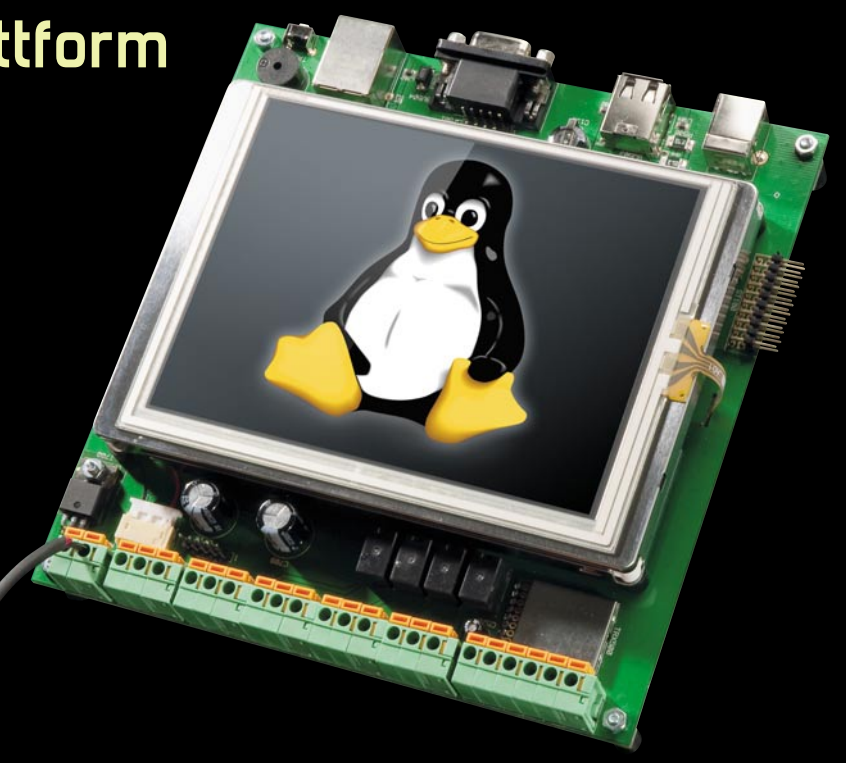
Universal-embedded Linux-PLattform



Linux-Control-Unit LCU 1, Teil 2

Mit der Linux-Control-Unit stellen wir eine komplette Hard- und Software-Plattform mit zahlreichen unterschiedlichen Schnittstellen, Eingängen, Ausgängen und Speichermöglichkeiten vor. Derjenige, der sich mit dem Thema Embedded-Linux-Systeme beschäftigen möchte, wird damit in die Lage versetzt, schnell und preiswert zu einer eigenen, kompletten Lösung zu kommen, vornehmlich im Steuerungsbereich. Im zweiten Teil zeigen wir die Möglichkeiten der Verbindungsaufnahme zur LCU 1 auf.

Boot-Reihenfolge

Für das Verständnis der Prozesse beim Start der LCU 1 zeigen wir hier den Ablauf des Boot-Vorgangs detailliert auf.
Nach dem Einschalten wird der ROM-Bootloader des AT91SAM9261 gestartet. Dieser durchsucht den angeschlossenen Speicher nach ausführbaren Programmen und startet den von Atmel im Quellcode bereitgestellten Bootloader „at91bootstrap" aus dem DataFlash.
Dieses wiederum initialisiert den SDRAM und lädt „u-boot" von Adresse 0x00008400 aus dem DataFlash. „u-boot" initialisiert danach die weiteren Subsysteme, lädt den Kernel aus dem DataFlash von Adresse 0x00042000 und startet diesen. Dabei übergibt „u-boot" eine Kernel-Kommandozeile, in der u. a. steht, woher das Root-Dateisystem zu beziehen ist. Der Kernel initialisiert das System und mountet das Root-Dateisystem aus dem NAND-Flash. Das heißt, dass ein Zugriff auf Dateiebene möglich ist.
Anschließend übergibt der Kernel die Kontrolle an den Initialisierungsprozess - /sbin/init. „init" startet nun weitere Programme, die in der Datei „/etc/inittab" angegeben sind. Zum Schluss wird eine „Shell" auf der seriellen Konsole (Zugriff mit „/dev/ttyS0" via Debug-UART) gestartet, an der sich der Nutzer anmelden kann.
Nach dem Booten laufen u. a. der Web-Server „lighttpd" und der SSH-Server „dropbear", die die Verbindung über Netzwerk oder USB ermöglichen.
Diese Möglichkeiten der Verbindungsaufnahme wollen wir gleich detaillierter betrachten.
Ein Hinweis noch vorab. Auf dem Board befindet sich ein Jumper, über den der DataFlash abschaltbar ist. In diesem Fall geht das Board in den CPU-internen Firmware-Update-Modus.

