1　ソースコード

ソースコード 1 に作成したコードを示す。

Listing 1　課題

```
#include "iodefine.h"
#include "typedefine.h"

#define printf ((int (*)(const char *,...))0x00007c7c)
#define SW6 (PD.DR.BIT.B18)
#define SW4 (PD.DR.BIT.B16)
#define LCD_RS (PA.DR.BIT.B22)
#define LCD_E (PA.DR.BIT.B23)
#define LCD_RW (PD.DR.BIT.B23)
#define LCD_DATA (PD.DR.BYTE.HH)

void main();
void wait_us(_UINT);
void LCD_inst(_SBYTE);
void LCD_data(_SBYTE);
void LCD_cursor(_UINT, _UINT);
void LCD_putch(_SBYTE);
void LCD_putstr(_SBYTE *);
void LCD_cls();
void LCD_init();

void main() {
    /* 変数宣言エリア */
    _SBYTE  addr0_str[17] = {'␣', '␣', 'A', 'D', 'D', 'R', '0', '␣', '=', '␣', '0', '0', '0', '0', '␣', '␣', '\0'};
    _SBYTE  addr1_str[17] = {'␣', '␣', 'A', 'D', 'D', 'R', '1', '␣', '=', '␣', '0', '0', '0', '0', '␣', '␣', '\0'};
    _UINT isStopping = 1;
    _UINT i;

    /* モジュールスタンバイ解除エリア */
    STB.CR4.BIT._AD0 = 0;
    STB.CR4.BIT._CMT = 0;

    /* 変換器設定エリアAD */
    AD0.ADCSR.BIT.ADM = 3;  // チャネルスキャンモードに設定2
    AD0.ADCSR.BIT.ADCS = 1;  // 連続スキャンモードに設定
    AD0.ADCSR.BIT.CH = 1;  // スキャンするチャネルを設定
    AD0.ADCR.BIT.ADST = 1;  // スキャンの開始

    /* 設定エリアCMT1 */
    CMT1.CMCSR.BIT.CKS = 3;  // 分周比の設定
    CMT1.CMCOR = 19531 - 1;  // カウント回数の設定
    CMT.CMSTR.BIT.STR1 = 1;  // カウントの開始

    LCD_init();  // の初期化LCD

    while (1) {
        /* スイッチにより停止状態と動作状態を切り替える */
        if (SW6) isStopping = 0;
        if (SW4) isStopping = 1;

        /* 状態によって処理を切り替える */
        if (isStopping) continue;  // 停止状態ならば処理をしない
        if (!AD0.ADCSR.BIT.ADF) continue;  // スキャンの途中ならば処理をしない
        if (!CMT1.CMCSR.BIT.CMF) continue;  // カウントの途中ならば処理をしない

        /* 各フラグを折る */
        AD0.ADCSR.BIT.ADF = 0;
        CMT1.CMCSR.BIT.CMF = 0;

        /* 表示する文字列の作成 */
        addr0_str[10] = '0' + ((AD0.ADDR0 >> 6) / 1000 % 10);
        addr0_str[11] = '0' + ((AD0.ADDR0 >> 6) / 100 % 10);
        addr0_str[12] = '0' + ((AD0.ADDR0 >> 6) / 10 % 10);
        addr0_str[13] = '0' + ((AD0.ADDR0 >> 6) % 10);
        addr1_str[10] = '0' + ((AD0.ADDR1 >> 6) / 1000 % 10);
        addr1_str[11] = '0' + ((AD0.ADDR1 >> 6) / 100 % 10);
```

```
67          addr1_str[12] = '0' + ((AD0.ADDR1 >> 6) / 10 % 10);
68          addr1_str[13] = '0' + ((AD0.ADDR1 >> 6) % 10);
69
70          /* 文字列の表示 */
71          LCD_cursor(0, 0);
72          LCD_putstr(addr0_str);
73          LCD_cursor(0, 1);
74          LCD_putstr(addr1_str);
75      }
76  }
77
78  void wait_us(_UINT us) {
79      _UINT val;
80
81      val = us * 10 / 16;
82      if (val >= 0xffff)
83          val = 0xffff;
84
85      CMT0.CMCSR.BIT.CKS = 1;
86      CMT0.CMCOR = val;
87      CMT0.CMCSR.BIT.CMF &= 0;
88      CMT.CMSTR.BIT.STR0 = 1;
89      while (!CMT0.CMCSR.BIT.CMF);
90      CMT0.CMCSR.BIT.CMF = 0;
91      CMT.CMSTR.BIT.STR0 = 0;
92  }
93
94  void LCD_inst(_SBYTE inst) {
95      LCD_E = 0;
96      LCD_RS = 0;
97      LCD_RW = 0;
98      LCD_E = 1;
99      LCD_DATA = inst;
100     wait_us(1);
101     LCD_E = 0;
102     wait_us(40);
103 }
104
105 void LCD_data(_SBYTE data) {
106     LCD_E = 0;
107     LCD_RS = 1;
108     LCD_RW = 0;
109     LCD_E = 1;
110     LCD_DATA = data;
111     wait_us(1);
112     LCD_E = 0;
113     wait_us(40);
114 }
115
116 void LCD_cursor(_UINT x, _UINT y) {
117     if (x > 15)
118         x = 15;
119     if (y > 1)
120         y = 1;
121     LCD_inst(0x80 | x | y << 6);
122 }
123
124 void LCD_putch(_SBYTE ch) {
125     LCD_data(ch);
126 }
127
128 void LCD_putstr(_SBYTE *str) {
129     _SBYTE ch;
130
131     while (ch = *str++)
132         LCD_putch(ch);
133 }
134
135 void LCD_cls() {
136     LCD_inst(0x01);
137     wait_us(1640);
138 }
139
140 void LCD_init() {
141     wait_us(45000);
142     LCD_inst(0x30);
143     wait_us(4100);
144     LCD_inst(0x30);
145     wait_us(100);
```

```
146     LCD_inst(0x30);
147
148     LCD_inst(0x38);
149     LCD_inst(0x08);
150     LCD_inst(0x01);
151     wait_us(1640);
152     LCD_inst(0x06);
153     LCD_inst(0x0c);
154 }
```

2　工夫した点

ソースコード 1 にて工夫した点を次にまとめる。

- continue 文の活用により、while 文内のネストを殲滅した。
- 状態遷移の考え方を導入することにより、わかりやすいコードを書けた。
- 表示の更新を行うまでの 0.5 秒間に押されるスイッチの検知を可能とした。