無線自律移動ロボット車の製作

山形大学工学部 技術部
情報技術室　石谷 幹夫

1．はじめに

私が配属している共通講座大槻研究室では、実機を使った自律移動ロボット車の走行制御の研究を行っている。これまで用いてきた実機はLEGO・Mind Stormsで組み立てたロボット車である。このロボット車は仕様上、駆動力やステアリング角の微調整が困難なことや工作精度等の問題もあった。

本稿では、工作精度が高く、速度制御やステアリング角度制御が可能な、電動ラジコンカー（以降 R/C と表記）をベースにしたＨ８マイコン制御の無線自律移動ロボット車を製作したのでその要点を紹介する。

2．電動ラジコンカー（R/C）

無線自律移動ロボット車のベースマシンとして、Kyosho 社製の電動ラジコンカー「Mini-Z Racer」を採用した。（図１）

この R/C は、サイズが「130mm×65mm」と小型ながら、速度制御は勿論のこと、独自のマイクロサーボ機構によりステアリングの角度制御が出来る本格的なマシンである。 ホイールベースが 90mm で小回りが効き、比較的狭いスペースでも走行できる。 なお、駆動輪の最終減速比は最大の［7：1］とした。

図 1　ベースマシン[Mini-Z Racer]

2．1　R/C のモーター

R/C の動力には標準でマブチモーター社の FC-130RA が使用されていた。（図２）

図 2 FC-130RA

FC-130RA の仕様は、電圧 4.5V，無負荷，13200 rpm である。この時 R/C の速度は、最終減速比［7：1］，タイヤ直径［φ25mm］なので、秒速 約５m（18Km/h）である。

この速度は自律移動ロボット車としては速すぎるので、ここでは低速回転のモーターとして 並木精密宝石(株)の DC コアレスモーター（SCL10-12： 電圧 2.5V，無負荷，17500 rpm）と ギヤヘッド（SSG10-15：減速比 15：1）とを組み合わせた「ギヤードモーター」を使用することにした。（図３）

図 3　ギヤードモーター（イメージ）

R/C の最終減速比との組み合わせにより総減速比は 105：1 となるため、無負荷時の速度は秒速 約 0.437m（1.6Km/h）である。

ギヤードモーターの寸法は、直径φ10mm，本体長 27mm の円筒形であるが、FC-130RA のケースにピッタリ収まるサイズなので、ケースを改造してギヤードモーターを組み込み R/C に取り付けた。

2．2　R/C のサーボ／アンプの制御信号

今回採用した R/C には、ステアリング角度制御用の**サーボ**とモーター制御用の**アンプ**が備わっている。これらの装置は 図4(a) に示す PWM (Pulse Width Modulation) 信号により制御されており、制御時間長[Ts]の長さで装置の状態が変化する。Tsの可変範囲は1.5ms を中心に ±0.5ms である。

R/C を動作させるには、サーボ用／アンプ用の２つの PWM 信号が必要である。

2．3　R/C の制御電波

通常 R/C を操縦するために[**プロポ**]と呼ばれる送信機を使用する。 R/C 付属のプロポが送信する２チャンネル分 (サーボ用/アンプ用) の制御信号を乗せた電波のイメージを図4(d) に示す。on で発信、off で発信停止である。

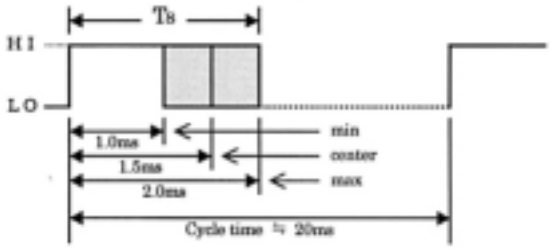
プロポは、 図4(b)に示すような PWM 信号の制御時間長[Ts]の先頭にパルス[tx]を付加した形の信号をチャンネル数分縦続接続して、図4(c) に示す PPM (Pulse Position Modulation) 信号を構築し、その信号で変調した電波を送信する。

R/C は、プロポから送信される PPM 変調された制御電波に乗っている複数チャンネルの PWM 信号を、専用の受信機によりサーボ用・アンプ用等のチャンネル毎に分離して各装置を駆動することで動作している。

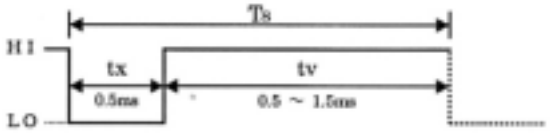
3．無線自律移動ロボット車の制御

R/C を自律移動ロボット車として動作させるために、 パソコン上で自律移動ロボット車の運動指令を生成し、それに応じた R/C の制御信号(PPM 信号)を H8 マイコンで作成し、送信モジュールから送信して制御する構成にした。　図5に制御系のブロック図を示す。

