マイコンカーラリーキット

マイコンカー
動作確認マニュアル
Vol.3 版

～スタートバー検出センサ基板　対応バージョン～

第 2.01 版
2006.07.06
ジャパンマイコンカーラリー実行委員会

注 意 事 項

著作権

・本マニュアルに関する著作権はジャパンマイコンカーラリー実行委員会に帰属します。
・本マニュアルは著作権法及び、国際著作権条約により保護されています。

禁止事項

ユーザーは以下の事を行う事は出来ません。
・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、複製等を行う事
・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事
・本マニュアルの一部又は全部を改変、除去する事
・本マニュアルを無許可で翻訳する事
・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、文章によるジャパンマイコンカーラリー実行委員会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、ジャパンマイコンカーラリー実行委員会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、ジャパンマイコンカーラリー実行委員会は、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、事前にマイコンカー公式ホームページ(http://www.mcr.gr.jp/)などを通じて公開される情報に常にご注意ください。

連絡先

ルネサステクノロジ　マイコンカーラリー事務局
〒162-0824　東京都新宿区揚場町 2-1　軽子坂MNビル
TEL (03)-3266-8510
E-mail:official@mcr.gr.jp

目　次

1. 概要 1

2. キットの構成 2

3. 動作確認用プログラムのインストール 3

3.1. ワーススペース「kit06」を開く 3
3.2. プロジェクトの変更 4
3.3. テストプログラムの書き込み 5

4. 動作テスト 7

4.1. 動作テスト一覧 7
4.2. LED のテスト 8
4.3. タクトスイッチのテスト 8
4.4. サーボのテスト 8
4.5. スタートバー検出センサ基板のテスト 10
4.6. 右モータのテスト 11
4.7. 左モータのテスト 11
4.8. センサのテスト 12
4.9. 直進テスト 14
4.10. テスト終了 14

5. プログラムソース 15

5.1. 「kit06test.c」のプログラム内容 15
5.2. 「kit06teststart.src」のプログラム内容 20

1. 概要

本マニュアルは、

●センサ基板製作マニュアル
●モータドライブ基板(Vol.3)製作マニュアル
●本体組み立て製作マニュアル(Vol.3 版)
●スタートバー検出センサ基板製作マニュアル

で製作、組み立てたマイコンカーの動作確認方法を説明するマニュアルです。
動作確認をするためには、下記の手順で行います。

1. パソコンにルネサス統合開発環境をインストールします
既にインストールしている場合、必要ありません。していない場合は、「ルネサス統合開発環境操作マニュアル」を参照して、インストールしてください。

2. サンプルプログラム（動作テストプログラム）をインストールします
既にインストールしている場合、必要ありません。していない場合は、「ルネサス統合開発環境操作マニュアル」を参照して、インストールしてください。

3. プログラムをマイコンカーの CPU ボードに書き込みます
これから行います。

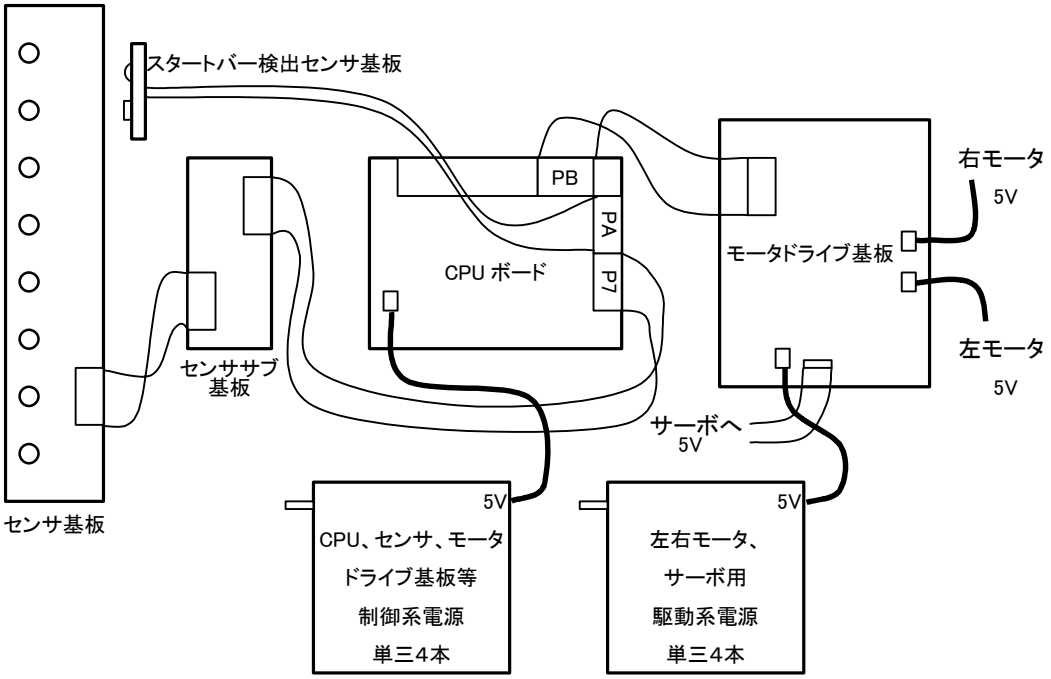
4. 本マニュアルに従って動作テストします
これから行います。

5. サーボセンタと最大切れ角を調整します
プログラム解説マニュアル「13. サーボセンタと最大切れ角の調整」を参照して、調整してください。

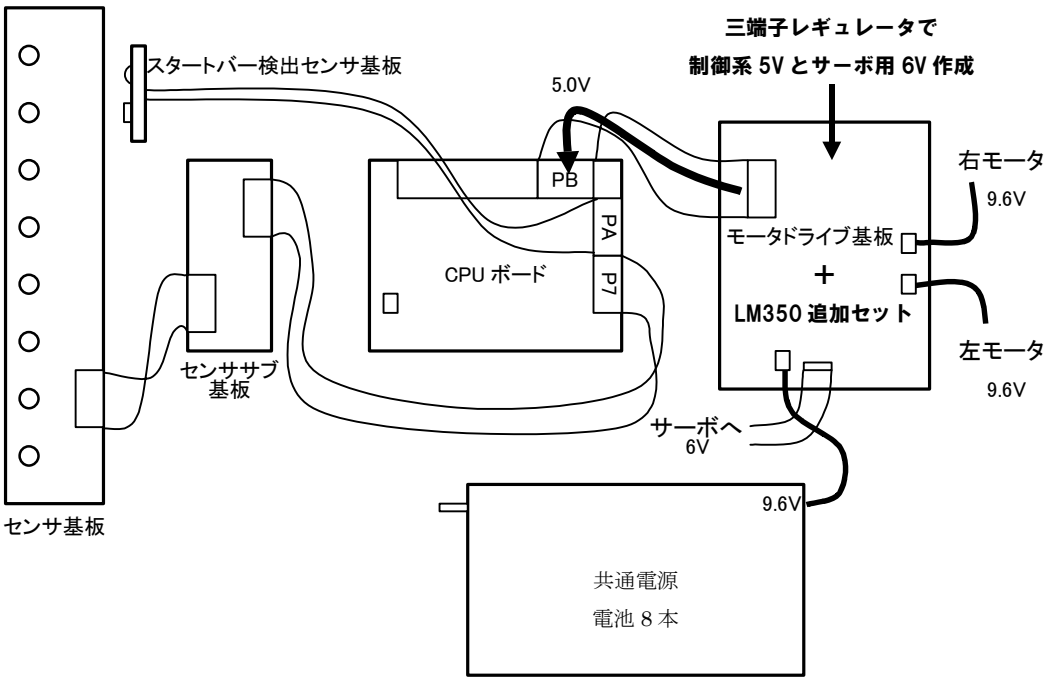
6. 走行プログラムを書き込みます
プログラム解説マニュアルを参照しながら、プロジェクト「kit06」の kit06.mot ファイルを CPU ボードに書き込み、コースを走らせてみてください。

2. キットの構成

下図のようなマイコンカーをテストすることが出来ます。



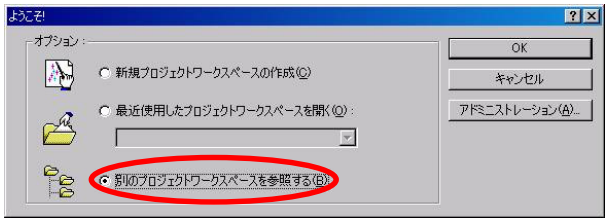
モータドライブ基板に、「LM350 追加セット」を追加し、電源電圧を上げても同じくテスト出来ます。



