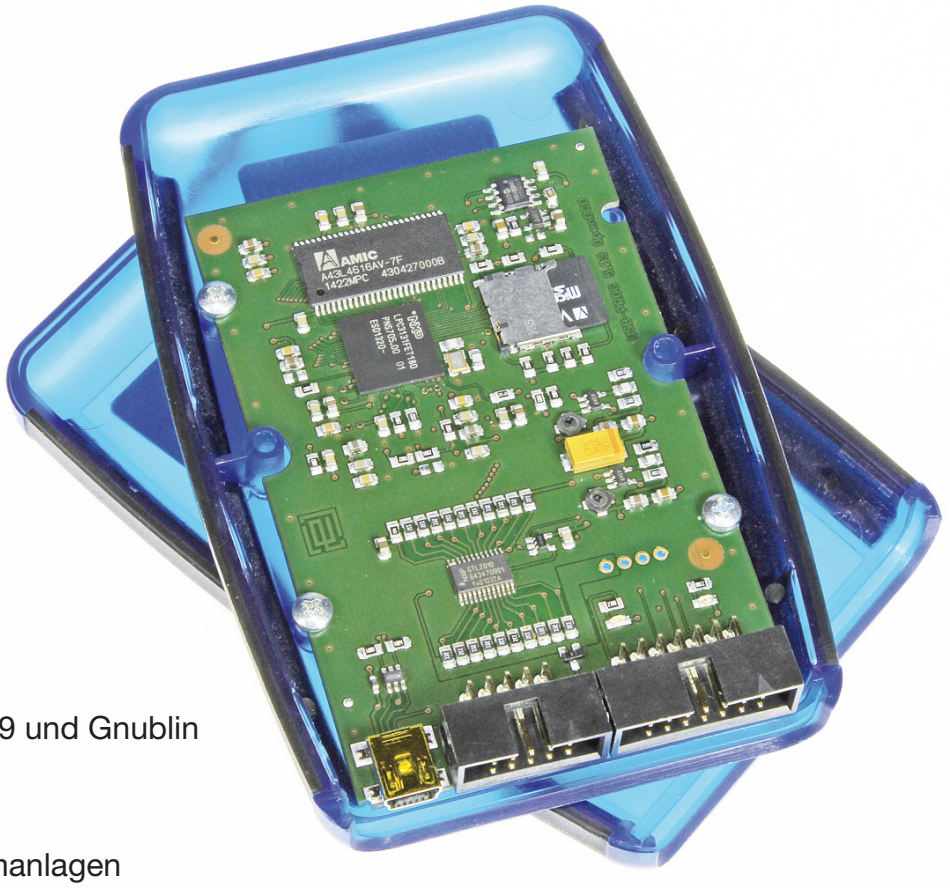
embedded projects JOURNAL
OPEN SOURCE SOFT-AND HARDWARE PROJECTS

Eine Open-Source Zeitschrift zum Mitmachen!

aufbruch in eine neue welt



[PROJECTS]

- Powerplant
- LED-Matrix mit IC MAX7219 und Gnublin
- WLAN-Router im Eigenbau
- USBprog 5.0
- Telefonwählmodul für Alarmanlagen

Einleitung

Ausgabe 04/2014

Embedded Projects Journal - Ausgabe No. 23

Einleitung

Danke mal wieder allen - an dieser Stelle - für die tollen Artikel!

Viele Spaß beim Lesen

Benedikt Sauter

und das embedded projects Team

P.S.: Wir können wieder spannende Artikel gebrauchen :o)
E-Mail an: sauter@embedded-projects.net

Werbeanzeigen
Embedded-Projects-
Journal

info@embedded-projects.net

EURO CIRCUITS

Die europäische Referenz für PCB Prototypen und Kleinserien

Tools zur Markteinführung Ihres Projekts - rechtzeitig und nach Budget

- Smartes Auftrags-Pooling senkt die Kosten
- Keine Einrichtkosten. Bestellung ab einer Leiterplatte.
- Sofortige Online-Bestellung - ohne bürokratische Verzögerungen
- Datenüberprüfung vor der Bestellung - keine Liefer-Verzögerungen
- Direkte Anzeige von DRC- und DFM-Fehlern - sofort lösbar
- Nutzen setzen und individuellen Text direkt am Bildschirm hinzufügen

Online Chat Support –
Unsere Experten zur Beantwortung Ihrer Fragen

PCB proto

spezieller Niedrigstpreis Protoypen-Service für Entwickler

- 1, 2 oder 5 Leiterplatten in 2, 3, 5 oder 7 Arbeitstagen
- 2 und 4 Lagen
- Vollständige Ausführung, elektrisch getestet

NEU

PCB PIXture ein grafischer Druck auf Ihrer Leiterplatte

Mehr Info in unser BLOG

STANDARD pool

Europas große Auswahl an Pooling-Optionen

- 1 - 16 Lagen. Wählen Sie aus über 700 definierten Multilayer Aufbaue
- Layout-Technologie bis 90µm
- Ab zwei Arbeitstagen

RF pool

Alle Pooling-Vorteile mit HF-Material

- Isola I-TERA und Rogers 4350B Material
- 2 - 4 Lagen, bis 100µm Technologie
- Ab 3 Arbeitstagen

IMS pool

Alu-Leiterplatten für (LED) Anwendungen mit hoher Wärmeableitung

- Einlagige Aluminiumkern-Leiterplatten
- Lötstopplack / Best.druck weiß / schwarz oder umgekehrt
- Ab 3 Arbeitstagen

BINDI pool

Indische Preise - Europäische Qualität

- 1 - 4 Lagen.
- Produziert in unserem neuen indischen Werk
- Gleiche Materialien, Technologien und Qualitätsstandards wie in unseren europäischen Werken.

www.eurocircuits.de

PCB Visualizer PCB Checker PCB Configurator

wawision.embedded-projects.net

waWision -
die Steuerzentrale für Ihre Firma

Verwal-tung	Plug & Play	Waren-eingang	Marke-ting
FiBu	Produk-tion	automa-tisches Lager	Online-Shops

EIN SYSTEM AUS EINER HAND

- keine Installation
- Betriebssystem unabhängig
- Standardhardware Plug & Play
- mitwachsend

DEMOVERSION

weitere Infos finden Sie auf unserer Internetseite

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

Powerplant - Optimierung der Gesamtenergiebilanz eines Hauses

Dominik Laton <dominik.laton@laton-project.de>

Sandra Boemmel <sandra.boemmel@hs-augsburg.de>

Im Moment nimmt der Anteil der erneuerbaren Energien stark zu. Da dieser Trend auch bei mir Zuhause Einzug gehalten hat und wir jetzt stolze Besitzer einer PV-Anlage sind, kam der Wunsch auf, die Effizienz des Gesamtsystems zu steigern. Wenn eine PV-Anlage (bis ca. 10kW) auf dem Hausdach betrieben wird, beträgt die Einspeisevergütung ca. 13 ct/kWh. Der Preis für den Bezug einer kWh vom Netzbetreiber liegt bei ca. 30 ct/kWh. Der Bezug einer kWh ist also deutlich teurer als die Vergütung einer kWh. Daher ist es wünschenswert, seine Verbraucher so weit wie möglich mit dem selbst erzeugten Strom zu betreiben.

Einleitung

Prinzipiell gibt es bereits einige Programme, um im Zusammenhang mit PV-Anlagen die Energien anzuzeigen. Meist stammen diese Programme von den Herstellern der Wechselrichter und sind somit nur in Zusammenhang mit den eigenen Produkten einsetzbar. Die Überwachung einer Anlage mit Wechselrichtern von verschiedenen Herstellern kann somit nicht über ein zentrales Programm, sondern nur mit verschiedenen herstellerspezifischen Programmen erfolgen. Durch die Spezialisierung auf Wechselrichter und nicht auf das Gesamtsystem, lässt sich in diesen Programmen auch nur die erzeugte Energie einsehen. Dadurch dass nur die erzeugte Energie geloggt wird und Informationen über die aktuell verbrauchte Energie fehlen, ergibt sich das Problem, dass eine Optimierung des Gesamtsystems unmöglich ist. Um diesem Problem zu entgehen, müssen die Daten dort erhoben werden, wo beide Informationen vorliegen, nämlich am Stromzähler.

Stromzähler ein PV-Anlage

Prinzipiell gelten viele Auflagen beim Einbau einer PV-Anlage, z.B. muss ein digitaler Stromzähler (EDL21) eingebaut werden. Dieser misst über drei Phasen die bezogene bzw. die eingespeiste Arbeit. Außerdem ermittelt der Zähler den momentanen Verbrauch und die „Richtung des Stromes“ (Bezug/Einspeisung).

Vereinzelt nutzen kommerzielle Lösungen diese Eigenschaften des Stromzählers und erheben die Daten über Strommesszangen. Diese Methode bringt jedoch große Ungenauigkeiten mit sich. Wesentlich weniger Produkte wählen das genauere Verfahren, die Daten direkt im Stromzähler zu ermitteln. Ein Beispiel für ein freies Projekt, welches diesen Weg beschreitet ist Volkszähler.

Leider ist dieses System für unseren Einsatzzweck zu groß und umfangreich. Die Suche nach einem Programm, das die Datenerhebung möglichst transparent durchführt, sehr klein ist, kaum Systemlast erzeugt und eine stark modulare Architektur aufweist war vergeblich. Somit entstand die Idee, ein auf unsere Bedürfnisse angepasstes Programm selbst zu entwickeln.

Daher habe ich vor ca. einem Jahr ein Testprojekt ins Leben gerufen, welches nun im Rahmen einer Vorlesung mit Gruppenprojekt optimiert und ausbaufähig gestaltet werden sollte.

Umsetzung

Das Schaubild (Abb. 1) stellt den schematischen Aufbau des Systems dar, welcher in diesem Projekt umgesetzt wurde.

Die Daten werden vom Stromzähler (EDL21) erhoben, im Embedded System (Cubieboard2) verarbeitet und über den Router im Internet zur Verfügung gestellt.

Der Stromzähler besitzt eine optische Schnittstelle, über die man mit einer Taschenlampe die verschiedenen Informationen abfragen kann. Außerdem ist in dieser Schnittstelle eine Infrarotdiode integriert, über welche per SML Protokoll (Smart Message Language) alle Informationen (aktuelle Leistung in W, Eingespeiste kWh, bezogene kWh) gestreamt werden. Die Daten werden alle ein bis zwei Sekunden aktualisiert.

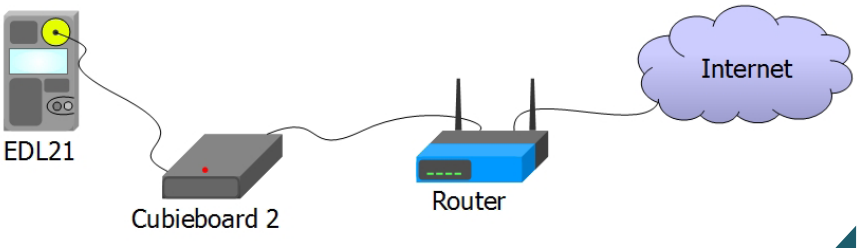


Abb. 1: Schematischer Systemaufbau

Da die Daten über eine Infrarotschnittstelle übertragen werden, benötigt man einen Adapter, der das Infrarotlicht in elektrische Signale umwandelt. Hierfür gibt es bereits Anleitungen im Netz, wie ein solcher gebaut werden kann. Nähere Informationen zum Adapter unter: Verwendete Hardware

Im ursprünglichen Testprojekt wurde als Embedded System ein RaspberryPi eingesetzt. Dieser verfügt von Haus aus nur über wenig Peripherie-Hardware für Bastler. In diesem Fall war eine UART-Schnittstelle notwendig, welche mit 3,3 V (TTL) Pegeln arbeiten kann. Da diese auf dem RaspberryPi leider nicht vorhanden

war, musste ein RS232 zu USB - Adapter (siehe: Verwendete Hardware) verwendet werden, welcher allerdings bei längerem Betrieb zu Problemen führte. Die Neuauflage dieses Projektes soll auf diesen Adapter verzichten können. Aufgrund dieser und anderer Anforderungen, ist die Entscheidung auf das Cubieboard 2 (siehe Verwendete Hardware) gefallen. Die Anwendung, welche auf diesem läuft, verarbeitet die SML-Pakete des Stromzählers, wertet sie aus, stellt die Daten auf einer Website zur Verfügung und loggt diese. Der tatsächliche Aufbau in einem Sicherungskasten ist in Abbildung 2 zu sehen.

Abb. 2: Der Aufbau im Sicherungskasten

Cubieboard 2

Im Auslieferungszustand ist auf dem Cubieboard 2 Android installiert. Dies ist zwar grafisch ganz nett und zum Beispiel für App-Entwickler interessant, jedoch für Embedded Entwicklungen sehr unbrauchbar. Daher ist es notwendig ein anderes Betriebssystem zu installieren.

Für dieses Projekt wird Cubian verwendet. Cubian ist ein angepasstes Debian, welches die Tools und Treiber von Linux-Sunxi bereits integriert hat.

Linux-Sunxi ist die freie Community des Chipherstellers Allwinner. Da auf dem Cubieboard 2 ein Allwinner A20 SoC verbaut ist, sind bei dieser Community alle relevanten Informationen und Tools zu finden.

Software

Die größte Einschränkung der Software ergibt sich durch den grundlegenden Aufbau eines Embedded Systems.

In diesen werden meist Flashspeicher als „Festplatte“ eingesetzt. Die Flashspeicher haben aufgrund ihrer Technik nur eine sehr begrenzte Anzahl an Lese-Schreib-Zyklen. Weswegen kann man nicht einfach jeden Datensatz in eine Datei schreiben kann, denn dadurch würde der Flashspeicher schnell defekt werden.

Deshalb ergibt sich als Randbedingung, dass alle Verarbeitungsschritte im Ram ausgeführt werden müssen.

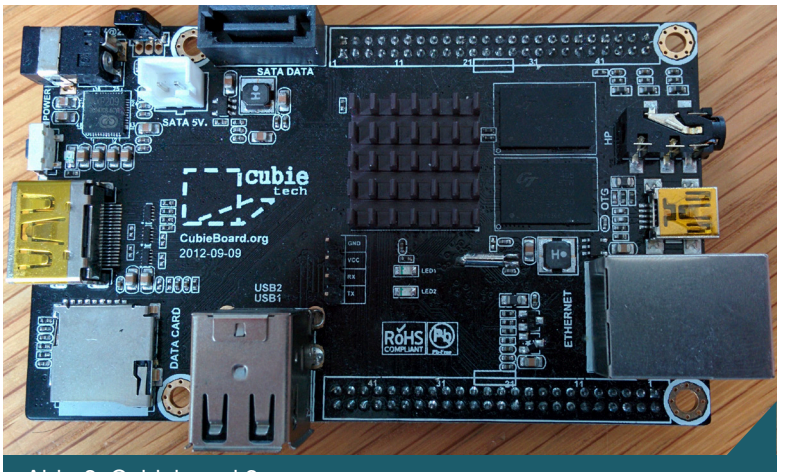


Abb. 3: Cubieboard 2

Da es sich beim Ram um einen flüchtigen Speicher handelt, was bedeutet, dass bei einem Stromausfall oder Powercycle alle Daten verloren gehen, müssen die ausgewerteten Ergebnisse gesammelt einmal am Tag in den Flashspeicher geschrieben werden.

Eine weitere Anforderung liefert die Bereitstellung einer Website. Im Normalfall werden einfache Websites statisch (eine Datei auf der Festplatte) angelegt. Dies hätte zur Folge, dass man wieder jeden Datensatz in den Flashspeicher schreiben müsste, was auf lange Sicht zur dessen Zerstörung führt. Deshalb wird hier wiederrum auf eine Technologie zurückgegriffen, welche es ermöglicht die Website nur im Ram zu halten und damit keine Schreibzugriffe auf den Flashspeicher erfolgen. Diese Technik nennt man Fastcgi. Bei Fastcgi kommunizieren der Webserver und die Applikation nur über Neztwerksockets und brauchen daher keine Dateien mehr. Eine Anfrage, die an den Webserver gestellt wird, wird an die Applikation weitergeleitet und diese generiert dann die entsprechende Website.

