**Henning Groß, Beuth Hochschule für Technik Berlin. Studiengang Technische Informatik Bachelor, 749992**

WS2012

**Bachelorarbeit**

**Einrichtung zur Erfassung und Visualisierung von Drehzahl, Geschwindigkeit und Abgastemperatur klassischer Motorroller**

**Henning Groß**

Inhaltsverzeichnis

[1. Selbstständigkeitserklärung 3](#_Toc348696651)

[2. Copyright 4](#_Toc348696652)

[3. Einleitung 5](#_Toc348696653)

[4. Aufgabenstellung, fachliche Anforderungen/ Rahmen- und Umgebungsbedingungen 7](#_Toc348696654)

[4.1. Messeinrichtung 7](#_Toc348696655)

[4.1.1. Wahl einer Plattform zur Erfassung der Messdaten 7](#_Toc348696656)

[4.1.2. Entwicklung der Messschaltungen, Erfassen der Signale/Messdaten 7](#_Toc348696657)

[4.1.2.1. Drehzahl 7](#_Toc348696658)

[4.1.2.2. Weg/Zeit (Tacho) 7](#_Toc348696659)

[4.1.2.3. Abgastemperatur 8](#_Toc348696660)

[4.2. Auswertung und Darstellung 8](#_Toc348696661)

[4.2.1. Wahl einer Plattform zur Auswertung und Darstellung der Messdaten 9](#_Toc348696662)

[4.2.2. Entwicklung der Algorithmen zur Verarbeitung der Messdaten 9](#_Toc348696663)

[4.2.3. Visualisierung der Messdaten 9](#_Toc348696664)

[4.3. Realisierung der Kommunikation zwischen Mess- und Auswertungseinrichtung 9](#_Toc348696665)

[4.3.1. Wahl eines Übertragungsverfahrens 9](#_Toc348696666)

[4.3.2. Entwicklung eines Datenformates und Protokolls 9](#_Toc348696667)

[5. Lösungsansätze 10](#_Toc348696668)

[5.1. Messeinrichtung 10](#_Toc348696669)

[5.1.1. Drehzahl 10](#_Toc348696670)

[5.1.1.1. Hall-Effekt 10](#_Toc348696671)

[5.1.1.2. Induktion am Zündkabel 12](#_Toc348696672)

[5.1.1.3. Lichtmaschine 13](#_Toc348696673)

[5.1.2. Tachosignal 15](#_Toc348696674)

[5.1.2.1. Reedkontakt 15](#_Toc348696675)

[5.1.2.2. Rechnerisch 16](#_Toc348696676)

[5.1.2.3. Hall-Effekt 16](#_Toc348696677)

[5.1.3. Abgastemperatur 17](#_Toc348696678)

[5.1.3.1. Umsetzung mit Analogschaltung 18](#_Toc348696679)

[5.1.3.2. IC 18](#_Toc348696680)

[5.2. Signalverarbeitung 20](#_Toc348696681)

[5.2.1. Drehzahl / Tacho 20](#_Toc348696682)

[5.2.1.1. Beispiele 20](#_Toc348696683)

[5.2.2. Abgastemperatur 21](#_Toc348696684)

[5.3. Datenübertragung 22](#_Toc348696685)

[5.3.1. Seriell über Kabel 23](#_Toc348696686)

[5.3.2. Can-Bus 24](#_Toc348696687)

[5.3.3. Bluetooth 24](#_Toc348696688)

[5.4. Auswertung und Darstellung 25](#_Toc348696689)

[5.4.1. Mikrocontroller mit LCD-Display 25](#_Toc348696690)

[5.4.2. Android 29](#_Toc348696691)

[6. Anforderungsdefinition, Systementwurf und Realisierung 30](#_Toc348696692)

[6.1. Systemübersicht und Modulbeschreibungen 30](#_Toc348696693)

[6.2. Datenpaket 31](#_Toc348696694)

[6.3. Messsystem 31](#_Toc348696695)

[6.3.1. Plattform 31](#_Toc348696696)

[6.3.2. Schaltpläne 34](#_Toc348696697)

[6.3.2.1. Gesamtschaltplan und Layout Messschaltung 34](#_Toc348696698)

[6.3.2.2. Drehzahlerfassung 36](#_Toc348696699)

[6.3.2.3. Auswertung Thermoelement (Abgastemperatur) 37](#_Toc348696700)

[6.3.2.4. Erfassung Tachosignal 37](#_Toc348696701)

[6.3.3. Detaillösungen Software Messsystem 38](#_Toc348696702)

[6.3.3.1. Schiebealgorithmus EGT/Max6675 38](#_Toc348696703)

[6.3.3.2. Paketserialisierung 39](#_Toc348696704)

[6.4. Einrichtung zur Auswertung und Visualisierung 39](#_Toc348696705)

[6.4.1. Klassendiagramme 39](#_Toc348696706)

[Paket de.gaffa.vescom.vesdroid 40](#_Toc348696707)

[6.4.1.1. Paket de.gaffa.vescom.vesdroid.bluetoothmanagement 41](#_Toc348696708)

[6.4.1.2. Paket de.gaffa.vescom.vesdroid.packet 42](#_Toc348696709)

[6.4.1.3. Android SDK: BluetoothChat 43](#_Toc348696710)

[6.5. Paketserialisierung/Deserialisierung 44](#_Toc348696711)

[6.5.1. Serialisierung in der Messeinheit 44](#_Toc348696712)

[6.5.2. Deserialisierung in der Visualisierungseinheit 45](#_Toc348696713)

[7. Zusammenfassung und Ausblick 47](#_Toc348696714)

[8. Abbildungsverzeichnis 48](#_Toc348696715)

[9. Quellen- und Literaturverzeichnis 49](#_Toc348696716)

# Selbstständigkeitserklärung

Henning Groß, 749992, Studiengang: Technische Informatik BA, Fachbereich VI

**Erklärung zur Abschlussarbeit**

Ich versichere, dass ich meine Abschlussarbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Berlin,

Unterschrift

# Copyright

Die vorliegende Arbeit steht unter Creative Commons-Lizenz (Attribution-Non-Commercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0)). Sie dürfen sie kopieren, veröffentlichen, übertragen und auch in Auszügen nutzen, so lang der ursprüngliche Autor und die Arbeit im Ergebnis kenntlich gemacht und in keiner Weise der Eindruck erweckt wird, es handele sich um das Ergebnis Ihrer Arbeit. Sie dürfen den Inhalt der Arbeit nur für nichtkommerzielle Zwecke verwenden. Wenn Sie diese Arbeit – auch in Auszügen – nutzen, erweitern oder verändern, so muss das Ergebnis unter der gleichen Lizenz veröffentlicht werden. In keinem Fall darf die Nutzung erweitert oder eingeschränkt werden.

Der dieser Arbeit beigefügte Quelltext steht zum Teil unter General Public License (GPL, <http://www.gnu.org/copyleft/gpl.html>, deutsche Fassung: <http://www.gnu.de/documents/gpl-3.0.de.html>) und zum Teil unter Apache License (<http://www.apache.org/licenses/LICENSE-2.0>). Quellcode ist in den Dateien mit Lizenzhinweisen versehen. Solche Dateien, die keine Lizenzhinweise enthalten, stehen unter GPL. Nutzung der Quellen, auch in Auszügen, ist nur unter Beachtung der Lizenzbedingungen gestattet.

# Einleitung

Klassische Motorroller der Firma Piaggio (Vespa) erfreuen sich unter anderem aufgrund der möglichen Leistungsoptimierungen großer Beliebtheit (vgl. European Scooter Challenge [7], German Scooter Forum [8]). Wettbewerbsfahrzeuge des Typs V50N werden im Rennsport von 1,7PS auf bis zu über 40PS Motorleistung (Hinterrad) bei Drehzahlen von bis zu 15.000 Umdrehungen/Minute optimiert.

Die verwendeten Auspuffanlagen der Zweitaktmotoren führen durch Gegendruck in einem bestimmten Resonanzbereich (Drehzahl) zu erheblichem Leistungszuwachs. Außerhalb des Resonanzbandes liegt deutlich weniger Leistung an. Siehe Diagramm ab ca. 6250 Umdrehungen/Minute.

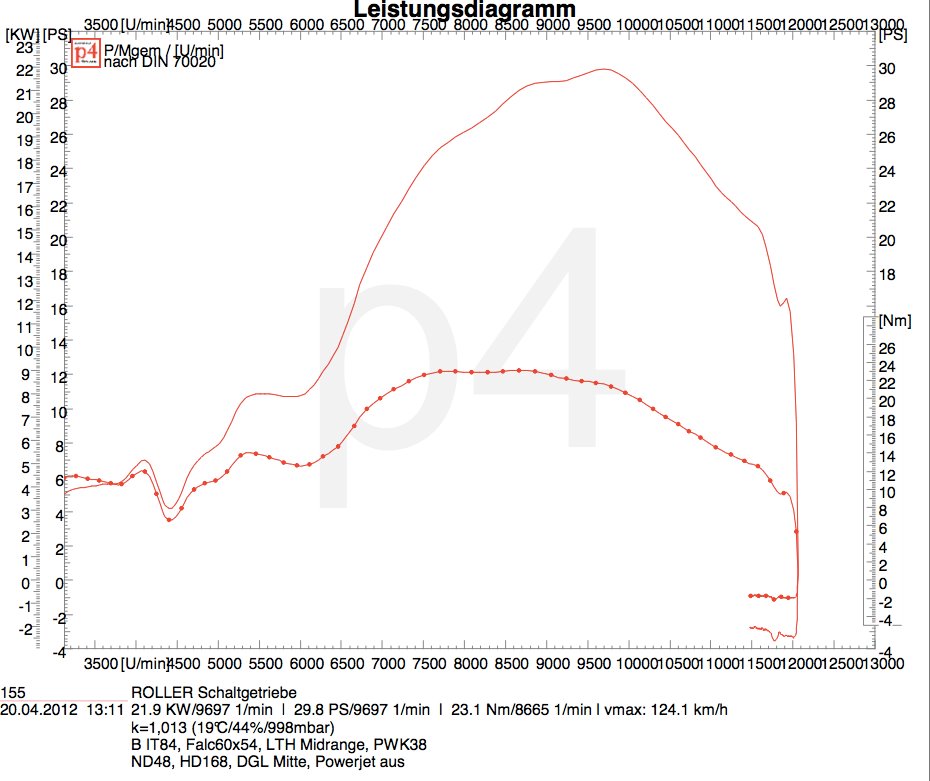


Abbildung ‑ - Leistungsdiagramm des Testfahrzeugs dieser Arbeit

Im Diagramm ist zu erkennen, dass vor Resonanzdrehzahl bei etwa 6250 Umdrehungen/Minute maximal 12 PS anliegen. Innerhalb eines Anstiegs der Drehzahl um ca. 750 Umdrehungen/Minute verdoppelt sich die Leistung auf 24 PS und steigt bis 29,8 PS.

Diese Charakteristik bedingt hohe Anforderungen an den Fahrer bei Schalt- und Startvorgängen. Diese können durch die Anpassung der Drehzahl effektiv optimiert werden. Die Optimierung besteht darin, bei Startvorgängen den Motor beim Einkuppeln auf einer Drehzahl zu halten, bei der der Motor ein möglichst hohes Drehmoment abgibt. Bei Schaltvorgängen muss unter Berücksichtigung der Getriebeübersetzung ein Zeitpunkt gewählt werden, bei dem im nächsten Gang eine Motordrehzahl erreicht wird zu der möglichst viel Leistung abgegeben wird sodass das Leistungsband optimal ausgenutzt wird.

Das Vornehmen solcher Anpassungen ist schwierig, wird aber durch eine übersichtliche Visualisierung der Motordaten beherrschbar.

Die Abgastemperatur lässt Rückschlüsse auf das Verbrennungsluftverhältnis zu. Niedrige Abgastemperatur deutet darauf hin, dass die Gemischeinstellung zu fett ist. Dies führt neben höherem Verbrauch zu direkten und indirekten Leistungseinbußen. So verrußt das Kolbendach und die Abgasanlage und Zündkerzen werden unbrauchbar. Eine hohe Temperatur deutet auf zu magere Vergasereinstellung, ungünstige Wahl des Zündzeitpunktes oder Undichtigkeiten des Systems (z.B. Zylinderkopf, Ansaugtrakt, Kurbelwellendichtring) hin und ist ein wichtiges Frühwarnzeichen, bevor es zu thermischen Schäden wie dem Verklemmen des Kolbens oder Durchbrennen des Kolbendachs kommt. Die bei Viertaktmotoren übliche Ermittlung des Verbrennungsluftverhältnisses mit Hilfe von Lambda-Sonden ist beim Zweitaktmotor nicht möglich, da die Sonden vom Öl im Abgasstrom mechanisch beschädigt werden. Somit ist die Überwachung der Abgastemperatur das effektivste Mittel zum Abstimmen und zur Überwachung von leistungsstarken Zweitaktmotoren.

