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

1 Auswahl des Mikrokontrollers

Für die Umsetzung eines Mikrokontrollers, der mit einem Ethernet Controller kommuniziert kamen mehrere Möglichkeiten in Frage. Zum einen war es möglich alles neu zu entwickeln und hierfür einen Mikrokontroller nach Vorgaben auszuwählen, oder aber auf bereits bestehende Projekte aufzubauen. Die Vorgaben hierfür waren, einen Mikrokontroller mit 40 Beinen zu nutzen, der mit einem Ethernet Controller kommunizieren soll und genügend Platz für einen Webserver aufweist. Durch eine Recherche sind zwei Projekte aufgefallen. Bei dem ersten Projekt handelt es sich um einen Webserver von Ulrich Radig. Dies ist gut dokumentiert und enthält einen Schaltplan mit einer Stückliste der benötigten Bauteile. Beim zweiten Projekt handelt es sich um den kommerziellen Bausatz AVR-NET-IO von Pollin, welcher inklusive aller Bauteile und einer kleinen Software bestellt werden kann. Eine Bestellung als Fertigmodul ist auch möglich, kostet aber entsprechend mehr.

Die Auswahl fiel deshalb auf den AVR-NET-IO von Pollin als Bausatz. Der Hauptgrund für die Verwendung des AVR-NET-IO ist der gute Support durch die frei verwendbare Ethersex-Firmware. Zudem ist der Bausatz günstig zu haben und lässt sich mit den nötigen Lötkenntnissen gut zusammenbauen. Für die Kommunikation zwischen dem Mikrokontroller und dem Ethernet Controller wird SPI verwendet. Des Weiteren besitzt das NET-IO eine ISP-Schnittstelle, die es erlaubt den verwendeten Controller direkt auf dem Board zu programmieren.

2 AVR-NET-IO

Der AVR-NET-IO, wie in Abbildung 1 zu sehen, besteht aus einer Ethernet-Platine mit einem Sockel für einen 40 Pin Mikrokontroller, einem Netzwerkcontroller ENC28J60 und einem Seriell Controller MAX232. Auf der Platine befinden sich 8 digitale Ausgänge, 4 digitale Eingänge und 4 ADC-Eingänge, die über TCP/IP abgerufen/geschaltet werden können. Weiterhin ist eine RJ45 Netzwerkbuchse, ein 9-poliger Sub-D Anschluss für RS232, eine ISP-Schnittstelle und alle weiteren benötigten Komponenten vorhanden. Die Platine muss mit 9V Betriebsspannung versorgt werden und hat eine Stromaufnahme von etwa 190mA. Die digitalen Ein- und Ausgänge sind mit 0V und 5V Pegeln als Richtwert angegeben. Die Platine hat mit den verwendeten Komponenten die Maße 108mm Länge, 76mm Breite und 22mm Höhe.

