Studuino

ブロックプログラミング環境

入門ガイド 後編

【DC モーター サーボモーター】



目次

5. C	DC モー	・ター	. 1
5.1	. DC	モーターの制御	. 1
5	5.1.1.	Studuino 基板と DC モーターの接続	. 1
5	5.1.2.	入出力ポート情報の設定	. 2
5	5.1.3.	プログラミング	. 3
5.2	. 車の	D制御	. 6
5	5.2.1.	プログラミング	. 6
5	5.2.2.	DC モーター校正	13
5.3	. 加速	を度センサーを使用した車の制御	15
5	5.3.1.	Studuino 基板と加速度センサーの接続	16
5	5.3.2.	入出力ポート情報の設定	16
5	5.3.3.	加速度センサーの動作確認	17
5	5.3.4.	プログラミング	18
6. <u>†</u>	ナーボモ	ミーターを動かす	28
6.1	. サ-	-ボモーターの角度校正	28
6	6.1.1.	サーボモーターの駆動軸角度の調整	28
6	6.1.2.	Studuino 基板とサーボモーターの接続	29
6	6.1.3.	入出力ポート情報の設定	29
6	6.1.4.	プログラミング	31
6.2	. サ–	-ボモーター3 つでアームロボをつくる	33
6	6.2.1.	ロボットの組み立て	33
6	6.2.2.	入出力ポート情報の設定	35
6	6.2.3.	プログラミング	35

5. DC モーター

5.1. DC モーターの制御

DC モーターを制御する基本的なプログラムを通して、DC モーターの動かし方と設定を学びます。

5.1.1. Studuino 基板と DC モーターの接続

以下の手順で車を組み立てます。

(1) DC モーターに以下のようにタイヤを取り付けます。 ※左右対称に2つ作る。



(5)電池ボックスを基板台座に固定します。



5.1.2. 入出カポート情報の設定

ブロックプログラミング環境に Studuino 基板のポート情報を設定します。ブロックプログ ラミング環境のメニューバーの「編集」から「入出力設定…」を選択して、入出力設定ダ イアログボックスを開きます。入出力設定ダイアログボックスの「DC モーター」エリアの M1、M2 がチェックされていることを確認してください。

入出力設定 🛛						
DC+-9- ☑ M1 ☑		M1、 M されて	2がチェッ こいること	ック :を	ボタク ☑ A0 ☑ A1	☑ A2 ☑ A3
センサーノレ	_ ED/ブザー 米か/#~	確認し	ってくださ	50		_
□ A1	光センサー	Ŧ	🗆 A5	 光センサー		
□ A2 □ A3	光センサー 光センサー	* *	🗆 A6	光センサー 光センサー		 ▼ ▼
□ A3 光センサー ・ □ A7 光センサー ・ チェックを全て外す						

5.1.3. プログラミング

M1 に取り付けた DC モーターを制御するプログラムを作成し、テストモードで動作を確認 しながら DC モーターの設定と動きを確認します。

① 「動き」パレットから 🥰 DCモーター 🛄 の速さを 🚥 にする ブロックと 🥰 DCモーター 🕅	^{1・を 正転・} ブ
ロックと ^{CE-y- MINを使止て} ブロックをドラッグし、それぞれくっつけ	ます。
Studuino BLOCK Programming Environment	
Based on Scratch 🕀 🔡 ファイル 編集 実行 ヘルプ	
サーボモーター D9 を 60 度 DCモーター M1 の速さを C0 DCモーター M1 の速さを 100 にする DCモーター M1 を 正転 DCモーター M1 を 正転 DCモーター M1 を 停止 プザー ▼ から 60 ▼ を出力す	
 「制御」パレットから¹⁰⁰⁰ブロックを^{CE-y-MITを正転す} CE-y-MITを使止 ブロックの問にくっつけます 	ブロックと
クロックの面にくうつけます。	
Studuino BLOCK Programming Environment	_ • ×
Based on Scratch 🕀 🔚 ファイル 編集 実行 ヘルプ	
③ 単御スタート ブロックにくっつけます。	
Studuino BLOCK Programming Environment	
Based on Scratch 🕀 🔒 ファイル 編集 実行 ヘルプ	
動き 期間 調べる ; 演算 (変数) 「「「開放を」ールする 「「「開放を」ールする ① ひぼう ① 砂(行つ) ① ひ(行つ)	

以上で、M1に接続した DC モーターを回転するプログラムができました。

🛤 制御スタート	
🎻 DCモーター M1 の速さを 100 にする	DC モーターの回転速度を 100 に設定する
	DC モーターの回転を開始する
1 秒待つ	1 秒待つ
- DCモーター M1 T を 停止 T	DC モーターの回転を停止する

- ④ DC モーターの動きを確かめます。Studuino 基板と PC が USB コードで接続されていることを確認し、メニューバーの「実行」から「テストモード開始」を選択してください。
- ⑤ テストモードに移行したら、 やをクリックしてください。



M1に接続した DC モーターが回転して、1 秒後に停止します。

6 DCモーター M1 の速さを 🕻	^{100 にする} ブロックの値を50 に設定し、	🎘 をクリックしてください。
Studuino BLOCK Programming Environmer	nt	
Based on Scratch from the MIT Media Lab の 日 つア・	イル 編集 実行 ヘルプ	
動き 制御 調べる 道宜 支数 「関数 「関数をコールする 1<秒待つ すっと	NWIXタート DCモーター MI の速 を 50 5 7 8 DCモーター MI ● 匹転 DCモーター MI ● 正転 1 移待 DCモーター MI ● を 停止■	センサー・ボード [A0] ボタン 3 [A1] ボタン 3 [A2] ボタン 3 [A3] ボタン 5 [A4] 未接続 0 [A5] 未接続 0 [A6] 未接続 0 [A7] 未接続 0

