

9. 割り込みを学ぼう

回路製作の詳細は[第0章](#)を参照してください.

;Interrupt test program

```
INCLUDE "p16F84.inc"
list p=16F84
```

このソースファイルを各自打ち込んで下さい。

```
__CONFIG _HS_OSC & _WDT_OFF & _PWRTE_OFF & _CP_OFF
```

```
Memory EQU 0x0C
WORK1 EQU Memory+1 ;WORK1 at 0C
TIME1 EQU Memory+2 ;TIME1 at 0D
TIME2 EQU Memory+3 ;TIME2 at 0E
TIME3 EQU Memory+4 ;TIME3 at 0F
```

```
ORG 0
GOTO START ;Main Program starts at START
```

```
ORG 4 ;Sub Program
MOVWF WORK1 ;Save the content of Working Register to WORK1
MOVLW B'00010000' ;'00010000' -> (W)
MOVWF INTCON ;(W) -> (INTCON), Inhibit another interrupt
CALL SUB1 ;SUB1 call
MOVLW B'10010000' ;'10010000' -> (W)
MOVWF INTCON ;(W)->(INTCON), Enable interrupt
MOVF WORK1, 0 ;(WORK1) -> (W)
RETFIE ;Return from interrupt
```

```
START
BSF STATUS, RP0 ;Select Bank1
MOVLW B'00000111' ;'00000111' -> (W)
MOVWF TRISB ;RB0-2: Input port; RB3-7: Output port
MOVLW B'10010000' ;'10010000' -> (W)
MOVWF INTCON ;(W) -> Intcon, Enable interrupt
MOVLW B'00000000'
MOVWF OPTION_REG
BCF STATUS, RP0 ;Select Bank0
```

ソースファイル(続き)

;Main Program

	MOVLW	B'00010000'	; '00010000' -> (W)
STEP1	MOVWF	PORTB	;(W) -> (PORTB), LED1 on
	GOTO	STEP1	

;Sub Program

SUB1	MOVLW	B'01000000'	; '01000000' -> (W)
	MOVWF	PORTB	;(W) -> (PORTB), LED3 on
	CALL	COUNT1	
	RETURN		

COUNT1	MOVLW	0x80
	MOVWF	TIME1
STEPM	MOVWF	TIME2
STEPM1	MOVWF	TIME3
STEPM2	DECFSZ	TIME3,1
	GOTO	STEPM2
	DECFSZ	TIME2,1
	GOTO	STEPM1
	DECFSZ	TIME1,1
	GOTO	STEPM
	RETURN	

END

デモプログラムでは80となっていますが、シミュレーションではくり返し回数が多すぎるので0x03を入力してみてください。

Debugger → Select Tool → MPLAB SIM → Make → View → Special Function Registers → View → File Registers → Stimulus → New Workbook → RB0+Toggle → F7 を押し続けながら ときどき RB0をFire → . . .

The screenshot shows the MPLAB IDE v8.84 interface. The main window displays the assembly code for 'Interrupt_test.asm'. The 'Special Function Registers' window is open, showing the status of various registers. The 'File Registers' window is also open, showing the status of file registers. The 'Stimulus' window is open, showing a table of actions for the RB0 pin. A red box highlights the Stimulus window, and a text box explains the toggle action.

Special Function Registers

Address	SFR Name	Hex	Decimal	Binary
00	WREG	0x03	3	00000011
01	INDF	--	--	--
02	TMR0	0x04	4	00000100
03	PCL	0x00	0	00000000
04	STATUS	0x18	24	00011000
05	FSR	0x00	0	00000000
06	PORTA	0x00	0	00000000
07	PORTB	0x40	64	01000000
08	EEDATA	0x00	0	00000000
09	EEADR	0x00	0	00000000
0A	PCLATH	0x00	0	00000000
0B	INTCON	0x00	0	00000000
81	OPTION_REG	0xFF	255	11111111
85	TRISA	0x1F	31	00011111
86	TRISB	0xFF	255	11111111

File Registers

Address	Hex	Decimal	Binary	Symbol Name
0C	0x10	16	00010000	Memory
0D	0x10	16	00010000	WORK1
0E	0x02	2	00000010	TIME1
0F	0x02	2	00000010	TIME2
10	0x01	1	00000001	TIME3

Stimulus - [Untitled]

