Diplomarbeit: netcon

Lipp, Pietryka

Zusammenfassung

Abstract

Danksagung

Vorwort

Sie wollen Umweltgrößen an mehreren Standorten (aus der Ferne) überwachen? An den Standorten ist lediglich ein gemeinsames Ethernet-Netzwerk verfügbar und für den Aufbau eines eigenen Netzes fehlt das Budget?

Im Rahmen dieser Diplomarbeit, wurde mit **netcon** ein quelloffenes, flexibles Messsystem auf Ethernet-Basis geschaffen.[1] Dabei wurde darauf Rücksicht genommen, erfahrene Endanwender, Unternehmen und Entwickler gleichermaßen zu bedienen. Je nach Anwendungsfall und Vorkenntnissen sollten Sie in der Lage sein ihr eigenes Messsystem aufzubauen.

Im ersten Kapitel folgt ein Überblick über die **allgemeine Konzeption** - Systemvoraussetzungen, Aufbau, Schnittstellen und grundsätzliche Funktionsweise. Ein zweites Kapitel gibt eine Einführung in **grundlegende Begriffe**, die für ein tieferes Verständnis der Entwicklungen erforderlich sind. Danach folgt das große Kapitel, **Hardware**, das den Aufbau und die Funktionsweise der Messmodule behandelt, sowie in die genaue Verwendung der Schnittstellen und Protokolle auf der Hardwareseite einführt. Das letzte Kapitel, **Software** beschreibt die Verwaltungsschicht und dessen Interfaces für die Anzeige der Moduldaten.

[1] Maßnahmen für die spätere Implementierung eines Aktornetzwerks wurden getroffen. Diese wurde aber aus Zeitgründen nicht durchgeführt.

Inhaltsverzeichnis

1 Überblick [Lipp]

Netcon ist zum einen ein Messsystem zur Einbindung in ein bestehendes Ethernet-Netzwerk, zum anderen aber auch das Ziel flexible Werkzeuge für die Erstellung eines solchen Systems bereitzustellen. Dabei stehen alle Entwicklungen unter OpenSource.

Da das System als Übertragungsmedium die Ethernet-Schnittstelle mit der TCP/IP-Protokollschicht verwendet, ist Echtzeitverhalten nicht garantiert. Dadurch ist es nur für unzeitkritische Aufgaben geeignet.

1.1 Zielgruppen

Erfahrene Endandwender, genauso Entwickler sollten mit netcon in der Lage sein, ein Messsystem zu realisieren. Es wurden hardware- und softwareseitig einfache Schnittstellen geschaffen um je nach Wunsch und vorherrschenden Kenntnissen eigene Anwendungen zu erstellen.

Der Anwender kann sich entscheiden, entweder entwickelt er auf Basis der Spezifikationen die netzwerkfähigen Module selbst, oder aber er verwendet die im Rahmen dieser Diplomarbeit gewählten Mikrocontroller-Systeme. Dazu stellt netcon die entwickelte Firmware zur Verfügung. Weiters besteht für netzwerktechnisch unerfahrene Entwickler die Möglichkeit, ihre Module netzwerkfähig zu machen.

Auch auf der Softwareseite stehen mehrere Wege offen. Entwickler können eigene Applikationen über die Schnittstelle der Verwaltungsschicht aufsetzen, oder aber auch die entwickelte Weboberfläche zur Anzeige und Steuerung der Module verwenden.

1.2 Anwendungsbeispiele

Netcon sieht in seiner Spezifikation mehrere Typen von Messmodulen vor. Folgende Liste zeigt Anwendungen, die unter anderem mit diesem System verwirklicht werden können:

- Spannungsmessung
- Temperaturmessung
- Zeitmessung

1.3 Systemvoraussetzungen

Das netcon Messsystem wurde für dein Einsatz in einem Ethernet-Netzwerk konzipiert. Dieses muss zumindest über folgende Komponenten verfügen:

- Anschlussmöglichkeiten für die Module (Router/Switch/WLAN)
- DHCP-Server für die IP-Adressvergabe
- **Server** - Javafähige Betriebsumgebung für die Verwaltungsschnittstelle z.B. PC, Embedded System
- **Client** - Anzeigegerät z.B. Smartphone, Computer

Zusätzlich wird gegebenenfalls ein PHP-fähiger Webserver benötigt, um die bereits entwickelte Website verwenden zu können. Die genauen Anforderungen an die Softwareumgebung, sowie die Einrichtung einer Java Runtimte Environment (JRE) und eines Webservers sind im Kapitel Software nachzulesen.

