はじめにお読みください「RPi-GP70 について」

本製品を正しく安全にお使いいただくための取り扱い方法、使用上の注意等について説明するものです。
ご使用の前に必ずお読みください。

機能概要

RPi-GP70 は、Raspberry Pi の GPIO 40Pin（I2C）に接続する絶縁型のRS485/RS422シリアル拡張ボードです。
本製品には以下の機能があります。

- Raspberry Pi GPIO40Pinコネクタに装着する絶縁型RS485/RS422シリアルボード
- RS485/RS422シリアルポートは２ポート搭載
- RS485/422Aに対応
- シリアルポートコネクタは、5ピンネジ式脱着型端子台
- I2C-シリアル２ポートコントローラとしてSC16IS752を使用
- 送受信各64ByteのFIFOバッファ搭載
- シリアルポートの絶縁(絶縁耐圧2.5kV)は、コントローラとドライバ/レシーバ間。　さらに、2ポート間（CN1とCN2間）も絶縁。
- Pythonモジュール「PyModbus」を使い、Modbus方式のシリアル通信を行うポートとして使用可能

ご使用の前に

内容物の確認

パッケージの中に下記のものがすべて揃っているかをご確認ください。
万一不足がありましたら、お手数ですが弊社サポートセンターまたは販売店までご連絡をお願いいたします。

- RPi-GP70
- 5ピン電線側ネジ式端子台 x 2（基板へ装着済み）
- GPIO 40PIN ピンヘッダー x1
- スペーサー（固定用 M2.6）x4
- ネジ（固定用 M2.6）x8
- 設定用ショートプラグ 4個(基板上のJP1,JP2,JP3,JP4に装着)
- ピン番号ラベル
- 保証書

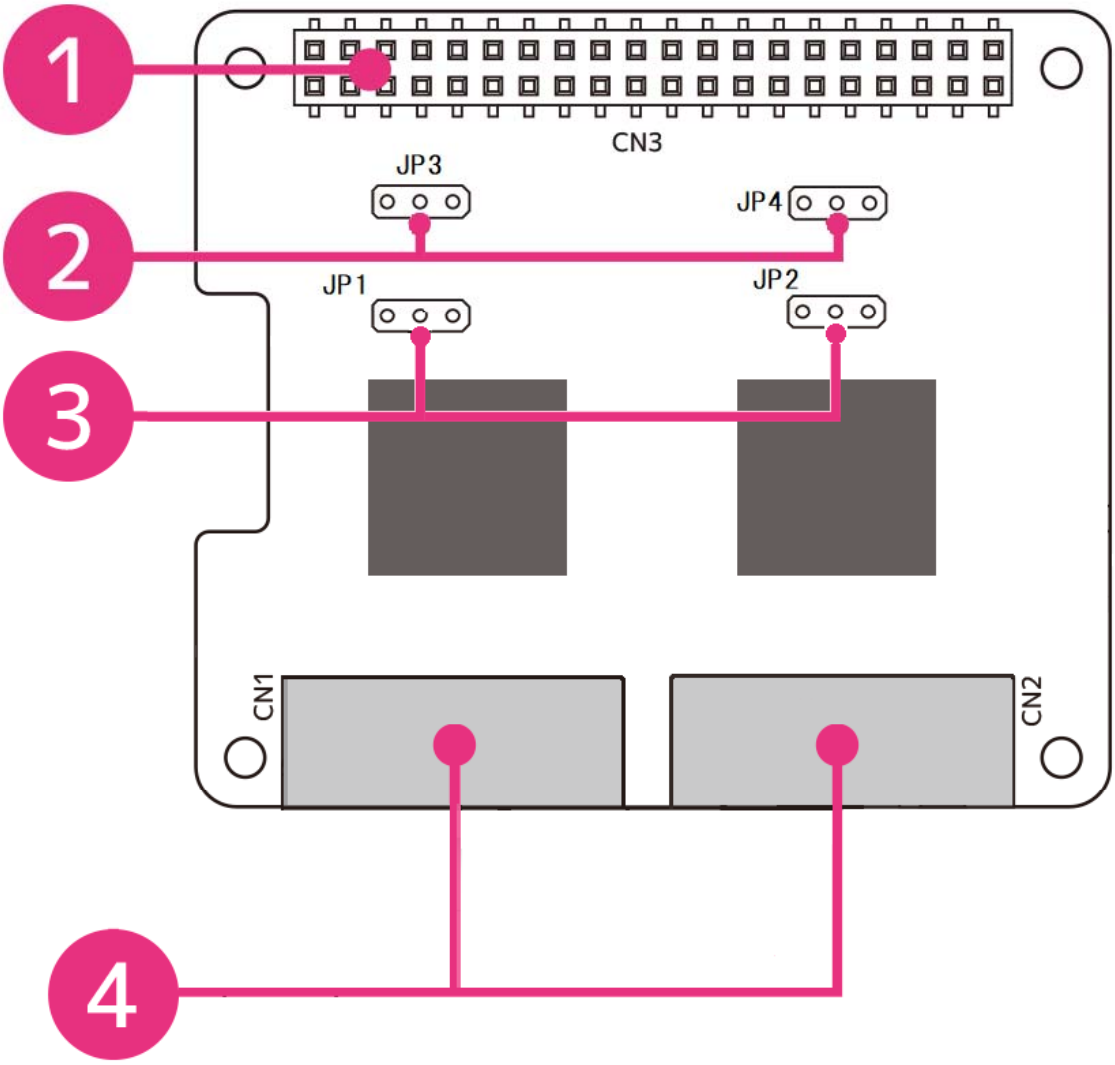
各項目については以下のリンクをご参照ください

Content

RPi-GP70 各部名称と説明

1. 基板構成

製品基板の各部名称は以下のとおりです。



No	名称	機能
1	GPIO 40PINコネクタ	Raspberry Pi GPIO
2	JP3/JP4 ジャンパピン	シリアルポート0,1 受信制御信号設定
3	JP1/JP2 ジャンパピン	シリアルポート0,1 終端抵抗設定

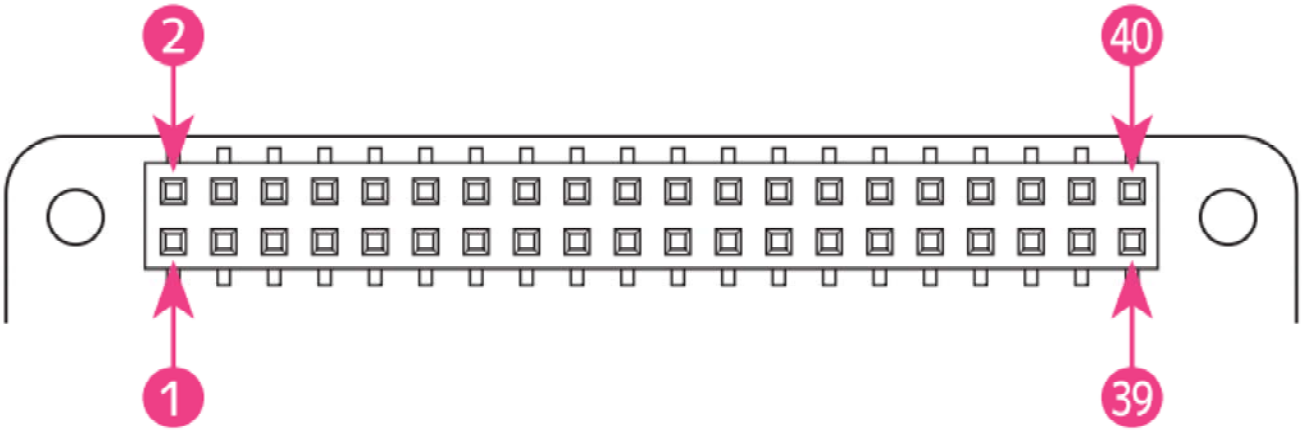
No	名称	機能
4	CN1/CN2 OMRON XW4A 5ピンネジ式脱着型端子台	シリアルポート0,1

※CN1,CN2(電線側ソケット XW4B-05B1)の適合電線はAWG28～AWG16, ストリップ長は7mmです。

2. 各端子・コネクタについて

2-1. GPIO 40PINコネクタ

GPIO 40 PIN の配列および使用ピンは以下のとおりです。



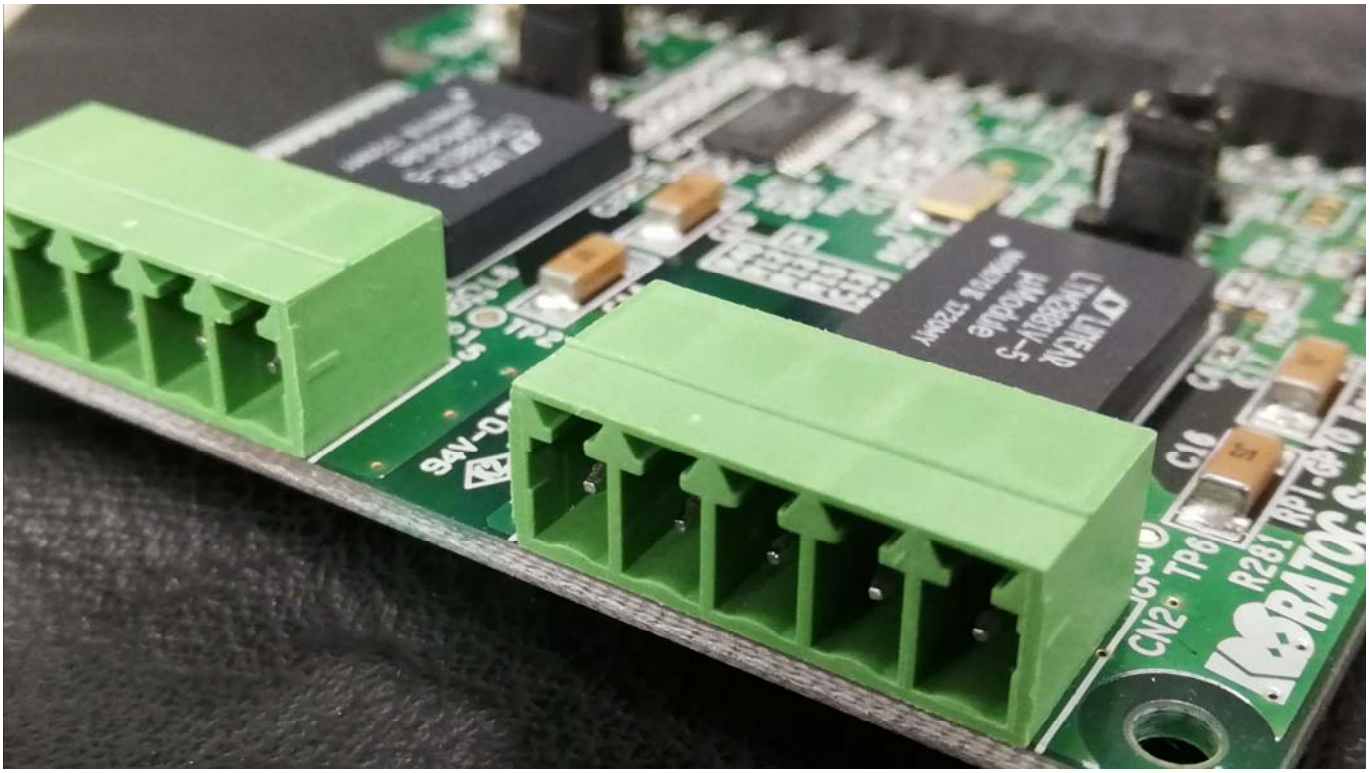
40PIN GPIOのピン配列と説明

PIN#	信号名	説明	PIN#	信号名	説明
1	3.3V	3.3V電源	2	5V	5V電源
3	I2C SDA1/GPIO 2	I2C SDA1	4	5V	5V電源
5	I2C SCL1/GPIO 3	I2C SCL1	6	GND	GND
7	GPIO 4	(未使用)	8	GPIO 14	(未使用)
9	GND	GND	10	GPIO 15	(未使用)
11	IRQ/GPIO 17	IRQ入力	12	GPIO 18	(未使用)
13	GPIO 27	絶縁電源制御	14	GND	GND
15	GPIO 22	(未使用)	16	GPIO 23	(未使用)
17	3.3V	3.3V電源	18	GPIO 24	(未使用)
19	SPI0 MOSI/GPIO 10	(未使用)	20	GND	GND
21	SPI0 MISO/GPIO 9	(未使用)	22	GPIO 25	(未使用)
23	SPI0 SCLK/GPIO 11	(未使用)	24	SPI CE0/GPIO 8	(未使用)
25	GND	GND	26	SPI CE1/GPIO 7	(未使用)

