Introduction:

This demo gives an idea about using the on-chip watchdog timer in the real time applications. Watchdog timer is one of the on-chip safety mechanisms to keep the workings of the CPU in a stable condition when the microcontroller is put into uncertain and noisy environments. When the CPU misses its defined flow of control, then the watchdog timer may reset the microcontroller and bring back the CPU into its original operating conditions. To get this implemented, the application should have a facility embedded along the course of control flow to refresh the watchdog timer before its predefined time gets lapsed.

When the watchdog timer misses any of its periodical refreshing signal, it completes its timing period and pushes the microcontroller into reset to force the CPU to start working from reset vector. This is how the microcontroller comes out of uncertain conditions. Because of the inclusion of the watchdog timer into the design, the reliability of the application is increased so much.

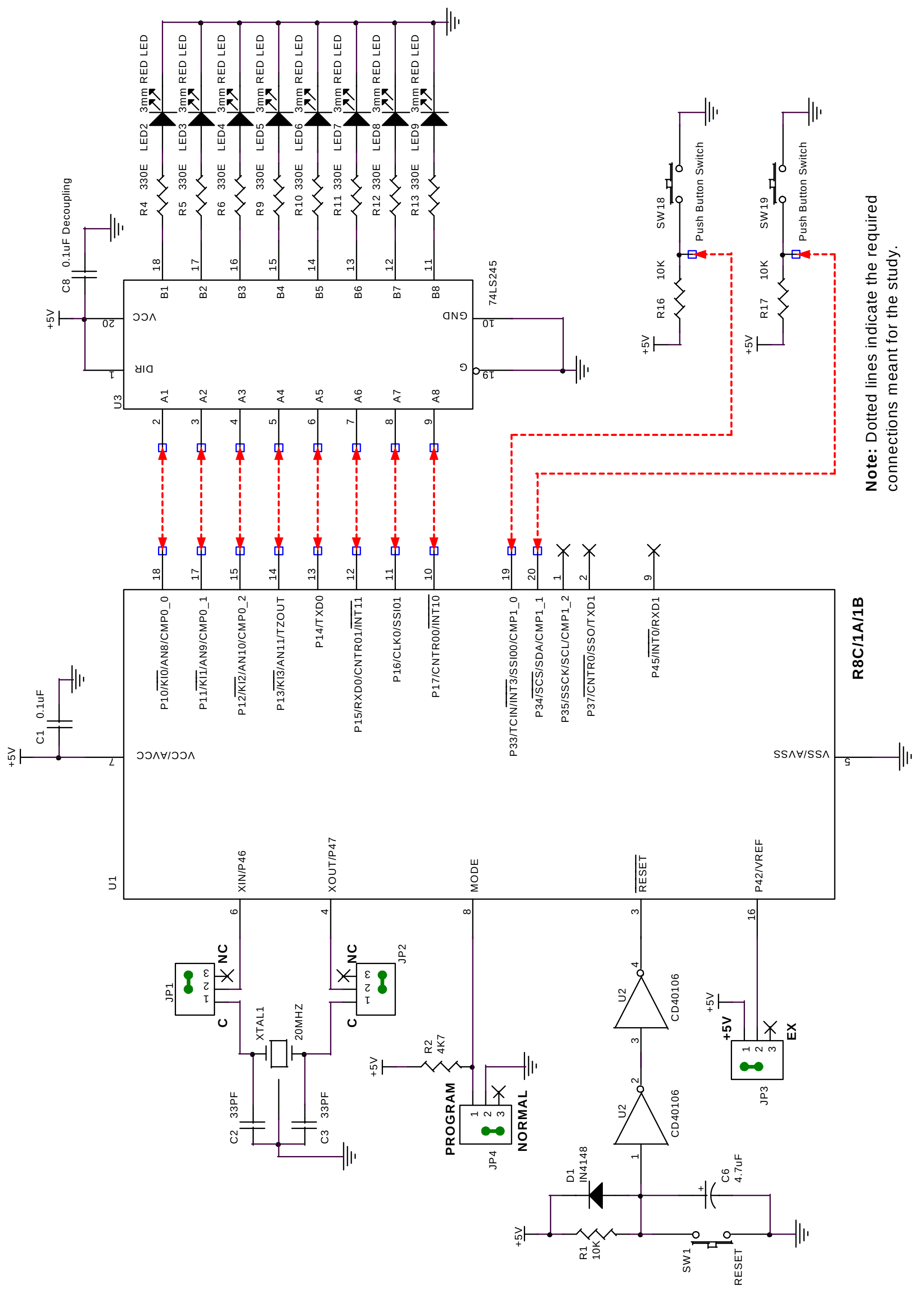
The R8C Tiny controllers come with an easy to use watchdog timer along with a simplified refresh mechanism to enable the designers include the required protection into their designs.

The study module makes use of point LEDs and push button switches to make the study on watchdog timer an easy and interesting one!

Hardware:

Eight point LEDs and two push button switches are used to study the on-chip watchdog timer. The LEDs are connected to the port lines P10 to P17 with proper drivers. The TTL buffer IC 74LS245 is used to drive the point LEDs. The two push button switches are connected to the port lines P33 and P34 for interacting with the micon.

Circuit:



Note: Dotted lines indicate the required connections meant for the study.

Connections:

1. Connect the port lines P10 to P17 to 8 point LEDs.
2. Connect port lines P33 and P34 to two push button switches.

Functional Description:

The R8C/Tiny‘s watchdog timer is designed around a 15bit counter, which counts down from 32768/4096 using a variety of clock options. The count value can be either 32768 or 4096 and selected by the CSPRO bit in Count Source Protection Mode Register CSPR. The watchdog timer can be operated in two modes with either count source protection mode enabled or disabled.

When this counter under flows (When the count reaches down to zero), the watchdog mechanism generates either an interrupt request or a system reset to the microcontroller. The watchdog timer gets its clock source from the CPU clock through a prescaler with either 16 or 128 dividing options. So, users get different time out periods from the CPU clock options as well as selecting the appropriate divider in the prescaler.

