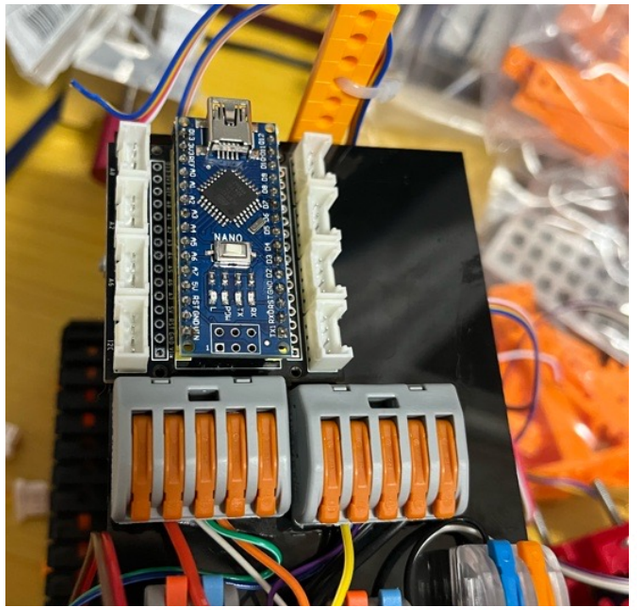
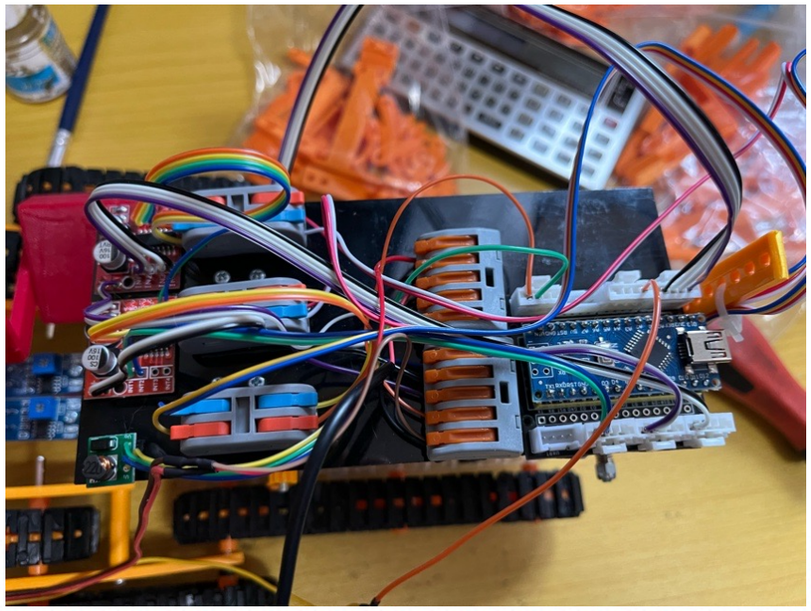
レスキューロボットをマイコン化しよう

会津大学復興知授業





タケルソフトウェア　2022

レスキューロボットをマイコン化しよう

今回の授業では、市販されている模型を改造してマイコンで制御できるようにしてみるよ。

市販品ならば、いろいろな模型（水・陸・空）を選ぶことができる。

それにマイコンを搭載して制御できると「確実に動くモデルを使って、効率よくロボットを作成することができる」んだ。

ちなみに使うモデルはタミヤ模型から出ている「楽しい工作工作シリーズ **No.169** レスキュークローラー工作セット **3ch**リモコン」になる。

このモデルは岐阜高専とタミヤがレスキューロボットコンテストなどの教育目的で市販化されたモデルで、小学４年生以上の子供たちでも保護者同伴で組み立てができ、ロボット教育にも用いられている。

パッケージには組み立て対象年齢は書かれていないけど、小学６年生くらいなら１人でも組み立てができると思う。

しかし、プラモデルなどの組み立て経験が少ない人にとっては、組み立て作業がちょっと難しく感じるかもしれない。

そこで、組み立ての際に気をつけて行うと良いポイントをまとめてみたよ。

1）部品をランナーから外すときに外しにくい場合は、ランナーごとニッパーで切り取ってから部品を切り離す。

2）組み立てを行う前に手順書に書かれている手順を一通り読むこと。

3）組み立てを行う前に、部品表に書かれている部品が実際にどこにあるのか確認しておくこと。
（部品表にAとかBとかアルファベットが書かれているものは部品のランナー部分にそのアルファベットが書かれているので確認すること）

4）ギヤ関係は部品に14Tとか22Tなど歯数が刻印されているので、きちんと確認すること。
（ギヤの組み立てで苦労する人が多いです）

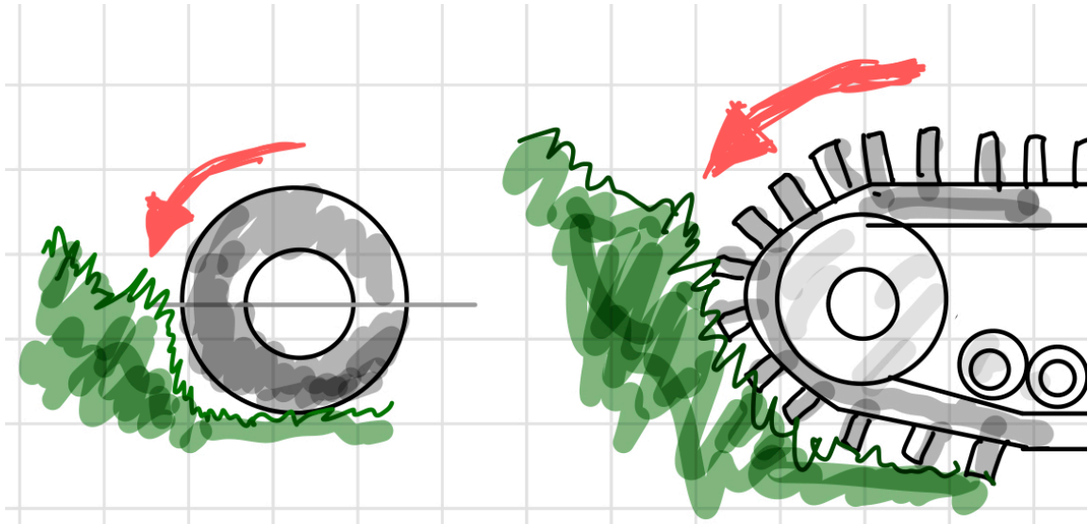
5）部品を組み立てる際には、手順書と同じ向きに手に持った部品を合わせると、位置関係を把握しやすい。

6）あまり丁寧に作業しすぎると時間が足りなくなるので、神経質に作業しすぎないこと。

では、さっそく組み立ててみよう！

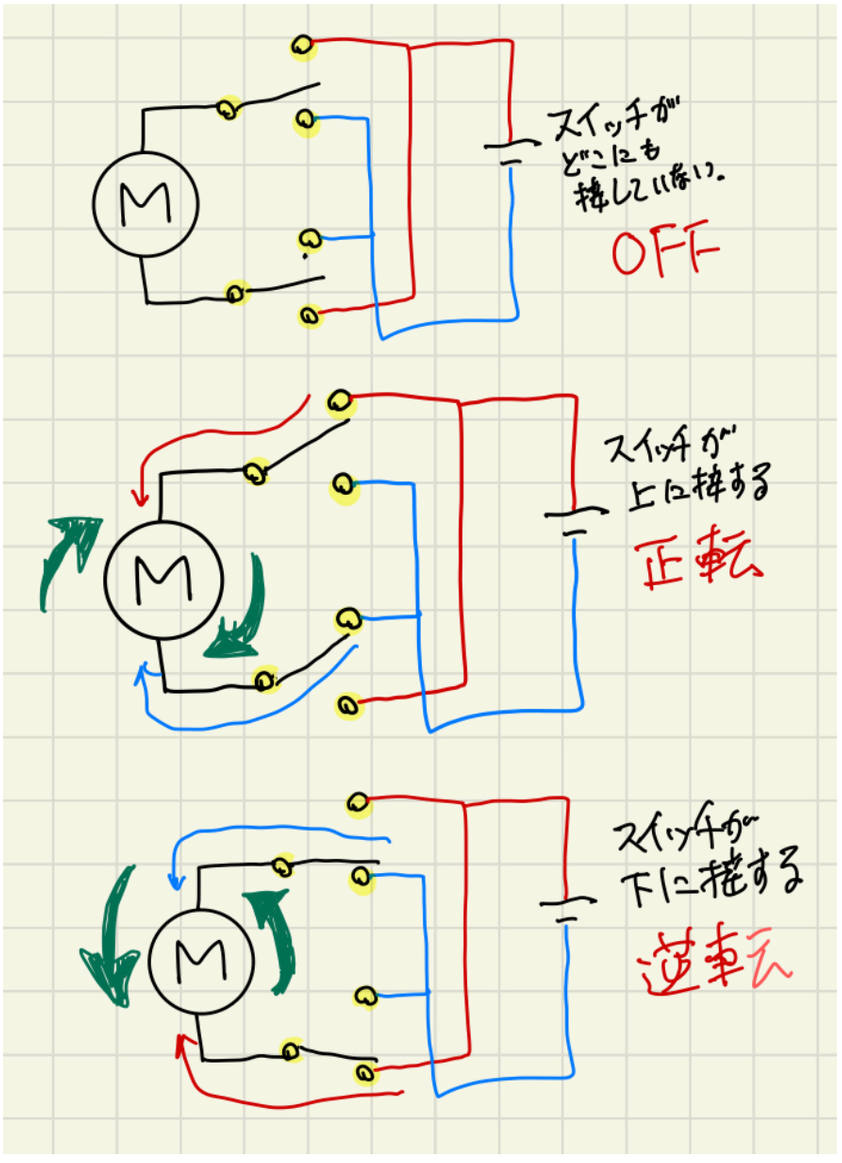
　出来上がったレスキューロボットを操作してみよう。
身の回りのものを使って段差の乗り越えなどにトライしてみよう。

　クローラロボットは瓦礫の上などの不整地を走行するのに適しているけれど、その反面、自動車のように高速に移動するのには適していない。



タイヤの場合は、半径を超える高さのものを乗り越えることができないけれど、クローラーの場合は、ベルト部分の突起が障害物に乗り上げることで走行することもできる。
しかしそれには限界があるので、どのくらいの高さまで乗り越えることができるか試してみると面白い。
　レスキュークローラロボットの場合、フリッパー部分のクローラを持ち上げたり下ろしたりできるので、より高さがある障害物も乗り越えることができる。
そのようなことから、災害対策用のロボットに採用されやすい走行方法なんだ。

　このレスキューロボットは、コントローラのレバー操作でモーターに流す電気の極性（＋と−）を切り替えて操作している。



マイコンで同じことをすれば、このロボットを制御できるようになるね。
次のページでそのアプローチを考えてみよう。

マイコン制御のためのアプローチ

１）現在の制御仕様を確認し、制御部品を選定する。

・電源は電池2本（1.5Vx2=3V）　→ リポバッテリー3.7V

・制御するモータは３つ（正転・逆転）→モータドライバ

・動作時の電流　→　？？？？

２）動作時の電流を調べてみよう。

動作時の電流は、電源装置を使うと簡単に測定できる。
また、組み立て途中の実験の時もショートなどして回路を壊してしまうような事態になるまえに電流を止める機能もあるので安全。

実際に電流を測るとモータ１つを動かした時に0.3Aくらいの電流が流れた。

そこで、３つのモータが動くとすると最大で0.9Aくらいの電流が流れることになる。

しかし、ぎりぎりの部品選定をしてしまうとちょっとの負荷の増大で簡単に止まってしまうので余裕を持って部品を選定する。

今回は１つのモーターにつき最大1.5A流せる、モータードライバを選定してみた。

３）改めてマイコン制御のための部品を選定してみる。

１）電源：リポバッテリー1個

２）モータードライバ：２個（今回は１つのモータードライバで２つのモータをつなげられるものを選定したので３つのモータは２個のモータドライバで制御できる）

３）マイコン：Arduino Nano　(5V)

４）配線用のGROVEシステム：1個

５）DC-DC変換基板：リポバッテリーの電圧を5Vに変換する。

６）端子台：3.7VとGNDの配線をやりやすくする。

７）バッテリーコネクタ：配線を簡単にするため。

８）モータコネクタ：すぐに元のリモコンにも戻せるようにするため。

使われる部品はこんな感じになったよ。

1）リポバッテリー

小さくて軽くて大電流が流せる。そのためドローンの電源によく利用されるが、ショートすると発火したりして危険。

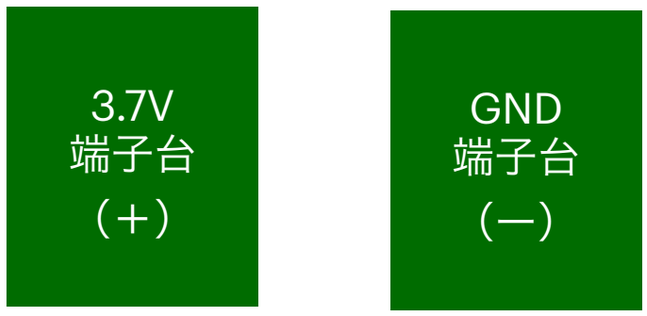
2）コネクタ

リポバッテリーを簡単に繋ぐことができるコネクタ。いろんな種類があるよ。

3）端子台

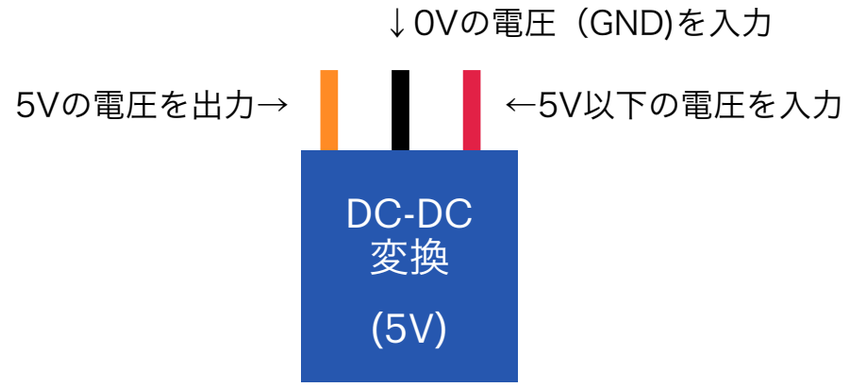
同じ信号や電源を分けたりするときに便利な部品。これを使わない場合、同じ用途の電線をハンダ付したりしないといけなくて不便。

端子台の場合、好きに抜き差しできるので変更も簡単。



4）**DC-DC**変換器

5V以下の電圧を昇圧して**5V**にしてくれる部品。最大で**2A**も流せる。小さいけどすごい部品。

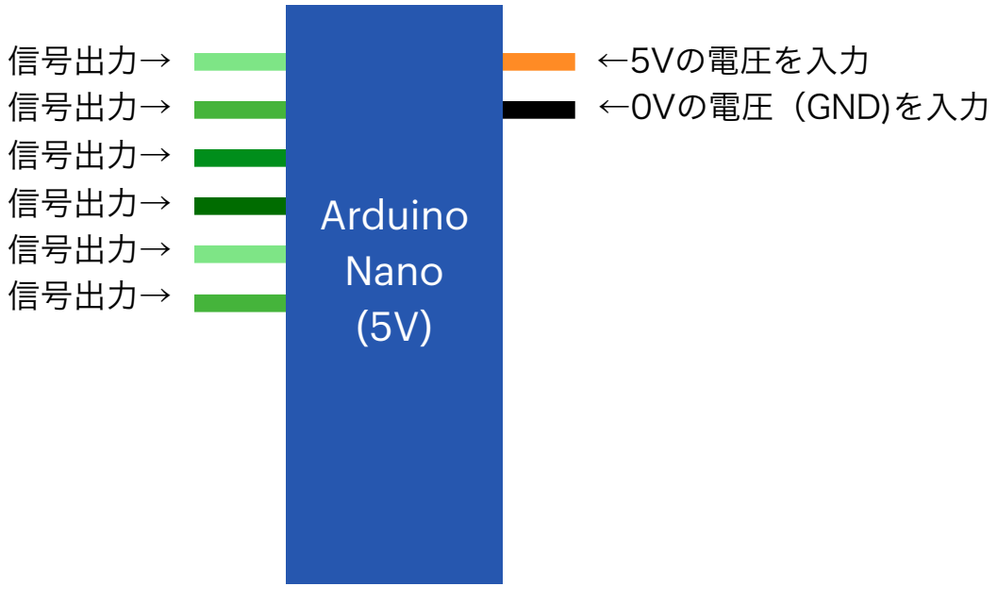


5）**Arduino Nano**

一部の漫画で２１世紀の大発明といわれるマイコン。

標準サイズの**UNO**、小さくなった**Nano**など大きさの種類も多い。

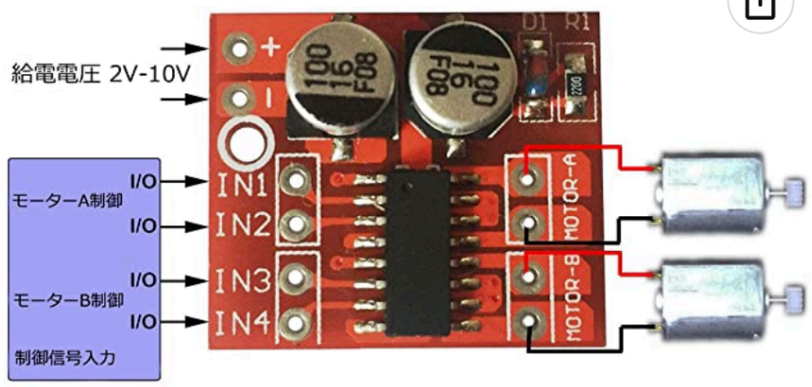
とても安く使いやすいマイコン。



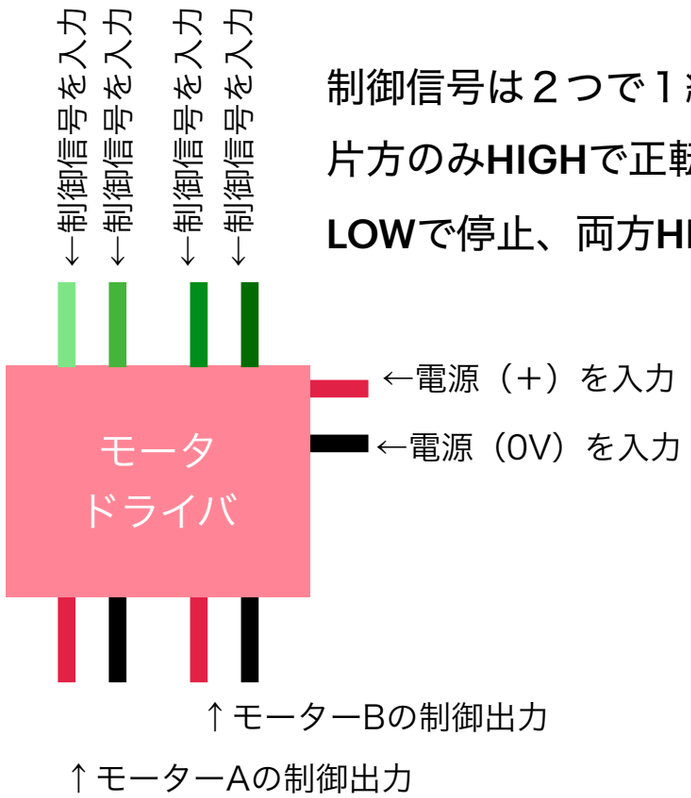
6）モータードライバ

電源線（２～１０Vまで）と制御線(IN1,2,3,4)を繋いであげると、モーターを２個まで制御（正転・逆転・停止・ブレーキ）できる部品。

信号入力は1.8~7Vまで対応しているので、5VのArduinoと直結できる。



単独ドライブ２チャンネル直流モーター;
INx 制御信号入力端，信号電圧範囲1.8V-7V
IN1,IN2制御モーターA
IN3,IN4制御モーターB



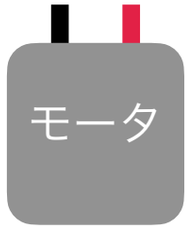
制御信号は２つで１組の制御になっており、片方のみHIGHで正転もしくは逆転、両方LOWで停止、両方HIGHでブレーキになる。

