

RPN-1550
96CH SCALER
取扱説明書

Ver.1.0

HAYASHI-REPIC CO., LTD

ハヤシレピック株式会社

目次

| | |
|---------------------------------------|----|
| 1. 概要 | 3 |
| 2. 仕様 | 3 |
| 3. 外観 | 4 |
| 4. 使い方..... | 6 |
| 4.1. 液晶画面の操作 | 6 |
| 4.1.1. 表示チャンネル範囲を切り替える | 6 |
| 4.1.2. 表示チャンネル数を切り替える..... | 7 |
| 4.1.3. カウントモードを切り替える | 7 |
| 4.1.4. イーサネット情報を確認する | 8 |
| 4.1.5. DHCP を有効にする | 8 |
| 4.1.6. 固定 IP を設定する..... | 10 |
| 4.2. リモート・コントロール | 12 |
| 4.2.1. データを取得する | 13 |
| 4.2.2. 計測状態を取得する | 13 |
| 4.2.3. 計測を開始する | 14 |
| 4.2.4. 計測を停止する | 14 |
| 4.2.5. データをリセットする | 14 |
| 4.2.6. カウントモードを取得する | 14 |
| 4.2.7. カウントモードを合計カウント数にする | 15 |
| 4.2.8. カウントモードを 1 秒当たりのカウント数にする | 15 |
| 4.2.9. バージョン情報を取得する | 15 |
| 4.2.10. マルチクライアント接続の注意事項..... | 15 |
| 4.2.11. サンプルコード | 16 |
| 5. モニターアプリ | 19 |

1. 概要

RPN-1550 は素粒子物理学及び原子核物理学実験用に開発されたネガティブ ECL または NIM 信号 96 入力 8 桁表示のスケーラーです。

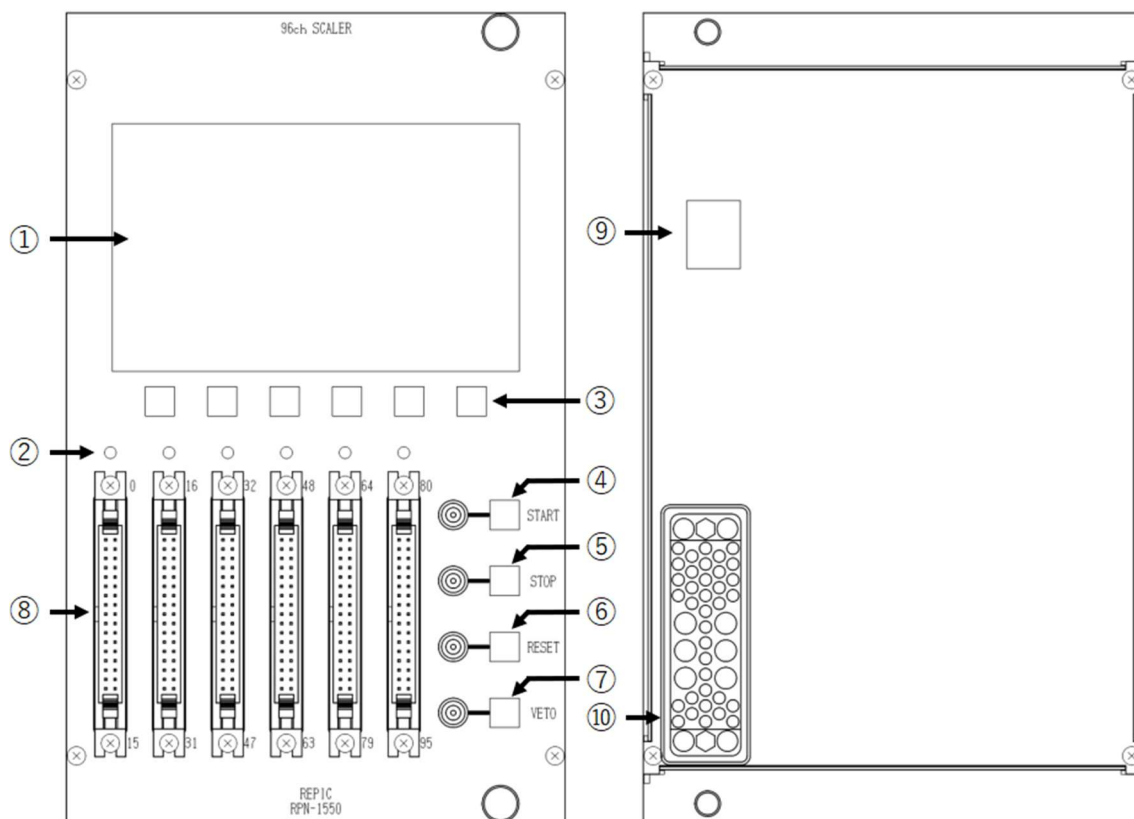
フロントパネルに液晶画面、プッシュスイッチと NIM 入力端子を搭載し、手動もしくは遠隔にて操作することが可能です。液晶画面には最大 96Ch のカウント値を同時に表示することができます。また、リアパネルには LAN コネクタを搭載し、カウント値をイーサネット経由で取得することが可能です。

