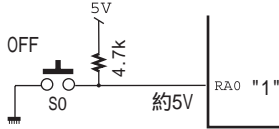


【前回の内容の確認】

- (1) プログラム作成のとき、まず目標の確認と必要なメモリの割り当てを行った。次にメモリ内のデータをどう変えるか、プログラムの設計を行った。
- (2) 図1の実験回路に対し、2進数でカウントアップし、プッシュオンスイッチS0をオンにすることにより表示値をゼロクリアするプログラムを作成した。

回路動作の再確認

- 1) S0が開いている場合、RA0端子には、5Vと4.7kを通じて、約5Vの電圧値が加わっている。プログラムでこの電圧を読み取ると、このビット位置に"1"が入る。



- 2) S0が閉じている場合、RA0端子には約0Vの電圧値が加わっている。プログラムでこの電圧を読み取ると、このビット位置に"0"が入る。

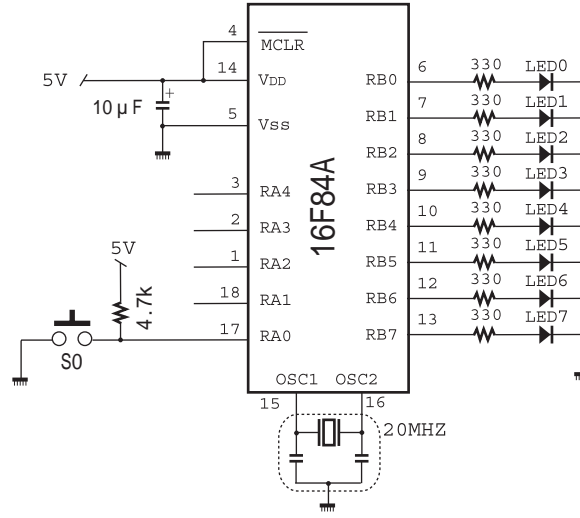
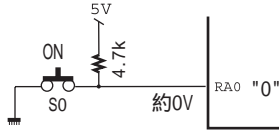


図1 実験回路図

プログラム動作の再確認 (前回作成した内容のものとは少し違っている)

;カウントアップとゼロクリア動作

```
include 16f84.h
.osc hs
.pwrt on
.wdt off
.protect off
```

SAMPLE.ASMと同じ内容。
PIC名称、セラミック振動子使用、電源リセット可、ウォッチドッグ
タイムは不使用、コード保護無し。

```
org 0ch
time ds 1 ..... カウントアップ回数を記憶するメモリ
tm1 ds 1 .....
tm2 ds 1 .....
tm3 ds 1 ..... むだ時間発生用メモリ
```

```
org 0
goto start ..... 電源オン時にstart番地へ飛ぶ
```

```
start mov !ra,#1fh ..... RA0~RA4は入力用となる。RA0にはスイッチが繋がっている。
mov !rb,#0 ..... PB0~RB7は出力用となる(LEDを光らせる)
clr time ..... カウントアップ回数のゼロクリア
clr rb ..... 表示値をゼロクリア
```

```
main mov w,time ..... カウントアップ回数データをWレジスタに持ってくる
mov rb,w ..... Wレジスタの内容をRB0~RB7に置く(LEDが光る)
call wait ..... 無駄時間を費やす
inc time ..... カウントアップの値を+1増加させる
```

```
btfscl ra,0 ..... スイッチS1はONになったか?
goto main ..... ONではない。カウントアップ動作継続
clr time ..... ONである。カウントアップ値をクリアする。
goto main ..... 表示プログラムへ戻る。
```

この命令は、条件を満足したとき、次の命令をスキップする。条件を満たしている場合は、timeメモリの内容をゼロクリアして、main位置へ戻る。

```
wait mov tm3,#2 ..... ムダ時間を費やすサブルーチン
wait2 clr tm1
wait0 clr tm2
wait1 nop
djnz tm2,wait1
djnz tm1,wait0
djnz tm3,wait2
ret
```