Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Bachelorarbeit

Felix Kneisler & Nico Dummer

Sensorik und Telemetrie für ein Brennstoffzellenfahrzeug mit Embedded Linux, TCP/IP und WLAN

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Felix Kneisler & Nico Dummer

Sensorik und Telemetrie für ein Brennstoffzellenfahrzeug mit Embedded Linux, TCP/IP und WLAN

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung
im Studiengang Technische Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer : Prof. Dr. rer. nat. Reinhard Baran
Zweitgutachter : Prof. Dr. rer. nat. Thomas Canzler

Abgegeben am 29. Mai 2008

Felix Kneisler & Nico Dummer

Thema der Bachelorarbeit

Sensorik und Telemetrie für ein Brennstoffzellenfahrzeug mit Embedded Linux, TCP/IP und WLAN

Stichworte

Telemetrie, mobile Systeme, OpenWrt, Linux, TCP/IP, WLAN, GPS, USB, LabVIEW

Kurzzusammenfassung

In dieser Diplomarbeit wird ein Telemetriesystem für die Datenerfassung und drahtlose Überwachung von einem mit einer Wasserstoff Brennstoffzelle betriebenen Fahrzeug entwickelt. Es ist möglich, Daten von einem GPS, einer Brennstoffzelle und Daten von weiteren Sensoren für Temperaturen, Spannungen und Gleichstrom per WLAN zu empfangen. Die Daten können in Realtime von einem Boxenteam mit einer auf LabView basierenden grafischen Oberfläche, ausgewertet werden.

Felix Kneisler & Nico Dummer

Title of the paper

Sensorik and telemetry for a fuel cell driven vehicle with embedded Linux, TCP/IP and WLAN

Keywords

Telemetry, mobile systems, OpenWrt, Linux, TCP/IP, WLAN, GPS, USB, LabVIEW

Abstract

This paper presents the development of a telemetrysystem for data acquisition and wireless monitoring of a fuel-cell driven vehicle. It is possible to get data from a GPS, fuel-cell and data from additional Sensors for temperature, voltage and dc-current through WLAN from the vehicle. The data can be monitored in realtime by a team with a LabView based graphical userinterface.

Danksagung

An dieser Stelle möchten wir all denjenigen danken, die uns bei unserer Arbeit unterstützt haben.

Zuerst möchten wir unseren Betreuer und den Leiter des ECO-Teams nennen. Sie ermöglichten es uns, unsere Bachelorarbeit als Abschluss des Studiums an der HAW-Hamburg anzufertigen.

Der Dank geht an:
Prof. Dr. rer. nat. Reinhard Baran
Prof. Dipl.-Ing. Hans-Dieter Stucke

Ein ganz besonderer Dank gilt den Assistenten der Informatik-Labore im 7.Stock der HAW-Hamburg. Namentlich möchten wir Herrn Bruno Carstensen erwähnen, für seine allzeit entgegengebrachte Hilfsbereitschaft und freundliche Unterstützung.

Vielen Dank auch an alle Mitglieder des ECO-Teams der HAW-Hamburg für die gute Zusammenarbeit, die vielen anregenden Diskussionen und den Spaß den wir als Teamgemeinschaft hatten. Natürlich auch ein Dank an die Teammitglieder des H.A.W.K.S. Racing-Teams für ihre Unterstützung.

Inhaltsverzeichnis

1 Einführung

Diese Arbeit entstand im Rahmen des Projektes ECO-Marathon der HAW Hamburg. Der ECO-Marathon ist der größte studentische Wettbewerb in Europa auf dem Gebiet der Energie-Effizienz für motorisierte Fahrzeuge, der vom Mineralölkonzern Shell gefördert wird. Die HAW Hamburg beteiligt sich zum ersten Mal an diesem Wettbewerb mit einem dafür zu konzipierenden Fahrzeug, das mit 1 Liter Benzin möglichst 1.000 Km oder mehr zurücklegen soll. Im Mai 2008 soll das Fahrzeug "Pingu II" auf einer Rennstrecke in Nogaro, Frankreich sich mit anderen Fahrzeugen europäischer Project-Teams messen. Das Ziel dieser Abschlussarbeit ist es, im Rahmen der Telemetrietechnik eine Datenoptimierung sowohl in der Erfassung als auch in der Auswertung der Daten zu erreichen, diese dem ECO-Team zur Verfügung zu stellen, um ein optimales Fahr-und Verbrauchsergebnis zu gewährleisten. Betreut wurde diese Arbeit von Herrn Prof. Dr. rer. nat. Baran und dem Leiter des ECO-Teams Prof. Dipl.-Ing. Hans-Dieter Stucke. Die Bearbeitung erfolgte grundsätzlich gemeinsam, wobei sich jedes Teammitglied auf eigene Aspekte des Projekts konzentriert hat. Die Aufgaben wurden bei meist regelmäßigen Meetings des Teams koordiniert und das weitere Vorgehen der Teammitglieder aufeinander abgestimmt.

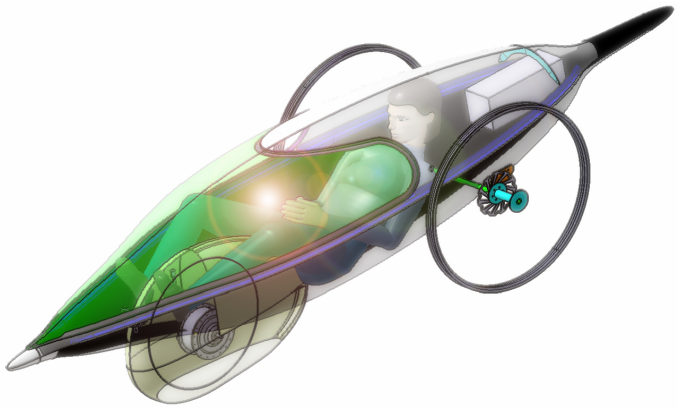


Abbildung 1.1: Pingu II, Bildquelle [6]

1.1 Motivation

Das Team: Das Eco-Team (EfficientCarOperation-Team) ist ein interdisziplinäres Team bestehend aus 18 Studentinnen und Studenten, sowie vier Professoren aus den Departments Fahrzeug-/Flugzeugbau, Technik und Informatik der HAW Hamburg. Das Projekt des "Pingu II" begann Mitte 2007 unter der Leitung von Herrn Professor Hans-Dieter Stucke, als sich eine Gruppe interessierter Studenten formierte, die sich der Herausforderung des Shell Eco-Marathons stellt. Ziel des Teams ist es, innerhalb von zehn Monaten ein fahrtüchtiges Gefährt zu konstruieren, zu fertigen und auf dem 24. Shell Eco-Marathon im Mai 2008 auf der Rennstrecke in Nogaro, Frankreich in das Ziel zu fahren. Das ECO-Team wird in der Klasse Prototypen mit Brennstoffzellen-Antrieb antreten.

Pingu II: Der "Pingu II" wurde an die Gestalt eines stromlinienförmigen Kaiserpinguins angelehnt, da diese Form im Windkanal optimale Werte lieferte. Der Strömungswiderstand eines Pinguins ist zehnmal geringer als der eines modernen Sportwagens. Als Grundmodell dient dabei ein Liegefahrrad. Das wasserstoffbetriebene Brennstoffzellen-Fahrzeug ist drei Meter lang und einen Meter breit. Die Energie der 1,2KW Niedrigtemperatur-Brennstoffzelle treibt den Radnabenmotor am Vorderrad an. Damit kann der "Pingu II" eine Geschwindigkeit von bis zu 38 km/h erreichen. Die Ober- und Unterschale sind aus GFK gefertigt. Das Fahrzeug hat eine mechanische Vorderradlenkung und wiegt ca. 70 kg.

Eco-Marathon: Der Shell Eco-Marathon findet einmal jährlich statt. Teams aus ganz Europa messen hier ihre Konzepte in verschiedenen Kategorien. Der Grundgedanke ist, mit immer weniger Treibstoff immer weiter zu fahren. Hierzu treffen sich jedes Jahr Studenten und Studentinnen in 200 Teams aus ganz Europa und treten mit ihren Fahrzeugen in verschiedenen Kategorien gegeneinander an. Bei einem Durchschnittstempo von mindestens 30 km/h müssen die Fahrzeuge eine Gesamtstrecke von ca. 27 km zurücklegen. Im Ziel werden die Verbrauchs-Ergebnisse der verschiedenen Klassen (Hydrogen/Solar usw.) hochgerechnet und der Sieger ermittelt. Der Rekord liegt bei einer Strecke von 3.836 km mit nur einem Liter Benzin.

1.2 Aufgabenstellung

Es ergab sich die Aufgabe, dem Team soviele Daten wie möglich zu dem Betriebsverhalten der einzelnen Komponenten ihres Fahrzeugs zu liefern, um eine Auswertung

und damit auch die Möglichkeit einer gezielten Verbesserung zu schaffen. Diese Aufgabe erwies sich von Anfang an als schwierig, denn die Zeit war knapp. Innerhalb von 10 Monaten musste das ECO-Team ein komplettes Fahrzeug konstruieren und passende Antriebskomponenten beschaffen. Um alle Bewerbungskriterien zu erfüllen, musste dabei die Konstruktion der Karrosserie bis Dezember 2007 abgeschlossen sein. Brennstoffzelle und Motor fehlten aber immer noch. Erschwert wurde die Konstruktion und der Bau des Prototypen durch ein 77 Punkte umfassendes Regelwerk von Shell.

2 Konzept

Daraus entstand das Konzept ein Messdatenerfassungssystem zu konstruieren, das den Regeln genügt und so klein ist, dass es sich leicht im Fahrzeug unterbringen lässt, wenig Energie verbraucht und einfach zu steuern ist. Dabei musste es möglichst unabhängig von den erst später ausgewählten Antriebskomponenten Daten liefern können, wozu eigene Sensoren einzuplanen waren. Das System sollte ausserdem leicht zu erweitern und anzupassen sein, um schnell auf Änderungen am Fahrzeug reagieren zu können. Das Telemetriesystem sollte sowohl während der Fahrt Daten an das Boxenteam liefern als auch gleichzeitig intern alle Messwerte für eine spätere Analyse loggen, was sowohl der Optimierung als auch der Fehlersuche dienen sollte. Um die Messdaten des Fahrzeugs schnell und einfach visualisieren zu können, kam von Anfang an nur ein System zur grafischen Datenauswertung in Frage. Dafür waren Daten wie Position und Geschwindigkeit sowie Temperaturen, Spannungen, Strom und eventuell dazukommende Digitalsignale zu erfassen. (siehe Abbildung 2.1)

2.1 Aufzuzeichnende Daten

Für das Konzept war es erforderlich, folgende Messdaten aufzunehmen:

- Geschwindigkeit des Fahrzeugs
- Spannung von Brennstoffzelle und Bordbatterie
- Strom des Motors
- Temperaturen des Motors, der Brennstoffzelle, der Wasserstoffflasche und des Innenraums
- Position des Fahrzeugs

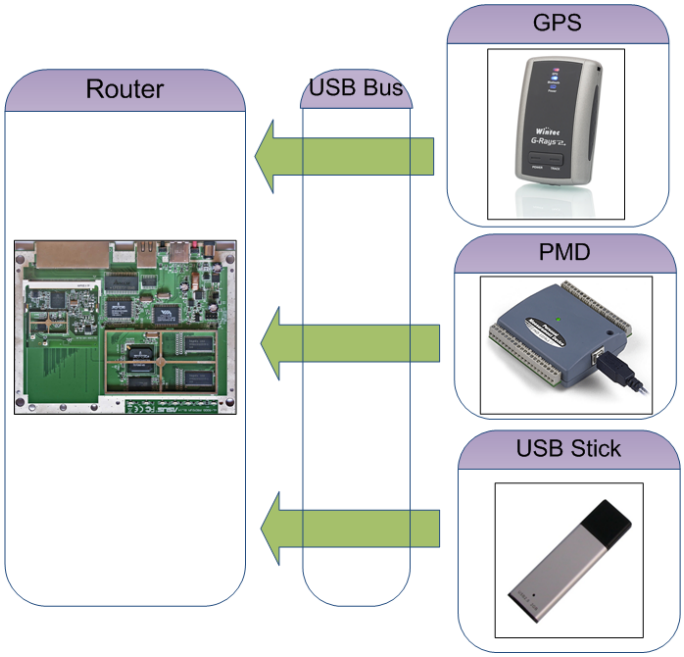


Abbildung 2.1: USB Geräte

2.1.1 Position des Fahrzeugs

Um die Position des Fahrzeugs zu bestimmen, wird ein GPS-Empfänger eingesetzt, da es die am besten zu realisierende Variante ist. Zudem sind GPS Empfänger mittlerweile ausreichend genau, um die Position eines sich bewegenden Fahrzeugs auf bis zu 2m genau festzustellen. Dieser liefert zusätzlich zu der Position weitere Daten wie Höhe, Geschwindigkeit und die Zeit. Der Vorteil, dass der GPS-Empfänger die aktuelle Zeit aus den empfangenen Daten errechnet, gewinnt durch die fehlende Echtzeituhr des Systems später noch an Bedeutung.

2.1.2 Daten der Brennstoffzelle

Die Brennstoffzelle liefert über eine eigene serielle Schnittstelle unter anderem Sensordaten, wie Temperaturen, Spannungen, Strom und den Wasserstoffdruck. Mittels einer eigenen Software können diese Daten visualisiert werden. Dadurch ist es möglich über diese serielle Schnittstelle Daten zu empfangen und auszuwerten. Hierfür ist das Übertragungsprotokoll der seriellen Schnittstelle vom Hersteller im Datenblatt beschrieben und wird später noch genauer erläutert.

2.1.3 Daten der Fahrzeugsensoren

Um Temperaturen, Geschwindigkeit, Spannungen und Strom messen zu können, ist eine Sensorplatine zum Erfassen von analogen und digitalen Sensordaten erforderlich. Diese Platine ermöglicht eine Anpassung dieser Daten an das System. Dieses wird über einen sogenannten PMD realisiert. Durch diese Sensordatenaufnahme ist es möglich, das System durch zusätzliche PMDs zu erweitern. Da es bei dem ECO-Marathon auf den Energieverbrauch des Fahrzeugs ankommt, war es erforderlich, möglichst sparsame Messmethoden zu wählen, wie z.B. berührungslos messende Sensoren. Da während der Planung noch nicht abzusehen war, dass die Brennstoffzelle selbst benutzbare Daten liefern würde, sind Messdaten wie Spannung und Strom sowohl von der Brennstoffzelle als auch von dem PMD verfügbar. Die Geschwindigkeit wird von dem GPS und dem Geschwindigkeitsmesser am PMD gemessen, wobei das GPS im unbewegten Zustand keine genauen Daten über die Geschwindigkeit liefert. Deswegen ist ein zusätzlicher Geschwindigkeitssensor im Fahrzeug verbaut. (siehe Abbildung 2.2)

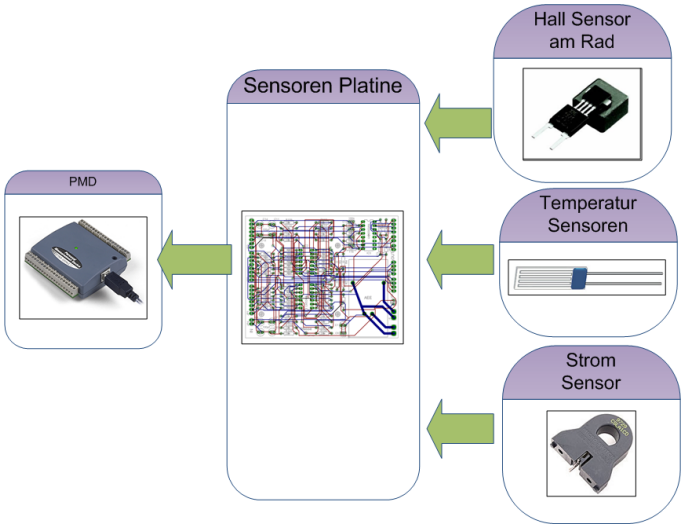


Abbildung 2.2: Sensoren

2.2 Darstellung der Daten

Um energieeffizient zu sein, ist eine genaue Messung der Durchschnittsgeschwindigkeit erforderlich, da das Team eine Mindestdurchschnittsgeschwindigkeit von 30km/h erzielen muss. Höhere Geschwindigkeiten würden den Energieverbrauch erhöhen. Dieses lässt sich durch die Aufzeichnung der Sensordaten im nachhin-

ein analysieren. Weitere Analysierungsmöglichkeiten bieten die Aufzeichnungen der weiteren Sensordaten. Zum Beispiel der Zusammenhang von Temperatur und Leistung, bei Motor und Brennstoffzelle. Weitere Möglichkeiten bietet die Auswertung der direkten Messdatenübertragung.

3 Theoretische Grundlagen

3.1 Telemetrie

Telemetrie bedeutet Fernmessung. Man unterscheidet in Nahfeldtelemetrie (Datenübertragung auf kurzen Distanzen, wie z.B. Temperaturen und Verbrauchsdaten bei fahrenden Autos) und Fernfeldtelemetrie (Datenübertragung auf größere Distanzen, wie z.B. Wetterdaten oder Daten von Raumfahrzeugen). Telemetrie heißt: Messdaten werden von einem Sensor an einem Standort ermittelt und an eine Empfangsstelle weitergegeben, die die Daten sammelt und ggfs. auch gleich auswertet. Um auf die gelieferten Daten evtl. reagieren zu können, wird ein Rück-Pfad zum erfassenden Sensor eingerichtet. Man spricht dann von Telekommandierung oder Tele-Command (z.B. ferngesteuerte Roboter oder unbemannte Flugobjekte).Die Telemetrie hat ihren Ursprung in der Übermittlung von Wetterdaten aus Wetterballonen und wurde erstmals 1920 eingesetzt.

3.1.1 Sensorik

Das Wort Sensorik leitet sich aus dem Lateinischen "sensere" = fühlen, wahrnehmen ab. Es bezeichnet entsprechend die Bestandteile eines Systems, die die Wahrnehmung betreffen. Unter Sensorik ist hier primär die "technische Sensorik" gemeint im Gegesatz zur menschlichen Sinneswahrnehmung. Innerhalb der Ingenieurwissenschaften versteht man unter Sensorik die Menge der Sensoren einer Steuerung, einer Messanlage oder eines Regelkreises. Die "technische Sensorik" bedient sich technischer Produkte, wie Sensoren, die nicht elektrische Messgrößen in elektrische Signale umwandelt. So könnte die Sensorik eines Roboters beispielsweise bestehen aus Kamera, Abstands-Sensor Initiator, Mikrofon, Infrarot-Detektor, Geschwindigkeits-Messer (Tachometer) und GPS-Empfänger. Sensoren finden heute fast auf allen Ebenen des täglichen Lebens ihren Einsatz, um Messungen und/oder Kontrollen durchzuführen. Beispiele: Lichtschranken, Bewegungsmelder, Temperaturmessung.

3.1.2 GPS

GPS ist ein Globales satellitengestützte Positionsbestimmungssystem. Mit GPS ist im allgemeinen das NAVSTAR-GPS (Navigational Satellite Timing and Ranging -

Global Positioning System) des US-Verteidigungsministeriums gemeint, das Ende der 80er Jahre zur weltweiten Positionsbestimmung und Zeitmessung entwickelt wurde. GPS wurde am 17. Juli 1995 offiziell in Betrieb genommen. (Quelle [13])

3.1.3 WLAN

Wireless Local Area Network - WLAN bezeichnet ein "drahtloses", lokales Funknetz, wobei meistens ein Standard der IEEE 802.11-Familie gemeint ist. Für WLAN im 2.4GHz Band ist in Europa nach ETSI Standard eine Sendeleistung von 20dBm Peak erlaubt, gute Module mit einer Eingangsempfindlichkeit von -96dBm erreichen dadurch bei Sichtverbindung Reichweiten von vielen hundert Metern. Als Modulation wird für 802.11b DSSS (Direct Sequence Spread Spectrum) verwendet, welche gegenüber Störungen sehr robust arbeitet. Für 802.11g wird stattdessen OFDM als Modulation benutzt, welche höhere Übertragungsraten auf Kosten der Störunempfindlichkeit ermöglicht. (Quelle [12])

Es gibt für WLAN zwei Betriebsmodi, die sich in der Arbeitsweise unterscheiden. Der meistverwendete Infrastruktur Modus, in dem es für jeden Client nur den Accesspoint (AP) oder auch mehrere als Kommunikationspartner gibt, die auf unterschiedlichen Kanälen arbeiten können. Jeder Accesspoints sendet alle 100ms ein Beacon mit seiner Macadresse und dem Netzwerknamen (SSID) das von den Clients empfangen und ausgewertet wird, so dass sie die Accesspoints und Netzwerke unterscheiden können. Jede Datenübertragung mit anderen Clients läuft über die Punkt-zu-Punkt Verbindung mit seinem AP ab, der sich dann um die Übertragung zum eigentlichen Ziel kümmert. Dadurch, dass nur zwei Partner direkt an einer Übertragung beteiligt sind, wird eine einfache und trotzdem sehr starke (dynamisch ausgehandelte) Verschlüsselung (WPA/WPA2) zwischen dem AP und dem Client ermöglicht.

Als Alternative gibt es die ADHOC Betriebsart, in der es nur Clients gibt. Jeder ist gleichberechtigt und kommuniziert dort direkt mit jedem Teilnehmer des Netzes der in seiner direkten Reichweite liegt. Die Verschlüsselung ist schwach (WEP) und gilt seit einigen Jahren als leicht zu überwinden. Die Kryptoschlüssel werden statisch eingetragen, da es in einem Adhocnetzwerk keinen zentralen Koordinator gibt. Der Funkkanal wird fest eingestellt, ob eine Übertragung erfolgreich verläuft, hängt davon ab, ob sich die Partner in gegenseitiger Reichweite befinden.

3.1.4 Übertragungsprotokolle

UDP: Das User Datagram Protocol (UDP) ist ein verbindungsloses Netzprotokoll ohne Fehlerkorrektur. Um Übertragungsfehler muss sich die Anwendung selber kümmern, es gibt nur eine Integritätsüberprüfung, die mit der mitgesendeten

Prüfsumme des Pakets arbeitet. Die Entwicklung von UDP begann 1977 als RFC 768, weil man für die Übertragung von Sprache ein einfacheres Protokoll als das bisherige verbindungsorientierte TCP benötigte. UDP ist nur für die Adressierung zuständig und verzichtet darauf die Datenübertragung zu sichern, da dies unter anderem zu unkalkulierbaren Verzögerungen bei zeitkritischen Übertragungen führen könnte. Aufgrund seiner Verbindungslosigkeit können UDP Pakete auch als Broadcast an alle Teilnehmer eines Netzwerks versendet werden. (Quelle [11])

TCP: Das Transmission Control Protocol (TCP) ist ein zuverlässiges, verbindungs- und paketorientiertes Transportprotokoll in Computernetzwerken. Es ist Teil der Internetprotokollfamilie, der Grundlage des Internets. Es wurde von Robert E. Kahn und Vinton G. Cerf. ab 1973 in mehreren Jahren entwickelt. Die erste Standardisierung von TCP erfolgte 1981 als RFC 793. Es gab danach noch viele Erweiterungen, die in neuen RFCs spezifiziert worden sind. Im Unterschied zum verbindungslosen UDP stellt TCP eine virtuelle Punkt-zu-Punkt Verbindung her. Auf diesem Kanal können in beide Richtungen Daten übertragen werden. Aufgrund seiner Eigenschaften, wie der automatischen Erkennung von Datenverlusten und deren automatischer Behebung, sowie der Verhinderung einer Netzwerküberlastung ist TCP ein sehr weit verbreitetes Protokoll zur zuverlässigen Datenübertragung. (Quelle [10])

3.1.5 USB

USB ist ein serieller Bus mit einem Host-Controller, der über Hubs bis zu 127 Geräte bedienen kann. Der Bus arbeitet mit differentieller Datenübertragung über zwei symetrische Kabelpaare. Das eine überträgt das Datensignal unverändert, das andere das invertierte Signal. Der Empfänger bildet die Differenzspannung beider Signale, eingestrahlte Störungen werden weitgehend eliminiert und die Übertragungssicherheit gesteigert. Durch die gute Bandbreite und die wenigen Adern sowie die integrierte Spannungsversorgung angeschlossener Geräte eignet sich der USB besonders zum einfachen Anschluss von Kleingeräten. (Quelle [9])

3.1.6 Linux

Linux ist ein Multiuser Betriebssystem, das von Linus Torvalds entwickelt und 1992 unter die GPLV2 gestellt wurde. Es basiert auf GNU, ist Unix ähnlich und aufgrund seiner breiten Architektur-, Hardware-, und Softwareunterstützung für eine Vielzahl

von Geräten, vom kleinen Mikrocontroller bis zum Supercomputer einsetzbar. Praktisch werden meist ganze Linux-Distributionen wie z.B. OpenWrt genutzt, in denen der Kernel und Softwarepakete zu einem Betriebssystem zusammengestellt wurden.

3.1.7 Brennstoffzelle

Eine Brennstoffzelle liefert durch kontinuierliche Zuführung eines Brennstoffes und Oxidationsmittels, elektrische Energie durch chemische Reaktionsenergie. Die Brennstoffzelle kehrt den Prozess, der aus dem Schulunterricht bekannten Elektrolyse, um. Die Brennstoffzelle nimmt die beiden Stoffe, Wasserstoffgas und Sauerstoff und wandelt sie wieder in Wasser. Dabei wird theoretisch die Menge elektrischer Energie wieder abgegeben, die bei der Elektrolyse zur Spaltung notwendig war. Der Brennstoff muss zur Stromproduktion der Brennstoffzelle kontinuierlich zugeführt werden, um Energie zu erzeugen. Sie speichert keine Energie, sondert wandelt sie. Energie aus chemischen Energieträgern wird heute meist durch Verbrennung in einer Wärmekraftmaschine, in Verbindung mit einem Generator, über den Umweg der thermischen und der Bewegungsenergie erzeugt. Die Brennstoffzelle tut dies effizienter, da sie ohne Umweg die Energie erzeugt und somit potenziell effizienter ist. Zudem ist der Aufbau einer Brennstoffzelle einfacher als eine herkömmliche Energiequelle und somit abnutzungsfester und zuverlässiger. Die meisten Brennstoffzellen funktionieren mit Luft, so dass der Sauerstoff nicht gespeichert werden muß. Daher werden sie meist auch mit Wasserstoff und Wasser in Verbindung gebracht. In diesem Fall wird eine Brennstoffzelle der Firma Ballard verwendet. (siehe Abbildung 3.1) Jedoch gibt es verschiedene Arten von Brennstoffzellen. (Quelle [8])

- Die alkalische Brennstoffzelle - AFC - ist, abgesehen von Grooves Prototypen, der älteste Brennstoffzellentyp. Diese findet man heute noch in der Raumfahrt und bei U-Boot-Antrieben. Sie ist die einzige Zellenart, die reinsten Sauerstoff und Wasserstoff zur Energieumwandlung benötigt, da schon geringste Verunreinigungen die Zelle zerstören.
- Die Polymer-Elektrolyt-Membran-Brennstoffzelle PEM - sie ist sehr unkompliziert, hat eine hohe Leistungsdichte bei geringem Gewicht und benötigt statt reinem Sauerstoff nur Luftsauerstoff als Reaktionsgas. Der Wasserstoff ist mit Hilfe eines Reformers zu erzeugen.
- Die Phosphorsäure Brennstoffzelle - PAFC - sie ist die am weitesten entwickelte Brennstoffzelle. Auf Grund ihrer hohen Betriebstemperatur ist sie ideal für den Einsatz in Blockheizkraftwerken. Als Reduktgase benötigt die PAFC Luftsauerstoff und Wasserstoff.
- Die Karbonatschmelzen Brennstoffzelle - MCFC arbeitet in hohen Temperaturbereichen von 580..660°C. Sie läßt sich direkt und ohne Reformer mit

Erdgas, Kohlegas, Biogas und Synthesegas betreiben.

