

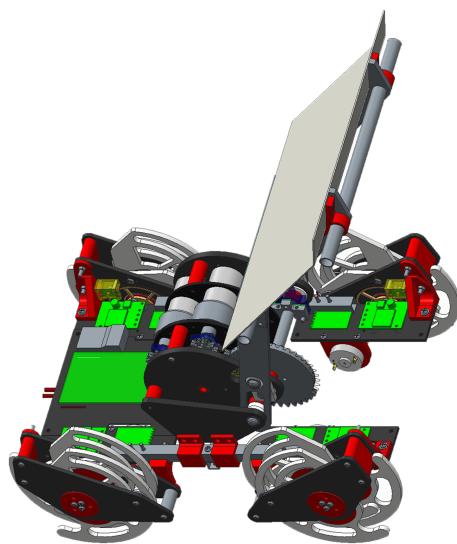
でしふろノート

Vol.06

Deshi Prompt 2018.8.10

Matlab/Simulink と Arduino で モータ回転数制御

Ver 1.0



でし・ふろんふと

<http://deshi-prompt.github.io/>

目次

第 1 章	はじめに	1
1.1	本誌 概要	1
1.2	かわさきロボット競技大会の概要	1
1.3	ロボット操縦の課題	2
1.4	Matlab/Simulink 開発環境構築	3
第 2 章	Matlab/Simulink と Arduino でモータを動かそう	6
2.1	ハード構成	6
2.2	回路図	6
2.3	Simulink モデル	7
2.4	実行	9
第 3 章	Matlab/Simulink と Arduino でモータ回転数を計測しよう	10
3.1	ハード構成	10
3.2	回路図	11
3.3	Simulink モデル	11
3.4	実行	16
第 4 章	Matlab/Simulink と Arduino でモータ回転数制御をしよう	17
4.1	ハード構成	17
4.2	PID 制御モデル	17
4.3	実行	19
4.4	ロボットへの実装例	19
参考文献		22
奥付		23

第1章

はじめに

1.1 本誌 概要

本誌ではロボットハードウェア製作において、モータ回転数制御の実現方法を記載しています。具体的な題材として、毎年川崎市で開催されている「かわさきロボット競技大会」の機体製作を取り上げます。

1.2 かわさきロボット競技大会の概要

かわさきロボット競技大会では、リング上で2台のロボットを戦わせます。勝敗は審判の判定によって決定されます。勝利条件の概略は下記の通りです。(大会HP[1]を参照)

- ラウンド内にリングの場外へ相手機体を押し出す
- ラウンド内にリング上で相手機体をダウン（走行不可能な状態）させ、
10カウントダウン状態をキープする

製作する機体の概略は下記の通りです。詳細はかわさきロボット競技大会公式サイト [1] に記載されています。参考までに筆者が過去に製作した機体は Fig.1.1 です。

- サイズ制限：全幅 250mm × 全長 350mm × 高さ 700mm（スタート時）
- 重量制限：3300g 以内
- 操縦は、競技大会の実行委員会が規定するプロポを用いること
- ロボットには、それぞれ1セット以上の脚機構、アーム機構が搭載されていること
- ロボットは走行用の脚部と相手機体への攻撃用のアーム部を有する

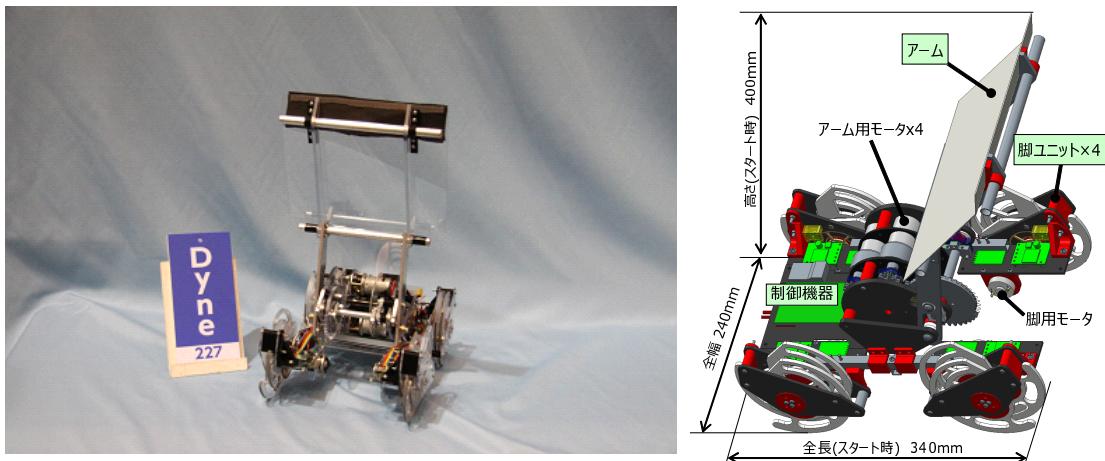


Fig.1.1 筆者のかわさきロボット出場機体

1.3 ロボット操縦の課題

製作したロボットは例えば左右の脚ユニットの負荷ばらつき、モータの性能ばらつき等で同じ操作量を入れても Fig.1.2 のように思った方向に進まない場合があります。

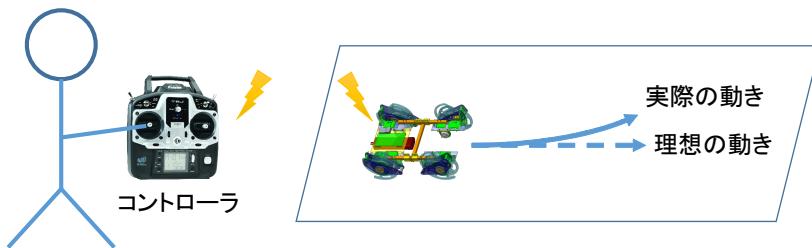


Fig.1.2 ロボットの動き

そこで、Fig.1.3 のようにモータの回転数に対してフィードバック制御を入れ、左右の回転数差を無くします。



Fig.1.3 フィードバック制御

ロボット制御システムのハードウェア構成例として、Fig.1.4の通りです。今回、制御プログラムを Matlab/Simulink 上で構築して Arduino で動かすことを目指します。

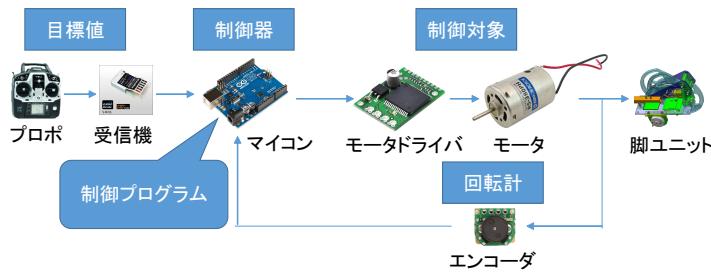


Fig.1.4 ロボット制御システムのハードウェア構成例

1.4 Matlab/Simulink 開発環境構築

本書執筆時の動作環境を下記に示します。

- OS : Windows 10 64bit
- Matlab/Simulink : R2018a
- Arduino : Mega 2560 R3 または Uno R3

1.4.1 Matlab/Simulink のインストール

個人ライセンス製品の MATLAB Home [2] を使います。

2018年8月時では、MATLAB本体は¥16,500、Simulinkは¥4,990です。

1.4.2 Simulink での Arduino 動作環境をインストール

Matlab起動時のメニューにある”アドオン” ⇒ ”ハードウェアサポートパッケージの入手”をクリックします。アドオン エクスプローラーで、Simulink Support Package for Arduino Hardware (Fig.1.5) を選択してインストールします。

