Raspberry Pi 3B: Soundbox Pi

Eine Soundbox mit Geräuschen für den Unterricht. Spiel Stefan Raab und untermale die Lektion mit akustischen Signalen. Alles zum Nachbau und zur Installation folgt hier:

Einkaufsliste:

Raspberry Pi 3B
https://www.amazon.de/Raspberry-Pi-Model-ARM-Cortex-A53-Bluetooth/dp/B01CD5VC92/ref=sr_1_3?__mk_de_DE=%C3%85M%C3%85%C5%BD%C3%95%C3%91&keywords=raspberry+pi+3b&qid=1703087575&sr=8-3
Widerstand 10k Ohm
https://www.distrelec.ch/de/metalloxidschicht-widerstand-250mw-10kohm-rnd-rnd-155mor0w4j0103a50/p/30088533?trackQuery=10%20kohm%20widerstand&pos=1&origPos=1&origPageSize=50&track=true&sid=60f54c163c30066cc6e52332eed78ce5aa0475b7&itemList=search
Jumper Wire
https://www.distrelec.ch/de/wire-jumpers-male-to-female-10-pcs-mikroelektronika-mikroe-512/p/30410309?trackQuery=jumper%20wire&pos=2&origPos=2&origPageSize=50&track=true&sid=755a52572d5dece1343a60ef702e2bbe6415cc7b&itemList=search
Breadboards zum Löten
https://www.amazon.de/GeeekPi-Steckplatine-vergoldete-experimentelle-doppelseitig/dp/B07G5CRQXK/ref=sr_1_1?keywords=breadboard%2Bzum%2Bl%C3%B6ten&qid=1702887917&sr=8-1&th=1
Netzteil 5V/2A für Raspberry Pi
https://www.amazon.de/Aukru-Micro-USB-Ladeger%C3%A4t-Stromversorgung-Raspberry/dp/B01566WOAG/ref=sr_1_7?__mk_de_DE=%C3%85M%C3%85%C5%BD%C3%95%C3%91&crid=1OLIX3XM3QPR4&keywords=netzteil+raspberry+pi&qid=1702908448&s=ce-de&sprefix=netzteil+raspberry+pi%2Celectronics%2C95&sr=1-7
Druckknöpfe
https://www.amazon.de/YIXISI-Druckschalter-Momentary-Drucktastenschalter-Mikroschalter/dp/B08FJ6GDZK/ref=sr_1_3?__mk_de_DE=%C3%85M%C3%85%C5%BD%C3%95%C3%91&crid=3GMEGHZE91ORE&keywords=druckknopf+arduino&qid=1702888004&sprefix=druckknopf+arduino%2Caps%2C82&sr=8-3

Weiteres:

- Holzbox aus Brockenhaus
- Dicker Draht für Brücken auf Breadboard
- Aktiv-Boxen (Lautsprecher)
- Evtl. 3D-Drucker (falls ein Gehäuse für den Pi gedruckt werden möchte, oder Gehäuse kaufen
- Draht, Lötkolben, Lötzinn, nötiges Kleinwerkzeug, Leim

Den Python-Code kann man auf unserer Webseite herunterladen.

Autostart des Python-Skripts:

Das Skript soll automatisch starten, sobald der Raspberry Pi Strom hat.
Hierzu wie folgt vorgehen:

In der Konsole den Befehl eingeben: *crontab –e*

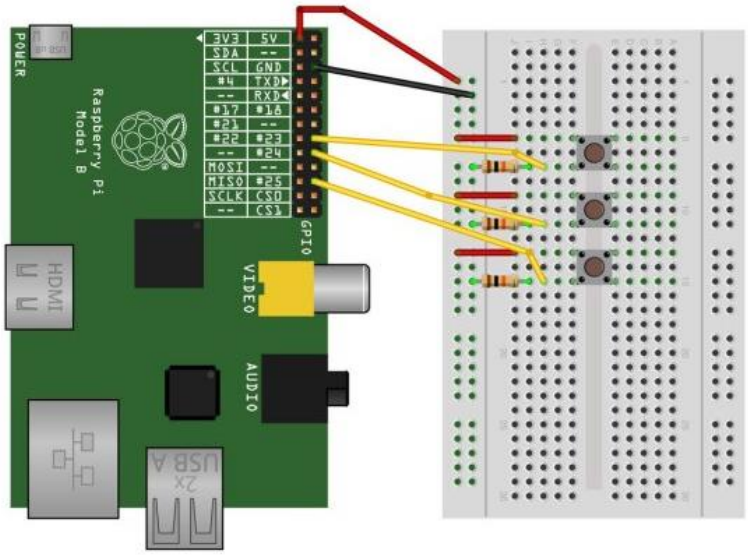
Wird dieser Befehl zum ersten Mal eingegeben, sollte man in der Vorgeschlagenen Auswahl */bin/nano* einrichten.

