Tonuino: Toniebox als OpenSource

Ich habe mich entschlossen einen Tonuino selbst zu bauen. Ein Tonuino ist Lautsprecher mit Knöpfen. Wenn man ein Objekt (mit RFID Chip) auf die Lesefläche stellt, wird das dazugehörige Musikstück abgespielt.

- Motivation
- Eckdaten Projekt
- Software + Elektronik
- Gehäuse (Work in Progress)
- Musik und Figuren (Work in Progress)

Motivation

Tonuino vs. Toniebox

Über meine Tochter (1 Jahr) bin ich mit dem Thema Toniebox in Berührung gekommen. Das ganze ist ein Musikspieler für Kinder. Ich fand das eigentlich recht praktisch, bis ich mich genauer informierte.

Dabei lernte ich folgendes: Die Audio-Inhlate sind nicht auf den Tonies (=die Figürchen die als Datenträger fungieren) gespeichert. Dafür reicht die RFID Technologie gar nicht aus. Auf einen Tonie passt in etwa eine SMS an Text. Statt dessen ist ein Lizenzschlüssel gespeichert, der zum Download der Audio-Inhalte berechtigt.

Folgende Punkte gefallen mir an dem Konzept nicht:

- Ich möchte kein Kinderspielzeug in meinem Netzwerk haben. Sonst muss das mit Security-Updates versorgt werden und kann gehackt werden.
- Ich möchte die Musikbox uneingeschränkt ohne Internet betreiben können.
- Ich möchte meine eigene Musik auf die Musikbox aufspielen können ohne teure "Kreativ-Tonies" kaufen zu müssen.

Eckdaten Projekt

Zeitbedarf	Für mich: Software: 8h Elektronik: 8h Gehäuse: TBD Zum Nachmachen: Wahrscheinlich ähnlich, wenn ihr die Geräte nicht kennt. Sonst dank meiner Doku hoffentlich deutlich schneller.
Kosten	ca. 30 Euro an einmaliger Hardware + evtl. zusätzliche Kosten für Werkzeuge, aber das sollte es eigentlich alles im Zam geben
Kontakt Name, Email oder Telefonnummer?	Markus Dutschke Ihr findet mich in Mattermost. Kontaktiert mich direkt oder über ein ZAM Mitglied und ich helfe euch gerne.

Software + Elektronik

Informationsquellen

Auch wenn alle Infos zum Bau des Tonuinos frei verfügbar sind, gibt es doch verschiedene Websiten mit sich teilweise wiedersprechenden Informationen. Dem entsprechend mal ein Überblick:

Was ich benutzt habe und was mir die empfohlenen Informationsquelle zu sein scheint:

https://www.tonuino.de/TNG/
Diese Seite verlinkt auch auf das Repository mit dem Source-Code:

https://github.com/tonuino/TonUINO-TNG
Dies scheint wohl das offizielle Tonuino Community-Projekt zu sein.

Ich habe mich für das einfache 3 Knöpfe Layout ohne jegliche Extras entschieden.

Weitere Informationsquellen sind:

- Ursprüngliches Projekt: https://www.voss.earth/tonuino/
- Trouble Shooting: https://discourse.voss.earth/t/fehler-finden-troubleshooting-fuer-neulinge/7700
- Tonuino Handbuch: https://discourse.voss.earth/uploads/short-url/vbl5o26fqzoDaQ77Q5y3Hkto41Q.pdf

Microkontroller bespielen

Zum Bauen des Tonuino muss die Steuerungssoftware auf einen Microkontrolller gespielt werden. Die Software ist frei verfügbar und Ihr müsst über keine Programmierkenntnisse verfügen. Trotzdem ist die Sache deutlich fitzliger als es sich anhört.

