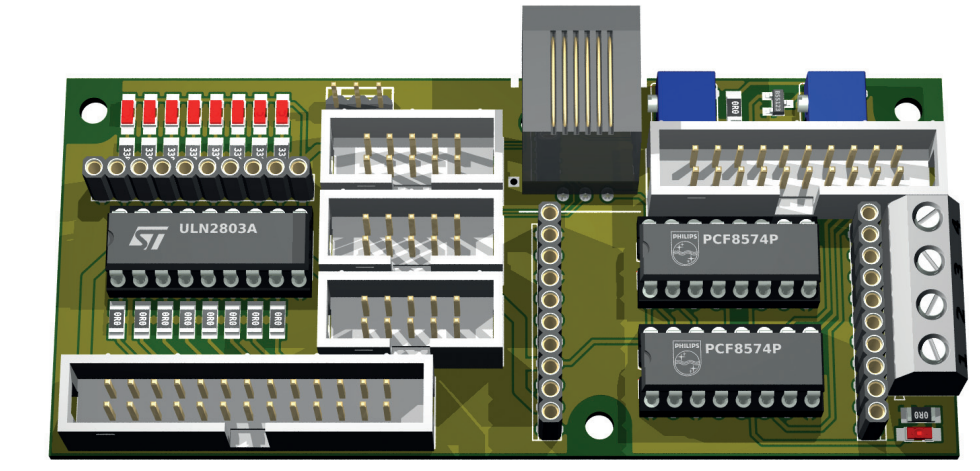
Unser neues Open-Source-Projekt

www.picosafe.de

„Berrybush Pi“

Eine Erweiterungsplatine für das Raspberry Pi (Dokumentation)

Marc Mack <Marc.Mack@gmx.net>



Die hier vorgestellte Erweiterungsplatine für das Raspberry Pi befindet sich derzeit noch im Entwicklungsstadium. Im weiteren Verlauf wird diese Platine mit der Bezeichnung Berrybush Pi erwähnt. Der Name soll die Vielseitigkeit der Erweiterungsplatine darstellen und dient als Metapher für einen Beerenstrauch, an welchem viele unterschiedliche Beeren hängen, diese wiederum sind über das Geäst in irgendeiner Art und Weise miteinander verbunden. Diese Vielseitigkeit spiegelt sich im Berrybush Pi wieder, denn es stellt eine allgemeine Ausgangsplattform für die low-level Hardwareentwicklung mit dem Raspberry Pi dar. Sicherlich gibt es genügend andere Erweiterungsplatinen auf dem Markt, jedoch stützt sich das Berrybush Pi auf die Steckverbindungsdefinitionen aus der Roboternetz Community. Dies bietet die Möglichkeit bereits existierende Module auch mit einem Raspberry Pi zu betreiben. Die hier vorgestellte Platine übernimmt dabei die Pegelanpassung für den 5V Betrieb. Somit muss man sich keine weiteren Gedanken über die Pegelunterschiede machen (außer bei einer Schnittstelle) und kann gleich mit der Peripherieentwicklung beginnen oder bereits verfügbare Hardware anbinden. Auch die Spannungsversorgung des Raspberry Pi findet über das Berrybush Pi statt. Somit kann über eine Energiequelle ein komplettes System versorgt werden.

Hardwarebeschreibung

Eine kurze Zusammenfassung der Leistungsmerkmale des Berrybush Pi:

- kompakte Maße: Länge: 100mm, Breite: 45mm
- Individuell anpassbare Energieversorgung
- Energieversorgung des Raspberry Pi
- Einfacher Zugang zu den Schnittstellen via Flachbandkabel
- 8x Darlington-Transistoren, mit Leuchtdioden Verbindung
- RS232, RS485, I2C und SPI Steckverbinder, teilweise mit 3,3V oder 5V
- LCD-Anschluss mit Potentiometer für Kontrast- und Helligkeitseinstellung
- Verwendung von DIL Bauelementen zur einfachen Reparatur im Fehlerfall

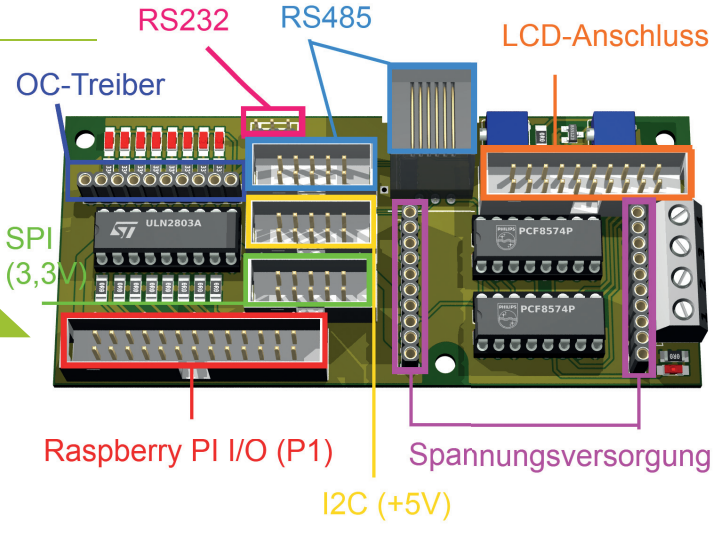
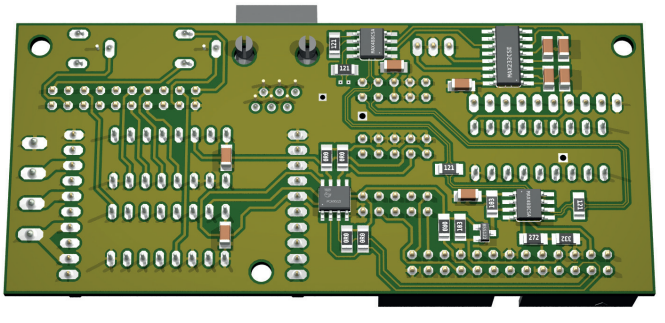


