Linux-Bedieneinheit

Display, Taster, Echtzeituhr, Buzzer und 16 GPIOs

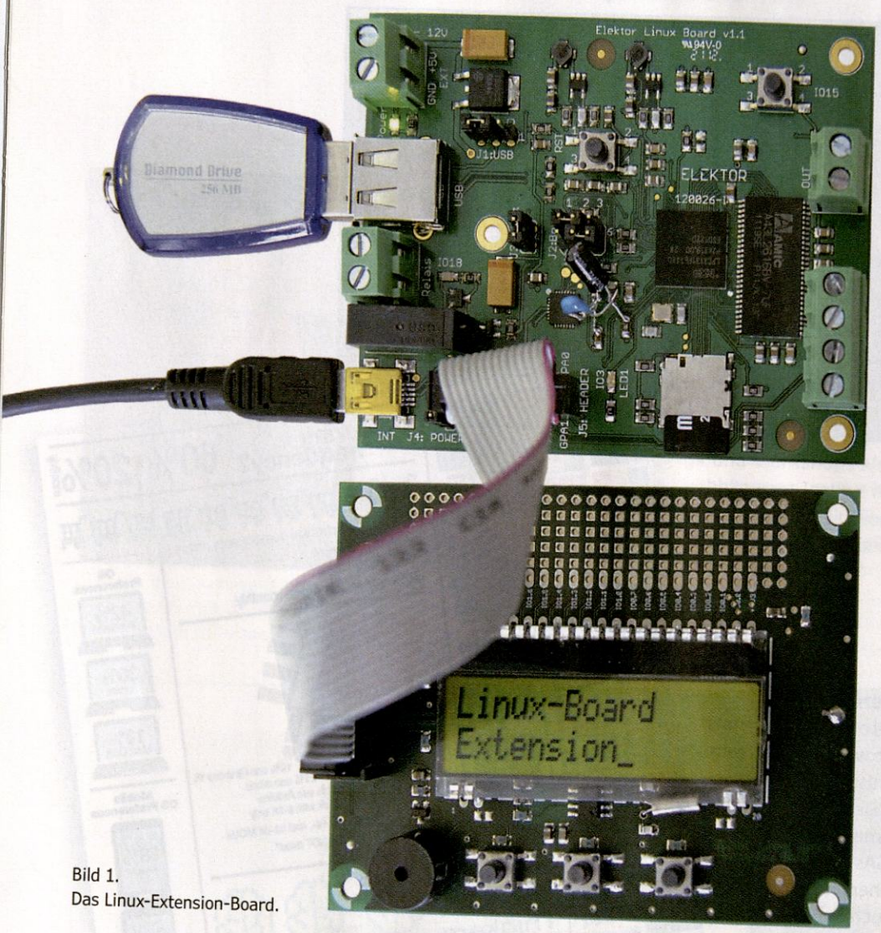

Bild 1.
Das Linux-Extension-Board.

Von **Benedikt Sauter** [1]

Wie man eine Bedienoberfläche mit modernen Webtechnologien für das Elektor-Linux-Board gestalten kann, haben wir bereits gezeigt. Oft gibt es aber Situationen, in denen man in die Steuerung eingreifen muss, aber keinen PC mit Browser zur Hand hat. Da ist man froh, wenn man mit einem Display und ein paar Tastern schnell die wichtigsten Parameter und Zustände aufrufen und bedienen kann. Hier kommt eine Lösung: Eine Erweiterungsplatine für das Elektor-Linux-Board.

Auf dem „Linux-Extension-Board" findet man alles, was man zur Bedienung der unterschiedlichsten Projekte benötigt. Drei Taster dienen zur Eingabe (**Bild 1**). Um Ausgaben darstellen zu können, ist ein 2x16-Zeichen-Display vorhanden. Da sich die Taster direkt darunter befinden, kann man ein kleines Menü umsetzen. Auf dem Board befindet sich zudem ein Piepser für akustische Ausgaben, eine Echtzeituhr (RTC) mit Batterie und eine Porterweiterung, so dass man 16 zusätzliche digitale Ein- und Ausgänge erhält. Und um noch etwas Platz für eigene Erweiterungen zu haben, gibt es ein rund 1,5 x 6 cm großes Lochrasterfeld. Auf dem Elektor-Linux-Board befindet sich mit J5 [2] eine 2x7-Stiftleiste als Erweiterungsstecker (**Bild 2**). Die gleiche Stiftleiste ist auch auf der Erweiterungsplatine zu finden, so dass man beide Boards über ein 14-poliges Flachbandka-

