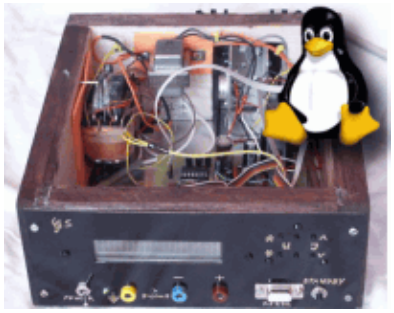
Ein Mikrocontroller gesteuertes Labornetzteil



by Guido Socher (homepage)

About the author:
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.

Abstract:

Dieses ist der vierte Artikel in der LinuxFocus AT90S4433 Mikrocontroller Serie. Es empfiehlt sich, vorangegangene Artikel aus der Serie im Hinblick auf folgende Punkte zu lesen:

1. Wie man die Linux AVR Entwicklungsumgebung installiert und wie man die Programmierer−Hardware baut:
   März 2002, Den AVR Mikrocontroller mit GCC programmieren
2. Wie man eine gedruckte Schaltung selbst herstellt:
   Mai 2002, Eine LCD Anzeige und Steuertasten für den Linux Server
3. Wie man ein Gehäuse für die Stromversorgung baut:
   September 2002, Frequenzzähler 1Hz−100Mhz mit LCD Display und RS232 Interface

Eines der wichtigsten und am häufigsten benötigten Geräte in einer Elektronikwerkstatt ist ein zuverlässiges Labornetzgerät. In diesem Artikel werden wir so ein Netzgerät bauen. Es wird Mikrocontroller gesteuert sein. Es hat eine LCD Anzeige und man kann es per Kommando von Linux aus über eine RS232 Schnittstelle steuern. Es hat ein sehr robustes Design.

Dieser Artikel zeigt auch, wie vielseitig Mikrocontroller sind. Es ist jedoch nicht die einfachste Schaltung. Falls du auf der Suche nach einer einfachen Stromversorgung bist, empfehle ich "Simple DC Power". Die Simple DC Schaltung ist gut, wenn man eine einfache Stromversorgung für andere elektronische Experimente von LinuxFocus braucht. Sie hat jedoch nichts mit Linux oder Software im allgemeinen zu tun.
Selbst wenn du die "Simple DC Power" Schaltung bauen möchtest, kann es interessant sein, diesen Artikel zu lesen. Er behandelt interessante Aspekte eines Mikrocontrollers.

Einführung

Diese Mikrocontroller basierte Stromversorgung ist nicht die einfachste Schaltung, aber ich kann dir versichern, dass du es nicht bereuen wirst, diese Schaltung gebaut zu haben. Sie ist extrem robust und zuverlässig. Sie ist technisch interessant, weil man sehen kann, wie man mit einem Mikrocontroller eine Gleichspannung erzeugen kann, ohne einen DA−Konverter−Chip zu benutzen.

Man braucht für diesen Artikel eine ganze Reihe Bauteile, aber alles sind billige Standardteile. Diese Schaltung ist überhaupt nicht teuer.

Was man braucht

Alle Bauteile die man braucht sind in der Bauteilliste aufgeführt. Man kann die Teile auch in dem Schaltplan weiter unten sehen. Unsere Stromversorgung kommt in drei Varianten. Außer dem Transformator und einem Widerstand unterscheiden sich die Versionen nur in der Software. Alle anderen Teile sind identisch für alle 3 Versionen:

1. 0−16V Max=2.2A
   man braucht einen Transformator mit 15V 2.5A
2. 0−24V Imax=2.2A
   man braucht einen Transformator mit 24V 2.5A
3. 0−30V Imax=3A
   man braucht einen Transformator mit 30V 3A