Abb. 1: Hardwarebeschreibung des des Berrybush Pi; Oberseite



Raspberry PI I/O (P1)

Über den 26-poligen Wannenstecker stellt das Raspberry Pi low-level Schnittstellen zur Verfügung (Abb. 3). Diese Schnittstellen eignen sich besonders gut, um eigene Anwendungen zu realisieren. Die Pinbelegung des 26-poligen Wannensteckers wurde auf dem Berrybush Pi auch verwendet. Dies ermöglicht eine einfache Verbindung der beiden Platinen über ein Flachbandkabel. Allerdings gibt es dabei ein Problem: Das Raspberry Pi arbeitet mit 3,3V und viele Hobbyelektroniker arbeiten mit 5V. Dies erschwert die Anbindung an ein 3,3V System wie dem Raspberry Pi. Diese Problematik wurde auf dem Berrybush Pi berücksichtigt. Es wandelt die 3,3V Pegel in 5V Pegel um.

Abb. 3: Pinbelegung des Wannensteckers, Raspberry Pi

Spannungsversorgung

Über die beiden Buchsenleisten J3 und J4 im Bild (Abb. 4) pink hervorgehoben kann eine individuelle Spannungsversorgung realisiert werden. Die vier Schraubklemmen von X7 ermöglichen eine einfache Verbindung zur Platine. Anschließend kann die Spannung an der Buchsenleiste J3 abgegriffen werden. Über J4 werden die erzeugten 3,3V und 5V Gleichspannung dann in das Board eingespeist. Zusätzlich ist es möglich die erzeugte Spannung an X7 zurückzuführen und anderweitig zu verwenden. Werden die beiden Buchsenleisten J3 und J4 gebrückt, können die benötigten 5V auch an der Schraubklemme X7 direkt gespeist werden. Somit ist die Versorgung sehr individuell gestaltbar. Was allerdings immer berücksichtigt werden muss, ist dass das Raspberry Pi immer über die +5V Leitung mit mindestens 1000mA versorgt wird! Niemals sollte man versuchen das Board über die 3,3V zu versorgen. Um dies zu verhindern, wurde ein Schutzwiderstand auf dem Berrybush Pi verbaut welcher den Strom auf 50mA zum Raspberry Pi begrenzt.

Open-Collector Treiber

Um die ersten Schritte bei der Programmierung der GPIO Schnittstelle machen zu können ist es immer hilfreich, wenn LEDs zur Verfügung stehen. Sie ermöglichen es, auf einfache Art und Weise die Funktion zu überprüfen. Aus diesem Grund wurde auf dem Berrybush Pi ein Treiber-Array mit acht Darlingtontransistoren verbaut (Abb. 5). Das Treiber-Array ist über Vorwiderstände (blau umrahmt) direkt mit den GPIO Pins des Raspberry Pi verbunden. Durch den Treiberbaustein wird es ermöglicht, induktive Lasten wie zum Beispiel Relais zu schalten. Diese können über die 10-polige Buchsenleiste angeschlossen werden. Aktuell kommen zwei verschiedene Konfigurationen des Treiber-Arrays in Frage. Entweder wird ein ULN2803A als IC verwendet und die Vorwiderstände sind lediglich 0 Ohm Brücken oder ein ULN2801A mit Vorwiderständen zwischen 800 Ohm und 1 kOhm. Abhängig von der Toleranz der Eingangspegel am ULN2803A.