7）モータ

模型用モータ。使用できる電圧の範囲は3~1.5Vの範囲。通常使用は1.5Vを推奨。

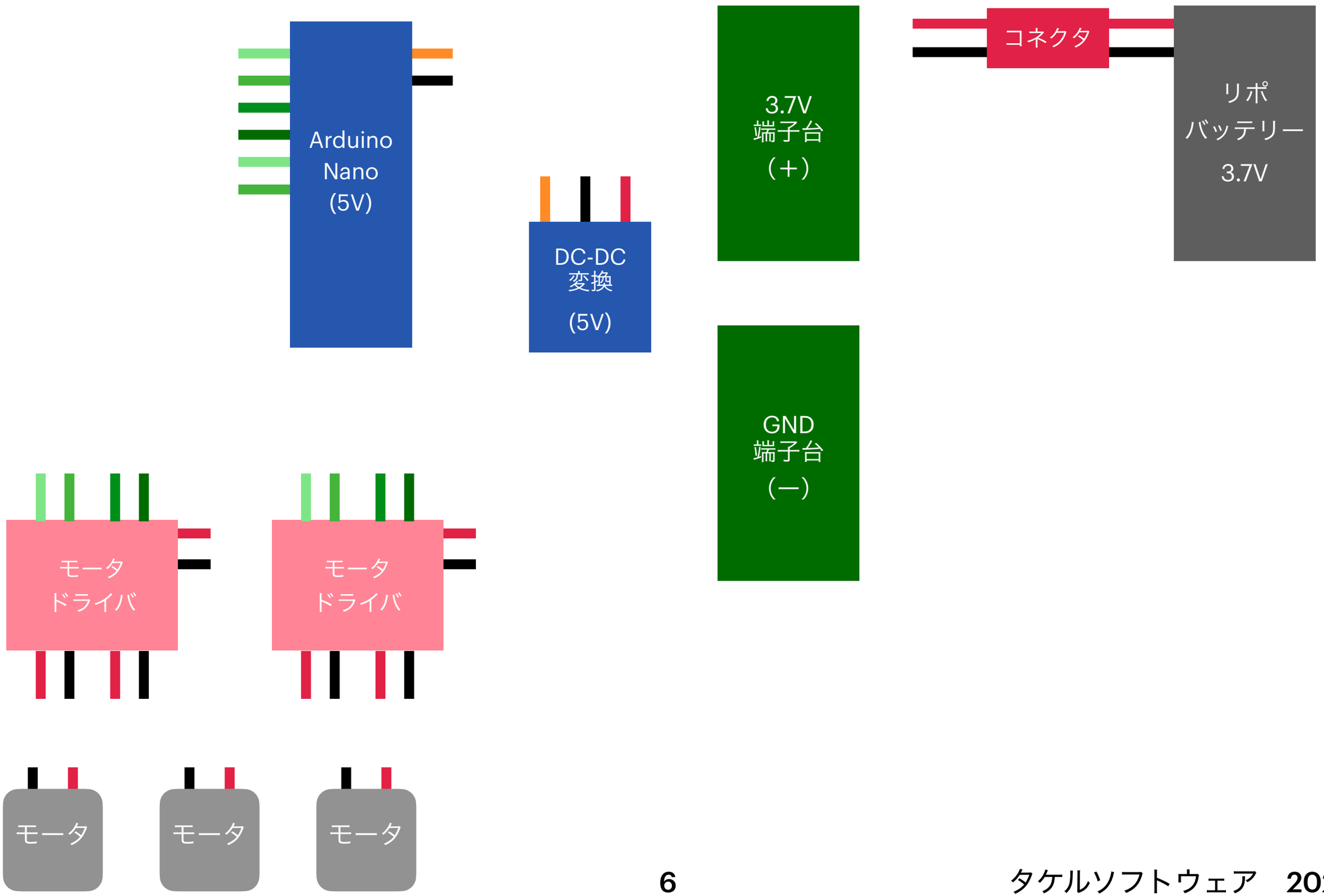
今回は、リポバッテリーが3.7Vなので電圧割り増しで動作させている。

短時間の動作ならば問題はないと思うが確実に寿命は縮んでいる。



レスキューロボットをマイコン化しよう

【部品がどのように繋がるか考えてみよう】



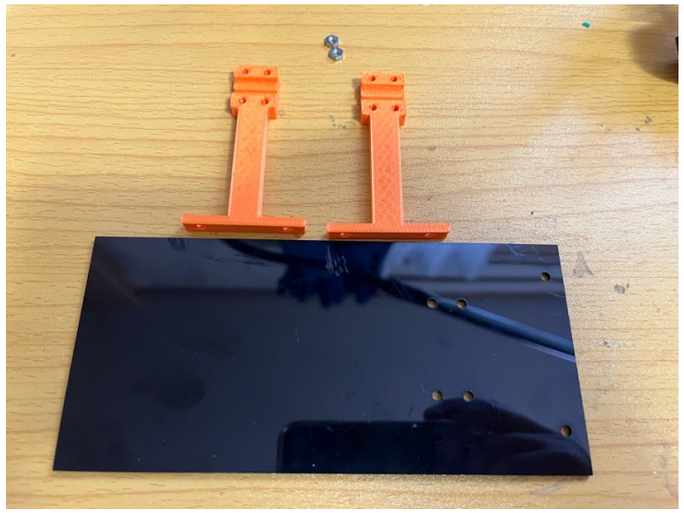
【レスキューロボットの改造の前に】

　ネジやナット類は各自必要な個数を自分で集め、持っていくこと。
　必要なネジ・ナットは以下の通り。

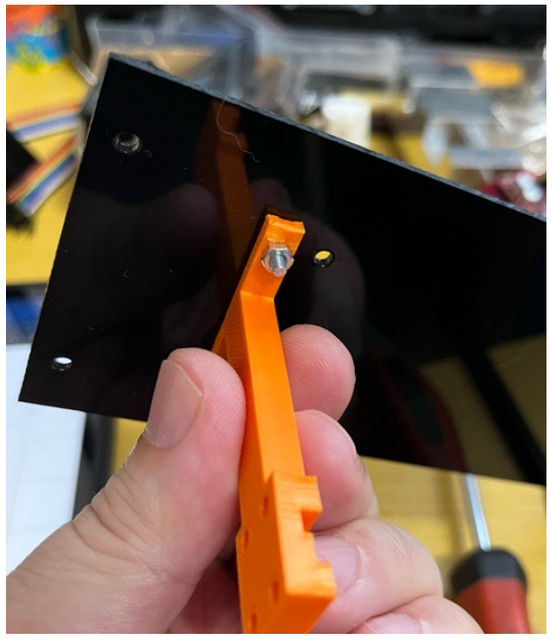
名称	長さ(mm)	個数
M2ナベ子ネジ	6	2
M2.5ナベ子ネジ	8	2
	10	6
	15	2
	20	4
M3ナベ子ネジ	16	2
2.5タッピングビス	10	5
皿ビス2	5	5
M2ナット	-	2
M2.5ナット	-	14
M3ナット	-	2

さて、ここからレスキューロボットを改造してみよう。

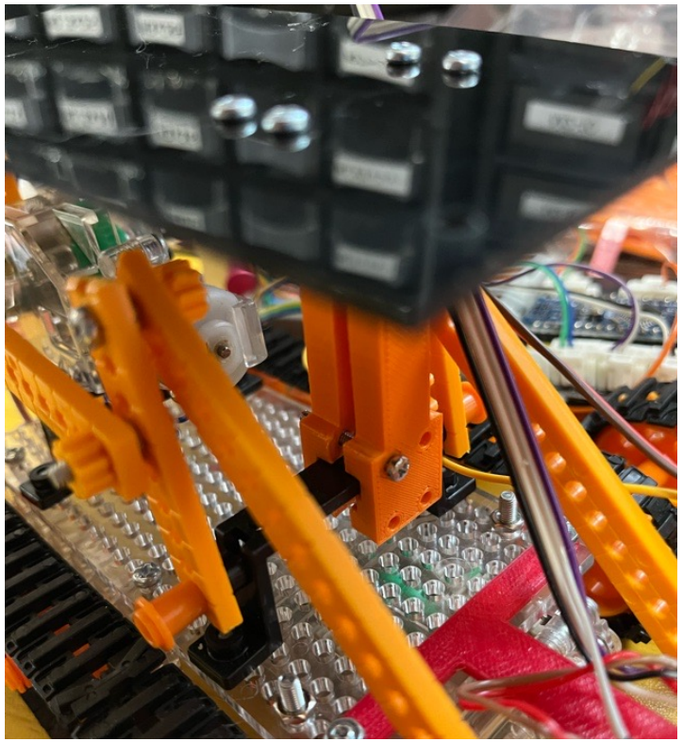
1）制御ベースプレートとベースステー２個、**M2.5x10 4**本、**M2.5x20 4**本、**M2.5**ナット **8**個を準備する。

