19 他ボードへの移植

この章では、今まで作ってきた関数を、他の CPU ボードに移植することを考えます。

関連する関数とサンプルプログラムを示します。どの程度の変更が必要か、変更を少なくするにはどのように改良すればよいかなどを考えながら読み進めてください。

レジスタ定義ファイルなどの詳細は示しませんでしたが、それぞれマニュアルをあたってください。

19.1 M16C への移植

OAKS16-LCD Board は OAKS16-62P BoardKit 用の書込み・拡張 I/O ボードです。オークス電子株式会社の製品です。

マイコンボード OAKS16-62P BoardKit には、CPU として M30620FCAFP が搭載されています。

OAKS16-LCD Board には LED8 個、スイッチ 8 個、割込みスイッチ 2 個、16 文字 ×2 行の LCD1 個、RS232C の D サブコネクタなどが実装されています。

このボードのための各種学習テキストがオークス電子株式会社のホームページで公開されています。開発環境の使い方から、C 言語のプログラムに至るまで、かなり丁寧な説明がなされているので、非常に勉強になります。

ここでは、このボードにこのテキストで作ってきた関数を移植し、サンプルプログラムを作ってみましょう。

19.1.1 M16C/62 の A/D 変換サンプルプログラム 1 (単発モード)

プログラム内容

- PSD センサ GP2D12 を利用
- AN4 を利用
- 8 ビット A/D 変換の結果をシリアル通信に送信
- PSD センサの計測待ちが必要な場合には、タイマ A を利用

設定におけるの注意点

- PSD センサの計測には約 55ms かかる
- 繰り返しモードでは、CPU の内部クロックは、メインクロックを分周せずに使う
- f_{AD} は 10MHz 以下にする
- AN4(P104) の方向レジスタは、入力に設定

設定内容

設定項目	設定内容
動作クロック ϕ_{AD}	$f_{AD}/4(=8MHz)$
分解能	8 ビット
アナログ入力端子	AN4
A/D 変換開始条件	ソフトウェアトリガ
拡張アナログ入力端子	使用しない
サンプルアンドホールド	使用する

サンプルプログラム

では PSD 変換の出力を (増幅せず)、8 ビットの分解能で A/D して、結果をシリアル通信で送信してみましょう。

ターミナルソフトで値を確認します。

sci.h

```
 1  #ifndef _SCI_H_
 2  #define _SCI_H_
 3
 4  #define SCI_RX_BUFF_SIZE 32
 5  #define SCI_RX_BUFF_MASK (SCI_RX_BUFF_SIZE-1)
 6
 7  typedef enum {
 8    br4800=207,
 9    br9600=103,
10    br14400=68,
11    br19200=51,
12    br28800=34,
13    br31250=31,
14    br38400=25,
15    br57600=16
16  } BaudRate;
17
18  typedef struct{
19    unsigned short start_index;
20    unsigned short data_count;
21    char data_buff[SCI_RX_BUFF_SIZE];
22  }SciRxBuffer;
23
24  void Sci1Init(BaudRate b);
25  int Sci1Putchar(char c);
26  int Sci1Puts(char _far *str);
27  int Sci1ReadTerm(void);
28
29  char Sci1Getchar(void);
30  void Sci1Gets(char _far *str);
31
32  #endif
```

End Of List

sci.c

```
/*****************************************************/
/*      ファイル名 :    sci.c                        */
/*      内容 :          シリアル通信用関数           */
/*      作成者 :        ono                          */
/*****************************************************/

/* インクルードファイル */
#include<oaks_sfr.h>       /* OAKS 用定義ファイル */
#include "sci.h"

#pragma INTERRUPT INT_SCI1_RXI1

#define CHECK_LOOP 0x200000
#define CR 0x0D
#define LF 0x0A

static SciRxBuffer rx1_buff;

/************************************************************************
*    uart1 初期設定関数 : Sci1Init()
************************************************************************/
void Sci1Init(BaudRate b)
{
  u1mr = 0x05;       /* 送受信モードレジスタ 内部クロック */
                     /* 非同期、8 ビット、パリティなし、*/
                     /* スリープなし */

  u1c0 = 0x10;       /* 送受信制御レジスタ 0 クロック f1 選択 */
  u1brg = b;         /* 転送速度レジスタ */

  s1ric = 0x06;      /* 割込みレベルの設定 (レベル 6) */
  u1c1 = 0x05;       /* 送受信制御レジスタ 1 送受信許可 */
  ucon = 0x00;       /* 送受信制御レジスタ 2 */
}

/************************************************************************
*    1 バイト送信 : Sci1Write()
************************************************************************/
int Sci1Write(char c)
{
  unsigned long i;
  unsigned short write_flag=0;

  /* 送信レジスタが空くまでポーリングで待つ */
  for(i=0; i<CHECK_LOOP; i++){
    if(ti_u1c1){
      write_flag=1;
      break;
    }
  }

  if(write_flag){
    u1tb=c;
    ti_u1c1=0;
    return(0);
  }
  return(-1);
}

/************************************************************************
*    1 バイト受信 : Sci1Read()
************************************************************************/
char Sci1Read(void)
{
  char read_char;
  while(!ri_u1c1){
    ;
  }
  read_char=u1rb;
  ri_u1c1=0;
  return(read_char);
}

/*******************************
   1 バイト受信関数 (割り込み有り)
*******************************/
void INT_SCI1_RXI1(void)
{
  char msg;
  int index;

  if(ri_u1c1){
    ri_u1c1=0;
    msg=u1rbl;
    if(rx1_buff.data_count < SCI_RX_BUFF_SIZE){
      index=(rx1_buff.start_index+rx1_buff.data_count) & SCI_RX_BUFF_MASK;
      rx1_buff.data_buff[index]=msg;
      rx1_buff.data_count++;
    }
```

```
 90    }
 91  }
 92
 93  int Sci1Rx1(char _far *msg)
 94  {
 95    if(rx1_buff.data_count > 0){
 96      *msg=rx1_buff.data_buff[rx1_buff.start_index];
 97      rx1_buff.start_index=(rx1_buff.start_index+1) & SCI_RX_BUFF_MASK;
 98      rx1_buff.data_count--;
 99      return(0);
100    }
101    *msg=0;
102    return(-1);
103  }
104
105  /***************************************
106     受信バッファクリア関数 :Rx1BuffClear
107  ***************************************/
108  void Rx1BuffClear(void)
109  {
110    rx1_buff.start_index=0;
111    rx1_buff.data_count=0;
112  }
113
114  /*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
115     汎用関数
116  +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
117
118  /*****************************************************************************
119  *     1 文字送信 : Sci1Putchar()(改行 (\n) は CR+LF に変換)
120  *****************************************************************************/
121  int Sci1Putchar(char c)
122  {
123
124    /* 改行は CR+LF に変換して送信 */
125    if(c=='\n'){
126      if(Sci1Write(CR)){
127        return(-1);
128      }
129      if(Sci1Write(LF)){
130        return(-1);
131      }
132    }else{  /* それ以外はそのまま送信 */
133      if(Sci1Write(c)){
134        return(-1);
135      }
136    }
137    return(0);
138  }
139
140  /*****************************************************************************
141  *     文字列送信 : Sci1Puts()(改行 (\n) は CR+LF に変換)
142  *****************************************************************************/
143  int Sci1Puts(char _far *str)
144  {
145    unsigned char _far *msg;
146
147    /* 終端文字まで 1 文字ずつ表示 */
148    for(msg=str; *msg!='\0'; msg++){
149      if(Sci1Putchar(*msg)){
150        return(0);
151      }
152    }
153    return(msg-str);
154  }
155
156  /************************************************
157  *     1 文字受信 (CR は改行 (\n) に変換) : Sci1Getchar
158  ************************************************/
159  int Sci1Getchar(void)
160  {
161    char c;
162    while(Sci1Rx1(&c)){
163      ;
164    }
165    if(c==CR){
166      c='\n';
167    }
168    return(c);
169  }
170
171  /************************************************
172  *     文字列受信 (CR は改行 (\n) に変換) : Sci1Gets
173  ************************************************/
174  void Sci1Gets(char _far *str)
175  {
176    int i=0;
177    char s;
178
179    while((s=Sci1Getchar()) != '\n'){
180      str[i]=s;
181      i++;
182    }
```

```
183   str[i]='\0';
184 }
185
```