Der folgende Befehl ist für den Autostart nötig:
@reboot /usr/bin/python /home/pi/soundboard/soundboard.py

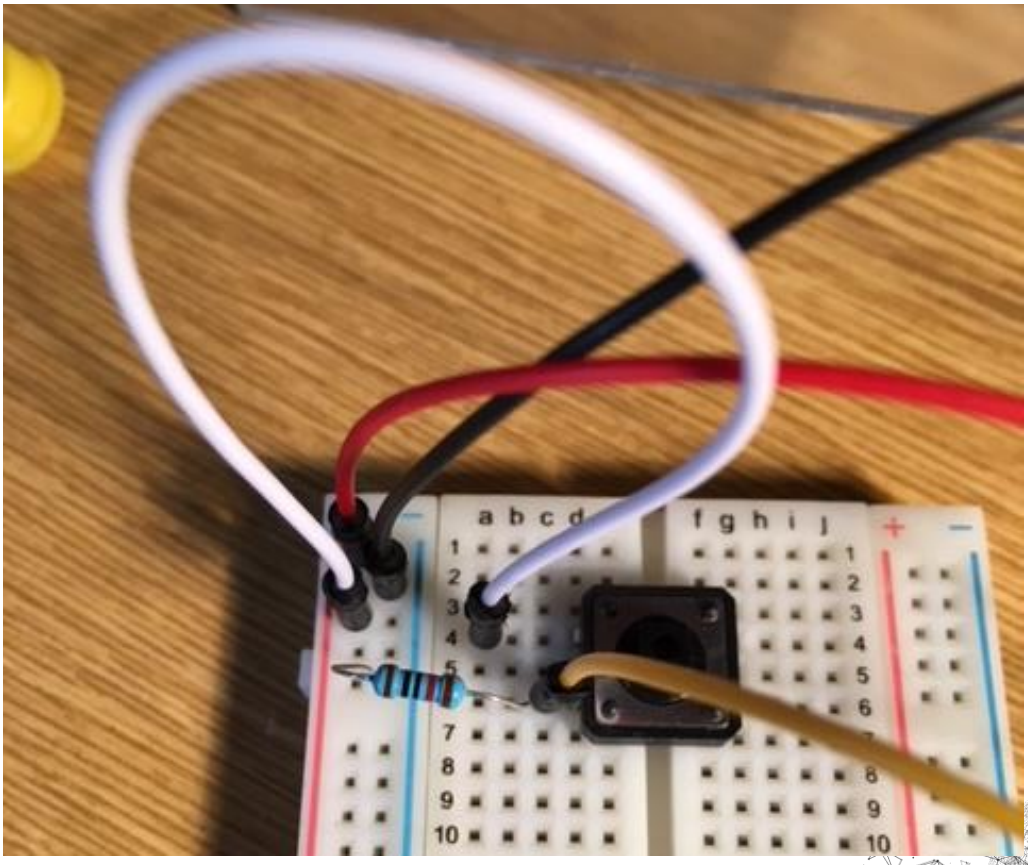
startet Python Pfad der Datei und Dateiname

Bilderreihe zum Zusammenbau:

Schema für den Anschluss der Druckknöpfe am Pi (Test mit Plastik-Breadboard):



Auf dem Steckboard:



Auf dem Raspberry Pi:

3.3V und GND sowie ein freier GPIO (hier Nr. 23)

