

AI制御の教育研究のための ミニ四駆プロジェクト Rizm mini4WD Project

<http://www.ritsumeai.ac.jp/se/re/izumilab/dist/mini4WD/>

立命館大学 理工学部 電子情報工学科
いずみ研究室



R 制御の問題設定

- 定値制御

一定の目標値への安定化

例...変動する負荷に対して出力電圧を一定にするレギュレータ、
外乱に対して姿勢を維持する倒立振子、など

- 追従制御

変化する目標値への追従

例...目標経路に沿って進む自動車両、人の動きに追従するロボット、など

- 行動制御

目的に対する目標値(系列)を生成

例...目的地を与えられ経路を決定する自動車両、ポーズを与えられそのための動きを生成するロボット、など

- それぞれで、何らかの最適化目標が与えられる

オーバーシュート、安定化までの時間、目標に対する誤差、経路長、到達時間、消費エネルギー、など



不確定性・自由度の拡大

- デバイスの多様化
 - 加速度、角速度、地磁気、GPS、画像...
 - 特性・精度はまちまち、センサ情報の統合が必要
- デバイスの選択肢拡大
 - 消費電力・サイズと性能・精度のトレードオフ
 - 価格と性能・精度のトレードオフ
- システムの高度化
 - 制御の自由度が拡大
 - 最適化の余地が拡大

例...多軸化により滑らかな動きが可能になるが、最適制御が難化
- 不確実性を取り扱い、より高度なシステムに対応

- 解の不確実性・多様性が大きく複雑
- 解析的に最適解を導出する方法を見出すことが難しい
- 経験的手法、確率・統計的アプローチ、機械学習
 - ファジィ
 - ニューラルネットワーク
 - カルマンフィルタ
 - パーティクルフィルタ
 - SVM
 - などなど...
- いわゆるAIによる最適制御の適用が広がる



R ミニ四駆AI大会

- AI技術を応用した小型モデルカーの高速制御
- 2013年から開始された公開型研究コンペティション
- 身近で安価なものをAIの力を用いて高性能化する(安価高有用)
- 限られた消費エネルギー・大きさ・重量で最大限の性能を引き出す(小型省資源高効率)

- ミニ四駆

(株)タミヤが製造販売するプラモデルのレーシングカー、単三電池2本とモーターで走行、専用のコースで速さを競う

- ミニ四駆AI大会実行委員会

代表 電気通信大学 西野順二先生

<https://sites.google.com/site/ai4wdcar/>



ミニ四駆AI大会と技術課題

ロボットとしてのメカ技術、駆動回路・センサ回路のための電子技術、ナビゲーションのための制御技術、そして、トータルな問題解決としての人工知能・システム工学技術や、情報処理技術、勝利の期待値を上げるためのゲーム理論など、未来の安価高有用システムのための基礎技術が含まれる

- コース学習
- 自己位置同定
- ブレーキ制御
- 最適行動計画
- モーター駆動回路
- センサシステム
- パワーウェイトレシオ



西野順二, "ミニ四駆AI大会が意味するもの", 第14回情報科学技術フォーラム(FIT2015), イベント企画「人工知能のため標準問題としてのゲーム研究」講演(4), 2015.

<https://www.ieice.org/publications/conference-FIT-DVDs/FIT2015/data/html/event/event164.pdf>

ミニ四駆AI用基板の例

- 小型・軽量・低消費電力のAI用電子基板が必要
- 一般的なマイコンベースの基板が主流
- RT Corp, AI Chip
 - 9軸慣性センサ、モータードライバ搭載、オプションでBluetooth
 - <https://www.rt-net.jp/products/aichip3>
 - ARM Cortex-M3, TI DRV8850RGYR, InvenSense MPU-9250, Microchip RN42XV, 3.7V LiPo
- Quantum, 春鹿ボード（現在は販売されていない）
 - 6軸慣性センサ、モーター用FET、uSD, コンパクト、拡張GPIO,I2C
 - <https://github.com/aks3g/AiModelCar>
 - Atmel XMEGA 32E5, LSM330DLC, 3.7V LiPo
- その他、Arduino, PIC, Raspberry Pi 等の流用

中島智晴, "ミニ四駆AI", 日本知能情報ファジィ学会誌 知能と情報, vol.29, no.4, pp136-140, 2017.

https://doi.org/10.3156/jsoft.29.4_136

いずみ研的プロジェクト

- 本業は Reconf ですっ！
 - 将来的には FPGA で制御したい
 - 将来的には FPGA 搭載の制御基板を開発する
- 学生にいきなり FPGA は辛い→まずはマイコンで検証
- しかし、市販品はブラックボックス

方針

- 独自基板 → ノウハウを蓄積
- Arduino → 熟成されたツール & ライブラリ、膨大な資料
- Grove like connection → センサ拡張性、試行 & 検討

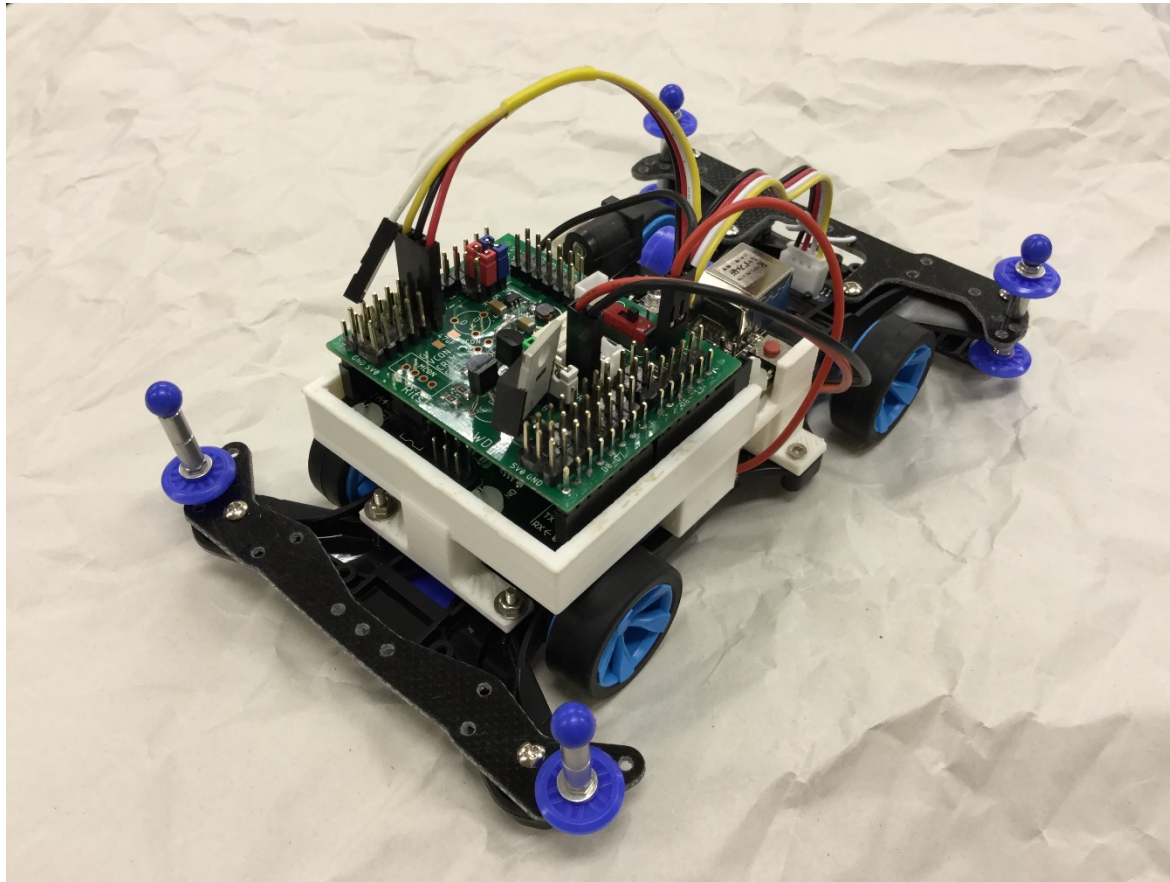
AI制御の教育研究のための Arduino互換 ミニ四駆 Rizm mini4WDuino

立命いずみ研的 ミニ四駆 × Arduino

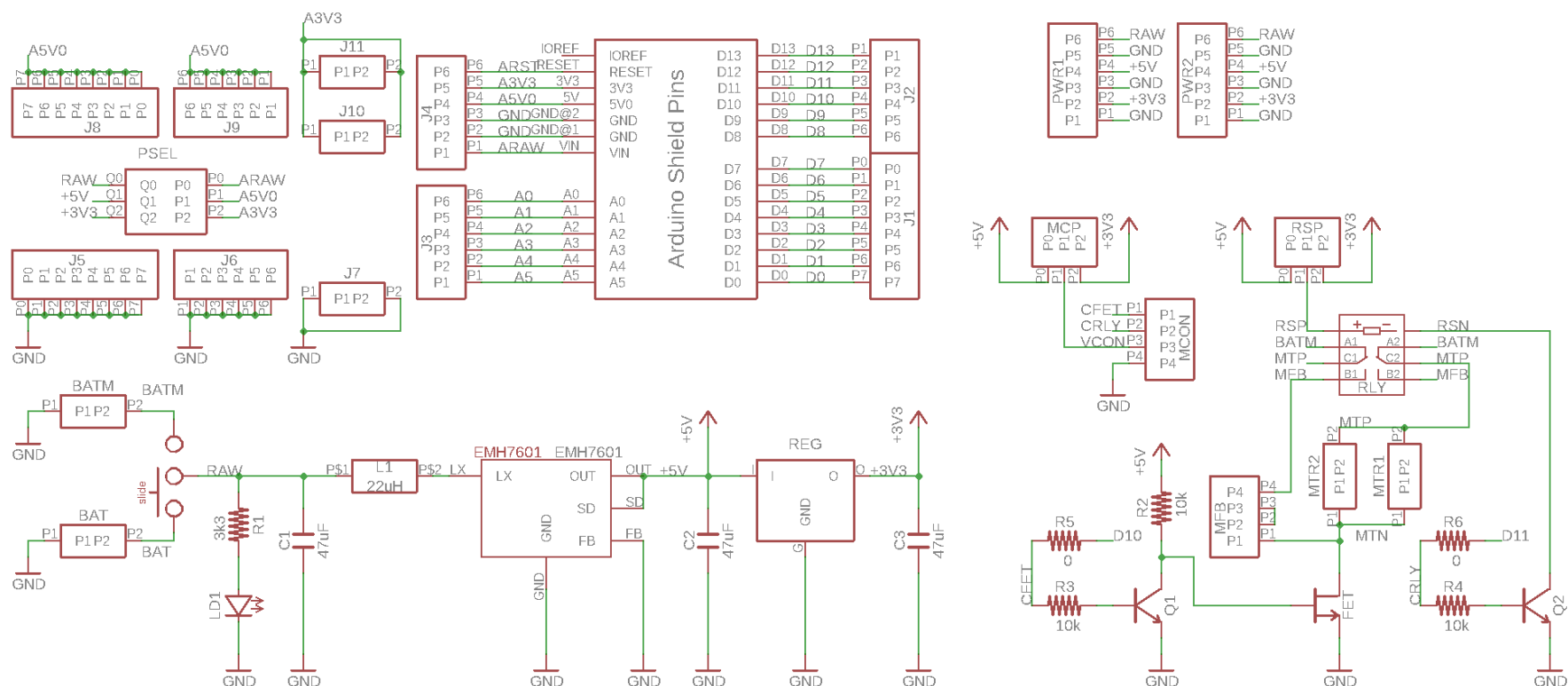
立命館大学 理工学部 電子情報工学科
いずみ研究室

Rizm mini4WDuino 1

- 目的: モーター制御回路やセンサモジュールの動作確認
- Arduino Uno + 自作シールド + 自作マウンタ + AR chassis

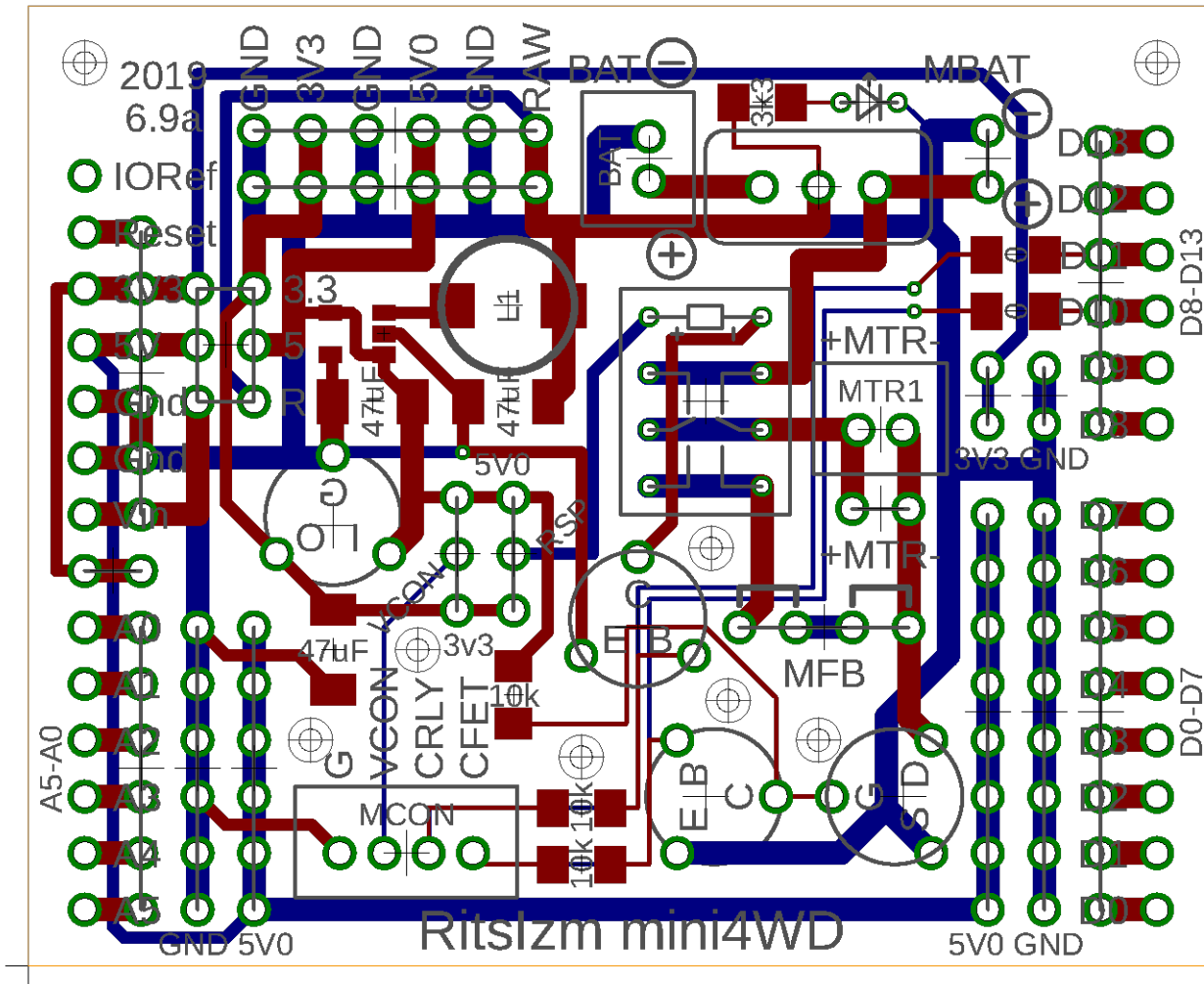


独自シールド～回路図



eagle 設計データ [mini4WD1_eagle.zip](#)

自作シールド～基板

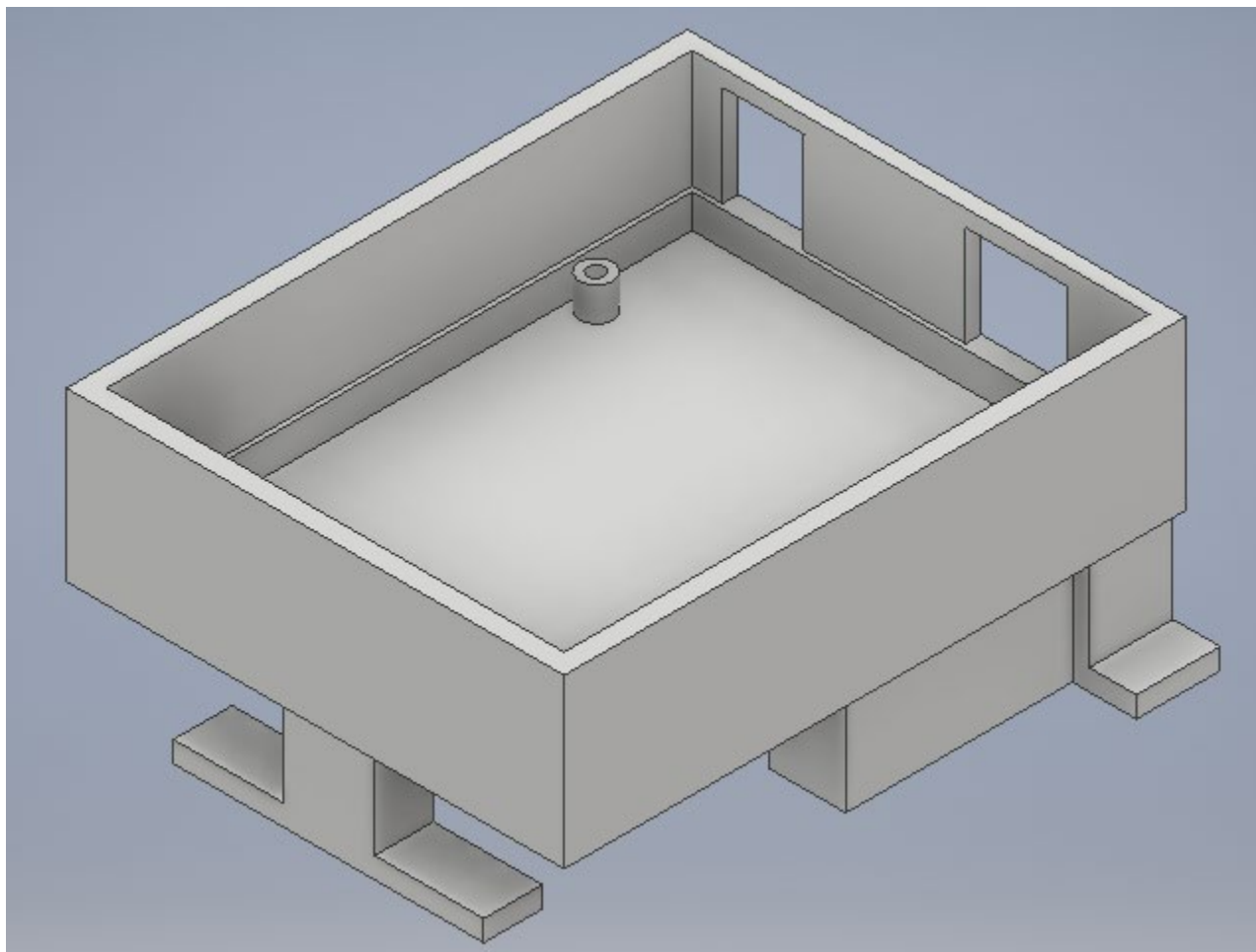


eagle 設計データ [mini4WD1_eagle.zip](#)

