

Martin Pyka

Simon Kesting

Mikrocontrollersysteme WS19/20

Prof. Dr.-Ing. Karl-Ragmar Riemschneider

Fahrzeug-Inforamtions-Display

Analoger Tachometer, Digitaler Tageskilometerzähler, Bewegungsrichtung

Inhaltsverzeichnis

[1. Einleitung und Zielsetzung 2](#_Toc32423185)

[1.1 Zeitplan und Arbeitsaufteilung im Team 3](#_Toc32423186)

[2. Umsetzung 5](#_Toc32423187)

[2.1 grundlegendes Konzept 5](#_Toc32423188)

[2.2 Softwarebeschreibung 6](#_Toc32423189)

[2.3 Aufteilung der Benutzeroberfläche 7](#_Toc32423190)

[Fazit und Ausblick 10](#_Toc32423191)

[Tabellenverzeichnis 11](#_Toc32423192)

[Abbildungsverzeichnis 11](#_Toc32423193)

[Quellenverzeichnis 11](#_Toc32423194)

# 1. Einleitung und Zielsetzung

Im Rahmen der Vorlesung zu Mikrocontroller-Systeme wurde die Projektaufgabe Fahrzeug-Informations-Display bearbeitet.

Die Aufgabe besteht darin, ein analoges Tachometer zur Geschwindigkeitsanzeige, einen digitalen Tageskilometerzähler und die Ausgabe der Bewegungsrichtung zu entwickeln.

Die Entwicklung besteht aus dem Entwurf, der Programmierung und dem Test mithilfe der vorgegebenen Rechnersysteme und Peripherie. [1]

Im Labor wurde das Programm mit der Entwicklungssoftware Code Composer Studio CCS von Texas Instruments in der Programmiersprache C realisiert.

Die Tiva Evaluation Boards mit dem Mikrocontroller TM4C1294 wurden zur Ansteuerung des Touch-Screen-Controllers XPT2046 und einem Wheel Sensor-Modul verwendet. [2]

## 1.1 Zeitplan und Arbeitsaufteilung im Team

Zu Beginn wurde das Projekt zeitlich und inhaltlich geplant. Dazu wurde ein Projektplan mit den einzelnen Arbeitsschritten erstellt.

Zunächst wurde in dem Team festgehalten, wie die Labortermine genutzt werden, um eine grobe Einteilung festzulegen.

Grundlage

* Modulare Aufteilung der Projektaufgaben
* Wiederholte Absprachen vor den einzelnen Terminen

Termin 1

* Rechnersysteme und Peripherie untersuchen
* Wheel Sensor -> Signalanalyse S1 und S2
* Display ansteuern und erste Tests
* Touchfunktion erste Tests
* Konzept Anpassung

Termin 2

* Display: grundlegende Funktionen implementieren, sinnvolle Darstellungen erarbeiten
* Schnittstellen Definition
* Interrupts und deren Handler implementieren und testen
* Buchstaben und Zahlen Bibliothek erstellen

Termin 3

* Wheel Sensor Signale zählen mit Interrupts
* Systick freischalten
* Systick-Handler implementieren
* Geschwindigkeitsberechnung
* Anzeige der der berechneten Werte
* Umstellen per Touch auf digitale Geschwindigkeit

Termin 4

* Systemtest (gesamt)

Grundlegend waren wiederholte Absprachen vor den einzelnen Terminen geplant, um die Labortermine effektiv nutzen zu können.