C++, Boost und Fcgi++

Durch die Verwendung von C++ kann man eine transparente Datenmodellierung schaffen. Im Vergleich zu C bietet C++ mit Hilfe seiner Templateprogrammierung die Möglichkeit sicherer und effizienter mit unterschiedlichen Datentypen zu interagieren. Durch den Einsatz von Boost kann eine Applikation mühelos plattformunabhängig gestaltet werden. Näheres hierzu unter Boost. Fcgi++ ist eine Bibliothek, die die Mechanismen die benötigt werden, um eine Fastcgi - Applikation zu schreiben gut in eigene Klassen kapselt und damit sehr lesbaren Programmcode ermöglicht. Näheres hierzu unter Fastcgi++.

Systemarchitektur

Die Architektur dieses Systems soll einen modularen Ansatz wiederspiegeln. Durch eine Trennung von Datenerhebung und Datenverarbeitung kann leicht ein anderes Interface oder eine andere Auswertung eingesetzt werden.

Eine Technik zur Unterstützung der Modularität stellt die Unterteilung der Teilaufgaben in einzelne Threads dar. Somit ist es möglich jeweils ein eigenes Timing einzustellen. Aus diesem Grund kann einfach die Systemlast reduziert werden. Ein weiterer Vorteil der Threads ist, dass man neue Teilaufgaben leicht hinzufügen oder herausnehmen kann. Im Schaubild ist die Unterteilung in Threads an den mit Thread 1/2/3 beschrifteten Trennlinien erkennbar.

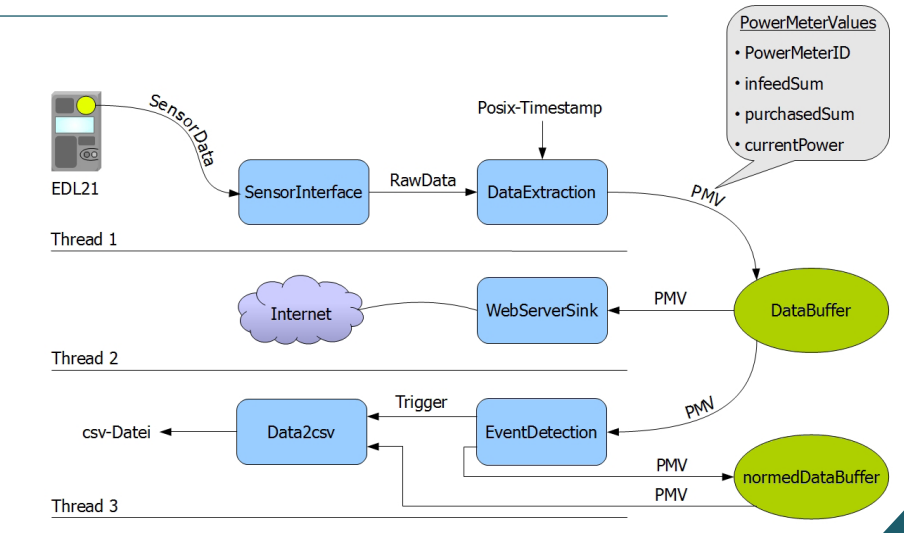


Abb. 4: Blockschaltbild des kompletten Hard- und Softwareaufbau

Modulbeschreibungen

PowerMeterValues: Die PowerMeterValues stellen eine Datenstruktur dar, die einen kompletten Datensatz enthält. Dieser besteht aus der PowerMeterID (ID des Stromzählers), infeedSum (Menge an eingespeister Energie in kWh), purchasedSum (Menge an bezogener Energie in kWh) und currentPower (aktuelle Leistung in W mit Vorzeichen "-" für Einspeisung / "+" für Bezug).

DataBuffer: Der DataBuffer ist ein Ringspeicher, welcher im Heap liegt. Dieser kann beliebig viele lesende Threads und einen schreibenden Thread bedienen. Er kann jegliche Daten beherbergen, wie z.B. eine Klasse oder in diesem Fall die PMVs. Zudem wird zu jedem Datensatz ein Zeitstempel abgelegt. Im Schaubild beinhaltet der DataBuffer also alle empfangenen PMVs mit entsprechendem Zeitstempel.

normedDataBuffer: Der normedDataBuffer ist prinzipiell der gleiche Ringspeicher wie der DataBuffer, mit dem Unterschied, dass in ihm die „ausgedünnten", also nur relevanten PMVs abgelegt werden. Diese haben den gleichen Zeitstempel, wie die, die im DataBuffer eingelagert wurden.

SensorInterface: Das SensorInterface dient als Abstraktionsschicht der Hardware. Es spricht die UART-Hardware an und legt das empfangene SML-Datenpaket in den Heap. Anschließend startet es das DataExtraction Modul, welches die Auswertung des Datenpaketes vornimmt. Wichtig ist hierbei, dass das DataExtraction Modul kein Wissen über die Erfassung des Datenpaketes haben muss, da dieses Wissen im SensorInterface Modul liegt. Deswegen besteht jederzeit die Möglichkeit durch eine Anpassung des SensorInterface Moduls die Art der Datenübertragung zu ändern, ohne dass das gesamte Programm geändert werden muss. Die Schnittstelle zum DataExtraktion Modul ist deshalb auch sehr einfach gehalten. Sie enthält nur einen Pointer auf den Anfang des Datenpaketes im Speicher und dessen Länge.

DataExtraction: Das DataExtraction Modul wertet ein SML-Paket aus und trägt die relevanten Werte in ein PMV ein. Anschließend wird das PMV mit dem Zeitstempel der Auswertung der Daten in den DataBuffer eingelagert.

WebserverSink: Die WebserverSink beherbergt die Fastcgi Implementierung. In diesem Modul wird das zuletzt eingelieferte PMV aus dem Speicher gelesen und dem Webserver bereitgestellt, sofern dieser eine Anfrage vorliegen hat. Wenn keine Anfrage ansteht, schläft dieser Thread.

EventDetection: Die EventDetection ist ein Modul, welches eine Reduzierung der Datenmenge an PMVs vornimmt. Da ca. jede Sekunde ein PMV erzeugt wird, wären das pro Tag ca. 86400 PMVs. Effektiv sind allerdings nur ca. 2000 davon interessant, da z.B. in der Nacht der Stromverbrauch nicht mehr als 10W schwankt. Das bedeutet es werden viele PMVs mit nahezu gleichem Inhalt abgespeichert. Um diesen „unnützen" PMVs entgegenzuwirken, werden nur Änderungen des Stromverbrauchs, die größer als 200W sind, in den normedDataBuffer übernommen. Solche Anstiege werden hier Events genannt, weswegen das Modul auch EventDetection heißt.

Eine weitere Aufgabe des Moduls stellt das Starten des Dumpingmoduls Data2csv dar. Da man nicht jedesmal wenn ein PMV als Event abgelegt wird auf die „Festplatte" schreiben will, wird einmal am Tag (23:59 Uhr), das Data2csv Modul gestartet und der komplette Inhalt des normedDataBuffers in eine Datei geschrieben.

Data2csv: Data2csv wird einmal am Tag durch das EventDetection Modul gestartet und speichert die PMVs, welche sich im normedDataBuffer befinden in eine Datei. Dabei werden die Informationen folgendermaßen abgelegt:

Zeitstempel;PowerMeterID;currentPower;purchasedSum;infeedSum;

Ein PMV entspricht genau einer Zeile in der .csv Datei. Der Dateinamen der Datei entspricht dem ISO Zeitstempel des Zeitpunktes der Erzeugung.

Links & Referenzen

[1] Link zur Projektseite: http://laton-projects.org/

[2] Aktuelle Daten des Powerplaten: http://powerplant.no-ip.org/

LED-Matrix mit IC MAX7219 und Gnublin

Rafael Radtke <rafael.radtke@hs-augsburg.de>

Einleitung

Der folgende Bericht entstand im Rahmen des Wahlfaches Embedded Linux an der Hochschule Augsburg. Ziel dieses Projekts war es eine 5x7 LED-Matrix über ein IC MAX7219 bedienen zu können. Hierfür sollte das Gnublin Board Standard und die SPI Schnittstelle zum ansteuern des IC's hergenommen werden.

IC MAX7219

Die Kommunikation des MAX7219 mit dem Gnublin Board erfolgt über drei Portleitungen: DIN (1), CLK (13) und LOAD (12), siehe Abbildung 1. Wie sich unschwer erkennen lässt, bietet sich hier eine Kommunikation über die SPI Schnittstelle an. Über die Ausgänge *Digits* (DIG0 bis DIG7) und *Segment* (SEG A bis SEG G und SEG DP) wird die LED-Matrix verbunden. Die Digits werden dabei mit den Spalten und die Segmente mit den Zeilen der LED-Matrix verbunden.

Am Ausgang DOUT (24) kann der Eingang DIN (1) eines weiteren MAX7219 angeschlossen werden. Auf diese Weise können mehrere MAX7219 kaskadiert werden und somit mehrere Matrizen gesteuert werden. 19 ist der Spannungseingang. Ausgänge 9 und 4 müssen auf GND gesetzt werden.

Mit Eingang 18 lässt sich die maximale Helligkeit der Anzeige bestimmen. Dabei soll ein Widerstand zwischen 18 und 19 angeschlossen werden.

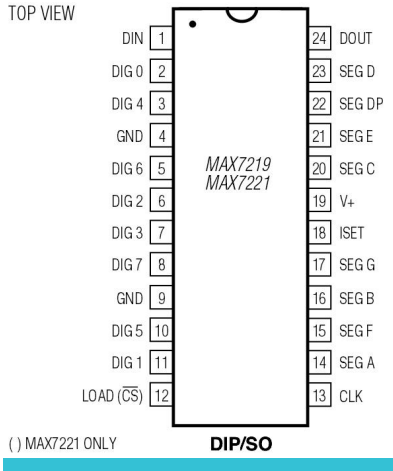


Abb. 1: Pin Description MAX7219

Signalmodelierung

Um den Max7219 steuern zu können muss das Signal wie in Abbildung 2 aufgebaut sein. In den IC werden die seriellen Daten an DIN bei jeder steigenden Signalflanke an CLK in ein internes 16-bit Schieberegister abgelegt. Dies geschieht unabhängig von Ereignissen, die auf LOAD auftreten.

Die Daten im Schieberegister werden anschließend mit einer steigenden Signalflanke an LOAD in ein Kontrollregister (Intensity, Decode Mode,...) oder ein Digitregister (Digit 0, Digit 1,...) geschrieben. LOAD muss unmittelbar nach 16 Takten an CLK auf „high" gehen, da sich nach 16 Takten ein Datenpaket vollständig im 16-bit Schieberegister befindet. Wird LOAD zu spät auf „high" gesetzt, gehen Daten verloren. Die Daten, die sich im Schieberegister befinden, tauchen 16,5 Taktzyklen später am Ausgangspin DOUT des MAX7219 auf. Die Daten werden mit der fallenden Flanke von CLK ausgegeben.

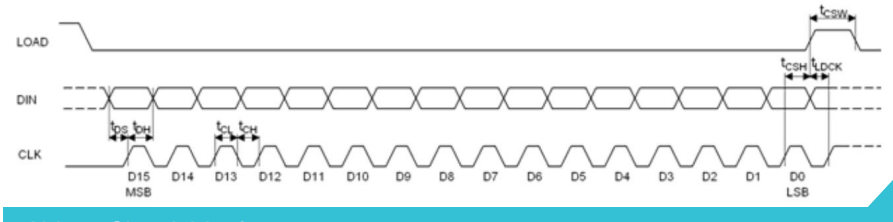


Abb. 2: Signalablauf

Register Belegung des MAX7219

Die Ansteuerung des MAX7219 erfolgt mit 16 Bit Paketen. Aufgebaut sind sie nach folgendem Schema. D12 bis D15 sind Dummy Bits und können beliebig beschrieben werden.

Über die Adresse werden die jeweiligen Register angesprochen. Ein genaue Auflistung liegt in der unteren Tabelle vor.

Register No-Op / Adresse 0x00: Dieses Register wird gebraucht, wenn mehrere MAX7219 kaskadiert werden.

Register Digit 0 bis Digit 7 / Adressen 0x01 bis 0x08: Das sind die Register für die jeweiligen Digits. Um eine Bestimmte LED zum leuchten zubringen muss erst das Digit gewählt werden und anschließend die Adresse des Segments.

Wenn man also z.B die erste und die dritte LED in der ersten Reihe zum leuchten bringen will muss man 0x0105 übertragen. Dabei setzen sich die letzten zwei Bit aus 1+4 zusammen. 1 für die erste LED und 4 für die dritte. Will man alle LEDs einer Reihe der ersten Spalte so muss man 0x017F übertragen.

REGISTER	ADDRESS					HEX CODE
	D15–D12	D11	D10	D9	D8	
No-Op	X	0	0	0	0	0xX0
Digit 0	X	0	0	0	1	0xX1
Digit 1	X	0	0	1	0	0xX2
Digit 2	X	0	0	1	1	0xX3
Digit 3	X	0	1	0	0	0xX4
Digit 4	X	0	1	0	1	0xX5
Digit 5	X	0	1	1	0	0xX6
Digit 6	X	0	1	1	1	0xX7
Digit 7	X	1	0	0	0	0xX8
Decode Mode	X	1	0	0	1	0xX9
Intensity	X	1	0	1	0	0xXA
Scan Limit	X	1	0	1	1	0xXB
Shutdown	X	1	1	0	0	0xXC
Display Test	X	1	1	1	1	0xXF

Abb. 3: Registeradressen

(1+2+4+8+16+32+64=127 => 0x7F)

Register Decode Mode / Adresse 0x09: Für die Ansteuerung einer LED-Matrix sind die Datenbits (D0 bis D7) dieses Registers mit 0 zu laden. (gemäß Abb. 4)

DECODE MODE	REGISTER DATA								HEX CODE
	D7	D6	D5	D4	D3	D2	D1	D0	
No decode for digits 7–0	0	0	0	0	0	0	0	0	0x00
Code B decode for digit 0 No decode for digits 7–1	0	0	0	0	0	0	0	1	0x01
Code B decode for digits 3–0 No decode for digits 7–4	0	0	0	0	1	1	1	1	0x0F
Code B decode for digits 7–0	1	1	1	1	1	1	1	1	0xFF

Abb. 4: Decode Mode Register

Register Intensity / Adresse 0x0A: Dieses Register steuert die Helligkeit der Anzeige. Die geringste Helligkeit kann mit 0x00, die höchste Helligkeit mit 0x0F gewählt werden. Der MAX7219 betreibt intern eine Pulsweitenmodulation, die die Helligkeit entsprechend dem gewählten Wert ändert.

Register Scan Limit / Adresse 0x0B: Mit diesem Register wird dem MAX7219 mitgeteilt wie viele Spalten die angeschlossene LED-Matrix besitzt. Diese Angabe ist notwendig, damit der MAX7219 nicht unnötig Daten an die nicht vorhandenen Spalten ausgibt. Eine falsche Einstellung kann die Helligkeit der Anzeige beeinträchtigen.

Register Shutdown / Adresse 0x0C: Mit diesem Register kann der MAX7219 ausgeschaltet werden. Das bedeutet, dass keine angeschlossenen Segmente leuchten. Dazu muss nur das Bit D0 im Register 0x0C gelöscht werden.

