AdafruitVS1053 Library Documentation

Release 1.0

Tony DiCola

Oct 20, 2019

Contents

1 Dependencies **3**
- 1.1 Installing from PyPI . . . 3

2 Usage Example **5**

3 Contributing **7**

4 Building locally **9**
- 4.1 Zip release files . . . 9
- 4.2 Sphinx documentation . . . 9

5 Table of Contents **11**
- 5.1 Simple test . . . 11
- 5.2 `adafruit_vs1053` . . . 13
  - 5.2.1 Implementation Notes . . . 13

6 Indices and tables **15**

Python Module Index **17**

Index **19**

Driver for interacting and playing media files with the VS1053 audio codec over a SPI connection.

NOTE: This is not currently working for audio playback of files. Only sine wave test currently works. The problem is that pure Python code is currently too slow to keep up with feeding data to the VS1053 fast enough. There's no interrupt support so Python code has to monitor the DREQ line and provide a small buffer of data when ready, but the overhead of the interpreter means we can't keep up. Optimizing SPI to use DMA transfers could help but ultimately an interrupt-based approach is likely what can make this work better (or C functions built in to custom builds that monitor the DREQ line and feed a buffer of data).

CHAPTER 1

Dependencies

This driver depends on:

- Adafruit CircuitPython
- Bus Device

Please ensure all dependencies are available on the CircuitPython filesystem. This is easily achieved by downloading the Adafruit library and driver bundle.

1.1 Installing from PyPI

On supported GNU/Linux systems like the Raspberry Pi, you can install the driver locally from PyPI. To install for current user:

```
pip3 install adafruit-circuitpython-vs1053
```

To install system-wide (this may be required in some cases):

```
sudo pip3 install adafruit-circuitpython-vs1053
```

To install in a virtual environment in your current project:

```
mkdir project-name && cd project-name
python3 -m venv .env
source .env/bin/activate
pip3 install adafruit-circuitpython-vs1053
```

CHAPTER 2

Usage Example

See examples/sdfile_play.py.

CHAPTER 3

Contributing

Contributions are welcome! Please read our Code of Conduct before contributing to help this project stay welcoming.

CHAPTER 4

Building locally

4.1 Zip release files

To build this library locally you'll need to install the circuitpython-build-tools package.

```
python3 -m venv .env
source .env/bin/activate
pip install circuitpython-build-tools
```

Once installed, make sure you are in the virtual environment:

```
source .env/bin/activate
```

Then run the build:

```
circuitpython-build-bundles --filename_prefix adafruit-circuitpython-vs1053 --library_
↪location .
```

4.2 Sphinx documentation

Sphinx is used to build the documentation based on rST files and comments in the code. First, install dependencies (feel free to reuse the virtual environment from above):

```
python3 -m venv .env
source .env/bin/activate
pip install Sphinx sphinx-rtd-theme
```

Now, once you have the virtual environment activated:

```
cd docs
sphinx-build -E -W -b html . _build/html
```

This will output the documentation to `docs/_build/html`. Open the index.html in your browser to view them. It will also (due to -W) error out on any warning like Travis will. This is a good way to locally verify it will pass.

CHAPTER 5

Table of Contents

5.1 Simple test

Ensure your device works with this simple test.

Listing 1: examples/vs1053_simpletest.py

```
# Example of sound playback from VS1053 FeatherWing.  Can be modified to work
# with the breakout by changing the SD card and SPI pins mentioned below.
# NOTE:
# Unfortunately this doesn't work--the loop isn't fast enough to feed the VS1053
# data at the rate it needs for playback.  You'll see very erratic behavior with
# the VS1053 making static, stopping, and eventually requiring a hard reset.
# We'll need to look into interrupt support perhaps to monitor DREQ like in the
# arduino library.  Basic sine wave playback does however work and monitoring
# of attributes like status register and other VS1053 state is accessible.
import board
import busio
import digitalio
import storage

import adafruit_sdcard
import adafruit_vs1053


# Define pins connected to VS1053:

# For FeatherWing with Feather M0:
SDCS  = board.D5    # Pin connected to SD card CS line.
MP3CS = board.D6    # Pin connected to VS1053 CS line.
DREQ  = board.D9    # Pin connected to VS1053 DREQ line.
XDCS  = board.D10   # Pin connected to VS1053 D/C line.