Ich habe mich für folgenden Weg entschieden (Stand Dez. 2024):

- Arduino IDE installieren (=Entwicklungsumgebung)
- Source-Code herunterladen: https://github.com/tonuino/TonUINO-TNG
- Bibliotheken IN DER RICHTIGEN VERSION installieren. Bei mir hat funktioniert:
  DFPlayer Mini Mp3 by Makuna (by Michael C. Miller, makuna@live.com): 1.2.3
  JC_Button (by Jack Christensen, jack.christensen@outlook.com): 2.1.5

MFRC522 (by GithubCommunity): 1.4.12
Adafruit NeoPixel by Adafruit: 1.12.5 (ohne Feature NEO_RING zu benutzen)
Die in der README vom Sourcecode angegebenen Versionen waren bei mir nicht verfügbar
- Nen paar Änderungen in Konfig Dateien vornehmen (siehe README).
- Software auf Mikrokontroller (Arduino Nano 3 Klon von Aliexpress) spielen. Sobald dieser Schritt endlich ohne Fehler klappt, war das Thema Software auch schon erledigt. Bis das klappt, empfehle ich die Fehlermeldungen zu Googeln / ChatGPTn und mal nen Hello-World (blinkende LED) Programm auf den Mikrokontroller zu laden um zicherzustellen, dass Bootloader / Mikrokontroller Einstellungen richtig sind.

Es scheint aber auch noch andere Möglichkeiten zu geben, den Tonuino Source-Code auf den Mikrocontroller zu spielen. Ich habe irgendwas von Web-Upload und

```
pip install platformio
```

Damit habe ich aber keine Erfahrung. Da der Weg über die IDE aber aufwändig war, lohnt es sich eventuell diese beiden Wege auszuprobieren.

Elektronik

Ich empfehle einfach auf Amazon das Tonuino Set zu kaufen. Stand April 2025: "AZDelivery TonUINO Set (Mp3 Player, AZ-ATmega328-Board, RFID Kit und 13,56 MHz RFID Karten) kompatibel mit Arduino inklusive E-Book! " für 15,98€. Allerdings kann man auch die EInzelkomponenten kaufen (einfach Bilder vom Chip mit den Bildern des AZDelivery TonUINO Set vergleichen) - billiger wird das aber nicht: nur aufwändiger und fehleranfälliger. ... Ist aber erfahrungsgemäß machbar ^^ ...

Den Schaltplan findet man hier:

https://i0.wp.com/www.voss.earth/wp-content/uploads/2018/09/TonUINO_Schaltplan.png

Ich habe das Ganze auf eine Lochrasterplatine gelötet. Reine Lötdauer etwa 2h. Anbei mal das Layout, so dass Ihr euch diesen Schritt immerhin sparen könnt.

250423-1111_pcb-layout.svg
Öffnet die Datei am besten mit Inkscape. Es gibt auch Beschriftungen außerhalb des sichbaren Bereichs. Der Sichtbare Bereich entspricht der Lochrasterplatine. Der Arduino ist ein Pin länger als die Lochrasterplatine. Dieser Pin hängt einfach in der Luft (Buchsenleiste leer). Ich wollte das recht Platzspaarend haben und hab mir vielleicht nicht die notwendige Zeit für ein sauberes Layout genommen, dem entsprechend geht es etwas drüber und drunter - sorry dafür!

Ich empfehle sowohl den Arduino als auch das MP3 Player Modul über Buchsenleisten auf die Platine aufzustecken. Wenn mal was kaputt sein sollte, lassen sich die Konponenten schnell tauschen. Sonst wirds dank quick & diry Layout aufwändig. Hier ein paar Fotos von den

verwendeten zusätzlichen Komponenten:

ueberblick.jpg

top.jpg

bottom.jpg

Gehäuse (Work in Progress)

Rückwand

Der Tonuino braucht 8 Ohm passiv Lautsprecher. Das sind Standard-Lautsprecher von kleinen Stereoanlagen, wie sie in jeder Billig-Stereoanlage vom Flohmarkt zu finden sind (daher würde ich auch empfehlen die Lautsprecher zu beziehen). Das öffnen der Lautsprecher kann sich etwas trickreich gestalten. Im Zweifelsfall einfach mit einem Multimeter messen, ob es der richtige Lautsprecher ist.

Ich habe mich entschlossen die Rückwand herauszusägen und dabei mit der Stichsäge die Kabel im inneren erwischt. Auch wenn es die Kabel aus der Verankerung in der Membran gerissen hatte, war es erstaunlicherweise doch tatsächlich möglich das ganze zu löten. Long story short: Ich würde 2 Lautsprecher kaufen und davon ausgehen, dass einer beim öffnen kaputt geht.

Anbei eine Lasercutter Datei, um die Rückwand wieder aufzubauen. Die Idee ist, dass der USB Stecker und die SD-Karte über Schlitze in der Rückwand erreichbar sind. Damit die selbst gelötete Platine auch die notwendigen Löcher trifft, gibt es vertikale und horizontale Einstellungsmöglichkeiten bei der Konstruktion. Einfach cutten, hin und her schieben bis alles passt und dann festleimen.

