平成30年度 創成工学実験

システム製作報告書

4学年 電気情報工学科 番号 21 氏名 佐藤凌雅

(共同実験者 番号 37氏名 中田和斗 ）

提出期限 平成31年1月 31 日　17:00

報告書受取 平成31年1月 29 日

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 番号 | 氏名 |  |  |  |

**１．表紙**

**２．製作システムの名称**

倒立振子

**３．製作システムの概要**

　通常，車などの車輪を使用した機構は3輪以上で構成される．なぜなら２輪では車体が安定せずに転倒してしまうからである．しかし，そのような通常では倒れてしまうような２輪の機構でもタイヤを絶妙な操作で動かし続ければ姿勢を保持してくれる．

　今回製作したシステムは前述した２輪だけで構成されたロボットを倒立させ続けるものである．

ロボットの電源を入れたらあとは勝手に倒立状態を保持しようと動き出す．余力があれば多少前後に動くような実装も行いたかったが，間に合わなかった．

このシステムを実現させるためにはマイコンのプログラミング知識だけでなく制御工学などの知識も多少必要となる為，周辺知識も合わせて学習した．

　授業ではPICマイコンを扱ったため，全ての処理をPICで行いたかったが，時間計測とI２C通信の実装がうまくいかなかったため，姿勢推定と出力計算をArduino，モータ駆動処理をPICで行うことにした．なお，両者のデータの受け渡しにはシリアル通信を使用している．

　また，インターネット上には同様の作品の情報が多く見つかるが，ここは制御工学の学習も兼ねてネットの情報には極力頼らないようにした．ネットの情報を参考としたのは角度の演算に使用するカルマンフィルタの実装方法と制御理論全般に関する事項，PICでのシリアル通信，CCPモジュールの使用方法，各種電子部品のデータシートのみとし，具体的なロボットの機械的な構成，回路構成，全体のシステム構成，制御プログラムは全て自分たちで考え，製作を行った．