Register Display Test / Adresse 0x0F: Mit diesem Register kann ein Selbsttest des MAX7219 erfolgen, d.h. alle Segmente leuchten mit der maximalen Helligkeit. Dazu muss nur das Bit D0 im Register 0x0F gesetzt werden.

SPI-Schnittstelle

Der SPI Treiber ist im Kernel fest mit einkompiliert. Wenn der Treiber funktioniert, dann sollte

ein spidev Attribut unter /sys/class/spidev zu finden sein:

```
modprobe spidev

ls /sys/class/spidev/

spidev0.11
```

Hier sieht man z.B. das spidev angelegt (Mit Busnummer = 0 und Chipselect = 11) worden ist und der Treiber funktioniert. Es wird auch eine Gerätedatei unter /dev/spidev0.11 angelegt.

Mit folgenden Befehl lässt sich der Treiber schnell testen:

```
echo 100 > /dev/spidev0.11
```

Die SPI Pinbelegung ist wie folgt aufgebaut:

- 3 - GPIO11 Chipselect
- 5 - SCK Taktleitung für SPI
- 7 - MISO Master in Slave out
- 8 - MOSI Master out Slave in

Wobei Pin 3 (GPIO11) oder Pin 4 (GPIO14) als Chipselect hergenommen werden können. Da wir keine Antwort vom IC erwarten, können wir Pin 8 (MOSI) in diesem Projekt vernachlässigen. Die genaue Position der Pins kann man in der Schnittstellenübersicht nachschauen (Abb. 5).

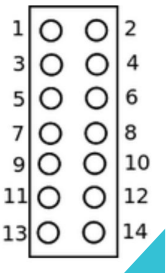


Abb. 5: SPI-Pins

Die 5x7 LED-Matrix

Dieser Abschnitt beschäigt sich mit der LED Matrix und deren Vernetzung. Die Genaue Bezeichnung ist „KINGBRIGHT TBC20-11EGWA“. Mit der LED Matrix von KINGBRIGHT hat man die Wahl zwischen zwei verschieden LED Farben. In diesem Projekt werden die Roten LED's verwendet. Im unteren Bild ist das Layout der Matrix zu sehen. (Abb. 6)

Anhand des Layouts kann der Netzplan zwischen Matrix und IC erstellt werden. Dabei werden die Digit Leitungen mit den Kathoden (siehe Abbildung 16: C1 bis C5) und die Segmente mit den Anoden (R1 bis R7) verbunden. Beachtet werden sollte hier, dass von C1 bis C5 jeweils die zweite Pinbelegung verwendet wird. Ansonsten bekommt man Grüne Spalten.

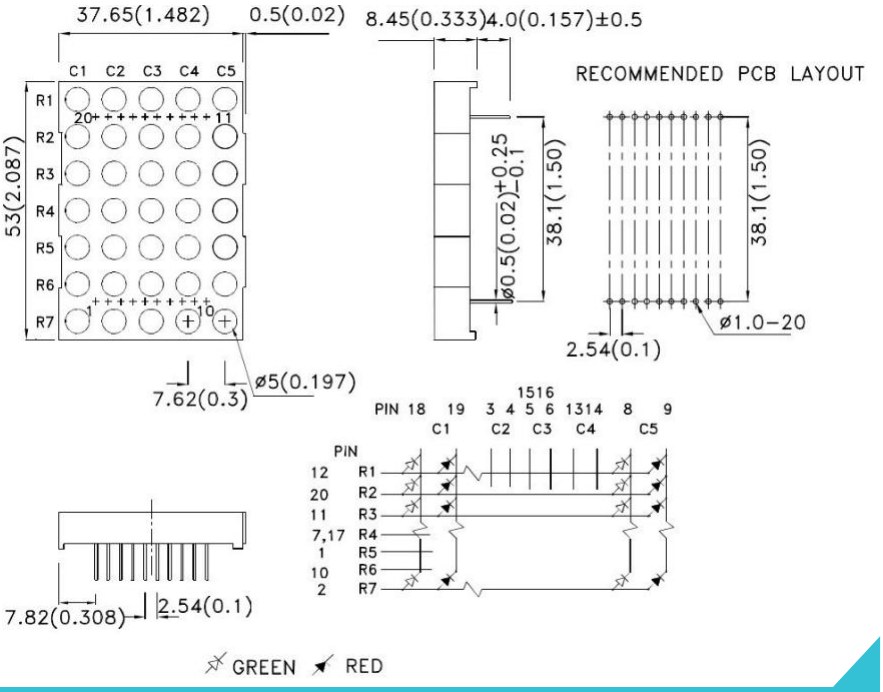


Abb. 6: LED Matrix Layout

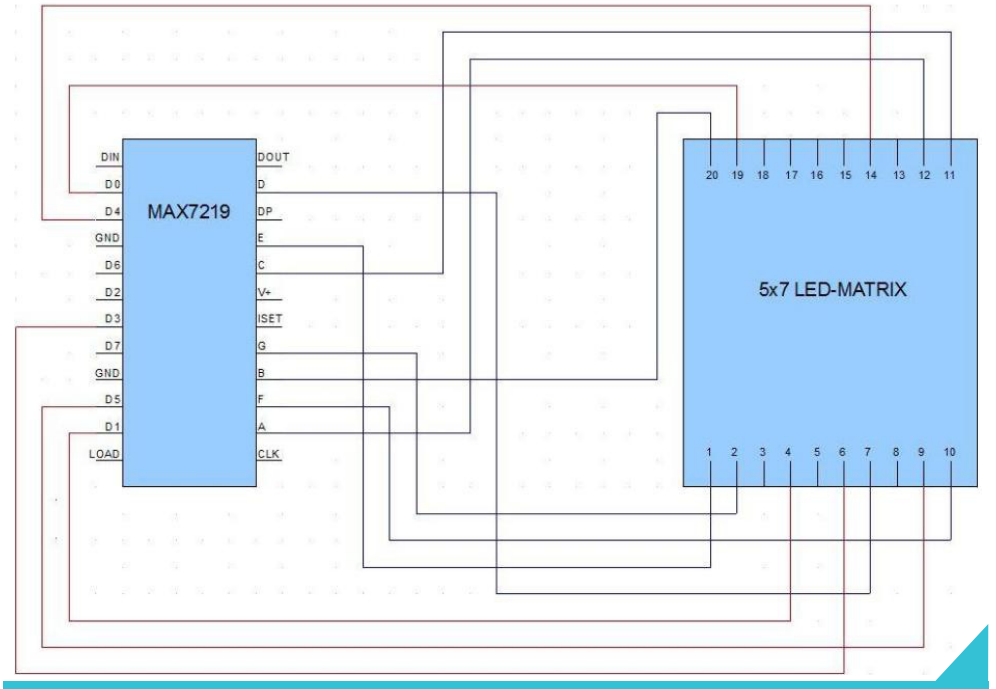


Abb. 7: Netzplan zwischen MAX7219 und der 5x7 LED-Matrix

Umsetzung

Vor dem löten sollte man auf einem Klemmbrett einige teile der Schaltung aufbauen um mögliche Fehler zu beseitigen. Als Spannungsquelle für die Schaltung wurde Der 3,3 V Ausgang vom Gnublin Board hergenommen.

Die Platine Wurde nach folgender Prinzipieller Schaltung Gelötet. Dabei wurden die Verbindungen zwischen IC und Matrix nach Abbildung 17 vernetzt. Für Rset sollte ein Geeigneter Widerstand eingesetzt werden, da es erfahrungsgemäß ohne, gelegentlich zu einem Flackern der Anzeige führt. Empfohlen wird 10 kOhm.

Software

Um das ganze zum laufen zu bringen, brauchen wir einen geeigneten Quellcode um den IC anzusteuern. Auf der Gnublin Homepage findet man die API „gnublin.h“ die einem das Leben erleichtert. Sie beinhaltet Klassen mit denen man die einzelnen Schnittstellen bedienen kann.

Bevor wir mit den eigentlichen Teil vom Code beginnen, müssen ein paar Einstellungen für die SPI Schnittstelle getroen werden.

- Mit spi.setSpeed wird der Takt auf 10 MHz gesetzt.
- Mit spi.setLSB(0) wird das Most Signifiant Bit zuerst gesendet.
- spi.setLength gibt die Wortlänge an. Diese soll auf 8 gesetzt werden, da unser Signal aus einem Address und einem Data teil besteht, die jeweils eine 8 Bit Länge haben.

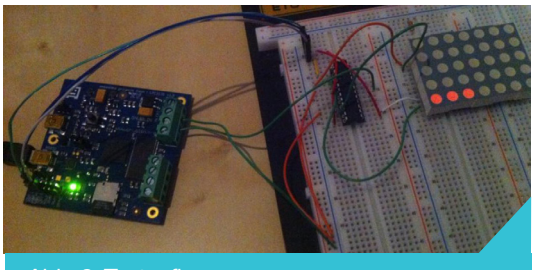


Abb. 8: Testaufbau

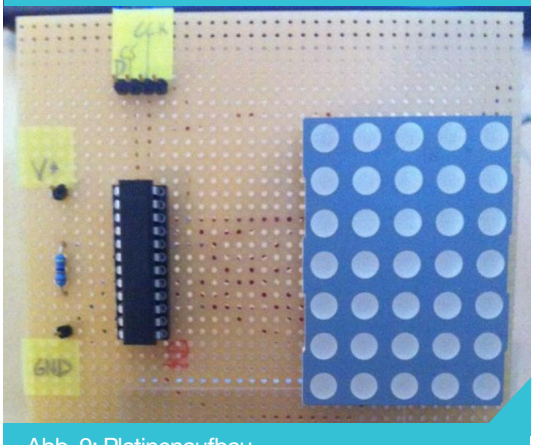


Abb. 9: Platinenaufbau

- gpio.pinMode(14,OUTPUT) initialisiert den Pin 14 als GPIO Ausgang.

Setzt man nach der Initialisierung den Display-Test auf 0x01 so aktiviert man alle LED’S.

Links & Referenzen

[1] Projektarbeit: https://elk.informatik.fh-augsburg.de/dav/elinux-14-berichte/08_Radtke/Projektarbeit.pdf

WLAN-Router im Eigenbau

Andreas Scheja <andreas.scheja@hs-augsburg.de>

Einleitung

Ein Embedded Board soll in meinem privaten Haushalt als WLAN-Router dienen und unter anderem folgende Dienste bieten:

- DHCP/DNS-Server + NAT für lokale Geräte
- W-LAN-Accesspoint
- Netzwerkbrücke zwischen LAN und WLAN
- Webserver als Statusanzeige (u.U. auch zur Teilkonfiguration, nur aus dem LAN erreichbar)
- LED-Display mit kleiner Statusanzeige und einige Buttons für die Steuerung dieses Displays
- Bindung der wechselnden IP-Adresse an eine Domain mittels eines DDNS-Services
- SSH-Server zur Administration, von außen erreichbar
- Schutz des Systems durch geeignete Paketfiltereinstellungen als Firewall
- Zugang zum Internet durch ein externes DSL-Modem

Außerdem wurde noch zusätzliche Hardware verwendet:

- DSL-Modem mit PPPoE-Ansteuerung
- Switch, um mehr als 1 Gerät über LAN anschließen zu können
- LCD-Display, Ansteuerung über UART
- Diverse elektronische Bauteile

Alle selbst geschriebenen Programme sollen in Python programmiert werden, die Weboberfläche wird durch PHP dynamisch gestaltet. Als Display wird ein LCD-Display der Marke Iteadstudio verwendet, das mit relativ einfachen Befehlen über eine serielle Schnittstelle gesteuert werden kann.

Das Board

Anforderungen an das verwendete Board:

- 2 Netzwerkports
- 1 W-LAN Karte
- 1 freier UART-Port, zumindest RX/TX, Spannung 3.3V
- 3.3 V Ausgang
- 5 V Ausgang
- 4 GPIO-Ports, als Interrupt-Quellen nutzbar
- 1 GPIO-Port, als Ausgang nutzbar

Das Olimex A13-Olinuxino-Wifi: Da ich bereits in Besitz eines A13-Olinuxino-Wifi war, hat sich das Board von Olimex angeboten, da es die benötigten Schnittstellen größtenteils aufwies und genug USB-Ports für 2 USB-To-Ethernet Adapter zur Verfügung standen. Außerdem ist bei der Wifi-Variante des Boards bereits ein W-LAN USB Adapter fest auf dem Board integriert. Die Leistungsaufnahme des Boards schwankt je nach Last und angeschlossener Hardware zwischen zwei und sechs Watt. Weitere Highlights des Boards sind ein Arm Cortex-A8 Prozessor mit 1Ghz Takt, 512 MByte RAM und ein integriertem 4 Gbyte NAND-Flash. Der Preis des Boards in dieser Ausführung liegt dabei bei 55€. Die Community um das Board ist unter dem Namen Sunxi aktiv.

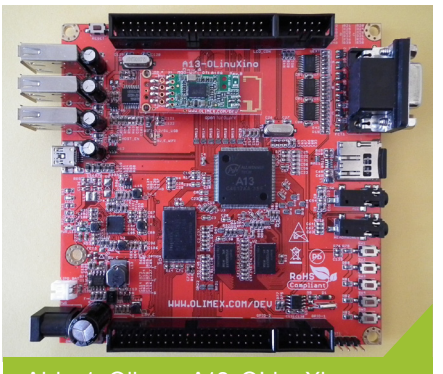


Abb. 1: Olimex A13-OLinuXino

Lötarbeiten

Ursprünglich geplant war nur eine Platine, die mit 4 Buttons bestückt werden sollte um dem Benutzer eine direkte Steuerungsmöglichkeit für das Display zu geben. Allerdings tauchte dann relativ bald ein Problem auf, das noch eine zweite Platine erforderte: Da das Display über UART angesteuert wird und auf die gesendeten Befehle Antworten zurücksendet, springt das Board beim neu starten durch den vom Board gesendeten Boot-Up Informationen in den manuellen U-Boot Modus und bleibt dort hängen. Im nachhinein gesehen hätte man beide Platinen auch auf eine Platine zusammenlöten können.

Der verwendete 10-Pin Blockstecker in nachfolgender Schaltung erklärt sich dadurch das der GPIO-1 Connector der A13-Olinuxino-Wifi Boards verwendet werden sollte, jedoch stellte sich im Nachhinein heraus, dass die GPIO-Pins dieses Connectors mit der Power-Managment-Unit verbunden sind und sich mit den aktuellen Treibern nicht korrekt ansteuern lassen.

Da das Display eine Spannung von 5V benötigt, die GPIO-Pins des Boards aber nur 3.3V ausgeben musste eine Schaltung her, mit der sich per GPIO-Signal 5V schalten lassen. Die benötigten Bauteile

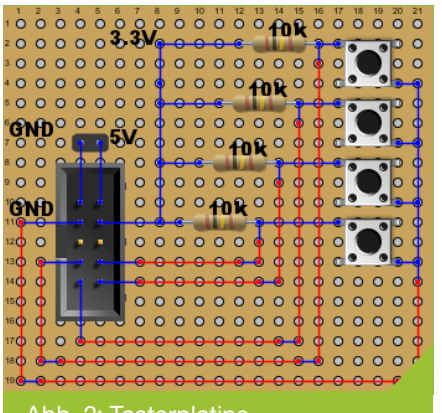


Abb. 2: Tasterplatine

für diese Schalten sind 3 10k Widerstände und je ein NPN und PNP Transistor. In nachstehender Abbildung ist der linke Transistor der NPN-, der rechte demzufolge der PNP-Transistor.

