Hochschule Karlsruhe
Technik und Wirtschaft
UNIVERSITY OF APPLIED SCIENCES

Projektarbeit II

Energie- und Automatisierungstechnik

E 6311

WS 2005/06

Thema: WLAN-gestützter Regler

Betreuer: Prof. Dr.-Ing. Fehrenbach

Bearbeiter:
Josef Dan Laza
Steffen Ziegler
Arthur Hardt

1. Inhaltsverzeichnis

2. Einführung

2.1. Aufgabenstellung

2.1.1. Ursprüngliche Aufgabenstellung

Thema: WLAN-gestützter Regler

Es gibt inzwischen mehrere WLAN-Router mit Firmware aus dem Opensource-Bereich. Diese Systeme verfügen über Schnittstellen wie RS232 oder USB. Damit verfügt man über ein preiswertes Embedded-System für Aufgaben aus dem Bereich Messen-Steuern-Regeln. Die Firmware lässt sich durch den Einsatz von OpenWRT an die eigenen Bedürfnisse anpassen.

Aufgaben: Mit Hilfe eines WLAN-Routers ist eine einfache Temperaturregelung aufzubauen.

Dabei sind folgende Aufgaben zu lösen:

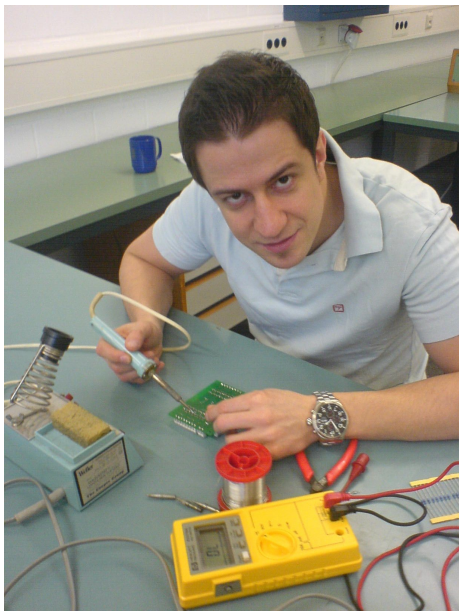
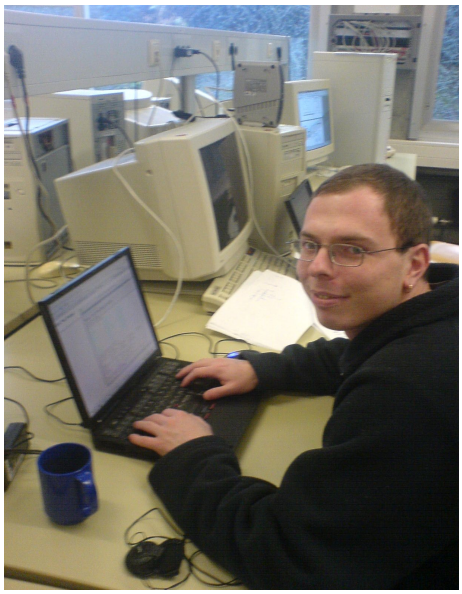
- Es ist eine Cross-Compiler-Umgebung für den verwendeten Prozessor (MIPS-CPU) in Betrieb zu nehmen.
- Es ist eine Umgebung aufzubauen mit der die Firmware des Routers leicht auf den neuesten Stand gebracht werden kann (siehe auch http://cyberforat.squat.net/openwrt/OpenWrt-HOWTO/x67.html).
- Für analoge und digitale Ein-/Ausgabe sind 1-wire-Module zu verwenden in Verbindung mit OWFS (1 -Wire-Filesystem, http://owfs.sourceforge.net).
- Es ist eine Temperatur-Regelstrecke aufzubauen. Das Einsatzszenario ist mit dem Betreuer abzusprechen.
- Es ist ein Regler zu konzipieren und umzusetzen.
- Die Regelgröße ist per WLAN auf einen Host-PC zu übertragen und anzuzeigen.

2.1.2. Änderungen in der Aufgabenstellung

Auf dem Dach des E-Gebäudes der Hochschule Karlsruhe befindet sich eine kleine Solaranlage, die das neue Studienobjekt wird. Die Temperaturregelstrecke entfällt aus der Aufgabenstellung. Sie wird ersetzt durch:

- An dieser Solaranlage gibt es Messwertaufnehmer für ca. acht relevante Größen.
- Alle Signale sind auf Messumformer geführt deren Ausgang eine Ausgangsspannung von 0 bis 10 V hat.
- Diese Signale sollen mit AD-Wandlern für den 1-Wire-Bus ausgelesen und mit Tcl/Tk dargestellt werden.

2.2. Bearbeiter

Josef Dan Laza	
Steffen Ziegler	
Arthur Hardt	

3. Rahmenbedingungen

3.1. Erste Schritte

Arbeitsumfeld: Als Projektraum steht uns das Labor LiU09 zur Verfügung. Diverse Rechner, Arbeitsunterlagen, Werkzeuge, etc. können dort hinterlassen werden und ein zentraler Treffpunkt ist somit vorhanden. Der Raum ist für unautorisierte Personen nicht zugänglich. Eigene Schlüssel können im Rektorat beantragt werden.

Arbeitsmittel: Uns wird ein Rechner (unten in Kapitel 3.2 als Rechner 1 aufgeführt) der Hochschule mit Internetzugang zur Verfügung gestellt. Ansprechpartner hierfür ist Herr Gantner (Li028), der für die IT-Systeme zuständig ist. Ein Laserdrucker befindet sich im Projektraum. Messgeräte und sonstige Hilfsmittel (Kabel, Werkzeuge, etc.) können bei Herrn Sudermann (Li021) ausgeliehen werden.

Zielsetzung: Das Ziel ist möglichst schnell ein Vorgehensweise auszuarbeiten, um in dem vorgegebenen Zeitraum zu einem positiven Ergebnis zu gelangen.

Zeitplan: Um den zeitlichen Verlauf des Projektes zu dokumentieren wird eine Projektplanung mit dem Meilenstein-Verfahren erstellt. Dieser definiert die einzelnen Daten der Teilaufgaben sowie den möglichen Abschlusstermin. Zuerst wollten wir die Projektarbeit noch vor Weihnachten zu ende bringen. Da es aber mit bestimmten Komponenten Lieferschwierigkeiten gab, mussten wir diesen Termin verschieben. Als Termin für die Abschlusspräsentation haben wir den 24.02.2006 festgelegt. Die Dokumentation soll dann ca. eine Woche später fertiggestellt werden.

Arbeitshergang: Das Arbeitslabor (LiU09) dient zur Forschung und Entwicklung der Hard- und Software. Zur Auswertung des Projektverlaufes werden in regelmäßigen Abständen Statusrunden mit allen Beteiligten. Im Projektraum wird die meiste Zeit investiert um Fortschritte in Bezug auf Softwareprogrammierung, Hardwareentwicklung, Routereinstellungen usw. zu erzielen. Erst zum Schluss haben wir unsere Hardware in die bestehende Anlage eingebaut.

Archivierung: Alle Projektdaten werden auf einer Backup-CD gesammelt um mögliche Verluste durch Rechnerabstürze etc. zu vermeiden.

3.2. Verwendete Hardware

Rechner 1: (LI U09)

- CPU: Intel Pentium II 400Mhz
- RAM: 256 MB
- HDD: 10 GB + 30 GB
- Optisches LW: CD-ROM
- Schnittstellen: 2x COM, 1x LPT, 2x USB 1.1, 2x USB 2.0
- Betriebsysteme: Windows XP, Xandros Desktop 2.0

Dieser PC wurde bereits bei unserer ersten Projektarbeit verwendet.

Laptop 1: (Privat)

- IBM Thinkpad R32
- CPU: Intel Pentium 4 Mobile 1600 Mhz
- RAM: 512 MB
- HDD: 20GB
- Optisches LW: DVD/CD-ROM
- Schnittstellen: 1x LPT, 2x USB 1.1
- Betriebsysteme: Windows XP, Fedora Core 4

Laptop 2: (Privat)

- IBM Thinkpad A22e
- CPU: Intel Celeron 800Mhz
- RAM: 256 MB
- HDD: 20GB
- Optisches LW: DVD/CD-ROM
- Schnittstellen: 1x COM, 1x LPT, 1x USB 1.1
- Betriebsysteme: Windows XP, Ubuntu Linux 5.10

WLAN-Router:

- ASUS WL-500G Deluxe
- CPU: 200MHz (BroadCom BCM4710)
- RAM: 32 MB
- Speicher: 4MB Flash ROM
- WAN: Anschluss für DSL Modem
- LAN: 4 LAN Ports (100 MBit/s Ethernet Switch)
- WLAN: 54MBit, Reverse SMA Anschluss + Antenne
- Schnittstellen: 2x USB 2.0 (Es sind noch 2x USB 2.0 Anschlüsse intern vorhanden die aber nicht herausgeführt wurden)
- LEDs: für Linkaktivität auf allen Ports (4xLAN,WAN,WLAN) und Power
- Und ein kleiner versenkter Taster auf der Rückseite
- Stromversorgung: Schaltnetzteil 5V bei 2A max.
- Über einen kleinen passiven USB HUB können auch mehrere Geräte angeschlossen werden.

3.3. Verwendete Software

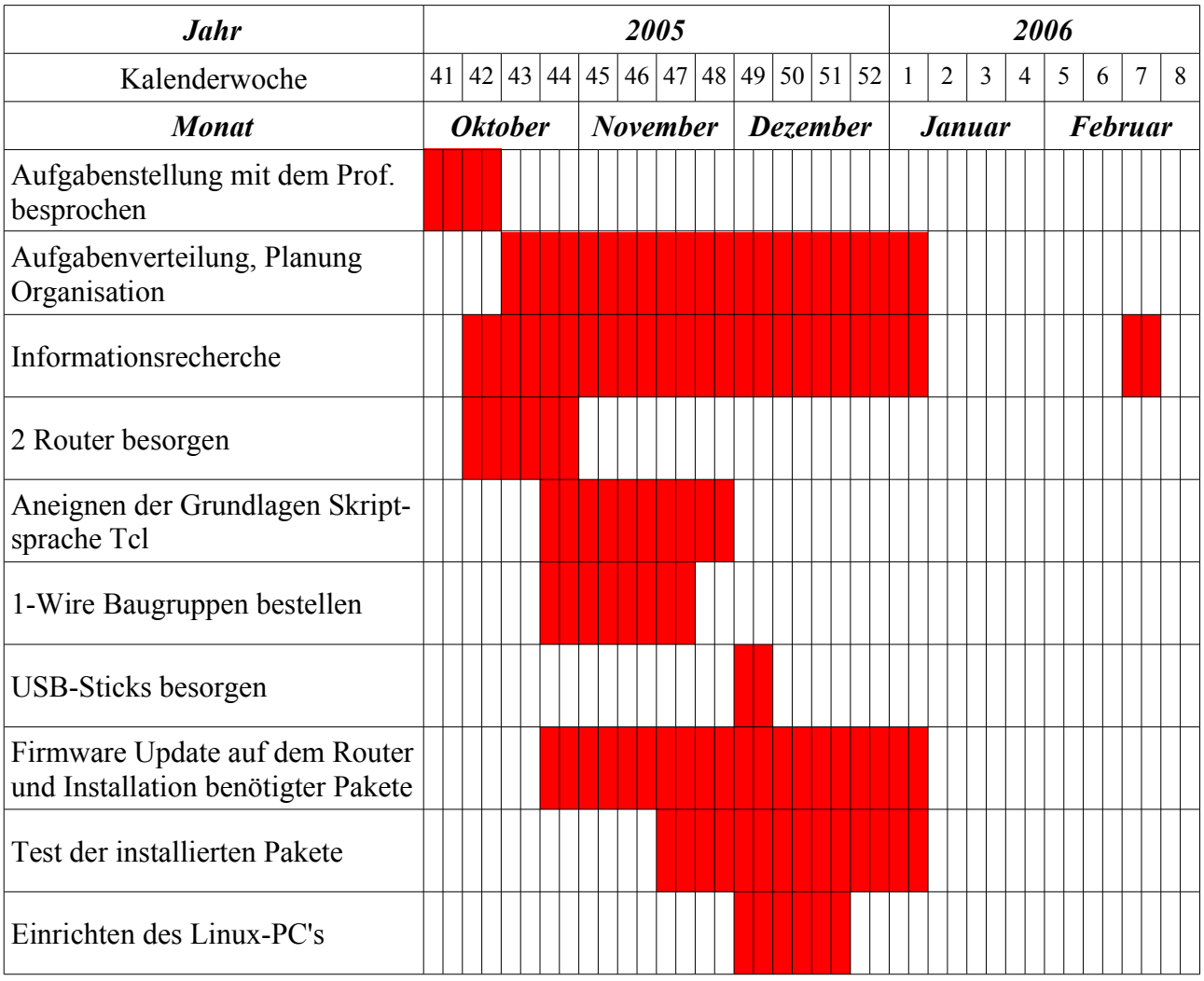
Die unten aufgelistete Software wurde nachinstalliert, mitgelieferte Software ist nicht aufgeführt:

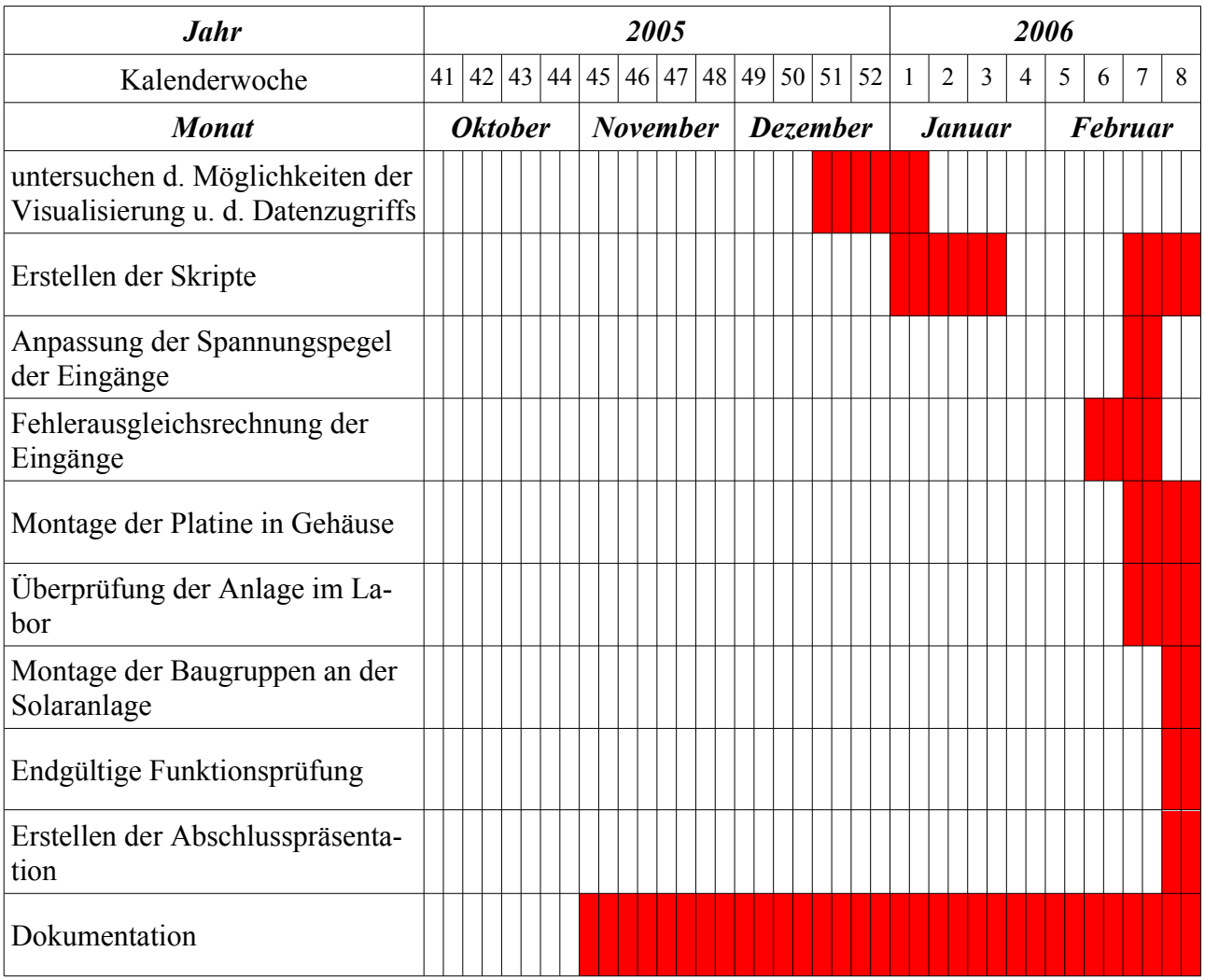
- OWFS-Life system (One Wire File System)
- OpenOfficeOrg 2.0 - Download unter: http://www.openoffice.org
- SSH Secure Shell oder PuTTY - Download unter: http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html
- Firmware Restoration Tool von Asus
- vi (Editor unter Linux)
- Edit Plus
- Firefox
- Opera
- XnView

3.4. Verteilung der Aufgaben

Aufgabe	*Wer?*
Router ASUS WL 500g Deluxe besorgen	Josef Dan Laza
OpenWRT auf dem Router installieren, konfigurieren und zusätzliche Pakete installieren, incl. Einbindung der USB-Sticks	Arthur Hardt
Tcl/Tk Programierung	Josef Dan Laza
Auswahl und Bestellung von 1-Wire-Geräten	Steffen Ziegler
CGI- Schnittstelle mit TCL; Programmieren der Ausgabe Seite.	Steffen Ziegler
Anpassung der analogen Eingangspegel an der Platine	Josef Dan Laza
Fehlerkorrektur der analogen Eingänge	Arthur Hardt
Montage der Platine in Gehäuse und verdrahten	Steffen Ziegler Josef Dan Laza
Ermittlung der Pinbelegung am 25-poligen sub-D Stecker	Arthur Hardt
Dokumentation	Alle

3.5. Zeitplan





3.6. Kosten des Projekts

	Beschreibung	Lieferant	Einzelpreis	Menge	Preis
1.	ASUS WL-500G Deluxe WLAN Router	KM Elektronik	89,79 €	2	179,58 €
2.	Dallas DS9490 [1)] USB to 1-Wire/iButton Adapter	Dallas/Maxim	0,00 €	1	0,00 €
3.	Dallas DS2890 [1)] 1-Wire Digital Potentiometer	Dallas/Maxim	0,00 €	2	0,00 €
4.	Dallas DS18B20 [1)] Programmable Resolution 1-Wire Digital Thermometer	Dallas/Maxim	0,00 €	2	0,00 €
5.	Dallas DS2482-800 [1)] Eight-Channel 1-Wire Master	Dallas/Maxim	0,00 €	2	0,00 €
6.	Dallas DS2413 [1)] 1-Wire Dual Channel Addressable Switch	Dallas/Maxim	0,00 €	2	0,00 €
7.	Dallas DS9502 [1)]	Dallas/Maxim	0,00 €	2	0,00 €

	Beschreibung	*Lieferant*	*Einzel-preis*	*Menge*	*Preis*
	ESD Protection Diode				
8.	Dallas DS2405 [1)] Addressable Switch	Dallas/Maxim	0,00 €	2	0,00 €
9.	Treckstor USB2.0 128MB USB-Stick	Saturn	12,99 €	2	25,98 €
10.	Verteilerdose [2)]			1	
11.	Platine mit Temperatursensor [2)]			1	
12.	Platine mit digitalen Ein- und Ausgängen [2)]			1	
13.	Platine mit 12 analogen 1-Wire-Eingängen [2)]			1	
14.	10kΩ ± 0,1% Messwiderstände [2)]			16	
15.	9V Block Batterien [3)]			4	
16.	RJ11 Leitung [2)]			1	
17.	RJ11 Stecker [3)]			1	
18.	Telefonleitung (in m) [3)]			2	
19.	25-pol. Sub-d Verlängerung [4)]				
					205,56 €

[1)] kostenlose Ansichtsexemplare
[2)] von Herrn Wolfrum bekommen (Preis unbekannt)
[3)] von der Werkstatt (Preis unbekannt)
[4)] von RZ (Preis unbekannt)

3.7. Zeitaufwand des Projekts

Jahr	*Monat*	*Steffen Ziegler*	*Josef Dan Laza*	*Arthur Hardt*	*Gesamt*	
2005	Oktober	15	15	24	54,0	16%
	November	29	17	40	86,0	26%
	Dezember	15	31	24	70,0	21%
2006	Januar	4	4	2	10,0	3%
	Februar	42	29	41	112,0	34%
Gesamt		105	96,0	131,0	332,0	
		32%	29%	39%		

Der Gesamtaufwand für das Projekt waren ca.330 Stunden.

4. Informationsrecherche

4.1. Einführung in Linux

4.1.1. Allgemein

Linux ist ein „freier“ Kern für Computer-Betriebssysteme. Da der Quelltext des Systems frei verfügbar ist, kann es im Gegensatz zu proprietären Systemen von jedem nach Belieben verändert und angepasst werden. Der Name Linux ist abgeleitet von dem Vornamen des Initiators, Linus Torvalds, und dem oftmals als Anspielung auf Unix genutzten.

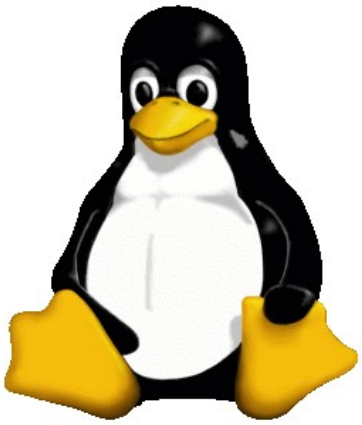


Abbildung 1: Das Linux-Maskottchen ist ein Pinguin namens Tux

Im engeren Sinne bezeichnet Linux nur den Linux-Kernel. Für den praktischen Einsatz eines Linux-Systems ist aber weitere Software notwendig, die dann mit dem Kernel zu einem Gesamtpaket gebündelt wird. Diese "Linux-Distribution" genannten Systeme greifen insbesondere auf das GNU-System des GNU-Projektes zurück, weshalb einige Entwickler diese Software-Bündel auch als GNU/Linux und nur den Kernel als Linux bezeichnen. Neben dem GNU-System wird auch die Software vieler anderer freier Software-Projekte häufig mit Linux ausgegeben. Beispiele sind dafür das X-Window-System X.Org-Server und die Desktopumgebungen KDE und GNOME.

Entwickelt wird der ursprünglich von Linus Torvalds geschriebene und später unter einer freien Lizenz veröffentlichte Betriebssystem Kern heute von Software-Entwicklern auf der ganzen Welt, die von Torvalds koordiniert werden. Torvalds selbst, der auch die Markenrechte für Linux hält, ist für diese Aufgabe ebenso wie der Kernel-Entwickler Andrew Morton beim Open Source Development Lab (OSDL) angestellt. Eine weitere Persönlichkeit der Kernel-Entwicklung und -Wartung ist Marcelo Tosatti.

4.1.1.1. Geschichte

Linus Torvalds Linux wurde in der Version 0.01 am 17. September 1991 von dem damals 21-jährigen finnischen Studenten Linus Torvalds erstmals öffentlich verfügbar gemacht. Seit dem hat es sich mit der Hilfe zahlreicher Entwickler und Nutzer auf der ganzen Welt stetig weiter entwickelt zu einem vollständigen Betriebssystem-Kern für verschiedenste Einsatzgebiete. Während die damalige Entwicklung nur einzig und allein durch die Community geleistet wurde, sind heutzutage auch Firmen und gemeinnützige Institutionen beteiligt, die ihre jeweils eigenen Interessen verfolgen, und auf diese Art und Weise Linux weiter voran bringen.

Doch von Anfang an gab es auch Gegenwind: Andrew Tanenbaum stellte dem System nur eine kurze Lebenszeit in Aussicht, Microsoft wurde sich der Konkurrenz bewusst, und die Firma SCO begann mit einem langjährigen Rechtsstreit um die eigentlichen Rechte an dem Quellcode von Linux.

4.1.1.2. Der Kernel

Der Linux-Kernel unterliegt ständigen Änderungen und einer ständigen Entwicklung. Die mittlerweile vierstellige Versionsnummer hat für jede Stelle eine exakt umrissene Bedeutung, die wiederum auf Stabilität und Entwicklungsstand zurückschließen lässt. Verschiedene Generationen des Kernels werden dabei von verschiedenen Entwicklern verwaltet, gepflegt und auch parallel weiter entwickelt. Als größte und bedeutendste Neuentwicklung der letzten Jahre ist dabei der Versionsübergang von Kernel 2.4 auf Kernel 2.6 zu betrachten, der neben einem neuen Scheduler zahllose weitere wichtige Neuerungen mit sich brachte. Dabei ist der alte wie der neue Kernel nicht auf eine spezielle Computer-Architektur beschränkt, sondern gehört mittlerweile zu den meist portierten Betriebssystemkernen überhaupt.

4.1.1.3. Distributionen

Schon kurz nach der Veröffentlichung von Linux haben Entwickler den eigentlichen Kern des Systems mit weiterer Software zusammen veröffentlicht, um das System und seine Fähigkeiten zu erweitern. Die so entstandenen Linux-Distributionen sind eine Zusammenstellung von hauptsächlich freier Software und enthalten neben dem eigentlichen Linux-Kernel noch weitere Software, um ein vollständiges Betriebssystem zur Verfügung stellen zu können und besser jeweiligen Ansprüchen an das Gesamtsystem gerecht zu werden.

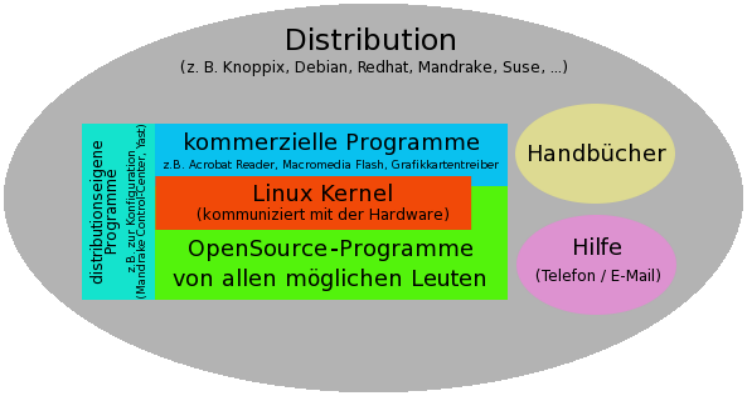


Abbildung 2: Distributionen

Dabei können sich sowohl die Hintergründe der einzelnen Distributionen als auch die Softwarezusammenstellung und der Zweck unterscheiden. Eine Auswahl der passenden Distribution für den jeweiligen Einsatz ist so von verschiedenen Faktoren abhängig.

4.1.1.3.1. Ubuntu

Ubuntu ist ein benutzerfreundliches GNU/Linux-Betriebssystem, das auf der Distribution Debian basiert. Ubuntu wird von Canonical Ltd gesponsert. Am 1. Juli 2005 wurde die Ubuntu Foundation mit einem Startkapital von 10 Mio. US-Dollar ins Leben gerufen.

Das Wort Ubuntu kommt aus den Sprachen der Zulu und der Xhosa. Es steht für „Menschlichkeit“ und „Gemeinsinn“, aber auch für den Glauben an ein „universelles Band des Teilens, das alles Menschliche verbindet“. Die meisten der ungefähr 40 hauptberuflichen Ubuntu-Entwickler kommen aus den Debian- und GNOME-Gemeinschaften.

Basisdaten	
Entwickler:	Ubuntu Foundation
Version:	5.10 (13. Oktober 2005)
Stammbaum:	\ Debian \ Ubuntu
Architekturen:	x86, ppc, x86-64
Lizenz:	GPL und andere Lizenzen
Sonstiges:	Preis: kostenlos
Sprache:	mehrsprachig
Desktop:	GNOME
Website:	www.ubuntulinux.org

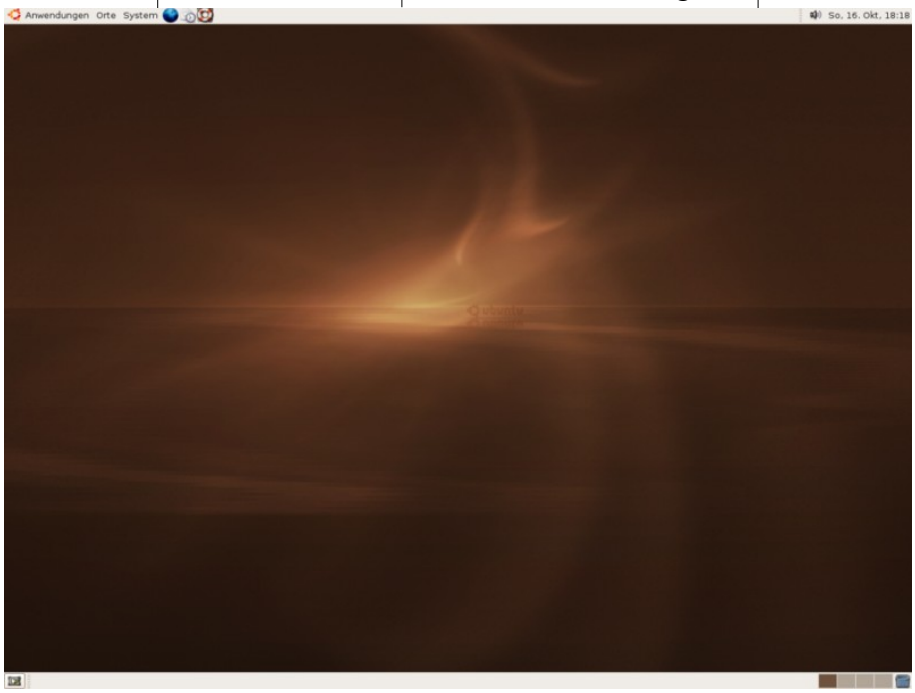


Abbildung 3: Ubuntu Breezy mit GNOME 2.12

4.1.2. OWFS

OWFS (One Wire Filesystem) erlaubt einfache Steuerung des 1-Wire Bussystems unter Linux. Der Bus kann entweder in einem Web Browser (OW-Httpd) oder als Dateisystem (OWFS) angesehen werden. Dieses bildet die Grundlage um die Signale der 1-Wire-Geräte auszulesen oder zu schreiben, dies ist moglich mit Shell-Skripten, Perl, C, Tcl, uvw. Programmiersprachen.

4.1.3. OpenWRT und Opennet

Bei OpenWRT handelt es sich um eine Linux-Distribution vor allem für WLAN-Geräte der Firma Linksys, insbesondere für die Routermodelle WRT54G und WRT54GS. Andere Broadcom-basierte Router werden mittlerweile auch unterstützt (z.B. Asus WL500g deluxe). Es wird an der Portierung von weiteren Geräten mit anderen Boards gearbeitet.

Da die Firma Linksys für ihre Router eine Software modifiziert hat, die als frei verfügbarer Code unter der Lizenz GPL steht, hat sie gemäß dieser Lizenz die eigene Software wiederum veröffentlicht. Dadurch war es möglich, das Betriebssystem des Routers wiederum zu modifizieren und weiterzuentwickeln und als OpenWRT zu veröffentlichen.

Bei der aktuellen OpenWRT-Version handelt sich um ein sehr kleines Linuxsystem, das auf dem 2.4.30er Linux-Kernel aufbaut. Eine OpenWRT-Installation besteht im allgemeinen aus 2. Teilen. Der 1. Teil ist ein circa 2 MB große Bereich, in dem alle Programme für die Grundfunktionen des Routers untergebracht sind. Dazu zählen zum Beispiel: Programme zur Netzwerkkonfiguration, NAT und Firewall sowie ein Editor und verschiedene Diagnosetools. Dieser Bereich kann nicht beschrieben werden. Die Größe des 2. Teil ist abhängig von der Hardware und ist gewöhnlich 2 bis 6 MB groß und kann im laufenden Betrieb beschrieben werden. Dadurch ist die Software beliebig erweiterbar und erlaubt Linux typischen Zugriff und Bedienung. Durch die Debian-ähnliche Paketarchitektur ipkg ist die Installation weiterer Software sehr einfach.

Es gibt mehrere Vereine die eine modifizierte Firmware mit Openwrt-Grundlage anbieten, z.B. die Opennet Initiative e.V. oder freifunk.net. Um eine positive Entscheidung zu treffen welche Firmware für unsere Projektarbeit die bessere ist haben wir erst einmal die Vor- und Nachteile der verschiedenen Firmwares angeschaut und mit unsere Prioritäten Verglichen. Da es keinen großen unterschied zwischen der Freifunk- und der Opennet-Firmware gibt, haben wir uns aus Zeitgründen entschieden nur eine von beiden unter die Lupe zu nehmen.

Abbildung 4: Logo der Opennet Initiative e.V.

OpenWRT RC3	
Vorteile	***Nachteile***
Die neuste verfügbare Version ist die RC3	Kein Webinterface vorhanden

Alle Unterschiede in den Paketen ließen sich einfach nachinstallieren oder updaten mit dem ipkg-Befehl.

Opennet 0.9.1b	
Vorteile	***Nachteile***
Es gibt viele Anleitungen in Deutscher Sprache sowie ein Forum: http://forum.opennet-initiative.de/	Ist nicht auf der neusten Version von OpenWRT aufgebaut nur auf RC2
Web-Interface für die Netzwerkkonfiguration	
Web-Interface ist in Deutscher Sprache	

Das die OpenWRT-Firmware kein Web-Interface besitzt ist nicht weiter tragisch da man auch alle Einstellungen und Statusabfragen über die Komanndozeile machen kann. Aber in Anbetracht unserer Linuxkenntnisse ist ein Webinterface von großem Vorteil für uns. Da auch der Speicher den beide Systeme benötigen ungefähr gleich sind haben wir uns für die Firmware der Opennet Initiative e.V. entschieden.

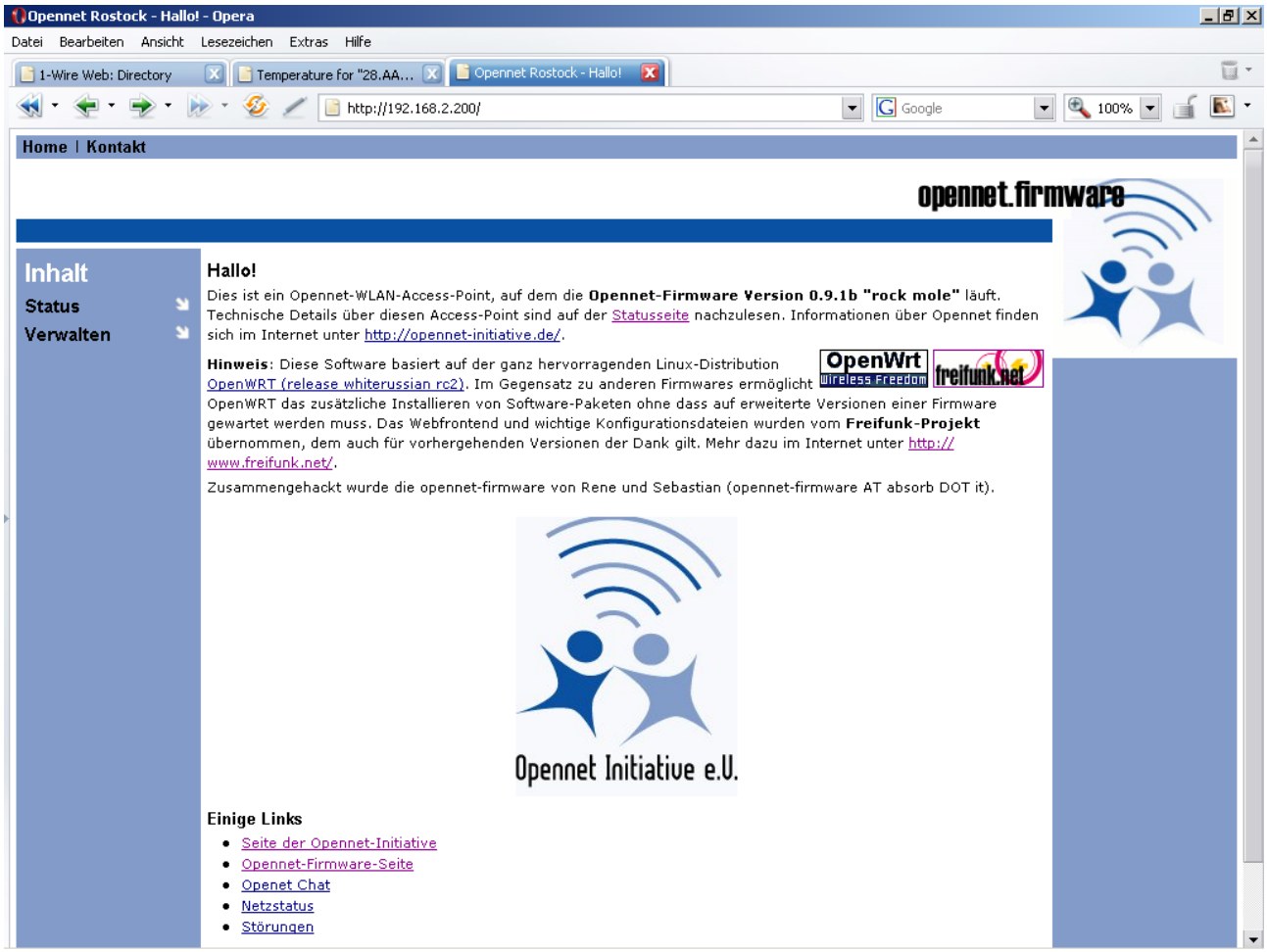


Abbildung 5: Webinterface der Opennet Initiative e.V.

Generell sind beide Distributionen als squashfs- und jffs2-Version erhältlich. Vor- und Nachteile der verschiedenen Versionen.

squashfs-Version		***jffs2-Version***	
Vorteile	***Nachteile***	***Vorteile***	***Nachteile***
Beim erneuten flashen durch eine neue Firmware, bleiben alle Einstellungen und Programme erhalten	Kein direkter Schreibzugriff direkt auf Konfigurationsdateien vorhanden.	Schreibzugriff direkt auf Konfigurationsdateien vorhanden.	Beim erneuten flashen durch eine neue Firmware, gehen alle Einstellungen und Programme verloren

4.1.4. Midnightcommander MC

Midnight Commander ist ein DateiManager - ein Clone des Norton-Commander für Linux/Console. Für Linux-Einsteiger, die auch schon unter DOS, Windows oder OS/2 ähnliches benutzt haben, eine sehr angenehme Arbeitserleichterung. Einer seiner größten Vorteile ist seine Vielfalt: Maus-Unterstützung (GPM), eingebauter FTP-Client, Entpacken von sämtlichen Archiven und Paketen (die

Programme sind natürlich vorher zu installieren) tar.gz, bzip, zip, rar, rpm, deb, ...Es mag ja sein, dass vieles per Kommandozeile schneller geht, aber mit mc ist alles einfach simpler, z.B. beim Kopieren, Löschen etc. von Daten, die sich mit Filtern nicht so einfach erfassen lassen. Besonders erfreulich für Anfänger, ist der eingebaute Editor mit Syntaxunterstützung für so ziemlich jede Sprache und einfacher Handhabung.

Da es eine Konsoleanwendung mit Funktionstasten ist, geht nach Eingewöhnung vieles schneller von der Hand als auf Kommandozeile oder grafischer Oberfläche. Er ist in vielen Dingen den GUI-Dateimanagern überlegen, insbesondere, was die Geschwindigkeit angeht.

