1. Kurzfassung

Um die Effizienz einer privaten Anlage, beispielsweise eines Solar-Heiz-Systems oder eines Gewächshauses zu steigern, indem man Vorgänge automatisiert und verbessert, benötigt man eine zentrale Steuereinheit. In einem Gewächshaus steuert diese zum Beispiel die Temperatur und den Lichteinfall durch das Heben und Senken von Fenstern und Abdeckungen. Auch eine automatische Ausrichtung von Solar-Panels zur Sonne oder eine Temperaturregelung in einem beheizten Wassertank ist möglich. Das Betreiben eines solchen Mechanismus erfordert viele Sensoren und Motoren, die reibungslos zusammenarbeiten müssen. Die Kontrolle dieses Systems kann nur durch ein computer-ähnliches Gerät gewährleistet werden. Da aber zum Beispiel der Einsatz eines normalen Computers die Energiegewinnung durch Solarzellen ineffizient macht und viel Platz verbraucht, sollte eine geeignete, kleine, energiesparende und programmierbare Netzwerkplatine verwendet werden. Diese Eigenschaften werden durch die Platine „avr-net-io“ der Firma Pollin erfüllt. Mit der Software „ethersex“ bietet sie alle benötigten Funktionen. Der avr-net-io-Chip ist ein Mikrocontroller. Mikrocontroller sind Prozessoren, die im Gegensatz zu herkömmlichen Computerchips den Speicher und die Ein- und Ausgänge auf einem Chip zusammenfassen und dadurch leichter, kleiner und energiesparender sind. Durch die AVR-Platine als „embedded system“ ist der Mikrocontroller über ein herkömmliches lokales Computer-Netzwerk steuer- und programmierbar und kann dadurch auch selbstständig agieren. Ein „embedded system“ bezeichnet hier den avr-net-io, der ein oben genanntes System kontrollieren und durch die Vernetzung mit dem Hausnetzwerk Daten verarbeiten soll. Der avr-net-io kann durch seinen Netzwerkanschluss bequem direkt mit einem Patchkabel an ein normales Local Area Network (LAN) angeschlossen werden, auch das Standardprotokoll funktioniert problemlos. Man kann die Platine im Onlineshop von Pollin bestellen, entweder als Bausatz, oder, um das Zusammenlöten und eventuelle Fehler zu vermeiden, als Fertigmodul. Um allerdings ein komplexeres System zu steuern, ist es sinnvoll, auch den Bausatz für die 25-polige SUB-D Anschlussplatine zu kaufen und zu löten, da sie es ermöglicht, noch mehr Sensoren anzuschließen. Das erste Problem ist das Aufspielen der ethersex-Software auf den Chip. Das geht nicht über das Netzwerk, da der benutzte Chip ATMega32 zu wenig Speicher hat, um mit der bereits installierten Firmensoftware das ethersex-System zu speichern. Um dieses zu installieren, muss der Speicher des Chips überschrieben werden. Das funktioniert nur mithilfe eines ISP-Kabels (ISP: In-System-Programmierung), welches eine direkte Verbindung mit einem Computer herstellt. Dieses muss aus einem seriellen Anschluss, einigen Widerständen und Kabeln gelötet werden. Nachdem ethersex kompiliert und installiert ist, kann man die Platine über das Netzwerk ansteuern. Ethersex bietet dem Benutzer viele Funktionen, Informationen über den Status von Gerät und Sensoren abzurufen und zu verändern. Als Beispiel wird der Temperatur-Sensor DS18S20+ mithilfe eines weiteren Kabels angeschlossen. Über das Netzwerk kann man sich mit zwei Befehlen die aktuelle Temperatur anzeigen lassen. Da dies relativ umständlich ist, wird eine Erweiterung von ethersex installiert, mit der man sich über einen Browser eine Temperaturkurve anzeigen lassen kann. Der Einsatz des avr-net-io ermöglicht also auch eine Aufzeichnung vielfältiger Daten, beispielsweise Statistiken über Temperaturen oder Luftfeuchtigkeit. Ein sehr großes Problem der ethersex-Software ist allerdings, dass sie nur sehr umständlich zu bedienen ist. Daher ist es wichtig, Neuanwendern und Benutzern die Handhabung der Software zu vereinfachen. Dazu bietet es sich an, ein Programm mit einer Oberfläche zu schreiben, welches die Funktionen von ethersex grafisch darstellt und damit die Benutzung leichter macht. Da einige Funktionen der Software auch über den Browser zugänglich sind, bietet es sich an, das Programm in PHP zu schreiben, um unnötige Schwierigkeiten zu vermeiden. Das fertige Programm soll auf einem privaten Webserver auf dem eigenen Computer laufen, wobei das Betriebssystem Linux empfohlen wird, außerdem soll es einfach und bequem zu benutzen sein. Das hat den Vorteil, dass man viel schneller eine Platine benutzen und einrichten kann und Benutzer nicht durch komplizierte Befehle entmutigt werden. Das Programm wird nach dem Wettbewerb unter der GPL-Lizenz im Internet veröffentlicht.

2. Einleitung

Erneuerbare Energien sind ein Top-Thema der modernen Diskussion. Es gibt daher auch viele Möglichkeiten, diese privat zu benutzen und zu produzieren. Die Nachfrage nach privaten Anlagen steigt, wie am Beispiel einer Statistik der installierten Photovoltaik-Systeme in den Jahren 2000-2007 zu sehen ist (siehe Abbildung 2.1). Hier stellt sich dem Nutzer aber die Frage: Sind die Systeme zur privaten Erzeugung von Energie effizient? Eine genaue Antwort lässt sich nicht geben, aber oftmals kann man sagen, dass sich der Einsatz nicht unbedingt lohnt, da die Anschaffungskosten höher sind als der Gewinn. Dies könnte sich zwar durch neue Technologien mit preiswerteren Produktionsverfahren ändern, aber die Effizienz bliebe doch relativ klein. Wie also könnte man die Nutzung erneuerbarer Energien effizienter gestalten? Eine Möglichkeit bietet der avr-net-io von Pollin (siehe: http://www.pollin.de/, Literaturverzeichnis), mit dem bestimmte Optimierungsvorgänge automatisiert werden können. Einige Beispiele im Bereich der Sonnenenergie wären:

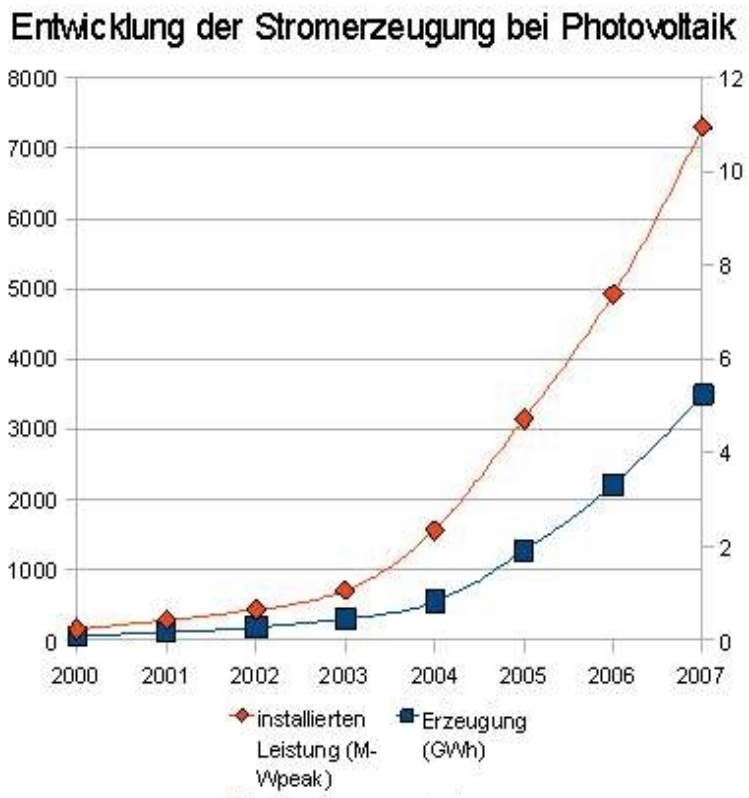

Abb. 2.1: Quelle: http://www.energie2null.de/, siehe Literaturverzeichnis, unwesentlich bearbeitet

- Einen mit Solarenergie beheizten Wassertank mithilfe von Temperatursensoren und Schaltern auf konstanter Temperatur zu halten, indem die Platine je nach Temperaturveränderung die Solaranlage an- und ausschaltet
- Solar-Panels auf dem Dach mit einigen Motoren je nach Uhrzeit oder mithilfe von Lichtsensoren nach der Sonne hin auszurichten
- Ein Gewächshaus mit einem System aus Motoren, Luftfeuchtigkeits-, Temperatur- und Lichtsensoren zu automatisieren, indem Abdeckungen und Fenster je nach Sensorendaten geöffnet und geschlossen werden und damit optimale Wachstumsverhältnisse erreicht werden können

