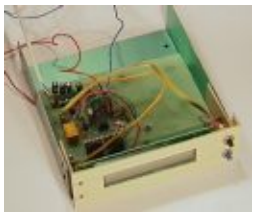
# Eine LCD Anzeige und Steuertasten für den Linux Server

von Guido Socher (homepage)

*Über den Autor:*

Guido mag Linux nicht nur, weil es Spaß macht, die großartigen Möglichkeiten, die dieses System bietet zu entdecken, sondern auch wegen der Leute, die an seiner Entwicklung beteiligt sind.

*Übersetzt ins Deutsche von:*
Guido Socher (homepage)



*Zusammenfassung*:

In diesem Artikel entwickeln wir ein LCD Kontrollfeld basierend auf einem Hitachi HD44780 LCD Display und dem AT90S4433 AVR 8−Bit RISC Microcontroller von Atmel. Beides sind geläufige Komponenten und sind nicht zu teuer. Das Kontrollfeld enthält neben Anzeige und zwei Tasten eine Watchdog, um den Server zu überwachen. Mit Hilfe der Anzeige und den Tasten kann man einen Dialog mit dem Benutzer führen und einigen Sachen einstellen. Man kann die IP Adresse, Netzmaske und Gateway Adresse setzen, außerdem ist es möglich den Rechner herunterzufahren (shutdown) oder Statistiken abzufragen. Eigentlich gibt es fast unbegrenzte Möglichkeiten. Der größte Teil der Logik ist in einem Perlscript implementiert und die ganze Schaltung ist über die RS232 Schnittstelle mit dem Server verbunden.

Für diesen Artikel brauchst du wenigstens Teile der Linux AVR Entwicklungsumgebung. Wie man diese Linux AVR Entwicklungsumgebung aufsetzt ist in dem Artikel "Programmierung des AVR Microcontroller mit GCC" beschrieben.

## Einführung



Die hier entwickelte Schaltung kombiniert Funktionalität, die schon in zwei vorhergegangenen Artikeln beschrieben wurde:

- Serial Line LCDs für Linux
- Ein Knopf zum Herunterfahren des Rechners für die serielle RS232 Schnittstelle mit LEDs

Unsere Neuentwicklung geht jedoch weit über das hinaus. Die neue Hardware enthält zwei Tasten für den Dialog mit dem Benutzer und eine Watchdog, um den Server ständig zu überwachen. Es ist außerdem ein analoger Eingang vorhanden, den wir hier jedoch nicht benutzen. Man könnte z.B einen Temperaturfühler anschließen.

Um diese Schaltung aufzubauen, braucht man etwas Geschick und Erfahrung im Bereich Hobbyelektronik. Die Bauteile für die Schaltung sind nicht teuer und kosten alle zusammen weniger als 40 Euro.

Die Idee hinter dem LCD Kontrollfeld ist, daß man damit einen Server ohne Monitor und Tastatur bedienen kann. Linux ist ein sehr zuverlässiges Serverbetriebssystem und kann gut per ssh, telent oder remote login verwaltet werden. Wenn man die Server jedoch das erste Mal mit dem Netzwerk verbindet, muß man eine IP Adresse, Netzmaske und Gateway setzen. Mit diesem Kontrollfeld kann man das machen. Es bietet außerdem die Möglichkeit, den Server herunterzufahren während man noch im Serverraum ist.

Das ganze Kontrollfeld ist sehr allgemein gehalten. All die Server spezifischen Teile sind in einem Perlscript implementiert. Die gesamte Hardware, der Zustand der Tasten, der Text im Dispaly, LEDs, ..., kann über ASCII Kommandos gesteuert werden. Man kann daher dieselbe Schaltung übernehmen, um einen mp3 Spieler zu bauen oder einen Toaster zu steuern, was immer du möchtest.

