

シート一覧

No.	シート名	内容	備考
1	【1】ラズベリーパイ初回起動時の設定手順	OSの初期設定手順	
2	【2】WindowsPCからSSHでリモート接続	Windowsからラズベリーパイにリモート接続(SSH)	
3	【3】SHから「raspi-config」でVNCを有効化手順	ラズベリーパイのVNCを有効化	
4	【4】WindowsPCからVNCでGUI画面にリモート接続	Windowsからラズベリーパイにリモート接続(VNC)	
5	【5】CO2センサ(mh_z19)を利用するまでの手順	CO2センサの接続から Pythonコマンドで測定値取得 1分ごとの測定値をCSVファイル出力 CSVデータをWebでトレンドグラフ表示	
6	【6】OLEDディスプレイ表示の手順	OLEDをGPIO接続して測定結果を表示するまでの手順	
7	【7】DHT11温湿センサの利用手順	DHT11をGPIO接続して測定結果を表示するまでの手順	
8	【8】アラームブザーの利用手順	BuzzerをGPIO接続してアラームを鳴らすまでの手順	
9	【9】OLEDにQRコード表示の利用手順	OLEDをGPIO接続してQRコードを表示するまでの手順	
10			

【1】ラズベリーパイ初回起動時の設定手順

①「NEXT」で次へ進む



②Country 「Japan」を選択して「NEXT」で次へ進む

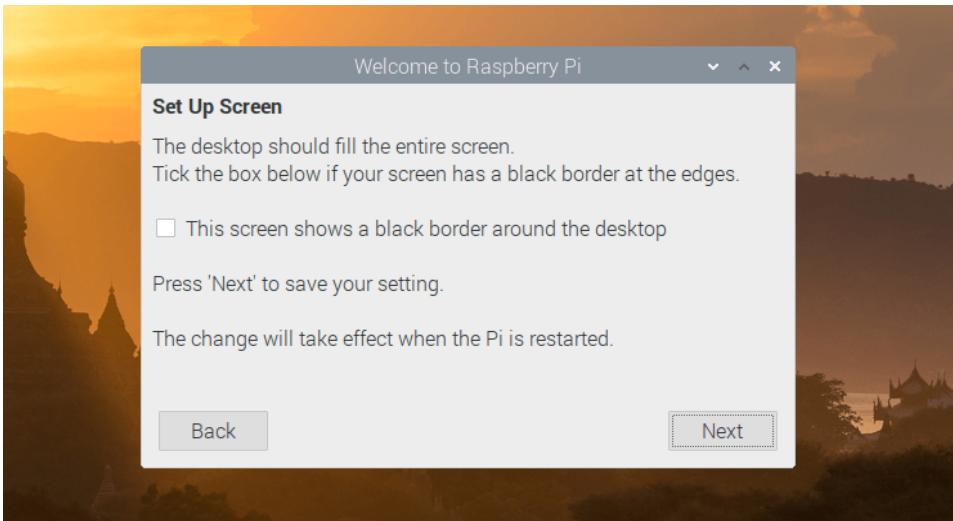


③新しいパスワードを入力して「NEXT」で次へ進む

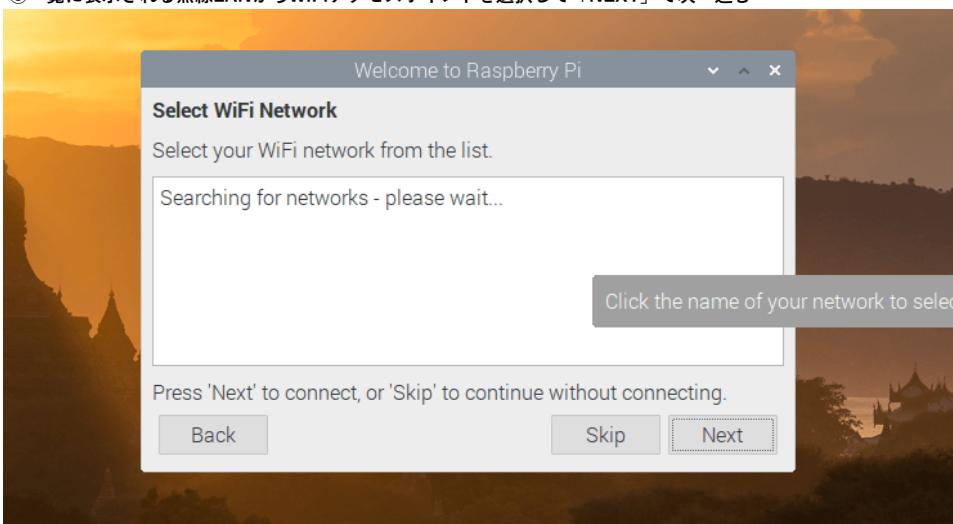
※初期パスワードは「raspberry」



④「NEXT」で次へ進む

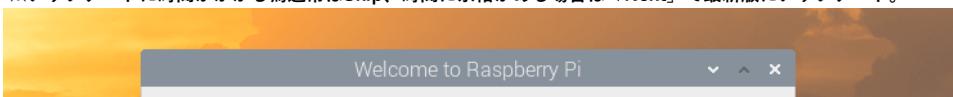


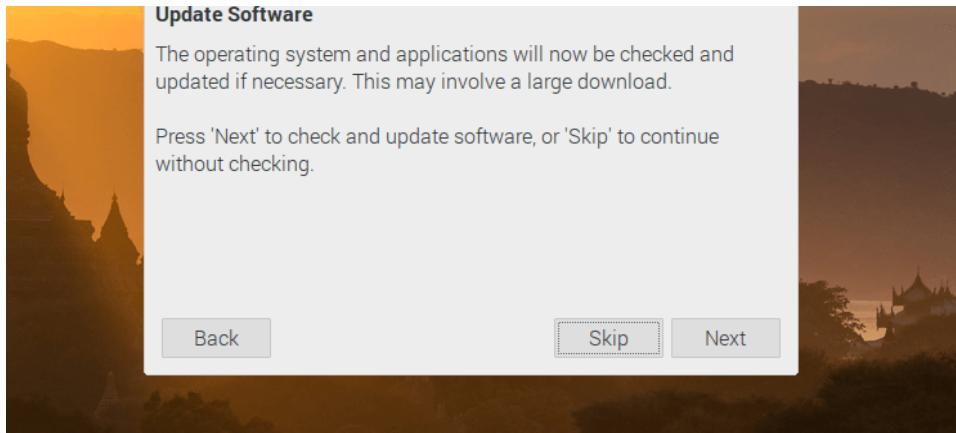
⑤一覧に表示される無線LANからWiFiアクセスポイントを選択して「NEXT」で次へ進む



