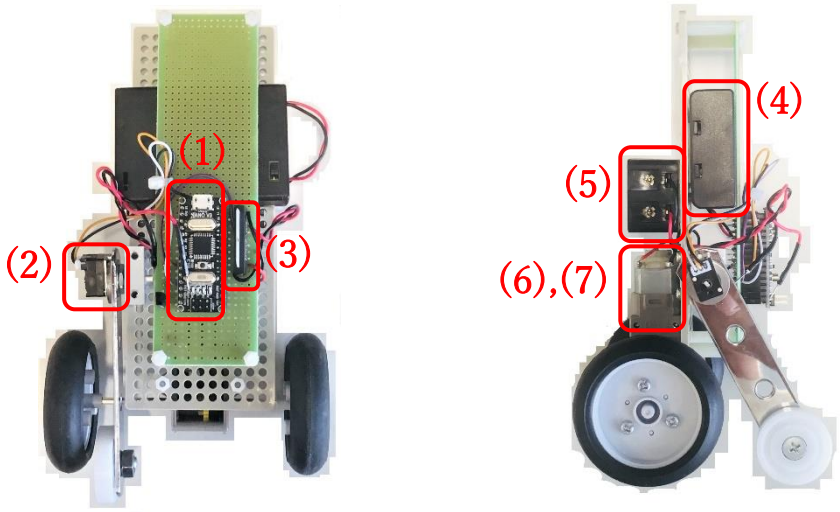
今回製作した倒立振子のハードウェアとその制御プログラムについて説明する．

1. ハードウェアについて

今回製作した倒立振子は，図 1 に示す車輪型倒立振子である．今回は，パラメータ調整を主に行っていくことから，プログラムを書き込みやすく，パソコンへのデータロギングも可能な Arduino Nano を使用している．支給されたモータのギア比が低速 203.7：1，中速 58.2：1，高速 16.6：1 である．倒立振子を製作する上でモータが頻繁に正転・逆転するため，負荷トルクが小さいギア比，高速 16.6：1 を利用した．



(a)前面　　(b)側面

図 1　実機全体図

使用した主な部品を表 1 に示す．

表 1　主な使用部品

No	部品名	個数
(1)	Arduino Nano	1
(2)	ポテンショメータ	1
(3)	モータドライバ IC	1
(4)	電池 1.5V	3
(5)	電池 9V	1
(6)	モータ	1
(7)	ギアボックス	1

回路図を図 2 に示す．Arduino Nano はプロセッサに ATMEGA328P を搭載しており，PWM ピンは D3，D5，D6，D9，D10，D11 であるため 6 ピンを利用した．モータドライバである TA7291P の制御用ピンには D7，D8 を利用した．ポテンシャメータから角度に応じた電圧を取

得するために，アナログピンの A5 を利用した．

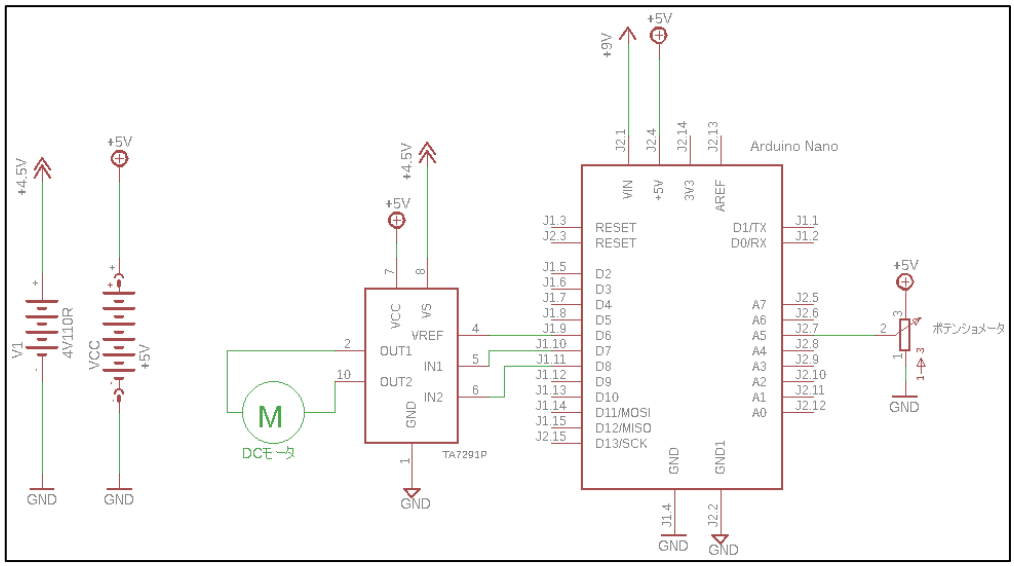


図 2　倒立振子の回路図

2.　製作プログラムとその解説

今回制作した倒立振子は PID 制御で制御を行った．PID 制御は P:比例，I:積分，D:微分の 3 つの補償動作から成り立っている．P 制御は現在の偏差に比例した修正量，I 制御は過去の偏差の累積値に比例した修正量，D 制御は偏差の増減の傾向に比例した修正量を出す．この PID 制御に基づいて作成したプログラムを表 2 に示す．

