

# ARMマイコン流星電波観測器 (MroSpec) 説明書

目次		(ページ)
1.	特長	2
2.	構成	2
3.	観測までの流れ	3
4.	必要機器	4
5.	使用ソフトウェア	5
6.	観測ソフトのパッケージ	7
7.	機器の確認と初期設定	10
8.	調整	15
9.	設定ファイルの準備	17
10.	観測	19
11.	観測データの容量と手動削除	21
12.	エコー計数とエコー例	22
13.	シャットダウン	25
14.	crontab の設定	27
15.	Ver 1.10 からの変更内容	28

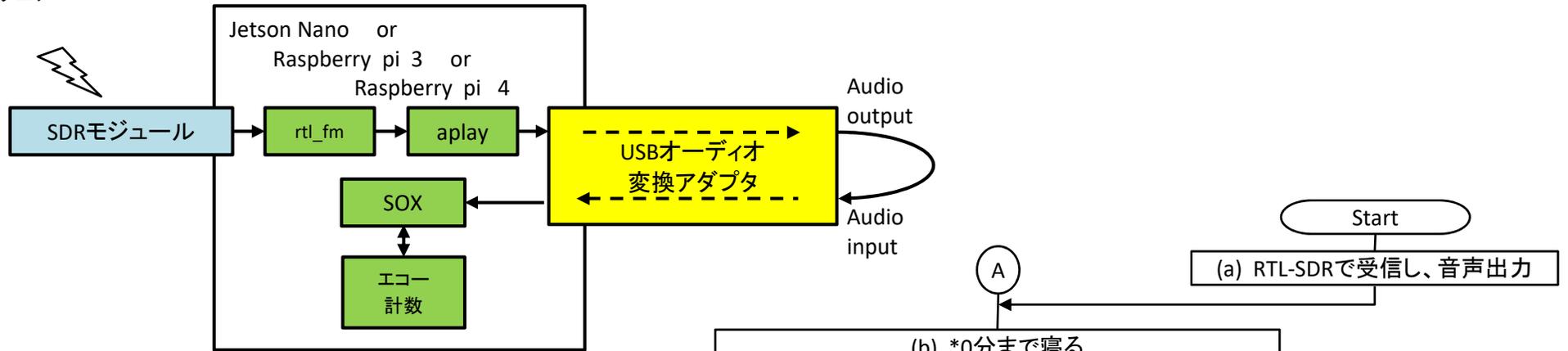
初版 : 2021/11/06 : ソフトVer 1.00 対応  
v1.01 : 2021/11/07 : 誤記修正  
v1.02 : 2021/11/08 : ソフトVer 1.10 対応  
(電源ONによる自動起動対応)  
v1.03 : 2021/11/08 : 誤記修正  
v1.04 : 2022/01/23 : ソフトVer 1.20 対応

## 1. 特長

- ・ ARMマイコン流星電波観測器 (MroSpec) は、Raspberry Pi などのARMマイコンキットにUSB接続のSDRモジュールを接続し、SDRソフトで受信した信号を解析する流星電波観測器です。
- ・ 観測前に簡単な調整を行い、この調整結果を元に送信局毎の設定ファイルを作成することで、観測環境に応じた観測をすぐに始められます。
- ・ ノイズの影響など受信環境による信号強度の変動は、設定ファイルのパラメータを変更することで、観測を中断することなく調整できます。
- ・ 観測データは、HROFFTと同様のスペクトログラムの他、エコー検出情報（時刻、継続時間）を保存すると共に、サーバーへ送信しライブ公開もできます。
- ・ 配布するシェルプログラム、Cプログラム、C++プログラムは、観測、解析など行いやすいよう自由に改造して利用してください。

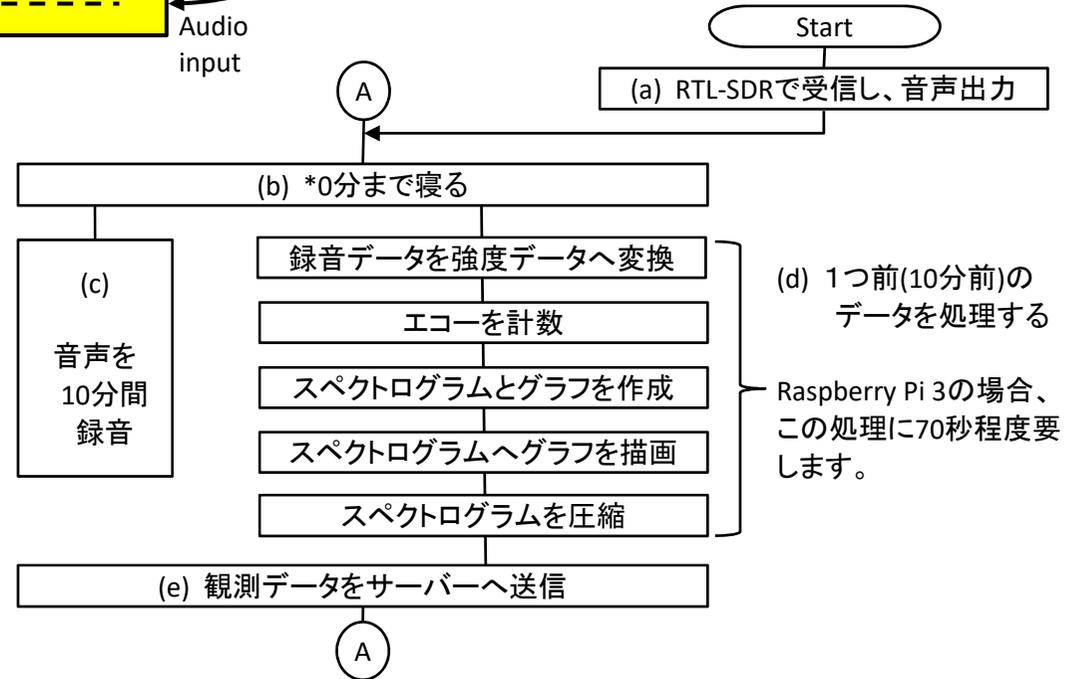
## 2. 構成

### (1) ハードウェア



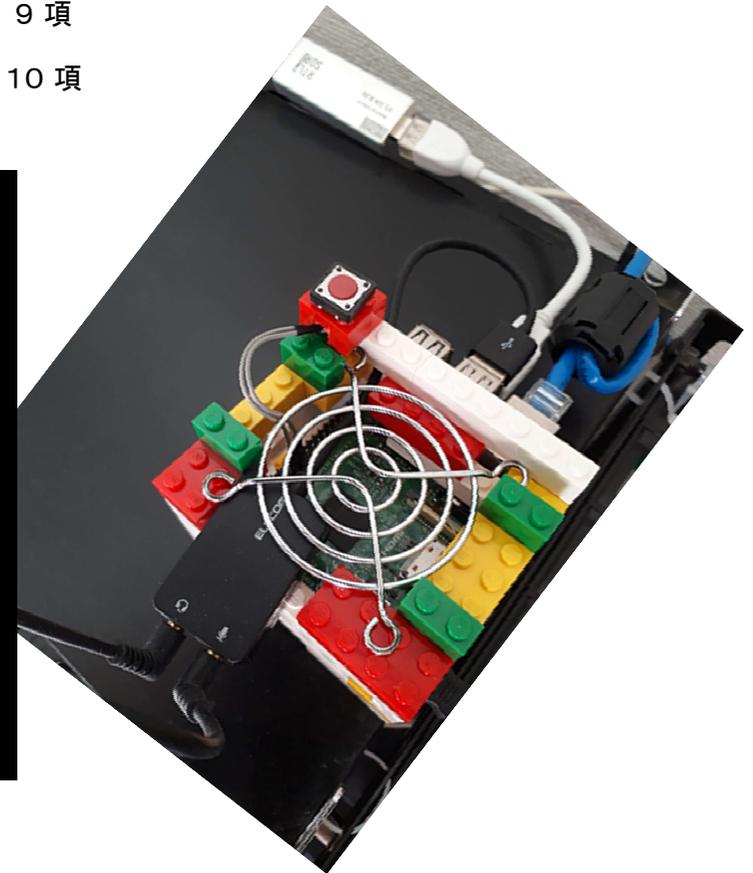
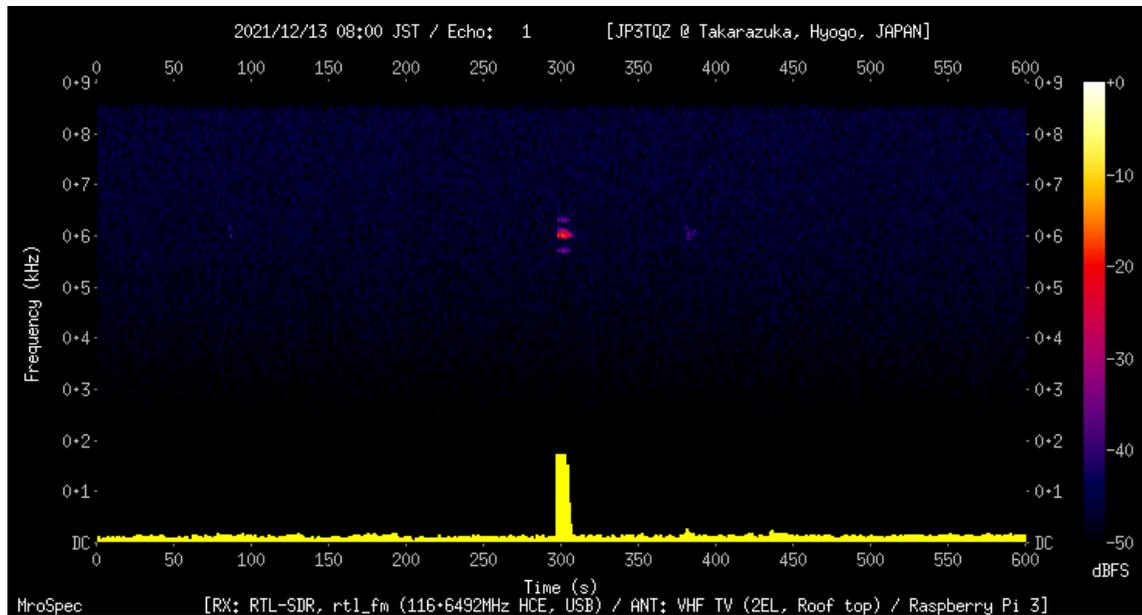
### (2) ソフトウェア (処理フロー)

