マイコンプログラミング

学籍番号:22120
組番号:222
名前: 塚田 勇人

2025 年 2 月 5 日

表 1: 実験環境

OS	Windows 11 Pro
CPU	AMD Ryzen 7 5800H
メモリ	16GB
コンパイラ	SH C/C++ Compiler
エディタ	HEW2

1 目的

MEMES ボードを用いて，簡単なゲームを作ることで，マイコンプログラミングの基礎を学ぶことを本課題の目的とする．

2 実験環境

3 プログラムの設計と説明

ここでは，本課題で追加したプログラムの仕様とその説明を行う．追加した仕様は以下の通りである．

- ゲームのスコアを 7 セグメント LED に表示する
- ゲーム開始時にスタート音を鳴らす
- 岩に触れたときにミス音を鳴らす
- 自機をジョイスティックで左右に移動できるようにする
- ゲームを一時停止，再開，リセットできるようにする

3.1 スコア表示

スコア表示は 7 セグメント LED を用いて行う．グローバル変数 score にスコアを格納し，`do_7seg()` 関数で 7 セグメント LED に表示する．`do_7seg()` 関数のソースコードをリスト 1 に示す．この関数をゲームのメインループ内で呼び出すことで，スコアを表示する．スコアは岩が左端に到達するごとに 1 点加算される．自機が岩に触れるとスコアが 5 点減少する．

リスト 1: do_7seg()

```
1  void do_7seg() {
2    static int keta;
3    int num1 = score % 10;
4    int num2 = (score / 10) % 10;
5    int num3 = score / 100;
```

```
6       PA.DR.BYTE.HL &= 0xf0;
7       DIG1 = DIG2 = DIG3 = 0;
8
9       if (keta == 0) {
10          PA.DR.BYTE.HL |= num1;
11          DIG1 = 1;
12          keta++;
13      } else if (keta == 1) {
14          PA.DR.BYTE.HL |= num2;
15          if (num2 == 0 && num3 == 0) {
16              DIG2 = 0;
17          } else {
18              DIG2 = 1;
19          }
20          keta++;
21      } else if (keta == 2) {
22          PA.DR.BYTE.HL |= num3;
23          if (num3 == 0) {
24              DIG2 = 0;
25          } else {
26              DIG3 = 1;
27          }
28          keta = 0;
29      }
30  }
```

3.2 スタート音

スタート音は startBeep() 関数で鳴らす．startBeep() 関数のソースコードをリスト 2 に示す．この関数では，MTU2 ch3 を用いて音を鳴らす．周期 $20MHz/5000 = 4000$ で用い，4kHz の音を 0.1 秒間で 2 回ならしている．

リスト 2: startBeep()

```
1   void startBeep() {
2     int i = 0;
3
4     // MTU2 ch3 sound
5     MTU23.TCR.BIT.TPSC = 0;
6     MTU23.TCR.BIT.CCLR = 1;
7     MTU23.TGRA = 5000 - 1;
8     MTU2.TSTR.BIT.CST3 = 1;
```

```
9
10      while (i < 100) {
11          if (MTU23.TSR.BIT.TGFA = 0) {
12              SPK ^= 1;
13              i++;
14          }
15      }
16      wait_us(100000);
17      i = 0;
18      while (i < 100) {
19          if (MTU23.TSR.BIT.TGFA = 0) {
20              SPK ^= 1;
21              i++;
22          }
23      }
24      MTU2.TSTR.BIT.CST3 = 0;
25  }
```

3.3 ミス音

ミス音は missBeep() 関数で鳴らす．missBeep() 関数のソースコードをリスト 3 に示す．この関数では，MTU2 ch3 を用いて音を鳴らす．周期 $20MHz/10000 = 2000$ で用い，2kHz の音を 0.1 秒間で 100 回ならしている．

リスト 3: missBeep()

```
1   void missBeep() {
2     int i = 0;
3
4     // MTU2 ch3 sound
5     MTU23.TCR.BIT.TPSC = 0;
6     MTU23.TCR.BIT.CCLR = 1;
7     MTU23.TGRA = 10000 - 1;
8     MTU2.TSTR.BIT.CST3 = 1;
9
10    while (i < 100) {
11        if (MTU23.TSR.BIT.TGFA = 0) {
12            SPK ^= 1;
13            i++;
14        }
15    }
16    MTU2.TSTR.BIT.CST3 = 0;
```

```
17 }
```