- Die Oxidkeramische Brennstoffzelle - SOFC - sie arbeitet mit Luftsauerstoff und Wasserstoff und hat eine Betriebstemperatur im Bereich von 800...1000°C. Der Aufwand für die Wasserstofferzeugung sinkt damit erheblich, da die hohen Temperaturen eine zellinterne Teilreformierung von Erdgas zu Wasserstoff erlaubt.
- Die Direct methanol fuel cell - DMFC - ist die einzige Zelle, die nicht Wasserstoff sondern Methanol als Energiequelle einsetzt. Sie selbst wandelt das Methanol in Wasserstoff um, ohne einen Reformer dafür zu nutzen. Dadurch kann sie sehr leicht und kompakt gebaut werden.

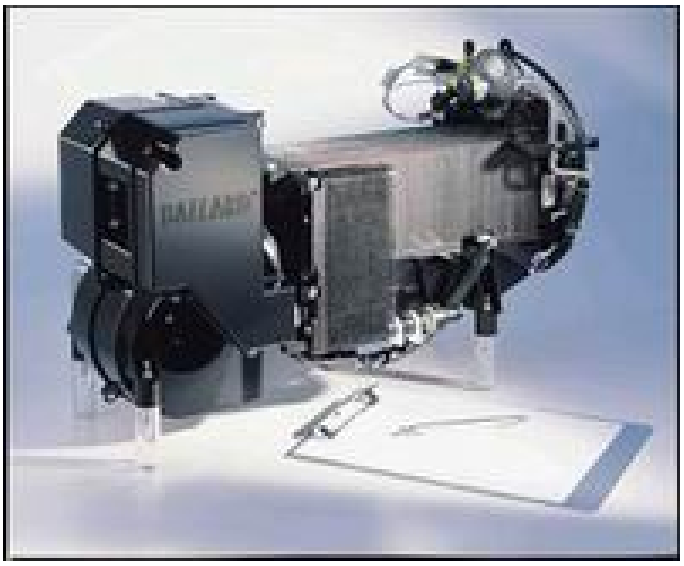


Abbildung 3.1: Verwendete Brennstoffzelle von Ballard Nexa; Quelle [28]

4 Design der Telemetriebox

4.1 Vorstellung der genutzten Software

4.1.1 OpenWrt

OpenWrt ist eine für Router entwickelte Linux-Distribution, die als Alternative zu einer Originalfirmware auf vielen Geräten eingesetzt werden kann. Die Firmware bringt ein voll beschreibbares jffs2 Dateisystem, das Konfigurationstool uci sowie den Paketmanager ipkg mit, über den sich Softwarepakete und Kernelmodule installieren lassen. Dadurch wird es auf einfache Weise ermöglicht, Hardware für ursprünglich nicht vom Hersteller vorgesehene Zwecke zu verwenden. Die Distribution bietet die Hard- und Softwareunterstützung des Linuxkernels, den Komfort eines vollwertigen Betriebssystems wie zum Beispiel USB-, Filesystem- und umfangreiche Netzwerkunterstützung. (Quelle [15])

4.1.2 LabVIEW

LabView "Laboratory Virtual Instrumentation Engineering Workbench" der Firma National Instruments, ist ein graphisches Programmiersystem. Seit 1986 ist die Software erstmalig auf dem Markt. Die Software ist mit der Programmiersprache VEE der Firma Hewlett-Packard vergleichbar, die seit 2007 von der Firma Agilent in der Version 8.0 zu Verfügung steht. Als VI's (Virtuelle Instruments) werden die Programme von LabView bezeichnet und bestehen aus einem Frontpanel und einem Blockdiagramm. Das Frontpanel beinhaltet die Benutzerschnittstelle (GUI - grafische Benutzeroberfläche) und das Blockdiagramm, den graphischen Programmcode, der compiliert wird und somit eine ähnliche Performance wie vergleichbare Hochsprachen besitzt. LabView wird daher vor allem in der Mess- und Automatisierungstechnik eingesetzt. Die Programmiersprache des Systems heißt "G", nach dem Datenfluss-Modell. Sie ist dadurch besonders gut für die Erfassung und Verarbeitung von Daten geeignet. Die Software steht aktuell in der Version 8.5.1 (Stand März 2008) für Windows, Linux und Solaris zur Verfügung. (Quelle [16])

4.1.3 EAGLE Layout Editor 4.16r2

Eagle ist ein Programm der Firma CadSoft um Leiterplatten zu erstellen. Es handelt sich um eine EDA (Electronic Design Automation) Software und gehört damit zu den CAD (Computer Aided Design) Programmen.Die Software besteht aus den Komponenten: Layout-Editor, SchaltpLAN-Editor, Autorouter und einer erweiterbaren Bauteil-Datenbank. Eine kostenlose nicht kommerzielle Version der Software mit der Beschränkung auf eine halbe Eurokarte (100 x 80 mm) ist genauso wie eine kommerzielle Version erhältlich. Eagle ist für die Plattformen Windows, Linus und OS X erhältlich. Aktuelle Version 5.0 (Stand Mai 2008). (Quelle [14])

4.1.4 GPS Deamon

Der GPS Deamon gpsd ist ein Netzwerkservice der Daten von GPS Empfängern in verschiedenen Formaten, wie zum Beispiel NMEA verarbeitet, aufbereitet und per TCP an Clientsoftware sendet, desweiteren gehören zu dem Deamon einige Diagnose- und Monitoringtools. Die aufbereiteten GPS Daten sind weniger aufwendig zu verarbeiten. Durch seine Spezialisierung kann der gpsd ausserdem mit Ungenauigkeiten, wie sie in der NMEA Spezifikation vorkommen, besser umgehen und ist als eigenständiger Systemdienst auch leichter mit updates zu versorgen als direkt in eine Software integrierter Code. (Quelle [17])

4.1.5 u-center der Firma u-blox

u-center ist eine GPS Evaluierungssoftware des Chipherstellers u-blox, die besonders auf deren Antaris Chipsatzreihe abgestimmt wurde und die Konfiguration des Empfängers sowie die Analyse der vom GPS bereitgestellten Daten ermöglicht. Vom Programm werden sowohl die NMEA Daten als auch das u-blox spezifische UBX protocol ausgewertet. Ausgegeben werden die zu den Satelliten vorhandenen Daten, dazu gehören die Signalstärken, Positionen und DGPS Korrekturdaten, Geschwindigkeit, Richtung, sowie 2D Pfade der berechneten Position und der Höhe. Aktuelle Version 5.00 (Stand Mai 2008). (Quelle [18])

4.1.6 Measurement Computing PMD

Die Software der Firma Measurement Computing Corporation beinhaltet verschiedene Programme:

- InstaCal, ein Dienstprogramm für die Installation, die Kalibrierung und das Austesten
- TracerDAQ, ein Anwendungsprogramm für Registrierstreifen/Datenprotokollierung

- Universal Library, eine Datenerfassungs- und Steuerelemente-Programmierbibliothek
- SoftWIRE® für VS .NET (voll funktionsfähig, Lizenz für einen begrenzten Zeitraum)
- SoftWIRE für VB6 (voll funktionsfähig, Lizenz für einen begrenzten Zeitraum)
- SoftWIRE MCC DAQ Komponenten für .NET
- SoftWIRE MCC DAQ Steuerelemente für VB6
- Universal Library für LabVIEW

Mittels des Programmes InstaCal wird der PMD kalibriert und getestet, ob die Datenerfassung und die Steuerhardware funktionieren. Durch Fehlerprüfmöglichkeiten wird der PMD eingerichtet und es kann eine Hardware-Konfigurations-Datei erzeugt werden, die in Programmier- oder Anwendungssoftware verwendet werden kann. Das Programm TracerDAQ beinhaltet virtuelle Instrumente, die zum Plotten und Protokollieren von Daten verwendet werden kann. Sie verhalten sich wie Oszilloskope. (Quelle [19])

4.1.7 NexaMon OEM

Die Software liefert mehr als 20 relevante Daten zum charakteristischen Betriebsverhalten eines Brennstoffzellensystems. Kenntnisse, die für die Integration der Brennstoffzelle in reale Anwendungen wichtig sind. Unter Anwendung dieser Kenntnisse kann das Paket anschließend in verschiedenste Applikationen eingesetzt werden. Das integrierte EEPROM liefert zusätzliche Informationen zu Laufzeit, Gesamtbetriebsdauer und evtl. Fehlbedienungen. NexaMon OEM ist zudem erforderlich bei Fehlern, die ein Eingreifen in die Hardware erfordern, um sicherheitsrelevante Fehler, wie ausgetretener Wasserstoff an der Zelle, zurückzusetzen.

4.1.8 Smart Install Maker v5.02

Smart Install Maker ist eine Software, mit der man eigene Programme in eine zu installierende Datei verpacken kann, die bei dem Ausführen eine Installationsroutine ablaufen lässt. Dadurch lassen sich viele Dateien einfach in unterschiedliche Verzeichnisse kopieren und z.B. Verknüpfungen erstellen. Der Anwender braucht sich dadurch keine Gedanken über die Installationsverzeichnisse machen, wodurch Fehler ausgeschlossen sind. Aktuelle Version 5.02 (Stand Mai 2008). (Quelle [20])

4.1.9 Ubuntu

Ubuntu ist eine seit Oktober 2004 existierende und auf Debian basierende Linux-Distribution. Mit dem Ziel, ein einfach zu installierendes und bedienbares Betriebs-

system mit darauf abgestimmter Software zu schaffen. Es ist heute eine der meist genutzten Linux-Distributionen. Ubuntu setzt GNOME als Desktopumgebung ein. Es existieren aber auch Versionen mit KDE Desktop sowie auf Server abgestimmte Versionen der Distribution. Für die Entwicklung der Box wurden die Versionen 7.10 sowie später 8.04 genutzt. (Quelle [21])

4.1.10 Packetyzer

Paketyzer ermöglicht die Aufzeichnung des Datenverkehrs einer Netzwerk-Schnittstelle und stellt die Daten in Form einzelner Pakete dar. Dabei können die Daten übersichtlich dargestellt und analysiert werden, der Inhalt von mitgeschnittenen Paketen betrachtet oder nach Inhalten gefiltert werden. Aktuelle Version 5.0.0 (Stand Mai 2008).

Für Windows wird dazu der Treiber WinPcap benötigt. Als Alternative für Linux gibt es Wireshark, das mit Packetyzer verwandt ist. (Quelle [22])

4.2 Vorstellung der genutzten Hardware

4.2.1 Telemetriebox

Die Telemetriebox wurde von Grund auf neu konstruiert. Sie entstand mit dem Ziel eine Box zu entwickeln, die mit Konsumerelektronik ausgestattet, einfach zu beschaffenden Bauteilen, preisgünstig und leicht zu modifizieren ist. Die Box besteht aus folgenden Komponenten:

- einer Zentralen Sensoren Platine, an die die Sensoren angeschlossen werden und die Eingangswerte für die Signalaufnahme umgewandelt werden.
- Sie übernimmt die Spannungsversorgung für alle Geräte in dem System, in diesem Fall der ASUS WL-500G Premium.
- eine Signalaufnahmebox (PMD), die die Werte über den USB Bus direkt an das System übermittelt. Durch den USB Bus sind weitere PMD oder andere USB Geräte problemlos anschliessbar, dadurch ist die Sensorenvielfalt groß.
- eine Kommunikationsplatine, mit der man mit der Box kommunizieren kann ohne einen Laptop anzuschließen.
- USB Hub, der die Verteilung der verschiedenen internen und externen Geräte ermöglicht. (siehe Abbildung 4.1)

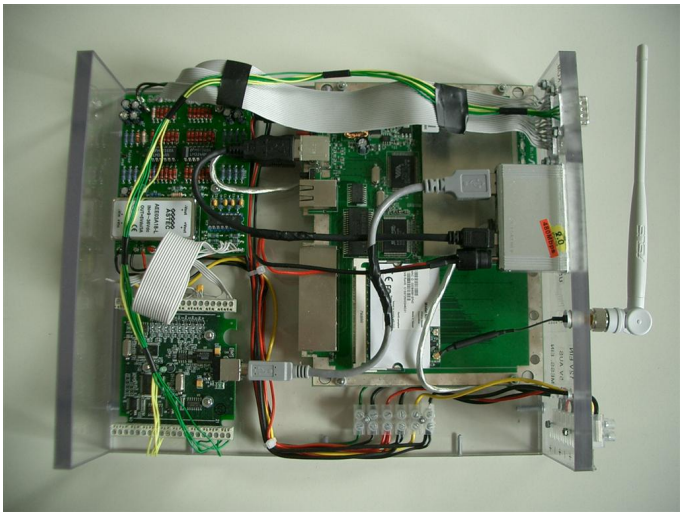


Abbildung 4.1: Die im Pingu II verbaute Telemetrie Box

WLAN Karte

Das Wistron CM9 802.11a/b/g MiniPCI WLANmodul basiert auf dem Atheros AR5004X Chipsatz und wird unter Linux von dem Madwifitreiber in das System eingebunden. Es erreicht Sendeleistungen bis zu 19dBm und -98dBm Eingangsempfindlichkeit bei einem maximalen Leistungsbedarf von 1,4W; es werden sowohl Infrastruktur- als auch Adhoc- und WDS-Modi unterstützt, teilweise sogar gleichzeitig. WLAN hat gegenüber anderen Funkübertragungen wie Bluetooth, den Vorteil, dass es direkt Ethernetpakete überträgt und bei einer maximalen Sendeleistung von 100mW eine relativ hohe Reichweite erzielt. (Quelle [23])

USB Hub

Der USB Hub ist ein Model mit Aluminiumgehäuse, 4 Ports und einem MiniUSB-Anschluss sowie einem 5V Hohlstecker Eingang zur eigenen Spannungsversorgung. Dadurch wird die Stromversorgung der Geräte vom Hub erledigt und das System von dieser Aufgabe befreit.

4.2.2 Zentrale Sensoren Platine

Die zentrale Sensoren Platine wurde für die Anforderungen des Projektes Eco-Team HAW entwickelt und mit weiteren frei belegbaren Sensoreneingängen geplant und ausgestattet. Sie besitzt acht analoge Eingänge für Spannungsverstärkungen, wie

z.B. für Temperatur- oder Stromsensoren. Zudem stehen vier Hall-Sensor Impulszähleingänge zur Verfügung. Dieser Teil wurde aus der Diplomarbeit von Bachir Blal Quelle [7] verwendet. Die Radsensorenausgänge (JP1) sind an die Eingänge des Treiberbausteins(IC1) angeschlossen. Der Treiber erfordert eine eigene Spannungsversorgung von 5V, die vom Kontroller(JP3) bezogen wird. Die Ausgänge des Treiberbausteins werden direkt mit dem PMD verbunden (siehe Abbildung /refBlachir)

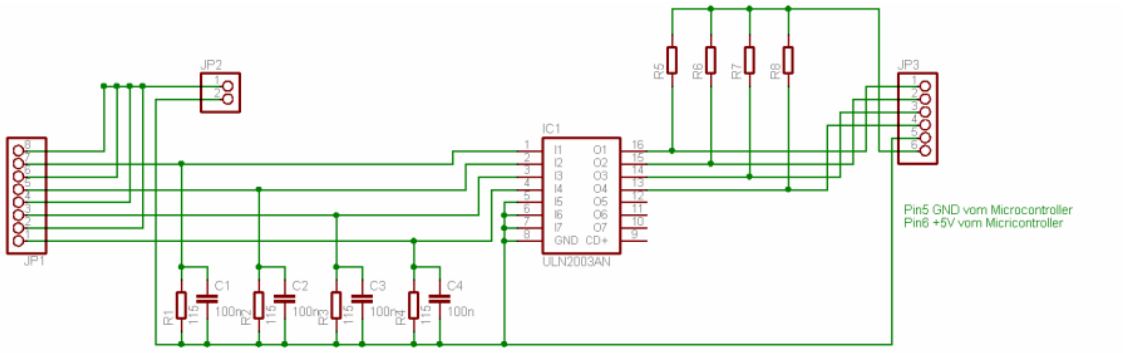


Abbildung 4.2: Schaltung für die Radsensoren; Quelle [7]

Desweiteren sind auf der Platine zwei Spannungsmesseingänge realisiert, die speziell auf die Anforderungen angepasst sind. Ein Eingang ist für die Überwachung der eigenen Spannungsversorgung, die aus einer 12V Spannungsquelle besteht, der Zweite für die Überwachung der Brennstoffzelle. Jedoch ist dieser durch Anpassung des Widerstandsverhältnisses auf der Platine modifizierbar. Die Platine wurde mit dem Programm EAGLE Layout Editor 4.16r2 entworfen. (siehe Abbildung 4.3 & 4.4)

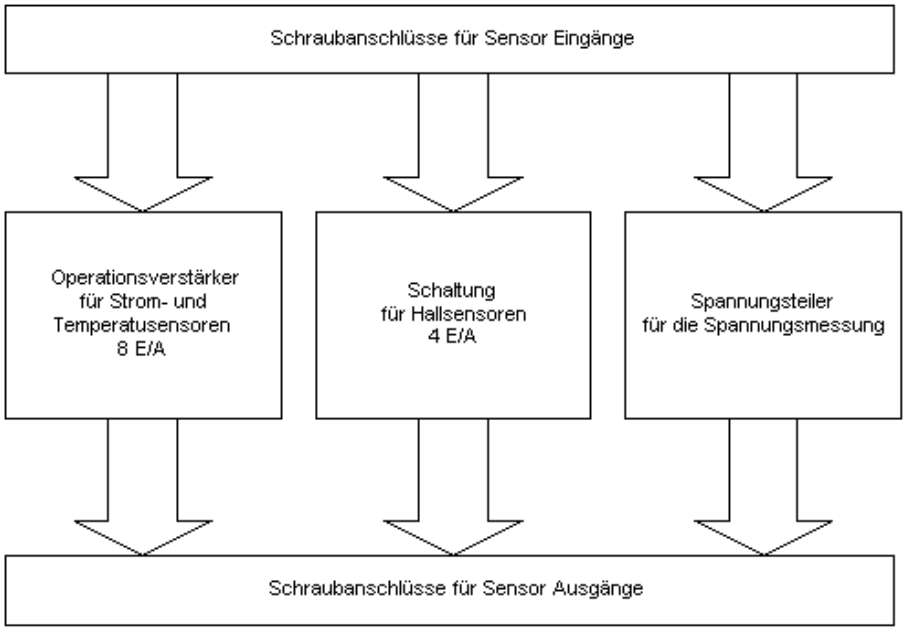


Abbildung 4.3: Blockschaltbild der Zentrale Sensoren Platine

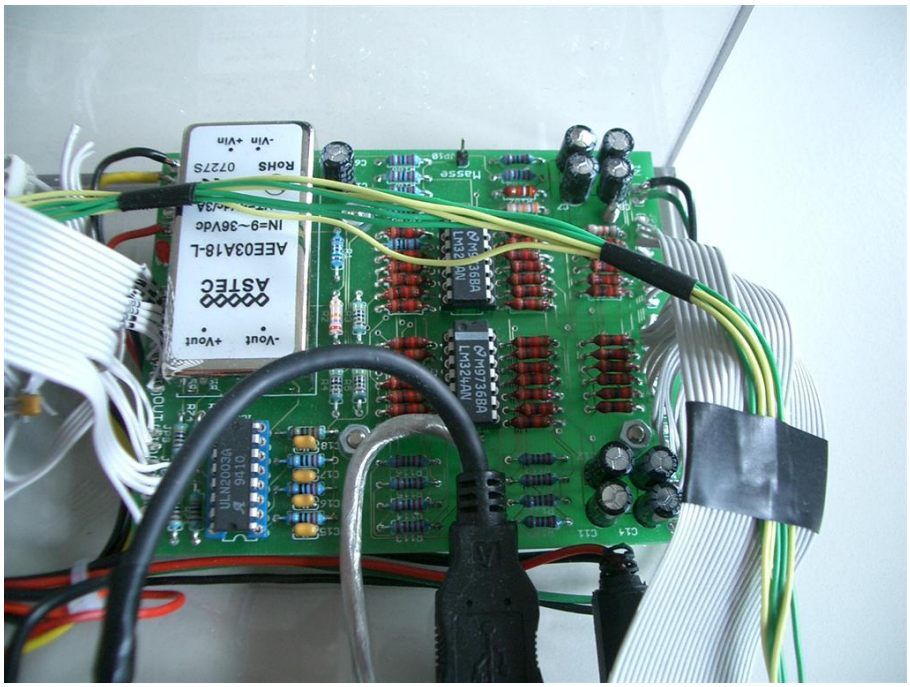


Abbildung 4.4: Die in der Telemetrie Box verbaute Zentrale Sensoren Platine

Operationsverstärker LM324AN

Hierbei handelt es sich um einen Baustein mit vier einzelnen Operationsverstärkern auf Basis von TTL. (siehe Abbildung 4.5) Er besitzt eine Bandbreite von 1Mhz, Spannungseingangs Offset von max 7mV und eine Anstiegszeit von 0,5µs. Der Baustein besitzt ein 14 Pin DIP-Gehäuse. (Quelle [24])

Abbildung 4.5: Treiberbaustein ULN2003A; Quelle [29]

Treiberbaustein ULN2003A

Bei diesem Treiberbaustein handelt es sich um ein Transistorarray, in dem sieben Open Collector Darlington-Transistoren enthalten sind. In einer Treiberstufe vom ULN2003A sind zwei Transistoren in einer Darlingtonschaltung kombiniert. Dadurch multiplizieren sich deren Stromverstärkungen (bis 500mA). (Quelle [25])

DC/DC-Wandler AEE03A18-L

Der kurzschlussfeste Wandler erreichet durch seine Abschirmung an allen sechs Seiten die Vorgaben der EN55022. (siehe Abbildung 4.6) Dadurch ist er für den Einsatz in empfindlichen Geräten geeignet. Eine MTBF (demonstrated) von über 1 Million Stunden und eine Spannungsgenauigkeit von +/-2% ermöglichen weite Anwendungsbereiche in den verschiedensten Marktsegementen. Der Wandler besitzt einen weiten Eingangsspannungsbereich und kann in allen Lötprozessen eingesetzt werden. (Quelle [30])

Abbildung 4.6: DC/DC Spannungswandler AEE03A18-L; Quelle [26]

4.2.3 System für Datenverarbeitung

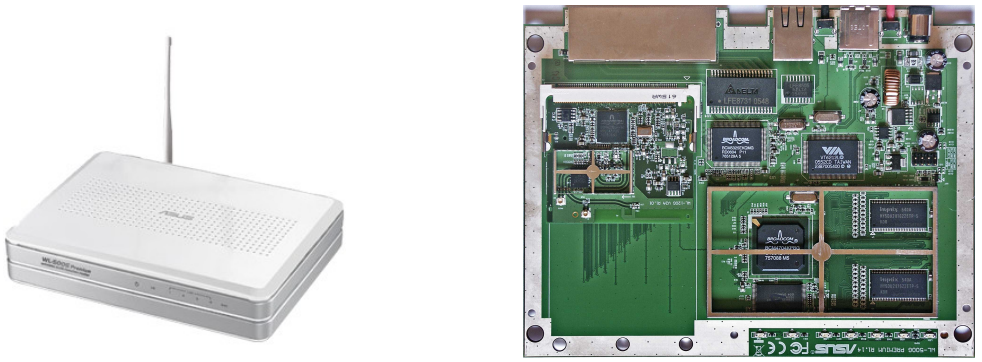
Es wurde eine Hardware benötigt, die den folgenden Anforderungen entspricht. Sie sollte stromsparend, klein, leicht, erweiterbar und günstig sein. Ausserdem sollte Linux und OpenWrt im Speziellen darauf lauffähig sein, um den Komfort eines Betriebssystems und die mit Linux einhergehende breite Hard- und Softwareunterstützung zu erhalten. Ein Wechsel der Hardwareplattform beschränkt sich so auf ein einfaches Neukompilieren der Software. Die folgenden Schnittstellen mussten entweder direkt oder per USB und MiniPCI nachrüstbar sein, USB2, WLAN, Ethernet und Seriell. USB, um die Messdaten zu speichern, die GPS Daten zu empfangen und Messwerte abzuspeichern. WLAN, um die aktuellen Messdaten an eine entfernte Station zu Übertragen. Seriell, um Messwerte und Diagnoseinformationen von der RS232 Schnittstelle der Brennstoffzelle zu empfangen. Ethernet, um das System einfach um weitere Hardware erweitern zu können und für die Einrichtung, Diagnose und Softwaretests einen einfachen und schnellen Zugriff auf das System zu erhalten.

Zur Auswahl standen folgende Systeme: (Quelle [27])

- NSLU2:
  - Einsatzzweck: Network Attached Storage Device für USB Festplatten 266MHz Intel XScale IXP420, 32MB SD-Ram, 8MB Flash, 1x Ethernet, 2x USB2 Anschlüsse, 5V Versorgung bis 10W
  - Vorteile: Gute Speicheraustattung, Prozessor und Schnittstellen recht schnell, stromsparend, günstig, gute Verfügbarkeit am Markt
  - Nachteile: Nur ein Ethernetanschluss und weder MiniPCI noch eingebautes WLAN
- ASUS WL-500g Premium:

  - Einsatzzweck: Router für DSL Internetzugänge 266MHz Broadcom BCM94704, 32MB Ram, 8MB Flash, 5x Ethernet, 2x USB2, 802.11b/g BCM4318 MiniPCI WLAN, 5V Versorgung bis 12W
  - Vorteile: Gute Speicherausstattung, umfangreiche Schnittstellenaustattung, stromsparend, günstig, gute Verfügbarkeit am Markt
  - Nachteile: Austausch des Broadcom WLAN Moduls zu Atheros wegen Treiberproblemen mit Linux Kernel 2.6 nötig
- MICROSPACE MPC20:
  - Einsatzzweck: MiniPC für Industrie- und Netzwerkanwendungen 500MHz AMD GEODE LX800, bis 1GB DDR-Ram, 2x Ethernet, 4x USB2, Sound, VGA, 1x MiniPCI, 1x CF Cardslot, 1x 44Pin IDE, 2x COM, 1x LPT, 12V Versorgung bis 10W
  - Vorteile: Sehr schnell, grosse Speicher, alle Schnittstellen eingebaut, MiniPCI WLAN möglich, Stromsparend
  - Nachteile: Für den vorgesehenen Zweck hohe Kosten

Die Entscheidung viel letztlich, wegen der direkten OpenWrt Unterstützung, der integrierten Schnittstellen, guter Erweiterbarkeit und Verfügbarkeit sowie wegen des moderaten Preises auf den ASUS WL-500g Premium Router. (siehe Abbildung 4.7) Die serielle Schnittstelle wurde genau wie das GPS, der USB Stick und die Messdatenerfassung per USB angeschlossen, zusätzlich benötigt wurde lediglich ein kleiner vier Port USB2 Hub.



