マイクロスイッチセット
C 言語
U ターン走行プログラム
解説マニュアル

第 1.02 版

2015 年 4 月 20 日

株式会社日立ドキュメントソリューションズ

注 意 事 項 (rev.6.0H)

著作権

・本マニュアルに関する著作権は株式会社日立ドキュメントソリューションズに帰属します。
・本マニュアルは著作権法および、国際著作権条約により保護されています。

禁止事項

ユーザーは以下の内容を行うことはできません。
・第三者に対して、本マニュアルを販売、販売を目的とした宣伝、使用、営業、複製などを行うこと
・第三者に対して、本マニュアルの使用権を譲渡または再承諾すること
・本マニュアルの一部または全部を改変、除去すること
・本マニュアルを無許可で翻訳すること
・本マニュアルの内容を使用しての、人命や人体に危害を及ぼす恐れのある用途での使用

転載、複製

本マニュアルの転載、複製については、文書による株式会社日立ドキュメントソリューションズの事前の承諾が必要です。

責任の制限

本マニュアルに記載した情報は、正確を期すため、慎重に制作したものですが万一本マニュアルの記述誤りに起因する損害が生じた場合でも、株式会社日立ドキュメントソリューションズはその責任を負いません。

その他

・本マニュアルに記載の情報は本マニュアル発行時点のものであり、株式会社日立ドキュメントソリューションズは、予告なしに、本マニュアルに記載した情報または仕様を変更することがあります。製作に当たりましては、最新の内容を確認いただきますようお願いします。
・すべての商標および登録商標は、それぞれの所有者に帰属します。

連絡先

株式会社 日立ドキュメントソリューションズ
〒135-0016 東京都江東区東陽六丁目3番2号 イースト21タワー
E-mail：himdx.m-carrally.dd@hitachi.com

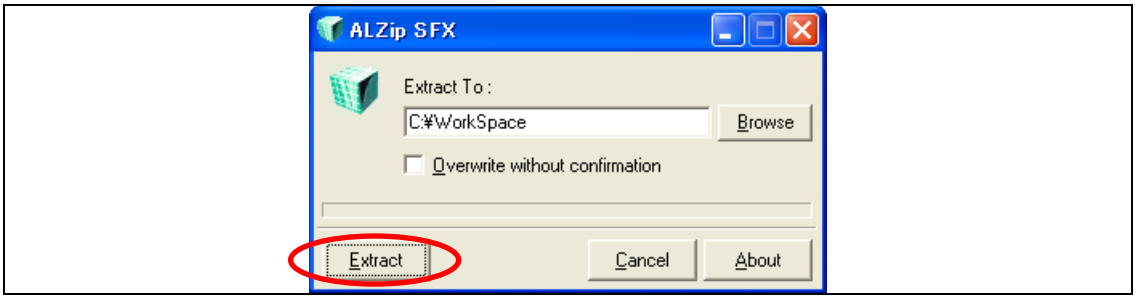
目次

1. 概要

本書では、マイクロスイッチを使用した、C 言語 U ターンプログラムの解説を行います。
開発環境の構築方法やプログラムのビルド、書き込みについては、「ミニマイコンカー製作キット Ver.2　C 言語走行プログラム解説マニュアル」の「3. インストール」「4. ミニマイコンカー Ver.2 の動作確認」を参照してください。

