19.4 SH7125F への移植

SH7125F はルネサスエレクトロニクス株式会社製の SH2 タイニーマイコンです。5V で駆動します。

このマイコンも HEW を用いて開発ができます。

回路図

以下の回路は、サーボモータを 24 個動かすための小型ロボット基板のための回路図です。

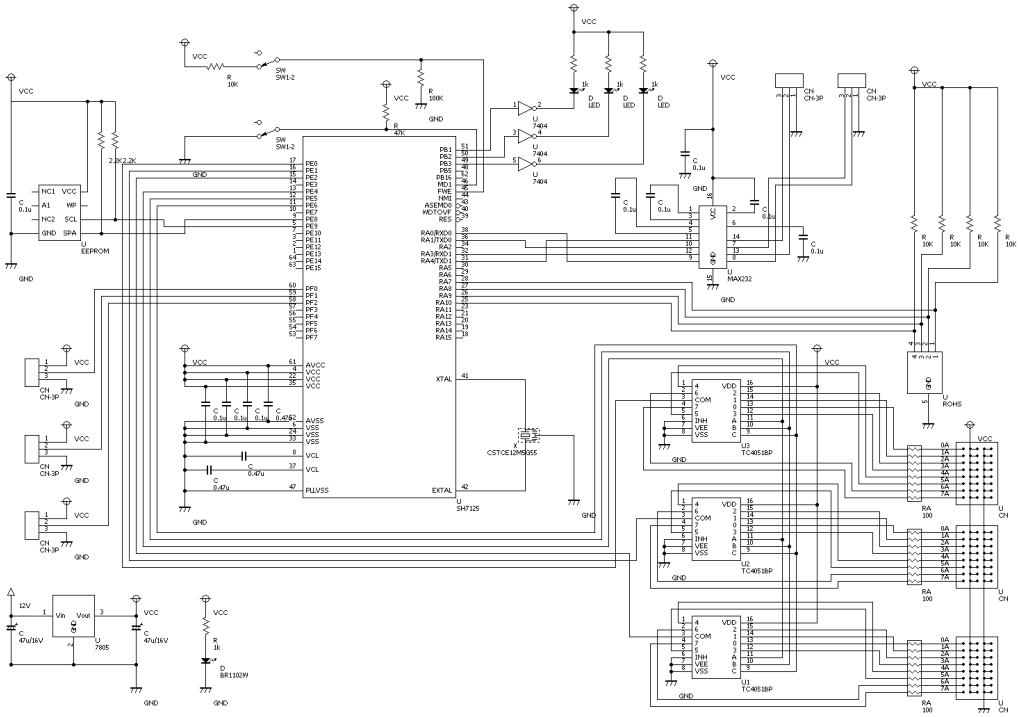

図 19.5 SH7125F 回路図

サーボモータの回路の一部を拡大しておきます。

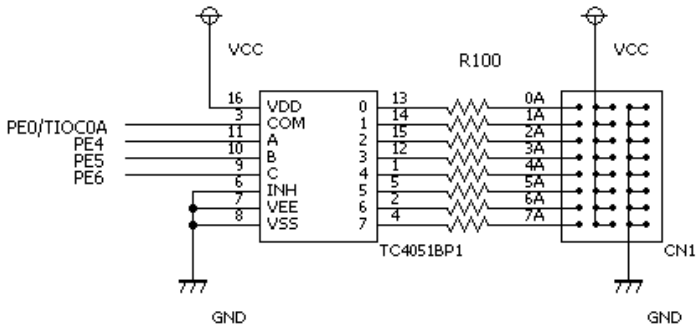

図 19.6 モータ回路図

サンプルプログラム

以下のサンプルプログラムは、図 19.6 以外に、ITOC0B、ITOC0C についても同様の回路を作り、24 個のロボット用モータを制御できるようにしたものです。

シリアルからサーボモータを動かすプログラム操作仕様

P.365 のサンプルプログラムと同じ仕様です。

- 起動時は、すべてのモータが中心位置 (ON 時間 1.5ms)。操作対象モータは 0。
- 操作対象のモータ番号とそのパルスの ON 時間を表示 (図 19.7 参照)。
- + を入力するとパルスの ON 時間が 10us 増加する。
- - を入力するとパルスの ON 時間が 10us 減少する。
- u を入力するとパルスの ON 時間が 100us 増加する。
- d を入力するとパルスの ON 時間が 100us 減少する。
- > を入力すると操作対象のモータ番号が 1 増える。
- < を入力すると操作対象のモータ番号が 1 減る。
- s を入力するとパルス出力スタート (起動時はパルスは出力設定)。
- e を入力するとパルス出力ストップ。

intprg.c の INT_MTU2_0_TGID0、INT_SCI1_RXI1、INT_SCI1_TEI1 をコメントアウトしてください。

intprg.c の INT_MTU2_0_TGID0、INT_SCI1_RXI1、INT_SCI1_TEI1 をコメントアウト

```
// 90 MTU2_0 TGIC0
void INT_MTU2_0_TGIC0(void){/* sleep(); */}
// 91 MTU2_0 TGID0
//void INT_MTU2_0_TGID0(void){/* sleep(); */}

.....

// 221 SCI1 RXI1
//void INT_SCI1_RXI1(void){/* sleep(); */}
// 222 SCI1 TXI1
void INT_SCI1_TXI1(void){/* sleep(); */}
// 223 SCI1 TEI1
//void INT_SCI1_TEI1(void){/* sleep(); */}
```