**４．ハードウェアについて**

* **ハードウェア構成図（ブロック図）と回路図および配線図**

図1，図2，図3にそれぞれハードウェア構成図，回路図，配線図を示す．

Arduino

IMU

モータドライバ

モータ

マイコン用電源

駆動用電源

スイッチ

PIC16F1827

スイッチ

図1 ハードウェア構成図

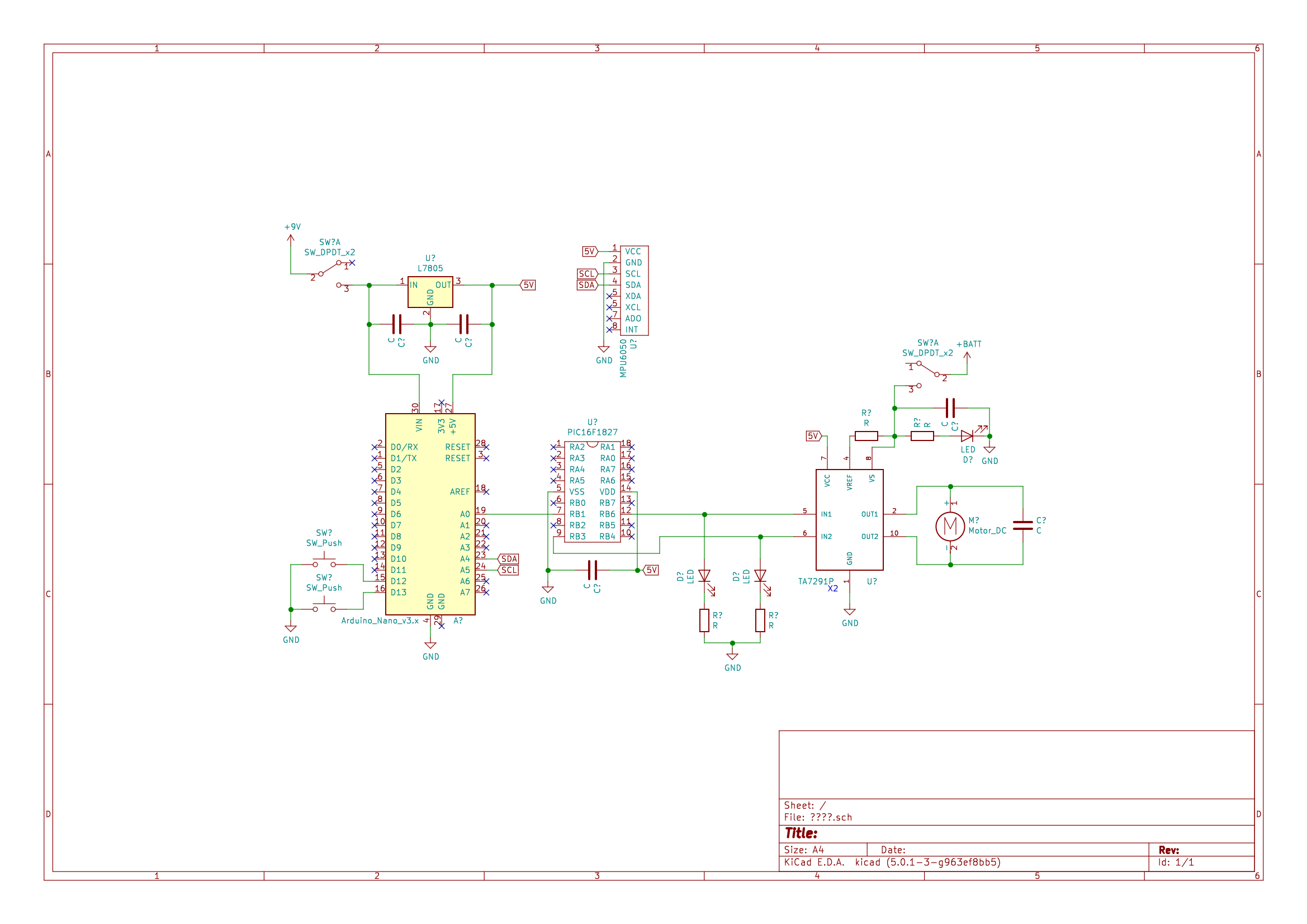


図2回路図

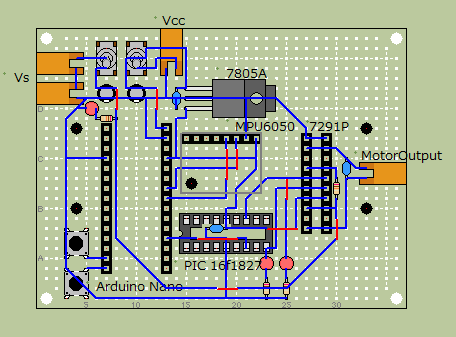


図3 結線図

* **使用した主なパーツ**

使用したパーツとその目的を表1に示す．

表1 システムに使用した回路部品

|  |  |
| --- | --- |
| パーツ名 | 役割 |
| IMU（MPU6050） | 車体に加わる加速度と角速度を測定する． |
| ArduinoNano | 姿勢の推定と出力の計算を行う． |
| PIC16F1827 | 出力司令値に応じてPWM出力を行う． |
| モータドライバ（TA7291P） | PWM信号に従い，モータを駆動する． |
| モータ（RE-260RA） | 力学的エネルギーに変換する． |

* **ハードウェアの動作説明**

倒立振子のシステムを作成するためには車体の現在の姿勢を推定すること，姿勢に応じた出力値を計算すること，計算した出力値に応じて車輪を回転させることが要求される．

車体の姿勢を推定するために，３軸の加速度センサと３軸のジャイロセンサが内蔵された慣性計測装置（IMU）を車体に固定した．このIMUはI2C通信によってマイコンから各種データを読み出すことが可能である．