PIN#	信号名	説明	PIN#	信号名	説明
27	I2C SDA0/GPIO 0	HAT_ID読み込み用I2C	28	I2C SCL0/GPIO 1	HAT_ID読み込み用I2C
29	GPIO 5	(未使用)	30	GND	GND
31	GPIO 6	(未使用)	32	GPIO 12	(未使用)
33	GPIO 13	(未使用)	34	GND	GND
35	GPIO 19	(未使用)	36	GPIO 16	(未使用)
37	GPIO 26	(未使用)	38	GPIO 20	(未使用)
39	GND	GND	40	GPIO 21	(未使用)

2-2. シリアルポート0,1 [5ピン端子台]

シリアルポート0[CN1], ポート1[CN2]は、5ピン端子台を使用しています。



ボード上のジャンパ[JP1/JP2] で終端抵抗の有効無効を設定します。
ボード上のジャンパ[JP3/JP4] で受信制御信号を設定します。
5ピン端子台は、独自の端子配列となります。
詳細についてはRPi-GP70の設定と装着を参照してください。

Raspberry Pi OSの設定

Raspberry Pi用のOSである Raspberry Pi OSの設定について説明します。
Raspberry Pi本体は'Raspberry Pi4 ModelB'、 Raspberry Pi Imagerは、'Raspberry Pi Imager 1.7.3'、 OSは'Raspberry Pi OS Version 2022-09-22'で説明します。

Raspberry Pi OSのインストール

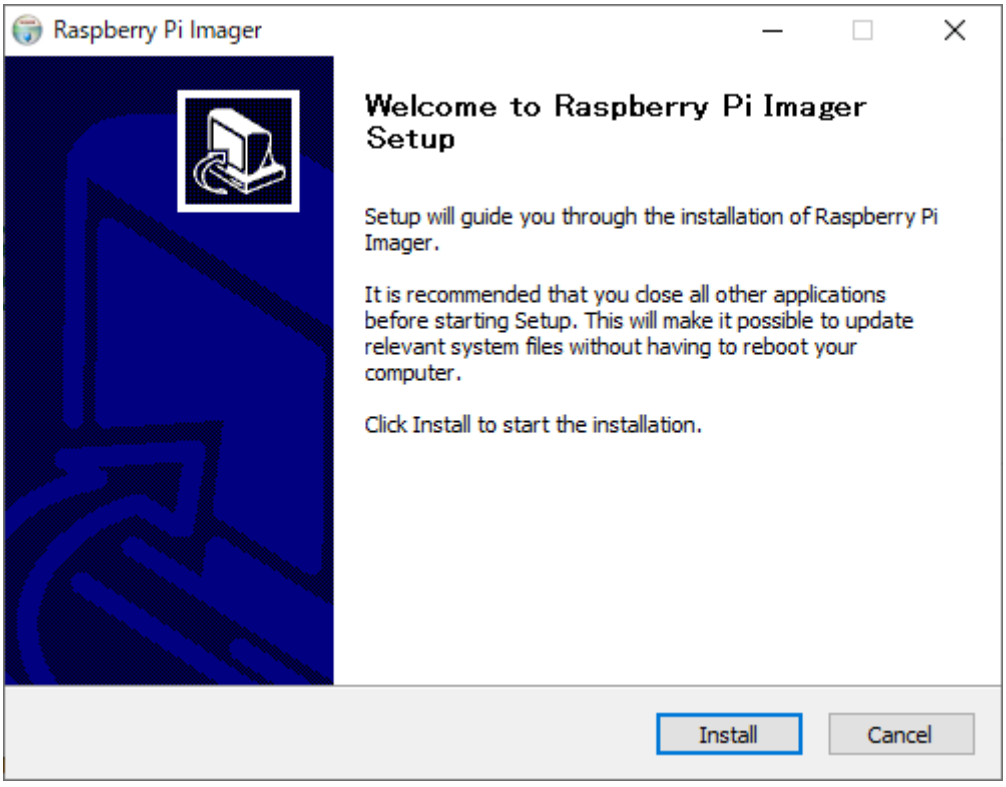
1. Class10のmicroSD(8～32G)を用意します。

*64GB*以上の*SD*カードの場合、*exFAT*でフォーマットされます。 *Raspberry Pi OS*は*exFAT*に対応していませんので、別のツールを使って*FAT16*または*FAT32*でフォーマットする必要があります。

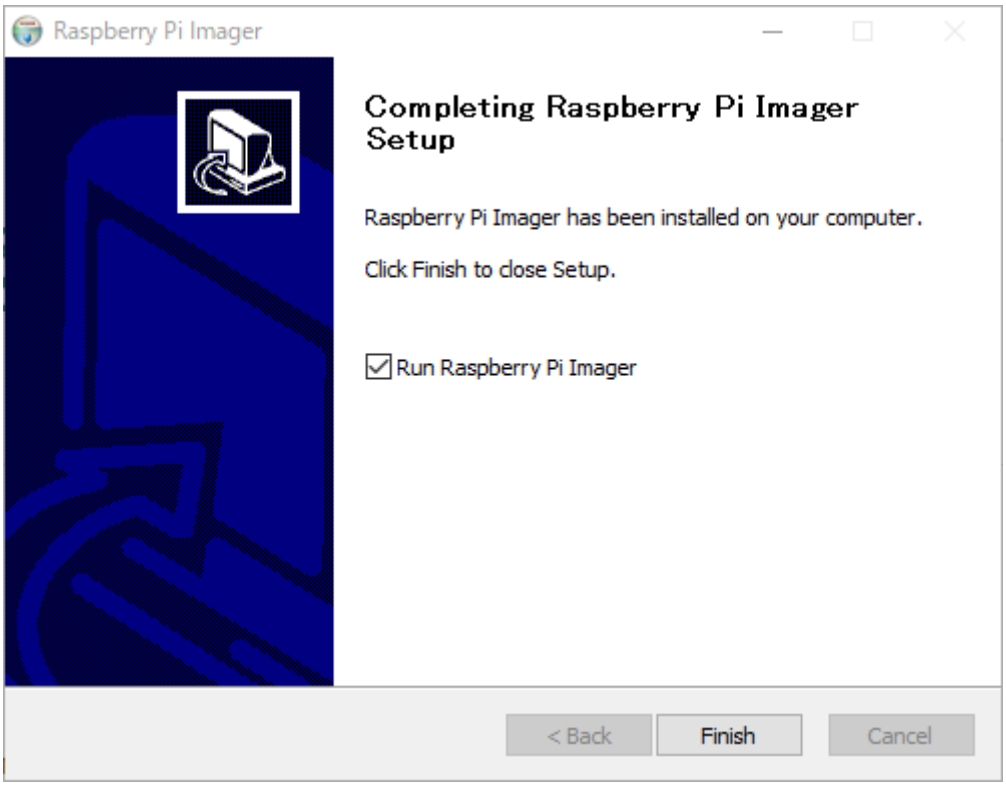
2. Raspberry財団公式ホームページ(https://raspberrypi.org/software/)でRaspberry Pi Imagerをダウンロードしてインストールします。