pwm02.c

```
/************************************************************************/
/*                                                                      */
/*  FILE        :pwm02.c                                                */
/*  DESCRIPTION :サーボモータををシリアルから駆動                       */
/*  CPU TYPE    :SH7125                                                 */
/*                                                                      */
/*  PWM1 PE0                                                            */
/*  PWM2 PE1                                                            */
/*  PWM3 PE2                                                            */
/*                                                                      */
/*  コントロール端子 A PE4                                              */
/*  コントロール端子 B PE5                                              */
/*  コントロール端子 C PE6                                              */
/*                                                                      */
/************************************************************************/

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

#define COEFF 24

char buf[6];
unsigned long n;
int c;
volatile unsigned short srv_curr_index=0;

void main(void)
{
  int a;
  Sci1Init(br38400);
  PwmInit();
  for(a=0; a<SRV_USE_IDX; a++){
    PwmSetDuty(a, (PWM_MAX_DUTY+PWM_MIN_DUTY)>>1);
  }
  PwmStart();
  while(1){
    Sci1Puts("*************************\n");
    Sci1Puts("パルス増減\n");
    StrLongToDec(srv_curr_index, buf);
    Sci1Puts(buf);
    Sci1Puts("のパルス ON 時間:約");
    n=PwmGetDuty(srv_curr_index);
    n=(n+1)/COEFF;
    StrLongToDec(n, buf);
    Sci1Puts(buf);
    Sci1Puts("uS\n");
    Sci1Puts("---------------------------\n");
    Sci1Puts("+:  +10uS\n");
    Sci1Puts("-:  -10uS\n");
    Sci1Puts("u: +100uS\n");
    Sci1Puts("d: -100uS\n");
    Sci1Puts(">:次のモータ\n");
    Sci1Puts("<:前のモータ\n");
    Sci1Puts("s:パルススタート\n");
    Sci1Puts("e:パルスストップ\n");

    c=Sci1Getchar();
    switch(c){
    case '+':
      PwmSetDuty(srv_curr_index, PwmGetDuty(srv_curr_index)+10*COEFF);
      break;
    case '-':
      PwmSetDuty(srv_curr_index, PwmGetDuty(srv_curr_index)-10*COEFF);
      break;
    case 'u':
```

```
67        PwmSetDuty(srv_curr_index, PwmGetDuty(srv_curr_index)+100*COEFF);
68        break;
69      case 'd':
70        PwmSetDuty(srv_curr_index, PwmGetDuty(srv_curr_index)-100*COEFF);
71        break;
72      case '>':
73        if(srv_curr_index < SRV_USE_IDX-1){
74          srv_curr_index++;
75        }
76        break;
77      case '<':
78        if(srv_curr_index > 0){
79          srv_curr_index--;
80        }
81        break;
82      case 's':
83        PwmStart();
84        break;
85      case 'e':
86        PwmStop();
87        break;
88      }
89    }
90  }
```

End Of List

pwm.h

```
 1  #ifndef _PWM_H_
 2  #define _PWM_H_
 3
 4  #define PWM_MAX_DUTY 55200-1 //2.3mS (2300*24)-1
 5  #define PWM_MIN_DUTY 16800-1 //0.7mS
 6  #define PWM_PERIOD   60000-1 //2.5mS
 7
 8  #define SRV_IDX_NUM 8
 9  #define SRV_IDX_MASK (SRV_IDX_NUM-1)
10  #define SRV_BANK_NUM 3
11  #define SRV_CH_NUM (SRV_BANK_NUM * SRV_IDX_NUM)
12
13  #define SRV_USE_IDX 16
14
15  void PwmInit(void);
16  void PwmStart(void);
17  void PwmStop(void);
18  void PwmSetDuty(unsigned short ch, unsigned short duty);
19  unsigned short PwmGetDuty(unsigned short ch);
20
21  #endif
```

End Of List

pwm.c

```
 1  /************************************************************************/
 2  /*                                                                      */
 3  /*  FILE        :pwm.c                                                  */
 4  /*  DATE        :Tue, Jan 22, 2008                                      */
 5  /*  DESCRIPTION :サーボモータ関連関数                                   */
 6  /*  CPU TYPE    :SH7125                                                 */
 7  /*                                                                      */
