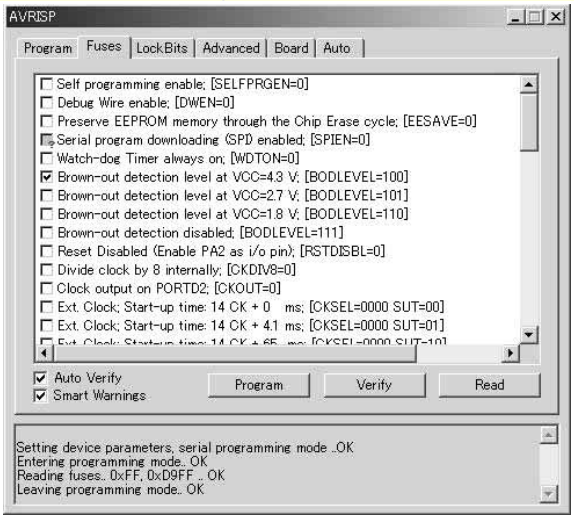

図2-18　AVRISPのヒューズ・ビット設定画面
AVRISPなどのプログラマを使用すれば，ロック・ビットやヒューズ・ビットのビット・パターンを数値で計算しなくても操作ソフトの設定ウィンドウから簡単に操作できる．

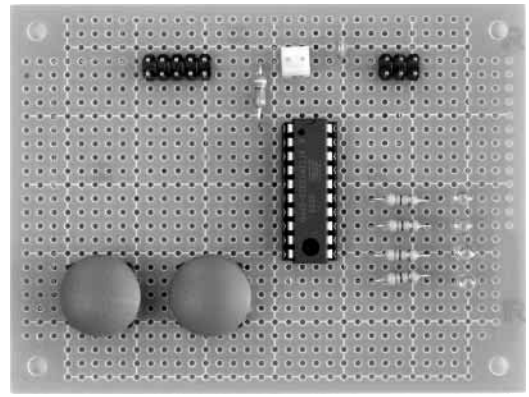

写真2-2　製作したATtiny2313テスト・ボード
スイッチ入力2個，LED出力2個．ISP用コネクタ，拡張コネクタの二つのコネクタを設置．

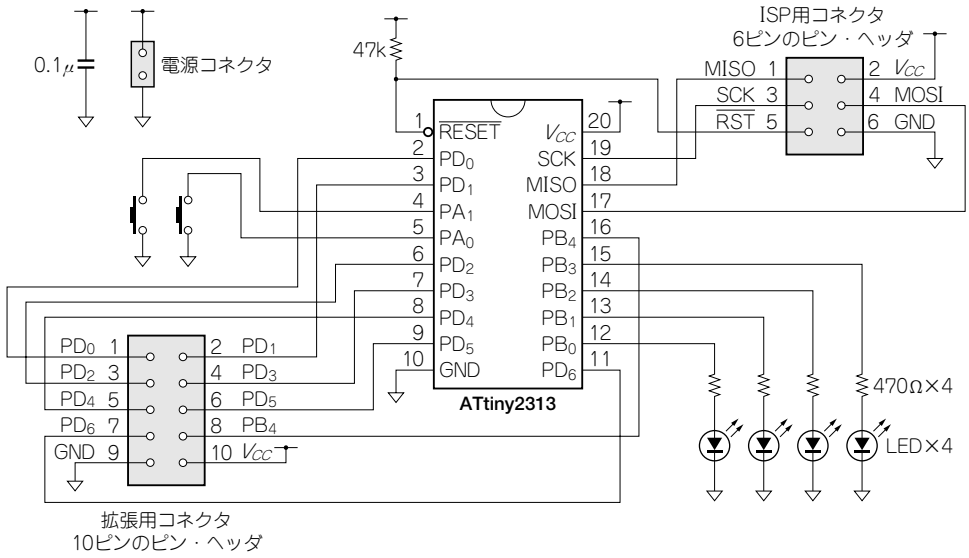

図2-19　ATtiny2313テスト・ボードの回路図
スイッチ入力2個，LED出力2個のシンプルな構成にする．残りのピンをピン・ヘッダに配線しておけば，後からさまざまな機能のテストが可能になる．

