

Berner Fachhochschule

Technik und Informatik



IPA 2010

Wireless LAN Webradio

Autor	Kevin Gerber
Kandidaten Nummer	5236
Datum	23. April 2010
Dokument Version	0.1

Autor: Kevin Gerber
Berner Fachhochschule für Technik und Informatik
Jlcoweg 1, CH-3400 Burgdorf
+41 (34) 426 68 66 / kevin.gerber@bfh.ch

Fachvorgesetzter: Martin Aebersold
Berner Fachhochschule für Technik und Informatik
Jlcoweg 1, CH-3400 Burgdorf
+41 (34) 426 68 97 / martin.aebersold@bfh.ch

Experten: Martin Gerber
Regionalverkehr Bern-Solothurn RBS
+41 (31) 925 56 26 / martin.gerber@rbs.ch

Wolfgang Schnabel
+41 (31) 932 21 44 / wolfgang.schnabel@gmx.ch

Inhaltsverzeichnis

1	Kurzzusammenfassung	1
2	Einleitung	2
2.1	Aufgabenstellung	2
2.2	Vorgehensweise	3
2.3	Zielsetzung	3
3	Verwendete Geräte & Software	4
Hardware		5
4	Beschreibung	5
5	Blockschaltbild	7
6	Pin Definitionen	8
6.1	WLAN Modul	8
6.2	OLED Display	8
6.3	Touchscreen Controller	8
6.4	Audiodecoder	8
6.5	Akku Überwachung	9
6.6	Standard Ein- und Ausgänge	9
Analyse / Realisierung		10
7	Server Kommunikation	10
7.1	WLAN	10
7.2	Kommunikationsablauf	11
8	Streaming	12
8.1	SHOUTcast Protokoll	12
8.2	Audiodecoder	14
9	Zeitlicher Ablauf	16
9.1	Berechnung Baudrate	16
9.2	Berechnung SPI Tackt	16
10	Betriebsdauer	18
11	Menüführung	19
11.1	Übersicht	19
Software		20
12	Grober Programmablauf	20
13	Buffer Konzept	21
14	Betriebssystem	22
15	Tasks	23
15.1	Synchronisation & Datenaustausch	24

15.2	Webradio Task.....	25
15.3	GUI Task.....	27
15.4	Streaming Task.....	28
15.5	Playing Task.....	29
Schlusswort		30
16	Stand der Arbeit	30
16.1	Testprotokoll	31
17	Probleme / Schwierigkeiten.....	32
17.1	Ad hoc Modus	32
17.2	Flow Control.....	32
17.3	Socket schliessen	32
18	Mögliche Erweiterungen.....	34
19	Erfahrungen	35
20	Dank.....	35
21	Bedienungsanleitung.....	35
Informationen		36
22	Abkürzungen.....	36
23	Glossar	37
24	Quellen.....	39
24.1	Internet	39
Anhang		40

Abbildungsverzeichnis

Abb. 1 Systemschaltbild	2
Abb. 2 Blockschaltbild	7
Abb. 3 TCP Kommunikationsablauf.....	11
Abb. 4 SHOUTcast Protokoll	13
Abb. 5 Audiodecoder Interfaces	14
Abb. 6 Menüstruktur	19
Abb. 7 Grober Programmablauf	20
Abb. 8 Buffer Konzept	21
Abb. 9 Task Synchronisation	24
Abb. 10 Webradio State Event Diagram	25
Abb. 11 Webradio Task Struktogramm	26
Abb. 12 Playing Task Struktogramm	29

1 Kurzzusammenfassung

Im praktischen Teil meiner Lehrabschlussprüfung entwickelte ich die Software, zu einer bereits existierenden Hardware, für einen Wireless LAN Webradio. Mit dem WLAN Webradio kann in ein Netzwerk verbunden werden und einen Audiostream bei einem SHOUTcast Server angefordert werden. Die META-Daten werden gefiltert und die Audiodaten mit dem Audiodecoder ausgegeben.

Die aktuelle Version 0.1 weist momentan noch Mängel auf:

- Die Wiedergabe stockt
- Socket wird nicht richtig geschlossen

2 Einleitung

Für den praktischen Teil meiner Lehrabschlussprüfung entschlossen mein Lehrmeister und Betreuer, Martin Aebersold, und ich einen Webradio mit Wireless LAN (WLAN) zu realisieren. Dazu wurde als Vorarbeit ein WLAN Terminal entwickelt. Das WLAN Terminal besitzt unter anderem ein 2.8" OLED Display mit Touchscreen, ein WLAN Modul, einen Audiodecoder und eine MicroSD Karte.

2.1 Aufgabenstellung

Die Arbeit besteht darin, die Software für einen Webradio mit einer grafischen Bedienoberfläche mit Touchscreen zu realisieren. Der Audio stream eines SHOUTcast-Servers soll über das WLAN Module empfangen und die MP3 Daten an den Audiodecoder gesendet werden.

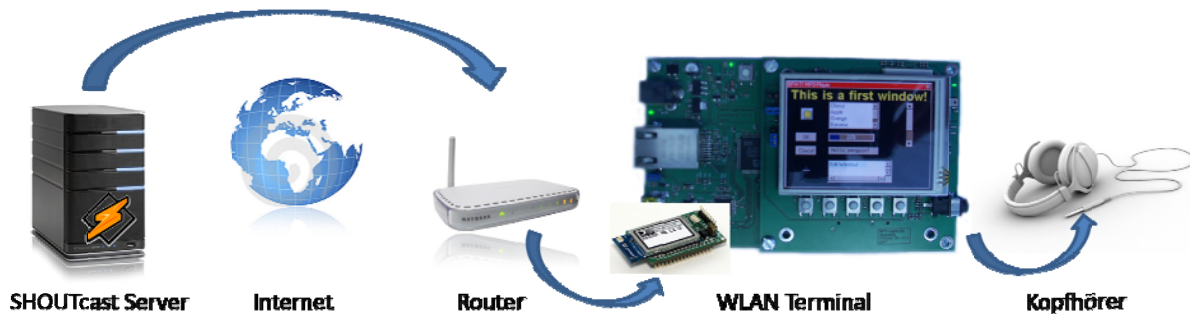


Abb. 1 Systemschaltbild

2.1.1 Detailliertere Arbeiten

- Grafische Bedienoberfläche mit Touchscreen
- Auswahl zwischen mehreren Radiostationen
- WLAN Verbindungsaufbau in einem Netzwerk mit Sockets
- Audio Stream empfangen und META-Daten herausfiltern
- Ausgabe der Audiodaten mit dem Audiodecoder
- Songinformationen auf dem Display anzeigen

2.2 Vorgehensweise

Das Projekt wird in zwei Phasen aufgeteilt. In die Vorbereitungsphase und in die eigentliche IPA.

2.2.1 Vorbereitungen

Die komplette Hardware wurde in der Vorbereitungsphase realisiert. Schemas, sowie das Layout habe ich selber gemacht. Ausserdem wurde der Prototyp bereits in Betrieb genommen und die Peripherie konnte in der Vorbereitungsphase getestet werden. Es wurde bereits eine Bibliothek für die Kommunikation mit dem WLAN Module, dem Audiodecoder und dem Touchscreencontroller geschrieben. Auf dem Mikrocontroller wird ein OpenSource Betriebssystem (TNkernel) mit einer integrierten Grafikbibliothek (Microwindows) eingesetzt. Leider reichte die Zeit nicht aus, um mich vor Beginn der IPA in das Betriebssystem einzuarbeiten. Mit der Grafikbibliothek konnte ich bereits ein paar kleine Versuche machen, doch es handelt sich bei Microwindows um eine riesige Grafikbibliothek mit unzähligen Möglichkeiten. In meinen Versuchen konnte ich nur das Erstellen von Buttons und das Ausgeben von Texten testen.

2.2.2 Individuelle Produktivarbeit (IPA)

In einer zweiten Phase wird die Software für den WLAN Webradio nach Aufgabenstellung geschrieben und das Projekt dokumentiert. Dies ist die eigentliche IPA.

2.3 Zielsetzung

Der WLAN Webradio soll über das Touchscreen mit dem Finger bedient werden können. Ausserdem soll er folgende Funktionen unterstützen:

- Nach verfügbaren Netzwerken suchen
- WLAN Verbindung mit einem Netzwerk herstellen
- WLAN Verschlüsselung Open und WEP-128 unterstützen
- Auswahl aus mehreren Radiostationen durch den Benutzer
- Audiostream empfangen
- META-Daten herausfiltern und auswerten
- Soundinformationen auf dem Display darstellen (Sender, Interpret, Titel, Genre)
- Ausgabe der Audiodaten mit dem Audiodecoder
- Audioausgabe pausieren oder beenden
- Lautstärkeneinstellung

2.3.1 Optionale Erweiterungen

Diese optionalen Erweiterungen müssen nicht unbedingt in der IPA realisiert werden. Sie werden nur gemacht, wenn dafür noch genügend Zeit zur Verfügung steht.

- Unterstützung von weiteren WLAN Verschlüsselungen (WPA-PSK, WPA2-PSK)
- Klangregelung (Bass und Treble)
- Softwaremässige Eingabemöglichkeit des WLAN Keys
- Radiostationen durch Eingabe des Benutzers hinzufügen

3 Verwendete Geräte & Software

- OpenOCD USB-JTAG-Debugger SAL1
- Power Supply HM8142 MN 001-4
- Oscilloskope Agilent 54622D MK-211-1
- Multimeter FLUKE 87 True RMS MM 108-11

- Desktopcomputer mit Windows XP
- OpenSource Entwicklungsumgebung Eclipse
- GNU-ARM Toolchain
- Microsoft Office Word 2007
- Microsoft Office Excel 2007
- Microsoft Office Visio 2007
- Adobe Photoshop CS3
- HUS Struktogrammer
- Doxygen

Hardware

4 Beschreibung

Die Hardware Namens WLAN Terminal wurde wie bereits erwähnt, vorgängig realisiert und in Betrieb genommen. Das Herzstück des WLAN Terminals ist ein 32 Bit Mikrokontroller der Marke NXP mit einem Cortex-M3 Kern. Seine genaue Bezeichnung lautet LPC1768FBD100. Auf dem WLAN Terminal befindet sich ausserdem folgende Peripherien:

- OLED Display mit Touchscreen

Das OLED Display hat eine Diagonale von 2.8 Zoll und eine Auflösung von 240x320 Pixel. Es wird im 8 Bit Parallelmodus angesteuert. Zusätzlich hat es einen resistiven Touchscreen.

- Touchscreencontroller

Der Touchscreencontroller übernimmt die komplette Messung der X, Y und Z Achse. Der integrierte Analogwandler arbeitet mit einer Auflösung von 12Bit. Ausserdem besitzt er einen Ausgang der auf High springt sobald der Touchscreen betätigt wird.

