```
/*------------------------------------------------------------------------*/
/*  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2   物体の位置に相当するＬＥＤを点灯させる。                       */
/*         (トグルスイッチ入力:ブリンク処理あり)                          */
/*                                                                        */
/*   センサーの前に物体を置き、その位置のＬＥＤを点灯させる。次に物体を移動   */
/*   させた時にＬＥＤの点灯位置も移動させる。物体を「８０ｃｍ」以上に遠くに */
/*   離した場合、ＬＥＤ０～７を消灯する。この時７セグメントＬＥＤは常に消灯 */
/*   しておく。                                                           */
/*                                                                        */
/*      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                              */
/*                                                                        */
/*   トグルスイッチからの信号が「Ｌ」ならば常時点灯、「Ｈ」ならば点滅する。 */
/*   点滅は１秒間に３回程度、点灯と消灯を繰り返す程度）                   */
/*                                                                        */
/*   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。                     */
/*   ブリンク処理を考慮し、ＬＥＤ点灯処理は割込処理とした。               */
/*                                                                        */
/*                                  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 int  itu4_count = 0;     /* 10msで+1のカウンタ(iTU0割り込みでカウント) */

  unsigned char brink_flag = 0;     /* 課題2 ブリンク処理 処理する or 処理しない */
  unsigned char hiyoji_flag = 0;    /* 課題2 ブリンク処理するにあたり 点灯 or 消灯 */


/*--------------------------------------------------*/
/* 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;            /*                                      */
```

```
 87                                      /* 0x00 = 0000 0000  全bit入力設定        */
 88          PA.DR.BYTE = 0x00;
 89
 90          PB.DDR = 0xff;              /*                                       */
 91                                      /* 0xff = 1111 1111  全bit出力設定        */
 92          PB.DR.BYTE = 0x00;
 93
 94          P5.DDR = 0xff;              /* 事前配布基板(1) LED or 7segLED指示端子 */
 95                                      /* 0xff = 1111 1111  全bit出力設定        */
 96          P5.DR.BYTE = 0x00;
 97
 98                                      /* 当日作成基板はP7ポート(P7は入力専用なので設定なし) */
 99 }
100
101
102 /*------------------------------------------------------------------------------*/
103 /*  モジュール名 Init_H8                                                        */
104 /*  処理概要     H8 タイマー初期化                                              */
105 /*  引数         なし                                                           */
106 /*  戻り値       なし                                                           */
107 /*------------------------------------------------------------------------------*/
108 /* 参考 H8/3048f(旧cpu) 14.7854MHz      */
109 /*       16ms   29490                   */
110 /*       10ms   18432                   */
111 /*        8ms   14745                   */
112 /*        5ms    9216                   */
113 /*        1ms    1843                   */
114 /*---------------------------------------*/
115 void Init_H8( void )
116 {
117     EI; /* 割り込み許可 */
118
119       /* タイマ設定      */
120          ITU.TMDR.BIT.PWM1 = 0;    /* ITU1:通常動作        */
121          ITU.TMDR.BIT.PWM2 = 0;    /* ITU2:通常動作        */
122          ITU.TMDR.BIT.PWM3 = 1;    /* ITU3:PWMモード       */
123          ITU.TMDR.BIT.PWM4 = 0;    /* ITU4:通常動作        */
124
125
126       /* iTu1 タイマ設定 */
127          ITU1.TCR.BIT.CCLR = 1;    /* カウンタクリア要因                          */
128          ITU1.TCR.BIT.CKEG = 0;    /* クロックエッジ                              */
129          ITU1.TCR.BIT.TPSC = 3;    /* タイマプリスケーラ 1.8432MHz                */
130          ITU1.GRA = 18432;         /* 割り込みインターバル 1.8432MHz/18432=100Hz */
131          ITU1.TIER.BIT.IMIEA = 1;  /* IMFAフラグによる割り込み許可                */
132
133       /* iTu2 タイマ設定 */
134          ITU2.TCR.BIT.CCLR = 1;    /* カウンタクリア要因                          */
135          ITU2.TCR.BIT.CKEG = 0;    /* クロックエッジ                              */
136          ITU2.TCR.BIT.TPSC = 3;    /* タイマプリスケーラ 1.8432MHz                */
137          ITU2.GRA = 18432;         /* 割り込みインターバル 1.8432MHz/18432=100Hz */
138          ITU2.TIER.BIT.IMIEA = 1;  /* IMFAフラグによる割り込み許可                */
139
140       /* iTu3 モータタイマ設定 */
141          ITU3.TCR.BIT.CCLR = 1;    /* カウンタクリア要因                          */
142          ITU3.TCR.BIT.CKEG = 0;    /* クロックエッジ                              */
143          ITU3.TCR.BIT.TPSC = 3;    /* タイマプリスケーラ 1.8432MHz                */
144          ITU3.GRA = PWM_CYCLE;     /* モータPWM周期 1.8432MHz/1843=1kHz           */
145          ITU3.GRB = 0;             /* モータPWM DUTY 0%                           */
146
147       /* タイマ設定 */
148          ITU4.TCR.BIT.CCLR = 1;    /* カウンタクリア要因                          */
149          ITU4.TCR.BIT.CKEG = 0;    /* クロックエッジ                              */
150          ITU4.TCR.BIT.TPSC = 3;    /* タイマプリスケーラ 1.8432MHz                */
