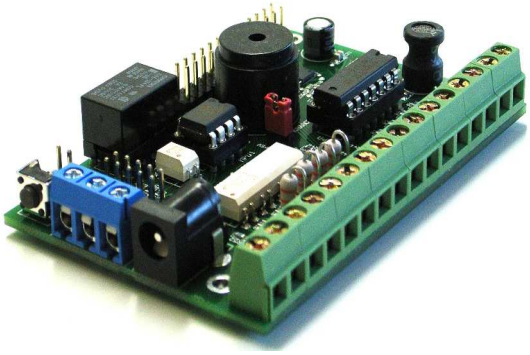
ラズベリーパイ用・拡張 I/O ボード

KARACRIX-KBRS31A (v1.1)

○RaspberryPi 基板を使用し合わせてたったの１万円前後で様々なシステムが組める。
○I/O は端子台出力で、全てネジ止めで組める。
○フォトカプラ&16Bit/ADC 搭載でプロにも使える。
○パイの死活監視機能があり無停止システム構築可能。
○フリーソフト(KaracrixBuilder32F/業務利用可)を利用して,SSL 警報メール発信,Web 監視操作,計測データ&操作履歴記録,自動制御ソフト作成などなど色々なアプリケーションが組めます。



● 1. はじめに

RaspberryPi/Model=A,A+,B(Rev2),B+,Pi2B,Pi3,Pi-Zero で使用できます。RaspberryPi と本ボードとの接続制御は、全て GPIO を介して行います。GPIO の使い方は、多くの出版書籍で詳しく書かれているのでそちらを参照ください。
簡単には：
GPIO は General Purpose Input/Output(汎用入出力)の略語であって、外部と接続する為のインターフェイス(接点)です。この GPIO を RaspberryPi 側のプログラムで ON(H)/OFF(L)させることによって本ボードをコントロールします。GPIO に ON/OFF を出力する手順は、該当チャネルの GPIO を出力モードにしておいてから 1(H)か 0(L)を書き込みます。GPIO の ON/OFF 状態を得る手順は、GPIO を入力モードにしておいてから指定 GPIO を読み込み 1 か 0 を知り ON か OFF を得ます。これら手順を具体的にどうプログラムに書き込むかは、使用するコンピュータ言語によって異なります。しかしながら、言語が異なろうとも、上記手順は全て同じです。

● 2. 電気仕様

1. 接点入力(Di/1-4)	--- 4 点
2. トランジスタ出力(Do/1-4)	--- 4 点
3. リレー出力(Do5)	--- 1 点
4. オンボードブザー&LED 出力(Do6)	--- 1 点
5. アナログ入力(Ai/1-4)	--- 4 点

6. 接点入力

a. 方式	無極性(AC)型フォトカプラ
b. 入力電圧	±5～±24V
c. 結線	絶縁(4 点 1 回路,コモン 1 点)

7. トランジスタ出力

a. 方式	オープンコレクタ(NPN)
b. Vceo/Ic/Pc	50V/500mA/0.2W
c. 結線	非絶縁,エミッタ GND 共通

8. リレー出力

a. 方式	脱着型・機械式リレー OMRON:G6E-134P-US DC5
b. 開閉部	1c 接点×1、Ag 合金
c. 許容電圧(電流)	AC125V (0.4A) DC30V (2A:抵抗負荷時)
d. 耐久性	10 万回以上

9. オンボード電子ブザー&LED 出力

a. ブザー型番	HDB06LFPN ※ブザー鳴動無効ジャンパ有り

10. アナログ入力

a. 分解能	16 ビット
b. 電圧	+6.144V,±3.072V,±6.144V
c. 入力オフセット	2～3mV
d. 入力バイアス電流	45～100nA (ハイ.インピ入力)
e. 計測基準電圧精度	0.1% 以下
f. 結線	非絶縁シングルエンド GND 共通

