コンピュータ工学 講義プリント(10 月 30 日)

今回は、PIC16F84A の主要な機械語命令について、解説する。また、クロック周波数と命令の実行時間の関係についても説明する。

・W レジスタ(教科書 P.24 参照)

PIC16F84A の機械語を理解するに当たって、W レジスタ、ファイルレジスタ、バンク切り替え、特殊レジスタおよび Z・DC・C の各フラグの働きを前もって理解しておく必要がある。

W レジスタに関しては、10 月 16 日の講義の割り込みの使い方の説明の中で、簡単に紹介したが、ここでももう一度取り上げておく。

W レジスタは、Working Resister の略である。加減算や論理演算などを行う ALU(算術論理演算装置)に接続した 8 ビット幅のレジスタ(高速メモリ)で、加減算や論理演算の結果を格納するためや、データを転送する際の一時的なデータの保管に多用される。非常に多くの命令が、W レジスタを利用する。W レジスタは、この後説明するファイルレジスタのメモリ空間の外に存在する(W レジスタはファイルレジスタの一種ではない)ので、アドレスを持たないレジスタである。

・ファイルレジスタ(教科書 P.25 参照)

ファイルレジスタは、データを格納する、8 ビット幅の高速なメモリである。アドレスバスの幅が 7 ビットなので、アドレスが 0H 番地～7F 番地までとなり、10 進数では 128 のアドレスがあることになる。
ただし、16F84A では、4FH 番地までしか使っていない。

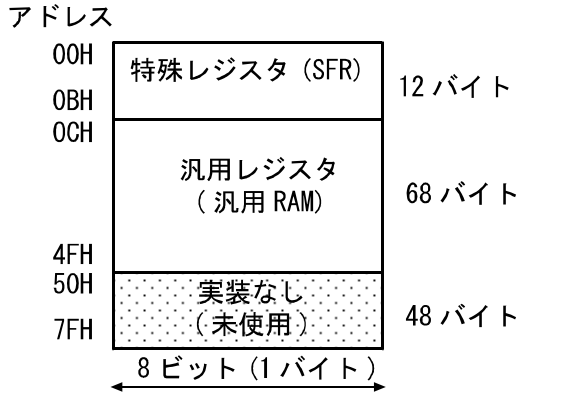

図 1、ファイルレジスタのアドレスマップ

0H 番地～0BH 番地までの 12 のアドレスは、後述する特殊レジスタ(SFR)に割り当てられている。

残りの 0CH 番地～4FH 番地までの 68 個のアドレスが、汎用レジスタ(汎用 RAM)として割り当てられている。汎用 RAM には、データを自由に記録しておく事ができるが、RAM なので、電源を切るとデータは消失する。

・バンク切り替え(教科書 P.25 参照)

ファイルレジスタのアドレスバスは 7 ビットなので、最大 128 バイトまでのデータしか扱えない。そこで、バンク切り替えという仕組みで、さらに大きな情報を扱えるようにしてある。

STATUS レジスタのビット 5 には RP0 という名前がついているが、これをファイルレジスタのアドレスバスの最上位ビットとみなす事で、アドレスバスを 8 ビットに拡張している。このことにより、ファイルレジスタの空間が 2 倍の 256 バイトにまで広がる。

RP0 を 0 に設定している時は、0H～7FH までのアドレスにアクセスできる。この範囲のアドレスをバンク 0 と呼ぶ。

RP0 を 1 に設定している時は、80H～FFH までのアドレスにアクセスできる。こ a の範囲のアドレスをバンク 1 と呼ぶ。

この様に、RP0 を書き換えることで、バンク 0 とバンク 1 を切り替えながら、ファイルレジスタにアクセスする事をバンク切り替えという。バンク 0 とバンク 1 を頻繁に切り替えてアクセスすると、バンク切り替えの度に、RP0 を書き換える命令(後述)を挿入する必要があり、プログラムが長くなったり、実行速度が低下したりする。

RP0 を 0 に設定して、バンク 0 に切り替えるには、後述する BCF 命令を使って BCF STATUS,RP0 とすればよい。また RP0 を 1 に設定して、バンク 1 に切り替えるには BSF STATUS,RP0 とすればよい。