Aufgrund des Alters der Fahrzeuge und der Anforderungen des ursprünglichen Leistungsniveaus, werden diese Fahrzeugdaten allerdings nicht erfasst und dargestellt. Bei Importmodellen fehlt häufig sogar eine Einrichtung zur Erfassung der Geschwindigkeit. Auch existieren auf dem Markt keine Geräte, welche die drei relevanten Informationen Drehzahl, Abgastemperatur und Geschwindigkeit übersichtlich und kombiniert darstellen. In Folge sind an Rennfahrzeugen häufig mehrere Messeinrichtungen untergebracht, was die Übersicht des Fahrers einschränkt.

Die zentrale Darstellung überwachter Messwerte bringt den Fahrer in die Lage, sein Fahrzeug ideal einzustellen und zu betreiben. Die Entwicklung einer solchen Einrichtung ist Ziel dieser Arbeit.

Die Erfassung der Daten soll mit einem Mikrocontroller mit Hilfe eines im Verlauf der Arbeit zu entwickelnden Softwareprogramms und entsprechender Messschaltungen erfolgen. Zur Erfassung der Drehzahl, des Tachosignals und der Abgastemperatur sind geeignete Sensorschaltungen zu entwickeln, deren Ausgangssignale vom Mikrocontroller erfasst und verarbeitet werden.

Die erfassten Daten sollen ausgewertet und dargestellt werden.

# Aufgabenstellung, fachliche Anforderungen/ Rahmen- und Umgebungsbedingungen

## Messeinrichtung

Zur Erfassung der Fahrzeugdaten Drehzahl, Abgastemperatur und Geschwindigkeit ist eine Messeinrichtung zu entwickeln, welche geeignete Sensorschaltungen und eine Einrichtung zur Erfassung der Sensordaten, sowie deren Übermittlung zu einer Visualisierungseinrichtung beinhaltet.

### Wahl einer Plattform zur Erfassung der Messdaten

Die Erfassung der Messdaten erfordert eine Echtzeitumgebung da das Zeitverhalten bei der Ermittlung von Taktdifferenzen kritisch ist. Die erfassten und gewandelten physikalischen Größen sollen digitalisiert und übertragen werden können. Die Plattform muss weiter über geeignete Eingänge zur Erfassung der Sensordaten verfügen.

### Entwicklung der Messschaltungen, Erfassen der Signale/Messdaten

#### Drehzahl

Die Motordrehzahl muss über einen geeigneten Sensor erfasst und in ein zu der Messeinrichtung kompatibles Signal gewandelt werden. Die Verarbeitung des Drehzahlsignales in der Messeinrichtung ist zu gestalten.

##### Erwarteter Wertebereich

Die minimale Standgasdrehzahl des Motors (der Kurbelwelle) beträgt ca. 800 Umdrehungen/Minute. Drehzahlen bis 15000 Umdrehungen/Minute sind zu erwarten.

##### Aktualisierungsintervall

Die Drehzahlanzeige soll mindestens 2 Mal pro Sekunde aktualisiert werden.

#### Weg/Zeit (Tacho)

Der zurückgelegte Weg muss über einen geeigneten Sensor erfasst und in ein zu der Messeinrichtung kompatibles Signal gewandelt werden. Die Verarbeitung des Tachosignals in der Messeinrichtung ist zu gestalten.

##### Erwarteter Wertebereich

Die Drehrichtung der Räder soll zur Ermittlung der Geschwindigkeit keine Rolle spielen. Der Minimalwert entspricht daher 0 (Stillstand). Die maximal erfassbare Geschwindigkeit soll 200km/h betragen.

##### Aktualisierungsintervall

Der Reifenumfang des Testfahrzeugs beträgt 1330mm. Die Geschwindigkeitsanzeige ist bei hohen Geschwindigkeiten mindestens 2 Mal pro Sekunde zu aktualisieren.

#### Abgastemperatur

Die Abgastemperatur muss über einen bezüglich Temperaturbereich und Trägheit geeigneten Sensor erfasst und in ein zu der Messeinrichtung kompatibles Signal gewandelt werden. Die Verarbeitung des Signals der Abgastemperatur in der Messeinrichtung ist zu gestalten.

##### Erwarteter Wertebereich

Die Abgastemperatur bei laufendem Motor ist mit 200°-750° Celsius anzunehmen. Die Höchsttemperatur kann kurzfristig bis zu 900° Celsius betragen.

##### Aktualisierungsintervall

Die Abgastemperatur soll ein Mal pro Sekunde ermittelt und übertragen werden.

## Auswertung und Darstellung

Die von der Messeinrichtung übermittelten Daten müssen verarbeitet und visualisiert werden. Zu diesem Zweck ist eine Einrichtung zu entwickeln welche ein zu der Messeinrichtung kompatibles Übertragungsverfahren unterstützt und leistungsfähig genug ist, Messdaten aufzuarbeiten und übersichtlich darzustellen.

### Wahl einer Plattform zur Auswertung und Darstellung der Messdaten

Zur Verarbeitung und Darstellung der Messdaten ist eine geeignete Plattform zu wählen. Anforderungen sind die Unterstützung eines zum Messsystem kompatiblen Übertragungsverfahrens sowie die  Möglichkeit, Messdaten übersichtlich zu visualisieren (Display, bevorzugt Grafisch).

### Entwicklung der Algorithmen zur Verarbeitung der Messdaten

Die Verarbeitung der von der Messeinrichtung übermittelten Werte der Drehzahl, des Tachosignals und der Abgastemperatur ist zu gestalten.

### Visualisierung der Messdaten

Die ermittelten Werte der Drehzahl, Geschwindigkeit und der Abgastemperatur sollen dargestellt werden.

## Realisierung der Kommunikation zwischen Mess- und Auswertungseinrichtung

Für die Kommunikation zwischen Mess- und Visualisierungseinheit ist ein geeignetes Verfahren zu wählen und zu implementieren.

### Wahl eines Übertragungsverfahrens

Es ist ein Übertragungsverfahren zu wählen, welches von der Mess- und Auswerteeinrichtung unterstützt wird. Weiter sollte das Übertragungsprotokoll Fehlerkorrektur unterstützen. Ein drahtloses Übertragungsverfahren ist zu bevorzugen, da die Verlegung von Leitungen aufgrund der beengten Verhältnisse am Fahrzeug problematisch ist. Die elektrische Trennung schützt die Einrichtung zur Visualisierung außerdem vor Schäden und verhindert die Einstreuung von Störungen.

### Entwicklung eines Datenformates und Protokolls

Für die Übertragung der Messdaten ist ein Nachrichtenformat zu spezifizieren. Start-, End- und Trennzeichen sind in Form eines einfachen Protokolls zu spezifizieren.

# Lösungsansätze

Dieses Kapitel beschreibt mögliche und evaluierte Lösungsansätze zur Erfüllung der beschriebenen Anforderungen.

## Messeinrichtung

### Drehzahl

Zur Erfüllung der Anforderungen im Hinblick auf die Erfassung der Drehzahl sind verschiedene Lösungsansätze evaluiert worden.

#### Hall-Effekt

Der Hall-Effekt beschreibt eine Spannung, die in einem stromdurchflossenen Leiter innerhalb eines Magnetfeldes auftritt. Dabei kommt es aufgrund der Kraft, die das Magnetfeld auf die bewegte elektrische Ladung auswirkt (vgl. Lorentzkraft [6]), zu einer Ladungstrennung welche eine messbare Spannung erzeugt (vgl. Hall-Spannung [1]). Dies wird in der folgenden Grafik verdeutlicht.

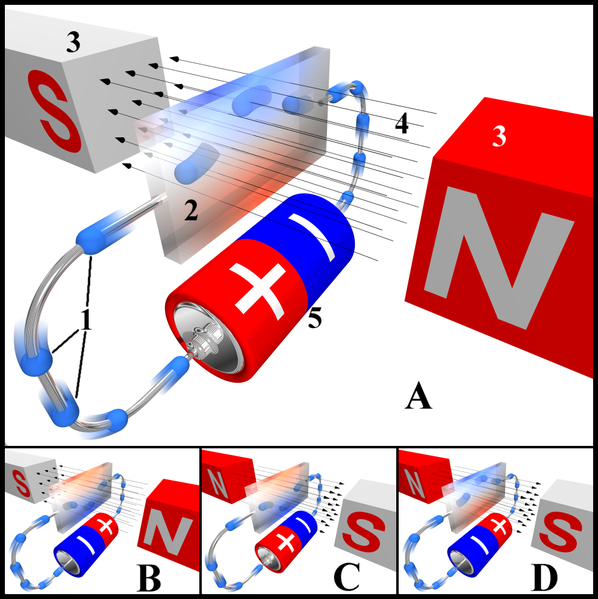


Abbildung ‑ - Hall-Effekt, Quelle: Wikipedia [1]

Es existieren einige verfügbare integrierte Schaltkreise, welche den Hall-Effekt nutzen. Exemplarisch hier die TLE49x5L-Familie von Infineon. Die gemessene Spannung wird verstärkt, ein Schmitt-Trigger sorgt für die binäre Ansteuerung eines Transistors der für einen TTL-Pegel am Ausgang sorgt.

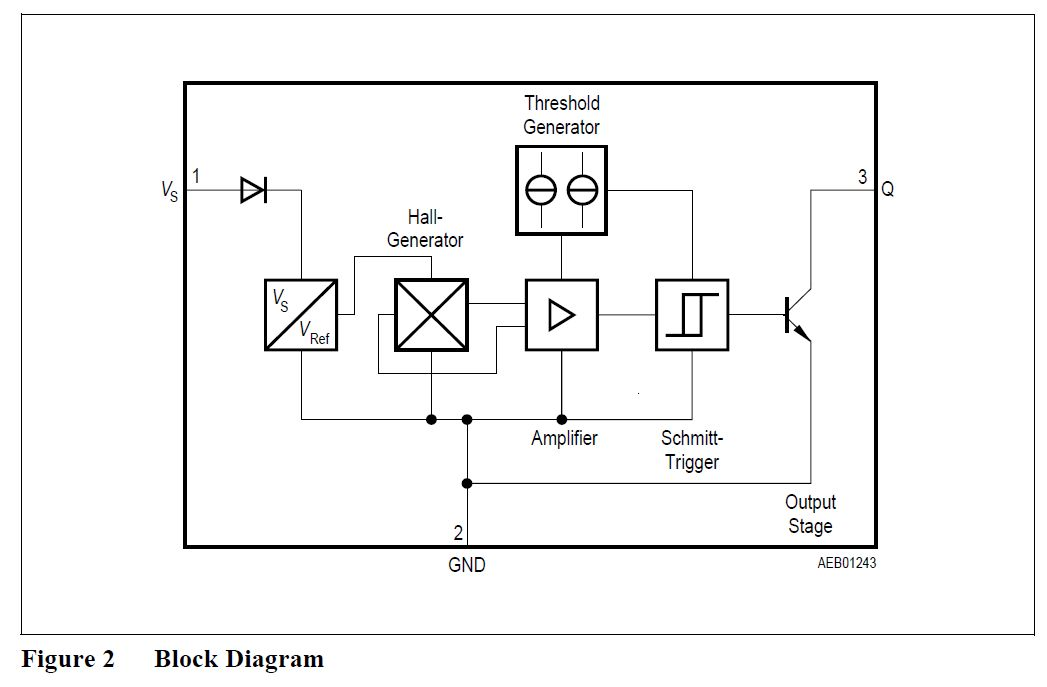


Abbildung ‑ - Blockdiagramm Hall-IC, Quelle: Infineon Datenblatt TLE49x5L [2]

Je nach Ausführung (Unipolar/Bipolar, Switch/Latch) existieren geringfügige Abweichungen, das Prinzip bleibt jedoch identisch. Durch die Art der Realisierung der Vespa-Lichtmaschine, bei der Spulen auf dem Stator durch Dauermagnete auf dem Rotor zur Induktion von Spannung genutzt werden, ist es möglich, einen solchen IC auf dem Stator zu befestigen um die Drehzahl zu erfassen.