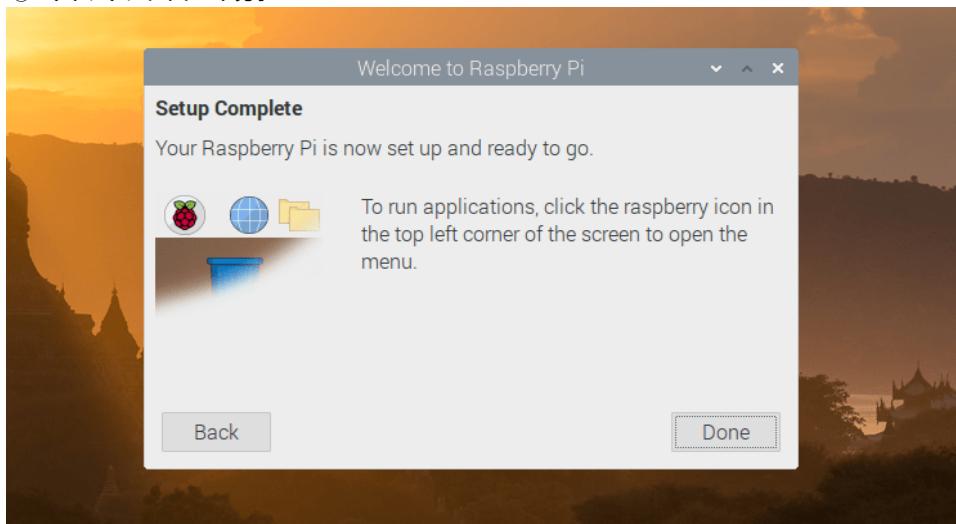
⑥「Skip」で次へ進む

※アップデートに時間がかかる為通常はSkip、時間に余裕がある場合は「Next」で最新版にアップデート。

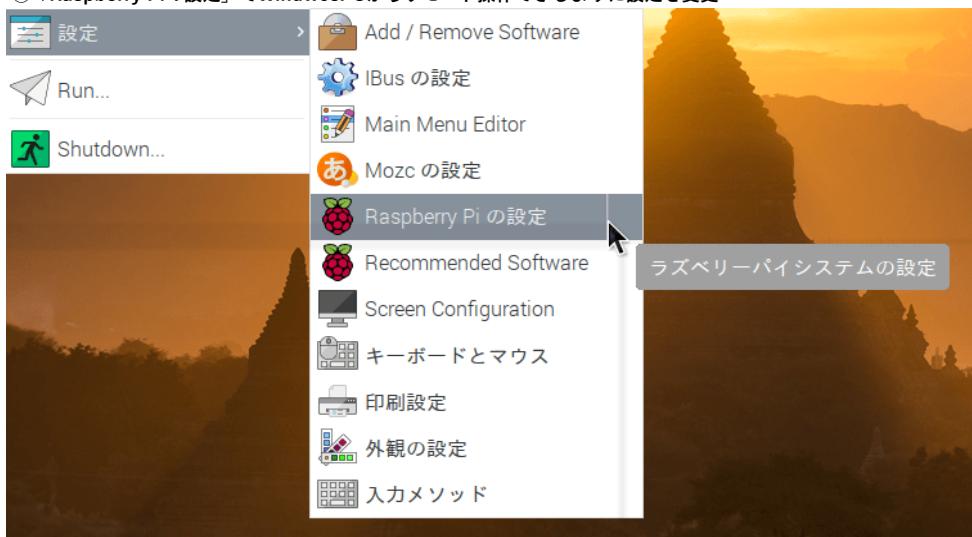




⑦セットアップウィザード完了

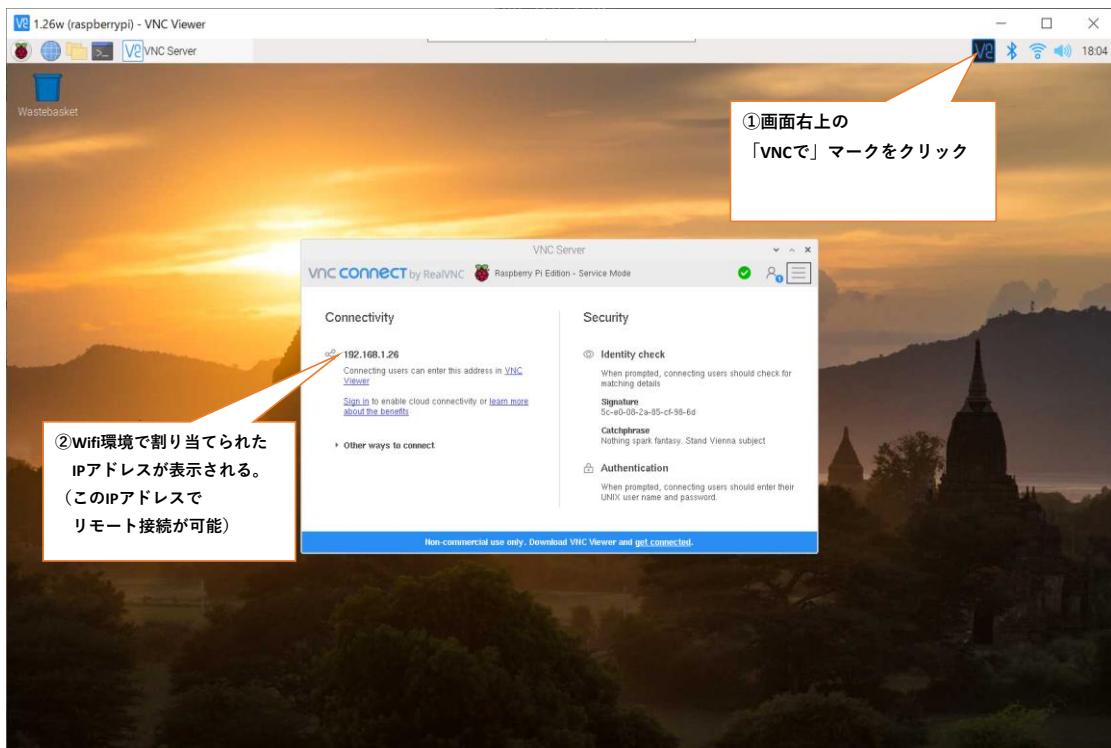


⑧「Raspberry Piの設定」でWindwosPCからリモート操作できるように設定を変更



- ⑨「インターフェース」タブの
【SSH】【VNC】【シリアルポート】を有効に変更
【シリアルコンソール】を無効に設定
「OK」で設定変更を有効化





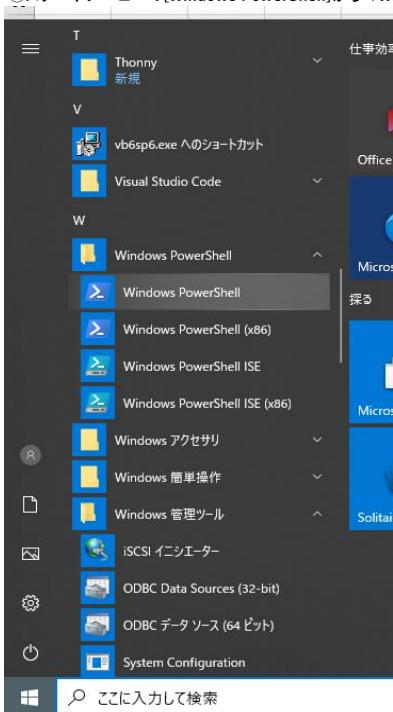
【2】WindowsPCからSSHでリモート接続

Windows標準機能の「PowerShell」でラズベリーパイに接続

※Pythonの実行やライブラリ追加作業はSSHでコマンド実行することで簡単に操作可能です。

CO2センサの操作するまでのセットアップもSSHのコマンド操作で可能です。

①スタートメニューの[Windows PowerShell]から「Windows PowerShell」を起動



②「ssh pi@192.168.x.xx」SSHコマンドで接続先のIPアドレスを指定して接続

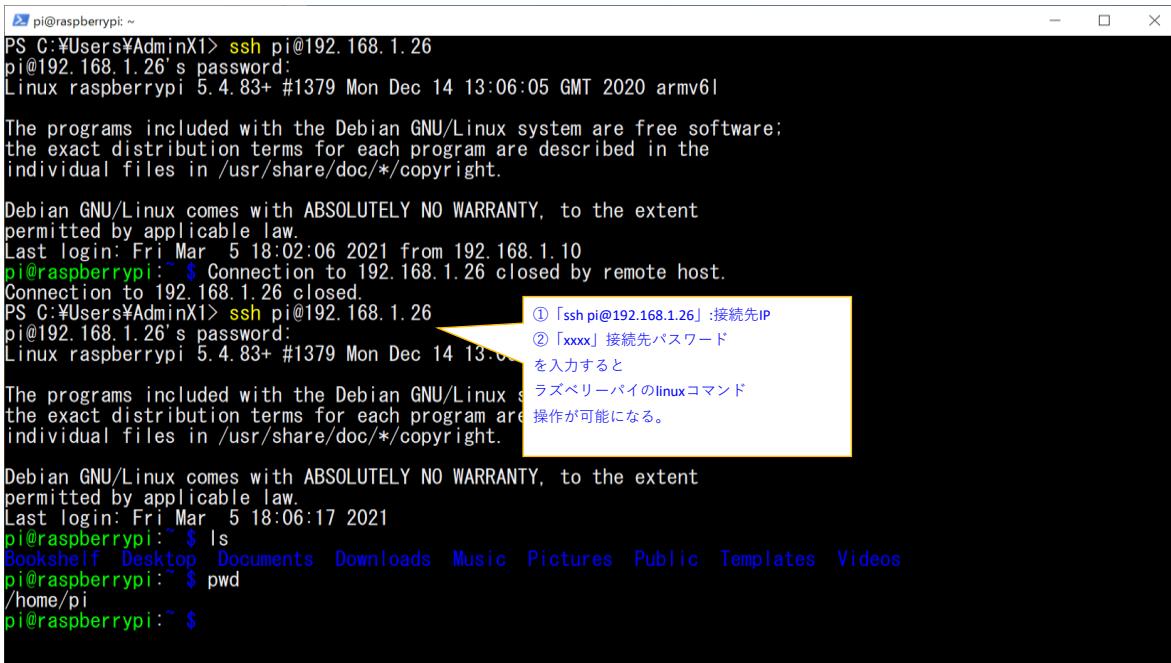
③カレントフォルダのファイル確認

「ls」

④カレントフォルダ位置の確認

「pwd」

※SSHからラズベリーパイを停止するコマンド「sudo halt」



```
pi@raspberrypi: ~
PS C:\Users\AdminX1> ssh pi@192.168.1.26
pi@192.168.1.26's password:
Linux raspberrypi 5.4.83+ #1379 Mon Dec 14 13:06:05 GMT 2020 armv6l

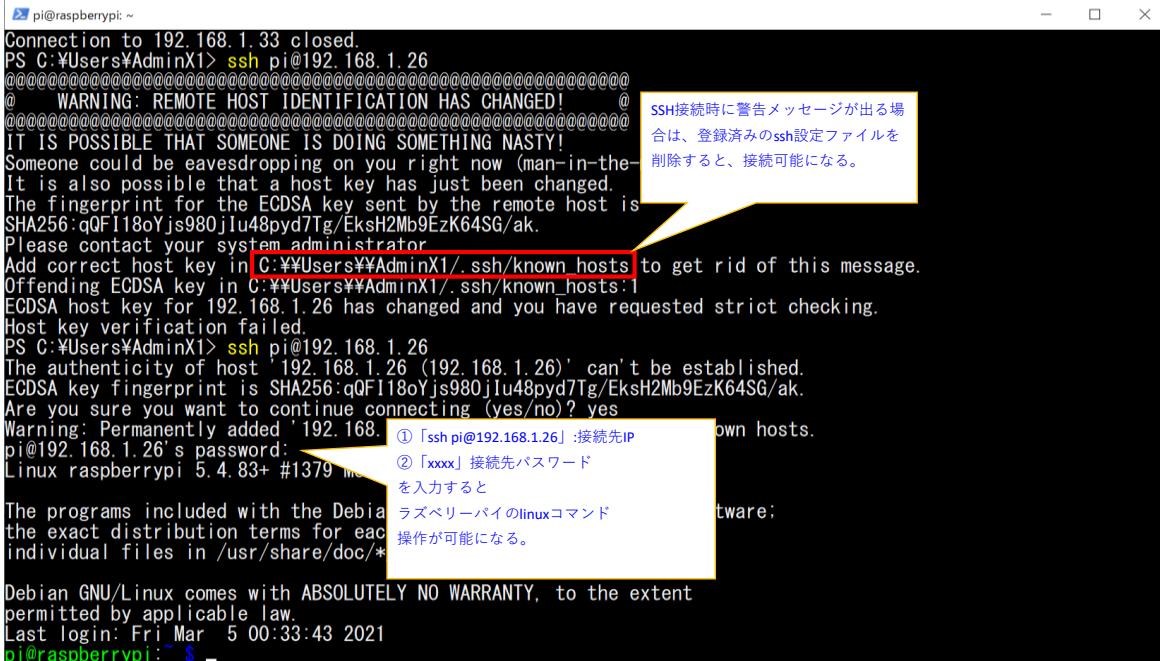
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Mar  5 18:02:06 2021 from 192.168.1.10
pi@raspberrypi: ~ Connection to 192.168.1.26 closed by remote host.
Connection to 192.168.1.26 closed.
PS C:\Users\AdminX1> ssh pi@192.168.1.26
pi@192.168.1.26's password:
Linux raspberrypi 5.4.83+ #1379 Mon Dec 14 13:06:05 GMT 2020 armv6l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Mar  5 18:06:17 2021
pi@raspberrypi: ~ $ ls
Bookshelf Desktop Documents Downloads Music Pictures Public Templates Videos
pi@raspberrypi: ~ $ pwd
/home/pi
pi@raspberrypi: ~ $
```

※接続先の認証情報が登録済みの場合は、一度「known_hosts」ファイルを削除すると接続可能になります。



```
pi@raspberrypi: ~
Connection to 192.168.1.33 closed.
PS C:\Users\AdminX1> ssh pi@192.168.1.26
@@@@@@@@@@@@@@@@@@@WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-
It is also possible that a host key has just been changed.
The fingerprint for the ECDSA key sent by the remote host is
SHA256:qQF118oYjs980jlu48pyd7Tg/EksH2Mb9EzK64SG/ak.
Please contact your system administrator
Add correct host key in C:\Users\AdminX1\.ssh\known_hosts to get rid of this message.
Offending ECDSA key in C:\Users\AdminX1\.ssh\known_hosts:1
ECDSA host key for 192.168.1.26 has changed and you have requested strict checking.
Host key verification failed.
PS C:\Users\AdminX1> ssh pi@192.168.1.26
The authenticity of host '192.168.1.26 (192.168.1.26)' can't be established.
ECDSA key fingerprint is SHA256:qQF118oYjs980jlu48pyd7Tg/EksH2Mb9EzK64SG/ak.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.1.26' (ECDSA) to the list of known hosts.
pi@192.168.1.26's password: [REDACTED]
Linux raspberrypi 5.4.83+ #1379 _[REDACTED]
The programs included with the Debian
the exact distribution terms for each
individual files in /usr/share/doc/*
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Mar 5 00:33:43 2021
pi@raspberrypi: ~ $
```

SSH接続時に警告メッセージが出る場合は、登録済みのssh設定ファイルを削除すると、接続可能になる。

①「ssh pi@192.168.1.26」:接続先IP
②「xxxx」接続先パスワード
を入力すると
ラズベリーバイのlinuxコマンド
操作が可能になる。

【3】SSHから「raspi-config」でVNCを有効化

①「sudo raspi-config」を入力してリターン

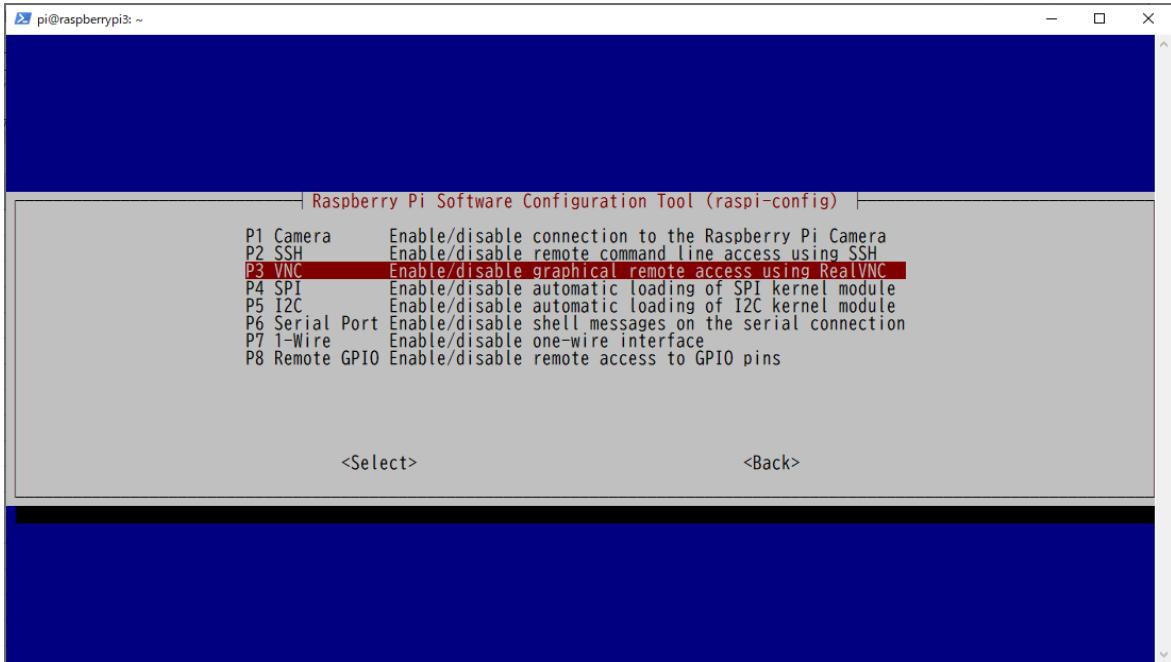
```
pi@raspberrypi3: ~
pi@raspberrypi3: $ sudo raspi-config
```