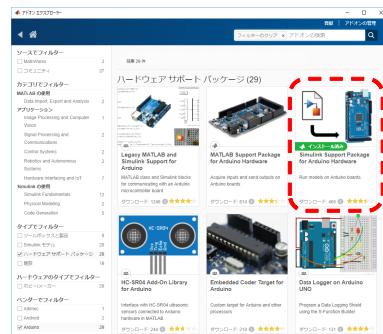


Fig.1.5 アドオン エクスプローラー

1.4.3 C 言語コンパイラのインストール

第3章に記載している S-Function Builder ブロックを使用するためには C 言語コンパイラが必要です。

C 言語コンパイラ有無の確認は、Matlab コマンドウインドウに下記を入力して実行します。

```
mex -setup
```

C 言語コンパイラがすでにインストールされていれば Fig.1.7 のような表示が出ます。

C 言語コンパイラがインストールされていなければ、インストールが必要です。

今回、C 言語コンパイラとして、フリーで入手できる Microsoft Windows SDK 7.1 を使用します。

ただし、Microsoft Windows SDK 7.1 は最新 OS だと通常のやり方ではインストールできません [4] [5]。

Microsoft Windows SDK 7.1 インストール手順のポイントを次に示します。

i. Visual C++ 2010 のランタイムをアンインストールする

下記プログラムが既にインストールされていると、SDK のインストールに失敗するので”アプリと機能” からアンインストールします。

- Microsoft Visual C++ 2010 x86 Redistributable
- Microsoft Visual C++ 2010 x64 Redistributable

ii. winsdk-web.exe のインストール

下記ホームページからインストーラをダウンロードして、インストールします。

<http://www.microsoft.com/en-us/download/details.aspx?id=8279>

インストール設定画面において、Microsoft Visual C++2010 のチェックを外してください (Fig.1.6)。

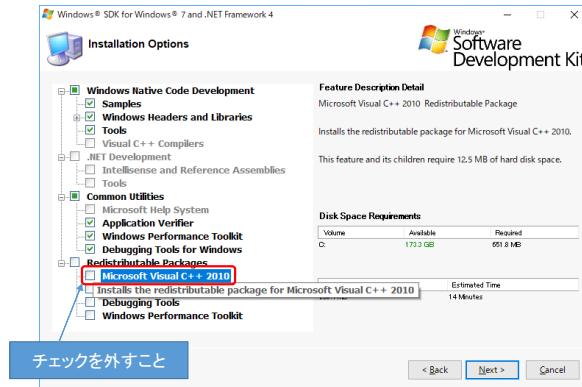


Fig.1.6 SDK 7.1 インストール設定画面

iii. VC-Compiler-KB2519277.exe のインストール

下記ホームページからインストーラをダウンロードして、インストールします。

<http://www.microsoft.com/en-us/download/details.aspx?id=4422>

iv. 正しくインストールされたか確認

Matlab コマンドウインドウに mex -setup を実行して、Fig.1.7の表示がでればインストールは完了です。

```
コマンドウインドウ
>> mex -setup
MEX は C 言語のコンパイルに 'Microsoft Windows SDK 7.1 (C)' を使用するよう設定されています。
警告: MATLAB C および Fortran API は、2^32-1 を超える要素のある MATLAB
変数をサポートするように変更されました。今後、新しい API を
利用するためのコードの更新が必要になります。
詳細については、次を参照してください:
https://www.mathworks.com/help/matlab/matlab\_external/upgrading-mex-files-to-use-64-bit-api.html。

別の言語を選択するには、次のいずれかを選択してください。
mex -setup C++
mex -setup FORTRAN
f> >>
```

Fig.1.7 Matlab コマンドウインドウ

第 2 章

Matlab/Simulink と Arduino でモータを動かそう

2.1 ハード構成

本章では、かわさきロボット大会指定の 380 モータではなく、手っ取り早くモータ制御を実現するために、市販されているハード (Fig.2.1) で構成しました。全てのハードは通販サイト、例えばスイッチサイエンス [6] から購入できます。既存のハードでモータ制御を学んだのちに、ロボット本体へ適用します。

- マイコン : Arduino Mega 2560 R3
- モータドライバ : VNH5019 搭載モータードライバ (POLOLU-1451)
- モータ : 75:1 シャフト付き超小型メタルギアドモーター HP (POLOLU-2215)

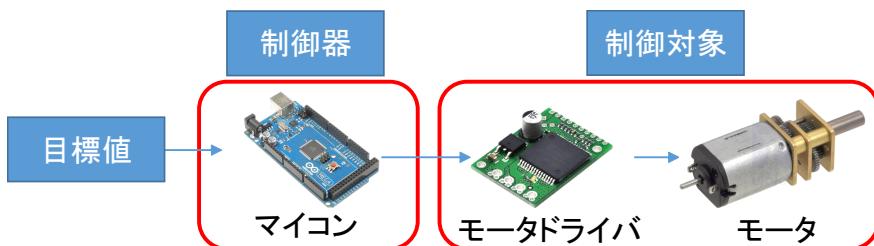


Fig.2.1 ハード構成

2.2 回路図

回路図は、Fig.2.2の通りです。Arduino Mega 2560 のピン 5~7 を使用してモータドライバを動かします。モータドライバ VNH5019 の詳細は、Pololu HP [7] に記載されています。

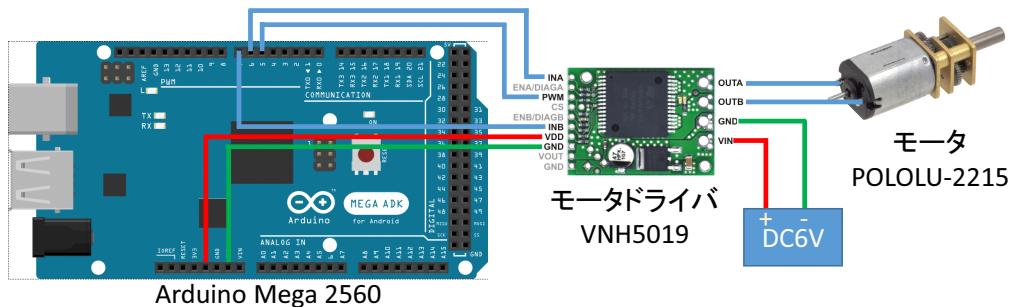


Fig.2.2 回路図

2.3 Simulink モデル

Simulink モデルは、Fig.2.3の通りです。Arduino を Simulink 上で実行できるように、シミュレーションモードを ”エクスターナル” に、シミュレーション終了時間を ”inf” に変更します。

コンフィギュレーション パラメーターを開き、ハードウェア実行 ⇒ ハードウェア ボードを ”Arduino Mega 2560” にします (Fig.2.4)。

指令値をモータドライバの制御信号に変換するために、MATLAB Function ブロックを使用します。ブロックの中身は、Fig.2.5の通りです。プロポからラジコン信号入力を想定して、1500usec 付近でモータ停止、1150～1850usec の間で速度変更する仕様とします。

