工学教育用高機能ロボットの開発

小松崎　年　雄*・横　地　弓　夫**

Development of High Function Robots for Engineering Education

Toshio KOMATSUZAKI* and Yumio YOKOCHI**

Abstract

The foundation of the robotics consists of the knowledge of the electronics, information and mechanical engineering. Therefore, the elementary knowledge of the electronics, information and mechanical engineering can be learned by making suitable robots. From this viewpoint, we developed a robot for the education, which can detect only obstacles in front in the previous paper. In this paper, by improving this robot, we show that we can easily make a robot which can detect the left-right obstacles and avoid it. Next, we put a sound sensor on this robot and it is made to have the interrupted function of the sound.

Key words: Robot, Interruption, Sound Sensor

1. はじめに

ロボット工学は，電気電子，情報および機械(機構学)工学の総合領域であり，適切な教育用ロボットを開発すれば，その「ロボット作り」を通して電気電子，情報および機械工学の基礎を学ぶことができる。前報告[1]では，その観点のもと，次の条件を満たすロボットを開発した。

1) 電気回路がシンプルでやさしく理解しやすい。また，電子部品になじめる。
2) 制作費が安く，一人一台づつ製作できる。
3) やさしく理解できるC言語，およびフリーソフトのCコンパイラを使用する。
4) 発展性がある。
5) 部品はリサイクルが可能である。

本報告では，前報告[1]で開発したロボットの条件4) の応用発展性について述べる。すなわち，前報告で述べたロボットは前方のみの障害物しか検出できなかったが，本報告ではそのロボットに赤外線発光ダイオードを1個追加し，さらにプログラムを少し変えることにより左右の障害物を感知し，また障害物を左右回転により回避することができることを示す。次に，音センサを取り付けて増幅回路，3端子レギュレータ回路を加え，さらにプログラムに音の割り込み処理を持たせた高機能のロボットについて述べる。

2. 左右の障害物を回避するロボット

本節では，前報告で述べたロボットの発展したものとして，検知用の赤外線発光ダイオード2個により左右の障害物を検知でき，また障害物を回避する行動も左右回転するロボットについて述べる。ワンチップマイコンは，PIC16F84[2]を用いた。

平成14年12月26日受理
* 電気電子工学科・教授
** 電気電子工学科・助教授

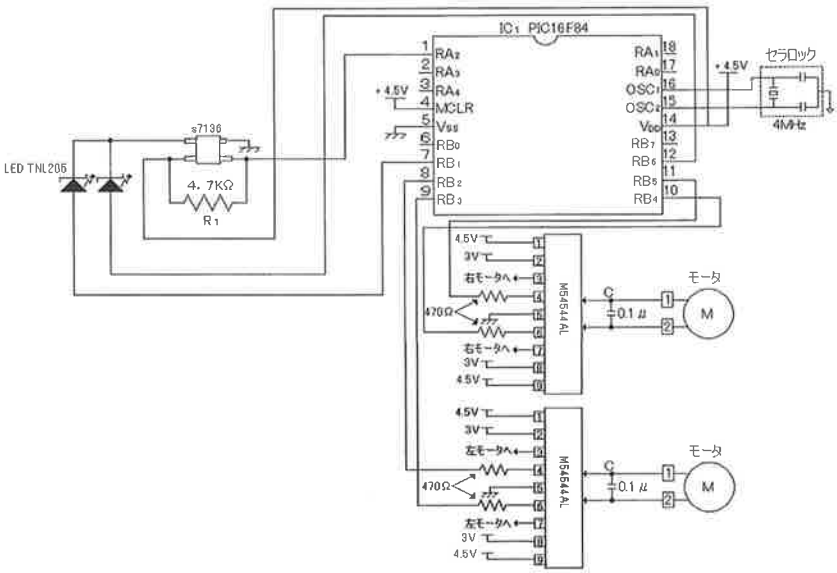

図1　左右の障害物回避ロボットの電気回路図

2.1　電気回路

(1)　全体の回路図

全体の回路図を図1に示す。

次に，各部について述べる。

(2)　センサまわりの回路

前報告で述べたロボットとの回路の違いは，このセンサまわりである。すなわち，前報告で述べたロボットの赤外線発光ダイオードの電源は，直接4.5ｖ電源から取った。ここでは，図1からわかるように2個の赤外線発光ダイオードの電源をワンチップマイコンPICのRB1ポート，RB6ポートよりとっている。また，はんだ付けによりセンサの検知距離が影響を受けるので，赤外線発光ダイオードとして検知距離の長いTLN205を用いた。

