

2018/3/8

Maxell インテリジェントバッテリー

並列接続アダプタ仕様書

Ver 1.2.2

空撮サービス株式会社

## 1 はじめに

### 1.1 インテリジェントバッテリーの必要性

ドローン用のバッテリーはラジコン用と異なり、より広範囲に使われることから非常に高い安全性が求められる。しかし DJI を除きドローン業界で安全なバッテリーへの採用はなかなか進んできていない。実際に墜落事故から発火事故に至っている例もあり、安全な運用にはインテリジェントバッテリーが必要である。マクセル社のリチウムイオン二次電池バッテリー（以下 Maxell バッテリー）はドローン用としては初めてインテリジェント機能を備え過充電・過放電・短絡に対する保護機能を備えて安全性を高めている。

### 1.2 ドローンでのバッテリー管理

ドローンでのバッテリー管理は主にバッテリー電圧で残容量を判断しているが、一般的な LIPO バッテリーと Maxell バッテリーは電圧が異なり、LIPO の常識で判断すると誤判断することになる。この様にバッテリーの電圧だけで残容量を判断する方式は誤解を生みやすく、一般人には理解不能で有り容量不足による墜落の元にもなりやすい。Maxell バッテリーは I2C による通信機能を持っており、バッテリーの電圧、電流、温度、セル毎の電圧、残電流量、最大電流量、エラーステータスなどを見ることができ、オートパイロットからログに残すことができる。この機能により正確な残容量を操縦者が常に確認することができるようになり、容量不足による墜落や電圧低下による不意の強制帰還中の衝突事故を防ぐことができる。またトラブルの場合でもログを調べることにより迅速に原因の究明ができる。

### 1.3 並列接続

この様に Maxell バッテリーは画期的なバッテリーであるが、残念ながら標準で 5200mAh の容量では enroute QC730 など中型機体では容量不足である事も否めない、そこで並列接続して 10400mAh の容量が確保できれば QC730 クラスで 15 分程度のフライト時間を確保することができ実用性が高まる。また並列にすることにより冗長性が高まり、どちらかに問題が起きたときでも安全に着陸でき安全性が高まる。しかしそのまま並列直結で接続した場

合、様々な問題が発生する。また I2C 接続もバッテリーの ID が共通で固定されているため、そのままでは片方の情報は読めますが両方の情報は読めない。

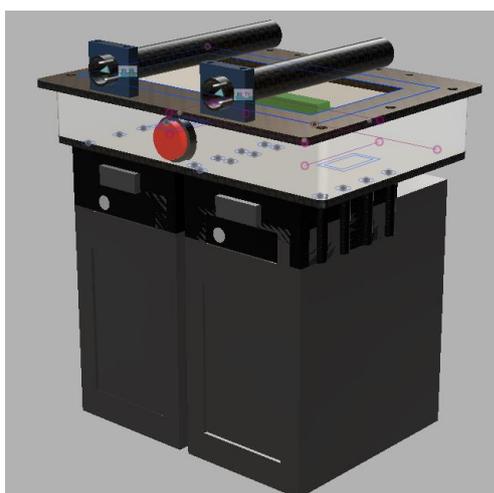
これらの問題を解決したのが、当社の並列アダプタです。

## 2 並列処理アダプター

これらの様々な問題を解決し、使いやすく安全な運用を可能とする並列アダプタを開発しました。

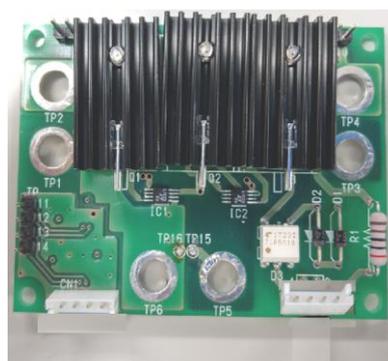
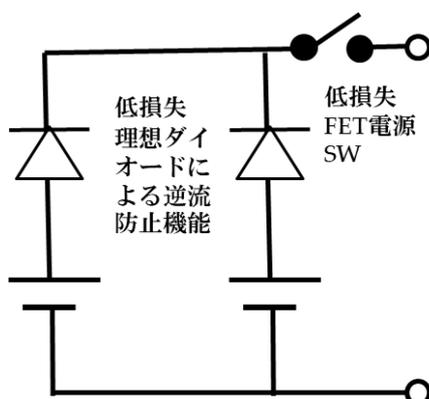
### 2.1 オン・オフボタン

