Lec 3 :TMR0

TIMER0 (TMR0)

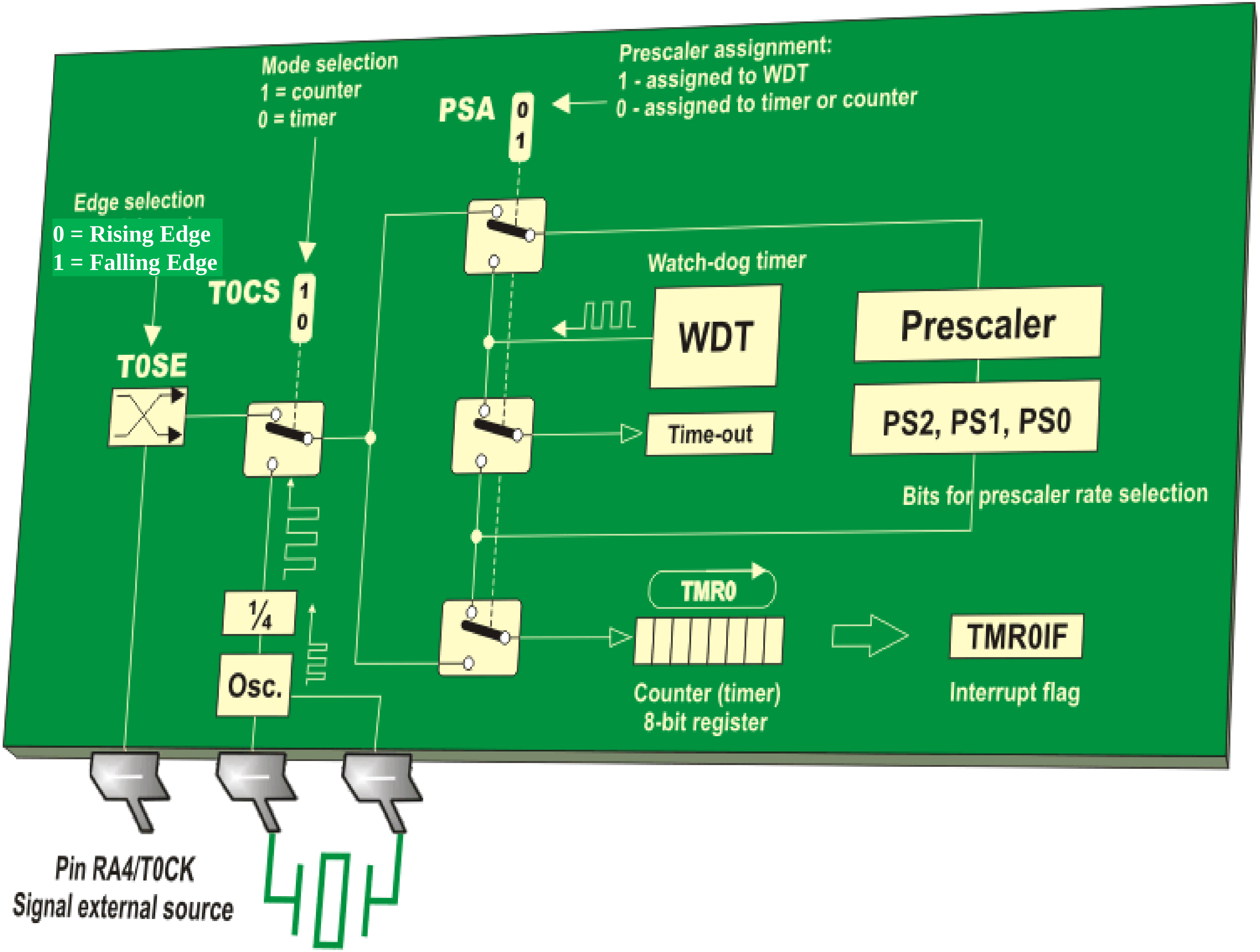
- Timer0 is an 8-bit Timer/Counter module (0→255) with the following features:
  1. 8-bit prescaler (shared with WDT).
  2. Selectable internal or external clock source.
  3. Interrupt on overflow (255→0).
  4. Source edge selection (positive or negative going edge).
- To configure the Timer0 module the **OPTION_REG** Special Function Register (SFR) is used.

Using OPTION_REG Register to Configure TMR0

- The OPTION_REG register contains various control bits to configure **TMR0/WDT** prescaler, timer TMR0, external interrupt and pull-ups on PORTB.