End Of List

str.h

```
 1 #ifndef _STR_H_
 2 #define _STR_H_
 3
 4 int IsDigit(char c);
 5 int IsHex(char c);
 6 int IsEos(char c);
 7 int StrDecToLong(char _far *str, long _far *n);
 8 int StrHexToLong(char _far *str, long _far *n);
 9 int StrLongToDec(long n,char _far *str);
10
11 #endif
```

End Of List

str.c

```
 1 #include "str.h"
 2
 3 /*******************************
 4    10 進数文字かチェック
 5 *******************************/
 6 int IsDigit(char c)
 7 {
 8   if(c>='0' && c<='9'){
 9     return(1);
10   }
11   return(0);
12 }
13
14 /*******************************
15    16 進数文字かチェック
16 *******************************/
17 int IsHex(char c)
18 {
19   if(c>='0' && c<='9'){
20     return(1);
21   }else if(c>='a' && c<='f'){
22     return(1);
23   }else if(c>='A' && c<='F'){
24     return(1);
25   }
26   return(0);
27 }
28
29 /*******************************
30    「\n」あるいは「\0」文字かのチェック
31 *******************************/
32 int IsEos(char c)
33 {
34   if(c=='\n' || c=='\0'){
35     return(1);
36   }
37   return(0);
38 }
39
40 /*******************************
41    10 進数文字から数値に変換
42 *******************************/
43 int StrDecToLong(char _far *str, long _far *n)
44 {
45   int minus=0;
46
47   /* マイナスかをチェック */
48   if(*str == '-'){
49     minus=1;
50     str++;
51   }
52
53   /* 10 進数文字を数値に変換 */
54   for(*n=0; IsDigit(*str); str++){
55     *n *= 10;
56     *n += (*str - '0');
57   }
58
59   /* マイナスの場合 */
60   if(minus){
61     *n = -1*(*n);
```

```
62    }
63
64    /* 終端文字まで達していない場合 */
65    if(!IsEos(*str)){
66      return(0);
67    }
68    return(1);
69  }
70
71  /*******************************
72     16 進数文字から数値に変換
73  *******************************/
74  int StrHexToLong(char _far *str, long _far *n)
75  {
76
77    /* 16 進数文字を数値に変換 */
78    for(*n=0; IsHex(*str); str++){
79      *n <<= 4;
80      if(*str >= '0' && *str <= '9'){
81        *n += (*str - '0');
82      }else if(*str >= 'a' && *str <= 'f'){
83        *n += (*str - 'a' + 10);
84      }else if(*str >= 'A' && *str <= 'F'){
85        *n += (*str - 'A' + 10);
86      }
87    }
88
89    /* 終端文字まで達していない場合 */
90    if(!IsEos(*str)){
91      return(0);
92    }
93    return(1);
94  }
95
96  /*******************************
97     数値を 10 進数文字に変換
98  *******************************/
99  int StrLongToDec(long n, char _far *str)
100 {
101   char dec[11]="0123456789";
102   char c;
103   int len=0;
104   long i, half;
105
106   /* 10 進数文字列へ変換 */
107   do{
108     str[len]=dec[n%10];
109     len++;
110     n = n/10;
111   }while(n != 0);
112
113   /* 文字列の順番を入れ替え */
114   half = (len >>1);
115   for(i=0; i<half; i++){
116     c=str[i];
117     str[i]=str[(len-1)-i];
118     str[(len-1)-i]=c;
119   }
120
121   /* 終端文字を挿入 */
122   str[len]='\0';
123   return(len);
124 }
```

End Of List

ad.h

```
1  #ifndef _AD_H_
2  #define _AD_H_
3
4  void AdIinit(void);
5  void AdStart(void);
6  void AdStop(void);
7  unsigned short AdRead(void);
8
9  #endif
```

End Of List

ad.c

```
1  /*********************************************************************/
2  /*                                                                   */
```

```
 3  /*  FILE        :ad.c                                                 */
 4  /*  DESCRIPTION :A/D 変換用関数                                       */
 5  /*                                                                    */
 6  /********************************************************************/
 7
 8  #include <oaks_sfr.h>
 9  #include "ad.h"
10
11  #pragma INTERRUPT INT_AD
12
13  unsigned int ad_flag=0;
14
15  /*******************************
16     AD のつながった端子を初期化
17  *******************************/
18  void AdInit(void)
19  {
20    adcon0=0x84;
21    /* AN4, 単発モード, ソフトウェアトリガ,A/D 変換停止,fAD/2 */
22    adcon1=0x20;    /* 8 ビット,Vref 接続,ANEX0,ANEX1 は使用しない */
23    adcon2=0x01;    /* サンプル&ホールドあり */
24    ir_adic=1;      /* A/D 割り込みあり */
25    adic=0x06;     /* 割り込みレベル設定 */
26  }
27
28  /*******************************
29     AD 変換開始
30  *******************************/
31  void AdStart(void)
32  {
33    adst=1;  /* A/D 変換開始 */
34  }
35
36  /*******************************
37     AD 変換中止
38  *******************************/
39  void AdStop(void)
40  {
41    adst=0;  /* A/D 変換停止 */
42  }
43
44  /*******************************
45     AD 変換読み取り
46  *******************************/
47  unsigned short AdRead(void)
48  {
49    unsigned short ad_r;
50
51    while(!ad_flag){
52      ;
53    }
54    ad_flag=0;
55    ad_r=ad4l;
56
57    return(ad_r);
58  }
59
60  /*******************************
61     AD 変換割り込み (終了)
62  *******************************/
63  void INT_AD(void)
64  {
65    ad_flag=1;
66  }
```

End Of List

ad01.c

```
 1  /*------------------------------------------------------*/
 2  /*  ファイル名：  ad01.c                                */
 3  /*  内容：        8 ビット AD 変換                       */
 4  /*  作成者：      ono                                   */
 5  /*------------------------------------------------------*/
 6
 7  #include<oaks_sfr.h>
 8  #include "ad.h"
 9  #include "sci.h"
10  #include "str.h"
11
12  #define STR_LEN 3 /* 表示桁数 */
13
14  char str[6];
15  unsigned short n;
16  int len, i;
17
18  void main(void)
19  {
20    Sci1Init(br38400);
21    AdInit();
```

```
22    Sci1Puts("**************************\n");
23    Sci1Puts("A/D 変換の結果\n");
24    Sci1Puts("**************************\n");
25    while(1){
26      AdStart();
27      n=AdRead();
28      len=StrLongToDec(n, str);
29      for(i=0; i<STR_LEN-len; i++){
30        Sci1Putchar('0');
31      }
32      Sci1Puts(str);
33      Sci1Putchar(0x0D);
34    }
35  }
36
```