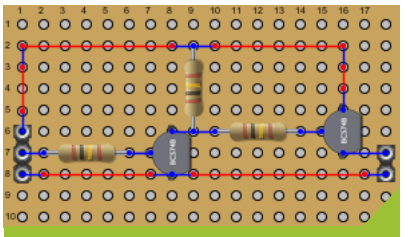


Abb. 3: Anschalter Platine

Das Betriebssystem

Kernel & Rootfilesystem: Als Kernel wird die an das Board angepasste Version 3.4 (Sunxi-Linux-3.4) verwendet. Gerade bei der Kompilierung bzw. Konfiguration des Kernels ist aber darauf zu achten, dass einige Kernelmodule benötigt werden, die zumindest in der Standardkonfiguration des Boardkernels nicht aktiviert sind. Dazu zählen unter anderem das PPP- und Netfiltermodul. Als Basis für das Rootfilesystem wird Debian 7.5 (wheezy) verwendet.

Benötigte Zusatzprogramme: Grundsätzlich gibt es hierbei meist mehrere Möglichkeiten, die sind dann im Funktions- und Konfigurationsumfang unterscheiden.

- Access Point Software: hostapd (1.0.3)
- DHCP/DNS-Server: dnsmasq (2.6.2)
- Webserver: apache2 + apache2-utils (2.2.22)
- Web-Skriptsprache: php5 (5.4.4)
- Datenbank: sqlite3 (3.7.13)
- Programmierung: python3 (3.2.3), python3-serial (2.5)
- Internetverbindung: pppoe (3.8), pppoeconf (1.20)

Konfiguration

Hier gehe ich etwas auf die Konfiguration der einzelnen installierten Softwarepakete ein.

Apache2 & PHP5: Bereits im Auslieferungszustand sind beide Programme größtenteils entsprechend vorkonfiguriert. Da der Webserver jedoch nur aus dem lokalen Netz erreichbar sein soll, muss die *Listen*-Direktive des Apache auf das Subnet *192.168.0.0/16* eingeschränkt werden. Außerdem soll der Zugang zur Weboberfläche mit einem Passwort gesichert sein. Dazu wird das Apache-Modul *mod_auth_digest* verwendet, wofür in der entsprechenden Host-Datei von Apache folgender Eintrag hinzugefügt werden muss (Listing 1).

Anmerkung: Anscheinend wird bei Apache Version 2.2 statt der *AuthDigestFile*-Direktive die *AuthUserFile*-Direktive verwendet. Die Datei *'/etc/apache2/auth'* wird mit dem Befehl *'htdigest -c /etc/apache2/auth router-conf USERNAME'* nach Eingabe eines Zugangspasswortes angelegt. Außerdem muss noch mit dem Befehl *'a2enmod auth_digest'* das entsprechende Modul aktiviert werden. Um die neue Konfiguration zu übernehmen muss Apache mit dem Befehl *'service apache2 restart'* neu gestartet werden.

SQLite3: Zweck einer Datenbank in dieser Anwendung soll vor allem die einfache Anzeige und Aktualisierung der Paketfilterregeln über die Weboberfläche sein.

Die kleine Datenbank SQLite3 zeichnet sich dadurch aus das praktisch keinerlei Einstellungen von nöten sind. Datenbanken werden dabei als einfache Dateien an beliebiger Stelle im Dateisystem angelegt, dabei ist jedoch zu beachten das wenn ein Prozess schreibend auf die Datenbank zugreifen will, der Benutzer dieses Prozesses Schreibrechte auf das Verzeichnis, in dem die Datenbankdatei liegt haben muss, da SQLite während des Schreibvorgangs temporäre Dateien anlegt. Da der Webserver (und PHP) unter der User-Id *www-data* laufen wird ein Verzeichnis */var/db* angelegt, in dem die Datenbankdatei abgelegt wird.

```
<Location />
  AuthType Digest
  AuthName „router-conf"
  AuthDigestDomain / http://localhost/
  AuthDigestFile /etc/apache2/auth
  Require valid-user
</Location>
```

Listing 1: Eintrag in der Host-Datei

pppoeconf: Das Programm *pppoeconf* sucht automatisch nach Modems auf allen Netzwerkinterfaces und leitet daraufhin einen Konfigurationsassistenten ein, der die wichtigsten Einstellungen automatisch vornimmt. Außerdem erzeugt pppoeconf bei Bedarf auch einen Eintrag in der *interfaces*-Datei, um die Verbindung zum Internet herzustellen, sobald der Rechner eingeschaltet wird. Es sollte jedoch darauf geachtet werden das das Modem während des Konfigurationsvorgangs eingeschaltet und mit der Telefonleitung verbunden ist, da erzeugte Konfiguration sonst nicht funktioniert.

hostapd: Zuerst einmal muss überprüft werden, ob die verwendete WLAN-Hardware überhaupt fähig ist, einen Access Point zu öffnen. Dazu muss der Befehl *"iw list"* ausgeführt werden. Sind im Abschnitt *supported interface modes:* der Ausgabe die Punkte *AP* und *AP/VLAN* zu finden, so ist die Hardware für den Gebrauch mit hostapd verwendbar.

Bei der Konfiguration gibt es sehr viel Möglichkeiten, deshalb hier nur die wirklich Essentiellen Punkte:

- *interface=wlan0*: Interface, das als Access Point verwendet werden soll
- *driver=nl80211*: Verwendeter Treiber
- *ssid=mein-wlan*: Die SSID des WLAN-Netzwerks
- *channel=3*: Der zu verwendende Funkkanal
- *hw_mode=g*: Übertragungsmodus
- *wpa=2*: Nur WPA2-Verschlüsselung erlauben
- *wpa_key_mgmt=WPA-PSK*: Art des Schlüssels
- *wpa_passphrase=1234567890*: Zugangsschlüssel

dnsmasq: Auch hier sind wieder mehr Einstellmöglichkeiten gegeben, als sich zählen lassen, also wieder einige Beispiele:

- *interface=eth0*: Beschränken des DHCP-Dienstes auf interne Netzwerkschnittstelle
- *dhcp-range=192.168.1.20,192.168.1.50,12h*: Der per DHCP zu vergebende Addressbereich und die Lease-Zeit

interfaces & iptables: Konfiguration der Interfaces und iptables werden wie folgt in Listing 2 vorgenommen.

24h-Disconnect & DynDNS: Da bei den meisten privaten Internetanbietern alle 24 Studen eine Zwangstrennung ansteht, bietet es sich an den Zeitpunkt der Trennung als einen cron-Job zu erledigen. Dazu wird in der Datei */etc/crontab* folgender Eintrag angehängt, damit jeden Tag um 6 Uhr die Internetverbindung neu aufgebaut wird:

```
0 6 1-31 * * root poff && pon dsl-provider
```

Da man bei jeder Trennung eine neue IP vom ISP zugewiesen bekommt, muss man diese Änderung seinem DynDNS Provider berichten, um weiterhin über die jeweilige URL verfügbar zu sein. Die Möglichkeiten, sind je nach Anbieter unterschiedlich. In diesem Beispiel verwende ich den kostenlosen Service von *changeip.com*. Neben einigen Vorgefertigen Programmen gibt es auch eine manuelle Methode, in der man eine http(s)-Verbindung zum Server *nic.changeip.com* aufbaut und per

```
GET /nic/update?u=USER&p=PW‘
```

den Benutzernamen und das Passwort übermittelt. Es ist allerdings sehr zu empfehlen https zu verwenden, da das Passwort sonst völlig ungeschützt übertragen wird. Außerdem ist zu empfehlen, die Benutzername und Passwort Base64 kodiert als HTTP-Basic Authentification zu versenden, da die angeforderte entschlüsselte URL samt Benutzername und Passwort sonst im Klartext in den Server-Log-Files auftauchen. Um nun bei jedem neuen Verbindungsaufbau den entsprechenden Befehl auszuführen, wird eine neue Datei *'/etc/ppp/if-up.d/5dyndns'* angelegt (siehe Listing 3).

```
auto lo
    iface lo inet loopback
    # LAN-Anschluss 1
    # an pppoe ist das DSL-Modem angeschlossen
    auto pppoe
    iface pppoe inet manual

    # pppoe Verbindung
    auto dsl-provider
    iface dsl-provider inet ppp
    pre-up /sbin/ifconfig pppoe up   # line maintained by pppoeconf
    provider dsl-provider

    #LAN-Anschluss 2
    #lokales netzwerk
    auto eth0
    iface eth0 inet static
    address 192.168.1.1
    netmask 255.255.255.0
    broadcast 192.168.1.255

    # Maskieren der LAN-Schnittstelle, Port-Forwarding & NAT aktivieren
    # vorhandene Regeln und Ketten zuerst loeschen
    up /sbin/iptables -F
    up /sbin/iptables -X
    up /sbin/iptables -t nat -F

    # Forwarding fuer alle verwendeten Schnittstellen im lokalen Netz aktivieren
    up /sbin/iptables -A FORWARD -o ppp0 -s 192.168.0.0/16 \
    -m conntrack --ctstate NEW -j ACCEPT
    up /sbin/iptables -A FORWARD -m conntrack --ctstate ESTABLISHED,RELATED\
    -j ACCEPT
    up /sbin/iptables -t nat -A POSTROUTING -o ppp0 -j MASQUERADE
    up /sbin/sysctl -w net.ipv4.ip _ forward=1

    # dnsmasq neu starten
    up /etc/init.d/dnsmasq restart
```

Listing 2: Konfiguration der interfaces und iptables

```
#!/bin/bash
USERNAME='xxx'
PASSWORD='yyy'
ENCODED=$(echo -n „$USERNAME:$PASSWORD" | base64
REQUEST="GET /nic/update HTTP/1.1\nhost: nic.  changeip.com\n\Authorization: Basic $ENCODED\n\n"
echo -e $REQUEST | openssl s _ client -connect nic.ChangeIP.com:443\ -quiet -ign _ eof
```

Listing 3: /etc/ppp/if-up.d/5dyndns

Programmierung

Für die Backendprogrammierung wurde Python eingesetzt. Der Webserver kann dann mittels PHP sockets Befehle (oder besser gesagt Events) an das Backend schicken.

Display Ansteuerung: Um die Ansteuerung des Displays aus Programmsicht zu vereinfachen, habe ich eine kleine Wrapper-Klasse geschrieben, mit der sich das Display steuern lässt. Für die Kommunikation mit der seriellen Schnittstelle wird *python3-serial* verwendet. Dieses Modul lässt sich unabhängig vom Rest des Systems verwenden, um das Display beispielsweise bequem aus einem Python Interpreter heraus verwenden zu können.

Server & Eventsystem: Der Server lauscht auf einem lokalen Port auf vordefinierte Events, die dann an einen kleinen Eventbus weitergegeben werden und von den dort angemeldeten Listenern abgearbeitet wird. Da hier für jedes abgearbeitete Event ein eigener Thread gestartet, der dann die jeweiligen Funktionen bei dem jeweiligen Listener abarbeitet. Da es für das Display jedoch keinen Sinn macht, parallele Schreibzugriffe auf zuzulassen, werden die Events im DisplayListener nacheinander abgearbeitet, wobei je hier je nach Einstellung unterschiedlich lange Blockadezeiten festgelegt sind. Die eigentliche Serverklasse, ebenfalls ein EventListener, gibt die Events, die an den Server geschickt wurden an den Event-Dispatcher weiter. Außerdem wird noch ein Thread für die Überwachung der GPIO-Buttons benötigt. Dieser richtet eine Liste übergebener GPIO-Ports ein und überwacht diese daraufhin mittels polling. Falls ein Button gedrückt wird, wird ein Event. das vorher an einen entsprechenden Button zugewiesen wurde, an den Event-Dispatcher weitergeleitet und dort dann von den entsprechenden Listenern bearbeitet.

```
class EventDispatcher(object):
    def __init__(self):
        self.listeners = []
        self.threads = dict()
    def fire(self, calledEvent):
        for eventname, eventListener in self.listeners:
            if calledEvent.getName() == eventname:
                self.threads[eventListener] = \
                threading.Thread(target=self.__call, args =\
                (eventListener, calledEvent)
                self.threads[eventListener].start()
        for thread in self.threads:
            if not thread == None:
                thread.join()
    def __call(self, eventListener, calledEvent):
        eventListener.catchEvent(calledEvent)
    def register(self, event, eventListener):
        self.listeners.append((event.getName(), eventListener))
    def unregister(self, event, eventListener):
        self.listeners.remove((event.getName(), eventListener))

class EventListener(object):
    def __init__(self, knownEvents = None):
        self.eventMap = dict()
        for element in knownEvents:
            self.eventMap[element] = None
    def catchEvent(self, event):
        raise NotImplementedYet(„function overwritten")
    def getKnownEvents(self):
        listing = list()
        for i, _ in self.eventMap:
            list.append(i)
        return listing
```

Listing 4: EventDispatcher und EventListener

Weboberfläche: Die Umsetzung der Weoberfläche erfolgt mittels einfachen HTML und PHP. Einstellungen werden dabei nicht direkt umgesetzt sondern zuerst in eine Datenbank geschrieben und dann durch einen an den oben beschriebenen Server geschickten Befehl übernommen. Auf diese Weise sind für den PHP-Prozess bzw. Apache Server keine Root-Rechte, beispielsweise für den Aufruf von *iptables*, nötig.

Quellen

[1] Olimex A13 OLinuXino: goo.gl/r9PlXO

[2] Sunxi Linux: http://linux-sunxi.org

[3] Ubuntuuser Forum: http://wiki.ubuntuusers.de

[4] Changeip: http://www.changeip.com/accounts/knowledgebase.php?action=displayarticle&id=34

USBprog 5.0

Open-Source-Programmer mit Webinterface

Benedikt Sauter <sauter@embedded-projects.net>

Jens Nickel <jens.nickel@eimworld.com>

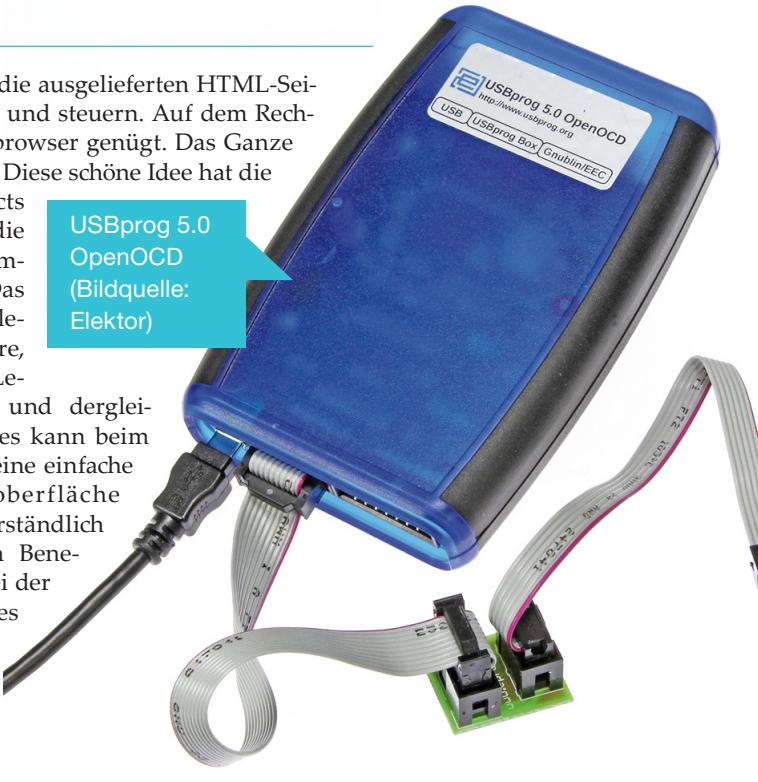
Der Open-Source-Programmer USBprog erfreute sich bei Entwicklern großer Beliebtheit, denn er war als „Multi-Tool“ für die unterschiedlichsten Controller zu gebrauchen. Hier kommt eine neue Version, die sogar mit einem Webserver ausgestattet ist. Dank der HTML-Oberfläche arbeitet der Programmer mit allen PC-Betriebssystemen und sogar Tablets zusammen, auch die Installation eigener Software entfällt. Eine Schnittstelle zur Automatisierung, eine Speichermöglichkeit für Hex-Files und die Integration des GDB-Debuggers für ARM sind weitere interessante Features.

