```
/*------------------------------------------------------------------------*/
/*  ものつくりコンテスト山形県大会08 電子回路組立部門                     */
/*                                                                        */
/* 全工長協会HP掲示  制御プログラム課題(例)                               */
/*                                                                        */
/* 課題5   物体の位置に相当するＬＥＤが点灯し、タクトスイッチからパルス信号が */
/*         入ればＬＥＤが左にシフトする。                                 */
/*                                                                        */
/*   センサーの前に物体を置き、その位置のＬＥＤを点灯させる。タクトスイッチを */
/*   押したとき、左にシフトする。物体が移動したときはその位置を点灯させ、その */
/*   場所から左シフトする。                                               */
/*                                                                        */
/*      範囲（目安）      測定位置      点灯LED                           */
/*      10      15cm      10cm          LED0                              */
/*      15      25cm      20cm          LED1                              */
/*      25      35cm      30cm          LED2                              */
/*      35      45cm      40cm          LED3                              */
/*      45      55cm      50cm          LED4                              */
/*      55      65cm      60cm          LED5                              */
/*      65      75cm      70cm          LED6                              */
/*      75      80cm      80cm          LED7                              */
/*                                                                        */
/*   初期状態                                                             */
/*     (1) センサは１ｍ以上前に何もおいていない。                         */
/*     (2) トグルスイッチはＬの位置にある。                               */
/*     (3) タクトスイッチは押されていない。                               */
/*     (4) 光センサは光を受光している。                                   */
/*     (5) タクトスイッチ動作による状態変化は、特に指定がない限り押した瞬間 */
/*         に行うものとする。                                             */
/*                                                                        */
/* ※今後確認                                                             */
/*   距離のデータはあらかじめ測定したものを使用した。                     */
/*   「タクトスイッチが押したとき」は タクトスイッチ OFF -> ON -> OFF とした */
/*                                                                        */
/*                                山形電波工業高等学校 情報技術科 & 電子工学科 */
/*                                                Ver. 1.0 平成20年5月 6日(火) */
/*                                                Ver. 2.0 平成20年5月10日(土) */
/*                                                                        */
/*------------------------------------------------------------------------*/
//#include "comment08.h"

/*---------------------------------*/
/* 組み込みファイル定義            */
/*---------------------------------*/
  #include <3048.h>
  #include "macro08.h"


/*-------------------------------------------------*/
/* グローバル変数宣言(どの関数からも参照、変更できる) */
/*-------------------------------------------------*/
  unsigned char  psd_data;          /* 割り込みで表示処理する */

  unsigned char  sift_flag = 0;     /* 課題5 シフト処理するかどうか */

/*-------------------------------------------------*/
/* このプログラム中で使用する関数の記述            */
/*-------------------------------------------------*/
//#include "dempalib08.h"


/*------------------------------------------------------------------------*/
/*  モジュール名 Init_Port                                                */
/*  処理概要     H8 ポート初期化                                          */
/*  引数         なし                                                     */
/*  戻り値       なし                                                     */
/*------------------------------------------------------------------------*/
void Init_Port( void )
{
       /* I/Oポート設定  0:入力  1:出力 */
          P6.DDR = 0x00;            /* on bord ディップスイッチ        */
                                    /* 0x00 = 0000 0000  全bit入力設定 */

          P4.DDR = 0xff;            /* 事前配布基板(1) LED & 7segLED   */
                                    /* 0xff = 1111 1111  全bit出力設定 */
          P4.DR.BYTE = 0x00;

          PA.DDR = 0xff;            /*                                 */
                                    /* 0x00 = 0000 0000  全bit入力設定 */
          PA.DR.BYTE = 0x00;

          PB.DDR = 0xff;            /*                                 */
                                    /* 0xff = 1111 1111  全bit出力設定 */
          PB.DR.BYTE = 0x00;

          P5.DDR = 0xff;            /* 事前配布基板(1) LED or 7segLED指示端子 */
```

```
 87                                      /* 0xff = 1111 1111  全bit出力設定        */
 88          P5.DR.BYTE = 0x00;
 89
 90                                      /* 当日作成基板はP7ポート(P7は入力専用なので設定なし) */
 91 }
 92
 93
 94 /*------------------------------------------------------------------------------*/
 95 /*  モジュール名 Init_H8                                                        */
 96 /*  処理概要     H8 タイマー初期化                                              */
 97 /*  引数         なし                                                           */
 98 /*  戻り値       なし                                                           */
 99 /*------------------------------------------------------------------------------*/
100 /* 参考 H8/3048f(旧cpu) 14.7854MHz      */
101 /*       16ms   29490                   */
102 /*       10ms   18432                   */