Von Vorteil ist, dass die ICs ein sauberes binäres Signal mit TTL-Pegel liefern und keine weitere (analoge) Verarbeitung des Signals vor Einspeisung in den Mikrocontroller nötig ist.

Nachteilig ist die nötige Modifikation am Stator. Der Rotor muss entfernt werden, die zuverlässige Befestigung der Schaltung am Stator ist wichtig. Eine zusätzliche Leitung muss gelegt werden.



Abbildung ‑ - Hall-IC-Schaltung auf Stator

#### Induktion am Zündkabel

Die Vespa-Zündung ist eine Kondensatorentladungszündung. Hier wird durch die Zündspule auf dem Stator ein Kondensator geladen und zum Zündzeitpunkt mit Hilfe eines Thyristors entladen (vgl. <http://de.wikipedia.org/wiki/Kondensatorentladungsz%C3%BCndung>). Hierbei entstehen Spannungen von bis zu 40kVolt an der Zündkerze.

Einige Wicklungen Kupferkabel um das Zündkabel führen zu messbarer Induktionsspannung. Da jedoch viele Parameter, wie die tatsächliche Zündenergie und das Umfeld sowie der Kerzenwiderstand stark variieren, ist die Anzahl der nötigen Wicklungen nicht einfach zu ermitteln und der Aufwand zur Aufarbeitung des Signals erheblich. Systeme wie Leistungsprüfstände (z.B. P4 der Firma Amerschläger) oder Stroboskoplampen zum Einstellen von Zündzeitpunkten arbeiten häufig mit Abnahme über das Zündkabel. Es ist folglich ein gangbarer wenn auch komplexer Ansatz.



Abbildung ‑ - Induktionsmessung am Zündkabel

#### Lichtmaschine

Die Vespa-Lichtmaschine liefert eine Wechselspannung von überlagerten Wellen (mehrere Erzeugerspulen). Die Frequenz ist proportional abhängig zur Drehzahl da der Rotor direkt auf der Kurbelwelle befestigt ist.

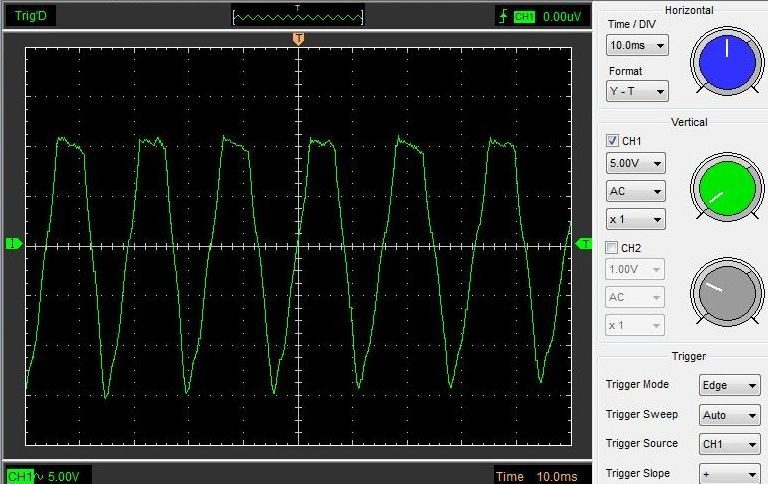


Abbildung ‑ - Lichtmaschinensignal Vespa

Durch Beschaltung mit einer Diode kann das Signal auf die positiven Halbwellen begrenzt werden.



Abbildung ‑ - Lichtmaschine mit Diode

Ein Tiefpassfilter glättet das Signal zusätzlich. Die Spannung wird über einen Spannungsteiler reduziert, eine Z-Diode schützt vor Überspannungen.

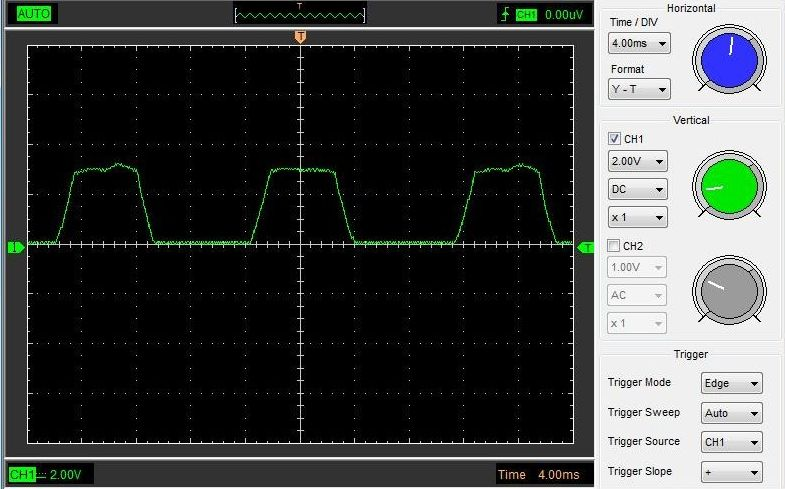


Abbildung ‑ - Lichtmaschinensignal mit Diode, Filter und Spannungsbegrenzung

In dieser Form kann das Signal direkt vom Mikrocontroller verarbeitet werden.

### Tachosignal

Zur Erfüllung der Anforderungen zur Erfassung des Tachosignals sind verschiedene Ansätze evaluiert worden.

#### Reedkontakt

Eine gängige Variante der Ermittlung der aktuellen Geschwindigkeit bzw. der zurückgelegten Distanz ist das Anbringen eines Reedkontakts (Reed-Relais) an der Gabel und eines Magnetes an der Felge. Beim Reed-Relais befindet sich ein Federkontakt innerhalb eines gasgefüllten Kolbens (Korrosionsschutz). Der Kontakt schließt, wenn er in ein Magnetfeld gebracht wird. Durch den Dauermagneten an der Felge wird er ausgelöst. Problematisch hierbei ist, dass Reed-Kontakte eine relativ hohe Prellneigung haben und das Signal so noch entprellt werden muss. Ein weiterer Nachteil ist, dass das Reed-Relais ein mechanisches Bauteil ist und somit Verschleiß unterliegt.

#### Rechnerisch

Die Motordrehzahl lässt sich über die Getriebeübersetzung und den Radumfang in die Geschwindigkeit umrechnen. Umgekehrt erfassen Leistungsprüfstände wie der P4 von Amerschläger so die Motordrehzahl über die Winkelveränderung der Prüfstandsrolle, nachdem die Übersetzung über eine Drehzahlmessung eingemessen wurde. Rutschende oder betätigte Kupplungen verfälschen das Ergebnis jedoch erheblich. Zusätzlich weichen die Getriebeübersetzungen verschiedener Roller sehr stark voneinander ab.

#### Hall-Effekt

Der als Ansatz zur Erfassung der Drehzahl beschriebene Hall-Effekt kann auch für das Tachosignal genutzt werden. Wie im Ansatz mit Reedkontakt beschrieben, werden hierbei Umdrehungen des Rades gemessen. Die Hall-IC-Schaltung wird wie der Reedkontakt an der Gabel angebracht, ein Dauermagnet am Reifen. Als einzig notwendiger Parameter muss der Radumfang bekannt sein. Gegenüber dem Reed-Relais weisen Hall-ICs weder Prellverhalten noch mechanischen Verschleiß auf. Das Ausgangssignal liegt in einer vom Mikrocontroller direkt verarbeitbaren Form vor (TTL (5V) bzw. Low Voltage TTL (3V3) je nach Ausführung).



Abbildung ‑ - Tachosensor-Prototyp befestigt mit Kabelbinder. Dauermagnet an Felge

### Abgastemperatur

Aufgrund des Temperaturbereichs bietet sich als Sensor ein Typ-K Thermoelement an. Themoelemente wandeln Wärme in elektrische Energie (vgl. Thermoelement [3]). Hierbei wird der Seebeck-Effekt genutzt, welcher die auftretende Spannung in einem Stromkreis zwischen zwei verschiedenen elektrischen Leitern bei einer Temperaturdifferenz der Kontaktstellen beschreibt (vgl. Seebeck-Effekt [9]). Die gemessene Spannung kann in Abhängigkeit der Materialkonstanten in eine Temperaturdifferenz zwischen Messstelle und Vergleichsstelle umgerechnet werden. Da es sich nur um einige Mikrovolt handelt, werden hohe Anforderungen an die Genauigkeit der Messeinrichtung gestellt.

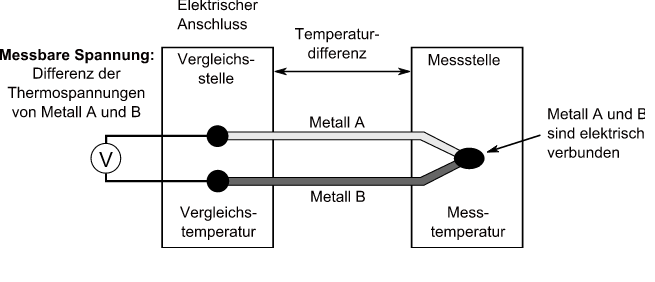


Abbildung ‑ - Thermoelement, Quelle: Wikipedia [3]

#### Umsetzung mit Analogschaltung

Die induzierte Spannung muss verstärkt werden, bevor sie sinnvoll verarbeitet werden kann. Dies kann durch einen Operationsverstärker realisiert werden. Eine Vergleichsstelle ist erforderlich, die Messschaltung muss in jedem Fall manuell eingemessen und justiert werden.

#### IC

Am Markt gibt es integrierte Schaltkreise zur Auswertung von Thermoelementen. Einige realisieren ausschließlich die Verstärkung, sodass eine durch Mikrocontroller sinnvoll messbare Spannung entsteht, andere messen und verrechnen bereits die Vergleichstemperatur. Der MAX6675 von Maxim liefert den Messwert seriell als Zahlenwert mit der Basis 2 (binär). Er kümmert sich also um Verstärkung, Referenz und Digitalisierung.

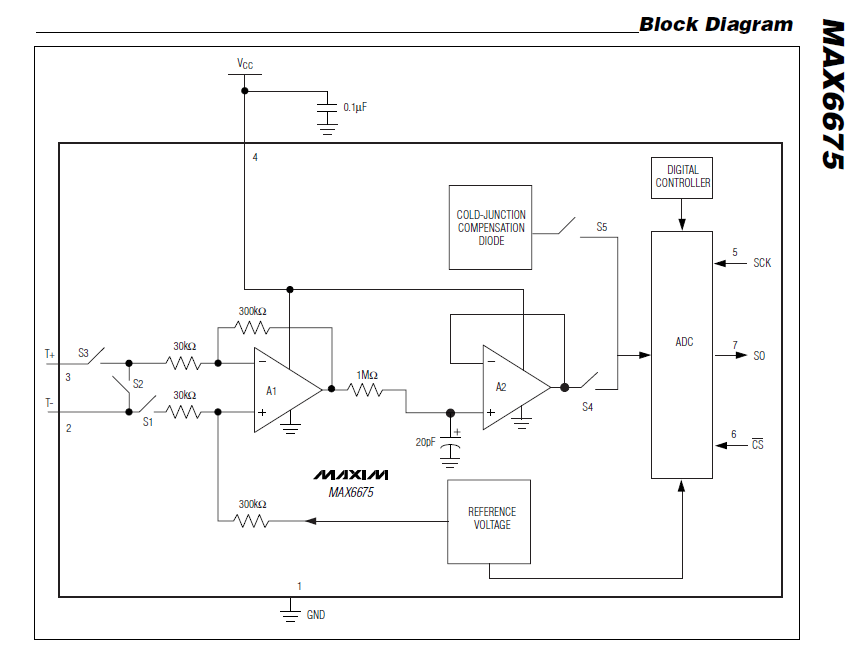


Abbildung ‑ - Blockdiagramm MAX6675, Quelle: Maxim-Datenblatt MAX6675 [4]

Die Ausgabe erfolgt über eine serielle Verbindung.