Einleitung

Immer mehr Geräte werden mit Webservern ausgestattet. Über die ausgelieferten HTML-Seiten lassen sie sich dann von einem Computer aus konfigurieren und steuern. Auf dem Rechner muss dazu keine Software installiert werden, ein Standardbrowser genügt. Das Ganze funktioniert mit jedem Betriebssystem und auch vom Tablet aus. Diese schöne Idee hat die Embedded Projects GmbH nun auf die Welt der Programmer ausgedehnt. Das Flashen und Auslesen von Firmware, das Setzen und Lesen von Fusebits und dergleichen mehr, das alles kann beim USBprog 5.0 über eine einfache HTML-Benutzeroberfläche erfolgen. Selbstverständlich hat das Team um Benedikt Sauter auch bei der neuesten Version des bekannten USBprog wieder voll auf den Open-Source-Gedanken gesetzt.

Eigenschaften

- AVR-Programmer (avrdude [5])
- ARM-JTAG-Debugger und -Programmer (openocd [4])
- Updatefähigkeit: Weitere unterstützte Controller in Vorbereitung
- Pegelwandler (einstellbar 1,8 V, 3,3 V und 5,0 V)
- Browseroberfläche für einfache Bedienung
- Automatische Bedienung per Kommandozeilen-Tool
- Verwendung aus Atmel Studio und anderen Programmen heraus
- Speichermöglichkeit für häufig verwendete Hexfiles



USBprog 5.0 OpenOCD (Bildquelle: Elektor)

Anschluss am PC

Der USBprog 5.0 ist nichts anderes als ein kleines Linux-Board im formschönen Gehäuse. Der kleine Kasten wird über USB mit dem PC verbunden und hierüber auch mit Strom versorgt. Per USB wird eine Netzwerkschnittstelle emuliert; unter Mac und Linux wird das Gerät automatisch gefunden und verbunden. Auf dem USBprog läuft ein DHCP-Server, der dem PC eine IP-Adresse zuteilt. Danach kann über die Adresse 10.0.0.1 per Browser auf den Programmer zugegriffen werden.

Unter Windows springt beim ersten Anschließen der bekannte Treiber-Dialog auf. Hier gibt man an, dass man einen Treiber manuell auswählen möchte. Als Geräteart wählt man „Network adapters“ bzw. „Netzwerkkarten“. Anschließend erscheint eine Auswahl von verschiedenen Herstellern. Hier wählt man „Microsoft Corporation“ und „Remote NDIS Compatible Device“.

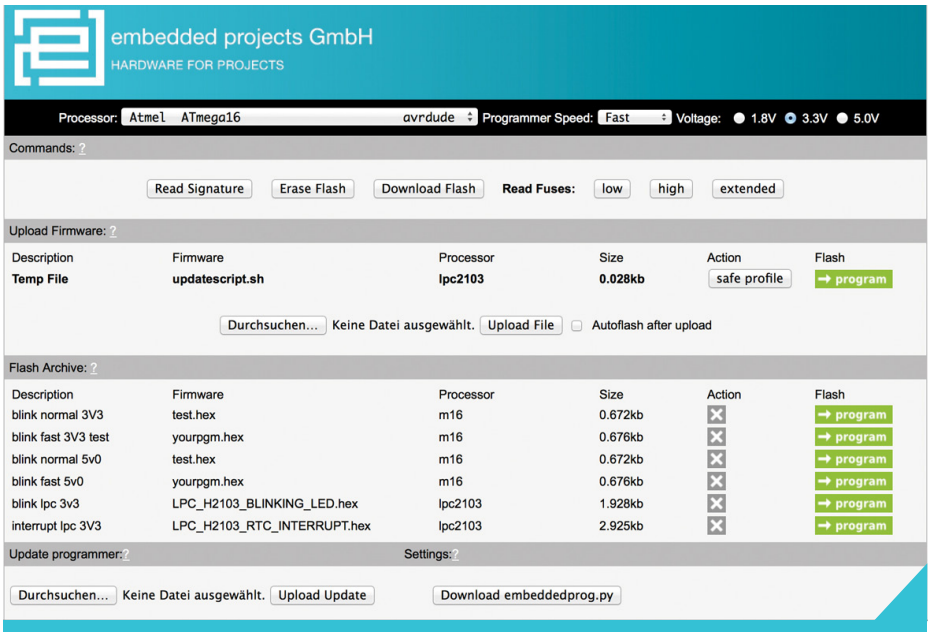


Abb. 1: Der Programmer lässt sich über eine HTML-Oberfläche steuern, die in jedem Browser dargestellt werden kann. (Bildquelle: Elektor)

Für ARM und AVR

Nachdem die Netzwerkverbindung steht, kann man sich über einen Browser (http://10.0.0.1) mit dem Programmer verbinden und sieht eine Oberfläche wie in Bild 1. Anschließend verbindet man den USBprog mit dem Zielprozessor; das Target wird dann auch mit Strom versorgt. Aktuell werden ARM-Prozessoren und Atmel-Mikrocontroller unterstützt; später werden noch PIC-Controller hinzukommen, ein Firmware-Update ist bereits in Vorbereitung.

Für die AVR-Controller bringt der Programmer den üblichen 10-poligen ISP-Wannenstecker mit (Bild 2), über ein Flachkabel geht es weiter zum entsprechenden Gegenstück auf dem Zielboard. Um sich auch mit 6-poligen ISP-Steckern verbinden zu können, bringt der USBProg eine kleine Adapterplatine mit. Die Wannenstecker muss man sich selbst einlöten, was aber keine großen Lötkünste erfordert.

Zum Flashen und Debuggen von ARM-Controllern ist ein Adapter auf einen 20-Pin-JTAG-Steckverbinder erforderlich, der ebenfalls zum Selbst-Löten mitgeliefert wird.

Daneben ist der updatefähige Programmer bereits mit einem 14-poligen Gnublin/EEC-Steckverbinder ausgestattet, über den sich später einmal die bekannten Gnublin-Erweiterungsboards steuern lassen werden – natürlich ebenfalls per Weboberfläche.

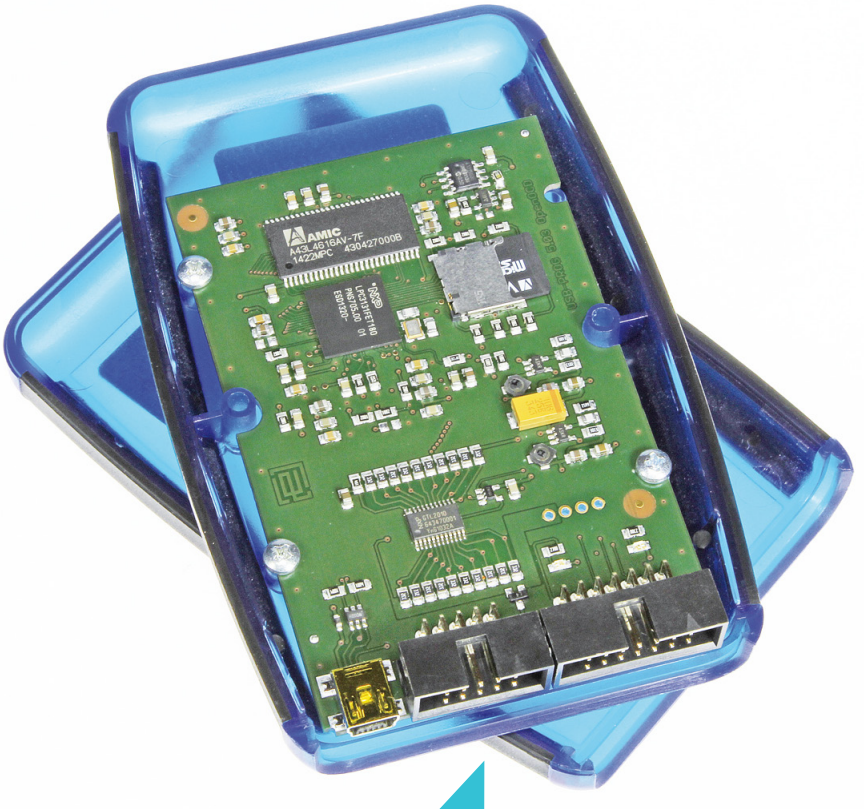


Abb. 2: Anschlüsse des USBprog 5.0. Rechts unten erkennt man den 14-poligen Gnublin/EEC-Steckverbinder; nach einem Firmware-Update werden sich später auch einmal die Gnublin-Erweiterungsboards ansteuern lassen. (Bildquelle: Elektor)

Manuelle Bedienung

In der Browseroberfläche wählt man nun zuerst den Prozessor aus, hierfür ist die Auswahlbox ganz oben zuständig. Um beispielsweise einen ATmega328P (Arduino Uno [1]) oder ATmega328 (T-Board 28 [2]) auszuwählen, reichen jeweils ein paar Buchstaben. Mit

den Radio-Buttons rechts daneben wird die Spannung des Zielprozessors ausgewählt.

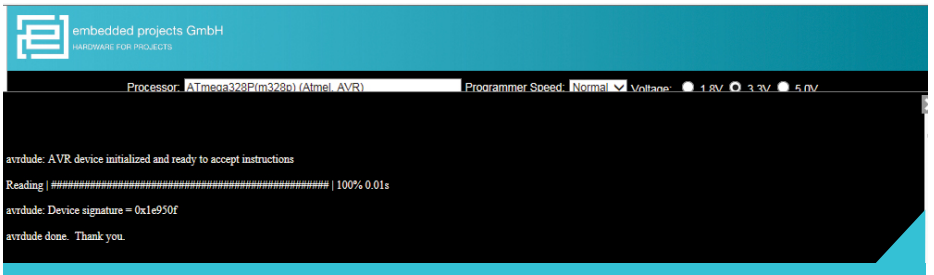


Abb. 3: Wie man hier sieht, nutzt der Programmer das Open-Source-Programmierinterface avrdude. (Bildquelle: Elektor)

Zuerst sollte man probeweise einmal die Signatur des Controllers oder die Fuses auslesen. Es sollte sich nach ein paar Sekunden jeweils ein schwarzes Textfenster mit dem Ergebnis öffnen (Bild 3). Danach geht es an das Programmieren eines ersten Hex-Files, hierfür ist der Abschnitt „Upload Firmware“ der Benutzeroberfläche zuständig. Mit „Browse“ wählt man ein File aus, mit „Upload File“ wird dieses zuerst auf den Programmer gespielt und schließlich mit dem grünen Button „-> program“ auf den Zielprozessor übertragen. Man kann das Hex-File auch zusammen mit einem kurzen Titel im Programmer abspeichern, wenn man auf den Button „safe profile“ klickt. Die abgespeicherten Files erscheinen im Abschnitt „Flash Archive“. So lassen sich zum Beispiel mehrere Kleinserien von Produkten recht rasch mit der jeweiligen Firmware versorgen.

Bedienung per Skript

Die manuelle Bedienung des Programmers im Webbrowser ist komfortabel, deckt aber noch nicht alle Anwendungsszenarien ab. Beim Entwickeln von Firmware arbeitet man meist mit einer Umgebung wie Atmel Studio; da spart es Zeit, wenn man aus der IDE heraus das kompilierte Programm auch gleich in den Controller flashen kann. Auch das automatisierte Flashen per Batch-Datei oder eigener PC-Software sollte mit dem Programmiergerät natürlich möglich sein.

Daher hat das Team um Benedikt Sauter ein kleines Tool in Python geschrieben, das auf dem Entwicklungsrechner per Kommandozeile aufgerufen wird. Ein Vorteil von Python ist, dass es Interpreter für alle gängigen Betriebssysteme gibt. Auf einem Mac oder Linux-Rechner kann man „embeddedprog.py“ direkt ohne zusätzliche Installation nutzen. Dort gehört Python zu den Standards des Betriebssystems. Man muss lediglich das Python-Tool selbst auf den Rechner spielen, was man über die Browseroberfläche mit dem Button „Download embeddedprog.py“ erledigen kann.

Auf der Kommandozeile ruft man nun den Python-Interpreter auf, der das Python-Programm ausführt. Der Prozessor und die Aktion, die der Programmer ausführen soll, werden als Parameter mitgegeben. Um beispielsweise die Signatur eines ATmega328P auszulesen, lautet der Aufruf (aus dem Ordner heraus, wo das Tool abgelegt ist):

```
python embeddedprog.py \
--processor m328p --speed 2
```

Weitere Kommando-Parameter sind auf der Web-Oberfläche unten aufgeführt.

Wichtig: Beim ersten Aufruf des Python-Tools muss einmalig die IP-Adresse des Programmers angegeben werden. Diese wird anschließend lokal in einer Konfigurationsdatei gespeichert und ab da automatisch mit angefügt.

```
python embeddedprog.py \
--eeprog-ip 10.0.0.1 \
--eeprog-port 8888 \
--processor m328p --speed 2
```

Unter Windows muss zuerst Python installiert werden [3]. Das Python-Tool embeddedprog.py kann man sich herunterladen und an passender Stelle ablegen, auf der Kommandozeile muss dann aber der Pfad auf den Python-Interpreter und auf das Python-Tool mitangegeben werden. Komfortabler fährt man mit einer Exe-Datei, die auf dem Windowsrechner installiert wird (die passende Setup-Datei lädt man sich über den Button „Download setup.exe“ herunter). Bei der Installation werden Windows auch die Pfade bekannt gemacht; außerdem wird die Konfigurationsdatei mit der IP-Adresse angelegt. Auf der Windows-Kommandozeile (die man im Startmenü mit „cmd“ erreicht) ruft man nun einfach die Exe-Datei auf:

```
embeddedprog.exe \
--processor m328p --speed 2
```

Mit vielen Programmiersprachen lassen sich Kommandozeilenaufrufe absetzen (und die Ausgabezeilen auswerten), so dass man den Programmer mit selbst geschriebener Software ansteuern kann. In einer der nächsten Ausgaben stellen wir eine entsprechende Erweiterung des EFL-Konfigurators vor.

Auch die Bedienung direkt aus Atmel Studio 6 heraus ist dank des Python-Tools möglich (siehe Kasten).

Debuggen von ARM-Prozessoren

Der neue USBprog kann auch zum Debuggen von ARM-Prozessoren verwendet werden. Hierfür wird der Open-Source-Debugger openocd [4] genutzt, der auf dem Programmer läuft. Vor dem Debuggen muss man die Firmware in den Prozessor übertragen. Anschließend startet man den Debugger per Browser („Start openocd“, siehe Bild 4) oder über die Kommandozeile. Der Debugger kann entweder über ein einfaches Telnet-Interface oder über eine GDB-Schnittstelle gesteuert werden. Größere Entwicklungsumgebungen bieten hier häufig ein entsprechendes Interface. Im Internet findet man viele Anleitungen, wenn man nach „ARM Eclipse GDB“ sucht. Als „Remote Address“ muss man anstelle des Eintrags „localhost“ die IP-Adresse des Debuggers eintragen (10.0.0.1).

Das Innenleben

Bild 5 zeigt einen Blick ins Innere des Programmers: Der Hauptprozessor ist ein LPC3131, dem Arbeitsspeicher vom Typ A43L4616 (8 MB) zur Seite gestellt ist. Für die Pegelwandlung wurde der Baustein GTL2010PW ausgewählt. Er ermöglicht eine einfache bi-

WÜRTH ELEKTRONIK

Willkommen an Bord

Die Würth Elektronik Gruppe fertigt und vertreibt elektronische, elektromechanische Bauelemente, Leiterplatten und intelligente Systeme. Weltweit sorgen über 6.000 Mitarbeiter dafür, dass der Bereich der elektronischen und elektromechanischen Bauelemente mit Sitz in Waldenburg einer der erfolgreichsten der Würth-Gruppe ist.