● アセンブラ版テスト・プログラム（LED点滅，スイッチの状態をLEDにエコーする）

それでは，アセンブラ言語で記述したシンプルな動作テスト・プログラムを走らせてみましょう．LEDを10回点滅させた後，スイッチの状態をLEDにエコーする動作を行うプログラムを**リスト2-1**に示します．**リスト2-1**はアトメル社から無償でダウンロードできる“AVR Studio”に付属するアセン

ブラの仕様に合わせた記述方法で書いています．

リスト2-1では，初めにAVRのチップの種類に対応したインクルード・ファイルを設定します．インクルード・ファイルには，レジスタのアドレスなど必要な定義がされています．このインクルード・ファイルにより，チップによって異なるI/OアドレスをAVR共通のシンボルとして記述することができます．**リスト2-1**中のSPLやDDRB，PORTAなどにチップ固有の値が割り当てられます．

プログラム本体では，初めにいくつかの設定を行います．まずスタック・ポインタの設定を行います．そのあと，使用するI/Oポートの入出力の設定を行います．その他の設定は，初期状態のままでも二つのI/Oポートをもつコントローラとして利用するための準備が整います．これらの操作には慣習的に

リスト2-1　アセンブラ版動作テスト・プログラム（prg0201.asm）

```
;
;prg0201.asm tiny2313シンプル・ボード用テスト・プログラム
;
.include "tn2313def.inc"

.org        0x0                                  ← 割り込みは使用しないのでベクトル省略
reset:      LDI     R16,LOW(RAMEND)              ← スタック・ポインタの設定
            OUT     SPL,R16

            LDI     R16,0b00001111               ← ポートB0～B3を出力に設定
            OUT     DDRB,R16

            ldi     R16,0b00000011               ← ポートA0，A1をプルアップ
            OUT     PORTA,R16

            LDI     R17,10                       ← 10回点滅

loop:       LDI     R16,0b00001111               ← 点灯
            OUT     PORTB,R16

            RCALL   wait05                       ← 0.5秒待ち

            LDI     R16,0b00000000               ← 消灯
            OUT     PORTB,R16

            RCALL   wait05                       ← 0.5秒待ち

            DEC     R17
            BRNE    loop

loop2:      IN      R16,PINA                     ← スイッチ入力
            COM     R16                          ← 反転
            ANDI    R16,0b00000011
            OUT     PORTB,R16                    ← LEDへエコー
            RJMP    loop2
```

R16レジスタを使用します．なぜR0でなくR16かというと，AVRのアセンブラ命令で直値を扱うもの，たとえば“LDI”や“ANDI”は，操作対象のレジスタがR16からR31までという制限があるからです．

続いて，プログラムは10回の点滅のループに入ります．レジスタのデクリメント（内容をマイナスする）を行う“DEC”の後にゼロ・フラグによる条件分岐を行う“BRNE”を置くという定番のスタイルになります．点滅ループの後，プログラムはスイッチからの入力を行いLEDへ出力する無限ループとなります．

今回のテスト・ボードのATtiny2313の場合，入力側のPAポートにはグラウンドに向かってスイッチが配線されているので，スイッチを押すと“L”の値を得ることになり，その結果をLEDに反映する

時間待ち　0.5秒/1MHzクロック

```
wait05:     LDI     R18,5

wait052:    RCALL   wait01
            DEC     R18
            BRNE    wait052

            RET
```

時間待ち　0.1秒/1MHzクロック

```
wait01:     LDI     R19,100
wait012:    RCALL   wait1m
            DEC     R19
            BRNE    wait012

            RET
```

時間待ち　約1ミリ秒/1MHzクロック

```
wait1m:     LDI     R20,250
wait1m2:    NOP
            DEC     R20
            BRNE    wait1m2

            RET
```