(a) Router **(b)** verbaute Router Platine

Abbildung 4.7: Der ASUS WL-500g Premium Router; Quelle [31] & [32]

4.2.4 Kommunikationsplatine

Die Kommunikationsplatine ist die Schnittstelle zwischen Mensch und Box. (siehe Abbildung 4.9 & 4.8) Sie besteht aus vier LEDs, die die unterschiedlichen Systemzustände anzeigen und drei Schaltereingänge, von denen nur einer gebraucht wird. Durch Betätigen des Schalters wird der USB-Stick gemountet und die Aufzeichnug durch die Telemetriebox gestartet. Um den Stick für die Auswertung zu entfernen, muss man nur den Schalter umschalten. Damit wird die Aufzeichnung beendet und der USB-Stick wird unmounted.

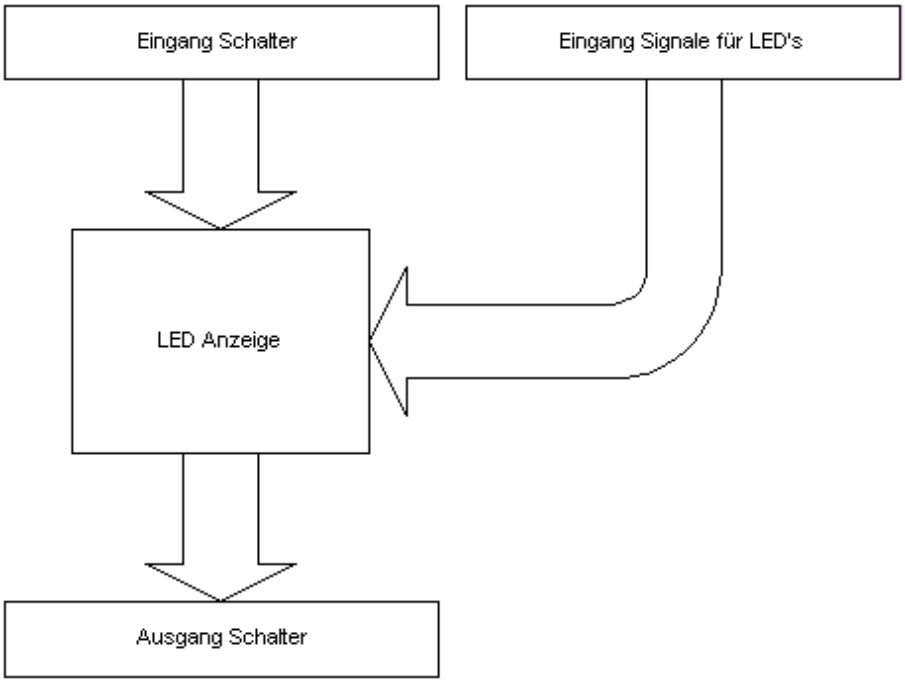


Abbildung 4.8: Blockschaltbild der Kommunikations Platine

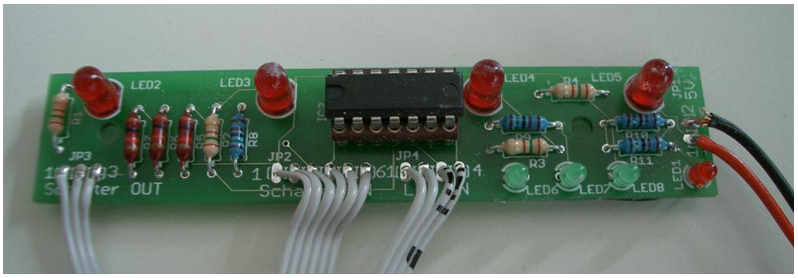


Abbildung 4.9: Kommunikations Platine

Treiberbaustein SN74LS06N

Hierbei handelt es sich um ein Baustein mit sechs einzelnen Invertern. Diese Hexagoninverterpuffer/ -treiber mit Hochspannungsausgängen eignen sich für High-Level-Schaltungen (wie MOS) oder als Inverter-Buffer zum Betreiben von TTL Eingängen.

4.2.5 Brennstoffzelle

Das Nexa-System bietet bei einer ungeregelten DC-Spannung von 26V eine Nennleistung von bis zu 1200 Watt. (siehe Abbildung 4.10) Mit der Verwendung des externen Brennstoffes Wasserstoff, kann die Zelle einen kontinuierlichen Berieb gewährleisten und ist nur durch die Menge des lagernden Brennstoffs beschränkt. Das Nexa Modul ist extrem ruhig, produziert Null-Schadstoffemissionen und ist dadurch auch für Indoor-Operationen einsetzbar. Das Nexa Power-Modul ist ein voll integriertes System, das seine Spannung aus Wasserstoff und Luft produziert. Sie enthält ein BALLARD Brennstoffzellen-Stack sowie alle notwendigen Zusatzeinrichtungen für den Brennstoffzellen-Betrieb. Onboard-Sensoren überwachen die System-Performance. Der automatisierte Betrieb wird durch eine Steuerplatine und den Einsatz eines Mikroprozessor gewährleistet. (Quelle [8])

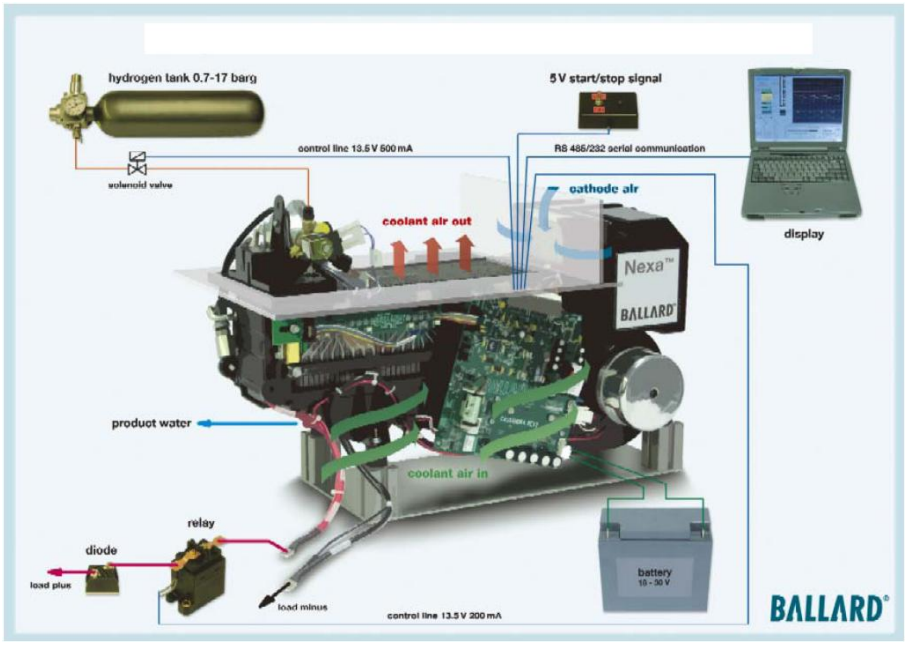
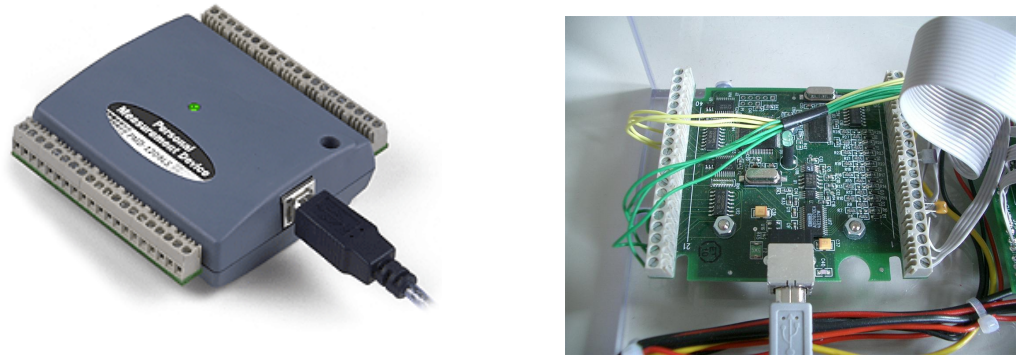


Abbildung 4.10: verbaute Brennstoffzelle; Quelle [28]

4.2.6 Measurement Computing PMD

Der PMD-1208FS ist ein USB-basiertes Datenerfassungsmodul mit 10-bit Genauigkeit bei +10V und 8 analogen Eingängen und bis zu 50KHz Samplingfrequenz, dazu kommen ein 32Bit Counter und 16 Digitale Ein-/Ausgänge die in zwei Ports gruppiert sind. (siehe Abbildung 4.11) Das Gerät wird durch die +5-V-Spannung des USB versorgt, wodurch keine externe Stromversorgung erforderlich ist. Die E/A-Verbindungen erfolgen über Schraubanschlüsse, die sich an beiden Seiten des PMD-1208FS befinden. Gegenüber einem Atmelboard, das z.B. bei dem H.A.W.K.S. Team der HAW-Hamburg im Einsatz ist, verfügt der PMD über eine USB-Schnittstelle und ist dadurch besser in das System zu integrieren.



(a) PMD **(b)** verbaute PMD Platine

Abbildung 4.11: PMD; Quelle [33]

4.2.7 GPS

An das GPS wurden folgende Anforderungen gestellt: Positionsbestimmung mit 2Hz bis 4Hz, integrierter USB Anschluss, hohe Empfindlichkeit, aktive Antenne, DGPS fähig, umfangreiche Konfigurations- und Diagnosemöglichkeiten, Möglichkeit zu Firmwareupdates, Bluetooth. Ausgehend von diesen Anforderungen, insbesondere mehr als 1Hz bei der Positionsbestimmung, blieben nur noch die Geräte mit dem u-blox Antaris 4 Chipsatz übrig. Der Wintec WBT201 Bluetooth GPSEmpfänger erfüllte dabei alle genannten Anforderungen. Tests mit dem Gerät bestätigten dabei die Eignung für den geplanten Einsatzzweck, guter Empfang und wenig Drift im stationären Betrieb. (siehe Abbildung 4.12) Der MiniUSB-Anschluss und die Bluetooth Schnittstelle ermöglichen selbst im eingebauten Zustand eine gleichzeitige Verbindung, sowohl mit dem Telemetriesystem, als auch für Diagnose- und Konfigurationszwecke per Bluetooth mit dem Notebook. Der im USB-Empfänger verbaute USB-Seriell Chip cp2101 wird von OpenWrt durch das kmod-usb-serial-cp2101

Kernelmodul Paket unterstützt. - Atmel / ublox ATR 0625 GPS Chipsatz - 16 parallele Satellitenverfolgungskanäle - Hohe Empfindlichkeit: 158 dBm - 4 HertzTechnik - Dual Funktion: per Kabel und Bluetooth nutzbar - Durch die intern verwendete Dekodierung ist keine externe Hardware für den Empfang des WAAS/EGNOSSignals erforderlich - Unterstützung von NMEA 0183 v2.3, UBX, RTCM Protokollen für den Datentransfer - Mittels Bluetooth kann die Datenübertragung innerhalb einer Reichweite von 10 Metern erfolgen - Spritzwasserdicht - Betriebstemperatur: -10°C bis +70°C

Abbildung 4.12: GPS; Quelle [35]

4.2.8 USB-RS232

Der Digitus DA-70146 USB 2.0 zu seriell Konverter DSUB 9M war bereits vorhanden, intern wird ein vom Linuxkernel unterstützter FTDI Chip eingesetzt. Dieser wird von OpenWrt durch das kmod-usb-serial-ftdi Paket unterstützt.

4.2.9 Sensoren

Um Sensordaten vom "Pingu II" zu bekommen, sind Sensoren unverzichtbar. Hier werden die einzelnen Sensoren angesprochen, die im "Pingu II" verbaut wurden.

Hallsensor

Der Hall-Sensor findet in der Automobilindustrie vielfältige Anwendung, z. B. im Gurtschloss, im Türschließsystem, bei der Pedalzustandserkennung, in der Getriebeschaltung, zur Erkennung des Zündzeitpunkts sowie der Geschwindigleitsmessung. Hauptvorteil ist die Unempfindlichkeit gegen (unmagnetischen) Schmutz und Wasser. (siehe Abbildung 4.13) Der Hall-Effekt ist nach dem amerikanischen Physiker Edwin Herbert Hall (1855 bis 1938) benannt und zeigt den magnetischen Einfluss auf stromführende Leitungen. Das Phänomen des Hall-Effekts basiert auf der Lorentzkraft. Es tritt dann auf, wenn sich Ladungsträger mit einer Geschwindigkeit bewegen und senkrecht zur Bewegungsrichtung ein Magnetfeld anliegt. In diesem Fall wirkt auf die Ladungsträger eine Kraft, deren Richtung senkrecht sowohl zum Magnetfeld als auch zur Bewegungsrichtung der Ladungsträger ist. Durch die Lorentzkraft findet im Hall-Element eine Ladungsverschiebung statt. Daraus ergibt sich ein elektrisches Feld, welches auf die Ladungsträger eine Kraft ausübt, die der Lorentzkraft entgegenwirkt. Zwischen diesen beiden Kräften stellt sich ein Gleichgewicht ein. Bringt man an den Längsseiten des Hall-Elementes Kontakte an, dann liegt an ihnen die Hallspannung. Die Höhe der Hallspannung ist proportional zur Lorentzkraft und ein Maß für die Dichte des magnetischen Flusses. Der Hall-Effekt wird u.a. in Sensoren eingesetzt, so in Strommess-Sensoren, bei denen das Hall-Element so nahe wie möglich an das Magnetfeld gelegt wird. Außerdem werden Hall-Elemente zur Untersuchung von Halbleitermaterialien benutzt.

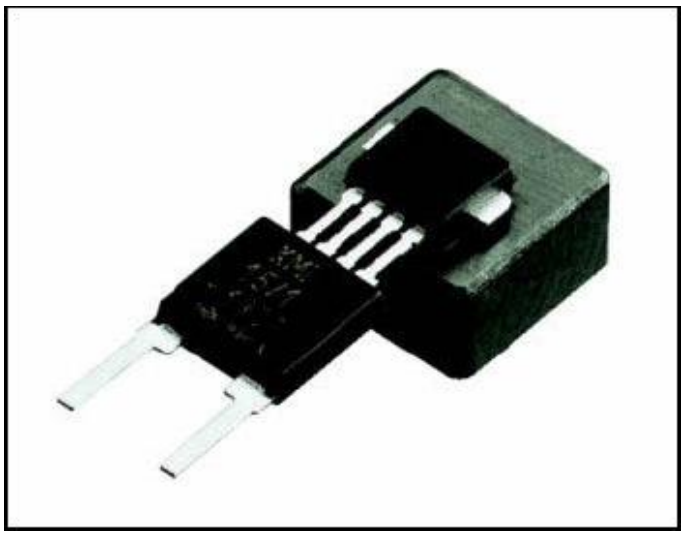


Abbildung 4.13: Geschwindigkeitssensor; Quelle [36]

Stromsensor

Stromsensoren sind elektrische Bauelemente, mit denen die Stromstärke in Kabeln und Stromschienen in der Regel galvanisch getrennt (berührungslos), anhand von deren Magnetfeldern, gemessen werden kann, dabei wird zwischen reinen Wechselstromsensoren und Gleichstromsensoren unterschieden, die zudem auch Wechselströme messen können. Direkt abbildende, auf einem Hallsensor basierende Stromsensoren, arbeiten mit einem geschlitzten Ringkern aus einem ferromagnetischen Material. Dabei wird entweder der stromführende Leiter umschlossen oder ein Leiter mit wenigen Windungen auf eine Primärwicklung aufgebracht. Der Sensor selbst ist im Luftspalt des Ringkerns untergebracht, zugleich begrenzt der Luftspalt die magnetische Flussdichte, linearisiert den Zusammenhang zwischen Magnetfeld und Strom und ermöglicht so einen großen Messbereich. Das am Hallsensor gemessene Signal ist proportional zum Magnetfeld und somit zum Strom. Nach diesem Prinzip arbeiten auch Stromzangen, diese bestehen aus einem aufklappbaren Ringkern. Die Wahl viel auf einen Honeywell CSLA1CD Stromsensor, da dieser eine gute Empfindlichkeit aufweist und Ströme zwischen 0A und 57A messen kann. (siehe Abbildung 4.14)

Abbildung 4.14: Stromsensor; Quelle [37]

Temperatursensor

Die Änderung des elektrischen Widerstandes gehört zu den häufigsten Methoden der elektrischen Temperaturmessung. Dabei hat sich Platin als Widerstandsmaterial wegen seiner chemischen Beständigkeit, der einfachen Bearbeitung und der guten Reproduzierbarkeit der elektrischen Eigenschaften, durchgesetzt. Es handelt sich dabei unter anderem um Kaltleiter (PtC), mit einer genormten linearen Werteta-

belle, die man auch als Widerstandsthermometer bezeichnen kann. Wegen der gegenüber einem NTC linearen Kennlinie ist die Entscheidung deshalb zugunsten des PT-1000 (PtC) ausgefallen. Ein Vorteil der Standardisierung des Nennwiderstands und der Widerstandsänderung ist dabei die leichte Austauschbarkeit der Temperaturfühler, ohne dass anschließend eine Neukalibrierung der Messkette notwendig wird. Als Widerstandsthermometer ist ein PtC genauer als z. B. Thermoelemente. Pt1000-Widerstände sind Temperaturfühler für Messungen im Bereich -200 °C bis 850 °C, die auf der elektrischen Widerstandsänderung eines Platindrahtes oder einer Platinschicht unter Temperatureinfluss basieren. (siehe Abbildung 4.15) Die Platin-Temperatursensoren werden durch ihren Nennwiderstand R0 bei einer Temperatur von 0 °C charakterisiert, gebräuchliche Typen sind z.B. Pt1000 (R0= 1 kOhm)

Abbildung 4.15: Temperatursensor; Quelle [38]

5 Realisierung

5.1 Das Gesamtsystem im Überblick

Der Hauptteil der Arbeit steckt in der Telemetriebox, die in dem Fahrzeug verbaut wurde. (siehe Abbildung 5.1) Sie bildet das Herzstück im gesamten Konzept der Telemetrieerfassung und -auswertung. Die Box ist so entwickelt worden, dass an ihr verschiedenste Geräte wie analoge Sensoren, GPS Empfänger oder digitale Ein- und Ausgabegeräte anschließbar sind. Realisiert sind in dieser Arbeit, ein GPS Empfänger, der über die USB Schnittstelle der Box angeschlossen wird, die Brennstoffzelle, die ihre Daten über eine serielle Schnittstelle ausgibt und der PMD, an ihm sind sechs analoge und ein digitaler Sensor über die Sensoren Platine angeschlossen. Drei Temperatur Sensoren, ein Strom Sensor, zwei Spannungen und ein Hall Sensor, der über einen Counter die Geschwindigkeit ermittelt. Die Temperatur Sensoren sind folgendermaßen verteilt, einer auf der Wasserstoffflasche, einer am Motor und einer im Innenraum. Die Spannungen werden an der Brennstoffzelle und der Versorgungsbatterie gemessen. Zudem ist die Kommunikationsplatine an den digitalen Ein- und Ausgängen des PMD angeschlossen. Die Platine dient der Interaktion mit der Box, ein Schalter startet und stoppt die Aufzeichnung der Sensordaten in einem Logfile auf dem USB-Stick. Mittels Leuchtdioden werden die drei Zustände "Box betriebsbereit", "Aufzeichnung läuft" und "3D-Fix" des GPS signalisiert. Die Telemetriebox besitzt zudem ein WLAN Modul, über das sie ständig die Sensordaten per Broadcast versendet. Diese werden mittels der Telemetrieauswertesoftware direkt verarbeitet und simuliert. Dies geschieht so lange, bis das Fahrzeug nicht mehr im Empfangsbereich ist. Die Daten, die während der gesamten Fahrt auf dem USB-Stick aufgezeichnet werden, können später über eine zweite Software ausgelesen und zur Analyse visualisiert werde.

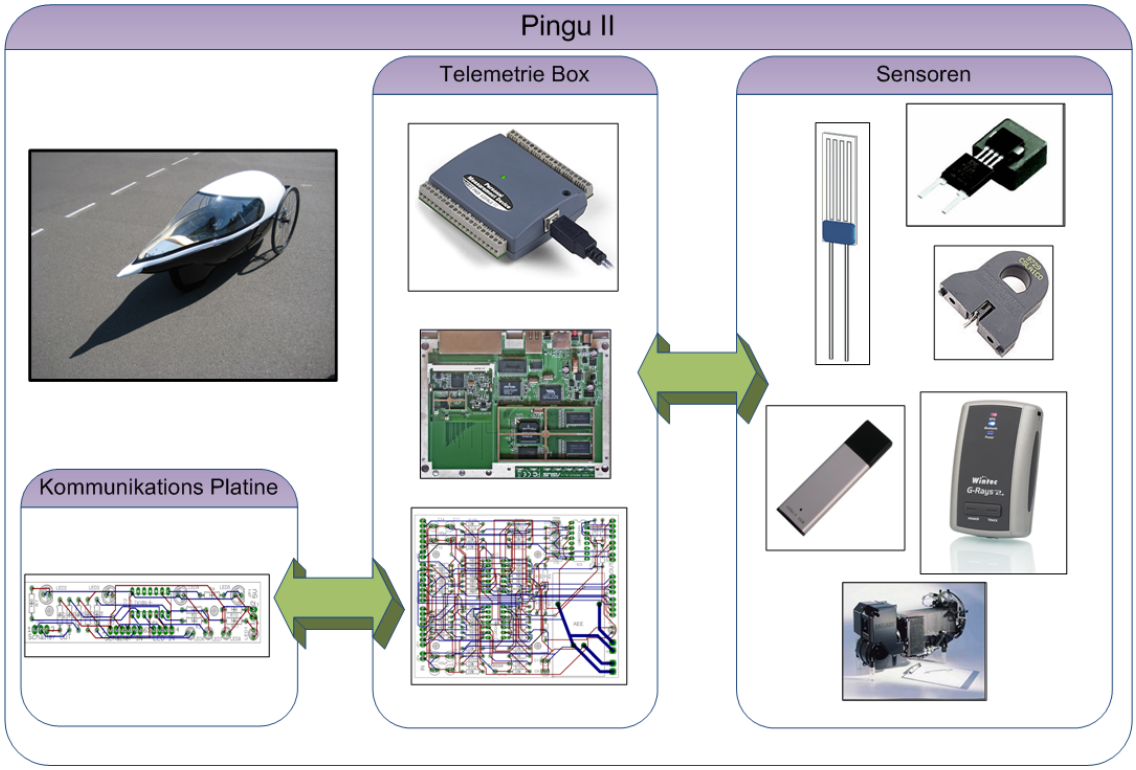


Abbildung 5.1: Blockschaltbild Gesamtsystem

5.2 Bau der Hardware

Im Abschnitt Bau der Hardware wird beschrieben, wie die einzelnen Komponenten geplant und entstanden sind.

5.2.1 Die Box

Beim Bau der Hardware ist von Anfang an auf geringes Gewicht, geringen Energiebedarf, Erweiterbarkeit und leicht auszutauschende Komponenten hin entwickelt worden. Da die Hardware in ein kleines Fahrzeug, wie dem "Pingu II" verbaut werden musste, waren dies einige Anforderungen, die an das System gestellt wurden. Geringer Energieverbrauch war hierbei besonders wichtig, da das Eco-Team mit sowenig Energie wie möglich auskommen muss. Das Reglement des Eco-Marathons schreibt vor, dass alle Technik sichtbar im Fahrzeug verbaut werden muss. Daher ist die Box aus transparentem Makrolon gefertig, wodurch sie sehr stabil sowie leicht ist und dem Reglement entspricht. (siehe Abbildung 5.2)

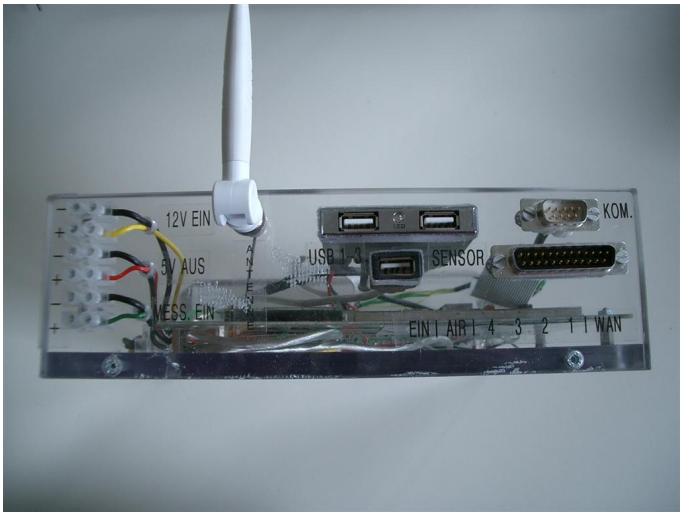


Abbildung 5.2: Die Telemetrie Box von vorne

5.2.2 Zentrale Sensoren Platine

Da die analogen Sensoren alle passiv sind, benötigen sie eine Verstärkung. Dies geschieht in diesem Fall über Operationsverstärker, die als Differenzverstärker beschaltet sind. Ein Differenzverstärker ist ein elektronischer Verstärker mit zwei Eingängen, bei dem nicht ein einzelnes Signal, sondern die Differenz der beiden Eingangssignale Ue1 und Ue2 verstärkt wird. Dieses war notwendig, da der Stromsensor einen Offset von U/2 besitzt. Das heißt, bei einer Versorgungsspannung von 12V, liegt der Offset des Sensors bei 6V, pro 1A verändert er die Spannung um ca. 50mv. Da der PMD ein Messbereich von 0-10V besitzt, wäre eine reine Verstärkung nicht ausreichend gewesen, um ein gutes Messergebnis zu erhalten. Daher wurde die Änderung um ein Vierfaches verstärkt. Für den Differenzverstärker ist es daher wichtig eine konstante Referenzspannung zu haben, um nicht durch Spannungsschwankungen die Messergebnisse zu beeinträchtigen. Daher wurde auf der Platine ein DC/DC Wandler eingebaut. Dieser liefert eine konstante 5 V Spannungsversorgung und somit die Referenzspannung für den Differenzverstärker. Dieser Wandler versorgt zudem alle weiteren Geräte in der Telemetrie Box mit 5V.

