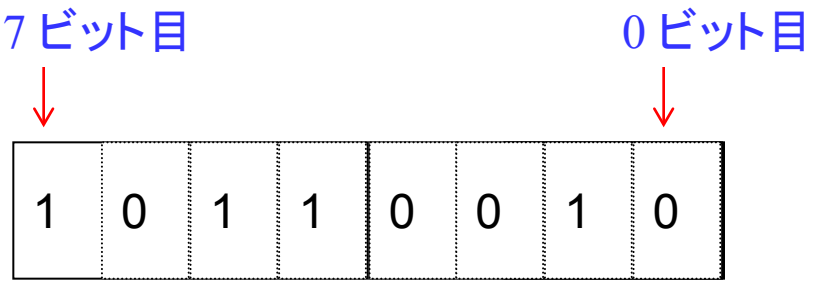
４．演算命令（つづき）
（足し算の桁上がり，Rotate, etc.）を学ぼう

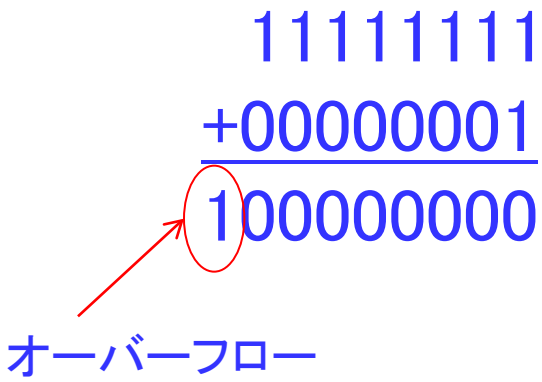
本稿のWebページ
http://www.mybook-pub-site.sakura.ne.jp/PIC/index.html

本章では足し算の桁上がり情報の格納場所の確認をするプログラムを学びます．

PIC16Fマイコンではデータは8ビットで表されています．最上位ビットは7ビット目，最下位ビットは0ビット目と数えられます．



以下の足し算を実行すると，8ビット目に桁上がりした1は8ビットのレジスタに収まらずオーバーフローします．この値は捨てられることなく，STATUSレジスタの0ビット目に格納されます．



このソースファイルを各自打ち込んで下さい.

```
; ADD Carry Check

        INCLUDE"p16F84A.inc"
        list p=16F84A

        __CONFIG  _HS_OSC  &  _WDT_OFF  &  _PWRTE_OFF  &  _CP_OFF

MEM1    EQU     0x0C                ;MEM1 at 0C

        ORG     0
        GOTO    START
        ORG     4

START

        BCF     STATUS,0

        MOVLW   0xFF                ; Load 0xFF to W

        MOVWF   MEM1                ; Move W to MEM1

        MOVLW   0x01                ; Load 0x01 to W

        ADDWF   MEM1, 0             ; W + MEM1 -> W


        GOTO    START

        END
```