Und nicht zu vergessen sind die wichtigsten Komponenten, die netzwerkfähigen Module. Diese können, wie bereits erwähnt, nach den netcon-Protokollen selbst entwickelt, oder aber auch nach Anleitung erstellt werden. Dazu mehr im nächsten Abschnitt.

1.4 Systemaufbau

Im folgenden sind die zwei grundlegenden netcon-Komponenten inkl. ihrer Schnittstellen beschrieben. Je nachdem wie netcon genutzt werden soll, wird auf weitere Kapitel verwiesen.

1.4.1 Module

Die **Module** sind Hardware, die über Ethernet und TCP/IP erreichbar und abfragbar sind.

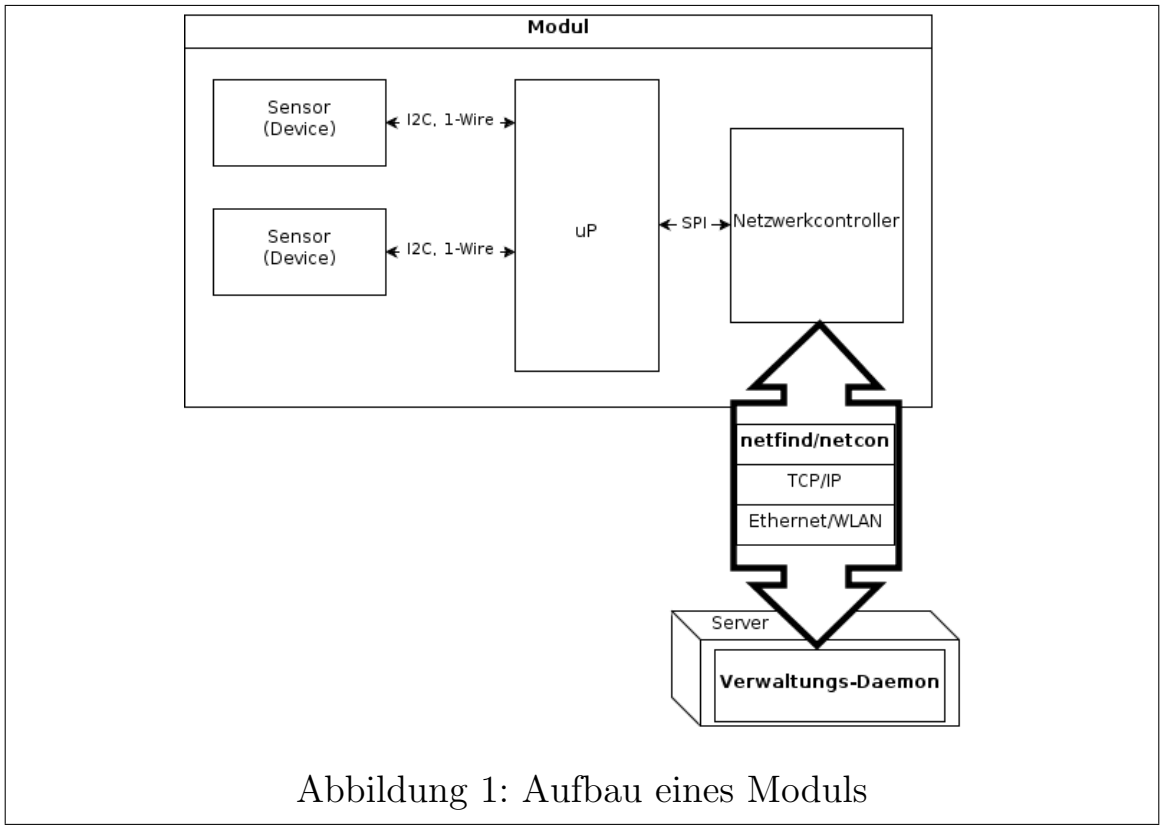

Abbildung 1: Aufbau eines Moduls

Sie vereinen alle benötigten Komponenten - Mess- und Netzwerkeinheit - auf einer Platine (siehe Abb. 1). Hier erfolgt die Übertragung zwischen den Sensoren und dem Mikroprozessor (uP) meist über Schnittstellen, wie I2C oder 1-Wire, während der Netzwerkcontroller per SPI mit dem uP kommuniziert. Über TCP und mit den beiden Protokollen *netfind* und *netcon* erfolgt die Abfrage und Steuerung durch den plattformunabhängigen Verwaltungs-Deamon *netcond*. Näheres dazu im nächsten Abschnitt.

Messmodule umfassen beispielsweise Sensoren für Temperatur, Luftdruck und Zeit. Jeder dieser Sensoren wird von netcon als **Device** bezeichnet und kann mit seiner ID abgefragt werden.