②「3 Interface Option」を選択してリターン

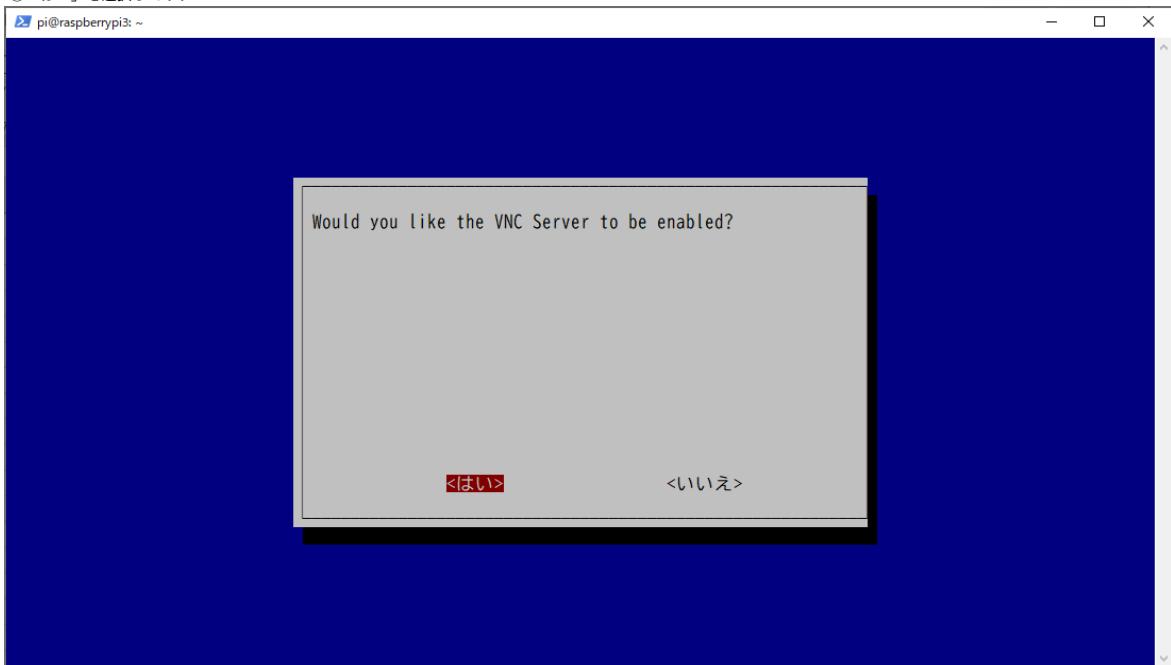
```
pi@raspberrypi3: ~
Raspberry Pi 3 Model B Rev 1.2
Raspberry Pi Software Configuration Tool (raspi-config) |
1 System Options      Configure system settings
2 Display Options    Configure display settings
3 Interface Options  Configure connections to peripherals
4 Performance Options Configure performance settings
5 Localisation Options Configure language and regional settings
6 Advanced Options   Configure advanced settings
8 Update             Update this tool to the latest version
9 About raspi-config  Information about this configuration tool

<Select>           <Finish>
```

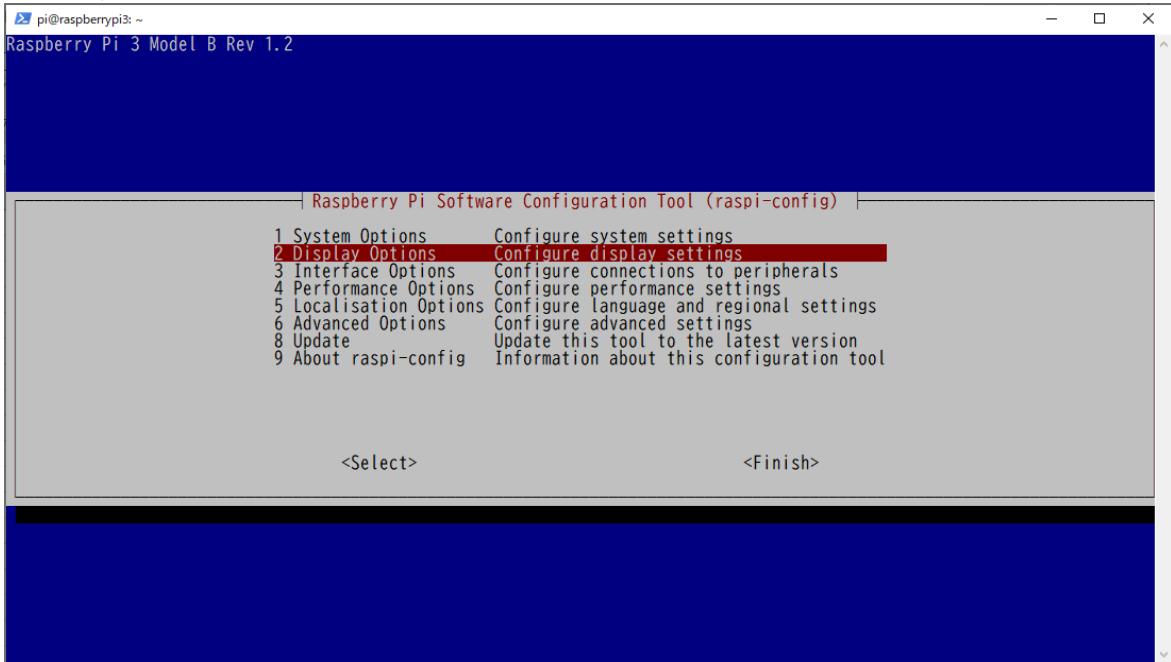
③ 「P3 VNC」を選択してリターン



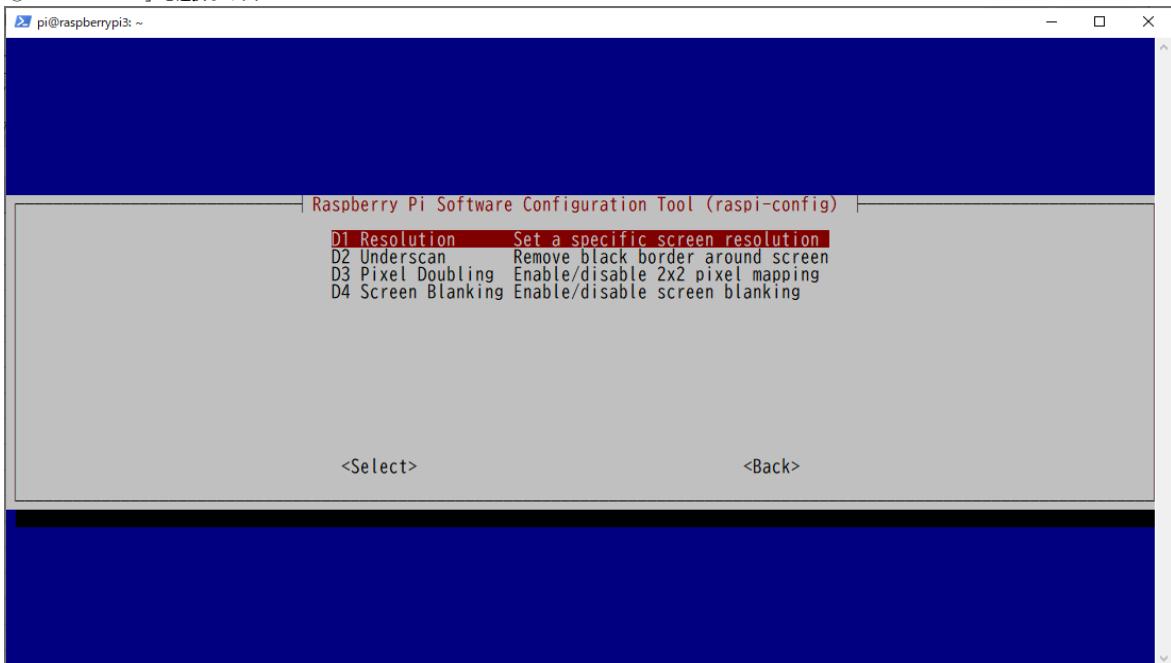
④ 「はい」を選択してリターン



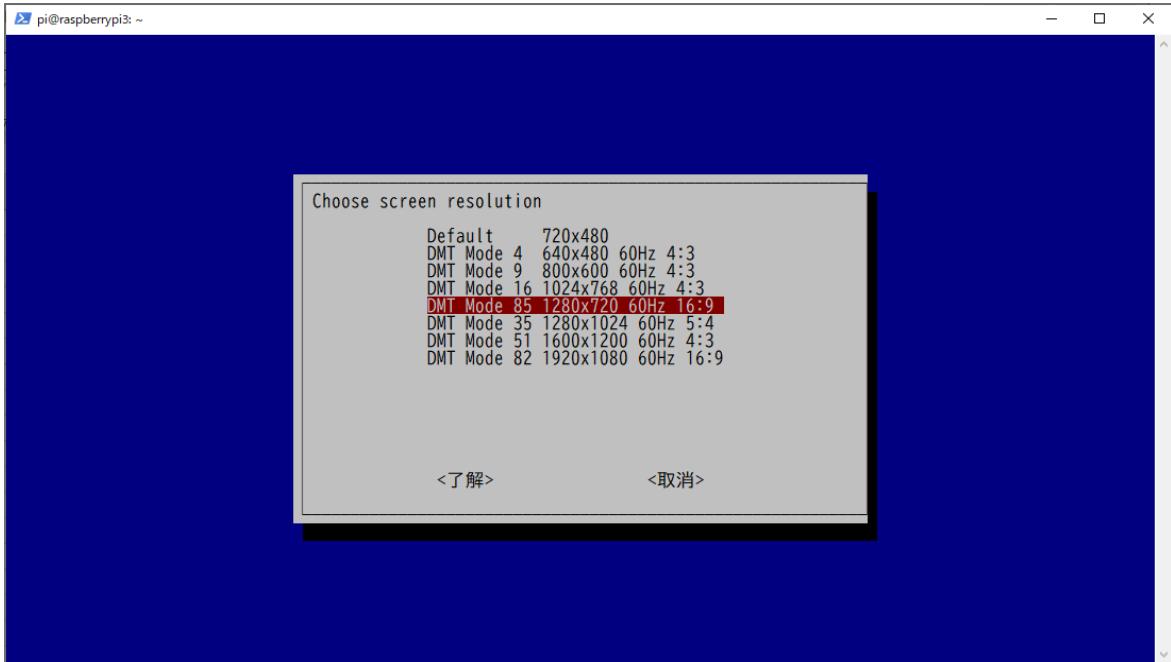
⑤「2 Display Option」を選択してリターン



⑥「D1 Resolution」を選択してリターン



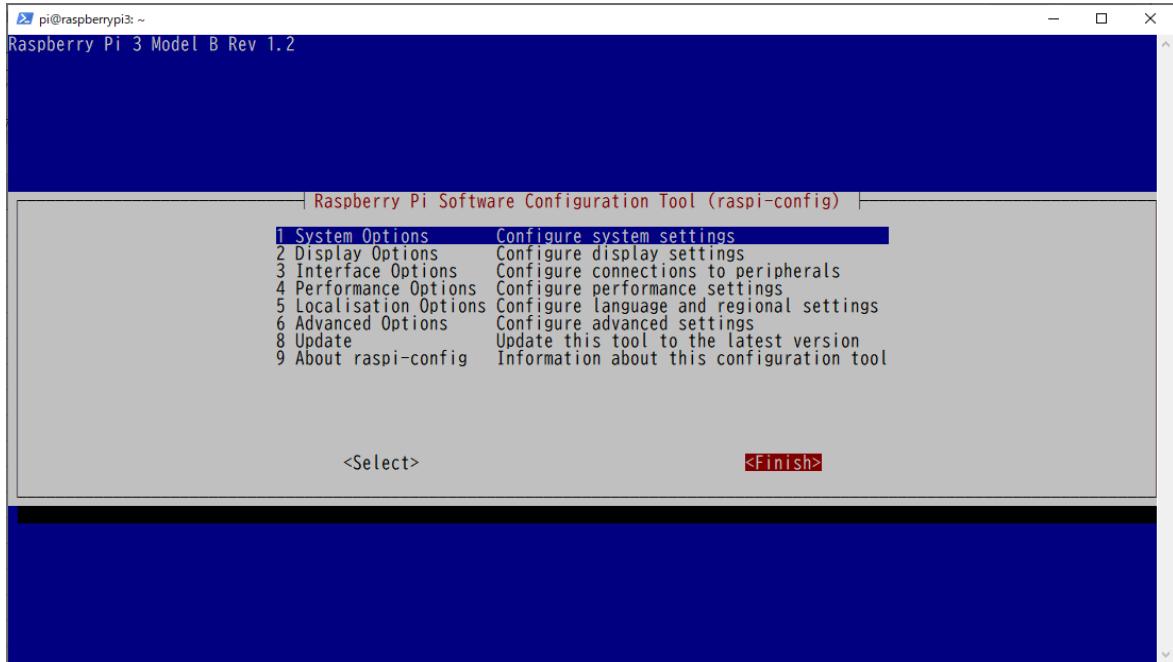
⑦ 「DTM Mode 85」を選択してリターン ※利用したい解像度を選択可能



⑧ 「了解」を選択してリターン



⑨ 「<Finish>」を選択してリターン



【4】WindowsPCからVNCでGUI画面にリモート接続

※WiFiの接続先変更やWiFiパスワードの入力はVNC画面での操作が分かりやすい。
WiFi接続で割り当てられた自分のIPアドレスの確認はGUIでの確認をお勧めします。

