基板マイコンカー製作キット
C 言語走行プログラム
解説マニュアル

第 1.01 版
2015 年 4 月 20 日
株式会社日立ドキュメントソリューションズ

注 意 事 項 (rev.6.0H)

著作権

・本マニュアルに関する著作権は株式会社日立ドキュメントソリューションズに帰属します。
・本マニュアルは著作権法および、国際著作権条約により保護されています。

禁止事項

ユーザーは以下の内容を行うことはできません。
・第三者に対して、本マニュアルを販売、販売を目的とした宣伝、使用、営業、複製などを行うこと
・第三者に対して、本マニュアルの使用権を譲渡または再承諾すること
・本マニュアルの一部または全部を改変、除去すること
・本マニュアルを無許可で翻訳すること
・本マニュアルの内容を使用しての、人命や人体に危害を及ぼす恐れのある用途での使用

転載、複製

本マニュアルの転載、複製については、文書による株式会社日立ドキュメントソリューションズの事前の承諾が必要です。

責任の制限

本マニュアルに記載した情報は、正確を期すため、慎重に制作したものですが万一本マニュアルの記述誤りに起因する損害が生じた場合でも、株式会社日立ドキュメントソリューションズはその責任を負いません。

その他

・本マニュアルに記載の情報は本マニュアル発行時点のものであり、株式会社日立ドキュメントソリューションズは、予告なしに、本マニュアルに記載した情報または仕様を変更することがあります。製作に当たりましては、最新の内容を確認いただきますようお願いします。
・すべての商標および登録商標は、それぞれの所有者に帰属します。

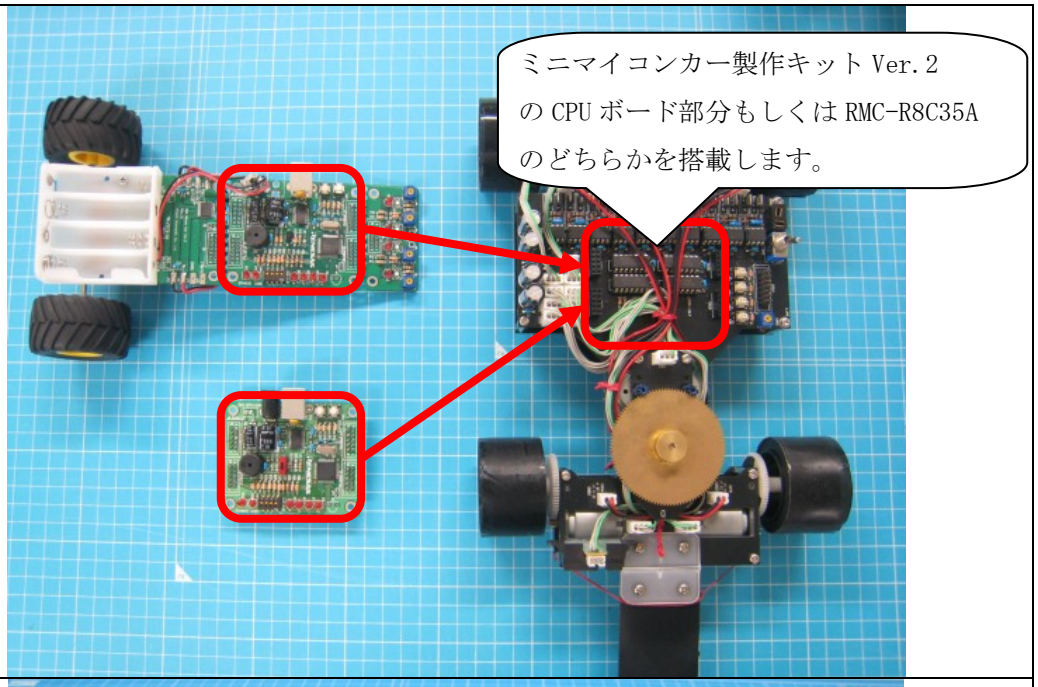
連絡先

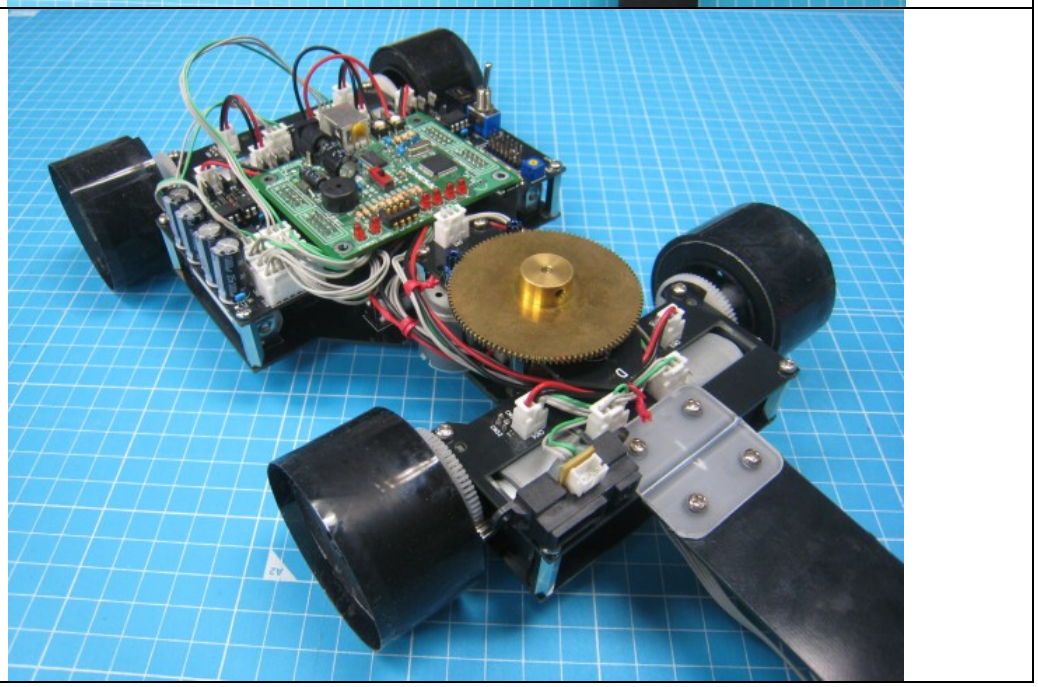
株式会社 日立ドキュメントソリューションズ
〒135-0016 東京都江東区東陽六丁目 3 番 2 号 イースト 21 タワー
E-mail：himdx.m-carrally.dd@hitachi.com

目次

1. 概要

基板マイコンカー製作キットは、ミニマイコンカー製作キット Ver.2（RMC-R8C35A）を使用し、ジャパンマイコンカーラリーの Advanced Class に出場できる車体にするためのキットです。
本書は、基板マイコンカー製作キットの組み立て方法を解説します。



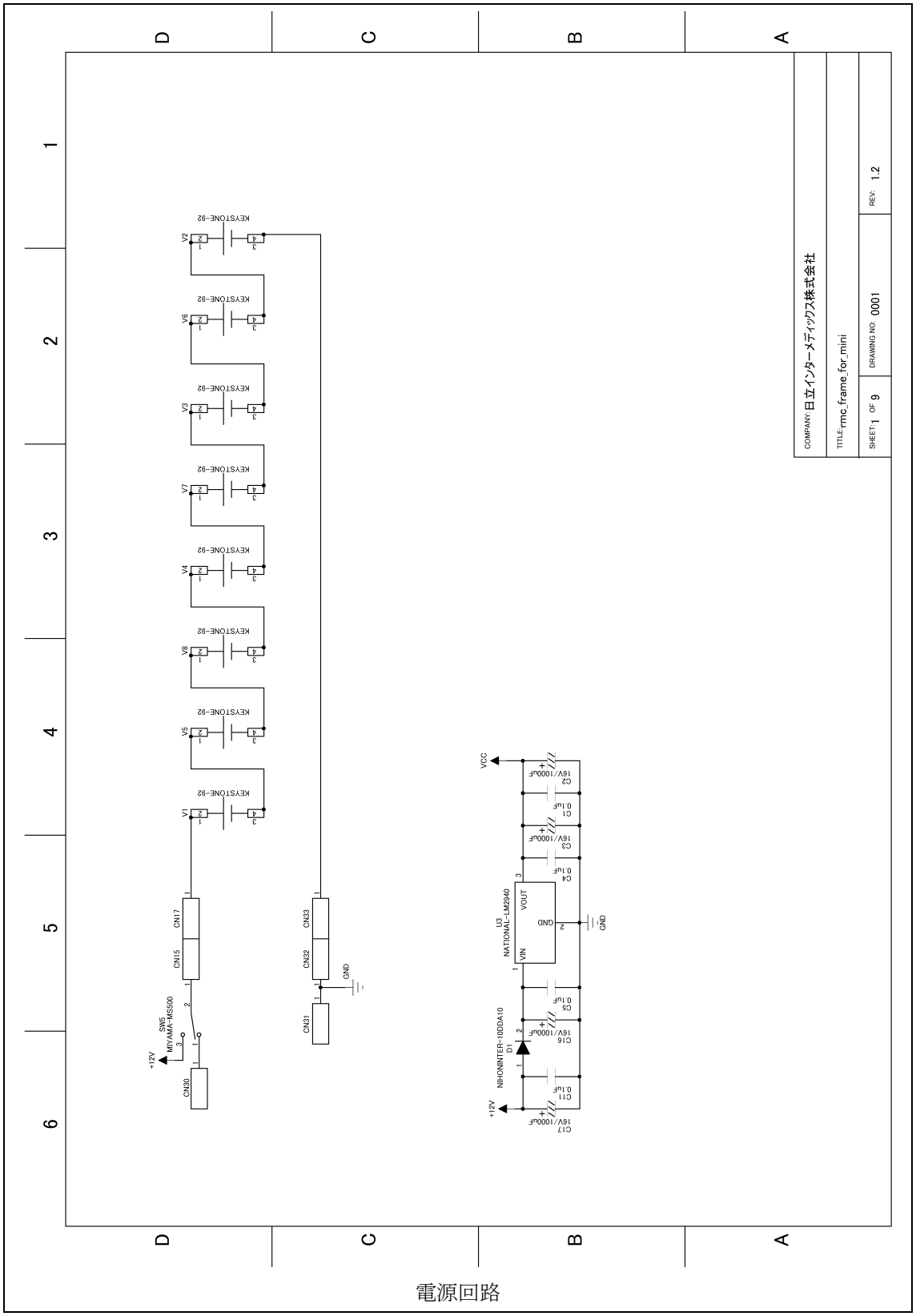


2. 仕様

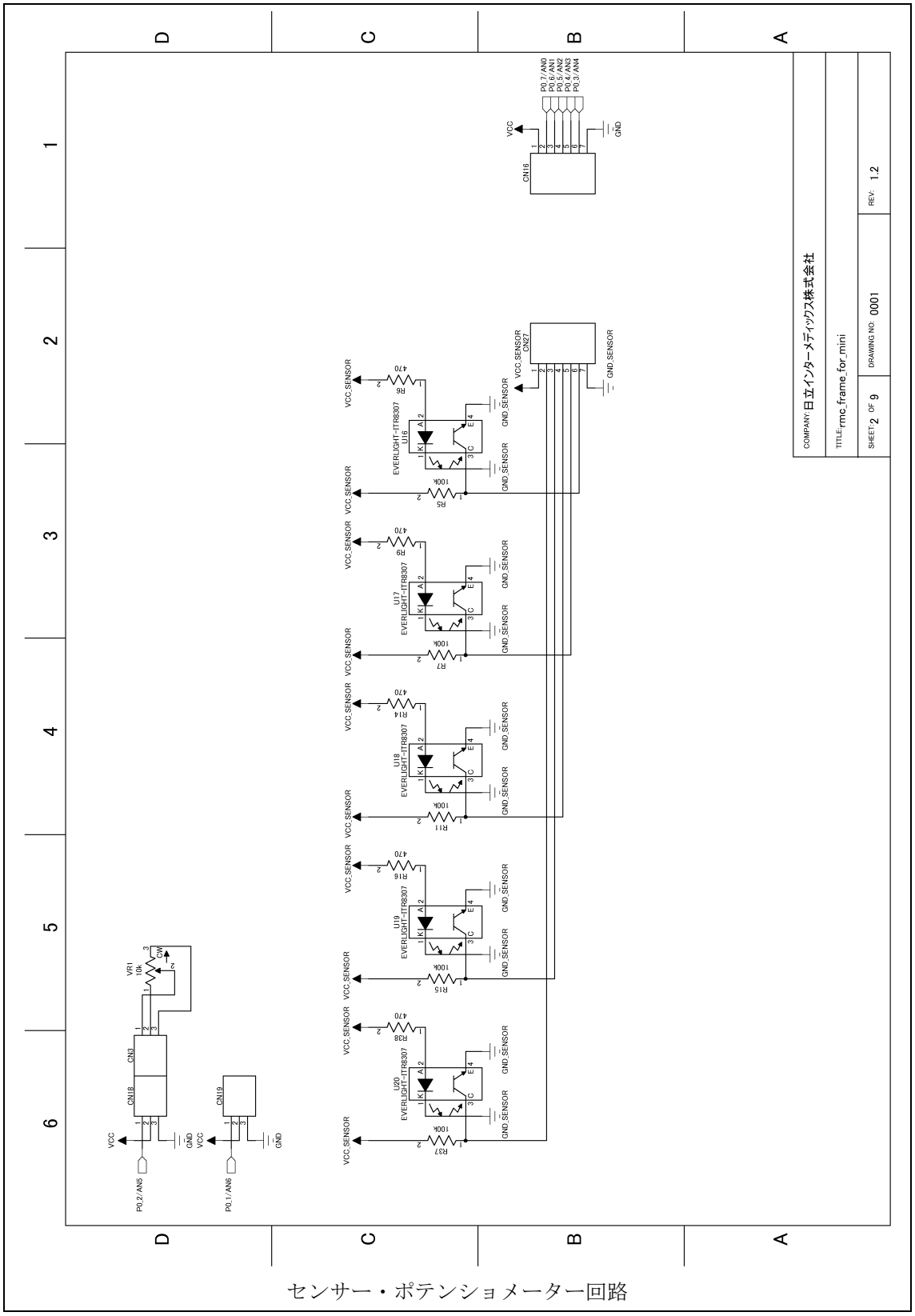
2.1 仕様

内容	詳細
全長	470mm
全幅	170mm
全高	70mm
重量	985g（電池含む）
ホイールベース	180mm
トレッド F/R	145mm/145mm
ギア比	駆動 8：1（8T/64T） ステアリング 58.7：1（8T/110T、15T/64T）
電源	単 3 型充電池×8 本
モータードライバ	ローサイド、ハイサイド Nch MOS FET H ブリッジ×5 個 タイマーRC と RD の PWM により、モーターを 5 個駆動可能 10kHz 駆動、フリー、短絡ブレーキ、逆転可能 （デューティ 100%では駆動できません。）
エンコーダー	1 回転 2 パルスのエンコーダー×4 個 タイマーRD のインプットキャプチャにより、各車輪の速度とパルス数を計測可能
センサー	アナログ入力の赤外線フォトインタラプタ×5 個
ポテンショメータ	360deg/10bit=0.35［deg/bit］分解能のポテンショメータ×1 個
その他	液晶コネクタ×1 個 スイッチ×4 個