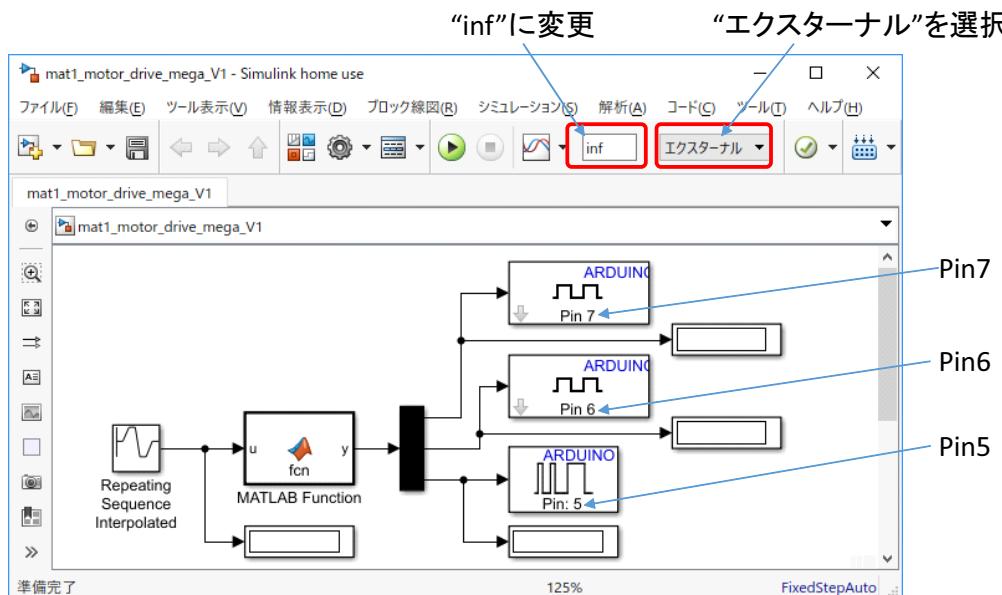


Fig.2.3 Simulink モデル



Fig.2.4 コンフィギュレーション パラメーター

```

エディター - Block: mat1_motor_drive_mega_V1/MATLAB Function
MATLAB Function + 
function y = fcn(u)
if(u<700) %STOP
    INA1=0;
    INB1=0;
    PWM1=0;
elseif(u<1150) %CW, MAX SPEED
    INA1=1;
    INB1=0;
    PWM1=255;
elseif(u<1450) %CW, Speed control
    INA1=1;
    INB1=0;
    PWM1=(1450-u)/300*255;
elseif(u<1550) %STOP
    INA1=0;
    INB1=0;
    PWM1=0; %Free
    %PWM1=255; %Brake
elseif(u<1850) %CCW, Speed control
    INA1=0;
    INB1=1;
    PWM1=(u-1550)/300*255;
elseif(u<2300) %CCW, MAX SPEED
    INA1=0;
    INB1=1;
    PWM1=255;
else %STOP
    INA1=0;
    INB1=0;
    PWM1=0;
end
y = [INA1;INB1;PWM1];

```

Fig.2.5 MATLAB Function ブロックの中身

2.4 実行

Fig.2.6に示すボタンをクリックすることでプログラムを実行できます。

Simulink と連動して動作させる場合は”実行”をクリックします。ビルドに成功して動作し始めると、Display ブロックに出力値がリアルタイムに表示されます。また、入力値として使用した Repeating Sequence Interpolated ブロックをダブルクリックし、パラメータを変更することで、実行しながらもパラメータ変更の挙動が確認できます。

Simulink と連動させず、Arduino をスタンドアロンで動かす場合は”ハードウェアに展開”をクリックします。一度書き込めば、Simulink と接続しなくてもプログラムが動くようになります。

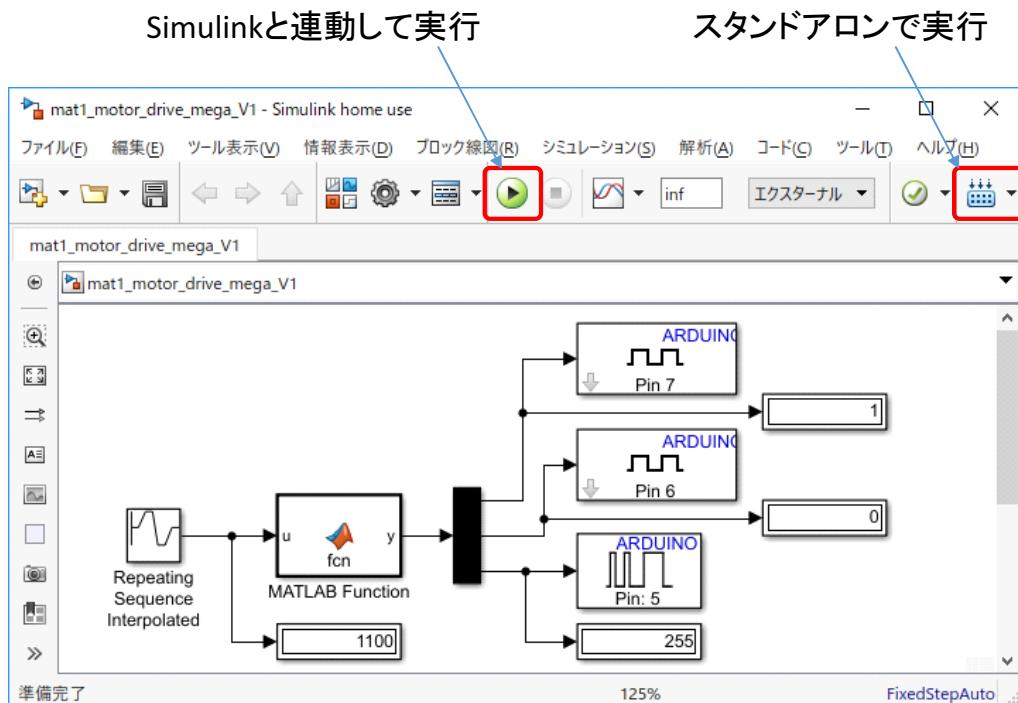


Fig.2.6 Matlab/Simulink モデルの実行

第3章

Matlab/Simulink と Arduino でモータ回転数を計測しよう

3.1 ハード構成

ハード構成は Fig.3.1の通りです。Fig.2.1よりエンコーダを追加することで、モータの回転数を取得します。

- マイコン : Arduino Mega 2560 R3
- モータドライバ : VNH5019 搭載モータードライバ (POLOLU-1451)
- モータ : 75:1 シャフト付き超小型メタルギアドモーター HP (POLOLU-2215)
- エンコーダ: シャフト付き超小型メタルギアドモーター用磁気式エンコーダ (POLOLU-3081)

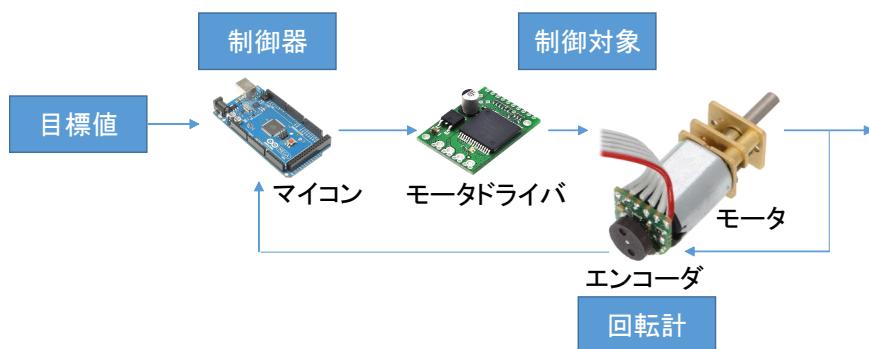


Fig.3.1 ハード構成

3.2 回路図

回路図は Fig.3.2 の通りです。Arduino Mega 2560 のピン 2~3 を使用してエンコーダ信号を読み取ります。エンコーダの詳細は、Pololu HP [8] に記載されています。