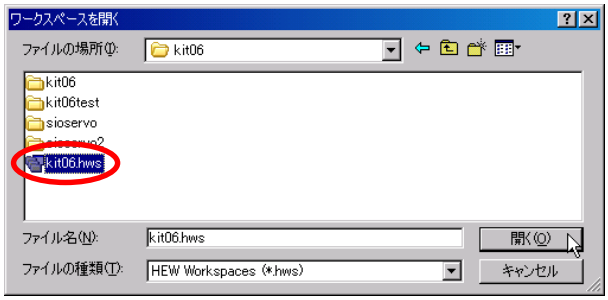
3. 動作確認用プログラムの書き込み

3.1. ワーススペース「kit06」を開く

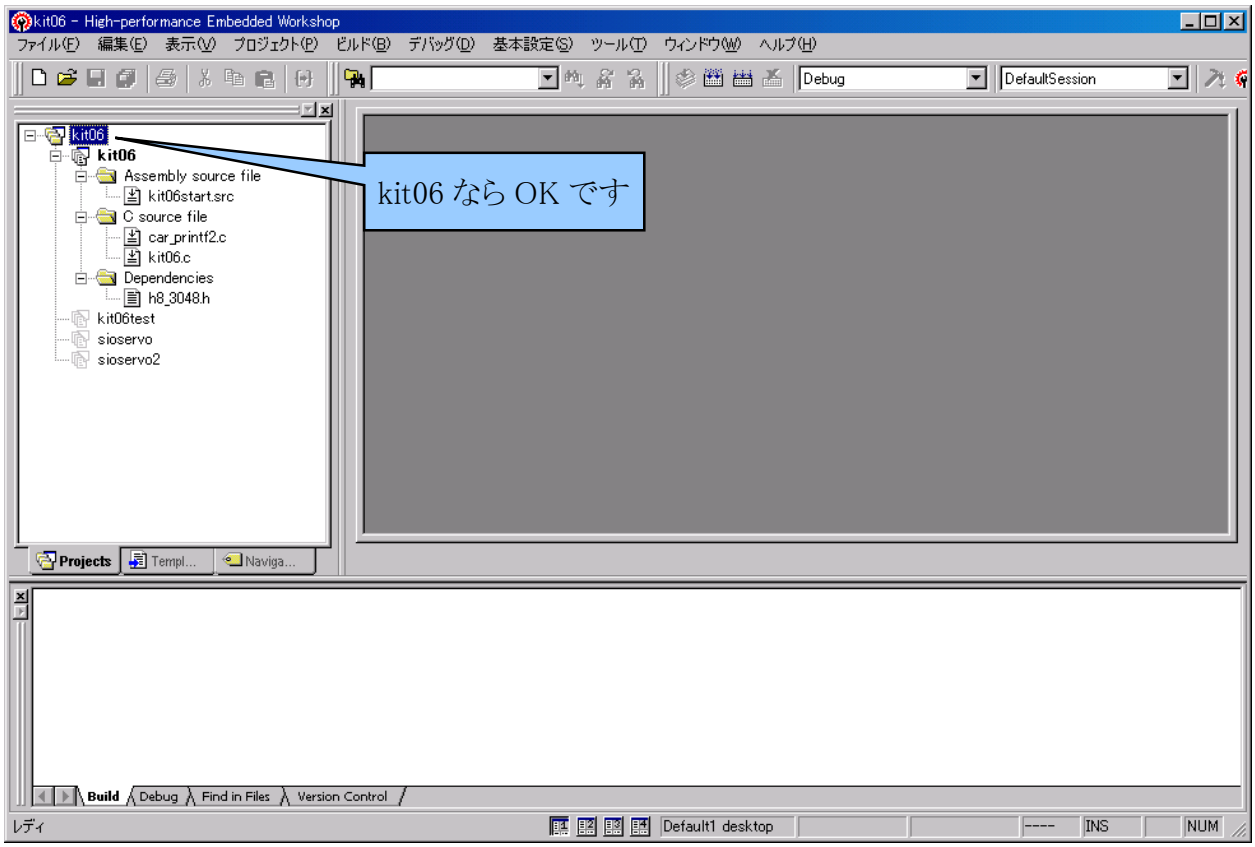
ルネサス統合開発環境を実行します。
もし、ルネサス統合開発環境がインストールされていない場合、「ルネサス統合開発環境　操作マニュアル」に従ってインストールしてください。



「別のプロジェクトワークスペースを参照する」を選択します。

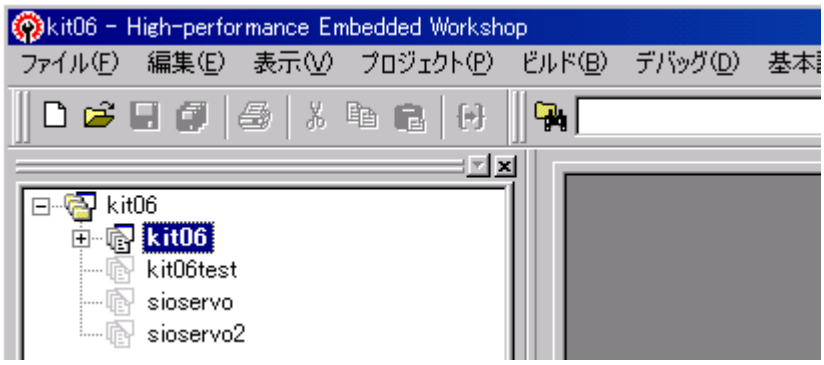


Cドライブ→Workspace→kit06 の「kit06.hws」を選択します。このファイルがない場合、ルネサス統合開発環境　操作マニュアル「3.2. プログラムのインストール」に従ってインストールしてください。