初期型の試行

- 昇圧回路により +3.3V, +5.0V 電源を生成
 - ロジック用バッテリー追加なしでも動作可能
 - EMH7601 (供給 0.93V以上、出力 3.3V or 5.0V)
- 動作電圧、オン抵抗の低い FET でモーター駆動
 - IRLB3034 (動作電圧 4.5V~、オン抵抗 1.4mΩ)
- リレーを用いた短絡ブレーキ
 - G6K-2P DC3 (動作電圧 3V、二回線)
- センサによるフィードバック
 - Grove Light Sensor (P) v1.1 (アンプ付、アナログ入力)

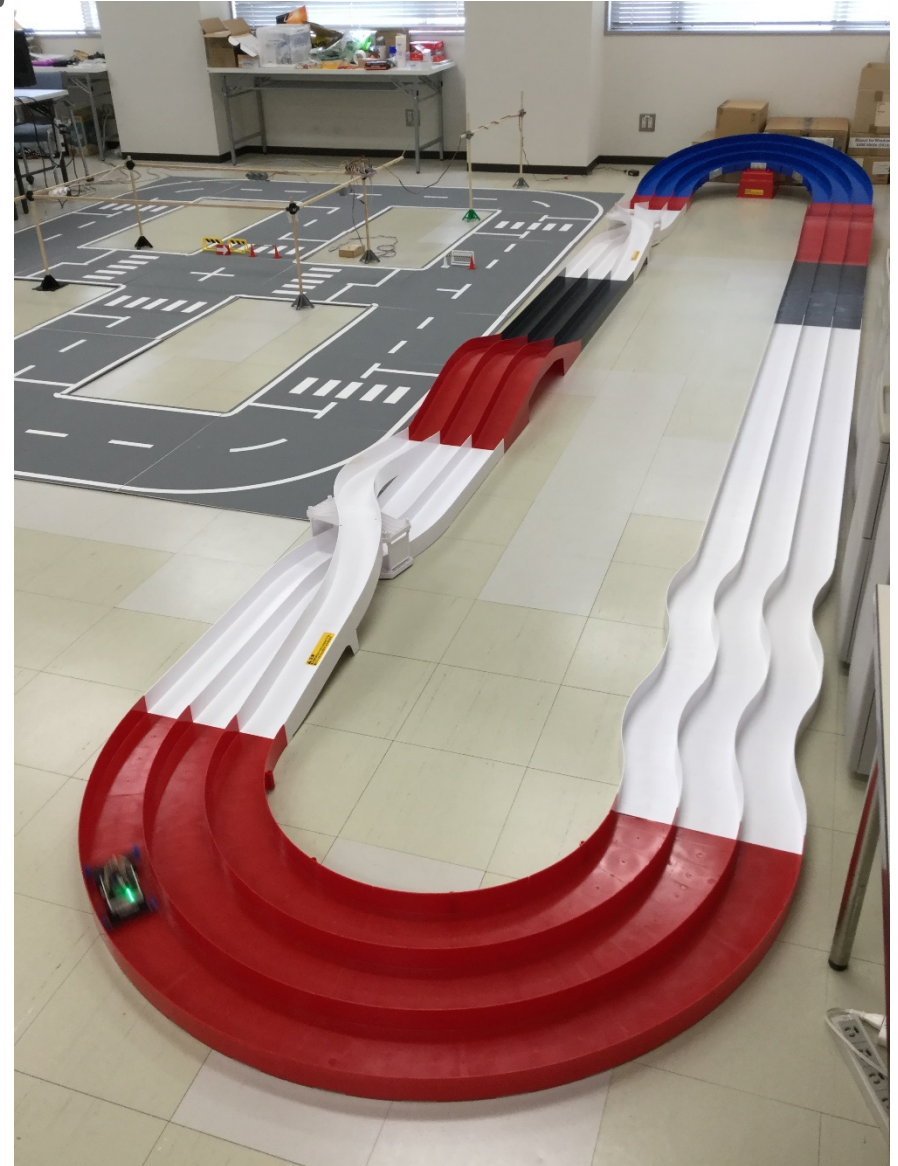
自作マウンタ



初期型テスト走行

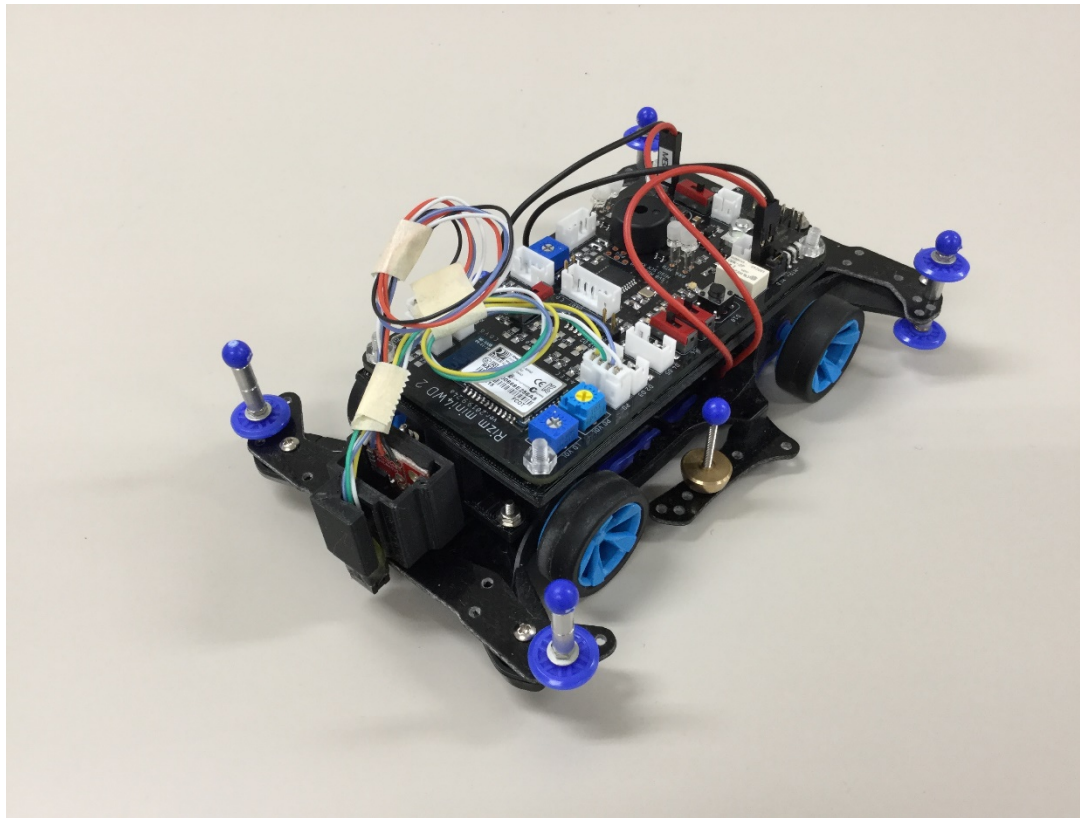
- マウンタ...OK
 - 修正 & 補強
- 電源回り...OK
 - 電池共有 + 昇圧回路
- モーター制御...OK
 - Power MOS FET で PWM Accel
 - Relay で Short Brake
- マイコンプログラム...OK
 - 光センサでフィードバック制御
 - LEDで状態表示

- 通常版ミニ四駆走行 [mini4WD_testrun.mov](#) [YouTube](#)
- 魔改造ミニ四駆走行 [mini4WD1_testrun2.mov](#) [YouTube](#)
- ソースコード [mini4WD1.ino](#)

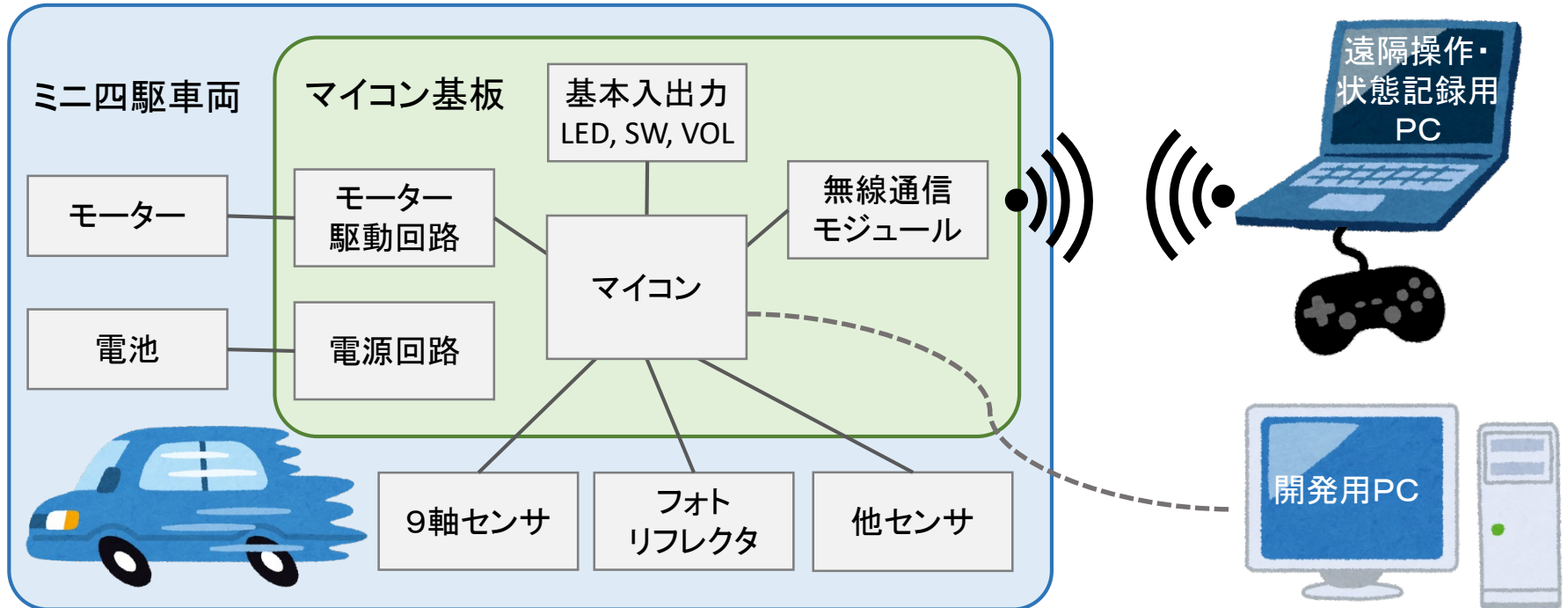


Rizm mini4WDuino 2

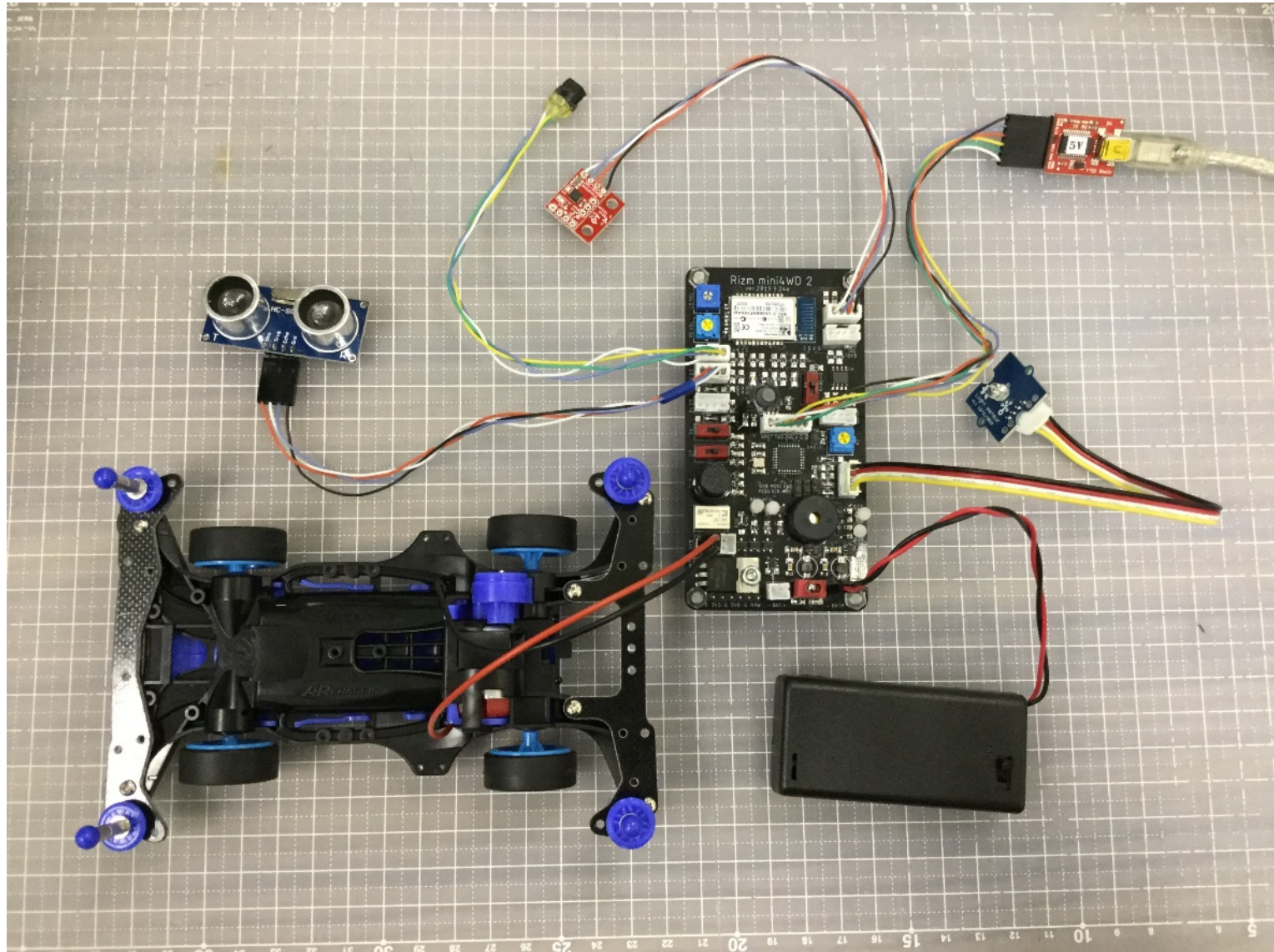
- 目的: 各種センサ類を搭載して走行し、試行評価する
- Arduino 互換ボード + 自作マウンタ + AR chassis
- 現在、テスト走行・評価中



システムの概要



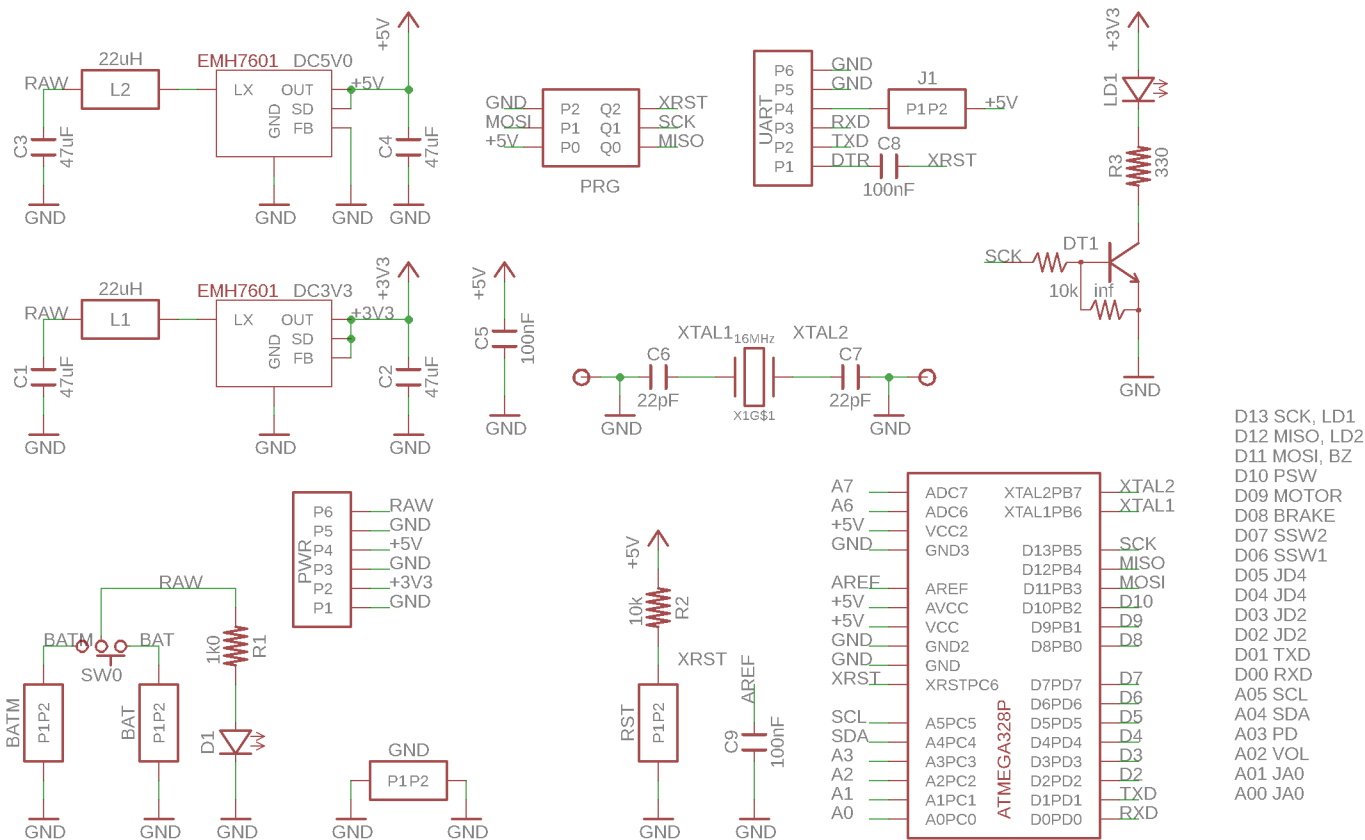
Components (1)