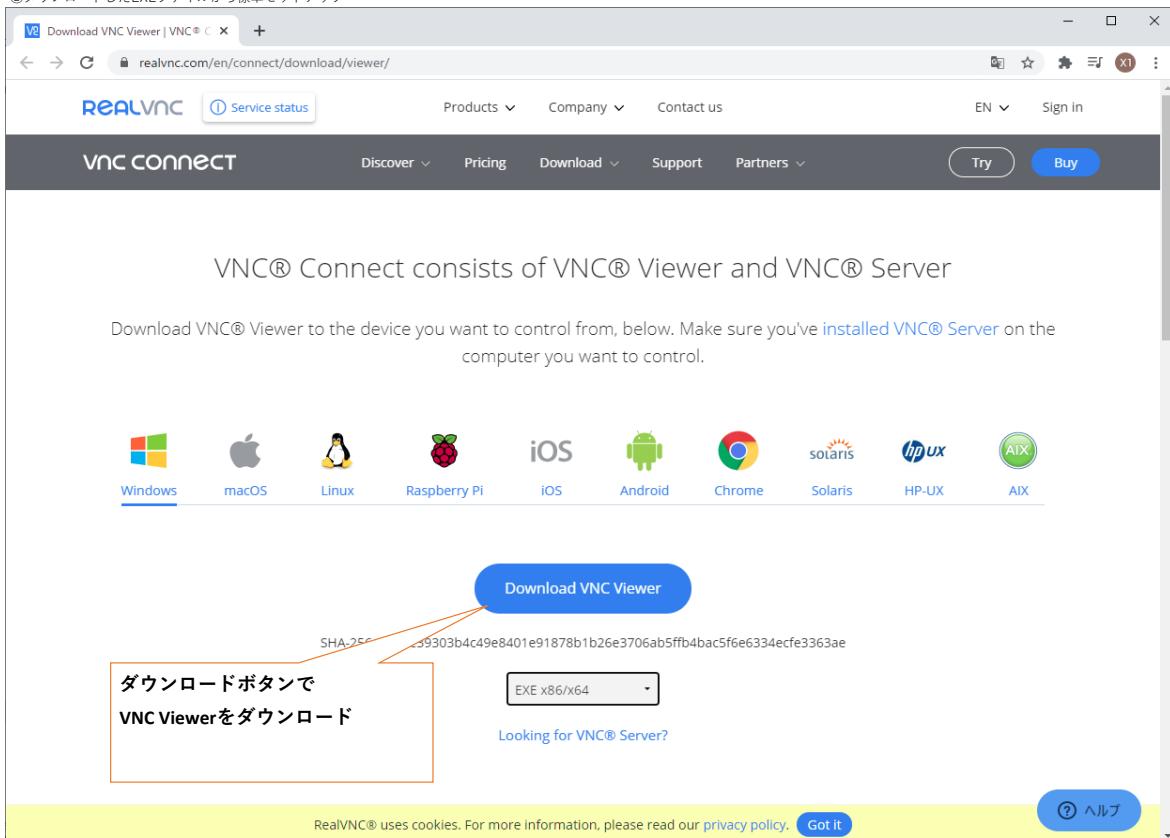
参照URL：

VNC Viewerダウンロードページ

「<https://www.realvnc.com/en/connect/download/viewer/>」

①ダウンロードボタンでVNC Viewerをダウンロード

②ダウンロードしたEXEファイルから標準セットアップ



VNC® Connect consists of VNC® Viewer and VNC® Server

Download VNC® Viewer to the device you want to control from, below. Make sure you've [installed VNC® Server](#) on the computer you want to control.

Windows macOS Linux Raspberry Pi iOS Android Chrome Solaris HP-UX AIX

Download VNC Viewer

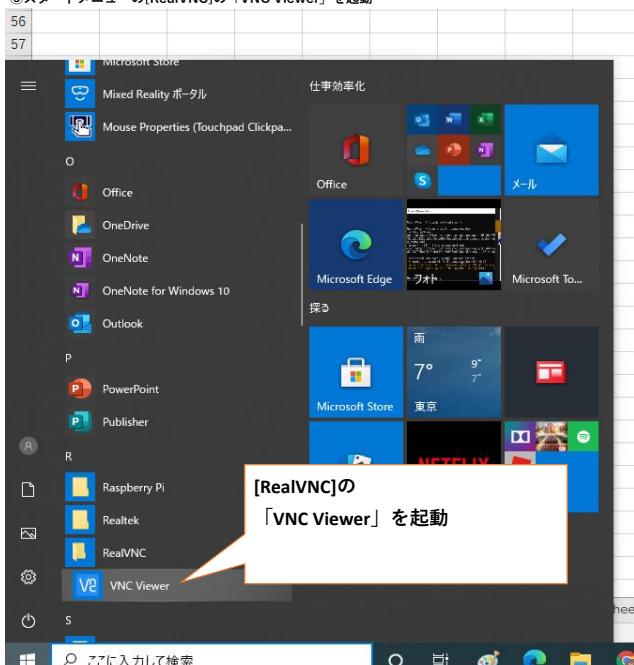
SHA-256: 59303b4c49e8401e91878b1b26e3706ab5ffb4bac5f6e6334ecfe3363ae

EXE x86/x64

Looking for VNC® Server?

RealVNC® uses cookies. For more information, please read our [privacy policy](#). [Got it](#)

③スタートメニューの[RealVNC]の「VNC Viewer」を起動



④接続するraspberrypiの「IPアドレス」入力

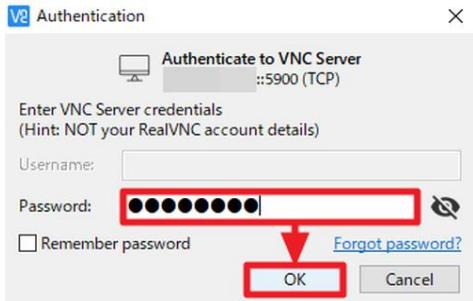


There are no computers in your address book at present.

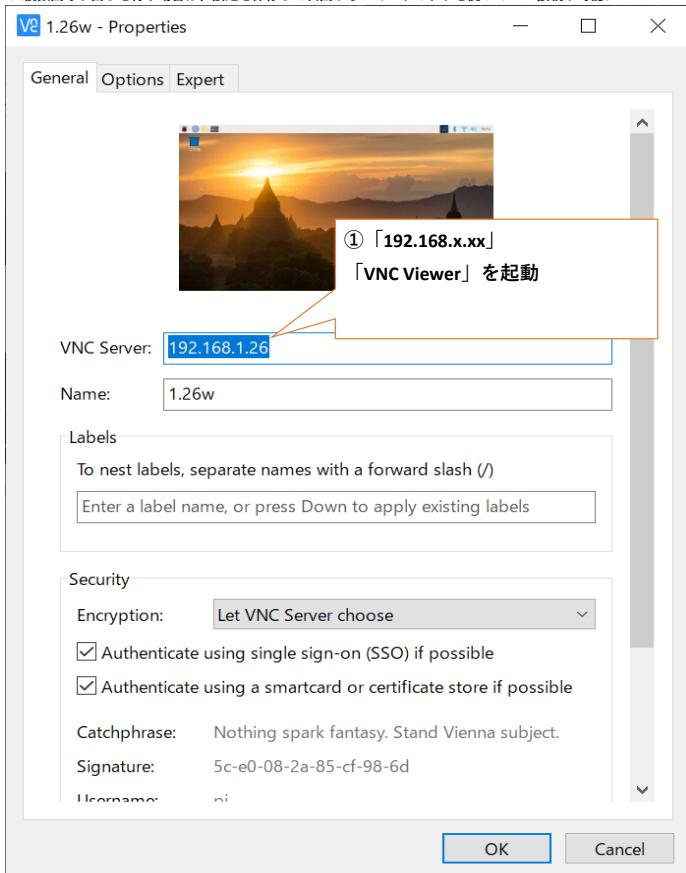
Sign in to your RealVNC account to automatically discover team computers.

Alternatively, enter the VNC Server IP address or hostname in the Search bar to connect directly.

⑤接続するraspberrypiの「パスワード」を入力



※複数回呼び出しを行う場合は、設定を保存して次回からショートカットを使ってVNC接続が可能に



【5】CO2センサ(mh_z19)を利用するまでの手順

(センサをGPIO接続して測定結果をグラフ表示するまでの手順)

1. CO2センサ(mh_z19)とラズベリーパイのGPIOを接続
2. Pythonライブラリの追加と実行
3. 取得結果利用のライブラリの追加
4. 取得結果のファイル出力、Web表示用のファイルをラズベリーパイにコピー
5. ファイル送信したバッチコマンドファイルを実行権限を設定して、スケジュール実行設定を登録
6. Pythonの簡易Webサーバ機能でco2、温度のトレンドグラフを表示

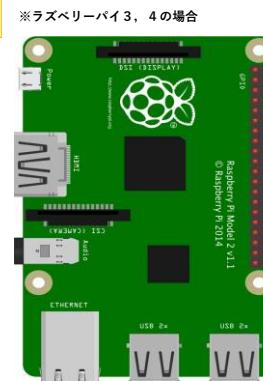
1. CO2センサ(mh_z19)とラズベリーパイのGPIOを接続

RPiポート④⑥⑧⑩(+5V,GND,TxD,RxD) ⇌ mh_z19(Vin,GND,TxD,RxD)



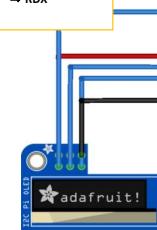
Raspberry Pi 2/3
Raspberry Pi Zero / Zero W

ポート④ : +5V
ポート⑥ : GND
ポート⑧ : TXD
ポート⑩ : RXD
を接続

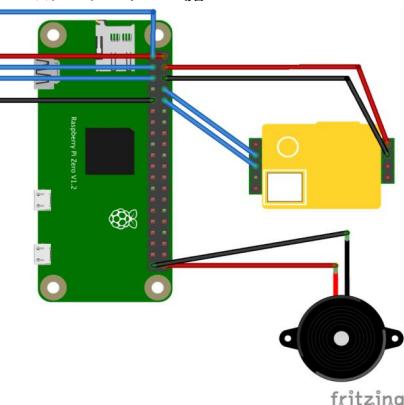


※ラズベリーパイ 3, 4 の場合

ポート④ : +5V → Vin
ポート⑥ : GND → GND
ポート⑧ : TXD → TDX
ポート⑩ : RXD → RDX



※ラズベリーパイ Zero の場合



2. Pythonライブラリの追加と実行

- ①CO2センサ(mh_z19)用のPython3用ライブラリ追加

「sudo pip3 install mh-z19」

②測定結果を取得

「sudo python3 -m mh_z19」 co2の測定値取得
「sudo python3 -m mh_z19 --all」 co2と温度の測定値取得

```
pi@raspberrypi: ~
Last login: Fri Mar  5 18:20:38 2021
Linux raspberrypi 5.4.83+ #1379 Mon Dec 14 13:06:05 GMT 2020 armv6l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Mar  5 18:20:38 2021
pi@raspberrypi: ~ $ sudo pip3 install mh-z19
Looking in indexes: https://pypi.org/simple, https://www.piwheels.org/simple
Collecting mh-z19
  Downloading https://www.piwheels.org/simple/mh-z19-3.0.1-py3-none-any.whl
Requirement already satisfied: RPi.GPIO in /usr/lib/python3/dist-packages (from mh-z19) (0.7.0)
Collecting getrpimodel (from mh-z19)
  Downloading https://www.piwheels.org/simple/getrpimodel/getrpimodel-0.1.19-py3-none-any.whl
Requirement already satisfied: pyserial in /usr/lib/python3/dist-packages (from mh-z19) (3.4)
Collecting argparse (from mh-z19)
  Downloading https://files.pythonhosted.org/packages/f2/94/3af39d34be01a24a6e65433d19e107099374224905f1e0cc6b
be1fd22a2f/argparse-1.4.0-py2.py3-none-any.whl
Requirement already satisfied: requests in /usr/lib/python3/dist-packages (from mh-z19) (2.21.0)
Installing collected packages: getrpimodel, argparse, mh-z19
Successfully installed argparse-1.4.0 getrpimodel-0.1.19 mh-z19-3.0.1
pi@raspberrypi: ~ $ sudo python3 -m mh_z19
[{"co2": 864}
pi@raspberrypi: ~ $ sudo python3 -m mh_z19 --all
[{"co2": 851, "temperature": 50, "TT": 90, "SS": 0, "UhUI": 0}
pi@raspberrypi: ~ $ sudo apt install -y jq
パッケージリストを読み込んでいます... 完了
依存関係ツリーを作成しています... 0%
```