2-1. ダウンロードしたファイルをダブルクリックします。

2-2. [Install] をクリックします。



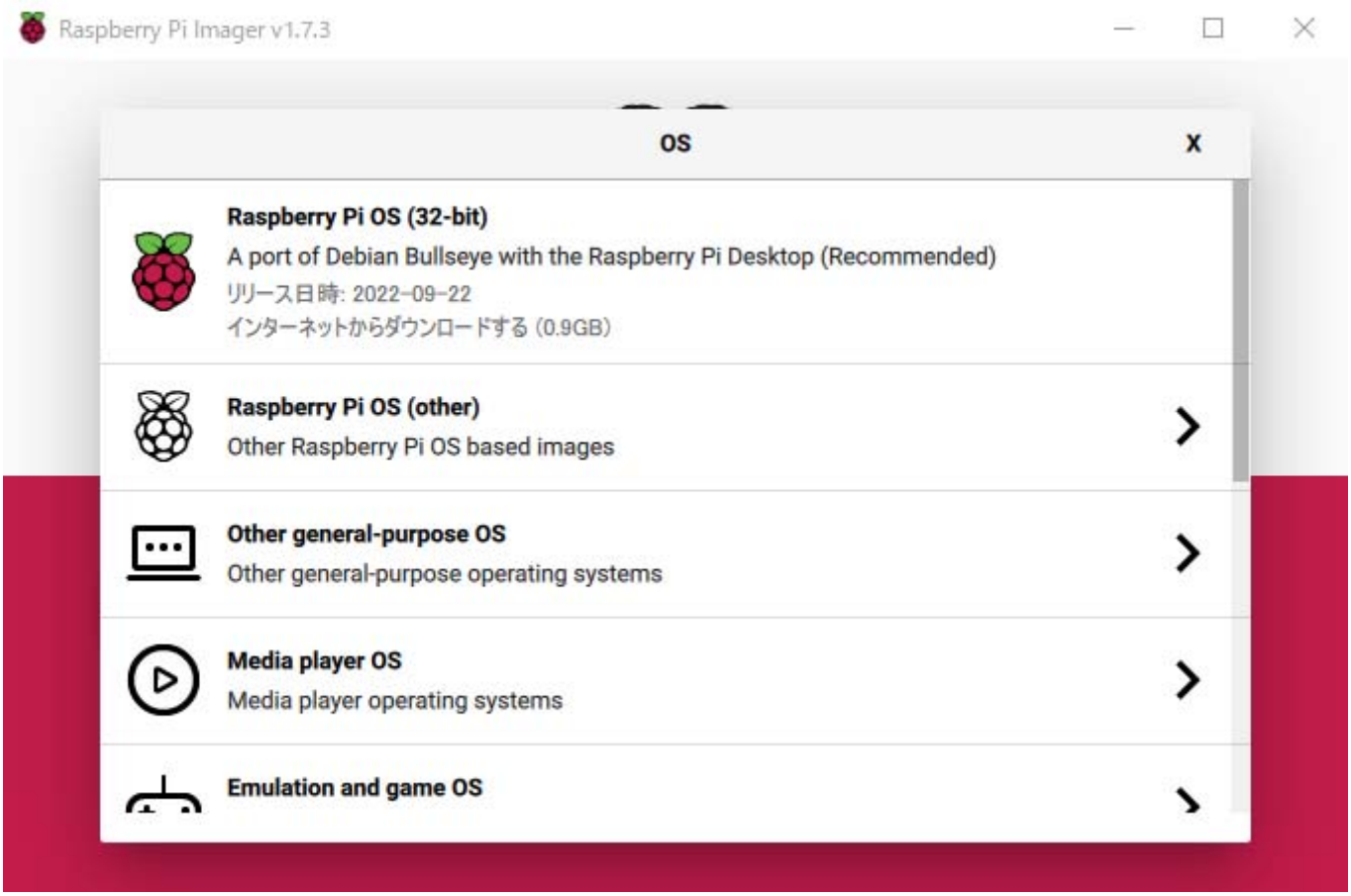
2-3. 終了画面で[Finish]をクリックします。



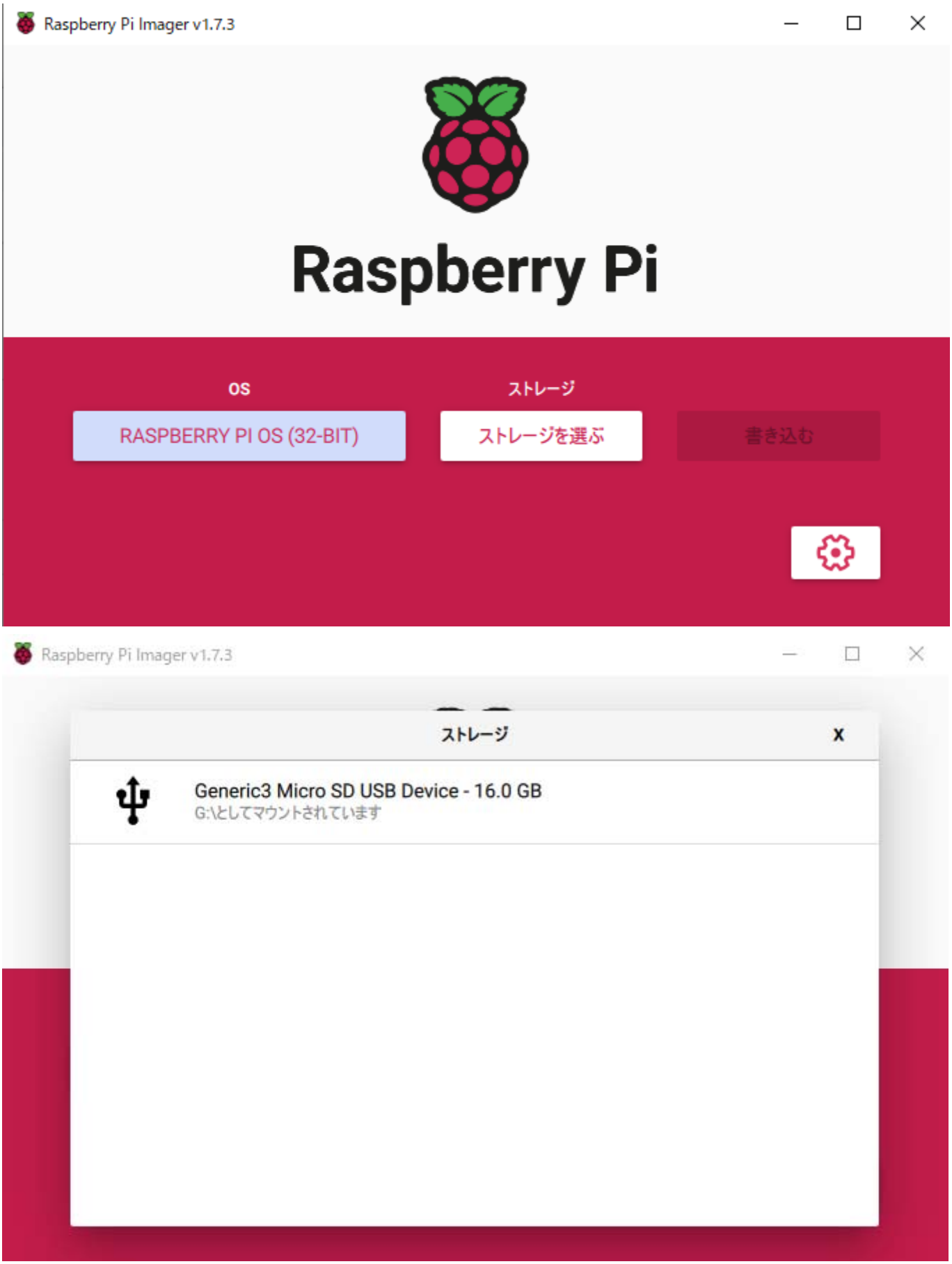
3. Raspberry Pi Imager を使って Raspberry Pi OS のSDカードを作成します。

3-1. Raspberry Pi Imagerが起動すると以下の画面が表示されます。

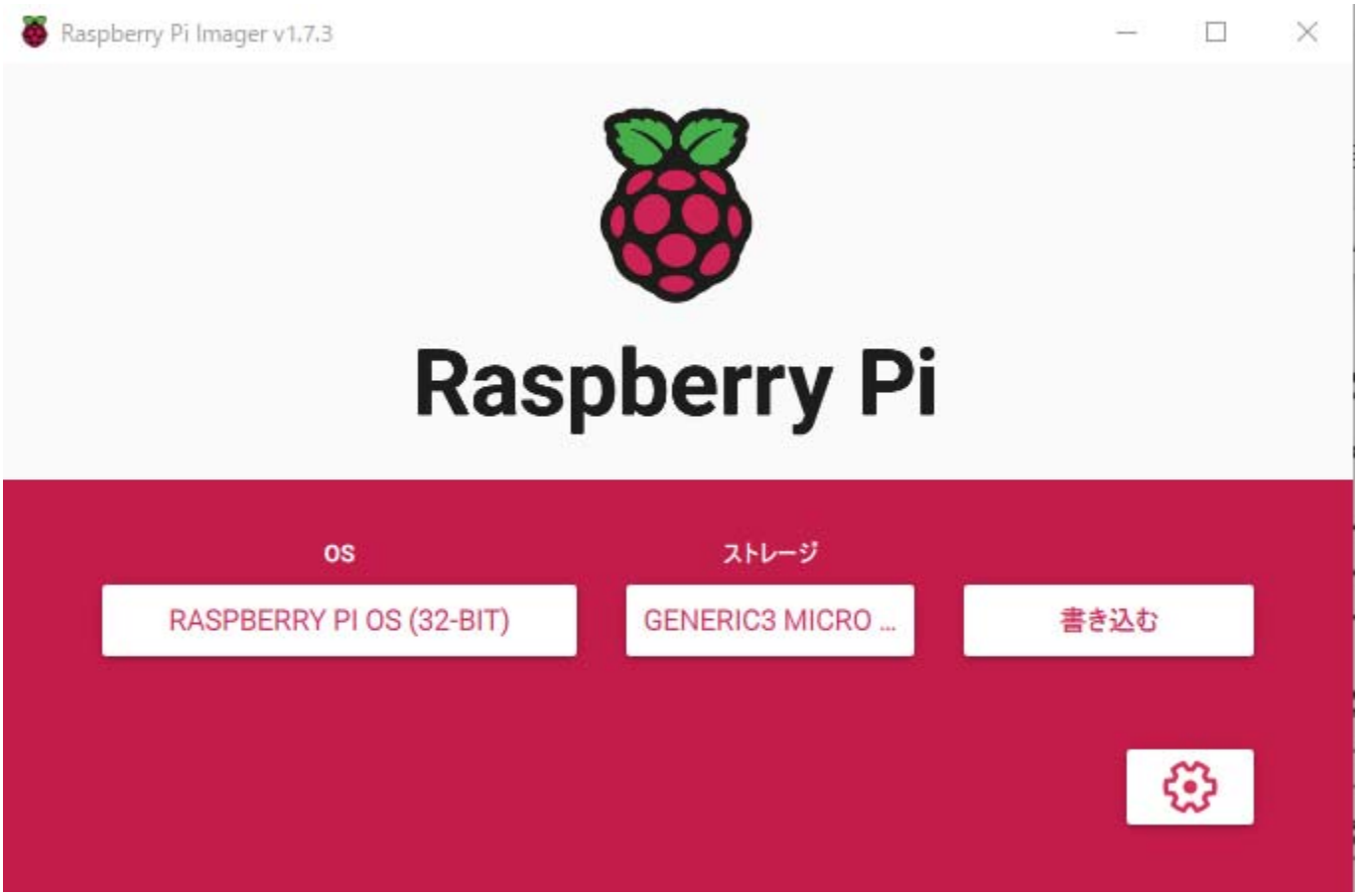
3-2. [OSを選ぶ]をクリックして[Raspberry Pi OS (32-bit)]を選択します。