OPTION_REG Register

	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	Features
OPTION	RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	Bit name
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	



Mode selection
1 = counter
0 = timer
PSA
0
1
Prescaler assignment:
1 - assigned to WDT
0 - assigned to timer or counter
Edge selection
0 = Rising Edge
1 = Falling Edge
T0CS
1
0
T0SE
Watch-dog timer
WDT
Prescaler
Time-out
PS2, PS1, PS0
Bits for prescaler rate selection
1/4
Osc.
TMR0
Counter (timer)
8-bit register
TMR0IF
Interrupt flag
Pin RA4/T0CK
Signal external source

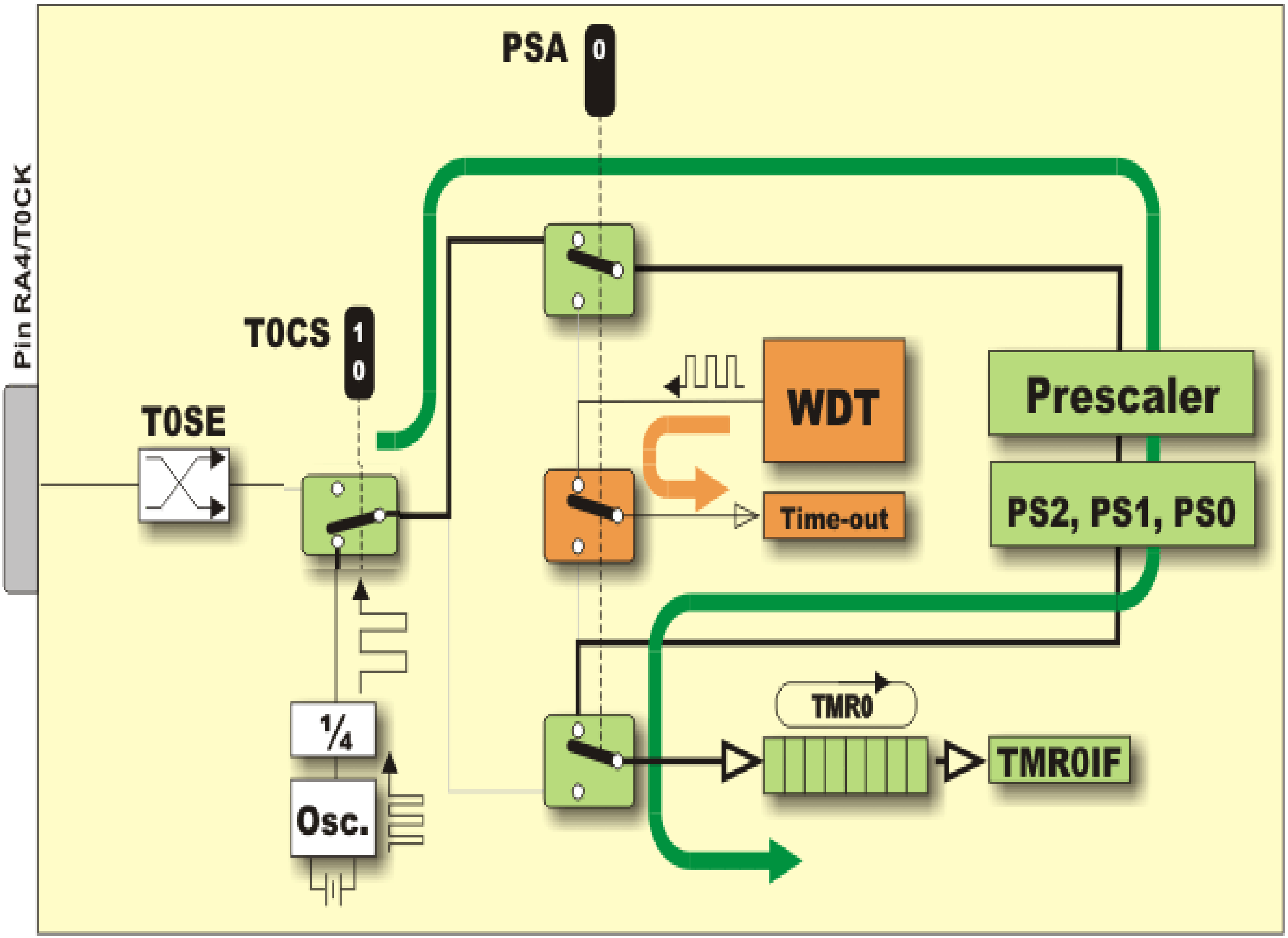
OPTION_REG Register

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	
	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	Features
OPTION	RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	Bit name