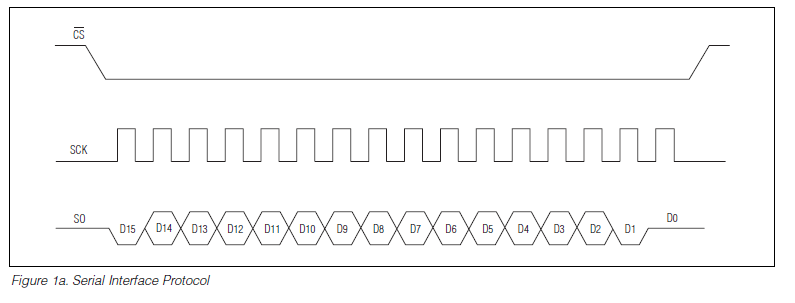


Abbildung ‑ - Protokoll MAX6675, Quelle: Maxim-Datenblatt MAX6675 [4]

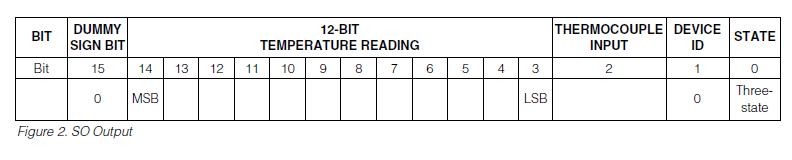


Abbildung ‑ - Datenframe/Bits MAX6675, Quelle: Maxim-Datenblatt MAX6675 [4]

## Signalverarbeitung

Dieser Abschnitt befasst sich mit Lösungsansätzen zur Verarbeitung der Sensordaten in der Messeinrichtung. Mikrocontroller der Atmega-Familie verfügen über Analog-Digital-Wandler sowie digitale Eingänge mit verschiedenen Interruptquellen. Geeignete Anschlüsse sind zu wählen und die Verarbeitung zu realisieren.

### Drehzahl / Tacho

Atmega-Mikrocontroller bieten die Möglichkeit, eine angelegte Spannung mit einer internen oder externen Vergleichsspannung zu vergleichen und bei Veränderungen Interrupts auszulösen. Dies bietet die Möglichkeit, Flankenwechsel bei einer (durch externe Beschaltung mit Spannungsteiler) wählbaren Spannung zu verarbeiten. In der Evaluation wurden ebenfalls gute Erfahrungen mit der Nutzung eines Pins der als externe Interruptquelle konfiguriert wurde gemacht, da Atmega-Mikrocontroller über integrierte Schmitt-Trigger (vgl. Schmitt-Trigger [10]) verfügen die das Signal vom Drehzahlsensor wertdiskretisieren (beim Tachosensor liegt das Signal bereits digital vor).   
Diese Interruptpins lösen Interrupts wahlweise bei positiver oder negativer Flanke (oder jedem Flankenwechsel) aus. So kann, wird gleichzeitig ein Timer zur Erfassung einer „Systemzeit“ genutzt, bei jeder erfassten Flanke die Zeitdifferenz (in Takten) zur letzten Messung erfasst werden. Außerhalb der Interruptroutine kann diese dann weiter verarbeitet (an die Auswerteeinheit übermittelt) werden. Innerhalb der Auswerteeinheit muss der zum Systemtakt relative Wert anschließend in die entsprechende Einheit gewandelt werden.

#### Beispiele

CLK = Systemtakt = 4\*10^6 Hz  
CLKDIV = Prescaler des Timers der Systemzeit = 256  
TPS = Ticks pro Sekunde des Systemzeit-Timers = CLK / CLKDIV = 15625 Hz  
T\_DIFF = Gemessene Takte zwischen zwei Flanken der Lichtmaschine bzw. zwischen zwei Flanken des Tachosensors  
SPR = Signale pro Umdrehung (zB. Zündung: zwei Signale aufgrund mehrerer Lichtspulen, Tacho: ein Signal aufgrund eines Magneten am Rad)  
U = Umfang des Rades mit dem Tachosensor in cm

##### Drehzahl in Umdrehungen pro Minute

RPM = TPS/T\_DIFF/SPR\*60

##### Geschwindigkeit in Kilometern pro Stunde

V = TPS/T\_DIFF/SPR\*3600\*U/10^5

Der Initialzustand bei Einschalten des Systems ist bei der Implementierung zu berücksichtigen und zu behandeln (ansonsten je nach Realisierung der Berechnung Auftreten von Fehlern aufgrund von Division durch 0 möglich).

### Abgastemperatur

Abbildung 5.12 zeigt den Aufbau eines Datenpaketes, Abbildung 5.11 die Kommunikation und Bitfolge. Für den Einsatzzweck reicht eine Auflösung von ~10 Grad Celsius. Zur genaueren Darstellung ist eine Abstufung bis auf 1 Grad Celsius wünschenswert. Die 12 Bit des Max6675 jedoch lösen mit einer Genauigkeit von 0,25 Grad Celsius auf. Für die Temperaturauswertung werden folglich nur die relevanten Bits 5-14 ausgewertet. Die Abwicklung der Kommunikation ist wie folgt zu realisieren:

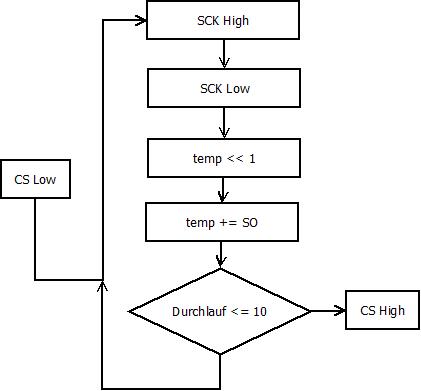


Abbildung ‑ - Ablaufdiagramm Kommunikation mit Max6675

Zunächst wird CS low gesetzt und der IC dadurch vom Mess- in den Kommunikationsmodus versetzt. Nun wird 10 Mal ein CLK-Toggle ausgeführt wodurch das nächste Datenbit auf SO abgerufen wird, welches in die Temperaturvariable „eingeshiftet“ (shift plus Addition) wird. Durch den Toggle im ersten Durchlauf wird das dummy-sign-bit (Bit 15) übersprungen. Dies ist beabsichtigt.

Zum Schluss wird CS wieder High gesetzt und der IC somit wieder in den Mess-Modus versetzt.

Das Diagramm verzichtet zur besseren Übersicht auf die Initialisierung der Wertvariablen „temp“ sowie auf Initialisierung und Manipulation der Durchlaufvariablen.

Bei der Implementierung ist darauf zu achten, dass der Max6675 keine so schnellen Schaltzeiten/Taktungen unterstützt wie Atmega-Mikrocontroller. Nach Flankenwechseln und zwischen einzelnen Messungen sind ggfs. Delays einzubauen.

## Datenübertragung

Die Datenübertragung zwischen der Mikrocontroller-Messeinrichtung und dem System zur Auswertung und Visualisierung ist zu realisieren.

### Seriell über Kabel

Da sowohl die Mikrocontroller der Atmega-Familie als auch das in der Evaluation zur Visualisierung getestete Android-Smartphone über integrierte UART-Schnittstellen verfügen, ist die serielle Kommunikation über Kabel naheliegend. Leider sind die für die Kommunikation mit dem HTC-Smartphone benötigten Stecker teuer (da proprietär), kaum verfügbar und schwer zu löten. Weiter wird ein „gerootetes“ Gerät (Gerät mit administrativem Zugriff) benötigt, da der Zugriff auf die serielle Schnittstelle nicht vorgesehen ist. Zu guter Letzt verfügt nicht jedes Android-Gerät überhaupt über eine serielle UART-Schnittstelle (oder diese ist nur schwer im Gehäuseinneren zugänglich). Dennoch wurde dieser Ansatz als erster evaluiert. In Abbildung 5.14 ist ein Testlauf dokumentiert.

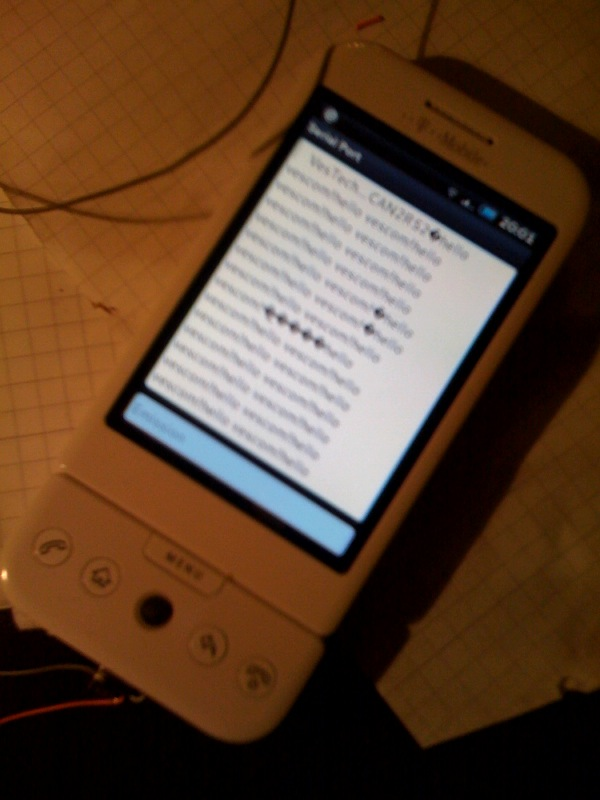


Abbildung ‑ -Mikrocontroller kommuniziert über serielle Kabelverbindung mit Android-Mobiltelefon

#### Can-Bus

Das störbehaftete Umfeld am Roller und andere Nachteile der seriellen Kommunikation haben zunächst zu der Evaluation der Übertragung der Daten via Can-Bus geführt. Diese Kommunikationsmöglichkeit wurde als gangbar vor allem zwischen zwei Mikroncontrollern mit einigem Abstand (zum Beispiel Messsystem im Motorbereich und Visualisierungseinrichtung im Lenkerbereich) befunden. Wenn jedoch Android-Geräte zur Visualisierung eingesetzt werden sollen bedarf es zusätzlich einer weiteren Übertragungseinrichtung, da diese über keine Can-Schnittstelle verfügen. Des Weiteren ist der Schaltungsaufwand erheblich.

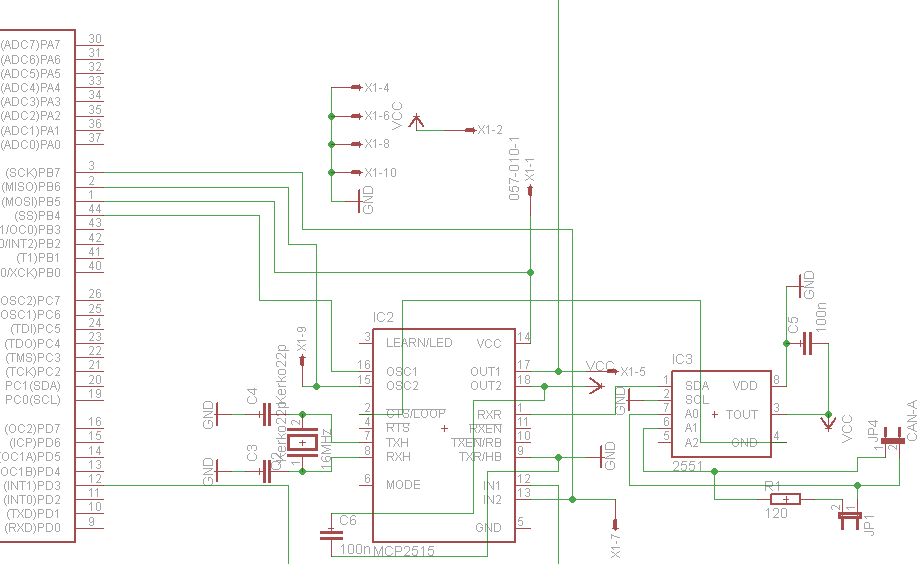


Abbildung ‑ - Auszug des Schaltplans für Kommunikation via Can-Bus

### Bluetooth

Alle gängigen Android-Geräte verfügen über Bluetooth-Hardware. Die Nutzung als serielle Schnittstelle ist ohne Erlangung von Administrationsprivilegien möglich. Weiter gibt es fertige ICs, welche über eine UART-Schnittstelle empfangene Daten per Bluetooth versenden. Durch die einfache Instrumentalisierung, eingebaute Fehlerkorrektur und entkoppelte Kommunikation (und die hierdurch nicht vorhandene Gefahr der physikalischen Schädigung der Android-Geräte durch die Messschaltung) sowie die geringen Kosten, stellt Bluetooth eine ideale Übertragungsmöglichkeit dar.

## Auswertung und Darstellung

Dieser Abschnitt befasst sich mit Lösungsansätzen zur Auswertung und Visualisierung der erfassten Messdaten.

### Mikrocontroller mit LCD-Display

Ein erster Prototyp verfolgte den Ansatz, ein LC-Display im Lenker unterzubringen und den Rest der Schaltung nahe des Motors.



Abbildung ‑ - Display eines Prototypen am Vespa-Lenker

Leider wurden hierbei Störungen durch die langen Datenleitungen eingestreut. Diese konnten auch durch Entstörwiderstände nicht vollständig eliminiert werden.