4クロックで1ループ

prg0201.hex

```
:020000020000FC
:100000000FED0DBF0FE007BB03E00BBB1AE00FE0E5
:1000100008BB0AD000E008BB07D01A95C1F709B3A6
:10002000095037008BBFBCF25E003D02A95E9F7C4
:10003000089534E603D03A95E9F708954AEF0000B1
:060040004A95E9F708955E
:00000001FF
```

ために反転処理を行います．

この後に，ループで使用した時間待ちのサブルーチンを三つ置きます．ATtiny2313のヒューズ・ビットの初期値は，内部8 MHz＋8分周の設定になっています．**リスト2-1**ではその設定をそのまま使用することにするので，1 MHzのクロック動作で，それぞれほぼ1 ms，100 ms，0.5秒の時間待ちです．

先にもふれましたが，ロック・ビット，ヒューズ・ビットの設定ですが，**リスト2-1**を動作させる場合は，とくに書き換えをしなくてもデフォルト値で動作します．デフォルトではメモリ保護，デバッグ，ウォッチドッグ・タイマ，低電圧検出は無効，RESETピン有効，内部8 MHzクロック，クロック8分周，起動時間最長の設定になっています．フラッシュ・メモリのプログラムはそのままにして，AVRISPなどのプログラマからクロック設定のヒューズ・ビットを書き換える，たとえば，クロック8分周を無効にする，などを行えば，LEDの点滅速度がそれに応じて変化します．

ここで注意する点は，一度クロックを外部クロックに設定して再起動すると，その後のヒューズ・ビットの書き換えに際しても外部クロックが必要になる点です．ですから，このテスト・ボードのようにセラロックなどの外部クロック用の回路が設置されていない動作回路においてヒューズ・ビットを「外部クロックを使用する」設定にしてしまうと，リセット後はもうその動作環境ではどうすることもできなくなります．

アセンブルを行った後のライタによる書き込みですが，**図2-20**に示すように，専用のダイアログ・

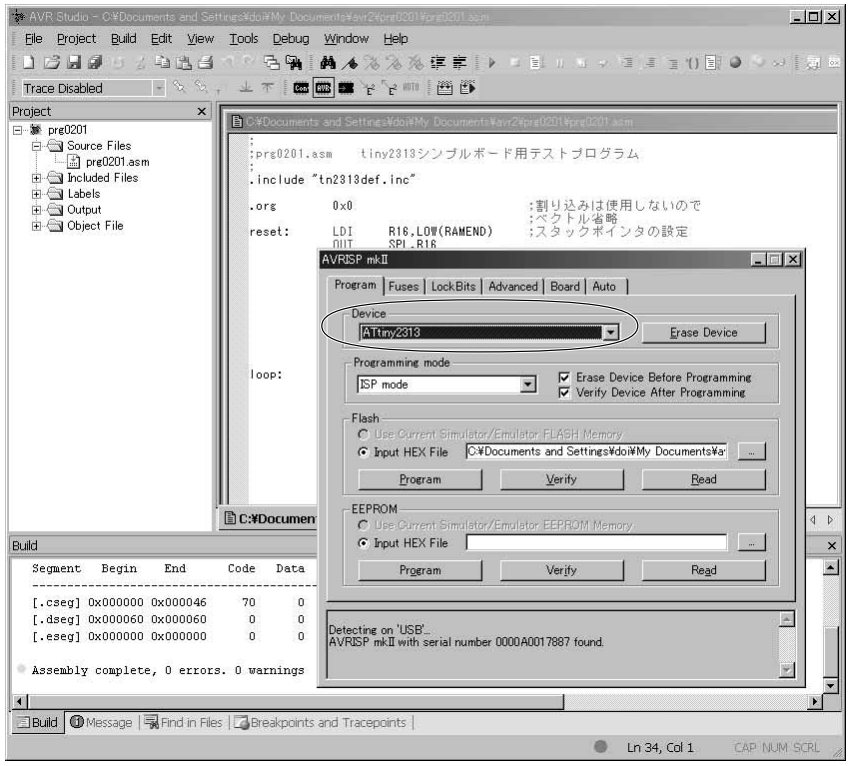

図2-20　ライタ操作のダイアログ・ボックス
「Device」欄で使用しているチップの設定を行った後に書き込み．