Der geschätzte Zeitbedarf zu jedem Arbeitsschritt und die Zuordnung zu den Teammitgliedern wurde in einer Excel-Tabelle dokumentiert.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Week |  | 42 | 42 | 43 | 43 | 44 | 44 | 45 | 45 | 46 | 46 | 47 | 47 | 48 | 48 | 49 | 49 |
|  |  | Plan | Real | Plan | Real | Plan | Real | Plan | Real | Plan | Real | Plan | Real | Plan | Real | Plan | Real |
| Concept/Planning | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | Simon | 4 | **4** | 1 | **1** | 1 | **0** | 2 | **4** | 1 | **1** | 1 | **0** | 2 | **1** | 1 | **1** |
|  | Martin | 3 | **3** | 1 | **1** | 1 | **0** | 2 | **5** | 1 | **1** | 1 | **0** | 2 | **1** | 1 | **1** |
| Display |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | Simon | 2 | **2** | 4 | **5** | 2 | **1** |  |  | 2 | **2** |  |  |  |  |  |  |
|  | Martin | 2 | **2** | 3 | **4** | 3 | **2** |  |  | 3 | **2** |  |  |  |  |  |  |
| Wheel Sensor-Modul (Interrupts) | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | Simon |  |  |  |  | 4 | **3** | 2 | **5** | 6 | **4** | 2 | **2** | 4 | **4** |  |  |
|  | Martin |  |  |  |  | 6 | **4** | 2 | **8** | 6 | **4** | 2 | **1** | 5 | **4** |  |  |
| Test/Verification | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | Simon |  |  | 1 | **1** | 1 | **0** | 2 | **4** | 1 | **0** | 0 | **1** | 1 | **1** | 3 | **4** |
|  | Martin |  |  | 1 | **1** | 1 | **0** | 2 | **4** | 1 | **0** | 0 | **1** | 1 | **1** | 3 | **5** |
| Documentation | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | Simon |  |  | 1 | **1** | 1 | **0** | 1 | **2** | 1 | 0 | 1 | **2** | 0 | **1** | 1 | **1** |
|  | Martin |  |  | 1 | **1** | 1 | **0** | 1 | **1** | 1 | 1 | 1 | **2** | 1 | **1** | 1 | **1** |
| Presentation | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | Simon |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | Martin |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Sum | Plan | 11 |  | 13 |  | 21 |  | 14 |  | 23 |  | 8 |  | 16 |  | 10 |  |
| Sum | **Ist** |  | **11** |  | **15** |  | **10** |  | **33** |  | **15** |  | **9** |  | **14** |  | **13** |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Week |  | 50 | 50 | 51 | 51 | 52 | 52 | 1 | 1 | Simon |  | Martin |  | Total |  |
|  |  | Plan | Real | Plan | Real | Plan | Real | Plan | Real | Plan | Real | Plan | Real | Plan | Real |
| Concept/Planning | |  |  |  |  |  |  |  |  |  |  |  |  | 29 | 28 |
|  | Simon | 1 | **0** | 1 | **0** | 0 | **1** |  |  | 15 | **13** |  |  |  |  |
|  | Martin | 1 | **2** | 1 | **0** | 0 | **1** |  |  |  |  | 14 | **15** |  |  |
| Display |  |  |  |  |  |  |  |  |  |  |  |  |  | 21 | 20 |
|  | Simon |  |  |  |  |  |  |  |  | 10 | **10** |  |  |  |  |
|  | Martin |  |  |  |  |  |  |  |  |  |  | 11 | **10** |  |  |
| Wheel Sensor-Modul (Interrupts) | |  |  |  |  |  |  |  |  |  |  |  |  | 39 | 45 |
|  | Simon | 0 | **1** |  |  | 0 | **2** |  |  | 18 | **21** |  |  |  |  |
|  | Martin | 0 | **1** |  |  | 0 | **2** |  |  |  |  | 21 | **24** |  |  |
| Test/Verification | |  |  |  |  |  |  |  |  |  |  |  |  | 23 | 33 |
|  | Simon | 3 | **2** | 0 | **2** |  |  | 0 | **1** | 12 | **16** |  |  |  |  |
|  | Martin | 2 | **1** | 0 | **3** |  |  | 0 | **1** |  |  | 11 | **17** |  |  |
| Documentation | |  |  |  |  |  |  |  |  |  |  |  |  | 30 | 30 |
|  | Simon | 1 | **1** | 3 | **2** | 5 | **6** | 1 | **0** | 16 | **16** |  |  |  |  |
|  | Martin | 0 | **1** | 3 | **2** | 3 | **2** | 1 | **2** |  |  | 14 | **14** |  |  |
| Presentation | |  |  |  |  |  |  |  |  |  |  |  |  | 10 | 12 |
|  | Simon |  |  | 2 | **2** | 2 | **2** | 1 | **2** | 5 | **6** |  |  |  |  |
|  | Martin |  |  | 2 | **2** | 2 | **2** | 1 | **2** |  |  | 5 | **6** |  |  |
| Sum | Plan | 8 |  | 12 |  | 12 |  | 4 |  | 76 |  | 76 |  | 152 |  |
| Sum | **Ist** |  | **9** |  | **13** |  | **18** |  | **8** |  | **82** |  | **86** |  | **168** |