機体の調整などしていると、頻繁に電源のオン・オフが必要になる。一般にドローンには電源のオン・オフ スイッチは付いていないのが普通である。そのため電池のコネクターを直接入れたり出したりしているのが普通である。しかしこの作業は間違っけがをしたり、ドローンを傷つけてしまうこともあり得る。本アダプタには前面にオンスイッチとオフスイッチが付いており、電源の入り切りが簡単にできます。低損失の FET を用いており、電圧降下はほとんどありません。オン状態は電氣的に保持されているので機械接点による誤切断もありません。



## 2.2 逆流防止機能

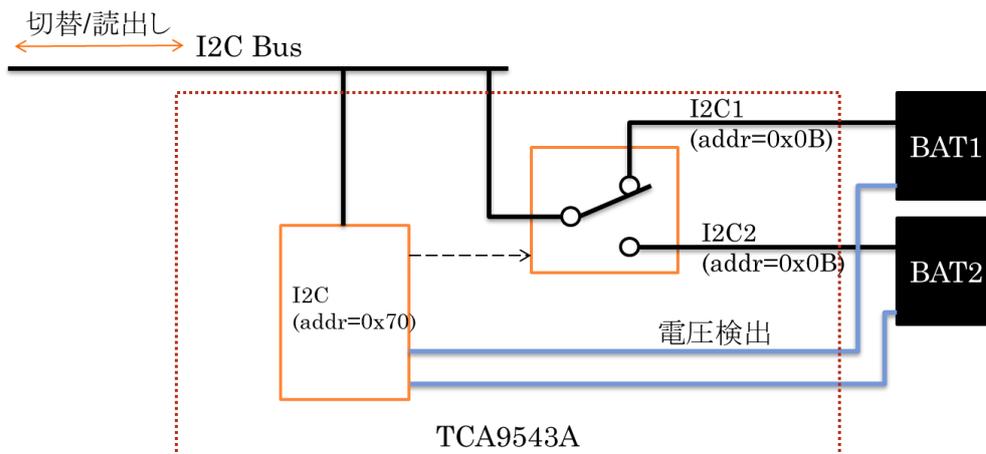
充電状態が異なるバッテリーを並列直結にすると、充電量の大きいバッテリーから充電量の少ないバッテリーに大量の充電電流が流れ過充電保護機能によりバッテリーのヒューズが溶断し使用不能になる。確かに RC 業界の一般常識では並列接続する場合は満充電のバッテリー同士を接続するというのが、常識ではあるが一般人にはなかなか徹底することは難しい。つい充電中のバッテリーを使ってしまうことも考えられる。この様なときにも安全にバッテリーを保護するにはダイオードによる逆流防止機能が必要であるが、ドローンのように大量の電流が流れる場合は大きな損失が発生して莫大な発熱がおき、実用的では無い。下図のように低損失 FET により理想ダイオードを実現しました。



FET と機能 IC を組み合わせた理想ダイオード回路により低損失で確実な逆流防止機能を実現した。

## 2.3 I2C 情報読取機能

Maxell インテリジェントバッテリーでは SMBus(I2C)により情報を読み出し可能。この機能により電圧・電流・残容量・温度などの重要な情報が読めるようになる。ただし、I2C の ID が同一のため二つのバッテリーを I2C BUS に接続しても正しく読めない。そこで本アダプターでは I2C スイッチによりバッテリーを切り替えて個々のバッテリーの情報を読み出している。



