

M32C/84, 85, 86, 87, 88 Group

Watchdog Timer

1. Abstract

The watchdog timer function allows users to detect when a program is out of control. Users can select whether to generate a watchdog timer interrupt or reset as an operation when the watchdog timer count value underflows.

For the watchdog timer count source, users can select the CPU clock or on-chip oscillator clock.

When the CPU clock is used for the count source, the watchdog timer periods are as follows:

When the main clock, PLL clock, or on-chip oscillator clock is used as the CPU clock:

$$\text{Watchdog timer period} = \frac{\text{Division ratio by prescaler (16 or 128)} \times \text{Watchdog timer count value (32768)}}{\text{CPU clock}}$$

When the sub-clock is selected as the CPU clock:

$$\text{Watchdog timer period} = \frac{\text{Division ratio by prescaler (2)} \times \text{Watchdog timer count value (32768)}}{\text{CPU clock}}$$

When the on-chip oscillator clock is selected for the watchdog timer count source, the watchdog timer period is as follows:

$$\text{Watchdog timer period} = \frac{\text{Watchdog timer count value (32768)}}{\text{On-chip oscillator clock}}$$

Table 1.1 lists a Watchdog Timer Period Example.

Table 1.1 Watchdog Timer Period Example

When XIN = 8 MHz, XCIN = 32.768 kHz, PLL clock = 32 MHz (main clock multiplied-by-4)

| Setting Value | | | | | | | Watchdog Timer Count Source | Watchdog Timer Period |
|---------------|------|------|------|------|------|---------|------------------------------------------------------|--------------------------|
| CM07 | CM17 | CM21 | PM22 | PM24 | WDC7 | MCD | | |
| 0 | 0 | 0 | 0 | 0 | 0 | 12h | Divided-by-16 CPU clock (main clock) | Approx. 65.5 ms |
| 0 | 0 | 0 | 0 | 0 | 1 | 12h | Divided-by-128 CPU clock (main clock) | Approx. 524.3 ms |
| 0 | 0 | 0 | 0 | 1 | 0 | (NOTE) | Divided-by-16 CPU clock (main clock) | Approx. 65.5 ms |
| 0 | 0 | 0 | 0 | 1 | 1 | (NOTE) | Divided-by-128 CPU clock (main clock direct mode) | Approx. 524.3 ms |
| 0 | 1 | 0 | 0 | 0 | 0 | 12h | Divided-by-16 CPU clock (PLL clock) | Approx. 16.4 ms |
| 0 | 1 | 0 | 0 | 0 | 1 | 12h | Divided-by-128 CPU clock (PLL clock) | Approx. 131.1 ms |
| 0 | 0 | 1 | 0 | 0 | 0 | 12h | Divided-by-16 CPU clock (on-chip oscillator) | Approx. 524.3 ms |
| 0 | 0 | 1 | 0 | 0 | 1 | 12h | Divided-by-128 CPU clock (on-chip oscillator) | Approx. 4.2 ms |
| 1 | 0 | 0 | 0 | 0 | - | (NOTE) | Divided-by-2 CPU clock (sub clock) | 2s |
| - | - | - | 1 | - | - | (NOTE) | On-chip oscillator | Approx. 32.8 ms |

-: Can be set to either 0 or 1.

CM07: Bit in the CM0 register, CM17: Bit in the CM1 register, CM21: Bit in the CM2 register

PM22: Bit in the PM2 register, WDC7: Bit in the WDC register

NOTE:

1. The MCB register value does not affect the watchdog timer period.

2. Introduction

The application example described in this document is applied to the following MCUs and parameter(s):

MCUs: M32C/84 Group
M32C/85 Group
M32C/86 Group
M32C/87 Group
M32C/88 Group

This program can be used with other M16C Family MCUs which have the same special function registers (SFRs) as the above MCUs. Check the manual for any additions and modifications to functions. Careful evaluation is recommended before using this application note.

3. Application Example

This section describes how to generate the watchdog timer interrupt or reset using the CPU clock for the watchdog timer counter source.