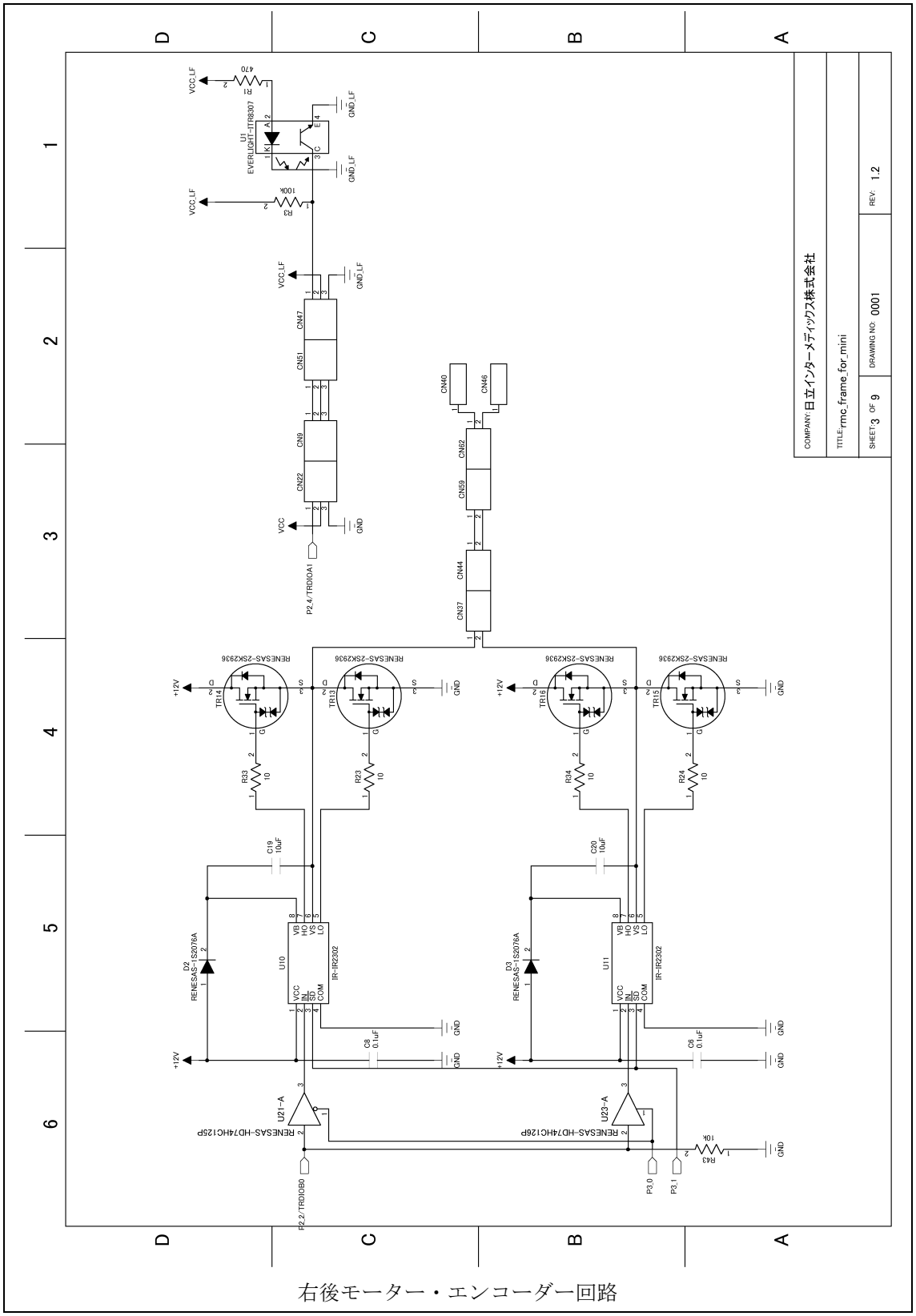
2.2 回路図


COMPANY: 日立インターメディックス株式会社
TITLE: rmc_frame_for_mini
DRAWING NO: 0001
REV: 1.2
SHEET 1 OF 9
KEYSTONE-92
V1
V2
V3
V4
V5
V6
V7
V8
CN17
CN15
CN30
CN33
CN32
CN31
SW5
MIYAMA-MSS500
+12V
GND
VCC
U3
NATIONAL-LM2940
VOUT
VIN
GND
D1
NIHONINTER-10DDA10
C1 0.1uF
C2 16V/100uF
C3 0.1uF
C4 0.1uF
C5 0.1uF
C16 16V/100uF
C17 16V/100uF