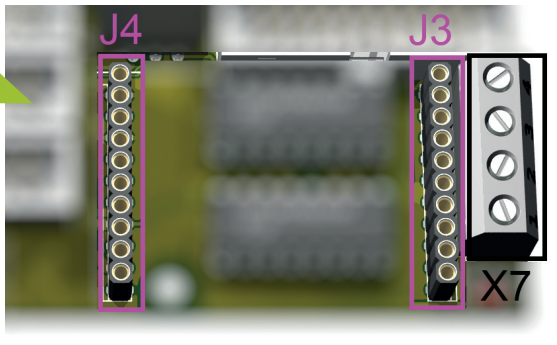


Abb. 4: Spannungsversorgung über die beiden Buchsenleisten J3 und J4

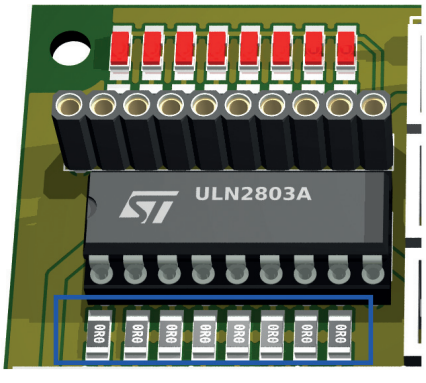


Abb. 5: Treiber-Array mit acht Darlingtontransistoren

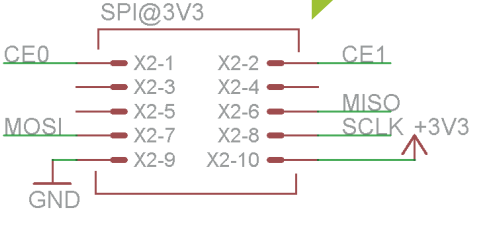
SPI-Interface

Bei der Pinbelegung dieses Interfaces orientierte man sich an der Vorgabe aus dem Roboternetz (siehe: Link[2]). Allerdings ist das Pinning nicht vollständig, weshalb auch nicht das für den Standard vorgesehene Logo verwendet wird.

Abb. 2: Bestückte Ansicht der Unterseite der Platine

Das SPI-Interface ist das einzige Interface auf dem Board, welches direkt an das Raspberry Pi durchgeschleift wird und absolut ohne Schutzbeschaltung versehen ist Abb. 6). Darum sollte man beim Umgang mit dieser Schnittstelle äußerst vorsichtig sein und die nötigen Schutzvorkehrungen und Pegelanpassungen auf dem anzuschließenden Gerät treffen.

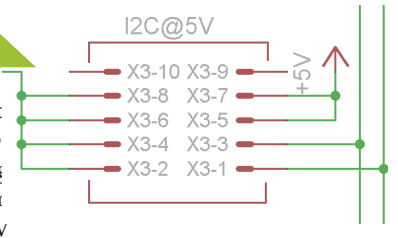
Abb. 6: SPI 3V3



I2C Bus

Der I²C-Bus ist ein sehr weit verbreiteter Bus um diverse elektronische Bauelemente miteinander zu verbinden. Er wurde bereits in den 90er Jahren von der Firma Philips entwickelt und befindet sich heute in der Version 4. Somit ist es wichtig, dass auch dieser Bus auf der Erweiterungsplatine bereit gestellt wird. Bei der Pinbelegung des 10-poligen Wannensteckers wurde wieder auf die Emp fehlung des Roboternetzes geachtet. Allerdings wurde hier die Leitung der Batteriespannung weggelassen. Wenn diese benötigt wird, muss sie separat über eine unabhängige Leitung zum Gerät geführt werden. Der I²C-Bus wird mit 5V Pegeln betrieben. Die Anpassung von 3,3V auf 5V übernimmt hier der PCA9515 Chip von Philips, welcher genau für diese Anwendung entworfen wurde (Abb. 7).

Abb. 7: I2C 5V



RS485 Bus

Ein weiterer Bus, welcher auf der Platine verwendet wird, ist der RS485. Dieser Bus arbeitet mit differenziellen Leitungspaaren, was ihn robuster gegen elektromagnetische Störungen macht.

Er kann in zwei Betriebsmodi verwendet werden: Vollduplex oder Halbduplex.

- Im Vollduplex-Betrieb kann der Bus zeitgleich Daten senden und empfangen. Dabei benötigt er vier Signalleitungen.
- Während im Halbduplex-Betrieb Daten entweder gesendet oder empfangen werden können. Es werden nur zwei Leitungen benötigt.

Der RS485 Bus wurde auf der Platine für den Vollduplex-Betrieb entworfen. Allerdings kann er durch kleine Modifikationen auch im Halbduplex-Modus betrieben werden.

Wie auch bei den vorherigen Busschnittstellen wurde bei der Pinbelegung nach der Roboternetz-Empfehlung gearbeitet jedoch die Batteriespannung weggelassen. Zusätzlich wird die Schnittstelle auch über einen RJ11 Stecker zur Verfügung gestellt.