DC モーターの回転速度が遅くなります。

7	🥳 DCモーター M1▼ の速さを 50 にす	^超 ブロックの値を100に戻し、	<mark>ア</mark> DCモーター M1▼ を 正転▼	を「逆転」
	に設定し、 🎘 をクリック	りしてください。		
	Studuino BLOCK Programming Environment			- • ×
	Based on Scratch from the MIT Media Lab 🕀 📙 ファイル 🗍	扁集 実行 ヘルプ		
	動き 調べる 注算 実数 ■ 開設 ■ 開設 第 日 日 日 日 一 の する し の う に 二 二 二 二 二 二 二 二 二 二 二 二 二	M御スタート DCモーター M1 の速さ 100 / 43 DCモーター M1 の速さ 100 / 43 DCモーター M1 (逆転) DCモーター M1 を 停止。	センサー・ボード [A0] ボタン 1 [A1] ボタン 1 [A2] ボタン 1 [A3] ボタン 1 [A4] 未接続 0 [A5] 未接続 0 [A6] 未接続 0 [A7] 未接続 0	

DC モーターの回転方向が逆になります。

8		ゴブロックの値を「正転」に戻し、 🧖 👓 🤜	- ^{MITを使止て} ブロック
	を「解放」に設定し	J、 <mark>や</mark> をクリックしてください。	
	Studuino BLOCK Programming Environ	nent	- • ×
	Based on Scratch from the MIT Media Lab	ファイル 編集 実行 ヘルプ	
	(動き) 制御 (演算) (演算) 変数 (調べる) 演算 変数 (開める) (演算) (開数を□-ルする) (引参) (引参) (引参) (引参) (引参) (引参)	Al御スタート Cモーター M1 の速さを 100 にする Cモーター M1 正転 DCモーター M1 正転 DCモーター M1 正転 DCモーター M1 (正転 DCモーター M1 ())	センサー・ボード [A0] ボタン [A1] ボタン [A3] ボタン [A3] ボタン [A3] ボタン [A4] 未接続 [A5] 未接続 [A6] 未接続 [A7] 未接続

DC モーターが惰性で停止します。

※DC モーターの回転速度は 100 が最大となります。100 以上の数値を入力してもそれ以上速度は上がりません。

※DC モーターの回転速度を一定以上低く設定すると、DC モーターが回転しません。

設定数値は DC モーターへ流す電流の量を表しているため、DC モーターを動かすためには、DC モーター が回転するために必要な最低限の電流を流す必要があります。

5.2. 車の制御

DC モーター2つを使用して、車を前進、後退、回転するプログラムを作成します。

5.2.1. プログラミング

最初に車の前進プログラムを作成します。

「制御」パレットから
 「勘御」
 ブロックをドラッグし、ブロックの▼をクリック
 し、「新規…」を選択します。ここでは表示されたダイアログボックスに関数名「forward」
 を設定し、OK をクリックします。



② 「動き」パレットから^{200 DCE-ター MINの速を 100 にする}ブロックと^{20 DCE-ター MIN を 正転 ブ ロックと^{200 DCE-ター MIN を 停止 ブ ブロックをドラッグし、^{100 関数}ブロックとそれ ぞれをくっつけます。}}



③ 「制御」パレットから 💷 🚧 ブロックを 🌠 🔤 🖽 ブロックと





以上で、車を前進させるプログラムができました。



- ・ ●の動きを確かめます。Studuino 基板と PC が USB コードで接続されていることを確認し、メニューバーの「実行」から「テストモード開始」を選択してください。
- テストモードに移行したら、 たっとうしょうしてください。

Studuino BLOCK Programming E	invironment	<u>ل</u> م	
Based on Scratch from the MIT Media Lab	📙 ファイル 編集 実行 ヘルプ	99	
動き 制御 し 、 演算 、 変数 forward 一間数 forward 一間数 forward 一間数 forward 一間数をコールする 1 秒待う	Milit2/2-ト forward 開設をコールする	forward 開設 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	センサー・ボード [A0] ボタン 1 [A1] ボタン 1 [A2] ボタン 1 [A3] ボタン 1 [A4] 未接続 0 [A5] 未接続 0 [A6] 未接続 0 [A7] 未接続 0
すっと 10 回線り返す らし 二 なら、すっと	プログラムがテストモ ードで実行されます		

車が前進して、1秒後に停止します。

⑧ ●をクリックしてテストモードを終了させてください。

ど Studuino BLOCK Programming Environment						
Based on Scratch from the MIT Media Lab	ファイル 編集 実行 ヘルプ	ク	リック			
動き 制御 同べる 注意 定数 forward 関数	<mark>▲ 制御スタート</mark> forward▼ 関数をコールする	forward NBM ● DCモーター M1 の速さを 100 にする ● DCモーター M2 の速さを 100 にする ● DCモーター M2 の速さを 100 にする	センサー・ボード [A0] ボタン 1 [A1] ボタン 1 [A2] ボタン 1 [A3] ボタン 1 [A3] ボタン 1 [A3] ボタン 1 [A3] ボタン 1 [A3] ボタン 1			

次に、車の後退プログラムを作成します。



 ・ 複製したブロックの

 「^{forward™ 関数}
 の▼をクリックし、「新規…」を選択します。表示
 されたダイアログボックスに関数名「backward」を設定し、OK をクリックします。