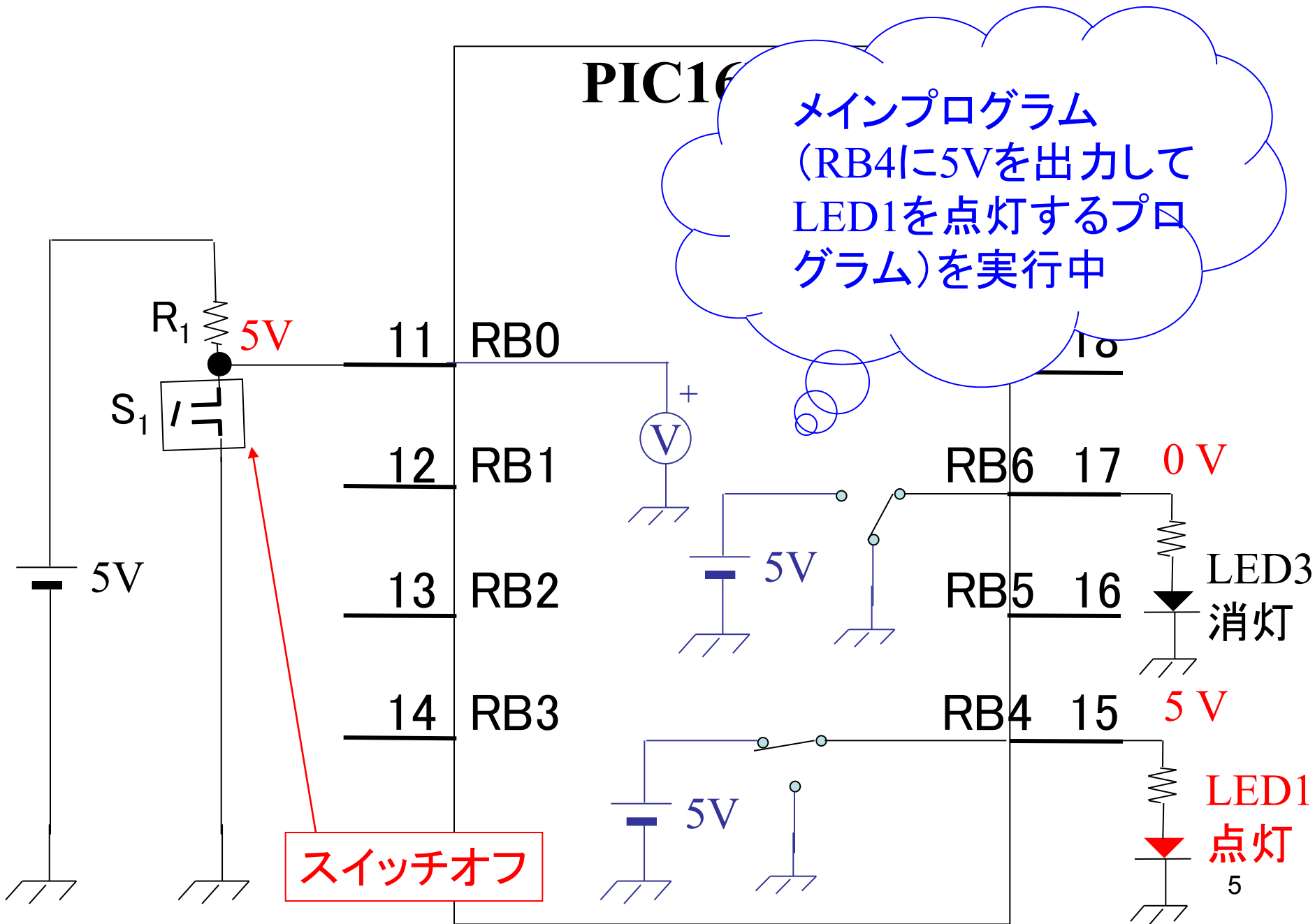
Fire	Pin / SFR	Action	Width	Units	Comments / Message
>	RB0	Toggle			