The result of watchdog timer’s under flow is selected by defining PM12 bit of the PM1 register. This PM12 bit only can be set to 1 level to activate the system reset. Once this bit is set to 1 level, the bit can not be reset to 0 level by the program control. When this bit is properly set to 0 level, then the underflow of the watchdog timer generates an interrupt request.

When the watchdog timer generates a reset signal, the microcontroller initializes its pins, CPU and the most of the SFR. Then the CPU starts executing programs indicated by the reset vector. After the reset, the CPU gets the low speed on-chip oscillator clock with the division of 8.

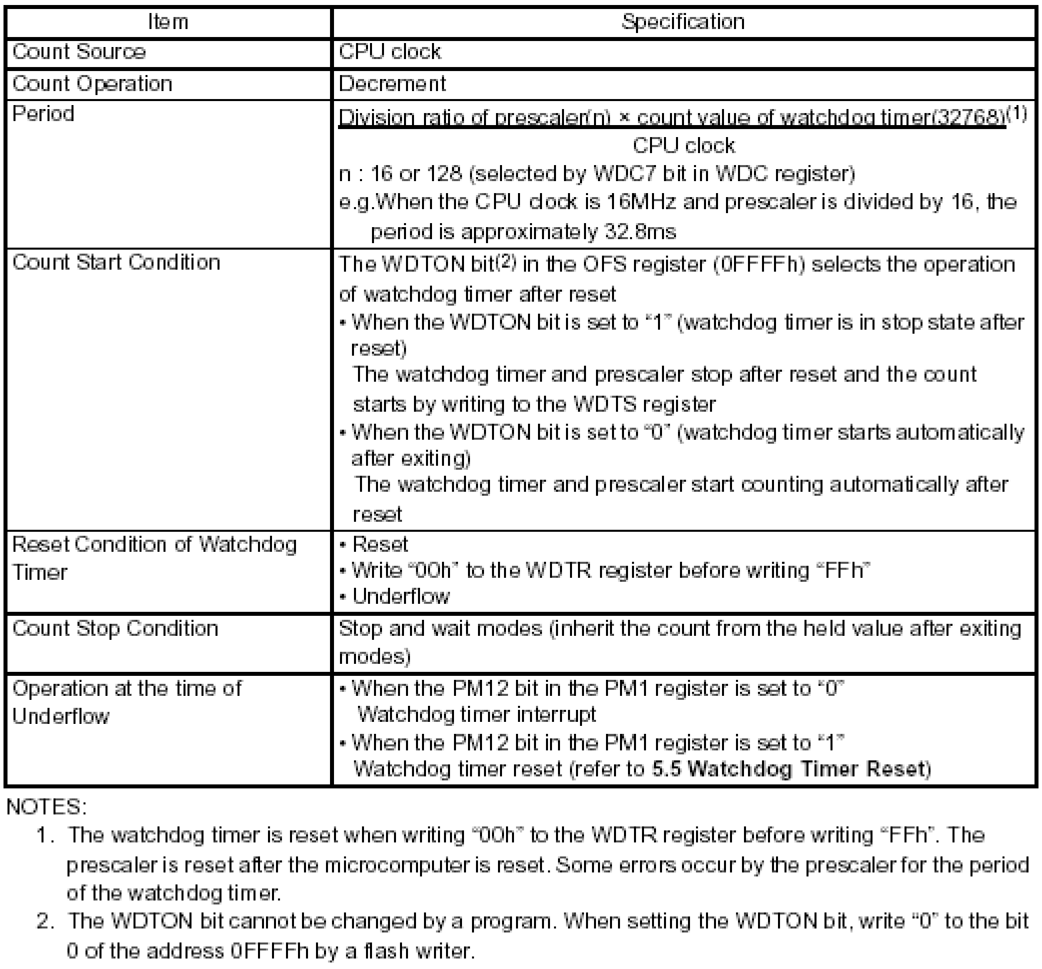
When the count source protection mode is in enabled condition, the operation and other details are tabulated as shown below:

Item	Specification
Count Source	Low-speed on-chip oscillator clock
Count Operation	Decrement
Period	$\frac{\text{Count value of watchdog timer (4096)}}{\text{Low-speed on-chip oscillator clock}}$ e.g. Period is approximately 32.8ms when the low-speed on-chip oscillator clock is 125 kHz
Count Start Condition	The WDTON bit[1] in the OFS register (0FFFFh) selects the operation of the watchdog timer after reset. • When the WDTON bit is set to “1” (watchdog timer is in stop state after reset) The watchdog timer and prescaler stop after reset and the count starts by writing to the WDTS register • When the WDTON bit is set to “0” (watchdog timer starts automatically after reset) The watchdog timer and prescaler start counting automatically after reset
Reset Condition of Watchdog Timer	• Reset • Write “00h” to the WDTR register before writing “FFh” • Underflow
Count Stop Condition	None (the count does not stop in wait mode after the count starts. The microcomputer does not enter stop mode)
Operation at the time of Underflow	Watchdog timer reset (refer to **5.5 Watchdog Timer Reset**)
Register, Bit	• When setting the CSPPRO bit in the CSPR register to “1” (count source protection mode is enabled)[2], the following are set automatically - Set 0FFFh to the watchdog timer - Set the CM14 bit in the CM1 register to “0” (low-speed on-chip oscillator on) - Set the PM12 bit in the PM1 register to “1” (The watchdog timer is reset when watchdog timer underflows) • The following states are held in count source protection mode - Writing to the CM10 bit in the CM1 register disables (It remains unchanged even if it is set to “1”. The microcomputer does not enter stop mode) - Writing to the CM14 bit in the CM1 register disables (It remains unchanged even if it is set to “1”. The low-speed on-chip oscillator does not stop)

NOTES:

1. The WDTON bit cannot be changed by a program. When setting the WDTON bit, write “0” to the bit 0 of the address 0FFFFh by a flash writer.
2. Even if writing “0” to the CSPROINI bit in the OFS register, the CSPRO bit is set to “1”. The CSPROINI bit cannot be changed by a program. When setting the CSPROINI bit, write “0” to the bit 7 of the address 0FFFFh by a flash writer.

