Hauptmenü →
A1
Willkommen im Rasputin-4 PiFace Menü (Raspberry Pi)
Favoriten →
A9
→ [1+2+3+4] → Favorite in A1 → [2+3] | Speed-Menüs | 2 | 7 | 12 | Smart
1 1live-plan-b-mit
2 1live-kassettendeck
3 1live-diggi
4 1live-klubbing
5 1live-rocker
6 1live-plan-b
7 1live-fiehe
8 1live
9 wdr2
10 wdr3
11 wdr4
12 wdr5
13 funkhaus-europa
14 fhe-radio-colonia
15 sradio.tv-mallorca-radio
16 rpr1
Streaming-Adressen
Prozesse
Mount
Inhalt-USB
Remote-Dokumentation
Remote-Code
Temp-1
Temp-x
Service
Menü-Prozesse
-Aktuell
-Übersicht
MP3-Playlist EXTERN
Menü-PDF
ip
Scripte
Album | Artist
← USB →
Titel | aktuell
Datei
USB-MP3-Playlist
USB-Playlist 1-10
Doku-PDF
Fotos
Videos
Sender | aktuell
← RADIO →
Titel | aktuell
Stream
Radiosream-Playlist
Radio-Playlist EXTERN
→ Internet - Radio ←
RADIO →
Playliste
← RADIO →
- RPR1.
-KENNY-LOGGINS---DANGER-ZONE-
→ USB - Stick ←
USB →
← USB →
1
2
3
4
1+2
3+4
1+3
1+4
2+3
Stop
Systemdaten
WLAN
Next
→ Icecream - Aufnahme ←
RECORD →
← RECORD →
→ Aktions ←
Icecream
Favoriten-Wahl B1 [2+4] →
Taste 1
Taste 2
Taste 3
Taste 4
10 Songs →
B1 [2+3]
A9
← | →
<--Auswahl
Favorite 1
A9
Favorite 2
A3
Favorite 3
A4
Favorite 4
A5

Nachdruck und/oder Vervielfältigungen,
dieser Beschreibung ist ausdrücklich gestattet.

Quellenangaben über Abbildungen in dieser Beschreibung:
- Der Tux mit dem Notebook stammt von der Webseite: http://www.openclipart.org/detail/60343

1. Auflage
Copyright 11-2013 by Dieter Broichhagen | Humboldtstraße 106 | 51145 Köln.

e-Mail: broichhagen@gmx.de
Web: http://www.broichhagen.de | http://www.illugen-collasionen.com

Satz, Layout, Screenshot´s: Dieter Broichhagen

Alle Rechte dieser Beschreibung liegen beim Autor

Gesamtherstellung: Dieter Broichhagen

Irrtum und Änderungen vorbehalten!
(Stand: 31.12.2013)

V o r w o r t

2012 gewann ich ein Buch bei einer Tombola am SFD-2012 über das ARDUINO-Board. Ich fand dies alles sehr interessant, was man mit dem ARDUINO anstellen kann. Ich konnte auch kurze Zeit später einen ARDUINO-UNO beim Franzis-Verlag für 20 € erstehen, habe aber bis heute nichts damit gemacht, weil der RaspberryPi mich voll und ganz in seinem Bann hielt und noch immer hält.

Seit Oktober 2012 beschäftige ich mich in meiner Freizeit mit diesem Minicomputer. Ich hatte auch viele Ideen im Kopf, was man so alles mit ihm anstellen könnte. Ich dachte da nicht an Garagentoröffner oder Videoballerspiele. Eher an Anwendungen wie z.B. die Sicherung einer SDRAM-Karte von der Digital-Kamera im Urlaub oder ein Internetradio, oder, oder, oder... Anwendungen, wo man keinen Monitor und keine Tastatur benötigt.

Na, jedenfalls erhielt ich in meinem Sommer-Urlaub 2013 eine Mail von Bruno H. Er machte mich darauf aufmerksam, dass ich beim O'Reilly-Verlag das Buch **„Die elektronische Welt mit Raspberry Pi entdecken“ von Erik Bartmann,** mit gut 1.000 Seiten als Rezensionsexemplar erstehen könnte. Bruno wusste, dass ich mich intensiv mit dem RaspberryPi beschäftigte und er wollte mir was gutes tun.

Ich nahm das Angebot dankbar an und erwarb ein kostenloses Rezensionsexemplar vom O'Reilly-Verlag. Das Buch ist toll geschrieben und ich kann es nur jedem weiter empfehlen. Darin erfuhr ich unter anderem von einem PiFace-Digital-Board, dass sehr großes Interesse in mir weckte. Es hatte all die Komponenten an Board, die ich für meine Ideen brauchte. LED's, Relais, Taster und Klemmleisten. Also keine Löterei mehr, sondern sofort loslegen, wie man mit Python LED's ansteuert und Tasterbedienungen abfragt.

Ich bestellte mir umgehend das Piface-Digital-Board und hatte es 2 Tage später schon in den Händen. Als nächstes brauchte ich ein neues Heim für die beiden Platinen. Ich baute mir mein bewährtes CD-Recycling-Gehäuse mit viel Glitzergedönz.

Am 26. August 2013 machte ich dann meinen ersten Schritte in der Python- und Bash-Programmierung. Bis dato hatte ich mich erfolgreich dagegen gewährt. Es war mir zu diesem Zeitpunkt nicht bewusst, auf was ich mich da einließ. Heute am 31.12.2013 habe ich schon über 800 Dateien (Scripte in Python, Bash, HTML und jede Menge grafische Elemente) produziert. Alles nur, um 4 Tasten zum Leben zu erwecken. Was für ein Aufwand. Es hat sich jedoch echt gelohnt. Es ist erstaunlich, was man mit nur 4 Tasten alles machen kann. Lassen Sie sich also überraschen.

Köln, den 31. Dezember 2013

Dieter Broichhagen

Am Anfang war der Code

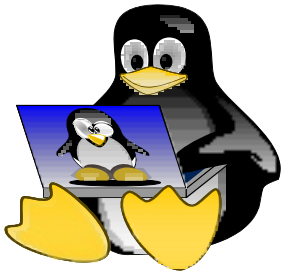
Nachdem ich mein Gehäuse fertig hatte und alle Bedingungen gem. Beschreibung für die Ansteuerung des PiFace-Digital-Board gem. o.a. Buch erfüllt wurden, probierte ich den Python-Code für die Ansteuerung der LED's und Taster aus.

Eine LED bringt man zum blinken mit:

```
0led.py
#!/usr/bin/python

# eine led
import time
import piface.pfio as pfio
pfio.init()

# endlosschleife
while True:
 pfio.digital_write(0, 1)
 time.sleep(1)
 pfio.digital_write(0, 0)
 time.sleep(1)
```



Alle Taster fragt man ab mit:

```
alletaster.py
#!/usr/bin/python

# alle taster abfragen
import time
import piface.pfio as pfio
pfio.init()

# endlosschleife
while True:
 allinputs = pfio.read_input()
 print allinputs
 time.sleep(1)
```

und einen Sprung in ein Bash-Programm bereitet man z.B. wie folgt vor:

```
favorite.py
#!/usr/bin/python

# MENUE-03
# Internetradio
# alle taster abfragen
import os
import time
import piface.pfio as pfio
```

mit [import os] erhalte ich die Möglichkeit, aus dem Python-Script in ein Bash-Programmen zu verzweigen z.B.

```
os.system("/home/dieter/scripte/str_radio01.sh ExecuteOnce &")
```

Das ist schon alles was man an Python-Code braucht, um damit eine gigantische Menüstruktur aufzubauen.

Meine Grundidee war, eine Abfrage zu erstellen, wo alle Tastenbedienungen abgefragt werden und bei Betätigung eines Tasters entsprechende LED's leuchten oder Blinken, damit man die Tastenbedienung durch die LED's bestätigt bekommt. Nach anfänglichen Schmerzen mit Python, es mag keine ää's, öö's oder üü's, hatte ich schnell die erste Abfrage für mein Grund-Menü Nummer EINS fertig (siehe nachfolgende Bash-Screen's).

<u>Starteinstellung für die LED's festlegen</u>

```
tasterled.py
#!/usr/bin/python

# MENUE-01
# Wlan-Einstellungen - Home - Android - iPhone
# alle taster abfragen
import os
import time
import piface.pfio as pfio
pfio.init()

pfio.digital_write(1,0)
pfio.digital_write(2,0)
pfio.digital_write(3,0)
pfio.digital_write(4,0)
pfio.digital_write(5,0)
pfio.digital_write(6,0)
pfio.digital_write(7,0)

pfio.digital_write(1,1)
pfio.digital_write(2,0)
pfio.digital_write(3,1)
pfio.digital_write(4,1)
pfio.digital_write(5,1)
pfio.digital_write(6,0)
pfio.digital_write(7,0)
os.system("/home/dieter/scripte/sw0.sh ExecuteOnce &")
```

Abfrage für Taster 1 festlegen

*tasterled.py

```
# endlosschleife
while True:
 allinputs = pfio.read_input()
 print allinputs
 time.sleep(1)
 if allinputs == 1:
#     print "Taster 1 gedrueckt."
     os.system("/home/dieter/scripte/wlan-home.sh ExecuteOnce &")
     os.system("/home/dieter/scripte/sw1.sh ExecuteOnce &")
     time.sleep(1)
     pfio.digital_write(1,0)
     pfio.digital_write(2,0)
     pfio.digital_write(3,0)
     pfio.digital_write(4,0)
     pfio.digital_write(5,0)
     pfio.digital_write(6,0)
     pfio.digital_write(7,0)
     time.sleep(1)
     pfio.digital_write(1,1)
     pfio.digital_write(2,0)
     pfio.digital_write(3,1)
     pfio.digital_write(4,1)
     pfio.digital_write(5,1)
     pfio.digital_write(6,0)
     pfio.digital_write(7,0)
     time.sleep(8)
```

Abfrage für Taster 2 festlegen

```
 if allinputs == 2:
#     print "Taster 2 gedrueckt."
     os.system("/home/dieter/scripte/android.sh ExecuteOnce &")
     os.system("/home/dieter/scripte/sw2.sh ExecuteOnce &")
     time.sleep(1)
     pfio.digital_write(1,0)
     pfio.digital_write(2,0)
     pfio.digital_write(3,0)
     pfio.digital_write(4,0)
     pfio.digital_write(5,0)
     pfio.digital_write(6,0)
     pfio.digital_write(7,0)
     time.sleep(1)
     pfio.digital_write(1,1)
     pfio.digital_write(2,0)
     pfio.digital_write(3,1)
     pfio.digital_write(4,0)
     pfio.digital_write(5,1)
     pfio.digital_write(6,0)
     pfio.digital_write(7,0)
     time.sleep(8)
```

Und so weiter und so fort, bis alle Tastenkombinationen die man abfragen möchte abgefragt sind.

Dabei stellt sich die Frage: „Wie viele Kombinationen sind denn mit 4 Tastern möglich?“ Nachfolgend sind alle Möglichkeiten aufgeführt.

| 1 | 2 | 3 | 4 | 1+2 | 1+3 | 1+4 | 2+3 | 2+4 | 3+4 | 1+2+3 | 2+3+4 | 3+4+1 | 1+2+4 | 1+2+3+4 |

Das sind 15 verschiedene Kombinationen. Man hätte also im Menü EINS 15 Aktionen, die man durchführen könnte. Hätte ich mich damit begnügt, dann wäre diese Beschreibung völlig überflüssig und viele Stunden Programmierarbeit ebenfalls. Ich wollte jedoch mehr, 15 Möglichkeiten waren mir einfach zu wenig.

