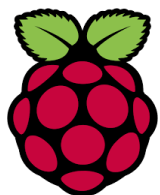


# WLOラズパイ倶楽部 ラズパイでラジオ放送?!



2018年1月25日

- ・Raspberry PiからFMラジオ放送

  - アンテナ

  - 必要なプログラムをインストール

  - サンプルファイルを放送

- ・いろんな音声を送信してみる

  - マイクから入力

  - open\_jtalkで文字列を放送

- ・音声ファイルの扱い方

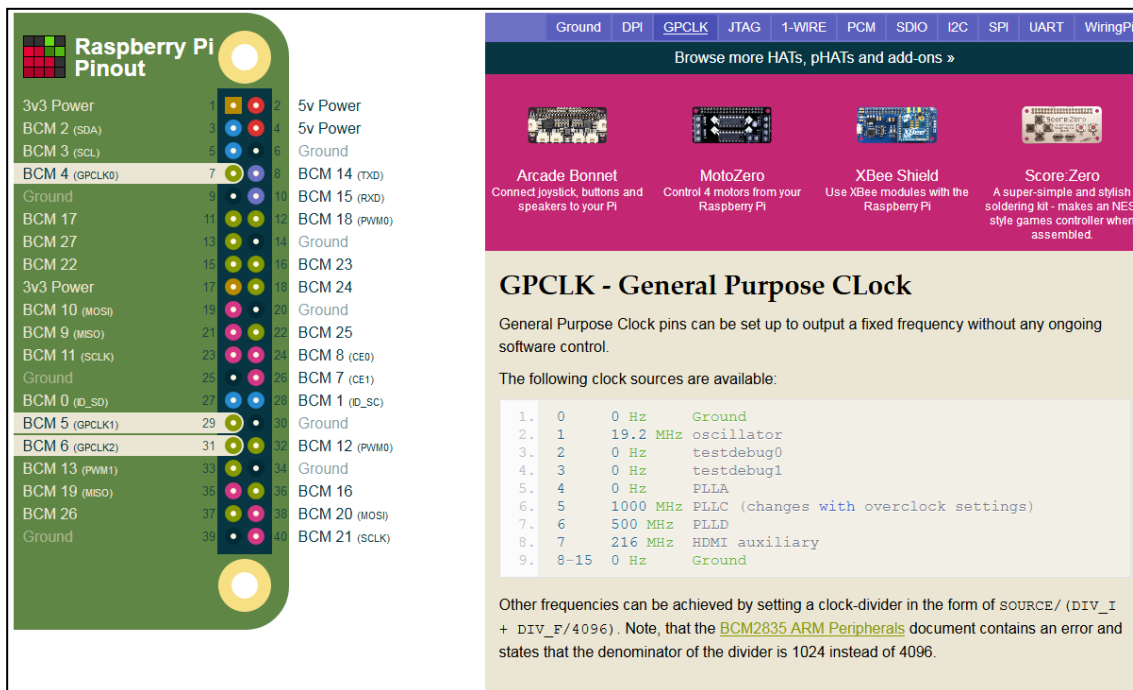
  - sox

  - ポッドキャストを放送してみる

## ・Raspberry Piの汎用クロック出力(GPCLK0)

Raspberry PiのプロセッサBCM2835/BCM2836/BCM 2837  
に備わっている汎用クロック出力 GPCLK  
(カメラやHDMI、LCDなど周辺機器の制御に使われる)

設定によりGPIOコネクタのpin7(GPIO4)にGPCLK0が出力できる  
→これを音声出力で周波数変調(FM)することで放送できる



The screenshot shows the Raspberry Pi Pinout website. On the left is a pinout diagram for a Raspberry Pi 3. The pin 7 (GPIO4) is highlighted in yellow and labeled 'BCM 4 (GPCLK0)'. On the right, the 'GPCLK' tab is selected in the top navigation bar. Below the navigation bar, there are four featured HATs: Arcade Bonnet, MotoZero, XBee Shield, and Score Zero. The main content area is titled 'GPCLK - General Purpose CLock'. It explains that General Purpose Clock pins can be set up to output a fixed frequency without any ongoing software control. It lists the following clock sources available:

Pin	Frequency	Source
0	0 Hz	Ground
1	19.2 MHz	oscillator
2	0 Hz	testdebug0
3	0 Hz	testdebug1
4	0 Hz	PLLA
5	1000 MHz	PLL (changes with overclock settings)
6	500 MHz	PLLD
7	216 MHz	HDMI auxiliary
8-15	0 Hz	Ground

Other frequencies can be achieved by setting a clock-divider in the form of `SOURCE / (DIV_I + DIV_F / 4096)`. Note, that the [BCM2835 ARM Peripherals](#) document contains an error and states that the denominator of the divider is 1024 instead of 4096.