103 /*        8ms   14745                   */
104 /*        5ms    9216                   */
105 /*        1ms    1843                   */
106 /*--------------------------------------*/
107 void Init_H8( void )
108 {
109     EI; /* 割り込み許可 */
110
111       /* タイマ設定      */
112          ITU.TMDR.BIT.PWM1 = 0;    /* ITU1:通常動作        */
113          ITU.TMDR.BIT.PWM2 = 0;    /* ITU2:通常動作        */
114          ITU.TMDR.BIT.PWM3 = 1;    /* ITU3:PWMモード       */
115          ITU.TMDR.BIT.PWM4 = 0;    /* ITU4:通常動作        */
116
117
118       /* iTu1 タイマ設定 */
119          ITU1.TCR.BIT.CCLR = 1;    /* カウンタクリア要因                          */
120          ITU1.TCR.BIT.CKEG = 0;    /* クロックエッジ                              */
121          ITU1.TCR.BIT.TPSC = 3;    /* タイマプリスケーラ 1.8432MHz                */
122          ITU1.GRA = 18432;         /* 割り込みインターバル 1.8432MHz/18432=100Hz */
123          ITU1.TIER.BIT.IMIEA = 1;  /* IMFAフラグによる割り込み許可                */
124
125       /* iTu2 タイマ設定 */
126          ITU2.TCR.BIT.CCLR = 1;    /* カウンタクリア要因                          */
127          ITU2.TCR.BIT.CKEG = 0;    /* クロックエッジ                              */
128          ITU2.TCR.BIT.TPSC = 3;    /* タイマプリスケーラ 1.8432MHz                */
129          ITU2.GRA = 18432;         /* 割り込みインターバル 1.8432MHz/18432=100Hz */
130          ITU2.TIER.BIT.IMIEA = 1;  /* IMFAフラグによる割り込み許可                */
131
132       /* iTu3 モータタイマ設定 */
133          ITU3.TCR.BIT.CCLR = 1;    /* カウンタクリア要因                          */
134          ITU3.TCR.BIT.CKEG = 0;    /* クロックエッジ                              */
135          ITU3.TCR.BIT.TPSC = 3;    /* タイマプリスケーラ 1.8432MHz                */
136          ITU3.GRA = PWM_CYCLE;     /* モータPWM周期 1.8432MHz/1843=1kHz           */
137          ITU3.GRB = 0;             /* モータPWM DUTY 0%                           */
138
139       /* タイマ設定 */
140          ITU4.TCR.BIT.CCLR = 1;    /* カウンタクリア要因                          */
141          ITU4.TCR.BIT.CKEG = 0;    /* クロックエッジ                              */
142          ITU4.TCR.BIT.TPSC = 3;    /* タイマプリスケーラ 1.8432MHz                */
143          ITU4.GRA = 18432;         /* 割り込みインターバル 1.8432MHz/18432=100Hz */
144          ITU4.TIER.BIT.IMIEA = 1;  /* IMFAフラグによる割り込み許可                */
145
146
147       /* タイマスタート */
148          ITU.TSTR.BIT.STR0 = 0;    /* まだスタートしない */
149          ITU.TSTR.BIT.STR1 = 0;    /* まだスタートしない */
150          ITU.TSTR.BIT.STR2 = 0;    /* まだスタートしない */
151          ITU.TSTR.BIT.STR3 = 0;    /* まだスタートしない */
152          ITU.TSTR.BIT.STR4 = 0;    /* まだスタートしない */
153 }
154
155
156
157 /*------------------------------------------------------------------------------*/
158 /*  モジュール名  start_itu0                                                    */
159 /*  処理概要      タイマーitu4のスタート                                        */
160 /*  引数          なし                                                          */
161 /*  戻り値        なし                                                          */
162 /*------------------------------------------------------------------------------*/
163 void start_warikomi( void )
164 {
165         ITU.TSTR.BIT.STR4 = 1; /* itu4スタート */
166 }
167
168
169 /*------------------------------------------------------------------------------*/
170 /*  モジュール名  stop_itu4                                                     */
171 /*  処理概要      タイマーitu4のストップ                                        */
172 /*  引数          なし                                                          */
```

```
173 /*  戻り値        なし                                                          */
174 /*--------------------------------------------------------------------------*/
175 void stop_warikomi( void )
176 {
177         ITU.TSTR.BIT.STR4 = 0;  /* itu4ストップ */
178 }
179
180
181 /*--------------------------------------------------------------------------*/
182 /*  モジュール名 課題5用 int_imia4                                           */
183 /*  処理概要                                                                */
184 /*  引数          なし                                                      */
185 /*  戻り値        なし                                                      */
186 /*--------------------------------------------------------------------------*/
187 void int_imia4( void )
188 {
189     int i;
190     unsigned char  w_data;        /* コンパイラ不良? worning対策 */
191
192           /*--------------------------*/
193           /* 通常点灯                 */
194           /*--------------------------*/
195                     if( psd_data >= 117 ) {
196                                         KIBAN081 = ~0x01;
197
198                     } else if( psd_data >= 80 ) {
199                                         KIBAN081 = ~0x02;
200
201                     } else if( psd_data >= 54 ) {
202                                         KIBAN081 = ~0x04;
203
204                     } else if( psd_data >= 41 ) {
205                                         KIBAN081 = ~0x08;
206
207                     } else if( psd_data >= 30 ) {
208                                         KIBAN081 = ~0x10;
209
210                     } else if( psd_data >= 26 ) {
211                                         KIBAN081 = ~0x20;
212
213                     } else if( psd_data >= 23 ) {
214                                         KIBAN081 = ~0x40;
215
216                     } else if( psd_data >= 20 ) {