Ich dachte mir ein System aus, dass prinzipiell unendlich erweiterbar ist. Ich glaube zwar nicht, dass es unendliche Möglichkeiten mit dem RaspberryPi gibt, es ist aber schön, dass es theoretisch möglich ist. Die Speicherkartenkapazität setzt mir die Grenzen.

Was habe ich gemacht? Nun, nachdem Menü EINS fertig war, es war nicht wirklich fertig (erkläre ich später), machte ich 16 Kopien von dem Skript. Warum 16? Es gibt 8 LED's auf dem Board, damit wollte ich die Menüpunkte darstellen.

Menü 1 – LED0 AN Rest AUS,
Menü 2 – LED0+LED1 AN Rest AUS

usw. dann

Menü 8 LED0 bis LED7 alle AN
Menü 9 LED0 AUS Rest AN
Menü10 LED0+LED1 AUS und Rest AN

usw. daher 16 Menüpunkte. Ich kam aber bei Zählung etwas in durcheinander, ist ja auch nicht so einfach bis 16 zu Zählen. Jedenfalls hat mein Menü A,17 Menüpunkte.

Diese Skripte verschaltete ich so, dass man von Skript zu Skript springen kann. Es gibt auch Abkürzungen, die es gestatten möglichst schnell zu irgend einen Menüpunkt zu gelangen. Dafür baute ich 3 Sprungmarken ein, die einem eine Abkürzung gestatten.

Zu Beginn sah die Struktur und die Bedienung noch nicht so aus, wie heute. Ich habe die Skripte mindestens gefühlte 20mal geändert. Immer wieder stellte ich fest, das sich das System unlogisch anfühlte. Das konnte ich dann nicht so bestehen lassen und ich änderte dies, im Glauben, es wäre nun besser.

Wie kann man so viele Python-Skripte für Tastenabfragen und LED-Ansteuerungen für eine Menüführung nutzen, ohne dass sich die einzelnen Skripte gegenseitig stören? Die Zauberworte lauten: **„kill und ps“** wobei ps (Prozesse ermitteln) nicht als Wort anzusehen ist**.**

Indem man den Python-Prozess per Bash-Programm ermittelt und ihn im Anschluss daran beendet mit **kill PROZESSNUMMER**. Anschließend wird einfach ein neues Python-Progamm im Bash-Programm gestartet und schon ist man im neuen Menü mit neuen

Funktionen. Das ist im Prinzip das ganze Geheimnis meines Menüs. Ich hielt es für eine gute Idee, das ganze noch mal zu verdoppeln, indem ich die Skripte in ein Unterverzeichnis kopierte und dann miteinander verschachtelte. So hatte ich jetzt 34 Menüs zur Verfügung. Je Menüpunkt stehen mir einige Tastenkombinationen nicht zur freien Verfügung, da diese für die Navigation im Menü erforderlich sind. Es bleibt jedoch bei den 34 Menüs genug übrig. Nach Abzug der Navigations-Tastenkombinationen bleiben immerhin 80 frei verfügbare Menüpunkte übrig. Kopiert man die ganze Struktur in ein weiteres Verzeichnis und verknüpft diese miteinander, erhält man schon 64 Menüs mit 160 Funktionsmöglichkeiten. Das könnte man dann immer so weiter betreiben bis zur Unendlichkeit.

Wir kommen zu einigen Besonderheiten.

Wie stelle ich fest, ob ich im Menü A oder im Menü B bin?

LED-7 hat einen Sonderstatus. Die Blinkt lustig vor sich hin. Was Blinkt sie denn? Sie wird über das Skript startled.py angesteuert, dass beim booten des Rasputin-3 automatisch startet und über einen Cronjob gesteuert wird. In der Datei startled.py wurde auch eine Zeitverzögerung von 50 Sekunden eingebaut. Erst nach dieser Zeit blinkt sie. Dann laufen alle Dienste, um sich per ssh anmelden zu können. Ab diesem Zeitpunkt kann man mit Rasputin-3 arbeiten.

Wer Morsecode beherrscht, ich war beider Marine Tastfunker, der wird feststellen, dass im Menü A diese LED ständig A TUX blinkt und im Menü B B TUX. Damit kann ich also immer sehen wo ich mich im Menü befinde. LED 7 sagt mir A oder B und die restlichen LED's zeigen mir Menü 1 oder 2 oder 3 etc. an.

Die Tastenkombination 1+3 (Wo bin ich?) erlaubt mir folgende Anzeige z.B. für Menü A4:
Einleitung der neuen LED-Anzeige:

- alles AUS: X X X X X
- alles EIN: X X X X X
- Buchstabe A: X X
- Zahl 4: X X X X X
- alles EIN: X X X X X
- Buchstabe A: X X
- Zahl 4: X X X X X

Diese Tastenkombination 1+3 kann ich überall im Menü verwenden um festzustellen, in welchem Menü man sich gerade befindet.

Menü A1 dient zur Festlegung des WLAN-Point's. Wo connectet sich Rasputin-3?

- Taste 1 = WLAN-Home und Systemdaten Aktualisierung (Erklärung folgt)
- Taste 2 = Android-Smartphone
- Taste 3 = iPhone-Smartphone

Außerdem kann man im Menü A1 das System mit 1+2 runter fahren bzw. mit 3+4 rebooten. Tastenbedienung 2+3 erlaubt direkten Sprung in das Favoritenmenü, dass man in allen Menüpunkten mit Ausnahme der Sprungmenüs mit der Tastenbedienung 1+2+3+4 festlegen kann. Befinde ich mich z.B. im Menü 3 und drücke die Tasten 1+2+3+4 gleichzeitig und warte die LED-Anzeige ab, bis alles aufleuchten, dann wird dieses Menü in das Favoritenmenü kopiert, dass sich wie oben beschrieben aufrufen lässt. Es ist außerdem beim booten des Rasputin-3 das Startmenü. Das habe ich gemacht, damit man sich nicht durch die Struktur hangeln muss, um das gewünschte Lieblings-Menü aufzurufen. Der Start des Menüs wird über einen Cronjob beim booten aktiviert.

Die Tastenbedienung 1+4 löst eine Systembereinigung aus, es wird dann alles in Grundstellung gebracht. Wird weiter unten beschrieben.

Die Taste 4 bringt mich in Menü 2, dass schon ein Sprungmenü ist. Von hier komme ich nach: 3 | 4 | 5 | 6 | 7 | 12. Menüs 7 und 12 sind die nächsten Sprungmenüs.

Mit Ausnahme der Sprungmenüs komme ich immer mit den Tasten 1+2 zum Menü A1 und mit den Tasten 3+4 in Menü B2. Menü B2 ist wiederum ein Sprungmenü analog zu Menü A2 jedoch in der Menüstruktur B. Wenn wir schon bei B sind, kommen wir erst einmal zu einem ganz besonderen Menü.

Menü B1 - Favoriten-Programmierung
Dieses Menü hat mich ein paar graue Haare gekostet. Mein Anspruch war, ein Menü zu basteln, dass wenigstens 4 Menüpunkte besitzt die frei programmierbar sind. Außerdem sollte man auch die Wahlmöglichkeit haben als Startmenü den zuletzt gewählten Favoriten oder das Menü B1 als Startmenü festzulegen. Nach einigen Programmierschmerzen habe ich folgende Funktionen eingebaut.

Nach B1 komme ich zunächst nur über Menü A1 mit der Tastenkombination 2+4. Menü B1 bietet mir folgende Möglichkeiten:

- 1 = Favorit 1 SAVE / GO
- 2 = Favorit 2 SAVE / GO
- 3 = Favorit 3 SAVE / GO
- 4 = Favorit 4 SAVE / GO
- 1+2 = MENÜ A1
- 3+4 = Programmier-Modus SAVE-EIN
- 1+4 = Programmier-Modus SAVE-AUS
- 2+3 = Startmenü → FAVORITEN-MENÜ
- 2+4 = Startmenü → FAVORITEN-AUSWAHL B1
- 1+3 = Menü-Anzeige (Wo bin ich?)

Nun die Erklärungen zu den Menüpunkten. Mit 3+4 = Programmier-Modus SAVE-EIN schaltet man den Programmier-Modus ein und dient dazu, dass aktuelle Favoriten-Menü mit Favorit 1 SAVE / GO bis Favorit 4 SAVE / GO zu sichern. Drückt man z.B. nach der Aktivierung von SAVE-EIN die Taste 1, dann wird das zuletzt gespeicherte Favoritenmenü auf die Taste 1 abgespeichert. Dies kann man dann mit den Tasten 2, 3 und 4 so

fortsetzten. Will man sich Favoriten-Menüs unter Menü B1 anlegen, ist folgendes zu beachten. Nachdem ein Favorit mit den Tasten 1+2+3+4 festgelegt wurde, sagen wir das Menü A3, muss man anschließend direkt über Menü A1 und der anschließenden Tastenkombinationen 2+4 ins Menü B1 wechseln, den Programmiermodus über die Tasten 3+4 aktivieren und dann sofort die Wunschtasten 1, 2, 3 oder 4 drücken um das Favoriten-Menü zu sichern. Es ist wichtig so vorzugehen, weil das System sonst den Zustand des zuletzt gewählten Favoriten verliert. Leider ist es mir nicht gelungen, dies zu vereinfachen.

Nachdem man sich den Favoriten im Menü B1 angelegt hat, drücke man bitte die Tasten 1+4 Programmier-Modus SAVE-AUS um den Programmiermodus zu verlassen. Nun kann man von B1 über die Tasten 1, 2, 3 und 4 die Lieblingsmenüs direkt aufrufen.

Mit den Tasten 2+3 legt man fest, dass der zuletzt gewählte Favorit als Startmenü gilt.

Mit den Tasten 2+4 legt man fest, dass Menü B1 als Startmenü gilt.

Das Startmenü ist das erste Menü, dass nach dem hochfahren des Rasputi-3 bedient werden kann. Ich finde die Arbeit hat sich gelohnt. Nun hat man eine große Flexibilität mit den Menüs und die freie Auswahl, welches Menü nach dem booten des Computers als erstes zur Verfügung stehen soll.

Menü A3
Das Menü enthält die erste Anwendung. Hier läuft der Mplayer und streamt Internetradio. Ich habe mal 16 Sender fest verdrahtet. Wenn man im Menü 3 ist und tippt auf die Taste 1, dann startet der 1. Sender **„1live-plan-b-mit“**. Tippt man auf die Taste 2 wird der nächste Sender ausgewählt **„1live-kassettendeck“**. Tippt man auf Tastet 3 wird wieder der Sender zuvor abgespielt. Manchmal spielt Rasputin-3 nicht mehr den Stream ab. Abhilfe schafft Taste 1. Einmal antippen und schon spielt der Sender kurze Zeit später wieder. Mit den Tasten 1+4 lässt sich ein Favoritensender speichern, den man dann mit den Tasten 1+3 aufrufen kann. Die Tasten 2+4 gestatten einen Sprung nach dem Aufnahmeprogramm „icecream“, das den zur Zeit laufenden Radiostream als MP3-Files auf die Speicherkarte ablegt, hierzu später mehr. Die Tasten 2+3 beenden den Mplayer. Taste 4 bringt uns ins nächste Menü Nr. A4.