Um den Modulen automatisch eine IP-Adresse zuweisen zu können und damit Konfigurationsarbeit zu ersparen, sollten diese auch das DHCP-Protokoll unterstützen.

Es bestehen grundsätzlich drei Möglichkeiten zur Erstellung von Modulen. Wenn Sie alle erforderlichen Kenntnisse besitzen, um die gesamte Entwicklung selbst zu übernehmen, informieren Sie sich im Kapitel Hardware über den Aufbau der netcon-Protokolle. Sind Sie in der Lage einfache Messmodule ohne Netzwerkfähigkeit zu erstellen, verbinden Sie doch ein zusätzliches Netzwerkmodul (siehe Abb. 2). Diese Möglichkeit erfordert lediglich die Implementierung der Seriellen Schnittstelle (UART). Genauso können die im Rahmen dieser Diplomarbeit konzpierten Module mit ihrer Firmware für die Erstellung eigener Module herangezogen werden. Egal welche Wahl Sie treffen, das Kapitel Hardware unterstützt Sie in allen drei Fällen.

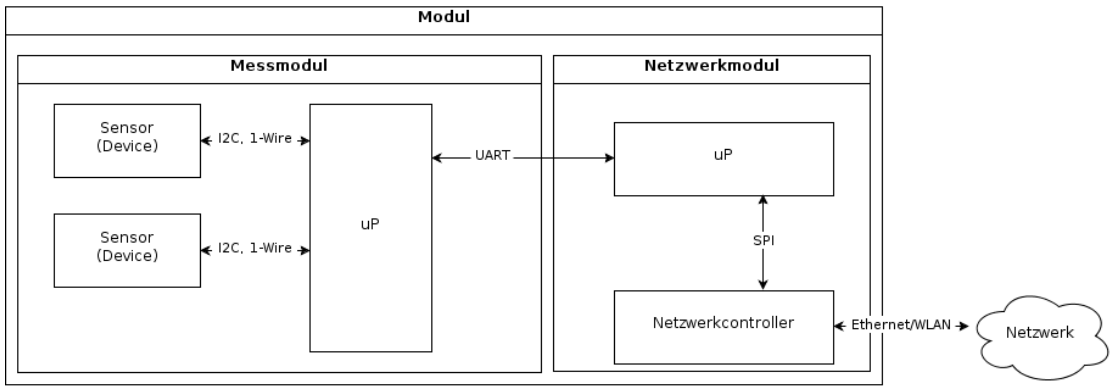

Abbildung 2: Netzwerkfähiges Messmodul mittels LAN-UART Umsetzer

1.4.2 Software

Die Verwaltungsschnittstelle **netcond** ist eine in Java geschriebene Hintergrundanwendung (Daemon), die sich um die Verwaltung der Module kümmert. Wie in Abb. 3 erkennbar können über das netcon Application Interface (netcon API) per TCP die Moduldaten abgefragt werden. Die Anwendung kann beispielsweise eine Website auf einem Webserver oder ein Smartphone-App sein.