RS232 Schnittstelle

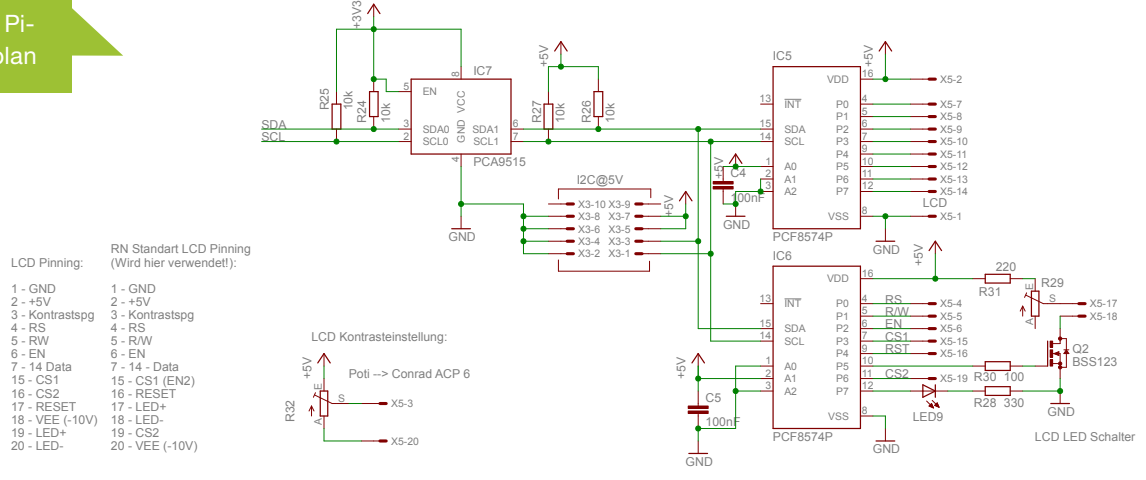
Die RS232 Schnittstelle stellt hier eine RS232 Schnittstelle in abgewandelter Form dar. Die Pegel werden über einen MAX232 so gewandelt, dass die Schnittstelle mit einem normalen PC an der seriellen Schnittstelle (COM-Port) verbunden werden kann. Allerdings wird auf dem Berrybush Pi das TTL Signal der seriellen Schnittstelle nicht direkt zum Raspberry Pi weitergeleitet sondern über einen RS485 Transceiver in den RS485-Bus eingespeist.

Dadurch soll es ermöglicht werden über ein Terminalprogramm die Datenübertragung auf dem RS485-Bus überwachen und debuggen zu können. Wenn keine weiteren RS485 Geräte am Bus hängen ist eine normale serielle Kommunikation mit dem Raspberry Pi ohne Busprotokoll möglich. Dann können auch normale serielle Endgeräte (ohne Hardware Handshake) angeschlossen werden.

LCD Schnittstelle

Als Highlight verfügt das Berrybush Pi über einen LCD-Anschluss. Die Pinbelegung orientiert sich auch hier an der Definition vom Roboternetz. Dabei wird die Pinbelegung eines 20-poligen Wannensteckers verwendet. Laut Definition sollte diese Pinbelegung eher nicht verwendet werden dennoch bietet sie die größtmögliche Flexibilität bei der Anbindung verschiedenster LC-Displays. Auf der Platine wurden auch gleich die Potentiometer zur Einstellung von Kontrast und Helligkeit verbaut (Abb. 8). Man benötigt also nur ein für das jeweilig verwendete LCD angepasstes Flachbandkabel und schon kann es losgehen. Das LCD ist über zwei fest adressierte Portexpander PCF8574 von Philips am I²C Bus angebunden. Einer der beiden Chips erledigt die Datenübertragung, während der andere die Datenflusssteuerung übernimmt. Der Chip mit der Datenflusssteuerung ermöglicht es außerdem, die Hintergrundbeleuchtung und eine zusätzliche LED ein- und auszuschalten. Möchte man kein LCD verwenden, stehen 14 I/O-Leitungen und eine LED zur Verfügung.

Schaltplan 1: Berrybush Pi-Stromlaufplan



Fazit

Das Berrybush Pi stellt eine Vielzahl an Schnittstellen für das Raspberry Pi bereit. Durch die individuell gestaltbare Spannungsversorgung und die enthaltene Schnittstelle für ein LC-Display ist ein schneller Einstieg in die low-level Programmierung des Raspberry Pi möglich.