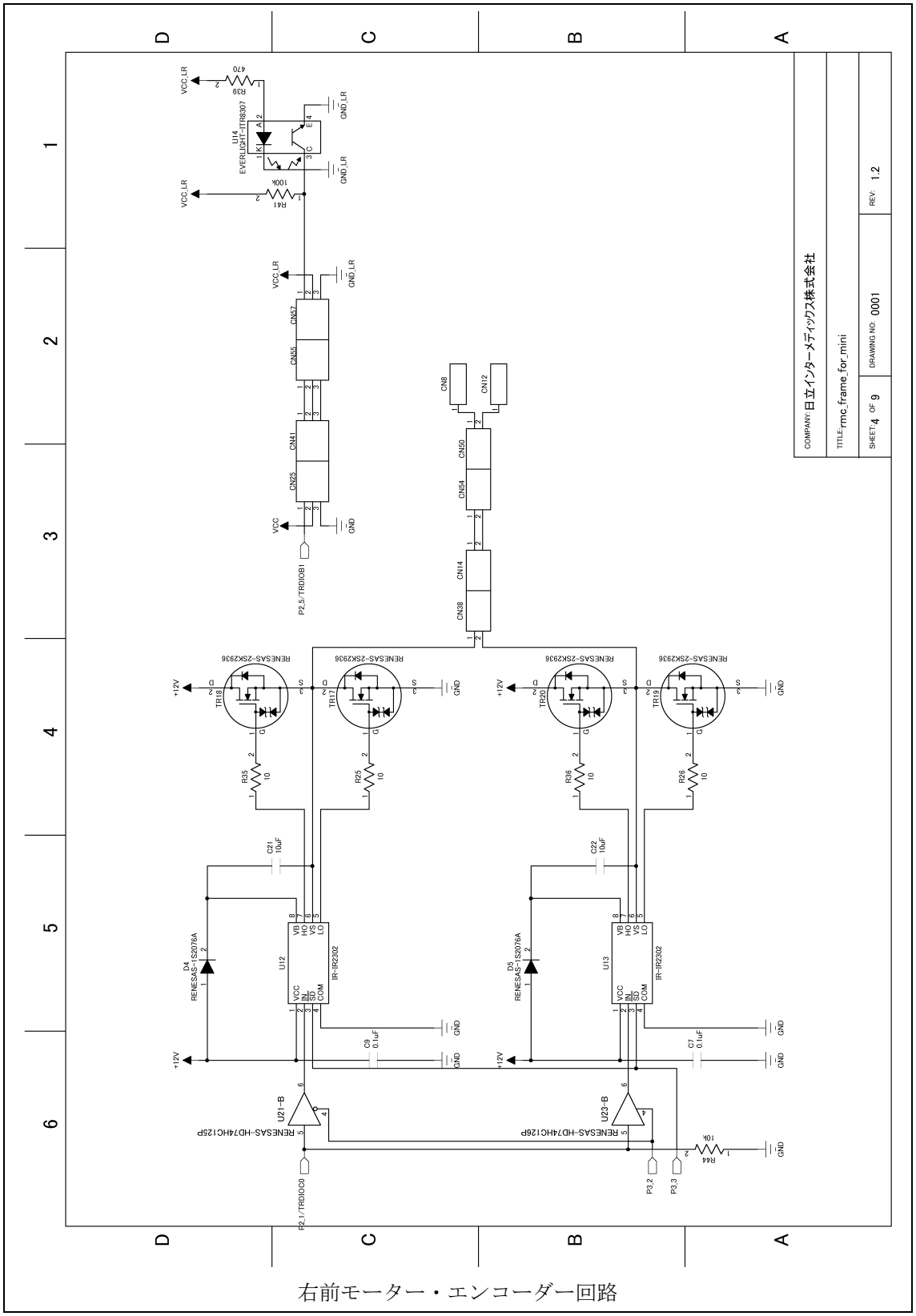
電源回路



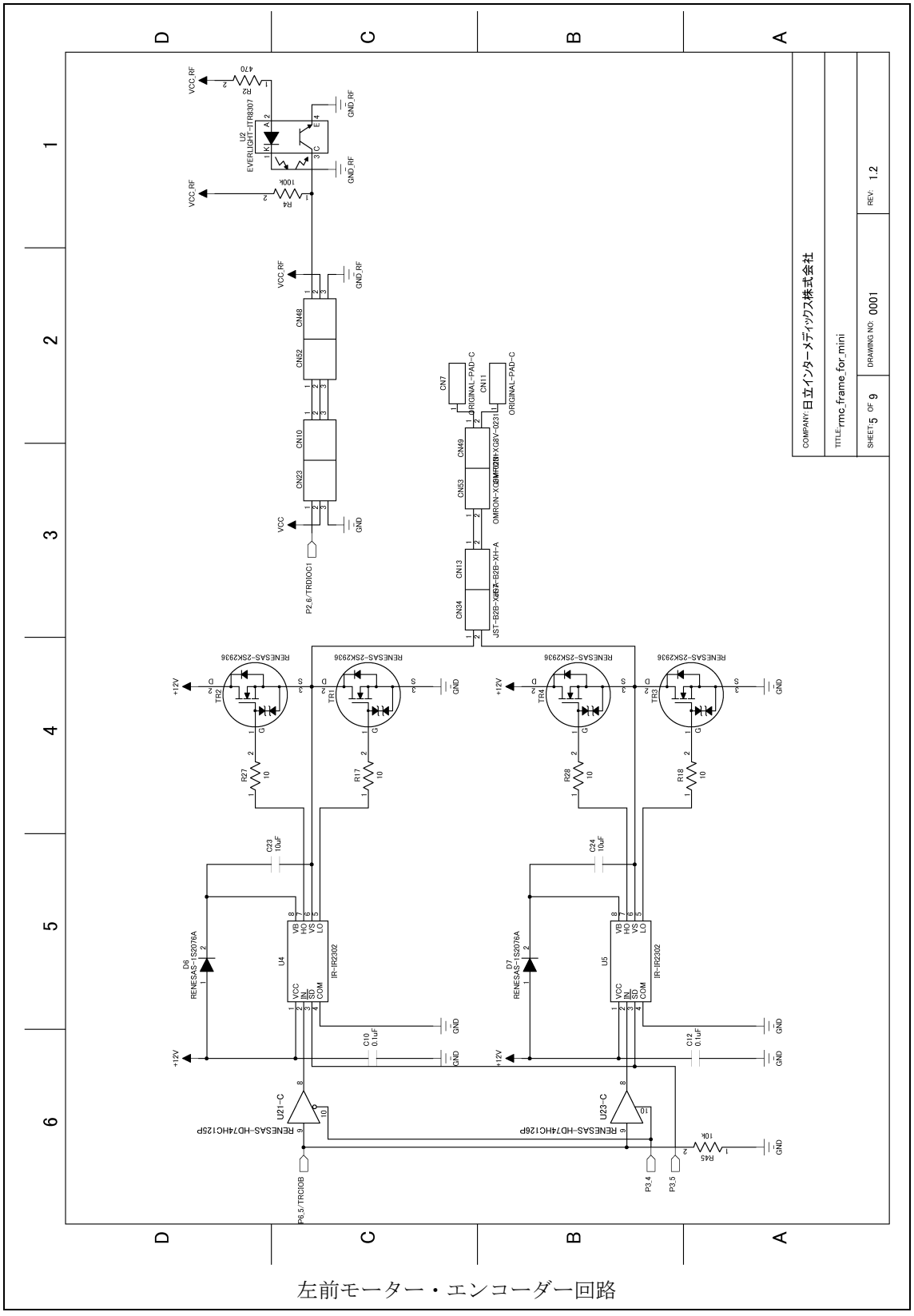
センサー・ポテンショメーター回路



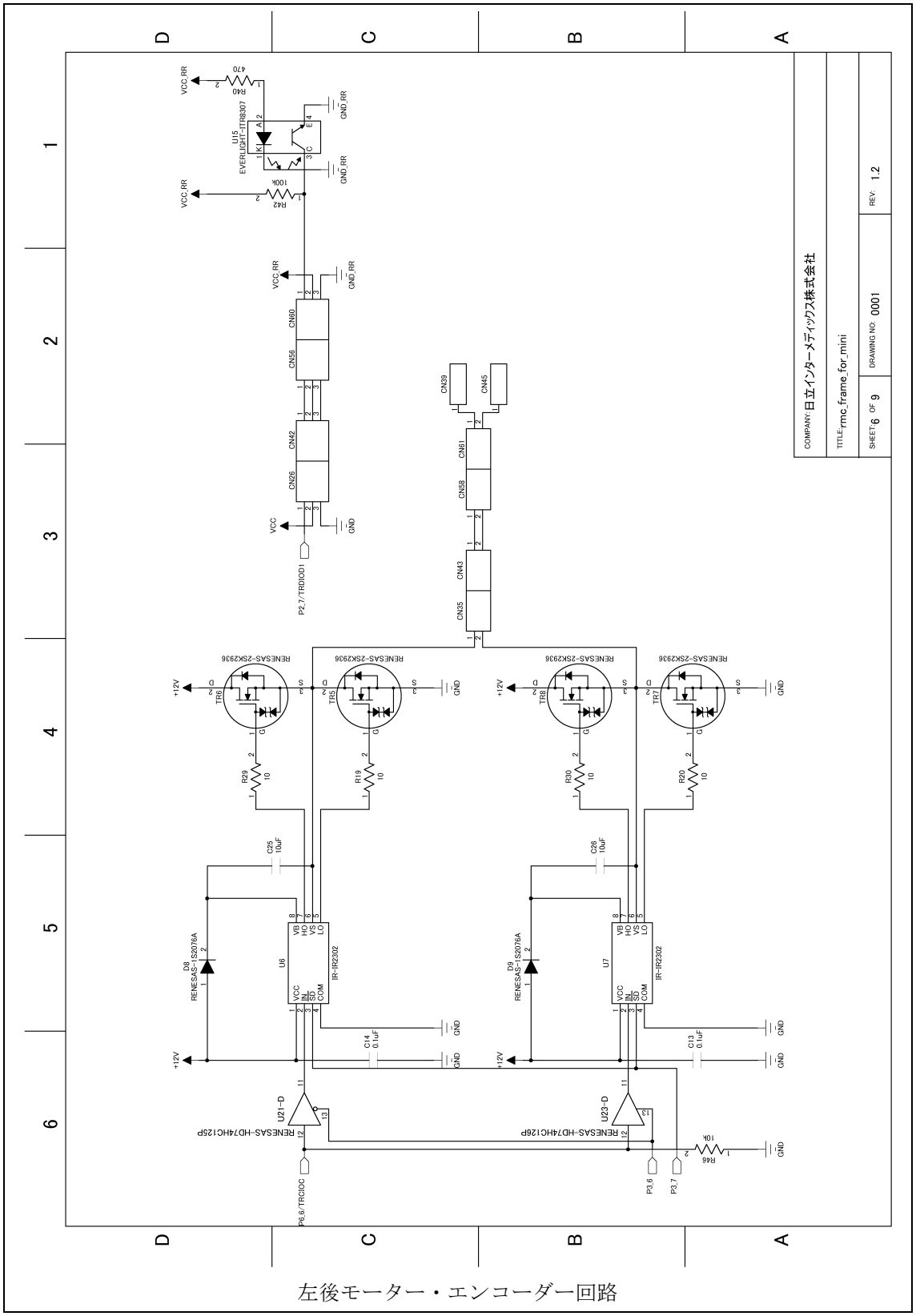
右後モーター・エンコーダー回路



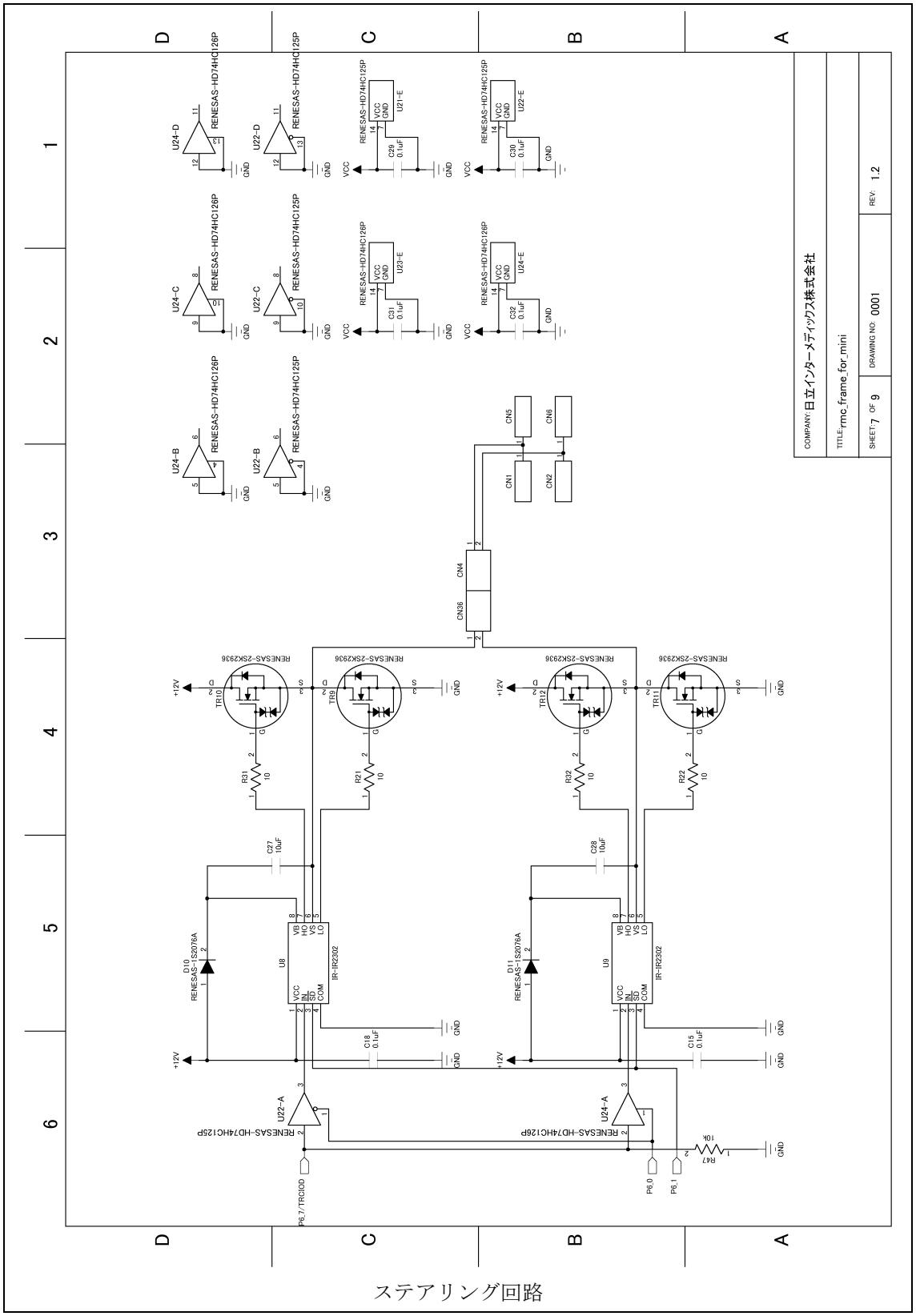
右前モーター・エンコーダー回路



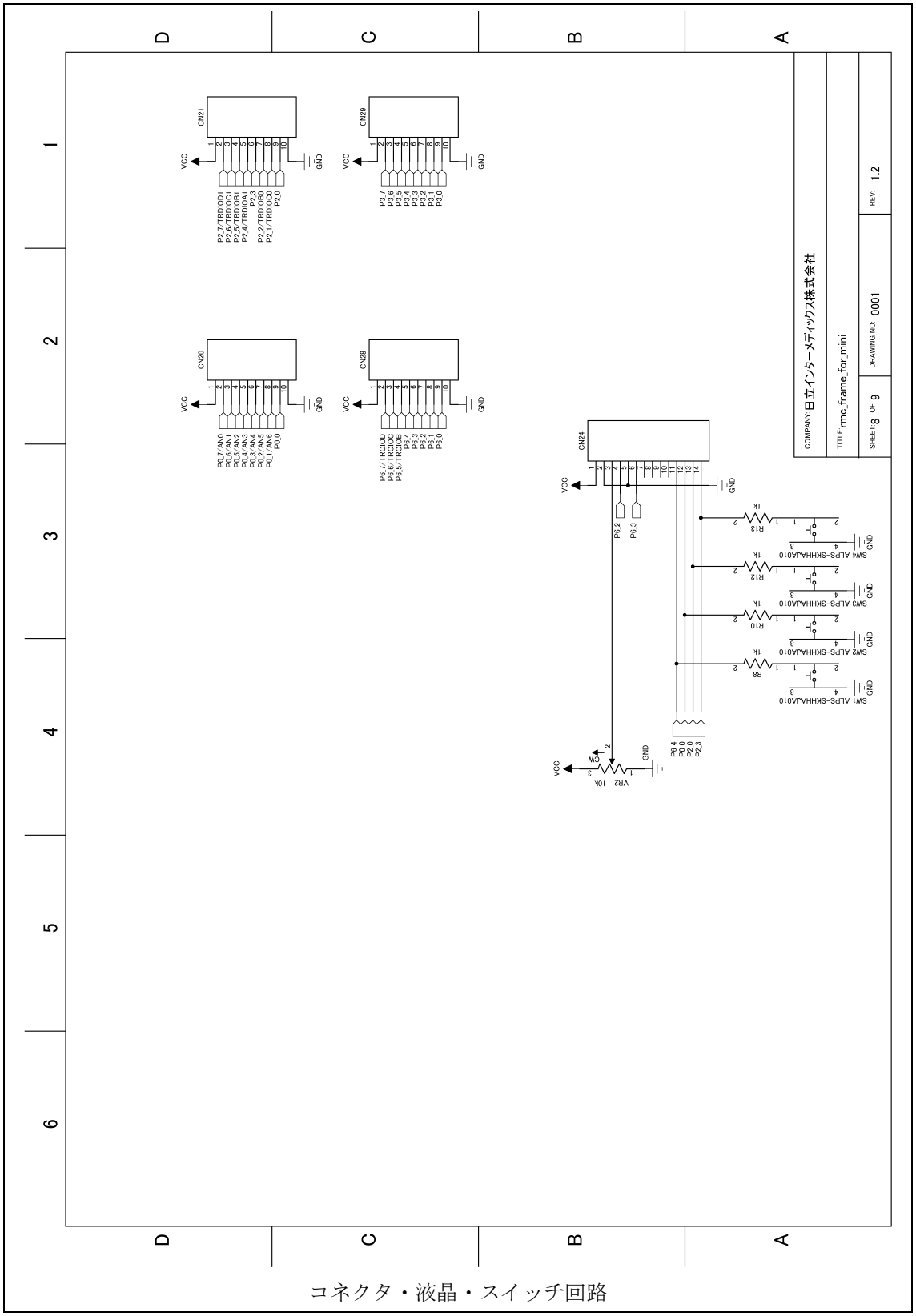
左前モーター・エンコーダー回路



左後モーター・エンコーダー回路


COMPANY 日立インターメディックス株式会社
TITLE rmc_frame_for_mini
DRAWING NO. 0001
REV. 1.2
SHEET 7 OF 9
RENESAS-HD74HC126P
RENESAS-2SK2936
RENESAS-1S2076A
IR-IR2302
P6_7/TRGIOD
P6_0
P6_1

ステアリング回路



コネクタ・液晶・スイッチ回路

2.3 ポート表

コネクタ	番号	端子名	機能
CN20	1	VCC	
	2	P0_7/AN0	センサー左
	3	P0_6/AN1	センサー左中
	4	P0_5/AN2	センサー中央
	5	P0_4/AN3	センサー右中
	6	P0_3/AN4	センサー右
	7	P0_2/AN5	ポテンショメータ
	8	P0_1/AN6	PSD
	9	P0_0	LCD/スイッチ
	10	GND	

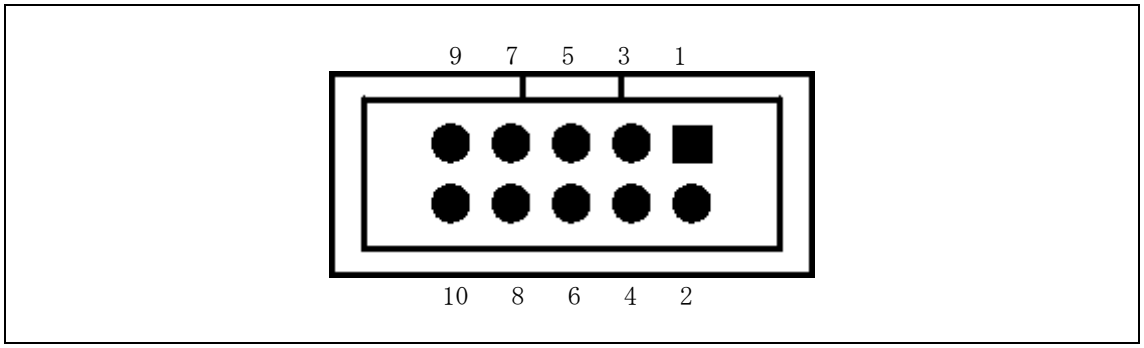
コネクタ	番号	端子名	機能
CN21	1	VCC	
	2	P2_7/TRDIOD1	ENC 右後
	3	P2_6/TRDIOC1	ENC 右前
	4	P2_5/TRDIOB1	ENC 左後
	5	P2_4/TRDIOA1	ENC 左前
	6	P2_3	LCD/スイッチ
	7	P2_2/TRDIOB0	PWM 右後
	8	P2_1/TRDIOC0	PWM 右前
	9	P2_0	LCD/スイッチ
	10	GND	

コネクタ	番号	端子名	機能
CN28	1	VCC	
	2	P6_7/TRCIOD	PWM ステアリング
	3	P6_6/TRCIOC	PWM 左後
	4	P6_5/TRCIOB	PWM 左前
	5	P6_4	LCD/スイッチ
	6	P6_3	LCD/スイッチ
	7	P6_2	LCD/スイッチ
	8	P6_1	¥FREE ステアリング
	9	P6_0	CCW ステアリング
	10	GND	

コネクタ	番号	端子名	機能
CN29	1	VCC	
	2	P3_7	¥FREE 左後
	3	P3_6	CCW 左後
	4	P3_5	¥FREE 左前
	5	P3_4	CCW 左前、ブザー
	6	P3_3	¥FREE 右前
	7	P3_2	CCW 右前
	8	P3_1	¥FREE 右後
	9	P3_0	CCW 右後
	10	GND	

2.4　ピン配置図

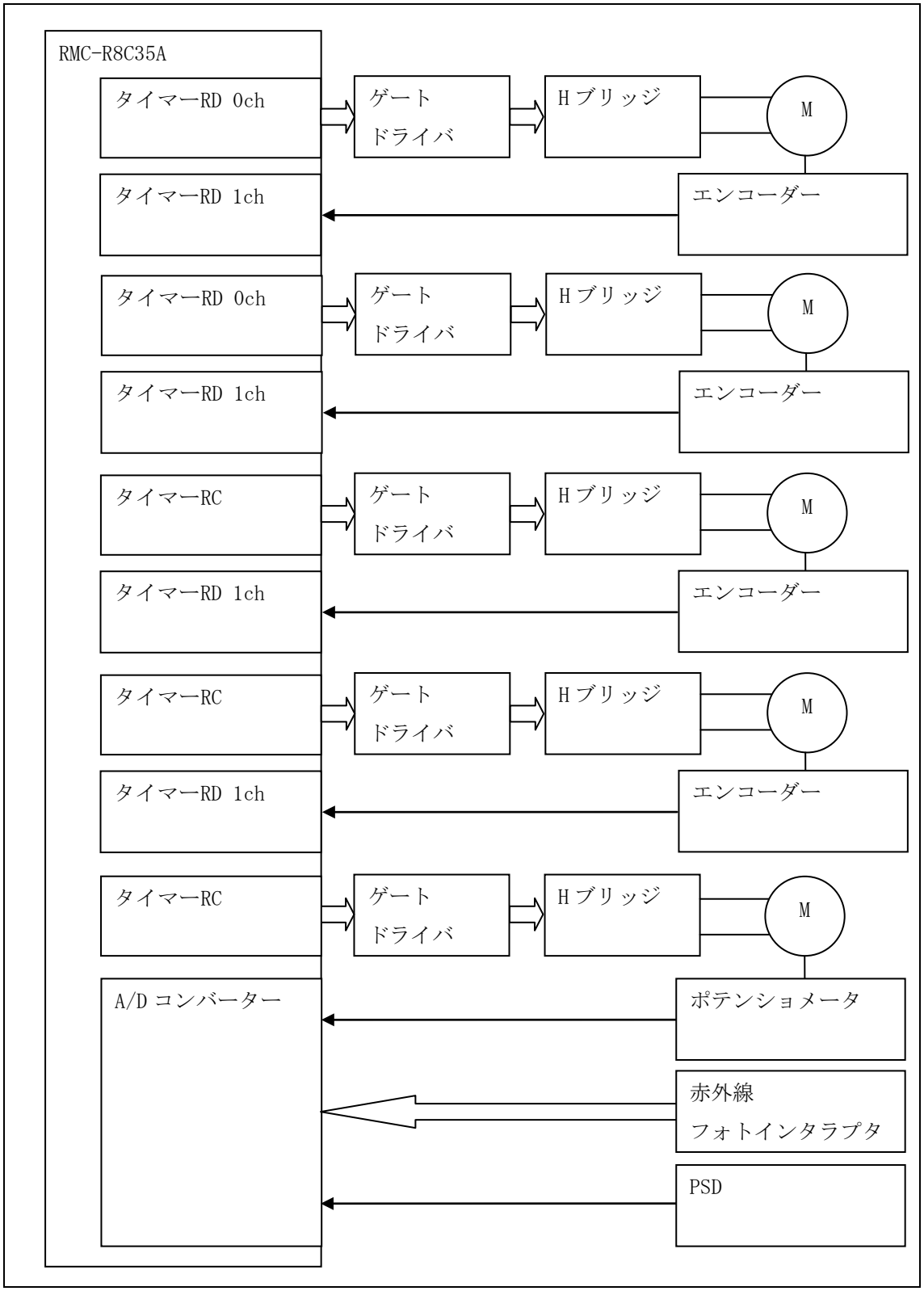
10 ピンコネクタのピン配置



2.5 内蔵周辺モジュールの役割

内蔵周辺モジュール		機能	動作
タイマーRB		コンペアマッチ割り込み TRBIC	1ms ごとにタイマーのカウント
			スイッチの入力処理
			液晶の表示処理
			エンコーダーのカウント処理
			モーターのフィードバック制御
			ステアリングのフィードバック制御
タイマーRC		PWM モード	2kHz の PWM 信号生成
タイマーRD	0ch	PWM モード	2kHz の PWM 信号生成
	1ch	インプットキャプチャー	エンコーダーの信号取得
		オーバーフロー割り込み TRD1IC	16 ビット以上にカウンタを拡張するための処理
10bitAD		繰り返し掃引モード	各センサーの AD 変換

2.6 回路ブロック図


RMC-R8C35A
タイマーRD 0ch
ゲート
ドライバ
Hブリッジ
M
タイマーRD 1ch
エンコーダー
タイマーRD 0ch
ゲート
ドライバ
Hブリッジ
M
タイマーRD 1ch
エンコーダー
タイマーRC
ゲート
ドライバ
Hブリッジ
M
タイマーRD 1ch
エンコーダー
タイマーRC
ゲート
ドライバ
Hブリッジ
M
タイマーRD 1ch
エンコーダー
タイマーRC
ゲート
ドライバ
Hブリッジ
M
A/D コンバーター
ポテンショメータ
赤外線
フォトインタラプタ
PSD