Verbindungen zum Zielsystem

Die Verbindung zur LCU 1 kann natürlich über „ssh" unter

Linux, aber auch unter MS Windows vorgenommen werden. Hier leistet das bekannte superkompakte Programm „PuTTY" gute Dienste. „PuTTY" ist ein kleiner Telnet-Client für Windows, der über das SSH-Protokoll einen sicheren Datenaustausch mit entfernten Rechnern ermöglicht (siehe auch „Elektronikwissen"). Damit ist eine relativ einfach aufzubauende Verbindung zum SSH-Server auf der LCU 1 möglich. Über das Terminal-Fenster können direkt Befehle in das System des entfernten Computers geschrieben und dort ausgeführt werden. Die Ausführung wird wiederum im PuTTY-Fenster angezeigt.
Übrigens – Windows-Nutzer können ein Linux-System quasi parallel als „Virtual Box" betreiben, denn ohne geht es nicht, die Entwicklungsumgebung basiert auf einem Linux-System, dem wir uns im dritten Teil des Artikels widmen. „Virtual Box" von Oracle/Sun (Abbildung 4) ist eine Freeware, die es ermöglicht, ein zweites, vom bisher installierten Betriebssystem – z. B. MS Windows oder Mac OS X – unabhängig arbeitendes Gastsystem, in unserem Falle also Linux, zu betreiben. Auch umgekehrt ist dies möglich. Dabei kann das Gastsystem die vorhandene Hardware virtuell nach entsprechender Konfiguration nutzen, z. B. RAM oder Laufwerke.
Bei der Anmeldung an der LCU 1 ist als Benutzer-Kennwort „root" und als Passwort „lcu" zu benutzen.

Serielle Verbindung

Zum Herstellen der seriellen Verbindung sind für den seriellen Port die Verbindungsparameter:
115.200 Baud, 8 Datenbits, No Parity, 1 Stoppbit
einzustellen. Je nach Betriebssystem ist darauf zu achten, dass die entsprechende serielle Schnittstelle auch im System so konfiguriert ist. Die serielle asynchrone Zweidraht-Schnittstelle (UART) führt TTL-Pegel! Sie kann entweder über einen RS232-Pegelwandler oder einen UART-zu-USB-Wandler wie den UO 100 oder den UM 2102 (Abbildung 5) erfolgen, indem auf dem PC ein virtueller COM-Port eingerichtet wird, der wiederum via PuTTY oder Hyperterm anzusprechen ist.

USB-Verbindung

Die Verbindung per USB erfolgt über den Device-USB-Port der LCU 1. Diese ist als Netzwerkgerät mit der festen IP-Adresse 10.101.81.52 oder (nur unter Windows) als „lcu.usb" via ssh (Linux) oder PuTTY (Windows) anzusprechen.

Netzwerk-Verbindung

Beim Anschluss an ein Netzwerk über den Ethernet-Port der LCU 1 bezieht diese eine IP-Adresse per DHCP vom Netzwerk-Router. Die IP-Adresse kann, je nach Routertyp unterschiedlich, über dessen Netzwerk-Dialog eingesehen werden, z. B. bei der verbreiteten Fritz-Box via „fritz.box -> Netzwerk -> bekannte Netzwerkgeräte - LAN".
Unter Linux erfährt man die Adresse via „ifconfig"-Befehl. Beispiel (nur relevante Teile aufgeführt):
ifconfig
eth0 Link encap:Ethernet HWaddr 3A:1F:34:08:54:54
inet addr:192.168.1.179 Bcast:192.168.1.255 Mask: 255.255.255.0
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1....
Hier ist die IP-Adresse 192.168.1.179. Sie ist unter Linux mit