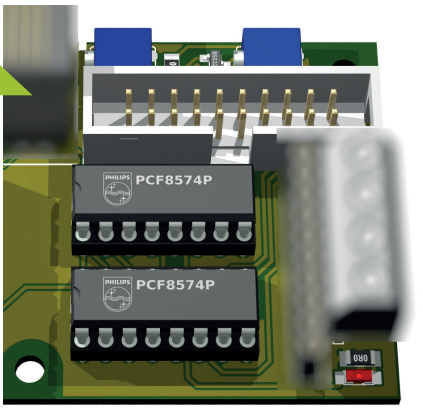
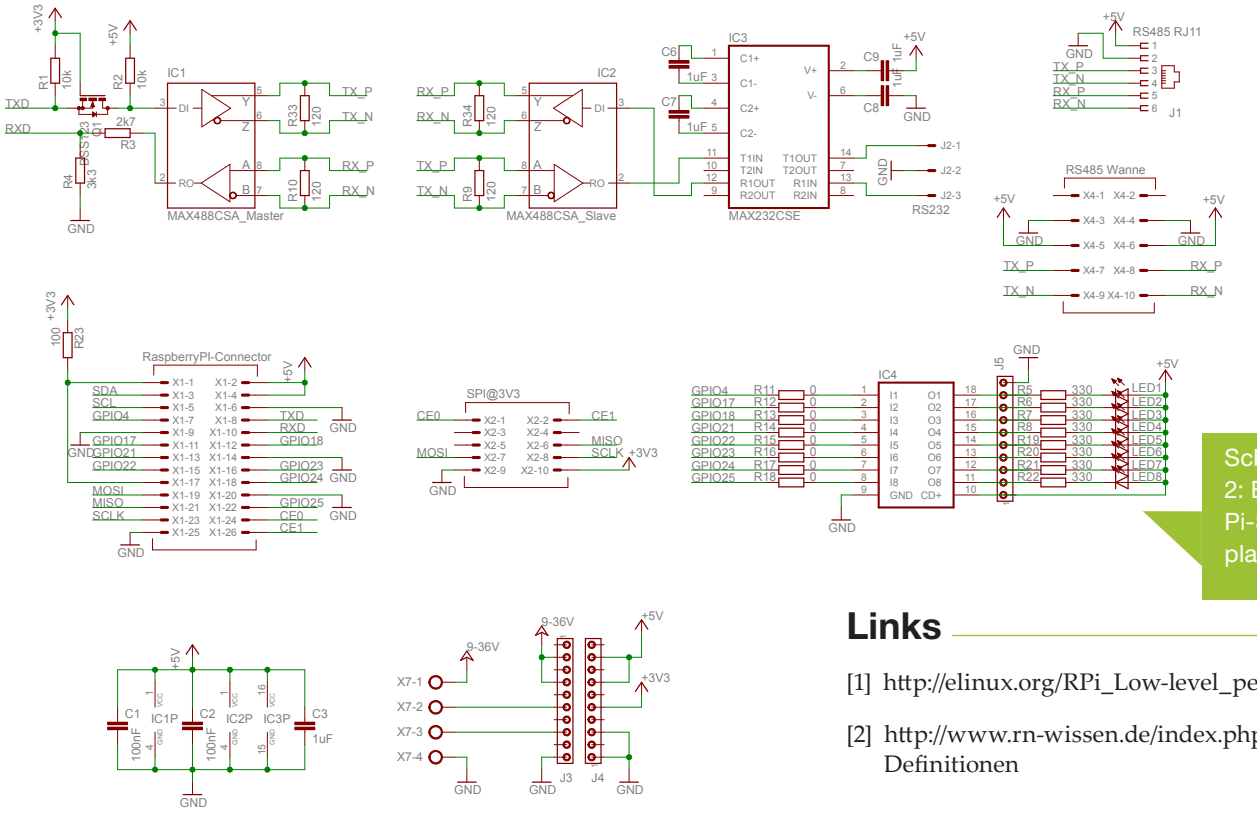


Abb. 8: Potentiometer zur Einstellung von Kontrast und Helligkeit

Es folgt ein Aufruf des Autors: Falls dieser Artikel Ihr Interesse an einem Berrybush Pi geweckt hat und Sie in der Lage sind die Platine selbst zu bestücken, können Sie mich gerne kontaktieren. Ich würde diese Möglichkeit gerne nutzen um, bei ausreichender Nachfrage, eine Sammelbestellung bei einem Leiterplattenhersteller aufzugeben.

ACHTUNG!
Dabei handelt es sich um einen frühen Prototypen der Platine und die Funktionalität kann nicht gewährleistet werden. Es kann passieren, dass die Platine nicht oder nur teilweise verwendet werden kann, oder dass das Raspberry Pi zerstört wird. In jedem Fall wird vom Autor keine Verantwortung übernommen. Die Verwendung des Berrybush Pi geschieht auf eigene Gefahr. Wenn Sie jetzt noch immer Interesse haben und sich mindestens 10 Leute melden, wird eine Sammelbestellung aufgegeben.



Schaltplan 2: Berrybush Pi-Stromlaufplan

Links

[1] http://elinux.org/RPi_Low-level_peripherals

[2] http://www.rn-wissen.de/index.php/RN-Definitionen

Stellenanzeige

stampay

Du bist technikbegeistert und schreibst gerne schöne Software?

Dann komm' zu uns in ein junges und tatkräftiges Startup nach Augsburg und gestalte mit uns die Zukunft der Kundenbindung.

Du bist clever, motiviert, selbstständig und willst mit Deiner Arbeit Spuren hinterlassen?

Wir entwickeln anspruchsvolle Software und verwenden dafür anspruchsvolle Technologien. Dazu bauen wir auf Ruby on Rails, Android und iOS.

Du suchst Herausforderungen, flexibles, eigenverantwortliches Arbeiten, und den Einstieg in ein junges und dynamisches Unternehmen?

Du bist Informatiker, Wirtschaftsinformatiker, Mathematiker oder Designer? Dann bist Du bei uns genau richtig. Bewirb Dich jetzt!