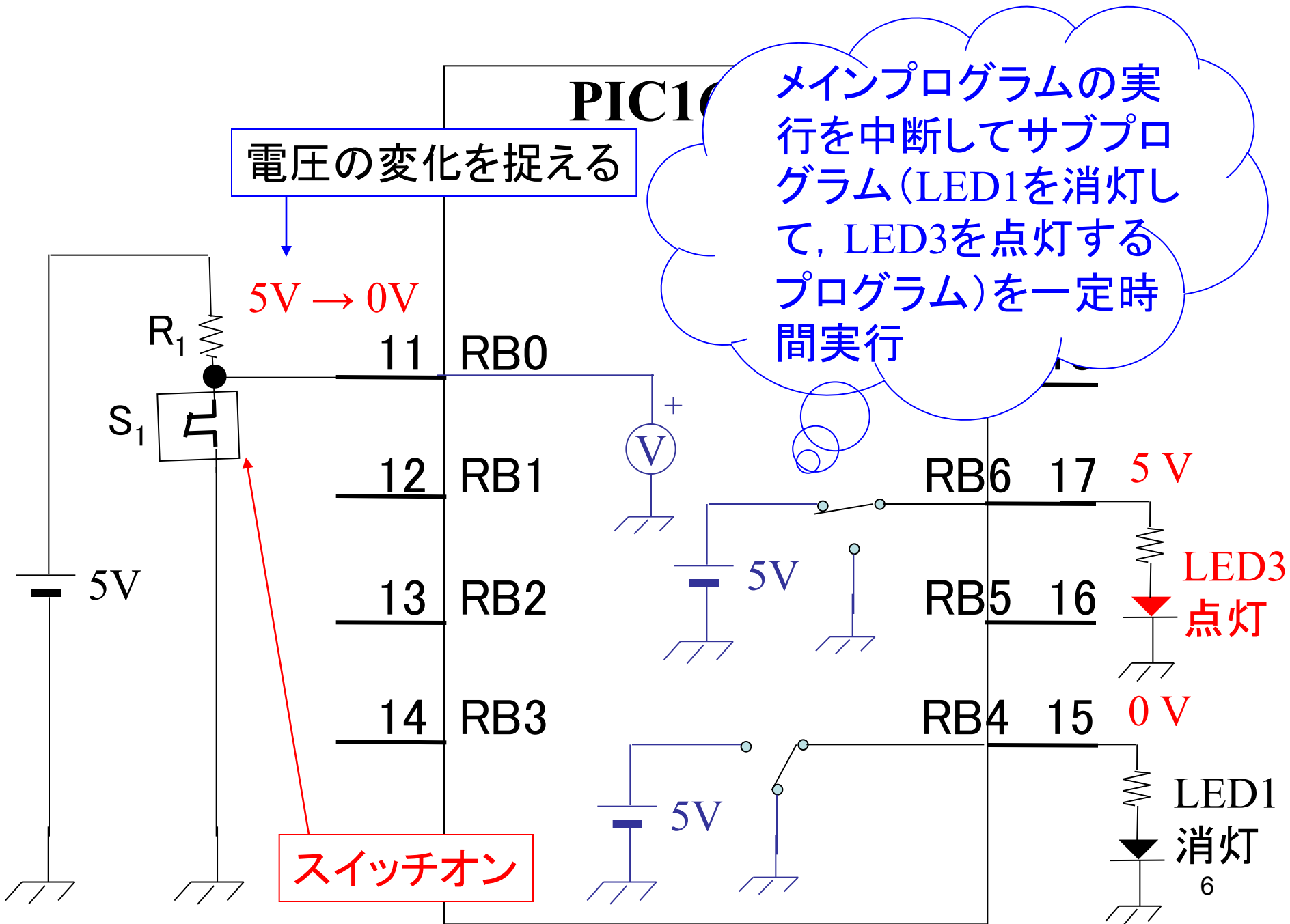
Release build of project C:\Program Files (x86)\Microchip\PIC16F84A_Test
 Language tool versions: MPASMWIN.exe v5.44, mplink.exe v4.42, mplib.exe v4.42
 Thu May 10 13:11:54 2012