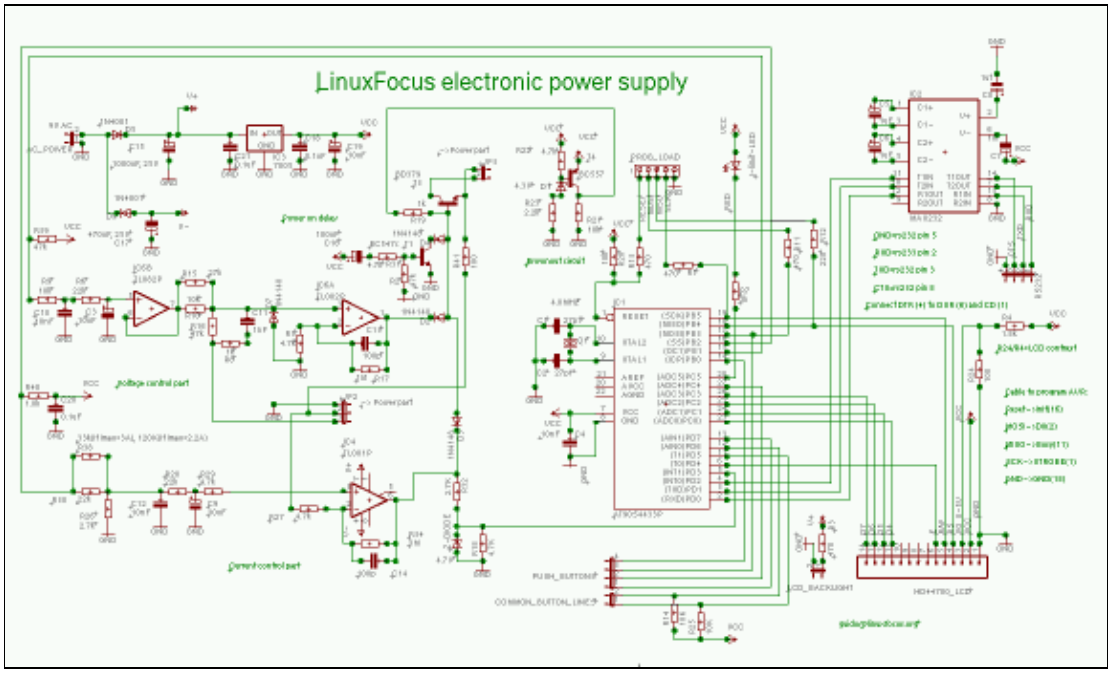
Beachte: In allen drei Fällen braucht man natürlich noch den 9V, 100mA Transformator für die Stromversorgung zur Hauptplatine.

Schaltplan und Platine

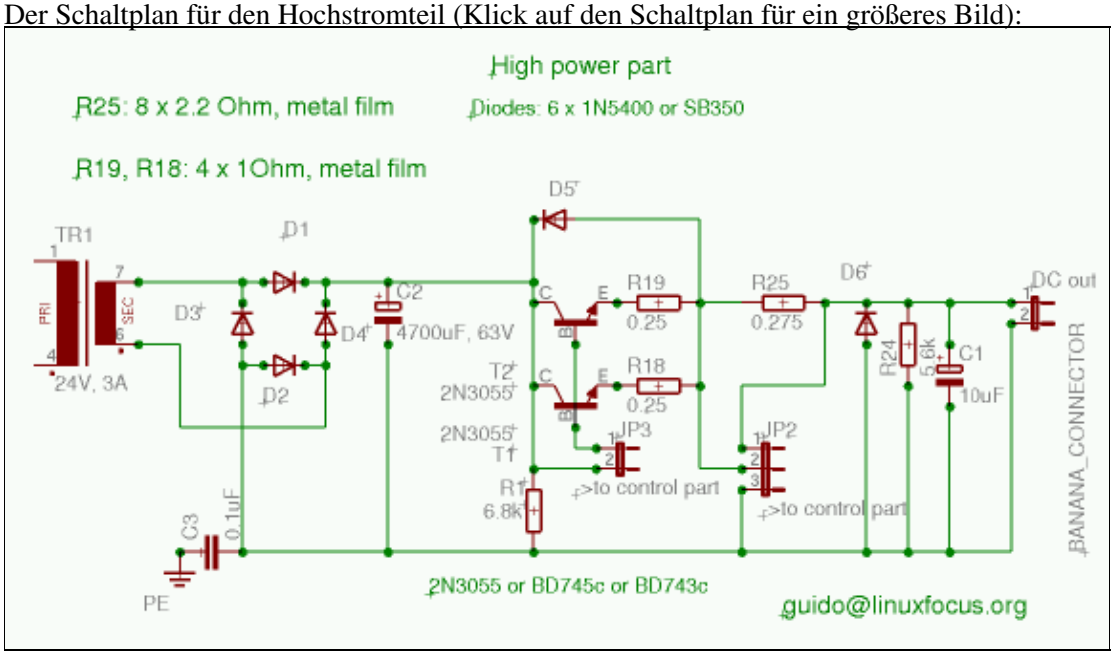
Ich habe eagle für Linux benutzt, um Schaltung und Platine zu entwickeln. Die Eagle Dateien sind zusammen mit der Software in dem tar.gz Paket enthalten. Dieses Paket kann man am Ende des Artikels herunterladen.