2. 仕様

- COUNTER 信号入力
 - ネガティブ ECL または NIM レベル（ご指定ください）
 - 16ch 入力 34 ピンコネクタ ×6
 - 偶数ピンがプラス信号（ネガティブ ECL）またはマイナス信号（NIM）**GND(NIM)**
 - 奇数ピンがマイナス信号（ネガティブ ECL）または GND（NIM）**Sig.(NIM)**
 - 33, 34 番ピンは GND もしくは未接続（ご指定ください）
- START 信号入力
 - ネガティブ NIM
 - 50Ωインピーダンス
- STOP 信号入力
 - ネガティブ NIM
 - 50Ωインピーダンス
- RESET 信号入力
 - ネガティブ NIM
 - 50Ωインピーダンス
- VETO 信号入力
 - ネガティブ NIM
 - 50Ωインピーダンス
- 計数範囲
 - 0～99999999
- ケース
 - NIM 規格 4 幅

3. 外観

本モジュールの外観を以下に示します。



① 液晶画面

各チャンネルのカウント数、イーサネット関連の情報等を表示します。

② チャンネル表示 LED

現在液晶画面に表示されているチャンネルのコネクタ上の LED が点灯します。

③ 液晶画面操作スイッチ

ボタンを押すことで液晶画面に表示するチャンネル数やチャンネルの範囲の変更を行います。

④ START 信号入力

NIM 入力もしくはプッシュスイッチによってカウンターの計測を開始します。

⑤ STOP 信号入力

NIM 入力もしくはプッシュスイッチによってカウンターの計測を停止します。

⑥ RESET 信号入力

NIM 入力もしくはプッシュスイッチによってカウンターの計測数を消去します。

⑦ VETO 信号入力

NIM 入力もしくはプッシュスイッチによってカウンターの計測を一時停止します。

⑧ カウンター信号入力

ネガティブ ECL または NIM レベル信号のカウンター信号を入力します。

⑨ LAN コネクタ

イーサネットによるデータ取得や計測動作制御を行う際に LAN ケーブルを接続します。

⑩ NIM 規格コネクタ

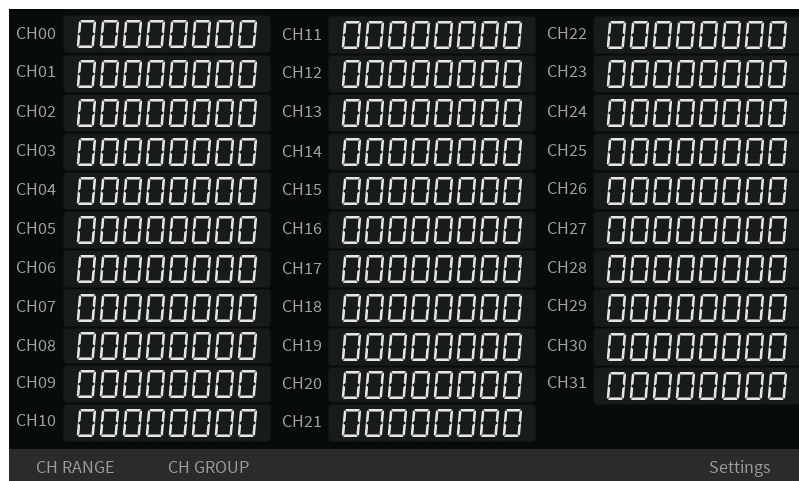
NIM クレートから電源供給を行いません。

4. 使い方

RPN-1550 の操作方法を説明します。

4.1. 液晶画面の操作

RPN-1550 を通電すると、フロントパネルの液晶画面に 32ch 画面が表示されます。



画面下部にメニューバーが表示され、液晶制御用ボタンを押すことで対応する機能呼び出すことができます。

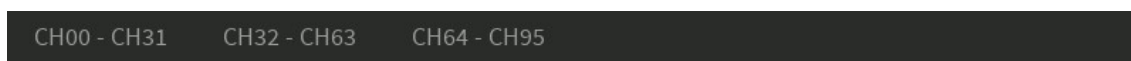
4.1.1. 表示チャンネル範囲を切り替える

1. メニューバーの[CH RANGE]を選択します

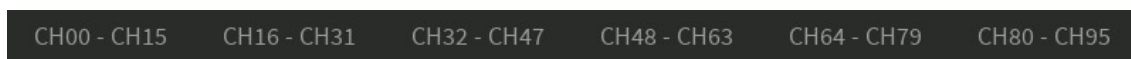


2. 画面に表示するチャンネル範囲を選択します

32ch 表示のメニュー



16ch 表示のメニュー



3. 選択したチャンネル範囲の画面に切り替わります

4.1.2. 表示チャンネル数を切り替える

1. メニューバーの[CH GROUP]を選択します



2. 画面に表示するチャンネル数を選択します



3. 選択したチャンネル数の画面に切り替わります

4.1.3. カウントモードを切り替える

1. メニューバーの[Settings]を選択します



2. メニューバーの[Count Mode]を選択します



3. 該当するモードを選択します



Total

入力された信号の総和を表示します。

Counts/s (CPS, Counts per second)