表 2　組み込みプログラム

1	#include <TimeOne.h>
2	int i;
3	volatile int pot_target=0;
4	volatile int pot_current=0;
5	volatile double epsilon_sum=0;
6	volatile double old_epsilon=0;
7	volatile double epsilon;
8	volatile int output,pid_mv;
9	const double Kp=17.0;
10	const double Ki=0.01;
11	const double Kd=13.5;
12	void setup(){
13	pinMode(6,OUTPUT);
14	pinMode(7,OUTPUT);
15	pinMode(8,OUTPUT);
16	delay(1000);

```
17    for(i=1;i<100;i++) pot_target +=analogRead(A5);
18    pot_target=pot_target/i;
19    Timer1.initialize();
20    Timer1.attachInterrupt(pid_ctl,1000);
21  }
22  void loop(){
23    if(output>=0){
24      analogWrite(6,output);
25      digitalWrite(7,HIGH);
26      digitalWrite(8,LOW);
27    } else{
28      analogWrite(6,-output);
29      digitalWrite(7,LOW);
30      digitalWrite(8,HIGH); }
31  }
32  void pid_ctl(){
33    pot_current=analogRead(A5);
34    epsilon=(double)pot_target-pot_current; //偏差を計算
35    epsilon_sum +=epsilon;  //偏差を積分
36    P=Kp*epsilon;
37    I=Ki*epsilon_sum;
38    D=Kd*(epsilon-old_epsilon);
39    pid_mv=P+I+D;
40    output=constrain(pid_mv,-255,255);  //-255~255 の範囲に限定
41    old_epsilon=epsilon;
42  }
```

作成したプログラムは，Arduino IDE によって，コンパイルする．そのため，図 3 のようなフローチャートとなる．

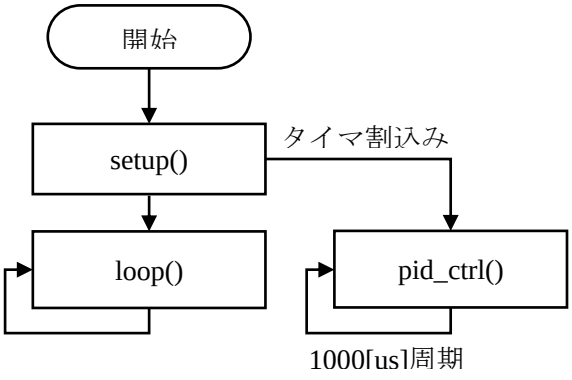


図 3　フローチャート

各関数の動作について説明する．

まず各変数(2~11 行目)の役割としては表 3 の通りである．

表 3　各変数の役割

変数修飾子	変数名	型名	役割
	int	i	平均回数の指定
volatile	int	pot_target	目標電圧
volatile	int	pot_current	現在の電圧
volatile	double	epsilon_sum	偏差積分値
volatile	double	old_epsilon	前回の偏差
volatile	double	epsilon	偏差
volatile	int	pid_mv	PID 制御の操作量
volatile	int	output	PWM 出力値
const	double	Kp	比例ゲイン
const	double	Ki	積分ゲイン
const	double	Kd	微分ゲイン

次に各関数について説明する．

・setup 関数　　12~21 行目

この関数は，Arduino Nano 起動時 1 回のみ呼び出される関数で，倒立振子を動作させるための初期設定を実行する関数である．DC モータを制御するための出力ピン，D7，D8 と PWM 制御するためのピン，D6 の初期化を行った(13~15 行目)．ポテンシャメータから電圧値で倒立振子を制御するため，倒立させる基準となる目標電圧を設定する．analogRead()関数でアナログピンからアナログデータを取得できるので，A5 ピンから電圧を取得し，100 回平均を目標電圧とした，またタイマ割込み処理を行うため，タイマ割込みを初期化(19 行目)し，1000[us]周期で pid_ctl 関数を実行するようにタイマを設定した(20 行目)．

・loop 関数　　22~31 行目

この関数は，モータドライバへ出力を与える関数である．後述する pid_ctl 関数から算出された PWM 値 output から正であれば 7 ピンを HIGH，8 ピンを LOW にする．output が負であれば 7 ピンを LOW，8 ピンを HIGH にする．またモータドライバへ PWM 出力を与えるために，analogWrite 関数を用いて，output の絶対値を 6 ピンから出力した．この処理を起動中，割り込まれながら繰り返す．なお PWM は 0~255 までが出力範囲であるため，output が負の場合は正に変換して出力させないと，予期せぬ動作が発生する可能性があるため，「-output」と記述した．

・pid_ctl 関数　　32~42 行目

この関数は PID 制御をする関数である．目標電圧と現在の電圧の差を偏差とした．9~11 行目で定数化された P，I，D 項のパラメータである比例ゲイン Kp，積分ゲイン Ki，微分ゲイン Kd はトライ

アンドエラーで調整し，Kp=17.0，Ki=0.01，Kd=13.5 と決定した．PID の操作量 pid_mv は先ほど設定した各ゲインから P，I，D 項をそれぞれ算出(36~38 行目)する各ゲインと項から PID の制御式は式(1)のようになる(39 行目)．

$$pid\_mv = 17.0 * epsilon + 0.01 * epsilon\_sum + 13.5 * (epsilon - old\_epsilon) \quad (1)$$

PID 制御で求めた操作量を PWM として出力し，モータを制御する． 操作量は偏差や偏差の累積等によって正負の増減が決定する．しかしモータ制御する PWM は 0~255 までしか出力できないため，pid_mv を PWM が出力できる範囲まで限定する必要がある．範囲を指定するために constrain 関数で値の範囲指定をした(40 行目)．第 1 引数は変換対象の変数，第 2 引数は範囲の下限値，第 3 引数は範囲の上限値である．constrain 関数で操作量が 255 を上回る場合は出力値を 255 に，-255 を下回る場合は-255 に変換した．最後に D 制御で過去の偏差を用いるため，今割込み処理で算出した偏差を代入した(42 行目)．

以上で，課題を満足する倒立振子を製作した．