姿勢に応じた出力値を計算には，ArduinoNanoというAVRマイコン（ATmega328）が内蔵されたマイコンボードを使用した．

計算した出力値に応じてPWM信号を発生させるために，PIC16F1827を使用した．ArduinoNanoとの通信にはシリアル通信（RS232C）を用いた．

また，PWM信号に従い，モータドライバ（TA7291P）がモータ（RE-260RA）に電圧を印加することによって，車輪が回転する．

* **ハードウェア製作での工夫、不具合、長所、短所、改善案等**

ハードウェアの製作では車体の重心を車体中央にすることを心がけた．ハードウェアの構成部品で比較的質量が大きい駆動用電池を車体前後の両方に分散して配置し，マイコン用の電池は車体の中心軸上に配置することによって，重心のバランスをとった．また，質量の大きい部品を車体下部に集中させ，低重心化を図った．さらに，車体はアルミ板を曲げ加工して一体構造にし，測定，計算，駆動に必要な回路を全て一つの基板にまとめて構成することで，車体の軽量化も実現した．

回路設計ではKiCADと呼ばれるCADを用いて設計を行い，各素子の配置，結線はPasSというユニバーサル基板配線用のCADで事前に配置図を作成してから，回路の作成を行うことにした．これにより，スムーズに回路の制作を行うことができた．

この回路は前述したように測定，計算，駆動が一つの基板にまとめられており，本システムのハードウェアで一番工夫した点である．一つの基板にまとめられていることで，結線ミスや接触不良などが起こりにくい構造になっているのが，この回路の長所である．しかし，駆動用の回路も一つの基板になっているため．駆動側のノイズがマイコン側にも影響を及ぼし，Arduinoがシャットダウンしてしまうトラブルが度々発生していた．これを解決するために，電源部分にコンデンサを配置したところ，Arduinoのシャットダウンは無くなったが，それでも信頼性には欠ける回路であると考える．これがこの回路の短所でもある．解決するためには，基板を分離させることや，フォトカプラなどでマイコン側と駆動側でGNDをアイソレートすることが改善案として考えられる．

**５．ソフトウェアについて**

* **フローチャート**

ジャイロと加速度の値を読む

電源ON

センサの値をフィルタして車体の姿勢を推定

推定された姿勢をもとにモータの出力値を決定

出力値をPICに送ってPICからPWMを出力

モータドライバICを通してモータを駆動

繰り返し

* **処理の流れ**

マイコン用電源を入れると，IMUが加速度と角速度を計測し始める．

ArduinoからIMUに対して，初期化のコマンドをI2C で送る．

初期化が完了したら，ArduinoはIMUに対して測定したデータをリクエストする．

IMUから受け取った重力加速度をもとに，車体の仮の角度を算出する．

初回の計測をキャリブレーションとしてとっておく．

仮の角度と車体の角速度をカルマンフィルタで処理し，車体角度の推定値を算出する．

推定された姿勢と目標の角度との偏差に応じてモータの出力値を決定する（PID制御）

この出力値をシリアル通信（RS232C）でPICに送る

PICはPWM信号をモータドライバ（TA7291P）に伝える．

PWM信号に従い，モータドライバがモータ（RE-260RA）に電圧を印加することによって，車輪が回転する．

逐次，車体の姿勢に応じて車輪の回転数を計算し，対応する電圧をモータに加えている．

* **PID制御**

PID制御とは，フィードバック制御の一種であり、入力値の制御を出力値と目標値との偏差、その積分、および微分の3つの要素によって行う方法のことである[1]

制御入力は式（１）で表される．

ただし，

今回rは0[deg]，yは車体の角度[deg]とし，uをPWMのデューティとした（-1〜1）

比例項は算出した角度に対して係数をかけて計算した．

積分項は車体角度に前回のループの時間をかけたものを積分値とし，それに係数をかけた．

微分項は角度の微分である，角速度をジャイロから直接得て，それに係数をかけた．

* **ソフトウェア製作での工夫、不具合、長所、短所、改善案等**

ソフトウェアを製作する際になるべく処理を関数に分割し，さらに役割ごとにソースファイルも分割することを心がけてコーディングをした．また，試行錯誤しながらコードを書いていくために，GitHubを用いてソースファイルの管理を行った．これにより，逐次コードのバックアップが取れるため，容易に前のバージョンに戻すことができ，開発の効率が向上した．