<https://pinout.xyz/pinout/gpclk>

<http://git.io/wlopi>

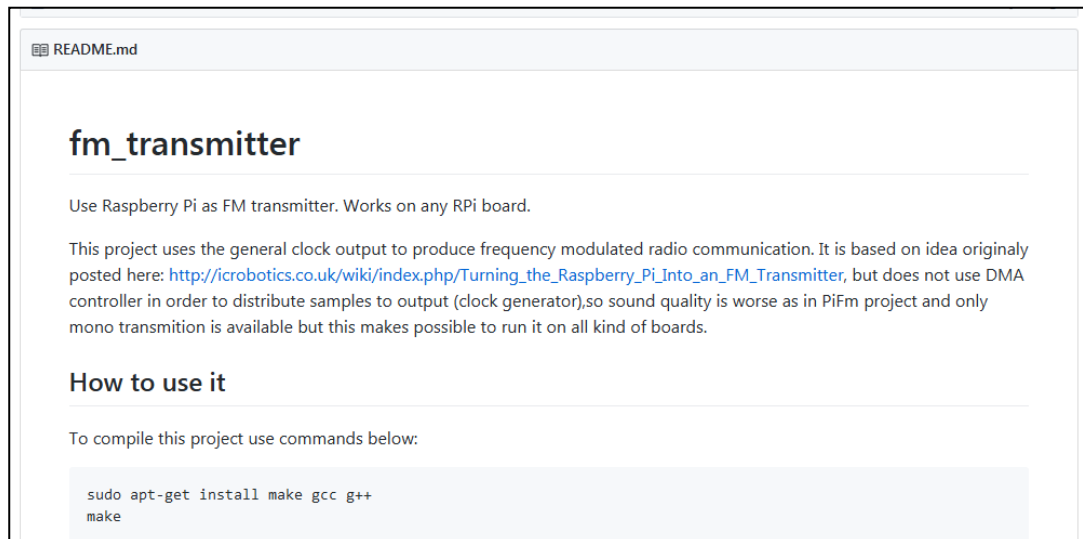
## ・アンテナ

GPIO4 (pin7)に20cm程度のリード線を接続する



## ・送信用ソフトウェア

[https://github.com/markondej/fm\\_transmitter](https://github.com/markondej/fm_transmitter)



**インストール手順** ※/home/pi/fm\_transmitterにインストールされます。

```
$ sudo apt install -y git make gcc g++
$ git clone https://github.com/markondej/fm\_transmitter.git
$ cd fm_transmitter
$ make
```

**簡易版インストール手順** ※後のページで使うソフトウェアもインストールされます。

```
$ cd
$ wget https://git.io/wlofmi
$ bash wlofmi
```

## ・サンプルファイル

[https://github.com/WLO-RaspiClub/20180125\\_RaspiFMStation/raw/master/SwelteringAbandon.wav](https://github.com/WLO-RaspiClub/20180125_RaspiFMStation/raw/master/SwelteringAbandon.wav)

※ jukedeck <http://jukedeck.com> で作成しました

### ダウンロード手順

※簡易版インストール手順ではダウンロード済みです

```
$ cd
$ cd fm_transmitter
$ wget https://github.com/WLO-RaspiClub/20180125_RaspiFMStation/raw/master/SwelteringAbandon.wav
```

## ・放送してみる

```
$ cd
$ cd fm_transmitter
$ sudo ./fm_transmitter -f 94.0 SwelteringAbandon.wav
```

※Raspberry Pi 1/2/Zero用

ポイント1: 引数-fで放送する周波数(MHz)を指定します。既存の放送曲がない周波数を指定します。

ポイント2: Raspberry Pi 3の場合は、周波数を1/2します。

```
$ cd
$ cd fm_transmitter
$ sudo ./fm_transmitter -f 47.0 SwelteringAbandon.wav
```