When the count source protection mode is in enabled condition, the operation and other details are tabulated as shown below:



Item	Specification
Count Source	CPU clock
Count Operation	Decrement
Period	$\dfrac{\text{Division ratio of prescaler(n)} \times \text{count value of watchdog timer(32768)}^{(1)}}{\text{CPU clock}}$ n : 16 or 128 (selected by WDC7 bit in WDC register) e.g.When the CPU clock is 16MHz and prescaler is divided by 16, the period is approximately 32.8ms
Count Start Condition	The WDTON bit(2) in the OFS register (0FFFFh) selects the operation of watchdog timer after reset • When the WDTON bit is set to "1" (watchdog timer is in stop state after reset) The watchdog timer and prescaler stop after reset and the count starts by writing to the WDTS register • When the WDTON bit is set to "0" (watchdog timer starts automatically after exiting) The watchdog timer and prescaler start counting automatically after reset
Reset Condition of Watchdog Timer	• Reset • Write "00h" to the WDTR register before writing "FFh" • Underflow
Count Stop Condition	Stop and wait modes (inherit the count from the held value after exiting modes)
Operation at the time of Underflow	• When the PM12 bit in the PM1 register is set to "0" Watchdog timer interrupt • When the PM12 bit in the PM1 register is set to "1" Watchdog timer reset (refer to **5.5 Watchdog Timer Reset**)

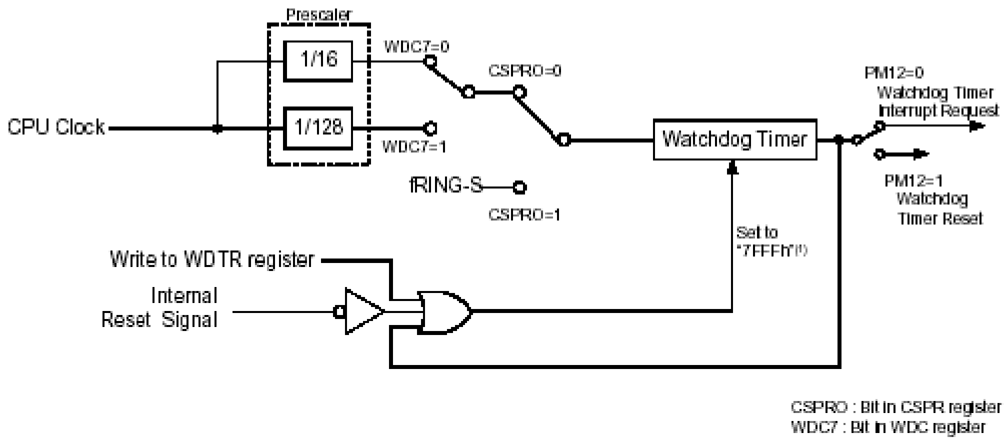
NOTES:
1. The watchdog timer is reset when writing "00h" to the WDTR register before writing "FFh". The prescaler is reset after the microcomputer is reset. Some errors occur by the prescaler for the period of the watchdog timer.
2. The WDTON bit cannot be changed by a program. When setting the WDTON bit, write "0" to the bit 0 of the address 0FFFFh by a flash writer.

The prescaler's divider is selected between 16 and 128 by defining WDC7 bit of the WDC register.

Both the watchdog timer and the prescaler becomes inactive immediately after reset. So, the watchdog timer has to be given the command to start working again. For a write operation to the WDTS (Watch Dog Timer Start) register, the watchdog timer starts counting down for every clock input signal. The source clock can be either the CPU clock or low speed ring oscillator clock according to the bit CSPRO bit value. To initialize the watchdog timer to its fixed defined value of 32,768 or 4096, another write operation is required for the WDTR (Watch Dog Timer Reset) register. From the next clock onwards, the counting continues downwards. Both these opera-

tions just need a simple writing to the registers with minimum program code.
In both stop and wait modes, the watchdog timer and the prescaler are stopped. When the device comes out of these modes, the watchdog timer resumes its operation from the stopped value.

The block diagram of the watchdog timer shown below:

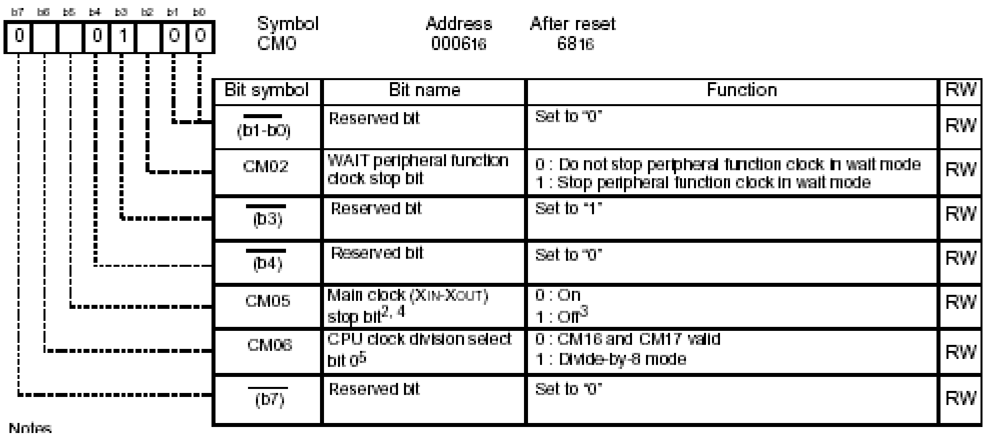


NOTES:
1. When the CSPRO bit is set to "1" (count source protection mode enabled), "0FFFh" is set.