Beim Entwurf der Zentralen Sensoren Platine, stand die Verstärkung der einzelnen Sensoren im Vordergrund. Ziel war es den Messbereich des PMD von 0 bis 10V so weit wie möglich auszuschöpfen, um eine größtmögliche Genauigkeit der Messung zu bekommen. Der PMD besitzt für den 0 bis 10V eine Messgenauigkeit von 10Bit, das entspricht 1024 Messwerten - 102 Messwerte pro V. Durch ungenaue Widerstände und äußere Einflüsse wie Einstreuungen des PWM (Pulsweitenmodu-

lation) Signals des Motors, ist die Genauigkeit der Sensoren auf 8Bit gesunken, was 256 Messwerten entspricht. Das Layouten der Platine wurde mit dem Programm EAGLE Layout Editor erstellt. (siehe Abbildung 5.3 und 5.4)

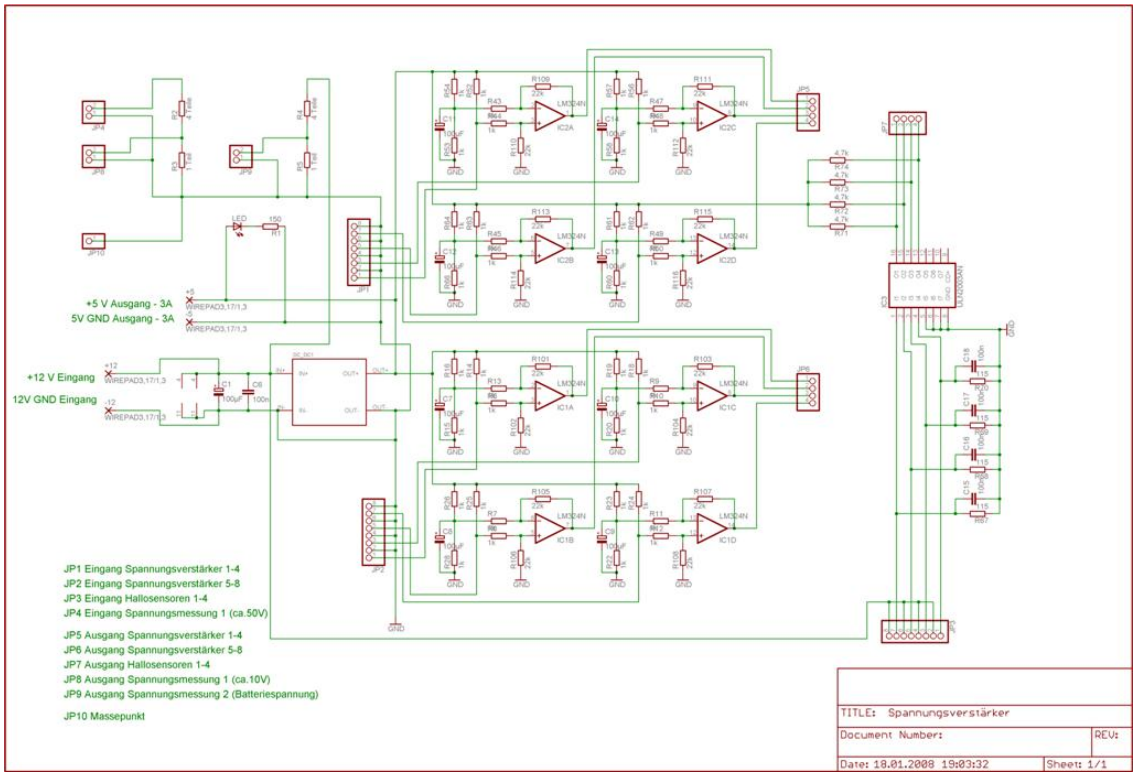


Abbildung 5.3: Zentrale Sensoren Platine - Schaltplan

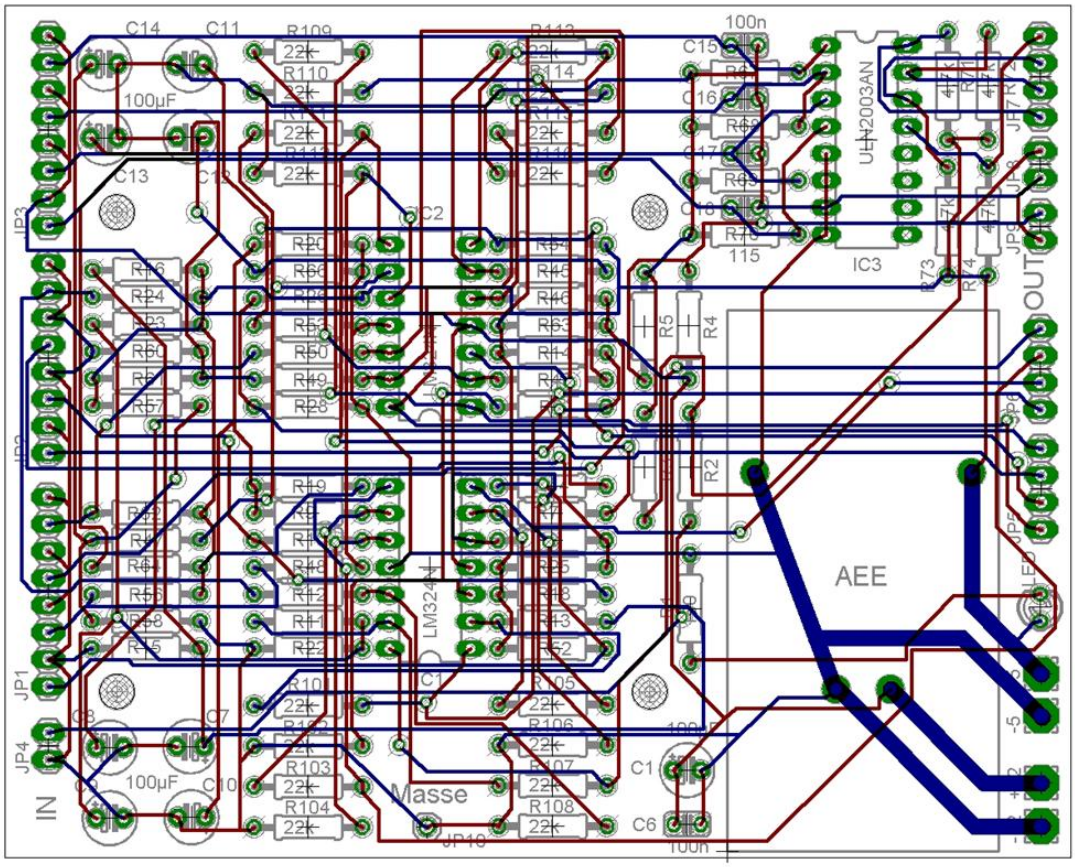


Abbildung 5.4: Zentrale Sensoren Platine - Platinenlayout

5.2.3 Kommunikationsplatine

Beim Entwurf der Kommunikationsplatine stand eine einfach zu bedienende Steuerung im Vordergrund. Daher wird hier auch nur ein Schalter zum Bedienen der Box benötigt. Zudem sind 4 LEDs für die Zustände der Software realisiert. Diese werden über den Treiberbaustein SN74LS06N angesteuert, um das vom PMD kommende Signal zu verstärken. Das Layouten der Platine wurde ebenso mit dem Programm EAGLE Layout Editor erstellt. (siehe Abbildung 5.5 und 5.6)

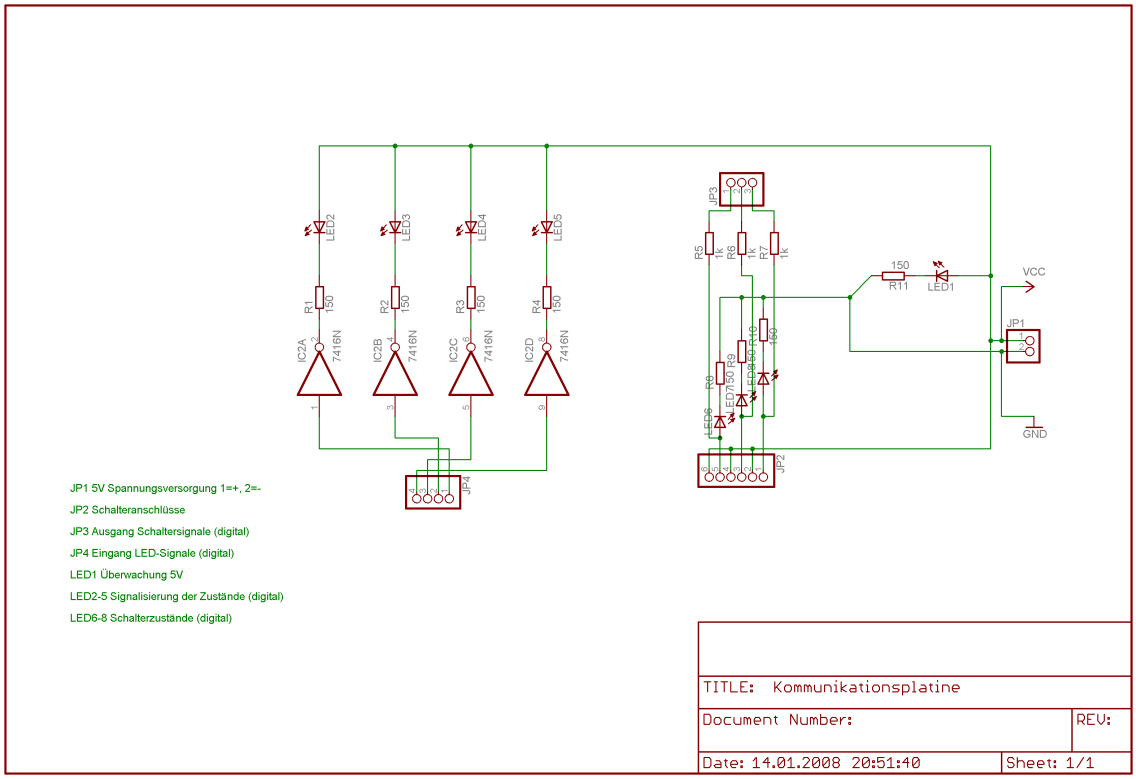


Abbildung 5.5: Kommunikationsplatine - Schaltplan

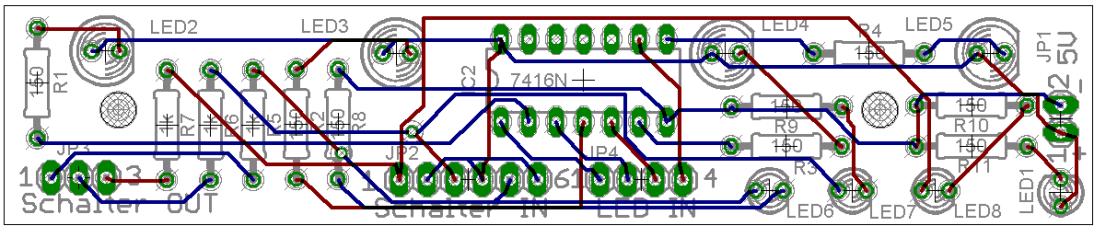


Abbildung 5.6: Kommunikationsplatine - Platinenlayout

5.3 Konfiguration der Hardware

Dieser Abschnitt beschreibt wie die am Telemetriesystem beteiligten Geräte eingerichtet wurden und welche Schritte dazu nötig waren.

GPS

Bei der Konfiguration des GPS im Programm "u-center" werden unter View / Configuration View folgende Einstellungen gemacht und vor jedem Seitenwechsel der "Send" Button des Configuration Views benutzt, um die Änderung an das GPS zu senden.

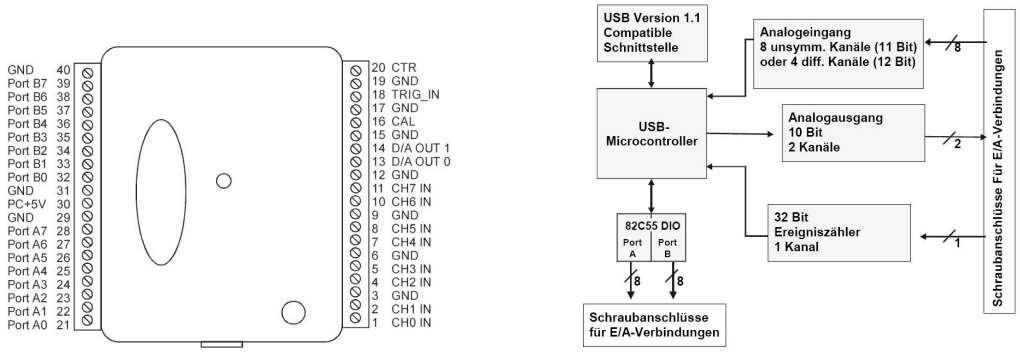
- RXM:
  - GPS Mode: "2 -High Sensitivity"
  - Power Mode: "0 - Normal"
- Rate:
  - Measurement Period: "500ms"
  - Navigation Rate: "1" cycle
- SBAS:
  - Subsystem: "Enabled"
  - Services: aktiviert
  - Testmode: aktiviert
  - Ranging: aktiviert
  - Apply SBAS Correction: aktiviert
  - Apply Integrity Information: aktiviert
  - Search Channels 1
  - PRN Codes EGNOS
- NAV2:
  - Dynamic Plattform Model: "3 -Automotive"
  - Static Hold Threshold: "0.3" (m/s)
  - Allow Alamanac Navigation: aktiviert
  - Initial Min C/NO (Fix) "15"(dBHz)
  - Min C/NO (Nav): "10"(dBHz)
  - Enable RAIM checks: aktiviert
- CFG:
  - Save current configuration

5.3.1 Serielle Schnittstelle

MAN5100078.pdf Bei Tests war die Serielle USB Schnittstelle stets standardmässig auf 9600 baud eingestellt, was mit der Brennstoffzelle übereinstimmt, deshalb wurde an der Konfiguration keinerlei Änderung vorgenommen.

5.3.2 PMD

Mit Instacal aus dem Measurement Computing Softwarepack, sollten alle Kanäle des PMD per "Calibrate" / "A/D" kalibriert werden. Dabei muss die komplette Kalibrierungsprozedur auf allen Kanälen durchgeführt werden, um die bestmögliche Genauigkeit der Messergebnisse zu gewährleisten. (siehe Abbildungen 5.7)



(a) Beschaltung PMD **(b)** Blockschaltbild PMD

Abbildung 5.7: Beschaltung des PMD; Quelle [34]

5.4 Erstellung der Firmware

Dieser Punkt soll die Schritte dokumentieren, welche nötig sind, um ein zu dem System passendes Firmwareimage mit den für den Betrieb des PMD nötigen Änderungen zu erstellen. (siehe dazu [40])

5.4.1 Systemsoftware

Ein Überblick der wichtigsten Software, die für das Kompilieren der OpenWrt Buildumgebung benötigt wird und später bei der Softwareentwicklung hilfreich ist.

Erforderliche Software

Um die Firmware kompilieren zu können, muss auf dem Arbeitssystem nachfolgende Software installiert sein. Die zum Bauen erforderliche Software lässt sich unter Ubuntu dabei wie folgt installieren:
sudo apt-get install

- build-essential
- binutils

- flex
- bison
- autoconf
- gettext
- texinfo
- sharutils
- subversion
- libncurses5-dev
- ncurses-term
- zlib1g-dev
- gawk

Nützliche Software

Nachfolgend zwei Programme, die sich während der Entwicklung der Telemetriesoftware als besonders hilfreich erwiesen haben.

- vsftpd: Die mit dem Compiler erstellten Binaries der Telemetriesoftware sollen auf möglichst effiziente Weise auf das System transportiert werden. Dies gelingt am einfachsten mit einem Skript, das die Dateien von dem Arbeitsrechner auf das System lädt, dazu bietet es sich an den FTP Deamon vsftpd zu installieren. Nach der Installation sind noch einige Änderungen nötig, die in "/etc/vsftpd.conf" vorgenommen werden.
  Dazu wird "anonymous_enable=yes" auskommentiert und "local_enable=YES" eingefügt. Auf dem System können per "wget ftp://user:password@host/directory/file" die Binaries dann von dem Arbeitsrechner heruntergeladen werden und sind einem "chmod 755 file" auch ausführbar.
- midnightcommander: Der midnightcommander hat sich sowohl auf dem Entwicklungs- als auch auf dem Telemetriesystem zum schnellen und einfachen Kopieren, Editieren und Dekomprimieren von Dateien als auch bei dem Löschen von gefüllten Verzeichnissen als sehr hilfreich erwiesen.

5.4.2 OpenWrt SVN Checkout

Mit Subversion kann der Code des jeweils aktuellen OpenWrt Entwicklungsstandes (trunk) oder ein normales Release (kamikaze_7.09), welches sich per Browser aus dem Repository herussuchen lässt, heruntergeladen werden. Dazu wird ein OpenWrt Arbeitsverzeichnis angelegt und dann folgende Befehle darin ausgeführt.

- svn checkout https://svn.OpenWrt.org/OpenWrt/trunk kamikaze
- svn checkout https://svn.OpenWrt.org/OpenWrt/packages packages

Danach sind sowohl der OpenWrt Code als auch die dazu passenden Programmpakete vorhanden, es hat sich dabei der Übersichtlichkeit und Verwaltbarkeit wegen als Vorteil erwiesen, die Pakete einzeln nach Bedarf in kamikaze/package/ zu kopieren.

Festlegung der Zielplattform

Bevor die Firmware für das System kompiliert werden kann, muss noch festgelegt werden auf welcher Hardware diese später benutzt werden soll. Daher muss im kamikaze Verzeichnis per "make menuconfig" das OpenWrt Buildsystem konfiguriert werden, für den ASUS WL-500GP mit Wistron CM9 WLANmodul sind dabei folgende Angaben zu der Hardware zu machen:

- Target System: "Broadcom BCM947xx/953xx [2.6]"
- Target Profile: "Atheros Wifi (default)"
- Target Images: "squashfs"
- Build the OpenWrt Image Builder: ja
- Build the OpenWrt SDK: ja

Unter Image Configuration ist die IPAdresse des Routers vorkonfigurierbar, bei dem Telemetriesystem ist dies 192.168.10.254 / 255.255.255.0, um Konflikte mit anderer während der Entwicklung vorhandener Hardware zu vermeiden.
Zusätzlich müssen die passenden Kernelmodule für den USB-Stick, PMD-1208FS und die seriellen Schnittstellen der Brennstoffzelle und des GPS angegeben werden, damit diese bei dem Kompilieren der Firmware gleich mit in das Firmwareimage geschrieben werden. Für eine Liste, der unter dem Menüpunkt "Kernel modules" mit einem "*" auszuwählenden Pakete siehe Anhang A1. Danach kann die Konfiguration beendet und bei der Abfrage gespeichert werden.

5.4.3 Paketmanager

Für das Paketmanagement ist auf einem OpenWrt basierten System der Paketmanager "ipkg" installiert. Eine Liste aller Pakete, die bei dem Einbau des System in das Fahrzeug installiert waren, ist in Anhang A2 zu finden.

- "ipkg list installed" Listet die installierten Pakete auf
- "ipkg list" Listet alle verfügbaren Pakete auf
- "ipkg install Package" Installiert ein Paket auf dem System
- "ipkg update" Aktualisiert die Liste der verfügbaren Pakete

5.4.4 Kernelkonfiguration

Damit der PMD-1208FS sowohl von dem System als HID Device erkannt wird, als auch im laufenden Betrieb keine USB-Resets mehr auslöst, müssen in zwei Bereichen nachfolgend beschriebene Konfigurationsänderungen am Kernel vorgenommen werden. Dazu wird im kamikaze Verzeichnis "make kernel_menuconfig" aufgerufen, und die nachfolgenden Änderungen in dem Menü durchgeführt.

USB

- Unter "Device Drivers" / "USB Support" müssen folgende experimentelle Optionen aktiviert werden.
- Full speed ISO transactions (EXPERIMENTAL)
- Root Hub Transaction Translators (EXPERIMENTAL)
- Improved Transaction Translator scheduling (EXPERIMENTAL)

USB-HID RAW-Devices

Unter "Device Drivers" / "HID Devices" muss "/dev/hiddev raw HID device support" aktiviert werden.

5.4.5 Firmwarekompilierung

Nachdem alle Änderungen und Konfigurationen durchgeführt sind, kann mit dem Aufruf von "make" die gesamte OpenWrt Umgebung inklusive des Firmwareimages und aller ausgewählten Pakete kompiliert werden. Nach einem fehlerfreien Compilevorgang befindet sich in "kamikaze/bin" die fertige Firmware.

5.4.6 Firmwareinstallation

Die Firmware kann per TFTP oder alternativen Methoden auf dem bis dahin mit der Originalfirmware arbeitenden Router wie in (siehe dazu [41]) beschrieben, installiert werden.
Nach einer Wartezeit von sechs Minuten kann das System dann per Telnet von dem LAN-Anschluss aus erreicht werden, um mit passwd ein neues Systempasswort zu setzen. Dies ist nötig, ist um anschließend per SSH auf dem System arbeiten zu können.

5.5 Systemkonfiguration

Um komfortabel mit dem System arbeiten, Software dafür entwickeln und es schliesslich zum Einsatz bringen zu können, sind weitere Anpassungen an der Schnittstellenkonfiguration und der Firewall nötig.

5.5.1 Schnittstellen

LAN und WAN

Die in "/etc/config/network" vorgenommene Schnittstellenkonfiguration für WAN und LAN ist in Anhang A3 zu finden. Das WAN Interface ist auf DHCP konfiguriert, während das LAN Interface mit der bei der Firmwareerstellung festgelegten IP-Konfiguration arbeitet.
Die Firewallkonfiguration für "/etc/firewall.user" in der die SSH und GPSD Ports zu Entwicklungszwecken zum WAN hin erreichbar sind, ist im Anhang A4 zu finden.

WLAN

Die Konfiguration aus "/etc/config/wireless" ist in Anhang A5 zu finden. Der Hexschlüssel 'key1' '65636f7465616d68617765636f' entspricht dabei dem 13 Zeichen Schlüssel "ecoteamhaweco", die Einstellungen werden dabei aus folgenden Gründen gewählt:

- Kanal 11 weil er nahezu überall verwendet werden darf.
- Modulation 11b weil sie im Gegensatz zu 11g eine gesteigerte Zuverlässigkeit gegenüber Rauschen und schmalbandigen Störungen bietet.
- Diversity 0 weil es nur eine Antenne am Mainanschluss der Karte gibt.
- Tx-/Rxantenna entsprach laut Tests dem Mainschluss der Karte.
- Disabled 0 um die Schnittstelle benutzen zu können.

5.5.2 Paketarchive

Damit für das System möglichst viele Pakete direkt und ohne Kompilieren installierbar sind, sollten in "/etc/ipkg.conf" wie in Anhang A6 beschrieben noch Links auf einige Paketarchive eingetragen und "ipkg update" aufgerufen werden.

5.5.3 Systemdienste

Damit die Telemetriesoftware bei dem Starten der Box als Dienst aufgerufen wird, muss unter "/etc/init.d/" ein File mit dem Namen teled und dem in Anhang A7

beschriebenen Inhalt angelegt und per "chmod 755 teled" ausführbar gemacht. Zusätzlich muss "ln -s /etc/init.d/teled /etc/rc.d/S97teled"
aufgerufen werden. Dadurch wird ein Symlink in "/etc/rc.d/" names "S97teled" auf das File in "/etc/init.d/" angelegt. Es hat sich bewährt solche Änderungen jeweils per "ls -l" zu kontrollieren, da auf diese Weise Dateien, wie der angelegte Link, mit seinem Ziel angezeigt werden.

5.5.4 Weitere Software

Auch sehr nützlich ist der NTPD, der in dem Zusammenspiel mit dem GPSD die Zeit aus den vom GPS übermittelten Daten als Systemzeit übernehmen kann. Der GPSD in OpenWrt ist dazu jedoch standardmässig nicht in der Lage, da ihm aus Platzersparnisgründen diese und verschiedene andere Funktionen nicht einkompiliert wurden.

5.6 Telemetriesoftware

Die Telemetriesoftware besteht aus vier Prozessen, die nachfolgend näher erklärt werden sollen. Drei erledigen die Messdatenaufnahme, während der vierte Prozess das Starten der anderen Prozesse und Systemfunktionen, wie das Mounten, sowie bei den Daten das Zusammenfassen, Loggen und Senden erledigt. Alle drei Prozesse schicken dazu ihre Datensätze zur weiteren Verarbeitung an den auf UDP localhost:10000 empfangenden Server. (siehe Abbildung 5.8) (Quellen zu UDP sowie nicht blockierenden I/O-Sockets siehe [44];[45])

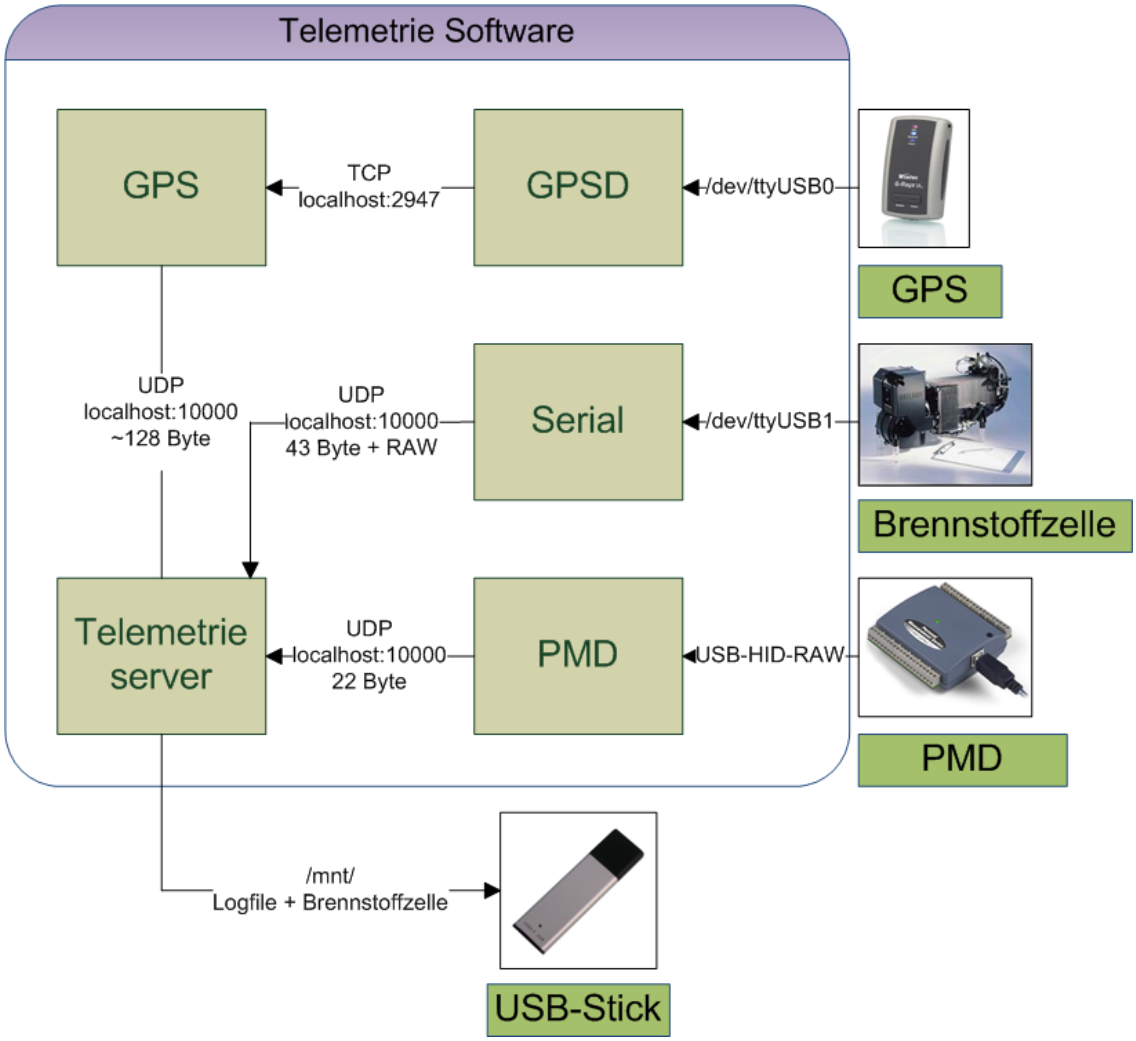


Abbildung 5.8: Blockschaltbild Telemetriesoftware

5.6.1 GPS

Aufgabe des GPS Prozesses ist es, aktuelle Daten wie die Zeit, Position und Geschwindigkeit zu liefern. Da es zu diesem Zweck schon einen Spezialisten wie den GPSD gibt, wird dieser in Verison 2.36 verwendet und das gegenüber dem NMEA Format wesentlich einfachere Protokoll des GPSD durch einen zweiten GPS Prozess gefiltert. Sämtliche Leerzeichen werden für das Protokoll gegen ";" ersetzt, ein Timestamp davor gesetzt und die Daten bei einem Newline per UDP an den Server weitergereicht.