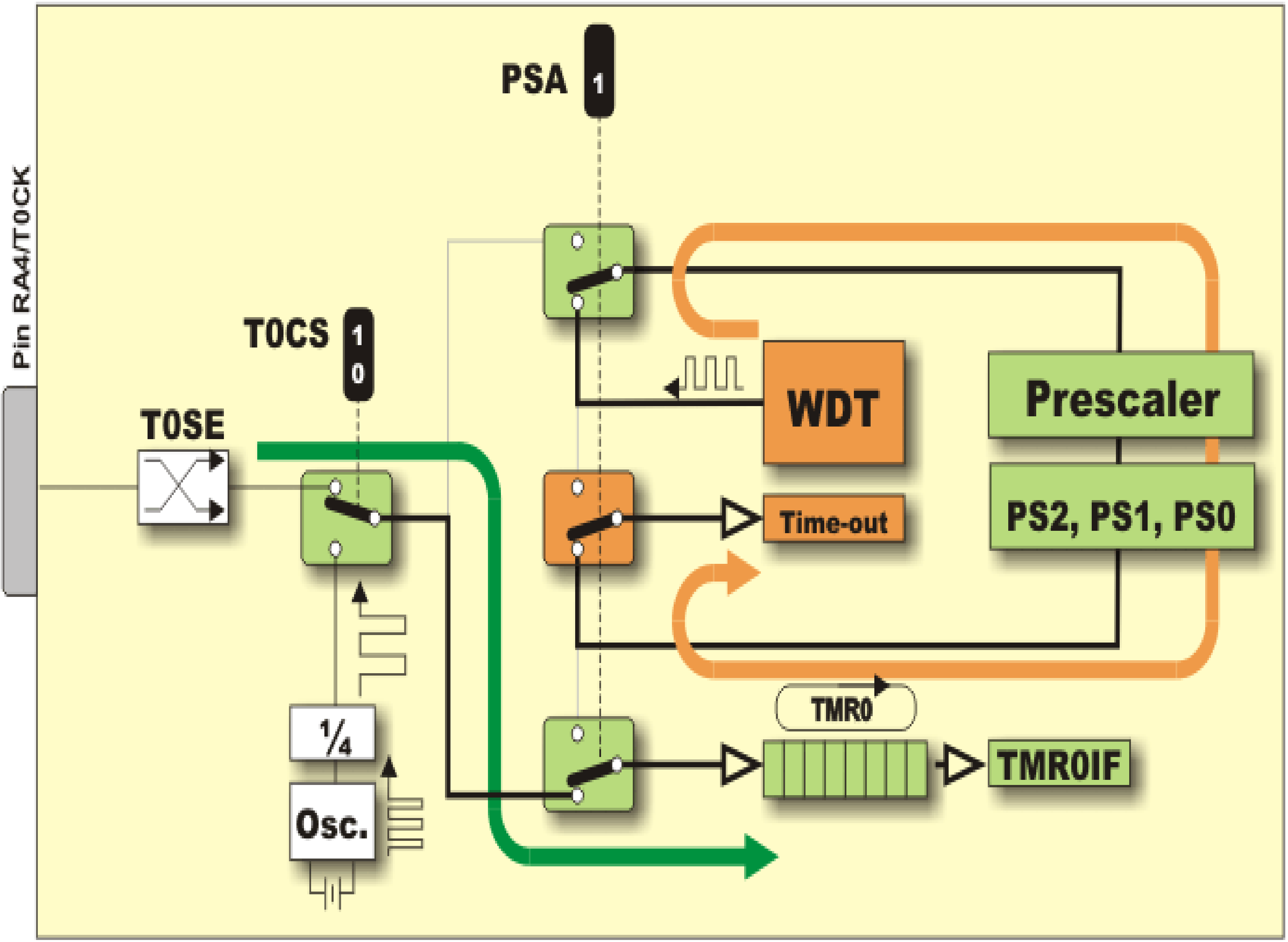
- **PSA (Prescaler Assignment bit):**

assigns prescaler (only one exists) to the timer or watchdog timer.

- **1** - Prescaler is assigned to the WDT.
- **0** - Prescaler is assigned to the TMR0.
- IF PSA is set (1), prescaler is assigned to watchdog timer and PS2:PS0 have no effect (TMR0 rate = 1:1).



Pin RA4/T0CK
T0SE
T0CS
1
0
PSA
0
WDT
Time-out
Prescaler
PS2, PS1, PS0
TMR0
TMR0IF
1/4
Osc.



PSA
1
T0CS
1
0
T0SE
Pin RA4/T0CK
WDT
Prescaler
Time-out
PS2, PS1, PS0
TMR0
TMR0IF
1/4
Osc.

OPTION_REG Register

	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	Features
OPTION	RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	Bit name
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	

➢ **PS2, PS1, PS0 (Prescaler Rate Select bits):**

- Prescaler rate is selected by combining these three bits. As shown in the following table, prescaler rate depends on whether prescaler is assigned to the timer (TMR0) or watch-dog timer (WDT).
- IF PSA is set (1), prescaler is assigned to watchdog timer and PS2:PS0 have no effect (TMR0 rate = 1:1).

