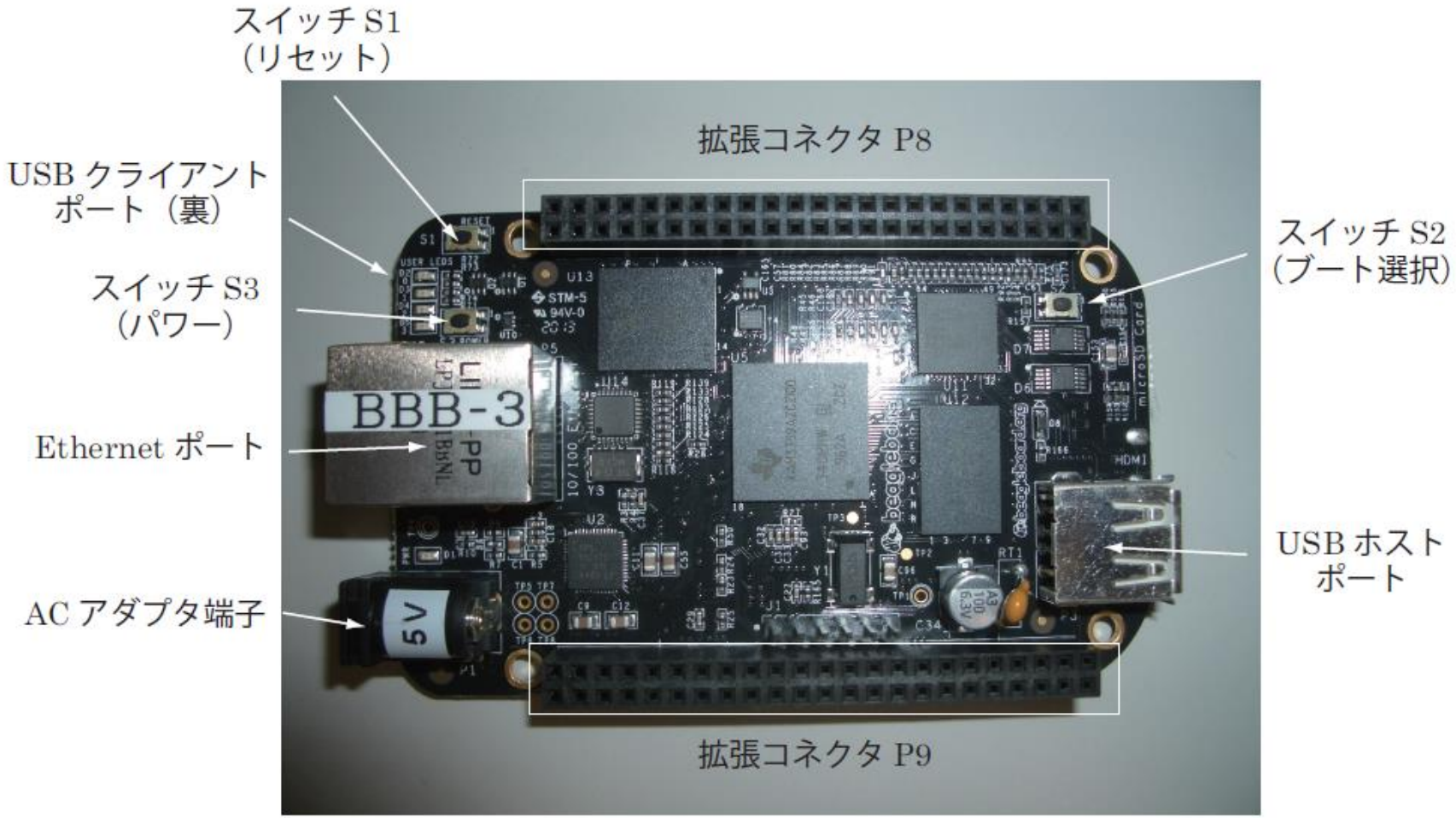
創造工学　勉強会－ 情報系－
BBBの使い方

以後　直樹

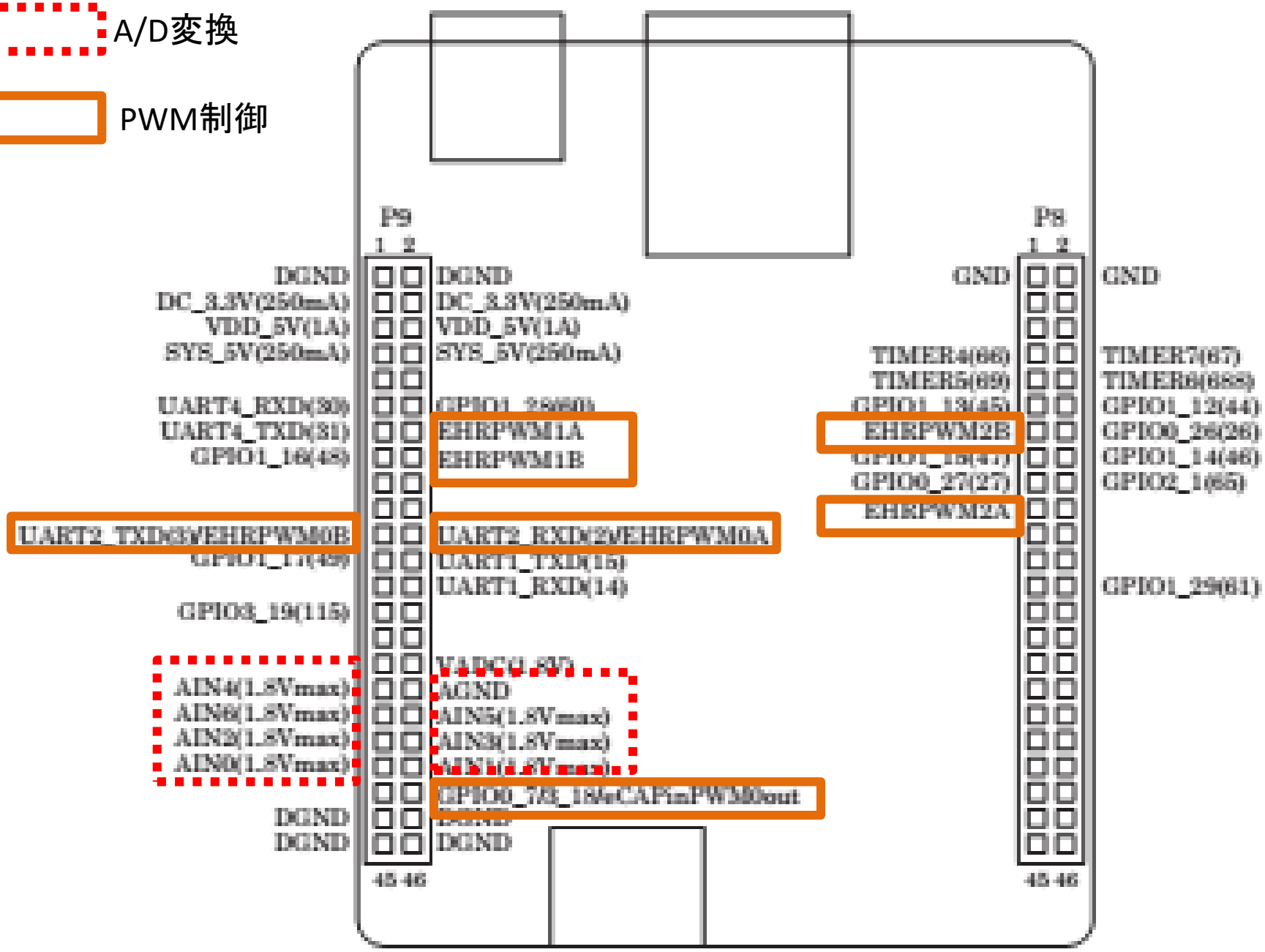
＜配布資料＞

- スライドのPDFファイル
- DCモータサンプルプログラム
- BBB用のWindows向けドライバ（64bitOS用）


スイッチ S1
（リセット）
USB クライアント
ポート（裏）
スイッチ S3
（パワー）
Ethernet ポート
AC アダプタ端子
拡張コネクタ P8
スイッチ S2
（ブート選択）
USB ホスト
ポート
拡張コネクタ P9