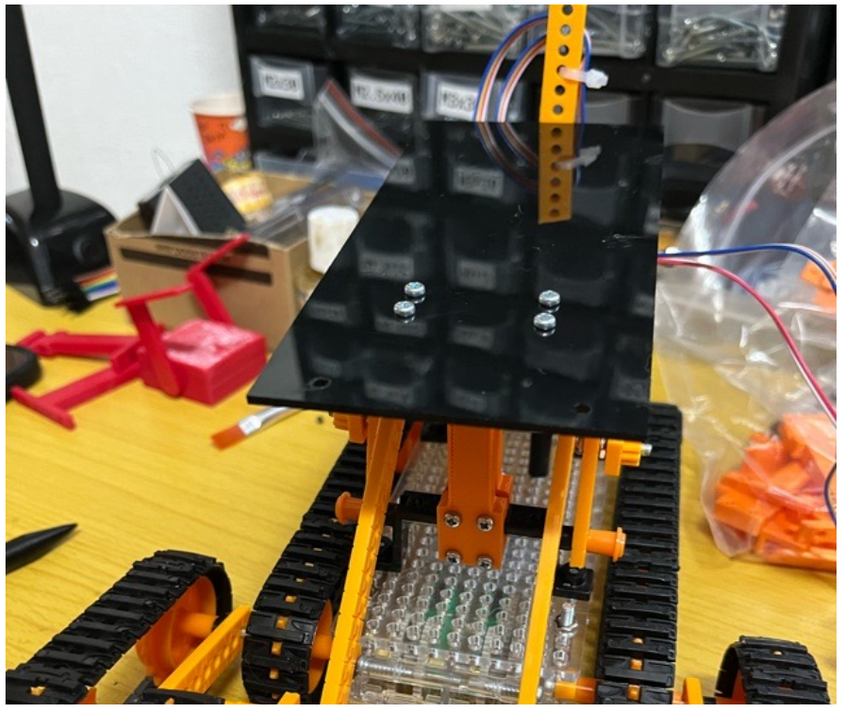


2）写真の様に**M2.5x10**ナベ子ネジ４本と**M2.5**ナットでベースステーを取り付ける。

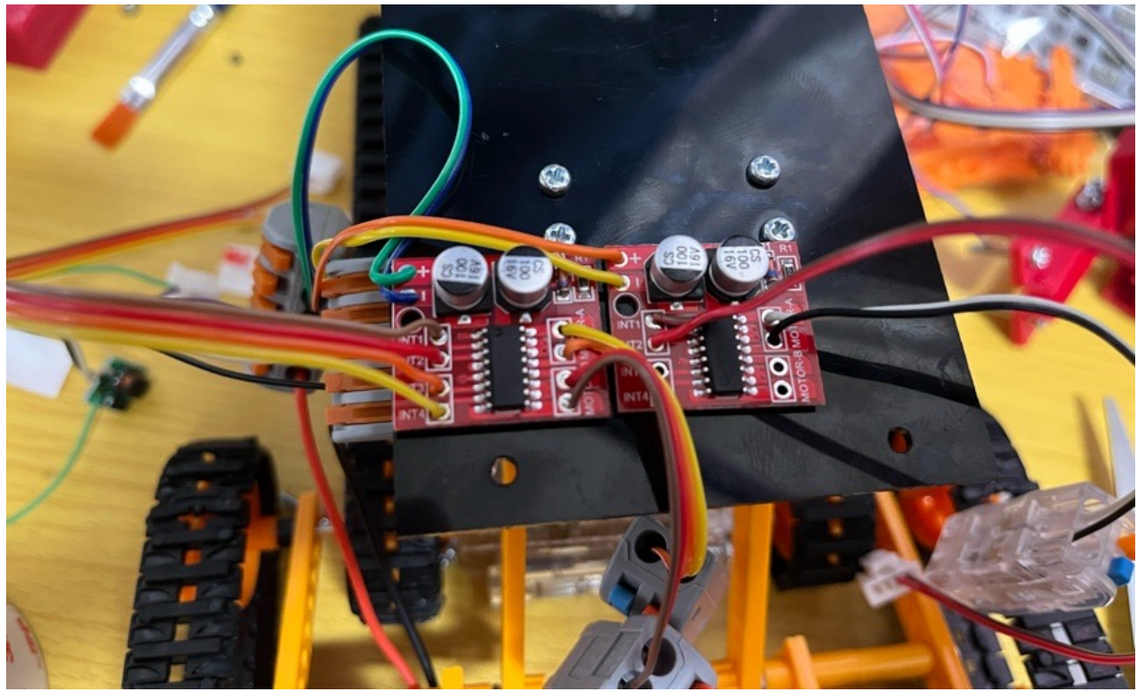


3）ロボットの部品にベースステーを写真のように挟み込み、**M2.5x20**ナベ子ネジ**4**本と**M2.5**ナット固定する。

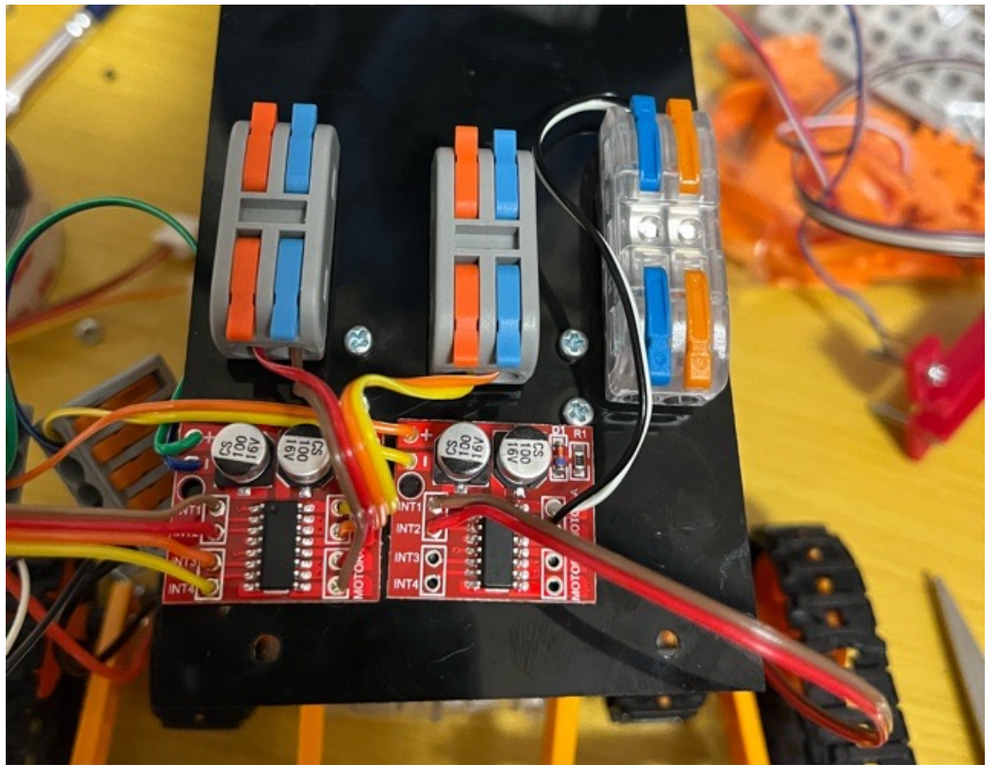




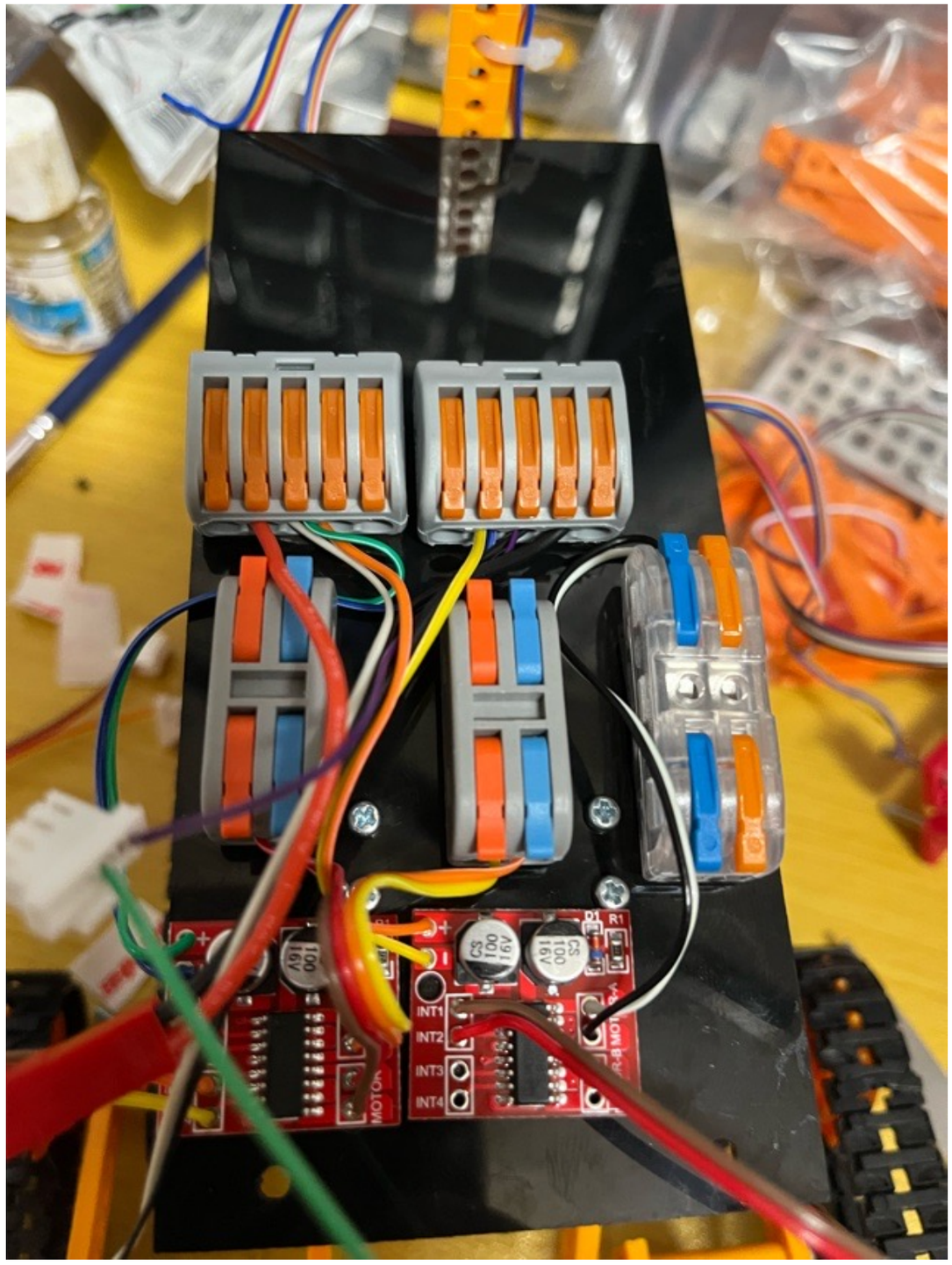
4）両面テープでクローラとフリッパーのモータドライバを貼り付ける。



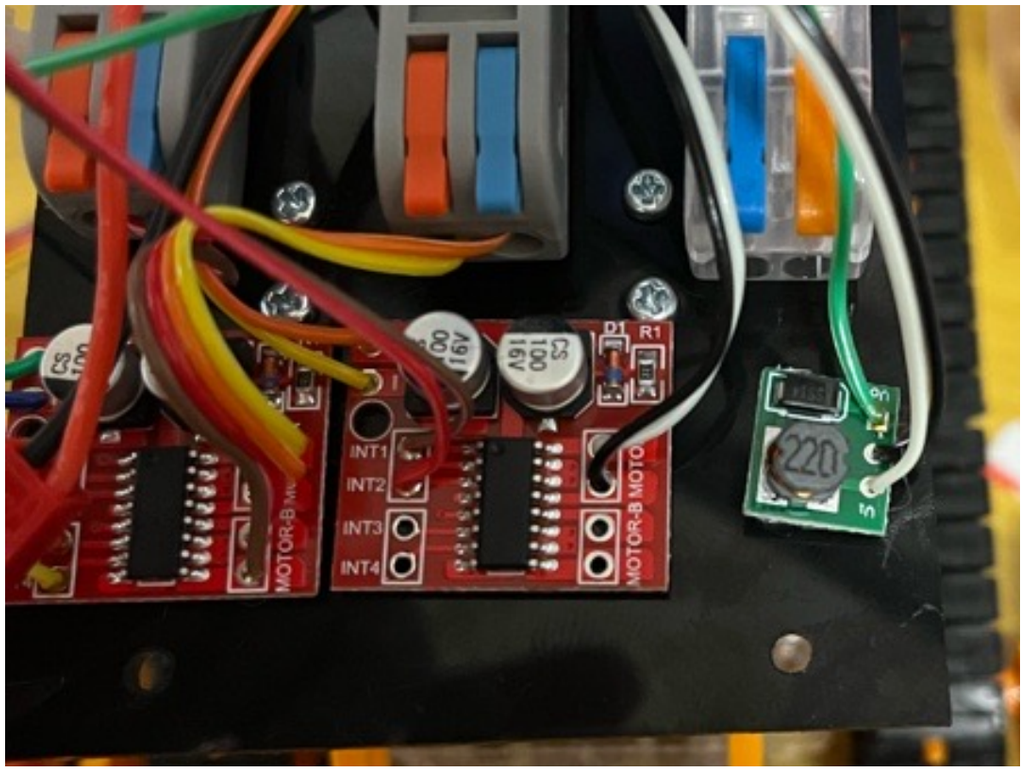
5）両面テープでモーターコネクタを貼る。



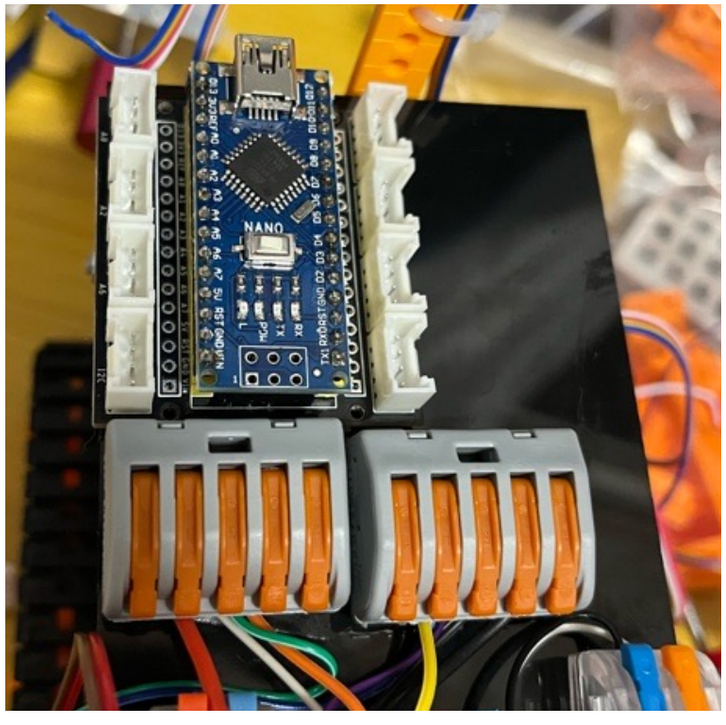
6）両面テープで端子台を貼る。



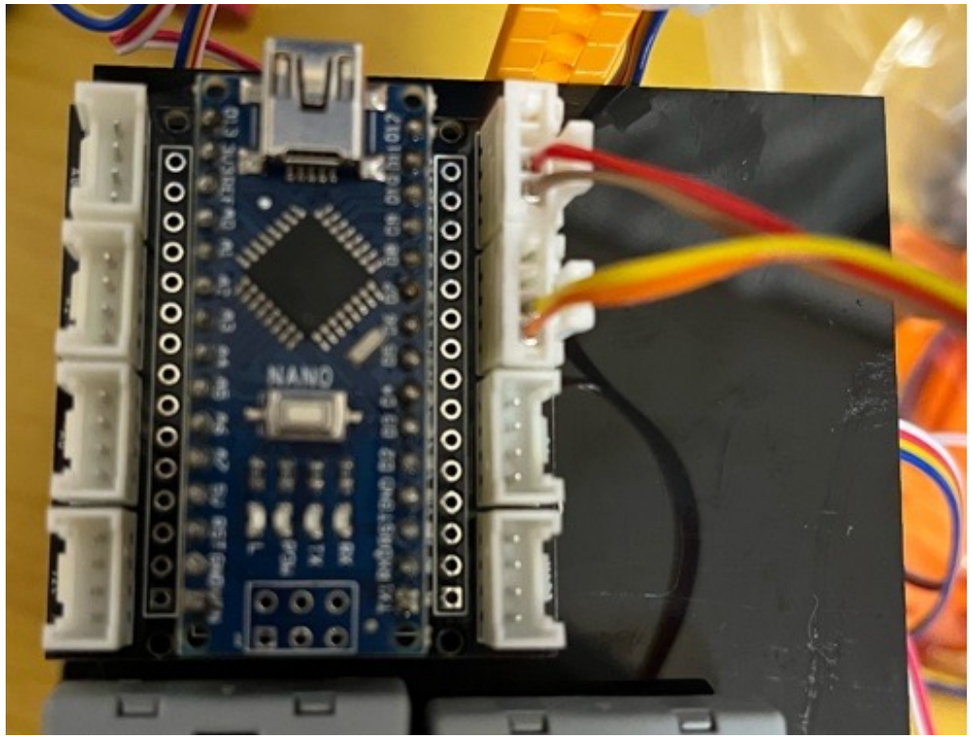
7）**DC-DC**コンバーターを両面テープで貼る。



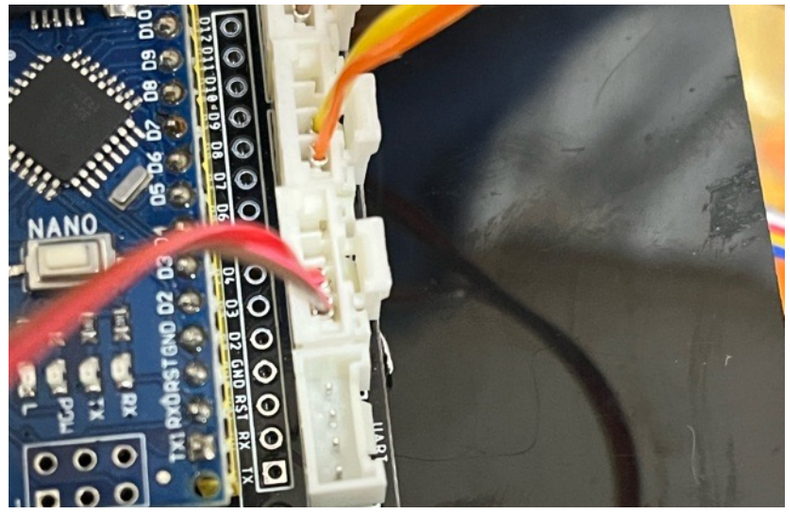
8）**Arduino**を乗せた**GROVE**ベースシールドを両面テープで貼る。



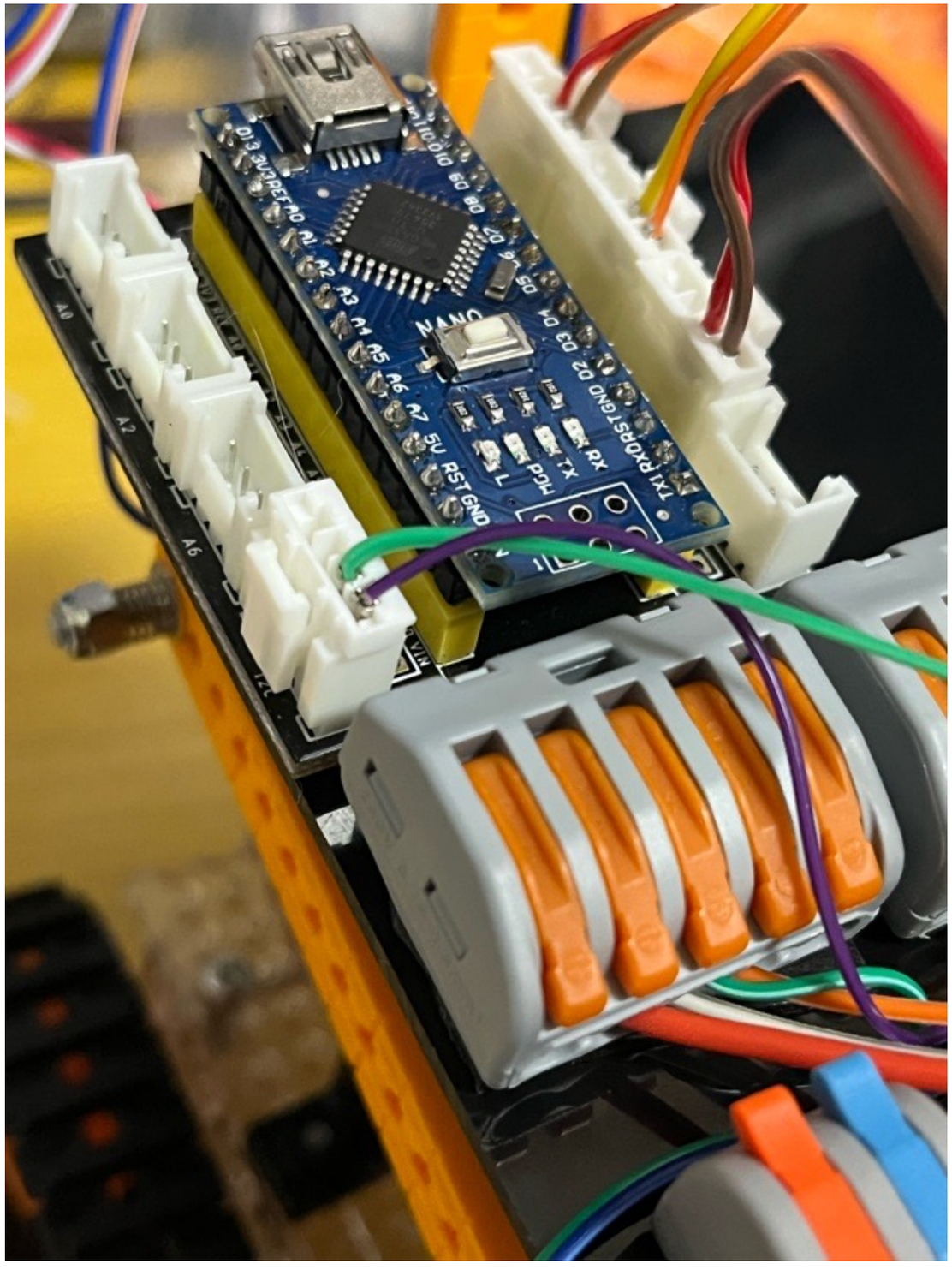
9）クローラモータドライバの**INT1**と**2**のコネクタを**D2**、**INT3**と**4**のコネクタを**D4**へ差し込む。



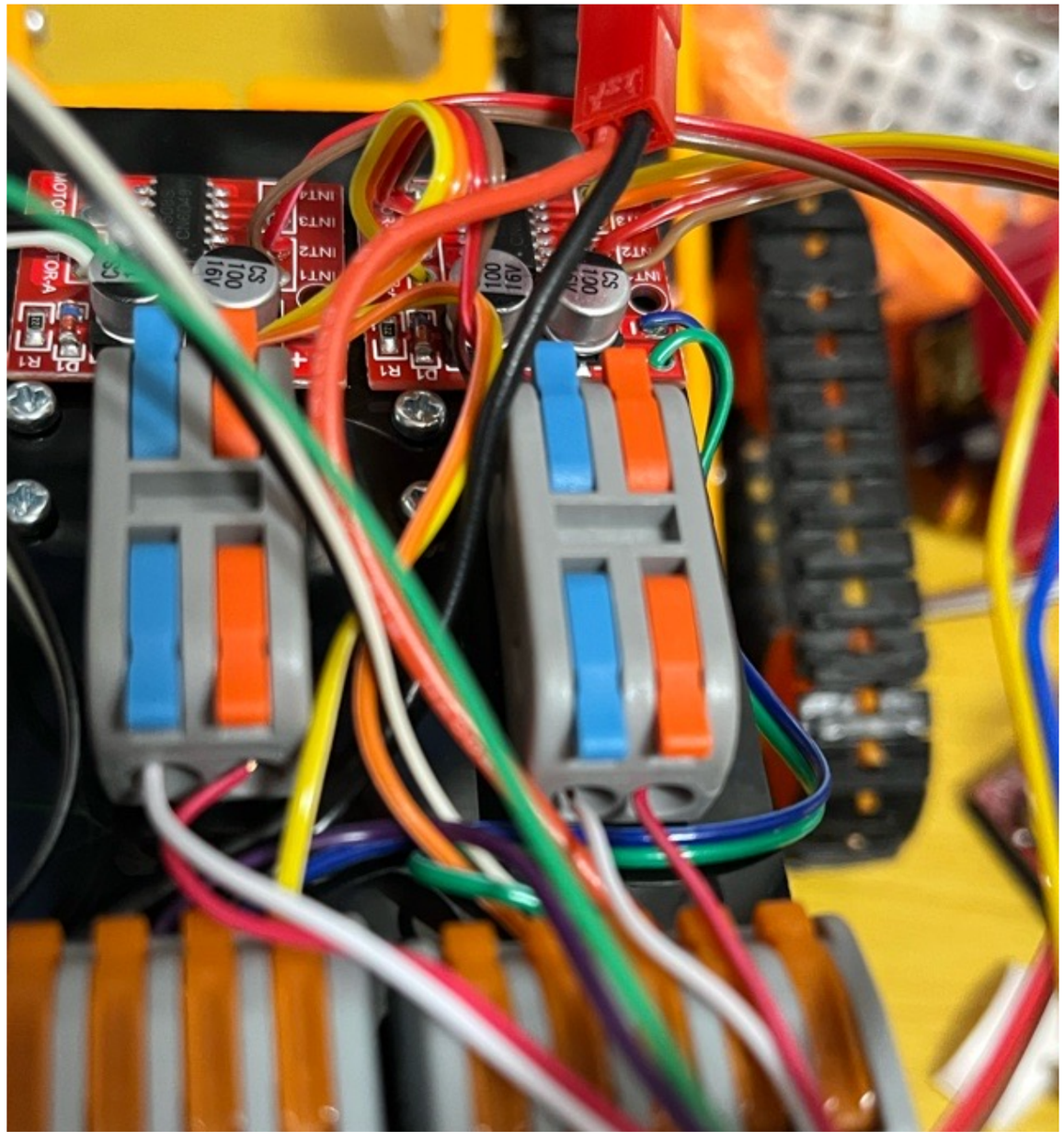
１０）フリッパーモータドライバの**INT1**と**2**のコネクタを**D6**へ差し込む。



１１）**DC-DC**コンバーターのコネクタを**I2C**に差し込む。



１２）モーターコネクタにモーターからの線を差し込む。
左のモーターからの線は**MOTOR-A**に右のモーターからの線は**MOTOR-B**から出てモーターコネクタに入っている線の方に差し込むこと。



１３）クローラモーターから出ている線をクローラモータコネクタに差し込んで完成。

※確認するポイント

1.コネクタの差している場所は間違っていないか？

2.線が外れているところはないか？

3.電子部品がベースにちゃんと張り付いているか？

１４）次のページから**ArduBlock**を使ってこのレスキュークローラロボットを**Arduino**で動かしてみるよ。

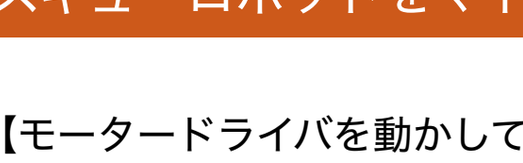
　動かす時にはリポバッテリーを使うけど、間違ってバッテリーコネクタに差して無理に押し込むとショートして大変に危険なので、差し込む際には、ちゃんとバッテリーの赤い電線、黒い電線とコネクタの赤い電線、黒い電線が合っていることを確認してから差し込む習慣をつけよう。