3. 取得結果可能用のライブラリの追加

- ①jqをインストールする
- JSON抽出コマンド。
- 『sudo apt install -y jq』

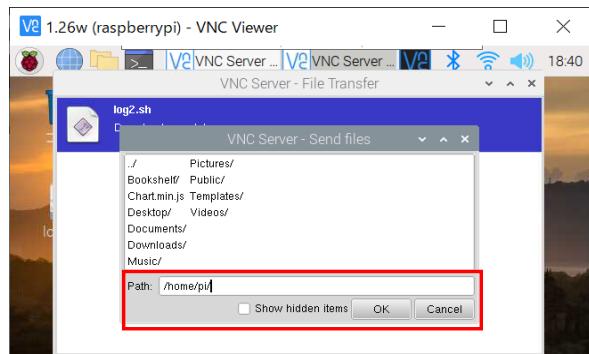
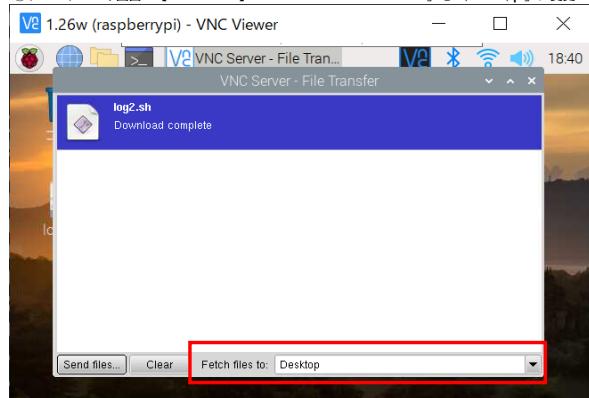
4. 取得結果のファイル出力、Web表示用のファイルをラズベリーバイにコピー

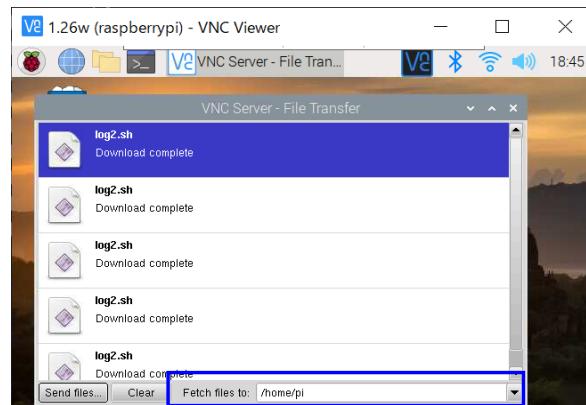
【パッチ処理用ファイル】	Python コマンドを実行して取得された測定結果の「co2」と「Temperature」の値をCSV形式でファイル出力する。
log2.sh	1 日分のco2測定結果をファイル出力
2021-03-05.log	1 日分のco2測定結果をファイル出力
2021-03-05_single.log	co2測定結果の直近 20 回分をファイル出力
【Web表示用ファイル】	
index.html	1 日分のco2測定結果をWebでグラフ表示
index_single.html	直近20回分のco2測定結果をWebでグラフ表示
mychart.js	1 日分のco2測定結果グラフ表示用JavaScript
mychart_single.js	直近20回分のco2測定結果グラフ表示用JavaScript (自動更新機能あり)
Chart.min.js	測定結果グラフ表示用JavaScriptライブラリ

- ①VNCでラズベリーパイに接続
②VNC画面の上側の「Transfer files」をクリック

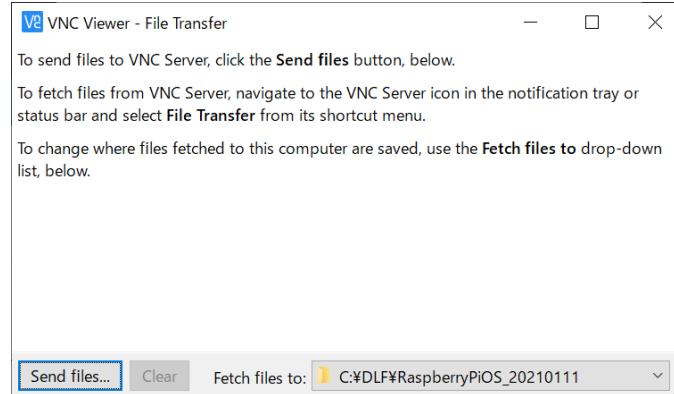


- ③ラズベリーパイ画面の【VNC Server】FileTransferの「Fetch file to」を「/home/pi」に変更

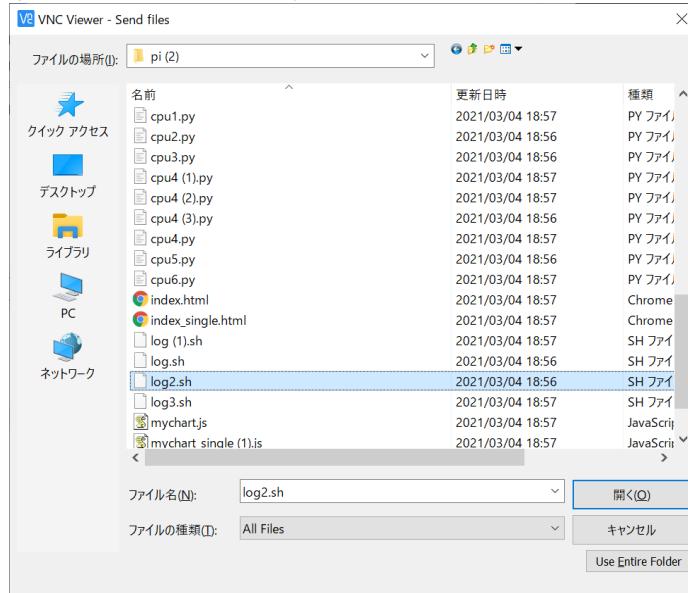




④ WindowsPCの【VNC Viewer】FileTransferの「Send files」をクリック



⑤WindowsPCで送信対象のファイルを選択する。



5. ファイル送信したバッチコマンドファイルを実行権限を設定して、スケジュール実行設定を登録

【バッチ処理用ファイル】

log2.sh Python コマンドを実行して取得された測定結果の「co2」と「Temperature」の値をCSV形式でファイル出力する。

```
#!/bin/sh
echo "hello world"

dateA="date '+%Y-%m-%d %H:%M:%S'"
dateB="date '+%Y-%m-%d.log'"
dateS="date '+%Y-%m-%d_single.log'"

result=`sudo python3 -m mh_z19 | jq '.co2'`
result2=`sudo python3 -m mh_z19 --all | jq '.temperature'`

echo "${dateA},${result},${result2}" >> "${dateB}"
tail "${dateB}" -n 20 > "${dateS}"
echo "${dateA},${result},${result2}"
```

※動作確認用のメッセージ

※年月日時分秒を取得

※年月日.logファイル名をセット（1日分記録用ファイル）

※年月日_single.logファイル名をセット（部分描画用ファイル）

Pythonコマンドでco2の測定結果を取得

Pythonコマンドでco2、温度の測定結果を取得

※1日分記録用ファイルにco2、気温のデータを追記

※年月日_single.logファイル名をセット（部分描画用ファイル）

①「log2.sh」に実行権限を設定
「sudo chmod 755 log2.sh」

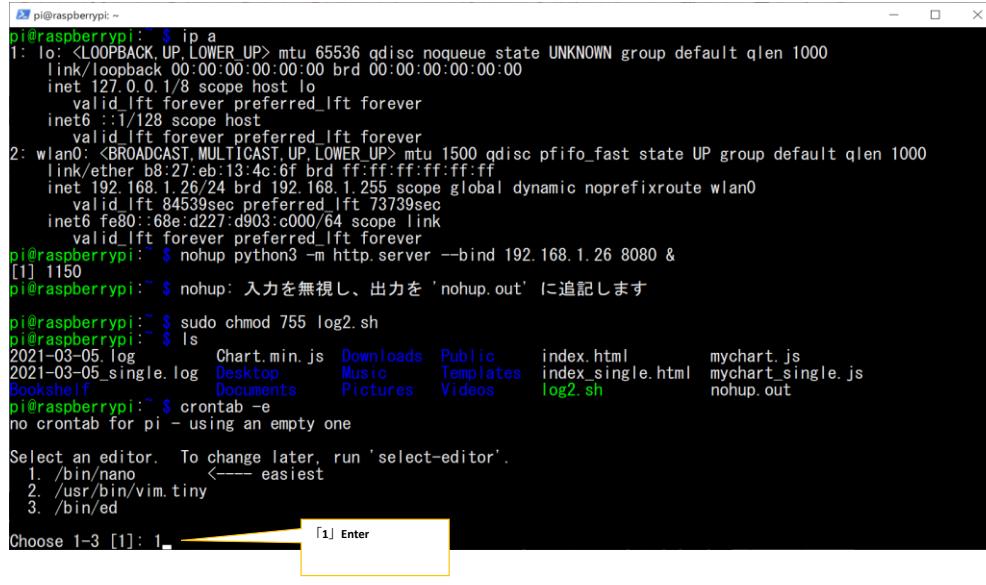
②「log2.sh」がスケジュール実行されるようにcron設定登録

「crontab -e」

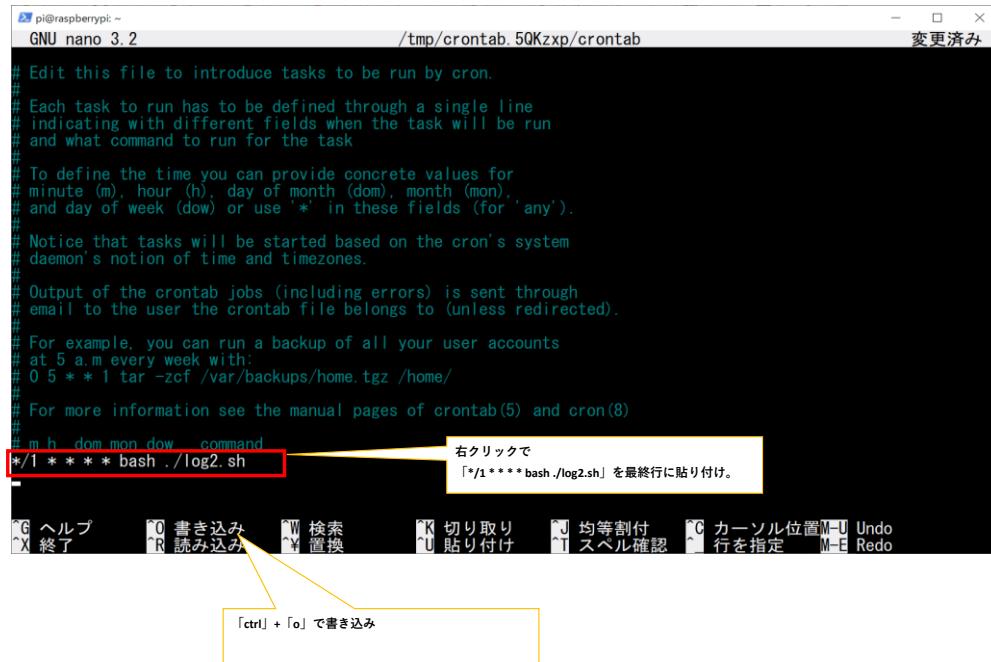
※「sudo」を付けない

③実行設定を追加

```
/*1 * * * * bash ./log2.sh
@reboot bash ./webstart.sh
@reboot sudo nohup python3 -m run_server.py >> /home/pi/web_https.log 2>&1 &】
を最終行に貼り付け。
「ctrlJ」+「O」
enter
「ctrlJ」+「X」
```



```
pi@raspberrypi: ~
pi@raspberrypi: ~ $ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
            valid_lft forever preferred_lft forever
        inet6 ::1/128 scope host
            valid_lft forever preferred_lft forever
2: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether b8:27:eb:13:4c:6f brd ff:ff:ff:ff:ff:ff
        inet 192.168.1.26/24 brd 192.168.1.255 scope global dynamic noprefixroute wlan0
            valid_lft 84539sec preferred_lft 73739sec
        inet6 fe80::b827:ebff:fe13:4c6f/64 scope link
            valid_lft forever preferred_lft forever
pi@raspberrypi: ~ $ nohup python3 -m http.server --bind 192.168.1.26 8080 &
[1] 1150
pi@raspberrypi: ~ $ nohup: 入力を無視し、出力を 'nohup.out' に追記します
pi@raspberrypi: ~ $ sudo chmod 755 log2.sh
pi@raspberrypi: ~ $ ls
2021-03-05.log  Chart.min.js  Downloads  Public  index.html  mychart.js
2021-03-05_single.log  Desktop  Music  Templates  index_single.html  mychart_single.js
Bookshelf  Documents  Pictures  Videos  log2.sh  nohup.out
pi@raspberrypi: ~ $ 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
```