TODO: upload svg

Bedienelemente

Die kreisrunden Löcher für die Knöpfe habe ich mit einem entsprechenden Bohrmaschinenaufsatz ausgesägt. Den RFID Card Reader mit (sehr starkem) doppelsetitigem Klebeband von innen an die Box geklebt.

Kindersicher

Um sicherzustellen, dass Kinder bei der Benutzund den Tonuino nicht zerstören, habe ich mich entschlossen einen Fließ an der Unterkante des Lautsprechers zu befestigen. Der TOnuino wird dann mit der Rückseite bündig an ein schweres Möbelstück gestellt (z.B. Sofa) und das Möbelstück auf den fließ gestellt. Damit bleibt der Tonuino in Position, kann nicht verschoben werden, und folgende Problem erledigen sich (hoffentlich):

- Kind lässt Tonuino auf den Boden fallen.
- Kind verbiegt USB Stecker mit Stromzufuhr

Musik und Figuren (Work in Progress)

Musik

Etwas unpraktisch ist, dass die Ordner/Dateinamen nur aus fortlaufend numerierten 2/3 stelligen Ziffern bestehen sollen. Die Erweiterung dieser Benennung ist sehr fehleranfällig.

Anbei ein Python-Skript, dass alle Ordner und Dateien in einen temporären Ordner kopiert und in das SD-Karten kompatible Format umbenennt . Damit kann mann den SD-Karten Inhalt am PC mit langen Dateinamen verwalten und bei Bedarf einfach einen Export für den Tonuino generieren.

```
"""
Developed and tested under Linux. Should work in Windows as well.

Use this script to export an arbitrary named mp3 folder structure to Tonuino SD-card.
Requirements to mp3 folder structure:
    - requires "advert" and "mp3" folder for sd card
    - maximal directory depth 1, i.e. no subfolders
All not mp3 files are ignored for the export.

Basic usage:
-------------

$ python to_tonuino.py

Plan for copying files:
/home/user/data/unterhaltung/musik/tonuino_sdCard/250424_dev/01_party/aSong.mp3 ->
/tmp/250502-1901_tonuino_sdCard_952_zqjt/01/001.mp3
/home/user/data/unterhaltung/musik/tonuino_sdCard/250424_dev/01_party/bSong.mp3 ->
/tmp/250502-1901_tonuino_sdCard_952_zqjt/01/002.mp3
/home/user/data/unterhaltung/musik/tonuino_sdCard/250424_dev/02_ambiente/ambiente1.mp3 ->
/tmp/250502-1901_tonuino_sdCard_952_zqjt/02/001.mp3
/home/user/data/unterhaltung/musik/tonuino_sdCard/250424_dev/02_ambiente/ambiente2.mp3 ->
```

```
/tmp/250502-1901_tonuino_sdCard_952_zqjt/02/002.mp3
/home/user/data/unterhaltung/musik/tonuino_sdCard/250424_dev/advert -> /tmp/250502-1901_tonuino_sdCard_952_zqjt/advert
/home/user/data/unterhaltung/musik/tonuino_sdCard/250424_dev/mp3 -> /tmp/250502-1901_tonuino_sdCard_952_zqjt/mp3

Playlist:
01: 01_party
001.mp3: aSong.mp3
002.mp3: bSong.mp3
02: 02_ambiente
001.mp3: ambiente1.mp3
002.mp3: ambiente2.mp3

Copying files ...
Finished copying files to /tmp/250502-1901_tonuino_sdCard_952_zqjt



Advanced usage with command line arguments:
-------------------------------------------

For testing call as:
$ python to_tonuino.py /run/media/user/EAGET --test
where /run/media/user/EAGET is your sdCard export-folder.

If you like the generated copy plan, use
$ python to_tonuino.py /run/media/user/EAGET
for actual export.
"""

import os
import shutil
from pathlib import Path
import tempfile
from typing import List, Tuple
from datetime import datetime
import argparse



SKIP_DIRS = ["advert", "mp3"]
```

```
def build_mp3_copy_plan(source_dir: Path, target_dir: Path, skip_dirs: List[str]) ->
List[Tuple[Path, Path]]:
    plan = []
    # dirs = [item for item in source_dir.iterdir() if item.is_dir()]
    dirs = sorted([item for item in source_dir.iterdir() if item.is_dir()])
    for dd, dir in enumerate(dirs):
        if dir.name in skip_dirs:
            continue
        fns = sorted(dir.glob('*.mp3'))
        for ff, fn in enumerate(fns):
            fp_from = dir / fn
            fp_to = target_dir / f"{dd+1:02}" / f"{ff+1:03}.mp3"
            plan.append((fp_from, fp_to))
    return plan

def print_playlist(plan: List[Tuple[Path, Path]]):

    playlist = ""
    dir_to_prev = None;
    for fp_from, fp_to in plan:
        if dir_to_prev != fp_to.parent.name:
            dir_to_prev = fp_to.parent.name
            playlist += f"{fp_to.parent.name}: {fp_from.parent.name}\n"
        playlist += f"\t{fp_to.name}: {fp_from.name}\n"
    return playlist[:-1]




if __name__ == "__main__":

    parser = argparse.ArgumentParser(description="Process an optional filepath.")
    parser.add_argument(
        "target_dir",
        nargs="?",
        type=str,
        default=None,  # default value if not provided
        help="Where to export sd Card content to."
```

```
    )
    parser.add_argument(
        "--test",  # Flag name
        action="store_true",  # Action to take (sets it to True if present)
        help="If testmode: do not copy any files and do not create any folders."
    )
    args = parser.parse_args()
    target_dir = args.target_dir
    mode_test = args.test




    skip_dirs = SKIP_DIRS
    source_dir = Path.cwd()
    if target_dir is None:
        target_dir = tempfile.mkdtemp(prefix=f"{datetime.now().strftime('%y%m%d-
%H%M')}_tonuino_sdCard_")
    target_dir = Path(target_dir)
    plan = build_mp3_copy_plan(source_dir, target_dir, skip_dirs=skip_dirs)

    print("\nPlan for copying files:")
    for fp_src, fp_dst in plan:
        print(f"\t{fp_src} -> {fp_dst}")
    for dir in skip_dirs:
        print(f"\t{source_dir/dir} -> {target_dir/dir}")

    playlist = print_playlist(plan)
    print(f"\nPlaylist:\n{playlist}")

    if mode_test:
        print("\n" + "Test mode on: no copying of files".center(50, "=") + "\n")
        exit(0)

    print("\nCopying files ...")
    for dir in skip_dirs:
        shutil.copytree(source_dir / dir, target_dir / dir)
    for fp_src, fp_dst in plan:
        os.makedirs(os.path.dirname(fp_dst), exist_ok=True)
        shutil.copy2(fp_src, fp_dst)
    print(f"Finished copying files to {target_dir}")
```

