**Project Specification: Voice-Controlled Piano Page Turner**

**Overview**

This project implements a voice-controlled page-turner for piano players. The system is based on a Teensy 4.1 microcontroller with a Teensy Audio Shield and a MAX9814 microphone (Adafruit). It connects via USB-C to a tablet, acting as a USB HID keyboard that sends page-up and page-down events.

The device recognises four spoken commands: next, weiter, zurück, and back. These are trained and recognised using TensorFlow Lite Micro (TFLM) with embedded MFCC (Mel-frequency cepstral coefficients) features. The firmware is developed using PlatformIO.

**Hardware Components**

* **Microcontroller**: Teensy 4.1
* **Audio interface**: Teensy Audio Shield
* **Microphone**: MAX9814
* **User Interface**:
  + Red LED: Recording active
  + Green LED: Listening mode active
  + Yellow LED: Model upload in progress
  + Push button: Initiates recording
* **Connectivity**:
  + USB-C to tablet (HID interface)
  + Serial connection to PC for training/testing
  + Debug serial output (via USB-serial converter)
  + U**ART to USB adapter:** usedduring training to get a log
* **Storage:** SD card32GB A1 in Teensy Audio Board:

**Audio & MFCC Configuration**

* **Format**: 16-bit PCM mono
* **Sampling Rate**: 16 kHz
* **Sample Duration**: 2 seconds per recording
* **MFCC Parameters**:
  + Window size: 30 ms
  + Stride: 10 ms
  + Coefficients: 13

**File Structure (on PC)**

/dataset/

/next/

/back/

/weiter/

/zurück/

/silence/

/background/

/model/

model\_v1.tflite

model\_v2.tflite

* Augmented data is stored in the same label directories
* Model files are versioned; the latest is renamed to model.tflite before upload

**Use Cases**

**1. Recording**

* Triggered by pressing the push button.
* Red LED lights up.
* Audio is streamed to the PC over serial.
* Recording ends after button release or 2s timeout.
* PC:
  + Receives audio
  + Pads/trims to 2s
  + Validates sample count
  + Saves .wav file under current label mode
  + Plays back the recording

**2. Testing**

* Teensy acts like in recording mode.
* PC uses incoming audio to run inference with current model.
* Detected category is printed.
* Same MFCC preprocessing is used as in training and Teensy.

**3. Training**

* Activated from the PC menu by pressing t
* PC trains a TensorFlow model using Speech Commands format
* Augmentation is run automatically before training
* Output: model\_vN.tflite in /model

**4. Transfer Model**

* Activated from the PC menu by pressing u
* Yellow LED lights up
* PC chunks model into 512-byte blocks and sends with CRC
* Teensy reassembles, validates full CRC, renames after successful upload
* Teensy confirms via serial
* Model is stored in the SD card on the audio board, that is connected to teensy’s SPI

**5. Listening**

* Default mode after startup if model is present
* Green LED is on
* Teensy runs real-time MFCC + model inference
* If keyword detected:
  + next/weiter → HID PAGE\_DOWN
  + back/zurück → HID PAGE\_UP
* If no model is found, listening is disabled

**Serial Communication**

* Packet Structure: [2 bytes header][payload length][payload][CRC-8]
* Supported Commands:
  + RECORDING\_START
  + RECORDING\_STOP
  + MODEL\_UPLOAD\_START
  + MODEL\_UPLOAD\_END
  + ACK / NACK / ERROR / READY
* Error Handling:
  + On CRC or protocol failure: discard input for 1s, then reset state
  + Teensy maintains a state machine (recording/listening/uploading)

**Audio Augmentation**

* Automatic before training
* All next/back/weiter/zurück files are mixed with silence and background (piano sound)
* SNR levels: 5 and 10
* Output file format: next/next\_5\_aug1.wav, etc.
* Augmented files are included automatically in training
* Additional augmented files are created by shifting audio snippets to get the model accustomed to the word not always been at the same position.

**Debouncing & Safeguards**

* **Button debounce**: Timing-based logic
* **Inference debounce**: 1s minimum wait between PAGE\_UP/PAGE\_DOWN triggers
* **Startup check**:
  + If model.tflite is found, green LED turns on
  + Otherwise, listening is disabled until upload

**Silence Detection**

* Silence is determined based on RMS threshold (e.g. −40 dB)
* Python applies:

rms = np.sqrt(np.mean(audio\*\*2))

rms\_db = 20 \* np.log10(rms + 1e-10)

if rms\_db < threshold\_db:

label = "silence"

* Background is defined as non-speech ambient piano noise
* Both are set manually via the PC menu mode

**Logging**

* Debug logs are printed to terminal over second serial port
* Format: [INFO] Recording started at t=123ms
* Logging is not saved to disk