217                                       /*KIBAN081 = ~0x80;    warning  */
218                                         w_data = 0x80;
219                                         KIBAN081 = ~w_data;
220
221                     } else {
222                                         KIBAN081 = ~0x00;
223                     }
224
225
226       /*-------------------------*/
227       /* reset interrupt request */
228       /*-------------------------*/
229          ITU4.TSR.BIT.IMFA = 0;
230 }
231
232
233 /*--------------------------------------------------------------------------*/
234 /*  モジュール名 wait                                                        */
235 /*  処理概要     ソフトウェアタイマー                                        */
236 /*  引数         タイマー値    1:  10[ms]                                    */
237 /*                            10: 100[ms] = 0.1[s]                          */
238 /*                            50: 500[ms] = 0.5[s]                          */
239 /*                           100:1000[ms] = 1.0[s]                          */
240 /*  戻り値       なし                                                        */
241 /*--------------------------------------------------------------------------*/
242 void wait( int iTimer )
243 {
244     int i;
245          while( iTimer ) {
246                for( i = 0; i < 5000; i++ );    /* H8/3048f */
247               /*for( i = 0; i < 8333; i++ );*/  /* H8/3048f-one */
248                     iTimer--;
249          }
250 }
251
252
253 /*--------------------------------------------------------------------------*/
254 /*  モジュール名  seg_cls                                                   */
255 /*  処理概要      7セグメントLED消去                                        */
256 /*  引数          なし                                                      */
257 /*  戻り値        なし                                                      */
258 /*--------------------------------------------------------------------------*/
```

```
259 void seg_cls( void )
260 {
261           KIBAN081= 0xff;    /* KIBAN081 = ~0x00; */
262
263 }
264
265
266 /*------------------------------------------------------------------------------*/
267 /*  モジュール名  openning                                                      */
268 /*  処理概要      オープニング(山形電波工高固有)                                */
269 /*  引数          なし                                                          */
270 /*  戻り値        なし                                                          */
271 /*------------------------------------------------------------------------------*/
272 void openning08( void )
273 {
274      signed char i;                     /* -128 ～ +127 */
275
276           SEG_SELECT = T7SEG;           /* 表示は7セグメントLED */
277           for( i = 3; i >= 0; i-- ) {
278                KIBAN081= tendo[ i ];    /* 表示 */
279                   wait( 50 );
280                seg_cls();               /* 消去 */
281                   wait( 50 );
282           }
283 }
284
285
286 /*------------------------------------------------------------------------------*/
287 /*  モジュール名 Init_ad                                                        */
288 /*  処理概要     ＡＤ初期化                                                     */
289 /*  引数         なし                                                           */
290 /*  戻り値       なし                                                           */
291 /*------------------------------------------------------------------------------*/
292 void Init_ad( void )
293 {
294        /* ADコントロールレジスタ設定 */
295           AD.CSR.BYTE = 0x00;     /* ch0(AN0 (P70) ) 単一モード */
296          /*AD.CSR.BYTE = 0x01;*/   /* ch1(AN1 (P71) ) 単一モード */
297          /*AD.CSR.BYTE = 0x02;*/   /* ch2(AN2 (P72) ) 単一モード */
298          /*AD.CSR.BYTE = 0x03;*/   /* ch3(AN3 (P73) ) 単一モード */
299          /*AD.CSR.BYTE = 0x04;*/   /* ch4(AN4 (P74) ) 単一モード */
300          /*AD.CSR.BYTE = 0x05;*/   /* ch5(AN5 (P75) ) 単一モード */
301          /*AD.CSR.BYTE = 0x06;*/   /* ch6(AN6 (P76) ) 単一モード */
302          /*AD.CSR.BYTE = 0x07;*/   /* ch7(AN7 (P77) ) 単一モード */
303
304 }
305
306
307 /*------------------------------------------------------------------------------*/
308 /* モジュール名 start_ad                                                        */
309 /* 処理概要     ＡＤ変換スタート                                                */
310 /* 引数         なし                                                            */
311 /* 戻り値       なし                                                            */
312 /*------------------------------------------------------------------------------*/
313 void start_ad( void )
314 {
315           AD.CSR.BIT.ADST = 1; /* AD変換開始 */
316
317 }
318
319
320 /*------------------------------------------------------------------------------*/
321 /* モジュール名 end_ad                                                          */
322 /* 処理概要     ＡＤ変換終了                                                    */
323 /* 引数         なし                                                            */
324 /* 戻り値       なし                                                            */
325 /*------------------------------------------------------------------------------*/
326 void end_ad( void )
327 {
328           while( AD.CSR.BIT.ADF != 1 ){}  /* AD変換終了チェック */
329
330 }
331
332