Die Schaltung ist in zwei Teile unterteilt: Hauptplatine und einen Teil, der nahe an den Leistungstransistoren sitzen sollte (Hochstromteil). Unten finden sich zwei getrennte Schaltpläne für diese Teile. Sie werden über Drähte miteinander verbunden.

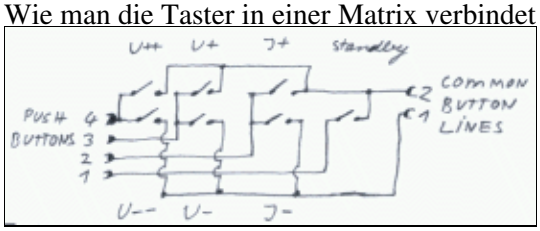
Der Schaltplan (Klick auf den Schaltplan für ein größeres Bild):



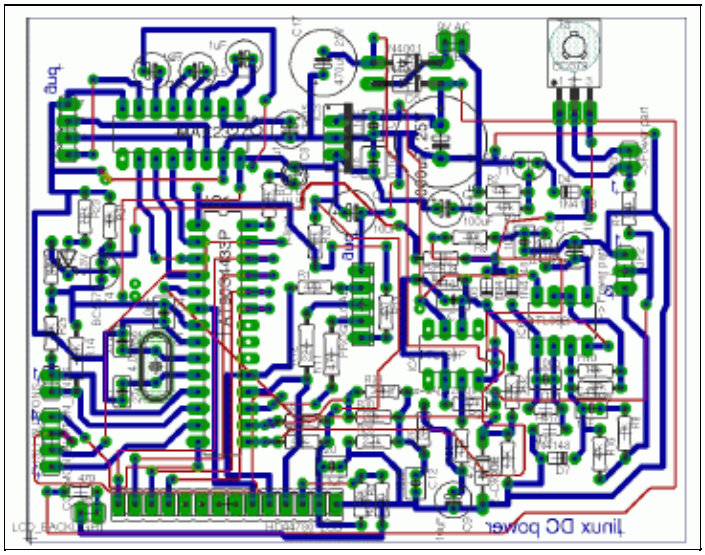
Der Schaltplan für den Hochstromteil (Klick auf den Schaltplan für ein größeres Bild):



Wie man die Taster in einer Matrix verbindet (Klick auf den Schaltplan für ein größeres Bild):



Die Hauptplatine. Ansicht von oben (Klick für eine größere Ansicht):



Die Platine wurde speziell für Hobbyelektronik entwickelt. Nur die blauen Bahnen werden geätzt. Die roten Bahnen sind Drähte. Es ist viel einfacher, eine einseitige Platine herzustellen, weil eine viel geringere Genauigkeit gebraucht wird. Man kann die roten Drähte so legen, dass sie möglichst kurz sind. In Eagle war das nicht so einfach möglich.