Code (Beispiel) → wird später ergänzt und angepasst:

```
import pygame.mixer
from time import sleep
import RPi.GPIO as GPIO
from sys import exit

GPIO.setmode(GPIO.BCM)
GPIO.setup(23, GPIO.IN)
GPIO.setup(24, GPIO.IN)
GPIO.setup(25, GPIO.IN)

pygame.mixer.init(48000, -16, 1, 1024)

sndA = pygame.mixer.Sound("buzzer.wav")
sndB = pygame.mixer.Sound("clap.wav")
sndC = pygame.mixer.Sound("laugh.wav")

soundChannelA = pygame.mixer.Channel(1)
soundChannelB = pygame.mixer.Channel(2)
soundChannelC = pygame.mixer.Channel(3)

print "Soundboard Ready."

while True:
    try:
        if (GPIO.input(23) == True):
            soundChannelA.play(sndA)
        if (GPIO.input(24) == True):
            soundChannelB.play(sndB)
        if (GPIO.input(25) == True):
            soundChannelC.play(sndC)
        sleep(.01)
    except KeyboardInterrupt:
        exit()
```

Initialize Pygame's mixer.

Load the sounds.

Set up 3 channels, one for each sound, so that we can play different sounds concurrently.

Let the user know the soundboard is ready.

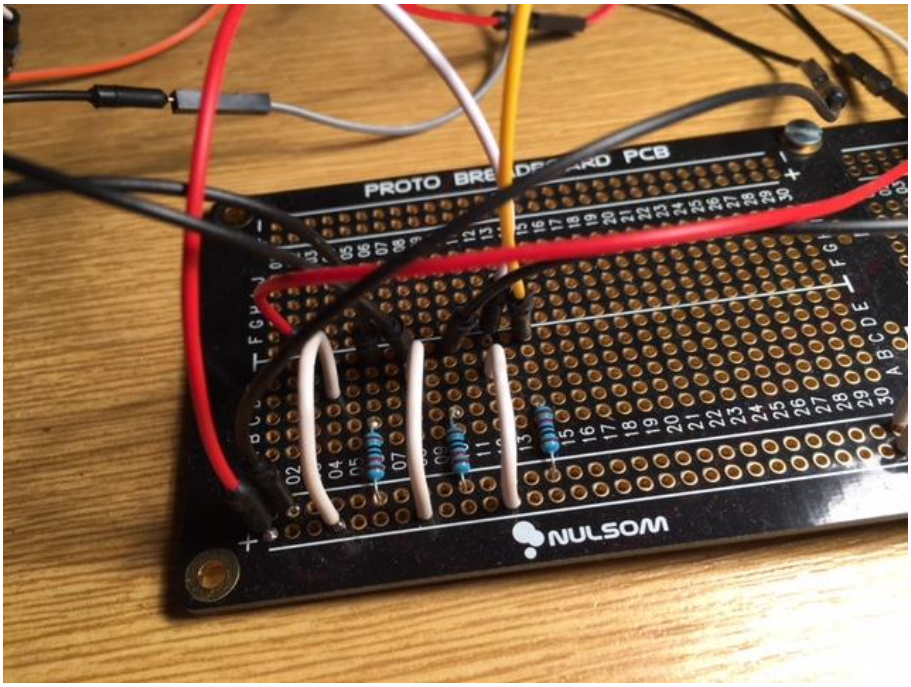
If the pin is high, execute the following line.

Play the sound.

Don't "peg" the processor by checking the buttons faster than we need to.

This will let us exit the script cleanly when the user hits CTRL+C, without showing the traceback message.

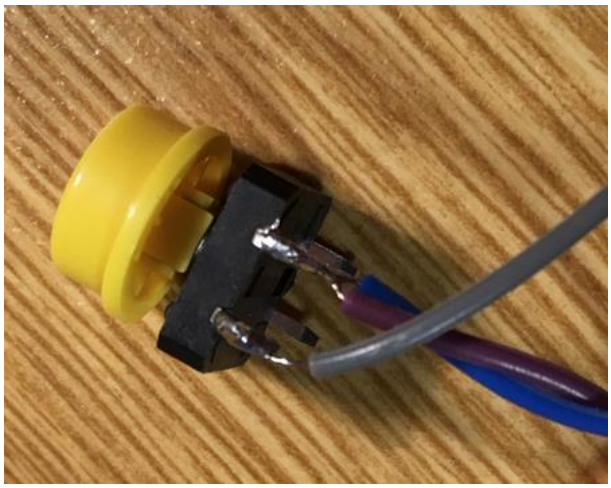
Wir werden nun aber mit einem lötbaren Breadboard arbeiten und die Knöpfe nicht direkt aufstecken, sondern verdrahten. Im Bild sind bereits drei Knöpfe eingerichtet.