 8  /*  PWM1 PE0                                                            */
 9  /*  PWM2 PE1                                                            */
10  /*  PWM3 PE2                                                            */
11  /*                                                                      */
12  /*  コントロール端子 A PE4                                              */
13  /*  コントロール端子 B PE5                                              */
14  /*  コントロール端子 C PE6                                              */
15  /*                                                                      */
16  /************************************************************************/
17
18  #include <machine.h>
19  #include "iodefine.h"
20  #include "vect.h"
21  #include "pwm.h"
22
23  volatile unsigned short srv_index;
24  volatile unsigned long pwm_duty[SRV_BANK_NUM][SRV_IDX_NUM];
25
26  /*****************************
27     PWM 出力端子を初期化
28  *****************************/
29
30  void PwmInit(void)
31  {
```

```
32    srv_index=0;
33
34    MTU2.TSTR.BIT.CST0=0; /* CTNT_0 カウント停止 */
35
36    /* I/O 設定 */
37    PFC.PECRL1.WORD = 0x0111;    /* TIOC0A-C 入出力 */
38    PFC.PECRL2.WORD = 0x0000;    /* PE4-6 は PE 入出力 */
39    PFC.PEIORL.BYTE.L |= 0x77;  /* 0-2,4-6 は出力 */
40
41    STB.CR4.BIT._MTU2=0;
42
43    MTU20.TMDR.BIT.MD=3;    /* PWM モード 2 */
44
45    MTU20.TCR.BIT.TPSC=0;  /* φでカウント (0.0417uS,MAX2.73mS) */
46    MTU20.TCR.BIT.CKEG=0;  /* 立ち上がりエッジでカウント */
47    MTU20.TCR.BIT.CCLR=6;  /* TGRD のコンペアマッチで TCNT クリア */
48
49    MTU20.TIER.BIT.TGIED=1;  /* TGRD のコンペアマッチで割り込み */
50
51    MTU20.TIOR.BIT.IOA=2;  /* 初期値 0, コンペアマッチで 1 */
52    MTU20.TIOR.BIT.IOB=2;  /* 初期値 0, コンペアマッチで 1 */
53    MTU20.TIOR.BIT.IOC=2;  /* 初期値 0, コンペアマッチで 1 */
54
55    MTU20.TGRA=PWM_PERIOD-((PWM_MAX_DUTY+PWM_MIN_DUTY)>>1);
56    MTU20.TGRB=PWM_PERIOD-((PWM_MAX_DUTY+PWM_MIN_DUTY)>>1);
57    MTU20.TGRC=PWM_PERIOD-((PWM_MAX_DUTY+PWM_MIN_DUTY)>>1);
58    MTU20.TGRD=PWM_PERIOD;
59
60    MTU20.TCNT=0x0000;
61
62    MTU2.TSTR.BIT.CST0=1; /* CTNT_0 カウント開始 */
63    /* 割り込みフラグのクリア */
64    MTU20.TSR.BIT.TGFD=0;
65
66    INTC.IPRD.BIT._MTU20G=13;
67    set_imask(10);
68  }
69
70  /*********************************
71    割り込み関数
72   *********************************/
73  void INT_MTU2_0_TGID0(void)
74  {
75    /* チャンネルごとに PWM デューティを設定 */
76    MTU20.TGRA = pwm_duty[0][srv_index];
77    MTU20.TGRB = pwm_duty[1][srv_index];
78    MTU20.TGRC = pwm_duty[2][srv_index];
79
80    /* 割り込みフラグのクリア */
81    MTU20.TSR.BIT.TGFD=0;
82
83    /* インデックスをセット */
84    PE.DRL.BYTE.L = (PE.DRL.BYTE.L & ~0x70) | (srv_index << 4);
85
86    /* インデックスの更新 */
87    srv_index++;
88    if(srv_index >= SRV_IDX_NUM){
89      srv_index = 0;
90    }
91  }
92
93  /*********************************
94    サーボ動作開始
95   *********************************/
96  void PwmStart(void)
97  {
98    MTU2.TSTR.BIT.CST0=1; /* TCNT_0 カウント開始 */
99  }
100
101 /*********************************
102   サーボ動作停止
103  *********************************/
104 void PwmStop(void)
105 {
106   MTU2.TSTR.BIT.CST0=0; /* TCNT_0 カウント停止 */
107 }
108
109 /*********************************
110   デューティの設定
111  *********************************/
112 void PwmSetDuty(unsigned short ch, unsigned short duty)
113 {
114   /* サーボチャンネルが範囲外なら無視 */
115   if(ch >= SRV_USE_IDX){
116     return ;
117   }
118
119   if(duty < PWM_MIN_DUTY){          /* デューティを最小値以上に限定 */
120     duty = PWM_MIN_DUTY;
121   }else if(duty > PWM_MAX_DUTY){    /* デューティを最大値以下に限定 */
122     duty = PWM_MAX_DUTY;
123   }
```

```
124   /* デューティを設定 */
125   pwm_duty[ch >> SRV_BANK_NUM][ch & SRV_IDX_MASK] = PWM_PERIOD-duty;
126 }
127
128 /*********************************
129   デューティの取得（パラメータから）
130  *********************************/
131 unsigned short PwmGetDuty(unsigned short ch)
132 {
133   /* サーボチャンネルが範囲外なら無視 */
134   if(ch >= SRV_USE_IDX){
135     return (0);
136   }
137
138   /* デューティを返す */
139   return (PWM_PERIOD-pwm_duty[ch >> SRV_BANK_NUM][ch & SRV_IDX_MASK]);
140 }
```