Tabelle 1. Signale am 14-poligen Stecker

Pin	Bezeichnung	Verwendung		Verwendung	Bezeichnung	Pin
1	GPA0	Taster links		Taster Mitte	GPA1	2
3	GPA3	Taster rechts		Buzzer	PWM	4
5	SCL (I2C)	RTC und Portexpander		RTC und Portexpander	SDA (I2C)	6
7	MOSI (SPI)	Display		Display	MISO (SPI)	8
9	System Clock	-		Display	SCK (SPI)	10
11	GPIO14	Display		Display	GPIO11	12
13	3.3V	Versorgung		Versorgung	GND	14

bel verbinden kann. Über das Kabel werden aber nicht nur Signale transportiert; das Linux-Extension-Board wird auch von der Hauptplatine mit 3,3 V versorgt.
In **Tabelle 1** kann man die Belegung der Steckverbinder sehen. Die Erweiterungsplatine lässt sich natürlich auch für andere (eigene) Projekte verwenden, hierbei sei auf den Artikel zur „Embedded Firmware Library" im nächsten Heft hingewiesen.
Das Linux-Extension-Board ist bei Elektor SMD-bestückt und getestet erhältlich, es sind lediglich das LCD und ein paar weitere Bauteile selbst einzulöten [3].

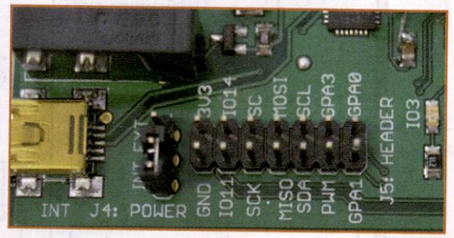

Bild 2.
Über den 14-poligen Stecker auf dem Elektor-Linux-Board schließt man die Erweiterungsplatine an.

Schaltplan

Der Schaltplan ist in **Bild 3** dargestellt. Die Verbindung zum Elektor-Linux-Board wird über K1 realisiert. Da wir nur eine 3,3-V-Stromversorgung haben, mussten natürlich 3,3-V-kompatible Bausteine ausgewählt werden. Bei der Wahl des Displays kam nur eine Ansteuerung per I2C oder SPI in Frage (die meisten alphanumerischen Displays werden ja im 4- oder 8-bit-Modus angesteuert). Die Wahl fiel auf das DOGM162L-A Display [4] von Electronic Assembly, das ein SPI-Interface besitzt.

Da über den Erweiterungsstecker drei analoge Eingänge herausgeführt werden (GPA0, GPA1 und GPA3) und diese auf unserem neuen Board nicht anderweitig verwendet sind, haben wir dort einfach Taster angebunden. Optional könnte man auch Mehrfachschalter anschließen und etwa mit Spannungsteilern verschiedene Positionen erkennen. In unserem Fall haben wir nur den Zustand „High" (A/D-Wandler misst maximale Spannung) und „Low" (A/D-Wandler misst 0). Der letztere Zustand entspricht dabei einem gedrückten Taster. Der Buzzer/Piepser hängt am FET T1, der mit dem PWM-Ausgang des LPC3131 verbunden ist. Dies ermöglicht eine Ausgabe eines klingenden Tons mit konstanter Frequenz. Da der Buzzer nur an 3,3 V hängt, ist die Lautstärke nicht maximal; eigentlich ist eine Versorgung mit 8..15 V vorgesehen (laut Datenblatt [5]).
Auf dem Board befindet sich mit IC2 zusätzlich noch eine Echtzeituhr MCP79401 [6]. Diese ist mit einer kleinen Knopfzelle gepuffert, so dass die Uhrzeit auch im ausgeschalteten Zustand des Geräts nicht verloren geht.
Und schließlich gibt es noch einen Portexpander PCA9555 (IC1) [7] von NXP für bis zu 16 zusätzliche digitale Ein- und Ausgänge, der über I2C angesteuert wird. An die GPIO-Pins kann man beispielsweise LEDs für Zustandsanzeigen und/oder weitere Taster (oder gar eine Tastaturmatrix) anschließen.