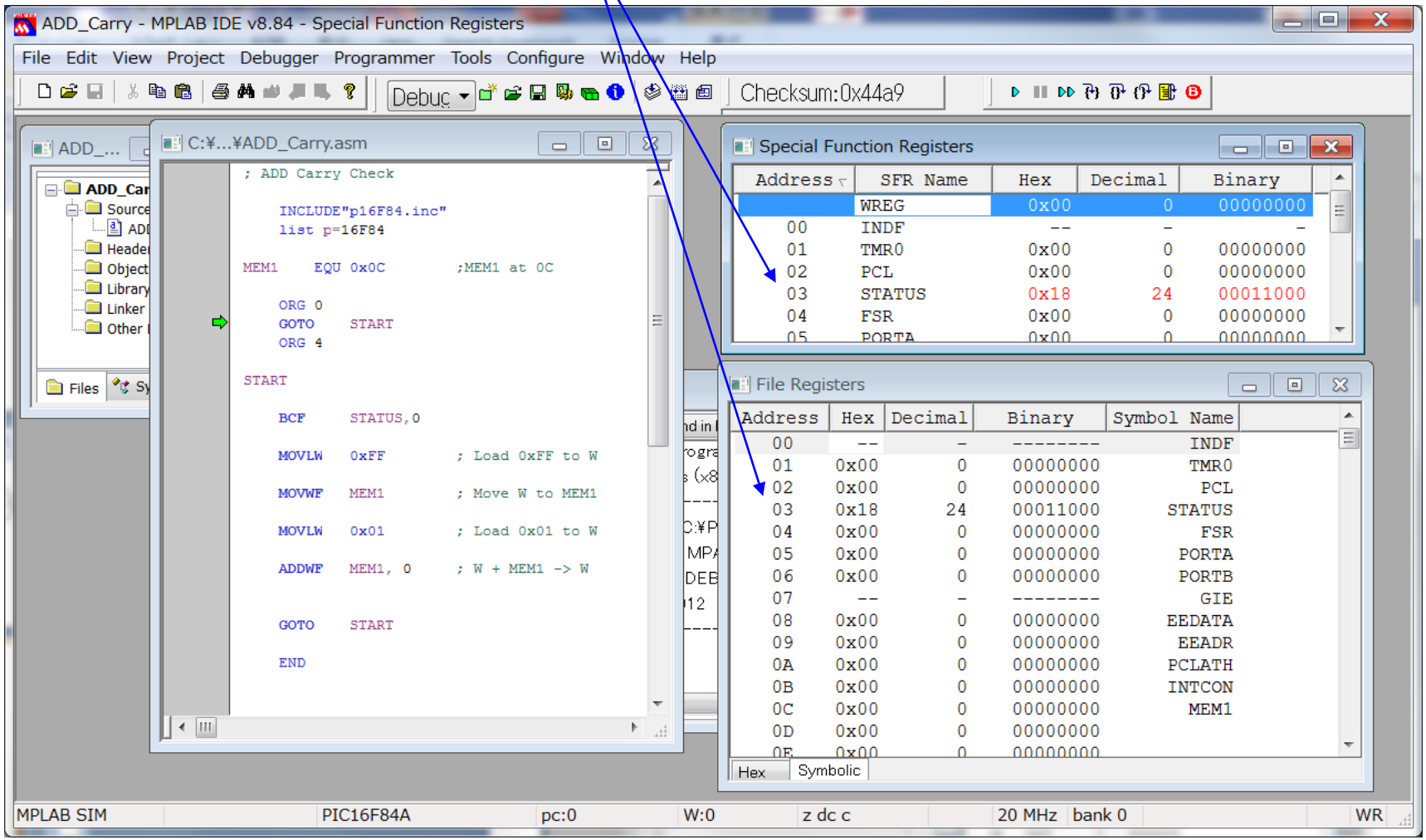
Data Sheet の REGISTER FILE MAP より

ファイルレジスタの03h番地と83h番地にSTATUSレジスタがあります．STATUSレジスタは1つだけです．どちらの番地からでも同じレジスタにアクセスできます．

TMR0	01h	OPTION_REG	81h
PCL	02h	PCL	82h
STATUS	03h	STATUS	83h
FSR	04h	FSR	84h
PORTA	05h	TRISA	85h
PORTB	06h	TRISB	86h
	07h		87h
EEDATA	08h	EECON1	88h
EEADR	09h	EECON2	89h
PCLATH	0Ah	PCLATH	8Ah
INTCON	0Bh	INTCON	8Bh
General Purpose Register 68バイト	0Ch	Mapped in Bank 0	8Ch
	4Fh		CFh
Bank 0		Bank 1	

03hとH'03'と0x03はいずれも16進数の3を表します．

STATUSレジスタはSpecial Function RegistersとFile Registersのどちらのウィンドウでも見ることができます.



```
; ADD Carry Check

            INCLUDE"p16F84A.inc"
            list p=16F84A


            __CONFIG   _HS_OSC  &   _WI

MEM1        EQU         0x0C

            ORG         0
            GOTO        START
            ORG         4

START
            BCF         STATUS,0

            MOVLW       0xFF

            MOVWF       MEM1                ; Move W to MEM1

            MOVLW       0x01                ; Load 0x01 to W

            ADDWF       MEM1, 0             ; W + MEM1 -> W


            GOTO        START

            END
```