151          ITU4.GRA = 18432;         /* 割り込みインターバル 1.8432MHz/18432=100Hz */
152          ITU4.TIER.BIT.IMIEA = 1;  /* IMFAフラグによる割り込み許可                */
153
154
155       /* タイマスタート */
156          ITU.TSTR.BIT.STR0 = 0;    /* まだスタートしない */
157          ITU.TSTR.BIT.STR1 = 0;    /* まだスタートしない */
158          ITU.TSTR.BIT.STR2 = 0;    /* まだスタートしない */
159          ITU.TSTR.BIT.STR3 = 0;    /* まだスタートしない */
160          ITU.TSTR.BIT.STR4 = 0;    /* まだスタートしない */
161 }
162
163
164
165 /*------------------------------------------------------------------------------*/
166 /*  モジュール名  start_itu0                                                    */
167 /*  処理概要      タイマーitu4のスタート                                        */
168 /*  引数          なし                                                          */
169 /*  戻り値        なし                                                          */
170 /*------------------------------------------------------------------------------*/
171 void start_warikomi( void )
172 {
```

```
173         ITU.TSTR.BIT.STR4 = 1; /* itu4スタート */
174 }
175
176
177 /*------------------------------------------------------------------------------*/
178 /*  モジュール名  stop_itu4                                                     */
179 /*  処理概要      タイマーitu4のストップ                                        */
180 /*  引数          なし                                                          */
181 /*  戻り値        なし                                                          */
182 /*------------------------------------------------------------------------------*/
183 void stop_warikomi( void )
184 {
185         ITU.TSTR.BIT.STR4 = 0;  /* itu4ストップ */
186 }
187
188
189 /*------------------------------------------------------------------------------*/
190 /*  モジュール名 課題2用 int_imia4                                              */
191 /*  処理概要                                                                    */
192 /*  引数          なし                                                          */
193 /*  戻り値        なし                                                          */
194 /*------------------------------------------------------------------------------*/
195 void int_imia4( void )
196 {
197     int i;
198     unsigned char  w_data;        /* コンパイラ不良? worning対策 */
199
200          if ( brink_flag == 0 ) {
201            /*--------------------------*/
202            /* 通常点灯                 */
203            /*--------------------------*/
204                       if( psd_data >= 117 ) {
205                                       KIBAN081 = ~0x01;
206
207                       } else if( psd_data >= 80 ) {
208                                       KIBAN081 = ~0x02;
209
210                       } else if( psd_data >= 54 ) {
211                                       KIBAN081 = ~0x04;
212
213                       } else if( psd_data >= 41 ) {
214                                       KIBAN081 = ~0x08;
215
216                       } else if( psd_data >= 30 ) {
217                                       KIBAN081 = ~0x10;
218
219                       } else if( psd_data >= 26 ) {
220                                       KIBAN081 = ~0x20;
221
222                       } else if( psd_data >= 23 ) {
223                                       KIBAN081 = ~0x40;
224
225                       } else if( psd_data >= 20 ) {
226                                    /*KIBAN081 = ~0x80;    warning  */
227                                       w_data = 0x80;
228                                       KIBAN081 = ~w_data;
229
230                       } else {
231                                       KIBAN081 = ~0x00;
232                       }
233
234                       hiyoji_flag = ON;           /* 次は点灯。点滅処理になったらまず消灯 */
235
236
237          } else {
238               /*--------------------------*/
239               /* 点滅点灯                 */
240               /*--------------------------*/
241
242                       if( hiyoji_flag == ON ) {       /* 前回は表示した。∴今回は消灯 */
243                        /*----------------------*/
244                        /* 点滅処理  消灯       */
245                        /*----------------------*/
246                           KIBAN081 = ~0x00;
247
248
249                               itu4_count++;           /* 消灯を30ms続ける */
250                               if( itu4_count == 3 ) {
251                                   hiyoji_flag = OFF;  /* 点滅処理 今回は消灯。次は点灯 */
252                                   itu4_count = 0;
253                               }
254
255
256                       } else {                    /* 前回は消灯した。∴今回は点灯 */
257                       /*----------------------*/
258                       /* 点滅処理 点灯        */
```

```
259                         /*-----------------------*/
260                             if( psd_data >= 117 ) {
261                                     KIBAN081 = ~0x01;
262
263                             } else if( psd_data >= 80 ) {