Unter gemeinsamer Flagge!

Teamgeist? Abenteuerlust? Werden Sie Teil unserer Mannschaft!

Traineeprogramm im Bereich Elektronikvertrieb I PLZ 80-87 & 94

Traineeprogramm im Bereich Elektronikvertrieb I PLZ 90-93 & 95-96

(Junior-) Vertriebsmitarbeiter (m/w) I Großraum München

(Junior-) Vertriebsmitarbeiter (m/w) I Großraum Nürnberg

(Junior-) Vertriebsmitarbeiter (m/w) I PLZ 90-93 & 95-96

(Junior-) Vertriebsmitarbeiter (m/w) I PLZ 86-87

www.we-online.de/karriere

more than you expect

direktionale Wandlung der Signale. Um die Spannung für die Zielhardware (1,8 V, 3,3 V und 5,0 V) per Software auswählen zu können, wurde ein LDO AP2127K-ADJ verwendet, der von einem digitalen Poti MCP4131 angesteuert wird. Bis zu 300 mA Ausgangsspannung sind möglich.

Auf dem Prozessor läuft ein Embedded-Linux, auf dem wiederum Standardsoftware wie avrdude (für das Programmieren der Atmel-Prozessoren [5]) ihre Arbeit verrichtet. Als zentrale Komponente wurde ein Server in Python geschrieben, der sich zur Auslieferung der Weboberfläche wiederum des Lighttpd-Webservers bedient. Außerdem stellt er einen Webservice zur Steuerung des Programmers bereit, mit dem wiederum die Python-Tools auf dem PC kommunizieren. Alle Quelltexte sind auf GitHub veröffentlicht [6].

Update per Weboberfläche

Um den Programmer in Zukunft einfach mit neuen Firmware-Updates versorgen zu können, wurde eine Updatefähigkeit gleich mit integriert. Hierzu lädt man sich die neueste Version des Updates (eine gepackte Datei) von der USBprog-Webseite [7] herunter. Das Update kann dann einfach über die Browseroberfläche eingespielt werden. Es wird nach dem Upload auf dem Programmieradapter entpackt und entsprechend an die richtigen Stellen im Betriebssystem kopiert.

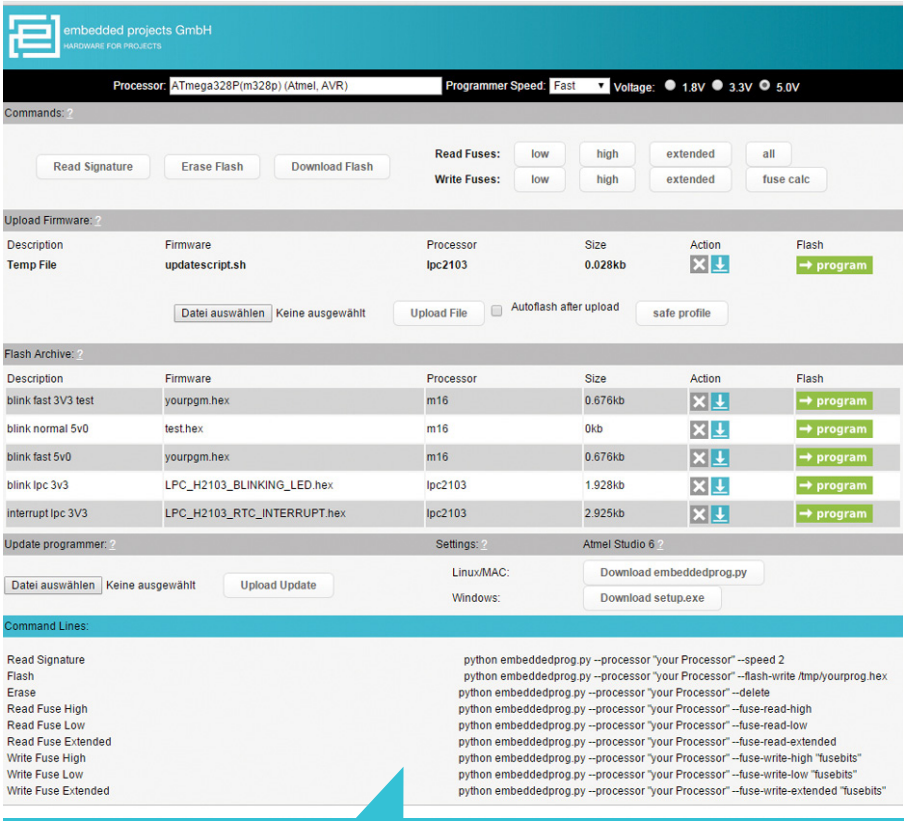


Abb. 4: Die Oberfläche nach dem Auswählen eines ARM-Controllers. (Bildquelle: Elektor)

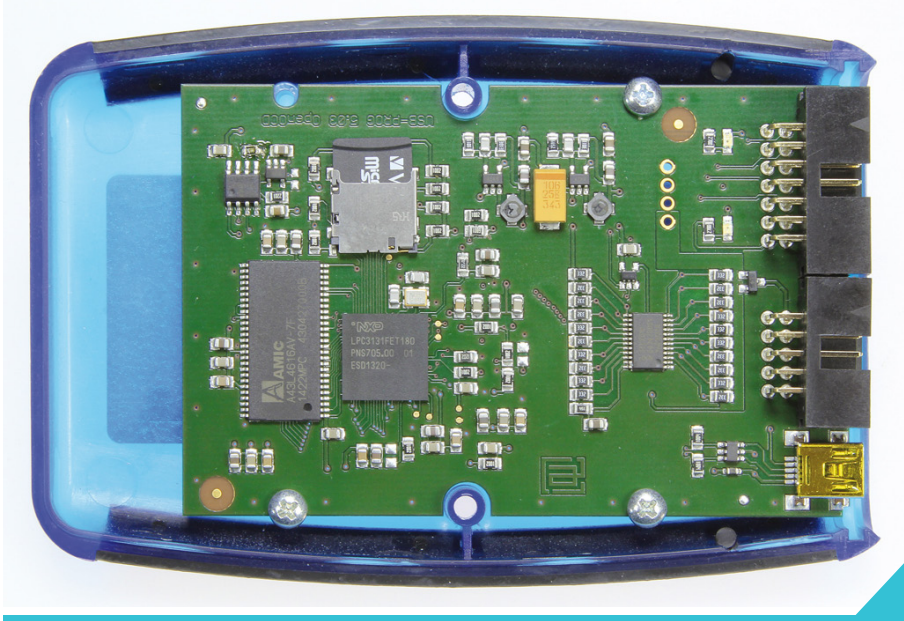


Abb. 5: Basis des Programmers ist ein LPC3131, auf dem Linux läuft. (Bildquelle: Elektor)

Zusammenarbeit mit Atmel Studio

Was wäre ein Programmer für AVR-Controller, der nicht auch aus der kostenlosen Entwicklungsumgebung Atmel Studio heraus bedienbar wäre? Das klappt auch beim USBprog - dank der Ansteuerung per Kommandozeilen-Tool (siehe Text). Zwei Aktionen dürften dabei am wichtigsten sein: nämlich das Flashen von Hexfiles auf den AVR-Controller und das Aufrufen der Browser-Oberfläche von Atmel Studio aus (zum Beispiel um Settings einzugeben).

Für beide Aktionen kann man in Atmel Studio unter Tools -> External Tools einen Menüpunkt anlegen. Nach einem Klick auf den Button „Add“ ist hierfür jeweils in den Textboxen „Title“, „Command“ und „Arguments“ ein Eintrag vorzunehmen, danach klickt man auf „Apply“. Wir beginnen mit „USBprog 5.0 Program“ für das Uploaden von Firmware (siehe Screenshot). Unter „Command“ ist jeweils der gesamte Pfad auf das Tool embeddedprog.exe (siehe Text) einzutragen.

Hat man für beide Aktionen einen Eintrag wie in den Screenshots gezeigt angelegt, dann erscheinen die beiden Einträge im Atmel-Studio-Hauptmenü unter „Tools“.

Betätigt man zum ersten Mal „USBprog 5.0 Program“, dann springt die Browseroberfläche auf. Hier muss man zuerst den Prozessor und dessen Betriebsspannung einstellen. Ab jetzt kann man über diesen Menüpunkt direkt aus Atmel Studio heraus Hexfiles in den Controller flashen.

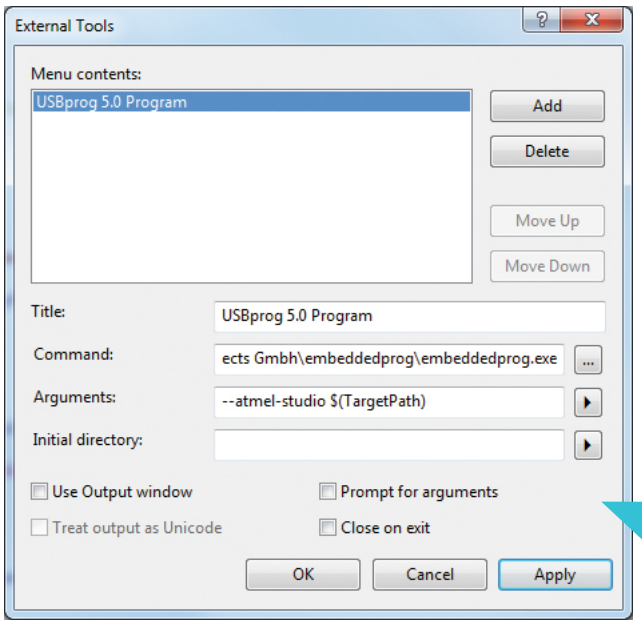


Abb. 6: In Atmel Studio ein neues „External Tool“ namens „USBprog 5.0 Program“ hinzufügen, um einen AVR aus der IDE heraus flashen zu können. (Bildquelle: Elektor)

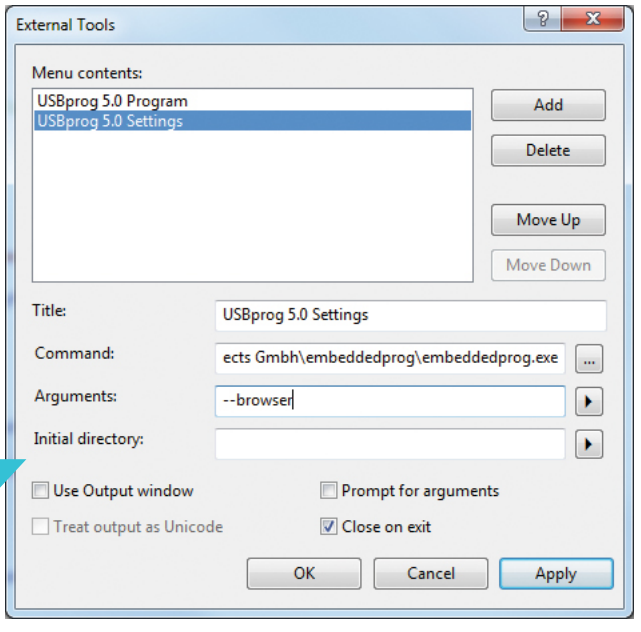


Abb. 7: Einrichten von USBprog 5.0 Settings erlaubt das Öffnen der Weboberfläche, in welcher weitere Einstellungen vorgenommen werden können. (Bildquelle: Elektor)

Weblinks

[1] www.elektor.de/arduino-uno

[2] www.elektor.de/t-board-28-130581-93

[3] https://www.python.org

[4] www.openocd.org

[5] www.nongnu.org/avrdude

[6] https://github.com/embeddedprojects/usbprog5

[7] http://usbprog5.embedded-projects.net

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

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!

elektor GOLD membership

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!

WEdirekt.

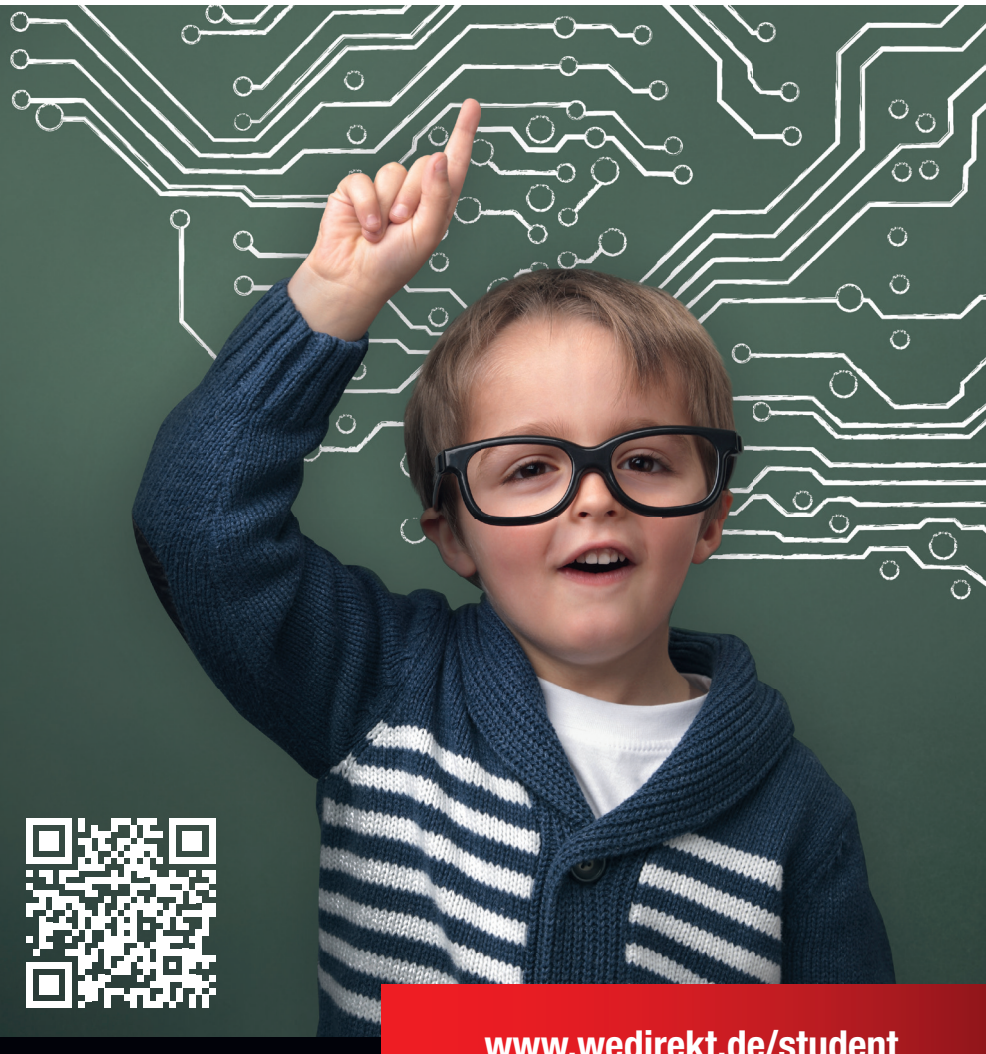
- Leiterplatten ab 1 Stück
- Industriequalität zu attraktiven Preisen
- Expresslieferung ab 2 AT in chem. Ni/Au
- Unkomplizierte Konfiguration und Bestellung

ONLINE PCBs

powered by Würth Elektronik

www.wedirekt.de

WEdirekt.



PCB Online Shop für Schüler und Studenten – jetzt Vorteile sichern.

