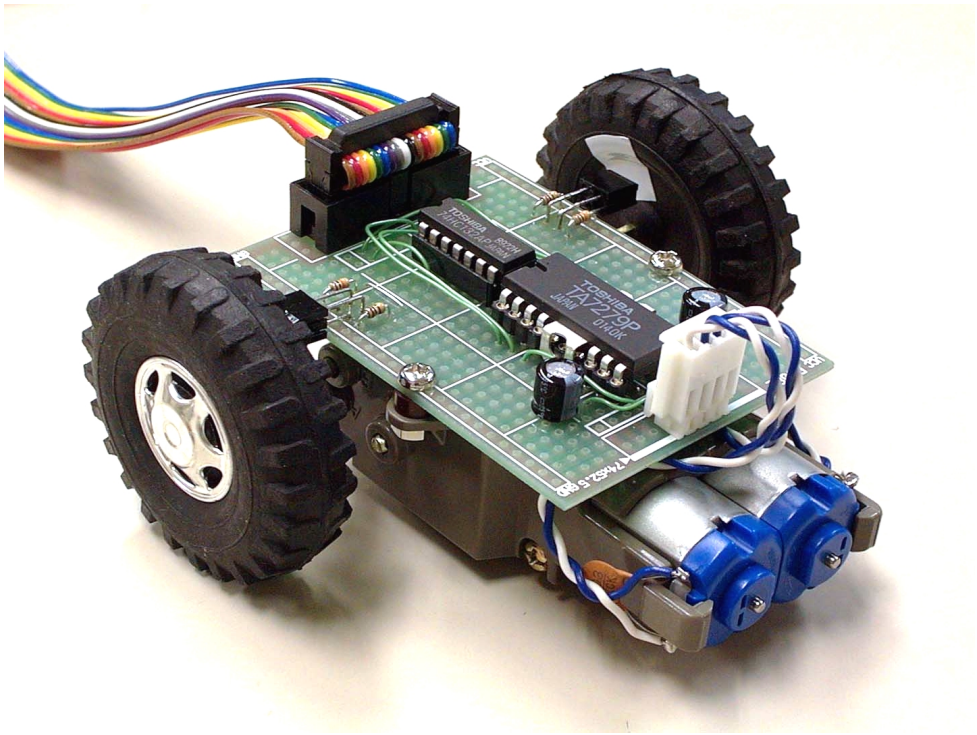
－TK-3687 Option－

DC モータの回転制御

Version 1.04



目　次

Toyo-linx,Co.,Ltd.

1 はじめに、、、

このキットはツインモータギアボックスとドライバ IC・TA7279P/AP を用いた DC モータ制御の学習キットです。単純な PIO での制御と PWM での制御を学習します。また、タイヤ付近に取り付けられたフォトリフレクタでタイヤの回転数を検出することが出来ます。

PWM（Pulse Wide Modulation）とはパルス幅変調というパルス信号の ON/OFF の比率を変化させる変調方式の一つです。例えば ON と OFF が各 50%と等しい場合、平均値は振幅の 50%になります。また、ON が 70%で OFF が 30%の場合、平均値は振幅の 70%の値になります。このように、On/OFF の比率を変えることにより平均値が変わりますから、回転数をモニタして ON/OFF の比率（パルス幅）を変えれば一定回転数が得られる、というのが PWM 変調による回転制御の原理です。

モータドライバ IC には TA7279P/AP(東芝)を使用します。ドライバを 2 チャンネル搭載し、2bit で正転・逆転・ブレーキ・停止のコントロールが出来ます。H8/3687 のタイマ Z には PWM を最大 6 相出力する機能が備わっています。今回使用するモータドライバ IC はモーター つに対して 2bit 信号を入力して制御しますので左右合計で 4 つの出力を使用します。

学習内容は、まず PIO でモータドライバ IC を制御しモータを回転させてみます。次に PWM での簡単なモータ制御を行い、最後に任意の回転数を一定に保つ制御を行います。作業の流れは、、、

ハードの組み立て
↓
モータ制御の基本（PIO でのモータ制御）
↓
PWM でのモータ制御
↓
リモコン走行
↓
回転数一定制御

の順で説明していきます。

作業を進めていくに当たり次のものを用意して下さい。

- **組み立て**

  ハンダごて、ハンダ、ニッパ、ワイヤストリッパ、ピンセット、＋ドライバ

- **プログラム**

  HEW(High-performance Embedded Workshop)、もしくは Motorola 形式 HEX ファイル(*.mot)を生成するアセンブラ(本文内で掲載しているリストは HEW を使用して作成しています)

- **動作確認**

  ハイパーターミナル（Windows 付属）、D-Sub9pin ストレートケーブル

2 ハードの組み立て

各プログラムを作成する前に付属のユニバーサル基板にハードを組み上げます。プリント基板と違いユニバーサル基板は全ての配線を自分で結線しなければなりません。P.3 の回路図を見ながら部品をハンダ付けしていく事によって、ハードの構成を理解してみましょう。

まず、工具・部品の確認を行いましょう。下記の部品表と照らし合わせて全ての部品が揃っている事を確認して下さい。

■工具

ハンダごて、ハンダ、ニッパ、ワイヤストリッパ、ピンセット、＋ドライバ

■部品

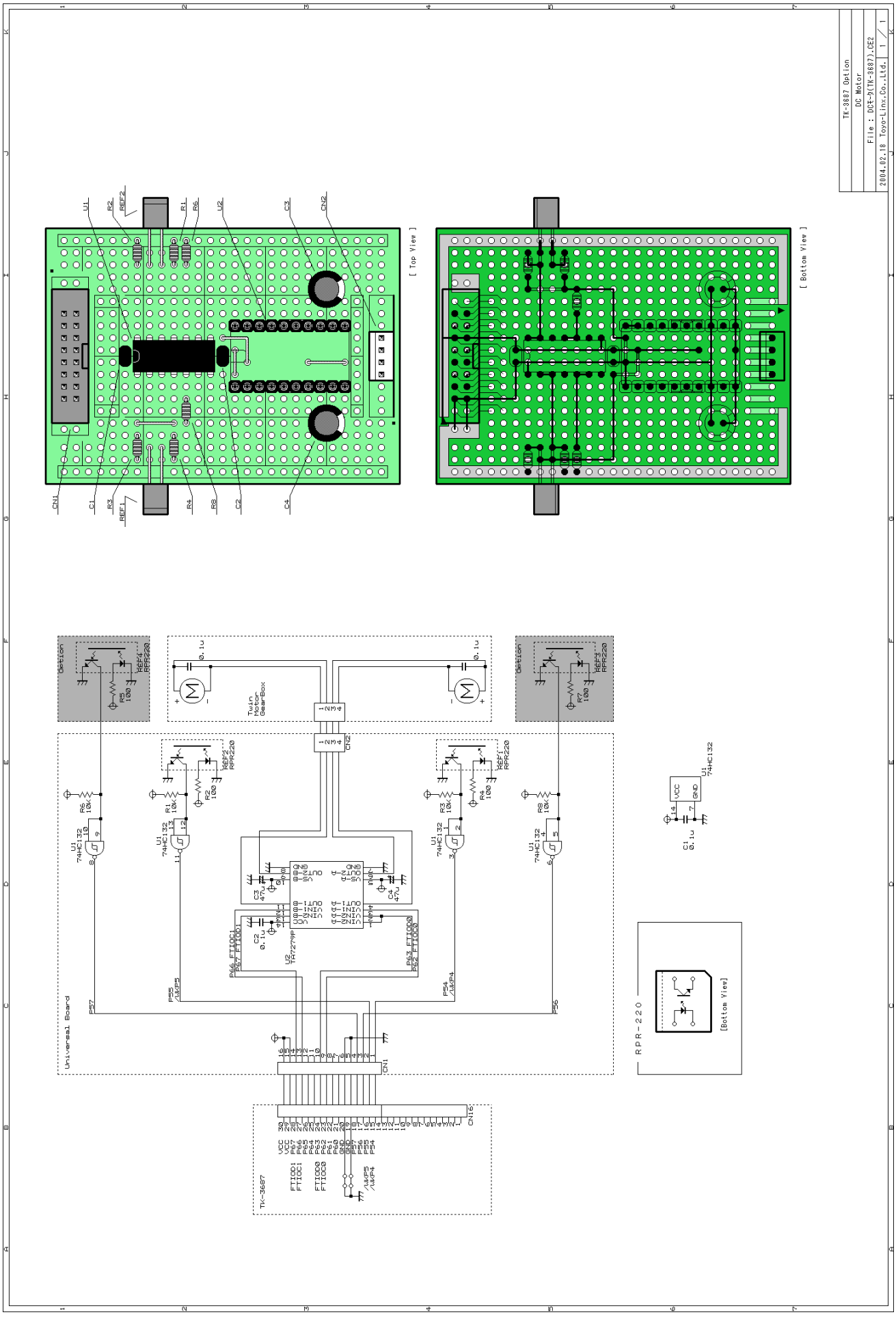
TK-3687オプション DCモータ 部品表	全数:	1

	部品番号	型名，規格	メーカー	数量	全数	備考
1	■ユニバーサル基板					
2	U1	74HC132		1	1	
3	U2	TA7279P/AP	東芝	1	1	
	U1用ICソケット	14pin		1	1	
4	U2用ソケット	丸ピン10連×2		1	1	
5	REF1～4	RPR-220	ROHM	2	4	REF3,4は付属していません
6	R2,4,5,7	100Ω		2	4	R5,7は付属していません
7	R1,3,6,8	10kΩ		4	4	
8	C1,2	0.1μF （積セラ）		2	2	
9	C3,4	47μF/16V （電解）		2	2	
10	CN1,16	16pinBOX		2	2	CN16はCPU基板用
11	CN2	4pinストレート		1	1	
12	ラッピングケーブル	50cm		1	1	
13	メッキ線					Vcc,GND等半田面結線用 ※1
14	基板	B6082	東洋リンクス	1	1	穴あけ加工済み
15	■モータ囲り					
16	ギアボックス	ツインモータギアボックス	田宮模型	1	1	
17	ノイズ吸収用コンデンサ	0.1～0.47μF		2	2	
18	モータ接続用ケーブル	4pin×10cm		1	1	
19	タイヤ			2	2	
20	スペーサ			2	2	ギアボックス内のスペーサを使用
21	■その他					
22	接続ケーブル	16pin×70cm		1	1	
23						

