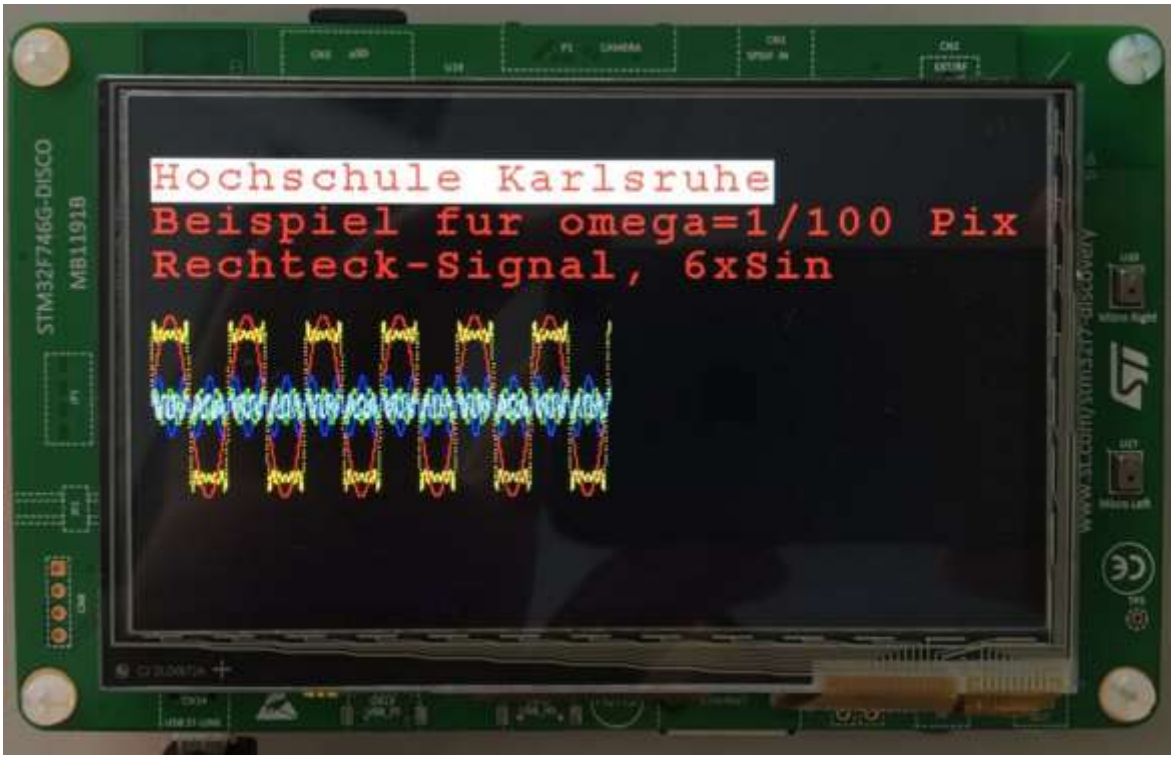
Hochschule Karlsruhe
Technik und Wirtschaft
UNIVERSITY OF APPLIED SCIENCES