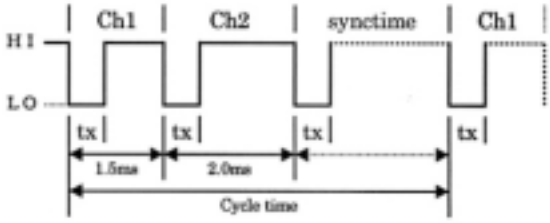
パソコンと送信モジュールとの間に H8 マイコンを介したのは、パソコンでは正確な



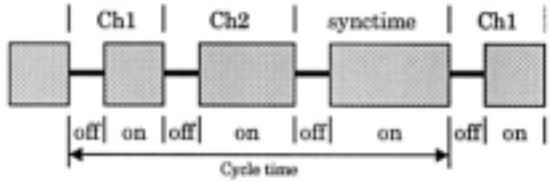
(a) サーボ／アンプの制御信号(PWM信号)



(b) PWM信号[Ts]の先頭にパルス[tx]を付加



(c) PPM信号（2つのPWM信号を縦続接続）



(d) PPM信号でAM変調された送信電波

図 4　R/C の制御信号と送信電波

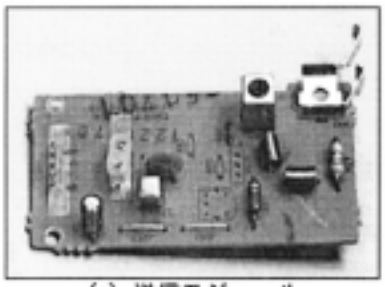
図 5　無線自律移動ロボット車の制御系

PPM 信号の発生が困難であり、また送信モジュールに直接接続できないためである。

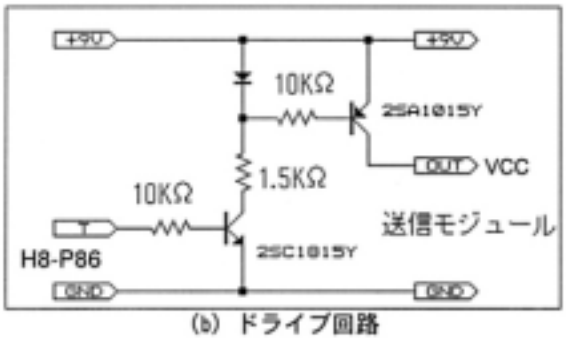
H8 マイコンには、秋月電子通商が販売する「AKI-H8 / 3664F(QFP) タイニーマイコンキット」を使用した。　H8 マイコンとパソコンとの通信は RS-232C インターフェースによるシリアル接続で行っている。

3．1　送信モジュール

R/C の制御電波を送信するために、R/C に付属するプロポ内の送信モジュールを分解して使用した。(図 6(a))



(a) 送信モジュール



(b) ドライブ回路

図 6　送信モジュールとドライブ回路

送信モジュールは、図 4 (d) に示すような PPM 信号で変調された 27MHz 帯のＡＭ電波を送信する。　調査の結果、電波の変調はモジュールの VCC 電源(9～12V) を ON/OFF して行っていた。　しかし、H8 マイコンの出力電圧は 5V であり、 電流も最大で 20mA しか取れないので、 送信モジュールを直接駆動することができない。　そこで、Ｈ８マイコンと送信モジュールとの間に 図 6 (b) に示す簡単なドライブ回路を入れて接続した。

3．2　H8 マイコンによる PPM 信号の発生

PPM 信号は H 8 /3664F 内蔵のタイマ W の PWM 動作モードを使って作り出している。

タイマ W は、タイマカウンタ (TCNT) と４つの周期レジスタ (GRA,GRB,GRC,GRD) を持ち、PWM 動作モード時の各レジスタは以下の機能を持つ。

GRA :　周期レジスタ
GRB :　コンペアマッチ・レジスタ
GRC :　コンペアマッチ・レジスタ
GRD :　コンペアマッチ・レジスタ

タイマカウンタ(TCNT)のカウント値と各レジスタ(GRA,GRB,GRC,GRD)の設定値が等しい時 (コンペアマッチ) に、内部割込みを発生させることができ、さらに TCNT と GRA とのコンペアマッチ時には TCNT がゼロクリアされる。