kit06 というワークスペースが開かれます。

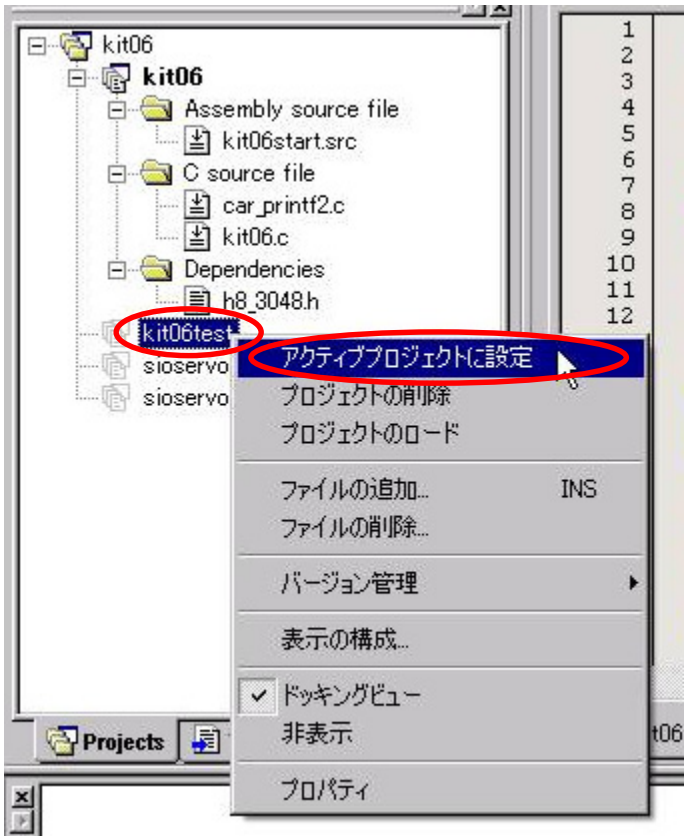
3.2. プロジェクトの変更



ワークスペース「kit06」には、4つのプロジェクトが登録されています。

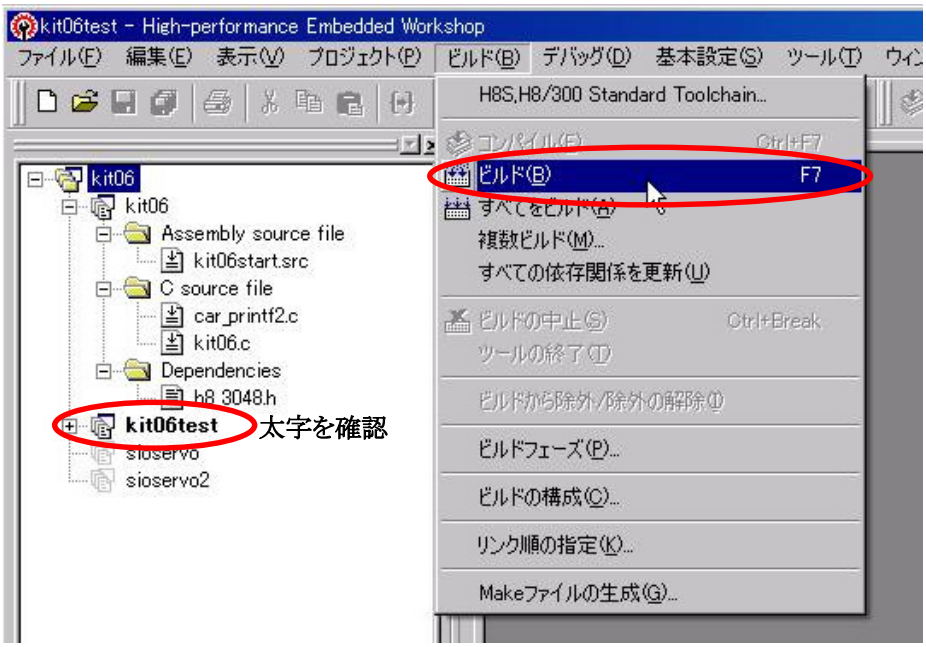
プロジェクト名	内容
kit06	マイコンカー走行プログラムです。
kit06test	**製作したマイコンカーのモータドライブ基板やセンサ基板、スタートバー検出センサ基板が正しく動作するかテストします。**
sioservo	サーボのセンタを調整するプログラムです。
sioservo2	サーボの最大切れ角を見つけるためのプログラムです。

「kit06test」をアクティブプロジェクトに設定します。

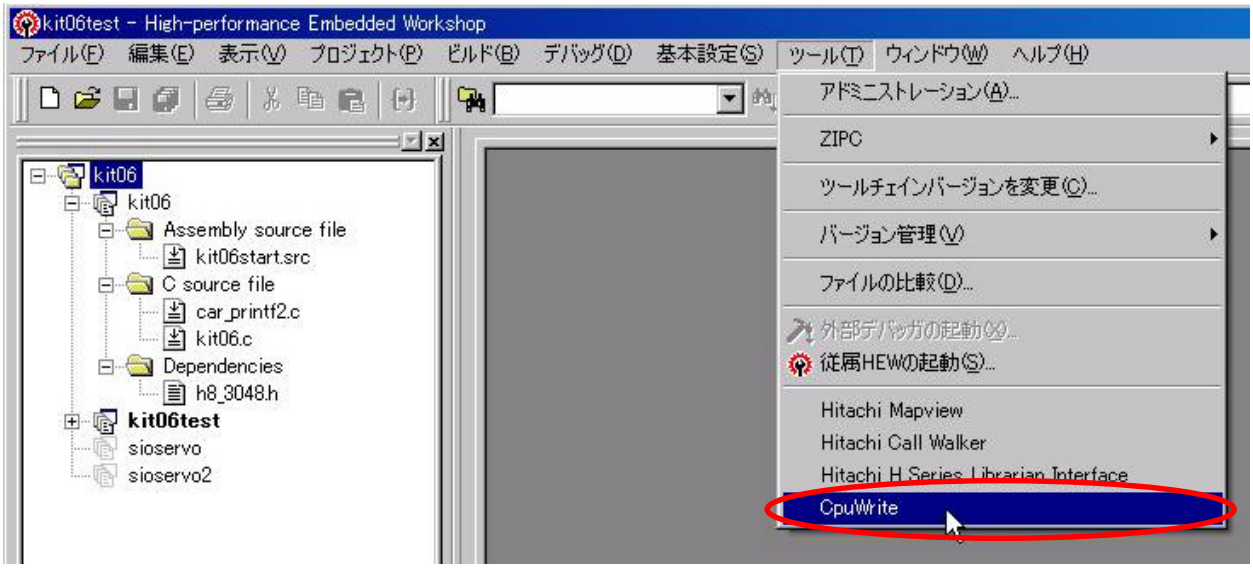


「kit06test」上で右クリックします。
「アクティブプロジェクトに設定」を選択します。

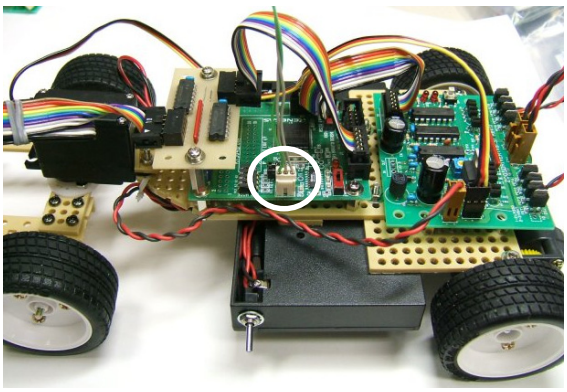
3.3. テストプログラムの書き込み

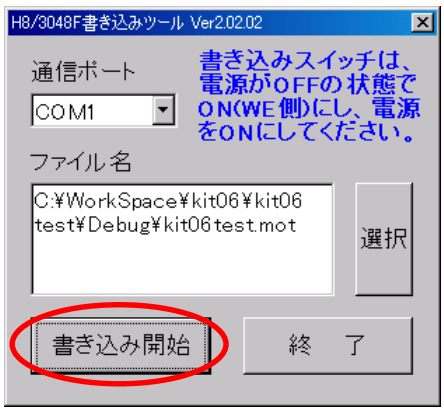


再度、「kit06test」が有効(太字)になっていることを確認して、「ビルド→ビルド」を実行します。



次に、「ツール→CpuWrite」で書き込みプログラムを起動します。もし、CpuWrite コマンドが無い場合は、ルネサス統合開発環境操作マニュアルの「3.3. 書き込みソフトをルネサス統合開発環境に登録」を参照して、登録してください。





通信ケーブルをパソコンと CPU ボードの 3 ピンコネクタに接続して、スイッチを書き込み(FWE)側にし、電源を ON にしてください。

書き込み開始ボタンをクリックして書き込みを開始します。正常に書き込まれない場合は、

1. 通信ケーブルの接続間違え、断線
2. パソコン側の問題。通信ポートが無効になっている、ポート番号が違っている、別の機器(赤外線ポート等)がすでに通信ポートを使用している等
3. CPU ボード側の問題。書き込みスイッチが正常でない、RXD1 切り換えスイッチが 3P コネクタ側になっていない、CPU が書き込み制限回数を超えている、CPU 電源電圧が 4.5～5.5V ではない　等

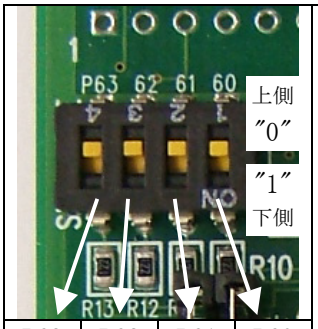
の理由が考えられます。これらの原因を解決して再度書き込みを行ってください。

順調に書き込まれたならマイコンカーの電源を OFF にして、書き込みスイッチを通常側に戻します。
次へ進んでください。

4. 動作テスト

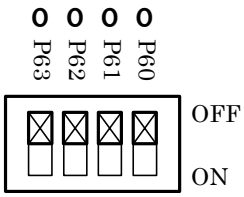
4.1. 動作テスト一覧

CPU ボードのディップスイッチの状態を変更することにより、マイコンカーのどの部分をテストするか選択し、動作のテストを行います。



P63	P62	P61	P60	内容
0	0	0	0	LED のテストをします。LED が 0.5 秒間隔で交互に点灯します。
0	0	0	1	プッシュスイッチのテストをします。 スイッチ OFF で LED0 が点灯、スイッチ ON で LED1 が点灯します。
0	0	1	0	サーボのテストをします。 サーボが、「0°　→右 30°　→左 30°　の繰り返し」の動作をします。
0	0	1	1	スタートバー検出センサのテストをします。スタートバー検出センサの反応が　OFF で LED0 が点灯、ON で LED1 が点灯します。
0	1	0	0	右モータのテストをします。 「正転→ブレーキ」動作を繰り返します。
0	1	0	1	右モータのテストをします。 「逆転→ブレーキ」動作を繰り返します。
0	1	1	0	左モータのテストをします。 「正転→ブレーキ」動作を繰り返します。
0	1	1	1	左モータのテストをします。 「逆転→ブレーキ」動作を繰り返します。
1	0	0	0	コースのセンサテストをします。 センサ bit1,0 の状態を LED1,0 に出力します。
1	0	0	1	コースのセンサテストをします。 センサ bit3,2 の状態を LED1,0 に出力します。
1	0	1	0	コースのセンサテストをします。 センサ bit5,4 の状態を LED1,0 に出力します。
1	0	1	1	コースのセンサテストをします。 センサ bit7,6 の状態を LED1,0 に出力します。
1	1	0	0	直進テストをします。 PWM50%で前進、 2 秒後にストップします。
1	1	0	1	直進テストをします。 PWM50%で前進、 5 秒後にストップします。
1	1	1	0	直進テストをします。 PWM100%で前進、 2 秒後にストップします。
1	1	1	1	直進テストをします。 PWM100%で前進、 5 秒後にストップします。