A/D変換

PWM制御


P9
1 2
DGND
DC_3.3V(250mA)
VDD_5V(1A)
SYS_5V(250mA)
UART4_RXD(30)
UART4_TXD(31)
GPIO1_16(48)
EHRPWM1A
EHRPWM1B
UART2_TXD(3)/EHRPWM0B
UART2_RXD(2)/EHRPWM0A
UART1_TXD(15)
UART1_RXD(14)
GPIO3_19(115)
AIN4(1.8Vmax)
AIN6(1.8Vmax)
AIN2(1.8Vmax)
AIN0(1.8Vmax)
AGND
AIN5(1.8Vmax)
AIN3(1.8Vmax)
DGND
DGND
45 46
P8
1 2
GND
TIMER4(66)
TIMER5(69)
EHRPWM2B
GPIO0_27(27)
EHRPWM2A
TIMER7(67)
TIMER8(68)
GPIO1_12(44)
GPIO0_26(26)
GPIO1_14(46)
GPIO2_1(65)
GPIO1_29(61)
45 46

P9コネクタのピン配置

GND	1	2	GND
DC_3.3V	3	4	DC_3.3V
VDD_5V	5	6	VDD_5V
SYS_5V	7	8	SYS_5V
PWR_BUT	9	10	SYS_RESETn
GPIO_30	11	12	GPIO_60
GPIO_31	13	14	GPIO_50(PWM)
GPIO_48	15	16	GPIO_51(PWM)
	17	18	
	19	20	
GPIO_3(PWM)	21	22	GPIO_2(PWM)
GPIO_49	23	24	GPIO_15
	25	26	GPIO_14
GPIO_115	27	28	
	29	30	
	31	32	VADC
AIN_4	33	34	AGND
AIN_6	35	36	AIN_5
AIN_2	37	38	AIN_3
AIN_0	39	40	AIN_1
GPIO_20	41	42	GPIO_7(PWM)
GND	43	44	GND
GND	45	46	GND

創造工学での使用法

- BBBと専用基板を合体させて使用する.
- 専用基板のコネクタにパラレルケーブルを接続することで，BBBの各ピンを使用する.
- 専用基板からBBBを抜くときは，ピンが曲がりやすいので注意

BBB専用基板　ピン配置

Beagle Bone Black 中継ボード　コネクタピン配置

2016.11.29

コネクタ1 pin NO.	Beagle Bone Black	pin NO.	コネクタ2 pin NO.	Beagle Bone Black	pin NO.	コネクタ3 pin NO.	Beagle Bone Black	pin NO.
1	GPIO1_28	P9 12	1	AIN4	P9 33	1	VDD_5V	P9 5
2	EHRPWM1A	P9 14	2	GNDA_ADC		2	GND	P9 1
3	SYS_5V	P9 8	3	AIN6	P9 35	3	SYS_5V	P9 7
4	EHRPWM0A	P9 22	4	GNDA_ADC		4	GND	
5	VDD_5V	P9 6	5	AIN2	P9 37	5	GPIO0_30	P9 12
6	VDD_5V	P9 6	6	GNDA_ADC		6	GND	
7	EHRPWM2B	P8 13	7	GPIO0_20	P9 41	7	GPIO1_16	P9 15
8			8	GNDA_ADC		8	GND	
9	GPIO0_7	P9 42	9	DGND	P9 46	9	GPIO0_3	P9 21
10	GND		10	GNDA_ADC		10	GND	
11	GPIO1_29	P8 26	11	GNDA_ADC	P9 40	11	GPIO1_17	P9 23
12	GND		12	GNDA_ADC		12	GND	
13	GPIO2_1	P8 18	13	VADC	P9 32	13	GPIO3_19	P9 27
14	GND		14	GND		14	GND	
15	GPIO1_14	P8 16	15	GPIO3_16	P9 30	15	GPIO0_27	P8 17
16	GND		16	GND		16	GND	
17	GPIO0_26	P8 14	17	GPIO0_14	P9 26	17	GPIO1_15	P8 15
18	GND		18			18	GND	
19	GPIO1_12	P8 12	19	GPIO0_15	P9 24	19	GPIO1_13	P8 11
20	GND		20	VDD_5V	P9 6	20	GND	
	コネクタ　con-harting-h 20H							