FAQ's unter: http://www.ibiblio.org/mc/FAQ

4.1.4.1. Wichtige Tasten

F1 .. F10	siehe Menüleiste ab unteren Bildschirmrand
Tab	Wechseln auf anderes Panel
Strg-O	Wegklappen der Panels, so dass der normale Bildschirminhalt sichtbar wird
STRG+x c	Infomodus im anderen Fenster
STRG+x q	Quickview im anderen Fenster
STRG+s / ALT+s	Namenssuche im aktuellen Fenster
Esc-P	vorheriges Kommando
Esc-Enter	aktuellen Dateinamen auf Kommandozeile übernehmen
Esc-Tab	Kommando-Vervollständigung (ggfs. auch zweimal)
ALT-o	Fenster angleichen (anderes Fenster erhält gleichen Pfad!)
ALT-c	Wechsel in ein anzugebendes Verzeichnis
ALT+TAB	Shell-Erweiterung (nicht unter X)
ALT+?	Dateisuche
ALT+t	(togle) Wechsel zw. verschiedenen Listing-Arten
+ / -	select / unselect (pattern matching)

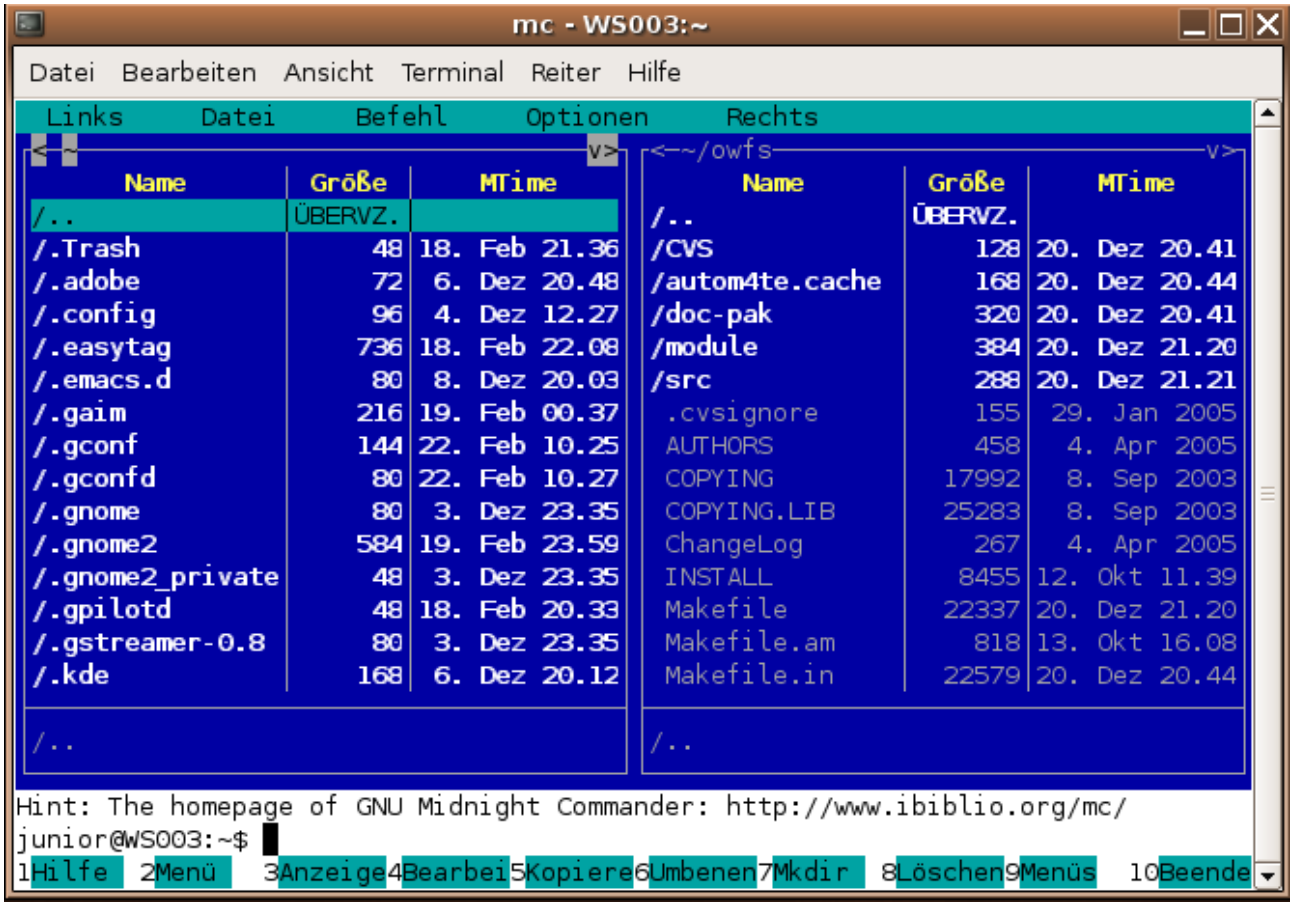


Abbildung 6: Midnight Commander

4.1.5. apt (Advanced Package Tool)

APT (Advanced Package Tool) ist der Name einer speziellen Software. Es handelt sich dabei um ein Paketmanagment-System, das im Bereich des Betriebssystems Debian GNU/Linux entstanden ist. Mittels APT ist es sehr einfach, Programmpakete zu suchen, zu installieren oder auch das ganze System komplett auf den neuesten Stand zu bringen.

4.1.5.1. Interna

In der Datei /etc/apt/sources.list stehen die sogenannten Repositories (engl. Lager, Depot), also Quellen für Pakete. Dies können entweder CDs oder DVDs, Verzeichnisse auf der Festplatte oder, öfter, Verzeichnisse auf HTTP- oder FTP-Servern sein. Befindet sich das gesuchte Paket auf einem Server (oder einem lokalen Datenträger), so wird dieses automatisch heruntergeladen und installiert.

Die Pakete liegen im .deb Paketformat vor, in dem auch die jeweiligen Abhängigkeiten der Programmpakete untereinander abgelegt sind. So werden automatisch für ein Programm auch eventuell erforderliche Programmbibliotheken mit herunter geladen und installiert.

4.1.5.2. Anwendungsbeispiele

apt-get install paketname installiert ein Paket und sämtliche Abhängigkeiten.
apt-get upgrade bringt alle Pakete auf den neuesten Stand.
apt-get update holt die neuesten Informationen über Pakete von dem Debian-Server.
apt-cache search suchwort sucht nach Programmen.
Will der Benutzer beispielsweise Gnome installieren:

```
# apt-get install gnome
Paketlisten werden gelesen... Fertig
Abhängigkeitsbaum wird aufgebaut... Fertig
Die folgenden zusätzlichen Pakete werden installiert:
  abiword-common abiword-gnome bluefish evolution gnome-office gtkhtml3.0
  libenchant1 libgal2.0-6 libgal2.0-common libgtkhtml3.0-4 libpq3 libsoup2.0-0
  planner
Vorgeschlagene Pakete:
  abiword-plugins abiword-plugins-gnome abiword-doc weblint gnome-spell
  postgresql-doc postgresql-client
Empfohlene Pakete:
  abiword abiword-help gnucash
Die folgenden NEUEN Pakete werden installiert:
  abiword-common abiword-gnome bluefish evolution gnome gnome-office
  gtkhtml3.0 libenchant1 libgal2.0-6 libgal2.0-common libgtkhtml3.0-4 libpq3
  libsoup2.0-0 planner
0 aktualisiert, 14 neu installiert, 0 zu entfernen und 136 nicht aktualisiert.
Es müssen noch 16,3MB von 17,9MB Archiven geholt werden.
Nach dem Auspacken werden 53,8MB Plattenplatz zusätzlich benutzt.
Möchten Sie fortfahren? [J/n]
```

4.1.6. Linux Befehle

apropos	Zeigt die Titel von man-pages zu einem gegebenen Stichwort an
ar	Archiv- und Bibliotheksverwaltung
basename	Gibt einen Dateinamen ohne Pfadangaben aus
cat	Fügt mehrere Textdateien zusammen
cd	Wechselt in das angegebene Verzeichnis
chmod	Zugriffsrechte ändern (rwx)
clear	Löscht die Konsole
cmp	Vergleicht zwei Dateien auf Übereinstimmung
compress	Komprimiert Dateien im Lempel-Ziv Verfahren
cp	Kopiert Dateien und Verzeichnisse (copy oder xcopy unter MS-DOS)
cpio	Kopiert Dateien in bzw. aus Archiven
dirname	Gibt nur den Pfad zu einer Datei aus
echo	Gibt einen Text auf der Konsole aus.
env	Gibt alle Umgebungsvariablen aus
exit	Aktuelle Session verlassen
file	Zeigt den Dateitypen einer Datei an.
find	Durchsucht den Verzeichnisbaum, ausgehend vom aktuellen Verzeichnis, nach einer Datei.
find	Umfangreiches Suchtool
gzip	Komprimiert und dekomprimiert Dateien im Lempel-Ziv Verfahren
init	Runlevel wechseln
locate	Sucht Dateien mittels eines Indexes, der durch updatedb erstellt wird

ls	Zeigt den Inhalt des aktuellen oder des angegebenen Verzeichnisses an (dir unter MS-DOS)
man	Zeigt man-pages an. (man-pages sind Seiten aus dem Linux Programmer's Manual. In diesem Manual sind so gut wie alle Programme auf Linux/Unix Basis dokumentiert)
mkdir	Legt ein neues Verzeichnis an (md unter MS-DOS)
mv	Verschiebt Dateien und Verzeichnisse (benennt auch um) (ren unter MS-DOS)
pwd	Zeigt das aktuelle Verzeichnis an
rm	Löscht Dateien und Verzeichnisse (del unter MS-DOS)
rmdir	Löscht ein Verzeichnis (rd unter MS-DOS)
strings	Extrahiert alle (lesbaren) Zeichenfolgen aus einer Datei/Eingabe
su	Neue Session mit anderem Account aufmachen (su - : root-Account).
sudo	Befehl als root ausführen.
tar	Komprimiert und dekomprimiert Archive mehrerer Dateien
unzip	Dekomprimiert zip-Archive.
updatedb	Erstellt einen Suchindex über das gesamte Dateisystem für locate
zip	Komprimiert Dateien. zip-Archive werden auch von PKZIP und WinZip (DOS/Win) verwendet

4.1.7. Editor vi

Vi ("vi" für "visual"; ausgesprochen [vi: ai], aber nicht [vi:] und auch nicht "sechs" oder "six" wie die römische Zahl VI) ist ein 1976 von Bill Joy für eine frühe BSD-Version geschriebener und von POSIX standardisierter Texteditor. Der Name stammt von "Visual Interface", weil sein Vorgänger ex ein einfacher Zeileneditor war.

Vi wurde schnell zum De-Facto-Standardeditor unter Unix, jedenfalls bis zum Aufstieg von Emacs um etwa 1984. Aufgrund seiner Schlankheit startet er schneller und benötigt weniger Speicherplatz als Emacs. Sogar auf einer Rettungsdiskette hat Vi auch heute noch Platz, so dass entweder Vi selbst oder einer seiner Klone (Vim, Nvi, elvis, ...) Bestandteil fast aller Unix-/Linux-Distributionen ist.

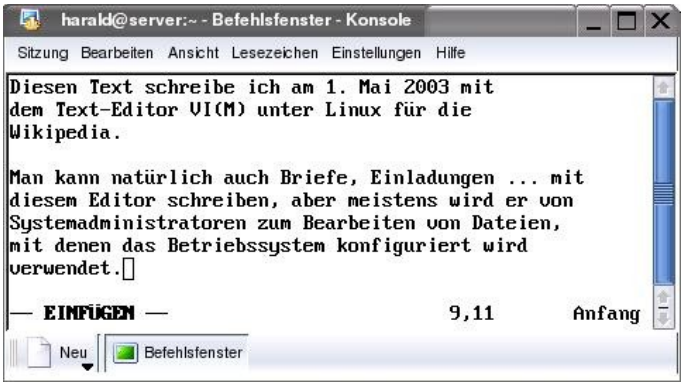


Abbildung 7: Editor vi

1991 benutzten ungefähr die Hälfte aller Teilnehmer einer Usenet-Umfrage den Vi. Auch heutzutage ist Vi zumindest in der Unixwelt noch sehr verbreitet, selbst Emacs-Benutzer greifen ab und zu auf ihn zurück, wenn es mal schnell gehen soll. Außerdem kann man mit diesem Editor in Kombination mit rsh oder ssh (früher mit Telnet) im Netzwerk sehr gut auf anderen Rechnern arbeiten.

4.1.7.1. Vi-Arbeitsmodi

Neueinsteiger brauchen für Vi eine hohe Frustrationstoleranz, da es sich um einen Editor mit drei grundsätzlich unterschiedlichen Arbeitsmodi handelt. Dabei ist es ganz einfach, wenn man erst einmal die Arbeitsweise der einzelnen Modi verstanden hat und weiss, wie man von einem in den anderen Modus wechselt (siehe Grafik unten). Die drei Modi sind:

Befehlsmodus (command mode)

Beim Start von Vi befindet man sich im Befehlsmodus. Dort können durch verschiedene Tastendrücke einfache Befehle ausgeführt, wie z.B. "Wort suchen", "Zeile löschen" usw. Durch drücken von [i] gelangt man in den Einfüge-Modus.

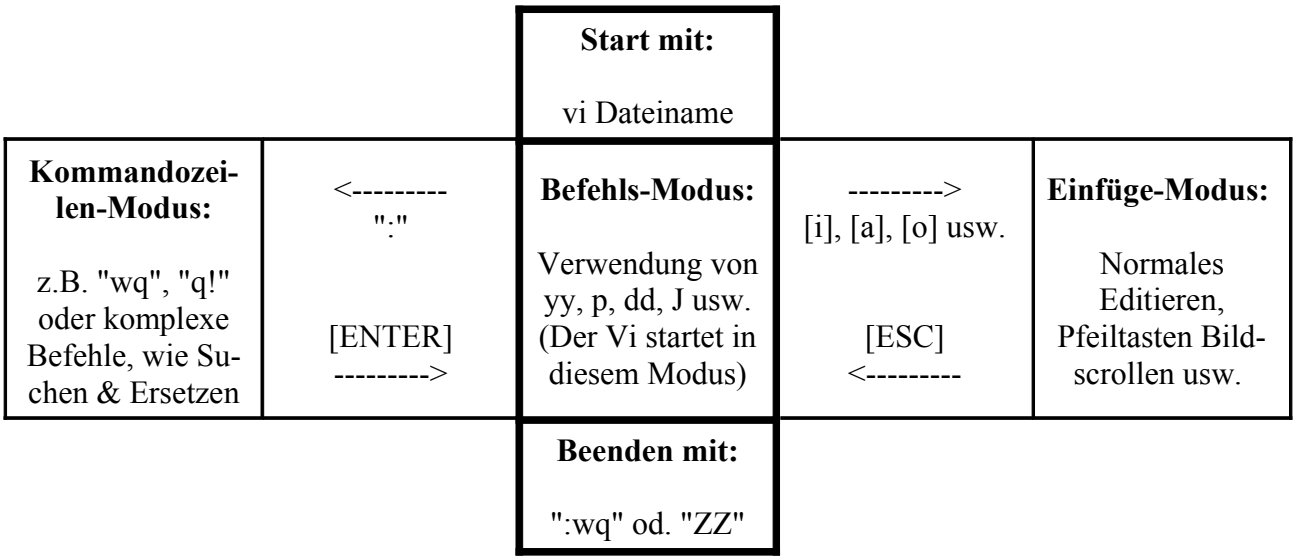
Einfügemodus (insert mode)

Im Einfügemodus ist die eigentliche Eingabe von Text möglich. Durch drücken von ESC gelangt man wieder in den Befehls-Modus zurück.

Kommandozeilenmodus (auch Komplexbefehlsmodus, colon mode oder ex mode)

Durch Eingabe von ':' gelangt man vom Befehlsmodus in den Kommandozeilenmodus. Dort können komplexere Befehle ausgeführt werden, wie etwa Suchen und ersetzen von Text (:s/alter_text/neuer_text/g).

Frühe Versionen gaben dem Benutzer kein Indiz, in welchem Modus er sich gerade befand, so dass es auch heute noch typisch für den Vi-Benutzer ist, immer vorsichtshalber noch einmal [ESC] zu drücken, um ganz sicher zu gehen, dass er sich wirklich im Befehlsmodus befindet (falls das vorher schon der Fall war, wird Vi einfach nur piepsen). Aktuelle Versionen von Vi deuten ihren augenblicklichen Modus auf der Statuszeile oder grafisch an (siehe Screenshot unterste Zeile - der Modus ist "EINFÜGEN").



(wenige Ausnahmen, wie z.B. das Zurückspringen des [r]-Befehls in den Befehlsmodus ohne drücken von [ESC], existieren)

4.1.7.1.1. Sichern und beenden

ZZ	Sichern und vi beenden
:wq	(write quit) Sichern und vi beenden
:w!	vi ohne Sicherung beenden
:w	Sichern
:w	dat In Datei „dat“ schreiben

4.1.7.1.2. Eingeben/Ändern im Eingabemodus

a	(append) Rechts vom Cuursor einfügen
A	(Append) Am Zeilenende anhängen
i	(insert) Links vom Cursor einfügen
I	(Insert) Am Zeilenanfang einfügen
o	(open) In neuer Zeile danach einfügen
O	(Open) In neuer Zeile davor einfügen
s	(substitute) Cursorzeichen ersetzen
S	(Substitute) Ganze Cursorzeile ersetzen
R	(Replace) Überschreiben einschalten
cw	(change word) nächstes Wort ersetzen
ncw	(change word) nächsten n Worte ersetzen
cc	(change) ganze Zeile ersetzen
C	(Change) Rest der Zeile ersetzen

4.1.7.1.3. Löschen, kopieren und verschieben

dw	(delete word) nächstes Wort löschen
ndw	(delete word) nächsten n Worte löschen
dd	(delete) ganze aktuelle Zeile löschen
ndd	nächsten n Zeilen löschen
d/was	Text bis zum nächsten „was“ löschen
dG	ab Cursor bis Dateiende alles löschen
D	Zeilenrest löschen
x	Zeichen an Cursorposition löschen
nx	nächsten n Zeichen löschen
X	Zeichen vor Cursor löschen
yy	kopiert aktuelle Zeichen in allgemeinen Puffer
nyy	kopiert nächsten n Zeilen in allgemeinen Puffer
y$	kopiert ab Cursor Zeilenrest in allgemeinen Puffer

dw	(delete word) nächstes Wort löschen
>%	Text bis korrespond. Klammer einrücken
„xyw	kopiert nächstes Wort in Puffer x
„ayy	kopiert aktuelle Zeile in Puffer a
„xnyy	kopiert nächsten n Zeilen in Puffer x
„add	löscht akt. Zeile und kopiert sie in Puffer a
„xndw	n Worte löschen und in Puffer x kopieren
p (put)	allgemeinen Puffer hinter Cursor kopieren
P (Put)	allgemeinen Puffer vor Cursor kopieren
xp	vertauscht zwei Zeichen
„xp	Puffer x hinter Cursor kopieren
„xP	Puffer x vor Cursor kopieren
:r	dat Datei „dat“ hinter aktuelle Zeile kopieren

4.1.7.1.4. Änderungen rückgängig machen

u	(undo) macht die letzte Änderung rückgängig
U	(Undo) Änderungen in akt.Zeile zurücknehemen
:e!	alle Änderungen seit letztem Sichern wegwerfen
:q!	vi ohne Sichern verlassen

4.1.7.2. Vorteile

Ein großer Vorteil von Vi ist, dass mehrere Befehle nacheinander ohne gleichzeitiges Betätigen der Alt-, Strg- oder sonstiger Modifier-Tasten abgesetzt werden können. Für den geübten Benutzer bedeutet das eine erhebliche Steigerung der Arbeitsgeschwindigkeit. So löscht zum Beispiel 3dw gleich drei Wörter auf einmal.

4.1.8. Automatische zeitsynchronisation - ntpclient

Das Network Time Protocol (NTP) ist ein Standard zur Synchronisierung von Uhren in Computersystemen über paketbasierte Kommunikationsnetze. Obwohl es meistens über UDP abgewickelt wird, kann es durchaus auch über andere Layer-4-Protokolle wie z.B. TCP transportiert werden. Es wurde speziell dazu entwickelt, eine zuverlässige Zeitgabe über Netzwerke mit variabler Paketlaufzeit (Ping) zu ermöglichen.

4.1.8.1. Grundlagen

NTP ist eines der ältesten noch immer verwendeten TCP/IP-Protokolle. Es wurde von David Mills an der Universität von Delaware entwickelt und 1985 als RFC 958 veröffentlicht. Unter seiner Leitung werden Protokoll und UNIX-Implementierung ständig von seinen Mitarbeitern weiterentwickelt. Gegenwärtig ist die Protokollversion 4 aktuell. Es benutzt den UDP Port 123.

NTP ist in UNIX-artigen Betriebssystemen in Form des Hintergrundprozesses ntpd implementiert. Dieser synchronisiert die lokale Uhr mit Hilfe von externen Zeitsignalen, die er entweder direkt von einem lokalen Empfänger (DCF77, GPS, Loran-C) oder per NTP von einem NTP-Server erhält. Damit die lokale Uhrzeit nicht nur zu den zyklischen Synchronisationszeitpunkten präzise mit dem externen Signal übereinstimmt, korrigiert der ntpd-Prozess nicht nur die Phase sondern auch die Frequenz des lokalen Zeitgebers mit Hilfe einer Software-PLL. Um den internen Zeitgeber mit Hilfe eines hochpräzisen Sekundensignals noch enger an einen externen Normalzeitempfänger zu koppeln haben einige UNIX-Varianten (unter anderem Linux und FreeBSD) die oben erwähnte Software-PLL im Kernel implementiert.

Die Zeitstempel im NTP sind 64 Bits lang. 32 Bits kodieren die Sekunden seit dem 1. Januar 1900 00:00:00 Uhr, weitere 32 Bits den Sekundenbruchteil. Auf diese Weise lässt sich ein Zeitraum von 232 Sekunden (etwa 136 Jahre) mit einer Auflösung von 2−32 Sekunden (etwa 0,25 Nanosekunden) darstellen. Obwohl diese Skala also alle 232 Sekunden umspringt, sind NTP-Implementationen in der Lage, die tatsächliche Zeit festzustellen, indem sie eine ungefähre Zeit aus anderen Quellen heranziehen. Da dies nur eine Genauigkeit von ein paar Jahrzehnten erfordert, sollte dies im Alltag kein Problem sein.

NTP nutzt ein hierarchisches System verschiedener Strata, wobei Systeme mit dem Stratum 1 direkt mit einer sehr genauen externen Uhr (z.B. eine GPS- oder andere Funkuhr) verbunden sind. Systeme mit dem Stratum 2 beziehen ihre Zeit von einem oder mehreren Systemen mit Stratum 1 usw. (Achtung: Der Begriff Stratum hat hier eine andere Bedeutung als sonst in der Telekommunikationstechnik üblich).

4.1.8.2. Algorithmus und Genauigkeit

NTP benutzt den Marzullo-Algorithmus (erfunden von Keith Marzullo von der Universität San Diego in dessen Dissertation) mit einer UTC-Zeitskala, und unterstützt Schaltsekunden. Durch die Betrachtung der Schaltsekunden im Protokoll kommt es dazu, dass mit jeder Schaltsekunde (welche jedoch selten vorkommen) eine neue Sekundenskala benutzt wird. Für die Skala der Systemzeit wird jedoch für gewöhnlich die tatsächlich vergangene Zeit seit einem bestimmten Zeitpunkt benutzt und Schaltsekunden kommen erst bei der Darstellung der Zeit ins Spiel.

NTPv4 kann die lokale Zeit eines Systems über das öffentliche Internet mit einer Genauigkeit von 10 Millisekunden halten, in lokalen Netzwerken sind unter idealen Bedingungen sogar Genauigkeiten von 200 Mikrosekunden und besser möglich. Bei einem hinreichend stabilen lokalen Taktgeber (thermostatgesteuerter Quarzofen, Rubidium-Oszillator etc.) lässt sich unter Verwendung der Kernel-PLL (siehe oben) der Phasenfehler zwischen Referenzzeitgeber und lokaler Uhr bis in die Größenordnung von wenigen Mikrosekunden reduzieren.

4.1.8.3. Parameter beim Starten

Benutzung: ntpclient [optionen]

Option	*Beschreibung*
-c count	stop after count time measurements (default 0 means go forever)
-d	print diagnostics (feature can be disabled at compile time)
-g goodness	causes ntpclient to stop after getting a result more accurate than goodness (microseconds, default 0 means go forever)
-h hostname	(mandatory) NTP server host, against which to measure system time
-i interval	check time every interval seconds (default 600)

Option	*Beschreibung*
-l	attempt to lock local clock to server using adjtimex(2)
-p port	local NTP client UDP port (default 0 means "any available")
-r	replay analysis code based on stdin
-s	simple clock set (implies -c 1)

4.1.9. Der Batchdämon crond

Der vielseitigste aller Dämonen ist der crond (gesprochen cron-d). Er führt beliebige Kommandos automatisch zu vorbestimmten Zeitpunkten aus, wie nach einem Fahrplan. Einmal pro Minute sieht der Dämon in seinen Terminkalender und führt zuverlässig und pünktlich alle anstehenden Kommandos aus.

Der unter Linux verbreitete crond von Paul Vixie (vixie-cron) erlaubt prinzipiell allen Benutzer, mit dem crontab-Kommando eine eigene Spalte in dem Terminkalender des Dämons einzurichten. Die in dieser Spalte eingetragenen Befehle werden dann „im Auftrag``, also mit der Benutzerkennung der Auftraggeber, ausgeführt.

Wie der Druckerdämon lpd wird der crond normalerweise während der Systeminitialisierung aus einer der rc*-Dateien aufgerufen. Der Dämon geht von selbst in den Hintergrund.

Der Terminkalender des Dämons befindet sich im Verzeichnis /var/spool/cron/crontabs und besteht aus beliebig vielen Spalten, jede in Form einer Datei mit dem Namen des Auftraggeber.

4.1.9.1. Der Terminkalender crontab

Die einzelnen Dateien für den Terminkalender des Dämons werden von den Auftraggebern verwaltet. Es handelt sich dabei um einfache ASCII-Textdateien, die beispielsweise mit dem elvis-Editor erzeugt werden können. Weil das ./crontabs-Verzeichnis des Dämons für normalsterbliche Systembenutzer nicht beschreibbar ist (es ist normalerweise nicht einmal lesbar), muß eine Vorlage für die Termindatei in einem anderen, beschreibbaren Verzeichnis angelegt werden. Die Veränderung des ./crontabs-Verzeichnisses wird dann vom crontab-Kommando mit Superuser-Rechten vorgenommen.

4.1.9.2. Die crontab-Datei

In einer Termindatei werden alle Zeilen, die weder leer sind noch mit einem #-Zeichen in der ersten Spalte als Kommentar markiert sind, vom Dämon bearbeitet. Solche Zeilen können entweder eine Umgebungsvariable für die Ausführung aller in dieser Datei aufgerufenen Kommandos definieren oder eine Zeitmaske mit zugehörigen Kommando enthalten.

Eine Zeitmaske besteht aus fünf Feldern, die durch Leerzeichen voneinander getrennt werden.

Die Zeitmaske:

Feldnummer	***1***	***2***	***3***	***4***	***5***
Bedeutung	Minute	Stunde	Monatstag	Monat	Wochentag
Bereich	0-59	0-23	0-31	0-12*	0-7*

6r*oder Namen

Die Monate und Wochentage können auch mit ihren englischen Namen angegeben werden. Die Namen können auf drei Zeichen abgekürzt werden, Groß-/Kleinschreibung wird ignoriert.

Bei Zahlendarstellung des Wochentages entsprechen 0 und 7 dem Sonntag.

Jedes Feld der Zeitmaske kann durch einen Asterisk `*' belegt werden, der auf jeden Termin paßt.

Die Felder können durch Komma getrennte Listen von Zeiteinträgen sowie Bereiche der Form von-bis enthalten.

In Bereichen dürfen keine Namen verwendet werden. Es können auch mehrere Bereiche aufgelistet werden. Zusätzlich können den Bereichen noch „Teiler`` zur Veränderung der Schrittweite nachgestellt werden.

Die restliche Zeile bis zum Zeilenende oder einem %-Zeichen wird als Kommando ausgeführt. Wenn ein %-Zeichen gefunden wird, das nicht durch einen Backslash entwertet ist, wird der Rest der Zeile als Eingabe an das Kommando geleitet.

4.1.9.3. Beispiele:

Eintrag	*Beschreibung*
0 8-18 0 * 1-5 /usr/lib/newsbin/input/newsrun	Das newsrun-Kommando wird montags bis freitags von 8 bis 18 Uhr zu jeder vollen Stunde aufgerufen.
0 17 24 12 * echo %schoene Bescherung	Heiligabend um 17 Uhr, egal was für ein Wochentag. Die Mitteilung wird per Mail an den Auftraggeber geschickt. In diesem Beispiel wird der Mitteilungstext nicht in der Kommandozeile an das echo-Kommando übergeben, sondern in seinen Standardeingabekanal geschrieben.
0-59/5 * * * ~/bin/remind	Das remind-Kommando wird alle fünf Minuten aufgerufen, jeden Monat, jeden Tag, jede Stunde ...
0 6,10,14,18,22 * * * /usr/lib/uucp/uucico -s uucphost	Das uucico-Kommando wird täglich um 6, 10, 14, 18 und 22 Uhr auf- und damit der uucphost angerufen.

Ein bestimmter Tag kann sowohl als Monatstag als auch als Wochentag bestimmt werden. Wenn beide Felder bestimmt, also nicht durch einen Asterisk besetzt sind, werden sie durch eine logische ODER-Verknüpfung ausgewertet.

4.1.9.4. Umgebungsvariable in der crontab-Datei

Wie bereits erwähnt, können in der crontab-Datei auch Umgebungsvariable für die Ausführung der Kommandos bestimmt werden. Die Variablen HOME, LOGNAME und SHELL werden von crond automatisch mit den Werten aus der passwd-Datei vorbelegt. Die Variable LOGNAME kann nicht verändert werden.

In der MAILTO-Variablen kann ein realer Account bestimmt werden, an den die Ausgabe der automatischen Kommandos geschickt wird. Wenn die Variable definiert, aber leer ist, geht die Ausgabe der Kommandos verloren. Wenn die Variable nicht definiert ist, wird die Ausgabe automatisch an den Eigentümer der Terminkalenderspalte gesendet.

Die Definition von Umgebungsvariablen erfolgt wie in normalen Shellscripts durch Zuweisung einer (möglicherweise leeren) Zeichenkette.

4.1.9.5. Das crontab-Kommando

Mit dem Anwenderkommando crontab werden die benutzerdefinierten Termindateien im Verzeichnis /var/spool/cron/crontabs verwaltet.

crontab erkennt folgende Optionen:

Option	***Beschreibung***
-u Benutzer	(user) legt den Benutzernamen fest, in dessen Auftrag eine Termindatei ausgeführt werden soll. Diese Option ist nur dem Superuser (root) zugänglich. Alle normalsterblichen Systembenutzer können (höchstens) ihre eigenen Termindateien bearbeiten.
-l	(list) zeigt den Inhalt der aktuellen Termindatei an.
-d	(delete) löscht die Termindatei aus dem Verzeichnis ./crontabs.
-r Datei	(replace) ersetzt die Termindatei im ./crontabs-Verzeichnis durch die angegebene.

Wenn im Verzeichnis /var/spool/cron die Datei allow existiert, kann das crontab nur von den darin aufgeführten Systembenutzern ausgeführt werden. Wenn anstelle der allow-Datei eine Datei mit dem Namen deny existiert, steht das crontab-Kommando allen Systembenutzern zur Verfügung, die NICHT darin aufgeführt sind.

Wenn keine der Dateien existiert, hängt das Verhalten von crontab von den Einstellungen bei der Compilierung ab. Entweder kann jeder das crontab-Kommando ausführen, oder nur der Superuser (root).

Wenn das ./crontabs-Verzeichnis vom crontab-Kommando verändert wird, liest der crond automatisch die neuen Daten, muß also nicht extra neu gestartet werden.

4.2. 1-Wire

4.2.1. Was ist ein 1-Wire Netzwerk?

Dallas Semiconductor/Maxim entwickelte eine sehr einfache Netzwerktechnologie namens 1-Wire. Auch unter dem Namen MicroLAN bekannt handelt sich dabei um ein "low cost“ Bus basierendes Netzwerk, welches ursprünglich entwickelt wurde um die Kommunikation naheliegender Geräte über einen einzelnen Pin eines Mikrocontrollerports zu ermöglichen. Dies war ein einfacher Weg die Speicherkapazität eines Mikrokontroller zu erweitern. Mit der Zeit wurde diese einfache Idee um viele Features erweitert und es entstand die 1-Wire Technologie.

Wie schon in der einfachen Version wird auch hier eine einzelne Leitung (und einen Bezugsleiter) für die Kommunikation und die Spannungsversorgung der einzelnen Teilnehmer verwendet. An einen Bus-Master können mehrere Slaves über ein einzelnes Twistet Pair Kabel angeschlossen werden. Für seine Identifizierung besitzt jeder Slave, egal welche Aufgabe er erfüllt, eine weltweit eindeutige und unveränderbare digitale Adresse. Die Anwendungspalette der 1-Wire Geräte reicht inzwischen von einfachen Speichern bis hin zu Java tauglichen Geräten.

4.2.2. Aufbau eines 1-Wire Netzwerks

Um die Funktionsweise eines 1-Wire Netzwerkes verstehen zu können, muss zuerst der Aufbau eines solchen geklärt werden.

4.2.2.1. Aufbau und Prinzip eines 1-Wire Netzwerks

Ein 1-Wire basierendes Netzwerk besteht grundsätzlich aus drei Teilen:

- dem Busmaster
- dem Verbindungselement
- dem 1-Wire Gerät

Der Busmaster mit der Kontrollsoftware übernimmt die Steuerung der Kommunikation im Netzwerk. Er könnte durch einen Standardmikrokontroller (z.B. 8051 von Intel) oder einen normalen PC repräsentiert werden. Das Verbindungselement stellt eine Verbindung zwischen dem Busmaster und den einzelnen 1-Wire Elementen her. Es besteht im einfachsten Fall aus einem Twistet Pair Kabel und den entsprechenden Anschlüssen für den Master und dem 1-Wire Element.

Das 1-Wire Gerät wird auch als Slave bezeichnet. In einem einfachen Netzwerk kann es vorkommen, dass nur eines davon existiert. Im Normalfall sind es natürlich mehrere Geräte. Bei diesen kann es sich um beliebige 1-Wire Geräte handeln, wie z.B. iButtons. Wichtig ist hierbei, dass jedes von ihnen eine weltweit eindeutige digitale Adresse hat, über die das Gerät angesprochen werden kann.

Bei einem 1-Wire Netzwerk handelt es sich vom Prinzip her um eine sehr einfaches Master- Slave Modell. Da die Kommunikation über ein einzelnes „Twisted Pair“ Kabel erfolgt, sind einige Einschränkungen erforderlich. Kein 1-Wire Gerät (Slave) darf sprechen bevor es vom Master dazu aufgefordert wird. Es gibt auch keine direkte Kommunikation zwischen den einzelnen Slaves. Sollte eine solche erforderlich sein, wird diese über den Master abgewickelt.

4.2.2.2. Topologie

Da 1-Wire Komponenten im Normalfall mit einem Twisted Pair Kabel aufgebaut werden kann, gibt es nur wenige verschieden Möglichkeiten um diese zu Verbinden. Die meisten 1-Wire Netzwerke lassen sich in folgende 3 Kategorien einordnen:

Lineare Topologie: Abbildung 8 zeigt einen solchen Aufbau. Das 1-Wire Netzwerk ist durch ein einzelnes Leitungspaar welches vom Master bis hin zum entferntesten Slave in einer Linie (ohne größere „Stubs“) verbunden.

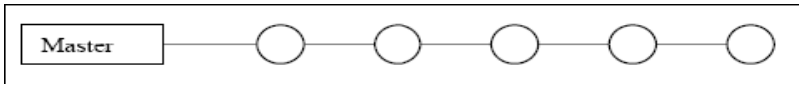


Abbildung 8: Lineare Topologie

"Stubbed“ Topologie: In Abbildung 9 ist eine solche Topologie dargestellt. Hier ist das Netzwerk aus einem Hauptleitungspaar, welches vom Master aus zum entferntesten Slave reicht, aufgebaut. Die anderen Slaves werden über so genannte „Stubs“, die länger als 3 Meter sind, an die Hauptleitung angeschlossen.

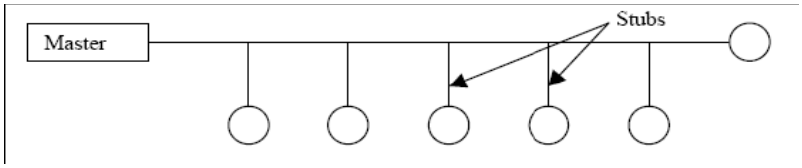


Abbildung 9: „Stubbed“ Topologie

Stern Topologie: Das Hauptleiterpaar wird am Masters (oder in seiner Nähe) aufgespaltet und in verschiedenen Zweige unterschiedlicher Länge fortgesetzt. Abbildung 10 zeigt eine Sterntopologie.

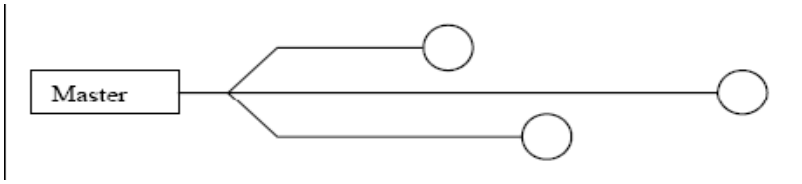


Abbildung 10: Lineare Topologie

Es ist natürlich auch möglich verschiedene Topologien untereinander zu mischen, allerdings wird es dadurch schwieriger ein zuverlässiges Netzwerk zu dimensionieren. Da viele Aspekte bei der Konstruktion eines zuverlässigen 1-Wire Netzwerkes zu berücksichtigen sind, wird der Aufbau eines solchen hier nicht näher erläutert.

4.2.3. Kommunikation im Netzwerk

Um vorweg Verwirrung über die verschiedenen Übertragungsraten und damit verbundene Zeitschlitze zu vermeiden, eine kurze Erklärung hierzu. Es gibt zwei verschiedene Geschwindigkeitsmoden. Die Normale Geschwindigkeit, bei dieser liegt ein Zeitschlitz zwischen 60µs und 120µs, und den sogenannten Overdrive Modus, bei dem liegt ein Zeitschlitz zwischen 6 und 16µs. Im Normalfall arbeitet ein 1-Wire Gerät im Normalmodus. Zur Verwendung des schnelleren Modus muss das Gerät dafür konfiguriert werden. Wie dies zu erfolgen hat ist dem entsprechenden Datenblatt zu entnehmen.

Daten in einem 1-Wire Netzwerk werden mittels Zeitschlitzen übertragen. Diese sind je nach Übertragungsgeschwindigkeit verschieden lang. Zum Beispiel bei einer Länge der Zeitschlitze von 120µs wird zur Übermittlung einer logischen Eins an einen Slave der Bus vom Master für maximal 15µs auf "low“ gesetzt. Der Rest des Zeitschlitzes wird die Leitung auf "high“ gesetzt. Solch eine Übermittlung ist in Abbildung 11 dargestellt.

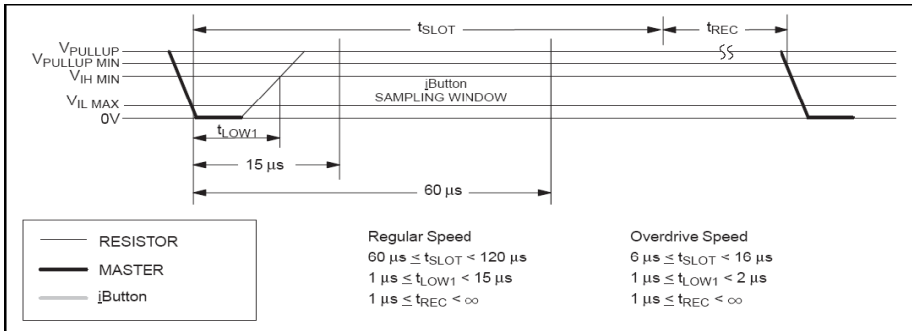


Abbildung 11: Zeitlicher Verlauf des Schreibvorgang einer logischen Eins vom Masterzum Slave

Um eine logische Null zu übermitteln setzt der Master den Bus für mindestens 60µs auf "low“. Dies ist in Abbildung 12 dargestellt. Am Ende jedes übertragenen Bits benötigt der Slave eine gewisse Zeit sich auf den Empfang des nächsten Bits vorzubereiten (ca. 1µs).

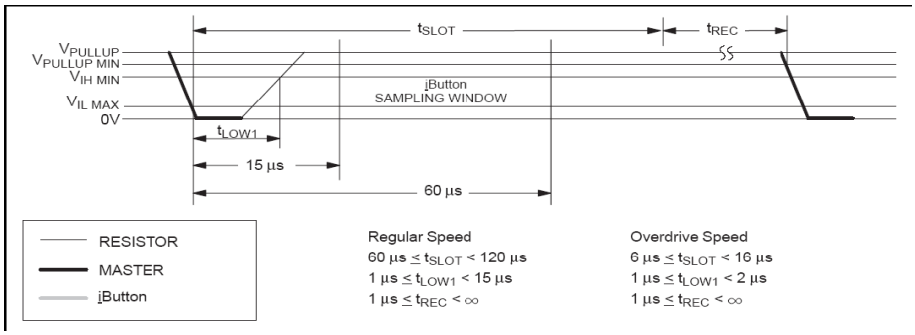


Abbildung 12: Zeitlicher Verlauf des Schreibvorgang einer logischen Eins vom Masterzum Slave

Das 1-Wire Netzwerk Protokoll verwendet dabei normale TTL- Pegel, d.h. bis 0.8 Volt wird als "low“ gewertet und ab 2.2 Volt als "high“. Um einen möglichst fehlerfrei Kommunikation zu ermöglichen tastet der Slave die Datenleitung ca. in der Mitte des Zeitschlitzes ab (siehe Abbildung 12 und 13).

Die oben erwähnten Schreibvorgänge dienen nur zur Übertragung der Daten vom Master zum Slave. Die Kommunikation in die andere Richtung (vom Slave zum Master) erfolgt folgendermaßen: der Master generiert Schreibzeitschlitze für den Slave. Diese haben die selbe Form wie der Zeitschlitz um eine logische Eins vom Master zum Slave zu schicken. Will der Slave eine logische Eins übertragen, lässt er den Zeitschlitz unverändert. Um eine logische Null zu übermitteln zieht der Slave die Leitung auf "low“ für die Zeit trdv. In Abbildung 13 ist dieser Vorgang dargestellt.