- Ein/Aus Schalter

Mit dem Ein/Aus Schalter wird die komplette Stromversorgung getrennt. Im ausgeschalteten Zustand wird nur noch die Real-Time Clock des Mikrocontrollers betrieben.

- Ein Taster
- 4 freie DIP-Switchs
- Litium-Polymer Akku

Der einzellige LiPoly Akku hat eine Ursprungsspannung von 3.7V und eine Kapazität von 1.5Ah.

- Akku Lader L6924D

Mit Hilfe des Akkuladers kann der LiPoly Akku entweder über den Powerjack oder über USB geladen werden. Mit dem Delta-Peak Verfahren wird detektiert, wann der Akku voll geladen ist, danach folgt eine Erhaltungsspannung.

- WLAN Module RN-121 von Roving Networks

Das WLAN Modul kann eigenständig eine drahtlose Verbindung herstellen und diese verwalten. Einen TCP/IP Stack ist auf dem Modul vorhanden, ebenfalls werden Netzwerkanwendungen wie FTP und Telnet angeboten. Für die Kommunikation zwischen dem WLAN Modul und dem Mikrokontroller wird ein UART gebraucht. Das Modul unterstützt eine Baudrate von bis zu 921600Bd. Über diese Schnittstelle wird das Modul mit ASCII-Befehlen konfiguriert.

- Audiodecoder VS1053 von VLSI

Dekodiert eigenständig folgende Audioformate: Ogg Vorbis, MP3, AAC, WMA, MIDI. Die analoge Soundausgabe erfolgt über einen internen 18 Bit Digital-Analog Wandler, der nach dem Delta-Sigma Verfahren arbeitet. Sämtliche Einstellungen und Daten werden über SPI übertragen.

- 3.5mm Kopfhörerausgang
- 3 frei programmierbare LEDs
- MicroSD Card

Die MicroSD Card wird mittels SPI angesprochen. Es werden nur SD Karten bis 2GB, welche mit dem FAT32 Filesystem formatiert wurden, erkannt.

- Debug USB (serielle Schnittstelle über USB)

Die serielle Schnittstelle UART0 des Mikrocontrollers wird mit dem RS232/USB Konverter FT232RL gewandelt und kann über den USB Mini B Stecker mit dem PC verbunden werden. Die ausgetauschten Daten können mit einem Programm (z.B. Teraterm) empfangen werden oder von dort aus an den Mikrocontroller gesendet werden.

- JTAG Schnittstelle

Mit der JTAG Schnittstelle wird das Programm auf dem Mikrocontroller geladen. Er dient ebenfalls zum debuggen.

5 Blockschaltbild

Aus der Beschreibung ergibt sich folgendes Blockschaltbild

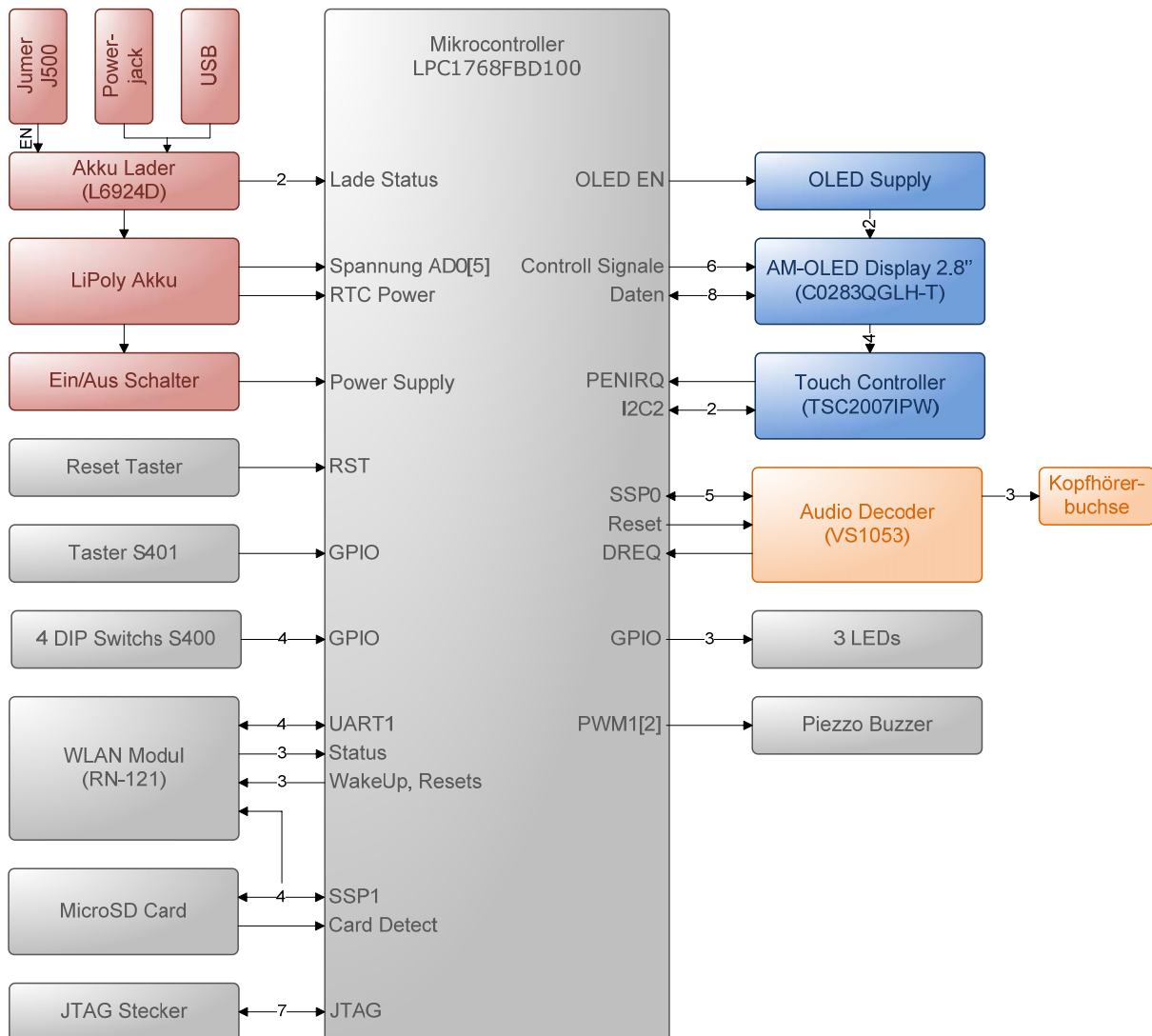


Abb. 2 Blockschaltbild

6 Pin Definitionen

6.1 WLAN Modul

Name	Port[Pin]	Funktion	Aktiv
WLAN_RX	P0[16]	UART1 (RXD1)	-
WLAN_TX	P0[15]	UART1 (TXD1)	-
WLAN_RTS	P0[22]	UART1 (RTS1)	Low
WLAN_CTS	P0[17]	UART1 (CTS1)	Low
SSP1_SCK	P0[7]	SSP (SCK1)	-
SSP1_MOSI	P0[9]	SSP1 (MOSI1)	-
SSP1_MISO	P0[8]	SSP1 (MISO1)	-
WLAN_CS	P1[29]	GPIO (Output)	Low
WLAN_WAKEUP	P1[1]	GPIO (Output)	
WLAN_SetFactoryDefaults	P1[0]	GPIO (Output)	High
WLAN_RESET	P1[4]	GPIO (Output)	High
WLAN_StatusConnecting	P0[20]	GPIO (Input)	
WLAN_StatusDataTransfer	P1[22]	GPIO (Input)	
WLAN_StatusAssociation	P0[23]	GPIO (Input)	

6.2 OLED Display

Name	Port[Pin]	Funktion	Aktiv
OLED_EN	P1[10]	GPIO Output	High
CS_OLED	P2[9]	GPIO Output	Low
OLED_RESET	P1[9]	GPIO Output	Low
FLM	P1[8]	GPIO Output	High
WRB	P2[8]	GPIO Output	Low
RS	P2[12]	GPIO Output	High
RDB	P2[13]	GPIO Output	Low
DB[1..8]	P2[0]..P2[7]	GPIO Input/Output	High

6.3 Touchscreen Controller

Name	Port[Pin]	Funktion	Aktiv
PENIRQ	P0[18]	GPIO Input INT	Low
I2C2_SCL	P0[11]	I2C2 (SCL2)	-
I2C2_SDA	P0[10]	I2C2 (SDA2)	-

6.4 Audiodecoder

Name	Port[Pin]	Funktion	Aktiv
SSPO_SCK	P1[20]	SSP0 (SCK0)	-
SSPO_MOSI	P1[24]	SSP0 (MOSI0)	-
SSPO_MISO	P1[23]	SSP0 (MISO0)	-
AUDIO_CS	P4[28]	GPIO Output	Low
AUDIO_DCS	P4[29]	GPIO Output	Low
AUDIO_DREQ	P0[24]	GPIO (Input)*	High
AUDIO_RESET	P1[19]	GPIO Output	Low

6.5 Akku Überwachung

Name	Port[Pin]	Funktion	Aktiv
BAT_CHARGE	P0[25]	GPIO (Input)	Low
BAT_COMPLETE	P0[26]	GPIO (Input)	Low
BAT_CONTROL	P1[31]	ADO (ADO[5])	Analog

6.6 Standard Ein- und Ausgänge

Name	Port[Pin]	Funktion	Aktiv
BTN	P0[6]	GPIO Input	High
	P3[26]	GPIO Input INT	
DIP_SWITCHS[1..4]	P1[14]..P1[17]	GPIO Input	High
LEDS[1..3]	P1[25]..P1[27]	GPIO Output	High
BUZZER	P3[25]	PWM1 (PWM1[2])	High

Analyse / Realisierung

7 Server Kommunikation

Die Kommunikation mit einem SHOUTcast Server wird mittels einer WLAN Verbindung über das Internet realisiert. Der SHOUTcast Server benutzt das MP3 oder das AAC+ Format für die Audiokomprimierung der Audiodateien. Für die Kommunikation wird das HTTP Transportprotokoll gebraucht.

7.1 WLAN

WLAN ist eine drahtlose Funkverbindung mit einer Reichweite von rund 100 Metern. WLAN ist die Abkürzung von *Wireless Local Area Network*, was auf Deutsch drahtloses lokales Netzwerk heisst. Mit WLAN können lokale PC Netzwerke geschaffen werden, damit sie untereinander kommunizieren können. Meistens werden verschiedene Geräte, wie z. B. Laptops, Handys oder unser WLAN Webradio über WLAN mit einem Access Point verbunden, damit sie ins Internet zugreifen können und eine Verbindung mit einem Server erstellen können.