3-3. [ストレージを選ぶ]をクリックして、書き込み先のデバイス（SD Card）を選択します。



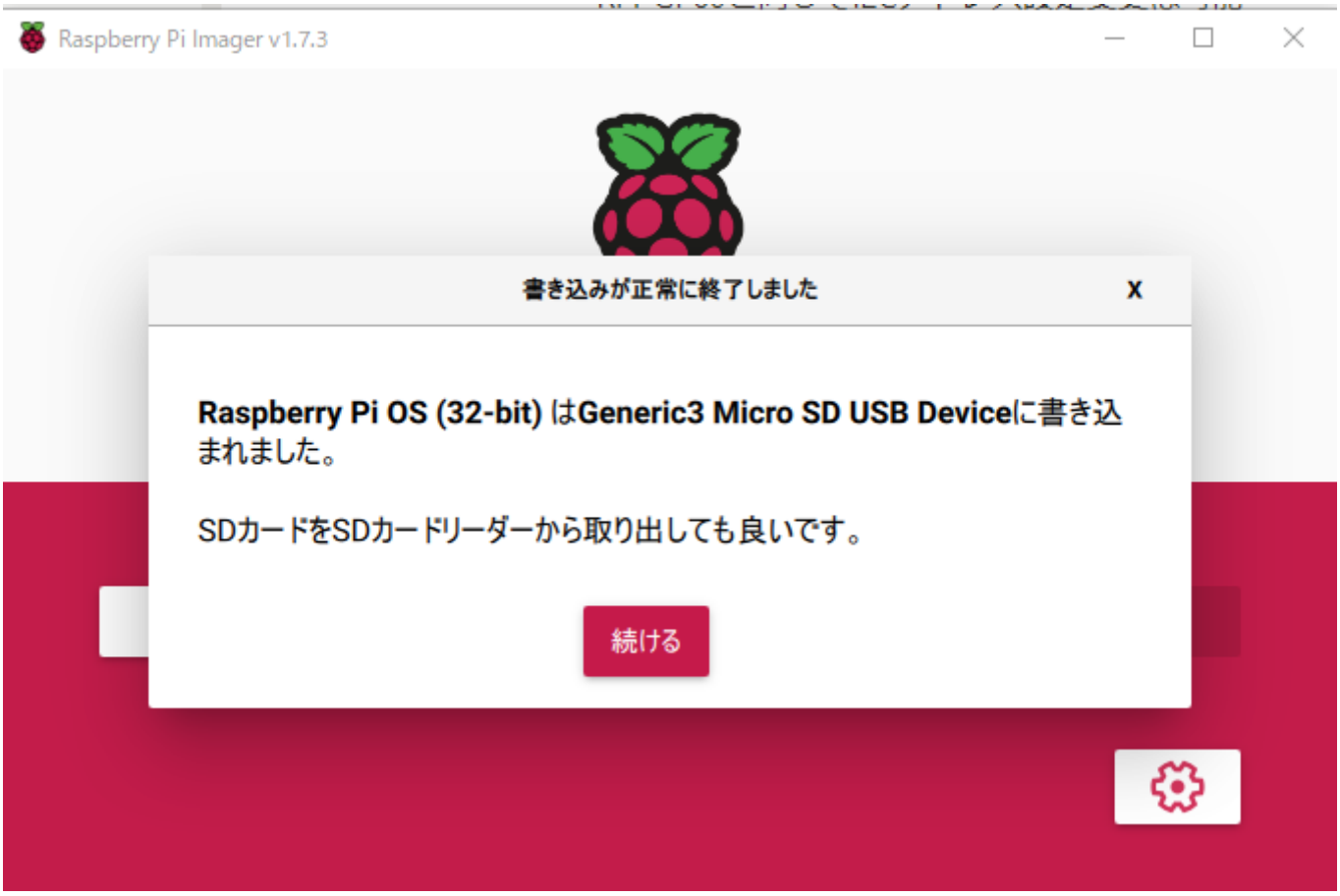
3-4. [書き込む] をクリックします。



3-5. 確認ダイアログ画面で [はい]を選択して、SD Cardへの書き込みを開始します。

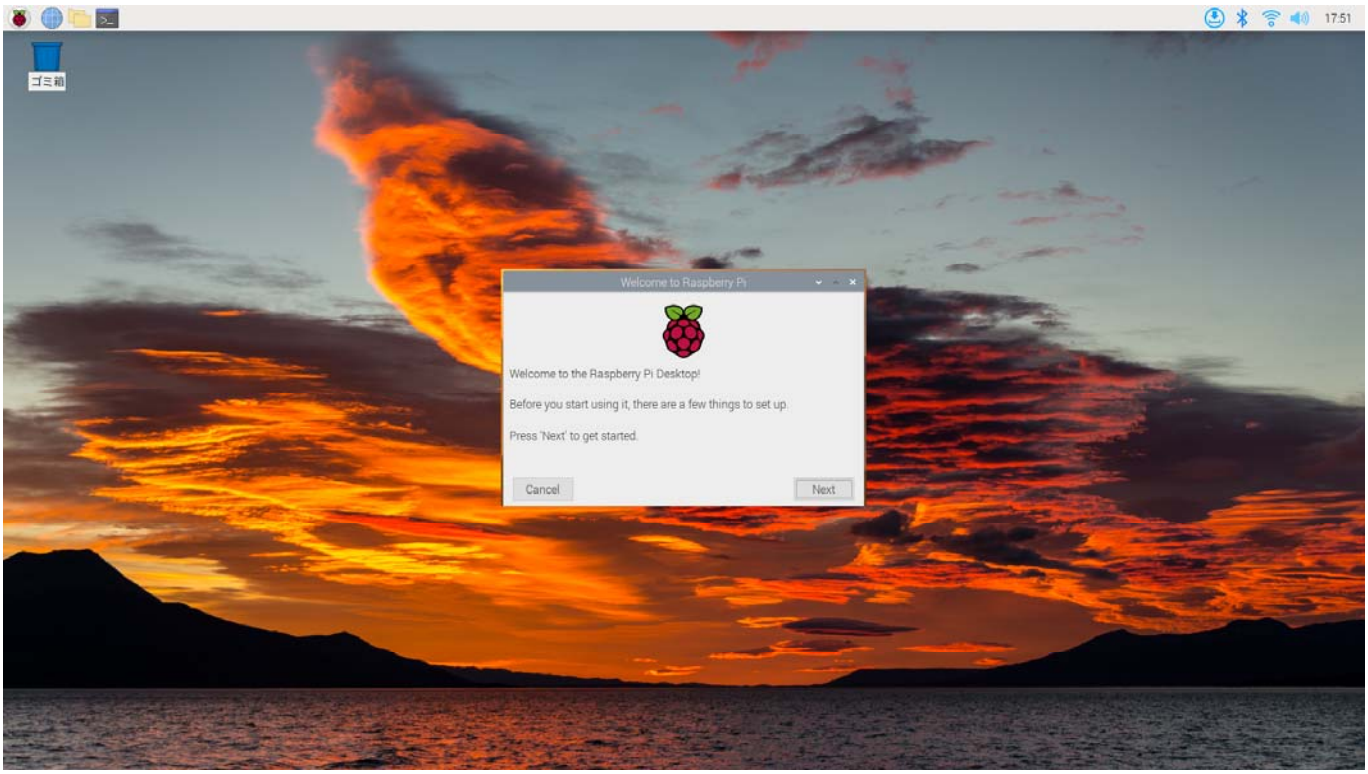
3-6. 途中、書き込みの進捗状況が表示されます。

3-7. 書き込みが終わると、以下の完了画面が表示されます。　　[続ける] をクリックして、microSDカードを取り外します。

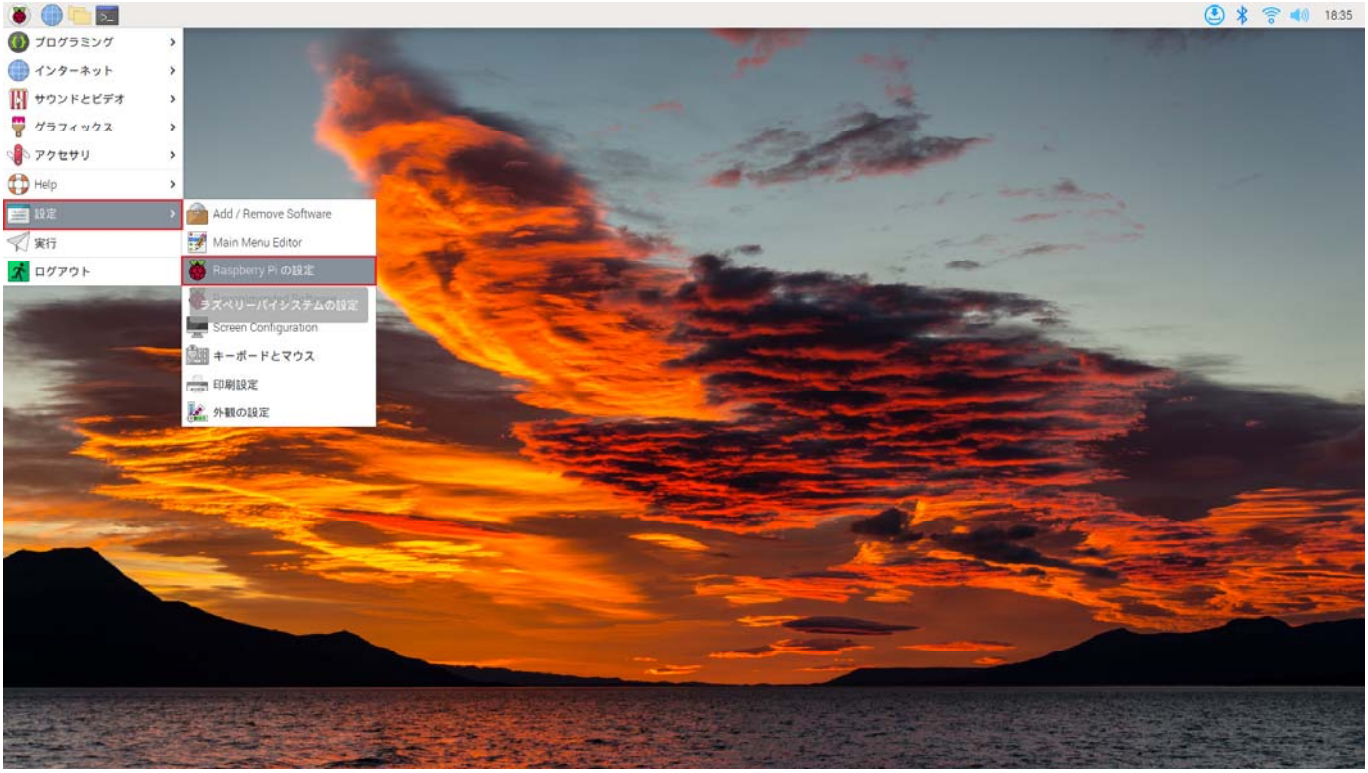


4. OSの起動とI2Cの有効設定

4-1. microSDカードをRaspberry Pi基板に接続し起動します。



4-2. [設定]-[ラズベリーパイシステムの設定]をクリックします。



4-3. [インターフェース]で"I2C"を有効にします。

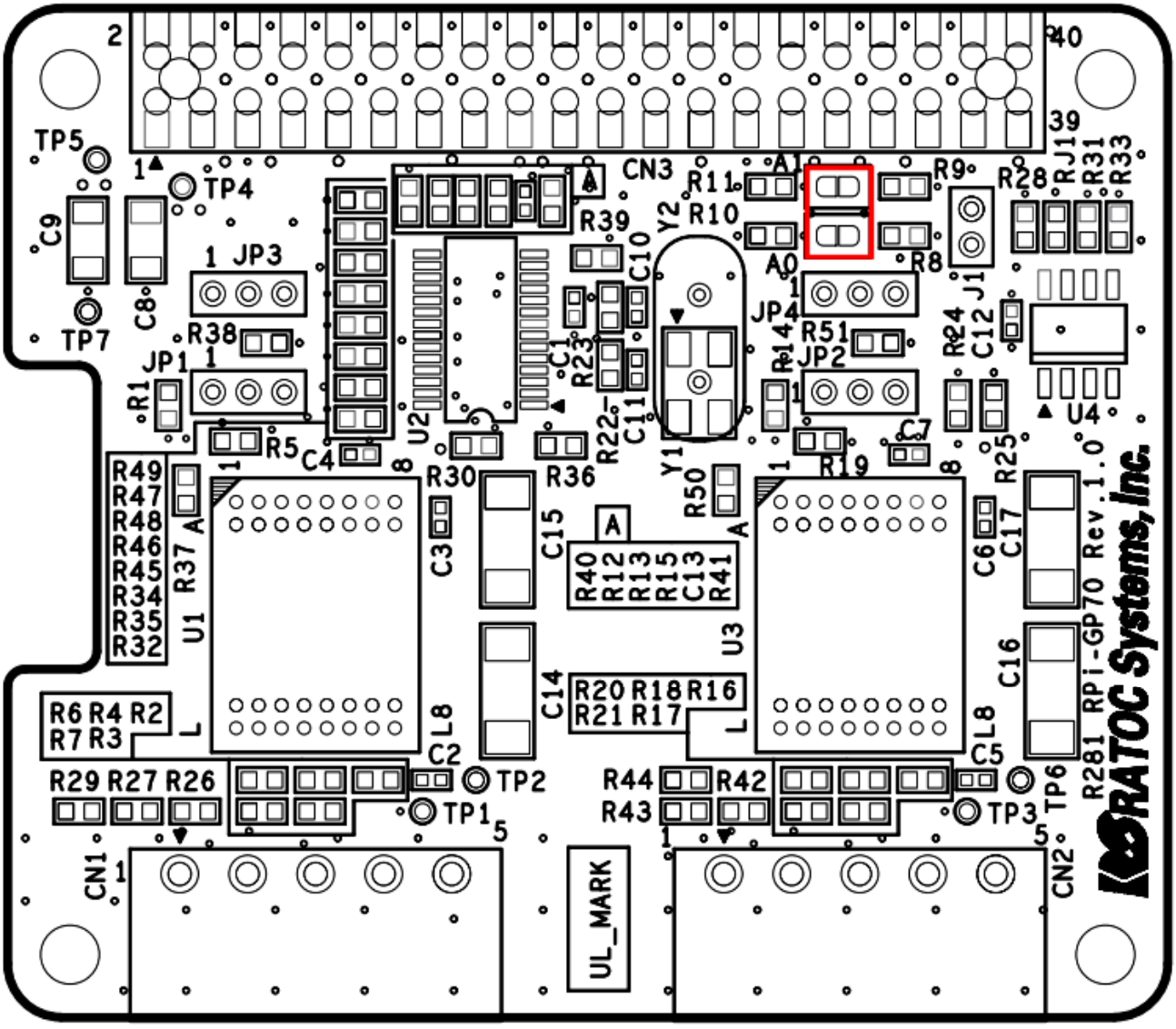
設定を反映させるため、OSを再起動すれば、Raspberry Pi OS のインストールと設定は完了です。

RPi-GP70の設定と装着

各種設定と本製品をRaspberry Pi GPIO 40PINに接続する方法を説明します。

1. 各種ハードウェア設定

1-1. I2Cアドレスの設定(A0/A1)