現状の制御では，目標角度の設定（キャリブレーション）が人力での設定となっており，起動時の角度が垂直でないとうまく倒立しない．改善策としては，PIDの積分項が収束しない場合にはキャリブレーションの値を更新するなどが改善策として考えられる．

**６．まとめ**

* **設計したシステムのでき具合や動作について各自で評価すること。**

数十秒であればほぼ確実に倒立できるものを作成できた．

外乱などが少なければ，数分間倒立することも可能である．

当初の目標であった，「倒立」という目標は安定的に達成することができた．一般的に製品化されているもの（セグウェイなど）には到底及ぶような作品ではなかったが，概ね古典制御理論に則った制御システムを作成できたと評価する．

* **後期の実験で修得したことや反省**

倒立振子ロボットをハードウェア・ソフトウェア共に一から製作することで，機体の作成と回路の設計製作，制御プログラムのコーディングというものづくりの一連の流れ，一手法について理解を深めることができた．また，カルマンフィルタの理論やPID制御をはじめとした古典制御理論，シリアル通信やI2Cなどの通信手段についても学ぶことができた．

なお，今回PIDの係数調整は試行錯誤で行ったため，開発時間の多くがゲイン調整に費やされてしまった．限界感度法やステップ応答法などを勉強して，もっと効率よくパラメータの調整を行うべきだったと考察する.

最後に制作後ネットで倒立振子について調査したところ，内部状態を考慮した解析を行い，状態方程式をモデルとして制御したものが多く紹介されていた．今回は車体角度と目標角との偏差に対してPID制御を行ったため，信号の入出力による解析しか行えていなかった．今後機会があれば現代制御やMPCにも挑戦したい.

**７．回路図**、**ソースファイル、製作システムの写真、その他参考資料**

* **製作の際に参考にした主な文献**

[1]PID制御 – Wikipedia ，https://ja.wikipedia.org/wiki/PID制御

[2]カルマンフィルタの考え方，https://logics-of-blue.com/kalman-filter-concept/

[3] TKJ Electronics A practical approach to Kalman filter and how to implement it，

http://blog.tkjelectronics.dk/2012/09/a-practical-approach-to-kalman-filter-and-how-to-implement-it/

[4] Kalman Filtering，https://academic.csuohio.edu/simond/courses/eec644/kalman.pdf