Display

Um das Display einfach aus eigenen Anwendungen ansteuern zu können, haben wir ein kleines Programm für die Kommandozeile in C geschrieben. Das Programm kann man sich als Quelltext oder fertig übersetzt herunterladen und auf die SD-Karte kopieren. Am einfachsten macht man dies mit Hilfe eines USB-Sticks. Zunächst lädt man sich die Datei [8](URL in eine Zeile ohne Umbruch eingeben) auf den Entwicklungs-Computer und kopiert diese Datei auf einen USB-Stick. Den USB-Stick steckt man dann an das Elektor-

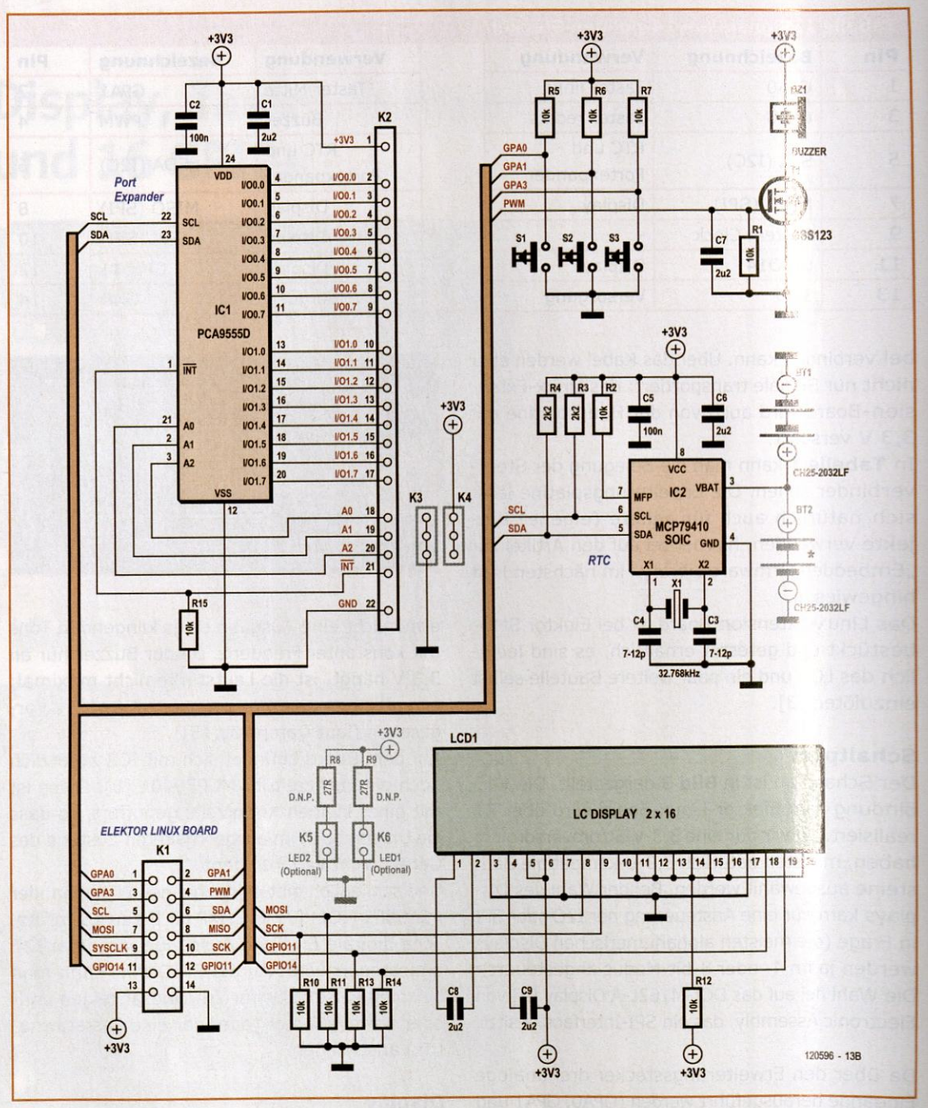

Bild 3.
Schaltplan des Linux-Extension-Boards.

Linux-Board (**Bild 4**). Auf der Konsole sollte eine Ausgabe nach dem Anstecken des Sticks erscheinen. Jetzt muss man den USB-Stick nur noch in das Dateisystem einhängen:

```
mount /dev/sda1 /mnt/
```

Um die Anwendung in Linux immer (unabhängig ob der Stick angeschlossen ist oder nicht) verwenden zu können, kopiert man diese noch in das lokale Dateisystem auf die SD-Karte.

```
cp /mnt/dog-app /usr/local/bin
```

Für einen Test des Programmes kann man es kurz auf der Konsole starten. Ein Aufruf von

```
dog-app -h
```

gibt den Hilfetext des Programmes aus.
Um das erste Mal Text auf dem Display anzuzeigen, muss die Erweiterungsplatine natürlich mit dem Elektor-Linux-Board verbunden sein (die rote Markierung am Flachbandkabel muss sich auf der

Seite von GPA1/GPA0 befinden). Mit folgendem Befehl kann man Text ausgeben:

```
dog-app -n -w "Hello Elektor"
```

(hier immer Standard-Anführungszeichen benutzen). Auf dem Display sollte nun „Hello Elektor" erscheinen.