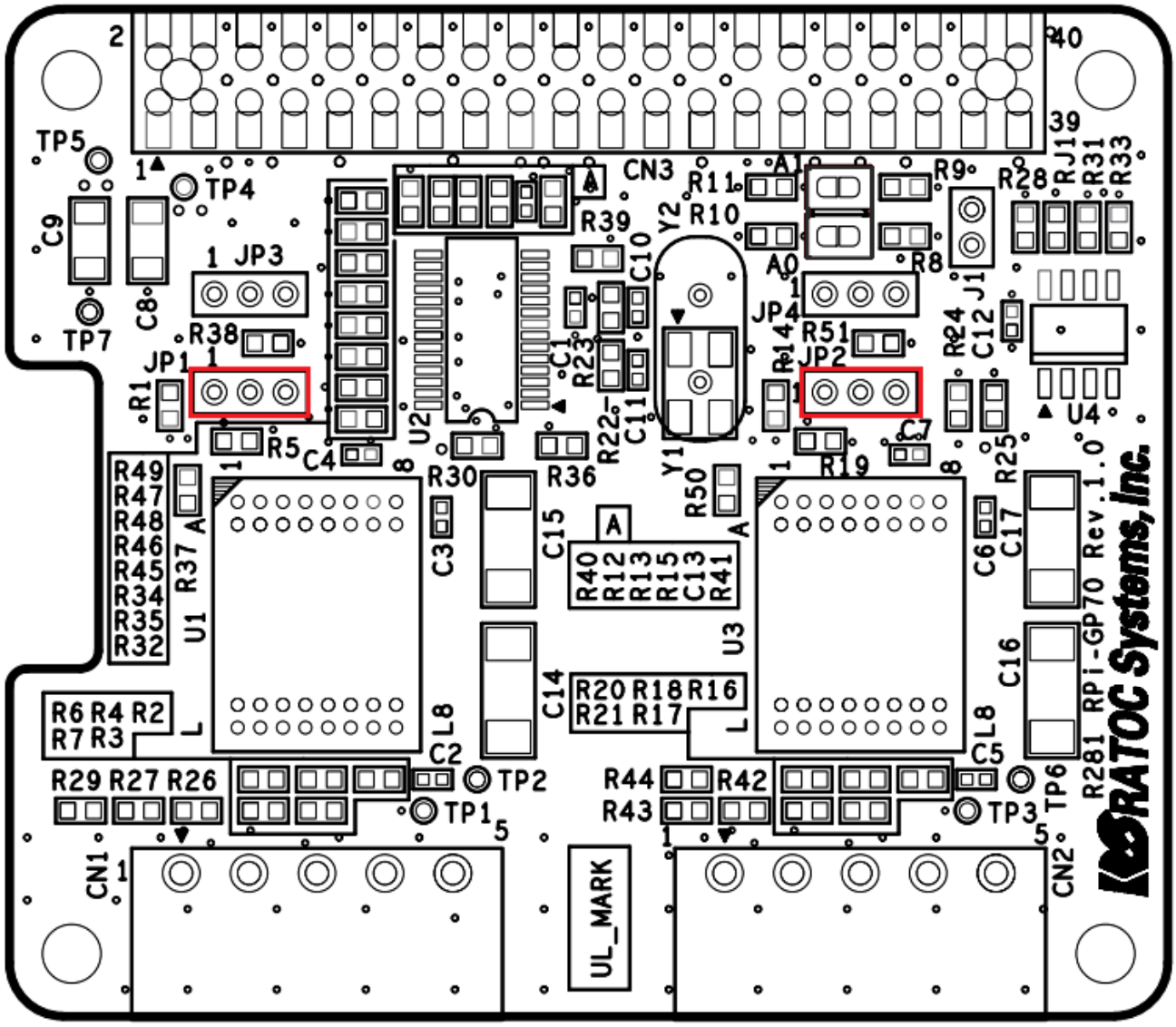


シリアルコントローラSC16IS752のI2Cアドレスを、半田ジャンパ(A0, A1)のオープン/ショートによって設定変更できます。
※初期設定は0x4D(A0～A1オープン)です。

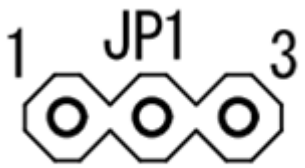
A1	A0	I2Cアドレス	
オープン	オープン	0x4D	※初期設定
オープン	ショート	0x4C	
ショート	オープン	0x49	

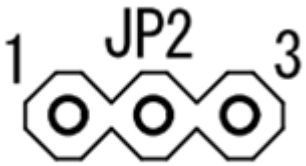
A1	A0	I2Cアドレス
ショート	ショート	0x48

1-2. 終端抵抗の設定(JP1/JP2)



RS485/422の差動レシーバに120Ω終端抵抗の無効/有効を設定します。 基板上のジャンパ設定ピンJP1でPort0(CN1),JP2でPort1(CN2)のポート毎に設定をします。 出荷時設定は2－3ショート(終端抵抗無効)です。

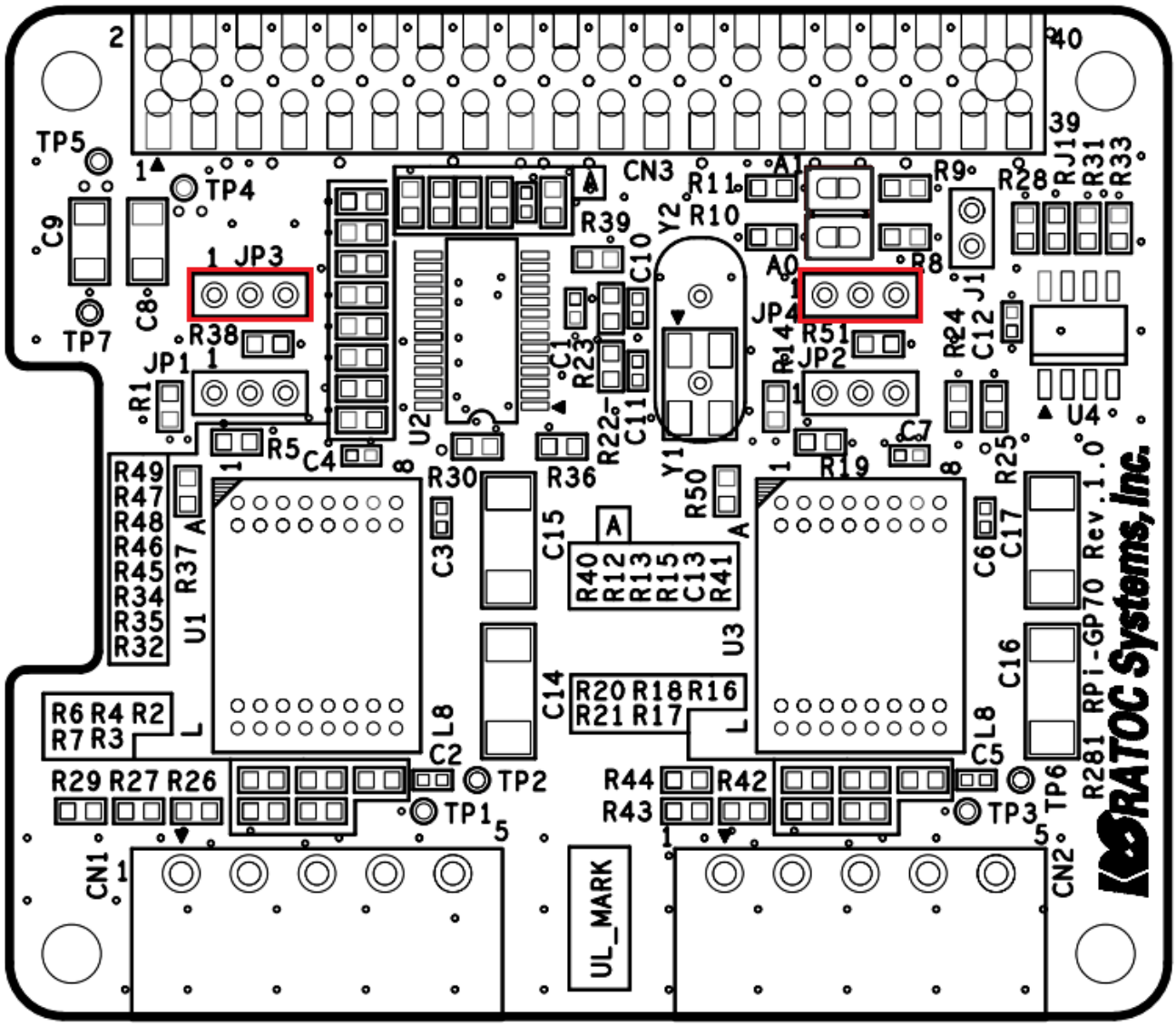




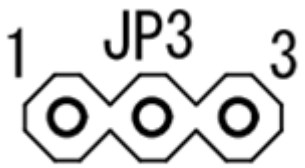
JP1/JP2 ピン設定	機能・説明
1-2 ショート	終端抵抗有効※
2-3 ショート	終端抵抗無効

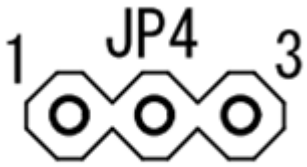
※終端抵抗は絶縁電源がONの時に有効となります。

1-3. 受信制御信号の設定(JP3/JP4)



基板上のジャンパ設定ピンJP3でPort0(CN1),JP4でPort1(CN2)のRS485/RS422の受信制御信号を設定します。
出荷時設定はオープン(常に受信有効)です。

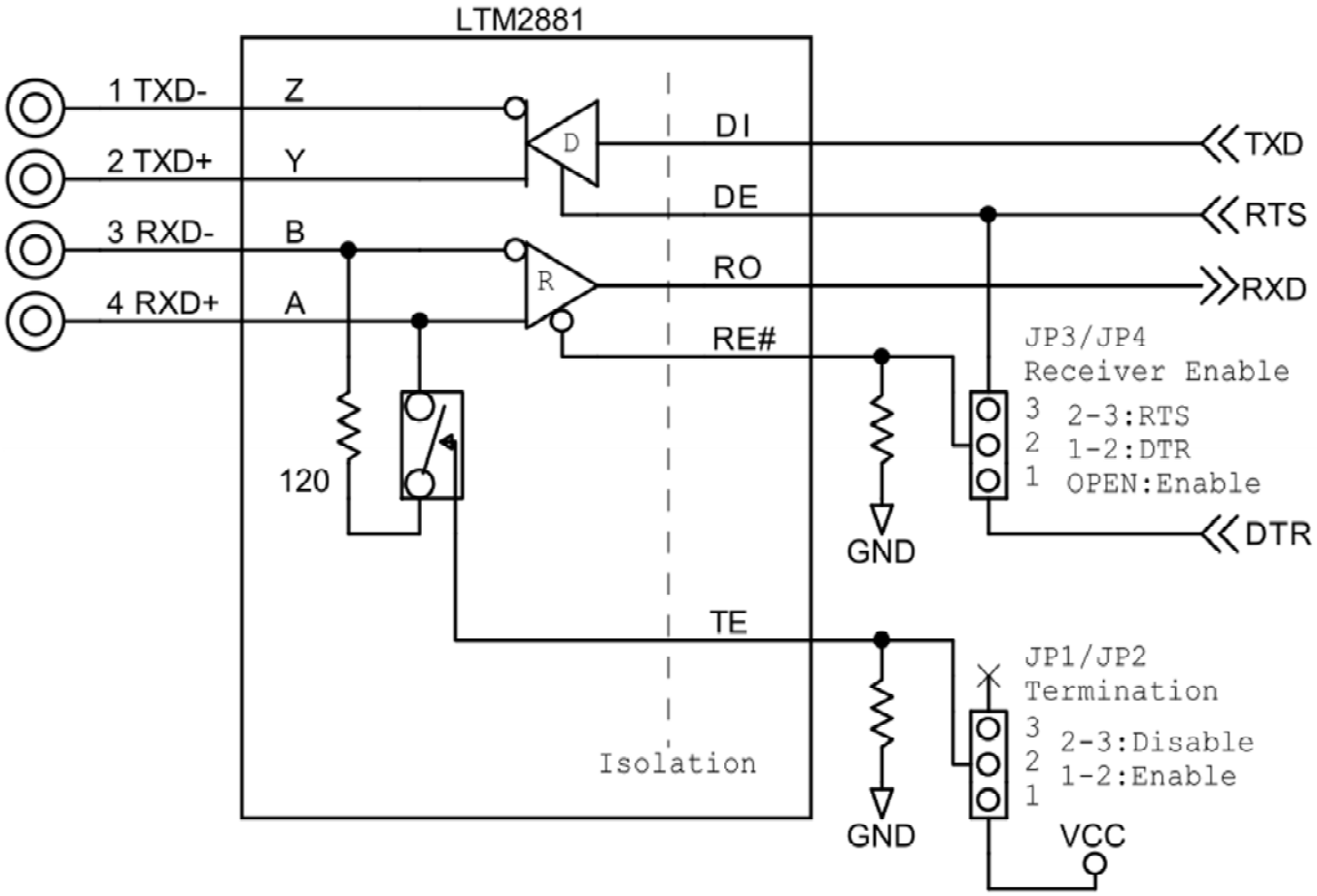




JP3/JP4 ピン設定	機能・説明
1-2 ショート	DTR制御（DTR OFFで受信有効）
2-3 ショート	RTS制御（RTS OFFで受信有効）
オープン	常に受信有効

