実技課題／解答及び解説

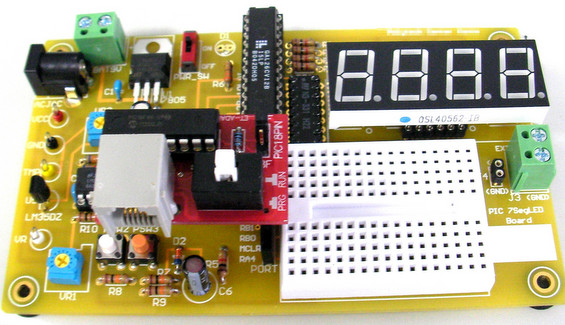
実技課題／　解答及び解説

　「Ｃ言語を用いたマイコンによる計測制御」

１．機器の仕様

**仕様**

|  |  |  |
| --- | --- | --- |
| 入出力機器 | 機能 | 設定例(PIC) |
| 押しボタンスイッチ1 | 計測開始 | RA2（負論理） |
| 押しボタンスイッチ2 | ブザー出力停止 | RA3（負論理） |
| センサ入力 | 計測信号入力 | RA1(A/D変換)  温度センサ、100mV/℃ |
| 表示器 | 計測値出力 | PORTB(RB0～RB7)  7セグメントLED4桁 |
| ブザー | 出力 | RA4（矩形波による駆動可） |



２．ソースファイル

　課題の解答は各処理機能を複数のファイルに分割している。基本はmain関数を実装しているmain.cのファイルのみを編集して完成するものとする。

|  |  |  |
| --- | --- | --- |
| ファイル名 | 機能 | 備考 |
| main.c | main関数、割込み関数、各機能の初期化等 | このファイルのみ編集 |
| common.h | ソースファイル間の共通設定 |  |
| display.c | 表示器の関数の実装 |  |
| display.h | 表示器の関数の定義 |  |
| ad.c | A/D変換の関数の実装 |  |
| ad.h | A/D変換の関数の定義 |  |