9



 Iorward 1082
 前規...
 DCモーター M2 0/252 100 にする

 forward 1082
 DCモーター M1 を 正転

 1064
 DCモーター M2 を (PL)

 1064
 DCモーター M2 を (PL)

 1064
 DCモーター M1 の使きを (DO) にする

 1070
 DEE

 1070
 DE

 1070

以上で、車を後退させるプログラムができました。

手順の⑥、⑦を行い、車が後退して、1秒後に停止することを確認してください。

次に、車を右に回転するプログラムを作成します。

③ 手順の⑨、⑩を行い、rotationRight 関数を作成してください。



以上で、車を右に回転させるプログラムができました。

手順の⑥、⑦を行い、車が右に回転して、1秒後に停止することを確認してください。

次に、車を左に回転するプログラムを作成します。

19 手順の⑨、⑩を行い、rotationLeft 関数を作成してください。



以上で、車を左に回転させるプログラムができました。

手順の⑥、⑦を行い、車が左に回転して、1秒後に停止することを確認してください。

5.2.2. DC モーター校正

DC モーターは個体差により同じ設定でも回転速度のズレが生じます。DC モーター 2 つで 移動する車を作った時、前進の際にまっすぐ走らない場合、DC モーター校正によりこのズ レを補正する必要があります。

① Studuino 基板の DC モーター用コネクターに DC モーターを接続します。

DC モーターを2つ並べ、回転速度の違いを確認できるようにします。



② 入出力設定を行います。

入出力設定画面で DC モーターの M1、M2 にチェックを入れます。

入出力設定				-	- 6		×
DOT-4	_	サーボモ	力			ボタン	
₩1 2	M2	□ D2	D D4	🗆 D7	🗆 D8	A0 A2	
		🗆 D9	🗖 D10	🗖 D11	🗖 D12	🗆 A1 🗖 A3	
センサー/	LED/	ブザー					
🗆 A0	光セン	/サー	-	🗆 A4	LED	-	
🗆 A1	光セン	/サー	-	🗖 A5	ブザー	-	
🗆 A2	光セン	/サー	-	🗖 A6	光センサー	-	
🗆 A3	光セン	/サー	-	🗆 A7	光センサー	-	
チェックを全て外す OK キャンセル							

③ メニューバーの「編集」から「モーター校正…」を選択します。

基板に USB ケーブルが接続されていることを確認してください。 Studuino BLOCK Programming Environment Robotist ファイル 編集 実行 ヘルプ クリック 削除の取り消し ステップ実行を開始 制御 動き ステップ実行を設定.. 🔎 制行 調べる 演算 Arduino言語に変換 変数 モーター校正.. 入出力設定...

モーター校正を選択すると、下図のようなウインドウが開きます。ここでは、下側の DC モーター校正を使用します。

④ 回転ボタンを押し、モーターを回転させます。

モーター校正	
サーボモーター校正 リセット D2 ① 1 度 D9 ① 1 度 D4 ② 1 度 D10 ② 1 度 D7 ③ 1 度 D11 ③ 1 度 D8 ③ 1 度 D12 ③ 1 度	回転ボタンをクリックすると、2 つのモーターが最大速度で 回転し始めます。
	クリック
M1 255	
OK	

⑤ スライダーで速度の調整を行います。

モーター校正	
サーポモーター校正 リセット D2 01 度 D9 01 度 D4 01 度 D10 01 度 D7 01 度 D11 01 度 D8 01 度 D12 01 度	回転速度の速い方のスライダーを調整し、もう一方 と回転がそろうようにします。 完了したら、停止ボタンをクリックし、回転を止め ます。
	スライダーで速度を調整

⑥ OK ボタンをクリックし、校正を完了します。



5.3. 加速度センサーを使用した車の制御

5.2 車の制御で作成したプログラムを使って、加速度センサーを利用した車の制御プログラムを作成します。

加速度センサーとは、加速度(速度が一定時間あたりにしてどれだけ変化したか)を計測 するセンサーであり、X,Y,Zのそれぞれの方向の加速度を読み取ります。



また、加速度セ**ン**サーは下図のようにセンサーの向きを変えると、それぞれの方向の値が 変化します。



加速度センサーを固定した状態でも加速度が0にならない方向があるのは、重力加速度[※] を検出しているためです。この性質を利用して、地面に対してどのくらい傾いているかを 知ることもできます。

※重力加速度とは重力の向きに常に働く加速度です。

5.3.1. Studuino 基板と加速度センサーの接続

Studuino 基板のセンサー/LED/ブザー用コネクターの A4、A5 に加速度センサーを取り付けます。



上側に灰色のコードがくるように。

5.3.2. 入出カポート情報の設定

ブロックプログラミング環境に Studuino 基板のポート情報を設定します。ブロックプログ ラミング環境のメニューバーの「編集」から「入出力設定...」を選択して、入出力設定ダ イアログボックスを開きます。入出力設定ダイアログボックスの「センサー/LED/ブザー」 エリアの A4、A5 をチェックして加速度センサーを選択して下さい。

Л	出力設定									
	DCモーター 2 M1 2	- M2	サーボモ □ D2 ☑ D9	- ター □ D4 ☑ D10	□ D7 ☑ D11	□ D8 □ D12	ボタン ☑ A0 ☑ A1	☑ A2 ☑ A3		
	センサーノレ	.ED/	ブザー							
	🗆 A0	光む	ノサー	*	🗹 A4	加速度セン	,サー	•		A4、A5 をチェックし
	🗆 A1	光む	ノサー	-	🗹 A5	加速度セン	,サー	-		加速度ヤンサーを
	🗆 A2	光む	ノサー	+	🗖 A6	光センサー		-		選択してください
	🗆 A3	光む	ノサー	-	🗆 A7	光センサー		-		
	チェックを全てく	かす				ОК	*+	ァンセル]	

5.3.3. 加速度センサーの動作確認

加速度センサーの値をチェックします。Studuino 基板と PC を USB コードで接続し、ブロックプログラミング環境のメニューバーの「実行」から「テストモード開始」を選択して ください。テストモードが開始されるとセンサー・ボードが表示され、「[A4/A5] 加速度センサー」が表示されます。