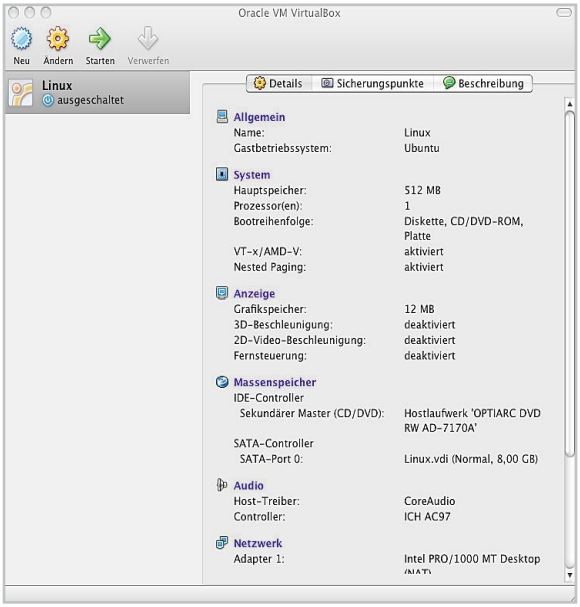


Bild 4: Die Oracle VM Virtual Box, hier in der Version Mac OS, macht eine einfache zusätzliche Linux-Installation auf einem unter einem anderen Betriebssystem laufenden (und parallel weiter nutzbaren) Rechner möglich.

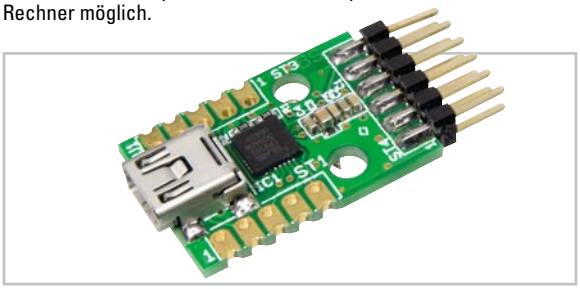


Bild 5: Geeignet für die Verbindung über die UART-Schnittstelle: der UM 2102 von ELV

ssh und unter Windows via „PuTTY" zu erreichen. Spricht man die LCU 1 über einen Web-Browser an, erscheint ein Dialog für ein über diesen Weg sehr einfach auszuführendes Firmware-Update. Hier ist lediglich der Pfad zur Update-Datei einzustellen.

Erste Kontakte

Nach der Herstellung einer Verbindung zur LCU 1 kann man von der Kommandozeile aus die Zustände der digitalen Eingänge sowie die Spannungswerte an den analogen Eingängen abfragen, die Relais schalten und sich die Partitionen des Flash-Speichers anzeigen lassen.

Werte an den analogen Eingängen abfragen
Es wird die Spannung in mV zurückgemeldet.

Abfrage Analog-Eingang 1:

cat /sys/class/hwmon/hwmon0/device/in0_input
Ergebnis z. B.: 0

Abfrage Analog-Eingang 2:

cat /sys/class/hwmon/hwmon0/device/in1_input
Ergebnis z. B.: 1500

Elektronikwissen – Secure Shell/SSH

Secure Shell ist ein Netzwerkprotokoll, das eine sichere Verbindung zu einem entfernten Rechner aufbaut, so dass man diesen auch in ungesicherten Netzen sicher erreichen kann. Es wird mit Hilfe eines SSH-Client-Programms, z. B. PuTTY für MS Windows (siehe Screenshot rechts), eine Terminalverbindung aufgebaut, über die man Kommandos auf dem entfernten Rechner direkt von dem Rechner aus eingeben und ausführen lassen kann, auf dem der SSH-Client installiert ist. Ebenso sind die Ausgaben der Kommandozeile des entfernten Rechners auf dem Client-Rechner (Konsole) sichtbar. Auf dem entfernten Rechner muss dazu ein SSH-Server installiert und aktiv sein, der sich gegenüber dem SSH-Client mit einem Schlüssel authentifiziert, z. B. einem Kennwort oder einem Identitätsschlüssel.
Bei Linux-Distributionen ist der SSH-Client meist bereits integriert und wird (außer auf Live-Distrubutionen wie z. B. Koppix) automatisch eingerichtet. Hier ruft man den Prozess einfach im Terminal-Fenster mit dem Befehl „ssh Benutzer@Rechnername" und Passwort auf.
Via SSH sind auch sichere Datei-Transfers zwischen den Rechnern möglich.

PuTTY für MS Windows, hier beim Einrichten einer Session via USB

Relais/Piezo-Signalgeber schalten
Kommando „1" bedeutet: Relais angezogen/Piezo-Signalgeber ein; Kommando „0": Relais abgefallen/Piezo-Signalgeber aus:

```
# echo -n 1 > /dev/inout/out0
# echo -n 0 > /dev/inout/out0
# echo -n 1 > /dev/inout/out1
# echo -n 1 > /dev/inout/out2
# echo -n 1 > /dev/inout/out3
# echo -n 1 > /dev/inout/out4
# echo -n 0 > /dev/inout/out4
# echo -n 0 > /dev/inout/out3
# echo -n 0 > /dev/inout/out2
# echo -n 0 > /dev/inout/out1
```

„out0" bis „out3" sind die vier Relais, „out4" ist der Piezo-Signalgeber.