7.1.1 Netzwerkverbindung

Als erstes muss der WLAN Webradio mit einem Netzwerk verbunden werden. Dazu muss das WLAN Modul nach verfügbaren Netzwerken suchen (genannt Scannen). Nun kann auf eines der gefundenen Netzwerke verbunden werden. Bereits beim Scannen ist ersichtlich, welche Netzwerke mit welcher Technik verschlüsselt sind. Handelt es sich um ein öffentliches Netzwerk (Open), so kann ohne Key auf dieses Netzwerk verbunden werden. Bei einer WEP-128 Verschlüsselung wird zusätzlich noch ein 26-stelliger Key gebraucht. Um in ein Netzwerk mit einer WPA-PSK oder WPA2-PSK Verschlüsselung zu verbinden, wird anstelle des Keys ein Passwort gebraucht.

7.2 Kommunikationsablauf

Die Kommunikation zwischen dem WLAN Webradio und dem SHOUTcast Server erfolgt über TCP (Transmission Control Protocol). Untenstehend ist der zeitliche Ablauf der Kommunikation abgebildet.

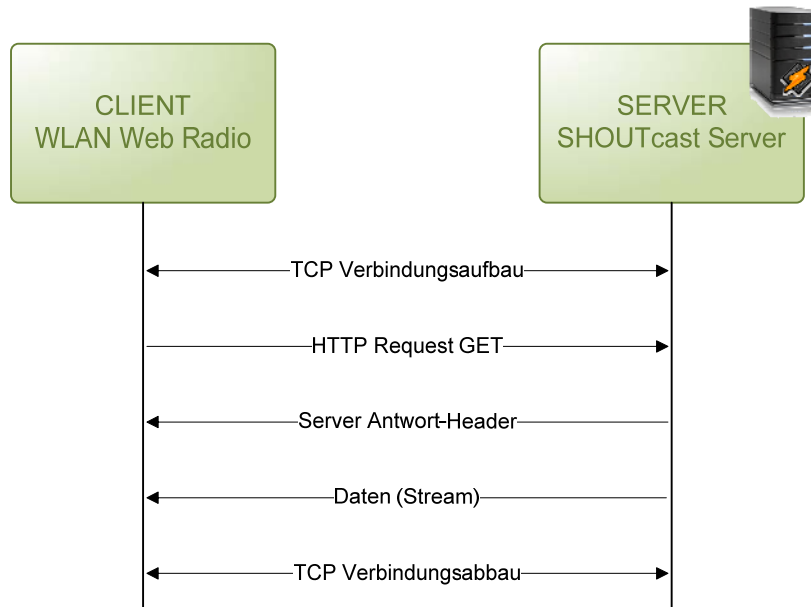


Abb. 3 TCP Kommunikationsablauf

7.2.1 Socket

Um die Kommunikation aufzubauen, wird als erstes eine Socket geöffnet. Ein Socket ist eine Verbindungseinheit auf Netzwerkbasis, der die zu übermittelten Daten in ein Protokoll verpackt. In unserem Fall handelt es sich, wie schon weiter oben erwähnt, um das TC Protokoll. Der Socket wird vollständig vom WLAN Modul erstellt und verwaltet.

7.2.2 HTTP Request

Nachdem der Socket geöffnet ist, kann eine Anfrage, dass wir den Audiostream empfangen möchten, gesendet werden. Dies geschieht mit einem HTTP Request. In ihm muss ein GET Parameter übergeben werden.

```

1 GET {/folder/data} HTTP/1.0[CR][LF]
2 User-Agent: WLAN WebRadio by BFH-TI[CR][LF]
3 Accept: /**[CR][LF]
4 Icy-MetaData: 1[CR][LF]
5 Connection: close[CR][LF]
6 [CR][LF]
  
```

Oben stehend ist ein solcher HTTP Request Header zu sehen. Der einzige variable Inhalt dieses Headers ist {/folder/data}. Diese geschweifte Klammer wird mit der Adresse des Streams, vom Serverroot ausgesehen, ersetzt.

8 Streaming

Dank dem Streaming Verfahren ist es möglich z.B. in ein MP3 File, welches sich auf dem Internet befindet, anzuhören ohne es vorher komplett auf die Festplatte herunterzuladen.

Es muss jedoch noch eine kurze Wartezeit in Kauf genommen werden, bis ein kleiner Teil der Daten in einem oben stehend gespeichert wird. Währenddessen der erste kleine Teil der Daten abgespielt wird, werden immer mehr Daten in den Oben stehend geschoben und abgespielt. Diese Wartezeit hängt davon ab, wie schnell die Internetverbindung ist und wie viel Daten pro Zeiteinheit (Bitrate: Angabe meistens in Bit/s) gebraucht werden.

8.1 SHOUTcast Protokoll

Um mit dem WLAN Webradio eine Radiosendung zu hören, muss die Radiostation zwingend ihre Sendungen über das SHOUTcast Protokoll senden. Es handelt sich hierbei um ein sehr simples Protokoll. Nachdem der HTTP Request gesendet wurde, antwortet der SHOUTcast Server. Als erstes folgt ein ICY als Bestätigung, dass der Server bereit ist um die Daten zu senden und unsere Anfrage verstanden hat. Danach folgt einmalig ein HTTP Header mit diversen Parametern und zum Schluss folgen kontinuierlich die Daten.¹

8.1.1 Antwort Header

Parameter	Bedeutung
icy-notice1	Allgemeiner Hinweis: Etwa welcher Client benutzt werden sollte
icy-notice2	Hinweise zur verwendeten Server-Software
icy-name	Sendername
icy-genre	Genre des Senders
icy-url	Webadresse des Senders
content-Type	MIME-Typ des Datenstroms (wie bei Webseiten üblich)
icy-pub	1 = Sender wird in Shoutcast.com eingetragen, sonst 0
icy-metaint	Grösse der Audioblöcke in Bytes
icy-br	Bitrate in kb/s

Nun folgt ein kleines Beispiel eines solchen Header. Die Adresse des Testservers lautet: scfire-dtc-aa05.stream.aol.com: 80/stream/1006

```

7  ICY 200 OK[CR][LF]
8  icy-notice1: <BR>This stream requires <a
      href="http://www.winamp.com/">Winamp</a><BR>[CR][LF]
9  icy-notice2: Firehose Ultravox/SHOUTcast Relay Server/Linux v2.6.0<BR>[CR][LF]
10 icy-name: Mostly Classical - S K Y . F M - Listen and Relax, it's good for you!
      www.sky.fm[CR][LF]
11 icy-genre: classical easy symphonic[CR][LF]
12 icy-url: http://www.sky.fm/classical/[CR][LF]
13 content-type: audio/mpeg[CR][LF]
14 icy-pub: 1[CR][LF]
15 icy-metaint: 16384[CR][LF]
16 icy-br: 64[CR][LF]
17 [CR][LF]
```

¹ <http://eternity.zerbe-net.org/doku.php?id=start:themen:webradiosender-suchmaschine> (16. April 2010)

8.1.2 META-Daten

Zwischen den Audioblöcken werden META-Daten gesendet. Die Daten beginnen immer mit einem Audioblock (im Bild: MP3 data). Die Grösse eines Audioblocks ist im Header angegeben. Es handelt sich um den Parameter *icy-metaint*. Dies ist eine ganzzahlige Zahl, die angibt, aus wie vielen Bytes ein Audioblock besteht. Diese *icy-metaint* muss gespeichert werden damit es möglich ist, die META-Daten zu filtern.

Je nachdem wie gross die Audioblöcke sind, kann es einige Zeit dauern, bis die Songinformationen auf dem Display dargestellt werden können. So entstehen auch kleine Verzögerungen, sobald ein neuer Song gespielt wird.



Abb. 4 SHOUTcast Protokoll²

Das erste Byte nach dem Audioblock ist das *Length Byte*. Dieses Byte muss mit 16 multipliziert werden, dann erhält man die Anzahl Bytes an META-Daten, die folgen werden.

Das *Length Byte* kann maximal 0xFF (255) sein, so wäre die maximalen META-Daten 4080 Bytes gross.

Die meiste Zeit wird das *Length Byte* aber 0 sein. Dies bedeutet, dass keine META-Daten übertragen werden. In diesem Fall muss nur das *Length Byte* weggefiltert werden und das nächste Byte gehört bereits wieder zum nächsten Audioblock und muss folglich an den Audiodecoder gesendet werden.³

8.1.2.1 Beispiel

```
18 StreamingTitle='Bon Jovi - It's My Life'StreamingUrl=''
```

Leider sind die META-Daten nicht standardisiert. Somit kann jede Radiostation selber wählen, welche Informationen sie den Clients und schlussendlich den Zuhörern mitteilen möchten. Es kann auch sein, dass eine Radiostation die META-Daten missbraucht um Werbung zu machen.

² <http://www.smackfu.com/stuff/programming/shoutcast.gif> (16. April 2010)

³ <http://www.smackfu.com/stuff/programming/shoutcast.html> (16. April 2010)

8.2 Audiodecoder

Der eingesetzte Audiodecoder VS1053 von der finnischen Firma VLSI kann u.a. die beiden Audioformate MP3 und AAC+ dekodieren. Er wird über SPI angesprochen und besitzt zwei Interfaces. Zum einen dem SCI (Serial Command Interface), mit dem wird er konfiguriert und alle Parameter ausgelesen, und zum anderen dem SDI (Serial Data Interface), in diesem Interface werden die Audiodaten gesendet.

SDI Pin	SCI Pin	Description
XDCS	XCS	Active low chip select input. A high level forces the serial interface into standby mode, ending the current operation. A high level also forces serial output (SO) to high impedance state. If SM_SDISHARE is 1, pin XDCS is not used, but the signal is generated internally by inverting XCS.
	SCK	Serial clock input. The serial clock is also used internally as the master clock for the register interface. SCK can be gated or continuous. In either case, the first rising clock edge after XCS has gone low marks the first bit to be written.
	SI	Serial input. If a chip select is active, SI is sampled on the rising CLK edge.
-	SO	Serial output. In reads, data is shifted out on the falling SCK edge. In writes SO is at a high impedance state.

Abb. 5 Audiodecoder Interfaces⁴

Mit dem Data Request Pin *DREQ* wird angezeigt, ob der Audiodecoder bereit ist Daten zu empfangen oder nicht. Ist *DREQ* high, so können dem Audiodecoder Daten oder Kommandos gesendet werden. Falls in das 2048 Bytes grosse FIFO nicht mehr als 32 Bytes Daten gesendet werden können, schaltet *DREQ* auf low. Der zweite Grund, wann er auf low schaltet, ist, wenn ein Kommando verarbeitet wird. In diesen Fällen dürfen keine weiteren Daten oder Kommandos geschickt werden.