【モータードライバを動かしてみよう】

モータドライバは**IN1**と**IN2**、**IN3**と**IN4**へ渡す信号の組み合わせで動きが決まる。**(IN3,4**の設定はモータ**B**に反映)

IN1	**IN2**	モータ**A**の動き
LOW	LOW	停止
HIGH	LOW	正転
LOW	HIGH	逆転
HIGH	HIGH	ブレーキ

Arduinoの**D2**にモータードライバの**IN1**、**D3**に**IN2**、**D4**に**IN3**、**D5**に**IN4**がつながっているので、**IN1**の信号を**LOW**にするのには、



このようにブロックを設定する。

逆に**IN1**の信号を**HIGH**にするのなら、

このように設定する。

モータードライバへの信号は、２つの**IN**端子が１セットなので、モーター**A**を停止させるなら、このように部品を配置すればよい。

さてここまで説明すれば、モーター**A**を正転状態にする方法はもうわかると思う。

モーター**A**を１秒間正転させて、停止するプログラムを書いてみよう。
下の図はデジタルピンに値を設定のところの部品が不足している。回るように補ってみよう。

プログラムを書き込んだ時にモーターは左側のクローラが回れば正解。

・逆の方のモーターが回った人は、クローラモーターコネクタの配線を交換してみよう。

・前進ではなく後退した人は、モーターコネクタの左右の配線を変えてみよう。

さて、クローラは左が反時計回りに、右が時計回りに回ることで前進をする。

今度は、**IN3,IN4**のモーターを前進させるプログラムを書いてみよう。

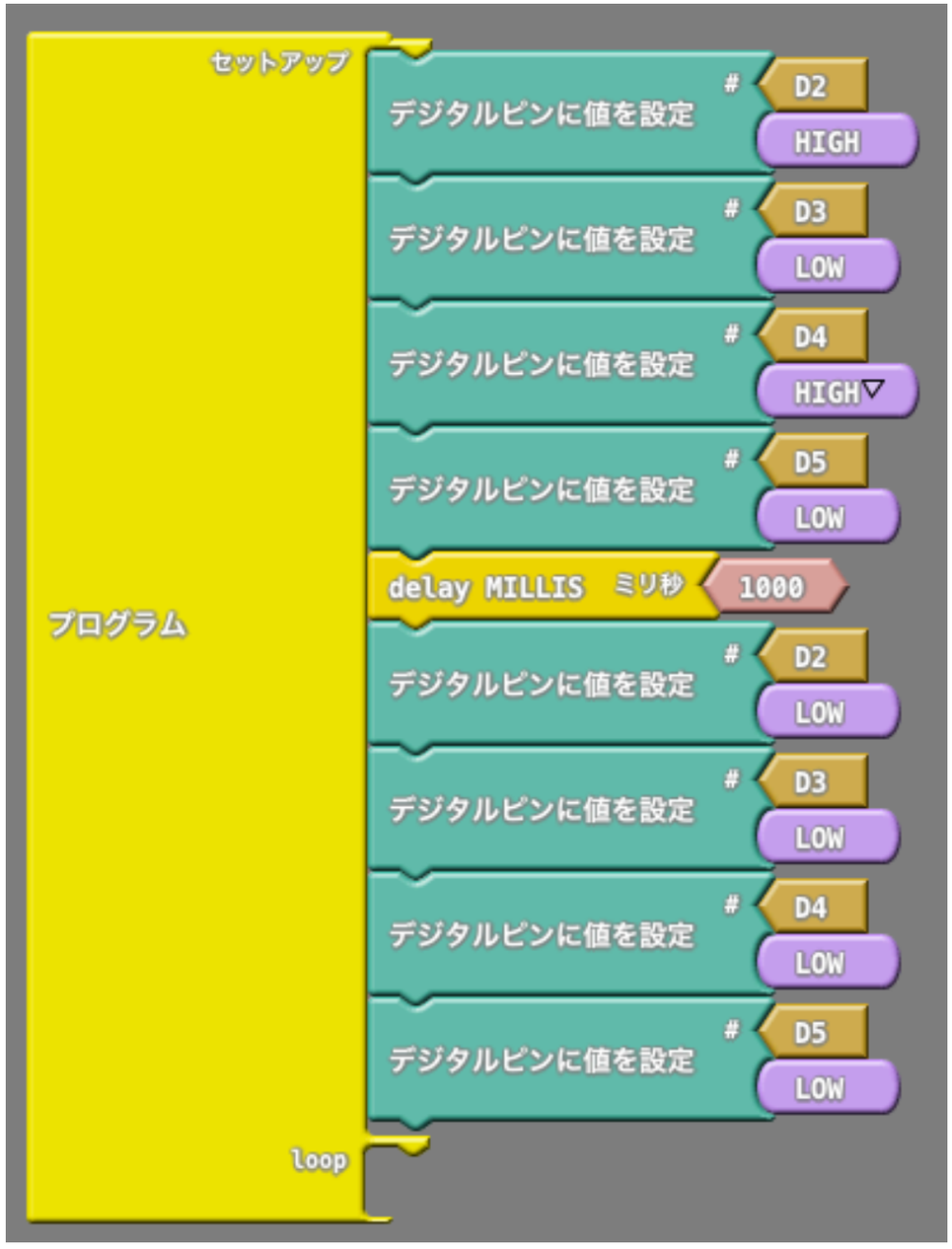
今のプログラムの**D2,D3**を**D4,D5**に変えるだけで反対側のモータが動くはずだよ。

そして**IN1,2,3,4**の信号を１セットにすると左右のクローラが回って前進するプログラムになる。

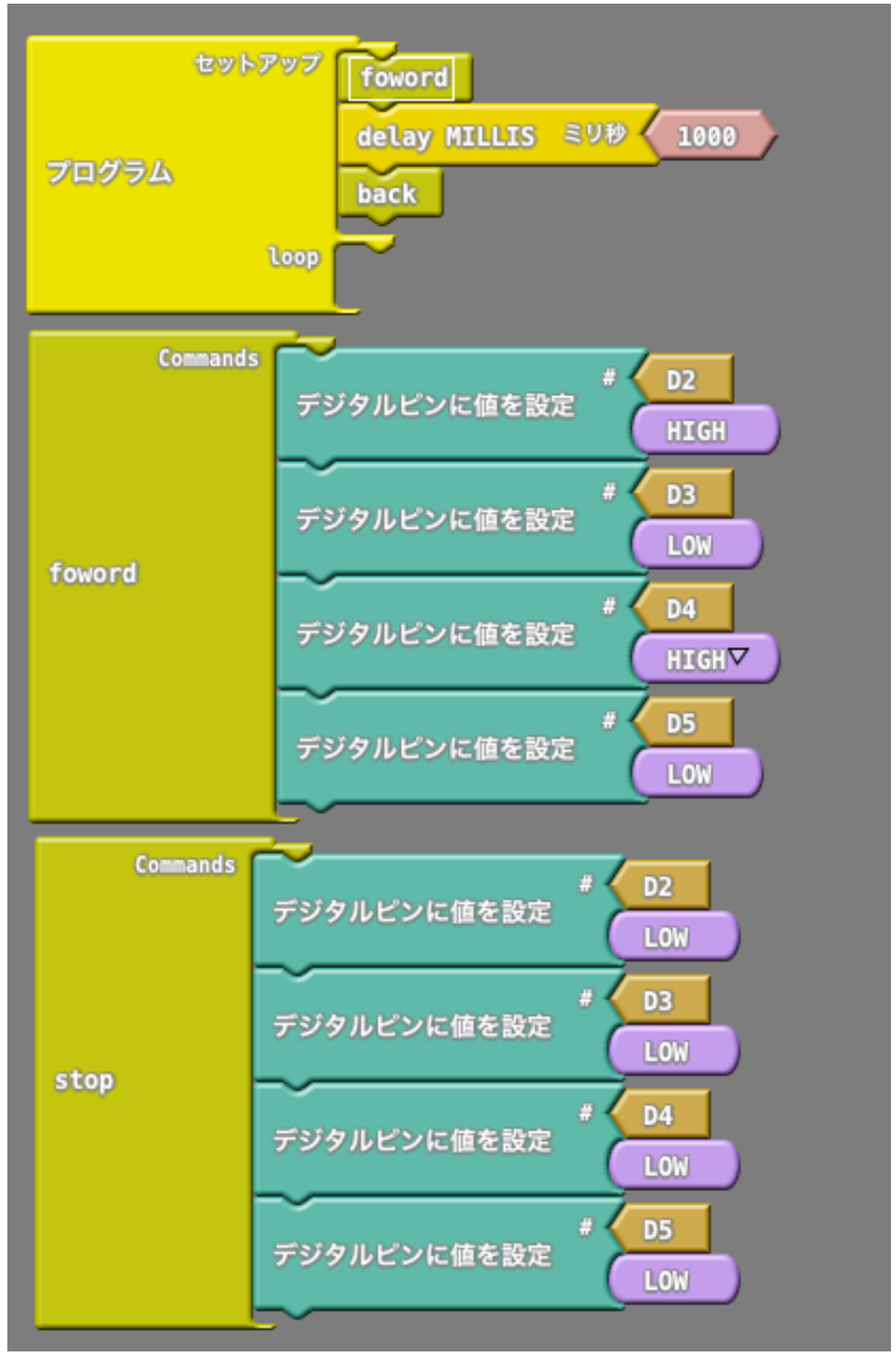
さて、これが前進するプログラムになるのだけど、これをサブルーチンにすると使いやすくなる。

【サブルーチンを使ってみよう】

　このプログラムは、前進・１秒その状態を保つ、そして停止するプログラムなので、前進(foword)、delay MILLIS1000、停止(stop)に分けることができる。



サブルーチン化するとこんな感じに分かりやすくなるんだ。
これなら、プログラムのところの流れを見るだけで
「前進」「１秒そのまま」「停止」になっているのがわかる。



【後退、右旋回、左旋回を自分で考えて作ってみよう】

D2とD4をHIGH、D3とD5をLOWにすると前進する制御ができた。

この信号を全部逆にすると後退するプログラムが作れる。
早速作ってみよう。

サブルーチン名は back にしようか。
では左旋回と右旋回を考えてみよう。

左旋回は、左側のクローラーを後退させて、右側のクローラーを前進させるとその動きになる。

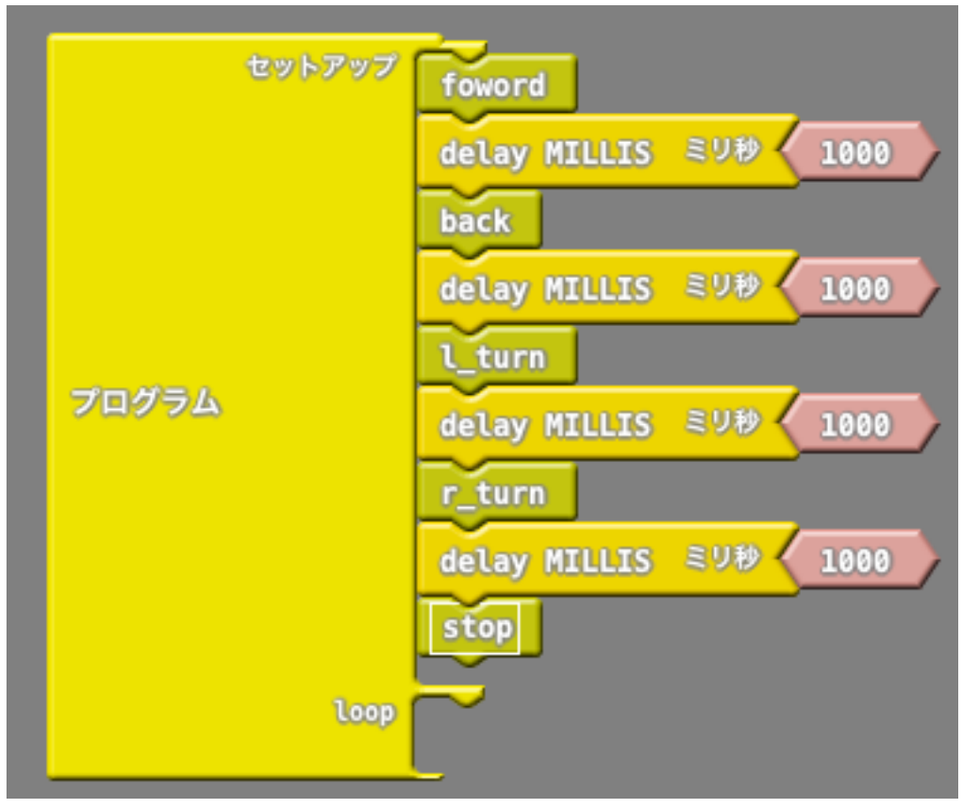
右旋回はその逆で、左側のクローラを前進させて、右側のクローラーを後退させる。

それぞれの名前をl_turn、r_turnとしてサブルーチンを作ってみよう。

さて、ここまでのプログラムでクローラロボットを自由に前後左右に動かせるようになった。

プログラムからこのようにサブルーチンを呼び出して、思った通りの動きになるか試してみよう。

この後に、自分でいろんな動きを作って（Delay MILLISの数字も色々と変えてみて）試してみよう。



　ここまで制御を理解して自分で作れるようになると、フリッパーの上下は自分で作れるようになると思う。

　フリッパーのモータードライバは**D6**と**D7**で制御できる。

サブルーチンを**f_stop**、**f_up**、**f_down**で作ってみよう。

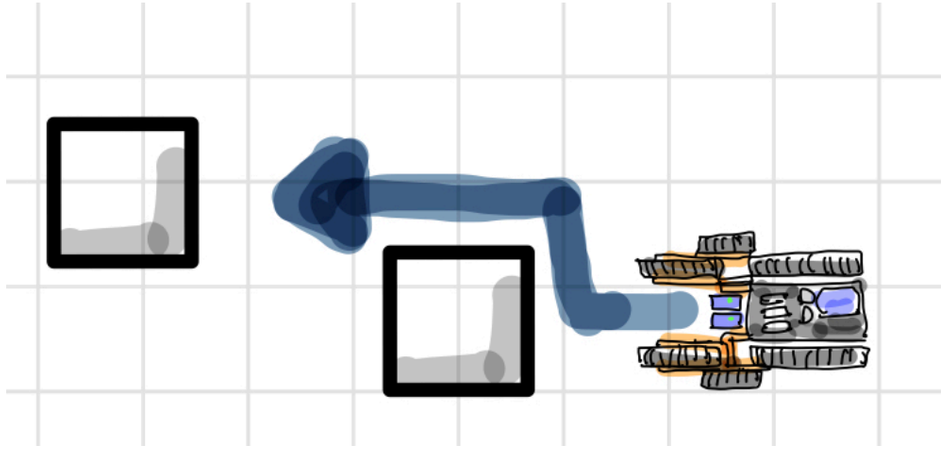
　ちなみに、このフリッパーは、１回転すると上下の動きが逆になる。

　なので、プログラムで制御する場合には回しすぎないように使うのがコツだ。

作ったプログラムは「**rescue1**」という名前でパソコンに保存しておこう。

【応用編】

　前進や旋回の動作と待ち時間を上手に調整して、置かれた箱などを回避して動くプログラムを作ってみよう。

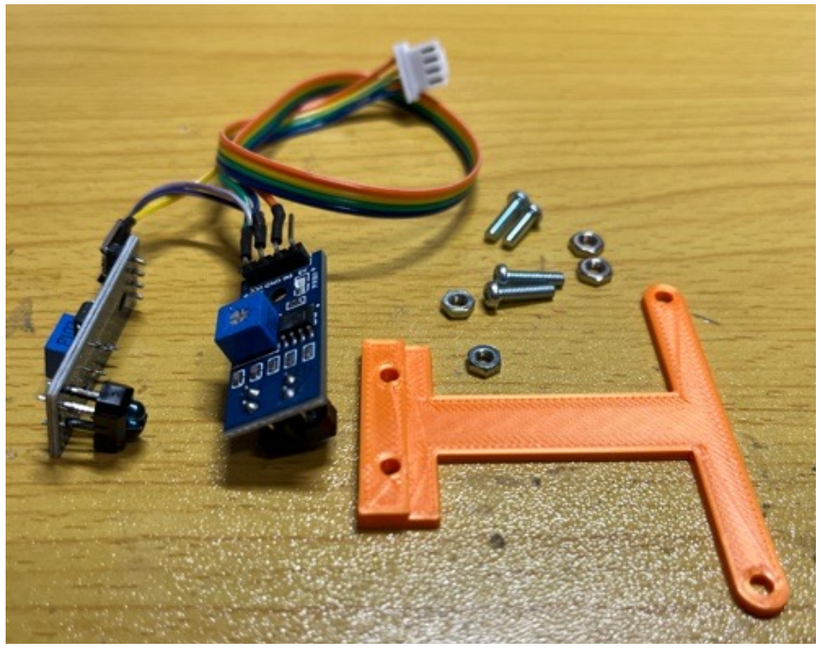


【ライントレーサーに改造してみよう】

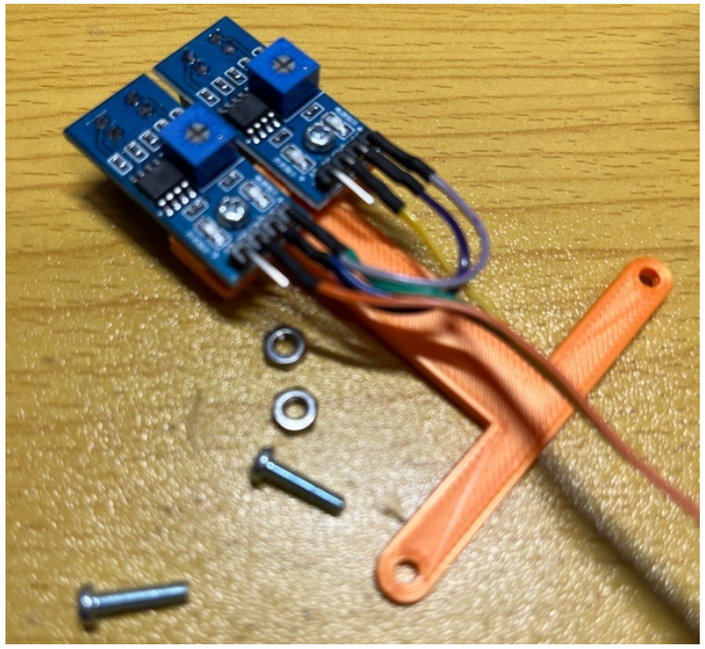
　このレスキュークローラーには、ラインセンサーを取り付けることができる。
　ラインセンサーを取り付けてプログラムを作ると、白い紙などに書かれた黒い線をたどって動く動作を行うことができる。