Components

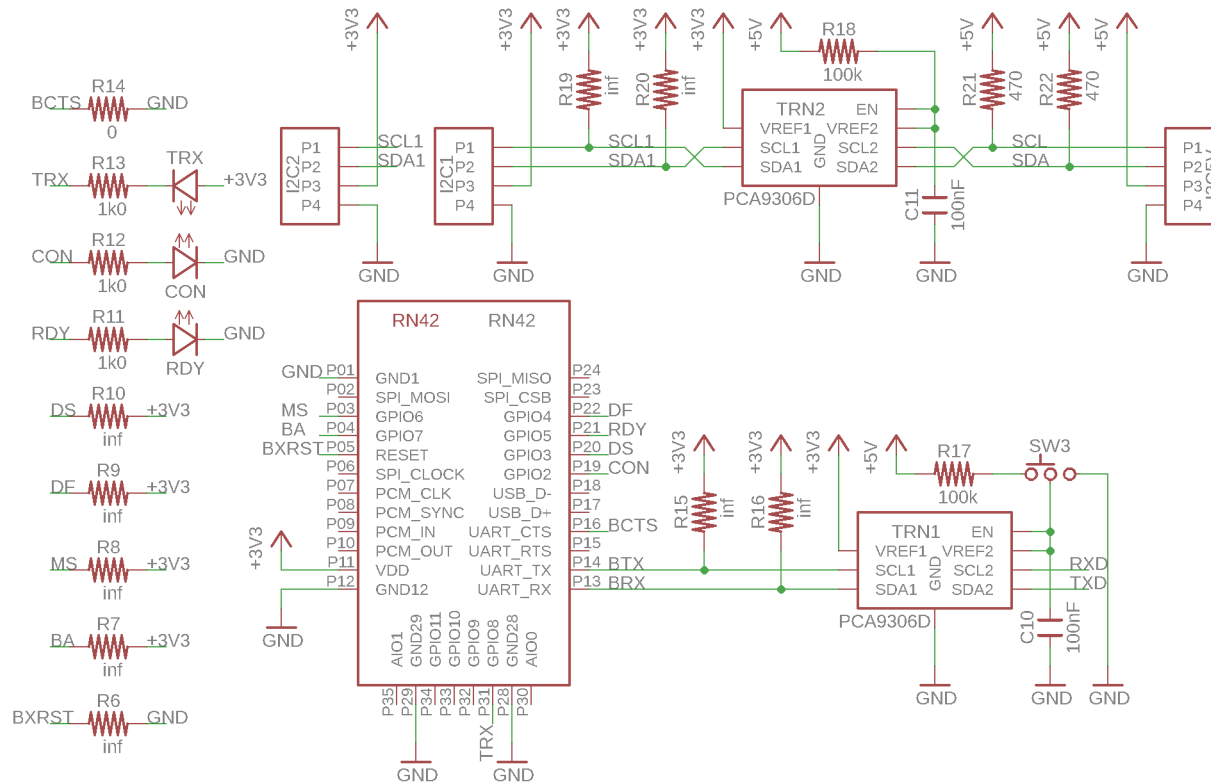
- マイコン ～ ATmega328, 最も普及している Arduino Uno ほかと互換
- モーター制御 ～ power FET (PWM) + relay (short brake)
- 電源 ～ モーター用電池または専用バッテリー + DC/DC
- 基本入出力 ～ LED (状態表示など)、スライドスイッチ (モード設定など)、プッシュスイッチ (スタートボタンなど)、ブザー (警告など)
- 通信 ～ RN42-I/RM, Bluetooth-UART モジュール
- センサ ～ Grove like connection で市販センサモジュール類を容易に接続可
- 他 ～ シンプルな Digital I/O, Analog I を利用可

独自基板～回路図1 マイコン、電源

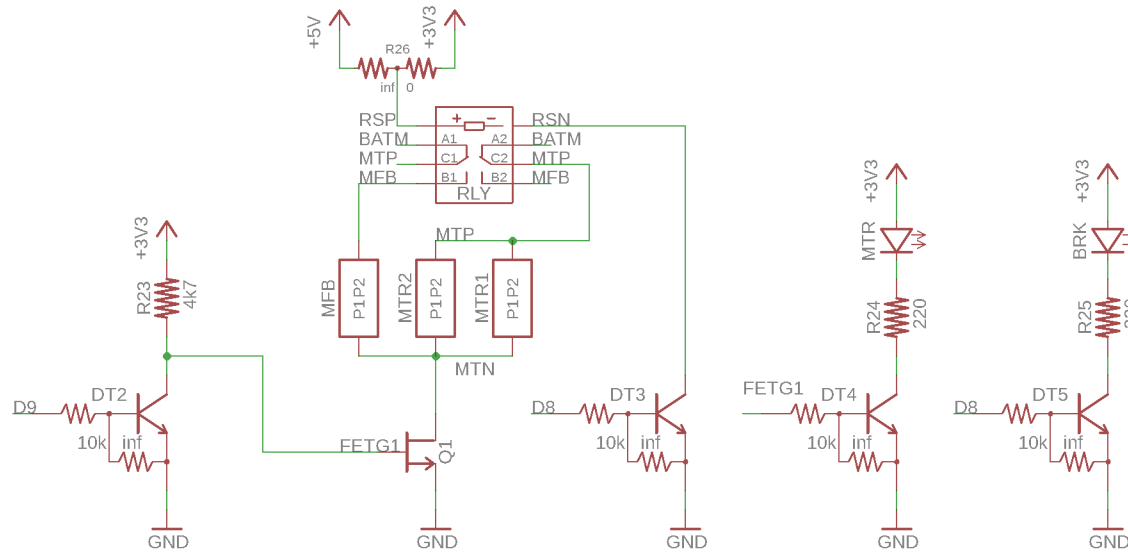


eagle 設計データ [mini4WD2_eagle.zip](#)

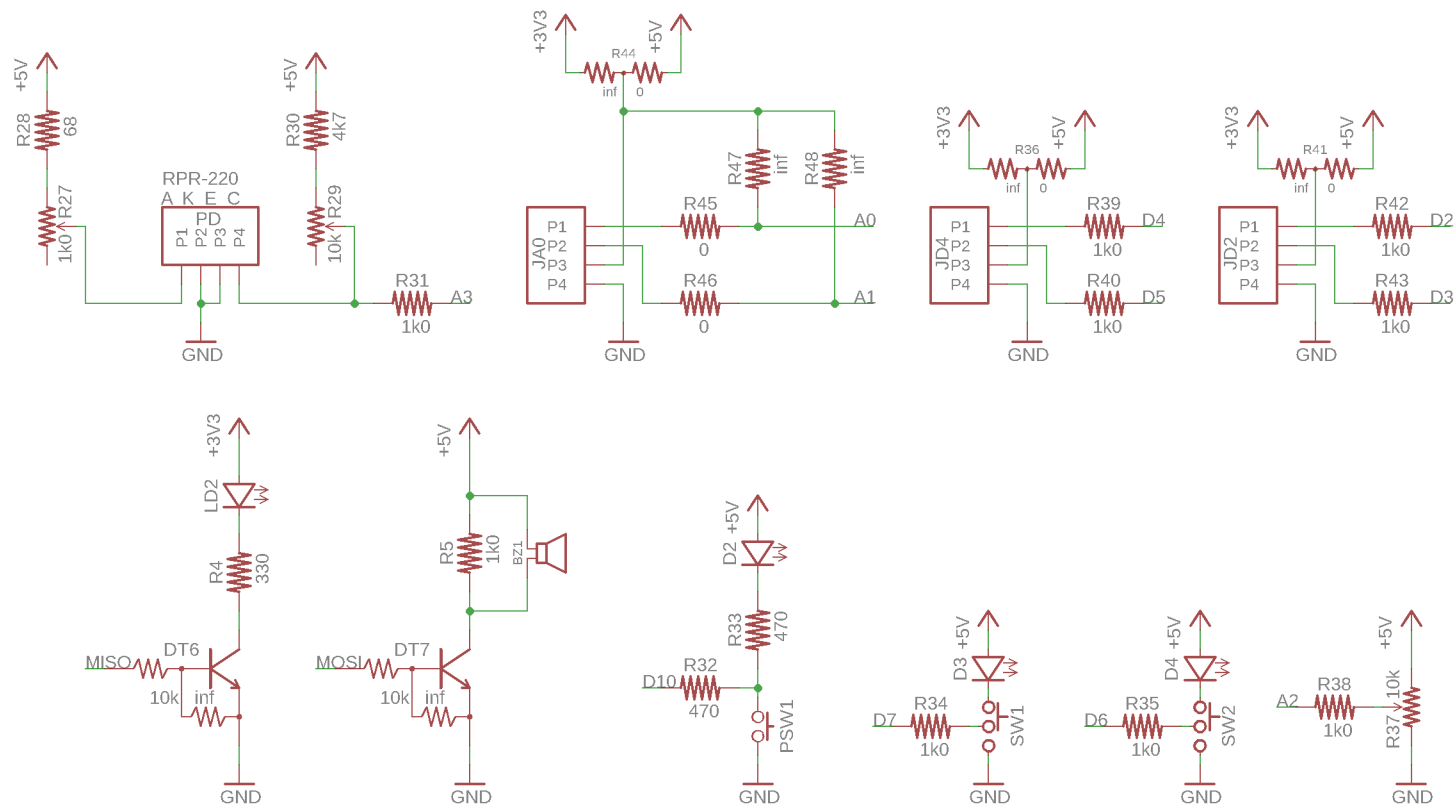
独自基板～回路図2 Bluetooth, I2C



独自基板～回路図3 モータ制御



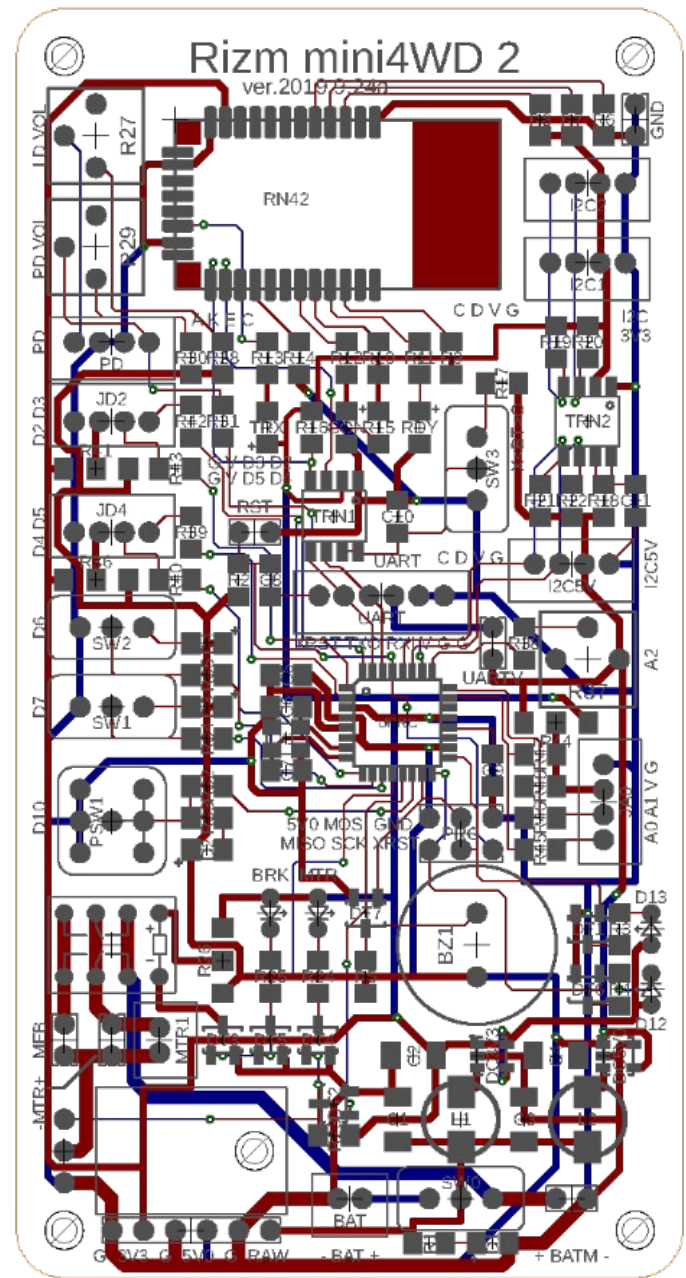
独自基板～回路図4 基本入出力、拡張コネクタ



R 独自基板

- 両面2層 最小線幅5mil
- 片面実装
- 2100mil × 4000mil
- 手作業で半田付けできる部品サイズと配置
- P-ban.com で製造

eagle 設計データ [mini4WD2_eagle.zip](#)



電源系

• 入力

- ロジック用バッテリー (BAT) 1.2～3.3V
- モーター用バッテリー (BATM) 単三乾電池2本
- モーター用バッテリーをロジックに使うことも可 (SW0)

注 ... ロジック用に Li-Po 3.7V バッテリーを使う場合はダイオード等で降圧が必要

• 電源系統

- ロジック用バッテリーからDC/DC昇圧回路で5.0Vと3.3Vを生成
- 5.0V...マイコン(16MHz動作のため)および直近の周辺
- 3.3V...Bluetooth-UART, I2C
- 一部、半田短絡により選択可 (R26, R36, R41, R44)

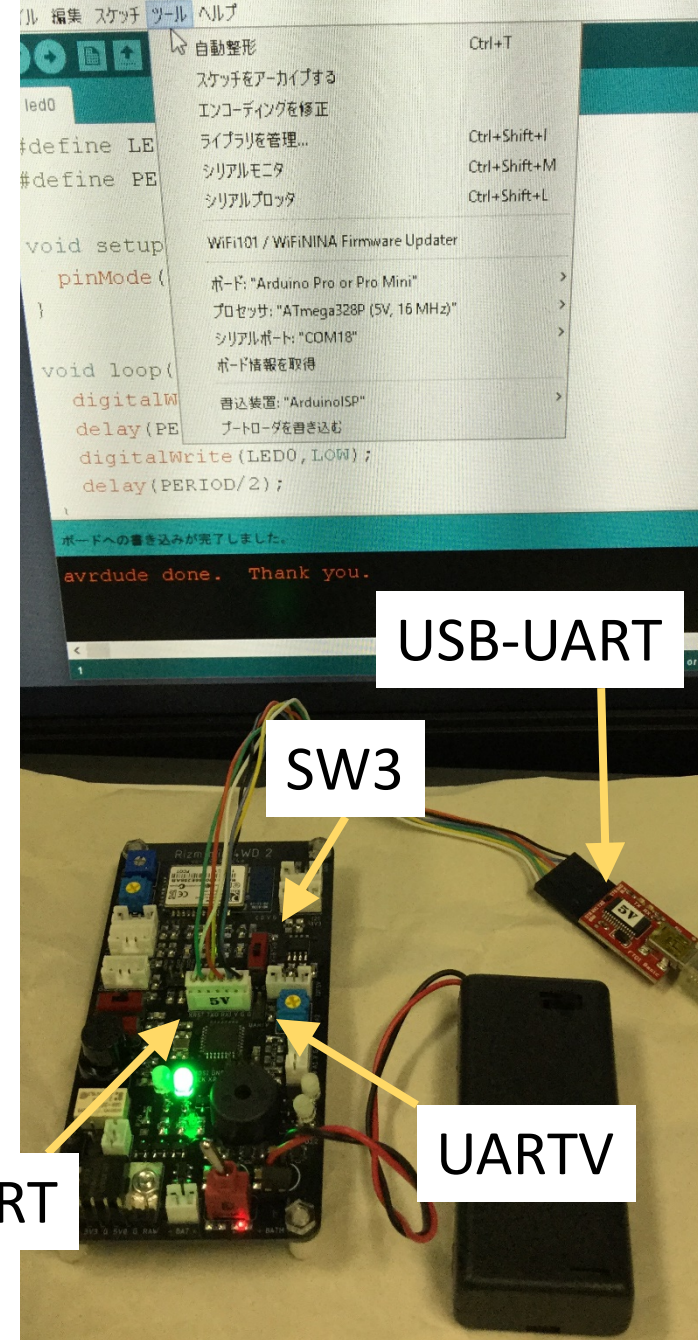


BAT SW0 BATM

プログラム書き込み

• Program 書き込み

- Bluetooth-UART を切り離す(SW3)
- UARTV は開放しておく
- 電源を接続
- UARTを接続
 - FTDI USB-UARTモジュール(5V設定) + JST PH6 コネクタケーブル
- Arduino IDE からプログラムを書きこむ
 - ボード: Arduino Pro or Pro Mini
 - プロセッサ: ATmega328P (5V, 16MHz)
 - 書込装置: ArduinoISP



R (Arduino bootloader 書き込み)