2.7 関数一覧

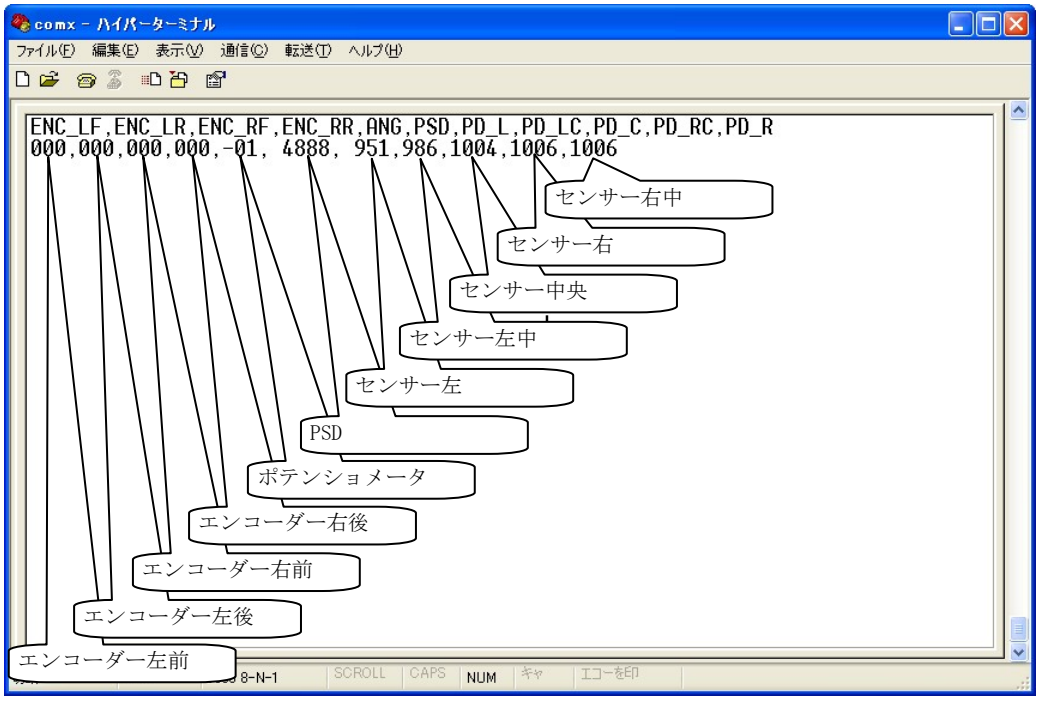
フォルダ名	ファイル名	関数名
common	enc.c	void enc_process(void)
		void enc_process2(void)
		unsigned short enc_speed_get(unsigned char ch)
		unsigned short enc_odo_get(unsigned char ch)
	motor.c	void motor_process(void)
		motor_speed_put(signed short lf, signed short lr, signed short rf, signed short rr, unsigned short speed)
	sensor.c	void init_sensor(void)
		signed short potentio_angle_get(void)
		unsigned short position_distance_get(void)
		unsigned char sensor_digital_get(void)
	steering.c	void steering_process(void)
		void steering_angle_put(signed short angle)
	lcd.c	液晶・microSD 基板　液晶・スイッチセットのプログラムを移植しているため、本書では解説しません。
	switch.c	液晶・microSD 基板　液晶・スイッチセットのプログラムを移植しているため、本書では解説しません。 ※SW2 は液晶接続時のみ動作します。
mini_mcr	mini_mcr.c	void main(void)
		void init(void)
		void intTRBIC(void)
		void intTRD1IC(void)
		void timer(unsigned long data1) ミニマイコンカー製作キット Ver.2　C 言語走行プログラム解説マニュアルで解説しているため、本書では解説しません。
		unsigned char dipsw(void) ミニマイコンカー製作キット Ver.2　C 言語走行プログラム解説マニュアルで解説しているため、本書では解説しません。
		void ayc(unsigned short duty ,unsigned short speed)

3. 動作確認

「mini_mcr_test」をアクティブなプロジェクトに設定し、ビルドを行い書き込みます。
ハイパーターミナルなどの通信ソフトで、ミニマイコンカーと接続すると、センサーの各種状態が確認できますので、正しく動作していることを確認してください。

ポートの設定

項目	設定値
ビット／秒	9600
データビット	8
パリティ	なし
ストップビット	1
フロー制御	なし



エンコーダー（左前、左後、右前、右後）	車輪を手で動かして、カウントアップされることを確認します。
ポテンショメータ	ステアリングを手で動かして、角度が表示されることを確認します。
PSD	PSD に手をかざして、距離が表示されることを確認します。
センサー（左、左中、中央、右中、右）	ラインセンサーをコースに置き、値が変化することを確認します。

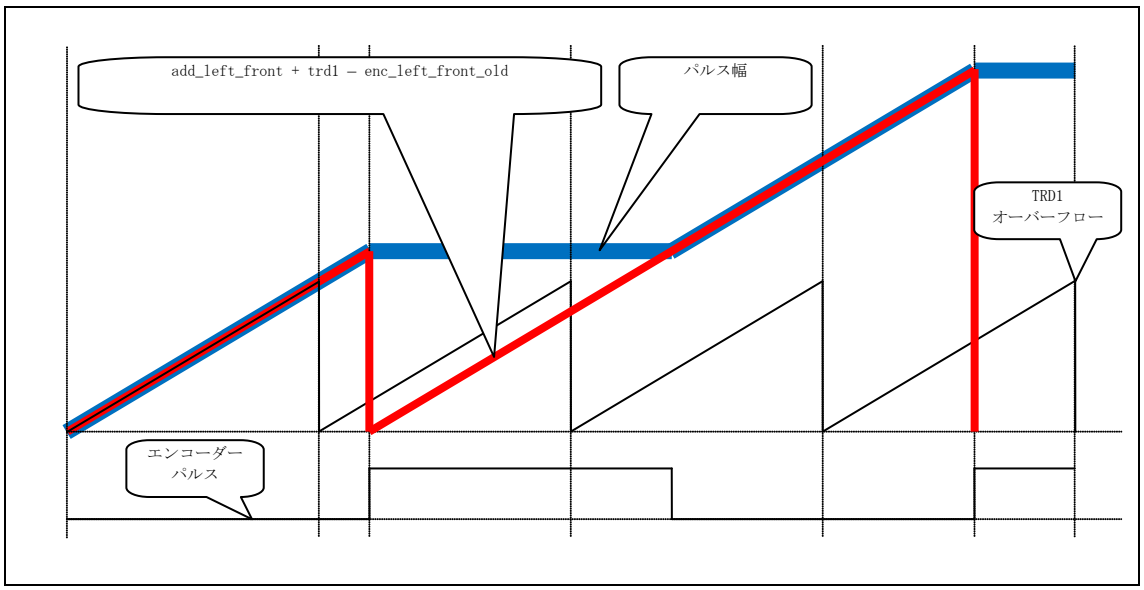
ハイパーターミナルで、センサーが問題ないことが確認できたら、
タイヤは設置させず、SW1 を押して、すべてのタイヤが正転していることを確認します。
また、ラインに追従することを確認します。
SW1 をもう一度押して、タイヤが逆転して停止することを確認します。

4. enc.c

enc.c は、エンコーダーの速度および距離の計算を行います。

void enc_process(void)	エンコーダーの速度・距離計算
void enc_process2(void)	インプットキャプチャのカウンタを 32 ビットに拡張するための処理
unsigned short enc_speed_get(unsigned char ch)	各車輪の速度取得
unsigned short enc_odo_get(unsigned char ch)	各車輪の距離取得

4.1 速度および距離の計算のしくみ

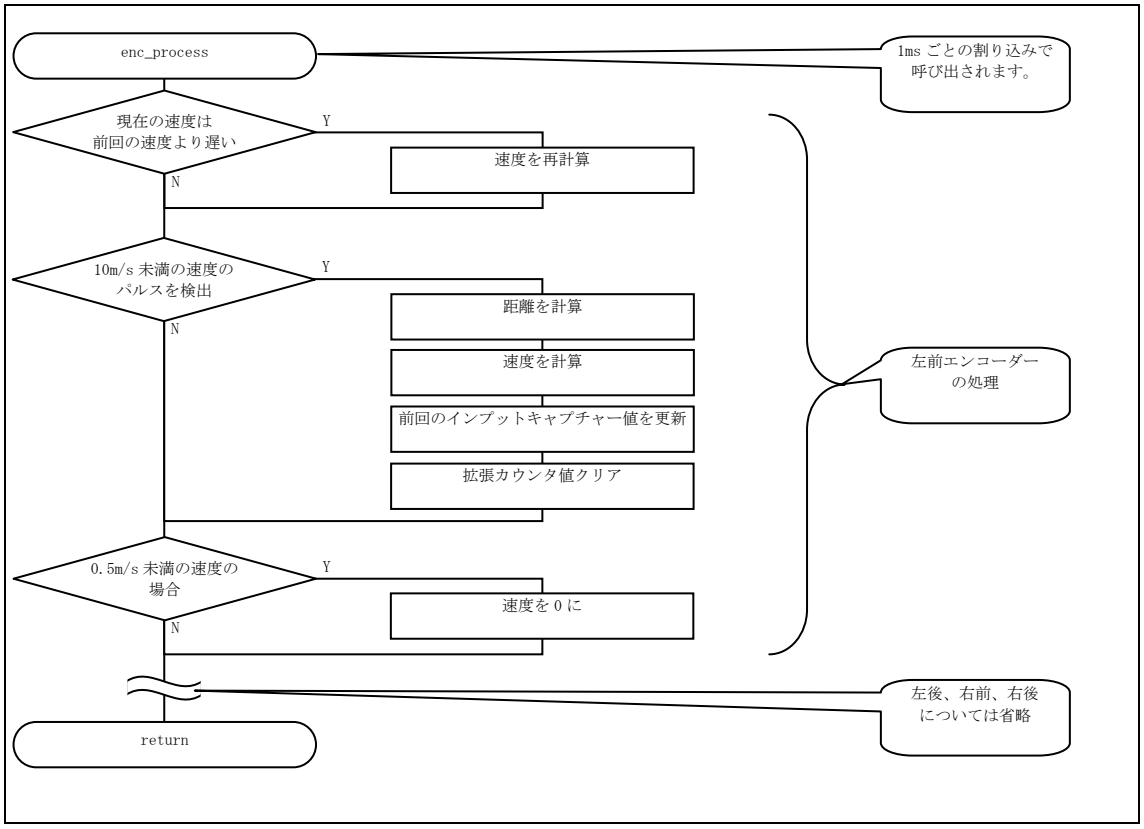


エンコーダーの立ち上がりエッジ間のパルス幅をインプットキャプチャーで測定し、速度を計算します。また、パルス数をカウントし、距離とします。
パルス幅は trd1 で測定しますが、4ch 分のエンコーダーの測定をしなくてはいけないので、カウンタのクリアを行うことはできません。
そのため、1ms ごとにインプットキャプチャーの値が更新されたかを監視し、更新前の値と、更新後の値の差分を取ることで、パルス幅を測定します。
また、差分を取ることでパルス幅を測定すると、trd1 のオーバーフローをまたぐような波形が入力されたときに、測定値が正しくなくなります。オーバーフロー時に各 ch のカウンタ拡張用の変数に 65536 を足して、それぞれのチャンネルが独立した 32 ビットのカウンタとなるようにしています。

4.2 enc_process 関数

enc_process 関数は、1ms ごとに割り込みで呼び出されます。
エンコーダーの速度および距離の計算を行います。

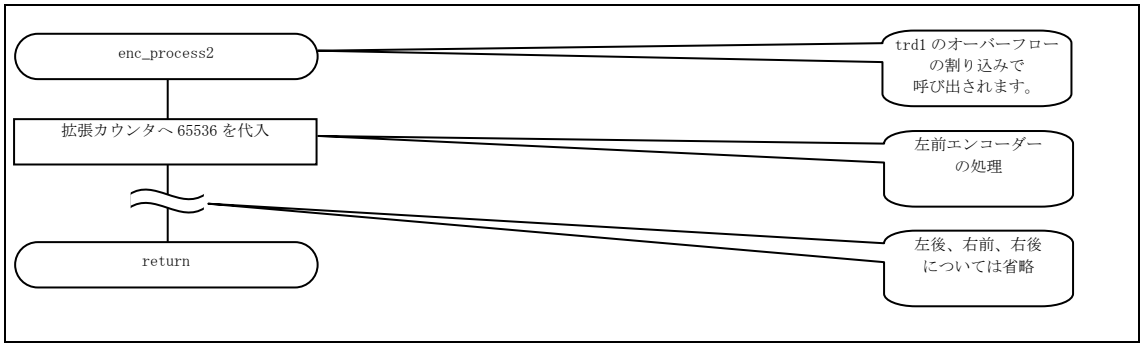
フローチャート



4.3 enc_process2 関数

enc_process2 関数は、trd1 オーバーフロー割り込みで呼び出されます。
オーバーフロー時に各 ch のカウンタ拡張用の変数に 65536 を足して、それぞれのチャンネルが独立した 32 ビットのカウンタとなるようにしています。

フローチャート

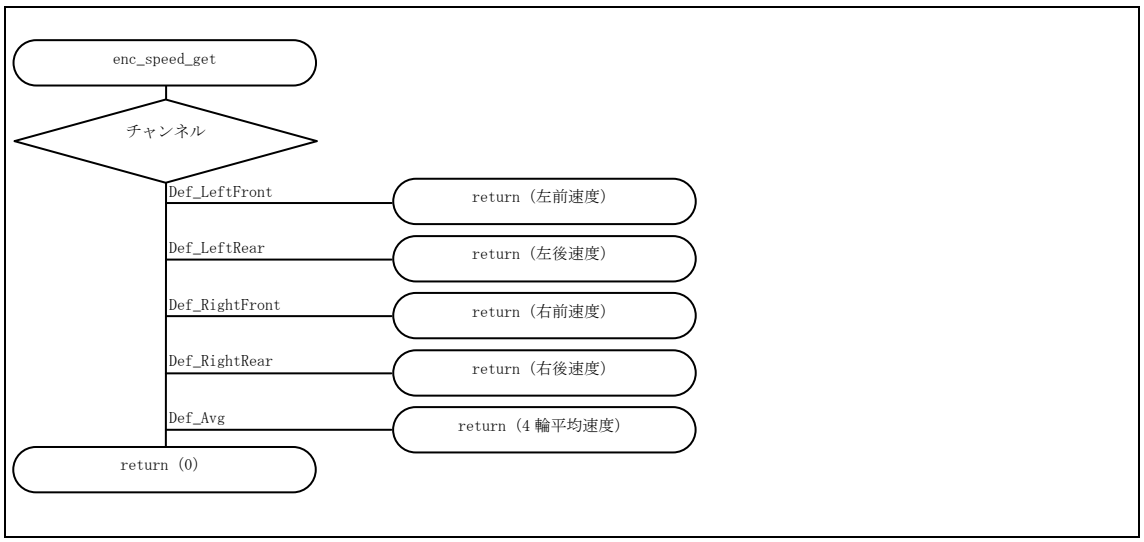


4.4 enc_speed_get 関数

enc_speed_get 関数は、引数でチャンネルを指定すると、各車輪の速度[mm/s]が取得できます。

引数	動作
Def_LeftFront	左前の速度を返します。
Def_LeftRear	左後の速度を返します。
Def_RightFront	右前の速度を返します。
Def_RightRear	右後の速度を返します。
Def_Avg	4 輪の平均速度を返します。
その他	0 を返します。