Die wenigen Teile für den Hochstromteil kann man auf einer einfachen Lochrasterplatine montieren. Die Hauptplatine und der Hochstromteil werden mit Drähten über die Verbinder JP2 und JP3 verbunden. Du wirst feststellen, dass Masse von der Hauptplatine mit Plus auf der Hochstrom–Platine verbunden ist. Das ist richtig. Aus diesem Grund brauchen wir zwei Transformatoren (einen für den Hochstromteil und einen für die Logik und den Mikrocontroller auf der Hauptplatine).

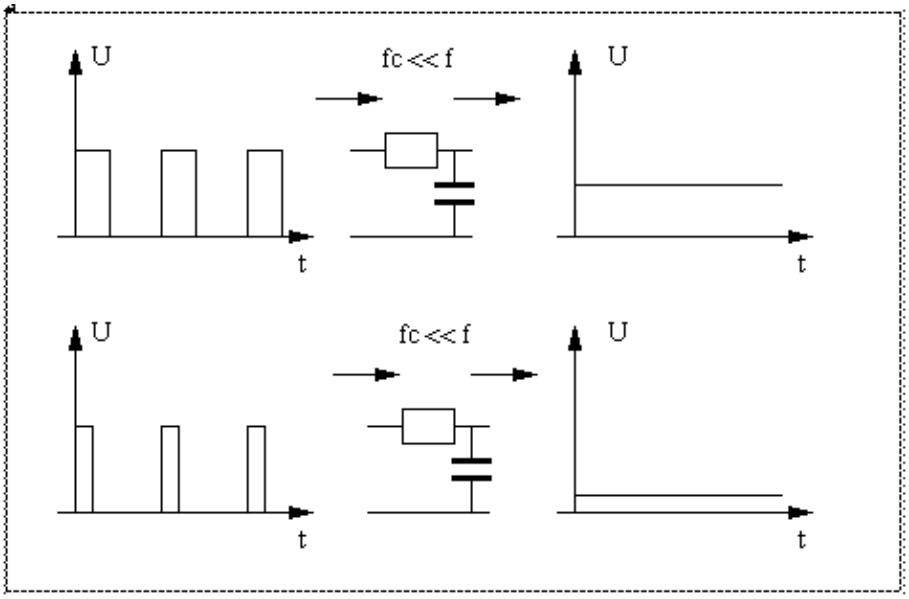
Und so funktioniert's

Betrachtet man den Schaltplan, dann sieht man, dass er aus 2 logisch getrennten Teilen besteht. Einer ist mit "current control" (Stromregelung) und der andere mit "voltage control" (Spannungsregelung) bezeichnet. Diese zwei Teile stellen unabhängige Regelkreise dar. Die eine Schleife regelt die Spannung und die andere den Spannungsabfall über dem 0.275 Ohm Widerstand. Der Spannungsabfall ist äquivalent zum Strom. Diese zwei Regelkreise werden "kombiniert" über die Dioden D2 und D3. Diese Dioden sind ein analoges ODER. Mit anderen Worten, wenn der Strom zu hoch ist, dann regelt die Stromregelung bis er passt und wenn die Spannung zu hoch ist, wird diese durch die Spannungsregelung begrenzt.

Das logische ODER funktioniert, weil der Transistor T3 über R19 an +5V angeschlossen ist. Wenn kein Operationsverstärker hinter D2 und D3 angeschlossen ist, dann hätte man maximale Ausgangsleistung. Die Operationsverstärker regeln also, indem sie dem Transistor T3 die Spannung wegnehmen (über die Dioden gegen Masse ziehen).

Die Spannungsregelung regelt gemäß der Spannung die an Pin 5 von IC6B liegt. Mit anderen Worten, Pin 5 ist äquivalent zur Ausgangsspannung multipliziert mit dem Verstärkungsfaktor, der sich aus R15, R10 und R16 ergibt. Das gleiche gilt für den Strom. Hier ist es nur die Spannung an Widerstand R30.

