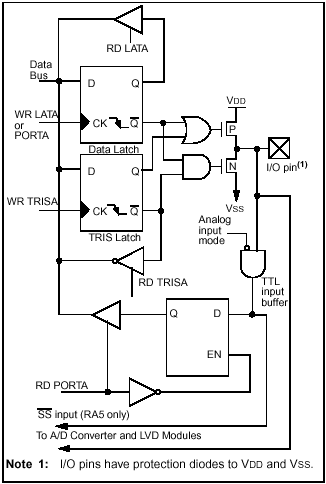
**PIC１８の入出力ピンの使い方**

**【入出力ピンの強化ポイント】**

　PIC18シリーズになって、入出力ピンの構造に強化が加えられています。  
その強化ポイントは下記のようになっています。  
  
(1) ビット操作命令を連続で使っても誤動作しないようにした。  
  
(2) クロック出力側のピンOSC2を汎用の入出力ピンRA6として使えるようにした。  
  
(3) RB3ピンがCCP2のサブ出力ピンとしても使えるようにした。

**【LAT＊レジスタの追加】**

　これはもともとあった出力ラッチをレジスタとして読めるようにして、ビット操作命令  
のときの事前読み出しを、直接入出力ピンから入力しないで、この出力ラッチからと  
することにより、外部回路の影響を無くしました。  
これで、いくら高速でビット操作命令を繰り返しても、出力が誤動作することはなくな  
りました。  
　この出力ラッチレジスタは「LAT\*」レジスタとして各ポート毎に用意されました。  
これを図で表すと、例えばPORT Aは下図のブロック図となり、Data Latchに付属する  
入力用ゲートが追加された回路です。  
  
  
この一番上にあるゲートが追加  
されたゲートでData Latchの内容  
を読み出すことができるように  
した。  
  
  
  
  
このLAT\*レジスタの追加によりポートへの出力の仕方が変更になります。  
  
**(1) 入出力ピンへの出力**  
　　PIC18シリーズになって、入出力ピンへの出力方法は２種類となりました。  
　　PORTつまり入出力ピンへの出力は、実際には、LAT\*レジスタへの出力と  
　　なります。この出力が実際にの入出力ピンに反映されるかどうかは、TRIS\*  
　　レジスタへの入出力モードに依存します。  
　　また、LAT\*レジスタへ出力すると、PORTへの出力と全く同じように動作します。  
　　それが外に出るかどうかはTRIS\*レジスタ次第ということになります。  
　　アセンブラ命令では下記のようになり、全く同じ動作をします。  
  
**MOVWF　　PORTB　 　;一旦LATBレジスタに書き込まれる  
  
　　　　MOVWF　　LATB　　　;LATBレジスタの直接書き込まれる**  
**(2) 入出力ピンからの入力**  
　　PIC18シリーズになって、ピンからの入力には２種類となりました。  
　　ひとつは、従来と同じようにPORTレジスタからの読み込みで、これは実際のピン  
　　の状態を入力します。  
　　もうひとつの入力は、LAT\*レジスタからの入力で、これは、実際のピンの状態に  
　　かかわらず、単にData Latchの現在状態を入力するだけです。  
　　アセンブラ命令では下記のようになります。  
  
**MOVF　　PORTB,W　　;実際の入出力ピンの状態読み込み  
  
　　　　MOVF　　LATB,W　 　;Data Latchの読み込み**  
**(3) ビット操作命令の連続実行**  
　　ビット操作命令は、一度現在状態を読込んでから、指定ビットのみ変更して  
　　書き戻すという処理をしています。PIC116シリーズではこの最初に読込む時、  
　　直接ピンの状態を読込んでいたため、容量性負荷の場合には、誤入力をする  
　　ことがあり、連続実行はしない方が良いとされていました。  
　　しかし、PIC18シリーズでは、ビット操作命令をPORTレジスタでは無く、LTA\*レジ  
　　スタに行うことで、現在状態をLAT\*レジスタから読込みます。従ってこの時には  
　　外部負荷の影響を受けることは無い為、連続実行も全く問題無く出来ます。  
  