PS2	PS1	PS0	TMR0	WDT
0	0	0	1:2	1:1
0	0	1	1:4	1:2
0	1	0	1:8	1:4
0	1	1	1:16	1:8
1	0	1	1:64	1:32
1	1	0	1:128	1:64
1	1	1	1:256	1:128

IF PSA =1 (prescaler is assigned to watchdog timer), TMR0 rate = 1:1

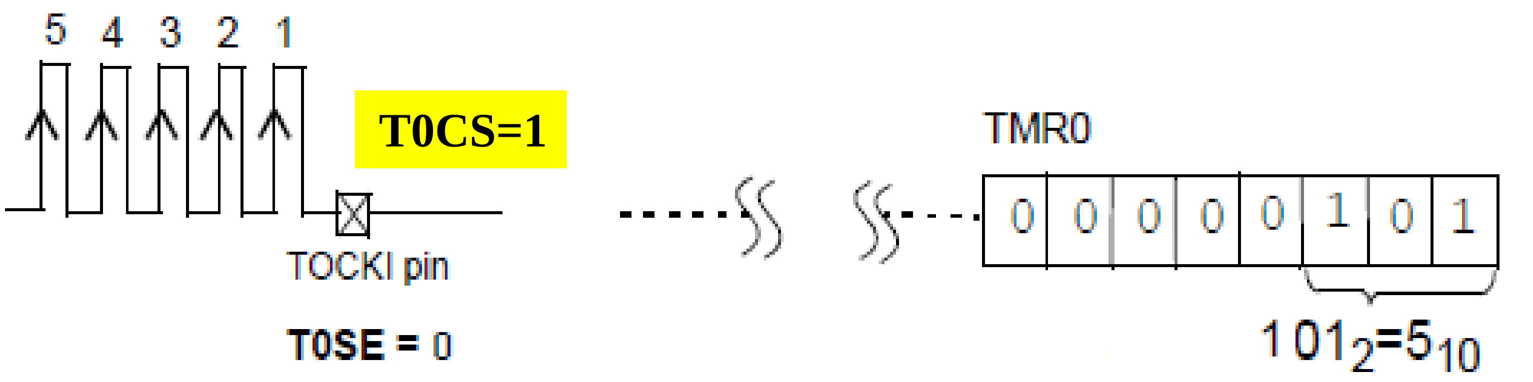
OPTION_REG Register

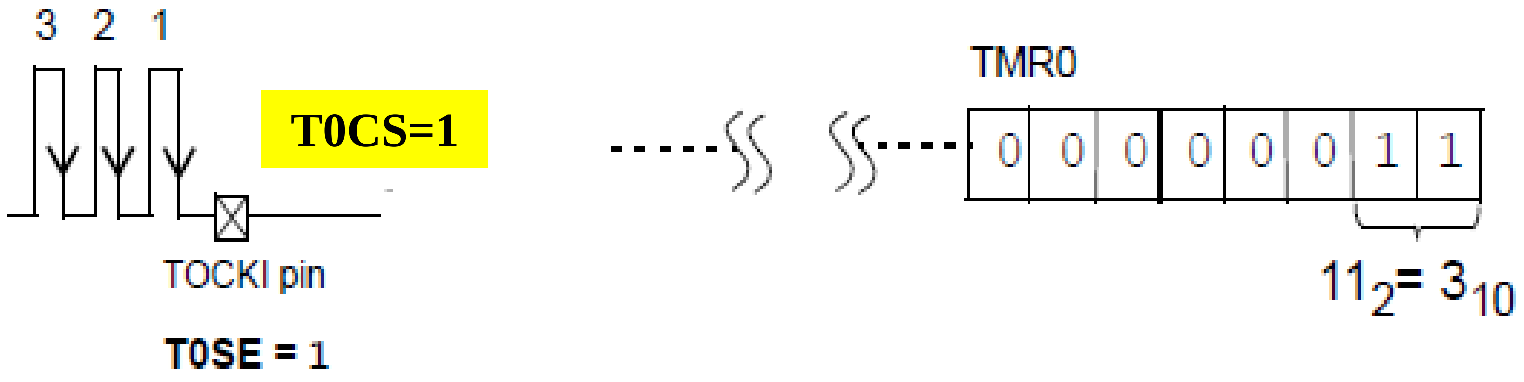
	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	Features
OPTION	RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	Bit name
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	