・特殊レジスタ(SFR)(教科書 P.26 参照)

ファイルレジスタの上位 12 バイト(バンク 0 の場合、アドレス 00H～0BH。バンク 1 の場合、アドレス 80H～8BH)は、SFR という、CPU や周辺機能のコントロール用のレジスタに割り当てられている。SFR の一覧は、教科書 P.27 の表 2.3 に掲載されている。

下位 7 ビットが同じアドレスに、バンク 0 とバンク 1 で同じ名前のレジスタが割り当てられている場合も多い(例えばアドレス 03H と 83H は、共に STATUS レジスタ)が、2 つの内どちらのアドレスにアクセスしても、同様の動作をする。

また、汎用 RAM に関しても、下位 7 ビットが同じアドレスには、バンク 0 とバンク 1 で同一の RAM が割り当てられている。(教科書 P.26 の図 2.14 を参照)

・Z フラグ、DC フラグ、C フラグ(教科書 P.67 参照)

教科書 P.67 の図 4.5 に書いてあるように、STATUS レジスタの下位 3 ビットは、Z、DC、C の各フラグとなっている。これらのフラグは、命令を実行した際に、実行結果に応じて勝手に書き換わる事がある。

Z フラグは、演算命令を実行した結果がゼロになった場合などに 1 になるフラグである。演算結果がゼロでなければ、Z フラグは 0 になる。

DC フラグは、加算命令と減算命令の実行時に変化するフラグである。加算命令においてはビット 3 とビット 4 の境界で繰り上がりが発生した場合に 1、繰り上がりが発生しなかったときに 0 となる。減算命令においては、ビット 3 とビット 4 の境界で繰り下がりが発生した場合に 0、繰り下がりがりが発生しなかったときに 1 となる。

C フラグは、加算命令、減算命令、およびローテイト命令の実行時に変化するフラグである。加算命令においては最上位ビットで繰り上がりが発生した場合に 1、繰り上がりが発生しなかったときに 0 となる。減算命令においては最上位ビットで繰り下がりが発生した場合に 0、繰り下がりがりが発生しなかったときに 1 となる。ローテイト命令の場合については、教科書 P.81～82 を参照せよ。

これら 3 つのフラグはいずれも、それぞれのフラグに影響しない命令の実行の前後では、値が保存される。

・PIC16F84A の命令の種類(教科書 P.69 参照)

PIC16F84A は非常にシンプルな CPU を搭載したマイコンで、命令がたった 35 種類しかない。Arduino UNO に搭載されている ATmega328P というマイコンには 131 の命令があるが、それと比較すると、かなり命令数が少ない。

PIC16F84A は命令数が少ない分だけ、処理能力が低いが、別の視点で物事を見れば、初学者には理解しやすいマイコンだとも言える。

PIC16F84A の命令は、以下の 4 つに分類できる。

(1) バイト対応命令

1 バイト単位で演算やデータの転送をする命令で、MOVF、MOVWF、SWAPF、ADDWF、SUBWF、INCF、DECF、ANDWF、IORWF、COMF、XORWF、RRF、RLF、INCFSZ、DECFSZ、CLRF、CLRW、NOP の 18 命令がある。

(2) ビット対応命令

ファイルレジスタの特定のビットを書き換えたり、値を調べたりする命令で、BCF、BSF、BTFSC、BTFSS の 4 命令がある。

(3) リテラル対応命令

リテラル(定数)を W レジスタに代入したり、リテラルと、ファイルレジスタあるいは W レジスタの内容の間で、演算を行った結果を、W レジスタに代入したりする命令で、MOVLW、ADDLW、SUBLW、ANDLW、IORLW、XORLW の 6 命令がある。

(4) 制御命令

特定のアドレスにジャンプしたり、サブルーチンを呼び出したりといった、プログラムの流れを制御するための命令で、CALL、GOTO、RETURN、RETLW、RETFIE、CLRWDT、SLEEP の 7 命令がある。