8.2.1 MP3

MP3 ist die Abkürzung für *MPEG 1 Layer 3*. Es handelt sich dabei um ein komprimiertes Audioformat. Wie die meisten komprimierten Audioformate, baut MP3 auf der Psychoakustik auf. Das heisst, es werden nur Töne gespeichert, welche für den Menschen hörbar sind, alle anderen Töne entfallen. Durch dieses Verfahren ist es möglich, dass die Qualität des Musikstückes trotz der Komprimierung nicht deutlich schlechter wird.

Der Aufbau einer MP3 Datei gleicht vielen anderen Audioformaten. Zuerst wird ein Frame-Header geschickt, welcher 32 Bit lang ist, anschliessend kommen Frame-Daten, welche die komprimierten Audiodaten enthalten.

Ein wichtiger Vorteil ist, dass das MP3-Format streamfähig ist.⁵

8.2.2 AAC+

AAC+ ist die Abkürzung von MPEG-4 High Efficiency Advanced Audio Coding. Dieses Audioformat wird u.a. auch HE-AAC oder AAC+ v1 genannt. Es handelt sich um eine verlustbehaftete Audiodatenkompressierung. Diese Kompressierung liefert bei niedrigen Bitraten, bis 96kb/s, vergleichsweise gute Ergebnisse. Deshalb wird dieses Format oft

⁴ VS1053 Datenblatt (Seite 18)

⁵ <http://de.wikipedia.org/wiki/MP3> (16. April 2010)

beim Live-Streaming oder im Mobilfunk eingesetzt. Für Bitraten ab 96kb/s eignet sich diese Kompressierung nicht mehr, da sie einen erheblichen Qualitätsverlust aufweist.⁶

⁶ <http://de.wikipedia.org/wiki/AAC%2B> (16. April 2010)

9 Zeitlicher Ablauf

Die meisten Radiostationen verteilen ihre Sendungen mit einer Bitrate von 32kb/s bis 128kb/s. Selten findet man auch eine Radiostation, die gar mit einer Bitrate von 256kb/s oder 320kb/s Musik streamt.

9.1 Berechnung Baudrate

Bei der Berechnung gehen wir von einem Stream mit einer Bitrate von 128kb/s aus.

$$128 \frac{kb}{s} = 131072 \frac{\text{bit}}{s} \rightarrow 16384 \frac{B}{s}$$

$$f_{\text{WLAN}} = 10 \cdot 16384 \frac{B}{s} = \underline{\underline{163840Bd}}$$

Somit wird für das Abspielen eines 128kb/s Streams mindestens eine Baudraten des WLAN Moduls von 230400Bd benötigt.

Möchte hingegen ein Stream mit 320kb/s gespielt werden, müsste die Baudrate des WLAN Moduls verdoppelt werden auf 460800Bd.

$$320 \frac{kb}{s} = 327680 \frac{\text{bit}}{s} \rightarrow 40960 \frac{B}{s}$$

$$f_{\text{WLAN}} = 10 \cdot 40960 \frac{B}{s} = \underline{\underline{409600Bd}}$$

9.2 Berechnung SPI Tackt

Bei der Wiedergabe eines 128kb/s Streams müssen die 16384 Bytes pro Sekunde an den Audiodecoder gesendet werden.

$$f_{\text{SSP}} = 8 \cdot 16384 \frac{B}{s} = \underline{\underline{131.1kHz}}$$

Für den 320kb/s Streams wäre es mindestens

$$f_{\text{SSP}} = 8 \cdot 40960 \frac{B}{s} = \underline{\underline{327.7kHz}}$$

Die tatsächliche SPI Frequenz muss aber deutlich höher gewählt werden, da die Daten nicht ununterbrochen gesendet werden können, sondern sie müssen vorher byteweise in das SPI TX Register geschrieben werden, sowie das Chip Select muss während dem senden auf low geschaltet werden. Ausserdem möchten wir zwischen den Daten auch noch Einstellungen am Audiodecoder vornehmen, oder das interne FIFO des Decoders füllen.

Der Audiodecoder erlaubt eine SPI Frequenz von $\text{Clock}/7^7$. Stromsparender Betrieb wäre, wenn man die PLL des Audiodecoders nicht gebrauchen müsste. In diesem Fall ist die Clockfrequenz gleich der Oscillatorfrequenz.

$$f_{\max} = \frac{12.288\text{MHz}}{7} = \underline{\underline{1.755\text{MHz}}}$$

Dies reicht vollkommen für das Abspielen eines bis zu 320kb/s Streams. Die SPI Frequenz wurde auf 1.5MHz eingestellt.

⁷ VS1053 Datenblatt (Seite 12)

10 Betriebsdauer

Der WLAN Webradio wird mit einem Litium-Polymer Akku betrieben, der eine Kapazität von 1.5Ah hat.

Die Stromaufnahme beim Abspielen einer Radiostation mit 64kb/s beträgt rund 160mA. Daraus lässt sich errechnen, wie lange der Webradio mit einer Akkuladung betrieben werden kann.

$$t = \frac{Q}{I} = \frac{1.5Ah}{160mA} = \underline{\underline{9h23min}}$$

Sucht das WLAN Modul nach verfügbaren Netzwerken, wird die grösste Leistung gebraucht (Stromaufnahme 200mA). Doch dies dauert nur 2.6 bis 3 Sekunden.

Die Betriebsdauer wird durch Radiostationen, die ihre Sendung mit einer höheren Bitrate senden, reduziert.

Die Stromaufnahme der einzelnen Komponenten kann aus den jeweiligen Datenblätter entnommen werden.

11 Menüführung

Die Menüführung soll möglichst einfach zu bedienen sein. Die komplette Struktur soll mit möglichst wenig verschachtelten Ebenen auskommen, damit der Benutzer schnell alle Einstellungen und Optionen betätigen kann.

Da es sich um eine batteriebetriebene Anwendung handelt, wird darauf geachtet, dass die Benutzeroberfläche mit dunklen Farben gestaltet wird. So kann dank dem OLED Display eine grosse Menge an Energie gespart werden. Bei einem OLED Display sind alle Pixel mit der Farbe schwarz komplett ausgeschaltet und benötigen so keine Energie. Um hingegen eine weisse Fläche darzustellen, müssen alle drei RGB Farben eines Pixels voll aufleuchten, hierfür wird viel mehr Energie benötigt.

Implementiert wurden die Menüführung und alle anderen Ausgaben auf dem Display mit Hilfe der OpenSource Grafikbibliothek Microwindows. Dabei handelt es sich um eine relativ einfache Grafikbibliothek, welche ausschliesslich in der Hochsprache C aufgebaut wurde.

11.1 Übersicht

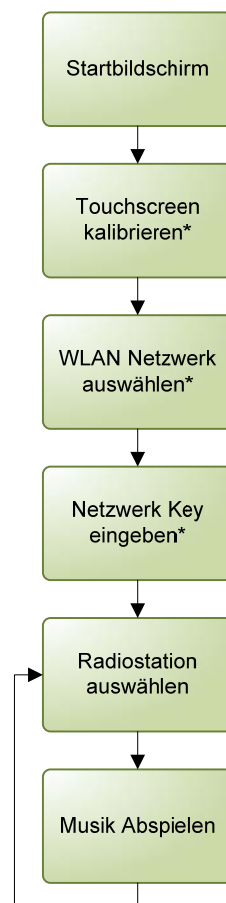


Abb. 6 Menüstruktur

* Alle Menüpunkte, die durch einen hochgestellten Stern gekennzeichnet sind, werden nicht in jedem Fall automatisch durchlaufen. Z.B. wenn ein Netzwerk ohne Verschlüsselung (Open) ausgewählt wird, muss kein Key angegeben werden. Also wird dieser Menüpunkt übersprungen.

Software

12 Grober Programmablauf

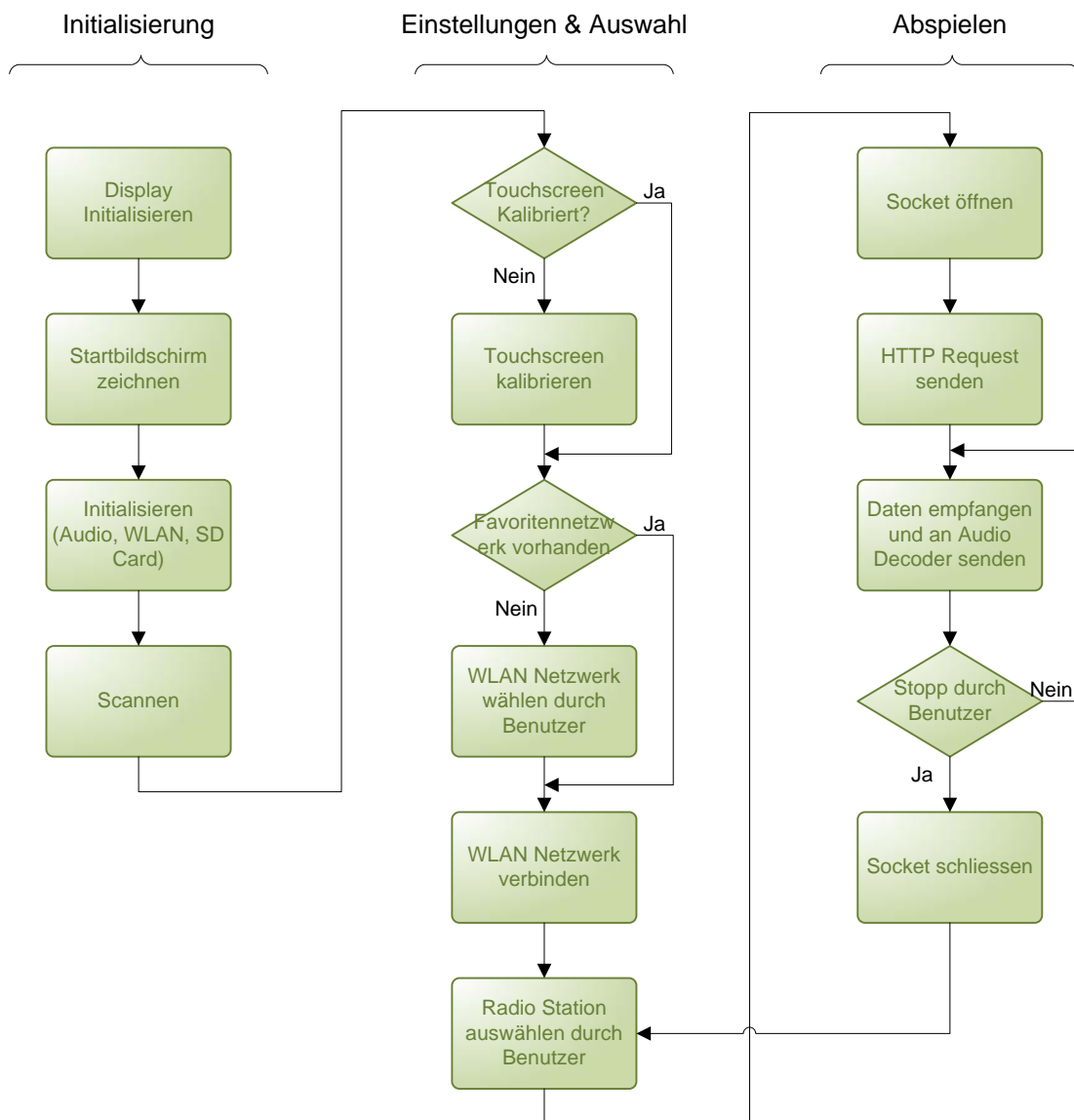


Abb. 7 Grober Programmablauf

13 Buffer Konzept

Um flexibel zu bleiben, werden auf dem Mikrokontroller zwei verschiedene Ringbuffer eingesetzt. Zusätzlich wird das SSP FIFO für noch schnellere Datenübertragungen an den Audiodecoder und weniger Programmunterbrüche durch Interrupts aktiviert.