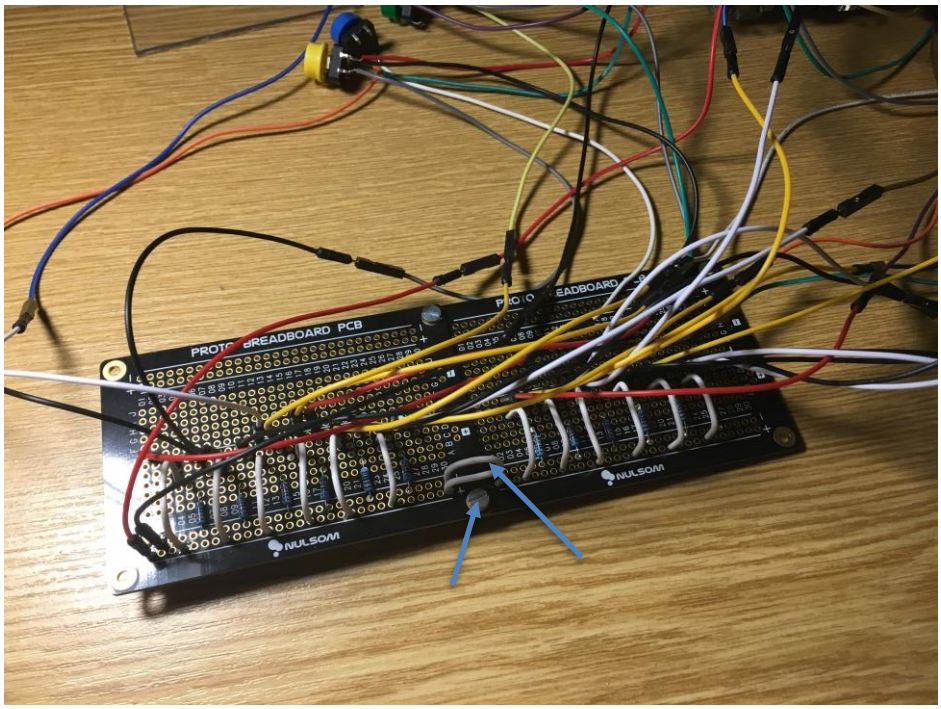
Strombrücke

Schema:

Der Draht hinter der weissen Strombrücke (siehe Bild auf S. 4 unten) führt zum linken Füsschen des Druckknopfs. Der Draht hinter dem Widerstand führt zum rechten Füsschen des Druckknopfs. Am rechten Füsschen des Druckknopfs wird noch ein zweiter Draht (Jumper female) angelötet, welcher zu einem GPIO auf dem Raspberry Pi führt.