なお、教科書では、リテラル対応命令と制御命令をまとめて扱っているが、これはリテラル対応命令と、CALL 命令・GOTO 命令を除く制御命令とでは、命令のフォーマットが同じだからである。(教科書 P.66 参照)

・MOVF 命令(教科書 P.70 参照)

ここからは、主だった命令の解説をする。

MOVF 命令はファイルレジスタの内容を、W レジスタまたは同一アドレスのファイルレジスタに転送する命令である。

MOVF 命令は f と d の 2 つのオペランドを取る。f はファイルレジスタのアドレスで、d は転送先を指定する 0 または 1 の数字である。d=0 ならば、転送先は W レジスタになり、d=1 ならば、転送先は、f と同一アドレスのファイルレジスタとなる。

この命令は、転送したデータが 0 ならば Z フラグが 1 に、転送したデータが 1 ならば Z フラグが 0 になる。

d=1 の場合は転送元と転送先が同じファイルレジスタであるので、Z フラグが変化する以外の効果は持たない。

f は 7 ビットの数であるので、ファイルレジスタのアドレスの下位 7 ビットしか指定できない。現在選択しているページと違うページのファイルレジスタを転送元とするには、先に STATUS レジスタの RP0 を書き換える命令を実行しておかなければならない。(MOVWF 命令など、f をオペランドに取る他の命令でも同様)

・MOVWF 命令(教科書 P.71 参照)

MOVWF 命令は、W レジスタの内容をファイルレジスタに転送するための命令である。オペランドは f のみで、ファイルレジスタのアドレス(の下位 7 ビット)を指定する。この命令は、どのフラグにも影響を与えない。

・SWAPF 命令(教科書 P.72 参照)

オペランド f で指定されたファイルレジスタの上位 4 ビットと下位 4 ビットを交換し、オペランド d で指定された転送先に転送する命令である。どのフラグにも影響を与えない。

・ADDWF 命令、SUBWF 命令、ANDWF 命令、IORWF 命令、XORWF 命令(教科書 P.73、74、77、78、80 参照)

これらの命令は、全て f と d の 2 つのオペランドを取る。

f で指定されるファイルレジスタと、W レジスタの間で演算を行い、演算結果を d で指定される転送先に転送する。d=0 の場合は W レジスタに転送する。d=1 の場合は、f で指定されたファイルレジスタに転送する。

ADDWF 命令の場合は加算、SUBWF 命令の場合は減算、ANDWF 命令の場合は論理積、IORWF 命令の場合は論理和、XORWF 命令の場合は排他的論理和の演算を行う。

演算結果が 0 か否かで、Z フラグが変化する。ADDWF 命令と SUBWF 命令の場合は、DC フラグと C フラグも、演算結果により変化する。

・INCF 命令と DECF 命令(教科書 P.75、76 参照)

どちらの命令も、f と d の 2 つのオペランドを取る。

f で指定されたファイルレジスタの値に 1 を足す(INCF 命令の場合)か 1 を引く(DEC 命令の場合)かして、その結果を d で指定された転送先に転送する。

演算結果により Z フラグが変化する。

・COMF 命令(教科書 P.79)

COMF 命令は f と d の 2 つのオペランドを取る。

f で指定されたファイルレジスタの値の論理否定を計算し、その結果を d で指定された転送先に転送する。

演算結果により Z フラグが変化する。

・RRF 命令と RLF 命令(教科書 P.81、82 参照)

どちらの命令も、f と d の 2 つのオペランドを取る。

f で指定されたファイルレジスタの値を、C フラグを含めて 1 ビット右にローテイト(RRF 命令の場合)または左にローテイト(RLF 命令の場合)し、その結果を d で指定された転送先に転送する。

演算結果により C フラグが変化する。

・INCFSZ 命令と DECFSZ 命令(教科書 P.83、84 参照)

どちらの命令も、f と d の 2 つのオペランドを取る。

f で指定されたファイルレジスタの値に 1 を足す(INCFSZ 命令の場合)か 1 を引く(DECFSZ 命令の場合)かして、d で指定された転送先に転送する。そして、演算結果が 0 の場合は、次の命令を 1 つスキップする(次の命令を実行しない)。

これらの命令は、どのフラグにも影響を与えない。

・NOP 命令(教科書 P.87 参照)