11. RaspberryPi 死活監視(RST)コントローラ

a.	専用 8 ビットマイコン

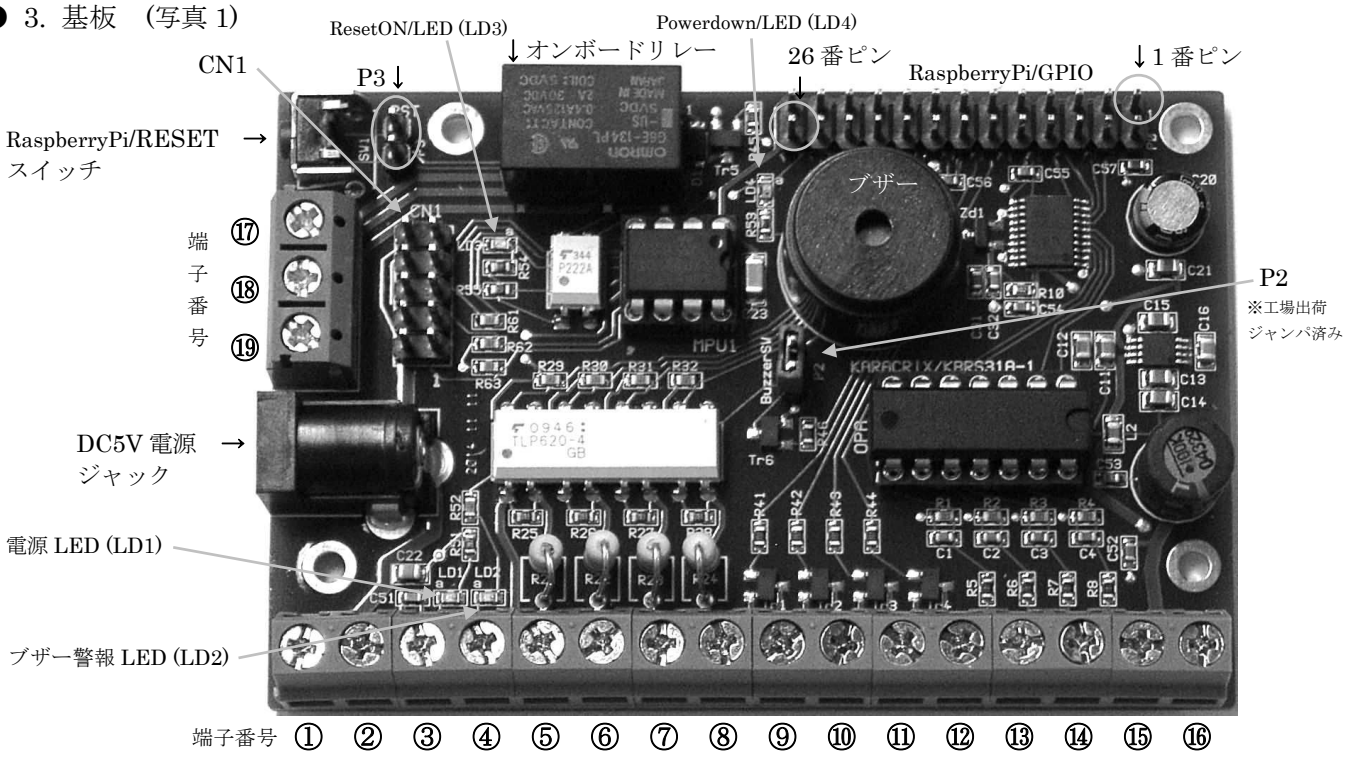
12. 入出力コネクタ

a. DC5V ジャック	内径 2.1mm/外形 5.5mmΦ センター・プラス
b. I/O 接続端子	16P,M2.5 ネジ,AWG(24-12) 4.4lb-in,むき=6～7mm
c. リレー端子	3P, M2.6 ネジ,AWG(22-14) 2.6lb-in,むき=5～6mm
d. マイクロ UPS 接続	10P(2.54mm)ヘッダピン

13. 電源電圧	DC5V (±5%厳守)
14. 本基板の消費電力	0.1W 以下(端子全開放時)
15. 動作&保存温度範囲	-40℃～+85℃ (結露厳禁) ※リレー(～+70℃),電子ブザー除く
16. サイズ(W×D)	85×56mm

説明書バージョン： v1.01 (2015.4.1)
開発: (株)エスアイ創房 http://www.karacrix.jp

● 3. 基板 (写真 1)



● 4. 回路 (図 1)

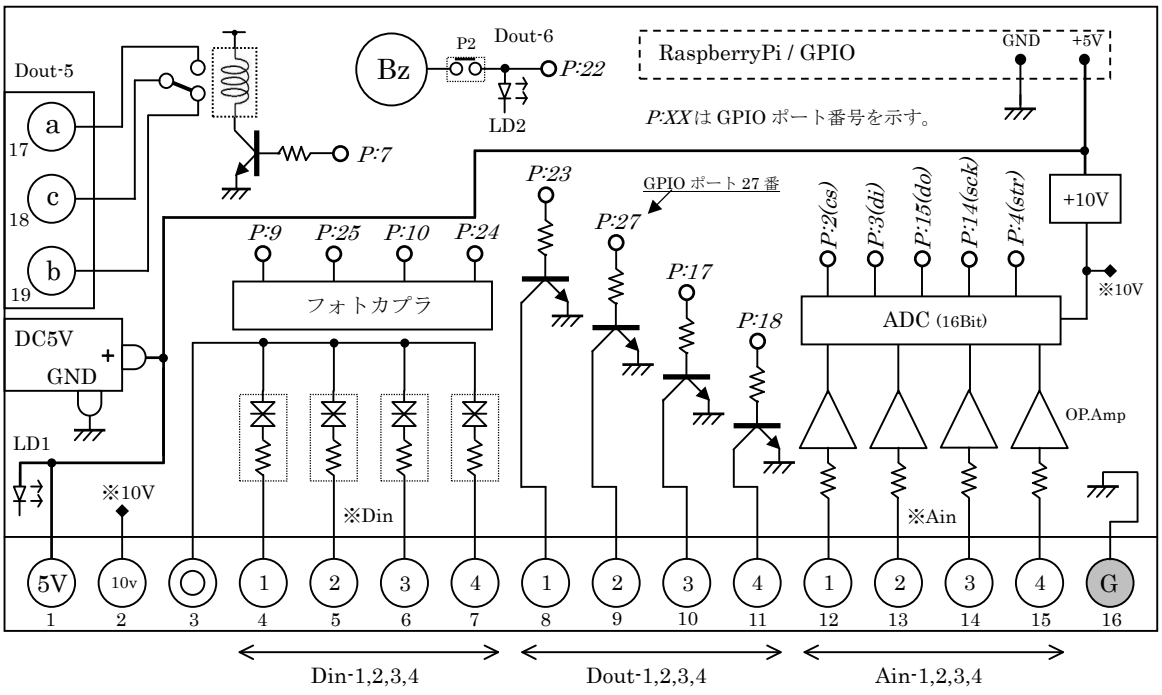
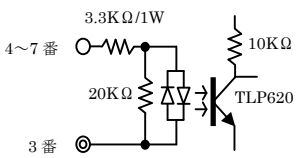


