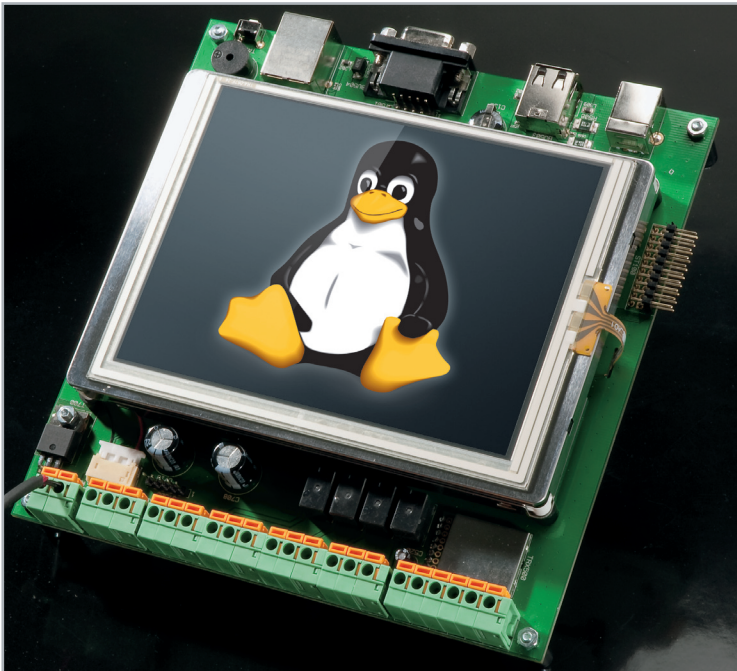


# Universal-Embedded Linux-Plattform

## Linux-Control-Unit LCU1

---

### Bedienungsanleitung



ELV AG • PF 1000 • D-26787 Leer  
Telefon 0491/6008-88 • Telefax 0491/6008-244

Bitte lesen Sie diese Bedienungsanleitung vor der Installation und Inbetriebnahme komplett und bewahren Sie die Bedienungsanleitung für späteres Nachlesen auf. Wenn Sie das Gerät anderen Personen zur Nutzung überlassen, übergeben Sie auch diese Bedienungsanleitung.

**ELV - [www.elv.com](http://www.elv.com) - Art.-Nr. 99377**

## **Inhalt:**

|       |  |    |
|-------|--|----|
| 1.    | Beschreibung und Funktion.....                       | 3  |
| 2.    | Sicherheits-, Betriebs- und Wartungshinweise.....    | 3  |
| 3.    | Anschlüsse, E/A-Elemente.....                        | 5  |
| 4.    | Hinweise zu Linux-Entwicklungstools, Bootloader..... | 6  |
| 5.    | Boot-Reihenfolge.....                                | 7  |
| 6.    | Verbindungen zum Zielsystem.....                     | 8  |
| 7.    | Erste Kontakte .....                                 | 9  |
| 8.    | Die Entwicklungsumgebung .....                       | 11 |
| 9.    | Die Demo-Applikation.....                            | 14 |
| 10.   | Weblinks, Quellen.....                               | 16 |
| 11.   | Technische Daten, Entsorgung .....                   | 16 |
| 12.   | Anhang.....  | 17 |
| 12.1. | Blockschaltbild der LCU1 .....                       | 17 |
| 12.2. | Auszüge aus dem Demoprogramm „demoapp“.....          | 18 |
| 12.3. | Secure Shell/SSH.....                                | 19 |

---

1. Ausgabe Deutsch 08/2011

Dokumentation © 2011 ELV Ltd. Hongkong

Alle Rechte vorbehalten. Ohne schriftliche Zustimmung des Herausgebers darf dieses Handbuch auch nicht auszugsweise in irgendeiner Form reproduziert werden oder unter Verwendung elektronischer, mechanischer oder chemischer Verfahren vervielfältigt oder verarbeitet werden.

Es ist möglich, dass das vorliegende Handbuch noch drucktechnische Mängel oder Druckfehler aufweist. Die Angaben in diesem Handbuch werden jedoch regelmäßig überprüft und Korrekturen in der nächsten Ausgabe vorgenommen. Für Fehler technischer oder drucktechnischer Art und ihre Folgen übernehmen wir keine Haftung. Alle Warenzeichen und Schutzrechte werden anerkannt. Printed in Hong Kong. Änderungen im Sinne des technischen Fortschritts können ohne Vorankündigung vorgenommen werden.

103516Y2011V1.0

## 1. Beschreibung und Funktion

Mit der Linux-Control-Unit steht eine komplette Hard- und Software-Plattform mit zahlreichen Schnittstellen, Ein- und Ausgängen und Speichermöglichkeiten zur Verfügung, die jeden, der sich mit dem Thema Embedded-Linux-Systeme beschäftigen möchte, in die Lage versetzt, schnell und preiswert zu einer eigenen kompletten Lösung, vornehmlich im Steuerungsbereich, zu kommen.

Die Hardware-Plattform der LCU 1 bietet dem ambitionierten Software-Entwickler eine universelle Grundlage mit vielen Optionen für die Entwicklung eigener Anwendungen. Sie kann sowohl als Entwicklungssystem als auch direkt als Embedded-System laufen, z. B. als Haustechnik-Rechner mit integrierter Anzeige- und Bedieneinheit, der zentral mit Aktoren, Sensoren und Bedienelementen zusammenarbeitet.

Via SSH ist eine sichere Terminalverbindung möglich, so dass die LCU 1 auch als abgesetztes Gerät betrieben und erreicht werden kann. Für erste Versuche wird eine Demo-Applikation inkl. Quelltext mitgeliefert.