3.1 Example Description

- (1) When the WDTS register is written, the watchdog timer is initialized to 7FFFh and starts counting.
- (2) When the WDTS register is rewritten during a count operation, the watchdog timer is initialized to 7FFFh and continues counting.
- (3) The watchdog timer holds its value and stops when in wait mode, stop mode or in hold. After exiting from wait mode or stop mode, the watchdog timer restarts counting (see NOTE below).
- (4) To use the watchdog timer interrupt:
When the watchdog timer underflows, the watchdog timer is initialized to 7FFFh and continues counting. At the same time, the watchdog timer interrupt is generated.
To use the watchdog timer reset:
When the watchdog timer underflows, the pins, CPU and SFR are initialized and the MCU starts the program beginning with the address indicated by the reset vector.

NOTE:

1. When the PM22 bit in the PM2 register is set to 1 (on-chip oscillator selected for the watchdog timer count source), the watchdog timer does not stop even when in wait mode or in a hold state.

Figure 1 shows the Watchdog Timer Operation.

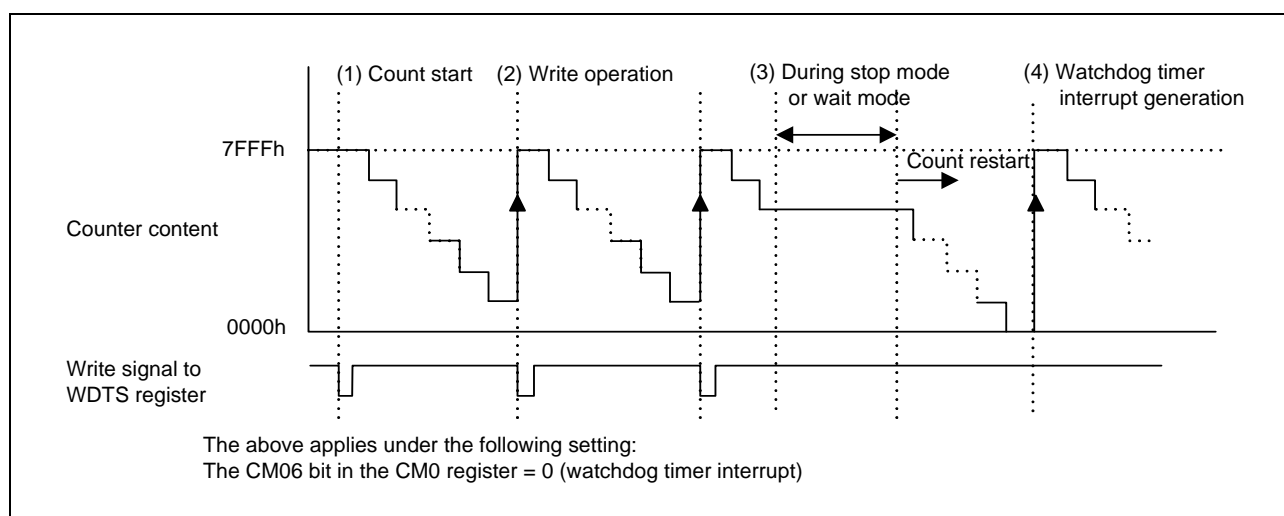


Figure 1 Watchdog Timer Operation

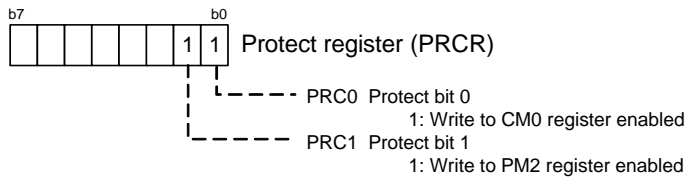
3.2 Setup

This section shows the setup sequence and values to perform the application example described in

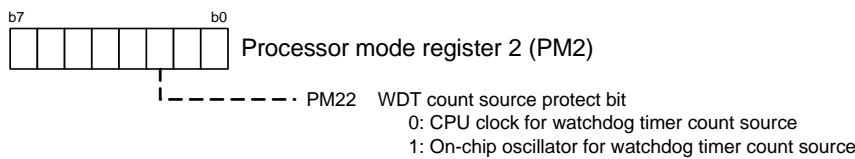
3.1 Example Description.

Refer to the each MCUs Hardware Manual for details of individual registers.