Solche Effizienzsteigerungen sind mit dieser Platine für den Privathaushalt möglich, aber das erste Einarbeiten in die Software ethersex und deren Bedienung fallen dem Einsteiger schwer. Die meisten aktuellen Benutzer der Platine, die größtenteils auf der Internetseite mikrocontroller.net aktiv sind (siehe: http://www.mikrocontroller.net/, Literaturverzeichnis), kennen sich in diesem Fachbereich sehr gut aus. Um aber die Möglichkeit der Nutzung dieser Technologie einer größeren Zahl zur Verfügung zu stellen und auch den Experten die Bedienung zu erleichtern, soll ein Programm geschrieben werden, das folgende Funktionen bzw. Eigenschaften erfüllt:

- grafische Oberfläche
- einfache Konfiguration

- transparente Bedienung
- GPL-Lizenz
- verständlicher Programmcode
- Plattformunabhängigkeit
- Schnittstellen für alle grundlegenden Funktionen der ethersex-Software: Abrufen von Informationen, Konfiguration und Senden von Befehlen
- einfache Möglichkeit, Erweiterungen hinzuzufügen (Beispiel: Temperaturkurven von Sensoren, etc.)

Ähnliche Programme existieren bereits, entsprechen aber nicht oder unvollständig den genannten Kriterien. Ein Beispiel ist das „AVR-NET-IO Steuerprogramm“ (siehe Abb. 2.2). Dieses erfüllt, soweit es von einem außen stehenden Betrachter beurteilt werden kann, folgende Kriterien:

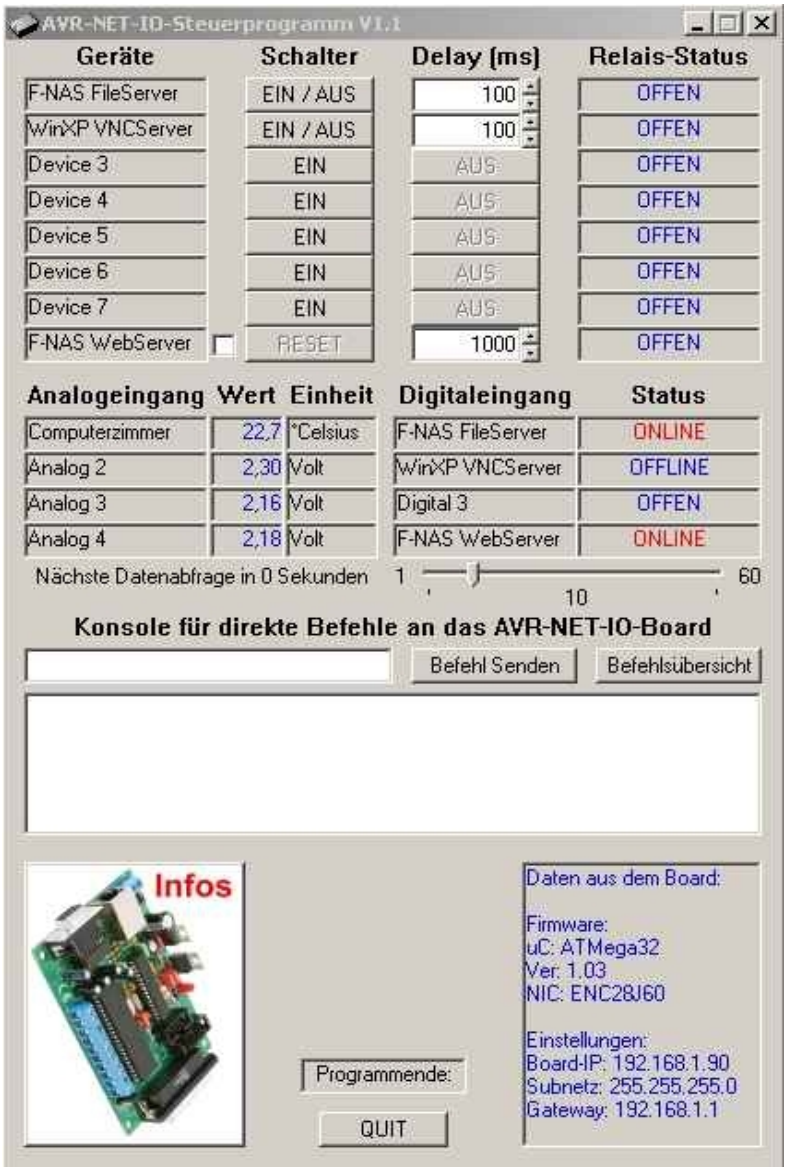

Abb. 2.2: Quelle: http://www.samurai1967.dyndns.org/avr-net-io.html, siehe Literaturverzeichnis, Erlaubnis zur Verwendung wurde erteilt

- grafische Oberfläche
- transparente Bedienung
- (Plattformunabhängigkeit)

(Damit letztere gegeben ist, muss das Programm zuerst portiert werden, da es in der Sprache PureBasic geschrieben ist.)

Die einfache Konfiguration ist nicht gewährleistet, da die Platine zwingend die IP-Adresse 192.168.0.90 besitzen muss. Sollte dies nicht der Fall sein, kann das Programm nicht verwendet werden und muss umgeschrieben werden. Auch arbeitet es mit der originalen Firmware von Pollin und nicht mit der ethersex-Software. Ob der Programmcode verständlich und öffentlich ist und es eine einfache Möglichkeit gibt, ihn zu erweitern, kann hier nicht gesagt werden. (Quelle: siehe Literaturverzeichnis)

Neben den genannten Kriterien soll beim fertigen Programm außerdem geprüft werden, ob das grafische Interface wirklich eine Erleichterung der Bedienung darstellt.

3. Herstellen der Arbeitsgrundlage

Um eine wie oben beschriebene Software zu programmieren, muss die Platine zuerst einmal zusammengebaut, ein Temperatursensor angeschlossen und das Gerät mit der Firmensoftware über ein Lokales Netzwerk in Betrieb genommen werden. Dazu werden im Onlineshop von Pollin (siehe: http://www.pollin.de/shop/, Literaturverzeichnis) folgende Artikel bestellt:

- 94-810 073 Fertiggerät AVR-NET-IO
- 94-810 055 Bausatz SUB-D-Anschlussplatine
- 94-350 086 Steckernetzteil, 3...12V-/500mA
- 94-180 014 Temperatur-Sensor DS18S20+
- 94-540 834 CAT.5 Patchkabel
- 94-560 430 Cinchverlängerung

(eventuelle Unklarheiten über die Verwendung einzelner Teile werden in diesem Abschnitt behandelt.)

Der avr-net-io gehört, wie der Name bereits ausdrückt, zur Hardwarefamilie der AVR-Geräte (**A**dvanced **V**irtual **R**ISC), die einen RISC-Prozessor benutzen (siehe: http://de.wikipedia.org/, Literaturverzeichnis). Er wird zusammengelötet bestellt, um Fehler zu vermeiden, was bei geringfügiger Löt-Erfahrung sinnvoll sein kann, da der Aufbau der Platine für Anfänger relativ kompliziert ist und leicht Bestandteile beschädigt werden können. Die Spezifikationen sind wie folgt (Quelle: siehe Literaturverzeichnis):

- Betriebsspannung: ~9V
- Stromaufnahme ca. 190 mA
- 8 digitale Ausgänge (0/5 V)
- 4 digitale Eingänge (0/5 V)
- 4 ADC-Eingänge (10 Bit)
- Netzwerkcontroller ENC28J60
- ATMega32-Prozessor

Der Bausatz für die Anschlussplatine ist aber nicht als Fertiggerät vorhanden und muss gelötet werden, was sich aber aufgrund des simplen Aufbaus als relativ leicht erweist. Um den avr-net-io in Betrieb zu nehmen, muss das Kabel des Steckernetzteils an einem Ende durchtrennt und manuell in die Platine geklemmt werden. Das Netzteil kann die Normalspannung von 230V der Steckdose in einen Wert des Bereichs 3-12V umwandeln. Für einen optimalen Betrieb der Netzwerkplatine ist eine Spannung von 9V empfohlen, wie aus den oben genannten Board-Spezifikationen hervorgeht. Diese lässt sich am Netzteil einstellen. Das CAT.5 Patchkabel ist ein normales Netzwerkkabel, mit dem die Platine problemlos an ein Computernetzwerk angeschlossen werden kann. Um zu vermeiden, dass über einen Metall-Untergrund Kurzschlüsse in der Platine stattfinden, was durch die nicht isolierten Lötstellen passieren kann, wird eine nicht leitende Kunststofffolie unter dem Board befestigt. Da aus Erfahrung ein Teil des avr-net-io nach längerem Betrieb leicht überhitzt, wird ein Kühler angebracht. (Für den fertigen Aufbau siehe Abbildung 3.3). Der Temperatursensor DS18S20+ muss parasitär (siehe Abbildung 3.1) an den EXT.-Port des avr-net-io angeschlossen werden (siehe Abbildung 3.3). Um ein entsprechendes Kabel herzustellen, eignet sich die bestellte Cinchverlängerung, da sie die benötigten zwei Verbindungen hat, einfach zu bearbeiten ist und die „Reichweite“ des Temperatursensors durch