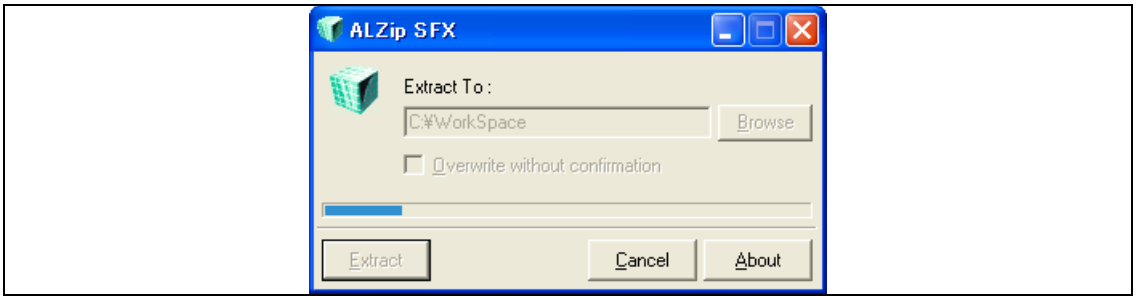
2. ワークスペースのインストール

株式会社日立ドキュメントソリューションズのマイコンカーラリー販売ページからワークスペースのインストーラー「mini_mcr2_microsw_vxxx.exe」（xxx はバージョン）をダウンロードします。
ダウンロードした「mini_mcr2_microsw_vxxx.exe」をダブルクリックし、インストーラーを実行します。

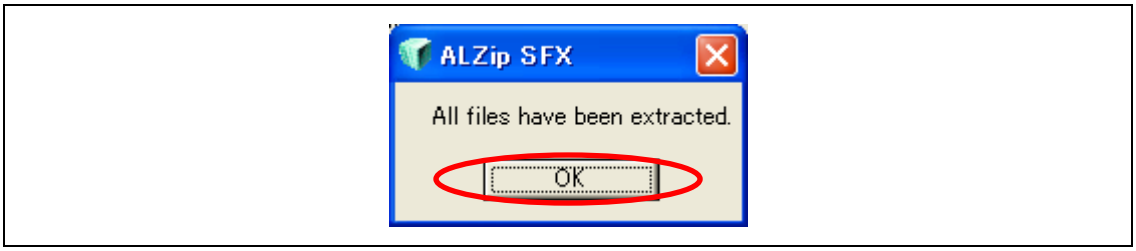


表示されたデフォルトのインストール先のフォルダ「c:¥WorkSpace」を確認して、「Extract」をクリックします。

《**補足**》 別のフォルダを選択する場合は、「Browse」をクリックしてください。



インストールが開始されます。



ワークスペースのインストールが完了しました。「OK」をクリックします。

以上でワークスペースのインストールは完了です。

3. プログラム解説「mini_mcr.c」

WorkSpace のファイルをそのままコンパイルしただけでは、U ターンがうまくいかないことがあります。moter 関数や timer 関数を呼び出すときの引数を調整して、正常に U ターンできるようにしましょう。
DIP スイッチの設定は、以下のようにしてください。