Tabelle 1: Excel-Tabelle mit Zeitplan und Arbeitsaufteilung

# 2. Umsetzung

Die Beschreibung des Konzeptes wird in Abschnitte unterteilt, welche sich an der Aufgabenstellung orientieren.

## 2.1 grundlegendes Konzept

-> Grafiken („fritzing“)

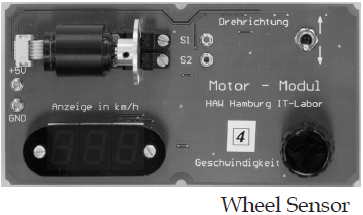
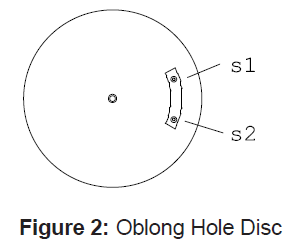


Abbildung 1: Peripheriebaustein - Rad Sensor und Signale s1, s2

[1]

Das „Wheel Sensor“ – Modul enthält einen Motor, welcher eine Scheibe mit zwei Löchern antreibt. Die Drehgeschwindigkeit und Drehrichtung kann verändert werden.

Es werden zwei kurze High-Signale (s1 und s2) bei jeder Umdrehung ausgegeben.

Aus der Reihenfolge der High-Impulse lässt sich die Bewegungsrichtung ermitteln.

Die Signale s1 und s2 müssen logisch erfasst werden.

Mit dem Oszilloskop wurde gemessen, um das Zeitverhalten zu analysieren.

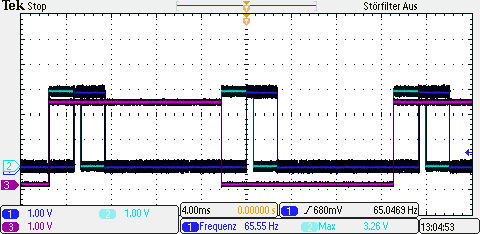


Abbildung 2: Oszilloskop, Messung des Zeitverhaltens der Sensorsignale

Es ist zu erkennen, dass in einem kurzen Moment beide Sensorsignale s1 und s2 auf high sind. Die Drehrichtung entscheidet darüber, welches Signal zuerst auf high geht.

Die Zustände (s1,s2):

(**1**,0) (1,**1**) (**0**,1) -> bei einer Vorwärtsdrehung

(0,**1**) (**1**,1) (1,**0**) -> bei einer Rückwärtsdrehung

sind zu erkennen. (In **Fett** sind die Änderungen markiert)

## 2.2 Softwarebeschreibung

-> modularer Aufbau

-> Klassendiagramme

Variante 1:

Die High-Signale von s1 und s2 werden von der Interrupt Service Routine einzeln gezählt. In der ISR wird die Drehrichtung gesetzt. Durch das einzelne zählen von s1 und s2 kann die Drehrichtung erkannt werden. Wenn der Zähler von s1 größer als der Zähler von s2 ist kann man von einer Vorwärtsdrehung ausgehen. Wenn der Zähler von s1 kleiner als der Zähler von s2 ist kann man von einer Rückwärtsdrehung ausgehen. Wenn beide Zähler gleich groß sind bleibt man in der aktuellen Richtung. Die ISR beider Signale sind in der Lage die Drehrichtung zu verändern. Da pro Umdrehung das erste Signal um einen größer ist als das zweite Signal, setzt das erste Signal die Drehrichtung. Das Zweite erkannte Signal hat den gleichen Wert wie sein Vorgänger und ändert nichts an der Richtung.