GPSD

Der GPSD wird auch deswegen eingesetzt, weil er sich selbständig auf die Schnittstellengeschwindigkeit des GPS einstellt und sich allgemein als recht zuverlässig arbeitende Software erwiesen hat. Nachfolgend das Format, welches der GPSD zur Ausgabe der GSA Zeile benutzt. (siehe dazu Quelle [42])

tag - A tag identifying the last sentence received. For NMEA devices this is just the NMEA sentence name; the talker-ID portion may be useful for distinguishing among results produced by different NMEA talkers in the same wire.

- timestamp - Seconds since the Unix epoch, UTC. May have a fractional part of up to .01sec precision.
- time error - Estimated timestamp error (%f, seconds, 95% confidence).
- latitude - Latitude as in the P report (%f, degrees).
- longitude - Longitude as in the P report (%f, degrees).
- altitude - Altitude as in the A report (%f, meters). If the mode field is not 3 this is an estimate and should be treated as unreliable.
- horizontal error estimate - Horizontal error estimate as in the E report (%f, meters).
- vertical error estimate - Vertical error estimate as in the E report (%f, meters).
- course over ground - Track as in the T report (%f, degrees).
- speed over ground - Speed (%f, meters/sec). Note: older versions of the O command reported this field in knots.
- climb/sink - Vertical velocity as in the U report (%f, meters/sec).
- estimated error in course over ground - Error estimate for course (%f, degrees, 95% confidence).
- estimated error in speed over ground - Error estimate for speed (%f, meters/sec, 95% confidence). Note: older experimental versions of the O command reported this field in knots.

estimated error in climb/sink - Estimated error for climb/sink (%f, meters/sec, 95% confidence).
mode - The NMEA mode (%d, ?=no mode value yet seen, 1=no fix, 2=2D, 3=3D). (This field was not reported at protocol levels 2 and lower.)

Datenverarbeitung

Der "gps" Prozess forkt direkt nach dem Start, um sich von seinem Mutterprozess zu lösen. Danach baut er eine Verbindung zu dem auf TCP localhost:2947 laufenden GPSD auf und speichert jede Zeile, die dieser überträgt. Da der GPSD mehr als nur die GGA Zeile sendet, wird diese aus den Daten herausgefiltert und zusätzlich der

Unix Timestamp extrahiert, um daraus Uhrzeit und Datum für das Protokoll zu errechnen. Dies ist wegen des Fehlens einer Echtzeituhr auf dem System nötig. Jedes Mal, wenn die Zeile gefunden und Datum und Uhrzeit daraus errechnet worden sind, werden die Daten als Paket an den auf UDP localhost:10000 laufenden Serverprozess übermittelt. Für den Aufbau der TCP Verbindung und das Einlesen der Daten orientiert sich das Programm dabei stark an gpspipe aus dem GPSD Paket.

5.6.2 Brennstoffzelle

Der für die Brennstoffzelle zuständige Prozess "serial" forkt direkt nach seinem Start und liest dann die Daten von dem zur seriellen Schnittstelle gehörenden File "/dev/ttyUSB1". Da die Schnittstelle bereits standardmässig passend auf 9600 Baud 8N1 konfiguriert ist, braucht dort nichts mehr verändert werden.

Protokoll

Die Kommunikation mit der Brennstoffzelle ist asynchron, dabei wird alle 200ms ein Datensatz von der Brennstoffzelle gesendet, diese Daten sind im SLIP Protokoll (Serial Line Internet Protocol, Internet RFC 1055) kodiert. Das SLIP Protokoll enthält ein Einbyte tag (0xC0) am Anfang und am Ende jedes Datensatzes. Drei Escape Characters, 0xDB, 0xDC, und 0xDD sind dabei für Fälle nötig, in denen ein 0xC0 in der Mitte eines Datensatzes auftaucht. Die Nachricht kann sich dadurch theoretisch auf bis zu 84Byte verlängern. Der Standarddatensatz der Brennstoffzelle enthält dabei immer mindestens 43Bytes, diese können sich bei Diagnoseoperationen nochmals um bis zu 100Bytes verlängern; können jedoch für dieses Projekt, bis zum Eintreffen des nächsten Starttags genauso gut ignoriert werden. (siehe Abbildung 5.9)

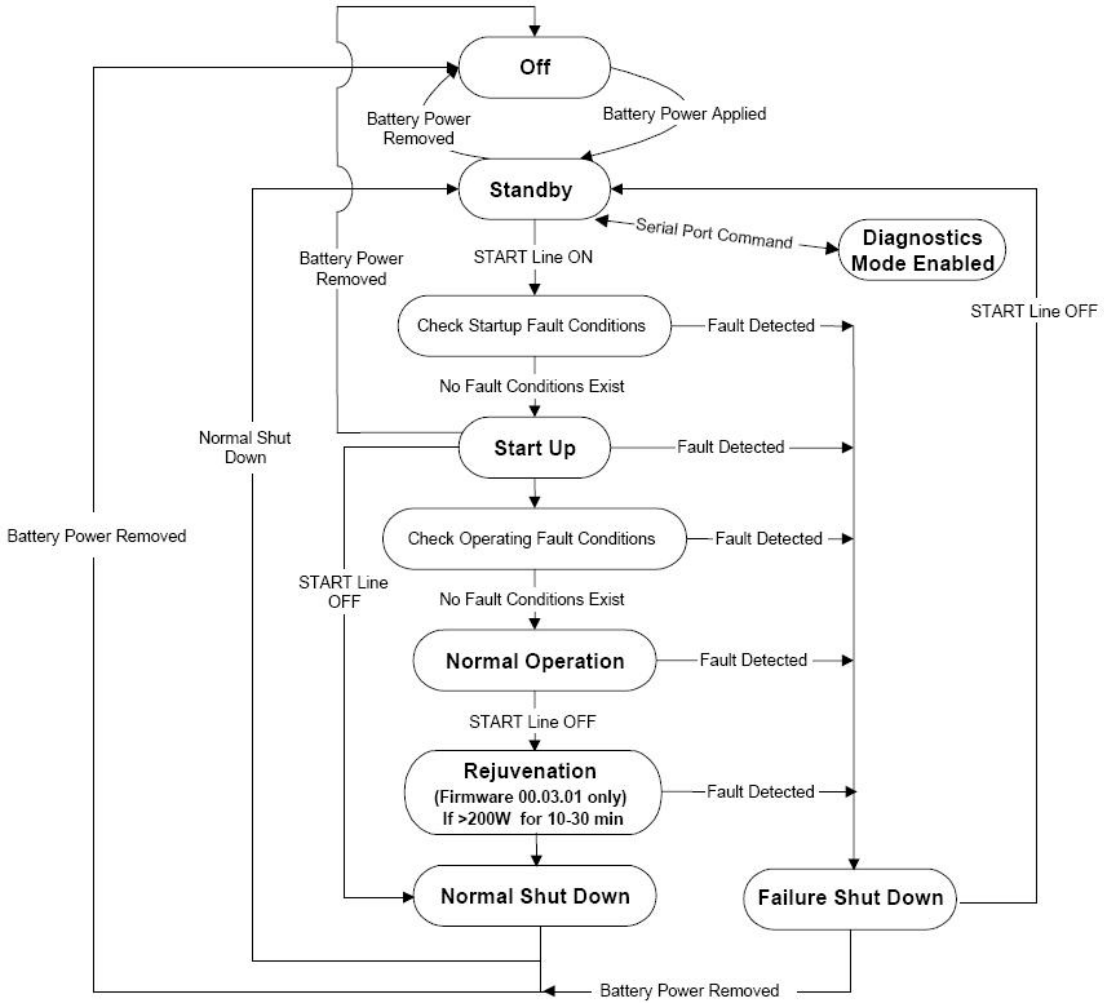


Abbildung 5.9: Operationszustände der Brennstoffzelle

Datenverarbeitung

Die von der Zelle eingetroffenen Daten werden einmal direkt gespeichert und als Rohdatensätze variabler Länge zum Server geschickt und einmal vom SLIP Protokoll befreit und als 43Byte Nachrichten ebenso zum Server geschickt.

5.6.3 PMD

Der Prozess "pmd" ruft wiederum als erstes fork() auf und initialisiert nach dem Starten zunächst die Digitalports, PortA als Ausgang und PortB als Eingang, danach wird der integrierte Counter auf 0 gesetzt. Der Prozess basiert dabei auf einem

unter der GPLV2 stehenden Softwarepaket für den PMD, (siehe dazu [43]) das viele Versionen der Personal Measurement Devices unterstützt und eine gute Grundlage für diesen Prozess gelegt hat.

Datenabfrage

Nachdem alle Ports initialisiert sind, werden zunächst die acht Analogeingänge ausgelesen und die 16Bit Werte zusammen in einem 16Byte Array abgelegt. Da die vom PMD kommenden Werte eine im 2er Komplement abgelegte signed 11Bit Zahl enthalten, wird diese in den Abfrageroutinen zunächst durch 16 geteilt (4Bit Downshift) und dann 1024 vom Ergebnis abgezogen, da keine Werte kleiner 0V gemessen werden müssen. Das Endergebnis ist ein 10Bit Wert für einen Messwert zwischen 0V und 10V. Danach werden der Counter und PortB, an dem die Schalter hängen, ausgelesen. Wenn alle Werte vorhanden sind, werden zunächst die 16Byte für die Analogwerte in das Paket kopiert, danach die 4Byte für den 32Bit Counter und je 1Byte für die Digitalschnittstellen, also den 8Bit Eingang und den 8Bit Ausgang. Das Resultat ist ein 22Byte Paket das an UDP localhost:10000 gesendet wird.

Zu beachten ist dabei, dass für die Analogeingänge im Singlendedmodus der richtige Gain (SE_10_00V / 0x0) und die Kanalnummern 8 bis 15 abgefragt werden müssen. Die Kanalnummern 0 bis 3 sind hingegen für den Differentialmodus, reserviert in dem je zwei Eingänge zu einem Kanal zusammengefasst sind.

5.6.4 Telemetrieserver

Der Telemetrieserver "udps" ist der Prozess, der als einziger direkt per init.d vom System aufgerufen wird, fork() ausführt, um sich von dem System zu lösen und danach selbst alle für den vollständigen Start des Telemetriesystems nötigen Aufgaben ausführt. (siehe Abbildung 5.10)

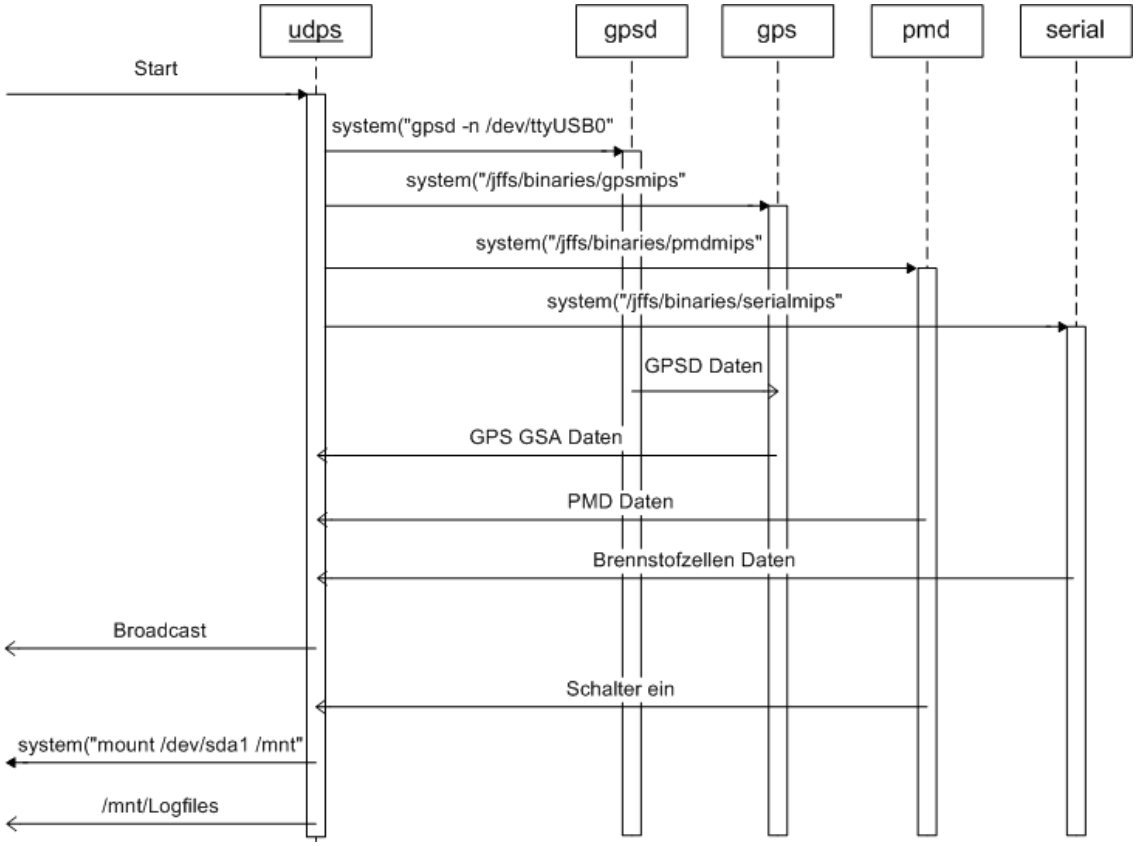


Abbildung 5.10: Sequenzdiagramm Telemetrieserver

Softwarestart

Als erstes werden per system() der GPSD und die anderen Prozesse wie GPS, PMD und Serial gestartet und "/dev/sda1" unmounted, um nach einem möglichen Automount das System in einen definierten Zustand zu bringen.

Datenempfang

Datenpakete empfängt der Server auf UDP localhost:10000, dabei werden die Datenpakete anhand ihrer Größe erkannt, 22Byte vom PMD, 43Byte von der Brennstoffzelle, 43 bis 84Byte Rohdaten im SLIP Format von der Brennstoffzelle und Nachrichten größer 84 Zeichen vom GPS.

Datenpakete

Die Datenpakete für das Logfile und den Broadcast werden dabei in folgender Reihenfolge zusammengesetzt, die GPS Nachricht mit variabler Länge, 43Byte für die Brennstoffzelle und 22Byte für den PMD, zwischen den Nachrichtenteilen wird jedesmal mit ";;;" getrennt, um Fehlerkennungen bei der Auswertung zu vermeiden.

Logfiles

Es werden zwei Logfiles geschrieben, eins, das den kompletten Satz aus GPS, Brennstoffzellendaten und PMD schreibt, sowie ein Logfile, das die unbearbeiteten Daten der Brennstoffzelle schreibt, damit diese später mit dem dazu ausgelieferten Originalprogramm einlesbar sind. So sind im Fehlerfall auch noch die längeren, nicht dokumentierten Diagnosemeldungen der Brennstoffzelle, auswertbar.

Broadcast

Der Broadcast wird auf UDP 192.168.10.255:10005 verschickt, er enthält unverändert die Zeile, die auch in das Logfile geschrieben wird und ist sowohl an den LAN-Schnittstellen als auch im WLAN zu empfangen. Dadurch, dass die Pakete nur ungefähr 200Byte lang sind und mit einer Rate von 1MBit/s im Adhoc WLAN verschickt werden, sind diese auch auf grössere Entfernungen noch gut zu empfangen. Aufgrund der etwas ungünstigen Einbaulage ergeben sich 500m Reichweite, wenn das Fahrzeug ohne Hindernisse in Sicht ist.

5.6.5 Kompilierung der Tasks

Für das Kompilieren der Tasks wird ein Skript, namens makemips.sh, angelegt und ausführbar gemacht, das diese Aufgabe einfach und schnell erledigt.

```
INCLUDE=/home/user/OpenWrt/kamikaze/toolchain_build_mipsel/
linux/include/linux
GCC=/home/user/OpenWrt/kamikaze/staging_dir_mipsel/bin/mipsel-linux-gcc
KERNEL=/home/user/OpenWrt/kamikaze/build_mipsel/linux

$GCC -O2 -I$KERNEL/include udpserver.c -o udpsmips
$GCC -O2 -I$KERNEL/include pmd.c -o pmdmips
$GCC -O2 -I$KERNEL/include serial.c -o serialmips
$GCC -O2 -I$KERNEL/include gps.c -o gpsmips
```

Wenn es Probleme mit Bibliotheken gibt, kann es sich trotz des größeren Platzbedarfs auch lohnen die Option –static zu verwenden, um das Programm unabhängiger von den auf dem Zielsystem vorhandenen Bibliotheken zu machen.

5.6.6 Transfer auf das System

Zum schnellen Transfer der kompilierten Programme auf das System wird mit "mkdir /jffs/binaries" ein Verzeichnis angelegt und dort das Skript "down.sh" mit nachfolgendem Inhalt angelegt und ausführbar gemacht.

```
rm gpsmips
rm pmdmips
rm serialmips
rm udpsmips
wget ftp://user:password@host/beispielcode/test/netzwerk/gpsmips
wget ftp://user:password@host/beispielcode/test/netzwerk/pmdmips
wget ftp://user:password@host/beispielcode/test/netzwerk/serialmips
wget ftp://user:password@host/beispielcode/test/netzwerk/udpsmips
chmod 755 gpsmips
chmod 755 pmdmips
chmod 755 serialmips
chmod 755 udpsmips
```

Dadurch können nach jedem Kompilieren ohne großen Aufwand, die alten Binaries gelöscht, die neuen heruntergeladen, ausführbar gemacht und getestet werden.

5.7 Betriebssystemsicherheit

Die Systemsicherheit wird durch den Einsatz der Protokolle SSH bei der Administration und Wartung sowie SCP beim Transfer von Updates gewährleistet, am WAN Anschluss werden durch den aktivierten Paketfilter alle Verbindungsversuche von aussen bis auf SSH- und GPSD-Verbindungen abgelehnt. Desweiteren werden alle zusätzlich über das WAN installierten Softwarepakete automatisch vom Paketmanager per Hashvergleich auf unautorisierte Änderungen überprüft. Bis auf den GPSD und den SSHD, die von den Entwicklern auch im Hinblick auf Sicherheit entwickelt wurden, sind ausserdem sämtliche Ports an localhost gebunden. Das WLAN wird per WEP gegen unbeabsichtigtes Verbinden geschützt, und das Netz so zusätzlich zur SSID und dem Kanal von anderen abgegrenzt. Die einzigen Daten, die das Gerät verlassen, sind das 10Hz WLAN Beacon sowie der 2Hz Telemetrie Broadcast.

5.8 Verwendung der Daten

In diesem Kapitel wird auf das Protokoll eingegangen, das von der Telemetriebox gesendet, auf den USB Stick gespeichert und von der Telemetrieauswertesoftware ausgelesen wird.

5.8.1 Aufbau des Protokolls

Das Protokoll besteht aus drei Teilen. Dem GPS Protokoll, dem Brennstoffzellen Protokoll und dem PMD Protokoll. Diese werden hintereinander, mittels Trennzeichen ';;;' voneinander getrennt, verschickt. In dem Protokoll sind daher unterschiedliche Datenformate enthalten. Die Daten des GPS bestehen aus ASCII Text mit variabler Länge, die Daten der Brennstoffzelle aus 4Byte Floats Little-endian mit fester Länge (siehe Abbildung 5.11) und die Daten des PMD aus jeweils 2Byte Daten für die analogen Eingänge, 4Byte Daten für den Counter und jeweils 1Byte für die digitalen Ein- und Ausgänge auch mit fester Länge. Da die Daten des GPS keine feste Länge haben, d.h. die Werte wie Höhe und Richtung als ASCII und nicht als feste Floatwerte übertragen werden, wird hinter jedem Wert, den das GPS ausgibt, ein ';' zur Trennung eingefügt. Im Ganzen sieht das Protokoll so aus: (gps;;;serial;;;pmd;;;n0) Für das GPS Protokoll siehe Kapitel 5.6.1 GPSD.

Tag	Status	Fail Code	Warning Bitmap	Last Command Acknowledge	Stack Temperature	Stack Voltage	Stack Current	Hydrogen Pressure

Hydrogen Concentration	Cumulative Hydrogen Consumption	Oxygen Concentration	Ambient Temperature	Purge Cell Voltage

Additional diagnostic and fault code bytes (0 to 100 extra bytes)	Check Sum	Tag

Abbildung 5.11: Aufbau des Brennstoffzellen Protokolls [39]

***Brennstoffzelle (feste Länge):

43Bytes:
1 Tag
2 Status
3 Fail Code
4 Warning Bitmap
5 Last Command Acknowledge
6 - 9 Stack Temperatur

10 - 13 Stack Spannung
14 - 17 Stack Strom
18 - 21 Wasserstoffdruck
22 - 25 Wasserstoffkonzentration
26 - 29 kumulativer Wasserstoffverbrauch
30 - 33 Sauerstoffkonzentration
34 - 37 Umgebungstemperatur
38 - 41 Purge Zellenspannung
42 Checksum
43 Tag

***PMD (feste Länge):

22Bytes pro PMD

Analog 1 bis 8: je 2Bytes = 16Bytes
Counter : 4Bytes
Digital I/O A+B: je 1Byte = 2Bytes

5.8.2 Daten Logfiles / WLAN

Das Protokoll wird zeilenweise in das Logfile geschrieben, so dass ein ganzer Datensatz mit dem Wagenrücklauf beendet ist. Über das WLAN wird jeweils eine Zeile pro Paket übertragen.

5.9 Telemetriedatenauswertung

Die Auswertung geschieht mittels Labview in zwei Programmen. Eines, um die Telemetrie des Fahrzeuges während der Fahrt über das WLAN auszulesen und die Daten direkt angezeigt zu bekommen. Das zweite, um die Logdatei, die die Telemetriebox im Fahrzeug aufzeichnet, einzulesen und die Daten nach der Fahrt zu analysieren.

Die Aufgaben von Labview sind:

- Verbindung zur Telemetriebox herstellen
- Einlesen der Daten über WLAN (UDP) oder via Logdatei
- Auswertung der empfangenen Daten
- Visualisieren der Daten

5.9.1 Dekodierung und Auswertung der Telemetriedaten

Um die Daten der Telemetriebox zu empfangen, öffnet Labview eine UDP Verbindung, an der es an Port:10005 auf Daten wartet. Da UDP eine verbingungslose Übertragungstechnik ist, wartet der Sender (die Telemetriebox) nicht auf eine Antwort. Das hat den Vorteil, dass immer Daten gesendet werden und kein Stau entstehen kann, wenn die Daten mal durch z.B. Verbindungsproblemen abreißen. Im Gegensatz zu TCP werden bei UDP Daten ohne Verbindung empfangen. Ein Vorteil ist, früher Daten von dem vorbeifahrenden Fahrzeug zu bekommen, bevor es wieder aus dem Funkbereich gefahren ist. Sobald Daten empfangen werden, werden die Daten in Labview visualisiert. Die zweite Software fordert beim Starten auf, eine Logdatei, die die Telemetriebox im Fahrzeug aufzeichnet, einzuspielen. Anhand dieser Daten, werden verschiedene Verlaufsdiagramme gezeichnet, in denen man in zeitlicher Abhängigkeit den Verlauf des Fahrzeuges und dessen Sensoren verfolgen kann.

Verbindung mit der Telemetriebox herstellen

Eine Verbindung wird mittels der in Labview beinhalteten Vi's für die Netzwerkverbindungen hergestellt. (siehe Abbildung 5.12) Die Übertragung von der Telemetrie Box ist über UDP 192.168.10.255 und dem Port 10005 voreingestellt. Durch die einfache Handhabung des Programmes ist die Anpassung an andere Netzwerke einfach und ohne großen Aufwand möglich. Um die Logdatei einlesen zu können, wird ein Sub VI zum Einlesen einer Tabellenkalkulationsdatei (siehe Abbildung 5.13) verwendet.

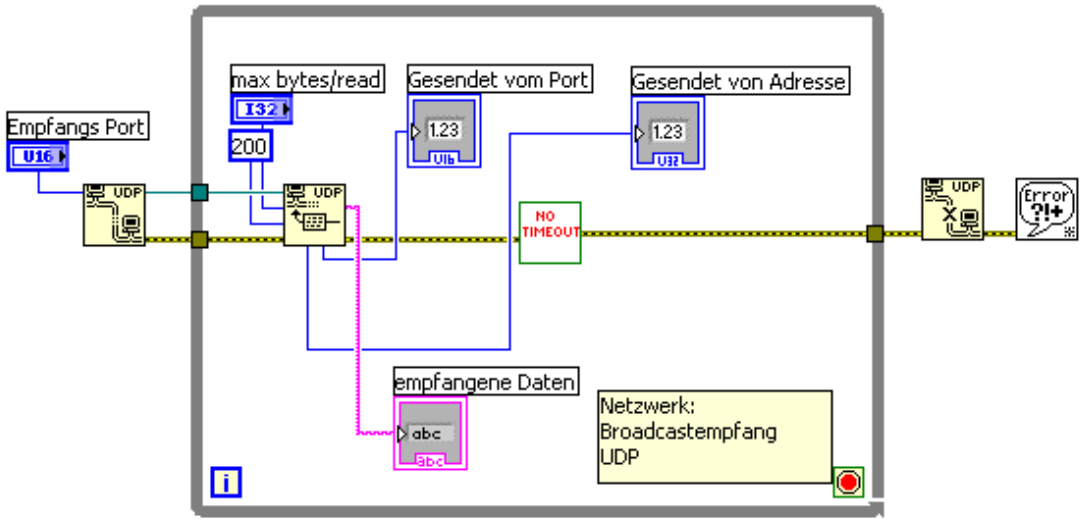


Abbildung 5.12: UDP-Netzwerkverbindung in LabView

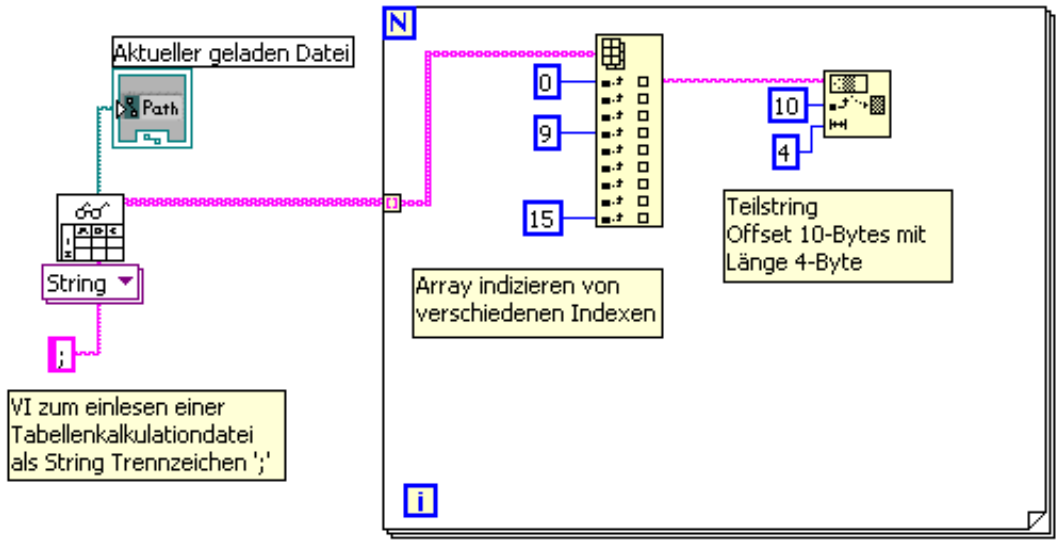


Abbildung 5.13: Dateieinlesen in LabView

Einlesen der Daten

Beim Einlesen der Daten per Netzwerk wird ein kontinuierlicher Datenstring erzeugt, der in einer Schleife alle 500ms abgefragt wird. Diese 500ms ergeben sich aus der Abfragezeit der einzelnen Telemetrieboxkomponenten und den daraus resultierenden Übertragungsintervallen. 500ms GPS, 200ms Brennstoffzelle und 200ms PMD. Als Übertragungsintervall werden die GPS Nachrichten verwendet, da das GPS auch die Zeit mit überträgt.