- ボード完成時に1度だけの作業
- 別途 Arduino を用意し、書込器にする
 - ファイル→スケッチの例→ArduinoISP
 - 書込器 Arduino をターゲットとして設定
 - コンパイル&転送
- PRGに接続する
 - 右表のとおり
- bootloader を書き込む
 - ボード: Arduino Pro or Pro Mini
 - プロセッサ: ATmega328P (5V, 16MHz)
 - 書込装置: Arduino **as** ISP
 - ブートローダを書き込む

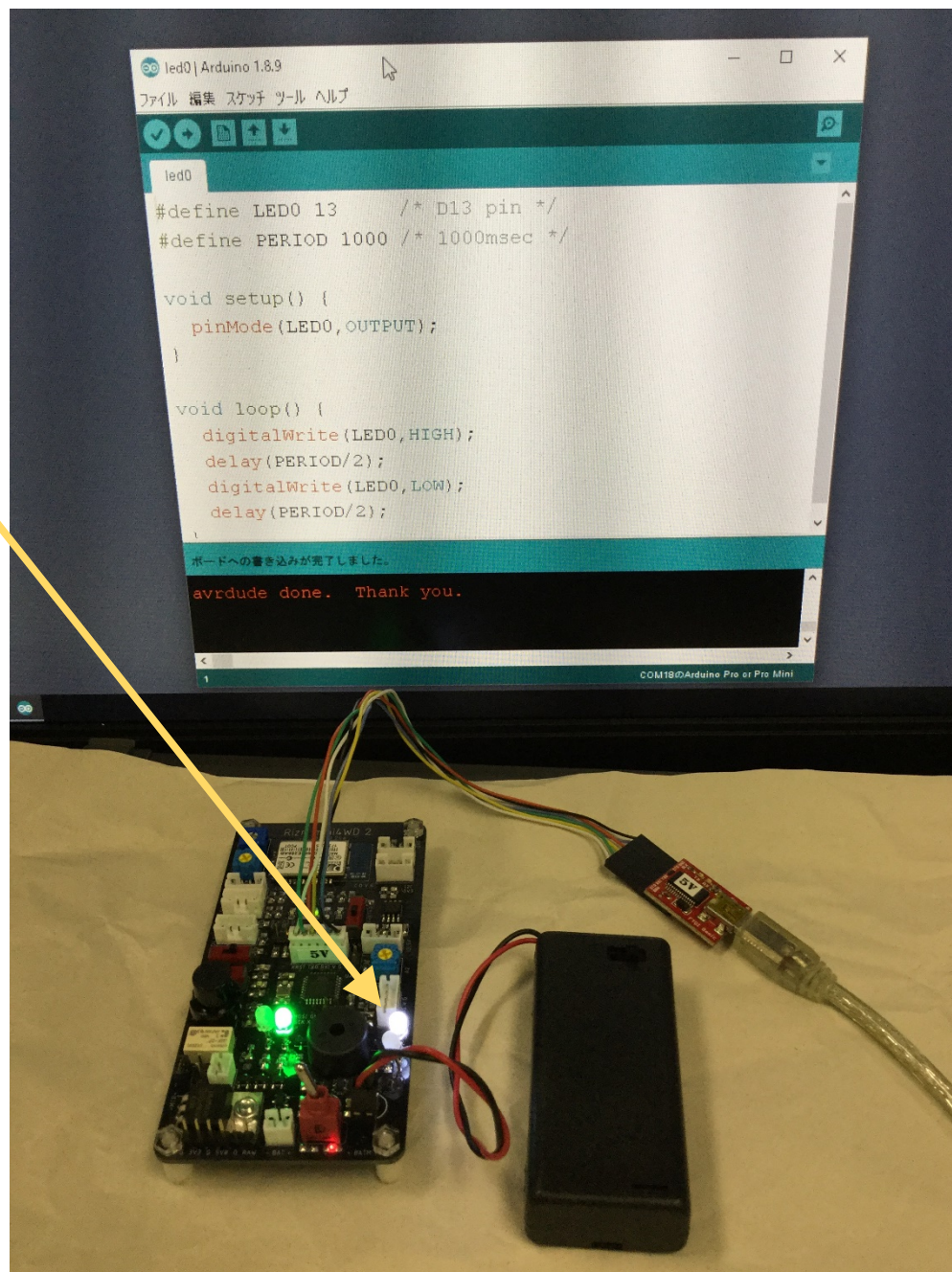
書込器	mini4WDuino PRG
5V	5V
D11	MOSI
GND	GND
D12	MISO
D13	SCK
D10	XRST

R まずは“Lチカ”

- Arduino のはじめの一歩
- 13番ピン LED の点滅
- サンプルそのまま

```
void setup() {  
  pinMode(13,OUTPUT);  
}
```

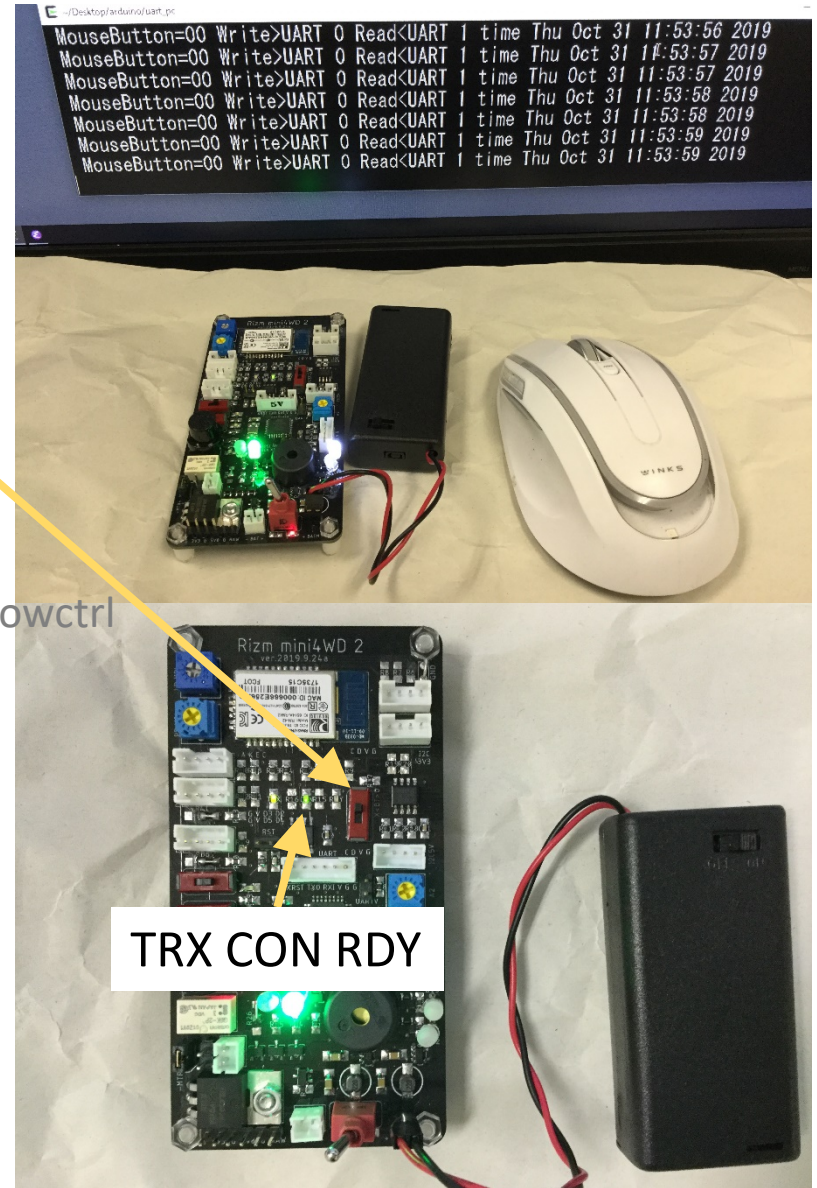
```
void loop() {  
  digitalWrite(13,HIGH);  
  delay(500);  
  digitalWrite(13,LOW);  
  delay(500);  
}
```



Bluetooth-UART 接続

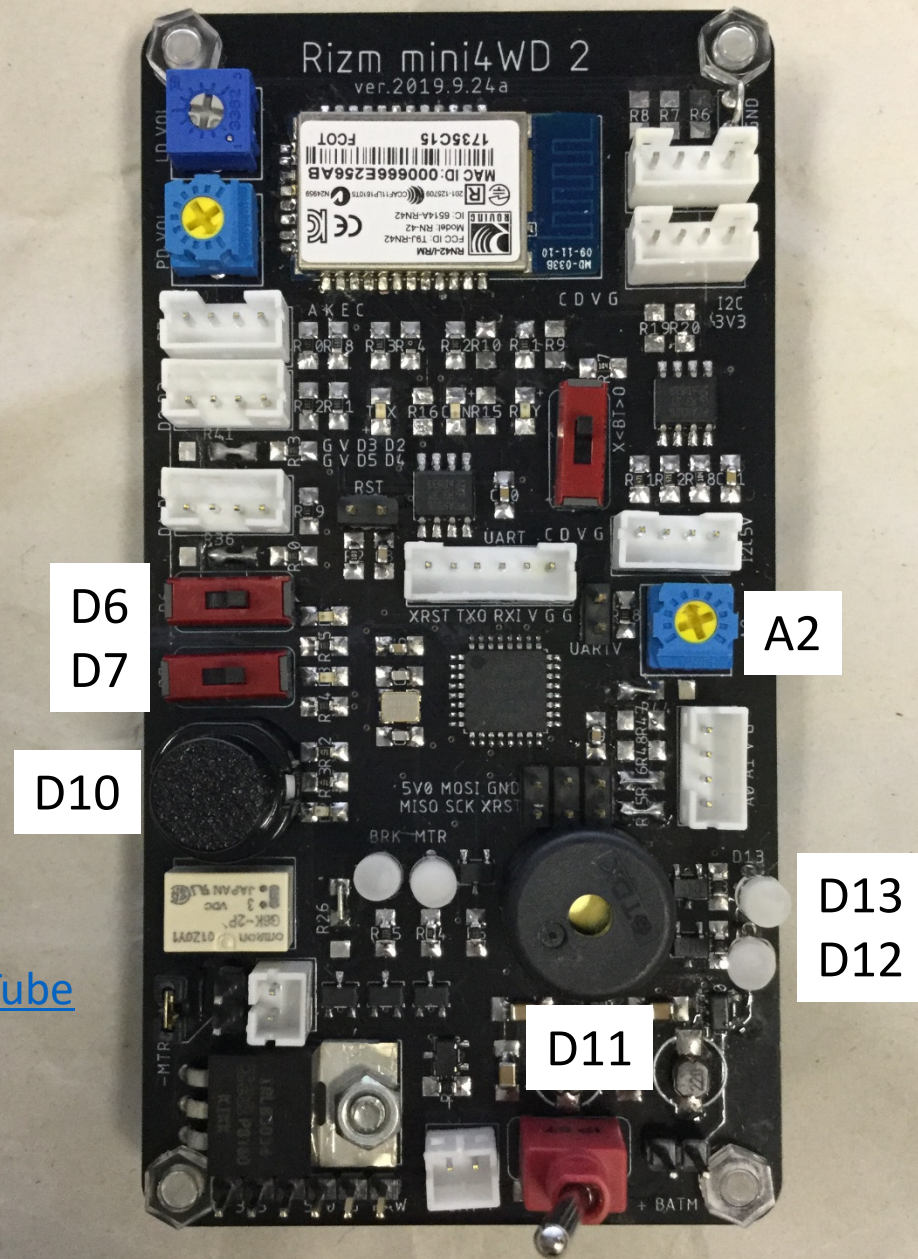
- RN42-I/RM を搭載
- 電圧変換IC経由で接続されている
- Bluetooth-UARTを接続する(SW3)
- Arduino 標準 TRX (D1,D0) に接続
- 既定値
 - デバイス名 RNBT-[MAC 下四桁]
 - 設定 115200bps, 8bit, 1stopbit, no-parity, no flowctrl
- 状態表示LED
 - 接続待 RDY 点滅
 - 接続中 CON 点灯
 - 通信中 TRX 点滅

- 動作の様子 [mini4WD2bd_btuart.mov](#) [YouTube](#)
- ソースコード [mini4WD2bd_btuart.zip](#)



基本入出力

- LED ... D13, D12
 - Slide SW ... D7, D6
 - ON時 LED としても使用可
 - Push SW ... D10
 - OFF時 LED としても使用可
 - Buzzer ... D11
 - Volume ... A2
- 動作の様子 [mini4WD2bd_basicio.mov](#) [YouTube](#)
- ソースコード [mini4WD2bd_basicio.zip](#)



モーター駆動回路

• 接続

- MTR モーター
- MFB 短絡

• アクセル (D9)

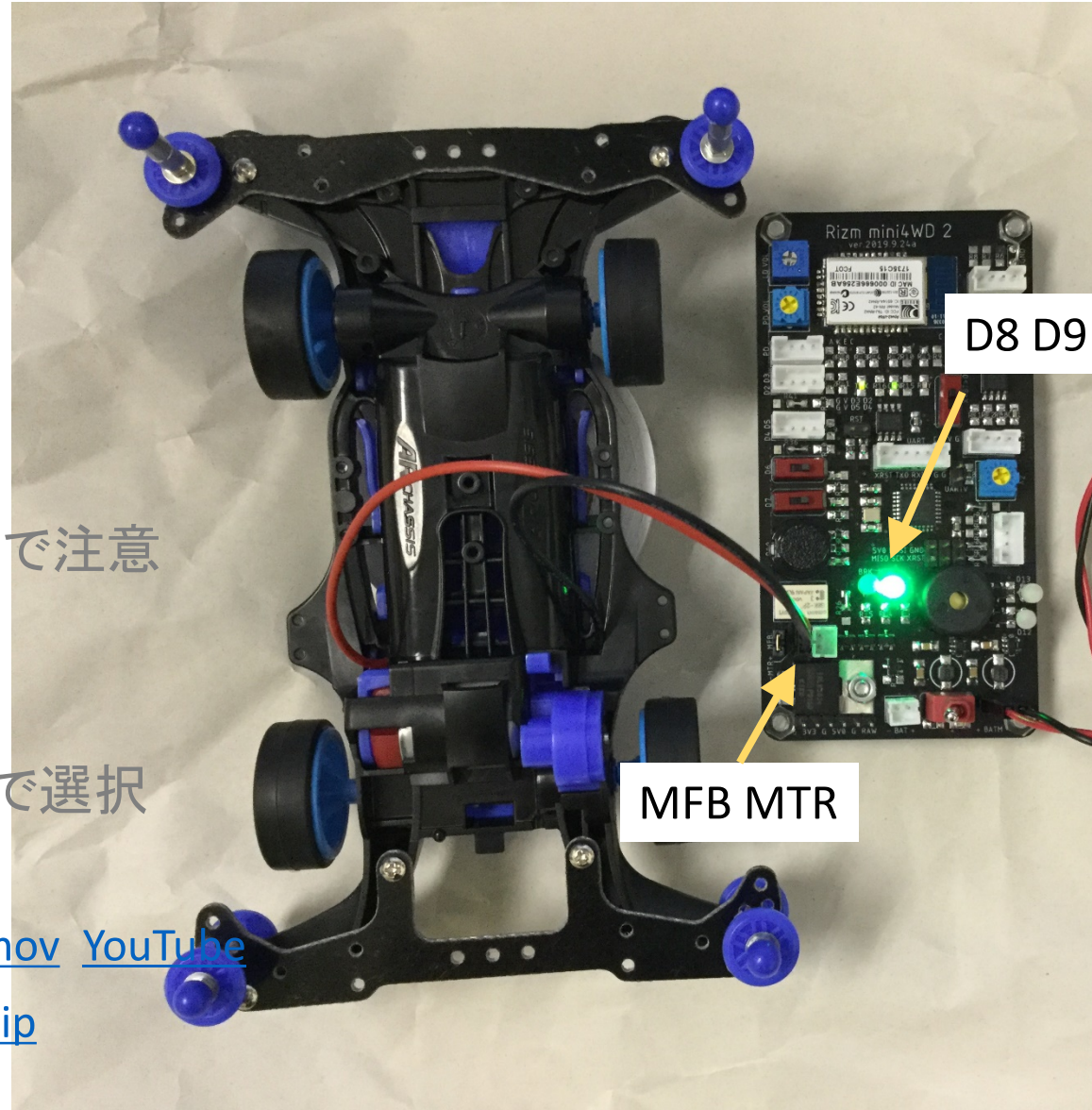
- LOW:ON, HIGH:OFF
- PWMで出力制御
- 起動時に車輪が回るので注意
- 状態LED付

• 短絡ブレーキ (D8)

- リレー電源は半田短絡で選択
- 状態LED付

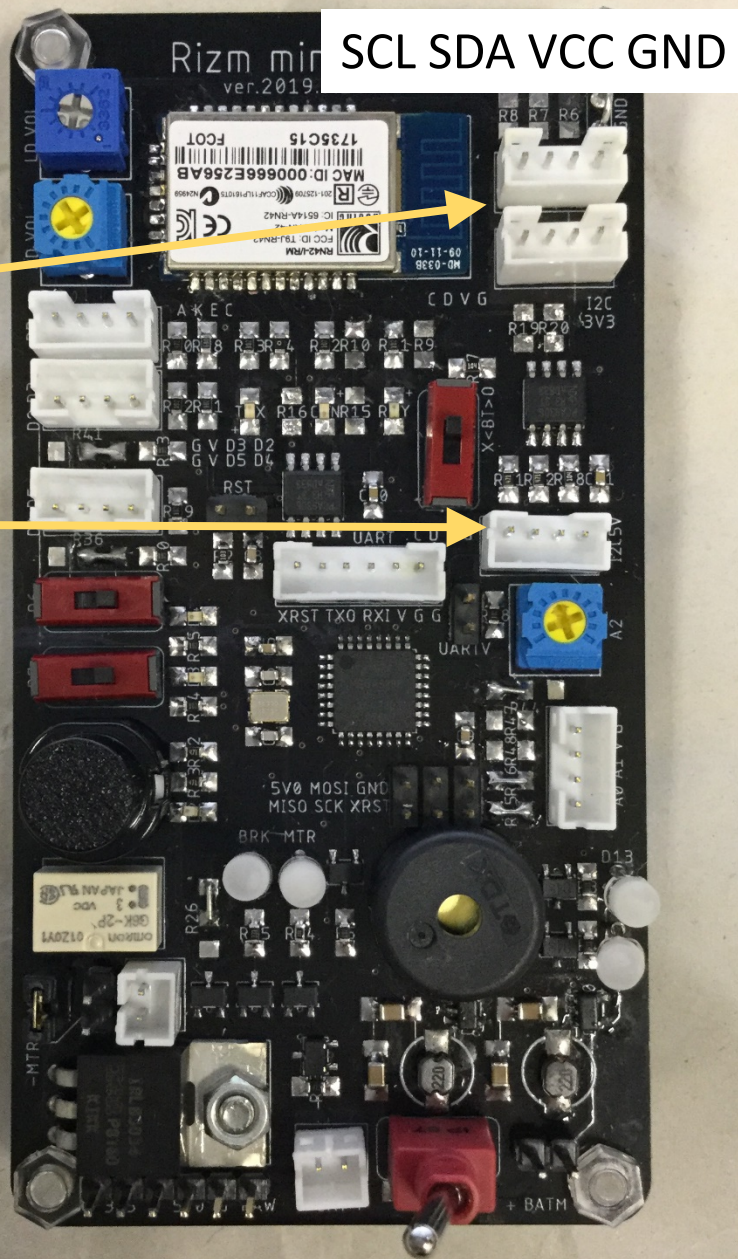
➤ 動作の様子 [mini4WD2bd_mtr.mov](#) [YouTube](#)