- RTL-SDRを指定周波数で動作させて受信信号をAudio出力する。
- \*0分まで寝る。
- 起床後、Audio入力からの音声データの録音を開始する。
- 一つ前(10分前)の録音データを使用して各種の処理を行う。
- 観測データをサーバーへ送信する。
- (b)～(e)を繰り返す。



## 3. 観測までの流れ

- |                                 |       |      |
|---------------------------------|-------|------|
| (1) 機器の準備                       | ..... | 4 項  |
| (2) 使用するソフトウェアのインストール           | ..... | 5 項  |
| (3) 観測ソフトのインストール                | ..... | 6 項  |
| (4) 機器の動作確認と機器の情報取得(初期設定)       | ..... | 7 項  |
| (5) 受信状態を確認して各種パラメータを調整         | ..... | 8 項  |
| (6) 調整したパラメータを元に、設定ファイルへ観測条件を記載 | ..... | 9 項  |
| (7) 観測                          | ..... | 10 項 |



## 4. 必要機器

<p>(1) アンテナなどRF系</p> <p>アンテナ</p> <p>同軸ケーブル</p> <p>変換コネクタ</p>	<p>使用する周波数帯に合ったアンテナ。</p> <p>アンテナとSDRモジュールを接続するために使用。</p> <p>同軸ケーブルとSDRモジュールのコネクタ形状が異なる場合、形状を変換するために使用。</p>
<p>(2) SDRモジュール</p> <p>USB接続のSDRモジュール 右記の製品は動作確認済み</p>	<p>RTL-SDR</p> <p>DS-DT305WH . . . 安価なワンセグチューナーのため、受信性能は悪い。</p> <div style="border: 1px solid black; padding: 5px; width: fit-content; margin-left: auto; margin-right: auto;"> <p>Airspy mini は動作しないことを確認済み。</p> </div>
<p>(3) ARMマイコンキット</p> <p>Linux OSが稼働するキット 右記の製品は動作確認済み</p> <p>Raspberry Pi 4 は動作の可能性あり。</p>	<p>Jetson Nano ( OS: Ubuntu 18.04.5 LTS ) <a href="https://www.nvidia.com/ja-jp/autonomous-machines/embedded-systems/jetson-nano/">https://www.nvidia.com/ja-jp/autonomous-machines/embedded-systems/jetson-nano/</a> <a href="https://developer.nvidia.com/embedded/downloads">https://developer.nvidia.com/embedded/downloads</a></p> <p>Raspberry Pi 3 ( OS: Raspberry Pi OS with desktop ) <a href="https://www.raspberrypi.com/software/operating-systems/">https://www.raspberrypi.com/software/operating-systems/</a></p> <p>Raspberry Pi 4 ( OS: Raspberry Pi OS with desktop ) <a href="https://www.raspberrypi.com/software/operating-systems/">https://www.raspberrypi.com/software/operating-systems/</a></p>
<p>(4) USBオーディオ変換アダプタ</p> <p>USBと音声の変換アダプタ 右記の製品は動作確認済み</p>	<p>USB-AADC02BK (エレコム) . . . 販売中 (2021年10月) <a href="https://www.elecom.co.jp/products/USB-AADC02BK.html">https://www.elecom.co.jp/products/USB-AADC02BK.html</a></p> <p>PL-US35AP (プラネックス) . . . 販売終了 <a href="https://www.planex.co.jp/product/usb/pl-us35ap/">https://www.planex.co.jp/product/usb/pl-us35ap/</a></p>
<p>(5) オーディオケーブル</p> <p>ステレオオーディオケーブル</p>	<p>USBオーディオ変換アダプタのヘッドフォン端子とマイク端子を接続して使用。 USBオーディオ変換アダプタのプラグに合ったケーブルを準備。</p>

## 5. 使用ソフトウェア

ソフト名称	説明	インストール方法	
<b>(1) SDRソフト</b>			
rtl_fm	<ul style="list-style-type: none"> <li>・サイト <a href="http://kmkeen.com/rtl-demod-guide/">http://kmkeen.com/rtl-demod-guide/</a></li> <li>・rtl-sdrソフトに同梱されている。</li> <li>・処理負荷が軽いSDRソフトであり、このソフトで電波を受信。</li> </ul>	インストール \$ sudo apt-get install rtl-sdr	
<b>(2) 音声出力ソフト</b>			
aplay	<ul style="list-style-type: none"> <li>・rtl_fmの出力を音声として出力するために使用。</li> <li>・SDRモジュールの情報を確認する用途にも使用。</li> </ul>	OSに同梱されているため、インストールは不要。	
<b>(3) 音声処理ソフト</b>			
SOX	<ul style="list-style-type: none"> <li>・サイト <a href="http://sox.sourceforge.net/">http://sox.sourceforge.net/</a></li> <li>・音声の録音、切り貼り、フォーマット変換、スペクトログラムの作成など多くの音声処理の機能を持っている。</li> <li>・各所で使用。</li> </ul>	インストール (本体) \$ sudo apt-get install sox インストール (提案パッケージ) \$ sudo apt-get install libsox-fmt-all	
<b>(4) グラフ作成ソフト</b>			
gnuplot	<ul style="list-style-type: none"> <li>・サイト <a href="http://www.gnuplot.info/">http://www.gnuplot.info/</a></li> <li>・いろいろな種類のグラフを作成する機能を持っている。</li> <li>・強度データをグラフ化するために使用。</li> </ul>	インストール (本体) \$ sudo apt-get install gnuplot	
<b>(5) 画像処理ライブラリ</b>			
OpenCV	<ul style="list-style-type: none"> <li>・サイト <a href="https://opencv.org/">https://opencv.org/</a></li> <li>・自由に画像を編集することができるライブラリ。</li> <li>・スペクトログラムへ強度データ(グラフ)を描くために使用。</li> </ul>	インストール \$ sudo apt-get install libopencv-dev	
<b>(6) PNG画像圧縮ソフト</b>			
pngquant	<ul style="list-style-type: none"> <li>・サイト <a href="https://pngquant.org/">https://pngquant.org/</a></li> <li>・PNG 画像専用の圧縮ソフト。</li> <li>・スペクトログラムを圧縮するために使用。</li> </ul>	インストール \$ sudo apt-get install pngquant	
<b>(7) 画像結合</b>			
自作 C++プログラム	<ul style="list-style-type: none"> <li>・スペクトログラムへ強度データ(グラフ)を描くためのソフト。</li> <li>・OpenCVを使用して、C++言語で自作。</li> </ul>	インストール 6項に記載。	※ ソースを同梱しています。 改造は自由です。
<b>(8) エコー計数</b>			
自作 C プログラム	<ul style="list-style-type: none"> <li>・録音データを変換した強度データを使用して、二乗平均平方根 (Root Mean Square) し、このデータが閾値より強いかどうかでエコーを判定するためのソフトであり、C言語で自作。</li> </ul>	インストール 6項に記載。	※ ソースを同梱しています。 改造は自由です。

(9) データのアップロード		
sftp, ftp	<ul style="list-style-type: none"> <li>・ファイルサーバーやWEBサーバーへ取得したデータを送信して保存、公開するするために使用。</li> </ul>	インストール (ftp) \$ sudo apt-get install ftp
expect	<ul style="list-style-type: none"> <li>・コマンドの対話操作を自動化するためのソフトで、sftp , ftp の処理で使用。</li> </ul> <p>※ 配布の観測ソフトでは、sftpは使用していない。</p>	インストール (expect 本体) \$ sudo apt-get install expect インストール (expect 提案パッケージ) \$ sudo apt-get install tk8.6 tcl-tclreadline
<div style="border: 1px solid black; padding: 5px; width: fit-content; margin-left: auto; margin-right: auto;">             sftp は OS に同梱されているため、インストールは不要。           </div>		
(10) 観測ソフト		
Linux Shell (bash)	<ul style="list-style-type: none"> <li>・上記の各種ソフトウェアの操作をプログラミングすることで、観測を自動化するためのシェルプログラム。</li> </ul>	インストール 6 項に記載。
(11) 音量調整ソフト		
amixer	<ul style="list-style-type: none"> <li>・コマンドライン操作により、音量を調整するためのソフトで、初期設定時に単独で使用。</li> </ul>	OSに同梱されているため、インストールは不要。

## 6. 観測ソフトのパッケージ

## (1) インストール

(a) ソフトウェアをPCへ保存。  
ファイル名 : MroSpec\_v120.zip

(b) 保存したファイルを解凍。  
「解凍後の構成」を参照。シャットダウン用シェルを同梱しています。

(c) 設定ファイルを編集して観測に必要なパラメーターを設定。

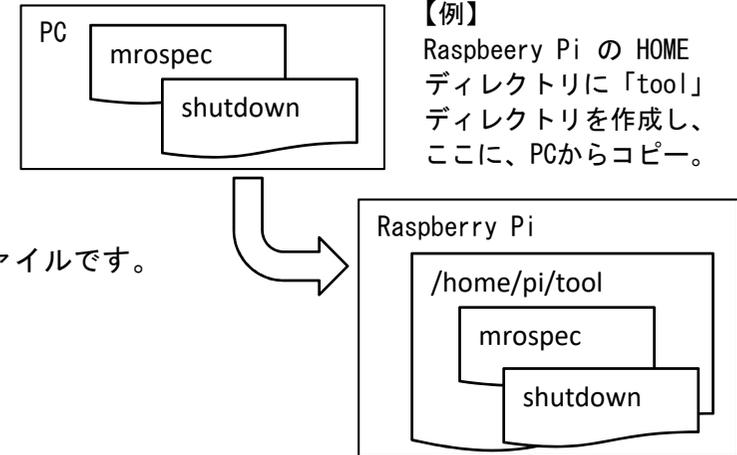
(d) 「mrospec」ディレクトリ以下をRaspberry Piへコピー。（参考：右図）  
シャットダウン用シェルを使用する場合は「shutdown」ディレクトリもコピー。

(e) シェルとC, C++ プログラムに実行権を付与。  
・対象のファイルは、「解凍後の構成 一覧」のファイルの右に☆印を付記しているファイルです。  
・実行権(755)の付与は、GUIタイプのFTPソフトやコマンド操作で設定します。  
コマンドで設定する場合は、「chmod」を使用します。

```
$ chmod 755 ファイル名
```