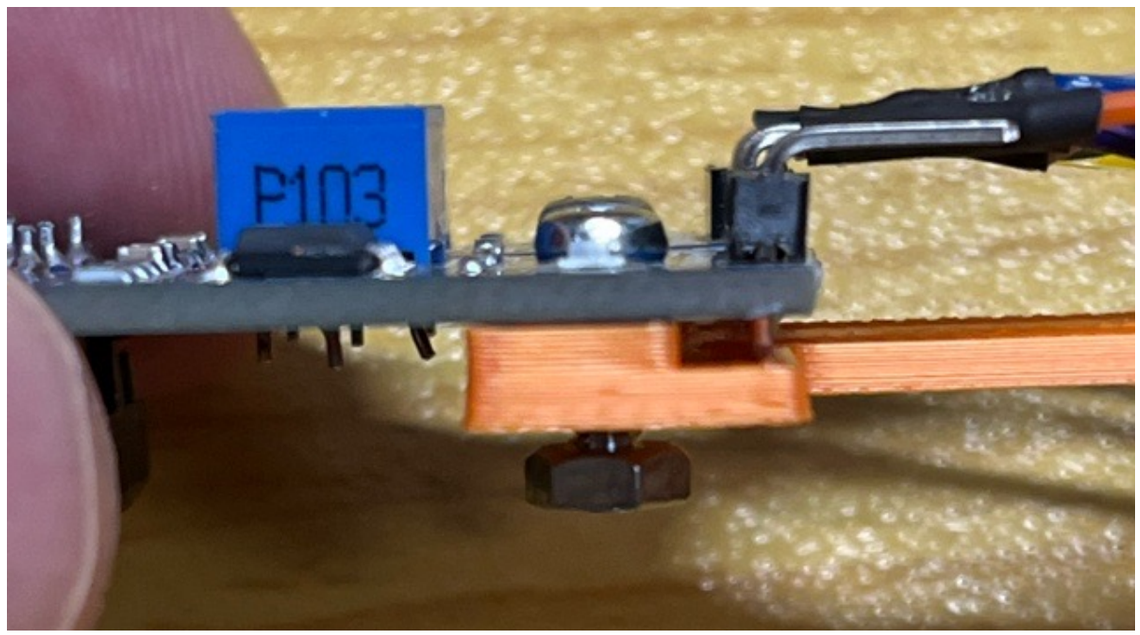
　早速その部品を付けてライントレース機能を拡張してみよう。

1）ラインセンサ**ASY**とラインセンサーステー、**M2.5x8 2**本、**M2.5x10 2**本、**M2.5**ナット**4**個を準備する。

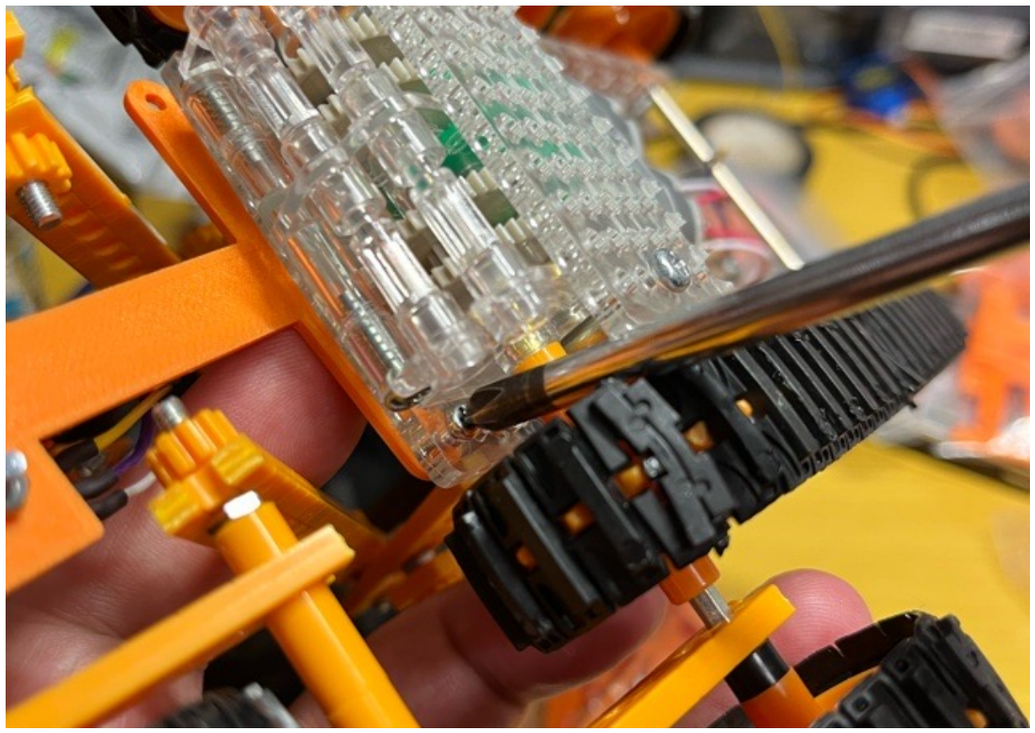


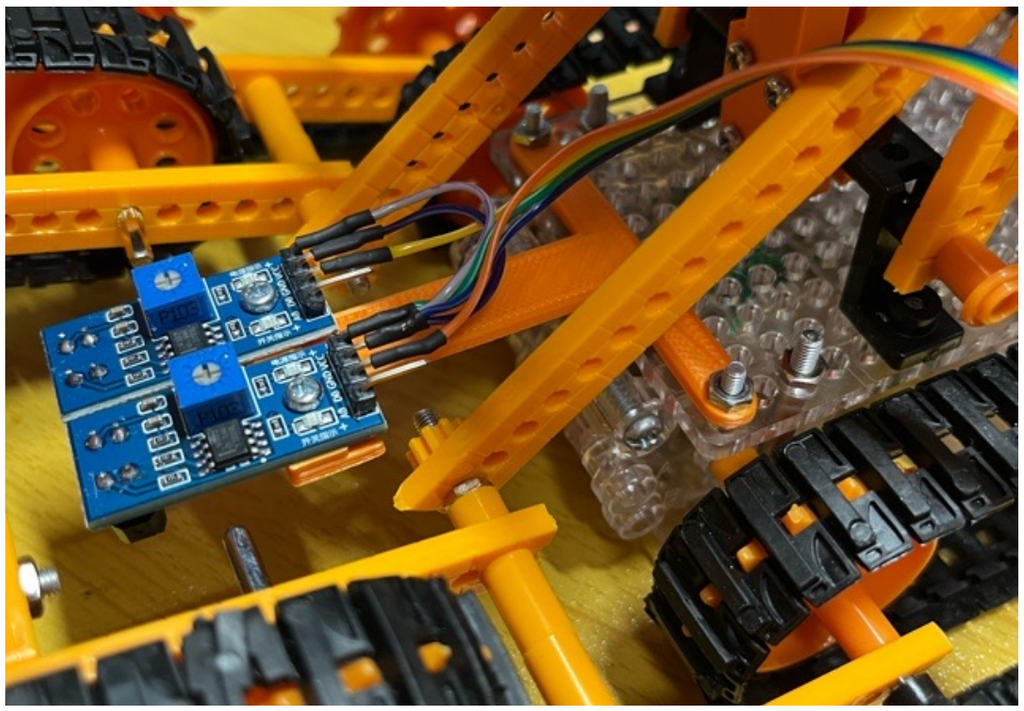
2）**M2.5x8 2**本でラインセンサーをラインセンサーステーに取り付ける。
この時、段差のある面を上にする。



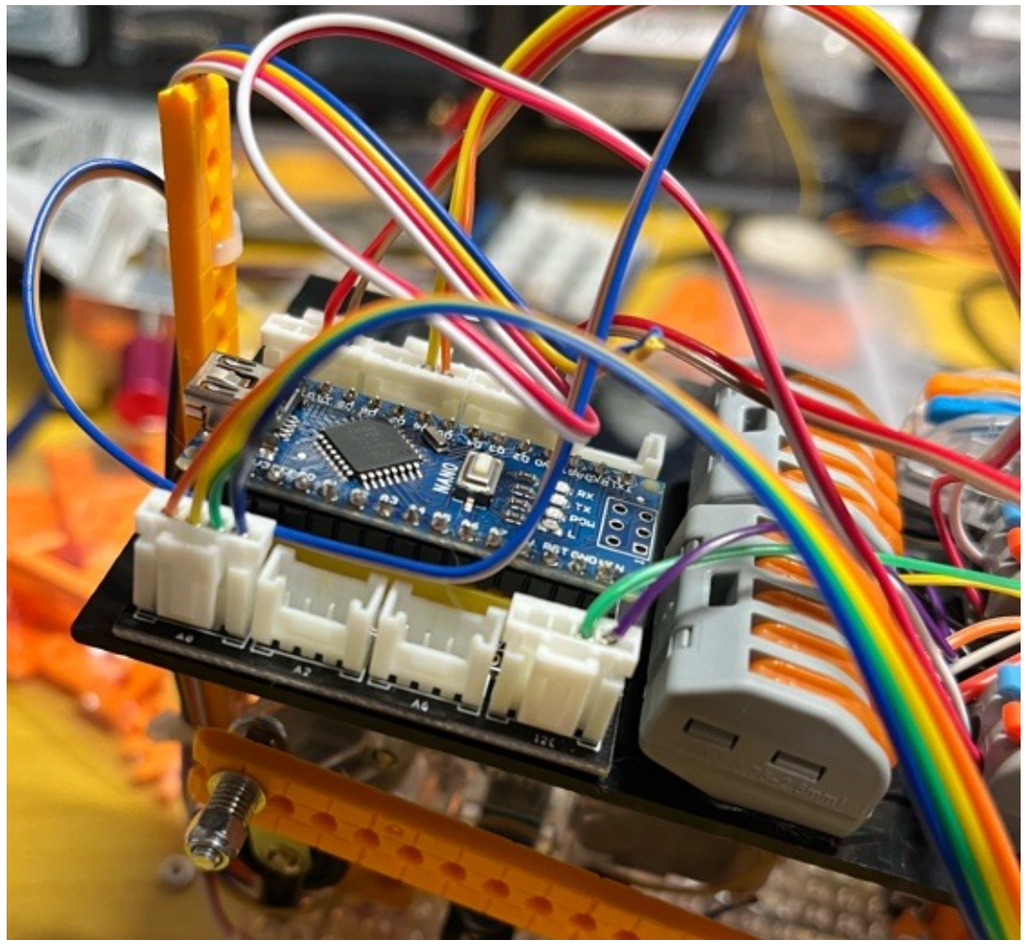


3）**M2.5x10 2**本でラインセンサーステーをロボットの裏側から固定する。





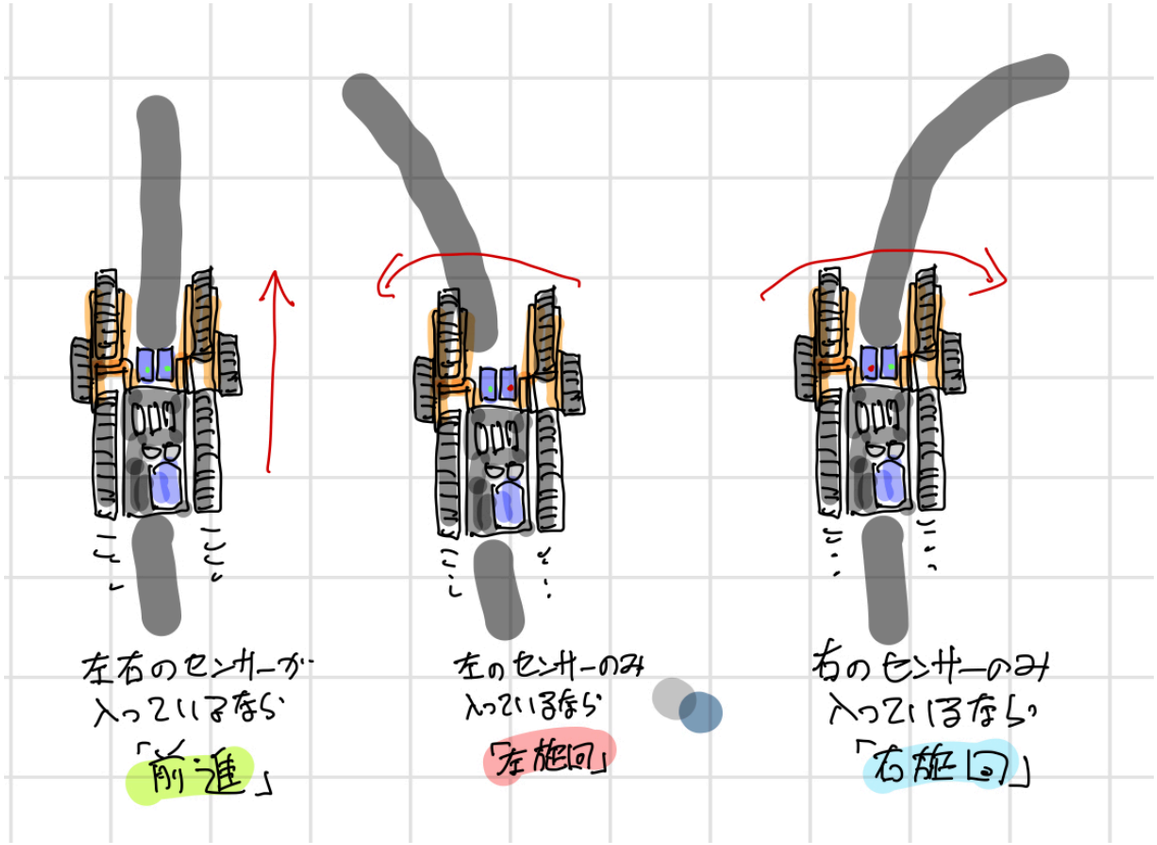
4）ラインセンサー**ASY**からのコネクタを**A0**に差し込み取り付けが完了する。



5）**A0**には左側のセンサーが**A1**には右側のセンサーが結線されているのでセンサーの入力を行える。

【ラインセンサーのプログラムを考える】

ラインセンサー**ASY**には左右にセンサーが並んでいる。
この２つのセンサーが黒いラインを検出しているときは、ロボットがラインから外れていないとして前進する。



　右のセンサーが外れて左のセンサーのみ入っている状態ならば、ラインから右にずれ始めているとして左旋回をして姿勢を修正し、左のセンサーが外れて右のセンサーのみ入っているのであれば、左にずれ始めているとして右旋回を行う。
　この動きを繰り返すことでラインをトレースして移動する。

１）センサーからの入力の仕方

　ラインセンサーには、トリマーボリュームが付いていて黒い線と白い地面の検出度合いを調整できる。
　調整方法は黒い線にセンサーを乗せた時にセンサーのインジケーターが消灯、そこから外れたら点灯するようにトリマーボリュームを回して調整する。
　反時計回りにすると感度が落ちて、時計回りに回すと感度が上がる。

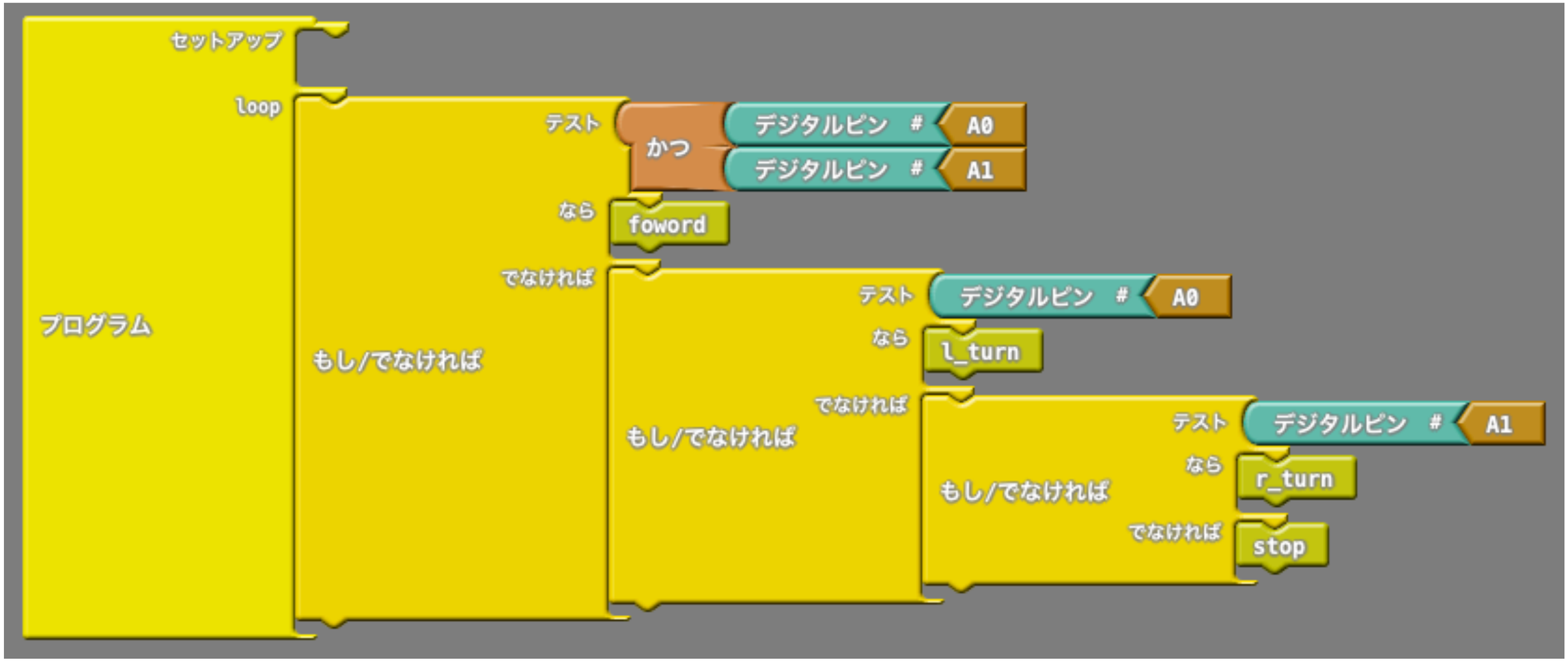
２）センサーの値の読み取り方

　デジタルピンの読み出しを使ってセンサーの入力を表示してみよう。

　L=は左センサーの入力、**R=**右センサーの入力状態がシリアルモニターで確認することができる。
　ちゃんと入力できているか確認してみよう。
出来上がったプログラムは「**rescue2**」でパソコンに保存しよう。

３）ラインセンサーの入力を判断してロボットの動作を決めるプログラム

　左(AO)と右(A1)が両方検出（黒い線の上にある）ならば前進動作をする。左(AO)だけならば左旋回をして右も入るようにする。右(A1)だけなら左も入るように右旋回を行う。左右（A1,AO)が入っていないならば停止する。