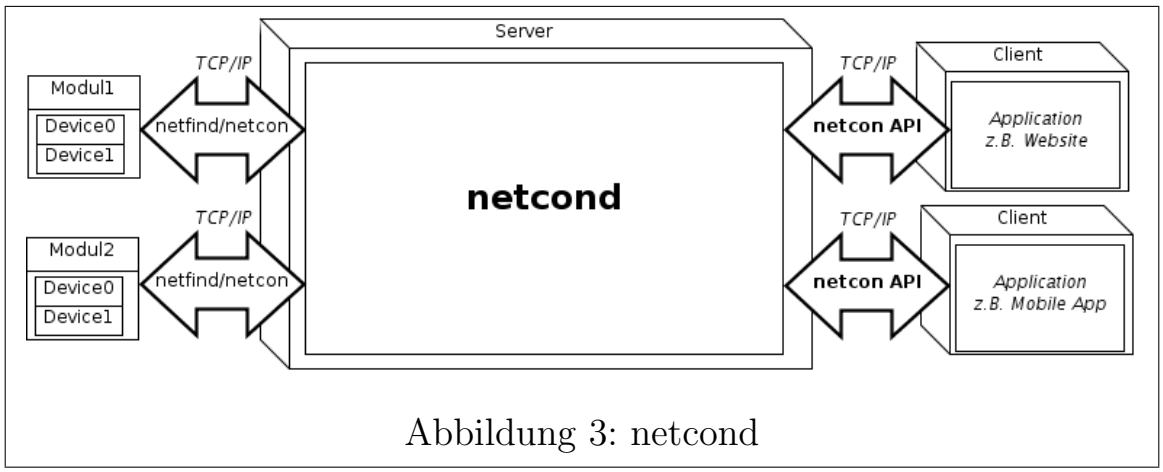

Abbildung 3: netcond

Soll die Anwendung zur Anzeige der Messdaten selbst entwickelt werden, führt das Kapitel Software in die Verwendung der Softwareschnittstelle ein. Sonst kann die bereits entwickelte Website **netcon web** (siehe Abb. 4) verwendet werden. Dazu ist zusätzlich zur JRE ein http-Webserver mit PHP-Unterstützung erforderlich. Deren Installation und Konfiguration ist im Kapitel Software erklärt.