**BSF　　LATB,5　　  
　　　　BSF　　LATB,7  
　　　　BCF　　TRISB,1**

**【OSC2を汎用の入出力ピンに】**

　クロック発振用のピンにはOSC1とOSC2の２ピンが用意されていますが、クロック  
発振モードによってはOSC2側は使いません。  
そこでこの空いたピンを汎用入出力ピンとして使えるようにして自由度を上げました。  
クロックの発振モードで下記２つの場合には、OSC2ピンは汎用の入出力ピン**RA6**と  
なります。  
  
**ECIO：外部クロックモード  
　　　RCIO：外部RC発振モード**

**【RB3ピンをCCP2出力にも切り替えられる】**

　CCP2の出力ピンは通常はRC1ピンとなっていますが、これをコンフィギュレーション  
ビットの、CCP2MXビットをイネーブルとすることでRB3に変更することが出来ます。  
これで、タイマー１の発振素子用にRC1/T1OSI/CCP2ピンを使っても、CCP2をRB3で  
使うことでタイマ１とCCP２を同時に使うことができます。  
また、ポートC側を有効に使いたい時にも、CCP2をポートBで扱うことが可能となり  
ます。

**【MPLAB-C18による入出力の仕方】**

MPLAB-C18を使って基本的なディジタル入出力を行う為には下記のようなプログラムと  
します。  
  
**(1) PORT全体に出力する場合**  
　　この場合には、まずTRIS\*レジスタを設定後、LAT\*レジスタに書き込みます  
　　下記のように単純に各レジスタを変数として定数を代入するだけです。  
　　これで自動的にAccess Bankが選択されますのでバンク指定は不要です。  
  
**TRISA = 0;　　　　　// ポートAをすべて出力ピンにする  
　　TRISB = 0;　　　　　// ポートBをすべて出力ピンにする  
　　TRISC = 0;　　　　　// ポートCをすべて出力ピンにする  
  
　　LATA = 0xFF; 　　　// ポートAの出力ラッチAにFFを出力  
　　　LATC = portc\_out;　// portc\_outの内容を出力ラッチCに出力  
　　LATB = 0x02;　　　 // ポートBの出力ラッチBに02を出力**  
**(2) ビット出力する場合**  
　　各PORTの各ビットは、標準のインクルードファイルの中で、構造体として定義されて  
　　います。従って、その構造体のメンバーとして各ビットを指定します。  
　　実際には、下記フォーマットで指定することになります  
  
　　　　　　**構造体名ｂｉｔｓ．ビット名**  
  
　　この構造体名とビット名は、標準インクルードファイルの中身を表示して見れば  
　　すぐに理解することが出来ます。  
　　下記が実際の例です。  
  
**TRISC = 0; 　　　　 　　// I/Oポート設定  
  
　　PORTCbits.RC2 = 1;　 　//RC2に1を出力  
　　PORTCbits.RC2 = 0;　　 //RC2を0にする**  
**(3) ＰＯＲＴ全体を入力する場合**  
　　この場合には、ＰＯＲＴをそのまま変数として扱うことが出来ます。  
　　  
**TRISB = 0xFF;　　　;全て入力モード  
  
　　datab = PORTB;　　 ;PORTBの内容を変数databに格納**  
  
**if(PORTB & 0x80)　 ;PORTBの８ビット目が0か1かの判定**  
  
  
**(4) ＰＯＲＴの１ビットを入力する場合**  
　　この場合にも出力と同様、各ビットが構造体で定義されていますから、その  
　　メンバーをそのまま変数として扱うことが出来ます。  
　　例えば、指定ビットを入力して０か１で判定する場合には下記のようにします。  
  
**TRISC = 0xF0;　　　　　　;上位を入力、下位を出力に  
　　　if(PORTCbits.RC7)　　　　;RC7を入力しテスト  
　　　　　PORTCbits.RC0 = 0;　 ;RC0を0に  
　　　else  
　　　　　PORTCbits.RC0 = 1;　　;RC0を1に**

**【実際の使用例】**

下記プログラムはMPLAB-C18で入出力ピンの入出力を行うプログラム例です。  
スイッチをPORT Dから読み込み、スイッチの内容に応じて、PORT Cの発光ダイオード  
の点滅制御を切り替えるようにしたものです。  
  
　　　　[★ 入出力プログラミング例](http://www.picfun.com/pic18/18ctest2.c)