3.4 自機の移動

自機の移動は move_me() 関数で行う．move_me() 関数のソースコードをリスト 4 に示す．この関数では，ジョイスティックを横に動かすと自機も左右に移動するようにする．また，岩に触れたときにミス音を鳴らし，スコアを減らすようにしている．

リスト 4: move_me()

```
1  void move_me(struct position *me, struct position rock[]) {
2    int i;
3    struct position old_position;
4
5    old_position.x = me->x;
6    old_position.y = me->y;
7    me->active = 1;
8
9    if (AD0.ADDR0 < 0x4000) {
10       // -- ジョイスティック上--
11       me->y = 0;
12   } else if (AD0.ADDR0 > 0xc000) {
13       // -- ジョイスティック下--
14       me->y = 1;
15   }
16   if (AD0.ADDR1 > 0xc000 && me->x > 0) {
17       me->x--;
18   } else if (AD0.ADDR1 < 0x4000 && me->x < 15) {
19       me->x++;
20   }
21
22   if (old_position.y != me->y || old_position.x != me->x) {
23       // -- 移動したとき.. 古い表示を消す--
24       LCD_cursor(old_position.x, old_position.y);
25       LCD_putch(' ');
26   }
27   for (i = 0; i < NMROF_ROCKS; i++) {
28       if (rock[i].active) {
29           if ((old_position.x == rock[i].x && old_position.y == rock[i].y)
                 ||
30               (me->x == rock[i].x &&
31                me->y == rock[i].y)) { // プレイヤーに触れたら岩を消す
```

```
32                  score -= 5;
33                  if (score < 0) score = 0;
34                  rock[i].active = 0;
35                  LCD_cursor(rock[i].x, rock[i].y);
36                  LCD_putch(' ');
37                  me->active = 0;
38                  missBeep();
39                  break;
40              }
41          }
42      }
43      if (me->active) {
44          LCD_cursor(me->x, me->y);
45          LCD_putch('>');
46      } else {
47          LCD_cursor(me->x, me->y);
48          LCD_putch(' ');
49      }
50  }
```

4 プログラム

プログラムをリスト 5 に示す.

リスト 5: main.c

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

2  /* FILE :memes_final.c */
3  /*****************************************************************************/

4  #include <stdlib.h>
5
6  #include "iodefine.h"
7  #include "typedefine.h"
8
9  #define printf ((int (*)(const char *, ...))0x00007c7c)
10
11 #define SW6 (PD.DR.BIT.B18)
12 #define SW5 (PD.DR.BIT.B17)
13 #define SW4 (PD.DR.BIT.B16)
14
```

```
15 #define LED6 (PE.DR.BIT.B11)
16 #define LED_ON (0)
17 #define LED_OFF (1)
18
19 #define DIG1 (PE.DR.BIT.B3)
20 #define DIG2 (PE.DR.BIT.B2)
21 #define DIG3 (PE.DR.BIT.B1)
22
23 #define SPK (PE.DR.BIT.B0)
24
25 #define LCD_RS (PA.DR.BIT.B22)
26 #define LCD_E (PA.DR.BIT.B23)
27 #define LCD_RW (PD.DR.BIT.B23)
28 #define LCD_DATA (PD.DR.BYTE.HH)
29
30 #define NMROF_ROCKS 6
31
32 enum STS { STOP, RUN, PAUSE1, PAUSE2, PAUSE3 };
33
34 int score;
35
36 struct position {
37     int x;
38     int y;
39     int active;
40 };
41
42 void wait_us(_UINT);
43 void LCD_inst(_SBYTE);
44 void LCD_data(_SBYTE);
45 void LCD_cursor(_UINT, _UINT);
46 void LCD_putch(_SBYTE);
47 void LCD_putstr(_SBYTE *);
48 void LCD_cls(void);
49 void LCD_init(void);
50 void missBeep(void);
51 void startBeep(void);
52
53 // --------------------
54 // -- 使用する関数群--
55 // --------------------
56 void wait_us(_UINT us) {
```