### Eigenschaften und Ausstattung:

- Betriebssystem: Linux-2.6-Distribution, Start mit Bootloader „U-Boot“
- CPU: Atmel-ARM AT91SAM9261, 190 MHz
- Speicher: 64 MB SDRAM, 256 MB NAND-Flash, 8 MB Data-Flash
- Display: 14,5-cm-Touchscreen, 320 x 240 Pixel, Color
- 6 Digital-Eingänge, 2 Analog-Eingänge, 4 Relais-Schaltausgänge über Klemmleiste
- 2 universell einsetzbare Taster, 1 Signalgeber
- RS232 (DB9/Stiftleiste)
- 2x USB-Host-Port, 1x USB-Device-Port,
- 1x RS485-Schnittstelle (Klemmleiste), 5x GPIO (Stiftleiste), 1x Debug-Schnittstelle (UART, Stiftleiste)
- weiter Betriebsspannungsbereich: 10–30 VDC

### Lieferumfang:

- 1 Linux-Control-Unit LCU1, betriebsfertig
- Software-Ausstattung: Linux, IDE, Bootloader, Tools, Demo-Applikation (inkl. Quelltext)

## 2. Sicherheits-, Betriebs- und Wartungshinweise

- Bitte lesen Sie diese Anleitung vor der ersten Inbetriebnahme komplett und sorgfältig, sie enthält zahlreiche Hinweise zum bestimmungsgemäßen Gebrauch des Gerätes.
- Bei Zweifel über die Arbeitsweise, die Sicherheit oder den Anschluss des Gerätes eine Fachkraft oder unseren Service kontaktieren.
- Das Gerät nicht verwenden, wenn es erkennbare Schäden bzw. eine Funktionsstörung aufweist. Im Zweifelsfall das Gerät von einer Fachkraft oder unserem Service prüfen lassen.

- Verpackungsmaterial nicht achtlos liegen lassen. Plastikfolien/-tüten, Styroporsteile etc. könnten für Kinder zu einem gefährlichen Spielzeug werden.
- Das Gerät darf nicht verändert oder umgebaut werden.
- Nehmen Sie keine eigenen Reparaturversuche vor, sondern geben Sie das Gerät zur Reparatur an unseren Service.
- Das Gerät darf nicht an einem feuchten Ort stehen, keinem Niederschlag, Wasser, Spritzwasser, Staub oder ständiger direkter Sonnenbestrahlung ausgesetzt sein.
- Starke mechanische Beanspruchungen, wie z. B. Druck oder Vibration sind zu vermeiden.
- Das Gerät nur mit einem trockenen Leinentuch reinigen, das bei starken Verschmutzungen leicht angefeuchtet sein darf. Zur Reinigung keine lösemittelhaltigen Reinigungsmittel verwenden. Darauf achten, dass keine Feuchtigkeit in das Geräteinnere gelangt. Beim Reinigen nicht zu stark auf das Display drücken!

## **Anschlusshinweise**

Folgende Hinweise müssen beim Anschluss der Linux-Control-Unit beachtet werden:

- Alle Anschlussleitungen von KL1 dürfen nicht länger als 30 m sein.
- Alle Anschlussleitungen müssen mit Ferrit-Ringkernen versehen werden. Idealerweise werden die Ringkerne mit Schrumpfschlauch überzogen.
- Die Zuführung von Signalen an die analogen Eingänge (KL1, Pin 3-5) muss über ein abgeschirmtes Kabel erfolgen.

## **Spannungsversorgung**

- Zur Gewährleistung der elektrischen Sicherheit muss es sich bei der speisenden Quelle um eine Sicherheits-Schutzkleinspannung handeln. Außerdem muss es sich um eine Quelle begrenzter Leistung gemäß EN60950-1 handeln, die nicht mehr als 15 W liefern kann.