# Benötigte Bauteile

Um dieses Kontrollfeld zu bauen, benötigst du die folgenden Teile:

1 x Atmel At90S4433 Microcontroller
1 x 28pin 7.25 mm IC Sockel
1 x 16pin IC Sockel
1 x MAX232
1 x kleines 5V Relai
1 x 4MHz Quarz
2 x LEDs (grün und rot)
1 x BC547 NPN Transistor
1 x BC557 PNP Transistor
4 x 1uF Kondensator (Folie oder Elko)
2 x 27pF Keramikkondensator
1 x 10nF Kondensator
1 x 100nF Kondensator
3 x Widerstand 4k7
2 x Widerstand 2k2
1 x Widerstand 10K
1 x Widerstand 3k3
2 x Widerstand 100 Ohm
3 x Widerstand 470 Ohm
3 x Widerstand 1k
1 x Widerstand 220 Ohm
1 x 4K7 Poti (Bauform so klein wie möglich)
1 x Z–Diode 4.3V
2 x kleine Taster
1 x Diode (e.g 1N4148, irgendeine kleine billige Diode)
1 x 2 Zeilen 20 Zeichen LCD Display mit HD44780 kompatiblem Interface.

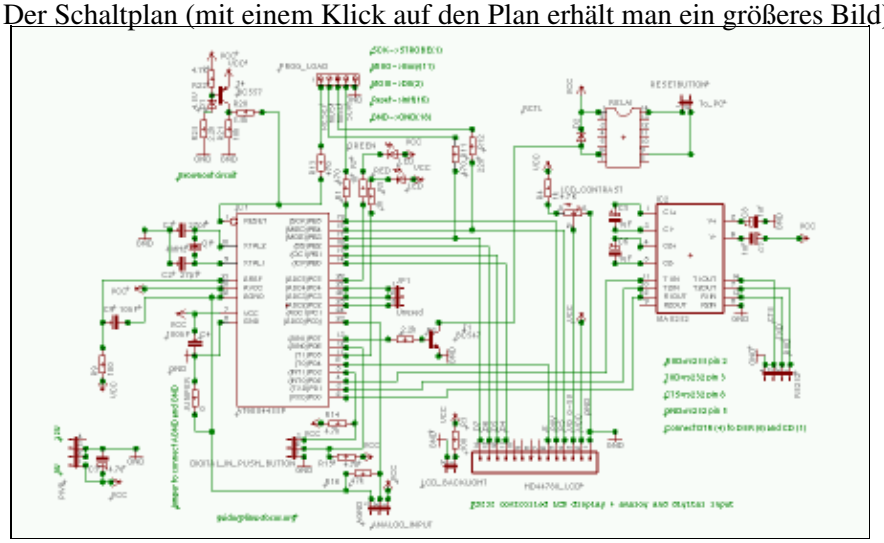
Alle LCD Displays mit 14 oder 16 Anschlüssen, die ich jemals gesehen habe, waren HD44780 kompatibel. Man kann auch ein 3 oder 4 zeiliges Display benutzen, aber dann muß man die Software für den Microcontroller etwas anpassen.

Zusätzlich zu diesen Teilen braucht man noch Drähte und Anschlüsse für Stromversorgung und RS2322. Das zweizeilige Display kann man an einem gebogenen Aluminiumblech befestigen und dann in einem freien 5.25" Schacht am Server montieren.