```

(continues on next page)

(continued from previous page)

```
# Other configuration:
PLAYBACK_FILE = '/sd/test.wav'  # Name of file to play.
                                # This should be the full path
                                # including /sd prefix if on
                                # sd card.

BUFFER_SIZE   = 128  # Size in bytes of the MP3 data buffer for sending data to
                     # the VS1053.


# Setup SPI bus (hardware SPI).
spi = busio.SPI(board.SCK, MOSI=board.MOSI, MISO=board.MISO)

# Setup SD card and mount it in the filesystem.
sd_cs = digitalio.DigitalInOut(SDCS)
sdcard = adafruit_sdcard.SDCard(spi, sd_cs)
vfs = storage.VfsFat(sdcard)
storage.mount(vfs, '/sd')

# To list all the files on the SD card root uncomment:
#import os
#print('SD card root contains:')
#print(os.listdir('/sd'))

# Setup VS1053.
vs1053 = adafruit_vs1053.VS1053(spi, MP3CS, XDCS, DREQ)

# Set volume of left and right channels.
# Value ranges from 0 to 255 for each channel, the lower the higher volume.
vs1053.set_volume(0, 0)

# Play a test tone (this works).
print('Playing test tone for two seconds...')
vs1053.sine_test(0x44, 2.0)
print('Done playing tone!')

# Play back a MP3 file by starting playback, then reading a buffer of data
# at a time and sending it to the VS1053.
# Unfortunately this doesn't work--the loop isn't fast enough to feed the VS1053
# data at the rate it needs for playback.  You'll see very erratic behavior with
# the VS1053 making static, stopping, and eventually requiring a hard reset.
# We'll need to look into interrupt support perhaps to monitor DREQ like in the
# arduino library.
print('Playing {}...'.format(PLAYBACK_FILE))
vs1053.start_playback()
with open(PLAYBACK_FILE, 'rb') as infile:
    music_data = infile.read(BUFFER_SIZE)
    while music_data is not None and music_data != '':
        while not vs1053.ready_for_data:
            pass
        vs1053.play_data(music_data, end=len(music_data))
        music_data = infile.read(BUFFER_SIZE)

print('Done!')
```

5.2 adafruit_vs1053

Driver for interacting and playing media files with the VS1053 audio codec over a SPI connection.

> NOTE: This is not currently working for audio playback of files. Only sine wave test currently works. The problem is that pure Python code is currently too slow to keep up with feeding data to the VS1053 fast enough. There's no interrupt support so Python code has to monitor the DREQ line and provide a small buffer of data when ready, but the overhead of the interpretor means we can't keep up. Optimizing SPI to use DMA transfers could help but ultimately an interrupt-based approach is likely what can make this work better (or C functions built in to custom builds that monitor the DREQ line and feed a buffer of data).

- Author(s): Tony DiCola

5.2.1 Implementation Notes

Hardware:

- Adafruit Music Maker FeatherWing - Synth Player
- Music Maker FeatherWing w/ Amp - Synth Player
- VS1053B MP3/WAV/OGG/MIDI Player & Recorder (CODEC) Chip
- VS1053 Codec + MicroSD Breakout - Play + Record - v4

Software and Dependencies:

- Adafruit CircuitPython firmware for the supported boards: https://github.com/adafruit/circuitpython/releases
- Adafruit's Bus Device library: https://github.com/adafruit/Adafruit_CircuitPython_BusDevice

class adafruit_vs1053.**VS1053**(*spi*, *cs*, *xdcs*, *dreq*)
Class-level buffer for read and write commands.

byte_rate
Return the bit rate in bytes per second (computed each second). Useful to know if a song is being played and how fast it's happening.

decode_time
Return the decode time register value. This is the amount of time the current file has been played back in seconds.

play_data(*data_buffer*, *start=0*, *end=None*)
Send a buffer of file data to the VS1053 for playback. Make sure the ready_for_data property is True before calling!

ready_for_data
Return True if the VS1053 is ready to accept data, false otherwise.

reset()
Perform a longer full reset with clock and volume reset too.

set_volume(*left*, *right*)
Set the volume of the left and right channels to the provided byte value (0-255), the lower the louder.

sine_test(*n*, *seconds*)
Play a sine wave for the specified number of seconds. Useful to test the VS1053 is working.

soft_reset()
Perform a quick soft reset of the VS1053.

start_playback()
Prepare for playback of a file. After calling this check the ready_for_data property continually until true and then send in buffers of music data to the play_data function.

stop_playback()
Stop any playback of audio.

version
Return the status register version value.

CHAPTER 6

Indices and tables

- genindex
- modindex
- search

Python Module Index

a

adafruit_vs1053, 12

Index

A

adafruit_vs1053 (*module*), 12

B

byte_rate (*adafruit_vs1053.VS1053 attribute*), 13

D

decode_time (*adafruit_vs1053.VS1053 attribute*), 13

P

play_data() (*adafruit_vs1053.VS1053 method*), 13

R

ready_for_data (*adafruit_vs1053.VS1053 attribute*), 13
reset() (*adafruit_vs1053.VS1053 method*), 13

S

set_volume() (*adafruit_vs1053.VS1053 method*), 13
sine_test() (*adafruit_vs1053.VS1053 method*), 13
soft_reset() (*adafruit_vs1053.VS1053 method*), 13
start_playback() (*adafruit_vs1053.VS1053 method*), 13
stop_playback() (*adafruit_vs1053.VS1053 method*), 14

V

version (*adafruit_vs1053.VS1053 attribute*), 14
VS1053 (*class in adafruit_vs1053*), 13