End Of List

sci.h

```
 1 #ifndef _SCI_H_
 2 #define _SCI_H_
 3
 4 #define SCI_TX_BUFF_SIZE 128
 5 #define SCI_TX_BUFF_MASK (SCI_TX_BUFF_SIZE-1)
 6 #define SCI_RX_BUFF_SIZE 32
 7 #define SCI_RX_BUFF_MASK (SCI_RX_BUFF_SIZE-1)
 8
 9 typedef enum{
10   br19200=39,
11   br38400=19,
12   br57600=13
13 }BaudRate;
14
15 typedef struct{
16   unsigned short start_index;
17   unsigned short data_count;
18   char data_buff[SCI_RX_BUFF_SIZE];
19 }SciRxBuffer;
20
21
22 void Sci1Init(BaudRate b);
23
24 void Rx1BuffClear(void);
25
26 char Sci1Read(void);
27 int Sci1Putchar(char c);
28 int Sci1Puts(char *str);
29
30 char Sci1Getchar(void);
31 void Sci1Gets(char *str);
32
33 #endif
```

End Of List

sci.c

```
 1 /************************************************************************/
 2 /*                                                                      */
 3 /*  FILE        :sci.c                                                  */
 4 /*  DESCRIPTION :シリアル用関数                                         */
 5 /*  CPU TYPE    :SH7125                                                 */
 6 /*                                                                      */
 7 /*  RxD1 PA3                                                            */
 8 /*  TxD1 PA4                                                            */
 9 /*                                                                      */
10 /************************************************************************/
11 #include <machine.h>
12 #include "iodefine.h"
13 #include "vect.h"
14 #include "sci.h"
15
16 #define CHECK_LOOP 0x200000
17 #define CR 0x0D
18 #define LF 0x0A
19
20 static SciTxBuffer tx1_buff;
21 static SciRxBuffer rx1_buff;
22
23 /*********************************
24    シリアルポートを初期化 :Sci1Init
25 *********************************/
26 void Sci1Init(BaudRate b)
27 {
```

```
28    STB.CR3.BIT._SCI1=0;     /* SCI1 へクロック供給 */
29    SCI1.SCSCR.BYTE=0x00;    /* 送受信、送受信割り込み禁止 */
30    SCI1.SCSMR.BYTE=0x00;
31     /* 調歩同期式、8 ビット、パリティなし、1 ストップビット */
32    SCI1.SCBRR=b;            /* ボーレートの設定 */
33
34    PFC.PACRL2.BIT.PA4MD=1; /* 端子を RXD1 入力に指定 */
35    PFC.PACRL1.BIT.PA3MD=1; /* 端子を TXD1 出力に指定 */
36
37    SCI1.SCSCR.BIT.TE=1;     /* 送信許可 */
38    SCI1.SCSCR.BIT.RE=1;     /* 受信許可 */
39    SCI1.SCSCR.BIT.RIE=1;    /* 受信割り込み許可 */
40
41    INTC.IPRL.BIT._SCI1=14; /* 割り込み優先レベル */
42    set_imask(10);           /* 割り込みマスク */
43  }
44
45  /*******************************
46     1 バイト送信関数 :Sci1Write
47  *******************************/
48  int Sci1Write(char c)
49  {
50    unsigned long i;
51    unsigned short write_flag=0;    /* 送信データ書き込みフラグ */
52
53    for(i=0; i<CHECK_LOOP; i++){
54      if(SCI1.SCSSR.BIT.TDRE){ /* 送信データ書き込み可能になるまで待つ */
55        write_flag=1;
56        break;
57      }
58    }
59    if(write_flag){
60      SCI1.SCTDR=c;           /* 送信データを格納 */
61      SCI1.SCSSR.BIT.TDRE=0; /* 送信データ書き込み可能レジスタクリア */
62      return(0);
63    }
64    return(-1);
65  }
66
67  /*******************************
68     1 バイト送信関数 (割り込み有り)
69  *******************************/
70  void INT_SCI1_TEI1(void)
71  {
72    char msg;
73
74    if(tx1_buff.data_count > 0){
75      msg=tx1_buff.data_buff[tx1_buff.start_index];
76      tx1_buff.start_index=(tx1_buff.start_index+1) & SCI_TX_BUFF_MASK;
77      tx1_buff.data_count--;
78      while(!SCI1.SCSSR.BIT.TDRE){ /* 送信データ書き込み可能になるまで待つ */
79        ;
80      }
81      SCI1.SCTDR=msg;           /* 送信データを格納 */
82      SCI1.SCSSR.BIT.TDRE=0; /* 送信データ書き込み可能レジスタクリア */
83    }else{
84      SCI1.SCSCR.BIT.TEIE=0;
85    }
86  }
87
88  int Sci1Tx1(char msg)
89  {
90    int index;
91
92    SCI1.SCSCR.BIT.TEIE=0;
93    if(tx1_buff.data_count < SCI_TX_BUFF_SIZE){
94      index=(tx1_buff.start_index+tx1_buff.data_count) & SCI_TX_BUFF_MASK;
95      tx1_buff.data_buff[index]=msg;
96      tx1_buff.data_count++;
97      SCI1.SCSCR.BIT.TEIE=1;
98      return(0);
99    }
100   SCI1.SCSCR.BIT.TEIE=1;
101   return(-1);
102 }
103
104 /*******************************
105    1 バイト受信関数 :Sci1Read
106 *******************************/
107 char Sci1Read(void)
108 {
109   unsigned long i;
110   char read_char;
111
112   while(!SCI1.SCSSR.BIT.RDRF){ /* データを受信するまで待つ */
113     ;
114   }
115   read_char=SCI1.SCRDR;      /* データを変数へ */
116   SCI1.SCSSR.BIT.RDRF=0;
117
118   return(read_char);
119 }
```

```
120
121  /*******************************
122     1 バイト受信関数 (割り込み有り)
123  *******************************/
124  void INT_SCI1_RXI1(void)
125  {
126    char msg;
127    int index;
128
129    if(SCI1.SCSSR.BIT.RDRF){
130      SCI1.SCSSR.BIT.RDRF=0;
131      msg=SCI1.SCRDR;
132      if(rx1_buff.data_count < SCI_RX_BUFF_SIZE){
133        index=(rx1_buff.start_index+rx1_buff.data_count) & SCI_RX_BUFF_MASK;
134        rx1_buff.data_buff[index]=msg;
135        rx1_buff.data_count++;
136      }
137    }
138  }
139
140  int Sci1Rx1(char *msg)
141  {
142    if(rx1_buff.data_count > 0){
143      *msg=rx1_buff.data_buff[rx1_buff.start_index];
144      rx1_buff.start_index=(rx1_buff.start_index+1) & SCI_RX_BUFF_MASK;
145      rx1_buff.data_count--;
146      return(0);
147    }
148    *msg=0;
149    return(-1);
150  }
151
152  /***********************************************
153  *    1 文字受信 : Sci1Read
154  ***********************************************/
155  char Sci1Read(void)
156  {
157    char c;
158    while(Sci1Rx1(&c)){
159      ;
160    }
161    return(c);
162  }
163
164  /*************************************
165     受信バッファクリア関数 :Rx1BuffClear
166  *************************************/
167  void Rx1BuffClear(void)
168  {
169    rx1_buff.start_index=0;
170    rx1_buff.data_count=0;
171  }
172
173  /*****************************************************************************
174     汎用関数
175  *****************************************************************************/
176
177  /*******************************************************
178     1 バイト送信関数 (改行 (\n) は CR+LF に変換) :Sci1Putchar
179  *******************************************************/
180
181  int Sci1Putchar(char c)
182  {
183    /* 改行 (\n) は CR+LF に変換して送信 */
184    if(c=='\n'){
185      if(Sci1Write(CR)){
186        return(-1);
187      }
188      if(Sci1Write(LF)){
189        return(-1);
190      }
191    }else{  /* それ以外はそのまま送信 */
192      if(Sci1Write(c)){
193        return(-1);
194      }
195    }
196    return(0);
197  }
198
199  /*******************************************************
200  *    文字列送信 (改行 (\n) は CR+LF に変換) :Sci1Puts
201  *******************************************************/
202  int Sci1Puts(char *str)
203  {
204    char *msg;
205
206    /* 終端文字まで 1 文字ずつ表示 */
207    for(msg=str; *msg!='\0'; msg++){
208      if(Sci1Putchar(*msg)){
209        return(0);
210      }
211    }
212    return(msg-str);  /* 表示文字数を返す */
```

```
213  }
214
215
216  /************************************************
217  *    1 文字受信 (CR は改行 (\n) に変換) : Sci1Getchar
218  ************************************************/
219  char Sci1Getchar(void)
220  {
221    char c;
222    c=Sci1Read();
223    if(c==CR){
224      c='\n';
225    }
226    return(c);
227  }
228
229  /************************************************
230  *    文字列受信 (CR は改行 (\n) に変換) : Sci1Gets
231  ************************************************/
232  void Sci1Gets(char *str)
233  {
234    int i=0;
235    char s;
236
237    while((s=Sci1Getchar()) != '\n'){
238      str[i]=s;
239      i++;
240    }
241    str[i]='\0';
242  }
```