➤ ソースコード [mini4WD2bd_mtr.zip](#)



I2C

- Arduino 標準A5,A4に接続
- 3.3V用コネクタ × 2
 - 電圧変換IC経由
- 5.0V用コネクタ × 1
- JST PH4
- [SCL, SDA, VCC, GND]
- Grove 可
(互換ではないがコネクタが挿せなくはない)

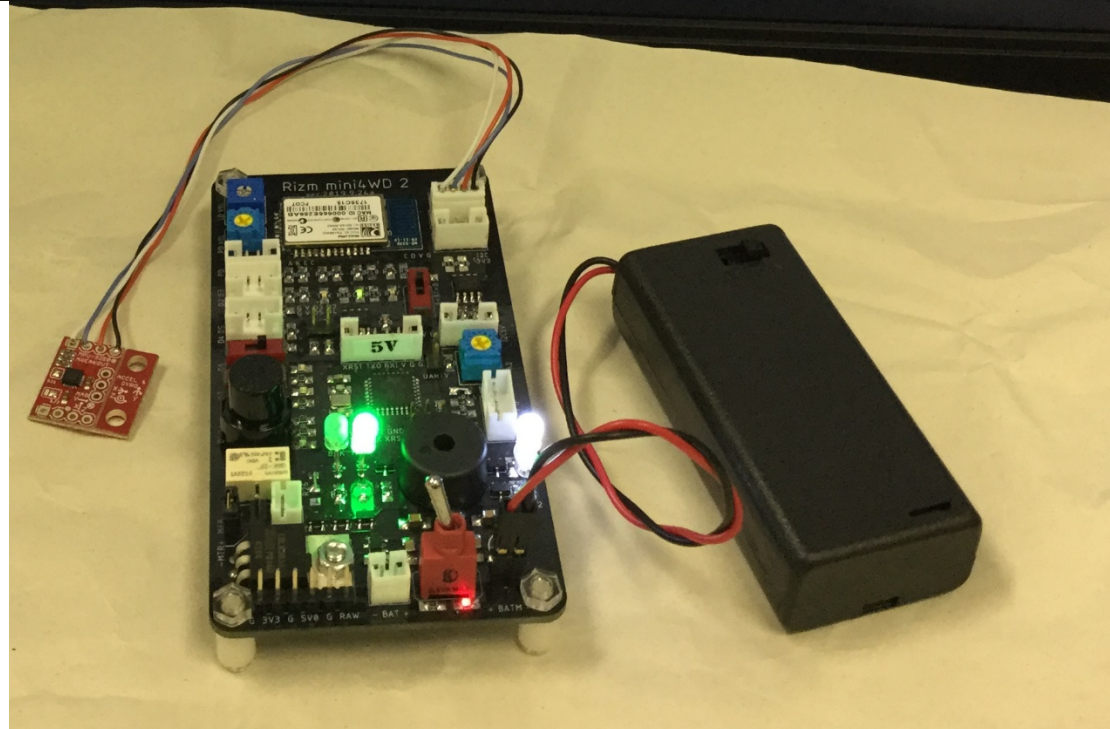


I2C 接続例: 9軸センサ

```

10 Acc1 Cnf2 0
37 IntPinCfg 2
0A Mag Ctrl1 16
end setup
time=2011 Sens -008c +014b +080d +0a00 -0020 +0002 -0010 -0034 +001a -001a +0010 2ce
time=4016 Sens -0099 +0143 +080a +09e0 -001b +0003 -000d -0038 +001c -0022 +0010 270
time=6021 Sens -0089 +0146 +07ec +0a30 -0022 +0003 -000b -003a +0022 -001c +0010 25f
time=8025 Sens -008d +014d +0800 +0a00 -0020 +0004 -000d -0030 +0018 -0022 +0010 256
time=10030 Sens -0092 +0142 +0806 +0a00 -0022 -0004 -000f -003b +001d -001d +0010 253
time=12035 Sens -0091 +0147 +07fc +0a50 -0020 +0006 -000d -002c +001e -001e +0010 250
time=14039 Sens -0093 +0148 +0805 +09f0 -001d +0002 -0010 -0032 +001c -0016 +0010 252
time=16044 Sens -008c +014b +0800 +0a60 -001f -0001 -0008 -0036 +001c -0020 +0010 24e
  
```

- MPU-9250
 - 加速度XYZ
 - 角速度XYZ
 - 磁気XYZ
 - 温度
- 3.3V I2C接続
 - ソースコード
[mini4WD2bd_accl.ino](#)



拡張用I/Oコネクタ

- いずれも JST PH4, Grove 可

- デジタル

[D2,D3,VCC,GND]

[D4,D5,VCC,GND]

- 電源は半田短絡で選択

- アナログ

[A0,A1,VCC,GND]

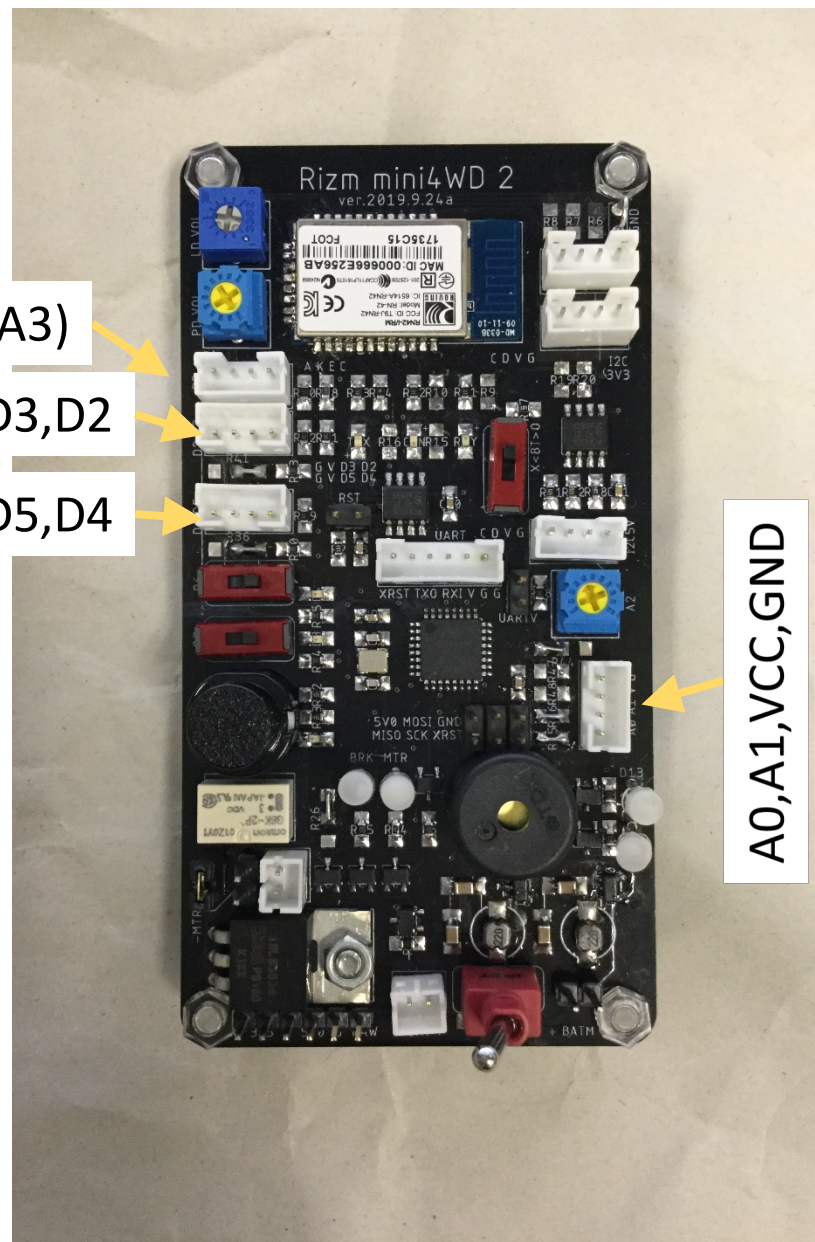
- 半田付けで電源、プルアップ抵抗、直列抵抗を設定

[A,K,E,C (A3)]

- 反射型光センサ用

A,K,E,C (A3)
GND,VCC,D3,D2
GND,VCC,D5,D4

A0,A1,VCC,GND

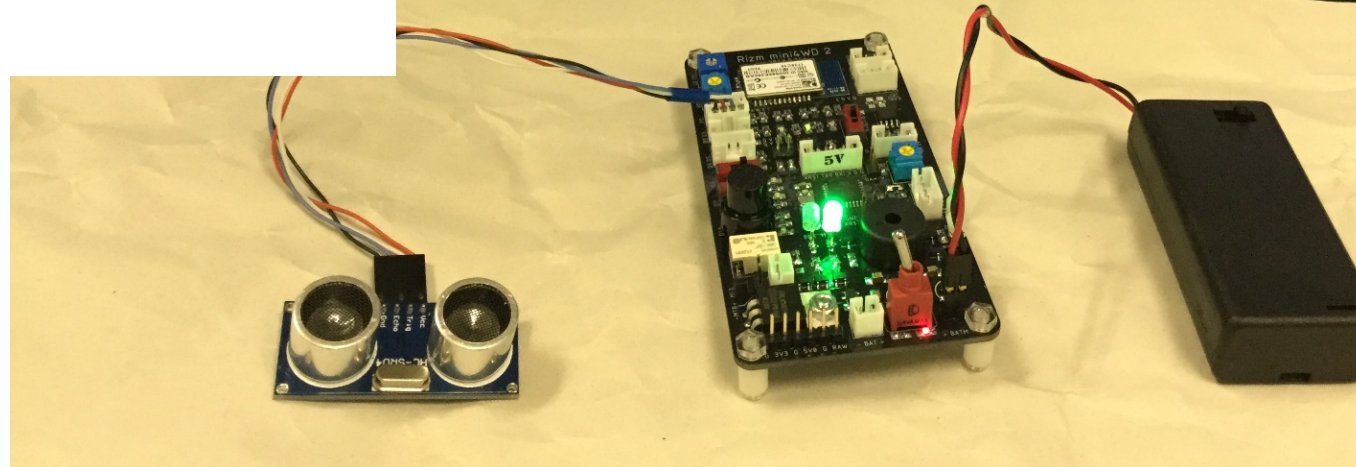


デジタル接続例：超音波距離センサ

- HC-SR04
- Trigger 信号で超音波発生
- Echo 信号到着までの時間で計測 pulseIn()
- 5V電源

➤ ソースコード [mini4WD2bd_us.ino](#)

```
Distance = 64 ㎜
Distance = 47 ㎜
Distance = 34 ㎜
Distance = 191 ㎜
Distance = 337 ㎜
Distance = 294 ㎜
Distance = 269 ㎜
Distance = 123 ㎜
Distance = 108 ㎜
Distance = 99 ㎜
Distance = 60 ㎜
Distance = 54 ㎜
Distance = 107 ㎜
Distance = 153 ㎜
Distance = 196 ㎜
Distance = 223 ㎜
Distance = 208 ㎜
Distance = 312 ㎜
Distance = 2172 ㎜
Distance = 2188 ㎜
Distance = 2176 ㎜
Distance = 2176 ㎜
Distance = 2180 ㎜
```



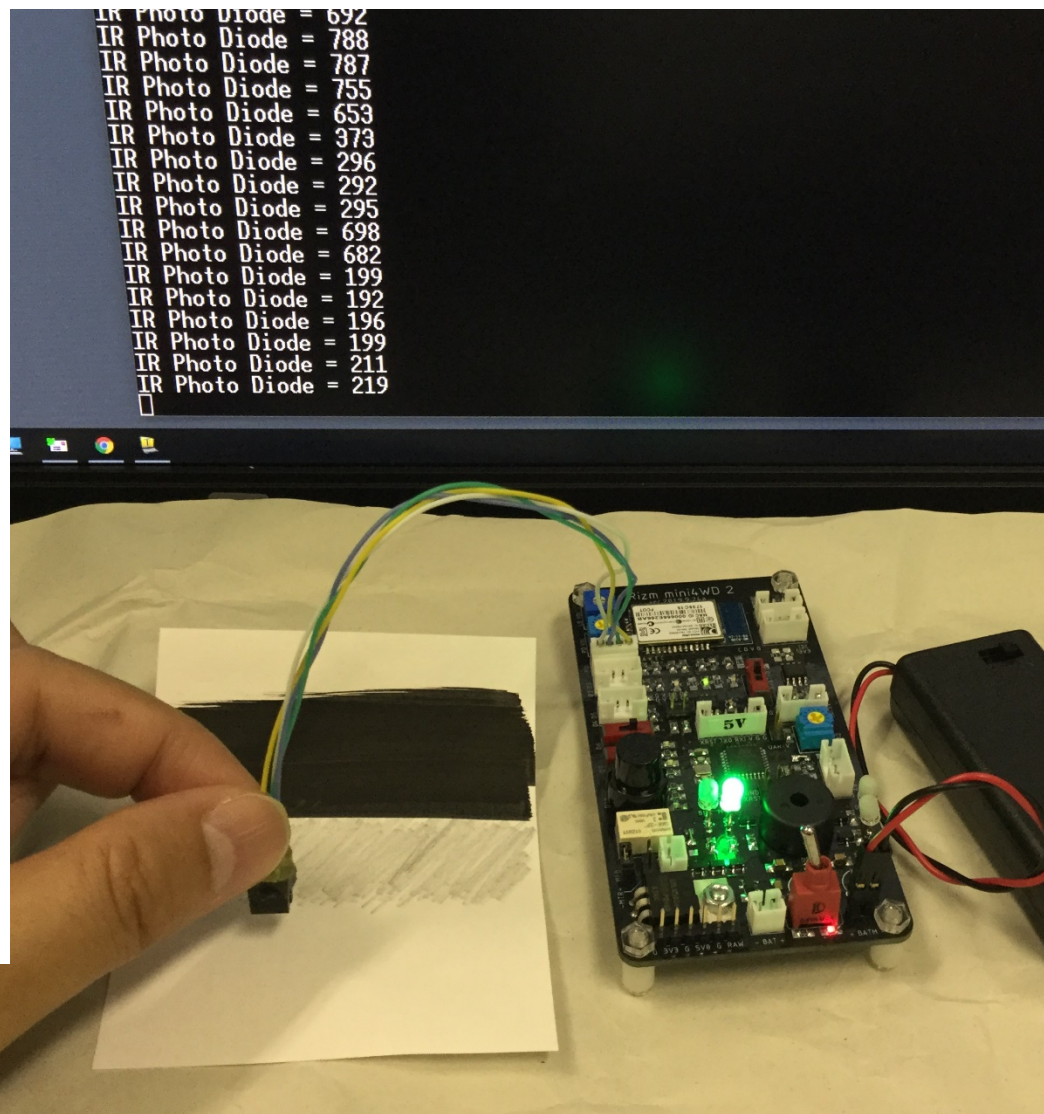
アナログ接続例：光センサ



- Grove Light Sensor v1.1
- 明るさ→電圧
- analogRead()
- 5V電源
- ソースコード [mini4WD2bd_pd.ino](#)