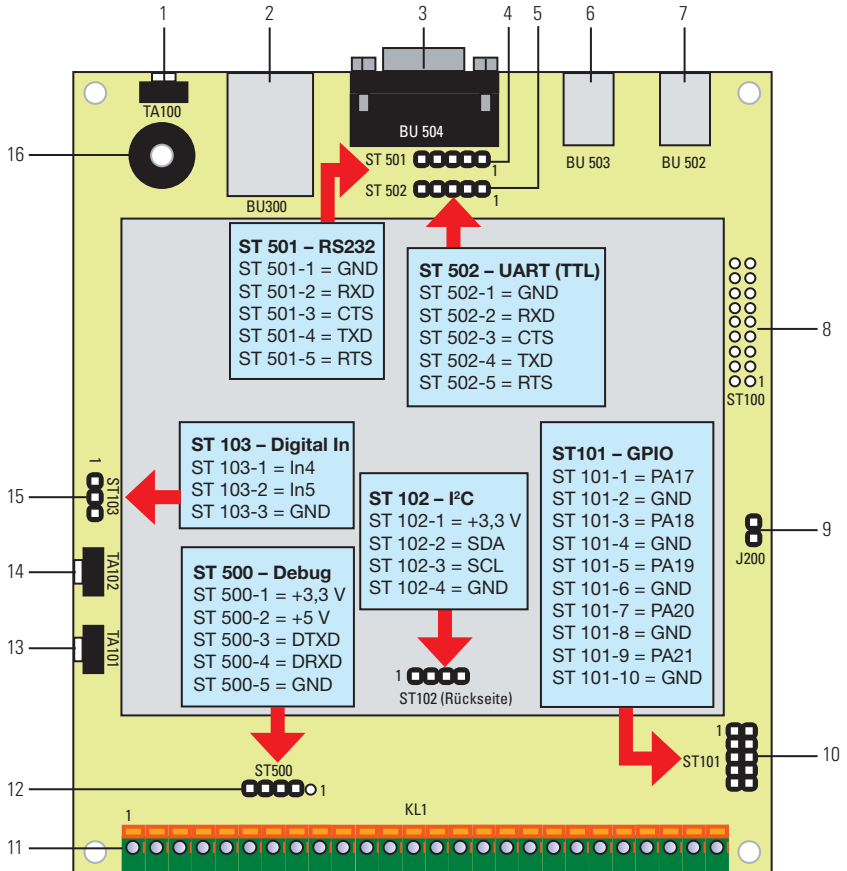
## **Gehäuseeinbau**

- Für einen ausreichenden Schutz vor elektrostatischen Entladungen ist der Einbau in ein geeignetes (nichtmetallisches) Gehäuse erforderlich, damit die Schaltung nicht durch eine Berührung mit den Fingern oder Gegenständen gefährdet werden kann. Versehen Sie das Gehäuse mit dem auf der letzten Seite abgebildeten Typenschild.

## **Hinweis zu Linux**

- Diese Anleitung ist eine Bedienungsanleitung zu Anschluss, Betrieb und ersten Schritten der Programmierung der LCU1. Sie vermittelt keine Linux-Grundkenntnisse, diese werden vorausgesetzt.

### 3. Anschlüsse, E/A-Elemente



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

\* Unter Linux ansprechbar mit `/dev/ttyS1`

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

\*\*\* Unter Linux ansprechbar mit `/dev/ttyS0`

## Belegung KL1

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

## 4. Hinweise zu Linux-Entwicklungs-Tools, Bootloader

**Im per Download (siehe Seite 15) bei ELV beziehbaren Softwarepaket befinden sich auch einige auf die LCU1 angepasste Software-Tools sowie erste Demoprogramme, auf die in den folgenden Abschnitten eingegangen wird.**

Der Programmierer, an den sich ja die LCU 1 wendet, hat eine große, freie Vielfalt an Werkzeugen zur Lösung seiner Aufgabe zur Auswahl wie beispielsweise:

- Laufzeitbibliotheken für C und C++ (GNU-Libraries)
- Qt/Embedded für die Entwicklung grafischer Anwendungen auf verschiedenen Hardware-Plattformen, speziell auf mobilen Geräten mit begrenzten Speicherkapazitäten
- den freien, modularen Web-Server „Lighttpd“ mit CGI (CGI ist eine Schnittstelle zwischen Web-Server und externer Software)

Diese Software-Bausteine machen die Entwicklung von grafischen HTML-/CGI-Anwendungen besonders einfach, auch der Einsatz eines solchen Embedded-Systems wie dem LCU 1 als Bedienterminal für Serverapplikationen ist so einfach lösbar.

### Kommandozeilenprogramm „busybox“

Für Embedded-Systeme ist die Bedienung per Kommandozeile sehr wichtig. Hier gibt es ein spezielles Kommandozeilenprogramm, „busybox“ [8], das alle wichtigen Tools und Kommandos hierzu in einem einzigen Programm vereint. Abgesehen von den verwendeten Systembibliotheken ist busybox bei der LCU 1 nur ca. 600 KByte groß.

### Compiler

Als Compiler kommt der GNU-Compiler (gcc) zum Einsatz. Für die grafische Interaktion mit dem Anwender kommt die Bibliothek „directfb“ [9] zum Einsatz. Sie ist schlank und setzt direkt auf das Linux-Framebuffer-Device auf. Das stellt einen Speicherbereich zur Verfügung, über den direkt die Displaypixel manipulierbar sind, das Display ist also direkt als Speicherbereich in die Anwendungsschicht eingebundet. „directfb“ bietet, darauf aufsetzend, die Möglichkeit, Grafiken, z. B. PNG, anzuzeigen, Rechtecke und Linien zu zeichnen, Text auszugeben oder auf Ereignisse (Maus, Touchscreen) zu reagieren. Auf der Basis von „directfb“ kann man in C eigene grafische Oberflächen erstellen.

## Buildroot

Alle Teile eines Embedded-Linux-Systems (u. a. Compiler, Bootloader, Systembibliotheken, Kernel, Anwendungen, evtl. grafische Bibliotheken) von den richtigen Stellen in den richtigen Versionen herunterzuladen und für das konkrete System anzupassen, ist sehr aufwändig. Diese Arbeit wird sehr erleichtert von „buildroot“ [10]. Dies ist eine Sammlung von Skripten und Makefiles, die vom Herunterladen über das Kompilieren bis zum Erstellen eines Dateisystems alle nötigen Schritte übernimmt. buildroot ist hoch flexibel und wird grafisch konfiguriert. Auch deshalb werden wir in der Folge eine auf buildroot basierende Entwicklungsumgebung verwenden, die sich durch Konfiguration von buildroot oder Hinzufügen eigener Programme erweitern lässt.

## Bootloader „U-Boot“

Der Bootloader ist verantwortlich dafür, dass nach dem Start des Systems der Linux-Kernel ordnungsgemäß in das System geladen und gestartet wird. Er ist Hardware-abhängig und sehr flexibel an die eigene Hardware-Plattform anpassbar. „U-Boot“ ist ein sehr kompakter Standard-Bootloader für Embedded-Systeme, der den Vorteil hat, auch vom NOR-Flash, NAND-Flash, Dataflash, USB, Netzwerk etc. booten zu können. Insbesondere das Booten per Netzwerk ist für Embedded-Systeme sehr nützlich, dadurch ist keine Neuprogrammierung des betreffenden Gerätes nötig. Die in der LCU 1 verwendete ARM-Version benötigt nur 200 KByte Speicher und ist damit besonders kompakt ausgeführt. In [11] und [12] ist der Bootloader ausführlich beschrieben. Für das Verständnis der Prozesse beim Start der LCU 1 zeigen wir im nächsten Kapitel den Ablauf des Boot-Vorgangs detailliert auf.