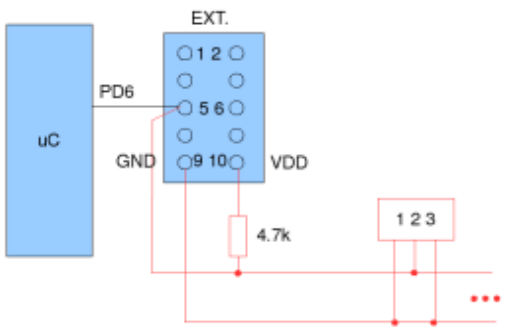

Abb. 3.1: Quelle: http://ethersex.de/, siehe Literaturverzeichnis

weitere Cinch-Kabel leicht verlängert werden kann (Cinch-Kabel lassen sich beliebig oft hintereinander stecken).

Um auch hier Kurzschlüsse und Beschädigungen zu vermeiden, ist es sinnvoll, sowohl einzelne Verbindungen und Lötstellen als auch den ganzen bearbeiteten Kabelabschnitt mit Klebeband zu isolieren. Es ist empfohlen, extrem stabiles und langlebiges Duct Tape zu verwenden. (siehe Abbildung 3.2).

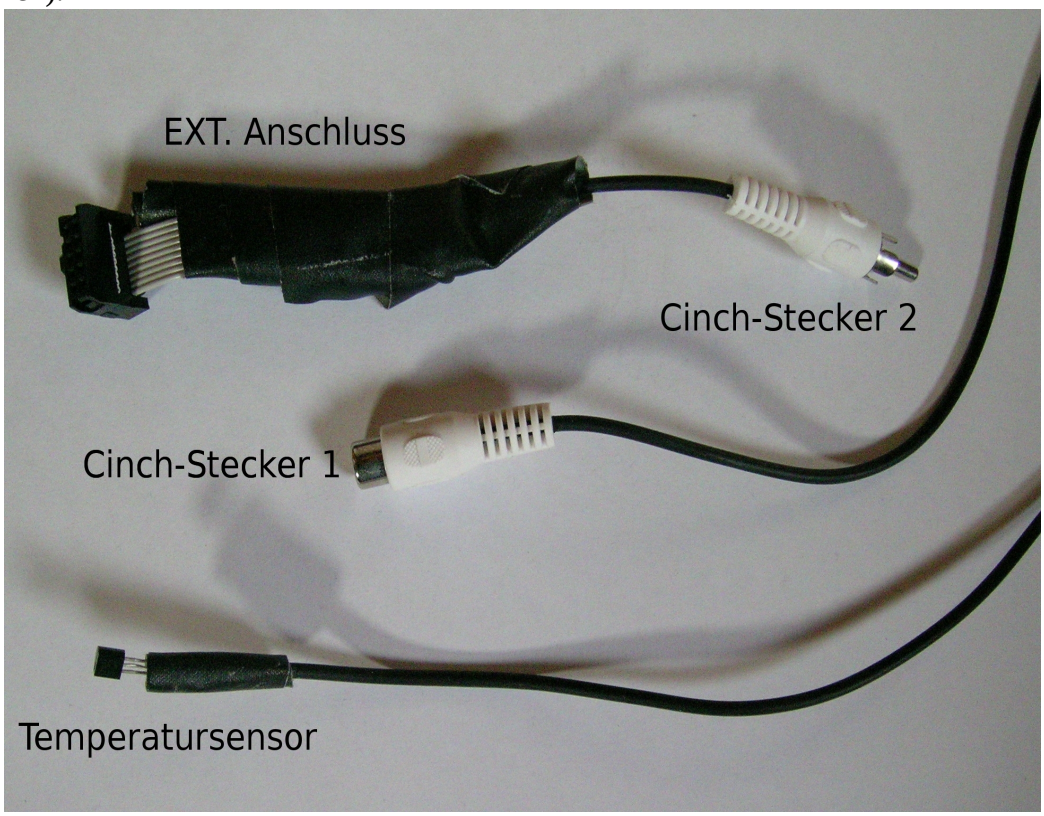

Abb. 3.2: Anschlusskabel des Temperatursensors

In Abbildung 3.3 sieht man den Aufbau der eigentlichen Platine (ohne Temperatursensor) und aller oben genannten benötigten Teile. Auch der EXT. Port ist sichtbar (siehe Beschriftung):