- **T0SE** **(Timer0 Source Edge Select bit):**

- This bit has an effect **only when an external clock source is used** (T0CKI = clock from **RA4 pin**)
- 1 = TMR0 register increments on **high-to-low** transition on T0CKI pin
- 0 = TMR0 register increments on **low-to-high** transition on T0CKI pin

Effect of Timer0 Source Edge Select Bit



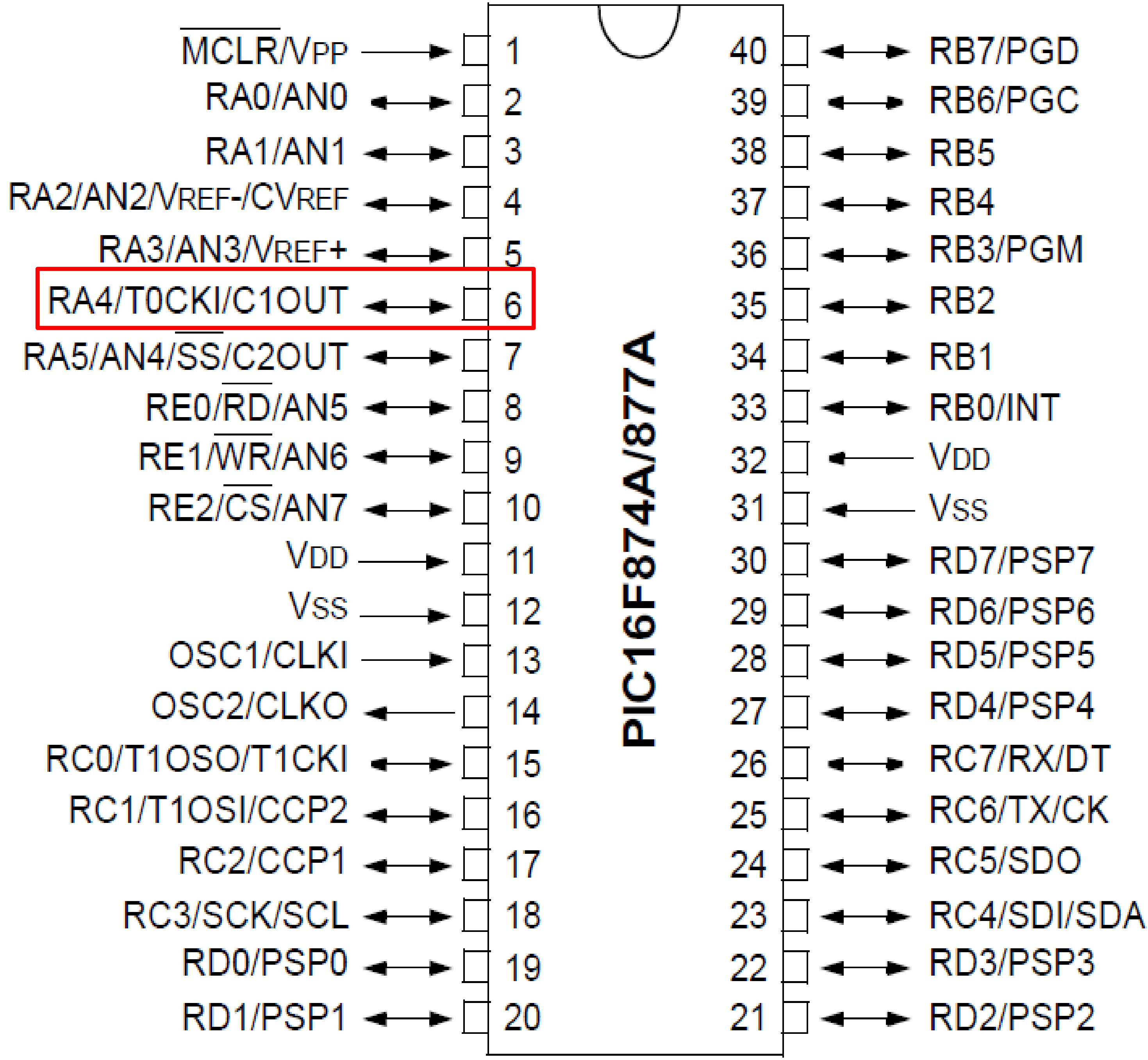


OPTION_REG Register

	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	Features
OPTION	RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	Bit name
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	

- **T0CS (Timer0 Clock Source Select bit):**

- 1 = Signal present on **T0CKI** (Timer0 Clock Input) pin (**RA4 pin**) used as Timer0 clock source.
- 0 = Internal instruction cycle clock used as source.
- Internal instruction cycle = (Microcontroller oscillator Frequency) / 4
- These timers run at a speed of 1/4 of the clock speed. So, if we use 4 MHz crystal, the internal timer will run at 1 MHz.