センサまわりの回路のスケッチ図を図2に示す。

(3)　PIC，モータドライブまわりの回路

上述のように赤外線発光ダイオードの電源をPICからとっているだけで，他は前報告のロボットの回路と同じである。センサ部の回路を

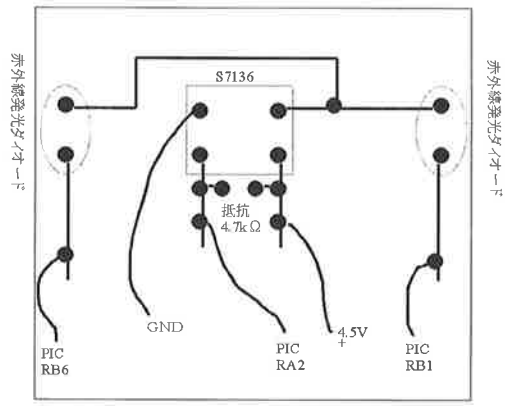

図2　センサまわりの回路図のスケッチ

一つの基板に配線したので，PICおよびモータドライブまわりの回路を一つの基板に配線した。

2.2　プログラム

プログラムは，左右の赤外線発光ダイオードを交互に発光させている。左の赤外線発光ダイオードが発光しているときにセンサが検知したときは，左に障害物があるのでバックした後に

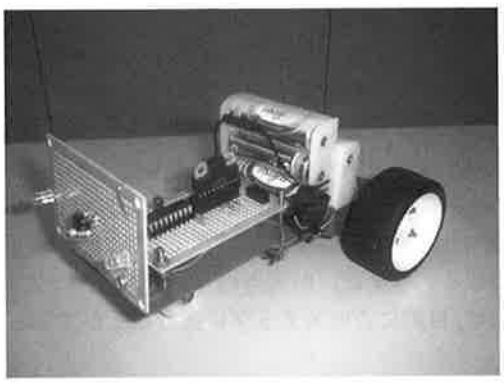

図3　左右の障害物回避ロボット完成図

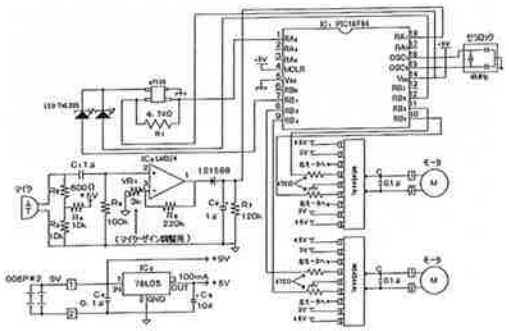

図4　割り込み処理を持つロボットの電気回路図

右に回転し，その後前進して障害物をさける行動をとる。また，右の赤外線発光ダイオードが発光している場合も同様で，このときはバック後に左に回転するようにプログラムしている。

プログラムを付録1に示す。

2.3　ロボットの組み立て

完成したロボットを図3に示す。

3.　音センサの割り込みを持つロボット

この節では，2節で述べたロボットにコンデンサマイクを取り付け，音による割り込み処理を持つロボットについて述べる。この機能により，どのような状態でも音の入力があったときには，あらかじめ決められた動作を優先的に行う。

3.1　電気回路

(1)全体の回路図

全体の回路図を図4に示す。

次に，各部について述べる。

(2)　センサ部回路

センサとしては，2節のロボットの検知用の赤外線発光ダイオード TLN205 を2個と赤外線センサ S71361 を個の他にコンデンサマイクを取り付け，これにより音を検知している。コンデンサマイクからの出力電流は微少なので，IC LM324 により増幅している。LM324 の電源は5Vなので3端子レギュレータにより5Vをとりだしている。

(3)PIC，モータドライブまわりの回路

PIC まわり，モータドライブまわりの回路は，2節の赤外線発光ダイオードを2個用いたロボットと同様である。ただし，コンデンサマイクからの信号が RA1 ポートに，また赤外線センサからの信号が RA2 に入力している。

3.2　プログラム

赤外線センサに対しては，2節で述べたロボットと同様に，左の赤外線発光ダイオードが発光しているとき赤外線センサが感知したときは右に回避し，右の赤外線発光ダイオードが発光しているとき感知したときは左に回避するようにプログラムしている。

一方，音センサーの入力に対しては，時間割り込みを用いている。すなわち，音入力があった場合，ロボットは優先的にバックするようにプログラムしている。

プログラムを付録2に示す。

3.3　ロボットの組み立て

完成したロボットを図5に示す。

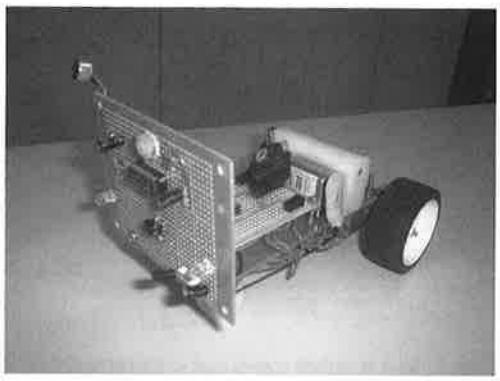

図5　割り込み処理を持つロボットの完成図

4. おわりに

本論では，前論で開発した正面の障害物を避ける教育用ロボットを発展させたロボットについて述べた。すなわち，はじめに赤外線発光ダイオードを2個取り付けて左右の障害物を避けるロボットについて，次にセンサとしてコンデンサマイクを取り付けて音の割り込み処理を持つ高機能のロボットについて述べた。これらは，電気回路的にも，またプログラムも自然に発展できる。ゆえに，前報告のロボットを終えた学生は，自然にこのステップに進むことができる。