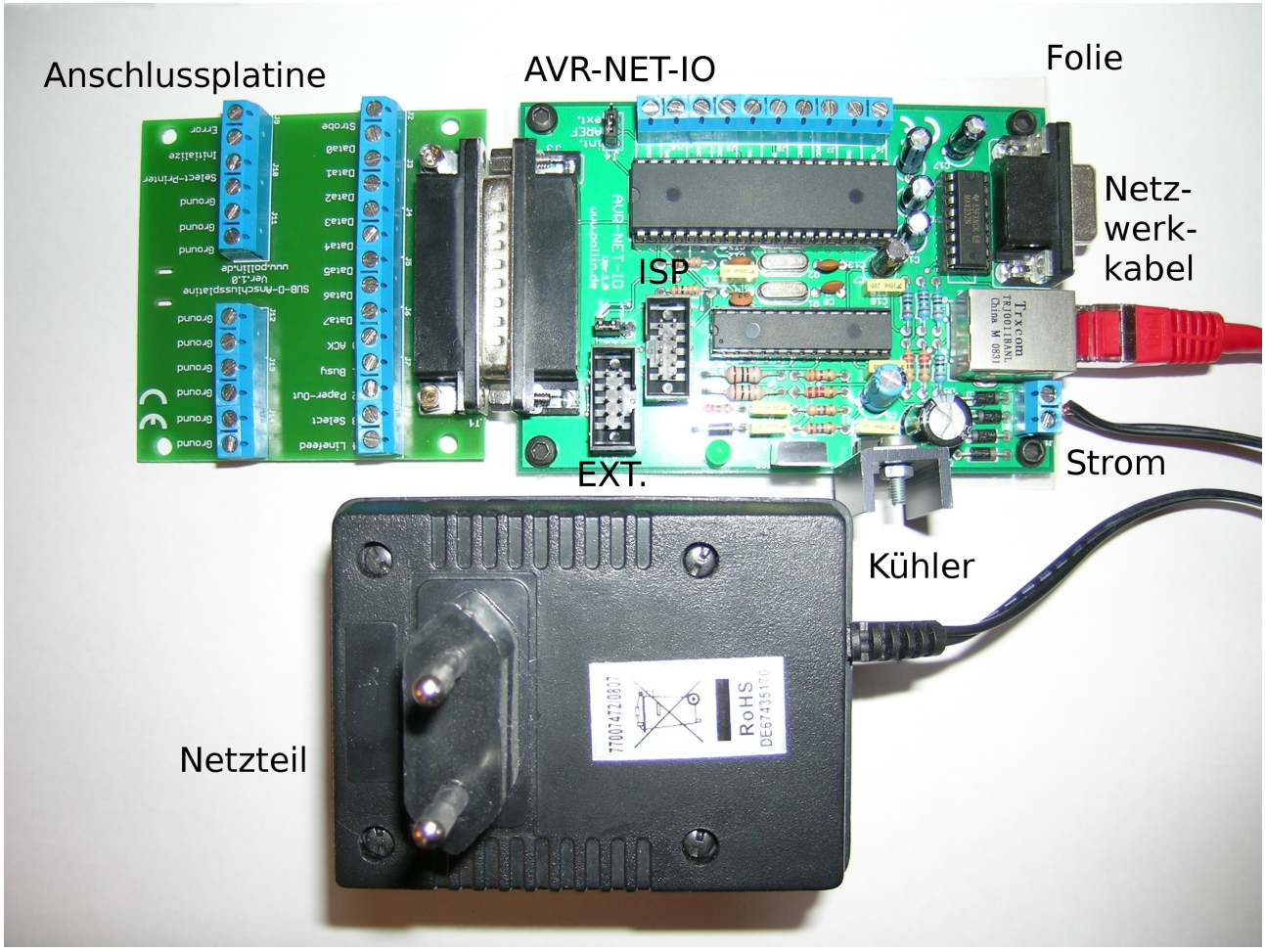

Abb. 3.3: Der Aufbau des AVR-NET-IO mit allen benötigten Teilen

Der avr-net-io kann nun schon in Betrieb genommen werden. Für das Anbringen weiterer Temperatursensoren müssen diese nur parallel zum ersten geschaltet werden.
Um aber das eigentliche Ziel erreichen zu können, muss die Software ethersex auf der Platine installiert werden. Das ist theoretisch über den Netzwerkport möglich, indem die bereits installierte Firmware über das Netzwerk die entsprechenden Daten erhält, abspeichert, installiert und dann gelöscht wird. Der benutzte ATMega32-Prozessor hat aber zu wenig Speicher, um gleichzeitig die vorinstallierte Firmware und die ethersex-Software zu speichern. Eine Möglichkeit, dieses Problem zu lösen, ist das Einbauen eines leistungsfähigeren Chips, beispielsweise des ATMega64. Da die Leistung dieses Prozessors aber nicht benötigt wird und der Kauf nur zusätzliche Kosten verursacht, gibt es auch die Alternative, die Firmware von Pollin direkt über den ISP-Port (In-System-Programmierung) mit der ethersex-Software zu überschreiben. Dazu ist ein weiteres simples Kabel nötig, das aus einem 25-poligen seriellen Anschluss und einigen Widerständen gelötet werden kann, wie in Abbildung 3.4 zu sehen ist (Aufgrund des avr-net-io eigenen ISP-Anschlusses (siehe Abbildung 3.3) muss die Schaltung leicht verändert werden). Wie beim Temperatursensor sollten die Lötstellen isoliert werden. Auch sollte man die Original-Hülle des seriellen Anschlusses anbringen, um Beschädigungen zu vermeiden (siehe Abbildung 3.5). Das fertige Kabel lässt sich an jeden gängigen Computer anschließen (außer bestimmte moderne Laptops etc.). Vor diesem Schritt ist es aber sinnvoll, mit einem digitalen Spannungsmessgerät die Verbindungen zu überprüfen, da durch Kurzschlüsse Schäden an Platine und Computer entstehen können. Um mit dem ISP-Kabel eine erfolgreiche Verbindung zwischen Computer und Platine herzustellen, darf außer dem ISP-Port kein anderer an der Platine belegt, also weder eine Netzwerkverbindung oder der Temperatursensor am EXT.-Port aktiv sein.

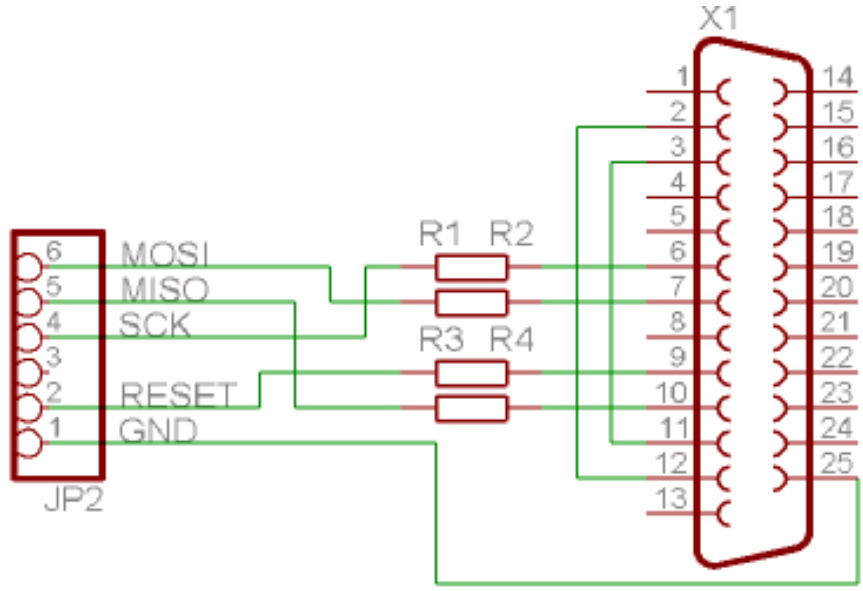

Abb. 3.4: Quelle: http://www.roboternetz.de/, Literaturverzeichnis

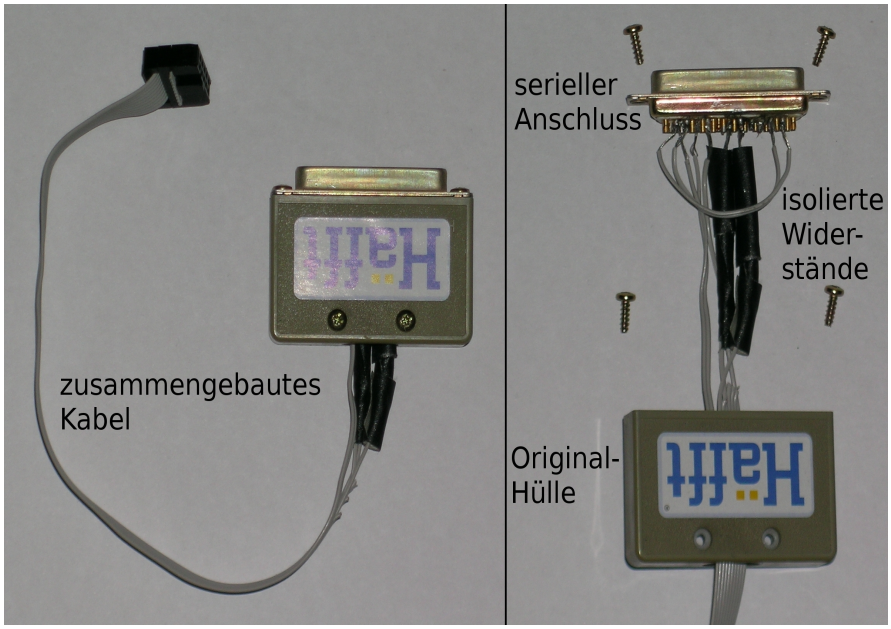

Abb. 3.5: Aufbau des ISP-Kabels

4. Programmieren des avr-net-io

4.1: Installation von ethersex

Mit dem ISP-Kabel ist es möglich, ethersex auf dem avr-net-io zu installieren. ethersex muss dazu im Format .hex vorliegen, also vorher aus dem Source Code gebaut bzw. „kompiliert“ werden. Dieser ist im Internet frei verfügbar (siehe http://www.ethersex.de). Das Kompilieren eines standardmäßig konfigurierten ethersex-Images kann auch online durchgeführt werden, sodass die fertige HEX-Datei nur noch heruntergeladen werden muss. Um aber Erweiterungen, wie z. B. die erwähnte Temperaturkurve, hinzuzufügen und die Platine programmieren zu können, sollte ethersex auf dem eigenen Computer genauer konfiguriert werden. Für das Kompilieren benötigt man unter Linux folgende Pakete: `gcc-avr, avr-libc, binutils-avr, m4, gawk, libncurses5-dev, make, dialog, git-core, avrdude, screen`

Diese lassen sich beispielsweise auf der weit verbreiteten und hier verwendeten Linux-Distribution **Ubuntu** in einer Shell mit dem Befehl

```
sudo apt-get install gcc-avr avr-libc  binutils-avr m4 gawk \
libncurses5-dev make dialog git-core avrdude screen
```

installieren. Danach wechselt man in das Verzeichnis mit den ethersex-Dateien und führt `make menuconfig` aus. Es erscheint ein Konfigurationsmenü (siehe Abbildung 4.1.1). Man kann hier die Standard- oder eine eigene Konfiguration laden und bearbeiten, auch speziell für den avr-net-io ist eine Konfiguration vorhanden. Neben den Netzwerkeinstellungen unter „`Network → Ethernet (ENC28J60) support`“, die unbedingt bearbeitet werden sollten, lohnt sich auch ein Blick in die Bereiche `Protocols` und `Applications`. Abgesehen vom Webserver, der auch für das genannte Beispiel der Temperaturkurve relevant ist, finden sich hier beispielsweise ein Jabber und IRC-Client oder Twitter-Service. Wurde die Software konfiguriert, kann das Menü auf dem Button `< Exit >` beendet werden. Die Frage nach dem Speichern der Konfiguration sollte mit `< Yes >` beantwortet werden, da sonst die Änderungen verworfen werden. Danach kann ethersex mit einem `make` kompiliert werden. War der Kompiliervorgang erfolgreich, wurde die Datei `ethersex.hex` angelegt, sowie eine Konfigurationsdatei, die für weitere Kompiliervorgänge wiederverwendet werden kann. Mit dem Programm `avrdude` kann das ethersex-Image jetzt über das ISP-Kabel auf die Platine übertragen werden. Im Falle des avr-net-io geht das mit dem Befehl:

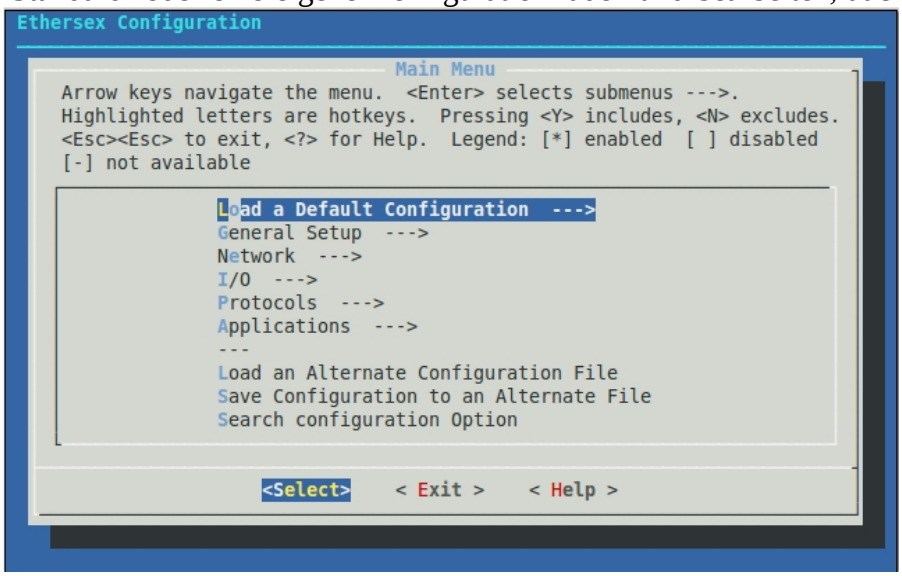

Abb. 4.1.1: Konfigurationsmenü von ethersex

```
avrdude -p m32 -c stk200 -u -U flash:w:/pfad/zu/ethersex/ethersex.hex:i
```

Da dieser aber je nach Platine und Computer variieren kann und schlecht zu merken ist, kann das Programm `kontrollerlab` (siehe Abbildung 4.1.2) verwendet werden, das durch eine grafische

Oberfläche einfacher zu bedienen ist und eine große Auswahl an verschiedenen Funktionen und Einstellungen bietet. Dieses kann von der Internetseite http://www.cadmaniac.org/ (siehe Literaturverzeichnis) im Bereich „download“ als Source Code oder fertiges Paket heruntergeladen und installiert werden.

Nach dem Start des Programms (je nach System kann es sein, dass es unter `root`, also mit Administrator-Rechten, laufen muss, da die Anschlüsse direkt angesprochen werden) muss `kontrollerlab` zuerst auf `avrdude` umgestellt werden, was durch die Auswahl der Option `AVRDUDE` im Menü `Project → Configure programmer` und Drücken des Buttons `OK` erreicht wird. Danach kann die HEX-Datei mit `Project → Upload hex file → (Auswahl) → ATMega32` auf die Platine geladen werden. `kontrollerlab` führt dabei den `avrdude`-Befehl automatisch aus.

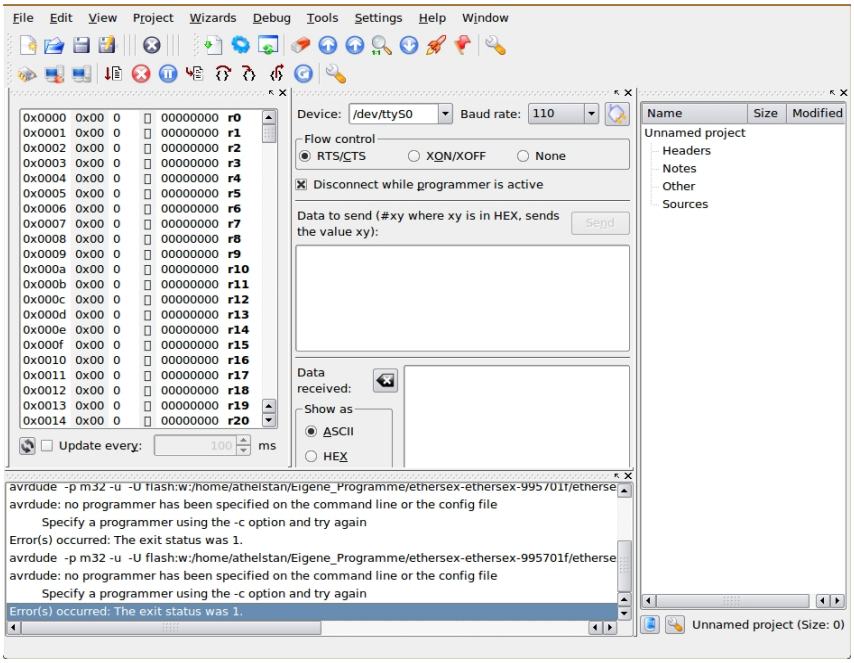

Abbildung 1: Abb. 4.1.3: Oberfläche von kontrollerlab, siehe Literaturverzeichnis

4.2: Analyse der installierten Software

Es gibt zwei Möglichkeiten, Befehle an die Platine zu schicken und eine Antwort zu erhalten. Die eine Methode läuft wieder über eine Shell mit dem Programm `netcat`. Hier kann man eine „Session“ (Sitzung) öffnen und Befehle eingeben. Das geht mit dem Kommando:

```
netcat 192.168.0.90 2701
```

(`192.168.0.90` ist die IP-Adresse, `2701` der Port der Platine. Beide sind je nach Konfiguration variabel.) Die Platine wird so lange auf Befehle warten, bis die Session manuell unterbrochen wird oder zu lange inaktiv ist. Eine andere Möglichkeit ist diese:

```
echo 'befehl' | netcat -q 1 192.168.0.90 2701
```

Hier wird der Befehl an den avr-net-io geschickt, die Antwort ausgegeben und `netcat` nach einer Sekunde beendet.

Die andere Methode ist das Schicken von Befehlen über den Webserver der Platine. Dazu wird er über einen normalen Browser geöffnet. Wurde ethersex mit dem Webserver richtig installiert, so erscheint eine Erfolgsmeldung (siehe Abbildung 4.2.1). Befehle können durch den Aufruf von

```
http://192.168.0.90/ecmd?befehl
```

gesendet werden. Die Ausgabe wird als einfacher Text angezeigt. Leerzeichen im Befehl werden durch ein Plus-Zeichen ersetzt. Für den Befehl `eins zwei drei` ist das dementsprechend:

```
http://192.168.0.90/ecmd?eins+zwei+drei
```

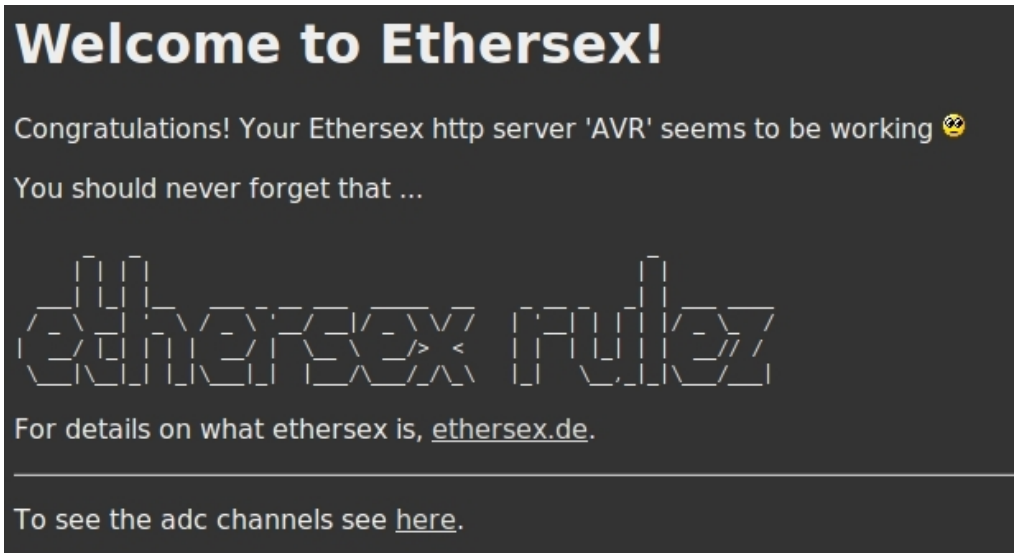

Abb. 4.2.1: Das ethersex-Webinterface

Beide Methoden können beim Schreiben der Steuersoftware verwendet werden, hier aber erstere, da der Webserver und seine Funktionen nicht zwingend installiert sind und die Ausgabe von bereits auf einer anderen Website angezeigtem Text unter Umständen komplizierter sein könnte.