Menü A8
Das Menü habe ich für „icecream“ vorgesehen. Das Programm streamt Radiosender und legt sie als einzelne Files in Form einer MP3 ab. Da icecream die Möglichkeit bietet:

- Anzahl Songs
- Speichergröße in MB
- Aufnahme in Minuten

habe ich das mit den Tasten:

- 1 = 10 Songs
- 2 = 100 MB
- 3 = 90 Minuten

berücksichtigt. Drückt man auf z.B. Taste 1, dann werden 10 Songs vom laufenden Radiostream aufgezeichnet. Taste 3 würde 90 Minuten lang die Aufnahmen laufen lassen, je Song jeweils eine Datei. Die Tasten 2+3 stoppt icecream und die Tasten 2+4 erlauben mir einen Rücksprung zum Menü 3 für den Mplayer.

Menü A4
Das Menü spielt MP3-Songs von einem externen Speichermedium, z.B. USB-Stick, ab. Nachdem man den Stick angeschlossen hat, muss dieser erst einmal auf MP3-Songs durchsucht werden. Dies erfolgt mit den Tasten 1+3+4. Jetzt wird der USB-Stick gemountet (in den Dateibaum eingehäng) dann eine Datenbank mit dem Inhalt des Sticks erzeugt und komplett nach MP3-Songs durchsucht. Danach wird eine Playlist erstellt. Dies kann gut 2 Minuten dauern. Bei Betätigung von 1+3+4 verdunkeln erst alle LED's und anschließend leuchtet nur noch eine in der Mitte. Nach erfolgreicher Anlage der Playlist leuchten dann wieder die ersten 4 LED's. Jetzt kann man mit Taste 1 die Playliste starten, die dann nacheinander abgearbeitet wird. Mit der Taste 2 wählt man den nächsten Song und mit Taste 3 den Song davor. Mit 2+3 stoppt man den Mplayer und mit 1+3 wird eine Zufallswiedergabe gestartet (vor und zurück blättern geht dann nicht). Mit den Tasten 1+2+3 kann man eine externe Playliste, die natürlich aus der aktuellen Playliste stammen muss, über einen USB-Stick importieren. Dies hat den Vorteil, dass man den Ablauf der Playliste bestimmen kann. An die aktuell erzeugte Playliste kommt man über eine Schnittstelle, die weiter unten beschrieben wird. Mit den Tasten 2+4 sowie 1+4 kann man dann wählen, ob die externe oder interne Playliste vom USB-Stick abgespielt werden soll.

Menü A5
Das Menü dient dazu, ein externes Speichermediums zu mounten und der Sicherung aller Scripte in ./scripte sowie HTML-Dateien in /var/www.

Menü A6
Das Menü ist noch nicht umgesetzt.

Menü A9
Das Menü ist eine zweite Variante des Internetradios. Hier werden nicht die fest verdrahteten Sender abgespielt, sondern Mplayer streamt die Sender anhand einer Playliste. Eine habe ich intern im Verzeichnis ../scripte/playlisten liegen. Man kann aber auch eine externe Playliste importieren. Diese muss zwingend playliste-radio.txt lauten. Diese Art des Abspielens vom Internetradio hat den Vorteil, dass man schnell zwischen Sendern hin und her blättern kann. Mplayer managt das anhand der Playliste selber und man muss nicht zusätzliche Scripte schreiben, die das ermöglichen wie im Menü A3.

Nachdem ich nun viele Worte über das Menü gemacht habe, folgt nun was ganz besonderes. Um die Langeweile etwas zu erleuchten kommt jetzt der Superknaller.

Wellnessbereich

Es gibt auch einen Wellnessbereich der sehr gesprächig ist. Wenn man mit dem Smartphone, Tablett oder PC die IP-Adresse des Rasputin-3 im Webbrowser aufruft, dann erscheint ein Display, dass einem immer anzeigt, in welchem Menü ich mich gerade befinde. Links finde ich Statusanzeigen und rechts die gesamte Menüstruktur. Das Linke Fenster wird alle 5 Sekunden aktualisiert und das rechte Fenster alle 20 Sekunden. Im linken Fenster wird oben das aktuelle Menü angezeigt sowie ob ich mich im Hauptmenü oder im Favoritenmenü befinde. Im Fenster wird weiterhin angezeigt, welche Taste zuletzt gedrückt und welches Programm aktiviert wurde, Mplayer, icecream, USB-Stick etc. Links unten findet man einen Screenshot über Systemdaten. Das ist die **Liveanzeige von conky**, direkt vom VNC-Desktop des Rasputin-3. Diese lässt sich jederzeit im Menü 1 mit der Taste 1 aktualisieren und wird per Cronjob alle 5 Minuten aktualisiert. Das rechte Fenster mit der Menüstruktur zeigt oben immer das aktuelle Menü an. Nachfolgend einige Screenshots vom Wellnesbereich und Erklärungen dazu.

Display und Menüführung im Webbrowser

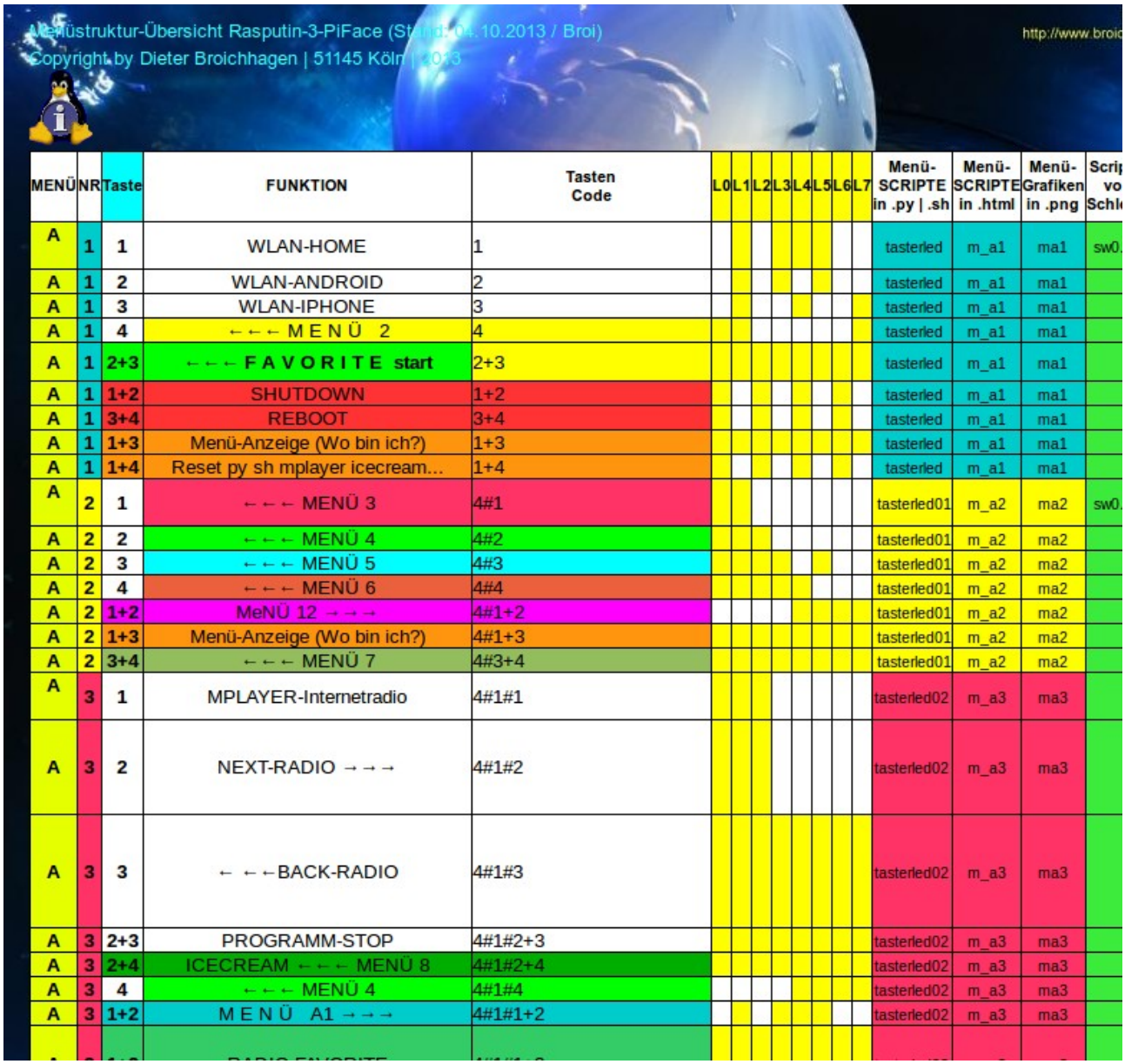
A	9	1	1		MPLAYER-Internetradio Playlistenreihenfolge	4#rechts#2#1	tasterled08	m_a9 ma9	
A	9	2	2		NEXT-RADIO → → →	4#rechts#2#2	tasterled08	m_a9 ma9	
A	9	3	3		← ← ← BACK-RADIO	4#rechts#2#3	tasterled08	m_a9 ma9	
A	9	2+3	push		PROGRAMM-STOP	4#rechts#2#push	tasterled08	m_a9 ma9	
A	9	2+4		prüfen	Switch Playliste extern		tasterled08	m_a9 ma9	
A	9	1+4		prüfen	Switch Playliste intern		tasterled08	m_a9 ma9	
A	9	1-3		prüfen	Playliste extern IMPORT	4#rechts#2#1+2+3	tasterled08	m_a9 ma9	
A	9	4	4		← ← ← MENÜ 10	4#rechts#2#4	tasterled08	m_a9 ma9	
A	9	1+2	rechts		M E N Ü A1 → → →	4#rechts#2#rechts	tasterled08	m_a9 ma9	
A	9	1-4	5		FAVORITE ANLEGEN	4#rechts#2#5	tasterled08	m_a9 ma9	
A	9	3+4	links		M E N Ü B2 → → →	4#rechts#2#links	tasterled08	m_a9 ma9	
A	10	1	1			4#rechts#3#1	tasterled09	m_a10 ma10	
A	10	2	2			4#rechts#3#2	tasterled09	m_a10 ma10	
A	10	3	3			4#rechts#3#3	tasterled09	m_a10 ma10	
A	10	4	4		← ← ← MENÜ 11	4#rechts#3#4	tasterled09	m_a10 ma10	
A	10	1+2	rechts		M E N Ü A1 → → →	4#rechts#3#rechts	tasterled09	m_a10 ma10	
A	10	1-4	5		FAVORITE ANLEGEN	4#rechts#3#5	tasterled09	m_a10 ma10	
A	10	3+4	links		M E N Ü B2 → → →	4#rechts#3#links	tasterled09	m_a10 ma10	
A	11	1	1			4#rechts#4#1	tasterled10	m_a11 ma11	
A	11	2	2			4#rechts#4#2	tasterled10	m_a11 ma11	
A	11	3	3			4#rechts#4#3	tasterled10	m_a11 ma11	
A	11	4	4		← ← ← MENÜ 12	4#rechts#4#4	tasterled10	m_a11 ma11	
A	11	1+2	rechts		M E N Ü A1 → → →	4#rechts#4#rechts	tasterled10	m_a11 ma11	
A	11	1-4	5		FAVORITE ANLEGEN	4#rechts#4#5	tasterled10	m_a11 ma11	
A	11	3+4	links		M E N Ü B2 → → →	4#rechts#4#links	tasterled10	m_a11 ma11	
A	12	1	1		← ← ← MENÜ 13	4#links#1	tasterled11	m_a12 ma12	sw0.sh
A	12	2	2		← ← ← MENÜ 14	4#links#2	tasterled11	m_a12 ma12	
A	12	3	3		← ← ← MENÜ 15	4#links#3	tasterled11	m_a12 ma12	
A	12	4	4		← ← ← MENÜ 16	4#links#4	tasterled11	m_a12 ma12	
A	12	1+2	rechts		MENÜ 7 → → →	4#links#rechts	tasterled11	m_a12 ma12	
A	12	3+4	links		M E N Ü 2 → → →	4#links#links	tasterled11	m_a12 ma12	
A	13	1	1			4#links#1#1	tasterled12	m_a13 ma13	
A	13	2	2			4#links#1#2	tasterled12	m_a13 ma13	
A	13	3	3			4#links#1#3	tasterled12	m_a13 ma13	
A	13	4	4		← ← ← MENÜ 14	4#links#1#4	tasterled12	m_a13 ma13	
A	13	1+2	rechts		M E N Ü A1 → → →	4#links#1#rechts	tasterled12	m_a13 ma13	
A	13	1-4	5		FAVORITE ANLEGEN	4#links#1#5	tasterled12	m_a13 ma13	
A	13	3+4	links		M E N Ü B2 → → →	4#links#1#links	tasterled12	m_a13 ma13	
A	14	1	1			4#links#2#1	tasterled13	m_a14 ma14	
A	14	2	2			4#links#2#2	tasterled13	m_a14 ma14	
A	14	3	3			4#links#2#3	tasterled13	m_a14 ma14	
A	14	4	4		← ← ← MENÜ 15	4#links#2#4	tasterled13	m_a14 ma14	
A	14	1+2	rechts		M E N Ü A1 → → →	4#links#2#rechts	tasterled13	m_a14 ma14	
A	14	1-4	5		FAVORITE ANLEGEN	4#links#2#5	tasterled13	m_a14 ma14	
A	14	3+4	links		M E N Ü B2 → → →	4#links#2#links	tasterled13	m_a14 ma14	
A	15	1	1			4#links#3#1	tasterled14	m_a15 ma15	
A	15	2	2			4#links#3#2	tasterled14	m_a15 ma15	
A	15	3	3			4#links#3#3	tasterled14	m_a15 ma15	
A	15	4	4		← ← ← MENÜ 16	4#links#3#4	tasterled14	m_a15 ma15	
A	15	1+2	rechts		M E N Ü A1 → → →	4#links#3#rechts	tasterled14	m_a15 ma15	
A	15	1-4	5		FAVORITE ANLEGEN	4#links#3#5	tasterled14	m_a15 ma15	
A	15	3+4	links		M E N Ü B2 → → →	4#links#3#links	tasterled14	m_a15 ma15	