End Of List

ad01_sect30.inc

```
 1      .lword    dummy_int     ; Key input interrupt(for user)(vect 14);
 2      .glb _adint
 3      .lword    _adint        ; A-D(for user)(vector 14)
 4      .lword    dummy_int     ; uart2 transmit(for user)(vector 15)
 5      .lword    dummy_int     ; uart2 receive(for user)(vector 16)
 6      .lword    dummy_int     ; uart0 transmit(for user)(vector 17)
 7      .lword    dummy_int     ; uart0 receive(for user)(vector 18)
 8      .lword    0fcb6bh       ; uart1 transmit(for user)(vector 19)
 9      .lword    0fcb6bh       ; uart1 receive(for user)(vector 20)
10      .glb _ta0int
11      .lword    _ta0int       ; timer A0(for user)(vector 21)
12      .lword    dummy_int     ; timer A1(for user)(vector 22)
```

End Of List

メインクロックを分周せずに使う (C:/MTOOL/SRC30/STARTUP/ncrt0.a30 の設定を変更)

ad01_ncrt0.a30

```
 1  mov.b #02h,0ah
 2  ; bset 1,0ah
 3  mov.b #00h,04h ;set processer mode
 4  ; bclr 1,0ah
 5  mov.b #00h,0ah
 6  ; 記述がない場合には、分周なしを追加 -----
 7  mov.b #01h,0ah
 8  mov.b #08h,06h
 9  mov.b #00h,0ah
10  ;---------------------------------------
```

End Of List

実行結果

- A/D 変換の結果がターミナルソフトに表示される。
- 手をかざして、値の変化を調べる。
- 表示がちらついて見にくいようならば、タイマ A などを用いて表示の間隔をあける。

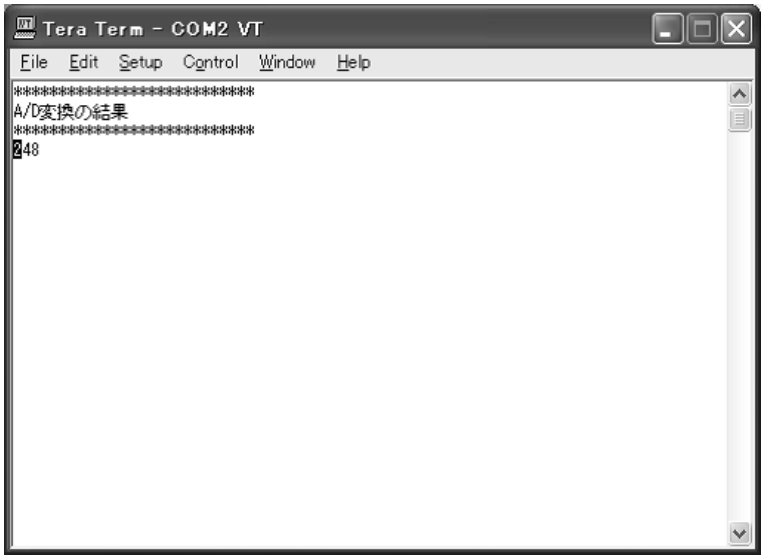

図 19.1 実行結果 (ターミナルソフトに表示)

19.2 H8/3052F への移植

株式会社秋月電子通商製の AKI-H8/3052F マイコンボードを、同じ株式会社秋月電子通商製 AKI-H8/3052LAN 開発キットに乗せて使ってみます。

AKI-H8/3052LAN 開発キットには、サーボモータを 4 個接続できるピン (5V) がありますので、4 個のサーボモータをモーション作成ソフト「意信電信」で動かすプログラムを移植してみましょう。

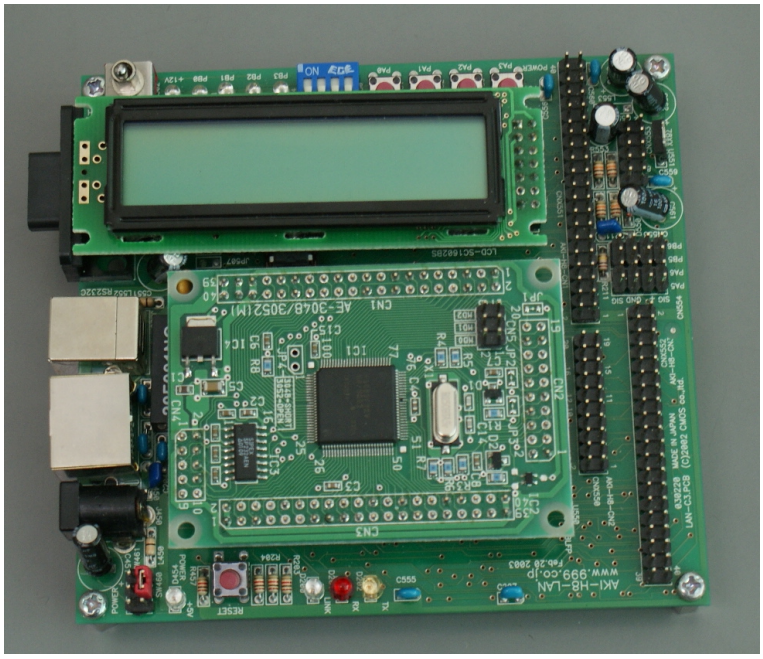

図 19.2 AKI-H8/3052F マイコンボードと AKI-H8/3052LAN 開発キット

開発環境は HEW を、書き込みには FDT を使います。

サーボモータは、専用の端子に接続してください。

- モータ 0 PA5
- モータ 1 PA6
- モータ 2 PB5
- モータ 3 PB6

intprg.c の INT_IMIA0、INT_IMIB0、INT_RXI1 をコメントアウトしてください。

intprg.c の INT_IMIA0、INT_IMIB0、INT_RXI1 をコメントアウト

```
//  vector 24 IMIA0
//__interrupt(vect=24) void INT_IMIA0(void) {/* sleep(); */}
//  vector 25 IMIB0
//__interrupt(vect=25) void INT_IMIB0(void) {/* sleep(); */}

......

//  vector 57 RXI1
//__interrupt(vect=57) void INT_RXI1(void) {/* sleep(); */}
```