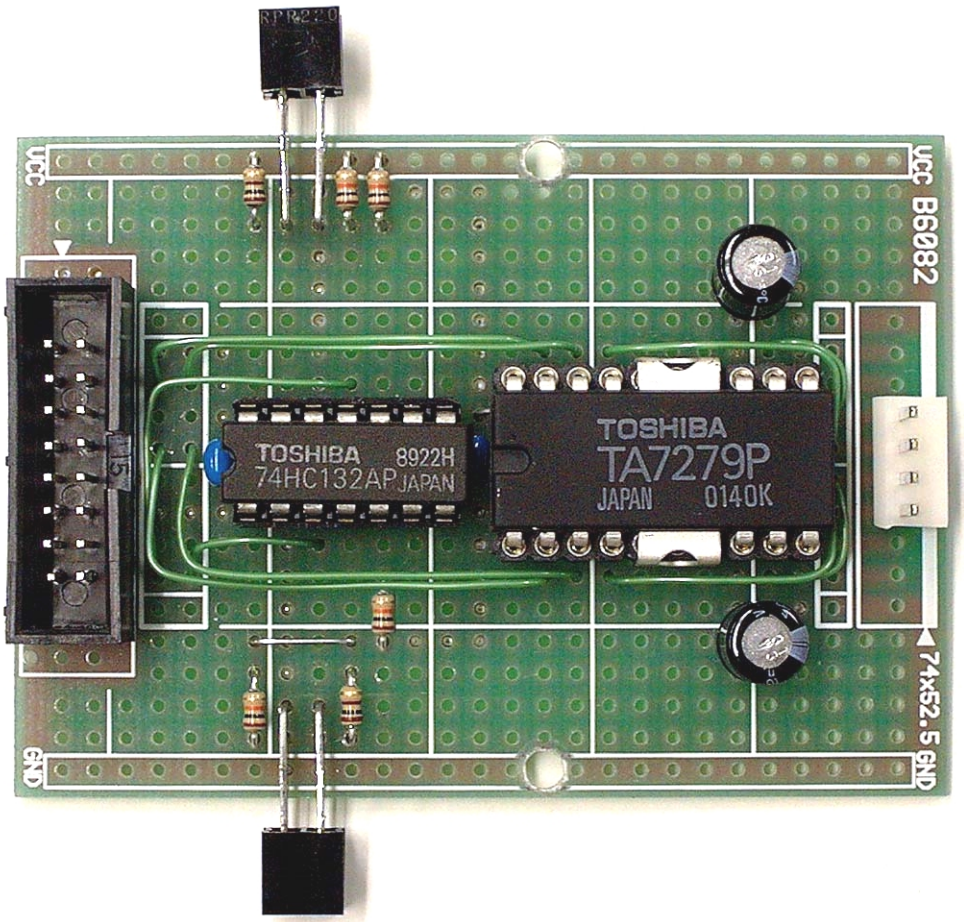
※相当品を使用する場合があります。
※1 ラッピングケーブルの被覆を剥し2本をよじって使用します。また抵抗やコンデンサの足も流用できます。

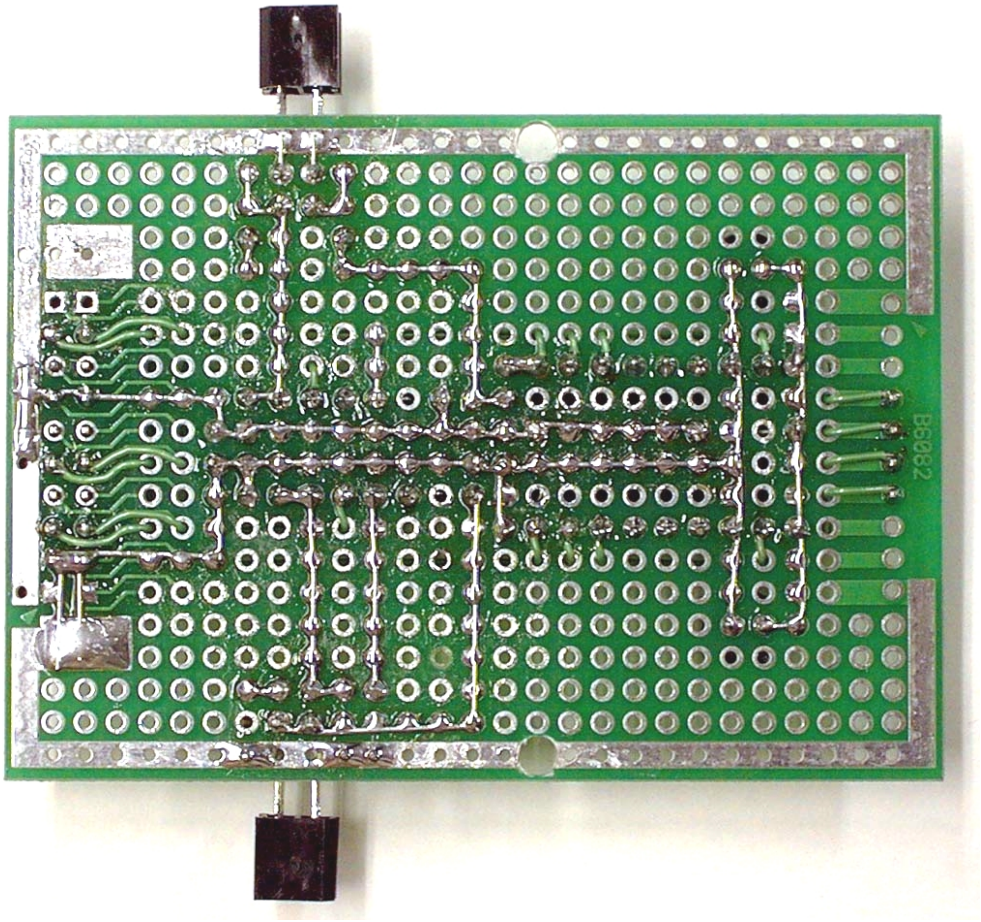
部品の確認が済みましたら、いよいよ組み立てです。3～6 ページの回路図・実装図、完成写真等をよく見ながらハンダ付けを行って下さい。ハンダ付けによるケーブル配線は大変なので、電源や GND、交差しない信号線等はハンダ面でメッキ線や部品の余ったリード線を流用して接続します（P.4 完成写真下段を参照）。ハンダ面のみでは配線しきれない信号線はラッピングケーブルで配線していきます。尚、ラッピングケーブルでの配線が初めての方は 5 ページの結線方法を参照して下さい。

■回路図・実装図


TK-3687 Option
DC Motor
File : DCE-9(TK-3687).CE2
2004.02.18 Toyo-Linx.Co.,Ltd.
[Top View]
[Bottom View]
Option
Twin Gear Box
Universal Board
TK-3687
[Bottom View]
RPR-220

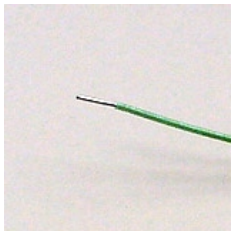
■完成写真


VCC
B6082
TOSHIBA
8922H
74HC132AP JAPAN
TOSHIBA
TA7279P
JAPAN 0140K
71x52.5
GND

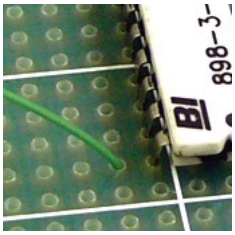

B6082

■ラッピングケーブルでの結線方法

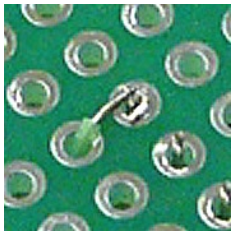
ハンダ面で結線されない配線はラッピングケーブルで結線していきます。ここではラッピングケーブルでの結線の仕方を説明します。



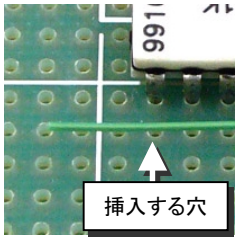
1.
被覆を剥きます
(5mm 位)



2.
部品面からに穴に通します



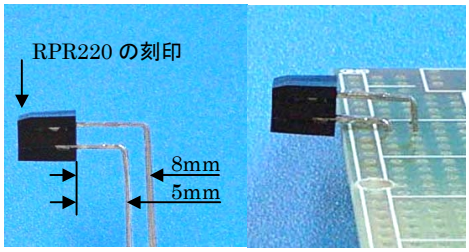
3.
ハンダ付けする部品の足にラッピングケーブルを絡げてからハンダ付けします



4.
結線先までケーブルを這わせ、挿入する穴から 3 つ先の穴辺りでカットします
1～3 の手順でハンダ付けします

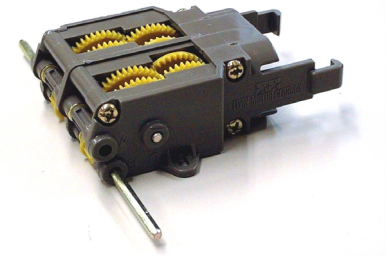
■フォトリフレクタの加工

右図のように、フォトリフレクタを横から見てセンサ面に斜めの切込み(RPR220 の刻印がある)が入っている側を上にし、上側の足を約 8mm、下側の足を約 5mm のところで直角に折り曲げます。加工したフォトリフレクタを部品面から奥まで挿し込み半田付けして下さい。



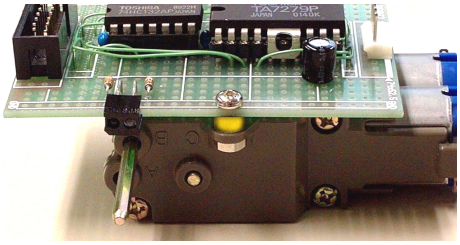
■ギアボックスの組み立て

ツインモータギアボックスの組み立ては、ギアボックス付属の説明書を参照して下さい。組み立てるタイプは**"B タイプ"**です。少々複雑なので間違えの無いよう組み立ててください。また、ハトメをしっかり奥まで差し込まないと回転の抵抗になりギアが上手く回らない場合があります。モータを取り付ける前に車軸を回し、滑らかに回る事を確認して下さい。



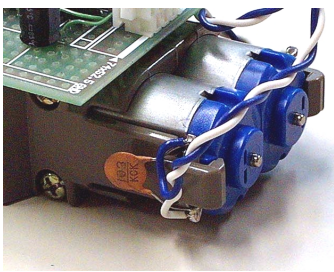
■基板とギアボックスの固定

ユニバーサル基板、ギアボックス、共に組み立て終えたら、右写真のようにギアボックスを裏返し、基板とギアボックスの間に黄色のスペーサを入れビス、ナットで固定してください(ギアボックス付属)。



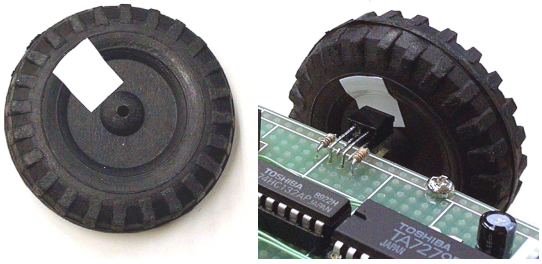
■モータへのノイズ吸収コンデンサとケーブルの接続

モータ用ノイズ吸収コンデンサとケーブルはモータをギアボックスにセットしてから接続します。まず、ケーブルの被覆を剥いてハンダメッキをして下さい。次にコンデンサの足をモータの端子に合わせて広げ、端子の穴にコンデンサのリードを通します。併せてケーブルも穴に通してハンダ付けします。この時コネクタ奇数番号のケーブルが基板側になるようにして下さい。逆に接続すると回転が反転してしまいます。