Um sich alle möglichen Befehle anzeigen zu lassen, schickt man das Kommando `help` an die Platine. Folgende Möglichkeiten werden ausgegeben: `mac, show mac, show ip, show netmask, show gw, show version, ip, netmask, gw, reset, wdreset, bootloader, io, 1w list, 1w get, 1w convert, adc get, d, help, eeprom reinit`

Diese Befehle lassen sich, was auch für die spätere Programmierung relevant ist, in verschiedene Bereiche unterteilen:

Bereich	**Befehle**
Network configuration	mac, ip, netmask, gw show mac/ip/netmask/gw/version
Resetting the controller	reset, wdreset, bootloader
Port I/O	io get ddr/mask/pin/port io set ddr/port
Dallas 1wire	1w list/get/convert
Miscellaneous	adc get, d, help, eeprom reinit

Tab. 4.2.2: Unterteilung der Befehle in Bereiche

Nicht alle diese Befehle sind hier relevant, die wichtigsten werden in Tabelle 4.2.3 kurz erläutert:

Befehl	**Erläuterung**
mac	MAC-Adresse der Platine festlegen
ip	IP-Adresse der Platine festlegen
netmask	Netzmaske der Platine festlegen
gw	IP-Adresse des benutzten Routers festlegen (Gateway)
show mac/ip/netmask/gw/version	MAC, IP, Netzmaske, Gateway oder Version anzeigen
1w list	Alle aktiven Sensoren anzeigen
1w get	Wert eines Sensors anzeigen
1w convert	Werte der Sensoren aktualisieren
adc get	Werte der adc-Ports anzeigen

Tab. 4.2.3: Erläuterung der wichtigsten Befehle

Um nun den Temperatursensor zu nutzen, muss `1w list` ausgeführt werden. Wurde der Sensor richtig angeschlossen, erscheint folgende Ausgabe (der HEX-Wert ist für jeden Sensor anders):

```
1w list
106511a901080085
OK
```

Bei mehreren Sensoren werden auch mehr HEX-Werte angezeigt, sind diese falsch oder nicht angeschlossen, lautet die Ausgabe nur `OK`. Bevor aber der Temperaturwert des Sensors abgelesen werden kann, muss zuerst eine Messung durchgeführt werden. Dies geht mit `1w convert` für alle Sensoren oder `1w convert 106511a901080085` für einen einzelnen. Danach kann die Temperatur mittels `1w get 106511a901080085` abgerufen werden (Im Gegensatz zu `convert` kann `get` nicht alle Sensoren auf einmal ansteuern). Die Ausgabe ist auf 0,5°C genau:

```
1w get 106511a901080085
Temperatur:  18.8
```

In einem Bash-Script oder anderem Programm kann man diesen Vorgang automatisieren, um beispielsweise Statistiken zu erstellen. ethersex bietet aber auch eine Möglichkeit, Echtzeit-Temperaturkurven zu benutzen. Die benötigten Dateien „Xow.ht“ und „ow.ht“ sind bereits vorhanden, müssen aber noch beim Kompilieren eingebunden werden. Dazu müssen der OneWire-Support (der bereits zum Abrufen der Temperatur benutzt wurde und standardmäßig aktiv ist), der Webserver und das Datei Inlining aktiviert sein. Diese Funktionen kann man im Konfigurationsmenü beim Kompiliervorgang einstellen. Sind die Dateien in ethersex eingebunden, kann die alle paar Sekunden aktualisierte Temperaturanzeige im Browser unter

```
http://192.168.0.90/ow.ht
```

bzw. als farbige Temperaturkurve (siehe Abbildung 4.2.4, die dargestellte Kurve zeigt den Temperaturverlauf beim Lüften eines Zimmers) unter

```
http://192.168.0.90/Xow.ht
```

angezeigt werden. Die nicht grafische Temperaturanzeige `ow.ht` besteht dabei nur aus der unter dem Liniendiagramm stehenden Tabelle. Für jeden weiteren angeschlossenen Temperatursensor kommt eine weitere Tabellenreihe und andersfarbige Temperaturkurve hinzu. Größe und Eigenschaften des Systems können vor dem Kompilieren in der Datei Xow.ht.m4 definiert werden.

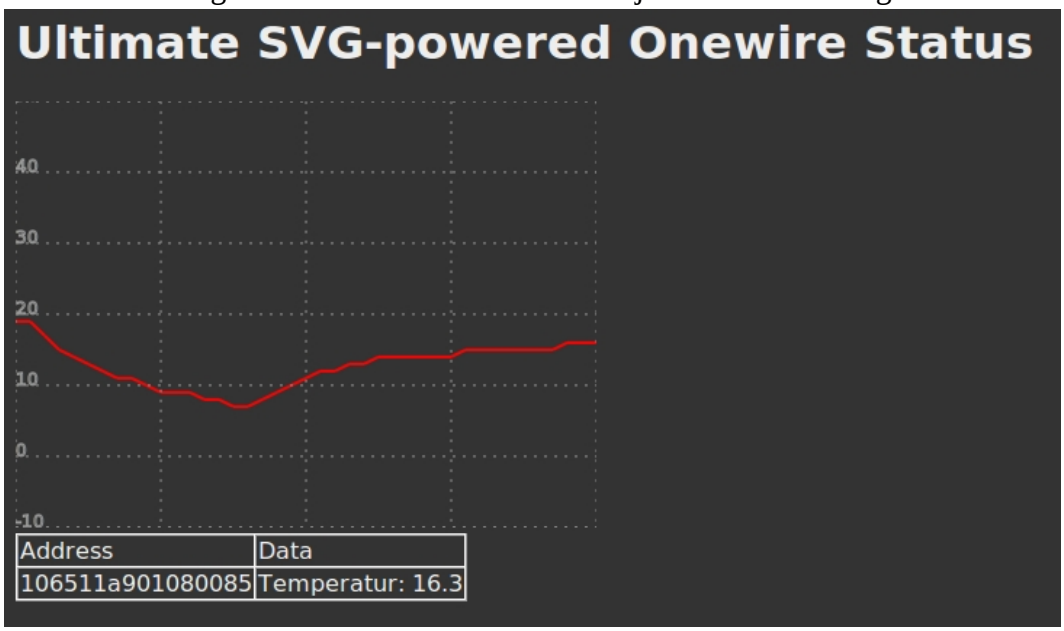

Abb. 4.2.4: Temperaturkurve beim Lüften

4.3 Programmieren der Steuersoftware

4.3a Grundlegende Struktur

Das Programm zum Steuern der Platine soll in PHP programmiert werden. Da PHP eine Scriptsprache ist, wird das Programm im Folgenden öfter mit dem Begriff „avr-net-io Steuerscript“ bezeichnet, der auch als Name für das Programm verwendet werden soll.
Die Oberfläche des Programms soll in vier Bereiche eingeteilt sein: Konfiguration (des Steuerscripts), Befehle, Ausgabe, Erweiterungen, wobei der Bereich „Befehle“ zusätzlich in die in Tabelle 4.2.2 genannten Unterbereiche geordnet sein soll. Bis auf die Konfiguration, die ganz oben auf der PHP-Seite stehen wird, werden die anderen Bereiche mittels einer HTML-Tabelle angezeigt. Dabei soll eine Spalte einem Bereich entsprechen. In der linken Spalte stehen alle verfügbaren Befehle, sortiert in die Unterbereiche. In der mittleren bzw. zweiten Spalte soll, wenn ein Befehl ausgeführt wurde, die Ausgabe und auch noch einmal der ausgeführte Befehl angezeigt werden. In der rechten bzw. in weiteren Spalten werden Erweiterungen angezeigt, zum Beispiel eine Temperaturkurve. Im Folgenden werden die einzelnen Bereiche des Programms analysiert. Das fertige Steuerscript wird in Abbildung 4.3.1 gezeigt.