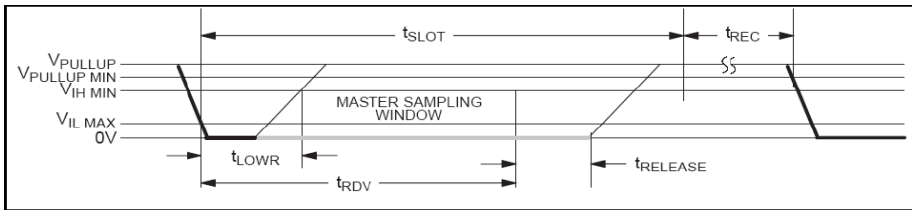


Abbildung 13: Zeitlicher Verlauf des Lesevorganges des Masters

Ein Systemtakt wird dabei nicht benötigt, da alle 1-Wire Bauteile über einen internen Oszillator selbst getaktet sind und dieser mit der fallenden Flanke vom Master synchronisiert werden.

4.2.3.1. Kommunikationsablauf / Protokoll

Da nun geklärt ist, wie die Daten zwischen Master und Slave ausgetauscht werden muss noch eine Protokoll festgelegt werden, um eine Kommunikation zu ermöglichen.

Um nun die Kommunikation zu starten wird vom Master ein Reset durchgeführt. Dies geschieht indem er den Bus für mindestens acht Zeitschlitze oder maximal 470µs auf "low“ setzt und danach wieder auf "high“. Die selbe Zeit wartet der Master. Innerhalb dieser Zeit antworten alle angeschlossenen Slaves indem sie die Leitung für mindestens 60µs auf "low“ setzen. Dieser Impuls wird auch als Anwesenheitsimpulses eines Slaves bezeichnet. Abbildung 14 stellt eine solche Vorgang dar.

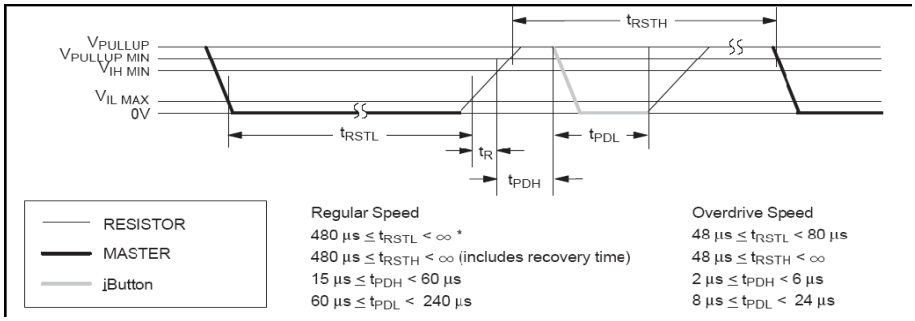


Abbildung 14: Zeitlicher Verlauf des Resetvorganges und des Anwesenheitsimpulses

Wird nun ein solcher Impuls vom Master empfangen, kann er über die eindeutige digitale Adresse jedes Slaves auf ihn zugreifen. Dies geschieht indem er zuerst den entsprechenden 8 Bit langen Kommandocode und anschließen die digitale Adresse des Slaves sendet. Nach dem er einen Slave für die Kommunikation ausgewählt hat, ist nur mehr dieser aktiv. Alle anderen Slaves werden deaktiviert, bis zum nächsten Resetimpuls des Masters. Über geräteabhängige 8 Bit lange Kommandos kann der Master nun auf die Daten bzw. Funktionen des 1-Wire Gerätes zugreifen und entsprechende Daten zwischen ihm und dem Slave austauschen. Abbildung 15 zeigt eine typische Kommunikationssequenz in einem 1-Wire Netzwerk.

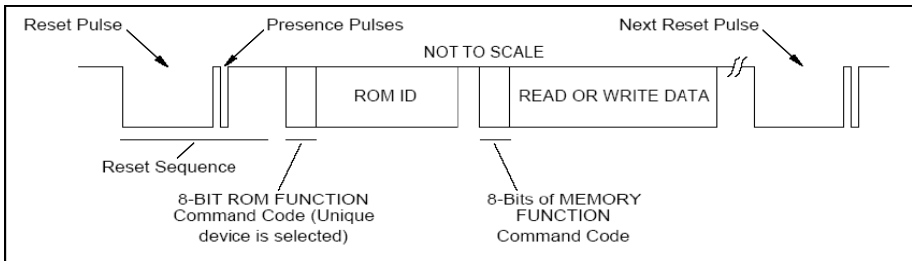


Abbildung 15: Typische Kommunikationssequenz in einem 1-Wire Netzwerk

4.2.4. Eigenschaften eines 1-Wire Netzwerks

Um ein 1-Wire Netzwerk aufzubauen müssen zuerst die Eigenschaften des Netzwerkes, und den Komponenten aus denen es aufgebaut ist, näher betrachtet werden. Diese Eigenschaften und auch damit verbundene mögliche Probleme sind hier aufgelistet und näher erläutert. Es werden auch Lösungsansätze für diese Probleme vorgestellt.

4.2.4.1. Länge des Netzwerkes, Anzahl der 1-Wire Geräte

Die wichtigsten Kriterien eines Netzwerks sind sicherlich die maximale Länge des Netzwerkes, die mögliche Anzahl der angeschlossenen Geräte und die Übertragungsgeschwindigkeit. Diese drei Eigenschaften werden hier nicht grundlos in einem Atemzug genannt, da zwischen ihnen bei 1-Wire Netzen ein enger Zusammenhang besteht. Leider kann aus diesem Grund auch keine allgemeine Aussage über diese drei Dinge gemacht werden. Ein kurzes Beispiel zur Erläuterung: Die Länge

und Anzahl der angeschlossenen Geräte steigern die Gesamtkapazität des Netzwerkes. Je größer diese ist, desto kleiner muss die Übertragungsgeschwindigkeit gewählt werden um eine problemlose Kommunikation zu ermöglichen. Leider sind dies nicht alle Faktoren die dabei eine Rolle spielen, auch die Eigenschaften des Masters (z.B. die Slew Rate) sind zu berücksichtigen.

4.2.4.2. Übertragungsgeschwindigkeit

Es gibt zwei verschieden Übertragungsmodi. Die genaue Übertragungsrate hängt von den gewählten Modus und der Länge der Zeitschlitze ab. Um einen Grenzwert zu erhalten wird der kleinste Zeitschlitz von 6µs herangezogen. Damit ergibt sich eine maximale Übertragungsrate von 20,8kB/s.

4.2.4.3. Adresse der 1-Wire Geräte

Wie schon erwähnt hat jedes 1-Wire Gerät eine weltweit eindeutige Identifikationsnummer (ID). Bei diese ID handelt es sich um eine 64 Bit lange Zahl, die bei der Herstellung in ein ROM geschrieben wird. Über diese ID wird das Gerät auch im Netzwerk identifiziert, daher kann man sie auch als Adresse des Gerätes bezeichnen. Die acht Bytes (64 Bit) der ID werden in drei Bereiche unterteilt und setzen sich wie folgt zusammen: Angefangen beim niederwertigsten Byte wird im ersten Byte der acht Bit lange „Family Code“ gespeichert, der den Gerätetyp spezifiziert. Die nächsten sechs Byte enthalten eine 48 Bit lange individuelle Adresse. Das achte und höchstwertigste Byte enthält eine Checksumme der ersten sieben Byte. Mit den für einen Gerätetyp 248 möglichen Adressen sollte genug verschieden Adressen zur Verfügung stehen.

4.2.4.4. Spannungsversorgung der Slaves

Eine Besonderheit des 1-Wire Systems ist, dass die meisten angeschlossenen Geräte keine extra Spannungsversorgung benötigen. Diese wird über die Anschlussleitung und den Master realisiert.

Um diesen Mechanismus zu erklären ist in Abbildung 16 der schematische Aufbau der Ein- / Ausgabeelement eines 1-Wire Gerätes dargestellt. Die Leitungen Data und Return sind die Anschlussleitungen vom Master (Return stellt dabei die Masseleitung dar). Befindet sich nun die Datenleitung auf „high“ (5 Volt) schaltet die Diode durch und lädt den internen 800pF Kondensator. Sinkt die Spannung an der Datenleitung unter die Spannung des Kondensators sperrt

die Diode und die Energie des Kondensators bleibt zur Versorgung des Gerätes erhalten. Dadurch wird erreicht, dass auch während der Zeitperiode in der die Datenleitung auf „low“ (0 Volt) ist, eine Spannungsversorgung des Gerätes möglich ist. Dieses Konzept des „Stehlen´s“ der Energie von der Datenleitung wird auch als „parasite power“ bezeichnet.

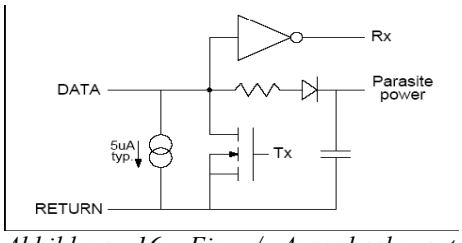


Abbildung 16: Ein- / Ausgabeelement eines 1-Wire Gerätes

4.2.4.5. Terminierung des Netzwerkes und Slew Rate des Masters

In einem typischen 1-Wire System welches über einen COM Port eines PC´s angesteuert wird, erfolgt die Kommunikation in Zeitschlitzen von 8.68µs gesteuert über den UART. Ein Kommunikationszyklus beginnt mit einem Reset des Masters. Dies geschieht durch einen Transistor, welcher die Leitung auf „low“ zieht. Diese Flanke von „high“ auf „low“ dient allen Slaves als Synchronisationsflanke. Da im Normalfall in einem 1-Wire Netzwerk mehrere Geräte installiert sind, erhalten diese den Synchronisationsimpuls (durch Signallaufzeiten) zu leicht unterschiedlichen Zeitpunkten.

Um die Funktion zu gewährleisten muss ein Signal ans Ende des Netzwerkes (der Leitung) und wieder zurück geschickt werden können. Dadurch entsteht die Einschränkung, dass die elektrische Länge des Netzwerks kleiner als die Hälfte eines Zeitschlitzes zur Übertragung eines Datenbits sein muss. Für den Fall eines COM Port Adapters bei dem die Zeitschlitze 8.68µs sind, würde das eine Ausbreitungszeit von 4.34µs bedeuten. Alle Geräte die hinter dieser Grenze liegen, werden vom Master nicht mehr wahrgenommen.

Bei einem COM Port Adapter geschieht eine Umschaltung von „high“ auf „low“ im Sub-Mikrosekundenbereich. Falls das Umschalten länger braucht als das übertragene Signal ans Ende des Netzwerkes und wieder zurück, können die Reflektionen am Ende der Leitung die Kommunikation stören. Normalerweise würde man solche Störungen verhindern, indem man das Ende der Leitung mit einem entsprechenden Widerstand terminiert. Dieser Widerstand würde die überschüssige Energie in Wärme umsetzen und so Reflektionen am Ende der Leitung verhindern.

Leider ist es im Fall der 1-Wire Technologie nicht möglich mit einem einfachen ohmschen Widerstandes diese Terminierung durchzuführen. Dies hat mehrere Gründe, aber auf Grund der Komplexität dieser, wird auf eine nähere Erklärung verzichtet. Da es also nicht möglich ist das Ende des Netzwerkes zu terminieren, muss eine andere Lösung gefunden werden. Die Alternative ist die Slew Rate des „pull down“ Transistors am Master zu kontrollieren. Die geforderte Slew Rate ist natürlich wieder von der Länge des Netzwerkes abhängig und liegt bei Netzwerken mit 100 Meter (und mehr) bei 1.1 Volt pro Mikrosekunde. Dies heißt wiederum, dass einen umschalten von „high“ auf „low“ in ca. 4µs geschieht (das umschalten des Pegels geschieht bei 0.8 Volt).

Beim Transistor handelt es sich dabei um einen normalen n-kanal FET. Die Transistoreigenschaften spielen dabei eine nicht all zu große Rolle, es ist daher auch möglich einen bipolar Transistor einzusetzen (mit einer Modifikation der anderen Bauteilwerte).

4.2.4.6. „Pull-up“ Widerstand des Masters

Wenn Master und Slave die Leitung freigeben zieht der „Pull-up“ Widerstand die Leitung wieder auf die Betriebsspannung (auf „high“). Durch die Länge des Netzwerkes und der Anzahl der 1-Wire Geräte werden die kapazitiven Eigenschaften des Netzwerks festgelegt, d.h. je größer diese Kapazität wird, desto Länger dauern die Umladevorgänge und desto länger dauert es bis die Leitung wieder auf „high“ ist. Abbildung 17 zeigt diesen Effekt, bei einer fixen Länge von 100 Meter und einer steigenden Anzahl von Slaves (von 1 bis 300). Aus der Gesamtkapazität (Leitung-, Geräte- und Streukapazitäten) und dem „Pull-up“ Widerstand des Masters entsteht eine Zeitkonstante. Steigt dieser Wert über den der definierten Wert der Zeitschlitze der Kommunikation an, kommt es zu Störungen und sogar zum Ausfall der Kommunikation im Netzwerk. In Abbildung 17 ist das elektrische Ersatzschaltbild eines 1-Wire Netzwerkes dargestellt. Hier sieht man, dass man die Kapazität der „parasite power“ Spannungsversorgung (800 pF) eigentlich in die Berechnung der Zeitkonstante mit einfließen lassen müsste. Dies ist allerdings nicht der Fall, weil sich diese Kapazität erst bemerkbar macht, wenn sich die Leitung schon im "high“ Zustand befindet.

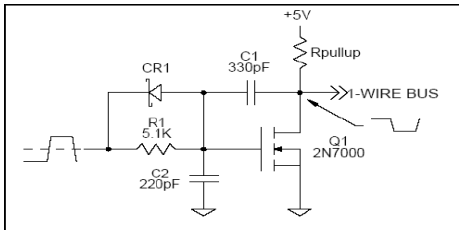


Abbildung 17: Schaltung zur Kontrolle der Slew Rate des Masters

Geht man nun davon aus, dass die Zeitschlitze 13.02µs lang sind (die urspr üngliche vorgesehene Länge der Zeitschlitze) und die Spannung einen Wert von 2.2 Volt erreichen muss um als „high“ zu gelten kann folgende Zeitkonstante berechnet werden.

$$\tau = \frac{13.02\,\mu s}{\ln\left(\frac{V_s}{V_s - 2.2\mathrm{V}}\right)} = 22.4\,\mu s$$

Bei Vs handelt es sich um die Versorgungsspannung die 5 Volt beträgt. Die Übertragungsrate bei der Verwendung eines Zeitschlitzes von 13.02µs beträgt dabei 9600 Bytes pro Sekunde.

Mit einem „Pull-up“ Widerstand des Masters von 1.5k und dem errechnetenWert der Zeitkonstante von 22.4µs darf die Gesamtkapazität des Netzwerkes maximal 12 nF betragen.

$$\tau = R * C \rightarrow C = \frac{\tau}{R}$$

Bei der Verwendung eines Kabels mit dem typischen Wert von 50 pF ergibt sich damit eine maximale Länge des Netzwerkes von 240 Meter. Dies ist allerdings nur in diesem Fall der maximale Wert der Länge des Netzwerkes. Würde man zum Beispiel ein anderes Kabel verwenden oder auch den „Pull-up“ Widerstand des Masters modifizieren, verändern sich die Eigenschaften dementsprechend. Die maximale Anzahl der angeschlossenen Geräte berechnet ist abhängig vom „Pull-up“ Widerstand des Masters (1.5k), der Versorgungsspannung (5 Volt), dem Spannungspegel ab dem sich die "parasite power“ Kapazität auswirkt (2.8 Volt) und dem maximalen Eingangsstrom der 1-Wire Geräte (15µA).

$$Fanout_{max} = \frac{5\mathrm{V} - 2.8\mathrm{V}}{1.5\mathrm{k}\,\Omega} = \frac{1.47\mathrm{mA}}{15\mu\mathrm{A}} = 98\mathrm{Geräte}$$

Man kann so errechnen, welchen maximalen Strom der Master über den „Pullup“ Widerstand liefern kann und dividiert diesen durch den maximalen Eingangsstrom der 1-Wire Geräte und erhält so die maximale Anzahl der anschließbaren Geräte.

Man sieht , dass die Eigenschaften eines 1-Wire Netzwerkes von vielen Faktoren abhängen sind. Ein wichtiger Faktor ist der „Pull-up“ Widerstand des Masters. Dieser sollte den oben angegebenen Wert von 1.5k nicht unterschreiten, da sonst das Netzwerk sehr störanfällig werden würde. Man kann jedoch anstatt eines einfachen ohmschen Widerstandes einen aktiven „Pull-up“ Widerstand

verwenden. Durch die Verwendung eines solchen, wird die Zeitkonstante wesentlich verbessert und das Umschalten von „low“ auf „high“ beschleunigt.

Dadurch werden natürlich auch die anderen Netzwerkeigenschaften verbessert. Dies bedeutet allerdings einen Mehraufwand beim Master.

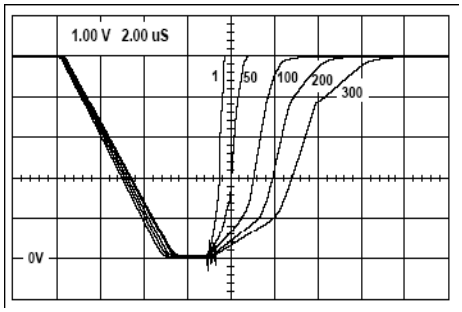


Abbildung 18: Spannungsverlauf beim Umschaltevorgang von „low" auf „high" miteiner unterschiedlichen Anzahl von Slaves

4.2.4.7. Das Kabel

Das 1-Wire Netzwerk kann zwar mit einem einzigen Leitungspaar aufgebaut werden, aber die Eigenschaften dieses Paares sind für den störungsfreien Betrieb wichtig. Es sollte ein Kabel der Kategorie 5 eingesetzt werden. In einem solchen Kabel sind meist mehrere Leitungspaare vorhanden. Diese sollten wenn möglich nicht angeschlossen werden (auch nicht geerdet). Auf jeden Fall darf kein zweites 1-Wire Netzwerk durch das selbe Kabel (an einem anderen Leitungspaar) betrieben werden. Dies führt mit ziemlicher Sicherheit zu einer Funktionsstörung in beiden Netzwerken.

4.2.5. Vor- und Nachteile von 1-Wire Netzwerken

Einer der größten Vorteile eines 1-Wire Netzwerkes ist die sehr einfache Verdrahtung. Durch die Verwendung von nur zwei Leitungen (Datenleitung und Masseleitung) können die angeschlossenen Geräte eine sehr robuste Bauform erhalten, da nur zwei Anschlüsse nötig sind. Durch die einfache Verdrahtung ist es relativ leicht möglich ein neues Netzwerk aufzubauen oder es auch in ein schon bestehendes Netzwerk bzw. Kabelsystem zu integrieren. Daher ist der Preis für die Verdrahtung eines solchen Netzwerkes relativ gering. Mit den geeigneten Bausteinen bzw. Software ist es auch möglich ein 1-Wire Netzwerk mit einem anderen zu verbinden. Abbildung 19 zeigt ein

solches Schema. Ein Schwachpunkt der 1-Wire Technologie ist die Übertragungsrate. Diese kann mit 20,8 kB/s mit heutigen schnellen Netzwerken sicher nicht mithalten. Ob eine höhere Übertragungsrate allerdings wirklich nötig ist, sei hier in Frage gestellt. Da alle 1-Wire Geräte eine weltweit eindeutige Adresse haben ist es nicht möglich ein 1-Wire Gerät vollständig selbst zu entwickeln und zu bauen. Es wird dazu immer ein Schnittstellenbaustein der Erzeugerfirma benötigt, welcher die digitale Adresse enthält. Andererseits ist mit diesem Baustein schon die Schnittstelle zum 1-Wire Netzwerk gegeben.

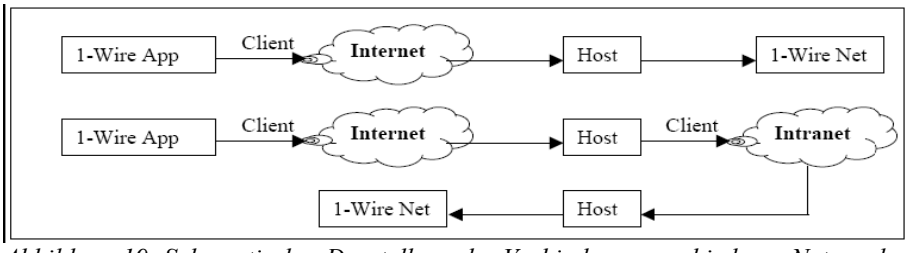


Abbildung 19: Schematischer Darstellung der Verbindung verschiedener Netzwerke

4.2.6. Vergleich 1-Wire Netzwerke mit anderen Netzwerken

Die Vor- und Nachteile eines 1-Wire Netzwerkes wurden in Kapitel 4.2.5 schon angeführt. Da es bei einem Netzwerk sehr auf den Verwendungszweck ankommt, ist ein direkter Vergleich mit anderen Netzwerken nicht wirklich sinnvoll. Um trotzdem einen Überblick zu bekommen, wird das „Open System Interconnection“ (OSI) Modell herangezogen und die einzelnen Schichten davon betrachtet und in Relation zu einem 1-Wire Netzwerk gestellt. Das OSI Modell besteht aus sieben Schichten. Diese wären der „Physical“, „Link“, „Network“, „Transport“, „Session“, „Presentation“ und der „Application“ Layer. Diese Schichten werden nun in einem 1-Wire Netzwerk betrachtet. Abbildung 20 zeigt hierzu die im 1-Wire Netzwerk vorhandenen Schichten.

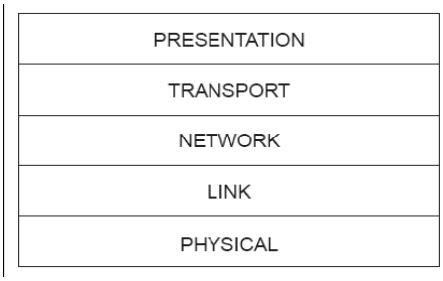


Abbildung 20: Im 1-Wire Netzwerk enthaltene Schichten des OSI Modells

- „Physical Layer“: In dieser Schicht sind die elektrischen Charakteristiken, wie Spannungspegel und Zeitintervalle (Zeitschlitze definiert).
- „Link Layer“: Hier werden die grundlegenden Kommunikationsfunktionen definiert. Dazu zählen zum Beispiel die Funktionsweise des Resets und der Antwortimpulse der Slaves.
- „Network Layer“: Hier sind die Identifikation der einzelnen 1-Wire Geräte und die damit verbundenen Netzwerkmöglichkeiten angesiedelt. Die 1-Wire Geräte werden anhand ihrer weltweit eindeutigen Adresse identifiziert. Dadurch ist die Adressierung in einem Netzwerk geklärt.
- Über den „Network Layer“ ist der Zugriff auf das ROM, den jedes 1-Wire Gerät hat, möglich.
- „Transport Layer“: Hier ist es Möglich auf den Speicher und Funktionen der 1-Wire Geräte zuzugreifen. Da manche Geräte keine zusätzlichen Funktionen oder Speicher haben, gibt es bei ihnen keinen "Transport Layer“. Da die Geräte teilweise eine unterschiedliche Struktur bei ihren speziellen Funktionen haben ist diese Schicht nicht für alle 1-Wire Geräte gleich.
- „Session Layer“: Dieser ist im Normalfall nicht vorhanden. "Presentation Layer“: Diese Schicht bietet die Möglich, auf den Speicher eines 1-Wire Gerätes wie auf eine Diskette zuzugreifen.
- „Application Layer“: stellt das fertige Programm des Anwenders dar, in diesem Fall gehört dies nicht zur 1-Wire Struktur.

4.3. Tcl/Tk

4.3.1. Allgemein

Tcl (ursprünglich „Tool command language“, also etwa „Werkzeugsprache“) ist eine OpenSource-Skriptsprache, die sehr einfach zu erlernen ist und mit der Applikationen innerhalb kurzer Zeit erstellt werden können. Tcl wird üblicherweise wie engl. tickle (kitzeln) ausgesprochen oder natürlich Te Ce El. Sie ist mit Unix-Shell-Sprachen verwandt. Die Syntax sieht C-ähnlich aus, aber auch starke Bezüge zu Lisp sind vorhanden.

Sehr bekannt ist Tcl durch das Toolkit Tk, mit dem sich portable grafische Benutzeroberflächen leicht programmieren lassen. Der grafische Werkzeugkasten "Tk" steht für eine Vielzahl von Betriebssystemen mit dem für die jeweiligen Windowmanager üblichen Aussehen ("native look and feel") zur Verfügung. Diese Programmierschnittstelle wird auch von Perl und Python benutzt. Die Kombination aus Sprache und Toolkit, auch bekannt als Tcl/Tk, ist jedoch besonders einfach zu verwenden; sie stammt von John Ousterhout (seit 1988). Tcl arbeitet standardmäßig mit Unicode, so dass Zeichenketten ohne besonderen Aufwand alle Zeichen bis U+FFFD enthalten können. Über 16 Bit hinausgehende Unicodes werden noch nicht unterstützt.

4.3.2. Syntax

Tcl ist im Grundsatz sehr einfach aufgebaut und grenzt sich gegen Sprachen wie Perl, APL und C durch absolut konsequenten Einsatz einer einheitlichen Syntax ab. Wer mit Kommandozeileninterpretern (Shell, MS-DOS) vertraut ist, kennt auch die Grundstruktur von Tcl-Kommandos. Ein Tcl-Skript besteht aus mehreren Kommandos. Ein Kommando besteht aus einem Kommandowort gefolgt von Argumenten (Parameter). Ein Kommando wird von einem Zeilenende oder Semikolon begrenzt. Gegenüber einfachen Kommandozeileninterpretern verfügt aber Tcl über die Möglichkeit, Kommandos ineinander zu verschachteln. Statt eines Argumentes in einem Kommando kann in eckigen Klammern ein weiteres Kommando angegeben werden. Die Unterkommandos werden zuerst ausgeführt. Ihr Resultat wird dann jeweils als Argument im übergeordneten Kommando eingesetzt. Der Mechanismus ist wie der der Backquotes bei der Unix-Shell.

Auch Konstrukte wie if und while, über Zuweisungen bis hin zu Kommentaren sind Kommandos. Der Kommentar ist ein Kommando, das einfach nichts tut. Die Kommandos folgen der Polnischen Notation, wie Lisp. Das Kommandowort steht am Anfang, dann folgen die Parameter.

```
kommandoWort par1 par2 ... parN
```

Ein umgekehrter Schrägstrich „\“ (Backslash) am Ende einer Zeile markiert, dass ein Kommando auf der nächsten Zeile fortgesetzt wird. Anführungsstriche schließen Zeichenketten ein. Damit keine Verwechslungen auftreten, sind einfache Anführungszeichen (Hochkommata) unbekannt. Auch das einfache Gleichheitszeichen (=) kommt nicht vor. Das doppelte Gleichheitszeichen dient Vergleichen.

Geschweifte Klammern schützen ihren Inhalt von Interpretation. Dadurch können Kommandos dann auch mehrere (auch viele) Zeilen umfassen. Als Beispiel sei das While-Kommando erwähnt. Es erwartet zwei Argumente, das erste enthält einen Befehl, der die Bedingung repräsentiert, das zweite die auszuführenden Kommandos innerhalb der While-Schleife.

```
while { Bedingung }  {
  FolgeVonTclKommandos
}
```

Die geschweiften Klammern schützen den Inhalt vor der Interpretation *vor* dessen Übergabe an den While-Befehl. Innerhalb des While-Befehls werden sie so wie nötig evaluiert.

If-Anweisung:

```
if { Bedingung }  {
  FolgeVonTclKommandos
}
```

oder

```
if { Bedingung }  {
  FolgeVonTclKommandos
} else {
  AlternativeFolgeVonTclKommandos
}
```

oder

```
if { Bedingung }  {
  FolgeVonTclKommandos
} elseif { Bedingung } {
  FolgeVonTclKommandos
} else {
  AlternativeFolgeVonTclKommandos
}
```

Es können beliebig viele elseif-Anweisungen folgen.

Zuweisungen geschehen mit dem Kommando set - ohne ein Gleichheitszeichen. So ist auch hier kein Bruch in der extrem einfachen Syntax vorhanden:

```
set variable value
```

Weiter gibt es ein leistungsfähiges switch-Kommando, ein For- und ein Foreach-Kommando. Vor allem letzteres ist deutlich mächtiger als Iterationen in anderen Sprachen. Beispiel, um aus einer Liste von x/y-Koordinaten und einer weiteren aus z-Koordinaten eine Liste aus x/y/z-Koordinaten zu erstellen:

```
set xyzlist {}
foreach {x y} $xylist z $zlist {lappend xyzlist [list $x $y $z]}
```

Es können also mehr als ein Element aus einer Liste, sowie Elemente aus mehreren Listen in einem Schleifendurchlauf verwendet werden.

Ein Kommando liefert einen Stringwert oder eine Liste als Resultat zurück.

```
glob aPattern
```

erzeugt eine Liste der Dateinamen im Arbeitsverzeichnis, deren Namen mit aPattern übereinstimmen. Pattern können ? (ein beliebiges Zeichen), * (0 oder mehr beliebige Zeichen) oder Klassen von Zeichen in eckigen Klammern enthalten. Dann aber sind letztere mit geschweiften Klammern vor zu früher Auswertung zu schützen:

```
glob {[abc]?-*.tcl}
```

liefert Datei- oder Verzeichnisnamen, die mit a, b oder c beginnen, danach aus einem beliebigen Zeichen und einem Bindestrich, danach einer beliebigen (auch leeren) Zeichenfolge und schließlich ".tcl" bestehen. Für dieses Kommando wird die Unix Bibliotheksfunktion glob() verwendet.

Arithmetische Ausdrücke werden durch das Kommando expr ausgeführt.

```
set res [expr 4 + 2 * 3]
```

```
puts $res
ergibt 10.
```

4.3.3. Datentypen

Tcl ist eine (nach außen hin) typlose Sprache. Jede Variable hat eine Zeichenkette als Wert. Dazu kann eine interne Repräsentation z. B. einer Ganzzahl, Fließkommazahl oder Liste treten. Die Verwendung einer nicht definierten Variable führt zu einem Fehler - im Gegensatz zur Programmierung mit dem Unix-Kommandozeileninterpreter (Shell) oder awk. Konstrukte wie assoziative Arrays (Hashtabelle) und Listen werden in Tcl oft angewendet.

Das folgende Programmschnipsel deklariert implizit einen assoziativen Array und füllt ihn mit Werten.

```
set hauptstadt(Frankreich)  Paris
set hauptstadt(Italien)     Rom
set hauptstadt(Deutschland) Berlin
set hauptstadt(Polen)       Warschau
set hauptstadt(Russland)    Moskau
set hauptstadt(Spanien)     Madrid
```

Alternativ ist folgende Schreibweise möglich, und oft kompakter:

```
array set hauptstadt {
   Frankreich Paris  Italien Rom  Deutschland Berlin .. ..
}
```

Ein bestimmter Wert kann wie folgt auf das Ausgabemedium geschrieben werden

```
puts $hauptstadt(Italien)
```

Eine Liste von allen Ländern zu denen Hauptstädte angegeben worden sind, erzeugt der folgende Tcl-Befehl

```
array names hauptstadt
```

Der Tcl-Interpreter antwortet mit z. B.

```
Polen Spanien Russland Deutschland Italien Frankreich
```

Die Liste ist nicht sortiert. Die Liste wird sortiert ausgegeben mit:

```
lsort [array names hauptstadt]
```

4.3.4. 1-wire Erweiterung für Tcl

OW-Tcl ist eine Tcl-Erweiterung, die eine Schnittstelle zu OWFS zur Verfügung stellt.

4.3.4.1. OW::init

```
OW::init interface <Interface ...> <Optionen>
```

Verbindung zum 1-Wire Adapter oder zum OW-Server. Interface definiert die Verbindung zum 1-Wire-Bus. Mögliche Einstellungen:

usb | usbN | u | uN	Direkte Verbindung zum 1-Wire Interface an der USB-Schnittstelle. -- DS9490. N – Adapternummer bei mehreren Adaptern (optional)
/dev/ttySx	Direkte Verbindung zum 1-Wire Interface an der Seriellen-Schnitt-

	stelle. -- DS9097U oder DS9097.
:port | host:port | socket:/local/socket/path	Ort von einem OW-Server-Dienst welches zum 1-Wire-Bus verbunden ist. OW-Tcl kann genauso wie OW-Fs und OW-Httpd auf den OW-Server-Dienst gleichzeitig zugreifen.

Optionen erlauben zusätzliche Betriebsparameter des 1-Wire-Busses einstellen. Mögliche Optionen:

-format value	Anzeigeformat der Namen der 1-Wire-Geräte. Mögliche Formate:

f.i	XX.YYYYYYYYYYYY (Standard)
fi	XXYYYYYYYYYYYY
f.i.c	XX.YYYYYYYYYYYY.ZZ
f.ic	XX.YYYYYYYYYYYYZZ
fi.c	XXYYYYYYYYYYYY.ZZ
fic	XXYYYYYYYYYYYYZZ

-cache value	Cache Time-Out (in Sekunden)
-celsius | -fahrenheit | -kelvin | -rankine	Einheit der Temperatur. Standard Celsius.
-readonly	Nur der Lesezugriff ist auf 1-Wire-Geräte erlaubt.
-error-print	Ort wo die Informationen/Fehler ausgegeben werden. ● stderr vordergrung / syslog Hintergrund (Standard) ● nur syslog ● nur stderr ● dev/null (unterdrückt alle).
-error-level	Welche Art von Informationen ausgegeben werden. ● Nur Fehler (Standard) ● verbinde und trennen der Verbindung ● Alle High-Level Aufrufe ● Alle Daten von jedem Aufruf

4.3.4.2. OW::finish

```
OW::finish
```

Beendet die Verbindung zum 1-Wire-Bus oder OW-Server.

4.3.4.3. OW::get

```
OW::get <Pfad>
```

Bringt den Inhalt des OWFS Verzeichnisses als die Liste zurück, wenn der abgegebene Pfad Name von OWFS-Datei ist dann gibt er den Wert aus. Wenn der Pfad Name eines OWFS Verzeichnisses ist, gibt er den Inhalt des Verzeichnisses zurück. Wenn man für Dateien * ALL angibt so gibt er eine Werteliste zurück. Wenn der Pfad nicht definiert wird, kommt Inhalt des Wurzel-Verzeichnisses von OWFS zurück.

4.3.4.4. OW::put

```
OW::put <Pfad> <Wert>
```

Setzt Wert in die OWFS Datei ein, die im Pfad angegeben wird. Für den Fall wenn die Datei * ALL nimmt er die Werteliste. Wenn <Wert> nicht definiert wird, wird eine leere Zeichenkette geschrieben.

4.3.4.5. OW::isdirectory oder OW::isdir

```
OW::isdirectory <Pfad>
OW::isdir path
```

Wenn <Pfad> der aktuelle Pfad ist, dann gibt er eine 1 zurück ansonsten eine 0.

4.4. HTML (Hypertext Markup Language)

Die Hypertext Markup Language (HTML) ist ein Dokumentenformat zur Auszeichnung von Hypertext im World Wide Web und wurde 1989 von Tim Berners-Lee am CERN in Genf festgelegt. Sie basiert auf der Metasprache SGML, die zur Definition von Auszeichnungssprachen verwendet wird. HTML ist also eine Auszeichnungssprache zur Beschreibung von Informationen in Hypertexten. HTML wurde vom World Wide Web Consortium (W3C) weiterentwickelt, ist aber mittlerweile zugunsten von XHTML aufgegeben worden.

4.4.1. Überblick

Namensgebend sind die Hypertext-Elemente, die zum Verweis auf andere Textstellen oder auf ein anderes Dokument dienen. Zur Adressierung anderer Dokumente im Internet werden innerhalb des Dokumentes Hyperlinks verwendet. Dies ist die Grundlage für das World Wide Web. Die Programme, die die Struktur des Dokuments interpretieren und als formatierte Webseiten (eventuell mit Interaktionselementen wie Links oder Formularen) darstellen, werden Webbrowser genannt.

Dem Text wird durch Auszeichnung (engl. markup) von Textteilen mit in der Regel paarweisen (öffnenden und schließenden) Tags eine Struktur verliehen. Die jeweils zusammengehörenden Tags bilden zusammen mit dem dazwischenliegenden Text (Inhalt) ein Element. Diese Elemente lassen sich nach Regeln, die in einer Dokumenttypdefinition (DTD) angegeben sind, verschachteln: Ein Textabsatz, der ein betontes Wort enthält.

Neben Elementen mit Start- und End-Tag gibt es auch leere Elemente, wie etwa Zeilenumbrüche oder Bilder: Eine Textzeile,
die hier fortgesetzt wird. <img src="E-Mail-Button.jpg" alt="E-Mail" title="Schreiben Sie mir eine E-Mail!">

Es geht in HTML um sogenannte logische, nicht um physische Textauszeichnung, auch wenn sich HTML in früheren Versionen dafür verwenden ließ. Das heißt, HTML-Tags sind keine Angaben zur Präsentation, die dem Webbrowser mitteilen, wie er den Text zwischen den Tags visuell zu formatieren hat. Vielmehr sind Tags eine strukturierende Auszeichnung, mit der sich Textbereichen eine Bedeutung zuordnen lässt, z. B. <h1>...</h1> für eine Überschrift, ... für einen Textabsatz und ... für betonten Text. Wie diese Bedeutung letztlich dem Benutzer vermittelt wird (im Falle eine Überschrift z. B. durch vergrößerte, fette Schrift), ist zunächst dem Webbrowser überlassen und hängt von der Ausgabe-Umgebung ab. Denn obwohl HTML-Dokumente in der Regel auf Computerbildschirmen dargestellt werden, können sie auch auf anderen Medien ausgeben werden, etwa auf Papier oder mittels Sprachausgabe.

Um auf die Präsentation eines HTML-Dokuments in verschiedenen Medien Einfluss zu nehmen, eignen sich CSS-Formatvorlagen. Daher gelten Elemente und Attribute zur Präsentation wie ..., ... und width als veraltet und sollten nach allgemeiner Auffassung vermieden werden. Die meisten Elemente und Attribute zur Präsentation wurden in der HTML-4.01-Spezifikation als missbilligt (engl. deprecated) markiert.

Wer diesen sogenannten Quelltext auf Webseiten ansehen möchte, kann dafür in den meisten Browsern die Funktion „Quelltext anzeigen“ oder „show source“ wählen. Der Browser öffnet dann ein neues Fenster und zeigt das HTML-Dokument so, wie er es vom Server empfangen hat.

4.4.2. Sprachtyp

HTML ist keine Seitenbeschreibungssprache wie etwa Postscript, da keine Papier-Seiten beschrieben werden, sondern Text strukturell und logisch ausgezeichnet wird. HTML ist eine Auszeichnungssprache und wird als solche auch nicht programmiert, sondern schlicht geschrieben.

Ein ähnliches Konzept (logische Beschreibung) wie hinter HTML steht hinter dem Satzsystem TeX/LaTeX, das im Unterschied zu HTML jedoch auf die Ausgabe per Drucker auf Papier zielt.

4.4.3. Versionen

- HTML (ohne Versionsnummer, 3. November 1992): Die Urversion, die sich nur an Text orientierte. Weblink
- HTML (ohne Versionsnummer, 30. April 1993): Zu Text kommen neben Attributen wie fette oder kursive Darstellung auch die Bildintegration dazu.
- HTML+ (November 1993) Geplante Erweiterungen, die in spätere Versionen einflossen, aber nie als HTML+ verabschiedet wurden. Weblink
- HTML 2.0 (November 1995): Die mit RFC 1866 definierte Version führt u.a. Formulartechnik ein. Der Status dieses Standards ist „HISTORIC“. Auch die Vorgänger sind veraltet.
- HTML 3.0: Die Version erscheint nicht, weil sie mit der Einführung des Netscape-Browsers in der Version 3 bereits vor der Veröffentlichung obsolet ist.
- HTML 3.2 (14. Januar 1997): Neu in dieser Version sind zahlreiche Feature wie Tabellen, Textfluss um Bilder, Einbindung von Applets.
- HTML 4.0 (18. Dezember 1997): Mit dieser Version werden auch Stylesheets, Skripte und Frames eingeführt. Auch eine Trennung in Strict, Frameset und Transitional erfolgt. Am 24. April 1998 erscheint eine leicht korrigierte Version dieses Standards.
- HTML 4.01 (24. Dezember 1999): Ersetzt HTML 4.0 mit vielen, kleineren Korrekturen. Es ist die letzte HTML-Version.
- XHTML 1.0 (26. Januar 2000): Eine Neuformulierung von HTML 4.01 mit Hilfe von XML. Am 1. August 2002 erscheint eine überarbeitete Version.
- XHTML 1.1 (31. Mai 2001): Nachdem XHTML in Module aufgeteilt wurde, wird mit XHTML 1.1 eine strikte Version definiert, bei der die mit HTML 4 eingeführten Varianten Frameset und Transitional entfallen.
- XHTML 2.0 (öffentlicher Entwurf): Diese Version basiert nicht mehr auf HTML 4.01 und führt einige neue Tags ein, so z.B. <nl> für Navigationslisten. Die Trennung von Auszeichnung und Stil soll in dieser Version vollendet werden

4.4.4. HTML-Struktur

4.4.4.1. Allgemeine Struktur

Ein HTML-Dokument besteht aus drei Bereichen:

- der Dokumenttypdeklaration ganz am Anfang der Datei, die die verwendete DTD angibt, z.B. HTML 4.01 Strict,

- dem HTML-Kopf (HEAD), der hauptsächlich technische oder dokumentarische Informationen enthält, die nicht direkt im Browser sichtbar sind und
- dem HTML-Körper (BODY), der anzuzeigende Informationen enthält.