05_PWM04.c

```
/************************************************************************/
/*                                                                      */
/*  FILE        :05_PWM04.c                                             */
/*  DESCRIPTION :サーボモータ 4 個「意信電信」スライダー対応版          */
/*  CPU TYPE    :H8/3052F                                               */
/*                                                                      */
/*  This file is generated by Renesas Project Generator (Ver.4.9).      */
/*                                                                      */
/************************************************************************/

#include "pwm.h"
#include "sci.h"

unsigned char HOME[SRV_USE_IDX]={100,100,100,100};
volatile int DAT_curIdx;
volatile int i;
volatile unsigned char data[20];
unsigned short id_c, dn;
/* id_c:データカウント用  */
volatile unsigned short loop, startf;
/*   loop:ループ数格納用
   startf:ループの始まりを格納  */

/*****************************
  モーション再生
*****************************/
void Play_motion(void)
{
  int loop_flag=0,loop_flag2=0;
/*  loop_flag:データの終了判定用
   loop_flag2:ループの終了判定用  */
  id_c=2;
  do{
    dn++;
    switch(data[id_c]){
      case DIV:
        id_c++;
        PwmSetPose(&data[id_c]);
        id_c=id_c+SRV_CH_NUM;
        break;
      case SLO:
        id_c++;
        loop=data[id_c];
        id_c++;
        startf=id_c;
        while(loop){
          loop_flag2=0;
          id_c=startf;
          do{
            switch(data[id_c]){
              case DIV:
                id_c++;
                PwmSetPose(&data[id_c]);
                id_c=id_c+SRV_CH_NUM;
                break;
              case ELO:
                loop_flag2=1;
                break;
            }
            id_c++;
          }while(!loop_flag2);
          loop--;
        }
        id_c--;
        break;
      case EOT:
        loop_flag=1;
        break;
    }
    id_c++;
  }while(!loop_flag);
```

```
 72  }
 73
 74  void main(void)
 75  {
 76    Sci1Init(br57600); /* ボーレートの設定 */
 77    for(i=0; i<SRV_USE_IDX; i++){
 78      PwmSetDuty(i, D_CONST+D_PROPO*HOME[i]); /* モータの初期位置設定 */
 79    }
 80    PwmInit();
 81
 82    while(1){
 83      DAT_curIdx=0;
 84      while((data[DAT_curIdx]=Sci1Read())!= SOH){ /* SOH が来るまで読み捨てる */
 85        ;
 86      }
 87      while((data[++DAT_curIdx]=Sci1Read())!=EOT){ /* EOT まで読み込む */
 88        ;
 89      }
 90      switch(data[1]){
 91        case SSD:
 92        PwmSetDuty(data[2], D_CONST+D_PROPO*data[3]); /* スライダからのデータを処理 */
 93        break;
 94        case SMD:
 95        /* モーション再生 */
 96          Play_motion();
 97          break;
 98      }
 99    }
100  }
101
```

End Of List

sci.h

```
 1  #ifndef _SCI_H_
 2  #define _SCI_H_
 3
 4  #define SCI_RX_BUFF_SIZE 32
 5  #define SCI_RX_BUFF_MASK (SCI_RX_BUFF_SIZE-1)
 6
 7  typedef enum{
 8    br19200=40,
 9    br38400=19,
10    br57600=13,
11    br115200=6
12  }BaudRate;
13
14  typedef struct{
15    unsigned short start_index;
16    unsigned short data_count;
17    char data_buff[SCI_RX_BUFF_SIZE];
18  }SciRxBuffer;
19
20  void Sci1Init(BaudRate b);
21
22  void Rx1BuffClear(void);
23
24  char Sci1Read(void);
25  int Sci1Putchar(char c);
26  int Sci1Puts(char *str);
27
28  char Sci1Getchar(void);
29  void Sci1Gets(char *str);
30
31  #endif
```

End Of List

sci.c

```
 1  #include <machine.h>
 2  #include "iodefine.h"
 3  #include "sci.h"
 4
 5  #define CHECK_LOOP 0x200000
 6  #define CR 0x0D
 7  #define LF 0x0A
 8
 9  static SciRxBuffer rx1_buff;
10
11  /*********************************
12     シリアルポートを初期化 :Sci1Init
13  *********************************/
14  void Sci1Init(BaudRate b)
15  {
16  //  STB.CR3.BIT._SCI1=0;    /* SCI1 へクロック供給 */
```

```
17    SCI1.SCR.BYTE=0x30;   /* 送受信許可、送受信割り込み禁止 */
18    SCI1.SMR.BYTE=0x00;
19     /* 調歩同期式、8 ビット、パリティなし、1 ストップビット */
20    SCI1.BRR=b;           /* ボーレートの設定 */
21
22    SYSCR.BIT.UE=0;
23  }
24
25  /*******************************
26     1 バイト送信関数 :Sci1Write
27  *******************************/
28  int Sci1Write(char c)
29  {
30    unsigned long i;
31    unsigned short write_flag=0;    /* 送信データ書き込みフラグ */
32
33    for(i=0; i<CHECK_LOOP; i++){
34      if(SCI1.SSR.BIT.TDRE){ /* 送信データ書き込み可能になるまで待つ */
35        write_flag=1;
36        break;
37      }
38    }
39    if(write_flag){
40      SCI1.TDR=c;           /* 送信データを格納 */
41      SCI1.SSR.BIT.TDRE=0; /* 送信データ書き込み可能レジスタクリア */
42      return(0);
43    }
44    return(-1);
45  }
46
47  /*******************************
48     1 バイト受信関数 (割り込み有り)
49  *******************************/
50  __interrupt(vect=57) void INT_RXI1(void)
51  {
52    char msg;
53    int index;
54
55    if(SCI1.SSR.BIT.RDRF){
56      SCI1.SSR.BIT.RDRF=0;
57      msg=SCI1.RDR;
58      if(rx1_buff.data_count < SCI_RX_BUFF_SIZE){
59        index=(rx1_buff.start_index+rx1_buff.data_count) & SCI_RX_BUFF_MASK;
60        rx1_buff.data_buff[index]=msg;
61        rx1_buff.data_count++;
62      }
63    }
64  }
65
66  int Sci1Rx1(char *msg)
67  {
68    if(rx1_buff.data_count > 0){
69      *msg=rx1_buff.data_buff[rx1_buff.start_index];
70      rx1_buff.start_index=(rx1_buff.start_index+1) & SCI_RX_BUFF_MASK;
71      rx1_buff.data_count--;
72      return(0);
73    }
74    *msg=0;
75    return(-1);
76  }
77
78  /************************************************
79  *     1 文字受信 : Sci1Read
80  ************************************************/
81  char Sci1Read(void)
82  {
83    char c;
84    while(Sci1Rx1(&c)){
85      ;
86    }
87    return(c);
88  }
89
90  /***************************************
91     受信バッファクリア関数 :Rx1BuffClear
92  ***************************************/
93  void Rx1BuffClear(void)
94  {
95    rx1_buff.start_index=0;
96    rx1_buff.data_count=0;
97  }
98
99  /*+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
100    汎用関数
101 +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*/
102
103 /*********************************************************
104    1 バイト送信関数 (改行 (\n) は CR+LF に変換) :Sci1Putchar
105 *********************************************************/
106
107 int Sci1Putchar(char c)
108 {
109   /* 改行 (\n) は CR+LF に変換して送信 */
```

```
110   if(c=='\n'){
111     if(Sci1Write(CR)){
112       return(-1);
113     }
114     if(Sci1Write(LF)){
115       return(-1);
116     }
117   }else{  /* それ以外はそのまま送信 */
118     if(Sci1Write(c)){
119       return(-1);
120     }
121   }
122   return(0);
123 }
124
125 /*******************************************************
126 *    文字列送信 (改行 (\n) は CR+LF に変換) :Sci1Puts
127 *******************************************************/
128 int Sci1Puts(char *str)
129 {
130   char *msg;
131
132   /* 終端文字まで 1 文字ずつ表示 */
133   for(msg=str; *msg!='\0'; msg++){
134     if(Sci1Putchar(*msg)){
135       return(0);
136     }
137   }
138   return(msg-str);  /* 表示文字数を返す */
139 }
140
141
142 /***********************************************
143 *    1 文字受信 (CR は改行 (\n) に変換) : Sci1Getchar
144 ***********************************************/
145 char Sci1Getchar(void)
146 {
147   char c;
148   c=Sci1Read();
149   if(c==CR){
150     c='\n';
151   }
152   return(c);
153 }
154
155 /***********************************************
156 *    文字列受信 (CR は改行 (\n) に変換) : Sci1Gets
157 ***********************************************/
158 void Sci1Gets(char *str)
159 {
160   int i=0;
161   char s;
162
163   while((s=Sci1Getchar()) != '\n'){
164     str[i]=s;
165     i++;
166   }
167   str[i]='\0';
168 }
```