### 2.3.1 I2C アクセス概要

TCA9543A(addr=0x70)に書き込みを行うことで、I2C バスに接続されるバッテリー (BAT1/BAT2)を切り替える。

TCA9543A(addr=0x70)からの読出しにより、各バッテリーの I2C バス接続状況、および電圧検出状況を知ることができる。

バッテリー切り替え後は addr=0x0B へのアクセスにより接続されているバッテリーの情報を読み出すことが可能になる。

### 2.3.2 TCA9543A コマンド(書込)

7	6	5	4	3	2	1	0
X	X	X	X	X	X	B2	B1

ビット位置	意味
B1	0: BAT1 を I2C バスから切断する 1: BAT1 を I2C バスに接続する
B2	0: BAT2 を I2C バスから切断する 1: BAT2 を I2C バスに接続する

注) BAT1 と BAT2 は I2C アドレスが同一のため、両方を I2C バスに接続すると正しく制御できなくなる

### 2.3.3 TCA9543A コマンド(読出し)

7	6	5	4	3	2	1	0
X	X	V2	V1	X	X	B2	B1

ビット位置	意味
B1	0: BAT1 が I2C バスから切断されている 1: BAT1 が I2C バスに接続されている
B2	0: BAT2 が I2C バスから切断されている 1: BAT2 が I2C バスに接続されている
V1	0: BAT1 の電圧が非検出 1: BAT1 の電圧が検出している
V2	0: BAT2 の電圧が非検出 1: BAT2 の電圧が検出している

注) バッテリーのヒューズ切断状態で外部に電圧が供給されていない状態でも I2C 情報上では内部電圧が表示されます。必ず V1・V2 をチェックして外部に電圧が出力されているかチェックし非検出の場合は優先的に 0V に設定する。

### 2.3.4 バッテリーコマンドのあらまし

バッテリーコマンドは Smart Battery System 仕様 V1.1 に準拠しています。SMBus(I2C からの派生仕様)仕様に基づき、データの読出しを行います。読出し時に用いられるプロトコル(SMBus 仕様)は主に以下の仕様があります。

Read Byte/Word

Block Read/Write

注) 詳細についてはバッテリーの仕様を参照ください。

### 2.3.5 利用可能なコマンド（抜粋）

コマンドコード	利用 プロトコル	内容
0x08	Read Word	バッテリー温度(0.1 度 K)
0x09	Read Word	バッテリー電圧(mV)
0x0A	Read Word	電流(mA)(*1)
0x0D	Read Word	フル充電容量に対する充電状態(%)
0x0E	Read Word	設計容量に対する充電状態(%)
0x0F	Read Word	残容量(mAh)(*1)
0x10	Read Word	フル充電容量(mAh)(*1)
0x53	Block Read	Bit 0 セル過放電 Bit1 セル過充電 ヒューズ断 Bit2 充電過電流

\*1 マクセルバッテリーでは読出し値が実際の値の半分になる

## 2.4 Ardupilot ドライバー

Maxell 並列接続アダプターを Ardupilot で利用するには専用のドライバーが必要になる。

このドライバーを利用することにより、ミッションプランナーで電圧や電流、残容量などを表示することができ、ログ情報にバッテリーの詳細な情報を格納することができる。

### 2.4.1 Mavlink 出力

Mavlink の出力は下記になります。(バッテリー関連のみ抜粋)