フローチャート

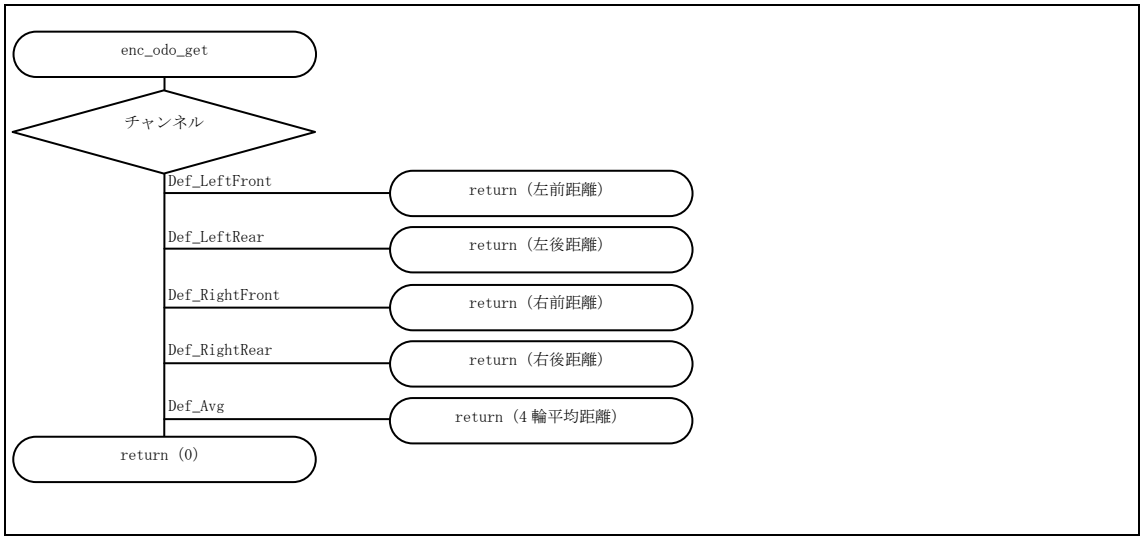


4.5 enc_odo_get 関数

enc_odo_get 関数は、引数でチャンネルを指定すると、各車輪の距離［パルス数］が取得できます。

引数	動作
Def_LeftFront	左前の距離を返します。
Def_LeftRear	左後の距離を返します。
Def_RightFront	右前の距離を返します。
Def_RightRear	右後の距離を返します。
Def_Avg	4 輪の平均距離を返します。
その他	0 を返します。

フローチャート



5. motor.c

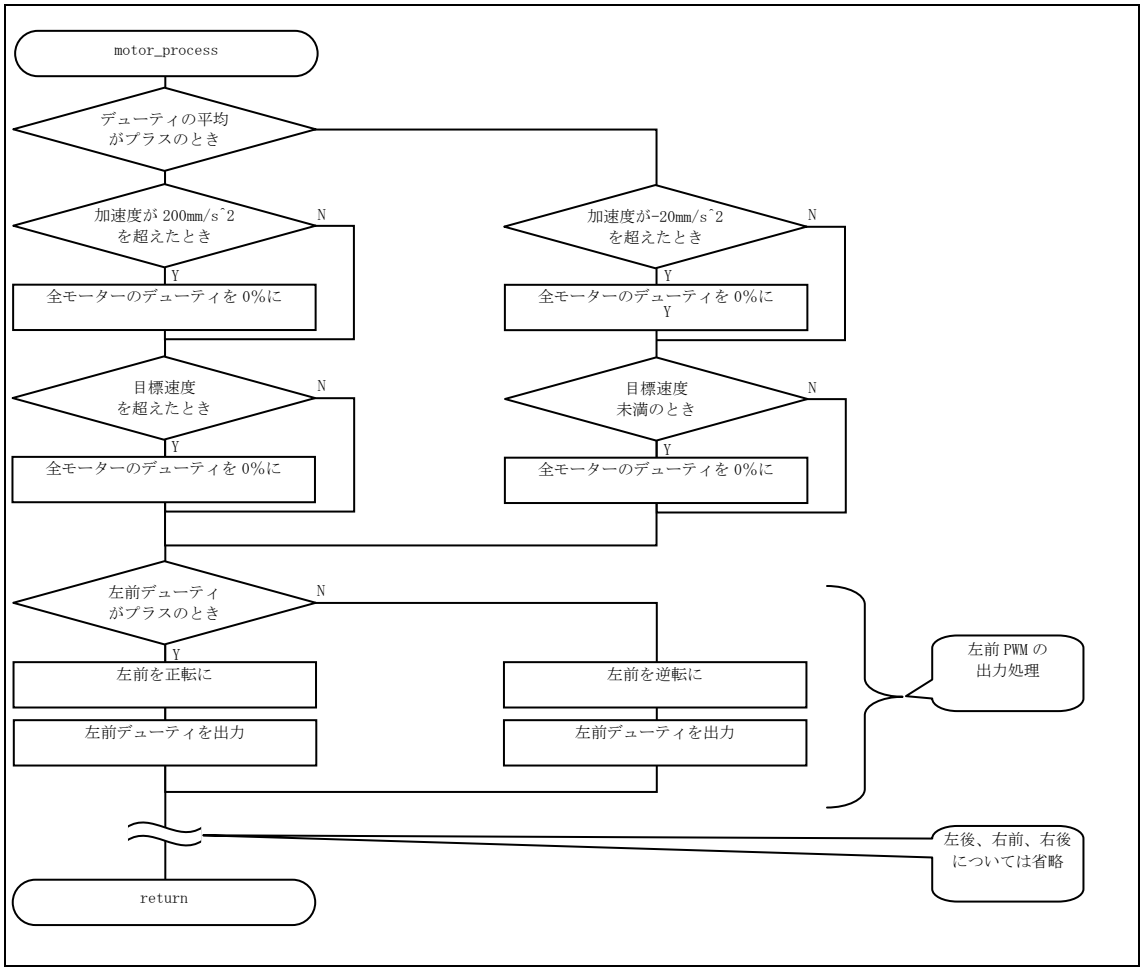
motor.c は、モーターの速度制御を行います。

void motor_process(void)	モーターの速度制御
motor_speed_put(signed short lf, signed short lr, signed short rf, signed short rr, unsigned short speed)	モーターの目標速度設定

5.1 motor_process 関数

motor_process 関数は、1ms ごとに呼び出されます。
モーターの速度制御を ON-OFF 制御で行います。逆転ブレーキをかけた場合、車体がバックしないように加速度を監視して、ショートブレーキに切り替える処理を行います。

フローチャート

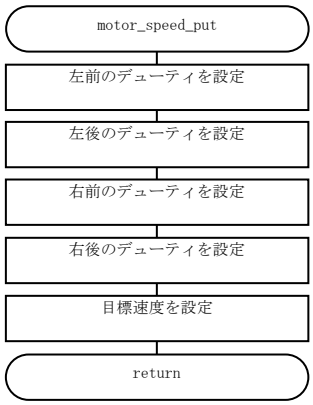


5.2 motor_speed_put 関数

motor_speed_put 関数は、引数で各車輪のデューティ［%］と目標速度を［mm/s］設定します。
デューティにマイナスの値を設定すると逆転ブレーキがかかります。
目標速度が速く、デューティが小さい場合、目標速度に到達しない場合があります。

引数	動作
lf	左前のデューティを設定します。
lr	左後のデューティを設定します。
rf	右前のデューティを設定します。
rr	右後のデューティを設定します。
speed	目標速度を設定します。

フローチャート



6. sensor.c

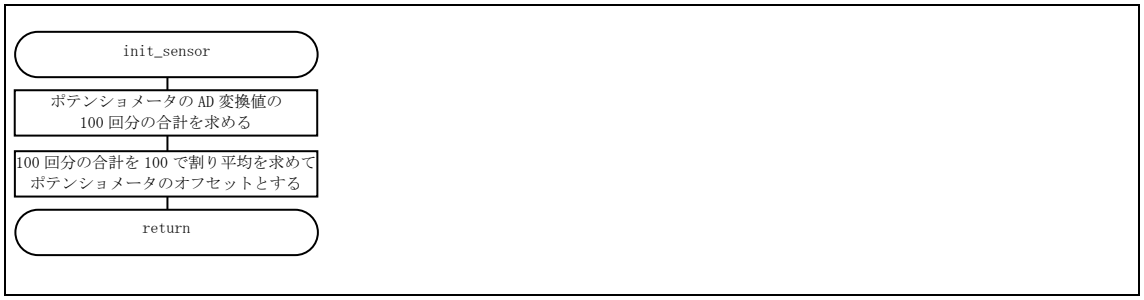
sensor.c は、ポテンショメータ、PSD、センサの処理を行います。

void init_sensor(void)	ポテンショメータの オフセット設定
signed short potentio_angle_get(void)	ポテンショメータの角度取得
unsigned short position_distance_get(void)	PSD の距離取得
unsigned char sensor_digital_get(void)	センサーのデジタル値取得

6.1 init_sensor 関数

init_sensor 関数は、ポテンショメータのオフセットを設定します。
この関数を呼び出した時点のステアリングの角度が 0 度となります。

フローチャート

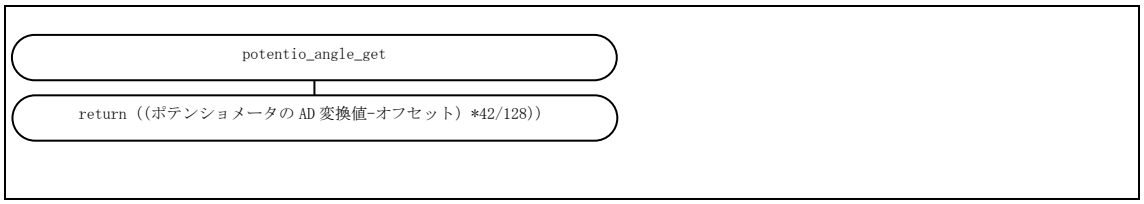


6.2 potentio_angle_get 関数

potentio_angle_get 関数は、ポテンショメータの角度［度］を返します。

ポテンショメータの角度と電圧の関係について、ポテンショメータのマニュアルから、
100%/333.3［度］の勾配の比例関係にある事が分かりますので次式により、AD 変換値 x から角度 y が求まります。
y=x*333.3/1024
x が 1023 の時、x*333.3 を計算すると、16 ビットの範囲を超えてしまいますので、分子分母ともに 8 で割り、
y=x*42/128
とすることで 16 ビットの範囲内で計算を行います。

フローチャート



6.3 position_distance_get 関数

position_distance_get 関数は、PSD で測定した距離［mm］を返します。

PSD の電圧と距離の関係について、PSD のマニュアルから、電圧と距離は反比例の関係にある事が分かります。そこで、2 回分の電圧 x と距離 y の関係を測定し、反比例の式
y=a/（x+b）…①
に代入し、連立方程式を解くことで、定数を求めます。
x=690［mV］のとき y=400［mm］
x=1240［mV］のとき y=200［mm］
上記の測定データが得られたとします（ミリ単位なのは小数を扱わないためです)。
このデータを①に代入し
400=a/（690+b）…②
200=a/（1240+b）…③
②③を a について解きます。
a=400（690+b）…④
a=200（1240+b）…⑤
④に⑤を代入し計算します。
400（690+b）=200（1240+b）
276000+400b=248000+200b
276000-248000=200b-400b
28000=-200b
b=-140…⑥
②に⑥を代入し計算します。
a=400（690-140）
a=220000…⑦
⑥⑦から、電圧と距離の関係は次式になるのが分かります。
y=220000/（x-140）

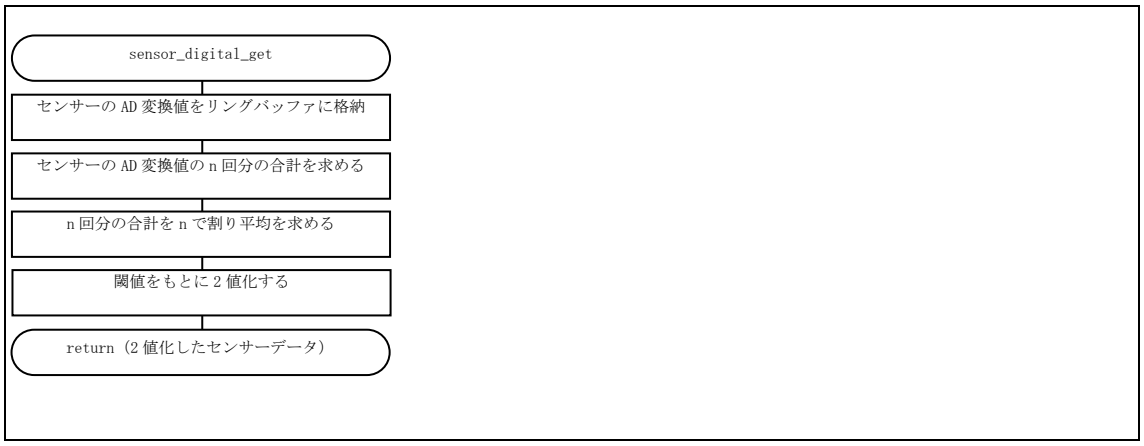
フローチャート

6.4 sensor_digital_get 関数

sensor_digital_get 関数は、センサーのアナログ値を 2 値化したデータを返します。
センサーのデータは 8 ビットで、下記の配置になります。

0	0	0	左	左中	中央	右中	右

フローチャート



7. steering.c

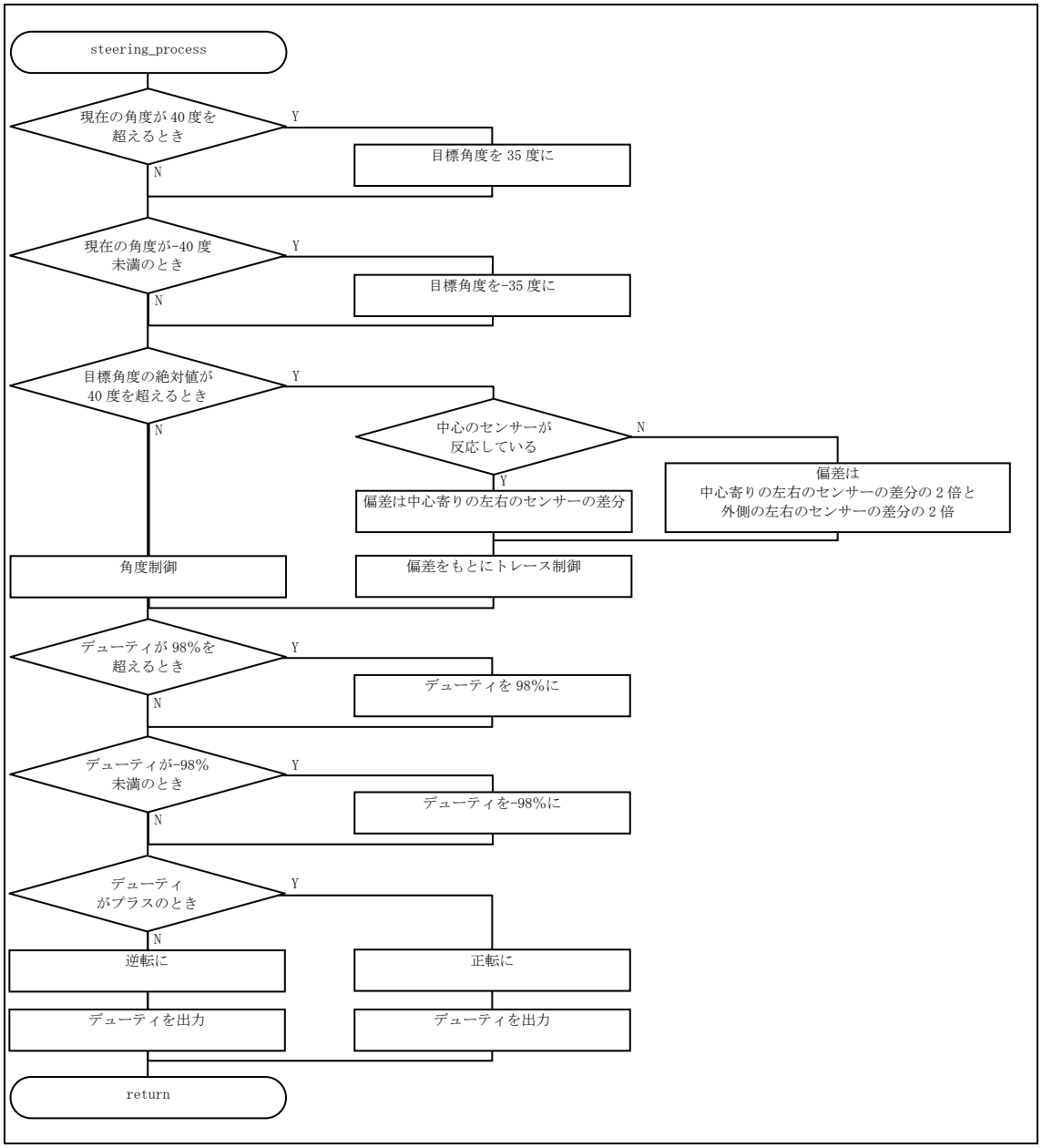
steering.c は、ステアリングモーターのトレース制御と、角度制御を行います。

void steering_process(void)	角度制御またはライン追従制御
void steering_angle_put(signed short angle)	ステアリングの目標角度設定