[5] PICマイコン（16F1827）のPWMのやり方，https://meideru.com/archives/1649

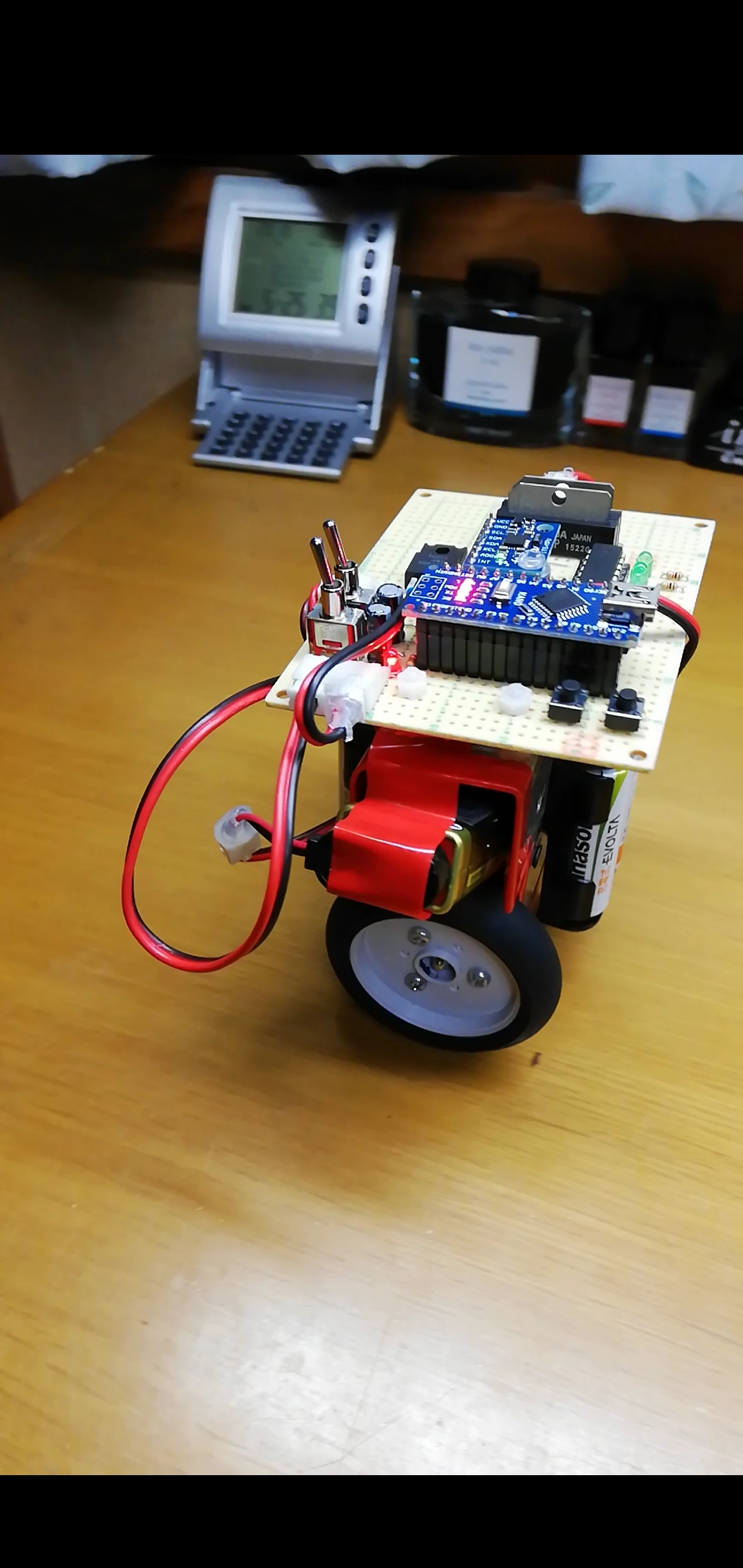
[6]C言語によるシリアル通信，http://www.picfun.com/serial22.html