End Of List

str.h

```
1   #ifndef _STR_H_
2   #define _STR_H_
3
4   int StrIsDigit(char);
5   int StrIsHex(char);
6   int StrIsEos(signed char c);
7   int StrDecToLong(char *str, long *n);
8   int StrHexToLong(char *str, long *n);
9   int StrLongToDec(long n,char *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 StrIsDigit(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 StrIsHex(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 StrIsEos(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 *str, long *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; StrIsDigit(*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(!StrIsEos(*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 *str, long *n)
75  {
76
77    /* 16 進数文字を数値に変換 */
78    for(*n=0; StrIsHex(*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(!StrIsEos(*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 *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

実行結果

実行結果も、P.368 の図 13.10 と同じものです。

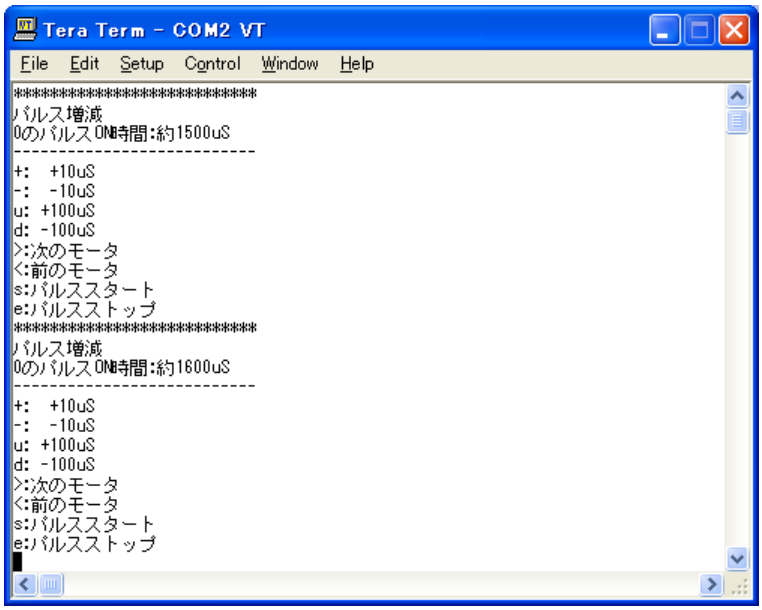

図 19.7 シリアル通信でサーボモータを動かす

19.5 SH7144F への移植

SH7144F はルネサスエレクトロニクス株式会社製の SH2 マイコンです。3.3V で駆動するので、MicroSD カードや 3.3V で駆動する液晶表示器をレベル変換なしに利用することができます。

CPU ボードとして株式会社秋月電子通商製の SH7144F マイコンボードを利用しました。クリスタルが 12.0MHz のものを用いました。内部 4 倍動作で 48MHz で動作します。

このマイコンも HEW を用いて開発ができます。

回路図

以下の回路は、ブザー、液晶表示器、MicroSD ソケットなどを搭載した SH7144F 基板の回路図です。

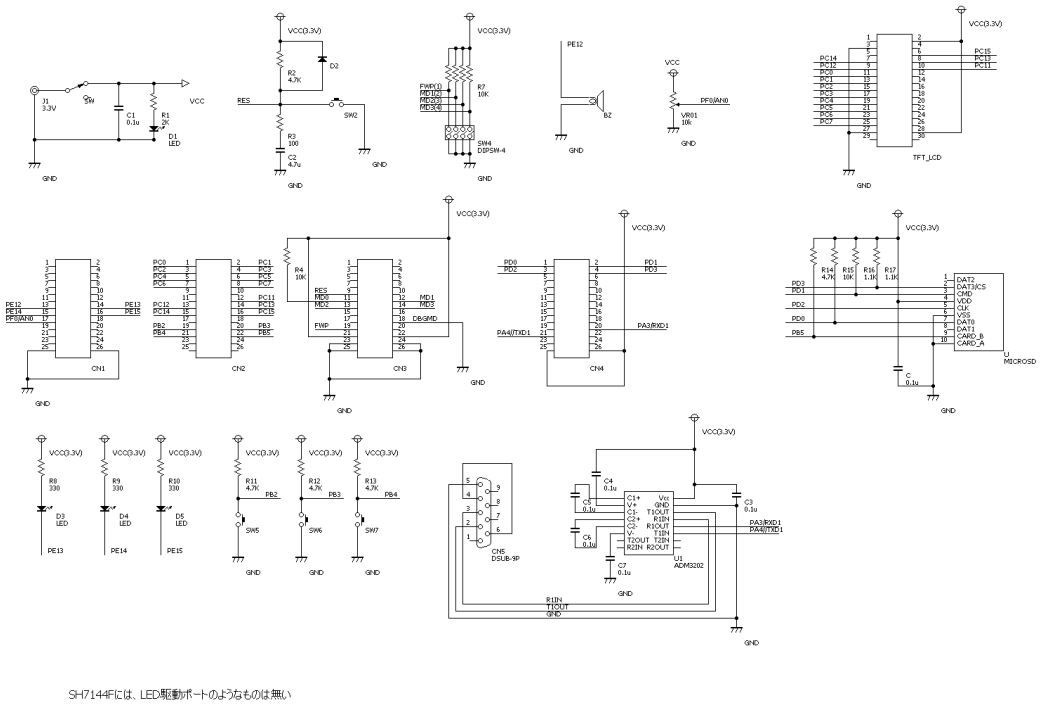

図 19.8 SH7144F 回路図

図 19.9 は作成した、SH7144F 基板基板です。

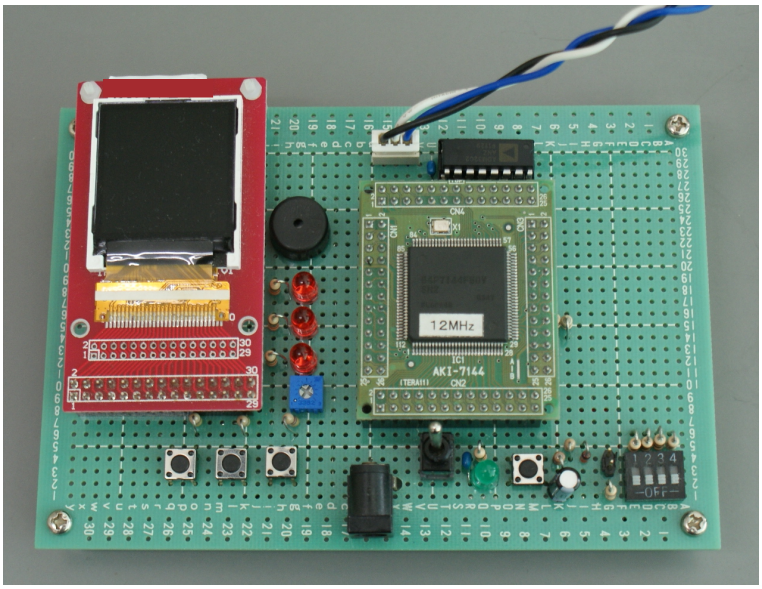

図 19.9 SH7144F 基板

今回は、音楽を再生するプログラムを移植してみましょう。

intprg.c の INT_CMT_CMT0 から関数 MusicPlay を呼び出してください。

intprg.c の INT_CMT_CMT0 から関数 MusicPlay を呼び出す

```
// 144 CMT CMT0
void INT_CMT_CMT0(void){MusicPlay();}
```

10_MUSIC01.c

```
 1  /***********************************************************************/
 2  /*                                                                     */
 3  /*  FILE        :10_MUSIC01.c                                          */
 4  /*  DESCRIPTION :音楽を鳴らす                                          */
 5  /*  CPU TYPE    :SH7144F                                               */
 6  /*                                                                     */
 7  /*  This file is generated by Renesas Project Generator (Ver.4.9).     */
 8  /*                                                                     */
 9  /***********************************************************************/
10
11  #include <machine.h>
12  #include "music.h"    /* 音階や音の長さの定義 */
13  #include "iodefine.h"
14
15  void main(void)
16  {
17
18    /*********************************************
19      音楽用タイマの初期化
20    *********************************************/
21    MusicInit();
22    set_imask(8);
23
24    while(1){
25      ;
26    }
27  }
```

End Of List

music.h

```
#ifndef _MUSIC_TIMER_H_
#define _MUSIC_TIMER_H_

/*------------------------------------------------------------------------*/
/*  ファイル名：  music_timer.h                                           */
/*        内容：  音楽用タイマ関数ヘッダファイル                          */
/*      作成者：  ono                                                     */
/*------------------------------------------------------------------------*/