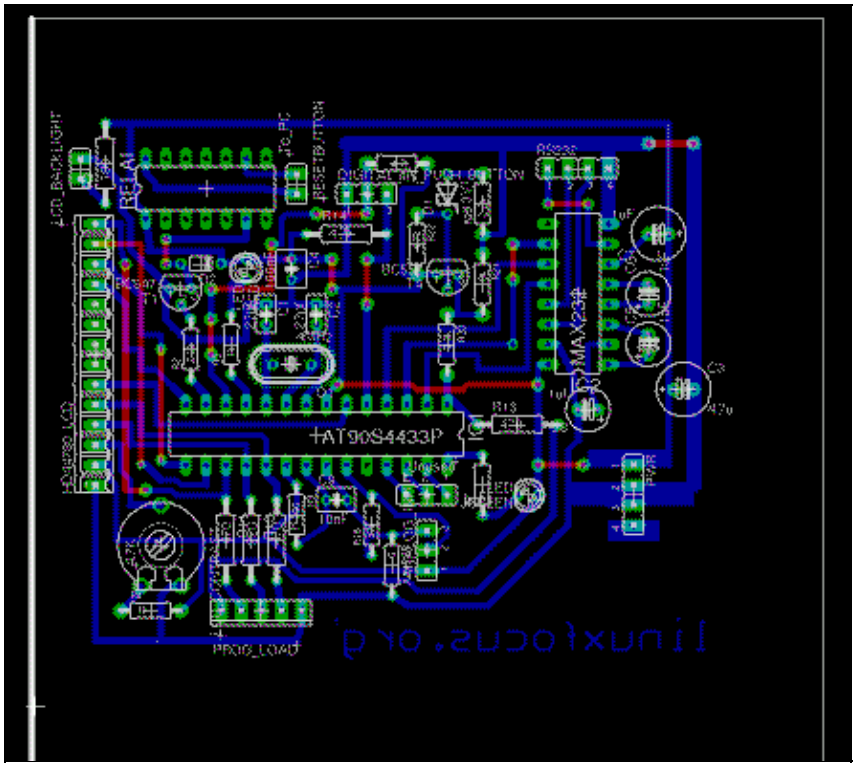
# Schaltplan und Platine

Ich habe Eagle für Linux benutzt, um den Schaltplan und die Platine zu entwerfen. Es ist eine ausgezeichnete Software, aber es braucht Zeit bis man sie bedienen kann. Für private Zwecke kann man eine kostenlose Version von Eagle bei cadsoftusa.com erhalten.

Der Schaltplan (mit einem Klick auf den Plan erhält man ein größeres Bild):



Die Platine (mit einem Klick auf die Platine erhält man ein größeres Bild):




Die Platine mit weißem Hintergrund, um das Drucken erleichtern: Platine in weiß (Achtung: Das ist nicht die Datei, die man zum Belichten der Platine braucht)

Die Eagle Dateien (komprimiert mit gzip. Achtung einige Webbrowser un−zippen die Dateien schon beim Herunterladen):

- linuxlcdpanel.brd.gz
- linuxlcdpanel.sch.gz

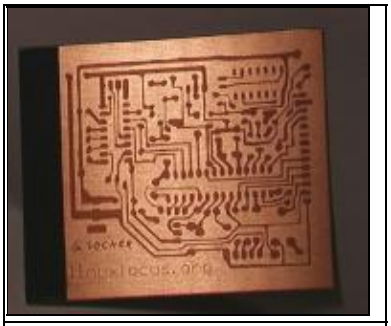
# Die Schaltung

Ich werde kurz den Schaltplan erklären. Der AT90S4433 hat 3 Ports: PB, PC und PD. PC kann als analoger oder digitaler Eingang benutzt werden. Alle Ports können als digitale Ein− oder Ausgänge benutzt werden. Dieses wird per Software über das DDR (Data Direction Register) festgelegt. Wir benutzen alle Pins bis auf 23 als digitale Ein− oder Ausgänge (0 oder 5V). Der Max232 ist ein Spannungswandler. Das RS232 Interface benutzt +−10V und der Max232 konvertiert das in 0−5V. An Pin 1 (Reset) befindet sich etwas namens Brownout−Schaltung. Diese Schaltung hält den Reset auf 0 (aktiv) während Perioden mit unzureichender Spannungsversorgung. Das verhindert, daß die CPU Anweisungen fehlerhaft ausführt. Normalerweise ist das für einige Millisekunden während des Einschaltens oder beim Ausschalten der Fall. Die Brownout−Schaltung stellt im wesentlichen sicher, daß der Microcontroller richtig startet.