注記：

本書では、Raspberry Pi 3 での作業環境を記載していますが、Raspberry Pi 4 , Jetson Nano (Ubuntu) を使用した場合でも、同等の作業で構築できます。



## ※ C, C++ プログラムについて

同梱している実行ファイル (mroecho, mromixs, calceco) は、以下の環境でビルドしたもので、他の環境で動作しない場合があります。その場合は、以下のようにソースファイルをビルド (gcc, g++) して実行ファイルを作成してください。

環境 : Raspberry Pi OS with desktop (Linux raspberrypi3 5.10.63-v7+)

・ mroecho.c

```
$ gcc -Wall -o mroecho mroecho.c -lm
```

・ calceco.c

```
$ gcc -Wall -o calceco calceco.c -lm
```

・ mromixs.cpp

OpenCVのバージョンによって指定するライブラリが異なります。

Ver 3.x.x の場合

```
$ g++ -Wall -o mromixs mromixs.cpp `pkg-config --cflags --libs opencv`
```

Ver 4.x.x の場合

```
$ g++ -Wall -o mromixs mromixs.cpp `pkg-config --cflags --libs opencv4`
```

【OpenCVのバージョン確認方法】 **朱書き部分で確認**

```
$ dpkg -l | grep libopencv
ii libopencv-calib3d-dev:armhf      3.2.0+dfsg-6
armhf      development files for libopencv-calib3d3.2
ii libopencv-calib3d3.2:armhf     3.2.0+dfsg-6
armhf      computer vision Camera Calibration library
:
```

## (2) データの管理

処理の途中で一時的に作成されるデータ以外は、各ディレクトリ（SDメモリ）に保存されます。

## (a) データの削除

保存されたデータは設定ファイルに記載した日数に従い、自動削除することができます。 ※ 削除しない設定もできます。  
削除するデータの単位は、時間ではなく日単位での削除となります。

たとえば、3日間のデータを残しておきたい場合は、設定ファイルの指定の項目に、「3」を記載します。

## 【例】

現在	2021年10月28日	※ 指定日数前のデータを削除する仕様のため、
残す日数	3日	この場合、10月24日などのデータは残ります。
削除対象のデータ	2021年10月25日のファイル	残ったデータを一括削除する方法：11項参照。
	<ogg> 202110250000.ogg ~ 202110252350.ogg	
	<sig> 202110250000.csv ~ 202110252350.csv	
	<png> 202110250000.png ~ 202110252350.png	
	202110250000sig.png ~ 202110252350sig.png	
	<eco> 20211025echo.csv	

削除のタイミングは1日2回：  
・ 00:10~00:20の観測が終了したタイミング（00:20過ぎ）  
・ 12:10~12:20の観測が終了したタイミング（12:20過ぎ）

## (b) サーバーへアップロードしているデータ

デフォルトは、以下のデータをアップロードしています。  
アップロードが不要なデータは、「mrospec.sh」のソースを修正してください。

- ・ スペクトログラム (png)
- ・ ~~強度グラフ (png)~~
- ・ エコー情報 (csv)

※ 行頭に「#」を付加することでその行の命令を無効にできます。

## 【 mrospec.sh (mrospec\_send 関数の中) 】

```
send ¥"put ${DIR_PNG}${SPC_PNG} ${SPC_PNG}¥r¥"   . . . スペクトログラム
expect ¥"ftp>¥"

# send ¥"put ${DIR_PNG}${SIG_PNG} ${SIG_PNG}¥r¥"   . . . 強度グラフ
# expect ¥"ftp>¥"

# send ¥"put ${DIR_ECO}${ECO_CSV} ${ECO_CSV}¥r¥"   . . . エコー情報
# expect ¥"ftp>¥"                                  行頭の# 無効化
```

## (3) 解凍後の構成

- ・ 「mrospec」ディレクトリが作成され、この中に各種ファイルとサブディレクトリを展開。
- ・ 観測データは、サブディレクトリに保存されます。
- ・ 朱書きのファイルは、展開時には存在しません。
- ・ 「cal」ディレクトリには調整用のソフトが入っています。

※ 朱書きで示した各ファイルは、観測時に保存される観測データ例として示したものです。  
※ 観測データは、設定ファイルで指定した日数以前のデータを自動で削除することができます。  
※ 同梱しているシャットダウン用のシェルが「shutdown」ディレクトリに展開されます。

## 解凍後の構成 一覧

## &lt;解凍したディレクトリ&gt;

<mrospec>  
<shutdown>

zipを解凍すると、これら2つのディレクトリと、この中に以下が展開されます。

<mrospec>			
mrospec.sh	☆	観測シェル	このシェルを実行することで観測を開始。
mroecho	☆	エコー計数プログラムのソース(.c)と実行ファイル	観測シェルから起動され、エコーを計数して、エコー検出時刻とエコー継続時間を保存。
mromixs	☆	画像結合プログラムのソース(.cpp)と実行ファイル	観測シェルから起動され、スペクトログラムへ強度グラフを描き込みます。
mroinit.sh	☆	観測シェルの起動シェル	crontabから起動され、観測シェルを起動するためのシェル。
mrokill.sh	☆	プロセス終了シェル	観測シェルで使用する各種ソフト(プロセス)を終了するためのシェル。
mrodele.sh	☆	データ削除シェル	観測データを一括で全て削除するためのシェル。
mroftps.txt		FTP用設定ファイル	FTPサーバーへアクセスするための情報を記載。
mroconf_0978.txt		設定ファイル	観測者や観測環境の情報、受信設定、データ処理設定などを記載。
202109280530.dat		生の強度データ	データ処理中の一時ファイルであり、処理後は自動削除。
<ogg>		録音データ	10分毎のファイルで保存。
<png>		スペクトログラム	観測データ(png)
		強度グラフ	画像:1秒/画素 強度グラフは、現在は対数にはなっていません。
<sig>		強度データ	10分毎のファイルで保存。このデータを使用して強度グラフを作成。
<eco>		エコー情報	1日毎のファイルで保存。エコー検出時刻とエコー継続時間を保存。
<cal>			
calcmro.sh	☆	調整用シェル	1分間のデータを受信して受信ゲインや録音ゲインなど設定を確認。
calceco	☆	調整用プログラムのソース(.c)と実行ファイル	調整用シェルから起動され、受信した1分間のデータ解析して受信状態を出力。
calceco.c			
calcdev.txt		調整用設定ファイル	Audio入力デバイス名を記載。
rescalc.ogg			1分間の録音データ
rescalc.dat			録音データから取得した生の強度データ
rescalc.txt		調整結果	生の強度データから算出したRMSデータ
rescalc.png			スペクトログラム
rescalc_log.txt			調整に使用したパラメータと信号強度の平均とピークを保存。
<shutdown>			
shutdown.sh		シャットダウン用シェル	13項を参照。

## 7. 機器の確認と初期設定

使用する機器の動作を確認すると共に、各種情報を取得し初期値を設定します。

### (1) USB機器の確認

#### (a) 「SDRモジュール」と「USBオーディオ変換アダプタ」の確認

「lsusb」コマンドを入力し、機器の情報が表示された場合は使用できます。

※ 表示される情報は、製品により異なります。

Device 005 . . . USBオーディオ変換モジュール (C-Media や Audio で判別できます)  
Device 004 . . . SDRモジュール (RTL000 で判別できます)

```
$ lsusb
Bus 001 Device 005: ID 0d8c:0014 C-Media Electronics, Inc. Audio Adapter (Unitek Y-247A)
Bus 001 Device 004: ID 0bda:2838 Realtek Semiconductor Corp. RTL2838 DVB-T
Bus 001 Device 003: ID 0424:ec00 Standard Microsystems Corp. SMSC9512/9514 Fast Ethernet Adapter
Bus 001 Device 002: ID 0424:9514 Standard Microsystems Corp. SMC9514 Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
$
```

※ 左の結果は、OSがデバイスとして認識していることを示しているものであって、各種ソフトウェアで使用できることを保証しているものではありません。

#### (b) 「SDRモジュール」の使用可能ゲインを確認

「rtl\_test」コマンドを入力し、表示された「Supported gain values」が使用可能なゲインを示しています。「rtl\_fm」コマンドのオプションで使用する値であり、設定ファイルに記載して可変設定を可能にしています。

※ 使用可能なゲインの値は、製品により異なります。

```
$ rtl_test
Found 1 device(s):
 0: Realtek, RTL2838UHIDIR, SN: 00000001

Using device 0: Generic RTL2832U OEM
Detached kernel driver
Found Rafael Micro R820T tuner
Supported gain values (29): 0.0 0.9 1.4 2.7 3.7 7.7 8.7 12.5 14.4 15.7 16.6 19.7 20.7 22.9 25.4 28.0 29.7 32.8 33.8 36.4 37.2 38.6
40.2 42.1 43.4 43.9 44.5 48.0 49.6
[R82XX] PLL not locked!
Sampling at 2048000 S/s.
```

```
Info: This tool will continuously read from the device, and report if
samples get lost. If you observe no further output, everything is fine.
```

```
Reading samples in async mode...
```

```
Allocating 15 zero-copy buffers
```

```
lost at least 116 bytes
```

```
^C
```

<--- Ctrl-C キーで終了

```
Signal caught, exiting!
```

```
User cancel, exiting...
```

```
Samples per million lost (minimum): 3
```

```
Reattached kernel driver
```

```
$
```

(c) 「USBオーディオ変換アダプタ」のデバイス名を確認

「 pactl 」 コマンドを入力し、デバイス名を確認します。

「 SOX 」 コマンドのオプションで使用する名称であり、設定ファイルに記載して使用します。

※ デバイス名は製品により異なります。

```
$ pactl list short sources
```

```
0 alsa_output.platform-bcm2835_audio.analog-stereo.monitor module-alsa-card.c s16le 2ch 44100Hz IDLE
```

```
1 alsa_output.usb-C-Media_Electronics_Inc._USB_Audio_Device-00.analog-stereo.monitor module-alsa-card.c s16le 2ch 44100Hz IDLE
```

```
2 alsa_input.usb-C-Media_Electronics_Inc._USB_Audio_Device-00.analog-mono module-alsa-card.c s16le 1ch 44100Hz IDLE
```

```
$
```