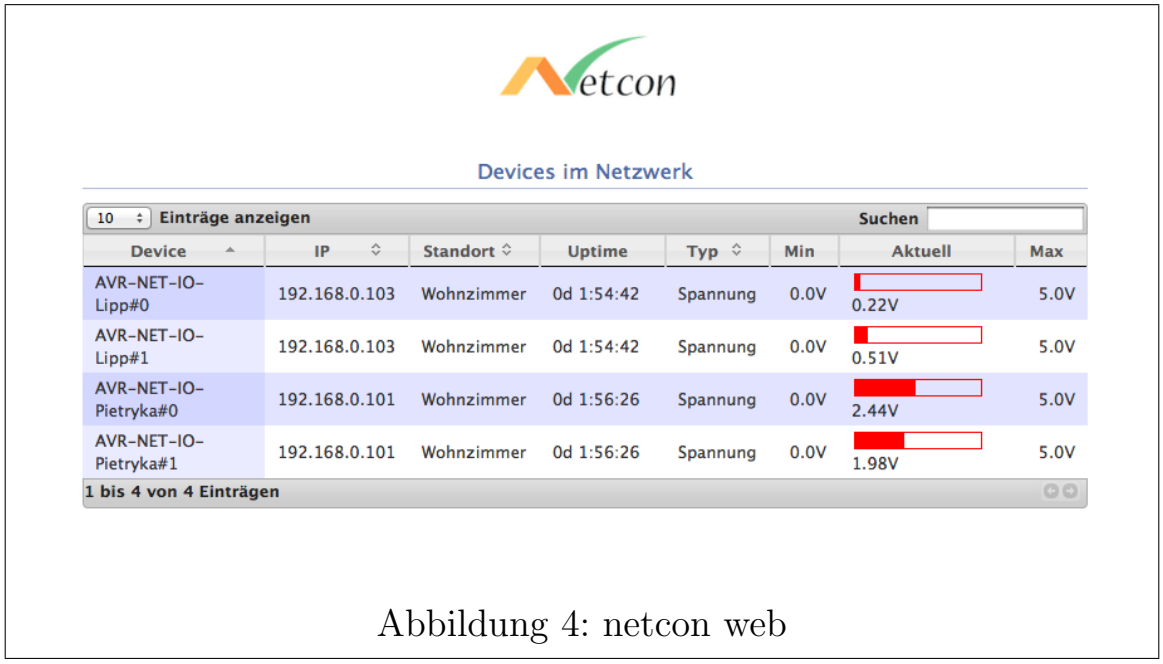

Abbildung 4: netcon web

1.5 Funktionsweise

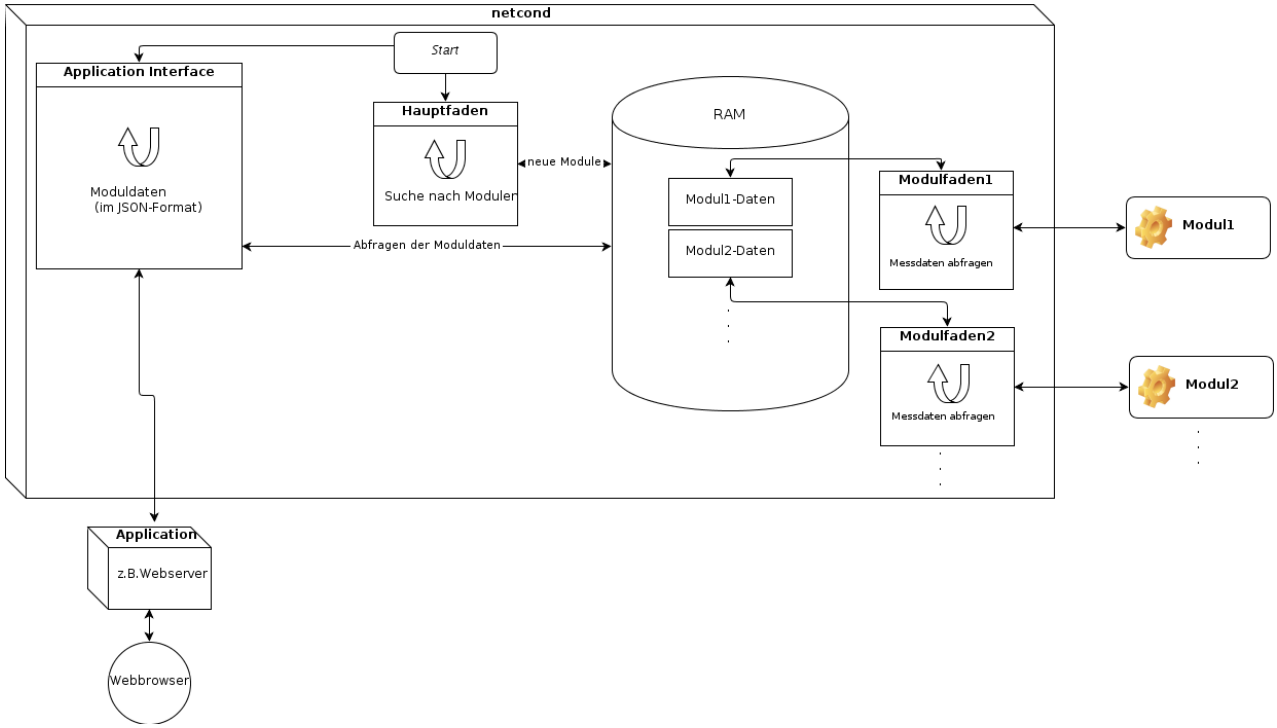

Abbildung 5: netcon - Funktionsprinzip

Der Java-Daemon netcond sucht nach dem Start alle paar Sekunden das Netzwerk nach verbunden Modulen ab und halt diese in einer Liste. Für jedes dieser Module wird ein neuer Programmfaden erzeugt, der ständig Messdaten abfragt und diese speichert. Zusätzlich startet der Daemon einen weiteren Subprozess, die Schnittstelle, über die eine Anwendung - in diesem Fall ein Webserver - die Daten abfragen kann. Verbindet sich ein Webbrowser zu diesem Webserver, weißt ein PHP-Script den Daemon an, die aktuellen Daten zu übermitteln. Diese werden dann auf der Website angezeigt. Dieses Prinzip ist nocheinmal in Abb. 5 grafisch verdeutlicht.