Field Name	Type	仕様	出力内容
voltage_battery	uint16_t	Battery voltage, in millivolts (1 = 1 millivolt)	スロット毎の Voltage の最大値
current_battery	int16_t	Battery current, in 10*milliamperes (1 = 10 milliampere), -1: autopilot does not measure the current	Current の全スロット分の合計(10mA 単位)
battery_remaining	int8_t	Remaining battery energy: (0%: 0, 100%: 100), -1: autopilot estimate the remaining battery	RemainingCapacity の全スロット合計 / (スロット毎の DesignCapacity 最大値 x2) * 100
<a href="#">BATTERY_STATUS (#147)</a>			
Field Name	Type	仕様	出力内容
Id	uint8_t	Battery ID	0 固定
battery_function	uint8_t	Function of the battery	0(MAV_BATTERY_FUNCTION_UNKNOWN)固定
Type	uint8_t	Type (chemistry) of the battery	0(MAV_BATTERY_TYPE_UNKNOWN)固定
temperature	int16_t	Temperature of the battery in centi-degrees celsius. INT16_MAX for unknown temperature.	Temperature の最大値(1/100°C単位)
voltages	uint16_t[10]	Battery voltage of cells, in millivolts (1 = 1 millivolt). Cells above the valid cell count for this battery should have the UINT16_MAX value.	スロット 1 または 2 に接続されている電池の Cell Voltage 1~6(mV)

current_battery	int16_t	Battery current, in 10*milliamperes (1 = 10 milliampere), -1: autopilot does not measure the current	Current の全スロット分の合計(10mA 単位)
current_consumed	int32_t	Consumed charge, in milliamperere hours (1 = 1 mAh), -1: autopilot does not provide mAh consumption estimate	FC 起動時からの消費容量の合計(mAh) 全スロットについて RemainingCapacity に変化があった場合の変化値の合計
energy_consumed	int32_t	Consumed energy, in HectoJoules (intergrated U*I*dt) (1 = 100 Joule), -1: autopilot does not provide energy consumption estimate	-1 固定
battery_remaining	int8_t	Remaining battery energy: (0%: 0, 100%: 100), -1: autopilot does not estimate the remaining battery	RemainingCapacity の全スロット合計 / (スロット毎の DesignCapacity 最大値 x2) * 100

## 2.4.2 DataFlash ログ出力

バッテリー 1 情報 左側のバッテリーの情報

Field Name	Type	出力内容
TimeUS	uint64_t	記録時時刻(FC 起動からのマイクロ秒)
Volt	Float	Voltage(V 単位)
Curr	Float	Current(A 単位)
CurrTot	Float	FC 起動時からの消費容量の合計(mAh 単位) RemainingCapacity の変化量
Temp	int16_t	Temperature(1/100°C単位)
V1	uint16_t	Cell Voltage 1(mV)
V2	uint16_t	Cell Voltage 2(mV)
V3	uint16_t	Cell Voltage 3(mV)
V4	uint16_t	Cell Voltage 4(mV)
V5	uint16_t	Cell Voltage 5(mV)
V6	uint16_t	Cell Voltage 6(mV)

V7	uint16_t	0 固定
V8	uint16_t	0 固定
V9	uint16_t	残容量
V10	uint16_t	0 固定

バッテリー2 も同様

### 2.4.3 並列統合情報

ArduPilot ではバッテリー1 本が搭載されていることしか想定していないため、並列接続された各バッテリーの情報を統合する必要がある。二つのバッテリーの統合した情報は下記で計算される。

注) 満充電状態のバッテリーを一つだけ接続した場合、残容量は50%以下になります。

#### 1. 残容量(%)

$$\text{統合残容量} = \frac{\text{REMAINING\_CAPACITY1} + \text{REMAINING\_CAPACITY2}}{\text{DESIGN\_CAPACITY} \times 2} \times 100$$

#### 2. バッテリー電圧

Battery 1 または Battery2 のどちらか高い方の電圧

(これは並列誤挿入防止回路があるときの条件、通常は同じとなる)