MCLR/VPP
1
RA0/AN0
2
RA1/AN1
3
RA2/AN2/VREF-/CVREF
4
RA3/AN3/VREF+
5
RA4/T0CKI/C1OUT
6
RA5/AN4/SS/C2OUT
7
RE0/RD/AN5
8
RE1/WR/AN6
9
RE2/CS/AN7
10
VDD
11
VSS
12
OSC1/CLKI
13
OSC2/CLKO
14
RC0/T1OSO/T1CKI
15
RC1/T1OSI/CCP2
16
RC2/CCP1
17
RC3/SCK/SCL
18
RD0/PSP0
19
RD1/PSP1
20
PIC16F874A/877A
40
RB7/PGD
39
RB6/PGC
38
RB5
37
RB4
36
RB3/PGM
35
RB2
34
RB1
33
RB0/INT
32
VDD
31
VSS
30
RD7/PSP7
29
RD6/PSP6
28
RD5/PSP5
27
RD4/PSP4
26
RC7/RX/DT
25
RC6/TX/CK
24
RC5/SDO
23
RC4/SDI/SDA
22
RD3/PSP3
21
RD2/PSP2

TMR0

1 **TMR0 As A Timer**

2 **TMR0 As A Counter**

1. TMR0 As A Timer

OPTION_REG Register

	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	Features
OPTION	RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	Bit name
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	

0	0	0	0	0			

1 / 0
according to
prescaler rate

2. TMR0 As A Counter

OPTION_REG Register

	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	Features
OPTION	RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	Bit name
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	

0	0	1	0	1	0	0	0

(1) TMR0 As A Timer

TMR0 Overflow Time Out (Time to overflow)

- Overflow Time of TMR0 = prescale x 4 x clock_period x (256 – T0)
  - T0 is the initial value assigned to TMR0
  - clock_period = 1/f *Sec*
- IF T0 =0 , overflow Time of TMR0 is the maximum possible value and is equal to:

  Max. overflow Time of TMR0 = prescale x 4 x clock_period x 256
- When overflow occurs (TMR0 value change from 255→0), the Timer0 Interrupt Flag bit T0IF (INTCON.T0IF) is set to 1.

- **Example1:**

If (1:256) Prescaler is assigned to TMR0 with f = 4MHz find:

1- The overflow time of TMR0

2- No. of required overflows to get time equal to 1 sec.

Assume two cases:

- Case1: The initial value of TMR0 =0 (TMR0 =0;)
- Case2:The initial value of TMR0 =39 (TMR0 =39;)

Case1: The initial value of TMR0 =0 (T0 =0)

- overflow Time of TMR0 = (256-T0) x prescale x 4 x clock_period

$= 256 \times 256 \times 4 \times 0.25*10^{-6}$ sec $= 65536\ \mu Sec$

$$1 \text{ overflow} \longrightarrow 65536\ \mu\text{Sec}$$

$$X \text{ overflows} \longrightarrow 1 \text{ Sec}$$

$$X = \frac{1}{65536 * 10^{-6}} = 15.2587 = 16$$

No. of required overflows to get time equal to 1 sec = 16 overflows.

Case2: The initial value of TMR0 =39 (T0 =39)

- overflow Time of TMR0 = (256-T0) x prescale x 4 x clock_period = (256-39) x 256 x 4 x $0.25*10^{-6}$ sec = 55552 μSec

1 overflow $\longrightarrow$ 55552 μSec

X overflows $\longrightarrow$ 1 Sec

$$X = \frac{1}{55552 * 10^{-6}} = 18$$

No. of required overflows to get time equal to 1 sec = 18 overflows.

- **Example 2:**

If (1:256) Prescaler is assigned to TMR0 with f = 4MHz find:

1- The overflow time of TMR0

2- No. of required overflows to get time equal to 1.5 sec.

Assume two cases:

- Case1: The initial value of TMR0 =0 (TMR0 =0;)
- Case2:The initial value of TMR0 =39 (TMR0 =39;)

Case1: The initial value of TMR0 =0 (T0 =0)

- overflow Time of TMR0 = 256 x prescale x 4 x clock_period

$= 256 \times 256 \times 4 \times 0.25 * 10^{-6}$ sec = 65536 μSec

1 overflow $\longrightarrow$ 65536 μSec

X overflows $\longrightarrow$ 1.5 Sec

$$X = \frac{1.5}{65536 * 10^{-6}} = 22.89 = 23$$

No. of required overflows to get time equal to 1.5 sec = 23 overflows.