※Raspberry Pi 3用(94.0MHz/2=47.0)

## ・ファイル名指定ではなく、標準入力(STDIN)を使う

```
$ cd
$ cd fm_transmitter
$ cat SwelteringAbandon.wav | sudo ./fm_transmitter -f 47.0 -
```

※Raspberry Pi 3用

## ・マイク入力音声を使う

※Raspberry Pi 3用

```
$ arecord -l
**** ハードウェアデバイス CAPTURE のリスト ****
カード 1: Device [USB PnP Sound Device], デバイス 0: USB Audio [USB Audio]
  サブデバイス: 1/1
  サブデバイス #0: subdevice #0
$ cd
$ cd fm_transmitter
$ arecord -D plughw:1,0 -c1 -d 0 -r 22050 -f S16_LE | sudo ./fm_transmitter -f 47.0 -
```

ポイント1: 音声は22050Hzのモノラルになるようにarecordを設定します

## • Open JTalk

<http://open-jtalk.sp.nitech.ac.jp/>

**Open JTalk**

**HMM-based Text-to-Speech System**  
**Open JTalk Demonstration Page**

入力された日本語テキストに基づいて自由な音声を生成するHMMテキスト音声合成システム, Open JTalkのデモンストレーションです。 [利用規約](#)

話者: 男性: M001  
声質: 0.55 (-0.8~0.8, 標準: 0.55)  
ピッチシフト: 0 (-24~24, 標準: 0)  
話速: 1 (0.5~2.0, 標準: 1.0)

合成テキスト  
(最大200字)

合成結果

**お知らせ**

- 2012/12/25 [Ver. 1.8]  
Open JTalkのバージョンを1.06に更新しました。  
女性話者「Mei (Happy)」 「Mei (Bashful)」 「Mei (Angry)」 「Mei (Sad)」を追加しました。  
音質を安定させました。
- 2011/12/25 [Ver. 1.7]

インストール手順 ※簡易版インストール手順ではインストール済みです

```
$ sudo apt install -y open-jtalk  
$ sudo apt install -u open-jtalk-mecab-naist-jdic hts-voice-nitech-jp-atr503-m001
```

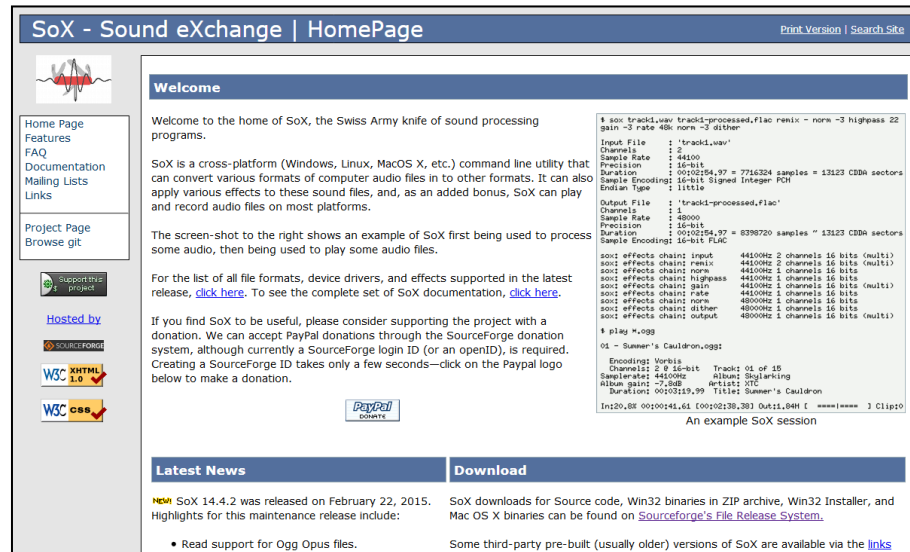
voice.txtに合成したい文字列を入れて、合成し、放送してみる

```
$ cd  
$ ./jtalk.sh voice.txt test.wav  
$ cd fm_transmitter  
$ sudo ./fm_transmitter -f 47.0 ../test.wav
```



## • SoX: Sound eXchange

<http://sox.sourceforge.net/>



**SoX - Sound eXchange | HomePage** [Print Version](#) [Search Site](#)

**Welcome**

Welcome to the home of SoX, the Swiss Army knife of sound processing programs.