[7]佐藤和也ほか（2018）「はじめての制御工学 改訂第2版」講談社．

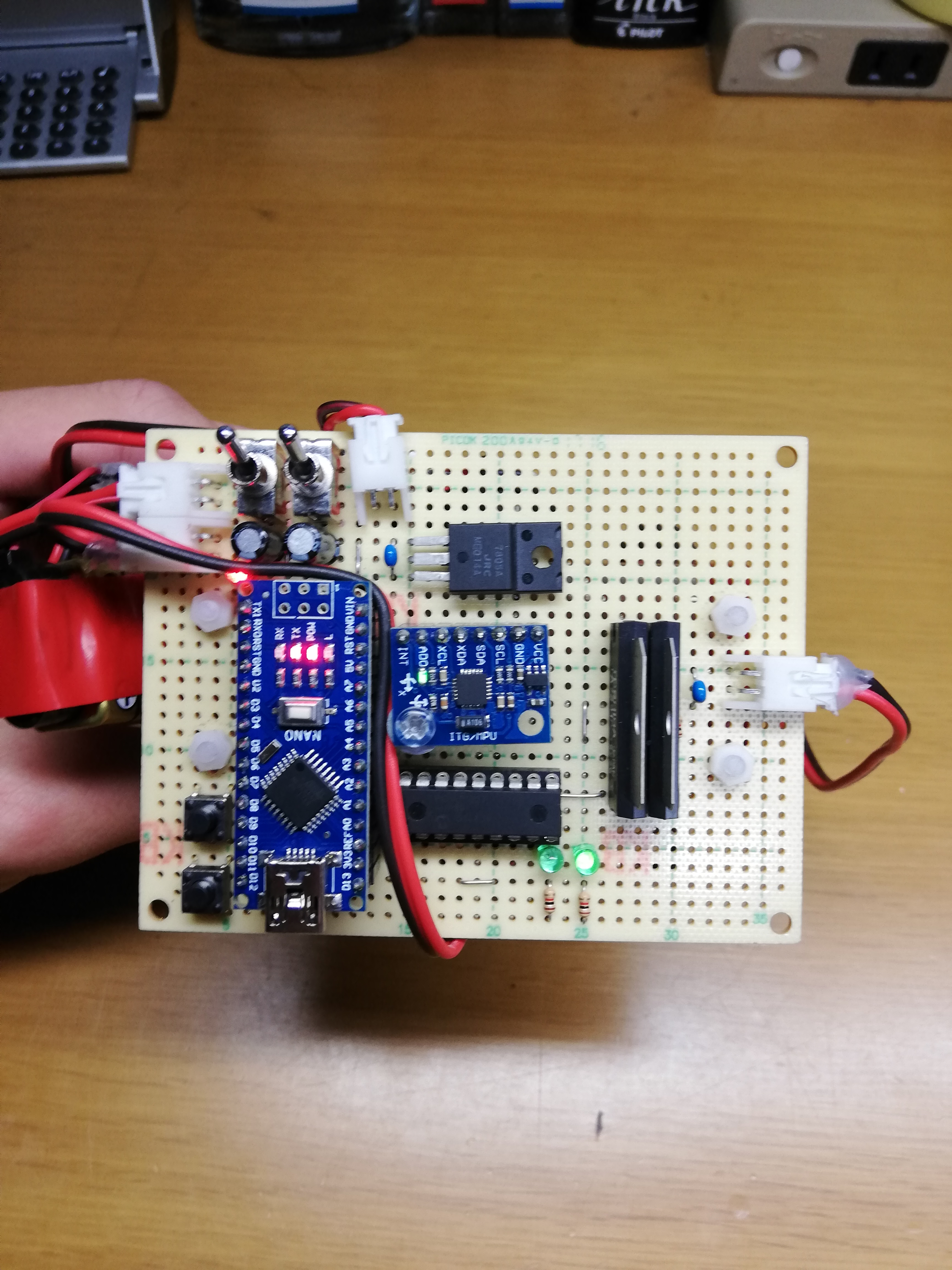
[8] SebastianThrunほか（2016）「確率ロボティクス (プレミアムブックス版) 」

上田隆一 訳, マイナビ出版．

* **製作システムの写真**



* **回路の写真**



* **ソースコード**

今回作成したシステムのソースコードおよび回路図，結線図，発表資料などは全てGitHub(https://github.com/surpace0924/InvertedPendulum/)にアップロードしてある．

* ArduinoNanoメイン処理（Arduino\_InvertedPendulum.ino）-（1）



* ArduinoNanoメイン処理（Arduino\_InvertedPendulum.ino）-（2）



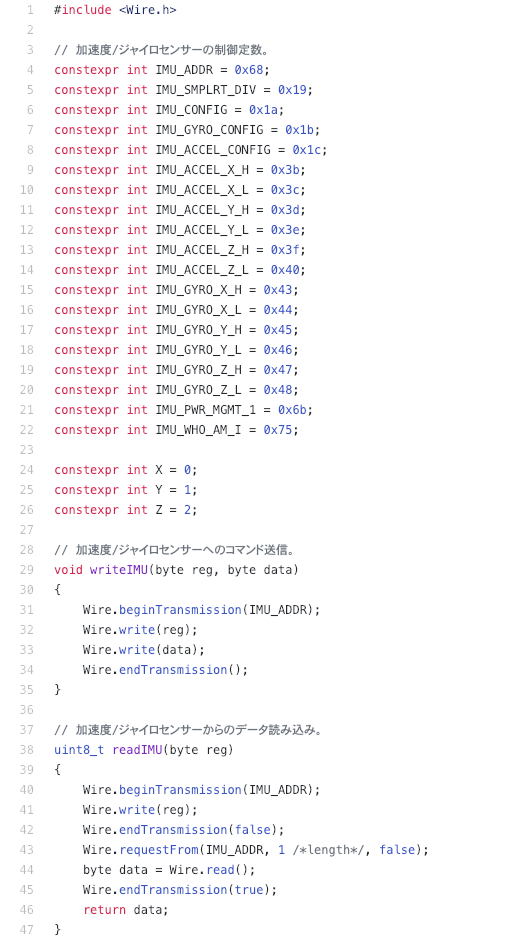
* ArduinoNanoメイン処理（Arduino\_InvertedPendulum.ino）-（3）