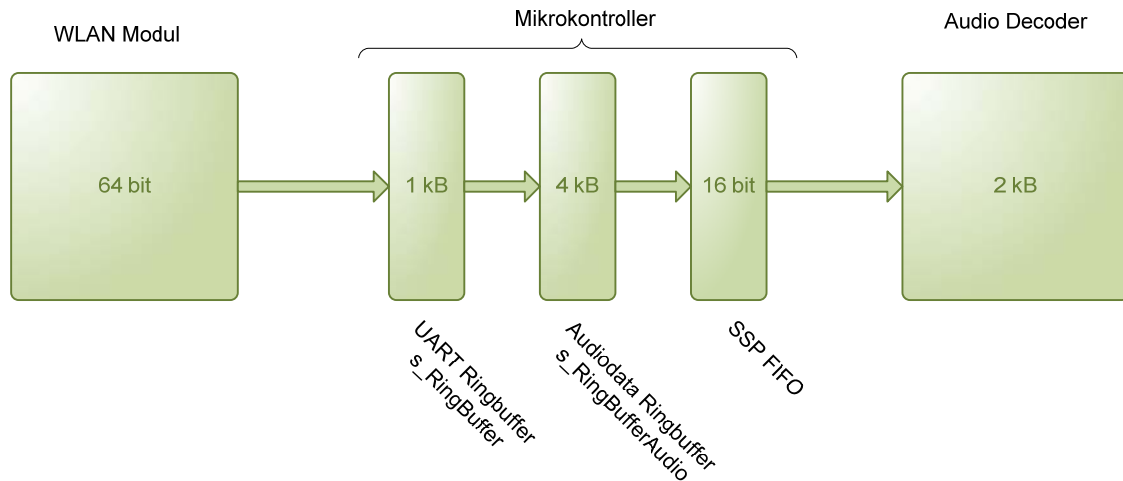


Abb. 8 Buffer Konzept

Alle empfangenen Zeichen vom WLAN Modul werden mit Interrupts ausgelesen und im UART Ringbuffer gespeichert, egal ob es sich dabei um Audiodaten, Parametern oder Rückgabewerte handelt. Um etwas an das WLAN Modul zu senden, steht ebenfalls ein Ringbuffer zur Verfügung, der in Interrupts die Zeichen über die serielle Schnittstelle sendet.

Die tatsächlichen Audiodaten werden vom UART Ringbuffer an den Audiodata Ringbuffer übertragen, der sie mit der Unterstützung eines 16Bit FIFOs über SSP an den Audiodecoder sendet. Dank dem separaten Audiodata Ringbuffer ist es möglich, ohne weiteres eine andere Quelle für die Audiodaten zu nehmen. Z.B. um ein MP3 File von der MicroSD Card abzuspielen.

14 Betriebssystem

Es wird das Open Source Betriebssystem TNKernel eingesetzt. TNKernel wurde vom Russen Yuri Tiomkin entwickelt. Es beinhaltet folgende Features:

- Tasks
Pseudo parallel laufende Funktionen (Programmteile).
- Semaphores
Regelt zeitkritische Aufgaben, wie z.B. der Hardwarezugriff damit nicht zwei verschiedene Tasks gleichzeitig dieselbe serielle Schnittstelle brauchen.
- Mutexes
Grundsätzlich das Gleiche wie *Semaphores*, mit der Ausnahme, dass nur Tasks, die die gemeinsam verwendete Hardware freigeben können, welche sie auch gesperrt haben.
- Data Queues
Mit *Data Queues* können Daten (auch ganze Strukturen) aus mehreren Tasks ausgetauscht werden. Sie arbeiten mit Pointer und besitzen ein FIFO.
- Events
Events werden hauptsächlich zum Synchronisieren von Tasks verwendet. Ein Event ist jeweils eins von 32 Bits und hat folglich nur zwei Zustände, entweder ist das Eventflag gesetzt oder nicht.
- Memory Pools
Ein *Memory Pool* wird dazu benötigt, um einen Datenblock mit konstanter Grösse an einen anderen Task zu senden.

15 Tasks

Die Entscheidung fiel auf insgesamt vier Tasks:

- Webradio Task

Der Webradio Task ist das Hauptprogramm. Er regelt den ganzen Ablauf und alle Möglichkeiten die dem Benutzer zu Verfügung stehen. Dieser Task erteilt den anderen Task Aufgaben, die sie erledigen müssen.

- GUI Task

Der GUI Task ist gegeben vom Microwindows. Dieser Task verwaltet das ganze Display, sowie den Touchscreencontroller. Die ganze Menüführung, resp. die Anzeige, wird in diesem Task gezeichnet.

- Streaming Task

Der Streaming Task erledigt die komplette Kommunikation mit dem WLAN Modul. Er erhält alle seine Aufgaben vom Webradio Task. Für das Streaming und das Filtern der META-Daten ist dieser Task ebenfalls zuständig, bevor er die Audiodaten an den Playing Task weitersendet.

- Playing Task

Der Playing Task übernimmt die komplette Steuerung des Audiodecoders. Sobald der Decoder bereit ist Daten zu empfangen, werden, sofern im Ringbuffer vorhanden, die Daten an den Audiodecoder gesendet. Einstellungen, wie z.B. für die Lautstärke, werden auch in diesem Task gemacht.

15.1 Synchronisation & Datenaustausch

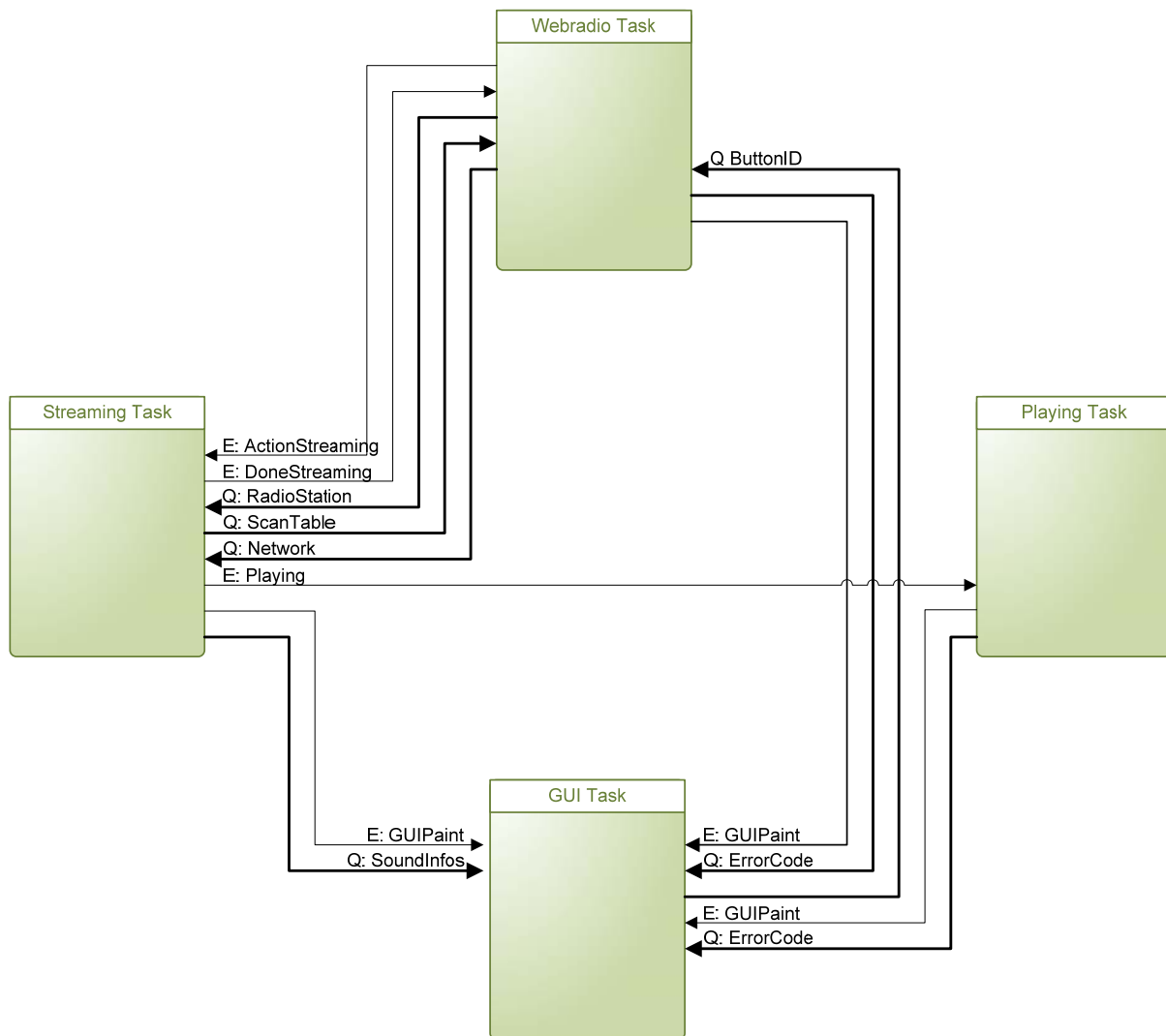


Abb. 9 Task Synchronisation

E: Events

Q: Data Queues

15.2 Webradio Task

Der Webradio Task wird bereits nach dem Initialisieren des Betriebssystems automatisch gestartet. Nachdem der Startbildschirm gezeichnet ist, startet er den *Playing* und den *Streamin Task*.