```
57      _UINT val;
58
59      val = us * 10 / 16;
60      if (val >= 0xffff) val = 0xffff;
61
62      CMT0.CMCOR = val;
63      CMT0.CMCSR.BIT.CMF &= 0;
64      CMT.CMSTR.BIT.STR0 = 1;
65      while (!CMT0.CMCSR.BIT.CMF);
66      CMT0.CMCSR.BIT.CMF = 0;
67      CMT.CMSTR.BIT.STR0 = 0;
68  }
69
70  void LCD_inst(_SBYTE inst) {
71      LCD_E = 0;
72      LCD_RS = 0;
73      LCD_RW = 0;
74      LCD_E = 1;
75      LCD_DATA = inst;
76      wait_us(1);
77      LCD_E = 0;
78      wait_us(40);
79  }
80
81  void LCD_data(_SBYTE data) {
82      LCD_E = 0;
83      LCD_RS = 1;
84      LCD_RW = 0;
85      LCD_E = 1;
86      LCD_DATA = data;
87      wait_us(1);
88      LCD_E = 0;
89      wait_us(40);
90  }
91
92  void LCD_cursor(_UINT x, _UINT y) {
93      if (x > 15) x = 15;
94      if (y > 1) y = 1;
95      LCD_inst(0x80 | x | y << 6);
96  }
97
98  void LCD_putch(_SBYTE ch) { LCD_data(ch); }
```

```
99
100 void LCD_putstr(_SBYTE *str) {
101     _SBYTE ch;
102
103     while (ch = *str++) LCD_putch(ch);
104 }
105
106 void LCD_cls(void) {
107     LCD_inst(0x01);
108     wait_us(1640);
109 }
110
111 void LCD_init(void) {
112     wait_us(45000);
113     LCD_inst(0x30);
114     wait_us(4100);
115     LCD_inst(0x30);
116     wait_us(100);
117     LCD_inst(0x30);
118
119     LCD_inst(0x38);
120     LCD_inst(0x08);
121     LCD_inst(0x01);
122     wait_us(1640);
123     LCD_inst(0x06);
124     LCD_inst(0x0c);
125 }
126
127 // ------------------------------------------
128 // -- ゲーム用の関数群--
129
130 // -- 自分を移動--
131 void move_me(struct position *me, struct position rock[]) {
132     int i;
133     struct position old_position;
134
135     old_position.x = me->x;
136     old_position.y = me->y;
137     me->active = 1;
138
139     if (AD0.ADDR0 < 0x4000) {
140         // -- ジョイスティック上--
```

```
141         me->y = 0;
142     } else if (AD0.ADDR0 > 0xc000) {
143         // -- ジョイスティック下--
144         me->y = 1;
145     }
146     if (AD0.ADDR1 > 0xc000 && me->x > 0) {
147         me->x--;
148     } else if (AD0.ADDR1 < 0x4000 && me->x < 15) {
149         me->x++;
150     }
151 
152     if (old_position.y != me->y || old_position.x != me->x) {
153         // -- 移動したとき.. 古い表示を消す--
154         LCD_cursor(old_position.x, old_position.y);
155         LCD_putch(' ');
156     }
157     for (i = 0; i < NMROF_ROCKS; i++) {
158         if (rock[i].active) {
159             if ((old_position.x == rock[i].x && old_position.y == rock[i].y)
                    ||
160                 (me->x == rock[i].x &&
161                  me->y == rock[i].y)) { // プレイヤーに触れたら岩を消す
162                 score -= 5;
163                 if (score < 0) score = 0;
164                 rock[i].active = 0;
165                 LCD_cursor(rock[i].x, rock[i].y);
166                 LCD_putch(' ');
167                 me->active = 0;
168                 missBeep();
169                 break;
170             }
171         }
172     }
173     if (me->active) {
174         LCD_cursor(me->x, me->y);
175         LCD_putch('>');
176     } else {
177         LCD_cursor(me->x, me->y);
178         LCD_putch(' ');
179     }
180 }
181 
```