オペランドは取らず、何もしない命令である。フラグも変化させない。単に命令 1 サイクル時間を消費するだけである。

・BCF 命令、BSF 命令(教科書 P.88、89 参照)

どちらの命令も、f と b の 2 つのオペランドを取る。b は、ファイルレジスタのビット位置を指定する 0～7 の数字である。(0 が最下位ビットで 7 が最上位ビット)

f 指定されたファイルレジスタの b ビット目を 0(BCF 命令の場合)または 1(BSF 命令の場合)にする。

フラグに影響は与えない。

・BTFSC 命令、BTFSS 命令(教科書 P.90、91 参照)

どちらの命令も、f と b の 2 つのオペランドを取る。

f 指定されたファイルレジスタの b ビット目が 0(BTFSC 命令の場合)あるいは 1(BTFSS 命令の場合)の場合に、次の命令をスキップする。

フラグには影響を与えない。

・MOVLW 命令(教科書 P.92 参照)

MOVLW 命令は一つのオペランド k を取る。k は 8 ビットリテラル(8 ビットの定数)である。

MOVLW 命令は、k を W レジスタに転送する。

フラグには影響を与えない。

・ADDLW 命令、SUBLW 命令、ANDLW 命令、IORLW 命令、XORLW 命令(教科書 P.93～97 参照)

これらの命令は、一つの 8 ビットリテラル k をオペランドに取る。

k と W レジスタの間で演算を行い、演算結果を W レジスタに転送する。

ADDLW 命令の場合は加算、SUBLW 命令の場合は減算、ANDLW 命令の場合は論理積、IORLW 命令の場合は論理和、XORLW 命令の場合は排他的論理和の演算を行う。

演算結果が 0 か否かで、Z フラグが変化する。また、ADDLW 命令と SUBLW 命令の場合は、DC フラグと C フラグも影響を受ける。

・CALL 命令(教科書 P.98 参照)

CALL 命令は一つのオペランド k を取る。k は 11 ビットのフラッシュメモリ(ROM)のアドレスである。

CALL 命令は、k 番地から始まるサブルーチンに制御を移す。さらに具体的には、復帰アドレス(CALL 命令の次の命令のアドレス)をスタックに積み、k 番地にジャンプする。

フラグには影響を与えない。

・GOTO 命令(教科書 P.99 参照)

GOTO 命令は一つのオペランド k を取る。

GOTO 命令は k 番地にジャンプする。

フラグには影響を与えない。

・RETURN 命令、RETLW 命令(教科書 P.101、102 参照)

RETURN 命令はオペランドを取らない。RETLW は 8 ビットリテラルの k をオペランドに取る。

RETURN 命令も RETLW 命令も、サブルーチンから復帰する。(スタックから復帰アドレスを読み出し、そのアドレスにジャンプする)RETLW 命令の場合は、サブルーチンから復帰する前に、W レジスタに k を代入する。

フラグには影響を与えない。

・RETFIE 命令(教科書 P.102 参照)

RETFIE 命令はオペランドを取らない。

割り込みサービスルーチン(ISR)から復帰する。RETURN 命令と同様に、スタックから復帰アドレスを読み出し、そのアドレスにジャンプするが、RETFIE はそのジャンプの前に INTCON レジスタの GIE ビットを 1 にセットして、割り込みを有効にする。

フラグには影響を与えない。

・命令の実行時間(教科書 P.38 参照)

PIC16F84A では、クロック発振器から与えられたクロック 4 つで、1 命令を実行する。この 4 クロックの事を、1 命令サイクルと呼ぶ。

ただし、CALL 命令、GOTO 命令、RETURN 命令などの様に、命令実行の流れを変えてしまう命令では、次に実行する命令をフェッチしなおさなければならないので、実行に倍の 8 クロック(2 命令サイクル)かかる。

INCFSZ 命令の様に、条件により次の命令を実行したり、スキップしたりする命令では、次の命令を実行する場合は 1 命令サイクル(4 クロック)で実行でき、次の命令をスキップする場合では 2 命令サイクル(8 クロック)かかるので、注意が必要である。