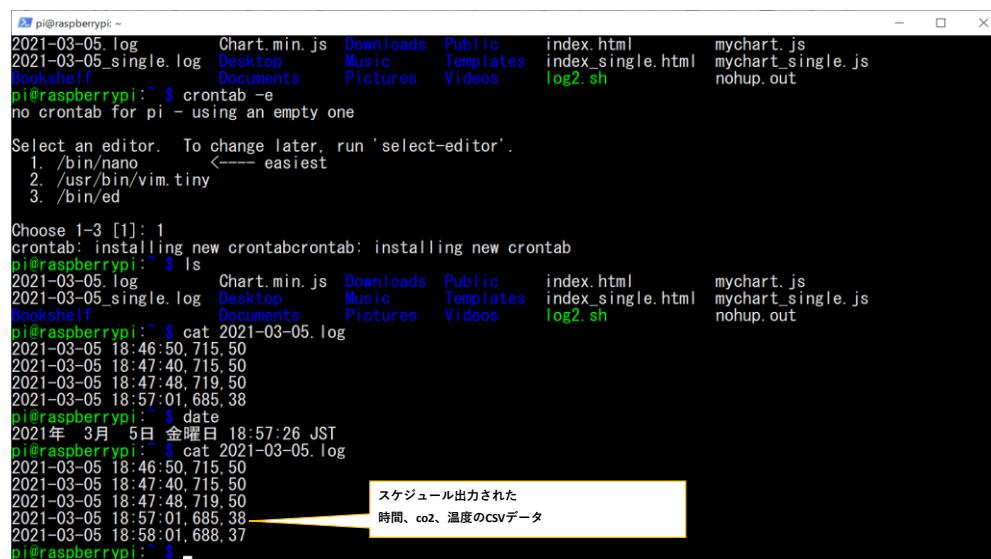


```
pi@raspberrypi: ~
GNU nano 3.2
/tmp/crontab.50Kxp/crontab
# Edit this file to introduce tasks to be run by cron.

# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow command
*/1 * * * * bash ./log2.sh
```

右クリックで
/*1 * * * * bash ./log2.sh を最終行に貼り付け。

「ctrl」+「o」で書き込み



```
pi@raspberrypi: ~
2021-03-05.log  Chart.min.js  Downloads  Public  index.html  mychart.js
2021-03-05_single.log  Desktop  Music  Templates  index_single.html  mychart_single.js
Bookshelf  Documents  Pictures  Videos  log2.sh  nohup.out
pi@raspberrypi: ~ $ 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
crontab: installing new crontab
pi@raspberrypi: ~ $ ls
2021-03.log  Chart.min.js  Downloads  Public  index.html  mychart.js
2021-03-05.log  Desktop  Music  Templates  index_single.html  mychart_single.js
Bookshelf  Documents  Pictures  Videos  log2.sh  nohup.out
pi@raspberrypi: ~ $ cat 2021-03-05.log
2021-03-05 18:46:50,715,50
2021-03-05 18:47:40,715,50
2021-03-05 18:47:48,719,50
2021-03-05 18:57:01,685,38
pi@raspberrypi: ~ $ date
2021年 3月 5日 金曜日 18:57:26 JST
pi@raspberrypi: ~ $ cat 2021-03-05.log
2021-03-05 18:46:50,715,50
2021-03-05 18:47:40,715,50
2021-03-05 18:47:48,719,50
2021-03-05 18:57:01,685,38
pi@raspberrypi: ~ $
```

スケジュール出力された

時間、co2、温度のCSVデータ

6. Pythonの簡易Webサーバ機能でco2、温度のトレンドグラフを表示

【Web表示用ファイル】

index.html

1日分のco2測定結果をWebでグラフ表示

index_single.html

直近20回分のco2測定結果をWebでグラフ表示

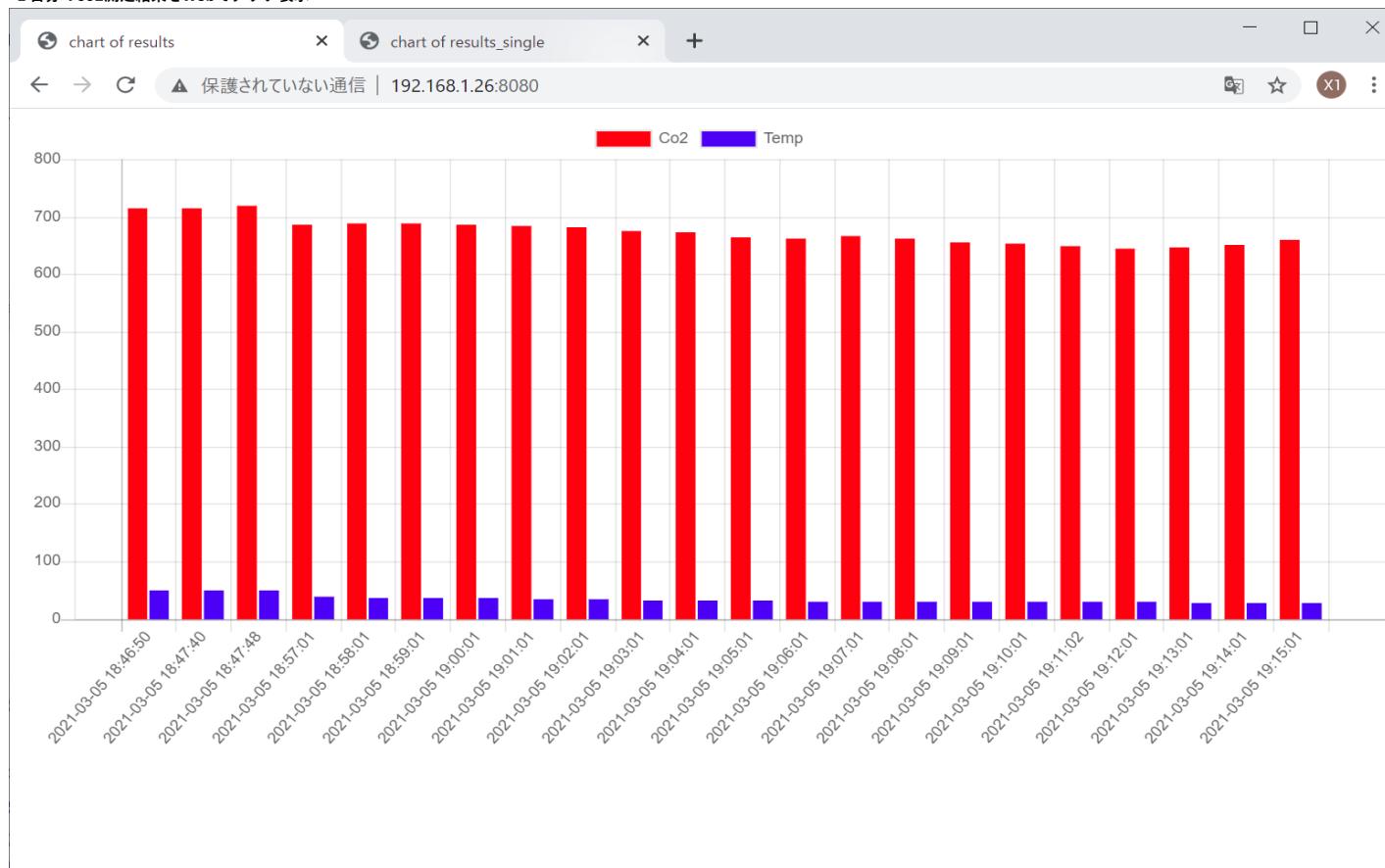
①WebサーバープロセスをIPアドレスを指定して実行

「nohup python3 -m http.server --bind 192.168.1.26 8080 &」

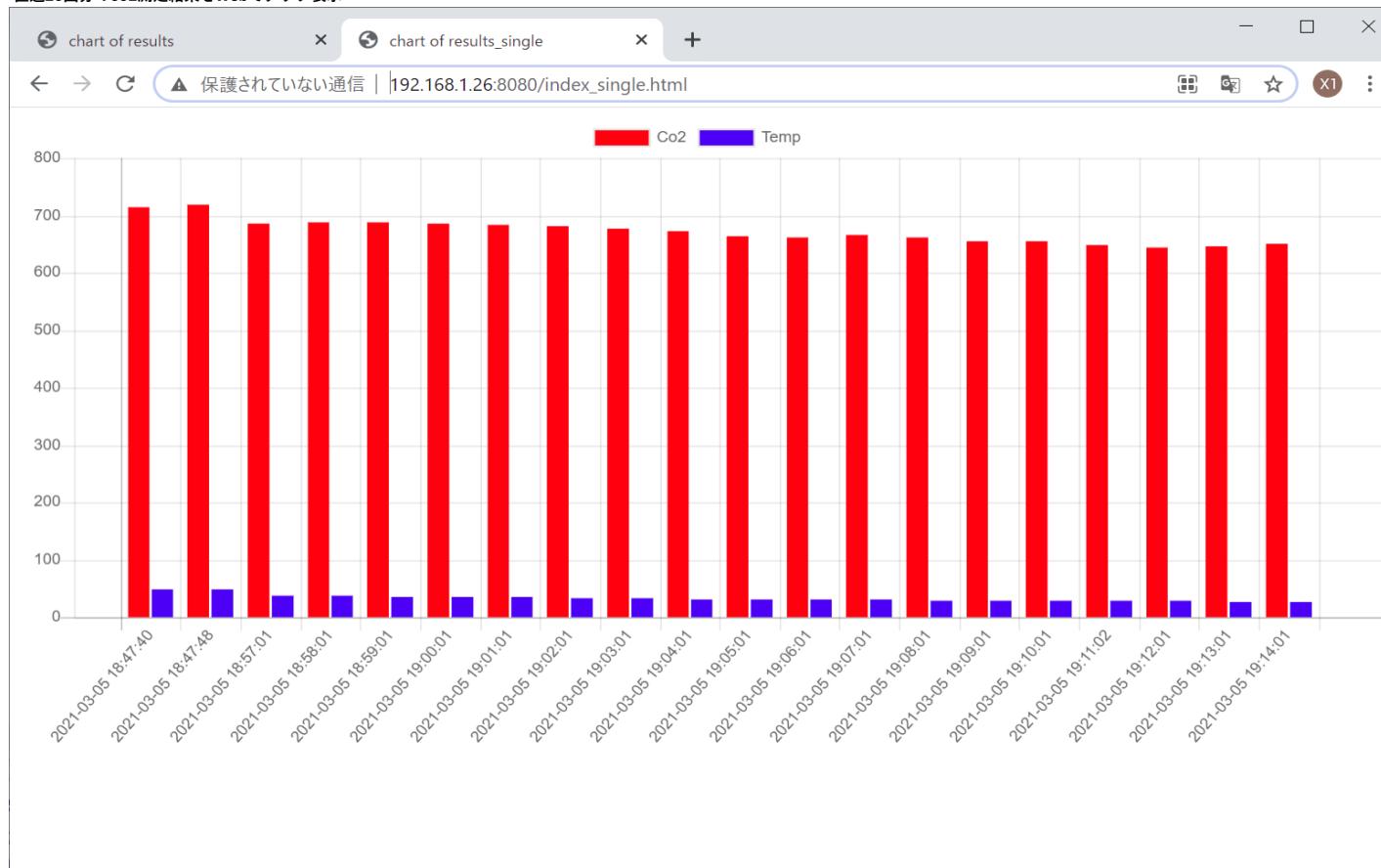
Enter

※IPアドレスにはラズベリーパイのIPアドレスをセット

1日分のco2測定結果をWebでグラフ表示



直近20回分のCO2測定結果をWebでグラフ表示



【6】OLEDディスプレイ表示の手順

(OLEDをGPIO接続して測定結果を表示するまでの手順)