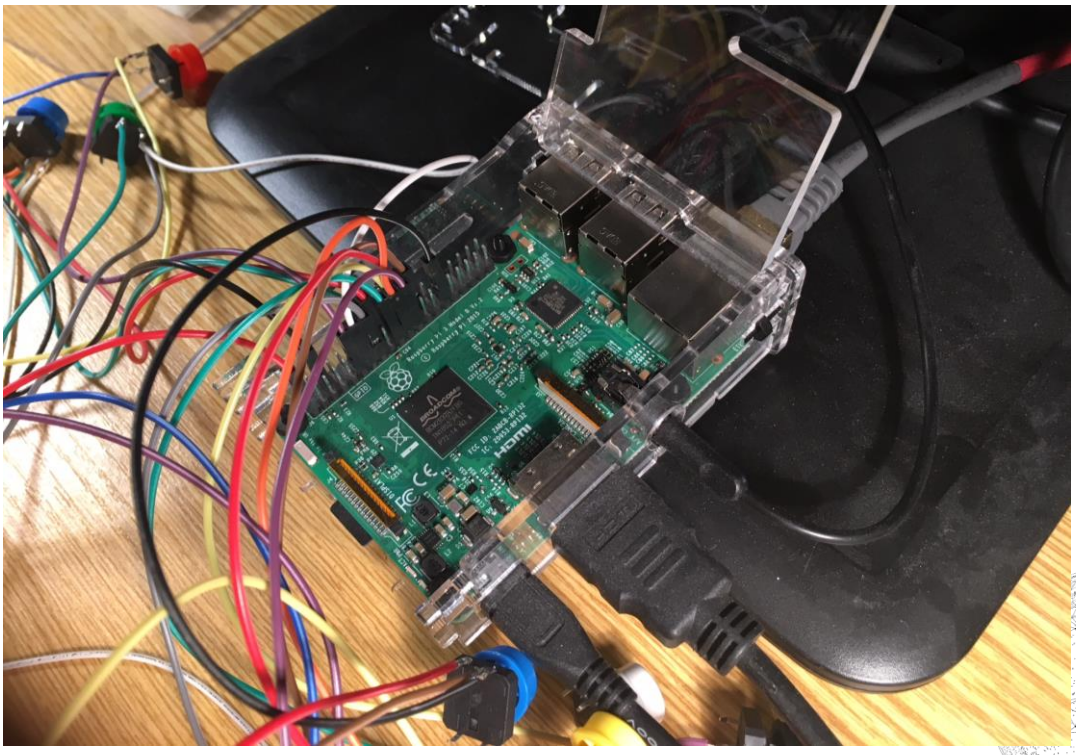


Wenn alles verlötet und gesteckt ist, sieht es auf dem Breadboard so aus:



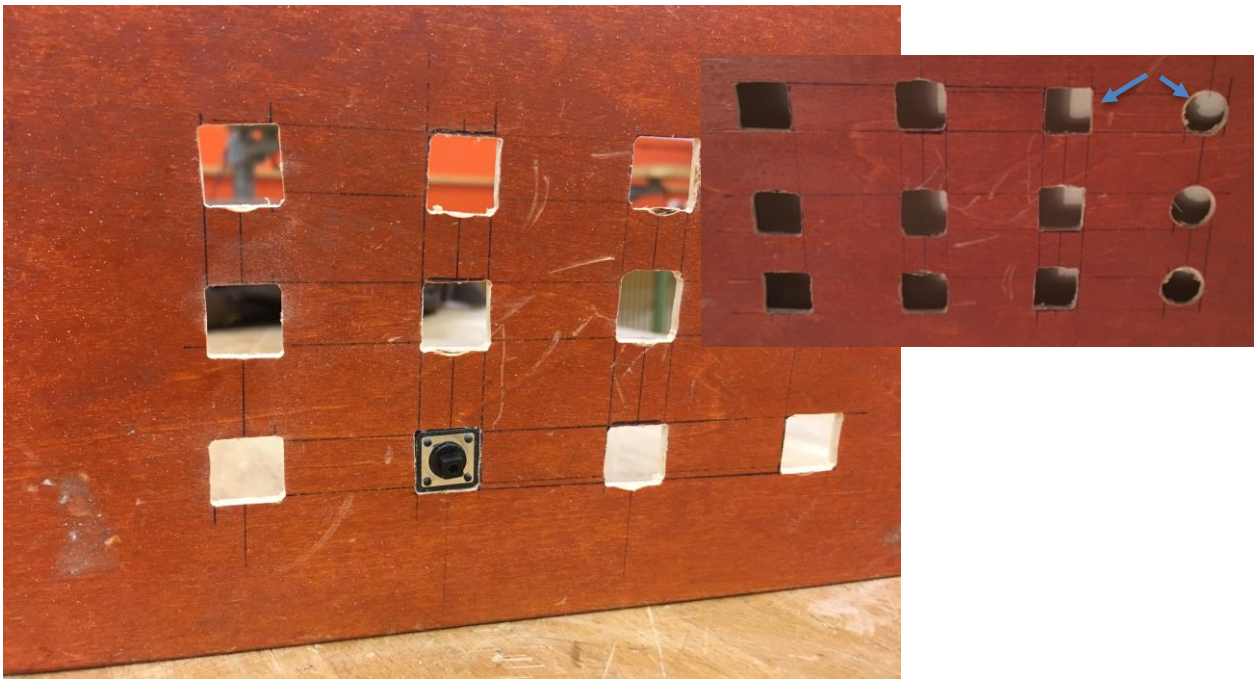
Die beiden Breadboards wurden übrigens zusammengeschraubt und mit einer Strombrücke verbunden.