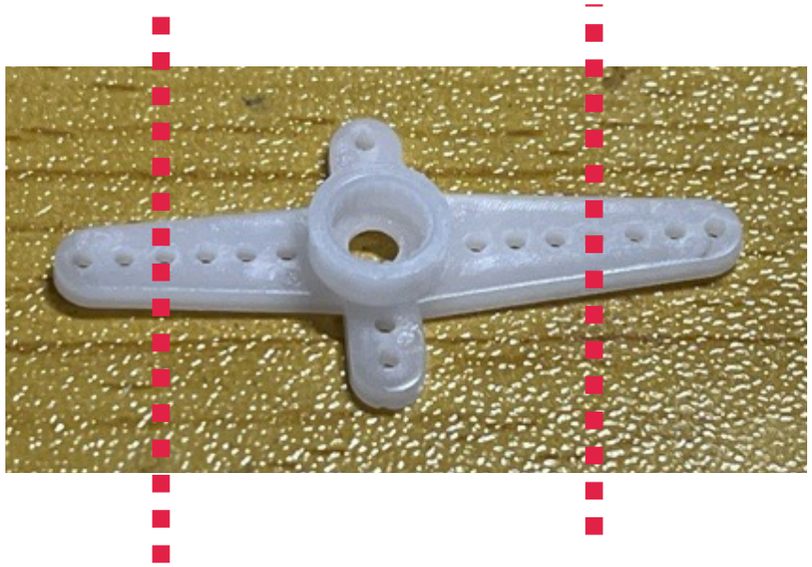
さて、これでうまく動くだろうか？論理的には合っているようだけど、実際にはロボットの速度やセンサーの反応速度、処理速度などの関係でうまく動かない方が多かったりする。

【仕事ができるロボットに改造しよう】

　これまでクローラーロボットを改造して、プログラムを組み立てて動くようにしたり、ライントレースをして自律移動するようにしてみた。

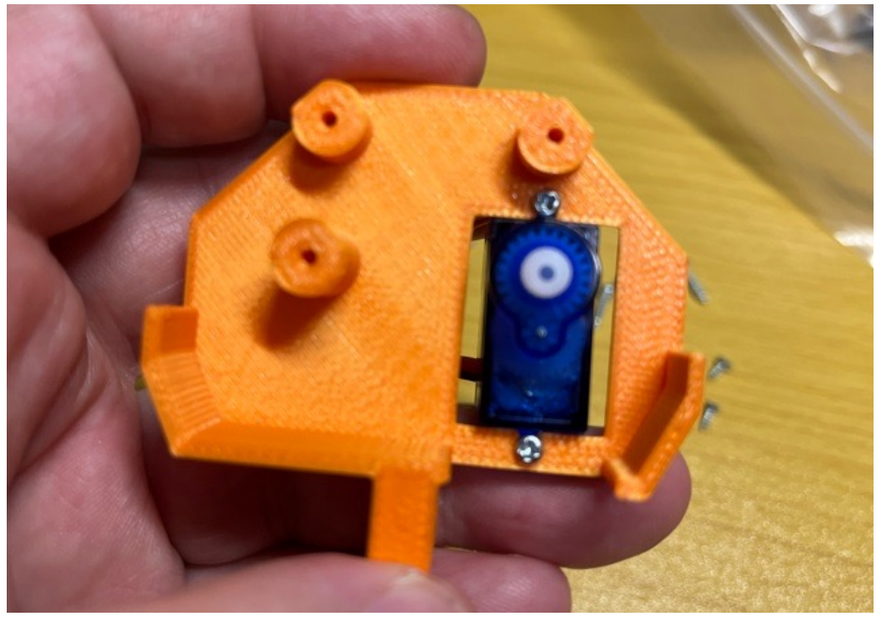
　しかし、移動だけで物を掴んで運んだりするようなことはできない。
　そこでこのロボットにハンドを付けてみることにしたよ。

１）サーボモータの袋に入っているこのサーボホーンを取り出し、穴が４つ目のあたり（赤線部分に沿って）ニッパーで切断する。



２）アームボディ、ギヤA、ギヤB、リンクA(2個)、リンクB(2個)、フィンガー(2個)、ステーA、ステーB、M3x16 2本、M2.5x15 2本、2.5x10タッピングビス 5本、皿ビス2x5 5本、M2x6 2本、M3ナット 2個、M2.5ナット 2個、M2ナット 2個、サーボモータを準備する。

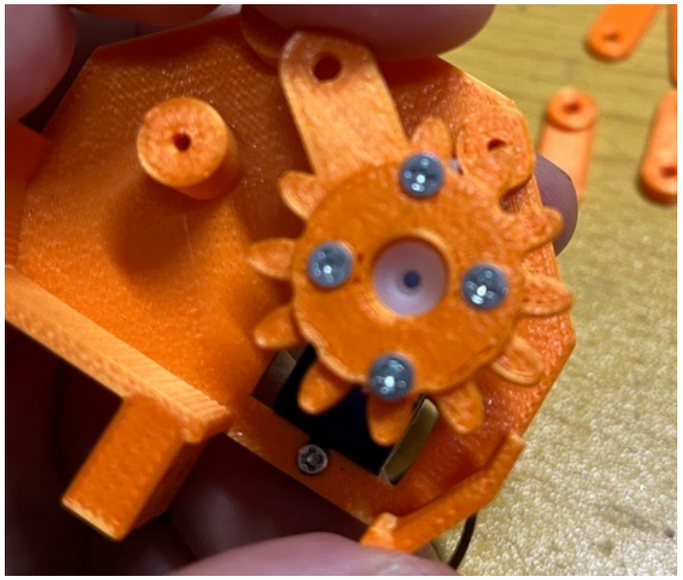
３）アームボディにサーボモータを**M2x6 2**本と**M2**ナット**2**個で写真のように固定する。



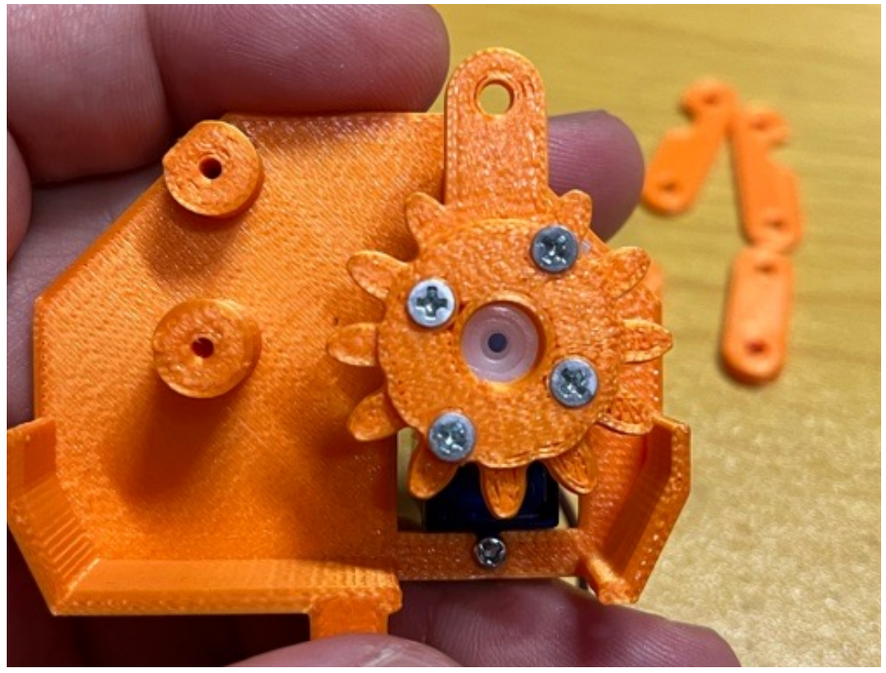
４）ギヤ**A**に**1)**で加工したサーボホーンを皿ビス**2x5 4**本で固定する。その際にギヤ**A**の表裏に注意。

５）サーボモータに**4)**で組み立てた部品を差し込み、反時計方向に回らなくなるまで軽い力で回す。

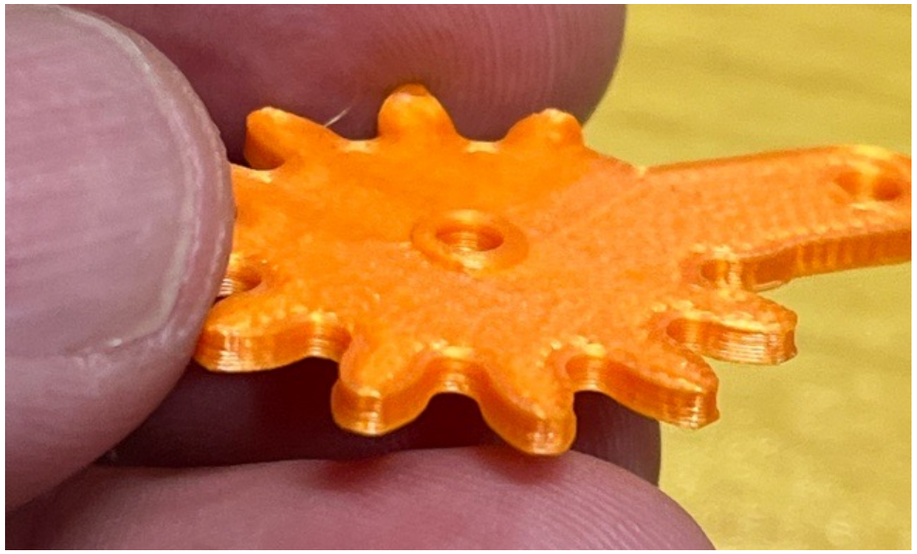
止まったところで、写真の位置に**4)**の部品を差し直す。



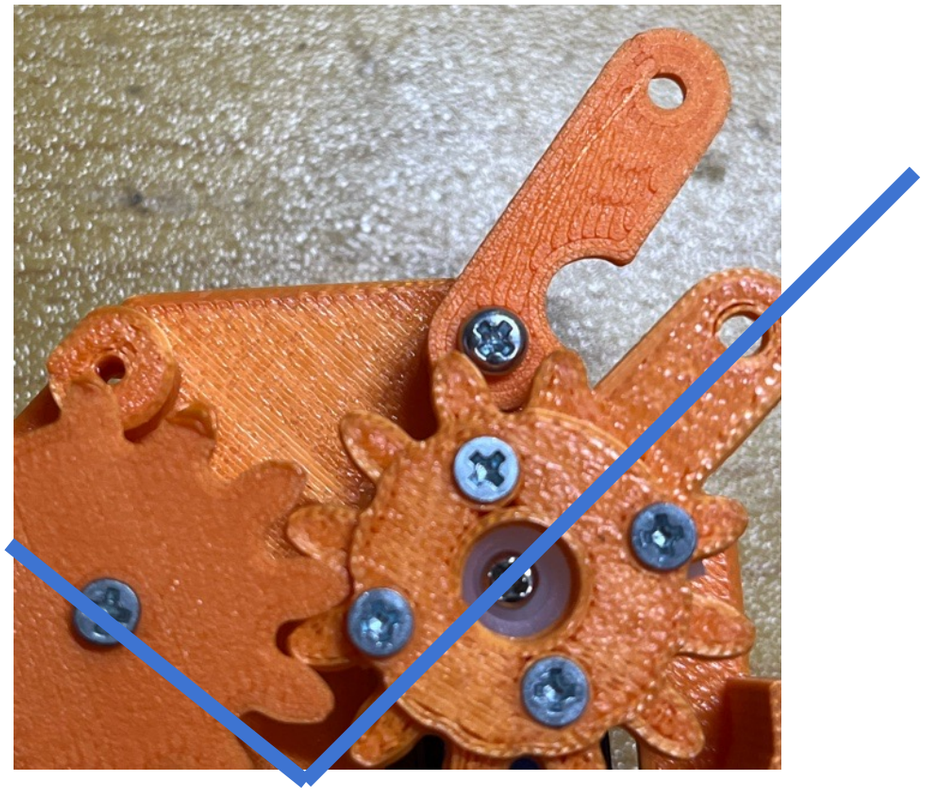
６）止まったところから**4)**の部品を付けたまま写真の位置まで戻す。その後、サーボに付属の一番小さいネジで固定する。



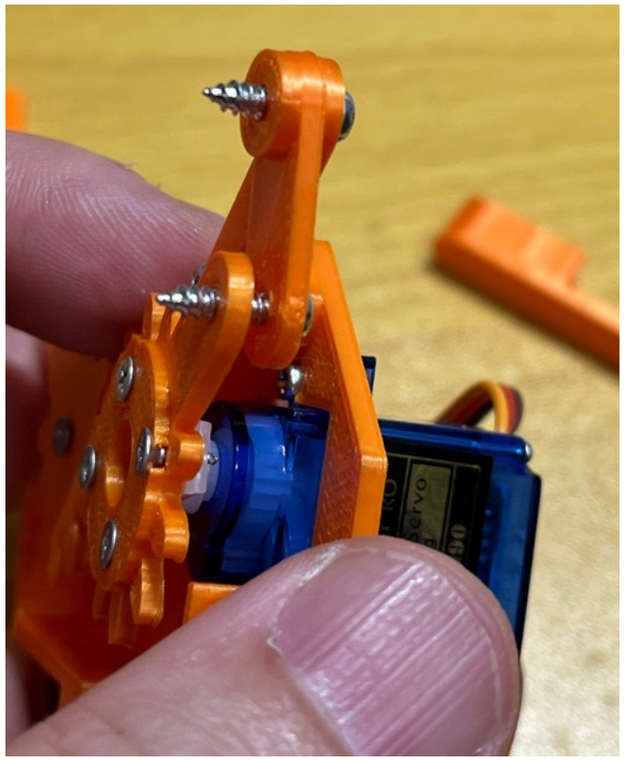
7）ギヤBを用意する。突起がある方が下になりアームボディ側になる。

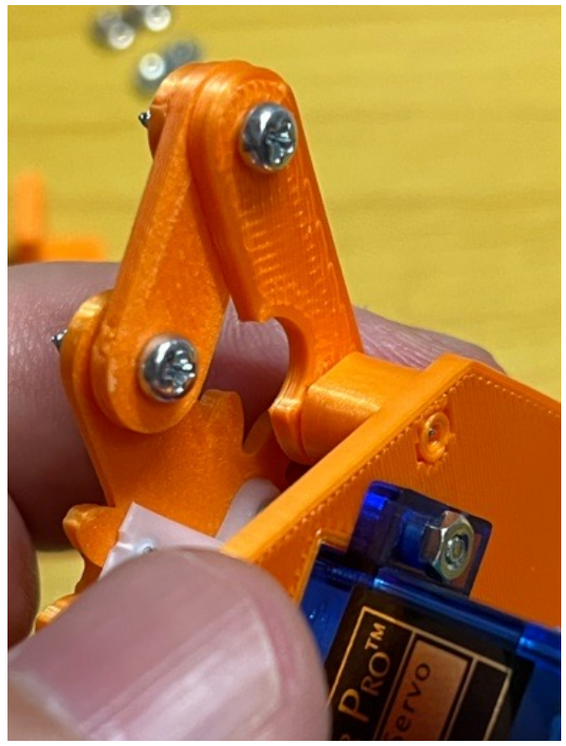


8）ギヤBをギヤAに対して直角になるように取り付けて、皿ビス **2x5** 1本で固定するが動く部分なので締めすぎないようにする。

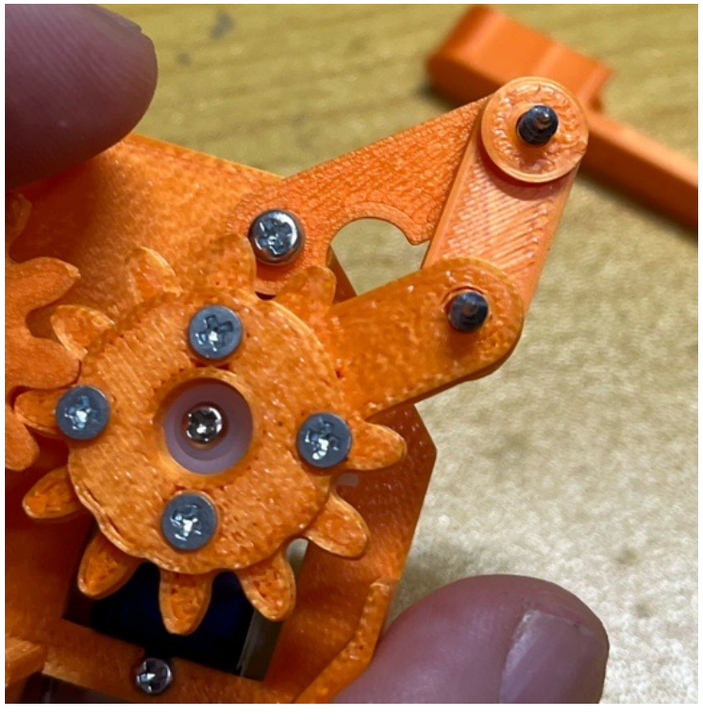


9）リンクAとリンクBを写真のようにギヤAに組み合わせて、**2.5x10**タッピングビス２本を通す。

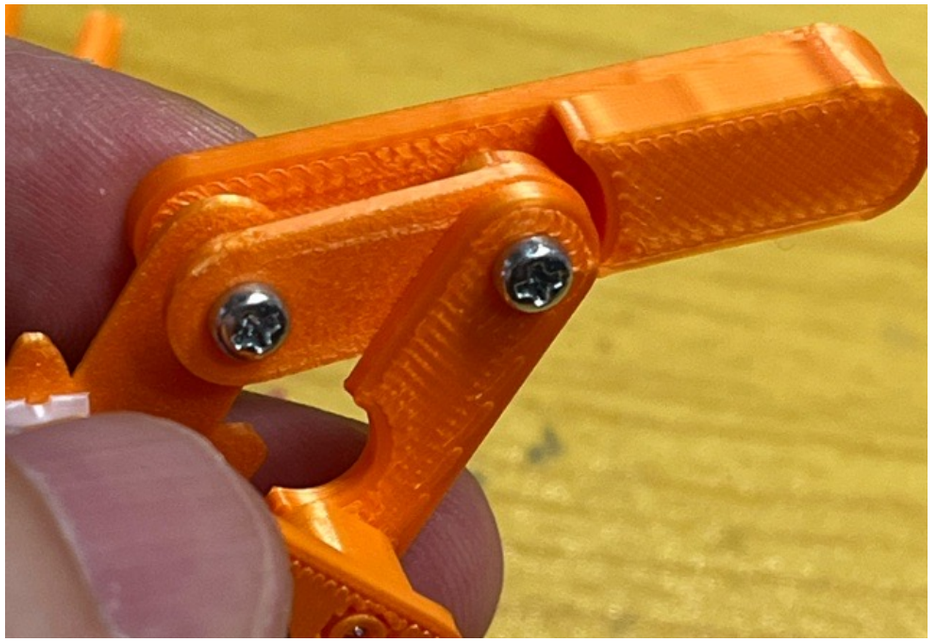




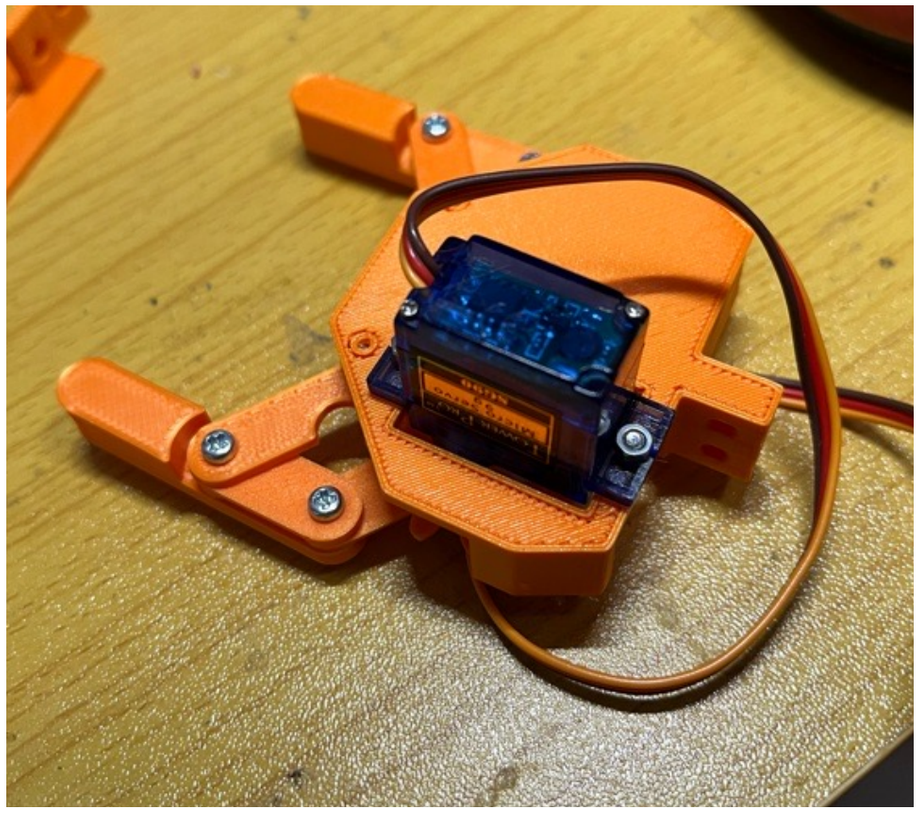
その後、リンクAをアームボディに**2.5x10**タッピングビスで緩く固定する。