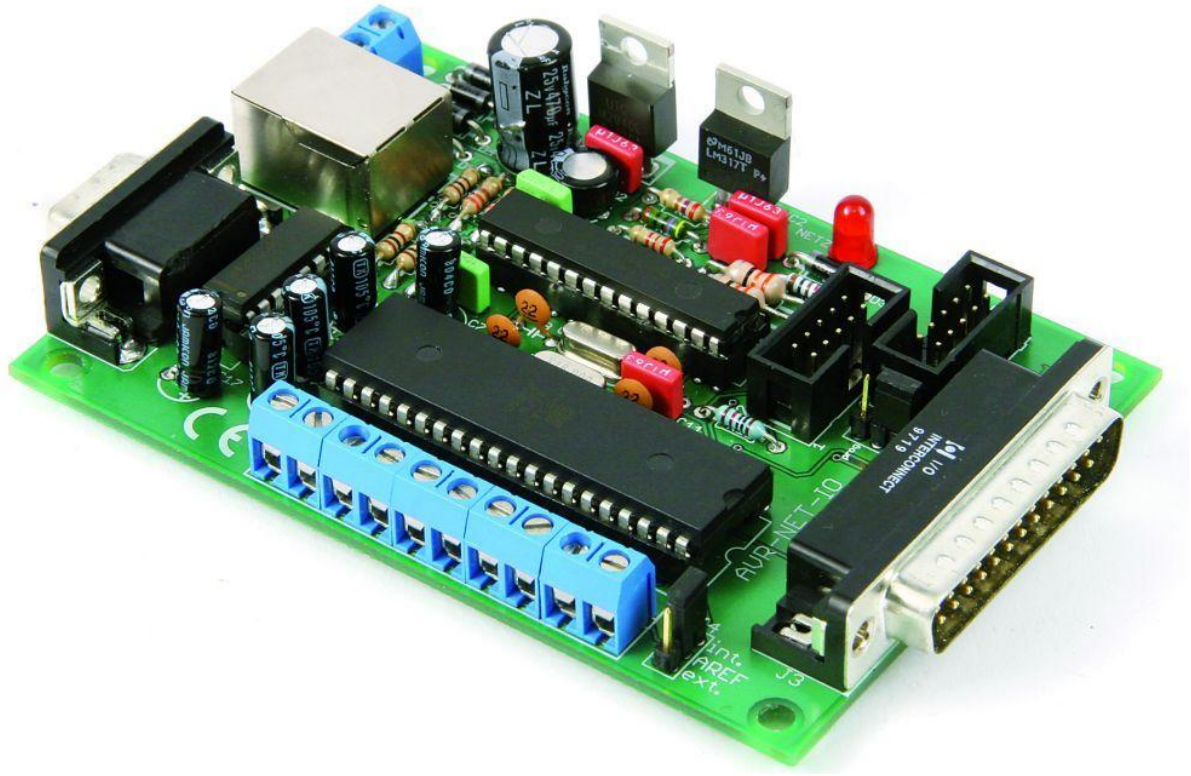

Abbildung 1: AVR-NET-IO von Pollin

3 Firmware

Die Recherche nach einer geeigneten Firmware für den Mikrokontroller brachte das „Ethersex“-Projekt als eine sehr interessante Möglichkeit hervor. Ethersex ist eine Firmware, welche für die Verwendung auf AVR 8-Bit Mikrokontroller mit Netzwerkanschluss zugeschnitten ist. Das Ethersex-Projekt unterstützt mehrere Boards, darunter das AVR-NET-IO von Pollin. Besonders zeichnet sich die Firmware durch einen bereits implementierten TCP/IP-Stack (IPv4 & IPv6) und zahlreiche weitere Features, wie z.B. ein HTTP-Server, aus.
Auf dem Mikrokontroller sorgt Ethersex für die Kommunikation mit dem Ethernet-Controller und die Verarbeitung der IP-Pakete. Ethersex erlaubt es außerdem TCP-Befehle (ECMD) an den Mikrokontroller zu schicken. So kann beispielsweise der Hostname des Mikrokontrollers an einem PC abgefragt werden.

4 Kommunikationskonzept

Das Kommunikationskonzept besteht aus mehreren Komponenten. Es muss eine Kommunikation zwischen einem Android Smartphone und dem Mikrokontroller aufgebaut werden. Hierfür wird per Android-App 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, wie Abbildung 2 zu sehen, 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.

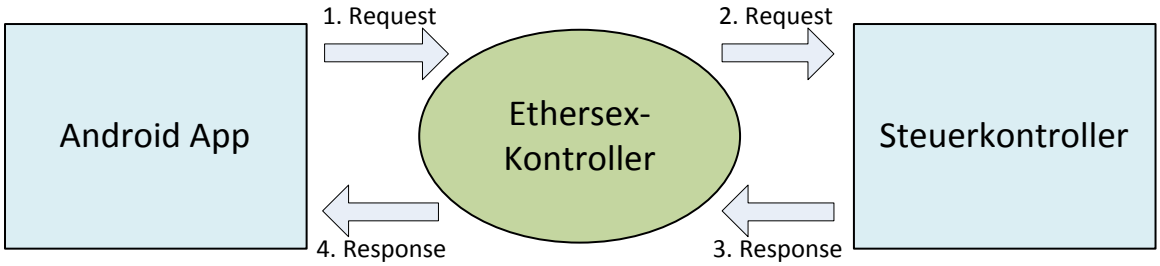

Abbildung 2: Client-Proxy-Server Konzept

5 Stand der Dinge

Ethersex:

Das AVR NET-IO wurde erfolgreich aufgebaut und mit einem ATmega644 bestückt (ausreichend Speicherkapazität). Die Ethersex-Firmware wurde angepasst, kompiliert und auf den Mikrokontroller übertragen. Mit dem Webbrowser ist es möglich den Webserver des Ethersex-Controllers aufzurufen. ECMD-Befehle (Ethersex-Commands) konnten erfolgreich über TCP an den Mikrokontroller gesendet werden.

Android-App:

Das Design ist im folgen Kapitel zu sehen. Jedoch funktioniert die Kommunikation über TCP noch nicht.

6 App-Design

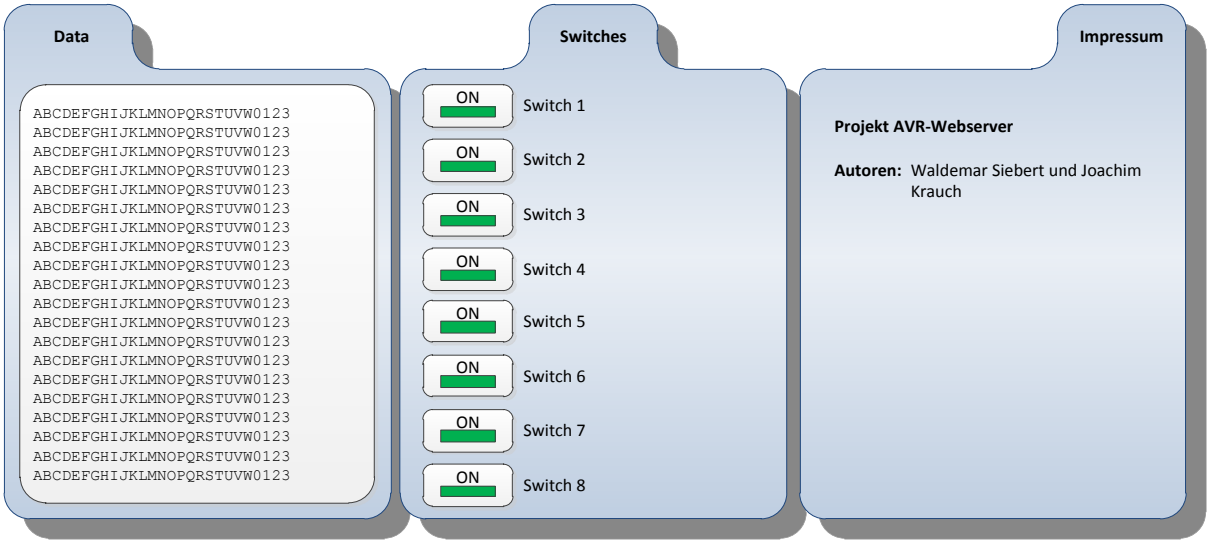

Abbildung 3: Konzept des App Designs

Abbildung 3 zeigt das App Design für Android. Auf dem ersten Tab sind die Messwerte. Auf dem zweiten Tab sind die Schalter und auf dem letzten Tab sieht man das Impressum. Die genaue Auslegung des Designs kann sich noch ändern.