参考文献

1) 横地，小松崎：工学教育用ロボットの開発，八戸工業大学紀要に掲載予定
2) 後閑哲也：PIC活用ガイドブック，技術評論社，平成12年

付録1

```
# include<pic.h>

void kaihi1(void);      //回避関数の宣言
void kaihi2(void);
void zensin(void);
void hantei(void);
void delay(long t);   //待ち時間関数の宣言

void main()
{
    TRISA = 1;      //A ポートを入力用に設定
    TRISB = 0;      //B ポートを出力用に設定

    RA2 = 1;        //赤外線センサーを初期化
    RB1 = 0;       //LED 出力用ポートの初期
    RB6 = 0;
    RB2 = 0;        //モータ用ポートを初期化
    RB3 = 0;
    RB4 = 0;
    RB5 = 0;

    while(1)
        {
            hantei();
        }
    }

//判定動作
void hantei(void)
{
            RA2 = 1;
            RB1 = 0;              //in
            RB6 = 1;              //out
            delay(5000);
              if(RA2 == 0){ //左側障害物検知
                    kaihi1();
                          }
              else zensin();   //障害物無前進

            RA2 = 1;              //初期化
            RB1 = 1;              //out
            RB6 = 0;              //in
            delay(5000);
              if(RA2 == 0{ //右側障害物検知
                    kaihi2();
```

```
                    }
            else zensin();   //障害物無前進
  }

  //前進関数
void zensin(void)
{
            RB2 = 0;        RB3 = 1;
            RB4 = 0;        RB5 = 1;
}

  //回避関数 1
void kaihi1(void)
{
        //後退
            RB2 = 1;        RB3 = 0;
            RB4 = 1;        RB5 = 0;
              delay(60000);
        //左に曲がる
            RB2 = 0;        RB3 = 0;
            RB4 = 0;        RB5 = 1;
            delay(60000);
  }

  //回避関数 2
void kaihi1(void)
{
        //後退
            RB2 = 1;        RB3 = 0;
            RB4 = 1;        RB5 = 0;
            delay(60000);
        //右に曲がる
            RB2 = 0;      RB3 = 1;
            RB4 = 0;      RB5 = 0;
            delay(60000);
  }

  //待ち時間関数
void delay(long t)
{
  long i;
  for(i = 0;i<= t;i＋＋);
}
```

付録 2

```
# include <16f84.h>
# use delay(clock = 10000000)
# define SECMAX 38
# byte port_a = 5
# byte port_b = 6
# int sec,flag,zen_tmp,ir_tmp;
void zenshin(void);
void koutai(void);
void turn_left(void);
void turn_right(void);
void stop(void);
int check_ir(void);
int check_mic(void);

# INT_RTCC        //タイマ 0 割り込みの指定
rtcc_isr() {          //タイマ 0 割込処理関数
    if(check_mic()== 1){
            output_low(PIN_B7);
            koutai();
            delay_ms(1000);
            output_high(PIN_B7);
                    }
            }

main()
      {
    set_tris_a(0xff);
    set_tris_b(0);
setup_counters(RTCC_INTERNAL,
    RTCC_DIV_256);
                    //タイマ 0 のモード設定
    sec = SECMAX;       //タイマ 0 初期設定
    enable_interrupts(RTCC_ZERO);
    enable_interrupts(GLOBAL);
          //グローバル割り込み許可
```

```
    while(1){
        if(input(PIN_A2)== 0){
                output_low(PIN_B5);
                                }
    else{
        output_high(PIN_B5);
        }
    }

void zenshin(void)
{
    zen_tmp = port_b & 0xc3;
    port_b = zen_tmp | 0x28;
}

void koutai(void)
{
    zen_tmp = port_b & 0xc3;
    port_b = zen_tmp | 0x14;
}

void turn_left(void)
{
    zen_tmp = port_b & 0xc3;
    port_b = zen_tmp | 0x08;
}

void turn_right(void)
{
    zen_tmp = port_b & 0xc3;
    port_b = zen_tmp | 0x20;
}

void stop(void)
{
    zen_tmp = port_b & 0xc3;
    port_b = zen_tmp | 0x00;
}

int check_ir(void)
{
    zen_tmp = port_b & 0xbd;
    port_b = zen_tmp | 0x40;
    delay_ms(10);
    if( (port_a & 0x4) != 0 )
        ir_tmp = 1;
    else
        ir_tmp = 0;

    port_b = zen_tmp | 0x00;
    delay_ms(10);

    port_b = zen_tmp | 0x02;
    delay_ms(10);
    if( (port_a & 0x4) != 0 )
        ir_tmp += 2;

    port_b = zen_tmp | 0x00;
    return(ir_tmp);
}

int check_mic(void)
{
    if( (port_a & 2) == 0)
        return(0);
    else
        return(1);
}
```