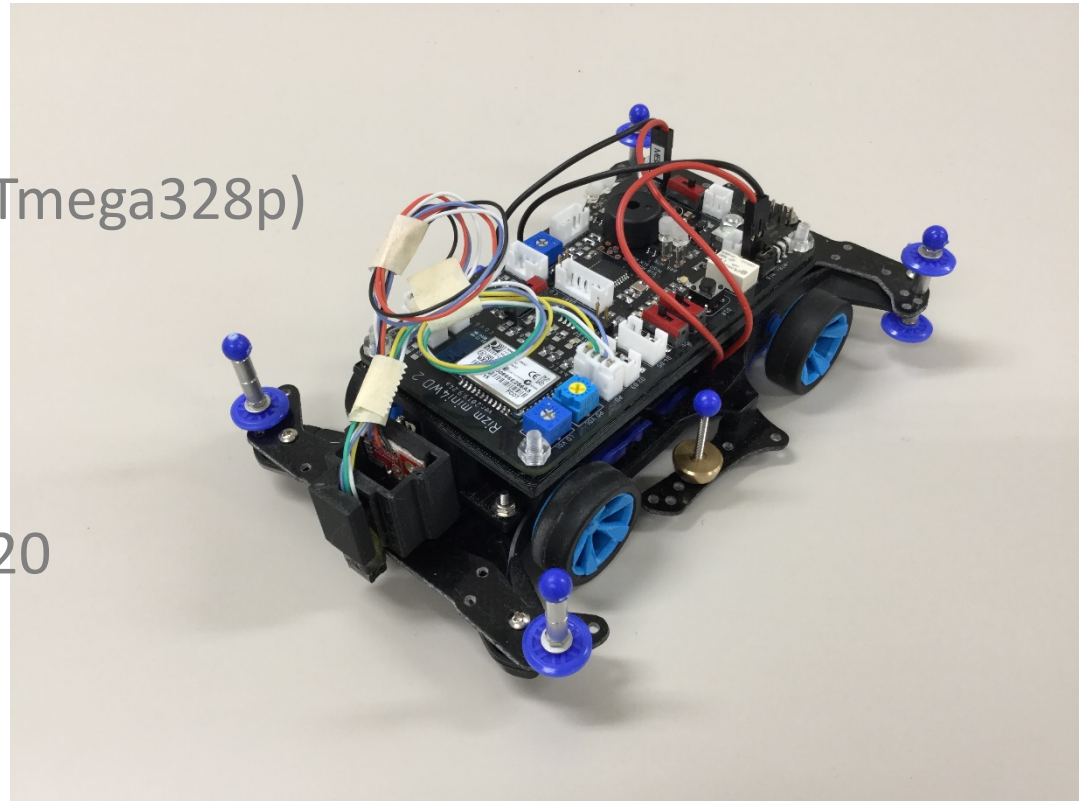
R アナログ接続例：反射型光センサ

- Rohm RPR-220
- PD port [A,K,E,C] に接続
- 発光ダイオード
 - A=anode, K=cathode 接続
 - LD VOL で光量調整
- フォトトランジスタ
 - E=emitter, C=collector 接続
 - PD VOL で感度調整
- 明るさ→電圧
- analogRead()
- 5V電源
- ソースコード [mini4WD2bd_irpd.ino](#)

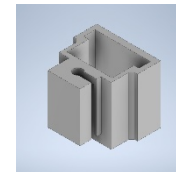
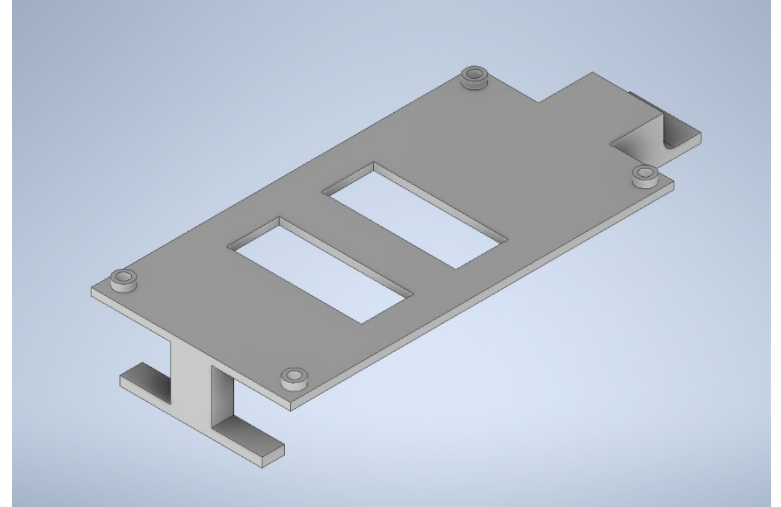


車両の開発

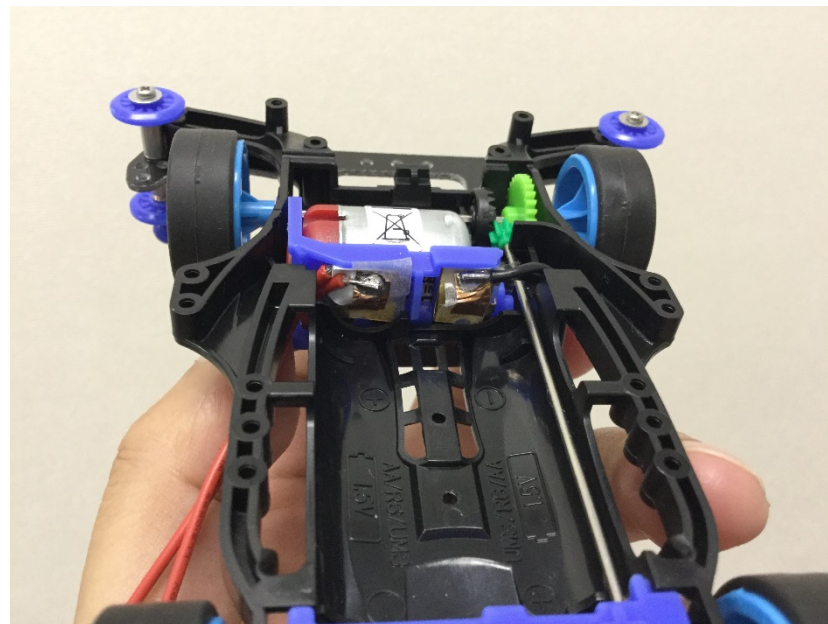
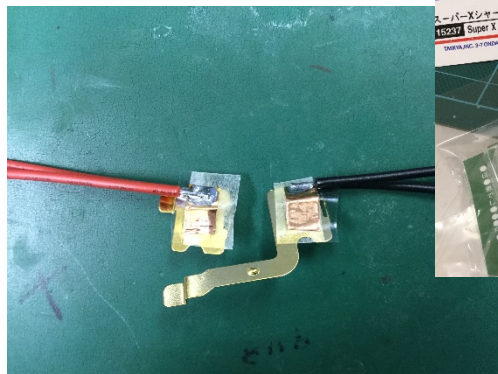
- 車体
 - TAMIYA ミニ四駆 AR chassis
 - Aero Avante, Startar Pack ベース
- 制御基板
 - Ritsizm mini4WD2 基板
 - Arduino互換マイコン(ATmega328p)
 - Bluetooth-UART (RN42)
- センサモジュール
 - 9軸センサ MP-9250
 - 反射型光センサ RPR-220



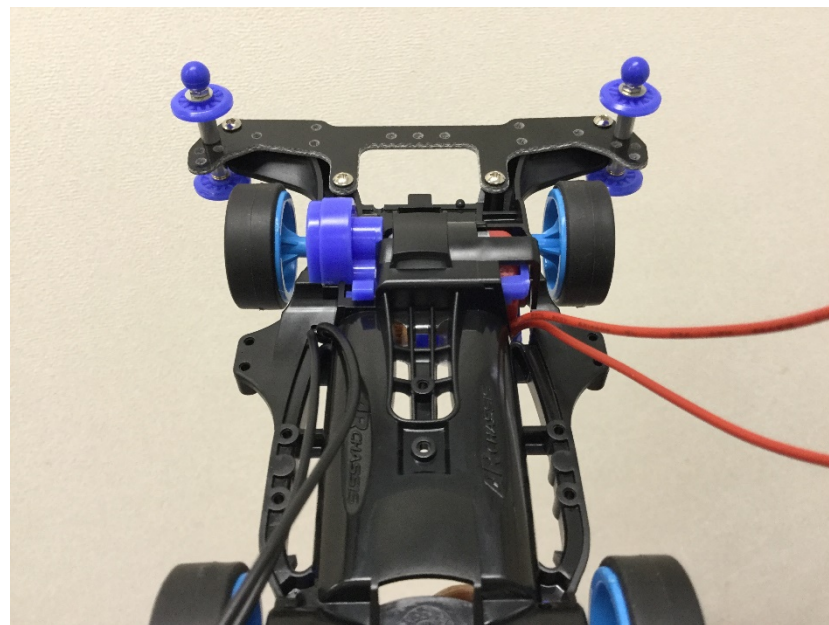
- TAMIYA ミニ四駆 AR chassis 用
 - 基板固定
 - Ritsizm mini4WD2 基板
 - センサ固定
 - 9軸センサ MPU-9250
 - 反射型光センサ RPR-220
 - 3D CAD
 - Autodesk Fusion360 で設計
 - 3D Printer
 - XYZ printing da Vinch nano で出力
 - 位置を合わせてφ2mm穴をあけてM2ネジとナットで固定
- 3D Printer 出力用データ (IGES) [mini4WD2mount.zip](#)



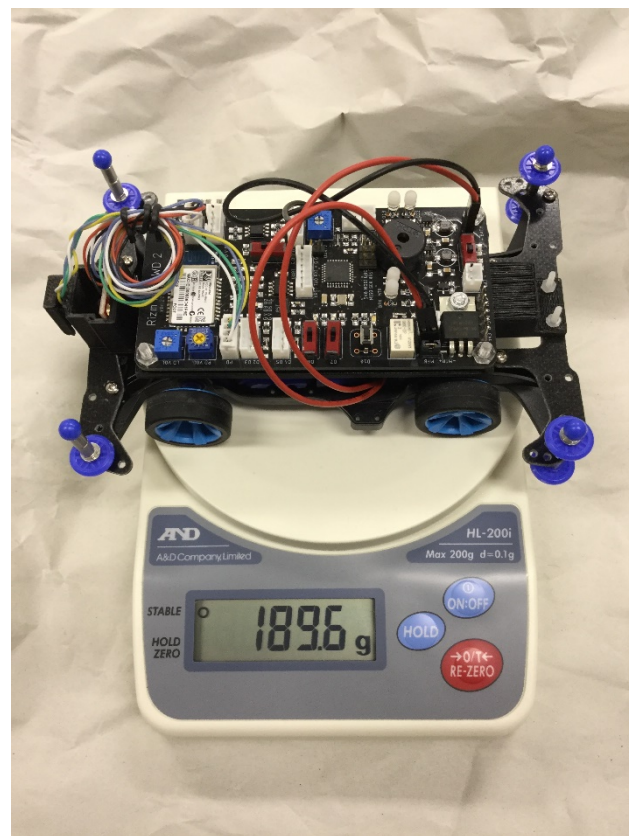
R 電源ライン



- TAMIYA純正電極と銅箔テープに導線を半田付けする
- 極性に注意
- 短絡に注意
- 断線に注意
- ケーブル取り回しは写真参照



- マイコン基板単体 30g
- 車両全体(含 マイコン基板、センサモジュール、単三電池 × 2) 190g



ロジックの消費電流

- 操作 & 記録時の車体側ロジックの消費電流
- Bluetooth 接続、モーターは切断
- 可変電圧電源 Sunhayato DK-920
- 3.0V 供給、電流表示 [?.00A] を読取
- 10～80Hz sampling で 0.12～0.13A 程度
- brake on で +0.05A 程度 (※このときモーターへの電流は停止)

無線通信の転送レート

- 車体送信プログラム → UART → Bluetooth → COM → PC受信プログラム
- 車体送信側: 一定時間の周期でデータパケットの送信を試みる
- PC受信側: パケットを受信し、実際の周期を測定する (cygwin 環境で構築)
- 通信設定は 115,200bps, 8bit, no parity bit, 1 stop bit, no flow control
- パケットサイズ4~32bytesに対する送受信が成功した最速の実周期をまとめる
パケット2048~16384個の平均値。パケットサイズ2bytesは4bytesとほぼ同じ、1bytesではかえって遅くなる。マイコンの送受信バッファの制約により64bytesは不可。
- 概ね仕様性能に近い転送性能を確認できた ($115\text{kbps} \times 8/(8+1) \div 102\text{kbps}$)

パケットサイズ	実測転送周期	実効転送レート
4bytes	400 μ s	80kbps
8bytes	680 μ s	94kbps
16bytes	1360 μ s	94kbps
32bytes	2720 μ s	94kbps

無線通信の遅延

- PC→車体→PCのBluetooth経由ループバックテスト
- 車体送信側: 受け取ったパケットを直ちに返送する
- PC受信側: タイムスタンプを付したパケットを一定間隔で送信し、受信したパケットのループバック遅延を測定する
- 通信設定は 115,200bps, 8bit, no parity bit, 1 stop bit, no flow control
- パケットサイズに対するループバック遅延をまとめる

計1024バイト分の平均。マイコンの送受信バッファの制約により64bytes以上は不可。

パケットサイズ	ループバック遅延	遅延÷2
4bytes	129ms	64ms
8bytes	216ms	108ms
16bytes	263ms	132ms
32bytes	262ms	132ms

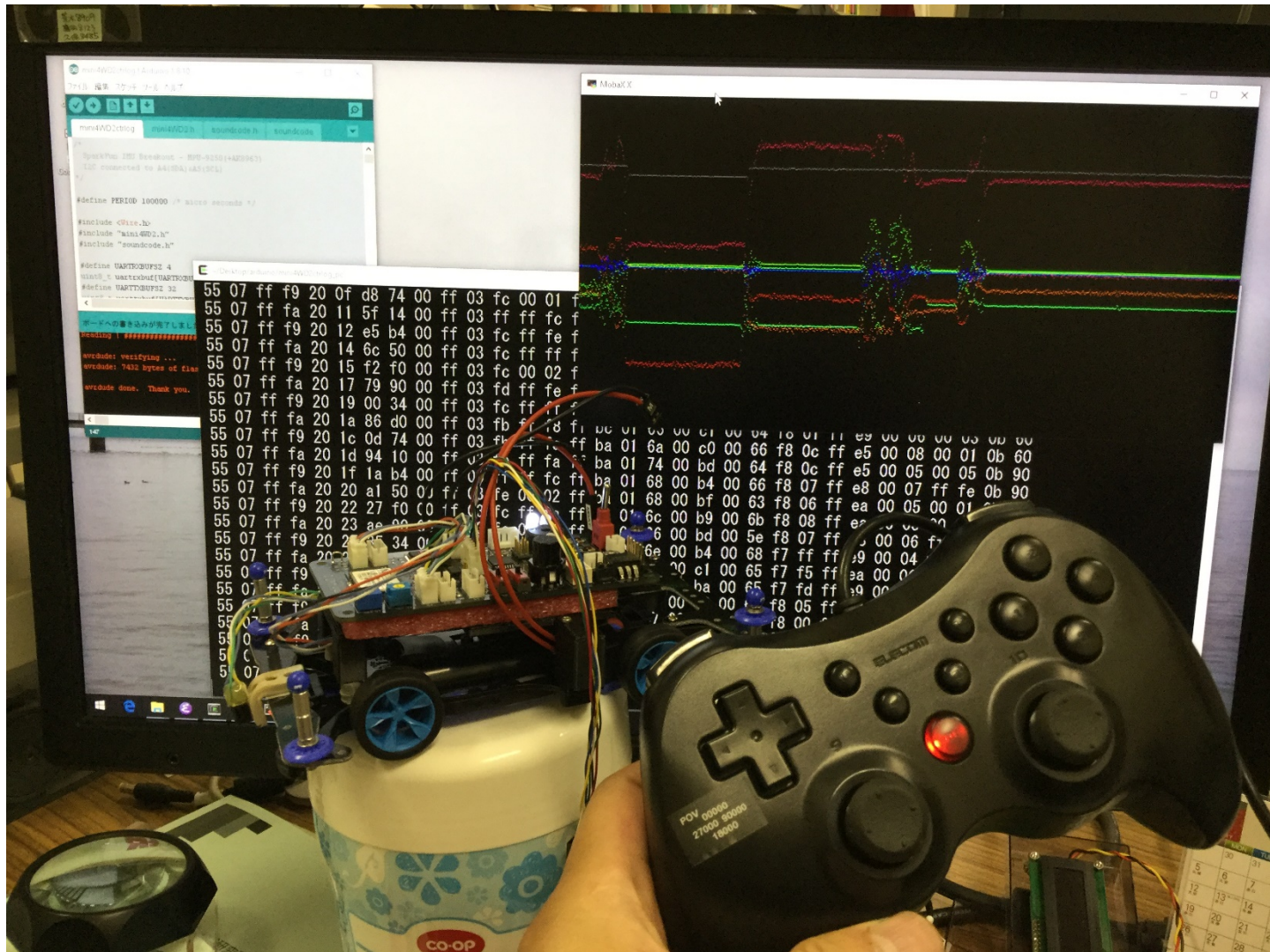
遠隔操作 & 状態記録システム

remote controller and logger

遠隔操作 & 状態記録～教師データ生成

- 特性の把握、機械学習の教師データ生成のため遠隔操作と状態記録が必要
- 車両 \leftrightarrow PC(操作する人)
- UART-Bluetooth で送受信
- Joypad で加速、ブレーキを操作
- 時刻、制御入力、センサー値をPCに転送
- グラフ表示 & ファイルに保存

遠隔操作 & 状態記録の様子



リモコン & ロギングの様子 [mini4WD2_ctrlog2.mov](#) [YouTube](#)

Controler&Logger 実装概要

- 車両側

- Rits mini4WD2 基板 + 反射型光センサ RPR-220 + 9軸センサ MPU-9250
- Arduino IDE 標準ライブラリ (Serial, Wire)
+ 独自Cプログラム

- コントローラ側

- Windows PC + Joypad
- Cygwin 環境 (gcc, Windows API, X11) + X windows server
+ 独自Cプログラム

プログラム mini4WD2ctrlog.zip

操縦 & 記録実験手順

- PCに Joypad を接続し X server を起動しておく
- ミニ四駆をコースに置いて起動する
- Cygwin コマンドラインから...
 - `mini4WD2ctrlog_pc.exe` [COMポート番号]
 - 状態が数値とグラフで表示される
 - ログと走行状態のデータがファイルに記録される(log日時.txt, rec日時.dat)
- 走行状態のデータの表示
 - `mini4WD2ctrlog_print.exe` [データファイル名]
 - これを修正して分析・制御方法の検討を行う