Aus der Logdatei wird ebenfalls ein Datenstring erzeugt, jedoch als ein 2D-Array. Da die einzelnen Datenpakete, GPS, Brennstoffzelle und die PMD Sensordaten mittels Trennzeichen voneinander getrennt sind, wird aus diesen Daten beim Aufrufen ein 2D-Array erzeugt. Dieses hat den Vorteil, dass alle einzelnen Sensorwerte (Arrayplatz 2) zu einer bestimmten Zeit (Arrayplatz 1) im Programm genau zuzuordnen sind. Die genaue Zeit wird vom GPS geliefert.

Auswertung der Empfangenen Daten

In dem Datenstring sind unterschiedliche Datenformate enthalten. Die Daten des GPS bestehen aus ASCII Text, die Daten der Brennstoffzelle aus 4Byte Floats Little-endian und die Daten des PMD aus jeweils 2Byte Daten für die analogen Eingänge, 4Byte Daten für den Counter und jeweils 1Byte für die digitalen Ein- und Ausgänge. Da der Counter vom Hallsensor immer weiter hochgezählt wird und nicht resettet wird, mittelt eine SubVI über die letzten 5 eingegangenen Counterstände, eine Durchschnittsgeschwindigkeit. (siehe Abbildung 5.14) Für die Konvertierung von Strings in Fließkommazahlen, ist es erforderlich, durch "String suchen und ersetzen" die in dem String enthaltenen "." durch "," zu ersetzen. (siehe Abbildung 5.15)

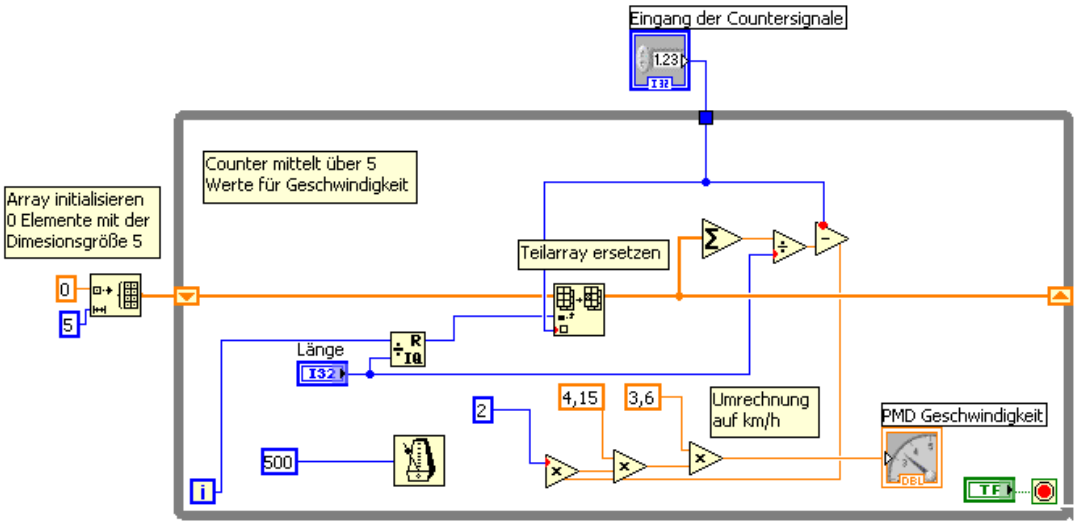


Abbildung 5.14: messung der Geschwindigkeit über counter in LabView

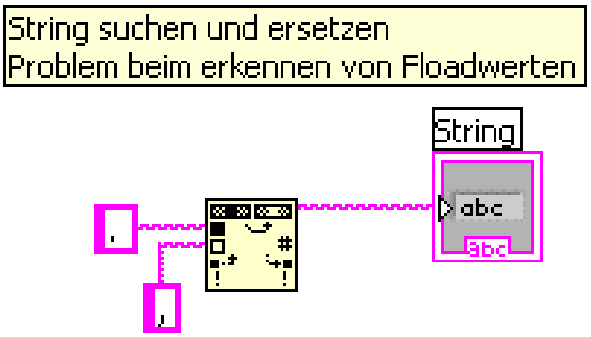


Abbildung 5.15: erforderliches suchen und ersetzen in Strings in LabView

Da die Daten unformatiert von der Telemetriebox ausgegeben werden, also keinerlei Anpassung z.B. der Temperaturen an den Nullpunkt geschieht, werden diese Umrechnungen in Labview vorgenommen. Dadurch ist ein einfaches Hinzufügen oder Wechseln von Sensoren durch Anpassungen der Software möglich.

- Software fürs WLAN: Der per Netzwerk übertragene Datenstring, wird daher mittels "Muster Suchen" in einzelne Stringpakete aufgeteilt, die die entspechenden Datenpakete zur Auswertung enthalten. Da die Daten des GPS

keine feste Länge haben, und ein ';' zur Trennung eingefügt ist, werden diese mittels vi (Match Pattern) aus dem String gesucht und die Daten bis zum nächsten Trenner ausgewertet. (siehe Abbildung 5.16) Da die Brennstoffzellen- und PMD Daten eine feste Länge besitzen, werden dieBytes gezählt und die einzelnen Werte ausgewertet. Die Brennstoffzellendaten müssen noch in Little-endian umgewandelt werden, da Labview mit Big-endian arbeitet, was durch ein Drehen des Strings in Labview einfach zu realisieren ist. Danach wird der String mittels "Type Cast" vi in das richtige Datenformat gecastet. (siehe Abbildung 5.17 und 5.18)

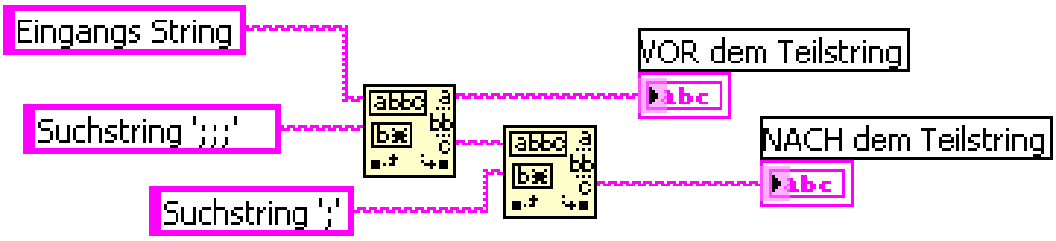


Abbildung 5.16: Such String in LabView

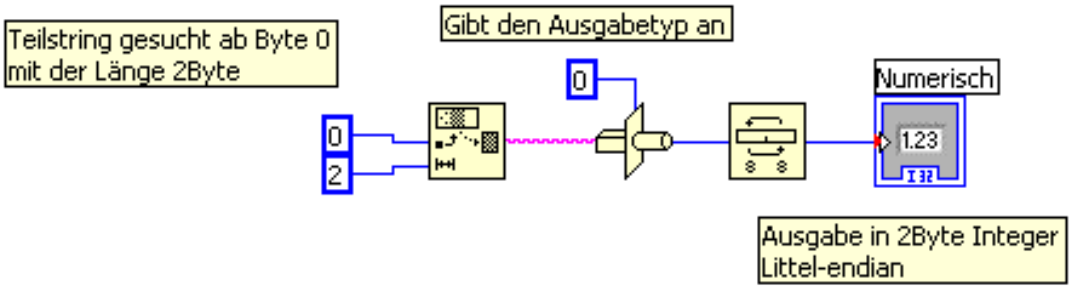


Abbildung 5.17: casten von 2Byte String und Korrektur der Endianness in LabView

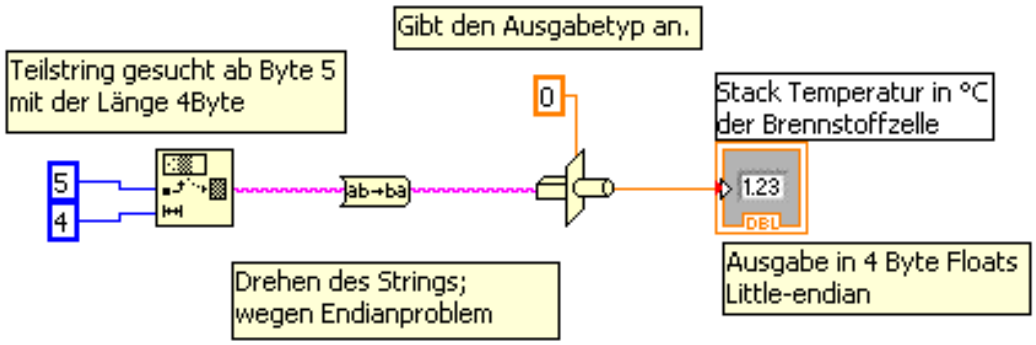


Abbildung 5.18: casten von 2Byte String und Korrektur der Endianness in LabView

- Software für Logdatei: In diesem Programm wird der 2D-Array Datenstring ähnlich wie der im vorherigen Programm aufgegliedert und verarbeitet. Jedoch geschieht dieses hier mittels (Index Array) VI, das das Array in seine einzelnen Teile zerlegt. Die Auswertung der Daten geschieht dann wie oben beschrieben.

Visualisieren der Daten

- Software fürs WLAN: Es gibt 8 Tabs auf der Oberfläche, die wegen der Übersicht und des Platzmangels aufgeteilt wurden.
  - Brennstoffzelle
  - GPS
  - PMD
  - PMD Verlaufsdiagramm
  - PMD Leistung
  - Vergleich PMD zu Brennstoffzelle
  - Netzwerk
  - Streckenführung

Bei dem GPS wird aus dem Datenpaket für die Richtung mittels "In Range and Coerce" eine Richtungsanzeige erzeugt. (siehe Abbildung 5.19) Zudem werden durch Bündelungen mehrerer Daten, Verlaufsdiagramme realisiert, die im Verhältnis zur Zeit z.B. die Geschwindigkeit und Höhe anzeigen. (siehe Abbildung 5.20) In dem SubVI "3D Curve" (siehe Abbildung 5.21) wird ein Streckenverlauf dargestellt, der die Höhe sowie die Breiten- und Längengrade der Position des GPS als Strecke aufzeichnet. Die Werte, die die Brennstoffzelle ausgibt, bestehen aus Wasserstoffdruck, Temperaturen, Stack-Spannung und Stack-Strom.

- Brennstoffzellensoftware für die Logdatei: Es gibt 6 Tabs auf der Oberfläche, die wegen der Übersicht und des Platzmangels aufgeteilt wurden.
  - Brennstoffzelle
  - Verlauf Leistung
  - GPS
  - PMD
  - PMD Verlauf
  - GPS Verlauf

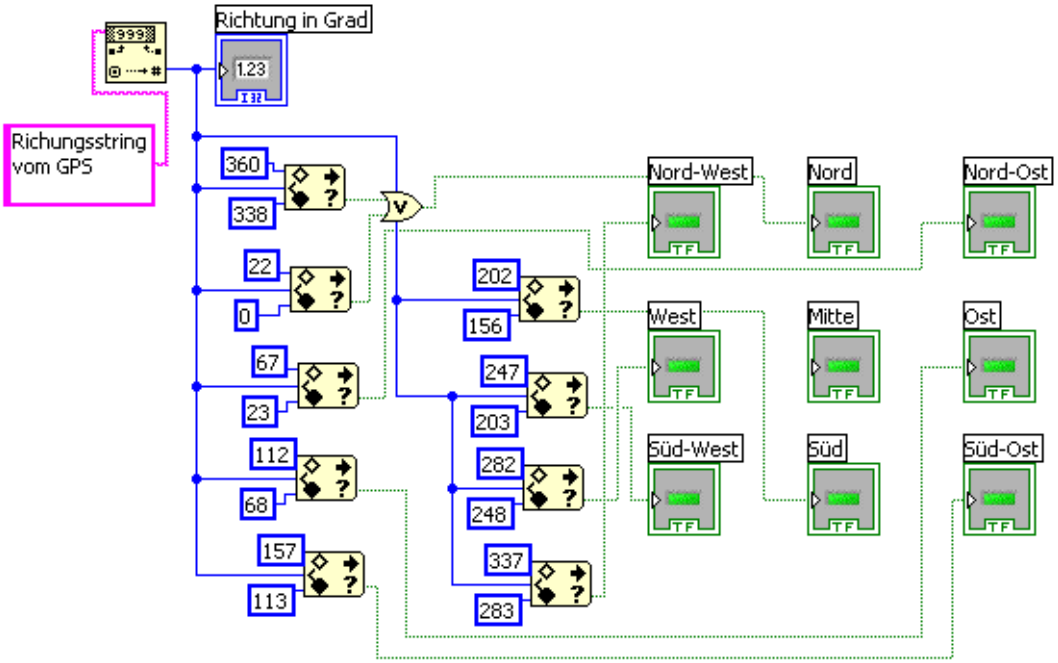


Abbildung 5.19: Richtungsanzeige in LabView

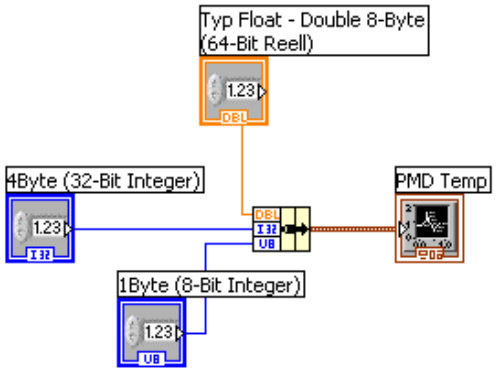


Abbildung 5.20: Verschiedene Wertebündelung in Verbindung eines Graphen in LabView

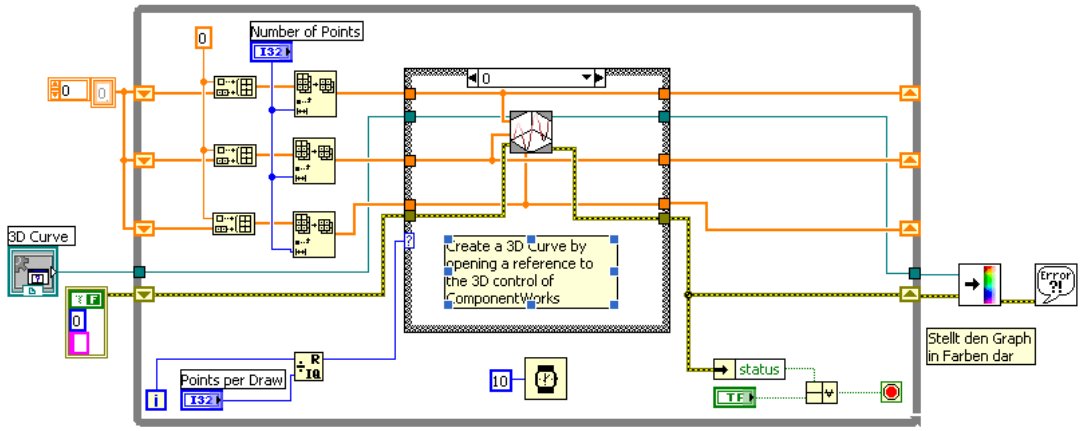


Abbildung 5.21: 3D-Graph in LabView

Beim Auslesen der Logdatei werden Verlaufsdiagramme für die Brennstoffzellendaten, die GPS Daten und die PMD Daten erzeugt. In den Diagrammen werden komplette Datensätze angezeigt, in die man hineinzoomen und sich Teile der einzelnen Messdaten ansehen kann. Die Diagramme beziehen sich auf die vom GPS gesendete Zeit. Zudem gibt es die Möglichkeit, die GPS und die PMD Daten in einer eigenen Zeitschleife, mittels Zeitregler verlangsamt oder in Echtzeit, ablaufen zu lassen. Dadurch ist es möglich die Fahrt im Nachhinein zu simulieren.

(weitere Informationen dazu im Kapitel 9, Anhang B)

6 Ergebnisse

Nachfolgend die Ergebnisse der Messkette, bestehend aus Sensorplatine, PMD und den angeschlossenen Sensoren.

6.1 Spannungsmessungen

Die Spannungsmessung auf der Sensorplatine ist über Spannungsteiler realisiert, die die Messergebnisse für die Brennstoffzellen- und Bordbatteriespannung in der folgenden Tabelle aufführt. (siehe Abbildung 6.1 & 6.2) Die Spannungsteiler errechnen sich aus den Ausgangsspannungen und dem Messbereich des PMD. Daraus ergibt sich ein Verhältnis von 1/6,8 bei der Brennstoffzelle und ein Verhältnis von 1/2 bei der Batterie.

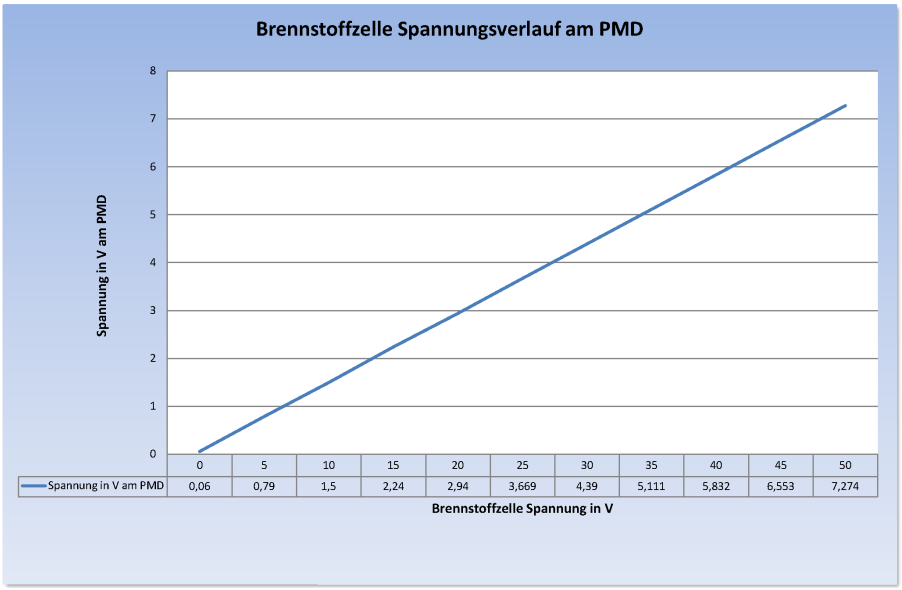


Abbildung 6.1: Spannungsmessung Brennstoffzelle

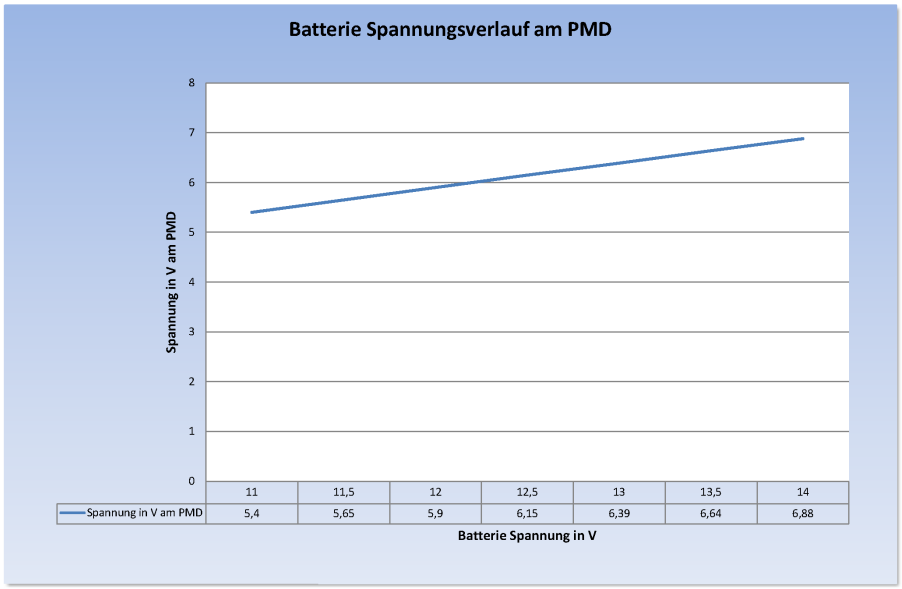


Abbildung 6.2: Spannungsmessung Batterie

6.2 Strommessung

Der Stromsensor liefert bei einem Strom von 1A, eine Spannungsdifferenz von 50mV bei einem Offset von Vcc/2. Daraus errechnet sich bei einem Strom von 20A eine Spannungsdifferenz von 1V, was 1/10 des Messbereichs entsprochen hätte. Das Signal wurde deswegen über einem Operationsverstärker, der als Differenzverstärker beschaltet ist, gemessen. Die Spannung verläuft bei einem Eingangsstrom von 0A bis 20A linear zwischen 6,85V und 3,14V. (siehe Abbildung 6.3) Durch den abfallenden Spannungsverlauf, ist der Messbereich von ca. 7V bis 0V größer als von 7V bis 10V des PMD Messbereichs. Die hohe Anfangsspannung entsteht durch den Offset des Stromsensors und die Versorgungsspannnung der 12V Bordbatterie.

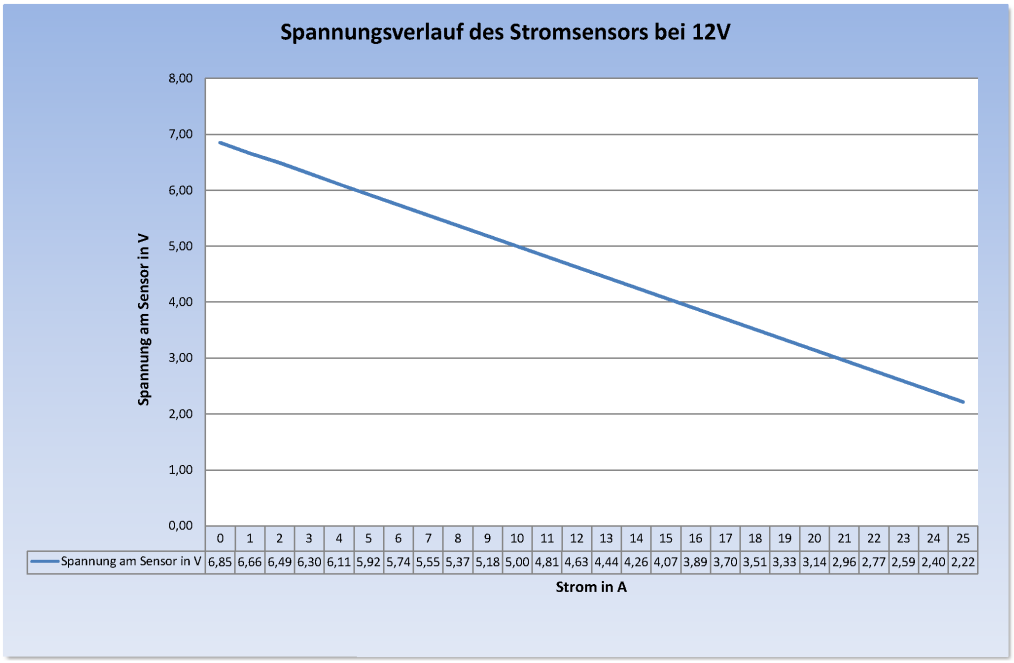
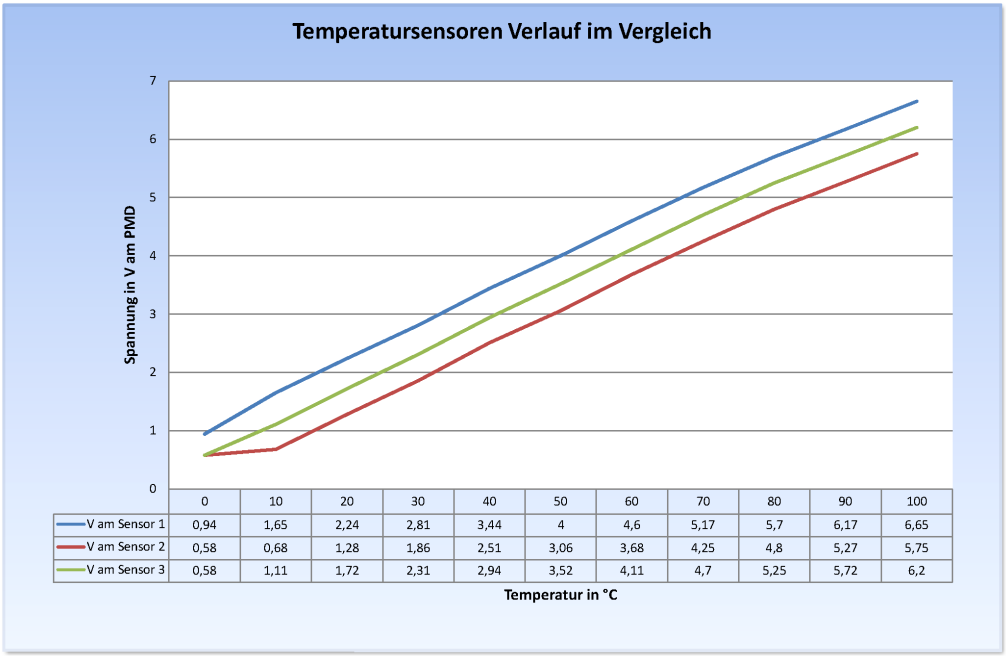


Abbildung 6.3: Strommessung

6.3 Temperaturmessung

Die drei PT1000 Temperatursensoren, sind in einem 1/2 Spannungsteiler realisiert und liefern eine Spannungsdifferenz von 0,4V für einen Temperaturbereich von 0°C bis 100°C. Deswegen sind die Messwerte ebenso durch Differenzverstärker verstärkt, welche nachstehende Messergebnisse liefern. (siehe Abbildung 6.4) Die verschiedenen Offsets entstehen durch unterschiedliche Widerstandswerte in den Referenzspannungsteilern der Differenzverstärker, welche durch die Toleranzen der einzelnen Widerstände verursacht werden.