Bewerbung unter
Jennifer West
working@stampay.com

Infos unter
www.stampay.com/jobs

Stampay GmbH
Auf dem Kreuz 11
86152 Augsburg
Telefon: +49 (0) 821 / 450 300 0

Marktplatz / Neuigkeiten

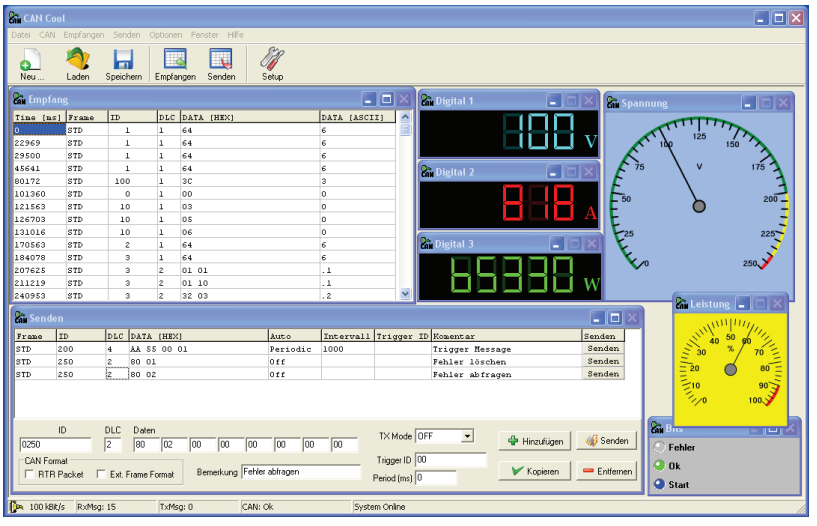
Die Ecke für Neuigkeiten und verschiedene Produkte

CANcool: Open-Source CAN Visualisierung

Klaus Demlehner von MHS-Elektronik hat ein neues CAN Analyse Tool entwickelt. Das Programm ist in Delphi V. 6 als Open-Source unter der GPL verfügbar. Als Hardware wird aktuell Tiny-CAN und SL-CAN Hardware (Lawicel) unterstützt. Die Software soll viele weitere CAN-Module in Zukunft unterstützten.

http://www.mhs-elektronik.de/userdata/downloads/can_cool.zip

Die aktuelle Version ist noch im Beta Stadium! Die Hardware muss beim starten des Programms bereits mit dem PC verbunden sein.



Kostenlose GNUBLIN Schulung in Augsburg

Einsteigerkurs für GNUBLIN von Benedikt Sauter. Nach einer kurzen Einführung in die wichtigsten Komponenten Rund um embedded GNU/Linux werden mit dem Gnublin Board kleine Beispiele gezeigt. Der Kurs dient als erste Orientierung und Einstieg.

- Session 1: Einführung GNU/Linux / Unterschiede Mikrocontroller
- Session 2: Gemeinsame Mini-Anwendung
- Session 3: Fragen & Diskussionen

Zu der Schulung kann man sich kostenlos auf der Internetseite anmelden:

http://schulungen.embedded-projects.net/

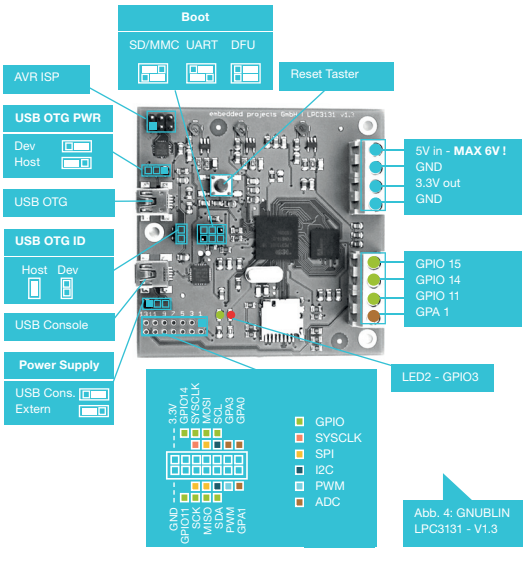


Abb. 4: GNUBLIN LPC3131 - V1.3

LinuxTage in Augsburg und Chemnitz

Wir freuen uns dieses Jahr auf den LinuxTagen in Augsburg und Chemnitz dabei sein zu können:

Chemnitz 16.-17. März 2013:

- Vortrag: Embedded GNU/Linux mit Gnublin (Benedikt Sauter)
- Vortrag: Picosafe: Open-Source USB-Stick für Sicherheitsanwendungen (Michael Hartmann)

Ein Projektstand haben wir ebenfalls dabei

http://chemnitzer.linux-tage.de/2013/

Augsburg 23. März 2013:

- Vortrag: Embedded GNU/Linux mit Gnublin (Benedikt Sauter)
- Vortrag: picosafe - eine abgesicherte mobile Plattform (Prof. Dr. Harald Görl)

http://www.luga.de/Aktionen/LIT-2013/

PANEL-POOL®
Beta LAYOUT

Frontpanel

Digitaldruck auf unterschiedlichsten Materialien

Jetzt auch Lasergravur und Befestigungsbolzen!