Display im Webbrowser (linker Frame)

Die grünen Punkte zeigen an, welche Menüpunkte zur Zeit aktiv sind und welche Taste zuletzt gedrückt wurde. In diesem Beispiel ist das Favoritenmenü A9 aktiv, es wird Internetradio gestreamt anhand der Playliste. Der linke Frame wird alle 5 Sekunden aktualisiert. Ein Klick auf Scripte und schon ist man im Verzeichnis der Skripte. Hier kann man sich alle Script-Inhalte ansehen. Man kann sich auch die laufenden Prozesse, die Playlisten, diese Dokumentation, die Menüstruktur oder Prozessor-Temperatur ansehen. Ein Klick auf Service zeigt einem alles noch ausführlicher in Frames an.

Menüstruktur im Webbrowser (rechter Frame)

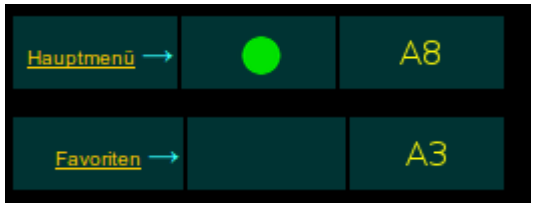
Das oberste Menü in der Anzeige vom Browser (ohne es zu scrollen) zeigt automatisch das aktuelle Menü an. Der rechte Frame wird alle 20 Sekunden aktualisiert. Alle 5 Sekunden wäre sehr nervig. Diese Einstellung ist nur ein Kompromiss. Würde ich diesen Automatismus abstellen, dann müsste man immer manuell das Fenster aktualisieren. Will man sich Menüstruktur in aller Ruhe ansehen, ohne das Gezappel alle 20 Sekunden, dann ruft man besser die Übersicht auf. Diese wird im neuen Fenster dargestellt und man bekommt eine ruhige Anzeige. Der Tasten-Code gibt an, was zu drücken ist, um zum Menü zu gelangen. Startpunkt hierfür ist immer Menü 1 (nicht das Favoriten-Menü).



MENÜ	NR	Taste	FUNKTION	Tasten Code	L0	L1	L2	L3	L4	L5	L6	L7	Menü-SCRIPTE in .py | .sh	Menü-SCRIPTE in .html	Menü-Grafiken in .png	Scrip vo Schl
A	1	1	WLAN-HOME	1									tasterled	m_a1	ma1	sw0.
A	1	2	WLAN-ANDROID	2									tasterled	m_a1	ma1	
A	1	3	WLAN-IPHONE	3									tasterled	m_a1	ma1	
A	1	4	←←← M E N Ü 2	4									tasterled	m_a1	ma1	
A	1	2+3	←←← F A V O R I T E start	2+3									tasterled	m_a1	ma1	
A	1	1+2	SHUTDOWN	1+2									tasterled	m_a1	ma1	
A	1	3+4	REBOOT	3+4									tasterled	m_a1	ma1	
A	1	1+3	Menü-Anzeige (Wo bin ich?)	1+3									tasterled	m_a1	ma1	
A	1	1+4	Reset py sh mplayer icecream...	1+4									tasterled	m_a1	ma1	
A	2	1	←←← MENÜ 3	4#1									tasterled01	m_a2	ma2	sw0.
A	2	2	←←← MENÜ 4	4#2									tasterled01	m_a2	ma2	
A	2	3	←←← MENÜ 5	4#3									tasterled01	m_a2	ma2	
A	2	4	←←← MENÜ 6	4#4									tasterled01	m_a2	ma2	
A	2	1+2	MeNÜ 12 →→→	4#1+2									tasterled01	m_a2	ma2	
A	2	1+3	Menü-Anzeige (Wo bin ich?)	4#1+3									tasterled01	m_a2	ma2	
A	2	3+4	←←← MENÜ 7	4#3+4									tasterled01	m_a2	ma2	
A	3	1	MPLAYER-Internetradio	4#1#1									tasterled02	m_a3	ma3	
A	3	2	NEXT-RADIO →→→	4#1#2									tasterled02	m_a3	ma3	
A	3	3	← ←←BACK-RADIO	4#1#3									tasterled02	m_a3	ma3	
A	3	2+3	PROGRAMM-STOP	4#1#2+3									tasterled02	m_a3	ma3	
A	3	2+4	ICECREAM ←←← MENÜ 8	4#1#2+4									tasterled02	m_a3	ma3	
A	3	4	←←← MENÜ 4	4#1#4									tasterled02	m_a3	ma3	
A	3	1+2	M E N Ü A1 →→→	4#1#1+2									tasterled02	m_a3	ma3	

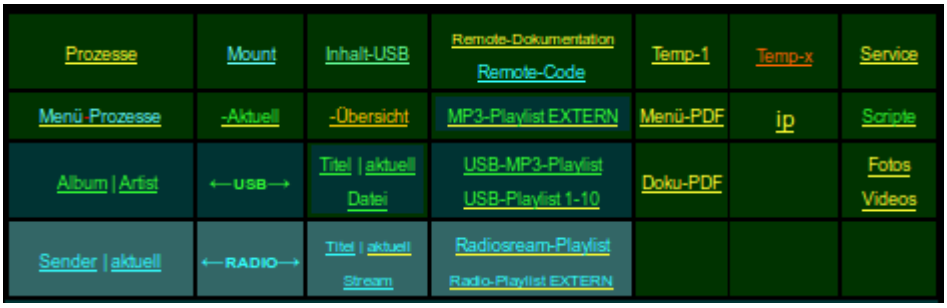
Aktuelles Menü

Hier wird das aktuelle Menü angezeigt und ob man sich im Hauptmenü oder im Favoritenmenü befindet.



Info-Bereich

Hier kann man sich die laufenden Prozesse von Rasputin-3 ansehen, ob ein USB-Stick gemountet ist, den Inhalt des USB-Stick's, welche MP3-Files es auf dem USB-Stick gibt (Playliste), aktuelle Temperatur des ARM-Prozessors, alles zusammen in mehreren Framefenstern über Service, den Morsecode etc.. Hier ist noch viel Platz für weitere Infos.



Aktivitäten-Anzeigen

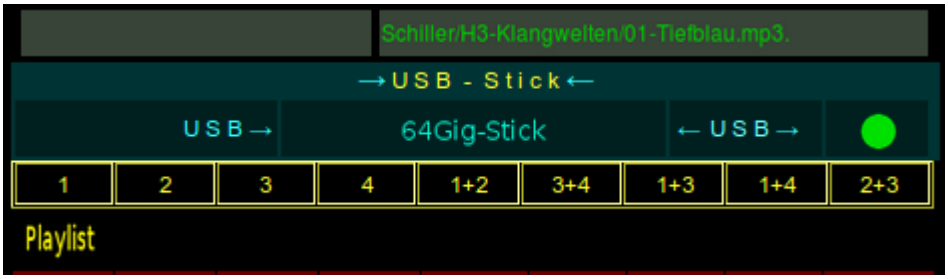
Hier erfährt man, welcher Radiosender gewählt wurde, ob ein USB-Stick aktiviert wurde, welche Taste gedrückt wurde, welcher Accesspoint gewählt wurde, selbst ein shutdown oder reboot wird angezeigt.

Anzeige Internetradio

Zur Zeit wird rpr1 abgespielt mit -KENNY-LOGGINS---DANGER-ZONE-

Anzeige welcher USB-Stick angeschlossen ist

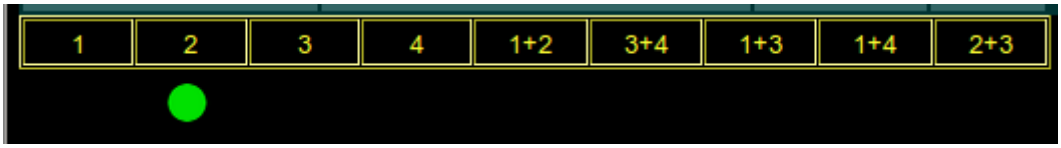
Kommt nur, wenn der Mplayer Musik vom Stick abspielt. Hier spielt gerade Schiller.