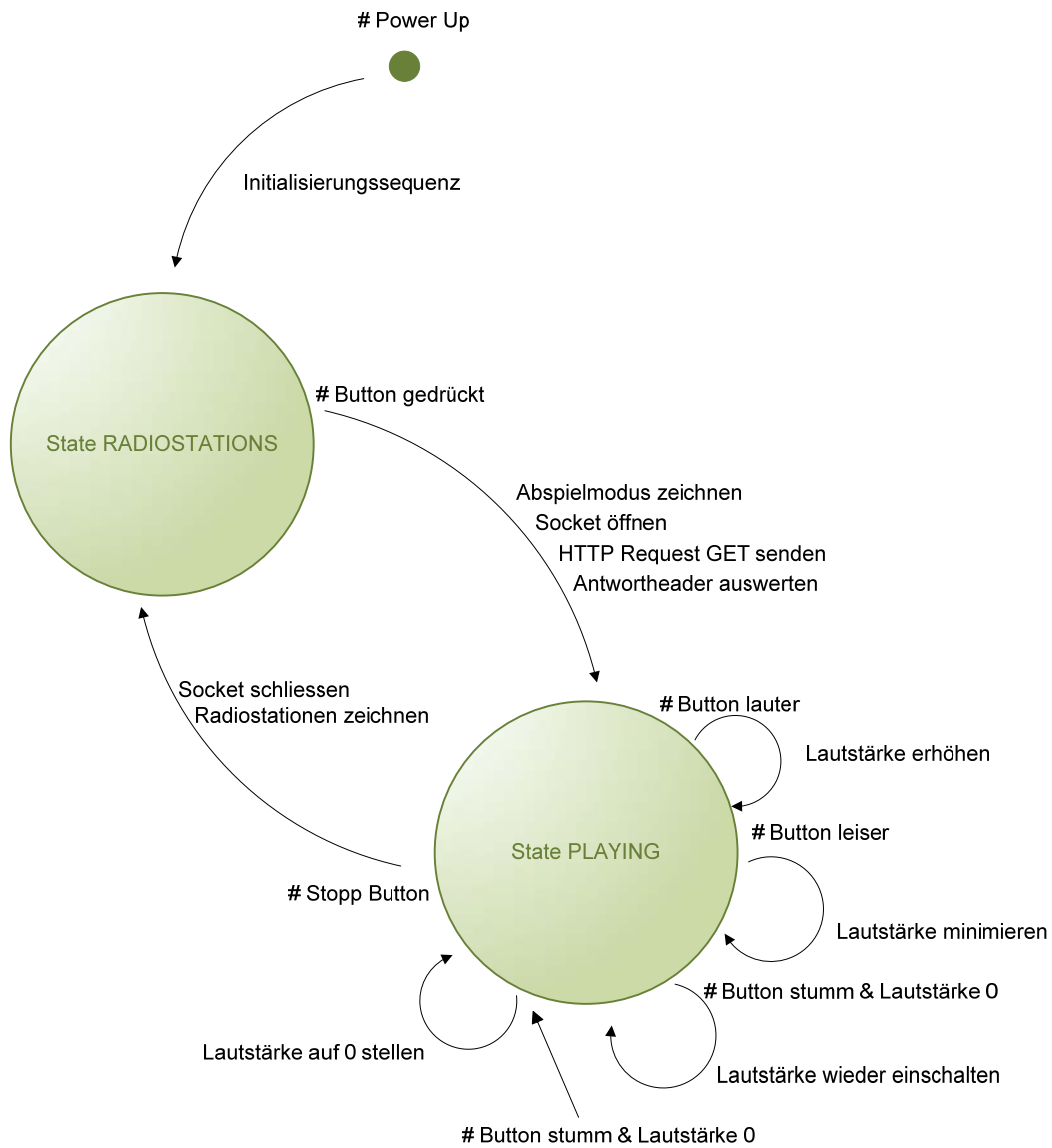


Abb. 10 Webradio State Event Diagram

15.2.1 Initialisierungssequenz

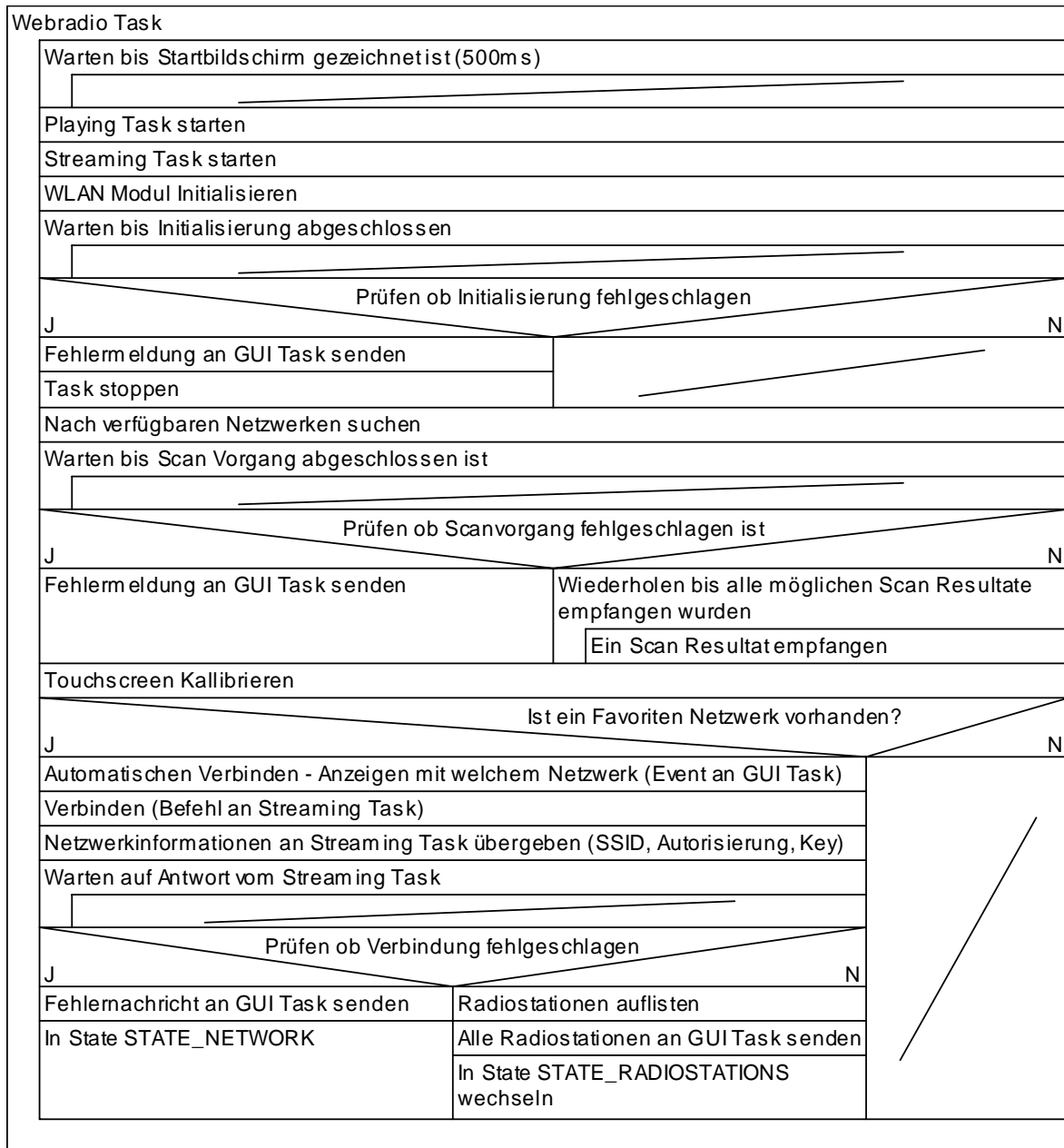


Abb. 11 Webradio Task Struktogramm

15.3 GUI Task

Der GUI Task wird ebenfalls nach dem initialisieren des Betriebssystems automatisch gestartet. Als erstes werden das Fenster und alle Buttons erstellt. Die Buttons dürfen aber noch nicht angezeigt werden. Später, wenn man sie anzeigen möchte, werden sie auf *visible* (sichtbar) gestellt. Dies wird von Microwindows unterstützt.

Die Grafikbibliothek arbeitet mit sogenannten Messages. Der Timer von Microwindows wurde so eingestellt, dass er alle 250ms auslöst und eine Message sendet. In der Callback Funktion hat man die Möglichkeit herauszufinden, wann eine Timermassage abgearbeitet wird. Sobald eine solche *WM_TIMER* Message am Abarbeiten ist, wird geprüft, ob von einem anderen Task ein Event gesetzt wurde, dass etwas auf dem Display gezeichnet werden soll. Die Events werden alle 250ms überprüft. Ist eines gesetzt, so wird der GUI Task diese Aufgabe erledigen und anschliessen das Eventflag löschen.

15.4 Streaming Task

Der Streaming Task wartet auf ein Eventflag, welches vom Webradio Task gesetzt wird. Zu seinen Aufgaben gehört folgendes:

- WLAN Modul initialisieren
- Nach verfügbaren Netzwerken Suchen
- In eine WLAN Netzwerk verbinden
- Verbinden mit einer Radiostation (Socket öffnen, Header senden und Antwortheader auswerten)
- Ankommender Stream verarbeiten
- META-Daten herausfiltern
- Verbindung mit der Radiostation schliessen
- Verbindung mit dem WLAN Netzwerk beenden

15.5 Playing Task

Der Playing Task hat die höchste Priorität. Seine Aufgabe ist es, den Audiodecoder zu initialisieren und alle Audiodaten an ihn zu senden.

Solange das Sende-FIFO der seriellen Schnittstelle SSP nicht leer ist, ist der Playing Task im wartenden Zustand. Falls kein Zeichen zum Senden im Audio Ringbuffer ist oder keine Einstellung vorgenommen werden soll, ist der Task ebenfalls im wartenden Zustand.