Signalverarbeitung mit Lautsprecher

Gruppe 12



Informationstechnik Labor Sommersemester 2016

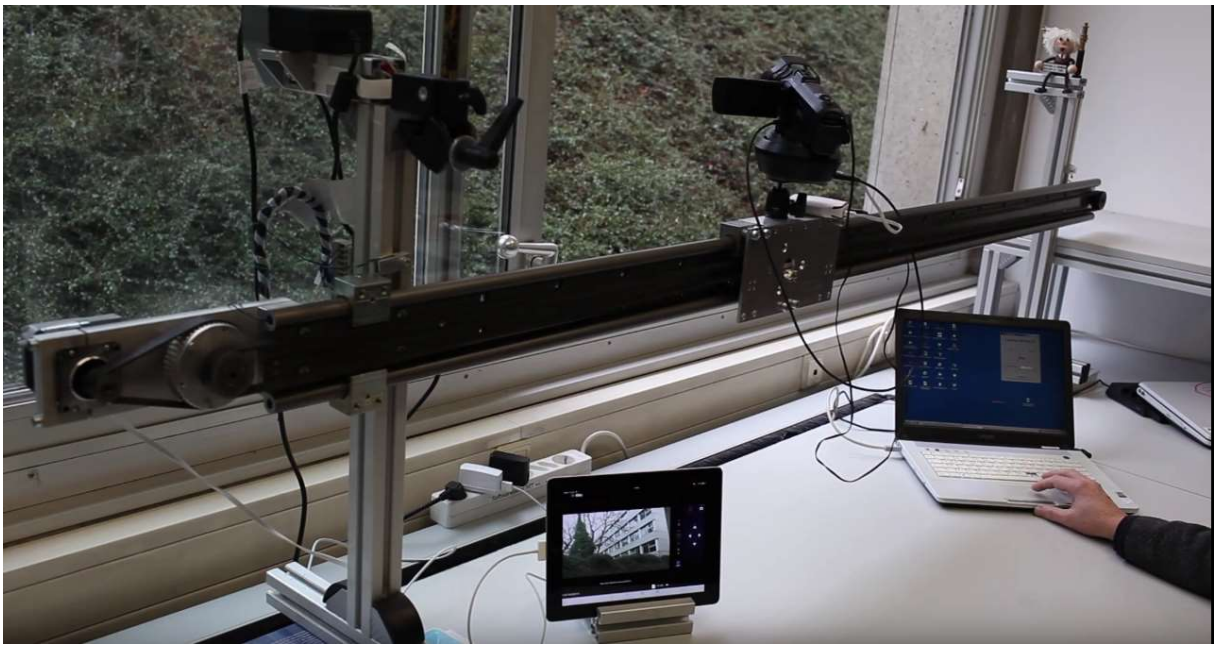
Prof. J. Walter

Gruppenmitglieder:
Nicola Gürpinar *43962*
Ralf Markanycz *40190*

Inhalt

Aufgabenstellung

Für eine bewegliche Kamerafahrt wie in Hollywoodfilmen, besitzt Herr Prof. Walter eine ca. zwei Meter lange Kameraschiene. Auf dieser Schiene wird ein beweglicher Schlitten mittels eines Elektromotors und einem Zahnriemen horizontal bewegt. Eine Kamera, die auf dem Schlitten montiert ist, kann während der „Fahrt“ Videos aufnehmen.



Die Aufgabe unserer Projektgruppe besteht nun darin, eine akustische Widergabe der Position, insbesondere der End- und Startposition (Links- und Rechtsanschlag der Schiene), sowie die Fahrtrichtung des Schlittens, dem Benutzer widerzugeben. Diese Informationswidergabe soll mit einem an der Kameraschiene befestigtem Lautsprecher erfolgen.

Stand der Technik

Die Steuerung der Kameraschiene soll über einen Mikrocontroller, der auf dem Namen STM32F7 von STMicroelectronics hört, erfolgen. Die technischen Daten sind in folgender Tabelle gelistet:

- STM32F746NGH6 microcontroller featuring 1 Mbytes of Flash memory and 340 Kbytes of RAM, in BGA216 package
- On-board ST-LINK/V2-1 supporting USB re-enumeration capability
- Mbed-enabled (mbed.org)
- USB functions: virtual COM port, mass storage, debug port
- 4.3-inch 480x272 color LCD-TFT with capacitive touch screen
- Camera connector
- SAI audio codec
- Audio line in and line out jack
- Stereo speaker outputs
- Two ST MEMS microphones
- SPDIF RCA input connector
- Two pushbuttons (user and reset)
- 128-Mbit Quad-SPI Flash memory
- 128-Mbit SDRAM (64 Mbits accessible)
- Connector for microSD card
- RF-EEPROM daughterboard connector
- USB OTG HS with Micro-AB connectors
- USB OTG FS with Micro-AB connectors
- Ethernet connector compliant with IEEE-802.3-2002
- Five power supply options:
  - ST LINK/V2-1
  - USB FS connector
  - USB HS connector
  - VIN from Arduino connector
  - External 5 V from connector
- Power supply output for external applications: 3.3 V or 5 V
- Arduino Uno V3 connectors
- Comprehensive free software including a variety of examples, part of STM32Cube package
- Supported by a wide choice of integrated development environments

Das STMicroelectronics 32F746GDISCOVERY Discovery Kit erleichtert das Auffinden auf dem ARM Cortex-M7-Kern basierender Mikrocontroller der Serie STM32F7. Es ermöglicht die Entwicklung und Weitergabe von Anwendungen und kann **Audio-/Video-Wiedergabegeräte**, Audiorekorder und Einbruchmeldeanlagen unterstützen. Außerdem können HMIs konzipiert werden, die Audio-, Video- und Farbtouchscreen-Funktionen nutzen.