#### 3. 電流値

Battery 1 と Battery2 の電流値を2倍して加算

#### 4. その他パラメータは、それぞれのバッテリーログに格納される。

### 2.5 バッテリーエラー情報

バッテリーから読み出せるエラー情報には 0X53 PF Status がある、Bit1 セル過充電ヒューズ断はセルが過充電になったため、保護回路が働きメインのヒューズを破断して最悪でも発火などが起きないようにする。これは充電状態が異なるバッテリーを直結したばあいなど逆流によっても発生する。ヒューズが破断するのでそのままでは使用できない。その他のエラーは、必ずしも検出できるとは限らない。またショートなどの場合、100A以上の電流でヒューズが溶断して出力電圧は0Vとなるが、この場合でもI2Cを通じて読み出される電圧値は内部状態を表示するので全て正常である。並列保護回路が入っているの

で、二つのバッテリーの内一方がヒューズ破断している場合、電流が流れないので使用中にもかかわらず何時までも電圧は高いままである、出力電圧は二つのバッテリーの内高い方の電圧が表示されるので、ドローン供給電圧表示は高い状態で固定となり正しい電圧が表示されない。そこでバッテリー外部出力電圧が供給されているかどうかを確認する回路を追加した、出ていなければヒューズ破断と見なし、I2C データを使用しない。

### 3 Ardupilot ドライバー機能

実際にマクセル並列処理ドライバーを導入した ArduPilot Ver 3.5 でのミッションプランナーでの表示例を各種紹介する。

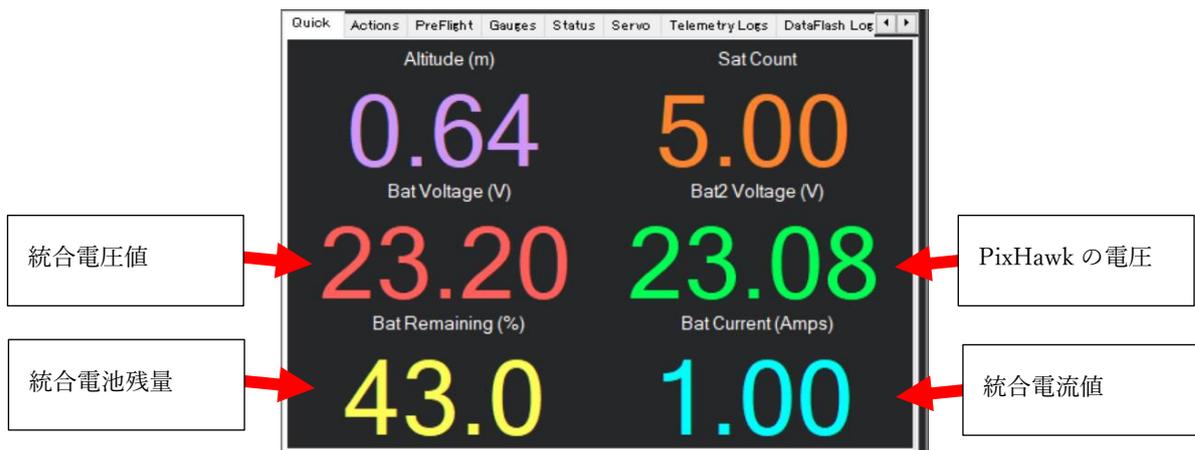
#### 3.1 正常並列接続

50%と30%程度残量があるバッテリーを並列に挿入しました。直結した場合はバッテリーが壊れますが、保護回路あるので問題ありません、Bat1



ミッションプランナーでの表示は下記内容になります。統合残容量は各バッテリーの残容量を合算し2台分の全容量で割ったものになります。

- 1) Bat1 Voltage(V) 各バッテリーの電圧を読み取って処理した電圧値
- 2) Bat2 Voltage(V) Pixhawk がアナログ端子から読み取った電圧値
- 3) Bat Remaining(%) ドライバーが計算した統合バッテリー残量、100%では 10400mAh となる。
- 4) Bat Current(A) ドライバーが各バッテリーの電流値を加算した値



### 3.2 正常単体接続

正常なバッテリーを片方に挿入したとき、バッテリー残量は2台分を100%としているので、半分の値となる。



### 3.3 出力ショートでヒューズ破断

出力がショートしてヒューズが破断（100A以上の電流が流れた）した状態のバッテリーを接続、出力電圧は0VだがI2C情報では電圧・残容量は正常に表示される。出力検出回路によりドライバーが電圧、残容量を0と表示している。



