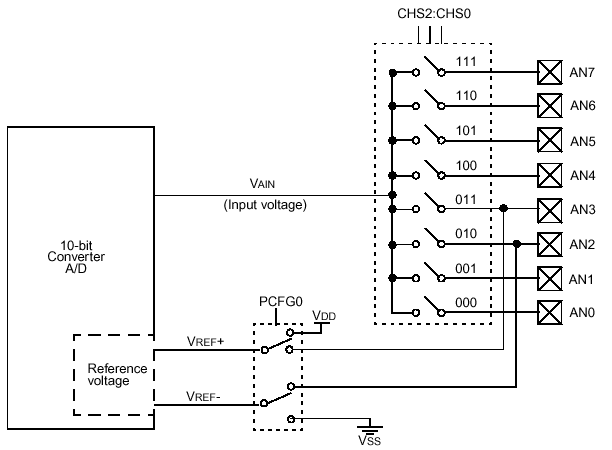
**http://www.picfun.com/graphics/greenball.gifA/D変換モジュールの使い方**

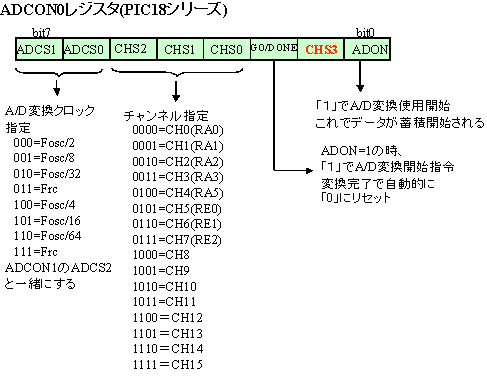
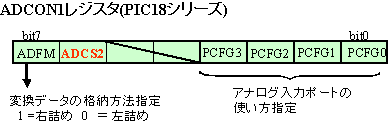
**【A/D変換モジュール概要】**

PIC18シリーズに内蔵のA/Dコンバータは、５または８チャンネルの１０ビット  
分解能を持つA/D変換モジュールとなっています。このA/D変換モジュール  
は基本的にPIC16F8ｘｘシリーズのものと同じです。

**【A/D変換モジュールの構成】**

A/D変換モジュールの内部構成は下図のようになっていて、動作は下記の  
順序で実行されます。  
  
(1) 動作モードを設定する。Vrefの指定、変換クロック速度の指定など。  
　　　(ADCON0、ADCON1レジスタで設定）  
(2) 入力チャンネルを選択する。  
　　　（ADCON0レジスタで指定）  
(3) サンプルホールドコンデンサの充電が開始される。（一定時間が必要）  
　　　　　（Acquisition Time)  
(4) A/D変換をスタートする。変換動作終了はフラグチェックか割込みで確認  
　　（ADCON0のGOビットOnでスタート、終了確認はGOビットが０になるのを待つ）  
(5) ２バイトのレジスタに変換結果が書き込まれている  
　　（ADRESL,ADRESHレジスタ）  
  


**【A/D変換制御用レジスタ】**

A/D変換モジュールを制御するためのレジスタには下記のように　ADCON0と  
ADCON1の２種類があります。このレジスタの内容は基本的にはPIC16シリーズと  
同じなのですが、チャンネル数が増えたのでチャネル指定ビットが増えたのと  
クロックの指定も細かくなったのでその指定ビットも増えています。  
  
  
  
  
  
  
  
  
  
  
  