Problemstellung

Die von der Hochschule Karlsruhe bisher verwendete Kameraschiene, besitzt weder eine optische, noch eine akustische Datenwidergabe der Position, Geschwindigkeit, Fahrtrichtung oder der Inbetriebnahme der Kamera. Der Anwender weiß also zu keinem Zeitpunkt, wie die aktuelle bzw. genaue Lage der auf dem Schlitten installierten Kamera ist.
In der Praxis kann eine solche fehlende akustische Sprach- oder Tonausgabe zu schwerwiegenden Folgen führen. Wenn sich beispielsweise in einer Produktionsstätte große oder schwere Gegenstände bewegen, in denen sich auch Menschen aufhalten, kann eine fehlende Signalwidergabe, egal ob akustisch oder visuell, schnell zu gefährlichen Situationen oder Arbeitsunfällen führen.

Abbildung 1 Ausschnitt aus: Staplerfahrer Klaus

Bei unserer Kamerafahrt wird es wohl nicht zu solch gefährlichen Momenten kommen. Jedoch ist dieses Projekt ein sehr schönes praktisches Beispiel, das auch auf unzähligen anderen mechatronischen Bereichen übertragen werden kann.

Hochschule Karlsruhe
Technik und Wirtschaft
UNIVERSITY OF APPLIED SCIENCES

Anforderungsliste

Informationstechnik Labor Fakultät MMT Hochschule Karlsruhe Technik und Wirtschaft	**Anforderungsliste** Für die **Signalverarbeitung mit Lautsprecher**	**MTB732** Sommersemester 2016
Art	**Anforderung**	**Wert/Daten**
Datenübertragung		
J/N	Datenübertragung erfolgt vom Drehgeber zu dem Mikrocontroller	
J	Datenübertragung erfolgt vom Mikrocontroller zu dem Lautsprecher	
Elektronik		
J	Mikrocontroller STM32F7	
J/N	Lautsprecher (Mit Line-In Eingang)	
N	USB Massenspeicher	Mindestens 10Mb groß
Mechanik		
J	Mechanische Verbindung des Mikrocontrollers an die Kameraschine	
J/W	Anschluss des Lautsprechers an den Mikrocontroller über das Line-Out Terminal	
Software		
N	Keil µVision 5 (C++)	
Restriktionen		
F	Vollautomatische Lautsprecherausgabe	
F	Projektabgabe	06.05.2016