End Of List

pwm.h

```
1  #ifndef _PWM_H_
2  #define _PWM_H_
3
4  #define PWM_MAX_DUTY  7500-1 //2.4mS
5  #define PWM_MIN_DUTY  2500-1 //0.4mS
6  #define PWM_PERIOD   15625-1 //5.0mS
7
8  #define SRV_CH_NUM 4
9  #define SRV_USE_IDX 4
10
11 #define D_CONST PWM_MIN_DUTY //char_duty と duty の変換式の定数部分
12 #define D_PROPO 25   //char_duty と duty の変換式の比例係数部分 (7500-2500)/200
13
14 /* モータの方向レジスタ */
15 #define MOT_DDR1 PA.DDR
16 #define MOT_DDR2 PB.DDR
17
18 /* モータのデータレジスタ */
19 #define MOT0_DR PA.DR.BIT.B5
20 #define MOT1_DR PA.DR.BIT.B6
21 #define MOT2_DR PB.DR.BIT.B5
22 #define MOT3_DR PB.DR.BIT.B6
23
24 /* タイマカウントフラグ */
25 #define ON_COUNT ITU0.GRA
26 #define PER_COUNT ITU0.GRB
```

```
27
28  #define SDV 239
29  #define SMD 241
30  #define DIV 242
31  #define SLO 243
32  #define ELO 244
33  #define SSD 249
34  #define SOH 254
35  #define EOT 255
36
37  /* 割込みフラグ */
38  #define ON_FLG ITU0.TSR.BIT.IMFA
39  #define PER_FLG ITU0.TSR.BIT.IMFB
40
41  /* タイマの動作開始・停止フラグ */
42  #define STR_FLG ITU.TSTR.BIT.STR0
43
44  void PwmInit(void);
45  void PwmStart(void);
46  void PwmStop(void);
47  void PwmSetDuty(unsigned short ch, unsigned short duty);
48  unsigned short PwmGetDuty(unsigned short ch);
49  void PwmSetPose(unsigned char char_duty[]);
50
51  #endif
```

End Of List

pwm.c

```
 1  /*********************************************************************/
 2  /*                                                                   */
 3  /*  FILE        :pwm.c                                               */
 4  /*  DESCRIPTION :サーボモータ関連関数                                */
 5  /*  CPU TYPE    :h8/3052F                                            */
 6  /*                                                                   */
 7  /*  ソフト的に割り振り                                               */
 8  /*  モータ 0 PA5                                                     */
 9  /*  モータ 1 PA6                                                     */
10  /*  モータ 2 PB5                                                     */
11  /*  モータ 3 PB6                                                     */
12  /*********************************************************************/
13
14  #include "iodefine.h"
15  #include "pwm.h"
16
17  volatile unsigned short srv_index; /* PWM の出力先 */
18  volatile unsigned long pwm_duty[SRV_CH_NUM]; /* PWM のデューティのデータ */
19  volatile unsigned int calc_duty_flg; /* 20mS ごとにセット、計算を行う */
20
21  /*******************************
22     モータ制御端子を初期化
23  *******************************/
24  void MotInit(void)
25  {
26     MOT_DDR1 |= 0x60;
27     MOT_DDR2 |= 0x6f;
28  }
29
30  /*******************************
31     PWM 出力端子を初期化
32  *******************************/
33  void PwmInit(void)
34  {
35    srv_index=0;
36    calc_duty_flg=0;
37
38    ITU0.TCR.BYTE=0x43;
39    ON_COUNT=PWM_PERIOD-pwm_duty[srv_index];
40    ITU0.GRB=PWM_PERIOD;
41    ITU0.TIER.BIT.IMIEA=1;
42    ITU0.TIER.BIT.IMIEB=1;
43    SYSCR.BIT.UE=0;
44    STR_FLG=1;  /* TCNT カウンタスタート */
45
46    MotInit();
47  }
48
49  /*********************************
50    割り込み関数
51   *********************************/
52  __interrupt(vect=24) void INT_IMIA0(void)
53  {
54    switch(srv_index){
55      case 0:
56         MOT0_DR=1;
57         break;
58      case 1:
59         MOT1_DR=1;
60         break;
```

```
61      case 2:
62         MOT2_DR=1;
63         break;
64      case 3:
65         MOT3_DR=1;
66         break;
67    }
68    ON_FLG = 0;
69  }
70  __interrupt(vect=25) void INT_IMIB0(void)
71  {
72    switch(srv_index){
73      case 0:
74         MOT0_DR=0;
75         break;
76      case 1:
77         MOT1_DR=0;
78         break;
79      case 2:
80         MOT2_DR=0;
81         break;
82      case 3:
83         MOT3_DR=0;
84         break;
85    }
86    srv_index++;
87    if(srv_index>= SRV_USE_IDX){
88      srv_index=0; /* モータ出力先を 0 に戻す */
89      calc_duty_flg=1;
90    }
91    ON_COUNT=PWM_PERIOD-pwm_duty[srv_index];
92    PER_FLG = 0;
93  }
94
95  /*********************************
96    サーボ動作開始
97   *********************************/
98  void PwmStart(void)
99  {
100   STR_FLG=1;
101 }
102
103 /*********************************
104   サーボ動作停止
105  *********************************/
106 void PwmStop(void)
107 {
108   STR_FLG=0;
109 }
110
111 /*********************************
112   デューティの設定
113  *********************************/
114 void PwmSetDuty(unsigned short ch, unsigned short duty)
115 {
116   /* サーボチャンネルが範囲外なら無視 */
117   if(ch >= SRV_USE_IDX){
118     return ;
119   }
120
121   if(duty < PWM_MIN_DUTY){          /* デューティを最小値以上に限定 */
122     duty = PWM_MIN_DUTY;
123   }else if(duty > PWM_MAX_DUTY){    /* デューティを最大値以下に限定 */
124     duty = PWM_MAX_DUTY;
125   }
126   /* デューティを設定 */
127   pwm_duty[ch] = duty;
128 }
129
130 /*********************************
131   デューティの取得（パラメータから）
132  *********************************/
133 unsigned short PwmGetDuty(unsigned short ch)
134 {
135   /* サーボチャンネルが範囲外なら無視 */
136   if(ch >= SRV_USE_IDX){
137     return (0);
138   }
139
140   /* デューティを返す */
141   return (pwm_duty[ch]);
142 }
143
144 /*-----------------------------------------------------------------------*/
145 /* ポーズ補間関数 0-200 で指定*/
146 /*-----------------------------------------------------------------------*/
147 void PwmSetPose(unsigned char char_duty[])
148 {
149   long cur_duty[SRV_CH_NUM], new_duty[SRV_CH_NUM];
150   long diff_duty[SRV_CH_NUM];
151   int i;
152   unsigned short div, count_i;
153
```

```
154    div=char_duty[0]; /* 分割数：20mS 単位 */
155    for(i=0; i<SRV_USE_IDX; i++){
156      new_duty[i]=D_CONST+D_PROPO*char_duty[i+1];/* 目標値 */
157      cur_duty[i]=PwmGetDuty(i); /* 現在のデューティ */
158      /* 現在のデューティと目標デューティの差 */
159      if(new_duty[i]-cur_duty[i]>=0){
160        diff_duty[i]=new_duty[i]-cur_duty[i];
161      }else{
162        diff_duty[i]=cur_duty[i]-new_duty[i];
163      }
164    }
165    /* 補間の計算 */
166    for(count_i=1; count_i<=div; count_i++){
167      /* 20mS 待ち */
168      while(!calc_duty_flg){
169        ;
170      }
171      calc_duty_flg=0;
172      /* 20mS ごとの値を計算 */
173      for(i=0; i<SRV_USE_IDX; i++){
174        if(new_duty[i]-cur_duty[i]>=0){
175          PwmSetDuty(i, cur_duty[i]+(diff_duty[i]*count_i)/div);
176        }else{
177          PwmSetDuty(i, cur_duty[i]-(diff_duty[i]*count_i)/div);
178        }
179      }
180    }
181  }
```