## 5. Boot-Reihenfolge

- 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 werden im nächsten Kapitel detaillierter betrachtet.

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.

## 6. 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 „Anhang“). 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.

### **Virtual Box**

Windows-Nutzer können auch ein Linux-System quasi parallel als „Virtual Box“ betreiben, denn ohne geht es nicht, die Entwicklungsumgebung basiert auf einem Linux-System. „Virtual Box“ von Oracle/Sun ist eine Freeware, die es ermöglicht, ein zweites, vom bisher installierten Betriebssystem – z. B. MS Windows oder Mac OS X – unabhängig arbeitendes Gastsystem, in diesem 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.

### **Anmeldung an der LCU1**

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 von ELV 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 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.

## 7. 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
```

### Relais/Piezo-Signalgeber schalten

Kommando „1“ bedeutet: Relais angezogen/Piezo-Signalgeber ein  
Kommando „0“ bedeutet: 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.

## 8. Die Entwicklungsumgebung

An dieser Stelle sei nochmals gesagt, dass die Kompilierung und Nutzung des Quellcodes der Entwicklungsumgebung Linux-Kenntnisse erfordert. Für Linux-Einsteiger seien die zahlreichen Einführungen und die zu den jeweiligen Distributionen vorhandenen Einsteigerforen, die auf den Seiten der Distributionen verlinkt sind, empfohlen.

### Linux installieren

Die Nutzung der mitgelieferten, im Quellcode vorliegenden Software und deren Komponenten erfordert zunächst die Installation eines Linux-Systems, wie z. B. Ubuntu oder Debian, auf dem Entwicklungsrechner.

Zusätzlich sind über die Programmpaketverwaltung des jeweiligen Systems noch der C/C++-Compiler und die Bibliothek zur Entwicklung grafischer Anwendungen „Qt“ zu installieren.

Ein frisch installiertes Linux fordert beim Kompilieren des Quellcodes häufig noch die Installation weiterer Paketdateien, die in den meisten Fällen ebenfalls über die Programmpaketverwaltung installierbar sind.

Danach kopiert man den Quellcode (lcu1-oss-src-0.5.tar.gz, Bezeichnung kann je nach aktuell ausgelieferter Version abweichen) der LCU-Software in ein per „mkdir“-Befehl zu erzeugendes Verzeichnis im Linuxsystem, z. B. /home/user/lcu1.

Nach dem Wechsel in dieses Verzeichnis:

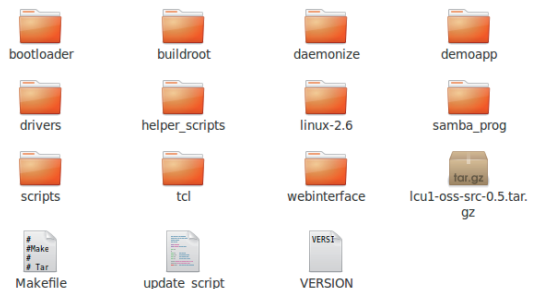
```
cd /home/user/lcu1
```

wird der Quellcode hiermit:

```
tar xzf lcu1-oss-src-0.5.tar.gz
```

entpackt.

Im Screenshot rechts sehen Sie die Reihe der entpackten Dateien bzw. Ordner:



Danach wird die Software mit den folgenden Schritten kompiliert und erstellt:

```
make -C buildroot
```

Damit wird „buildroot“, ein Basissystem mit Tool-chain zur Cross-Kompilierung, erstellt. Dieser Vorgang kann mehrere Stunden dauern, ist jedoch nur einmalig nötig. Die mit buildroot erstellte Toolchain wird dann zum Erzeugen der eigentlichen Firmware verwendet. Dieser Erstellungsprozess wird gestartet mit:

```
make
```

Schließlich erzeugt man die installierbare Firmware-Version mit:

```
make install
```

Dieser Befehl bringt die durch „make“ erstellten kompilierten Dateien in eine Form, die zum Installieren auf der Hardware geeignet ist. Es werden folgende Dateien erzeugt:

- **bootloader\at91bootstrap\binaries\at91sam9261ek-dataflashboot-2.10.bin**

Dies ist der First-Level-Bootloader, er wird ins DataFlash der LCU 1 installiert. Buildroot enthält zwar auch bereits eine Version von „at91bootstrap“, unsere Version wurde jedoch so modifiziert, dass der Watchdog des AT91SAM9261 weiter nutzbar ist.

- **bootloader\u-boot\u-boot.bin**

Dies ist der bereits vorgestellte Bootloader „U-Boot“, auch er wird ins Dataflash der LCU 1 installiert. Er wurde hier um den Support für die LCU erweitert.

- **ulmage**

Die (komprimierte) Imagedatei des Linux-Kernels. Sie wird ebenfalls ins Dataflash installiert und später von „U-Boot“ geladen und gestartet. Der Linux-Kernel (2.6) wurde um den Support für die LCU 1 erweitert. Er enthält gegenüber dem Standard-Kernel noch Bugfixes für: Treiber, Touch-Controller ADS7843, Atmel-SPI-Controller und USB-Gadget-Ethernet.

- **jffs2.image**

Unter dieser Image-Datei verbirgt sich das Root-Dateisystem für den NAND-Flash, sie wird in das NAND-Flash installiert.

- **lcu1-firmware-0.5-img**

Dieses Firmware-Image enthält die Dateien ulmage und jffs2.image sowie ein Shell-Skript, welches ein Firmware-Update durchführt. Dieses Image ist für die Installation der Firmware über das Web-Interface geeignet.

## SAM-BA

Für die Erstinstallation ist die von Atmel bereitgestellte Software „SAM-BA“ nötig. Diese gibt es für Windows und für Linux. Da die Linux-Version aufgrund des verwendeten Gerätetreibers für USB-zu-RS232-Umsetzer schwierig zu verwenden ist, insbesondere, wenn sich noch weitere USB-zu-RS232-Umsetzer im System befinden, empfehlen wir die Verwendung der Windows-Version. Bei Verwendung der Linux-Version ist die Vorgehensweise aber prinzipiell dieselbe.

Nach der Installation von SAM-BA, enthalten im Paket „AT91-ISP v1.13“, muss die Software durch Entpacken von „sam-ba 2.9\_patch\_lcu.zip“ (im Ordner „samba-prog“) nach

C:\ATMEL Corporation\AT91-ISP v1.13

um den Support für die LCU 1 erweitert werden.

SAM-BA verwendet den im ROM des AT91SAM9261 enthaltenen Bootloader. Damit dieser gestartet wird, darf keine andere Firmware (z. B. im Dataflash) vorhanden sein.

***Daher muss für den Updatevorgang über SAM-BA beim Einschalten der LCU 1 der Jumper J 500 gesetzt sein.*** Das Dataflash wird hierdurch deaktiviert. Vor dem eigentlichen Zugriff mit SAM-BA muss der Jumper wieder entfernt werden.

Die LCU 1 wird per USB an den PC angeschlossen. Nach dem ersten Anschließen der LCU 1 an den PC erfolgt automatisch die Windows-Treiberinstallation.

Danach ist im Ordner „samba-prog“ die Batchdatei „program-all.bat“ zu starten. In diesem Ordner befinden sich noch weitere Batchdateien, die nur einzelne Teile des Speichers programmieren.

Die Firmware lässt sich bei bereits installierter Firmware auch über den Web-Browser updaten mit:

```
http://<lcu-IP>/fwupdate.cgi
```

## Weitere Programmbestandteile

Die oben mit Buildroot erstellte Toolchain kann auch zum Erstellen eigener Programme verwendet werden. Dazu ist das Compiler-[Toolchain-Verzeichnis (buildroot/output/staging/usr/bin) in den Suchpfad aufzunehmen. Es handelt sich dabei um eine „arm-linux“-Toolchain. Der Compiler „gcc“ wird daher als „arm-linux-gcc“ aufgerufen.

Die Konfiguration des Linux-Kernels kann über eine Qt-basierte grafische Oberfläche angepasst werden. Dazu kann der folgende Befehl verwendet werden:

```
• make kernel_xconfig
```

Ebenso lässt sich „buildroot“ per Qt konfigurieren:

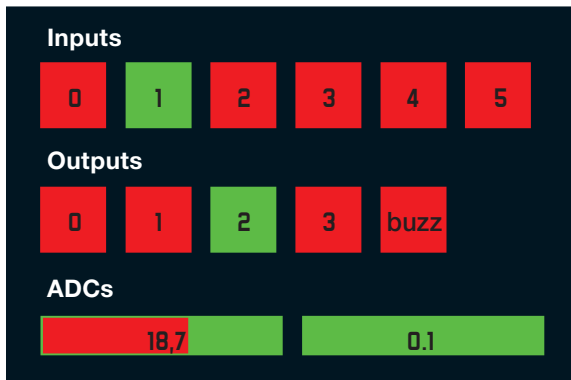
```
• make -C buildroot xconfig
```

## 9. Die Demo-Applikation

Die mitgelieferte Demo-Applikation ermöglicht ein erstes Kennenlernen der grafischen Schnittstelle und der Hardwareausstattung der LCU 1. Gestartet wird diese an der Kommandozeile der LCU durch Eingabe von

• `demoapp`

Auf dem Touchscreen wird ein komplettes Anzeige- und Bedienfeld dargestellt:



Das Demoprogramm in Aktion: Digital-Eingang 1 ist aktiviert, ebenso der Relais-Ausgang 2, und am Analog-Eingang 1 werden 18,7 V gemessen.

- In der oberen Reihe „Inputs“ erfolgt die Anzeige des Zustands der Digital-Eingänge. Ist ein Eingang aktiv, wechselt er die Farbe auf Grün.
- In der mittleren Reihe „Outputs“ lassen sich die Relais und der Signalgeber über berührungsempfindliche Bedienfelder schalten. Auch hier wird der aktivierte Zustand grün angezeigt.
- In der Reihe „ADCs“ schließlich zeigen zwei Felder den Spannungswert der beiden ADC-Eingänge sowohl numerisch als auch durch einen mit dem Wert wachsenden Farbbalken an.

In den Auszügen aus dem Demoprogramm im Anhang sind als Beispiel die Programmteile dargestellt, die für die beschriebenen Teile der Demo-Applikation federführend sind. Da das Programm offenliegt, kann man sich sofort an eigene Experimente machen bzw. Teile des Programms in eigene Applikationen übernehmen.

Der Quellcode findet sich in der Entwicklungsumgebung unter „demoapp“. Er wird beim Kompilieren des Gesamt-Quellcodes standardmäßig mit kompiliert und installiert, aber nicht automatisch gestartet.

Soll dies erfolgen, kann man in der Entwicklungsumgebung im Makefile „scripts/Makefile“ das Doppelkreuz in der letzten Zeile:

```
#$(ROMFSINST) -p 755 /etc/init.d/S99demoapp
```

löschen. Dann wird die Startdatei „S99demoapp“ nach „/etc/init.d/S99demoapp“ auf dem Zielsystem installiert und beim Systemstart ausgeführt.

Wer als erfahrener Linux-Nutzer nur allein die Demo-Applikation kompilieren möchte, kann dies mit:

```
# make subdir_demoapp
```

tun.

Die eigene (modifizierte) Version des Programms lässt sich z. B. mit Hilfe des kostenlosen Open-Source-FTP-Programms „Filezilla“ oder unter Linux mit dem scp-Befehl auf das Zielsystem übertragen:

```
scp demoapp/demoapp root@10.101.81.51/usr/local/demoapp
```

Diese eigene Version wird dann unter Angabe des kompletten Pfades mit

```
# /usr/local/demoapp
```

gestartet.

Soweit die Einführung in die Installation der Software und die Konfiguration der Hardware der LCU 1.

Unter Nutzung der mitgelieferten Quellcode-Software ist es dem Linux-Programmierer somit möglich, die LCU 1 für das eigene Projekt zu konfigurieren, zu programmieren und fernzusteuern.

#### **Download-Hinweis:**

Der Quellcode der im Produkt enthaltenen Software kann auf

<http://www.elv.de>

kostenlos\* heruntergeladen werden.

Nutzen Sie hierzu folgenden Webcode in der Suchfunktion: #7002

\*exklusive Ihrer Online-Verbindungskosten

## 10. Weblinks, Quellen

Die hier aufgeführten Links und Quellen, die nicht innerhalb der Bedienungsanleitung aufgeführt sind, dienen der weiteren und allgemeinen Information über die Hardware und Linux.

- [1] <http://infocenter.arm.com>
- [2] [www.atmel.com/at91](http://www.atmel.com/at91)
- [3] [www.uclibc.org](http://www.uclibc.org)
- [4] [www.kernel.org](http://www.kernel.org)
- [5] [www.kernelplanet.org](http://www.kernelplanet.org)
- [6] [www.kernelnewbies.org](http://www.kernelnewbies.org)
- [7] [www.ibm.com/developerworks/linux/library/l-linux-kernel](http://www.ibm.com/developerworks/linux/library/l-linux-kernel)
- [8] [www.buypass.com](http://www.buypass.com)
- [9] [www.directfb.org](http://www.directfb.org)
- [10] <http://buildroot.uclibc.org/>
- [11] [www.linux-arm.org/LinuxBootLoader/WebHome](http://www.linux-arm.org/LinuxBootLoader/WebHome)
- [12] [www.denx.de/wiki/U-Boot/WebHome](http://www.denx.de/wiki/U-Boot/WebHome)
- [13] [www.at91.com](http://www.at91.com)
- [14] [www.at91.com/linux4sam](http://www.at91.com/linux4sam)
- [15] [http://upload.wikimedia.org/wikipedia/commons/f/f8/Linux\\_Kernel\\_Stuktur.svg](http://upload.wikimedia.org/wikipedia/commons/f/f8/Linux_Kernel_Stuktur.svg)
- [16] [http://upload.wikimedia.org/wikipedia/de/b/b3/Linux\\_Schichten.svg](http://upload.wikimedia.org/wikipedia/de/b/b3/Linux_Schichten.svg)

## 11. Technische Daten

|                                 |                                   |
|---------------------------------|-----------------------------------|
| Spannungsversorgung .....       | 10 - 30 V DC                      |
| Stromaufnahme (ohne Last) ..... | ca. 200 mA @ 24 V DC              |
| Relaisausgänge .....            | 4xUM, max. 0,75 A, max. 30 V DC   |
| Digitaleingänge .....           | 4, 0 - 15 V                       |
| Analogeingänge .....            | 2, 0 - 30 V                       |
| RAM .....                       | 64 MB SDRAM                       |
| FLASH .....                     | 256 MB NAND-FLASH, 8 MB DataFlash |
| Displayauflösung .....          | QVGA (320 x 240 Pixel)            |
| Abmessungen .....               | 170 x 160 x 20 mm                 |

## Entsorgungshinweis

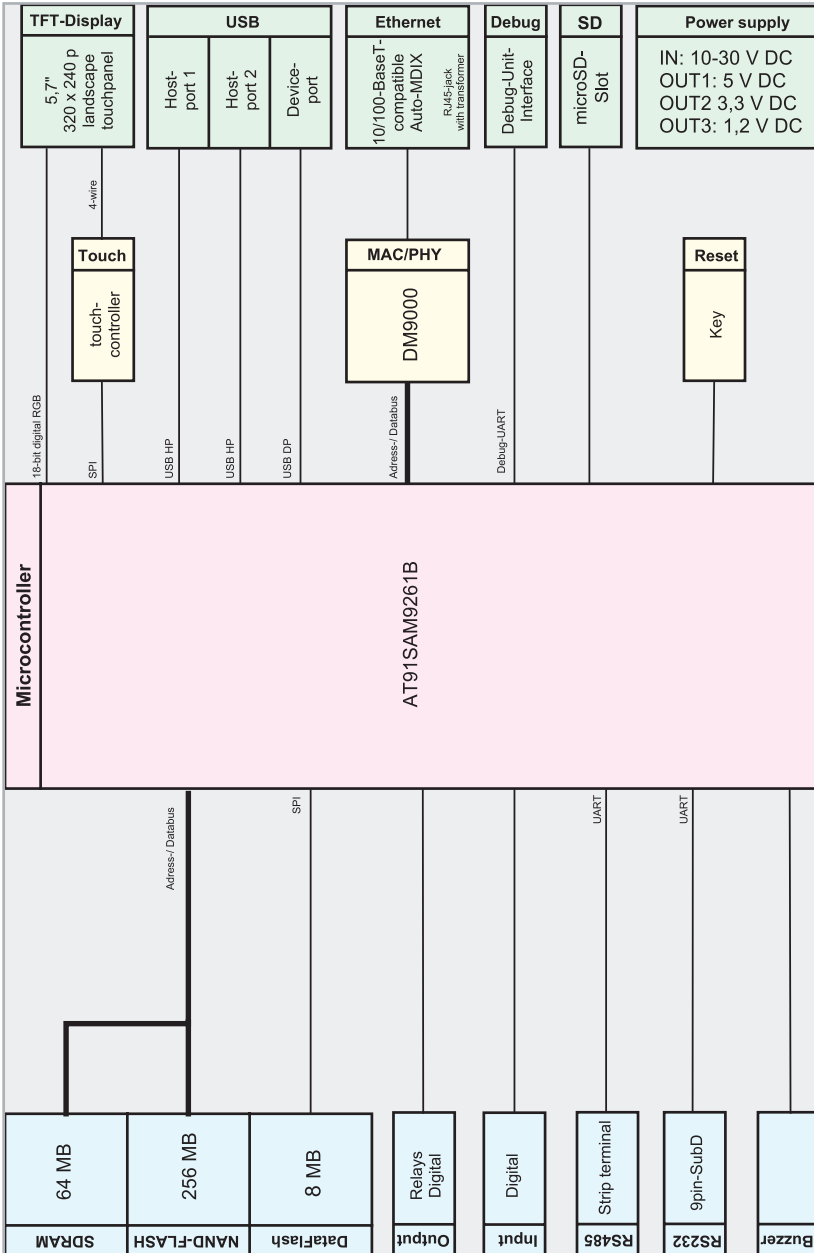
### **Gerät nicht im Hausmüll entsorgen!**

Elektronische Geräte sind entsprechend der Richtlinie über Elektro- und Elektronik-Altgeräte über die örtlichen Sammelstellen für Elektronik-Altgeräte zu entsorgen!



# 12. Anhang

## 12.1. Blockschaltbild der LCU1



## 12.2. Auszüge aus dem Demoprogramm „demoapp“