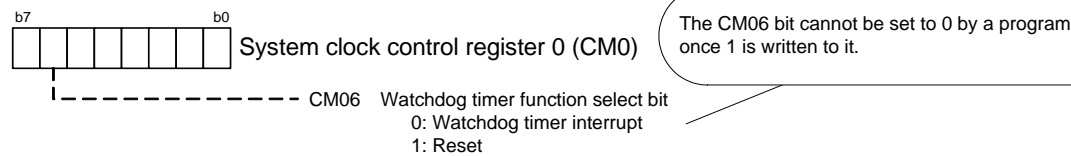
(1) Set the protect register



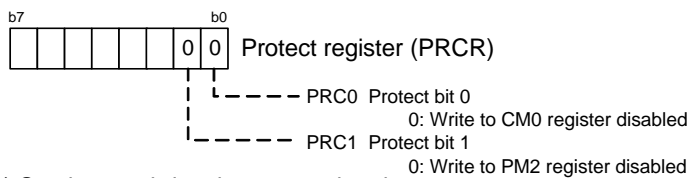
(2) Set processor mode register 2



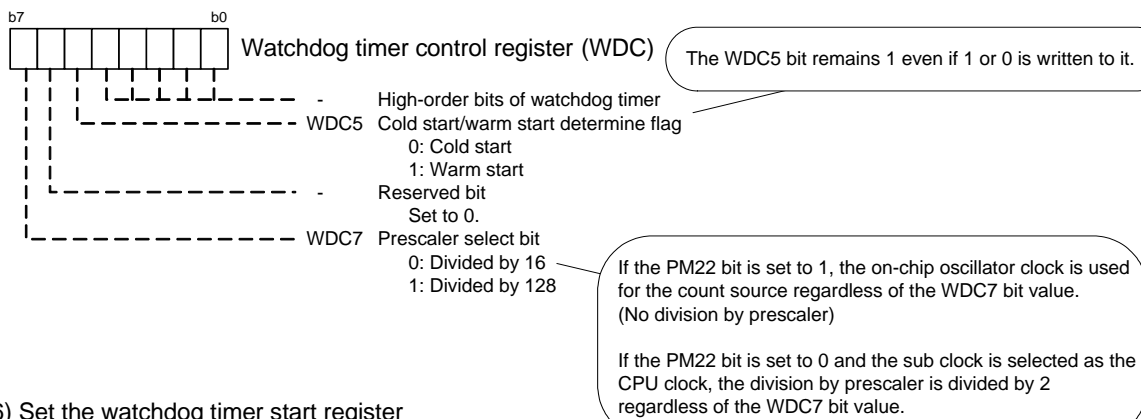
(3) Set system clock control register 0



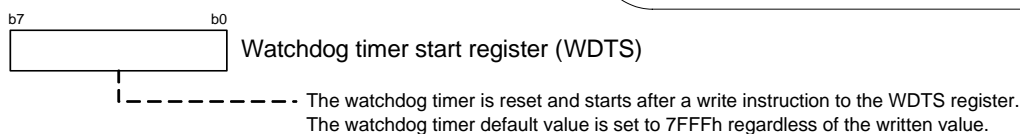
(4) Set the protect register



(5) Set the watchdog timer control register



(6) Set the watchdog timer start register



4. Sample Programming Code

4.1 To Use Watchdog Timer Interrupt

Increment the P10 port indication by writing to the WDTS register. When the P10 port output is set to 40h, finish writing to the WDTS register to stop updating the P10 port indication.