Digitale Eingänge abfragen
Mit dieser Sequenz lässt sich der aktuelle Zustand („high"oder „low") der digitalen Eingänge darstellen.
Ist die Abfrage gestartet, erscheint in der Folge jede Zustandsänderung des betreffenden Eingangs:
0 = „high", 1 = „low"

```
# cat /dev/inout/in0
# cat /dev/inout/in1
# cat /dev/inout/in2
# cat /dev/inout/in3
```

Der Abbruch der Abfrage erfolgt mit Ctrl-C.

Touchscreen kalibrieren
Hierüber wird die Routine zur Kalibrierung des Touchscreens aufgerufen:

```
# ts_calibrate
```

Flash-Partitionen anzeigen lassen
Mit diesem Kommandozeilenbefehl erfolgt der Aufruf der Aufteilung des Flash-Speichers (DataFlash + NAND-Flash):

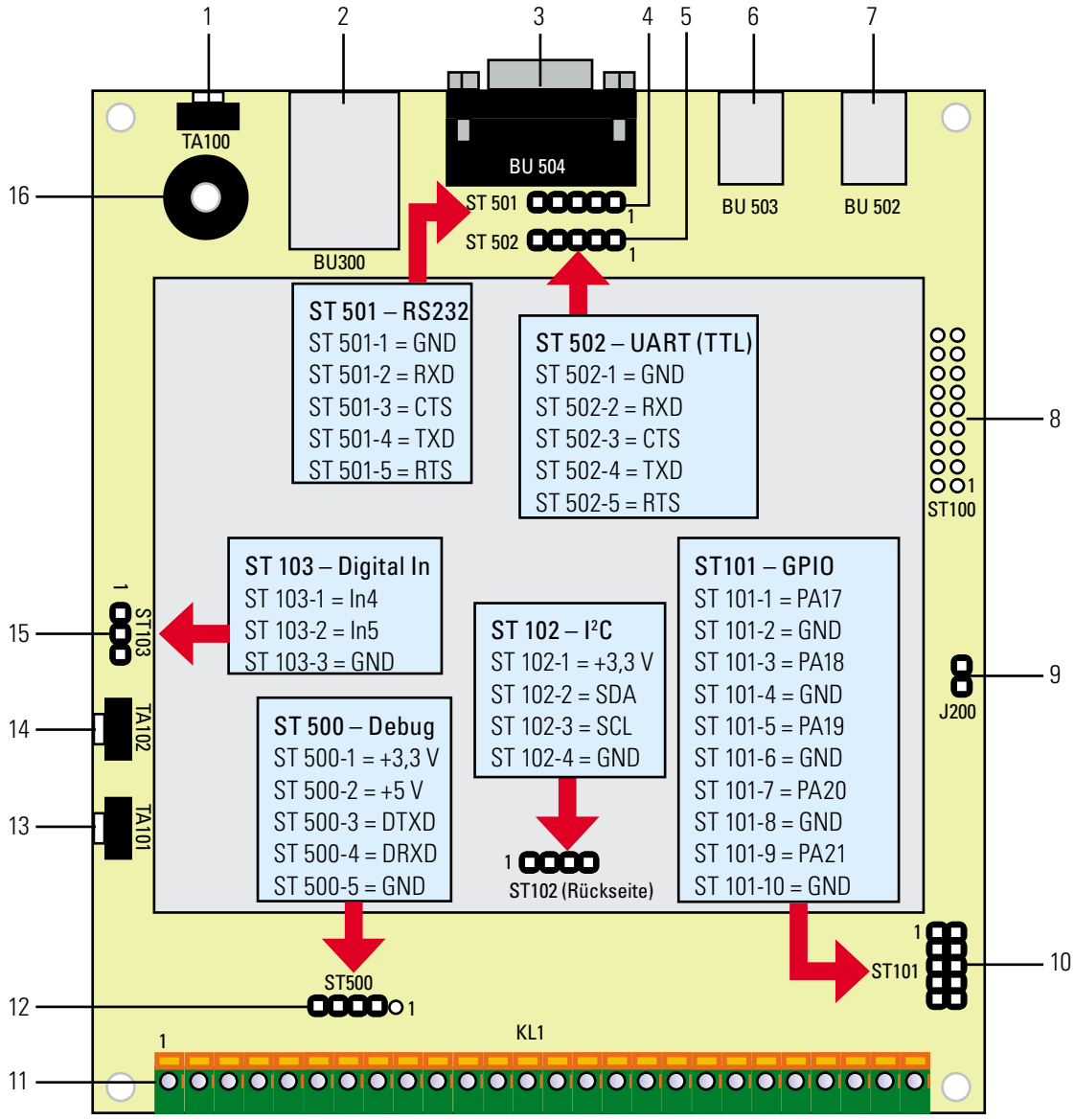
```
# cat /proc/mtd
```

Es erscheint die folgende Partitionstabelle:

```
dev:    size     erasesize  name
mtd0: 08000000 00020000 „Root"
mtd1: 08000000 00020000 „User"
mtd2: 00042000 00000420 „u-boot"
mtd3: 00210000 00000420 „kernel"
mtd4: 005ee000 00000420 „free"
```

Zur Erklärung:
Die Spalte „dev" gibt die Gerätedatei zum Ansprechen der jeweiligen Partition an, z. B. /dev/mtd0 bzw. /dev/mtdblock0.
Die Bezeichnung „mtd" bedeutet „Memory Technology Devices" und ist ein Oberbegriff für Flash-Bausteine.
Die Partitionen „mtd0" und „mtd1" sind die beiden Partitionen des NAND-Flash, die Partitionen „mtd2", „mtd3" und „mtd4" die des DataFlash.
Die Partition „mtd0" ist das Root-Dateisystem und standardmäßig nur auslesbar (Read only). Sie ist als Wurzelverzeichnis unter „/" eingebunden.
Die Partition „mtd1" ist mit Nutzerdaten beschreibbar. Sie ist unter „/usr/local" in den Verzeichnisbaum eingebunden.
Die Partition „mtd2" ist die DataFlash-Partition für den Bootloader, sie ist unter Linux nicht eingebunden.