```
/* Anzahl der Digitaleingänge */
#define NUM_INPUTS 6

/* Anzahl der Digitalausgänge */
#define NUM_OUTPUTS 5

/* Anzahl der Analogeingänge */
#define NUM_ADCS 2

/* Größe der Quadrate für die Digitaleingänge und -ausgänge in Pixeln */
#define RECT_SIZE 40
/* Rand um die Quadrate für die Digitaleingänge und -ausgänge in Pixeln */
#define RECT_MARGIN 6
/* Y-Position der Oberkante der Quadrate für die Digitalausgänge in Pixeln */
#define OUTPUT_RECTS_TOP 120
/* Y-Position der Oberkante der Quadrate für die Digitaleingänge in Pixeln */
#define INPUT_RECTS_TOP 30

/* Y-Position der Oberkante der Rechtecke für die Analogeingänge in Pixeln */
#define ADC_TOP 210
/* Breite der Rechtecke für die Analogeingänge in Pixeln */
#define ADC_RECT_WIDTH 150
/* Höhe der Rechtecke für die Analogeingänge in Pixeln */
#define ADC_RECT_HEIGHT 25
/* Rand um die Rechtecke für die Analogeingänge in Pixeln */
#define ADC_RECT_MARGIN 6
/* Innerer Rand um den Wertebalken für die Analogeingänge in Pixeln */
#define ADC_RECT_INTERNAL_MARGIN 2
/* Maximaler ADC-Wert der Analogeingänge. Einheit mV */
#define ADC_MAX_VALUE 3850

/* Bitflags für die Verarbeitung von Touchscreen-Ereignissen */
/* Ein "Klick" wird erkannt, wenn nach einem Tastendruckereignis noch eine X-Koordinate und eine Y-Koordinate empfangen
wurden */
#define EVT_FLAG_X_RECEIVED 1 /* X-Koordinate empfangen */
#define EVT_FLAG_Y_RECEIVED 2 /* Y-Koordinate empfangen */
#define EVT_FLAG_PENDOWN_WAIT 4 /* Tastendruckereignis empfangen */
#define EVT_FLAG_PENDOWN 8 /* Tastendruckereignis und Koordinaten empfangen */
#define EVT_FLAG_COMPLETE (EVT_FLAG_X_RECEIVED|EVT_FLAG_Y_RECEIVED|EVT_FLAG_PENDOWN_WAIT)

/* Dateidescriptoren für die Gerätedateien der Digitaleingänge */
static int _inputFds[NUM_INPUTS];
/* Dateidescriptoren für die Gerätedateien der Digitalausgänge */
static int _outputFds[NUM_OUTPUTS];
/* Dateidescriptoren für die Gerätedateien der Analogeingänge */
static int _adcFds[NUM_ADCS];

/* Aktuelle Zustände der Digitaleingänge */
static bool _inputStates[NUM_INPUTS];
/* Aktuelle Zustände der Digitalausgänge */
static bool _outputStates[NUM_OUTPUTS];
/* Aktuelle Werte der Analogeingänge */
static int _adcValues[NUM_ADCS];