DIP スイッチ（ON：0、OFF：1）			
P5_7（3）	P4_5（2）	P4_4（1）	P4_3（0）
0	0	0	1

3.1 プログラムリスト

```
 1 :  //-----------------------------------------------------------------------------
 2 :  // 対象マイコン R8C/35A
 3 :  // ﾌｧｲﾙ内容     U ターン走行プログラム
 4 :  // バージョン   Ver.1.00
 5 :  // Date         2009.08.25
 6 :  // Copyright    ルネサスマイコンカーラリー事務局
 7 :  //              日立インターメディックス株式会社
 8 :  //-----------------------------------------------------------------------------
 9 :  //-----------------------------------------------------------------------------
10 :  // インクルード
11 :  //-----------------------------------------------------------------------------
12 :  #include "sfr_r835a.h"
13 :
14 :  //-----------------------------------------------------------------------------
15 :  // シンボル定義
16 :  //-----------------------------------------------------------------------------
17 :  #define TIMER_CYCLE     155             // 1ms:0.001/(1/(20000000/128))-1
18 :  #define PWM_CYCLE       39999           // 16ms:0.016/(1/(20000000/8))-1
19 :
20 :  #define Def_500Hz       4999            // 500Hz:(1/500)/(1/(20000000/8))-1
21 :  #define Def_1000Hz      2499            // 1000Hz:(1/1000)/(1/(20000000/8))-1
22 :
23 :  #define Def_C3          19083           // ド:(1/131)/(1/(20000000/8))-1
24 :  #define Def_D3          17006           // レ:(1/147)/(1/(20000000/8))-1
25 :  #define Def_E3          15151           // ミ:(1/165)/(1/(20000000/8))-1
26 :  #define Def_F3          14285           // ファ:(1/175)/(1/(20000000/8))-1
27 :  #define Def_G3          12754           // ソ:(1/196)/(1/(20000000/8))-1
28 :  #define Def_A3          11362           // ラ:(1/220)/(1/(20000000/8))-1
29 :  #define Def_B3          10120           // シ:(1/247)/(1/(20000000/8))-1
30 :  #define Def_C4          9541            // ド:(1/262)/(1/(20000000/8))-1
31 :
32 :  #define DI()            asm("FCLR I")   // 割り込み禁止
33 :  #define EI()            asm("FSET I")   // 割り込み許可
34 :
35 :  //-----------------------------------------------------------------------------
36 :  // 関数プロトタイプの宣言
37 :  //-----------------------------------------------------------------------------
38 :  void init( void );
```

```
 39 :  unsigned char sensor( void );
 40 :  void motor( int data1, int data2 );
 41 :  void timer( unsigned long timer_set );
 42 :  void beep( int data1 );
 43 :  unsigned char dipsw( void );
 44 :  unsigned char pushsw( void );
 45 :  unsigned char microsw( void );
 46 :
 47 :  //-----------------------------------------------------------------------------
 48 :  // グローバル変数の宣言
 49 :  //-----------------------------------------------------------------------------
 50 :  unsigned long   cnt0 = 0;               // timer 関数用
 51 :  unsigned long   cnt1 = 0;               // main 内で使用
 52 :  int             pattern = 0;            // パターン番号
 53 :
 54 :  //-----------------------------------------------------------------------------
 55 :  // メインプログラム
 56 :  //-----------------------------------------------------------------------------
 57 :  void main(void)
 58 :  {
 59 :          // 初期化
 60 :          init();
 61 :
 62 :          // 起動音
 63 :          beep(Def_500Hz);
 64 :          timer(100);
 65 :          beep(Def_1000Hz);
 66 :          timer(100);
 67 :          beep(0);
 68 :
 69 :          while( pushsw() == 0 ){
 70 :          }
 71 :
 72 :          beep(Def_1000Hz);
 73 :          timer(1000);
 74 :          beep(0);
 75 :
 76 :
 77 :          while(1){
 78 :
 79 :                  if( ( sensor() & 0x04 ) == 0x04 ){
 80 :                          // 右から 3 番目のセンサーの反応があった場合
 81 :                          motor( 0, 50 );
 82 :                  }else{
 83 :                          // 右から 3 番目のセンサーの反応がなかった場合
 84 :                          motor( 50, 0 );
 85 :                  }
 86 :
 87 :                  if( microsw() == 1 ){
 88 :                  // マイクロスイッチが押された場合
 89 :
 90 :                          motor( -50, -50 );
 91 :                          timer(1000);
 92 :
 93 :                          motor( -50, 0 );
 94 :                          timer(3000);
 95 :
 96 :                          while( ( sensor() & 0x04 ) == 0x00 ){
 97 :                          }
 98 :
 99 :                  }
100 :          }
101 :  }
```

```
102 :
103 : //-----------------------------------------------------------------------
104 : // R8C/35A の内蔵周辺機能の初期化