Datum: 22.04.2016

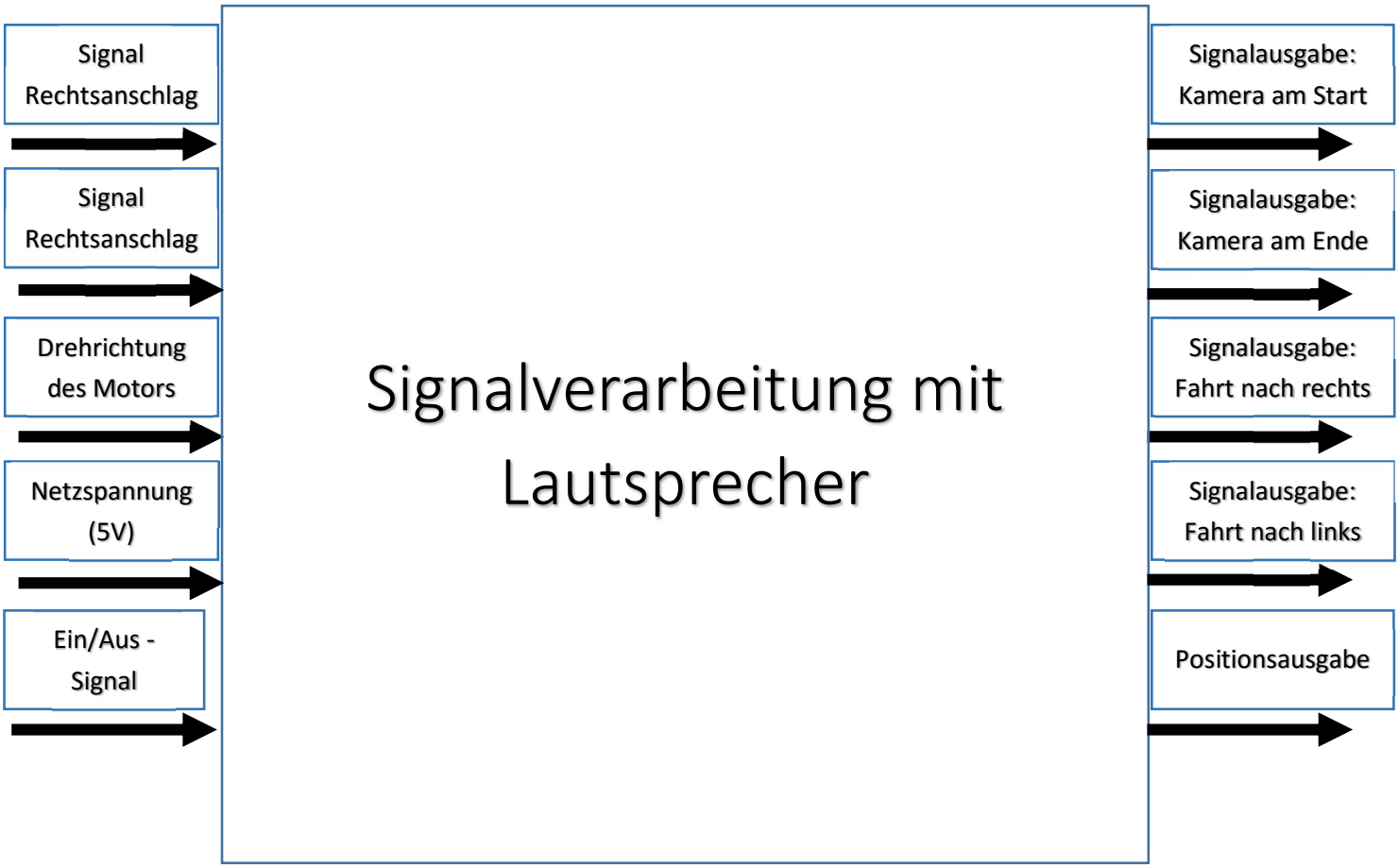
Unterschrift des Dozenten: 

Blackbox

Die Blackbox bezeichnet ein Teil eines Funktionssystems, von welchem nur die im Zusammenhang äußeren stehenden Verhalten betrachtet werden. Die innere Struktur bleibt dabei völlig unbekannt. Es sind nur die am Ausgang austretenden Reaktionen auf bekannte Eingangssignale sichtbar.
Somit kann eine Blackbox hervorragend verwendet werden, um auch komplizierte Systeme oder Projekte mit ihren Ein- und Ausgangsgrößen auf einen Blick überschaubar darzustellen.

Die Blackbox der „Signalverarbeitung mit Lautsprecher“ enthält diverse Eingangsgrößen die zum Verarbeiten der Positionsbestimmung und Fahrtrichtung der Kamera notwendig sind. Ferner sind physikalische und technische Störgrößen sowie vorgegebene Restriktionen als Eingangsgrößen verknüpft.
Ausgangsgrößen beinhalten benutzte, fertige oder bearbeitet Größen, die alle innerhalb der Blackbox verarbeitet und ausgegeben werden.