モータ系統　　AD変換・GPIO系統　　GPIO系統

＊ピン配置の詳細は回路班に確認

GPIO番号へ変換

ピン配置には，「GPIO▲_数字」と記載

ここから，GPIO番号を以下の式で計算

GPIO番号＝32×▲＋数字

BBBの入出力電圧

- DC_3.3V　電源：許容電流　250[mA]
- VDD_5V　電源：電池ボックスと直結
- SYS_5V　電源：許容電流　250[mA]
- GPIO：MAX19チャンネル、電圧　3.3[V]
- PWM：3チャンネル、電圧　3.3[V]
- AIN：3チャンネル、入力最大電圧　1.8[V]
  ⇒1.8[V]以上の時は、分圧が必要
  ⇒分圧回路は回路班と要相談

BBBの使用する機能

- GPIO：超音波センサ，「0」or「1」の入出力
  DCモータの速度制御をしない場合

- PWM：DCモータの速度制御，
  サーボモータの制御

- AIN（A/D変換）：測距センサ(赤外線距離)

GPIO

① 使用するピンを選択

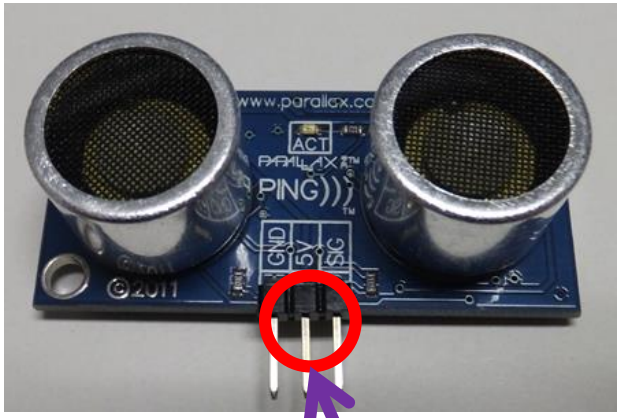
② GPIOの有効化
「/sys/class/gpio/export」にGPIO番号を書込み
GPIO番号：ピン配置のP×_■の■に相当

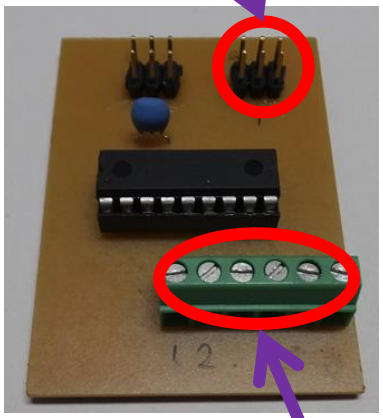
③ GPIOの初期化
「/sys/class/gpio/gpio■/direction」
⇒「in」もしくは「out」を書き込む
■：GPIO番号(ピン番号ではないので注意)