7.1 steering_process 関数

steering_process 関数は、1ms ごとに割り込みで呼び出されます。
角度制御またはライン追従制御を行います。

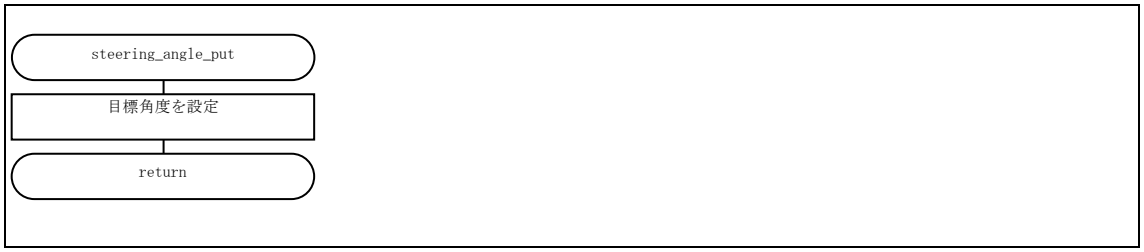
フローチャート



7.2 steering_angle_put 関数

steering_angle_put 関数は、角度制御の目標角度［度］を設定します。
40 度を超える値に設定した場合は、トレース制御になります。

フローチャート



8. mini_mcr.c

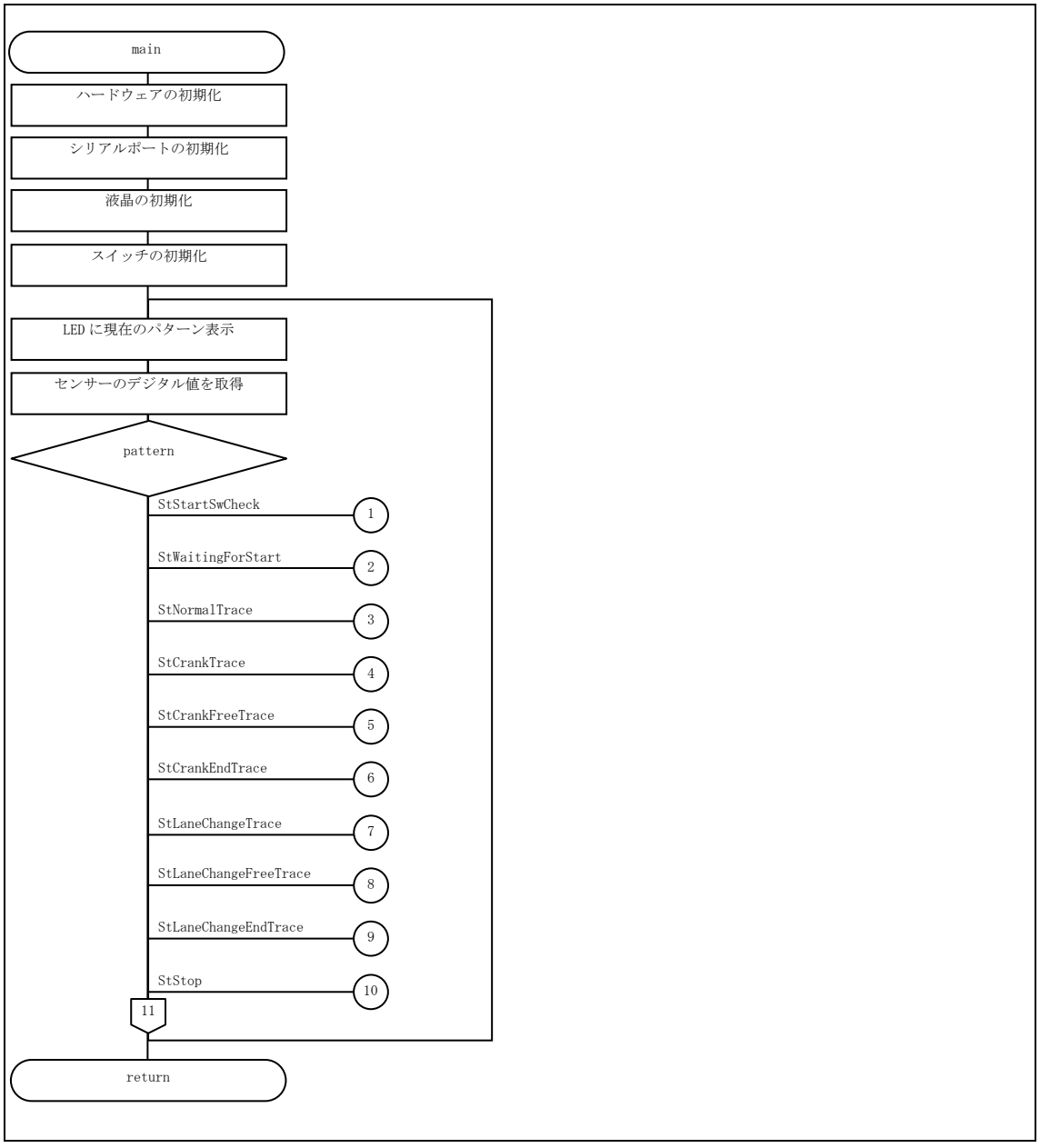
main_mcr.c は、走行プログラム、割り込みプログラム、ハードウェアの初期化が記述されています。

void main(void)	走行プログラム
void init(void)	周辺モジュールの初期化
void intTRBIC(void)	1ms タイマー割り込み
void intTRD1IC(void)	trd1 オーバーフロー割り込み
void timer(unsigned long data1) ミニマイコンカー製作キット Ver.2　C 言語走行プログラム解説マニュアルで解説しているため、本書では解説しません。	
unsigned char dipsw(void) ミニマイコンカー製作キット Ver.2　C 言語走行プログラム解説マニュアルで解説しているため、本書では解説しません。	
void ayc(unsigned short duty ,unsigned short speed)	各車輪の駆動分配

8.1 main 関数

main 関数は、走行制御を行います。

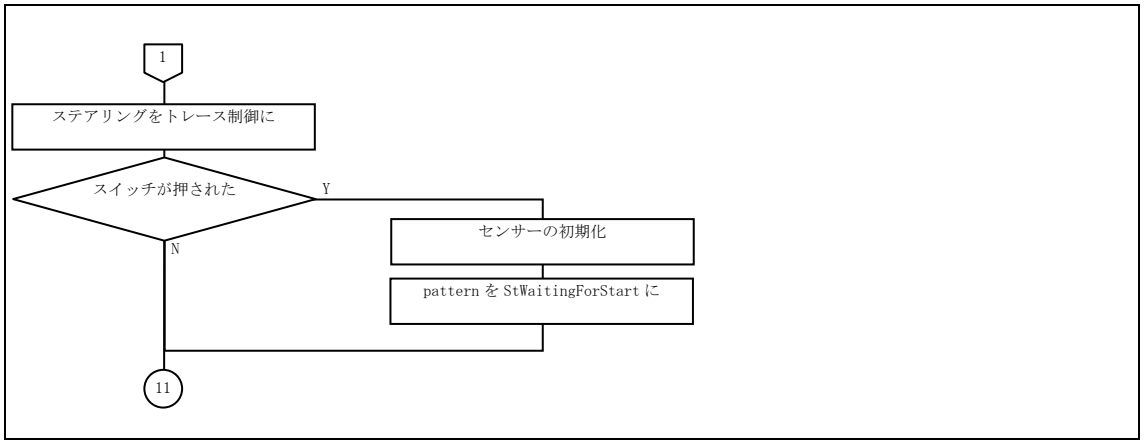
フローチャート



8.1.1 StStartSwCheck パターン

StStartSwCheck パターンは、基板上の SW1 の入力待ちを行います。SW1 が押されると、PSD によるスタートバー開閉検出に移行します。

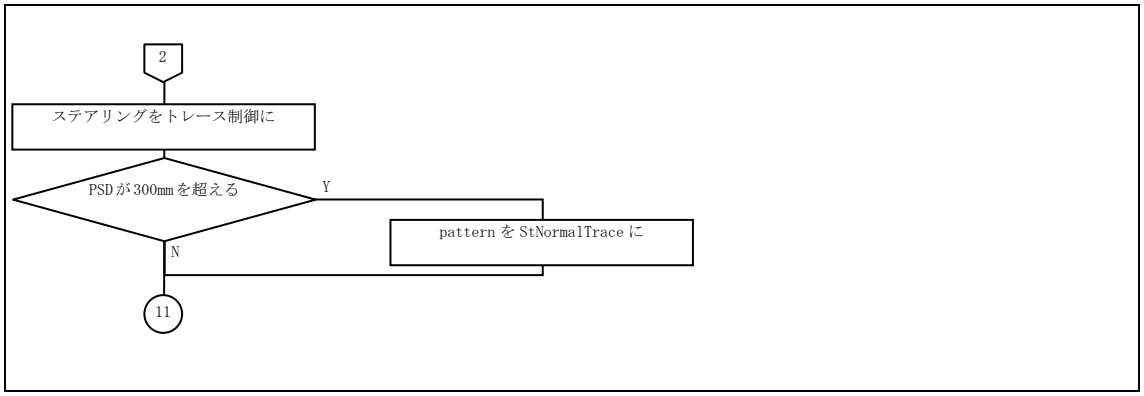
フローチャート



8.1.2 StWaitingForStart パターン

StWaitingForStart パターンは、PSD によるスタートバー開閉検出を行います。PSD の検出距離が 300mm を超えた場合、通常のトレース制御に移行します。

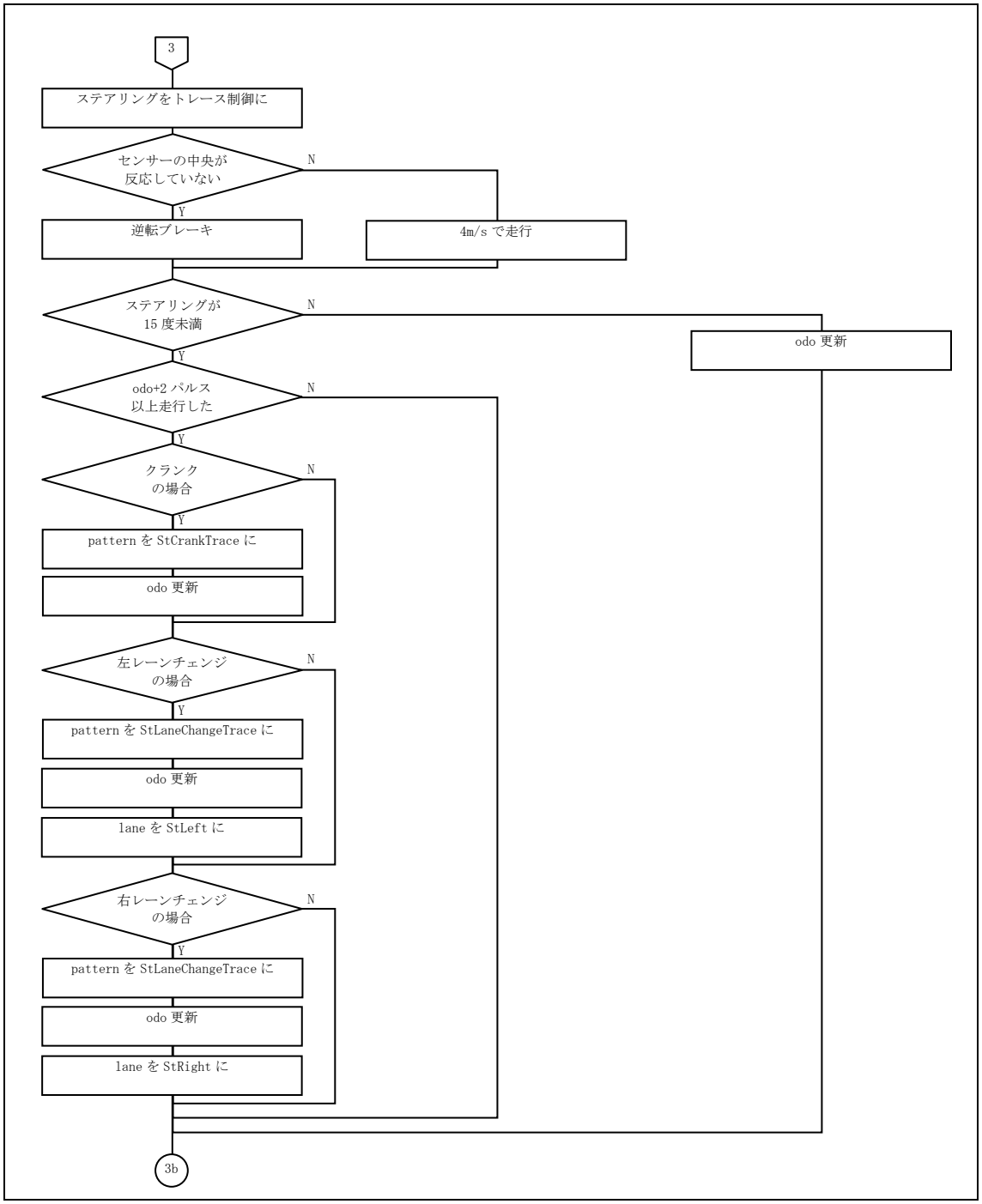
フローチャート



8.1.3 StNormalTrace パターン

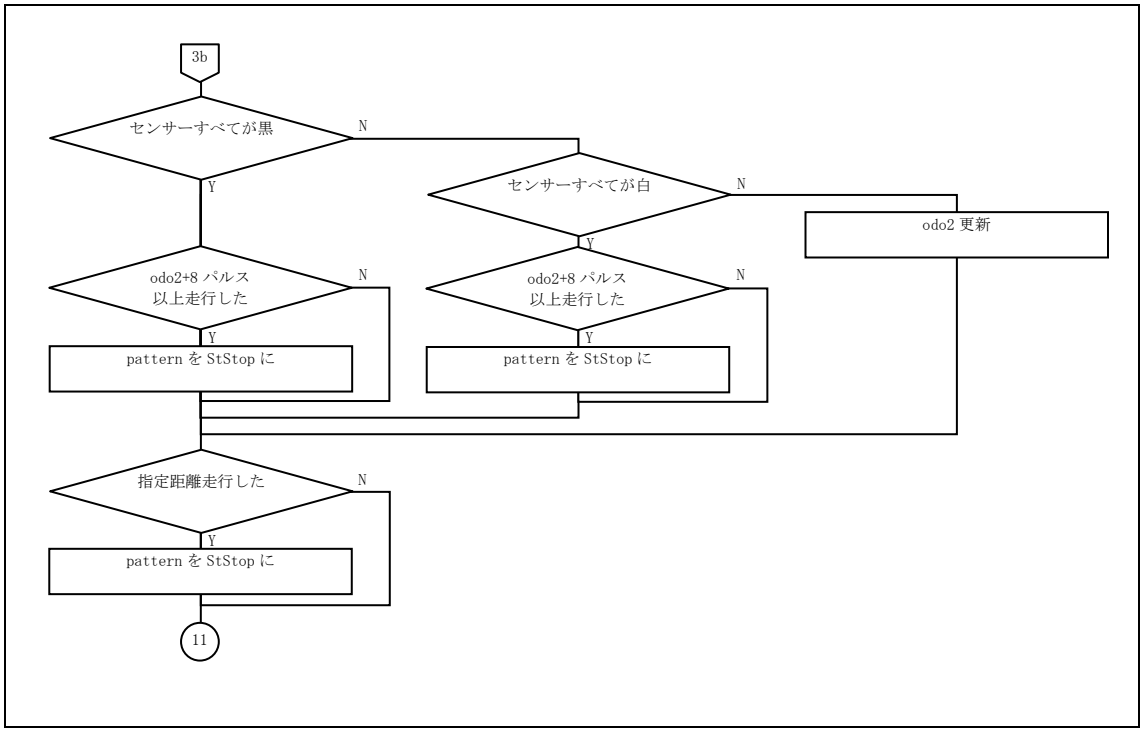
StNormalTrace パターンは、通常のトレース制御を行います。
ステアリングが 15 度未満の状態になってから、2 パルス（タイヤ 1 回転）走行した場合に、クランク、レーンチェンジの検出を開始します。