If a watchdog timer underflow occurs, increment the P10 port indication in the watchdog timer interrupt handler by writing to the WDTS register. When the P10 port output is set to 00h, stop updating the P10 port indication.

```

/*****
 *
 * FILE NAME : interrupt_src.c
 * CPU      : M32C/84 Group
 * Function  : Operation of Watchdog Timer
 *
 * Version   : 1.00 (2006-05-18) Initial
 *
 * Copyright (C)2006, Renesas Technology Corp.
 *
 *****/
/*****
 * include file
 *****/
#include "sfr32c84.h"

unsigned int i;
/*****
 * main
 *****/
void main(void) {

    p10 = 0x00;
    pd10 = 0xff;

    ta0mr = 0x40;          /* Select timer mode
                           Gate function select bit (00: Gate function not available)
                           Count source (01:f8)
                           */

    ta0 = 4000-1;         /* Set counter value (1 msec @ 32 MHz, f8)
                           */

    ta0ic = 0x00;         /* Set interrupt priority levels in timer A0
                           */

    ta0s = 1;             /* TimerA0 count start
                           */

    wdc = 0x00;           /* Prescaler Select Bit (0: Divided by 16)
                           (WDT cycle approx. 16.384 ms Xin = @32 MHz)
                           */

    i = 0;
    prcr = 0x03;
    pm2 = 0x00;           /* WDT Count Source Protect Bit (0: Selects CPU clock as count source of the
                           watchdog timer)
                           */

    cm0 = 0x08;          /* Watchdog Timer Function Select Bit (0: Watchdog timer interrupt)
                           */

    prcr = 0x00;

    wdts = 0xFF;         /* The watchdog timer is initialized and starts counting
                           with a write instruction to this register.
                           The watchdog timer value is always initialized to "7FFFh"
                           regardless of the value written.
                           */

    while (1) {
        while(ir_ta0ic == 0){};
        ta0ic = 0x00;
        i++;
        if(i == 500){
            i = 0;
            p10++;
        }
        if(p10 >= 64){
            p10 = 64;
        }
    }
}

```

```

        }else{
            wdts = 0xFF;          /* Set watchdog timer start register
                                   */
        }
    }
}

#pragma interrupt _watchdog_int
void watchdog_int(){
    wdts = 1;
    while(1){
        while(ir_ta0ic == 0){};
        ta0ic = 0x00;
        i++;

        wdts = 0xFF;          /* Set watchdog timer start register
                                   */

        if(i == 500){
            i = 0;
            if(p10 != 0){
                p10--;
            }
        }
    }
}

```

4.2 To Reset on Watchdog Timer Underflow

Increment the P10 port indication by writing to the WDTS register. When the P10 port output is set to 40h, finish writing to the WDTS register to stop updating the P10 port indication. If a watchdog timer underflow occurs, the MCU is reset.

```

/*****
 *
 * FILE NAME : reset_src.c
 * CPU      : M32C/84 Group
 * Function  : Operation of Watchdog Timer
 *
 * Version  : 1.00 (2006-05-18) Initial
 *
 * Copyright (C)2006, Renesas Technology Corp.
 *
 *****/
/*****
 * include file
 *****/
#include "sfr32c84.h"

unsigned int i;
/*****
 * main
 *****/
void main(void) {

    p10 = 0x00;
    pd10 = 0xff;

    ta0mr = 0x40;          /* Select timer mode
                           Gate function select bit (00: Gate function not available)
                           Count source (01:f8)
                           */

    ta0 = 4000-1;         /* Set counter value (1 ms @ 32 MHz, f8)
                           */

    ta0ic = 0x00;         /* Set interrupt priority levels in timer A0
                           */

    ta0s = 1;             /* TimerA0 count start
                           */

    wdc = 0xFF;          /* Prescaler Select Bit (0: Divided by 16)
                           (WDT cycle approx. 16.384ms Xin = @ 32 MHz)
                           */

    i = 0;
    prcr = 0x03;
    pm2 = 0x00;          /* WDT Count Source Protect Bit (0: Selects CPU clock as count source of the
                           watchdog timer)
    
```

```

        */
cm0 = 0x48;          /* Watchdog Timer Function Select Bit (1: Reset)
                    */
prcr = 0x00;

wdts = 0x00;        /* The watchdog timer is initialized and starts counting
                    with a write instruction to this register.
                    The watchdog timer value is always initialized to "7FFFh"
                    regardless of the value written.
                    */

while (1) {
    while(ir_ta0ic == 0){};
    ta0ic = 0x00;
    i++;
    if(i == 500){
        i = 0;
        p10++;
    }
    if(p10 >= 64){
        p10 = 64;
    }else{
        wdts = 0xFF;          /* Set watchdog timer start register
                              */
    }
}

#pragma interrupt _watchdog_int
void watchdog_int(){
    wdts = 1;
    while(1){
        while(ir_ta0ic == 0){};
        ta0ic = 0x00;
        i++;

        wdts = 0xFF;        /* Set watchdog timer start register
                              */

        if(i == 500){
            i = 0;
            if(p10 != 0){
                p10--;
            }
        }
    }
}

```

5. Reference Documents

Hardware Manuals

M32C/84 Group Hardware Manual

M32C/85 Group Hardware Manual

M32C/86 Group Hardware Manual

M32C/87 Group Hardware Manual

M32C/88 Group Hardware Manual

The latest version can be downloaded from the Renesas Technology website.