Somit sieht die Grundstruktur einer Webseite wie folgt aus:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
 <html>
     <head>
       <title>Titel der Webseite</title>
       <!-- Evtl. weitere Kopfinformationen -->
     </head>
     <body>
        Inhalt der Webseite
     </body>
 </html>
```

4.4.4.2. HTML-Kopf

Im Kopf können sieben verschiedene Elemente angewandt werden:

- title bezeichnet den Titel der Seite, der im Browserfenster sowie oft in Suchmaschinen angezeigt wird.
- meta kann vielfältige Metadaten enthalten. Siehe Meta-Tags.
- base gibt entweder eine Basis-URI an oder einen Basisframe.
- link dient zur Angabe von logischen Beziehungen zu anderen Ressourcen. Am häufigsten zur Einbindung von Stylesheets benutzt.
- script bindet Code in einer bestimmten Skriptsprache ein, hauptsächlich JavaScript.
- style enthält Stylesheet-Code, hauptsächlich CSS-Regeln.
- object bindet eine externe Datei ein. Browser dürfen solche Objekte im Dokumentkopf nicht darstellen.

4.4.4.3. HTML-Körper

Im Wesentlichen unterscheidet HTML zwischen Block- und Inline-Elementen. Der wesentliche Unterschied ist, dass erstere in der Ausgabe einen eigenen Block erzeugen, in dem der Inhalt untergebracht wird, während die Inline-Elemente den Textfluss nicht unterbrechen. Zu den Block-Elementen gehören z.B. die Überschriften und die Tabellen.

Eine Hauptüberschrift wird so ausgezeichnet:

```
 <h1>Hauptüberschrift</h1>
```

h1 steht für Heading 1.

Der Inhalt zwischen den beiden Tags wird nun als Hauptüberschrift interpretiert. Weiter möglich sind h2 bis h6, Überschriften zweiter bis sechster Ordnung, mit denen sich die Gliederung einer Seite verdeutlichen lässt. Die Präsentation dieser Überschriften ist von ihrer strukturierenden Bedeutung unabhängig und kann mit CSS beeinflusst werden. Die Zweckentfremdung der Überschrift-Elemente zur Vergrößerung von Text wird allgemein als Missbrauch angesehen.

Hyperlinks: Hyperlinks sind Verweise auf andere Dateien, meistens ebenfalls HTML-Dateien, die üblicherweise im Browser durch Klick verfolgt werden können.

```
<a href="http://www.example.com/">Gehe zu example.com</a>
```

Hier wird auf die Ressource http://www.example.com/ verwiesen. Der Text Gehe zu example.com wird dabei als Link dargestellt.

Zur Logik stehen zum Beispiel die Elemente strong und em bereit, mit denen sich stark hervorgehobener oder betonter Text auszeichnen lässt. Gängigerweise werden strong-Elemente durch Fettschrift und em-Elemente durch kursive Schrift visuell kenntlich gemacht.

Die logische Beschreibung der Struktur des Textes, wie sie die obigen Beispielen veranschaulichen, vereinfacht es zum Beispiel, dass der Text auch einem Sehbehinderten vorgelesen oder als Braille ausgegeben werden kann, und ermöglicht Suchmaschinen die Auswertung der Texte unter Berücksichtigung ihrer Bedeutung.

4.4.5. HTML-Varianten

Beim Entwurf der letzten HTML-Version 4 sollte der Tatsache, dass in vielen HTML-Dokumenten noch Elemente und Attribute zur Präsentation eingesetzt werden, Rechnung getragen, aber auch eine stilistisch saubere Dokumenttypdefinition angeboten werden. Das Ergebnis waren schließlich drei Varianten:

4.4.5.1. Strict

Diese DTD umfasst den Kernbestand an Elementen und Attributen. Es fehlen die meisten Elemente und Attribute zur Beeinflussung der Präsentation, unter anderem die Elemente font, center und u sowie die Attribute bgcolor, width, height und target. Deren Rolle sollen in Strict-Dokumenten Stylesheets übernehmen. Text und nicht-blockbildende Elemente innerhalb der Elemente body, form, blockquote und noscript müssen sich grundsätzlich innerhalb eines Container-Elements befinden, zum Beispiel in einem p-Element.

4.4.5.2. Transitional

Die Transitional-Variante enthält noch ältere Elemente und Attribute, die auch physische Textauszeichnung ermöglichen. Durch diese DTD soll auch Webautoren, die noch nicht logische Strukturierung und Präsentation voneinander trennen, die Möglichkeit gegeben werden, standardkonformes HTML zu schreiben. Gleichzeitig soll sie sicherstellen, dass bestehende Webseiten weiterhin durch aktuelle Webbrowser angezeigt werden können.

4.4.5.3. Frameset

Diese Variante enthält zusätzlich zu allen Elementen der Transitional-Variante noch die Elemente für die Erzeugung von Framesets.

4.4.5.4. Zukunft der Varianten

Mit XHTML 1.1, der neuesten Version von HTML, verzichtet das W3-Consortium wieder auf die Unterscheidung und führt lediglich die Strict-Variante als „reine Lehre“ weiter. Dies setzt sich auch im momentan (2005) im Entwicklungsstadium begriffenen XHTML 2.0 fort.

Elemente, die bislang nur in den Varianten Transitional und Frameset vorkamen, werden beinahe alle entfallen. Entsprechende Effekte müssen dann durch CSS, Javascript, XFrames oder andere Methoden erreicht werden, da sie zur logischen Textauszeichnung nicht notwendig sind.

Ein Attribut, das aus der Transitional-Variante voraussichtlich den „Sprung“ in XHTML 2.0 schafft, ist das „value“-Attribut bei nummerierten Listenpunkten, das erlaubt, die Nummer des Listenpunktes selbst festzulegen.

4.4.6. Zusatztechniken und Weiterentwicklungen

4.4.6.1. Stylesheets

Im Laufe der Jahre ist HTML um Elemente erweitert worden, die sich mit der Gestaltung des Dokuments befassen, was der ursprünglichen Idee der Systemunabhängigkeit entgegen lief. Eine Rückbesinnung auf die Trennung von Inhalt (Struktur) und Layout wurde durch die Definition von CSS vorgenommen. So soll das Aussehen des Dokuments in einer separaten Datei, dem sogenannten Stylesheet, festgelegt werden. Dies verbessert die Anpassungsfähigkeit des Layouts an das jeweilige Ausgabegerät und an spezielle Bedürfnisse der Benutzer, z.B. von Sehbehinderten. Heutzutage ist die CSS-Unterstützung der Browser ausreichend, um damit eine anspruchsvolle Gestaltung zu realisieren.

4.4.6.2. Dynamisches HTML

Schon sehr früh in der Geschichte von HTML wurden Zusatztechniken erfunden, die es ermöglichen, HTML-Dokumente während der Anzeige im Browser dynamisch zu verändern. Die gebräuchlichste ist JavaScript. Man spricht bei solchen interaktiven Dokumenten von dynamischem HTML. Diese Techniken wurden von verschiedenen Browser-Herstellern, allen voran Microsoft und Netscape, unabhängig voneinander entwickelt. Daher gab es erhebliche Probleme bei der Umsetzung der Techniken zwischen den verschiedenen Browsern. Mittlerweile interpretieren alle verbreiteten JavaScript-fähigen Browser das Document Object Model (DOM). Dadurch ist es möglich, in allen Browsern lauffähige Skripte zu schreiben. Es gibt jedoch noch immer Differenzen bei der Unterstützung des DOM-Standards.

4.4.6.3. XML

Mittlerweile wurde die letzte Version des HTML-Standards (HTML 4.01) in der Metasprache XML neu formuliert. Das daraus entstandene XHTML 1.0 genügt den im Vergleich zu SGML strengeren syntaktischen Regeln von XML, ist aber in seinen drei DTD-Varianten semantisch mit der jeweils entsprechenden DTD-Variante von HTML 4.01 identisch. Aktuell liegt der XHTML-Standard in der Version 1.1 vor, der eine zusätzliche Modularisierung der Elemente einführt.

Als zukünftiges universelles Format wird zunehmend XML selbst eingesetzt. Dieses arbeitet ebenfalls mit Tags, die jedoch per Dokumenttypdefinition sehr restriktiv eingesetzt werden können. XML ermöglicht somit ein selbstbeschreibendes Datenmodell. Zur Umwandlung von in XML gespeicherten Daten in die entsprechenden Anzeigeformate (z.B. XHTML) kann die Stylesheet-Sprache XSL Transformation verwendet werden.

4.4.7. Falsche Interpretation von Webdokumenten

Die verbreiteten Browser folgen bei der Auswertung von HTML nicht ganz dem Standard. Dies zwingt Webautoren dazu, ihre Dokumente in verschiedenen Browsern zu testen und gegebenenfalls anzupassen. Eine solche Anpassung an die Gegebenheiten auf der Leserseite ist zwar durchaus von Vorteil, aber häufig kommt es zu schwer umgehbaren Fehlinterpretationen. Durch die wohlüberlegte Verwendung von strukturierendem HTML kann immerhin gewährleistet werden, dass ein Webdokument in jedem Fall zumindest grundlegend zugänglich und lesbar ist.

Der am weitesten verbreitete Webbrowser, der Internet Explorer, ist gutmütig und lässt so manchen Fehler durchgehen, hat jedoch insbesondere in CSS-Belangen einige Defizite, welchen Rechnung zu tragen ist. Die Browser mit Gecko-Engine (darunter Mozilla und Mozilla Firefox), Opera, Safari und Konqueror haben hier am wenigsten Probleme, tolerieren Codefehler jedoch weniger.

4.4.8. HTML lernen

SELFHTML ist ein deutschsprachiges und sehr umfangreiches Projekt, das für viele Themen rund um HTML Referenzmaterial bietet.

Die Lektüre einer modernen linearen Einführung (siehe Weblinks) und die Handarbeit direkt in einem Texteditor ist empfehlenswert, um HTML richtig zu verstehen und voll auszunutzen.

Für schnelle Erstellen von Webseiten ohne tiefere HTML-Kenntnis mag der Gebrauch eines so genannten WYSIWYG-Editors zunächst genügen. Diese Editoren produzieren einen HTML-Code, der die optischen Vorstellungen weitestgehend widerspiegelt. Allerdings wird die strukturelle und logische Auszeichnung, die dem Text erst einen echten Mehrwert gibt, dabei vernachlässigt. Diese erfordert ein gutes Verständnis von HTML, das ein WYSIWYG-Editor nicht ersetzen kann. Hinzu kommt, dass diese Editoren gelegentlich ungültiges HTML produzieren, was die Darstellung des Dokuments von der Fehlertoleranz des Webbrowsers abhängig macht. In der professionellen Webgestaltung werden solche HTML-Editoren daher häufig nur als zusätzliches Hilfsmittel benutzt, um etwa das Grundgerüst von Webseiten zu erstellen. Die Feinarbeit erfolgt dann in einem Texteditor direkt am HTML-Code.

4.5. CGI (Common Gateway Interface)

Die CGI-Schnittstelle (Common Gateway Interface – in etwa Allgemeine Vermittlungsrechner-Schnittstelle) ist ein Standard im Web für den Datenaustausch zwischen auf Webservern bereitstehenden Programmen (Scripts) und den sie aufrufenden Webbrowsern. Hierbei können die serverseitigen Programme, die z.B. von HTML-Dateien aus aufgerufen werden können, sowohl Daten vom Browser empfangen (etwa Formulareinträge) als auch neu generierte Daten an den Browser verschicken (etwa eine HTML-Seite). CGI ist also eine schon länger bestehende Variante, Webseiten dynamisch bzw. interaktiv zu machen.

Um die CGI-Schnittstelle zu verwenden, muss diese von der Webserver-Software unterstützt werden. Dabei ist wichtig, dass diese Software dem Programm/Script immer 3 Dinge zur Verfügung stellt:

Umgebungsvariablen (z.B. SERVER_NAME), deren Inhalte dem Programm helfen, sich „vor Ort“ zu orientieren und über aktuelle Einstellungen zu informieren.

Weiterleitung von Ausgaben, meistens als dynamisch erzeugte HTML-Seite (oder Seitenteile), aber auch als Einträge in Fehlerprotokolldateien.

Einholen von Formulareingaben oder Aufrufparametern z.B. aus HTML-Seiten, damit das CGI-Programm/-Script auf diese reagieren kann. Dabei können solche Daten als Umgebungsvariable (GET-Methode) oder über einen Eingabe-Kanal (POST-Methode) Eingang ins Programm/Script finden, wobei letztere Möglichkeit sicherer ist.

Wie diese Daten strukturiert sind, ist die eigentliche Schnittstellenbeschreibung („interface“).

CGI-Programme können also in allen möglichen Programmiersprachen geschrieben sein. Es muss lediglich auf dem Server ein entsprechender Laufzeitinterpreter vorhanden sein, oder das Programm muss für das Serverbetriebssystem kompiliert worden sein. Am weitesten verbreitet ist hierbei Perl.

Ein Nachteil, der der CGI-Ausführung nachgesagt wird, ist, dass sie langsamer sei als andere Möglichkeiten (z.B. Servlets), da für jeden CGI-Aufruf eine neue Programm-Instanz ausgeführt werden muss. Deshalb wird CGI heutzutage nicht mehr so oft eingesetzt, denn selbst Ansätze wie FastCGI, welches gewisse Nachteile von CGI aufhebt, konnten sich zumindest nicht auf breiter Front durchsetzen. Andererseits wird dieser Nachteil von modernen Webserver-Typen für einige Programmiersprachen behoben (z.B. bietet der Webserver Apache mit dem Modul mod_perl die Möglichkeit, einen Perl-Interpreter in den Webserver selbst einzubetten, was – neben anderen Vorteilen – die Ausführungsgeschwindigkeit massiv erhöhen kann). Alle derartigen Lösungen sind jedoch nicht mehr programmiersprachenunabhängig.

Die Tatsache, dass über CGI auf dem Webserver eines kommerziellen Providers Programme ausgeführt werden können, die ein Kunde des Providers selbst erstellt hat, ist in höchstem Maße sicherheitsrelevant. Daher muss sichergestellt sein, dass ein über CGI gestartetes Programm nur bestimmte eingeschränkte Typen von Programmroutinen ausführen darf (z.B. kein Löschen von Dateien des Webservers u.ä.). Bei dem Apache-Webserver wird die Ausführung von CGI-Programmen mit Hilfe des Modules suexec gegen solche Cracker-Angriffe gesichert, die das Eindringen als Root-User zum Ziel haben. Die Sicherheitsmaßnahmen sind dabei mehrstufig aufgebaut und so streng, dass viele Server-Administratoren dazu übergegangen sind, auch andere serverseitige Sprachen über CGI laufen zu lassen. So wird zum Beispiel PHP bei den meisten Providern als CGI-Modul eingebunden.

4.6. Solaranlage

4.6.1. Die Geschichte der Photovoltaik

Das Hauptelement der Photovoltaik ist die Solarzelle. Die Solarzelle wandelt die von der Sonne empfangene elektromagnetische Strahlungsenergie auf direktem Weg in elektrische Energie um. Die Funktionsweise beruht dabei auf dem photovoltaischen Effekt, der auch als innerer lichtelektrischer Effekt bezeichnet wird. Im Jahre 1839 beobachtete der französische Physiker A. E. Becquerel, daß zwischen zwei Platinelektroden, die in einer elektrolytischen Lösung eingetaucht sind, eine elektrische Spannung auftritt, wenn eine der Elektroden beleuchtet und die andere dunkel gehalten wird. Diese Zellenanordnung liefert einen Strom, der mit der Beleuchtungsstärke ansteigt. Der Wirkungsgrad solcher Zellen erreichte kaum 1%, so daß an eine weitergehende Verwendung zur Erzeugung von elektrischer Energie aus Sonnenstrahlung nicht zu denken war.

Die Situation änderte sich 1954, als es amerikanischen Forschern gelang, photovoltaische Zellen aus hochreinem Silizium herzustellen. In die Kristallgitterstruktur des Siliziums wurden gezielt Fremdatome eingebracht, wodurch die Zellen den Aufbau von großflächigen Dioden erhielten. Der Wirkungsgrad dieser Zellen betrug 4% und konnte in den folgenden drei Jahren auf 8% gesteigert werden. Wegen der hohen Kosten bei der Herstellung der Siliziumsolarzellen blieb der Einsatz auf Sonderfälle, z.B. Raumfahrttechnik beschränkt. Erst Mitte der 70er Jahr, nach der Ölpreiskrise, wurde mit Hilfe staatlich finanzierter Förderung der Einsatz von Solarzellen vorangetrieben.

Früher ging man davon aus, daß der Wirkungsgrad von Solarzellen, je nach Material, theoretisch an die 30% betragen könnte. Nach einer Neuberechnung des Max-Planck-Institutes für Festkörperforschung in Stuttgart liegt er sogar bei 43%. In der Praxis werden gegenwärtig etwa 13 bis 15% erzielt. Wenn die Photovoltaik bisher dennoch nur einen verschwindend geringen Anteil an der Deckungsrate des gesamten Strombedarfes hat, so liegt dies weniger am Wirkungsgrad als an den bislang noch hohen Herstellungskosten in Verbindung mit der geringen „Leistungsdichte“ der Sonnenstrahlung. Diese beträgt höchstens 1 Kilowatt pro Quadratmeter. Dies bedeutet in sonnenreichen Gegenden eine nutzbare Leistung von jährlich etwa 2200 kWh je Quadratmeter, in unseren Breiten jedoch nur etwa 1000 kWh je Quadratmeter. Es sind daher sehr große Flächen erforderlich, um mit Hilfe der Photovoltaik größere Leistungen erzielen zu können.

4.6.2. Aufbau und Funktionsweise von Solarzellen

Für die Leistungsanwendung werden in der Regel Solarzellen aus kristallinem Silizium verwendet. Ein Silizium-Atom hat in der äußeren Elektronenschale vier Bindungselektronen. In der Kristallstruktur des Siliziums ist jedes Atom über Elektronenpaarbindungen mit vier Nachbaratomen verbunden.

In der Siliziumsolarzelle ist eine dünne obere Schicht mit fünfwertigem Phosphor dotiert, so daß überzählige Elektronen vorhanden sind (n-Schicht).

Die Basisschicht der Siliziumzelle ist dagegen mit dreiwertigen Bor dotiert. Die so entstehenden Fehlstellen bei der Elektronenpaarbindung verhalten sich wie freibewegliche positive Ladungsträger (p-Schicht). An der Grenzfläche beider Schichten besetzen Leitungselektronen aus der phosphordotierten Schicht die Plätze der Fehlstellen in der Basisschicht. An den zurückbleibenden Gitterplätzen entstehen ortsfeste Raumladungen; ein elektrisches Feld baut sich auf, das weiteres Wandern der Elektronen von der n- in die p-Schicht verhindert.

4.6.3. Solarmodul

Datenblatt Solarmodul MQ 36/53 D

Mechanische Daten:

- Basismaterial: Monokristallines Silizium
- Zellengröße: 10 x 10 cm²
- Kapselung: Glas/EVA-Glas
- Rahmen: Edelstahl
- Abmessung: 977 mm Länge, 462 mm Breite, 68 mm Höhe
- Gewicht: ca.: 5500 g

Elektrische Daten:

Temperatur:	**25°C**	**60°C**
Leerlaufspannung:	21,6 V	19,1 V
Kurzschlußstrom:	3,25 A	3,30 A
Spannung im MPP:	17,2 V	14,2 V
Strom im MPP:	3,08 A	3,13 A
Nennleistung:	53 W	44,5 W

Die Werte gelten für eine Einstrahlung von AM 1,5 (1000 W pro m^2) und eine Solarzellentemperatur von 25°C.

Temperaturabhängigkeit

- Spannungsänderung: 0,33% pro °C
- Stromänderung: 0,05% pro °C
- Leistungsänderung: 0,46% pro °C

Zulässige Betriebsbedingungen

- Temperatur: -50 bis +90°C
- Luftfeuchte: bis 100% rel. Feuchte bei +90°C
- Winddruck: bis 160 km/h stabil
- Korrosion: korrosionsfest bei Seewasser und Seeklima

4.6.4. Solaranlage auf dem Dach des E-Gebäudes

Die Solaranlage von der wir die Daten abfragen besteht aus zwei Panels. Die gewonnene Energie wird in Akkus gespeichert, denn die Anlage besitzt keinen Wechselrichter um ins Netz zu Speisen. Die Akkus werden mit Hilfe einer Laderegelung geladen. Als Verbraucher ist eine Kompakt-Leuchtstofflampe angeschlossen, die zu- und abgeschaltet werden kann. Leider haben wir in der Dokumentation der Anlage nicht entdeckt wie oder wann dies geschehen soll, es ist auch möglich, dass die Kompakt-Leuchtstofflampe bereits durchgebrannt ist, was auf Grund des Alters der Anlage sein kann.

Abbildung 21: Solaranlage auf dem Dach des E-Gebäudes

In Gerät 1 sind die Messeinrichtungen, Ladeüberwachung und Laststeuerung eingebaut.

In Gerät 2 sind die Messwandler und Laderegelung eingebaut.

Abbildung 22: oben: Messwandler und Laderegelung -- unten: Messeinrichtungen, Ladeüberwachung und Laststeuerung

Abbildung 23: Anzeigen des Steuergeräts

Aufgrund des schlechten Zustandes einiger Baugruppen waren wir überrascht, das die Anlage noch relativ gut funktioniert. Der Schaltschrank in dem der Überspannungsschutz montiert ist war offen und es stand auch Wasser drin (siehe 2 Photos unten).

Abbildung 24: Zustand des Überspannungsschutzes I

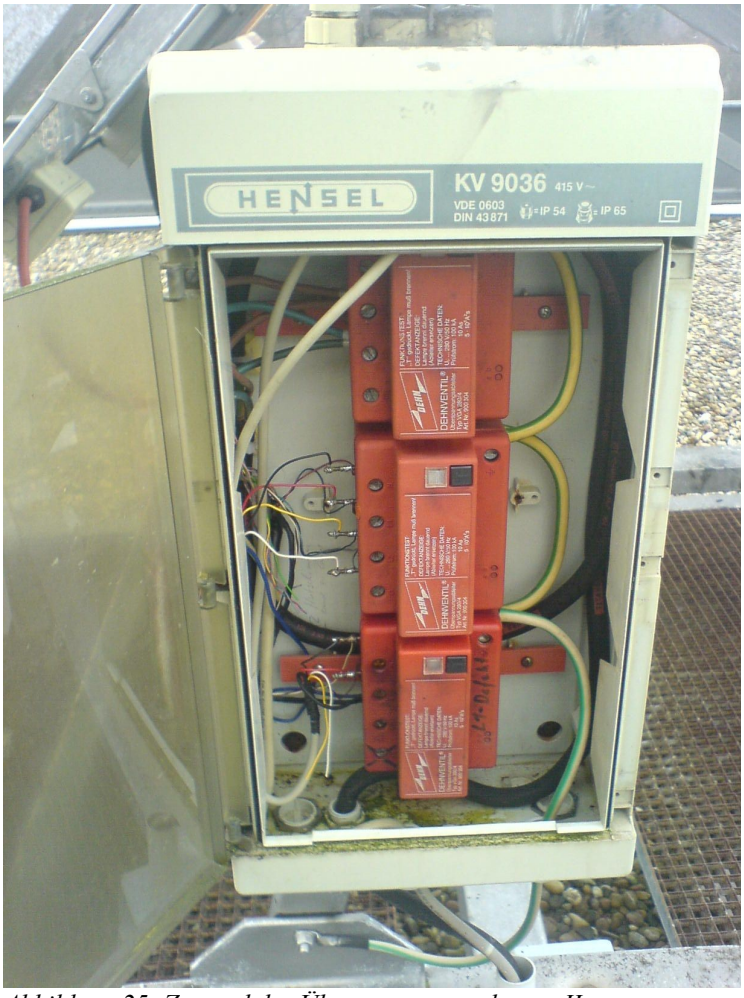


Abbildung 25: Zustand des Überspannungsschutzes II

4.6.5. Signalausgänge

Das Gerät 2 Stellt uns Messwerte von der Solaranlage zur Verfügung. Diese Messwerte werden mit Hilfe von Messwandlern auf einen Spannungsbereich 0 V > U > 10 V genormt. Die mit einem Apostroph gekennzeichneten Größen sind normierte Größen mit der Einheit V.

Messgröße	***Plausible Werte***	***Umrechnungsformeln***
Solarspannung U_S	$0\ V \leq U_S \leq 44\ V$	$U_S = 5 * U_S'$
Solarstrom I_S	$0\ A \leq I_S \leq 3{,}4\ A$	$I_S = \frac{A}{2{,}7\ V} * I_S'$
Verbraucherspannung U_V	$0\ V \leq U_V \leq 29\ V$	$U_V = 3{,}3 * U_V'$
Verbraucherstrom I_V	$0\ A \leq I_V \leq 2{,}5\ A$	$I_V = \frac{A}{3{,}9\ V} * I_V'$
Akkuspannung U_A	$20\ V \leq U_A \leq 29\ V$	$U_A = 3{,}3 * U_A'$
Akkustrom I_A	$-3{,}4\ A\ V \leq I_A \leq 2{,}5\ A$	$I_A = -3{,}25\ A + \frac{A}{V} * I_A'$
Bestrahlungsstärke E	$0\ W/m^2 \leq E \leq 1200\ W/m^2$	$E = 680 \frac{lx}{V} E'$
Außentemperatur T_S	$-20\ °C \leq U_S \leq 40°C$	$T = -50\ °C + \left(\frac{40\ °C - (-50\ °C)}{4V} \right) * T_S'$

5. Modifikation der Router-Firmware

5.1. Opennet Firmware Asus Erstinstallation

5.1.1. Installation unter Windows

Für die Installation benötigt man folgende Dateien:

Firmware Restoration Tool von Asus: http://dlsvr02.asus.com/pub/ASUS/Wireless/WL-500g-03/Ger_1380.zip

Firmware-Datei: http://absorb.it/whiterussian/opennet-openwrt-brcm-2.4-squashfs_0.9.1b.trx

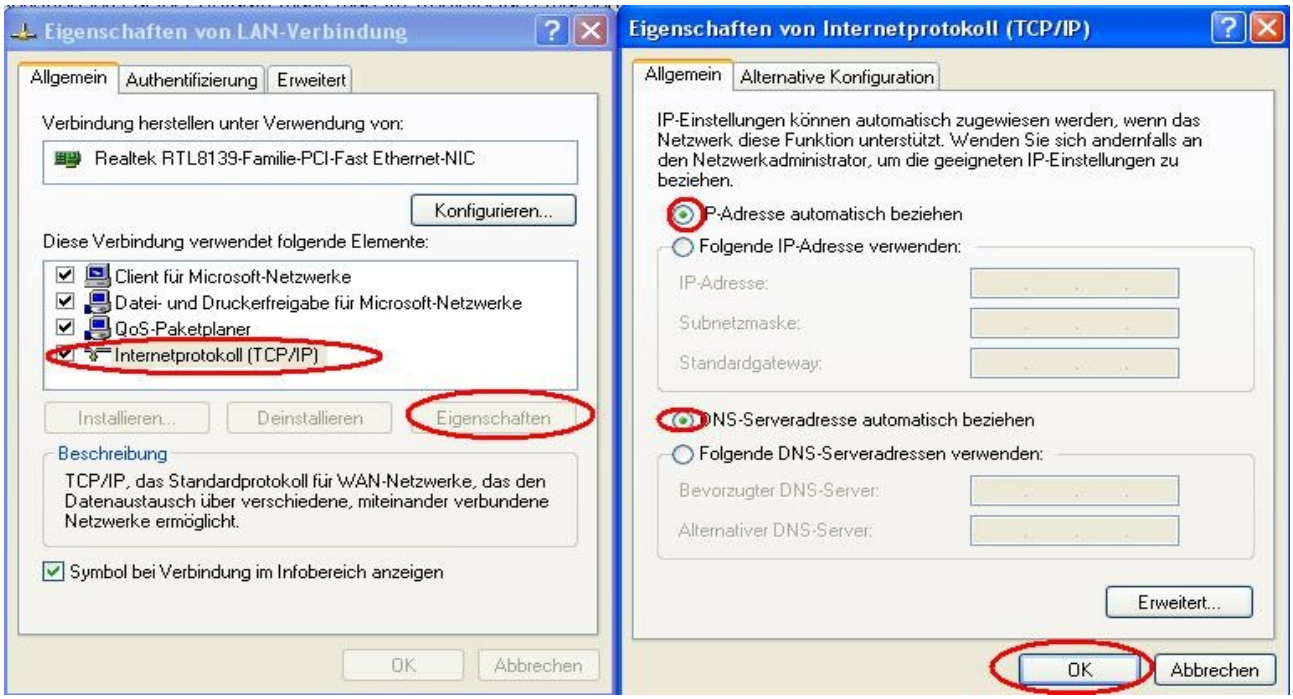


Abbildung 26: IP-Adresse automatisch beziehen

Zuerst konfiguriert man wie im nebenstehenden Bild gezeigt - die Netzwerkumgebung so, dass sie die TCP/IP-Konfiguration automatisch bezieht. Der Router muss an einen Rechner angeschlossen sein und man sollte das WebInterface von ASUS sehen, wenn im Browser die Adresse http://192.168.1.1 aufgerufen wird.

Die Firmware muss mit Hilfe des 'Firmware Restoration Tools' auf den Router geladen werden. Entpacke der Datei und installiere die 'ASUS Wireless Utilities' indem man Setup.exe startest. Danach findet man im Menu das Programm 'Firmware Restauration' – dieses muss gestartet werden.

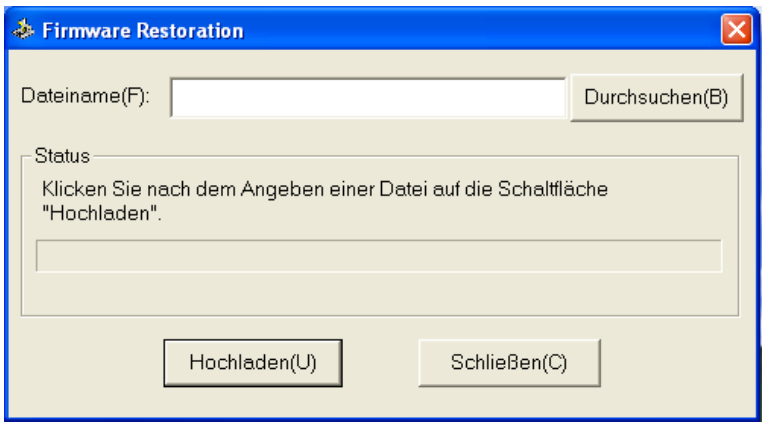
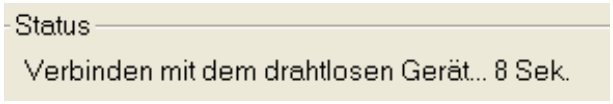
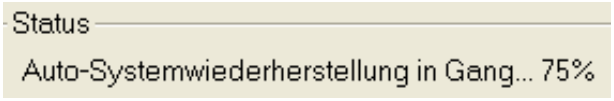


Abbildung 27: Firmware Restauration Tool

Klicken Sie auf 'Durchsuchen' und wählen Sie die Firmware-Datei, die heruntergeladen wurde, aus. Klicke anschliessend auf 'Hochladen'. Das Programm versucht sich nun, mit dem 'drahtlosen Gerät' zu verbinden.

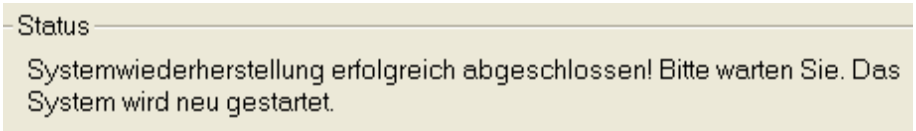


Ziehe den Netzstecker des ASUS heraus und stecke diesen, während man den kleinen 'Restore'-Taster direkt neben dem Stecker mit einem Stift oder ähnlich gedrückt hälst, wieder hinein. Das Programm sollte nun den ASUS-Router finden, die Firmware-Datei heraufladen und mit der 'Systemwiederherstellung' beginnen.



Nach einigen Sekunden sollte die Übertragung beendet sein, das Restauration-Tool zeigt dies an. Nun wird die neue Firmware installiert und konfiguriert, das kann durchaus eine ganze Weile (etwa 5 Minuten) dauern.

Der Router starten nun neu und ist - nach der Aktualisierung der Netzwerkeinstellungen des Computers, bspw. durch einen Neustart - über das Opennet-Firmware-Webfrontend unter http://172.16.0.1 erreichbar.



5.1.2. Installation unter Linux

Für die Installation benötigt man folgende Firmware-Datei:

Firmware-Datei: Download: http://absorb.it/whiterussian/opennet-openwrt-brcm-2.4-squashfs_0.9.1b.trx

Zusätzlich wird das Programm tftp benötigt.

Achtung: Die Erklärung an diese Stelle ist nicht auf einem ASUS deluxe getestet und läuft laut der Dokumentation zur OpenWRT-Installation etwas anders ab.

Man zieht beim Accesspoint den Netzstecker, wartet einige Sekunden und hält den Resetknopf des Accesspoints gedrückt während man den Netzstecker wieder einsteckt. Die Power-LED des Routers sollte nun - etwa im Sekundentakt - blinken. Die IP des AccessPoints ist nun unverändert. Solange die Power-LED des Routers blinkt, kann man die neue Firmware mit der folgenden Befehlssequenz übertragen.

```
user@localhost ~ $ tftp 192.168.1.1
tftp> binary
tftp> trace
tftp> get ASUSSPACELINK\x01\x01\xa8\xc0 /dev/null
tftp> put opennet-openwrt-brcm-2.4-squashfs_0.9.1.trx
ASUSSPACELINK
```

(Die Adresse 192.168.1.1 muss gegebenenfalls durch die vorherige lokale IP des APs ersetzt werden)

Nach dem Upload hört die Power LED des Accesspoint auf zu blinken und die Air LED fängt nach einer Weile an zu flackern. Der Router ist nun neu gestartet und über das Opennet-Firmware-Webfrontend unter http://172.16.0.1 erreichbar.

5.2. Konfiguration des Routers

5.2.1. Mit SSH Secure Shell am Router Anmelden

Da an der Hochschule Karlsruhe an jedem Rechner SSH Secure Shell installiert ist, verwenden wir ihn auch. Bei der Freeware PuTTY für SSH dürfte es keine großen Abweichungen geben.

1. SSH Secure Shell starten
2. Auf  Connect (Verbinden) klicken.
3. Bei Hostname die IP-Adresse eingeben: 192.168.2.200

   User Name: root

   Port Number: 22

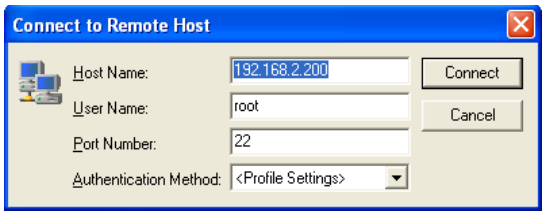


Abbildung 28: Anmeldemaske zu 3.

4. Passwort: Solar#01

Abbildung 29: Passwortabfrage zu 4.

5. Jetzt sind Sie mit dem Router verbunden. Auf der Kommandozeile werden Befehle eingegeben und mit der Eingabe-Taste bestätigt. Die wichtigsten Befehle sind in Kapitel 4.1.6 beschrieben.

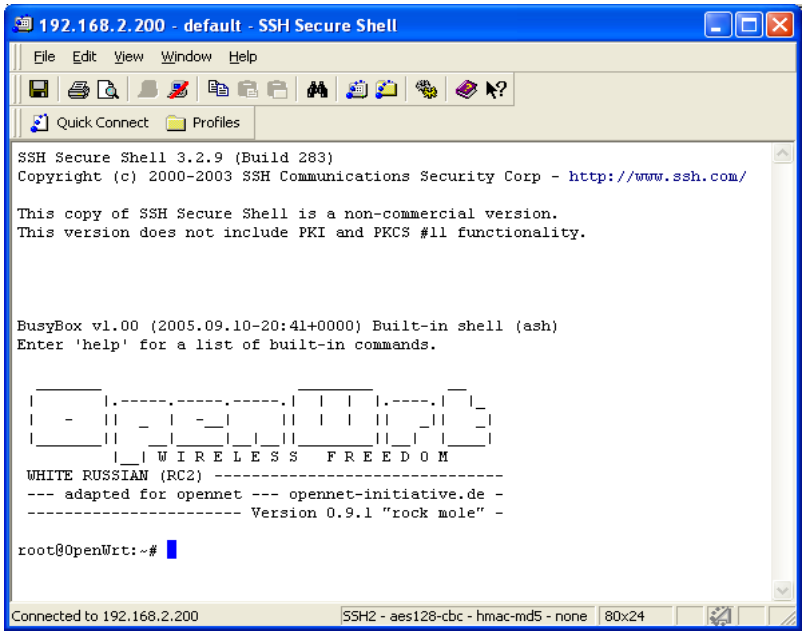


Abbildung 30: Shell auf dem Router (Eingabeaufforderung) zu 5.

6. Um nach der Sitzung die Verbindung zu trennen auf  Disconnect (Verbindung trennen) klicken

5.2.2. Löschen der vorherigen Einstellungen

Im NVRAM werden die Einstellungen des Routers gespeichert, von daher ist es nicht verkehrt ihn vor dem Beginn der Konfiguration zu löschen, damit alte Inhalte keine Seiteneffekte hervorrufen können.