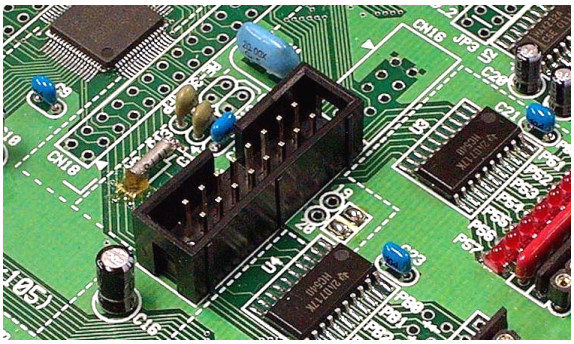
■タイヤにインデックスを付ける

フォトリフレクタで回転を検出できるように、タイヤの内側に白いビニールテープ等で写真のようにラインを入れます。フォトリフレクタはタイヤの黒色とこのラインの白を検出し信号を作ります。



■TK-3687 の作業

ユニバーサル基板とケーブルで接続する為に TK-3687 にボックスコネクタをハンダ付けします。CN16 の 30 番ピン側にボックスコネクタの 16 番ピンがくるように半田付けして下さい。次に GND を取るために 19、20 のスルーの隣端子二箇所をハンダブリッジでショートしてください。



■動作チェック

付属 CD-R 内の DC モータフォルダ（CD-ROM:¥TK-3687¥ｵﾌﾟｼｮﾝ¥DC ﾓｰﾀ¥ｱｾﾝﾌﾞﾗ¥ﾌﾟﾛｸﾞﾗﾑ）の中にあるチェックプログラム“DCM_CHK.mot”をモニタプログラムでロードし、スキップトレースで確認してみましょう。左右それぞれ正転（前進）→ブレーキ→逆転（後退）→停止を行いますのでタイヤの回転方向が合っているか確認します。モータのある方が前です。全く動作しない場合はもう一度ユニバーサル基板の結線等、各ハンダ付けのチェックを行います。回転方向が逆の場合はモータの結線やドライバ IC の接続をチェックして下さい。

```
;---------------------------------------------------------
;
;   FILE        :DCM_CHK.src
;   DATE        :Thu, Sep 18, 2003
;   DESCRIPTION :Main Program
;   CPU TYPE    :H8/3687
;
;   This file is generated by Hitachi Project Generator (Ver.2.1
;
;---------------------------------------------------------
;ＰＩＯでのモータ回転
;単純なＰＩＯ出力で正転・反転・ブレーキ・停止の制御を行う
;モニタのトレース実行で動作確認

     .include   "io3687F_equ.inc"

     .export    _main


;=========================================================
;          メインプログラム
;=========================================================
_main:
     mov.b      #H'FF,r0l
     mov.b      r0l,@PCR6       ;P6 all out

?000:
     xor.b      r0l,r0l
     mov.b      r0l,@PDR6       ;Motor stop (Hi-Z)
     bset       #3,r0l
     mov.b      r0l,@PDR6       ;CW/CCW in1=0 / in2=1
     bset       #2,r0l
     mov.b      r0l,@PDR6       ;Brake  in1=1 / in2=1
     bclr       #3,r0l
     mov.b      r0l,@PDR6       ;CCW/CW in1=1 / in2=0
     bclr       #2,r0l
     mov.b      r0l,@PDR6       ;Stop   in1=0 / in2=0
?100:
     xor.b      r0l,r0l
     mov.b      r0l,@PDR6       ;Motor stop (Hi-Z)
     bset       #7,r0l
     mov.b      r0l,@PDR6       ;CW/CCW in1=0 / in2=1
     bset       #6,r0l
     mov.b      r0l,@PDR6       ;Brake  in1=1 / in2=1
     bclr       #7,r0l
     mov.b      r0l,@PDR6       ;CCW/CW in1=1 / in2=0
     bclr       #6,r0l
     mov.b      r0l,@PDR6       ;Stop   in1=0 / in2=0

     bra        ?000            ;loop

;=========================================================
          .end
```

【リスト 2-1　DCM_CHK.src】

3 モータ制御の基本

この章ではまず PIO でモータドライバ IC を制御してモータを回転させ、モータドライバ IC の使い方をマスタします。

■モータドライバ IC“TA7279P/AP”

TA7279P/AP は正転・逆転・ブレーキ・停止の 4 モードがコントロールできる DC モータドライバ IC です。以下に TK-3687 との接続と動作ファンクションを示します。

TA7279P/AP	IN1	IN2	OUT1	OUT2	MODE
	1	1	L	L	BRAKE
	0	1	L	H	CW/CCW
	1	0	H	L	CCW/CW
	0	0	ハイインピーダンス		STOP
TK-3687	P62	P63			
	P66	P67			

＜表 3-1 TA7279P/AP 動作ファンクションと TK-3687 との接続＞

MODE の CW/CCW は正転・逆転を示しています。このキットでは IN1=0,IN2=1 で正転（前進）、IN1=1,IN2=0 で逆転（後退）するハードの使用になっています。PIO での制御では表に従って、タイヤを前進、つまり正転させたいときは P62 もしくは P66 に 0 を、P63 もしくは P67 に 1 を出力すればモータは前進方向に回転します。

まずは単純に回転させて見ましょう。ポート 6 を出力に設定して IN1 に“0”を、IN2 に“1”を出力して正転させるプログラムです（DCM_01）。

```
;------------------------------------------------------------------------
;                                                                       |
;   FILE        :DCM_01.src                                             |
;   DATE        :Thu, Sep 18, 2003                                      |
;   DESCRIPTION :Main Program                                           |
;   CPU TYPE    :H8/3687                                                |
;                                                                       |
;   This file is generated by Hitachi Project Generator (Ver.2.1).      |
;                                                                       |
;------------------------------------------------------------------------
;ＰＩＯでのモータ回転

     .include  "io3687F_equ.inc"

     .export   _main

;=====================================================================
;    メインプログラム
;=====================================================================
_main:
     mov.b     #H'FF,r0l
     mov.b     r0l,@PCR6  ;P6 all out

     mov.b     #H'88,r0l  ;IN1:P62,P66=1 / IN2:P63,P67=0
     mov.b     r0l,@PDR6  ;CCW/CW in1=1 / in2=0

     bra       $          ;loop

;=====================================================================
     .end
```

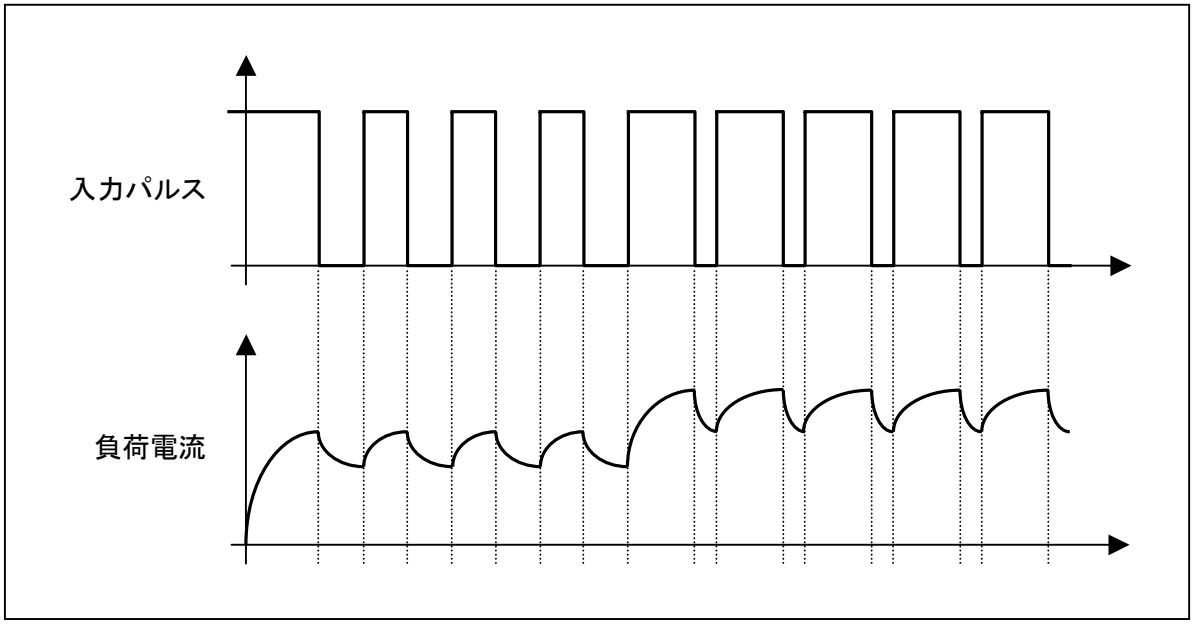
【リスト 3-1 DCM_01.src】

“DCM_01”を実行するとタイヤは勢いよく回転しだします。また R0L に入力する値を H’44 にすれば逆回転になります。

4 PWM でのモータ制御

3 章でモータの回転方向は出力するポートの値でコントロール出来るの事が解ったと思います。しかし、この制御では回転方向をコントロールできても回転速度はコントロールできません。そこで、冒頭で出てきた PWM での制御になります。

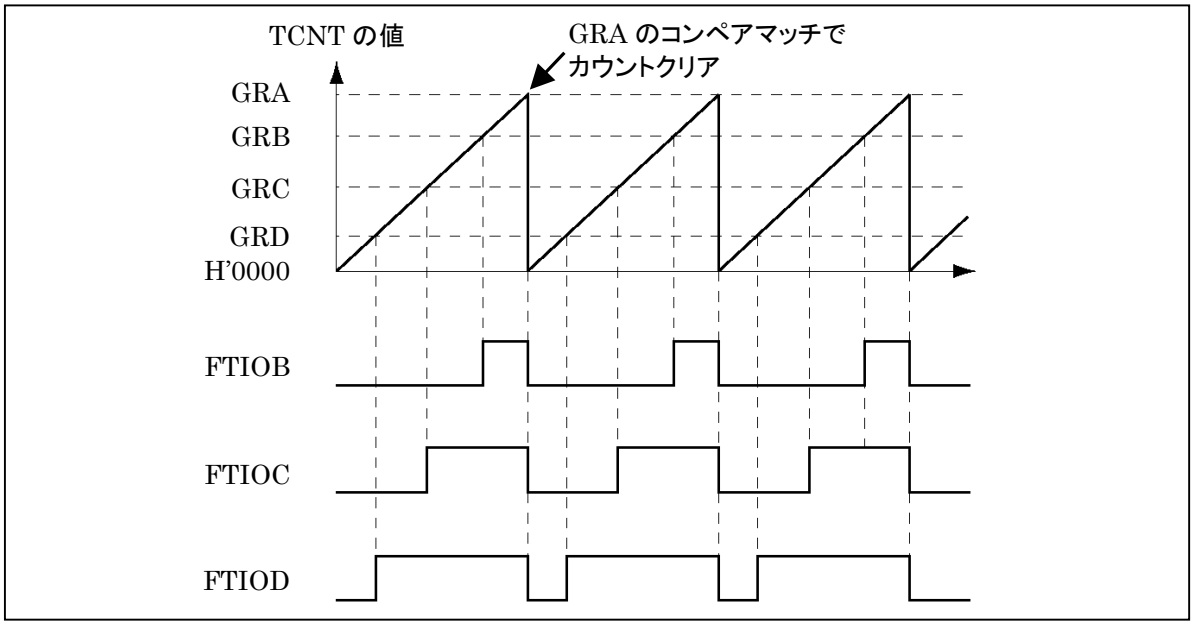
PWM は入力信号レベルに応じて ON/OFF の比率（パルス幅）を変化させる制御と説明しました。モータ（コイル）には電流を流そうとしたときにはなかなか流れない、また電流を切ろうとしたときは電流がなかなか減らないという特性があります。PWM 制御はこの特性を利用してモータにパルス信号を入力し三角波のような負荷電流でモータを駆動します。また、パルスの比率を変えることで負荷電流を多くすることも少なくすることも出来ます。その関係を図に表したのが図 4-1 です。