Statusanzeigen

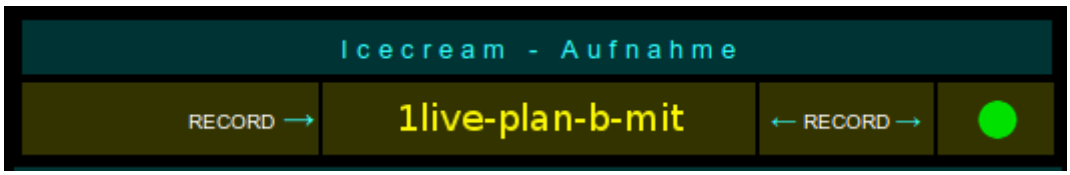
Für WLAN-POINT'S, reboot, shutdown und ob das Serviceprogramm zur Zeit läuft

Anzeige, welche Tastenbedienung erfolgte



Senderliste

Anzeige für Icecream



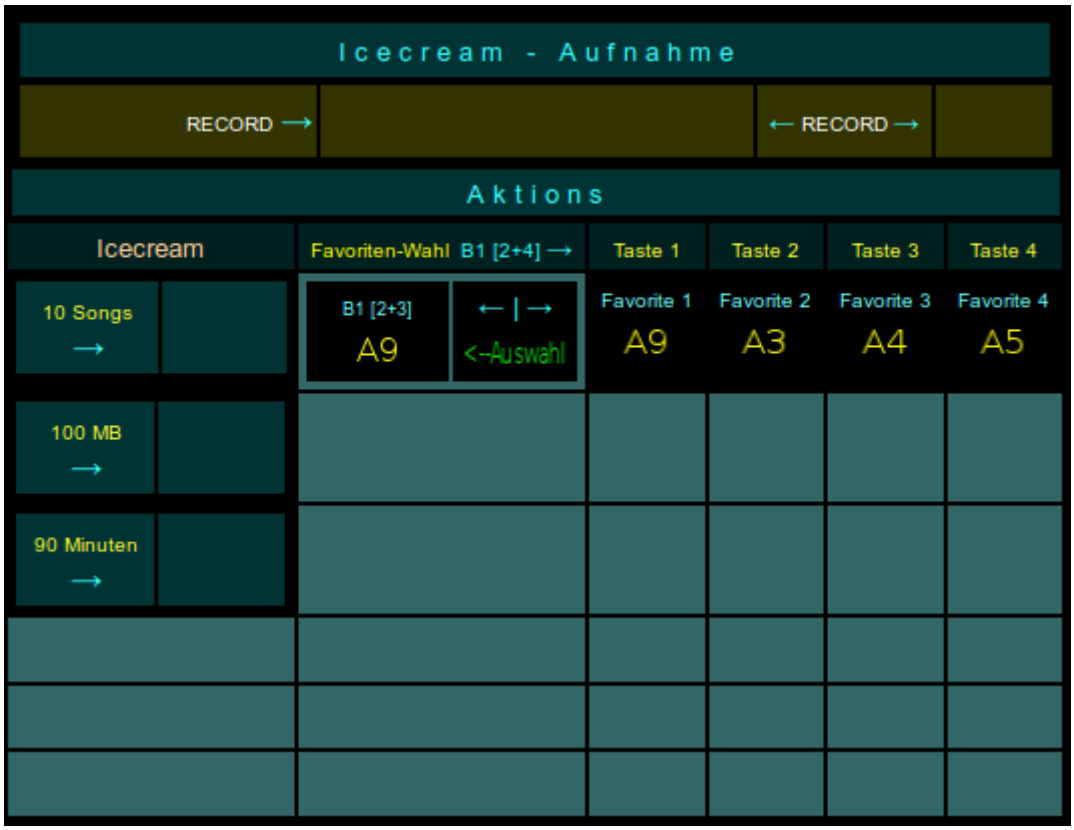
Favoriten-Auswahl

Das Feld B1 [2+3] mit A9 zeigt einem an, welches das letzte Favoritenmenü war.
Das Feld ← | → mit dem Inhalt <--Auswahl zeigt einem mit dem Pfeil an, welches Menü als Startmenü deklariert wurde (hier links Menü A9). Würde das Feld ← | → Auswahl--> anzeigen, dann wäre Menü B1, das Startmenü (hier mit den Favoritenmenüs: A9, A3, A4, A5).

Selbstverständlich wird das aktuelle Menü im rechten Frame oben im Webbrowser an erster Stelle angezeigt. So ist man schnell im Bilde, in welchem Menü man sich gerade befindet.

Aktions-Bereich

Es ist ebenfalls noch viel Platz für weitere Menü-Infos zum Ausbauen.
Hier z.B mit der Anzeige von Icecream ob 10 Songs, 100 MB oder 90 Minuten aufgezeichnet wird. Oder die programmierten Favoriten-Menüs.



Systemdaten-Anzeige

Dies ist die Anzeige von conky, ein Programm, das im virtuellen Display vom VNC-Server läuft. Es ist eine Liveanzeige, die im Menü A1 mit Taste 1 aktualisiert werden kann. Er wird per Cronjob alle 5 Minuten aktualisiert.

Der „Wo-bin-ich-Code“

Durch die gleichzeitige Betätigung der Tasten 1+3 bekommt man im Morsecode angezeigt, in welchem Menü man sich zur Zeit befindet. In der u.a. Tabelle ist der Anzeige-Code dargestellt. LED AN für kurzer Morsecode und LED AUS für langer Morsecode. Ich dachte mir, dass diese Darstellung besser zu erkennen ist, als die Blink-LED 7, die mir ja auch ständig sagt, ob ich im Menü A oder im Menü B bin.

Rasputin-3-PiFace-Menü

CODE Tabelle	L0	L1	L2	L3	L4	L5	L6	L7	Bemerkungen
A									Zeichen am Morsecode angelehnt
B									Farbe gelb bedeutet did (kurz)
C									Farbe weiß bedeutet da (lang)
0									Da es nur 8 Led's gibt musste ich
1									einen Kompromis finden. Daher
2									habe ich am Ende die letzten Led's
3									leuchten lassen, damit man die
4									daa's (lang) auch noch am Ende erkennt
5									
6									
7									
8									
9									

A								
1								

Anzeige 2 Sekunden dann Wechsel nach 1
hier wieder 2 Sekunden dann alles aus und
5 Sekunden Pause dann alles noch mal wiederholen

A								
2								

A								
3								

A								
4								

A								
5								

Rasputin-3 mit PiFace-Board

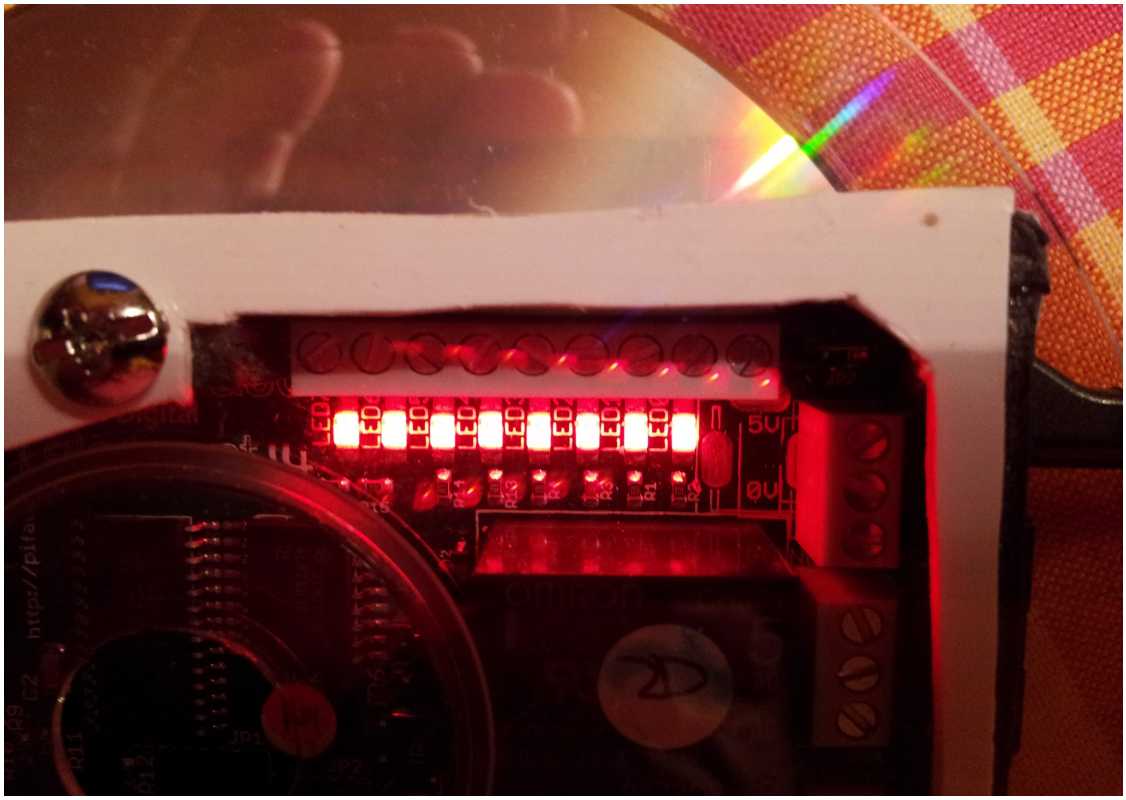
In meinem CD-Designer-Gehäuse Marke **„Selvmade“.**

Tasten

Unscheinbar aber gigantische Funktionen. Taste 1 links bis Taste 4 rechts.

8 LED's

LED0 rechts bis LED7 links. LED7 blinkt ständig lustig vor sich hin (A TUX oder B TUX).



Rasputin-3 starten

Nach ca. 2 Minuten Startzeit landet man immer im Favoritenmenü. So kann man sich gleich mit diesem Menü beschäftigen. Gesteuert wird dies im Cronjob. Da habe ich festgelegt, welches Menü zuerst gestartet werden soll. Soll der Access-Point geändert werden, an dem sich Rasputin-3 anmeldet, dann betätigt man einfach die Tasten 1+2 gleichzeitig und landet so im Hauptmenü A1 und erhält folgende LED-Anzeige, wenn man von oben auf den Rasputin-3 schaut und die LED's oben rechts und die Tasten unten links sind:

dauer blinkt	LED7	LED6	LED5	LED4	LED3	LED2	LED1	LED0
WLAN								
ANDROID								
iPHONE								

Der WLAN-USB-Stick N150 von Netgear muss vor dem Starten eingesteckt sein, damit sich Rasputin-3 im Netz connecten kann. Gesteuert wird das ganze über die Skripte.

- wlan-home.sh
- android.sh
- iphone.sh

Der Inhalt eines Skriptes sieht z.B. bei wlan-home.sh so aus:

wlan-home.sh

```
#!/bin/sh

# wlan per wlan-home

sudo kill -9 `ps -ef | grep startled.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep startled.sh | grep -v grep | cut -c10-14`

sleep 1

# sudo cp /etc/network/android /etc/network/interfaces
# sudo cp /etc/network/iphone /etc/network/interfaces
# sudo cp /etc/network/wlan /etc/network/interfaces
sudo cp /etc/network/wlan-home /etc/network/interfaces

sudo cp /home/dieter/scripte/startled-wlan.py /home/dieter/scripte/startled.py

sudo cp /home/dieter/scripte/sw-wlan.png /var/www/sw-wlan.png
sudo cp /home/dieter/scripte/sw-leer.png /var/www/sw-android.png
sudo cp /home/dieter/scripte/sw-leer.png /var/www/sw-iphone.png


sudo cat /root/wlan.txt > /home/pi/wlaninfo.txt
sudo cat /root/wlan.txt > /home/dieter/wlaninfo.txt

# vns-snapshot machen

sleep 1

vncsnapshot -passwd ~/.vnc/passwd -rect 640x0+600+30 :1 vnc-screen.jpeg < /dev/null &
sleep 3
sudo cp /home/dieter/vnc-screen.jpeg /var/www/screen/vnc-screen.jpeg
sleep 1

# sudo python /home/dieter/scripte/startled.py
# sleep 1
```

Über die Tasten 1 bis 3 im Menü A1 stellt man den Access-Point ein, wo Rasputin-3 sich anmelden soll. Die Zugangsdaten befinden sich in der Datei /etc/network/interfaces, die ich fest verdrahtet habe.