```
mtd erase nvram
reboot
```

5.2.3. Eigenes Passwort setzen

Wer bereits ein eigenes Passwort gesetzt hat, kann diesen Schritt überspringen.
Ruft das Webinterface (http://172.16.0.1/) auf, verwendet den Benutzernamen root und das Standardpasswort admin. Setzt nun ein neues Passwort im Bereich Verwalten > Kennwort (http://172.16.0.1/cgi-bin/password.html).
Hinweis: Das OLSR-Funknetz ist normalerweise unverschlüsselt. Beim Abruf von Verwaltungsseiten wird das Kennwort bei jedem Seitenabruf unverschlüsselt übertragen. Zur Sicherheit sollten daher die Verwaltungsseiten nur über drahtgebundenes Netzwerk bedient werden.
Passwort: Solar#01

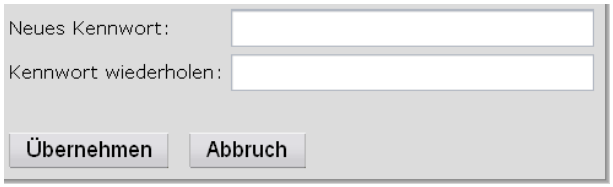


Abbildung 31: Verwaltung: Kennwort

5.2.4. LAN Konfigurieren

IP-Adresse: 192.168.2.200
Sub-Net-Mask: 255.255.255.0
ACHTUNG: Wird bei 'LAN_Protokoll' 'DHCP' ausgewählt, sucht der AccessPoint im LAN nach einem anderen DHCP-Server und versucht eine Adresse zu bekommen. Dies ist oft nicht gewollt. Um auf dem AccessPoint eine DHCP-Server zu betreiben, muss diesem eine statische IP (z.B. 172.16.0.1) zugewiesen werden und im unteren Teil die maximale Zahl der DHCP-Benutzer größer 0 gesetzt werden.

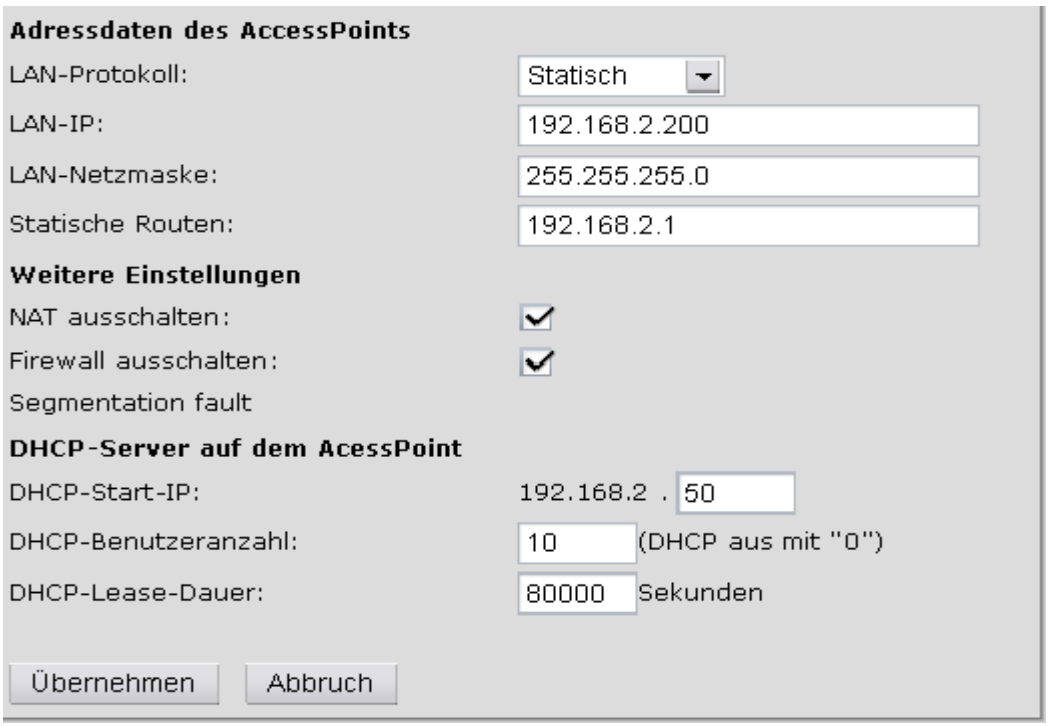


Abbildung 32: Verwaltung : LAN

5.2.5. WAN Ausschalten (DSL-Funktion)

Da das Projekt keinen DSL-Zugang benötigt, haben wir die Funktion abgeschaltet um Fehlerquellen zu vermeiden und keine Systemresoucen zu verschwenden.
Tipp: Für einen bequemen Netzzugriff sollte der Rechnername (einfacher Name ohne Punkte) und die interne Domain (mehrfacher Name mit Punkten getrennt) angegeben sein. Beispiel: Setze Rechnername auf "meinwrt" und Domain auf "meinnetz.freifunk.net" um die Seiten des Gerätes mit http://meinwrt.meinnetz.freifunk.net/ oder sogar nur mit http://meinwrt/ aufzurufen.

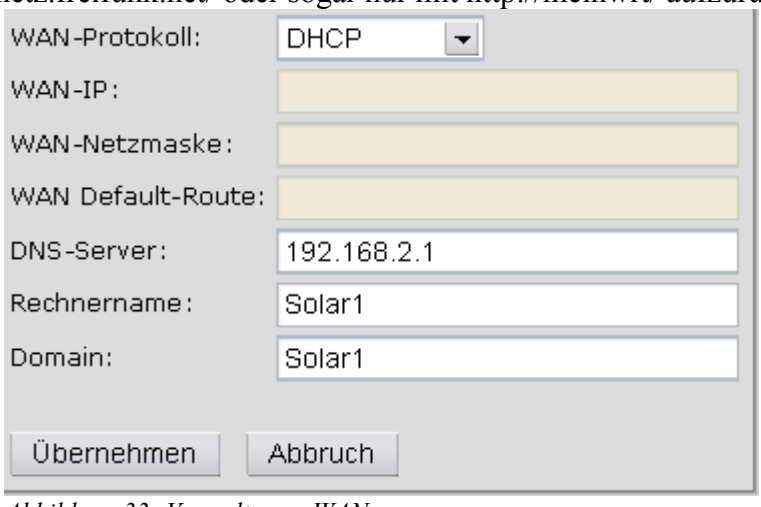


Abbildung 33: Verwaltung : WAN

5.2.6. WLAN Konfigurieren

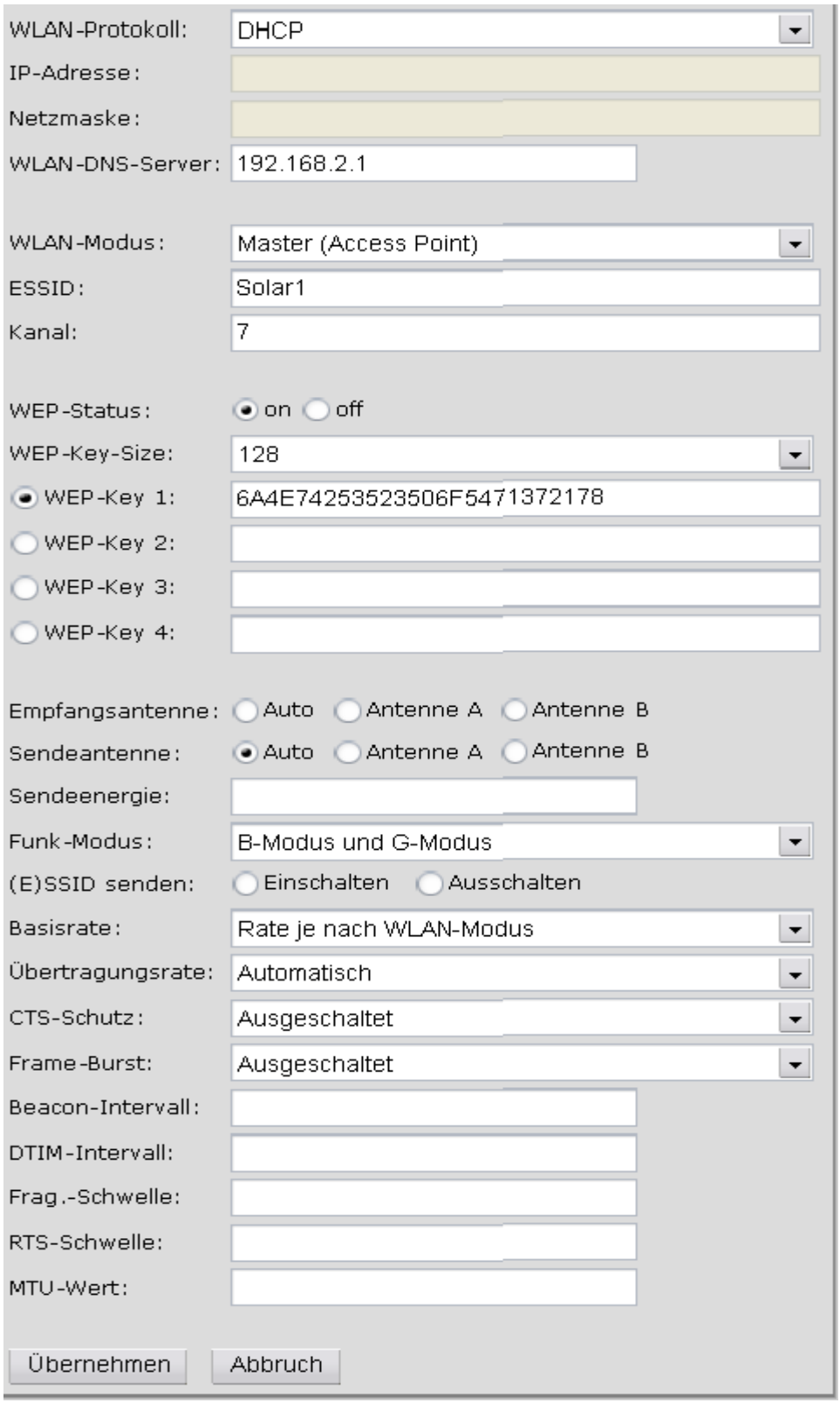


Abbildung 34: Verwaltung : Drahtlos

5.2.7. Einstellen über die Kommandozeile

Alle Einstellungen lassen sich auch über Telnet und SSH Secure Shell machen. Eine Verbindung über Telnet ist nur möglich wenn nichts über den WAN-Port angeschlossen ist.

Befehl	***Beschreibung***
`tftp -i <Router-IP> PUT <image.trx>`	Image auf Router Spielen
`passwd`	Passwort ändern
`reboot`	Router Neustarten
`mtd erase nvram`	Löschen der bisher gemachten Einstellungen
`nvram set lan_ipaddr=192.168.0.1`	IP-Adresse des Routers
`nvram commit`	
`nvram set wan_ifname=vlan1`	
`nvram set wan_proto=dhcp`	
`nvram set lan_ifname=br0`	
`nvram set lan_ifnames="vlan0 eth1"`	
`nvram set lan_proto=static`	
`nvram set lan_netmask=255.255.255.0`	
`nvram set lan_gateway=192.168.0.1`	Standardgateway
`nvram set dhcp_start=192.168.0.50`	Erste DHCP-IP-Adresse
`nvram set dhcp_end=192.168.0.75`	Lezte DHCP-IP-Adresse
`nvram set wl0_channel=2`	WLAN-Kanal
`nvram set wl0_mode=ap`	WLAN-Modus
`nvram set wl0_ssid=TestWLAN`	Name des WLAN-Routers
`nvram set wl0_infra=1`	
`nvram set wl0_closed=0`	
`nvram set wl0_akm=psk`	
`nvram set wl0_crypto=tkip`	
`nvram set wl0_wpa_psk=MySecretPassword`	

`nvram set wl0_key1=XXXXXXXXXX`
`nvram set wl0_key2=XXXXXXXXXX`
`nvram set wl0_key3=XXXXXXXXXX`
`nvram set wl0_key4=XXXXXXXXXX`
	WEP-Key's

`nvram set wl0_closed=0`	
`nvram set wl0_wep=enabled`	WEP-Verschlüsselung aktivieren
`nvram set wl0_wep_bit=64`	Schlüssellänge
`nvram set wl0_key=1`	

5.3. Benötigte Pakete nachinstallieren

5.3.1. Manuell de- und installieren

Eine Software-Datei muss möglicherweise manuell installiert und konfiguriert werden. Lade dazu zunächst die Software-Datei. Die Installation erfolgt dann mit der SSH-Kommandozeile (neu-deutsch: Shell) des Linux-Betriebssystems.

Tipp: Windows-Benutzer können die Freeware PuTTY für SSH verwenden.

Beispiel: Es soll die Software-Datei Beispiel_1.0_mipsel.ipk installiert werden. Nach dem Laden der IPK-Datei erfolgt auf der SSH-Kommandozeile die Installation mit dieser Eingabe: ipkg install /tmp/beispiel_1.0_mipsel.ipk [Eingabe-Taste].

Ist ein Internet-Verbindung vorhanden, können auch folgende Befehle verwendet werden:

ipkg update: Aktualisiert Paketlisten von den in /etc/ipkg.conf eingetragenen Servern.

ipkg list|less: Zeigt Informationen über verfügbare Software-Pakete an.

ipkg install name: Installiert ein Software-Paket via Internet.

ipkg remove name: löscht ein Software-Paket

Auf der Kommandozeile werden Befehle eingegeben und mit der Eingabe-Taste bestätigt. Die wichtigsten Befehle sind in Kapitel 4.1.6 beschrieben.

5.3.2. Automatische Installation

1. Web-Browser öffnen
2. http://192.168.2.200
3. Verwalten
4. Software

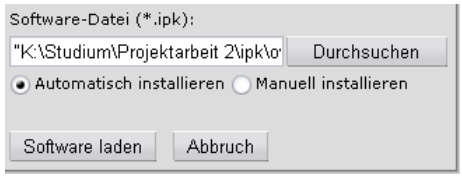


Abbildung 35: Verwalten > Software

5. Durchsuchen
6. dann gewünschtes Paket wählen
7. Software Laden

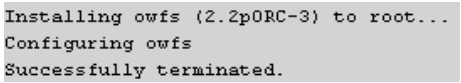


Abbildung 36: Installation Erfolgreich

5.3.3. Speicherplatzabfrage

Bei der Installation von Software muss man beachten, dass der Router nur begrenzten Speicherplatz besitzt. Der Speicher ist schneller voll als man denkt, deshalb sollte man ihn im Auge behalten. Abgefragt wird er mit:

```
df -h
Filesystem                Size      Used Available Use% Mounted on
/dev/root                 1.8M      1.8M         0 100% /rom
/dev/mtdblock/4           1.4M      1.2M    236.0k  84% /
none                     14.9M    240.0k     14.7M   2% /tmp
/dev/scsi/host0/bus0/target0/lun0/part1
                        119.5M      3.0k    119.5M   0% /stick
```

Wenn der eingebaute Flash-Speicher für die Software nicht ausreicht, dann gibt es die Möglichkeit das System auch von einem externen Speicher (USB-Stick oder HDD) laufen zu lassen. Nur mit dem internen Flash-Speicher konnte nicht mal ein Update des Systems durchgeführt werden, da der Speicherplatz nicht ausreicht. Mehr dazu unter:

http://wiki.openwrt.org/UsbStorageHowto#head-0a12e438860955ebd81c911a67c9222f80b89ad0

Da diese Methode zum jetzigen Zeitpunkt nicht erprobt war haben wir uns entschieden, Dateien die nicht im Rom-Bereich des Flash-Speichers und beim booten nicht benötigt werden, bevor der USB-Stick eingebunden wird, auf den USB-Stick auszulagern und auf das alte Verzeichnis zu verlinken. Dies kann man mit dem Befehl `ln` machen. Um die Tipparbeit ein bisschen zu vereinfachen haben wie eine Stapelverarbeitungsdatei /stick/usr/bin/links.sh geschrieben.

```
#!/bin/sh
mkdir -p $2
mv $1 $2
ln -s $1$2
```

Das Sktipt wird wie folgt verwendet:
`links.sh` <Datei> <Verteichnis>
`links.sh owfs /stick/usr/bin/`
Wobei die Datei die Verschoben und verlinkt wird im aktuellen Verzeichnis liegen muss.

5.3.4. Zugriff auf Dateien über SSH Secure File Transfer Client

Um den Zugriff über den SSH Secure File Transfer Client zu ermöglichen muss das Paket openssh-sftp-server installiert werden. Dazu müssen folgende Befehle in der Shell eingegeben werden:

```
ipkg update
ipkg install openssh-sftp-server
reboot
```

Anmeldung erfolgt wie im SSH Secure Shell Client.

Hinweis: Aus Speicherplatzgründen müssen neue Dateien auf den Stick verschoben und verlinkt werden. Siehe Kapitel 5.3.3

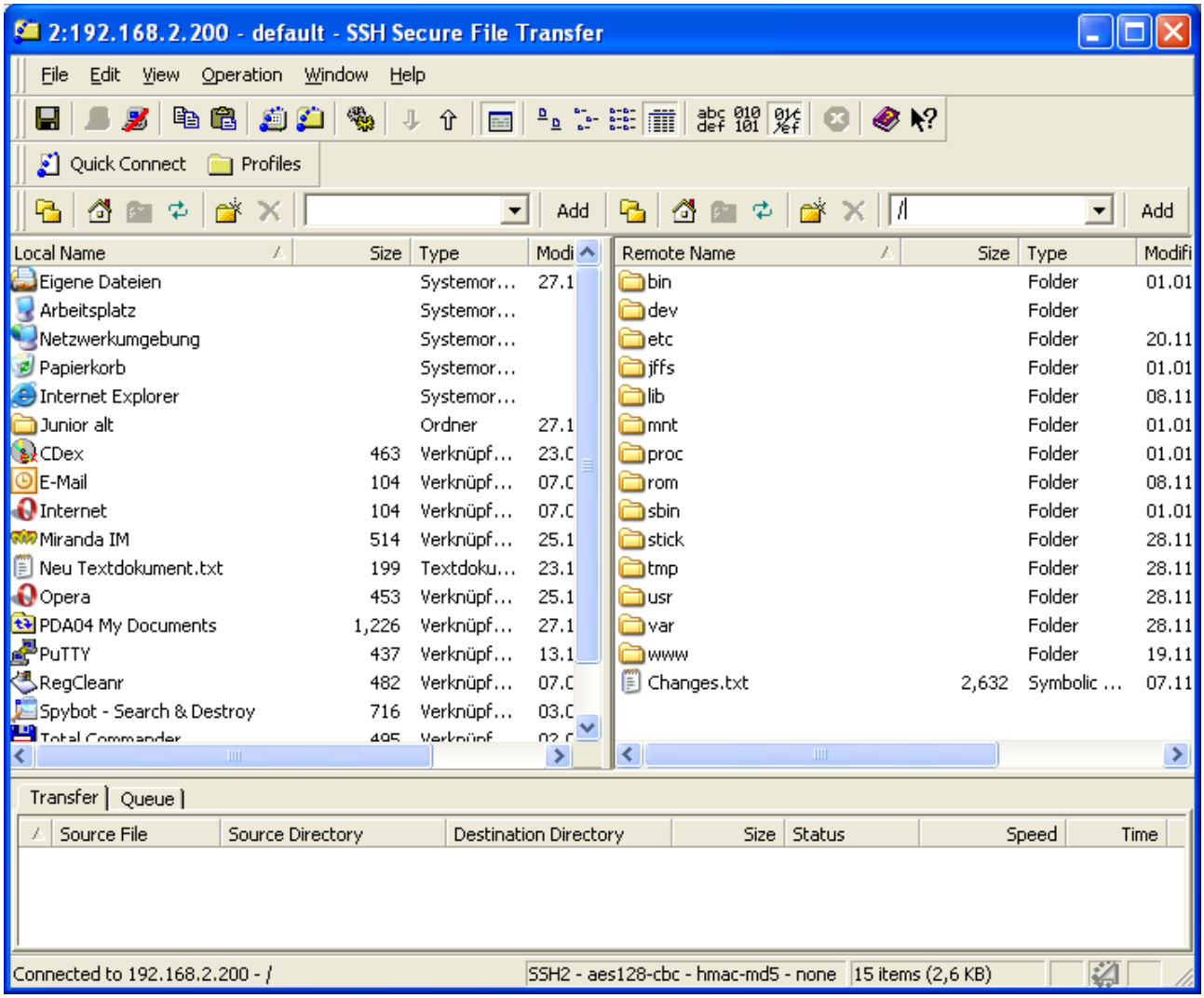


Abbildung 37: SSH File Transfer Client

5.3.5. Zeit automatisch Synchronisieren

1. Reponsitories aktualisieren:

```
ipkg update
Downloading http://downloads.openwrt.org/whiterussian/packages/Packages
Updated list of available packages in /usr/lib/ipkg/lists/whiterussian
Downloading http://downloads.openwrt.org/whiterussian/packages/non-
free/Packages
Updated list of available packages in /usr/lib/ipkg/lists/non-free
Successfully terminated.
```

2. Paket ntpclient installieren:

```
ipkg install ntpclient
Installing ntpclient (2003_194-1) to root...
Downloading
http://downloads.openwrt.org/whiterussian/packages/ntpclient_2003_194-
1_mipsel.ipk
Configuring ntpclient
Successfully terminated.
```

3. Konfigurationsdatei erstellen /etc/TZ mit dem Inhalt:

```
echo „CET-1CEST-2,M3.5.0/02:00:00,M10.5.0/03:00:00“ >>
/etc/TZ
```

Der Router mach die Umstellung Sommer-/Winterzeit automatisch.

4. Start-Skript /etc/init.d/S60ntpclient ersetzten durch:

```
#!/bin/sh
/usr/sbin/ntpclient -s -c 0 -i 600 -g 1000000 -h pool.ntp.org
```

Jetzt holt sich der Router alle 10 Minuten die aktuelle Uhrzeit.

5. Router neu booten:

```
reboot
```

5.3.6. Installation der USB-Schnittstelle incl. Massenspeicher

5.3.6.1. USB-Schnittstelle

Mindestanforderungen: Diese Funktion wird unterstützt von Routern mit einer USB-Schnittstelle (z.B. Asus WL-500G). Als Betriebssystem sollte ein OpenWRT-Linux (RC2 oder neuer) laufen, Systeme die auf OpenWRT aufbauen, wie das OpenNET sind ebenso geeignet. Wenn die ausgeführten USB-Ports nicht ausreichen sollten, kann auch ein USB-Hub verwendet werden.

Hinweis zur Installation: Manche Router sind USB 1.1 und 2.0 kompatibel. Um beide Standards zu nutzen sollten beide Versionen installiert sein.

Modul	***Beschreibung***
kmod-usb-core	Dies ist ein USB-Basismodul, es wird für beide Versionen benötigt.
kmod-usb-uhci	Für die USB 1.1 Unterstützung. Hinweis: Die meisten Chipsätze haben UHCI Controller. Wenn dieser nicht funktionieren sollte, ist einer der wenigen OHCI Controller im Einsatz.
kmod-usb-ohci	
kmod-usb2	USB 2.0 Unterstützung.

1. Reposnitories aktualisieren:

```
ipkg update
Downloading http://downloads.openwrt.org/whiterussian/packages/Packages
Updated list of available packages in /usr/lib/ipkg/lists/whiterussian
Downloading http://downloads.openwrt.org/whiterussian/packages/non-
free/Packages
Updated list of available packages in /usr/lib/ipkg/lists/non-free
Successfully terminated.
```

2. Benötigte Pakete installieren:

```
ipkg  install kmod-usb-core kmod-usb-uhci kmod-usb2 lsusb
Installing kmod-usb-core (2.4.30-brcm-2) to root...
Downloading http://downloads.openwrt.org/whiterussian/packages/kmod-usb-
core_2.4.30-brcm-2_mipsel.ipk
Installing kmod-usb-uhci (2.4.30-brcm-2) to root...
Downloading http://downloads.openwrt.org/whiterussian/packages/kmod-usb-
uhci_2.4.30-brcm-2_mipsel.ipk
Installing kmod-usb2 (2.4.30-brcm-2) to root...
Downloading http://downloads.openwrt.org/whiterussian/packages/kmod-
usb2_2.4.30-brcm-2_mipsel.ipk
Installing lsusb (0.71-1) to root...
```

```
Downloading
http://downloads.openwrt.org/whiterussian/packages/lsusb_0.71-
1_mipsel.ipk
Configuring kmod-usb-core
Configuring kmod-usb-uhci
Configuring kmod-usb2
Configuring lsusb
Successfully terminated.
```

3. Link auf /etc/init.d/S10boot:

```
rm /etc/init.d/S10Boot
```

4. Originaldatei aus dem Archiv kopieren:

```
cp /rom/etc/init.d/S10boot /etc/init.d/S10boot
```

5. Folgende Zeile am Ende der Datei (/etc/init.d/S10boot) einfügen

```
mount -t usbfs none /proc/bus/usb
```

6. Jetzt sollte man der Router neu starten. Im SSH-Client:

```
reboot
```

7. Nach dem Neustart sollte überprüft werden, ob die Geräte richtig erkannt wurden. Im SSH-Client:

```
dmesg
usb.c: registered new driver usbdevfs
usb.c: registered new driver hub
uhci.c: USB Universal Host Controller Interface driver v1.1
PCI: Enabling device 01:02.0 (0000 -> 0001)
uhci.c: USB UHCI at I/O 0x100, IRQ 2
usb.c: new USB bus registered, assigned bus number 1
hub.c: USB hub found
hub.c: 2 ports detected
PCI: Enabling device 01:02.1 (0000 -> 0001)
uhci.c: USB UHCI at I/O 0x120, IRQ 2
usb.c: new USB bus registered, assigned bus number 2
hub.c: USB hub found
hub.c: 2 ports detected
hub.c: new USB device 01:02.0-2, assigned address 2
usb.c: USB device 2 (vend/prod 0xd7d/0x100) is not claimed by any active
driver.
Initializing USB Mass Storage driver...
usb.c: registered new driver usb-storage
scsi0 : SCSI emulation for USB Mass Storage devices
  Vendor: Apacer    Model: Drive             Rev: 1.05
  Type:   Direct-Access                      ANSI SCSI revision: 02
Attached scsi removable disk sda at scsi0, channel 0, id 0, lun 0
SCSI device sda: 256000 512-byte hdwr sectors (131 MB)
sda: Write Protect is off
Partition check:
 /dev/scsi/host0/bus0/target0/lun0: p1
WARNING: USB Mass Storage data integrity not assured
USB Mass Storage device found at 2
USB Mass Storage support registered.
```

8. Abfrage der Geräte, die an den USB-Ports hängen:

```
root@Solar1:~# lsusb
Bus 003 Device 001: ID 0000:0000
Bus 002 Device 001: ID 0000:0000
Bus 001 Device 001: ID 0000:0000
```

```
Bus 001 Device 002: ID 09a6:8001 Poinchips
```

5.3.6.2. Massenspeicher

Mindestanforderungen: Eine externe Festplatte oder USB-Stick sollte vorhanden sein.

Modul	*Beschreibung*
kmod-usb-storage	Modul für USB-Massenmedien
kmod-vfat	VFAT/MSDOS Unterstützung.
kmod-ext2	EXT2 Unterstützung.
kmod-ext3	EXT3 Unterstützung.

Abbildung 38: TrekStor USB-Stick 128MB

1. Reponsitories aktualisieren:

```
ipkg update
Downloading http://downloads.openwrt.org/whiterussian/packages/Packages
Updated list of available packages in /usr/lib/ipkg/lists/whiterussian
Downloading http://downloads.openwrt.org/whiterussian/packages/non-
free/Packages
Updated list of available packages in /usr/lib/ipkg/lists/non-free
Successfully terminated.
```

2. Benötigte Pakete installieren:

```
ipkg  install kmod-usb-storage kmod-ext2 kmod-ext3 kmod-vfat
Installing kmod-usb-storage (2.4.30-brcm-2) to root...
Downloading http://downloads.openwrt.org/whiterussian/packages/kmod-usb-
storage_2.4.30-brcm-2_mipsel.ipk
Installing kmod-ext2 (2.4.30-brcm-2) to root...
Downloading http://downloads.openwrt.org/whiterussian/packages/kmod-
ext2_2.4.30-brcm-2_mipsel.ipk
Package kmod-ext2 (2.4.30-brcm-2) installed in root is up to date.
Installing kmod-ext3 (2.4.30-brcm-2) to root...
Downloading http://downloads.openwrt.org/whiterussian/packages/kmod-
ext3_2.4.30-brcm-2_mipsel.ipk
Installing kmod-vfat (2.4.30-brcm-2) to root...
Downloading http://downloads.openwrt.org/whiterussian/packages/kmod-
vfat_2.4.30-brcm-2_mipsel.ipk
Configuring kmod-usb-storage
Configuring kmod-ext2
Configuring kmod-ext3
Configuring kmod-vfat
Successfully terminated.
```

3. Installation des fdisk-Tools:

```
ipkg install
http://downloads.openwrt.org/people/nico/testing/mipsel/packa
ges/fdisk_2.12r-1_mipsel.ipk
```

4. Abfrage der Geräte die an den USB-Ports hängen:

```
root@Solar1:~# lsusb
Bus 003 Device 001: ID 0000:0000
Bus 003 Device 002: ID 090c:1000 Feiya Technology Corp.
Bus 002 Device 001: ID 0000:0000
Bus 001 Device 001: ID 0000:0000
```

5. Erstellen eines Verzeichnisses, als Einhängepunkt: /mnt

```
mkdir -p /mnt
```

6. Auswählen der Partition, die eingehängt werden soll:

```
fdisk -l
```

7. Um auf einen Datenträger zuzugreifen muss man ihn mounten (einhängen):

```
mount /dev/scsi/host0/bus0/target0/lun0/part1 /mnt
```

8. Aushängen eines Datenträgers:

```
umount /mnt
```

5.3.7. Installation von OWFS

Hinweis: Bei der Verwendung von USB-1Wire Adaptern muss auch die USB-Schnittstelle installiert und konfiguriert sein

1. Download der benötigten Pakete von http://home.mag.cx/openwrt/packages:
   owfs_2.2p0RC-9_mipsel.ipk
   owlib_2.2p0RC-9_mipsel.ipk
   libpthread_0.9.27-1_mipsel.ipk
   kmod-fuse_2.4.30brcm+2.4.1-1_mipsel.ipk
   libfuse_2.4.1-1_mipsel.ipk
   libusb_0.1.10a-1_mipsel.ipk
   fuse-utils_2.4.1-1_mipsel.ipk
2. Jetzt müssen die heruntergeladenen Dateien installiert werden, wie in Kapitel 5.3.2 beschrieben ist.

   Hinweis: Um Probleme zu vermeiden müssen die ow-Systeme gestoppt werden und die Start-Scripte gelöscht werden. Wenn man dies nicht macht, dann stürzt das System ab und friert ein. Auch ein erneutes booten bringt nichts, der Router bleibt nicht mehr ansprechbar. Nur durch erneutes über- und einspielen der Firmware konnten wir das System wieder lauffähig machen. Die Problemstellung haben wir auch in einem Forum gefunden, die Ursache konnte leider auch nicht festgestellt werden, allerdings wurde eine Abhilfe gefunden, die funktioniert.

```
/etc/init.d/S70owserver stop
/etc/init.d/S80owfs stop
/etc/init.d/S80owhttp stop
rm /etc/init.d/S70owserver
rm /etc/init.d/S80owfs
rm /etc/init.d/S80owhttpd
rm /usr/bin/run-owfs
```

3. Versuche jetzt owfs manuell zu starten:

```
insmod fuse
/usr/bin/owserver -u /dev/ttyS0 /dev/ttyS1 -p 3333
mkdir -p /tmp/1Wire
/usr/bin/owfs -s 3333 /tmp/1Wire
/usr/bin/owhttpd -s 3333 -p 3001
```

4. Wenn der Start des 1Wire-System erfolgreich war, muss noch ein Start-Script mit den verwendeten Befehlen angelegt werden:

```
vi /etc/init.d/S70owsystem
```

5. Bein nächsten Neustart des Routers wird das 1-Wire-System automatisch ausgeführt.
6. Jetzt können die 1-Wire Geräte von jedem Rechner, im Netzwerk, abgefragt werden.

   Browser starten

   http://<Router-IP>:3001/

Hinweis: Aus Speicherplatzgründen müssen neue Dateien auf den Stick verschoben und verlinkt werden. Siehe Kapitel 5.3.3.

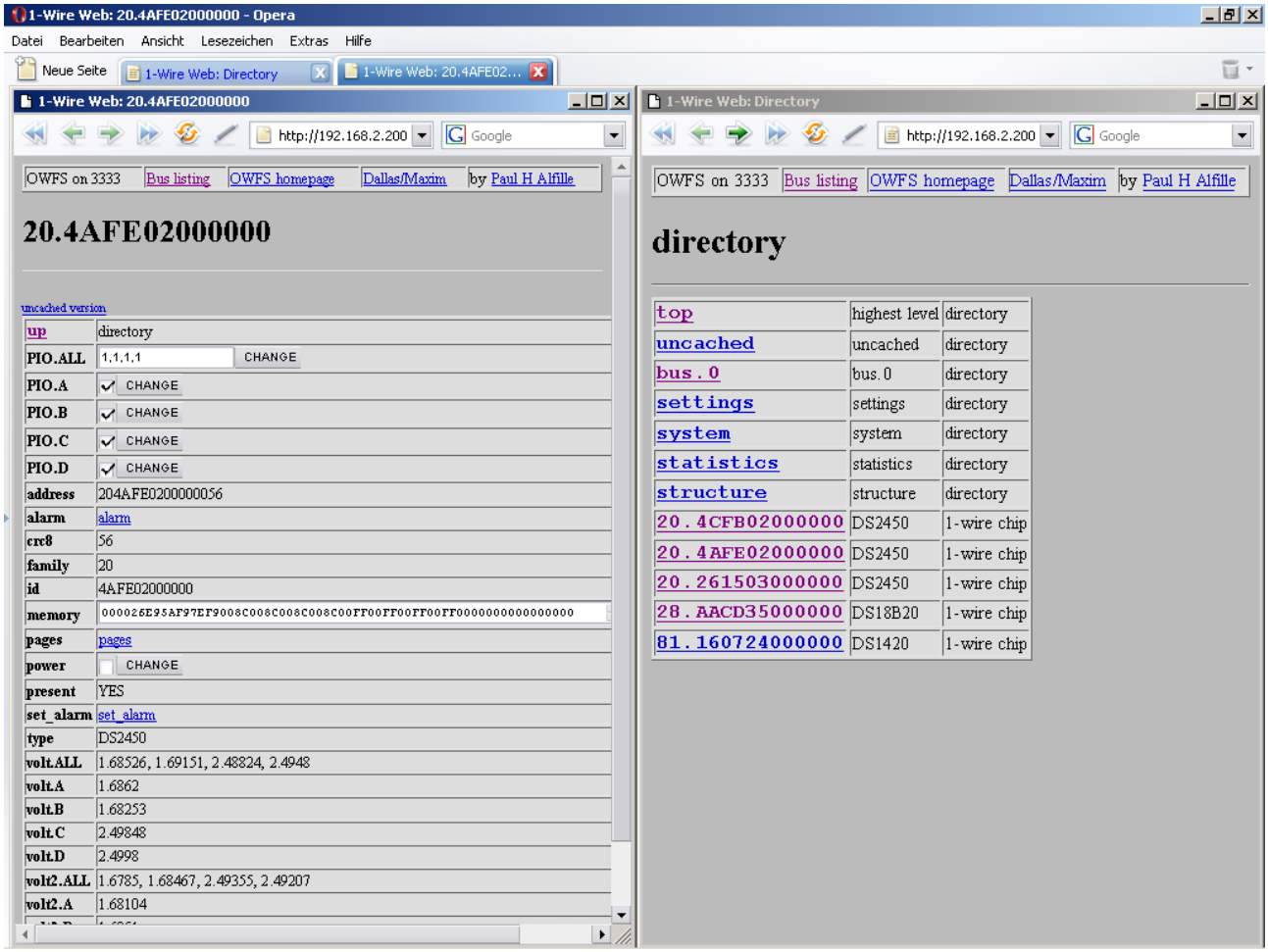


Abbildung 39: http-Ausgabe des OW-Fs

5.3.8. Installation von Temploggerd

Hinweis: Bei der Verwendung von Temploggerd Adaptern muss auch das 1Wire OWFS und die USB-Schnittstelle installiert und konfiguriert sein.

1. Download der benötigten Pakete von http://home.mag.cx/openwrt/packages:

   librrd1_1.0.50-1_mipsel.ipk
   rrdcgi1_1.0.50-1_mipsel.ipk
   rrdtool1_1.0.50-1_mipsel.ipk
   temploggerd_1.3.2-2_mipsel.ipk
2. Jetzt müssen die heruntergeladenen Dateien installiert werden, wie in Kapitel 5.3.2 beschrieben.
3. Versuche jetzt Temploggerd manuell zu starten (Dies kann 30s dauern.):

```
/etc/init.d/S81temploggerd
```

4. Bein nächsten Neustart des Router wird das 1-Wire-System automatisch gestartet.
5. Jetzt können die 1-Wire Temperatursensoren von jedem Rechner im Netzwerk abgefragt werden.

   Browser starten

   http://<Router-IP>/temploggerd
6. Die generierten Webseiten können, durch Änderung der Vorlage Dateien angepasst werden. Die Dateien befinden sich in /usr/share/temploggerd/templates/ und müssen vor dem Start von temploggerd angepasst werden. Die Grunddateien heissen:

   index.html.templ all_temperature.cgi.templ sensor_temperature.cgi.templ all.cgi.templ

Hinweis: Aus Speicherplatzgründen müssen neue Dateien auf den Stick verschoben und verlinkt werden. Siehe Kapitel 5.3.3.

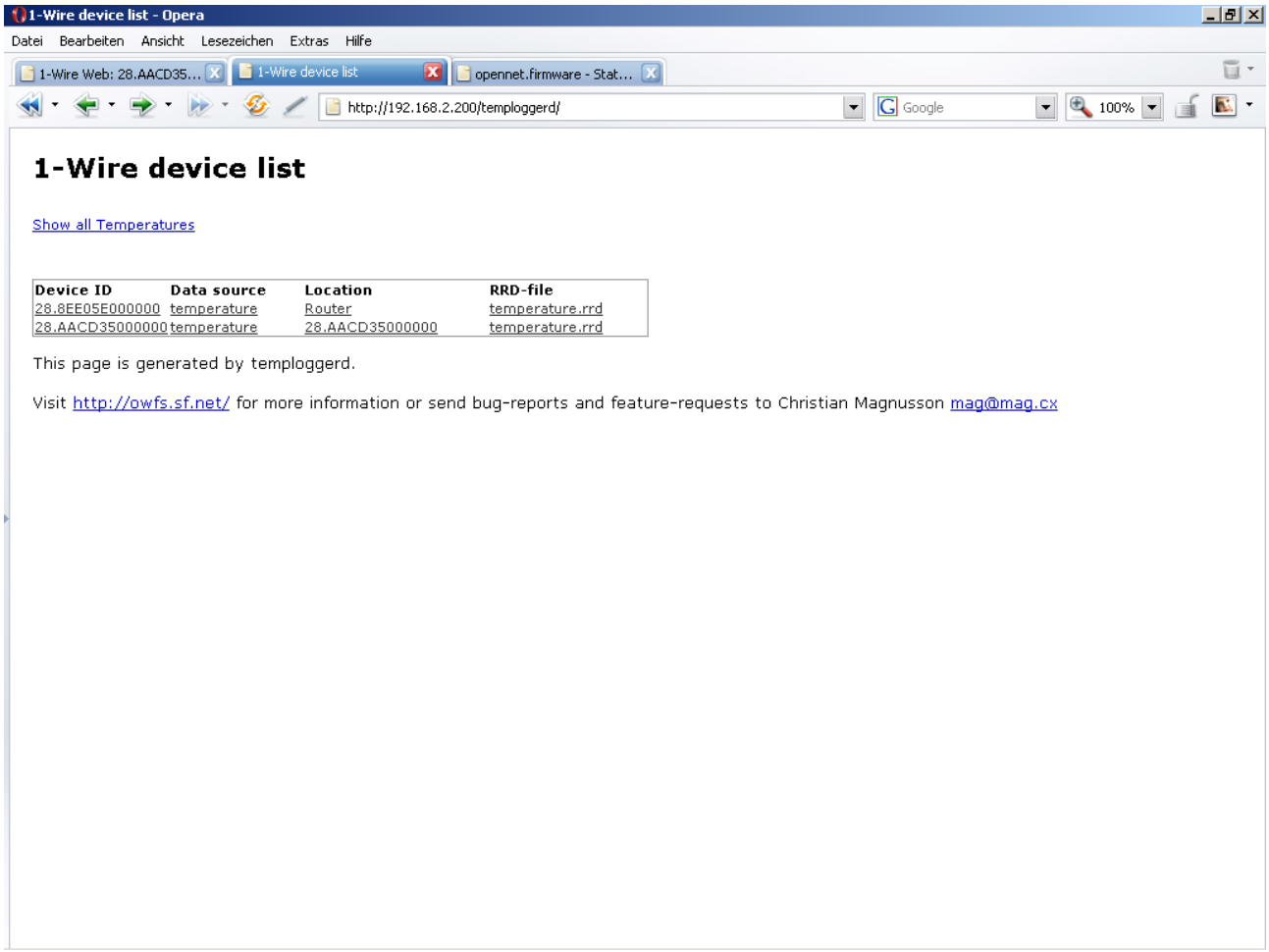


Abbildung 40: Temperaturlogger Hauptseite

5.3.8.1. Automatisches sichern und zurückspielen der Daten für Temploggerd

Alle Dateien, die von Temploggerd generiert werden werden standardmäßig im /var/lib/temploggerd/ Verzeichnis gespeichert. Diese Daten gehen verloren, wenn der Router neu startet (z.B. mit dem Befehl reboot oder bei Stromausfall), denn sie liegen im Ramdisk (ramfs). Aus diesem Grund haben wir den Router veranlasst die Daten regelmäßig auf einem USB-Stick zu sichern. Diese Methode schont den USB-Stick, da er nur auf eine bestimmte Anzahl von Schreibzyklen ausgelegt ist.

Die meisten USB-Sticks sind ausgelegt für eine Lebensdauer von ca. 10 Jahren und 1.000.000 Schreibzyklen.

$$T_{tot} = \frac{\overbrace{1000000}^{max.\ Anz.\ Schreibzyklen}}{\underbrace{\frac{4}{h}}_{Anz.\ Sich.\ pro\ Stunde} * 24\frac{h}{day} * 365\frac{day}{a}} = 28{,}62\,a$$

Selbst wenn wir 4 mal pro Stunde sichern, reichen uns die 1.000.000 Schreibzyklen für 28 Jahre. Den einzigen Nachteil, den diese Methode hat, ist dass bei einem Neustart des Routers ein paar Datensätze verloren gehen. Eine Festplatte hat zwar keine Begrenzung an Schreibzyklen und hat viel mehr Speicherplatz ist aber auch viel teurer (ca.60€), braucht mehr Energie und hat mechanische Verschleißteile. Nach der Abwägung der Gründe haben wir uns für einen USB-Stick entschieden.

1. Anlegen der Stapelverarbeitungsdatei zum sichern der Dateien von Temploggerd:

```
vi /usr/bin/sicher.sh
```

   mit dem Inhalt:

```
rm -r /stick/temploggerd
cp -r /var/lib/temploggerd /stick/
```

2. Einstellen, dass die Datei /usr/bin/sicher.sh gestartet wird und die Daten regelmäßig sichert:

```
vi /etc/init.d/S51crond
```

   Der Inhalt muss wie folgt geändert werden:

```
#!/bin/sh

test -n "$FAILSAFE" && exit
#test -z "$(awk 'sub(":","") {print $1}' /proc/net/Wireless)"
&& exit

if [ ! -d /var/spool/cron/crontabs ]; then
  mkdir -p /var/spool/cron/crontabs
  cat>/var/spool/cron/crontabs/root<<EOF
0-59/1 * * * * /usr/sbin/cron.minutely
0 * * * * /usr/sbin/cron.hourly
0 0 * * * /usr/sbin/cron.daily
0 * * * * /usr/bin/sicher.sh
EOF
fi
/usr/sbin/crond -L /dev/null
```

3. Jetzt wird eine Stapelverarbeitungsdatei erstellt die automatisch beim booten den USB-Stick mountet und die Daten für Temploggerd rekonstruieren soll.