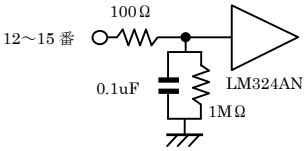
表 1. 端子台番号と機能

1	ボード 5V 電源	11	Do -4
2	10V (許容電流:12mA)	12	Ai -1
3	Di -片端共通端子	13	Ai -2
4	Di -1	14	Ai -3
5	Di -2	15	Ai -4
6	Di -3	16	GND
7	Di -4	17	リレーa(Do-5) 初期/OFF
8	Do -1	18	リレーc(Do-5) センター
9	Do -2	19	リレーb(Do-5) 初期/ON
10	Do -3	内部	ブザー(Do-6)

※Din 入力回路 (図 2)



※Ain 入力回路 (図 3)



● 5. 電源

本ボード上の DC ジャック(5V)あるいは 5V/GND 端子に 5V 電源を与えると、その電源は本ボードへの供給と供に、RaspberryPi/GPIO の電源ピン(フラットケーブルあるいはコネクタ)を経由し RaspberryPi 本体にも供給されます。(DC ジャックと 5V/GND 端子台は内部で繋がっています。ケーブルをネジ止めしたい場合は端子台をお使い下さい)
RaspberryPi 本体の microUSB より電源を供給している場合は、本ボードからの電源供給は必要ありません。

● 6. RaspberryPi との接続

RaspberryPi 本体/GPIO ピンと本ボードの 26P/GPIO ピンを接続します。
Model-A,B(Rev2)においては 26P フラットケーブル(例:秋月様通販コード C-06322//2014.12 現在 150 円)をご使用下さい。
Model-A+,B+,Pi2B においてもフラットケーブルは使用できますがケーブルヘッドが ID_SD(27),ID_SC(28)ピンに当たる為そのピンをカット(写真 6)して装着する必要があります。(ID_SD,ID_SC を使用することはあまり無いと思いますが,もし使用する場合は弊社コネクタをお使いください)
Model-A+,B+,Pi2B においては、弊社販売コネクタ(型番:NB-CON-26A//800 円)を使用して接続することが出来ます。
Model-A,A+,B(Rev2),B+,Pi2B の接続に関して、RaspberryPi 本体/GPIO(1～26)および本ボード GPIO(1～26)のピン番号が同じ番号同士合うように接続して下さい。
RaspberryPi/Model-A,B モデルは 26 ピン仕様ですが、Model-A+,B+,Pi2B は 40 ピン仕様です。
Model-A+,B+, Pi2B の場合 GPIO の 1 番ピンから 26 ピンまでをお繋ぎ下さい。

写真 2. RaspberryPi-A+,B+,Pi2B を NB-CON-26A にて繋げた様子
※ピンカット必要なし

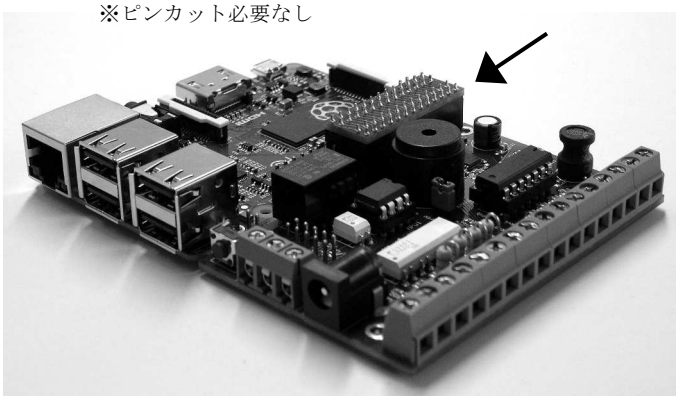


写真 3. RaspberryPi-A+,B+,Pi2B をフラットケーブルにて繋げた様子
※ピンカット必要あり

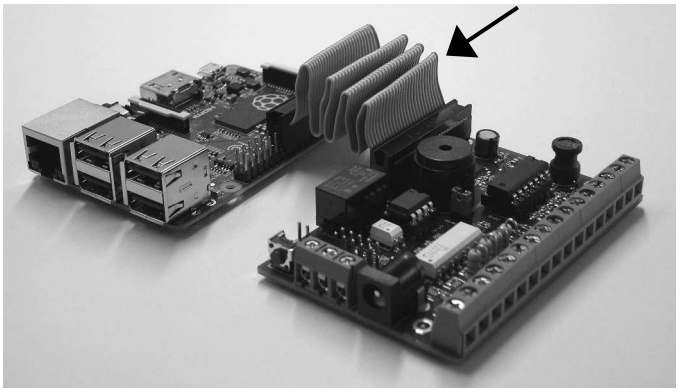
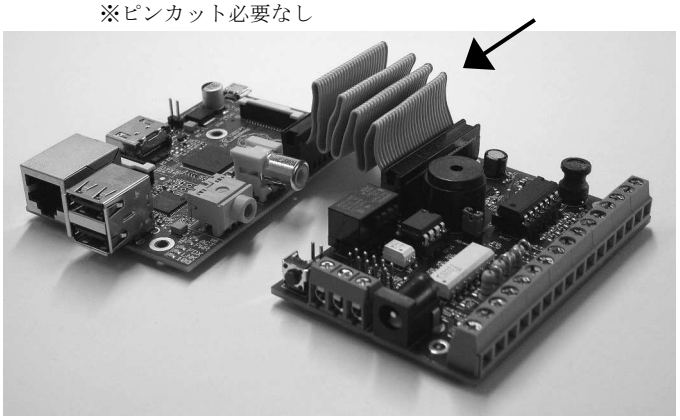
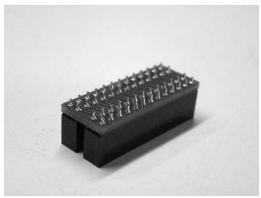