1. OLEDとラズベリーパイのGPIOを接続

2. OLED用のPythonライブラリのダウンロードとセットアップ

```
「wget https://github.com/h-nari/raspioled/archive/refs/heads/master.zip」
```

```
「sudo mv master.zip raspioled_master.zip」
```

```
「unzip raspioled_master.zip」
```

```
「cd raspioled-master/」
```

```
「$ sudo python3 setup.py install」
```

3. OLED処理で使用するライブラリの追加

```
「$ sudo pip3 install Pillow」
```

4. 取得結果のファイル出力、OLED表示用のファイルをラズベリーパイにコピー

①OLED_speak_text_0602.py(OLEDにCO2濃度、温度、しきい値判定メッセージ表示)

②OLED_text_2ROW_0602.py(OLEDにURLテキスト表示)

③OLED_image_0602.py(OLEDにQRコードイメージ表示)

④log0602_MHZ_OLED.sh

(CO2測定結果取得、OLED結果表示、CO2結果ページQRコードOLED表示)

5. 表示用の日本語フォントを追加。

```
sudo apt-get install fonts-noto
```

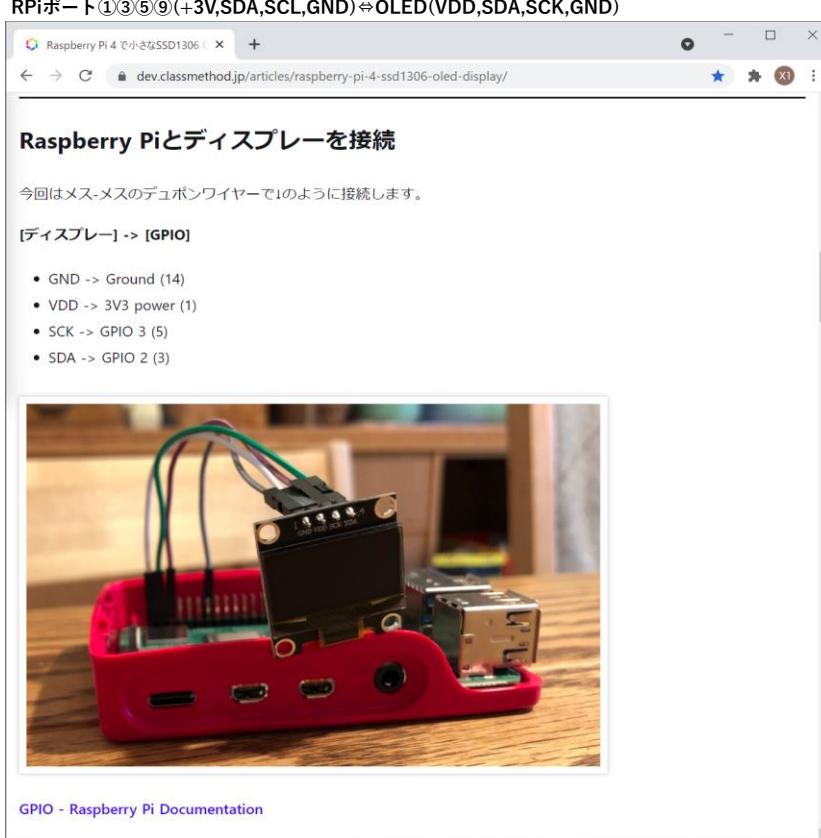
6. ファイル送信したバッチコマンドファイルを実行権限を設定して、スケジュール実行設定を登録

```
*/1 * * * * bash ./log0602_MHZ_OLED.sh
```

7. Pythonでco2、温度を表示

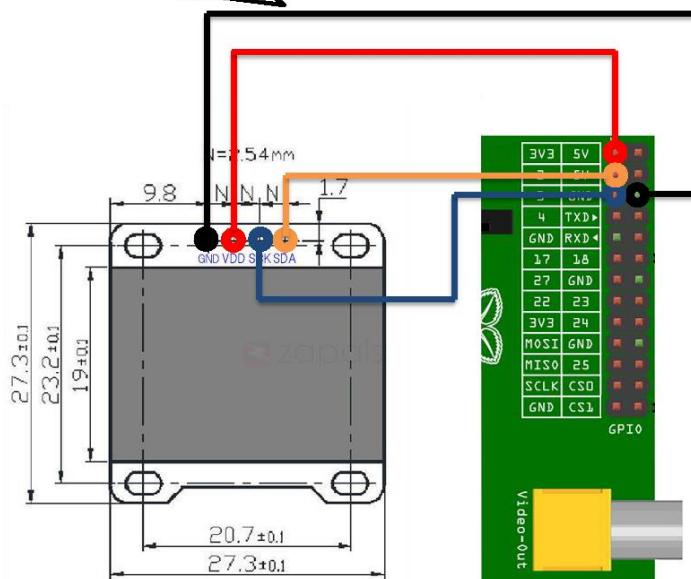
1. OLEDとラズベリーパイのGPIOを接続

RPiポート①③⑤⑨(+3V,SDA,SCL,GND) ⇄ OLED(VDD,SDA,SCK,GND)

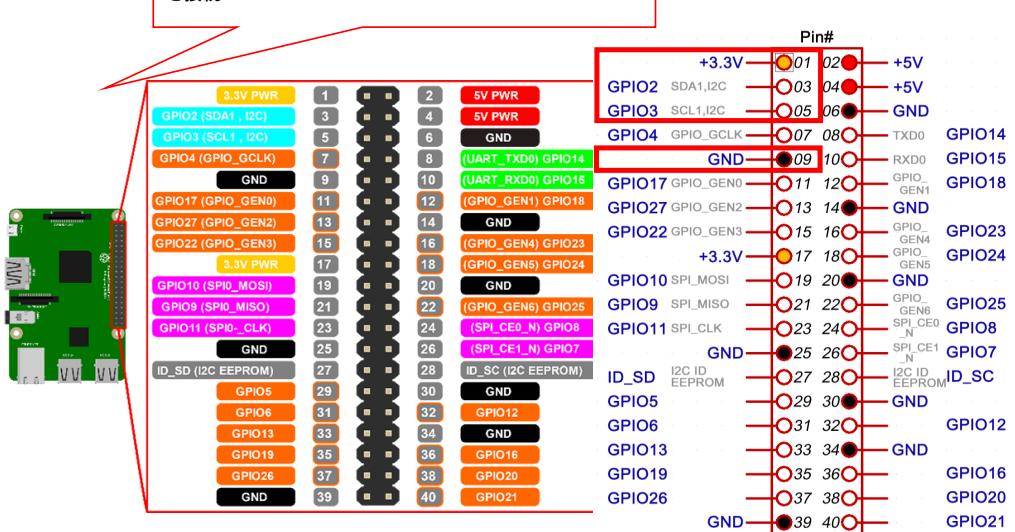


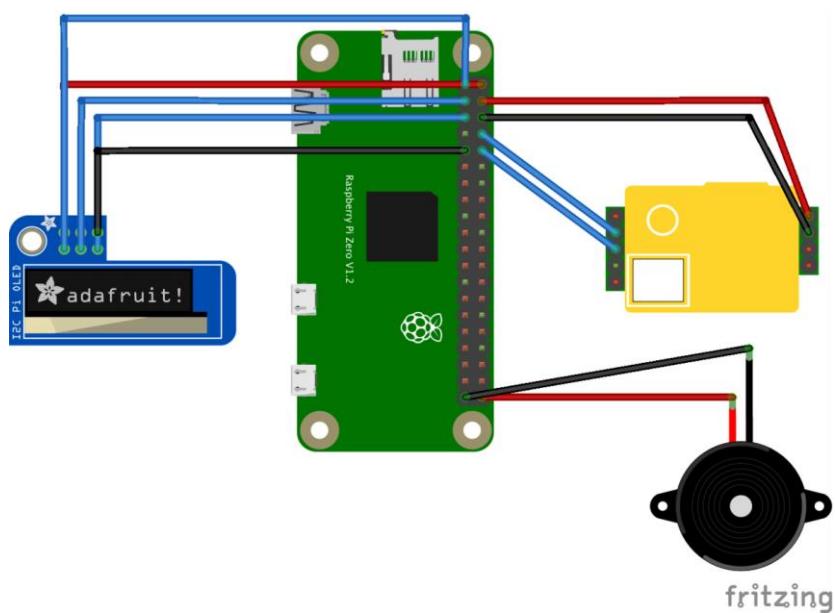
電源のピン配列に注意!!

ピン配列が VDD-GND-SCK-SDA の順の OLED もあり、その場合 VDD と GND の配線がこの図と逆になります。



ポート① : +3V \Rightarrow VDD
 ポート③ : SDA \Rightarrow SDA
 ポート⑤ : SCL \Rightarrow SCK
 ポート⑨ : GND \Rightarrow GND ※図と異なるGNDでも可です。
 を接続





【7】DHT11温湿度センサの利用手順

(DHT11をGPIO接続して測定結果を表示するまでの手順)