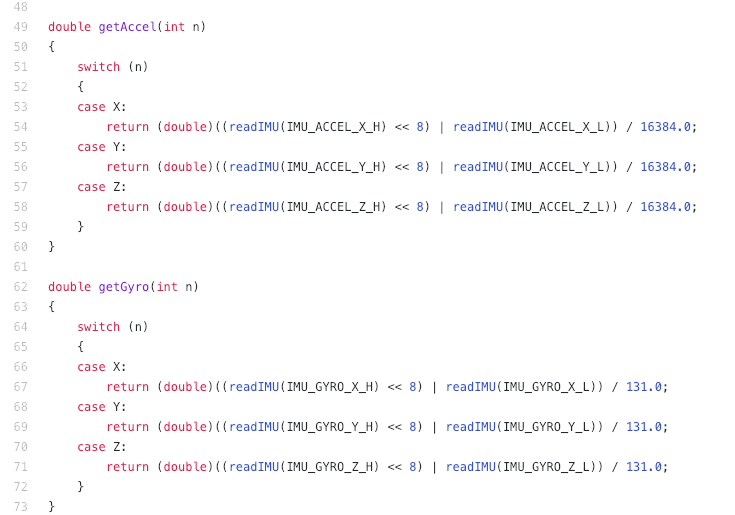
* ArduinoNanoメイン処理（Arduino\_InvertedPendulum.ino）-（4）