```
vi /etc/init.d/S50stick
```

   Mit dem Inhalt:

```
mkdir -p /stick/
mount /dev/scsi/host0/bus0/target0/lun0/part1 /stick
mkdir -p /var/lib
cp -r /stick/temploggerd /var/lib
```

4. Jetzt muss nur noch der Router neu gebootet werden:

```
reboot
```

Es gibt noch eine andere Möglichkeit die Daten von Temploggerd zu sichern: In der Datei /etc/temploggerd.conf gibt es zwei Einstellungsmöglichkeiten „backup_dir“ und „backup_freq“ die standardmäßig mit „#“ auskommentiert sind. Mit „backup_dir“ wird das Verzeichnis angegeben, in dass gesichert werden soll (backup_dir /opt/backup). Mit „backup_freq“ wird das Zeitintervall (in Sekunden) angegeben wie oft gesichert werden soll (backup_dir /opt/backup).

Hinweise:

1. Vor dem manuellen Neustart des Routers sollte die Stapelverarbeitungsdatei /usr/bin/sicher.sh ausgeführt werden um die Daten vom Temploggerd zu sichern.
2. Aus Speicherplatzgründen müssen neue Dateien auf den Stick verschoben und verlinkt werden. Siehe Kapitel 5.3.3.

5.3.9. Installation von Tcl und OW-Tcl

Hinweis: Bei der Verwendung von USB-1Wire Adaptern muss auch die USB-Schnittstelle installiert und konfiguriert sein

1. Download der benötigten Pakete von http://home.mag.cx/openwrt/packages:
   owtcl_2.2p0RC-9_mipsel.ipk
   tcl8_8.4.11-5_mipsel.ipk
2. Jetzt müssen die heruntergeladenen Dateien installiert werden, wie in Kapitel 5.3.2 beschrieben ist.

Hinweis: Aus Speicherplatzgründen müssen neue Dateien auf den Stick verschoben und verlinkt werden. Siehe Kapitel 5.3.3.

5.3.10. Installation von Php und OW-Php

Hinweis: Bei der Verwendung von USB-1Wire Adaptern muss auch die USB-Schnittstelle installiert und konfiguriert sein

1. Download der benötigten Pakete von http://home.mag.cx/openwrt/packages:
   owphp_2.2p0RC-9_mipsel.ipk
   php4-cli_4.3.11-3_mipsel.ipk
   php4-mod-sockets_4.3.11-3_mipsel.ipk
2. Jetzt müssen die heruntergeladenen Dateien installiert werden, wie in Kapitel 5.3.2 beschrieben ist.

Hinweis: Aus Speicherplatzgründen müssen neue Dateien auf den Stick verschoben und verlinkt werden. Siehe Kapitel 5.3.3.

5.3.11. Installierte Pakete auf dem Router

Die unten aufgelisteten Pakete sind zum Zeitpunkt der Abgabe auf dem Router installiert. Sie kann abgefragt mit dem Befehl:

```
ipkg list_installed
```

Wenn einige Pakete unterschiedliche Versionsnummer zu den oben genannten aufweisen, dann wurden von diesen Paketen im nachhinein noch neuere Versionen nachinstalliert, Paketabhängigkeiten wurden berücksichtigt und auch installiert.

Paketname	*Version*	*Beschreibung*
bridge	1.0.6-1	Ethernet bridging tools
busybox	1.00-2	
bwm	1.1.0-1	A very tiny bandwidth monitor
bwm-ng	0.5-1	
dnsmasq	2.22-1	
dropbear	0.45-4	a small SSH 2 server/client designed for small memory environments.
Fuse-utils	2.4.1-1	Filesystem in Userspace (utilities)
ip	2.6.11-050330-1	iproute2 routing control utility
ipkg	0.99.149-2	lightweight package management system
iptables	1.3.3-1	The netfilter firewalling software for IPv4
kernel	2.4.30-brcm-2	
kmod-brcm-et	2.4.30-brcm-2	Proprietary driver for Broadcom Ethernet chipsets
kmod-brcm-wl	2.4.30-brcm-2	Proprietary driver for Broadcom Wireless chipsets
kmod-diag	2.4.30-brcm-2	Driver for Router LEDs and Buttons
kmod-fuse	2.4.30brcm+2.4.1-1	
kmod-ppp	2.4.30-brcm-2	PPP support
kmod-pppoe	2.4.30-brcm-2	PPP over Ethernet support
kmod-tun	2.4.30-brcm-2	Kernel TUN/TAP extension
kmod-usb-core	2.4.30-brcm-2	Kernel Support for USB
kmod-usb-storage	2.4.30-brcm-2	Kernel modules for USB storage support
kmod-usb-uhci	2.4.30-brcm-2	Kernel driver for UHCI USB controllers
kmod-usb2	2.4.30-brcm-2	Kernel driver for USB2 controllers
kmod-vfat	2.4.30-brcm-2	Kernel modules for VFAT filesystem support
kmod-wlcompat	2.4.30-brcm-2	
libfuse	2.4.1-1	Filesystem in Userspace (library)
liblzo	1.08-1	a real-time data compression library
libopenssl	0.9.7g-1	
libpthread	0.9.27-1	POSIX threads library
librrd1	1.0.50-1	Round Robin Database (RRD) management library.
Libusb	0.1.10a-1	a Library for accessing Linux USB devices
lsusb	0.71-1	A program to list USB devices
ntpclient	2003_194-3	
olsrd	0.4.9-1	The olsr.org OLSR daemon
olsrd-mod-httpinfo	0.4.9-1	a small informative web server plugin for olsrd

Paketname	*Version*	*Beschreibung*
openssh-sftp-server	4.2p1-1	OpenSSH SFTP server
openvpn	2.0.5-1	Open Source VPN solution using SSL
openwrt-utils	2	
owfs	2.2p0RC-9	1-Wire file system
owlib	2.2p0RC-9	1-Wire file system
owphp	2.2p0RC-9	1-Wire file system
owtcl	2.2p0RC-9	1-Wire file system
php4-cli	4.3.11-3	PHP4 CLI (Command Line Interface)
php4-mod-sockets	4.3.11-3	Sockets module for PHP4
ppp	2.4.3-7	a PPP (Point-to-Point Protocol) daemon (with MPPE/MPPC support)
ppp-mod-pppoe	2.4.3-7	a PPPoE (PPP over Ethernet) plugin for PPP
rrdcgi1	1.0.50-1	Round Robin Database (RRD) CGI graphing tool.
rrdtool1	1.0.50-1	Round Robin Database (RRD) management tools.
Tcl8	8.4.11-5	TCL8 CLI (Command Line Interface)
temploggerd	1.3.3-1	temperature logger
wificonf	2	
wireless-tools	28.pre7-1	Tools for setting up WiFi cards using the Wireless Extension
zlib	1.2.2-2	an implementation of the deflate compression method (library)

5.3.12. Verschobene und verlinkte Dateien und Verzeichnisse

/bin/ipkg ==> /stick/bin/ipkg
/lib/libpthread-0.9.27.so ==> /stick/lib/libpthread-0.9.27.so
/sbin/ifup.pppoe ==> /stick/sbin/ifup.pppoe
/usr/bin/add_new_sensor.sh ==> /stick/usr/bin/add_new_sensor.sh
/usr/bin/fusermount ==> /stick/usr/bin/fusermount
/usr/bin/kill-owfs ==> /stick/usr/bin/kill-owfs
/usr/bin/kill-temploggerd ==> /stick/usr/bin/kill-temploggerd
/usr/bin/links.sh ==> /stick/usr/bin/links.sh
/usr/bin/owfs ==> /stick/usr/bin/owfs
/usr/bin/owhttpd ==> /stick/usr/bin/owhttpd
/usr/bin/owserver ==> /stick/usr/bin/owserver
/usr/bin/php ==> /stick/usr/bin/php
/usr/bin/rrd_replace ==> /stick/usr/bin/rrd_replace
/usr/bin/rrdcgi ==> /stick/usr/bin/rrdcgi
/usr/bin/rrdtool ==> /stick/usr/bin/rrdtool
/usr/bin/rrdupdate ==> /stick/usr/bin/rrdupdate
/usr/bin/run-owfs ==> /stick/usr/bin/run-owfs

/usr/bin/run-temploggerd	==>	/stick/usr/bin/run-temploggerd
/usr/bin/sicher.sh	==>	/stick/usr/bin/sicher.sh
/usr/bin/tclsh8.4	==>	/stick/usr/bin/tclsh8.4
/usr/bin/temploggerd	==>	/stick/usr/bin/temploggerd
/usr/lib/libfuse.so.2.4.1	==>	/stick/usr/lib/libfuse.so.2.4.1
/usr/lib/libipkg.so.0.0.0	==>	/stick/usr/lib/libipkg.so.0.0.0
/usr/lib/liblibrrd.so.0.0.0	==>	/stick/usr/liblibrrd.so.0.0.0
/usr/lib/libow.so.0.0.0	==>	/stick/usr/lib/libow.so.0.0.0
/usr/lib/libowphp.so.0.0.0	==>	/stick/usr/lib/libowphp.so.0.0.0
/usr/lib/librrd.so.0.0.0	==>	/stick/usr/lib/librrd.so.0.0.0
/usr/lib/libtcl8.4.so	==>	/stick/usr/lib/libtcl8.4.so
/usr/lib/php	==>	/stick/usr/lib/php/
/usr/lib/sftp-server	==>	/stick/usr/lib/sftp-server
/usr/lib/tcl8.4	==>	/stick/usr/lib/tcl8.4
/usr/sbin/dropbear	==>	/stick/usr/sbin/dropbear
/usr/sbin/iptables	==>	/stick/usr/sbin/iptables
/usr/sbin/lsusb	==>	/stick/usr/sbin/lsusb
/usr/sbin/pppd	==>	/stick/usr/sbin/pppd
/usr/share/temploggerd	==>	/stick/usr/share/temploggerd/
/etc/default	==>	/stick/etc/default/
/etc/ipkg.conf	==>	/stick/etc/ipkg.conf
/etc/php.ini	==>	/stick/etc/php.ini
/etc/temploggerd.conf	==>	/stick/etc/temploggerd.conf
/etc/temploggerd.conf.default	==>	/stick/etc/temploggerd.conf.default
/www/solar	==>	/stick/solar/
/www/cgi-bin/cgi-solar	==>	/stick/solar/cgi-solar

6. Einrichtung des Linux-PC's

Zuerst haben wir versucht alles unter Windows zu machen, aber wir mussten bald feststellen, dass manche Anleitungen besser oder nur für ein Linux-System gab, deshalb sind wir auch zweigleisig gefahren.

6.1. Installation von Ubuntu Linux 5.10

Eine Anleitung für die Installation von Ubuntu findet man unter folgender URL:

http://wiki.ubuntuusers.de/Ubuntu_installieren

6.2. Nachinstallation der benötigten Pakete

Die benötigten Pakete wurden aus Quellen heruntergeladen und installiert, die in der Datei /etc/apt/sources.list festgelegt sind. Inhalt der Datei /etc/apt/sources.list:

```
deb cdrom:[Ubuntu 5.10 _Breezy Badger_ - Release i386 (20051012)]/
breezy main restricted
deb http://de.archive.ubuntu.com/ubuntu breezy main restricted
deb-src http://de.archive.ubuntu.com/ubuntu breezy main restricted
deb http://de.archive.ubuntu.com/ubuntu breezy-updates main
restricted
deb-src http://de.archive.ubuntu.com/ubuntu breezy-updates main
restricted
deb http://de.archive.ubuntu.com/ubuntu breezy universe
deb-src http://de.archive.ubuntu.com/ubuntu breezy universe
deb http://de.archive.ubuntu.com/ubuntu breezy-backports main
restricted universe multiverse
deb-src http://de.archive.ubuntu.com/ubuntu breezy-backports main
restricted universe multiverse
deb http://security.ubuntu.com/ubuntu breezy-security main
restricted
deb-src http://security.ubuntu.com/ubuntu breezy-security main
restricted
deb http://archive.ubuntu.com/ubuntu hoary main restricted
deb-src http://archive.ubuntu.com/ubuntu hoary main restricted
deb http://archive.ubuntu.com/ubuntu hoary universe
deb-src http://archive.ubuntu.com/ubuntu hoary universe
deb http://archive.ubuntu.com/ubuntu/ hoary main restricted
deb-src http://archive.ubuntu.com/ubuntu/ hoary main restricted
deb http://archive.ubuntu.com/ubuntu/ hoary universe multiverse
deb-src http://archive.ubuntu.com/ubuntu/ hoary universe
deb http://www.linuxbh.org/naarea/ pacotes/
deb http://www.mpe.mpg.de/~ach/debian/sid/ ./
deb http://ftp.inf.tu-dresden.de/os/linux/dists/ubuntu hoary main
restricted universe multiverse
deb http://ftp.inf.tu-dresden.de/os/linux/dists/ubuntu hoary-
security main restricted universe multiverse
deb http://ftp.inf.tu-dresden.de/os/linux/dists/ubuntu hoary-
```

```
updates main restricted universe multiverse
deb http://ftp.inf.tu-dresden.de/os/linux/dists/ubuntu hoary-
backports main universe multiverse restricted
deb http://ubuntu-backports.mirrormax.net/ hoary-extras main
universe multiverse restricted
deb http://ftp.inf.tu-dresden.de/os/linux/dists/ubuntu/ breezy-
security main multiverse universe restricted
deb http://ftp.inf.tu-dresden.de/os/linux/dists/ubuntu/ breezy-
updates main multiverse restricted universe
deb http://ftp.inf.tu-dresden.de/os/linux/dists/ubuntu/ breezy
main multiverse universe restricted
deb http://de.archive.ubuntu.com/ubuntu breezy-backports main
universe multiverse restricted
deb http://archive.ubuntu.com/ubuntu breezy-backports main
restricted universe multiverse
deb http://ubuntu-backports.mirrormax.net/ breezy-extras main
restricted universe multiverse
deb http://ubuntu-backports.mirrormax.net/ breezy-extras-staging
main restricted universe multiverse
deb http://www.kruyt.org/debian /
deb http://ubuntu.tower-net.de/ubuntu/ hoary java
deb http://debian.meebey.net/ ./
deb http://www.jarre-de-the.net/computing/debian/ stable main
deb http://deb.opera.com/opera/ etch non-free
deb http://people.ubuntu.com/~doko/OOo2 ./
deb http://kubuntu.org/packages/kde351 breezy main
deb ftp://bolugftp.uni-bonn.de/pub/kde/stable/3.5.1/kubuntu breezy
deb ftp://ftp.tux.org/java/debian/ sarge non-free
```

Zuerst muss man die Repositories updaten mit:

```
apt-get update
```

Dann sollte man das System auf den neusten Stand bringen mit:

```
apt-get upgrade
```

Jetzt kann man die benötigten Pakete installieren:

```
apt-get install <Paketname>
```

Folgende Pakete müssen auf den PC vorhanden sein oder nachinstalliert werden:

- cvs
- tcl8.4
- tcl8.4-doc
- tcl8.4-dev
- tclx8.4
- tclx8.4-doc
- tk8.4-devtcl8.4
- tk8.4-doc
- tk8.4-dev

- gcc
- cpp
- g++
- automake-1.8
- autoconf
- autoheader
- libtool
- swig
- python-2.4
- python-2.4-dev

Es sind nur die Pakete die wir nachinstalliert habe und von denen wir Kenntnis haben dass sie benötigt werden. Natürlich müssen alle Paketabhängigkeiten berücksichtigt werden.

6.3. Kompilieren von OWFS mit TCL-Einbindung

OWFS gibt es auch als fertiges RPM-Paket und DEB-Paket, mit denen es aber nicht geht die 1-Wire Geräte anzusprechen, da vermutlich das OWFS-Paket ohne Tcl/Tk support compiliert wurde.

Kurzanleitung:

```
●cvs -d:pserver:anonymous@cvs.sourceforge.net:/cvsroot/owfs login
```

(leeres Passwort eingeben, Achtung, es kann sein, dass die FH Firewall da Probleme macht, notfalls auf die login-Maschine gehen)

```
●cvs -z3 -d:pserver:anonymous@cvs.sourceforge.net:/cvsroot/owfs co
-P owfs
```

Der Rest läuft Lokal auf dem Rechner ab