Um die Spannung oder den Strom einzustellen, braucht man also nur entsprechende Spannungen an den Punkten Pin 5 von IC6B und Widerstand R30 anlegen. Das ist, was der Mikrocontroller hier macht. Wie erzeugt man aber eine Referenzgleichspannung mit einem Mikrocontroller? Betrachte folgendes Bild:



Hier kann man sehen, wie ein gepulstes Signal in Gleichspannung umgewandelt werden kann. Alles was man tun muß, ist das Signal durch einen Tiefpass zu schicken, der eine sehr niedrige Grenzfrequenz hat. Hundertmal (oder mehr) niedriger. Da unser Mikrocontroller mit 4Mhz arbeitet, ist es nicht so schwer, solch einen Tiefpass zu bauen. Selbst wenn wir die Pulse über Software erzeugen und eine Frequenz von einigen kHz erreichen, ist der Tiefpass immer noch sehr klein, was die Größe der Bauteile angeht.

Den Unterschied in dem Bild zwischen dem oberen und unterem Diagramm nennt man Pulsweitenmodulation. Indem man die Länge des Pulses ändert, kann man die Spannung hinter dem Filter ändern.

Genial, nicht wahr? Wir können ein exaktes Gleichspannungssignal aus einem digitalen Signal erzeugen!

Der AT90S4433 Mikrocontroller hat zwei interne Zähler. Einer ist 16bit breit und der andere 8bit. Der 16bit Zähler kann für Pulsweitenmodulation (PWM) benutzt werden und alles nötige ist in dem AT90S4433 schon in Hardware implementiert. Die Auflösung ist hierbei 10bit. Der 8bit Zähler bietet das nicht, aber wir können es in Software implementieren. Das ist immer noch schnell genug. Wir benutzen den 16bit Zähler für die Spannungsregelung. Das gibt uns 10bit=1023 Stufen. Der 8bit Zähler, 255 Stufen, wird benutzt um den Strom zu regeln, 1−3000mA. Das bedeutet wir haben etwa 12mA (oder weniger) Auflösung. Ausreichend für eine Strombegrenzung.

Alle anderen Teile in der Schaltung sind für Stromversorgung, Referenzspannung (der 7805 ist unser Referenzpunkt) und um sicherzustellen, dass die Schaltung beim Ein− oder Ausschalten nicht instabil wird.

Die Software

Die Software für den Mikrocontroller benutzt viele Dinge, die wir schon aus früheren Artikeln kennen (uart für rs232, LCD Treiber, Zähler im Interruptmode). Du kannst hier einen Blick auf die Software werfen: linuxdcp.c.

Interessant ist vielleicht die Software−PWM (Pulse Width Modulation, Pulsweitenmodulation). Die Variable ipwm_phase implementiert zusammen mit ipwm_h die PWM für den Strom. Wir lassen den 8bit Zähler im Interruptmodus laufen und jedes Mal, wenn der Zähler überläuft, wird die Funktion "SIGNAL(SIG_OVERFLOW0)" aufgerufen. Hier prüfen wir ipwm_phase, um festzustellen, ob eine 1 oder

eine 0 erzeugt werden soll. Danach starten wir den Zähler wieder. Ganz einfach.

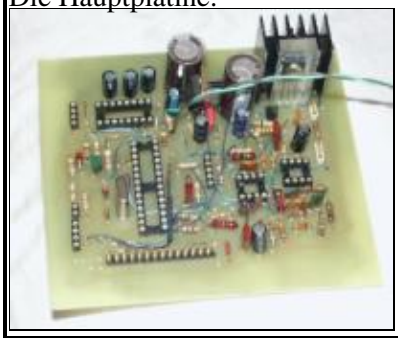
Die Software ist überhaupt nicht kompliziert, aber um sie zu verstehen, muß man das Datenblatt für den 4433 kennen (siehe Referenzen).