//音の長さの定義
#define TEMPO 50     /* L1 の値に反映させる */
#define L1  32*TEMPO    //全音符長
//#define L1  16*TEMPO    //全音符長
#define L2  (L1/2) //2 分音符長
#define L4  (L1/4) //4 分音符長
#define L8  (L1/8) //8 分音符長
#define L16 (L1/16)//16 分音符長
#define L32 (L1/32)//32 分音符長
#define F2  ((L1*3)/4) //付点 2 分音符長
#define F4  ((L1*3)/8) //付点 4 分音符長
#define L3  (L4/3) //3 連符長
#define L6  (L4/6) //6 連符長

//音の高さの定義 ラ：440Hz
#define do0  5733 //ド
#define dos0 5411 //ド#
#define re0  5108 //レ
#define res0 4821 //レ#
#define mi0  4551 //ミ
#define fa0  4295 //ファ
#define fas0 4054 //ファ#
#define so0  3827 //ソ
#define sos0 3612 //ソ#
#define ra0  3409 //ラ
#define ras0 3218 //ラ#
#define si0  3037 //シ
#define do1  2867 //ド
#define dos1 2706 //ド#
#define re1  2554 //レ
#define res1 2411 //レ#
#define mi1  2275 //ミ
#define fa1  2148 //ファ
#define fas1 2027 //ファ#
#define so1  1913 //ソ
#define sos1 1806 //ソ#
#define ra1  1704 //ラ
#define ras1 1609 //ラ#
#define si1  1519 //シ
#define do2  1433 //ド

//音のデータ
struct NoteData{
  unsigned int Freq;  //音程
  unsigned int  Len;  //長さ
};

void MusicInit(void);
#endif
```

End Of List

music.c

```
/************************************************************************/
/*                                                                      */
/*  FILE        :music   .c                                             */
/*  DESCRIPTION :音楽用関数                                             */
/*  CPU TYPE    :SH7144FF                                               */
/*                                                                      */
/************************************************************************/

/* インクルードファイル */
#include"iodefine.h"
#include "music.h"    /* 音階や音の長さの定義 */
#include "music_data"       /* 楽譜データ */

volatile struct NoteData CNote; /* Freq：音程, Len：音の長さ */
volatile int NoteCnt=0;   /* 音符の何番目を演奏しているか */
volatile int ZeroCnt=50;   /* 音の最後の消音 1mS 単位 */
```

```
17  volatile unsigned int fdata; /* 音階を格納する変数 */
18  int note_max=NOTEMAX; /* 楽譜データの要素数 */
19
20  /*********************************************
21      音の開始・停止操作
22  *********************************************/
23  void StopPwmTimer(void)
24  {
25     MTU.TSTR.BIT.CST4=0;
26  }
27  void StartPwmTimer(void)
28  {
29     MTU.TSTR.BIT.CST4=1;
30  }
31  /*********************************************
32      音の長さをカウントするタイマの開始・停止操作
33  *********************************************/
34  void StopLengthTimer(void)
35  {
36     CMT.CMSTR.BIT.STR0=0;
37  }
38  void StartLengthTimer(void)
39  {
40     CMT.CMSTR.BIT.STR0=1;
41  }
42
43  /*********************************************
44      音用タイマの初期化 24MHz 16 分周
45      TIOC4A(CN1-17) に出力
46  *********************************************/
47  void PwmTimerInit(void)
48  {
49
50    MST.CR2.BIT._MTU=0;
51    MTU4.TCR.BYTE=0x22;
52    MTU4.TMDR.BYTE=0xC0;
53    MTU4.TIOR.BIT.IOA=3;
54    MTU.TOER.BIT.OE4A=1;
55    PFC.PECRL1.BIT.PE12MD=1;
56    PFC.PEIORL.BIT.B12=1;
57  }
58
59  /*********************************************
60      音の長さ用タイマの初期化 1ms ごとに割り込みをかける設定
61       25MHz 8 分周
62  *********************************************/
63  void LengthTimerInit(void)
64  {
65      MST.CR2.BIT._CMT =0;    /* モジュールスタンバイモードを解除 */
66      CMT0.CMCSR.BIT.CMIE =1; /* コンペアマッチ割込み許可 */
67      CMT0.CMCSR.BIT.CKS =0;  /* 8 分周 */
68      CMT0.CMCSR.BIT.CMF =0;  /* コンペアマッチフラグクリア */
69      CMT0.CMCOR = 3000-1;   /* 1m 秒をカウント */
70      INTC.IPRG.BIT._CMT0 = 13; /* 割込み優先度 */
71  }
72
73  /*********************************************
74      音をセット
75  *********************************************/
76  void  SetData(unsigned int freq)
77  {
78      MTU4.TGRA=(freq>>1);   /* ON, OFF は半々 エンベロープを入れるならここを変更 */
79  }
80
81  /*********************************************
82      音の長さのタイマ割込みフラグをクリア
83  *********************************************/
84  void  ClearLegthFlag(void)
85  {
86      CMT0.CMCSR.BIT.CMF =0;
87  }
88
89  /*-------------------------------------------
90     以下、ハードウェア依存性の無いプログラム
91     ただし、二つのタイマの割り込み関数は修正の必要あり
92     (今の場合音はトグル出力で出しているので割込み関数は無い)
93  -------------------------------------------*/
94
95  /*********************************************
96      最初のデータをセット　休符のときの処理
97  *********************************************/
98  void  SetInitData(void)
99  {
100   CNote=Note[NoteCnt];
101   if(CNote.Freq == 0){
102     StopPwmTimer();
103   }else{
104     SetData(CNote.Freq);
105     StartPwmTimer();
106   }
107 }
```

```
108
109 /*******************************************
110     音楽演奏のための初期化関数
111 *******************************************/
112 void MusicInit(void)
113 {
114     PwmTimerInit();
115     LengthTimerInit();
116     SetInitData();
117     StartPwmTimer();
118     StartLengthTimer();
119 }
120
121 /*******************************************
122     IMIA1 割込み関数 (音の長さ)
123 *******************************************/
124 void MusicPlay(void){
125   ClearLegthFlag();
126   if(CNote.Len>0){
127     CNote.Len--; /* エンベロープを入れるならここで対応 */
128   }else if(NoteCnt<(note_max-1) && ZeroCnt>0){ /* 消音を入れて同じ音が続いたときの対処 */
129     if(ZeroCnt==50){
130       StopPwmTimer();
131     }
132     ZeroCnt--;
133   }else if(NoteCnt<(note_max-1) && ZeroCnt<=0){  /* CNote.Len==0 音の終了時 */
134     ZeroCnt=50;
135     NoteCnt++;
136     CNote=Note[NoteCnt];
137     fdata=CNote.Freq;   /* Freq：音階 タイマ A0 */
138     if(fdata==0){       /* 音階データが 0 の場合には音を出さない */
139       StopPwmTimer(); /* タイマ A0 を停止 */
140     }else{
141       SetData(fdata);
142       StartPwmTimer();
143     }
144   }else{ /* 楽譜の最後で音を止める */
145     StopPwmTimer();
146     StopLengthTimer();
147   }
148
149 }
```

End Of List

music_data