Wir sind Dein Partner!

powered by Würth Elektronik

www.wedirekt.de/student

Telefonwählmodul für Alarmanlagen
mit Microchip-Controller und DTMF-Transceiver

Kai Dorau <kai.dorau@gmx.net>

Hausbesitzer kennen folgende Situation. „Wie schütze ich Hab und Gut vor Einbrechern?“ Man verbringt seinen Urlaub an der Ostsee und die Sorge um das Eigenheim ist groß. Eine Alarmanlage schützt die Immobilie in vielen Fällen. Moderne Anlagen können so konfiguriert werden, dass der Besitzer bei einem Alarm darüber entweder per Internet oder per Telefonwahl informiert wird. Wer allerdings eine ältere Alarmanlage sein Eigen nennt oder wem der Alarm-Service eines Sicherheitsdienstes seiner professionellen Alarmanlage zu teuer ist, möchte trotzdem über einen Alarm informiert werden.

Ich besitze eine alte Alarmanlage von Raab-Karcher, die ich bei dem Kauf meines Hauses mit übernommen habe. Diese Anlage verfügt über ein ISDN-Modul, das im Falle eines Alarms eine Service-Nummer eines Raab-Karcher Sicherheitsdienstes wählt. Die Services des Sicherheitsdienstes waren mir einfach zu teuer. Aus Kostengründen und mit netten Nachbarn gesegnet, habe ich mich entschlossen, eine eigene Warnquelle zu bauen. Nachfolgend möchte ich eine Telefonwahleinrichtung vorstellen, die ich mit meiner Alarmanlage gekoppelt und in Betrieb genommen habe.

Funktionalität

- Benachrichtigung bei Alarm ins Festnetz oder Mobilnetz
- Abspeicherung von beliebig vielen Telefonnummern, die bei Alarm angerufen werden
- kein Internet oder Internet-Provider nötig
- Eindeutige Benachrichtigung durch „Alle meine Entchen“
- Betriebsspannung über Alarmanlage
- Bei Netzausfall wird das Wählmodul über USV der Alarmanlage gespeist und funktioniert weiter
- Günstige Alternative zu teuren Alarm-Services
- Alarmeingang kann an eine vorhandene Alarmanlage angepasst werden
- Läuft 24h 365 Tage im Jahr
- Niedriger Stromverbrauch von ~80 mA in Bereitschaft
- Eingangsspannung von 7V..18V

Die Idee

Um Kontakt mit meiner Alarmanlage herzustellen, gab es zwei Möglichkeiten. Entweder nutze ich das Internet oder das Telefonnetz. Ich entschied mich für das letztere, weil es das umfangreichste Netz weltweit ist und die Telekommunikation den günstigsten Service bereitstellt. Darüber hinaus besaß ich vor einigen Jahren noch kein Smartphone, sondern nur ein Mobile ohne Internetzugang. Telefonieren konnte ich bisher an jedem Ort zu jeder Zeit. Aus dieser Erfahrung entstand ein Gerät, das ich gerne vorstellen möchte und das mich seit Jahren zuverlässig benachrichtigt, wenn meine Katze mal wieder den Fokus eines Bewegungsmelders im Haus durchquert. Glücklicherweise gab es bisher noch keinen Ernstfall.

Die Alarmanlage

Meine Raab-Karcher-Alarmanlage verfügt zwar über ein sehr kostenintensives ISDN-Modul, aber besitzt leider keinen ausgewiesenen Alarmausgang. Für einen Elektronikbastler ist das aber kein Problem. Ich habe meine Tochter an einen der angeschlossenen Bewegungsmelder positioniert. Anschließend habe ich an der einzigen Pfostenleiste am ISDN-Modul vom ersten bis zum letzten Pin nacheinander ein Vielfachmessgerät angeschlossen, die Anlage scharf geschaltet und meine Tochter animiert, doch mal zu zappeln. Nach einer halben Stunde lag das Ergebnis vor:

PinX, 0..12V, active-low bedeutet:

- kein Alarm: PinX führt 12V
- Alarm: PinX führt 0V

Jetzt hatte ich einen Eingang für meine Telefonwähleinrichtung. Darüber hinaus bot sich eine Betriebsspannung von 12V für das Wählmodul geradezu an.

Das Telefonwählmodul

An dieser Stelle gab es für mich intuitiv nur eine Möglichkeit: Eine Schaltung aus einer Kombination aus Mikrocontroller und Telekommunikations-IC. Nach einer intensiven Internetrecherche kristallisierte sich der MT8880C von Zarlink heraus. Hierbei handelt es sich um einen DTMF-Transceiver, der in der Lage ist, Tonsignale zu senden, aber auch zu empfangen. DTMF steht für Dual-Tone-Multi-Frequency (oder auch Mehrfrequenzwahlverfahren) und ist das aktuelle analoge Wahlverfahren in Deutschland. In Tabelle 1 sind die Tasten eines Telefons und die entsprechenden Dual-Töne aufgelistet.

Der MT8880C DTMF-Transceiver

Die Funktion des MT8880C ist aus dem Blockschaltbild sehr gut er-sichtlich.

Links befindet sich die a/b-Schnittstelle:

- TONE: Ausgang
- IN+, IN-: Eingang

Rechts befindet sich die Controller-Schnittstelle:

- D0..D3: Bidirektionale Daten (Tastennummer)
- IRQ/CP (Call Progress): Ausgang (IRQ: Tastennummer empfangen, CP: Anzeige des Wählstatus)
- φ, CS, R/W, RS0: Steuer-Eingänge

Sonstiges:

- OSC1/2: Clock
- VDD, VRef, VSS: Spannungsversorgung

Eine typische Applikationsschaltung findet sich in der Spezifikation des ICs, die in dem Telefonwählmodul größtenteils Einzug hielt.

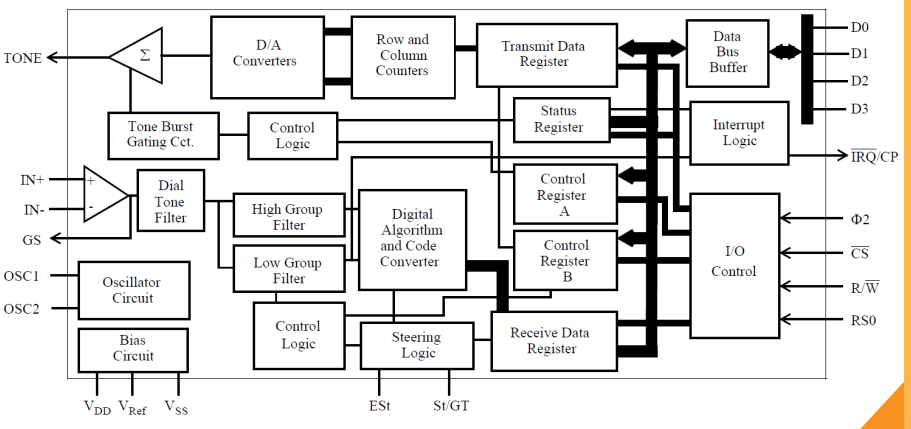


Abb. 1: MT8880C Blockschaltbild

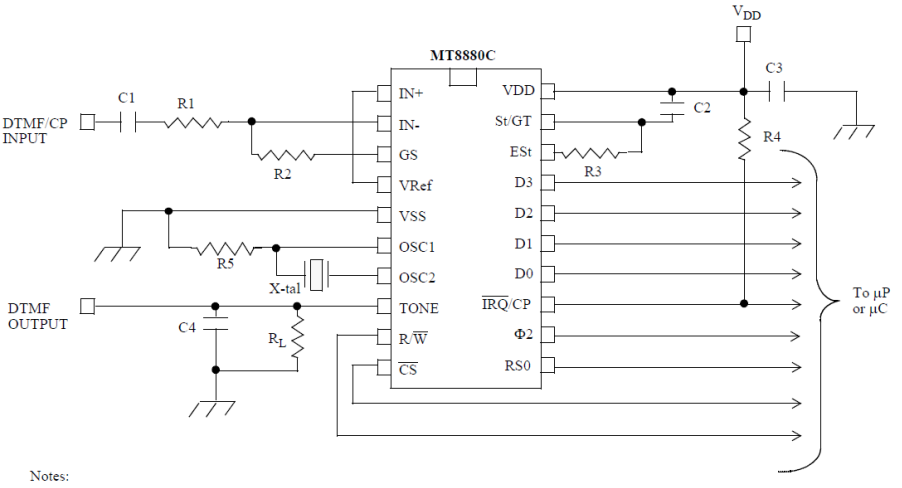
Tabelle 1: Dual-Töne der Tasten eines Telefons

Frequenzen	1209 Hz	1336 Hz	1477 Hz	1633 Hz
697 Hz	1	2	3	A
770 Hz	4	5	6	B
852 Hz	7	8	9	C
941 Hz	*	0	#	D

Der PIC 16f628A

Die Wahl viel auf dem 16f628A, weil der Preis besonders niedrig ist. Er hat genug I/O-Pins, um den MT8880C anzuschließen:

- 16 I/O-Pins für den MT8880C
- USART: Zum Debuggen
- EEPROM 128 Bytes für die Telefonnummern, die das Wähl-modul nacheinander wählen soll.
- 1,75 € bei Reichelt



Notes:
R1, R2 = 100 kΩ 1%
R3 = 374 kΩ 1%
R4 = 3.3 kΩ 10%
R5 = 4.7 MΩ 10%
R_L = 10 k Ω (min.)
C1 = 100 nF 5%
C2 = 100 nF 5%
C3 = 100 nF 10%*
C4 = 10 nF 10%
X-tal = 3.579545 MHz

* Microprocessor based systems can inject undesirable noise into the supply rails. The performance of the MT8880 can be optimized by keeping noise on the supply rails to a minimum. The decoupling capacitor (C3) should be connected close to the device and ground loops should be avoided.

Abb. 2: MT8880C Applikation

Der Stromlaufplan

Die Schaltung des Telefonwählmoduls ist relativ übersichtlich aufge-baut. Die Funktionen der einzelnen Bauteilegruppen sind nachstehend erklärt:

MT8880C: Die Daten- und Kontrollleitungen des MT8880C sind mit den I/O-Pins des 16F628A verbunden. Der IRQ/CP-Ausgang ist an einem triggerbaren monostabilen Mutlivibrator 74LS123 angeschlossen. Ansonsten ist die MT8880C-Beschaltung entsprechend nach Spezifikation. Er wird separat mit 3,579545 Mhz getaktet.

16F628A: Der Microcontroller ist für die gesamte Steuerung des Moduls zuständig. Er wird mit einem 4 Mhz- Quarz getaktet. 2 LEDs sind für Statusanzeigen vorgesehen.

74LS123: Der triggerbare monostabilen Mutlivibrator ist nötig, weil die 425 Hz Wählzeichenpakete direkt über den IRQ/CP-Ausgang des MT8880C anliegen und entsprechend zusammengefasst werden, um sie von der Software besser zählen zu können.

Relais K1: Das Relais verbindet oder unterbricht das Wählmodul mit dem a/b-Netz. Gesteuert wird es über T2 vom Microcontroller 16F626A.

Trafo U$3: Der Telekom-Trafo mit einem Übertragungsverhältnis 1:1 und 600Ω Impedanz sorgt für die galvanische Trennung inkl. Leistungsanpassung zwischen dem a/b-Netz und dem Wählmodul. Er ist mit dem Eingang und Ausgang des MT8880C verbunden.

Optokoppler OK1: Der Optokoppler CNY17 sorgt einerseits für die galvanische Trennung zwischen Alarmanlage und Wählmodul und andererseits wird

die Ausgangsspannung der Alarmanlage bei Alarm von 12V active-low auf 5V active-low angepasst.

7805 IC4: Der Optokoppler OK1 arbeitet primär mit 12V Betriebsspannung aus der Alarmanlage. Genau diese Spannung wird für die Speisung des Wählmoduls verwendet. Dazu ist ein kleines Netzteil mit einem Linearregler 7805 nötig. Die Verlustleistung eines Linearreglers ist relativ hoch vor allen Dingen, wenn das Modul ständig in Betrieb ist. Besser wäre an dieser Stelle ein kleiner Schaltregler. Ein ander Vorteil der Betriebsspannung aus der Alarmanlage ist, das sie über eine USV verfügt. Sollte ein Eindringling den Strom zum Haus kappen, läuft die Alarmanlage und das Wählmodul weiter.

Die Funktion der Schaltung sollte nun jedem Hobby-Elektroniker keine großen Probleme bereiten.

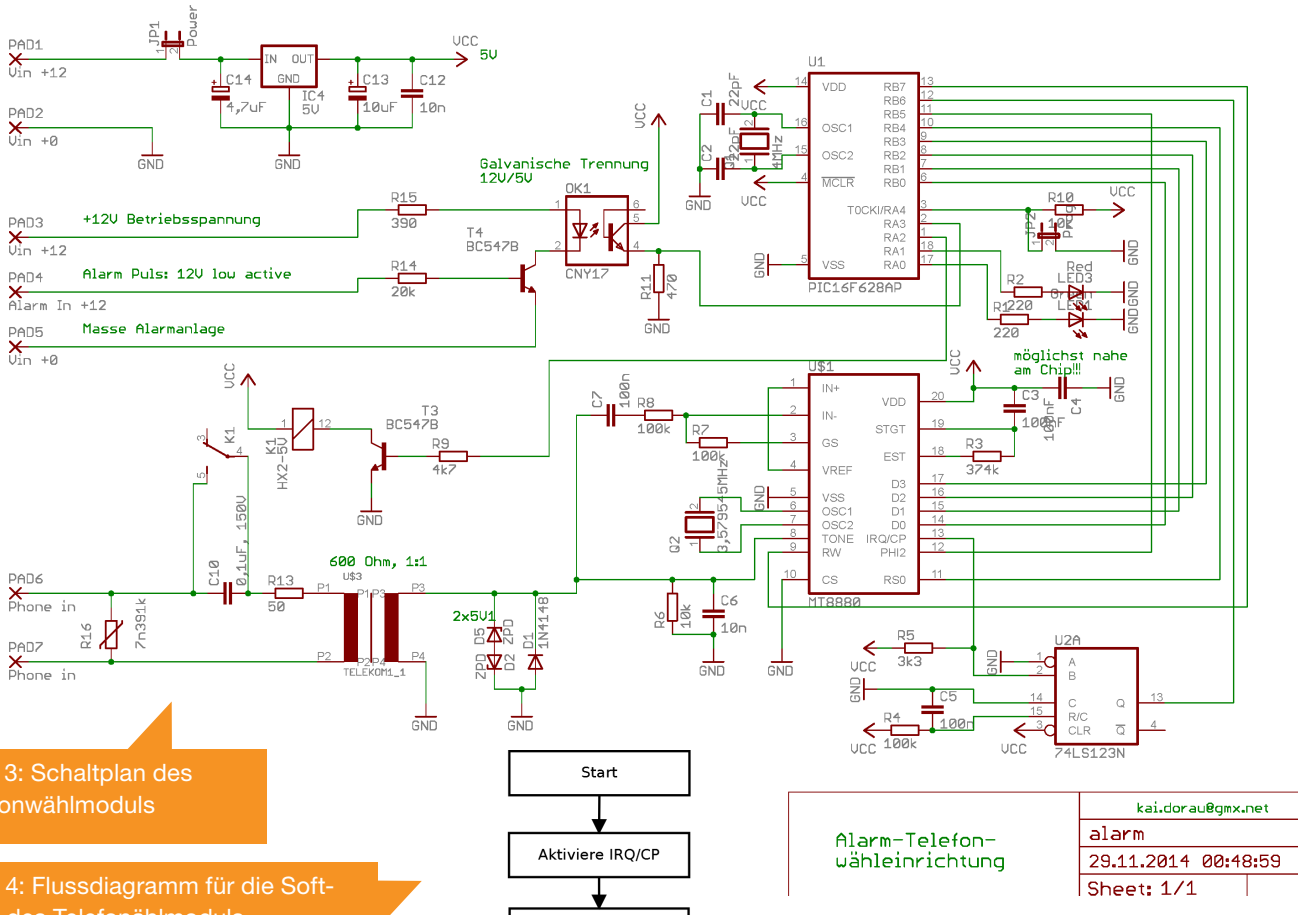


Abb. 3: Schaltplan des Telefonwählmoduls

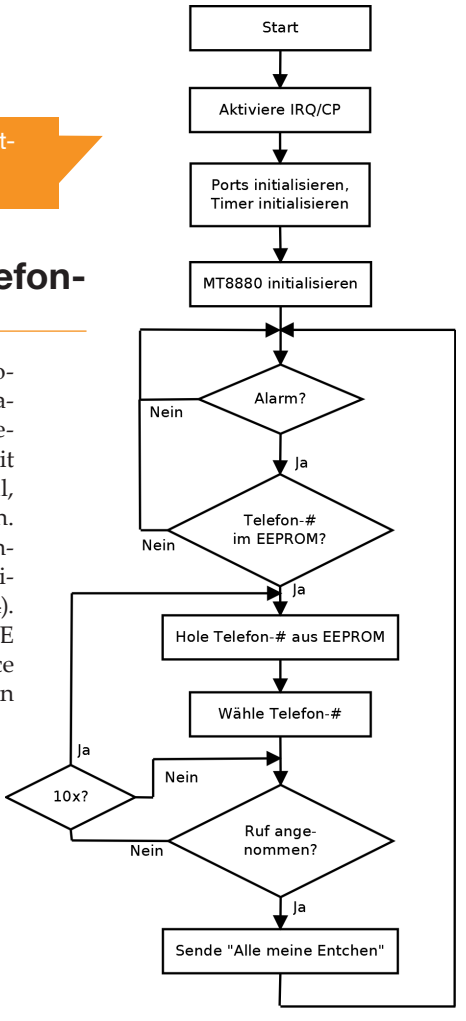


Abb. 4: Flussdiagramm für die Software des Telefonählmoduls

Die Software des Telefonwählmoduls

Der 16f628A ist ein Controller, der programmiert werden will. Die Applikation des Wählmoduls ist in C geschrieben. Assembler stand zu keiner Zeit zur Debatte. Nun ist es nicht sinnvoll, den gesamten C-Code abzubilden. Deshalb möchte ich ein selbsterklärendes Flussdiagramm für die Beschreibung der Software zeigen (Abb. 4). Die Entwicklungsplattform ist die IDE Piklab mit dem SDCC (Small Device C Compiler) und der Brenner5 von sprut.