## (2) 音量の初期設定

「USBオーディオ変換アダプタ」の入出力の音量のベースを設定します。

ここでは、少し大きめの音量になるよう設定した方が良いでしょう。

音量は、受信する周波数や周辺ノイズにより変動するため、個別の音量は、SOXの録音ゲインを設定ファイルに記載することで調整します。

## (a) 現在の音量を確認

```
$ amixer -i
Simple mixer control 'Master',0
  Capabilities: pvolume pswitch pswitch-joined
  Playback channels: Front Left - Front Right
  Limits: Playback 0 - 65536
  Mono:
  Front Left: Playback 31609 [48%] [on]
  Front Right: Playback 31609 [48%] [on]
Simple mixer control 'Capture',0
  Capabilities: cvolume cvolume-joined cswitch cswitch-joined
  Capture channels: Mono
  Limits: Capture 0 - 65536
  Mono: Capture 35542 [54%] [on]
$
```

## (b) 設定する IDを確認

```
$ amixer controls
numid=4, iface=MIXER, name='Master Playback Switch'
numid=3, iface=MIXER, name='Master Playback Volume'      出力
numid=2, iface=MIXER, name='Capture Switch'
numid=1, iface=MIXER, name='Capture Volume'              入力
$
```

## (c) 音量を設定

```
$ amixer cset numid=3 80%      出力
numid=3, iface=MIXER, name='Master Playback Volume'
; type=INTEGER, access=rw-----, values=2, min=0, max=65536, step=1
; values=52429, 52429
$
$ amixer cset numid=1 80%      入力
numid=1, iface=MIXER, name='Capture Volume'
; type=INTEGER, access=rw-----, values=1, min=0, max=65536, step=1
; values=52429
$
```

## (d) 設定変更後の音量を確認

(a) 項のコマンドを実行し、% の値が設定値になっていることを確認。

## (3) 音声出力の確認

地元のFM放送を受信してUSBオーディオ変換アダプタのヘッドフォン端子で放送を聴くことで、音声出力を確認します。 ※ ヘッドフォンなどを接続してください。

以下は、80.2MHzのFM放送を受信して、音声を出力する例です。

```
$ rtl_fm -f 80.2M -M wbfm -s 200k -r 48k | aplay -r 48000 -f S16_LE -D plughw:1,0
Ctrl-C キーで終了
```

※ 「-D plughw:1,0」は通常は不要ですが、音が出なかった場合は、(b)を参考に設定して試してください。

(a) (2)項で実施した音量を最大（100%）に設定する。

(b) 音声出力先を指定する場合

「aplay -i」を実行し、USB Audio の項目を確認します。

-D plughw:1,0 . . . 出力先を指定するためのオプション設定で、「カード 1」と「デバイス 0」を示しています。

```
$ aplay -l
**** ハードウェアデバイス PLAYBACK のリスト ****
カード 0: Headphones [bcm2835 Headphones], デバイス 0: bcm2835 Headphones [bcm2835 Headphones]
  サブデバイス: 8/8
  サブデバイス #0: subdevice #0
  サブデバイス #1: subdevice #1
  サブデバイス #2: subdevice #2
  サブデバイス #3: subdevice #3
  サブデバイス #4: subdevice #4
  サブデバイス #5: subdevice #5
  サブデバイス #6: subdevice #6
  サブデバイス #7: subdevice #7
カード 1: Device [USB Audio 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
$
```

## (4) 音声入力の確認

「USBオーディオ変換アダプタ」のヘッドフォン端子とマイク端子をオーディオステレオケーブルで接続し、「rtl\_fm」で受信した音声を「SOX」に入力して確認します。

- ・ FM放送を受信して確認。
- ・ 観測に使用する周波数を受信して確認

「rtl\_fm」で指定の周波数を受信し音声を出力した状態で、「SOX」を起動し、音声の入力状態を確認します。

```
$ rtl_fm -f 80.2M -M wbfm -s 200k -r 48k | aplay -r 48000 -f S16_LE
```

Ctrl-C キーで終了

```
$ sox -t pulseaudio alsa_input.usb-C-Media_Electronics_Inc._USB_Audio_Device-00.analog-mono -n
```

Ctrl-C キーで終了

バックグラウンドで実行する場合は、  
行末に「&」を付ける。  
終了は、EnterやCtrl-Cでプロンプトを出した  
後、「killall」コマンドを実行する。

```
$ killall rtl_fm
```

2つのコマンドを同時に使用するため、  
以下のどちらかの方法で実施してください。

- 「rtl\_fm」をバックグラウンドで  
実行する。
- それぞれのコマンドを別のターミナル  
(SSH接続を2つ使用)で実行する。

右の画面は(b)の方法で確認している時の  
ものです。

- ・ Windows PCから TeratermでSSH接続。
- ・ 左ウインド: rtl\_fm を実行
- ・ 右ウインド: amixer で音量調整し、  
SOX を実行

```

192.168.0.24:22 - pi@raspberrypi3: ~ - VT
ファイル(F) 編集(E) 設定(S) コントロール(O) ウィンドウ(W) ヘルプ(H)
pi@raspberrypi3:~$ rtl_fm -f 97.7993M -M usb -s 6000 -g 49.6 | aplay -r 6000 -f S16_LE
Found 1 device(s):
 0: Realtek, RTL2838UHIDIR, SN: 00000001

Using device 0: Generic RTL2832U OEM
Detached kernel driver
Found Rafael Micro R820T tuner
Tuner gain set to 49.60 dB.
Tuned to 98049800 Hz.
oversampling input by: 167x.
Oversampling output by: 1x.
Buffer size: 8.18ms
Exact sample rate is: 1002000.012542 Hz
Create UDP thread
Created UDP thread
Main socket started! (-): Tuning enabled on UDP/6020
Allocating 15 zero-copy buffers
Sampling at 1002000 S/s.
Output at 6000 Hz.
再生中 raw データ 'stdin': Signed 16 bit Little Endian
アンダーラン!!! (少なくとも 1044.127 ms)
[

192.168.0.24:22 - pi@raspberrypi3: ~ - VT
ファイル(F) 編集(E) 設定(S) コントロール(O) ウィンドウ(W) ヘルプ(H)
In:0.00% 00:02:25.41 [00:00:00.00] Out:6.98M [ ] Clip:0 ^C
Aborted.
pi@raspberrypi3:~$ ^C
pi@raspberrypi3:~$ ^C
pi@raspberrypi3:~$ amixer controls
numid=4,iface=MIXER,name='Master Playback Switch'
numid=3,iface=MIXER,name='Master Playback Volume'
numid=2,iface=MIXER,name='Capture Switch'
numid=1,iface=MIXER,name='Capture Volume'
pi@raspberrypi3:~$ amixer cset numid=1 100%
numid=1,iface=MIXER,name='Capture Volume'
; type=INTEGER,access=rw-----,values=1,min=0,max=65536,step=1
; values=65536
pi@raspberrypi3:~$ sox -t pulseaudio alsa_input.usb-C-Media_Electronics_Inc._USB_Audio_Device-00.analog-mono -n
Input File : 'alsa_input.usb-C-Media_Electronics_Inc._USB_Audio_Device-00.analog-mono' (pulseaudio)
Channels : 2
Sample Rate : 48000
Precision : 16-bit
Sample Encoding: 16-bit Signed Integer PCM
In:0.00% 00:03:02.02 [00:00:00.00] Out:8.73M [ ===|=== ] Clip:0 [

```





9. 設定ファイルの準備

(1) 観測用設定ファイル

観測に必要な情報を設定ファイルに記載します。  
 観測環境ごとに設定ファイルの名称を変更して作成しておくとも便利です。

注意点

- ・ 1文字目が「 # 」の行はコメント行となり、処理には影響しません。 設定値の行間に記載することも可能です。
- ・ 各行は、1文字目から記載してください。
- ・ 行の順番を入れ替えることはできません。
- ・ 全角文字の使用はできません。

<pre># HEADER : for Spectrogram (Max 60 char) # COMMENT : for Spectrogram (Max 90 char) # rtl_fm : Frequency # rtl_fm : Option # aplay : Rate (Same as rtl_fm "-s") # SOX : Record gain # SOX : Spectrogram rate # SOX : Spectrogram option # SOX : Audio input device # gnuplot : Y range Min:Max # Echo : Threshold offset # File : File storage days (0: Not delete) # JP3TQZ @ Takarazuka, Hyogo, JAPAN RX: RTL-SDR, rtl_fm (97.8MHz-600Hz, USB) / ANT: VHF TV (2EL, Roof top) / Raspberry Pi 3 97.7994M -M usb -s 6000 -g 49.6 6000 15 1.8k -X 1 -y 304 -z 50 -p 1 -a -w Dolph alsa_input.usb-C-Media_Electronics_Inc._USB_Audio_Device-00.analog-mono 0.03:0.1 0.007 3</pre>	<p>} コメント</p>	<p>..... スペクトログラムの上部に記載する内容(処理への影響なし)</p> <p>..... スペクトログラムの下部に表示する内容(処理への影響なし)</p> <p>..... rtl_fm の受信周波数 (M : MHz を示す)</p> <p>..... rtl_fm のオプション (Mode, Sample rate, Gain)</p> <p>..... aplay のオプション (Sample rate : rtl_fmと同じ値)</p> <p>..... SOX の録音ゲイン (これで音量の微調整を行う)</p> <p>..... SOX spectrogram のオプション (Sample 周波数)</p> <p>..... SOX spectrogram のオプション (画面や処理ウィンドウ)</p> <p>..... SOX の録音時に設定する Audio入力デバイス名</p> <p>..... gnuplot のグラフ縦軸の範囲 [下限 : 上限]</p> <p>..... Echo 判定の閾値 (エコー &gt; 10分間の平均値+閾値)</p> <p>..... 観測データを残す日数 (0: 消去しない)</p>
--	---------------	--

## (2) FTP用設定ファイル (mroftps.txt)

FTPサーバーの情報を設定します。  
ログイン情報を平文で記載しますので、取扱には注意してください。

## 注意点

- ・ ファイル名は、「 mroftps.txt 」から変更できません。
- ・ コメント行は記載できますが、セキュリティリスクを低減するために、書かない方がいいでしょう。
- ・ 各行は、1文字目から記載してください。
- ・ 行の順番を入れ替えることはできません。
- ・ 全角文字の使用はできません。
- ・ 送信先のディレクトリは、FTPサーバーへログインした際のディレクトリからのパスを記載します。

## 【 mroftps.txt の例 (2例) 】

```
abcdefg  
xyz123  
xxxx.yyyy.jp  
public_html/data
```

```
qwert@vvvv.ssss.jp  . . . User ID  
xyz123              . . . Password  
111.222.333.444    . . . Host (ドメイン名 または IP アドレス)  
data                . . . 送信先のディレクトリ
```