```
1  //チューリップ
2
3  #define NOTEMAX 42
4
5  const struct NoteData Note[NOTEMAX]={
6    {do0,L4},{re0,L4},{mi0,L4},{  0,L4},{do0,L4},{re0,L4},{mi0,L4},{ 0,L4},
7    {so0,L4},{mi0,L4},{re0,L4},{do0,L4},{re0,L4},{mi0,L4},{re0,L2},
8    {do0,L4},{re0,L4},{mi0,L4},{  0,L4},{do0,L4},{re0,L4},{mi0,L4},{ 0,L4},
9    {so0,L4},{mi0,L4},{re0,L4},{do0,L4},{re0,L4},{mi0,L4},{do0,L2},
10   {so0,L4},{so0,L4},{mi0,L4},{so0,L4},{ra0,L4},{ra0,L4},{so0,L2},
11   {mi0,L4},{mi0,L4},{re0,L4},{re0,L4},{do0,L2}
12 };
```

End Of List

実行結果

童謡「チューリップ」を演奏する。

19.6 ATMEGA328P への移植

ATMEGA328P は Atmel Corporation(アトメル社) の製品です。

今回は開発環境として AVR Studio と Win AVR を使いました。

クリスタルは使わず、1MHz で動かしています。

回路図

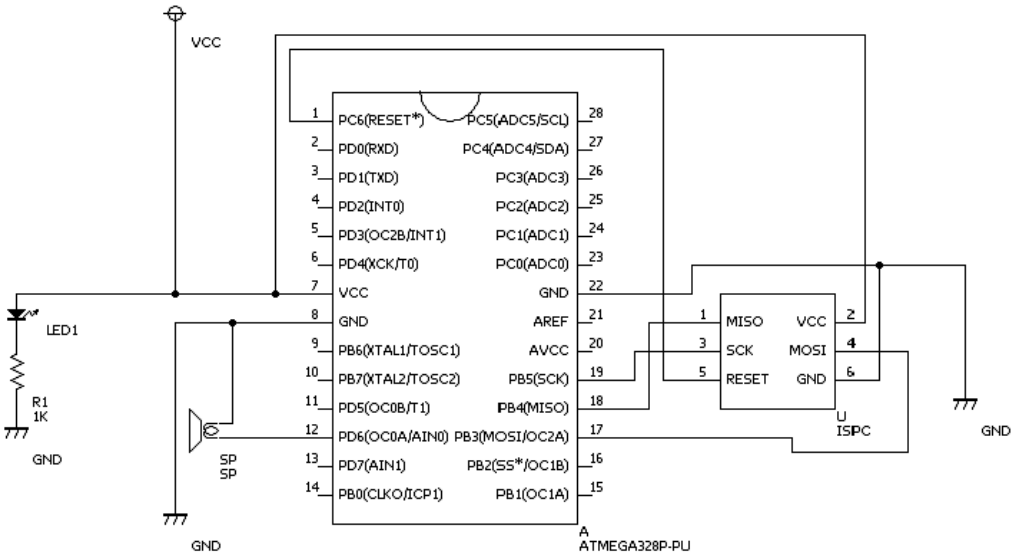

図 19.10 ブザーの回路

サンプルプログラムとして、音楽再生プログラムを移植してみました。

buzzer03.c

```
 1  /*-----------------------------------------------*/
 2  /*  ファイル名：         buzzer02.c               */
 3  /*         内容：         ブザーで音楽を鳴らす     */
 4  /*                        8 ビットタイマ 0         */
 5  /*       作成者：         ono                      */
 6  /*-----------------------------------------------*/
 7
 8  #include"music_timer.h"
 9
10  int main(void){
11
12    MusicTaInit();
13    while(1){
14      ;
15    }
16  }
```

End Of List

music_timer.h

```
1  #ifndef _MUSIC_TIMER_H_
2  #define _MUSIC_TIMER_H_
3
```

```
 4  /*-------------------------------------------------------------------------*/
 5  /*  ファイル名：  music_timer.h                                            */
 6  /*        内容：  音楽用タイマ関数ヘッダファイル                           */
 7  /*        日付：  2006/08/09                                               */
 8  /*      作成者：  ono                                                      */
 9  /*-------------------------------------------------------------------------*/
10
11  //音の長さの定義
12  #define TEMPO 50     /* L1 の値に反映させる */
13  #define L1  32*TEMPO    //全音符長
14  //#define L1  16*TEMPO    //全音符長
15  #define L2  (L1/2) //2 分音符長
16  #define L4  (L1/4) //4 分音符長
17  #define L8  (L1/8) //8 分音符長
18  #define L16 (L1/16)//16 分音符長
19  #define L32 (L1/32)//32 分音符長
20  #define F2  ((L1*3)/4) //付点 2 分音符長
21  #define F4  ((L1*3)/8) //付点 4 分音符長
22  #define L3  (L4/3) //3 連符長
23  #define L6  (L4/6) //6 連符長
24
25  //音の高さの定義
26  #define COE 16 //割り込み周期 500nS 単位
27  #define do0  3822/COE //ド
28  #define dos0 3608/COE
29  #define re0  3405/COE
30  #define res0 3214/COE
31  #define mi0  3034/COE
32  #define fa0  2863/COE
33  #define fas0 2703/COE
34  #define so0  2551/COE
35  #define sos0 2408/COE
36  #define ra0  2273/COE
37  #define ras0 2145/COE
38  #define si0  2025/COE
39  #define do1  1911/COE //ド
40  #define dos1 1804/COE
41  #define re1  1703/COE
42  #define res1 1607/COE
43  #define mi1  1517/COE
44  #define fa1  1432/COE
45  #define fas1 1351/COE
46  #define so1  1276/COE
47  #define sos1 1204/COE
48  #define ra1  1136/COE
49  #define ras1 1073/COE
50  #define si1  1012/COE
51  #define do2   956/COE //ド
52  #define dos2  902/COE
53  #define re2   851/COE
54  #define res2  804/COE
55  #define mi2   758/COE
56  #define fa2   716/COE
57  #define fas2  676/COE
58  #define so2   638/COE
59  #define sos2  602/COE
60  #define ra2   568/COE
61  #define ras2  536/COE
62  #define si2   506/COE
63  #define do3   478/COE //ド
64
65  //音のデータ
66  struct NoteData{
67    unsigned int Freq;  //音程
68    unsigned int  Len;  //長さ
69  };
70
71
72  void MusicTaInit(void);
73  #endif
```

End Of List

music_timer.c