4.2. LED のテスト

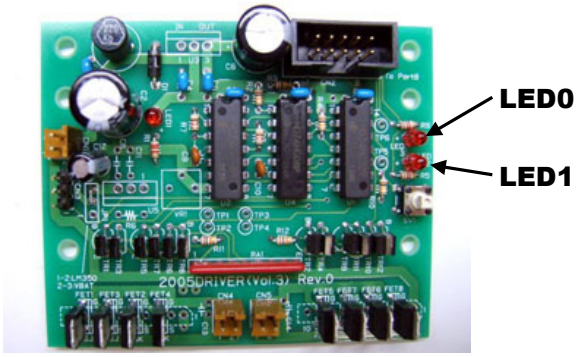


スイッチの状態

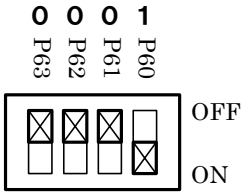
ディップスイッチを”0000”の状態で CPU ボード用電源を ON します。ディップスイッチは、ON が”1”、OFF が”0”です。
モータドライブ基板の LED0 と LED1 が 0.5 秒毎に交互に点灯します。

点灯しない場合は、CPU ボードとモータドライブ基板とを接続するフラットケーブルの不良、LED の半田付け不良、半田ブリッジ(ショート)、LED の向きが逆など考えられます。目視チェック、テスタ等で原因を突き止めてください。

※LED0、LED1 の位置



4.3. タクトスイッチのテスト

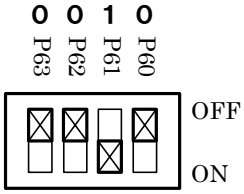


スイッチの状態

ディップスイッチを”0001”の状態で CPU ボード用電源を ON します。
モータドライブ基板上のスイッチが押されていない状態なら LED0 が点灯、押されたら LED1 が点灯します。

LED0 のみしか点灯しない場合は、スイッチまでの回路が半田付け不良、LED1 が付きっぱなしの場合は半田ブリッジが考えられます。目視チェック、テスタ等で原因を突き止めてください。

4.4. サーボのテスト



スイッチの状態

ディップスイッチを”0010”の状態で CPU ボード用電源、モータ用電源を ON します。
サーボが 1 秒毎に、「0度→右30度→左30度」の動作を繰り返します。

サーボが動作しない場合は、サーボまでの回路が半田付け不良、サーボコネクタの向きが逆等が考えられます。また、モータドライブ基板の電源 LED が点灯しているかも確認してください。目視チェック、テスタ等で原因を突き止めてください。

キットに付属ではないサーボに交換したとき、「0度→**左**30度→**右**30度」の動作になる場合、左右の回転が逆なサーボです。その場合、handle 関数内の 417 行を下記のように変更します。

434 行	変更前	ITU4_BRB = SERVO_CENTER - angle * HANDLE_STEP;
	変更後	ITU4_BRB = SERVO_CENTER **+** angle * HANDLE_STEP;

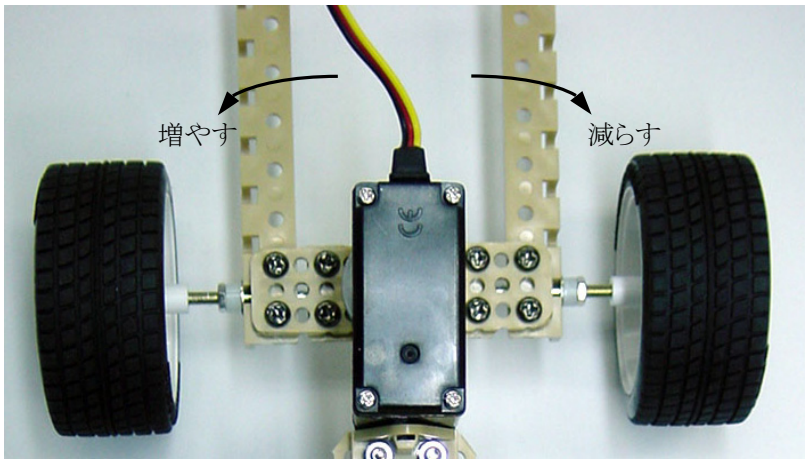
左右が入れ代わり、動作が「0度→右30度→左30度」となります。

●サーボのセンタ調整

ほとんどの場合、電源を ON にしても 0 度のはずなのにまっすぐ向いていないと思います。サーボのセンタ値は、サーボによってそれぞれ違うのでマイコンカー1 台 1 台違う値になります。「kit06test.c」の SERVO_CENTER の「**5000**」という値を変えてまっすぐになるように調整します。26 が約 1 度です。

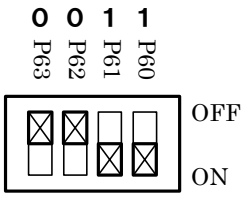
```
53 : #define        SERVO_CENTER     5000     /* サーボのセンタ値          */
```

値を増やせば進行方向に向かって左側、減らせば右側に向きます。



サーボの調整は、プログラム解説マニュアルの「13. サーボセンタと最大切れ角の調整」を参照しながら行うと便利です。

4.5. スタートバー検出センサ基板のテスト

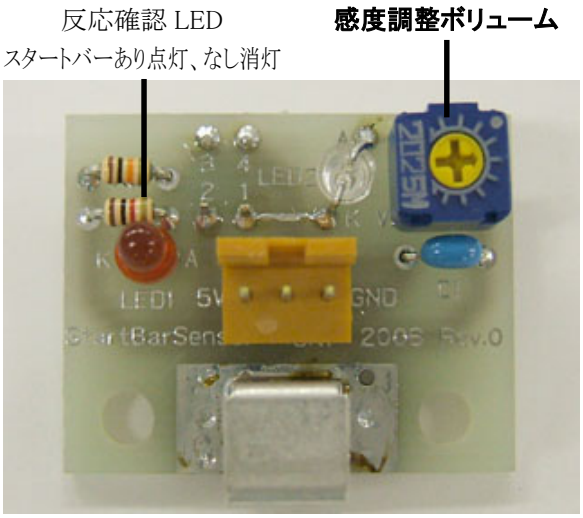


スイッチの状態

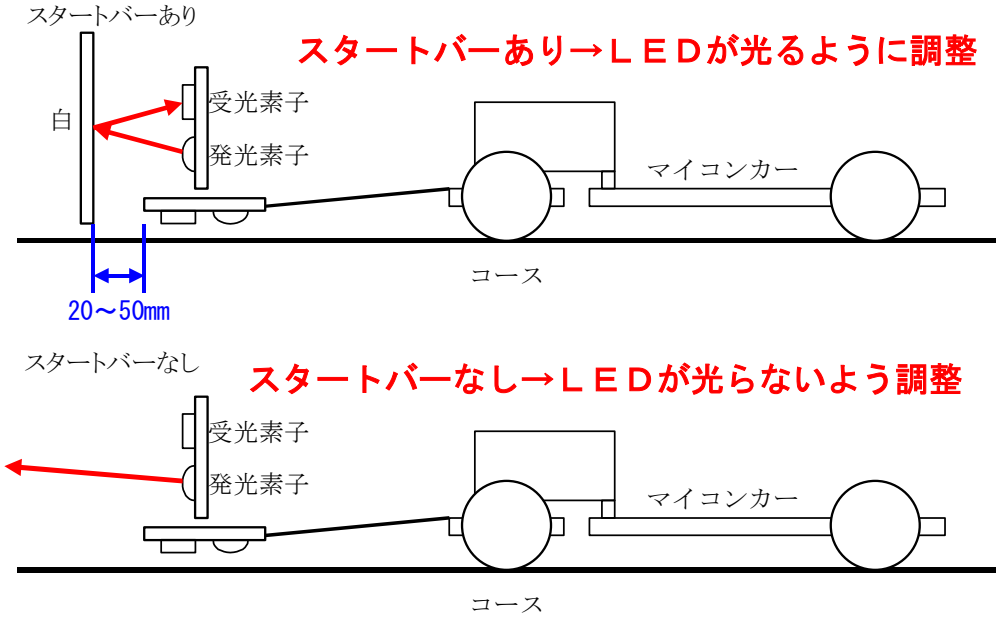
ディップスイッチを”0011”の状態で CPU ボード用電源、モータ用電源を ON します。
スタートバー検出センサ基板の反応がなければ(LED 消灯)LED0 が点灯、反応があれば(LED 点灯)LED1 が点灯します。