Abbildung ‑ - Display mit Störungen

Ein späterer Prototyp enthält einen Mikrocontroller, die CAN-Beschaltung, welche im vorangegangenen Kapitel erläutert wurde, sowie zwei Taster und das Display. Schaltungen für Drehzahl- und Tachoauswertung sind direkt integriert. Die Messschaltung für die Abgastemperatur sollte per CAN-Bus angekoppelt werden.

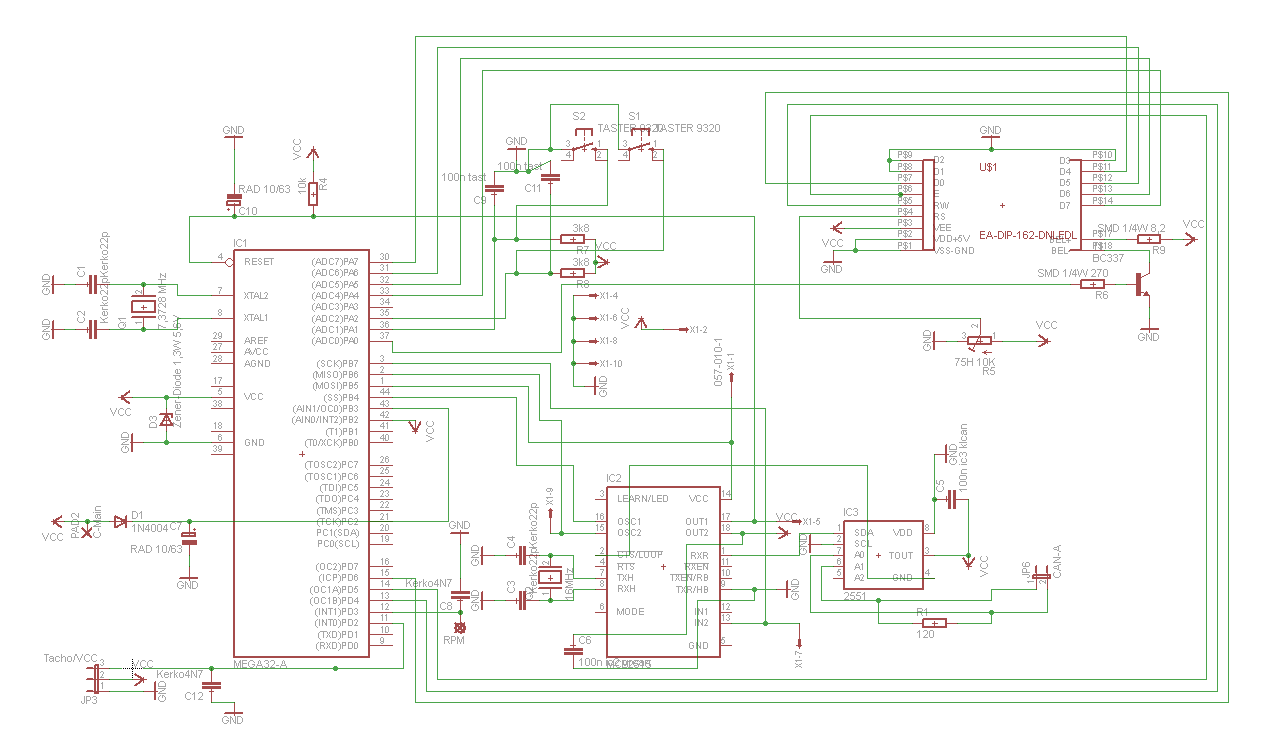


Abbildung ‑ - Schaltplan Prototyp integrierte Mess-, Auswerte- und Visualisierungseinheit

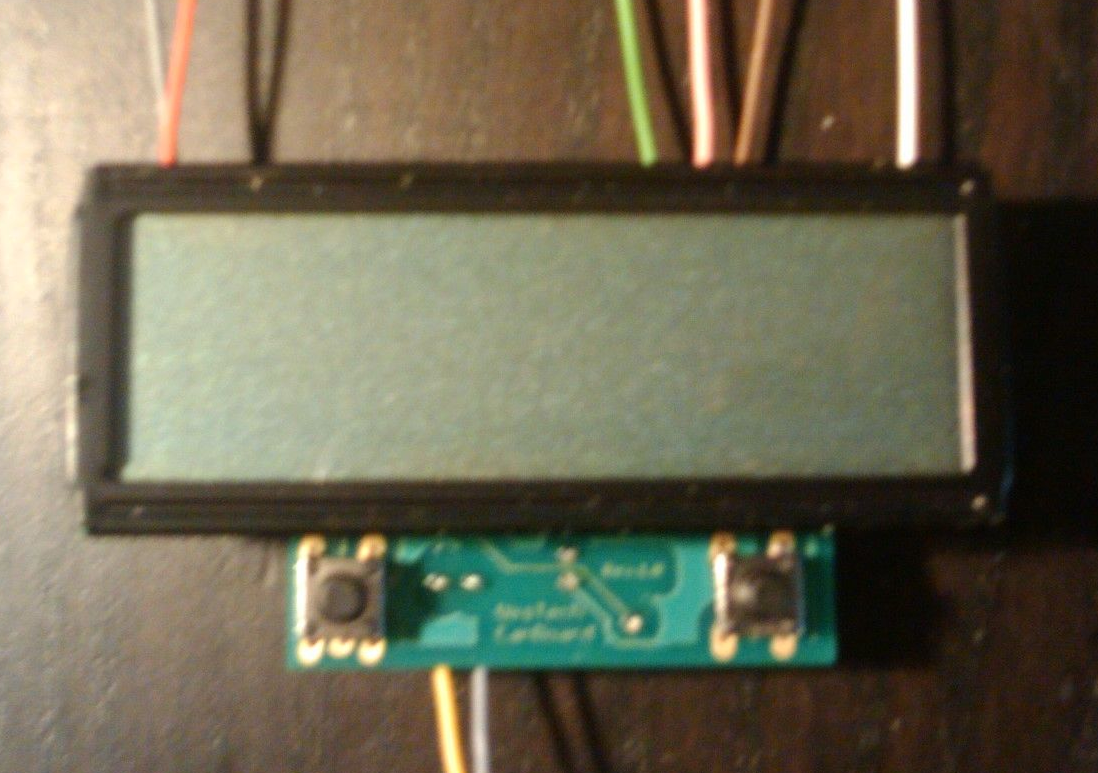


Abbildung ‑ - Prototyp integrierte Mess-, Auswerte- und Visualisierungseinheit

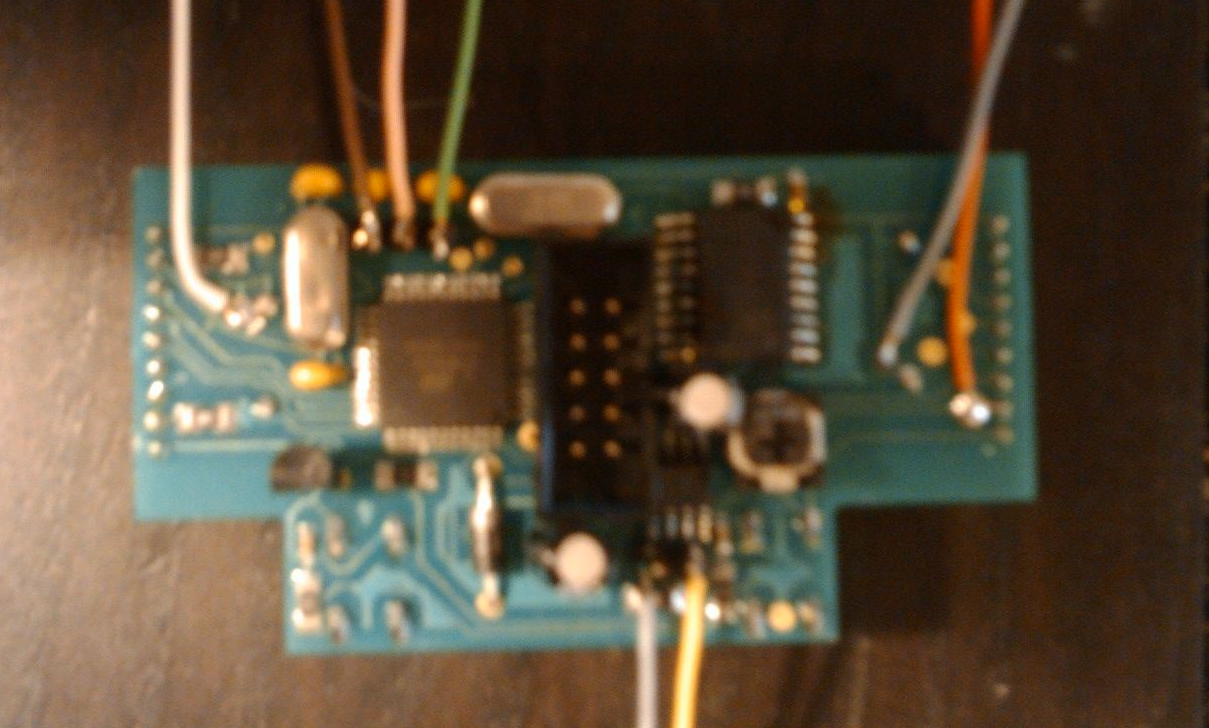


Abbildung ‑ - Prototyp integrierte Mess-, Auswerte- und Visualisierungseinheit

Dieser Weg erschien zunächst vielversprechend. Aufgrund von beengten Platzverhältnissen im Vespa-Lenker und beweglicher Teile wie der Schalt- und Gasrolle sowie des bei manchen Modellen vorhandenen Zündschlosses im Lenker, ist die Unterbringung aber schwierig, und es kam immer wieder zu Beschädigungen an den Prototypen. Auch drang regelmäßig Flüssigkeit in das System ein (die Taster mussten nutzbar bleiben um die Ansichten zu wechseln, was die Abdichtung erschwerte).

### Android

Der letzte evaluierte Lösungsweg ist die Nutzung eines Android-Gerätes. Der große Vorteil hierbei ist, dass die Auswertung und Visualisierung innerhalb eines viel leistungsfähigeren Umfeldes erfolgen kann. Bei Berechnungen ist nicht auf überlaufende Register zu achten, Visualisierung kann viel komfortabler als auf einem LC-Display erfolgen. Auch kann so der Fokus stärker auf die Entwicklung der Messschaltungen und Messalgorithmen gesetzt werden, da der Aufwand im Bereich Auswertung/Darstellung sinkt.

# Anforderungsdefinition, Systementwurf und Realisierung

Dieses Kapitel befasst sich mit dem Entwurf und der Realisierung des System

## Systemübersicht und Modulbeschreibungen

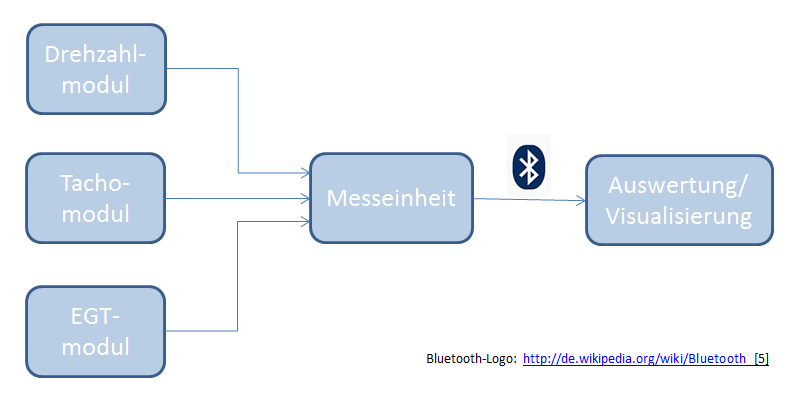


Abbildung ‑ - Systemübersicht

|  |  |  |  |
| --- | --- | --- | --- |
| **Modul** | **Zweck** | **Eingänge** | **Ausgänge** |
| Drehzahlmodul | Wandlung des Lichtmaschinensignals in ein zu der Messeinheit kompatibles Signal (annähernd Rechteck, direkt proportional zur Motordrehzahl) | - | Motorumdrehungen |
| Tachomodul | Erfassung der Radumdrehungen durch Hall-Effekt und Übertragung als zur Messeinheit kompatiblem Signals (Rechteck, entsprechend der Radumdrehungen) | - | Radumdrehungen |
| EGT-Modul | Wandlung des Ausgangssignals eines Typ-K Thermoelements (Spannung) in ein zur Messeinheit kompatiblen Signals (binär, serielle Übertragung) | - | Abgastemperatur |
| Messeinheit | Erfassen der Frequenzen/Zeitdeltas von Drehzahl- und Tachomodul sowie Verarbeitung der Daten vom EGT-Modul. Übertragung via Bluetooth | Motorumdrehungen Radumdrehungen Abgastemperatur | Datenpakete für Auswertung/ Visualisierung |
| Auswertung/ Visualisierung | Empfang der Datenpakete der Messeinheit via Bluetooth. Auswertung und Visualisierung der Messdaten. | Datenpakete der Messeinheit | - |

## Datenpaket

Das Datenpaket, welches von der Mess- an die Auswerte-/Visualisierungseinheit verschickt wird, enthält folgende Daten

|  |  |
| --- | --- |
| **Feld** | **Beschreibung** |
| PACKET\_ID | Paketnummer |
| RPM\_DIFF | Takte zwischen den letzten zwei Flanken am Lichtmaschinensensor |
| V\_DIFF | Takte zwischen den letzten zwei Flanken am Radumdrehungssensor |
| EGT | Abgastemperatur |

## Messsystem

Das Drehzahlmodul wird über den unter Lösungsansätze/Drehzahl/Lichtmaschine beschriebenen Ansatz realisiert.