操縦 & 記録 Joypad 操作 ver.2019.12.21

- 12 buttons + 2 sticks + 1POV

注...仕様はプログラム作成者の気分でコロコロ変わります



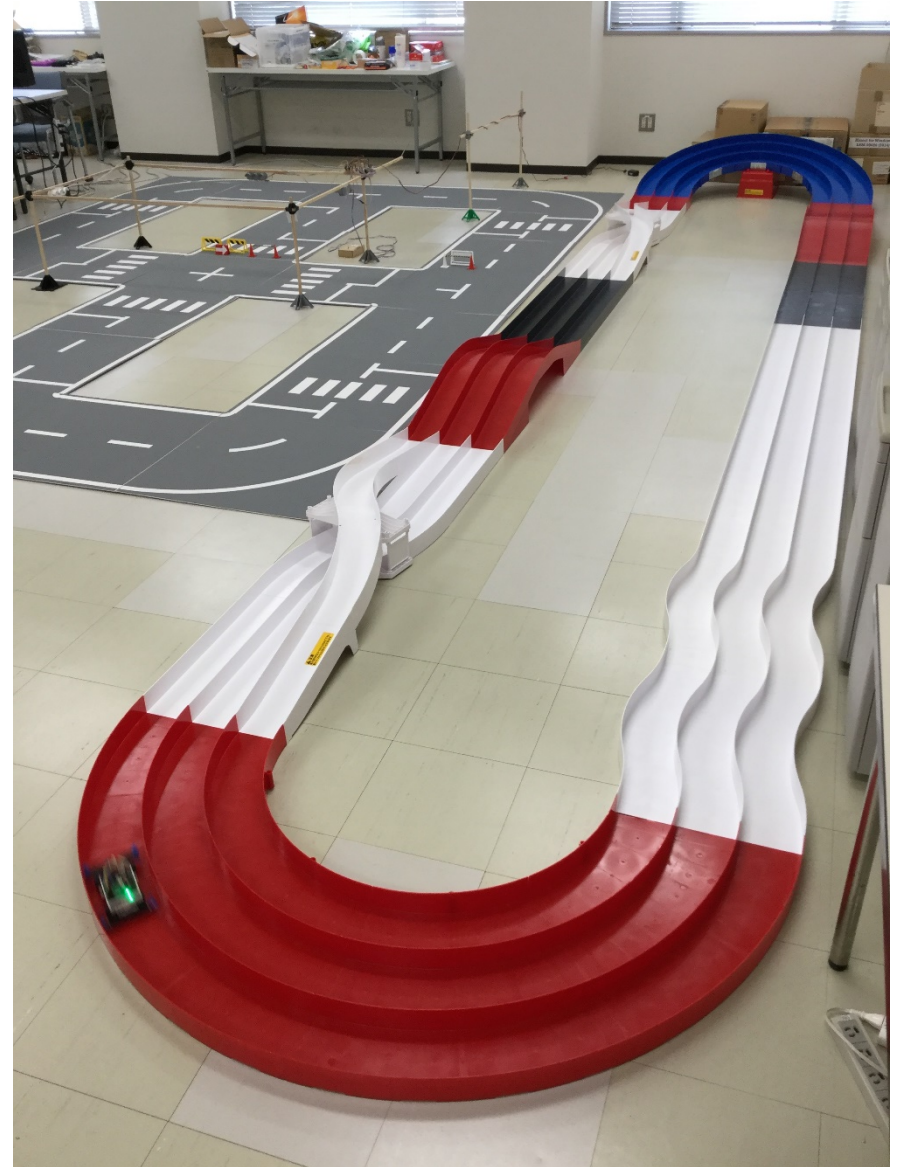
制御パケットの構成

- 4bytes packet : $c0_{[7:0]}$, $c1_{[7:0]}$, $c2_{[7:0]}$, $c3_{[7:0]}$
- 詳細は mini4WD2packet.[c,h,ino]参照
- $c0_{[7:0]}$... 同期ワード 0x55
- $c1_{[0]}$... ブレーキ 0=off, 1=on
- $c1_{[1]}$... ブレーキ 0=off, 1=on
- $c1_{[2]}$... ブレーキ 0=off, 1=on
- $c1_{[3]}$... ブレーキ 0=off, 1=on
- $c1_{[7:4]}$... サンプリング周波数 / 拡張データ
- $c2_{[7:0]}$... アクセル 0=max, 255=off / 拡張データ
- $c3_{[7:0]}$... 拡張データ

基礎評価

R 遠隔操縦走行テストと概算速度

- 右写真の長円コース
 - コースタイル内訳: 直線8、山4、波状路2、交差6、カーブ4、バンクカーブ4
 - 直線1タイル54cm
- 遠隔操縦で走らせて3周15～17秒程度
- 1タイルあたり0.2秒弱
- 右の長円コース×3周≒40m強
- 2.5m/s (9km/h) 程度



加減速性能の基礎評価

- 加速、減速の性能評価
- アクセル(PWM)値を一定に加速
- 指定した時刻、場所での減速
- 要求仕様
 - 高精度な計時...車体側マイコンのクロック
 - アクセル一定加速...PWM値指定 & 固定
 - 高精度な位置推定...路面色 + 光センサ
 - 低遅延のブレーキング...車体側マイコンで自動制御

加減速実験プログラム概要

- 車両側
 - Rits mini4WD2 基板 + 反射型光センサ RPR-220
 - Arduino IDE 標準ライブラリ (Serial, Wire)
+ 独自Cプログラム
- コントローラ側
 - Windows PC + Joypad
 - Cygwin 環境 (gcc, Windows API, X11) + X windows server
+ 独自Cプログラム

プログラム [mini4WD2brktst.zip](#)

操作 & 記録実験手順

- PCに Joypad を接続し X server を起動しておく
- ミニ四駆をコースに置いて起動する
- Cygwin コマンドラインから...
 - `mini4WD2brktst_pc.exe` [COMポート番号] [固定モーター出力] [条件ブレーキ比較方法: more/less] [条件ブレーキ閾値] [ブレーキ方法(ショートブレーキ): on/off]
 - 状態が数値とグラフで表示される
 - ログと走行状態のデータがファイルに記録される(log日時.txt, rec日時.dat)
- 走行状態のデータの表示
 - `mini4WD2brktst_print.exe` [データファイル名]
 - これを修正して分析・制御方法の検討を行う

加減速実験 Joypad 操作 ver.2019.12.22

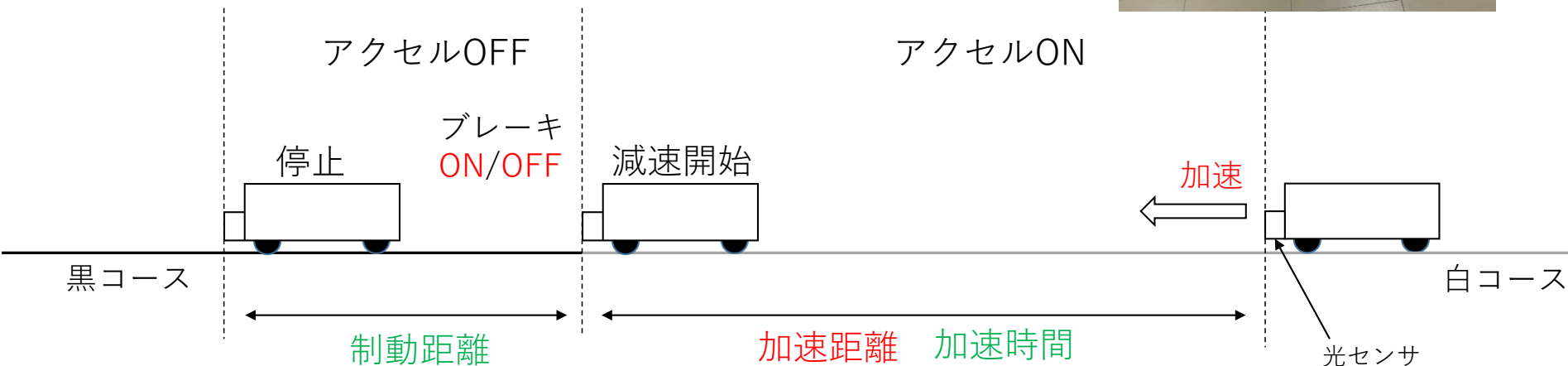
- 12 buttons + 2 sticks + 1POV

注...仕様はプログラム作成者の気分でコロコロ変わります

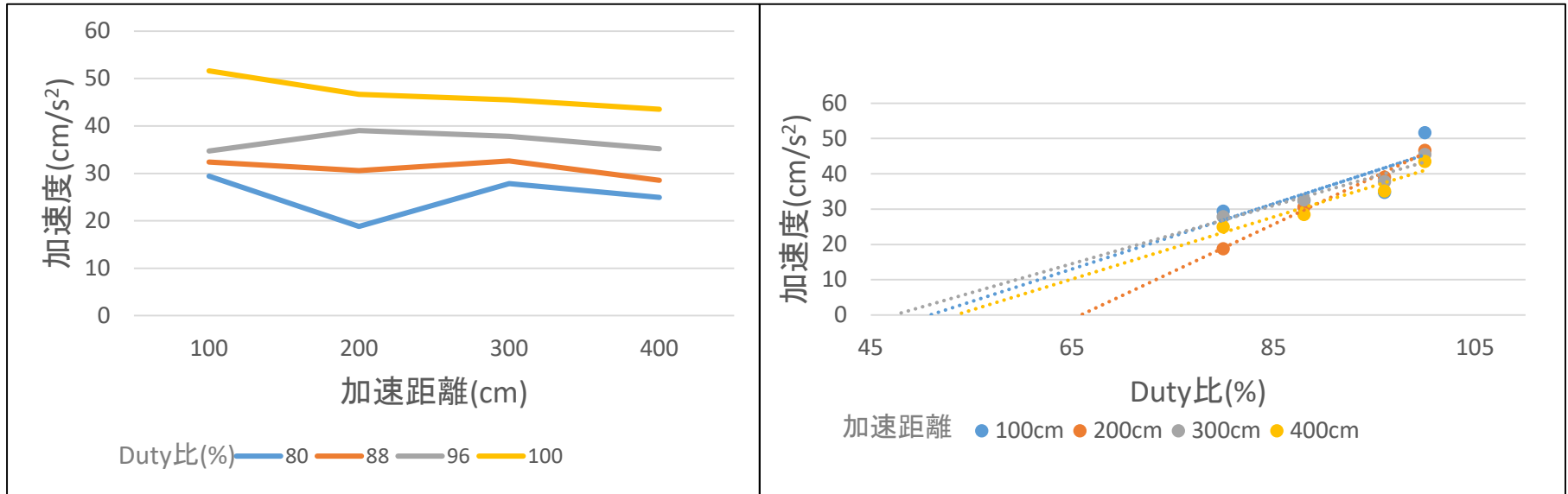


加減速性能評価実験

- 直線コース、白で加速、黒判定で減速
- PWMのDuty比：約80～100%
- ショートブレーキ：ON/OFF
- 加速距離：100～400cm
- 加速時間、制動距離を計測
計32通りを各10回実験

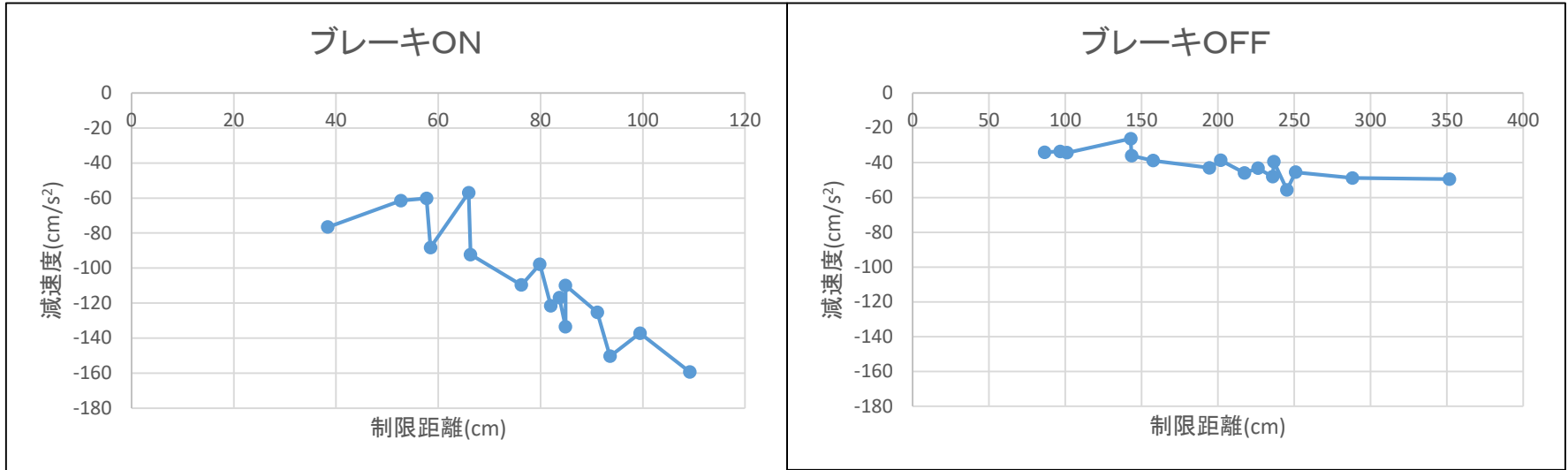


加速距離・Duty比と加速度



- 加速度一定と仮定し、加速距離・加速時間から $a=2l/t^2$ より算出
- Duty比100% 加速距離100cm で 約52(cm/s²)
- Duty比を増加させると加速度が増加
- 加速距離が増加するに従ってやや減少傾向

制動距離と減速度



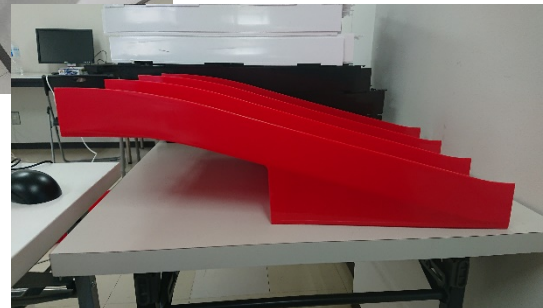
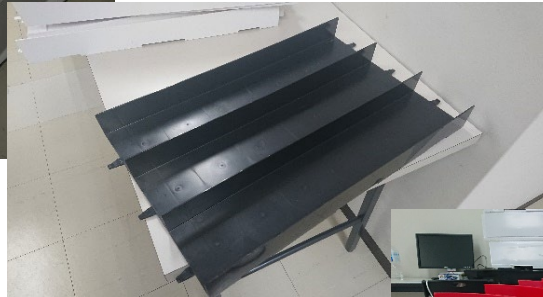
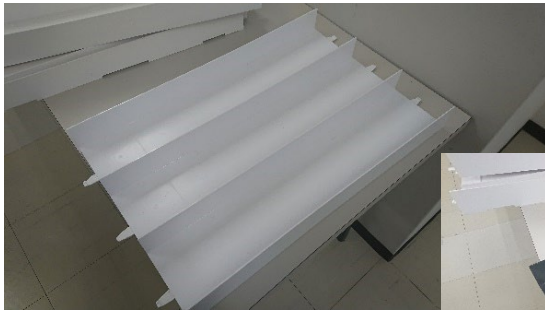
- 減速度一定と仮定し、加速時の平均速度と制動距離から $a=2v^2/l$ より減速度を算出
- ショートブレーキによって 3 倍以上の減速性能

コースタイル推定の基礎検討

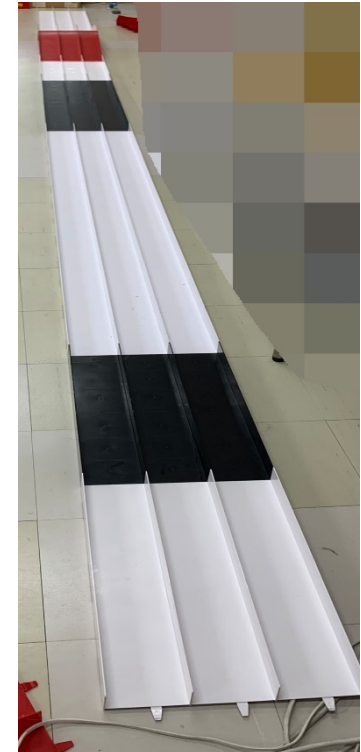
- 搭載したセンサ値から、現在走行中のコースタイル種の推定の可能性を検討する

9 軸慣性センサ → 車体の傾き → タイルの傾斜

フォトリフレクタ → 路面の明るさ → タイルの色



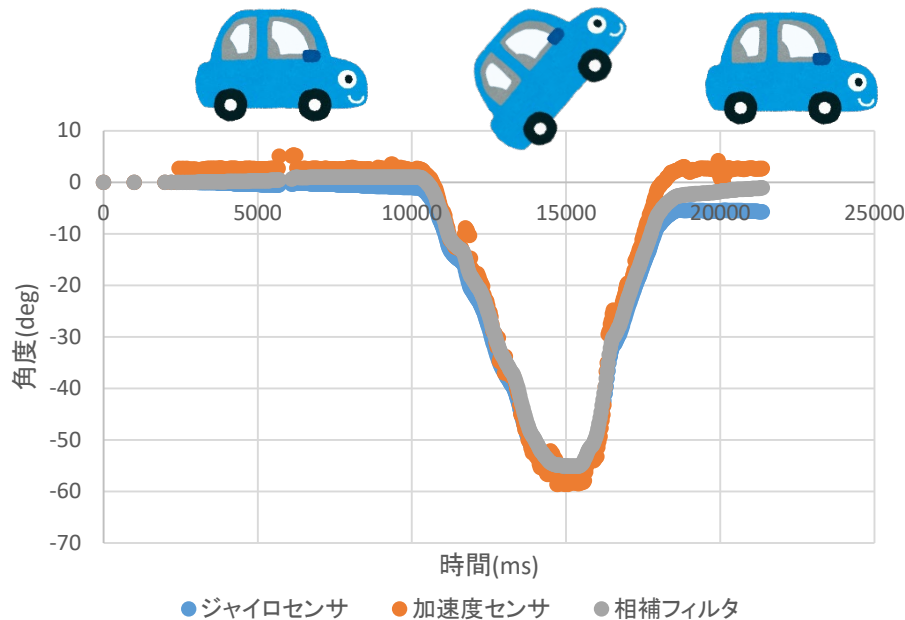
タミヤ公式ミニ四駆コース
ジャパンカップジュニアサーキット
ストレート(白・黒) & スロープセクション(赤)