```
●cd owfs
●rm -rf *cache
●aclocal-1.8
●autoheader
●autoconf
●automake-1.8
●./configure --with-tcl=/usr/lib/tcl8.4/ --enable-owtcl --enable-
owperl --enable-owpython --enable-owphp --enable-usb --enable-
owcapi --prefix=/usr/local/owfs
●make
●make install
```

7. Hardware

7.1. Platine analoge Eingängen

7.1.1. Allgemein

Diese Platine ist mit 12 Analogen Eingängen bestückt. Die Eingänge der Bausteine sind auf eine Spannung von 0 bis 5 V Ausgelegt. Die Platine muss mit einer Spannung von 9 bis 12 V versorgt werden.

7.1.2. Pläne der Platine

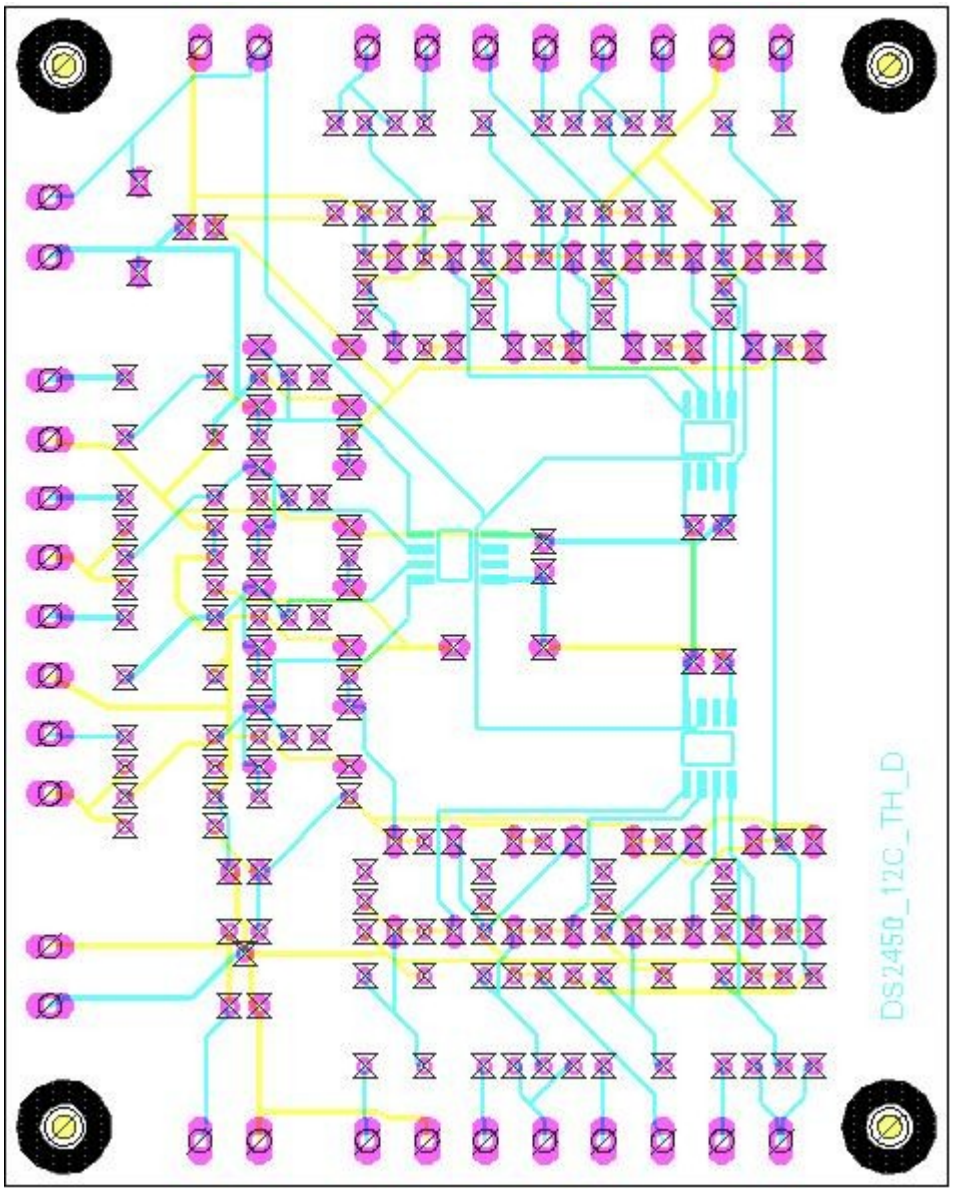


Abbildung 41: Layout der Platine

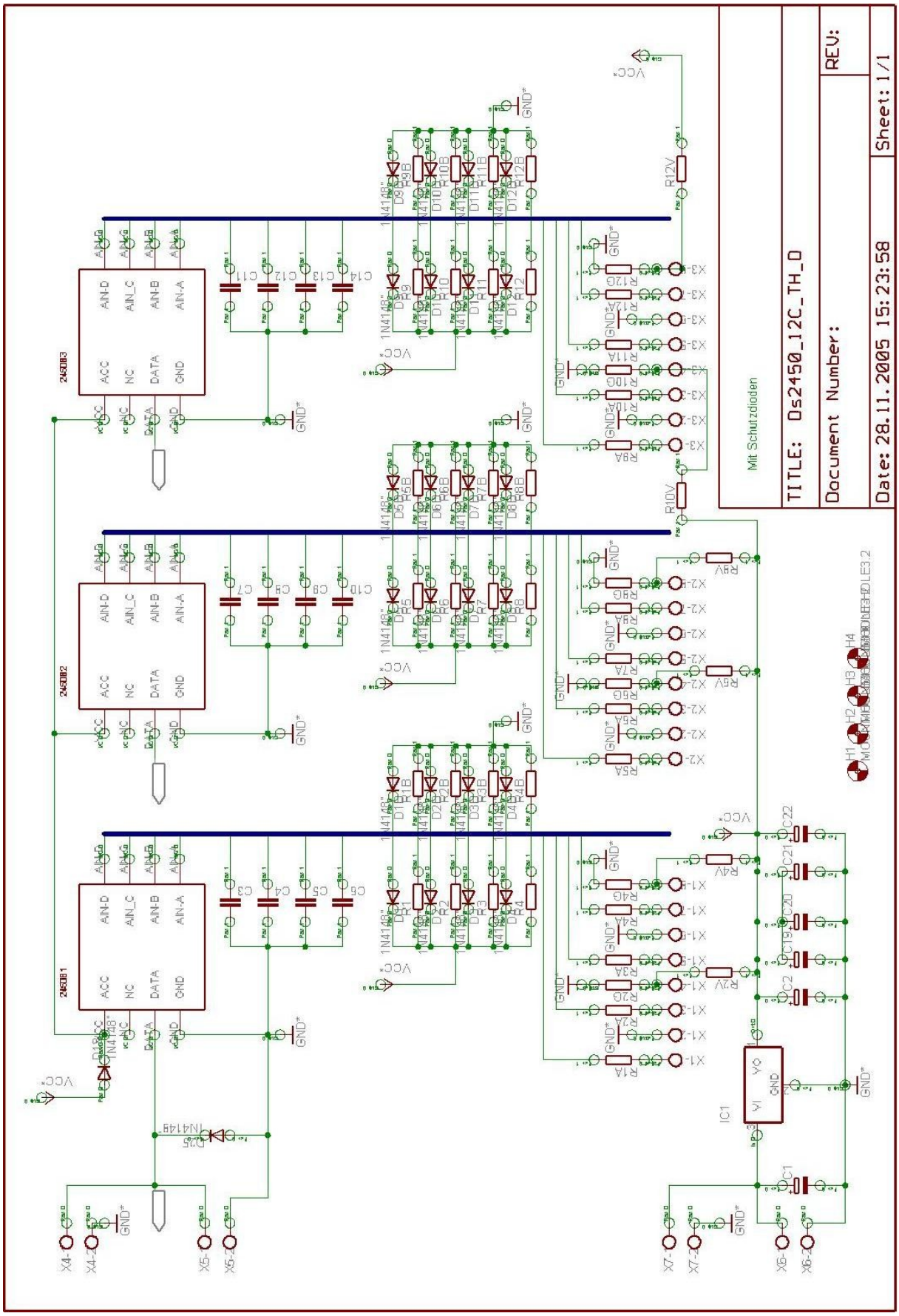


Abbildung 42: Schaltplan der Platine

7.1.3. Änderung der Platine

Um die Eingänge an unserer Platine zu vom Spannungspegel anzupassen müssen ein paar Brücken auf der Platine entfernt werden und durch ein paar Widerstände ersetzt werden.

Baustein-adresse	*Baustein-port*	*Eingang*	*Widerstands-Nr-*	*Alt*	*Neu*	*Belegung*
20.4AFE02000000	volt.A	1	R1A	0 Ω	10 kΩ ± 0,1%	Solar-spannung U_S
			R1B	---	10 kΩ ± 0,1%	
	volt.B	2	R2A	0 Ω	10 kΩ ± 0,1%	Solarstrom I_S
			R2G	---	---	
			R2B	---	10 kΩ ± 0,1%	
	volt.C	3	R3A	0 Ω	10 kΩ ± 0,1%	Verbrau-cher-spannung U_V
			R3B	---	10 kΩ ± 0,1%	
	volt.D	4	R4A	0 Ω	10 kΩ ± 0,1%	Verbrau-cherstrom I_V
			R4G	---	---	
			R4B	---	10 kΩ ± 0,1%	
			R4V	0 Ω	---	
20.4CF-B02000000	volt.A	5	R5A	0 Ω	---	Reserve
			R5B	---	---	
	volt.B	6	R6A	0 Ω	---	Reserve
			R6B	---	---	
			R6V	---	---	
	volt.C	7	R7A	0 Ω	---	Reserve
			R7B	---	---	
	volt.D	8	R8A	0 Ω	---	Reserve
			R8G	---	---	
			R8B	---	---	
			R8V	0 Ω	---	
20.2615030000000	volt.A	9	R9A	0 Ω	10 kΩ ± 0,1%	Akku-spannung U_A
			R9B	---	10 kΩ ± 0,1%	
	volt.B	10	R10A	0 Ω	10 kΩ ± 0,1%	Akkustrom I_A
			R10G	---	---	
			R10B	---	10 kΩ ± 0,1%	
			R10V	---	---	
	volt.C	11	R11A	0 Ω	10 kΩ ± 0,1%	Bestrah-lungsstärke E
			R11B	---	10 kΩ ± 0,1%	

Baustein-adresse	*Baustein-port*	*Eingang*	*Widerstands-Nr-*	*Alt*	*Neu*	*Belegung*
	volt.D	12	R12A	0 Ω	10 kΩ ± 0,1%	Außentem-peratur T_S
			R10G	---	---	
			R12B	---	10 kΩ ± 0,1%	
			R12V	0 Ω	---	

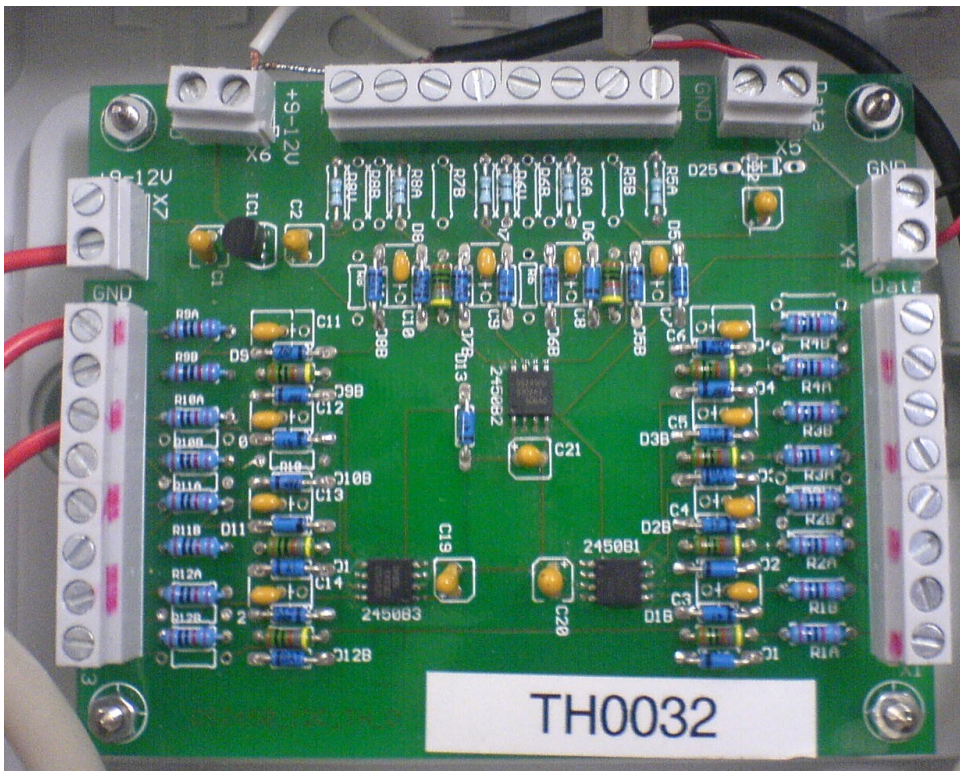


Abbildung 43: Platine mit Analogen Eingängen nach dem Umbau

7.1.4. Fehlerkorrektur

Eingangsspannung ist die Spannung die die am Eingang des AD-Wandlers anliegt. Angezeigte Spannung ist die Spannung die im System ausgegeben wird U_A. Errechnete Spannung U_E wird über die Parameter a & b errechnet.

$$a_{Kanal} = U_{Kanal,0V}$$

$$b_{Kanal} = \frac{U_{A,Kanal,8V} - U_{A,Kanal,0V}}{U_{E,Kanal,8V} - U_{E,Kanal,0V}}$$

$$U_{A,Kanal} = a + U_{E,Kanal} * b$$

Um im Skript die richtigen Werte anzuzeigen muss man die eingelesenen Werte wie folgt korrigieren:

$$U_{E,Kanal} = \frac{U_{A,Kanal} - a}{b}$$

In den Diagrammen stellt die blaue Linie die gemessenen Werte dar, und die violette über die oben genannten Formeln errechnet.

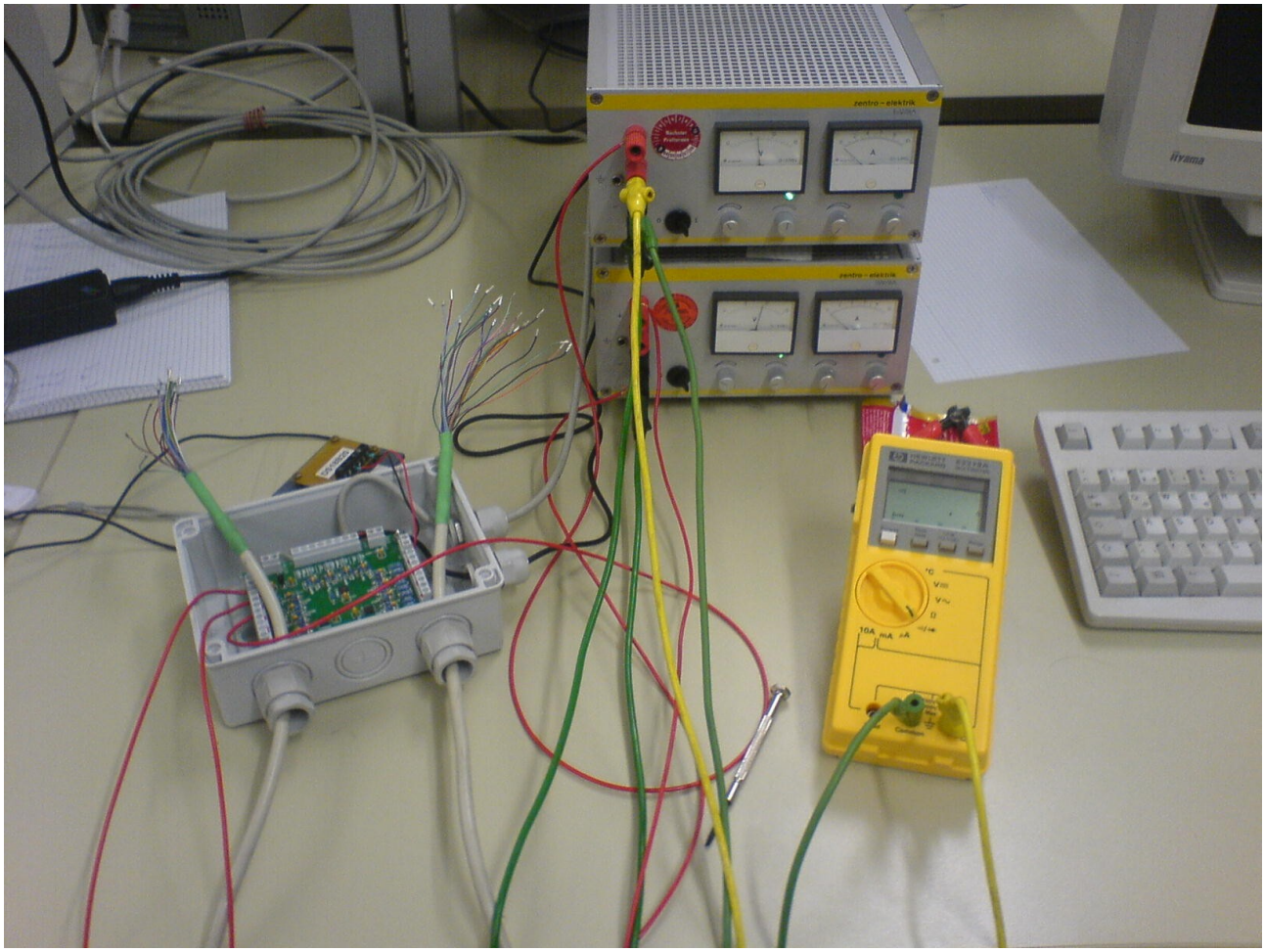


Abbildung 44: Messaufbau zur Bestimmung der Messabweichungen

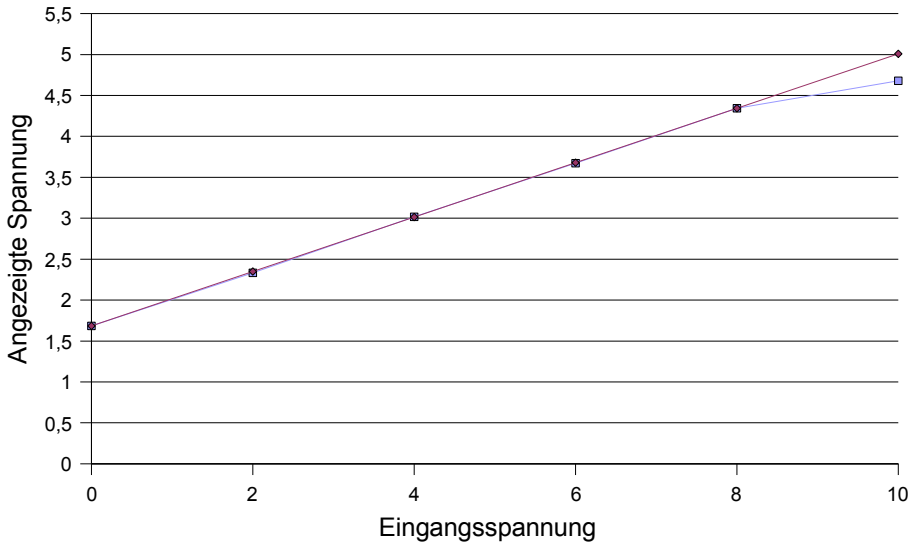
Kanal 1:

Eingang	Anzeige	Errechnet
0	1,68	1,68
2	2,33	2,35
4	3,02	3,01
6	3,67	3,68
8	4,34	4,34
10	4,68	5,01

Parameter:

$a_1 = 1{,}68424$

$b_1 = 0{,}332506$



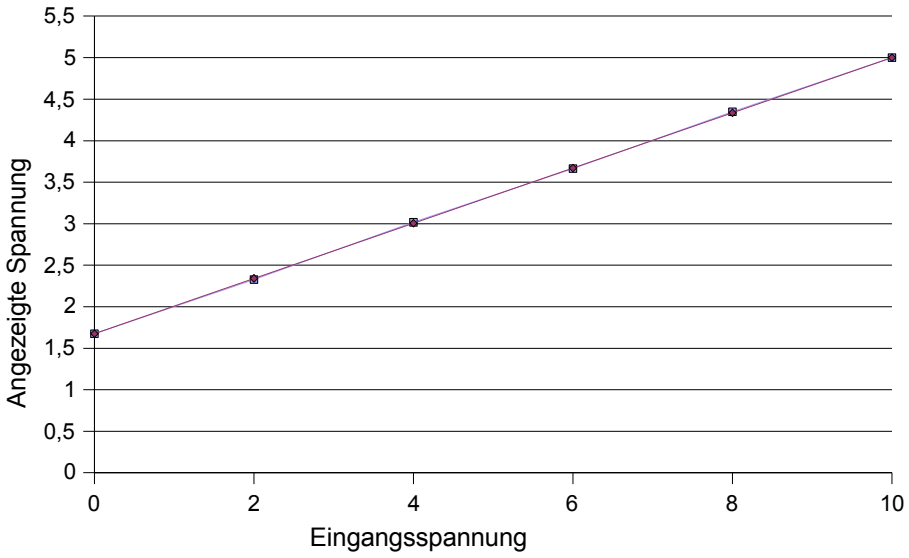
Kanal 2:

Eingang	Anzeige	Errechnet
0	1,68	1,68
2	2,33	2,34
4	3,02	3,01
6	3,66	3,67
8	4,35	4,34
10	5	5

Parameter:

$a_2 = 1{,}67635$

$b_2 = 0{,}332435$



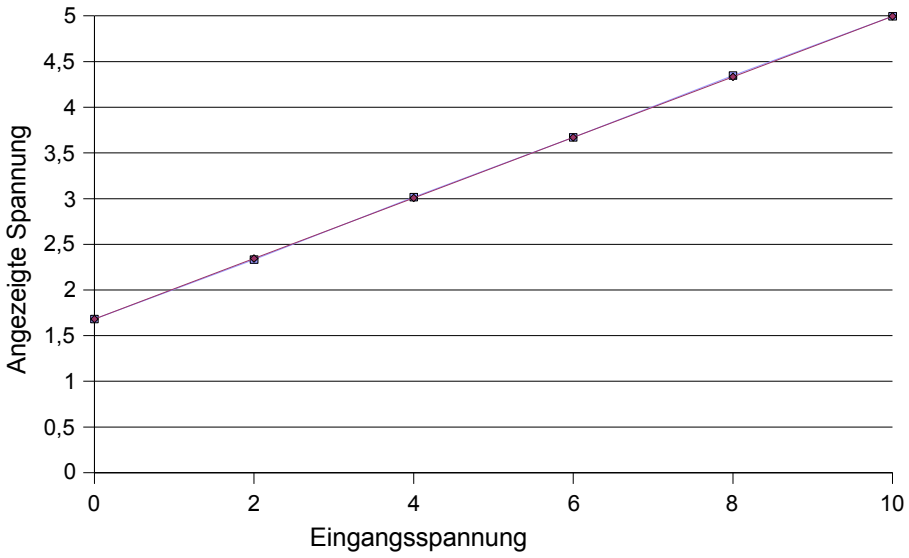
Kanal 3:

Eingang	Anzeige	Errechnet
0	1,68	1,68
2	2,33	2,34
4	3,02	3,01
6	3,67	3,67
8	4,35	4,33
10	5	5

Parameter:

$a_3 = 1{,}68151$

$b_3 = 0{,}331505$



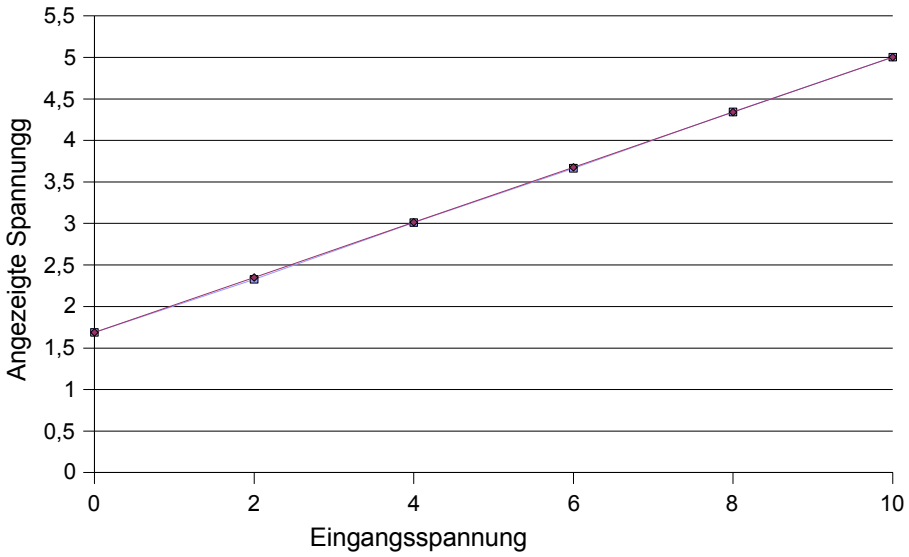
Kanal 4:

Eingang	Anzeige	Errechnet
0	1,69	1,69
2	2,33	2,35
4	3,01	3,01
6	3,66	3,68
8	4,34	4,34
10	5	5

Parameter:

$a_4 = 1{,}68745$

$b_4 = 0{,}331599$



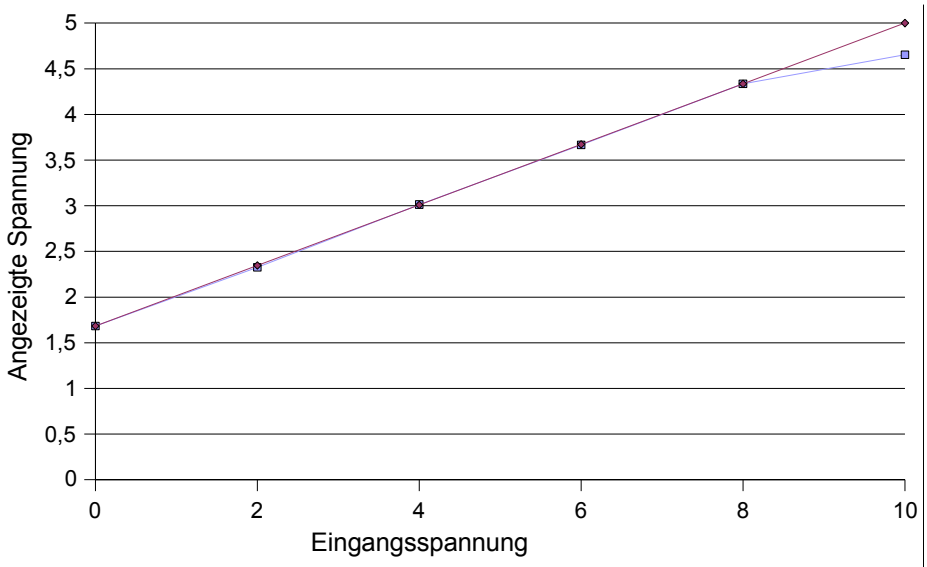
Kanal 9:

Eingang	Anzeige	Errechnet
0	1,68	1,68
2	2,33	2,35
4	3,01	3,01
6	3,67	3,67
8	4,34	4,34
10	4,65	5

Parameter:

$a_9 = 1{,}68354$

$b_9 = 0{,}331578$



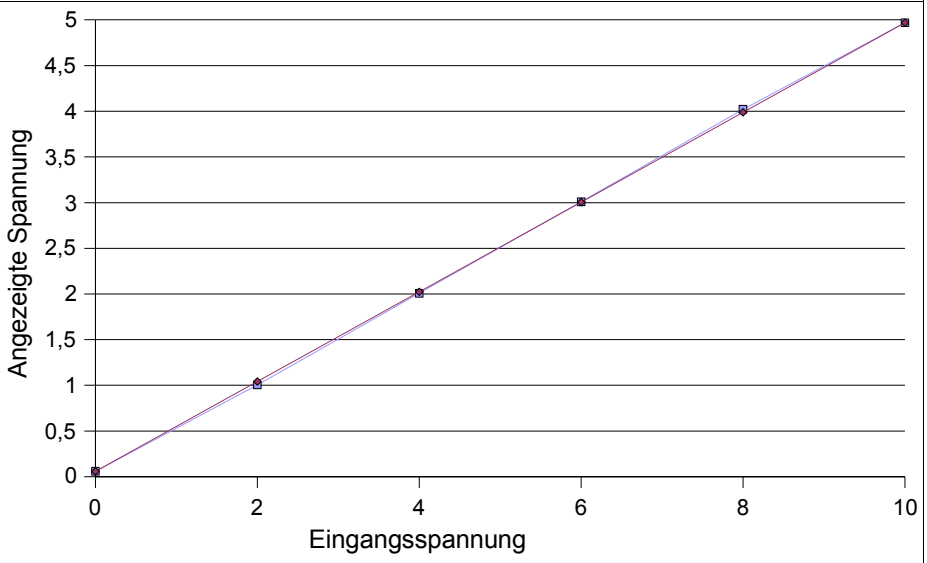
Kanal 10:

Eingang	Anzeige	Errechnet
0	0,06	0,06
2	1	1,04
4	2,01	2,02
6	3,01	3,01
8	4,02	3,99
10	4,97	4,97

Parameter:

$a_{10} = 0{,}0591415$

$b_{10} = 0{,}491133$



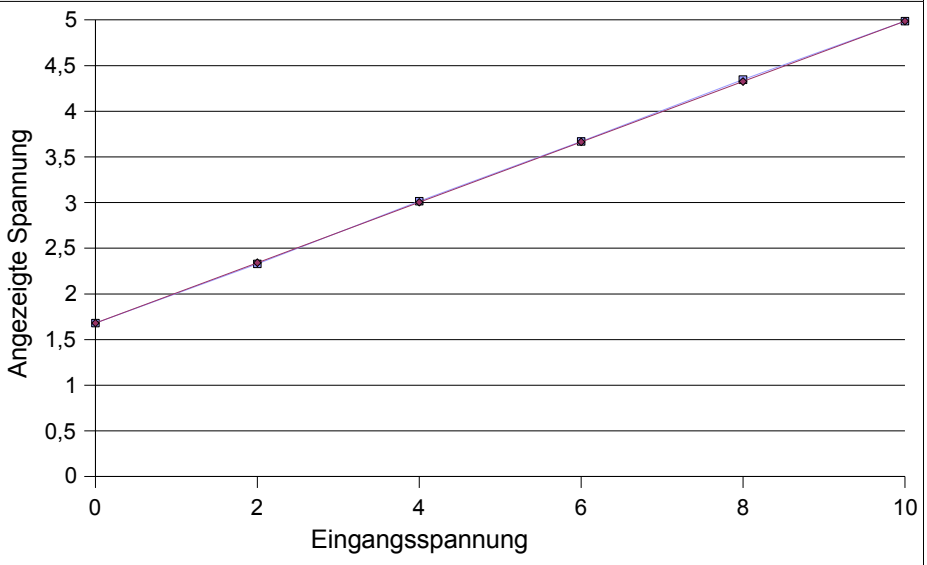
Kanal 11:

Eingang	Anzeige	Errechnet
0	1,68	1,68
2	2,33	2,34
4	3,01	3
6	3,67	3,66
8	4,35	4,33
10	4,99	4,99

Parameter:

$a_{11} = 1{,}68128$

$b_{11} = 0{,}330497$



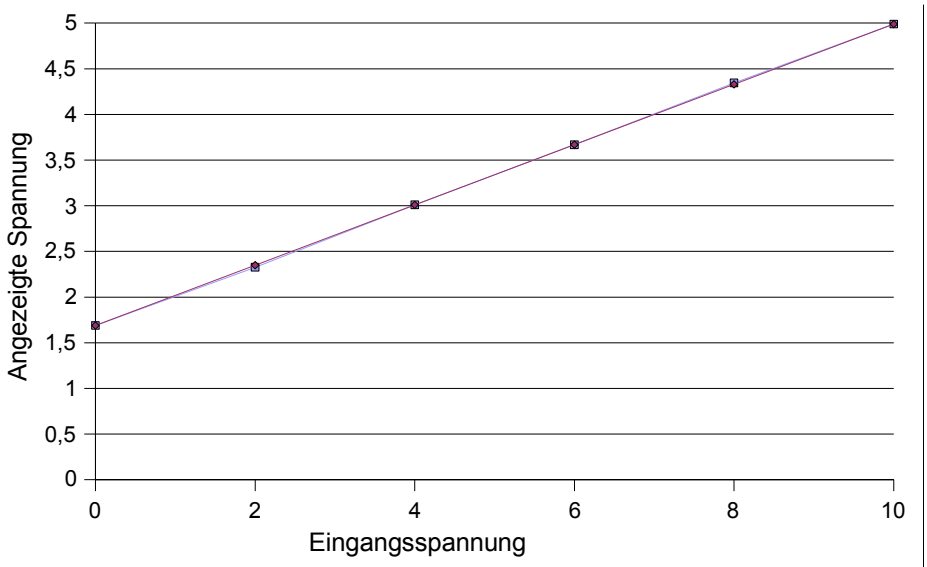
Kanal 12:

Eingang	Anzeige	Errechnet
0	1,69	1,69
2	2,33	2,35
4	3,01	3,01
6	3,67	3,67
8	4,34	4,33
10	4,99	4,99

Parameter:

$a_{12} = 1{,}68878$

$b_{12} = 0{,}330059$



7.1.5. Pinbelegung der 25 pol. Stecker

Da wir aus der Dokumentation der älteren Projekte, die Belegung des 25-poligen sub-D Steckers an dem SPS-Gestell oder des 36-poligen sub-D Steckers an den Messwandlern, nicht gefunden haben, mussten wir sie mit Hilfe der Pläne der Platinen herausmessen.

Pin	***Farbe***	***Belegung***	***Kanalnummer***
1.	schwarz-grau	Außentemperatur T_S	Kanal 12
2.	Braun grau	Bestrahlungsstärke E	Kanal 11
3.	Rot grau	Akkustrom I_A	Kanal 10
4.	hellblau	Verbraucherstrom I_V	Kanal 4
5.	orange / grau	Solarstrom I_S	Kanal 2
6.	hellblau sw	Solarspannung U_S	Kanal 1
7.	gelb schwarz	Verbraucherspannung U_V	Kanal 3
8	hellgrün	Akkuspannung U_A	Kanal 9
9	grün grau	Masse	
10	hellgrün-schwarz	Masse	
11	dunkelblau grau	Masse	
12	lila weiss	Reserve	
13	grau schwarz	Reserve	
14	schwarz	Reserve	
15	braun	Reserve	
16	rot	Reserve	
17	orange	Reserve	
18	gelb	Reserve	
19	dunkelgrün	Reserve	

Pin	*Farbe*	*Belegung*	*Kanalnummer*
20	blau	Reserve	
21	lila	Reserve	
22	grau	Reserve	
23	weiss	Reserve	
24	rosa	Reserve	
25	Rosa schwarz	Reserve	
Schirm	blank	Reserve	

7.2. Platine mit Temperatursensor

Mit dem Temperatursensor haben wir sie ersten Erfahrungen mit dem 1-Wire Bus gesammelt. Weiter haben sie benutzt zum herumexperimentieren und und um bei der Abschlusspräsentation ein paar Testprogramme zu präsentieren. Den Strom kann der Sensor über den 1-Wire Bus beziehen.

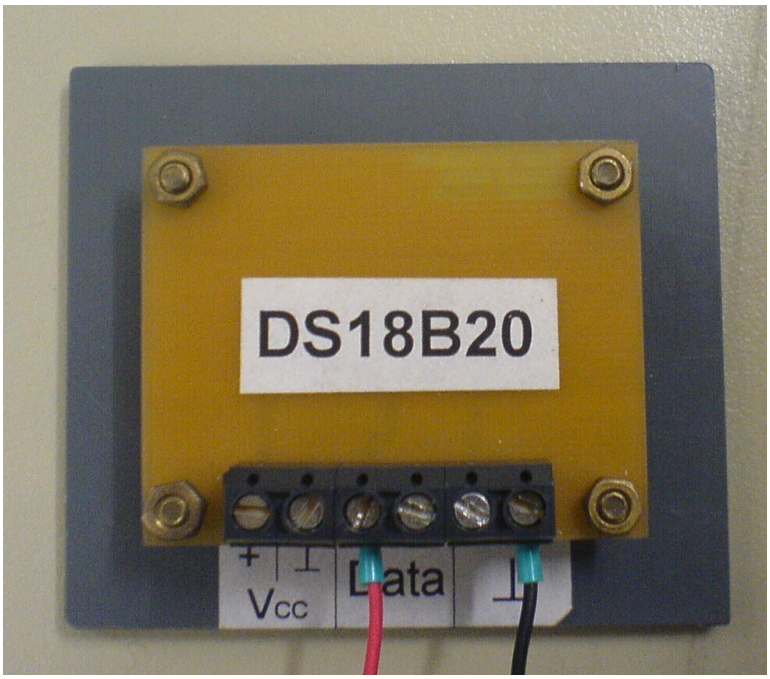


Abbildung 45: Temperatursensor

7.3. Platine mit digitalen Ausgängen

Diese Platine ist mit Digital Ausgängen bestückt. Wir haben sie benutzt, zum herumexperimentieren und und um bei der Abschlusspräsentation einen Schaltvorgang zu zeigen. Die Platine muss mit einer Spannung von 9 bis 12 V versorgt werden.

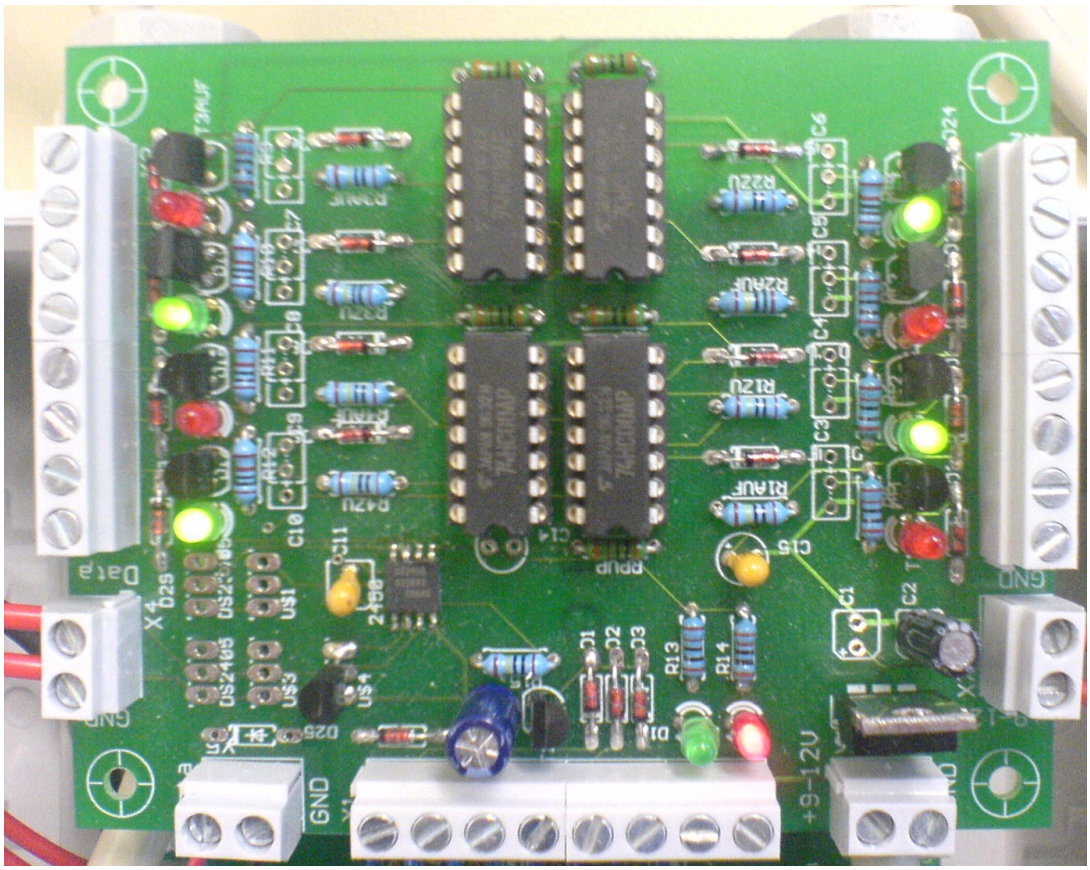
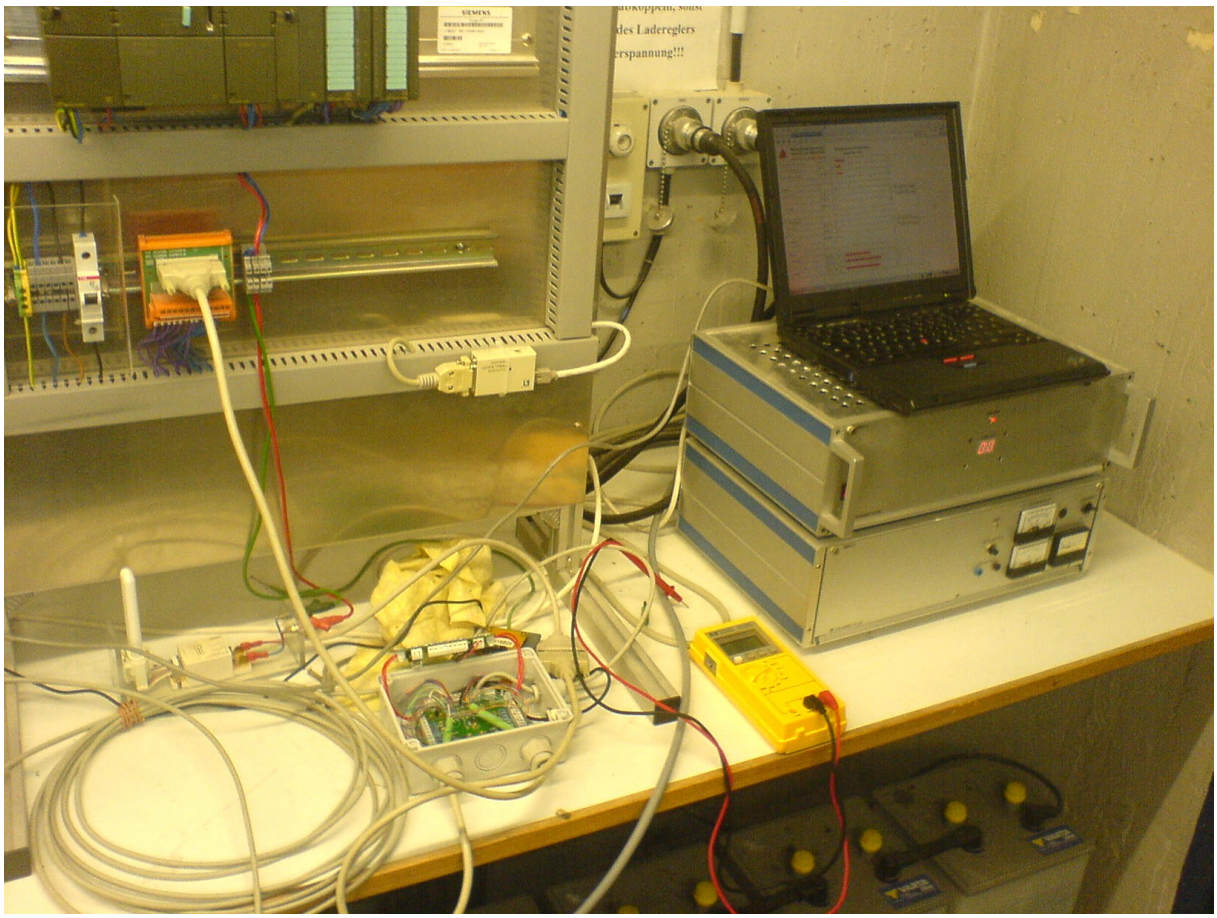
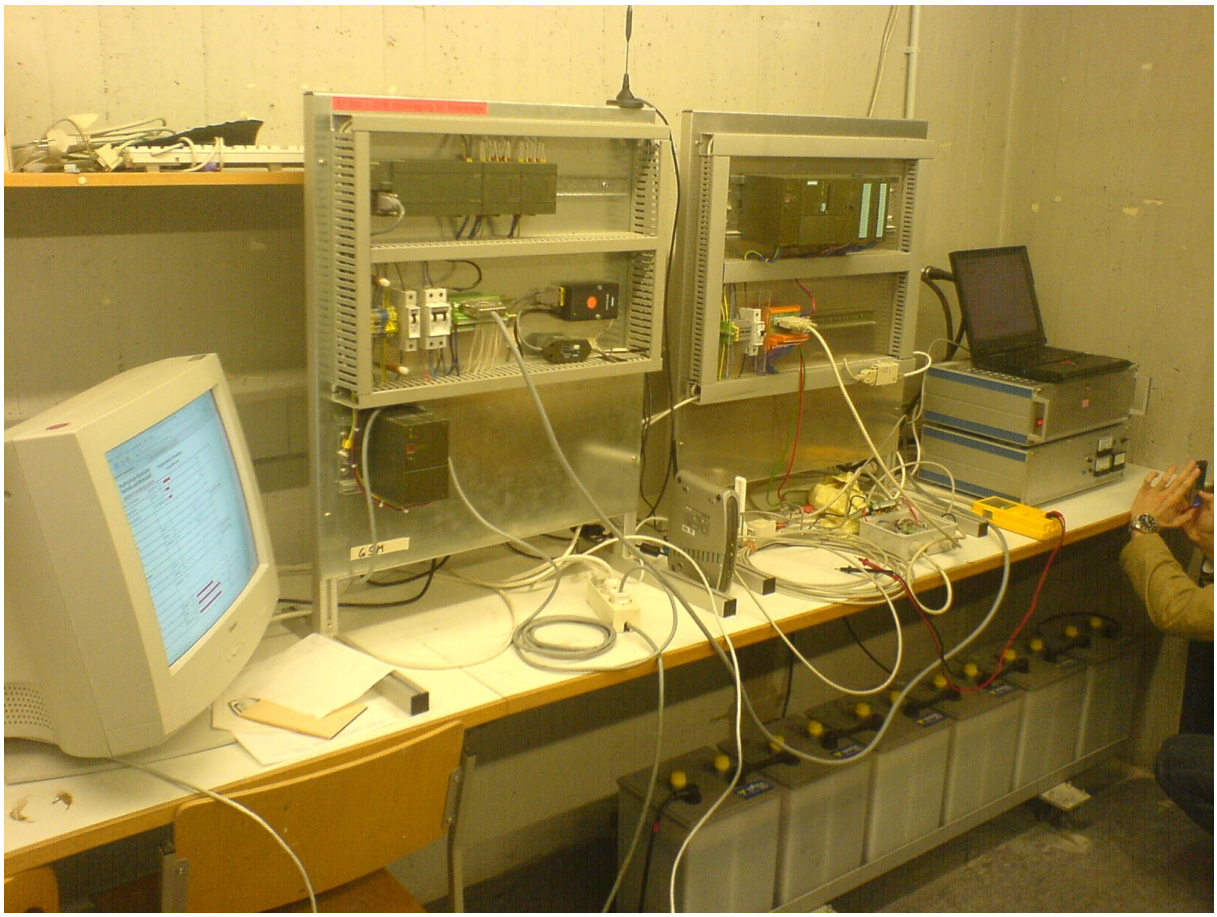


Abbildung 46: Digitale I/O Platine

7.4. Installation der Hardware an der Solaranlage

An der Solaranlage oben greifen wir unsere Signale für unsere Hardware zwischen die Messumformer und der SPS ab. Sie werden durchgeschleift, so dass sie SPS die Signale weiterhin verarbeiten kann. Die Platine bezieht ihre elektrische Energie über ein Steckernetzteil. Der 1-Wire Bus wird über eine geschirmte Telefonleitung mit Hilfe eines Adapter an die USB-Schnittstelle angeschlossen.





8. Software

8.1. Datenflussmodell

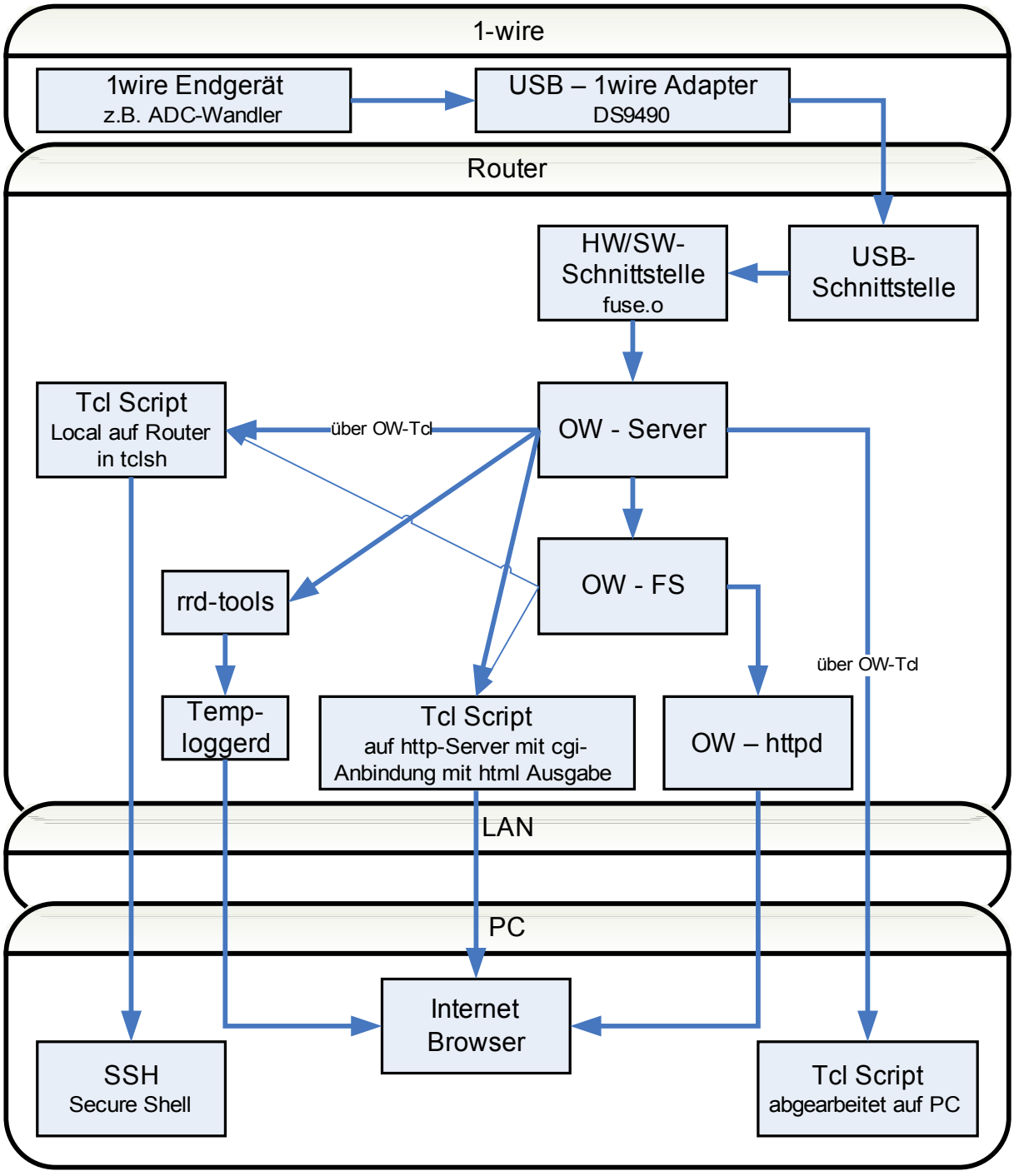


Abbildung 47: Datenflussmodell

8.2. Ansprechen der Sensoren Sensors

8.2.1. Mit Tcl vom OW-Server mit dem OW-Tcl-Modul

Befehlszeile auf dem Router	*Befehlszeile auf dem PC*	*Beschreibung*
`Package require ow`		Überprüft ob die Erweiterung OW-Tcl verfügbar ist
`OW::init u :<Port>`	`OW::init` <Router IP>`:<Port>`	Initialisierung des OW-Servers
`OW::get <Sensor ID>/<Variablen-Name>`		Wert abfragen
`OW::finish`		Bindung zu OW-Server schließen

8.2.2. Mit Tcl vom OW-Fs

Befehlszeile	*Beschreibung*
`set tmp [open „/tmp/1wire/<Sensor-ID>/<Variablen-Name>"]`	Öffnet die Datei und speichert in temporäre Variable tmp
`set data [read $tmp]`	Ließt die Variable ein und speichert sie als Zahl ab
`close $tmp`	Löscht temporäre Variable tmp

8.3. Skripte auf Router erzeugen dynamische HTML-Ausgabe

8.3.1. Allgemeine Hinweise

Bei dieser Methode arbeitet der Router das Skript ab und erzeugt eine dynamische HTML-Ausgabe. Die Daten holt sich das Programm vom OW-FS. Dies hat den Vorteil, dass man keinen Exklusivzugriff auf das ganze System besitzt. Weitere positive Effekt sind:

- Auf dem PC muss kein Tcl und OWFS-Erweiterung installiert haben
- Keine speziellen Erweiterungen nötig, da man die Daten mit einfachen Befehlen aus Dateien auslesen kann.

Ausgewählte Programme und zusätzliche Kommentare liegen im Router /stick/solar/cgi-solar. Diese Programme sind alle auf dem Router 2 getestet worden.

Wichtiger Hinweis: neue Programme benötigen die Dateiattribute: 755, müssen also nach dem Erstellen umgeändert werden (chmod)

8.3.2. Einbindung von CGI + TCL testen

Programm: test_1.cgi
Beschreibung: Testet die richtige Einbindung von CGI + TCL und deren Ausgabe

```
#!/usr/bin/tclsh
puts "testausgabe"
set a 4
set b 5
puts "a*b=[expr $a*$b]"
```

Abbildung 48: Browserausgabe con test_1.cgi

8.3.3. Auslesen der Temperatur und HTML Formatierung

Programm: test_2.cgi

Beschreibung: Testet das Auslesen der Temperatur, des Sensors: DS18B20 + HTML Formatierung

```
#!/usr/bin/tclsh
set tp [open "/tmp/1Wire/28.AACD35000000/temperature" r]
set data [read $tp]
close $tp
puts "Content-Type: text/html"
puts {
<HTML>
<HEAD>
<TITLE>Projekt HS Testbereich</TITLE>
```

```
</HEAD>

<BODY>
Aktuelle Temperatur:
}
puts "$data"
puts {
</BODY>
</HTML>
}
```

Abbildung 49: Browserausgabe test_2.cgi

8.3.4. Auslesen der Temperatur und als Balken ausgeben

Programm: test_3.cgi

Beschreibung: Auslesen der Temperatur, Festlegung einer Skalierung, grafische Ausgabe nach Skalierung

```
#!/usr/bin/tclsh

set skala 400            ;#Skalenteile
set temp_max 40          ;#max Temperatur

set tp [open "/tmp/1Wire/28.AACD35000000/temperature" r]
set data [read $tp]
close $tp

puts "Content-Type: text/html"
puts {
<HTML>
<HEAD>
<TITLE>Temperatur Anzeige</TITLE>
</HEAD>
<BODY>
}
puts {
<table width="600" border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td>
     Aktuelle Temperatur Kanal 1:}
puts "<b>$data </b>°C"
puts {
     <br>
    <br>
     <table width="400" border="0" cellspacing="0"
```

```
cellpadding="0">
        <tr>
        <td background="../../solar/anzeige_gruen.jpg">
          <img src="../../solar/anzeige_rot.jpg" width="}
puts "[expr $data*$skala/$temp_max]"
       puts {
            \" height="20"></td>
      </tr>
        <tr>
        <td>
       <table width="400" border="0" cellspacing="0"
cellpadding="0">
          <tr>
            <td>0&deg; C </td>
            <td><div align="right">40&deg; C </div></td>
          </tr>
        </table>
          </td>
</tr>
    </table></td>
  </tr>
</table>
</BODY>
</HTML>
}
```

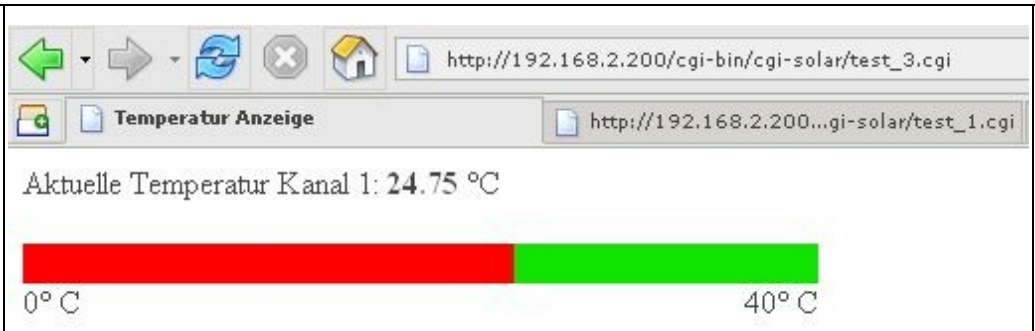


Abbildung 50: Browserausgabe der test_3.cgi

8.3.5. Auslesen der Temperatur und Ausgabe als vertikale Mehrfachanzeige

Programm: test_4.cgi

Beschreibung: Auslesen der Temperatur, Festlegung einer Skalierung, grafische Ausgabe nach Skalierung, vertikale Mehrfachanzeige)

```
#!/usr/bin/tclsh
set skala 300
set uc_max 40
set kanal_a [open "/tmp/1Wire/28.AACD35000000/temperature" r]
set data_a [read $kanal_a]
close $kanal_a
puts "Content-Type: text/html"
```

```
puts {
<HTML>
<HEAD>
<TITLE>Spannungs Anzeige</TITLE>
</HEAD>
<BODY>
}
puts {
<table width="600" border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td>
     Spannungsanzeige<br>
     Aktuelle Spannung Kanal A:}
puts "<b>$data_a </b> V"
puts {
     <br><br>
          <table width="400" border="0" cellspacing="0"
cellpadding="0">
      <tr>
        <td background="../../solar/anzeige_gruen.jpg">
          <img src="../../solar/anzeige_rot.jpg" width="}
puts "[expr $data_a*$skala/$uc_max]"
       puts {
            \" height="20"></td>
      </tr>
        <tr>
        <td>
       <table width="400" border="0" cellspacing="0"
cellpadding="0">
          <tr>
            <td>0&deg; C </td>
            <td><div align="right">40&deg; C </div></td>
          </tr>
        </table>
          </td>
</tr>
    </table></td>
  </tr>
</table><br>
       <table id="Skalen" width="600" border="0" cellpadding="0"
cellspacing="0" >
  <tr>
    <td>Kanal A
       <table  id="Kanal A" height="300" border="1"
cellpadding="0" cellspacing="0">
  <tr>
    <td width="15" valign="bottom"
background="../../solar/anzeige_gruen.jpg">
           <img src="../../solar/anzeige_rot.jpg" width="100%"
height="
```

```
            }
puts "[expr $data_a*$skala/$uc_max]"
       puts {
            255"></td>
    <td><table height="100%" border="0" cellpadding="0"
cellspacing="0">
      <tr>
        <td valign="top"><nobr>40 V </nobr></td>
      </tr>
      <tr>
        <td valign="bottom">0 V </td>
      </tr>
    </table></td>
  </tr>
</table>
          </td>
     <td>
          Kanal B
            <table  id="Kanal B" height="300" border="1"
cellpadding="0" cellspacing="0">
  <tr>
    <td width="15" valign="bottom"
background="../../solar/anzeige_gruen.jpg">
          <img src="../../solar/anzeige_rot.jpg" width="100%"
height="
            }
puts "[expr $data_a*$skala/$uc_max]"
       puts {
            255"></td>
    <td><table height="100%" border="0" cellpadding="0"
cellspacing="0">
      <tr>
        <td valign="top"><nobr>40 V </nobr></td>
      </tr>
      <tr>
        <td valign="bottom">0 V </td>
      </tr>
    </table></td>
  </tr>
</table>
     </td>
     <td>
          Kanal C
            <table  id="Kanal C" height="300" border="1"
cellpadding="0" cellspacing="0">
  <tr>
    <td width="15" valign="bottom"
background="../../solar/anzeige_gruen.jpg">
          <img src="../../solar/anzeige_rot.jpg" width="100%"
height="
```

```
            }
puts "[expr $data_a*$skala/$uc_max]"
       puts {
            255"></td>
    <td><table height="100%" border="0" cellpadding="0"
cellspacing="0">
      <tr>
        <td valign="top"><nobr>40 V </nobr></td>
      </tr>
      <tr>
        <td valign="bottom">0 V </td>
      </tr>
    </table></td>
  </tr>
</table>
     </td>
    <td>
          Kanal D
            <table  id="Kanal D" height="300" border="1"
cellpadding="0" cellspacing="0">
  <tr>
    <td width="15" valign="bottom"
background="../../solar/anzeige_gruen.jpg">
          <img src="../../solar/anzeige_rot.jpg" width="100%"
height="
            }
puts "[expr $data_a*$skala/$uc_max]"
       puts {
            255"></td>
    <td><table height="100%" border="0" cellpadding="0"
cellspacing="0">
      <tr>
        <td valign="top"><nobr>40 V </nobr></td>
      </tr>
      <tr>
        <td valign="bottom">0 V </td>
      </tr>
    </table></td>
  </tr>
</table>
          </td>
  </tr>
</table>
</BODY>
</HTML>
```

Abbildung 51: Browserausgabe der test_4.cgi

8.3.6. Auslesen der Temperatur, vergleicht die Temperatur mit einem Grenzwert

Programm: test_6.cgi

Beschreibung: Testet das Auslesen der Temperatur, vergleicht die Temperatur mit einem vorgegebenen Maximalwert und gibt aus ob größer oder kleiner (If-Abfrage)

```
#!/usr/bin/tclsh
set tp [open "/tmp/1Wire/28.AACD35000000/temperature" r]
set data [read $tp]
close $tp

set tempwert 23
puts "Content-Type: text/html"
puts {
<HTML>
<HEAD>
<TITLE>Projekt HS Testbereich</TITLE>
</HEAD>
```

```
<BODY>
Aktuelle Temperatur:
}
puts "<br>$data"
puts "<br>Begrenzungswert: <b>$tempwert</b>"
puts "<br>Die Temperatur ist<br><b>"
if {$tempwert<$data} then {
     puts "größer"
     } else {
     puts "kleiner"
     }
puts "</b><br>als der Begrenzungswert"
puts {
</BODY>
</HTML>
}
```

Abbildung 52: Browserausgabe der test_6.cgi (I)

Abbildung 53: Browserausgabe der test_6.cgi (I)

8.3.7. Auslesen der Temperatur und Ausgabe einer blinkenden Warnung bei Grenzwertüberschreitung.

Programm: test_7.cgi

Beschreibung: Auslesen der Temperatur, Darstellung mit skalierten Balken und Ausgabe eines blinkenden Warnungs Zeichens bei Grenzwertüberschreitung.

```
#!/usr/bin/tclsh
set tp [open "/tmp/1Wire/28.AACD35000000/temperature" r]
set data [read $tp]
close $tp
set tempwert 23
set skala 300           ;#Skalenteile
set temp_max 40         ;#max Temperatur

puts "Content-Type: text/html"
puts {
<HTML>
<HEAD>
<TITLE>Projekt HS Testbereich</TITLE>
</HEAD>
```

```
<BODY>
Aktuelle Temperatur:
}
puts "<b>$data</b>°C"
puts "<br>Begrenzungswert: <b>$tempwert</b>°C"
puts {
     <table width="400" border="0" cellspacing="0"
cellpadding="0">
        <tr>
        <td><img src="../../solar/anzeige_rot.jpg" width="}
puts "[expr $data*$skala/$temp_max]"
       puts {
            \" height="20"></td>
      <td>}
if {$tempwert<$data} then {
     puts "<blink><b> WARNUNG </b></blink>"
     }
puts {</td></tr>
            </table>
</BODY>
</HTML>
}
```

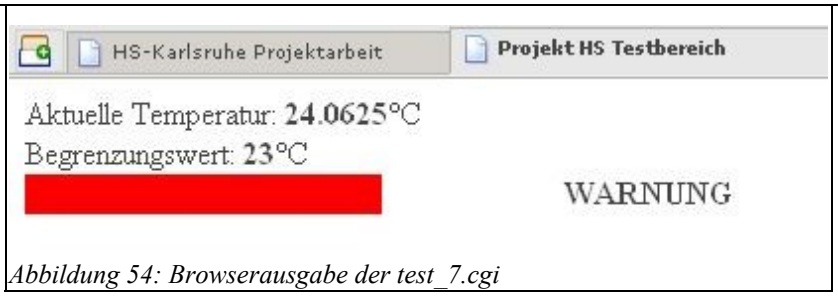


Abbildung 54: Browserausgabe der test_7.cgi

8.3.8. Liest Daten der Solaranlage aus und gibt diese aus

Programm: ausgabe_1.cgi

Beschreibung: Liest die Daten der Solaranlage aus, Wiedergabe

```
#!/usr/bin/tclsh

# nur lokaler Temp Sensor zu Testzwecken
#kanal_temp Temperatursensor: DS18B20
set kanal_temp [open "/tmp/1Wire/28.AACD35000000/temperature" r]
set data_temp [read $kanal_temp]
close $kanal_temp


#kanal_a Aussentemperatur Eingang 12
set kanal_a [open "/tmp/1Wire/20.261503000000/volt.D" r]
set data_a [read $kanal_a]
close $kanal_a

#kanal_b Beleuchtungsstärke Eingang 11
```

```
set kanal_b [open "/tmp/1Wire/20.261503000000/volt.C" r]
set data_b [read $kanal_b]
close $kanal_b

#kanal_c Zellen Spannung Eingang 1
set kanal_c [open "/tmp/1Wire/20.4AFE02000000/volt.A" r]
set data_c [read $kanal_c]
close $kanal_c

#kanal_d Zellen Strom Eingang 2
set kanal_d [open "/tmp/1Wire/20.4AFE02000000/volt.B" r]
set data_d [read $kanal_d]
close $kanal_d

#kanal_e Zellen Leistung
set data_e [expr $data_c*$data_d]

#kanal_f Verbraucher Spannung Eingang 3
set kanal_f [open "/tmp/1Wire/20.4AFE02000000/volt.C" r]
set data_f [read $kanal_f]
close $kanal_f

#kanal_g Verbraucher Strom Eingang 4
set kanal_g [open "/tmp/1Wire/20.4AFE02000000/volt.D" r]
set data_g [read $kanal_g]
close $kanal_g

#kanal_h Verbraucher Leistung
set data_h [expr $data_f*$data_g]

#kanal_i Akku Spannung Eingang 9
set kanal_i [open "/tmp/1Wire/20.261503000000/volt.A" r]
set data_i [read $kanal_i]
close $kanal_i

#kanal_j Akku Strom Eingang 10
set kanal_j [open "/tmp/1Wire/20.261503000000/volt.B" r]
set data_j [read $kanal_j]
close $kanal_j

#kanal_h Akku Leistung
set data_k [expr $data_i*$data_j]

puts "Content-Type: text/html"
puts {

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
```

```
<meta http-equiv="Content-Type" content="text/html; charset=iso-
8859-1">
<title>HS-Karlsruhe Projektarbeit</title>
</head>

<body>
 <font size="+1"><strong><img src="/solar/logo_fhkarlsruhe.gif"
width="270" height="60"><br>
 <br>
 Projektarbeit: Solaranlage<br>
</strong></font><strong><br>
Anlagedaten:</strong>
<table width="600" border="0" cellspacing="3" cellpadding="0">
  <tr>
    <td width="150"><div align="left">Temperatur Testplatine:
</div></td>
    <td>
     }
puts "<b>$data_temp &deg;C</b>"
puts {

    <!--  XX_aussentemp --></td>
    <td> </td>
  </tr>
<tr>
    <td width="150"><div align="left">Aussentemperatur:
</div></td>
    <td>
     }
puts "<b>$data_a &deg;C</b>"
puts {

    <!--  XX_aussentemp --></td>
    <td> </td>
  </tr>
          <tr>
    <td width="150">Beleuchtungsst&auml;rke:
    </td>
    <td >
          }
puts "<b>$data_b lx</b>"
puts {
          <!--  XX_beleuchtstaerke --></td>
    <td> </td>
  </tr>
  <tr>
    <td> </td>
    <td> </td>
    <td> </td>
  </tr>
```

```
  <tr>
    <td><strong>Zellen</strong></td>
    <td> </td>
    <td> </td>
  </tr>
  <tr>
    <td>Spannung:</td>
    <td><!--  XX_spannug_zellen -->
          }
puts "<b>$data_c V</b>"
puts {
        </td>
    <td> </td>
  </tr>
  <tr>
    <td>Strom: </td>
    <td>
}
puts "<b>$data_d A</b>"
puts {
     <!--  XX_strom_zellen --></td>
    <td> </td>
  </tr>
  <tr>
    <td>Leistung:</td>
    <td><!--  XX_leistung_zellen -->
         }
puts "<b>$data_e W</b>"
puts {
</td>
    <td> </td>
  </tr>
  <tr>
    <td> </td>
    <td> </td>
    <td> </td>
  </tr>
  <tr>
    <td><strong>Verbraucher</strong></td>
    <td> </td>
    <td> </td>
  </tr>
  <tr>
    <td>Spannung:</td>
    <td><!--  XX_spannung_verbraucher -->
         }
         puts "<b>$data_f V</b>"
puts {
     </td>
    <td> </td>
```

```
  </tr>
  <tr>
    <td>Strom:</td>
    <td><!--  XX_strom_verbraucher -->
          }
          puts "<b>$data_g A</b>"
puts {
     </td>
    <td> </td>
  </tr>
  <tr>
    <td>Leistung:</td>
    <td><!--  XX_leistung_verbraucher -->
                  }
          puts "<b>$data_h W</b>"
puts {
     </td>
    <td> </td>
  </tr>
  <tr>
    <td> </td>
    <td> </td>
    <td> </td>
  </tr>
  <tr>
    <td><strong>Akku</strong></td>
    <td> </td>
    <td> </td>
  </tr>
  <tr>
    <td>Spannung:</td>
    <td><!--  XX_spannung_akku -->
          }
          puts "<b>$data_i V</b>"
puts {
     </td>
    <td> </td>
  </tr>
  <tr>
    <td>Strom:</td>
    <td><!--  XX_strom_akku -->
          }
          puts "<b>$data_j A</b>"
puts {
     </td>
    <td> </td>
  </tr>
  <tr>
    <td>Leistung:</td>
    <td><!--  XX_leistung_akku -->
```

```
          }
          puts "<b>$data_k W</b>"
puts {
     </td>
    <td> </td>
  </tr>
</table>
</body>
</html>
}
```

Abbildung 55: Browserausgabe der ausgabe_1.cgi

8.3.9. Korrektur der eingelesenen Daten

Programm: ausgabe_2.cgi
Beschreibung: Liest alle Daten aus, korrigiert diese mit Berechnungsfaktor und gibt diese aus. Ermittlung des Faktors siehe Dokumentation Hardware Bereich.