PANEL-POOL® ist eine eingetragene Marke der Beta LAYOUT GmbH

www.panel-pool.com

Beta LAYOUT
create:electronics

eSTORE®
Beta LAYOUT

Entwickeln, Löten und Bestücken

€ 129,00

Big Beta-Reflow-Kit

Reflow-Controller

€ 129,00

LED Wechselblinker Bausatz

€ 6,00

Arduino Mega (ATMega 1280-16AU) kompatibel

€ 36,50

Tool-Kit Extended

€ 149,00

FUNK AMATEUR

Anwenderbericht
www.beta-estore.com/bericht

Video
www.beta-estore.com/video

eSTORE® ist eine eingetragene Marke der Beta LAYOUT GmbH

www.beta-eSTORE.com

Beta LAYOUT
create:electronics

Interesse an einer Anzeige?

info@embedded-projects.net

WEEng GmbH

SMD-Rework
Fine-Pitch / QFN / BGA

Prototypen / Kleinserien

SMD-Sample-Kits
Widerstände / Kondensatoren
0402 / 0603 / 0805 / usw.

SMD-Bestückung

www.weeng.net

→ firma.embedded-projects.net

DAS HARDWARE FOR YOUR PROJECTS-PORTAL

In unserem Online-Shop finden Sie eine große Auswahl verschiedenster Mikrocontrollerboards, Programmer, Debugger u.v.m.

→ shop.embedded-projects.net

Unser Büro in Augsburg besteht aus leidenschaftlichen Entwicklern. Sprechen Sie uns an, wir finden eine Lösung für Ihr Problem.

→ projekte.embedded-projects.net

Speziell für Studenten und Hochschulen, bieten wir diese Ausbildungsinitiative an. Mikrcontrollerboards für den kleinen Geldbeutel.

→ student.embedded-projects.net

Holzbachstraße 4, D-86152 Augsburg
Tel +49 (0) 821 279599-0
Fax +49 (0) 821 279599-20
info@embedded-projects.net

embedded projects GmbH
HARDWARE FOR PROJECTS

FIND www.f-y-e.de

your engineer

Der Experten-Wegweiser
zu Ihrem Elektronikentwickler

- Elektronik- / Softwareentwicklung
- Layout
- Mechatronik
- Bestücker / EMS-Dienstleister
- EMV-Dienstleister

Find-Your-Engineer ist ein persönliches Empfehlungsnetzwerk. Firmen die Elektronik-Experten suchen, wenden sich bitte direkt an:

Markus Kessler
kontakt@find-your-engineer.de

Mit der besten Empfehlung!

Sie denken bei ARCHITEKTUR nicht an Häuser?

Dann entwickeln Sie mit uns Embedded Systems & Software.

www.mixed-mode.de

MIXED MODE
technik.mensch.leidenschaft

embedded - projects.net
JOURNAL
OPEN SOURCE SOFT-AND HARDWARE PROJECTS

embedded projects GmbH
Holzbachstraße 4
D - 86152 Augsburg

Name / Firma

Straße / Hausnummer

PLZ / Ort

Email / Telefon / Fax

◯ Ich möchte jede zukünftige Ausgabe erhalten

◯ Wir möchten als Hochschule / Ausbildungsbetrieb jede weitere Ausgabe bekommen. Bitte gewünschte Anzahl der Hefte pro Ausgabe ankreuzen. 5 10

◯ Ich möchten im embedded projects Journal werben oder eine Stellenanzeige aufgeben. Bitte schicken Sie mir Infomaterial, Preisliste etc. zu.

Werdet aktiv!

Das Motto: Von der Community für die Community !

Das Magazin ist ein Open Source Projekt.

Falls Du Lust hast, Dich an der Zeitschrift durch einen Beitrag zu beteiligen, würden wir uns darüber sehr freuen. Schreibe deine Idee an:

journal@embedded-projects.net

Regelmäßig

Die Zeitschrift wird über mehrere Kanäle verteilt. Der erste Kanal ist der Download als PDF - Datei. Alle Ausgaben sind auf der Internetseite [1] verfügbar. Diejenigen, die lieber eine Papierversion erhalten möchten, können den zweiten Kanal wählen. Man kann sich dort auf einer Internetseite [2] in eine Liste für die gesponserten Abos oder ein Spendenabo eintragen. Beim Erscheinen einer neuen Ausgabe wird dank Sponsorengeldern an jeden auf der Liste eine Ausgabe des aktuellen Journal versendet. Falls man den Versandtermin verpasst hat, kann man das Heft auch über einen Online - Shop [2] beziehen.

[1] Internetseite (Anmeldeformular gesponserte Abos): http://journal.embedded-projects.net

[2] Online - Shop für Journal:
http://www.embedded-projects.net

Sponsoren gesucht!

Damit wir weiterhin diese Zeitschrift für jeden frei bereitstellen können, suchen wir dringend Sponsoren für Werbe- und Stellenanzeigen. Bei Interesse meldet Euch bitte unter folgender Telefonnummer: 0821 / 2795990 oder sendet eine E-Mail an die oben genannte Adresse.