④ GPIOの入出力
「/sys/class/gpio/gpio■/value」
「1」を書込み

⑤ GPIOの無効化
「/sys/class/gpio/unexport」にGPIO番号を書込み

超音波センサ





BBB

測定範囲：3[cm]～3.3[m]
＊カタログスペック

GPIOの入力機能を使用
⇒ON(1)になっているパルスの時間を計測
⇒ONになっている時間から距離へ変換

コンパイル時は，リアルタイム拡張のオプション「-lrt」を付けること！！

詳しい詳細は，工学実験を思い出して下さい．

ライントレーサー　その1

- 迷路上の白いテープと黒の地面を判別することが可能

- 白いテープに沿うように移動することで，迷路を進むことが可能

- GPIOを使用．ただし，ライントレーサーからの出力は分圧する必要あり．

  ⇒分圧回路は回路班に作成依頼しよう

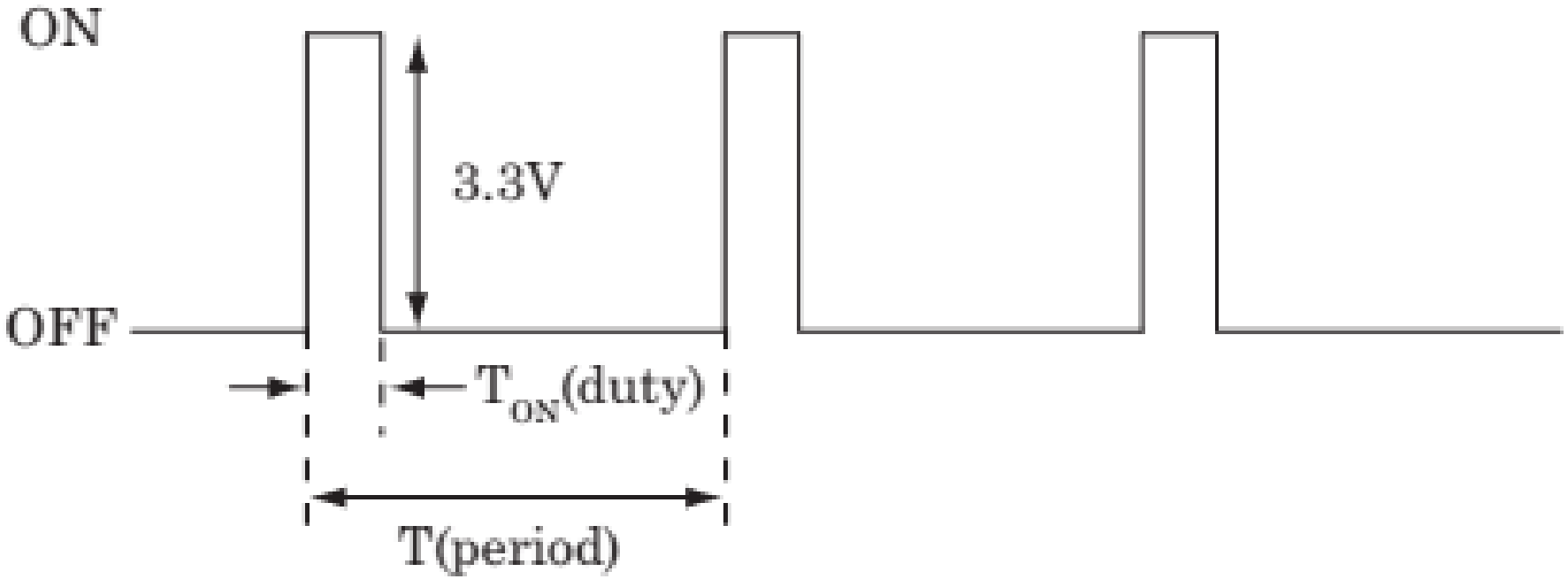
ライントレーサー　その2

- データの読み込みには，「read（）関数」を使用

(例)

```
//gpio_numberは事前に「in」でオープンしておく
char c;
fd = gpio_open(gpio_number, "value", O_RDONLY);
read(fd, &c, 1);
close(fd);
```