```
182 // -- 岩を移動--
183 void move_rock(struct position rock[]) {
184     int i;
185 
186     for (i = 0; i < NMROF_ROCKS; i++) {
187         if (rock[i].active) {
188             // 画面上に岩が存在する
189             LCD_cursor(rock[i].x, rock[i].y);
190             LCD_putch(' ');
191             if (rock[i].x == 0) {
192                 // 消去
193                 score++;
194                 rock[i].active = 0;
195             } else {
196                 rock[i].x--;
197                 LCD_cursor(rock[i].x, rock[i].y);
198                 LCD_putch('*');
199             }
200         }
201     }
202 }
203 
204 // -- 新しい岩を作る--
205 void new_rock(struct position rock[]) {
206     int i;
207 
208     for (i = 0; i < NMROF_ROCKS; i++) {
209         if (rock[i].active == 0) {
210             // -- 新しい岩--
211             rock[i].active = 1;
212             rock[i].x = 15;
213             rock[i].y = rand() % 2;
214             LCD_cursor(rock[i].x, rock[i].y);
215             LCD_putch('*');
216             break;
217         }
218     }
219 }
220 
221 void do_7seg() {
222     static int keta;
223     int num1 = score % 10;
```

```
224     int num2 = (score / 10) % 10;
225     int num3 = score / 100;
226     PA.DR.BYTE.HL &= 0xf0;
227     DIG1 = DIG2 = DIG3 = 0;
228 
229     if (keta == 0) {
230         PA.DR.BYTE.HL |= num1;
231         DIG1 = 1;
232         keta++;
233     } else if (keta == 1) {
234         PA.DR.BYTE.HL |= num2;
235         if (num2 == 0 && num3 == 0) {
236             DIG2 = 0;
237         } else {
238             DIG2 = 1;
239         }
240         keta++;
241     } else if (keta == 2) {
242         PA.DR.BYTE.HL |= num3;
243         if (num3 == 0) {
244             DIG2 = 0;
245         } else {
246             DIG3 = 1;
247         }
248         keta = 0;
249     }
250 }
251 
252 void missBeep() {
253     int i = 0;
254 
255     // MTU2 ch3 sound
256     MTU23.TCR.BIT.TPSC = 0;
257     MTU23.TCR.BIT.CCLR = 1;
258     MTU23.TGRA = 10000 - 1;
259     MTU2.TSTR.BIT.CST3 = 1;
260 
261     while (i < 100) {
262         if (MTU23.TSR.BIT.TGFA = 0) {
263             SPK ^= 1;
264             i++;
265         }
```

```
266     }
267     MTU2.TSTR.BIT.CST3 = 0;
268 }
269
270 void startBeep() {
271     int i = 0;
272
273     // MTU2 ch3 sound
274     MTU23.TCR.BIT.TPSC = 0;
275     MTU23.TCR.BIT.CCLR = 1;
276     MTU23.TGRA = 5000 - 1;
277     MTU2.TSTR.BIT.CST3 = 1;
278
279     while (i < 100) {
280         if (MTU23.TSR.BIT.TGFA = 0) {
281             SPK ^= 1;
282             i++;
283         }
284     }
285     wait_us(100000);
286     i = 0;
287     while (i < 100) {
288         if (MTU23.TSR.BIT.TGFA = 0) {
289             SPK ^= 1;
290             i++;
291         }
292     }
293     MTU2.TSTR.BIT.CST3 = 0;
294 }
295
296 // ------------------------------------------
297 // -- メイン関数--
298 void main() {
299     struct position me; // 自分の車の座標
300     struct position rock[NMROF_ROCKS]; // 岩の座標
301     char strPrint[16];
302     int move_timing, new_timing;
303     int ad, i;
304     int stop_sw, run_sw, pause_sw;
305     int status;
306
307     score = 0;
```