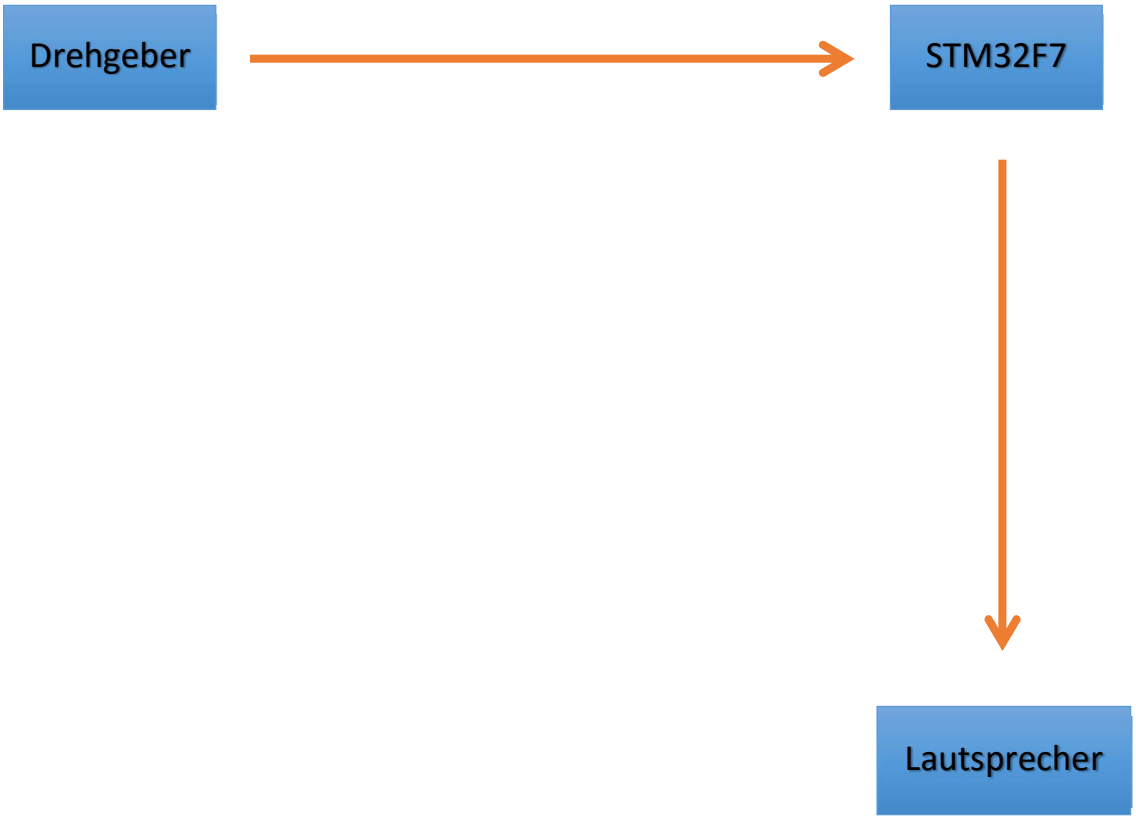
- Umwelteinflüsse (Temperatur, Luftfeuchtigkeit)
- Magnetische und elektrische Störfelder (EMV)
- Mechanische Stöße



- Vollautomatische Lautsprecherausgabe
- Verwendung des Mikrocontrollers STM32F7

Blockschaltbild

Das Blockschaltbild der Signalverarbeitung mit Lautsprecher zeigt uns, woher die Informationen oder Daten kommen und an welches Bauteil sie weitergereicht und verarbeitet werden.
Demnach gibt uns der Drehgeber mit Hilfe seiner Sensoren vor, an Welcher Stelle sich die Kamera aktuell befindet, ob die Kameraschine am linken oder rechten Anschlag anstößt oder ob sie sich nach links oder nach rechts bewegt.
Die vom Drehgeber ausgesandten Daten werden vom Mikrocontroller STM32F7 ausgewertet und anschließen an einem an dem Board angeschlossenen Lautsprecher weitergeleitet. Dieser Lautsprecher gibt nun die tatsächliche Lage der Kameraschine akustisch wieder.