1 秒毎に自動リセットを行い、1 秒間に計測されたカウント数を表示します。

4.1.4. イーサネット情報を確認する

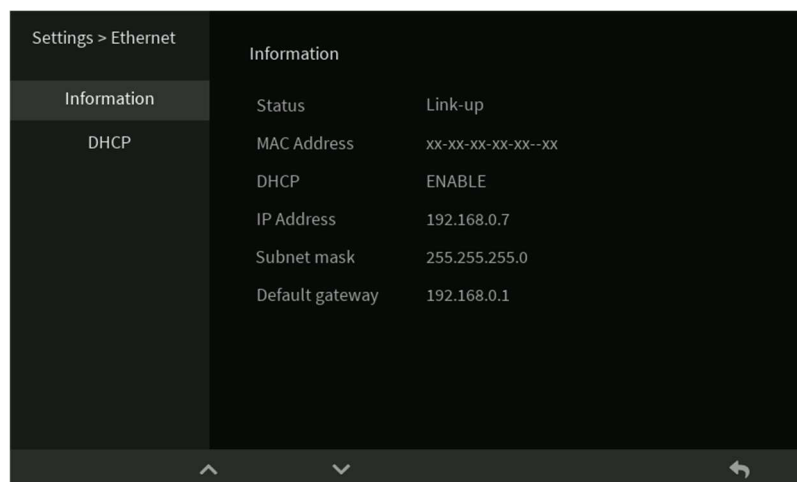
1. メニューバーの[Settings]を選択します



2. メニューバーの[Ethernet]を選択します



3. イーサネットの情報画面が表示されます



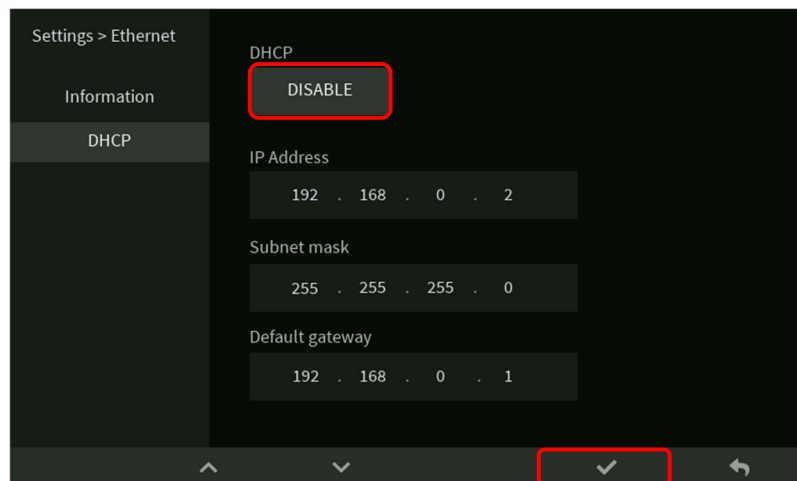
4.1.5. DHCP を有効にする

1. 「イーサネット情報を確認する」の手順によりイーサネット情報画面を表示します

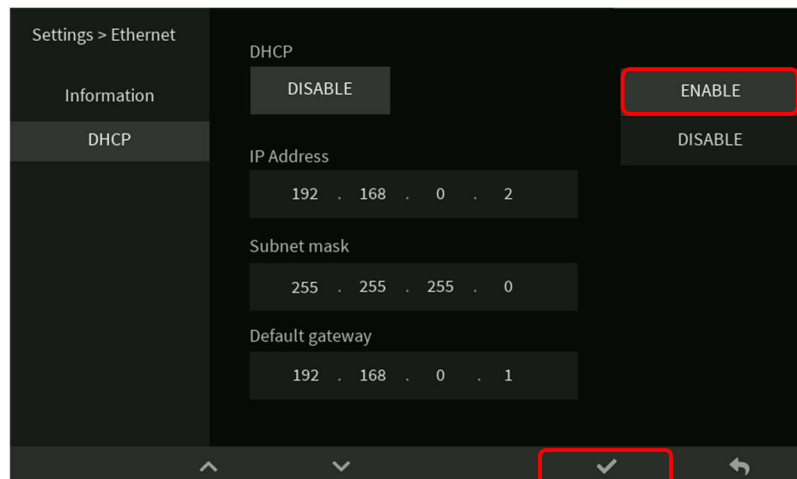
2. メニューバーの下ボタンを選択します



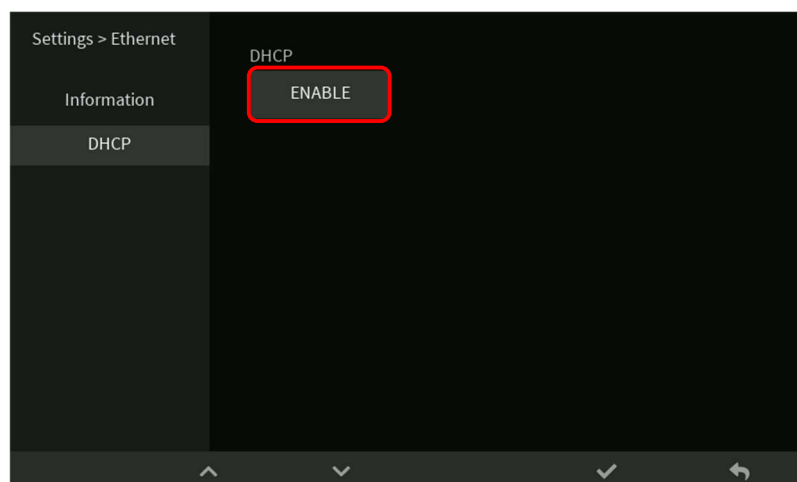
3. DHCP 設定画面に切り替わり、✓ボタンを押すと DHCP 欄が選択状態になるので、このまま再度✓ボタンを押します