C:¥Program Files (x86)¥Microchip¥MPASM Suiteの中にp16F84A.incファイルがあります．これをテキストエディタを使って開くと

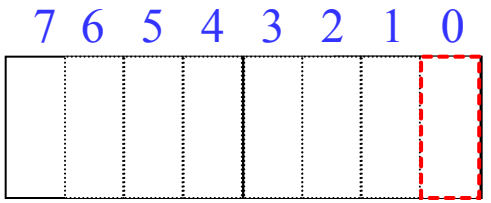
```
;==========================================
;       Register Definitions
;==========================================
;-----Bank0-----------------
STATUS          EQU  H'0003‘
```

という記述が見られます．0x03もH’0003’もどちらも16進数の3を意味します．すなわちSTATUSという言葉は16進数の3と同じ意味であることを宣言しています．

ここでSTATUSを使っています．既にincファイル内にてSTATUSが3を表すことが宣言されています．従って，これはファイルレジスタの3番地，すなわちSTATUSレジスタに対して操作を行うことを意味します．

```
; ADD Carry Check

            INCLUDE"p16F84A.inc"
            list p=16F84A


            __CONFIG   _HS_OSC  &   _WDT_OFF  &   _PWRTE_

MEM1        EQU         0x0C                ;MEM1 at 0C

            ORG         0
            GOTO        START
            ORG         4

START
            BCF         STATUS,0

            MOVLW       0xFF                ; Load 0xFF to W

            MOVWF       MEM1                ; Move W to MEM1

            MOVLW       0x01                ; Load 0x01 to W

            ADDWF       MEM1, 0             ; W + MEM1 -> W


            GOTO        START

            END
```

STATUSレジスタ



STATUSレジスタの第0ビット（キャリーフラグという名前がつけられています．）を0にします．
BCF f, b　Bit Clear f の略．
fレジスタの第bビットをクリア（0に）する命令です．

```
; ADD Carry Check

        INCLUDE"p16F84A.inc"
        list p=16F84A


        __CONFIG  _HS_OSC  &  _WDT_OFF  &  _PWRTE_OFF

MEM1    EQU       0x0C                ;MEM1 at 0C

        ORG       0
        GOTO      START
        ORG

START

        MOVWF     MEM1                ; Move W to MEM1

        MOVLW     0x01                ; Load 0x01 to W

        ADDWF     MEM1, 0             ; W + MEM1 -> W


        GOTO      START

        END
```

足し算の結果に桁上がりがあるとキャリーフラグが1になります.

STATUSレジスタ

7	6	5	4	3	2	1	0
							1

11111111
+ 00000001
100000000

このソースファイルを各自打ち込んで下さい.

Rotate命令を学んで下さい.

```
; Rotate

            INCLUDE"p16F84A.inc"
            list p=16F84A

            __CONFIG  _HS_OSC  &  _WDT_OFF  &  _PWRTE_OFF  &  _CP_OFF

MEM1        EQU       0x0C                ;MEM1 at 0C

            ORG       0
            GOTO      START
            ORG       4

START
            MOVLW     B'00110000'         ; Load 0x30 to W

            MOVWF     MEM1                ; Move W to MEM1

            BCF       STATUS,0            ;Clear the 0-th bit of STATUS register

            RLF       MEM1,1              ; Rotate MEM1 Left
            RLF       MEM1,1              ; Rotate MEM1 Left
            RLF       MEM1,1              ; Rotate MEM1 Left
            RLF       MEM1,1              ; Rotate MEM1 Left
            RLF       MEM1,1              ; Rotate MEM1 Left

            RRF       MEM1,1              ; Rotate MEM1 Right
            RRF       MEM1,1              ; Rotate MEM1 Right
            RRF       MEM1,1              ; Rotate MEM1 Right
            RRF       MEM1,1              ; Rotate MEM1 Right
            RRF       MEM1,1              ; Rotate MEM1 Right

            GOTO      START

            END
```