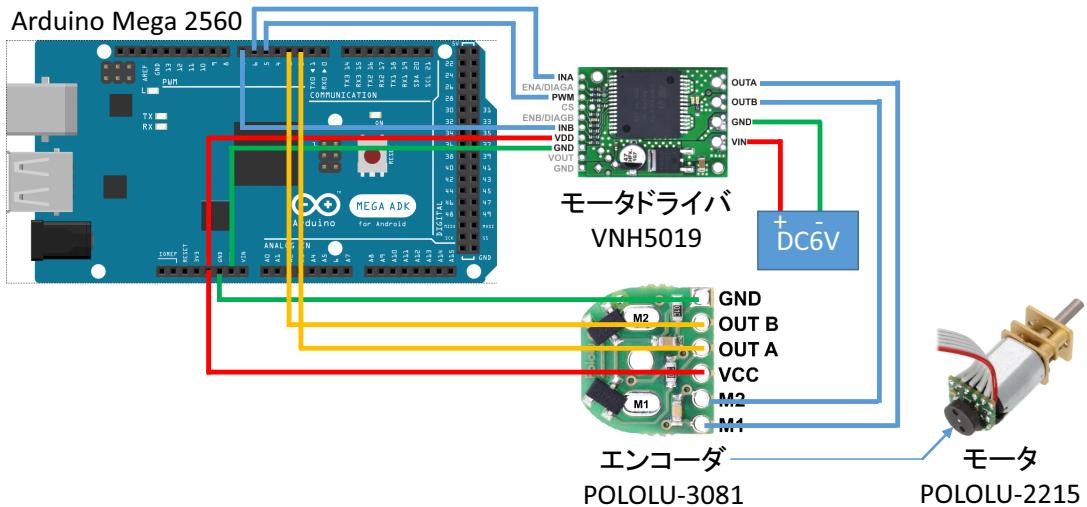


Fig.3.2 回路図

3.3 Simulink モデル

Simulink モデルは、Fig.3.3 の通りです。エンコーダの出力信号を読み取るために、S-Function Builder ブロックを使用します。ブロックの中身は、Fig.3.4～Fig.3.7 の通りです。それぞれのタブに設定を入れ、“ライブラリ”タブの“インクルード”、“出力”タブ、“更新”タブに Fig.3.8～Fig.3.10 に示すプログラムを書き込んで、“ビルド”を押すことでコンパイルができます。

プログラムは “離散状態の更新” でエンコーダに接続した割込みピンの設定を行い、“インクルード” で呼び出し関数の定義を行い、“出力” でプログラムをループさせ、エンコーダ回転数 [RPM] を S-Function Builder ブロックから出力させます。