15.5.1 Struktogramm

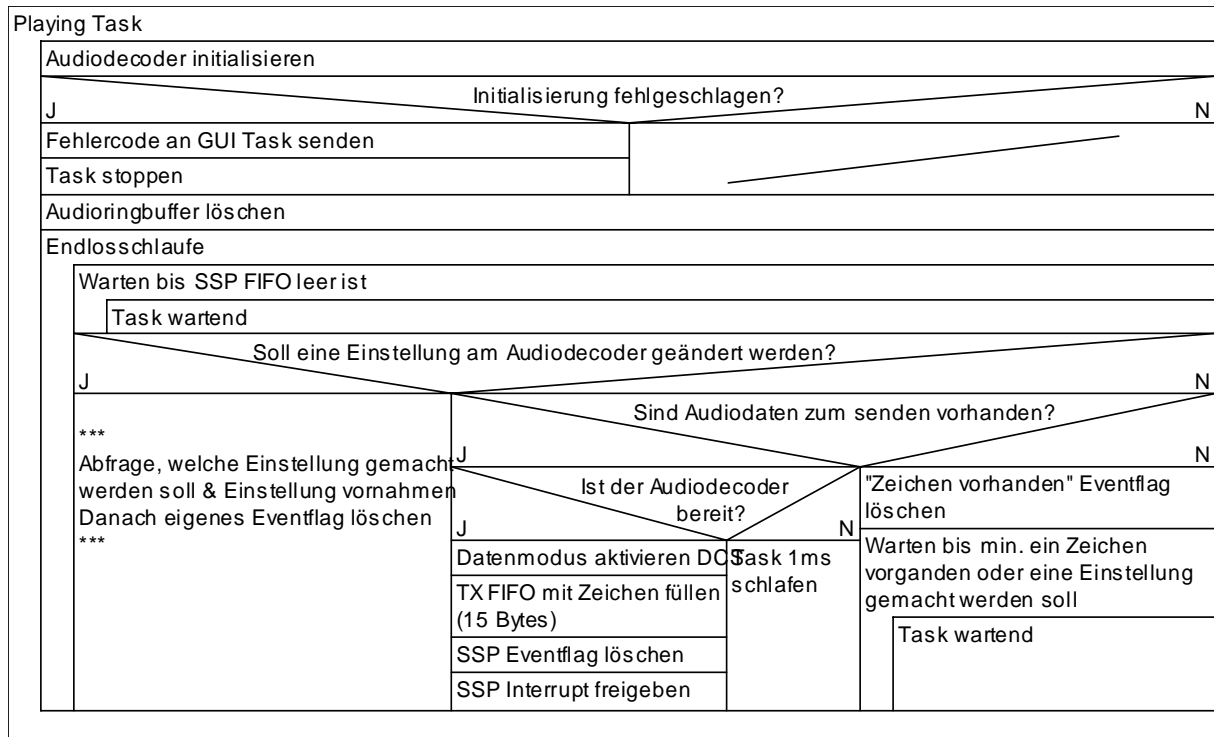


Abb. 12 Playing Task Struktogramm

Schlusswort

16 Stand der Arbeit

Im jetzigen Stand ist das Projekt Wireless LAN Webradio noch nicht abgeschlossen. Es tauchten während der Entwicklung einige Probleme auf. Momentan werden alle Module (Display, Touchscreencontroller, WLAN Modul und Audiodecoder) korrekt initialisiert. Während der Initialisierungssequenz wird der Startbildschirm mit dem Logo und der Softwareversionsnummer angezeigt. Danach kann der Touchscreen kalibriert werden.

Bereits währenddessen der Startbildschirm noch angezeigt wird, wird das erste mal nach Netzwerken gesucht. Die gefundenen Netzwerke werden korrekt in einem Struktur-Array abgespeichert. Doch können sie momentan nicht auf dem Display angezeigt werden. Eine Überprüfung ob ein Favoritennetzwerk gefunden wurde, wird nicht gemacht. Stattdessen wird versucht, auf das Favoritennetzwerk *WiFi-T204* zu verbinden. Somit ist das Netzwerk als fix im Programm gegeben. Während dem Verbindungsvorgang sieht der Benutzer in welches Netzwerk verbunden wird.

Nach dem erfolgreichen Verbinden in das Netzwerk, kann der Benutzer aus bis zu 10 Radiostationen auslesen. Die Radiostationen werden in einem Struktur-Array gespeichert und können momentan nur im Programmcode geändert werden.

Wird eine Radiostation angeklickt, so öffnet das WLAN Modul einen Socket und sendet den HTTP Request Header ab. Vom SHOUTcast Server erfolgt ein korrekter HTTP Header als Antwort, welcher richtig verarbeitet wird.

Bei der Wiedergabe der Audiodaten taucht das erste Problem auf. Die Daten kommen vom WLAN Modul schneller an, als sie der Audiodecoder verarbeiten kann. Das WLAN Modul sendet die Daten paketweise. Einen insgesamt 8kb Buffer auf dem Mikrocontroller reicht nicht aus um die Zeit zu überbrücken, während der Audiodecoder mit dem DREQ Pin blockiert.

Sobald nur ein Zeichen verloren geht, funktioniert das Herausfiltern der META-Daten nicht mehr. Es wird durch den Verlust einen Versatz geben. In diesem Fall wird anstelle des *Length Byte* ein anderes Byte genommen und so zum Teil grosse Teile (bis 4080kb) des Liedes weggefiltert. Ausserdem sind die META-Daten, falls sie am Audiodecoder ausgegeben werden, störend zu hören. Um dieses Timing Problem in den Griff zu bekommen, versuchte ich das WLAN Modul mit dem CTS zu sperren, sobald mein Buffer voll ist. Doch das Flow Control des WLAN Moduls funktioniert nicht. Auch wenn das CTS Signal High ist, sendet das Modul weiter. Die drei LEDs auf dem WLAN Terminal wurden so konfiguriert, dass sie aufleuchten sobald die UART Schnittstelle des WLAN Moduls gesperrt werden sollte. So wird sichtbar, wann und wie für lange, Zeichen verloren gehen.

Um das funktionierende Herausfiltern der META-Daten dennoch zu demonstrieren, werden dem Audiodecoder die Daten, unabhängig davon ob er bereit ist, zugesendet. So wird der Ringbuffer niemals überlaufen, da über SPI die Daten viel schneller gesendet

werden können, als sie vom UART empfangen werden. Dies macht sich beim Hören als Springen im Lied (spulen) bemerkbar. Doch auch wenn vom UART Zeichen verloren gehen, hört es sich so an. Doch bei dieser Methode können wenigstens die META-Daten richtig gefiltert werden.

Die folgenden META-Daten werden auf dem Display angezeigt:

1. Name des Senders
2. Genre des Senders
3. Bitrate des Streams
4. Interpret des Liedes
5. Titel des Liedes

Die ersten drei Punkte werden aus dem einmaligen Antwort-Header genommen. Der Interpret kann bei den getesteten Radiostationen nicht entnommen werden, da er nicht in den META-Daten vorhanden ist. Aus den getesteten Sendern kann einzig der Titel entnommen werden. Aus zeitlichen Gründen reichte es nicht, die META-Daten zu *parse*n, doch glücklicherweise ist der Aufbau der META-Daten bei den eingetragenen Radiostationen gleich. So kann mit einem Offset als Startwert gearbeitet werden und bis zum Titel abschliessendem Zeichen ("\") eingelesen werden.

Die Lautstärke kann in insgesamt 15 Stufen eingestellt werden. Dazu stehen zwei Buttons zur Verfügung. Zusätzlich gibt es einen Stummtaster, der beim zweiten Klick wieder auf die vorangehende Lautstärke zurückstellt.

Klickt man auf den Stopp Button so wird der aktuelle Socket geschlossen und der Benutzer hat die Möglichkeit eine andere Radiostation auszuwählen. Trotz der Bestätigung vom WLAN Modul, wird der Socket nicht richtig geschlossen. Sobald der Socket wieder geöffnet wird, werden die Daten weitergeschickt, ohne dass eine Anfrage hierfür abgesendet wurde. Deshalb ist es beim jetzigen Stand nicht möglich, ein zweites Mal eine Radiostation auszuwählen und diese anzuhören. Wird dennoch eine zweite, andere Radiostation ausgewählt, hört man zwei verschiedene Lieder pseudoparallel.

Aus zeitlichen Gründen, wird die Fehlerausgabe auf einen blauen Bildschirm gezeichnet und nicht wie geplant, in einem neuen Popupfenster über dem Hauptfenster. Dies hat der Nachteil, dass bei jedem Fehler das System neu gestartet werden muss. Bei der Lösung mit dem Popup Fenster, wäre ein Button zum Abbrechen vorgesehen, damit man bei kleineren Fehlern das System nicht neu starten muss, sondern nach dem Bestätigen wieder zurückwechseln kann.

16.1 Testprotokoll

Das Testprotokoll liegt im Anhang bei.

17 Probleme / Schwierigkeiten

Während meiner IPA hatte ich mit drei grösseren Problemen zu kämpfen. Alle der drei Probleme haben mit dem WLAN Modul zu tun. Eines davon konnte ich lösen, die anderen beiden sind auf Fehler des WLAN Modul zurückzuführen. Es handelt sich bei diesem WLAN Modul um eine junge Generation, es kam erst im Dezember, letztes Jahr auf den Markt und an der Firmware wird auch immer weitergearbeitet. Ich arbeitete mit der Firmwareversion 2.15. Seit dem 16. März 2010 wäre die neuste Version 2.19 verfügbar.

17.1 Ad hoc Modus

Am Anfang verlor ich viel Zeit, weil ich keine Verbindung mit einem Netzwerk erstellen konnte. Der DHCP Server konnte dem WLAN Modul keine IP vergeben, deshalb wurde nach einem Timeout vom Modul selber eine 169.xxx.x.x IP-Adresse zugewiesen.

Es stellte sich heraus, dass es sich um ein sehr hardwarenahes Problem handelte. Der PIO9 Pin (Set factory Defaults) ist mit dem Mikrocontroller verbunden. Da ich diese Funktion nicht gebrauche, habe ich diesen Pin auch nicht als Ausgang gesetzt. Beim Mikrocontroller sind alle GPIO standardmässig als Eingänge mit Pull-Ups geschaltet. Dieser Pull-Up ist niederohmiger als der Pull-Down Widerstand der WLAN Moduls am PIO9. So wurde beim Aufstarten des WLAN Moduls am PIO9 ein high gelesen und ein Ad hoc Modus startete. Dieser Modus wäre dazu gedacht, das Modul über diese Ad hoc Verbindung zu konfigurieren.

Die Lösung des Problems war einfach: Der GPIO P1[0] des Mikrocontrollers, der mit PIO9 vom WLAN Modul verbunden ist, musste als Ausgang und logisch low geschaltet werden.

17.2 Flow Control

Die Audiodaten kommen vom WLAN Modul blockweise an. Der Audiodecoder kann diese Daten nicht schnell genug verarbeiten. Währenddessen der Audiodecoder das Empfangen weiterer Zeichen mit dem DREQ Pin sperrt, werden die Ringbuffer gefüllt. Alle Zeichen die empfangen werden, wenn der Ringbuffer voll ist, gehen verloren.

Da für noch grössere Buffer keinen Speicherplatz mehr auf dem Mikrocontroller vorhanden ist, müsste man dem WLAN Modul mitteilen, dass es keine Daten mehr senden darf. Dazu ist bei der seriellen Schnittstelle (UART) das Flow Control vorgesehen. Doch auch bei eingeschaltetem Flow Control und CTS high, werden weiter Daten an den Mikrocontroller gesendet.

Das CTS des WLAN Moduls funktioniert also nicht.

17.3 Socket schliessen

Nachdem die Wiedergabe einer Radiostation beendet wird, soll der Socket geschlossen werden. Das WLAN Modul bestätigt, dass der Socket geschlossen sei, doch er wird nicht richtig geschlossen. Das Modul hält die Verbindung zum SHOUTcast Server aufrecht und sobald ein neuer Socket geöffnet wird, kommen weiter Daten an, obwohl noch kein HTTP Request Header gesendet wurde. Somit ist es nicht möglich eine zweite Radiostation abzuspielen.