Variante 2:

In den einzelnen ISR für s1 und s2 wird geprüft, welchen Zustand der Pin hat der nicht die ISR aktiviert hat. Ist der andere Pin High wird nichts gemacht, ist der andere Pin Low kann die Richtung bestimmt werden.

Da die GPIO-Interrupts nur auf Rising-Egde triggern werden Interrupts nur ausgelöst, wenn aus einer „0“ eine „1“ wird. Für Vorwärts sind das die Zustände (1,0) und (1,1). Für Rückwärts sind das die Zustände (0,1) und (1,1).

Der Zustand (1,1) wird nicht beachtet, da er keine Informationen zur Drehrichtung enthält.

Somit gibt der Interrupt welcher zuerst aktiviert wird, ohne dass der andere Pin High ist, die Drehrichtung an. Wir haben (1,0) als Vorwärts und (0,1) als Rückwärts definiert.

Für die Raddrehzahl Erkennung wird eine globale Variable von der s1- und s2-ISR inkrementiert. Nur wenn kein Richtungswechsel erfolgte wird die globale Variable inkrementiert. Bei einem Richtungswechsel wird bei eins gestartet. Die lese und schreib Zugriffe sind mit je einem Mutex gesichert. Mit einem Timer werden zu fest definierten Zeiten der Zählerwert der Signale abgespeichert. Mit dem festen Zeitabstand und der Anzahl der Umdrehungen kann die Drehzahl/sec berechnet werden.

Drehzahlt/s = (Zählerwert/Timer\_Zeit) \* (Wert, damit aus Timer\_zeit \* Wert = 1s/1000ms)

Darauf ergibt sich dann die Geschwindigkeit in km/h wie folgt:

Geschwindigkeit = Radumfang(1m) \* Drehzahl/s

Der Tageskilometerzähler wird immer um eins inkrementiert wenn ein Meter gefahren wurde. Da eine Umdrehung des Rades einem Meter entspricht wird der Zähler bei einer Umdrehung inkrementiert.

## 2.3 Aufteilung der Benutzeroberfläche

Der „Start Up“- Bildschirm wird direkt nach der Initialisierung in das Grafik-Display geladen.

#### Start Up

Abbildung 3: Display - Start Up

Die Balkenfarbe und die Kasten Farbe für die Fahrtrichtung passen sich an die Fahrtrichtung an: Vorwärts (grün), Rückwärts (gelb).

#### Vorwärts

V

123,47km

Abbildung 4: Display - Vorwärts

Durch Berührung lässt sich die Tageskilometerzähler-Anzeige auf die digitale Geschwindigkeits-Anzeige ändern.

#### Rückwärts

128,31 km/h

R

Abbildung 5: Display - Rückwärts

Es ist die Frage zu beantworten, ob es sinnvoll sei das Fahrzeug-Informations-Display während der Testphase auf dem PC-Bildschirm auszugeben.

Es wäre sinnvoll für Arbeiten außerhalb des Labors, allerdings könnte sich das Display in dem Labor anders verhalten. Das Ziel ist es, das System vollständig im Labor zu testen. Mit vorhandener Hardware wäre die Darstellung auf dem PC nicht nötig.

Es wird ein Interrupt-Handler benötigt. Die Analyse der Wheel-Sensor Signale S1 und S2 müssen im Zweiten Termin mithilfe von Interrupts und Timern analysiert werden. Interrupts sind sinnvoll, da die Signale der Drehzahlsensoren zeitlich stark variieren. Die Polling variante führt dazu, dass Signale möglicherweise nicht erkannt werden. Da aktiv auf die Signale gewartet werden muss, kann man keine anderen Prozesse ausführen.

Die Interrupts werden von den Sensoren über die Pins direkt getriggert.

Die Interrupt Handler zählen die Umdrehungsanzahl.