1. DHT11とラズベリーパイのGPIOを接続

2. Pythonライブラリの追加

```
pip3 install dht11
```

3. 測定結果取得用のファイルをラズベリーパイにコピー

```
DHT11_Read_pin7_GPIO4.py
```

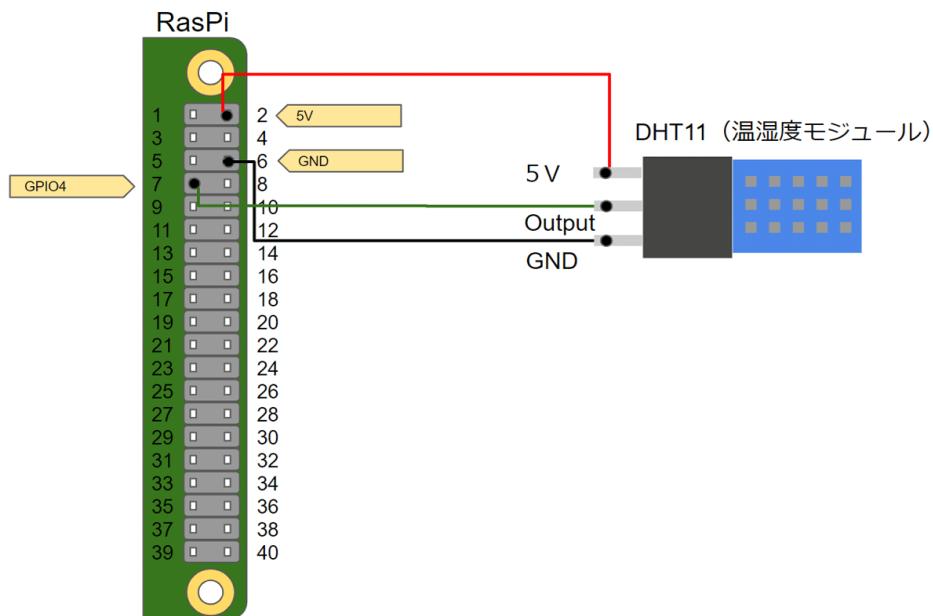
4. 測定結果取実行

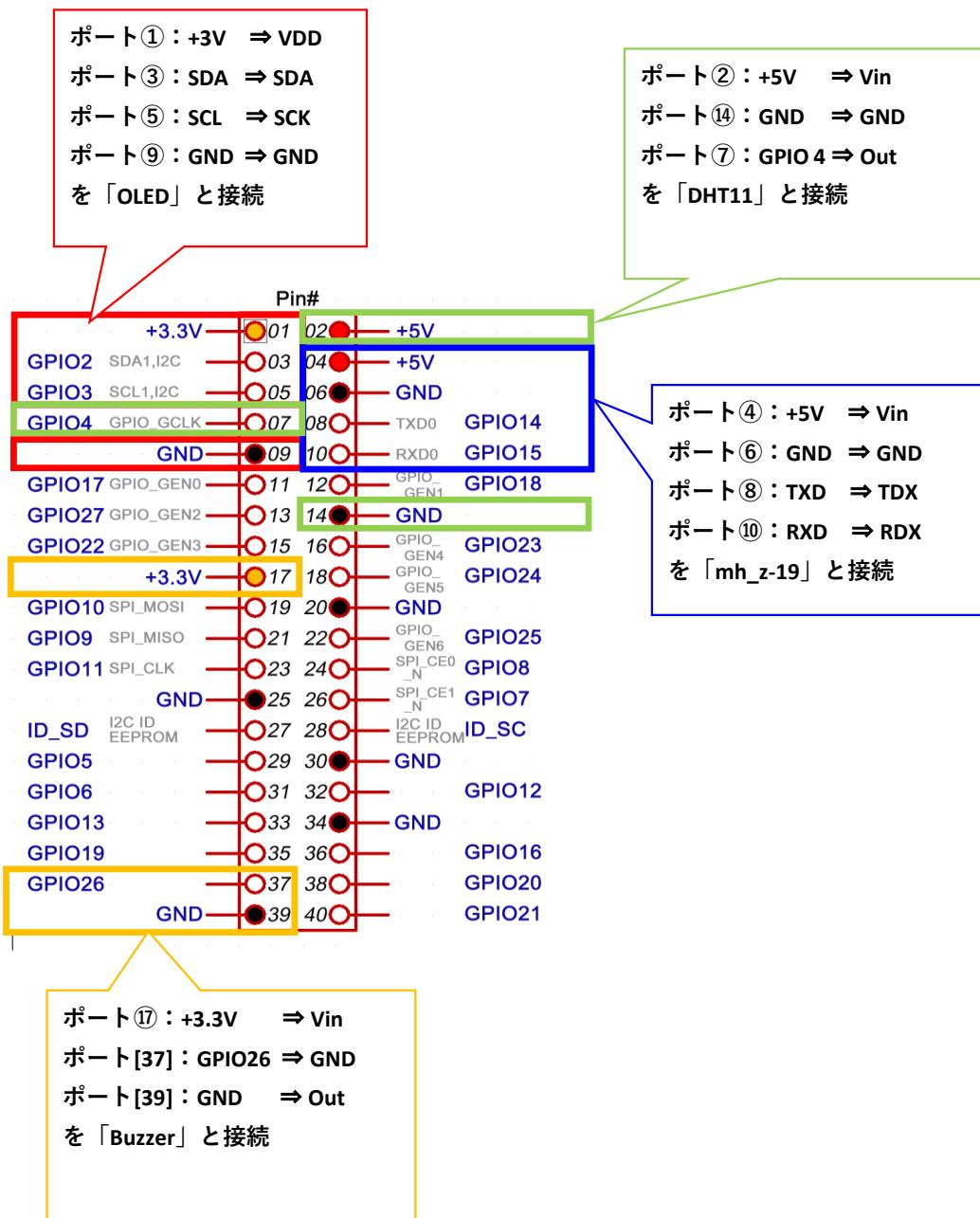
```
python3 DHT11_Read_pin7_GPIO4.py
```

1. DHT11とラズベリーパイのGPIOを接続

RPiポート②⑭⑦(+5V,GND,GPIO 4(GCLK)) \leftrightarrow DHT11(5V,GND,Out)

DHT11とラズパイとの接続





【8】アラームブザーの利用手順

(アクティブBuzzerをGPIO接続して測定結果を表示するまでの手順)

1. アクティブブザーとラズベリーパイのGPIOを接続

2. Pythonライブラリの追加と実行

3. 取得結果利用のライブラリの追加

pip3 install wiringpi

4. アクティブブザー動作用ファイルコピー

Buzzer_pin40_GPIO21_0602.py

5. アクティブブザー動作実行

python3 Buzzer_pin40_GPIO21_0602.py

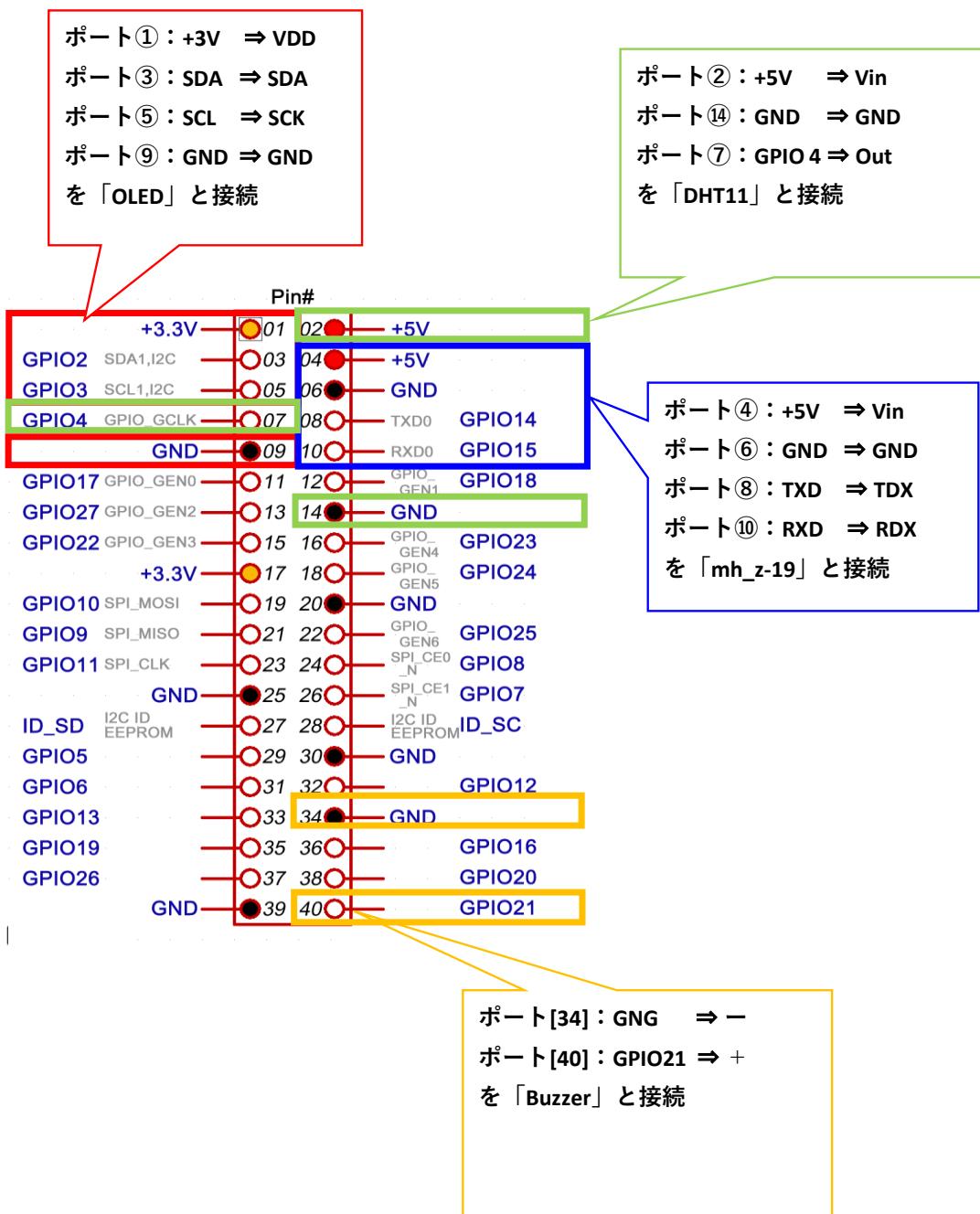
1. アクティブブザーとラズベリーパイのGPIOを接続

RPiポート[34][40](GND,GPIO 21↔Buzzer(−、+)

The screenshot shows a product page for a 12mm UDB-05LFPN electronic buzzer. The page includes the following details:

- Product Name:** 電子ブザー 12mm UDB-05LFPN
- Part Number:** [UDB-05LFPN]
- Stock Code:** P-09704
- Release Date:** 2016/05/23
- Manufacturer Category:** DB Products Limited.

The page also features a large image of the buzzer, a dimension diagram, and a list of related products. At the bottom, there is a product selector with the part number [P-09704] and a price of 1個 ¥80 (税込). The selector includes fields for '購入数量' (Purchase Quantity) with a value of 1, 'かごに入れる' (Add to Cart), and 'かごの中身を見る' (View Cart).



【9】OLEDにQRコード表示の利用手順

(OLEDをGPIO接続してQRコードを表示するまでの手順)

1. OLEDとラズベリーパイのGPIOを接続
 2. Pythonライブラリの追加と実行
 3. QRコードライブラリの追加

pip3 install qrcode

MkQr.py

python3 MkQr.py

- ## 5. OLEDにQRコード画像表示用pyをコピー

OLED image 0602.py

6 OLEDにQRコード画像表示実行

8. SEEDEDQR4 - Python3 OLED image

python3 OLED_image_0002.py http://raspberrypi.local:8080

4. QRコードPNG作成用pyをコピーして実行。
[URLをQRコード画像として保存するサンプル]

【URLをQRコード画像として保存するサンプル】

```
Python, Pillow, qrcodeでQRコード画像を生成、保存 (2021.05.04-22).txt[C:\DLF\Ra... - ファイル(E) 編集(E) 検索(S) ウィンドウ(W) その他(O) ヘルプ(H)
```

83 もろもろの設定をすると以下のようになる。↓

84 ↓

85 qr = qrcode.QRCode(↓

86 version=12,↓

87 error_correction=qrcode.constants.ERROR_CORRECT_H,↓

88 box_size=2,↓

89 border=8,↓

90)↓

91 qr.add_data('test text')↓

92 qr.make()↓

93 img = qr.make_image(fill_color="red", back_color="#23dda0")↓

94 img.save('data/dst/qrcode_test_2.png')↓

95 source: qrcode_test.py↓

96 python qrcode advance↓

97 ↓

98 以下、それぞれの引数について説明する。↓

99 ↓

100 ★【QRコードで「URL」を作成してPNGファイル出力】↓

101 ↓

102 import qrcode↓

103 qr = qrcode.QRCode(↓

104 version=1,↓

105 box_size=2,↓

106 border=1↓

107)↓

108 qr.add_data('https://naname-45.com')↓

109 img = qr.make_image()↓

110 img.save('qr_https_naname-45.com.png')↓

111 ↓

112 ↓

113 生成されるQRコードのバージョン (引数version) ↓

114 QRCodeクラスの引数versionで、生成されるQRコードのバージョンを設定できる。↓

115 ↓

116 QRコードの情報量とバージョンの関係についてはデンソーウェーブ公式の以下のページ↓

117 ↓

118 QRコードの情報量とバージョン | QRコードドットコム | 株式会社デンソーウェーブ↓

119 バージョンの値が大きくなるとQRコードに含まれるセル（四角い黒白の点）の数が多↓

120 ↓

121 引数versionを与えることで、最適なバージョン（最小限のサイズ）よりも大きいQRこ↓

122 ↓

「MkQr.py」

```
import qrcode

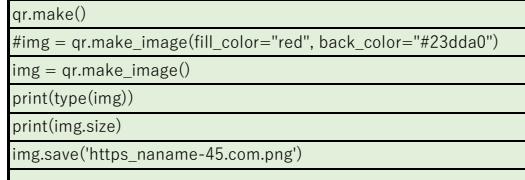
img = qrcode.make('https://naname-45.com/')

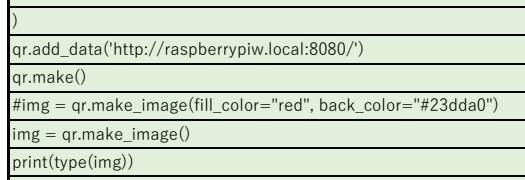
print(type(img))
print(img.size)
# <class 'qrcode.image.pil.PilImage'>
# (290, 290)

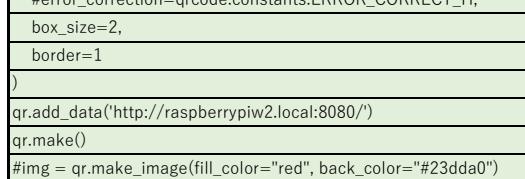
img.save('qrcode_test01_0529.bmp')

#02
qr = qrcode.QRCode()
qr.add_data('test text02')
qr.make()
img = qr.make_image()
print(type(img))
print(img.size)
img.save('qrcode_test02_0529.png')

#03
qr = qrcode.QRCode(
```

```
version=1,
#error_correction=qrcode.constants.ERROR_CORRECT_H,
box_size=2,
border=1
)
qr.add_data('https://naname-45.com/')
qr.make()

print(type(img))
print(img.size)
img.save('https_naname-45.com.png')

#04
qr = qrcode.QRCode(
    version=1,
    #error_correction=qrcode.constants.ERROR_CORRECT_H,
    box_size=2,
    border=1
)
qr.add_data('http://raspberrypiw.local:8080/')
qr.make()

print(type(img))
print(img.size)
img.save('http_raspberrypiw.local.png')

#05
qr = qrcode.QRCode(
    version=1,
    #error_correction=qrcode.constants.ERROR_CORRECT_H,
    box_size=2,
    border=1
)
qr.add_data('http://raspberrypiw2.local:8080/')
qr.make()

print(type(img))
print(img.size)
img.save('http_raspberrypiw2.local.png')
```