車体の傾き推定

- 9 軸慣性センサの加速度センサとジャイロセンサの値から相補フィルタを用いて角度を推定する

例：車両を水平→上向き→水平と動かしたときのピッチ角の推定



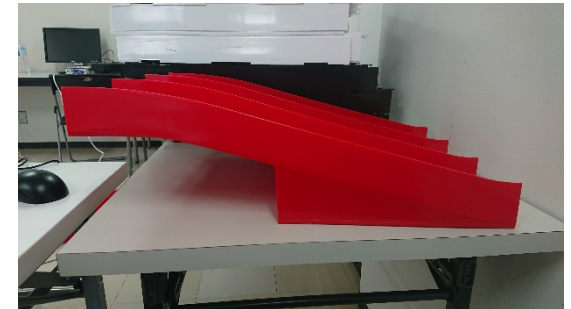
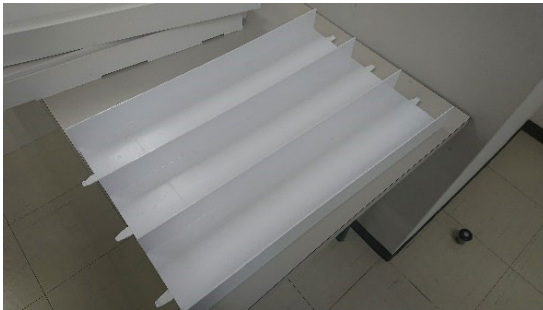
$$\frac{a_x}{\sqrt{a_y^2 + a_z^2}} = \frac{-\sin\theta \times g}{\sqrt{(\sin^2\varphi + \cos^2\varphi)\cos^2\theta \times g^2}} = -\tan\theta$$

$$\theta = \arctan\left(\frac{-a_x}{\sqrt{a_y^2 + a_z^2}}\right)$$

$$\theta_n = (1 - k) \times \theta_a + k \times (\theta_{n-1} + x \times dt)$$

コーススタイルの色推定

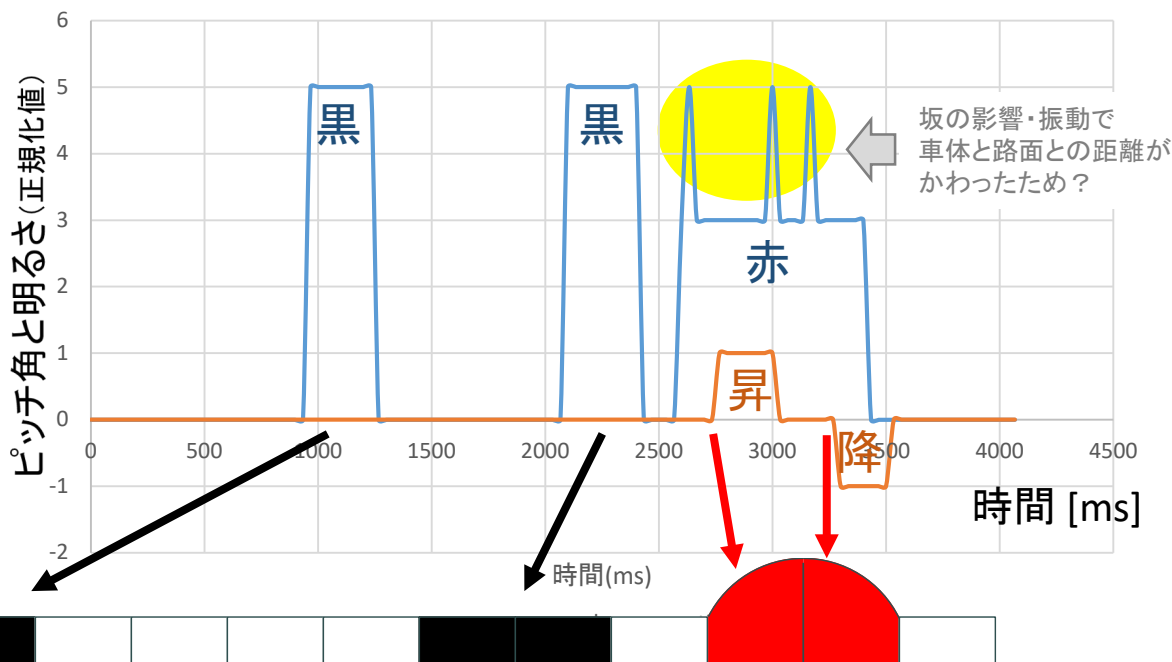
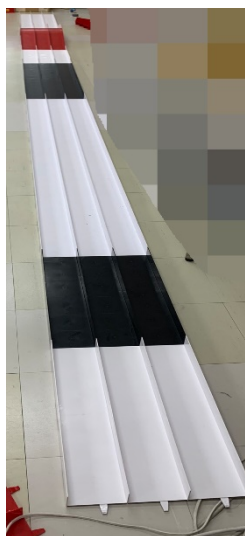
- コーススタイルに車両を置いてフォトリフレクタの値を読み取った
- 10bit ADC の値、0が最も明るく、1023が最も暗い
- 白タイルと黒タイルの差が大きくなるように、LED光量調整VOL、PD感度調整VOLを調整した



コーススタイル	フォトリフレクタの値
ストレート白	50～150程度
ストレート黒	700以上
スロープ赤	500～700程度

コーススタイル推定の試行

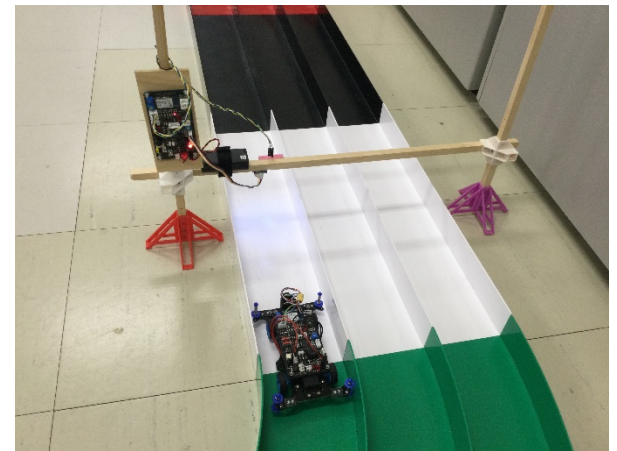
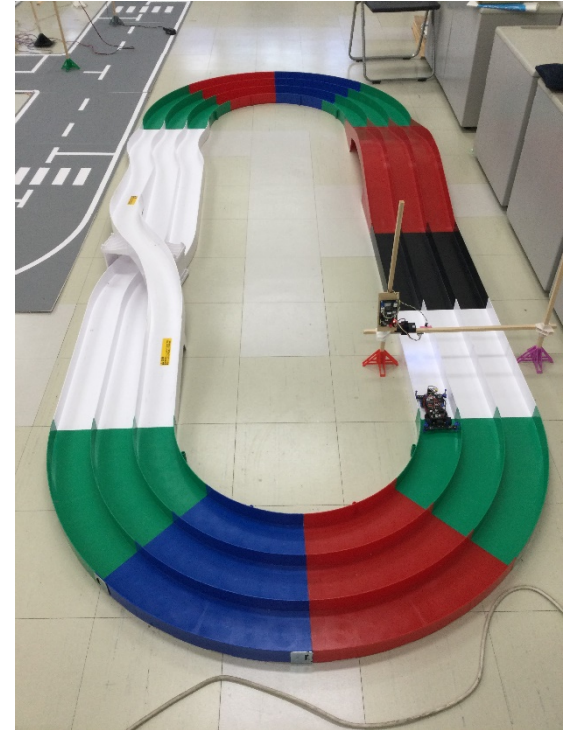
- 推定ピッチ角とフォトリフレクタ値からコーススタイルの推定を試みる
- 図のコースを走行
- 適切な閾値でコーススタイル種を概ね推定できる見込み有り



ラップタイム計測環境

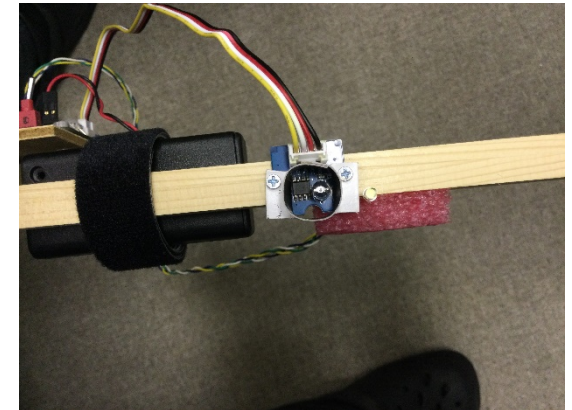
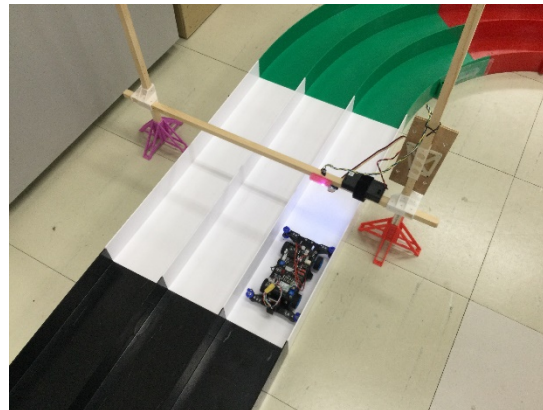
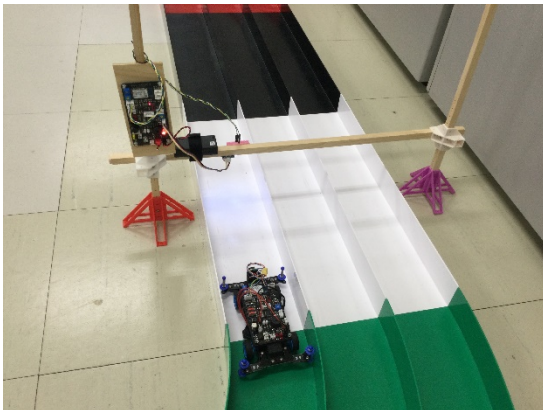
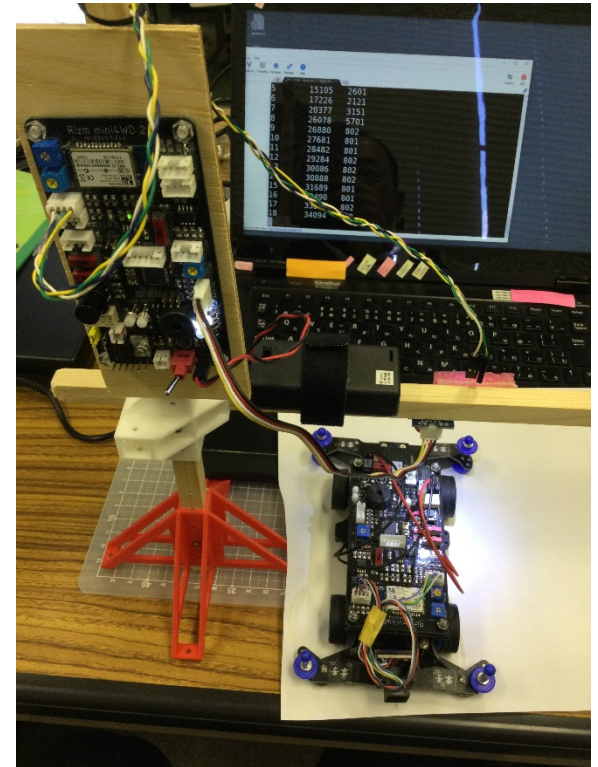
ラップタイム計測

- 評価にはラップタイム計測が必要
- 全レーン走行の3周で1セット
- 0.1～0.01秒程度の精度が必要
- 走行制御(センサ)に影響を与えてはならない
- 設置・配線、データ収集の利便性が求められる
- 手間とコストは抑えたい



計測環境の構築

- ミニ四駆用マイコン基板を流用
 - マイコン、汎用I/O、無線通信、高い拡張性
- 明るさセンサで車体を検知
- Bluetooth-Serial でラップタイムをPCに送信
- 木材と3Dプリンタでゲートを作成



mini4WD2 ラップタイマの仕様

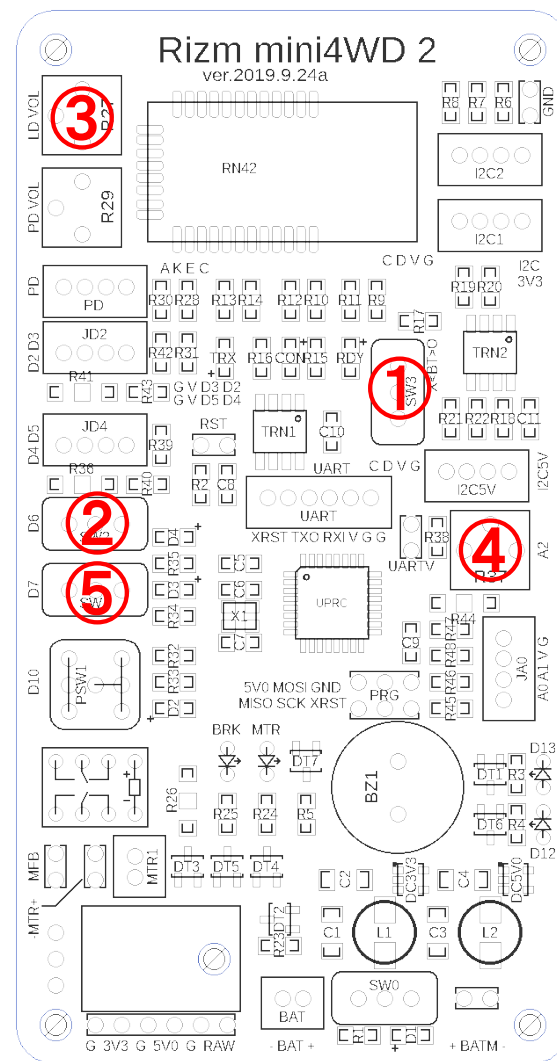
- メインボード
 - mini4WD2
- 明るさセンサ
 - Grove Light Sensor (P) v1.1 (アンプ付、アナログ入力)
 - Grove互換コネクタ(A0)に接続
- 補助照明
 - 高輝度白色LED
 - フォトリフレクタ用LEDコネクタに接続
 - 明るさ調整可
- バッテリ
 - Ni-MH 単四 × 2
- 動作の様子 [laptime.mov](#)
- ソースコード [mini4WD2laptime.zip](#)
- 明るさセンサ (A0)
 - 車体検出用
- 半固定抵抗 (A2)
 - 車体判定閾値の設定
- スライドスイッチ (D7)
 - 閾値以上／以下の切替
- スライドスイッチ (D6)
 - 計測／状態表示の切替
- プッシュボタン (D10)
 - ラップリセット、ヘルプ表示
- LED (D13/D12)
 - 閾値以下/以上で点灯
- Serial
 - ラップタイム／状態を送信

計測準備(PC側)

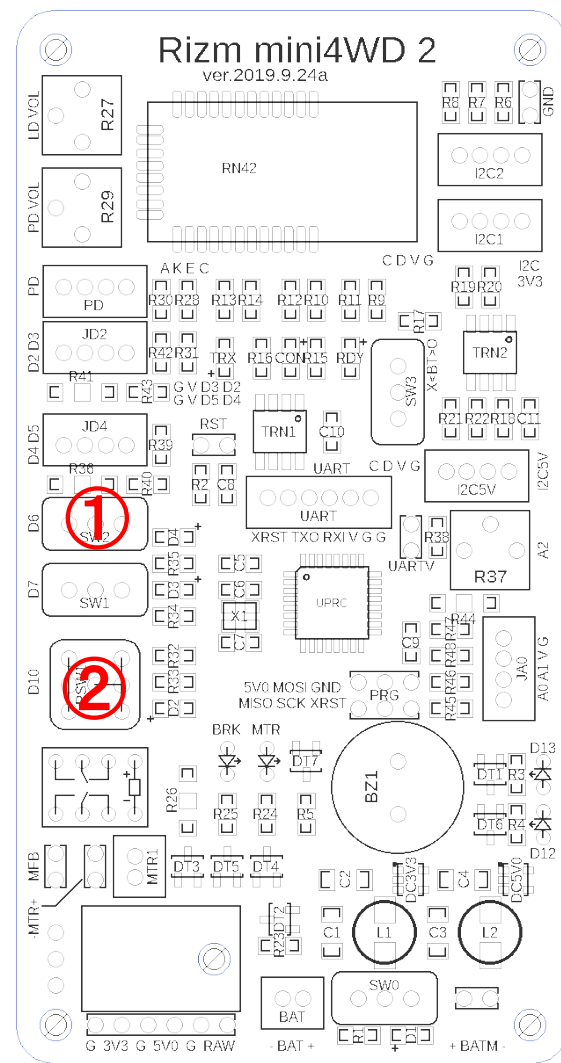
1. Bluetooth デバイスをペアリングする
 - デバイス名 RNBT-[MAC下四桁]
2. 適当な Terminal Emulator で接続する
 - Teraterm, MobaXterm, etc.
 - 115200bps, 8bit, 1stopbit, no-parity, no-flowctrl
3. 必要に応じて受信内容をファイルに保存する

計測準備(計測器側)

1. スイッチ X<BT>O でマイコンと Bluetooth を接続する
2. スイッチ D6 で状態表示モードにする
3. 車体の有無による明るさセンサ値 (A0) の差をできるだけ大きくする
 - 部屋の照明を調整
 - 半固定抵抗 LD VOL をまわして補助照明を調整
 - 走行制御に影響を与えない範囲で
4. 半固定抵抗 A2 をまわして判定閾値を車体の有無の中間の値に設定
5. スイッチ D7 で車体があるときの条件 (閾値以上／以下)を設定



1. スイッチ D6 で計測モードにする
2. ボタン D10 でラップをリセットする
3. 走らせて計測開始
 - 表示時間の単位は ms



まとめ

- AI制御の教育研究のためのミニ四駆キット
- 消費エネルギー制限の下での高速AI制御
- 独自基板の開発
- モーター制御等基本動作の検証、済 (ver.1)
- 通信機能、センサ拡張可能なマイコン基板を開発 (ver.2)
- 車体に搭載して、走行実験、データ収集、評価中

今後の課題

- 軽量AIアルゴリズムの実装と評価
- FPGA搭載基板の開発