Registers Used:

CM0	- System clock control register 0
CM1	- System clock control register 1
OCD	- Oscillation stop detection register 1
WDTS	- Watchdog timer start register
WDTR	- Watchdog timer reset register

CM0 - System Clock Control Register 0:



b7	b6	b5	b4	b3	b2	b1	b0
0			0	1		0	0

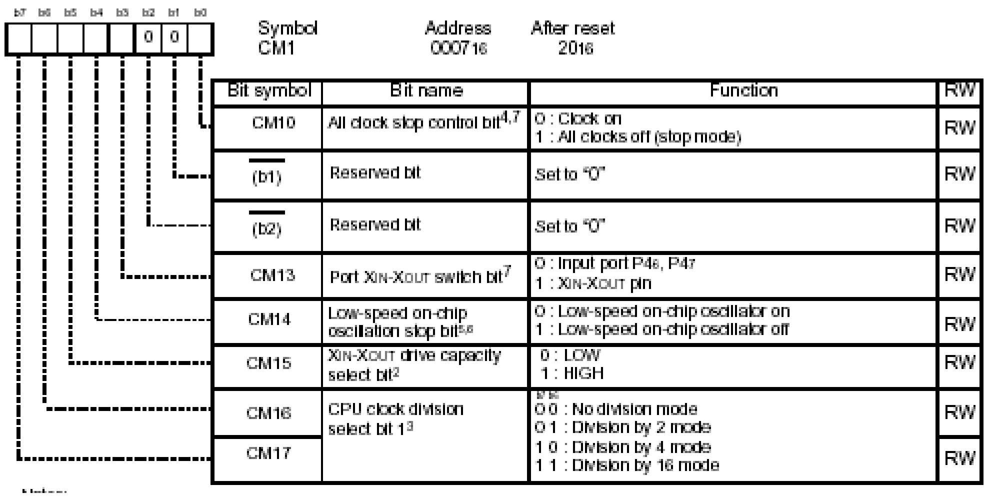
Symbol CM0 — Address 0006_{16} — After reset 68_{16}

Bit symbol	Bit name	Function	RW
$\overline{}$ (b1-b0)	Reserved bit	Set to "0"	RW
CM02	WAIT peripheral function clock stop bit	0 : Do not stop peripheral function clock in wait mode 1 : Stop peripheral function clock in wait mode	RW
$\overline{}$ (b3)	Reserved bit	Set to "1"	RW
$\overline{}$ (b4)	Reserved bit	Set to "0"	RW
CM05	Main clock (XIN-XOUT) stop bit[2, 4]	0 : On 1 : Off[3]	RW
CM06	CPU clock division select bit 0[5]	0 : CM16 and CM17 valid 1 : Divide-by-8 mode	RW
$\overline{}$ (b7)	Reserved bit	Set to "0"	RW

Notes

The main clock is switched off by setting CM05 bit. Allow division of clock according to the CM16 and CM17 bits contents by clearing CM06 bit.

CM1 - System Clock Control Register 1:



b7	b6	b5	b4	b3	b2	b1	b0
					0	0	

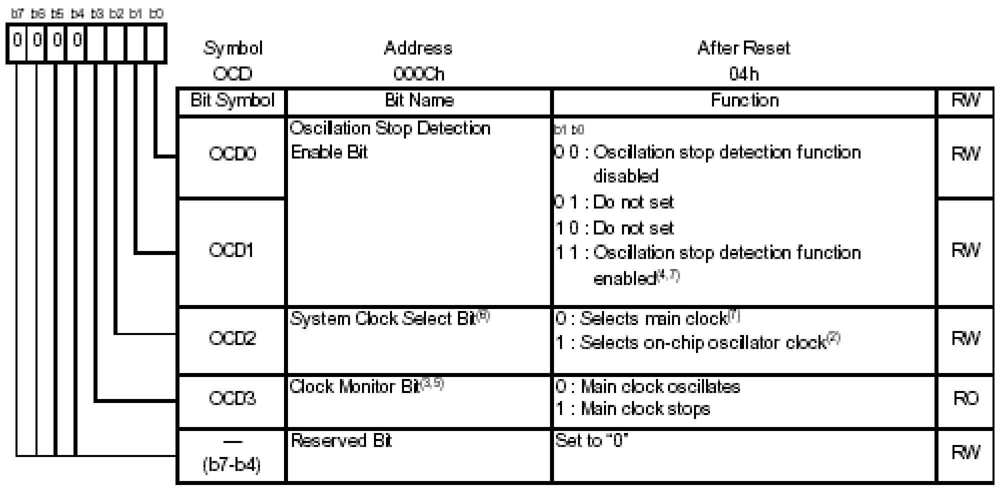
Symbol CM1 — Address 0007_{16} — After reset 20_{16}

Bit symbol	Bit name	Function	RW
CM10	All clock stop control bit[4,7]	0 : Clock on 1 : All clocks off (stop mode)	RW
$\overline{}$ (b1)	Reserved bit	Set to "0"	RW
$\overline{}$ (b2)	Reserved bit	Set to "0"	RW
CM13	Port XIN-XOUT switch bit[7]	0 : Input port $P4_6$, $P4_7$ 1 : XIN-XOUT pin	RW
CM14	Low-speed on-chip oscillation stop bit[5,6]	0 : Low-speed on-chip oscillator on 1 : Low-speed on-chip oscillator off	RW
CM15	XIN-XOUT drive capacity select bit[2]	0 : LOW 1 : HIGH	RW
CM16	CPU clock division select bit 1[3]	b7 b6 0 0 : No division mode 0 1 : Division by 2 mode 1 0 : Division by 4 mode 1 1 : Division by 16 mode	RW
CM17			RW

Notes:

Turn on low speed on-chip oscillator by clearing CM14 bit.