Möchte man etwas in die zweite Zeile schreiben, muss man verstehen, wie das Display arbeitet. Jede Zeile eines Displays besteht aus mehreren Segmenten. Man hat nun die Möglichkeit, den Cursor auf ein bestimmtes Segment zu setzen. Das erste Segment in der ersten Zeile entspricht dem Offset 128. Das erste Segment der zweiten Zeile spricht man über den Offset 192 an. Mit dem folgenden Befehl schreibt man in die zweite Zeile:

```
dog-app -o "192" -w "Hello 2"
```

Taster

Die Taster sind an die A/D-Wandler-Eingänge des LPC3131 angeschlossen. Wie die A/D-Wandler verwendet werden, wurde bereits in der Artikelreihe zum Elektor-Linux-Board beschrieben. Mit dem folgenden Befehl erhält man Zugriff auf den linken Taster:

```
echo 0 > /dev/lpc313x_adc
```

Liest man die Datei /dev/lpc313x_adc (bzw. gibt man diese mit dem cat-Befehl aus) erhält man im offenen Zustand einen sehr großen Wert (nahezu 1023).

```
root@gnublin:~# cat /dev/lpc313x_adc

0x3ff
```

Ist der Taster gedrückt, erhält man 0 als Ausgabe:

```
root@gnublin:~# cat /dev/lpc313x_adc

0x000
```

Der Taster in der Mitte muss mit …

```
echo 1 > /dev/lpc313x_adc
```

… aktiviert werden. Und der Taster ganz rechts mit:

```
echo 3 > /dev/lpc313x_adc
```

Buzzer / Piepser

Einen Ton kann man über das Dateisystem ganz einfach erzeugen …

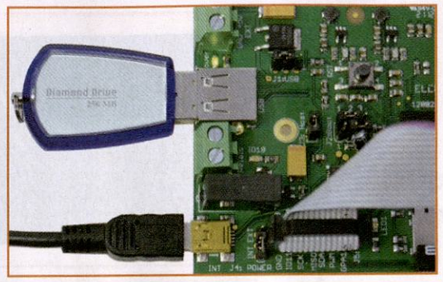

Bild 4.
Das Elektor-Linux-Board mit USB-Stick.

```
echo ff > /dev/lpc313x_pwm
```

… und abschalten:

```
echo 00 > /dev/lpc313x_pwm
```

Will man Piep-Signale generieren, kann man auf der Konsole mit usleep und der Angabe der Länge Pausen zwischen den Tönen einfügen.

Echtzeituhr

Für die RTC benötigt man einen eigenen Kernel-Treiber. Im Standard-Dateisystem ist dieser bereits vorhanden und muss nur noch geladen werden:

```
modprobe rtc-mcp7940
```

Soll der Treiber automatisch beim Start des Systems geladen werden, kann man dies in /etc/modules in einer eigenen Zeile fest hinterlegen:

```
rtc-mcp7940
```

Dem Kernel muss man jetzt noch mitteilen, welche Adresse der RTC-Baustein auf dem I2C-Bus hat. Dies macht man wie folgt:

```
echo mcp7940 0x6f > /sys/bus/i2c/devices/
i2c-1/new_device
```

Als Nächstes muss man kurz nachsehen, was für eine Major-Nummer durch den Kernel vergeben worden ist, um die Gerätedatei entsprechend anlegen zu können.

```
cat /proc/devices
```

Standardmäßig sollte die Nummer 251 vergeben worden sein. Die Gerätedatei muss man nur einmal anlegen:

```
mknod /dev/rtc0 c 251 0
```

Nun setzt man das Datum und die Uhrzeit von Hand im Format [MMDDhhmm[[CC]YY]], zum Beispiel mit …

```
case "$1" in
        start)
                # start here
                if [ ! -d /sys/bus/i2c/devices/1-006f/ ]; then
                        echo mcp7940  0x6f > /sys/bus/i2c/devices/i2c-1/new_device
                fi
                if [ ! -c /dev/rtc0 ]; then
                        mknod /dev/rtc0 c 251 0 2>/dev/null 1>&2
                fi
                # end
                if [ "$VERBOSE" != no ]
                then
                        echo "System time was `date`."
                        echo "Setting the System Clock using the Hardware Clock as reference..."
                fi
```

Bild 5.
Setzen von Uhrzeit und Datum beim Systemstart.

```
Elektor GNUBLIN ELDK (Built by Poky 5.0) 5.0 gnublin ttyS0
http://www.gnublin.org

gnublin login: root
root@gnublin:~# modprobe pca953x
root@gnublin:~# echo 98 > /sys/class/gpio/export
root@gnublin:~# echo out > /sys/class/gpio/gpio98/direction
root@gnublin:~# echo 1 > /sys/class/gpio/gpio98/value
root@gnublin:~#
```

Bild 6.
Mit Hilfe des Kernel-Moduls „pca953x“ kann man die Portexpander-GPIOs ganz einfach ansprechen.