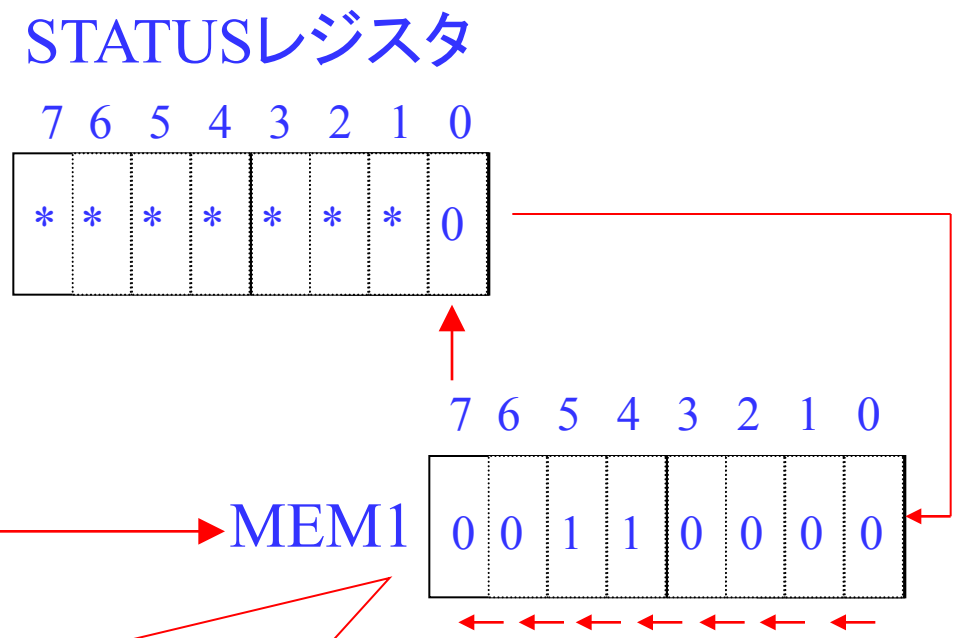
```
; Rotate

            INCLUDE"p16F84A.inc"
            list p=16F84A

            __CONFIG  _HS_OSC  &  _WDT_OFF  &  _PWRTE

MEM1        EQU       0x0C                ;MEM1 at 0C

            ORG       0
            GOTO      START
            ORG       4

START
            MOVLW     B'00110000'         ; Load 0x30 to W

            MOVWF     MEM1                ; Move W to MEM1

            BCF       STATUS,0            ;Clear the 0-th bit of STATUS register

            RLF       MEM1,1              ; Rotate MEM1 Left
            RLF       MEM1,1              ; Rotate MEM1 Left
            RLF       MEM1,1              ; Rotate MEM1 Left
            RLF       MEM1,1              ; Rotate MEM1 Left
            RLF       MEM1,1              ; Rotate MEM1 Left

            RRF       MEM1,1              ; Rotate MEM1 Right
            RRF       MEM1,1              ; Rotate MEM1 Right
            RRF       MEM1,1              ; Rotate MEM1 Right
            RRF       MEM1,1              ; Rotate MEM1 Right
            RRF       MEM1,1              ; Rotate MEM1 Right

            GOTO      START

            END
```

STATUSレジスタ

7	6	5	4	3	2	1	0
*	*	*	*	*	*	*	0

STATUSレジスタの第0ビット（キャリーフラグ）を0にします.

```
; Rotate

        INCLUDE"p16F84A.inc"
        list p=16F84A

        __CONFIG  _HS_OSC  &  _WI

MEM1    EQU     0x0C

        ORG     0
        GOTO    START
        ORG     4

START
        MOVLW   B'00110000'

        MOVWF   MEM1
        BCF     STATUS,0

        RLF     MEM1,1
        RLF     MEM1,1
        RLF     MEM1,1
        RLF     MEM1,1
        RLF     MEM1,1

        RRF     MEM1,1
        RRF     MEM1,1
        RRF     MEM1,1
        RRF     MEM1,1
        RRF     MEM1,1

        GOTO    START

        END
```