## 10. 観測

## (1) 手動で起動する方法

※ バックグラウンド起動すると、Logout後も観測が続きます。  
(&を付けて起動)

起動オプションと観測用設定ファイルを引数としてシェルを実行します。

オプション M : モニターモード ・ Audio入力の音量を確認することができます。

実行例 \$ ./mrospec.sh M mroconf\_0978.txt

オプション L : ローカルモード ・ 観測データをサーバーに送信しないモードです。  
・ 観測データは、SDメモリに保存されています。  
ただし保存日数の設定に依存します。

実行例 \$ ./mrospec.sh L mroconf\_0978.txt

オプション 0 : オープンモード ・ 観測データをサーバーに送信するモードです。  
・ 観測データは、SDメモリにも保存されています。  
ただし保存日数の設定に依存します。

実行例 \$ ./mrospec.sh 0 mroconf\_0978.txt

## (2) 電源 ON で起動する方法

(a) 観測ソフト「mrospec.sh」を起動するためのシェル「mroinit.sh」に  
設定ファイルを記載。

「mrospec.sh」はバックグラウンドで起動されます。

(b) 「crontab」に以下の2つのソフトの起動を設定。

- ・ 「pulseaudio」・・・録音時に「SOX」が使用。
- ・ 「mroinit.sh」

※ 「crontab」の使用方法(編集)は、14項に記します。

## (3) 観測ソフトの終了(実行停止)

(a) フォアグラウンドで実行している場合、「Ctrl-C」キーで終了します。

(b) (2)項で起動した場合、または、バックグラウンドで実行している場合は、  
「mrokill.sh」を実行して観測ソフトを終了してください。

## 【実行方法】

\$ ./mroskill.sh

## 【mrokill.shの内容】

```
killall mrospec.sh      ・ ・ ・ ・ ・ 観測ソフトを終了
killall sox             ・ ・ ・ ・ ・ SOXを終了
killall aplay           ・ ・ ・ ・ ・ aplayを終了
                        rtl_fm も終了します
```

## 【mroinit.sh】

```
CONF_MRO="mroconf_0978.txt"      ・ ・ ・ 設定ファイルを記載

echo "Wait for 120 sec."
sleep 120                          ・ ・ ・ Systemの起動待ち

BASE_DIR=`dirname $0`             ・ ・ ・ ディレクトリを
cd ${BASE_DIR}                    シェルの場所へ移動

./mrospec.sh 0 ${CONF_MRO} &     ・ ・ ・ mrospec.sh を起動

exit
```

※ プロセスが起動していない場合は、  
以下のようになりますが問題ありません。



```
pi@raspberrypi3:~/tool/mrospec $ ./mrokill.sh
mrospec.sh: no process found
sox: no process found
aplay: no process found
```

## (4) 動作状態(例)

朱書きの部分が関係するプロセスです。

## 【 観測開始待ち 】

```
pi@raspberrypi3:~ $ ps xww
PID TTY      STAT   TIME COMMAND
 525 ?        SI     0:05 /usr/bin/pulseaudio --start
1029 ?        S      0:00 /bin/bash ./mrospec.sh 0 mroconf_0978.txt
1047 ?        SI     1:27 rtl_fm -f 97.7994M -M usb -s 6000 -g 49.6
1048 ?        SI     0:00 /usr/bin/aplay -r 6000 -f S16_LE
1052 ?        S      0:00 sleep 431
1431 ?        Ss     0:00 /lib/systemd/systemd --user
1432 ?        S      0:00 (sd-pam)
1446 ?        R      0:00 sshd: pi@pts/0
1447 pts/0    Ss     0:00 -bash
1470 pts/0    R+     0:00 ps xww
```

## 【 観測中 】

```
pi@raspberrypi3:~ $ ps xww
PID TTY      STAT   TIME COMMAND
 495 ?        SI     9:20 /usr/bin/pulseaudio --start
1031 ?        S      0:00 /bin/bash ./mrospec.sh 0 mroconf_0978.txt
1051 ?        SI    76:27 rtl_fm -f 97.7994M -M usb -s 6000 -g 49.6
1052 ?        SI     0:24 /usr/bin/aplay -r 6000 -f S16_LE
20176 ?       RI     0:12 /usr/bin/sox -q -t pulseaudio alsa_input.usb-C-Media_Electronics_Inc._USB_Audio_Device-00.analog-
mono ./ogg/202111071800.ogg channels 1 gain 18 trim 0 600
20210 ?        Ss     0:00 /lib/systemd/systemd --user
20211 ?        S      0:00 (sd-pam)
20225 ?        R      0:00 sshd: pi@pts/0
20228 pts/0    Ss     0:00 -bash
20387 ?        S      0:00 sleep 300
20416 pts/0    R+     0:00 ps xww
pi@raspberrypi3:~ $
```

※ 突然、正常に動作しなくなった場合  
シャットダウン

↓

電源ケーブルを抜く

↓

1分程度待つ

↓

電源投入(起動)

↓

観測

(ftpが正常に動作しなくなった経験あり)

## 1.1. 観測データの容量と手動削除

設定ファイル内で観測データの保存日数を指定して自動削除する機能を実装していますが、間欠的な観測では削除しきれないデータが残ってしまいます。

対策として、観測データを一括で削除するためのソフトを同梱しています。

### 注意点

観測中に実行すると、観測中のデータも削除され、観測処理で異常が発生します。

### (1) 実行

```
$ ./mrodele.sh
```

※ キャンセルする場合は、「n」を入力。  
(入力キーの1文字目で判断しています)

### (2) 削除対象のデータ

以下のディレクトリ内のデータを全て削除します。

```
./ogg/  
./png/  
./sig/  
./eco/
```

```
pi@raspberrypi3:~/tool/mrospec $ ./mrodele.sh  
Delete the observation data? (yes or no) : n  
canceled!!  
pi@raspberrypi3:~/tool/mrospec $ ./mrodele.sh  
Delete the observation data? (yes or no) : y  
pi@raspberrypi3:~/tool/mrospec $ ls -lt
```

### (3) 各データの容量と1日の最大消費量

データの種類	時間	管理単位	1日分の容量
		容量	
録音データ (ogg)	10分	5.2 Mbyte	748.8 Mbyte
スペクトログラム (png)	10分	40 K~130 Kbyte	Max 18 Mbyte
強度グラフ (png)	10分	2 Kbyte	288 Kbyte
強度データ (csv)	10分	9 Kbyte	1,296 Kbyte
エコー情報 (csv)	1日	1件 : 28 byte	Max 1.2 Mbyte ※
合計 (Max)			770 Mbyte

※ Max 1.2 Mbyte : 2秒に1回エコーを検出 : 43,200件

参考 : 観測前のディスク容量 (32GbyteのSDメモリを使用)

```
pi@raspberrypi3:~/tool/mrospec $ df -h  
ファイルシステム サイズ 使用 残り 使用% マウント位置  
/dev/root          29G  3.5G  24G  13% /  
devtmpfs           430M  0  430M  0% /dev  
tmpfs              462M  0  462M  0% /dev/shm  
tmpfs              462M  13M  450M  3% /run  
tmpfs              5.0M  4.0K  5.0M  1% /run/lock  
tmpfs              462M  0  462M  0% /sys/fs/cgroup  
/dev/mmcblk0p1    253M  49M  204M  20% /boot  
tmpfs              93M  0  93M  0% /run/user/109  
tmpfs              93M  0  93M  0% /run/user/1000  
pi@raspberrypi3:~/tool/mrospec $
```

12. エコー計数とエコー例

(1) 方法

エコー計数の方法として、以下の2種類の方法を実装し、選択して使用することができる。  
 どちらの方法であっても、エコーが連続する場合は1件にまとめ、この連続する時間はエコー継続時間となる。

RMS: 二乗平均平方根  
(Root Mean Square)

- (a) 10分間の信号強度(振幅)のRMSを算出して、この値が一定値以上の信号をエコーとして検出。
- (b) 10分間の信号強度(振幅)のRMSを算出し、更に標準偏差を算出して、標準偏差の外に存在する信号をエコーとして検出。(Ver 1.20の追加機能)