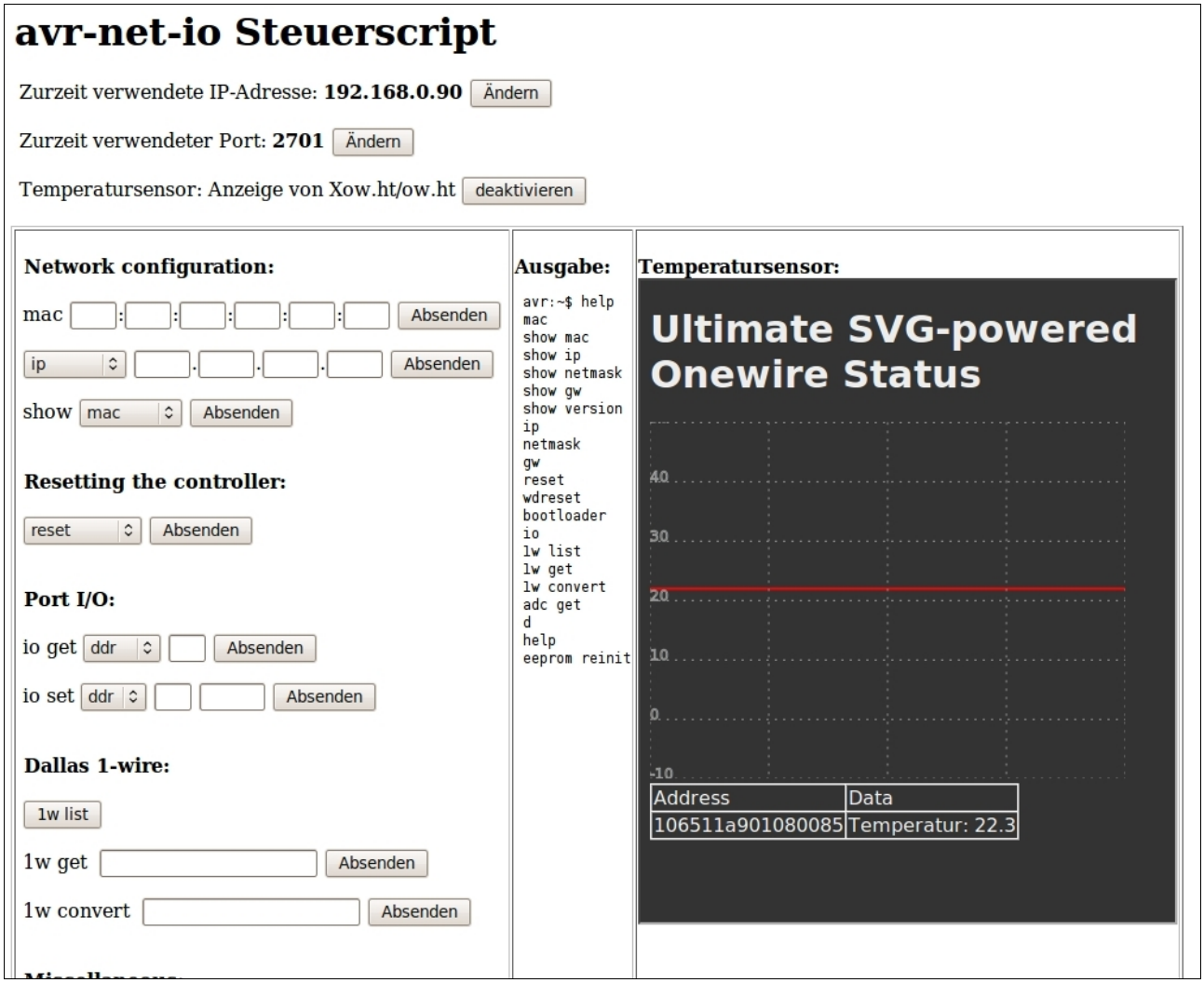

Abb. 4.3.1: Das fertige avr-net-io Steuerscript

Bevor dieses aber benutzt werden kann, muss zuerst noch ein Webserver mit PHP eingerichtet werden. Unter Ubuntu werden die benötigten Pakete mit

```
sudo apt-get install apache2 php5
```

installiert. Nach einem Neustart des Systems kann das Script mit

```
sudo cp /pfad/zum/script /var/www/avr/index.php
```

in das Verzeichnis `avr` des Webservers kopiert und im Browser mit

```
http://localhost/avr/index.php
```

ausgeführt werden.

4.3b Konfiguration

Die Konfiguration des Scripts soll nur drei Optionen beinhalten. Das sind die IP-Adresse der Platine, der verwendete Port und das Aktivieren bzw. gegebenenfalls Deaktivieren einer Erweiterung. Die Daten werden von PHP in Textdateien gespeichert, um auch bei einer späteren Benutzung dieselbe Konfiguration verwenden zu können. Sind die entsprechenden Textdateien nicht vorhanden, wurde das Script auch noch nicht konfiguriert und zeigt eine Eingabe für die Daten an (siehe Abbildung 4.3.2). Dies passiert auch, wenn man auf den Button „Ändern“ klickt. Wenn aber bereits Informationen gespeichert wurden, wird statt einem Eingabefeld der festgelegte Wert angezeigt (siehe Abbildung 4.3.1). Hier wird eine if-Anweisung verwendet, wie auch an vielen anderen Stellen im Programm. Allerdings soll der eigentliche Programmcode hier nicht behandelt werden, da er zu spezifisch ist (Für den Code, siehe Anhang). Für die Eingabe werden, wie bei allen Befehlen auch, html-Formulare verwendet. Mit diesen werden die eingegebenen Daten an das PHP-Script geleitet, welches sie verarbeitet, also zum Beispiel in einer Textdatei speichert, und die Seite des Steuerscripts aktualisiert, damit die jeweilige Ausgabe angezeigt wird. Sollte ein Befehl ausgeführt oder eine Erweiterung aktiviert werden, ohne dass die Konfiguration vorher vorgenommen wurde, so gibt das Programm eine Fehlermeldung und die Aufforderung, IP-Adresse und Port zu bestimmen, aus.

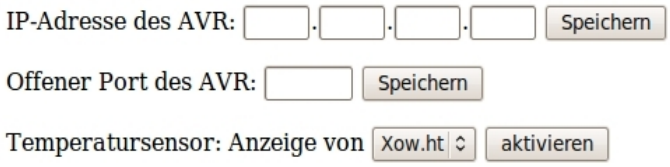

Abb. 4.3.2: Konfiguration

Als Beispiel wird der Port gespeichert: Der Benutzer gibt eine Zahl, vorzugsweise den Standardport `2701` ein und klickt auf den Button „Speichern“. Dadurch wird das Script neu aufgerufen, allerdings mit zwei Übergabewerten:

```
http://localhost/avr/index.php?action=saveport&port=2701
```

Diese wurden in dem Formular für den Port definiert. Das Steuerscript erkennt durch `action=saveport`, dass der Port gespeichert werden soll. Unabhängig davon, ob bereits ein Port konfiguriert wurde, öffnet es die Datei `savedport` . Ist diese nicht vorhanden, wird sie erstellt, andernfalls überschrieben. In diese Datei schreibt das Programm den übergebenen Wert (in diesem Fall `2701`) und schließt die Datei. Anschließend wird die Seite automatisch aktualisiert und die neue Konfiguration wird aus der Datei gelesen und angezeigt.

4.3c Befehle

In der linken Spalte der HTML-Tabelle befinden sich die Befehle, die nach den in Tabelle 4.2.2 genannten Bereichen sortiert sind. Je nach Befehl werden Eingabefelder oder Auswahllisten angezeigt. Die Eingabefelder sind speziell auf den jeweiligen Befehl abgestimmt, sodass klar erkennbar ist, welcher Wert erwartet wird. Hat ein Befehl keine weiteren Optionen (wie zum

Beispiel `help` oder `adc get`), so wird nur ein Button dargestellt (siehe Abbildung 4.3.1 links). Ähnlich wie im Bereich Konfiguration werden Formulare benutzt, um Daten an das Steuerscript zu leiten. Hier benutzt das PHP-Programm die Funktion `passthru()` , mit der es möglich ist, direkte Systembefehle auszuführen. In diesem Fall wird die in 4.2 genannte Möglichkeit benutzt, einen Befehl per Shell an die Platine zu senden. Das Steuerscript ruft dazu die Konfigurationsdaten aus den Textdateien ab und leitet die Ausgabe von `echo 'befehl' | netcat -q 1 IP-Adresse Port` in eine Textdatei um, da die Ausgabe eines Systembefehls in PHP nicht direkt angezeigt werden kann. Hierbei muss beachtet werden, dass das Steuerscript nicht gegen manipulierende Eingaben geschützt ist. Es sollte daher nicht öffentlich verfügbar gemacht werden, da ein Angreifer leicht die durch das System ausgeführten Befehle ausnutzen und sich Zugang zu diesem verschaffen könnte. Allerdings sind die meisten Eingabefelder durch ein Zeichenlimit beschränkt, was das Risiko schwächt.
Als Beispiel wird der Befehl `show mac` ausgeführt. Durch das Formular wird das Script mit zwei Übergabewerten aufgerufen:

```
http://localhost/avr/index.php?befehl=show&n1=mac
```

Das Script erkennt durch `befehl=show`, dass der Befehl `show` mit dem in `n1` gespeicherten Übergabewert `mac` ausgeführt werden soll. Das Script liest IP-Adresse und Port aus den Textdateien aus und leitet die Ausgabe von

```
echo 'show mac' | netcat -q 1 192.168.0.90 2701
```

in die temporäre Textdatei `temp` um (speichert diese in der Textdatei), da die Ausgabe einer `passthru()` Funktion, also eines Systembefehls, nicht direkt durch PHP wiedergegeben werden kann.