	0	10	20	30	40	50	60	70	80	90	100
V am Sensor 1	0,94	1,65	2,24	2,81	3,44	4	4,6	5,17	5,7	6,17	6,65
V am Sensor 2	0,58	0,68	1,28	1,86	2,51	3,06	3,68	4,25	4,8	5,27	5,75
V am Sensor 3	0,58	1,11	1,72	2,31	2,94	3,52	4,11	4,7	5,25	5,72	6,2

Abbildung 6.4: Temperaturmessung

7 Zusammenfassung und Ausblick

7.1 Zusammenfassung

Diese Arbeit beschreibt ein Konzept, das Sensordaten eines Fahrzeuges aufnimmt, aufzeichnet, per WLAN übermittelt und auf einem PC auswertet und visualisiert. Dieses Konzept wurde für das Fachbereich übergreifende Projekt "Eco-Team" der HAW-Hamburg entworfen. Es sollte dem Fahrzeug "Pingu II" verhelfen effizienter mit seinem Treibstoff "Wasserstoff" umzugehen, um eine bessere Plazierung bei dem Rennen in Nogaro, Frankreich zu erzielen. In den theoretischen Grundlagen wurden die verwendeten Technologien, wie Sensorik, Telemetrie sowie Übertragungstechnologien und GPS erläutert. Nachfolgend wurde beim Design der Telemetriebox auf das Konzept, die verwendete Software sowie Hardware, eingegangen. Genauer auf die Zentrale Sensoren Platine, das System und die Kommunikationsplatine. Die Realisierung zeigt, wie die Telemetriebox und deren Hard- und Software entwickelt wurde. Dabei wird besonders auf den Bau der einzelnen Komponenten der Telemetriebox eingegangen. Diese besteht aus den Komponenten, System und Telemetriesoftware, zentrale Sensorenplatine für die Verstärkung der Sensordaten, der Kommunikationsplatine für die Interaktion mit dem System und dem PMD, der die analogen- und digitalen Ein- und Ausgänge für die Software aufbereitet. Neben dem GPS und der Brennstoffzelle, wird auch auf die Sensoren, die im Auto verbaut wurden sowie auf deren Auswertung und Visualisierung eingegangen. Die Software in der Telemetriebox, die die Daten sammelt, speichert und versendet wird ebenso erläutert, wie die Telemetrieauswertesoftware, die die Daten entweder per WLAN empfängt oder aus einer Logdatei einliest, auswertet und visualisiert.

7.2 Ausblick

Im Laufe der Fertigstellung dieser Arbeit haben sich einige Verbesserungsmöglichkeiten ergeben. Dies liegt daran, dass inzwischen weiterentwickelte Hardware verfügbar ist und das Fahrzeug erst relativ spät während der Hardwareentwicklung fertig wurde. Die Zeit für Tests war deswegen sehr knapp bemessen. Verbesserungen an Hard- und Software mussten auf kleine Anpassungen beschränkt werden, die weder den Rennerfolg des Teams, noch die schon gegebene Funktionalität der Box

gefährden konnten. Zuallererst wäre dabei die Zentrale Sensoren Platine zu nennen, bei der sich durch bessere Widerstände noch genauere Messergebnisse erzielen lassen. Dennoch wird es bei analogen Sensoren immer notwendig bleiben eine Kennlinie zu erstellen. Der Temperatur Sensor könnte mit einem Stecker versehen und in den Motor hinein verlegt werden. Auch müsste eine neue verkleinerte und leichtere Box entworfen werden, die nur noch die Platinenmaße der in Zukunft verwendeten Hardware berücksichtigt. Dazu könnte es vorteilhaft sein, eine neue Sensorenplatine mit platzsparenden SMD Bauteilen zu entwickeln. Die auf einem Router basierende Hardware könnte gegen ein sehr viel leistungsfähigeres x86 basiertes Embedded System, wie das auf einem AMD Geode basierende alix2c2, ausgetauscht werden, welches genauso gut unterstützt wird, in Handhabung und Leistungsfähigkeit aber sehr viel besser abschneidet. Das GPS kann gegen die neue Antaris 5 Generation ersetzt werden, welche noch empfindlicher ist und einen schnelleren GPS Fix erreicht. Besonders zu erwähnen wäre dabei das u-blox Antaris 5 Evaluation Kit, welches unter anderem einen externen Antenneneingang bietet. Die Software kann so weiterentwickelt werden, dass sie mehrere PMDs und eventuell weitere zukünftig in das Auto zu integrierende Hardware unterstützt. Zuletzt bleibt noch die Auswertungssoftware zu erwähnen, die alle Daten des Fahrzeugs übersichtlich darstellen und auswerten muss.

8 Anhang A

8.1 Festlegung der Zielplattform

Filesystems/
kmod-fs-vfat
kmod-nls-cp437
kmod-nls-iso8859-1

USB Support/
kmod-usb-core
kmod-usb-hid
kmod-usb-ohci
kmod-usb-serial
kmod-usb-serial-cp2101
kmod-usb-serial-ftdi
kmod-usb-storage
kmod-usb-uhci
kmod-usb2

8.2 Paketauswahl

base-files-brcm47xx - 12-r11095 -
bridge - 1.0.6-1 -
busybox - 1.8.2-1 -
dnsmasq - 2.41-1 -
dropbear - 0.50-3 -
glib1 - 1.2.10-1 -
gpsd - 2.36-1 -
hotplug2 - 0.9+r102-2 -
iptables - 1.4.0-1 -
kernel - 2.6.23.16-brcm47xx-1 -
kmod-diag - 2.6.23.16-brcm47xx-4 -

kmod-fs-vfat - 2.6.23.16-brcm47xx-1 -
kmod-hid - 2.6.23.16-brcm47xx-1 -
kmod-input-core - 2.6.23.16-brcm47xx-1 -
kmod-input-evdev - 2.6.23.16-brcm47xx-1 -
kmod-ipt-nathelper - 2.6.23.16-brcm47xx-1 -
kmod-madwifi - 2.6.23.16+r3314-brcm47xx-1 -
kmod-nls-base - 2.6.23.16-brcm47xx-1 -
kmod-nls-cp437 - 2.6.23.16-brcm47xx-1 -
kmod-nls-iso8859-1 - 2.6.23.16-brcm47xx-1 -
kmod-ppp - 2.6.23.16-brcm47xx-1 -
kmod-pppoe - 2.6.23.16-brcm47xx-1 -
kmod-scsi-core - 2.6.23.16-brcm47xx-1 -
kmod-switch - 2.6.23.16-brcm47xx-1 -
kmod-usb-core - 2.6.23.16-brcm47xx-1 -
kmod-usb-hid - 2.6.23.16-brcm47xx-1 -
kmod-usb-ohci - 2.6.23.16-brcm47xx-1 -
kmod-usb-serial - 2.6.23.16-brcm47xx-1 -
kmod-usb-serial-cp2101 - 2.6.23.16-brcm47xx-1 -
kmod-usb-serial-ftdi - 2.6.23.16-brcm47xx-1 -
kmod-usb-storage - 2.6.23.16-brcm47xx-1 -
kmod-usb-uhci - 2.6.23.16-brcm47xx-1 -
kmod-usb2 - 2.6.23.16-brcm47xx-1 -
libgcc - 4.1.2-12 -
libncurses - 5.6-1 -
libpthread - 0.9.28-10 -
libreadline - 5.1-1 -
libuci - 0.3.4-1 -
mc - 4.6.1-2 -
mtd - 6 -
ntpd - 4.2.4-1 -
ppp - 2.4.3-10 -
ppp-mod-pppoe - 2.4.3-10 -
procps - 3.2.7-1 -
uci - 0.3.4-1 -
uclibc - 0.9.29-12 -
uclibcxx - 0.2.2-1 -
udevtrigger - 106-1 -
wireless-tools - 29-2 -

8.3 LAN und WAN - /etc/config/network

```
config 'switch' 'eth0'
option 'vLAN0' '1 2 3 4 5*'
option 'vLAN1' '0 5'

config 'interface' 'loopback'
option 'ifname' 'lo'
option 'proto' 'static'
option 'ipaddr' '127.0.0.1'
option 'netmask' '255.0.0.0'

config 'interface' 'LAN'
option 'type' 'bridge'
option 'ifname' 'eth0.0'
option 'proto' 'static'
option 'ipaddr' '192.168.10.254'
option 'netmask' '255.255.255.0'

config 'interface' 'wan'
option 'ifname' 'eth0.1'
option 'proto' 'dhcp'
```

8.4 LAN und WAN - /etc/firewall.user

```
#!/bin/sh
# Copyright (C) 2006 OpenWrt.org

iptables -F input_rule
iptables -F output_rule
iptables -F forwarding_rule
iptables -t nat -F prerouting_rule
iptables -t nat -F postrouting_rule

# The following chains are for traffic directed at the IP of the
# WAN interface

iptables -F input_wan
```

```
iptables -F forwarding_wan
iptables -t nat -F prerouting_wan

### Open port to WAN
## – This allows port 22 to be answered by (dropbear on) the router
iptables -t nat -A prerouting_wan -p tcp –dport 22 -j ACCEPT
iptables -A input_wan -p tcp –dport 22 -j ACCEPT
iptables -t nat -A prerouting_wan -p tcp –dport 2947 -j ACCEPT
iptables -A input_wan -p tcp –dport 2947 -j ACCEPT

### Port forwarding
## – This forwards port 8080 on the WAN to port 80 on 192.168.1.2
# iptables -t nat -A prerouting_wan -p tcp -dport 8080 -j DNAT -to 192.168.1.2:80
# iptables -A forwarding_wan -p tcp –dport 80 -d 192.168.1.2 -j ACCEPT

### DMZ
## – Connections to ports not handled above will be forwarded to 192.168.1.2
# iptables -t nat -A prerouting_wan -j DNAT –to 192.168.1.2
# iptables -A forwarding_wan -d 192.168.1.2 -j ACCEPT
```

8.5 LAN und WAN - /etc/config/wireless

```
config 'wifi-device' 'wifi0'
option 'type' 'atheros'
option 'country' 'de'
option 'channel' '11'
option 'agmode' '11b'
option 'diversity' '0'
option 'txantenna' '1'
option 'rxantenna' '1'
option 'disabled' '0'

config 'wifi-iface'
option 'device' 'wifi0'
option 'network' 'LAN'
option 'mode' 'adhoc'
option 'ssid' 'ECO-TEAM_HAW'
option 'encryption' 'wep'
```

option 'key' '1'
option 'key1' '65636f7465616d68617765636f' #ecoteamhaweco
option 'key2' '65636f7465616d68617765636f' #ecoteamhaweco
option 'key3' '65636f7465616d68617765636f' #ecoteamhaweco
option 'key4' '65636f7465616d68617765636f' #ecoteamhaweco

8.6 Paketarchive - /etc/ipkg.conf

src release http://downloads.OpenWrt.org/kamikaze/7.09/brcm47xx-2.6/packages
src packages http://downloads.OpenWrt.org/kamikaze/packages/mipsel
src packages2 http://downloads.x-wrt.org/xwrt/kamikaze/snapshots/brcm47xx-2.6/packages
dest root /
dest ram /tmp

8.7 Systemdienste - /etc/init.d/

```
#!/bin/sh /etc/rc.common
# Copyright (C) 2006 OpenWrt.org

START=97
start() {
/jffs/binaries/udpsmips
}

stop() {
killall udpsmips
}
```

9 Anhang B

9.1 Kurzbeschreibung der Telemetrieauswerte Software für das Eco-Team HAW

- Es gibt zwei Versionen der Software. Eine, um die Telemetrie des Fahrzeugs während der Fahrt über das WLAN auszulesen und die Daten direkt angezeigt zu bekommen und eine um die Logdatei, die die Telemetriebox im Fahrzeug aufzeichnet, einzulesen und die Daten nach der Fahrt zu analysieren.
  - Die Software für das WLAN
  - Die Software für die Logdatei

9.2 Software für das WLAN

Es gibt 8 Tabs auf der Oberfläche, die wegen der Übersicht und des Platzmangels aufgeteilt wurden.

- Brennstoffzelle
- GPS
- PMD
- PMD Verlaufsdiagramm
- PMD Leistung
- Vergleich PMD zu Brennstoffzelle
- Netzwerk
- Streckenführung

9.2.1 Brennstoffzelle

Im "Tab Brennstoffzelle" werden die aktuellen Werte der Brennstoffzelle angezeigt. Zudem sind zwei Verlaufsdiagramme zur besseren Übersicht eingefügt. Das eine zeigt die Leistung im Verhältnis der Zeit und das andere die Stack Temperatur, Stack Spannung, Stack Strom, Wasserstoffdruck und die Umgebungstemperatur im Vergleich. Bei diesem Verlaufsdiagramm können die einzelnen Werte ein- und ausgeschaltet werden, um eine bessere Übersicht zu bekommen. (siehe Abbildung 9.1)

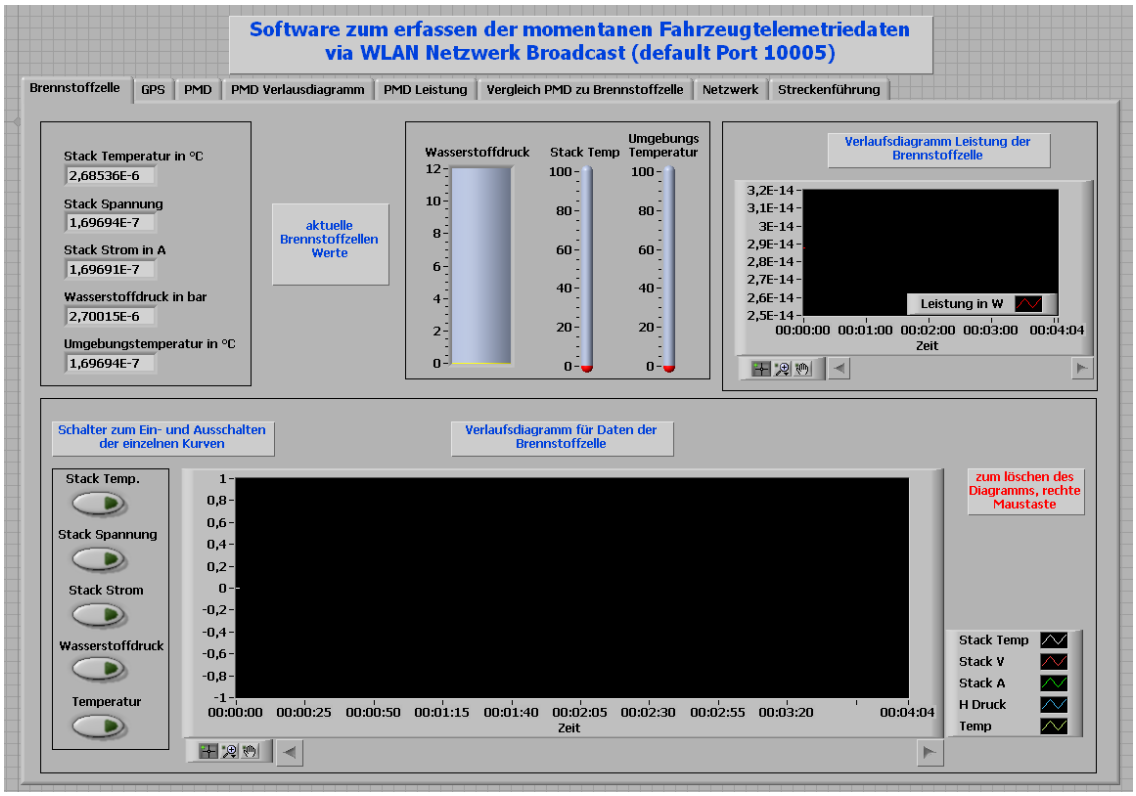


Abbildung 9.1: TAP Brennstoffzellen

9.2.2 GPS

Im "Tab GPS" werden die aktuellen Werte des GPS angezeigt. Hier ist ein Verlaufsdiagramm eingebunden, das die Höhe und die Geschwindigkeit in km/h im Verhältnis der Zeit anzeigt. Zudem gibt es eine Richtungsanzeige und die Information über den FIX des GPS (keinen FIX, 2D FIX oder 3D FIX) Der GPS Mode zeigt den momentanen NMEA Mode an. No Fix - keine Ortsbestimmung möglich 2D Fix - keine Höhenangaben möglich (ungenau) 3D Fix - Guter Empfang (min. 3 Satelliten) (siehe Abbildung 9.2)

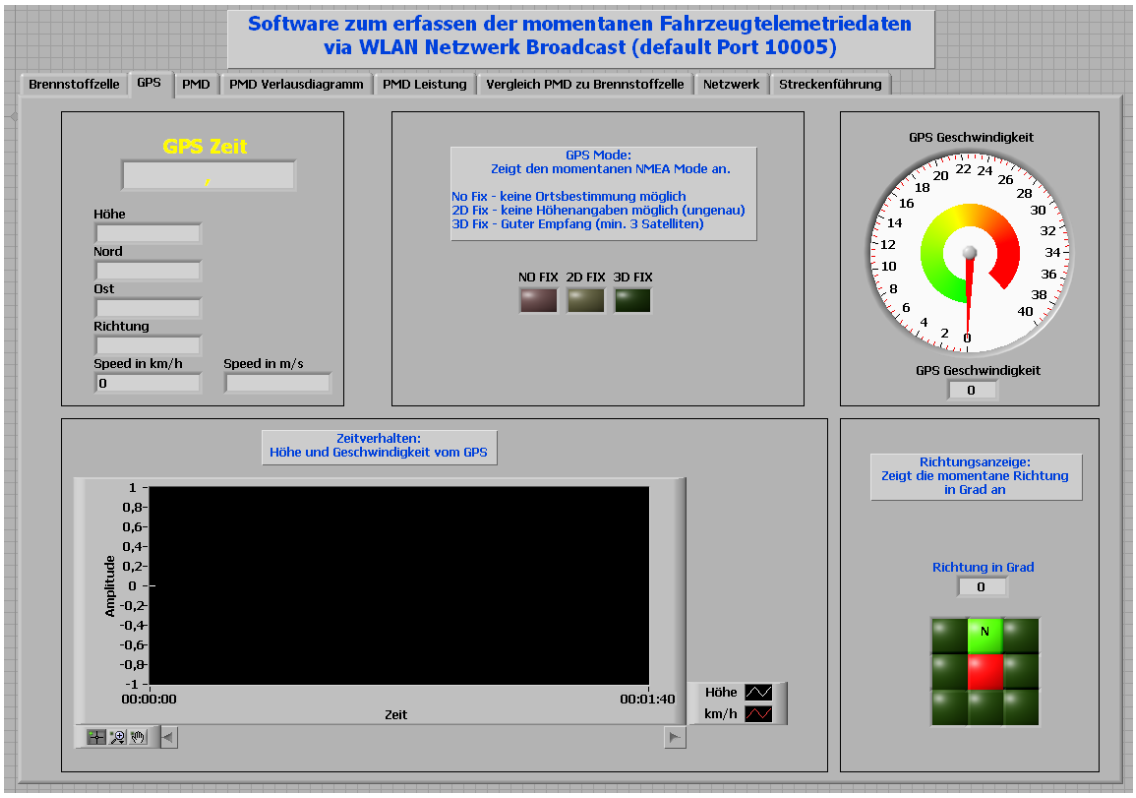


Abbildung 9.2: TAP GPS

9.2.3 PMD

Im "Tab PMD" werden die aktuellen Werte des PMD angezeigt. Spannungen, Strom, Temperaturen sowie Geschwindigkeit und Leistung. Zudem wird über eine LED signalisiert, ob die Telemetriebox die Telemetriedaten aufzeichnet. (siehe Abbildung 9.3)

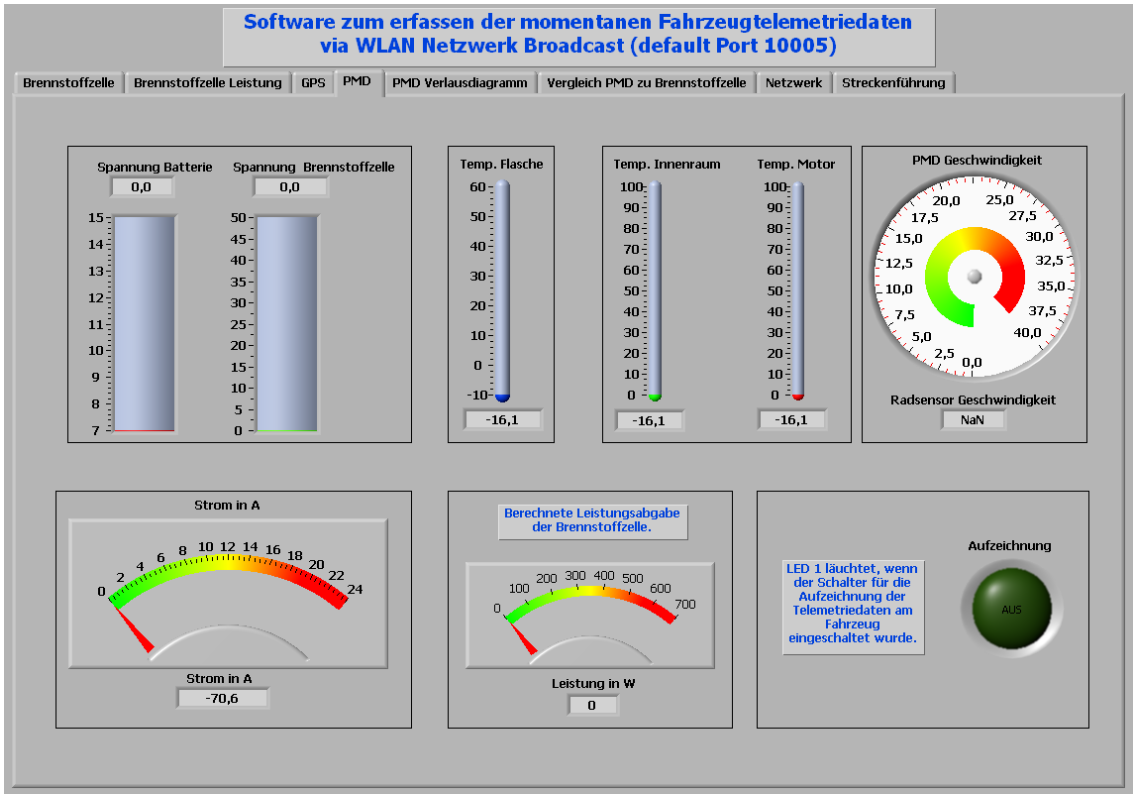


Abbildung 9.3: TAP PMD

9.2.4 PMD Verlaufsdiagramm

Im "Tab PMD Verlaufsdiagramm" werden die Werte in zwei Verlaufdiagrammen dargestellt. (siehe Abbildung 9.4)

Abbildung 9.4: TAP PMD Verlaufsdiagramm

9.2.5 PMD Leistung

Im "Tab PMD Leistung" wird die berechnete Leistung, aus der gemessenen Spannung und dem Strom, im Verlaufdiagrammen dargestellt. (siehe Abbildung 9.5)

Abbildung 9.5: TAP PMD Leistung

9.2.6 Vergleich PMD zu Brennstoffzelle

Im "Tab Vergleich PMD zu Brennstoffzelle" werden die aktuellen Werte des PMD und der Brennstoffzelle zueinander in Vergleich gesetzt. (siehe Abbildung 9.6)

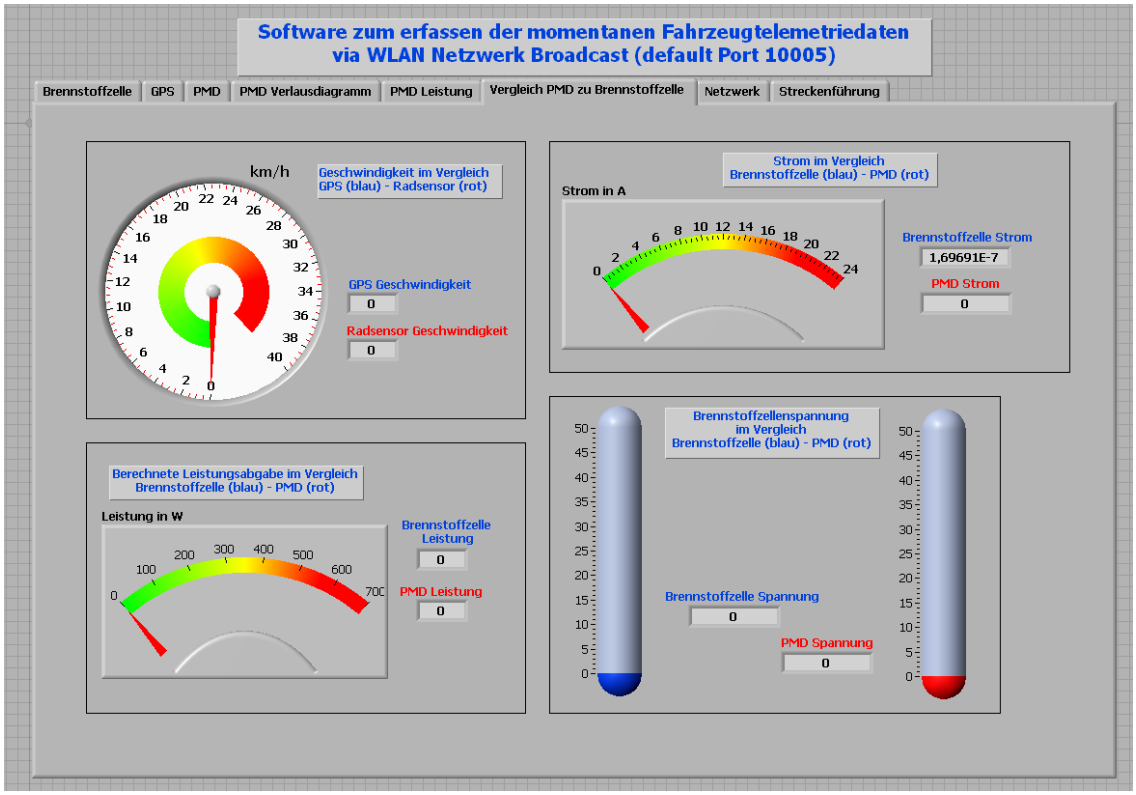


Abbildung 9.6: TAP Vergleich PMD zu Brennstoffzelle

9.2.7 Netzwerk

Im "Tab Netzwerk" werden die aktuellen Werte des Netzwerks angezeigt. Hier lässt sich auch der Port der Netzwerkverbindung verändern, der Defaultwert ist Port 10005. Zudem werden die Daten angezeigt, die über die Verbindung übertragen werden. (siehe Abbildung 9.7)