|  |
| --- |
| /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ファイル名:main.c  概要:温度センサの値取得と表示  ターゲット:PIC16F88(20MHz)  ポートA:アナログ入力(RA0, RA1)、入力(RA2, RA3)  ポートB:LED(RB0-RB7)  作成日:2012/07/20  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  //各種ファイルのインクルード  #include "common.h"  #include "ad.h"  #include "display.h"  //コンフィグレーションビットの設定  \_\_CONFIG(FOSC\_HS & WDTE\_OFF & PWRTE\_OFF & MCLRE\_ON & BOREN\_OFF &  LVP\_OFF & CPD\_OFF & WRT\_OFF & CCPMX\_RB3 & CP\_OFF & DEBUG\_OFF);  \_\_CONFIG(FCMEN\_OFF & IESO\_OFF);  #define BEEP\_CYCLE 0xB5 //ブザー音の半周期(7オクターブのドの音階)  #define DISP\_CYCLE 0xCF2C  #define DISP\_LOW (unsigned char)(DISP\_CYCLE & 0xFF)  #define DISP\_HIGH (unsigned char)(DISP\_CYCLE >> 8)  //スイッチの置換(負論理で接続)  #define PSW1 RA2 //押しボタンスイッチ1はRA2に接続  #define PSW2 RA3 //押しボタンスイッチ1はRA3に接続  #define SW\_ON 0 //押しボタンスイッチON  #define SW\_OFF 1 //押しボタンスイッチOFF  //プロトタイプ宣言  void interrupt isr(void);  void io\_init(void);  void tmr0\_init(void);  void tmr1\_init(void);  unsigned short move\_average(unsigned short add);  //グローバル変数・定数  unsigned char seg[4];  #define AVERAGE\_COUNT 3 //移動平均の数3 = 2の3乗 = 8個の平均を取る  //main関数  void main(void)  {  unsigned short temp; //温度の値(25.6℃ => 2560)  unsigned short ave; //移動平均値  unsigned char i;  io\_init(); //I/Oの初期化  tmr0\_init(); //タイマ0の初期化  tmr1\_init(); //タイマ1の初期化  ad\_init(); //A/Dの初期化  //割込みの設定  TMR0IE = 0; //TMR0割込み禁止(後の処理で許可)  PEIE = 1; //周辺機能割込み許可  GIE = 1; //全割込み許可  while (1) { //無限ループ  dec2temp(seg, 0);  i = 50;  //スイッチ1がOFFのとき以下のループ  while (PSW2 == SW\_OFF) { //入力待ち専用表示(0000を点滅)  while (i--) seg\_disp4(seg);  SEG\_OFF();  \_\_delay\_ms(400);  continue;  }  TMR1IF = 0; //フラグ初期化  TMR1IE = 1; //タイマ0割込み許可  while (1) {  //温度センサをA/D変換して100mV/℃として代入  temp = (unsigned short long)ad\_get(1) \* 5000 >> 10;  ave = move\_average(temp); //移動平均値の算出  dec2temp(seg, ave); //"25.6c"のように表示  //温度が既定値(30℃)以上のとき  if (ave >= (ad\_get(0) << 2)) break;  \_\_delay\_ms(20); //A/D変換のサンプリング周期(適当)  }  //ブザー出力  TMR0IF = 0; //フラグ初期化  TMR0IE = 1; //タイマ0割込み許可T  //スイッチ2がONになるまでループ  while(PSW2 == SW\_OFF) ;  TMR0IE = 0; //タイマ0割込み禁止  TMR1IF = 0; //フラグ初期化  TMR1IE = 0; //タイマ0割込み許可  }  }  //I/O初期化関数  void io\_init(void)  {  ANSEL = 0x03; //RA0とRA1をアナログ入力、他をディジタルI/O  TRISA = 0x0F; //RA0-RA3を入力、他を出力  TRISB = 0x00; //PORTBを全出力  PORTB = 0x00; //7セグは全てOFF(消灯)  }  //タイマ0初期化関数  void tmr0\_init(void)  {  T0CS = 0; //システムクロックをカウント  PSA = 0; //プリスケーラを使用する  PS2 = 0; //プリスケーラ1/16  PS1 = 1;  PS0 = 1;  TMR0 = BEEP\_CYCLE; //タイマの初期値  }  //タイマ1初期化関数  void tmr1\_init(void)  {  T1CKPS0 = 0; //プリスケーラ1:1  T1CKPS1 = 0;  T1OSCEN = 0; //外部オシレータパワーオフ  TMR1CS = 0; //Fosc/4(0.2us)を使用  TMR1ON = 1; //タイマ有効  TMR1L = DISP\_LOW; //65536 - 12500 = 53036 = 0xCF2C  TMR1H = DISP\_HIGH; //0.2us x 12500 = 2.5ms  }  //割込み関数  void interrupt isr(void)  {  static unsigned char i = 0; //桁の切替用静的変数  //タイマ0割込み(ブザー音)  if (TMR0IF & TMR0IE) { //TMR0フラグ検出  RA4 = !RA4; //ブザーの信号を反転  TMR0 = BEEP\_CYCLE;  TMR0IF = 0; //TMR0フラグ初期化  }  //タイマ1割込み(2ms毎に発生)  if (TMR1IF & TMR1IE) { //TMR1フラグ検出  if (i++ >= 3) { //3桁表示済かを検出  i = 0; //初期化  }  SEG\_OFF(); //消灯  if (i == 2) { //ドットポイントの有無  SEG\_ON\_DP(i, seg[i]); //ドットポイント含めて表示  }  else {  SEG\_ON(i, seg[i]); //ドットポイントなしで表示  }  TMR1L = DISP\_LOW; //65536 - 12500 = 53036 = 0xCF2C  TMR1H = DISP\_HIGH; //0.2us x 12500 = 2.5ms  TMR1IF = 0; //TMR1フラグ初期化  }  }  //移動平均用の関数(初回呼び出し時は配列内の各要素は0)  //この関数は動作中にテストできないのでパソコン上で動作する環境での確認となる。  unsigned short move\_average(unsigned short add)  {  static unsigned char index; //入替対象の要素番号  static unsigned short sum, array[1 << AVERAGE\_COUNT];    sum -= array[index]; //最古のデータを合計から削除  array[index] = add; //最新のデータを最古の場所に追加  sum += array[index]; //最新のデータを合計値に追加    index++; //入れ替える要素番号を+1  //平均数が8のとき index = index & (0x07)と等価  index &= (1 << AVERAGE\_COUNT) - 1; //要素番号の範囲を超えないようにマスク  return sum >> AVERAGE\_COUNT; //合計値から平均個数を除算して平均値  } |