```
308
309     STB.CR4.BIT._AD0 = 0;
310     STB.CR4.BIT._CMT = 0;
311     STB.CR4.BIT._MTU2 = 0;
312
313     CMT0.CMCSR.BIT.CKS = 1;
314
315     // MTU2 ch0
316     MTU20.TCR.BIT.TPSC = 3; // 選択 1/64
317     MTU20.TCR.BIT.CCLR = 1; // のコンペアマッチでクリア TGRA
318     MTU20.TGRA = 31250 - 1; // 100ms
319     MTU20.TIER.BIT.TTGE = 1; // A/変換開始要求を許可 D
320
321     // AD0
322     AD0.ADCSR.BIT.ADM = 3; // シングルモード
323     AD0.ADCSR.BIT.CH = 1; // AN0
324     AD0.ADCSR.BIT.TRGE = 1; // からのトリガ有効 MTU2
325     AD0.ADTSR.BIT.TRG0S = 1; // コンペアマッチでトリガ TGRA
326
327     // MTU2 ch1
328     MTU21.TCR.BIT.TPSC = 3; // 選択 1/64
329     MTU21.TCR.BIT.CCLR = 1; // のコンペアマッチでクリア TGRA
330     MTU21.TGRA = 31250 - 1; // 100ms
331
332     // MTU2 ch2 7seg
333     MTU22.TCR.BIT.TPSC = 3;
334     MTU22.TCR.BIT.CCLR = 1;
335     MTU22.TGRA = 1000 - 1;
336
337     LCD_init();
338
339     MTU2.TSTR.BIT.CST0 = 1; // MTU2 スタート CH0
340     MTU2.TSTR.BIT.CST1 = 1; // MTU2 スタート CH1
341     MTU2.TSTR.BIT.CST2 = 1; // MTU2 スタート CH2
342
343     PFC.PAIORH.BYTE.L |= 0x0F;
344     PFC.PEIORL.BIT.B1 = 1;
345     PFC.PEIORL.BIT.B2 = 1;
346     PFC.PEIORL.BIT.B3 = 1;
347
348     me.x = me.y = 0;
349     for (i = 0; i < NMROF_ROCKS; i++) rock[i].active = 0;
```

```
350
351     status = STOP;
352     move_timing = new_timing = 0;
353     while (1) {
354         if (MTU21.TSR.BIT.TGFA) {
355             MTU21.TSR.BIT.TGFA = 0;
356
357             // 100に回、スイッチを読む ms1
358             stop_sw = SW4;
359             pause_sw = SW5;
360             run_sw = SW6;
361
362             if (status == RUN) {
363                 // ゲーム中
364                 move_me(&me, rock); // 自分移動
365                 if (move_timing++ >= 2) {
366                     move_timing = 0;
367                     move_rock(rock); // 岩を移動
368                     if (new_timing-- <= 0) {
369                         new_timing = rand() * 2 / (RAND_MAX + 1) + 1;
370                         new_rock(rock); // 新しい岩が出現
371                     }
372                 }
373                 if (pause_sw) {
374                     status = PAUSE1;
375                     LCD_cls();
376                     LCD_cursor(0, 0);
377                     LCD_putstr("PAUSE");
378                     LCD_cursor(0, 1);
379                     LCD_putstr("PUSH switch5");
380                 }
381                 if (stop_sw) {
382                     status = STOP;
383                 }
384             } else if (status == PAUSE1) {
385                 if (!pause_sw) // pause_sw がOFF なら
386                     status = PAUSE2; // status をPAUSE2 へ
387             } else if (status == PAUSE2) {
388                 if (pause_sw) // pause_sw がON
389                     status = PAUSE3; // status をPAUSE3 へ
390             } else if (status == PAUSE3) {
391                 if (!pause_sw){ // pause_sw がOFF なら
```

```
392                     startBeep();
393                 LCD_cls();
394                                 }
395                 status = RUN; // sutatus をRUN へ
396             } else { // がのとき statusSTOP
397                 LCD_cls();
398                 LCD_cursor(0, 0);
399                 LCD_putstr("MEMES GAMES");
400                 LCD_cursor(0, 1);
401                 LCD_putstr("PUSH switch6");
402                 // 停止中
403                 if (run_sw) {
404                     startBeep();
405                     status = RUN;
406                     LCD_cls();
407                 }
408             }
409         }
410         // if (@@@@@) としてセグ用のタイマフラグ7 を見るようにするとよい
411         if (MTU22.TSR.BIT.TGFA) {
412             MTU22.TSR.BIT.TGFA = 0;
413             do_7seg();
414         }
415     }
416 }
```

4.1 一時停止，再開，リセット

一時停止, 再開, リセットはSW4, SW5, SW6 を用いて行う．プログラムを始めるとMEME6S GAMES Push switch6 と表示される．

5 実行結果

6 考察