### 3.4 充電過電流でヒューズは断

異なる残容量のバッテリーを直結などで過充電したときは内部回路でヒューズが溶断される。この状態では出力は0Vであり、I2Cの過充電エラーフラグがオンとなっている。I2Cで電圧は読めるがドライバーは電圧を0Vにしている。



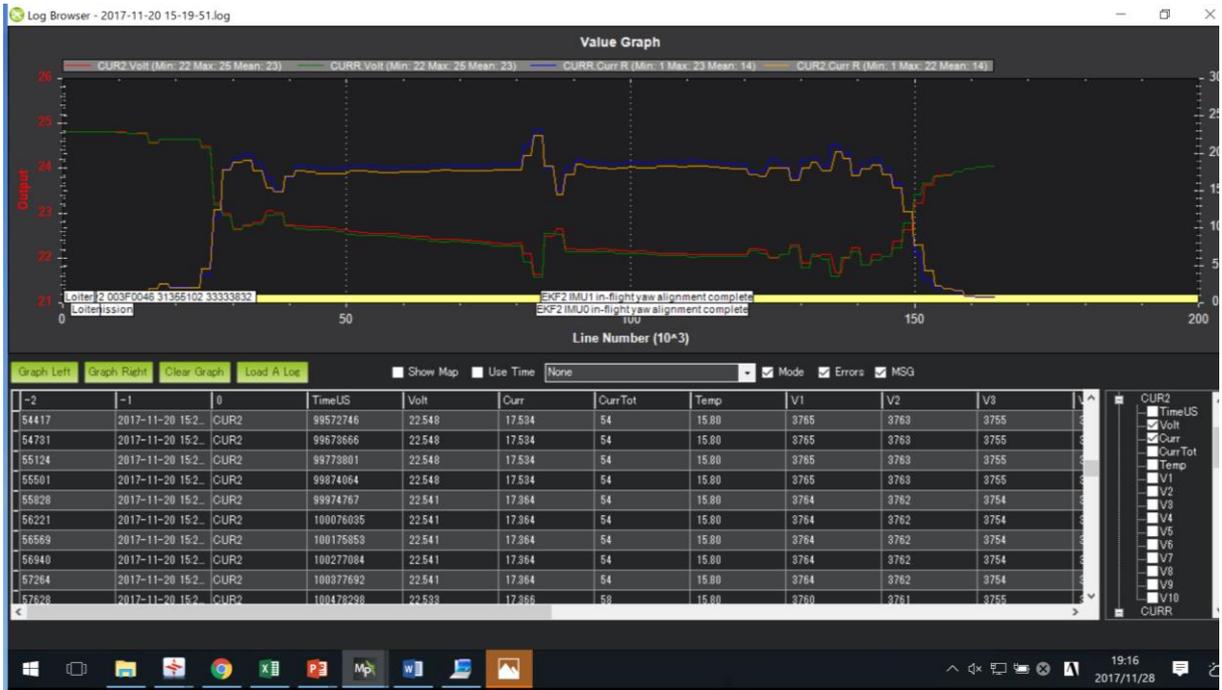
### 3.5 過放電状態

バッテリーを放電し続けると過放電状態となり充電はできなくなる。この場合でも出力電圧は若干あるので、ユニットの電源をオンすることはできる。I2Cでは過放電エラーとなっているので、ドライバーは残量、出力電圧ともに0Vとしている。下記の例ではPixHawkでは4.36Vの出力を検出している。