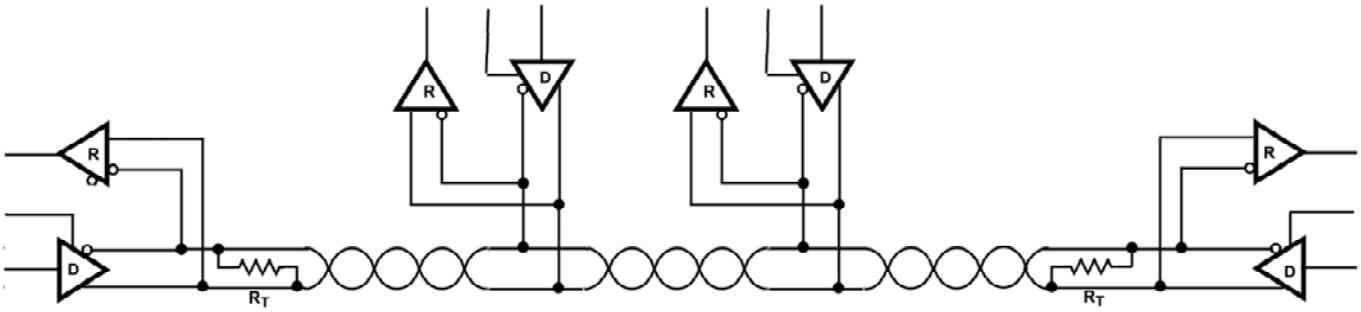
1-4. RS485/422ドライバ/レシーバ

RS485/422のドライバ/レシーバ部の回路構成です。



1-5. RS485半二重接続例

デージーチェーン接続された複数のドライバとレシーバで、双方向でのデータ伝送が可能です。（マルチポイント方式）
最大32台のデバイスが接続できます。
配線長が一番長くなる両端の終端抵抗を有効にし、それ以外の終端抵抗は無効にしてください。
また、電線にツイストペアケーブルを使用することで、電磁誘導などによるコモンモードノイズを差動レシーバによって効率的に除去できます。



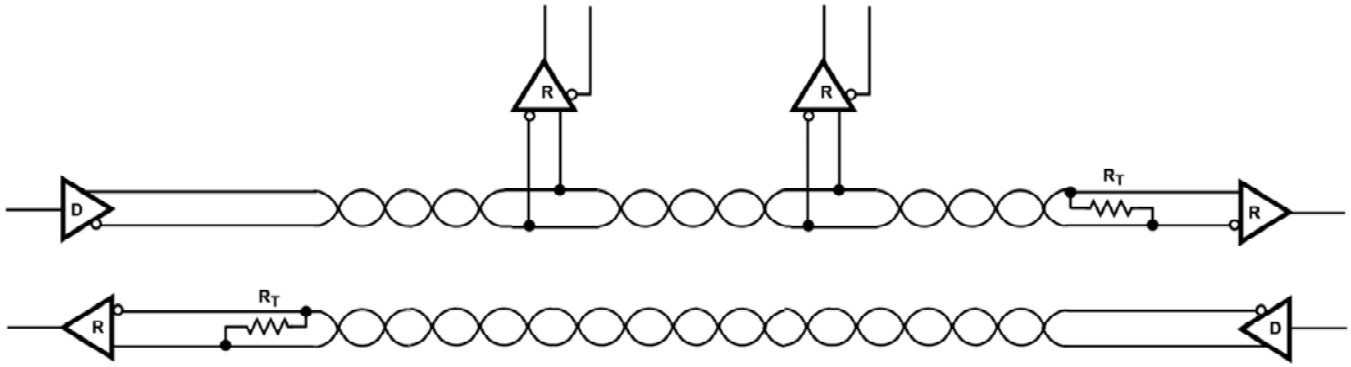
RS-485の半二重の送信・受信

機能	RTS/DTR制御
送信	RTS ON

機能	RTS/DTR制御
受信	JP3/JP4の設定参照

1-6. RS485/422全二重接続例

一つのマスタに対して複数のスレーブを接続できます。ドライバ１台に最大10台のレシーバが接続できます。（マルチドロップ方式）
配線長が一番長くなるレシーバの終端抵抗を有効にして、それ以外のレシーバの終端抵抗は無効にしてください。
また、電線にツイストペアケーブルを使用することで、電磁誘導などによるコモンモードノイズを差動レシーバによって効率的に除去できます。



RS-485の全二重の送信・受信

機能	RTS/DTR制御
送信	RTS ON
受信	JP3/JP4の設定参照

1-7. RS-485/422 各規格の比較

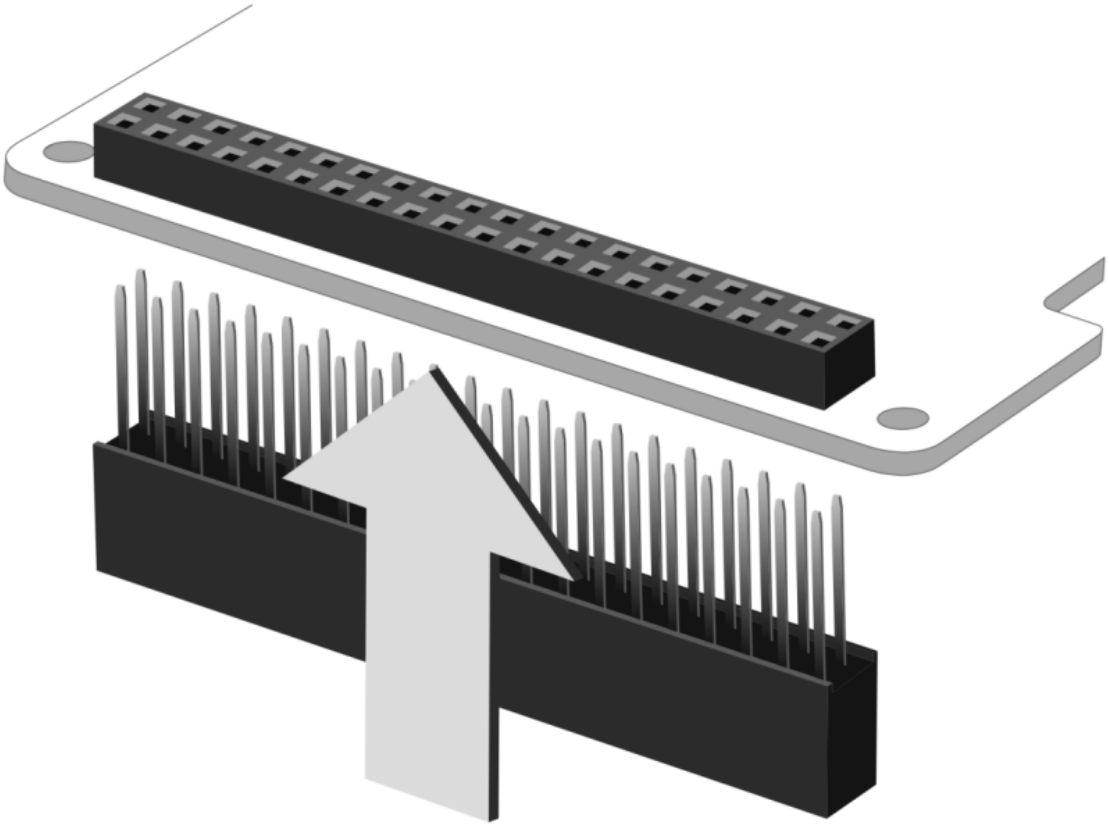
規格上は以下のように定められていますが、実際の使用環境では最大値を下回る場合があります。

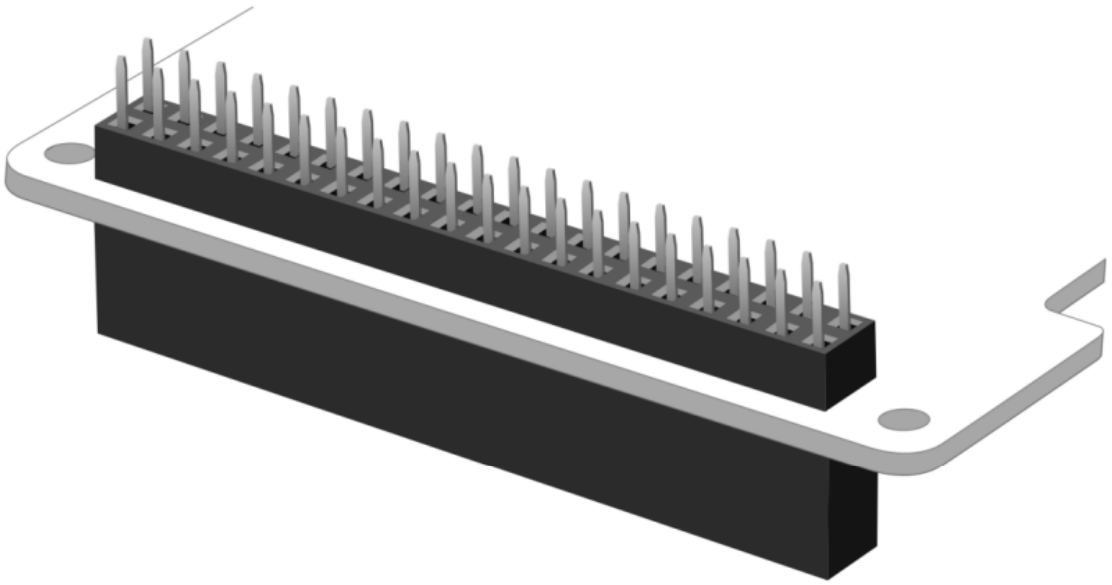
項目	RS-485	RS-422A
準拠規格	TIA/EIA-485-A	TIA/EIA-422-B
端子配列規格	独自	独自
動作モード	平衡型 (差動)	平衡型 (差動)
最大接続可能台数	32ドライバ、32レシーバ 半二重でのマルチポイント方式に対応	1マスタドライバに10スレーブレシーバ 全二重でのマルチドロップ方式
最大ケーブル長(規格値)	1200m ツイストペアケーブルを推奨	1200m ツイストペアケーブルを推奨
最大伝送速度(規格値)	10m - 35Mbit/s 1200m – 100kbit/s	1.2m – 10Mbit/s 1200m – 100kbit/s

項目	RS-485	RS-422A
最大伝送速度(実力値)	920kbps	920kbps
特徴	長距離のN対N通信 半二重/全二重	長距離の１マスタ対Nスレーブ通信 全二重