Das Tachomodul wird über den unter Lösungsansätze/Tachosignal/Hall-Effekt beschriebenen Ansatz realisiert.

Die Abgastemperatur wird über den unter Lösungsansätze/Abgastemperatur/IC beschriebenen Ansatz mit Hilfe des Maxim ICs Max6675 realisiert.

Angeschlossen werden die Messschaltungen an einen Mikrocontroller der Atmega-Familie, der über den USART mit einem Bluetooth-Modul kommuniziert.

### Plattform

Das Entwicklungsboard Bluecontroller ([www.bluecontroller.com](http://www.bluecontroller.com)) wird als Basis genutzt. Es enthält das BTM-222 Bluetooth-Modul, einen 3,3V Spannungsregler für das Bluetooth-Modul (welches auf 3,3 statt 5V arbeitet), einen Atmega-Mikrocontroller, den nötigen Levelshifter zwischen Bluetooth-Modul und Mikrocontroller, Status-LEDs, zwei Taster und eine Chipantenne. Bei sehr kleiner Bauform und günstigem Preis reduziert das Board somit Entwicklungsarbeit und Kosten bei der Umsetzung.

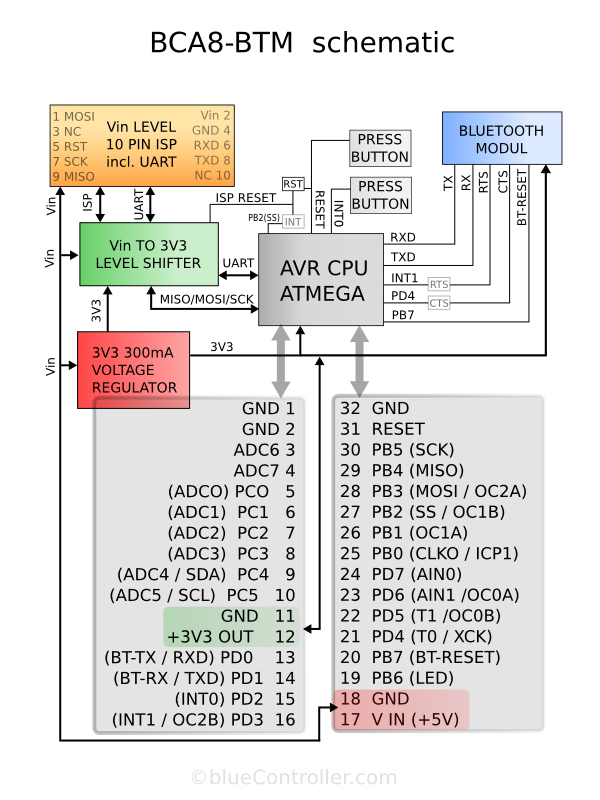


Abbildung ‑ -Blockschaltbild Bluecontroller, Quelle: www.bluecontroller.com

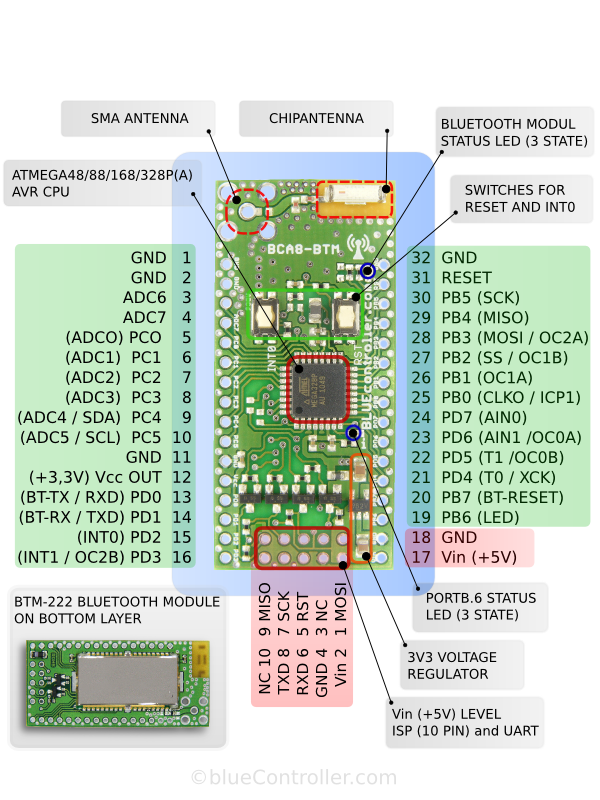


Abbildung ‑ - Bluecontroller, Bordlayout mit Legende, Quelle: www.bluecontroller.com

### Schaltpläne

#### Gesamtschaltplan und Layout Messschaltung

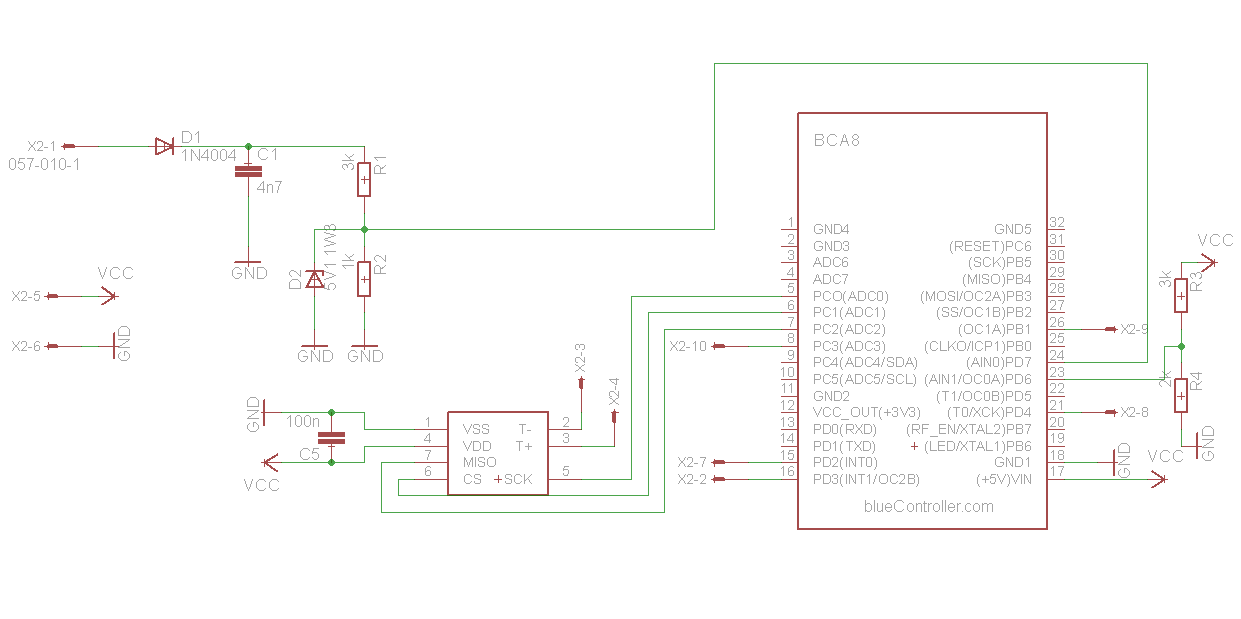


Abbildung ‑ - Gesamtschaltplan Messsystem

Der Gesamtschaltplan zeigt die Sensorschaltung zur Drehzahlerfassung um C1. Weiter ist der Max6675 zur Erfassung der Abgastemperatur sowie der Bluecontroller beschaltet.

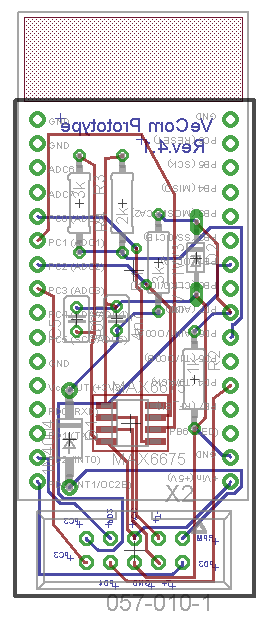


Abbildung ‑ - Bordlayout Messsystem

#### Drehzahlerfassung

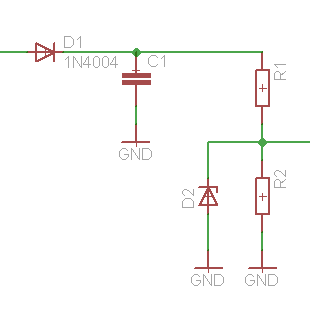


Abbildung ‑ - Schaltplan Drehzahlerfassung Messsystem

Die Messschaltung zur Erfassung der Drehzahl besteht aus einer Diode D1 zur Beschränkung auf die positive Halbwelle, einem Kondensator C1 zur Glättung und in Folge einem Spannungsteiler zur Begrenzung der Amplitude und einer Zenerdiode D2 zum Schutz des Eingangs am Mikrocontroller. Die Dimensionierung des Spannungsteilers muss derart erfolgen, dass bei schwankender Bordspannung die Schaltschwelle der Schmitt-Trigger im Mikrocontroller nicht unterschritten, die maximale Eingangsspannung jedoch nicht überschritten wird. Weiter muss der Widerstand R1 derart ausgelegt werden, dass der Strom durch D2 bei Spannungsspitzen/hoher Bordspannung D2 nicht schädigt.

#### Auswertung Thermoelement (Abgastemperatur)

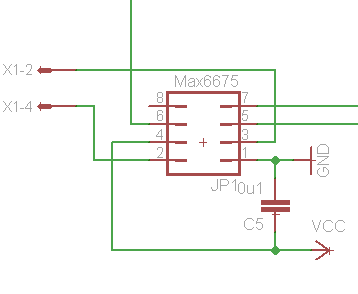


Abbildung ‑ - Schaltplan Beschaltung Max6675

Die Beschaltung des Maxim-ICs Max6675 zur Auswertung von Typ K Thermoelementen gemäß Datenblatt.

#### Erfassung Tachosignal

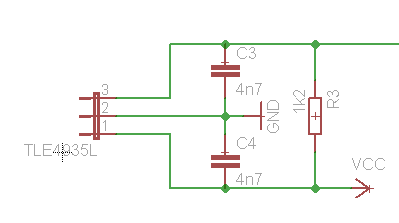


Abbildung ‑ - Schaltplan Beschaltung TLE49x5

Die Beschaltung des Hall-ICs TLE4935L von Infineon zur Erfassung des Tachosignals gemäß Datenblatt.

### Detaillösungen Software Messsystem

In diesem Abschnitt sollen zum Verständnis hilfreiche Detaillösungen aufgeführt werden. Für den gesamten Quelltext stehen die Codelistings im Anhang sowie der beigefügte Datenträger zur Verfügung.

#### Schiebealgorithmus EGT/Max6675

Der hier als Codeauszug aufgeführte Algorithmus implementiert den unter Lösungsansätze/Signalverarbeitung/Abgastemperatur modellierten Algorithmus.