Der 4433 ist ein 8bit Mikrocontroller und seine mathematischen Fähigkeiten sind begrenzt. Die Funktionen divXbyY und multiXbyY implementieren 24bit Mathematik. Das gibt uns die Genauigkeit, die wir brauchen, um die Pulslänge aufgrund der vorgegebenen Spannung zu errechnen.

Das Netzgerät hat 7 Taster. 6 Taster um Strom und Spannung zu verändern und ein Taster für "standby". Mit dem "standby" kann man die Ausgangsspannung absenken und dann Strom und Spannungsbegrenzung ändern. Der Zustand der Taster wird in der Hauptschleife abgefragt. Die Variable ignorebutton wird benutzt, um die Taster zu entprellen. Wenn man mit der Hand einen Taster drückt, dann federt er etwas und schließt nicht nur einmal sondern zehnmal oder hundertmal. Ein Mensch merkt das nicht, aber der Mikrocontroller ist viel zu schnell. Mit der ignorebutton Variable warten wir nach dem ersten Puls vom Taster etwas und verhindern dadurch das Prellen.

Die gedruckte Schaltung bauen

Die Hauptplatine:



Das Gehäuse, Holz an den Seiten und Alu−Bleche oben, unten und vorne:

Die Frontseite:

Das Softwarepaket enthält eine Postscript Datei (linuxDCpower.ps) für die gedruckte Schaltung. Persönlich finde ich das die Lötaugen immer etwas zu klein sind. Ich empfehle daher, einen Lackstift zu nehmen und alle Augen von Hand nachzuzeichnen. Wie man eine gedruckte Schaltung macht ist in Mai 2002, Eine LCD Anzeige und Steuertasten für den LinuxServer beschrieben.
Wie man das Gehäuse baut ist in September 2002, Frequenzzähler 1Hz−100Mhz mit LCD Display und

RS232 Interface beschrieben. Das Gehäuse und die Frontseite, wie ich sie gebaut habe, kann man rechts sehen.

Testen

Wie bei jeder Schaltung, die man gebaut hat sollte man nicht alles auf einmal an die volle Spannung anschließen. Es ist viel besser, schrittweise alles zu testen. Man macht immer kleine Fehler und auf diese Weise kann man sie finden, ohne dass eine Rauchwolke aufsteigt.

1. Baue die Hauptplatine mit allen Teilen zusammen, aber stecke die ICs noch nicht in die Sockel.
2. Nimm eine 9V Batterie und schließe sie in der Schaltung an die Pins, die mit AC_POWER markiert sind, an. Plus an Pin 2, Minus an Pin 1. Nimm ein Voltmeter und prüfe, dass du +5V an dem max232 zwischen 8 und 16 hast und an dem Mikrocontroller Pin 7 und 8. An den OPs sollten fast 9V liegen.
3. Jetzt die Batterie umdrehen (Pin 1 Plus) und prüfe, dass du ca. −9V an der negativen Stromversorgung der OPs hast.
4. Wenn das geklappt hat, funktioniert die Stromversorgung für die Hauptplatine und max 232 sowie Mikrocontroller können in die Fassung gesteckt werden.
5. Nimm wieder die 9V Batterie und schließe sie so an, daß +5V vorhanden sind (siehe oben). Schließe das Programmierkabel an den Parallelport und die Hauptplatine an.
   Entpacke die Software (siehe Referenzen am Ende), "cd" in das Verzeichnis, das beim Entpacken entsteht und dann:
   make avr_led_lcd_test.hex
   make testload
   make ttydevinit