PWM



period：周期[ns]
duty：ONの時間[ns]、プログラム実行中に変更可

PWM

① PWMの有効化
「/sys/devices/bone_capemgr.◆/slots」
「am33xx_pwm」を書込み

```
fp = fopen("/sys/devices/bone_capemgr.◆/slots","w");
fprintf(fp, "am33xx_pwm");
```

② ピンの設定
「/sys/devices/bone_capemgr.◆/slots」
「bone_pwm_ポート番号_ピン番号」を書込み

```
sprintf(path, "bone_pwm_%s ",PIN_PWM);
fp = fopen("/sys/devices/bone_capemgr.◆/slots", "w");
fprintf(fp, path);
```

↓

「/sys/devices/ocp.■/pwm_test_ポート番号_ピン番号.●」のディレクトリが作成
ポート番号:「P9」、「P8」　●:有効化順で変化

PWM

③ PWMの周期設定(単位[ns])
「/sys/devices/ocp.■/pwm_test_ポート番号_ピン番号.●/period」

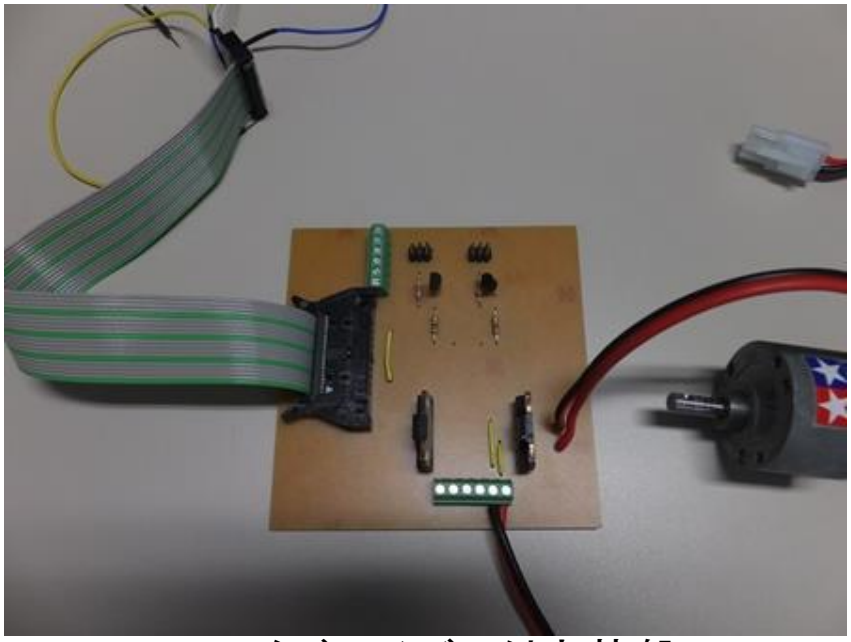
④ PWMのON時間設定(単位[ns])
「/sys/devices/ocp.■/pwm_test_ポート番号_ピン番号.●/duty」

⑤ ピンの設定
「/sys/devices/ocp.■/ _ポート番号_ピン番号.● /run」
出力「1」、停止「0」を書込み

DCモータ

- 駆動電源は，青いバッテリーから7.2[V]供給
- BBBからのPWM指令をモータドライバICを通して，モータへ入力
- PWM制御の「duty」をセンサ情報から変化させることで，速度制御等を行う.
- 2輪駆動の場合，同じ「duty」でも，直進せずに曲がる場合もあるので，調整が必要.
  ⇒モータの個体差や指令信号がでるタイミングの違いが影響