SoX is a cross-platform (Windows, Linux, MacOS X, etc.) command line utility that can convert various formats of computer audio files in to other formats. It can also apply various effects to these sound files, and, as an added bonus, SoX can play and record audio files on most platforms.

The screen-shot to the right shows an example of SoX first being used to process some audio, then being used to play some audio files.

For the list of all file formats, device drivers, and effects supported in the latest release, [click here](#). To see the complete set of SoX documentation, [click here](#).

If you find SoX to be useful, please consider supporting the project with a donation. We can accept PayPal donations through the SourceForge donation system, although currently a SourceForge login ID (or an openID), is required. Creating a SourceForge ID takes only a few seconds—click on the PayPal logo below to make a donation.

[PayPal](#) Donate

**Latest News**

**Download**

```
$ sox track1.wav track1-processed.flac remix - norm -3 highpass 22
gain -3 rate 48k norm -3 dither

Input File      : 'track1.wav'
Channels        : 2
Sample Rate     : 44100
Precision       : 16-bit
Duration        : 00:02:54.97 = 7716324 samples = 13123 CDDA sectors
Sample Encoding : 16-bit Signed Integer PCM
Endian Type     : little

Output File     : 'track1-processed.flac'
Channels        : 2
Sample Rate     : 48000
Precision       : 16-bit
Duration        : 00:02:54.97 = 8396720 samples = 13123 CDDA sectors
Sample Encoding : 16-bit FLAC

sox effects chain: input 44100Hz 2 channels 16 bits (multi)
sox effects chain: remix 44100Hz 2 channels 16 bits (multi)
sox effects chain: norm 44100Hz 1 channels 16 bits
sox effects chain: highpass 44100Hz 1 channels 16 bits
sox effects chain: gain 44100Hz 1 channels 16 bits (multi)
sox effects chain: rate 44100Hz 1 channels 16 bits
sox effects chain: norm 48000Hz 1 channels 16 bits
sox effects chain: dither 48000Hz 1 channels 16 bits
sox effects chain: output 48000Hz 1 channels 16 bits (multi)

$ play *.ogg
01 - Summer's Cauldron.ogg
Encoding: Vorbis
Channels: 2 @ 16-bit Track: 01 of 15
Samplerate: 44100Hz album: Skyarking
Album gain: -7.0dB Artist: 170
Duration: 00:03:19.99 Title: Summer's Cauldron
In:20.8K 00:00:45.64 [00:02:38.38] Out:1.04H [==== ] Clip:0

An example SoX session
```

音声ファイルの形式、レート、  
音声品質の変更や  
音声ファイル同士の合成などができる

インストール手順 ※簡易版インストール手順ではインストール済みです

```
$ sudo apt install -y sox libsox-fmt-mp3
```

※soxの標準ではmp3を扱えないので、libsox-fmt-mp3を追加インストールする

- ・基本: sox 入力ファイル 出力ファイル

```
$ sox infile.mp3 outfile.wav
```

ファイルの拡張子から音声フォーマットを類推してくれる

- ・それぞれのファイルフォーマットを指定する

入出力のファイル名の前に音声フォーマットや品質を指定する

```
$ sox infile.mp3 -r 22010 -c 1 outfile.wav
```

-r: 音声レート

-c 1: 音声チャンネル(-c 1でモノラル)

- ・合成、切り出し、エフェクト、無音削除などさまざまな機能がある

参考: SoXチートシート - コマンドラインで音声編集

<https://qiita.com/moutend/items/50df1706db53cc07f105>

## ・ポッドキャストファイルの取得

ブラウザでダウンロードする

(ダウンロードしたファイルは/home/pi/Downloads に格納されます)

## ・soxでファイル形式変換

22050Hzのモノラルのwavファイルに変換します。

```
$ sox /home/pi/Downloads/podcast.mp3 -r 22010 -c 1 podcast.wav
```

## ・変換したファイルを放送する

```
$ cd  
$ cd fm_transmitter  
$ sudo ./fm_transmitter -f 47.0 ../podcast.wav
```