Und so auf dem Raspberry Pi:

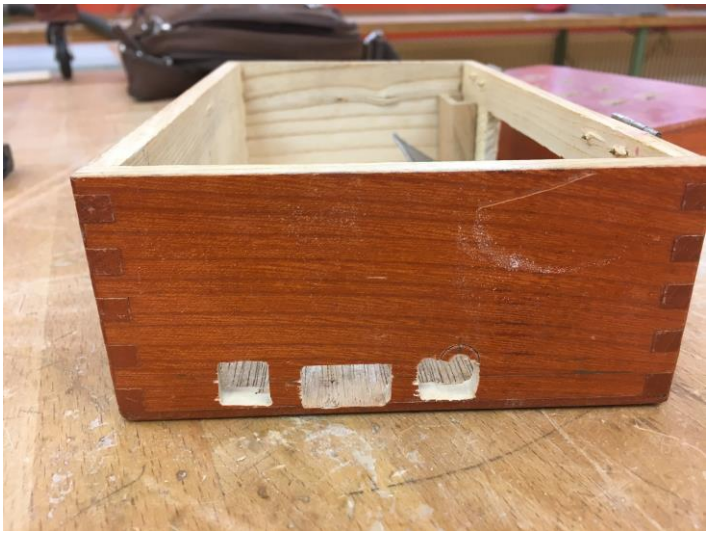


Für die Box haben wir eine Holzkiste im Brockenhaus gekauft, worin einmal ein Mikroskop war. Es ginge auch eine Zigarrenkiste oder etwas in der Art. Wichtig: genug Höhe, damit alles Platz hat.

Im Deckel bohren wir zuerst möglichst grosse Löcher und schleifen sie zu Quadraten, bis die Druckknöpfe genau Platz haben.