   Nun sollte die Testsoftware in der Schaltung geladen sein. Auf dem LCD sollte "hello" stehen, die rote LED sollte blinken und wenn du die RS232 Schnittstelle anschließt, solltest du sehen, dass "ok" gedruckt wird. (initialisiere die rs232 Verbindung mit ttydevinit, und tippe dann cat /dev/ttyS0, oder cat /dev/ttyS1 für COM2).
6. Bau nun den Hochstromteil zusammen, aber schließe den Transformator noch nicht an. Anstelle des Transformators schließe die 9V Batterie an. Egal, in welcher Richtung die Batterie angeschlossen ist, der 4700uF Kondensator sollte sich immer bis auf 9V aufladen. Überprüfe das mit einem Voltmeter.
7. Wenn der letzte Test erfolgreich war, sollte man nochmal alle Kabel prüfen und dann die Transformatoren anschließen. Ohne Operationsverstärker (OP) im Sockel sollte man maximale Ausgangsspannung erhalten. Bitte aufpassen, dass man keinen Kurzschluss verursacht, da ohne OPs keine Strombegrenzung vorhanden ist und die Leistungstransistoren bei einem Kurzschluss durchbrennen würden..
8. Schalte wieder aus und stecke die Operationsverstärker in die Schaltung. Schließe nun auch wieder das Programmierkabel an. Einschalten und dann folgende Zeilen tippen:
   make
   make load
9. Nun sollte das Netzteil voll funktionieren. Beachte, dass, solange das Programmierkabel angeschlossen ist, die Ausgangsspannung leicht falsch ist. Entferne das Kabel, um exakte Werte zu erhalten.

Da ist es: Unser eigenes Netzteil

Du hast oben gesehen, dass es 3 Versionen gibt. Die unveränderte Software ist für 16V, 2.2A. Um das zu ändern, muß man die Datei linuxdcp.c editieren und nach MAX_U, IMINSTEP, MAX_I, suchen. In der

Funktion set_i muß man die Kalibrierung ändern, wenn man 3A max. Ausgansstrom hat. Der Code enthält entsprechende Kommentare, die alles erklären.

Nun noch einige Bilder von der Stromversorgung, wie ich sie gebaut habe. Es ist einiges an Arbeit, aber es ist wirklich ein gutes und robustes Netzteil. Zeit und Mühe sind gut investiert, da ein Netzteil wirklich eines der wichtigsten Geräte ist.

Das Netzteil benutzen

Es ist sicherlich fast offensichtlich, wie man das Netzteil benutzt. Es gibt 4 Taster zum Verändern der Spannung. 2 Taster, um die Spannung in 1V Schritten zu ändern und 2 Taster für 0.1V Schritte. Die Strombegrenzung kann mit 2 Tastern verändert werden. Hier sind die Stufen nicht linear. Für kleinere Werte kann man den Strom in 50mA Stufen ändern und für Werte über 200mA in 100mA Stufen. Über 1A in 200mA Schritten. Auf diese Art ist es einfach mit nur 2 Tastern durch den ganzen Wertebereich zu kommen.
Mit Standby kann man die Spannung ausschalten, aber Anzeige und Einstellungen bleiben.
Die rote LED geht an, wenn die Strombegrenzung anspricht und sie blinkt im Standby.
Das Netzteil kann völlig über die RS232 Schnittstelle gesteuert werden. Folgende Ascii Befehle stehen zur Verfügung:

u=X Setzt die Spannung (z.B. u=105 setzt die Spannung auf 10.5V)
i=Xmax setzt den Maximalstrom (z.B. i=500 heißt 500mA)
s=1 oder s=0 standby
u=? oder i=? oder s=? druckt die augenblicklichen Einstellungen. Das sieht so aus:
u: 50 s:0 i: 100 l:0
u: bedeutet Spannung=50 =5V, s:0 bedeutet standby aus, i: 100 ist 100mA, und l:0 bedeutet, dass die Strombegrenzung nicht erreicht ist.

Mit diesen ASCII Befehlen als Sprache könnte man auch eine grafische Bedienungsschnittstelle für das Netzteil schreiben. Um die Rs232 Verbindung zu nutzen, muß man sie erst initialisieren. Das geht mit dem Befehl ttydevinit, der im Softwarepaket enthalten ist. Das ist auch in September 2002, Frequenzzähler 1Hz−100Mhz mit LCD Display und RS232 Interface beschrieben.