・加速度センサーの値は、0~100で表します。

・X 方向(前ページの図参照)を水平に置いた場合、加速度センサー(X)の値は 50 になります。

・X 方向が上向きになるように傾けるにつれて、加速度センサー(X)の値は増え、垂直になったとき、75 となります。

・反対に、X 方向が下向きになるように傾けるにつれて、加速度センサー(X)の値は減り、 垂直になったとき、25 となります。

・重力加速度以上の加速度を与えたとき(加速度センサーを X 方向に動かしたとき)は、 さらに値の変化は大きくなります。

※Y、Z方向についても同じです。



確認ができたら、メニューバーの「実行」から「テストモード終了」を選択し、テストモードを終了してください。

5.3.4. プログラミング

5.2.1 で作成した車を前進・後退・回転するプログラムを使用します。

C- SPE BLOCK			
Based on Scratch from the MIT Media Lab 🕀 🔜 ファイル 編集 実行 ヘルプ			
		dc	/ • •
変数 御師以夕一下	forward BB数	rotationRight 段微	
	CEーター M1 の速さを 100 にする	DCE-9- M1 00256 100 (273	
backward T 開設	CCE-9- M2- M2→ Origization (100) (273		
backward 開設をコールする			
0 9/20			
	DCモーター M1 を停止す		
3 22	□Cモーター M2 を 停止 ▼		
	backward ▼ 開設	rotationLeft 图数	
60 286	🤣 DCモーター M1 の速さを 100 にする	😴 DCモーター M1〒 の速さを 100 にする	
FL 26	🚀 DCモーター M2 の速さを 100 にする	🛷 DCモーター M2 🗸 の速さを (100) にする	
TADAX		🚀 DCモーター M1 友 正転 🗍	
		CCモーター M2 を 運転 -	
		1 秒待つ	
- よで線の返す		<u> </u>	
	CM DOCTOR NOT S INT	CE-9- M2 & 191	

① 1000 ブロックの設定を「0.1」に変更します。



2 🧖	^{DCモーター M1▼ を ^{停止▼} ブロック}	フと 🧖 DCモーター M2 を 修	「フロックの設定を	を「解放」に変
更し	[,] ます。			
dc- SPE BLOCK Based on Scr from the MIT Med	tch 伊 🖥 ファイル 編集 実行 ヘルプ			
Bas B BUCS 3 Baskward Bast Baskward Bast Baskward Bast Baskward Bast Baskward Bast		forward # GBQ Image: Control Image: C	Image: Constraint of the section of the	
60 26 702 60 26 60 26 60 26 60 26 76 70 70 70 70 70 70 70 70 70 70 70 70 70		Deckward Digit Oct-クー MIL の速たを 100 にする Oct-クー MIL の速たを 100 にする Oct-クー MIL の速たを 100 にする Oct-クー MIL を 逆圧	RetionLet* 158 DCモーター M1_00ほなを 100 にする DCモーター M2_00ほなを 100 にする DCモーター M2_1 を 近数	
3 forw	ard 関数の 🧟 DCモーター M1 🛛	の速さを 🚥 にする ブロック 速さを 💷 にする	7と <mark>愛</mark> DCモーター MIN を I	転 ブロック
を <u>切</u> す	り離し、 🌱 ロᡄモーター 🔤 🗤	^{速さを 100 にする} をブロック	ウパレットにドラック	^ブ して削除しま
9 o				
Pacedon Str Hed Shift Hed Shift Hed Shift Shift	tth: 中日 ファイル 編集 実行 ヘルゴ コー コー コー コー コー コー コー コー コー コー	 接続を Converse gut Dot-9- Mill ond 1 100 LTS 	久す	

 Dodeword
 005

 Dodeword
 005
 </

 Optimute
 Rot

 ジ
 DCモーク・用い、のまたを1000 にする

 ジ
 DCモーク・用い、のまたを1000 にする

 ジ
 DCモーク・用い、のまたを1000 にする

 ジ
 DCモーク・用い、0100 ほうち

 ジ
 DCモーク・用い、0100 ほうち

ak_____ _____

6U 🔿 16 ~~~ 2/40/11/4 まで待つ - まで線0返す

19

(4) forward -	蹴 ブロックと	DCモーター M1▼ を 正転▼ I	ブロックをくっつけます。	
C dc- SPE BLOCK				
Based on Scratch from the MIT Media Lab	冒 ファイル 編集 実行 ヘルプ			
(abs (alia)			dc	/ * •
		forward T 問版	rotationRight _ 問故	
2.17/			🧭 ロCモーター 🕅 の速さを 💷 にする	
backward - 関鉄		- DCモーター M2 を 正転 -	🛷 DCモーター M2 の迷さを 100 にする	
backward = 関数をコールする			CT-ター MI を 逆転 -	
			CCモーター MI を 解放 -	
1			○ DCモーター M2 支 解放 ●	
10 回線り返す				
				
もし <u>ねら、ずっと</u>				
			rotationLeft	
6U 200			CE-9- M2- の速さき 100 にする のCE-9- M1-1を 下転、	
Chitalia		🚀 DCモーター M2 🛛 を 逆転 🗸		
100 arriso			0.1 8(8)	
まで織り返す			CE-ター M1 を 解放す	
				-

- ⑤ 他の関数も手順③、④と同様の手順で^{2001年の通知を1001にする}

ac- SPE BLOCK				
Based on Scratch from the MIT Media Lab	🔚 ファイル 編集 実行 ヘルプ			
Bit Bit Bit Bit Bit Bit Batward Bit Batward Bit	Mara-F	Torward Bbt ● DCE-D- ML % 正転 ● DCE-D- ML % 正転 ● DCE-D- ML % ● DCE-D- ML %	reasonnight EED マ OCE-9- ML E 送話 02 GE マ OCE-9- ML E 送話 03 GMO マ OCE-9- ML E 新政 02 GMO マ OCE-9- ML E 新政 02 GMO	dc 🍋
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1		backward 1000 ● DCE-9- M2 を 2005 ● DCE-9- M2 を 2005 ● DCE-9- M2 を 用取 ● DCE-9- M2 を 用取 ● DCE-9- M2 を 用取	PotalionLeft 単数	