2. 本体の組み立て

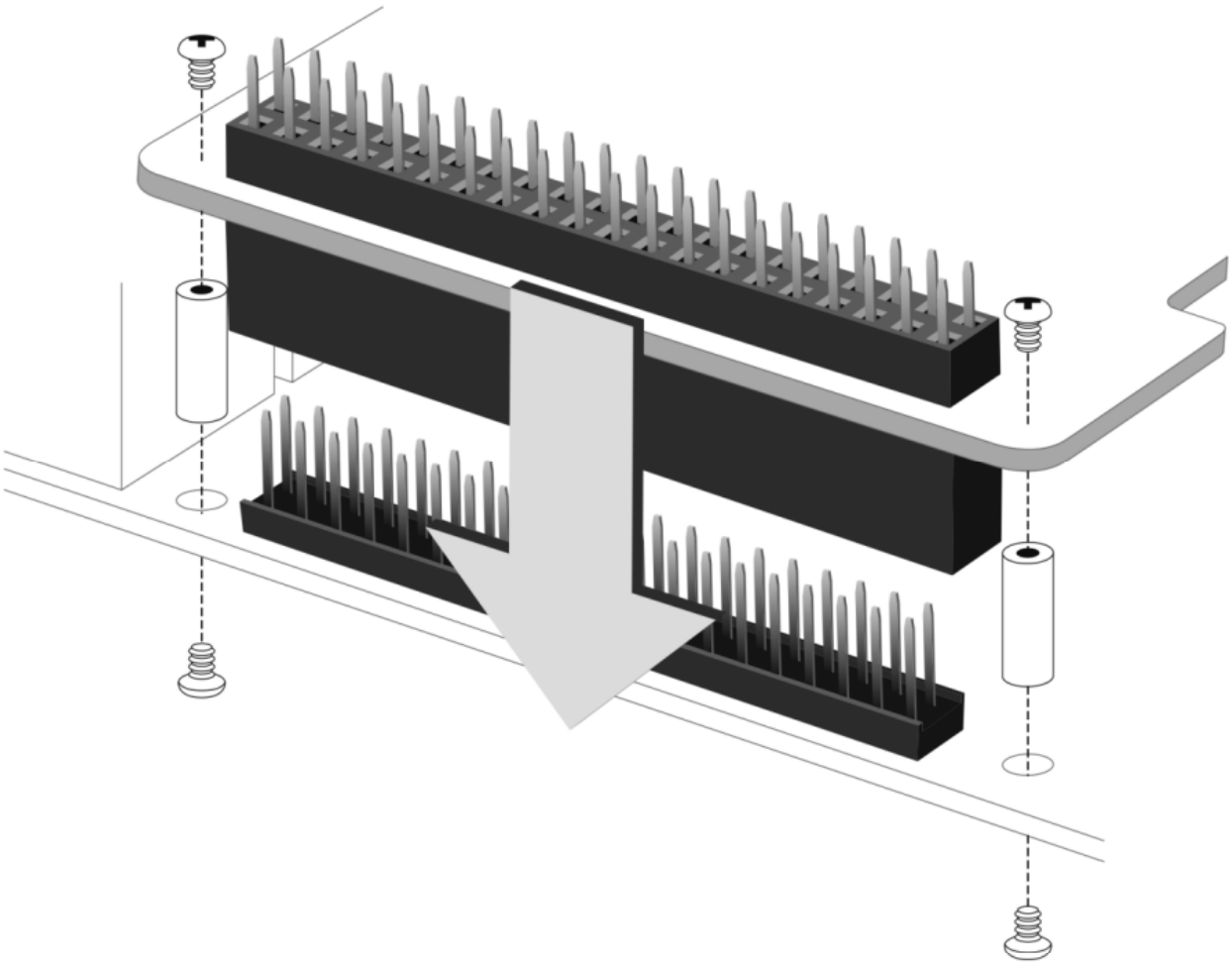
製品付属の40PIN ピンヘッダーを本製品の底面より垂直に装着します。
※40PINのピンヘッダーの先端は尖っていますので、怪我には十分ご注意ください。





3. Raspberry Piボードとの接続

Raspberry Pi 本体のGPIO ピンヘッダと本製品を接続します。
スペーサー（付属）を本製品本体のネジ穴にあわせ、スペーサーを付 属のネジ（4 本）で固定し、そのまま RaspberryPi のGPIO ピンへ垂 直に差し込みます。
スペーサーがRaspberryPi 基板に合わされば接続は完了です。
あとはRaspberryPi の背面より付属のネジ（4 本）を使用し、スペーサーを固定します。
※反対側にも同じようにスペーサーとネジを使用し本体を固定してください。



4. シリアルドライバ設定

SC16IS752のI2CドライバはRaspberry-piに標準で用意されています。 以下の手順で登録することで、2つのシリアルポートがttyとして認識されます。

SC16IS752のI2Cドライバ登録と確認手順

1. config.txtをnanoで編集する

```
$ sudo nano /boot/config.txt
```

2. 次の2行を最終行へ追加する

```
gpio=27=op,dh
dtoverlay=sc16is752-i2c,int_pin=17,addr=0x4d,xtal=14745600
```

「1-1. I2Cアドレスの設定」でアドレスを変更している場合は、[addr=0x4d]を適宜書き換えてください。

3. CTRL+Oで書き込み、CTRL+Xで終了する

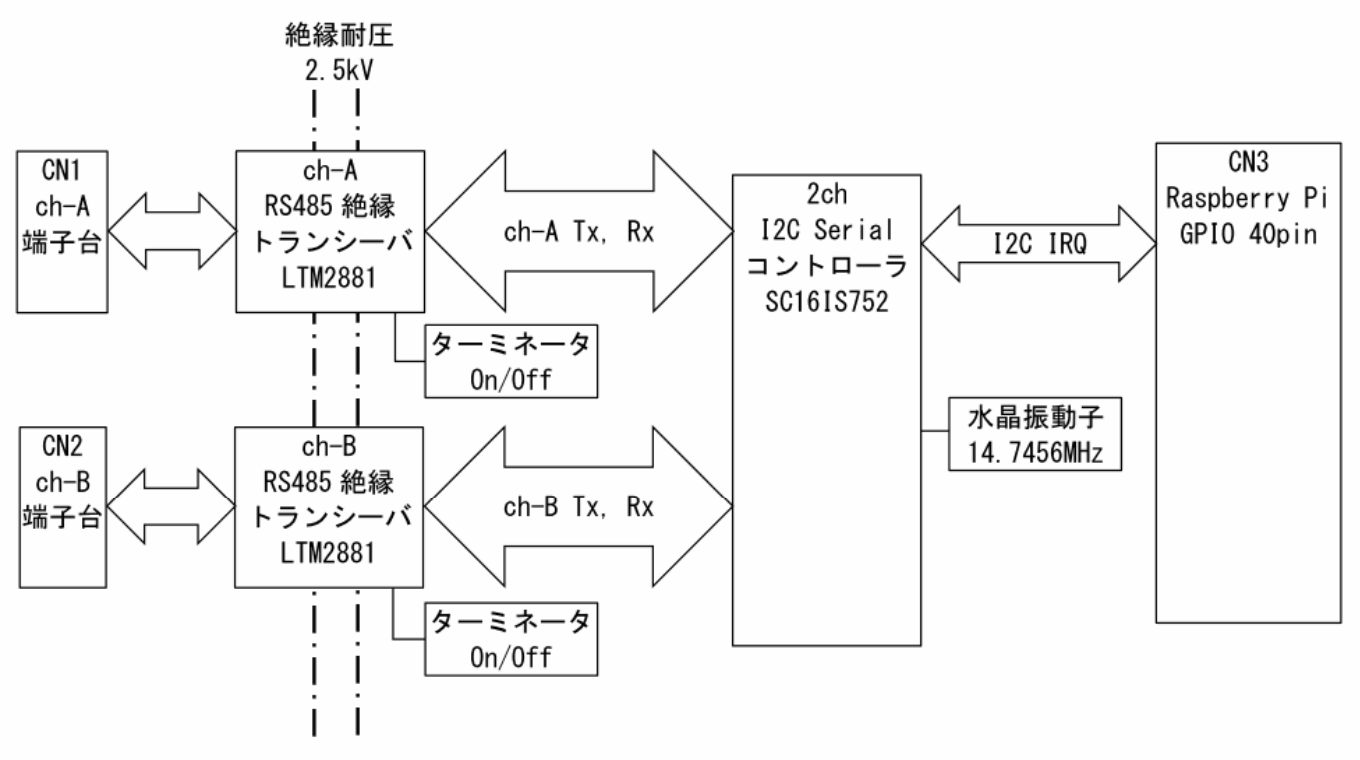
4. システムを再起動する

5. 再起動後、デバイスツリーに以下のttyポート[/ttySC0],[/ttySC1]が追加されていることを確認する

```
$ ls /dev/ttySC*
/dev/ttySC0  /dev/ttySC1
```

RPi-GP70の機能と説明

各インターフェースについて説明します。ブロック図は以下の通りです。



1. インターフェース

1-1. Raspberry Pi GPIO 40pin

RPi-GP70を制御するために、GPIO 40pinの下記の信号を使用します。

制御信号

PIN#	名称	機能説明
3	I2C SDA1	シリアルコントローラ用I2C
5	I2C SCL1	シリアルコントローラ用I2C
11	GPIO17	シリアルコントローラ割り込み要求 1:なし / 0:あり
13	GPIO27	絶縁電源制御出力 1:ON / 0:OFF
27	GPIO0/ID_SD	HAT_ID読み込み用I2C
28	GPIO1/ID_SC	HAT_ID読み込み用I2C

電源端子

PIN#	説明

PIN#	説明
1pin 17pin	3.3V
2pin 4pin	5V
6pin 9pin 14pin 20pin 25pin 30pin 34pin 39pin	GND

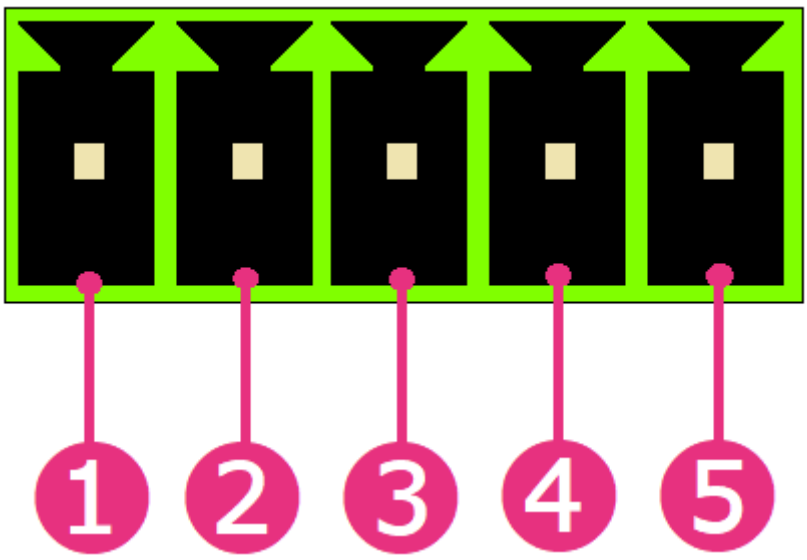
1-2. I2Cアドレス

シリアルコントローラを制御するためのI2Cアドレスは以下のとおりです。

２ポートシリアルコントローラ(SC16IS752)
基板上の半田ジャンパでシリアルコントローラ(SC16IS752)のI2Cアドレス設定変更が可能です。
出荷時のI2Cアドレス(7bit)は0x4Dに設定されています。
設定変更方法についてはRPi-GP70の設定と装着を参照してください。

1-3. ポート0,1 シリアルコネクタ

ポート0,1 シリアルコネクタの端子配列



ピン番号	RS485/422 信号名	I/O	説明
1	TXD-	O	送信データ差動－側出力
2	TXD+	O	送信データ差動＋側出力

ピン番号	RS485/422 信号名	I/O	説明
3	RXD-	I	受信データ差動－側入力
4	RXD+	I	受信データ差動＋側入力
5	GND	-	信号用接地