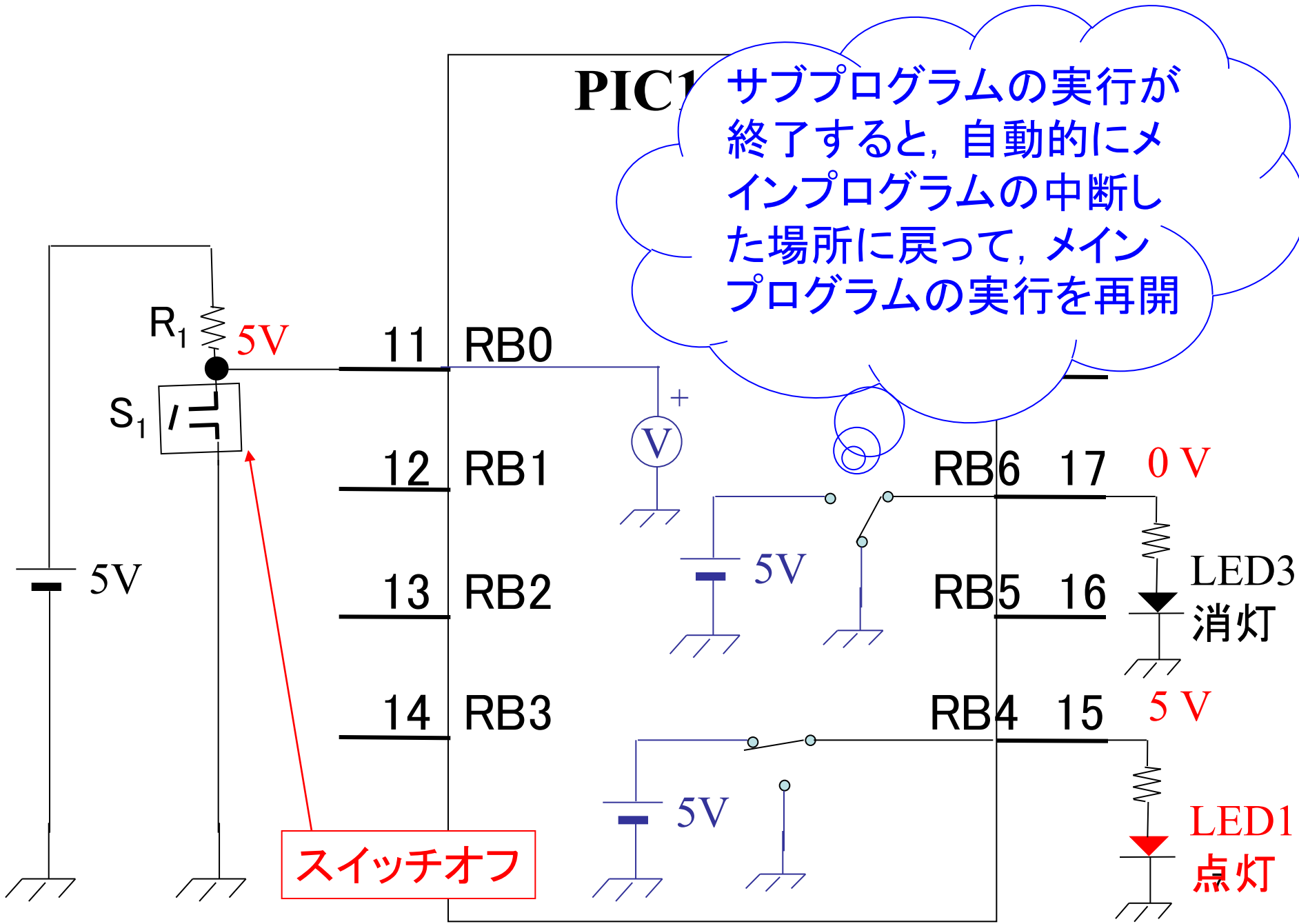
BUILD SUCCEEDED

MPLAB SIM

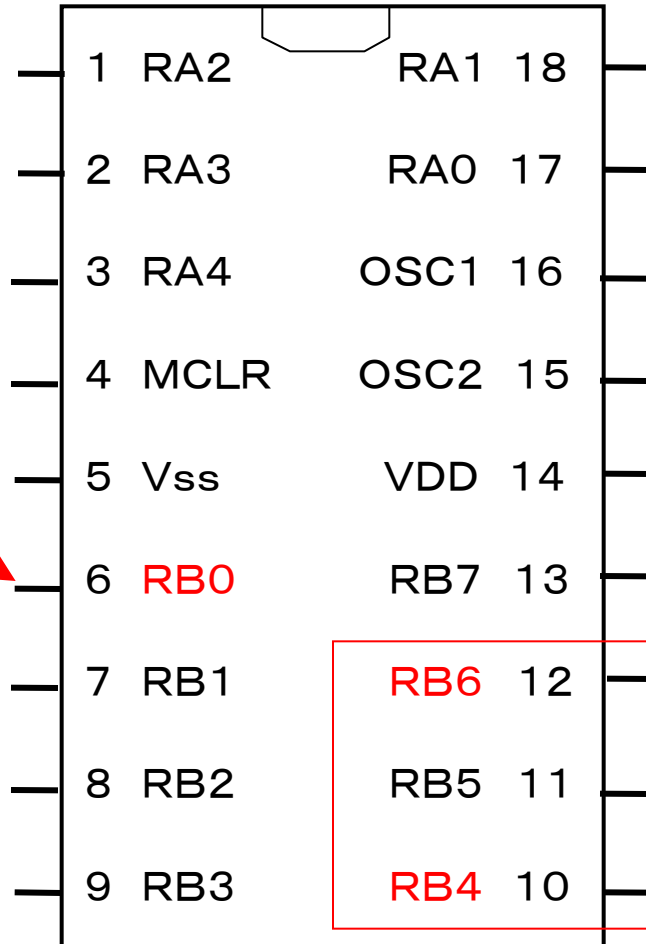
Toggle を選定すると, FireをクリックするたびにPinへの入力が反転する。







PIC16F84



割り込み
入力用端子

出力端子として
利用

LEDを点灯
させる。

本章のポイント

INTCONレジスタ
p.119

B'10010000'
を書き込む
7ビット目：GIE
割り込み許可ビット
1:許可, 0:禁止

4ビット目：RB0
からの割り込みを
許可するビット
1:許可, 0:禁止

バンク0		バンク1	
アドレス(番地)	間接アドレス	間接アドレス	アドレス(番地)
00h			80h
01h	TMRO	OPTION_REG	81h
02h	PCL	PCL	82h
03h	STATUS	STATUS	83h
04h	FSR	FSR	84h
05h	PORTA	TRISA	85h
06h	PORTB	TRISB	86h
07h			87h
08h	EEDATA	EECON1	88h
09h	EEADR	EECON2	89h
0Ah	PCLATH	PCLATH	8Ah
0Bh	INTCON	INTCON	8Bh
0Ch			8Ch
	汎用ファイルレジスタ	汎用ファイルレジスタ	
4Fh			CFh