Technical News/Technical Update

The latest information can be downloaded from the Renesas Technology website.

Website and Support

Renesas Technology website
<http://www.renesas.com/>

Inquiries
<http://www.renesas.com/inquiry>
csc@renesas.com

| | |
|------------------|-------------------------------------------------|
| REVISION HISTORY | M32C/84, 85, 86, 87, 88 Group Watchdog Timer |
|------------------|-------------------------------------------------|

| Rev. | Date | Description | |
|------|--------------|-------------|----------------------|
| | | Page | Summary |
| 1.00 | Sep 10, 2006 | – | First Edition issued |
| | | | |

Notes regarding these materials

1. This document is provided for reference purposes only so that Renesas customers may select the appropriate Renesas products for their use. Renesas neither makes warranties or representations with respect to the accuracy or completeness of the information contained in this document nor grants any license to any intellectual property rights or any other rights of Renesas or any third party with respect to the information in this document.
2. Renesas shall have no liability for damages or infringement of any intellectual property or other rights arising out of the use of any information in this document, including, but not limited to, product data, diagrams, charts, programs, algorithms, and application circuit examples.
3. You should not use the products or the technology described in this document for the purpose of military applications such as the development of weapons of mass destruction or for the purpose of any other military use. When exporting the products or technology described herein, you should follow the applicable export control laws and regulations, and procedures required by such laws and regulations.
4. All information included in this document such as product data, diagrams, charts, programs, algorithms, and application circuit examples, is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas products listed in this document, please confirm the latest product information with a Renesas sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas such as that disclosed through our website. (<http://www.renesas.com>)
5. Renesas has used reasonable care in compiling the information included in this document, but Renesas assumes no liability whatsoever for any damages incurred as a result of errors or omissions in the information included in this document.
6. When using or otherwise relying on the information in this document, you should evaluate the information in light of the total system before deciding about the applicability of such information to the intended application. Renesas makes no representations, warranties or guaranties regarding the suitability of its products for any particular application and specifically disclaims any liability arising out of the application and use of the information in this document or Renesas products.
7. With the exception of products specified by Renesas as suitable for automobile applications, Renesas products are not designed, manufactured or tested for applications or otherwise in systems the failure or malfunction of which may cause a direct threat to human life or create a risk of human injury or which require especially high quality and reliability such as safety systems, or equipment or systems for transportation and traffic, healthcare, combustion control, aerospace and aeronautics, nuclear power, or undersea communication transmission. If you are considering the use of our products for such purposes, please contact a Renesas sales office beforehand. Renesas shall have no liability for damages arising out of the uses set forth above.
8. Notwithstanding the preceding paragraph, you should not use Renesas products for the purposes listed below:
 - (1) artificial life support devices or systems
 - (2) surgical implantations
 - (3) healthcare intervention (e.g., excision, administration of medication, etc.)
 - (4) any other purposes that pose a direct threat to human life

Renesas shall have no liability for damages arising out of the uses set forth in the above and purchasers who elect to use Renesas products in any of the foregoing applications shall indemnify and hold harmless Renesas Technology Corp., its affiliated companies and their officers, directors, and employees against any and all damages arising out of such applications.
9. You should use the products described herein within the range specified by Renesas, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas shall have no liability for malfunctions or damages arising out of the use of Renesas products beyond such specified ranges.
10. Although Renesas endeavors to improve the quality and reliability of its products, IC products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Please be sure to implement safety measures to guard against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other applicable measures. Among others, since the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
11. In case Renesas products listed in this document are detached from the products to which the Renesas products are attached or affixed, the risk of accident such as swallowing by infants and small children is very high. You should implement safety measures so that Renesas products may not be easily detached from your products. Renesas shall have no liability for damages arising out of such detachment.
12. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written approval from Renesas.
13. Please contact a Renesas sales office if you have any questions regarding the information contained in this document, Renesas semiconductor products, or if you have any other inquiries.