OCD - Oscillation Stop Detection Register 1:

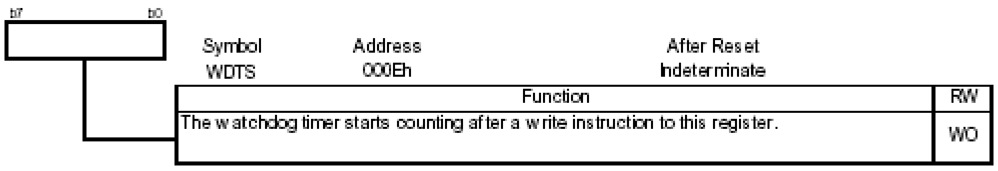

b7 b6 b5 b4 b3 b2 b1 b0
0 0 0 0

Symbol	Address	After Reset
OCD	000Ch	04h

Bit Symbol	Bit Name	Function	RW
OCD0	Oscillation Stop Detection Enable Bit	b1 b0 0 0 : Oscillation stop detection function disabled 0 1 : Do not set 1 0 : Do not set 1 1 : Oscillation stop detection function enabled(4,7)	RW
OCD1			RW
OCD2	System Clock Select Bit(6)	0 : Selects main clock(7) 1 : Selects on-chip oscillator clock(2)	RW
OCD3	Clock Monitor Bit(3,5)	0 : Main clock oscillates 1 : Main clock stops	RO
— (b7-b4)	Reserved Bit	Set to "0"	RW

Main clock is switched off by setting the bit OCD3 and the on-chip oscillator is selected by setting OCD2 bit.

WDTS - Watchdog Timer Start Register

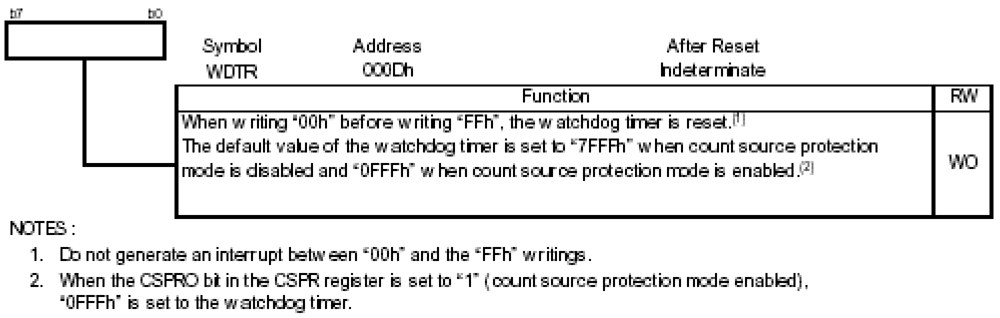


Symbol	Address	After Reset
WDTS	000Eh	Indeterminate

Function	RW
The watchdog timer starts counting after a write instruction to this register.	WO

Write a dummy data H'FF to this register to start watchdog timer.

WDTR - Watchdog Timer Reset Register



Symbol	Address	After Reset
WDTR	000Dh	Indeterminate

Function	RW
When writing "00h" before writing "FFh", the watchdog timer is reset.(1) The default value of the watchdog timer is set to "7FFFh" when count source protection mode is disabled and "0FFFh" when count source protection mode is enabled.(2)	WO

NOTES :
1. Do not generate an interrupt between "00h" and the "FFh" writings.
2. When the CSPRO bit in the CSPR register is set to "1" (count source protection mode enabled), "0FFFh" is set to the watchdog timer.

Write a dummy data H'FF to this register to reset the watchdog timer.

Software Description:

To implement this study, eight point LEDs are connected to the port 1 and two push button switches are connected with port lines P33 and P34.

After the reset, the microcontroller starts sequentially flashing the point LEDs from first to last with a time delay and then keep flashing the same LEDs from last to first one with the delay. This continuous sequential LED flashing indicates a simple control flow without any watchdog timer. But along the course of program flow, the control looks for a press of push button switch connected with the port line P33.

When the program flow identifies a key press at P33, the microcontroller immediately stops sequential LED flashing and starts the watchdog timer. An LED is triggered to flash for indicating watchdog operation. Apart from this LED flashing, the program control looks for the key press at the pin P34.

Now, on sensing the second key press, the program stops the watchdog timer's refresh to let the timer gets its time lapsed out. Since, the watchdog timer's output is enabled to reset the microcontroller, the flashing of the point LED continues till the end of watchdog timer's time out period and then starts flashing all the LEDs in sequence when the controller comes out of the reset state.

In this study the low speed on-chip oscillator is used as clock source for CPU and other peripherals. The divide by 16 option is selected for the prescaler and the time out period of the watchdog timer will be 4.194 seconds.

4.194 seconds = (16 * 32768)/125KHZ.