Einige Leute mögen sich wundern, warum eine Diode parallel zur Spule liegt und so gepolt ist, daß sie sinnlos scheint. Diese Diode ist sehr wichtig. Beim Ausschalten des Stromes an dem Relai würde die Spule eine sehr hohe Spannung induzieren, die den Microcontroller sofort zerstören kann. Diese induzierte Spannung hat entgegengesetzte Polarität zur Betriebsspannung an der Spule. Die Diode kann eine ganz kleine billige Diode im Glasgehäuse sein, aber es ist wichtig, daß sie vorhanden ist.

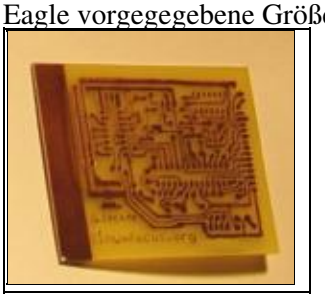
# Wie man eine Platine erstellt

Um die Platine zu ätzen, muß zuerst diese Postscriptdatei (linuxlcdpanel.ps.gz) auf eine transparente Folie ausgedruckt werden. In Läden für Architekten kann man eine semitransparente Plasikfolie namens Sinolit finden. Sie wird von der Firma Regulus hergestellt und ist für den Offsetdruck gedacht. Eine andere gute Alternative ist 60g Papier + Pausklarspray von Kontakt Chemie. Der Vorteil von Papier und Sinolit ist, daß der Druckertoner dort 100%ig haftet.
Ich habe die Postscriptdatei auch in PDF umgewandelt für den Fall, daß du kein Postscript drucken kannst, aber die Qualität ist sehr schlecht.
Die Belichtungszeit für fotobeschichtete Platinen hängt von der Lichtquelle ab. Bei einem Heimsolarium sind es etwa 1−2 Minuten. Man kann auch Tageslicht benutzen, aber direkte Sonneneinstrahlung sollte man vermeiden, da die Strahlung einfach zu stark ist. Man sollte vorher die beste Belichtungszeit mit einem kleinen Streifen Platine ermitteln. Die Folie und die Platine legt man während des Belichtens am besten unter eine Glasplatte, damit die Folie ganz plan liegt.



Die belichtete und entwickelte Platine vor dem Ätzen

Die belichtete Platine wird für einige Minuten in Natriumhydroxid (NaOH) entwickelt. Danach sollte man das Ergebnis sorgfältig prüfen und gegebenenfalls Korrekturen mit einem ätzfesten Edding 780 Lackstift (Edding 780 paint marker) vornehmen. Ich zeichne gewöhnlich die Pads für die Bauteile etwas größer, weil die von Eagle vorgegegebene Größe für Hobbyzwecke zu klein ist.



Das fertige Board vor dem Bohren der Löcher

Aus irgendeinem Grund scheint es unmöglich für die Hersteller von Relais zu sein, sich auf eine Anschlußbelegung zu einigen. Ich habe ein kleines Relai, das von Matsushita hergestellt wurde benutzt. Dein Relai wird vermutlich anders aussehen und du mußt daher die Platine anpassen. Das geht am besten mit so einem ätzfesten Edding Lackstift.

Wenn Du mit den Bahnen auf der Platine zufrieden bist, kannst Du sie in FeCl3 (Eisendreichlorid) ätzen. FeCl3 hat eine gute Ätzgeschwindigkeit bei Raumtemperatur. Es ist einfach zu benutzen und daher gut für Hobbyzwecke geeignet. Man bekommt die besten Ergebnisse, wenn die Platine in der Ätzlösung steht.

Kupferionen sind schwerer als Eisen und sammeln sich deshalb am Boden des Gefäßes an. Liegt dort die Platine, so verlangsamt sich die Ätzgeschwindigkeit sehr schnell.

Wenn die Platine fertig ist, spühlt man sie mit Wasser ab und wischt den Eddingstift mit Terpentin ab. Den Fotolack kann man auf den Bahnen lassen. Er verdampft beim Löten und schützt die ungelöteten Bahnen.