アナログ入力ポートの使い方の設定は下表のようになります

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PCFG3－0 | RE2 | RE1 | RE0 | RA5 | RA3 | RA3 | RA1 | RA0 |
| 0000 | A | A | A | A | A | A | A | A |
| 0001 | A | A | A | A | Vref+ | A | A | A |
| 0010 | D | D | D | A | A | A | A | A |
| 0011 | D | D | D | A | Vref+ | A | A | A |
| 0100 | D | D | D | D | A | D | A | A |
| 0101 | D | D | D | D | Vref+ | D | A | A |
| 011x | D | D | D | D | D | D | D | D |
| 1000 | A | A | A | A | Vref+ | Vref- | A | A |
| 1001 | D | D | A | A | A | A | A | A |
| 1010 | D | D | A | A | Vref+ | A | A | A |
| 1011 | D | D | A | A | Vref+ | Vref- | A | A |
| 1100 | D | D | D | A | Vref+ | Vref- | A | A |
| 1101 | D | D | D | D | Vref+ | Vref- | A | A |
| 1110 | D | D | D | D | D | D | D | A |
| 1111 | D | D | D | D | Vref+ | Vref- | D | A |

　Aはアナログ入力　Dはディジタル入出力　Vref+　Vref-は基準電圧入力  
　基準電圧入力の指定が無いときは、電源電圧が基準となる。

**【A/D変換の精度】**

１０ビット精度のA/D変換モジュールですが、実際に１０ビットの精度を出すためには  
条件があります。下記条件を守らないと１０ビット精度は悪化します。  
  
**条件　VrefH　－　VrefL　＞　３V**  
  
つまりリファレンス電圧差が３V以上無いと精度は保証されません。

**【A/D変換用C言語関数】**

MPLAB-C18で用意されているA/D変換モジュール用のC言語ライブラリを  
説明します。  
A/D変換の制御は比較的やさしいのですが、Acquisition Timeを十分確保  
しないと精度が出なくなるので要注意です。  
  
**(1) ライブラリ関数の種類と機能**  
　　MPLAB-C18にあらかじめ用意されているA/D変換モジュール用の関数は  
　　下記表となっています。

|  |  |  |
| --- | --- | --- |
| 関数名 | 機能と基本形 | 使用例 |
| BusyADC | A/D変換中の時１を返す ADCON0レジスタのGOビットの状態を返す | while(BusyADC( )); |
| char BusyADC(void); |
| CloseADC | まずADCON0のADONビットをクリアしてA/Dコンバータを 未使用状態にしてから割り込みも禁止する | CloseADC( ); |
| void CloseADC(void); |
| ConvertADC | A/D変換を開始する。（ADCON0のGOビットを１にする） | ConvertADC(); |
| void ConvertADC(void); |
| OpenADC | A/D変換機能の使用モードを設定し動作を開始させる。 設定内容は、変換クロック、割込み、データ配置、 基準電圧、チャンネル数、チャネル選択 | OpenADC(ADC\_FOSC\_32 & 　ADC\_RIGHT\_JUST &  　ADC\_1ANA\_0REF, 　ADC\_CH & ADC\_INT\_OFF); |
| void OpenADC(unsigned char config, 　　　　　　　　　unsigned char config2); |
| ReadADC | A/D変換した結果の符号付２バイトのデータを返す。 右詰、左詰はOpenADCの設定による | int result; result = ReadADC( ); |
| int ReadADC(void); |
| SetChanADC | 一度チャンネル０に戻してから指定したチャンネルを選択する。 | SetChanADC(ADC\_CH0); |
| void SetChanADC(unsigned char channel); |