【図 4-1 PWM 制御の原理】

図 4-1 では途中から入力パルスの ON の比率が大きくなっています。すると負荷電流も合わせて大きくなっているのが読み取れます。このようにパルスの ON/OFF の比率を変化させると負荷電流をコントロールすることが出来る、つまりモータの回転速度をコントロールすることが出来るのです。

それでは次に PWM 信号をどのように作るかです。H8/3687 のタイマ Z には PWM モードが備わっています。この PWM モードはフリーランしているタイマ Z のカウント値とジェネラルレジスタの値を比較（コンペア）し、ジェネラルレジスタの値以上か、以下かで出力を決定します。ジェネラルレジスタには A～D（GRA～GRD）の 4 つがあり、GRA でタイマ Z のカウント周期を、残りのジェネラルレジスタで出力する PWM 信号のデューティ比を設定します。GRA の半分の値を GRB にセットすればデューティ比 50%のパルスが FTIOB に、また GRA の値の 70%の値を GRC にセットすればデューティ比 70%のパルスが FTIOC に得られます。タイマ Z は 2 チャンネル有り、それぞれにジェネラルレジスタを持っていますので、最大 6 相の PWM 信号を取り出すことが出来ます。動作例を図 4-2 に示します。



【図 4-2 タイマ Z の PWM 動作】

4 ページの回路図から分かるようにモータドライブ IC の IN1 には FTIOC 出力が、IN2 には FTIOD 出力が接続されていますので、今までのポート出力をタイマ Z の出力に切り替え、タイマ Z を PWM 動作させればモータを PWM で制御することが出来ます。

■タイマ Z のイニシャライズ

タイマ Z を PWM モードで動作させる手順と各レジスタの意味を説明していきます。

タイマ Z を PWM モードで動作させるには次の手順で行います。( )内は使用するレジスタです。

1. カウンタクロックの選択(タイマコントロールレジスタ・TCR)
   ↓
2. カウンタクリア要因の選択(タイマコントロールレジスタ・TCR)
   ↓
3. PWM モードの設定(TPMR)
   ↓
4. 初期出力レベルの設定(TOCR)
   ↓
5. 出力レベルの選択(POCR)
   ↓
6. GR の設定(GRA～GRD)
   ↓
7. 波形出力の許可(TOER)
   ↓
8. カウンタ動作開始(TSTR)

使用するレジスタの内容、及び、PWM モード時の設定値も併せて示します。レジスタ説明の表は上から、ビット番号、ビット名、内容、PWM 時の設定値、設定内容となっています。PWM モードを主に説明していますので、より詳しい内容は“H8/3687 ハードウェアマニュアル”を参照して下さい。

・タイマコントロールレジスタ【TCR_0,TCR_1】

カウンタクロック選択、カウンタクリア要因の選択を行います。各チャンネル毎に 1 本、計 2 本の TCR があります。

7	6	5	4	3	2	1	0
CCLR2	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0
カウンタクリア			クロックエッジ		タイマプリスケーラ		
0	0	1	0	0	0	0	0
GRA のコンペアマッチで TCNT クリア			外部クロック選択時に使用		内部クロック、1/1 でカウント		

・タイマ PWM モードレジスタ【TPMR】

PWM モードの設定を行います。

7	6	5	4	3	2	1	0
-	PWMD1	PWMC1	PWMB1	-	PWMD0	PWMC0	PWMB0
-	PWM モードチャンネル 1			-	PWM モードチャンネル 0		
0	1	1	0	0	1	1	0
各チャンネルの FTIOC,FTIOD を PWM モードに設定							

・タイマアウトプットコントロールレジスタ【TOCR】

コンペアマッチが起こるまでの初期出力を設定をします。

7	6	5	4	3	2	1	0
TOD1	TOC1	TOB1	TOA1	TOD0	TOC0	TOB0	TOA0
出力レベルセレクトチャンネル 1				出力レベルセレクトチャンネル 0			
0	0	0	0	0	0	0	0
アクティブ Hi なので初期出力は Low に設定							

・PWM モードアウトプットレベルコントロールレジスタ【POCR】

PWM モードのアクティブレベルを設定します。POCR は各チャンネルに 1 本、計 2 本あります。

7	6	5	4	3	2	1	0
-	-	-	-	-	POLD	POLC	POLB
-	-	-	-	-	アウトプットレベルコントロール		
0	0	0	0	0	1	1	0
使用する FTIOC,FTIOD をアクティブ Hi に設定							

・ジェネラルレジスタ A、B、C、D【GRA、GRB、GRC、GRD】

GR は 16 ビットのレジスタで各チャンネルに 4 本、計 8 本あります。

GRA はタイマのカウンタクリアの値を設定します。設定値は H'FFFE にして下さい。

GRC,GRD には出力する PWM のデューティ比に相当する値をセットします。GRA の値が基準になりますので、デューティ 50%なら H'8000 となります。ちなみに 100%ON を設定する場合は H'0000 を、0%の場合は H'FFFF をセットします。

・タイマアウトプットマスタイネーブルレジスタ【TOER】

チャンネル 0、1 の出力を許可／禁止します。

7	6	5	4	3	2	1	0
ED1	EC1	EB1	EA1	ED0	EC0	EB0	EA0
マスタイネーブルチャンネル 1				マスタイネーブルチャンネル 0			
0	0	1	1	0	0	1	1
使用する FTIOC,FTIOD の出力を許可							

・タイマスタートレジスタ【TSTR】

TCNT の動作／停止を選択します。

7	6	5	4	3	2	1	0
-	-	-	-	-	-	STR1	STR0
-	-	-	-	-	-	カウンタスタート	
0	0	0	0	0	0	1	1
チャンネル 0、1 共に動作							

■PWM でのモータ回転プログラム

では実際にプログラムを作成してみましょう。前述したようにタイマ Z を PWM モードにイニシャライズし、FTIOC、FTIOD に PWM 信号を出力します。パルスのデューティ比は 50%で前進方向（正転）させましょう。IN1 を 0 に、IN2 に 1 をセットすれば正転方向でしたね。PWM 制御の場合、1 をセットするポートに PWM 信号を出力します。0 をセットするポートには PWM 信号は必要なく、今までと同じく 0 をセットします。以上を元に作成したプログラム“DCM_02”のリストを次ページに示します。

```
;------------------------------------------------------------------------
;                                                                       |
;   FILE        :DCM_02.src                                             |
;   DATE        :Thu, Sep 18, 2003                                      |
;   DESCRIPTION :Main Program                                           |
;   CPU TYPE    :H8/3687                                                |
;                                                                       |
;   This file is generated by Hitachi Project Generator (Ver.2.1).      |
;                                                                       |
;------------------------------------------------------------------------
;ＰＷＭでのモータ回転
;タイマＺのＰＷＭを用いてモータを正転させる
;実行はＧコマンド

     .include  "io3687F_equ.inc"

     .export   _main

;=======================================================================
;          メインプログラム
;=======================================================================
_main:
PWM_INIT:
     mov.b     #B'00100000,r0l ;GRA ｺﾝﾍﾟｱﾏｯﾁｸﾘｱ CLK*1/1
     mov.b     r0l,@TCR_0
     mov.b     r0l,@TCR_1
     mov.b     #B'01100110,r0l ;FTIO-C0,D0,C1,D1 PWM Mode
     mov.b     r0l,@TPMR
     mov.b     #B'00000000,r0l ;初期出力 = all Low.
     mov.b     r0l,@TOCR
     mov.b     #B'00000110,r0l ;Hi active.
     mov.b     r0l,@POCR_0
     mov.b     r0l,@POCR_1
     mov.w     #H'FFFE,r0      ;R0=GRA_0
     mov.w     #H'FFFF,r1      ;R1=GRC_0
     mov.w     #H'8000,r2      ;R2=GRD_0
     mov.w     r0,@GRA_0
     mov.w     r1,@GRC_0
     mov.w     r2,@GRD_0
     mov.w     r0,@GRA_1
     mov.w     r1,@GRC_1
     mov.w     r2,@GRD_1
     mov.b     #B'00110011,r0l ;出力ｲﾈｰﾌﾞﾙ
     mov.b     r0l,@TOER

     mov.b     #B'00000011,r0l ;Run
     mov.b     r0l,@TSTR

     bra       $

;=======================================================================
     .end
```

【リスト 4-1　DCM_02.src】

■PWM 動作時でのジェネラルレジスタの変更

常に一定速度で回転させているのであれば、パルスのデューティ比を変える必要はありません。しかし実際は、例えば走行カーなら内輪の回転数を落としてカーブを曲がったり減速や加速をしたり、という具合に常に一定とは限りません。そこで PWM 動作中でのパルスのデューティ比の変更方法を示します。

動作中のパルスのデューティ比の変更は、PWM 動作を停止せずにジェネラルレジスタを書き替えることで変更できます。変更のタイミングによっては 1 パルス想定外のパルスが出る可能性がありますが、今回使用しているギアボックスではそこまで厳密にする必要はありません。