写真 4. RaspberryPi-A,B(旧タイプ)をフラットケーブルにて繋げた様子
※ピンカット必要なし





NB-CON-26A
A+,B+ タイプ使用可
A.B タイプ使用不可

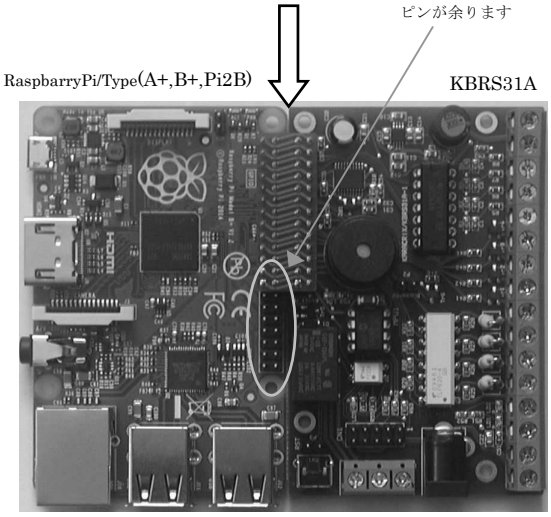


写真 5. NB-CON-26A の使用

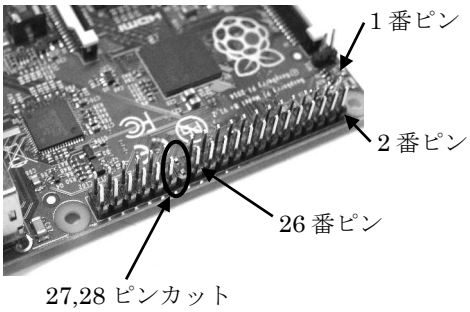


写真 6. RaspberryPi ピン加工

● 7. 基板説明と<u>ご注意</u>

1. 電源の+-極性は保護されていませんので、<u>極性を間違えないように電源を与えて下さい</u>。
2. 本ボード側と RaspberryPi 本体側の<u>両方で電源を与えないで下さい。</u>事故に繋がります。
3. 基板上の LD1 と書かれた LED は、電源 LED です。
4. 基板上の LD2 と書かれた LED は、GPIO-22 に繋がれた LED です。GPIO-22 には、電子ブザーが取り付けられておりブザーと LED は連動して動作します。なお、電子ブザーは P2 のジャンパを外すことによって GPIO-22 から切り離すことが出来ます。LD2 の LED は、インジケータでもあり LD1 の LED より少し明るく光るよう設計されています。
5. 端子台番号 2 から、10V を得ることが出来ます。但しこの電源容量は少ないので 12mA 以内でお使い下さい。また、出力電圧も負荷電流により 10V～8V の範囲で変動します。なお、本電源には保護回路が付いておりませんので使用にはご注意下さい。

● 8. 止め穴寸法

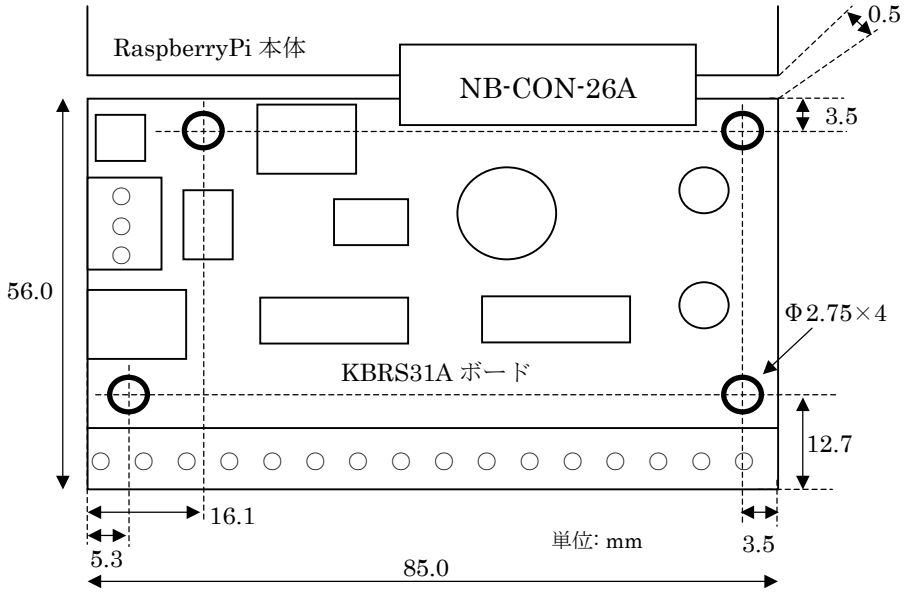


図 4

● 9. I/O の使い方

1. Di-1～4ch の ON/OFF 状態を得る場合は、GPIO-9,25,10,24 のそれぞれを読み込んで下さい。
2. Do-1～6ch を操作する場合は、GPIO-23,27,17,18,7,22 にそれぞれ 1(ON)か 0(OFF)を書き込んで下さい。
3. アナログ状態を得る場合は、ADC にマキシム社の MAX1301 を使用しておりますのでそのマニュアルに従って計測データを得て下さい。なお次頁に計測アルゴリズムの一例をサンプルにしまとめたものがありますので参考にして下さい。
4. ウオッチドッグ信号を生成する場合は、GPIO-8 に 1 及び 0 を交互に書き込んで下さい。パワーダウン(外部電源停止)信号を検出する場合は、GPIO-11 を読み込んで下さい。