6	「動き」バレ	ットから 🖉 ロᡄモーダ	- M1▼ の速さを 100 にする	ブロックを2つ	ドラッグし、
	AM御スタート	ブロックにくっつけ	、2 番目のブロッ	クの設定を「M2」	に設定します。

Based on Scratch from the MIT Media Lab 🕀 📄 ファイル 編集 実行 ヘルプ			
	forward 開設	rotationRight 問題	dc 🎘 🔴
レンジャング DCモーター M1 の速さを 100 にする	🛷 DCモーター M1× を 正転×	🚀 DCモーター M1 を 逆転 🔹	
🚳 サーボモーター D9 を 90 度	🚀 DCモーター M2▼ を 正転▼	🚀 DCモーター M2、を 正転、	
- DCモーター M1 の速さ 10	0.1 秒待つ	0.1 秒待つ	
ドラッグ&ドロップ	- DCモーター M1 * を 解放 *	Cモーター M1▼ を 解放▼	
CUCTOP WA & LEAR		CCモーター M2 を 解放 -	
● ブザー ● から 60● を出力す	backward 型数	rotationLeft 関数	
【▶ ブザー ▼ の出力を停止する	🦪 DCモーター M1 🛛 を 逆転 🗸	🚀 DCモーター M1 🛛 を 正転 🗸	
	🧬 DCモーター M2 🛛 を 逆転 🗸	🚀 DCモーター M2 🛛 を 逆転 🗸	
	0.1 秒待つ	0.1 秒待つ	
	Cモーター M1 を解放 -	CE-ター M1 を解放。	
	● DCモーター M2 を 解放 ▼	OCモーター M2 を 解放▼	

⑦ 「制御」パレットから ブロックをくっつけます。

📴 dc- SPE BLOCK			
Based on Scratch of Based on Scratch from the MIT Media Lab 🕀 📙 ファイル 編集 実行 ヘルプ			
H/b H/B EX-3 IXII Dackward IXIII backward IXIII backward IXIIII backward IXIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII	Tormand Gab ○ DCE-3- ML & 正程 ○ DCE-3- MZ & 正程 ○ DCE-3- MZ & 振放	retationRight get の DCモーシー M1 を 道弦 の DCモーシー M2 を 道弦 の DCモーシー M1 を 解放 の DCモーシー M1 を 解放 の DCモーシー M2 を 解放	dc 🧖 🍋
** ** ** ** ** ** ** ** ** **	Backward FBS1 マン DCモーター ML を EBE マン DCモーター MZ を EBE 03 MAO MAO マン DCモーター ML を MAC マン DCモーター ML を MAC マン DCモーター ML を MAC マン DCモーター ML を MAC	rotationLeft pgg の の たころ- ML を 正転 の たころ- MZ を 道転 の の たーター ML を 新設 の してーター ML を 新設	

⑧ 「演算」パレットから () ブロックをくっつけます。

ac- SPE BLOCK			
Based on Scratch from the MIT Media Lab 🕀 🔚 ファイル 編集 実行 ヘルプ			
	Forward post G Oct-クー M1 を 正程 G Dct-クー M2 を 正程 D1 1040 Dct-クー M1 を 前放 G Dct-クー M1 を 前放 Oct-クー M2 を 前放 Dct-クー M2 を 前放	CotationRight: MBE DCE-2- M1 を 逆転 DCE-2- M2 を 逆転 D2 UE-2- M2 を 逆転 DCE-2- M1 を 約33 DCE-2- M1 を 約33 DCE-2- M2 を 約33	dc 🔊
 ドラッグ&ドロップ たう、 たう、 たけの、 たけの、 そのではかの、 そのではかの	backward ppp の た た た た し た - か し た - か し た - か - か し た - か - か - か - か - か - か - か - か	rotationLeft 「REE の DCモーター M1 を 正転で の DCモーター M2 を 正転で 0.3 砂柿つ の DCモーター M1 を 新設で の DCモーター M1 を 新設で	