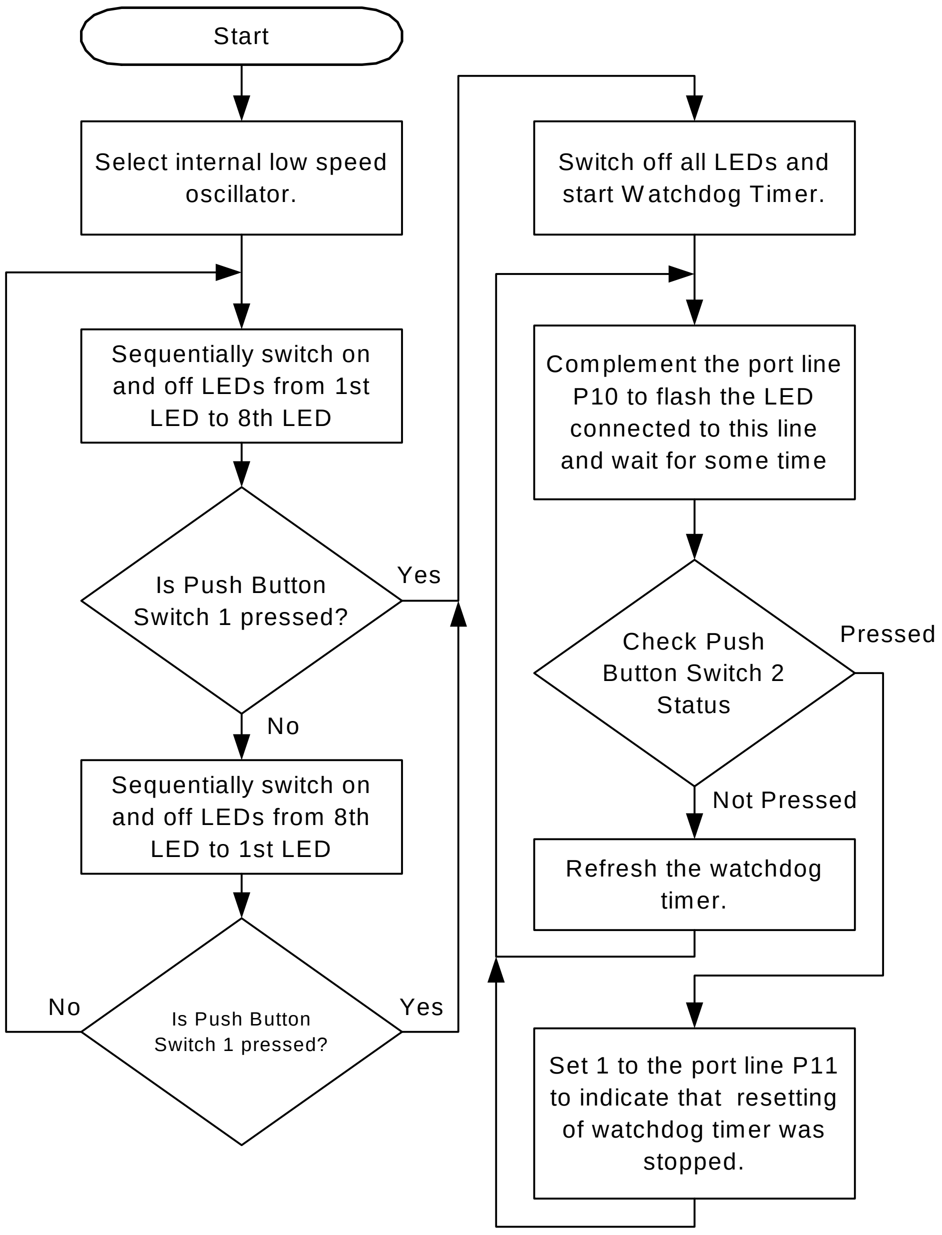
The files used in this module are listed below:

Files	*Description*
Demo13.C	Main file for this module, will gives an idea about using the watchdog timer in a real applications with 8 point LEDs and two push button switches

The functions in the file “**Demo13.C**” and short descriptions are listed below:

Functions	***Description***
main	This is the main function of this module and will sequentially swicthes 8 LEDs from 1st LED to 8th LED and from 8th LED to 1st LED with time interval. Starts the watchdog timer when user presses first key and stops resetting watchdog timer after the second switch is pressed by the user. **Input**: None. **Output** : None.
MCUInitialize_Int_Low	Selects the on-chip low speed oscillator as clock source for the CPU and other peripherals. **Input**: None. **Output** : None.
InitializeIO	Initializes the port lines P10 to P17 as output lines to drive LEDs. **Input**: None. **Output** : None.

Program Flow:


Start
Select internal low speed oscillator.
Sequentially switch on and off LEDs from 1st LED to 8th LED
Is Push Button Switch 1 pressed?
Yes
No
Sequentially switch on and off LEDs from 8th LED to 1st LED
No
Is Push Button Switch 1 pressed?
Yes
Switch off all LEDs and start Watchdog Timer.
Complement the port line P10 to flash the LED connected to this line and wait for some time
Check Push Button Switch 2 Status
Pressed
Not Pressed
Refresh the watchdog timer.
Set 1 to the port line P11 to indicate that resetting of watchdog timer was stopped.

Execute Demo:

The program will sequentially switch 8 LEDs from first one to the last and then start from the last LED to the first one with time interval. Now press push button switch 1 (Connected to P33) to start the watchdog timer.