表 2.1 RaspberryPi/GPIO の物理ピン番号と GPIO ポート番号と入出力の方向と対応する機能

PIN 番号	GPIO ポート番号	方向	対象	PIN 番号	GPIO ポート番号	方向	対象
21	9	IN	Di-1ch	15	22	OUT	Do-6ch(Buzzer)
22	25	IN	Di-2ch	3	2	OUT	Adc-cs
19	10	IN	Di-3ch	5	3	OUT	Adc-din
18	24	IN	Di-4ch	10	15	IN	Adc-dout
16	23	OUT	Do-1ch(Tr)	8	14	OUT	Adc-sclk
13	27	OUT	Do-2ch(Tr)	7	4	IN	Adc-strb
11	17	OUT	Do-3ch(Tr)	24	8	OUT	WatchdogTimer(H->L->H->L..)
12	18	OUT	Do-4ch(Tr)	23	11	IN	Powerdown(H:V1<V2)
26	7	OUT	Do-5ch(Relay)				

※Di は、端子オープン時に GPIO 入力レベル H、 端子ショート時に GPIO 入力レベル L となります。

※GPIO-11 は、スタート時にマイコンの起動を知らせる為 80msec×8 回点滅します。

表 2.2 RaspberryPi 物理ピン番号と GPIO ポート番号(Pxx)及び電源(+)とグランド(GND)の位置

GPIO-PORT	GND	P11	P9	P10	+3.3v	P22	P27	P17	GND	P4	P3	P2	+3.3v
PIN-NO	25	23	21	19	17	15	13	11	9	7	5	3	1
PIN-NO	26	24	22	20	18	16	14	12	10	8	6	4	2
GPIO-PORT	P7	P8	P25	GND	P24	P23	GND	P18	P15	P14	GND	+5v	+5v

★ アナログデータの計測アルゴリズム (サンプル)

アナログデータの取得は、GPIO を介して ADConverter とやり取りして得ます。
そのやり取りのアルゴリズムの一例を以下に示します。記述は C 言語風に書いています。アルゴリズム(操作手順)は言語に関係ないので他の言語をご使用の場合は、その記述に合わせ書き換えてお使い下さい。

○サンプル記述に出てくる関数(サブルーチン)等の説明

msleep(10) は、10msec 待つという意味です。
SET(GPIO 番号, データ) は、指定 GPIO にデータ(1 か 0)をセットするという意味です。
GET(GPIO 番号) は、指定 GPIO の状態を取得するという意味です。返り値よりデータが得られます。
sdc_reset()　　AD コンバータをリセットさせます。
adc_rangeset()　入力電圧のレンジを設定します。設定は 1 回行えば維持します。
adc_rangeset(1, 0) の記述は、2 チャンネルのアナログ入力のレンジを+6V にするという意味になります。
adc_getdata()　　入力電圧を計測取得します。値は、0～65535 の範囲で得られます。
sdc_getdata(3)　の記述は、4 チャンネルのアナログ入力データを得るという意味になります。
P2_cs は、cs に使う 2 番の GPIO を意味します。以下、P14_sclk は 14 番、P3_din は 3 番、P15_dout は 3 番、P4_strb は 4 番を意味します。

```
adc_reset()
{
   int i,cmd;
      SET( P2_cs,     1 );
      SET( P14_sclk, 0 );
      msleep( 10 );
      cmd = 0x00C8;
      SET( P2_cs, 0 );
      for(i=0;i<8;i++){
         SET( P3_din, cmd & (0x80>>i) );
         SET( P14_sclk, 1 );
         SET( P14_sclk, 0 );
      }
      SET( P2_cs,     1 );
      SET( P14_sclk, 0 );
      msleep( 10 );
      cmd = 0x00A8;
      SET( P2_cs, 0 );
      for(i=0;i<8;i++){
         SET( P3_din, cmd & (0x80>>i) );
         SET( P14_sclk, 1 );
         SET( P14_sclk, 0 );
      }
      SET( P2_cs,     1 );
      SET( P14_sclk, 0 );
}
```

```
adc_rangeset( int ch /*0-3*/, int range /*0-2*/ )
{
   int i,cmd,range;
      switch( range ){
      case 0: range = 0x03; break; /*( +6v)*/
      case 1: range = 0x01; break; /*(+-3v)*/
      case 2: range = 0x04; break; /*(+-6v)*/
      }
      cmd = (ch<<4) | range | 0x80;
      SET( P2_cs, 0 );
      for(i=0;i<8;i++){
         SET( P3_din, cmd & (0x80>>i) );
         SET( P14_sclk, 1 );
         SET( P14_sclk, 0 );
      }
      SET( P2_cs,     1 );
      SET( P14_sclk, 0 );
}
```

```
int
adc_getdata( int ch /*0-3*/ )
{
   int      i,cmd,status,bit,val;
      SET( P2_cs,     1 );
      SET( P14_sclk, 0 );
      SET( P3_din,    0 );
      SET( P2_cs,     0 );
      cmd = (ch<<4) | 0x80;
      for(i=0;i<8;i++){
         SET( P3_din, cmd & (0x80>>i) );
         SET( P14_sclk, 1 );
         SET( P14_sclk, 0 );
      }
      SET( P2_cs, 1 );
      for(i=0;i<20;i++){
         /* Wait strb Hi */
         status = GET( P4_strb );
         if( status > 0 ) break;
         msleep(0.1);
      }
      SET( P2_cs, 0 );
      val = 0;
      for(i=0;i<16;i++){
         SET( P14_sclk, 1 );
         status = GET( P15_dout );
         if( status > 0 ) bit = 1;
         else             bit = 0;
         val = (val<<1) | bit;
         SET( P14_sclk, 0 );
      }
      SET( P2_cs, 1 );
      SET( P3_din, 0 );
      return val;
}
```

```
main()
{
   int ch,data_1ch;
      adc_reset();
      for(ch=0;ch<4;ch++) adc_rangeset(ch,0);
      data_1ch = adc_getdata(0);
      printf("1ch:adcval = %d¥n", data_1ch);
}
```