4. DHCP 設定のウィンドウが表示された後、上下ボタンを使って[ENABLE]を選択し、✓ボタンを押します



5. DHCP 欄が ENABLE に切り替わります



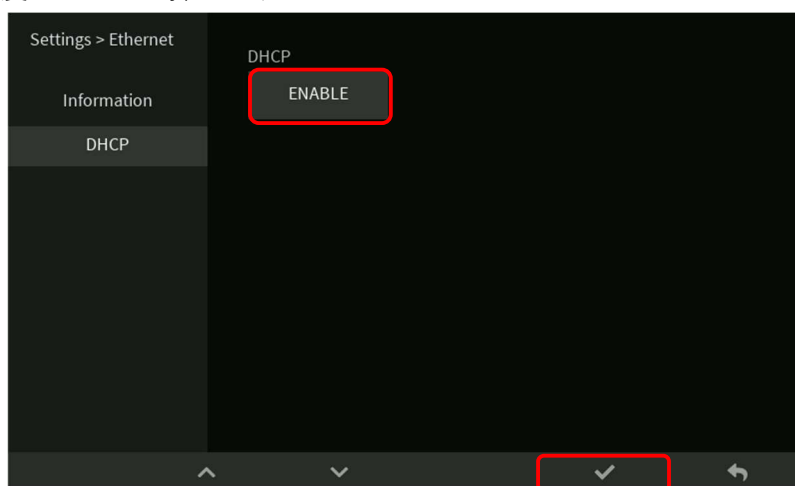
6. RPN-1550 を再起動します

4.1.6. 固定 IP を設定する

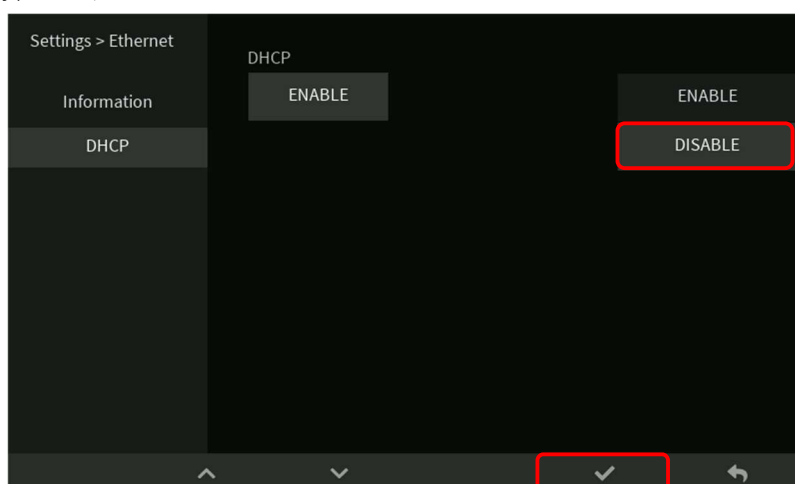
1. 「イーサネット情報を確認する」の手順によりイーサネット情報画面を表示します