```
 1  /*-------------------------------------------------------------------------*/
 2  /*  ファイル名：  music_timer.c                                            */
 3  /*        内容：  音楽用タイマ関数                                         */
 4  /*        日付：  2006/08/09                                               */
 5  /*      作成者：  ono                                                      */
 6  /*-------------------------------------------------------------------------*/
 7
 8  /* インクルードファイル */
 9  #include <avr/io.h>
10  #include <avr/interrupt.h>
11  #include "music_timer.h" /* 音階や音の長さの定義 */
```

```
12  #include "music_data"       /* 楽譜データ */
13
14  volatile struct NoteData CNote; /* Freq：音程, Len：音の長さ */
15  volatile int NoteCnt=0;   /* 音符の何番目を演奏しているか */
16  volatile int ZeroCnt=50;   /* 音の最後の消音 1mS 単位 */
17  volatile unsigned int fdata; /* 音階を格納する変数 */
18  int note_max=NOTEMAX; /* 楽譜データの要素数 */
19
20  #define BUZZER_PORT PORTD
21  #define BUZZER_DDR  DDRD
22  #define BUZZER_MASK 0x40
23  #define BUZZER_SHIFT 6
24
25  /*----------------------------
26    BUZZER で使用する端子の初期化
27  ----------------------------*/
28
29  void BuzzerInit(void){
30    BUZZER_DDR = BUZZER_MASK;
31  }
32
33  /*----------------------------
34    BUZZER を鳴らす
35  ----------------------------*/
36
37  void BuzzerOn(void){
38    BUZZER_PORT |= BUZZER_MASK;
39  }
40
41  /*----------------------------
42    BUZZER を止める
43  ----------------------------*/
44
45  void BuzzerOff(void){
46    BUZZER_PORT &= ~BUZZER_MASK;
47  }
48
49  /*-------------------------------------------------------------------------*/
50  /* タイマ 0 開始 */
51  /*-------------------------------------------------------------------------*/
52  void MusicTaStart(void)
53  {
54    TCCR0B=0x02;  /* 8 分周 (8uS) */
55  }
56  /*-------------------------------------------------------------------------*/
57  /* タイマ 0 停止 */
58  /*-------------------------------------------------------------------------*/
59  void MusicTaStop(void)
60  {
61    TCCR0B=0x00;  /* タイマ 0 停止 */
62  }
63
64  /*-------------------------------------------------------------------------*/
65  /* タイマ 0 カウント数変更 */
66  /*-------------------------------------------------------------------------*/
67  void MusicTaSet(int onkai)
68  {
69    OCR0A=onkai;
70  }
71
72  /*-------------------------------------------------------------------------*/
73  /* タイマ A0,1 初期化 */
74  /*-------------------------------------------------------------------------*/
75  void MusicTaInit(void)
76  {
77    BuzzerInit();
78    CNote=Note[NoteCnt];
79    TCCR0A=0x42;  /* OC0A 一致でトグル、比較一致 */
80    TCCR0B=0x02;  /* 8 分周 (8uS) */
81    TCCR1A=0x00;  /* OC1A 出力禁止 */
82    TCCR1B=0x0A;  /* 比較一致,8 分周 (8uS) */
83    OCR1AL=125;   /* 1mS */
84    TIMSK1=0x02;  /* 比較 1A 割り込み許可 */
85    if(CNote.Freq == 0){
86      MusicTaStop();
87      BuzzerOff();
88    }else{
89      MusicTaSet(CNote.Freq>>1);   /* ON, OFF は半々 エンベロープを入れるならここを変更 */
90      MusicTaStart();
91      BuzzerOn();
92    }
93    sei();//割り込み許可
94  }
95
96  /*-------------------------------------------------------------------------*/
97  /* タイマ 1 割り込み関数 1mS ごとの割り込み*/
98  /*-------------------------------------------------------------------------*/
99  ISR(TIMER1_COMPA_vect){
100 //  int i;
101   if(CNote.Len>0){
102     CNote.Len--; /* エンベロープを入れるならここで対応 */
103   }else if(NoteCnt<(note_max-1) && ZeroCnt>0){ /* 消音を入れて同じ音が続いたときの対処 */
```

```
104       if(ZeroCnt==50){
105         MusicTaStop();
106         BuzzerOff();
107       }
108       ZeroCnt--;
109     }else if(NoteCnt<(note_max-1) && ZeroCnt<=0){  /* CNote.Len==0 音の終了時 */
110       ZeroCnt=50;
111       NoteCnt++;
112       CNote=Note[NoteCnt];
113       fdata=CNote.Freq;   /* Freq：音階 タイマ A0 */
114       if(fdata==0){       /* 音階データが 0 の場合には音を出さない */
115         MusicTaStop();
116         BuzzerOff();
117       }else{
118       MusicTaSet(CNote.Freq>>1);
119       MusicTaStart();
120       }
121     }else{ /* 楽譜の最後で音を止める */
122       MusicTaStop();
123       BuzzerOff();
124     }
125   }
126
```

End Of List

music_data

```
1  //チューリップ
2
3  #define NOTEMAX 42
4
5  const struct NoteData Note[NOTEMAX]={
6    {do0,L4},{re0,L4},{mi0,L4},{  0,L4},{do0,L4},{re0,L4},{mi0,L4},{ 0,L4},
7    {so0,L4},{mi0,L4},{re0,L4},{do0,L4},{re0,L4},{mi0,L4},{re0,L2},
8    {do0,L4},{re0,L4},{mi0,L4},{  0,L4},{do0,L4},{re0,L4},{mi0,L4},{ 0,L4},
9    {so0,L4},{mi0,L4},{re0,L4},{do0,L4},{re0,L4},{mi0,L4},{do0,L2},
10   {so0,L4},{so0,L4},{mi0,L4},{so0,L4},{ra0,L4},{ra0,L4},{so0,L2},
11   {mi0,L4},{mi0,L4},{re0,L4},{re0,L4},{do0,L2}
12 };
```

End Of List

実行結果

童謡「チューリップ」を演奏する。