● 10. ウォッチドッグリセット機能

本ボードには、RaspberryPi の異常時に RaspberryPi をリセットスタートさせる機能が搭載されています。
RaspberryPi の異常とは、RaspberryPi が本ボードにウオッチドッグ信号を送れなくなった状態を言います。

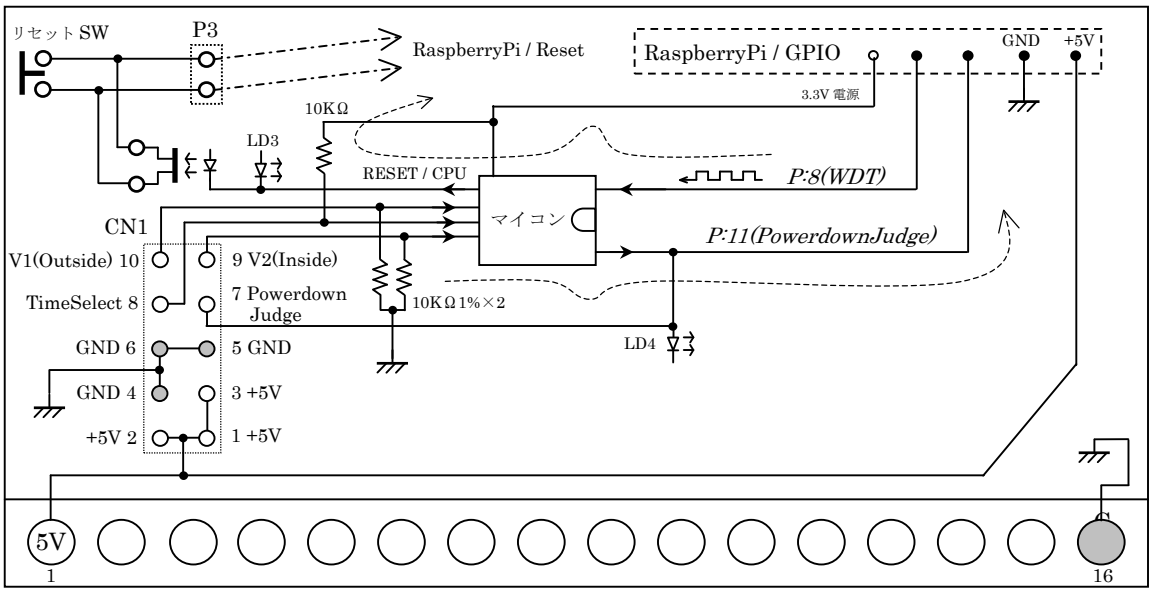


図 5

表 3.　CN1 ピン番号と機能

1	ボード 5V 電源	6	GND
2	ボード 5V 電源	7	GPIO-11 (Powerdown)
3	ボード 5V 電源	8	TimeSelect (Open / 10 分)
4	GND	9	V2 (基準電圧:ボード電源)
5	GND	10	V1 (計測電圧:外部電源)