本章のポイント

OPTIONレジスタ
p.119

B'00000000'を書き
込む

6ビット目：割り込
みエッジ選択ビッ
ト
1:RB0の電圧が0 V
→5 Vに変化したと
き割り込みをかけ
る.
0: RB0の電圧が5 V
→0 Vに変化したと
き割り込みをかけ

図1. 16 ファイルレジスタの配置, p. 15

表1.2 特殊レジスタ一覧(p.16)

アドレス	名 称	ビット7	ビット6	ビット5	ビット4	ビット3	ビット2	ビット1	ビット0
------	-----	------	------	------	------	------	------	------	------

バンク0

00h	INDF	FSRの内容のアドレスのデータメモリ(物理的には存在しない)							
01h	TMR0	8ビットリアルタイム・クロック／カウンタ							
02h	PCL	プログラムカウンタ(PC)の下位8ビット							
03h	STATUS	IRP	RP1	RP0	TO	PD	Z	DC	C
04h	FSR	間接データメモリアドレスポインタ							
05h	PORTA	—	—	—	RA4/T0CKI	RA3	RA2	RA1	RA0
06h	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0/INT
07h		使用しない、「0」としてリードされる							
08h	EEDATA	EEDATAEEPROMデータレジスタ							
09h	EEADR	EEADREEPROMアドレスレジスタ							
0Ah	PCLATH	—	—	—	PCの上位5ビットへの書き込みバッファ				
0Bh	INTCON	GIE	EEIE	TOIE	INTE	RBIE	TOIF	INTF	RBIF

バンク1

80h	INDF	FSRの内容のアドレスのデータメモリ(物理的には存在しない)							
81h	OPTION_REG	RBPU	INTEDG	TOCS	TOSE	PSA	PS2	PS1	PS0
82h	PCL	プログラムカウンタ(PC)の下位8ビット							
83h	STATUS	IRP	RP1	RP0	TO	PD	Z	DC	C
84h	FSR	間接データメモリアドレスポインタ							
85h	TRISA	—	—	—	PORTAデータ入出力設定レジスタ				
86h	TRISB	PORTBデータ入出力設定レジスタ							
87h		使用しない、「0」としてリードされる							
88h	EECON1	—	—	—	EEIF	WRERR	WREN	WR	RD
89h	EECON2	EEPROM制御レジスタ2(物理的には存在しない)							
0Ah	PCLATH	—	—	—	PCの上位5ビットへの書き込みバッファ				
0Bh	INTCON	GIE	EEIE	TOIE	INTE	RBIE	TOIF	INTF	RBIF

 : バンク 0, 1 で共通

;Interrupt test program

```
INCLUDE "p16F84.inc"
list p=16F84
```

```
__CONFIG _HS_OSC & _WDT_OFF & _PWRTE_OFF & _CP_OFF
```

```
Memory EQU 0x0C
WORK1 EQU Memory+1 ;WORK1 at 0C
TIME1 EQU Memory+2 ;TIME1 at 0D
TIME2 EQU Memory+3 ;TIME2 at 0E
TIME3 EQU Memory+4 ;TIME3 at 0F
```

```
ORG 0
GOTO START
```

```
ORG 4
MOVWF WORK1
MOVLW B'00010000'
MOVWF INTCON
CALL SUB1
MOVLW B'10010000'
MOVWF INTCON
MOVF WORK1, 0
RETFIE
```

START

```
BSF STATUS, RP0
MOVLW B'00000111'
MOVWF TRISB
MOVLW B'10010000'
MOVWF INTCON
MOVLW B'00000000'
MOVWF OPTION_REG
BCF STATUS, RP0
```

割り込みプログラム.

RB0に割り込み信号が入ると、そのときのプログラムカウンタの値がスタック(p.12)に格納され、プログラムカウンタ(p.12)には0004番地が書き込まれて、4番地から書かれているこのプログラムが実行される.

RETFIE (割り込みからのReturn)を実行すると、スタックに格納した番地がプログラムカウンタに戻され、割り込みがかかった時のメインプログラムに戻る.

;Interrupt test program

```
INCLUDE"p16F84.inc"
list p=16F84
```

```
__CONFIG _HS_OSC & _WDT_OFF & _PWRTE_OFF & _CP_OFF
```

```
Memory    EQU        0x0C
WORK1     EQU        Memory+1    ;WORK1 at 0C
TIME1     EQU        Memory+2    ;TIME1 at 0D
TIME2     EQU        Memory+3    ;TIME2 at 0E
TIME3     EQU        Memory+4    ;TIME3 at 0F
```

```
ORG        0
GOTO       START
```

```
ORG        4
MOVWF      WORK1
MOVLW      B'00010000'
MOVWF      INTCON
CALL       SUB1
MOVLW      B'10010000'
MOVWF      INTCON
MOVF       WORK1, 0
RETFIE
```

START

```
BSF        STATUS, RP0
MOVLW      B'00000111'
MOVWF      TRISB
MOVLW      B'10010000'
MOVWF      INTCON
MOVLW      B'00000000'
MOVWF      OPTION_REG
BCF        STATUS, RP0
```