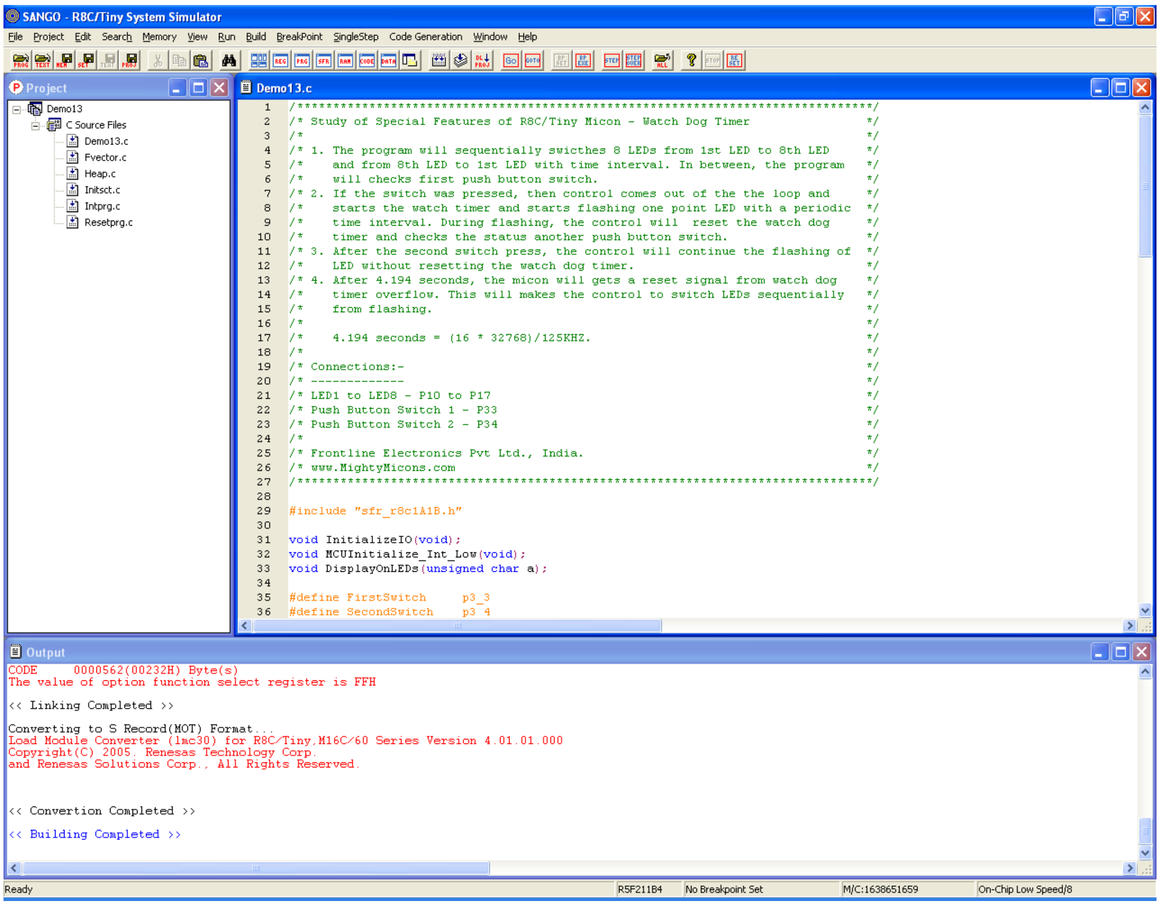
The control comes out of the loop and enables the watchdog timer and starts flashing one point LED with a periodic time interval. During flashing, the control will keep refreshing the watch dog timer. To stop the refreshing of watchdog timer, press the push button switch 2 (connected to P34). The LED connected to the port line P11 will now stay flashing on till the watchdog's period gets lapsed. Here the time out period is 4.194 seconds.

When the time out period lapsed, the watchdog timer generates a reset to the device and this makes the CPU start operating from the reset vector and then starts displaying all the LEDs in sequence one by one.

Use Topview Simulator to Verify the Design.

Open the project Demo13 in the R8C/Tiny System Simulator using **Open Project** option from **Project menu**. The project window opens up along with the Demo13.c file. Use **Build** option from **Build** menu to compile the project. An output window captures the compiler ouput.

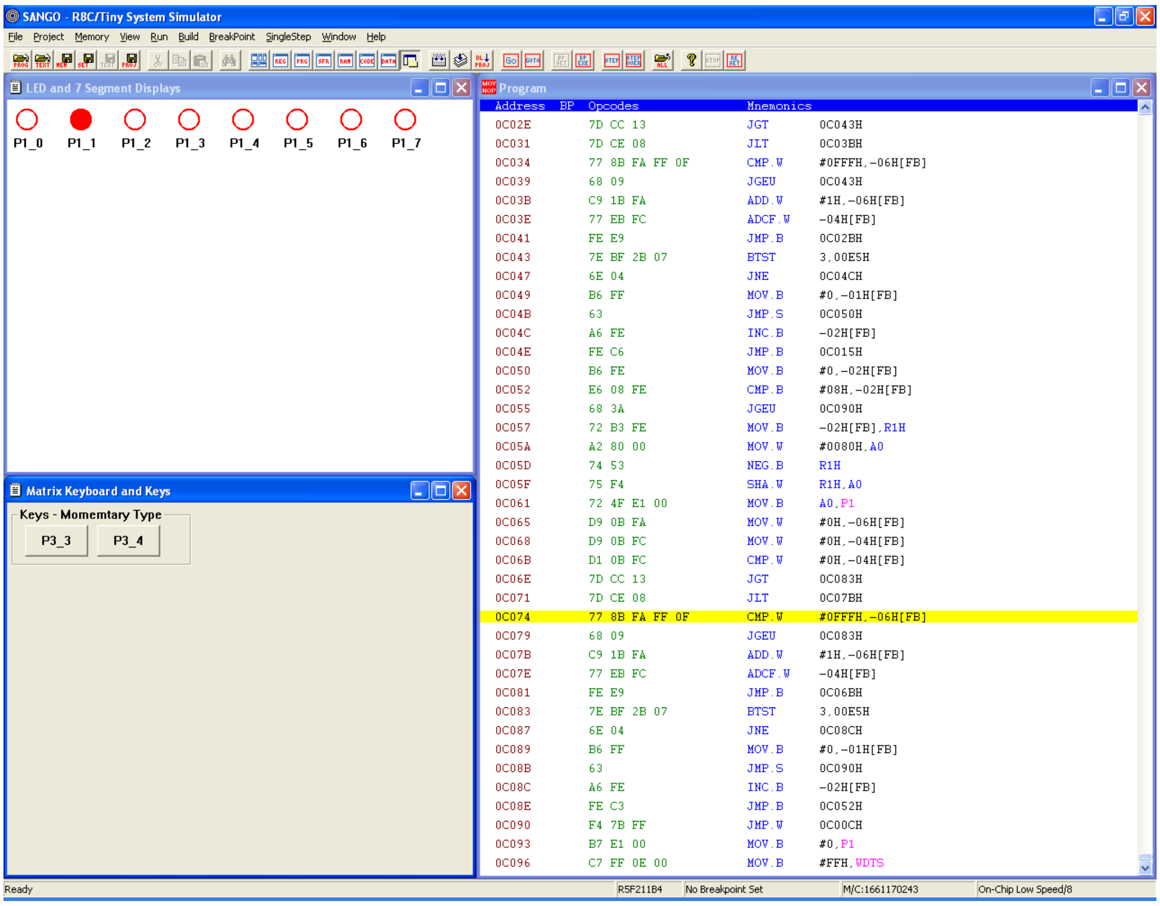
Use **Project** -> **Download Project** from main menu to download the .mot file into the simulator's memory for simulation.



```c
/*****************************************************************************/
/* Study of Special Features of R8C/Tiny Micon - Watch Dog Timer            */
/*                                                                           */
/* 1. The program will sequentially swicthes 8 LEDs from 1st LED to 8th LED */
/*    and from 8th LED to 1st LED with time interval. In between, the program */
/*    will checks first push button switch.                                  */
/* 2. If the switch was pressed, then control comes out of the the loop and */
/*    starts the watch timer and starts flashing one point LED with a periodic */
/*    time interval. During flashing, the control will  reset the watch dog */
/*    timer and checks the status another push button switch.               */
/* 3. After the second switch press, the control will continue the flashing of */
/*    LED without resetting the watch dog timer.                             */
/* 4. After 4.194 seconds, the micon will gets a reset signal from watch dog */
/*    timer overflow. This will makes the control to switch LEDs sequentially */
/*    from flashing.                                                         */
/*                                                                           */
/*    4.194 seconds = (16 * 32768)/125KHZ.                                   */
/*                                                                           */
/* Connections:-                                                             */
/* -------------                                                             */
/* LED1 to LED8 - P10 to P17                                                 */
/* Push Button Switch 1 - P33                                                */
/* Push Button Switch 2 - P34                                                */
/*                                                                           */
/* Frontline Electronics Pvt Ltd., India.                                    */
/* www.MightyMicons.com                                                      */
/*****************************************************************************/

#include "sfr_r8c1A1B.h"

void InitializeIO(void);
void MCUInitialize_Int_Low(void);
void DisplayOnLEDs(unsigned char a);

#define FirstSwitch     p3_3
#define SecondSwitch    p3_4
```

```
CODE     0000562(00232H) Byte(s)
The value of option function select register is FFH

<< Linking Completed >>

Converting to S Record(MOT) Format...
Load Module Converter (lmc30) for R8C/Tiny,M16C/60 Series Version 4.01.01.000
Copyright(C) 2005. Renesas Technology Corp.
and Renesas Solutions Corp., All Rights Reserved.


<< Convertion Completed >>

<< Building Completed >>
```