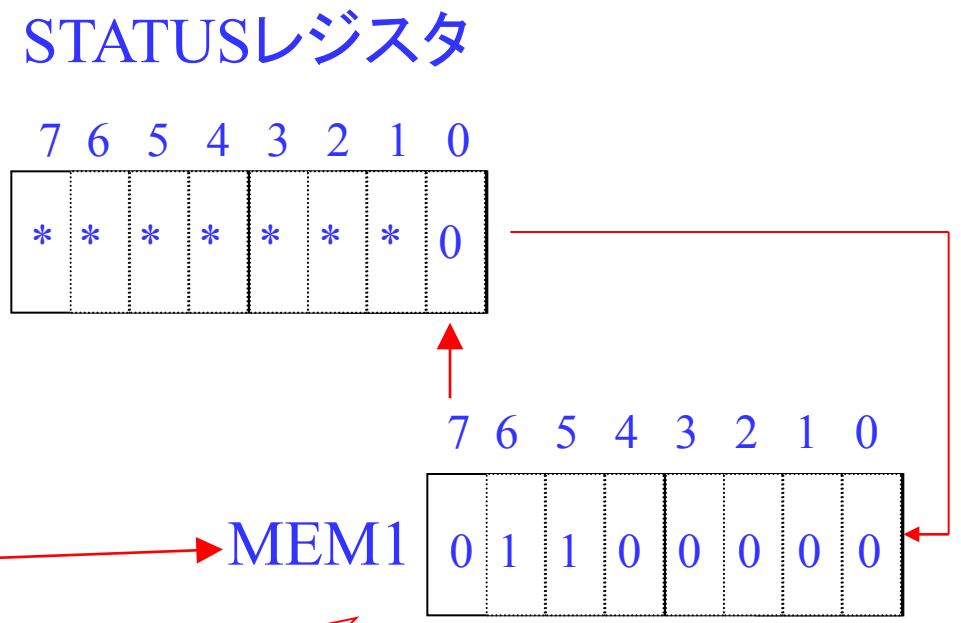


MEM1の中身を1ビット左に移動させます．一番左の第7ビット目の中身は外に出てしまうので，これをキャリーフラグに入れます．一番右の第0ビット目にはキャリーフラグの中身を転送します．その結果をMEM1に転送します．

RLF f, d　　Rotate Left f の略

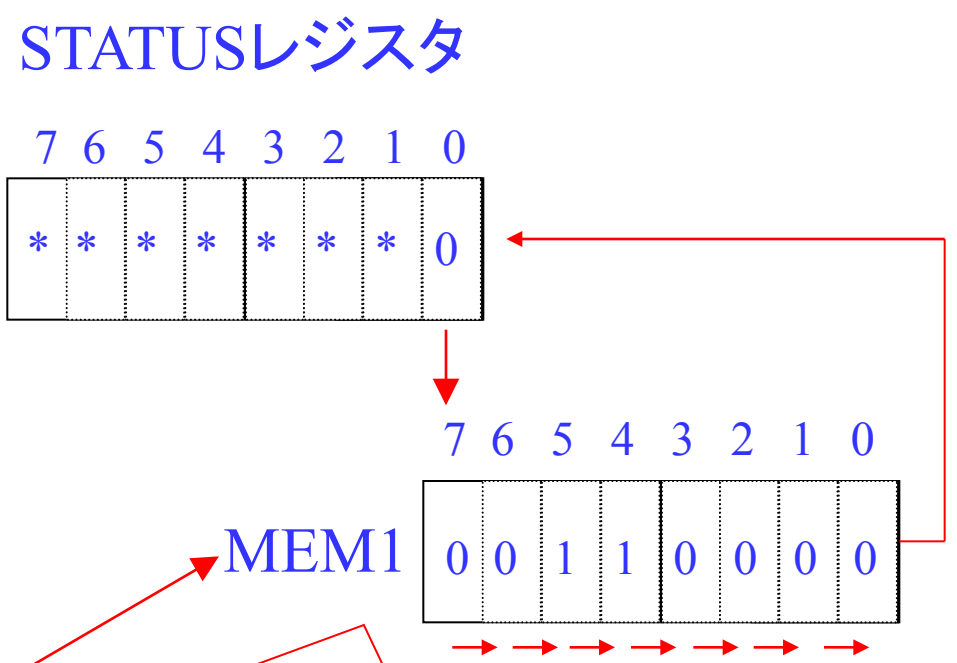
d = 1のとき転送先は f レジスタ，d = 0 のときはwレジスタです．

```
; Rotate

        INCLUDE"p16F84A.inc"
        list p=16F84A

        __CONFIG  _HS_OSC  &  _WDT

MEM1    EQU     0x0C

        ORG     0
        GOTO    START
        ORG     4

START
        MOVLW   B'00110000'

        MOVWF   MEM1
        BCF     STATUS,0

        RLF     MEM1,1
        RLF     MEM1,1
        RLF     MEM1,1
        RLF     MEM1,1
        RLF     MEM1,1

        RRF     MEM1,1
        RRF     MEM1,1
        RRF     MEM1,1
        RRF     MEM1,1
        RRF     MEM1,1

        GOTO    START

        END
```