2. シリアルコントローラ

シリアルコントローラとして、NXP社SC16IS752を使用しています。
I2Cアドレス(7bit)の初期値は0x4Dで、基板上のハンダジャンパで設定変更が可能です。
設定方法の詳細についてはRPi-GP70の設定と装着を参照してください。

2-1. SC16IS752レジスタマップ

SC16IS752のレジスタマップは以下のとおりです。各レジスタの機能詳細については、NXP社のSC16IS752データシートを参照してください。
ただし、SC16IS752のシリアルドライバがTTYとしてカーネルから制御されている場合は、ユーザープログラムはレジスタに対してI2Cコマンドでのアクセスができません。
pythonの場合はpyserialモジュールを使用することで、ドライバ経由での制御が使用可能です。
pyserialモジュールを使用したシリアル通信サンプルプログラムも参照願います。

Adrs	name	Read mode	Write mode
0x00	RHR/THR	Receive Holding Register (RHR)	Transmit Holding Register (THR)
0x01	IER	Interrupt Enable Register (IER)	Interrupt Enable Register
0x02	IIR/FCR	Interrupt Identification Register (IIR)	FIFO Control Register (FCR)
0x03	LCR	Line Control Register (LCR)	Line Control Register
0x04	MCR	Modem Control Register (MCR)	Modem Control Register
0x05	LSR	Line Status Register (LSR)	n/a
0x06	MSR	Modem Status Register (MSR)	n/a
0x07	SPR	Scratchpad Register (SPR)	Scratchpad Register
0x06	TCR	Transmission Control Register (TCR)	Transmission Control Register
0x07	TLR	Trigger Level Register (TLR)	Trigger Level Register
0x08	TXLVL	Transmit FIFO Level register	n/a
0x09	RXLVL	Receive FIFO Level register	n/a
0x0A	IODir	I/O pin Direction register	I/O pin Direction register
0x0B	IOState	I/O pins State register	n/a
0x0C	IOIntEna	I/O Interrupt Enable register	Interrupt Enable register

Adrs	name	Read mode	Write mode
0x0E	IOControl	I/O pins Control register	I/O pins Control register
0x0F	EFCR	Extra Features Control Register	Extra Features Control Register
0x00	DLL	Divisor Latch LSB (DLL)	Divisor Latch LSB
0x01	DLH	Divisor Latch MSB (DLH)	Divisor Latch MSB
0x02	EFR	Enhanced Features Register (EFR)	Enhanced Features Register
0x04	XON1	Xon1 word	Xon1 word
0x05	XON2	Xon2 word	Xon2 word
0x06	XOFF1	Xoff1 word	Xoff1 word
0x07	XOFF2	Xoff2 word	Xoff2 word

2-2. ボーレート設定

SC16IS752のクロックとして、14.7456MHzの水晶振動子を使用しています。
代表的なボーレートジェネレータ(DLL,DLH)の設定値は以下のとおりです。

ボーレート[bps]	設定値	備考
110	8378	
300	3072	
1200	768	
2400	384	
4800	192	
9600	96	
14400	64	
19200	48	
38400	24	
57600	16	
115200	8	I2C速度(標準100kHz)を超えるのでフロー制御使用推奨
230400	4	
460800	2	
921600	1	RS485/422の上限(コントローラの上限)

RPi-GP70用Pythonサンプルファイル

RPi-GP70用Pythonサンプルファイルの使用方法について説明します。
Raspberry Piは'Raspberry Pi4 ModelB'、OSは'Raspberry Pi OS Version 2022-09-22'で説明します。 サンプルファイルは`sampleGp70.py`です。

準備

Raspberry PiにRPi-GP70を接続

下記の準備をおこなってください。

- OS(Raspberry Pi OS)のインストール
- GPIO40pinのI2C設定
- 'Raspberry Pi'に'RPi-G70'を接続
- シリアルドライバの設定

Pythonサンプルファイルを実行するディレクトリを作成

1. 'mkdir'コマンドを使って'RPi-GP70'という名前のディレクトリを作成します。(ディレクトリ名や作成場所は自由です)

```
$ mkdir RPi-GP70
```

2. 'ls'コマンドを実行して'RPi-GP70'ディレクトリが作成されていること確認します。

```
$ ls
```

3. 'cd'コマンドで'RPi-GP70'ディレクトリに移動します。

```
$ cd RPi-GP70
```

PythonサンプルファイルをGitHubからダウンロード

GitHubからPythonサンプルファイルをダウンロードします。

1. sampleGp70.pyをダウンロード

```
$ wget https://github.com/ratocsystems/rpi-
gp70/raw/master/python/sampleGp70.py
```

2. `ls`コマンドを実行してPythonサンプルファイル`sampleGp70.py`がダウンロードされていることを確認します。

```
$ ls
sampleGp70.py
```

Pythonサンプルファイルについて

`sampleGp70.py`

RPi-GP70を使用してシリアルの送受信を行うPythonサンプルプログラムです。
サンプルプログラムでは下記の処理を行っています。

1. **RPi-GP70の初期設定 init_GP70()**
   GPIOの初期設定を行います。
   ※ハードウェアに依存する設定ですので変更しないでください。

   - GPIOをGPIO番号で指定するように設定
   - 絶縁回路用電源をONに設定
     電源ON後、安定するまで待ちます。

2. **シリアル設定情報変更 input_param(serial)**
   pyserialモジュールのシリアル設定パラメータを入力値へ設定変更します。
   以下のパラメータを指定可能です。変更しない場合はenterのみを入力してください。

   - ボーレート[bps]
   - バイト長[bit]
   - パリティ[なし,奇数,偶数,スペース,マーク] ※
   - ストップビット長[bit]
   - 受信タイムアウト時間[sec]
   - フロー制御 XON/XOFF設定[False/True]
   - フロー制御 RTS/CTS設定[False/True]
   - フロー制御 DSR/DTR設定[False/True] ※
     ※ Pyserial 3.5 では、パリティのスペース、マークとフロー制御のDTR/DSRの設定は、本製品には無効です。

3. **メニュー表示**
   次のメニューを表示します。
   `1:送受信テスト(RS422全二重) 2:1:送受信テスト(RS485半二重) 3:設定 0:終了 >`

4. **送受信テスト**
   メニューで1または2が入力された場合は、シリアル送受信テストを行います。
   送信ポート番号と受信ポート番号(0,1)を入力します。enterのみでメニューに戻ります。
   `送信ポート番号(0, 1) enter:戻る >`
   `受信ポート番号(0, 1) enter:戻る >`
   メニューで2が入力されていればRS485の半二重モード動作(差動ドライバの自動イネーブル制御)を有効にします。

送信データ入力 enter:戻る >
で、送信する文字列を入力します。送信後に受信文字列を表示します。
受信は設定されたタイムアウトが発生するまで継続されます。
タイムアウト後に、送信データ入力に戻ります。
送信データ入力時にenterのみでメニューに戻ります。

5. **設定**
   メニューで3が入力された場合は、シリアル設定を変更します。
   ポート0とポート1についてシリアル設定情報変更 input_param() を行います。

6. **終了**
   メニューで0が入力された場合は、プログラムを終了します。

Pythonサンプルファイルの使い方

サンプルファイル名の前に、python3をつけて実行します。

- **シリアル送受信テスト**
  例）ボーレート115200bpsでポート0からポート1へ折り返し送受信テストをする場合

```
$ python3 sampleGp70.py
RPi-GP70 検査プログラム
1:送受信テスト(RS422全二重) 2:1:送受信テスト(RS485半二重) 3:設定 0:終了 > 3
[Port0 現在の設定]
Serial<id=0x76a49b90, open=False>(port='/dev/ttySC0', baudrate=9600,
bytesize=8, parity='N', stopbits=1, timeout=0.5, xonxoff=False,
rtscts=False, dsrdtr=False)
設定値を入力 enter:変更なし >

baudrate(300,1200,2400,4800,9600,14400,19200,38400,57600,115200,230400,46080
0,921600) = 115200
 bytesize(5, 6, 7, 8) =
 parity('N','E','O','S','M') =
 stopbits(1, 1.5, 2) =
 timeout =
 xonxoff(False, True) =
 rtscts(False, True) =
 dsrdtr(False, True) =
[Port1 現在の設定]
Serial<id=0x76a49bb0, open=False>(port='/dev/ttySC1', baudrate=9600,
bytesize=8, parity='N', stopbits=1, timeout=0.5, xonxoff=False,
rtscts=False, dsrdtr=False)
設定値を入力 enter:変更なし >

baudrate(300,1200,2400,4800,9600,14400,19200,38400,57600,115200,230400,46080
0,921600) = 115200
 bytesize(5, 6, 7, 8) =
 parity('N','E','O','S','M') =
 stopbits(1, 1.5, 2) =
 timeout =
```

```
 xonxoff(False, True) =
 rtscts(False, True) =
 dsrdtr(False, True) =
1:送受信テスト(RS422全二重) 2:1:送受信テスト(RS485半二重) 3:設定 0:終了 > 1
送信ポート番号(0, 1) enter:戻る > 0
受信ポート番号(0, 1) enter:戻る > 1
送信データ入力 enter:戻る > ABCD
受信データ：ABCD

送信データ入力 enter:戻る > EFGH
受信データ：EFGH

送信データ入力 enter:戻る > UUUU
受信データ：UUUU

送信データ入力 enter:戻る >
1:送受信テスト(RS422全二重) 2:1:送受信テスト(RS485半二重) 3:設定 0:終了 > 0
```

RPi-GP70のハードウェア仕様

製品仕様	
製品名称	Raspberry Pi I2C 絶縁型RS485ボード
製品型番	RPi-GP70
インターフェイス	GPIO40ピン：I2C
シリアルコントローラ	NXP SC16IS752
シリアルポート	2ポート (5ピンネジ式脱着型端子台)
RS-485/422A規格	TIA/EIA-485-A, TIA/EIA-422-B 準拠 (半二重/全二重) ※半二重通信は送受信自動切り替えに対応
シリアル設定	基板上のジャンパーピンにより設定が可能
対応ボーレート	110, 300, 1200, 2400, 4800, 9600, 14400, 19200, 38400, 57600, (115200, 230400, 460800, 921600) ※I2Cの速度制限(標準100kHz)により、115200bps以上はフロー制御を推奨
FIFOサイズ	送受信各64Byte
絶縁耐圧	2.5kV（シリアルトランシーバー部分） ※２つのシリアルポート間も絶縁
静電耐圧	15kV（シリアルトランシーバ性能値）
電源	+5V/+3.3V（GPIOポートから供給）
消費電流	最大 5V / 220 mA 、3.3V / 10 mA
動作環境	温度：0～40℃、湿度：20～80%（結露なきこと）
基板寸法	65.0 x 56.5 mm （突起物含まず）
重量	約 27 g
パッケージ内容	RPi-GP70本体 GPIO 40P ピンヘッダー 5ピン電線側ネジ式端子台 x2（基板へ装着済み） M2.6スペーサー x4 M2.6ネジ x8 設定用ショートプラグ x4（基板へ装着済み） ピン番号ラベル 保証書
生産	日本
保証期間	1年

【ご注意】　※ Raspberry Pi本体、キャリアボード・I/Oボード、パッケージ記載以外のケーブル類は添付していません。　※ GPIO ピンヘッダーコネクタは、ピンが曲がらないよう注意し、まっすぐに挿入してくだ

さい。

- 本製品は国内仕様となっており、海外での使用はできません。
- 本製品の運用を理由とする損失、逸失利益等の請求につきましては、いかなる責任も負いかねますので、予めご了承ください。
- 予告なく外観または仕様の一部を変更することがあります。
- 記載されている名称・商品名は各社の商標または登録商標です。