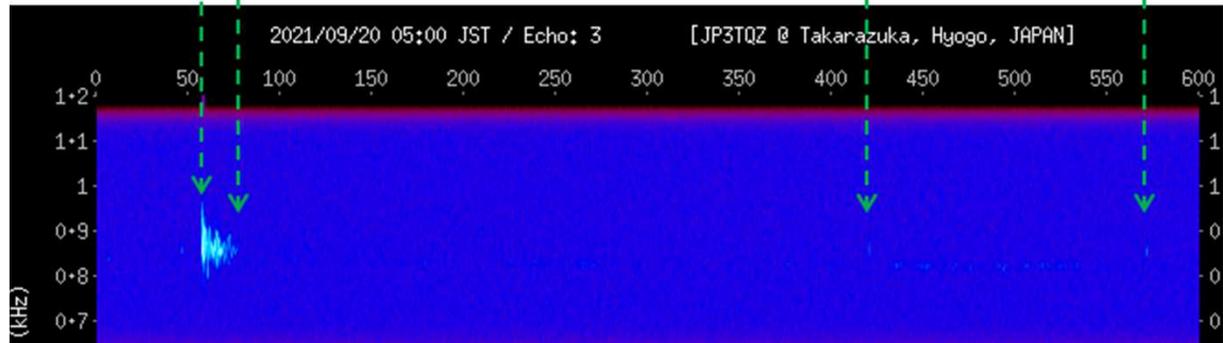
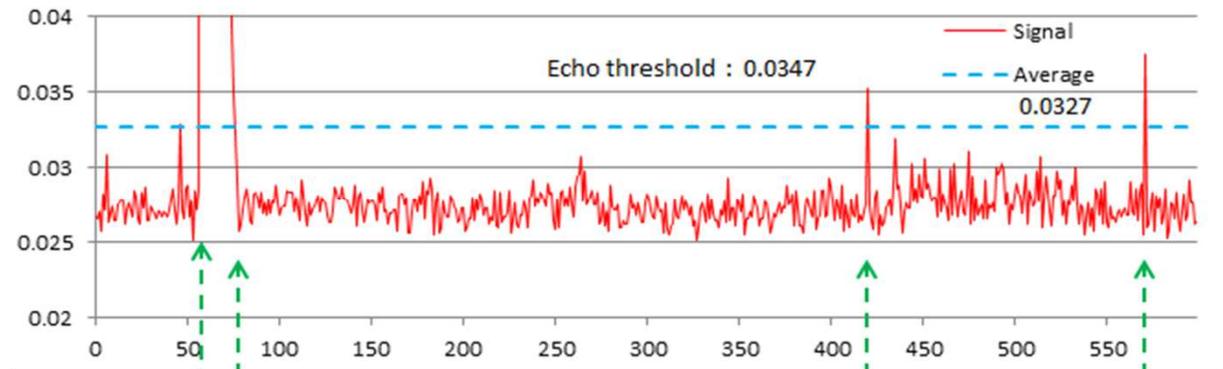
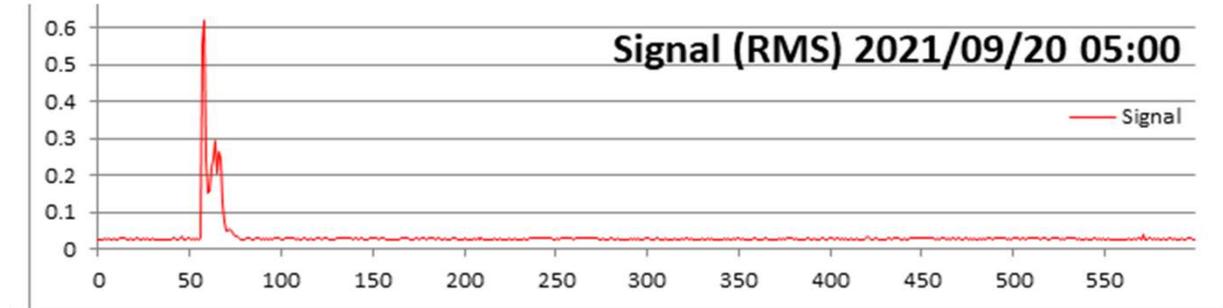
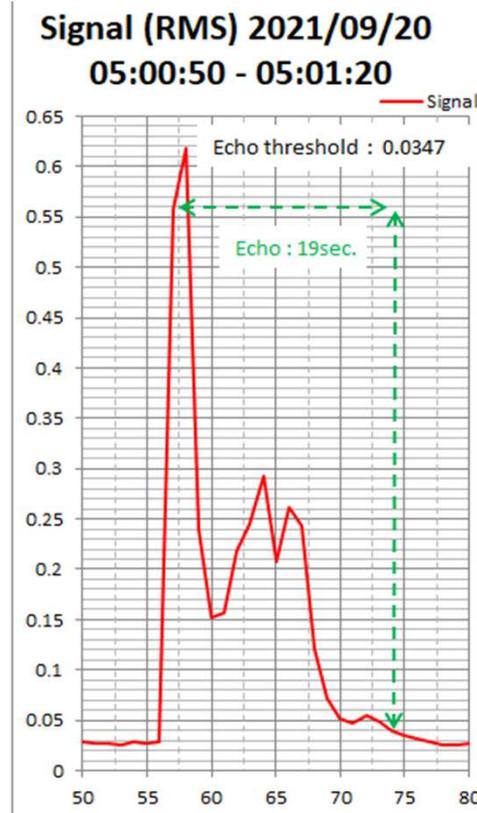
(2) 閾値

(a) 10分間の平均値+設定値で判定。

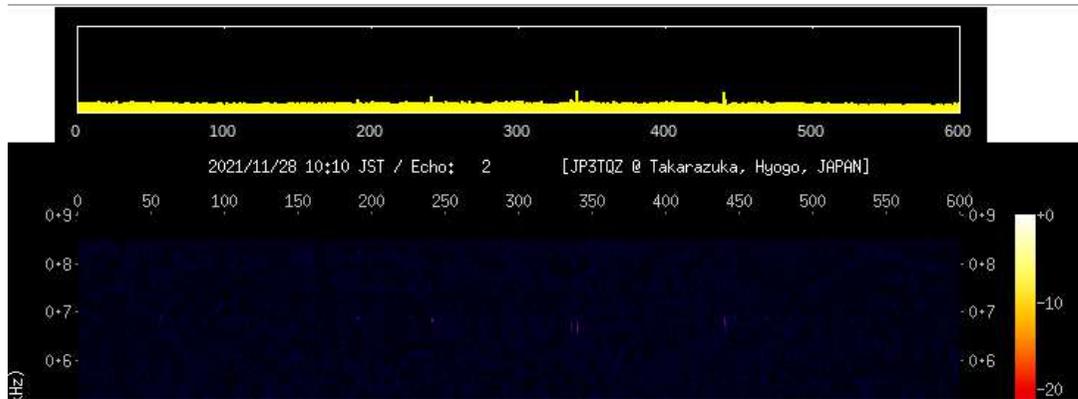
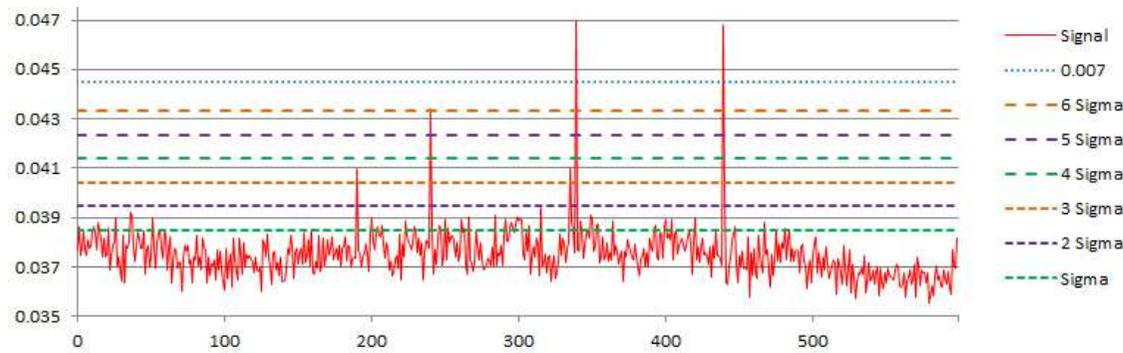
【例】

$$\begin{aligned} \text{閾値} &= \text{平均値}(0.0327) + \text{設定値}(0.002) \\ &= 0.0347 \end{aligned}$$

51	0.026733
52	0.027816
53	0.025185
54	0.028418
55	0.027104
56	0.02858
57	0.557854
58	0.61872
59	0.240024
60	0.152291
61	0.156572
62	0.218419
63	0.24468
64	0.292189
65	0.207426
66	0.262458
67	0.242779
68	0.121956
69	0.071212
70	0.051622
71	0.047549
72	0.054686
73	0.048066
74	0.039343
75	0.034909
76	0.03221
77	0.029115
78	0.025708
79	0.026264

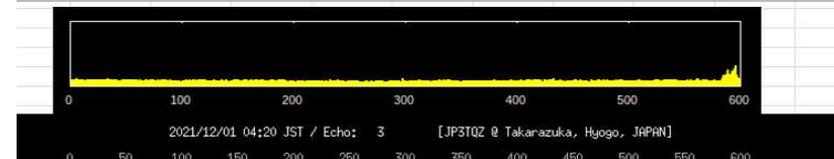
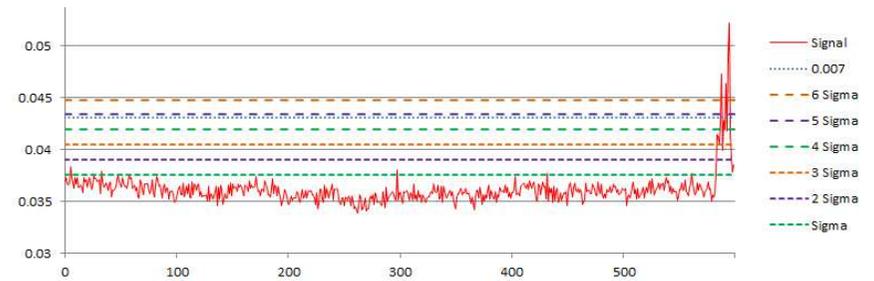


(b) 10分間の標準偏差で判定。



【標準偏差】

- ・ 初期値 : 5σ
- ・ シェル (mrospec. sh) を編集して変更可。
- ・ 0 (ゼロ) を設定すると、(a) の仕様で動作。

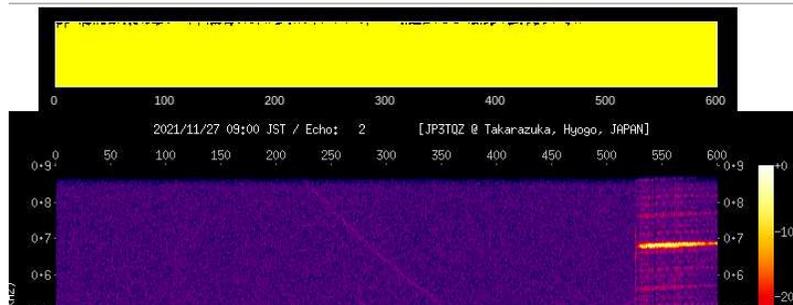
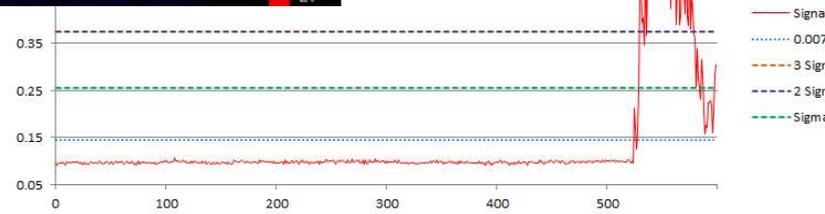


(c) 閾値の変更方法

観測シェル「mrospec.sh」のヘッダーの少し下に記載している変数「STD\_SGM」の設定値を編集して変更します。

- ・ STD\_SGM = 0 : 10分間の平均値 + 設定値
- ・ STD\_SGM = 0 以外 : 標準偏差 (初期値 5σ)

```
##
# Set SD(standard deviation) to judg echo.
#
STD_SGM=5
```