以下に示すプログラム“DCM_03”は、直進→約 180° 回転→直進→停止の動作を行うプログラムです。片

輪を逆回転させ 180° 回転を行います。回転の制御はジェネラルレジスタを書き替えることで行っています。

```
;---------------------------------------------------------
;
;   FILE        :DCM_03.src
;   DATE        :Thu, Sep 18, 2003
;   DESCRIPTION :Main Program
;   CPU TYPE    :H8/3687
;
;   This file is generated by Hitachi Project Generator (Ver.2.1
;
;---------------------------------------------------------
;ＰＷＭでのモータ回転・直進→反転→直進→停止
;タイマ動作中にジェネラルレジスタを書き替えてＰＷＭ出力を変
;化させる
;１．５秒直進後、方輪を０．５秒逆転、その後１．５秒直進して停止
;する

     .include   "io3687F_equ.inc"

     .export    _main

;=========================================================
;     メインプログラム
;=========================================================
_main:
     bsr        PWM_INIT:16     ;TimerZ(PWM)ｲﾆｼｬﾗｲｽﾞ
                                ;直進設定
     mov.w      #H'FFFF,r1      ;R1=GRC_0
     mov.w      #H'A000,r2      ;R2=GRD_0
     mov.w      r1,@GRC_0
     mov.w      r2,@GRD_0
     mov.w      r1,@GRC_1
     mov.w      r2,@GRD_1
                                ;動作開始
     mov.b      #B'00000011,r0l ;Run
     mov.b      r0l,@TSTR
                                ;直進
     bsr        TIMER500m:16
     bsr        TIMER500m:16
     bsr        TIMER500m:16
                                ;方輪反転設定
     mov.w      #H'A000,r1      ;R1=GRC_0
     mov.w      #H'FFFF,r2      ;R2=GRD_0
     mov.w      r1,@GRC_1
     mov.w      r2,@GRD_1
                                ;方輪反転
     bsr        TIMER500m:16
                                ;直進設定
     mov.w      #H'FFFF,r1      ;R1=GRC_0
     mov.w      #H'A000,r2      ;R2=GRD_0
     mov.w      r1,@GRC_1
     mov.w      r2,@GRD_1
                                ;直進
     bsr        TIMER500m:16
     bsr        TIMER500m:16
     bsr        TIMER500m:16
                                ;停止
     xor.b      r0l,r0l
     mov.b      r0l,@TSTR       ;ﾀｲﾏ ｽﾄｯﾌﾟ
     mov.b      #B'00000000,r0l ;初期出力 = all Low.
     mov.b      r0l,@TOCR

     bra  $
```

```
;=========================================================
;          サブルーチンプログラム
;=========================================================
;---------------------------------------------------------
;          ＰＷＭ(タイマＺ)イニシャライズ
;---------------------------------------------------------
PWM_INIT:
     mov.b      #B'00100000,r0l ;GRA ｺﾝﾍﾟｱﾏｯﾁｸﾘｱ CLK*1/1
     mov.b      r0l,@TCR_0
     mov.b      r0l,@TCR_1
     mov.b      #B'01100110,r0l ;FTIO-C0,D0,C1,D1 PWM Mode
     mov.b      r0l,@TPMR
     mov.b      #B'00000000,r0l ;初期出力 = all Low.
     mov.b      r0l,@TOCR
     mov.b      #B'00000110,r0l ;Hi active.
     mov.b      r0l,@POCR_0
     mov.b      r0l,@POCR_1
     mov.w      #H'FFFE,r0      ;R0=GRA_0
     mov.w      r0,@GRA_0
     mov.w      r0,@GRA_1
     mov.b      #B'00110011,r0l ;出力ｲﾈｰﾌﾞﾙ
     mov.b      r0l,@TOER
     rts


;---------------------------------------------------------
;          500msec ウェイト
;---------------------------------------------------------
TIMER500m:                      ;8
     mov.l      #H'1967E8,er6   ;6
?000:
     dec.l      #1,er6          ;2
     bne        ?000            ;4
     rts                        ;8
;=========================================================
          .end
```

【リスト 4-2　DCM_03.src】

5 リモコン走行

今までの応用として、パソコンのキーボードから回転制御を行い走行させて見ましょう。キーボードのカーソルキーを用いて、上キーで前進、下キーで後退、左キーで左旋回、右キーで右旋回を行わせます。キーデータの送信にはハイパーターミナルを用います。送信されたキーデータを受信し押されているキーを判定、各動作を行わせます。ハイパーターミナルのバージョンによってはカーソルキーのコードを出力しない場合がありますのでテンキーにも割り付けておきましょう。8 が前進、2 が後退、4 が左旋回、6 が右旋回です。

■ソフト

前述の説明の通り割り付けられたキーデータを受信したら各回転制御を行います。受信データと動作の対応を表 6-1 に表します。

カーソルキー	コード	テンキー	コード	動作
↑	H'5B41	8	H'38	前進
↓	H'5B42	2	H'32	後退
←	H'5B44	4	H'34	左旋回
→	H'5B43	6	H'36	右旋回
入力無し	－	入力無し	－	停止

【表 5-1 受信データと動作対応表】

カーソルキーは 2 バイトデータです。H'5B の後のデータで判定しますがプログラムを簡略化する為に H'5B は無視して後の H'4x で判定します。テンキーは 1 バイトデータなのでそのまま判定します。

前進は左右同じ回転数で前進方向にタイヤを回転させます。回転速度は Run_CNST で定義しています。後退は前進と同様左右同じ回転数で後退方向にタイヤを回転させます。回転速度は Back_CNST で定義しています。旋回はそれぞれ軸になるタイヤ（左旋回なら左タイヤ、右旋回なら右タイヤ）を停止し外輪のみを回転させて旋回します。旋回の回転速度は左右それぞれ Left_CNST、Right_CNST で定義しています。

何もキーが押されていない、つまりデータを受信していない時は停止です。しかしキーを押しつづけてもパソコンからは間欠で送られてくるので、そのまま“受信データが無い＝停止”としてしまうと動作と停止を繰り返し行ってしまいます。そこで一定期間受信データが無ければ停止処理を行うようにします。タイマ V で 100μsec 毎に割り込みをかけ割り込み処理内で D'2000 カウントし、カウントしたらフラグを立てます。メインループではフラグが立ったら何もキーが押されていないと判断し停止処理を行います。100μsec×D'2000 ですので 200msec データの受信が無ければキー入力無しとみなす訳です。

以上を踏まえて作成したプログラム“DCM_04”のリストを次ページに記します。

```
;------------------------------------------------------
;
;   FILE        :DCM_04.src
;   DATE        :Thu, Sep 18, 2003
;   DESCRIPTION :Main Program
;   CPU TYPE    :H8/3687
;
;   This file is generated by Hitachi Project Generator (Ver.2.1
;
;------------------------------------------------------
;ＰＷＭでのモータ回転・ターミナルからのコントロール
;ハイパーターミナルのカーソルキー（もしくはテンキー）でＤＣモ
;ータをコントロール　キーを押している間各キーに相当する定回
;転を行う　０．２秒間受信を受け付けなかったら停止処理を行う
;■キー役割
;　↑：前進　／　↓：後退　／　→：右旋回　／　←：左旋回
;　※テンキー使用時
;　８：前進　／　２：後退　／　６：右旋回　／　４：左旋回
;■カーソルキーのコード
;　↑：Ｈ'５Ｂ４１　／　↓：Ｈ'５Ｂ４２
;　→：Ｈ'５Ｂ４３　／　←：Ｈ'５Ｂ４４

     .include  "io3687F_equ.inc"

     .export   _main

     .export   TV_1000CNT
     .export   TV_FLG

Run_CNST   .equ H'4E20-H'2000   ;前進
Back_CNST  .equ H'4E20-H'2000   ;後退
Right_CNST .equ H'4E20-H'2000   ;右旋回
Left_CNST  .equ H'4E20-H'2000   ;左旋回
Stop_CNST  .equ H'4E21          ;停止


;=======================================================
;          メインプログラム
;=======================================================
_main:
;-----  イニシャライズ  ----------------------------------
     bsr       PWM_INIT:16     ;ﾀｲﾏ Z(PWM) ｲﾆｼｬﾗｲｽﾞ
     bsr       INIT_SCI:16     ;SCI3 ｲﾆｼｬﾗｲｽﾞ
     bsr       TV_INIT:16      ;ﾀｲﾏ V ｲﾆｼｬﾗｲｽﾞ

     xor.w     r0,r0
     mov.w     r0,@TV_1000CNT  ;ｸﾘｱ
     mov.b     r0l,@TV_FLG     ;割り込みﾌﾗｸﾞ ｸﾘｱ

     andc      #H'7F,ccr       ;割り込み許可

     mov.b     #B'00000011,r0l ;PWM 動作開始
     mov.b     r0l,@TSTR

;-----  メインループ  ------------------------------------
_main_loop:
     bsr       RECV_SCI:16     ;受信ﾁｪｯｸ
     bcs       _RECV_CHK       ;ﾃﾞｰﾀ有り→ﾃﾞｰﾀ判定へ
_mloop_00:
     mov.b     @TV_FLG,r0l     ;ﾀｲﾏ V ﾁｪｯｸ
     bne       _Stop:16        ;1 秒受信が無ければ停止へ
     bra       _main_loop
```

```
_RECV_CHK:                     ;受信ﾃﾞｰﾀ判定
     cmp.b     #"[",r0l        ;ｶｰｿﾙｷｰ 次のﾃﾞｰﾀ待ち
     beq       _main_loop
     cmp.b     #"A",r0l        ;前進?
     beq       _Run
     cmp.b     #"8",r0l
     beq       _Run
     cmp.b     #"B",r0l        ;後退?
     beq       _Back
     cmp.b     #"2",r0l
     beq       _Back
     cmp.b     #"C",r0l        ;右旋回?
     beq       _Right
     cmp.b     #"6",r0l
     beq       _Right
     cmp.b     #"D",r0l        ;左旋回?
     beq       _Left:16
     cmp.b     #"4",r0l
     beq       _Left:16
     bra       _mloop_00       ;その他は無視

_Run:                          ;前進
     mov.w     #Run_CNST,r1
     mov.w     #Run_CNST,r2
     mov.w     #Stop_CNST,r3
     mov.w     r3,r4
     bra       _GR_Set:16

_Back:                         ;後退
     mov.w     #Stop_CNST,r1
     mov.w     r1,r2
     mov.w     #Back_CNST,r3
     mov.w     r3,r4
     bra       _GR_Set

_Right:                        ;右旋回
     mov.w     #Stop_CNST,r1
     mov.w     #Left_CNST,r2
     mov.w     r1,r3
     mov.w     r3,r4
     bra       _GR_Set

_Left:                         ;左旋回
     mov.w     #Right_CNST,r1
     mov.w     #Stop_CNST,r2
     mov.w     r2,r3
     mov.w     r3,r4
     bra       _GR_Set

_Stop:                         ;停止
     mov.w     #Stop_CNST,r1
     mov.w     r1,r2
     mov.w     r2,r3
     mov.w     r3,r4
     bra       _GR_Set

_GR_Set:                       ;ｼﾞｪﾈﾗﾙﾚｼﾞｽﾀへｾｯﾄ
     mov.w     r1,@GRD_0
     mov.w     r2,@GRD_1
     mov.w     r3,@GRC_0
     mov.w     r4,@GRC_1

     xor.w     r0,r0           ;ﾀｲﾏ V ﾌﾗｸﾞ,ｶｳﾝﾀ ｸﾘｱ
     mov.w     r0,@TV_1000CNT
     mov.b     r0l,@TV_FLG

     bra       _main_loop
```