Dieses Socket Problem ist ein Bug des WLAN Moduls. Mein Lehrmeister testete dieses Phänomen mit der aktuellsten Firmwareversion 2.19, dort funktionierte das Schliessen der Socketverbindung auch nicht wie erwartet.

18 Mögliche Erweiterungen

Bevor man mit Erweiterungen anfangen sollte, wäre es sinnvoll die geschilderten Probleme zu lösen. Dazu wird am besten Kontakt mit dem Hersteller des WLAN Moduls, Roving Networks, aufgenommen.

Weiter sollte der Webradio mit den Funktionen erweitert werden, welche ich aus zeitlichen Gründen nicht während der IPA realisieren konnte. Es sollte dem Benutzer die Möglichkeit gegeben werden, das WLAN Netzwerk selber auszuwählen und den Key einzugeben. Die Eingabe wird am besten mit einer eingeblendeten Tastatur auf dem Display realisiert. Für den Benutzer wäre es auch viel spannender, wenn er die Radiostationen selber definieren könnte, diese also nicht mehr im Programmcode fix eingetragen sind. Um dies zu realisieren, muss der Benutzer Radiostationen löschen oder überschreiben können und neue Radiostationen mit Eingabe der URL und dem Port hinzuzufügen.

Die Erweiterungen sind dank der umfangreichen Hardware riesig. Mit dem Beschleunigungssensor könnte man das Display automatisch, je nach Lage, umkehren. Sinnvoll wäre auch eine MP3 Player Applikation. Falls man unterwegs kein WLAN Netzwerk zur Verfügung hat, könnte man MP3 Files von der MicroSD Card abspielen lassen. Es wäre auch möglich auf die SD Card Radiosendungen zu speichern und diese zu einem späteren Zeitpunkt zu hören. Für unterwegs oder bei Besprechungen wäre es sicherlich nicht schlecht, wenn man noch auf eine Diktierfunktion zurückgreifen könnte. Das Mikrofon hierfür ist schon vorhanden und kann vom Audiodecoder gelesen werden. Dieser kann eine WAV-Datei erstellen, die man auf der MicroSD Card speichern kann.

19 Erfahrungen

Ich fand meine IPA eine hochspannende Arbeit. Das erste Mal in meiner Lehre stand ich vor einer solchen komplexen Softwareaufgabe. Zu Beginn meiner Arbeit wusste ich nicht genau wie ich die Aufgabe lösen könnte und ob ich so meine Softwareanalyse überhaupt umsetzen kann. Das lag u.a. daran, dass ich noch keine Erfahrungen mit einem Betriebssystem gemacht hatte. Doch ich verstand sehr schnell die Möglichkeiten, die unser Betriebssystem bietet. Ich hatte auch kein Problem mit den Tasks und dem Synchronisieren. Das Betriebssystem TNKernel läuft sehr stabil und zuverlässig.

Für mich war der Einsatz des Betriebssystem die beste Erfahrung, davon kann ich sicherlich viel profitieren. Denn in der Industrie wird sehr oft ein Betriebssystem eingesetzt, da es das Design vereinfacht. Die wenigsten meiner Schulkameraden werden jemals in ihrer Lehre die Möglichkeit haben, mit einem Betriebssystem zu arbeiten.

Die vorbereiteten Bibliotheken für das WLAN Modul und den Audiodecoder mussten noch auf das Betriebssystem angepasst werden, da sie eigentlich für sequenziell ablaufende Programme geschrieben wurden und auch nur so getestet wurden.

Es war auch das erste Mal, dass ich ein Grafikdisplay nicht nur zu Testzwecken oder kleineren Übungen gebraucht habe. Zu der eingesetzten Grafikbibliothek Microwindows gibt es leider keine Dokumentation, in der alle Möglichkeiten und wie man sie realisiert beschrieben sind. So musste ich Beispiele, die mir mein Lehrmeister gegeben hat, studieren und versuchen diese umzusetzen.

20 Dank

Zum Schluss möchte ich mich ganz herzlich bedanken, bei allen, die mich während meiner IPA unterstützten, insbesondere meinem Betreuer und Lehrmeister Martin Aebersold, sowie den beiden Experten Martin Gerber und Wolfgang Schnabel.

21 Bedienungsanleitung

Eine ausführliche Bedienungsanleitung ist separat vorhanden.

Informationen

22 Abkürzungen

Wort	Beschreibung
AAC+	Advanced Audio Coding, Komprimiertes Audioformat
ASCII	American Standard Code for Information Interchange, 7 Bit Zeichencodierung
CS	Chip Select, Aktiviert Slave SPI Schnittstelle
CTS	Clear To Send, UART Signal
DCS	Data Chip Select
FIFO	First In First Out, Verfahren der Datenspeicherung
FTP	File Transfer Protocol, ein Netzwerkprotokoll zur Dateiübertragung
GPIO	General Purpose Input Output, Allzweck Ein- oder -Ausgang
GUI	Graphical User Interface, Grafische Benutzeroberfläche
HTTP	Hypertext Transfer Protocol, Protokoll zur Datenübertragung
ICY	
IP	Internet Protocol, weit verbreitetes Netzwerkprotokoll
JTAG	Joint Test Action Group, Debugschnittstelle
MIME-Type	Internet Media Type, klassifiziert die Daten im Rumpf einer Nachricht im Internet
MISO	Master in Slave out
MOSI	Master out Slave in
MP3	MPEG 1 Layer 3, Komprimiertes Audioformat
MPEG	Moving Picture Experts Group
OLED	Organic Light Emitting Diode, organisches Leuchtmaterial, Display Technologie
PLL	Phase-locked loop, elektronische Schaltungsanordnung zur Frequenz-Erhöhung
RTS	Request to send, UART Signal
SCI	Serial Command Interface
SCL	Serial Clock Line, I2C Klock-Signal
SDA	Serial Data Line, I2C Daten-Signal
SDI	Serial Data Interface
SPI	Serial Peripheral Interface, ein von Motorola entwickeltes serielles Bussystem
TCP	Transmission Control Protocol, ein Netzwerkprotokoll im Internet
UART	Universal Asynchronous Receiver Transmitter, serielle, asynchrone Datenschnittstelle
URL	Uniform Resource Locator, identifiziert eine Ressource über das verwendete Netzwerkprotokoll und den Ort der Ressource in

	Computernetzen
WAV	Unkomprimiertes Audioformat
WEP-128	Wired Equivalent Privacy, ein WLAN-Verschlüsselungsalgorithmus
WLAN	Wireless Local Area Network, kabelloses, lokales Netzwerk zur Datenübertragung
WPA	WiFi Protected Access, Verschlüsselungsmethode für ein Drahtlosnetzwerk (Wireless LAN)
SSP	Synchronous Serial Port, ähnlich wie SPI mit Erweiterungen
I2C	Inter-Integrated Circuit, serieller, Adressen basierter Datenbus

23 Glossar

Wort	Beschreibung
Access Point	Elektronisches Gerät, das als Schnittstelle für kabellose Kommunikationsgeräte fungiert
Audiokomprimierung	Spezialisierte Arten der Datenkomprimierung, um digitale Audiodaten effektiv in ihrer Größe zu reduzieren
Bass	Schallwellen mit tiefen Frequenzen
Baudrate	Einheit für die Schrittgeschwindigkeit des UARTs (1 Baud ist die Geschwindigkeit, wenn 1 Symbol pro Sekunde übertragen wird)
Bitrate	Verhältnis einer Datenmenge zu einer Zeit, Einheit Bit/s
Callback-Funktion	Rückruffunktion
Debugger	Ist ein Werkzeug zum Diagnostizieren von Programmfehlern
Delta-Peak	Ladeverfahren von Akkus
Delta-Sigma	Verfahren zum Analog/Digital Wandeln
File	Eine Datei
Flow Control	Datenflusskontrolle um Verluste bei asynchronen Datenübertragungen zu verhindern
Header	Kopfdaten
High	Logisch 1 (TRUE)
HTTP Request	Eine Anfrage über das Hypertext Transfer Protocol
Key	Verschlüsselungspasswort
Low	Logisch 0 (FALSE)
MicroSD Card	Sehr kompaktes Flash-Speicherkartenformat
Pull-Down	Paralleler Widerstand auf GND
Pull-Up	Paralleler Widerstand auf VCC
Resistiven Touchscreen	Resistive Touchscreens reagieren auf Druck, der zwei elektrisch leitfähige Schichten stellenweise verbindet. Die Schichten bilden so einen Spannungsteiler, an dem der elektrische Widerstand gemessen wird, um die Position der Druckstelle zu ermitteln
Ringbuffer	
Scannen	Nach verfügbaren WLAN Netzwerken suchen
Server	
Serverroot	Stammverzeichnis des Servers
SHOUTcast	Freeware-Streaming-Server für Audiostreams/TV Streams von der Firma Nullsoft
Socket	Verbindungseinheit auf Netzwerkbasis
Stack	Stapelspeicher, für eine Datenstruktur in der Informatik
Telnet	Telecommunication Network, im Internet weit verbreitetes

	Netzwerkprotokoll
Timer	Eigenständiger Zähler im Mikrocontroller
Treble	Schallwellen mit hohen Frequenzen
Wireless	Engl. Kabellos
Interrupt	Programmunterbruch

24 Quellen

Hier werden alle Quellen, die bei dieser Arbeit verwendet wurden, genau beschrieben.

24.1 Internet

Parameter des ICY Antwort Header

Link: <http://eternity.zerbenet.org/doku.php?id=start:themen:webradiosender-suchmaschine>

Autoren: Unbekannt

Beschreibung: Detaillierte Übersicht aller Parameter im ICY Antwortheader.

Aufbau der SHOUTcast Protokolls

Link: <http://www.smackfu.com/stuff/programming/shoutcast.html>

Autoren: Scott McIntyre

Beschreibung: Aufbau des SHOUTcast Protokoll und Anleitung wie man die META-Daten herausfiltern kann.

Informationen über das MP3 Format

Link: <http://de.wikipedia.org/wiki/MP3>

Autoren: Pittimann, Darkking3, TXiKiBoT, Arkanosis

Beschreibung: Eine ausführliche Beschreibung des MP3 Formates.

Informationen über das AAC+ Format

Link: <http://de.wikipedia.org/wiki/AAC%2B>

Autoren: VolkovBot, SieBot, GrouchoBot, Aka

Beschreibung: Eine ausführliche Beschreibung des AAC+ Formates.

Anhang

Aus ökologischen Gründen sind alle Anhänge nur auf der CD vorhanden und nicht in Papierform.

- Laborjournal
- Zeitplan
- Bedienungsanleitung
- Testprotokoll
- Schemas
- Stückliste
- Pin Definitionen
- Source Code (Design Workspace)
- Doxygen Dokumentation