105 : //-----------------------------------------------------------------------
106 : void init( void )
107 : {
108 :         unsigned char i = 0;
109 :
110 :         // 割り込み禁止
111 :         DI();
112 :
113 :         // クロック発生回路の XIN クロック設定
114 :         prc0 = 1;
115 :
116 :         cm13 = 1;
117 :         cm05 = 0;
118 :         while(i <= 50) i++;
119 :         ocd2 = 0;
120 :
121 :         prc0 = 0;
122 :
123 :         // I/O ポートの入出力設定
124 :         prc2 = 1;                       // pd0 レジスタへの書き込み許可
125 :         pd0 = 0xe0;                     // P0_0～P0_3:センサー
126 :                                         // P0_4:マイクロスイッチ
127 :                                         // P0_5～P0_7:LED
128 :         prc2 = 0;                       // pd0 レジスタへの書き込み禁止
129 :         pd1 = 0xdf;                     // P1_0～P1_3:LED
130 :                                         // P1_4:TXD0
131 :                                         // P1_5:RXD0
132 :         pd2 = 0xfe;                     // P2_0:スイッチ
133 :                                         // P2_1:AIN1
134 :                                         // P2_2:PWMA
135 :                                         // P2_3:BIN1
136 :                                         // P2_4:PWMB
137 :                                         // P2_5:SERVO
138 :                                         // P2_6:AIN2
139 :                                         // P2_7:BIN2
140 :         pd3 = 0xfb;                     // P3_2:赤外線受信
141 :                                         // P3_4:ブザー
142 :         pd4 = 0x80;                     // P4_2:VREF
143 :                                         // P4_3～P4_5:DIPSW
144 :                                         // P4_6:XIN
145 :                                         // P4_7:XOUT
146 :         pd5 = 0x40;                     // P5_7:DIPSW
147 :         pd6 = 0xff;                     //
148 :
149 :
150 :
151 :         mstcr = 0x00;                   // モジュールストップ解除
152 :
153 :
154 :
155 :         // タイマ RB の 1ms 割り込み設定
156 :         trbmr = 0x00;                   // カウントソースは f1
157 :         trbpre = 128 - 1;               // プリスケーラ
158 :         trbpr = TIMER_CYCLE;            // プライマリカウンタ
159 :         trbic = 0x01;                   // タイマ RB の割り込みレベル設定
160 :         trbcr = 0x01;                   // カウントを開始
161 :
162 :         // タイマ RC の PWM モード
163 :         trccr1 = 0xb0;                  // カウントソースは f8
164 :         trcgra = 0;                     // 圧電サウンダの周期
```

```
165 :          trcgrc = 0;                     // 圧電サウンダのデューティ比
166 :          trccr2 = 0x02;                  // TRCIOC 端子はアクティブレベル H
167 :          trcoer = 0x0b;                  // TRCIOC 端子の出力許可
168 :          trcpsr1 = 0x02;                 // TRCIOC 端子を P3_4 に割り当て
169 :          trcmr = 0x8a;                   // カウントを開始
170 :
171 :          // タイマ RD のリセット同期 PWM モード
172 :          trdpsr0 = 0x08;                 // TRDIOB0 端子を P2_2 に割り当て
173 :          trdpsr1 = 0x05;                 // TRDIOB1 端子を P2_5 に割り当て
174 :                                          // TRDIOA1 端子を P2_4 に割り当て
175 :          trdmr = 0xf0;                   // レジスタをバッファ動作にする
176 :          trdfcr = 0x01;                  // リセット同期 PWM モードに設定
177 :          trdoer1 = 0xcd;                 // TRDIOB1 の出力許可
178 :                                          // TRDIOA1 の出力許可
179 :                                          // TRDIOB0 端子の出力許可
180 :          trdcr0 = 0x23;                  // カウントソースは f8
181 :          trdgra0 = trdgrc0 = PWM_CYCLE;  // 周期
182 :          trdgrb0 = trdgrd0 = 0;          // TRDIOB0 端子（左モータ）
183 :          trdgra1 = trdgrc1 = 0;          // TRDIOA1 端子（右モータ）
184 :          trdgrb1 = trdgrd1 = 0;          // TRDIOB1 端子（サーボ）
185 :          trdstr = 0x0d;                  // カウントを開始
186 :
187 :          // 割り込み許可
188 :          EI();
189 :  }
190 :
191 :  //------------------------------------------------------------------------------
192 :  // 割り込み
193 :  //------------------------------------------------------------------------------
194 :  #pragma interrupt intTRBIC (vect=24)
195 :  void intTRBIC( void )
196 :  {
197 :          p0_7 = ~p0_7;
198 :
199 :          if( p0_7 == 0 ){
200 :                  //p0_1、p0_3 のモニタが可能
201 :                  p0_5 = ~p0_1;
202 :                  p0_6 = ~p0_3;
203 :          }else{