Die Taste 1 im Menü A1 hat eine Doppelfunktion. Zum einen schaltet man auf WLAN um und gleichzeitig werden mit dieser Taste die Systemdaten-Anzeige (Screenshot von conky) aktualisiert.

Die Tastenkombination 1+4 startet ein Service-Programm das viel erledigt. Es werden alle Menüscripte (Bash-Programme und Python-Programme) beendet, alle Radiosender für Mplayer und Icecream werden neu sortiert, dass sie der Liste im Web-Display entsprechen. Die Datenbank für locate wir aktualisiert, alle Grafik-Symbole im Webdisplay werden zurückgesetzt. Inhalte vom USB-Stick ausgelesen, MP3-Files auf dem Stick ermittelt, Prozessliste neu ermittelt. VNC-Server neu starten und installierte Pakete ermitteln. Das Programm sorgt für Ordnung und bringt alles in einen definierten Grundzustand. (Siehe Bash-Programm Stand: 16.11.2013)

_____service.sh_____

```
#!/bin/sh

# service  starten

sudo updatedb &
sudo ifconfig wlan0 | grep "inet Adresse:" | awk '{ print $2 }' | awk -F ':' '{ print $2 }' > /home/dieter/scripte/prozesse/ip
sudo cp /home/dieter/scripte/prozesse/ip /var/www/ip

sudo cp /home/dieter/scripte/sw-service.png /var/www/sw-service.png

sudo cp /home/dieter/scripte/sw-leer.png /var/www/aktive.png
sudo cp /home/dieter/scripte/sw-green.png /var/www/aktive-favorite.png

sudo cp /home/dieter/scripte/sw-leer.png /var/www/sw-wlan.png
sudo cp /home/dieter/scripte/sw-leer.png /var/www/sw-android.png
sudo cp /home/dieter/scripte/sw-leer.png /var/www/sw-iphone.png
sudo cp /home/dieter/scripte/sw-leer.png /var/www/sw-shutdown.png
sudo cp /home/dieter/scripte/sw-leer.png /var/www/sw-reboot.png
sudo cp /home/dieter/scripte/sw-leer.png /var/www/sw1.png
sudo cp /home/dieter/scripte/sw-leer.png /var/www/sw2.png
sudo cp /home/dieter/scripte/sw-leer.png /var/www/sw2_3.png
sleep 2

sudo cp /var/www/menue_favorite.html /var/www/menue_aktuell.html

sudo kill -9 `ps -ef | grep tasterled.py | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled01.py | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled02.py | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled03.py | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled04.py | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled05.py | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled06.py | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled07.py | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled08.py | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled09.py | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled10.py | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled11.py | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled12.py | grep -v grep | cut -c10-14`
```

```
sudo kill -9 `ps -ef | grep tasterled13.py | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled14.py | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled15.py | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled16.py | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep favorite.py | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep favorite-a.py | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep favorite-b.py | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep favorite-c.py | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep favorite-d.py | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep favorite-go.py | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep favorite-sic.py | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep favorite-save.py | grep -v grep | cut -c10-14`
sleep 1

sudo kill -9 `ps -ef | grep tasterled.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled01.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled02.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled03.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled04.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled05.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled06.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled07.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled08.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled09.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled10.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled11.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled12.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled13.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled14.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled15.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep tasterled16.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep wlan.sh.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep wlan-home.sh.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep android.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep iphone.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep ice01.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep killicecream.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep killmplayer.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep killtasterled01.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep mplay01.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep mplay02.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep mplay-step-up.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep mplay-step-down.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep radiostream.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep speed02.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep speed07.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep speed12.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep favorite-go-a.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep favorite-go-b.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep favorite-go-c.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep favorite-go-d.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep favorite-go-save.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep favorite-save-a.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep favorite-save-b.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep favorite-save-c.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep favorite-save-d.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep favorite-save.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep favorite-go.sh | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep mplayer | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep icecream | grep -v grep | cut -c10-14`
sudo kill -9 `ps -ef | grep conky | grep -v grep | cut -c10-14`
sleep 1
```

```
# vnc-server neu starten inkl. conky
sudo /etc/init.d/vncserver start
sleep 10
sudo /etc/init.d/vncserver restart
sleep 10

sudo cp /home/dieter/scripte/sw-leer.png /var/www/icecream.png
sudo cp /home/dieter/scripte/sw-leer.png /var/www/icecream-record.png
sudo cp /home/dieter/scripte/sw-leer.png /var/www/radio.png
sudo cp /home/dieter/scripte/sw-leer.png /var/www/10songs.png
sudo cp /home/dieter/scripte/sw-leer.png /var/www/100mb.png
sudo cp /home/dieter/scripte/sw-leer.png /var/www/90minuten.png
sudo cp /home/dieter/scripte/sw-leer.png /var/www/str_radio.png
sudo cp /home/dieter/scripte/sw-leer.png /var/www/sw_usb_mp3.png
sudo cp /home/dieter/scripte/sw-leer.png /var/www/usb-mp3.png

sudo dpkg --get-selections >/home/dieter/scripte/prozesse/pakete
sudo cp /home/dieter/scripte/prozesse/pakete /var/www

sudo ps -A > /home/dieter/scripte/prozesse/prozesse-all
sudo mount | grep sda > /home/dieter/scripte/prozesse/mount-sda

sudo cp /home/dieter/scripte/prozesse/prozesse-all /var/www
sudo cp /home/dieter/scripte/prozesse/mount-sda /var/www

sudo cp /home/dieter/scripte/icecream/icecream-songs/*.sh /home/dieter/scripte/icecream-songs
sudo cp /home/dieter/scripte/icecream/icecream-volumen/*.sh /home/dieter/scripte/icecream-volumen
sudo cp /home/dieter/scripte/icecream/icecream-time/*.sh /home/dieter/scripte/icecream-time
sudo cp /home/dieter/scripte/icecream/icecream-songs/*.sh /home/dieter/scripte
sudo cp /home/dieter/scripte/mplayer/radiostream/*.sh /home/dieter/scripte
sudo cp /home/dieter/scripte/playlisten/playlist.txt /home/dieter/scripte/playlist.txt

sudo cp /home/dieter/scripte/sw-leer.png /var/www/sw-service.png
sleep 2

# python /home/dieter/scripte/tasterled.py
python /home/dieter/scripte/favorite.py

sleep 1
```

_____service.sh_____

Bash-Scripte, Python-Scripte, HTML-Code, Grafik-Symbole

(Stand: 31.12.2013)

Es folgt nun eine Aufstellung aller Skripte und deren Funktionen.

Menü-A besteht aus:

- html-Dateien = 25
- pdf-Dateien = 2
- png-Dateien = 94
- py-Dateien = 30
- sh-Dateien = 290 davon mplayer = 64 und icecream 112
- Prozess-Dateien = 12

Warum so viele Skripte für mplayer und icecream?

Die Skripte im Verzeichnis ../scripte/mplayer/radiostream sind lediglich Sicherheitskopien

mit einer definierten Reihenfolge. Sie werden nur benötigt, wenn man sich die Skripte mal verhunzt hat.

Die im Verzeichnis ../scripte/radiostream werden vom Skript service.sh aufgerufen. Dieses Skript sorgt für Ordnung im System, wenn mal die Reihenfolge der Senderliste nicht mehr stimmt und die Anzeige in der Webschnittstelle falsche Sender anzeigt, z.B. rpr1 läuft aber wdr5 wird angezeigt. Mit dem Service-Skript werden alle Streaming-Skripte, auch die von icecream, in Grundstellung gebracht, damit alles wieder synchron ist.

Die im Verzeichnis ../scripte/favorite-radiostream liegenden Skripte enthalten den festgelegten Zustand des gewählten Favoritensenders.

Das Festlegen des Favoritensenders erfolgt über das Skript:
save-favorite-radiostream.sh. Damit werden die str_radio-Skripte in das o.a. Verzeichnis kopiert und durch das Skript - restore-favorite-radiostream.sh - aktiviert.

Die Scripte ../scripte/icecream-volumen, ../scripte/icecream-time, ../scripte/icecream-songs dienen dem Aufnahmemodus: 100MB, 90Minuten, 10Songs. Und im Verzeichnis icecream liegen nochmals alle eben genannte Verzeichnisse mit den darin befindlichen Skripten als Kopie, wenn mal die Reihenfolge in Unordnung gerät. Mit dem o.a. Skript service.sh lässt sich dann wieder Ordnung herstellen.

Menü A4, MPLAYER-USB-STICK

Für dieses Menü musste ich erst einmal die umfangreichen Funktionen vom Mplayer verstehen. Man kann diesen Player auch aus der Ferne steuern. Der Schlüssel dafür sind 2 fifo-Dateien, ich musste dafür einige Zeit im Web suchen.

Die fifo-Dateien legt man sich wie folgt an, am besten im lokalen Verzeichnis /.mplayer . Ich habe das wie folgt gemacht:

- **mkfifo /home/dieter/.mplayer/mplayer_fifo_IN**
- **mkfifo /home/dieter/.mplayer/mplayer_fifo_OUT**

Der mplayer startet man dann wie folgt.
mplayer -slave -idle -quiet -input file=/home/dieter/.mplayer/mplayer_fifo_IN > /home/dieter/.mplayer/mplayer_fifo_OUT &

Nun wartet mplayer auf die Dinge die da kommen .

Dann gibt man ihm eine Playliste über die Konsole oder einem Skript zum abarbeiten mit:
echo "loadlist /home/dieter/scripte/playlist.txt" > /home/dieter/.mplayer/mplayer_fifo_IN &

Zu guter Letzt lässt man sich dann die Ausgabe Präsentieren über:
cat /home/dieter/.mplayer/mplayer_fifo_OUT < /dev/null &

Drückt man im Menü A4 die Taste 1 wird o.a. Prozedur abgearbeitet inkl. Erstellung einer Playliste die wie oben angegeben dem mplayer übergeben wird, siehe hierzu auch Skript mplay02.sh.

Jetzt kann man die Playliste per Skript oder Konsole vor bzw. zurück blättern mit:

- **echo "pt_step 1" > /home/dieter/.mplayer/mplayer_fifo_IN (vor)**
- **echo "pt_step -1" > /home/dieter/.mplayer/mplayer_fifo_IN (zurück)**

Das Blättern durch die Liste erfolgt mit den Tasten 2 und 3. Die Tasten 2+4 gleichzeitig gedrückt spielt die o.a. Playliste zufällig ab. Man kann hier zur Zeit nicht blättern.

Menü A9, Mplayer Internetradio

Da es so schön war, habe ich das ganze noch mal in diesem Menü mit der Playliste für Radio-Streaming gemacht. Ablauf ist wie zuvor beschrieben, diesmal jedoch mit einer Playliste nur mit Webadressen von Radiosendern.Die **playliste-radio.txt** ist beliebig erweiterbar. Sie ist nicht mit icecream gekoppelt und völlig unabhängig vom Menügeschehen von Menü A3. Damit es auch so richtig Spaß macht, habe ich auch noch eine Importmöglichkeit einer externen Playliste eingebaut. Mit den Tasten 1+2+3 importiert man eine Datei, die immer **playliste-radio.txt** lauten und sich in der root vom USB-Stick liegen muss. Mit den Tasten 2+4 (Playliste extern) bzw. 1+4 (Playliste intern) schaltet man die Playlisten um und mit Taste 1 startet man dann den neuen Streaming.