 ⑨ 「調べる」パレットから(加速度セッサー図=00億)ブロックを ● ブロックの左辺にくっつけ、 右辺に 40 を設定します。



⑩ 「制御」パレットの「「^{関数をコールする}ブロックを

■ブロックの開

いているところに入れ、forward 関数を選択します。

ac- SPE BLOCK		
Based on Scratch from the MIT Media Lab 🕀 🖥 ファイル 編集 実行	トルプ	
	Di送を (200)にする の 送をを (200)にする の 送をを (200)にする の 送 な (200)にする の 切 く (20) から の 切 く (20) から の 切 く (20) から の の く (20) から の の の の の く (20) から の の の の の く (20) から の	CC 一〇 FORMURAL NEW CO CO
 もし (加速度センサー×ーの値)< (1) 	 	7し、複製を選択します。
ac- SPE BLOCK		
Based on Scratch from the MIT Media Lab 🕀 🖥 ファイル 編集 実行		
Backword BBR Backword BBR <td>0.122 € (100) (173) (1173) 0.122 € (100) (174) (1173) 0.122 € (100) (174) (1173) 0.122 € (100) (174) (1173) 0.122 € (100) (174) (1173) 0.122 € (100) (174) (1173) 0.122 € (100) (174) (1173) 0.122 € (100) (174) (1173) 0.123 € (100) (174) (1173) 0.124 € (100) (174) (1174) (1174) 0.124 € (1174) (1174) (1174) 0.124 € (1174) (1174) (1174) 0.124 € (1174) (1174) (1174) 0.124 € (1174) (1174) (1174) 0.124 € (1174) (1174) (1174) 0.124 € (1174) (1174) (1174) (1174) 0.124 € (1174) (1174) (1174) (1174) (1174) 0.124 € (1174) (1174</td> <td>сс № становления № сред-у-М1 € 228 № сред-у-М2 € 228 № сред-у-М2 € 268 №</td>	0.122 € (100) (173) (1173) 0.122 € (100) (174) (1173) 0.122 € (100) (174) (1173) 0.122 € (100) (174) (1173) 0.122 € (100) (174) (1173) 0.122 € (100) (174) (1173) 0.122 € (100) (174) (1173) 0.122 € (100) (174) (1173) 0.123 € (100) (174) (1173) 0.124 € (100) (174) (1174) (1174) 0.124 € (1174) (1174) (1174) 0.124 € (1174) (1174) (1174) 0.124 € (1174) (1174) (1174) 0.124 € (1174) (1174) (1174) 0.124 € (1174) (1174) (1174) 0.124 € (1174) (1174) (1174) (1174) 0.124 € (1174) (1174) (1174) (1174) (1174) 0.124 € (1174) (1174	сс № становления № сред-у-М1 € 228 № сред-у-М2 € 228 № сред-у-М2 € 268 №

② 複製したブロックを複製元のブロックにくっつけます。



③ ▲ ブロック < 記号上で右クリックし、>を選択します。





もし (加速度t	2ンサー<u>×ー</u>0値 < 40)なら ブロックで右クリックし、複製を選択します	- o
dc- SPE BLOCK		
Based on Scratch from the MIT Media Lab	冒 ファイル 編集 実行 ヘルプ	
5/5 100 19/5 190 10/5 190	And	

16 複製したブロックを複製元のブロックにくっつけます。







以上で、加速度センサーを使って車を制御するプログラムができました。





- ⑧ テストモードに移行したら、 ▶をクリックしてください。



加速度センサーの傾きに反応して車が動きます。

19 テストモードで動作が確認できたら、メニューバーの「実行」から「プログラム作成・
 転送」を選択してください。

※テストモードでは、一つ一つのプログラムの処理に時間がかかるため、車はスムーズに動きません。プログラム転送後はスムーズに動くようになります。

6. サーボモーターを動かす

ここでは、サーボモーターを使用する基本的なプログラムの作成を通して、サーボモータ ーの動かし方、変数ブロックと演算ブロックの使い方を説明します。

6.1. サーボモーターの角度校正

サーボモーターは個体差により同じ角度設定にしても数度のズレが生じます。サーボモー ター角度校正により、このズレを補正する必要があります。

6.1.1. サーボモーターの駆動軸角度の調整

サーボモーターを取り付ける前に、サーボモーターの駆動軸が正常に取り付けられている か、以下の方法で確認してください。

駆動軸を基準位置の状態から、左右に止まる位置まで回転させたときのボディとの角度が、左右で大きく異なる場合は、駆動軸のブロック部分がスリップして正常な位置からずれてしまっています。



これは駆動軸に大きな力がかかった時、内部のギヤか破損しないように、一定の力でフロック部分の みがスリップして回転することで、内部のギヤの破損を防ぐ仕組みになっているため起こります。 駆動軸のブロック部分をずれている方向と反対方向に手で回転させると、カチッと音が鳴りズレが元 に戻ります。



※むやみにスリップさせないでください。 サーボモーターの劣化、破損の原因となります。
※微小な角度のズレは、後のソフトウェアでの 角度校正で補正できます。

6.1.2. Studuino 基板とサーボモーターの接続

Studuino 基板のサーボモーター用コネクターの D9 にサーボモーターを接続します。



6.1.3. 入出カポート情報の設定

ブロックプログラミング環境に Studuino 基板のポート情報を設定します。ブロックプログ ラミング環境のメニューバーの「編集」から「入出力設定…」を選択して、入出力設定ダ イアログボックスを開きます。入出力設定ダイアログボックスの「サーボモーター」エリ アで、D9 のみチェックしてください。

入出力設定 🗾							
DCモーター	サーボモ	-9-			ボタン		
🗹 M1 🔽 M2	🗆 D2	🗆 D4	🗆 D7	🗆 D8	🗹 A0	🗹 A2	
	☑ D9	🗖 D10	🗆 D11	🗖 D12	🗹 A1	🗹 A3	
eJy-7LEU			_				
			🗆 A4	光センサー		*	
D9 のみチェ	.ッ ク	-	🗖 A5	光センサー		+	
してくださ	きい	-	🗖 A6	光センサー		-	
		-	🗆 A7	光センサー		-	
チェックを全て外す				ОК	÷+	ッシセル]

1 メニューバーの「編集」を選択し、表示されたウインドウから「サーボモーター角度校正」を選択します。

基板に USB ケーブルが接続されていることを確認してください。

🔚 Studuino BLOCK Programmi	ng Environment	クリック	クリック	
Based on Scratch from the MIT Media Lab 動き 調べる 演算 変数	⊕ 🗄 ファイル	編集 削除の取り消し ステップ実行を開始 ステップ実行を設定 Arduino言語に変換 サーボモーター角度校正 入出力設定		

サーボモーター角度校正を選択すると、接続されているサーボモーターが全て 90 度の 位置で固定されると同時に、自動的にテストモードとなり、下図のようなウインドウが 開きます。

※基板に電池ボックスをつなぎ、電源を ON にした状態で確認してください。



駆動軸とボディが平行

サーボ	サーボモーター角度校正				
			リセット		
D2	0 🗧 度	D9) 🗧 度		
D4 [0 🔶 度	D10	0 🔶 度		
D7 [0 🌩 度	D11	0 🗧 度		
D8	0 🔷 度	D12	0 🗧 度		
	ОК		Cancel		