１０）フィンガーを**9)**で組み立てたものに取り付け**2.5x10**タッピングビスを締める。

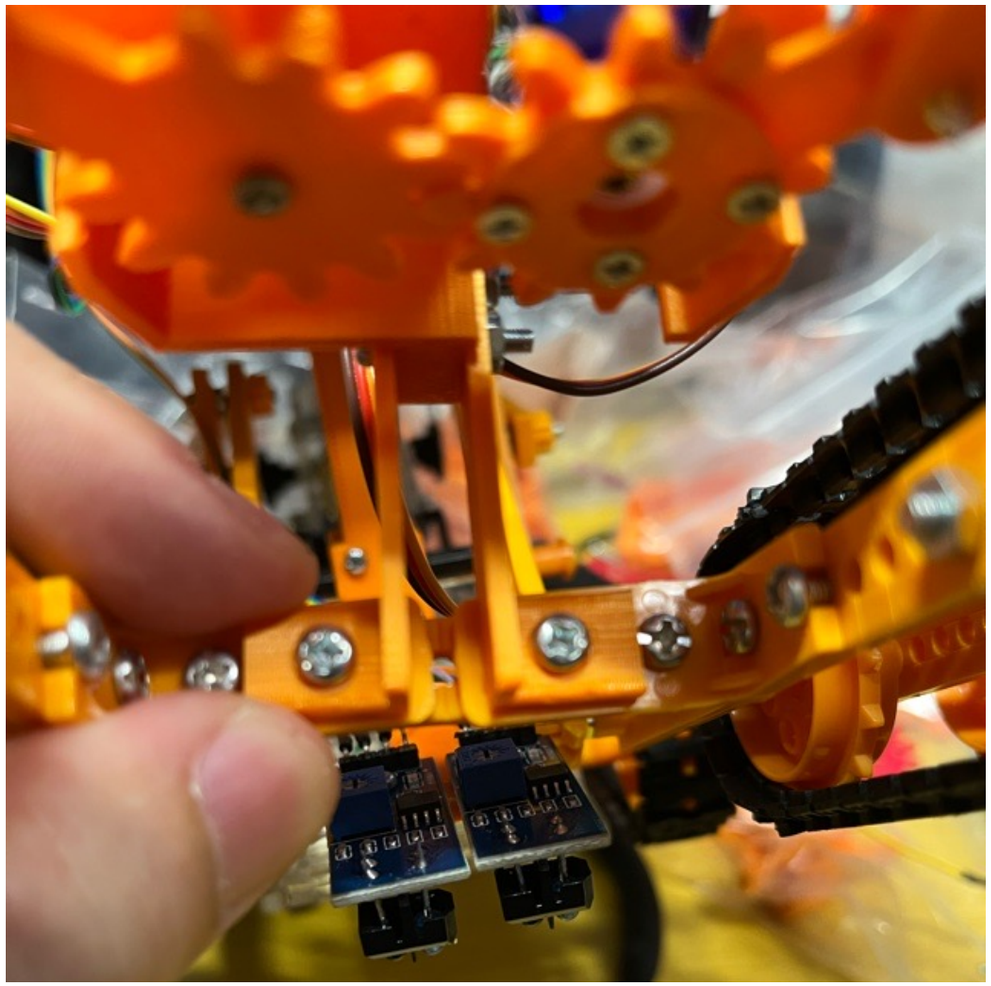


１１）反対側も同じように組み立てる。



１２）ステー**A**と**B**を写真のように**11)**まで組み立てたハンドに取り付け、**M2.5x10 2**本と**M2.5**ナット**2**個で固定する。

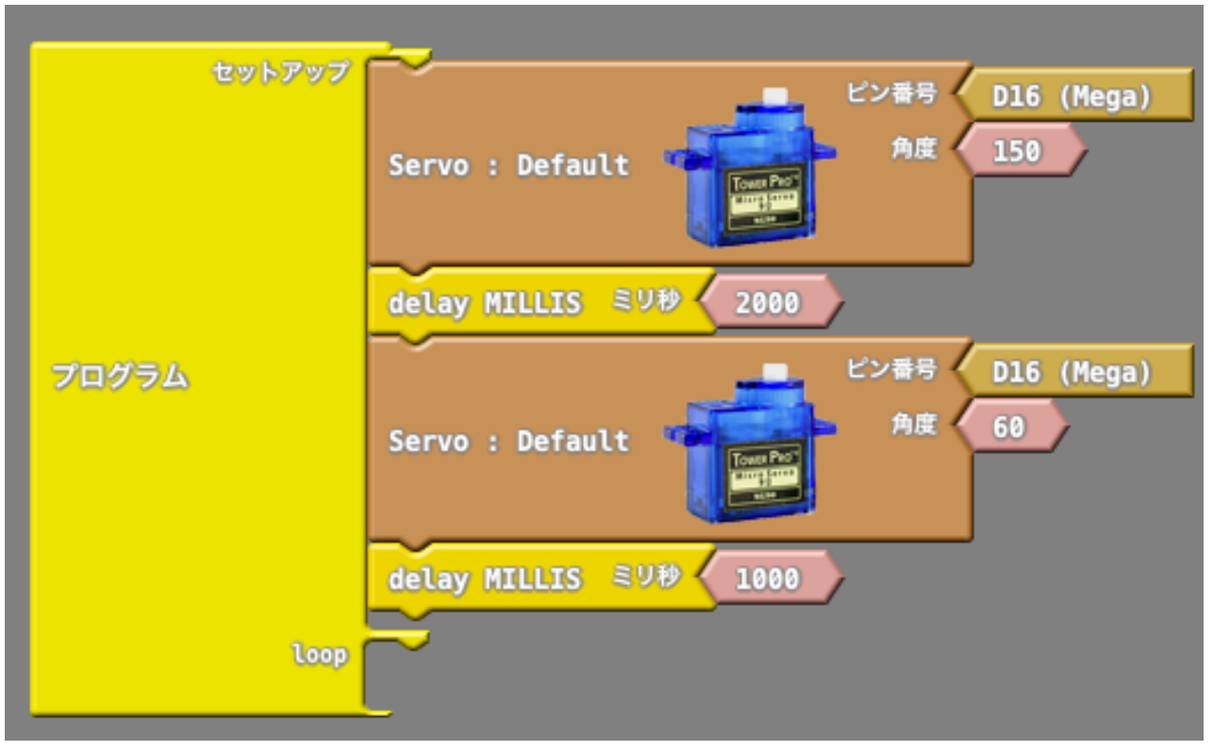
１３）**M3x16 2**本と**M3**ナット**2**個で**12)**まで組み立てたハンドをロボットに取り付ける。



１４）サーボモータから出ているコネクタを**GROVE**の**A2**コネクタに取り付けて完成。

【ハンドを動かしてみよう】

　ハンドはArdublockのサーボの部品で動かすことができる。
Arduino nanoのA2コネクタはデジタル入出力端子のD16端子と共用なので、ピン番号をD16にすること。
このプログラムを書いて動くことを確認してみよう。
（ちなみにサーボ部品の角度の値（150とか60など）は組み付けによる個体差があるので自分のロボットに合わせて調整すること。
　異音が出たり、動きがおかしいときは90に近づく様に調整すること。



できたプログラムは「rescue3」でパソコンに保存しよう。

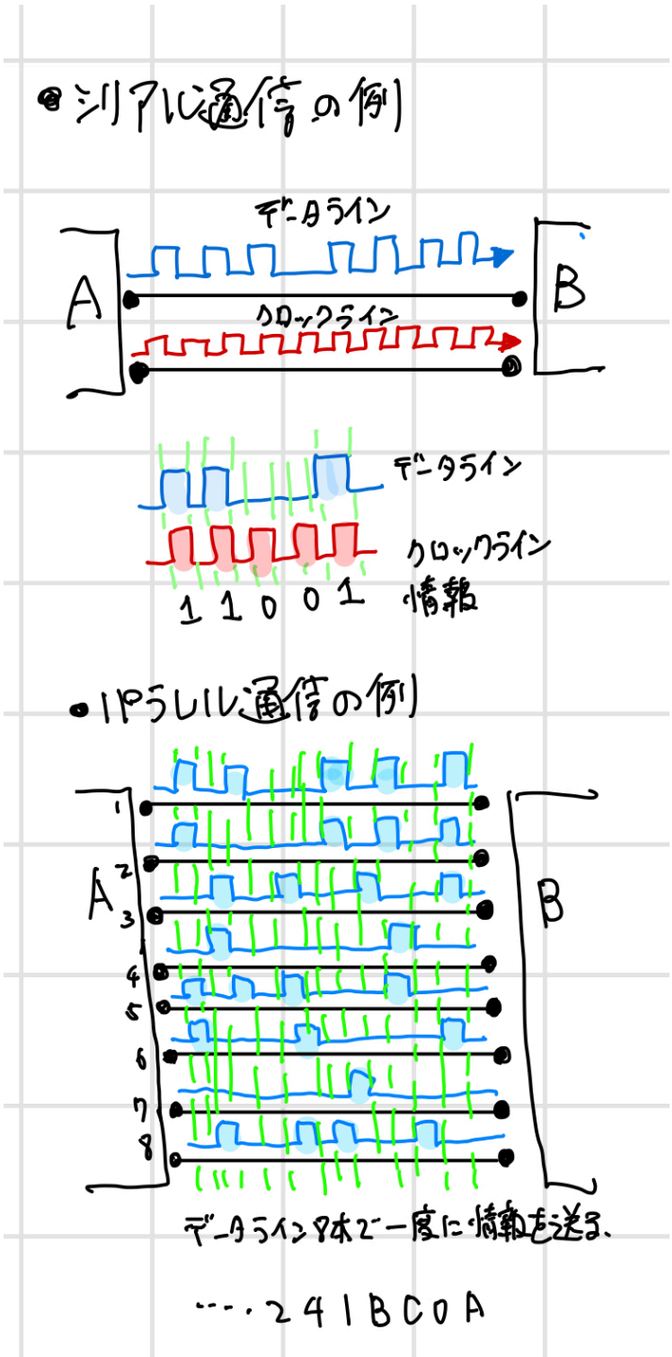
※注意点

・ハンドを開きすぎるとフィンガーの開ける限界を超えてしまうので、フィンガーがアームボディに当たらないようにする。

・ハンドを閉じた時に、閉じすぎるとフィンガーの動きがふらつくのであまり閉じないようにすること。

【シリアル通信で動くようにしてみよう】

Arduinoとパソコンは、シリアル通信ができる。
シリアル通信というのはデータのON/OFFを１つづつ送る方法だ。
（複数送る方法はパラレル通信という）



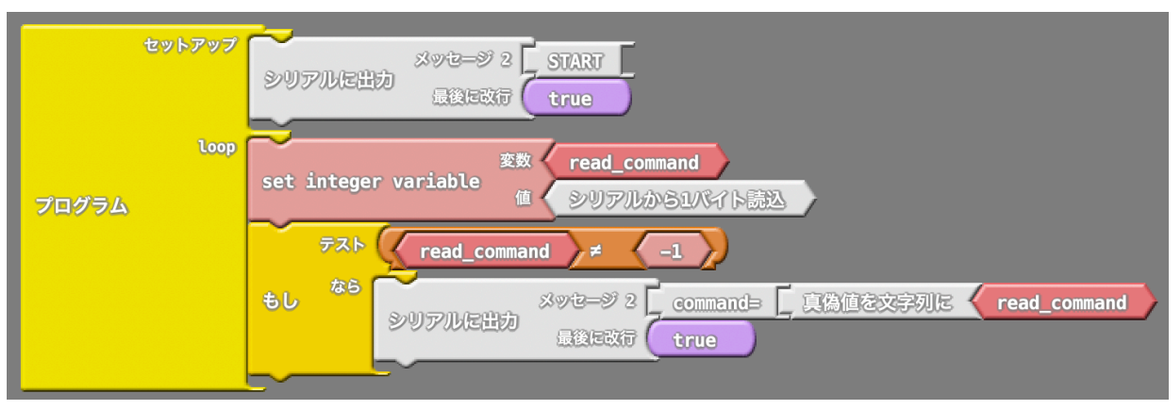
シリアル通信の場合は、クロックラインからくる信号を数えていて、例えば８つON信号をもらった時に重なっているデータラインのONの位置で情報を組み立てる。
例えば重なった時のデータラインの信号が、
01101111の組み合わせの場合、0x6Fという16進数の情報として取得できる。

これと別にパラレル通信の場合は、データラインが1~8まであって、8つの信号を同時に送れる。
つまり、
データライン１：1
データライン２：1
データライン３：1
データライン４：1
データライン５：0
データライン６：1
データライン７：1
データライン８：0
という情報を一度にまとめて送ることができる。

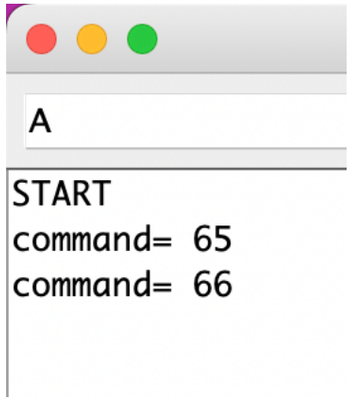
シリアルは少ない線で情報が送れる、クロックラインで何個のクロックで１つのデータを表すかを決めることで８つでも１６でも３２でも自由に設定ができるメリットがある。
パラレル一度に情報を送る分だけ速いけど、データ数を増やすためにはデータラインの配線を増やさないといけないデメリットがある。

【シリアルモニターでArduinoにデータを送ってみよう】

ArduBlockで以下のプログラムを書いてみよう。
このプログラムは、シリアルモニタからPCが送信した文字を受け取り、command=の後に文字コードを10進数で表示する。



このプログラムを書き込んで、シリアルモニタにAと入力し送信ボタンを押すとArduinoが受信した文字のコードが表示される。
Aの場合は65、Bの場合は66になるので確認してみよう。



こんな感じにArduinoに対して情報を渡せると、もらった情報に合わせてArduinoの制御をすることができるようになる。

【Arduinoに送った文字でレスキューロボットを動かそう】

レスキューロボットは、前後移動と左右旋回ができる。
パソコンのテンキーを使って

7 8 9

4 5 6

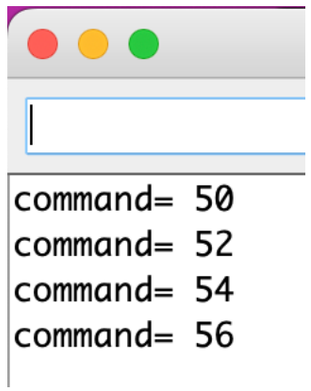
1 2 3

８を送ったら前進、２を送ったら後退、４を送ったら左旋回、６を送ったら右旋回になるようにするとレスキューロボットを操作できるようになる。

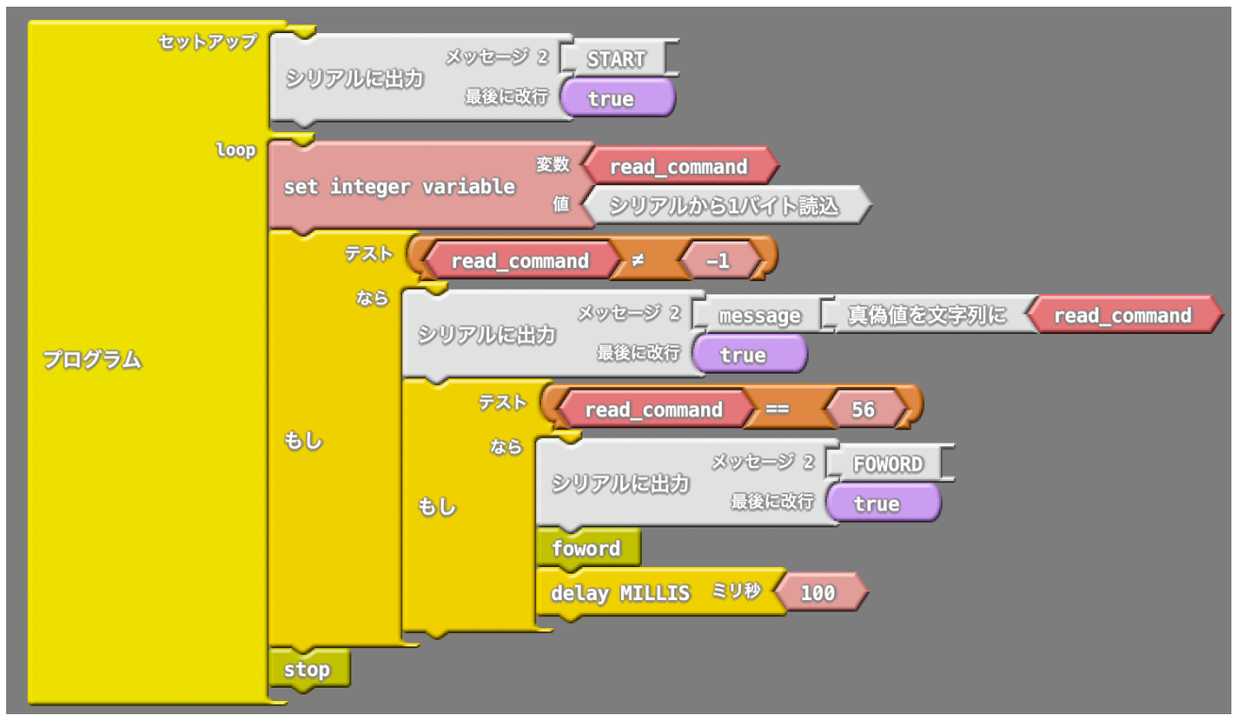
パソコンに保存した「rescue1」を呼び出して、「rescue4」という名前で保存をし直して、プログラムを変更してみよう。