# Die Software für deinen Microcontroller

Die Software für den Microcontroller ist auf folgende Dateien aufgeteilt:

- lcd.c, lcd.h, lcd_hw.h: Das ist eine allgemeine LCD Library. Sie basiert auf der Arbeit von Peter Fleury (http://jump.to/fleury). Diese Version ist jedoch etwas modifiziert und flexibler. Man kann die LCD Hardware an jeden Pin des Microcontrollers anschließen. Dazu muß man lediglich die Definitionen in lcd_hw.h ändern.
- avr−util.c, avr−util.h: Funktionen zur Zeitverzögerung.
- uart.c, uart.h: Das ist die Library für das RS232 Interface. Es werden Hardwareinterrupts benutzt. Jedesmal, wenn ein Zeichen vom Computer empfangen wird, wird die Funktion SIGNAL(SIG_UART_RECV) aufgerufen und das Zeichen wird in einen String kopiert. Die ASCII Befehle für das Kontrollfeld sind so aufgebaut, daß ein kompletter Befehl immer eine Zeile ist (mit Newline endet). Wenn ein Newline gefunden wird, dann wird die Variable uart_rx_linecomplete gesetzt. Das bedeutet, daß man die Befehle nicht zu schnell hintereinander schicken darf. Es muß immer einige Millisekunden Pause geben, damit der String ausgewertet werden kann, bevor er durch einen neuen Befehl überschrieben wird. Jeder Befehl wird durch ein Ergebnis, ein ok oder err (für Error) beantwortet. Das kann man als Auslöser benutzen, um den nächsten Befehl zu schicken.
- analog.c, analog.h: Der Programmcode für den Analogdigitalwandler. Er funktioniert auch mit Interrupts. Ein Konvertierungsvorgang wird gestartet und dann wartet das Programm auf den SIG_ADC Interrupt, um das Ergebis aus dem ADC Register zu lesen.
- hardwarewd.c, hardwarewd.h: Das ist der Code für die Watchdog. Wir benutzen einen internen Teiler (durch 1024), um Pulse für den Timer zu liefern. Der Timer ist ein 16 bit register. Wenn dieses überläuft, dann zählen wir eine 8 Bit Variable herunter. Bei einem 4MHz Quarz wird diese 8 Bit Variable ungefähr alle 16 Sekunden heruntergezählt. Das Perlprogramm auf dem Server zeigt, daß es am Leben ist, indem es die 8 Bit Variable regelmäßig wieder auf einen Anfangswert setzt. Sollte das mal nicht passieren (weil der Server hängt), dann wird die 8 Bit Variable irgendwann auf Null gehen und das Relai macht klick−klack und der Server bekommt ein CPU reset.
- linuxlcdpanel.c: Das ist das Hauptprogramm. Es überprüft regelmäßig, ob Kommandos am RS232 Interface angekommen sind oder einer der Taster gedrückt wurde.

Um die Software im Detail zu verstehen, empfehle ich das Datenblatt für den Microcontroller zu lesen. Das Datenblatt findet man im Abschnitt Referenzen am Ende des Artikels oder auf der Atmel Webseite (www.atmel.com).
Um jedoch dieses Kontrollfeld benutzen zu können, braucht man die Software überhaupt nicht zu verstehen. Man muß lediglich den Quellcode auspacken (linuxlcdpanel−0.7.tar.gz gibt's auf der download Seite) und dann

```
make
make load
```

tippen. Man kann sogar einfach die schon vorkompilierte Software benutzen und sie mit

```
make loadprebuild
```

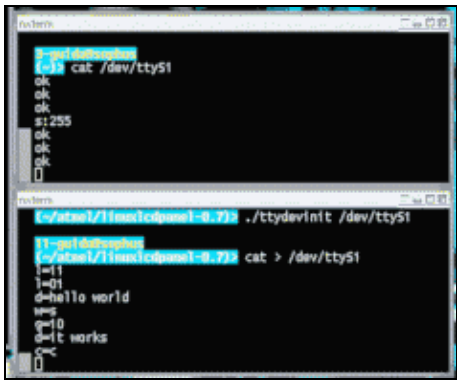
in den Microcontroller laden. Ganz einfach. Eine Beschreibung, wie man den Microcontroller programmiert und was man dazu braucht, findet sich im ersten Artikel: Den AVR Microcontroller mit GCC programmieren.