Tabelle 2. Ansprechen der PCA9555-Pins mit dem Export-Befehl

Pin-Bezeichnung	Wert für /sys/class/gpio/export
IO0_0	98
IO0_1	99
IO0_2	100
IO0_3	101
IO0_4	102
IO0_5	103
IO0_6	104
IO0_7	105
IO1_0	106
IO1_1	107
IO1_2	108
IO1_3	109
IO1_4	110
IO1_5	111
IO1_6	112
IO1_7	113

```
date 122014342012
```

was zu folgendem Ergebnis führt:

```
Thu Dec 20 14:34:00 UTC 2012
```

Um Uhrzeit und Datum in den Baustein zu übertragen, ruft man einmalig:

```
hwclock -w
```

auf. Möchte man sich die Uhrzeit und das Datum vom Baustein abholen, geht das wie folgt:

```
hwclock -r
```

Als Ausgabe erscheint:

```
Thu Dec 20 14:34:17 2012 0.000000 seconds
```

Ist die Batterie im Halter, so bleibt die Uhrzeit auch nach dem Ab- und Anstecken der Stromversorgung erhalten. Der Batteriehalter kann übrigens sowohl auf dem Board als auch auf der Unterseite montiert werden (siehe Schaltplan). Um die Uhrzeit des Systems beim Start automatisch zu setzen, muss man die Datei „/etc/init.d/hwclock.sh“ wie in **Bild 5** gezeigt erweitern. Wichtig ist dabei, dass das Modul rtc-mcp7940 entsprechend in /etc/modules eingetragen worden ist.

Portexpander

In Folge 7 der Artikelreihe [9] haben wir den Portexpander bereits angesprochen, direkt mit den I2C-Tools und auch in C.

Mit Hilfe des Kernel-Moduls „pca953x“ kann man die Portexpander-GPIOs ganz einfach so wie die Hardware-GPIOs des LPC3131 ansprechen [10]. In **Bild 6** sieht man die dazu nötigen Befehle. In **Tabelle 2** stehen die entsprechenden Werte für den Export-Befehl.

Im nächsten Heft werden wir eine kleine Anwendung des Erweiterungsboards vorstellen. Mit den Tastern und dem LCD realisieren wir ein Menü, das man sich natürlich für eigene Zwecke anpassen kann.

(120518)

Weblinks

[1] sauter@embedded-projects.net
[2] www.elektor-magazine.de/120181
[3] www.elektor-magazine.de/120596
[4] www.lcd-module.de/pdf/doma/dog-m.pdf
[5] http://goo.gl/ToKZZ
[6] http://ww1.microchip.com/downloads/en/DeviceDoc/22266D.pdf
[7] www.ti.com/lit/ds/symlink/pca9555.pdf
[8] http://gnublin.googlecode.com/git/lpc3131/gnublin_package/src/gnublin-dogm/usr/bin/gnublin-dogm
[9] www.elektor-magazine.de/120518
[10] www.elektor-magazine.de/120146

Stückliste

Widerstände:
R1,R2,R5,R6,R7,R10,R11,R12,R13,R14,R15 = 10 k
R3,R4 = 2k2
R8,R9 = 27 Ω (optional)

Kondensatoren:
C1,C6..C9 = 2µ2
C2,C5 = 100 n
C3,C4 = 7..12 p

Halbleiter:
T1 = BSS123
IC1 = PCA9555D SO-24
IC2 = MCP79410 SOIC-8

Außerdem:
X1 = Uhrenquarz 32,768 kHz
K1 = Wannenstecker 14-polig
K2 = Stiftkontaktleiste 3-polig oder optional 22-polig
K3,K4 = Stiftkontaktleiste 3-polig
K5,K6 = Stiftkontaktleiste 2-polig (optional)
BT1 oder BT2 = Batteriehalter CH25-2032LF
BZ1 = Buzzer
S1,S2,S3 = Drucktaster 9314 SMD
LCD1 = 2x16 LCD Electronic Assembly DOGM162L-A
Platine

Modul/Kit 120596-91:
SMD-bestückte und getestete Platine + LCD1, X1, K1..K4, BZ1, BT1 zur Selbstbestückung