フローチャート



コースアウト時に停止するように、すべてのセンサーが白か黒の状態で、8 パルス（タイヤ 4 回転）走行した場合に、停止に移行します。また、指定距離を走行した場合にも、停止に移行します。

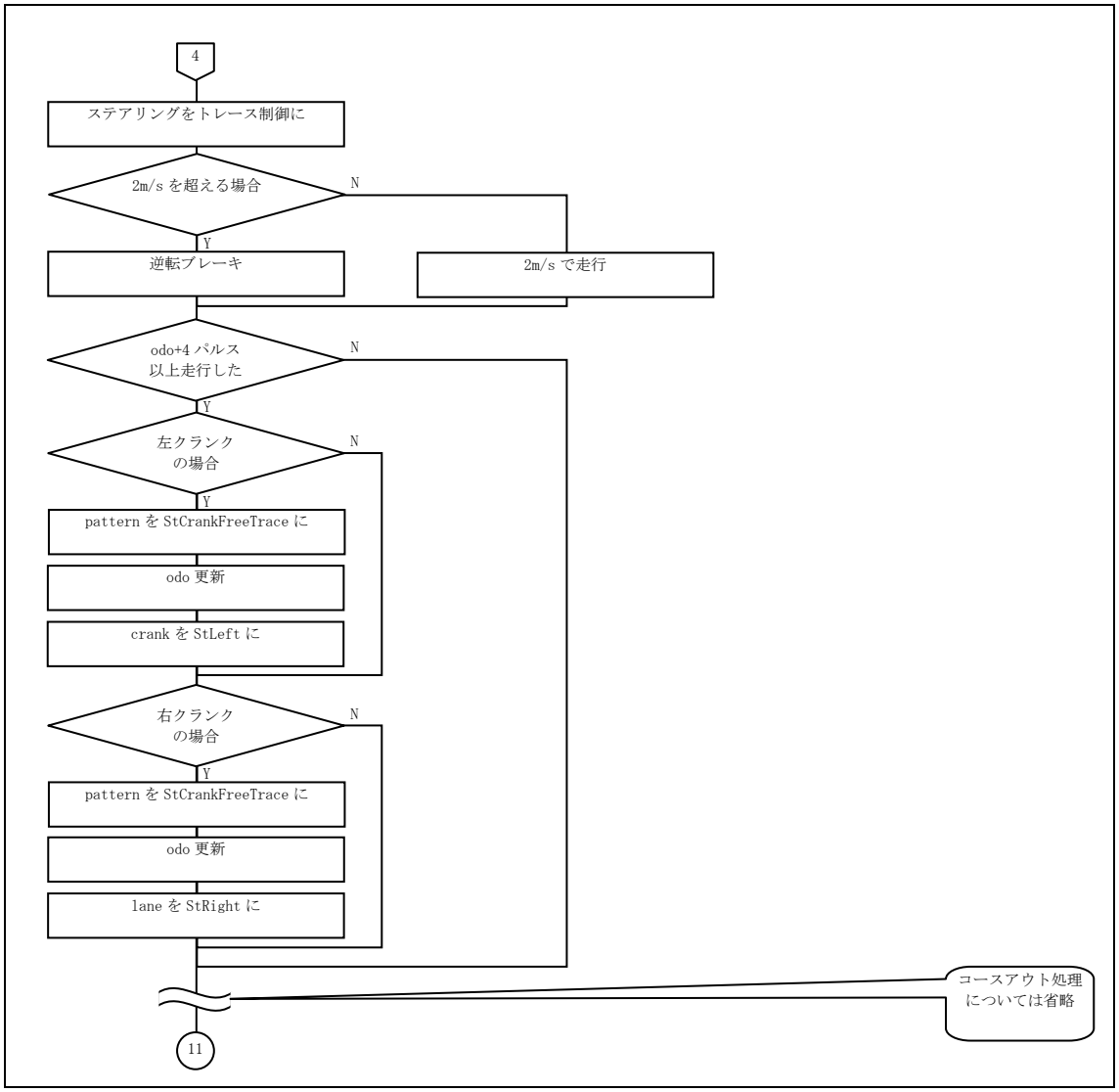
フローチャート



8.1.4 StCrankTrace パターン

クランクの白線を検出してから、曲がり角までの走行を行います。

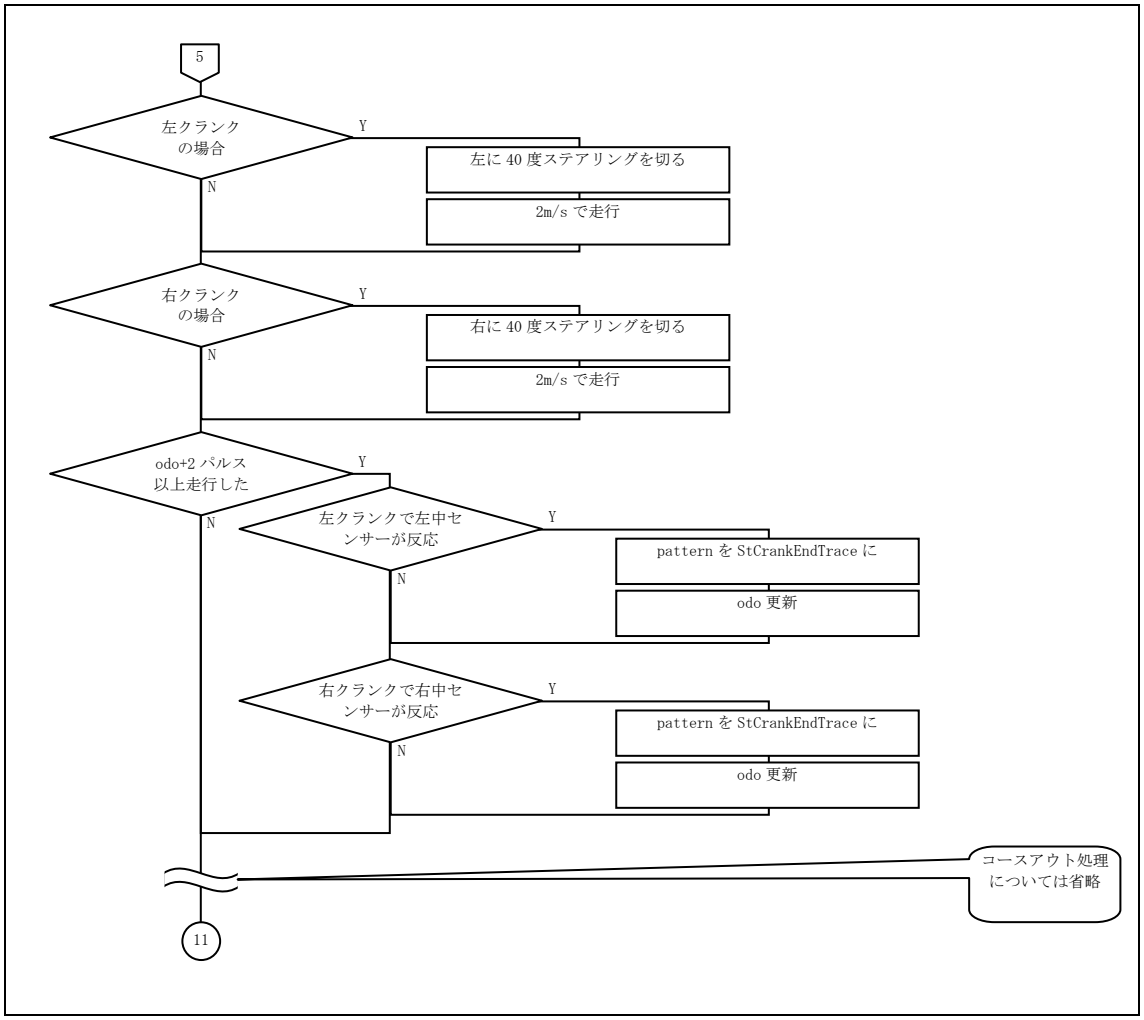
フローチャート



8.1.5　StCrankFreeTrace パターン

曲がり角から一定距離の走行を行います。

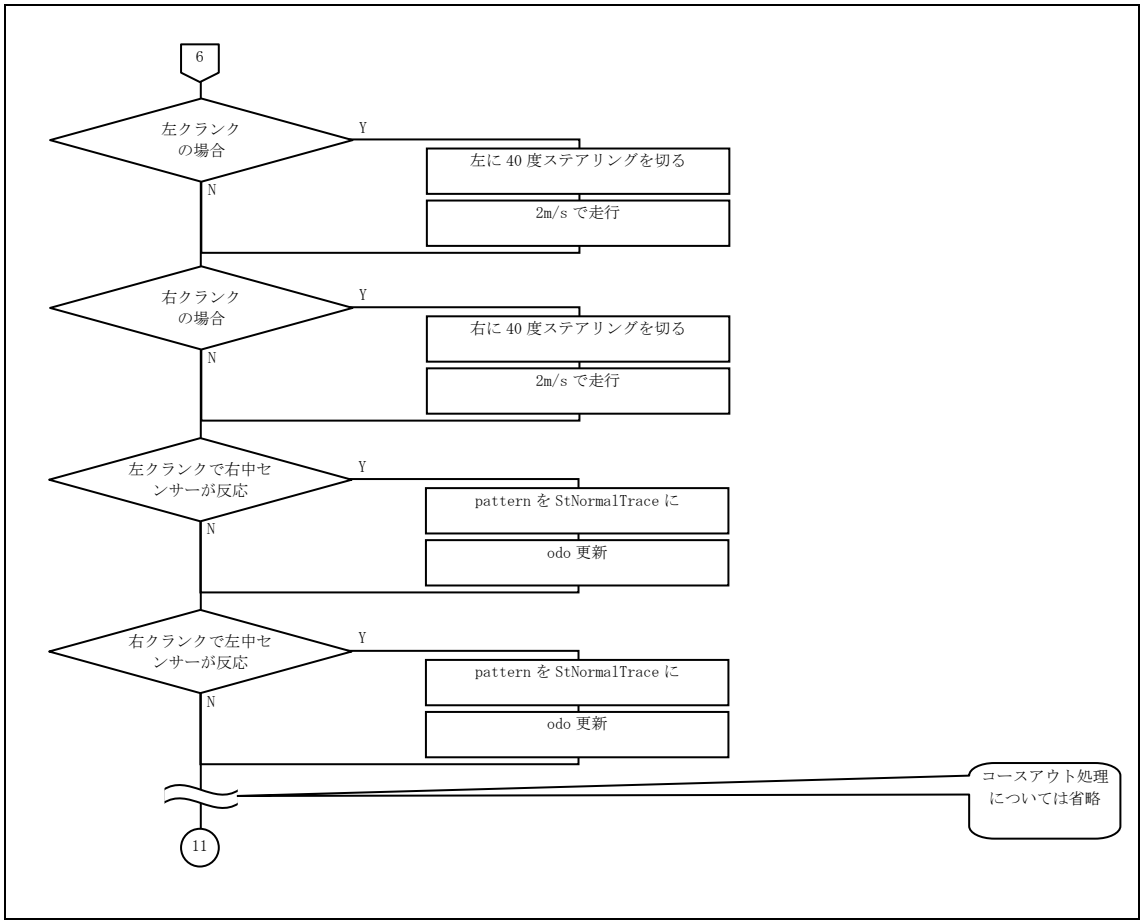
フローチャート



8.1.6　StCrankEndTrace パターン

クランクの処理から通常のトレース制御に戻るまでの走行を行います。

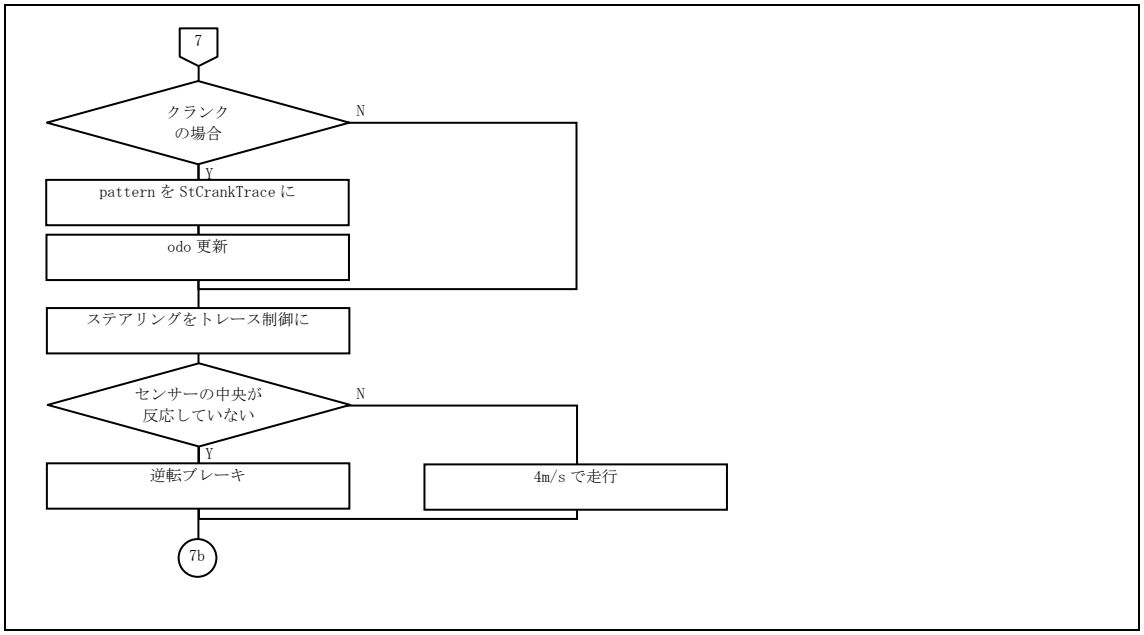
フローチャート

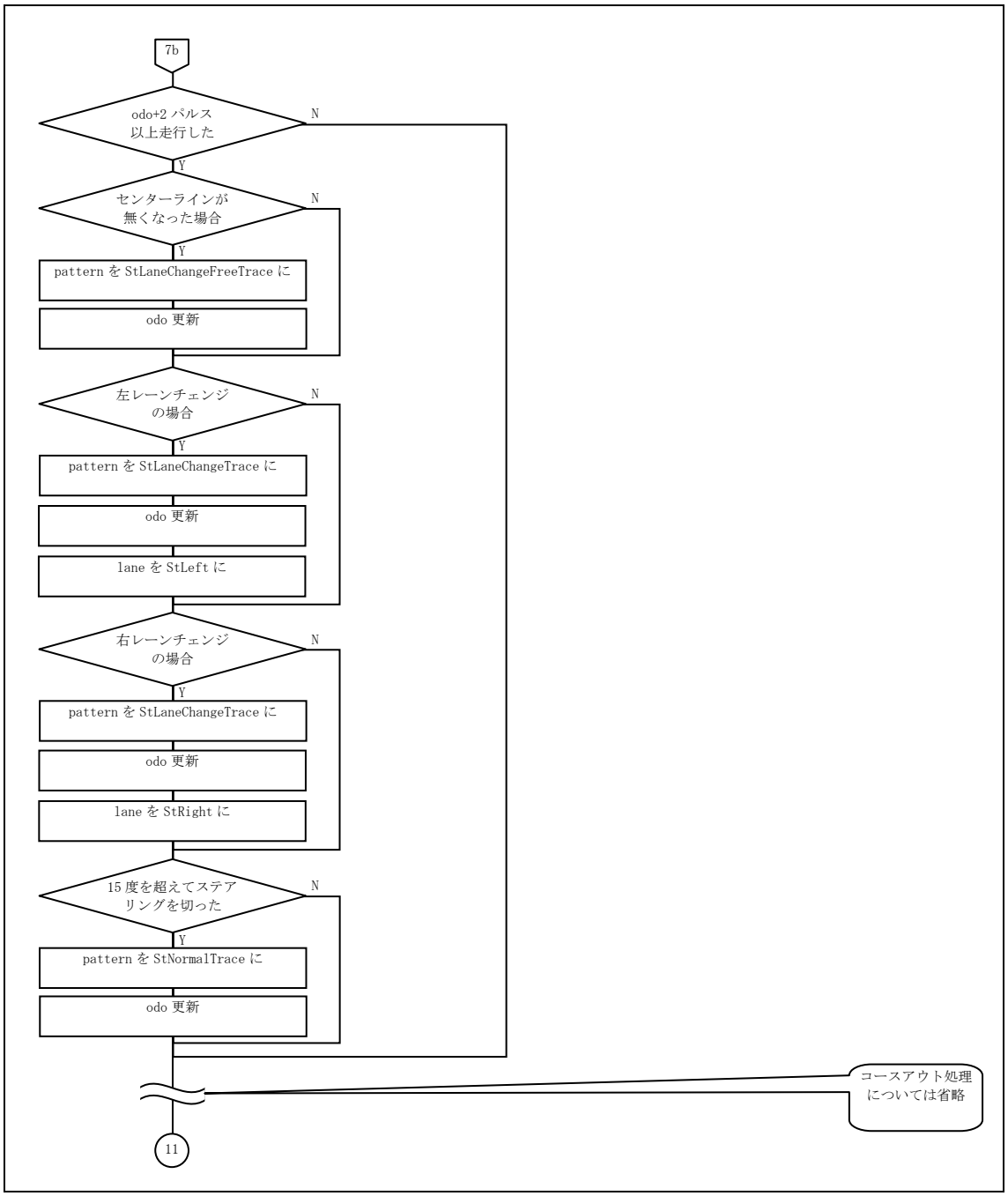


8.1.7　StLaneChangeTrace パターン

レーンチェンジの白線を検出してから、直線が途切れるまでの走行を行います。
ここには、コーナー進入時やクランクの白線に斜めに進入した場合に、レーンチェンジの白線と同様のセンサーの反応をして、誤ったパターンに遷移する場合があります。
その対策として、クランクの白線の検出、レーンチェンジの白線の検出、ステアリングを 15 度を超えて切った場合の通常トレースへの復帰する処理を行っています。