/\*\*

\* communciates with max6675 to determine egt

\* all delay-calls are because the max6675 does not support timings as fast as the mega

\*/

uint16\_t get\_egt(){

uint16\_t temp=0;

//cs low for data

EGT\_PORT &= ~(1<<EGT\_CS);

\_delay\_ms(1);

//now go through relevant bits and shift them in

for(int i=0; i<10; i++){

//clk-toggle for next bit (skips irrelevant msb in first iteration)

EGT\_PORT |= (1 << EGT\_CLK);

\_delay\_ms(EGT\_DELAY);

EGT\_PORT &= ~(1 << EGT\_CLK);

\_delay\_ms(EGT\_DELAY);

// shift

temp = temp << 1;

// add pin state (0 or 1)

temp += EGT\_PIN & (1<<EGT\_DATA\_PIN);

}

//cs high for measure

EGT\_PORT |= (1<<EGT\_CS);

return temp;

}

#### Paketserialisierung

Die Serialisierung und Deserialisierung von Datenpaketen von Messsystem und Visualisierungseinrichtung ist übersichtlich im Abschnitt „Paketserialisierung/Deserialisierung“ zusammengefasst.

## Einrichtung zur Auswertung und Visualisierung

### Klassendiagramme

Im Folgenden wird der softwaretechnische Systementwurf für die Erstellung der Android-Software zum Empfang, der Auswertung und Visualisierung der Messdaten mit Hilfe von Klassendiagrammen beschrieben. Vertiefend wird das Studium der Programmquellen aus der Anlage oder dem beigefügten Datenträger empfohlen. Begonnen wird mit einer Übersicht. Anschließend werden die einzelnen Pakete beschrieben.

Zunächst das Klassendiagramm aller im Rahmen dieser Arbeit zu erstellenden Klassen.

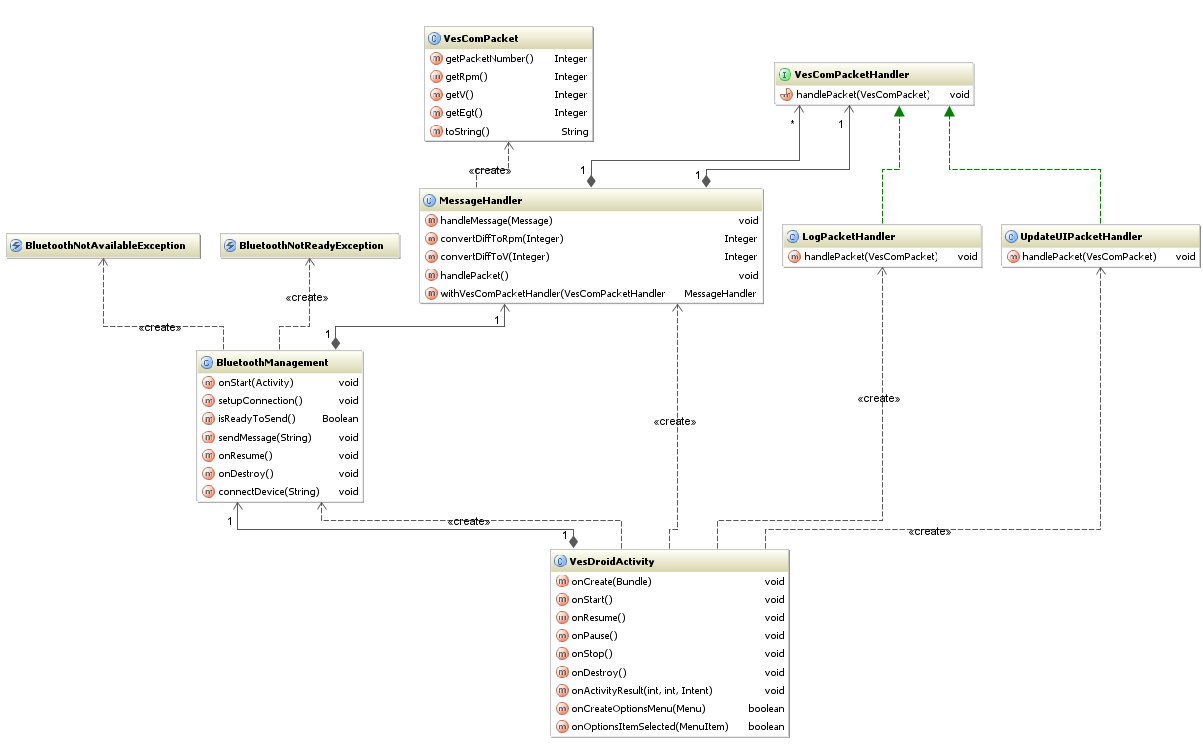


Abbildung ‑ - Klassendiagramm Visualisierungseinheit

#### Paket de.gaffa.vescom.vesdroid

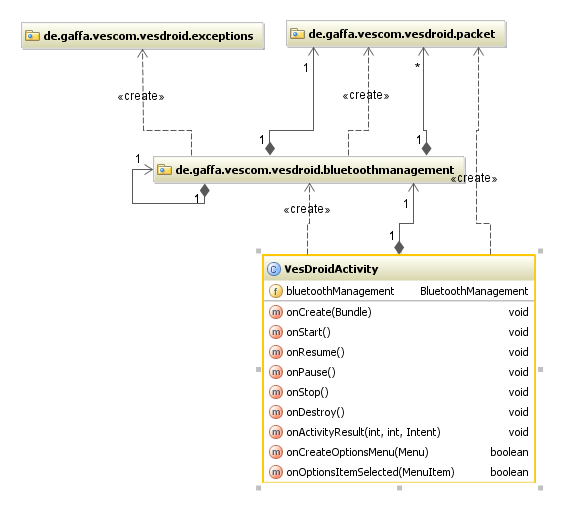


Abbildung ‑10 - Klassendiagramm Paket de.gaffa.vescom.vesdroid

Das Paket *de.gaffa.vescom.vesdroid* ist das „Basispaket“ der Anwendung. Es enthält die Pakete *exception*, *packet* und *bluetoothmanagement*, deren Klassen in den Folgeabschnitten beschrieben werden sowie die Klasse *VesDroidActivity*, welche die Android-Activity zur Visualisierung der Daten enthält und alle anderen Objekte initialisiert/instanziiert.

#### Paket de.gaffa.vescom.vesdroid.bluetoothmanagement

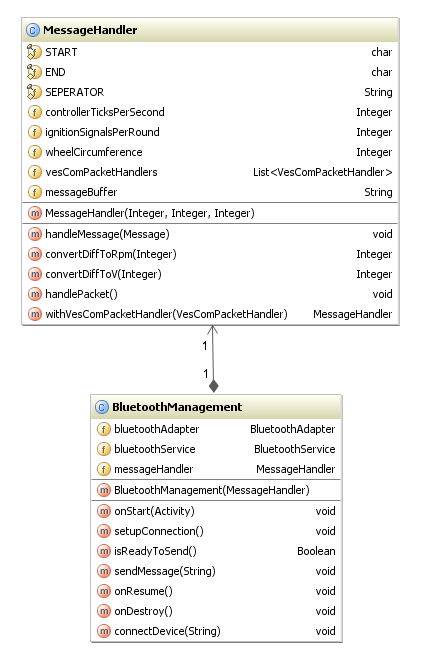


Abbildung ‑11 – Klassendiagramm Paket de.gaffa.vescom.vesdroid.bluetoothmanagement

Das Paket *de.gaffa.vescom.vesdroid.bluetoothmanagement* ist für die Bluetooth-Kommunikation zuständig. Es enthält die Klassen *Bluetoothmanagement* und *MessageHandler*. Die Aufgabe von *Bluetoothmanagement* ist die Verwaltung des Bluetooth-Adapters sowie das Erstellen und Pflegen von Verbindungen zu Geräten (dem Messsystem). *Messagehandler* ist dafür zuständig, die seriell erhaltenen Daten in ein vereinbartes Format (*VesComPacket*) zu parsen und registrierte Handler vom Typ *VesComPacketHandler* mit der weiteren Verarbeitung zu beauftragen. Hier erkennen wir das Observer-Pattern. Beliebige Klassen, die das entsprechende Interface implementieren können als Handler, durch den Aufruf von *withVesComPacketHandler()* registriert werden.

#### Paket de.gaffa.vescom.vesdroid.packet

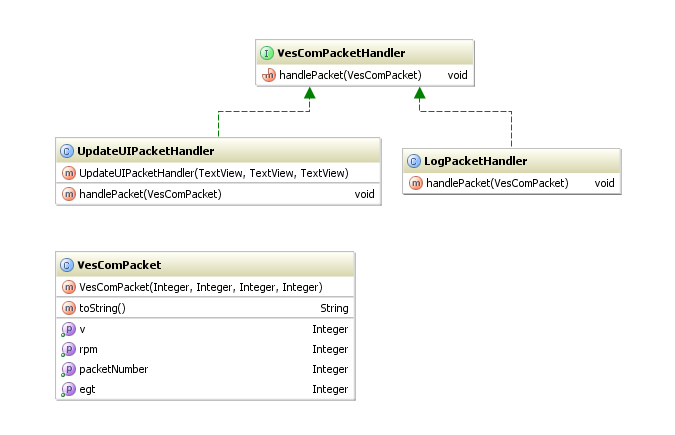


Abbildung ‑12 - Klassendiagramm Paket de.gaffa.vescom.vesdroid.packet

Das Paket *de.gaffa.vescom.vesdroid.packet* enthält das Objekt VesComPacket, das Interface VesComPacketHandler, welches Handler implementieren müssen, die ein VesComPacket bei Empfang verarbeiten wollen, sowie die Implementierungen UpdateUIPacketHandler, welche sich um die Aktualisierung der Benutzeroberfläche (also der eigentlichen Visualisierung) kümmert, und LogPacketHandler, welche die empfangenen Daten ins Systemlog schreibt.

#### Android SDK: BluetoothChat

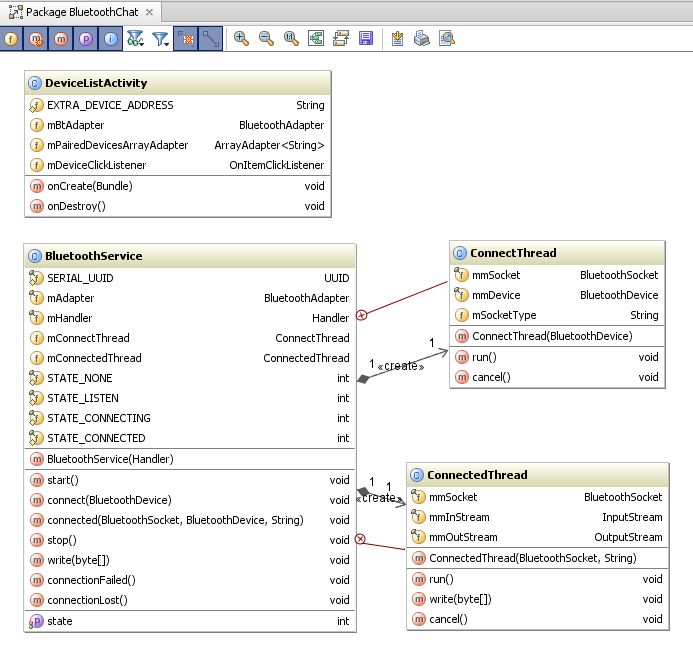


Abbildung ‑ - Klassendiagramm Android SDK: BluetoothChat

Das im Android SDK enthaltene Beispielprogramm Bluetoothchat wurde als Basis für die Kommunikationsimplementierung in der Visualisierungseinrichtung genutzt. So wurde die Klasse DeviceListActivity zur Auswahl eines gekoppelten Bluetoothgerätes verwendet und auch die Threadmanagementimplementierung für Verbindungen des BluetoothSerivce mit Modifikationen übernommen.

## Paketserialisierung/Deserialisierung

Zur Kommunikation zwischen der Messeinrichtung und der Einrichtung zur Verarbeitung und Visualisierung der Messdaten wurde ein Datenpaket vereinbart (siehe „Anforderungen, Systementwurf und Realisierung/Datenpaket“). Dieses Paket muss auf Seite der Messeinrichtung seriell via Bluetooth übertragen und auf Seite der Einrichtung zur Verarbeitung und Visualisierung deserialisiert werden, damit es weiter verarbeitet werden kann. Im Folgenden werden die Lösungen für dieses Problem anhand von Codebeispielen erläutert.