333 /*------------------------------------------------------------------------------*/
334 /* モジュール名 get_ad8                                                         */
335 /* 処理概要     データ読み込み                                                  */
336 /*              上位10bitにデータが格納されるので、下位8bitにシフトし(右に8bit) */
337 /*              変換データを確定する。(下位2bitゴミとしてカット)                */
338 /* 引数         なし                                                            */
339 /* 戻り値       8bit data                                                       */
340 /*              1111 1111 11-- ---- : 変換後データ                              */
341 /*              ---- ---- 1111 1111 : 8bitシフト後データ                        */
342 /*                                                                              */
343 /*              ---- ---- 0000 0000 : 0x00      0                               */
344 /*                        :                     :                               */
```

```
/*             ---- ---- 1111 1111 : 0xff     255  0～255 の 256通りのデータ   */
/*-------------------------------------------------------------------------------*/
unsigned char get_ad8( void )
{
     unsigned char indata;        /* unsigned intで 0 ～ 255 まで扱える */

          indata = AD.DRA >>8;    /* 上位10bitを右に8bitシフトする。下位2bit無視 */
                                  /* 1111 1111 1100 0000           */
                                  /* ---- ---- 1111 1111           */

        /*indata = AD.DRB >>8;*/  /* 上位10bitを右に8bitシフトする。*/
        /*indata = AD.DRC >>8;*/  /* 上位10bitを右に8bitシフトする。*/
        /*indata = AD.DRD >>8;*/  /* 上位10bitを右に8bitシフトする。*/
        //indata = indata & 0xff;

          return indata;
}


/*-------------------------------------------------------------------------------*/
/*  モジュール名 lsift                                                           */
/*  処理概要                                                                     */
/*  引数         なし                                                            */
/*  戻り値       なし                                                            */
/*-------------------------------------------------------------------------------*/
void lsift( void )
{
     unsigned char  i;
     unsigned char  w_data;        /* コンパイラ不良? worning対策 */

     unsigned char  sift_start_data;


             stop_warikomi();


                          /*--------------------------------*/
                          /* タクトスイッチ ON になった確認 */
                          /*--------------------------------*/
                             SEG_SELECT = T7SEG;
                             KIBAN081 = tendo[ 0 ];


           /*----------------------------*/
           /* タクトスイッチonのまま待つ */
           /*----------------------------*/
              while( DIP_SW_L0 ) {}
              while( DIP_SW_L0 ) {}

           /*----------------------------*/
           /* 割り込みによる表示処理停止 */
           /*----------------------------*/
           // stop_warikomi();

              KIBAN081 = ~0x00;
              SEG_SELECT = LED;


           /*--------------------------------*/
           /* タクトスイッチがoffになったので */
           /*--------------------------------*/

                 /*--------------------------------*/
                 /* あらためてpsdセンサのデータ取得 */
                 /*--------------------------------*/
                    start_ad();                /* AD変換スタート     */
                    end_ad();                  /* AD変換終了チェック */
                    psd_data = get_ad8();      /* psd読み込み        */


                 /*--------------------------------*/
                 /* センサデータの表示とシフト表示  */
                 /*--------------------------------*/
                        if( psd_data >= 117 ) {
                                          KIBAN081 = ~0x01;
                                          wait(50);
                                               sift_start_data = 0x01;
                                               for( i=1; i<8; i++ ) {
                                                    sift_start_data = sift_start_data<<1;
                                                    KIBAN081 = ~sift_start_data;
                                                    wait(50);
                                               }

                        } else if( psd_data >= 80 ) {
                                          KIBAN081 = ~0x02;
                                          wait(50);
```

```
                                        sift_start_data = 0x02;
                                        for( i=1; i<7; i++) {
                                             sift_start_data = sift_start_data<<1;
                                             KIBAN081 = ~sift_start_data;
                                             wait(50);
                                        }

                         } else if( psd_data >= 54 ) {
                                        KIBAN081 = ~0x04;
                                        wait(50);
                                             sift_start_data = 0x04;
                                             for( i=1; i<6; i++) {
                                                  sift_start_data = sift_start_data<<1;
                                                  KIBAN081 = ~sift_start_data;
                                                  wait(50);
                                             }