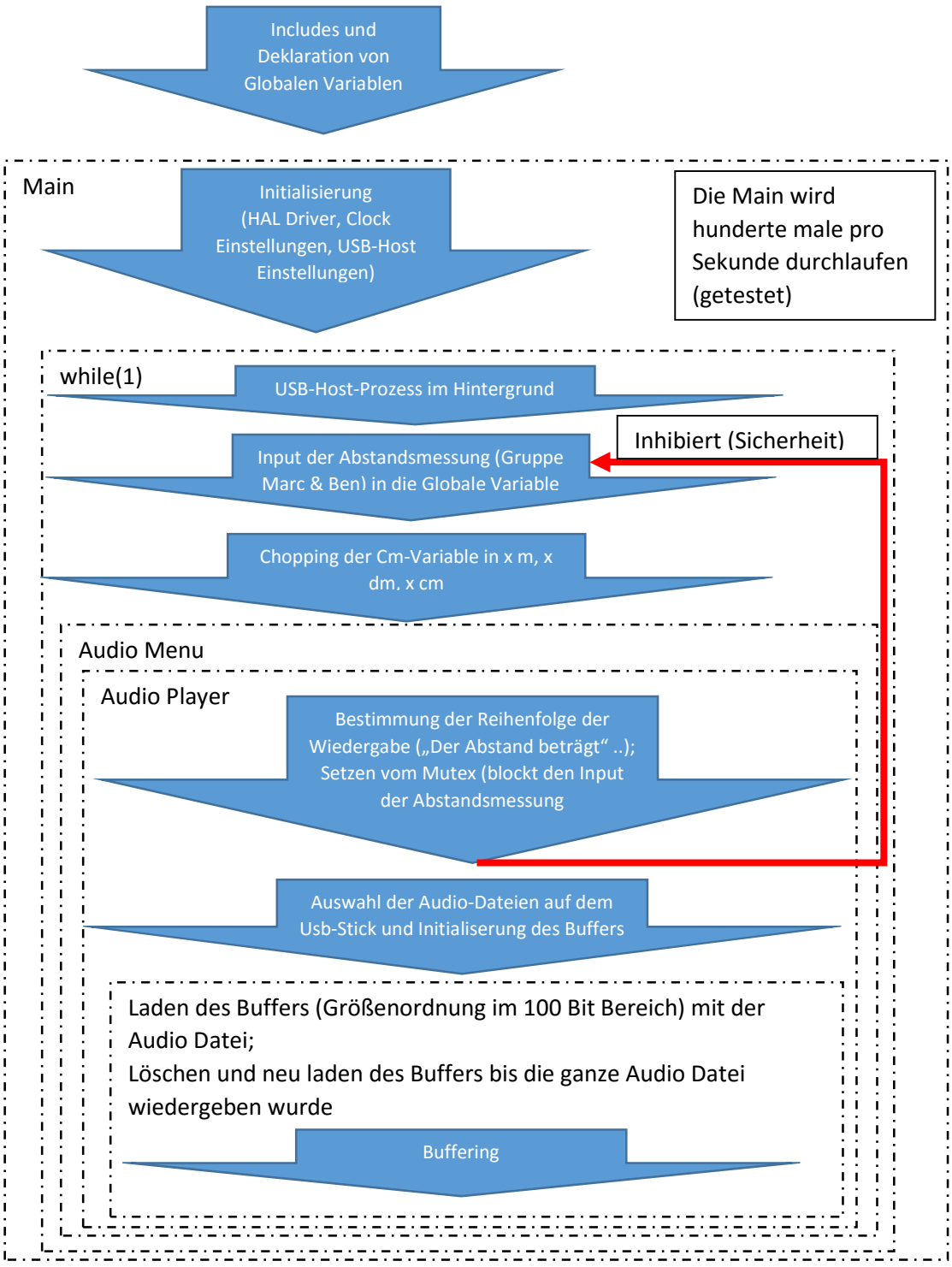


Zeitplan

Nicola Gürpinar **Ralf Markanycz**		**Zeitplan für die Signalbearbeitung mit Lautsprecher**								**Hochschule Karlsruhe - Technik und Wirtschaft** **21.03.2016**
Nr.	**Aktivitäten**	2016								**Jahr**
		März			April				Mai	**Monat**
		11	12	13	14	15	16	17	18	**Kalenderwoche**
1	Projektdefinition erstellen	X								
2	Einarbeitung in den Mikrocontroller		x	X						
3	Absprache mit den anderen Projektgruppen			X						
4	Erstellung der ersten Programme/Codes			x	x	X				
5	Beschaffung der Vollversion von Keil µVision					X				1
6	Testphase und erneute Absprache mit anderen Projektgruppen					X	x			
7	Fertigstellung unserer Aufgabenstellung							X		2
8	Abschlusspräsentation								X	P
		Meilensteine: 1 2								Präsentation: P