ここで、クロック発振器の発振周波数を 20MHz と仮定して、具体的に命令の実行時間を計算してみる。20MHz のクロックの周期は $1\div(20\times10^{6})=50\times10^{-9}$[s]=50[ns]である。この 4 倍が 1 命令サイクルであるから、50×4=200[ns]が 1 命令サイクルとなる。よって、ほとんどの命令の実行時間は 200[ns]となる。CALL 命令などの 2 命令サイクルかかる命令では、実行に倍の 400[ns]の時間が必要である。

・16 ビットの演算

PIC16F84A には、8 ビットの演算の機能しかない。よって、16 ビットの演算を行うには、8 ビットの演算を 2 回行う必要がある。

16 ビットの 2 つの数 1234H と AA55H の論理和を計算するプログラムの例をリスト 1 に示す。

リスト 1、16 ビットの論理和を計算するプログラム

```
; 16 ビットの論理和の計算をするプログラム(A OR B → C)
        LIST    P=PIC16F84A
        INCLUDE "P16F84A.INC"
AL      EQU     0CH     ; A の下位 8 ビット
AH      EQU     0DH     ; A の上位 8 ビット
BL      EQU     0EH     ; B の下位 8 ビット
BH      EQU     0FH     ; B の上位 8 ビット
CL      EQU     010H    ; C(演算結果)の下位 8 ビット
CH      EQU     011H    ; C(演算結果)の上位 8 ビット
        ORG     0
; A に 1234H をセットする
        MOVLW   034H
        MOVWF   AL
        MOVLW   012H
        MOVWF   AH
; B に AA55H をセットする
        MOVLW   055H
        MOVWF   BL
        MOVLW   0AAH
        MOVWF   BH
; 下位 8 ビットの論理和を計算する
        MOVF    AL,0
        IORWF   BL,0
        MOVWF   CL
; 上位 8 ビットの論理和を計算する
        MOVF    AH,0
        IORWF   BH,0
        MOVWF   CH
; この時点で C に計算結果の BA75H が入っている
; 無限ループする
LOOP    GOTO    LOOP
        END
```

このリストを見ると、下位 8 ビットと上位 8 ビットで別々に論理和の計算をしていることが分かる。

足し算や引き算になると、上位 8 ビットと下位 8 ビットの間で、繰り上がりや繰り下がりが発生するので、さらに話が複雑になる。

2 つの 16 ビットの数、1234H と ABCDH の和を計算するプログラムの例を、リスト 2 に示す。

リスト 2、16 ビットの算術和を計算するプログラム

```
; 16 ビットの加算をするプログラム(A + B → C)
        LIST    P=PIC16F84A
        INCLUDE "P16F84A.INC"
AL      EQU     0CH     ; A の下位 8 ビット
AH      EQU     0DH     ; A の上位 8 ビット
BL      EQU     0EH     ; B の下位 8 ビット
BH      EQU     0FH     ; B の上位 8 ビット
CL      EQU     010H    ; C(演算結果)の下位 8 ビット
CH      EQU     011H    ; C(演算結果)の上位 8 ビット
        ORG     0
; A に 1234H をセットする
        MOVLW   034H
        MOVWF   AL
        MOVLW   012H
        MOVWF   AH
; B に ABCDH をセットする
        MOVLW   0CDH
        MOVWF   BL
        MOVLW   0ABH
        MOVWF   BH
; 下位 8 ビットの和を計算する
        MOVF    AL,0
        ADDWF   BL,0
        MOVWF   CL
; 上位 8 ビットの和を計算する
        MOVF    AH,0
        BTFSC   STATUS,C    ; C フラグが 0 ならスキップ
        ADDLW   1           ; 繰り上がり処理
        ADDWF   BH,0
        MOVWF   CH
; この時点で C に計算結果の BE01H が入っている
; 無限ループする
LOOP    GOTO    LOOP
        END
```

このリストを見ると、上位 8 ビットの和を計算するときに、C フラグが 1 であれば(下位 8 ビットからの繰り上がりがあれば)、繰り上がり処理を行っている。C フラグの状態の判定には BTFSC 命令を用いている。