フローチャート

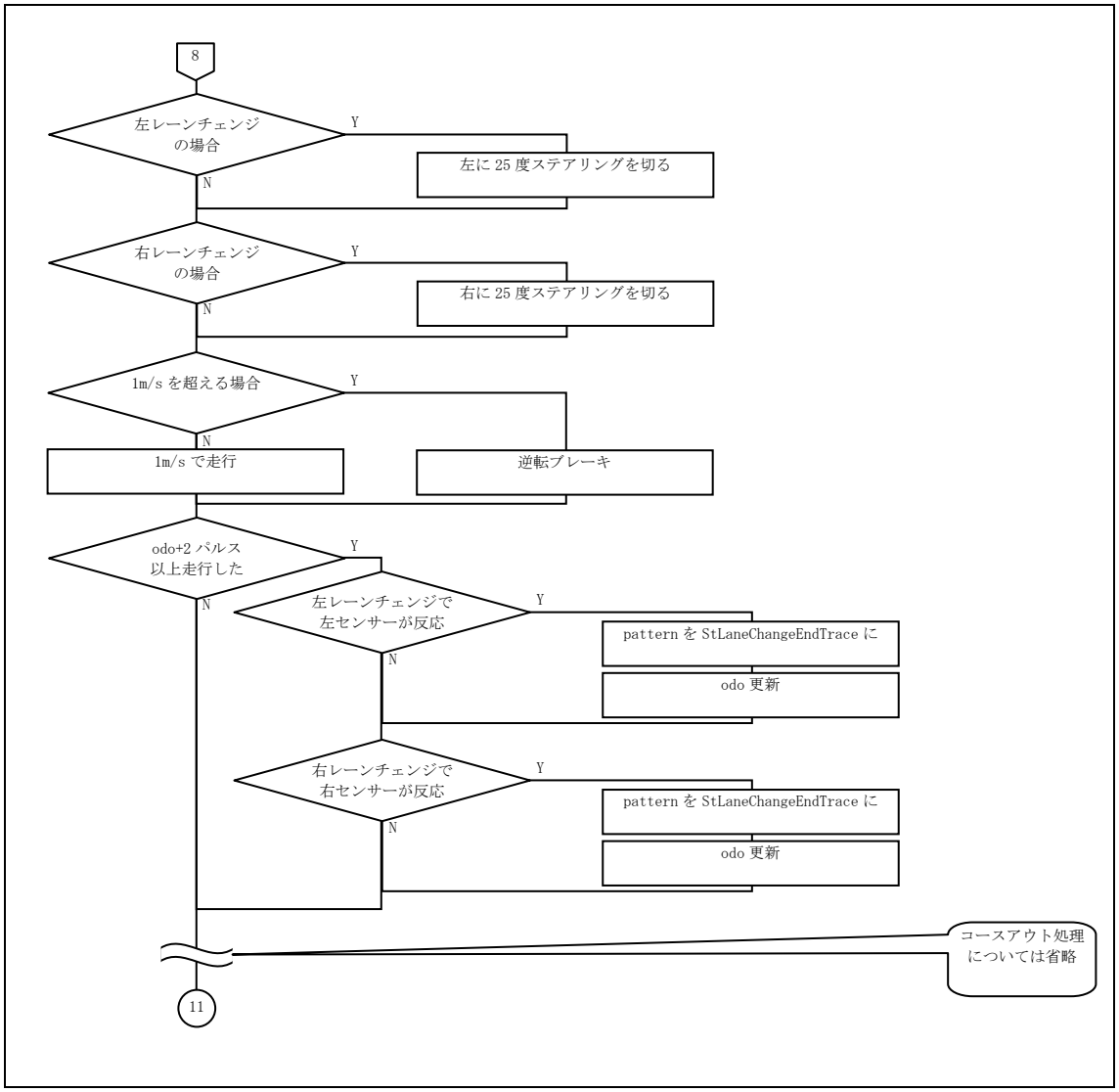



7b
odo+2 パルス
以上走行した
N
Y
センターラインが
無くなった場合
N
Y
pattern を StLaneChangeFreeTrace に
odo 更新
左レーンチェンジ
の場合
N
Y
pattern を StLaneChangeTrace に
odo 更新
lane を StLeft に
右レーンチェンジ
の場合
N
Y
pattern を StLaneChangeTrace に
odo 更新
lane を StRight に
15 度を超えてステア
リングを切った
N
Y
pattern を StNormalTrace に
odo 更新
コースアウト処理
については省略
11

8.1.8　StLaneChangeFreeTrace パターン

直線が途切れてから一定距離の走行を行います。

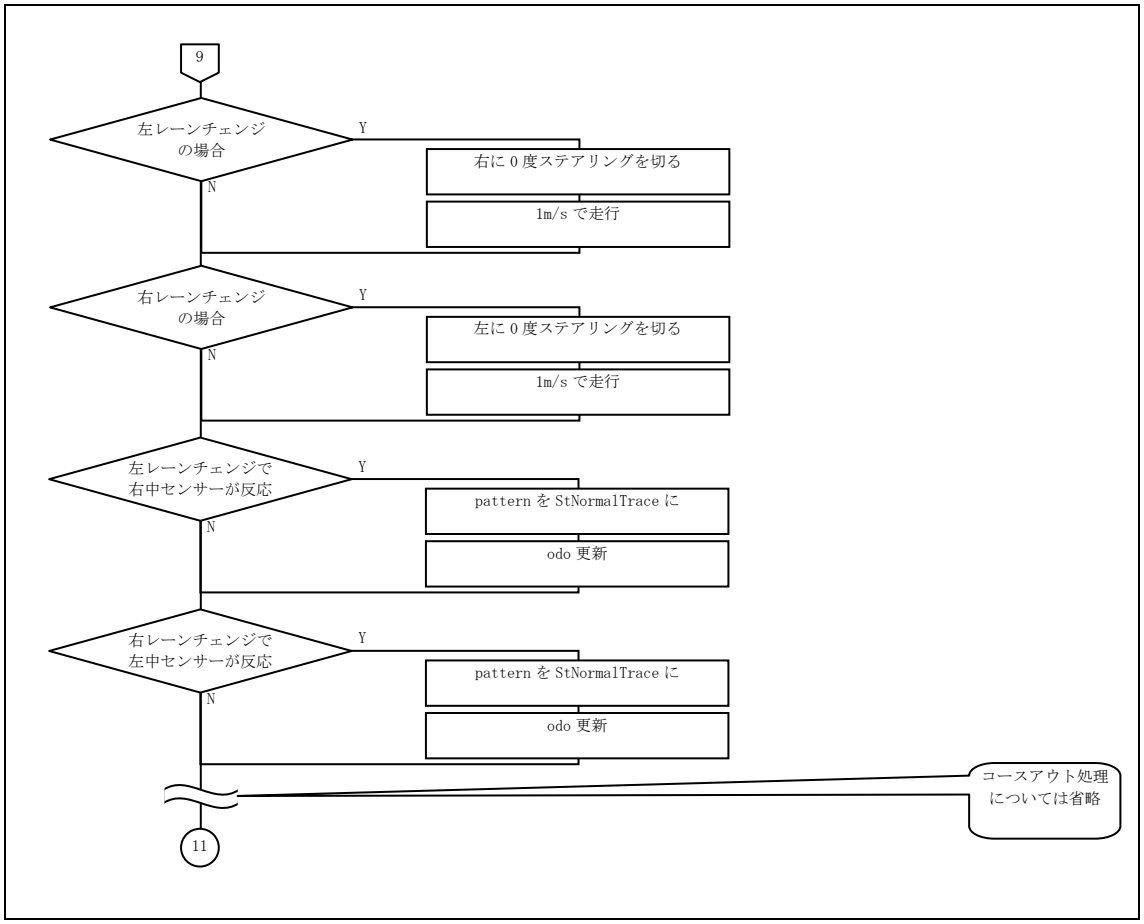
フローチャート



8.1.9　StLaneChangeEndTrace パターン

レーンチェンジ処理から通常のトレース制御に戻るまでの走行を行います。

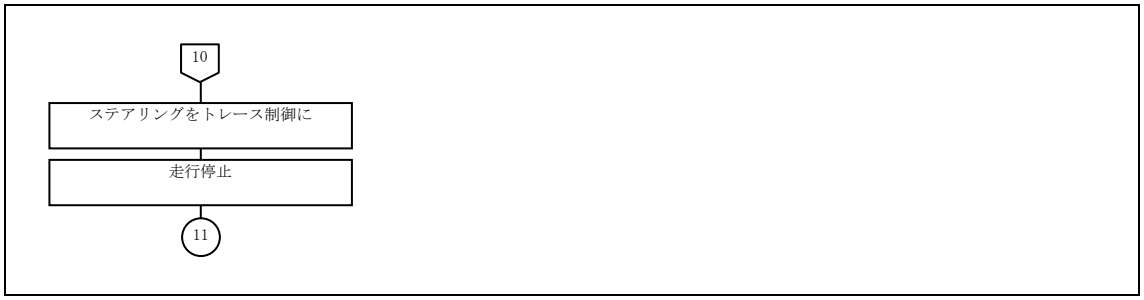
フローチャート



8.1.10 StStop パターン

走行を停止します。
ステアリングの制御は続けるようにしています。

フローチャート



8.2 init 関数

ミニマイコンカーC 言語走行プログラムからの変更点は、

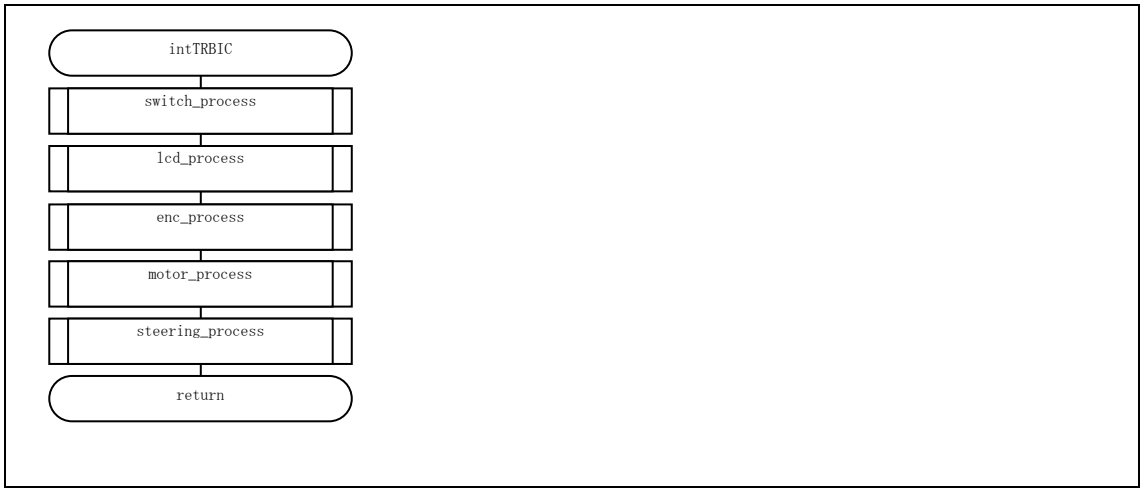
・入出力設定の変更
・PWM の割り当て場所の変更
・インプットキャプチャの設定

です。詳細についてはプログラムをご確認ください。

8.3 intTRBIC 関数

intTRBIC 関数は、1ms ごとに割り込みで実行されます。
定時刻で実行する必要がある関数を呼び出します。

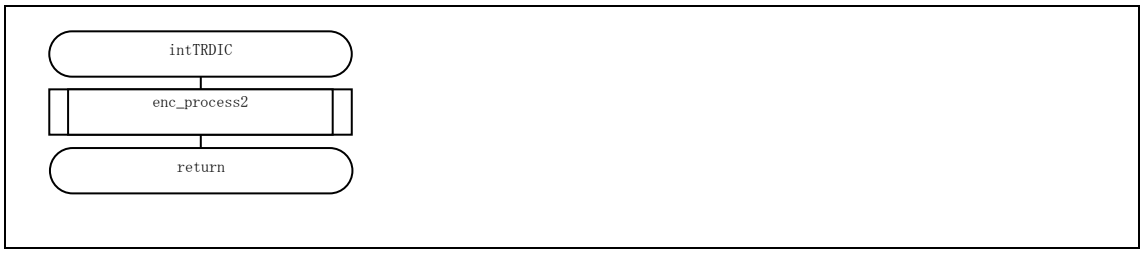
フローチャート



8.4 intTRD1IC 関数

intTRD1IC 関数は、trd1 オーバーフローで実行されます。

フローチャート



8.5　ayc 関数

ayc 関数は、ステアリングの角度、角速度に応じて、4 輪のデューティの分配を制御する関数です。

角速度は、角度の微分により求めますが、計算を簡略化するため、Δx÷Δt の計算は行わず、角度の差分を使用します。角度の差分を求める周期は、この関数を呼び出す周期に依存します。

回転中心から左右タイヤの距離により、回転数の差が発生するので、通常はステアリング角度から三角関数を使用して回転数の差の計算を行いますが、この関数では角度に応じて直線的に変化させています。

```
lf = duty - ( angle - angle_old ) * 100 - angle * 2 - abs( ( angle - angle_old ) ) * 0 - abs( angle ) * 0;
rf = duty + ( angle - angle_old ) * 100 + angle * 2 - abs( ( angle - angle_old ) ) * 0 - abs( angle ) * 0;
lr = duty - ( angle - angle_old ) * 100 - angle * 4 - abs( ( angle - angle_old ) ) * 0 - abs( angle ) * 0;
rr = duty + ( angle - angle_old ) * 100 + angle * 4 - abs( ( angle - angle_old ) ) * 0 - abs( angle ) * 0;
```

黄色：ステアリング速度に応じた、左右輪のデューティ差を計算します。
緑色：ステアリング角度に応じた、左右輪のデューティ差を計算します。
水色：ステアリング速度に応じた、ブレーキ量を計算します。
紫色：ステアリング角度に応じた、ブレーキ量を計算します。

フローチャート