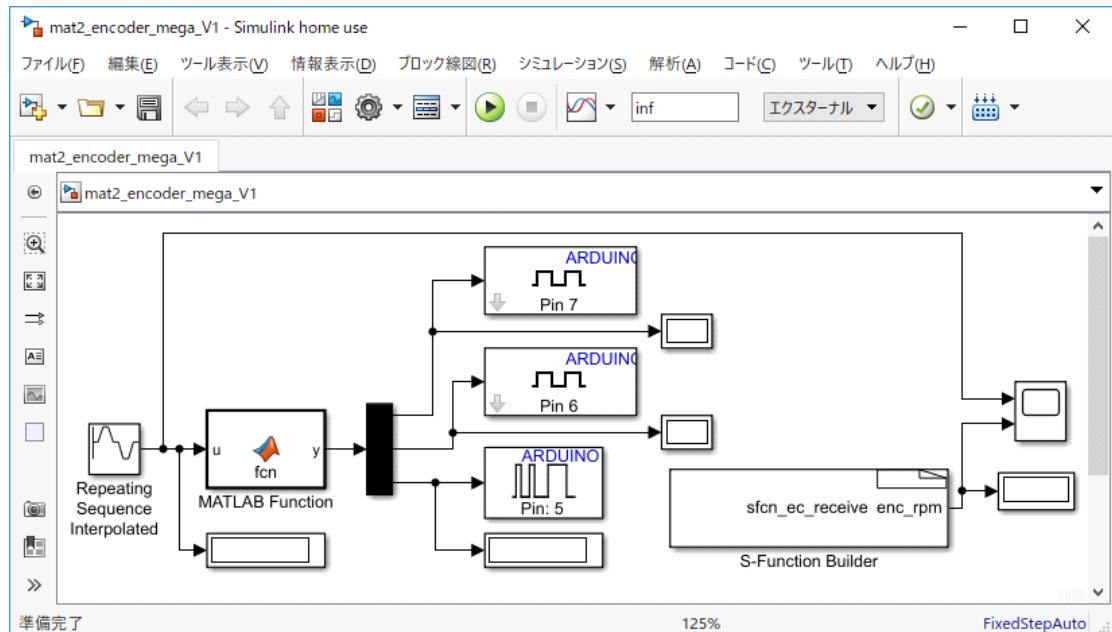


Fig.3.3 Matlab/Simulink モデル

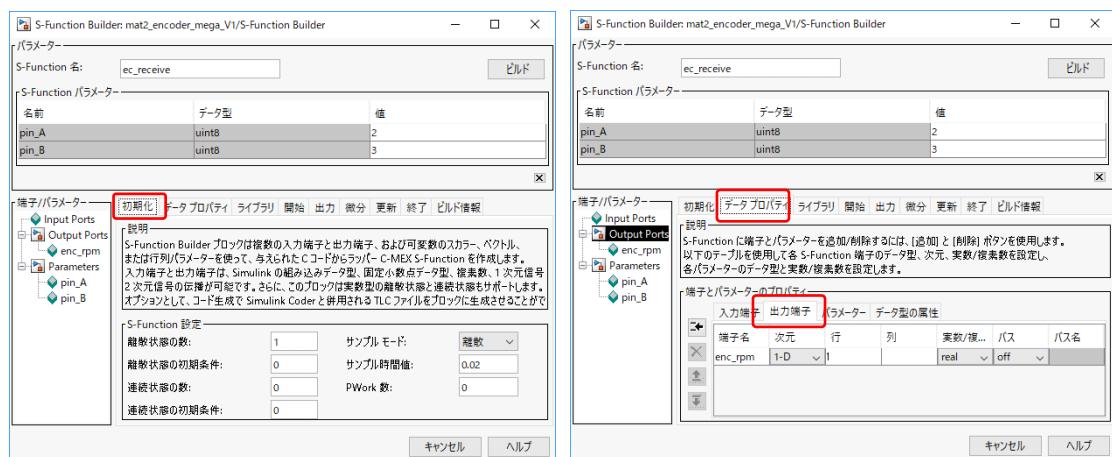


Fig.3.4 S-Function Builder

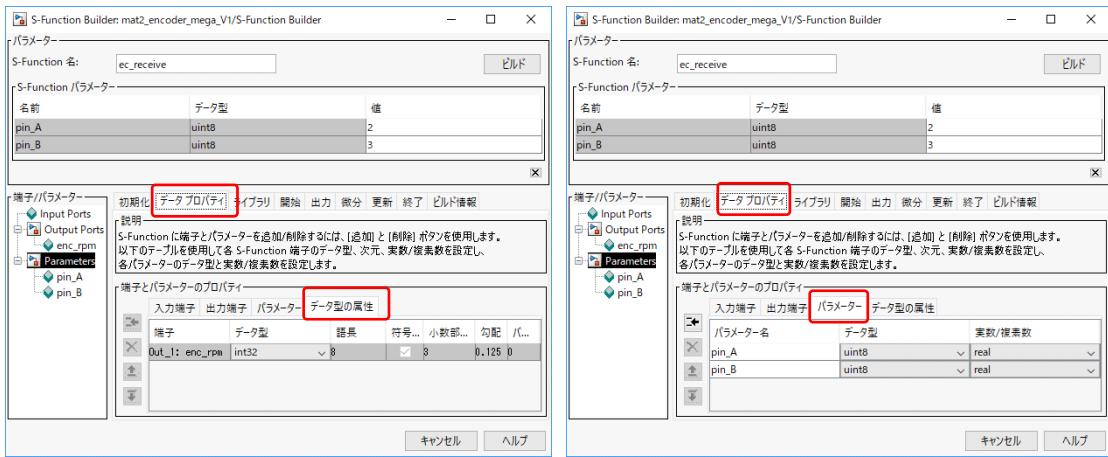


Fig.3.5 S-Function Builder

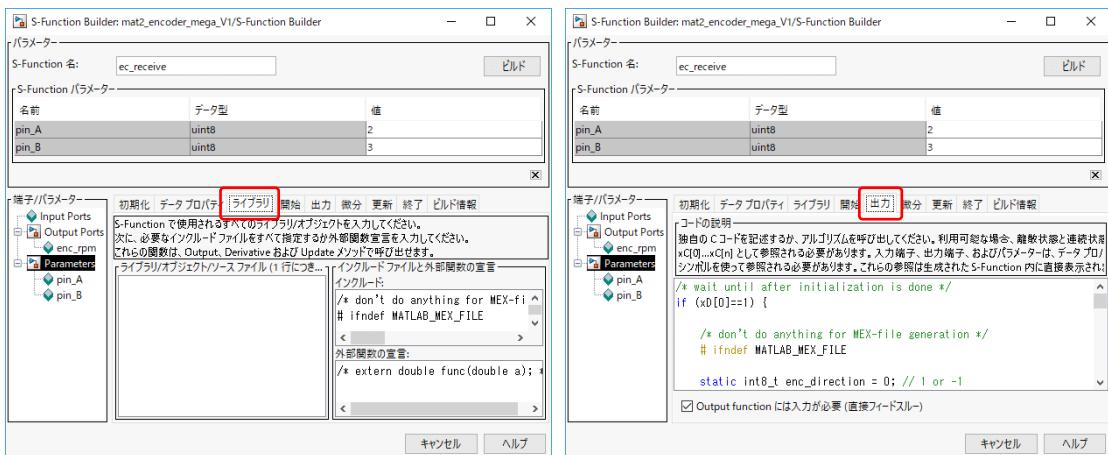


Fig.3.6 S-Function Builder

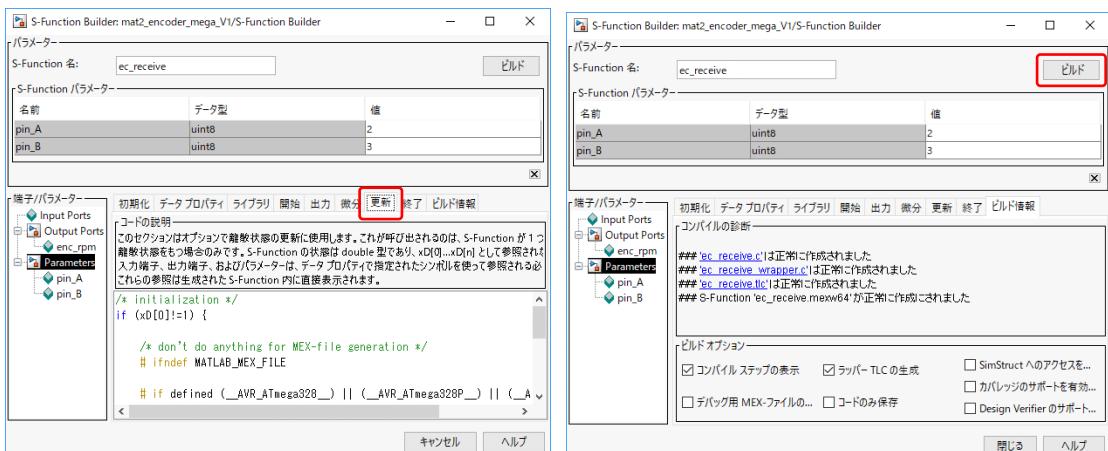


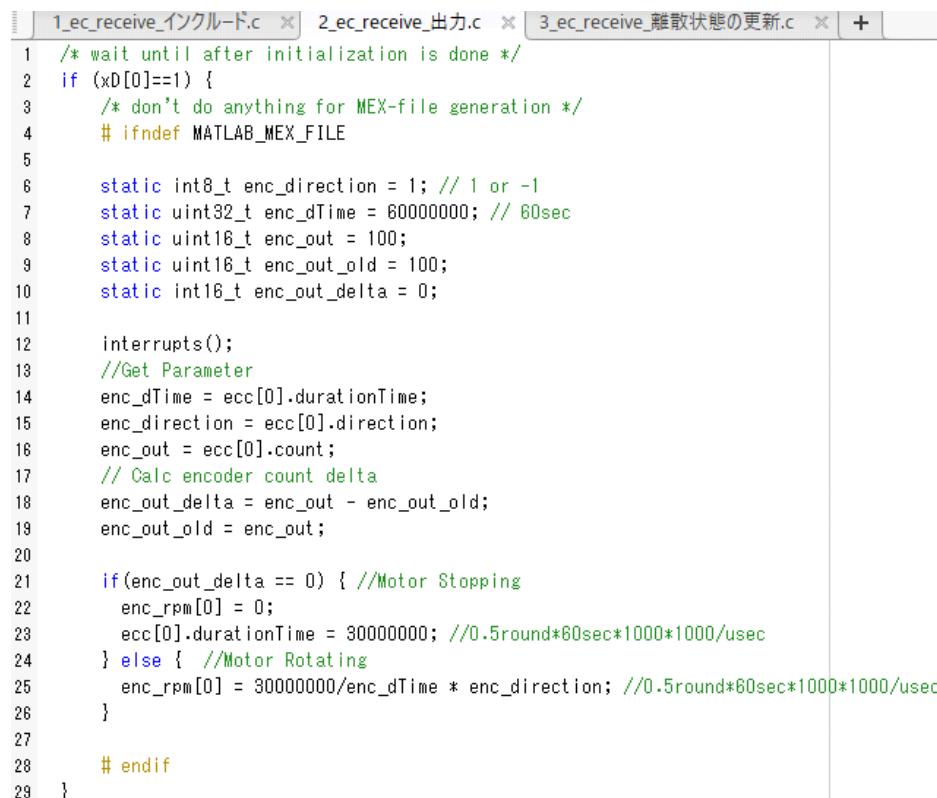
Fig.3.7 S-Function Builder

```

1  /* don't do anything for MEX-file generation */
2  #ifndef MATLAB_MEX_FILE
3  # include <Arduino.h>
4  typedef struct {uint8_t pinNumA; uint8_t pinNumB; uint8_t direction;
5  	uint16_t count; uint32_t startTime; uint32_t durationTime;} encChannel;
6  volatile encChannel ecc[1] = {{0,0,0,100,0,6000000}}; // startTime & durationTime: usec
7  volatile int oldEncB;
8  volatile int flag_time;
9
10 /* returns the interrupt number for a given interrupt pin
11  see https://www.arduino.cc/en/Reference/AttachInterrupt */
12 int getIntNum(int pin) {
13 	switch(pin) {
14  	case 2:
15  	    return 0; //int.0
16  	case 3:
17  	    return 1; //int.1
18  	case 21:
19  	    return 2; //int.2
20  	case 20:
21  	    return 3; //int.3
22  	case 19:
23  	    return 4; //int.4
24  	case 18:
25  	    return 5; //int.5
26  	default:
27  	    return -1;
28  }
29 }
30
31 // Pololu Magnetic Encoder Interrupt
32 void ecalc0() {
33 	noInterrupts();
34 	//Get Parameter
35 	int newEncB = digitalRead(ecc[0].pinNumB);
36 	int newEncA = digitalRead(ecc[0].pinNumA);
37 	unsigned long ecc_Time = micros(); //overflow (go back to zero), after approximately 70 minutes
38
39 //Rotating Direction
40 	if (newEncA) {
41  	    if (oldEncB && !newEncB) { // up
42  	        ecc[0].count++;
43  	        ecc[0].direction = 1;
44  	    } else { // down
45  	        ecc[0].count--;
46  	        ecc[0].direction = -1;
47  	    }
48 	} else {
49  	    if (oldEncB && !newEncB) { // down
50  	        ecc[0].count--;
51  	        ecc[0].direction = -1;
52  	    } else { // up
53  	        ecc[0].count++;
54  	        ecc[0].direction = 1;
55  	    }
56 	}
57 	oldEncB = newEncB;
58
59 //Duration time
60 	if(flag_time == 0) { //Get start time & Count up flag
61  	ecc[0].startTime = ecc_Time;
62  	flag_time++;
63  } else if(flag_time < 3) { //6 magnetic pole * 0.5round = 3
64  	flag_time++;
65  } else { //Calc duration time
66  	ecc[0].durationTime = ecc_Time - ecc[0].startTime; //0.5round
67  	flag_time = 0;
68  }
69 	interrupts();
70 }
71 #endif