2 Grundlagen [Pietryka]

2.1 OSI-Schichtenmodell

Das OSI-Schichtenmodell ist ein von der ISO im Jahre 1983 standardisiertes Modell, welches als Designgrundlage für Kommunikationsprotokolle in Computernetzen dient. Dabei wird die Kommunikation beim OSI-Modell auf sieben Schichten bzw. Layern aufgeteil. Für jede dieser Schichten sind Anforderungen und Aufgaben definiert, welche von entsprechenden Protokollen realisiert werden müssen. Eine konkrete Umsetzung ist aber nicht vorgegeben, daher gibt es für eine Schicht mehrere in Frage kommenden Protokolle.

2.1.1 Schicht 1: Physical Layer

2.2 Ethernet

2.3 MAC-Adresse

Die MAC-Adresse (Media-Access-Controll-Adresse) ist eine Adresse welche Netzwerkgeräte eindeutig identifiziert, Ethernet Pakete werden mithilfe dieser Adresse adressiert. Diese Adresse ist 48-Bit lang und ist auf der Welt eindeutig für jedes Netzwerkgerät. Die Darstellung erfolgt in Hexadezimaler Darstellung wie 00:80:41:ae:fd:7e oder 00-80-41-ae-fd-7e. Will man seine Netzwerkfähigen Geräte am Markt verkaufen, so benötigt man gültige MAC-Adressen, diese kann man in Blöcken kaufen. Kleinere Unternehmen können Blöcke mit 4096 MAC-Adressen kaufen, größere Unternehmen besitzen die Möglichkeit einen Block mit 16,8 Millionen Adressen zu kaufen.

Entwickelt man aber noch geringere Stückzahlen, so gibt es beispielsweise von der Firma Microchip EEPROMS, welche mit einer bereits vorprogrammierten und natürlich gültigen MAC-Adresse geliefert werden. Eine weitere Möglichkeit

gültige MAC-Adressen während der Entwicklung zu vergeben sind die sogenannten lokal administrierten Adressen, dabei wird im ersten Byte das zweite Bit auf 1 gesetzt. Ist dies der Fall, so handelt es sich um eine lokal administrierte Adresse, der Nachteil hierbei ist natürlich, dass diese Adresse nicht mehr eindeutig ist.

2.3.1 Broadcast

Bei einem sogenannten Broadcast wird im Ethernet-Header eine spezielle MAC-Adresse als Ziel angegeben, nämlich FF:FF:FF:FF:FF:FF, Pakete mit dieser Zieladresse werden an alle Geräte in einem LAN-Netzwerk verschickt, jedoch nicht in ein anderes Netzwerk geroutet.