PORT Bの設定(p.122)

RB0-RB2: 入力ポート

RB3-RB7:出力ポート

;Interrupt test program

```
INCLUDE "p16F84.inc"
list p=16F84
```

```
__CONFIG _HS_OSC & _WDT_OFF & _PWRTE_OFF & _CP_OFF
```

```
Memory    EQU        0x0C
WORK1      EQU        Memory+1    ;WORK1 at 0C
TIME1      EQU        Memory+2    ;TIME1 at 0D
TIME2      EQU        Memory+3    ;TIME2 at 0E
TIME3      EQU        Memory+4    ;TIME3 at 0F
```

```
ORG        0
GOTO       START
```

```
ORG        4
MOVWF      WORK1
MOVLW      B'00010000'
MOVWF      INTCON
CALL       SUB1
MOVLW      B'10010000'
MOVWF      INTCON
MOVF       WORK1, 0
RETFIE
```

START

```
BSF        STATUS, RP0
MOVLW      B'00000111'
MOVWF      TRISB
MOVLW      B'10010000'
MOVWF      INTCON
MOVLW      B'00000000'
MOVWF      OPTION_REG
BCF        STATUS, RP0
```

B'10010000'をINTCON (Interrupt Control)レジスタ(p.119)に書き込む

7ビット目:割り込み許可ビット
1:許可, 0:禁止

4ビット目:RB0からの割り込みを許可するビット
1:許可, 0:禁止

;Interrupt test program

```
INCLUDE "p16F84.inc"
list p=16F84
```

```
__CONFIG _HS_OSC & _WDT_OFF & _PWRTE_OFF & _CP_OFF
```

```
Memory EQU 0x0C
WORK1 EQU Memory+1 ;WORK1 at 0C
TIME1 EQU Memory+2 ;TIME1 at 0D
TIME2 EQU Memory+3 ;TIME2 at 0E
TIME3 EQU Memory+4 ;TIME3 at 0F
```

```
ORG 0
GOTO START
```

```
ORG 4
MOVWF WORK1
MOVLW B'00010000'
MOVWF INTCON
CALL SUB1
MOVLW B'10010000'
MOVWF INTCON
MOVF WORK1, 0
RETFIE
```

START

```
BSF STATUS, RP0
MOVLW B'00000111'
MOVWF TRISB
MOVLW B'10010000'
MOVWF INTCON
MOVLW B'00000000'
MOVWF OPTION_REG
BCF STATUS, RP0
```

OPTIONレジスタ(p.119)に
B'00000000'を書き込む

6ビット目: 割り込みエッジ選択ビット

1: RB0の電圧が0 V → 5 Vに変化したとき割り込みをかける.