```

Fig.3.8 S-Function Builder ライブラリ インクルード

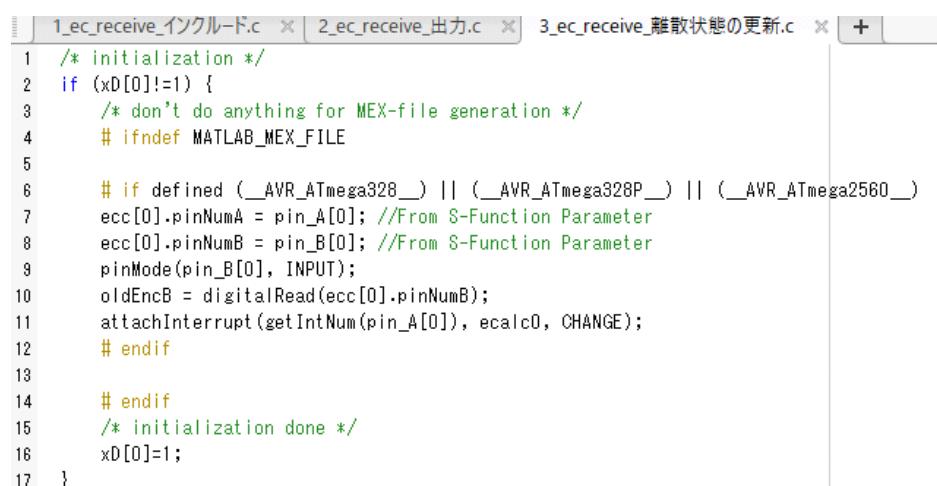


```

1 /* wait until after initialization is done */
2 if (xD[0]==1) {
3     /* don't do anything for MEX-file generation */
4     #ifndef MATLAB_MEX_FILE
5
6     static int8_t enc_direction = 1; // 1 or -1
7     static uint32_t enc_dTime = 60000000; // 60sec
8     static uint16_t enc_out = 100;
9     static uint16_t enc_out_old = 100;
10    static int16_t enc_out_delta = 0;
11
12    interrupts();
13    //Get Parameter
14    enc_dTime = ecc[0].durationTime;
15    enc_direction = ecc[0].direction;
16    enc_out = ecc[0].count;
17    // Calc encoder count delta
18    enc_out_delta = enc_out - enc_out_old;
19    enc_out_old = enc_out;
20
21    if(enc_out_delta == 0) { //Motor Stopping
22        enc_rpm[0] = 0;
23        ecc[0].durationTime = 30000000; //0.5round*60sec*1000*1000/usec
24    } else { //Motor Rotating
25        enc_rpm[0] = 30000000/enc_dTime * enc_direction; //0.5round*60sec*1000*1000/usec
26    }
27
28    #endif
29 }

```

Fig.3.9 S-Function Builder 出力



```

1 /* initialization */
2 if (xD[0]!=1) {
3     /* don't do anything for MEX-file generation */
4     #ifndef MATLAB_MEX_FILE
5
6     #if defined (__AVR_ATmega328__) || (__AVR_ATmega328P__) || (__AVR_ATmega2560__)
7     ecc[0].pinNumA = pin_A[0]; //From S-Function Parameter
8     ecc[0].pinNumB = pin_B[0]; //From S-Function Parameter
9     pinMode(pin_B[0], INPUT);
10    oldEncB = digitalRead(ecc[0].pinNumB);
11    attachInterrupt(getIntNum(pin_A[0]), ecalco, CHANGE);
12    #endif
13
14    #endif
15    /* initialization done */
16    xD[0]=1;
17 }

```

Fig.3.10 S-Function Builder 离散状態の更新

3.4 実行

Scope ブロックのグラフ表示を立ち上げた状態で”実行”することで、回転数測定結果をリアルタイムにグラフを描写できます。実行例として Fig.3.11 に示します。操作量に応じて、回転数が高くなることが確認できます。

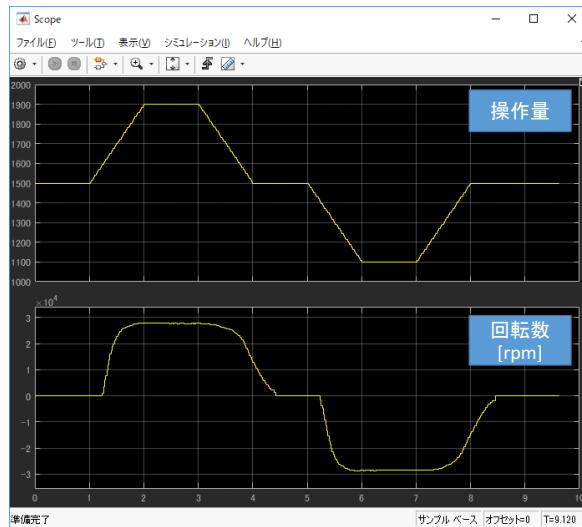


Fig.3.11 回転数の計測結果

正しく回転数が計測できているか確認する方法として、次のような外部計測器 (Fig.3.12) を使う方法があります。例えば回転計を使用する場合は、エンコーダの円盤に反射シールを張り付け、モータを定常回転させれば回転数を計測できます。

- 回転計：非接触デジタルタコメーター DT2234B
- オシロスコープ：MiniDSO DS203



Fig.3.12 計測器

第4章

Matlab/Simulink と Arduino でモータ回転数制御をしよう

4.1 ハード構成

ハード構成、および電気回路は第3章と同じで、Fig.4.1になります。

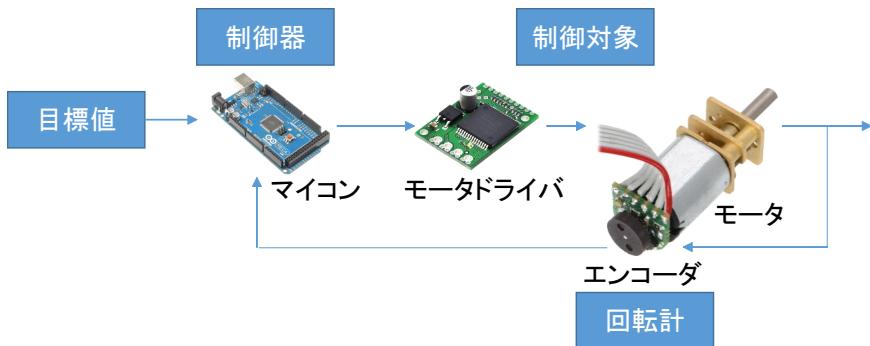


Fig.4.1 ハード構成

4.2 PID 制御モデル

Simulink による PID 制御モデルは Fig.4.2 の通りです。PID Controller ブロックのパラメータは Fig.4.3 の値を入力しました。Saturation ブロックのパラメータは Fig.4.4 の値を入力しました。今回のモータは DC6V で 29000~30000rpm にて回るので、Saturation1 ブロックにはモータ最大回転数より若干低い値を入力します。Saturation2 ブロックには操作量の上下限で飽和するようにします。