シリアルモニタから2,4,6,8を送った時にArduinoがどのコードで受け取っているか調べてみた。

[2]=50
[4]=52
[6]=54
[8]=56
という結果になった。



これをもとに、8を送った時に前進状態を100ミリ秒維持するプログラムを書いてみた。



最初に「START」という文字をシリアルモニタに表示する。

次に、シリアルから１文字を読み取り、read_command変数に入れる。

そして「もし」でread_commandが-1以外（シリアルから1文字の入力があった場合、どんな文字コードを読み取ったかシリアルモニタに表示する。

さらに、read_commandが56（８）ならば、シリアルモニタに「FOWORD」と表示する。

その後に前進動作のサブルーチンfowordを呼び出し、その動作を100ミリ秒保持する（この数字の変更で移動する時間が変わる）。

最後にサブルーチンstopを呼び出してロボットの移動を止める。

このプログラムを書いて、Arduinoに書き込みシリアルモニタを開いてキーボードで8を入力し送信してみよう。

fowordで前進ができると、**2**を送信して後退もできる。

２は、文字コードでいうと**50**になる。

そして**foword**を**back**にすれば後退動作になる。
前進のプログラムを複製して以下のようにしたものを連結すれば良い。

同じ考え方で**4**の時に左旋回、６の時に右旋回のプログラムも書いてみよう。
さあ、みんなのロボットは送信する文字で自由に動くようになったかな？

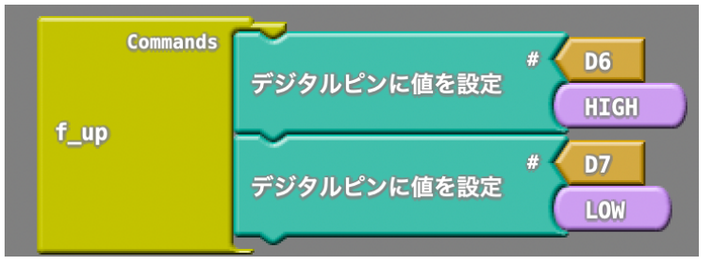
あとはフリッパーの上下とハンドの開閉を追加して操作できるようになると、このロボットをフルコントロールできるようになる。
フリッパーを下げる操作を「７」
フリッパーを上げる操作を「９」
ハンドを閉じる動作を「１」
ハンドを開く動作を「３」に割り当てると

```
command=  55
command=  57
command=  49
command=  51
```

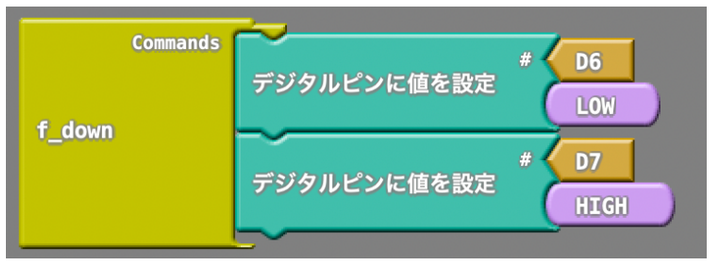
[7]=55
[9]=57
[1]=49
[3]=51
になる。

また、フリッパーを下げるサブルーチンを**f_down**、上げるサブルーチンを**f_up**、動作を止めるサブルーチンを**f_stop**、ハンドを閉じる動作を**hand_close**、ハンドを開く動作を**hand_open**として次のページにそのサブルーチンを記載しておく。

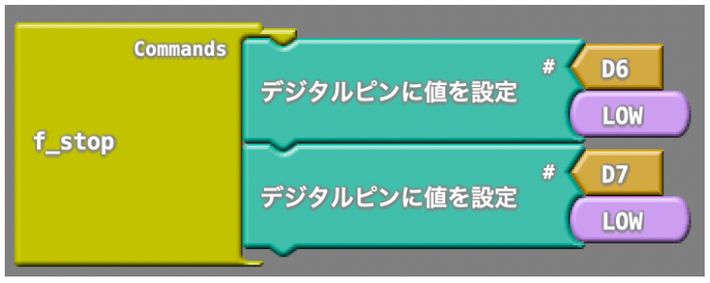
・フリッパーを上げる



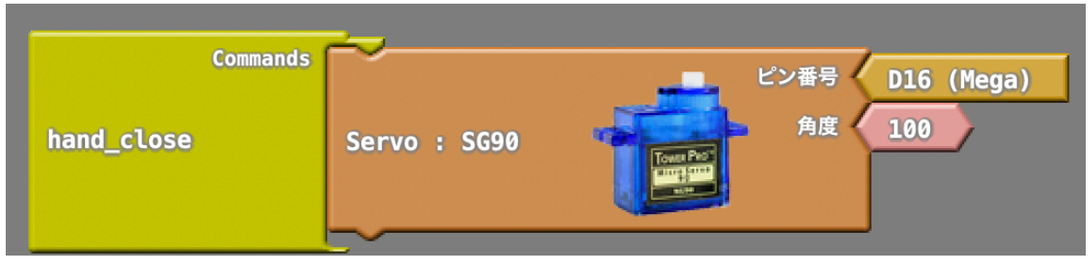
・フリッパーを下げる



・フリッパーを止める



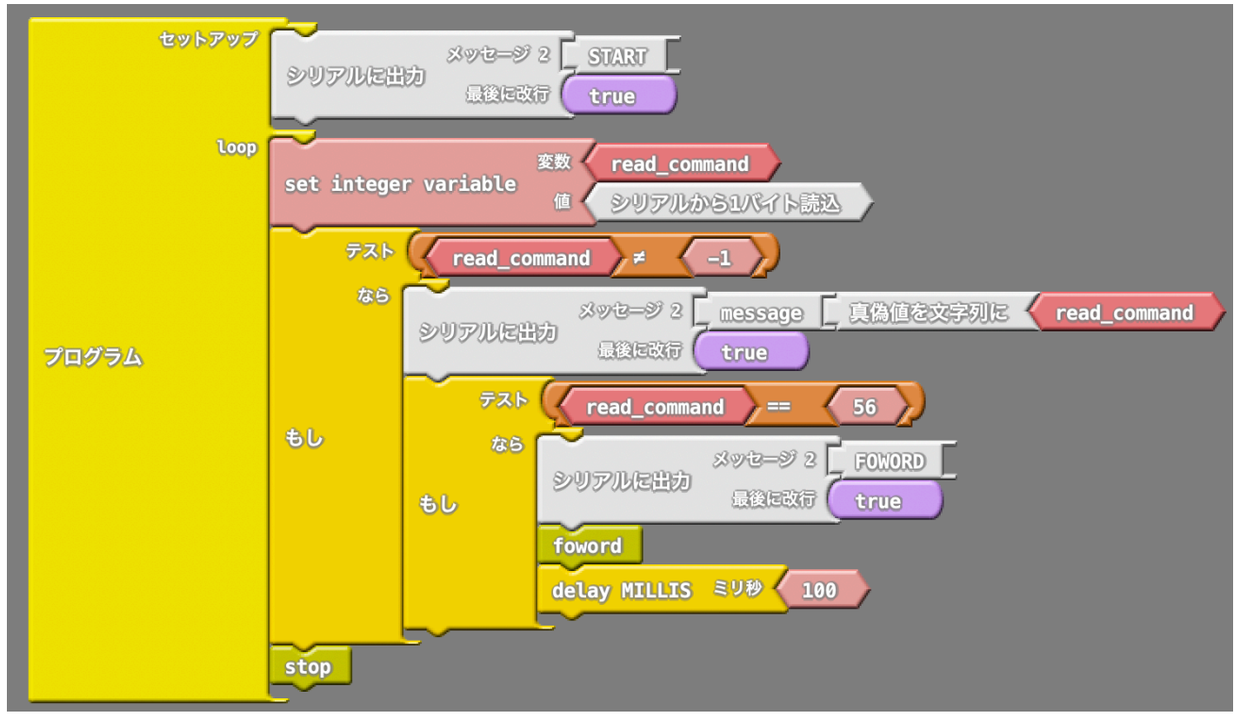
・ハンドを閉じる



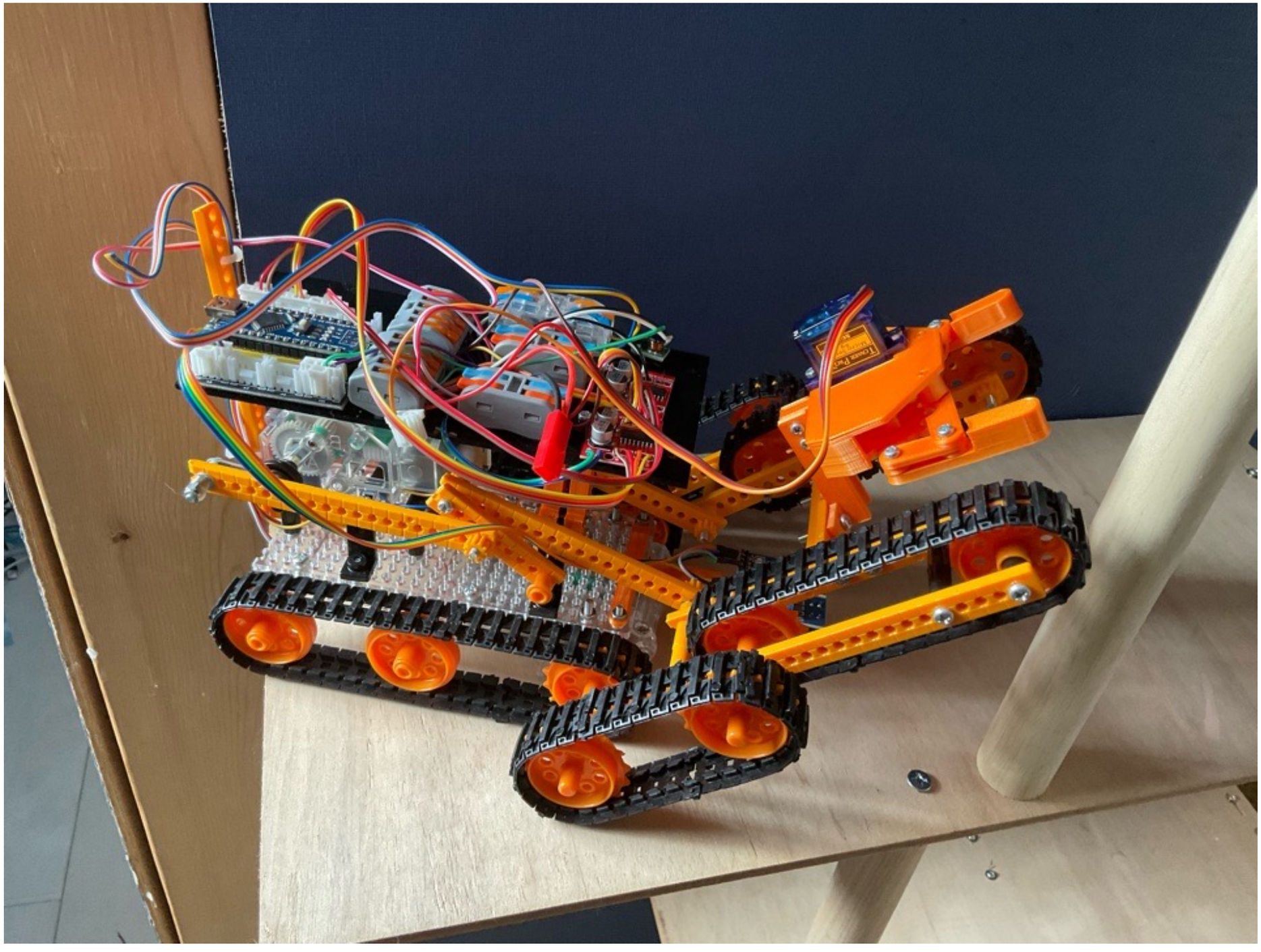
・ハンドを開く

注意点

移動の時は最後にstopを入れて動作を止めていた。なので、フリッパーを動かした時は最後にf_stopを入れておくこと。



ここまでのプログラムで、レスキューロボットをパソコンから送る文字コードで自由に操作できるようになる。
障害物を設置して走行させてみよう。

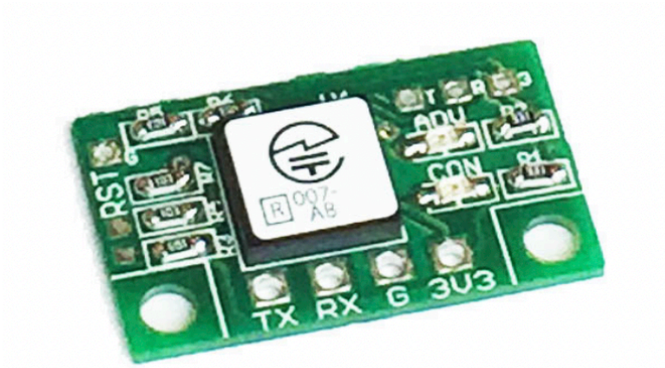


【ワイヤレスでロボットを動かしてみよう】

　今の時点のレスキューロボットは、パソコンとUSBケーブルで繋ぎシリアル通信で操作をすることができる。

　このケーブルを外し、Bluetooth規格のシリアル通信アダプタをロボットに繋ぐことで無線に置き換えることができる。

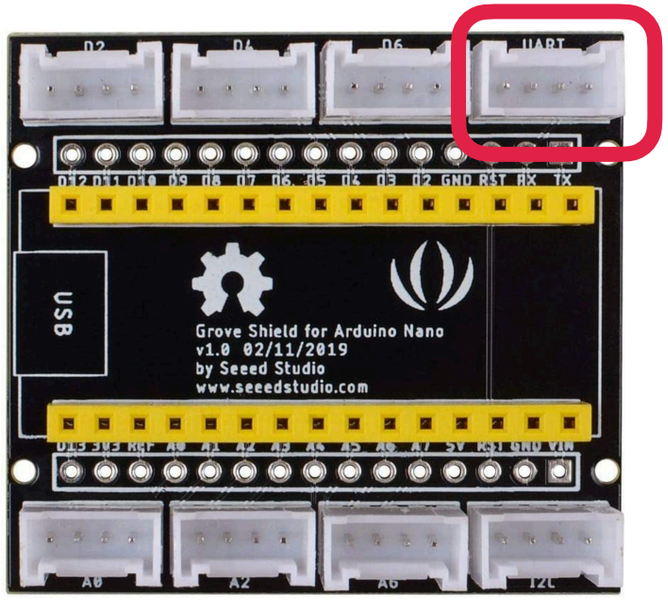
　プログラムはそのままでいいのでとても楽ちんだ。



　さらに、wifiでつながるカメラをレスキュークローラロボットに搭載すれば、カメラ画像をみながら遠隔操作できるロボットに改造することもできる。

　ただし、数十人も同時に操作をすると飛び交う電波でロボットの動作も不安定になるので今回のセミナーでは同時に操作するのは２台までとするよ。

GROVEシールドのUARTコネクタはシリアル通信専用のコネクタになる。



　ここのコネクタの機能は、パソコンと繋いだUSBケーブルによる通信と同じなので、この先にBluetooth規格のシリアル通信アダプタを接続すれば、ロボット側の準備は完了だ。

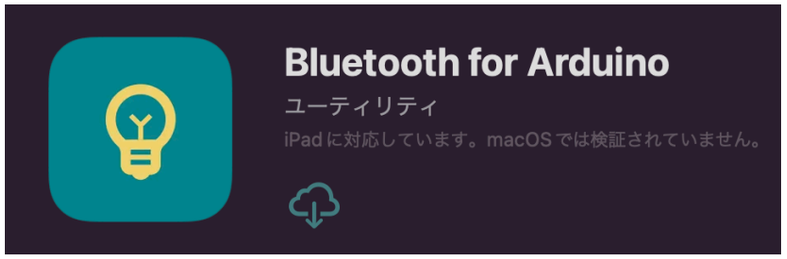
　ロボットは通信データをBluetoothで受信する機能を有するようになったが、じゃあBluetoothを使って文字コードを送信する機能はどうすればいいだろうか？

　もちろん、Arduinoを使って送信機を作ることもできるけどもっと簡単な方法がある。

　それはみんなが使っているスマホにアプリを入れることなんだ。

・iOSの場合

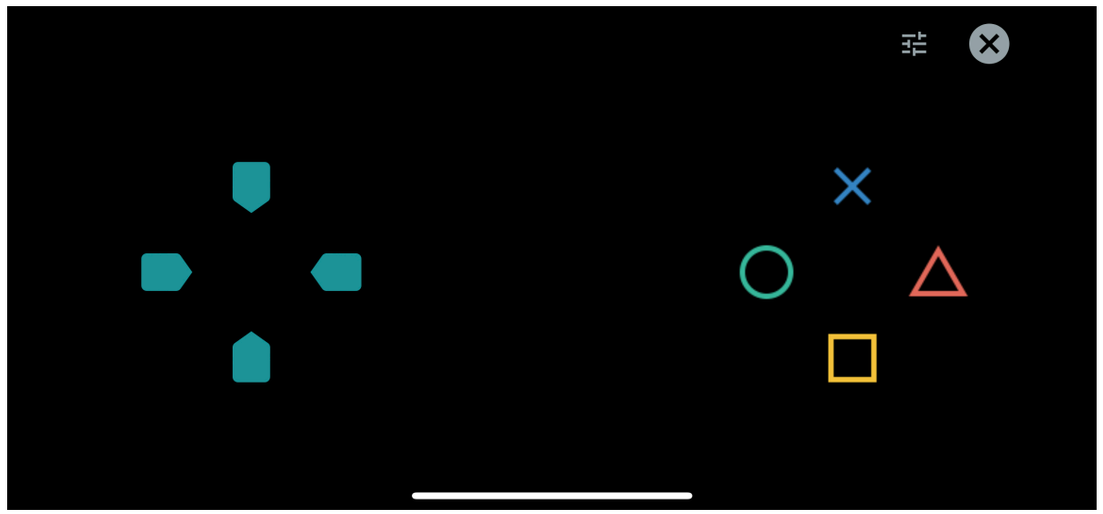
Bluetooth for Arduino



・Androidの場合

Arduino & ESP32 Bluetooth Controller App - Dabble

iOSの場合、Bluetooth for Arduinoをインストールし、有料登録をするとこのようなゲームコントローラが使えるようになる。



これで、移動キーに8,4,6,2、フリッパー上下の×に７、□に９、ハンド開閉に◯に1、△に3を割り当てればパソコンからキーを押して送った時と同じようにレスキューロボットをコントロールすることができる。

カメラに関してはwifiアクセスポイント機能の持ったペット用の市販の見守りカメラなどが利用できる。

今回はすでにこれらアプリを設定したスマホを準備しているので、遠隔操作を体験してみよう。