Grundlegende Aufgaben in Bezug auf die Realisierung sind eine übersichtliche, hierarchische Codierung und die Dokumentation der Ergebnisse. Mithilfe von Git findet eine Versionierung statt.

Zur Realisierung des Displays ergeben sich folgende Aufgaben:

* Bibliothek für „Buchstaben und Zahlen zeichnen“
* Segmentdarstellung
* Wechsel von Tageskilometer zu Digitale Geschwindigkeit durch Touch
* Schnittstelle zur Verfügung stellen
* Start Up Bild einmalig starten und einzelne Segmente überschreiben
* Nur das zu ändernde Feld löschen und neu beschreiben

Zur Realisierung der Drehzahlmessung ergeben sich folgende Aufgaben:

* Drehrichtung erkennen (s1 u s2)
* Pins Initialisieren (AF)
* Interrupts initialisieren (Pins)
* I-Handler schreiben
* Zählen (Overflow abfangen)
* Drehrichtung Setzen (Vergleich der Zähler)
* Definition von Vorwärts (V) und Rückwärts (R)
* Drehzahl berechnen (m/s)

*Uint32\_t detect\_frequency uint32\_t gcounter)*

* Timer initialisieren (one shot, ISR, highest prio)
* Zählerwert von s1/s2 holen nach bestimmter Zeitdauer
* Berechnung der Drehzahl anhand von Umdrehungen pro fester Timerzeit
* Zähler der s1 und s2 Signale zurücksetzten
* Timer neustarten
* Drehgeschwindigkeit berechnen (s1/s2)

*unsigned int detect\_speed (unsigned int s1/s2)*

* Berechnung der Geschwindigkeit anhand von Umdrehung/s und Umfang des Rads.
* Testen von Interrupt und Display Kommunikation (zeitliche Abstimmung)

# Fazit und Ausblick

Das Projekt hat durch eine freiere Aufgabenstellung wesentlich mehr Eigenständigkeit abverlangt als vergleichbare Praktika. Die Vorgabe von Arbeitsphasen, welche sich an der Planung und Konzeptionierung orientieren, haben geholfen den Einstieg in das Projekt zu finden. Im Laufe der Programmierung konnten die Planungen und Überlegungen im Konzept zur Hilfe genommen werden.

Einige Programmabschnitte mussten durch neue Erkenntnisse verändert werden.

Banale Aufgaben wie das Erstellen einer eigenen Buchstaben und Zahlen Bibliothek haben mehr Zeit in Anspruch genommen als erwartet, führten jedoch zu einem ansehnlichen Ergebnis. Die Funktionen der StellarisWare Driver Library waren trotz viel Dokumentation schwer in Betrieb zu nehmen und haben länger gedauert zu implementieren als die Register direkt zu beschreiben.

Das Programm erfüllt die Mindestanforderungen, mit der zusätzlichen Funktion, dass durch Berühren des Displays zwischen dem Tageskilometerzähler und einer digitalen Geschwindigkeitsanzeige gewechselt werden kann.

# Tabellenverzeichnis

[Tabelle 1: Excel-Tabelle mit Zeitplan und Arbeitsaufteilung 4](#_Toc28953896)

# Abbildungsverzeichnis

[Abbildung 1: Peripheriebaustein - Rad Sensor und Signale s1, s2 5](#_Toc29219044)

[Abbildung 2: Oszilloskop, Messung des Zeitverhaltens der Sensorsignale 5](#_Toc29219045)

[Abbildung 3: Display - Start Up 7](#_Toc29219046)

[Abbildung 4: Display - Vorwärts 7](#_Toc29219047)

[Abbildung 5: Display - Rückwärts 7](#_Toc29219048)

# Quellenverzeichnis

[1] Projektanleitung „Fahrzeug-Informations-Display“, HAW, Fakultät Technik und Informatik

[2] XPT2046 Touch Screen Controller, Data Sheet, XPTEK, 2007.5

[3] TivaWare Peripheral Driver Library, Users Guide, SPMU298D, März 2013 – Revised July 2016