**(2) OpenADC用パラメータ**  
　　OpenADC関数にはA/D変換モジュールの動作モードを設定するための多くのパラメータが  
　　必要となりますが、これらは下記のようになっています。

|  |  |  |
| --- | --- | --- |
| 設定項目 | パラメータ名称 | 意味内容 |
| クロック選択 | ADC\_FOSC\_2 ADC\_FOSC\_4 ADC\_FOSC\_8 ADC\_FOSC\_16 ADC\_FOSC\_32 ADC\_FOSC\_64 ADC\_FOSC\_RC | Fosc/2 Fosc/4 Fosc/8 Fosc/16 Fosc/32 Fosc/64 内部クロック |
| 結果の詰め方 | ADC\_RIGHT\_JUST ADC\_LEFT\_JUST | 右詰め 左詰め |
| 使用チャンネル数 とリファレンス | ADC\_8ANA\_0REF  ADC\_7ANA\_1REF  ADC\_5ANA\_0REF ADC\_4ANA\_1REF ADC\_3ANA\_0REF ADC\_2ANA\_1REF ADC\_0ANA\_0REF ADC\_6ANA\_2REF ADC\_6ANA\_0REF ADC\_5ANA\_1REF ADC\_4ANA\_2REF ADC\_3ANA\_2REF ADC\_2ANA\_2REF ADC\_1ANA\_0REF ADC\_2ANA\_0REF | Vref+=Vdd,Vref-=Vss ８チャンネル全部使用 AN3=Vref+　他全チャン ネル使用 Vref=vdd Vref-=Vss AN3=Vref+ Vref+=Vdd、Vref-=Vss AN3=Vref+ 全部ディジタル AN3=Vref+、AN2=Vref- Vref+=Vdd、Vref-=Vss AN3=Vref+、Vref-=Vss AN3=Vref+、AN2=Vref- AN3=Vref+、AN2=Vref- AN3=Vref+、AN2=Vref- AN0のみ AN3=Vref+、AN2=Vref- |
| チャンネル指定 | ADC\_CH0 ADC\_CH1 ADC\_CH2 ADC\_CH3 ADC\_CH4 ADC\_CH5 ADC\_CH6 ADC\_CH7 | SetChanADC関数の パラメータとしても使用する |
| 割込み | ADC\_INT\_ON ADC\_INT\_OFF | 割込みを使用 割込み使わず |

**【A/D変換のプログラム例】**

実際にA/D変換を使ったプログラム例です。ここでは、PIC18活用製作例の  
汎用ボードを使ったテストプログラムです。  
  
２チャンネルのアナログ信号を読み取って、液晶表示器の各行に表示する  
動作を一定時間間隔で実行します。  
  
　　[★A/D変換モジュールテストプログラム（18test6.c)](http://www.picfun.com/pic18/18ctest6.c)  
  
  
《プログラム例》　メインルーチン部分のみは下記となっています。  
  
**//\*\*\*\*\*\* Main Function  
void main(void)  
{  
　　char Message1[8] = "Start!";  
　　int data;  
　　char str[7]; //define ASCII  
  
　　TRISB = 0;  
　　OpenXLCD(FOUR\_BIT);  
　　//define mode of A/D  
　　OpenADC(ADC\_FOSC\_16 & ADC\_RIGHT\_JUST & ADC\_3ANA\_0REF,  
　　　　　　　ADC\_CH0 & ADC\_INT\_OFF);   
  
　　putsXLCD(Message1); 　　　　　//Display start message  
  
　　while(1)  
　　{  
　　　　SetChanADC(ADC\_CH0);　　　//Select Channel 0  
　　　　Delay10TCYx(5); 　　　　　//20usec delay  
　　　　ConvertADC(); 　　　　　　//Start A/D  
　　　　while(BusyADC()); 　　　　//Wait end of conversion  
　　　　data = ReadADC(); 　　　 //Get A/D data  
　　　　itoa(data,str); 　　　　 //Convert to ASCII  
　　　　while(BusyXLCD());　　　 //Check LCD busy  
　　　　WriteCmdXLCD(0x01);　　　//Clear LCD  
　　　　putsXLCD(str); 　　　　　//Display ASCII data  
  
　　　　SetChanADC(ADC\_CH1); 　 //Select Channel 1  
　　　　Delay10TCYx(5); 　　　　 //20usec delay  
　　　　ConvertADC();　　　　　　//start A/D  
　　　　while(BusyADC());　　　　//wait end of conversion  
　　　　data = ReadADC();　　　 //get data  
　　　　itoa(data,str);　　　　 //convert to ASCII  
　　　　while(BusyXLCD());　　　//check LCD busy   
　　　　SetDDRamAddr(0x40);　　 //move to next line  
　　　　putsXLCD(str);　　　　　//display data  
　　　　Delay10KTCYx(200);　　 //wait about 0.8sec  
　　}  
}**