【リスト 5-1　DCM_04.src （1/3)】

```
;=========================================================
;          サブルーチンプログラム
;=========================================================
;---------------------------------------------------------
;          ＳＣＩ３受信
;---------------------------------------------------------
RECV_SCI:
     mov.b     @SSR,r0l         ;ｼﾘｱﾙｽﾃｰﾀｽﾚｼﾞｽﾀ
     btst      #6,@SSR          ;受信ﾁｪｯｸ
     beq       RSCI_ND
     mov.b     @RDR,r0l         ;ﾃﾞｰﾀ受信
     orc       #H'01,CCR        ;Cy=1
     rts

RSCI_ND:                        ;ﾃﾞｰﾀ無し
     andc      #H'FE,CCR        ;Cy=0
     rts

;---------------------------------------------------------
;          PWM(タイマZ)イニシャライズ
;---------------------------------------------------------
PWM_INIT:
     mov.b     #B'00100000,r0l ;GRA ｺﾝﾍﾟｱﾏｯﾁｸﾘｱ CLK:1/1
     mov.b     r0l,@TCR_0
     mov.b     r0l,@TCR_1
     mov.b     #B'01100110,r0l ;FTIO-C0,D0,C1,D1 PWM ﾓｰﾄﾞ
     mov.b     r0l,@TPMR
     mov.b     #B'00000000,r0l ;初期出力 = all Low.
     mov.b     r0l,@TOCR
     mov.b     #B'00000110,r0l ;Hi active.
     mov.b     r0l,@POCR_0
     mov.b     r0l,@POCR_1
     mov.w     #H'4E20,r0      ;R0=GRA_0
     mov.w     #H'4E21,r1      ;R1=GRC_0
     mov.w     #H'4E21,r2      ;R2=GRD_0
     mov.w     r0,@GRA_0       ;ｼﾞｪﾈﾗﾙﾚｼﾞｽﾀ初期値設定
     mov.w     r1,@GRC_0
     mov.w     r2,@GRD_0
     mov.w     r0,@GRA_1
     mov.w     r1,@GRC_1
     mov.w     r2,@GRD_1
     mov.b     #B'00110011,r0l ;出力ｲﾈｰﾌﾞﾙ
     mov.b     r0l,@TOER
     rts

;---------------------------------------------------------
;          タイマＶイニシャライズ
;---------------------------------------------------------
TV_INIT:
     xor.b     r0l,r0l
     mov.b     r0l,@TCRV0      ;CLK off

     mov.b     #D'125,r0l      ;TCORA=D'125=100usec(CLK*16)
     mov.b     r0l,@TCORA

     mov.b     #B'01001010,r0l ;ｺﾝﾍﾟｱｷｬﾌﾟﾁｬ A 割り込み&ｸﾘｱ
     mov.b     r0l,@TCRV0      ;CLK*1/16
     mov.b     #B'00000000,r0l ;CLK*1/16
     mov.b     r0l,@TCRV1      ;

     rts
```

```
;---------------------------------------------------------
;          ＳＣＩ３イニシャライズ
;---------------------------------------------------------
MHz       .equ D'20            ;X=20MHz
BAUD      .equ 38400           ;9600,19200,38400
BITR      .equ (MHz*D'1000000)/(BAUD*D'32)-1
WAIT_1B   .equ (MHz*D'1000000)/D'6/BAUD

INIT_SCI:
     bset      #1,@PMR1
     xor.b     r0l,r0l
     mov.b     r0l,@SCR3
     mov.b     r0l,@SMR
     mov.b     #BITR,r0l
     mov.b     r0l,@BRR
     mov.w     #WAIT_1B,r0
INITSCI_00:
     dec.w     #1,r0
     bne       INITSCI_00
     mov.b     #H'30,r0l
     mov.b     r0l,@SCR3
     rts


;=========================================================
;          ワークエリア
;=========================================================
     .section  D,data

TV_1000CNT     .res.w    1
TV_FLG         .res.b    1

;=========================================================
     .end
```

【リスト 5-1 DCM_04.src (2/3)】

```
;------------------------------------------------------
;
;   FILE        :intprg.src
;   DATE        :Thu, Sep 18, 2003
;   DESCRIPTION :Interrupt Program
;   CPU TYPE    :H8/3687
;
;   This file is generated by Hitachi Project Generator (Ver.2.1
;
;------------------------------------------------------

    .include    "vect.inc"
    .include    "io3687f_equ.inc"

     .import         TV_1000CNT
     .import         TV_FLG

    .section    IntPRG, code

;1 Reserved
_INT_Reserved1
;2 Reserved
_INT_Reserved2
;3 Reserved
_INT_Reserved3
;4 Reserved
_INT_Reserved4
;5 Reserved
_INT_Reserved5
;6 Reserved
_INT_Reserved6
;7 NMI
_INT_NMI
;8 TRAP #0
_INT_TRAP0
;9 TRAP #1
_INT_TRAP1
;10 TRAP #2
_INT_TRAP2
;11 TRAP #3
_INT_TRAP3
;12 Address break
_INT_ABRK
;13 SLEEP
_INT_SLEEP
;14 IRQ0
_INT_IRQ0
;15 IRQ1
_INT_IRQ1
;16 IRQ2
_INT_IRQ2
;17 IRQ3
_INT_IRQ3
;18 WKP
_INT_WKP
;19 RTC
_INT_RTC
;20 Reserved
_INT_Reserved20
;21 Reserved
_INT_Reserved21
;22 Timer V
_INT_TimerV:
     push.w     r0              ;R0 退避

     bclr       #6,@TCSRV       ;TV ｽﾃｰﾀｽ ｸﾘｱ

     mov.b      @TV_FLG,r0l
     bne        ?100
     mov.w      @TV_1000CNT,r0  ;200msec 用ｶｳﾝﾄ(100u*2000)
     inc.w      #1,r0
     mov.w      r0,@TV_1000CNT
     cmp.w      #D'2000,r0      ;2000 ｶｳﾝﾄ終了?
     bne        ?100
     xor.w      r0,r0
     mov.w      r0,@TV_1000CNT
     mov.b      #1,r0l          ;ﾌﾗｸﾞｾｯﾄ
     mov.b      r0l,@TV_FLG

     mov.b      @PDR5,r0l       ;確認 LED 点滅
     xor.b      #H'02,r0l
     mov.b      r0l,@PDR5
?100:

     pop.w      r0              ;R0 復帰
     rte

;23 SCI3
_INT_SCI3
;24 IIC2
_INT_IIC2
;25 ADI
_INT_ADI
;26 Timer Z0
_INT_TimerZ0
;27 Timer Z1
_INT_TimerZ1
;28 Reserved
_INT_Reserved28
;29 Timer B1
_INT_TimerB1
;30 Reserved
_INT_Reserved30
;31 Reserved
_INT_Reserved31
;32 SCI3_2
_INT_SCI3_2
    sleep
    nop
    .end
```

【リスト 5-1 intprg.src (3/3)】

6 回転数一定プログラム

4 章で基本的な PWM の制御を行いました。6 章ではそれをもとに応用プログラムに挑戦しましょう。

組み立てたユニバーサル基板上にはタイヤの回転を検知するフォトリフレクタあります。このセンサを利用して回転数を一定に保つプログラムを作成します。4 章で作成したデューティ 50%で正転するプログラムがありました。あれも見た目は回転数は一定です。しかしマイコンは回転が本当に一定かは判断できません。もしかするとタイヤに何かが引っかかりその抵抗で回転数が落ちているかもしれません。パルスは常に一定間隔で出力しているからといって回転も常に一定とは限らないのです。そこで、センサを用いて回転数を検出し、所定の回転数より実際の回転数が落ちていたら回転数を上げる。逆に実際の回転が目標の回転数よりも速ければ回転数を落とす、という処理を行います。

■ハード

P.3 の回路図を見てください。回転を検出するフォトリフレクタの信号はポート 5 の P54,P55 にそれぞれ接続されています。この P54,P55 は割り込み入力の/WKP 端子と兼用ピンになっています。そこで、回転の検出処理は割り込みで行います。

■ソフト

回転数を一定に保つ方法として、ある一定間隔内の回転数をカウントしそのカウント数を検証し、カウントが少なければ回転数を上げ、カウントが多ければ回転数を下げるという処理を行います。回転数をカウントする時間は 1sec とします。つまり“1 秒間に何回転しているか”を監視するわけです。

1sec の間隔はタイマ V を使用します。タイマ V はそのままでは 1sec のカウントは行えないので、まず約 1msec のカウントを行い割り込みをかけます。割り込み処理内では、割り込み回数をカウントし 1000 カウント、つまり、1msec × 1000=1sec になったらフラグを立てます。

回転数のカウントは、回転検出センサの割り込み（/WKP4,/WKP5）でカウントし、割り込みがかかったらカウントを+1 します。

メインループでは、ます 1sec のフラグがたつのを待ちます。フラグがたったら回転数のカウントを読み出し目標回転数と比較、回転が足らなければ PWM 信号の ON 信号の比率を大きく、逆に回転数が多ければ OFF 信号の比率を大きくして回転数をコントロールします。

各割り込みがちゃんとかかっているか確認する為に、ポート 5 の P50 と P51 の LED を使用し、割り込みがかかるとポートの出力を反転、LED を点滅させます。