End Of List

実行結果

- モータ 0 PA5
- モータ 1 PA6
- モータ 2 PB5
- モータ 3 PB6

モーション作成ソフト「意信電信」でサーボモータを動かすことができる。

19.3 H8/3052F(HOS) への移植

AKI-H8/3052F マイコンボード上では、μITRON 仕様 4.03 準拠のリアルタイム OS、Hyper Operating System(HOS) を動かすことができます。

開発環境も HEW を利用することができるので、ここではこの OS に移植することを考えましょう。

HOS では独自のレジスタ定義ファイル h8_3048.h を用意しています。これは HEW が標準で出力するレジスタ定義ファイル iodefine.h とは内容が異なります。レジスタ定義ファイル h8_3048.h は OS が利用しているため、変更するには OS のソースコードを書き換えなくてはなりません。

また、複数のレジスタ定義ファイルが同一のレジスタを違う名前で宣言するとコンパイルが通らないようです。

そこで、HOS で用意されているレジスタ定義ファイル h8_3048.h にあわせて書き換える必要があるのです。

また、HOS ではシリアル通信の関数が用意されていますので、これも用いることにします。ただし、1 文字送信関数と 1 文字受信関数だけですので、その他の関数をこれを用いて実装しましょう。

プログラムは、HOS のサンプルプログラムを変更して作りましょう。

以下は移動体追尾システムに利用したモータ制御プログラムです。

移動体追尾システムとは、USB カメラの画像を処理することにより、移動体を検知し、二つのサーボモータを制御することによって、先端に取り付けた USB カメラを移動体の方向に向けるシステムです (図 19.3)

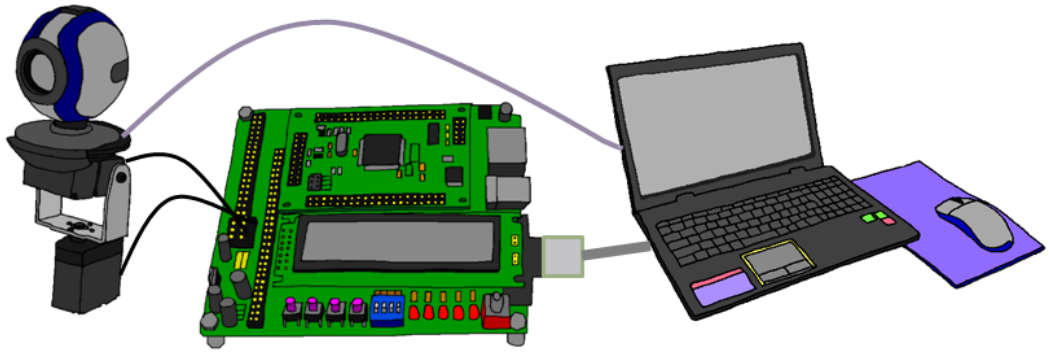

図 19.3 移動体追尾システム

P.436 の「H8/3052F への移植」同様に、以下のようにサーボモータをつなぎ、ターミナルソフトから文字を送ることで、サーボモータを動かすことができます。

- モータ 0 PA5
- モータ 1 PA6

なお、ここではサーボモータを二つしか使っていませんが、割込み関数は 4 個のサーボモータを制御できるように記述しています。

19.3.1 イベントフラグを用いたプログラムの解説

リアルタイム OS を用いると、マルチタスクプログラムが効率よく開発できます。

ここでは、シリアル通信を行うタスク (シリアルタスク) とモータを制御するタスク (モータタスク) を生成し、イベントフラグを用いて通信することで、ターミナルソフトからの命令でサーボモータを制御します (図 19.4)。

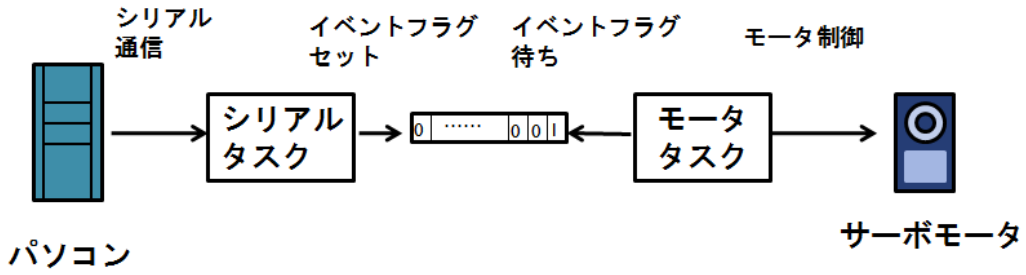

図 19.4 イベントフラグを用いたタスク間通信

以下のサンプルプログラムでは、ターミナルソフトから 1 文字を送信して、サーボモータを動かすことができます。

- 「l」: モータ 0 PA5 反時計回り
- 「r」: モータ 0 PA5 時計回り
- 「u」: モータ 1 PA6 反時計回り
- 「d」: モータ 1 PA6 時計回り

system.cfg

```
1  /* ------------------------------------------------------------------------ */
2  /*  Hyper Operating System V4  サンプルプログラム                           */
3  /*    コンフィギュレーションファイル                                        */
4  /*                                                                          */
5  /*                                Copyright (C) 1998-2002 by Project HOS  */
6  /*                                http://sourceforge.jp/projects/hos/     */
7  /* ------------------------------------------------------------------------ */
8
9
10
11 INCLUDE("\"sample.h\"");
12 INCLUDE("\"ostimer.h\"");
13 INCLUDE("\"h8_sci.h\"");
14
15
16 /* HOS 独自の設定 */
17 HOS_KERNEL_HEAP(0);      /* カーネルヒープの設定 (省略時 0) */
18 HOS_TIM_TIC(1, 1);       /* タイムティックの設定 (省略時 1/1 ) */
19 HOS_MAX_TPRI(8);       /* 最大優先度 (省略時 16) */
20 HOS_MIN_INTNO(0);      /* 割り込み番号の最小値 (省略時 0) */
21 HOS_MAX_INTNO(256);      /* 割り込み番号の最大値 (省略時 0) */
22 HOS_MAX_TSKID(8);      /* 最大タスク ID 番号 (省略時静的生成に必要なだけ) */
23
24
```

```
25  /* OS タイマ */
26  ATT_INI({TA_HLNG, 0, OsTimer_Initialize});
27  ATT_ISR({TA_HLNG, 0, 24, INT_IMIA0});
28  ATT_ISR({TA_HLNG, 0, 25, INT_IMIB0});
29
30  /* SCI */
31  ATT_ISR({TA_HLNG, 0, 57, Sci_RxiHandler});
32
33  /* サンプルプログラム */
34  ATT_INI({TA_HLNG, 0, Initialize});
35  CRE_TSK(TSKID_SAMPLE1, {TA_HLNG | TA_ACT, 1, Task_SCI, 2, 256, NULL});
36  CRE_TSK(TSKID_SAMPLE2, {TA_HLNG | TA_ACT, 0, Task_MOT2, 1, 256, NULL});
37
38  CRE_FLG(FLG_ID, {TA_TFIFO | TA_WSGL | TA_CLR, 0});
39
40  /* ------------------------------------------------------------------------ */
41  /*  Copyright (C) 1998-2002 by Project HOS                                  */
42  /* ------------------------------------------------------------------------ */
```