264                                     KIBAN081 = ~0x02;
265
266                             } else if( psd_data >= 54 ) {
267                                     KIBAN081 = ~0x04;
268
269                             } else if( psd_data >= 41 ) {
270                                     KIBAN081 = ~0x08;
271
272                             } else if( psd_data >= 30 ) {
273                                     KIBAN081 = ~0x10;
274
275                             } else if( psd_data >= 26 ) {
276                                     KIBAN081 = ~0x20;
277
278                             } else if( psd_data >= 23 ) {
279                                     KIBAN081 = ~0x40;
280
281                             } else if( psd_data >= 20 ) {
282                                   /*KIBAN081 = ~0x80;    warning  */
283                                     w_data = 0x80;
284                                     KIBAN081 = ~w_data;
285                             } else {
286                                     KIBAN081 = ~0x00;
287                             }
288
289
290                                  itu4_count++;           /* 点灯を30ms続ける。 */
291                                  if( itu4_count == 3 ) {
292                                      hiyoji_flag = ON;  /* 点滅処理 今回点灯。次は消灯 */
293                                      itu4_count = 0;
294                                  }
295
296
297                         }
298
299
300           }
301
302         /*-------------------------*/
303         /* reset interrupt request */
304         /*-------------------------*/
305           ITU4.TSR.BIT.IMFA = 0;
306 }
307
308
309 /*--------------------------------------------------------------------------*/
310 /*  モジュール名 wait                                                       */
311 /*  処理概要     ソフトウェアタイマー                                       */
312 /*  引数         タイマー値    1:  10[ms]                                   */
313 /*                            10: 100[ms] = 0.1[s]                        */
314 /*                            50: 500[ms] = 0.5[s]                        */
315 /*                           100:1000[ms] = 1.0[s]                        */
316 /*  戻り値       なし                                                       */
317 /*--------------------------------------------------------------------------*/
318 void wait( int iTimer )
319 {
320      int i;
321           while( iTimer ) {
322                 for( i = 0; i < 5000; i++ );    /* H8/3048f */
323               /*for( i = 0; i < 8333; i++ );*/  /* H8/3048f-one */
324                     iTimer--;
325           }
326 }
327
328
329 /*--------------------------------------------------------------------------*/
330 /*  モジュール名  seg_cls                                                   */
331 /*  処理概要      7セグメントLED消去                                        */
332 /*  引数          なし                                                      */
333 /*  戻り値        なし                                                      */
334 /*--------------------------------------------------------------------------*/
335 void seg_cls( void )
336 {
337           KIBAN081= 0xff;   /* KIBAN081 = ~0x00; */
338
339 }
340
341
342 /*--------------------------------------------------------------------------*/
343 /*  モジュール名  openning                                                  */
344 /*  処理概要      オープニング(山形電波工高固有)                            */
```

```
345 /*  引数          なし                                                    */
346 /*  戻り値        なし                                                    */
347 /*--------------------------------------------------------------------------*/
348 void openning08( void )
349 {
350     signed char i;                    /* -128 ～ +127 */
351
352          SEG_SELECT = T7SEG;          /* 表示は7セグメントLED */
353          for( i = 3; i >= 0; i-- ) {
354               KIBAN081= tendo[ i ];   /* 表示 */
355                  wait( 50 );
356               seg_cls();              /* 消去 */
357                  wait( 50 );
358          }
359 }
360
361
362 /*--------------------------------------------------------------------------*/
363 /*  モジュール名 Init_ad                                                    */
364 /*  処理概要     ＡＤ初期化                                                 */
365 /*  引数         なし                                                       */
366 /*  戻り値       なし                                                       */
367 /*--------------------------------------------------------------------------*/
368 void Init_ad( void )
369 {
370       /* ADコントロールレジスタ設定 */
371          AD.CSR.BYTE = 0x00;     /* ch0(AN0 (P70) ) 単一モード */
372         /*AD.CSR.BYTE = 0x01;*/   /* ch1(AN1 (P71) ) 単一モード */
373         /*AD.CSR.BYTE = 0x02;*/   /* ch2(AN2 (P72) ) 単一モード */
374         /*AD.CSR.BYTE = 0x03;*/   /* ch3(AN3 (P73) ) 単一モード */
375         /*AD.CSR.BYTE = 0x04;*/   /* ch4(AN4 (P74) ) 単一モード */
376         /*AD.CSR.BYTE = 0x05;*/   /* ch5(AN5 (P75) ) 単一モード */