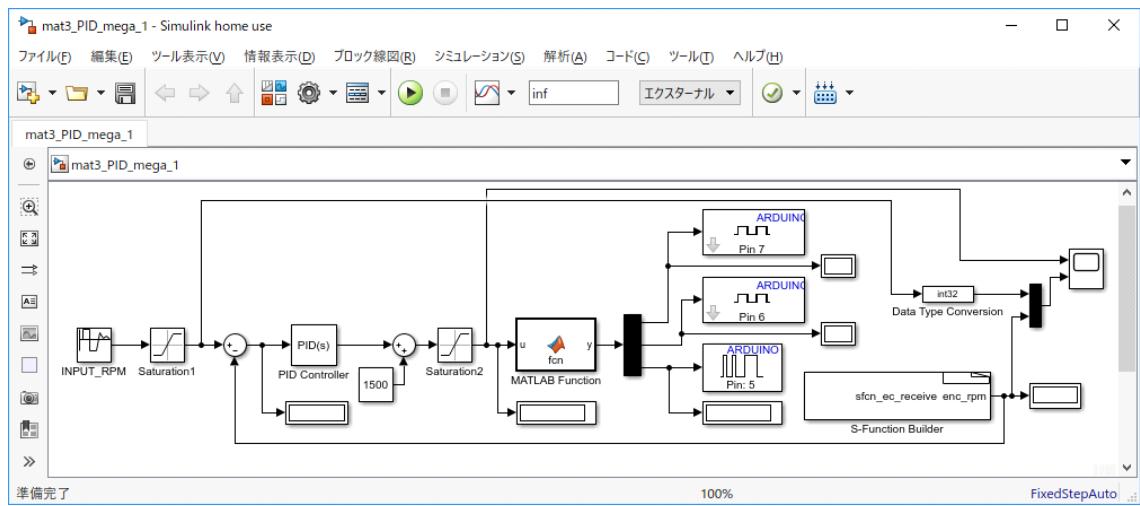


Fig.4.2 PID 制御の Simulink モデル

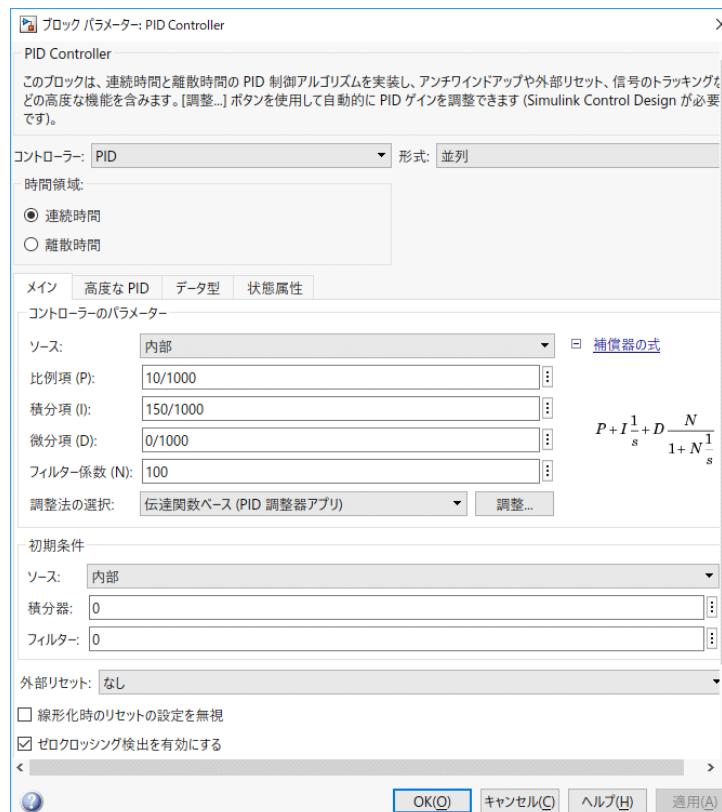


Fig.4.3 PID Controller ブロック パラメータ

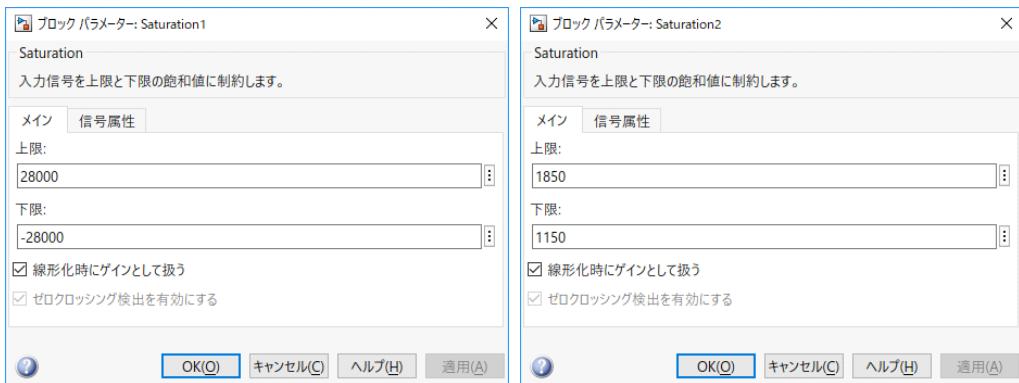


Fig.4.4 Saturation ブロック パラメータ

4.3 実行

回転数の計測結果を Fig.4.5に示します。回転数の黄色線が目標の回転数で、青線が実測値になります。目標に対して実測値がおおむね追従していることが確認できます。

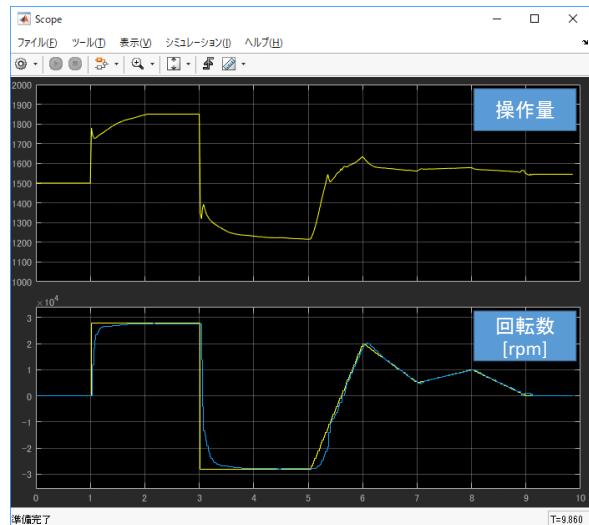


Fig.4.5 回転数の計測結果

4.4 ロボットへの実装例

Fig.1.1に示したロボットへの実装例になります。本ロボットは脚ユニットを 4 脚使用しています。そのハード構成は Fig.4.6の通りです。中央制御基板で受信機からのラジコン信号を読み取り、I2C 通信で各脚制御基板に回転数目標値を出力しています。

Fig.4.7に中央制御基板の Simulink モデルを示します。ラジコン信号の読み取り方法は Using

an RC Controller with Arduino and Simulink[10] を参考にしました。読み取ったラジコン信号のパルス幅をローパスフィルタにかけ、パルス幅から回転数目標値に変換し、I2C ブロックに入力するため、回転数目標値を + - の符号を含め 3byte のデータに変換しています。

Fig.4.8に脚制御基板の Simulink モデルを示します。I2C 通信を S-function ブロックで実現する方法は ADLX345 i2c Driver for Arduino Mega[11] を参考にしました。I2C Master から読み取った回転数目標値は、3byte データを復号して Fig.4.2 で作製したモデルの入力側に接続しています。