② この時、90 度の位置から少しずれている場合は、サーボモーター角度校正画面に数値 を入力することで、サーボモーターの角度を微調整することができます。90 度の位置 になるように数値を入力してください。



ーボモーター角度校正	
リセット	
D2 0 束 度 D9 3 🗣 度	
D4 0 夏 D10 0 夏 クリ	ック
D7 0 🖢 度 D11 0 🖤 度	
D8 0 🖢 度 D12 0 ≜ 度	
OK Cancel	

駆動軸とボディがずれている

※校正の際に取り付けたコネクターに別のサーボモーターに付け替えた場合は、再度サ ーボモーター角度校正が必要です。

角度校正を終えたサーボモーターには、サーボモーター用コネクターの番号と同じ番号 のシールを貼りつけて、識別できるようにしてください。



6.1.4. プログラミング

① 「動き」パレットから^{(サーボモーター D9] を 90 度にする}ブロックを^{制御スタート}ブロックに

くっつけます。



② サーボークー D9 を 30 度にする ブロックの「90」に「0」を入力します。テストモードで動き を確認します。Studuino と PC が USB コードで接続されていることを確認し、メニュ ーバーの「実行」から「テストモード開始」を選択してください。

③ テストモードに移行したら、 🎘 をクリックしてください。

start- SPE BLOCK		
Based on Scratch from the MIT Media Lab	ファイル 編集 実行 ヘルプ	
動き 制御 調べる 演算 変数 (演算) ひてモーター D9 を 90 度 ひてモーター M1 の速さを 10 ひてモーター M1 を 正転二 ロン DCモーター M1 を 使止 ロン DCモーター M1 を 使止 ロン DCモーター M1 を (登上) ロン フザー から 60 を出力す	またまでは、またします。 またまでは、またします。 またまでは、またします。 またまでは、またします。 	Start センサー・ボード [A0] ボタン [A1] ボタン [A2] ボタン [A3] ボタン [A4] 未接続 [A5] 未接続 [A6] 未接続 [A7] 未接続

- サーボモーターが0度に動くことを確認します。
- ④ 同様に サーボモーター DPT を DD 度に すう ブロックの「0」に「180」を入力し、 たをクリックしてサーボモーターが 180度に動くことを確認します。



6.2. サーボモーター3 つでアームロボをつくる

サーボモーターを 3 つ使い、物をつかむアームロボを作成します。基板についているスイ ッチも使用して、ボタンを押す度に角度が変わるアームを作成します。変数ブロック、演 算ブロック、ボタンブロックを用いてボタンを押すごとに状態が変わる処理を作成します。

6.2.1. ロボットの組み立て

以下の手順でアームロボットを組み立てます。





6.2.2. 入出カポート情報の設定

ブロックプログラミング環境に Studuino 基板のポート情報を設定します。ブロックプログ ラミング環境のメニューバーの「編集」から「入出力設定...」を選択して、入出力設定ダ イアログボックスを開きます。入出力設定ダイアログボックスの「サーボモーター」エリ アで、D9、D10、D11 がチェックされていることを確認してください。

入出力設定		3
DCモーター 図 M1 図 M2	サーボモーター ボタン □ D2 □ D4 □ D7 □ D8 ☑ A0 ☑ A2 ☑ D9 ☑ D10 ☑ D11 □ D12 ☑ A1 ☑ A3	
センサー/LED/	ブザー サー D9,D10,D11 が チェックされている事を 確認してください	
チェックを全て外す	OK キャンセル	

6.2.3. プログラミング

サーボモーター1つで動きを確認します。

 「変数」パレットで「新しい変数を作る」をクリックし、表示される変数名設定ダイア ログで変数「flag1」を作成します。

Sservo_arm1- SPE BLOCK			
Based on Scratch from the MIT Media Lab) 変数名を入力	
動き 制御 順べる 深 変数 第 新しい変数を作る リストを作 変数名設定ダイ アログが表示さ れます	制御スタート ? 変える? flag1 OK キャンセル	3servo_arm1	
	3 2	ーー リック	

- 「変数」パレットから「flag1」を0 にするブロックを ^{● 制御スタート}ブロックにくっつけます。 3servo_arm1- SPE BLOCK Based on Scratch 🕀 🔚 ファイル 編集 実行 ヘルプ /**A** 3servo_arm1 動き 🚺 制御 調べる 演算) flag1 🔲 📙 制御スタート flag1 🔹 を 🕕 にする 新しい変数を作る 変数を削除 🗹 (flag1 ッグ&ドロップ flag1 - を 0 にする flag1 - を 1 ずつ変;
- プッシュスイッチ A0 が押される度に flag1 の値を変える処理部分を作成します。「制御」
 し (2005)
- パレットから

④ 「演算」パレットから ()=) ブロックをくっつけます。

3servo_arm1- SPE BLOCK	
Based on Scratch from the MIT Media Lab 🕀 📄 ファイル 編集 実行 ヘルプ	
動き 割御 調べる 注算 実数 「前回1 を ① にする ・・ ・ ・ ・	3servo_arm1 A
ドラッグ&ドロップ	

⑤ 「調べる」パレットから^{★☆ ▲●} ●● ブロックを ●● ブロックの左辺にくっつけ、右辺
 に0を設定します。



- ⑥ 「変数」パレットから^{「10g1」を0 にする}ブロックを ブロックの開いているところに
 - 入れます。

🧧 3servo_arm1- SPE BLOCk	к	
Based on Scratch from the MIT Media Lab	🕀 🖥 ファイル 編集 実行 ヘルプ	
動き 制御 調べる 清算 変数 第しい変数を作る 変数を削除 マ flag1 60 (こすこ)	3servo_arm1 / #//#12-ト flag1 で ドラッグ&ドロップ	
flag1 を 1 ずつ変える リストを作る		