/*
 * Ausgabe der Grafikdarstellung
 */
static void renderScreen()
{
    _primarySurface->SetFont( _primarySurface, _font );

    /* Schriftfarbe setzen */
    _primarySurface->SetColor( _primarySurface, 0xff, 0xff, 0xff, 0xff );

    /* Überschriftstexte ausgeben */
    _primarySurface->DrawString( _primarySurface, "Inputs", -1, 10, INPUT_RECTS_TOP - TEXT_HEIGHT - RECT_MARGIN,
DSTF_TOPLEFT );
    _primarySurface->DrawString( _primarySurface, "Outputs", -1, 10, OUTPUT_RECTS_TOP - TEXT_HEIGHT - RECT_MARGIN,
DSTF_TOPLEFT );
    _primarySurface->DrawString( _primarySurface, "ADCS", -1, 10, ADC_TOP - TEXT_HEIGHT - ADC_RECT_MARGIN,
DSTF_TOPLEFT );

    /* Rechtecke zeichnen für die Digitaleingänge */
    for( int i=0; i<NUM_INPUTS; i++ )
    {
        /* Farbauswahl, grün für aktiv, rot für nicht aktiv */
        if( _inputStates[i] )
        {
            _primarySurface->SetColor( _primarySurface, 0x00, 0xff, 0x00, 0xff );
        }else{
            _primarySurface->SetColor( _primarySurface, 0xff, 0x00, 0x00, 0xff );
        }
        /* Zeichnen eines gefüllten Rechtecks */
        _primarySurface->FillRectangle( _primarySurface, RECT_MARGIN + i*(RECT_SIZE + 2*RECT_MARGIN), INPUT_RECTS_TOP +
RECT_MARGIN, RECT_SIZE, RECT_SIZE );
    }
}
```

```

/* Aktuelle Werte der Analogeingänge lesen */
readAdcValues();

/* Balken zeichnen für die Analogeingänge */
for( int i=0; i<NUM_ADCS; i++ )
{
    /* Grünes Rechteck als Hintergrund */
    _primarySurface->SetColor( _primarySurface, 0x00, 0xff, 0x00, 0xff );
    _primarySurface->FillRectangle( _primarySurface, ADC_RECT_MARGIN + i*(ADC_RECT_WIDTH + 2*ADC_RECT_MARGIN), ADC_TOP
+ ADC_RECT_MARGIN, ADC_RECT_WIDTH, ADC_RECT_HEIGHT );

    /* Darin rotes Rechteck mit Breite proportional zum Spannungswert */
    _primarySurface->SetColor( _primarySurface, 0xff, 0x00, 0x00, 0xff );
    int width = (ADC_RECT_WIDTH - 2*ADC_RECT_INTERNAL_MARGIN) * adcValues[i] / ADC_MAX_VALUE;
    _primarySurface->FillRectangle( _primarySurface, ADC_RECT_MARGIN + ADC_RECT_INTERNAL_MARGIN + i*(ADC_RECT_WIDTH +
2*ADC_RECT_MARGIN), ADC_TOP + ADC_RECT_MARGIN + ADC_RECT_INTERNAL_MARGIN, width, ADC_RECT_HEIGHT-
2*ADC_RECT_INTERNAL_MARGIN );

    /* Gemessene Spannung als Text mittig im Balkendiagramm ausgeben */
    char buffer[8];
    sprintf(buffer, "%.1f", double( adcValues[i])/1000 );
    _primarySurface->SetColor( _primarySurface, 0x00, 0x00, 0x00, 0xff );
    _primarySurface->DrawString( _primarySurface, buffer, -1, ADC_RECT_MARGIN + ADC_RECT_WIDTH/2 + i*(ADC_RECT_WIDTH +
2*ADC_RECT_MARGIN), ADC_TOP + ADC_RECT_MARGIN + ADC_RECT_HEIGHT/2 - TEXT_HEIGHT/2, DSTF_TOPCENTER );
}

```

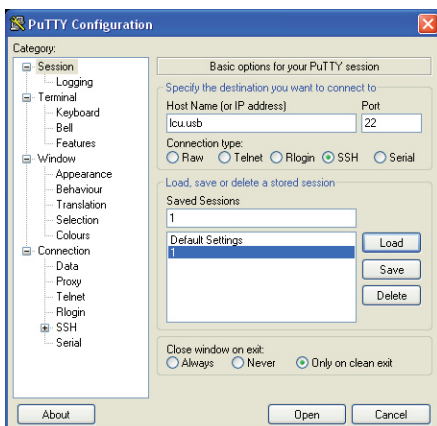
## 12.3. Secure Shell/SSH

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 unten), 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.

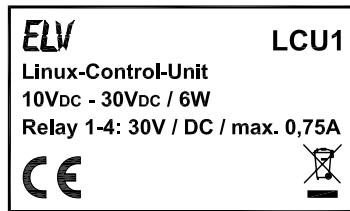
Ebenso sind die Ausgaben der Kommandozeile des entfernten Rechners auf dem Client-Rechner (Konsole) sichtbar. Auf dem entfernten Rechner muss dazu ein SSH-Server installiert und aktiv sein, der sich gegenüber dem SSH-Client mit einem Schlüssel authentifiziert, z. B. einem Kennwort oder einem Identitätsschlüssel.

Bei Linux-Distributionen ist der SSH-Client meist bereits integriert und wird (außer auf Live-Distributionen 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



**ELV AG • PF 1000 • D-26787 Leer**  
**Telefon 0491/6008-88 • Telefax 0491/6008-244**