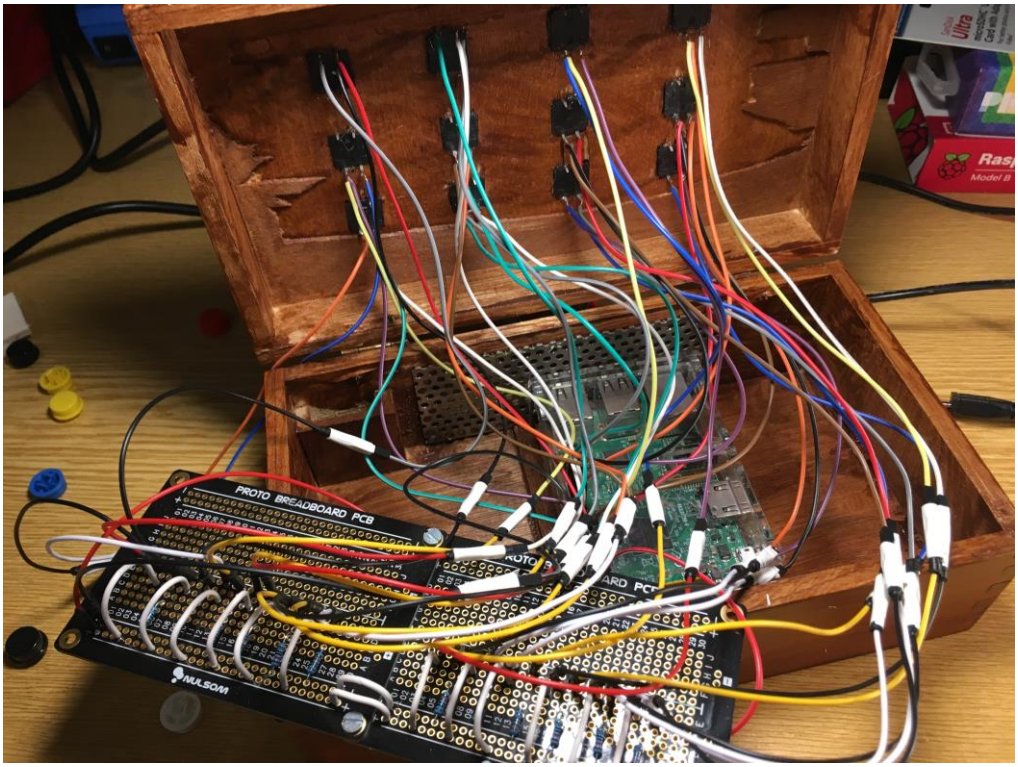


Am Gehäuse braucht es Schlitze für das Audiokabel und die Stromversorgung sowie eine Lüftung.

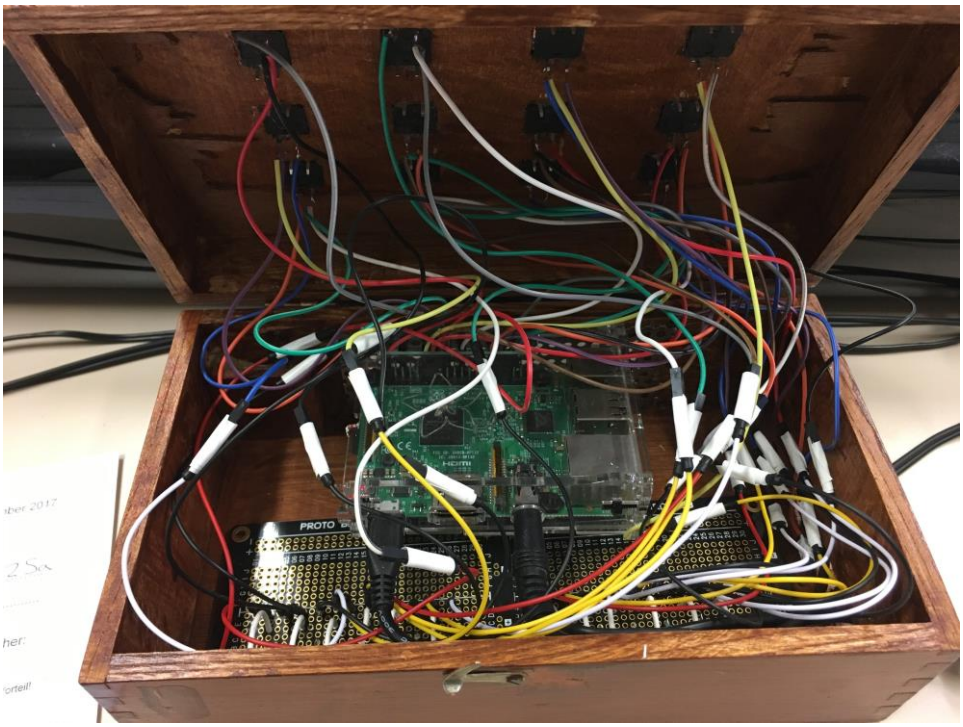


Je nach Zustand der Schachtel: sprayen, anstreichen oder wie in unserem Beispiel beizen.

Danach alles einbauen und die Druckknöpfe von unten einleimen. Kontaktstellen der Jumper Kabel mit Isolierband abkleben:



Breadboard unten, Raspberry Pi oben:



Kabel anschliessen und fertig – Viel Spass ☺

Der Code und die Sounds können auf unserer Seite heruntergeladen werden:

https://www.schularena.com/ict/informatik/make-it/soundbox-pi

Auf der folgenden Seite ist er auch abgedruckt.