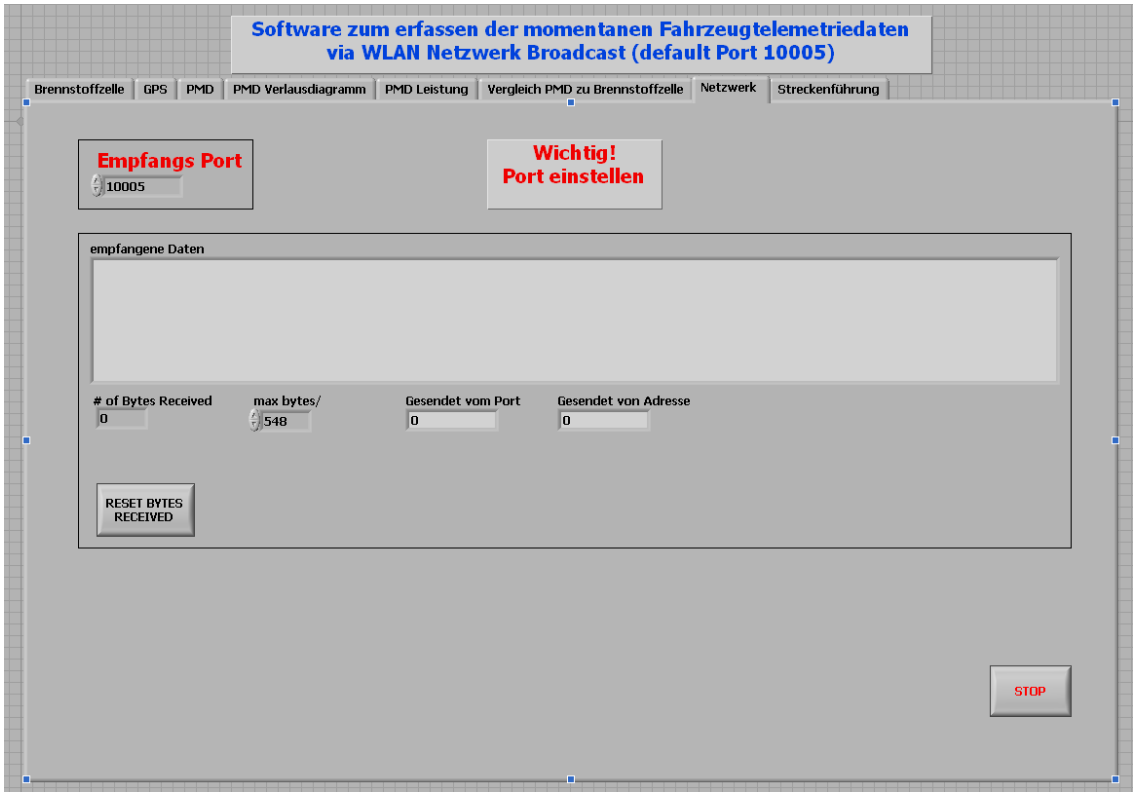


Abbildung 9.7: TAP Netzwerk

9.2.8 Streckenführung

Im "Tab Streckenführung" werden die aktuellen Werte des GPS in einem 3D-Graph angezeigt. Die Streckenführung wird in den 3 Achsen, Höhe, Nördliche und Östliche Breiten und Längengerade angezeigt. (siehe Abbildung 9.8)

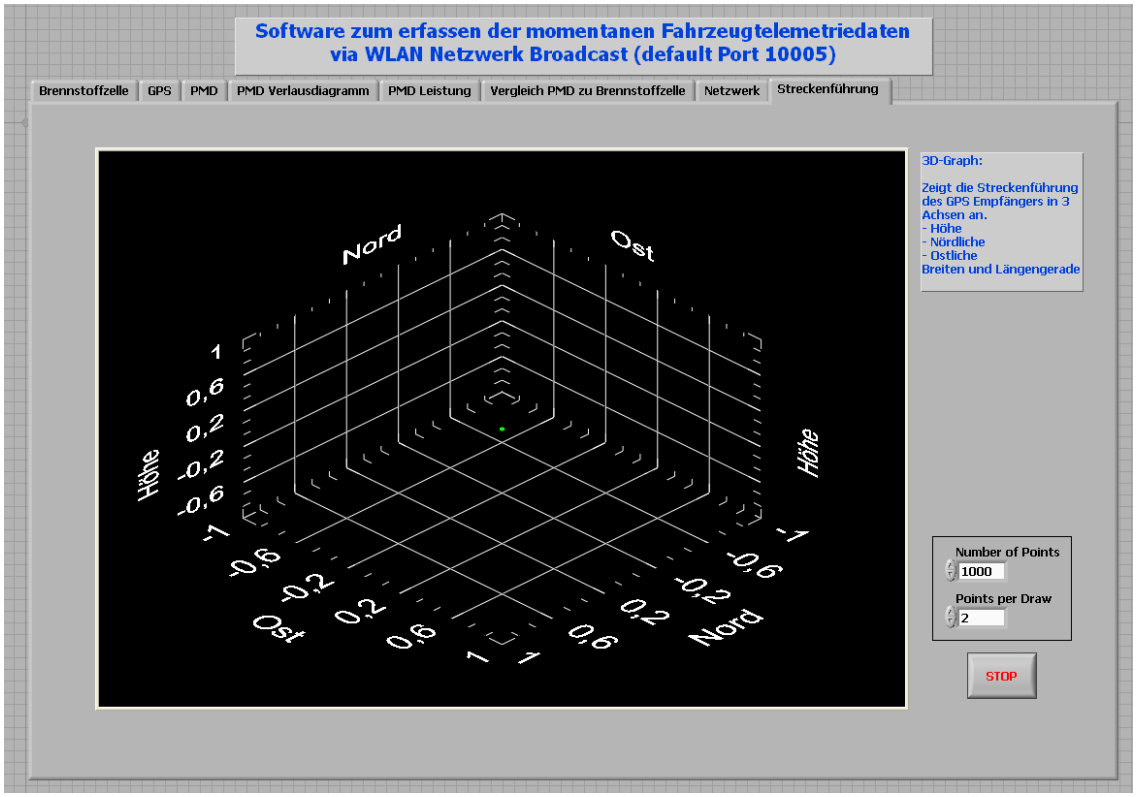


Abbildung 9.8: TAP Streckenführung

9.3 Die Software für die Logdatei

Es gibt 6 Tabs auf der Oberfläche, die wegen der Übersicht und des Platzmangels aufgeteilt wurden.

- Brennstoffzelle
- Verlauf Leistung
- GPS
- PMD
- PMD Verlauf
- GPS Verlauf

9.3.1 Brennstoffzelle

Im "Tab Brennstoffzelle" werden die Werte der Brennstoffzelle, die in der Logdatei stehen, in einem Verlausdiagramm angezeigt. (siehe Abbildung 9.9)

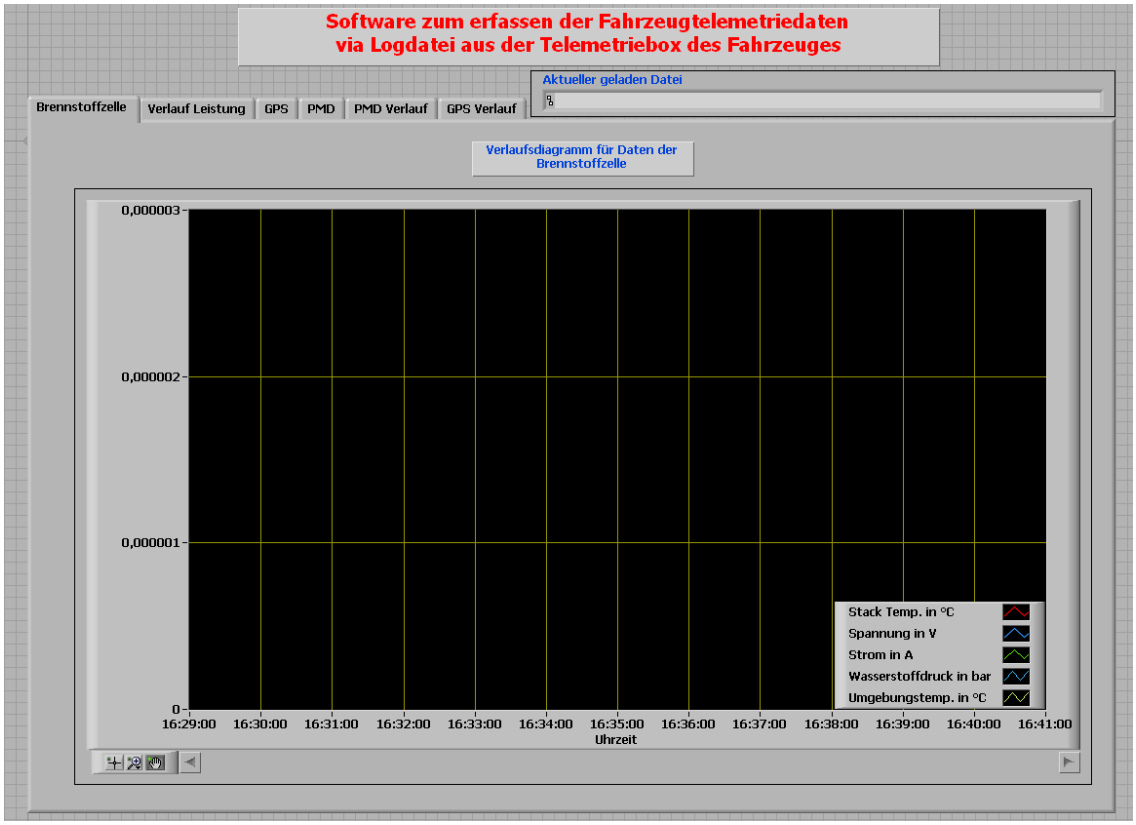


Abbildung 9.9: TAP Brennstoffzelle

9.3.2 Verlauf Leistung

Im "Tab Verlauf Leistung" werden die errechneten Werte der Brennstoffzelle, die in der Logdatei stehen, in einem Verlausdiagramm angezeigt. (siehe Abbildung 9.10)

Abbildung 9.10: TAP Verlauf Leistung

9.3.3 GPS

Im "Tab GPS" werden die Werte der GPS, die in der Logdatei stehen, in einem Verlausdiagramm angezeigt. (siehe Abbildung 9.11)

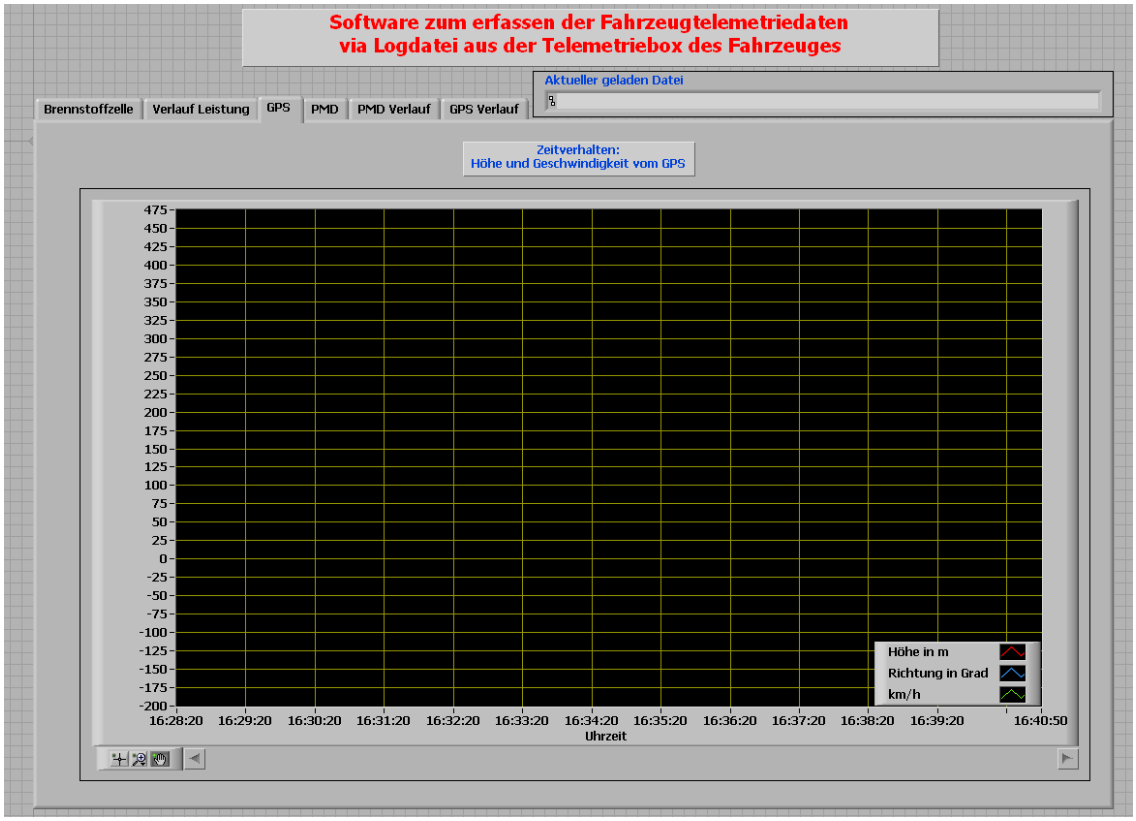


Abbildung 9.11: TAP GPS

9.3.4 PMD

Im "Tab PMD" werden die Werte der PMDs, die in der Logdatei stehen, in zwei Verlausdiagrammen angezeigt. (siehe Abbildung 9.12)

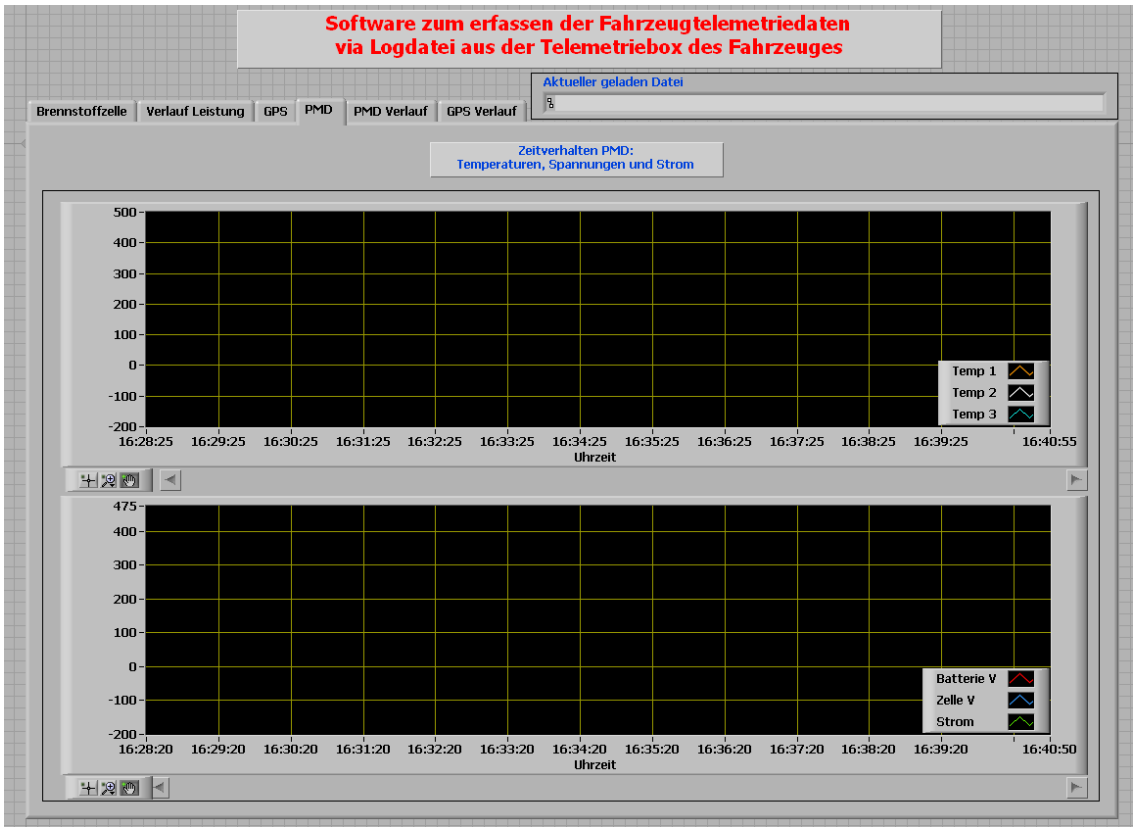


Abbildung 9.12: TAP PMD

9.3.5 PMD Verlauf

Im "Tab PMD Verlauf" werden die Werte der PMDs, die in der Logdatei stehen angezeigt. Hier ist es möglich die aktuellen Daten, die gerade aus der Datei ausgelesen werden, angezeigt zu bekommen. Dieses erreicht man, indem man die Geschwindigkeit über einen Schieberegler einstellt, mit der die Simulation ablaufen soll. (siehe Abbildung 9.13)

Abbildung 9.13: TAP PMD Verlauf

9.3.6 GPS Verlauf

Im "Tab GPS Verlauf" werden die Werte der GPS, die in der Logdatei stehen angezeigt. Hier ist es möglich die aktuellen Daten, die gerade aus der Datei ausgelesen werden, angezeigt zu bekommen. Dieses erreicht man, indem man die Geschwindigkeit über einen Schieberegler einstellt, mit der die Simulation ablaufen soll. (siehe Abbildung 9.14)

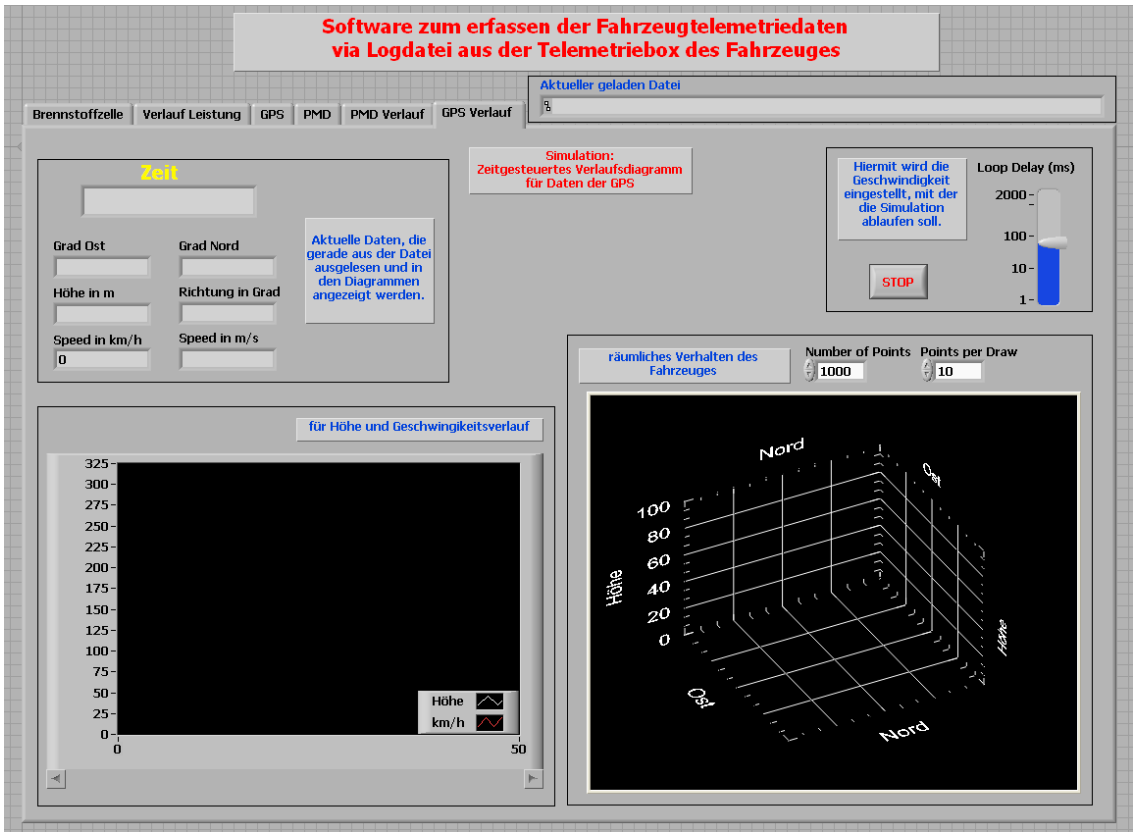


Abbildung 9.14: TAP GPS Verlauf

Versicherung über Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Arbeit im Sinne der Prüfungsordnung nach §24(5) ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, 29. Mai 2008	
Ort, Datum	Unterschrift

- Von Herrn Felix Kneisler bearbeitet:
  - In Kapitel 3 - Theoretische Grundlagen
    3.1.1, 3.1.7
  - In Kapitel 4 - Design der Telemetriebox
    4.1.2, 4.1.3, 4.1.6, 4.1.7, 4.1.8, 4.2.2, 4.2.4, 4.2.5, 4.2.9
  - In Kapitel 5 - Realisierung
    5.2, 5.8, 5.9
  - und Anhang B

Versicherung über Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Arbeit im Sinne der Prüfungsordnung nach §24(5) ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, 29. Mai 2008	
Ort, Datum	Unterschrift

- Von Herrn Nico Dummer bearbeitet:
  - In Kapitel 3 - Theoretische Grundlagen
    3.1.2, 3.1.3, 3.1.4, 3.1.5, 3.1.6
  - In Kapitel 4 - Design der Telemetriebox
    4.1.1, 4.1.4, 4.1.5, 4.1.9, 4.1.10, 4.2.1, 4.2.3, 4.2.6, 4.2.7, 4.2.8
  - In Kapitel 5 - Realisierung
    5.3, 5.4, 5.5, 5.6, 5.7
  - und Anhang A

Literaturverzeichnis

[5] Titel: Halbleiter-Schaltungstechnik / U. Tietze; Ch. Schenk; Verfasser: Tietze, Ulrich *1946-* ; Schenk, Christoph Ausgabe: 12. Aufl. / unter Mitarb. von E. Gamm; Erschienen: Berlin [u.a.] : Springer, 2002; ISBN: 3-540-42849-6

[2] Linux-Unix-Programmierung von Jürgen Wolf (Autor) - Verlag: Galileo Press; Auflage: 2., aktualis. und erw. A. (Dezember 2005) ISBN-10: 3898427498 http://www.pronix.de/pronix-6.html 28. Mai 2008

[3] C von A bis Z von Jürgen Wolf (Autor) - Verlag: Galileo Press; Auflage: 2., akt. A. (Januar 2006) - ISBN-10: 3898426432

[5] Titel: Einführung in LabVIEW : mit 133 Aufgaben / Wolfgang Georgi; Ergun Metin Ausgabe: 3. Erschienen: München : Fachbuchverl. Leipzig im Carl-Hanser- Verl., 2007 ISBN: 3-446-41109-7

[5] Titel: LabVIEW : das Grundlagenbuch ; [bis Version LabVIEW 7.1] / Rahman Jamal; Andre Hagestedt Ausgabe: 4. Aufl. Erschienen: München [u.a.] : Addison-Wesley, 2004, ISBN: 3-8273-2051-8*Gb.

[6] http://www.eco-marathon.de/ 27. Mai 2008

[7] Diplomarbeit von Bachir Blal - Design und Implementation eines Konzeptes zur Geschwindigkeitsbestimmung eines autonomen Fahrzeugs unter Verwendung eines AVRMikrokontrollers in Kombination mit Hall-Sensorik, Hochschule für Angewandte Wissenschaften Hamburg vom 10. August 2007

[8] http://www.hydrogeit.de/brennstoffzelle.htm
http://www.agenda21-treffpunkt.de/lexikon/Brennstoffzelle.htm 27. Mai 2008

[9] http://www.usb-infos.de/ http://www.usb.org/developers/ 27. Mai 2008

[10] http://www.faqs.org/rfcs/rfc793.html 27. Mai 2008

[11] http://www.faqs.org/rfcs/rfc768.html 27. Mai 2008

[12] http://www.abcdata.de/wlan_Standards_einleitung.htm 27. Mai 2008

[13] http://www.kowoma.de/gps/Geschichte.htm 27. Mai 2008

[14] http://www.cadsoft.de/ 27. Mai 2008

[15] http://openwrt.org/ 27. Mai 2008

[16] http://www.ni.com/labview/d/ 27. Mai 2008

[17] http://gpsd.berlios.de/ 27. Mai 2008

[18] http://www.u-blox.de/products/u_center.html 27. Mai 2008

[19] http://www.measurementcomputing.com/cbicatalog/directory.asp?dept_id=403 27. Mai 2008

[20] http://www.sminstall.com/ 27. Mai 2008

[21] http://www.ubuntu.com/ 27. Mai 2008

[22] http://www.paglo.com/opensource/packetyzer_thankyou 27. Mai 2008

[23] http://www.atheros.com/pt/archivefiles_updated041707/AR5004XBulletin.htm 27. Mai 2008

[24] http://www.produktinfo.conrad.com/datenblaetter/150000-174999/155589-da-01-en-OP-VERST_LM324AN_STM.pdf 27. Mai 2008

[25] http://www.ortodoxism.ro/datasheets/texasinstruments/uln2003a.pdf 27. Mai 2008

[26] http://www.produktinfo.conrad.com/datenblaetter/150000-174999/154596-da-01-en-DC_DC_WAND_15W_9_36V_5V_AEE03A18.pdf 27. Mai 2008

[27] http://wiki.openwrt.org/TableOfHardware

[28] http://www.heliocentris.com/de/fe_loesungen/12_kw_nexa.html 11. Mai 2008

[29] http://www.datasheetcatalog.com/datasheets_pdf/L/M/3/2/LM324AN.shtml 27. Mai 2008

[30] http://www.mercateo.com/p/108EL-149(2d)736/DC_DC_Wandler_AEE03A18_L_Herstellerartikelnr_AEE03A18_L.html 27. Mai 2008

[31] http://tweakers.net/ext/i/productsurvey/5722/4189.jpg 23. Mai 2008

[32] http://www.timo-holle.de/artikel/wlan-router/local/wl500gpMB.jpg
23. MAi 2008

[33] http://www.jor.se/res/Jorbilder/pmd1208lssmall.png 27. Mai 2008

[34] http://www.measurementcomputing.com/PDFManuals/usb-1208fs.pdf
28. Mai 2008

[35] http://www.tonerklau.de/images/product_images/original_images/837_0.jpg
27. Mai 2008

[36] bachir_blal_-_konzept_geschwindigkeitsmessung_mit_hall-sensorik-1.pdf
27. Mai 2008

[37] http://img.directindustry.de/images_di/photo-p/306124.jpg 27. Mai 2008

[38] http://images.conrad.de/m/1000_1999/1700/1710/1717/
171778_BB_01_FB.EPS.jpg 27. Mai 2008

[39] Nexa User's Manual/MAN5100078.pdf 27. Mai 2008

[40] http://downloads.openwrt.org/kamikaze/docs/openwrt.html 27. Mai 2008
http://forum.openwrt.org/viewtopic.php?id=9180 27. Mai 2008

[41] http://wiki.openwrt.org/OpenWrtDocs/Hardware/Asus/WL500GP
27. Mai 2008

[42] http://gpsd.berlios.de/gpsd.html 27. Mai 2008

[43] ftp://lx10.tx.ncsu.edu/pub/Linux/drivers/PMD/usbhid/PMD.1.9.tgz
27. Mai 2008

[44] http://www.pronix.de/pronix-300.html 28. Mai 2008

[45] http://www.pronix.de/pronix-303.html 28. Mai 2008

Abbildungsverzeichnis