### 3.6 ログ情報

Ver3.5 では Bat1 の情報しかログできなかったが、並列処理ドライバーでは Bat2 の情報も正しく記録されています。



## 4 注意事項

### 4.1 使用の制限

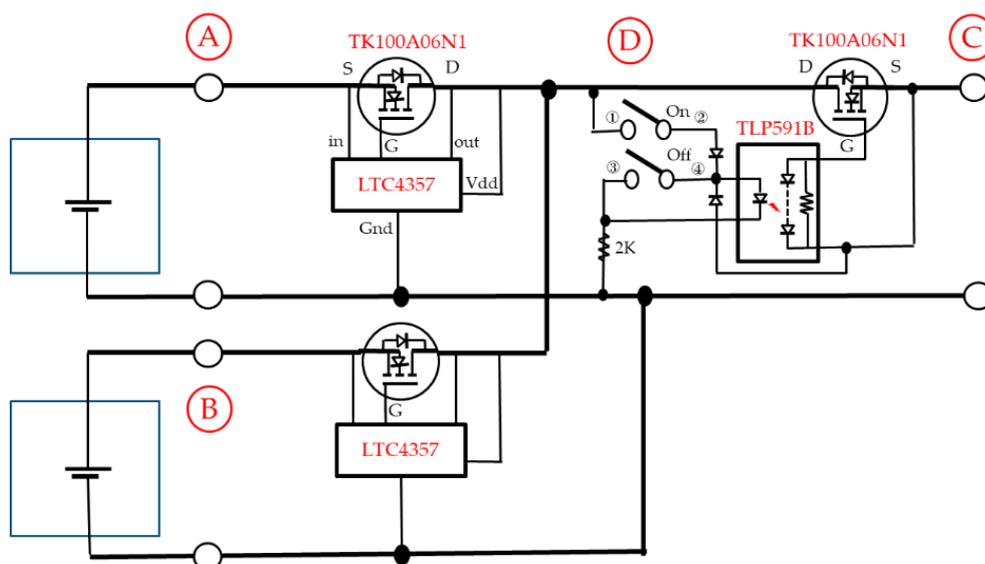
本基板はマクセルバッテリーの並列使用向けに設計されております。他の電池で利用される場合は回路の保証はできませんのでご注意ください。

### 4.2 用途

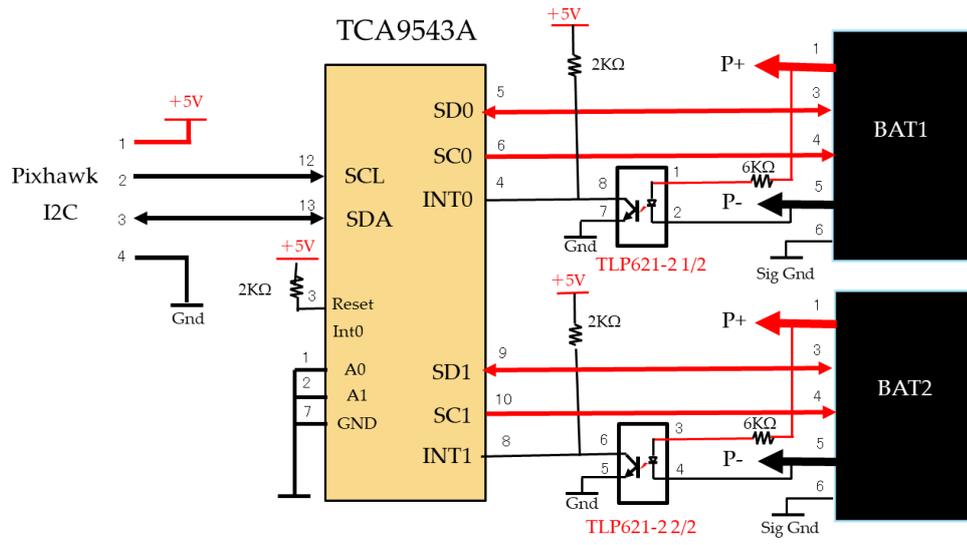
この基板はシステムに組み込んでご使用ください。

## 5 回路

### 5.1 並列接続回路



### 5.2 I2C スイッチ



### 5.3 接続図