Code und Snippets

Im folgenden Abschnitt wird erläutert, wie unser Programm, basierend auf der Beispielanwendung des WAV-Players mit USB-Hosting, funktioniert. Im Wesentlichen wird aus dem vorhandenen Code der Anwendung (verfügbar in der Zip-Datei der STM32F7 Cube Software) der Player behalten. Der ursprüngliche Code ermöglicht das Abspielen der Audio-Dateien auf dem USB-Stick mit Buttons (Play, Pause, Next, Volume) und Aufnehmen von Audio-Dateien durch die Mikrofone. Für unsere Zwecke, sprich der gezielten Wiedergabe von Dateien in Abhängigkeit der Position der Kamera auf der Schiene, mussten bis auf das Laden der Dateien in den Buffer, weitreichende Veränderungen des Codes vorgenommen werden.



menu.c | waveplayer.c | main.c*

```
165   while (1)
166   {
167     a++;
168     /* USB Host Background task */
169     USBH_Process(&hUSBHost);
170
171
172     // Die Magie beginnt hier :) In die funktion "in cm hier rein" soll das Input der Gruppe Mozi-Ben kommen
173     // (in Zentimeter) :P
174
175
176     if(mutex)
177     {
178       incmhierrein(50*sin(a)+100);
179     }
180
181     // tickstart=HAL_GetTick();
182
183     // HAL_Delay(10);
184
185     // Mit HAL_Delay(50)  könnte man testen ob das Programmm in die Main zurükkehrt, und ja, tut es !
186     // Wenn ein audio file abspielt, ist dieses durchgeschnitten durch den Delay, hört sich auch ekelhaft an
187
188     /* AUDIO Menu Process */
189     AUDIO_MenuProcess();
190
191     BSP_LED_Toggle(LED1);
192     previousabstand=aktuellerabstand;
193     //aktuellerabstand++; // Die Main wird pro Sekunde hunderte/tausende Male durchlaufen
194     // Daher schwierig das input so zu simulieren
195
196
197   }
```

Abbildung 2: while(1)-Schleife in der Main

```
138   if(AUDIO_GetWavObjectNumber() > m)
139   { // zvar=aktuellerabstand;
140
141     if(aktuellerabstand <2 && (counter==0) && mutex)
142     {GetFileInfo(15, &WaveFormat); counter =6; booli=1;}
143
144     if(aktuellerabstand >= abstandmax && (counter==0)&& mutex)
145     {GetFileInfo(14, &WaveFormat); counter =6; booli=1;}
146
147     if(counter==0 && !bool)    // Counter gleich 0 heisst erstes mal diese funktion ausführen, sprich erste Audiofile
148     {mutex=0; GetFileInfo(8, &WaveFormat);} // Die Nummer idx=8 ist die Datei "Der Abstand beträgt"
149
150     if(counter==1 && !bool)     // Die Meterzahl, bei der Ansage "Der Abstand beträgt: 1,28 Meter" bspw. die 1 !
151     {GetFileInfo(m, &WaveFormat);}
152     // GetFileInfo(idx, &WaveFormat);
153
154     if(counter==2 &&  !bool)    // Als drittes file kommt "komma", die Kaschie ist ja schliesslich nur 2 Meter lang,
155     // sonst bräuchten wir ja 2 Audiodateien für die (vor)kommazahlen
156     {GetFileInfo(10, &WaveFormat);}
157
158     if(counter==3 &&  !bool)    // Hier die dezimeter (bspw. bei 1,28 Meter die 2) rein
159     {GetFileInfo(dez, &WaveFormat);}
160
161     if(counter==4 &&  !bool)    // Hier die cm rein
162     {GetFileInfo(cm, &WaveFormat);}
163
164     if(counter==5 && !booli)    // Als letztes "Meter"
165     {GetFileInfo(11, &WaveFormat); mutex=1; } // Erlaubnis für die Anschlag-Benachrichtigung
166
167
168     // Ohne Erlaubnis kommt sonst mitten im Satz "Der Abstand beträgt ..." das Anschlag-Audiofile
```