Eigene Befehle zur Fernsteuerung des Rasputin-3

Ist damit alles geklärt? Nein, es kommt noch besser. Alle Menüs lassen sich auch per Webschnittstelle ferngesteuert aktivieren. Dazu benötigt man lediglich mein „Lieblings-Administration-Programm - WEBMIN -“ Dieses Programm hat eine nette Möglichkeit, die ich mal ausgiebig genutzt habe. Man kann sich eigene Befehle anlegen und diese dann per Knopfdruck nutzen. Ich habe einen Webminbenutzer angelegt mit dem Namen **piface.** Nachdem man sich an Webmin angemeldet hat über: https://ip-adresse:10000 erhält man folgende Anzeige:

Einen Klick auf den Button A1 oder B5 etc. aktiviert das gewählte Menü. Dies kann man im Webdisplay kontrollieren und sehen, dass das Menü umgeschaltet wurde. Die „Eigenen Befehle“ lassen sich hier wunderbar erweitern. So kann man Rasputin-3 aus der Ferne steuern. Die Flexibilität lässt sich nun kaum noch steigern. Man muss das Menü nur noch mit Leben füllen.

Nachfolgend werden alle Scripte mit deren Funktion aus Menü A aufgeführt:

Favoriten-Menü
favorite.py

Start-LED's-Zustandsanzeige
startled.py
startled-android.py
startled-iphone.py
startled-wlan.py

LED-Anzeige festlegen und Tasterbedienung abfragen
tasterled.py
tasterled01.py
tasterled02.py
tasterled03.py
tasterled04.py
tasterled05.py
tasterled06.py
tasterled07.py
tasterled08.py
tasterled09.py
tasterled10.py
tasterled11.py
tasterled12.py
tasterled13.py
tasterled14.py
tasterled15.py
tasterled16.py

Favoriten-Steuerung
favorite-a.py
favorite-b.py
favorite-c.py
favorite-d.py
favorite-go.py
favorite_one.py
favorite.py
favorite-save.py

Aus den Python-Skripten werden alle Aktionen gesteuert. Das bedeutet, dass von hier aus alle unten aufgeführten Bash-Skripte aufgerufen werden.

Access-Point wählen
android.sh
iphone.sh
wlan-home.sh
wlan.sh

Hiermit wird festgelegt, an welchen Accesspoint sich Rasputin-3 anmeldet.

Schieberegister-icecream
change-ice-down.sh
change-ice-up.sh

Das Schieberegister macht folgendes. Es benennt die nachfolgende ice-Skripte um. So dass der Inhalt von ice01.sh in das Skript ice02.sh wandert und der Inhalt von ice02.sh in das Skript ice03.sh wandert und so weiter, bis alle ice-Skipte abgearbeitet wurden. Das Verschieben erfolgt in allen ice-Skripten inkl. der Unterverzeichnisse -volumen, -time, -songs. Dadurch erreicht man, dass man einen anderen Sender aufzeichnet, als den man hört. Daher braucht man auch das Service-Skript, damit man alles wieder synchronisiert bekommt.

icecream
ice01.sh
ice02.sh
ice03.sh
ice04.sh
ice05.sh
ice06.sh
ice07.sh
ice08.sh
ice09.sh
ice10.sh
ice11.sh
ice12.sh
ice13.sh
ice14.sh
ice15.sh
ice16.sh

16 Sender = 16 Skripte. Es wird immer Skript ice01.sh über Pythonskript aufgerufen. Durch das Schieberegister ändern sich die Inhalte der Skripte. Dadurch erreicht man dann alle Sender per Tastendruck.

Radiostream mit mplayer starten

str_radio01.sh
str_radio02.sh
str_radio03.sh
str_radio04.sh
str_radio05.sh
str_radio06.sh
str_radio07.sh
str_radio08.sh
str_radio09.sh
str_radio10.sh
str_radio11.sh
str_radio12.sh
str_radio13.sh
str_radio14.sh
str_radio15.sh
str_radio16.sh

16 Sender = 16 Skripte. Es wird immer Skript str_radio01.sh über Pythonskript aufgerufen. Durch das Schieberegister ändern sich die Inhalte der Skripte. Dadurch erreicht man dann alle Sender per Tastendruck.

Playlisten

playlist-radio-import.sh
playlist-songs-import.sh
playlist-stream-import.sh
playlist_sw_radio_ext.sh
playlist_sw_radio_int.sh
playlist_sw_songs_ext.sh
playlist_sw_songs_int.sh

Switch-Aufnahmemodus-icecream

change-songs.sh
change-time.sh
change-volumen.sh

Damit wird der Aufnahmemodus festgelegt und auch im Webdisplay angezeigt.

Favoriten erstellen/starten mplayer

restore-favorite-radiostream.sh
save-favorite-radiostream.sh

Hiermit legt man seinen Lieblingssender fest, den man dann immer schnell aufrufen kann über die Tasten 1+3 - RADIO-FAVORITE - im Menü A3.

Schieberegister-mplayer / Playlistenswitch -import

change-ice-down.sh
change-ice-up.sh
change-songs.sh
change-str-radio-down.sh
change-str-radio-up.sh
change-time.sh
change-volumen.sh

Mit diesen beiden Skripten beeinflusst man den abzuspielenden Sender. Es funktioniert genau so wie es oben bei Icecream beschrieben ist.

Menü-Favoriten

favorite02.sh
favorite03.sh
favorite04.sh
favorite05.sh
favorite06.sh
favorite08.sh
favorite09.sh
favorite10.sh
favorite11.sh
favorite13.sh
favorite14.sh
favorite16.sh
favorite-go-a.sh
favorite-go-b.sh
favorite-go-c.sh
favorite-go-d.sh
favorite-go.sh
favorite-save-a.sh
favorite-save-b.sh
favorite-save-c.sh
favorite-save-d.sh
favorite-save.sh
favorite.sh
favorite-switch-go.sh
favorite-switch-one.sh

Hiermit werden die Menü-Favoriten festgelegt über die Tastenkombination 1+2+3+4

Prozesse beenden

killall.sh
killice-cream.sh
killicecream.sh (wird nicht mehr verwendet)*
killm-player.sh
killmplayer.sh (wird nicht mehr verwendet)*
kill-python.sh
killtasterled01.sh

* Ich musste diese Skripte umbenennen, da beim Aufruf von kill -9 …. mplayer auch das Skript das es aufrief beendete. Ich wunderte mich immer, warum das Script sich so komisch verhielt bis ich begriff, dass das Skript killmplayer.sh selbst gekillt wurde. Blöd wenn man den Ast absägt, auf den man sitzt. Analog gilt das auch für killicecream.sh

Diese kill-Skripte beenden die entsprechenden Prozesse von icecream, mplayer, Bash-Scripte sowie Python-Skripte. killall.sh erledigt gnadenlos alle zu vorgenannten Prozesse. Das nachfolgende Skript service.sh erledigt das ebenfalls. Es räumt gnadenlos auf und bringt alles wieder in Grundstellung.

System bereinigen, in Grundstellung bringen, Inhalte ermitteln
service.sh

Service ist notwendig, wenn mal alles hakelig erscheint. Man bringt alles in Grundstellung. Man bekommt dadurch einen definierten Anfang. Außerdem wird ermittelt ob ein USB-Stick gemountet ist und welche Dateien sich darauf befinden. Auch nach MP3-Songs wird gesucht. Prozessliste erstellt, welche Pakete installiert sind, Grafiksymbole im Webdisplay in Grundstellung gebracht, Playliste erstellen etc.

Prozesse ermitteln
prozesse.sh

Das Skript ermittelt alle Pythonscript-Prozesse, Prozesse von icecream sowie vom mplayer. Das Skript wird durch einen Cronjob minütlich gestartet. Das Ergebnis kann man sich im Webdisplay ansehen.

mplayer-starten
mplay01.sh (wird nicht mehr verwendet)
mplay02.sh (nach Playlisten-Reihenfolge abspielen)
mplay02rnd.sh (zufällige Wiedergabe)

mplay02.sh Startet den mplayer um Songs vom USB-Stick abzuspielen. Ist noch in der Testphase und muss noch ausgebaut werden.

System beenden/neu starten
osdown.sh
osreboot.sh

Backup Scripte auf USB-Stick
sic-scripte.sh

Hiermit werden alle Skripte inkl. HTML-Code von /var/www auf USB-Stick gesichert und eine Historie angelegt. Sicher ist sicher!

VNC-Server starten
vnc1.sh

Mit diesem Skript wird per Cronjob der VNC-Server gestartet. Da conky beim ersten Anlauf nicht starten will, starte ich in dem Skript den VNC-Server und nach 3 Sekunden machen ich einen Restart des VNC-Servers. Dadurch konnte ich conky dazu überreden, zu starten und mir die nette Systemübersicht darzustellen, die man im Webbrowser sieht.

Snapshot vom VNC-Desktop machen

vnc-screen.sh

Mit diesem Script mache ich alle 5 Minuten ein Snapshot vom VNC-Desktop und stelle dies dann in der Webschnittstelle dar (nur der Bereich von conky).

Sprungmarken im Menü

speed02b.sh
speed02.sh
speed07.sh
speed12.sh

Dient der schnellen Navigation in der Menüstruktur

Tasterbedienung sichern für Webseiteanzeige

sw0.sh
sw1_2.sh
sw1_3.sh
sw1_4.sh
sw1.sh
sw2_3.sh
sw2.sh
sw3_4.sh
sw3.sh
sw4.sh
sw_favorite_b1_a.sh
sw_favorite_b1_b.sh
sw_favorite_b1_c.sh
sw_favorite_b1_d.sh
sw_favorite_b1.sh
sw_favorite_end.sh
sw_favorite.sh
sw-playlist.sh
sw_usb_mp3_off.sh
sw_usb_mp3.sh

Dient der Display-Anzeige in der Webschnittstelle.

Menüsprünge

tasterled01.sh
tasterled02.sh
tasterled03.sh
tasterled04.sh
tasterled05.sh
tasterled06.sh
tasterled07.sh
tasterled08.sh
tasterled09.sh
tasterled10.sh
tasterled11.sh
tasterled12.sh
tasterled13.sh
tasterled14.sh
tasterled15.sh

tasterled16.sh
tasterledb.sh
tasterled.sh

Dient der Menünavigation. Led-Ansteuerungen und Tastenbedienungen abfragen.