# Testen

Das LCD Kontrollfeld ist so gebaut, daß man es an die 5V interne Stromversorgung im Linux Server anschließen kann (rotes Kabel=5V, schwarzes Kabel=Masse=0V). Man sollte die Schaltung jedoch niemals sofort und ungetestet an den Computer anschließen. Es könnte sein, daß du einen kleinen Fehler beim Aufbau gemacht hast. Die Stromversorgung des Rechners ist sehr leistungsfähig und ein Fehler könnte den Rechner beschädigen und die Schaltung in Rauch aufgehen lassen.
Man sollte zum Testen eine strombegrenzte und elektronisch stabilisierte Laborspannungsquelle benutzen. Nachdem die Versorgungsspannung angeschlossen ist, kann man die Software wie oben angegeben in den EEPROM laden. Sobald das geschehen ist, sollte man einen Lauftext mit "linuxfocus.org" in der LCD Anzeige sehen. Als nächstes wird die RS232 Schnittstelle angeschlossen:

MAX232 Pin 14 an CTS (DB−9 Pin 8)
MAX232 Pin 7 an RXD (DB−9 Pin 2)
MAX232 Pin 13 an TXD (DB−9 Pin 3)
Man muß außerdem DTR, DSR und CD miteinander verbinden (DB−9 Pin 4, 6 und 1)

Das ist auch kurz in dem Schaltplan oben angegeben.




Bevor man die serielle Schnittstelle benutzen kann, muß man sie initialisieren. Das Quellcodearchiv, linuxlcdpanel−0.7.tar.gz, enthält ein Programm namens ttydevinit, das diese Initialisierung übernimmt. Falls das Kontrollfeld und COM2 (ttyS1) angeschlossen sind, dann muß man einmal folgendes Kommando ausführen:

./ttydevinit /dev/ttyS1

Nun ist die serielle Schnittstelle im Server initialisiert für eine Übertragungsrate von 9600 Baud und du kannst mit dem Kontrollfeld "reden". Dazu öffnet man zwei xterm Fenster. In einem tippt man "cat /dev/ttyS1" und in dem anderen "cat > /dev/ttyS1". Nun kann man z.B l=11 (LED 1 wird eingeschaltet) oder l=10 (LED 1 aus) eintippen. In dem ersten xterm Fenster sieht man, daß das Kontrollfeld immer auf die Befehle mit "ok" antwortet.
Alle verfügbaren Befehle sind in der Datei README.commands erklärt.

Das Quellcodearchiv enthält auch ein Perlscript namens ttytest.pl, das nichts anderes macht, als die rote LED in Abständen ein und wieder auszuschalten. Es ist gedacht als ein einfaches Beispielprogramm, in dem man sehen kann, wie das Kontrollfeld angesteuert wird. Du kannst es benutzen, um deine eigenen Programme zu schreiben. Dazu brauchst Du nur Grundkenntnisse in Perl.

# Die Watchdog anschließen

Die Watchdog ist standardmäßig aus. Man schaltet sie mit dem Befehl w=1 an und man setzt die Zeit mit s=x, wobei x einen Wert zwischen 1 und 255 haben muß. s=10 setzt den Watchdog−timeout auf 10*16sec=160sec. Das Ansteuerprogramm muß diesen Wert periodisch setzen, um zu verhindern, daß die Watchdog zuschlägt.

Falls sich der Server irgendwann mal aufhängt, dann wird dieser Wert nicht gesetzt, und wenn er auf Null heruntergezählt hat, gibt es einen Reset. Linuxserver sind sehr stabil und bleiben fast nie stehen. Wenn es jedoch mal passiert, ist meist niemand da, der den Resetknopf drücken könnte oder niemand weiß, wo der Server steht, weil er noch nie Probleme gemacht hat.

Die Watchdog ist so gebaut, daß sie nur einmal zuschlägt. Das verhindet, daß sie während des vermutlich folgenden Filesystem Check schon wieder zuschlägt.