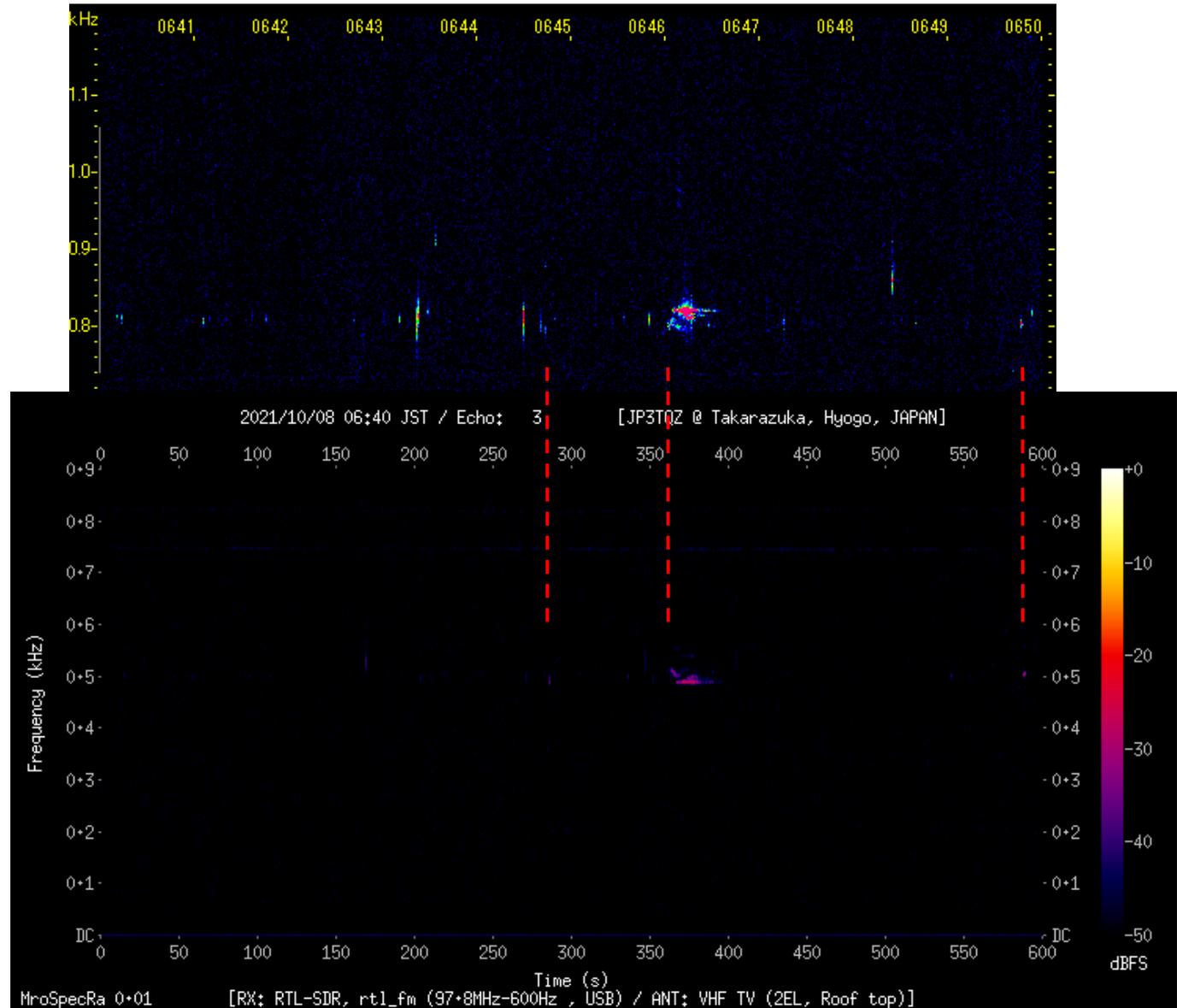
(3) エコー情報の保存

検出したエコーの情報は、検出時刻と継続時間を 1 日／ファイルとして保存。

(4) エコー例

(a) エコー (2021/10/08 06:46:02) 比較

上 : 97.0MHz (CW LSB) 受信  
 下 : 97.8MHz (USB) 受信



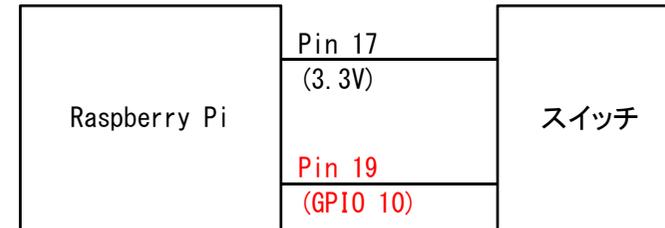
### 13. シャットダウン

ボタンの長押しでシャットダウンする方法です。  
添付のシェル「 shutdown.sh 」は、以下の仕様で作成しています。

- ※ 「 shutdown.sh 」は、配布パッケージに(zipファイル)に同梱しています。  
解凍したディレクトリに「 shutdown 」ディレクトリが作成され、この中に展開されています。  
観測シェルと同様に「 shutdown.sh 」に対して実行権を付与してください。

#### (1) 接続

使用するPinへの拘りはありません。  
使用するPin、特に、GPIOを変更した時は、シェルに記載の  
「 GPIO\_NUMS 」の値 (10) を変更してください。



#### (2) 動作

5秒以上のスイッチ長押しで、シャットダウンが始まります。  
シャットダウンの状況が見えるよう、基板に実装されている  
ACT LED (緑) を制御しています。

長押し中：点滅 (5 秒間)  
シャットダウンを検知：点灯 (2 秒間)  
シャットダウン開始

※ ACT LEDが消灯した後、安全のために 20~30秒待つて  
電源を落としましょう。

※ LEDの場所は、USB電源供給コネクタの近く。  
Red (赤) LEDは電源LEDで、コンセントが接続されていると、  
シャットダウンしていても点灯します。

※ ACT LEDは、通常、SDメモリのアクセス状態を示しています。

#### (3) シェルの起動

「 crontab 」を使ってシステム起動時にシャットダウン用シェルを起動します。  
「 crontab 」の使用方法 (編集) は、14項に記します。

## 【 shutdown.sh の例 】

```
#!/bin/bash

#####
#
# Shutdown shell
#
#####

##
# Initialize
#
TIME_PUSH=5      # Detect time (sec)
GPIO_NUMS=10    # GPIO No. (Pin 19)

gpio -g mode $GPIO_NUMS in      # Set input mode.
gpio -g mode $GPIO_NUMS down    # Set pull down resistance.

##
# Check push button
#
TIME_CONT=0
while [ ${TIME_CONT} -lt ${TIME_PUSH} ]
do
  if [ `gpio -g read ${GPIO_NUMS}` -eq "1" ]
  then
    if [ ${TIME_CONT} -eq 0 ]
    then
      # ACT LED blink
      echo heartbeat > /sys/class/leds/led0/trigger
    fi
    TIME_CONT=`expr ${TIME_CONT} + 1`
  else
    if [ ${TIME_CONT} -ne 0 ]
    then
      # ACT LED normal
      echo mmc0 > /sys/class/leds/led0/trigger
    fi
    TIME_CONT=0
  fi
  sleep 1
done

##
# Shutdown sign.
# ACT LED on (2 sec) -> ACT LED normal
#
echo none > /sys/class/leds/led0/trigger
echo 1 > /sys/class/leds/led0/brightness
sleep 2
echo 0 > /sys/class/leds/led0/brightness
echo mmc0 > /sys/class/leds/led0/trigger

##
# Shutdown execution.
#
shutdown -h now
```

※ シェルの改行コードは、「LF」です。  
 ※ シェルの権限を「755」に設定してください。

## 1 4. crontabの設定

「 crontab 」に、電源ON時に起動する処理を設定します。

※ ファイルパスは、環境に合わせて変更してください。

### (1) crontab の立ち上げ (右図 参照)

初めて立ち上げた時は、設定を編集するためのエディタを選択するメッセージが表示されます。(初回のみ表示)

### (2) Shutdownシェルの起動を設定

同梱のShutdownシェルを使用する場合は、以下を設定。

```
# For Shutdown.
@reboot sudo /home/pi/tool/shutdown/shutdown.sh &
```

### (3) 観測ソフト「 mrospec.sh 」の起動を設定

以下の2つのソフトの起動を設定します。

- ・ 録音時に「 SOX 」が使用する「 pulseaudio 」。
- ・ 観測ソフト「 mrospec.sh 」を起動するためのシェル「 mroinit.sh 」。

```
# For Pulseaudio.
@reboot /usr/bin/pulseaudio --start
# For MroSpec.
@reboot /home/pi/tool/mrospec/mroinit.sh
```

### (4) 編集後の保存

- ・ 書き込みを選択。
  - ・ ・ ・ Ctrl 大文字のO キーを押下。その後、Enterキーを押下。
- ・ 編集を終了。
  - ・ ・ ・ Ctrl 大文字のX キーを押下。
- ・ 設定内容の確認方法。
  - ・ ・ ・ 「 \$ crontab -l 」
- ・ 再起動する。
  - ・ ・ ・ 「 \$ sudo reboot 」

```
pi@raspberrypi3:~/tool/mrospec $ crontab -e
no crontab for pi - using an empty one
```

Select an editor. To change later, run 'select-editor'.