204 :                  //p0_0、p0_2 のモニタが可能
205 :                  p0_5 = p0_0;
206 :                  p0_6 = p0_2;
207 :          }
208 :
209 :          cnt0++;
210 :          cnt1++;
211 :  }
212 :
213 :  //------------------------------------------------------------------------------
214 :  // センサー状態検出
215 :  // 引数          なし
216 :  // 戻り値        センサ値
217 :  //------------------------------------------------------------------------------
218 :  unsigned char sensor( void )
219 :  {
220 :          volatile unsigned char  data1;
221 :
222 :          data1 = ~p0;                    // ラインの色は白
223 :          data1 = data1 & 0x0f;
224 :
225 :          return( data1 );
226 :  }
227 :
```

```
228 :  //---------------------------------------------------------------------------
229 :  // モーター速度制御
230 :  // 引数         左モータ:-100～100、右モータ:-100～100
231 :  //              0 で停止、100 で正転 100%、-100 で逆転 100%
232 :  // 戻り値       なし
233 :  //---------------------------------------------------------------------------
234 :  void motor( int data1, int data2 )
235 :  {
236 :          volatile int    motor_r;
237 :          volatile int    motor_l;
238 :          volatile int    sw_data;
239 :
240 :          sw_data = dipsw() + 5;
241 :          motor_l = (long)data1 * sw_data / 20;
242 :          motor_r = (long)data2 * sw_data / 20;
243 :
244 :          if( motor_l >= 0 ) {
245 :                  p2_1 = 0;
246 :                  p2_6 = 1;
247 :                  trdgrd0 = (long)( PWM_CYCLE - 1 ) * motor_l / 100;
248 :          } else {
249 :                  p2_1 = 1;
250 :                  p2_6 = 0;
251 :                  trdgrd0 = (long)( PWM_CYCLE - 1 ) * ( -motor_l ) / 100;
252 :          }
253 :
254 :          if( motor_r >= 0 ) {
255 :                  p2_3 = 0;
256 :                  p2_7 = 1;
257 :                  trdgrc1 = (long)( PWM_CYCLE - 1 ) * motor_r / 100;
258 :          } else {
259 :                  p2_3 = 1;
260 :                  p2_7 = 0;
261 :                  trdgrc1 = (long)( PWM_CYCLE - 1 ) * ( -motor_r ) / 100;
262 :          }
263 :  }
264 :
265 :  //---------------------------------------------------------------------------
266 :  // 時間稼ぎ
267 :  // 引数         タイマ値 1=1ms
268 :  // 戻り値       なし
269 :  //---------------------------------------------------------------------------
270 :  void timer( unsigned long data1 )
271 :  {
272 :          cnt0 = 0;
273 :          while( cnt0 < data1 );
274 :  }
275 :
276 :  //---------------------------------------------------------------------------
277 :  // 音を鳴らす
278 :  // 引数         (1/音の周波数)/(1/(クロック周波数/8))-1
279 :  // 戻り値       なし
280 :  //---------------------------------------------------------------------------
281 :  void beep( int data1 )
282 :  {
283 :          trcgra = data1;                 // 周期の設定
284 :          trcgrc = data1 / 2;             // デューティ 50%のため周期の半分の値
285 :  }
286 :
287 :  //---------------------------------------------------------------------------
288 :  // DIP スイッチ状態検出
289 :  // 引数         なし
290 :  // 戻り値       0～15、DIP スイッチが ON の場合、対応するビットが 0 になります。
```

```
291 :  //------------------------------------------------------------------------------
292 :  unsigned char dipsw( void )
293 :  {
294 :          volatile unsigned char  data1;
295 :
296 :          data1 = ( ( p5 >> 4 ) & 0x08 ) | ( ( p4 >> 3 ) & 0x07 );
297 :
298 :          return( data1 );
299 :  }
300 :
301 :  //------------------------------------------------------------------------------
302 :  // プッシュスイッチ状態検出
303 :  // 引数          なし
304 :  // 戻り値        スイッチが押されていない場合:0、押された場合:1
305 :  //------------------------------------------------------------------------------
306 :  unsigned char pushsw( void )
307 :  {
308 :          unsigned char data1;
309 :
310 :          data1 = ~p2;
311 :          data1 &= 0x01;
312 :
313 :          return( data1 );
314 :  }
315 :
316 :  //------------------------------------------------------------------------------
317 :  // マイクロスイッチ状態検出
318 :  // 引数          なし
319 :  // 戻り値        スイッチが押されていない場合:0、押された場合:1
320 :  //------------------------------------------------------------------------------
321 :  unsigned char microsw( void )
322 :  {
323 :          unsigned char data1;
324 :
325 :          data1 = ~( p0 >> 4 );
326 :          data1 &= 0x01;
327 :
328 :          return( data1 );
329 :  }
```

