Studienarbeit I

Fernüberwachung und Fernsteuerung über Internet (Android Applikation)

Studiengang Informationstechnik

an der Dualen Hochschule Baden-Württemberg-Stuttgart

von

Hans Joachim Krauch

und

Waldemar Siebert

Kurs TIT09IN

Betreuer Klaus Hartmann

Inhaltsverzeichnis

Aufgabenstellung 3

Warum wir uns für den Net-IO entschieden haben 4

Warum wir uns für Ethersex entschieden haben 5

Beschreibung von Ethersex 6

Zusammenspiel Android-App, Ethersex und Steuercontroller 7

App-Design, Webserver-Design 8

Datenverwaltung 9

SPI-Protokoll 10

Die erste Doku soll so ca. 5 Seiten umfassen 11

[Warum wir uns für den Net-IO entschieden haben 3](#__RefHeading__200_162417683)

[Warum wir uns für Ethersex entschieden haben 4](#__RefHeading__200_162417683)

[Beschreibung von Ethersex 5](#__RefHeading__200_162417683)

[Zusammenspiel Android-App, Ethersex und Steuercontroller 6](#__RefHeading__200_162417683)

[App-Design, Webserver-Design 7](#__RefHeading__200_162417683)

[Datenverwaltung 8](#__RefHeading__200_162417683)

[SPI-Protokoll 9](#__RefHeading__200_162417683)

[Die erste Doku soll so ca. 5 Seiten umfassen 10](#__RefHeading__200_162417683)

# Aufgabenstellung

Thema: Fernüberwachung eines Mikrokontrollers

Quellen:

* AVLFreaks
* U. Radig
* [http://www.androidpit.de/de/android/wiki/view/Android\_Anf%C3%A4nger\_Workshop](http://www.androidpit.de/de/android/wiki/view/Android_Anfänger_Workshop)
* [http://www.ethersex.de](http://www.ethersex.de/)

Beispielvorgehen:

1. Kontroller + AVR Architektur + Code verstehen
2. Kontroller wählen und kaufen (40Pin)
3. TCP/IP Stack implementieren  AVR Webserver
4. Android App: feste Felderanzahl, Messwerte sind irrelevant
5. Daten kommen von einem zweite Kontroller über SPI
6. Soll über das Internet ansprechbar sein (Auslesen und Schalter)
7. Schalter soll dargestellt werden, erst wenn auch umgeschaltet wurde soll die Stellung verändert werden

Ziel der Dokumentation:

* Ist eher für die Wartung  Modulbeschreibung, Aufbau, usw.
* Bedienungsanleitung
* Installationsanleitung

# Warum wir uns für den Net-IO entschieden haben

Für die Umsetzung eines Mikrokontrollers, der mit einem Ethernet Controller kommuniziert kamen mehrere Möglichkeiten in Frage. Bei einer Recherche wurde ermittelt, dass es bereits ähnliche Projekte gibt, sodass hierfür entweder die Dokumentation von U. Radig hinzugezogen werden konnte, oder aber der kommerzielle Selbsbaukasten NET-IO von Pollin, der sowohl einen Ethernet Controller und Anschluss, als auch einen RS232 Controller und Anschluss hat.

Hauptgrund für die Verwendung des NET-IO ist der gute Support durch die verwendete Ethersex-Firmware. Zudem ist der Bausatz recht güngstig und lässt sich mit etwas Lötkenntnissen schnell aufbauen. Desweiteren besitzt das NET-IO eine ISP-Schnittstelle, mit der der verwendete Controller direkt auf dem Board programmiert werden kann.

# Warum wir uns für Ethersex entschieden haben

Bei der Recherche nach einer geeigneten Firmware für den Microcontroller, sind wir auf das „Ethersex“-Projekt gestoßen. Ethersex ist eine Firmware, welche für die Verwendung auf AVR 8-Bit Microcontroller mit Netzwerkanschluss zugeschnitten ist.