スタートバー検出センサ基板の LED 反応が無い場合は、スタートバー検出センサ基板の半田付け不良、半田ブリッジ(ショート)、LED の向きが逆など考えられます。スタートバー検出センサ基板の LED 反応があってもモータドライブ基板の LED 反応がない場合、スタートバー検出センサ基板と CPU ボード間の不具合が考えられます。テスタ等で原因を突き止めてください。

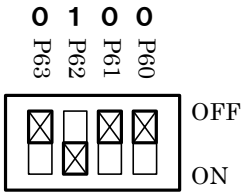
※スタートバー検出センサ基板の感度調整



感度調整用ボリュームで、感度を調整します。マイコンカーの先頭から 20～50mm の位置に白板(スタートバー)を置き、白板ありで LED 点灯、なしで LED 消灯するよう調整します。



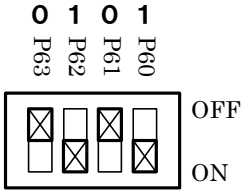
4.6. 右モータのテスト



スイッチの状態

ディップスイッチを”0100”の状態で CPU ボード用電源、モータ用電源を ON します。
右モータが 1 秒ごとに「正転→ブレーキ」を繰り返します。

右モータが正転しない場合は、右モータ制御回路の半田付け不良が考えられます。回転し続ける場合は、半田ブリッジしている可能性があります。目視チェック、テスタ等で原因を突き止めてください。
　また、タイヤが逆転した場合は、モータのケーブルが逆です。コネクタの 1 ピンと 2 ピンの端子を入れ替えて下さい。

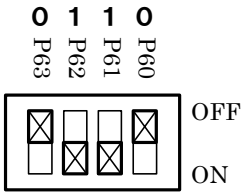


スイッチの状態

一度電源を OFF して、ディップスイッチを”0101”の状態で CPU ボード用電源、モータ用電源を ON します。
右モータが 1 秒ごとに「逆転→ブレーキ」を繰り返します。

右モータが逆転しない場合は、半田付け不良やショートが考えられます。目視チェック、テスタ等で原因を突き止めてください。

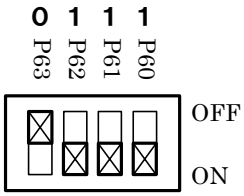
4.7. 左モータのテスト



スイッチの状態

ディップスイッチを”0110”の状態で CPU ボード用電源、モータ用電源を ON します。
左モータが 1 秒ごとに「正転→ブレーキ」を繰り返します。

左モータが正転しない場合は、左モータ制御回路の半田付け不良が考えられます。回転し続ける場合は、半田ブリッジしている可能性があります。目視チェック、テスタ等で原因を突き止めてください。
また、タイヤが逆転した場合は、モータのケーブルが逆です。コネクタの 1 ピンと 2 ピンの端子を入れ替えて下さい。

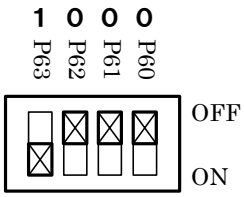


スイッチの状態

一度電源を OFF して、ディップスイッチを”0111”の状態で CPU ボード用電源、モータ用電源を ON します。
左モータが 1 秒ごとに「逆転→ブレーキ」を繰り返します。

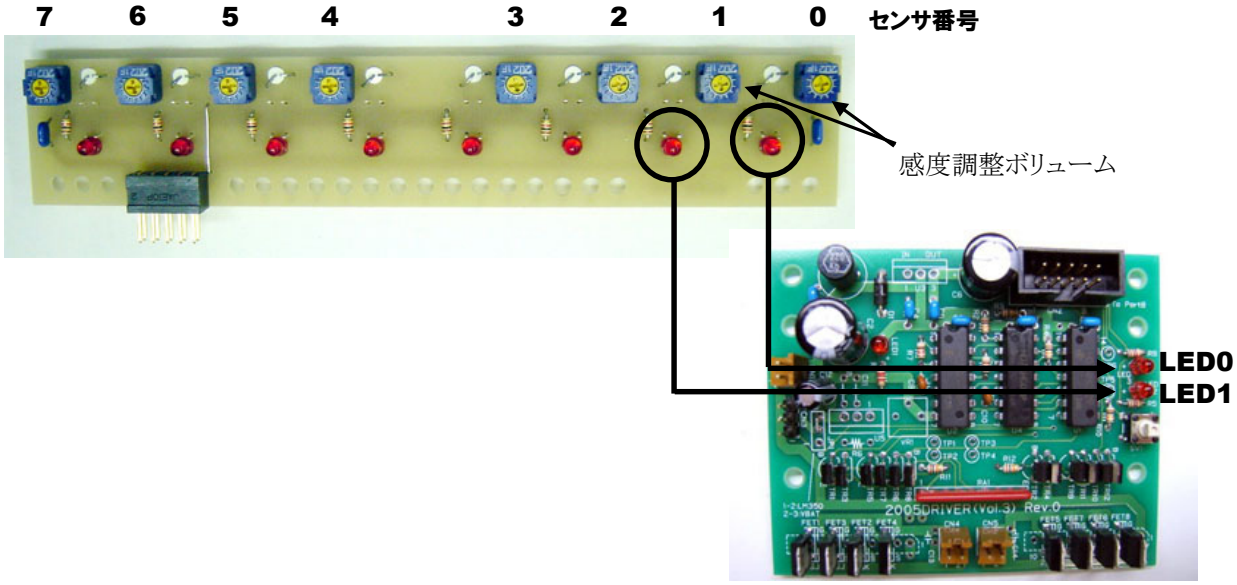
左モータが逆転しない場合は、半田付け不良やショートが考えられます。目視チェック、テスタ等で原因を突き止めてください。

4.8. センサのテスト



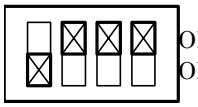
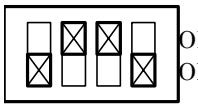
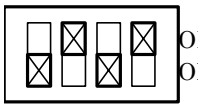
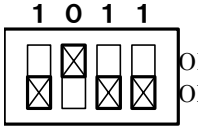
スイッチの状態

ディップスイッチを”1000”の状態で CPU ボード用電源を ON します。
下図のようにセンサ基板のセンサ0とセンサ1の状態が、モータドライブ基板の 2 つの LED にモニタされます。センサ基板上の LED も点灯しますので、同じ反応かテストします。センサの感度はボリュームで調整します。

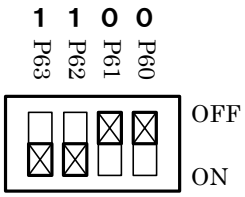


もしセンサ基板上の LED が点灯しない場合は、センサ基板の半田付け不良、半田ブリッジ、部品の逆差し等が考えられます。センサ基板上の LED は点灯するのにモータドライブ基板上の LED が点灯しない場合は、サブ基板やコネクタ周りで不具合が発生している可能性があります。目視チェック、テスタ等で原因を突き止めてください。

次にスイッチを”1001”に換えるとセンサ2，センサ3、”1010”ならセンサ4，センサ5、”1011”ならセンサ6，センサ7がモータドライブ基板の LED にモニタされます。

スイッチ	モータドライブ基板のLED1に出力されるセンサ番号	モータドライブ基板のLED0に出力されるセンサ番号
1 0 0 0  OFF ON	1	0
1 0 0 1  OFF ON	3	2
1 0 1 0  OFF ON	5	4
1 0 1 1  OFF ON	7	6

4.9. 直進テスト



スイッチの状態

ディップスイッチを”1100”の状態で CPU ボード用電源、モータ用電源を ON します。
2 秒後に PWM 値 50%で 2 秒間直進します。廊下など平らで直線の長い場所でマイコンカーを走らせ、まっすぐに進むかテストします。曲がってしまう場合は、「4.4. サーボのテスト」で説明したとおり、SERVO_CENTER の値を調整して直進するように調整します。直進性はマイコンカーのスピードが上がると非常に重要になりますので必ず行います。最終的には1ずつ値を微調整するぐらいの心がけで調整するとOK です。

直進テストは、PWM 値とストップするまでの時間の違いで 4 パターンあります。テスト走行する場所の広さに合わせて調整して下さい。

スイッチ	PWM 値	停止するまでの時間
1 1 0 0 OFF ON	50%	2 秒
1 1 0 1 OFF ON	50%	5 秒
1 1 1 0 OFF ON	100%	2 秒
1 1 1 1 OFF ON	100%	5 秒