4.3d Ausgabe

Eine Ausgabe erfolgt nur unmittelbar nach dem Ausführen eines Befehls, was an das Script gemeldet wird. Ist dies der Fall, liest das Programm den Inhalt der im vorherigen Schritt beschriebenen Textdatei `temp` aus. Dann zeigt es diesen, nachdem der ausgeführte Befehl noch einmal ausgegeben wird, in der zweiten HTML-Tabellenspalte an. Wurde vorher kein Befehl ausgeführt und ist keine Ausgabe benötigt, wird auch keine Tabellenspalte erstellt.
Als Beispiel wurde gerade der Befehl `show mac` ausgeführt. Die Ausgabe wurde in `temp` gespeichert und das Steuerscript wird mit

```
http://localhost/avr/index.php?action=print&cmd=show mac
```

aufgerufen. Der Übergabewert `action=print` sagt dem Script, dass eine Ausgabe erfolgen muss, folglich liest es den Inhalt von `temp` aus. Der Wert `cmd=show mac` bezeichnet den ausgeführten Befehl. Das Script erstellt eine Tabellenspalte, schreibt den Befehl ähnlich der Gestaltung einer Shell in der Form

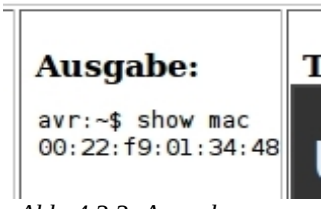

Abb. 4.3.3: Ausgabe von `show mac`

```
avr:~$ show mac
```

hinein und zeigt darunter die gespeicherte und ausgelesene Ausgabe an (siehe Abbildung 4.3.3)

4.3e Erweiterungen

Erweiterungen sollen in jeweils einer weiteren Tabellenspalte dargestellt werden. Die einzige in diesem Steuerscript programmierte Erweiterung ist die Anzeige der in 4.2 genannten Temperaturkurve. Diese ist eine Webanwendung und kann deshalb nicht durch einen Systembefehl abgerufen und angezeigt werden. Hier können eingebettete Frames bzw. Inlineframes (iframes) verwendet werden. In diesen kann man in einem vorgegebenen Bereich den Inhalt einer anderen Seite anzeigen. Da das Diagramm der Temperaturkurve sich nicht vergrößert oder verkleinert, kann ein iframe von der Größe 450x760 Pixel definiert werden. Die Höhe dieses iframes entspricht dabei genau der Befehlsleiste, wodurch das Steuerscript nicht unnötig nach unten wächst, aber die Tabelle unter der Temperaturkurve (siehe Abbildung 4.2.4) noch einige weitere Temperatursensoren aufnehmen kann.
Im Steuerscript wird bei jedem Aufruf überprüft, ob die Textdatei `savedow` existiert. Ist dies nicht der Fall, wird die Temperaturkurve selbstverständlich auch nicht angezeigt. Gibt es die Textdatei, wird sie ausgelesen und dem Script wird übermittelt, ob die grafische Anzeige oder nur die einfache Tabelle aktiviert ist, was man in der Konfiguration (siehe 4.3b) einstellen kann. Der iframe wird erstellt und der Inhalt der Seite angezeigt. Auch beim selbstständigen Aktualisieren des Diagramms treten keine Probleme auf.

4.4 Zusammenfassung

Das angestrebte Ziel der Programmierung einer Steuersoftware wurde erreicht. Von den in 2 genannten Kriterien werden, soweit dies von diesem Standpunkt aus gesagt werden kann, folgende erfüllt:

- grafische Oberfläche
- GPL-Lizenz
- Plattformunabhängigkeit
- Schnittstellen für alle grundlegenden Funktionen der ethersex-Software: Abrufen von Informationen, Konfiguration und Senden von Befehlen
- einfache Möglichkeit, Erweiterungen hinzuzufügen (Beispiel: Temperaturkurven von Sensoren, etc.)

Die Punkte der einfachen Konfiguration, der transparenten Bedienung und des verständlichen Programmcodes müssen von außenstehenden Betrachtern beurteilt werden, wobei beim Erstellen des avr-net-io Steuerscripts aber versucht wurde, auch auf diese Aspekte zu achten. Nach der geplanten Veröffentlichung im Internet wird sich wohl auch darüber mehr sagen lassen können. Das Script stellt aber, wie in der anfangs formulierten Hypothese, eine Erleichterung der Bedienung des avr-net-io dar.

5. Diskussion

In Anbetracht der Geschwindigkeit, mit der sich die Technik allgemein und auch der Bereich erneuerbare Energien weiterentwickelt, lässt sich sagen, dass das Programmieren eines Steuerscripts im Bezug auf die gesteigerte Effizienz von privaten Energieanlagen durch den avr-net-io wohl nicht auf längere Zeit sinnvoll ist, da es sicher bald oder schon jetzt besser automatisierte Systeme zu kaufen gibt. Wer aber eventuell Geld sparen möchte und technisch versierter ist, kann sich mit dem avr-net-io auseinandersetzen.
Aus einer anderen Warte ist das Steuerscript aber interessant: Für das automatische Sammeln von Messdaten für Forschungszwecke eignet sich der avr-net-io sehr gut, da beliebig viele unterschiedliche Sensoren zu einem System aufgebaut werden können. Besonders in der Physik, wo man aus den bereits vorhandenen Geräten oft nicht das benötigte zusammenstellen kann, könnte die Platine nützlich sein, da für komplexe Versuche eine Fülle an Sensoren und Motoren verwendet werden kann, unter anderem Temperatursensoren, Lichtschranken, Infrarotsensoren und ähnliches. Um also das Erstellen eines Versuchsaufbaus zu erleichtern, kann das avr-net-io Steuerscript verwendet werden. Aber auch im privaten Bereich kann die Platine noch Anwendung finden, um zum Beispiel über das Internet vom Arbeitsplatz oder anderswo Temperaturen in und um das Haus zu überprüfen und damit einen Ofen oder eine Zisterne mit beheiztem Wassertank zu kontrollieren. Hier stehen dem Anwender viele Möglichkeiten offen.
Das Script soll nach dem Wettbewerb Jugend Forscht im Internet veröffentlicht werden. Das Programm wird aber auf keinen Fall vermarktet, sondern kostenlos unter der GPL-Lizenz angeboten. Das bedeutet, dass jeder das Steuerscript benutzen und sogar selbst weiterentwickeln kann, aber den Quellcode der Weiterentwicklung freigeben muss.

Anhang

Literaturverzeichnis:

Anmerkung: Die meisten Internetseiten wurden bereits früher eingesehen, aber zum Erstellen der schriftlichen Arbeit erneut abgerufen, um das Datum zu notieren und eventuelle Änderungen festzustellen.

http://cadmaniac.org/projectMain.php?projectName=kontrollerlab: 20.12.2009, Martin Strasser, kontrollerlab

http://www.energie2null.de/files/energie2null.de/statistik/Photovoltaik-installierte-gesamtleistung.jpg: 04.12.2009, Bastian Hammer, installierte Photovoltaik-Anlagen 2000-2007

http://www.ethersex.de/index.php/Dallas_1-wire_Bus: 13.12.2009, Jochen Roessner , Sensoren mit ethersex

http://www.mikrocontroller.net/topic/109988: 01.12.2009, Andreas Schwarz, Diskussion über avr-net-io

http://www.pollin.de/ , darunter:
http://www.pollin.de/shop/dt/MTQ5OTgxOTk-/Bausaetze/Diverse/Bausatz_AVR_NET_IO.html
http://www.pollin.de/shop/dt/NDQ5OTgxOTk-/Bausaetze/Diverse/Bausatz_SUB_D_Anschlussplatine.html
http://www.pollin.de/shop/dt/MzE5OTQ2OTk-/Stromversorgung/Netzgeraete/Steckernetzgeraete/Universal_Steckernetzteil.html
http://www.pollin.de/shop/dt/NTg5OTE4OTk-/Bauelemente/Aktiv/Sensoren_Peltier_Elemente/Temperatur_Sensor_DS18S20_.html
http://www.pollin.de/shop/dt/NTYxOTU0OTk-/Computer_und_Zubehoer/Hardware/Kabel_Stecker_Adapter/CAT_5_Patchkabel.html
http://www.pollin.de/shop/dt/OTY1OTM0OTk-/HiFi_Car_HiFi_Video_TV/HiFi/Kabel/.html
27.11.2009, Pollin Electronic GmbH, benötigte Bestandteile für avr-net-io

http://www.roboternetz.de/phpBB2/viewtopic.php?p=402368: 16.12.2009, Frank Brall, ISP-Kabel

http://www.samurai1967.dyndns.org/avr-net-io.html: 22.12.2009, Oliver Schlenker, avr-net-io Steuerprogramm

http://http://de.wikipedia.org/wiki/Reduced_Instruction_Set_Computing: 07.12.2009, Wikimedia Foundation Inc., RISC-Prozessoren

Weiterhin zur allgemeinen Recherche verwendet:

http://www.google.de/
http://www.hackerboard.de/
http://www.mikrocontroller.net/
http://php.net/
http://de.wikipedia.org/