ここでは、周期用の GRA、および、２つの PWM 信号用に GRB, GRC のコンペアマッチ割込みを利用して PPM 信号を発生させた。

図７にタイマＷを使ったＰＰＭ信号の発生の様子を示す。　ここで、GRB, GRC の値を適宜変更することで Ts1, Ts2 の時間を変化させる事ができる。

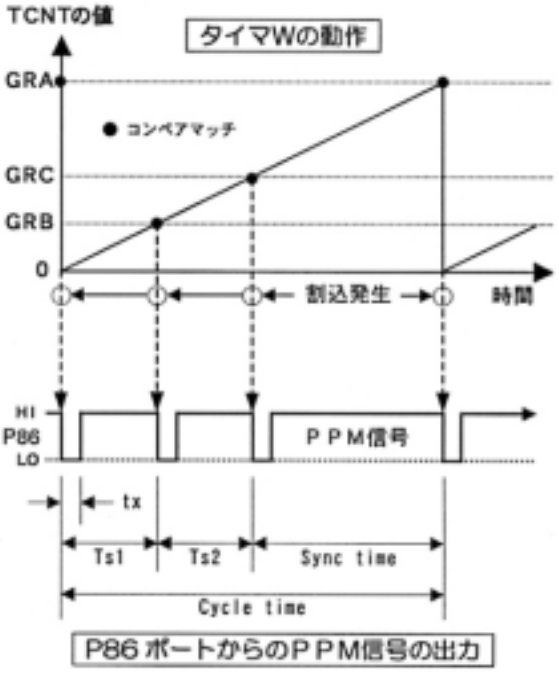


図 7　タイマＷを使ったＰＰＭ信号の発生

PPM 信号は H8 マイコンの I/O ポート 8 のビット 6（P86）から出力するようにプログラムして、図 6 (b) の送信モジュールのドライブ回路へ接続した。

PPM 信号の出力は、GRA,GRB,GRC の各コンペアマッチ割り込み発生時に P86 出力ポートを‘LO’レベルにし、プログラムループで PPM 信号のパルス[tx] 時間経過させた後に‘HI’レベルに戻すことで行っている。

ここで、GRC の設定値は Ts2 時間のカウント値に Ts1 時間のカウント値(GRB 値) を加えた値であるので、注意が必要である。

リスト 1 に H8/3664 用に日立開発環境システムが提供していた Tiny/スーパーローパワー用無償版コンパイラ「HEW2」によるプログラムの抜粋を示す。

4．さいごに

電動ラジコンカー「Mini-Z Racer」をベースマシンに用いたＨ８マイコン制御による無線自律移動ロボット車の製作の要点について紹介した。　皆様の参考になれば幸いである。

【謝辞】

日頃から御助言と御協力を頂いている共通講座大槻恭士助教授に感謝申し上げます。

参考文献

H8/3664 シリーズ ハードウェアマニュアル

H8/3664 シリーズ プログラミングマニュアル

参考ＵＲＬ：

Kyosho：
http://www.kyosho.co.jp/

並木精密宝石：
http://www.namiki.net/

マブチモーター：
http://www.mabuchi-motor.co.jp/

秋月電子通商：
http://akizukidenshi.com/

リスト１　HEW2 によるプログラムの抜粋

```
/* hwsetup.c */
/* タイマWの初期設定関数 */
void TwSetup( void )
{
  TW.TCNW.BIT.CKS  = 3;   /* TCNT Clock = 16MHz/8 */
                          /* 0.5 [msec/count]     */
  TW.TCNW.BIT.CCLR = 1;   /* TCNT クリア */
  TW.TMDW.BIT.PWMB = 1;   /* PWM mode B */
  TW.TMDW.BIT.PWMC = 1;   /* PWM mode C */
  TW.TCNW.BIT.TOB  = 1;
  TW.TCNW.BIT.TOC  = 1;
  TW.GRA = 32000;         /* 1周期：16msec  */
  TW.TIERW.BIT.IMIEA = 1; /* GRA 割込み 許可 */
  TW.TIERW.BIT.IMIEB = 1; /* GRB 割込み 許可 */
  TW.TIERW.BIT.IMIEC = 1; /* GRC 割込み 許可 */
}

void IOSetup(void)
{
  IO.PCR8 |= 0xC0;      /* P87 and P86 Output Terminal Function */
  IO.PDR8.BIT.B6 = 1;   /* HI level */
}

/*------------------------------------------------------*/
/* intprg.c */
/* 割り込みベクタと処理ルーチンの定義 */
// vector 21 Timer W
void twint(void);
_interrupt(vect=21) void INT_TimerW(void) { twint(); }

/*------------------------------------------------------*/
/* main.c */
/* PPMパルス「tx」作成関数 */
void OneShot(void)
{
  int n = 1300;   // PPM_PULSE_TIME: 0.5 [us]
  IO.PDR8.BIT.B6 = 0;
    while( n > 0 ){ --n; }  /* wait loop */
  IO.PDR8.BIT.B6 = 1;
}

/* タイマWの割り込み処理関数*/
void twint(void)
{
  switch( TW.TSRW.BYTE & 0x07 )
  {
    case 1:   /* IMFA(GRA) */
    case 2:   /* IMFB(GTB) */
    case 4:   /* IMFC(GRC) */
      OneShot();
      break;
  }
  TW.TSRW.BYTE = 0;         /* 割込みフラグ クリア */
}

/* メイン関数 */
void main(void)
{
  short grb = 3000;   /* 1.5ms */
  short grc = 3000;   /* 1.5ms */
  // タイマW 初期化
  TW.GRB = grb;
  TW.GRC = grb + grc;
  TW.TCNT = 0;
  TW.TMDW.BIT.CTS = 1;      /* Start TimerW conter */

  while(1)
  {
    /* 変数 grb、grc の値を更新してレジスタにセットする */
    TW.GRB = grb;
    TW.GRC = grb + grc;
  }
}
```