2. メニューバーの下ボタンを選択します



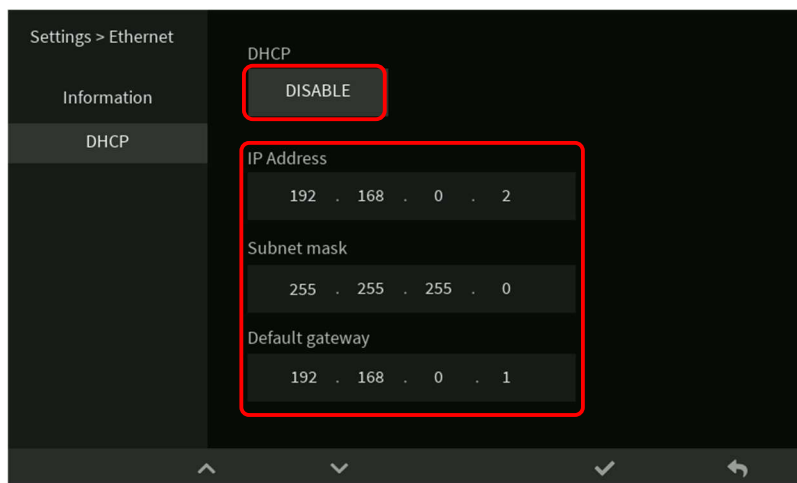
3. DHCP 設定画面に切り替わり、✓ ボタンを押すと DHCP 欄が選択状態になるので、このまま再度✓ ボタンを押します



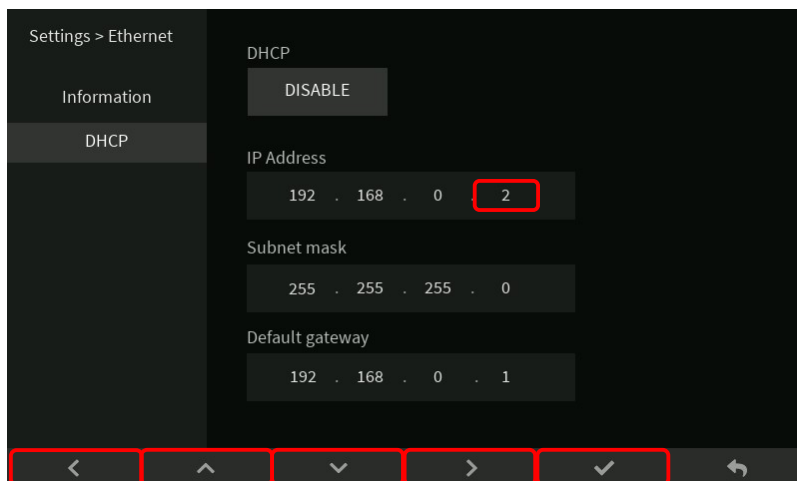
4. DHCP 設定のウィンドウが表示された後、上下ボタンを使って[DISABLE]を選択し、✓ ボタンを押します