2.4 IP

3 Hardware [Pietryka]

3.1 Der Ethernet Controller

3.2 Auswahl

Damit ein Mikrocontroller über das Ethernet kommunizieren kann, wird eine entsprechende Hardware benötigt, der sogenannte Ethernet Controller. Ein Ethernet Controller übernimmt dabei die Aufgaben der OSI-Schichten 1(Physical) und 2(Data-Link). Der Controller benötigt zudem einen entsprechend großen Empfangspuffer, um mindestens einen vollwertigen Ethernet-Frame(1542 Byte) aufzunehmen zu können. Dabei standen für 8-Bit Mikrocontroller vorerst zwei verschiedene Bausteine zur Auswahl, einmal der CP2200 von SiLabs, und einmal der ENC28J60 von Microchip. Beide Controller haben, was die Netzwerkkommunikation angeht, so ziemlich die selben Features, der gravierende Unterschied liegt jedoch in der Ansteuerung dieser. Der CP2200 wurde von SiLabs, wie es scheint, nur für die Verwendung mit einem Mikrocontroller vom Typ 8051 entwickelt, die Ansteuerung erfolgt deshalb über einen parallelen Adress-/Datenbus wodurch man mindestens 16 Leitungen und Pins am Mikrocontroller benötigt. Beim ENC28J60 erfolgt die Kommunikation über den SPI-Bus, daher benötigt man nur vier Leitungen(MOSI, MISO, SCK, CS), dadurch hat auch der Netzwerkcontroller selber nur 28 Pins und ist auch im "bastlerfreundlichen"DIP-Gehäuse zu bekommen. Ein anderer Faktor für die Auswahl des ENC28J60 war das Vorhandensein einer günstigen Entwicklungsplatine, es gibt bei Pollin den AVR-NET-IO Bausatz, dieser kostet nur 20 € und enthält alle für die Netzwerkprogrammierung benötigten Komponenten(ATmega32, ENC28J60, RJ-45 Buchse).

3.3 ENC28J60 Beschaltung

Die Aussenbeschaltung benötigt neben einigen Standardbauelementen auch einige 1% Widerstände und einen 1:1 Übertrager, jedoch gib es RJ-45 Buchsen

in denen bereits der Übertrager, sowie die LEDs bereits eingebaut sind.

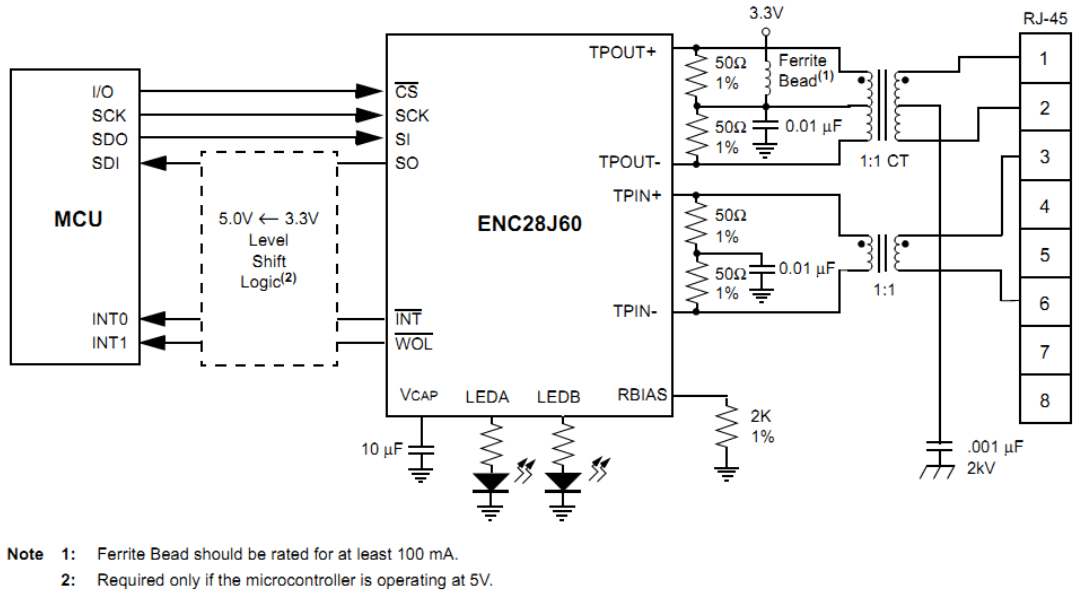

Abbildung 6: Aussenbeschaltung ENC28J60

3.4 ENC28J60 Treibersoftware

Die erste Aufgabe war es eine Bibliothek zu schreiben, welche es erlaubt, mithilfe des ENC28J60, Ethernet Pakete über das Netzwerk zu verschicken. Es waren zwar im Internet verschiedene Programme für den ENC28J60 vorhanden, diese waren aber teilweise schlampig und unschön geschrieben. Am Ende Stand eine Bibliothek zur Verfügung, die auch für andere Projekte genutzt werden kann, diese wurde für den AVR geschrieben und besteht aus zwei Dateien ënc28j60.hünd ënc28j60.c". Zurzeit übernimmt die Bibliothek auch die Initialisierung des SPI-Busses, will man dies verhindern, so muss man die Zeilen 187-197 in der Datei ënc28j60.cëntfernen, weiterhin wird es eventuell nötig sein die Pins für das Chip-Select anzupassen, dazu muss man die Funktionen get_cs() und release_cs() in den Zeilen 24-32 ebenfalls anpassen. Für die Benutzung der Bibliothek stehen dann drei Funktionen zur Verfügung. Die Bibliothek selber finde sich im Ordner "netcon Module/Pollin AVR-NET-IO/enc28j60/".