Connect point LEDs to the port lines P10 to P17 using LED module setting and connect two push button switches to the port lines P33 and P34. Open the LED and Keyboard windows and arrange them as shown for better visibility.


SANGO - R8C/Tiny System Simulator
File Project Memory View Run Build BreakPoint SingleStep Window Help
LED and 7 Segment Displays
P1_0 P1_1 P1_2 P1_3 P1_4 P1_5 P1_6 P1_7
Matrix Keyboard and Keys
Keys - Momemtary Type
P3_3 P3_4
Program
Ready

```
Address  BP  Opcodes          Mnemonics
0C02E        7D CC 13         JGT      0C043H
0C031        7D CE 08         JLT      0C03BH
0C034        77 8B FA FF 0F   CMP.W    #0FFFH,-06H[FB]
0C039        68 09            JGEU     0C043H
0C03B        C9 1B FA         ADD.W    #1H,-06H[FB]
0C03E        77 EB FC         ADCF.W   -04H[FB]
0C041        FE E9            JMP.B    0C02BH
0C043        7E BF 2B 07      BTST     3,00E5H
0C047        6E 04            JNE      0C04CH
0C049        B6 FF            MOV.B    #0,-01H[FB]
0C04B        63               JMP.S    0C050H
0C04C        A6 FE            INC.B    -02H[FB]
0C04E        FE C6            JMP.B    0C015H
0C050        B6 FE            MOV.B    #0,-02H[FB]
0C052        E6 08 FE         CMP.B    #08H,-02H[FB]
0C055        68 3A            JGEU     0C090H
0C057        72 B3 FE         MOV.B    -02H[FB],R1H
0C05A        A2 80 00         MOV.W    #0080H,A0
0C05D        74 53            NEG.B    R1H
0C05F        75 F4            SHA.W    R1H,A0
0C061        72 4F E1 00      MOV.B    A0,P1
0C065        D9 0B FA         MOV.W    #0H,-06H[FB]
0C068        D9 0B FC         MOV.W    #0H,-04H[FB]
0C06B        D1 0B FC         CMP.W    #0H,-04H[FB]
0C06E        7D CC 13         JGT      0C083H
0C071        7D CE 08         JLT      0C07BH
0C074        77 8B FA FF 0F   CMP.W    #0FFFH,-06H[FB]
0C079        68 09            JGEU     0C083H
0C07B        C9 1B FA         ADD.W    #1H,-06H[FB]
0C07E        77 EB FC         ADCF.W   -04H[FB]
0C081        FE E9            JMP.B    0C06BH
0C083        7E BF 2B 07      BTST     3,00E5H
0C087        6E 04            JNE      0C08CH
0C089        B6 FF            MOV.B    #0,-01H[FB]
0C08B        63               JMP.S    0C090H
0C08C        A6 FE            INC.B    -02H[FB]
0C08E        FE C3            JMP.B    0C052H
0C090        F4 7B FF         JMP.W    0C00CH
0C093        B7 E1 00         MOV.B    #0,P1
0C096        C7 FF 0E 00      MOV.B    #FFH,WDTS
```

```
R5F211B4    No Breakpoint Set    M/C:1661170243    On-Chip Low Speed/8
```

Down load the program using **Download Project** command in **Project** menu.

Run the program using **Go** command in **Run** menu.

The program will sequentially swicthes 8 LEDs from 1st LED to 8th LED and from 8th LED to 1st LED with time interval. In between, the program will checks first push button switch.

when the switch was pressed (connected to P33), then control comes out of the the loop and starts the watch timer and starts flashing one point LED with a periodic time interval. During flashing, the control will reset the watch dog timer and checks the status another push button switch.

After the second switch press (P34), the control will continue the flashing of LED without resetting the watch dog timer. After 4.194 seconds, the micon will gets a reset signal from watch dog timer overflow. This will makes the control to switch LEDs sequentially from flashing.