End Of List

sample.h

```
 1  /* ------------------------------------------------------------------------ */
 2  /*  Hyper Operating System V4  サンプルプログラム                           */
 3  /*                                                                          */
 4  /*                                  Copyright (C) 1998-2002 by Project HOS  */
 5  /*                                  http://sourceforge.jp/projects/hos/     */
 6  /* ------------------------------------------------------------------------ */
 7
 8
 9  #ifndef __PROJECT_HOS__sample_h__
10  #define __PROJECT_HOS__sample_h__
11
12
13
14  void Initialize(VP_INT exinf);
15  void Task_MOT(VP_INT exinf);
16  void Task_MOT2(VP_INT exinf);
17  void Task_SCI(VP_INT exinf);
18  void INT_IMIA0(void);
19  void INT_IMIB0(void);
20
21  #endif /* __PROJECT_HOS__sample_h__ */
22
23
24
25  /* ------------------------------------------------------------------------ */
26  /*  Copyright (C) 1998-2002 by Project HOS                                  */
27  /* ------------------------------------------------------------------------ */
```

End Of List

sample.c

```
 1  /* ------------------------------------------------------------------------ */
 2  /*  Hyper Operating System V4  サンプルプログラム                           */
 3  /*                                                                          */
 4  /*                                  Copyright (C) 1998-2002 by Project HOS  */
 5  /*                                  http://sourceforge.jp/projects/hos/     */
 6  /* ------------------------------------------------------------------------ */
 7
 8  #include "kernel.h"
 9  #include "kernel_id.h"
10  #include "h8_3048.h"
11  #include "pwm.h"
12  #include "h8_sci.h"
13  #include "sci.h"
14
15  #define MOTION_NUM 4 /* モーション数 */
16  #define CENTER 5000
17  #define DIFF 80
18
19
20  /* メイン関数 */
21  int main()
22  {
23    /* SCI の初期化 */
24    Sci_Initialize(SCI_19200);
25
26    /*
27    ASTCR = 0xff;
28    WCR   = 0xf0;
29    */
30
31    /* 開始メッセージ */
```

```
32    Sci_PutChar('H');
33    Sci_PutChar('O');
34    Sci_PutChar('S');
35    Sci_PutChar('\r');
36    Sci_PutChar('\n');
37
38    sta_hos();
39
40    return 0;
41  }
42
43  /* 初期化ハンドラ */
44  void Initialize(VP_INT exinf)
45  {
46    int i;
47      PwmInit(); /* PWM の初期化 */
48    /* act_tsk(TSKID_SAMPLE1); */
49
50    /* モータを初期位置にセット */
51      for(i=0; i<SRV_USE_IDX; i++){
52          PwmSetDuty(i, CENTER);
53      }
54  }
55
56  /* モータ制御タスク */
57  void Task_MOT2(VP_INT exinf)
58  {
59    int i, dutyx=CENTER, dutyy=CENTER;
60    FLGPTN getflg;
61
62      while(1){
63      wai_flg(FLG_ID, 0x8000, TWF_ANDW, &getflg);
64      switch(getflg&0x000f){
65        case 1:
66        dutyx=dutyx-DIFF;
67        if(dutyx<PWM_MIN_DUTY){
68          dutyx=PWM_MIN_DUTY;
69        }
70        PwmSetDuty(0, dutyx);
71        break;
72        case 2:
73        dutyx=dutyx+DIFF;
74        if(dutyx>PWM_MAX_DUTY){
75          dutyx=PWM_MAX_DUTY;
76        }
77        PwmSetDuty(0, dutyx);
78        break;
79        case 4:
80        dutyy=dutyy+DIFF;
81        if(dutyy<PWM_MIN_DUTY){
82          dutyy=PWM_MIN_DUTY;
83        }
84        PwmSetDuty(1, dutyy);
85        break;
86        case 8:
87        dutyy=dutyy-DIFF;
88        if(dutyy>PWM_MAX_DUTY){
89          dutyy=PWM_MAX_DUTY;
90        }
91        PwmSetDuty(1, dutyy);
92        break;
93      }
94      }
95  }
96
97  /* シリアル受信タスク (イベントフラグに設定) */
98  void Task_SCI(VP_INT exinf)
99  {
100   char getSci;
101
102   while(1){
103     getSci=Sci1Getchar();
104     switch(getSci){
105       case 'l':
106         set_flg(FLG_ID, 0x8001);
107         break;
108       case 'r':
109         set_flg(FLG_ID, 0x8002);
110         break;
111       case 'd':
112         set_flg(FLG_ID, 0x8004);
113         break;
114       case 'u':
115         set_flg(FLG_ID, 0x8008);
116         break;
117
118     }
119   }
120 }
121
122 /*********************************
123   割り込み関数
124  *********************************/
```

```
125  void INT_IMIA0(void)
126  {
127    switch(srv_index){
128      case 0:
129         PADR |= (1 << 5);
130         break;
131      case 1:
132         PADR |= (1 << 6);
133         break;
134      case 2:
135         PBDR |= (1 << 5);
136         break;
137      case 3:
138         PBDR |= (1 << 6);
139         break;
140    }
141    TSR0 &= ~(0x01);
142  }
143
144  void INT_IMIB0(void)
145  {
146    switch(srv_index){
147      case 0:
148         PADR &= ~(1 << 5);
149         break;
150      case 1:
151         PADR &= ~(1 << 6);
152         break;
153      case 2:
154         PBDR &= ~(1 << 5);
155         break;
156      case 3:
157         PBDR &= ~(1 << 6);
158         break;
159    }
160        srv_index++;
161        if(srv_index>= SRV_USE_IDX){
162          srv_index=0; /* モータ出力先を 0 に戻す */
163      calc_duty_flg=1;
164        }
165        GRA0=PWM_PERIOD-pwm_duty[srv_index];
166      TSR0 &= ~(0x02);
167
168  }
169
170  /* ------------------------------------------------------------------------ */
171  /*  Copyright (C) 1998-2002 by Project HOS                                  */
172  /* ------------------------------------------------------------------------ */
```

End Of List

pwm.h

```
 1  #ifndef _PWM_H_
 2  #define _PWM_H_
 3
 4  #define PWM_MAX_DUTY  7500-1 //2.4mS
 5  #define PWM_MIN_DUTY  2500-1 //0.4mS
 6  #define PWM_PERIOD   15625-1 //5.0mS
 7
 8  #define SRV_CH_NUM 4
 9  #define SRV_USE_IDX 4
10
11  #define D_CONST PWM_MIN_DUTY //char_duty と duty の変換式の定数部分
12  #define D_PROPO 25   //char_duty と duty の変換式の比例係数部分 (7500-2500)/200
13
14  extern unsigned short srv_index; /* PWM の出力先 */
15  extern unsigned short pwm_duty[SRV_CH_NUM]; /* PWM のデューティのデータ */
16  extern unsigned int calc_duty_flg; /* 20mS ごとにセット、計算を行う */
17
18  void PwmInit(void);
19  void PwmStart(void);
20  void PwmStop(void);
21  void PwmSetDuty(unsigned short ch, unsigned short duty);
22  unsigned short PwmGetDuty(unsigned short ch);
23  void PwmSetPose(unsigned char char_duty[]);
24
25  #endif
```