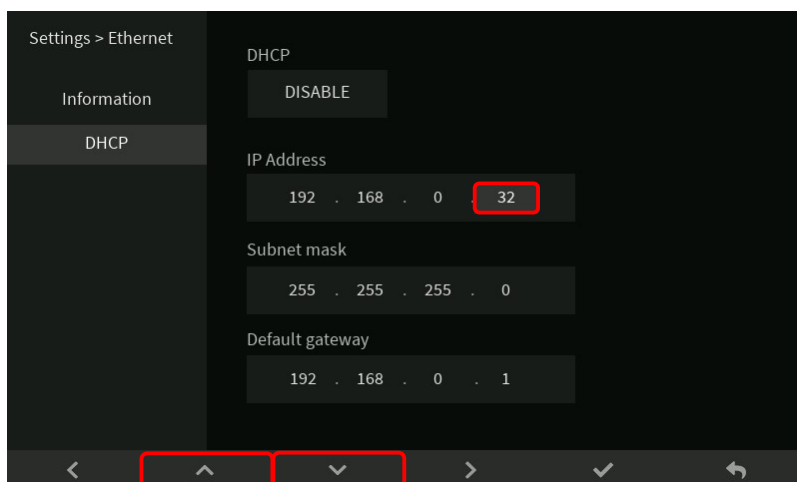
5. DHCP 欄が DISABLE に切り替わり、各ネットワーク設定の項目が表示されます



6. 上下左右ボタンおよび✓ボタンを操作して編集する項目を強調表示します



7. 上下ボタンを操作して数値を増減します（長押しすると連続で増減します）



8. ✓ ボタンを押して編集を確定します
9. RPN-1550 を再起動します

4.2. リモート・コントロール

RPN-1550 は、Web サーバーを搭載しています。HTTP による REST API 使用してリモートから計数の取得や計測の開始、停止を制御できます。

すべての API は HTTP の GET メソッドでアクセスします。すべてのデータは JSON で応答します。

以下は、IP アドレスが 192.168.0.2 に割り振られた RPN-1550 のバージョン情報を curl コマンドで取得する例です。

```
curl --get 192.168.0.2/api/version
{"version":"1.0.0"}
```

以下に URI に対応する RPN-1550 の機能を示します。

表 REST API 一覧

| URI | 機能 |
|--------------------------------|---------------------------------------|
| /api/data | 計数とオーバーフロー状態を取得する |
| /api/measure | 計測状態を取得する |
| /api/measure?state=start | 計測を開始する |
| /api/measure?state=stop | 計測を停止する |
| /api/reset?data | 計数を 0、オーバーフロー状態をクリアする |
| /api/settings/count | カウントモードを取得する |
| /api/settings/count?mode=total | 計測を合計のカウント数にする |
| /api/settings/count?mode=cps | 計測を 1 秒当たりのカウント数にする (1 秒毎にリセットを行う) |
| /api/version | バージョンを取得する |

サーバーは、API へのアクセスに対して HTTP ステータスコードを返します。GET 以外のメソッドまたは無効な URI をリクエストしたときは、エラーコードを返します。

表 アクセスに応じた HTTP ステータスコード

応答例

```
{"state":"start"}
```

4.2.3. 計測を開始する

計測を開始します。フロントパネルの START ボタンが点灯します。

URI /api/measure?state=start

応答

```
{"state":"start"}
```

4.2.4. 計測を停止する

計測を停止します。フロントパネルの STOP ボタンが点灯します。

URI /api/measure?state=stop

応答

```
{"state":"stop"}
```

4.2.5. データをリセットする

全チャンネルのカウント数を 0 にし、オーバーフロー状態を解除します。

URI /api/reset?data

4.2.6. カウントモードを取得する

カウントモードを取得します。

URI /api/settings/count

| データ | オブジェクト名 | 値 |
|---------|---------|-----------------------------------|
| カウントモード | mode | total=合計カウント数 cps=1 秒当たりのカウント数 |