実行例を Fig.4.9 に示します。手で操作したプロポの挙動に応じて、回転数を制御することができます。

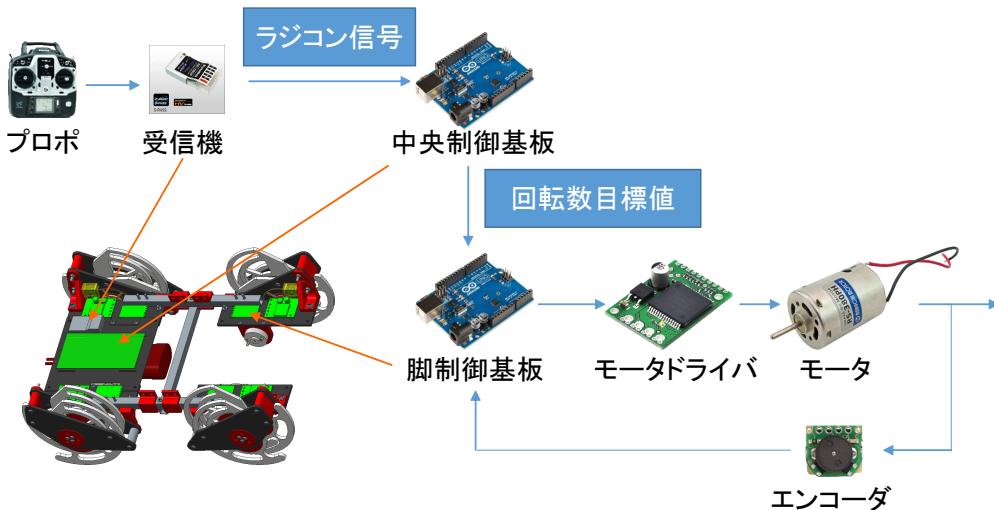


Fig.4.6 ロボットへの実装

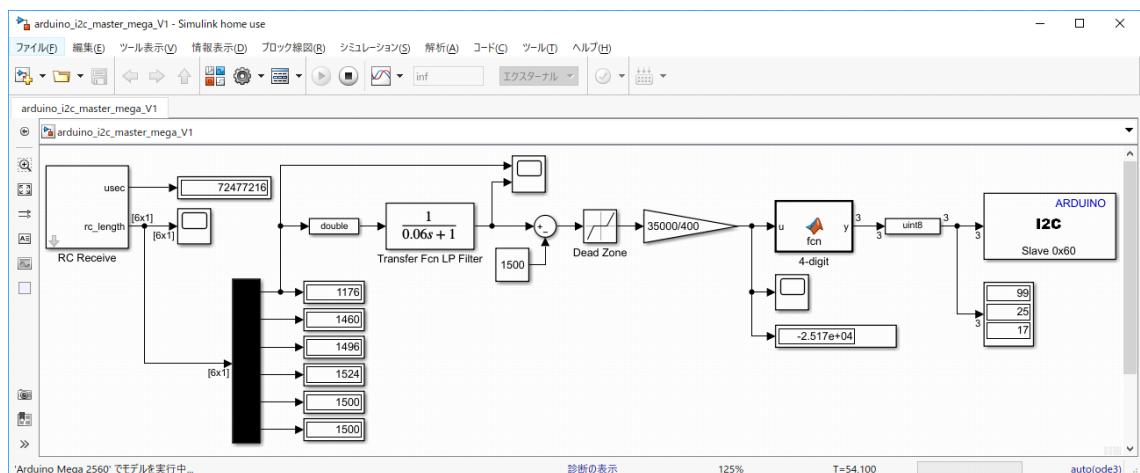


Fig.4.7 中央制御基板の Simulink モデル (I2C Master)

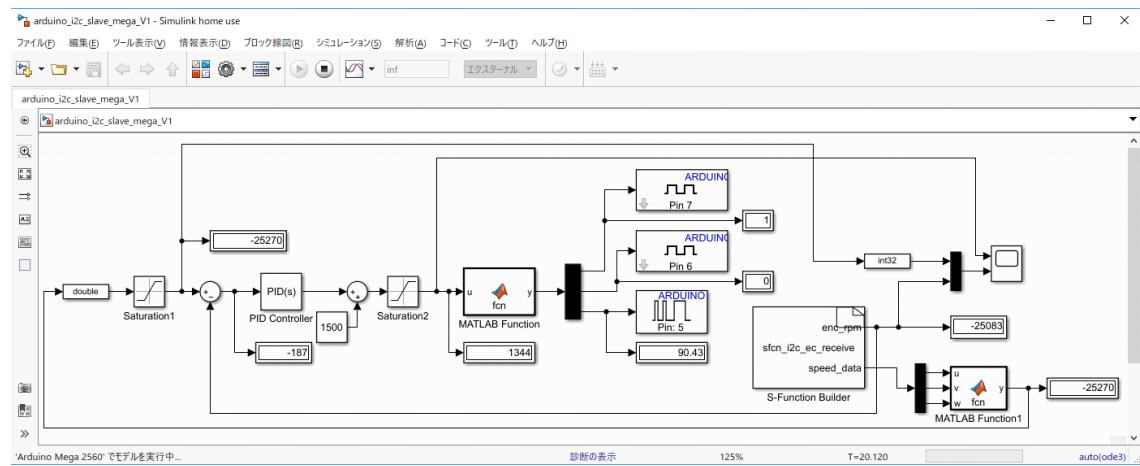


Fig.4.8 脚制御基板のSimulinkモデル(I2C Slave)

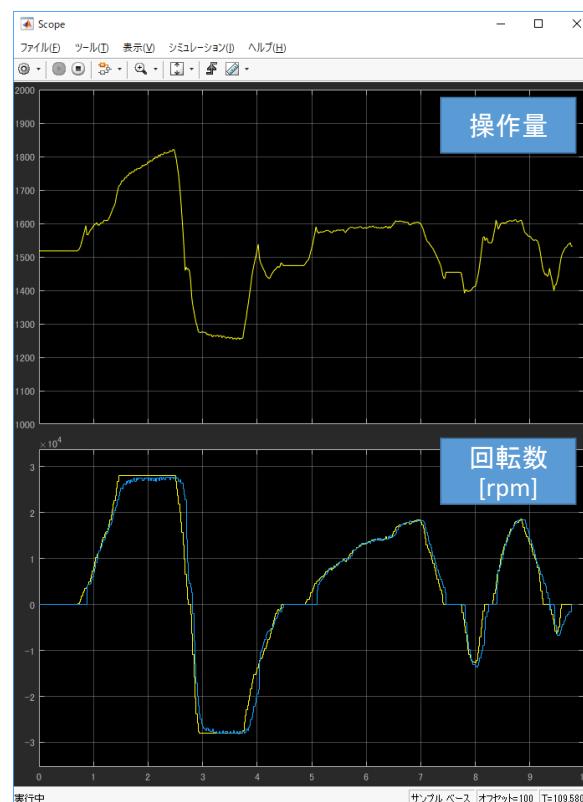


Fig.4.9 実行例

参考文献

- [1] かわさきロボット競技大会 公式 HP
<http://www.kawasaki-net.ne.jp/robo/>
- [2] MathWorks ストア
<https://jp.mathworks.com/store/link/products/>
- [3] Simulink Support Package for Arduino Hardware Examples
<http://jp.mathworks.com/help/supportpkg/arduino/examples.html>
- [4] Windows SDK 7.1 のインストール時のまりポイント
<http://d.hatena.ne.jp/kfujieda/20140827/1409137229>
- [5] How can I install sdk 7.1 on windows 10 ???
<https://jp.mathworks.com/matlabcentral/answers/233850-how-can-i-install-sdk-7-1-on-windows-10/>
- [6] スイッチサイエンス
<https://www.switch-science.com/>
- [7] Pololu HP モータドライバ VNH5019
<https://www.pololu.com/product/1451>
- [8] Pololu HP 磁気式エンコーダ
<https://www.pololu.com/product/3081>
- [9] ウィキペディア PID 制御
https://ja.wikipedia.org/wiki/PID_制御
- [10] Using an RC Controller with Arduino and Simulink
<https://jp.mathworks.com/videos/using-an-rc-controller-with-arduino-and-simulink-91738.html>
- [11] ADXL345 i2c Driver for Arduino Mega
<https://jp.mathworks.com/matlabcentral/fileexchange/41027-adlx345-i2c-driver-for-arduino-mega/>

奥付

発行

でし・ぶろんぶと

2018年8月10日 初版第1刷発行

著者

santny

フオロ一

takarakasai

kyo46

印刷

製本直送.com

Web

<http://deshi-prompt.github.io/>

Mail

ada.robo1@gmail.com