以上がこのソフトの使用です。これを元に作成したプログラム“DCM_05”のリストを次ページに示します。

```
;------------------------------------------------------
;
;   FILE        :DCM_05.src
;   DATE        :Thu, Sep 18, 2003
;   DESCRIPTION :Main Program
;   CPU TYPE    :H8/3687
;
;   This file is generated by Hitachi Project Generator (Ver.2.1
;
;------------------------------------------------------
;ＰＷＭでのモータ回転・回転数監視
;１msec 毎に回転数をカウント、目標回転数に近づくようにＰＷＭ
;を制御

     .include  "io3687F_equ.inc"

     .export   _main
     .export   ROT_CNT
     .export   TV_CNT
     .export   TV_FLG

ROT_CNST       .equ D'2        ;1sec の目標回転数

;======================================================
;          メインプログラム
;======================================================
_main:
;-----  イニシャライズ  ----------------------------------
     bsr       PWM_INIT:16     ;ﾀｲﾏ Z(PWM) ｲﾆｼｬﾗｲｽﾞ
     bsr       TV_INIT:16      ;ﾀｲﾏ V ｲﾆｼｬﾗｲｽﾞ
     bsr       WKP_INIT:16     ;P5(WKP) ｲﾆｼｬﾗｲｽﾞ
     xor.w     r0,r0
     mov.w     r0,@ROT_CNT     ;回転数ｸﾘｱ
     mov.b     r0l,@TV_FLG     ;割り込みﾌﾗｸﾞｸﾘｱ
     mov.w     #H'FFFF,r2      ;GRD 初期値 H'FFFF=停止
     mov.w     r2,@GRD_0
     mov.w     r2,@GRD_1
     mov.w     r2,@GRD_CNT
     andc      #H'7F,ccr       ;割り込み許可
     mov.b     #B'00000010,r0l ;PWM 動作開始
     mov.b     r0l,@TSTR

;-----  メインループ  ----------------------------------
?000:
     mov.b     @TV_FLG,r0l
     and.b     r0l,r0l         ;割り込みﾁｪｯｸ(1sec INT.)
     beq       ?000

     xor.w     r0,r0
     mov.b     r0l,@TV_FLG     ;割り込みﾌﾗｸﾞｸﾘｱ
     mov.w     @ROT_CNT,r1     ;回転数読み込み
     mov.w     r0,@ROT_CNT     ;回転数ﾊﾞｯﾌｧｸﾘｱ
     cmp.w     #ROT_CNST,r1    ;目標値回転数との比較
     beq       ?500
     bcc       ?300

?200:                          ;Cy=1:回転数を上げる
     mov.w     @GRD_CNT,r0
     cmp.w     #H'100,r0       ;これ以上、上げられない
     bcs       ?400
     sub.w     #H'100,r0       ;回転数を上げる
     bra       ?400

?300:                          ;Cy=0:回転数を下げる
     mov.w     @GRD_CNT,r0
     cmp.w     #H'FFFF-H'100,r0     ;これ以上、下げられない
     bcc       ?400
     add.w     #H'100,r0
?400:
     mov.w     r0,@GRD_CNT
     mov.w     r0,@GRD_0
     mov.w     r0,@GRD_1
?500:
     bra       ?000

;------------------------------------------------------
;          ＰＷＭ(タイマＺ)イニシャライズ
;------------------------------------------------------
PWM_INIT:
     mov.b     #B'00100000,r0l ;GRA ｺﾝﾍﾟｱﾏｯﾁｸﾘｱ CLK*1/1
     mov.b     r0l,@TCR_0
     mov.b     r0l,@TCR_1
     mov.b     #B'01100110,r0l ;FTIO-C0,D0,C1,D1 PWM Mode
     mov.b     r0l,@TPMR
     mov.b     #B'00000000,r0l ;初期出力 = all Low.
     mov.b     r0l,@TOCR
     mov.b     #B'00000110,r0l ;Hi active.
     mov.b     r0l,@POCR_0
     mov.b     r0l,@POCR_1
     mov.w     #H'FFFE,r0      ;R0=GRA_0
     mov.w     #H'FFFF,r1      ;R1=GRC_0
     mov.w     r0,@GRA_0
     mov.w     r1,@GRC_0
     mov.w     r0,@GRA_1
     mov.w     r1,@GRC_1
     mov.b     #B'00110011,r0l ;出力ｲﾈｰﾌﾞﾙ
     mov.b     r0l,@TOER
     rts

;------------------------------------------------------
;          タイマＶイニシャライズ
;------------------------------------------------------
TV_INIT:
     mov.b     #D'156,r0l      ;TCORA=D'156=1msec(CLK*128)
     mov.b     r0l,@TCORA
     mov.b     #B'01001011,r0l ;ｲﾝﾌﾟｯﾄｷｬﾌﾟﾁｬ A 割り込み
     mov.b     r0l,@TCRV0      ;CLK*1/128
     mov.b     #B'00000001,r0l ;CLK*1/128
     mov.b     r0l,@TCRV1
     rts

;------------------------------------------------------
;          割り込みイニシャライズ
;------------------------------------------------------
WKP_INIT:
     mov.b     #B'00100000,r0l
     mov.b     r0l,@PMR5       ;WKP5 enable
     mov.b     #B'00001111,r0l ;確認 LED 点灯の為
     mov.b     r0l,@PCR5       ;P50-P53:Out
     mov.b     #B'00100000,r0l ;/WKP ｲﾈｰﾌﾞﾙ
     mov.b     r0l,@IENR1
     xor.b     r0l,r0l
     mov.b     r0l,@IEGR2
     rts

;======================================================
;          ワークエリア
;======================================================
          .section  D,data

ROT_CNT    .res.w          1
GRD_CNT    .res.w          1
TV_CNT     .res.w          1
TV_FLG     .res.b          1

;======================================================
          .end
```

【リスト 6-1 DCM_05.src (1/2)】

```
;------------------------------------------------------------
;
;   FILE        :intprg.src
;   DATE        :Thu, Sep 18, 2003
;   DESCRIPTION :Interrupt Program
;   CPU TYPE    :H8/3687
;
;   This file is generated by Hitachi Project Generator (Ver.2.1
;
;------------------------------------------------------------

    .include    "vect.inc"
    .include    "io3687f_equ.inc"

     .import    TV_CNT
     .import    TV_FLG
     .import    ROT_CNT

    .section    IntPRG, code

;1 Reserved
_INT_Reserved1
;2 Reserved
_INT_Reserved2
;3 Reserved
_INT_Reserved3
;4 Reserved
_INT_Reserved4
;5 Reserved
_INT_Reserved5
;6 Reserved
_INT_Reserved6
;7 NMI
_INT_NMI
;8 TRAP #0
_INT_TRAP0
;9 TRAP #1
_INT_TRAP1
;10 TRAP #2
_INT_TRAP2
;11 TRAP #3
_INT_TRAP3
;12 Address break
_INT_ABRK
;13 SLEEP
_INT_SLEEP
;14 IRQ0
_INT_IRQ0
;15 IRQ1
_INT_IRQ1
;16 IRQ2
_INT_IRQ2
;17 IRQ3
_INT_IRQ3
;18 WKP
_INT_WKP:
     push.w     r0              ;R0 退避
     xor.w      r0,r0           ;ﾌﾗｸﾞ ｸﾘｱ
     mov.b      r0l,@IWPR
     mov.b      #B'00110011,r0l ;PWM ｱｳﾄﾌﾟｯﾄ ｲﾈｰﾌﾞﾙ
     mov.b      r0l,@TOER
     mov.w      @ROT_CNT,r0     ;回転数+1
     inc.w      #1,r0
     mov.w      r0,@ROT_CNT
     mov.b      @PDR5,r0l       ;確認 LED 点滅
     xor.b      #H'01,r0l
     mov.b      r0l,@PDR5
     pop.w      r0              ;R0 復帰
     rte
```

```
;19 RTC
_INT_RTC
;20 Reserved
_INT_Reserved20
;21 Reserved
_INT_Reserved21
;22 Timer V
_INT_TimerV:
     push.w     r0              ;R0 退避
     bclr       #6,@TCSRV       ;ｺﾝﾍﾟｱﾏｯﾁﾌﾗｸﾞ ｸﾘｱ
     mov.w      @TV_CNT,r0      ;TV_CNT+1
     inc.w      #1,r0
     mov.w      r0,@TV_CNT
     cmp.w      #D'1000,r0      ;1sec=1msec*D'1000
     bne        ?000
     xor.w      r0,r0           ;TV_CNT ｸﾘｱ
     mov.w      r0,@TV_CNT
     mov.b      #1,r0l          ;ﾌﾗｸﾞｾｯﾄ
     mov.b      r0l,@TV_FLG
     mov.b      @PDR5,r0l       ;確認 LED 点滅
     xor.b      #H'02,r0l
     mov.b      r0l,@PDR5
?000:
     pop.w r0                   ;R0 復帰
     rte

;23 SCI3
_INT_SCI3
;24 IIC2
_INT_IIC2
;25 ADI
_INT_ADI
;26 Timer Z0
_INT_TimerZ0
;27 Timer Z1
_INT_TimerZ1
;28 Reserved
_INT_Reserved28
;29 Timer B1
_INT_TimerB1
;30 Reserved
_INT_Reserved30
;31 Reserved
_INT_Reserved31
;32 SCI3_2
_INT_SCI3_2
    sleep
    nop
    .end
```

【リスト 6-1 intprg.src (2/2)】

ところで、このプログラムには 1 つ問題があります。一体なんでしょうか？

回転数をフォトリフレクタで検出し PWM で速度をコントロールする、一見何も問題無さそうです。では、実際回転しているタイヤをよく観察してみましょう。1 秒間で何回転しているか、タイヤの内輪に記したインデックスをもとに数えてみてください。100%で回転させてもせいぜい 5、6 回転程度、とても 10 回転以上しているようには見えません。

つまりプログラムの大きな問題点、それは回転数の検出精度がとても低いということです。1 秒間の回転数を数えているので、例えば 2 回転／秒の設定で実際は 2.5 回転／秒していても検出出来ません。3 回転／秒以上ずれてきてから初めて減速のコントロールが行われます。もし回転が毎秒数百、数千回転するような高速なものなら問題は無かったのですが、1 秒間に数回転しかしないようなものではとてもコントロールしているとは言えません。

それでは低速回転のコントロールを考えてみましょう。

まず回転速度をどのようにカウントするかを考えます。前回のプログラムは“1 秒間で何回転しているか”をカウントしていましたが、それではカウントが少なすぎました。そこで、“1 回転で何カウントするか”を数えます。これならばカウントパルスの周波数を高くすれば精度を上げられます。この場合、カウント値は回転数の逆数、即ち周期となり、カウント数が大きければ低速回転、少なければ高速回転していることになります。

次に回転数をコントロールするタイミングを考えてみましょう。使用しているギアボックスはあまり反応が良くないので早くコントロールしても意味がありません。かといってゆっくりしすぎても応答が悪くなります。今回は 100msec 毎にコントロールすることにしましょう。100msec 毎に 1 回転あたりのカウント数を目標カウント値と比較し、目標値よりも少ないカウントなら早く回りすぎているので減速、逆に目標値よりも多いカウントなら回転が遅いので加速します。

PWM の制御ピッチはデューティ比 0～100 を 0.1 ピッチでコントロールすることにします。PWM のパルス間隔を 1msec とし、ジェネラルレジスタ GRA の値を 1msec／CLK=D’20000（H’4E20）に設定します。デューティ比を変えるときには D’20 を加減算します。以上からソフトを次のように変更します。

タイマ V 割り込み

割り込み間隔はコントロールタイミングの正確な 100msec を作る為に 100μsec 毎にします。また、1 回転毎のカウントもタイマ V で行います。1 回転毎のカウントは TV_CNT でカウントし、オーバーフローしてカウントがループしないようにしておきます。、コントロールタイミングのカウントは TV_1000CNT でカウントし、1000 カウント＝100msec 経過したら TV_1000CNT をクリア、フラグ TV_FLG を立てます。