Der Treiber selbst kann als stabil angesehen werden, es sind in den mehreren Monaten der Arbeit mit diesem keine Fehler aufgefallen, auch wurde das Geräte mehrere Stunden über die Nacht laufen gelassen, mit Erfolg.

3.4.1 void enc28j60_init(const uint8_t *mac_addr)

Initialisiert die Hardware des ENC28J60 mit der angegebenen MAC-Adresse, diese muss als ein Zeiger auf ein Array mit 6 Byte übergeben werden. Zurzeit wird auch die SPI-Hardware eines ATmega32 ebenfalls initialisiert, ist dies nicht gewünscht, so muss der Code entsprechend abgeändert werden.

```
const uint8_t mac_addr[6] = {0x02, 0x00, 0x00, 0x00, 0x00, 0x01};
enc28j60_init(mac_addr);
```

3.4.2 void enc28j60_transmit(const uint8_t *data, uint16_t len)

Sendet ein Ethernet-Paket in das Netzwerk, welches über einen Zeiger auf data referenziert wurde, mit der Länge len.

```
uint8_t buf[512];

strcpy(buf, "Das ist ein ungueltiges Paket");
enc28j60_transmit(buf, strlen(buf));
```

3.4.3 uint16_t enc28j60_receive(uint8_t *data, uint16_t max_len)

Schaut ob ein Paket im Puffer des ENC28J60 angekommen ist, ist dies der Fall, so werden bis zu max_len Bytes in den durch data referenzierten Bereich kopiert. Als Rückgabewert liefert die Funktion die Anzahl der erfolgreich gelesenen Byte. Ist kein neues Paket beim Aufruf der Funktion vorhanden, so ist

der Rückgabewert 0.

```
uint8_t buf[512], len;

while(1 > 0)
{
    len = enc28j60_receive(buf, 512);
    if(len > 1)
    {
        // Neues Paket
    }
}
```

3.4.4 Ethernet-DK Port

Weiterhin, wurde eine Portierung des ENC28J60C Treibers auf das Ethernet-DK von SiLabs durchgeführt. Dies lässt sich mit dem freien C-Compiler SDCC compilieren, die entsprechenden Projektdateien sind im Ordner "netcon Module/Ethernet-DK/enc28j60". Damit dieses Projekt erfolgreich compiliert muss der SDCC Compiler entsprechen in der SiLabs IDE eingestellt konfiguriert werden, zudem muss sowohl dem Linker als auch dem Compiler der Parameter -model-large"übergeben werden.

3.5 CP2200

Auch wenn der CP2200 Ethernet-Controller nicht weiter verwendet wurde, so wurde ebenfalls ein Treiber geschrieben, sowie der uIP Stack auf diesen portiert, diese Projektdateien finden sich im Ordner "netcon Module/Ethernet-DK/cp2200". Über die Stabilität kann keine Auskunft gegeben werden, jedoch hat die Netzwerkkommunikation über eine Stunde hinweg funktioniert. Auf eine weitere Dokumentation wird hier verzichtet, da der Treiber nur als Nebenprodukt verschiedener Experimente anzusehen ist.

4 Software [Lipp]

4.1 Aufgaben

was sind die grundsätzlichen aufgaben der Software, wofür ist sie da?

4.2 Java

warum java?

4.3 Aufbau und Funktionsweise

Aufbau der Software Programmabläufe (Flussdiagramme) Genaue Beschreibung der Schnittstellen

4.4 Installation

4.4.1 JRE

4.4.2 Webserver