4.10. テスト終了

全機能が正常に動作したなら、走行プログラムを書き込み、コースを走らせてみましょう。その前に、調整項目があります。

・プログラム解説マニュアル「13. サーボセンタと最大切れ角の調整」でサーボセンタと最大切れ角を調整してください。
・プロジェクト kit06 の「kit06.mot」ファイルを書き込み、コースを走らせてみてください。

5. プログラムソース

5.1.「kit06test.c」のプログラム内容

```
 1 :  /*****************************************************************************/
 2 :  /* マイコンカーテストプログラム「kit06test.c」                                 */
 3 :  /*                          2006.04 ジャパンマイコンカーラリー実行委員会     */
 4 :  /*****************************************************************************/
 5 :
 6 :  /*
 7 :  キット用センサ基板、モータドライブ基板、スタートバー検出センサの
 8 :  テストを行います。
 9 :  CPU基板のディップスイッチによりテスト内容を変更します。
10 :     DipSW
11 :  bit3 2 1 0
12 :     0 0 0 0 LEDのテスト       LEDが0.5秒間隔で交互に点灯
13 :     0 0 0 1 プッシュスイッチのテスト OFF時：LED0点灯 ON時：LED1点灯
14 :     0 0 1 0 サーボのテスト    0°→右30°→左30° の繰り返し
15 :     0 0 1 1 スタートバー検出センサのテスト OFF時：LED0点灯 ON時：LED1点灯
16 :     0 1 0 0 右モータのテスト 正転→ブレーキの繰り返し
17 :     0 1 0 1                  逆転→ブレーキの繰り返し
18 :     0 1 1 0 左モータのテスト 正転→ブレーキの繰り返し
19 :     0 1 1 1                  逆転→ブレーキの繰り返し
20 :
21 :     1 0 0 0 センサテスト      センサbit1,0をLED1,0に出力
22 :     1 0 0 1                  センサbit3,2をLED1,0に出力
23 :     1 0 1 0                  センサbit5,4をLED1,0に出力
24 :     1 0 1 1                  センサbit7,6をLED1,0に出力
25 :
26 :     1 1 0 0 直進テスト        PWM  50%で前進、 2秒後ストップ
27 :     1 1 0 1 直進テスト        PWM  50%で前進、 5秒後ストップ
28 :     1 1 1 0 直進テスト        PWM 100%で前進、 2秒後ストップ
29 :     1 1 1 1 直進テスト        PWM 100%で前進、 5秒後ストップ
30 :  */
31 :
32 :  /*=====================================*/
33 :  /* インクルード                        */
34 :  /*=====================================*/
35 :  #include    <machine.h>
36 :  #include    "h8_3048.h"
37 :
38 :  /*=====================================*/
39 :  /* シンボル定義                        */
40 :  /*=====================================*/
41 :
42 :  /* 定数設定 */
43 :  #define         TIMER_CYCLE     3071    /* タイマのサイクル 1ms     */
44 :                                          /* φ/8で使用する場合、      */
45 :                                          /* φ/8 = 325.5[ns]          */
46 :                                          /* ∴TIMER_CYCLE =           */
47 :                                          /*      1[ms] / 325.5[ns]   */
48 :                                          /*                = 3072    */
49 :  #define         PWM_CYCLE       49151   /* PWMのサイクル 16ms       */
50 :                                          /* ∴PWM_CYCLE =             */
51 :                                          /*      16[ms] / 325.5[ns]  */
52 :                                          /*                 = 49152   */
53 :  #define         SERVO_CENTER    5000    /* サーボのセンタ値         */
54 :  #define         HANDLE_STEP     26      /* 1°分の値                 */
55 :
56 :  /*=====================================*/
57 :  /* プロトタイプ宣言                    */
58 :  /*=====================================*/
59 :  void init( void );
60 :  unsigned char sensor_inp( unsigned char mask );
61 :  unsigned char dipsw_get( void );
62 :  unsigned char pushsw_get( void );
63 :  unsigned char startbar_get( void );
64 :  void led_out( unsigned char led );
65 :  void speed( int accele_l, int accele_r );
66 :  void handle( int angle );
67 :  char unsigned bit_change( char unsigned in );
68 :
69 :  /*=====================================*/
70 :  /* グローバル変数の宣言                */
71 :  /*=====================================*/
72 :  unsigned long   cnt0;                   /* timer関数用              */
73 :  unsigned long   cnt1;                   /* main内で使用             */
74 :
75 :  /*****************************************************************/
76 :  /* メインプログラム                                                */
77 :  /*****************************************************************/
78 :  void main( void )
79 :  {
80 :      unsigned char   now_sw;             /* 現在ディップスイッチ記憶 */
```

```
 81 :      unsigned char   before_sw;          /* 前回ディップスイッチ記憶 */
 82 :      unsigned char   c;                  /* 作業用                   */
 83 :      int             i;                  /* 作業用                   */
 84 :
 85 :      /* マイコン機能の初期化 */
 86 :      init();                             /* 初期化                   */
 87 :      set_ccr( 0x00 );                    /* 全体割り込み許可         */
 88 :
 89 :      /* 変数初期化 */
 90 :      before_sw = dipsw_get();
 91 :      cnt1 = 0;
 92 :
 93 :      /* マイコンカーの状態初期化 */
 94 :      handle( 0 );
 95 :      speed( 0, 0 );
 96 :      led_out( 0x0 );
 97 :
 98 :      while( 1 ) {
 99 :      /* ディップスイッチ読み込み */
100 :      now_sw = dipsw_get();
101 :
102 :      /* 前回のスイッチ値と比較 */
103 :      if( before_sw != now_sw ) {
104 :          /* 不一致なら前回値更新、タイマ値のクリア */
105 :          before_sw = now_sw;
106 :          cnt1 = 0;
107 :      }
108 :
109 :      /* ディップスイッチの値によりテストモードの選択 */
110 :      switch( now_sw ) {
111 :
112 :          /* LEDのテスト LEDが0.5秒間隔で交互に点灯 */
113 :          case 0:
114 :              if( cnt1 < 500 ) {
115 :                  led_out( 0x1 );
116 :              } else if( cnt1 < 1000 ) {
117 :                  led_out( 0x2 );
118 :              } else {
119 :                  cnt1 = 0;
120 :              }
121 :              break;
122 :
123 :          /* プッシュスイッチのテスト OFF時：LED0点灯 ON時：LED1点灯 */
124 :          case 1:
125 :              led_out( pushsw_get() + 1 );
126 :              break;
127 :
128 :          /* サーボのテスト 0°→右30°→左30° の繰り返し */
129 :          case 2:
130 :              if( cnt1 < 1000 ) {
131 :                  handle( 0 );
132 :              } else if( cnt1 < 2000 ) {
133 :                  handle( 30 );
134 :              } else if( cnt1 < 3000 ) {
135 :                  handle( -30 );
136 :              } else {
137 :                  cnt1 = 0;
138 :              }
139 :              break;
140 :
141 :          /* スタートバー検出センサのテストOFF時：LED0点灯 ON時：LED1点灯 */
142 :          case 3:
143 :              led_out( startbar_get() + 1 );
144 :              break;
145 :
146 :          /* 右モータのテスト 正転→ブレーキの繰り返し */
147 :          case 4:
148 :              if( cnt1 < 1000 ) {
149 :                  speed( 0, 100 );
150 :              } else if( cnt1 < 2000 ) {
151 :                  speed( 0, 0 );
152 :              } else {
153 :                  cnt1 = 0;
154 :              }
155 :              break;
156 :
157 :          /* 右モータのテスト 逆転→ブレーキの繰り返し */
158 :          case 5:
159 :              if( cnt1 < 1000 ) {
160 :                  speed( 0, -100 );
161 :              } else if( cnt1 < 2000 ) {
162 :                  speed( 0, 0 );
163 :              } else {
164 :                  cnt1 = 0;
165 :              }
166 :              break;
167 :
168 :          /* 左モータのテスト 正転→ブレーキの繰り返し */
169 :          case 6:
170 :              if( cnt1 < 1000 ) {
171 :                  speed( 100, 0 );
```

```
172 :             } else if( cnt1 < 2000 ) {
173 :                 speed( 0, 0 );
174 :             } else {
175 :                 cnt1 = 0;
176 :             }
177 :             break;
178 :
179 :         /* 左モータのテスト 逆転→ブレーキの繰り返し */
180 :         case 7:
181 :             if( cnt1 < 1000 ) {
182 :                 speed( -100, 0 );
183 :             } else if( cnt1 < 2000 ) {
184 :                 speed( 0, 0 );
185 :             } else {
186 :                 cnt1 = 0;
187 :             }
188 :             break;
189 :
190 :         /* センサテスト センサbit1,0をLED1,0に出力 */
191 :         case 8:
192 :             c = sensor_inp( 0x03 );
193 :             led_out( c );
194 :             break;
195 :
196 :         /* センサテスト センサbit3,2をLED1,0に出力 */
197 :         case 9:
198 :             c = sensor_inp( 0x0c );
199 :             c = c >> 2;
200 :             led_out( c );
201 :             break;
202 :
203 :         /* センサテスト センサbit5,4をLED1,0に出力 */
204 :         case 10:
205 :             c = sensor_inp( 0x30 );
206 :             c = c >> 4;
207 :             led_out( c );
208 :             break;
209 :
210 :         /* センサテスト センサbit7,6をLED1,0に出力 */
211 :         case 11:
212 :             c = sensor_inp( 0xc0 );
213 :             c = c >> 6;
214 :             led_out( c );
215 :             break;
216 :
217 :         /* 直進テスト PWM  50%で前進、 2秒後ストップ */
218 :         case 12:
219 :             if( cnt1 < 2000 ) {
220 :                 speed( 0, 0 );
221 :             } else if( cnt1 < 4000 ) {
222 :                 speed( 50, 50 );
223 :             } else {
224 :                 speed( 0, 0 );
225 :             }
226 :             break;
227 :
228 :         /* 直進テスト PWM  50%で前進、 5秒後ストップ */
229 :         case 13:
230 :             if( cnt1 < 2000 ) {