                         } else if( psd_data >= 41 ) {
                                        KIBAN081 = ~0x08;
                                        wait(50);
                                             sift_start_data = 0x08;
                                             for( i=1; i<5; i++) {
                                                  sift_start_data = sift_start_data<<1;
                                                  KIBAN081 = ~sift_start_data;
                                                  wait(50);
                                             }

                         } else if( psd_data >= 30 ) {
                                        KIBAN081 = ~0x10;
                                        wait(50);
                                             sift_start_data = 0x10;
                                             for( i=1; i<4; i++) {
                                                  sift_start_data = sift_start_data<<1;
                                                  KIBAN081 = ~sift_start_data;
                                                  wait(50);
                                             }

                         } else if( psd_data >= 26 ) {
                                        KIBAN081 = ~0x20;
                                        wait(50);
                                             sift_start_data = 0x20;
                                             for( i=1; i<3; i++) {
                                                  sift_start_data = sift_start_data<<1;
                                                  KIBAN081 = ~sift_start_data;
                                                  wait(50);
                                             }

                         } else if( psd_data >= 23 ) {
                                        KIBAN081 = ~0x40;
                                        wait(50);
                                             sift_start_data = 0x40;
                                             for( i=1; i<2; i++) {
                                                  sift_start_data = sift_start_data<<1;
                                                  KIBAN081 = ~sift_start_data;
                                                  wait(50);
                                             }

                         } else if( psd_data >= 20 ) {
                                       /*KIBAN081 = ~0x80;    warning  */
                                        w_data = 0x80;
                                        KIBAN081 = ~w_data;
                                        wait(50);
                                             sift_start_data = 0x80;
                                             for( i=1; i<1; i++) {
                                                  sift_start_data = sift_start_data<<1;
                                                  KIBAN081 = ~sift_start_data;
                                                  wait(50);
                                             }

                         } else {
                                        KIBAN081 = ~0x00;
                                        sift_start_data = 0x00;
                         }


           /*---------------------------*/
           /* 割り込みによる表示処理再開 */
           /*---------------------------*/
              start_warikomi();

}


/*------------------------------------------------------------------------*/
/* モジュール名 main                                                      */
```

```
517 /* 処理概要     メイン処理                                                    */
518 /* 引数         なし                                                          */
519 /* 戻り値       なし                                                          */
520 /*--------------------------------------------------------------------------*/
521 int main( void )
522 {
523
524       /*-----------------*/
525       /* H8のポート初期化 */
526       /*-----------------*/
527          Init_Port();
528
529
530       /*-----------------*/
531       /* H8タイマー初期化 */
532       /*-----------------*/
533          Init_H8();
534
535
536       /*---------------------------*/
537       /* オープニング(山形電波固有) */
538       /*---------------------------*/
539          openning08(); /* プログラムスタートの知らせ */
540
541
542       /*-----------------------------------------------------*/
543       /* 課題処理プログラム                                  */
544       /*-----------------------------------------------------*/
545
546            /*-----------------*/
547            /* AD初期化        */
548            /*-----------------*/
549               Init_ad();
550
551            /*-----------------*/
552            /* 表示先設定      */
553            /*-----------------*/
554               SEG_SELECT = LED;               /* 表示先はLED */
555
556            /*-------------------------*/
557            /* 割り込みにて表示スタート */
558            /*-------------------------*/
559               start_warikomi();
560
561
562               while(1) {
563
564                     sift_flag = DIP_SW_LO;
565                     if( sift_flag == 1 ) lsift();
566
567                     start_ad();               /* AD変換スタート     */
568                     end_ad();                 /* AD変換終了チェック */
569                     psd_data = get_ad8();     /* psd読み込み */
570
571               }/* end of while(1) */
572
573 }
```