DCモータ



モータドライバIC付き基盤

GPIO×2＋PWM×1

IN1←GPIO信号

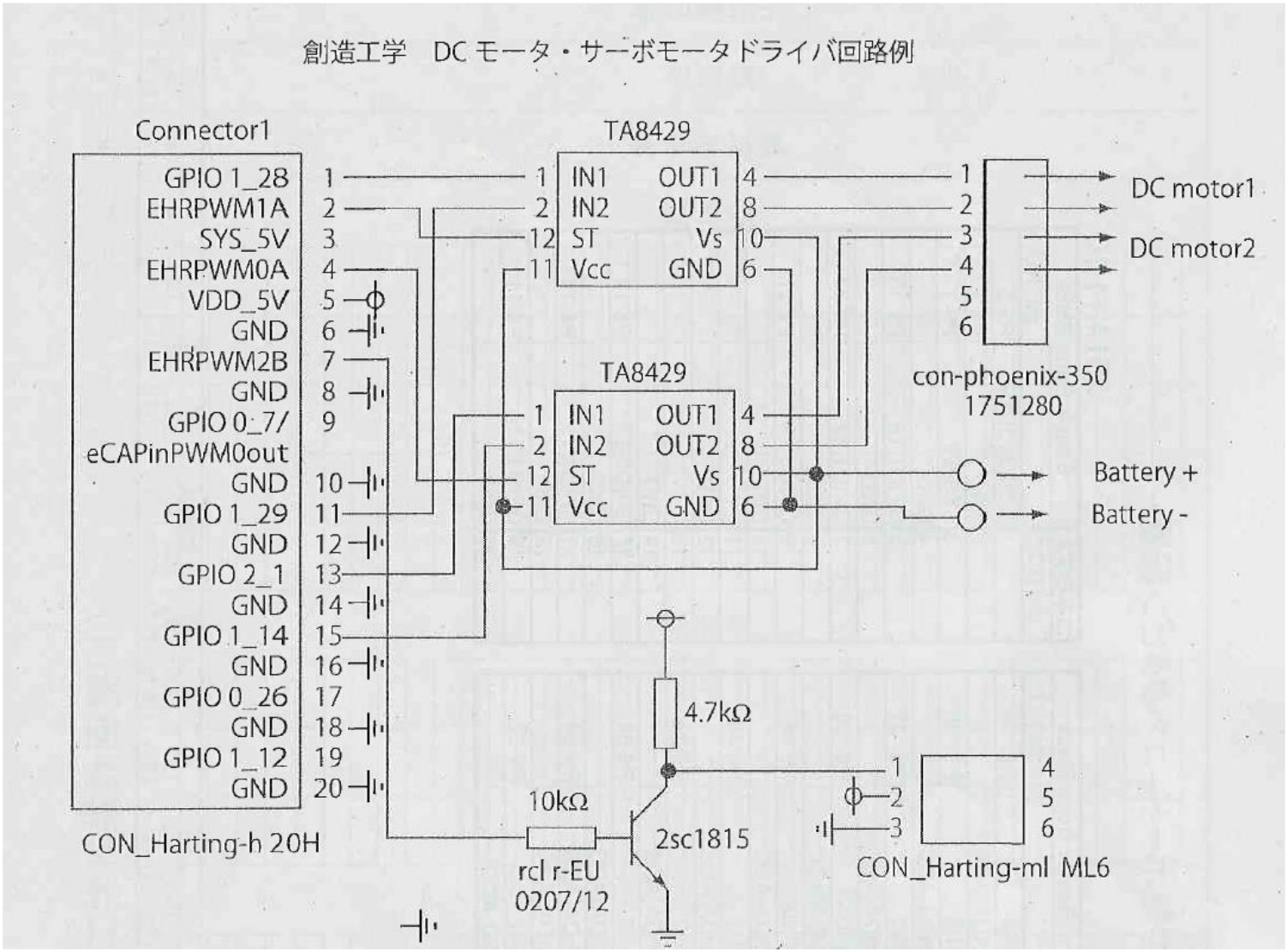
IN2←GPIO信号

ST←PWM信号

モータドライバICの入出力関係(TA8429H)

入力			出力		出力モード
IN1	IN2	ST	OUT1	OUT2	
H	H	H	L	L	ブレーキ
L	**H**	**H**	**L**	**H**	**逆転(正転)**
H	**L**	**H**	**H**	**L**	**正転(逆転)**
L	L	H	OFF(ハイ　インピーダンス)		ストップ
H/L	H/L	L	OFF(ハイ　インピーダンス)		スタンバイ

BBBとDCモータ接続回路



DCモータサンプルプログラム1

- 今年は，工学実験の様子を見ていると，厳しいように思えたので，モータを動かす関数を作ってきました.