231 :                 speed( 0, 0 );
232 :             } else if( cnt1 < 7000 ) {
233 :                 speed( 50, 50 );
234 :             } else {
235 :                 speed( 0, 0 );
236 :             }
237 :             break;
238 :
239 :         /* 直進テスト PWM 100%で前進、 2秒後ストップ */
240 :         case 14:
241 :             if( cnt1 < 2000 ) {
242 :                 speed( 0, 0 );
243 :             } else if( cnt1 < 4000 ) {
244 :                 speed( 100, 100 );
245 :             } else {
246 :                 speed( 0, 0 );
247 :             }
248 :             break;
249 :
250 :         /* 直進テスト PWM 100%で前進、 5秒後ストップ */
251 :         case 15:
252 :             if( cnt1 < 2000 ) {
253 :                 speed( 0, 0 );
254 :             } else if( cnt1 < 7000 ) {
255 :                 speed( 100, 100 );
256 :             } else {
257 :                 speed( 0, 0 );
258 :             }
259 :             break;
260 :
261 :         /* どれでもないなら */
262 :         default:
```

```
263 :             break;
264 :     }
265 :     }
266 : }
267 :
268 : /*********************************************************************/
269 : /* H8/3048F内蔵モジュール　初期化                                    */
270 : /*********************************************************************/
271 : void init( void )
272 : {
273 :     /* ポートの入出力設定 */
274 :     P1DDR = 0xff;
275 :     P2DDR = 0xff;
276 :     P3DDR = 0xff;
277 :     P4DDR = 0xff;
278 :     P5DDR = 0xff;
279 :     P6DDR = 0xf0;                       /* CPU基板上のDIP SW        */
280 :     P8DDR = 0xff;
281 :     P9DDR = 0xf7;                       /* 通信ポート               */
282 :     PADDR = 0xf7;                       /* スタートバー検出センサ   */
283 :     PBDR  = 0xc0;
284 :     PBDDR = 0xfe;                       /* モータドライブ基板Vol.3  */
285 :     /* ※センサ基板のP7は、入力専用なので入出力設定はありません     */
286 :
287 :     /* ITU0 1ms毎の割り込み */
288 :     ITU0_TCR = 0x23;
289 :     ITU0_GRA = TIMER_CYCLE;
290 :     ITU0_IER = 0x01;
291 :
292 :     /* ITU3,4 リセット同期PWMモード 左右モータ、サーボ用 */
293 :     ITU3_TCR = 0x23;
294 :     ITU_FCR  = 0x3e;
295 :     ITU3_GRA = PWM_CYCLE;               /* 周期の設定               */
296 :     ITU3_GRB = ITU3_BRB = 0;            /* 左モータのPWM設定        */
297 :     ITU4_GRA = ITU4_BRA = 0;            /* 右モータのPWM設定        */
298 :     ITU4_GRB = ITU4_BRB = SERVO_CENTER; /* サーボのPWM設定          */
299 :     ITU_TOER = 0x38;
300 :
301 :     /* ITUのカウントスタート */
302 :     ITU_STR = 0x09;
303 : }
304 :
305 : /*********************************************************************/
306 : /* ITU0 割り込み処理                                                 */
307 : /*********************************************************************/
308 : #pragma interrupt( interrupt_timer0 )
309 : void interrupt_timer0( void )
310 : {
311 :     ITU0_TSR &= 0xfe;                   /* フラグクリア             */
312 :     cnt0++;
313 :     cnt1++;
314 : }
315 :
316 : /*********************************************************************/
317 : /* センサ状態検出                                                    */
318 : /* 引数　マスク値                                                    */
319 : /* 戻り値 センサ値                                                   */
320 : /*********************************************************************/
321 : unsigned char sensor_inp( unsigned char mask )
322 : {
323 :     unsigned char sensor;
324 :
325 :     sensor  = P7DR;
326 :
327 :     /* 新センサ基板は、向かって左がbit0、右がbit7と前回センサ基板と     */
328 :     /* 逆なので、互換を保つためビットを入れ替える                       */
329 :     sensor  = bit_change( sensor );     /* ビット入れ替え           */
330 :     sensor &= mask;
331 :
332 :     return sensor;
333 : }
334 :
335 : /*********************************************************************/
336 : /* ディップスイッチ値読み込み                                        */
337 : /* 戻り値 スイッチ値 0～15                                           */
338 : /*********************************************************************/
339 : unsigned char dipsw_get( void )
340 : {
341 :     unsigned char sw;
342 :
343 :     sw  = ~P6DR;                        /* ディップスイッチ読み込み */
344 :     sw &= 0x0f;
345 :
346 :     return  sw;
347 : }
348 :
349 : /*********************************************************************/
350 : /* プッシュスイッチ値読み込み                                        */
351 : /* 戻り値 スイッチ値 ON:1 OFF:0                                      */
352 : /*********************************************************************/
353 : unsigned char pushsw_get( void )
```

```
354 :  {
355 :      unsigned char sw;
356 :
357 :      sw  = ~PBDR;                        /* スイッチのあるポート読み込み */
358 :      sw &= 0x01;
359 :
360 :      return  sw;
361 :  }
362 :
363 :  /*********************************************************************/
364 :  /* スタートバー検出センサ読み込み                                     */
365 :  /* 戻り値 センサ値 ON(バーあり):1 OFF(なし):0                         */
366 :  /*********************************************************************/
367 :  unsigned char startbar_get( void )
368 :  {
369 :      unsigned char b;
370 :
371 :      b  = ~PADR;                         /* スイッチのあるポート読み込み */
372 :      b &= 0x08;
373 :      b >>= 3;
374 :
375 :      return  b;
376 :  }
377 :
378 :  /*********************************************************************/
379 :  /* LED制御                                                            */
380 :  /* 引数  スイッチ値 LED0:bit0 LED1:bit1  "0":消灯 "1":点灯            */
381 :  /* 例)0x3→LED1:ON LED0:ON  0x2→LED1:ON LED0:OFF                      */
382 :  /*********************************************************************/
383 :  void led_out( unsigned char led )
384 :  {
385 :      unsigned char data;
386 :
387 :      led = ~led;
388 :      led <<= 6;
389 :      data = PBDR & 0x3f;
390 :      PBDR = data | led;
391 :  }
392 :
393 :  /*********************************************************************/
394 :  /* 速度制御                                                           */
395 :  /* 引数   左モータ:-100～100 , 右モータ:-100～100                     */
396 :  /*        0で停止、100で正転100%、-100で逆転100%                      */
397 :  /*********************************************************************/
398 :  void speed( int accele_l, int accele_r )
399 :  {
400 :      unsigned char   sw_data;
401 :      unsigned long   speed_max;
402 :
403 :      sw_data  = dipsw_get() + 5;         /* ディップスイッチ読み込み */
404 :      speed_max = (unsigned long)(PWM_CYCLE-1) * sw_data / 20;
405 :
406 :      /* 左モータ */
407 :      if( accele_l >= 0 ) {
408 :          PBDR &= 0xfb;
409 :          ITU3_BRB = speed_max * accele_l / 100;
410 :      } else {
411 :          PBDR |= 0x04;
412 :          accele_l = -accele_l;
413 :          ITU3_BRB = speed_max * accele_l / 100;
414 :      }
415 :
416 :      /* 右モータ */
417 :      if( accele_r >= 0 ) {
418 :          PBDR &= 0xf7;
419 :          ITU4_BRA = speed_max * accele_r / 100;
420 :      } else {
421 :          PBDR |= 0x08;
422 :          accele_r = -accele_r;
423 :          ITU4_BRA = speed_max * accele_r / 100;
424 :      }
425 :  }
426 :
427 :  /*********************************************************************/
428 :  /* サーボハンドル操作                                                 */
429 :  /* 引数   サーボ操作角度：-90～90                                     */
430 :  /*        -90で左へ90度、0でまっすぐ、90で右へ90度回転                 */
431 :  /*********************************************************************/
432 :  void handle( int angle )
433 :  {
434 :      ITU4_BRB = SERVO_CENTER - angle * HANDLE_STEP;
435 :  }
436 :
437 :  /*********************************************************************/
438 :  /* ビット入れ替え                                                     */
439 :  /* 引数   入れ替えする値                                              */
440 :  /* 戻り値 入れ替え後の値                                              */
441 :  /*********************************************************************/
442 :  char unsigned bit_change( char unsigned in )
443 :  {
444 :      unsigned char ret;
```

```
445 :     int i;
446 :
447 :     for( i = 0;  i < 8; i++ ) {
448 :         ret >>= 1;                    /* 戻り値の右シフト              */
449 :         ret |= in & 0x80;             /* ret bit7 = in bit7            */
450 :         in  <<= 1;                    /* 引数の左シフト                */
451 :     }
452 :     return ret;
453 : }
454 :
455 : /*********************************************************************/
456 : /* end of file                                                       */
457 : /*********************************************************************/
```