377         /*AD.CSR.BYTE = 0x06;*/   /* ch6(AN6 (P76) ) 単一モード */
378         /*AD.CSR.BYTE = 0x07;*/   /* ch7(AN7 (P77) ) 単一モード */
379
380 }
381
382
383 /*--------------------------------------------------------------------------*/
384 /* モジュール名 start_ad                                                    */
385 /* 処理概要     ＡＤ変換スタート                                           */
386 /* 引数         なし                                                       */
387 /* 戻り値       なし                                                       */
388 /*--------------------------------------------------------------------------*/
389 void start_ad( void )
390 {
391          AD.CSR.BIT.ADST = 1; /* AD変換開始 */
392
393 }
394
395
396 /*--------------------------------------------------------------------------*/
397 /* モジュール名 end_ad                                                      */
398 /* 処理概要     ＡＤ変換終了                                               */
399 /* 引数         なし                                                       */
400 /* 戻り値       なし                                                       */
401 /*--------------------------------------------------------------------------*/
402 void end_ad( void )
403 {
404          while( AD.CSR.BIT.ADF != 1 ){}  /* AD変換終了チェック */
405
406 }
407
408
409 /*--------------------------------------------------------------------------*/
410 /* モジュール名 get_ad8                                                     */
411 /* 処理概要     データ読み込み                                             */
412 /*              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) */
413 /*              変換データを確定する。（下位2bitゴミとしてカット）         */
414 /* 引数         なし                                                       */
415 /* 戻り値       8bit data                                                  */
416 /*              1111 1111 11-- ---- : 変換後データ                         */
417 /*              ---- ---- 1111 1111 : 8bitシフト後データ                   */
418 /*                                                                         */
419 /*              ---- ---- 0000 0000 : 0x00      0                        */
420 /*                        :                     :                        */
421 /*              ---- ---- 1111 1111 : 0xff     255  0～255 の 256通りのデータ */
422 /*--------------------------------------------------------------------------*/
423 unsigned char get_ad8( void )
424 {
425     unsigned char indata;        /* unsigned intで 0 ～ 255 まで扱える */
426
427          indata = AD.DRA >>8;    /* 上位10bitを右に8bitシフトする。下位2bit無視 */
428                                  /* 1111 1111 1100 0000           */
429                                  /* ---- ---- 1111 1111           */
430
```

```
431         /*indata = AD.DRB >>8;*/  /* 上位10bitを右に8bitシフトする。*/
432         /*indata = AD.DRC >>8;*/  /* 上位10bitを右に8bitシフトする。*/
433         /*indata = AD.DRD >>8;*/  /* 上位10bitを右に8bitシフトする。*/
434         //indata = indata & 0xff;
435
436           return indata;
437 }
438
439
440 /*--------------------------------------------------------------------------*/
441 /* モジュール名 main                                                        */
442 /* 処理概要     メイン処理                                                  */
443 /* 引数         なし                                                        */
444 /* 戻り値       なし                                                        */
445 /*--------------------------------------------------------------------------*/
446 int main( void )
447 {
448
449       /*-----------------*/
450       /* H8のポート初期化 */
451       /*-----------------*/
452          Init_Port();
453
454
455       /*-----------------*/
456       /* H8タイマー初期化 */
457       /*-----------------*/
458          Init_H8();
459
460
461       /*---------------------------*/
462       /* オープニング(山形電波固有) */
463       /*---------------------------*/
464          openning08(); /* プログラムスタートの知らせ */
465
466
467       /*-----------------------------------------------------*/
468       /* 課題処理プログラム                                  */
469       /*-----------------------------------------------------*/
470
471            /*-----------------*/
472            /* AD初期化        */
473            /*-----------------*/
474               Init_ad();
475
476            /*-----------------*/
477            /* 表示先設定      */
478            /*-----------------*/
479               SEG_SELECT = LED;               /* 表示先はLED */
480
481            /*-------------------------*/
482            /* 割り込みにて表示スタート */
483            /*-------------------------*/
484               start_warikomi();
485
486
487               while(1) {
488
489                     brink_flag = DIP_SW_LO;
490
491                     start_ad();               /* AD変換スタート     */
492                     end_ad();                 /* AD変換終了チェック */
493                     psd_data = get_ad8();     /* psd読み込み */
494
495               }/* end of while(1) */
496
497 }
```