表 4.　P3 ピン番号と機能

1	リセット用フォト MOS リレー出力 A, スイッチ A
2	リセット用フォト MOS リレー出力 B, スイッチ B

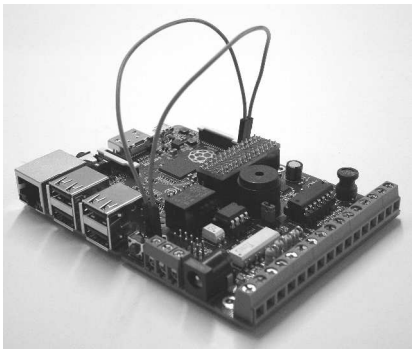


写真 7.　P3 と RaspberryPi のリセットピンを繋いでいる様子

1. 機能の実行
   本機能は、P3 ピンと RaspberryPi のリセットピンがワイヤーで繋がっている場合にのみ有効です。
   この場合、RaspberryPi は本ボードに対してウオッチドッグ信号を送り続ける必要があります。
   ワイヤーが繋がっていない場合は、本機能は無意味でまた実行されません。
   ※RaspberryPi の基板にはリセットピンホールがありますがピンが立っていません。使用する時はこれに考慮する必要があります。
2. ウオッチドッグ信号の発生
   ウオッチドッグ信号は、GPIO-8 の H/L のレベル変化を言います。
   このウオッチドッグ信号を RaspberryPi 側が発生させ、本ボードがこれを受け取ります。
   ウオッチドッグ信号の発生は、GPIO-8 のレベルを H->L->H->L..と変化させることで、以下にその仕様を示します。
   a. H 時間幅と L 時間幅の比(デューティ比)に制限はありません。
   b. H/L 変化させる最少時間は、プログラムで制御する限りどんなに短くても構いません。
   c. H/L 変化させる最大時間は、1 分以内にして下さい。
   d. H/L 変化させる推奨時間は、おおよそ 100msec～10sec です。
3. リセットの実行 (リセットの実行は LED(LD3)の点灯(約 1 秒)でも確認できます)
   本ボードは、ウオッチドッグ信号(GPIO-8 のレベル変化)の受信が途絶えてから 10 分(3 分)※1 後に RaspberryPi をリセットします。もしこのリセットで RaspberryPi が再起動せず、ウオッチドッグ信号が途絶え続けた場合、本ボードは再び 10 分(3 分)後に RaspberryPi をリセットし、この動作は永遠に続きます。
   ※1：CN1 の 7(TimeSelect)ピンを GND(6)にショートさせると 3 分になります。無接続(オープン)時は 10 分になります。
4. リセット時間誤差
   周囲温度によって 1～3%の誤差が発生します。

● 11. 外部電圧コンパレータ機能（外部電源停止時の RaspberryPi シャットダウンに利用）

CN1 の 10 番ピンに印加される電圧(V1)と 9 番ピンに印加される電圧(V2)を比較して、V2 が約 0.1V 以上でかつ V1＜V2 となった時、GPIO-11 のレベルを H（LED(LD4)も点灯）にします。RaspberryPi は、この GPIO-11 を利用して shutdown などのコマンドを利用できます。RaspberryPi でこの GPIO-11 をどう使うかは自由です。

※V1,V2 で計測できる電圧制限範囲は 0～3.3V です。計測電圧はこの範囲を超えないで下さい。

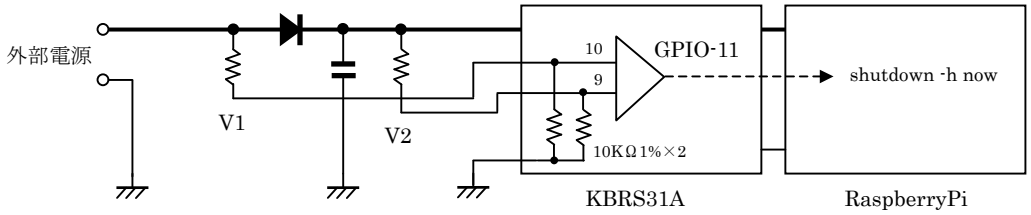


図 6

● 12. 結線例

KBRS31A への結線例を示します。

接点入力(Di)にスイッチを一つ、トランジスタ出力(Do)にリレーを取り付け 100V 電球を駆動させます。アナログ入力(Ai)として温度センサを一つ取り付けてみました。