- 「4S_sample_motor.c」

- 動作チェックはしてないので，もしかするとバグがあると思います.

DCモータサンプルプログラム2

（関数の内容：工学実験で使用）

```
//gpioの有効化関数
void gpio_export(int n);

 //gpioの有効化解除の関数
 void gpio_unexport(int n);

 //gpioの設定ファイルを開く関数
 int gpio_open(int n, char *file, int flag);

//キー入力関数
int kbhit(void);
```

DCモータサンプルプログラム3

（関数の内容：新規追加）

//PWM初期化関数

void init_pwm(int motor_num);

モータの番号：0～

//モータPWM出力関数

void run_pwm(int motor_num,int duty,int drive_mode);

速さを決める
実験と同じ

0：停止
1：正転
-1：逆転

//PWM終了関数

void close_pwm(int motor_num);

DCモータサンプルプログラム4

（使い方）

① 初期設定のために，init_pwm関数をモータ個数分呼び出す

② run_pwm関数を呼び出すとPWMの出力開始する．ループ中でdutyを変更すれば，速度変化可，drive_modeで回転方向指定

③ close_pwmをモータ個数分呼び出し終了

サーボモータ

- 信号線に入力される周期的なパルス幅に応じた角度を出力する.

- 周期的なパルスをBBBのPWM信号(3.3[V])を用いて，モータに入力する．BBBの3.3[V]だと足りない可能性があるので，昇圧回路(回路班のモータ基板に実装予定)を挟み(5.0[V])とする.

サーボモータ

- 使用モータ：HITEC HS-635HB　　()あり：6.0[V]
- トルク：6.4[kg･cm](7.8[kg･cm])　()なし：4.8[V]
- 周期(period)：1500～3000[μs]
- 40[deg]/400[μs]

- PWMの信号をサーボモータの信号線に入れる.
  ⇒5.0[V]に昇圧する. 回路は回路班が作成予定
- 電源は，電池BOXから取る予定.

アナログ入力（AIN）

- アナログ入力をデジタルへ変換する時に使用
- 入力最大電圧　1.8[V]
  ⇒これ以上の入力は分圧回路が必要.
  ⇒故障します. 8000円？が一瞬で飛びます.

アナログ入力（AIN）

① A/D変換の有効化
「/sys/devices/bone_capemgr.●/slots」に
「cape-bone-iio」を書き込む

```
fp=fopen(“/sys/devices/bone_capemgr.●/slots","w");
fprintf(fp,"cape-bone-iio");
```

② 入力ファイルのパス設定
「/sys/devices/ocp.▲/helper.■」に7つのデバイスファイルが生成される（AIN0～AIN6）

```
sprintf(path,“/sys/devices/ocp.▲/helper.■/AIN%d",AIN_NUMBER);
```

アナログ入力（AIN）

③ A/D変換の実行

②で指定したデバイスファイルから，変換値を取得する．0～1800[mV]の整数値で取得可能．安全のために，2回連続で読込，2回目の値を採用すること

```
fp=fopen(path,“rb”);
fscanf(fp,“%d”,&value_voltage);
fscanf(fp,“%d”,&value_voltage);   ←こちらのデータ
```

＜バックグラウンド起動＞

nohup　./実行ファイル名　&

＜バックグラウンドプログラム終了＞

kill　プロセスID

＜プロセスID確認＞

ps　-x

BBBのLEDは光っているが，SSHでログインできない場合，OSを再イントールする直る可能性があるので，以後のところまでBBBを持って来て下さい.

USBを差してもBBBのLEDが光らない場合は，本体が故障している可能性が高いです.

そうなると，直すことが難しくなりますので，取り扱には，ご注意ください.