Figuren

Der Tonuino akzeptiert Tags in verschiedenen Formaten. Ich persönlich habe schon getestet: NATG215 und MIFARE Classic 1K. Ein paar Ideen, um Kindertaugliche Datenträger zu erstellen sind:

- RFID Karten mit schönen Aufklebern bekleben.
- RFID Coins (kleine nicht klebende Plättchen) kaufen und Spielfiguren (z.B. Ü-Ei) daraufkleben.
- RFID Aufkleber auf Spielzeug kleben, z.B. Duplo Männchen.
- Ntag 216 RFID Implantate (Achtung auf 13,56 MHz achten, nicht 13,... kHz), Größe ca. 12mm x 2mm in Plastiktiere mit Lötkolben einschmelzen.

Folgende Erfahrungen haben **WIR ALLE** mit den verschiedenen Konzepten gemacht. (Bitte tragt hier gerne eure Erfahrungen mit ein.

Konzept	Alter Kind	Erfahrung	Ansprechpartner
RFID Coins (kleine nicht klebende Plättchen) kaufen und Spielfiguren (z.B. Ü-Ei) daraufkleben.	1 Jahr	Verschluckgefahr falls sich RFID Coin ablöst.	Markus Dutschke
RFID Aufkleber auf Spielzeug kleben, z.B. Duplo Männchen.			
Ntag 216 RFID Implantate (Achtung auf 13,56 MHz achten, nicht 13,... kHz), Größe ca. 12mm x 2mm in Plastiktiere mit Lötkolben einschmelzen.			