Das Ethersex-Projekt unterstützt mehrere Boards darunter das AVR NET-IO von Pollin, für welches wir uns entschieden haben.

Die Firmware zeichnet sich durch einen bereits implementierten TCP/IP-Stack (IPv4 & IPv6) und zahlreiche weitere Features, wie z.B. ein HTTP-Server, aus.

Auf dem Microcontroller sorgt Ethersex für die Kommunikation mit dem Ethernet-Controller und die Verarbeitung der IP-Pakete. Ethersex erlaubt es außerdem, was für unser Vorhaben äuserst nützlich ist, TCP-Befehle (ECMD) an den Microcontroller zu schicken. So kann bspw. Der Hostname des Microcontrollers an einem PC abgefragt werden. Näheres zu Ethersex siehe <http://old.ethersex.de/index.php/Ethersex-Artikel>.

# Zusammenspiel Android-App, Ethersex-Controller und Steuercontroller

Per Android-App wird ein bestimmter TCP-Befehl an den am Netz angeschlossenen Ethersex-Controller gesendet. Der Ethersex-Controller reagiert auf den Befehl und empfängt die übergebenen Parameter. Ohne diese zu verarbeiten, leitet er diese an den über SPI angeschlossenen Steuercontroller weiter.

Im Steuercontroller wird der Befehl verarbeitet und der Rückgabewert anschließend wieder über die SPI-Schnittstelle dem Ethersex-Controller mitgeteilt. Dieser packt diesen in ein bzw. mehrere IP-Pakete und sendet diese dem Android-App als Antwort zurück.

Der Ethersex-Controller verhält sich im Grunde wie ein Proxy-Server: Er empfängt einen Request und leitet diesen an den Steuercontroller weiter. Die Antwort vom Steuercontroller wird ebenso an die Android-App weitergeleitet.

@WALDE: In MS Word die Abbildung 6.1 aus dem Linkohr sein Script (S. 18) als Smart-Art darstellen. Client=Android-App, Proxy=Ethersex-Controller, Server=Steuercontroller

# Stand der Dinge

**Ethersex**:

Das AVR NET-IO wurde erfolgreich aufgebaut und mit einem ATMEGA 644 bestückt (ausreichend Speicherkapazität). Die Ethersex-Firmware wurde angepasst, kompiliert und auf Microcontroller übertragen. Mit dem Webbrowser ist es möglich den Webserver des Ethersex-Controllers aufzurufen. ECMD-Befehle (Ethersex-Commands) konnten erfolgreich über TCP an den Microcontroller gesendet werden.

**Android-App:**

**@Walde**

# App-Design, Webserver-Design

1. Textwert dann Messwert dann Einheit als eine Zeile, **in Folge geschickt**.

ASCII Zeile wird ausgelesen, Spalte wird übersprungen bei def. Sonderzeichen.

# Datenverwaltung

Wie die Daten geholt werden, ob Zwischenspeichern auf dem Controller und wann von dem andern Controller ler geholt werden will. Nur bei Zugriff, oder mit Aktualisierungstaste.

Bei Zugriff und wenn auf Aktualiserungs-Taste gedrückt wird.

→ Kein Zwischenspeichern auf dem Controller

# SPI-Protokoll

festlegen, zum Steuercontroller bzw. Android-App (SCPI?)

Von dir frei wählbar, muss allerdings noch net in die erste Doku m.M. nach

# Die erste Doku soll so ca. 5 Seiten umfassen

Es soll folgendes enthalten sein:

Ein Konzept mit 1-2 Möglichkeiten für bestimmte Entscheidungen, die wir getroffen haben.

1. Hardwareauswahl
2. Wie die Oberfläche der App/Webserver gestaltet werden soll
3. Wie die Kommunikation gestaltet werden soll

Es soll grob wiedergeben, was bisher geleistet wurde.

Im 6. Semester soll eine technische Beschreibung folgen.