3.2　メインプログラムを説明する前に

main 関数は、main 関数の後に記載されている関数を組み合わせてプログラムしていますので、先に main 関数以外の関数の解説を初めに行います。

microsw 以外の関数については「ミニマイコンカー製作キット Ver.2　C 言語走行プログラム解説マニュアル」の「5. プログラム解説「mini_mcr.c」」を参照してください。

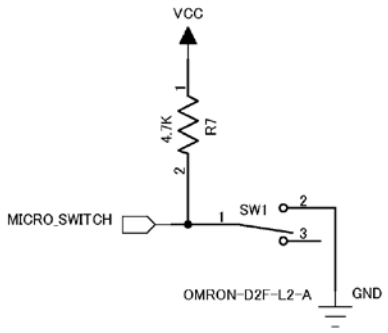
3.3 マイクロスイッチ状態検出：microsw 関数

microsw 関数は、マイクロスイッチが OFF のときに“0”、ON のときに“1”の値を返します。

プログラム

```
321 :  unsigned char microsw( void )
322 :  {
323 :          unsigned char data1;
324 :
325 :          data1 = ~( p0 >> 4 );
326 :          data1 &= 0x01;
327 :
328 :          return( data1 );
329 :  }
```

回路図




```
325 :          data1 = ~( p0 >> 4 );
```

P0 レジスタを読み込み、4 ビット右シフトして、反転します。マイクロスイッチは、P0 の端子につながっていますので、P0 レジスタを読み込むことにより、状態を検出できます。マイクロスイッチを押した場合 GND とショート状態になり、P0 の端子は L になります。マイクロスイッチを押した場合に“1”にしたいので、4 ビット右シフトして、反転をします。

```
326 :          data1 &= 0x01;
```

マスクをかけます。P0 レジスタを読み込む場合、8 ビット単位で読み込まれます。プッシュスイッチは、ポート 0 の 4 の端子にしかつながっていませんので、4 ビット右シフトした P0 レジスタの 1～7 ビットには必要のない値が入っています。そこで、0x01 と AND をとることにより、1～7 ビットを“0”にします。

```
328 :          return( data1 );
```