Case2: The initial value of TMR0 =39 (T0 =39)

- overflow Time of TMR0 = (256-T0) x prescale x 4 x clock_period = (256-39) x 256 x 4 x $0.25*10^{-6}$ sec = 55552 μSec

1 overflow $\longrightarrow$ 55552 μSec

X overflows $\longrightarrow$ 1.5 Sec

$$X = \frac{1.5}{55552 * 10^{-6}} = 27$$

No. of required overflows to get time equal to 1.5 sec = 27 overflows.

- **Example 3:**

Write a PIC16F877A C program to toggle all bits of PORTB every 1 sec using timer0. Assign (1:256) prescaler to TMR0 with $f = 4MHz$.

➢ **OPTION_REG Configuration:**

OPTION_REG Register

	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	Features
OPTION	RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	**Bit name**
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	

0	**0**	**0**	**0**	**0**	**1**	**1**	**1**

OPTION_REG = 0x07;

➢ No. of required overflows to get time equal to 1 sec

- Assume The initial value of TMR0 =0 (T0 =0)
- overflow Time of TMR0 = (256-T0) x prescale x 4 x clock_period

$= 256 \times 256 \times 4 \times 0.25*10^{-6}$ sec = 65536 μSec

1 overflow $\longrightarrow$ 65536 μSec

X overflows $\longrightarrow$ 1 Sec

$$X = \frac{1}{65536 * 10^{-6}} = 15.2587 = 16$$

No. of required overflows to get time equal to 1 sec = 16 overflows.

```
void main() {
    int overflows_no = 0;
    TRISB = 0x00;         // PORTB is output
    PORTB = 0X55;     // Initialize PORTB
    OPTION_REG = 0x07;       // Prescaler  (1:256) is assigned to the timer TMR0
    TMR0 = 0;                              // Initial value of Timer0  ( T0 )

    while(1) {

        if (INTCON.TMR0IF = = 1) {                // check if  (1) overflow occurs
            overflows_no++;           // 1 overflow occurs causes counter to be incremented by 1
            INTCON.TMR0IF = 0;
          //    TMR0 = T0;        ( initial value of timer/counter TMR0 )
        }

        if (overflows_no = = 16)  {
           PORTB = ~PORTB;       // Toggle PORTB every 1 Sec
           overflows_no = 0;
        }
     }
}
```

(2) TMR0 As A Counter

TMR0 As A Counter

OPTION_REG Register

	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	R/W (1)	Features
OPTION	RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0	Bit name
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	

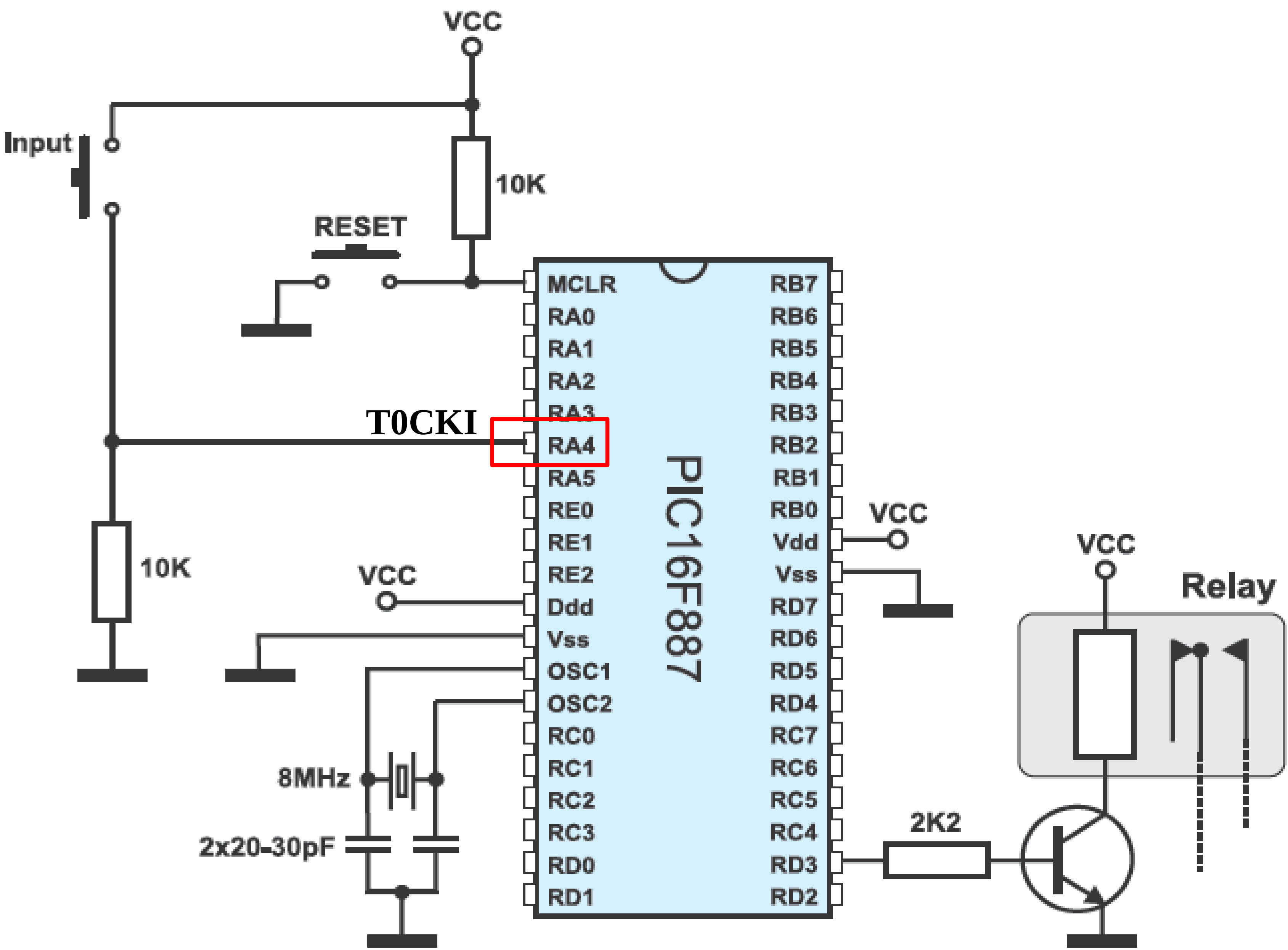
		Bit 5	Bit 4	Bit 3			
0	**0**	**1**	**0**	**1**	**0**	**0**	**0**