```
#!/usr/bin/tclsh

set skala 300
set uc_max 40
```

```
# nur lokaler Temp Sensor zu Testzwecken
#kanal_temp Temperatursensor: DS18B20
set kanal_temp [open "/tmp/1Wire/28.AACD35000000/temperature" r]
set data_temp [read $kanal_temp]
close $kanal_temp

#kanal_a Aussentemperatur Eingang 12
set kanal_a [open "/tmp/1Wire/20.261503000000/volt.D" r]
set data_a1 [read $kanal_a]
set data_a [expr ($data_a1-1.6888)/0.3301]
close $kanal_a

#kanal_b Beleuchtungsstärke Eingang 11
set kanal_b [open "/tmp/1Wire/20.261503000000/volt.C" r]
set data_b1 [read $kanal_b]
set data_b [expr ($data_b1-1.68128)/0.33050]
close $kanal_b

#kanal_c Zellen Spannung Eingang 1
set kanal_c [open "/tmp/1Wire/20.4AFE02000000/volt.A" r]
set data_c1 [read $kanal_c]
set data_c [expr ($data_c1-1.68424)/0.33251]
close $kanal_c

#kanal_d Zellen Strom Eingang 2
set kanal_d [open "/tmp/1Wire/20.4AFE02000000/volt.B" r]
set data_d1 [read $kanal_d]
set data_d [expr ($data_d1-1.67635)/0.33244]
close $kanal_d

#kanal_e Zellen Leistung
set data_e [expr $data_c*$data_d]

#kanal_f Verbraucher Spannung Eingang 3
set kanal_f [open "/tmp/1Wire/20.4AFE02000000/volt.C" r]
set data_f1 [read $kanal_f]
set data_f [expr ($data_f1-1.68151)/0.33151]
close $kanal_f

#kanal_g Verbraucher Strom Eingang 4
set kanal_g [open "/tmp/1Wire/20.4AFE02000000/volt.D" r]
set data_g1 [read $kanal_g]
set data_g [expr ($data_g1-1.68745)/0.33160]
close $kanal_g

#kanal_h Verbraucher Leistung
set data_h [expr $data_f*$data_g]

#kanal_i Akku Spannung Eingang 9
set kanal_i [open "/tmp/1Wire/20.261503000000/volt.A" r]
```

```
set data_i1 [read $kanal_i]
set data_i [expr ($data_i1-1.68354)/0.33158]
close $kanal_i

#kanal_j Akku Strom Eingang 10
set kanal_j [open "/tmp/1Wire/20.261503000000/volt.B" r]
set data_j1 [read $kanal_j]
set data_j [expr ($data_j1-0.0591415)/0.491133]
close $kanal_j

#kanal_h Akku Leistung
set data_k [expr $data_i*$data_j]
-----weiter siehe ausgabe_1-----
```

8.3.10. Datenabfrage Solaranlage (endgültige Version)

Programm: ausgabe_4.cgi

Beschreibung: Alle Parameter werden skaliert und korrigiert. Blinkende Warnmeldung bei Grenzwertunter- bzw. überschreitung. Ausgabe Datei bei Projektabgabe

```
#!/usr/bin/tclsh

### Festlegung aller Grenz / Maximalwerte ###
#für Anzeigenskallierung
set skala 300;           #Skalenteile
set temp_max 90;         #max Temperatur (lokaler Sensor)
set temps_max 80;        #max Temperatur Solarzelle
set max_e 1400;               #max Bestrahlungsstärke
set max_U_s 50;               #max Solarspannung
set max_I_s 5;           #max Solarstrom
set max_W_s 200;         #max Solarleistung
set max_U_v 50;               #max Verbraucherspannung
set max_I_v 5;           #max Verbraucherstrom
set max_W_v 200;         #max Verbraucherleistung
set max_U_a 50;               #max Akkuspannung
set max_I_a 7;           #max Akkustrom
set max_W_a 200;         #max Akkuleistung

#Grenzwerte für Alarmmeldungen
set grenz_temp 23;       #Grenzwert Temperatur
set grenz_U_s_min 0;     #Grenzwert min Solarspannung
set grenz_U_s_max 44;    #Grenzwert max Solarspannung
set grenz_U_v_min 0;     #Grenzwert min Verbraucherpannung
set grenz_U_v_max 29;    #Grenzwert max Verbraucherpannung
set grenz_U_a_min 20;    #Grenzwert min Akkupannung
set grenz_U_a_max 29;    #Grenzwert max Akkupannung
set grenz_I_s_min 0;     #Grenzwert min Solarstrom
set grenz_I_s_max 3.4;   #Grenzwert max Solarstrom
set grenz_I_v_min 0;     #Grenzwert min Verbraucherstrom
set grenz_I_v_max 2.5;   #Grenzwert max Verbraucherstrom
set grenz_I_a_min -3.4;  #Grenzwert min Akkustrom
```

```
set grenz_I_a_max 2.5;   #Grenzwert max Akkutrom
set grenz_e    1200;          #Grenzwert Bestrahlungsstärke

#nur lokaler Temp Sensor zu Testzwecken
#kanal_temp Temperatursensor: DS18B20
set kanal_temp [open "/tmp/1Wire/28.AACD35000000/temperature" r]
set data_temp [read $kanal_temp]
close $kanal_temp

#kanal_a Aussentemperatur Eingang 12
set kanal_a [open "/tmp/1Wire/20.261503000000/volt.D" r]
set data_a1 [read $kanal_a]
set data_a [expr ($data_a1-1.6888)/0.3301]
# set data_a [format %8.3f $data_a2]             # Formatierung (8
Stellen, 3 Nachkomma)
close $kanal_a

#kanal_b Beleuchtungsstärke Eingang 11
set kanal_b [open "/tmp/1Wire/20.261503000000/volt.C" r]
set data_b1 [read $kanal_b]
set data_b [expr (($data_b1-1.68128)/0.33050)*680]
close $kanal_b

#kanal_c Zellen Spannung Eingang 1
set kanal_c [open "/tmp/1Wire/20.4AFE02000000/volt.A" r]
set data_c1 [read $kanal_c]
set data_c [expr (($data_c1-1.68424)/0.33251)*5]
close $kanal_c

#kanal_d Zellen Strom Eingang 2
set kanal_d [open "/tmp/1Wire/20.4AFE02000000/volt.B" r]
set data_d1 [read $kanal_d]
set data_d [expr (($data_d1-1.67635)/0.33244)/2.7]
close $kanal_d

#kanal_e Zellen Leistung
set data_e [expr $data_c*$data_d]

#kanal_f Verbraucher Spannung Eingang 3
set kanal_f [open "/tmp/1Wire/20.4AFE02000000/volt.C" r]
set data_f1 [read $kanal_f]
set data_f [expr (($data_f1-1.68151)/0.33151)*3.3]
close $kanal_f

#kanal_g Verbraucher Strom Eingang 4
set kanal_g [open "/tmp/1Wire/20.4AFE02000000/volt.D" r]
set data_g1 [read $kanal_g]
set data_g [expr (($data_g1-1.68745)/0.33160)/3.9]
close $kanal_g
```

```
#kanal_h Verbraucher Leistung
set data_h [expr $data_f*$data_g]

#kanal_i Akku Spannung Eingang 9
set kanal_i [open "/tmp/1Wire/20.261503000000/volt.A" r]
set data_i1 [read $kanal_i]
set data_i [expr ($data_i1-1.68354)/0.33158*3.3]
close $kanal_i

#kanal_j Akku Strom Eingang 10
set kanal_j [open "/tmp/1Wire/20.261503000000/volt.B" r]
set data_j1 [read $kanal_j]
set data_j [expr ($data_j1-0.0591415)/0.491133-3.25]
close $kanal_j

#kanal_h Akku Leistung
set data_k [expr $data_i*$data_j]

puts "Content-Type: text/html"
puts {

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-
8859-1">
<meta http-equiv="refresh" content="5;">
<title>HS-Karlsruhe Projektarbeit</title>
</head>

<body>
<table border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td valign="middle"><img src="/solar/logo_fhkarlsruhe.gif"
width="270" height="60"></td>
     <td width="300" align="center" valign="middle"><font
size="+1"><strong>Projektarbeit: Solaranlage
     <br></font>Anlagenübersicht</strong>
     </td>
  </tr>
</table>
<table border="1" bordercolor="#000000" cellspacing="0"
cellpadding="3">
  <tr>
    <td width="150"><div align="left">Temperatur Testplatine:
</div></td>
    <td><!--  XX_aussentemp -->
     }
puts "<b>"
```

```
puts [format %4.2f $data_temp]
puts " °C</b>"
puts {
    </td>
    <td>
     <table width="360" border="0" cellspacing="0"
cellpadding="0">
      <tr>
        <td width="30"><font size="-1">0°C</font></td>
        <td><img src="../../solar/anzeige_rot.jpg" width="}
puts "[expr $data_temp*$skala/$temp_max]"
       puts {
            \" height="10"></td>
        <td width="30"><font size="-1">90°C</font></td>
      </tr>
    </table>
          </td>
     <td width="130">
          }
if {$grenz_temp<$data_temp} then {
     puts "<blink><b> WARNUNG </b></blink>"
     }
puts {
          </td>
  </tr>
<tr>
    <td width="200"><div align="left">Aussentemperatur:
</div></td>
    <td>
     }
puts "<b>"
puts [format %8.1f $data_a]
puts "&deg;C</b>"
puts {

    <!--  XX_aussentemp --></td>
    <td>

          <table width="360" border="0" cellspacing="0"
cellpadding="0">
      <tr>
        <td width="30"><font size="-1">-20°C</font></td>
        <td><img src="../../solar/anzeige_rot.jpg" width="}
puts "[expr ($data_a*$skala/$temps_max)+20]"
       puts {
            \" height="10"></td>
        <td width="30"><font size="-1">60°C</font>
              </td>
      </tr>
    </table>
```

```
    </td>
  </tr>
          <tr>
    <td width="150">Beleuchtungsst&auml;rke:
    </td>
    <td >
          }
puts "<b>"
puts [format %8.0f $data_b]
puts " W/m²</b>"
puts {
          <!--  XX_beleuchtstaerke --></td>
    <td>
                    <table width="360" border="0" cellspacing="0"
cellpadding="0">
      <tr>
        <td width="35"><font size="-2">0 W/m²</font></td>
        <td><img src="../../solar/anzeige_rot.jpg" width="}
puts "[expr $data_b*$skala/$max_e]"
       puts {
            \" height="10"></td>
        <td width="35"><font size="-2">1400 W/m²</font>
              </td>
      </tr>
    </table>
          </td>
                    <td width="100">
          }
if {$grenz_e<$data_b} then {
     puts "<blink><b> WARNUNG </b></blink>"
     }
puts {
          </td>
  </tr>
  <tr>
    <td> </td>
    <td> </td>
    <td> </td>
  </tr>
  <tr>
    <td><strong>Zellen</strong></td>
    <td> </td>
    <td> </td>
  </tr>
  <tr>
    <td>Spannung:</td>
    <td><!--  XX_spannug_zellen -->
           }
puts "<b>"
```

```
puts [format %3.2f $data_c]
puts " V</b>"
puts {
         </td>
    <td>
     <table width="360" border="0" cellspacing="0"
cellpadding="0">
      <tr>
        <td width="30"><font size="-1">0 V</font></td>
        <td><img src="../../solar/anzeige_rot.jpg" width="}
puts "[expr $data_c*$skala/$max_U_s+1]"
       puts {
            \" height="10"></td>
        <td width="30"><font size="-1">50 V</font>
              </td>
      </tr>
    </table>
              </td>
                   <td>
          }
if {$grenz_U_s_min>$data_c} then {
     puts "<blink><b> WARNUNG (min)</b></blink>"
     }
     if {$grenz_U_s_max<$data_c} then {
     puts "<blink><b> WARNUNG (max)</b></blink>"
     }
puts {
          </td>
  </tr>
  <tr>
    <td>Strom: </td>
    <td>
}
puts "<b>"
puts [format %3.2f $data_d]
puts " A</b>"
puts {
     <!--  XX_strom_zellen --></td>
    <td>
         <table width="360" border="0" cellspacing="0"
cellpadding="0">
      <tr>
        <td width="30"><font size="-1">0 A</font></td>
        <td><img src="../../solar/anzeige_rot.jpg" width="}
puts "[expr $data_d*$skala/$max_I_s+1]"
       puts {
            \" height="10"></td>
        <td width="30"><font size="-1">5 A</font>
              </td>
      </tr>
```

```
    </table>
          </td>
     <td>
          }
if {$grenz_I_s_min>$data_d} then {
     puts "<blink><b> WARNUNG (min)</b></blink>"
     }
     if {$grenz_I_s_max<$data_d} then {
     puts "<blink><b> WARNUNG (max)</b></blink>"
     }
puts {
          </td>
  </tr>
  <tr>
    <td>Leistung:</td>
    <td><!--  XX_leistung_zellen -->
          }
puts "<b>"
puts [format %4.2f $data_e]
puts " W</b>"
puts {
</td>
    <td>
     <table width="360" border="0" cellspacing="0"
cellpadding="0">
      <tr>
        <td width="30"><font size="-1">0 W</font></td>
        <td><img src="../../solar/anzeige_rot.jpg" width="}
puts "[expr $data_e*$skala/$max_W_s+1]"
       puts {
            \" height="10"></td>
        <td width="30"><font size="-1">200 W</font>
              </td>
      </tr>
    </table>
</td>
  </tr>
  <tr>
    <td> </td>
    <td> </td>
    <td> </td>
  </tr>
  <tr>
    <td><strong>Verbraucher</strong></td>
    <td> </td>
    <td> </td>
  </tr>
  <tr>
    <td>Spannung:</td>
    <td><!--  XX_spannung_verbraucher -->
```

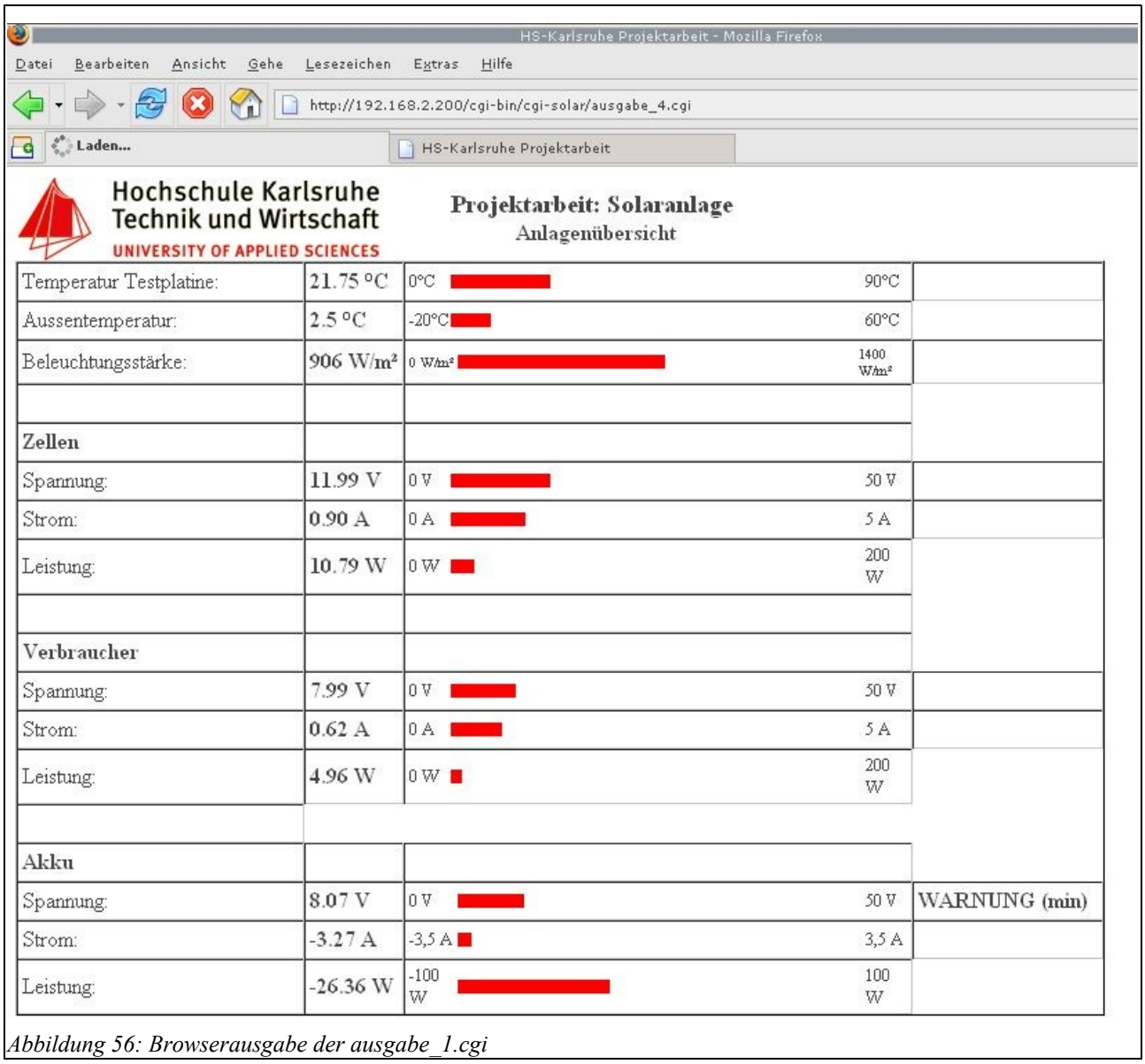
```
          }
          puts "<b>"
          puts [format %4.2f $data_f]
puts " V</b>"
puts {
     </td>
    <td>
                    <table width="360" border="0" cellspacing="0"
cellpadding="0">
      <tr>
        <td width="30"><font size="-1">0 V</font></td>
        <td><img src="../../solar/anzeige_rot.jpg" width="}
puts "[expr $data_f*$skala/$max_U_v]"
       puts {
            \" height="10"></td>
        <td width="30"><font size="-1">50 V</font>
              </td>
      </tr>
    </table>
              </td>
              <td>
          }
if {$grenz_U_v_min>$data_f} then {
     puts "<blink><b> WARNUNG (min)</b></blink>"
     }
     if {$grenz_U_v_max<$data_f} then {
     puts "<blink><b> WARNUNG (max)</b></blink>"
     }
puts {
          </td>
  </tr>
  <tr>
    <td>Strom:</td>
    <td><!--  XX_strom_verbraucher -->
          }
          puts "<b>"
                    puts [format %4.2f $data_g]
puts " A</b>"
puts {
     </td>
    <td>
                    <table width="360" border="0" cellspacing="0"
cellpadding="0">
      <tr>
        <td width="30"><font size="-1">0 A</font></td>
        <td><img src="../../solar/anzeige_rot.jpg" width="}
puts "[expr $data_g*$skala/$max_I_v]"
       puts {
            \" height="10"></td>
        <td width="30"><font size="-1">5 A</font>
```

```
                </td>
      </tr>
    </table>
                </td>
                          <td>
          }
if {$grenz_I_v_min>$data_g} then {
     puts "<blink><b> WARNUNG (min)</b></blink>"
     }
     if {$grenz_I_v_max<$data_g} then {
     puts "<blink><b> WARNUNG (max)</b></blink>"
     }
puts {
          </td>
  </tr>
  <tr>
    <td>Leistung:</td>
    <td><!--  XX_leistung_verbraucher -->
                     }
          puts "<b>"
          puts [format %4.2f $data_h]
puts " W</b>"
puts {
     </td>
    <td>
     <table width="360" border="0" cellspacing="0"
cellpadding="0">
      <tr>
        <td width="30"><font size="-1">0 W</font></td>
        <td><img src="../../solar/anzeige_rot.jpg" width="}
puts "[format %4.0f [expr $data_h*$skala/$max_W_v+1]]"
       puts {
            \" height="10"></td>
        <td width="30"><font size="-1">200 W</font>
                </td>
      </tr>
    </table>
                </td>
  </tr>
  <tr>
    <td> </td>
  </tr>
  <tr>
    <td><strong>Akku</strong></td>
    <td> </td>
    <td> </td>
  </tr>
  <tr>
    <td>Spannung:</td>
    <td><!--  XX_spannung_akku -->
```

```
          }
          puts "<b>"
          puts [format %4.2f $data_i]
puts " V</b>"
puts {
     </td>
    <td>
<table width="360" border="0" cellspacing="0" cellpadding="0">
      <tr>
        <td width="35"><font size="-1">0 V</font></td>
        <td><img src="../../solar/anzeige_rot.jpg" width="}
puts "[expr $data_i*$skala/$max_U_a]"
       puts {
            \" height="10"></td>
        <td width="30"><font size="-1">50 V</font>
              </td>
      </tr>
    </table>
     </td>
                          <td>
          }
if {$grenz_U_a_min>$data_i} then {
     puts "<blink><b> WARNUNG (min)</b></blink>"
     }
     if {$grenz_U_a_max<$data_i} then {
     puts "<blink><b> WARNUNG (max)</b></blink>"
     }
puts {
          </td>
  </tr>
  <tr>
    <td>Strom:</td>
    <td><!--  XX_strom_akku -->
          }
          puts "<b>"
          puts [format %4.2f $data_j]
puts " A</b>"
puts {
     </td>
    <td>
<table width="360" border="0" cellspacing="0" cellpadding="0">
      <tr>
        <td width="35"><font size="-1">-3,5 A</font></td>
        <td><img src="../../solar/anzeige_rot.jpg" width="}
puts "[expr ($data_j+3.5)*$skala/$max_I_a]"
       puts {
            \" height="10"></td>
        <td width="30"><font size="-1">3,5 A</font>
              </td>
      </tr>
```

```
    </table>
     </td>
          <td>
          }
if {$grenz_I_a_min>$data_j} then {
     puts "<blink><b> WARNUNG (min)</b></blink>"
     }
     if {$grenz_I_a_max<$data_j} then {
     puts "<blink><b> WARNUNG (max)</b></blink>"
     }
puts {
          </td>
  </tr>
  <tr>
    <td>Leistung:</td>
    <td><!--  XX_leistung_akku -->
          }
          puts "<b>"
          puts [format %4.2f $data_k]
puts " W</b>"
puts {
     </td>
    <td>
          <table width="360" border="0" cellspacing="0"
cellpadding="0">
      <tr>
        <td width="35"><font size="-1">-100 W</font></td>
        <td><img src="../../solar/anzeige_rot.jpg" width="}
puts "[expr ($data_k+100)*$skala/$max_W_a]"
       puts {
            \" height="10"></td>
        <td width="30"><font size="-1">100 W</font>
               </td>
      </tr>
    </table>
               </td>
  </tr>
</table>
</body>
</html>

}
```



Temperatur Testplatine:	21.75 °C	0°C	90°C	
Aussentemperatur:	2.5 °C	-20°C	60°C	
Beleuchtungsstärke:	906 W/m²	0 W/m²	1400 W/m²	
Zellen				
Spannung:	11.99 V	0 V	50 V	
Strom:	0.90 A	0 A	5 A	
Leistung:	10.79 W	0 W	200 W	
Verbraucher				
Spannung:	7.99 V	0 V	50 V	
Strom:	0.62 A	0 A	5 A	
Leistung:	4.96 W	0 W	200 W	
Akku				
Spannung:	8.07 V	0 V	50 V	WARNUNG (min)
Strom:	-3.27 A	-3,5 A	3,5 A	
Leistung:	-26.36 W	-100 W	100 W	

Abbildung 56: Browserausgabe der ausgabe_1.cgi

8.4. Tcl-Skript auf Linux-PC

Programm: led.tcl

Beschreibung: Steuerung einer LED über die Außentemperatur (tcl Script wird über externen Rechner ausgeführt)

```
#!tclsh
package require ow

# Verbindung zum OW-server herstellen
set host 192.168.2.200
set port 3333
OW::init u $host:$port

# Auslesen eines Wertes
set a [OW::get 28.AACD35000000/temperature]
puts $a

if {$a>23.0} then {
  set b 1
} else {
  set b 0
}

puts $b

# Schaltaugabe auf LED's
OW::put 20.386403000000/PIO.A $b
OW::put 20.386403000000/PIO.B $b
OW::put 20.386403000000/PIO.C $b
OW::put 20.386403000000/PIO.D $b

# Beenden der Verbindung zum OW-server
OW::finish
```

Dieses Skript ist auch auf den Router lauffähig, dazu muss man die IP-Adressen auf localhost ändern. Der Datenabgriff erfolgt am OW-Server. Nachteil dieser Methode ist, dass in der Zeit in der man Daten abruft oder schreibt niemand mehr auf den OW-Server Zugreifen kann. Vom öffnen bis zum schließen der Verbindung hat man einen Exklusivzugriff.

9. Ausblick

Weiterführende Projekte könnten an folgenden Punkten arbeiten:

- Verringerung der Zugriffszeiten
- Messdaten Protokollieren (Datenbank)
- Diagramme Zeichnen auf Basis der Protokollierten Daten
- Automatische Benachrichtigung (Status- & Alarmmeldung) per E-Mail oder SMS
- Regelung & Steuerung von Anlagen
- Ausweitung auf weitere Anlagen (neue Solaranlage)
- Gebäudemanagment (z.B. Zutrittskontrolle über i-button, Fernsteuerung etc.)

10. Anhang

10.1. Abbildungsverzeichnis

10.2. Literatur

Folgende Literatur wurde zur Hilfe genommen:

Mark Harrison
Effektiv Tcl/Tk programmieren / Mark Harrison und Michael McLennan
Addison Weslay Longman Verlag, 1998
1. Auflage
ISBN 3-8273-1409-7

Brent B. Welch
Practical Programming in Tcl and Tc
3. Auflage
ISBN 0-13-022028-0

Reiner Maurer
Html und CGI-Programmierung: dynamische WWW-Seiten erstellen / Reiner Maurer
Heidelberg: dpunkt, Verl. für digitale Technologie
1. Auflage
ISBN 3-920993-28-4

10.3. Quellen aus dem Internet

www.wikipedia.de
www.maxim-ic.com
www.ibutton.com
http://hkrott.iicm.edu/docs/seminar/sem2003_oneWire.pdf
http://pdfserv.maximic.com/en/an/tb1.pdf
http://pdfserv.maxim-ic.com/en/an/AN1796.pdf
http://pdfserv.maximic.com/en/an/appibstd.pdf
http://pdfserv.maxim-ic.com/en/an/app74.pdf
http://pdfserv.maxim-ic.com/en/an/app126.pdf
http://pdfserv.maxim-ic.com/en/an/app148.pdf
http://pdfserv.maxim-ic.com/en/an/app159.pdf
http://pdfserv.maxim-ic.com/en/an/app193.pdf

http://pdfserv.maxim-ic.com/en/an/app214.pdf
http://pdfserv.maximic.com/en/an/app2420.pdf
http://cyberforat.squat.net/openwrt/OpenWrt-HOWTO/x67.html
http://owfs.sourceforge.net
http://www.cyber-wulf.de/a_wl500g.html
http://wiki.opennet-initiative.de/index.php/Opennet_Firmware_Asus_Erstinstallation
http://freshmeat.net/projects/owfs/
http://www.my-space.li/schule/editor_VI.pdf
http://wiki.openwrt.org/UsbStorageHowto#head-0a12e438860955ebd81c911a67c9222f80b89ad0
http://forum.opennet-initiative.de/thread.php?threadid=441&sid=&hilight=time
http://wiki.openwrt.org/OpenWrtDocs/Using
http://doolittle.faludi.com/ntpclient/README

10.4. Inbetriebnahme von AP an der HS-Karlsruhe

10.4.1. Dienstvereinbarung

Dienstvereinbarung
über den Betrieb und die Nutzung eines WLANs
an der Fachhochschule Karlsruhe - Hochschule für Technik

Zwischen
der Fachhochschule Karlsruhe - Hochschule für Technik
und dem Personalrat
an der Fachhochschule Karlsruhe - Hochschule für Technik
wird nach § 73 i.V. mit § 79 Abs. 1 Nr. 8 LPVG Baden-Württemberg
in der Fassung vom 06.12.1999
folgende Dienstvereinbarung geschlossen:

§ 1 Gegenstand

Die Dienstvereinbarung regelt die Installation und den Einsatz von kabellosen Funknetzen, so genannten Wireless-Local-Area Networks (WLAN) und deren Komponenten (Access-Points, AP) an der Fachhochschule Karlsruhe - Hochschule für Technik.

§ 2 Geltungsbereich

Die Dienstvereinbarung gilt für den gesamten Bereich der Fachhochschule Karlsruhe - Hochschule für Technik.

§ 3 Zweckbestimmung und Ziel

Das Einrichten von WLANs an der Fachhochschule Karlsruhe - Hochschule für Technik dient dem Zweck, es Hochschulangehörigen, insbesondere Studierenden, zu ermöglichen, mit nicht kabelgebundenen Endgeräten, z. B. Laptops, mit dem Datennetz der Fachhochschule Karlsruhe in Verbindung zu treten. Die Erfüllung der grundlegenden Sicherheits- und Gesundheitsanforderungen (siehe §§ 4, 5) ist für den Betrieb von Funknetzen zwingend notwendig. Diese Anforderungen müssen verantwortungsbewusst angewandt werden, um den Stand der Technik beim Betrieb sowie technische und wirtschaftliche Erfordernisse zu berücksichtigen.

§ 4 Datensicherheit

Die Übertragung der Daten erfolgt mit starker Verschlüsselung vom Endgerät bis zum Ausgang des Funknetzes. Dies entspricht einem WLAN-Access Point, der den Zugang über das VPN-Netz zum VPN-Gateway realisiert. Hinter dem VPN Gateway wird die übliche leitungsgebundene, unverschlüsselte Übertragung genutzt. Von dieser Regelung können Teilnetze für Lehre und Forschung

ausgenommen werden, wenn sie vom sonstigen Netz der Fachhochschule Karlsruhe zuverlässig und sicher abgetrennt sind und dies für die Durchführung von Laborversuchen oder Forschungsarbeiten notwendig ist.

§ 5 Berücksichtigung von Gesundheitsschutz

Die Hochschule wird beim Betrieb des WLANs die gesetzlichen Grenzwerte der 26. Verordnung zur Durchführung des Bundes-Immissionsschutzgesetzes einhalten. Der Grenzwert findet sich ebenso in der Unfallverhütungsvorschrift „Elektromagnetische Felder" GUV-V B11 vom Juli 2002 wieder.

§ 6 Leistungsmerkmale

Es sollen nur WLAN-Karten in den PCs bzw. Notebooks eingesetzt werden, die den zuständigen AP mit der geringsten Ausgangsleistung erreichen können.

§ 7 Installationsorte von Access-Points

Beim Aufbau von WLAN Infrastrukturen sind die Standorte von APs so zu wählen, dass eine Strahlenbelastung an persönlichen Arbeitsplätzen möglichst vermieden wird.

§ 8 Installation und Betrieb

Für die korrekte Installation der Zugangssicherheit bzw. der Datensicherheit und des Betriebes trägt die betreibende Organisationseinheit die Verantwortung. Das Rechenzentrum (RZ) unterstützt dabei und hat das Recht, Installationen zu überprüfen, die Konfiguration dem aktuellen Standard der Sicherheitsrichtlinien anzupassen und ggf. den Betrieb nicht hinreichend abgesicherter Installationen zu unterbinden. Dies schließt auch die Installation und den Betrieb separater Teilnetze (siehe $4) ein.

§ 9 Kennzeichnung

Die APs werden von den betreibenden Organisationseinheiten nach den Vorgaben des RZs deutlich sichtbar gekennzeichnet. Vor Inbetriebnahme neuer APs werden die betroffenen Mitarbeiterinnen und Mitarbeiter informiert. Die installierten APs werden durch das RZ in einem Verzeichnis erfasst. Der Personalrat und die Fachkraft für Arbeitssicherheit können jederzeit Einsicht in das Verzeichnis nehmen.

§ 10 Änderungen und Erweiterungen

Diese Dienstvereinbarung kann jederzeit durch übereinstimmenden Beschluss der Vertragspartner geändert werden.

§ 11 Inkrafttreten und Geltungsdauer

Diese Vereinbarung tritt mit der Unterzeichnung durch beide Vertragspartner in Kraft. Sie gilt unbefristet.

§ 12 Kündigung

Eine Kündigung dieser Vereinbarung ist jederzeit aus triftigem Grund möglich.

§ 13 Salvatorische Klausel

Sollten einzelne Bestimmungen dieser Vereinbarung unwirksam sein oder werden, so berührt dies die Gültigkeit der übrigen Bestimmungen dieser Vereinbarung nicht. Die Parteien verpflichten sich, unwirksame Bestimmungen durch neue Bestimmungen zu ersetzen, die der in den unwirksamen Bestimmungen enthaltenen Regelungen in rechtlich zulässiger Weise gerecht werden. Entsprechendes gilt für in der Vereinbarung enthaltenen Regelungslücken. Zur Behebung der Lücke verpflichten sich die Parteien auf eine Art und Weise hinzuwirken, die dem am nächsten kommt, was die Parteien nach dem Sinn und Zweck der Vereinbarung bestimmt hätten, wenn der Punkt von ihnen bedacht worden wäre.

Karlsruhe, 12.Jan 2005

10.4.2. Antrag

WLAN Anschlüsse (Accesspoints) Stand 2005-06-20
Die folgenden Vorgaben des RZ hinsichtlich des Anschlusses von WLAN an das Netzwerk der Hochschule sind zu beachten:

- Der Betrieb von WLAN ist ausschließlich im Einklang mit der Verwaltungsund Benutzungsordnung, der Betriebsordnung, sowie der Dienstvereinbarung über den Betrieb und die Nutzung eines WLANs zulässig.
- Jeder Anschluss eines Accesspoints ist dem RZ vom zuständigen IT-Administrator mit den gewählten Konfigurationsdaten vor der Inbetriebnahme zur Genehmigung vorzulegen.
- Änderungen genehmigter Konfigurationsdaten sind dem RZ ebenfalls vorab zur Zustimmung vorzulegen.
- Der Anschluss von Accesspoints ist aus Sicherheitsgründen ausschließlich an das Laptopnetz erlaubt. Der Anschluss an Pool- und Wissenschaftsnetze ist untersagt. Gleiches gilt in den Gebäuden, in welchen Laptop- und Poolnetz noch gemeinsam betrieben werden bis zur Trennung derselben.
- Jeder Accesspoint ist in unmittelbarer Umgebung deutlich sichtbar zu kennzeichnen (§9 der Dienstvereinbarung über den Betrieb und die Nutzung eines WLANs). Die Kennzeichnung geschieht durch Anbringen des Hinweisschildes. Auch sind die betroffenen Mitarbeiterinnen und Mitarbeiter vor Inbetriebnahme eines neuen Accesspoints zu informieren. Dies geschieht in geeigneter Weise durch die den Accesspoint betreibende Organisationseinheit.
- Das RZ leistet keinen aktiven Support, d.h. die Wartung der Accesspoints als auch die Beantwortung von Nutzeranfragen hinsichtlich derselben ist vom zuständigen IT-Administrator zu erbringen. Der zuständige IT-Administrator ergibt sich aus dem in der SSID zu verwendenden OU-Kürzel (siehe unten).
- Das RZ empfiehlt die Verwendung der Kanäle 1, 6, 11 um die vorhandenen Frequenzbereiche optimal zu nutzen. Hierbei sollte auf einen grossen räumlichen Abstand zwischen zwei Stationen welche die selbe Kanalnummer verwenden geachtet werden.
- Zur eindeutigen Identifikation eines Accesspoints ist die SSID welche die Accesspoints verwenden zwingend wie folgt zu vergeben: 'Gb' Gebäudekürzel '-' OU-Kürzel '-AP' zweistellige laufende Nummer Zum Beispiel: GbM-WI-AP04 oder GbK-W-APO1
- Der Betrieb von DHCP Servern in Accesspoints kann die Nutzung durch andere Teilnehmer stören und ist darum untersagt.
- Der Aufbau von Netzwerkschleifen (Loop) mittels Accesspoints kann die Nutzung durch andere Teilnehmer stören und ist darum untersagt.
- Bei Betriebsstörungen behält sich das Rechenzentrum die sofortige Abschaltung der betroffenen Accesspoints ohne weitere Rückfrage vor.
- Das Rechenzentrum behält sich vor diese Liste entsprechend zusätzlicher Erkenntnisse zu erweitern. Es gilt jeweils die aktuelle Liste.

Anschlusswunsch eines Accesspoints an das Laptopnetz der
Hochschule Karlsruhe - Technik und Wirtschaft
Datum:
Organisationseinheit/IT-Administrator:
Hardwaretyp:

Aufstellungsort (Gebäude, Raum, etc):
gewünschte LAN-Dose:
Verwendete Standards 802.11[abg]:
Verwendete Kanalnummer:
Verwendete SSID:
Verwendeter Antennentyp (Rundstrahler, Richtantenne, etc):

10.5. Datenblätter

Alle Datenblätter sind auf der Beigefügten CD.

10.6. Passwörter & IP-Adressen

Im Auslieferungszustand:

- Benuzername: admin
- Passwort: admin

Im opennet-openwrt-Firmware original:

- Benuzername: root
- Passwort: admin
- IP: 172.16.0.1
- Subnet-Mask: 255.255.0.0
- Webinterface: http://172.16.0.1

Momentane einstellungen Router 1:

- Webinterface: http://192.168.1.1
- LAN-IP: 192.168.1.1
- Subnet-Mask: 255.255.255.0
- DHCP: on
- Erste DHCP-Adresse: 192.168.1.50
- Erste DHCP-Adresse: 192.168.1.60
- Benutzername: root
- Passwort: Solar#01
- WLAN: ausgeschaltet
- WAN-IP:

Momentane einstellungen Router 2:

- Webinterface: http://192.168.2.200
- LAN-IP: 192.168.2.200
- Subnet-Mask: 255.255.255.0
- DHCP: on
- Erste DHCP-Adresse: 192.168.2.50
- Erste DHCP-Adresse: 192.168.2.60
- Benutzername: root
- Passwort: Solar#02
- WLAN: ausgeschaltet
- WAN-IP:

11. CD mit allen Daten