0: RB0の電圧が5 V → 0 Vに変化したとき割り込みをかける

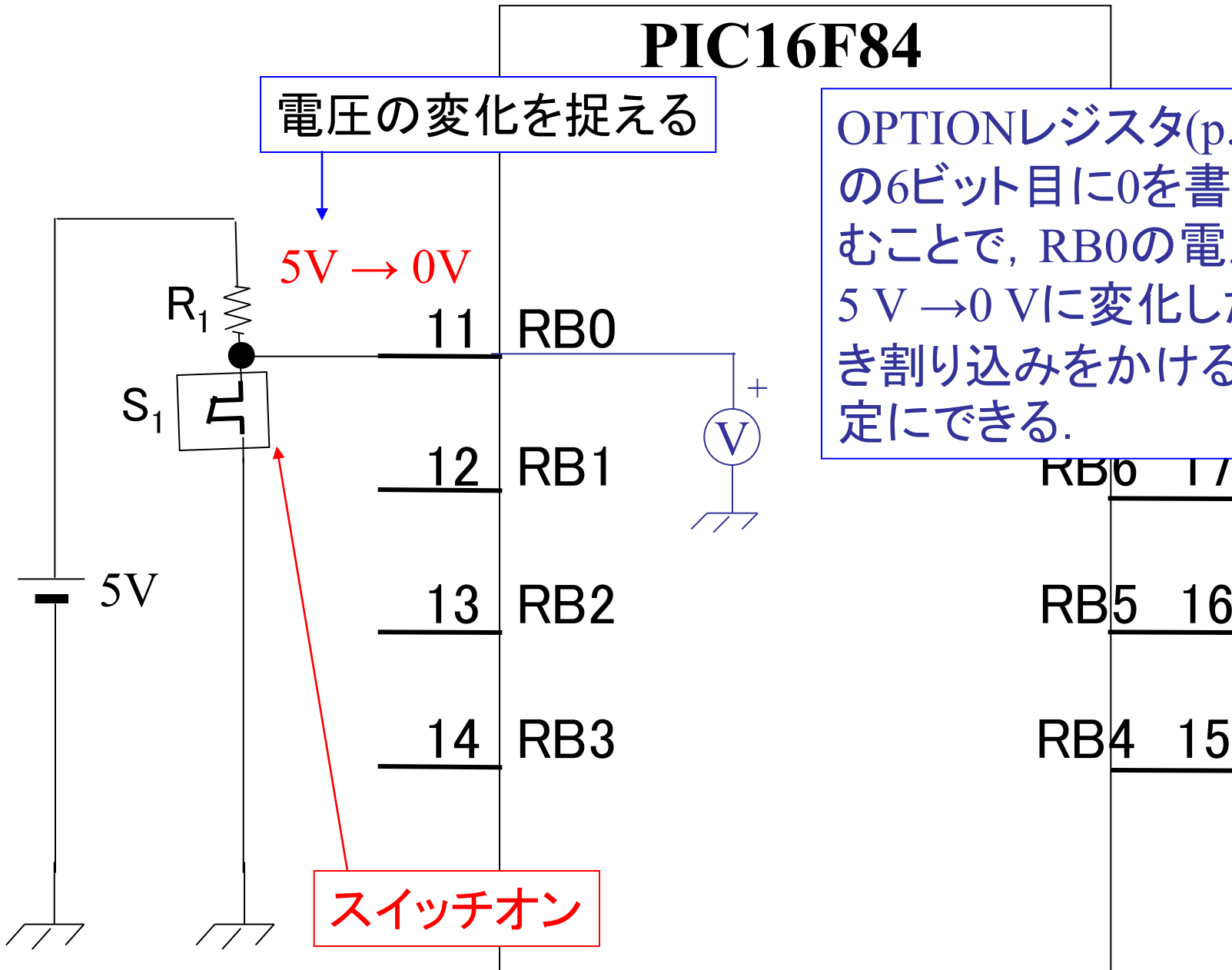
PIC16F84

電圧の変化を捉える

5V → 0V

OPTIONレジスタ(p.119)
の6ビット目に0を書き込
むことで, RB0の電圧が
5 V → 0 Vに変化したと
き割り込みをかける設
定にできる.

スイッチオン



ソースファイル(続き)

;Main Program

```
STEP1    MOVLW    B'00010000'  
          MOVWF    PORTB  
          GOTO     STEP1
```

;Sub Program

```
SUB1      MOVLW    B'01000000'  
          MOVWF    PORTB  
          CALL     COUNT1  
          RETURN
```

```
COUNT1    MOVLW    0x80  
          MOVWF    TIME1  
STEPM     MOVWF    TIME2  
STEPM1    MOVWF    TIME3  
STEPM2    DECFSZ   TIME3,1  
          GOTO     STEPM2  
          DECFSZ   TIME2,1  
          GOTO     STEPM1  
          DECFSZ   TIME1,1  
          GOTO     STEPM  
          RETURN
```

END

メインプログラム
LED1を点灯するという信号を
PORTBに出し続けるプログラム

このメインプログラムを実行中にRB0に割り込み信号が入ると、そのときのプログラムカウンタの値がスタック(p.12)に格納され、プログラムカウンタ(p.12)には0004番地が書き込まれて、4番地から書かれているプログラムが実行される。

;Interrupt test program

```
INCLUDE "p16F84.inc"
list p=16F84
```

割り込み時のメインプログラムのデータの
退避場所.

```
__CONFIG _HS_OSC & _WDT_OFF & _PWRITE_OFF & _CP_OFF
```

```
Memory EQU 0x0C
WORK1 EQU Memory+1 ;WORK1 at 0C
TIME1 EQU Memory+2 ;TIME1 at 0D
TIME2 EQU Memory+3 ;TIME2 at 0E
TIME3 EQU Memory+4 ;TIME3 at 0F
```

```
ORG 0
GOTO START
```

```
ORG 4
MOVWF WORK1
MOVLW B'00010000'
MOVWF INTCON
CALL SUB1
MOVLW B'10010000'
MOVWF INTCON
MOVF WORK1, 0
RETFIE
```

Wレジスタの内容をWORK1に退避させる。
サブプログラムもWレジスタを使用するの
で、メインプログラムで実行していた値を
退避させておかないと、書き換えられてし
まうため。

この他、Wレジスタに限らず、サブプログラムに書き換えられて
は困るものがある場合は、WORK2, WORK3等をファイルレジ
スタに定義しておいて、割り込みがかかったときには、ここに退
避させるようにする。

START

```
BSF STATUS, RP0
MOVLW B'00000111'
MOVWF TRISB
MOVLW B'10010000'
MOVWF INTCON
MOVLW B'00000000'
MOVWF OPTION_REG
BCF STATUS, RP0
```