回転検出割り込み（/WKP4、/WKP5）

一回転を検出したらタイマ V でカウントしていた TV_CNT を ROT_CNT へ移動し、TV_CNT をクリアします。

メインループ

メインではフラグ TV_FLG が立つのを待ち、立ったら目標カウント値と 1 回転カウント値を比較し PWM のデューティ比をコントロールします。

以上を元に変更したプログラム“DCM_06”のリストを以下に示します。

```
;---------------------------------------------------------
;
;   FILE        :DCM_06.src
;   DATE        :Thu, Sep 18, 2003
;   DESCRIPTION :Main Program
;   CPU TYPE    :H8/3687
;
;   This file is generated by Hitachi Project Generator (Ver.2.1
;
;---------------------------------------------------------
;ＰＷＭでのモータ回転・回転数監視（精度向上版）
;タイヤが１回転する間にタイマＶ割り込みが何カウントされたか
;で、スピードをコントロール　０～１００％まで０．１刻みでコン
;トロール

     .include  "io3687F_equ.inc"

     .export   _main
     .export   TV_CNT
     .export   TV_1000CNT
     .export   TV_FLG
     .export   ROT_CNT

ROT_CNST   .equ H'900          ;目標ｶｳﾝﾄ数

;=========================================================
;          メインプログラム
;=========================================================
_main:
;-----  イニシャライズ  ----------------------------------
     bsr       PWM_INIT:16     ;ﾀｲﾏ Z(PWM) ｲﾆｼｬﾗｲｽﾞ
     bsr       TV_INIT:16      ;ﾀｲﾏ V ｲﾆｼｬﾗｲｽﾞ
     bsr       WKP_INIT:16     ;P5(WKP) ｲﾆｼｬﾗｲｽﾞ

     xor.w     r0,r0
     mov.w     r0,@TV_CNT      ;ｸﾘｱ
     mov.w     r0,@ROT_CNT     ;回転数ｸﾘｱ
     mov.b     r0l,@TV_FLG     ;割り込みﾌﾗｸﾞ ｸﾘｱ
;    mov.b     r0l,@ROT_FLG    ;ﾌﾗｸﾞ ｸﾘｱ

     mov.w     #H'3000,r2      ;GRD 初期値
     mov.w     r2,@GRD_0
     mov.w     r2,@GRD_1

     andc      #H'7F,ccr       ;割り込み許可

     mov.b     #B'00000010,r0l ;PWM 動作開始
     mov.b     r0l,@TSTR

;-----  メインループ  ------------------------------------
?000:
     mov.b     @TV_FLG,r0l     ;ﾀｲﾏ V ｶｳﾝﾄｵｰﾊﾞｰ ﾁｪｯｸ
     and.b     r0l,r0l
     beq       ?000

     xor.w     r0,r0
     mov.b     r0l,@TV_FLG     ;割り込みﾌﾗｸﾞ ｸﾘｱ

?102:
     mov.w     #ROT_CNST,r0    ;R0 = 目標ｶｳﾝﾄ値
     mov.w     @ROT_CNT,r1     ;R1 = 実際ｶｳﾝﾄ値
     beq       ?110            ;R1=0=停止状態
     cmp.w     r0,r1           ;R1<>R0 目標値回転数との比較
     beq       ?000            ;一致ならそのまま
     bcs       ?120

?110:                          ;CC:Cy=0:回転数を上げる
     mov.w     @GRD_0,r0
     cmp.w     #D'20,r0
     bcc       ?112
     xor.w     r0,r0           ;最大回転
     bra       ?130
?112:
     sub.w     #D'20,r0        ;回転数を上げる
     bra       ?130

?120:                          ;CS:Cy=1:回転数を下げる
     mov.w     @GRD_0,r0
     cmp.w     #H'4E20-D'20,r0
     bcs       ?122
     mov.w     #H'4E21,r0      ;停止
     bra       ?130
?122:
     add.w     #D'20,r0        ;回転数を下げる
     bra       ?130

?130:
     mov.w     r0,@GRD_0
     mov.w     r0,@GRD_1
     bra       ?000

;---------------------------------------------------------
;          PWM(タイマＺ)イニシャライズ
;---------------------------------------------------------
PWM_INIT:
     mov.b     #B'00100000,r0l ;GRA ｺﾝﾍﾟｱﾏｯﾁｸﾘｱ CLK*1/1
     mov.b     r0l,@TCR_0
     mov.b     r0l,@TCR_1
     mov.b     #B'01100110,r0l ;FTIO-C0,D0,C1,D1 PWM Mode
     mov.b     r0l,@TPMR
     mov.b     #B'00000000,r0l ;初期出力 = all Low.
     mov.b     r0l,@TOCR
     mov.b     #B'00000110,r0l ;Hi active.
     mov.b     r0l,@POCR_0
     mov.b     r0l,@POCR_1
     mov.w     #H'4E20,r0      ;R0=GRA_0
     mov.w     #H'FFFF,r1      ;R1=GRC_0
     mov.w     r0,@GRA_0
     mov.w     r1,@GRC_0
     mov.w     r0,@GRA_1
     mov.w     r1,@GRC_1
     mov.b     #B'00110011,r0l ;出力ｲﾈｰﾌﾞﾙ
     mov.b     r0l,@TOER
     rts

;---------------------------------------------------------
;          タイマＶイニシャライズ
;---------------------------------------------------------
TV_INIT:
     xor.b     r0l,r0l
     mov.b     r0l,@TCRV0      ;CLK off
     mov.b     #D'125,r0l      ;TCORA=D'125=100usec(CLK*16)
     mov.b     r0l,@TCORA
     mov.b     #B'01001010,r0l ;ｺﾝﾍﾟｱﾏｯﾁ A 割り込み&ｸﾘｱ
     mov.b     r0l,@TCRV0      ;CLK*1/16
     mov.b     #B'00000000,r0l ;CLK*1/16
     mov.b     r0l,@TCRV1
     rts
```

【リスト 6-2　DCM_06.src（1/4)】

```
;---------------------------------------------------------
;          割り込みイニシャライズ
;---------------------------------------------------------
WKP_INIT:
     mov.b     #B'00100000,r0l
     mov.b     r0l,@PMR5       ;WKP5 enable
     mov.b     #B'00001111,r0l ;確認 LED の為
     mov.b     r0l,@PCR5       ;P50-P53:Out
     xor.b     r0l,r0l
     mov.b     r0l,@IEGR2
     mov.b     #B'00100000,r0l ;/WKP ｲﾈｰﾌﾞﾙ
     mov.b     r0l,@IENR1
     rts
```

```
;=========================================================
;          ワークエリア
;=========================================================
          .section   D,data

ROT_CNT         .res.w     1
TV_CNT          .res.w     1
TV_1000CNT      .res.w     1
TV_FLG          .res.b     1

;=========================================================
          .end
```

【リスト 6-2 DCM_06.src (2/4)】

```
;---------------------------------------------------------
;
;   FILE        :intprg.src
;   DATE        :Thu, Sep 18, 2003
;   DESCRIPTION :Interrupt Program
;   CPU TYPE    :H8/3687
;
;   This file is generated by Hitachi Project Generator (Ver.2.1
;
;---------------------------------------------------------

     .include    "vect.inc"
     .include    "io3687f_equ.inc"

     .import    TV_CNT
     .import    TV_1000CNT
     .import    TV_FLG
     .import    ROT_CNT

    .section    IntPRG, code

;1 Reserved
_INT_Reserved1
;2 Reserved
_INT_Reserved2
;3 Reserved
_INT_Reserved3
;4 Reserved
_INT_Reserved4
;5 Reserved
_INT_Reserved5
;6 Reserved
_INT_Reserved6
;7 NMI
_INT_NMI
;8 TRAP #0
_INT_TRAP0
;9 TRAP #1
_INT_TRAP1
```

```
;10 TRAP #2
_INT_TRAP2
;11 TRAP #3
_INT_TRAP3
;12 Address break
_INT_ABRK
;13 SLEEP
_INT_SLEEP
;14 IRQ0
_INT_IRQ0
;15 IRQ1
_INT_IRQ1
;16 IRQ2
_INT_IRQ2
;17 IRQ3
_INT_IRQ3
;18 WKP
_INT_WKP:
          push.w    r0                    ;R0 退避

          xor.w r0,r0
          mov.b r0l,@IWPR       ;/WKP ｽﾃｰﾀｽﾌﾗｸﾞ ｸﾘｱ
          mov.b #B'00110011,r0l
          mov.b r0l,@TOER

          mov.w @TV_CNT,r0      ;ﾀｲﾏ V ｶｳﾝﾄ数を
          mov.w r0,@ROT_CNT           ;ﾊﾞｯﾌｧへ移動

          xor.w r0,r0           ;ﾀｲﾏ V ｶｳﾝﾄ数
          mov.w r0,@TV_CNT      ;ｸﾘｱ

          mov.b @PDR5,r0l       ;確認 LED 点滅
          xor.b #H'01,r0l
          mov.b r0l,@PDR5

          pop.w r0                    ;R0 復帰
          rte

;19 RTC
_INT_RTC
```

【リスト 6-2 intprg.src (3/4)】

```
;20 Reserved
_INT_Reserved20
;21 Reserved
_INT_Reserved21
;22 Timer V
_INT_TimerV:
          push.w    r0                    ;R0 退避

          bclr #6,@TCSRV        ;TV ステータスフラグ クリア

          mov.w@TV_CNT,r0       ;タイマ V カウント数+1
          inc.w#1,r0
          mov.wr0,@TV_CNT
          cmp.w#H'FFFF,r0
          bne       ?000
          mov.w#H'FFFE,r0
          mov.wr0,@TV_CNT
?000:

          mov.w@TV_1000CNT,r0  ;100msec 用カウント(100u*1000)
          inc.w#1,r0
          mov.wr0,@TV_1000CNT
          cmp.w#D'1000,r0       ;1000 カウント終了?
          bne       ?100
          xor.wr0,r0
          mov.wr0,@TV_1000CNT
          mov.b#1,r0l               ;フラグ セット
          mov.br0l,@TV_FLG

          mov.b@PDR5,r0l        ;確認 LED 点滅
          xor.b#H'02,r0l
          mov.br0l,@PDR5
?100:

          pop.wr0                   ;R0 復帰
          rte
```

```
;23 SCI3
_INT_SCI3
;24 IIC2
_INT_IIC2
;25 ADI
_INT_ADI
;26 Timer Z0
_INT_TimerZ0
;27 Timer Z1
_INT_TimerZ1
;28 Reserved
_INT_Reserved28
;29 Timer B1
_INT_TimerB1
;30 Reserved
_INT_Reserved30
;31 Reserved
_INT_Reserved31
;32 SCI3_2
_INT_SCI3_2
    sleep
    nop
    .end
```

【リスト 6-2 DCM_06.src (4/4)】

★ご質問、お問い合わせはメール又は FAX で、、、

（株）東洋リンクス

〒102-0093　東京都千代田区平河町 1-2-2 朝日ビル

TEL: 03-3234-0559　／　FAX: 03-3234-0549

URL: http://www2.u-netsurf.ne.jp/~toyolinx

E-Mail: toyolinx@va.u-netsurf.jp

※本書の内容は将来予告無しに変更することがあります（2004 年 5 月作成）

20040720