Die Partition „mtd3" ist die DataFlash-Partition für den Kernel, sie ist unter Linux nicht eingebunden, denn der Kernel läuft im SDRAM.
Die Partition „mtd4" ist für den Benutzer verfügbarer Platz im DataFlash. Hier kann ein eigenes Dateisystem eingerichtet werden. Im dritten Teil beschreiben wir die Entwicklungsumgebung und eine Demo-Applikation für die LCU 1.

LCU-1- Anschlüsse, E/A-Elemente



1 - Reset-Taster
2 - Netzwerkbuchse RJ45
3 - RS232-Schnittstelle* DB9
4 - RS232-Schnittstelle*, Stiftleiste
5 - UART-Schnittstelle* TTL, Stiftleiste
6 - USB-Host-Port 1/2
7 - USB-Device-Port
8 - JTAG-Programmierschnittstelle (nicht bestückt)
9 - Jumper für Abschalten des DataFlash (CS an/aus**)
10 - GPIOs für individuelle Nutzung, Stiftleiste
11 - Klemm-Anschlussleiste
12 - Debug-Schnittstelle***, Stiftleiste
13 - Taster, als zusätzlicher Digital-Eingang nutzbar (In4)
14 - Taster, als zusätzlicher Digital-Eingang nutzbar (In5)
15 - Digital-Eingänge In4/5, parallel zu TA 101/102
16 - Piezo-Signalgeber (out4)

Belegung KL 1

1 - +UB: 10–30 V_{DC}	9 - Relais 1 (out3), NO	18 - Relais 4 (out0), NO
2 - GND	10 - Relais 1, NC	19 - Relais 4, NC
3 - Analog-Eingang 1, in0	11 - Relais 1, COM	20 - Relais 4, COM
4 - Analog-Eingang 2, in1	12 - Relais 2 (out1), NO	21 - GND
5 - GND analog	13 - Relais 2, NC	22 - GND
6 - RS485 A	14 - Relais 2, COM	23 - Digital In1
7 - RS485 B	15 - Relais 3 (out2), NO	24 - Digital In2
8 - GND	16 - Relais 3, NC	25 - Digital In3
	17 - Relais 3, COM	26 - Digital In4

* Unter Linux ansprechbar mit /dev/ttyS1

** Schaltet Chip Select des DataFlash. Bei Abschalten des DataFlash wird nur der ROM-Bootloader gestartet, damit ist über SAM-BA das Übertragen von Firmware möglich

*** Unter Linux ansprechbar mit /dev/ttyS0