割り込み処理の終了時には、退避させて
いた、値を元のWレジスタにもどしてから、
メインプログラムに帰っていく。

;Interrupt test program

```
INCLUDE "p16F84.inc"
list p=16F84
```

```
__CONFIG _HS_OSC & _WDT_OFF & _PWRTE_OFF & _CP_OFF
```

```
Memory EQU 0x0C
WORK1 EQU Memory+1 ;WORK1 at 0C
TIME1 EQU Memory+2 ;TIME1 at 0D
TIME2 EQU Memory+3 ;TIME2 at 0E
TIME3 EQU Memory+4 ;TIME3 at 0F
```

```
ORG 0
GOTO START
```

```
ORG 4
MOVWF WORK1
MOVLW B'00010000'
MOVWF INTCON
CALL SUB1
MOVLW B'10010000'
MOVWF INTCON
MOVF WORK1, 0
RETFIE
```

INTCONレジスタの7ビット目を0とすることで、割り込み処理中に、RB0に割り込み信号が入っても、割り込み処理に割り込み処理が入らないようにする。

サブプログラムをコール。

START

```
BSF STATUS, RP0
MOVLW B'00000111'
MOVWF TRISB
MOVLW B'10010000'
MOVWF INTCON
MOVLW B'00000000'
MOVWF OPTION_REG
BCF STATUS, RP0
```

ソースファイル(続き)

;Main Program

```
STEP1    MOVLW    B'00010000'  
         MOVWF    PORTB  
         GOTO     STEP1
```

;Sub Program

```
SUB1     MOVLW    B'01000000'  
         MOVWF    PORTB  
         CALL     COUNT1  
         RETURN
```

```
COUNT1   MOVLW    0x80  
         MOVWF    TIME1  
STEPM    MOVWF    TIME2  
STEPM1   MOVWF    TIME3  
STEPM2   DECFSZ   TIME3,1  
         GOTO     STEPM2  
         DECFSZ   TIME2,1  
         GOTO     STEPM1  
         DECFSZ   TIME1,1  
         GOTO     STEPM  
         RETURN
```

RETURN

END

サブプログラム
LED1を消灯し, LED3
を一定時間点灯する
プログラム

LED1を消灯し, LED3を点灯する信号
をPORTBに出力する.

LED1を消灯し, LED3を点灯する信号
をPORTBに出力する.

一定時間, 時間を稼ぐプログラム

ソースファイル(続き)

;Main Program

```
STEP1    MOVLW    B'00010000'  
         MOVWF    PORTB  
         GOTO     STEP1
```

;Sub Program

```
SUB1     MOVLW    B'01000000'  
         MOVWF    PORTB  
         CALL     COUNT1  
         RETURN
```

```
COUNT1   MOVLW    0x80  
         MOVWF    TIME1  
STEPM    MOVWF    TIME2  
STEPM1   MOVWF    TIME3  
STEPM2   DECFSZ   TIME3,1  
         GOTO     STEPM2  
         DECFSZ   TIME2,1  
         GOTO     STEPM1  
         DECFSZ   TIME1,1  
         GOTO     STEPM  
         RETURN
```

END

サブプログラム
LED1を消灯し, LED3
を一定時間点灯する
プログラム

初めにTIME1,2,3に0x80を入れる. TIME3を一つずつ減らしていき, 0になったら, TIME2を一つ減らして, TIME3に0x80を入れて, 再び, TIME3を一つずつ減らしていき, 0になったら, TIME2を一つ減らして, TIME3に0x80を入れて, 再び……と, くり返し, やがて, TIME2が0になったら, TIME1を一つ減らしてTIME2, 3に0x80を入れて……とくり返す. TIME1が0になったら終了.
全部で80×80×80回のくり返し演算を行う.

;Interrupt test program

```
INCLUDE "p16F84.inc"
list p=16F84
```

```
__CONFIG _HS_OSC & _WDT_OFF & _PWRTE_OFF & _CP_OFF
```

```
Memory    EQU        0x0C
WORK1     EQU        Memory+1    ;WORK1 at 0C
TIME1     EQU        Memory+2    ;TIME1 at 0D
TIME2     EQU        Memory+3    ;TIME2 at 0E
TIME3     EQU        Memory+4    ;TIME3 at 0F
```

```
ORG        0
GOTO       START
```

```
ORG        4
MOVWF      WORK1
MOVLW      B'00010000'
MOVWF      INTCON
CALL       SUB1
MOVLW      B'10010000'
MOVWF      INTCON
MOVF       WORK1, 0
RETFIE
```

割り込み処理が終了したので、INTCONレジスタの7ビット目を1として、割り込み処理を可として、メインプログラムに戻るようになる。

START

```
BSF        STATUS, RP0
MOVLW      B'00000111'
MOVWF      TRISB
MOVLW      B'10010000'
MOVWF      INTCON
MOVLW      B'00000000'
MOVWF      OPTION_REG
BCF        STATUS, RP0
```

割り込み処理の終了時には、退避させていた、値を元のWレジスタにもどしてから、メインプログラムに帰っていく。

演習問題21. メインプログラムにおいてW, MEM1レジスタを使うプログラムを書け. そして, 割り込み処理が発生した場合に, これらW, MEM1レジスタをWORK1, WORK2に退避するプログラムを書け.

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