Wie du in dem Schaltplan gesehen hast, benutzen wir zwei Trafos und die Masseleitung der Steuerlogik ist mit dem Plus−Ausgang verbunden. Die zwei Trafos trennen die Spannung und es gibt normalerweise kein Problem. Wir müssen das so beschalten, um die richtige Polarität für die Regelkreise zu haben.
Eine Warnung: Das bedeutet auch, daß die Masseleitung der Rs232 mit dem Plus−Ausgang verbunden ist! Mit anderen Worten, man kann die Rs232 Schnittstelle nicht benutzen, wenn man das Netzteil und eine Schaltung anschließt, die auch irgendwie mit der Masseleitung des Computers verbunden ist. Es ist sicher eine gute Idee einen Aufkleber auf das Netzteil zu kleben, auf dem steht: "Die Masse−Leitung der RS232 Schnittstelle ist intern mit dem Plus−Ausgang verbunden". Wenn du sicherstellen möchtest, daß du keinen Kurzschluss über die Masseleitung deines Computer verursachst, dann benutze entweder einen mit Batterie betriebenen Laptop oder nimm die RS232 Schnittstelle nicht, wenn die Schaltung noch irgendwo anders angeschlossen ist. Du brauchst jetzt nicht zu schockiert zu sein. Wenn die Strombegrenzung im Netzteil auf maximal 250mA steht, dann kann nichts passieren und das Netzteil wird dir sagen, wenn du einen Fehler gemacht hast (die rote Strombegrenzungslampe geht an). Unter 250mA besteht keine Gefahr für deinen Computer, selbst wenn du einen Fehler gemacht hast.

Sicherheit

Die Schaltung enthält Transformatoren, die ans Netz angeschlossen sind (230V oder 110V je nach Land). Bitte sorge für richtige Isolierung. Wenn das dein erstes Netzteil ist, dann bitte eine erfahrene Person, Sicherheit und Isolation zu prüfen, bevor die Schaltung mit dem Netz verbunden wird.

Tuning

Die Software für das Netzteil ist schon kalibriert. Vermutlich wirst du nichts ändern müssen. Auf der Hardwareseite hängt die Kalibrierung nur von 7805, R15, R10, R16 und R38, R30, R26 ab. Nur diese Bauteile bestimmen Strom und Spannung. Um die Kalibrierung zu ändern, kann man entweder diese Bauteile ändern oder die Software ändern. Beachte, dass das Programmierkabel die Einstellungen beeinflusst. Es muß entfernt werden, bevor man genaue Messungen macht. In der Software kann man die Funktionen set_u und set_i ändern. Kommentare im Code erklären das: linuxdcp.c

Referenzen

1. Die AVR UISP Programmiersoftware: www.amelek.gda.pl/avr/
   lokale Kopie: uisp−20011025.tar.gz
2. Wie man die Linux AVR Entwicklungsumgebung installiert und wie man die Programmierer−Hardware baut:
   März 2002, Den AVR Mikrocontroller mit GCC programmieren
3. Der Quellcode für diesen Artikel: linuxdcpower−0.1.tar.gz, 1201K . Der Schaltplan, die Eagle Dateien und Fotos sind auch enthalten.
4. Alle Software (Updates werden hier auftauchen) und Dokumente :Software/Datenblätter
5. Datenblatt für bd379 bd379.pdf 44K
6. Datenblatt für TL082 TL082.pdf 110K
7. Datenblatt für TL071 TL071.pdf 268K
8. Datenblatt für 2n3055 2n3055.pdf 64K

9. Datenblatt für MAX232 MAX220−MAX249.pdf 448K
10. Datenblatt für ST232, eine billige Variante des echten MAX232: st232.pdf 100K
11. Datenblatt für Atmel AT90S4433 avr4433.pdf 2356K
12. Atmel Webseite: www.atmel.com/
13. Eagle für Linux cadsoftusa.com

2005−01−11, generated by lfparser_pdf version 2.51