Impressum

embedded projects GmbH
Holzbachstraße 4
D-86152 Augsburg
Telefon: +49(0)821 / 279599-0
Telefax: +49(0)821 / 279599-20

Veröffentlichung: 4x / Jahr
Ausgabenformat: PDF / Print
Auflagen Print: 2500 Stk.
Einzelverkaufspreis: 1 €

Layout / Satz: EP
Druck: flyeralarm GmbH
Titelbild: Edith Högl

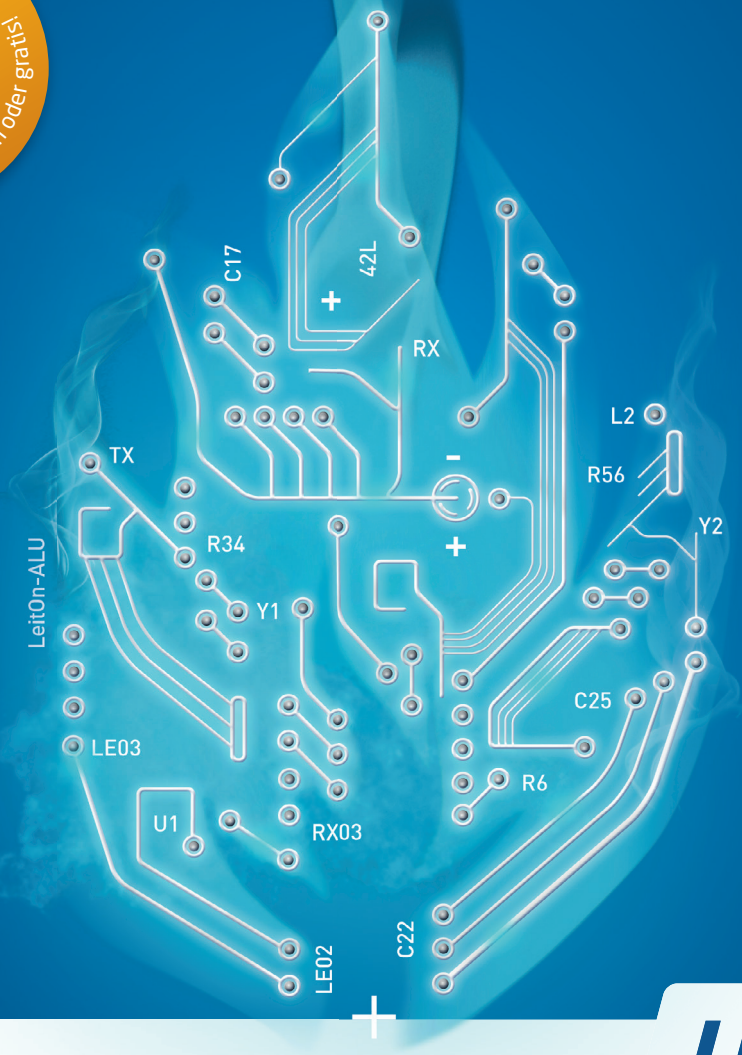
Alle Artikel in diesem Journal stehen unter der freien Creativ Commons Lizenz. Die Texte dürfen, wie bekannt von Open Source, modifiziert und in die eigene Arbeit mit aufgenommen werden. Die einzige Bedingung ist, dass der neue Text ebenfalls wieder unter der gleichen Lizenz, unter der dieses Heft steht veröffentlicht werden muss und zusätzlich auf den originalen Autor verwiesen werden muss. Ausgenommen Firmen- und Eigenwerbung.

Dies ist ein Open Source Projekt.

GERADE WENN'S MAL HEISS HERGEHT.

LEITERPLATTEN AUS ALUMINIUM ONLINE BESTELLEN.

ab 12h Leiterplatten-Express! Pünktlich oder gratis!



www.HTQN.de

LEITON
RECHNEN SIE MIT BESTEM SERVICE

Vorsicht, heiß! Starke Hitzeeinwirkung ist eine echte Herausforderung – besonders in der Hochleistungs-LED-Technik oder wenn wichtige Leistungsbauteile permanent hohen Temperaturen ausgesetzt sind. Aluminium-Platinen von LeitOn überzeugen durch **hohe Hitzeresistenz** sowie einen Wärmeleitwert von bis zu 3,0 W/mK in der Premiumvariante „Polytherm TC-Lam 3.0". Die vielen Vorteile liegen auf der Hand: **Kostenreduktion** durch höhere Lebensdauer und Zuverlässigkeit, **Platzersparnis** dank Integration der aufwendigen Kühlmechanismen sowie **Performancesteigerung** durch höhere Leistungsdichte der Anwendung. Sie möchten mehr darüber wissen? Wir bieten persönliche Beratung am Telefon und einen kompetenten Außendienst. Sie können bei LeitOn immer mit dem besten Service rechnen.

www.leiton.de | **kontakt@leiton.de** | Info-Hotline **+49 (0)30 701 73 49 0**