End Of List

pwm.c

```
1  /*************************************************************************/
2  /*                                                                       */
3  /*  FILE        :pwm.c                                                   */
```

```
 4  /*  DATE        :Fri, Nov 02, 2012                                    */
 5  /*  DESCRIPTION :サーボモータ関連関数                                 */
 6  /*  CPU TYPE    :h8/3052F                                             */
 7  /*                                                                    */
 8  /*  ソフト的に割り振り                                                */
 9  /*  モータ 0 PA5                                                      */
10  /*  モータ 1 PA6                                                      */
11  /*  モータ 2 PB5                                                      */
12  /*  モータ 3 PB6                                                      */
13  /*********************************************************************/
14
15  #include "h8_3048.h"
16  #include "pwm.h"
17  #include "kernel.h"
18
19  unsigned short srv_index; /* PWM の出力先 */
20  unsigned short pwm_duty[SRV_CH_NUM]; /* PWM のデューティのデータ */
21  unsigned int calc_duty_flg; /* 20mS ごとにセット、計算を行う */
22
23  /*******************************
24     モータ制御端子を初期化
25  *******************************/
26  void MotInit(void)
27  {
28     PADDR |= 0x60;
29     PBDDR |= 0x60;
30  }
31
32  /*******************************
33     PWM 出力端子を初期化
34  *******************************/
35  void PwmInit(void)
36  {
37    srv_index=0;
38    calc_duty_flg=0;
39
40     TCR0=0x43;
41     GRA0=PWM_PERIOD-pwm_duty[srv_index];
42     GRB0=PWM_PERIOD;
43     TIER0|=3; /* IMFA, IMFB 割込み許可 */
44     SYSCR&= ~(1<<3);
45     PwmStart();  /* TCNT カウンタスタート */
46
47    MotInit();
48  }
49
50  /*********************************
51    サーボ動作開始
52   *********************************/
53  void PwmStart(void)
54  {
55     TSTR|=0x01;  /* TCNT カウンタスタート */
56  }
57
58  /*********************************
59    サーボ動作停止
60   *********************************/
61  void PwmStop(void)
62  {
63     TSTR&= ~0x01;  /* TCNT カウンタストップ */
64  }
65
66  /*********************************
67    デューティの設定
68   *********************************/
69  void PwmSetDuty(unsigned short ch, unsigned short duty)
70  {
71    /* サーボチャンネルが範囲外なら無視 */
72    if(ch >= SRV_USE_IDX){
73      return ;
74    }
75
76    if(duty < PWM_MIN_DUTY){          /* デューティを最小値以上に限定 */
77      duty = PWM_MIN_DUTY;
78    }else if(duty > PWM_MAX_DUTY){    /* デューティを最大値以下に限定 */
79      duty = PWM_MAX_DUTY;
80    }
81    /* デューティを設定 */
82    pwm_duty[ch] = duty;
83  }
84
85  /*********************************
86    デューティの取得（パラメータから）
87   *********************************/
88  unsigned short PwmGetDuty(unsigned short ch)
89  {
90    /* サーボチャンネルが範囲外なら無視 */
91    if(ch >= SRV_USE_IDX){
92      return (0);
93    }
94
```

```
 95    /* デューティを返す */
 96    return (pwm_duty[ch]);
 97  }
 98
 99  /*-----------------------------------------------------------------------*/
100  /* ポーズ補間関数 0-200 で指定*/
101  /*-----------------------------------------------------------------------*/
102  void PwmSetPose(unsigned char char_duty[])
103  {
104    long cur_duty[SRV_CH_NUM], new_duty[SRV_CH_NUM];
105    long diff_duty[SRV_CH_NUM];
106    int i;
107    unsigned short div, count_i;
108
109    div=char_duty[0]; /* 分割数：20mS 単位 */
110    for(i=0; i<SRV_USE_IDX; i++){
111      new_duty[i]=D_CONST+D_PROPO*char_duty[i+1];/* 目標値 */
112      cur_duty[i]=PwmGetDuty(i); /* 現在のデューティ */
113  /* 現在のデューティと目標デューティの差 */
114  if(new_duty[i]-cur_duty[i]>=0){
115        diff_duty[i]=new_duty[i]-cur_duty[i];
116    }else{
117      diff_duty[i]=cur_duty[i]-new_duty[i];
118    }
119    }
120    /* 補間の計算 */
121    for(count_i=1; count_i<=div; count_i++){
122      /* 20mS 待ち */
123      while(!calc_duty_flg){
124      ;
125    }
126    calc_duty_flg=0;
127    /* 20mS ごとの値を計算 */
128      for(i=0; i<SRV_USE_IDX; i++){
129      if(new_duty[i]-cur_duty[i]>=0){
130          PwmSetDuty(i, cur_duty[i]+(diff_duty[i]*count_i)/div);
131      }else{
132          PwmSetDuty(i, cur_duty[i]-(diff_duty[i]*count_i)/div);
133      }
134      }
135    }
136  }
```

End Of List

sci.h

```
 1  #ifndef _SCI_H_
 2  #define _SCI_H_
 3
 4  int Sci1Putchar(char c);
 5  int Sci1Puts(char *str);
 6  char Sci1Getchar(void);
 7  void Sci1Gets(char *str);
 8
 9  #endif  /* _SCI_H_ */
```

End Of List

sci.c

```
 1  #include "kernel.h"
 2  #include "kernel_id.h"
 3  #include "h8_3048.h"
 4  #include "h8_sci.h"
 5  #include "sci.h"
 6
 7  #define CR 0x0D
 8  #define LF 0x0A
 9
10  /*1 文字送信*/
11  int Sci1Putchar(char c)
12  {
13    if(c=='\n'){
14      Sci_PutChar(CR);
15      Sci_PutChar(LF);
16    }else{
17      Sci_PutChar(c);
18    }
19    return(0);
20
21  }
22
23  /*文字列送信*/
```

```
24  int Sci1Puts(char *str)
25  {
26    char *msg;
27    for(msg=str; *msg!='\0'; msg++){
28      if(Sci1Putchar(*msg)){
29        return(0);
30      }
31    }
32    return(msg-str);
33  }
34
35  /*1 文字受信*/
36  char  Sci1Getchar(void)
37  {
38    char c;
39    while((c=Sci_GetChar())<0){
40      ;
41    }
42    if(c==CR){
43      c='\n';
44    }
45    return(c);
46  }
47
48  /*文字列受信*/
49  void Sci1Gets(char *str)
50  {
51    int i=0;
52    char s;
53
54    while((s=Sci1Getchar())!='\n'){
55      str[i]=s;
56      i++;
57    }
58    str[i]='\0';
59  }
```

End Of List

実行結果

- モータ 0 PA5
- モータ 1 PA6

ターミナルソフトから 1 文字を送信して、サーボモータを動かすことができる。

- 「l」: モータ 0 PA5 反時計回り
- 「r」: モータ 0 PA5　時計回り
- 「u」: モータ 1 PA6 反時計回り
- 「d」: モータ 1 PA6　時計回り