1. /bin/nano <---- easiest
2. /usr/bin/vim.tiny
3. /bin/ed

Choose 1-3 [1]: 1

```
crontab: installing new crontab
pi@raspberrypi3:~/tool/mrospec $
```

← 入力すると編集画面が立ち上がります。  
← 編集終了時に表示され、画面が戻ります。

```
192.168.0.24:22 - pi@raspberrypi3: ~/tool/mrospec VT
ファイル(F) 編集(E) 設定(S) コントロール(O) ウィンドウ(W) ヘルプ(H)
GNU nano 3.2 /tmp/crontab.m08odY/crontab 変更済み ^
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow  command
#
# For Shutdown.
@reboot sudo /home/pi/tool/shutdown/shutdown.sh &
# For Pulseaudio.
@reboot /usr/bin/pulseaudio --start
# For MroSpec.
@reboot /home/pi/tool/mrospec/mroinit.sh
書き込むファイル: /tmp/crontab.m08odY/crontab
^G ヘルプ M-D DOS フォーマット M-A 末尾に追加 M-B バックアップファ
^C 取消 M-M Mac フォーマット M-P 先頭に追加 ^T ファイラ
```

## 15. Ver 1.10 からの変更内容

## (1) 不具合修正

不具合内容	修正内容	対象ファイル
a. 信号強度グラフのX軸(幅)  不具合 : 1.002 sec/pixel 仕様 : 1 sec/pixel	gnuplotのX軸(幅)を修正。  誤 : set term png font arial 8 size 634,100 background "black" 正 : set term png font arial 8 size 635,100 background "black"	mrospec.sh
b. Sample rateの入力  不具合 : 入力データに対して、 不要な文字を付加。	入力データを変数へ設定する時に付加されている不要な文字を削除。  誤 : RTL_RAT="\$ {KEY_DAT} M" 正 : RTL_RAT="\$ {KEY_DAT} "	calcmro.sh

## (2) 仕様変更

変更内容	変更の差異	対象ファイル
a. エコー計数処理の閾値を変更。  詳細 : 12項に記載。	変更前 : 10分間の平均値+設定値 変更後 : 10分間の平均値+設定値 または 10分間の標準偏差 (mrospec.shを編集して、平均値と標準偏差の切り替えが可能)	mrospec.sh mroecho.c, mroecho
b. エコー情報のフォーマットを変更。	変更前 : 年/月/日/時:分:秒,経過秒,エコー数 変更後 : 年/月/日,時:分:秒,経過秒,エコー数 (日と時のセパレーターを / から , へ変更)	mroecho.c, mroecho
c. スペクトログラムを圧縮する機能を追加。  詳細 : (3)項に記載。	変更前 : --- 変更後 : スペクトログラムを圧縮。	mrospec.sh
d. スペクトログラムへ信号強度グラフを描画(結合)する機能を追加。  詳細 : (3)項に記載。	変更前 : スペクトログラムと信号強度グラフを別々のファイルで作成。 変更後 : スペクトログラムへ信号強度グラフを描画。 この変更により、信号強度グラフをサーバーへアップロードする機能は、コメントアウトで無効化。	mrospec.sh mromixs.cpp, mromixs
e. 信号強度グラフの表示範囲を自動切換えする機能を追加。  詳細 : (3)項に記載。	変更前 : 固定 変更後 : 信号強度(平均)がグラフ表示範囲(高さ)の半分以上になった場合、自動で表示範囲を1/4底上げする。	mrospec.sh

(3) 仕様変更の詳細

(a) スペクトログラムを圧縮

(b) 項の機能を追加した結果、スペクトログラムのファイル容量が元のファイル容量の2倍に増加する結果となったため、対策として圧縮機能を追加。

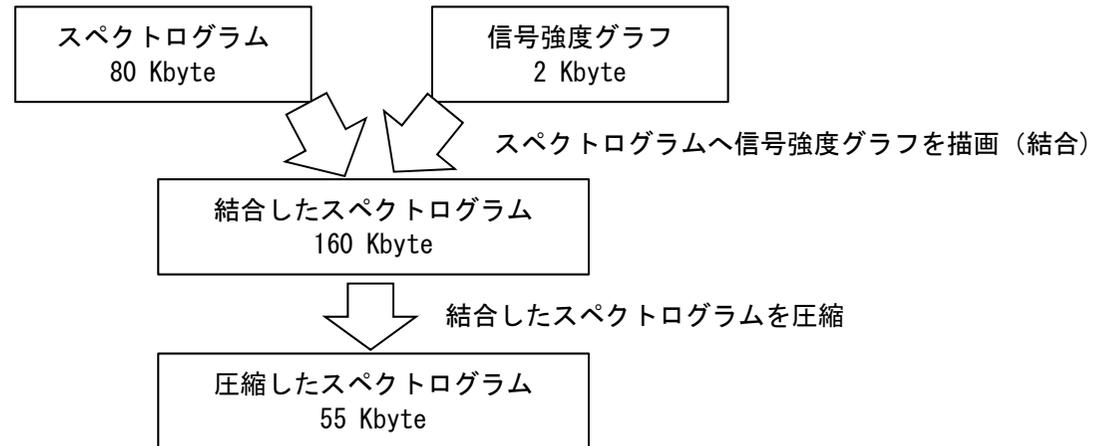
対策

PNG画像の圧縮ソフト (pngquant) を使用して圧縮する機能を追加。

結果

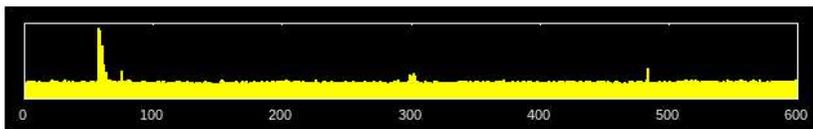
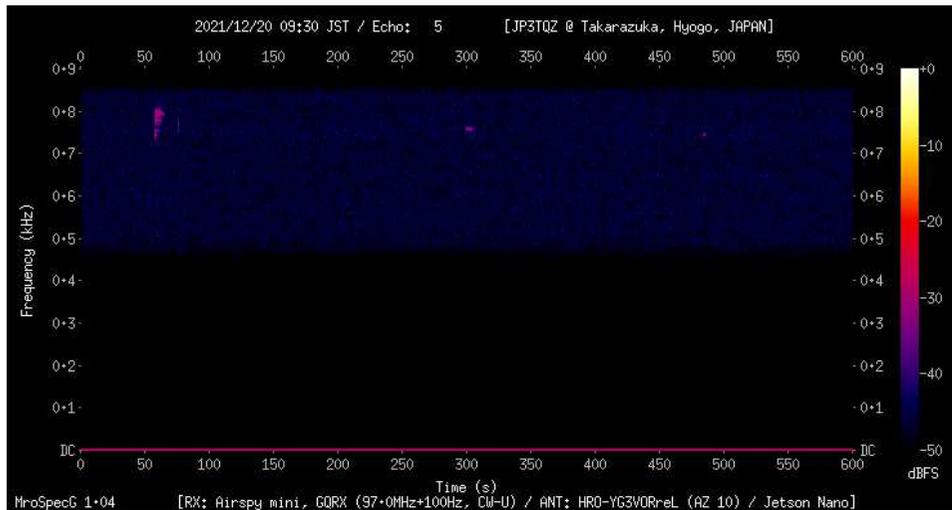
元のスペクトログラムよりもファイル容量を削減。

【例】

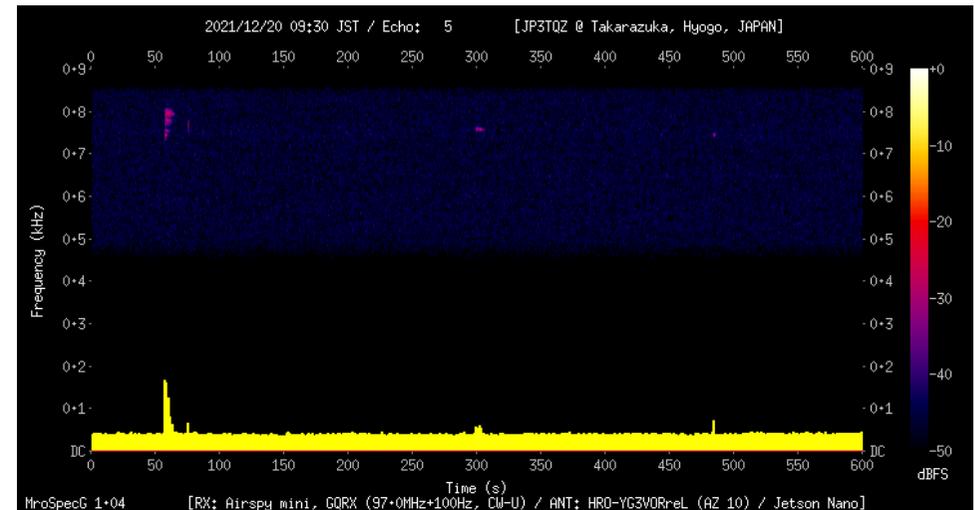


(b) スペクトログラムへ信号強度グラフを描画 (結合)

Ver 1.10



Ver 1.20



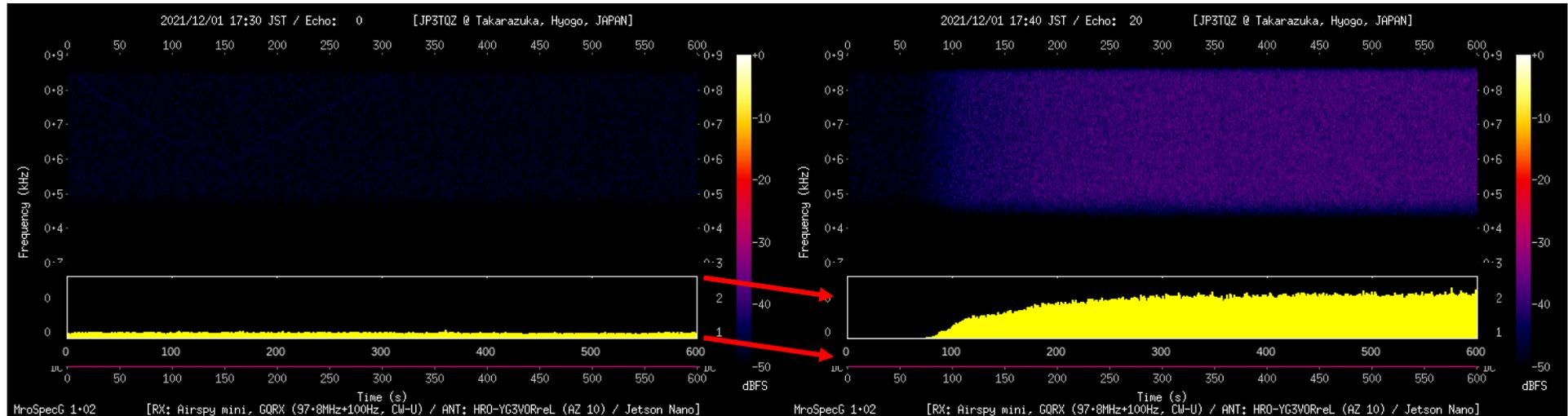
画像処理ライブラリ OpenCV を使用して、信号強度グラフの指定領域をスペクトログラムの指定領域へコピーして、画像を結合。

## (c) 信号強度グラフの表示範囲を自動切換え

Ver 1.10 . . . フロアノイズが上昇したり、長時間の強い信号を受信した場合に、これらの信号がグラフの表示領域を埋め尽くし、信号の強弱を確認できなくなる場合がある。

Ver 1.20 . . . 信号強度の平均が表示範囲(高さ)の 1/2 以上になった場合、表示範囲(高さ)を 1/4 底上げする。以下に動作例を示します。

## 【例 1】



## 【例 2】