### Serialisierung in der Messeinheit

Der Beginn des Pakets wird durch einen Stern (Ascii 42, „\*“) markiert. Es folgt die aktuelle Paketnummer, wiederum gefolgt von einem Komma (Ascii 44, „,“). Nun werden noch Drehzahltaktdifferenz, Tachotaktdifferenz und Abgastemperatur, jeweils getrennt durch ein Komma, hinzugefügt. Das Ende des Pakets wird durch eine Raute (Ascii 35, „#“) markiert. Die so aufgebaute Zeichenkette wird dann über die serielle Schnittstelle an das Bluetooth-Modul übermittelt, welches sie überträgt.

/\*\*

\* sends a vescompacket via uart from given values

\*/

void sendPacket(int rpm\_diff, int v\_diff, int egt){

char message[29];

int pointer = 0;

message[pointer] = '\*';

pointer++;

pointer += addIntToString(message, pointer, getPacketNumber());

message[pointer] = ',';

pointer++;

pointer += addIntToString(message, pointer, rpm\_diff);

message[pointer] = ',';

pointer++;

pointer += addIntToString(message, pointer, v\_diff);

message[pointer] = ',';

pointer++;

pointer += addIntToString(message, pointer, egt);

message[pointer] = '#';

message[pointer+1] = '\0';

uart\_puts(message);

}

Die hier genutzte Funktion addIntToString() wandelt einen Integer-Wert in ein Character-Array (String) und fügt sie dem als Parameter übergebenen String hinzu. Die Funktion getPacketNumber() ist dafür zuständig, einen Paketzähler zu inkrementieren und somit stets die nächsthöhere Paketnummer zur Verfügung zu stellen, sodass eine fortlaufende Paketnummerierung gewährleistet wird.

### Deserialisierung in der Visualisierungseinheit

Im MessageHandler wird ein Buffer vorgehalten. Dieser wird beim Empfang des Startzeichens („\*“) neu initialisiert. Darauf folgende Zeichen werden bis zum Empfang des Endzeichens („#“) in den Puffer geschrieben.

public void handleMessage(Message msg) {

// handle message received event

if (msg.what == Properties.MESSAGE\_READ) {

byte[] readBuf = (byte[]) msg.obj;

// construct a string from the valid bytes in the buffer

String readMessage = new String(readBuf, 0, msg.arg1);

//mConversationArrayAdapter.add(mConnectedDeviceName + ": " + readMessage);

for (char c : readMessage.toCharArray()) {

if (c == START) {

messageBuffer = new String();

} else if (c == END) {

handlePacket();

} else if (messageBuffer != null) {

messageBuffer += c;

}

}

}

}

Beim Empfang des Endzeichens („#“) wird die Methode handlePacket() aufgerufen, welche die Deserialisierung vornimmt. Hierbei wird der Puffer am Seperator (“,”) geteilt. Sind danach vier Werte vorhanden, wird von einem gültigen Paket ausgegangen und die empfangenen Daten werden verarbeitet und in eine VescomPacket-Instanz überführt.

/\*\*

\* converts a bluetooth-message to a vesdroid-packet and notifies all handlers

\*/

private void handlePacket() {

// create value-array from string by splitting at the separator-character

String[] fields = messageBuffer.split(SEPARATOR);

try {

if (fields.length != 4) {

// message seems to be invalid as 4 values are expected

throw new Exception("Message contained more or less than 4 fields");

}

// assign values

Integer packetNumber = Integer.parseInt(fields[0]);

Integer rpm = convertDiffToRpm(Integer.parseInt(fields[1]));

Integer v = convertDiffToV(Integer.parseInt(fields[2]));

Integer egt = Integer.parseInt(fields[3]);

// create packet with values

VesComPacket packet = new VesComPacket(packetNumber, rpm, v, egt);

// notify all registered handlers

for (VesComPacketHandler vesComPacketHandler : vesComPacketHandlers) {

// just put entries in the log if anything goes wrong with a single handler. we still want

// the other handlers to get notified

try {

vesComPacketHandler.handlePacket(packet);

} catch (Throwable th) {

Log.e("VesDroid", "The handler: " + vesComPacketHandler + " caused an Exception: ", th);

}

}

} catch (Throwable tr) {

// log all exceptions that occur (even

Log.e("VesDroid", "Fatal. A Message could not be converted to a VesDroid-Packet.", tr);

}

}

So ist die Visualisierung der nunmehr aufgearbeiteten Daten möglich. Zum tieferen Verständnis der Implementierung wird das Studium der beigefügten Quellen empfohlen.

# Zusammenfassung und Ausblick

Bei der Lösung der Aufgabenstellung hat sich die Arbeit mit analogen Signalen aus der „realen Welt“ schwieriger als erwartet dargestellt. Störungsumfelder und nicht erwartete Signalschwankungen wirken sich gravierend aus. Schnell führt eine Rechenoperation auf einen fehlerhaft erfassten Wert zum Controllerabsturz oder wird ein Eingangspin oder ein ganzer Mikrocontroller zerstört. Solche unvorhergesehenen Ereignisse haben die Verteilung des zeitlichen Aufwandes in erheblichem Maße gegen die Erwartung/Planung beeinflusst sodass der Aufwand der praktischen Umsetzung deutlich über der Zeitplanung lag. Dennoch sind alle maßgeblichen Kriterien erfüllt und das Projekt kann als erfolgreich bewertet werden.

Der Fahrer wird durch das entwickelte Instrument in die Lage gebracht, die Fahrzeugdaten Drehzahl, Abgastemperatur und Geschwindigkeit übersichtlich auf einer einzigen Visualisierungseinrichtung zu erfassen und somit bei der Anpassung der Motordrehzahl und der Überwachung des Verbrennungsvorganges unterstützt.

Zu einer Markt- bzw. Serienreife fehlen noch Konfigurationsfeatures (zum Beispiel des Reifenumfangs, der Anzahl der Magnete am Reifen, der Anzahl der Spulen/Magnete der Lichtmaschine). Auch hat das minimale Benutzerinterface der Visualisierungseinrichtung Prototypcharakter. Es kommt noch zu „Ausbrechern“ bei den Messwerten von Drehzahl und Geschwindigkeit.

Theoretisch mögliche Features, wie das Errechnen einer Leistungskurve aus den erfassten Daten, würde ein Datenübertragungsprotokoll erfordern, welches mit dem Verlust von Paketen umgehen kann (zB. erneutes Senden der Messeinrichtung bis Bestätigung der Visualisierungseinheit).

Weitere Features, wie Warnung bei Überschreiten einer konfigurierbaren Abgastemperatur, Speichern und Visualisieren der zurückgelegten Distanz und Tageskilometerzähler sind denkbar. Das Projekt wird nach Abschluss der Arbeit weiter entwickelt.

# Abbildungsverzeichnis

[Abbildung 3‑1 - Leistungsdiagramm des Testfahrzeugs dieser Arbeit 5](#_Toc348696620)

[Abbildung 5‑1 - Hall-Effekt, Quelle: Wikipedia [1] 10](#_Toc348696621)

[Abbildung 5‑2 - Blockdiagramm Hall-IC, Quelle: Infineon Datenblatt TLE49x5L [2] 11](#_Toc348696622)

[Abbildung 5‑3 - Hall-IC-Schaltung auf Stator 12](#_Toc348696623)

[Abbildung 5‑4 - Induktionsmessung am Zündkabel 13](#_Toc348696624)

[Abbildung 5‑5 - Lichtmaschinensignal Vespa 14](#_Toc348696625)

[Abbildung 5‑6 - Lichtmaschine mit Diode 14](#_Toc348696626)

[Abbildung 5‑7 - Lichtmaschinensignal mit Diode, Filter und Spannungsbegrenzung 15](#_Toc348696627)

[Abbildung 5‑8 - Tachosensor-Prototyp befestigt mit Kabelbinder. Dauermagnet an Felge 17](#_Toc348696628)

[Abbildung 5‑9 - Thermoelement, Quelle: Wikipedia [3] 18](#_Toc348696629)

[Abbildung 5‑10 - Blockdiagramm MAX6675, Quelle: Maxim-Datenblatt MAX6675 [4] 19](#_Toc348696630)

[Abbildung 5‑11 - Protokoll MAX6675, Quelle: Maxim-Datenblatt MAX6675 [4] 19](#_Toc348696631)

[Abbildung 5‑12 - Datenframe/Bits MAX6675, Quelle: Maxim-Datenblatt MAX6675 [4] 20](#_Toc348696632)

[Abbildung 5‑13 - Ablaufdiagramm Kommunikation mit Max6675 22](#_Toc348696633)

[Abbildung 5‑14 -Mikrocontroller kommuniziert über serielle Kabelverbindung mit Android-Mobiltelefon 23](#_Toc348696634)

[Abbildung 5‑15 - Auszug des Schaltplans für Kommunikation via Can-Bus 24](#_Toc348696635)

[Abbildung 5‑16 - Display eines Prototypen am Vespa-Lenker 25](#_Toc348696636)

[Abbildung 5‑17 - Display mit Störungen 26](#_Toc348696637)

[Abbildung 5‑18 - Schaltplan Prototyp integrierte Mess-, Auswerte- und Visualisierungseinheit 27](#_Toc348696638)

[Abbildung 5‑19 - Prototyp integrierte Mess-, Auswerte- und Visualisierungseinheit 28](#_Toc348696639)

[Abbildung 5‑20 - Prototyp integrierte Mess-, Auswerte- und Visualisierungseinheit 28](#_Toc348696640)

[Abbildung 6‑1 - Systemübersicht 30](#_Toc348696641)

[Abbildung 6‑2 -Blockschaltbild Bluecontroller, Quelle: www.bluecontroller.com 32](#_Toc348696642)

[Abbildung 6‑3 - Bluecontroller, Bordlayout mit Legende, Quelle: www.bluecontroller.com 33](#_Toc348696643)

[Abbildung 6‑4 - Gesamtschaltplan Messsystem 34](#_Toc348696644)

[Abbildung 6‑5 - Bordlayout Messsystem 35](#_Toc348696645)

[Abbildung 6‑6 - Schaltplan Drehzahlerfassung Messsystem 36](#_Toc348696646)

[Abbildung 6‑7 - Schaltplan Beschaltung Max6675 37](#_Toc348696647)

[Abbildung 6‑8 - Schaltplan Beschaltung TLE49x5 37](#_Toc348696648)

[Abbildung 6‑9 - Klassendiagramm Visualisierungseinheit 39](#_Toc348696649)

[Abbildung 6‑10 - Klassendiagramm Android SDK: BluetoothChat 43](#_Toc348696650)

# Quellen- und Literaturverzeichnis

[1] Wikipedia: Hall-Effekt, Artikel, <http://de.wikipedia.org/wiki/Hall-Effekt>

[2] Infineon: TLE49x5 Hall-ICs, Datenblatt, <http://www.infineon.com/dgdl/Data_Sheet_TLE49x5L_Family_V1_5.pdf?folderId=db3a304412b407950112b408e8c90004&fileId=db3a304412b407950112b4090265005b>

[3] Wikipedia: Thermoelement, Artikel, <http://de.wikipedia.org/wiki/Thermoelement>

[4] Maxim: MAX6675 Typ-K Thermoelement ICs, Datenblatt, <http://datasheets.maxim-ic.com/en/ds/MAX6675.pdf>

[5] Wikipedia: Bluetooth-Logo, Grafik, <http://de.wikipedia.org/wiki/Bluetooth>

[6] Wikipedia: Lorentzkraft, Artikel, <http://de.wikipedia.org/wiki/Lorentzkraft>

[7] European Scooter Challenge: Website, <http://www.eurochallenge.de/home.html>

[8] German Scooter Forum: Website, <http://www.germanscooterforum.de/>

[9] Wikipedia: Seebeck-Effekt, Artikel, <http://de.wikipedia.org/wiki/Thermoelektrizit%C3%A4t#Seebeck-Effekt>

[10] Wikipedia: Schmitt-Trigger, Artikel, <http://de.wikipedia.org/wiki/Schmitt-Trigger>

[11] Manfred Albach (2008): Grundlagen der Elektrotechnik I, Buch, 352 Seiten

[12] Prof. Dr.-Ing. Ulrich Teppner (2009): Grundlagen Digitaler Systeme, Skript zur Vorlesung, 88 Seiten

[13] Prof. Dr.-Ing. Ulrich Teppner (2010): Digitaltechnik, Skript zur Vorlesung, 109 Seiten

[14] Prof. Dr.-Ing. R. Wambach: Automatisierungstechnik, Skript zur Vorlesung, 159 Seiten

[15] Mikrocontroller.net: AVR-GCC-Tutorial, Online-Einführung in Atmega-C-Entwicklung,   
http://www.mikrocontroller.net/articles/AVR-GCC-Tutorial