USB-Stick mounten / unmounten / Playliste erstellen

mount1.sh
umount1.sh

Play-Anzeige „Internet-Sender und USB-Stick“

wz-sender.sh
wz-title-aktuell.sh
wz-title-radio.sh
wz-title-usb.sh
sender.sh

Grafische Elemente

1live-diggi.png
1live-fiehe.png
1live-kassettendeck.png
1live-klubbing.png
1live-plan-b-mit.png
1live-plan-b.png
1live.png
1live-rocker.png
aktive-favorite.png
aktive.png
fhe-radio-colonia.png
funkhaus-europa.png
leer.png
ma10.png
ma10s.png
ma11.png
ma11s.png
ma12.png
ma12s.png
ma13.png
ma13s.png
ma14.png
ma14s.png
ma15.png
ma15s.png
ma16.png
ma16s.png
ma17.png
ma17s.png
ma1_.png
ma1.png
ma1s.png
ma2.png
ma2s.png
ma3.png
ma3s.png
ma4.png
ma4s.png

ma5.png
ma5s.png
ma6.png
ma6s.png
ma7.png
ma7s.png
ma8.png
ma8s.png
ma9.png
ma9s.png
mb1.png
on.png
PenguinInfo.png
playliste.png
radio-on.png
record.png
rpr1.png
sradio-tv.png
sw1_2.png
sw1_3.png
sw1_4.png
sw1.png
sw2_3.png
sw2.png
sw3_4.png
sw3.png
sw4.png
sw-android.png
sw-favorite-a.png
sw-favorite-b.png
sw-green.png
sw-iphone.png
sw-leer.png
sw-next.png
sw-playlist.png
sw-reboot.png
sw-save.png
sw-service.png
sw-shutdown.png
sw-stop.png
sw_usb_mp3.png
sw-who.png
sw-wlan.png
sw-zufall.png
usb-mp3.png
wdr2.png
wdr3.png
wdr4.png
wdr5.png
title.png
usb-mp3.png
Wz_sender.png
Wz_title.png

HTML-Scripte

m_a10.html
m_a11.html
m_a12.html
m_a13.html
m_a14.html
m_a15.html
m_a16.html
m_a17.html
m_a1.html
m_a2.html
m_a3.html
m_a4.html
m_a5.html
m_a6.html
m_a7.html
m_a8.html
m_a9.html
m_ab.html
m_a.html
menue_aktuell.html
menue_favorite_b1_a.html
menue_favorite_b1_b.html
menue_favorite_b1_c.html
menue_favorite_b1_d.html
menue.html

Diese Skripte dienen dazu, sofort in das Aktuelle Menü zu navigieren, damit man erkennt, wo man sich zur Zeit im Menü befindet. Ruft man die Webseite des Rasputi3 auf, dann hat man eine Frameanzeige. Der linke Frame zeigt ein Menüdisplay und der rechte Frame das aktuelle aktive Menü in die Menüstruktur (aktuell ganz oben).

Menü-B besteht aus:

html-Dateien	= 19
png-Dateien	= 52
py-Dateien	= 21
sh-Dateien	= 57

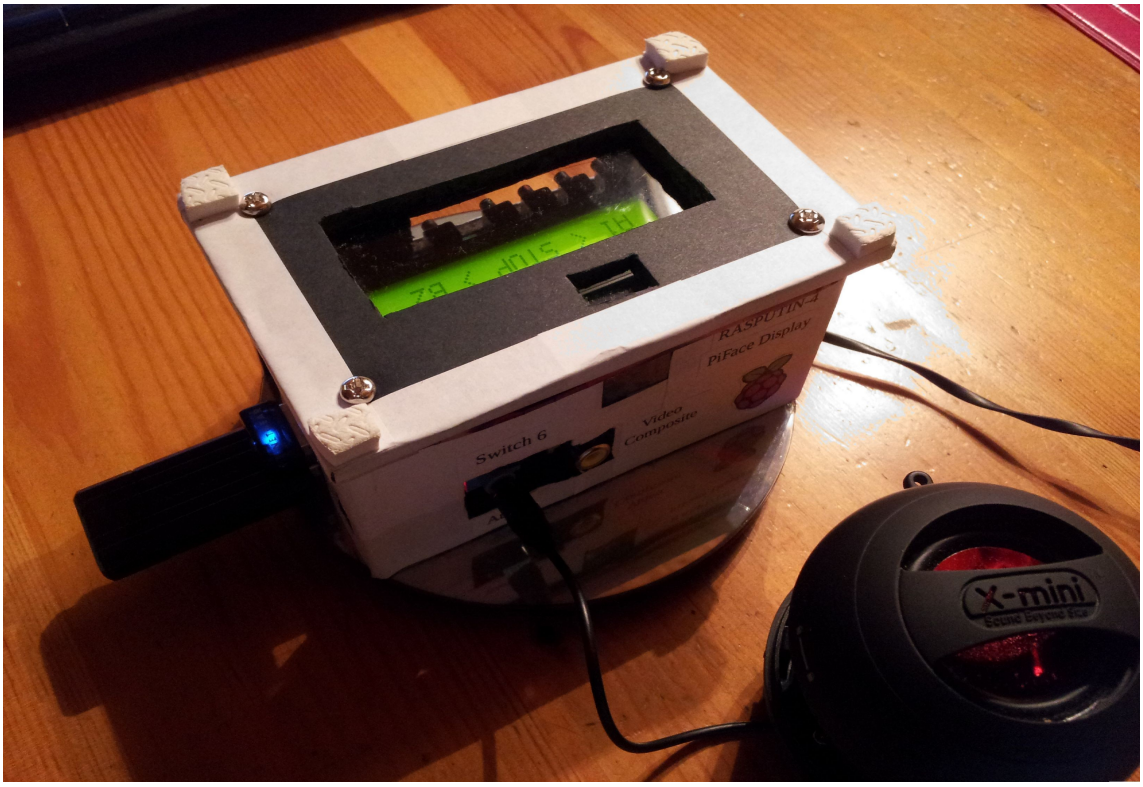
Das Menü B ist noch nicht mit Aufgaben belegt. Man kann lediglich darin navigieren.

Dazu kommen im Verzeichnis /var/www/ 182 Dateien für das Web-Display, die aus HTML-Code, PDF-Dateien und PNG-Dateien bestehen.

Macht alles zusammen 821 Dateien. Ist das ein Menümonster geworden!
(Stand: 31.12.2013)

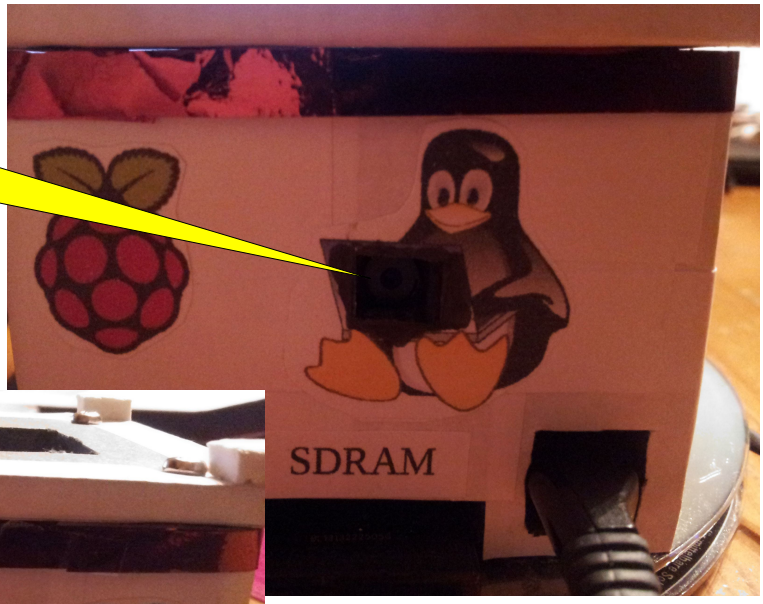
Kommt noch was?

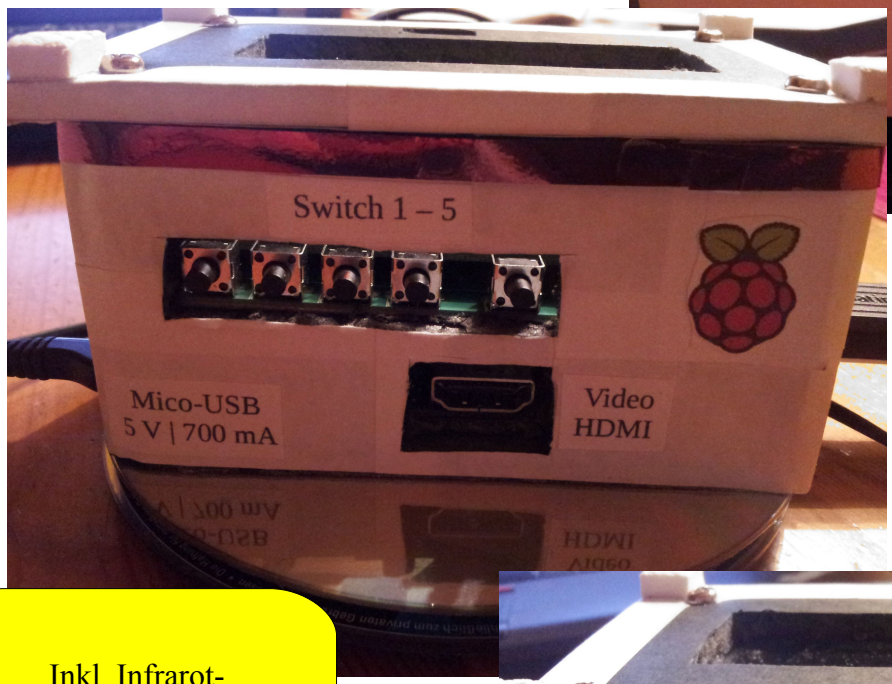
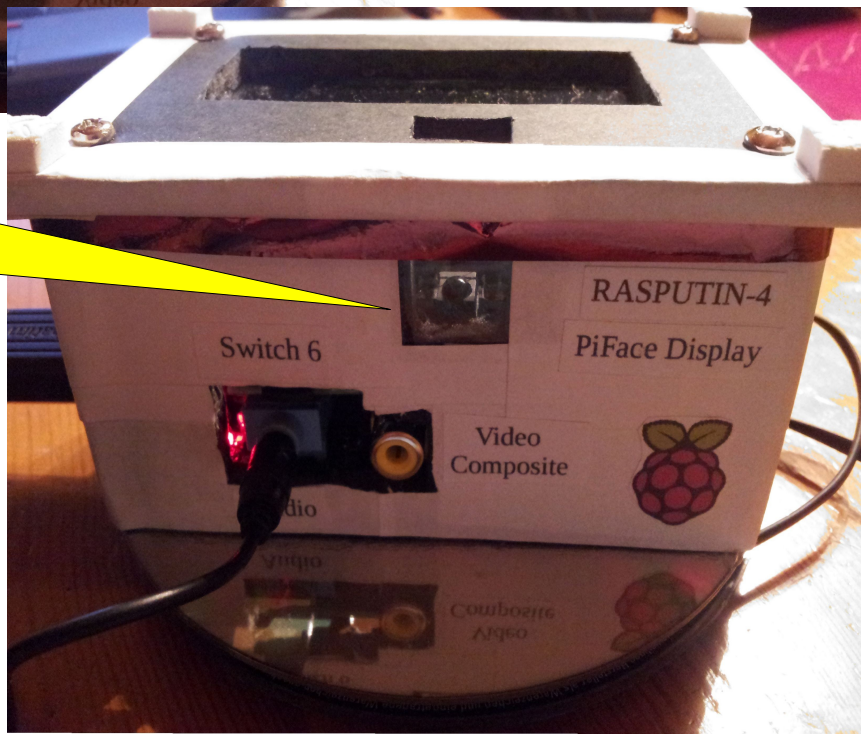
Eine weitere Dokumentation für **Rasputin-4** mit **PiFace Display** und Infrarot-Empfänger ist in Vorbereitung und folgt bald. Einige Fotos finden Sie schon mal auf den nächsten Seiten.


RASPUTIN-3
PiFace Display
Switch 6
Video Composite
X-mini

A9|EIN->-<-A10-F
A1 < STOP > B2

Inkl. Raspberry Pi Camera-Modul





Inkl. Infrarot-Empfänger für Remote-Kontrolle über eine Fernbedienung