Um die Watchdog physikalisch anzuschließen, muß man die zwei Drähte finden, die zu dem Resettaster am Server führen. Das Relai unserer Watchdog wird parallel dazu angeschlossen.

# Wie man die Watchdog benutzt

Die Watchdog stellt sicher, das der Server immer Programme ausführen kann. Sie stellt nicht sicher, daß ein Webserver läuft oder die Datenbank antwortet. Um so etwas zu überprüfen, sollte man crontab oder ein anderes Programm benutzen. Man kann recht sicher sein, daß crontab vermutlich funktioniert, weil die Watchdog sicherstellt, daß Software im allgemeinen immer noch ausgeführt werden kann.

Man kann z.B ein Script schreiben, das von cronjob regelmäßig ausgeführt wird und alle 15 Minuten eine Webseite von dem Webserver herunterlädt. Es ist jedoch Vorsicht geboten. Ein Webserver kann mit zu vielen Anfragen überladen sein. In diesem Fall ist es normal, daß er nicht alle beantwortet. Man sollte daher zählen, wie oft der Server versagt hat und ihn erst über das Script rebooten, falls er z.B die letzten 10 mal nicht geantwortet hat. Hier reicht ein normaler reboot (shutdown −r), ein Reset über die Watchdog ist nicht erforderlich.

Abgesehen von den Applikationen sollte man auch den freien Plattenplatz überwachen. Das folgende Shellscript gibt z.B. nur etwas zurück, wenn 80% oder mehr von einer Partition belegt sind.

df | egrep ' (8.%|9.%|100%) '

Über crontab kann man damit regelmäßig prüfen, ob noch genug Platz vorhanden ist und eine e−Mail schicken, wenn der Platz knapp wird (corntab verschickt Bildschirmausgaben automatich per e−Mail).

# Die Scripte auf dem Server

Fast die gesamte Logik für unser Linux LCD Kontrollfeld ist in einem Perlscript namens llp.pl implementiert. Kopiere diese Datei nach /usr/sbin/. Als nächstest kopiere die Datei ttydevinit nach /usr/bin und die Datei ifconfig_llp.txt (aus dem etc Verzeichnis im Quellcodearchiv) nach /etc. Nun editiere die Datei ifconfig_llp.txt und ändere die Adressen wie gewünscht.

NETMASK=255.255.255.0
IPADDR=10.0.0.4
GATEWAY=10.0.0.2

Nun sollte man eine Sicherheitskopie der Datei /etc/rc.d/init.d/network erstellen und dann die Datei etc/network aus dem Quellcodearchiv nach /etc/rc.d/init.d/network kopieren. Dieses Script und die Verzeichnisnamen beziehen sich auf Redhat und Mandrake. Das Script etc/network_all_distributions ist ein etwas einfacheres Script, das für alle Distributionen geeignet ist, man muß jedoch selbst herausfinden, wo in einer gegebenen Linuxdistribution die init−rc Verzeichnise sind. Das ist immer etwas unterschiedlich von Linuxdistribution zu Linuxdistribution.

Editiere nun die Datei /etc/rc.d/init.d/network und ändere die Zeile

/usr/sbin/llp.pl /dev/ttyS1&,

falls du nicht COM2 benutzt.

Nun kann man

/etc/rc.d/init.d/network start

tippen und das Kontrollfeld füllt sich mit Leben. Beachte: es ist unproblematisch, die IP Adressen über das Kontrollfeld zu ändern. Sie werden erst beim nächsten Reboot wirksam. Man kann daher alles austesten und anschließend die Werte wieder zurücksetzen (man kann auch einfach die Datei /etc/ifconfig_llp.txt editieren, um die Änderungen rückgängig zu machen).