※センサー＆アクチュエータの接続に関し、弊社 HP「実用ガイド 1 章→巻末付録」などを参考にして下さい。

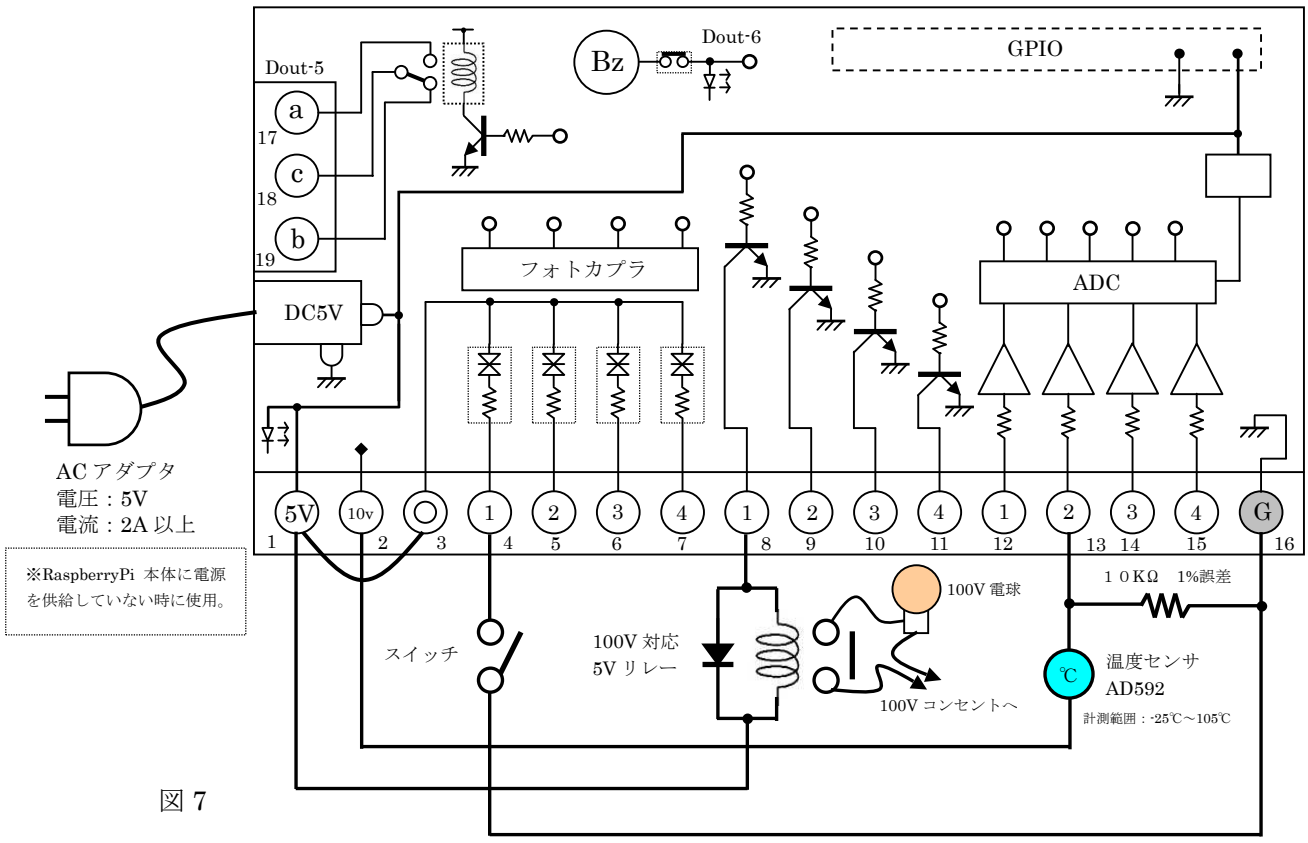


図 7

● 13. 付録

弊社 HP よりダウンロードできるフリーソフト KaracrixBuilder32F（通信制御ドライバ S1(オープンソース)）では、KBRS31A を使用しています。KBRS31A 全ての Di,Do,Ai を読み込み出力しています。また、ウォッチドッグ信号を発生させ、パワーダウン時には RaspberryPi/OS をシャットダウンさせるコードも書かれています。C 言語で書かれていますが、何かの参考にご利用下さい。

なお、フリーソフトのプログラム部に関し、概要を除きサポートは有料です。予めご了承下さい。

■ KBRS31A お取り扱い上の注意

本製品は電子機器です。お取り扱いについては、次の点にご注意下さい。

(1) 落とす、ぶつけるなどの衝撃を与えないで下さい。
(2) 振動の激しい場所で使用、保管しないで下さい。
(3) 温度の高い場所、直射日光の当たる場所で使用、保管しないで下さい。
(4) 湿度の高い場所や、水に濡れる場所で使用、保管しないで下さい。
(5) 温度、湿度の変化の激しい場所で使用、保管しないで下さい。
(6) 磁界、電界の強い場所で使用、保管しないで下さい。
(7) 電源の不安定な場所や、高調波の含まれる場所で、使用しないで下さい。
(8) 塵埃の多い場所で使用、保管しないで下さい。
(9) 液体等の異物を、機器に接触したり混入させないで下さい。
(10) 発熱器具の近くで使用、保管しないで下さい。
(11) 子供の手の届く場所で使用、保管しないで下さい。
(12) 人の生命や安全に係わる使用はしないで下さい。
(13) 電子部品及びリード線等に直接体で触らないで下さい。
(14) 外部電源用の端子を、ショートさせないで下さい。
(15) 本機に電源が入っている状態で配線を行わないで下さい。

■ 製品の保証範囲

(1) 本製品は、検査装置にて回路配線及び全機能(WDT/RESET 時間含む)検査後、出荷されます。
(2) 本製品の保証期間は、購入後 6 ヶ月です。
(3) 保証期間内における本製品の初期故障、自然故障による不具合に対しては、無償修理を行います。
但し、間違った使用(「お取扱い上の注意」に反する使用及び「製品仕様」を超えた使用等)、改造、盗難、火災、天災（地震、落雷、津波、噴火、風害、水害、ガス害、塩害、公害、地盤変動、地盤沈下）などの災害による故障については、保証の対象外とさせていただきます。
(4) 保証期間内のトラブルであっても、保証期間終了後にご相談された場合は、保証の対象外とさせてさせていただきます。
(5) 本製品をご使用することによる、又は、ご使用できなかったことによるお客様及び第三者に生じた損害について、弊社及び供給者は、その保証を免れるものとさせていただきます。

■ 製品サポートについて

故障修理については、センドバック方式で行わせて頂きます。事前に日時、内容等を弊社までご連絡して頂いてから、弊社出荷時と同等の梱包をしていただき返送して下さい。弊社への配送料は、お客様の負担とさせて頂きます。修理後に、送料弊社負担にてご返送させていただきます。但し、報告された現象が検査開始後 72 時間以内に再現されない場合は、原則としてお預かりしたままの状態でお返しいたします。
また、保証条件外のご使用による故障、保証期間後の故障については、修理可能な場合には、有償にて修理致します。

■ 製品内容 (説明書はインターネットよりダウンロードしてください)

(1) RaspberryPiBoard-KBRS31A 本体 ×1
(2) 保証書

■ おことわり

本製品の仕様は、将来予告無く変更する場合があります。

■ 説明書改定履歴

第 1.0 版 2014/12/20
第 1.01 版 2015/4/1 表 2 の GPIO ポート番号を PIN 番号と記述していた為、修正しました。

KARACRIX