Praktischer Aufbau des Telefonwählmoduls

Die Platine des Telefonwählmoduls ist relativ klein und misst 100mm x 75mm. Das Gehäuse habe ich bei Conrad gekauft und ist 150mm x 80mm x 50mm groß.

Aussenansicht: Die Beschriftung ist nicht ganz so gut gelungen. Die rote LED zeigt den a/b-Netzstatus (besetzt, Wählzeichen) an und die grüne LED signalisiert die Wahl einer Telefonnummer. Wenn eine Verbindung besteht, wird dem Empfänger mit „Alle meine Entchen“ mitgeteilt, das ein Alarm aufgetreten ist.

Innenansicht: Links unten ist das Kabel von der Alarmanlage angeordnet. Die drei Leitungen sind Betriebsspannung (grün), Masse (braun) und Alarm (gelb). Aus der Betriebsspannung 12V wird die Versorgungsspannung für das Modul gewonnen. Die a/b-Schnittstelle rechts unten wird mit einem analogen Telefon-

anschluss oder – wie in meinem Fall – an die Telefonanlage (a/b-Anschluss) angeschlossen.

Ich habe überwiegend bedrahtete Bauteile verwendet, also auf SMD-Technologie bewusst verzichtet. Der Grund ist schlicht, weil ich jede Menge Altbestände aus den 80ern und 90ern in der Werkstatt liegen habe.

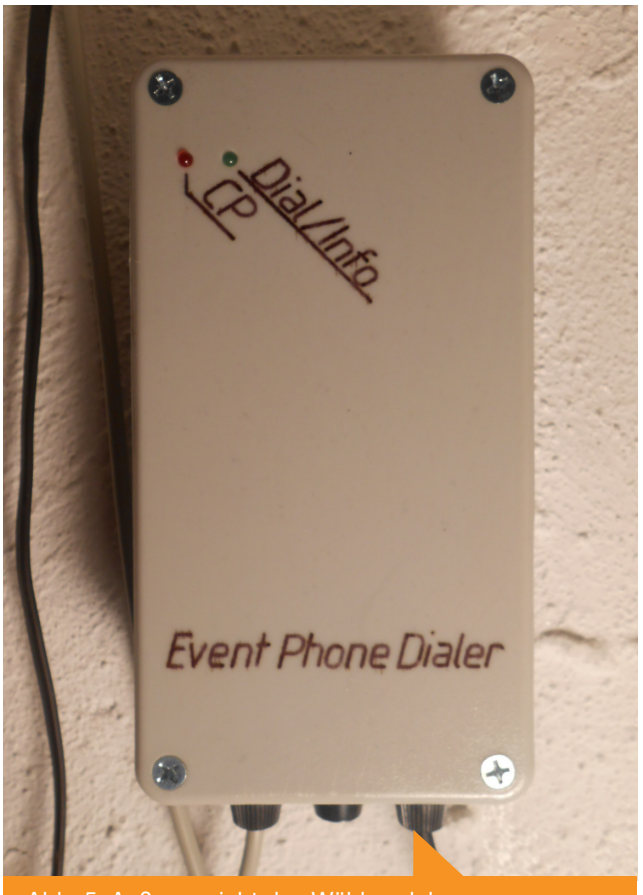


Abb. 5: Außenansicht des Wählmoduls

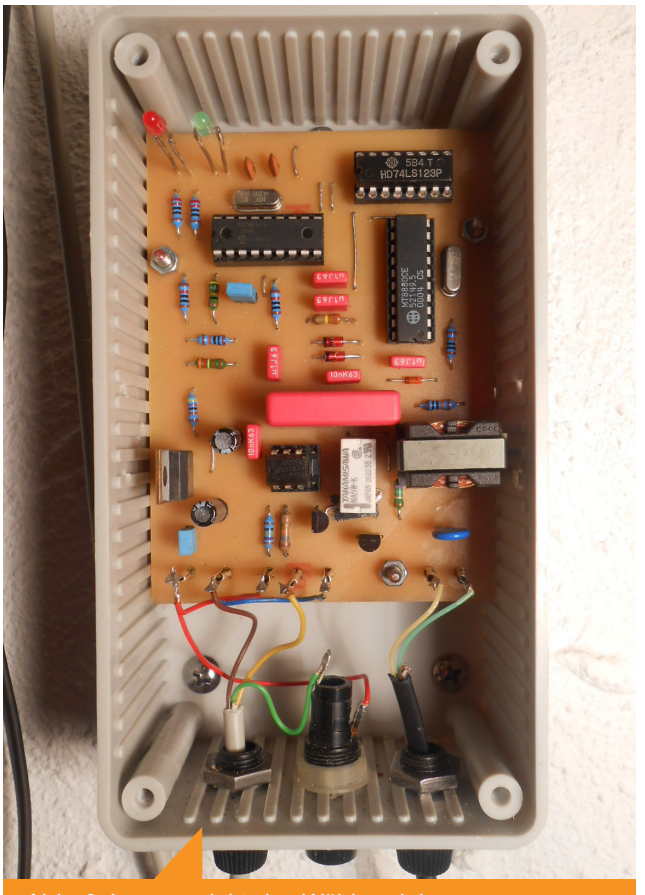


Abb. 6: Innenansicht des Wählmoduls

Zusammenfassung und Ausblick

Das vorgestellte Telefonwählmodul versieht seinen Dienst 24 Stunden am Tag mit meiner Raad-Karcher-Alarmanlage bisher ohne Beanstandung. Meine Katze löst des Öfteren Alarm aus, weil sie doch – entgegen meiner Vermutung – anatomisch etwas größer gebaut ist und den einen oder anderen Bewegungsmelder reizt. Mein offener Kamin sorgt manchmal zusätzlich für einen Rauchalarm. So bekomme ich in aller Regelmäßigkeit ein Feedback über die Funktion meines selbstgebauten Telefonwählmoduls.

Die Telefonwähleinrichtung versieht ihren Dienst genau so, wie sie es soll. Zu nennen sind einige Verbesserungen oder Optimierungen:

- Einbau eines Schaltreglers statt eines Linealreglers
- Stromsparmodus des 16f628A nutzen
- Programmierung der Telefonnummern via RS-232
- Verwendung von SMD-Bauteilen

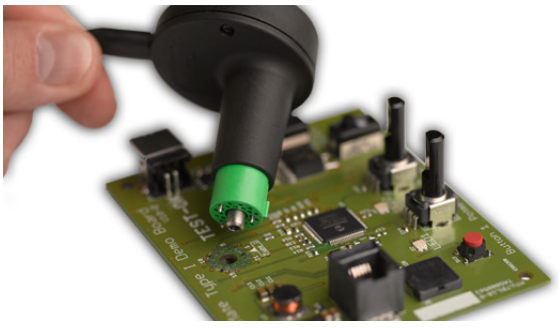
Quellen

- 16f628A Datasheet; http://ww1.microchip.com/downloads/en/DeviceDoc/40044G.pdf
- MT8880C Datasheet; http://www.microsemi.com/index.php?option=com_docman&task=doc_download&gid=127042
- Bauteile; www.reichelt.de; www.conrad.de
- Brenner5; http://www.sprut.de
- Piklab; http://piklab.sourceforge.net
- SDCC; http://sdcc.sourceforge.net
- Alarm Phone Dialer; Alan Parekh

Marktplatz / Neuigkeiten

Die Ecke für Neuigkeiten und verschiedene Produkte

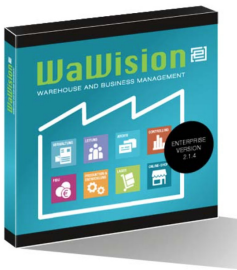
Tragbares Test-System zum funktionalen Testen bestückter Leiterplatten



eC-test-mate ist ein tragbares Test-System zum funktionalen Testen bestückter Leiterplatten, ohne die Notwendigkeit aufwändiger Test-Adapter. Integrieren Sie einfach die eC-test-mate Footprints in Ihr Leiterplatten-Layout, schreiben Sie Ihr Test-Programm, platzieren Sie den eC-test-mate Test-Kopf auf Ihrer Platine und führen Sie den Test aus. Keine Steckverbindungen, keine Test-Aufbauten, nur einfache Kontaktflächen. eC-test-mate ist eine preiswerte Lösung zum Testen kleiner Serien bestückter Leiterplatten.

http://www.eurocircuits.de/test

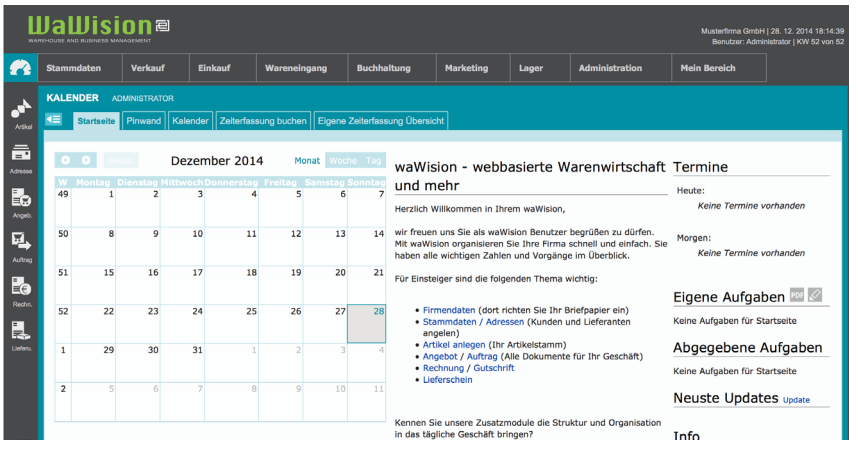
ERP-Software: Open-Source Version von WaWision 3.1 veröffentlicht



Die Software WaWision ist noch ein vergleichbar junger Kandidat eines webbasierten Warenwirtschaftssystems. Ursprünglich war sie als Eigenentwicklung der Firma embedded projects GmbH gestartet, um alle Informationen in einer Firma in einer freien Datenbank (MySQL) zentral zu sammeln: eine Zeiterfassung und Abrechnung für Dienstleistung, Import-Schnittstelle für Online-Shops, Lagerverwaltung, Produktionsplanungs- und Beschaffungstool für den Einkauf u.v.m. Seit der Veröffentlichung 2013 ist daraus mittlerweile ein komplettes ausgereiftes Softwareprodukt geworden. Neben einer Enterprise Version gibt es eine umfassende Open-Source Version, die vor allem Einsteigern und Startups einen günstigen Start in ein ERP System ermöglichen soll.

Dank der vielen Anregungen aus laufenden Projekten und der Open-Source Community entstand nun pünktlich zum Jahreswechsel eine neue Version. Sie erscheint mit einem neuen Layout, mit verbesserten Funktionen wie z.B. erweiterten Lagerprozessen und umfangreicheren Einstellmöglichkeiten. Da WaWision als PHP Quelltext verfügbar ist, können einfach eigene Erweiterungen und Anpassungen vorgenommen werden.

Als Besonderheit bietet WaWision mit Hilfe einer kleinen Adapterbox eine einfache Anbindung von Hardware wie Barcodescanner, Waagen und Kameras an, um so einfach über die Webanwendung auf Geräte zugreifen zu können. Die Box hat einen Netzwerkanschluss und auf der anderen Seite steckt man ein Gerät wie ein Etikettendrucker, eine Waage, Kamera o.ä. an. Aus der Webanwendung kann so direkt mit der Hardware über eine REST API kommuniziert werden. So können z.B. schnell Artikel- oder Seriennummern gedruckt werden, ohne in einen Druckerdialog eines PC gehen zu müssen. Ebenso kann das Gewicht einer Lieferung für die automatische Erstellung der Paketmarke ermittelt werden.



Homepage: http://www.wawision.de/
Download: http://sourceforge.net/projects/wawision/files/?source=navbar

Interesse an einer Anzeige?

info@embedded-projects.net

embedded-projects
JOURNAL
OPEN SOURCE SOFT-AND HARDWARE PROJECTS

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.

Bitte frei machen

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.

Impressum

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

Verteilt durch:

embedded projects GmbH
HARDWARE FOR PROJECTS

Veröffentlichung: 4x / Jahr
Ausgabenformat: PDF / Print

Auflagen Print: 2500 Stk.
Einzelverkaufspreis: 1 €

Layout / Satz: EP
Druck: flyeralarm GmbH
Titelbild: elektor

MIT FLEXIBILITÄT MEHR BEWEGEN.

FLEXIBLE LEITERPLATTEN ONLINE BESTELLEN.

STARR FLEX 4 – Bis zu 4 Lagen auf Anfrage möglich!

LEITON
RECHNEN SIE MIT BESTEM SERVICE

Erfolgreich ist, wer flexibel auf neue Marktanforderungen reagiert. Gefragt sind heute kompakte, komplexe sowie sehr leichte Aufbauten, welche dynamische Biegebelastbarkeit aufweisen und dabei höchste Zuverlässigkeit der elektrischen Verbindungen bieten. Die Lösung lautet **flexible Leiterplatten von LeitOn.** Damit sparen Sie gleich dreimal: **Platzersparnis** durch optimales Anpassen der Baugruppen an die Gehäuse, **Gewichtsersparnis** aufgrund sehr dünner Folien sowie **Kostenersparnis** wegen der Reduktion von Steckverbindungen. Und Sie gewinnen **mehr Flexibilität** dank persönlicher Beratung am Telefon, einem kompetenten Außendienst und Angeboten auch per E-Mail in Windeseile. Sie können bei LeitOn immer mit bestem Service rechnen.

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