5.2.「kit06teststart.src」のプログラム内容

```
 1 : ;=====================================
 2 : ; 定義
 3 : ;=====================================
 4 : RESERVE: .EQU    H'FFFFFFFF            ; 未使用領域のアドレス
 5 :
 6 : ;=====================================
 7 : ; 外部参照
 8 : ;=====================================
 9 :         .IMPORT _main
10 :         .IMPORT _interrupt_timer0
11 :
12 : ;=====================================
13 : ; ベクタセクション
14 : ;=====================================
15 :         .SECTION V
16 :         .DATA.L RESET_START           ;  0 H'000000   ﾘｾｯﾄ
17 :         .DATA.L RESERVE               ;  1 H'000004   ｼｽﾃﾑ予約
18 :         .DATA.L RESERVE               ;  2 H'000008   ｼｽﾃﾑ予約
19 :         .DATA.L RESERVE               ;  3 H'00000c   ｼｽﾃﾑ予約
20 :         .DATA.L RESERVE               ;  4 H'000010   ｼｽﾃﾑ予約
21 :         .DATA.L RESERVE               ;  5 H'000014   ｼｽﾃﾑ予約
22 :         .DATA.L RESERVE               ;  6 H'000018   ｼｽﾃﾑ予約
23 :         .DATA.L RESERVE               ;  7 H'00001c   外部割り込み NMI
24 :         .DATA.L RESERVE               ;  8 H'000020   ﾄﾗｯﾌﾟ命令
25 :         .DATA.L RESERVE               ;  9 H'000024   ﾄﾗｯﾌﾟ命令
26 :         .DATA.L RESERVE               ; 10 H'000028   ﾄﾗｯﾌﾟ命令
27 :         .DATA.L RESERVE               ; 11 H'00002c   ﾄﾗｯﾌﾟ命令
28 :         .DATA.L RESERVE               ; 12 H'000030   外部割り込み IRQ0
29 :         .DATA.L RESERVE               ; 13 H'000034   外部割り込み IRQ1
30 :         .DATA.L RESERVE               ; 14 H'000038   外部割り込み IRQ2
31 :         .DATA.L RESERVE               ; 15 H'00003c   外部割り込み IRQ3
32 :         .DATA.L RESERVE               ; 16 H'000040   外部割り込み IRQ4
33 :         .DATA.L RESERVE               ; 17 H'000044   外部割り込み IRQ5
34 :         .DATA.L RESERVE               ; 18 H'000048   ｼｽﾃﾑ予約
35 :         .DATA.L RESERVE               ; 19 H'00004c   ｼｽﾃﾑ予約
36 :         .DATA.L RESERVE               ; 20 H'000050   WDT MOVI
37 :         .DATA.L RESERVE               ; 21 H'000054   REF CMI
38 :         .DATA.L RESERVE               ; 22 H'000058   ｼｽﾃﾑ予約
39 :         .DATA.L RESERVE               ; 23 H'00005c   ｼｽﾃﾑ予約
40 :         .DATA.L _interrupt_timer0     ; 24 h'000060   ITU0 IMIA0
41 :         .DATA.L RESERVE               ; 25 H'000064   ITU0 IMIB0
42 :         .DATA.L RESERVE               ; 26 H'000068   ITU0 OVI0
43 :         .DATA.L RESERVE               ; 27 H'00006c   ｼｽﾃﾑ予約
44 :         .DATA.L RESERVE               ; 28 H'000070   ITU1 IMIA1
45 :         .DATA.L RESERVE               ; 29 H'000074   ITU1 IMIB1
46 :         .DATA.L RESERVE               ; 30 H'000078   ITU1 OVI1
47 :         .DATA.L RESERVE               ; 31 H'00007c   ｼｽﾃﾑ予約
48 :         .DATA.L RESERVE               ; 32 H'000080   ITU2 IMIA2
49 :         .DATA.L RESERVE               ; 33 H'000084   ITU2 IMIB2
50 :         .DATA.L RESERVE               ; 34 H'000088   ITU2 OVI2
51 :         .DATA.L RESERVE               ; 35 H'00008c   ｼｽﾃﾑ予約
52 :         .DATA.L RESERVE               ; 36 H'000090   ITU3 IMIA3
53 :         .DATA.L RESERVE               ; 37 H'000094   ITU3 IMIB3
54 :         .DATA.L RESERVE               ; 38 H'000098   ITU3 OVI3
55 :         .DATA.L RESERVE               ; 39 H'00009c   ｼｽﾃﾑ予約
56 :         .DATA.L RESERVE               ; 40 H'0000a0   ITU4 IMIA4
57 :         .DATA.L RESERVE               ; 41 H'0000a4   ITU4 IMIB4
58 :         .DATA.L RESERVE               ; 42 H'0000a8   ITU4 OVI4
59 :         .DATA.L RESERVE               ; 43 H'0000ac   ｼｽﾃﾑ予約
60 :         .DATA.L RESERVE               ; 44 H'0000b0   DMAC DEND0A
61 :         .DATA.L RESERVE               ; 45 H'0000b4   DMAC DEND0B
62 :         .DATA.L RESERVE               ; 46 H'0000b8   DMAC DEND1A
63 :         .DATA.L RESERVE               ; 47 H'0000bc   DMCA DEND1B
64 :         .DATA.L RESERVE               ; 48 H'0000c0   ｼｽﾃﾑ予約
65 :         .DATA.L RESERVE               ; 49 H'0000c4   ｼｽﾃﾑ予約
66 :         .DATA.L RESERVE               ; 50 H'0000c8   ｼｽﾃﾑ予約
67 :         .DATA.L RESERVE               ; 51 H'0000cc   ｼｽﾃﾑ予約
68 :         .DATA.L RESERVE               ; 52 H'0000d0   SCI0 ERI0
69 :         .DATA.L RESERVE               ; 53 H'0000d4   SCI0 RXI0
70 :         .DATA.L RESERVE               ; 54 H'0000d8   SCI0 TXI0
```

```
71 :          .DATA.L RESERVE                 ; 55 H'0000dc   SCI0 TEI0
72 :          .DATA.L RESERVE                 ; 56 H'0000e0   SCI1 ERI1
73 :          .DATA.L RESERVE                 ; 57 H'0000e4   SCI1 RXI1
74 :          .DATA.L RESERVE                 ; 58 H'0000e8   SCI1 TXI1
75 :          .DATA.L RESERVE                 ; 59 H'0000ec   SCI1 TEI1
76 :          .DATA.L RESERVE                 ; 60 H'0000f0   A/D ADI
77 :
78 :  ;=====================================
79 :  ; スタートアッププログラム
80 :  ;=====================================
81 :          .SECTION P
82 :  RESET_START:
83 :          MOV.L   #H'FFF10,ER7            ; スタックの設定
84 :          JSR     @_main                  ; Ｃ言語のmain()関数へジャンプ
85 :  OWARI:
86 :          BRA     OWARI
87 :
88 :          .END
```