最初のRLF実行後のMEM1の中身です．
以降のRLFを順次STEP実行し，MEM1およびSTATUSレジスタの中身の変化を確認してください．

```
; Rotate

            INCLUDE"p16F84A.inc"
            list p=16F84A

            __CONFIG   _HS_OSC  &  _

MEM1        EQU         0x0C

            ORG         0
            GOTO        START
            ORG         4

START
            MOVLW       B'00110000'

            MOVWF       MEM1
            BCF         STATUS,0

            RLF         MEM1,1
            RLF         MEM1,1
            RLF         MEM1,1
            RLF         MEM1,1
            RLF         MEM1,1

            RRF         MEM1,1
            RRF         MEM1,1
            RRF         MEM1,1
            RRF         MEM1,1
            RRF         MEM1,1

            GOTO        START

            END
```



MEM1の中身を1ビット右に移動させます．一番右の第0ビット目の中身は外に出てしまうので，これをキャリーフラグに入れます．一番左の第7ビット目にはキャリーフラグの中身を転送します．

RRF f, d　　Rotate Right f の略

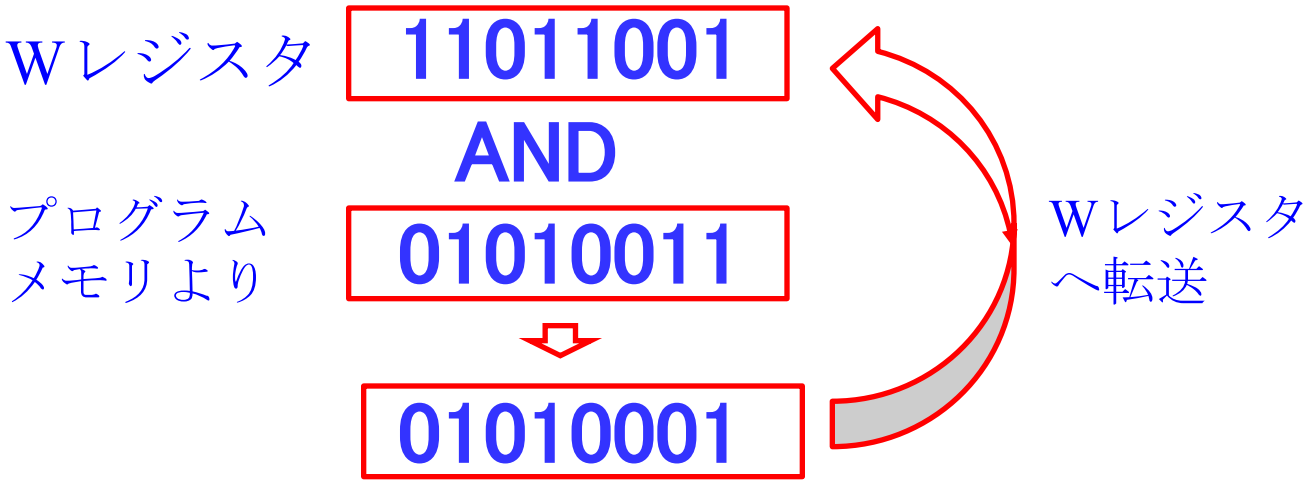
d = 1のとき転送先は f レジスタ，d = 0 のときは wレジスタです．

演習問題７．Wレジスタに0xFFをロードし，これと0x03のAND演算した結果をWレジスタに転送するプログラムを作成せよ．

ANDLW　B’01010011’　　AND Literal and W の略

B’01010011’とWレジスタの中身のANDをとって，その結果をWレジスタに転送させる命令です．

　今，WレジスタにB’11011001’が入っていたとして，上の命令を実行するとWレジスタにはB’01010001’が得られます．



演習問題７．解答

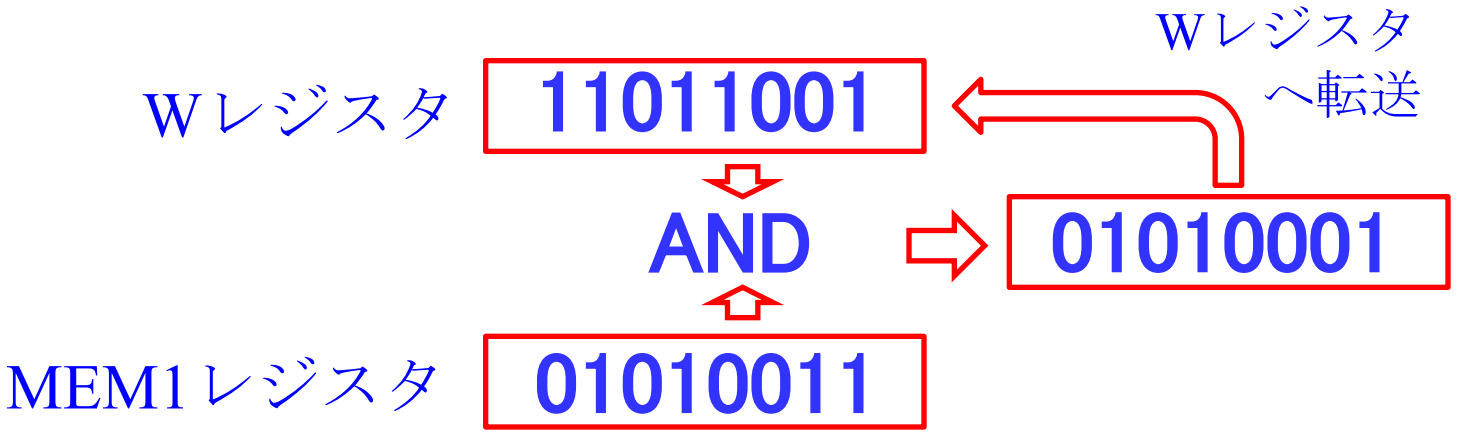
Wレジスタに0xFFをロードし，これと0x03のAND演算（p.60)した結果をWレジスタに転送するプログラムを作成せよ.

```
;Problem 7

        INCLUDE"p16F84A.inc"
        list p=16F84A

        __CONFIG  _HS_OSC  &  _WDT_OFF  &  _PWRTE_OFF  &  _CP_OFF

        ORG     0
        GOTO    START
        ORG     4

START

        MOVLW           0xFF            ; Load 0xFF to W
        ANDLW           0x03            ; W AND 0x03 -> W

        GOTO            START
        END
```