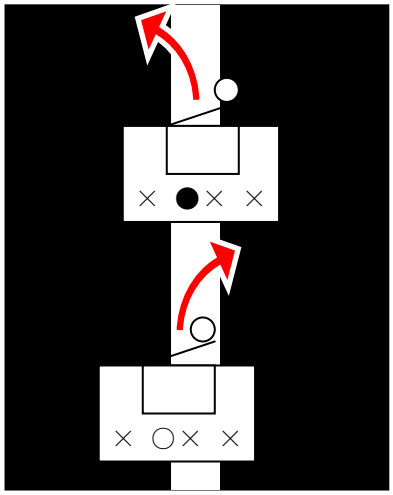
関数の呼び出し元に値を返します。

3.4 メインプログラム：main 関数

main 関数は、スタートアップルーチンから呼び出され、最初に実行される C 言語のプログラムです。

```
 57 :  void main(void)
 58 :  {
 59 :          // 初期化
 60 :          init();
 61 :
 62 :          // 起動音
 63 :          beep(Def_500Hz);
 64 :          timer(100);
 65 :          beep(Def_1000Hz);
 66 :          timer(100);
 67 :          beep(0);
 68 :
 69 :          while( pushsw() == 0 ){
 70 :          }
 71 :
 72 :          beep(Def_1000Hz);
 73 :          timer(1000);
 74 :          beep(0);
 75 :
 76 :
 77 :          while(1){
 78 :
 79 :                  if( ( sensor() & 0x04 ) == 0x04 ){
 80 :                          // 右から 3 番目のセンサーの反応があった場合
 81 :                          motor( 0, 50 );
 82 :                  }else{
 83 :                          // 右から 3 番目のセンサーの反応がなかった場合
 84 :                          motor( 50, 0 );
 85 :                  }
 86 :
 87 :                  if( microsw() == 1 ){
 88 :                  // マイクロスイッチが押された場合
 89 :
 90 :                          motor( -50, -50 );
 91 :                          timer(1000);
 92 :
 93 :                          motor( -50, 0 );
 94 :                          timer(3000);
 95 :
 96 :                          while( ( sensor() & 0x04 ) == 0x00 ){
 97 :                          }
 98 :
 99 :                  }
100 :          }
101 :  }
```

```
79 :                    if( ( sensor() & 0x04 ) == 0x04 ){
80 :                            // 右から 3 番目のセンサーの反応があった場合
81 :                            motor( 0, 50 );
82 :                    }else{
83 :                            // 右から 3 番目のセンサーの反応がなかった場合
84 :                            motor( 50, 0 );
85 :                    }
```

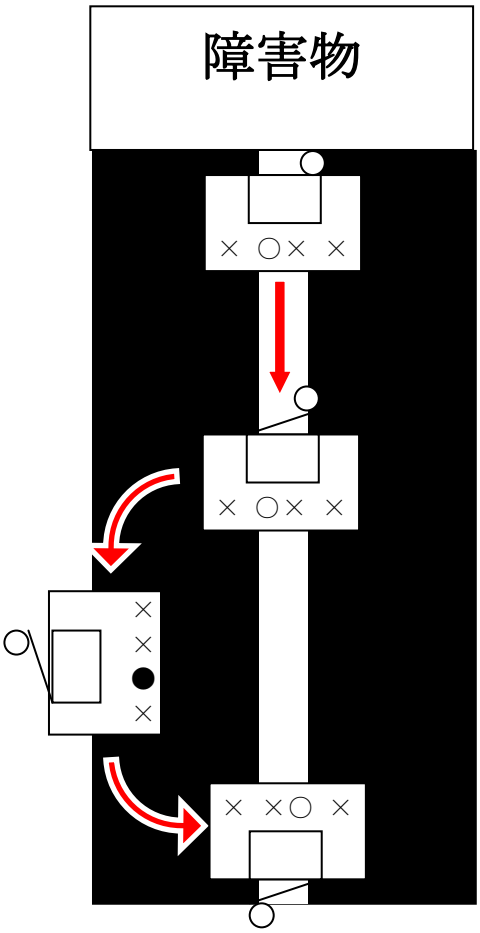


右から 3 番目のセンサーの反応があった場合、右のモーターを前進させて左旋回します。

右から 3 番目のセンサーの反応がなかった場合、左のモーターを前進させて右旋回します。

この処理が連続して繰り返されるので、白と黒の境目をジグザグ走行します。

```
87 :                    if( microsw() == 1 ){
88 :                    // マイクロスイッチが押された場合
89 :
90 :                            motor( -50, -50 );
91 :                            timer(1000);
92 :
93 :                            motor( -50, 0 );
94 :                            timer(3000);
95 :
96 :                            while( ( sensor() & 0x04 ) == 0x00 ){
97 :                            }
98 :
99 :                    }
```



障害物に当たってマイクロスイッチが押された場合、1 秒間後進します。

1 秒間後進したあとは、左モーターを 3 秒間後進させて左後ろ旋回します。

右から 3 番目のセンサーが反応していない場合は、左後ろ旋回を続けます。

右から 3 番目のセンサーが反応した場合は、ジグザグ走行の処理に戻ります。