**Python Console Interface**

* Keyboard-controlled menu (non-blocking input)
* Microcontroller-style loop (no threads)
* Polls keyboard and serial input alternately
* Menu states:
  + Change label mode
  + Start training
  + Start model transfer
  + Play back dataset
  + Switch to test mode

**Notes**

* All models and data structures follow TensorFlow Speech Commands conventions
* Teensy only switches to listening mode if model is present
* Teensy always loads model.tflite on boot
* CRC is applied per model chunk and for full model file after reassembly
* PC always sends the most recent model version

# Proposed PIN layout:

Here’s a suggested mapping for all of your peripherals on a Teensy 4.0 (Rev D) with the Audio Shield stacked. I’ve grouped by function and tried to avoid any pin-function conflicts. Feel free to shuffle around any of the general-purpose I/O pins (LEDs, switch, display control lines) to suit your layout or breadboard wiring.

| **Function** | **Teensy 4.0 Pin** | **Notes** |
| --- | --- | --- |
| **Audio (SGTL5000)** |  |  |
| MCLK (Audio master clock) | 23 | I2S master clock, 11.29 MHz [wholesale.pimoroni.com](https://wholesale.pimoroni.com/products/audio-adaptor-board-for-teensy-4-0?utm_source=chatgpt.com) |
| BCLK (Audio bit clock) | 21 | I2S bit clock, 1.41 / 2.82 MHz [wholesale.pimoroni.com](https://wholesale.pimoroni.com/products/audio-adaptor-board-for-teensy-4-0?utm_source=chatgpt.com) |
| LRCLK (LR word clock) | 20 | I2S left/right clock, 44.1 kHz [wholesale.pimoroni.com](https://wholesale.pimoroni.com/products/audio-adaptor-board-for-teensy-4-0?utm_source=chatgpt.com) |
| DIN (TX to codec) | 7 | I2S data out from Teensy → codec [wholesale.pimoroni.com](https://wholesale.pimoroni.com/products/audio-adaptor-board-for-teensy-4-0?utm_source=chatgpt.com) |
| DOUT (RX from codec) | 8 | I2S data in from codec → Teensy [wholesale.pimoroni.com](https://wholesale.pimoroni.com/products/audio-adaptor-board-for-teensy-4-0?utm_source=chatgpt.com) |
| I²C SDA (control data) | 18 | SGTL5000 control bus [wholesale.pimoroni.com](https://wholesale.pimoroni.com/products/audio-adaptor-board-for-teensy-4-0?utm_source=chatgpt.com) |
| I²C SCL (control clock) | 19 | SGTL5000 control bus [wholesale.pimoroni.com](https://wholesale.pimoroni.com/products/audio-adaptor-board-for-teensy-4-0?utm_source=chatgpt.com) |
| **SD card (on Audio Shield)** |  |  |
| SPI SCK | 13 | shared SPI bus [wholesale.pimoroni.com](https://wholesale.pimoroni.com/products/audio-adaptor-board-for-teensy-4-0?utm_source=chatgpt.com) |
| SPI MISO | 12 | shared SPI bus [wholesale.pimoroni.com](https://wholesale.pimoroni.com/products/audio-adaptor-board-for-teensy-4-0?utm_source=chatgpt.com) |
| SPI MOSI | 11 | shared SPI bus [wholesale.pimoroni.com](https://wholesale.pimoroni.com/products/audio-adaptor-board-for-teensy-4-0?utm_source=chatgpt.com) |
| SD CS | 10 | SD card chip-select [wholesale.pimoroni.com](https://wholesale.pimoroni.com/products/audio-adaptor-board-for-teensy-4-0?utm_source=chatgpt.com) |
| **SPI display (e.g. ILI9xxx/ST77xx)** |  |  |
| SPI SCK | 13 | same SPI bus |
| SPI MOSI | 11 | same SPI bus |
| SPI MISO (if needed) | 12 | same SPI bus |
| Display CS | 24 | any free GPIO (bottom pads) |
| Display DC (D/C) | 25 | data/command select |
| Display RST | 26 | reset line |
| **Extra UART for log output** |  |  |
| Serial3 RX (Serial3 on pins 15/14) | 15 | RX3 for external log reader [pjrc.com](https://www.pjrc.com/teensy/td_libs_SoftwareSerial.html?utm_source=chatgpt.com) |
| Serial3 TX | 14 | TX3 for log output [pjrc.com](https://www.pjrc.com/teensy/td_libs_SoftwareSerial.html?utm_source=chatgpt.com) |
| **User I/O** |  |  |
| LED 1 | 2 | simple status LED |
| LED 2 | 4 | another status LED |
| LED 3 | 5 | third LED |
| Switch (button) | 9 | with INPUT\_PULLUP, interrupt-capable |