OPTION_REG Register

- T0CS (Bit 5) must be = 1 (external clock source from **T0CKI** (**RA4 pin**) must be used to be counted)
- T0SE (Bit 4) = 1 /0 (counting on falling edge / rising edge clock change)
- PSA (Bit 3) must be = 1 (TMR0 rate = 1:1)

EX2: TMR0 As A Counter

- Write a microcontroller PIC16F877A C program to use TMR0 as a counter such that:
  - The counter input is connected to a push button so that any button press causes timer TMR0 to count one pulse.
  - When the number of pulses matches the number stored in the TEST variable, a logic one (5v) appears on the RD3 pin.
  - This voltage activates an electromechanical relay, and this bit is called ‘Relay’ in the program.


VCC
Input
10K
RESET
T0CKI
MCLR
RA0
RA1
RA2
RA3
RA4
RA5
RE0
RE1
RE2
Ddd
Vss
OSC1
OSC2
RC0
RC1
RC2
RC3
RD0
RD1
RB7
RB6
RB5
RB4
RB3
RB2
RB1
RB0
Vdd
Vss
RD7
RD6
RD5
RD4
RC7
RC6
RC5
RC4
RD3
RD2
PIC16F887
10K
VCC
VCC
VCC
Relay
8MHz
2x20-30pF
2K2

```
sbit Relay at PORTD.B3;    // RD3 is called Relay
void main() {
   int TEST = 5;                                   // Variable TEST = 5
   TRISA = 0xFF;                              // All portA pins are configured as inputs
   TRISD = 0;   // Pin RD3 is configured as an output
   PORTD = 0;                // Reset port D ( initial state of the motor is off )
   OPTION_REG.F5 = 1;         // Counter TMR0 receives pulses through the RA4 pin (T0CKI)
   OPTION_REG.F4 = 0;        // counting on rising edge
   OPTION_REG.F3 = 1;         // Prescaler rate is 1:1
   TMR0 = 0;                           // Initial value of timer/counter TMR0
    while(1) {
         if (TMR0 = = TEST) {              // Does the number in timer match constant TEST?
                 Relay = 1;      // (PORTD.B3=1) Numbers match. Set the RD3 bit (output RELAY)
         }
    }                              // Remain in endless loop
}
```

sbit Data Type

- When we declare a sbit variable, it points to a specific bit in registers, Special Function Register (SFR) or variables.
- The declaration should be done before the main function (as a global variable).
- Declaring a sbit variable is not possible via F0, F1, … F15 identifiers.

sbit Motor at PORTB.F1; // is not allowed

EX1:

```
  sbit Motor at PORTB.B1;      // variable Motor is used to point to RB1
  sbit LS at PORTB.b2;         // variable LS is used to point to RB2
  ...............
void main()
{
  ...............
}
```

EX2:

```
  char temp;
  sbit MSB at temp.b7;    // variable MSB is used to point to bit 7 in variable temp
  ...............
void main()
{
  ...............
}
```