# Log files

Das llp.pl Script schreibt ein Logfile nach /var/log/llp.log. Diese Datei wird nur sehr langsam größer. Es sollte normalerweise keine Notwendigkeit geben, sie über logrotate zu rotieren. Man kann es jedoch machen. Es ist kein post rotate Befehl in logrotate dazu nötig. Ein Konfigurationseintrag für logrotate könnte so aussehen:

```
/var/log/llp.log {
nocompress
monthly
}
```

Das Log enthält Einträge, wenn das System manuell heruntergefahren wird, eine IP Adresse geändert wurde (IP, GW, netmask) oder wenn es einen Watchdog Reset gab. Naturgemäß kann man den Watchdog Timeout nicht loggen, wenn er passiert. Er wird beim nächsten Hochfahren in das Log geschrieben.

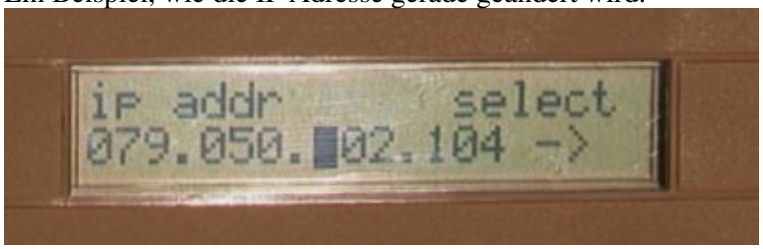
# Das Kontrollfeld im Betrieb

Hier sind einige Fotos der LCD Anzeige. Es sind nur Beispiele. Die Anzeige verfügt über wesentlich mehr Möglichkeiten und man kann seine eigenen Sachen hinzufügen.

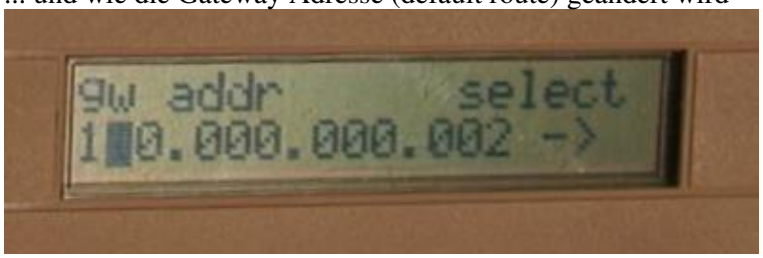
Das Hauptmenü. Es zeigt einen Namen (Linuxfocus in diesem Fall) und die Uptime sowie CPU load. Es ändert sich periodisch von selbst.


Das IP Konfigurationsmenü


Ein Beispiel, wie die IP Adresse gerade geändert wird.




... und wie die Gateway Adresse (default route) geändert wird




# Zusammenfassung

Um dieses Kontrollfeld zu bauen, ist etwas an Hobbyelektronikgeschick nötig, aber es ist keine komplizierte Schaltung. Dieses Linux Kontrollfeld bietet mehr Funktionen als jedes andere LCD Bedienfeld, das ich bisher gesehen habe, es ist sehr allgemein gehalten und preiswert.

Frohes löten :–)

# Referenzen

- Die uisp AVR Programmiersoftware: www.amelek.gda.pl/avr/
- Der Quellcode für diesen Artikel linuxlcdpanel–0.7.tar.gz
  Der Schaltplan, die Eagle Dateien und ein paar Fotos sind auch enthalten.
- Alle Software und Dokumentation aus diesem Artikel
- Datenblatt für MAX232 MAX220–MAX249.pdf 448K

- Datenblatt für ST232, eine billige Variante, oft anstelle des echten MAX232 verwendet. st232.pdf 100K
- Datenblatt für Atmel AT90S4433 avr4433.pdf 2356K
- Die Atmel Webseite: www.atmel.com
- Eagle für Linux cadsoftusa.com

---


| | |
|---|---|
| Der LinuxFocus Redaktion schreiben<br>© Guido Socher<br>"some rights reserved" see linuxfocus.org/license/<br>http://www.LinuxFocus.org | Autoren und Übersetzer:<br>en −−> −− : Guido Socher (homepage)<br>en −−> de: Guido Socher (homepage) |


2005−02−12, generated by lfparser_pdf version 2.51