Abbildung 3: Abschnitt des Wave-Players

```
160       {
161         AudioDemo.state = AUDIO_DEMO_WAIT;
162       }
163       break;
164
165
166     case AUDIO_DEMO_PLAYBACK:
167       if(appli_state == APPLICATION_READY)
168       { BSP_TS_GetState(&TS_State);
169         ////////////////////////////////////////////////////////////////////
170       if(TS_State.touchDetected == 1)
171       {
172         if ((TS_State.touchX[0] > 90) && (TS_State.touchX[0] < 380) &&
173             (TS_State.touchY[0] > 212) && (TS_State.touchY[0] < 252))
174         {
175           secretvar=1;
176         }
177       }
178         ////////////////////////////////////////////////////////////////////
179         if(AudioState == AUDIO_STATE_IDLE)
180         {
181           /* Start Playing */
182           AudioState = AUDIO_STATE_INIT;
183
184           /* Clear the LCD */
185           LCD_ClearTextZone();
```

Abbildung 4: Abschnitt des Menus; Im Case AUDIO_DEMO_PLAYBACK wird anschliessend in den Wave-Player gesprungen; Definition eines geheimen Button auf dem "HsKa"-Feld auf dem Display

Fazit

Zu Beginn des Projektes „Signalverarbeitung mit Lautsprecher“ bestand für unsere Projektgruppe die Herausforderung, sich in den Mikrocontrollers STM32F7 einzuarbeiten.
Ziel war es, eine akustische Widergabe der Position, sowie die End- und Startposition (Links- und Rechtsanschlag der Schiene) und die Fahrtrichtung des Schlittens, dem Benutzer widerzugeben.
Aufgrund unserer exzellenten Vorkenntnisse in der C++ Programmierung, die wir in der Hochschule Karlsruhe vermittelt bekommen haben, machten wir uns an die Arbeit, von Grund auf ein eigenes Programm auf dem Mikrocontroller zu schreiben, um die bevorstehende Aufgabenstellung zu meistern. Jedoch kamen wir schnell an einem Punkt, an dem wir feststellten, dass uns die kurze Zeit, (ein knappes halbes Semester) die uns zur Verfügung stand, nicht ausreichen würde. Somit suchten wir nach einer Alternative. Diese Alternative war, ein bereits bestehendes Programm, das für unsere Problemstellung hilfreich sein kann, zu verwenden und zu modifizieren. Ein Beispielprogramm eines WAV-Players mit USB-Hosting, war demnach das Grundgerüst auf dem wir aufgebaut haben (s. Quellcode).
Nach einigen, sehr langen Nächten, die von schier unendlichen Fehlermeldungen des Compilers begleitet wurden und nur mit Fertigessen und reichhaltigen Koffeingetränken zu überleben waren, hatten wir es endlich geschafft☺. Das Programm funktionierte wie es sollte.
Mit einem auf dem Mikrocontroller programmierten Zufallsgenerator, konnten wir aus unserem, an das Board angeschlossenem Lautsprecher, die verschiedensten Positionsangaben widergeben. Auch eine manuelle Eingabe der End- oder Startposition, wurde von einer gebrochenen, leicht erotisch wirkenden Frauenstimme angesagt☺.
Der weitere Ablauf bestünde nun darin, die von der Gruppe „Positionsbestimmung mittels Drehgeber“ aufgenommenen Informationen, in unser Programm zu inkludieren bzw. einzubinden. Danach sollte es möglich sein, einen kleinen voll funktionstüchtigen und automatischen Computer zu erhalten, der die Positionsbestimmung der Kameraschine akustisch ausgibt.

Somit bedanken wir uns bei Herr Prof. Walter für die Betreuung und engagierte Hilfestellung dieses Projektes, sowie die von ihm praktisch gestaltet Lehrveranstaltung, die uns beiden unheimlich viel Spaß gemacht hat.

Nicola Gürpinar 43962

Ralf Markanycz 40190