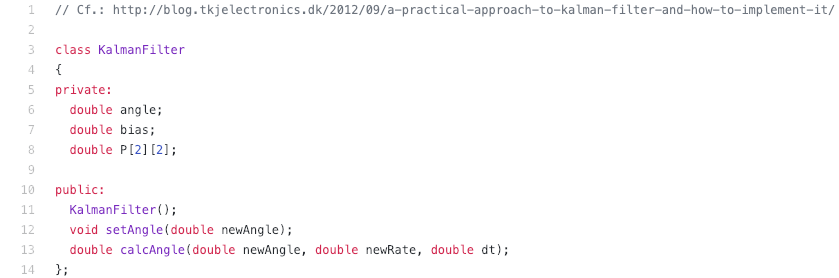
* ArduinoNanoIMU処理（IMU.h）-（1）



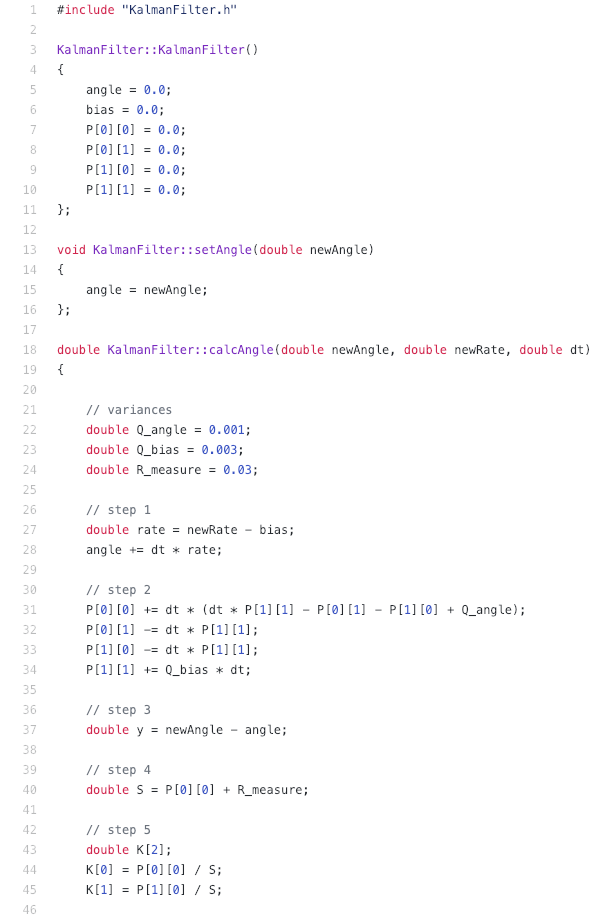
* ArduinoNanoIMU処理（IMU.h）-（2）



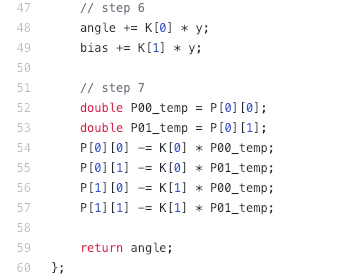
* ArduinoNanoカルマンフィルタ処理（KalmanFilter.h）-（1）



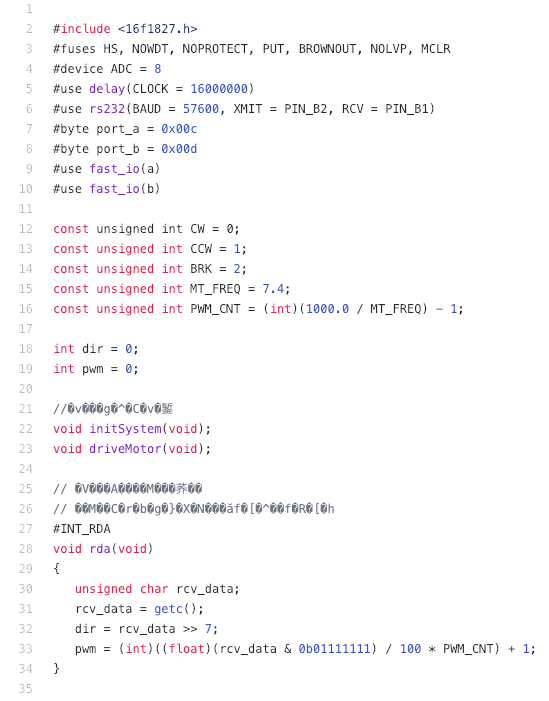
* ArduinoNanoカルマンフィルタ処理（KalmanFilter.cpp）-（1）



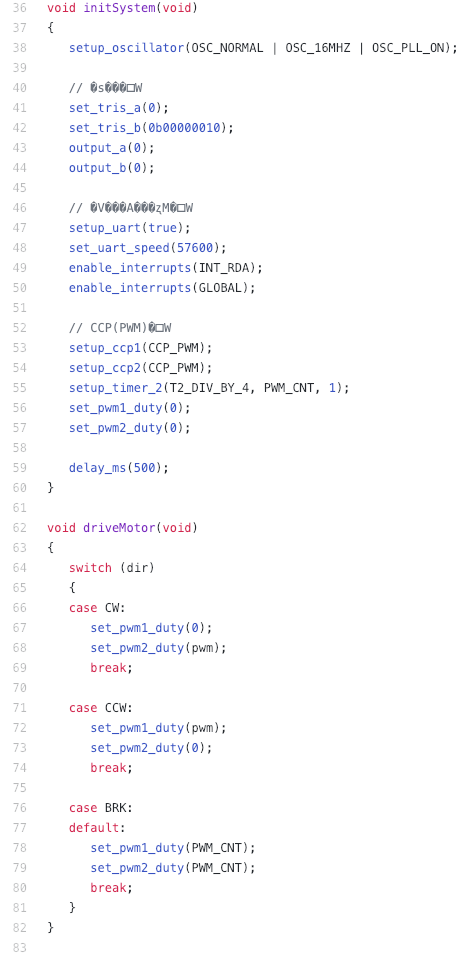
* ArduinoNanoカルマンフィルタ処理（KalmanFilter.cpp）-（2）



* PIC16F1827モータ駆動処理（PIC\_InvertedPendulum.c）-（1）



* PIC16F1827モータ駆動処理（PIC\_InvertedPendulum.c）-（2）



PIC16F1827モータ駆動処理（PIC\_InvertedPendulum.c）-（3）