⑦ 「演算」パレットから ご ブロックをくっつけます。

Sservo_arm1- SPE BLOCK		
Based on Scratch from the MIT Media Lab	ファイル 編集 実行 ヘルプ	
<mark>し</mark> 動き しんしん 制御		3servo_arm1 🦰 🦰 🔴
調べる <u>演算</u>		flag1 0
	flag1 _ & (0) (273)	
	ドラッグ&ドロップ	
1 から 10 までの乱数		
	flagi (. 58	
() mo ()		

⑧ 「変数」パレットから(1001)ブロックを○○ブロックの右辺にくっつけ、左辺に1を設定します。



以上で、プッシュスイッチ A0 が押された場合に flag1 の値を切り替える処理部分が完成しました。このブロックでは、プッシュスイッチ A0 が押されたとき、flag1 の値が 0 であれば 1 に、1 であれば 0 になります。



⑨ flag1の値に応じてサーボモーターの角度を変える処理を作成します。「制御」パレット

でなければ		
から	・ ブロックをドラッグします。	
Sservo_arm1- SPE BLOCK		
Based on Scratch from the MIT Media Lab	ファイル 編集 実行 ヘルプ]
動き 制御 調べる 演算 変数 演算 関数をコールする 1 1 砂待つ すっと 1 50 回線の返す 60 なら、ずっと 60 なら 1 なら 1 まで待つ	#################################	3servo_arm1

⑩ 「演算」パレットから ● ブロックをくっつけます。

ಕರ್ 🗢 ಭಕಿ

3servo_arm1- SPE BLOCK		
Based on Scratch from the MIT Media Lab	ファイル 編集 実行 ヘルプ	
(動き)	3servo_arm1	/ A
	flag1 & 0 (298)	
		- 1
■ から ■□ までの利数	もし < <u> ポタン A0 の</u> の 個 = () なら	
	flag1 t 1 flag1 (38	
ドラッグ&	ドロップ	
3700	etatina	
を力める		

 「変数」パレットから(1001)ブロックを()=)ブロックの左辺にくっつけ、右辺に1を設 定します。



ブロックの開いているところにくっつけます。 くっつけた上下のブロックの値部分にそ れぞれ 115, 90 を入力します。



以上で flag1 の値に応じてサーボモーターの角度を変える処理部分が完成しました。

13 12で作成したブロックとくっつけます。



これまでのステップで、プッシュスイッチを押す度にサーボモーターの角度を変える処理 が出来あがりましたが、このループの中で使われている処理は、すべて命令を与えた後即 座に次のステップに移ります。そのため、一度ボタンを押している間に何度もループを回 り、その都度 flag1 の値が切り替わり、違う角度が設定されてしまいます。これを回避する ため、次のステップで繰り返しの中に適当な待ち時間を設けます。この例では 0.2 秒としま す。



⑤ 「制御」パレットから¹⁰⁰⁰⁰ブロックをドラッグし、図の位置にくっつけます。

⑯ 1 秒 ブロックに 0.2 を設定します。



これにより、0.2 秒ごとにボタンの状態を判定するようになりました。

⑦ ブロックを^{1101 を 0 にする}ブロックにくっつけます。



以上で、プッシュスイッチ A0 を押すごとに D9 につながれたサーボモーターの角度を 115 度、90 度と変えるプログラムが作成できました。

ル メニューバーの「実行」から「プログラム作成・転送」を選択して、実際の動きを確認
 しましょう



プします。



② 「変数」パレットで「新しい変数を作る」をクリックし、表示される変数名設定ダイア
 ログで変数「flag2」を作成します。



22 複製したブロックの「flag1」を「flag2」に置き換えます。



② 1002ブロックを ブロックの右辺にドラッグします。もともとあった ブロックが飛び出すので、右クリックで削除します。下側の ブロックも同様にドラッグし、飛び出した ブロックを削除します。



29 ^{142 00} ブロックに A1 を選択します。さらに^{99 サーボモーター D9 を 90 度はる}ブロックに

D10 を選択します。

Studuino BLOCK Programming Environment				
Based on Scratch from the MIT Media Lab	⊕ 🗄	ファイル 編集	実行 ヘルプ	
 動き 割(可) 調べる 注算 変数 新しい変数を作る 変数を削除 「 flag1 べ flag2 flag1 を 0 にする flag1 を 1 ずつ変える 			もし ポペーム1 個 = 0 なら Hag2 を - Hag2 にする もし Hag2 こ 1 加 ウレ Hag2 こ 1 カ ウレ Hag2 こ 1 カ ウレ サーポモータ D10 を 115 度にする でなければ ・ ・ D10 を 90 度にする	flag1 1 flag2 0



³³ 出来上がった2つのブロックをさらに複製し、複製元とくっつけます。

28 3 つ目のサーボモーターの設定をします。 変数 flag3 を作成します。



② 2 つ目の時と同様の手順で、複製したブロックの値を変えていきます。変数は flag3 を、
 サーボモーターは D11 を選択します。



28 サーボモーターの角度を 50 度、150 度に変更します。





29 それぞれのブロックを白線の位置にくっつけます。

3) 「「「」」を 0 にする ブロックを 2 つドラッグし、図の位置にくっつけます。それぞれ flag2, flag3

を選択します。



以上で、ボタン A0, A1, A2 を押すごとに D9, D10, D11 につながれたサーボモーターの角度 を変えるアームのプログラムが作成できました。メニューバーの「実行」から「プログラ ム作成・転送」を選択して下さい。転送が完了すると、ボタン A0, A1, A2 を押すことによ り、アームが動作します。