応答例

```
{"mode":"total"}
```

4.2.7. カウントモードを合計カウント数にする

入力信号の合計をカウントします。

URI /api/settings/count?mode=total

4.2.8. カウントモードを 1 秒当たりのカウント数にする

1 秒毎にカウント数をリセットし、1 秒当たりの入力信号をカウントします。

URI /api/settings/count?mode=cps

4.2.9. バージョン情報を取得する

RPN-1550 のファームウェアバージョンを取得します。

URI /api/version

応答例

```
{"version":"1.0.0"}
```

4.2.10. マルチクライアント接続の注意事項

RPN-1550 が処理可能な最大セッション数は 8 ですが、推奨は 2 以下です。

使用例として、1 台目のクライアントは計数のモニターと制御用として接続し、2 台目のクライアントはカウント数をデータベースなどに保存するために接続します。3 つ以上のクライアントから計数を取得する必要がある場合は、モジュールへの直接通信を避け、データベースなどのデータ保存先から取得することをお勧めします。

リモート・コントロールが不要になったクライアントは、必ずソケットを閉じてサーバーとの切断を完了してください。接続状態が残ったまま他のクライアントとの接続を繰り返

すとリソースを逼迫し、最悪の場合、リモート・コントロールができなくなります。

RPN-1550 にセキュリティ機能はありません。任意のクライアントからのアクセスはすべて実行されます。第 3 者からのアクセスを抑制する必要がある場合は、ファイアウォール機能が搭載されたルーターなどを介して使用してください。

4.2.11. サンプルコード

バージョン情報を取得するサンプルコードを以下に示します。実行環境の構築やインストール手順については各公式ドキュメントを参照してください。

コマンドプロンプト (Windows) またはターミナル (MacOS、Linux) で入力する必要がある場合は、**太字**で示します。

1. Python3 (<https://www.python.org/>)

python を使用したサンプルコードを示します。

ファイル sample.py を以下の内容で作成し、python (または python3) を実行します。

sample.py

```
import http.client

conn = http.client.HTTPConnection("192.168.0.2", timeout=5)
conn.request("GET", "/api/version")
res = conn.getresponse()
data = res.read()
print(data)
conn.close()
```

実行と結果

```
python sample.py
200 OK
b'{"version":"1.0.0"}'
```

2. C#/.NET (<https://dotnet.microsoft.com/>)

RPN-1550 用クライアントのプロジェクトを作成し、REST API を呼び出すサンプルコードを示します。

プロジェクトの作成

```
dotnet new console --name RPN1550Client
cd RPN1550Client
```

Program.cs を以下のように編集します。

```
using System;
using System.Net.Http;
using System.Threading.Tasks;

namespace RPN1550Client
{
    class Program
    {
        private static readonly HttpClient client = new HttpClient();

        static async Task Main(string[] args)
        {
            var uri = "http://192.168.0.2/api/version"
            var res = await client.GetStringAsync(uri);

            Console.WriteLine(res);

            client.Dispose();
        }
    }
}
```

プロジェクト内で実行して、結果を確認します。

```
dotnet run
{"version":"1.0.0"}
```

さらに詳しい情報は、.NET ドキュメントの REST クライアントを参照してください。

<https://docs.microsoft.com/ja-jp/dotnet/csharp/tutorials/console-webapiclient>

3. C++/ROOT (<https://root.cern/>)

ROOT 6 のサンプルコードを示します。HTTP クライアントがフレームワークに用意されていないため、TCP ソケットを使って HTTP リクエストを送信する必要があります。

以下の内容でファイル rpn1550client.C を作成し、ROOT で実行します。

rpn1550client.C

```
void rpn1550client()
{
```

```

char recv_buffer[128] = {};
string host = "192.168.0.2";
TSocket *sock = new TSocket(host.c_str(), 80);

string send_str;
send_str = "GET /api/version HTTP/1.1\r\n";
send_str += "Host: " + host + "\r\n\r\n";

sock->SendRaw(send_str.c_str(), send_str.length());
sock->SetOption(kNoBlock, 1);
sock->Select();
sock->RecvRaw(recv_buffer, 128);
sock->Close();

cout << recv_buffer << endl;

delete sock;
}

```

実行と結果

```

root -l rpn1550client.C
HTTP/1.1 200 OK
Content-Type: application/json
Connection: keep-alive
Content-Length: 19

{"version":"1.0.0"}

```

5. モニターアプリ

RPN-1550 専用のモニターアプリを Microsoft Store にて公開しています。Window 10 から計数をモニターすることができます。



<https://www.microsoft.com/store/apps/9NBDL133007W>