演習問題８．MEM1に0xAA，Wレジスタに0x0Bをロードし，これらのAND演算した結果をWレジスタに転送するプログラムを作成せよ．

ANDWF　f, d　　AND W and f の略

例えば，
ANDWF　MEM1, 0
は，WレジスタにB’11011001’，MEM1にB’01010011’が入っていたとすると，WレジスタとMEM1レジスタの中身のANDをとって，その結果をWレジスタに転送させる命令です．WレジスタにはB’01010001’が得られます．



演習問題８．解答

MEM1に0xAA，Wレジスタに0x0Bをロードし，これらのAND演算した結果をWレジスタに転送するプログラムを作成せよ．

```
;Problem 8
        INCLUDE"p16F84A.inc"
        list p=16F84A

        __CONFIG  _HS_OSC  &  _WDT_OFF  &  _PWRTE_OFF  &  _CP_OFF

MEM1    EQU     0x0C

        ORG     0
        GOTO    START
        ORG     4

START
        MOVLW           0xAA            ; Load 0xAA to W
        MOVWF           MEM1            ; W -> MEM1
        MOVLW           0x0B            ; Load 0x0B to W
        ANDWF           MEM1,0          ; W AND MEM1 -> W

        GOTO            START
        END
```

演習問題９．Wレジスタに0x30をロードし，これと0x03のOR演算した結果をWレジスタに転送するプログラムを作成せよ．

IORLW　B’00000011’　　Inclusive OR Literal and W の略

（通常のOR演算です．Exclusive OR演算との区別をはっきりさせるために，Inclusiveと明記しています．）

演習問題１０．MEM1に0x4A，Wレジスタに0xABをロードし，これらのOR演算した結果をWレジスタに転送するプログラムを作成せよ．

IORWF　f, d　　Inclusive OR W and f の略

演習問題１１．MEM1にF0を転送し，上位４ビットと下位４ビットを交換（SWAPF命令）した結果をWレジスタに転送するプログラムを作成せよ．

SWAPF　f, d　　Swap halves f の略

演習問題９．解答
Wレジスタに0x30をロードし，これと0x03のOR演算した結果をWレジスタに転送するプログラムを作成せよ．

```
;Problem 9

        INCLUDE"p16F84A.inc"
        list p=16F84A

        __CONFIG  _HS_OSC  &  _WDT_OFF  &  _PWRTE_OFF  &  _CP_OFF

        ORG     0
        GOTO    START
        ORG     4

START

        MOVLW           0x30            ; Load 0x30 to W
        IORLW           0x03            ; W OR 0x03 -> W

        GOTO            START
        END
```

演習問題１０．解答

MEM1に0x4A，Wレジスタに0xABをロードし，これらのOR演算した結果をWレジスタに転送するプログラムを作成せよ．

```
;Problem 10

        INCLUDE"p16F84A.inc"
        list p=16F84A

        __CONFIG  _HS_OSC  &  _WDT_OFF  &  _PWRTE_OFF  &  _CP_OFF

MEM1    EQU     0x0C

        ORG     0
        GOTO    START
        ORG     4

START
        MOVLW           0x4A            ; Load 0x4A to W
        MOVWF           MEM1            ; W -> MEM1
        MOVLW           0xAB            ; Load 0xAB to W
        IORWF           MEM1,0          ; W OR MEM1 -> W

        GOTO            START
        END
```

演習問題１１．MEM1にF0を転送し，上位４ビットと下位４ビットを交換(SWAPF命令）した結果をWレジスタに転送するプログラムを作成せよ.

```
;Problem 11

        INCLUDE"p16F84A.inc"
        list p=16F84A

        __CONFIG  _HS_OSC  &  _WDT_OFF  &  _PWRTE_OFF  &  _CP_OFF

MEM1    EQU     0x0C

        ORG     0
        GOTO    START
        ORG     4

START
        MOVLW           0xF0            ; Load 0xF0 to W
        MOVWF           MEM1
        SWAPF           MEM1,0          ; SWAP MEM1 -> W

        GOTO            START
        END
```

2004年8月

著者： 古橋武
名古屋大学工学研究科計算理工学専攻
furuhashi@cse.nagoya-u.ac.jp