Natürlich sind die Sounds beliebig austauschbar, müssen aber im Format wav sein.

```
# code by shawn wallace, modified by marcel isler
# code and sound files must be in the same folder
# for auto start of the program use crontab -e
# command for this project: @reboot /usr/pin/python /home/pi/soundbar/soundbar.py

import pygame.mixer
from time import sleep
import RPi.GPIO as GPIO
from sys import exit
# import the pygame mixer

GPIO.setmode(GPIO.BCM)
GPIO.setup(18, GPIO.IN)
GPIO.setup(23, GPIO.IN)
GPIO.setup(24, GPIO.IN)
GPIO.setup(25, GPIO.IN)
GPIO.setup(12, GPIO.IN)
GPIO.setup(16, GPIO.IN)
GPIO.setup(17, GPIO.IN)
GPIO.setup(27, GPIO.IN)
GPIO.setup(22, GPIO.IN)
GPIO.setup(5, GPIO.IN)
GPIO.setup(6, GPIO.IN)
GPIO.setup(13, GPIO.IN)
#set all the pins on the rasperry pi

pygame.mixer.init(48000, -16, 1, 1024)
# initialize the pygame's mixer

pygame.mixer.set_num_channels(13)
# set numbers of sounds / the default value is only for eight sounds

sndA = pygame.mixer.Sound("/home/pi/soundboard/gong.wav")
soundChannelA = pygame.mixer.Channel(1)
sndB = pygame.mixer.Sound("/home/pi/soundboard/bigben.wav")
soundChannelB = pygame.mixer.Channel(2)
sndC = pygame.mixer.Sound("/home/pi/soundboard/birthdaysound.wav")
soundChannelC = pygame.mixer.Channel(3)
sndD = pygame.mixer.Sound("/home/pi/soundboard/bombe.wav")
soundChannelD = pygame.mixer.Channel(4)
sndE = pygame.mixer.Sound("/home/pi/soundboard/boooo.wav")
soundChannelE = pygame.mixer.Channel(5)
sndF = pygame.mixer.Sound("/home/pi/soundboard/feel_good.wav")
soundChannelF = pygame.mixer.Channel(6)
sndG = pygame.mixer.Sound("/home/pi/soundboard/houstonproblem.wav")
soundChannelG = pygame.mixer.Channel(7)
sndH = pygame.mixer.Sound("/home/pi/soundboard/halleluj.wav")
soundChannelH = pygame.mixer.Channel(8)
sndI = pygame.mixer.Sound("/home/pi/soundboard/lachen.wav")
soundChannelI = pygame.mixer.Channel(9)
sndK = pygame.mixer.Sound("/home/pi/soundboard/tarzan.wav")
soundChannelK = pygame.mixer.Channel(10)
sndL = pygame.mixer.Sound("/home/pi/soundboard/herzschlag.wav")
soundChannelL = pygame.mixer.Channel(11)
sndM = pygame.mixer.Sound("/home/pi/soundboard/sobs.wav")
soundChannelM = pygame.mixer.Channel(12)
# set up the channels for each sound / wav format is important

print "sampler ready"
```

```
while True:
        try:
                if (GPIO.input(18) == True and not soundChannelA.get_busy()):
                        soundChannelA.play(sndA)
                sleep(.01)
                if (GPIO.input(23) == True and not soundChannelB.get_busy()):
                        soundChannelB.play(sndB)
                sleep(.01)
                if (GPIO.input(24) == True and not soundChannelC.get_busy()):
                        soundChannelC.play(sndC)
                sleep(.01)
                if (GPIO.input(25) == True and not soundChannelD.get_busy()):
                        soundChannelD.play(sndD)
                sleep(.01)
                if (GPIO.input(12) == True and not soundChannelE.get_busy()):
                        soundChannelE.play(sndE)
                sleep(.01)
                if (GPIO.input(16) == True and not soundChannelF.get_busy()):
                        soundChannelF.play(sndF)
                sleep(.01)
                if (GPIO.input(17) == True and not soundChannelG.get_busy()):
                        soundChannelG.play(sndG)
                sleep(.01)
                if (GPIO.input(27) == True and not soundChannelH.get_busy()):
                        soundChannelH.play(sndH)
                sleep(.01)
                if (GPIO.input(22) == True and not soundChannelI.get_busy()):
                        soundChannelI.play(sndI)
                sleep(.01)
                if (GPIO.input(5) == True and not soundChannelK.get_busy()):
                        soundChannelK.play(sndK)
                sleep(.01)
                if (GPIO.input(6) == True and not soundChannelL.get_busy()):
                        soundChannelL.play(sndL)
                sleep(.01)
                if (GPIO.input(13) == True and not soundChannelM.get_busy()):
                        soundChannelM.play(sndM)
                sleep(.01)

        except KeyboardInterrupt:
                exit()
#play the sounds once only and wait when the button is pressed
```